University of East London

School of Architecture, Computing and Engineering

Predictive modelling of retail banking transactions for credit scoring, cross-selling and payment pattern discovery

by

Hakim Harrach

A thesis submitted to the University of East London in fulfilment of the requirements for the degree of Professional Doctorate in Data Science.

September 2019

Declaration

I, Mr. Hakim Harrach, hereby certify that the work presented in this thesis is the result of my own investigation except where references has been made to published literatures and where acknowledgement is made for unpublished data. During the course of this research programme, I have not been registered or enrolled for another award from any academic or professional institutions.

Student: Mr Hakim Harrach

Student Number: 1159133

Supervisory Team: Professor Allan J Brimicombe
Dr. Yang Li

School of Architecture Computing and Engineering
Centre for Geo-Information
University of East London
September 2019

Figure 0-1: Front piece - word cloud of the doctoral thesis

Abstract

Evaluating transactional payment behaviour offers a competitive advantage in the modern payment ecosystem, not only for confirming the presence of good credit applicants or unlocking the cross-selling potential between the respective product and service portfolios of financial institutions, but also to rule out bad credit applicants precisely in transactional payments streams. In a diagnostic test for analysing the payment behaviour, I have used a hybrid approach comprising a combination of supervised and unsupervised learning algorithms to discover behavioural patterns. Supervised learning algorithms can compute a range of credit scores and cross-sell candidates, although the applied methods only discover limited behavioural patterns across the payment streams. Moreover, the performance of the applied supervised learning algorithms varies across the different data models and their optimisation is inversely related to the pre-processed dataset. Subsequently, the research experiments conducted suggest that the Two-Class Decision Forest is an effective algorithm to determine both the cross-sell candidates and creditworthiness of their customers. In addition, a deep-learning model using neural network has been considered with a meaningful interpretation of future payment behaviour through categorised payment transactions, in particular by providing additional deep insights through graph-based visualisations. However, the research shows that unsupervised learning algorithms play a central role in evaluating the transactional payment behaviour of customers to discover associations using market basket analysis based on previous payment transactions, finding the frequent transactions categories, and developing interesting rules when each transaction category is performed on the same payment stream. Current research also reveals that the transactional payment behaviour analysis is multifaceted in the financial industry for assessing the diagnostic ability of promotion candidates and classifying bad credit applicants from among the entire customer base. The developed predictive models can also be commonly used to estimate the credit risk of any credit applicant based on his/her transactional payment behaviour profile, combined with deep insights from the categorised payment transactions analysis. The research study provides a full review of the performance characteristic results from different developed data models. Thus, the demonstrated data science approach is a possible proof of how machine learning models can be turned into cost-sensitive data models.

Keywords: transactional payment pattern, (un-)supervised learning, machine learning, hybrid data science approach, cost-sensitive data model, predictive model

Acknowledgements

This doctorial thesis is dedicated to my beloved wife Asmae and my entire family for their continuous support and patience during my absence and for sharing their fruitful thoughts to its completion.

First, I would like to express my great appreciation to Professor Allan J Brimicombe and Dr. Yang Li for their professional guidance, constructive suggestions, encouragement, assistance and critical comments towards the completion of this piece of research work.

Lastly, I am also grateful to all remaining subject matter experts for their assistance and especially for our meaningful conversations related to my study. Meanwhile, I appreciate the support from the staff of the School of Architecture for assisting me during the period of this research.

Finally, I would also like to thank the remaining doctorial students, colleagues and friends, and all those involved directly or indirectly for their support and patience.

Table of contents

List of figures

List of tables

Abbreviations

ACM	Association for Computing Machinery
AI	Artificial Intelligence
ANN	Artificial Neural Network
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
AUC	Area Under Curve
awk	Aho Weinberger Kernighan
BFGS	Broyden-Fletcher-Goldfarb-Shanno
CDRC	Consumer Data Research Centre
CHAID	Chi-square Automatic Interaction Detection
CRISP	Cross Industry Standard Process for Data Mining
CRM	Customer Relationship Management
cSPADE	Sequential PAttern Discovery using Equivalence classes
CSV	Comma-separated values
DAG	Directed Acyclic Graphs
DNA	deoxyribonucleic acid
EDA	Exploratory Data Analysis
FIMPL	Founded-almost-implication
FinTech	Financial technology
GIS	Geographic Information System
GUHA	General Unary Hypotheses Automaton
GUI	Graphical User Interface
GZS	Gesellschaft für Zahlungssysteme
IEEE	Institute of Electrical and Electronics Engineers
ILP	Inductive Logic Programming

KYC	Know your customer
lhs	left-hand-side
MBA	Market Basket Analysis
ML	Machine Learning
MLP	Feedforward Multilayer Perceptron
MRDM	Multi-relational data mining
MS	Microsoft
MSE	Mean Squared Error
NGO	Non-Government Organisation
NN	Nearest Neighbour
NN	Neural Network
PCA	Principle Component Analysis
PKDD	Principles and Practice of Knowledge Discovery in Databases
PS	Piatetsky-Shapiro
RBF	Radial Basis Function
RFM	Recency, Frequency and Monetary
rhs	right-hand-side
ROC	Receiver Operating Characteristic
ROI	Return on Investment
sed	stream editor
SQL	Structured Query Language
SLR	Systematic Literature Review
SME	Subject Matter Expert
SMOTE	Synthetic Minority Oversampling Technique
SVMs	Support Vector Machines

Chapter 1

1. Introduction

This chapter introduces the research theme and places the research in the context of transactional behaviour analysis by focusing on financial services. It outlines the state of research and emphasises the relevance of research in exploring transactional payment behaviour in transactional data streams. Similarly, there is a lack of dependable research results concerning the specific research objectives, which will be explained in the following sections. Therefore, the broader field of research will be summarised within a defined scope of research. Finally, the chapter also explains the significance of the current research. The chapter closes with an overview of the doctoral thesis structure.

1.1 Payment transactional behaviour research statement

Banks such as HSBC, Barclays, Deutsche Bank, BNP Paribas, Citigroup and many others face many common challenges including digitalisation, fierce competition and growing enormous cost pressure, which require a deep understanding of customer behaviours, preferably on a real-time basis. Many authors generally contribute to the research area of customer behaviour analysis with a variety of different published research studies (Till and Hand, 2003; Zahir Azami, Torabi and Tanabian, 2004; Boyer and Hult, 2006; Bao and Chang, 2014). With falling profit margins, increasing customer expectations and increasing competition from financial technology firms, especially banks need to cut costs and improve their offering to avoid further attacks mainly on their core business. However, it will be crucial to understand how existing as well as new bank customers affect bank strategies, as the banking business remains under pressure in the digital age. For this purpose, it is essential to scrutinise the complex picture of financial performance effectively relevant to bank customers, where the individual's information is valued as the greatest asset of any banks given that “the volume, variety, velocity of data is increasing at an exponential rate” (Kitchin, 2014). Banks need to use the wealth of data that they own. In particular, financial institutions like banks are processing large amounts of transactional data containing different valuable information relative to their customer.

The underlying need for transactional datasets is to analyse different categorised and uncategorised payment transactions using the payment transaction ecosystem of an unknown major Czech bank, containing all payment transactions occurring in a dedicated cycle of relevant data. In order to analyse customer payment experiences, categories and behaviours during these financial transactions, the Czech dataset ecology is identified as embodying specific valuable characteristics in the payments arena, which holds interest to our research area. For instance, it stores historical payment experiences across different payment types containing categorised and uncategorised transactional data across a diverse ecosystem of bank customers.

The goal of this investigation is to analyse the digital footprint of bank customer behaviour differences and possible 360° trends in the financial transaction ecosystem using different account, savings account and credit card transactions as a starting point. The financial data ecology will be examined by categorised and uncategorised payment transactions, and – in addition to complemend the research on transactional payment behavioural analysis – the current piece of work explores the model performance for credit scorings and cross-sell candidates. The background of this research study is to investigate various customer payment behavioural patterns and ultimately their different payment categorisations, which banks probably flag in the majority of cases when a transactional payment is made.

In short, the research study seeks to identify and predict significant payment behavioural patterns using supervised and unsupervised learning methods in customers’ transactions across their payment history through categorised and/or uncategorised expenses, as well as assessing various machine learning algorithms for predicting credit scorings and cross-sell candidates. For example, it seeks to ascertain whether bank customers belong to specific payment groups and recommend a best performance algorithm for both research issues, modelling credit scores and cross-sells to foster operational efficiency and much more. The rationale behind this study is understanding the change drivers and key characteristics of their transactional behaviour based on different client expenses. The research outcomes will hold interest not only to companies such as major banks but also to other partners in the ecosystem such as fintech start-ups, the consumer industry, NGOs and governments as well as consumer protection and financial institutions. The digital revolution has provided banks with greater choices and convenience as the new currency of banking institutions is data, representing today's 'digital' oil. The more data that banks collect and analyse from their clients, the more value they can add to their operating business if the banks know how to generate profit from their transactional data.

However, banking clients are initially different, and we must also analyse how and why changes in customer behaviour can occur and how inter-dependencies regarding client characteristics and their probably negative business impacts can be mitigated. I will assess whether predicting customer behavioural changes by looking at categorised as well as uncategorised expenses based on advanced analytics is an accurate instrument for predicting promising next transactions of selected bank customers. Analysing different types of transactional data using a Czech dataset increases our understanding of various banking customer behaviours and their evolution in the digital age and helps us to efficiently quantify the weaknesses in the research. Using different datasets and various machine learning algorithms also helps us to identify the relevant key characteristics to be applied in a high-performance predictive model in modelling credit scorings or cross-selling candidates. However, by identifying the key set of customer characteristics based on payment transactions f(x) for predicting good cross-sell candidates or credit scorings (y), the research will also investigate a range of different statistical models in the area of predictive modelling, comparing them to find the best algorithm(s) for this research issues. Ultimately, merging various research issues allows us to better understand associated customer behavioural patterns in the payment transaction ecosystem. We consider this as the DNA of a digital banking client: it does not replace fundamental analysis of transactional behaviour but adds another information layer.

The following sections outlines the data required for undertaking the research to obtain insights and research outcomes using a data-driven and flexible predictive model. Accordingly, the research project will explore data-driven approaches to predict bank customers' digital behaviour within a transactional ecosystem and help to fill this knowledge gap. Thus, predictive analytics is changing the way in which banks are conducting business and can help customer leverage and unlock the power of transactional data and develop more accurate forecasts based on payment data with the help of machine learning and statistical algorithms.

1.1.1 Data quality and lacking research regarding transactional datasets

Financial institutions are constantly required to reshape and extend their data governance and quality framework to meet regulations and internal decision-making requirements. Accordingly, data science approaches can help to meet these requirements at lower costs or increase their return on equity by improving credit risk calculations based on revolutionised customer transactional behaviour through data science. Note that the research remains in the fledgling stage and there is a long research haul ahead of us. However, the study of Wang and Ma (2011) generally presents a good entry into existing literature for credit risk predictions.

Beyond meeting these requirements, innovative data science approaches are useful to solve different business challenges in terms of increasing cross-selling, marketing partnerships to develop the enhanced profile of customer (KYC) or targeted offers delivered on digital channels. With machine learning and its intelligent algorithms, data can be collected, prepared and models can be trained to evaluate their accuracy as well as monitoring their relevance over time. Therefore, machine learning helps to make better predictions than traditional analytics, especially by applying machine learning methods for credit scoring or cross-sells. Finally, technology helps to deploy high-quality models specifically based on research needs to improve their model accuracy and reduce model development time at an increased speed and scale. In today’s digital banking world, it is insufficient to know about clients’ transactions; rather, banks need reliable foresights on what their clients will do next.

The planning and preparation of the research highlighted a lack of possibilities to access and analyse real-world data in this specific research area. Unfortunately, I have spent a lot of time searching for an appropriate dataset or to gain easy access to any suitable transactional dataset regarding the chosen research field and questions. Most of the accessed datasets were not of good quality (i.e. descriptions of data fields were missing, the data size was too large/small etc.), whereby the dataset was not suitable for the research purpose itself. The basis of good reasonable research outcomes and interpretable analysis results is a dataset of excellent data quality. For a given dataset, the research design and methodology can improve its quality through the pre-processing phase with the help of various data quality initiatives, although the data preparation is also limited due to the original raw dataset. This circumstance is probably also why this field of research is not very pronounced. The higher the quality of accessed data, the more successful that the research and development will be. The paper by Zemke (2000) outlines interesting common pitfalls and possibilities when processing data for developing a financial prediction system. Their guideline can also help researchers to contribute to research success.

1.1.2 Dataset for transactional data

In order to research transactional payment behaviour, the Berka dataset was selected. The dataset was originally released for the “PKDD'99 Discovery Challenge“, and can be accessed online at http://lisp.vse.cz/pkdd99/Challenge/chall.htm. The main advantage of this transactional dataset is that it comprises real data from a Czech bank and the required data quality is already present with a certain degree of information quality (Lee *et al.*, 2002). The bank stores data about their clients with demographic data, accounts (over one million transactions), loans that the bank has already granted, and credit cards that they have issued. Transactional data about the clients, their 4,500 accounts (Pijls, 1999) and credit cards is real bank data, which were already anonymised as only the client and account ID is provided. The Berka dataset comprises eight tables, each of which is in ascii format (Pijls, 1999). It comprises the following relations, described in the figure below.

Coufal, Holeňa and Sochorová (1999) state that each account has both static characteristics (e.g. date of creation, address of the branch) given in the "account" relation and dynamic characteristics (e.g. payments debited or credited, balances) given in the "permanent order" and "transaction" relations. The "client" relation describes the characteristics of persons who can manipulate the accounts. One client can have more accounts, and more clients can manipulate a single account, whereby clients and accounts are related together in the "disposition" relation. The "loan" and "credit card" relations describe some services that the bank offers to its clients. More credit cards can be issued to an account, while at most one loan can be granted for an account. The "demographic data" relation provides some publicly-available information about the district (e.g. the unemployment rate), from which additional information about the clients can be deduced.

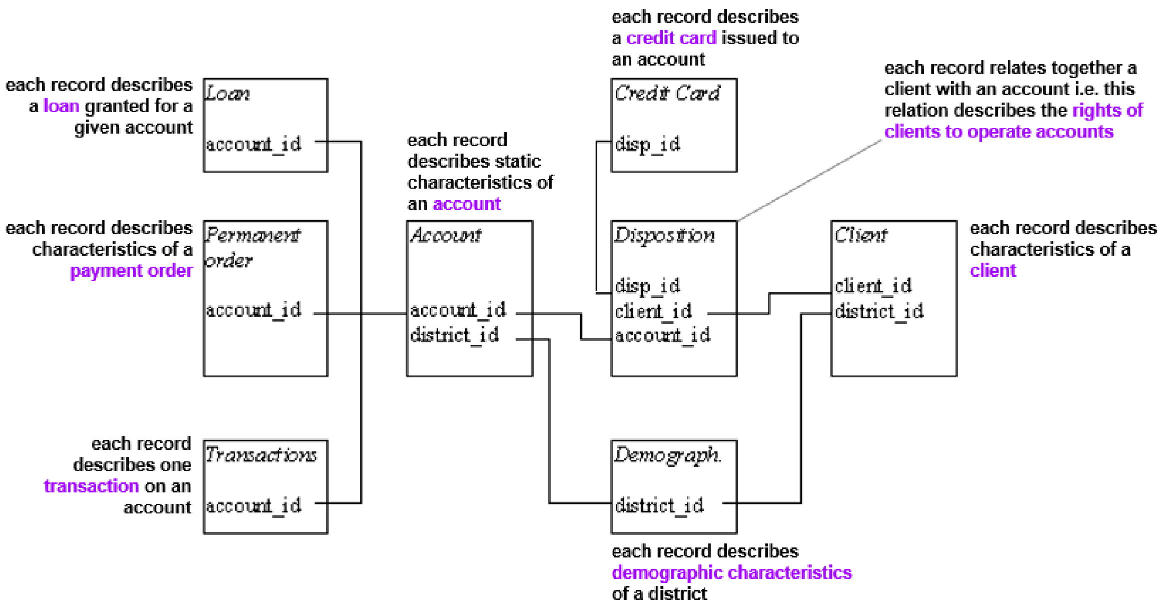

each record describes a loan granted for a given account
Loan
account_id
each record describes characteristics of a payment order
Permanent order
account_id
each record describes one transaction on an account
Transactions
account_id
each record describes static characteristics of an account
Account
account_id
district_id
each record describes a credit card issued to an account
Credit Card
disp_id
Disposition
disp_id
client_id
account_id
each record relates together a client with an account i.e. this relation describes the rights of clients to operate accounts
Client
client_id
district_id
each record describes characteristics of a client
Demograph.
district_id
each record describes demographic characteristics of a district

Figure 1-1: Overview of the entity relationship in the Berka dataset. Figure is adapted from Vaghela, Kalpesh H and Nilesh K (2014)

Another option of having access to alternative transactional datasets was considered while contacting the Consumer Data Research Centre (CDRC). The institution provides researchers access to relevant datasets based on a very restrictive data request process. The CDRC service is certainly beneficial, although I have rejected this data source for different reasons: first, in order to minimise the risk of data quality issues as the data provider is not the owner of the data and requests ethical approval for the controlled data by the data sponsors due to the context-sensitive information within the transactional dataset; and second, the observed environments of the CDRC labs in which the data analysis must be undertaken might restrict the research field as the CDRC policies request ethical approvals due to their data security requirements. In addition, there is no guarantee that the project manager assigned by CDRC will resolves all issues occurring (i.e. providing data access to context-sensitive information, etc.) during the research. I have aimed to minimise the risk of these uncertainties and proceeded with the peer-reviewed Berka dataset.

1.1.3 Building a relevant dataset for data mining

This section will provide a brief summary overview of how a relevant dataset can be built to conduct the current research.

First, the complex data structure of the original dataset must be harmonised into single records to build our target analysis table for the research. Figure 1-2 represents how the other tables and their relationships are linked to the target table. The semantic graph below helps to link the original tables step-by-step with the relevant target table. Vaghela, Kalpesh H and Nilesh K (2014) stated that “feature selection is an important preprocessing step to machine learning”. An effective subset of the original dataset can improve the performance of data processing and speed up the applied classification and clustering algorithm. For instance, a unique relevant dataset can predict categorised transactions for promoting bank products such as credit cards that might be essential for a bank due to various reasons. Van der Putten (1999) explains that the bank can cross-sell to existing clients and further strengthen the customer relationship.

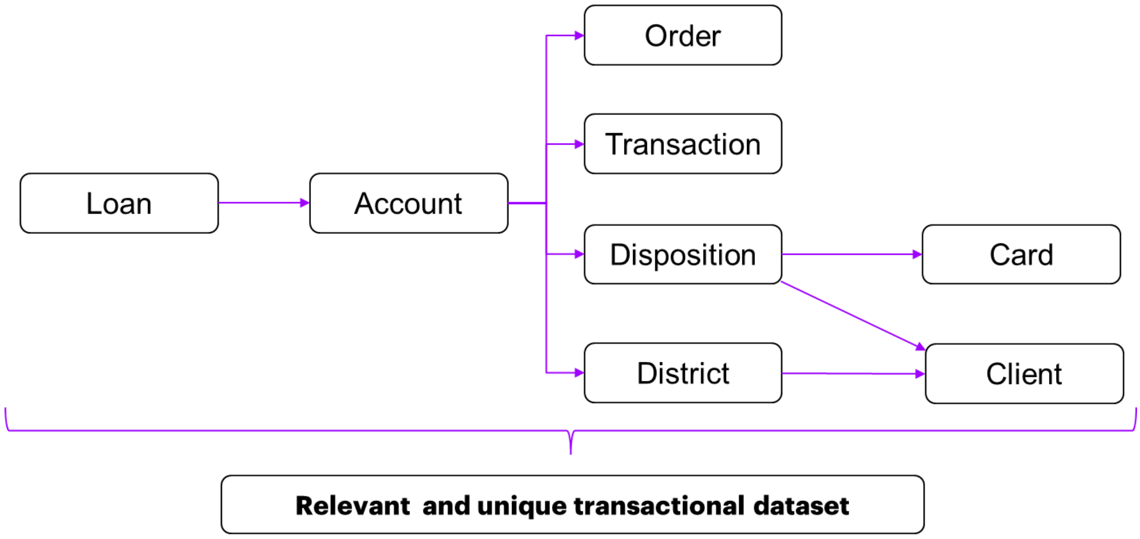


Figure 1-2: Semantic relationship graph of the transactional dataset from PKDD challenge. Figure adapted from Vaghela, Kalpesh H and Nilesh K (2014)

They are several techniques for building a relevant dataset, as InfoDist or Pearson’s calculation (Vaghela, Kalpesh H and Nilesh K, 2014). Chapter 4 will discuss the mining approach applied and how a high-end transactional dataset has been built for its research plans in detail. The current research study uses the well-known Berka dataset from the PKDD Challenge described in the above figures as a starting point for developing suitable datasets. However, not every original attribute was ultimately considered in the underlying research work. Regarding the research analysis approach, every future resulting dataset depends on the research question itself.

The following section discusses the scope of research and presents the main drivers for every single research project.

1.2. Scope of research

The thesis deals with three major fields of research for which data science approaches comprising supervised and unsupervised learning algorithms can be applied:

(1) Evaluating the creditworthiness of a bank's customer base based on a level analysis of customers' credit files and sensing the best-fitting applied machine learning algorithms concerning their performance outcomes.

The creditworthiness of bank customers is based on information that the credit bureau keeps on file about buyers. For example, if a buyer has defaulted on a previous loan, the credit institution reports it and this person has a "bad" scoring, whereby a new credit will probably not be granted. Thus, the underwriting process is used to calculate the creditworthiness of a customer base. It should be noted that in the context of the PKDD'99 Discovery Challenge different research was conducted for calculating credit scoring. Details will be discussed in the chapter 3 through the literature review, in which I have also worked out how the current work adds new knowledge to existing research.

The study undertakes customer profiling by using a range of data science techniques and tools such as the open-source software R Studio, Python and the Anaconda framework with the Spyder or Microsoft Azure ML application. Through data analytics, customer profiles are scored based on payment history, amounts owed, the length of credit history and types of credit used. Therefore, the research is designed to develop a new procedure for data-gathering, preparation and mining to evaluate the effectiveness of the new prediction models using various machine learning algorithms for credit scoring. In the research study, I had the aims to test the performance outcome of the algorithms with the help of the large-scale dataset provided by the major Czech bank. Developing new mathematical algorithm does not fall within the scope of the research. However, the research study seeks to optimise the predictive model developed on the primary research results.

(2) Evaluating cross-selling opportunities to a bank's customer base based on a deeper level analysis of customers' transactional behaviour or payment practice and sensing the best-fitting applied machine learning algorithms concerning their performance outcomes.

In physical stores, staff are trained to sell additional banking products or services to customers. Additionally, the products are strategically positioned so that the customer can “accidentally” see related items and eventually buy more. By using data analytics and especially data science practices, the seller can build a solution affinity map based on the customer order history and products viewed. For example, in the context of the PKDD'99 Discovery Challenge, different research was conducted for promoting a credit card (bank product). For instance, van der Putten (1999) decided to focus on the business objective of promoting credit card usage.

The study also conducts customer profiling in respect of the first research project by using a range of data science techniques and tools such as open-source software R studio or Microsoft Azure ML. Accordingly, the research approach is designed to match most likely cross-sell candidates and suitable products as marketing normally advertises the items as a package with a reduced price. The aim of this research is to test the performance outcome of the applied machine learning algorithms and select the most promising of the selected classification and regression algorithms for further model optimisation.

(3) Designing a timely analysis of customer behaviour based on historical transactional payment data streams to create more customer value through personalised product or service offerings in respect to the second research project by implementing various data mining methods. Data mining is the process of discovering knowledge and useful or suspicious patterns from data (Han and Kamber, 2006; Kovalerchuk and Vityaev, 2010; Zhao, 2012; Vaghela, Kalpesh H and Nilesh K, 2014; Olafsson, Li and Wu, 2019).

The research conducts a data-driven customer behaviour analysis and is in line with the first and second research projects by facilitating supervised and especially unsupervised learning algorithms to increase cross-selling between diverse customers. Therefore, the payment transaction data provides information about the purpose, recipient, value, payment method of articles bought.

In the context of the PKDD'99 Discovery Challenge, no research was conducted to analyse the categorised transaction payments in such a way that (un-)categorised transactions can be predicted or a large amount of categorised data can be examined in search of hidden patterns and predictive payment behaviour information. The scope of this research project is to develop a predictive model to assess the possibility of forecasting effectively which categorial transaction type will be made next. The machine learning algorithms applied as well as the association rules implemented within the transactional dataset will make a significant contribution towards promoting cross-selling through customer behaviour analysis.

According to knowledge based on the insights gained during my literature review, this study aims to solve the issues of predicting (un-)categorised transactions or identifying best modelling algorithms for credit scoring and cross-selling candidates using an innovative data science approach to deploy a cost-sensitive data model. The research combines different data mining tools and develops a tailored research design and methodology to realise the evaluation.

The following section provides an overview of current research aims and objectives related to the three major fields of research.

1.3. Research aim and objectives

This research aims to evaluate banking customer behaviour based on financial payment history by looking at different expense categories. Thus, the research strives to determine the key drivers of a valuable predictive model in terms of more accurately forecasting transaction categories. Thereby, the study also considers the quantity structure of the required dataset for the data mining objectives. The study aims to ascertain whether identifiable patterns can be explored based on a data science approach. For instance, a comparison of the given transaction types can reveal new knowledge during the data exploration phase.

Likewise, I had the research goals to explore predictive models and their performance by clustering banking customer behaviours regarding categorised payment transactions and discussing possible relevant research questions for the future. For instance, if a bank knows what kind of customers their clients will be when opening a new banking account, it can offer more precise product offerings. Mohan and M. (2016) emphasise that customer classification is also necessary for productive marketing. Accordingly, analysing spending behaviour to gain more customer insights can assist providing relevant offers. Banks can probably increase their profitability by collecting more precise data from their clients, assuming that banks can extract value from such vast amounts of payment data. The research also deals with the operational efficiency of a bank in terms of whether they have the capability to predict cross-sell candidates with good performance or if banks can manage their mostly expensive external credit scorings by themselves over the next years to increase their profitability.

The field of inquiry here will centre around multiple questions. For instance, one research focus area is to evaluate, explore and analyse the characteristics of payment transaction habits. Therefore, dominant and extraordinary patterns in customers' payment history can be reflected in a detailed manner with an in-depth descriptive data analysis. The research further identifies the most popular categories (i.e. payments frequency and sequence) and which payment method (i.e. current account, savings account or credit card) is used. However, a closer look may bring to light radically different payment behaviour in the course of their transactional behaviour history. Here, data analytics can play a major role, enabling banks to spot patterns and events indicating that customers are moving to new life stages and priorities. These customer behavioural insights can also support banks to partner with their customers, building long-term relationships to help identify relevant needs and providing customised products to their clients, as well as increasing cross-selling capabilities based on the digital transactional footprint of their demanding clients.

Therefore, further questions arise in terms of what kind of statistical models' banks can develop to provide more customised products as well as increasing cross-selling or effectively assessing their clients. In this context, the process of building a relevant dataset should not be underestimated as the outcomes of predicting whether a client is a good or bad borrower or cross-sell candidate also depends on the underlying dataset and its selected target attributes. Other research questions increasingly being asked in this study include which other forecast and predictive models can be used to predict customer behaviour throughout a transactional dataset more precisely, as well as their performance characteristics. A profound comparison of various statistical models may lead to significant new research questions and research needs.

One goal of the research project is to develop a data-driven approach for predicting bank customers' digital transactional footprint through various (un-)categorised payment transaction. The usage of a subset of advanced supervised and unsupervised learning models can better explain existing customer behaviour patterns in the datasets.

The following section emphasises the significance of the current research objectives.

1.4. Significance of the research study

This research has the potential to contribute original knowledge to existing customer behaviour knowledge as well as producing novel interpretations within the payment ecosystem and providing in-depth insights based on profound data science concepts and approaches. This research should hold significance since the study is conducted with a data science approach using R Studio, Python and further emerging technologies such as Microsoft Azure Machine Learning to implement an alternative and innovative research design for the promising investigation.

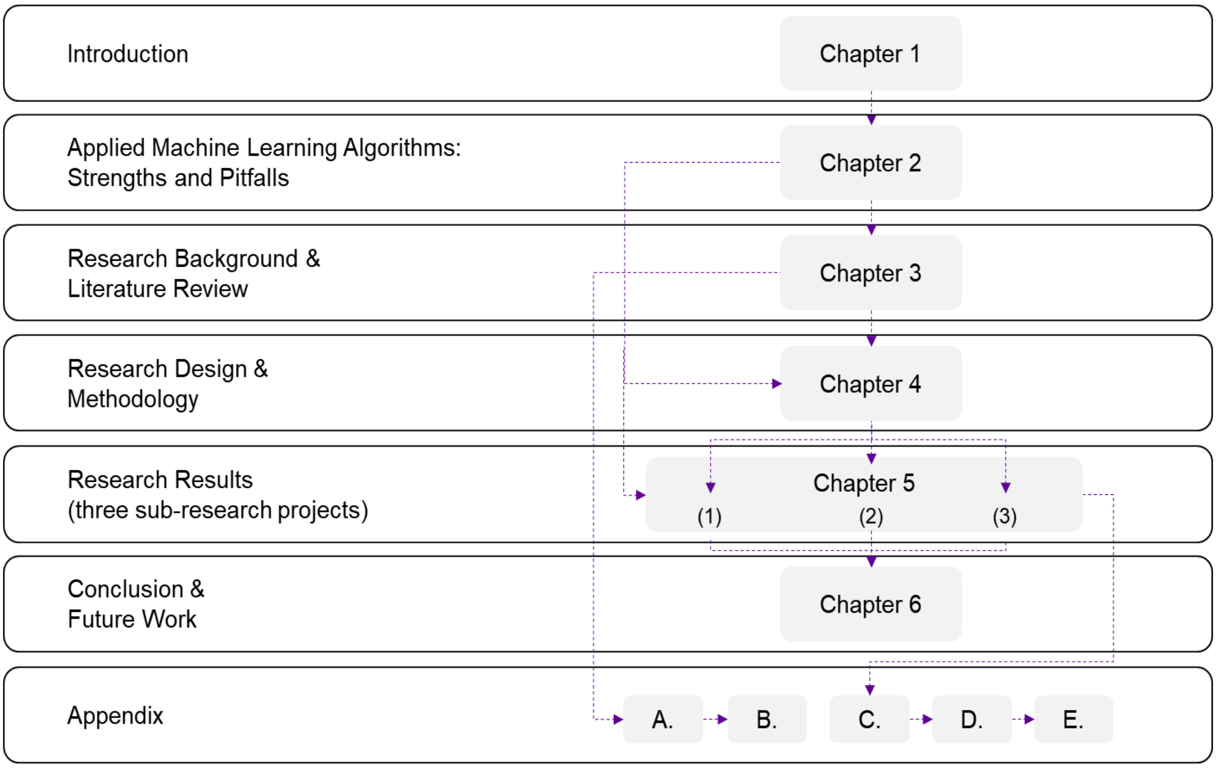
The study will discover and explore hidden behavioural patterns in the Berka dataset by looking at different categorised payment transactions. Accordingly, the research will gather new bank customer behavioural insights into payment transactions; for example, specific payment categories that are occurring in an unknown, given frequency and/or chronological order. This paper aims to explore, analyse and isolate hidden differences (i.e. seasonal outliers) in the Berka data ecology by comparing the various target attributes in the relevant transactional datasets built for the research purpose. Moreover, one of multiple research challenges will be to appraise the effectiveness of applying machine learning techniques to identify and predict categorised expenses in uncategorised datasets. The research will cluster novel and probably shifting payment categories by looking at specific categories under the application of suitable statistical algorithms. The analysis results will provide useful knowledge on early events such as modelling and predicting digital behavioural trends by promoting special bank offers, increasing the operational efficiency of a bank and making the bank customers entirely transparent with respect to existing data protection regulations.

Regarding the significance of this study, the research will highlight the need for advanced data models for analysing the characteristics of banking client habits to predict good or bad borrowers or cross-selling candidates more precisely and accurately based on a minimum viable gathered dataset. Beyond that, the research has the potential to increase the awareness of the importance of data science practice in general, as well as within the transactional payment ecosystem.

The final section of this chapter provides a detailed outline of the research thesis structure and an outlook of the entire research work.

1.5. Thesis structure

The following figure 1-3 depicts the structure of the current doctorial thesis, as well as how the respective chapters are linked. The thesis structure is sketched out according to the research background, research aims, and objectives introduced in the previous sections.



Introduction
Chapter 1
Applied Machine Learning Algorithms: Strengths and Pitfalls
Chapter 2
Research Background & Literature Review
Chapter 3
Research Design & Methodology
Chapter 4
Research Results (three sub-research projects)
Chapter 5
(1)
(2)
(3)
Conclusion & Future Work
Chapter 6
Appendix
A.
B.
C.
D.
E.

Figure 1-3: Graphical representation of the doctorial research thesis structure

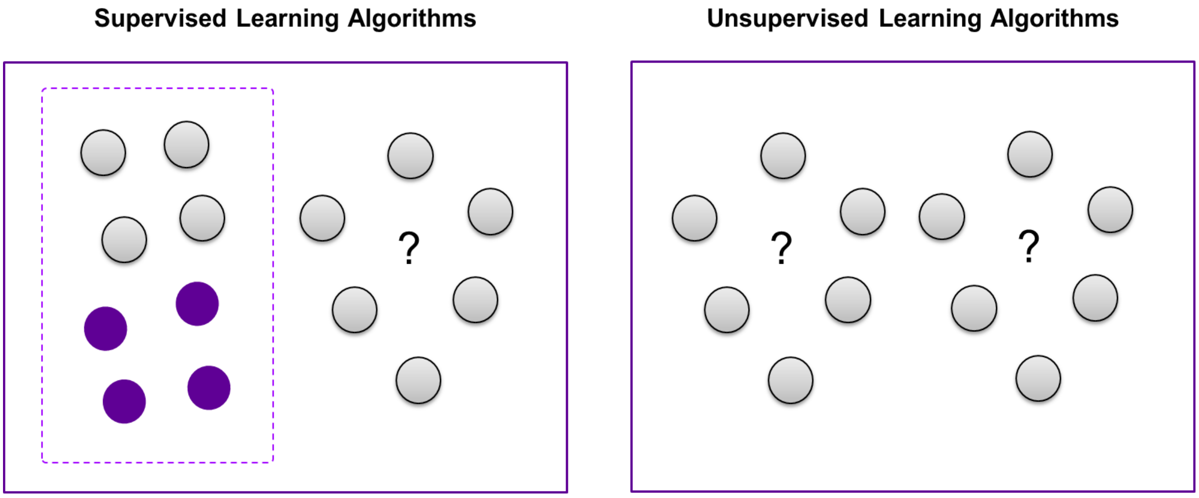
Chapter 1 provides a brief introduction to the research thesis, which forms the theoretical basis of the approach to meet the research objectives. The following Chapter 2 presents the most important algorithms used for the analysis of transactional behaviour in the different research experiments. Relevant references to literature with an analysis of the research goals are described in Chapter 3, supplemented by appendix A and B. The comprehensive systematic literature review was conducted based on research in transactional behaviour analysis, research papers submitted around the PKDD99/00 Discovery Challenge, the analysis of related research papers around the PKDD99/00 Cup – including Mendeley’s literature recommendations – and especially general studies of customer behaviour in transactional payment data streams. In Chapter 4, the research work describes the research design and methodology of the present research, as well as how the transactional payment behaviour in case of cross-selling, credit scoring and categorised payments using R

Studio, Python and MS Azure ML are modelled to pre-process the original dataset for data analysis, mining and visualisation. A set of machine learning algorithms applied for the predictive models are also included in accordance with Chapter 2. Chapter 5 presents the knowledge obtained from the research analysis, divided into descriptive and predictive results of the three main research projects, supplemented by appendices C, D and E. In Chapter 6, I have summarised and concluded the research thesis and comments upon future work.

Chapter 2

2. Applied Machine Learning Algorithms: Strengths and Pitfalls

The chapter introduces the mining algorithms used primarily to conduct a variety of research experiments aligned with the three research projects presented in the previous chapter. Therefore, the following sections describe the basic functions of the selected mining algorithms and provide a handful of practical examples to increase the understanding of the applied algorithms on the one and set their application in the research context on the other. Thus, the theoretical foundations of the applied algorithms are presented separately for each research project. It should be noted that the underlying chapter is the baseline for the algorithms used for conducting the entire research, however, the following chapters also show that selected data mining tool providers have partly modified the introduced basic algorithms. The chapter will close with a summary of all applied machine learning algorithms.



Supervised Learning Algorithms

Unsupervised Learning Algorithms

Figure 2-1: Graphical visualization of supervised vs. unsupervised learning algorithms

In general, figure 2-1 above shows the two main types of machine learning algorithms applied in current research theses: supervised and unsupervised learning algorithms. Supervised learning builds a model by learning from known labels or results, such as good/bad credit applicant or a suitable cross-sell candidate at a given time (labeled training data). A model is prepared through a training process, in which predictions are made and corrected when those predictions are wrong (if necessary). The training process continues until the model achieves a desired level of accuracy on the training data.

In contrast, unsupervised learning methods learn the common features from unknown labels or results (unlabeled training data). A model is prepared by deducing structures present in the input data. This may be to extract general rules. It may be through a mathematical process to systematically reduce redundancy, or it may be to organize data by similarity.

2.1. Critical assessment of applied machine learning algorithms for the 1st research project - credit scoring model

The section primarily describes the machine learning algorithms used for the data analysis of the first research project. Therefore, basic characteristics and functionality of the selected algorithms are presented to increase the understanding of the developed credit scoring models and their prediction results. The focus was not set to present the detailed mathematical implementation and definition of each supervised learning algorithm, but to explain the procedure of the applied machine learning algorithms by way of examples.

2.1.1 The multiclass neural network

The aim of the first two research projects is to apply a range of classification models for credit scoring and cross-selling in order to understand which supervised learning algorithm works best. To gain these performance insights, I have processed different types of modified algorithms. What makes neural networks so special is that a successful net can be created without understanding how it works. However, I have learned and explained the characteristic of these model outcomes using various modified neural networks.

One of the research experiments uses the multiclass classification model, which solves the problem of classifying instances into one of three or more instances. The applied multiclass neural network from MS Azure ML solves the multiclass classification problem based on neural networks. The distinction of that MS Azure ML algorithm is to have more than one neuron in the output layer. In practice, the final layer of a neural network is based on N logistic classifier. The adjusted algorithm neural network model, “Multiclass Neural Network”, can be used to predict a target that has multiple values. The classification uses a tagged transactional dataset that includes the label column ‘status’ for the first and ‘cardholder’ for the second research project, as illustrated in figure 2-2.

The neural network is primarily used for predictive modelling of the credit scoring case and cross-selling case, in which the adjusted multiclass neural network algorithm is trained on a pre-processed dataset. The acceptable range of the output is usually between 0 and 1. The connections between the input, hidden and output layer are modelled as weights. Negative values reflecting an inhibitory connection and positive values are reflecting an excitatory connection. According to this theory, various types of MS Azure ML neural network algorithms use this concept to model complex relationships between inputs and outputs (Steinhaeuser, Chawla and Ganguly, 2015). Training a neural network is the process of finding values for its weights and bias terms, which are used in conjunction with values given in the input layer to generate outputs and predictions given in the output layer. The created model is used to make predictions on transactional data with unknown outputs.

Schetinin *et al.* (2003) describes how a multiclass neural network can be learned from a large-scale clinical electroencephalogram’s dataset. The algorithm trains hidden neurons separately to classify all the pairs of classes in order to find best pairwise classifiers relevant to the classification problem. Regarding the current research project, an n-class model should be learnt from the large-scale transactional Berka dataset to correctly classify credit scores or cross-sell candidates of the training and test set.

The idea of the multiclass neural network is to separately train the hidden neurons of the neural network. The algorithm learns to divide the examples from each pair of classes. The aim is to learn n (n - 1) / 2 binary classifiers from n classes. Schetinin *et al.* (2003) defines a multiclass neural network as follows:

Let $f_{i,j}$ be a threshold activation function of a hidden neuron which learns to divide the examples x of *i*th and *j*th classes Ω_i and Ω_j respectively. The output y of the hidden neuron is:

$$y = f_{i,j}(x) = 1, \forall x \in \Omega_i, \text{ and } y = f_{i,j}(x) = -1, \forall x \in \Omega_j$$

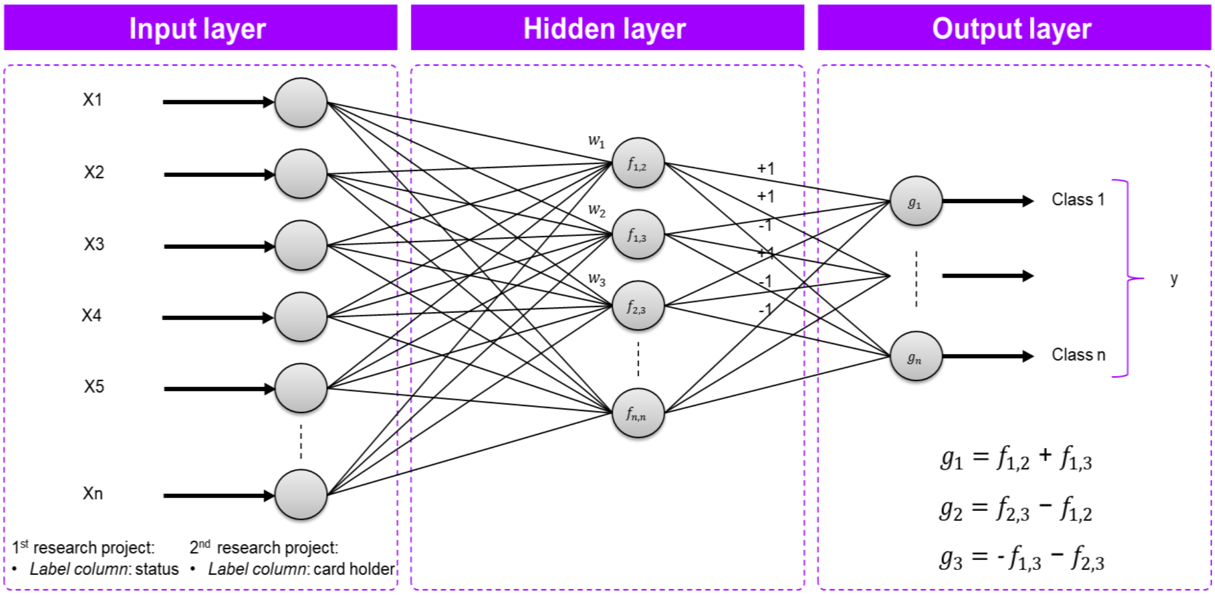
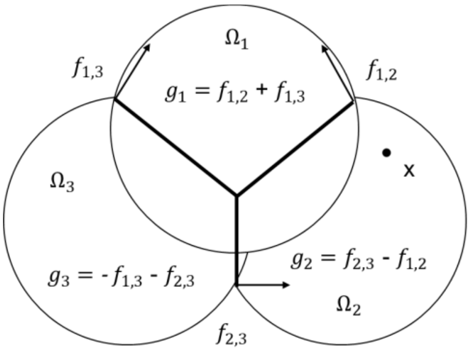


Figure 2-2: Visualization of a multiclass neural network

Assume q = 3 classification problem with overlapping classes Ω_1, Ω_2 and Ω_3 centered into C_1, C_2, and C_3, as figure 2-3 depicts. The number of hidden neurons for this example is equal to 3. In figure 2-2 and 2-3, lines $f_{1,2}$, $f_{1,3}$ and $f_{2,3}$ depict the hyperplanes of the hidden neurons trained to divide the examples of three pair of the classes, which are (1) Ω_1 and Ω_2, (2) Ω_1 and Ω_3, and (3) Ω_2 and Ω_3.



Ω_1
$f_{1,3}$
$f_{1,2}$
$g_1 = f_{1,2} + f_{1,3}$
Ω_3
x
$g_3 = -f_{1,3} - f_{2,3}$
$g_2 = f_{2,3} - f_{1,2}$
Ω_2
$f_{2,3}$

Figure 2-3: The dividing surfaces for the hyperplanes

By combining these hidden neurons into n = 3 groups, the algorithm built new hyperplanes g_1, g_2, and g_3. The first one, g_1, is a superposition of the hidden neurons $f_{1,2}$ and $f_{1,3}$., i.e., $g_1 = f_{1,2} + f_{1,3}$. The second and third hyperplanes are $g_2 = f_{2,3} - f_{1,2}$ and $g_3 = -f_{1,3} - f_{2,3}$ correspondently. Figure 2-3 above also shows that in the general case for n > 2 classes, the neural network consists of n output neurons $g_1, \ldots, g_n$ and n (n – 1) / 2 hidden neurons $f_{1,2}, \ldots, f_{i,j}, \ldots, f_n$ - 1/n, where *i* < *j* = 2, …, n.

Learning classification models from a transactional dataset are still a complex problem because of the following: First, transactions are generally not static data which depends on an individual payment behaviour of bank customers; second, the transactional dataset can be noisy and incomplete; third, a given set of transaction attributes may contain attributes which are non-important to the classification problem and may probably diminish the classification results; and fourth, transactional datasets are large-scale data which are recorded during several time-periods, and for this reason the learning time is crucial.

A common criticism of applying neural networks, adapted to current research objectives, is that they require a large diversity of training for real-world transactional datasets. The reason is that any learning machine needs sufficient representative samples in order to identify the hidden behavioural structure that allows it to generalize for new credit scoring cases.

2.1.2 The two-class neural network

The two-class neural network from the MS Azure ML library reduces the multiclass classification problem to a binary classification problem. The algorithm has been developed based on a neural network. Therefore, the two-class neural network model predicts a target that has only two values.

For example, figure 2-4 depict the used neural network model to predict binary results, such as whether a bank customer has a certain credit score or not, or whether a bank customer is a suitable cross-selling candidate for banking product promotion or not. Regarding binary classification (something belongs to class A or class B in case of the dependent variable ‘status’ as well ‘cardholder’), the research experiments use the output layer of a neural network to run the two-class neural network. The algorithm of MS Azure ML uses 1 output node. In practice, this means output 0 (<0.5) is considered class A and 1 (>=0.5) is considered class B.

For each observation X_i the research experiments can have one output variable O_n that can take two values: The applied MS Azure two-class neural network creates a binary classifier “good” or “bad” for the dependent variable ‘status’ for the first research project credit score, and a binary classifier “yes” or “no” for the dependent variable ‘cardholder’ for the second research project.

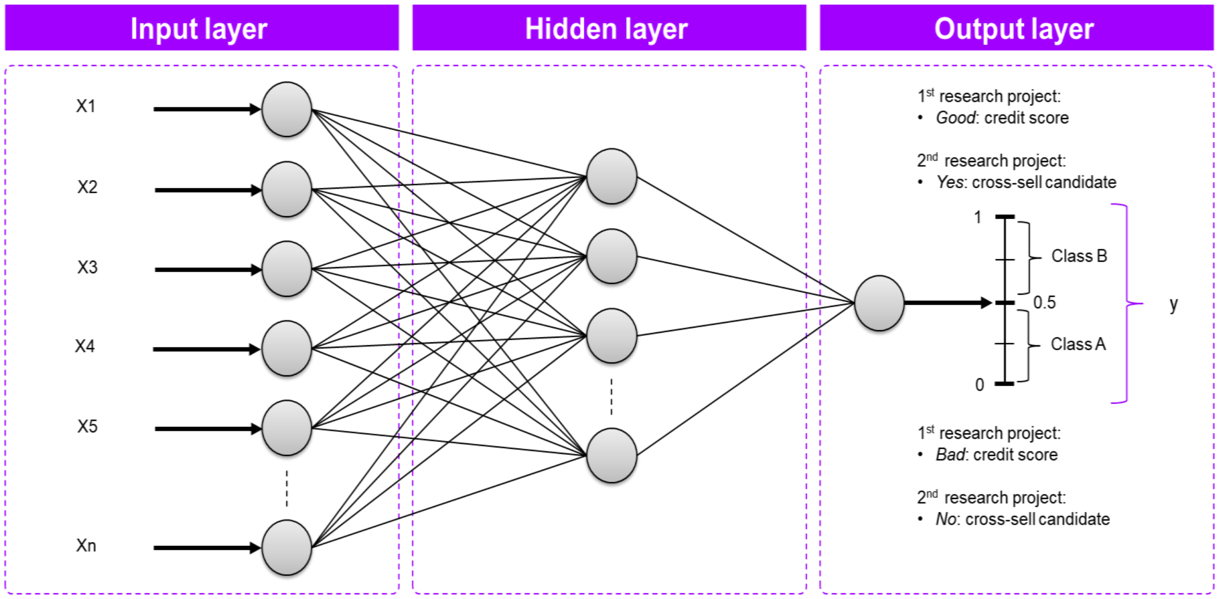


Figure 2-4: Visualization of a two-class neural network

The relationship between inputs and outputs is learned from training the neural network on the transactional data. The direction of the graph proceeds from the inputs through the hidden layer and to the output layer. All nodes in a layer are connected by the weighted edges to nodes in the next layer. To compute the output of the network for the transactional data input, a value is calculated at each node in the hidden layers and in the output layer. The value is set by calculating the weighted sum of the values of the nodes from the previous layer. An activation function is then applied to that weighted sum.

2.1.3 The two-class decision forest

Decision tree methods generally construct a model of decisions made based on actual values of attributes in the data. The decisions fork in tree structures until a prediction decision is made for a given record. The two-class decision forest algorithm from MS Azure ML is thereby a good choice if one wants to predict a target with a maximum of two results. The two-class classification model creates multiple decision trees and then votes on the most popular output class. Using the entire transactional dataset, many individual classification trees are created, but with different starting points.

A decision tree is a tree in which the nodes represent decisions (a square box), random transitions (a circular box) or terminal nodes, and the edges or branches are binary (good/bad) representing possible paths from one node to another.

The used Berka dataset consists of different entities with many features such as income (numeric), age (numeric), balance (numeric), etc. An example of a learned decision tree for classification and prediction is illustrated in figure 2-5 below.

Figure 2-5 describes the decision-making process of whether a bank customer is a good or bad credit applicant. For instance, the root or topmost node of the tree (and there is only one root) is the decision node that splits the Berka dataset using the feature 'income' that results in the best splitting metric evaluated for each subset or class in the dataset that results from the split. The decision tree learns by recursively splitting the Berka dataset from the root onwards (in a greedy, node by node manner), according to the splitting metric at each decision node (i.e., 'age' and 'balance'). The terminal nodes, such as 'balance', are reached when the splitting metric is at a global extremum.

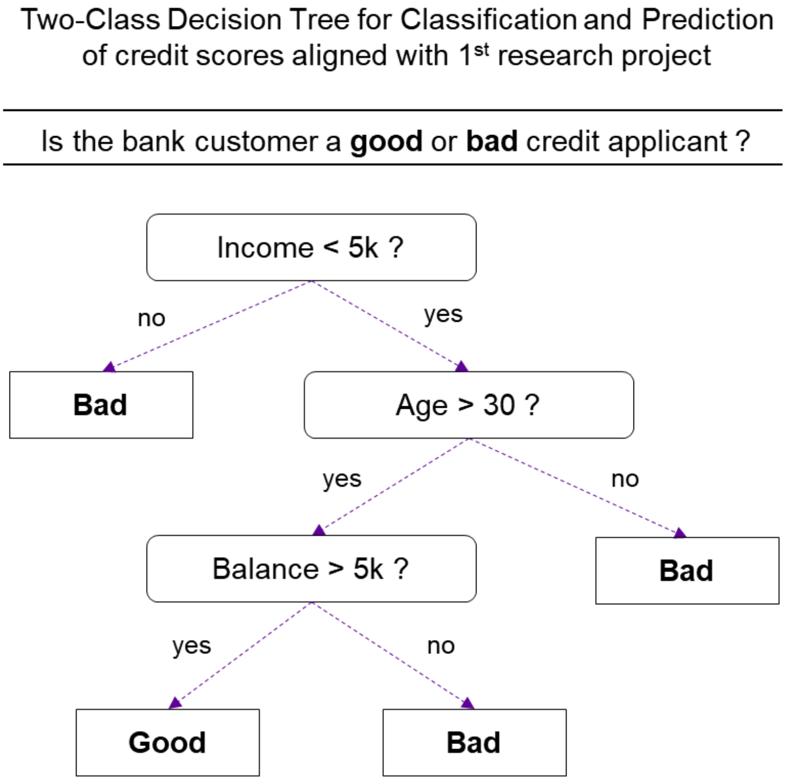

Two-Class Decision Tree for Classification and Prediction
of credit scores aligned with 1st research project
Is the bank customer a **good** or **bad** credit applicant ?
Income < 5k ?
no
yes
Bad
Age > 30 ?
yes
no
Balance > 5k ?
Bad
yes
no
Good
Bad

Figure 2-5: Visualization of a two-class decision forest for credit scores

Imagine that the Berka dataset consists of many numbers provided through the entire dataset, and the example shows only an excerpt of that dataset at the top of the figure above. The features 'income', 'age' and 'balance' consist of numerical values and aim to separate the classes (yes or no, in our case) using their features. The features (a square box) represent (> vs. <) a decision node and whether the observation is correct or not. So, in practice, the two-class decision forest algorithm works as follows:

The feature ‘income’ seems like a pretty obvious root feature to split, as all but a few of the numerical values are > 5k. In doing so, the algorithm can use the question, “Is the income < 5k?” to split the first node. The answer is displayed in a node of a tree as the decision point where the path splits into two—observations that meet the criteria income > 5k go down the “yes” branch, and ones that do not go down the “no” branch.

The “no” branch (the bad credit applicants) contains all banking clients who have an income of less than 5k so the algorithm is done there, but the “yes” branch can still be split further. The two-class decision forest algorithm can use the second feature ‘age’ and ask, “Is the age > 30?” to make a second split.

A bank customer whose age is greater 30 go down the “yes” subbranch, and a bank customer whose age is less than 30 goes down the right subbranch and the algorithm is done. The “yes” subbranch can still be split further by using the third feature ‘balance’ and ask, “Is the balance >5k?” to make the last determining split in order to predict a credit applicant with suitable credit scores. Observations that meet the criteria balance >5k go down the “yes” branch (good credit applicant) and ones that do not go down the “no” branch (bad credit applicant).

The binary decision tree was able to use the three features to split up the data perfectly. It should be noted that in real life the data obviously will not be this clean, but the logic that a two-class decision forest tree employs remains the same. At each node in the decision tree, the algorithm will ask what feature can split the observations at hand in a way that the resulting groups of bank customers are as different from each other as possible, and the bank customers of this subgroups are as similar to each other as possible.

In general, applying a two-class decision forest algorithm has many advantages for classifying good or bad credit scores as well as cross-selling candidates. Decision trees allows us to capture non-linear decision boundaries. Large amounts of data can be used for trainings and predictions because they are efficient in calculation and memory usage. The feature selection is integrated into the training and classification process, and the trees can handle noisy data, many features and datasets with different distributions. However, simple decision trees can overfit on the transactional data and are less generalizable than tree ensembles. The average of this decision forest is a tree that avoids overfitting. It should be noted that decision forests can use a lot of memory. The reason behind this is that each tree in the decision forest algorithm returns an unnormalized frequency histogram of classes. The aggregation process sums these histograms and normalizes the result to get the "probabilities" for each class. The trees with high predictive reliability have a higher weighting in the final decision of the group.

2.1.4 The two-class support vector machine

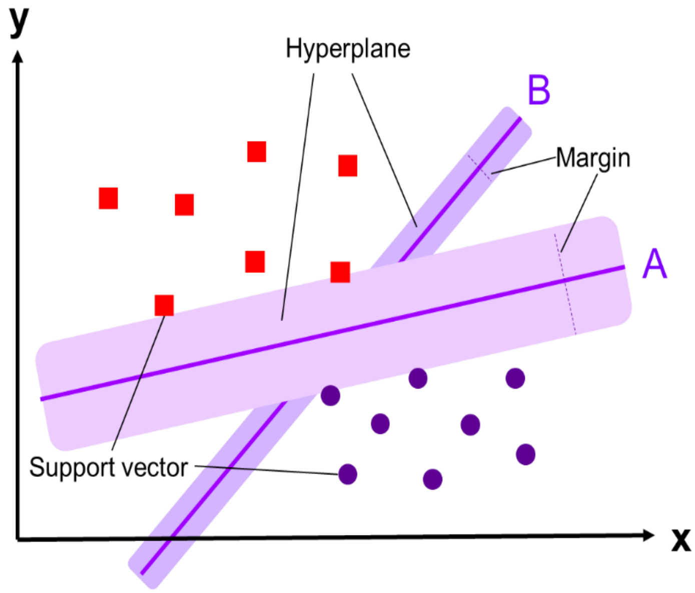
The two-class support vector machine algorithm from MS Azure ML creates a binary classification model which is suited to prediction of two possible outcomes, based on the categorical variable ‘status’ for the credit scoring case or ‘cardholder’ for the cross-selling case (aligned with the second research project).

The algorithm analyses transactional data and recognizes patterns in a multi-dimensional feature space called the hyperplane. A hyperplane in an n-dimensional Euclidean space is a flat, n-1 dimensional subset of that space that divides the space into two disconnected parts (Arreola, Fehr and Burkhardt, 2007). All transactional data are represented as points in this space and are mapped to output classes (good/bad) for the first research project or to output classes (yes/no) for the second research project in such a way that the category ‘status’ is divided by as wide and clear a gap as possible. The predictive model assigns new examples into one class or the other, mapping them into that same space.

A support vector machine divides a set of objects into classes so that as wide an area as possible remains free of objects around the introduced class boundaries. The theory behind the two-class support vector machine algorithm is simple, as the binary classification model creates a line or a hyperplane which separates the data into classes described above. The major goal is to find the ideal separating line or hyperplane between data of two classes if possible. In terms of machine learning, the best result will be to get a more generalized separator.

In practice, the starting point for building a support vector machine is a set of training objects, each of which knows to which class it belongs. Each object is represented by a vector in a vector space. The task of the support vector machine is to fit a hyperplane into this space, which acts as a dividing surface and divides the training objects into two classes (good/bad). The distance (margin) between the vectors closest to the hyperplane is maximized. This wide, empty border should later ensure that even objects that do not correspond exactly to the training objects are classified as reliably as possible. The separator line or hyperplane for which the margin is maximum is the optimal line or hyperplane. This also means that the margin has no interior training objects.

The two-class SVM only optimizes the weights between the input features and the output. Training is done by solving a quadratic optimization problem to optimize the weights (Abdullah, Veltkamp and Wiering, 2009). Figure 2-6 below shows a support vector which contains all information for constructing the decision function of a classifier. The two-class SVM algorithm will solve the optimization problem in a linear way, since otherwise the standard kernel activations become too complex.



Set of trainings data:

$$\{(\mathbf{x}_i, y_i) | i = 1, \ldots, m; y_i \in \{-1, 1\}\}$$

x_i is a feature vector representation
y_i the class label (negative or positive) of a training compound i

The mathematical formulation fot the class affiliation is described as:

$$y_i = \operatorname{sgn}(\langle \mathbf{w}, \mathbf{x}_i \rangle + b)$$

w is the weight vector,
x is the input feature vector
b is the bias

Figure 2-6: Visualization of a linear SVM model - two possible dividing lines with different edge sizes

As an example, the illustration describes two possible dividing lines with different edge sizes. When inserting the hyperplane, A or B, it is not necessary to consider all training vectors. Vectors that are further away from the hyperplane and are "hidden" to a certain extent behind a front of other vectors do not influence the position and orientation of the parting plane. The hyperplane is only dependent on the vectors closest to it—and only these are needed to exactly describe the plane mathematically.

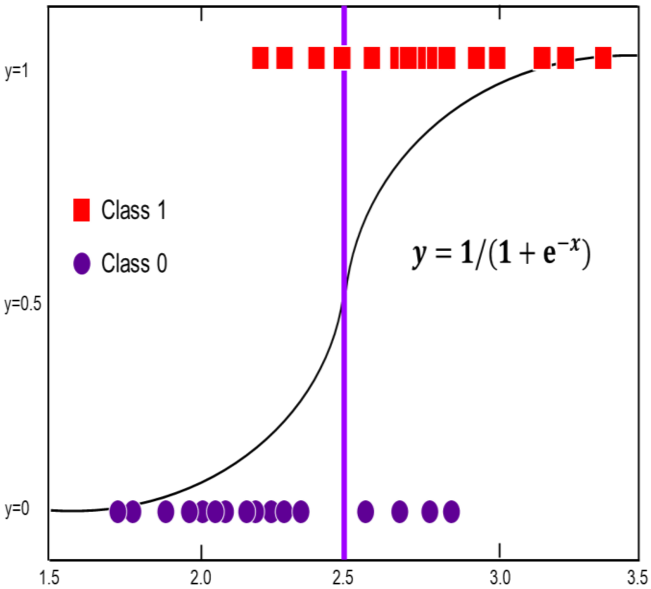
In sum, a binary classification model uses a linear SVM model with the goal to calculate the parameters w and b of this "best" hyperplane. Detailed mathematical discussions can be followed from various studies (e.g., Cristianini and Shawe-Taylor, 2000; Kumar, Bhattacharyya and Gupta, 2014; Hastie, T; Tibshirani, R; Friedman, 2017; Abu El-Atta, Moussa and Hassanien, 2018)

Huang *et al.* (2018) emphasize that SVM is a powerful method for building a classifier. The target of the conducted research experiment is to create a decision boundary (hyperplane) between two classes (good vs. bad) that enables the prediction of labels from one or more feature vectors. The classifier of the two-class support vector machine is useful for predicting between two possible outcomes that depend on the categorical predictor variable ‘status’ or ‘credit cardholder’ (for the second research project). It is recommended to normalize the transactional dataset before using a two-class support vector machine to train the classifier. It is worth mentioning that the algorithm works well on simple datasets when the research goal is speed (uses a small subset of training data in the decision function to be memory efficient) over accuracy.

2.1.5 The two-class logistic regression

Regression methods are a workhorse of statistics and have been co-opted into statistical machine learning. This may be confusing because the literature uses regression to refer to the class of problem and the class of algorithm. However, applying regression algorithm to solve a classification problem is a process whose important concept can be described as follows: Jurafsky and Martin (2019) define logistic regression as a supervised machine learning classifier that “extracts real-valued features from the input, multiplies each by a weight, sums them, and passes the sum through a sigmoid function (logistic function) to generate a probability” for decision-making based on a threshold.

Although a logistic regression confusingly includes “regression” in the name, the method is actually a powerful machine learning algorithm for two-class classification. Figure 2-7 below shows an “S”-shaped curve instead of a straight line, which makes the algorithm a natural fit for dividing data into groups. It is important to note that logistic regression gives linear class boundaries. A linear approximation between the input variables with the output goes hand in hand with the usage of the two-class logistic regression algorithm. Data transformations of the input variables given in the dataset that expose this linear relationship can result in a more accurate model. With respect to the research questions, building a cost-sensitive predictive data model also assumes to identify highly correlated input variables first by calculating pairwise correlations between all inputs, and second, removed from the model to avoid overfitting in order to increase accurate predictions.



Logistic regression assumes a logistic distribution of the data, where the probability that an example belongs to class 1 is the formula:

$$p\ (x;\ \beta 0, \ldots, \beta D - 1)$$

- *x* is a D-dimensional vector containing the values of all the features of the instance
- *p* is the logistic distribution function
- ß{0},..., ß{D-1} are the unknown parameters of the logistic distribution

The algorithm tries to find the optimal values for ß{0},..., ß{D-1} by maximizing the log probability of the parameters given in the inputs. Maximization is performed by using a popular method for parameter estimation, called Limited Memory BFGS.

Figure 2-7: Visualization of a two-class logistic regression

The figure above illustrates a logistic regression to two-class data with just one feature; the class boundary is the point at which the logistic curve is just as close to both classes (i.e., good/bad or yes/no). According the plot above, the numbers between 1.5 and 3.5 transformed into the output values are modeled in a binary value (0 or 1) using the logistic function.

For example, the research experiments model credit scores (good or bad) as well cross-sell candidates (yes or no) from their transactional payments data, so the first class could be good/yes and the logistic regression model could be written as the probability of good credit score/cross-sell candidate given a bank customer transactional payment behaviour, or more formally as given in the formula above: P (credit score = good | transactional data) or P (cross-sell candidate = yes | transactional data). If the curve goes to positive infinity, class predicted (dependent variable) will become 1, and if the curve goes to negative infinity, class predicted will become 0. If the output of the logistic function is more than 0.5, we can classify the outcome as good credit score or yes for cross-sell candidate, and if it is less than 0.5, we can classify it like bad credit score or no cross-sell candidate. If the output of a research experiment is 0.8, we can say in terms of probability as the following: There is an 80% chance that the bank customer will be a good credit borrower or a suitable cross-sell candidate.

Learning a logistic regression model from the transactional payment data is realized by using a maximum-likelihood estimation. Therefore, a minimization algorithm is applied to optimize the best values for the unknow parameters ß for the training data. This is implemented by MS Azure ML in practice, using efficient numerical optimization algorithm L-BFGS (limited memory Broyden-Fletcher-Goldfarb-Shanno). Detailed configuration parameters of the algorithm are given in Appendix D.7. The best unknown parameters ß would result in a model that would predict a value very close to 1 (e.g., good/yes) for the default class and a value very close to 0 (e.g., bad/no) for the other class. The intuition for maximum-likelihood for logistic regression is that a search procedure seeks values for the unknown parameters (Beta values) that minimize the error in the probabilities predicted by the model to those in the transactional data (e.g., probability of 1 if the data is the primary class).

Logistic regression is one of the most popular regression algorithms. It is a type of regression analysis used for predicting the outcome of a categorical criterion variable (a variable that can take on a limited number of categories) based on at least one predictor variables (i.e., ‘status’ or ‘credit cardholder’). The probabilities describing the possible outcome of a single trial are modelled, as a function of explanatory variables, using a logistic function.

The two-class logistic regression model allows the prediction of two (and only two) results. The application of this supervised learning method requires a dataset that already contains the results for training the model. This means that the dataset must contain a label or class column which contains exactly two values (outcomes). The logistic regression method is used to predict the probability of a continuous-valued outcomes (such as a credit score of bank customers’ transactional payment data). For this purpose, the model predicts the probability of occurrence of an event by fitting data to a logistic function. A detailed mathematical description of the logistic regression can be found in Fahrmeir *et al.* (2013). The implementation of the two-class logistic regression from the MS Azure ML library was used to conduct the research experiments of the first two research projects.

However, most problems of logistic regression involve classifying a new observation into one of the many possible classes based on the value of its explanatory variable. The logistic regression may be better suitable for cases in which the dependent variable is dichotomous, such as yes/no or good/bad, while the independent variables can be nominal, ordinal, ratio or interval. According to the first two research projects, the two-class logistic regression algorithm creates a logistic regression model to predict the dependent variable ‘status’ for being a (good/bad) credit borrower or ‘credit cardholder’ for being a (yes/no) cross-sell candidate.

The efficiency of a two-class logistic regression algorithm is one of many advantages, since the implementation does not require high computation power. The model is highly interpretable and does not require input features to be scaled or any other advanced tuning steps. The model outputs are well-calibrated predicted probabilities for observations. However, feature engineering plays an important role in regards to the general performance of logistic regression, for instance, removing input variables that are unrelated to the output variable as well as variables that are correlating to each other will result to better performance. The application of the two-class logistic regression is due to its simplicity, and straightforward nature a good research baseline for performance measurements of other more complex supervised learning algorithms.

On the other hand, a two-class logistic regression also has drawbacks, since it is not the most powerful algorithm and it is not able to handle a large number of categorical features. A transformation of non-linear features is required to avoid overfitting. Finally, the performance of a two-class logistic regression is also affected by the independent variables x that are not correlated to the target variable y and are very similar or correlated to each other.

2.1.6 The random forest

One of the main research aims and objectives presented in the previous introduction part is to develop a cost-sensitive data model for the first two research projects. For this purpose, I have created an optimized classification model using a random forest algorithm with RStudio’s varImp() function. This procedure is necessary to measure the variable importance on the Berka dataset.

A random forest is a classification method consisting of several uncorrelated decision trees. All decision trees have grown under a certain type of randomisation during the learning process. For a classification, each tree in that forest can make a decision and the class with the most votes decide the final classification. Following characteristics are describing the procedure of a random forest algorithm in detail.

First, the classifier trains very quickly: This advantage results from the short training or construction time of a single decision tree and from the fact that the training time for a random forest increases linearly with the number of trees. Second, the evaluation of a test example is done individually on each tree and can therefore be parallelized, and a random forest is very efficient for large amounts of data. Lastly, important classes can be identified, and the relationship between classes can be recognized.

According to Pavlov (2019), the following principles should be applied for each decision tree in the forest:

First step is to draw n bootstrap samples. Second, from M features (features or dimensions) of the training data that are displayed at each node in the tree m ≪ M, characteristics are selected randomly and should be considered as criteria for the cut (split). For instance, the subsequent selection of a characteristic from this set can be done by minimizing entropy. Finally, the tree is fully expanded and not pruned back (Pruning).

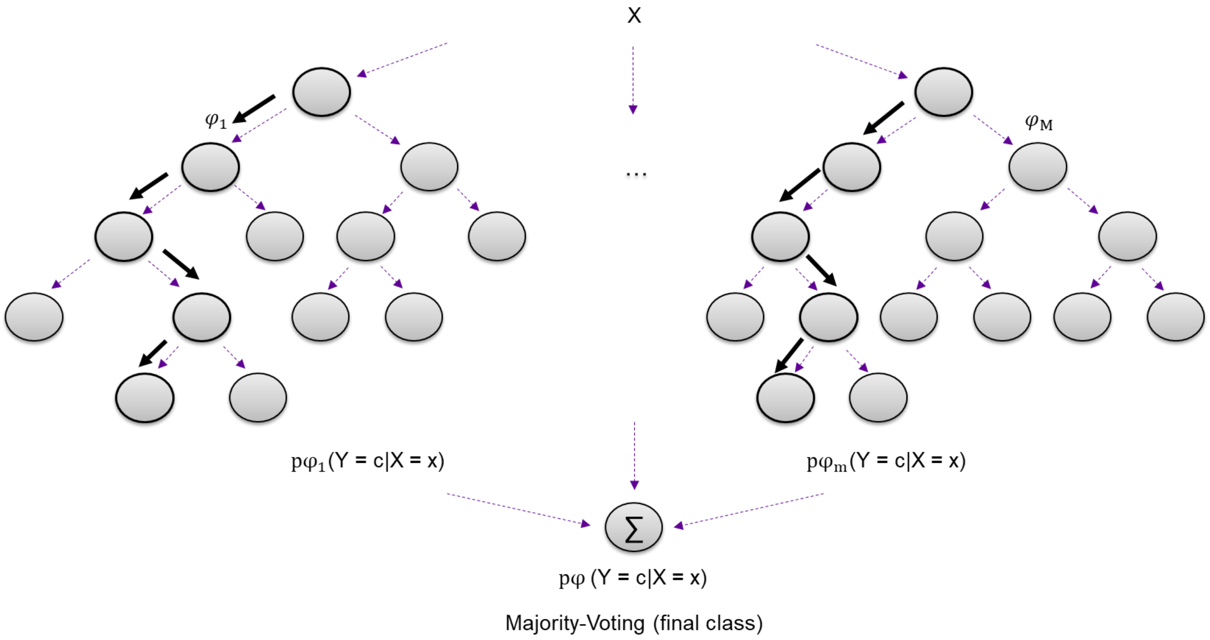

X
φ_1
φ_M
…
$p\varphi_1$(Y = c|X = x)
$p\varphi_m$(Y = c|X = x)
Σ
$p\varphi$ (Y = c|X = x)
Majority-Voting (final class)

Figure 2-8: Visualization of a random forest

Figure 2-8 shows that random forest is an extension of decision trees. One of the main advantages of using random forests is the ease with which we can see what features or variables contribute to the classification and their relative importance based on their location depth-wise in the tree. As shown in the illustration above, important features tend to be at the top of each tree and unimportant variables are located near the bottom.

To classify an input, it is evaluated in each tree. The class that is selected most often is the output of the random forest. The research used the described random forest algorithm above to rank the importance of variables in a classification problem in a natural way. This technique is implemented in the R package “randomForest” and is used to build a cost-sensitive data model for the first two research projects.

The first step in measuring the variable importance in the Berka dataset is to fit a random forest to the transactional data:

$$Dn = \{(X_i, Y_i)\}_{i=1}^{n}$$

During the fitting process, the out-of-bag error for each data point is recorded and averaged over the forest. To measure the importance of the x-th feature after training, the values of the x-th feature are permuted among the training data and the out-of-bag error is again computed on this perturbed transactional dataset. The importance score for the x-th feature is computed by averaging the difference in out-of-bag error before and after the permutation over all trees. The score is normalized by the standard deviation of these differences. A detailed statistical definition of the variable importance measure is analyzed by Zhu et al. (2015). Finally, the higher the score of a feature, the more important the feature in the transactional dataset is.

2.2 Critical assessment of applied machine learning algorithms for the 2nd research project - cross-selling model

The section primarily describes the delta of machine learning algorithms used for the data analysis of the second research project. Therefore, basic characteristics of the selected algorithms are presented to increase the understanding of the developed cross selling models and their prediction results. Further machine learning algorithms used in the analysis of the second research project are presented in the previous section.

2.2.1 The two-class decision jungle

A two-class decision jungle algorithm is based on decision trees, a foundational machine learning concept. There are many forms of decision trees, but they all have the same approach: subdivide the feature space into regions of roughly uniform labels.

Decision trees are trained on data for classification problems, and they are often fast and accurate and a big favorite in machine learning.

Figure 2-9 describes a two-class decision forest algorithm for building a classification and prediction model of cross-sell candidates. Main characteristics of the decision tree include that only one path to every node is allowed, and a tree is used as the base learner. The detailed algorithm procedure is explained in the previous section and can be applied in the same way to classify and predict cross-sell candidates. Compared to a decision jungle, multiple paths from root to each leaf are allowed, and a directed acyclic graph (DAG) is employed as the base learner.

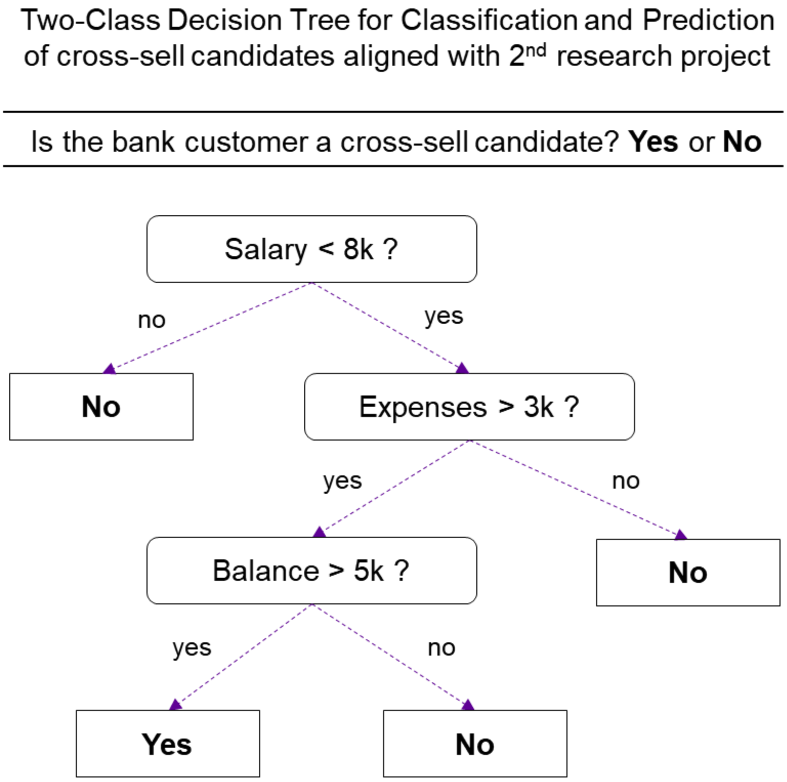

Two-Class Decision Tree for Classification and Prediction
of cross-sell candidates aligned with 2nd research project
Is the bank customer a cross-sell candidate? Yes or No
Salary < 8k ?
no
yes
No
Expenses > 3k ?
yes
no
Balance > 5k ?
No
yes
no
Yes
No

Figure 2-9: Visualization of a two-class decision tree for cross-sell candidates

Shotton, Nowozin, *et al.* (2013) defines the concept of a decision jungle through a set of definitions as follows:

Definition (decision DAG) - A decision DAG is a rooted DAG G = (V,E) for which all directed path from the root node r to a node v are of the same lengths and whose nodes v ∈ V are augmented with the following attributes:

- If v is a leaf node, then v is associated with a class histogram h_v
- If v is not a leaf node (i.e., the root node or an internal root), then v is augmented the tuple $(d_v, 0_v, l_v, r_v)$ ∈ {1, …, n} x ℝ x {w ∈ V: (v,w) ∈ E} where

- d_v is the feature dimension

- 0_v is the threshold

- l_v is the left child node

- r_v is the right child node

Definition (Binary decision tree) - A binary decision tree is a decision DAG G = (V,E) whose nodes v ∈ V have in(v) ≤1.

Definition (Random decision DAG) - A random decision DAG is a decision DAG for which the attributes 0_v of each internal node v ∈ V are independently and identically distributed random variables.

Definition (Decision jungles) - A decision jungle J = (G_1,…, G_m) is an ensemble of m random decision DAGs G_1,…, G_m.

The two-class decision jungle algorithm from MS Azure ML creates a two-class classification model using the decision jungle algorithm, which returns an untrained classifier. The model is then trained on a labeled training dataset. Figure 2-10 depicts a decision jungle algorithm which is a recent extension to decision forest algorithm illustrated in figure 2-9 above. A decision jungle consists of an ensemble of decision directed acyclic graphs (Shotton, Nowozin, *et al.*, 2013). The concept of a decision jungle is closely related to that of random forests as proposed by Pavlov (2019). It is also a considerable alternative to random forests.

Shotton, Girshick, *et al.*, (2013) and Shotton, Nowozin, *et al.*, (2013) define a decision jungle algorithm as illustrated in the figure below. Therefore, a set of definitions for decision trees used in classification problems are provided in order to highlight the relation to jungles.

In practice, figure 2-10 shows the classifier for a two-class classification problem using the decision jungle algorithm. The major difference between the two models described in figure 2-9 and figure 2-10 is that in the two-class decision jungle, algorithm nodes may have more than one parent node. The leaf nodes (marked by edges) show the class label (yes/no) for cross-sell candidates that is assigned to a data point (i.e., balance or expenses) passed to the node.

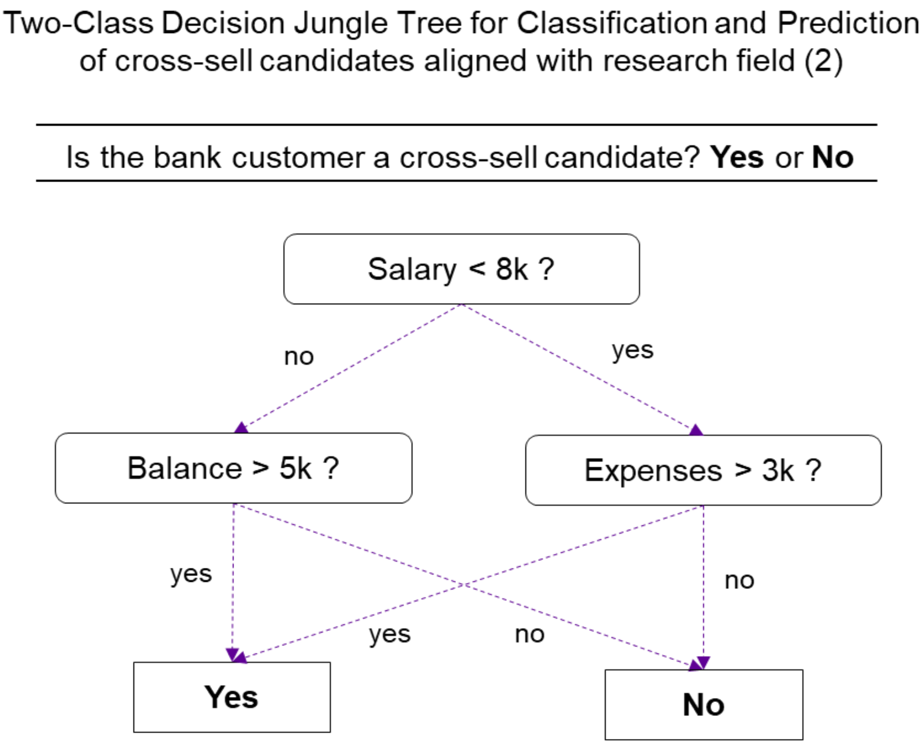

Two-Class Decision Jungle Tree for Classification and Prediction of cross-sell candidates aligned with research field (2)
Is the bank customer a cross-sell candidate? **Yes** or **No**
Salary < 8k ?
no
yes
Balance > 5k ?
Expenses > 3k ?
yes
no
yes
no
Yes
No

Figure 2-10: Visualization of a two-class decision jungle

The data point (i.e., balance or expense) that is supposed to be classified (yes/no) is passed to each DAG individually. Each DAG then votes for the class label (yes/no) it predicts for the data point balance or expense. Finally, the class (yes/no) which received the highest number of votes is chosen as the prediction result for the entire ensemble.

The application of decision jungles has some major advantages. For instance, tree branches can be merged so that a decision DAG results in a lower memory footprint and better generalization performance than a decision tree. These non-parametric models can perform classifications with somewhat longer training time and are resilient in the presence of noisy features.

2.2.2 The two-class locally-deep support vector machine

The MS Azure two-class locally-deep support vector machine creates a two-class, non-linear SVMs classifier that is optimized for efficient prediction. The adjusted algorithm optimizes these predictive models in order to efficiently scale to larger training sets. MS Azure ML uses the kernel function for mapping data points to the feature space. The reason behind this approach is to reduce the time needed for training while maintaining most of the classification accuracy

Jose, Goyal and Aggrwal (2013) define the supervised learning algorithm which learns a non-linear kernel $K(x_i, x_j)$ = $K_L(x_i, x_j)\ K_G(x_i, x_j)$ as the product of a local kernel $K_L = \phi_L^t \phi_L$ and a global kernel $K_G = \phi_G^t \phi_G$ leading to the following prediction function as follows:

$$
\begin{aligned}
y(x) &= sign\left(\sum_i \alpha_i y_i K(x, x_i)\right) \\
&= sign\left(\sum_{ijk} \alpha_i y_i \phi_{Gj}(x_i) \phi_{Gj}(x)\ \phi_{Lk}(x_i) \phi_{Lk}(x)\right) \\
&= sign\left(w^t\ (\phi_G\ (x) \oplus \phi_L(x))\right) \\
&= sign\ (\phi_L^t(x)\ W^t\ \phi_G(x)) \\
&= sign\ (W^t(x)\ \phi_G(x))
\end{aligned}
$$

where $w_k = \sum_i \alpha_i y_i \phi_{Lk}(x_i)\ \phi_G(x_i)$, ϕ_{Lk} denotes dimension *k* of $\phi_L \epsilon R^M$, $W = [w_1, \ldots, w_M]$, $W\ (x) = \ W\phi_L(x)$ and $\oplus$ is the Kronecker product. Thus, the algorithm can be thought of as either learning a single fixed linear classifier $\phi_G \oplus \phi_L$ space or a different classifier for each point in the given global feature space ϕ_G.

However, their paper focuses on the problem of speeding up SVMs given a single global feature. Therefore, a local-deep kernel learning formulation was developed to speed up non-linear SVM prediction by learning deep local features.

Huang *et al*. (2018) define machine learning with maximization (support) of separating margin (vector) as support vector machine learning. The detailed procedure of the two-class support vector machine is described in the previous section. To sum up their characteristics for further comparison, the algorithm tries to find the best separating margin between two classes with a hyperplane (also known as a decision boundary). A kernel is used to measure the similarity between points from each of the classes, and the kernels are applied globally to all points. The closest points to the decision boundary are called support vectors.

The two-class locally-deep SVM is a modification of the two-class SVM where the kernels used for measuring similarity are composed of a local and global kernel. The algorithm tries to learn the local embeddings that are high dimensional, sparse and computationally deep, which allows the two-class SVM to decouple the cost from the calculation from the number of support vectors. Therefore, two-class locally-deep SVM is much faster than a two-class SVM, but might make a sacrifice for accuracy.

The major difference between a two-class SVM and the two-class locally-deep SVM lies first in how the classifiers are calculated. The algorithm creates non-linear classifiers by using a kernel function to maximum-margin hyperplanes. Therefore, a non-linear classification rule is learned, which corresponds to a linear classification rule for the transformed data points to solve the optimization problem. This approach results in a transformed higher-dimensional feature space, which also increases the generalization error of SVMs, although the algorithm still performs well. It is common knowledge that the larger the margin, the lower the generalization error of the classifier.

The cost and performance of calculating the separating hyperplanes based on a two-class SVM is linearly proportional to the training data. Compared to a two-class locally-deep SVM, the cost and performance of calculating the separating hyperplanes does not increase linearly with the training data.

Finally, it is recommended to apply a two-class locally-deep SVM for non-linear datasets and classification problems that need to be optimized. In case of increasing the classification accuracy, the two-class SVM should be a better choice.

The two-class locally-deep SVM is a modification of two-class SVM using the kernel method, which enables us to model higher dimensional, non-linear models. Huang *et al.* (2018) explained that in a non-linear problem, a kernel function could be used to add additional dimensions to the raw data and thus make it a linear problem in the resulting higher dimensional space. The kernel function will do certain calculations much more quickly, which would need computations in high dimensional space. Figure 2-11 below shows the approach by applying the kernel function to separate and transform the data by a non-linear SVM. The main idea of this approach is to combine kernel activations in non-linear ways (Abdullah, Veltkamp and Wiering, 2009).

Jose *et al.* (2013) explained that the kernel functions are used to calculate the scalar product between two data points in a higher dimensional space without explicitly calculating the mapping from the input space to the higher dimensional space. Computing the kernel while going to the higher dimensional space is almost trivial compared to computing the inner product of two feature vectors. The two-class locally-deep SVM from MS Azure ML library is based on the Localized Multiple Kernel Learning approach, and contains multiple layers of SVMs instead of a single adjustable layer of weights (Abdullah, Veltkamp and Wiering, 2009). Thus, the algorithm tries to learn a different kernel for each data point. It is important to mention that the performance of a standard SVM model is affected by the choice of kernel function, among other factors. There is no way to figure out which (parameterized) kernel is the best for our classification problem; however, kernel functions are not flexible. Although, it is not part of current research scope.

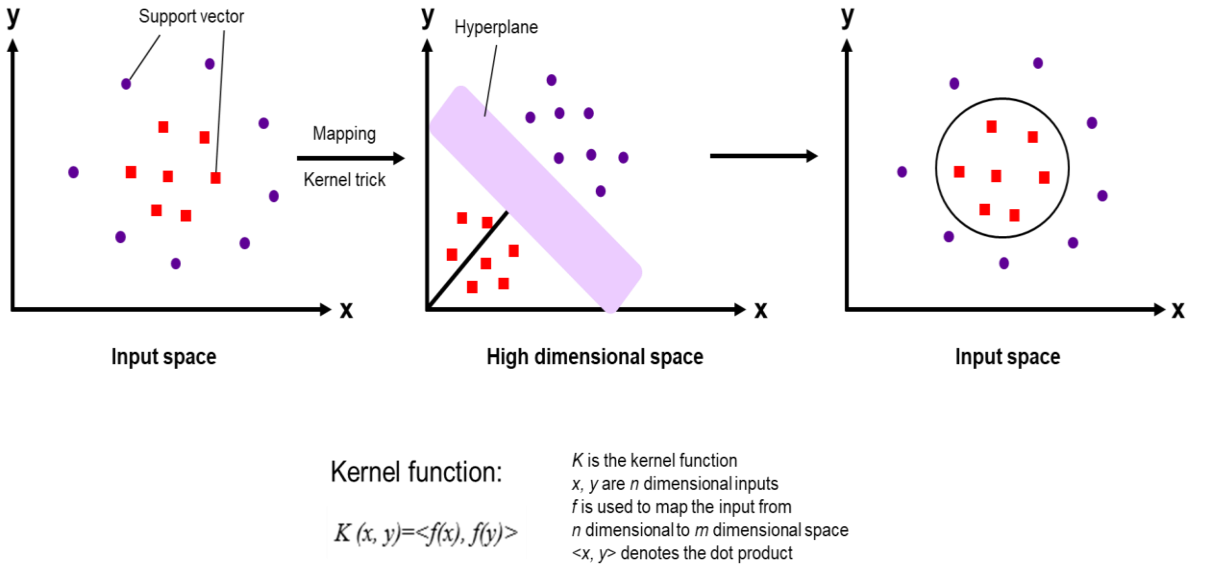


Figure 2-11: Visualization of a two-class locally-deep SVM using kernel function

The current research will choose the best performing machine learning algorithm to solve the defined classification problems through trials. The research starts to experiment with a variety of MS Azure ML algorithms and then experiment to further optimize the performance of the best supervised learning algorithm. Depending on the nature of the classification problem, it is possible that one algorithm is better than the others. An optimal machine learning algorithm can be selected from a fixed set of MS Azure ML algorithms in a statistically rigorous fashion by using a set of performance measurements.

2.3 Critical assessment of applied machine learning algorithms for the 3rd research project - categorized transactional payment behaviour

The section primarily describes the machine learning algorithms used for the data analysis of the third research project. Therefore, basic characteristics of the selected algorithms are presented to increase the understanding of the developed data models to evaluate categorised transactional payment behaviour.

2.3.1 The Apriori algorithm

The section describes the most popular association rule learning algorithm, known as Apriori algorithm, which was used to conduct the research experiments for the third research project. The unsupervised learning method extract rules that best explain observed relationships between variables in the transactional dataset. The algorithm allows the discovery of important and commercially useful associations in large multidimensional datasets, such as the underlying Berka dataset.

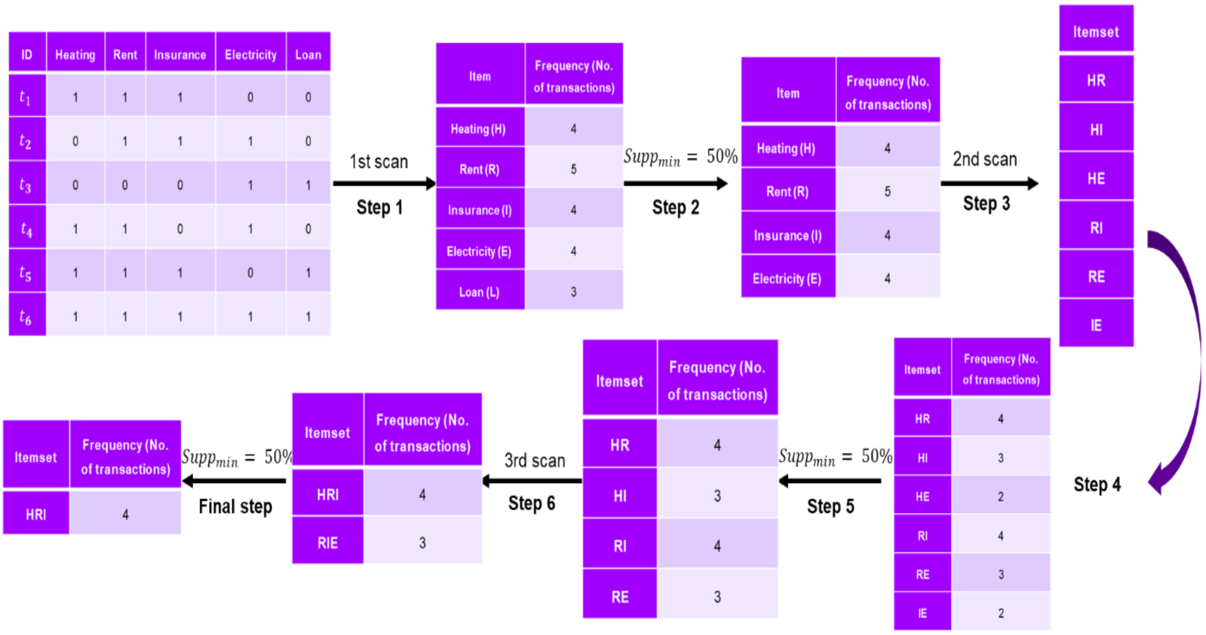
Agrawal, Imieliński and Swami (1993) define the concept of association rule learning through a set of formal definitions as follows:

Definition (Association rule) - Let I = $\{i_1, i_2, i_3,\ldots,i_n\}$ be a set of n payment types known as items and D = $\{t_1, t_2, t_3,\ldots,t_n\}$ be the set of transactions known as database. Every transaction, t_i in D has a unique transaction ID, and it consists of a subset of itemsets in I. A rule can be defined as an implication, X⟶Y where X and Y are subsets of I (X, Y⊆I), and they have no element in common, for instance, X∩Y. X and Y are the antecedent and the consequent of the rule, respectively.

Prerequisite:

(1) All subsets of a frequent itemset must be frequent.

(2) Similarly, for any infrequent itemset, all its supersets must be infrequent too.



ID	Heating	Rent	Insurance	Electricity	Loan
t_1	1	1	1	0	0
t_2	0	1	1	1	0
t_3	0	0	0	1	1
t_4	1	1	0	1	0
t_5	1	1	1	0	1
t_6	1	1	1	1	1

Item	Frequency (No. of transactions)
Heating (H)	4
Rent (R)	5
Insurance (I)	4
Electricity (E)	4
Loan (L)	3

Item	Frequency (No. of transactions)
Heating (H)	4
Rent (R)	5
Insurance (I)	4
Electricity (E)	4

Itemset
HR
HI
HE
RI
RE
IE

Itemset	Frequency (No. of transactions)
HR	4
HI	3
HE	2
RI	4
RE	3
IE	2

Itemset	Frequency (No. of transactions)
HR	4
HI	3
RI	4
RE	3

Itemset	Frequency (No. of transactions)
HRI	4
RIE	3

Itemset	Frequency (No. of transactions)
HRI	4

Figure 2-12: Visualization of frequent itemset generation using Apriori learning algorithm

Figure 2-12 describes a small and practical example of frequent itemset generation applying an Apriori learning algorithm on payment transactions. The set of itemsets, I = {Heating, Rent, Insurance, Electricity, Loan}, consists of six payment transactions.

Each payment transaction is a tuple of 0s (absence of an item) and 1s (presence of an item). On that basis, it is possible to identify multiple interesting and significant rules from a transactional dataset by looking at required measures such as support, confidence and lift:

Definition (Support) - The support of an itemset X, supp(X) is the proportion of transaction in the database in which the item X appears. It describes the popularity of an itemset.

$$supp\ (X) = \frac{\text{Number of transactions in which X appears}}{\text{Total number of transactions}}$$

Following the figure above, supp(heating) = $\frac{4}{6}$ = 0.66667.

Definition (Confidence) - Confidence of a rule is defined as follows:

$$conf\ (X \rightarrow Y) = \frac{supp\ (X \cup\ Y)}{supp\ (X)}$$

It shows the likelihood of a payment type (item) Y being executed when the payment type (item) X is executed. In the example above, the rule {Heating, Rent} → {Insurance} is correct for 75% of the payment transactions. However, this measure takes only the popularity of itemset X into account, and not the popularity of Y. The measure lift will overcome this drawback as follows:

Definition (Lift) - The lift of a rule is defined as:

$$lift\ (X \rightarrow Y) = \frac{supp\ (X \cup\ Y)}{supp\ (X) * supp\ (Y)}$$

It shows the likelihood of the itemset Y being executed when the payment type (item) X is executed, while taking into account the popularity of Y. A lift value of greater than 1 indicates that the itemset Y is likely to be executed with itemset X, while a lift value of less than 1 means that itemset Y is unlikely to be executed if the itemset X is executed.

The general learning process of the Apriori algorithm for frequent itemset generation is illustrated step-by-step in figure 2-12. More detailed explanation about the functionality of the entire Apriori algorithm is provided by Zaki (2001), Fournier-Viger *et al.* (2012), and Anastasiu, Iverson, Smith and Karypis (2014). If the prerequisites in the above example are met, the algorithm works as follows:

The first step creates a frequency table of all the payment types (item) that occur in all six transactions. The second step selects only those important elements for which the support threshold is ≥ 50%, and single payment types (items) that are executed by the bank customers frequently are provided. Next, step 3 brings all possible pairs of the important payment transaction types together without taking care about their order. Step 4 counts the occurrences of each pair in all six transactions. After conducting step 5, only those important itemsets which cross the support threshold of 50% are left. Step 6 consists of a self-join to create a set of x items another rules in order to apply again the threshold rule of ≥ 50% finalize the last step.

$$\boxed{\begin{array}{l} HR \rightarrow I \\ HI \rightarrow R \\ RI \rightarrow H \\ I \rightarrow HR \\ R \rightarrow HI \\ H \rightarrow RI \end{array}}$$

Association Rule Learning Algorithms

Figure 2-13: Visualization of association rule generation using Apriori learning algorithm

According to the example above, figure 2-13 depicts the outcome of association rule generation using Apriori learning algorithm. Detailed explanation of the theory for rule generation using an Apriori algorithm can be found in Hahsler, Grün and Hornik (2005), Hahsler and Chelluboina (2011), Hahsler *et al.* (2011), Anastasiu, Iverson, Smith, Karypis, *et al.* (2014), and Johnson (2018). The general two-step approach for finding association rules efficiently works as follows:

The first step, frequent itemset generation, is about finding all itemsets for which the support is greater than the threshold support, following the process from step 1-6 described in figure 2-12. The learning algorithm finally returns the frequent itemset “HRI.”

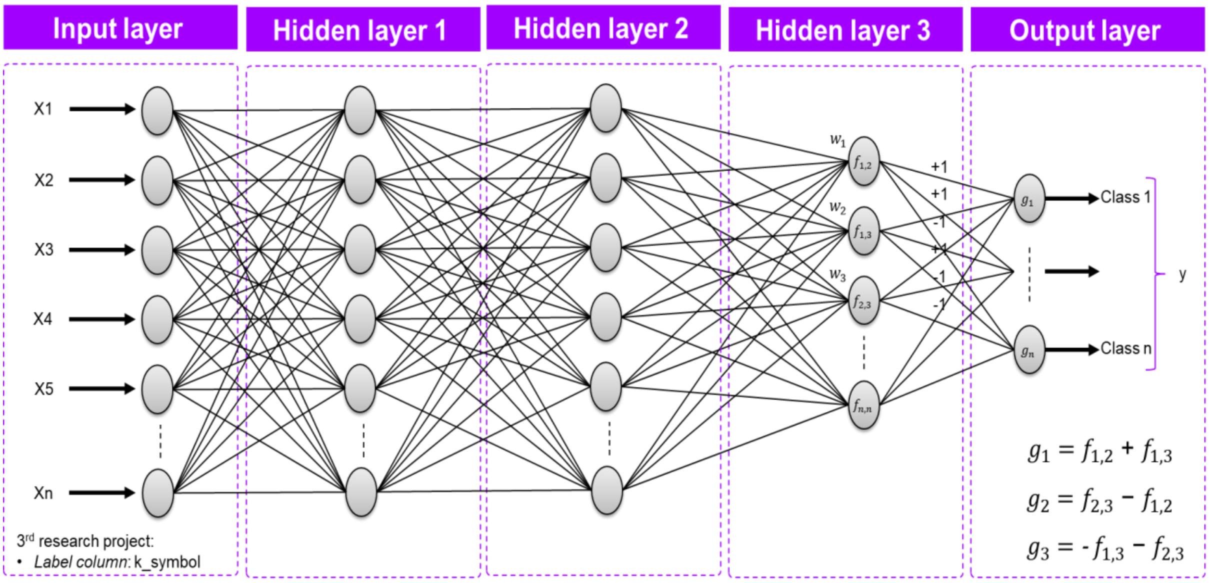
The second step, rule generation, is creating candidate rules from each frequent itemset using the binary partition of frequent itemsets and seeking for the ones with high confidence. The frequent itemset in our example consists of 3 elements (k=3); all possible candidate association rules $(2^k - 2)$ using “HRI” are shown in figure 2-13.

For example, one possible association rule would be {Heating (H), Rent (R)} → {Insurance (I)}, which means if a transaction for heating and rent are executed, banking clients also perform a transaction for insurance.

2.3.2 The deep learning algorithm

A deep learning model can be seen as an extension of the introduced multiclass neural network. Deep learning methods are a popular algorithm that exploit abundant cheap computation. Note, I have separated out deep learning from the introduced multiclass neural networks in the previous section because of the massive growth and popularity in the field of analyzing transactional behaviour. However, current research experiments for the third research project are also concerned with the more classical multiclass neural network method, since I have also used the "neuralnet" package of RStudio to benchmark the research results objectively with the outcomes from an applied deep learning algorithm. The basic characteristics and principles of the applied multiclass neural network are explained in detail in the first section of this chapter.

Deep learning models are concerned with building much larger and more complex neural networks through linear or non-linear relationships, and, as explained in the previous sections, many methods are concerned with supervised learning problems where large datasets contain labeled data. Researchers such as Lecun, Bengio and Hinton (2015) and Schmidhuber (2015) have shown that deep neural networks with many layers can be very effective in complex tasks.



3rd research project:
- *Label column:* k_symbol

Figure 2-14: Visualization of a deep learning algorithm

A deep neural network can be simply defined as a feedforward network with many hidden layers. There is no definition given in the literature as to what deep network really means, but a deep neural network obviously consists of at least two or more hidden layers, which allows a deep structure learning of our classification problem. The algorithm creates a map of neurons and calculates numerical values (weights $w_1, \dots, w_n$) to connections between the neurons. The output returns values between 0 and 1 when multiplying the weights and inputs in the example above through the three illustrated hidden layers.

Figure 2-14 above illustrates that the deep learning algorithm transforms the raw input data through multiple layers in order to progressively extract higher level features more effectively than a multiclass neural network would. Each level in a deep learning architecture learns to transform its input data into a slightly more abstract and composite representation layer one by one. For instance, in analysing transactional behaviour, the raw input is based on the pre-processed Berka dataset, the first representational layer may abstract the relevant features and encode the related payment transaction types given in the label column "k_symbol" to train the model in an unsupervised way. Then, the second and third layers may compose and encode more transactional insights by training the model in a supervised way that fine tunes the input features in order to classify related payment transaction types. Finally, the output layer may predict the next payment transactional type with a calculated probability.

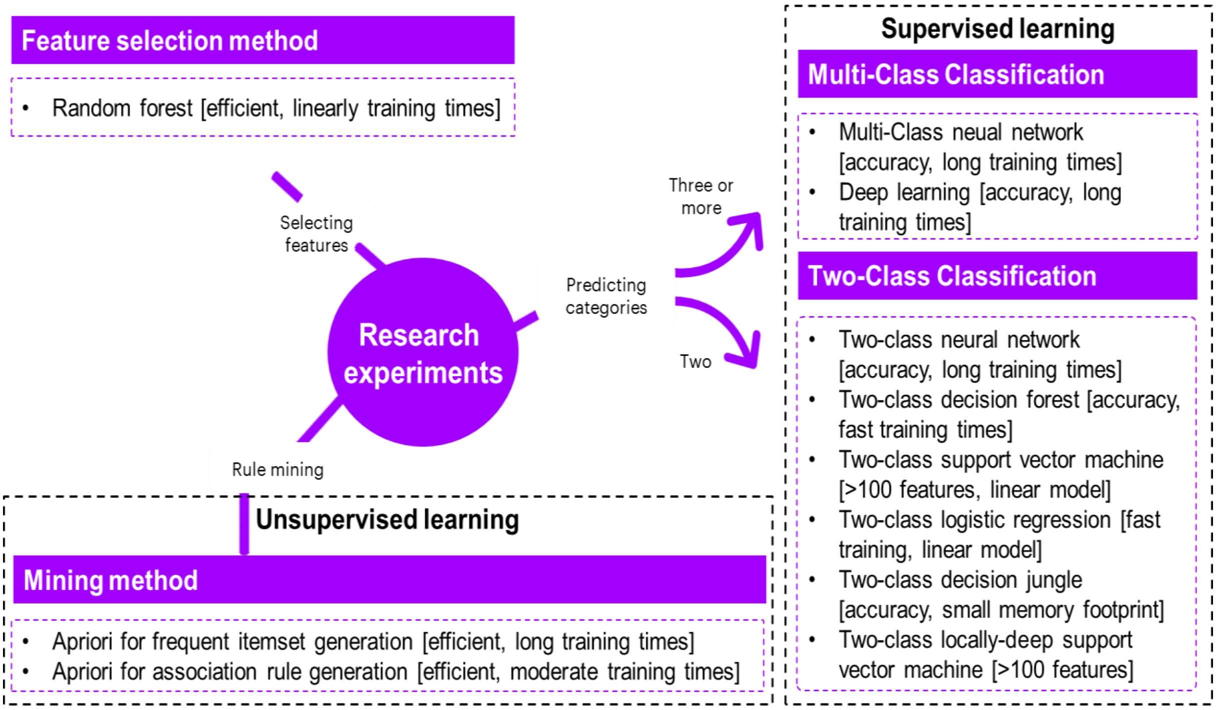
A deep learning process can learn which features from the raw input to optimally place in which hidden layer on its own, through the causal connections between the input and output node. In general, lower layers may identify a couple of related features to roughly characterize the transactional payment behaviour of their banking clients, while higher layers may identify the concepts relevant to analyse their transactional payment behaviour, such as relevant input parameters to predict next payment transactional types accurately. Goodfellow, Bengio and Courville (2015) underline that the accuracy of a deep learning model consistently increases with increasing depth, since deeper networks generalize better.

2.4 Summary of the applied machine learning algorithms

The purpose of this chapter is to show how selected supervised and unsupervised learning algorithms work in the current research context, and to achieve the research objectives for analysing transactional payment behaviour through a variety of research experiments outlined below.

The following chart shows the selected MS Azure ML algorithms to conduct the research experiments for all three research projects. The algorithm selection was also driven by both the nature of the underlying dataset and the research questions I tried to answer: Which algorithm is performing best? How can a cost-sensitive predictive model be built? What kind of behavioural insights are given in the transactional dataset? What are the most popular payment patterns?

The three major research projects also deal with the selection of the most valuable features in the underlying dataset, predict powerful credit scores for banking clients and cross-sell candidates for bank promotions, and finally use rule mining to identify payment patterns in transactional data streams.



Feature selection method

- Random forest [efficient, linearly training times]

Supervised learning

Multi-Class Classification

- Multi-Class neual network [accuracy, long training times]
- Deep learning [accuracy, long training times]

Two-Class Classification

- Two-class neural network [accuracy, long training times]
- Two-class decision forest [accuracy, fast training times]
- Two-class support vector machine [>100 features, linear model]
- Two-class logistic regression [fast training, linear model]
- Two-class decision jungle [accuracy, small memory footprint]
- Two-class locally-deep support vector machine [>100 features]

Unsupervised learning

Mining method

- Apriori for frequent itemset generation [efficient, long training times]
- Apriori for association rule generation [efficient, moderate training times]

Figure 2-15: Overview of applied machine learning algorithm for various research experiments

Table 2-1 below summarizes a range of applied machine learning algorithms and their general characteristics. Detailed configuration parameters of every applied supervised learning algorithm from MS Azure ML library are provided in Appendix D.7. Different aspects, such as accuracy, training time, linearity, number of parameters and features, should be considered when selecting and applying these algorithms. However, the following characteristics should not be ignored when interpreting the predictive results of the supervised learning models.

Algorithm	Accuracy	Training time	Linearity	Parameters	Notes
Multiclass Neural Network	●			9	Customization is possible
Two-Class Neural Network	●			9	Customization is possible
Two-Class Decision Forest	●	○			
Two-Class Support Vector Machine		○	●	5	Good for large feature sets
Two-Class Logistic Regression		●	●	5	
Two-Class Decision Jungle	●	○		6	Low memory footprint
Two-Class Locally-Deep Support Vector Machines	○			8	Good for large feature sets
Deep Learning	●		●		Good for large feature sets
Random Forest		●	●		Good for large feature sets

● - shows excellent accuracy, fast training times, and the use of linearity

○ - shows good accuracy and moderate training times

Table 2-1: Overview of the general properties of all applied supervised learning algorithms

Accuracy is one of the key performance indicators to choose the best performing algorithm. Getting the most accurate credit score, cross-sell candidate results or next payment transaction category predictions tend to overfit the model and may also increase the processing time.

Training time of a model is often closely tied to accuracy, so that both characteristics typically accompany each other. As the conducted research experiments have no time limit and the current Berka dataset is manageably large, the prediction accuracy should be in the foreground to answer the research questions.

Linearity in our current Berka dataset can have an impact on the performance of the developed models. Although two-class logistic regression and two-class support vector machine models tend to be algorithmically simple and fast to train, a non-linear class boundary relying on a linear classification algorithm can bring accuracy down.

Spanning the right parameter settings of an algorithm can affect the error tolerance (accuracy) or number of iterations (training time). A high number of parameters indicates a great flexibility as well as longer training times of an algorithm. The right combination of the parameter settings often results in very good accuracy results. It is interesting to check whether the multiclass neural network and the two-class locally-deep support vector machine algorithm also deliver accurate prediction results.

Lastly, a large number of features in the dataset may have a negative impact on some learning models, but two-class support vector machines are well suited and can still provide accurate predictions.

In addition, a cost-sensitive data model is being developed in the first and second field of research. Therefore, I have used the random forest algorithm presented in the previous section to apply RStudio’s varImp() function for identifying most important variables in the pre-processed dataset of the credit scoring and cross-selling case. The calculation of a modified correlation matrix, known as flattenCorrMatrix in RStudio, is also an important step in the development of a cost-sensitive data model for the first two research projects. Details and the basic characteristics of the flatten correlation matrix applied in RStudio are explained in Chapter 5 - research results.

Finally, Table 2-2 below summaries the used unsupervised learning algorithms and their general characteristics to conduct the research experiments for transactional behaviour analysis across payment streams.

Algorithm	Accuracy	Training time	Linearity	Parameters	Notes
Apriori for frequent itemset generation		○			Computationally expensive for large itemsets or low support threshold
Apriori for association rule generation		○			Computationally expensive for finding large number of candidate rules or for support calculation

● - shows excellent accuracy, fast training times, and the use of linearity

○ - shows good accuracy and moderate training times

Table 2-2: Overview of the general properties of all applied unsupervised learning algorithms

Chapter 3

3. Research Background and Literature Review

This chapter provides a comprehensive literature review in general and describes the research background. Therefore, the literature review will assign the research aims and objectives, starting from the general and proceeding to the specific research projects, whereby the research will be placed in the context of transactional payment behaviour. The following sections also set out why the research needs a distinctive systematic literature review to ultimately identify the research gaps. As part of the research approach, subsequent sections will describe the developed search strategy for the literature review and explain the process to ensure that the research context has been critically framed. Finally, the chapter will justify the research scope, as well as how the research will fill these important gaps in the research.

The following five elements of the systematic literature review framework for the research study are described in figure 3-1 below. Finally, the presented synthesis framework supports the conducted research to cover significant research contributions related to the defined research scope in the area of transactional payment behaviour and discover important research gaps.

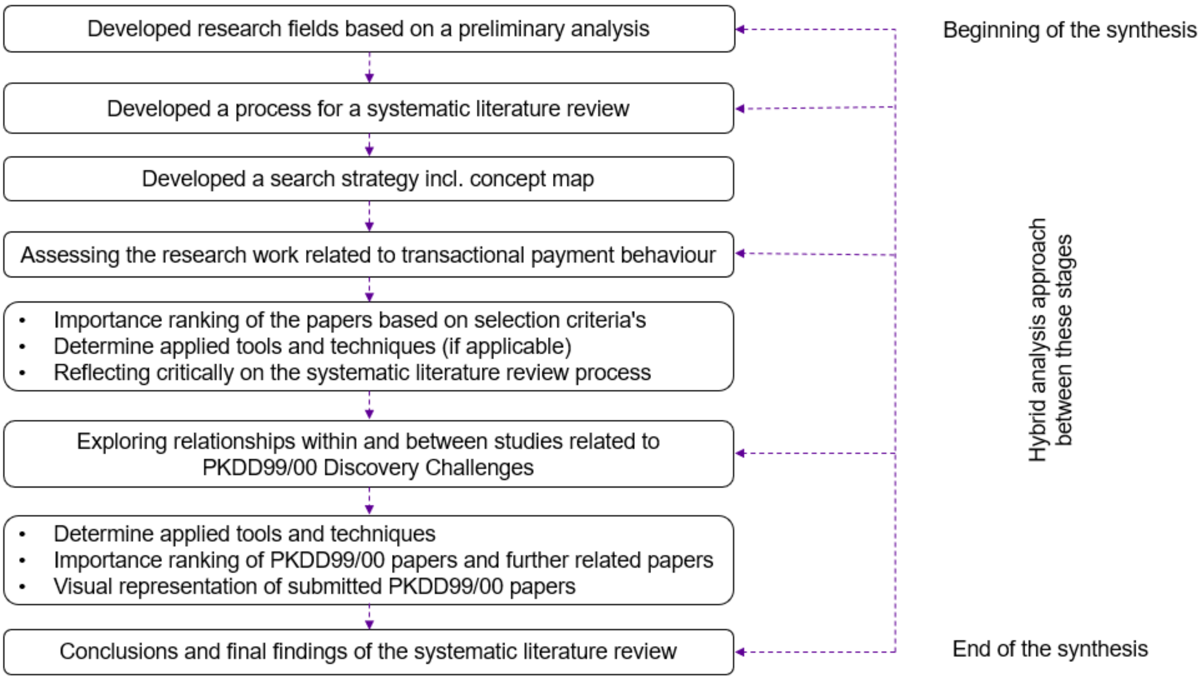

Developed research fields based on a preliminary analysis
Developed a process for a systematic literature review
Developed a search strategy incl. concept map
Assessing the research work related to transactional payment behaviour
• Importance ranking of the papers based on selection criteria's
• Determine applied tools and techniques (if applicable)
• Reflecting critically on the systematic literature review process
Exploring relationships within and between studies related to PKDD99/00 Discovery Challenges
• Determine applied tools and techniques
• Importance ranking of PKDD99/00 papers and further related papers
• Visual representation of submitted PKDD99/00 papers
Conclusions and final findings of the systematic literature review
Beginning of the synthesis
Hybrid analysis approach between these stages
End of the synthesis

Figure 3-1: Synthesis framework applied for the systematic literature review

3.1. The process of systematic literature review and search strategy

This section describes the procedure that the conducted research follows to perform an extensive systematic literature review on the current state of research for transactional payment behaviour and highlights the search strategy developed to identify the most significant literature relating to the research objectives introduced in the chapter 1.

A significant problem with any literature review in the field of behavioural analysis is that there is no consensus about what transactional payment behaviour or customer behaviour predictability means in transactional category terms, as well as expenses for banking clients. If it involves systematically excluding industries, stocks or marketing etc., the theory is clear that this reduces the payment transaction universe and therefore is likely to have applied machine learning algorithms in the context of transactional behaviour returning cross-selling products or calculated credit scores. However, in assessing the performance of various credit scoring or cross-selling algorithms to identify how well each applied classification algorithm performs in terms of its accuracy and especially developing cost-sensitive data models, only limited literature can be found. Nothing interesting can be found in the literature for the second research project in case of developing a cost-sensitive data model to efficiently forecast cross-sell opportunities. The latter research project of mining categorised payments from a data science perspective is a largely uncharted field of research.

A systematic review of existing literature on payment transaction analysis methods within the financial services is performed to (a) assess the content and procedure by which the statistical methods are used to specify the current research gap in the literature, (b) evaluate the use of machine learning algorithms in payment transactional ecosystems, (c) evaluate the effectiveness of advanced forecasting and predictive methods and (d) explore the effective application of data science in changing customer behaviour identification within the financial services industries. For instance, this aims to explore using data science techniques for effective decision-making within the banking industries to approach banking clients at the right time in their life stage with the suitable business proposal (i.e. home loans, vehicle financing, car insurance, etc.). Mohan and M. (2016) highlighted that accurate analysis – which is the backbone of accurate decision-making – can lead to operational efficiency and cost and risk reduction within a bank.

The systematic review also considers the research questions, especially those relating the effectiveness of existing exploratory analysis methodologies in terms of using (a) transactional data in customer behaviour prediction, (b) current descriptive analysis methods, their suitability and robustness, (c) the use of (un)categorised payments data in predictive analytics, (d) whether using categorised payments data can be more effective and robust in changing customer behaviour detection and (e) the use of advanced forecast and predictive models in case of developing a cost-sensitive data model for credit applicants and cross-sell promotions.

The literature review is generally tailored based on a two-step search strategy approach. The first step within this systematic process is dedicated to a broader literature review of transactional payment behaviour along the following process. Databases for science and engineering are searched for major publications regarding the use of payment transaction datasets to identify changing customer behaviour in terms of categorised payment transaction streams, creditworthiness and product recommendations within banking industries to sort out relevant articles supporting the research hypothesis. Second, the research will examine all research papers published around the PKDD Discovery Challenges based on the sponsored Berka dataset. Subsequently, this work will take additionally a closer look at related work to the papers published around the Berka dataset to ensure the distinctiveness of current piece of work. The databases and digital libraries include the ACM Portal, Elsevier Science (Academic Science Research), Emerald (Academic Research), IEEE (Computer Science), IIM Management Science, INFORMS (Informs online), John Wiley & Sons Publications, Sage Publications and ScienceDirect (Academic Science Research) databases.

Online journals for science and engineering are searched for publications about changing customer behaviour identification using payment transactions data. Relevant journals and published papers (conference proceedings, technical reports or archival journals) as well as bibliographies of other related articles are also searched. Reference lists of the related journals will also be scanned to find appropriate studies that may not have been provided in the aforementioned studies. Only peer-reviewed journals from 1983 to 2019 are considered.

Search terms are constructed using the methods described in the Cochrane Handbook for Systematic Reviews of Interventions (Higgins and Green, 2011). Search strings are applied by combining the keywords (payments and transaction data) with the assessment terms (identify, investigate, predict, system, model), result-related terms (i.e. changing behaviour, mining behaviour) and then industry-type terms (banking, future banking, digital banking) and other related terms (categorisation, credit score, cross-selling, pattern). The search was based on the descriptors above that occur in the title or abstract of the paper. Wild cards are also used to include international spelling variations.

The final search string is proposed as (payments and transaction data) AND (identify OR investigate*OR categorise* OR predict* OR analysis OR assess OR forecast* OR model* OR credit score* OR cross-selling*) AND (changing behaviour OR mining behaviour) AND (banking OR future banking OR digital banking) AND (categorisation OR credit score OR cross-selling OR pattern). Where this search string is found to be less effective, alternative search strings are investigated. Only peer-reviewed studies including the adoption of machine learning algorithms are considered.

Although most research in the field has been conducted with a banking or digital banking inclination, this study focuses on data science. Hence, only research involving the use of data science and statistical techniques is considered. Any studies involving general banking industries and general customer behaviour are excluded.

The selection criterion was used to eliminate journals whose titles or abstracts do not meet the objectives of the research. All titles that likely relate to the research aims and objectives and especially the scope of research are selected and assessed accordingly.

Figure 3-2 below documents the quantitative searches in different database resources by using the introduced search terms to develop a detailed concept map for the literature analysis. This unique approach supports the conducted research by more efficiently structuring and presenting the knowledge about transactional payment behaviour. It allows current research to eliminate duplicates through the entire literature review and identify highly-related papers with respect to the defined research questions. For this purpose, I have also followed the instructions described by (Brereton *et al.*, 2007). With respect to the research questions, the study converts the need for interesting information, looks for the best evidence and critically assesses the key outcomes given in the literature. The different search terms result in more than 300 studies that I have initially flagged as significant to the research fields, although after employing the inclusion and/or exclusion criteria as described in the previous paragraphs the following concept map was developed. Overall, 54 papers (see tabulated results matrix in the appendix A) were left in the set of important papers (high, medium and low).

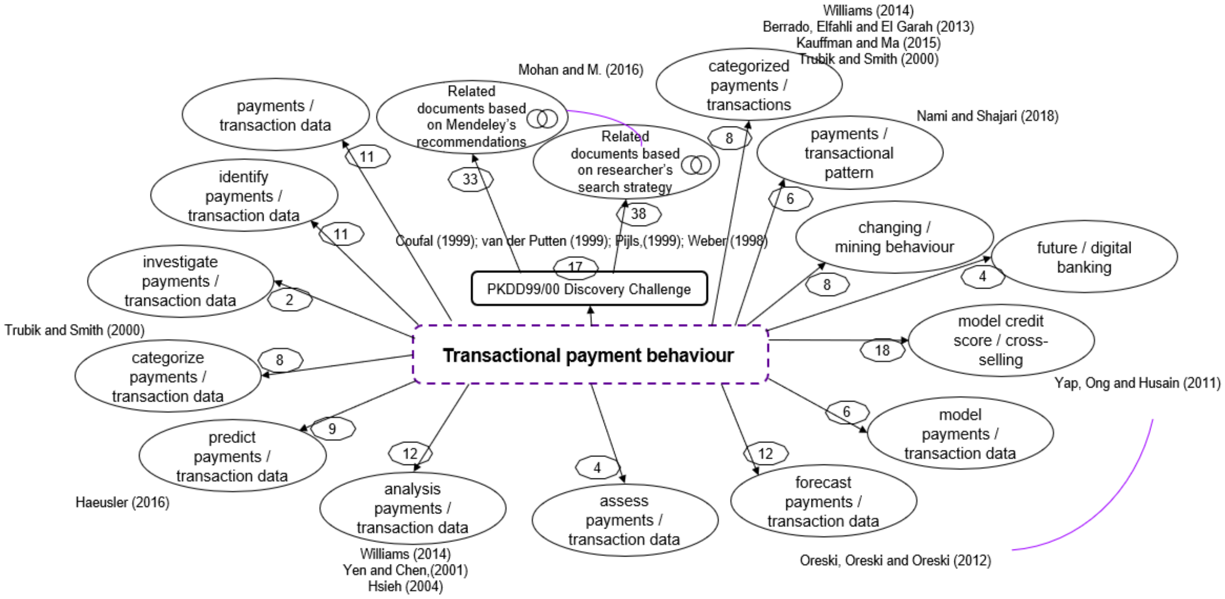

Williams (2014)
Berrado, Elfahli and El Garah (2013)
Kauffman and Ma (2015)
Trubik and Smith (2000)
Mohan and M. (2016)
categorized payments / transactions
Nami and Shajari (2018)
payments / transaction data
Related documents based on Mendeley's recommendations
Related documents based on researcher's search strategy
payments / transactional pattern
11
33
8
identify payments / transaction data
38
6
11
Coufal (1999); van der Putten (1999); Pijls,(1999); Weber (1998)
changing / mining behaviour
future / digital banking
investigate payments / transaction data
17
PKDD99/00 Discovery Challenge
8
4
2
Trubik and Smith (2000)
model credit score / cross-selling
categorize payments / transaction data
8
Transactional payment behaviour
18
Yap, Ong and Husain (2011)
6
model payments / transaction data
9
predict payments / transaction data
12
12
4
Haeusler (2016)
analysis payments / transaction data
assess payments / transaction data
forecast payments / transaction data
Williams (2014)
Yen and Chen,(2001)
Hsieh (2004)
Oreski, Oreski and Oreski (2012)

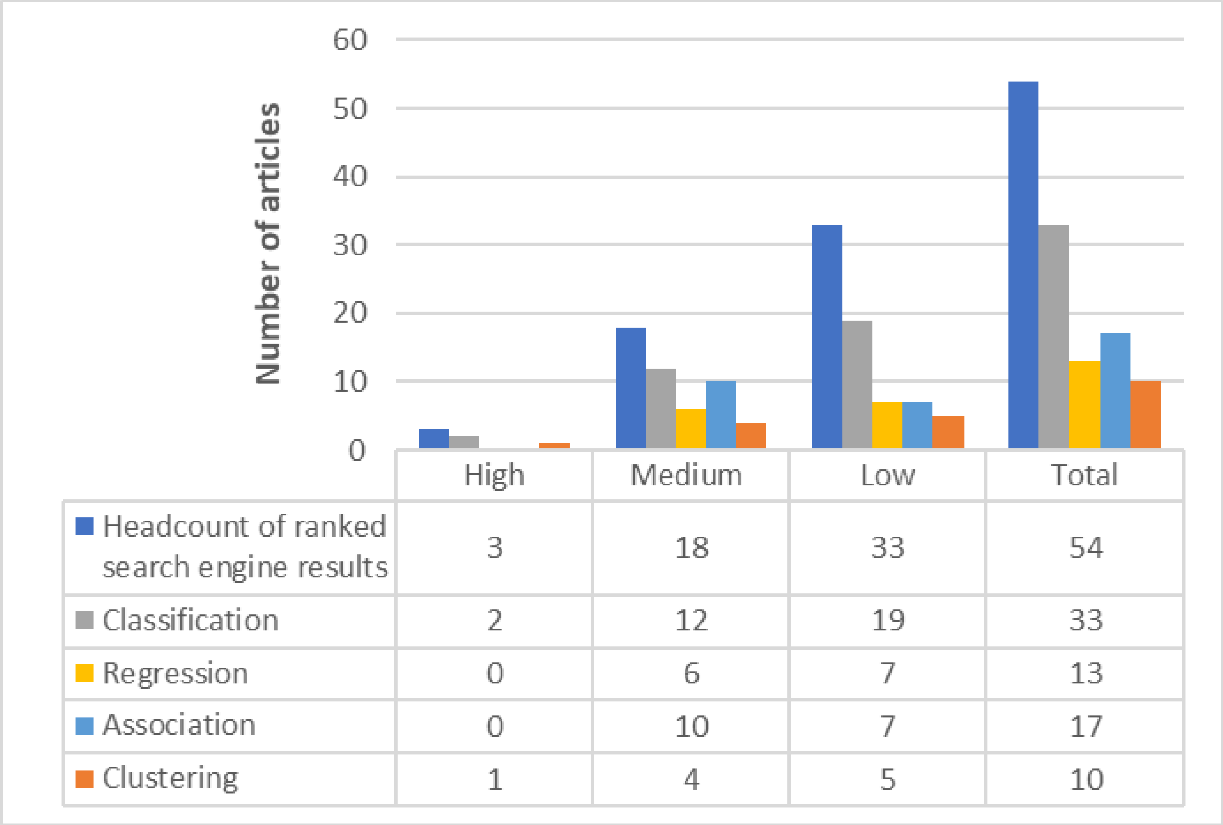
Figure 3-2: Concept map - transactional payment behaviour

As can be seen with the visualised concept map, the search of transactional payment behaviour resulted in more than 130 papers with high and medium importance that were published between 1983 and 2019. It reflects the amount of research conducted in each search string of the synthesis framework applied for the systematic literature review and provides the context about what the research study wants to evaluate relating to the pre-defined research questions. In addition, a detailed timeline for significant research papers is provided in the appendix B-1 detailed and accurate investigation is also mapped into the three current research projects.

Finally, the search results were prepared by various tabulating results matrices aligned with the two-step search strategy approach. Detailed information about the primary sources that were potentially significant as well as targeting the underlying research projects were stored in a matrix that tabulates the results by the publishing author and year, used mining step(s), applied method(s), deployed tool(s) (if applicable), key outcome(s) and importance rank.

3.2. Usage of supervised and unsupervised learning methods in transactional payment datasets

This section will provide an overview of the machine learning algorithm used in the context of analysing transactional payment behaviour in general. Therefore, the following tabulated chart summarises the key results conducted through the systematic literature review process. Note that the summary overview of applied machine learning techniques only targets papers for which I have created a rank by importance with respect to the current fields of research.



	High	Medium	Low	Total
■ Headcount of ranked search engine results	3	18	33	54
■ Classification	2	12	19	33
■ Regression	0	6	7	13
■ Association	0	10	7	17
■ Clustering	1	4	5	10

Figure 3-3: Filtering of machine learning techniques applied to transactional payments

Figure 3-3 above illustrates a generic overview of what kinds of supervised and unsupervised learning methods are used by financial institutions as well as other institutions with a payment ecosystem integrated. All highly-ranked search engine results primarily apply classification techniques such as k-nearest neighbour neural network to predict scores for credit card ownership (van der Putten, 1999) or a complex neural network for credit scoring in retailing (Oreski, Oreski and Oreski, 2012) as well as clustering techniques such as k-mean clustering to classify previous customer transactions (Mohan, L. and M., 2016).

The majority of machine learning algorithms applied belong to the category of supervised rather than unsupervised learning methods. The most popular supervised learning methods applied to transactional payment datasets are classification algorithms, followed by association algorithms from the family of unsupervised learning methods. In comparison with unsupervised learning methods, clustering algorithms are less popular methods, although this does not mean that this method is not suitable for our current fields of research. For example, I have the aim to cluster categorised payment transactions to coherently identify transactional patterns linked to general customer payment behaviour.

However, there is a clear observable tendency between the medium- and low-ranked papers that association rule mining is becoming increasingly popular and relevant in case of transactional payment behaviour analysis. For instance, the methodology of mining frequent itemsets (Pijls, 1999), discover interesting rules (Sołdacki and Protaziuk, 2013), expoloring association rules in large transactions (Agrawal et al., 1996; Arvind and Badhe, 2016), cluster-related transactions (Zaki, M. J. ; Parthasarathy, S. ; Ogihara, M.; Li, 1997) and developing a graph-based approach for large transaction mining (Yen and Chen, 2001) should therefore be considered in the following research design and methodology chapter. In addition, no relevant literature was found through the systematic literature review applying a deep-learning neural network algorithm to a categorised transactional dataset. Most classification algorithms such as neural networks are used for credit card usage behavioural analysis (Tsai, 2007, 2008), behavioural scoring modelling (Hsieh, 2004), to detect and predict fraudulent transactions in credit card transactions of a bank (Brause, Langsdorf and Hepp, 1999), to predict fraudulent attacks in e-Banking (Malekpour, Khademi and Minae-bidgoli, 2016), evaluate credit risks (Yu, Wang and Lai, 2008a) as well as conducting credit risk assessments (Bekhet and Eletter, 2014), to model credit scorings in retail transactions (Oreski, Oreski and Oreski, 2012), predict late transactional payments (Gschwind, 2007) and ultimately model customer cross-selling for direct marketing initiatives (Liu and Cai, 2008).

Finally, Mohan and M. (2016) emphasises that there have "been several classification models developed both in literature and academia to categorize banking customers based on their transactions performed". Regarding the underlying research objectives, the selected mining methods should be closely aligned with the objectives of the data analysis conducted. In this context, an overview of all relevant and peer-reviewed paper as tabulated is provided in the appendix tables A-1, A-2, A-3 and A-4. Regarding the three different research projects and their research aims, the research probably needs to combine the usage of supervised and unsupervised learning methods to answer the developed research questions. Han and Kamber (2006) noted that several data mining techniques can be proposed, although all such algorithms depend on the quality of input data to result in accurately classifying customers' transactional behaviour. As the amount and the quality of training as well as test data increases, the accuracy of classification will also increase. An efficient data science-driven approach is required to identify the best-performing machine learning algorithm in case of transactional payment behaviour analysis, especially for the first two fields of research.

The following section is the next building block within the systematic literature review process, providing an overview of work related to transactional payment behaviour analysis in general, albeit still with respect to the current research objectives.

3.3. Research projects related to transactional payment behaviour analysis

The section reviews the research papers, studies and previous literature published related to the subject of transactional payment behaviour. The systematic literature analysis is based on the high-level concept map, which frames the research topic by illuminating research conducted regarding transactional payment behaviour in transactional datasets in general. Thus, the research has focused the examination around the search terms presented in the previous section. Accordingly, the following sections will provide an overview of related research work in different areas around transactional payment behaviours. In addition, a brief overview of relevant referenced literature used in the research is provided in the appendix A, which includes a timeline for transactional payment behaviour topics addressed between 1993 and 2019 differing in importance. Details can be seen in the appendix, table B-1.

The subsequent paragraphs present the examination results of various pieces of work related to the search term of ‘payment transaction’.

The report by Williams (2014) evaluates a Medicare 2011 transaction dataset using data mining techniques such as the Naïve Bayes classification algorithm to predict future charges or payments in Medicare based on the available financial transaction values. The rationale behind this work was also to better understand the drivers of Medicare transactions at a programmatic level. The average model accuracy is up to 88%, although the researcher emphasises that a certain transaction data size from first and second payers must be given to solidify the model accuracy statement.

Berrado, Elfahli and El Garah (2013) scrutinise Moroccans’ behaviour towards the adoption of mobile payments based on a technology acceptance model. The objective of their work is to investigate the key influencing factors through an empirical analysis by using modern data mining techniques like random forests, association rules and multi-dimensional correlation analysis to determine customers’ behavioural attitudes towards mobile payment adoption in Morocco. For instance, the applied association rule method shows that ease of use, usefulness, risk perception and transaction fees drive people's intention to adopt mobile payments.

Patze-Cornell, Tagaras and Eisenhardt (1990) propose a stochastic model to monitor the cash flow and make short term decisions to optimise the management of financial risks in case of a liquidity squeeze. They proposed a model of cash flow management that provides a systematic basis to construct a real-time cash flow warning system.

The article by Kauffman and Ma (2015) provides an overview of the latest research on payments and credit cards, debit cards and other forms of digital money addressed in the global fintech revolution. The referenced papers are associated with prevailing financial services operations around the world. The authors share current perspectives on changing patterns in the use of internet banking, cash, credit and debit cards in different countries and regions of the world. Although analysing transactional payment behavioural patterns was not a prime concern, their interesting work and thoughts provide a useful overview of the current state of research.

The following paragraphs provide the literature analysis results of various papers related to the search term of 'categorise payment transaction'.

Trubik and Smith (2000) developed a model of customer defection in the Australian bank industry. In their study, they examined customer profitability and customer channel preferences to identify those at risk of leaving based on the entire customer database. The authors look at customer transactional behaviour with savings accounts combined with an offensive and defensive strategy to secure their client retention. The initial objective of the study is to categorise customers' accounts as "closed" or "open" in terms of being active or inactive with the bank. Therefore, they model specific behaviours of customers based on a range of variable (i.e. time at bank, age, major channel, closed an account, etc.) to predict customer defection.

Butler and Butler (2015) designed a web-based survey to evaluate consumers' online banking behaviour based on a risk-profiling. Key findings for a safer online banking environment include using different authentication methods in accordance with customers' browser (before identification) and demographic data (after identification).

The subsequent paragraphs summarise the literature results resulting from the search term 'forecast payment transactions'.

Oreski, Oreski and Oreski (2012) showcase a hybrid system with genetic algorithm and artificial neural networks and their application for retail credit risk assessments. One of their research goals was to evaluate the extent to which the entire dataset – owned by a Croatian bank – can be a useful basis for predicting the credit quality of the borrower. Therefore, the authors propose several feature selection techniques to find an optimum feature subset that enhances the classification accuracy of neural network classifiers. Their experiments show that the hybrid system with genetic algorithm is competitive and can be used as a feature selection technique to discover the most significant features in determining the risk of default. The researchers also recommended to assess the accuracy of other artificial intelligence methods than enhancing only the classification accuracy of neural network classifiers. The current research will close this gap by examining various combinations of the input data in terms of their contribution to correct classification of the credit applicant from the aspect of credit risks, as well as developing a cost-sensitive data model for the best-performing algorithm. In this context, the authors stress the importance of the prediction accuracy of a good or bad credit applicant, which can be improved by a good selection of input variables, applying the best mining methods and finally combining the results of different classification methods. Note that the study by Hand and Henley (1998) provides a useful overview of existing classification methods in credit scoring. However, the focus of their research study was primarily on the feature selection process and how their results can improve the classification accuracy of one single classification algorithm, namely the neural network.

The study by Yap, Ong and Husain (2011) uses data mining techniques to improve the process of assessing credit worthiness during the credit evaluation process. The management of a recreational club aims to identify potential defaulters and seeks to predict late payments. In their experiments, the authors compared the classification performance of a credit scorecard model (27.9%), logistic regression model (28.8%) and decision tree model (28.1%) to assess their error rates based on only type I / II errors.

Abdou, Pointon and El-Masry (2008) investigate the ability of neural nets such as probabilistic neural nets and multi-layer feed-forward nets as well as conventional techniques such as discriminant analysis, probit analysis and logistic regression in assessing the credit risk in an Egyptian bank's personal loans dataset. The research outcome revealed that the neural nets models provide a better average correct classification rate based on type I / II errors compared with the other data mining techniques. Again, no further performance criteria such as AUC or precision and recall were included in their evaluation.

The study by Gschwind (2007) demonstrates that late payments in tenant behaviour can be predicted based on basic tenant data, account receivables and government-published data. Their work underlines that using data mining techniques is better than a dartboard approach whenever transactional payment behaviour is analysed.

Huang, Chen and Wang (2007) construct a hybrid SVM-based credit scoring model to evaluate the applicant's credit score based on their input features. The performance of the new proposed model was compared with neural networks, genetic programming and decision tree classifiers. One of the key results is that the suggested support vector machines classifier achieved an identical classificatory accuracy with relatively few input features. For instance, Vanneschi *et al.* (2018) also use genetic programming for their developed model to predict the probability of defaulting on transactional payments in e-Commerce.

In another study by Huang, Chen and Wang (2007), they also suggested a credit scoring model based on support vector machines and assess their classification accuracy against neural networks, genetic programming and decision tree classifiers. The experimental results show that support vector machines are a promising method compared with existing data mining methods. However, their model accuracy does not exceed 87%. A good classification performance might be achieved by optimising the parameters and feature subset. The same can be said about the research study conducted by Chye, Chin and Peng (2004), in which various data mining techniques were introduced, including their benefits, applications and limitations. The authors highlight that the prediction accuracy of a decision tree model (74.2%) is best, followed by neural network (73.4%) and logistic regression models (71.1%).

The short paper by Vojtek and Kocenda (2006) introduces the most common credit scoring methods such as linear discriminant analysis, logit analysis, k-nearest neighbour classifiers, classification and regression trees and neural networks, and presents some indicators (demographic, financial, employment and behavioural) that are typically important in retail credit scoring models.

The search terms ‘identify’ as well as ‘investigate transactional payment behaviour’ did not yield significant research results. Nonetheless, the search term ‘model credit score’ revealed the following interesting peer-reviewed research results.

The paper by Bekhet and Eletter (2014) proposes two credit scoring models using a logistic regression model and radial basis function to support loan decisions for Jordanian commercial banks. The analysis results indicate that the logistic regression model performed slightly better than the radial basis function model in terms of the overall accuracy rate, measured only by type I / II errors. However, their study provides insights into the potential and limitations of using both quantitative models and does truly reveal which of the two algorithms performs best or how a cost-sensitive data model can be built.

The paper by Bijak and Thomas (2012) targeted the question whether segmentation will improve the model performance in credit scoring. The authors applied a two-step approach in which logistic regression follows classification and regression trees or chi-squared automatic interaction detection trees. In addition, the research applied a new, simultaneous method, in which both segmentation and scorecards are optimised at the same time, through logistic trees with unbiased selection. The study then measured the model performance and compared the results in terms of whether there is an improvement due to the segmentation methods used. It emerged that segmentation does not always improve model performance in credit scoring.

The major goal of the paper by Hooman *et al.*, (2016) is to provide a complete literature survey of applied data mining methods in credit scoring. In general, their findings support researchers to identify the most suitable methods. The advantages and limitations of various methods are presented, such as discriminant analysis, logistic regression, k-nearest neighbour, Bayesian classifiers, decision trees, neural networks, survival analysis, fuzzy rule-based systems, support vector machines and various hybrid methods.

Zhang *et al.* (2010) propose a vertical bagging decision trees model for credit scoring. The new bagging method obtains an aggregation of classifiers by combining predictive attributes. The model performance is outstanding in terms of prediction classification accuracy when using the aggregation strategies that combine collecting individual machine learning models rather than building a single high-capacity model such as the C4.5 algorithm, neural network and support vector machine.

The search term ‘model cross-selling’ results in a range of interesting research papers, albeit what do not directly target the specific research scope. For instance, the paper by Swain and Mohapatra (2013) explores the cross-selling techniques adopted by banks in India. Therefore, a comparative study was conducted from a banker’s perspective, in which various elements of successful cross-selling practices were investigated. Accordingly, no specific data science approaches were implemented to understand how data models can benefit cross-selling practices. The same can be stated about the study by Karadag and Akman (2015), who examined the role of SME banking by analysing business models and strategies in both international and local contexts instead of developing a data-driven approach to enhance cross-selling opportunities.

Liu and Cai (2008) investigate selected customer demographic data and study the relation between the variables and customers’ cross-selling potential based on different data mining techniques. The authors set up a cross-selling model and developed a recommender purchase system based on a counter propagation network. The key results of the proposed model are the successful prediction of customer cross-selling potential according to customer demography data, namely age, income, gender, and educational level.

The article by Hong and Lee (2014) examines the relationship between cross-buying determinants and customers' cross-buying intentions in East Asia. Through face-to-face interviews with professionals, senior managers and academics, they evaluate the cultural impact on customers' cross-buying behaviour, particularly in South Korea and Taiwan. Their study ascertained that 'perceived value', 'trust', 'image' and 'satisfaction' are key determinants of customers' cross-buying intentions on bank assurance, among which 'trust' and 'satisfaction' are significantly influenced by 'collectivism'.

The article by Qiu, Wang and Bi (2011) built a hybrid classifier that integrates logistic regression, with decision stump and voting feature intervals to score credit card customers for cross-sell opportunities in home loans. Their analysis is based on a mixed resampling approach, which results in an AUC value of 0.684 on the testing set. However, their study is not focused on cross-sell opportunities in credit card promotions and it seems that the accuracy of the mining process can be improved to achieve the best performance when applying sophisticated machine learning algorithms as well as advanced feature selection methods through the model-building process.

The study by Schutte, Van Der Merwe and Reyneke (2017) proposes an integrated data mining and customer behaviour scoring model based on transaction history, recency, frequency and monetary background. They analysed customers' mobile banking usage behaviour to effectively target marketing strategy assignments. As a result, the study provides evidence that customer segmentation is a useful approach to identify the transactional behaviour of bank customers and customise certain cross-sell opportunities that best suit these behaviours.

The following paragraphs provide an overview of the most interesting papers when applying the search term 'model payment transactional pattern' to various search engines. Hence, the article by Zareapoor and Seeja (2013) proposes an intelligent credit card fraud detection model called fraud miner. The model creates separate legal transaction patterns (customer buying behaviour pattern) and fraud transaction patterns (fraudster behaviour pattern) on an imbalanced transaction dataset for each banking client by using frequent itemset mining. In order to assess the performance of the fraud miner model, the results were compared with other state-of-the-art classifiers, namely a support vector machine, k-nearest neighbour classifier, naïve Bayes classifier and random forest.

Nami and Shajari (2018) construct an effective fraud detection model employed on a dynamic random forest algorithm to investigate transactional data from a real private bank. Their results indicate that the transactional behaviour of credit cardholders exerts a considerable effect on decision-making regarding the evaluation of fraudulent or legitimate transactions. There are many studies devoted to model credit card fraud detection on transactional datasets, but no relevant research was found that analyses transactional behaviour regarding the current defined research projects. However, the paper provides a useful literature overview of methods applied in the area of analysing transactional behaviour and their evaluation metrics. Instead of strongly focusing on credit card fraud detection Li *et al.* (2012) seek to identify the signs of fraudulent accounts and patterns of fraudulent transactions among ATM phone scams by applying Bayesian classification and association rules. The primary goal is to detect transactional patterns of fraudulent accounts.

The paper by Shih *et al.*, (2011) focuses on analysing credit risks and applying data mining techniques to customers' behaviour to perform risk analysis based on credit card payments. Therefore, the authors generalised a set of clustering rules applying a Kohonen Feature Map (SOM) to identify high risk customer groups. Their goal is to group customers with similar payment patterns into three different clusters (i.e. revolver, translator and convenience users) based on historical transactional payments. The study does not analyse any correlations between the input variables, nor does it implement any predictive model for transactional payment behaviours.

The article by Haeusler (2016) describes how behavioural prediction of transactions in online gambling from bwin.com can be conducted. Therefore, Haeusler examines transactional data to investigate specific payment behaviours to predict future self-exclusion. He noted that the validity of the developed multivariate model is lower than comparable models such as logistic regression and artificial neural networks.

The search term 'predict transactional payment behaviour' has highlighted the interesting study by Black (2005), which depicts the prediction of consumers' willingness to make online payments. The author analysed the influence factors of online consumer behaviour based on consumer auction transactions. The key findings of the analysis are that several features such as general demographic, geographic and economic variables of consumers can be used as input variables for the predictive model. With respect to the underlying dataset, current research seeks to verify whether the entire database can be considered in the feature selection process.

The search term 'categorised transactional payment behaviour' yielded the study by Schutte, Van Der Merwe and Reyneke (2017), in which they apply data mining techniques to determine specific digital banking behaviour and specify behavioural characteristics to increase business value through targeted marketing. The study focuses on banking clients who should always receive an outstanding service in every interaction with the bank due to their large revenue contribution. Their payments, login behaviour, demographical characteristics and online access were analysed and mined. The results were then segmented into three major electronic banking groups. The established categories enable the marketing departments of financial institutions to provide a more accurate and specialised service based on clients' preferences and online banking behaviour.

The subsequent papers are selected from several search enginges based on the search term ‘analysing transactional data’. One selected retrieval outcome of the results list is the study by Hsieh (2004), which proposes an integrated data mining and behavioural scoring model using a neural network and association rules to manage existing credit card customers in a bank by using an account and transactional dataset. Therefore, a two-stage approach for behavioural scoring analysis of implicit knowledge was presented. In the first step, the researcher applied a self-organising map neural network to identify customer groups based on repayment behaviour and recency, frequency, monetary behavioural scoring predicators. The employed technique classifies customers into three major profitable groups, namely revolver user, transactor user, and convenience user. Accordingly, the study discovers hidden behavioural patterns through a combined dataset and provides better banking services based on the developed behavioural scoring. Taghva, Hosseini Bamakan and Toufani (2011) also used a self-organising map neural network to identify groups of customers based on the same approach. In the second step, customer profiling was created by customers’ feature attributes determined using an Apriori association rule inducer. The approach demonstrates that financial institutions can improve their marketing strategies through self-developed behavioural scoring models. However, the first two research projects of the current research study will focus on building cost-sensitive behavioural scoring models for creditworthiness and cross-selling and seek to increase the accuracy of the best-performing classification or regression model with various kinds of statistical techniques. Beyond that, misclassification patterns frequently appear in all three fields of research due to the multi-dimensional transactional payment datasets.

The paper by Chen *et al.*, (2009) discovers recency, frequency, and monetary (RFM) sequential patterns from customers’ purchasing data provided by a retail chain in Taiwan. The authors developed a novel RFM-Apriori algorithm to generate transactional behavioural patterns. Therefore, a new framework for generating valuable information on customer purchasing behaviour for managerial decision-making was proposed. It aims to aggregate various groups of patterns and detect possible changes in purchasing patterns over time. Regarding the last research project, the current research objective is to identify bank customer behavioural changes through categorised payment transactions by mining sequential patterns with a focus on frequent patterns. This essential criterion will also determine the value of a pattern, because there could be other important ones in the transactional dataset. For example, the values of specific payment categories could be a critical criterion for the marketing departments in their banking product recommendations. The paper by Cortes *et al.*, (2016) presents a domain-specific language named Hancock to analyse large-scale transactional data streams. It enables the researcher to compute massive data streams and describes how the developed programming language Hancock addresses these problems from an architectural perspective (i.e. parallelism, static and dynamic checking, etc.).

The short paper by Yen and Chen (2001) proposes a graph-based approach to generate various types of association rules from a large retailer customer database. Put briefly, the authors present three different mining algorithms for generating primitive, generalised and multiple-level association patterns. Their approach includes the five mining phases of numbering, large item generation, association graph construction, association pattern generation, and the association rule generation phase. The researchers' aim is to analyse past transactional data and identify customer purchasing behaviour to enhance the quality of business decisions. Accordingly, they explored the purchased items in a retail transaction. Customer behavioural rules are discovered through large customer transactions. However, the presented approach also has its limitations; for instance, a sufficient memory space is required to discover large databases. The same challenge might occur in conducting the research experiments of the last research project. An association graph and its traverses into a large itemset needs a cost-intensive database scan. The researcher's empirical evaluations show that their applied algorithms outperform well compared with others. Regarding our huge number of categorised payment transactions, current research probably must check whether the cost of generating a large itemset can be reduced. A divide-and-conquer data mining approach might solve this computational issue. Tseng (2013) suggested a hierarchical partitioning approach to mine frequent item sets in large transactional databases, which avoids extra costs for re-scanning the original database during the frequent itemset mining process. However, the current research will not use this complex data mining approach.

Femina and Sudheep (2015) propose an efficient CRM data mining framework including two classification models to predict customers' behaviour for the enhancing decision-making process whenever valued customers should be retained. The Naïve Bayes and multilayer perception neural network models studied are applied to transactional dataset results from a range of direct banking marketing campaigns. It emerged that the performance of the neural network was comparatively better when predicting whether the customer subscribes to a deposit scheme or not. However, no research is conducted to build a cost-sensitive data model for predicting transactional customer behaviour, nor are detailed measurement metrics applied to further data mining algorithms.

Anand *et al.* (1998) developed a hybrid methodology comprising an eight-stage data mining process to solve cross-selling problems in financial institutions by using characteristic rule discovery and deviation detection. The authors highlighted that due to the available dataset, the case cannot be solved by applying a classification algorithm; for instance, there is only a positive example set of sold household insurance accessible. They also underline that the selected algorithm is not as accurate as classification rules will be, although they consider their approach as the most appropriate solution for this kind of issue.

The paper by Islam and Ahsan (2015) implemented a data mining procedure to predict prospective business sectors based on existing customer transactional behavioural data in retail banking. Their objective is to disburse loans with the help of a predictive model. Therefore, an optimal number of clusters were pre-defined through an account and transactional dataset to apply a decision tree classification model. Their field test shows that the developed prediction model is very accurate, and the target-oriented campaign is very promising.

The following document research results primarily arise from the search term ‘machine learning payment transaction data’. Kvamme *et al.* (2018) demonstrate how mortgage defaults can be predicted based on raw account transactional data. They applied convolutional neural networks to checking accounts, savings accounts, and credit card data. The proposed algorithm also performs very well and achieves a ROC and AUC of 0.918 for the networks, and 0.926 for the networks in combination with a random forest classifier.

The study by Yeh and Lien (2009) compares the predictive accuracy of customers’ default credit card payments among a set of different data mining methods (i.e. k-nearest neighbour classifiers, logistic regression, discriminant analysis, Naïve Bayesian classifier, artificial neural networks and classification trees). Only artificial neural networks can perform classification more accurately than the other methods.

However, these results can also be tested by the first research project in terms of whether a neural network is truly the best-performing algorithm in relation to credit scoring. Note that the authors made no efforts to optimise the suggested machine learning algorithms nor used any further interesting supervised learning algorithm.

Most of the studies found in the systematic literature review depict customer spending behaviour analysis rather than behaviour analysis through transactional payments. Finally, it must be noted that not every developed search term (i.e. such as transactional payment behaviour) suggested in the introduced high-level concept map leads to search engine results that are relevant to the current research projects.

The next section is the next building block within the systematic literature review process, providing an overview of work related to our research objectives referenced around the PKDD99/00 challenges with respect to the Berka dataset.

3.4. Research projects around the PKDD99/00 Discovery Challenge

This section examines the scientific papers submitted to the International Conference on Knowledge Discovery and Data Mining around the PKDD99/00 (Principles and Practice of Knowledge Discovery in Databases) challenges. The literature analysis addresses the questions concerning what kind of research topics and which data mining tools as well as methods the researcher has focused on surrounding the freely-accessible Berka dataset.

One of van der Putten's (1999) objectives was to demonstrate some of the potential of knowledge discovery methods for detecting consumption patterns in bank transaction data. This is “a long, tedious process, that requires cooperation of people from different areas” (Berka and Rauch, 2007), and the success of the research process depends on many factors. Further initial exploratory data analysis revealed various emerging business problems that could be approached with data science techniques, such as loan approvals, lifetime value estimation and retail network planning (van der Putten, 1999). The more bank products that a client owns, the less likely it is that the client will switch to a competitor’s bank. In addition, the bank receives a yearly fee and a certain percentage of all purchases (van der Putten, 1999).

Van der Putten (1999) suggests following a strategy of promoting credit card usage: first, credit cards should be advertised within the existing customer base; and second, credit card owners should be stimulated to use their card frequently. Van der Putten (1999) constructed k-nearest neighbour models to predict a score for the ‘owns credit card’ attribute. The model selected the top 20% prospect from the test set and the selection contains 52.2% credit card owners. Compared with a random selection, only 17% of credit card owners could be found. The research highlighted that customer value attribute might be a better target to predict cross-sell candidates for promoting credit cards since clients only own a credit card to a certain degree. If the top 20% prospects are selected according to predicted customer value, then the selection contains 61.4% credit card owners. The credit card ownership model shows a small but significant improvement by using a credit card value as a target value.

Van der Putten (1999) planned to construct prediction models for cross-selling and for upgrading credit card customers, although due the poor data quality of the extracted dataset no further detailed analysis was conducted. The underlying research work will fill the gap of constructing promising prediction models by applying various machine learning algorithms on a subset of a cleaned dataset. Van der Putten does not focus on the data pre-processing part in terms of how reasonable prediction results can be achieved for cross-sell candidates.

Pijls (1999) released a new algorithm for mining frequent item sets within the financial dataset using Unix tools like ‘sed’ and ‘awk’, as at this time no visual or interactive tools are useful for analysing huge datasets. His goal was to discover new knowledge that might be interesting for a bank manager. The examination focuses on analysing some frequencies in the transaction file without having a specific goal in advance. Accordingly, he presents some basic information in relation to the ‘date’ field within the transaction file. In addition, he investigated the correlation between the average balance of an account and the values in the ‘operation’ and ‘k_symbol’ field. No statistical computations such as correlation and regression analysis were conducted. The underlying research work will fill the research gap through an exhaustive descriptive analysis including correlation analysis of the pre-processed payment data by focusing on the transaction field ‘k_symbol’. The research also seeks to perform a frequency analysis to discover more payment behavioural insights focusing on the categorised transaction data given by this specific ‘k_symbol’ transaction field.

Weber (1998) examined interesting rules by determining the statistical criteria implication intensity combined with a minimum coverage requirement and discovered top sub-groups of clients such as credit card holders. However, this investigation was not successful compared with its first objective of ascertaining a significant indicator for bad or good loans. The research approach did not focus on mining transactional categories but turned its attention to discovering promising relationships between selected characteristics towards bad or good loan takers. Regarding the first research objective, the article shows a few interesting rules to indicate potential new card holders or indicate opportunities to enhance bank services. Any research analysis was carried out through rule generation by determining transactional payment behaviour based on the entire categorised transactions of the Berka dataset.

The article “financial data challenge” examined relationships among bank-affiliated branches and tried to find clusters of regions exhibiting similar behaviour, as well as identifying indicators of successful or unsuccessful loans (Miksovsky, Zelezny, Stepankova, Pechoucek, 1999). The focus was set on loan prediction applying the C5.0 algorithm. The outcome of the evaluation rules applied to the dataset resulted in the misclassification of 52 good loan takers (Weber, 1998), and 76 clients were correctly classified as bad loan takers (Weber, 1998). The applied algorithm shows a classification accuracy of 7.6%, which is not one of the best results. However, Miksovsky, Zelezny, Stepankova, Pechoucek (1999) research did not provide more performance characteristics (e.g. true or false positive / negative nor information about precision or recall) of the applied C5.0 algorithm.

In the context of the PKDD99 Discovery Challenge, Levin et al. (1999) present a data mining tool called ‘WizWhy’ to answer very specific questions such as credit card promotion or loan defaults using proprietary association rules algorithms. They explained their data mining approach and their analysis results from a practical perspective to discover accounts holders who are unlikely to repay their loans or to whom the bank should offer a credit card based on unexplained rules. Mining or predicting the creditworthiness or cross-sell candidates based on the entire transactional data was not addressed.

In the context of the 3rd European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'99), the research paper by Coufal, Holeňa and Sochorová (1999) presents 50 hypotheses discovered by GUHA (Petr, 2003; Hájek, Holeňa and Rauch, 2010) on relations among properties of defined data objects. The method generates some interesting insights through a generalised quantifier composed from an antecedent and a succedent, whereby the antecedent increases the probability of the succedent. Besides that, the measure of this association can be described more precisely by the statistical value of ‘Fisher’. They analysed static characteristics of the table account and dynamic characteristics of the table transaction from “Fisher quantifier’s point of view” (Coufal, 1999). Regarding the research field, their focus was not set on measuring the performance of predictive models nor investigating the frequencies or associations of payment categories based on a market basket analysis. However, it should be noted that Coufal, Holeňa and Sochorová (1999) discover a strong association between absence of sanction interest payment and good loan payments, and that good loan payment is associated with the presence of any transaction.

Coufal (1999) extended his research with respect to the loan status classification of clients by performing a structural analysis via hierarchical testing of hypotheses with the GUHA method (Petr, 2003), which consequently forms the basis of the decision tree design process for a loan status prediction. The approach provides a structured description of the analysed dataset and highlighted that a “bad or good payment policy is associated with particular year of account’s or loan’s establishment” (Coufal, 1999).

The research paper by Mohan and M. (2016) explores the scope of analysing bank transaction data to categorise customers by using k-means clustering algorithm, which could help “the bank in efficient marketing, improved customer service, better operational efficiency, increased profit and many other hidden benefits”. They highlighted that the applied algorithm can be replaced with support vector machines (SVMs) or neural networks (NN) for better accuracy. Regarding this issue, we can retain that current research uses both algorithms in the specified research project to forecast the creditworthiness of a bank customer or appropriate cross-sell candidates.

Hotho and Maedche (1999) emphasise that “analyzing concrete customer behaviour delivers important information about customer and customer segments”. Their research paper examines how client profiles can be generated from a large-scale transactional dataset that also occurs in the telecommunication and e-commerce sector, aside from the financial services industry. During the pre-processing stage, the focus was set on PIVOT operations to derive client profiles, and they also applied a principal component analysis to identify the most relevant attributes for their machine learning models. Self-organising maps and intensional rules were also used to cluster the transactional behaviour. It emerged that the study primarily targeted insights into meaningful customer segments about the transaction fields ‘operation’ and ‘type’ in combination with the transaction field ‘k_symbol’. One of the key research projects of the current research study is reduced to analysing the customer payment behaviour based only on the transaction field ‘k_symbol’ to derive more in-depth insights along their detailed characterised transactions using seven possible values (e.g. insurance payment, payment for statement, interest credited, sanction interest, household, old-age pension and loan payment). Finally, Hotho and Maedche (1999) discovered five interesting client profiles and underline that the socio-demographic data provided does not infer meaningful information through the customer segmentation process. Current research also tests whether the given socio-demographic data in the Berka dataset holds any importance for the developed modelling process.

The subsequent paragraphs deal with the examination results of various work related to the PKDD99/00 Discovery Challenge. The selection of the most relevant contributing papers to the current research objectives of this thesis are in line with the ranked evaluation results of the PKDD99/00 Discovery Challenge, given in the appendix A. Thus, current research is only exploring related papers based on the primary studies around the PKDD99/00 Discovery Challenge that are ranked with high and medium importance.

With respect to the results from the PKDD99 Cup, the following works are related to the initially highly-ranked research outcomes from van der Putten's (1999) research study.

The study by Xiong et al. (2013) introduces a personal bankruptcy prediction system running on credit card data. They emphasise that sequence pattern information in credit card data from a major Canadian bank should be taken into account as the main predictor when modelling the personal bankruptcy of banking clients based on a support vector machine (SVM) classifier. Therefore, a novel model-based k-means algorithm for clustering categorical sequences was designed to investigate client behavioural patterns.

The empirical research study by Li and Liao (2011) investigated the performance of five different data mining techniques to recommend “an optimal credit scoring model to reassess the default risk of credit card holders for credit card issuing banks in Taiwan”. It emerged that the decision tree method provides the best classification performance in terms of accuracy and sensitivity. The researcher applied a principal component analysis to select the relevant variables for their predictive models.

The paper by Brause, Langsdorf and Hepp (1999) shows how fraudulent transactions can be detected in credit card payments through a combined probabilistic and neuro-adaptive approach. The researchers applied advanced data mining techniques such as association rules and neural network algorithm to obtain a high fraud coverage combined with a low false alarm rate of fraudulent transactions. However, in comparison with our research, the study has no relevant exposure to one of our targeted research projects. Only the applied association rules as well as neural network on transactional payment data might hold interest in this context.

Both papers from Tsai (2007, 2008) study the credit card usage behaviour motivation. The researcher analysed credit card usage behaviour in terms of specific time period changes by conducting a customer profile, a customer segmentation based on neural network to deduct usage behaviour rules based on a fuzzy decision tree algorithm. The study's main concern is to determine whether there is a behavioural change foreseeable in credit card usage. With reference to our third research project in analysing transactional payment behaviour on categorised transactions, the most interesting point in this context is that the researcher chose a fuzzy decision tree instead of association rules to discover payment behaviour rules of interesting customer groups. Further details about the chosen mining methods in current research will be discussed in the next chapter.

The study by Malekpour, Khademi and Minae-bidgoli (2016) has built a predictive model for fraud activities in e-Banking through a combined usage of supervised and unsupervised learning methods. The feature selection for the model was based on a principal component analysis, which ultimately results in the total extraction of 6 out of 41 variables. The best model performance evaluated through accuracy measures was achieved by applying boosting ANN and bagging ANN methods. Compared with our current research objectives, the study also simulates some experiments based on different supervised learning algorithms for the proposed hybrid model combined with ensemble methods to improve the overall accuracy of the predictive models. However, the study seeks to predict fraudulent attacks in e-banking, and neither cross-selling candidates nor the creditworthiness of bank customers through transactional payment data, although the research evaluated the performance of different classification algorithms based on the measurement characteristic accuracy, which might hold interest when evaluating a model performance.

The following works are related to the initially medium-ranked research outcomes from Weber's (1998) research study.

The article by Sołdacki and Protaziuk (2013) describes the theoretical concepts of association rules and frequent itemset when mining financial data. They explain why it is important – for instance – from a sales perspective what kind of items appear in a transaction, what their distinctive relationships are between each other and at what frequency they occur. Therefore, two approaches were presented to process the financial data with weights, including a brief overview of their advantages and disadvantages. In addition, the article proposes a range of interesting measures such as support, confidence, coverage, lift, conviction or PS (Piatetsky-Shapiro measure), which can be applied to discover interesting rules in financial datasets. However, the focus was set on analysing the financial dataset without using weights in the association rules discovery process.

The research study by Arvind and Badhe (2016) developed a model to analyse various nuances of uncertainty in transactional databases. The authors underline that the measures used in association rules discovery may hold interest when applying a market basket analysis to mine hidden knowledge from a large transactional database. The proposed approach was developed through a vague association rules theory and employed on a dataset from a retail store with the objective of identifying interesting buying behavioural patterns and new schemes of their consumers. However, the first research results in this area were presented by Agrawal, Imielinski and Swami (1993), and their intension was to discover relationships among different items in a transactional database and provide all significant association rules between items in the database. Therefore, the study proposed an algorithm that mines databases by creating candidate itemsets and then frequent itemsets. Referring to our current research objectives, current research aims is to unleash interesting customer payment behaviour through payment categories (items) given in the payment transaction dataset from Berka, such as identifying the most important itemset (minimum confidence), and how often this frequent itemset occurs (minimum transactional support). No research papers were found through the literature review that close this research gap.

With respect to the results from the PKDD00 Cup, the following works are related to the initially highly-ranked research outcomes from (Mohan, L. and M., 2016) research study.

The study by Wang, Wang and Lai (2005) describes a new fuzzy support vector machine to distinguish between good and bad creditors. The researchers reformulated the standard two-group classification problem into a quadratic programming problem because the newly-developed fuzzy support vector machine treats every creditor sample as both positive and negative classes with different memberships generated by three basic credit scoring models, namely a linear regression, a logistic regression and an artificial neural network. Based on empirical tests on three public datasets, the study shows that the newly-developed fuzzy SVM can achieve better discriminatory power than the standard SVM when three different kernels – linear, polynomial and RBF – were compared. However, the research also applies a range of machine learning algorithms for credit risk analysis and highlights the need for efficient and reliable predictive models. Regarding our current credit scoring case, the researchers mentioned that various approaches such as non-linear regression models, logistic regression, probit regression, linear programming, integer programming, k-nearest-neighbour, classification trees and neural networks have been already applied in a vast body of literature to solve this classification problem. The researchers underline that neural networks are "the most promising credit scoring models and have been adopted by many credit-scoring systems", although over-fitting and an opaque mechanism are core challenges, whereby neural networks are far from being optimal classifiers. With respect to our research setup, the research experiments will also prove these claims by identifying the best-performing ML algorithm for predicting the creditworthiness of banking clients.

The research by Yu, Wang and Lai (2008) developed a six-stage neural network ensemble learning model to realise a credit risk assessment and compared its performance with other existing credit risk assessment techniques such as logit regression, artificial neural networks, support vector machines, neuro-fuzzy systems and fuzzy SVM. The evaluation experiments are realised on two published consumer credit card application datasets from the real world by measuring the criteria type I, type II and total accuracy of every applied classification algorithm. The authors found that the neural network model performs the best, followed by single SVM and logit regression, and the two-hybrid classification model also performs relatively well compared with the single classification models. However, it seems that the accuracy evaluation criteria are not sufficient to assess the overall performance of the applied algorithm. The current research study will conduct the performance assessment on a comprehensive list of evaluation criteria. The researcher provides twelve variables to model the firm's characteristics, although any attribute corresponds with the feature selection of the current research study. One reason might be that their experiments target corporate credits and not loans for retail clients.

It should be mentioned that no interesting works were found that are related to the initially medium-ranked research outcomes from Coufal's (1999) research study.

The subsequent paragraphs present extended examination results conducted through a comprehensive literature review around selected research papers from the PKDD99/00 Cup that interface with our research fields. Most of the scientific papers reference the primary studies of the authors who contributed to the PKDD99/00 Discovery Challenge.

Salleb (2000) addresses the problem of extracting solid, multi-level, spatial and non-spatial association rules in geographic information systems (GIS). The author proposed an algorithm that handles hierarchical multi-valued attributes to produce general spatial association rules between geographic layers. The research work developed a prototype that was applied on a real and large geographic database in the field of mineral exploration. Their goal was to discover rules between a reference and descriptive layer according to spatial relations and non-spatial attributes. Regarding the research area of transactional behaviour, there is no indication concerning whether their suggested algorithm can be useful in mining categorised payment transaction.

Agrawal *et al.* (1996) presented two enhanced data mining algorithms (Apriori and AprioriTid) to discover all significant association rules of sufficient support and confidence in large databases. The research showed that the best features of the proposed algorithms can also be combined into a complex hybrid algorithm with somewhat better performance. The authors discussed the usage of their algorithm by identifying buying patterns based on basket data within retail organisations. They emphasise that finding association rules from basket data is generally valuable for goal-oriented cross-marketing actions. However, applying association rules based on an Apriori algorithm might be a feasible method to generate more insights through large categorised payment transactions in case of specific banking product promotions. Another point considered in this work is the performance of the data mining techniques applied through various scaled-up experiments with synthetic data assessed against other mining algorithms (AIS and SETM). The key results are that Apriori significantly outperforms AprioriTid on large itemsets, although the study creates the awareness of several problems such as missing quantities or values of the items bought in a transaction.

Padillo, Luna and Ventura (2016) presented an interesting study about sub-group discovery on big data by proposing two exhaustive search algorithms (AprioriKSubgroup Discovery Optimistic Estimates and Parallel FP-Growth Subgroup Discovery Optimistic Estimates). The researchers conducted a detailed experimental study to prove the efficiency and scalability of the algorithms including a comparative study with traditional techniques. They proved that both algorithms are highly efficient in mining sub-groups on big data. The detailed examination of the performance shows that the AprioriK-SD-OE performs best when a huge or very low number of sub-groups are extracted.

The paper from Zaki, Parthasarathy and Ogihara (1997) proposed four new data mining algorithms depending on the clustering and lattice traversal scheme for the fast discovery of association rules in large transactional databases. The presented algorithms use novel itemset clustering techniques to approximate the set of potentially maximal frequent itemsets. The authors also use a vertical database layout to cluster related transactions together. Although the study presents efficient methods to discover frequent itemsets, no details about frequent categorised transactions are examined.

Wang, Wang and Lai (2005) proposed a bilateral weighted fuzzy support vector machine to discriminate good from bad creditors with different memberships generated by some basic credit scoring methods such as linear regression, logit regression and neural networks. The credit scoring issue was reformulated from a two-group classification problem into a quadratic programming problem. The experimental tests on three public datasets showed that the developed algorithm can have better discriminatory power than the standard support vector machine and the fuzzy support vector machine if appropriate kernel and membership generation methods are chosen. The authors mentioned that neural networks are the most promising credit scoring models adopted by many credit scoring systems. There are also many publications of neural network applications in credit scorings. While the general approach of credit analysis is explained in a vast body of literature, I could not find literature in which a data science approach was proposed to optimise the characteristics of a dataset in credit scoring predictions.

The paper by Benos and Papanastasopoulos (2007) developed a hybrid model for credit risk measurement based on the standard Merton approach. The authors extended the basic approach to estimate a new risk neutral distance to default. They ascertained how both the in-sample fit of credit ratings and the out-of-sample predictability of defaults can be improved by enriching the model with financial ratios and accounting variables. They also emphasised that the proposed predictive models do not reflect all available information regarding the credit quality of a firm. The focus was set on measuring firms’ creditworthiness instead of assessing the creditworthiness of private banking clients, which will be examined in the underlying research.

Ouardighi, Akadi and Aboutajdine (2007) address the issues of the feature selection process on supervised classifications when using Wilk’s Lambda statistics. They evaluated the performances of the method used in a discriminant analysis. Their research also considers the rationale of how to build the classifiers more efficiently. It emerged that the volume of computation can be reduced in some cases and the prediction power can be increased, especially when irrelevant features for classification are deleted. One of the current research goals is also to develop a cost-sensitive data model to forecast the creditworthiness or cross-selling candidates with the highest prediction accuracy through a minimum subset of features. Therefore, the research uses diversified sets of datasets so that the performance of the data science approach can be assessed in a variety of applied machine learning algorithms. However, current research wants to test the efficiency of the feature selection procedure based on a random forest model and identify the best performance regarding the selected features by comparing the accuracy and error rates of the supervised learning algorithms.

The next section summarises all relevant results from the three building blocks of the systematic literature review process presented at the beginning of this chapter.

3.5. Summary of the systematic literature review findings

This section summarises the key findings from the systematic literature review. I have developed a suitable process to undertake the literature review including a search strategy applied to selected search engines to identify the most relevant research contributions aligned to the current research projects. First, I have examined studies dealing with the research field of transactional payment behaviour in general, and before considering scientific papers submitted to the PKDD99/00 Discovery Challenge, analysing various papers that hold relevance in relation to the freely-accessible Berka dataset. In order to facilitate the analysis of literature, I have introduced a SLR framework comprising four key elements and organised the transactional payment behaviour research under the presented framework.

The systematic retrieval delivered many selected primary studies and sources that were compared with pre-defined exclusion and inclusion selection criteria. It emerged that many studies deal with neural networks to assess credit risks or various related business context. The review was aligned with a two-stage iterative process and the chosen inclusion and exclusion selection criteria. Therefore, the review selection criteria and procedures were applied to the full references of primary sources.

Moreover, papers evaluating transactional payment behaviour around both PKDD Cups were also important to the evaluation because the released Berka dataset is part of the research object. Hence, the following inclusion criteria were applied:

1. Research publications/studies such as peer-reviewed scholarly journals or technical reports that describe transactional payment behaviour studies, in which advanced data science approaches were used or applied to categorised payment transactions streams, credit scorings or cross-selling cases.
2. Research publications that examine the three specified research projects when modelling customers' transactional behaviour.
3. Methods, data mining techniques and tools used.

The SLR excluded those studies that match the criteria below:

4. Research publications that do not discuss or relate closely to the introduced research projects.
5. General publications related to transactional behaviour beyond transactional datasets.

Finally, the principal selection of primary sources was grounded on an investigation of the title, abstract, keywords, introduction and conclusion. “Only those primary sources that appeared to be completely irrelevant were excluded” (Brereton *et al.*, 2007). All other primary sources that were not excluded were compared against the exclusion and/or inclusion criteria outlined above. Each primary source was updated in the tabulated results overview provided in the appendix A to determine whether the primary source holds strong importance in the SLR. Only literature was included that addresses the defined research projects.

The final findings include a list of more than 60 journals from many different publishers focusing on the primary sources in this review for searching significant papers. The literature analysis showed that most of the selected research works were case studies covering certain transactional payment behaviour areas by proposing a model to predict customer payment behaviour in a specific course, whereas there is a lack of research for generalised models that help to discover payment behaviour based on categorised transactions and thereby classify hidden transactional patterns.

Table A-4 summarise the literature analysis of papers around the theme of transactional payment behaviour. The analysis of the papers revealed that the characteristics such as author/year, mining steps, method, tool, outcome and rank shown in the tabulated results provided in the appendix A have been investigated. As expected, papers for analysing, measuring or predicting credit scores are well represented. Although there is a vast body of literature given in this research area, no relevant research was found that focuses on developing cost-efficient data models. The same can be noted for the second research project, whereby most literature does not use machine learning techniques to discover cross-selling cases, nor does it propose any cost-sensitive data model to predict cross-selling candidates accurately. In addition, no research was found that uses a combined tooling set existing of R studio, Python and MS Azure ML.

Altogether, various papers from the financial dataset data analysis challenge were presented at the PKDD99 and PKDD00 conferences. Most of the contributions deal with the classification of customer behaviour, although research has also been conducted on temporal aspects of the financial dataset. Berka (2006) distinguished all of the contributed research into two types: “method/algorithm-oriented” research papers and “problem-oriented” research papers. The current research thesis is a combination of both types as the thesis focuses on describing a novel data science approach and uses the given financial dataset more or less to demonstrate the performance measurement of different applied machine learning algorithms, as well as responding to the formulate research challenge in predicting uncategorised payment data streams that can be interesting for banks, domain experts or other third parties.

Table A-3 in appendix A summarises all of the papers related to the PKDD99 and PKDD00 challenge in terms of the problem solved (outcome), data mining steps described, mining algorithms, methods and system used and a relevance ranking related to the current research goals. The literature search followed established procedures and selection criteria, which resulted at least in the classification of three valuable transactional payment behaviour publications. However, research on the proposed research projects are clearly under-represented. The most significant research related to the current research thesis was conducted by Oreski, Oreski and Oreski (2012), Mohan and M. (2016) and van der Putten (1999).

Surprisingly, I did not identify important papers focusing exclusively on forecasting categorised payment transactions or mining the relationships between theses categorised transactions using a combination of supervised and unsupervised learning methods. Only the paper by Schutte, Van Der Merwe and Reyneke (2017) proposes a data mining approach to model customer behaviour scoring based on transaction history. Regarding the tabulated results given in the appendix A, it can be seen that some data mining models such as decision trees, neural networks and support vector machines are applied to the Berka dataset, although there is very little published literature concerning categorised payment transaction predictions in the data mining field. It is well known that the most commonly-used mining method in predicting the creditworthiness of a bank customer is credit scoring models. Most of the applied models are based on empirical knowledge because the input variables comprise analysed statistics to select suitable characteristics related to creditworthiness or cross-selling candidates. Compared with the selected published literature related to the first two research projects, there is no study available that seeks to forecast credit scores, or cross-selling candidates based on a cost-sensitive data model and demonstrates promising prediction performance.

It is even more surprising that research analysing associations or frequencies between (categorised) transactions or forecasting transactional payment categories – for instance, on a deep-learning model – is almost non-existent. A further key finding from the systematic literature review was the absence of building a cost-sensitive data model to predict credit scores or cross-sell candidates. An exhaustive performance comparison of various applied machine learning algorithms such as classification, regression, clustering and/or association rules is also missing. Most notable exceptions are the research work by Yap, Ong and Husain (2011), who investigate late payments, Chye, Chin and Peng (2004), who examine credit risks, Li and Liao (2011), who evalutate the default risk of credit card holders, and Malekpour, Khademi and Minae-bidgoli (2016) who mined fraudulent attacks in e-Banking. All other research studies do not discuss the performance comparison of more than three or four different mining algorithms and focus only on up to two or three different mining algorithms.

As can be seen in the trend chart below, to date various research efforts have been made by analysing transactional payment behaviour in general as well as around the PKDD99/00 challenge and including the research activities related to them. Therefore, the bulk of this research has concentrated on scoring good or bad loan payments with the objective of evaluating customers' creditworthiness (Weber, 1998; Levin, Cheskis, Gefen, Vorobyov, 1999; Coufal, 1999; Miksovsky, Zelezny, Stepankova, Pechoucek, 1999; Wang, Wang and Lai, 2005; Tsai, 2007). Otherwise, only a limited number of research studies were found that relate to the second research project. For instance, Liu and Cai (2008) use a neural network to build a customer cross-selling model to identify further business potentials through direct marketing. After conducting the SLR, no research study is known that focuses solely on predicting credit card promotion candidates to improve cross-sells by applying a best-performing machine learning algorithm on a cost-sensitive data model.

Research studies on predicting categorised payment transactions with respect to analysing transactional payment behavioural patterns are infrequent in the relevant literature and – to my best knowledge – no such study exists that forecasts categorised transactions as well as evaluating customer behavioural relationships on payment streams. Most of the literature deals with applying association rules to mine transactions on various transactional databases (Agrawal, Imielinski and Swami, 1993; Agrawal et al., 1996; Zaki, Parthasarathy, Ogihara and Li, 1997; Brause, Langsdorf and Hepp, 1999; Hsieh, 2004; Sołdacki and Protaziuk, 2013; Arvind and Badhe, 2016). For instance, Yen and Chen (2001) developed a graph-based approach to explore large transactions from a retailer database. Some other studies have employed association rules mining for fraud detection (Tsai, 2007, 2008; Malekpour, Khademi and Minae-bidgoli, 2016), and others have used association rules to mine frequent itemsets on transactional datasets (Pijls, 1999; Sołdacki and Protaziuk, 2013; Zareapoor and Seeja, 2013).

The current scope of research will be distinctive when the research seeks to unlock transactional payment behaviour through categorised transactions by applying in particular a combination of supervised and unsupervised learning methods. Current research aims to close this research gap formulated through the last research project by providing new payment behavioural insights with the help of machine learning algorithms and make a lasting contribution to research in the area of data science practices.

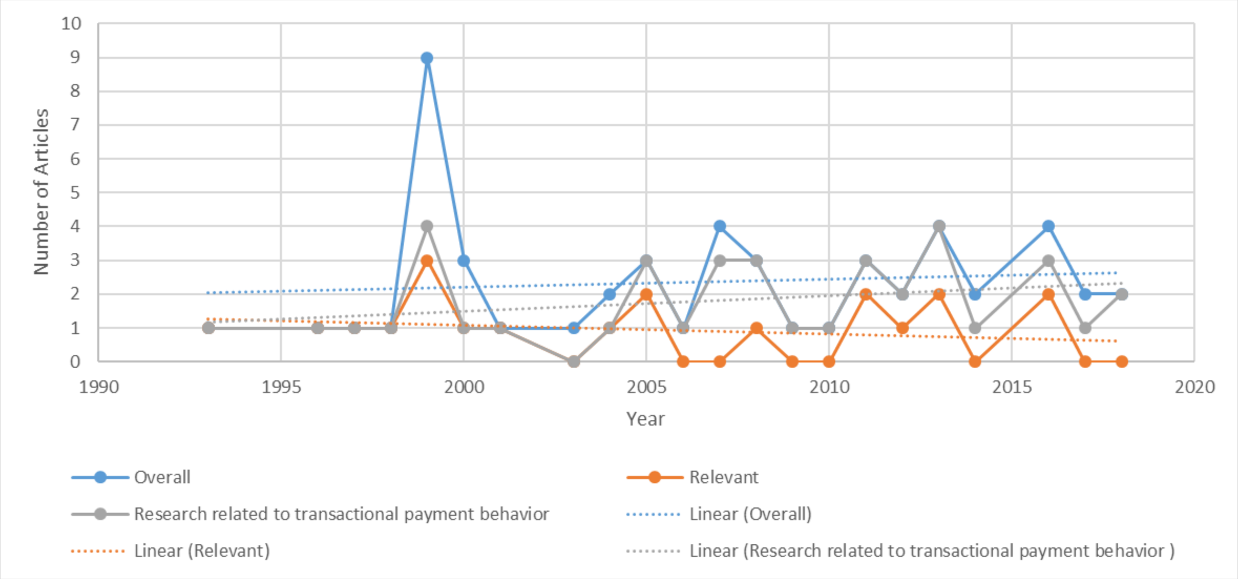


Figure 3-4: Overall trend of the ranked research activities (studies around PKDD99/00 challenge, related to them and studies of transactional payment behaviour analysis)

In summary, there is a slight increase in the number of research contributions related to transactional payment behaviour in general between 2005 and 2018. Within the same period, the trend line for relevant research contribution depicts that substantial research activities in the area of transactional payment behaviour slightly declined. As noted in the figure above, there is an outlier around 1999 and 2000 for overall research contributions given that I have consolidated an extraordinary and extensive literature review around the Berka dataset used in the PKDD99/00 challenge.

The research has proposed a concept map to classify the articles available in the academic database of literature between the periods of 1993–2019 covering a vast number of different journals. More than 100 interesting articles were identified and reviewed for their direct relevance to the current differing fields of research. I have found that the research area of transactional payment behaviour received most research attention, although researchers have also applied different data mining techniques to analyse payment behavioural patterns in various business contexts.

Even by reading all of the papers shown in the figure below, the current research analyses transactional data by focusing on predicting creditworthiness, cross-selling opportunities as well as categorised payment transactions. The reason behind this approach is to unlock hidden behavioural pattern and discover new transactional behavioural insights to contribute new knowledge to research. Likewise, given the focus on peer-reviewed papers, many putative significant papers were excluded to ensure the quality of the research presented in these studies. Regarding our tabulated results, many important research papers (Yen and Chen, 2001; Miksovsky, Matousek and Kouba, 2003; Yan et al., 2011; Padillo, Luna and Ventura, 2016) were efficiently extracted from the IEEE database. However, the remaining database such as ACM Portal, Elsevier Science and Emerald also provide tangible search results concerning the specified scope of research.

The conducted SLR reveals no studies on transactional payment behaviour that address issues such as assessing the performance of various applied machine learning algorithms in case of modelling the creditworthiness of bank customers or modelling cross-sell candidates for bank marketing campaigns, and especially no important research was found studying customer behaviour only on categorised payment transactions by using supervised and/or unsupervised learning algorithms. To summarise, a few articles mentioned some smaller parts of current scope of research, albeit ultimately in a different research context. For instance, the SLR highlighted that descriptive analysis techniques along with developing cost-sensitive data models that can increase the prediction model accuracy are discussed less in research. Therefore, one of the main objectives of the current research is to propose a data-driven approach to optimise the prediction accuracy of data models themselves. Therefore, a detailed research process is demonstrated as part of the research design and methodology in the next chapter.

As noted, a range of studies on current research in transactional payment behaviour have been carried out. Figure 3-5 below also shows the volume of research activities around transactional payment behaviour. It seems that most research studies analysed based on the literature selection criteria are related to the classification and association methods.

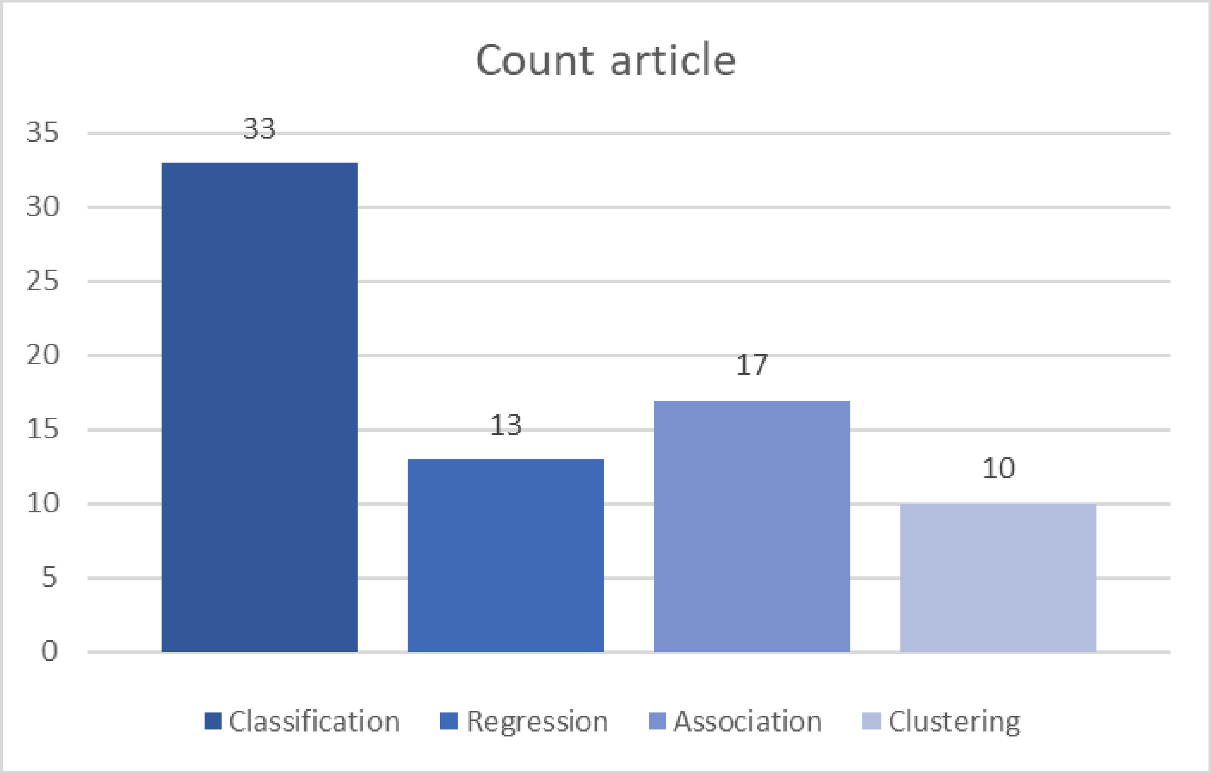


Figure 3-5: The extent of data mining techniques used in the transactional payment behaviour analysis between 1993 and 2019

Some of the published papers used a combination of clustering and classification methods to determined digital banking behaviour (Schutte, Van Der Merwe and Reyneke, 2017), model behavioural scoring (Hsieh, 2004) or predict fraudulent attacks based on transactional datasets (Malekpour, Khademi and Minae-bidgoli, 2016). However, most puhlished papers used a combination of classification and regression methods to conduct their research. This relates in particular to Li and Liao (2011), who model credit card behavioural usage, research studies evaluating credit risk/scores (Wang, Wang and Lai, 2005; Vojtek and Kocenda, 2006; Yu, Wang and Lai, 2008b; Bijak and Thomas, 2012; Bekhet and Eletter, 2014), Benos and Papanastasopoulos (2007), who assess credit quality, Trubik and Smith (2000), who model customer leaving patterns, Abdou, Pointon and El-Masry (2008), who realise a performance comparison of credit scoring models, Gschwind (2007), who predicts late payments, Black (2005), who forecast future online payments, as well as Schutte, Van Der Merwe and Reyneke (2017), who determine login banking behaviour.

Moreover, it was found that many researchers explicitly use association rule mining as single or primary mining techniques in their research, including the studies by Levin, Meidan, Cheskis, Gefen and Vorobyov (1999) for loan predictions, Brause, Langsdorf and Hepp (1999) for detecting fraudulent transactions, Sołdacki and Protaziuk (2013) as well Arvind and Badhe (2016) for interesting rule discovery, Agrawal, Imielinski and Swami (1993) and Agrawal et al. (1996) for association rule mining in a retail item set or other transactional datasets (Zaki, Parthasarathy, Ogihara and Li, 1997; Yen and Chen, 2001; Hsieh, 2004) and the research study by Berrado, Elfahli and El Garah (2013), assessing the main drivers of mobile payment adoption.

The next chapter relates to the research design and research methodology developed for the current piece of work. It discusses in further detail the feature selection process, the data pre-processing stages, the data analysis and the theoretical concepts used in mining transactional payment behaviour for the three pre-defined research projects.

Chapter 4

4. Research Design and Methodology

This chapter deals with all aspects of the concepts, the research design, research methodology and the realisation process of the entire research study. The following research process depicts the relevant steps within our distinctive approach based on various phases of the CRISP-DM process to answer the developed research aims and objectives.

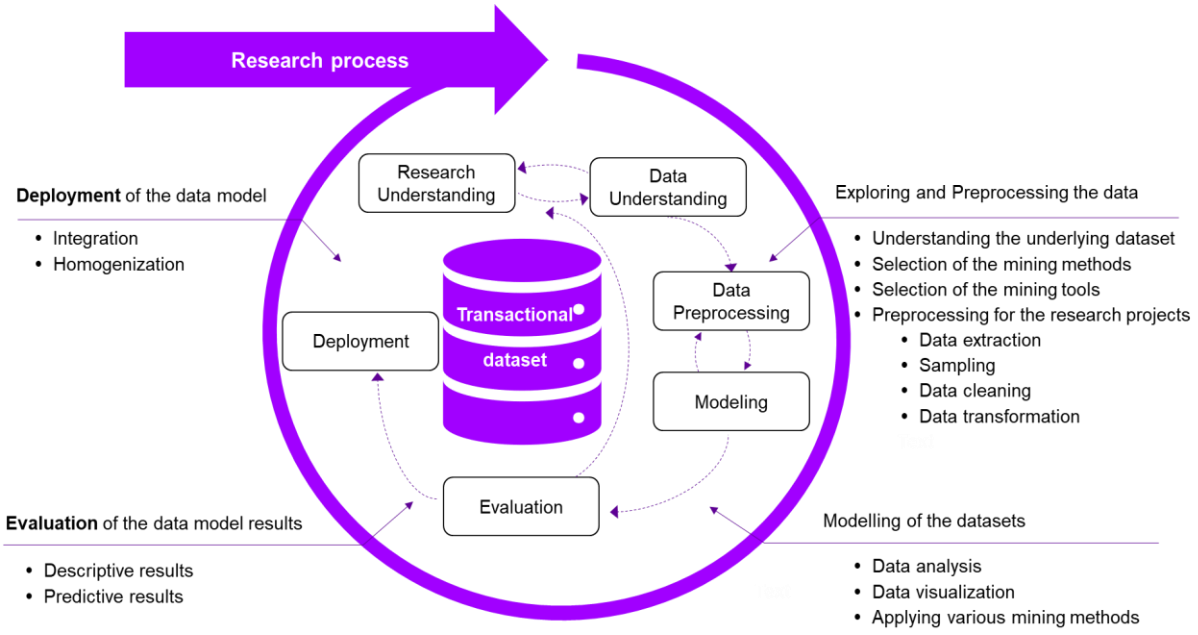

Research process
Research Understanding
Data Understanding
Data Preprocessing
Modeling
Evaluation
Deployment
Transactional dataset
Deployment of the data model
• Integration
• Homogenization
Exploring and Preprocessing the data
• Understanding the underlying dataset
• Selection of the mining methods
• Selection of the mining tools
• Preprocessing for the research projects
• Data extraction
• Sampling
• Data cleaning
• Data transformation
Evaluation of the data model results
• Descriptive results
• Predictive results
Modelling of the datasets
• Data analysis
• Data visualization
• Applying various mining methods

Figure 4-1: Structure of the research process developed from Chapman *et al.* (2000)

The research aim is to follow as closely as possibly the research process as shown in figure 4-1 above. The process passes through a series of research phases – research and data understanding, data pre-processing, modelling, evaluation and deployment – with a distinct intensity scale. The various phases are divided into different sections in which the study examines every research project in depth and detail.

4.1. Overview of the research design and research methodology

The research has a twofold purpose, the first of which is to assess the performance of a range of supervised classification and clustering algorithms by calculating credit scorings and/or cross-sell candidates. In this context, the research takes a closer look at how well selected algorithms deal with these issues. Based on the knowledge gained from the literature review in chapter 3, no evidence was found that any researcher has yet explored in greater detail the performance of various applied supervised machine learning algorithms when comparing their outputs with the given financial dataset. The second research purpose is to investigate customers' digital footprint by looking at customer behavioural changes considering categorisation types on existing payment transactions from a data science perspective and developing a forecast model that can identify relevant patterns and predict categorised payment transactions within an uncategorised payment ecosystem with a small percentage of error. At present, the systematic literature review shows that there are presumably no significant research contributions available that provides new and in-depth insights into this specific research field of customer behavioural changes nor applies a data-driven approach to predict customer behavioural changes through various payment transaction channels in the digital age.

In this thesis, we also aim to review the theoretical concepts behind the data science techniques applied as well as predictive analytics science and how it works. Subsequently, we will provide an overview of the tools applied in the underlying research and how we can leverage them to answer the research questions and gain more valuable insights. Van der Putten (1999) already mentioned that a large number of choices can be made when detailing data mining objectives, preparing the data, evaluating the newly-gained insights, applied models and their results. Generally, the research methodology is derived from the following data science process illustrated in figure 4-2 below. As a result, the process flowchart reflects the logical structure of the entire chapter.

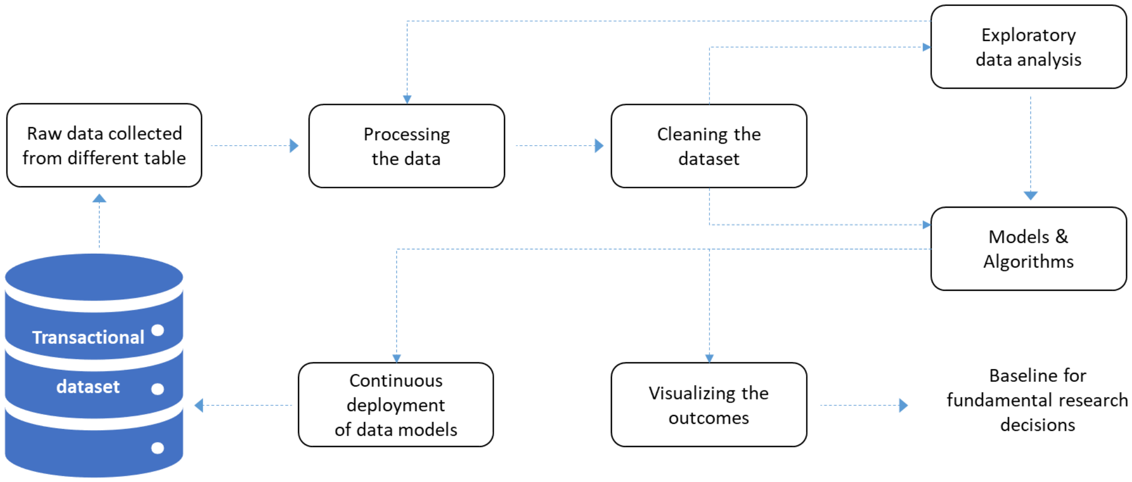

Exploratory data analysis
Raw data collected from different table
Processing the data
Cleaning the dataset
Models & Algorithms
Transactional dataset
Continuous deployment of data models
Visualizing the outcomes
Baseline for fundamental research decisions

Figure 4-2: Visual guide to the data science process flowchart

The flowchart provides an overview of the key data mining steps along the applied data science journey for this research. Figure 4-2 also outlines the theoretical framework of the research study, which is based on a four-phase approach described as follows:

(1) Assess the performance of a set of supervised classification and clustering algorithms in terms of whether they are accurate for credit scoring and/or cross-sell candidate predictions.

(2) Investigate the effectiveness of advanced forecasting and predictive methods and assess whether they are suitable and applicable for changing customer behaviour identification based on the categorised payment history.

(3) Obtaining and investigating categorised payments data to advance precursor events in uncategorised payments data and assess whether supervised or unsupervised learning algorithms are the most appropriate methods.

(4) Different forecast models are explored and applied to the transactional dataset to help predict future credit scores for credit applicants and cross-selling candidates for promotions.

The research approach comprises various elements, which will also be included in this investigation. Regarding the data science process displayed in figure 4-2, certain types of transactions will be diagnosed during the data pre-processing phase, and the exploratory analysis phase uses different data analysis techniques that can be applied in transactional datasets. Other research elements to be mentioned are the more effective use of data in existing data analytical methods.

Further research elements of this thesis include scrutinising the effectiveness of advanced forecasting and predictive methods in the transactional dataset. The research investigates the use of statistical and machine learning techniques for the specific field of payment transaction flows. In fact, applying different forecast models or a combination of cost-sensitive models is also a major theme of this thesis. The reason behind this is to value the effectiveness of different forecast models for the first two research projects and identify a high-performing mining algorithm for every single research project. The newly-gained in-depth insights by using various machine learning algorithms such as clustering, decision trees, logistic regression, random forest models and neural network algorithms or other supervised learning algorithms like support vector machines can also be useful for the last research project. These outcomes can serve as a starting point for pre-processing the data in a different way to predict uncategorised transactions more efficiently.

Overall, the research considers many important research elements, whereby the four-phase approach also requires a well-managed research map. For instance, the process for building an important data-driven model should not be underestimated. Figure 4-3 below highlights that various steps within the process will allocate different timeframes among these illustrated building blocks. It is widely known that modelling is only a minor part of building high-end models in the research area of artificial intelligence (AI). Researchers devote roughly 80% of their time to preparing and managing the data for analysis. Thus, data munging[1] is the most time-consuming part of the research study.

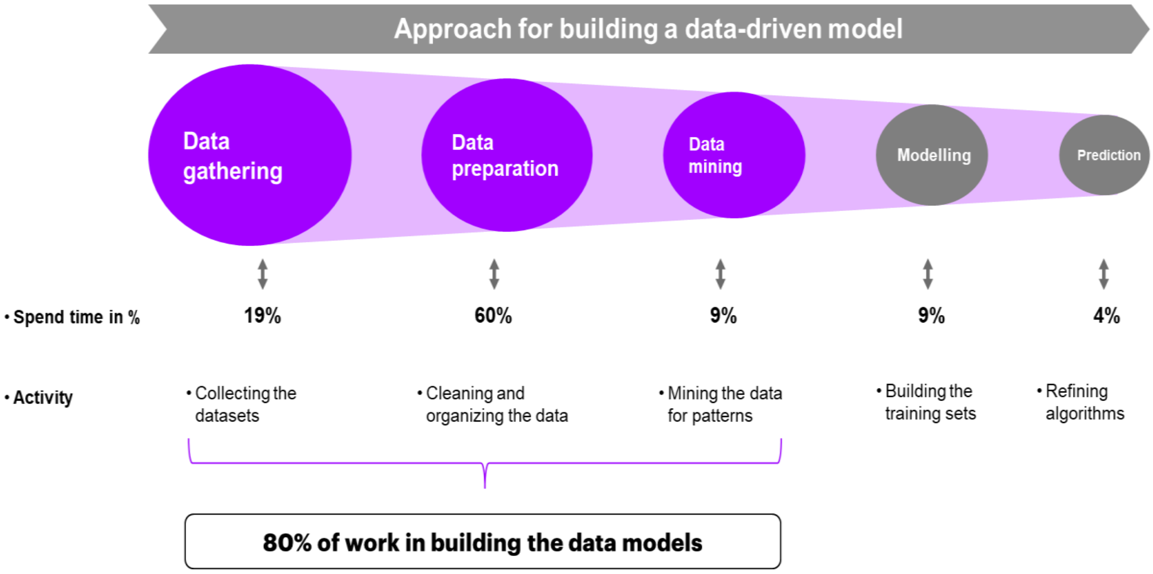

Approach for building a data-driven model
Data gathering
Data preparation
Data mining
Modelling
Prediction
• Spend time in %
19%
60%
9%
9%
4%
• Activity
• Collecting the datasets
• Cleaning and organizing the data
• Mining the data for patterns
• Building the training sets
• Refining algorithms
80% of work in building the data models

Figure 4-3: Approach for building a data-driven model

[1] https://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says/#349631686f63, accessed on 27th December 2018.

The research questions are implemented by using a real dataset of a Czech bank. The goal of the data science approach demonstrated above is to show common pitfalls in data munging and how to avoid them, given that the success of research depends on the ability to collect the dataset, as well as cleaning and organising it for data mining purposes in an appropriate way. The remaining mining activities such as building the training set and refining the machine learning algorithms used are only allocated up to 20% of the entire time. For instance, Miksovsky, Matousek and Kouba (2003) highlighted that the success of every data mining algorithm is strongly dependent on a quality of data processing, which can result in a very complicated and challenging task.

Further details about the research design and the research methodology applied will be discussed in the following sections.

4.2. Understanding the underlying dataset

The data is sourced from payment transaction databases from a Czech bank and classified based on existing data analytic methods with the aim of being investigated for the research. This also involves pre-examining sizeable datasets to ascertain their suitability for the research. The data is classified in every experiment of the different research projects as training data that may constitute 80% of the data, with the remaining 20% being classified as test data. The datasets are then analysed using specific mining and forecasting methods, whereby details will be explained in the next section.

The raw dataset is generally described by Coufal, Holeňa and Sochorová (1999) as follows: “Variables of static characteristic are given by tables account, client, disposition, permanent order, loan, credit card and demographic data” and “variables of dynamic characteristic are given by table transaction”. Regarding the previous section, the illustrated research process considers both characteristics during the data pre-processing phase, as the processed dataset must be aligned with the research project itself. As a result of this causal link, Daniel Keys Moran determines that “you can have data without information, but you cannot have information without data”. From this, it can be deducted that the scientific research is based solely on data.

In order to ensure a good quality of the pre-processed dataset, the following figure 4-4 describes the data science approach applied to gain more insights from the dataset, and particularly to improve the understanding of the research field.

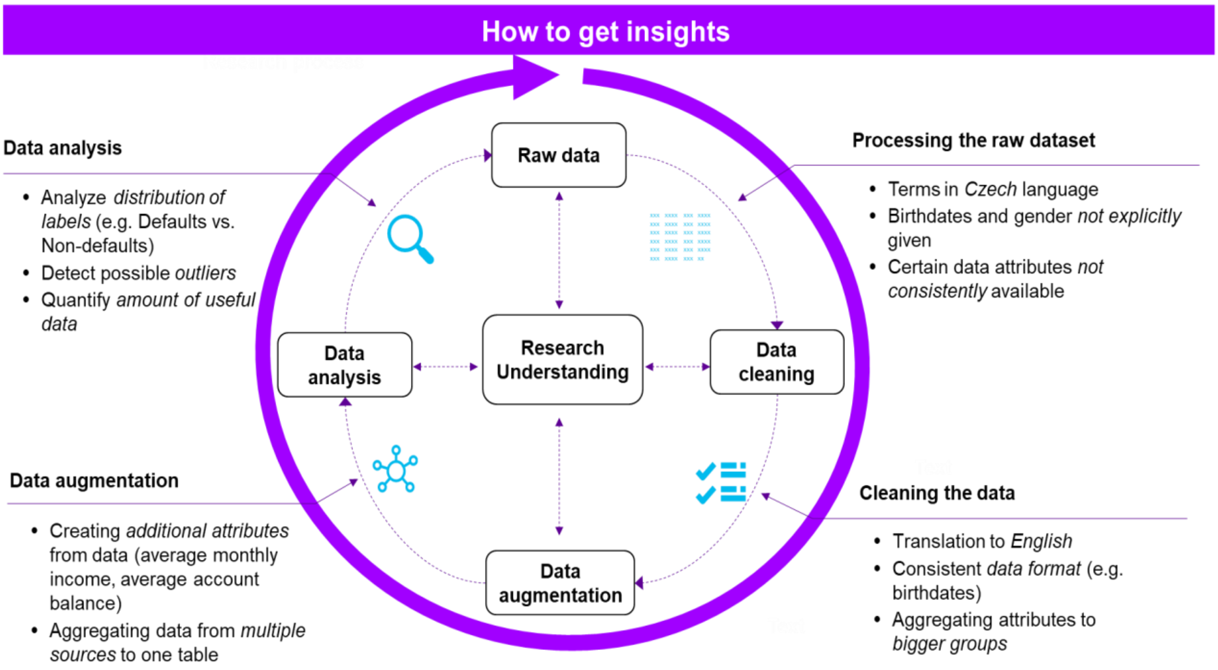

How to get insights
Data analysis
• Analyze distribution of labels (e.g. Defaults vs. Non-defaults)
• Detect possible outliers
• Quantify amount of useful data
Processing the raw dataset
• Terms in Czech language
• Birthdates and gender not explicitly given
• Certain data attributes not consistently available
Raw data
Data analysis
Research Understanding
Data cleaning
Data augmentation
Data augmentation
• Creating additional attributes from data (average monthly income, average account balance)
• Aggregating data from multiple sources to one table
Cleaning the data
• Translation to English
• Consistent data format (e.g. birthdates)
• Aggregating attributes to bigger groups

Figure 4-4: Analysis approach for the underlying dataset

The approach to understanding the underlying dataset from Berka is determined as a rolling process that forms the given dataset step-by-step with the objective of hedging a significant dataset for the research purpose. Figure 4-4 shows the iterative flow through various analysis stages such as raw data, data cleaning, data augmentation and data analysis and its close interaction with the research understanding as a whole.

In general, the first step is always to import the raw data into R Studio or Python depending on the research field that the researcher wants to analyse. Therefore, “Czech expressions within the raw data was replaced by English words” (Pijls, 1999) during the pre-processing phase, and birthdates and gender are not explicitly given, so that “the date field was split up into three separate fields for year, month and day” (Pijls, 1999). The second step along the iterative approach is to start cleaning the data for advanced processing. As an example, empty strings and strings comprising one space in the attributes ‘operation’, ‘k_symbol’ and ‘frequency’ are also cleaned. Pijls (1999) found that the ‘type’ attribute is functionally dependent on the ‘operation’ attribute and is redundant, whereby we can also remove this attribute in the cleaned dataset.

The next step in the research understanding process is the data augmentation, in which I create additional attributes, features or labels and aggregate multiple table from the Berka dataset into one single dataset aligned to the research field. These activities must be seen as an overall basis for the entire research approach, which will provide parsing and gathering steps during the data pre-processing phase with the goal of identifying systematic patterns in the financial dataset. For instance, analysing the given demographic data can serve as an example. Figure 4-5 below shows how the pre-processed data can be visualised to increase the understanding of the underlying dataset. Examples among this stage include illustrations of “NoDefaultsPerRegion”, “NoLoansPerRegion”, “NoDefaultsPerDistrict” or “NoLoansPerDistrict”.

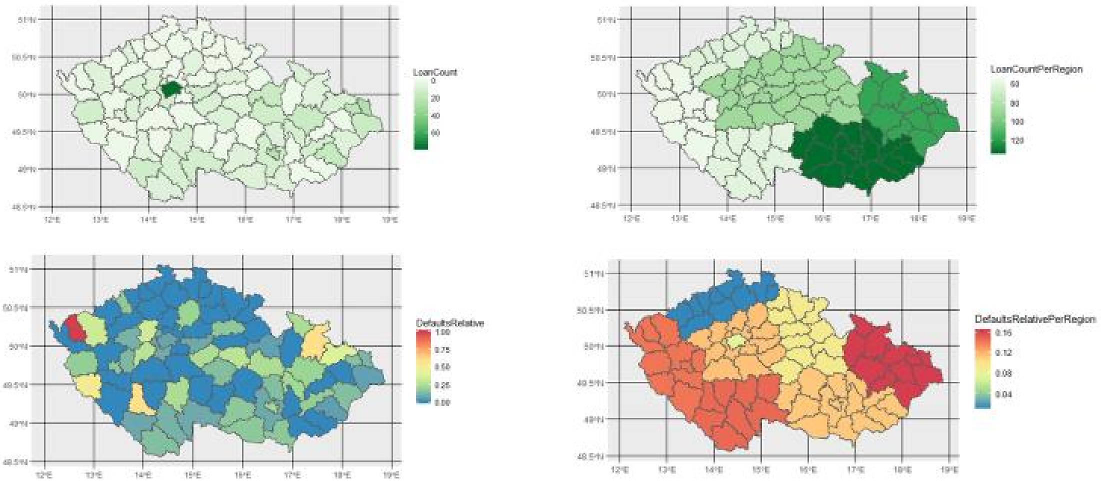


Figure 4-5: Sparse the data for deeper analysis

The final step in the approach of understanding the underlying dataset is the data analysis activities, which enable the researcher to gain more in-depth insights with respect to particular research questions. The data mining steps include outlier detection, analysing the distribution of specific labels (see figure 4-6 below) or quantifying the useful amount of relevant data. For example, Pijls (1999) discarded the code of the bank of the partner as “it appeared that the occurrence of bank codes of distributed uniformly”.

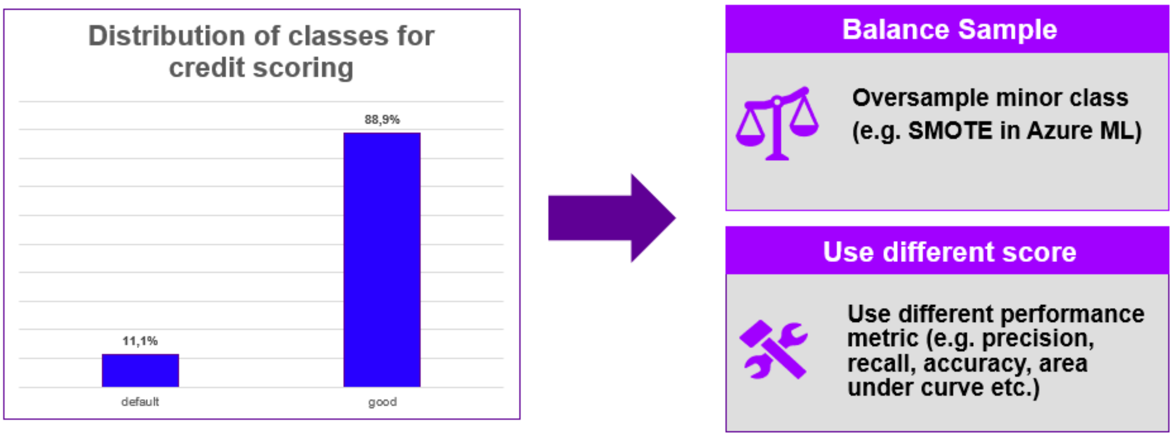

Distribution of classes for credit scoring
11,1%
88,9%
default
good
Balance Sample
Oversample minor class (e.g. SMOTE in Azure ML)
Use different score
Use different performance metric (e.g. precision, recall, accuracy, area under curve etc.)

Figure 4-6: Sample for imbalanced classes for credit scoring

Finally, the presented iterative approach fosters the data understanding and will also increase the research understanding of the research projects. Vaghela, Kalpesh H and Nilesh K (2014) underline that data selection is an essential data processing step within the data mining process. Details will be discussed in the following sections.

4.3. Selection of the data model and tools

Vaghela, Kalpesh H and Nilesh K (2014) highlighted that the "most existing data mining algorithms (including algorithms for classification, clustering, association analysis, outlier detection, etc.) work on single tables". Data pre-processing steps with the support of most suitable tools are decisive as many classification algorithms can only be applied to a single relation. Therefore, it is necessary to fulfil these requirements during the pre-processing phase, whereby appropriate tools can help to do so effectively. Apart from this, applying data mining techniques on a transactional dataset can lead to several challenges such as not having a customer-centred dataset built, or the transactional dataset not being aggregated to meaningful attributes. Kovalerchuk and Vityaev (2010) highlighted that the only realistic approach that has proven successful is providing comparisons between different machine learning algorithms, showing their strengths and weaknesses relative to research questions and leaving the selection of the method that likely fits the specific research objectives to the researcher. In general, a clear understanding of the dataset and data models applied is more important than the data mining tools as it can significantly change and improve the research design.

Figure 4-7 below shows that an increased complexity within the models will also reduce the ability to interpret the results, and a danger of over-fitting the model will emerge. Regarding Occam's razor problem-solving principle (a law of simplicity), we should choose the model to be only as complex as necessary to conduct the research with the fewest assumptions. Kovalerchuk and Vityaev (2010) adhere that there is a need to build models that can be very quickly evaluated in terms of both accuracy and interpretability. As part of the research design, the tool choice plays an essential role in increasing the quality of the evaluation results in terms of validity and significance through the research process.

Figure 4-7 illustrates a gross characterisation of the mathematical models and their complexity, divided according to the various scripts created for the research. Details about the selected mining methods including their theoretical concepts for every research project will be explained in the following sections.

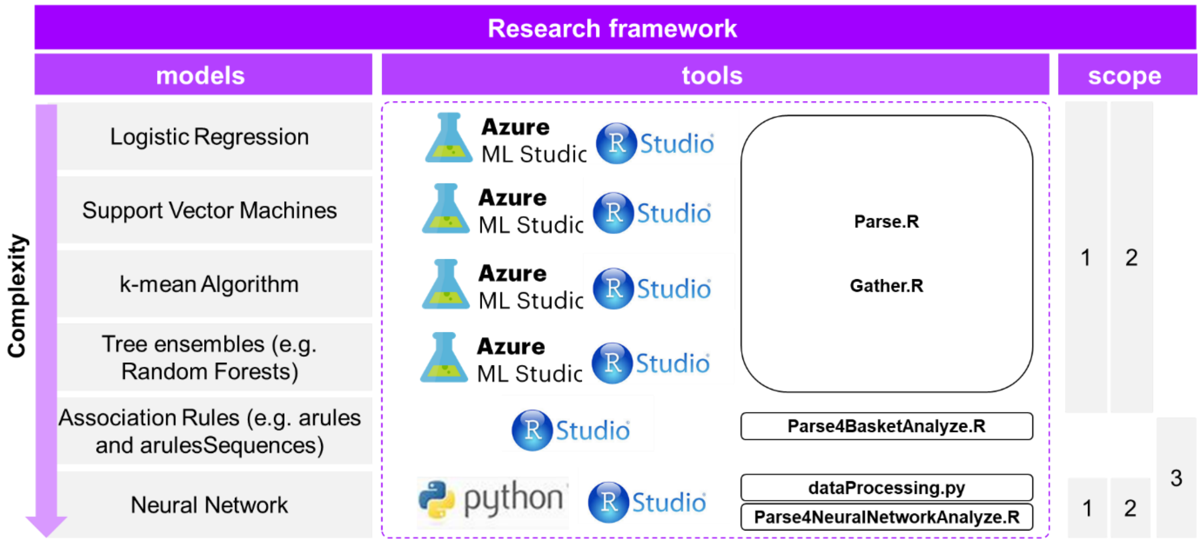


Figure 4-7: Research framework including data models, tools and research scope

Instead of relying on a single technology to process the underlying research objectives, the research thesis will use a combination of strategies like MS Azure ML Studio, R Studio, Python, Anaconda framework, Spyder and Jupyter applications including TensorFlow for efficient data analysis and data visualisation. MS Azure ML Studio[2] is a GUI-based, integrated and fully-managed cloud-based development environment for constructing and operationalising machine learning workflows. It enables us to easily build, deploy and share predictive analytics solutions.

[2] See detailed information about MS Azure ML Studio tool at https://studio.azureml.net/

R Studio[3] is an open-source data analysis software that is widely used by statisticians, data scientists and analysts for analysis, visualisation and predictive modelling. It allows writing scripts and functions in a complete, interactive, and object-oriented way. I have selected R Studio because it is a very efficient and easy-to-code data analysis and visualisation tool. As a result of this, it is primarily used for every research project in differing degrees, as described in figure 4-7. Other technology such as Python has been used in pre-processing the large-scale financial dataset by evaluating the third research project. An additional reason behind this subset of tool selection is that I am very familiar with these free open-source technologies.

The next section will provide an overview of the pre-processed datasets for further data analysis in every research project.

4.4. Data pre-processing and data exploration

Various data science techniques including data mining, data exploration and analysis are applied to help extract new customer behavioural insights and advanced information through prescriptive and descriptive analysis of the datasets. Berka and Rauch (2007) highlighted that “preprocessing is the most difficult and most time-consuming step” in the whole research process. The data pre-processing procedure comprises two important steps, namely extracting and/or building relevant target attributes for the data mining objectives and transforming the data in a suitable scheme for the machine learning algorithms to be used.

The data will also be explored using a range of machine learning techniques, neural networks and association rules. Regarding the research framework presented in the previous section, various models will be investigated with advanced or state-of-the-art techniques. This will additionally help reduce subjectivity from the research and help to detect in-depth insights that can then be further investigated when applying predictive models.

The following figures illustrates the pre-processing approaches with the given financial dataset from Berka applied for the proposed research projects.

[3] See more detailed information about R Studio at https://www.rstudio.com/

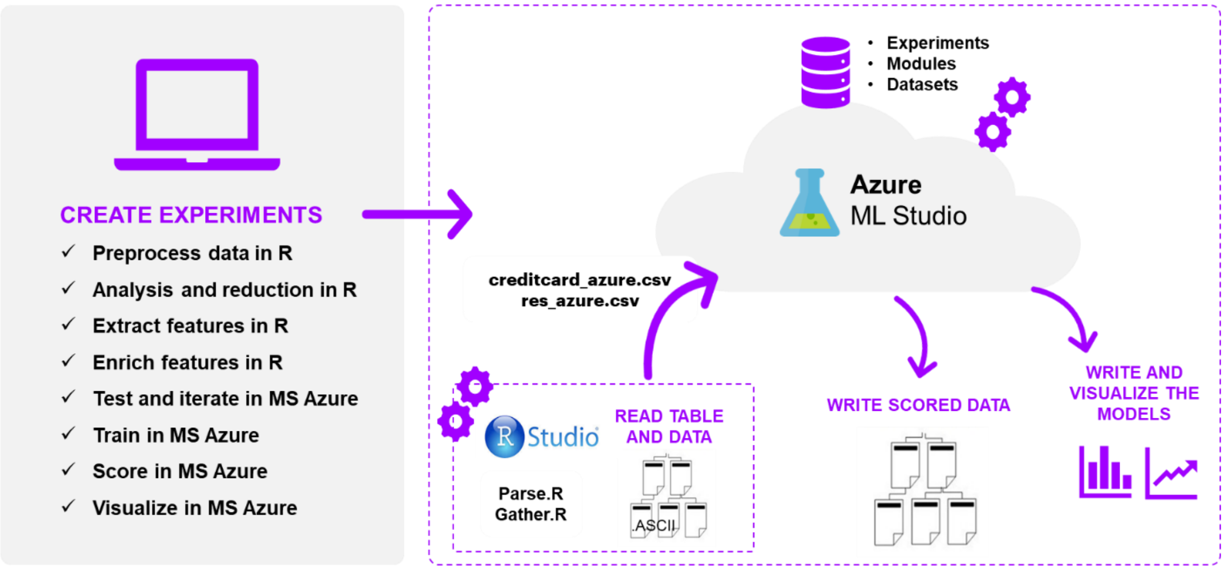


Figure 4-8: Overview of the methodology process for the 1st and 2nd research project

Regarding to the first and second research projects, the research process will use the R Studio tool for pre-processing and analysing the data, as well as the simple and scalable tool Microsoft Azure ML to test and train selected data models. The reason behind this approach was that this technology advances the point that machine learning capabilities can be deployed at scale with greater speed and efficiency for our research purposes. Thus, it enables a range of new data science capabilities such as innovative data visualisation techniques including the usage of various supervised learning algorithms while returning optimised evaluation results, processing various machine learning algorithm in real time, scoring predictive models, scaling to process millions of transactional data records efficiently (if required), and processing responses that are fed back into models for model recalibration.

Figure 4-9 below describes the research design for gathering, preparing, mining and modelling the data for enhanced predictions to evaluate the defined research questions from the third research project. The Anaconda framework including Python and the Spyder application was used to conduct the research process due to its ease of use for data processing as well as its high level of efficiency in complex and processing-intensive calculations. The reason behind this approach was the computing-intensive task for pre-processing the high amount (in particular, more than a half million records) of given transaction categories within the transactional datasets.

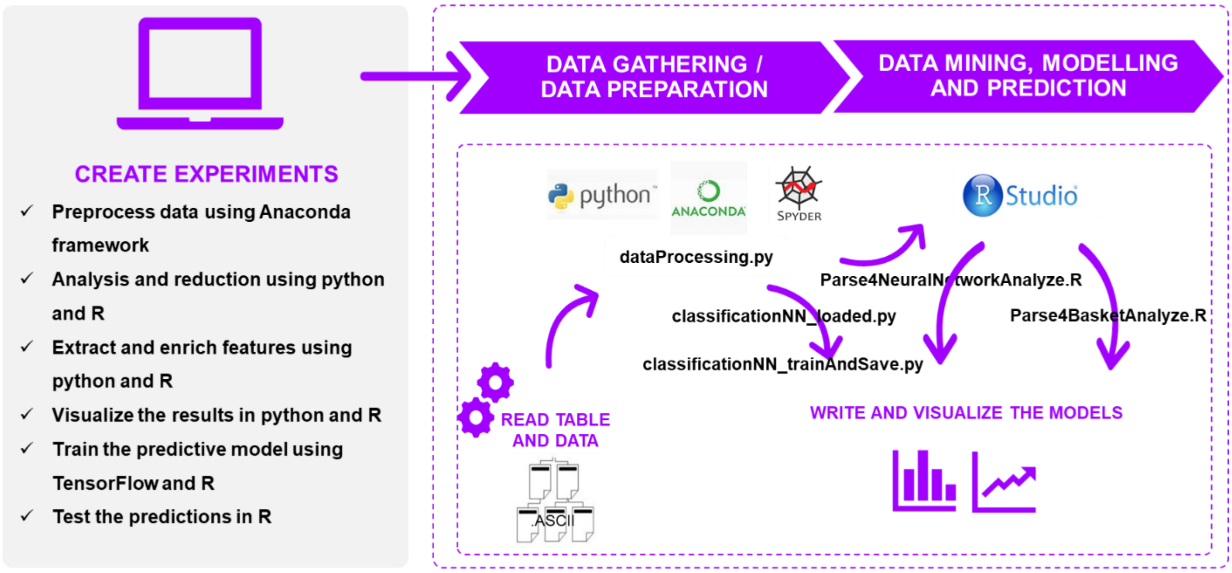

CREATE EXPERIMENTS
✓ Preprocess data using Anaconda framework
✓ Analysis and reduction using python and R
✓ Extract and enrich features using python and R
✓ Visualize the results in python and R
✓ Train the predictive model using TensorFlow and R
✓ Test the predictions in R
DATA GATHERING / DATA PREPARATION
DATA MINING, MODELLING AND PREDICTION
python
ANACONDA
SPYDER
RStudio
dataProcessing.py
Parse4NeuralNetworkAnalyze.R
classificationNN_loaded.py
Parse4BasketAnalyze.R
classificationNN_trainAndSave.py
READ TABLE AND DATA
ASCII
WRITE AND VISUALIZE THE MODELS

Figure 4-9: Overview of the methodology process for the 3rd research project

After pre-processing the data with the help of the Anaconda framework including Python and the Spyder application depicted in figure 4-9, the data was mined, modelled and predicted with TensorFlow. In addition, the powerful tool R Studio is also used for data analysis, data visualisation and prediction of the (un-)categorised payment transactions to minimise the subjectivity of the research results and make the evaluation results more comparable. The adopted two-pronged research approach finally assists in verifying the accuracy of the prediction results.

During the pre-processing and exploration phases, several interesting features of the original dataset were found, which will be explained for every research project in detail. However, the pre-processed datasets for the different research experiments are defined in the following sections.

4.4.1. Declaration of the dataset exploring the 1st research project - credit scoring

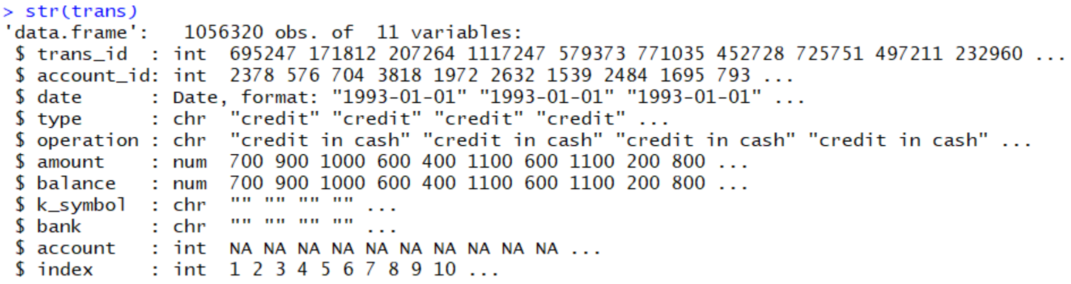
The pre-processing of the original dataset was conducted in two phases. The first phase involved parsing the raw dataset for data cleansing objectives, and then gathering step-by-step the cleaned dataset to compute new variables described in the tables below. According to the processing approach using R Studio, in the first exploration phase the research generates many temporary tables labelled as "…_adap" in the corresponding R scripts. As a result, various resulting temporary tables are depicted in the following figures. The corresponding R scripts for the pre-processing steps are given in the appendix E.1.

(1) We pre-processed “trans.csv” to the following temporary table below by transforming the attribute “date” into the format “YYYY-MM-DD” and translating Czech terms into English ones based on the values given in the attributes “type”, “operation”, “k_symbol”. Esssentially, the following mutations were performed:

“type” includes the Czech terms: "PRIJEM" assigned to "credit", "VYDAJ" assigned to "withdrawal", "VYBER" assigned to "withdrawal".

“operation” includes the Czech terms: "VYBER KARTOU" assigned to "credit card withdrawal", "VKLAD" assigned to "credit in cash", "PREVOD Z UCTU" assigned to "collection from another bank", "VYBER" assigned to "withdrawal in cash" and "PREVOD NA UCET" assigned to "remittance to another bank".

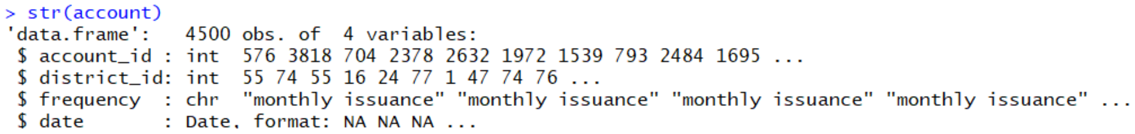
“k_symbol” includes the Czech terms: “POJISTNE” assigned to “insurance payment”, “SIPO” assigned to “household”, “SLUZBY” assigned to "payment for statement", "UVER" assigned to "loan payment", “UROK” assigned to "interest credited", "SANKC. UROK" assigned to "interest if negative balance" and “DUCHOD” assigned to "old-age pension".



```
> str(trans)
'data.frame':	1056320 obs. of  11 variables:
 $ trans_id  : int  695247 171812 207264 1117247 579373 771035 452728 725751 497211 232960 ...
 $ account_id: int  2378 576 704 3818 1972 2632 1539 2484 1695 793 ...
 $ date      : Date, format: "1993-01-01" "1993-01-01" "1993-01-01" ...
 $ type      : chr  "credit" "credit" "credit" "credit" ...
 $ operation : chr  "credit in cash" "credit in cash" "credit in cash" "credit in cash" ...
 $ amount    : num  700 900 1000 600 400 1100 600 1100 200 800 ...
 $ balance   : num  700 900 1000 600 400 1100 600 1100 200 800 ...
 $ k_symbol  : chr  "" "" "" "" ...
 $ bank      : chr  "" "" "" "" ...
 $ account   : int  NA NA NA NA NA NA NA NA NA NA ...
 $ index     : int  1 2 3 4 5 6 7 8 9 10 ...
```

Figure 4-10: Structure of data after pre-processing stage - transaction temporary table

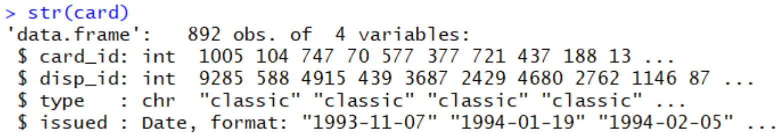
(2) We pre-processed “account.csv” to the following temporary table below by transforming the attribute “date” into the format “YYYY-MM-DD” and translating Czech terms into English one based on the values given in the attribute “frequency”. Therefore, the following mutations were performed: "POPLATEK MESICNE" assigned to "monthly issuance", "POPLATEK TYDNE" assigned to "weekly issuance" and "POPLATEK PO OBRATU" assigned to "issuance after transaction".



```
> str(account)
'data.frame':	4500 obs. of  4 variables:
 $ account_id : int  576 3818 704 2378 2632 1972 1539 793 2484 1695 ...
 $ district_id: int  55 74 55 16 24 77 1 47 74 76 ...
 $ frequency  : chr  "monthly issuance" "monthly issuance" "monthly issuance" "monthly issuance" ...
 $ date       : Date, format: NA NA NA ...
```

Figure 4-11: Structure of data after pre-processing stage - account temporary table

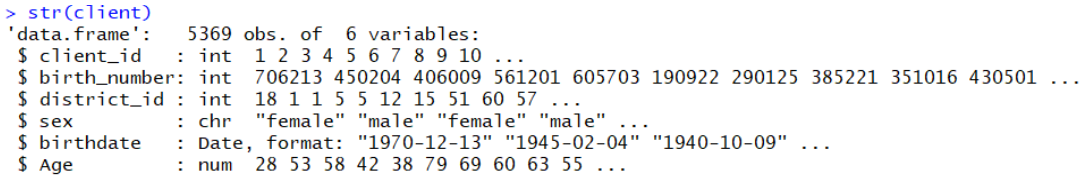
(3) We pre-processed “card.csv” to the following temporary table below by primarily transforming the attribute “issued” into a date format such as “YYYY-MM-DD”.



```
> str(card)
'data.frame':	892 obs. of  4 variables:
 $ card_id: int  1005 104 747 70 577 377 721 437 188 13 ...
 $ disp_id: int  9285 588 4915 439 3687 2429 4680 2762 1146 87 ...
 $ type   : chr  "classic" "classic" "classic" "classic" ...
 $ issued : Date, format: "1993-11-07" "1994-01-19" "1994-02-05" ...
```

Figure 4-12: Structure of data after pre-processing stage - card temporary table

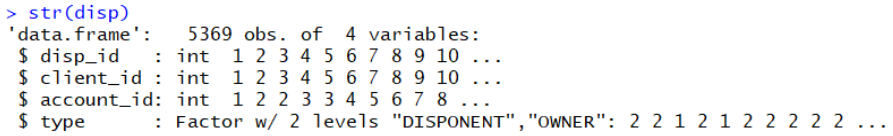
(4) We pre-processed “client.csv” to the following temporary table below by mutating the given values in the attribute “birth_number” as follows: The “birth number” was transformed with the help of the implemented “GetBirthdate” function as well as the “GetSex” function to three additional attributes: sex, birth date and age of a client.



```
> str(client)
'data.frame':	5369 obs. of  6 variables:
 $ client_id   : int  1 2 3 4 5 6 7 8 9 10 ...
 $ birth_number: int  706213 450204 406009 561201 605703 190922 290125 385221 351016 430501 ...
 $ district_id : int  18 1 1 5 5 12 15 51 60 57 ...
 $ sex         : chr  "female" "male" "female" "male" ...
 $ birthdate   : Date, format: "1970-12-13" "1945-02-04" "1940-10-09" ...
 $ Age         : num  28 53 58 42 38 79 69 60 63 55 ...
```

Figure 4-13: Structure of data after pre-processing stage - client temporary table

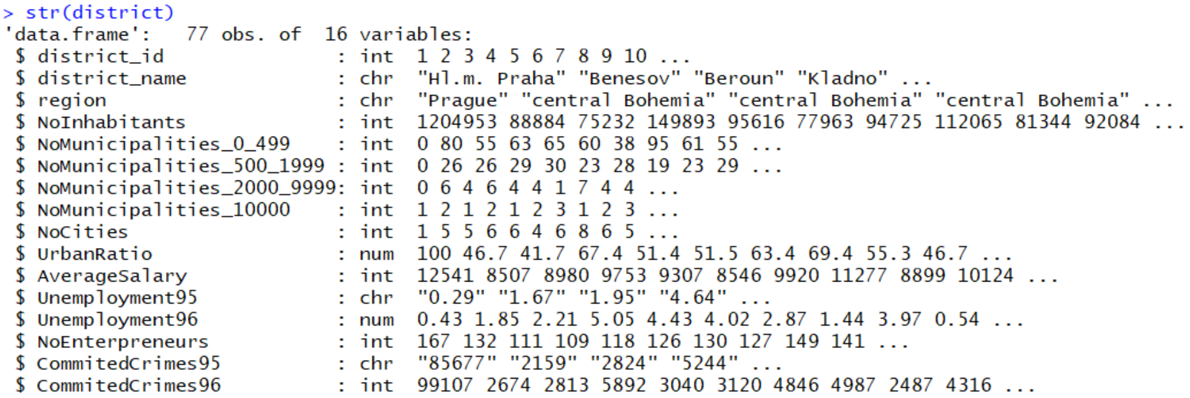
(5) We pre-processed "disp.csv" to the following temporary table below without any further transformation in the first instance.



```
> str(disp)
'data.frame':	5369 obs. of  4 variables:
 $ disp_id   : int  1 2 3 4 5 6 7 8 9 10 ...
 $ client_id : int  1 2 3 4 5 6 7 8 9 10 ...
 $ account_id: int  1 2 2 3 3 4 5 6 7 8 ...
 $ type      : Factor w/ 2 levels "DISPONENT","OWNER": 2 2 1 2 1 2 2 2 2 2 ...
```

Figure 4-14: Structure of data after pre-processing stage - disposition temporary table

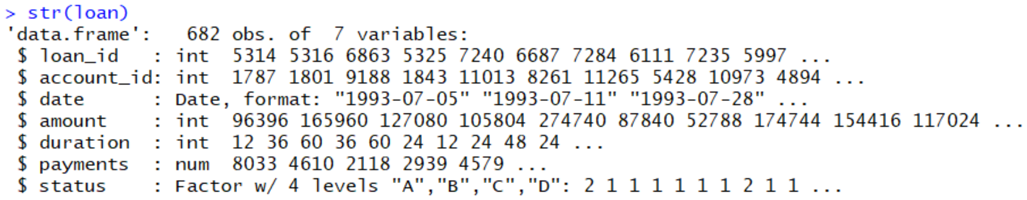
(6) We pre-processed "district.csv" to the following temporary table below by labelling all columns of the raw table with appropriate names and transforming missing values within the attributes “Unemployment95” and “CommitedCrimes95” into “NA” values.



```
> str(district)
'data.frame':	77 obs. of  16 variables:
 $ district_id                : int  1 2 3 4 5 6 7 8 9 10 ...
 $ district_name              : chr  "Hl.m. Praha" "Benesov" "Beroun" "Kladno" ...
 $ region                     : chr  "Prague" "central Bohemia" "central Bohemia" "central Bohemia" ...
 $ NoInhabitants              : int  1204953 88884 75232 149893 95616 77963 94725 112065 81344 92084 ...
 $ NoMunicipalities_0_499     : int  0 80 55 63 65 60 38 95 61 55 ...
 $ NoMunicipalities_500_1999  : int  0 26 26 29 30 23 28 19 23 29 ...
 $ NoMunicipalities_2000_9999: int  0 6 4 6 4 4 1 7 4 4 ...
 $ NoMunicipalities_10000     : int  1 2 1 2 1 2 3 1 2 3 ...
 $ NoCities                   : int  1 5 5 6 6 4 6 8 6 5 ...
 $ UrbanRatio                 : num  100 46.7 41.7 67.4 51.4 51.5 63.4 69.4 55.3 46.7 ...
 $ AverageSalary              : int  12541 8507 8980 9753 9307 8546 9920 11277 8899 10124 ...
 $ Unemployment95             : chr  "0.29" "1.67" "1.95" "4.64" ...
 $ Unemployment96             : num  0.43 1.85 2.21 5.05 4.43 4.02 2.87 1.44 3.97 0.54 ...
 $ NoEnterpreneurs            : int  167 132 111 109 118 126 130 127 149 141 ...
 $ CommitedCrimes95           : chr  "85677" "2159" "2824" "5244" ...
 $ CommitedCrimes96           : int  99107 2674 2813 5892 3040 3120 4846 4987 2487 4316 ...
```

Figure 4-15: Structure of data after pre-processing stage - district temporary table

(7) We pre-processed "loan.csv" to the following temporary table below by transforming the attribute “date” into the format “YYYY-MM-DD”.

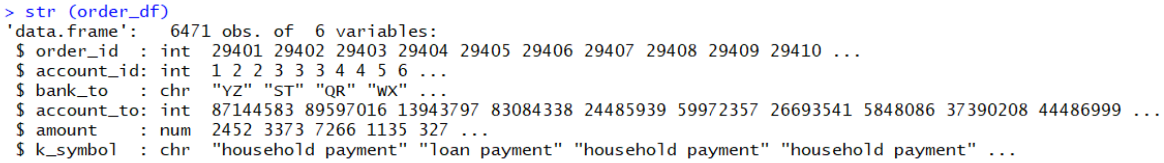


```
> str(loan)
'data.frame':   682 obs. of  7 variables:
 $ loan_id   : int  5314 5316 6863 5325 7240 6687 7284 6111 7235 5997 ...
 $ account_id: int  1787 1801 9188 1843 11013 8261 11265 5428 10973 4894 ...
 $ date      : Date, format: "1993-07-05" "1993-07-11" "1993-07-28" ...
 $ amount    : int  96396 165960 127080 105804 274740 87840 52788 174744 154416 117024 ...
 $ duration  : int  12 36 60 36 60 24 12 24 48 24 ...
 $ payments  : num  8033 4610 2118 2939 4579 ...
 $ status    : Factor w/ 4 levels "A","B","C","D": 2 1 1 1 1 1 1 2 1 1 ...
```

Figure 4-16: Structure of data after pre-processing stage - loan temporary table

Regarding the first research project, the dataset describes clients causing problems and losses with bad loans. Weber (1998) stated that “bad loans are loans still running or finished with problems, good loans are all loans finished or still running without problems”. “The consequence is that we aimed only on data of clients with granted loan” (Coufal, Holeňa and Sochorová, 1999). This is also why the prepared dataset comprises only 682 loans, 76 of which are bad. Miksovsky, Zelezny, Stepankova, Pechoucek (1999) highlighted that the amount of data is not sufficient for a larger number of attributes, although only those records that can be potentially interesting for a data analysis to measure the performance of various applied predictive models are extractable. Note that every client owns exactly one account (Coufal, Holeňa and Sochorová, 1999).

(8) We pre-processed "order.csv" to the following temporary table below by translating Czech terms into English ones based on the values given in the attribute “k_symbol”. Therefore, the following mutations were performed: "POJISTNE" assigned to "insurance payment", "SIPO" assigned to "household payment", "LEASING" assigned to "leasing" and "UVER" assigned to "loan payment".



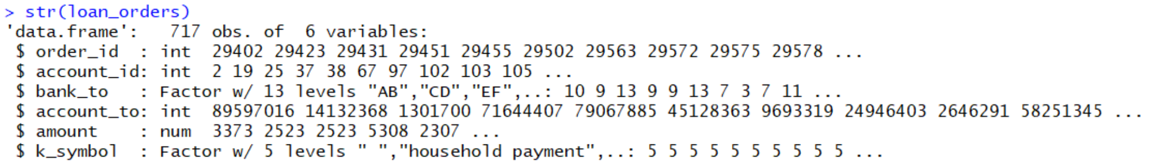
```
> str (order_df)
'data.frame':   6471 obs. of  6 variables:
 $ order_id  : int  29401 29402 29403 29404 29405 29406 29407 29408 29409 29410 ...
 $ account_id: int  1 2 2 3 3 3 4 4 5 6 ...
 $ bank_to   : chr  "YZ" "ST" "QR" "WX" ...
 $ account_to: int  87144583 89597016 13943797 83084338 24485939 59972357 26693541 5848086 37390208 44486999 ...
 $ amount    : num  2452 3373 7266 1135 327 ...
 $ k_symbol  : chr  "household payment" "loan payment" "household payment" "household payment" ...
```

Figure 4-17: Structure of data after pre-processing stage - order temporary table

Regarding the second phase of the pre-processing stage, the parsed raw dataset with its temporary tables will be gathered in terms of enhancing the data cleansing processes as well as aggregating the attributes with the aim of accessing one final dataset for research analysis purposes. Based on the temporary tables created from the original Berka dataset, the following steps were pre-processed:

(1) We processed all eight temporary tables for further preparation in terms of data-gathering. Hence, we mutated every single record for the attribute “amount” within the transaction table if the attribute “type” comprises a “withdrawal”. In addition, we created one further attribute “Month”, which represents the month and year of every single transaction ID.

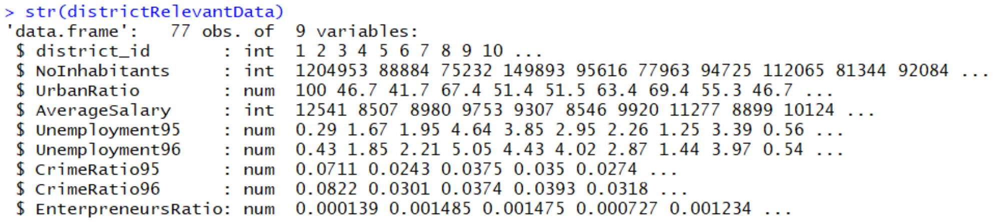
(2) We constructed helping data frames from the temporary tables “order” and “district”. In the first step, we created a table called “loan_orders” by filtering only the value “loan payment” within the attribute “k_symbol” from the loan temporary table. Finally, the resulting “loan_orders” table comprises 717 records with six attributes.



```
> str(loan_orders)
'data.frame':	717 obs. of  6 variables:
 $ order_id  : int  29402 29423 29431 29451 29455 29502 29563 29572 29575 29578 ...
 $ account_id: int  2 19 25 37 38 67 97 102 103 105 ...
 $ bank_to   : Factor w/ 13 levels "AB","CD","EF",..: 10 9 13 9 9 13 7 3 7 11 ...
 $ account_to: int  89597016 14132368 1301700 71644407 79067885 45128363 9693319 24946403 2646291 58251345 ...
 $ amount    : num  3373 2523 2523 5308 2307 ...
 $ k_symbol  : Factor w/ 5 levels " ","household payment",..: 5 5 5 5 5 5 5 5 5 5 ...
```

Figure 4-18: Structure of data after data-gathering stage - loan_orders table

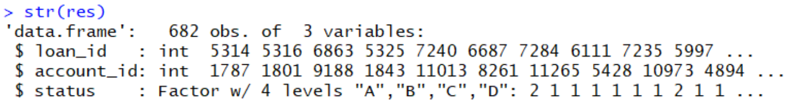
(3) The next step was processed to create district-relevant data by mutating the attributes “CrimeRatio95”, “CrimeRatio96” and “EnterpreneursRatio” and choosing the number of selected attributes of the temporary table district. Finally, the resulting “districtRelevantData” table comprises 77 records with seven attributes.



```
> str(districtRelevantData)
'data.frame':	77 obs. of  9 variables:
 $ district_id       : int  1 2 3 4 5 6 7 8 9 10 ...
 $ NoInhabitants     : int  1204953 88884 75232 149893 95616 77963 94725 112065 81344 92084 ...
 $ UrbanRatio        : num  100 46.7 41.7 67.4 51.4 51.5 63.4 69.4 55.3 46.7 ...
 $ AverageSalary     : int  12541 8507 8980 9753 9307 8546 9920 11277 8899 10124 ...
 $ Unemployment95    : num  0.29 1.67 1.95 4.64 3.85 2.95 2.26 1.25 3.39 0.56 ...
 $ Unemployment96    : num  0.43 1.85 2.21 5.05 4.43 4.02 2.87 1.44 3.97 0.54 ...
 $ CrimeRatio95      : num  0.0711 0.0243 0.0375 0.035 0.0274 ...
 $ CrimeRatio96      : num  0.0822 0.0301 0.0374 0.0393 0.0318 ...
 $ EnterpreneursRatio: num  0.000139 0.001485 0.001475 0.000727 0.001234 ...
```

Figure 4-19: Structure of data after data-gathering stage - districtRelevantData table

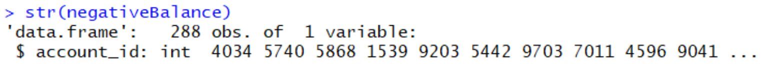
(4) In the next step, we initialised the resulting data frame called “res” by selecting the attributes ‘loan_id’, ‘account_id’ and ‘status’ from the temporary table “loan”. Finally, the resulting “res” table comprises 682 records with three attributes.



```
> str(res)
'data.frame':	682 obs. of  3 variables:
 $ loan_id   : int  5314 5316 6863 5325 7240 6687 7284 6111 7235 5997 ...
 $ account_id: int  1787 1801 9188 1843 11013 8261 11265 5428 10973 4894 ...
 $ status    : Factor w/ 4 levels "A","B","C","D": 2 1 1 1 1 1 1 2 1 1 ...
```

Figure 4-20: Structure of data after data-gathering stage - initialised credit scoring table res (1)

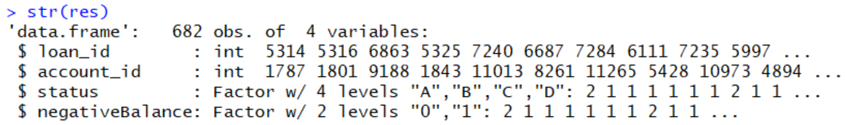
(5) We implemented an indicator to verify whether the account has a negative balance. Finally, the resulting “negativeBalance” table comprises 288 records with all existing accounts with a negative balance.



```
> str(negativeBalance)
'data.frame':   288 obs. of  1 variable:
 $ account_id: int  4034 5740 5868 1539 9203 5442 9703 7011 4596 9041 ...
```

Figure 4-21: Structure of data after data-gathering stage - negativeBalance table

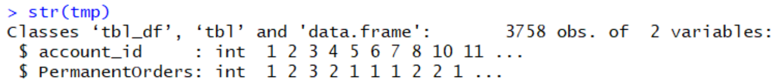
(6) The latest structure of the data frame “res” and the temporary table “trans” was used for further computations to add an additional attribute called “negativeBalance” to the recent data frame “res”.



```
> str(res)
'data.frame':   682 obs. of  4 variables:
 $ loan_id        : int  5314 5316 6863 5325 7240 6687 7284 6111 7235 5997 ...
 $ account_id     : int  1787 1801 9188 1843 11013 8261 11265 5428 10973 4894 ...
 $ status         : Factor w/ 4 levels "A","B","C","D": 2 1 1 1 1 1 1 2 1 1 ...
 $ negativeBalance: Factor w/ 2 levels "0","1": 2 1 1 1 1 1 1 2 1 1 ...
```

Figure 4-22: Structure of data after data-gathering stage - initialised credit scoring table res (2)

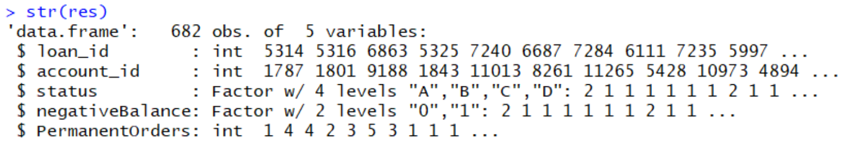
(7) In the following pre-processing step, we calculated the number of permanent orders by joining the temporary table “loan” with the temporary table “order”. For this purpose, we built a temporary table called “tmp” to summarise all existing permanent orders for every account ID.



```
> str(tmp)
Classes ‘tbl_df’, ‘tbl’ and 'data.frame':       3758 obs. of  2 variables:
 $ account_id     : int  1 2 3 4 5 6 7 8 10 11 ...
 $ PermanentOrders: int  1 2 3 2 1 1 1 2 2 1 ...
```

Figure 4-23: Structure of data after data-gathering stage - temporary table for number of orders

Thus, we joined the temporary table “tmp” with the other both temporary tables “loan” and “order” with the goal of adding an additional attribute called “PermanentOrders” to the latest data frame “res”.

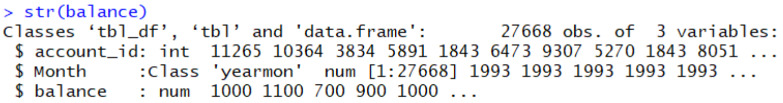


```
> str(res)
'data.frame':   682 obs. of  5 variables:
 $ loan_id        : int  5314 5316 6863 5325 7240 6687 7284 6111 7235 5997 ...
 $ account_id     : int  1787 1801 9188 1843 11013 8261 11265 5428 10973 4894 ...
 $ status         : Factor w/ 4 levels "A","B","C","D": 2 1 1 1 1 1 1 2 1 1 ...
 $ negativeBalance: Factor w/ 2 levels "0","1": 2 1 1 1 1 1 1 2 1 1 ...
 $ PermanentOrders: int  1 4 4 2 3 5 3 1 1 1 ...
```

Figure 4-24: Structure of data after data-gathering stage - initialised credit scoring table res (3)

After this pre-processing stage, the initialised table includes 682 records with five attributes.

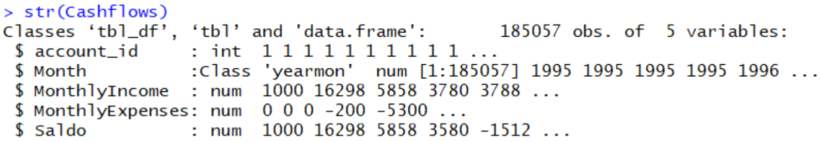
(8) The next step involved calculating the balance at end of month by joining the temporary tables “loan” and “trans”. Finally, the resulting “balance” table comprises 27,668 records with 3 attributes.



```
> str(balance)
Classes ‘tbl_df’, ‘tbl’ and 'data.frame':       27668 obs. of  3 variables:
 $ account_id: int  11265 10364 3834 5891 1843 6473 9307 5270 1843 8051 ...
 $ Month     :Class 'yearmon'  num [1:27668] 1993 1993 1993 1993 1993 ...
 $ balance   : num  1000 1100 700 900 1000 ...
```

Figure 4-25: Structure of data after data-gathering stage - balance table

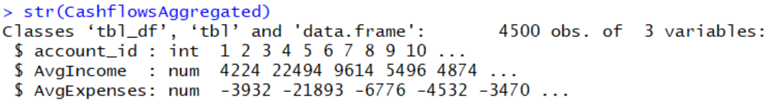
(9) We then calculated the balance to loan payment by looking at average income and expenses given in the temporary table “trans”. The resulting “cashflow” table bases on the created attributes “Month”, “MonthlyIncome”, “MonthlyExpenses” and “Saldo”. Finally, the resulting table “Cashflows” comprises 185,057 records with five attributes.



```
> str(Cashflows)
Classes ‘tbl_df’, ‘tbl’ and 'data.frame':       185057 obs. of  5 variables:
 $ account_id     : int  1 1 1 1 1 1 1 1 1 1 ...
 $ Month          :Class 'yearmon'  num [1:185057] 1995 1995 1995 1995 1996 ...
 $ MonthlyIncome  : num  1000 16298 5858 3780 3788 ...
 $ MonthlyExpenses: num  0 0 0 -200 -5300 ...
 $ Saldo          : num  1000 16298 5858 3580 -1512 ...
```

Figure 4-26: Structure of data after data-gathering stage - cashflows table

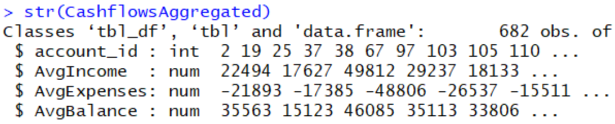
Moreover, we created two additional attributes “AvgIncome” and “AvgExpenses” based on the “Cashflows” table and added these attributes into a further resulting table called “CashflowsAggregated”. The structure of this table is displayed below, and it comprises 4,500 records with three attributes.



```
> str(CashflowsAggregated)
Classes ‘tbl_df’, ‘tbl’ and 'data.frame':       4500 obs. of  3 variables:
 $ account_id : int  1 2 3 4 5 6 7 8 9 10 ...
 $ AvgIncome  : num  4224 22494 9614 5496 4874 ...
 $ AvgExpenses: num  -3932 -21893 -6776 -4532 -3470 ...
```

Figure 4-27: Structure of data after data-gathering stage - cashflowsAggregated table (1)

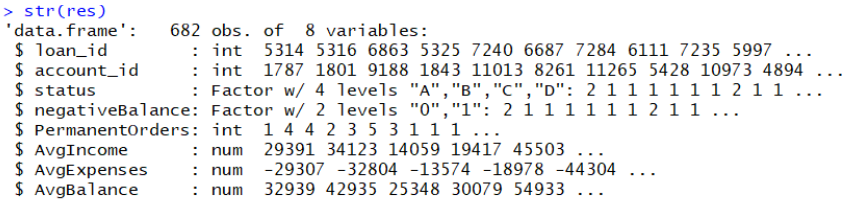
Finally, we created another temporary table “tmp” to calculate an additional attribute labelled “AvgBalance” to join this information with the recent resulting table “CashflowsAggregated”. The structure of this table is shown below, and it comprises 682 records with four attributes.



```
> str(CashflowsAggregated)
Classes ‘tbl_df’, ‘tbl’ and 'data.frame':       682 obs. of  4 variables:
 $ account_id : int  2 19 25 37 38 67 97 103 105 110 ...
 $ AvgIncome  : num  22494 17627 49812 29237 18133 ...
 $ AvgExpenses: num  -21893 -17385 -48806 -26537 -15511 ...
 $ AvgBalance : num  35563 15123 46085 35113 33806 ...
```

Figure 4-28: Structure of data after data-gathering stage - cashflowsAggregated table (2)

The last step within the extraordinary pre-processing steps was to add the data gathered for the resulting table “CashflowsAggregated” to the table “res”.

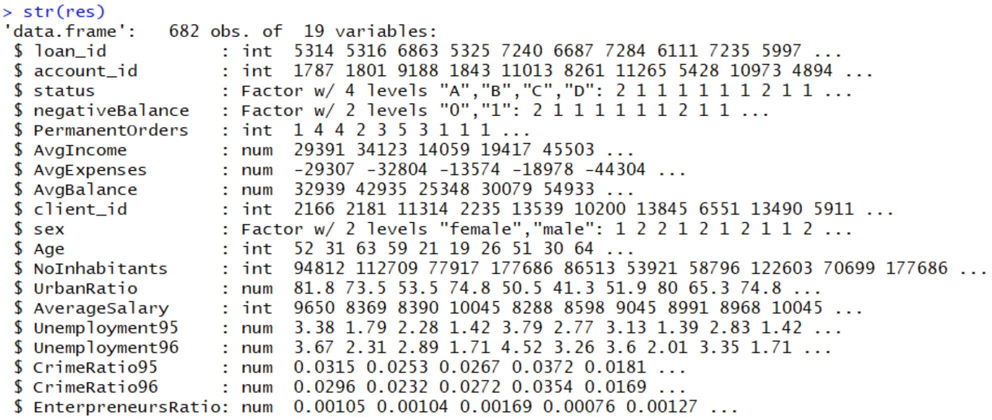


```
> str(res)
'data.frame':   682 obs. of  8 variables:
 $ loan_id        : int  5314 5316 6863 5325 7240 6687 7284 6111 7235 5997 ...
 $ account_id     : int  1787 1801 9188 1843 11013 8261 11265 5428 10973 4894 ...
 $ status         : Factor w/ 4 levels "A","B","C","D": 2 1 1 1 1 1 1 2 1 1 ...
 $ negativeBalance: Factor w/ 2 levels "0","1": 2 1 1 1 1 1 1 2 1 1 ...
 $ PermanentOrders: int  1 4 4 2 3 5 3 1 1 1 ...
 $ AvgIncome      : num  29391 34123 14059 19417 45503 ...
 $ AvgExpenses    : num  -29307 -32804 -13574 -18978 -44304 ...
 $ AvgBalance     : num  32939 42935 25348 30079 54933 ...
```

Figure 4-29: Structure of data after data-gathering stage - initialised credit scoring table res (4)

The initialised credit scoring table now comprises 682 records with eight attributes.

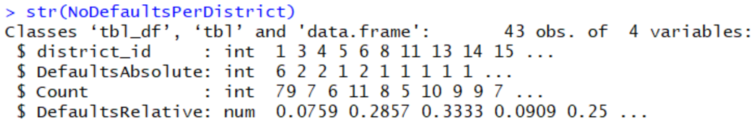
(10) The next pre-processing step was to add demographic data to the recent built “res” table. As an interim step in the pre-processing phase, we created a “GetAge” function to use in an intermediate step to create a temporary table “tmp” to join with the resulting table “districtRelevantData”. Finally, the latest resulting table “res” comprises 682 records with nineteen attributes.



```
> str(res)
'data.frame':   682 obs. of  19 variables:
 $ loan_id          : int  5314 5316 6863 5325 7240 6687 7284 6111 7235 5997 ...
 $ account_id       : int  1787 1801 9188 1843 11013 8261 11265 5428 10973 4894 ...
 $ status           : Factor w/ 4 levels "A","B","C","D": 2 1 1 1 1 1 1 2 1 1 ...
 $ negativeBalance  : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 1 2 1 1 ...
 $ PermanentOrders  : int  1 4 4 2 3 5 3 1 1 1 ...
 $ AvgIncome        : num  29391 34123 14059 19417 45503 ...
 $ AvgExpenses      : num  -29307 -32804 -13574 -18978 -44304 ...
 $ AvgBalance       : num  32939 42935 25348 30079 54933 ...
 $ client_id        : int  2166 2181 11314 2235 13539 10200 13845 6551 13490 5911 ...
 $ sex              : Factor w/ 2 levels "female","male": 1 2 2 1 2 1 2 1 1 2 ...
 $ Age              : int  52 31 63 59 21 19 26 51 30 64 ...
 $ NoInhabitants    : int  94812 112709 77917 177686 86513 53921 58796 122603 70699 177686 ...
 $ UrbanRatio       : num  81.8 73.5 53.5 74.8 50.5 41.3 51.9 80 65.3 74.8 ...
 $ AverageSalary    : int  9650 8369 8390 10045 8288 8598 9045 8991 8968 10045 ...
 $ Unemployment95   : num  3.38 1.79 2.28 1.42 3.79 2.77 3.13 1.39 2.83 1.42 ...
 $ Unemployment96   : num  3.67 2.31 2.89 1.71 4.52 3.26 3.6 2.01 3.35 1.71 ...
 $ CrimeRatio95     : num  0.0315 0.0253 0.0267 0.0372 0.0181 ...
 $ CrimeRatio96     : num  0.0296 0.0232 0.0272 0.0354 0.0169 ...
 $ EnterpreneursRatio: num  0.00105 0.00104 0.00169 0.00076 0.00127 ...
```

Figure 4-30: Structure of data after data-gathering stage - initialised credit scoring table res (5)

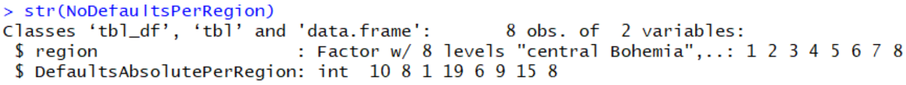
(11) The next pre-processing step involved calculating the number of defaults per district based on two helping data frames “NoDefaultsPerDistrict” and “NoLoansPerDistrict”. The resulting table “NoDefaultsPerDistrict” comprises 43 records with four attributes.



```
> str(NoDefaultsPerDistrict)
Classes ‘tbl_df’, ‘tbl’ and 'data.frame':       43 obs. of  4 variables:
 $ district_id     : int  1 3 4 5 6 8 11 13 14 15 ...
 $ DefaultsAbsolute: int  6 2 2 1 2 1 1 1 1 1 ...
 $ Count           : int  79 7 6 11 8 5 10 9 9 7 ...
 $ DefaultsRelative: num  0.0759 0.2857 0.3333 0.0909 0.25 ...
```

Figure 4-31: Structure of data after data-gathering stage - NoDefaultsPerDistrict table

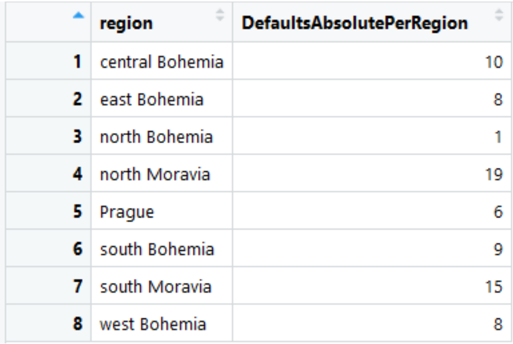
(12) The following pre-processing step used the same approach for calculating the number of defaults per region. As an intermediate step, the structure and the output results of the feature “NoDefaultsPerRegion” are shown below.



```
> str(NoDefaultsPerRegion)
Classes ‘tbl_df’, ‘tbl’ and 'data.frame':	8 obs. of  2 variables:
 $ region                   : Factor w/ 8 levels "central Bohemia",..: 1 2 3 4 5 6 7 8
 $ DefaultsAbsolutePerRegion: int  10 8 1 19 6 9 15 8
```

Figure 4-32: Structure of data after data-gathering stage - NoDefaultsPerRegion table

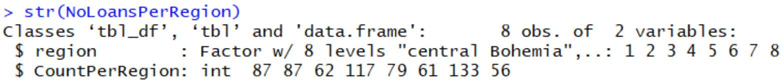
Figure 4-33 below displays the output results for NoDefaultsPerRegion in detail.



	region	DefaultsAbsolutePerRegion
1	central Bohemia	10
2	east Bohemia	8
3	north Bohemia	1
4	north Moravia	19
5	Prague	6
6	south Bohemia	9
7	south Moravia	15
8	west Bohemia	8

Figure 4-33: View the NoDefaultsPerRegion table

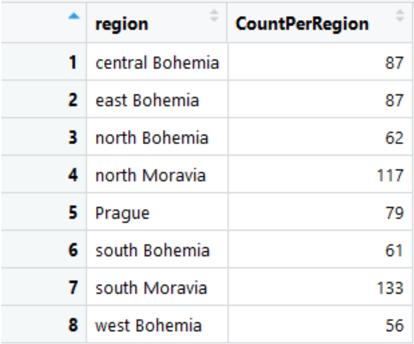
Second, the structure and the output results of the feature “NoLoansPerRegion” are shown below.



```
> str(NoLoansPerRegion)
Classes ‘tbl_df’, ‘tbl’ and 'data.frame':	8 obs. of  2 variables:
 $ region        : Factor w/ 8 levels "central Bohemia",..: 1 2 3 4 5 6 7 8
 $ CountPerRegion: int  87 87 62 117 79 61 133 56
```

Figure 4-34: Structure of data after data-gathering stage - NoLoansPerRegion table

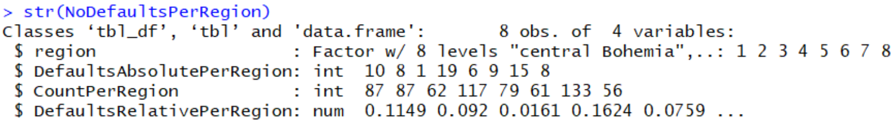
Figure 4-35 below displays the output results for NoLoansPerRegion in detail.



	region	CountPerRegion
1	central Bohemia	87
2	east Bohemia	87
3	north Bohemia	62
4	north Moravia	117
5	Prague	79
6	south Bohemia	61
7	south Moravia	133
8	west Bohemia	56

Figure 4-35: View the NoLoansPerRegion table

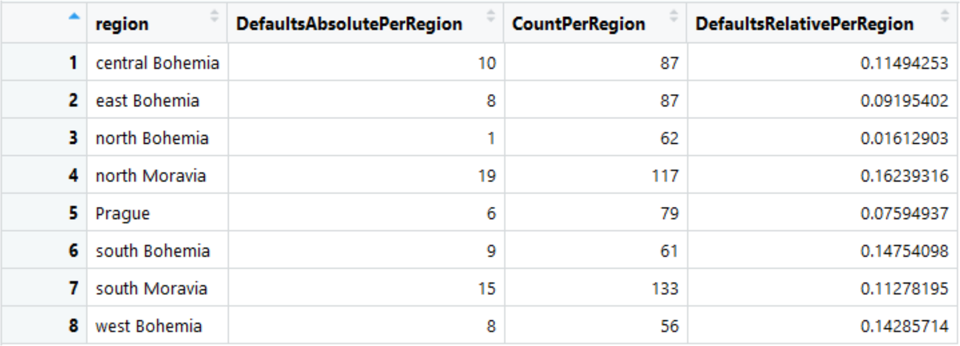
Finally, the resulting table “AbsolutNoDefaultsPerRegion” comprises eight records with four attributes.



```
> str(NoDefaultsPerRegion)
Classes ‘tbl_df’, ‘tbl’ and 'data.frame':	8 obs. of  4 variables:
 $ region                   : Factor w/ 8 levels "central Bohemia",..: 1 2 3 4 5 6 7 8
 $ DefaultsAbsolutePerRegion: int  10 8 1 19 6 9 15 8
 $ CountPerRegion           : int  87 87 62 117 79 61 133 56
 $ DefaultsRelativePerRegion: num  0.1149 0.092 0.0161 0.1624 0.0759 ...
```

Figure 4-36: Structure of data after data-gathering stage - AbsolutNoDefaultsPerRegion table

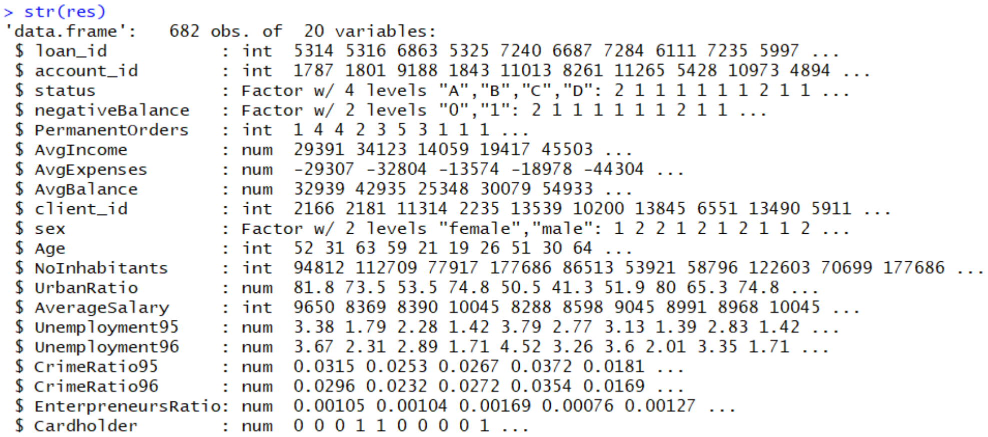
The figure below displays the output results for AbsolutNoDefaultsPerRegion in detail.



	region	DefaultsAbsolutePerRegion	CountPerRegion	DefaultsRelativePerRegion
1	central Bohemia	10	87	0.11494253
2	east Bohemia	8	87	0.09195402
3	north Bohemia	1	62	0.01612903
4	north Moravia	19	117	0.16239316
5	Prague	6	79	0.07594937
6	south Bohemia	9	61	0.14754098
7	south Moravia	15	133	0.11278195
8	west Bohemia	8	56	0.14285714

Figure 4-37: View the AbsolutNoDefaultsPerRegion table

(13) Finally, the structure of the final credit scoring dataset for the underlying research objectives is as follows.



```
> str(res)
'data.frame':   682 obs. of  20 variables:
 $ loan_id           : int  5314 5316 6863 5325 7240 6687 7284 6111 7235 5997 ...
 $ account_id        : int  1787 1801 9188 1843 11013 8261 11265 5428 10973 4894 ...
 $ status            : Factor w/ 4 levels "A","B","C","D": 2 1 1 1 1 1 1 2 1 1 ...
 $ negativeBalance   : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 1 2 1 1 ...
 $ PermanentOrders   : int  1 4 4 2 3 5 3 1 1 1 ...
 $ AvgIncome         : num  29391 34123 14059 19417 45503 ...
 $ AvgExpenses       : num  -29307 -32804 -13574 -18978 -44304 ...
 $ AvgBalance        : num  32939 42935 25348 30079 54933 ...
 $ client_id         : int  2166 2181 11314 2235 13539 10200 13845 6551 13490 5911 ...
 $ sex               : Factor w/ 2 levels "female","male": 1 2 2 1 2 1 2 1 1 2 ...
 $ Age               : int  52 31 63 59 21 19 26 51 30 64 ...
 $ NoInhabitants     : int  94812 112709 77917 177686 86513 53921 58796 122603 70699 177686 ...
 $ UrbanRatio        : num  81.8 73.5 53.5 74.8 50.5 41.3 51.9 80 65.3 74.8 ...
 $ AverageSalary     : int  9650 8369 8390 10045 8288 8598 9045 8991 8968 10045 ...
 $ Unemployment95    : num  3.38 1.79 2.28 1.42 3.79 2.77 3.13 1.39 2.83 1.42 ...
 $ Unemployment96    : num  3.67 2.31 2.89 1.71 4.52 3.26 3.6 2.01 3.35 1.71 ...
 $ CrimeRatio95      : num  0.0315 0.0253 0.0267 0.0372 0.0181 ...
 $ CrimeRatio96      : num  0.0296 0.0232 0.0272 0.0354 0.0169 ...
 $ EntrepreneursRatio: num  0.00105 0.00104 0.00169 0.00076 0.00127 ...
 $ Cardholder        : num  0 0 0 1 1 0 0 0 0 1 ...
```

Figure 4-38: Structure of data after data-gathering stage - initialised credit scoring table res (final)

Table 4-1 summarises the final dataset for the first research project, which includes 20 attributes relating to 682 accounts. The main efforts were to define new useful attributes whose constellation has not already been investigated by another researcher. The novelty and contribution to research is a selection of possibly useful attributes to assess the performance of various algorithms in predicting a reliable credit score, as well as building a cost-sensitive data model for this purpose.

	relation from original Berka dataset	Attribute	explanation	values
1	loan	loan_id	unique id for the loan	4959 - 7308
2	account	account_id	unique id for the account	2 - 11362
3	loan	status	good = {a, c}, bad = {b, d}	A, B, C, D

4	transaction	negativeBalance	indicator whether a negative balance exists for the account	1 = yes or 0 = no
5	order	permanentOrders	number of permanent orders exists for the account	1 - 5
6	transaction	avgIncome	average of incomes of the account owner	5191.16 - 75197.18
7	transaction	avgExpenses	average of expenses of the account owner	-4938.75 - 72330.92
8	transaction	avgBalance	average of balance of the account owner	7716.37 - 63647.53
9	client	client_id	unique id for the client	2 - 13971
10	client	sex	sex of the account owner	male, female
11	client	age	year of the account owner	19 - 64
12	district	noInhabitants	number of the inhabitants of account's owner district	42821 - 1204953
13	district	urbanRatio	rate of urbanity of account's owner district	33.9 - 100.0
14	transaction	averageSalary	average salary of the account owner	8110 - 12541
15	district	unemployment95	rate of unemployment of account's owner district	0.29 - 7.34
16	district	unemployment96	rate of unemployment of account's owner district	0.43 - 9.4
17	district	crimeRatio95	rate of crime of account's owner district	0.041 - 0.071
18	district	crimeRatio96	rate of crime of account's owner district	0.039 - 0.082
19	district	enterpreneursRatio	rate of entrepreneurs of account's owner district	0.00013 - 0.00289

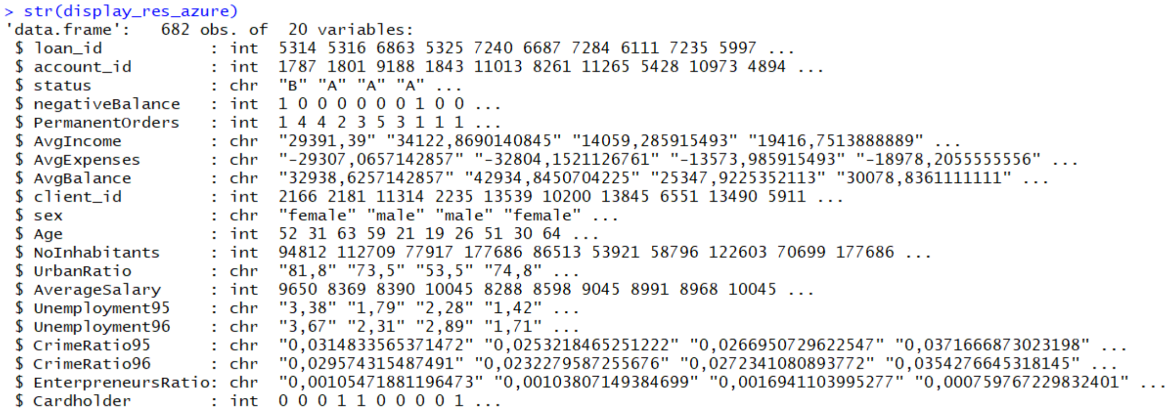
20	card	cardholder	a credit card exists for the account	1 = yes 0 = no

Table 4-1: Attributes of the final dataset for predicting credit scoring (res_azure.csv)

Hence, overall 20 variables describe the predictive model used in the data analysis and mining stage for credit scoring. Table 4-1 above describes the relevant attributes for the results of the pre-processing stage, i.e. those that appear in the selected models explored by R Studio and MS Azure ML.

Spenke and Beilken (1999) stated that “it does not make a difference whether the owner of the account is male, female, young, or old”. They also ascertained that the ‘payments’ field is redundant since the monthly payment can be exactly calculated by the formula amount/duration. Moreover, “there is practically no correlation with the demographic data like unemployment rate, or average salary” (Spenke and Beilken, 1999). These insights were also considered while processing the data.

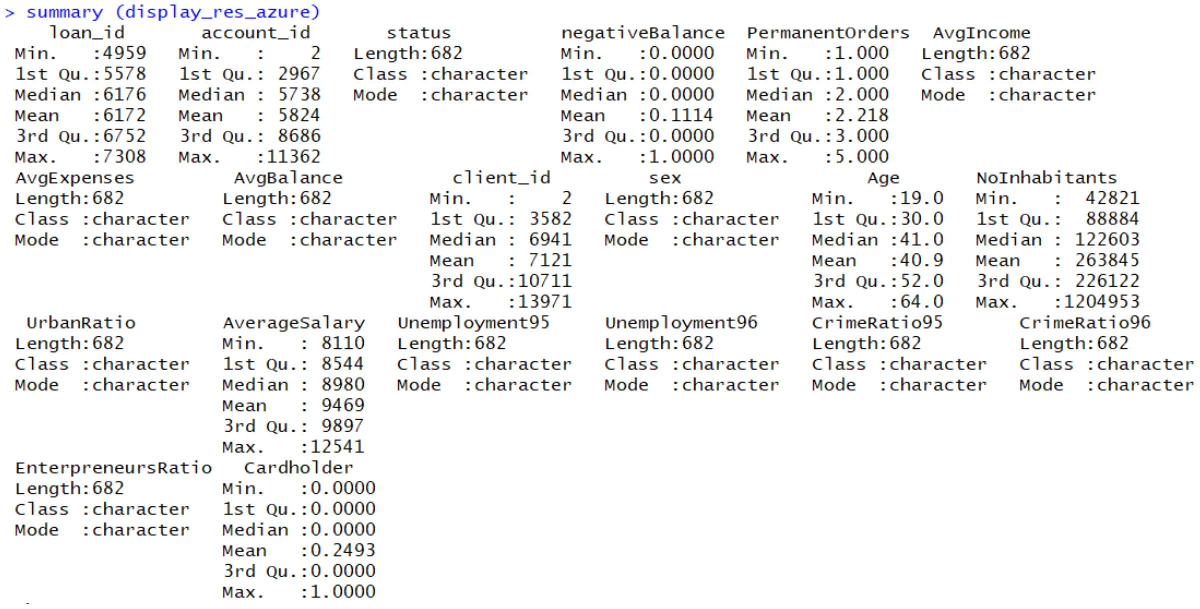
After data cleaning, the final structure of the fetched data is as shown in the figure below.



```
> str(display_res_azure)
'data.frame':	682 obs. of  20 variables:
 $ loan_id           : int  5314 5316 6863 5325 7240 6687 7284 6111 7235 5997 ...
 $ account_id        : int  1787 1801 9188 1843 11013 8261 11265 5428 10973 4894 ...
 $ status            : chr  "B" "A" "A" "A" ...
 $ negativeBalance   : int  1 0 0 0 0 0 0 1 0 0 ...
 $ PermanentOrders   : int  1 4 4 2 3 5 3 1 1 1 ...
 $ AvgIncome         : chr  "29391,39" "34122,8690140845" "14059,285915493" "19416,7513888889" ...
 $ AvgExpenses       : chr  "-29307,0657142857" "-32804,1521126761" "-13573,985915493" "-18978,2055555556" ...
 $ AvgBalance        : chr  "32938,6257142857" "42934,8450704225" "25347,9225352113" "30078,8361111111" ...
 $ client_id         : int  2166 2181 11314 2235 13539 10200 13845 6551 13490 5911 ...
 $ sex               : chr  "female" "male" "male" "female" ...
 $ Age               : int  52 31 63 59 21 19 26 51 30 64 ...
 $ NoInhabitants     : int  94812 112709 77917 177686 86513 53921 58796 122603 70699 177686 ...
 $ UrbanRatio        : chr  "81,8" "73,5" "53,5" "74,8" ...
 $ AverageSalary     : int  9650 8369 8390 10045 8288 8598 9045 8991 8968 10045 ...
 $ Unemployment95    : chr  "3,38" "1,79" "2,28" "1,42" ...
 $ Unemployment96    : chr  "3,67" "2,31" "2,89" "1,71" ...
 $ CrimeRatio95      : chr  "0,0314833565371472" "0,0253218465251222" "0,0266950729622547" "0,0371666873023198" ...
 $ CrimeRatio96      : chr  "0,029574315487491" "0,0232279587255676" "0,0272341080893772" "0,0354276645318145" ...
 $ EnterpreneursRatio: chr  "0,00105471881196473" "0,00103807149384699" "0,0016941103995277" "0,000759767229832401" ...
 $ Cardholder        : int  0 0 0 1 1 0 0 0 0 1 ...
```

Figure 4-39: Structure of data after the final cleaning stage - 1st research project (display_res_azure)

Figure 4-40 below provides an overall summary of all fetched data for conducting the research objectives for the credit scoring case and using the declared dataset as a starting point to realise a cost-sensitive data model.



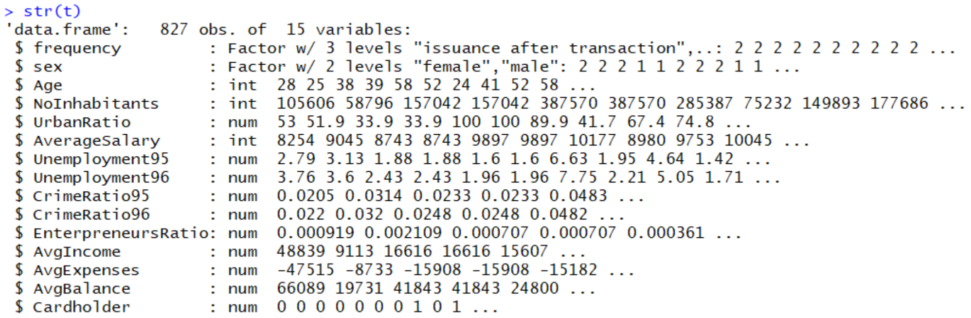
```
> summary (display_res_azure)
    loan_id       account_id        status           negativeBalance  PermanentOrders   AvgIncome
 Min.   :4959   Min.   :    2   Length:682         Min.   :0.0000   Min.   :1.000   Length:682
 1st Qu.:5578   1st Qu.: 2967   Class :character   1st Qu.:0.0000   1st Qu.:1.000   Class :character
 Median :6176   Median : 5738   Mode  :character   Median :0.0000   Median :2.000   Mode  :character
 Mean   :6172   Mean   : 5824                      Mean   :0.1114   Mean   :2.218
 3rd Qu.:6752   3rd Qu.: 8686                      3rd Qu.:0.0000   3rd Qu.:3.000
 Max.   :7308   Max.   :11362                      Max.   :1.0000   Max.   :5.000
 AvgExpenses         AvgBalance            client_id         sex                 Age          NoInhabitants
 Length:682         Length:682         Min.   :    2   Length:682         Min.   :19.0   Min.   :  42821
 Class :character   Class :character   1st Qu.: 3582   Class :character   1st Qu.:30.0   1st Qu.:  88884
 Mode  :character   Mode  :character   Median : 6941   Mode  :character   Median :41.0   Median : 122603
                                       Mean   : 7121                      Mean   :40.9   Mean   : 263845
                                       3rd Qu.:10711                      3rd Qu.:52.0   3rd Qu.: 226122
                                       Max.   :13971                      Max.   :64.0   Max.   :1204953
  UrbanRatio         AverageSalary    Unemployment95     Unemployment96     CrimeRatio95       CrimeRatio96
 Length:682         Min.   : 8110   Length:682         Length:682         Length:682         Length:682
 Class :character   1st Qu.: 8544   Class :character   Class :character   Class :character   Class :character
 Mode  :character   Median : 8980   Mode  :character   Mode  :character   Mode  :character   Mode  :character
                    Mean   : 9469
                    3rd Qu.: 9897
                    Max.   :12541
 EnterpreneursRatio   Cardholder
 Length:682         Min.   :0.0000
 Class :character   1st Qu.:0.0000
 Mode  :character   Median :0.0000
                    Mean   :0.2493
                    3rd Qu.:0.0000
                    Max.   :1.0000
```

Figure 4-40: Summary of fetched data - 1st research project (credit scoring)

4.4.2. Declaration of the dataset exploring the 2nd research project - cross-selling

All introduced pre-processing steps from 1 to 12 of the previous section can be re-used to prepare the dataset for the second research project in terms of assessing the performance of determining cross-selling candidates. The next steps during the data-gathering process was conducted as follows:

(13) We processed the resulting tables “card” and “disp” to join the extracted data with the resulting tables “account”, “client”, “districtRelevantData” and “CashflowsAggregated”. Finally, we created a cross-selling dataset with the final fetched structure as shown in the figure below.



```
> str(t)
'data.frame':   827 obs. of  15 variables:
 $ frequency         : Factor w/ 3 levels "issuance after transaction",..: 2 2 2 2 2 2 2 2 2 2 ...
 $ sex               : Factor w/ 2 levels "female","male": 2 2 2 1 1 2 2 2 1 1 ...
 $ Age               : int  28 25 38 39 58 52 24 41 52 58 ...
 $ NoInhabitants     : int  105606 58796 157042 157042 387570 387570 285387 75232 149893 177686 ...
 $ UrbanRatio        : num  53 51.9 33.9 33.9 100 100 89.9 41.7 67.4 74.8 ...
 $ AverageSalary     : int  8254 9045 8743 8743 9897 9897 10177 8980 9753 10045 ...
 $ Unemployment95    : num  2.79 3.13 1.88 1.88 1.6 1.6 6.63 1.95 4.64 1.42 ...
 $ Unemployment96    : num  3.76 3.6 2.43 2.43 1.96 1.96 7.75 2.21 5.05 1.71 ...
 $ CrimeRatio95      : num  0.0205 0.0314 0.0233 0.0233 0.0483 ...
 $ CrimeRatio96      : num  0.022 0.032 0.0248 0.0248 0.0482 ...
 $ EnterpreneursRatio: num  0.000919 0.002109 0.000707 0.000707 0.000361 ...
 $ AvgIncome         : num  48839 9113 16616 16616 15607 ...
 $ AvgExpenses       : num  -47515 -8733 -15908 -15908 -15182 ...
 $ AvgBalance        : num  66089 19731 41843 41843 24800 ...
 $ Cardholder        : num  0 0 0 0 0 0 0 1 0 1 ...
```

Figure 4-41: Structure of data after data-gathering stage - initialised credit selling table t (final)

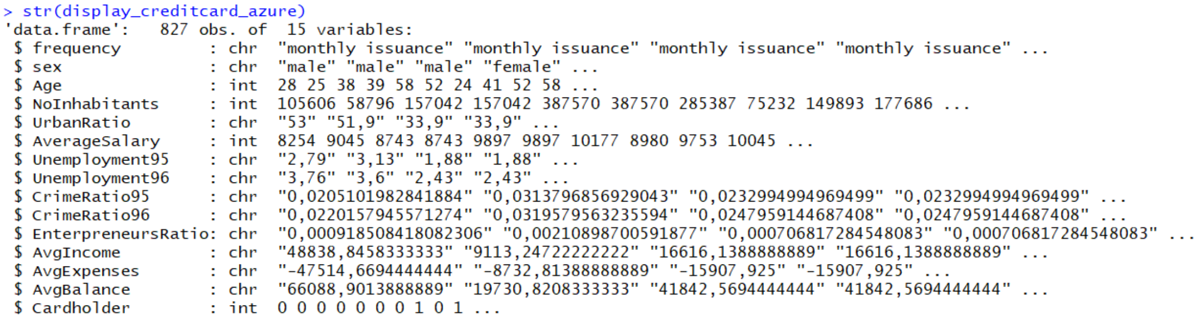
Table 4-2 summarises the final dataset for the second research project, which includes fifteen attributes for 827 accounts. The main efforts were to define new useful attributes whose constellation has not already been investigated by another researcher. The novelty and contribution to research is also a selection of possibly useful attributes to assess the performance of various algorithms in predicting reliable cross-selling candidates for credit card promotions, as well as building a cost-sensitive data model for this purpose.

	relation from original Berka dataset	attribute	explanation	values
1	account	frequency	frequency of issuance of statements	monthly issuance, weekly issuance, issuance after transaction
2	client	sex	sex of the account owner	male, female
3	client	age	year of the account owner	19 - 64
4	district	noInhabitants	number of the inhabitants of account's owner district	42821 - 1204953
5	district	urbanRatio	rate of urbanity of account's owner district	89.9 - 100
6	transaction	averageSalary	average salary of the account owner	8110 - 12541
7	district	unemployment95	rate of unemployment of account's owner district	0.29 - 7.34
8	district	unemployment96	rate of unemployment of account's owner district	0.43 - 9.4
9	district	crimeRatio95	rate of crime of account's owner district	0.0135 - 0.0711
10	district	crimeRatio96	rate of crime of account's owner district	0.0159 - 0.0822
11	district	enterpreneursRatio	rate of entrepreneurs of account's owner district	0.0159 - 0.0822

12	transaction	avgIncome	average of incomes of the account owner	0.000138 - 0.002895
13	transaction	avgExpenses	average of expenses of the account owner	-10130 - -9982
14	transaction	avgBalance	average of balance of the account owner	11702 - 80180
15	card	cardholder	a credit card exists for the account	1 = yes 0 = no

Table 4-2: Attributes of the final dataset for predicting cross-selling candidates (creditcard_azure.csv)

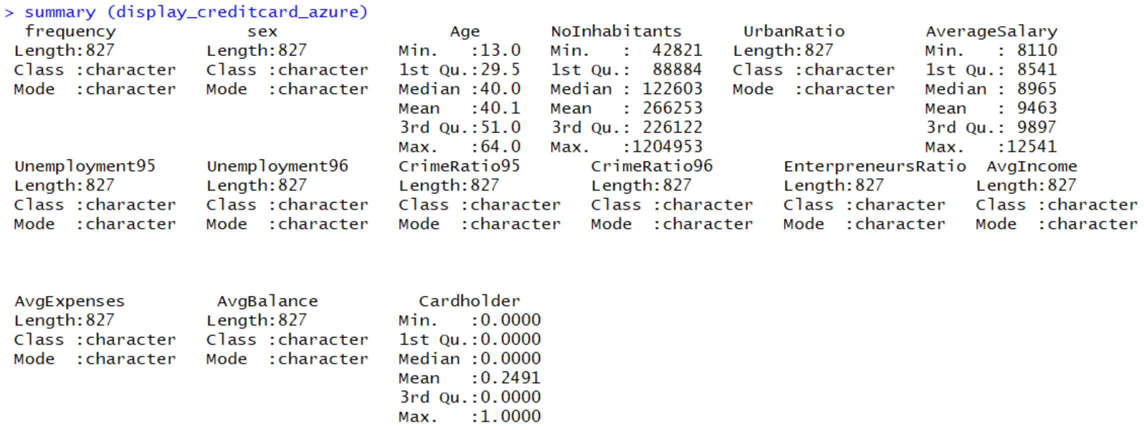
We cleaned the data again by removing non-applicable values within the raw dataset and saving the files in comma-separated format (i.e. creditcard_azure.csv) for further processing in MS Azure ML. After data cleaning, the structure of the fetched data is as shown in figure 4-42 below.



```
> str(display_creditcard_azure)
'data.frame':	827 obs. of  15 variables:
 $ frequency         : chr  "monthly issuance" "monthly issuance" "monthly issuance" "monthly issuance" ...
 $ sex               : chr  "male" "male" "male" "female" ...
 $ Age               : int  28 25 38 39 58 52 24 41 52 58 ...
 $ NoInhabitants     : int  105606 58796 157042 157042 387570 387570 285387 75232 149893 177686 ...
 $ UrbanRatio        : chr  "53" "51,9" "33,9" "33,9" ...
 $ AverageSalary     : int  8254 9045 8743 8743 9897 9897 10177 8980 9753 10045 ...
 $ Unemployment95    : chr  "2,79" "3,13" "1,88" "1,88" ...
 $ Unemployment96    : chr  "3,76" "3,6" "2,43" "2,43" ...
 $ CrimeRatio95      : chr  "0,0205101982841884" "0,0313796856929043" "0,0232994994969499" "0,0232994994969499" ...
 $ CrimeRatio96      : chr  "0,0220157945571274" "0,0319579563235594" "0,0247959144687408" "0,0247959144687408" ...
 $ EnterpreneursRatio: chr  "0,000918508418082306" "0,00210898700591877" "0,000706817284548083" "0,000706817284548083" ...
 $ AvgIncome         : chr  "48838,8458333333" "9113,24722222222" "16616,1388888889" "16616,1388888889" ...
 $ AvgExpenses       : chr  "-47514,6694444444" "-8732,81388888889" "-15907,925" "-15907,925" ...
 $ AvgBalance        : chr  "66088,9013888889" "19730,8208333333" "41842,5694444444" "41842,5694444444" ...
 $ Cardholder        : int  0 0 0 0 0 0 0 1 0 1 ...
```

Figure 4-42: Structure of data after the final cleaning stage - 2nd research project (display_creditcard_azure)

Figure 4-43 below provides an overall summary of all fetched data for conducting the research objectives for the cross-selling candidates case and using the declared dataset as a starting point to realise a cost-sensitive data model.



```
> summary (display_creditcard_azure)
  frequency              sex                 Age         NoInhabitants      UrbanRatio          AverageSalary
 Length:827         Length:827         Min.   :13.0   Min.   :  42821   Length:827         Min.   : 8110
 Class :character   Class :character   1st Qu.:29.5   1st Qu.:  88884   Class :character   1st Qu.: 8541
 Mode  :character   Mode  :character   Median :40.0   Median : 122603   Mode  :character   Median : 8965
                                       Mean   :40.1   Mean   : 266253                      Mean   : 9463
                                       3rd Qu.:51.0   3rd Qu.: 226122                      3rd Qu.: 9897
                                       Max.   :64.0   Max.   :1204953                      Max.   :12541
 Unemployment95     Unemployment96     CrimeRatio95       CrimeRatio96       EnterpreneursRatio  AvgIncome
 Length:827         Length:827         Length:827         Length:827         Length:827         Length:827
 Class :character   Class :character   Class :character   Class :character   Class :character   Class :character
 Mode  :character   Mode  :character   Mode  :character   Mode  :character   Mode  :character   Mode  :character


 AvgExpenses          AvgBalance          Cardholder
 Length:827         Length:827         Min.   :0.0000
 Class :character   Class :character   1st Qu.:0.0000
 Mode  :character   Mode  :character   Median :0.0000
                                       Mean   :0.2491
                                       3rd Qu.:0.0000
                                       Max.   :1.0000
```

Figure 4-43: Summary of fetched data - 2nd research project (cross-selling)

4.4.3. Declaration of the dataset exploring the 3rd research project - categorised transactional payment behaviour

The pre-processing and exploration of the original dataset was implemented in several steps. The first step involved gathering the raw dataset step-by-step and computing new variables, features and labels described in the tables below. According to the data processing approach using the Anaconda framework and Python, in the first step the research generates a relevant subset of the original dataset in the corresponding Python scripts. As a result, the processed data depicted in this section was used in the second step, aiming at data mining and predictive modelling objectives for the third research project using TensorFlow with Python and R Studio. This section discusses the pre-processing stage in which the dataset for the final research project was created by using two different analysis approaches and one unique pre-processing approach. The research dataset was derived from the original transactional table called “trans.ascii” and saved in the comma-separated format “trans.csv”. We only pre-processed the dynamic variables of the table transaction comprises more than one million transactions. In the following, only six attributes (type of transaction, mode of transaction, amount, balance, bank of the partner, transaction category) have been derived for this specific learning task. The corresponding Python scripts for the pre-processing steps are given in appendix E.2.

A complex workflow was developed in Python, from the pre-processing of the raw data to build a relevant classified dataset, ensuring high-quality processing for the large-scale dataset. Before explaining the pre-processing strategy for the research project, the content and information provided in the original transactional table is briefly explained in table 4-3.

	Attribute	Explanation	Values
1	trans_id	record identifier	1 - 3682987
2	account_id	account, the transaction deals with	1 - 11382
3	date	date of transaction	in the form YYMMDD; 930101 - 981231
4	type	type of transaction	credit and withdrawal
5	operation	mode of transaction	5 possible values: credit card withdrawal, credit in cash, collection from another bank, withdrawal in cash, remittance to another bank
6	amount	amount of money	0 - 87400
7	balance	balance after transaction	-41126 - 209637
8	k_symbol	detailed characterisation of the transaction	7 possible values: insurance payments, payment for statement, interest credited, sanction interest, household, old-age pension, loan payment
9	bank	bank of partner	each bank has unique two-letter code
10	account	account of the partner	

Table 4-3: Description of the original transaction table from Berka dataset

(1) We pre-processed the raw data “trans.ascii” to the temporary table ‘data’ by transforming the attributes [3,4,5,6,8] given in the table above into indices and filtering out useable values. It emerged that half of the transactions have no characterisations and would not be useful for our research objectives.

(2) The next pre-processing step was essentially to perform the following mutations for the subsequent attributes:

“dictTypeTranslation” includes the Czech terms: "PRIJEM" assigned to "credit", "VYDAJ" assigned to "withdrawal".

“dictOperationTranslation” includes the Czech terms: "VYBER KARTOU" assigned to “credit card withdrawal”, "VKLAD" assigned to “credit in cash”, "PREVOD Z UCTU" assigned to “collection from another bank”, "VYBER" assigned to “withdrawal in cash”, "PREVOD NA UCET" assigned to “remittance to another bank”.

“dictK_symbolTranslation” includes the Czech terms: “POJISTNE” assigned to “insur. payment”, “SLUZBY” assigned to “payment for statement”, “UROK” assigned to “interest credited”, “SANKC. UROK” assigned to “sanction interest if negative balance”, “SIPO” assigned to “household”, “DUCHOD” assigned to “old-age pension”, “UVER” assigned to “loan payment”.

(3) The next step was processed to create dictionaries to change values of the attributes “type”, “operation” and “partner bank” to floats for further data processing. Therefore, the dictionaries “dictType”, “dictOperation”, “dictPartnerBank” and “dictK_symbol” was transformed to build unique numbers. The reason behind these steps was that feature pre-processing becomes necessary since part of the data is categorical:

“type” includes the following transformed data: 0 = "PRIJEM" stands for credit, 1 = "VYDAJ" stands for withdrawal.

“operation” includes the following transformed data: 0 = "VYBER KARTOU" credit card withdrawal, 1 = "VKLAD" credit in cash, 2 = "PREVOD Z UCTU" collection from another bank, 3 = "VYBER" withdrawal in cash, 4 = "PREVOD NA UCET" remittance to another bank.

“partner bank” includes the following transformed data: Each of fourteen banks has unique two-letter code which will be assigned to the values 0 to 13.

“k_symbol” includes the following transformed data: 0 = "POJISTNE" stands for insurance payment, 1 = "SLUZBY" stands for payment for statement, 2 = "UROK" stands for interest credited, 3 = "SANKC. UROK" sanction interest if negative balance, 4 = "SIPO" stands for household, 5 = "DUCHOD" stands for old-age pension, 6 = "UVER" stands for loan payment.

(4) Next gathering step was performed to create “bins” (or "bucket") for the attribute’s “amount” and “balance”. The Pandas functionality supports current research to divide the entire range of values into a series of ten buckets (by default). Every “Bin” is a categorical object that can be used later on in the predictive model or is useful in constructing a histogram of the processed data. Finally, for the features “amount” and

“balance” we built ten buckets spanning equidistant intervals over the ranges of the features aiming at customer profiling for analysis purpose:

“amount” includes the following transformed data: The function creates 10 bins from 0 to 9.

“balance” includes the following transformed data: The function creates 10 bins from 0 to 9.

(5) In the next step, the pre-processed data was only performed for descriptive analysis. The detailed outcomes of the data visualisation steps are shown in the following section as well as appendix C3.

(6) Finally, we export the cleaned data to a csv file and named each column given in the table below.

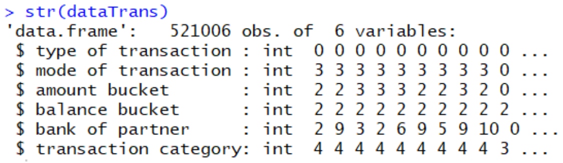
	relation from original Berka dataset	attribute	explanation	values
1	transaction	type of transaction	transactional type	['credit', 'withdrawal']
2	transaction	mode of transaction	transactional mode	['collection from another bank', 'None', 'remittance to another bank', 'withdrawal in cash']
3	transaction	amount	buckets for amount of money	[2, 3, 0, 5, 1, 4, 7, 9, 6, 8]
4	transaction	balance	buckets for balance after transaction	[2, 3, 4, 6, 5, 1, 8, 7, 9, 0]
5	transaction	bank of the partner	partner banks - each bank has unique two-letter code	['YZ', 'UV', 'MN', 'OP', 'AB', 'CD', 'nan', 'GH', 'ST', 'EF', 'WX', 'KL', 'QR', 'IJ']
6	transaction	transaction category	characterisation of the transaction type	['old-age pension', 'interest credited', 'household', 'payment for statement', 'insur. payment', 'sanction

	interest if negative balance', 'loan payment']

Table 4-4: Attributes of the final dataset for analysing transactional behaviour on categorised transactions (dataTrans.csv)

Apart from relying on a single tool like R Studio to pre-process the data for analysis, I have made use of the following set of tools to ease the challenging pre-processing step. Hence, Python, Anaconda, Spyder, Jupyter and TensorFlow are widely used in applications where large-scale data processing is necessary, and thus enables current research faster processing of the more than one million transactions given in the Berka dataset. For instance, Blockeel and Uwents (2004) were unable to include the transaction relation into their training dataset, caused by the size of the financial dataset from Berka.

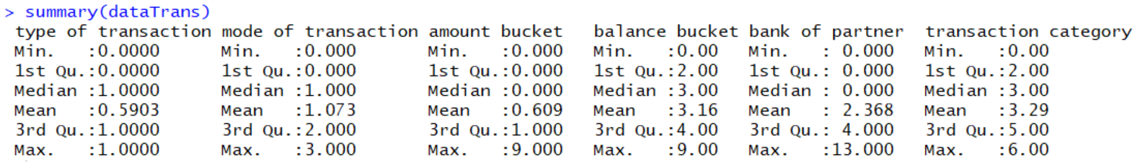
(7) After the data cleaning steps in Python, we also imported the csv file into R Studio for further processing. The structure of the fetched data is as shown in the figure below.



```
> str(dataTrans)
'data.frame':   521006 obs. of  6 variables:
 $ type of transaction : int  0 0 0 0 0 0 0 0 0 0 ...
 $ mode of transaction : int  3 3 3 3 3 3 3 3 3 0 ...
 $ amount bucket       : int  2 2 3 3 3 2 2 3 2 0 ...
 $ balance bucket      : int  2 2 2 2 2 2 2 2 2 2 ...
 $ bank of partner     : int  2 9 3 2 6 9 5 9 10 0 ...
 $ transaction category: int  4 4 4 4 4 4 4 4 4 3 ...
```

Figure 4-44: Structure of data after pre-processing stage - categorised transaction table for the 3rd research project

Figure 4-44 above describes the data structure of the processed data in Python, which was fetched to analyse categorised transactional payment behaviour for bank customers. The final data frame comprises six variables with a total of 521,006 transactions.



```
> summary(dataTrans)
 type of transaction mode of transaction amount bucket   balance bucket bank of partner  transaction category
 Min.   :0.0000      Min.   :0.000       Min.   :0.000   Min.   :0.00   Min.   : 0.000   Min.   :0.00
 1st Qu.:0.0000      1st Qu.:0.000       1st Qu.:0.000   1st Qu.:2.00   1st Qu.: 0.000   1st Qu.:2.00
 Median :1.0000      Median :1.000       Median :0.000   Median :3.00   Median : 0.000   Median :3.00
 Mean   :0.5903      Mean   :1.073       Mean   :0.609   Mean   :3.16   Mean   : 2.368   Mean   :3.29
 3rd Qu.:1.0000      3rd Qu.:2.000       3rd Qu.:1.000   3rd Qu.:4.00   3rd Qu.: 4.000   3rd Qu.:5.00
 Max.   :1.0000      Max.   :3.000       Max.   :9.000   Max.   :9.00   Max.   :13.000   Max.   :6.00
```

Figure 4-45: Summary of fetched data - 3rd research project (categorised transactions)

4.5. Data analysis and data visualisation

In this section, we will briefly explain the tools and features used for visualising the analysis results as well as how we formed an important dataset for performing the data analysis. After pre-processing the data as presented in the previous sections, the selected tools could be used since all relevant information from different tables and groups are extracted.

All experimental results presented in this research are computed on a personal computer with Microsoft operating system, and the predictive models are executed in the MS Azure ML cloud. This research used various supervised learning and unsupervised learning algorithms, which resulted in different comparable evaluation results. The attributes introduced in the previous section are mostly categorial and numerical ones. Therefore, most natural data analysis applies classification algorithms, which can easily handle numeric values. In case numeric values are not given, we transformed the attribute into an appropriate format during the pre-processing stage. Hence, in order to visualise the data and its analysis results for the first and second research project more effectively without writing complicated R or Python codes, we can use the data visualisation capabilities of MS Azure ML to focus more on the time-consuming pre-processing part during the parsing and gathering phases.

Data visualisation is one of the most successful techniques to interpret evaluation results as “visualization helps to find several interesting results” (Miksovsky, Zelezny, Stepankova, Pechoucek, 1999). The visualisation is realised by using MS Azure ML and different visualisation packages from R Studio, especially for the third research project in which (un-)categorised transactions were analysed. Accordingly, we also visualised the pre-processed data with Python visualisation packages. Regarding the first two research projects, MS Azure ML enables current research to interactively explore different visualisations of the data as it introduces a unique visualisation of the entire dataset on single scalable response charts. One of the main advantages of using such powerful tools is that the “user gets a feeling of the data, detects interesting knowledge, and gains a deep understanding of the dataset” (Spenke and Beilken, 1999). It allows me to interactively adjust the threshold as well as the accuracy under curve (AUC) of the algorithms applied very easily, and the performance characteristics were simply to investigate. Moreover, MS Azure ML can quickly perform selected algorithms and replace them with others, as well as immediately displaying their results.

The research will illustrate how the data analysis results can be interpreted using various mining algorithms, implementing and visualising them using R Studio, Python and MS Azure ML. In order to discover interesting characteristics within the dataset, the underlying research approach used a descriptive profile analysis technique, namely univariate deviation methods for outlier detection (Aggarwal, 2013). For continuous attributes such as average income and expenses or balance at the end of the month, descriptive results can be illustrated easily in MS Azure ML to measure interesting attributes and compare the results with all other clients. In addition, in-depth insights can also be distinguishing by comparing affiliated attributes with each other. Finally, the descriptive analysis describes whether the findings are statistically significant and visualises the most interesting characteristics. Detailed visualisation results are given in appendices C and D.

The following sub-sections describe the procedure applied to build an important dataset for every examined research project.

4.5.1 Building a relevant dataset for a cost-sensitive data model - credit scoring

Regarding the entire research structure introduced at the beginning of the current chapter, the research approach can optimise the existing proceeding for the data modelling process. Accordingly, we identify the most relevant attributes within the modelling process based on the pre-processed data in the previous sections for the first research project. The research objective was to ultimately build a cost-sensitive data model for the best-performing classification algorithm or regression algorithm by comparing the applied algorithms results on the original as well as the optimised dataset.

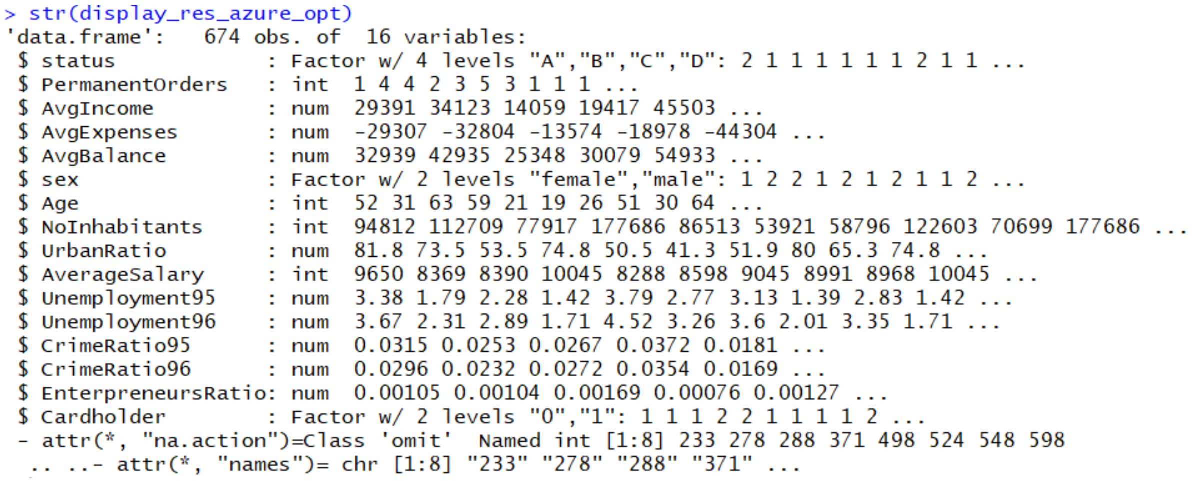
In the first step, the final pre-processed dataset labelled as ”display_res_azure.csv” from the previous section is completely read and prepared for the further pre-processing steps. Thus, we excluded non-relevant attributes such as loan_id, account_id, client_id and negativeBalance from the original dataset, and transformed the attributes cardholder, sex and status into factor variables. Finally, we removed all missing values in the remaining objects. The corresponding R-script is given in appendix E.1.

Table 4-5 below summarises the final optimised dataset for the first research project, which ultimately includes only sixteen attributes (instead of 20 attributes) for 674 accounts. The novelty and contribution to research is to efficiently build a cost-sensitive data model to assess the performance of the applied algorithm in predicting a reliable credit score on an optimised dataset.

	attribute	explanation	values
1	status	good = {a, c}, bad = {b, d}	A, B, C, D
2	permanentOrders	number of permanent orders exists for the account	1 - 5
3	avgIncome	average of incomes of the account owner	5191.16 - 75197.18
4	avgExpenses	average of expenses of the account owner	-4938.75 -72330.92
5	avgBalance	average of balance of the account owner	7716.37 - 63647.53
6	sex	sex of the account owner	male, female
7	age	year of the account owner	19 - 64
8	noInhabitants	number of the inhabitants of account's owner district	42821 - 1204953
9	urbanRatio	rate of urbanity of account's owner district	33.9 - 100.0
10	averageSalary	average salary of the account owner	8110 - 12541
11	unemployment95	rate of unemployment of account's owner district	0.29 - 7.34
12	unemployment96	rate of unemployment of account's owner district	0.43 - 9.4
13	crimeRatio95	rate of crime of account's owner district	0.041 - 0.071
14	crimeRatio96	rate of crime of account's owner district	0.039 - 0.082
15	enterpreneursRatio	rate of entrepreneurs of account's owner district	0.00013 - 0.00289
16	cardholder	a credit card exists for the account	1 = yes or 0 = no

Table 4-5: Attributes of the final optimised dataset for predicting credit scoring (res_azure_opt.csv)

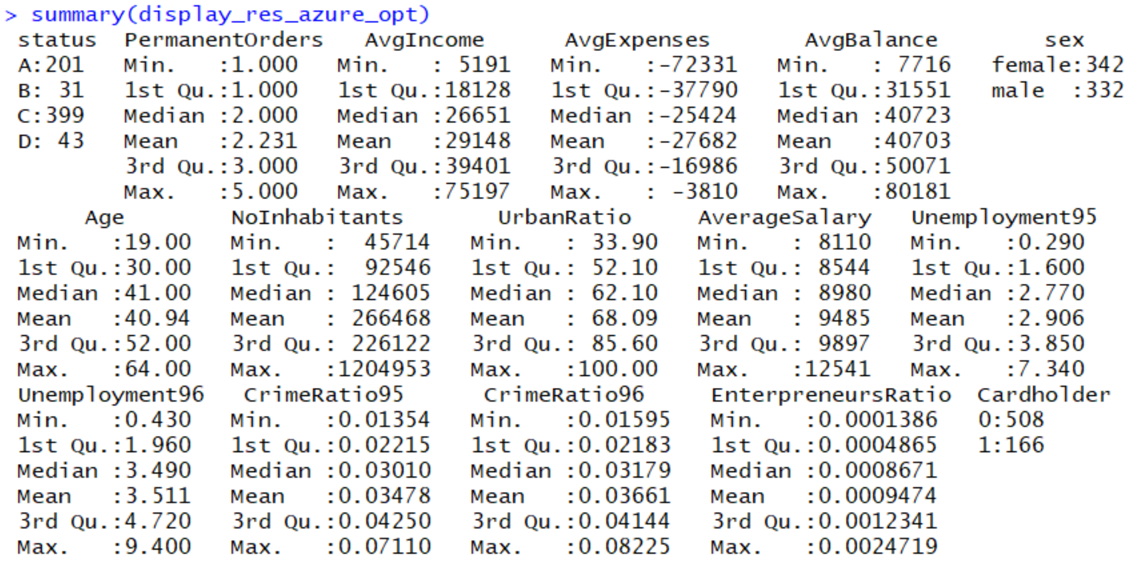
After data cleaning, the final structure of the fetched data is as shown in figure 4-46 below.



```
> str(display_res_azure_opt)
'data.frame':	674 obs. of  16 variables:
 $ status            : Factor w/ 4 levels "A","B","C","D": 2 1 1 1 1 1 1 2 1 1 ...
 $ PermanentOrders   : int  1 4 4 2 3 5 3 1 1 1 ...
 $ AvgIncome         : num  29391 34123 14059 19417 45503 ...
 $ AvgExpenses       : num  -29307 -32804 -13574 -18978 -44304 ...
 $ AvgBalance        : num  32939 42935 25348 30079 54933 ...
 $ sex               : Factor w/ 2 levels "female","male": 1 2 2 1 2 1 2 1 1 2 ...
 $ Age               : int  52 31 63 59 21 19 26 51 30 64 ...
 $ NoInhabitants     : int  94812 112709 77917 177686 86513 53921 58796 122603 70699 177686 ...
 $ UrbanRatio        : num  81.8 73.5 53.5 74.8 50.5 41.3 51.9 80 65.3 74.8 ...
 $ AverageSalary     : int  9650 8369 8390 10045 8288 8598 9045 8991 8968 10045 ...
 $ Unemployment95    : num  3.38 1.79 2.28 1.42 3.79 2.77 3.13 1.39 2.83 1.42 ...
 $ Unemployment96    : num  3.67 2.31 2.89 1.71 4.52 3.26 3.6 2.01 3.35 1.71 ...
 $ CrimeRatio95      : num  0.0315 0.0253 0.0267 0.0372 0.0181 ...
 $ CrimeRatio96      : num  0.0296 0.0232 0.0272 0.0354 0.0169 ...
 $ EnterpreneursRatio: num  0.00105 0.00104 0.00169 0.00076 0.00127 ...
 $ Cardholder        : Factor w/ 2 levels "0","1": 1 1 1 2 2 1 1 1 1 2 ...
 - attr(*, "na.action")=Class 'omit'  Named int [1:8] 233 278 288 371 498 524 548 598
  .. ..- attr(*, "names")= chr [1:8] "233" "278" "288" "371" ...
```

Figure 4-46: Structure of data after the final cleaning stage - 1st research project (display_res_azure_opt)

Figure 4-47 below provides an overall summary of all fetched data for conducting the research objectives for the first research project in building cost-sensitive data models.



```
> summary(display_res_azure_opt)
 status  PermanentOrders   AvgIncome      AvgExpenses       AvgBalance         sex     
 A:201   Min.   :1.000   Min.   : 5191   Min.   :-72331   Min.   : 7716   female:342  
 B: 31   1st Qu.:1.000   1st Qu.:18128   1st Qu.:-37790   1st Qu.:31551   male  :332  
 C:399   Median :2.000   Median :26651   Median :-25424   Median :40723               
 D: 43   Mean   :2.231   Mean   :29148   Mean   :-27682   Mean   :40703               
         3rd Qu.:3.000   3rd Qu.:39401   3rd Qu.:-16986   3rd Qu.:50071               
         Max.   :5.000   Max.   :75197   Max.   : -3810   Max.   :80181               
      Age        NoInhabitants       UrbanRatio      AverageSalary   Unemployment95 
 Min.   :19.00   Min.   :  45714   Min.   : 33.90   Min.   : 8110   Min.   :0.290  
 1st Qu.:30.00   1st Qu.:  92546   1st Qu.: 52.10   1st Qu.: 8544   1st Qu.:1.600  
 Median :41.00   Median : 124605   Median : 62.10   Median : 8980   Median :2.770  
 Mean   :40.94   Mean   : 266468   Mean   : 68.09   Mean   : 9485   Mean   :2.906  
 3rd Qu.:52.00   3rd Qu.: 226122   3rd Qu.: 85.60   3rd Qu.: 9897   3rd Qu.:3.850  
 Max.   :64.00   Max.   :1204953   Max.   :100.00   Max.   :12541   Max.   :7.340  
 Unemployment96   CrimeRatio95      CrimeRatio96     EnterpreneursRatio  Cardholder
 Min.   :0.430   Min.   :0.01354   Min.   :0.01595   Min.   :0.0001386   0:508     
 1st Qu.:1.960   1st Qu.:0.02215   1st Qu.:0.02183   1st Qu.:0.0004865   1:166     
 Median :3.490   Median :0.03010   Median :0.03179   Median :0.0008671             
 Mean   :3.511   Mean   :0.03478   Mean   :0.03661   Mean   :0.0009474             
 3rd Qu.:4.720   3rd Qu.:0.04250   3rd Qu.:0.04144   3rd Qu.:0.0012341             
 Max.   :9.400   Max.   :0.07110   Max.   :0.08225   Max.   :0.0024719             
```

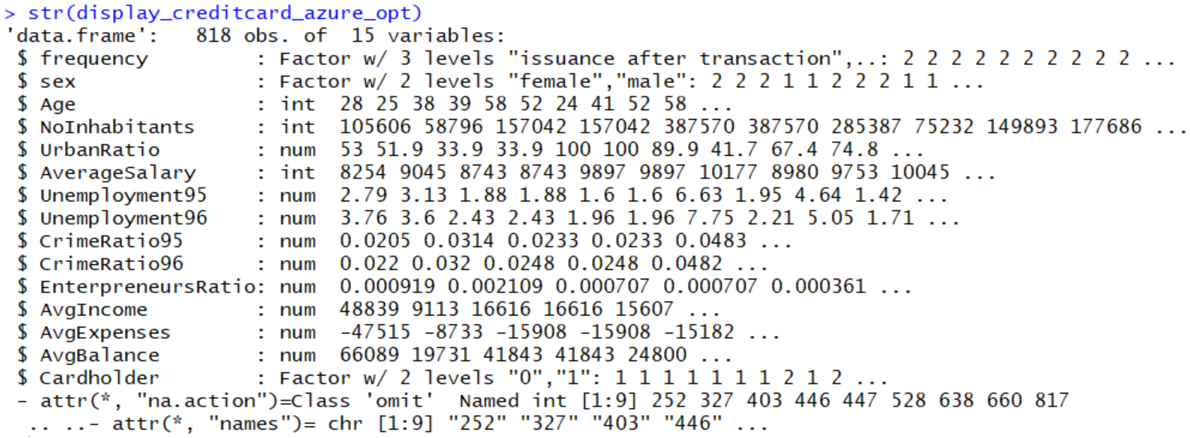
Figure 4-47: Summary of fetched optimised dataset - 1st research projcet (credit scoring)

4.5.2 Building a relevant dataset for a cost-sensitive data model - cross-selling

A further research objective for the second research project was to build a cost-sensitive data model for the best-performing classification algorithm by comparing the algorithms results applied on the original as well as the optimised dataset with respect to the cross-selling case.

In the first step, the final pre-processed dataset named ”display_creditcard_azure.csv” from the previous section is completely read and prepared for the further pre-processing steps. The corresponding R-script is given in appendix E.1. Table 4-2 from the previous section summarises the final optimised dataset for the second research project, which ultimately includes fifteen attributes for 818 accounts. The novelty and contribution to research is also to efficiently build a cost-sensitive data model to assess the performance of the applied supervised learning algorithm in predicting cross-selling candidates on an optimised dataset.

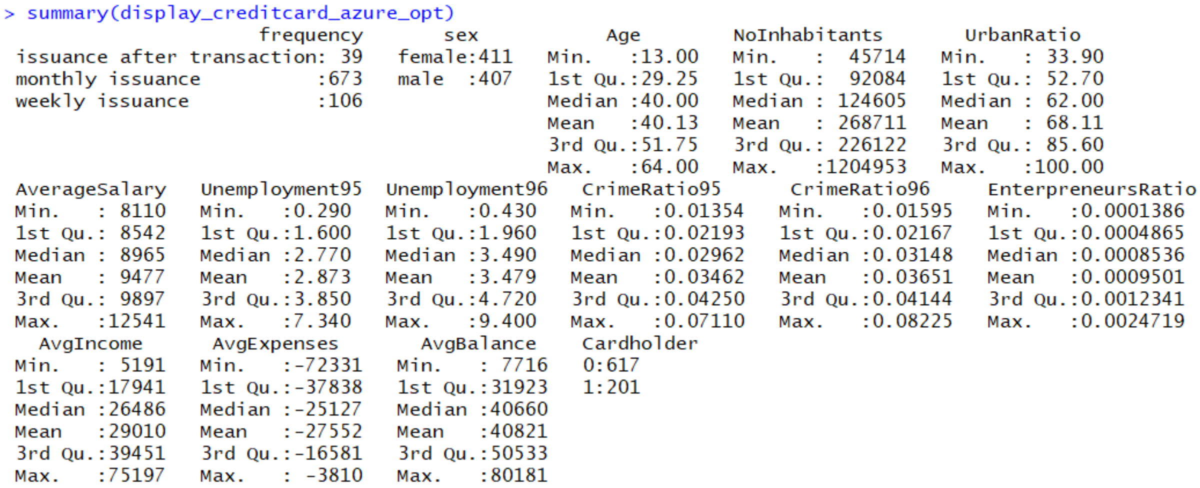
After several data cleaning steps (i.e. converting variables into factors and removing missing values in an object), the final structure of the fetched data is as shown in figure 4-48 below.



```
> str(display_creditcard_azure_opt)
'data.frame':   818 obs. of  15 variables:
 $ frequency         : Factor w/ 3 levels "issuance after transaction",..: 2 2 2 2 2 2 2 2 2 2 ...
 $ sex               : Factor w/ 2 levels "female","male": 2 2 2 1 1 2 2 2 1 1 ...
 $ Age               : int  28 25 38 39 58 52 24 41 52 58 ...
 $ NoInhabitants     : int  105606 58796 157042 157042 387570 387570 285387 75232 149893 177686 ...
 $ UrbanRatio        : num  53 51.9 33.9 33.9 100 100 89.9 41.7 67.4 74.8 ...
 $ AverageSalary     : int  8254 9045 8743 8743 9897 9897 10177 8980 9753 10045 ...
 $ Unemployment95    : num  2.79 3.13 1.88 1.88 1.6 1.6 6.63 1.95 4.64 1.42 ...
 $ Unemployment96    : num  3.76 3.6 2.43 2.43 1.96 1.96 7.75 2.21 5.05 1.71 ...
 $ CrimeRatio95      : num  0.0205 0.0314 0.0233 0.0233 0.0483 ...
 $ CrimeRatio96      : num  0.022 0.032 0.0248 0.0248 0.0482 ...
 $ EnterpreneursRatio: num  0.000919 0.002109 0.000707 0.000707 0.000361 ...
 $ AvgIncome         : num  48839 9113 16616 16616 15607 ...
 $ AvgExpenses       : num  -47515 -8733 -15908 -15908 -15182 ...
 $ AvgBalance        : num  66089 19731 41843 41843 24800 ...
 $ Cardholder        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 2 1 2 ...
 - attr(*, "na.action")=Class 'omit'  Named int [1:9] 252 327 403 446 447 528 638 660 817
  .. ..- attr(*, "names")= chr [1:9] "252" "327" "403" "446" ...
```

Figure 4-48: Structure of data (optimised) after the final cleaning stage - 2nd research project (display_creditcard_azure_opt)

Figure 4-49 below provides an overall summary of all fetched data for conducting the research objectives for the second research project in building a cost-sensitive data model.



```
> summary(display_creditcard_azure_opt)
                      frequency      sex            Age         NoInhabitants      UrbanRatio
 issuance after transaction: 39   female:411   Min.   :13.00   Min.   :  45714   Min.   : 33.90
 monthly issuance          :673   male  :407   1st Qu.:29.25   1st Qu.:  92084   1st Qu.: 52.70
 weekly issuance           :106                Median :40.00   Median : 124605   Median : 62.00
                                               Mean   :40.13   Mean   : 268711   Mean   : 68.11
                                               3rd Qu.:51.75   3rd Qu.: 226122   3rd Qu.: 85.60
                                               Max.   :64.00   Max.   :1204953   Max.   :100.00
 AverageSalary   Unemployment95  Unemployment96   CrimeRatio95       CrimeRatio96      EnterpreneursRatio
 Min.   : 8110   Min.   :0.290   Min.   :0.430   Min.   :0.01354   Min.   :0.01595   Min.   :0.0001386
 1st Qu.: 8542   1st Qu.:1.600   1st Qu.:1.960   1st Qu.:0.02193   1st Qu.:0.02167   1st Qu.:0.0004865
 Median : 8965   Median :2.770   Median :3.490   Median :0.02962   Median :0.03148   Median :0.0008536
 Mean   : 9477   Mean   :2.873   Mean   :3.479   Mean   :0.03462   Mean   :0.03651   Mean   :0.0009501
 3rd Qu.: 9897   3rd Qu.:3.850   3rd Qu.:4.720   3rd Qu.:0.04250   3rd Qu.:0.04144   3rd Qu.:0.0012341
 Max.   :12541   Max.   :7.340   Max.   :9.400   Max.   :0.07110   Max.   :0.08225   Max.   :0.0024719
   AvgIncome      AvgExpenses       AvgBalance     Cardholder
 Min.   : 5191   Min.   :-72331   Min.   : 7716   0:617
 1st Qu.:17941   1st Qu.:-37838   1st Qu.:31923   1:201
 Median :26486   Median :-25127   Median :40660
 Mean   :29010   Mean   :-27552   Mean   :40821
 3rd Qu.:39451   3rd Qu.:-16581   3rd Qu.:50533
 Max.   :75197   Max.   : -3810   Max.   :80181
```

Figure 4-49: Summary of fetched optimised dataset - 2nd research project (cross-selling)

4.5.3 Building a relevant dataset for a predictive data model - categorised transactions

The most interesting attributes from the original Berka dataset were selected based on the transaction table to build and train the data models for the last research project (3). Figures 4-50 and 4-51 below describe the relevant attributes of the final pre-processed features and labels that we include in the modelling process. The corresponding Python script is given in appendix E.2.

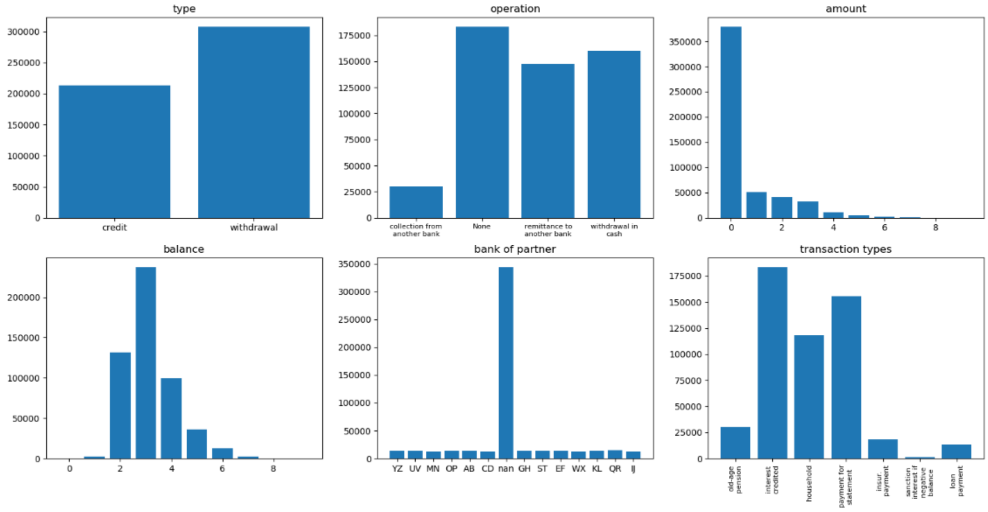


Figure 4-50: Bar charts (in Python) of all pre-processed features and labels (dataTrans.csv)

The boxplots in figure 4-51 below reveal the distribution of all values given in the processed features. Regarding our data analysis approach, we can ascertain in which area the features and labels are largely based. This is a major benefit in the research design and development and less of an error source when interpreting the research results. The corresponding R-script for the data visualisation is given in appendix E.1. The multi-boxplot below indicates that there are some outliers within the variables bucket of amount, bucket of balance and bank of partner. Regarding the range of the values given in the variable’s mode of transaction, bucket of balance and transaction category, most of them are uniformly distributed.

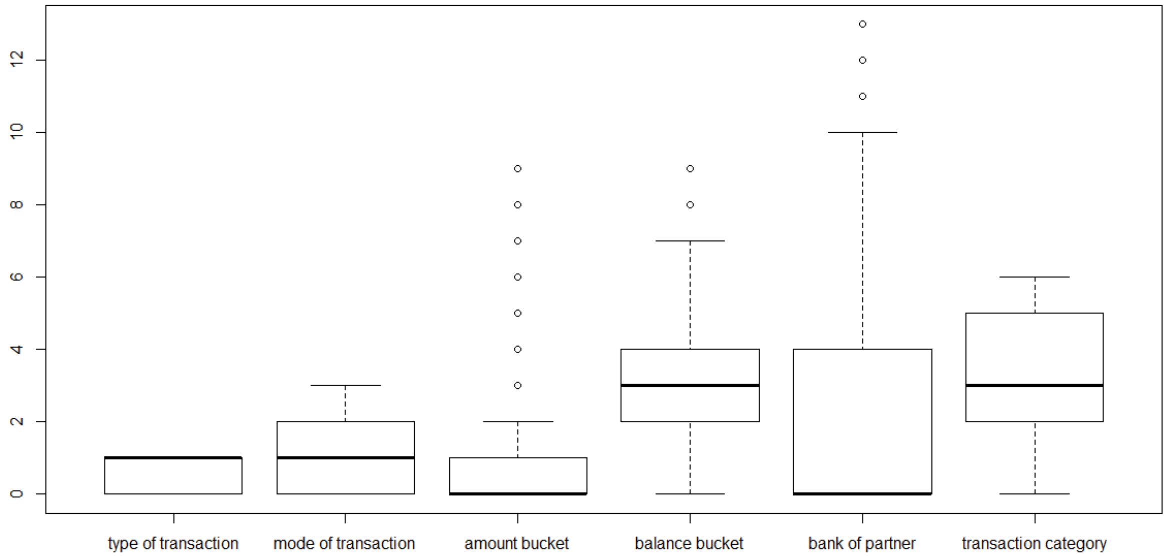


Figure 4-51: Multi-boxplot (in R Studio) of all pre-processed features and labels (dataTrans.csv)

Beyond that, the research approach uses correlations to understand more about the pre-processed data, especially in the feature selection before any kind of statistical modelling is employed. Figure 4-52 below shows associations between the variables of the data table “dataTrans”, whereby the observable pattern is that only a subset of variables strongly correlates with each other. A correlation analysis is a vital tool for feature selection and multivariate analysis during the data pre-processing and data exploration phases. Below correlations such as transaction category vs. transaction type (no relationship: -0.071) mode of transaction (large positive relationship: 0.737) or bucket of amount (large negative relationship: -0.708) or bucket of balance (no relationship: 0.075) or bank of partner (large negative relationship: -0.755) indicate established relationships between the selected variables.

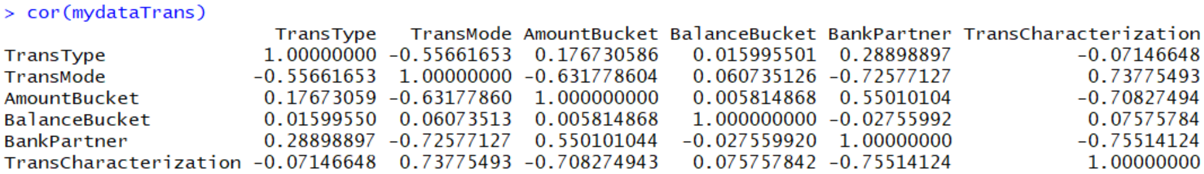

```
> cor(mydataTrans)
                          TransType   TransMode AmountBucket BalanceBucket BankPartner TransCharacterization
TransType                1.00000000 -0.55661653  0.176730586   0.015995501  0.28898897           -0.07146648
TransMode               -0.55661653  1.00000000 -0.631778604   0.060735126 -0.72577127            0.73775493
AmountBucket             0.17673059 -0.63177860  1.000000000   0.005814868  0.55010104           -0.70827494
BalanceBucket            0.01599550  0.06073513  0.005814868   1.000000000 -0.02755992            0.07575784
BankPartner              0.28898897 -0.72577127  0.550101044  -0.027559920  1.00000000           -0.75514124
TransCharacterization   -0.07146648  0.73775493 -0.708274943   0.075757842 -0.75514124            1.00000000
```

Figure 4-52: Correlations between multiple variables of the data frame - mydataTrans

Regarding the below correlation matrix plot, the distribution of each variable shown on the diagonal highlights that the variables bucket of amount, bucket of balance and bank of partner are distributed to the left. On the bottom of the diagonal, the bivariate scatter plots for every variable pairs with a fitted line are displayed. The value of the correlation plus the significance level as stars are displayed on the top of the diagonal. Every pair of variables shows its significance level, which is associated with a corresponding star depending on their p-value results (0 = “***”, 0.001 = “**”, 0.01 = “*”, 0.05 = “.”, 0.1 = " “, 1 =). For instance, if the p-value is greater than the significance level of 0.05, there is inconclusive evidence about the statistical significance of the association between the variable’s transaction category vs. mode of transaction, transaction category vs. bucket of amount and transaction category vs. bank of partner.

Further visualisations of the correlation matrix as well as detailed results of the correlation coefficients and computed p-values summarised in a flatten correlation matrix are given in appendix C.3.

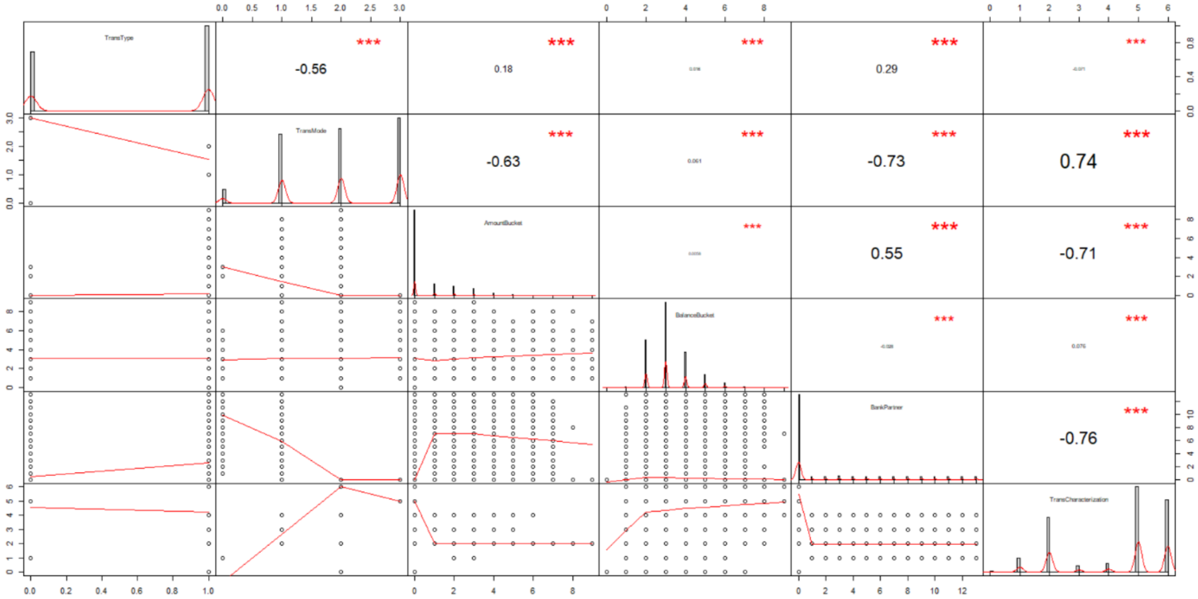


Figure 4-53: Correlation matrix (performance analytics) of the data table - mydataTrans

Finally, appendix C also provides an exhaustive exploratory data analysis of the processed categorised transactions implemented in R Studio.

4.6. Theoretical concepts of the applied supervised and unsupervised learning algorithm

This section reviews basis concepts of the chosen supervised and unsupervised learning algorithm. The review is given in a sufficient way, whereby a reader interested in more mathematical details is referred to more specific teaching books and documents. Any changing customer behavioural information obtained from the data exploration and analysis is investigated using various descriptive and predictive models, including the multiclass neural network, two-class neural network, two-class logistic regression, two-class decision forest and two-class support vector machine models. The performance of these models is investigated to determine their predictive efficiency based on the recommendations given in Hyndman (2014). Note that a perfect model fit requires a model with sufficient parameters and good prediction not only depends on well-fitted models. For instance, every “prediction has to be based on information the bank knows in time of asking for the loan” (Coufal, Holeňa and Sochorová, 1999). Therefore, over-fitting models must be avoided in the different areas of the research projects.

The prediction models for the three research projects are based only on supervised learning algorithms. Every research project uses a separate training and test set, comprising 20% and 80% of all clients, respectively. The only free parameters (scalable threshold and area under curve) of the applied machine learning algorithms were set to 0.5 for the threshold as default by MS Azure ML.

The subsequent sections provide a detailed overview of all selected mining methods across the various research projects.

4.6.1. Mining methods for credit scoring according to the 1st research project

The data modelling process is an initial part of the research design and crucial for the data analysis and its expected results. The majority of machine learning algorithms use supervised learning techniques to build an essential data model. Regarding the preceding data pre-processing and data exploration section, the declared credit scoring dataset can be used to respond to the research questions by applying selected supervised learning algorithms from MS Azure ML as described in figure 4-54 below. Therefore, the declared dataset will be divided into a set of test data and training data. The goal is to use a best-fitting machine learning algorithm to learn mapping and approximate the function f for predicting the output y in an iterative approach. The learning of the various algorithms below may be terminated when the machine learning algorithm reaches an acceptable performance level. The applied mining methods can be roughly broken down into two categories of supervised learning algorithms. First, the classification category, in which the output variable is a category or class:

The multiclass neural network creates a multiclass classification model using a neural network algorithm.

The two-class neural network creates a binary classifier using a neural network algorithm.

The two-class decision forest creates a two-class classification model using the decision forest algorithm.

The two-class support vector machine creates a binary classification model using the support vector machine algorithm.

The second applied supervised learning category is the regression in which the output variable is a real value:

The two-class logistic regression creates a two-class logistic regression model.

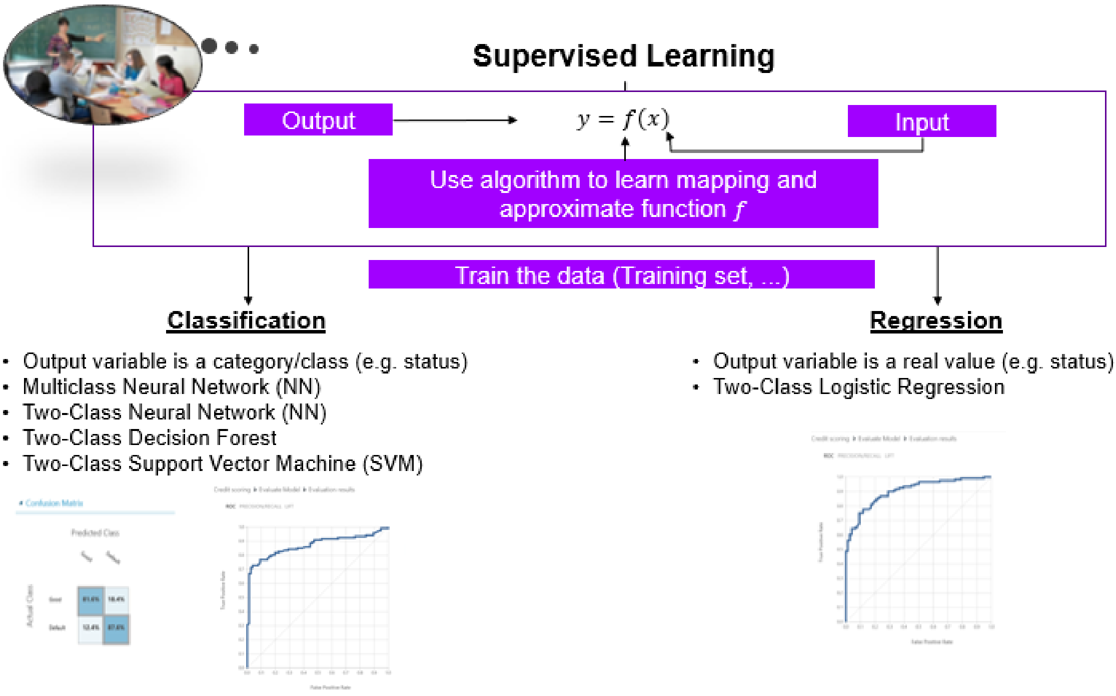

Supervised Learning
Output
$y = f(x)$
Input
Use algorithm to learn mapping and approximate function f
Train the data (Training set, ...)
Classification
• Output variable is a category/class (e.g. status)
• Multiclass Neural Network (NN)
• Two-Class Neural Network (NN)
• Two-Class Decision Forest
• Two-Class Support Vector Machine (SVM)
Regression
• Output variable is a real value (e.g. status)
• Two-Class Logistic Regression

Figure 4-54: Data mining methods for the 1st research project

Finally, regarding the pre-processing steps for building a cost-sensitive data model for the credit scoring case, we used the “polycor” package from R Studio to compute the statistical correlations of the selected variables within the optimised dataset and test their predictive modelling results accordingly by using introduced MS Azure ML algorithms. In addition, the R packages “caret” and “randomForest” are applied to display the descriptive and predictive modelling results based on the optimised dataset. The corresponding R-script is given in appendix E.1.

4.6.2. Mining methods for cross-selling according to the 2nd research project

Regarding the preceding data pre-processing and data exploration section, the declared cross-selling candidate dataset can be used to respond to the research questions by applying the selected supervised learning algorithms as described in figure 4-55 below. While doing so, we introduce only the delta of mining algorithms in detail that have not been presented in the previous section.

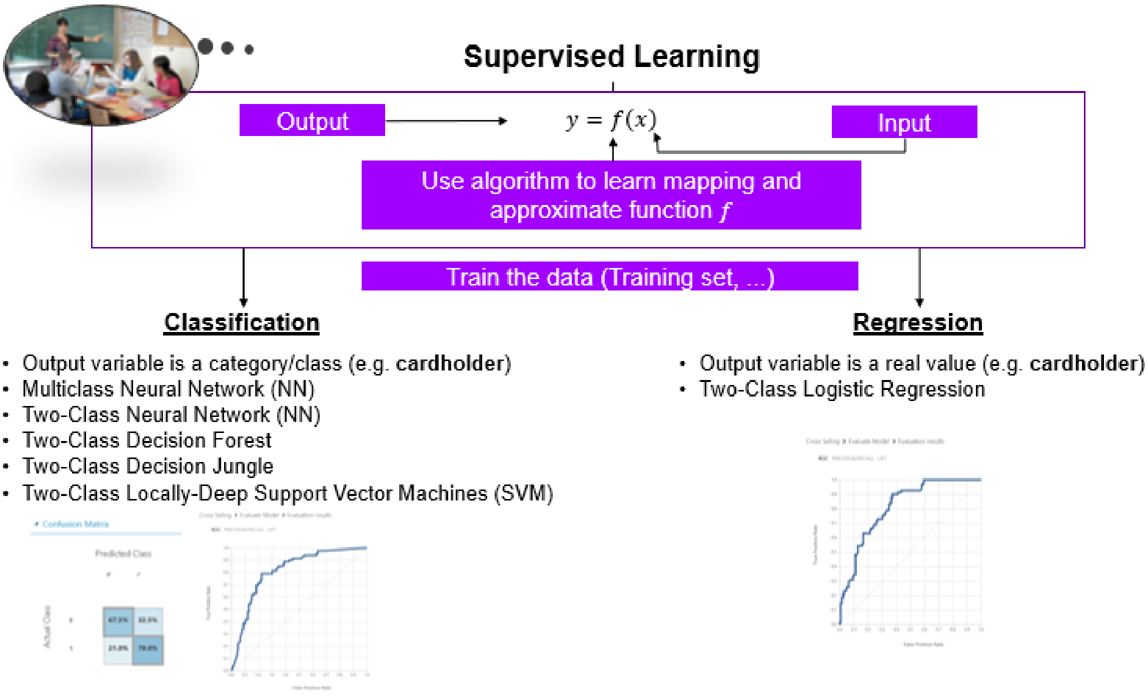

Supervised Learning
Output
$y = f(x)$
Input
Use algorithm to learn mapping and approximate function f
Train the data (Training set, ...)
Classification
• Output variable is a category/class (e.g. cardholder)
• Multiclass Neural Network (NN)
• Two-Class Neural Network (NN)
• Two-Class Decision Forest
• Two-Class Decision Jungle
• Two-Class Locally-Deep Support Vector Machines (SVM)
Regression
• Output variable is a real value (e.g. cardholder)
• Two-Class Logistic Regression
Confusion Matrix
Predicted Class
Actual Class

Figure 4-55: Data mining methods for the 2nd research project

The data modelling results for the second research project are also based on the following classification algorithms given in the MS Azure ML library:

The two-class decision jungle creates a two-class classification model using the decision jungle algorithm.

The two-class locally-deep support vector machine creates a binary classification model using the locally-deep support vector machine algorithm.

With a view to building a cost-sensitive data model for the cross-selling case, I have also used the “polycor” package from R Studio to compute the statistical correlations of the selected variables within the optimised dataset and test their predictive results using MS Azure ML tool. Thereby, the R packages “caret” and “randomForest” are again applied to display the descriptive and predictive modelling results based on the optimised dataset. Finally, the corresponding R-script is given in appendix E.1.

4.6.3. Mining methods for categorised transactional payment behaviour according to the 3rd research project

The section describes the various mining methods applied in the third research project. Therefore, the chart below sets out the theoretical concepts of supervised as well as unsupervised learning algorithms that I have applied to analyse the payment behaviour based on the given categorised transactional data. The key differentiator between the two learning algorithms is that unsupervised learning algorithms do not need training data and there is no right or wrong according to their modelling output results. In general, modelling the underlying transactional data structure and the distribution of the data to learn more about (inter-) related, associated and mutually-supportive transactions hold primary importance in analysing customer payment behaviour.

The applied mining methods can also be roughly broken down into two categories of unsupervised learning algorithms. On the one hand, there are a number of association algorithms (e.g. association rules, frequency and sequence analysis) that we applied in the last research project, while on the other hand there are various clustering algorithms that support classifying inherent groups of data and customer groups with the same purchasing behaviour. The latter category of unsupervised learning algorithm clustering has been ruled out due to the lack of an appropriated transactional dataset. I have focused on applying association rules using a market basket analysis on categorised transactions to discover interesting rules. Compared with other applied supervised learning algorithms such as the multiclass neural network algorithm, the researcher does not need to define what characteristics the machine should be looking for. The last research project also uses a classification algorithm to predict the next transactional category for their banking clients. However, the data diagnostic check by computing a correlation matrix in the previous section also shows that a linear regression algorithm is suitable and will probably generate reliable outputs due to the low number of correlations in the declared dataset for the last research project.

Regarding the data pre-processing and data exploration section, the declared categorised transaction dataset can be used to respond to the research questions by applying the selected supervised and unsupervised learning algorithms as summarised in figure 4-56 below. Therefore, I have considered two different analysis approaches within the presented research design to ensure and increase the objectivity of the research outcomes. The research project reported in this section can be divided into two distinct approaches in which the transactional behaviour was studied with different formal algorithms presented in the following chart.

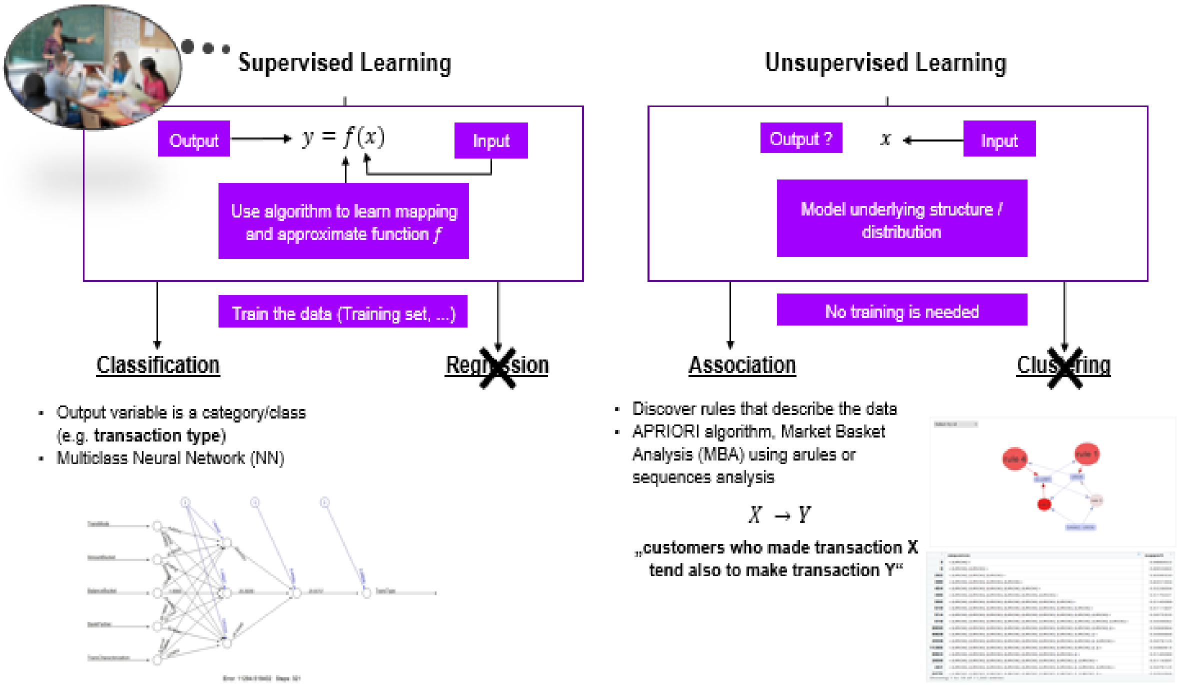

Supervised Learning
Output
$y = f(x)$
Input
Use algorithm to learn mapping and approximate function f
Train the data (Training set, ...)
Classification
Regression
Output variable is a category/class (e.g. transaction type)
Multiclass Neural Network (NN)
Unsupervised Learning
Output ?
x
Input
Model underlying structure / distribution
No training is needed
Association
Clustering
Discover rules that describe the data
APRIORI algorithm, Market Basket Analysis (MBA) using arules or sequences analysis
$X \rightarrow Y$
„customers who made transaction X tend also to make transaction Y"

Figure 4-56: Data mining methods for the 3rd research project

Figure 4-57 below illustrates a high-level concept of the applied neural network for the multiclass prediction of transaction categories. The neural network comprises neurons and weighted edges connecting them. In the feedforward multilayer perceptron (MLP), the neurons are layered, and each layer's neuron is connected to all of the neurons in the two neighbouring layers, with two exceptions: the input layer has no predecessors, and the output layer has no successors. The layers between input and output layers are called hidden layers. The number of hidden layers varies but rarely exceeds two. Regarding the current research experiment, the MLP can be considered as a black box that trains itself by presenting the arguments in the input layer (e.g. 5 features) and the values in the output layer (e.g. 7 categories). The research goal is that the neural network learns the mapping rule based on the input values and thereby is able to generalise the function history, whereby even an untrained x is mapped to a meaningful y.

In the first step of the analysis, the final dataset generated in the pre-processing part is essentially revised using the One-Hot-Procedure. These categorised transactions are trained into a neural network model using the TensorFlow library from Google with

Python. The corresponding Python scripts with the classificationNN_trainAndSave.py file and the classificationNN_loaded.py file are given in appendix E.2. The first Python "classificationNN_trainAndSave.py" script trains a neural network to predict the described features shown in figure 4-57 below. In doing so, the trained model is saved and can be re-used in the second Python script for predictive modelling. Note that the intermediate step with the second Python script "classificationNN_loaded.py" helps the conducted research to save time at a later time within the research approach as the one-time training of the model is relatively time-consuming.

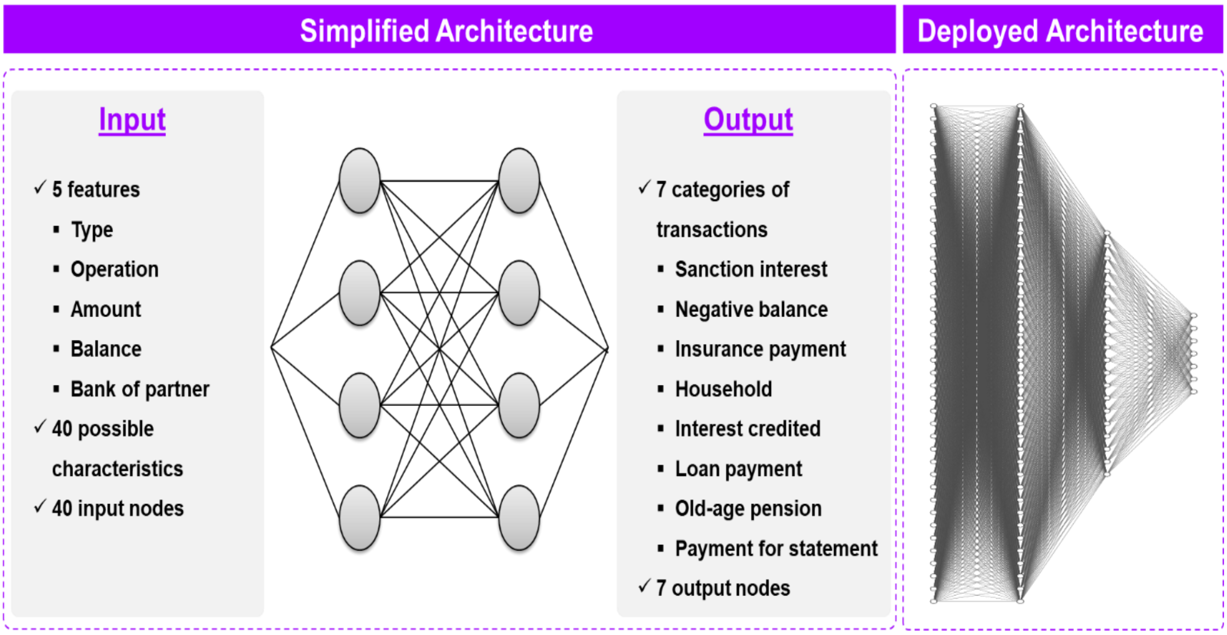


Figure 4-57: High-level concept of neural network multiclass prediction of transaction category

The first Python script "classificationNN_trainAndSave.py" is characterised by the following steps:

(1) First, the final dataset file "dataTrans.csv" must be read in by Python using the application Spyder for further processing.

(2) Since the processed data comprises categorical data, the Python script makes use of a one-hot-encoder method to transform the data into an array. For instance, the number 2 in five possible categories is converted into the vector [0;0;1;0;0] ^T. Ultimately, number 2 is the third element of the array [0,1,2,3,4] comprising five elements.

(3) From the five features (type of transaction, mode of transaction, amount, balance, bank of partner) with 40 different characteristics, a vector with 40 lines is created.

(4) The labels are transformed with the same one-hot-encoder procedure. After all, the Python script creates a vector with seven lines because there are seven possible characteristics given.

(5) The model is trained on 80% of the dataset and the results are stored. The accuracy of the modelling results is tested on the remaining 20% of the dataset.

The second Python script “classificationNN_loaded.py” is based on the first Python script and comprises the following steps:

(6) The script loads the trained model based on the previous steps from the first script and uses it to undertake the corresponding predictions efficiently.

In the second step of the analysis, the categorised transaction dataset is also trained into a neural network using R Studio. The reason behind this is that the developed research approach will improve the comparability, objectivity and reliability of the modelling results from both distinctive analysis steps. The corresponding R-script “Parse4NeuralNetworkAnalyze.R” is given in appendix E.1. Thus, the R-script is characterised by the following steps:

(1) The first lines of the R-code call the complex “dataProcessing.py” Python script to process the data executed in the Spyder Notebook within the Anaconda framework.

(2) The second part describes data mining steps to explore and understand the processed data with the objective of visualising the data with simple bar plots, simple histograms and boxplots. The initial exploratory data analysis is the first step for deepening further data modelling objectives.

(3) The third part demonstrates the mining steps to construct the predictive model. For this purpose, the data was normalised using the max-min normalisation function in R Studio to accurately compare predicted and actual values.

(4) In the next step, the pre-processed dataset is divided into training data (train set) comprising 80% of the observations and the remaining 20% of the observations is assigned to the test dataset (test set).

(5) The subsequent step involves training the neural network model using “neuralnet” package from R Studio.

(6) The results are tested against the test set, and the predicted results are compared with the actual results.

(7) Finally, the modelling results are rounded up using a confusion matrix to compare the number of true/false positives and negatives.

Further mining steps during the data analysis using R Studio include targeting the customer behaviour based on a market basket analysis for the specific transactional categories by applying association rules and frequency analysis. The mining for association rules was used to solve the questions concerning relations between characteristics of the payment categories. In the first step, the analysis deals with exploring strong association rules with the help of the Apriori algorithm, and in the second step the mining process is reinforced by a frequency analysis of the existing payment categories.

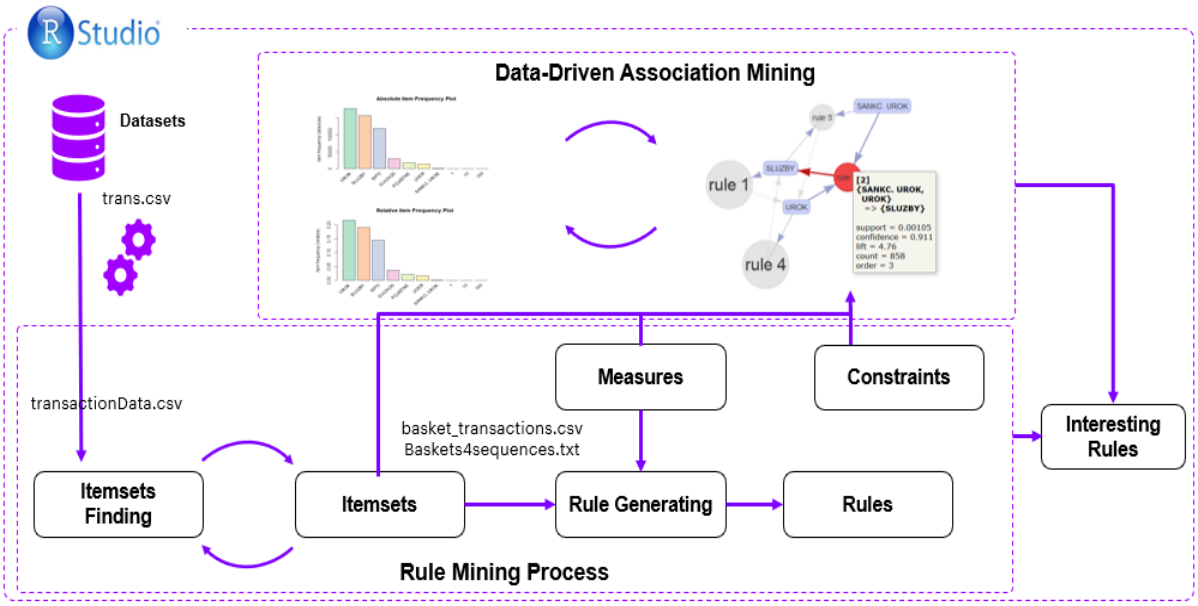
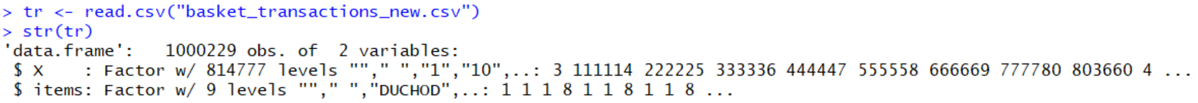


Figure 4-58: High-level reinforced association rule mining pipeline for the payment category

Figure 4-58 above illustrates the mining process to explore customer payment behaviour based on transactional categories. The pipeline integrates a suite of visual exploration and representation such as graph-based visualisation. In connection with these various ways of discovering deeper knowledge within the transactional dataset, it is worth recalling that there is only one single approach that results in true and correct research outcomes. The corresponding R-script “Parse4BasketAnalyze.R” is given in appendix E.1. Thus, the R-script is characterised by the following parts:

(1) In the first step, the R-script imports the pre-processed data from the original dataset file “trans.csv” and stores all ten relevant attributes into a data frame named “transactionData” for continuing processing.

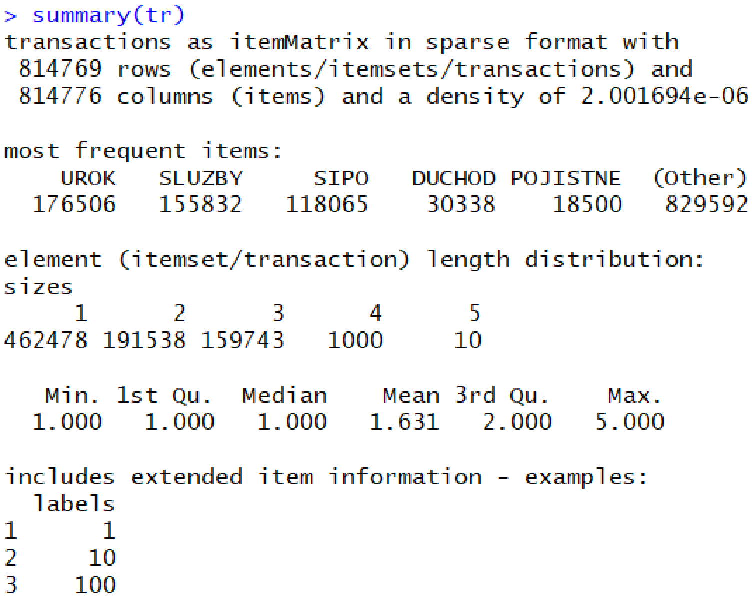
(2) The next steps are cleaning the data frame “transactionData” by initially reducing the data frame into three interesting variables with the following structure shown in figure 4-59 below. The output is stored in an intermediate step as a csv file named “basket_transactions.csv”.



```
> tr <- read.csv("basket_transactions_new.csv")
> str(tr)
'data.frame':	1000229 obs. of  2 variables:
 $ X    : Factor w/ 814777 levels ""," ","1","10",..: 3 111114 222225 333336 444447 555558 666669 777780 803660 4 ...
 $ items: Factor w/ 9 levels ""," ","DUCHOD",..: 1 1 1 8 1 1 8 1 1 8 ...
```

Figure 4-59: Structure of data after pre-processing stage - basket transactions (tr)

(3) The “transactionData” is then reduced into one single item column with all existing transaction categories and stored in another intermediate step as a txt file named “baskets4sequences.txt”.



```
> summary(tr)
transactions as itemMatrix in sparse format with
 814769 rows (elements/itemsets/transactions) and
 814776 columns (items) and a density of 2.001694e-06

most frequent items:
    UROK   SLUZBY     SIPO   DUCHOD POJISTNE  (Other)
  176506   155832   118065    30338    18500   829592

element (itemset/transaction) length distribution:
sizes
     1      2      3      4      5
462478 191538 159743   1000     10

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   1.000   1.000   1.631   2.000   5.000

includes extended item information - examples:
  labels
1      1
2     10
3    100
```

Figure 4-60: Summary of fetched transaction category data - basket transactions (tr)

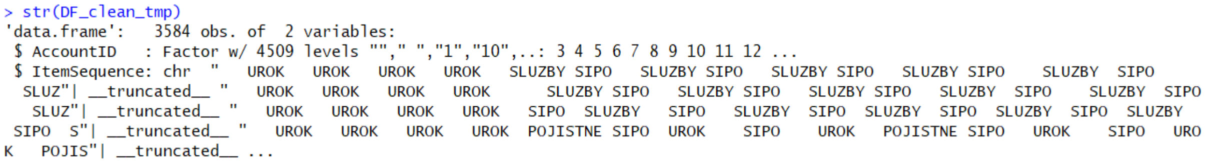
The summary above shows some basic statistics of the fetched categorical dataset; for example, the fact that the dataset is rather sparse with a density just above 200%, ‘UROK’ (interest credited) is the most popular item in the basket and the average transaction contains fewer than two items.

(3) The previous procedure allows the conducted research to store the items into an object and visualise the data by plotting an item frequency chart.

(4) The next step is to mine the rules by using the apriori() function from the R-package “arules” and identifying interesting association rules with graph-based visualisations. Details of the descriptive and predictive analysis results will be displayed in the next chapter.

(5) The subsequent step is conducted to mine the rules by exploring frequent patterns within the item list of payment categories using the cSPADE algorithm from the R-package “clickstream”. Therefore, the data was processed from “transactionData” and cleaned with helping data frame tables for further sequential pattern recognition.

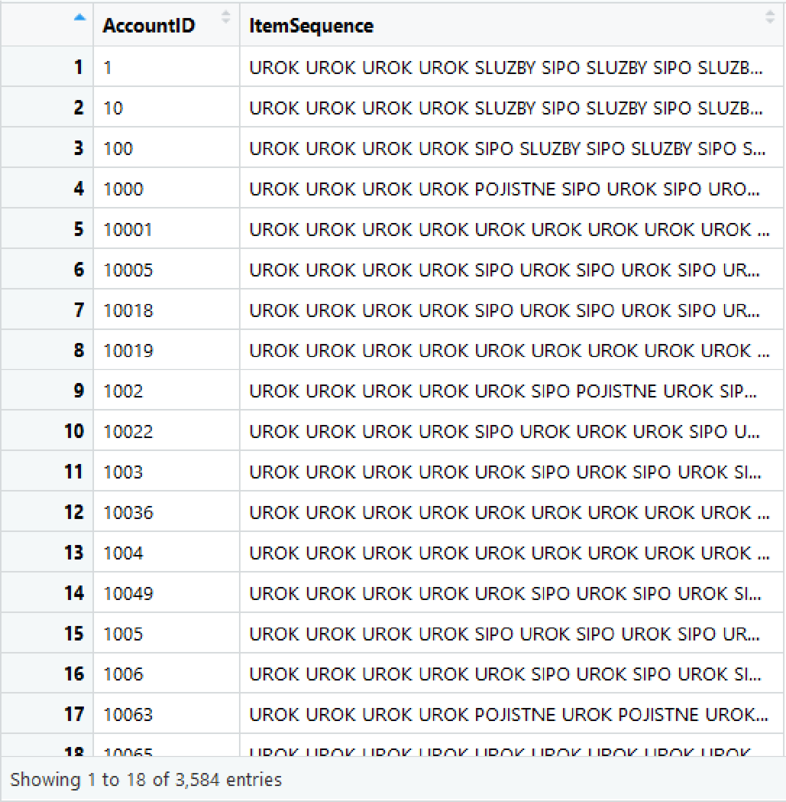
Regarding the pre-processing of mining the sequences of payment transaction data, we processed the dataset developed in step (3) for continuous investigations. Therefore, we prepared the fetched data in several intermediate steps illustrated below, and cleaned the final dataset by replacing existing “,” with blanks within the sequences.



```
> str(DF_clean_tmp)
'data.frame':   3584 obs. of  2 variables:
 $ AccountID   : Factor w/ 4509 levels "",","  ","1","10",..: 3 4 5 6 7 8 9 10 11 12 ...
 $ ItemSequence: chr  "   UROK    UROK    UROK    UROK    SLUZBY SIPO    SLUZBY SIPO    SLUZBY SIPO    SLUZBY SIPO     SLUZBY   SIPO
  SLUZ"| __truncated__ "   UROK    UROK    UROK    UROK       SLUZBY SIPO    SLUZBY SIPO    SLUZBY SIPO    SLUZBY  SIPO     SLUZBY  SIPO
   SLUZ"| __truncated__ "   UROK    UROK    UROK    UROK    SIPO  SLUZBY    SIPO    SLUZBY  SIPO  SLUZBY  SIPO  SLUZBY  SIPO  SLUZBY
 SIPO  S"| __truncated__ "   UROK    UROK    UROK    UROK   POJISTNE SIPO  UROK     SIPO     UROK    POJISTNE SIPO   UROK     SIPO    URO
K   POJIS"| __truncated__ ...
```

Figure 4-61: Structure of data after pre-processing stage - item frequency per account

Subsequently, we split the pre-processed data into lists of frequent sequences using the str_split() function from R-package “stringr” and converted the data to a class of transactions using the “clickstream”-package from R Studio. The final data for mining the transactions frame comprises two variables displaying the account Id and the item sequence of 3,584 interesting accounts. Table 4-6 below shows a sample of the item sequences for each account.



	AccountID	ItemSequence
1	1	UROK UROK UROK UROK SLUZBY SIPO SLUZBY SIPO SLUZB...
2	10	UROK UROK UROK UROK SLUZBY SIPO SLUZBY SIPO SLUZB...
3	100	UROK UROK UROK UROK SIPO SLUZBY SIPO SLUZBY SIPO S...
4	1000	UROK UROK UROK UROK POJISTNE SIPO UROK SIPO URO...
5	10001	UROK UROK UROK UROK UROK UROK UROK UROK UROK ...
6	10005	UROK UROK UROK UROK SIPO UROK SIPO UROK SIPO UR...
7	10018	UROK UROK UROK UROK SIPO UROK SIPO UROK SIPO UR...
8	10019	UROK UROK UROK UROK UROK UROK UROK UROK UROK ...
9	1002	UROK UROK UROK UROK UROK SIPO POJISTNE UROK SIP...
10	10022	UROK UROK UROK UROK SIPO UROK UROK UROK SIPO U...
11	1003	UROK UROK UROK UROK UROK SIPO UROK SIPO UROK SI...
12	10036	UROK UROK UROK UROK UROK UROK UROK UROK UROK ...
13	1004	UROK UROK UROK UROK UROK UROK UROK UROK UROK ...
14	10049	UROK UROK UROK UROK UROK SIPO UROK SIPO UROK SI...
15	1005	UROK UROK UROK UROK SIPO UROK SIPO UROK SIPO UR...
16	1006	UROK UROK UROK UROK UROK SIPO UROK SIPO UROK SI...
17	10063	UROK UROK UROK UROK POJISTNE UROK POJISTNE UROK...
18	10065	UROK UROK UROK UROK UROK UROK UROK UROK UROK

Showing 1 to 18 of 3,584 entries

Table 4-6: View of the table - item sequences per account

(6) In the last step, we analysed the sequences within the item list comprising different payment categories by applying the "arulesSequences" package from R Studio. Details of the descriptive and predictive analysis results for the sequences analysis will be displayed in the next chapter.

Finally, regarding the knowledge gained from the systematic literature review, no approaches have currently evaluated a categorised transactional payment data by performing a market basket analysis using association rules and conducting a frequency analysis. The mining process proposed for the research design also combines a complicated neural network in which non-pre-defined attribute selection is at the forefront, and in which the overall modelling results are double-examined in accordance with the research objectivity. The research approach for the last research project can be considered as a hybrid research approach combining supervised and unsupervised learning algorithms, where attribute selection and aggregation are tested out easily and the data model can be tuned to provide maximal value with respect to the prediction outcomes, especially for the applied machine learning algorithms.

4.7. Summary of the research design and methodology

This chapter has presented the research approach and methodology applied to analyse customer behaviour based on transactional payments. Taking together all sections of this chapter, the research design and methodology can be summed up in one research process that applies best practices to avoid common pitfalls.

The first step within the research process is to understand the research issues related to the meaning of the underlying dataset. In particular, the entire chapter provides a clear understanding of the research problem that current research work aims to solve and examine, as well as how it generally affects the research field of transactional payment behaviour. Regarding the research objectives, the available dataset is explored, described and evaluated during the data pre-processing phase for every single pre-defined research project. The second step through the research process deals with the data model and data mining tool selection for the three research projects. It emerges that the model and tool selection is a crucial success factor along the designed research approach. For instance, the selected tools assist current research in quite different ways by visualising the pre-processed data to gain more understanding of the data-related research questions (i.e. identify outliers, quality checks of the data, etc.) and ensures the objectivity, quality and independence of the research analysis.

The data preparation phase includes various activities along the proposed research process. It starts by collecting suitable data for every research project, consolidating and cleaning it, selecting the appropriate data (sampling, variables and features) according to the research questions and transforming the data (i.e. create new variables) into an entire dataset for further data analysis. Finally, I have declared various datasets for all different research projects. A further relevant part of the data modelling process through the data analysis and visualisation stages is the procedure introduced for building relevant datasets in every single research project.

The modelling and evaluation phase deals with the applied theoretical concepts aligned to the pre-defined research objectives for the research projects and their corresponding pre-processed as well as optimised datasets in case of building cost-sensitive data models. Therefore, the mining methods differed between supervised and unsupervised learning algorithms are introduced, which are used in the various research projects. Regarding the research objectives, the appropriate mathematical techniques can be applied to answer the research questions. However, the data mining approach involves constructing a range of predictive models for assessing the performance of calculated credit scorings or cross-sell candidates, in which various supervised classification and regression algorithms are utilised.

In conclusion, the power of the different mining approaches for evaluating transactional payment behaviour will be determined through the analysis outcomes of every single research project in the next chapter, although the initial data exploration reveals that advanced mining methods such as supervised classification and regression algorithms can be represented and learned.

Chapter 5

5. Research Results

This chapter deals with the application of the algorithms introduced in the previous sections and shows the results of the mining algorithm applied in every pre-defined research project of the scrutinised transactional payment behaviour. The evaluation results of the unique research projects can be roughly divided into the main sections of description and prediction results and grouped in sub-sections covering the research analysis results of the three research projects described in the introduction part of this thesis.

5.1. Descriptive results of individual research projects

The section focuses on the descriptive and diagnostic evaluation results of the thesis, including a comprehensive description of observations in terms of payment behaviour when processing the transactional dataset from Berka and the data mining procedures for pattern detection in categorised payment transactions. Before describing the various descriptive results of the pre-processed datasets to create the target attributes and a customer-oriented payment behaviour profile history in case of credit scoring, cross-selling or categorised transactions, I want to highlight some underlying assumptions. When using the evaluation results to identify prospects for credit card usage (cross-selling) or calculating the creditworthiness (credit scoring) within the existing customer base, I assume that the given dataset from Berka is a representative sample of the entire customer base. For instance, van der Putten (1999) stated that there are some anomalies in the dataset, such as the huge increase in active accounts and total balance over only a few years.

The subsequent sections provide a detailed overview of the descriptive and diagnostic analysis results across the various research projects. It provides insight into what occurred across the pre-processing steps of the different datasets into the various datasets in case of comparing interesting variables with each other or what went wrong or right during the building process for the data models.

5.1.1. Analysis results of the credit scoring dataset

The following section describes the overall evaluations and findings made by analysing the credit scoring dataset. In the pre-processing phase, different tables were joined into single cleaned temporary tables before different target attributes could be created: average income and expenses, balance at end of month, indicator whether negative balance, district-relevant data, and number of permanent orders. Independent attributes mainly comprised socio-demographics derived from the client's district, information on loans and totals on balance and the different transaction operations (van der Putten, 1999). Regarding the various processed intermediate steps, the calculated totals – for instance – help to extract the frequency and the monetary characteristics from a transactional payment history. For credit scoring, labelled as res_azure.csv, we only considered 682 normalised clients. However, Pijls (1999) already recognised that "the utilization of credit cards is poor". Independent of this, Miksovsky, Zelezny, Stepankova, Pechoucek (1999) mentioned that the original Berka dataset is a good example of a "practical dataset where there are no strong and easy-to-find dependencies without deep expert domain knowledge".

Figure 5-1 below shows an overview of the created MS Azure Mini Map implemented to realise the research experiments at scale that are being constructed in the first field of research. The mini map provides a high-level overview of the research setup, which is very powerful and useful for the research understanding.

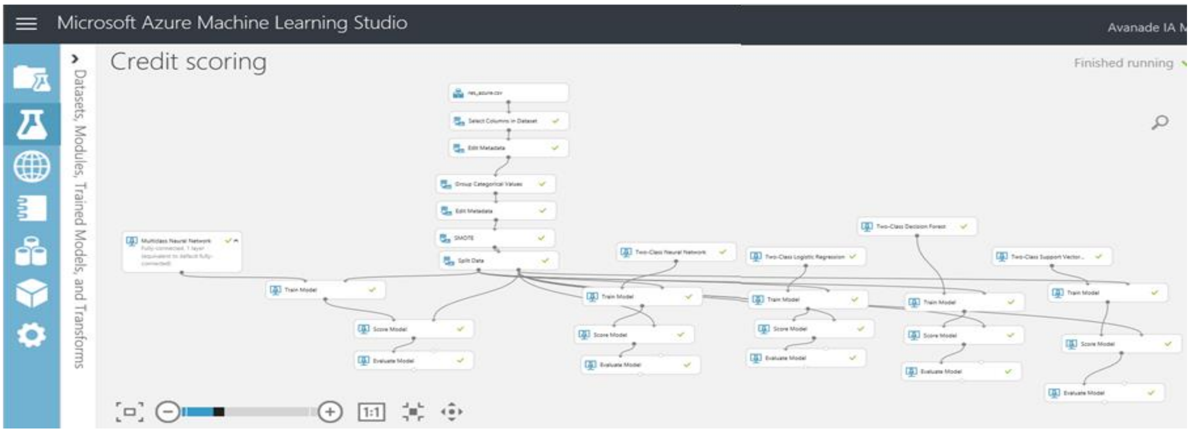


Figure 5-1: MS Azure's Mini Map overview for processed dataset (res_azure.csv)

The first step was to read the pre-processed dataset “res_azure.csv” and exclude non-relevant columns such as ‘loan_id’, ‘account_id’, ‘client_id’ and ‘negativeBalance’ from the dataset. The next step was to flag the column ‘status’ as categorical and indicate the grouped categorical values ‘A’ and ‘C’ as good scorings. Before splitting the data into test (20%) and training (80%) data, I have applied the SMOTE module in Azure ML Studio due to the given imbalanced dataset (see appendix C.1 - section 4.2 - figure 4-6). The module returns an advanced “res_azure” dataset that contains the original samples, plus an additional number of synthetic minority samples with a percentage of up to 400%. I have connected the original dataset with the SMOTE module because the targeted category ‘status’ has a rare population of default credits (11.1%). The under-represented class increases with cases in the “res_azure” dataset in a balanced way. The outcome is a more general sample by increasing the percentage of only minority credit default cases using multiples of 400 and a number of nearest neighbours of 1. Note that the SMOTE function does not guarantee building more accurate models. The final steps of the mini map show the connecting (intermediate) steps such as various applied mining algorithms for the prepared dataset with the aim to train, score and evaluate the predictive models. Detailed results will be explained in the next section.

Appendix C.1 comprises profound visual descriptions (consisting of boxplot, histogram and statistics) for the credit scoring dataset “res_azure.csv”. The research initially considers interesting variables by using the visualisation functions of MS Azure ML and thereafter current research seeks to gain more deep insights while comparing exploratory variables of the pre-processed dataset. The main results of the description phase can be summarised as follows:

Figure C-3 “EDA - status” depicts the distribution of the amount along the variable ‘status’. As can be seen in the histogram, the values ‘B’ and ‘D’ are under-represented and indicate bad credit scorings. With the histogram for the variable ‘negative balance’, figure C-4 “EDA - negative balance” shows that a minority (76 accounts) of the selected 682 accounts have a negative balance. Figure C-5 “EDA - permanent orders” describes that the majority of permanent orders per account do not exceed three orders. The corresponding histogram shows that roughly 16.3% of the processed accounts have four or five permanent orders. As can be seen in figure C-6 “EDA - average income”, the distribution of the values for the variable ‘average income’ is inclined to the left. From the statistical analysis (mean = 29,266) and the displayed histogram, it can be deducted that most banking clients have a lower average income compared with the entire customer base. The exploratory data analysis for the variable ‘average expenses’ in figure C-7 “EDA - average expenses” shows that the clients have on an average less expenditures (mean = -27794) than incomes. This can also be concluded from the histogram, which reflects that the distributed values are inclined to the right. However, the exploratory data analysis for the variable ‘average balance’ highlighted in figure C-8 “EDA - average balance” regarding the preliminary analyses of the banking accounts shows that the distribution of the balances is slightly skewed to the left with many banking accounts. Beyond that, it is appropriate to mention that the average salary (mean = 9,469) explored in figure C-14 “EDA - average salary” is heavily distributed to the left for many banking accounts. Finally, it should be highlighted that these characteristics from the underlying processed dataset are a moderate sign of the good healthiness of a bank account whenever the expenditures are not higher than the revenues. This fact generally has a positive impact on the credit scoring outcome or increases the opportunity for cross-selling activities.

Figure C-10 “EDA - age” and figure C-11 “EDA - sex” reveal that the account owner is female or male with a probability of at least 50% and with an average age of 40.9, which is evenly distributed over the customer base, assuming that Berka has provided a representative dataset. The variable ‘no inhabitant’ visualised in figure C-12 “EDA - no inhabitants” shows that two-thirds of the account owners’ district are associated with a district whose number of inhabitants are greater than the average of 263,844. For instance, this circumstance can be an influence factor when calculating credit scorings. Further exploratory analysis results of the socio-demographic variables such as given in figure C-13 “EDA - urban ratio”, figure C-15 “EDA - unemployment95”, figure C-16 “EDA - unemployment96”, figure C-17 “EDA - crimeratio95”, figure C-18 “EDA - crimeratio96” and figure C-19 “EDA - entrepreneur’s ratio” can also have an indirect impact by assessing the credit scores.

Taken together, the descriptive and diagnostic analysis can be gathered out of two building knowledge blocks. By generating new associated knowledge by advanced data exploration as well as gaining more valuable and meaningful insights into the pre-processed data, current research has compared interesting variables with each other and especially with socio-demographic variables, whereby the main results will be summarised in the following paragraphs.

Figure 5-2 “EDA - age, average balance, average expenses and average income compared with unemployment95” demonstrates that the older the bank account owner, the higher the value of the variable ‘unemployment95’. Accordingly, the average expenses of the bank accounts will also decrease, although the average balance of every bank account will remain nearly equally distributed throughout the customer base. Whether this interesting observation might have an impact in the modelling building process will be assessed at a later stage.

**Age, average balance, average expenses and average income compared with unemployment95**

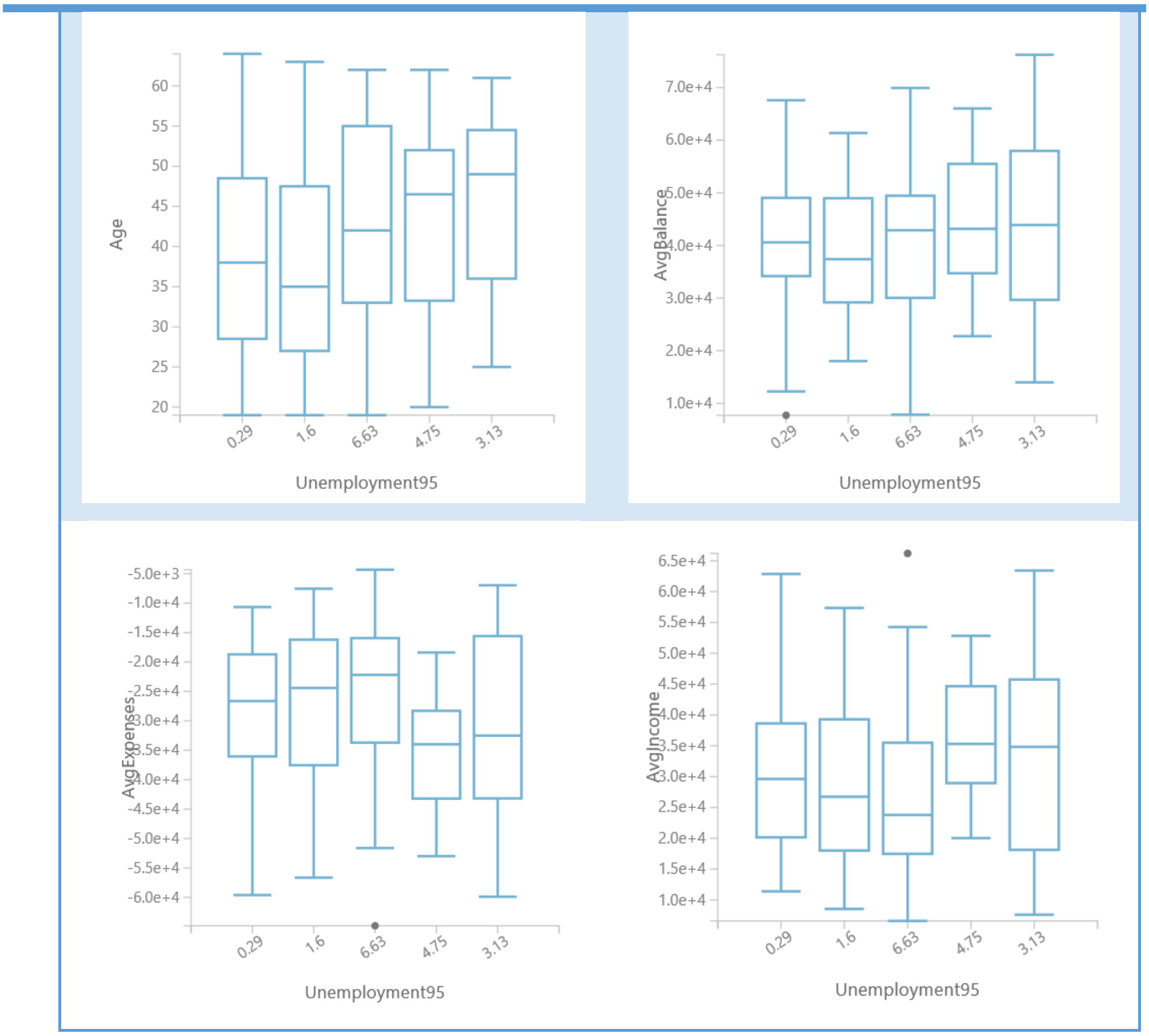


Figure 5-2: Exploratory data analysis - various variables compared with unemployment95

The boxplots in figure 5-3 “average income compared with sex” show that there is no significant difference between females and males in the context of average income. However, the boxplots in figure 5-2 “average income compared with unemployment95” underline that the average income correlates with a high value of ‘unemployment95’.

The variable ‘sex’ is distributed equally around the selected customer base to both variables ‘entrepreneur’s ratio’ and ‘status’. As can be seen in figure 5-3, the attributes ‘entrepreneur’s ratio’ and ‘average income’ have also some detected outliers for male bank customers.

Sex **to entrepreneur’s ratio** **to status** **average income to**

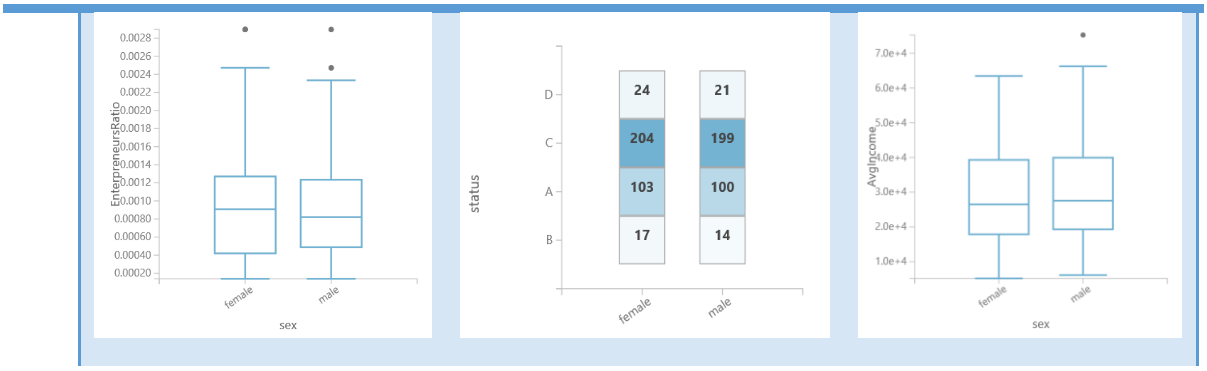


Figure 5-3: Exploratory data analysis - sex compared with various variables

Figure 5-4 “EDA - average expenses / income compared with crime ratio95” shows that the comparison of both variables ‘average expenses’ and ‘average income’ to ‘crime ratio95’ appear to be the same, behaving like a mirror image of each other. The spreading of these values proves that a balanced account does not correlate with a specific ‘crime ratio95’ value.

crime ratio95 **to average expenses** **to average income**

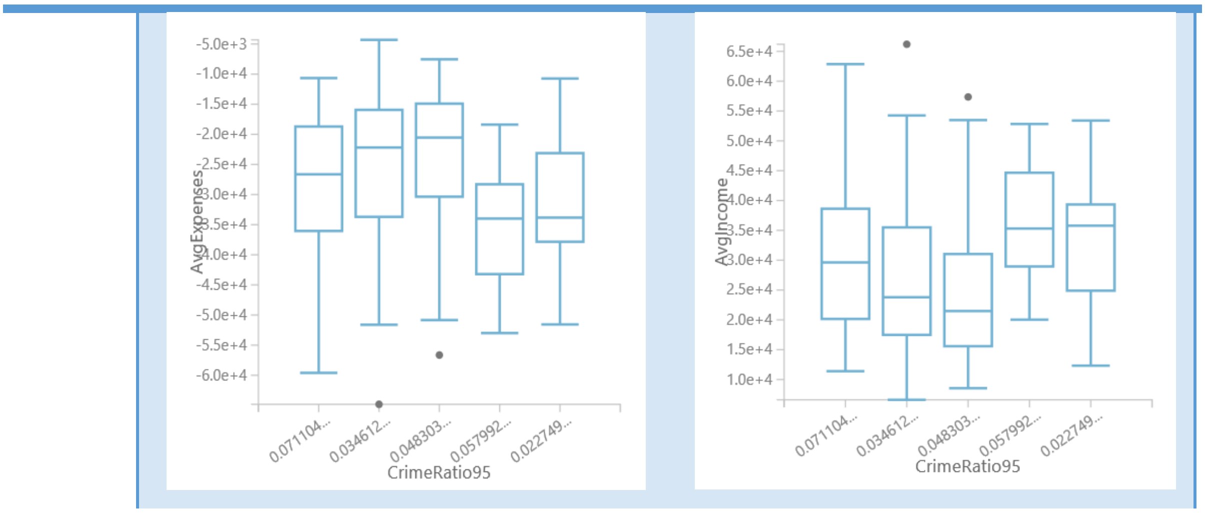
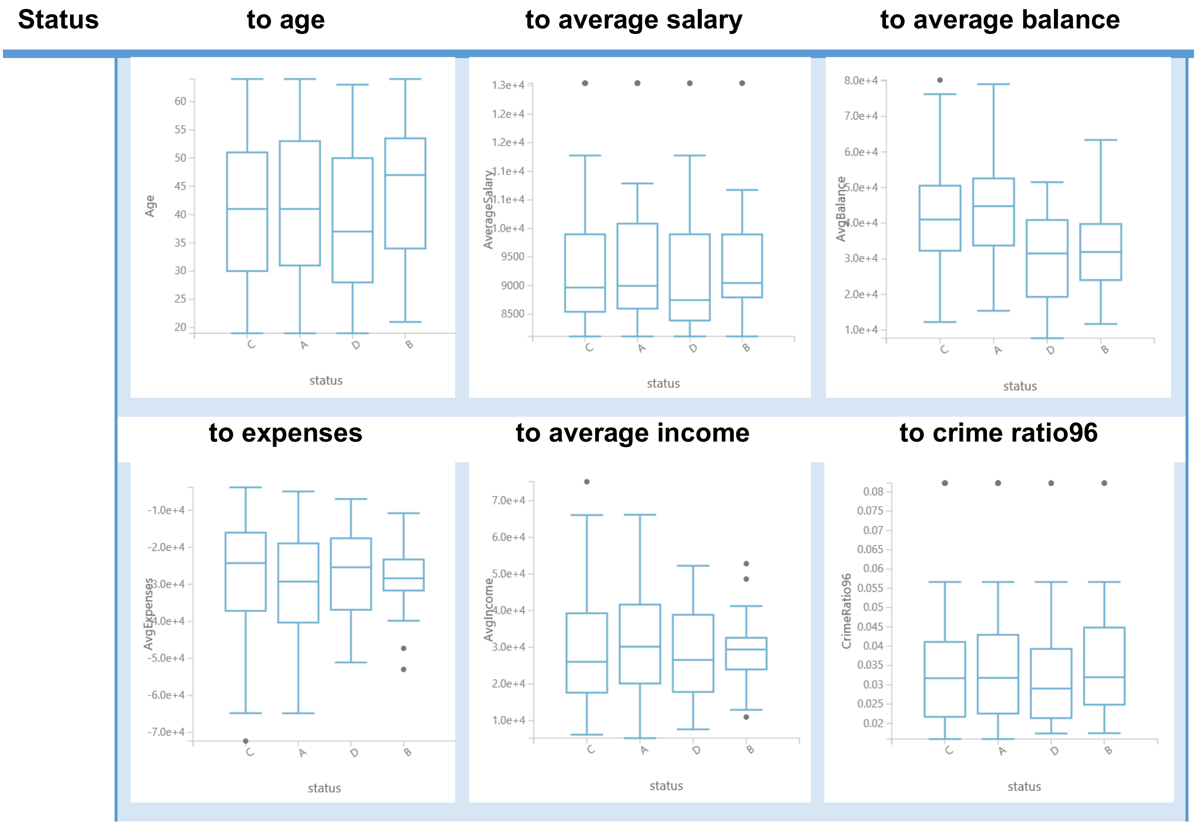
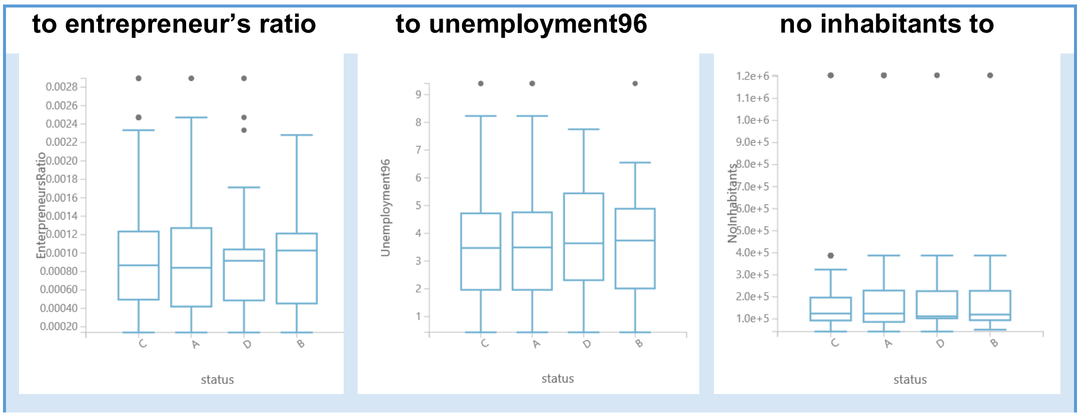


Figure 5-4: Exploratory data analysis - average expenses / income compared with crime ratio95

The following descriptive examination results refer especially to the most important variable ‘status’ within the processed dataset as the predictive models calculate the creditworthiness based on this output variable. In principle, we can say that customers with an average of 47 years old (assigned to status ‘B’) or 37 years old (assigned to status ‘D’) receive a bad credit scoring, and the average salary of the peer group from status ‘D’ is considerably less than for the others. The peer groups from status ‘A’ and ‘C’ have a higher average balance than the others, which also indicates that they demonstrably handle money more responsibly. Having examined this, all peer groups have nearly the same level of average expenses and the average income for the peer groups ‘D’ and ‘B’ are on average not significantly different than the others. It can therefore be concluded that the predictive model must include further relevant variables to more accurately calculate the credit scores.

When comparing the variable ‘status’ with the other socio-demographic variables, I have made quite a few interesting observations about the entrepreneur’s ratio and no inhabitants. The boxplots of “status compared with entrepreneur’s ratio” shows that clients assigned to the peer groups ‘D’ and ‘B’ show high values for the entrepreneur’s ratio. It seems that these clients probably do not invest in promising business ideas. In this context, it is conspicuous that the peer group ‘D’ is associated with cities with a lower population.




to entrepreneur's ratio

to unemployment96

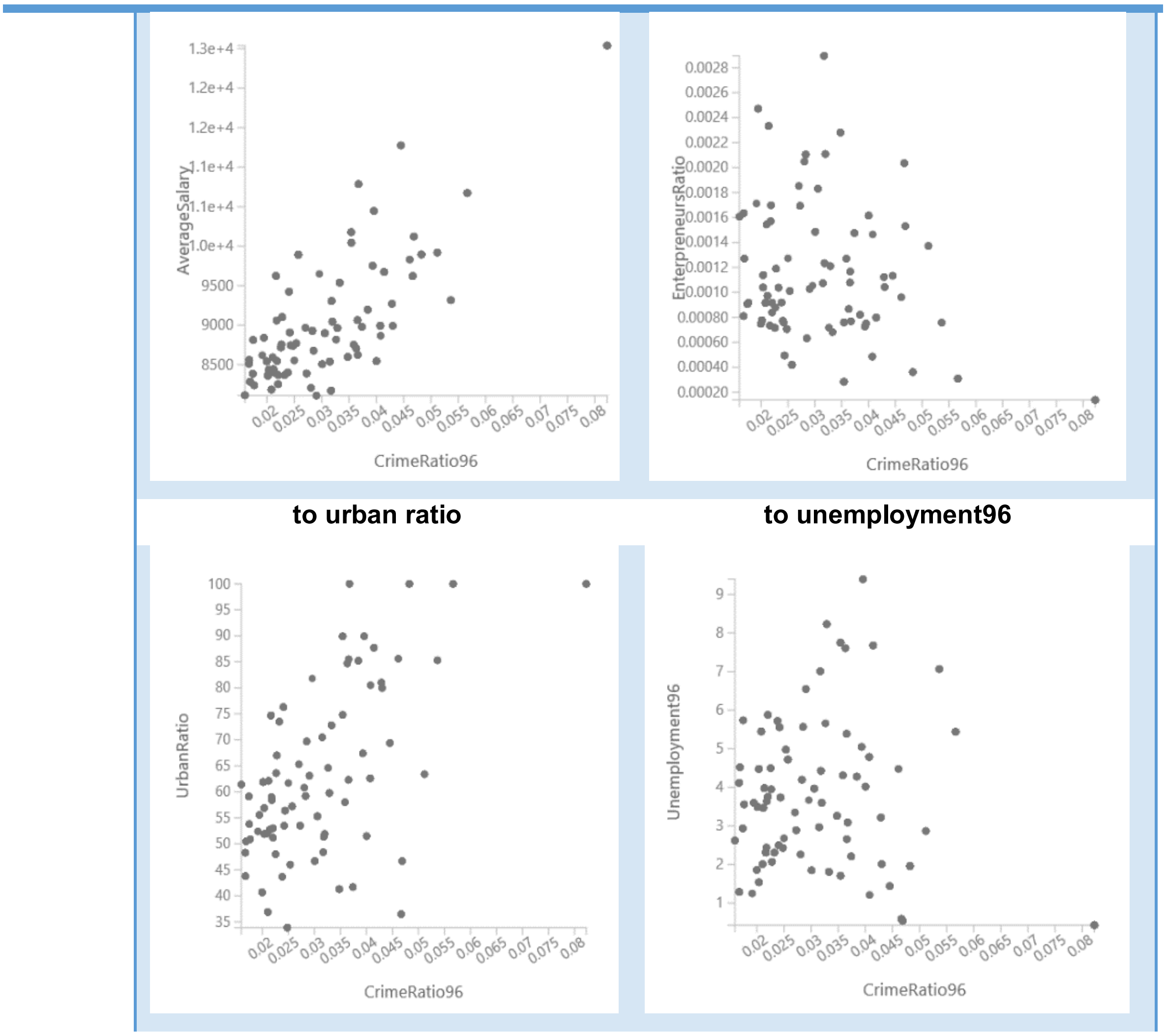
no inhabitants to

Figure 5-5: Exploratory data analysis - status compared with other variables

The following exploratory data analysis in figure 5-6 "EDA - crime ratio96 compared with other variables" provides examination results in terms of the socio-demographic variable 'crime ratio96' compared with the average salary, entrepreneur's ratio, urban ratio and unemployment96. The scatterplot for average salary shows a positive correlation between the two variables, as the variable 'crime ratio96' increases while the value of the 'average salary' increases. It can be therefore concluded that if the salary increases the 'crime ratio96' value dramatically declines, as the scatterplot displays a concentration of data points (crime ratio96: below 0.04 and average salary: below 9,500) in the lower left corner of the plot. The lower the crime ratio96, the lower the average salary of the customer. The scatterplot of the entrepreneur's ratio indicates a slightly low negative relationship with the average salary of a customer. The diagram highlights that the more strongly that the 'crime ratio96' value grows, the lower the value of the entrepreneur's ratio. At (0.04, 0.0005 - 0.00016), it indicates that the value of 0.04 crime ratio96 in the customer base will lead to a reduced entrepreneur's ratio between 0.0005 and 0.00016. It can be argued that an increased entrepreneur's ratio does not have a significant impact on the crime ratio96, assuming that a lower value indicates that crime does not exist.

The graphs given in figure 5-6 "EDA - crime ratio96 compared with other variables" illustrate a low positive correlation between the values of 'crime ratio96' compared with the socio-demographic variables 'urban ratio' and 'unemployment96'. Both scatterplots show how the 'crime ratio96' value might change depending on the 'urban ratio' or 'unemployment96' rate. It can be concluded from the weak positive correlation graphs "crime ratio96 compared with urban ratio" and "crime ratio96 compared with unemployment96" that as the 'crime ratio96' values increase, the 'urban ratio' and 'unemployment96' moderately increase. The formation of clusters happens from below ‘crime ratio96’ values in the urban ratio area of 45-65. Accordingly, the plot shows a total of roughly three coherent clusters for the ‘unemployment96’ values, whereby the first cluster includes the area between 1 to 3, the second cluster the area between 3 and 5 and the third cluster in the area between 5 and 6.

Crime ratio96 **to average salary** **to entrepreneur’s ratio**



to urban ratio **to unemployment96**

Figure 5-6: Exploratory data analysis - crime ratio96 compared with other variables

The following paragraphs describe the descriptive and diagnostic analysis results concerning the evaluation of the model-building process for optimising the predictive modelling results. Therefore, detailed predictive results will be presented in the next section. Figure 5-7 below provides an overview of the created MS Azure Mini Map implemented to realise the research experiments at scale for the optimised pre-processed dataset, which is constructed as the concluding part for the first field of research. The mini map provides a high-level overview of the research design, which is also aligned with the preceding research questions. The research approach ensures that the evaluations are independent and objective as the research design corresponds with the upper MS Azure Mini Map diagram excluding minor data-related adjustments to establish the comparability of the research results.

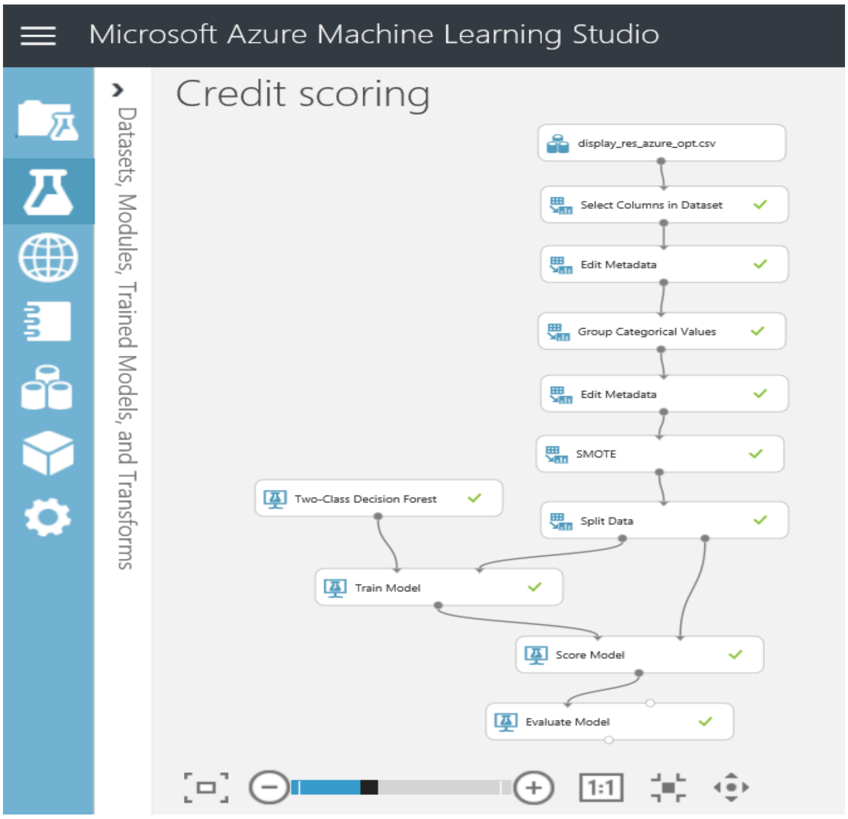

Microsoft Azure Machine Learning Studio
Datasets, Modules, Trained Models, and Transforms
Credit scoring
display_res_azure_opt.csv
Select Columns in Dataset
Edit Metadata
Group Categorical Values
Edit Metadata
SMOTE
Two-Class Decision Forest
Split Data
Train Model
Score Model
Evaluate Model
1:1

Figure 5-7: MS Azure's Mini Map overview for optimised dataset (display_res_azure_opt.csv)

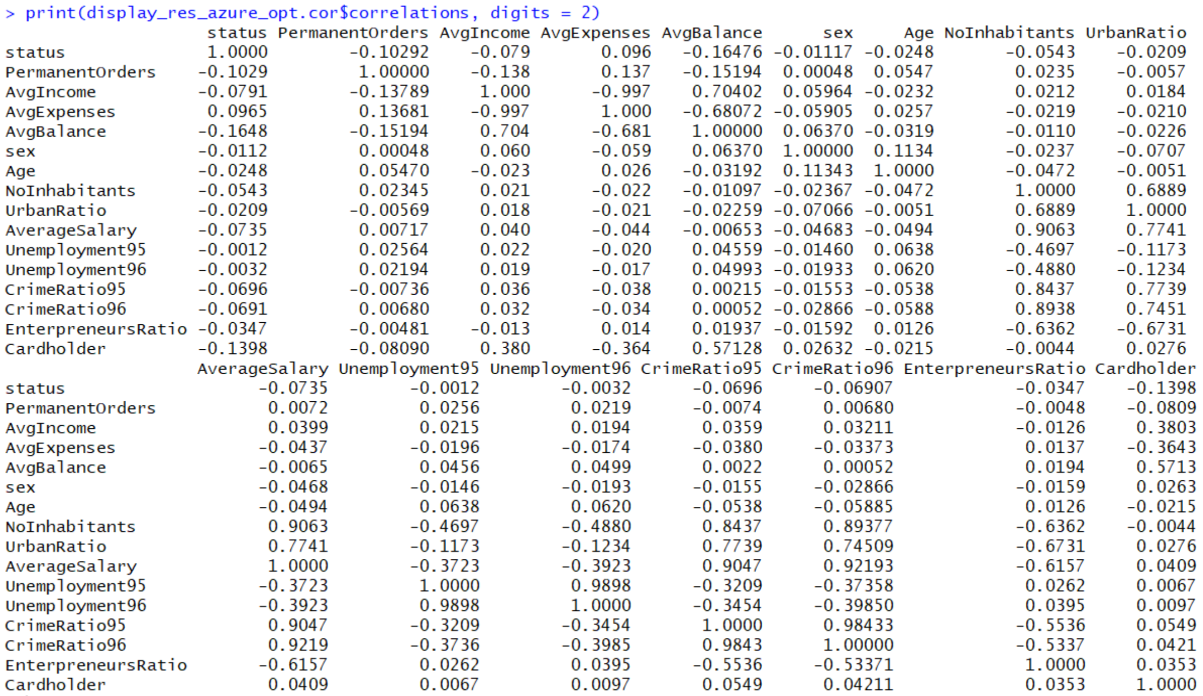
The first step was to read the pre-processed dataset “display_res_azure_opt.csv” and include only relevant attributes such as ‘AvgIncome’, ‘AvgExpenses’, ‘AvgBalance’ and ‘Age’ from the dataset. The selection process of the above variables is based on the statistical calculations when estimating the variable importance for the data model. Detailed results will be introduced later in this section. The next step was to flag the column ‘status’ as categorical and indicate the grouped categorical values ‘A’ and ‘C’ as good scorings. The following connected “Edit Metadata” module in MS Azure ensures that the values (categorical) and the data types (unchanged) in the dataset are not altered after the previous operation. The module changes the ‘status’ field to a class label (target variable) as these are the values that I want to predict. Note that the data values will not change; rather, the modules only ensure that the applied supervised learning algorithms handle the data correctly. Next up in the mini map is the applied SMOTE module in Azure ML Studio due to the given imbalanced dataset. The module returns a “display_res_azure_opt” dataset that contains the original samples, plus an additional number of synthetic minority samples with a percentage of up to 400%. The under-represented class of default credits (74 out of 674 banking clients) will be increased with cases in the “display_res_azure_opt” dataset in a balanced way using multiples of 400 and a number of nearest neighbours of 1. The more general sample will support the objectivity of the evaluation results. The module “Split Data” splits the rows of the prepared dataset into two distinct datasets with a fraction of 20% test data and 80% training data. The final steps of the mini map overview displayed in figure 5-7 above show the connecting (intermediate) steps to the applied two-class decision forest algorithms from the MS Azure ML library for the optimised dataset with the aim to train, score and evaluate the predictive models. Detailed predictive results will be explained in the next section.

As mentioned in the previous paragraph, the last sections will describe the descriptive results while extracting important variables from the random forest model using varImp() function from the R-package “caret” to create an optimised classification model. The function will help to assess the prediction power of the chosen random forest model. In the first step, current research computes a correlation matrix using the R function cor() on the optimised dataset “display_res_azure_opt” to evaluate the linear dependence between two variables. The Pearson correlation coefficient measures every possible pair of variables and correlation coefficients are shown in figure 5-8 below.

The results matrix below shows that the Pearson correlation between the variables ‘AvgIncome’ and ‘AvgExpenses’ is about -0.997, which indicates that there is a strong negative relationship between the variables as ‘AvgIncome’ increase and ‘AvgExpenses’ decreases. The Pearson correlation coefficient of 0.704 between ‘AvgIncome’ and ‘AvgBalance’ indicates a moderate positive relationship. Therefore, an increasing average income has a positive effect on the average balance. Another observable pattern is the correlation between the variables ‘AvgExpenses’ and ‘AvgBalance’. The coefficient of -0.681 indicates a negative relationship, which means that an increased average expense will result in a reduced average balance.

However, the conducted research does not emphasise the low Pearson correlation coefficients given in the results matrix due to the research design, in which initially significant variables were considered and receive higher priority along the research analysis. Nonetheless, this research approach does not mean that no relationship exists between the remaining variables since the variables ultimately may have a non-linear relationship or will be out of scope like the analysis between the demographic variables (i.e. ‘NoInhabitants’ vs. ‘CrimeRatio96’ results in 0.8938, etc.). The outcomes of the variables ‘PermanentOrders’ or ‘Age’ add much to the underlying research proceedings since most of the correlations are close to 0 and thus have almost no linear relationship with the remaining variables, although the varImp() function shows a certain relevance for the modelling building process. Details will be provided in the following paragraphs.

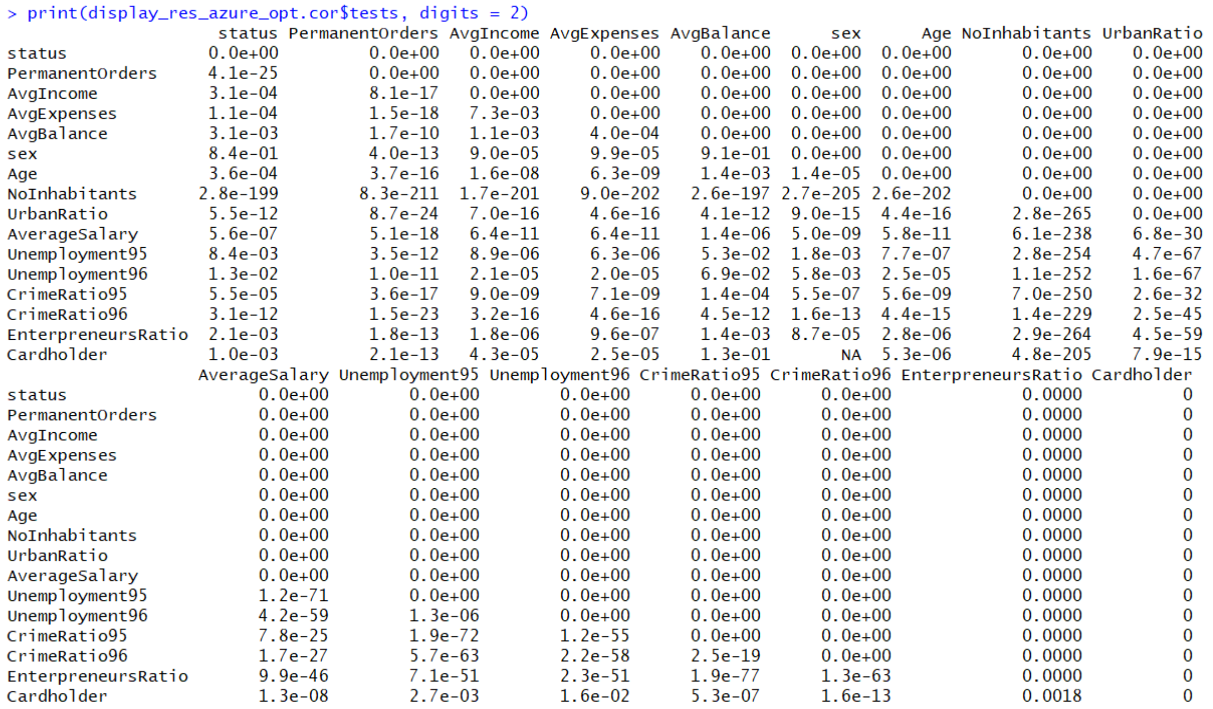
The other extreme is the correlation results of the variable ‘AverageSalary’, which shows a different strength of linear relationships with the other variables but ultimately indicates less importance than the variable ‘PermanentOrders’ when applying the varImp() function. A strong uphill linear relationship is given between ‘AverageSalary’ and ‘NoInhabitants’ (0.9063), ‘AverageSalary’ and ‘UrbanRatio’ (0.7741), ‘AverageSalary’ and ‘CrimeRatio95’ (0.9047), ‘AverageSalary’ and ‘CrimeRatio96’ (0.9219). A weak uphill linear relationship is given between ‘AverageSalary’ and ‘Unemployment95’ (-0.3723) as well as ‘AverageSalary’ and ‘Unemployment96’ (-0.3923). The overall diagnostic check by computing the correlation matrix also shows that the predictive results of the applied two-class logistic regression algorithm is reliable since there is not a high number of correlations given in the matrix below.



```
> print(display_res_azure_opt.cor$correlations, digits = 2)
                    status PermanentOrders AvgIncome AvgExpenses AvgBalance      sex     Age NoInhabitants UrbanRatio
status              1.0000        -0.10292    -0.079       0.096   -0.16476 -0.01117 -0.0248       -0.0543    -0.0209
PermanentOrders    -0.1029         1.00000    -0.138       0.137   -0.15194  0.00048  0.0547        0.0235    -0.0057
AvgIncome          -0.0791        -0.13789     1.000      -0.997    0.70402  0.05964 -0.0232        0.0212     0.0184
AvgExpenses         0.0965         0.13681    -0.997       1.000   -0.68072 -0.05905  0.0257       -0.0219    -0.0210
AvgBalance         -0.1648        -0.15194     0.704      -0.681    1.00000  0.06370 -0.0319       -0.0110    -0.0226
sex                -0.0112         0.00048     0.060      -0.059    0.06370  1.00000  0.1134       -0.0237    -0.0707
Age                -0.0248         0.05470    -0.023       0.026   -0.03192  0.11343  1.0000       -0.0472    -0.0051
NoInhabitants      -0.0543         0.02345     0.021      -0.022   -0.01097 -0.02367 -0.0472        1.0000     0.6889
UrbanRatio         -0.0209        -0.00569     0.018      -0.021   -0.02259 -0.07066 -0.0051        0.6889     1.0000
AverageSalary      -0.0735         0.00717     0.040      -0.044   -0.00653 -0.04683 -0.0494        0.9063     0.7741
Unemployment95     -0.0012         0.02564     0.022      -0.020    0.04559 -0.01460  0.0638       -0.4697    -0.1173
Unemployment96     -0.0032         0.02194     0.019      -0.017    0.04993 -0.01933  0.0620       -0.4880    -0.1234
CrimeRatio95       -0.0696        -0.00736     0.036      -0.038    0.00215 -0.01553 -0.0538        0.8437     0.7739
CrimeRatio96       -0.0691         0.00680     0.032      -0.034    0.00052 -0.02866 -0.0588        0.8938     0.7451
EnterpreneursRatio -0.0347        -0.00481    -0.013       0.014    0.01937 -0.01592  0.0126       -0.6362    -0.6731
Cardholder         -0.1398        -0.08090     0.380      -0.364    0.57128  0.02632 -0.0215       -0.0044     0.0276
                   AverageSalary Unemployment95 Unemployment96 CrimeRatio95 CrimeRatio96 EnterpreneursRatio Cardholder
status                   -0.0735        -0.0012        -0.0032      -0.0696     -0.06907            -0.0347    -0.1398
PermanentOrders           0.0072         0.0256         0.0219      -0.0074      0.00680            -0.0048    -0.0809
AvgIncome                 0.0399         0.0215         0.0194       0.0359      0.03211            -0.0126     0.3803
AvgExpenses              -0.0437        -0.0196        -0.0174      -0.0380     -0.03373             0.0137    -0.3643
AvgBalance               -0.0065         0.0456         0.0499       0.0022      0.00052             0.0194     0.5713
sex                      -0.0468        -0.0146        -0.0193      -0.0155     -0.02866            -0.0159     0.0263
Age                      -0.0494         0.0638         0.0620      -0.0538     -0.05885             0.0126    -0.0215
NoInhabitants             0.9063        -0.4697        -0.4880       0.8437      0.89377            -0.6362    -0.0044
UrbanRatio                0.7741        -0.1173        -0.1234       0.7739      0.74509            -0.6731     0.0276
AverageSalary             1.0000        -0.3723        -0.3923       0.9047      0.92193            -0.6157     0.0409
Unemployment95           -0.3723         1.0000         0.9898      -0.3209     -0.37358             0.0262     0.0067
Unemployment96           -0.3923         0.9898         1.0000      -0.3454     -0.39850             0.0395     0.0097
CrimeRatio95              0.9047        -0.3209        -0.3454       1.0000      0.98433            -0.5536     0.0549
CrimeRatio96              0.9219        -0.3736        -0.3985       0.9843      1.00000            -0.5337     0.0421
EnterpreneursRatio       -0.6157         0.0262         0.0395      -0.5536     -0.53371             1.0000     0.0353
Cardholder                0.0409         0.0067         0.0097       0.0549      0.04211             0.0353     1.0000
```

Figure 5-8: Correlation overview of the optimised dataset (display_res_azure_opt)

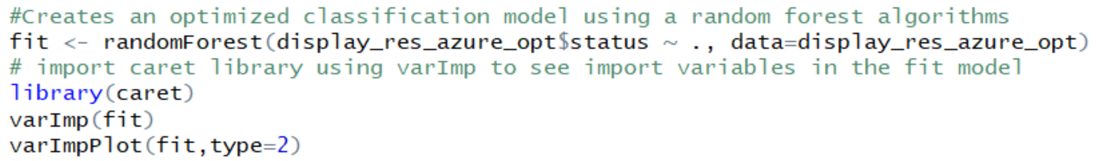
The subsequent step involves testing the correlation results by performing a correlation test on the optimised dataset “display_res_azure_opt”.



```
> print(display_res_azure_opt.cor$tests, digits = 2)
                     status PermanentOrders AvgIncome AvgExpenses AvgBalance      sex      Age NoInhabitants UrbanRatio
status              0.0e+00         0.0e+00   0.0e+00     0.0e+00    0.0e+00  0.0e+00  0.0e+00       0.0e+00    0.0e+00
PermanentOrders     4.1e-25         0.0e+00   0.0e+00     0.0e+00    0.0e+00  0.0e+00  0.0e+00       0.0e+00    0.0e+00
AvgIncome           3.1e-04         8.1e-17   0.0e+00     0.0e+00    0.0e+00  0.0e+00  0.0e+00       0.0e+00    0.0e+00
AvgExpenses         1.1e-04         1.5e-18   7.3e-03     0.0e+00    0.0e+00  0.0e+00  0.0e+00       0.0e+00    0.0e+00
AvgBalance          3.1e-03         1.7e-10   1.1e-03     4.0e-04    0.0e+00  0.0e+00  0.0e+00       0.0e+00    0.0e+00
sex                 8.4e-01         4.0e-13   9.0e-05     9.9e-05    9.1e-01  0.0e+00  0.0e+00       0.0e+00    0.0e+00
Age                 3.6e-04         3.7e-16   1.6e-08     6.3e-09    1.4e-03  1.4e-05  0.0e+00       0.0e+00    0.0e+00
NoInhabitants      2.8e-199        8.3e-211  1.7e-201    9.0e-202   2.6e-197 2.7e-205 2.6e-202       0.0e+00    0.0e+00
UrbanRatio          5.5e-12         8.7e-24   7.0e-16     4.6e-16    4.1e-12  9.0e-15  4.4e-16      2.8e-265    0.0e+00
AverageSalary       5.6e-07         5.1e-18   6.4e-11     6.4e-11    1.4e-06  5.0e-09  5.8e-11      6.1e-238    6.8e-30
Unemployment95      8.4e-03         3.5e-12   8.9e-06     6.3e-06    5.3e-02  1.8e-03  7.7e-07      2.8e-254    4.7e-67
Unemployment96      1.3e-02         1.0e-11   2.1e-05     2.0e-05    6.9e-02  5.8e-03  2.5e-05      1.1e-252    1.6e-67
CrimeRatio95        5.5e-05         3.6e-17   9.0e-09     7.1e-09    1.4e-04  5.5e-07  5.6e-09      7.0e-250    2.6e-32
CrimeRatio96        3.1e-12         1.5e-23   3.2e-16     4.6e-16    4.5e-12  1.6e-13  4.4e-15      1.4e-229    2.5e-45
EnterpreneursRatio  2.1e-03         1.8e-13   1.8e-06     9.6e-07    1.4e-03  8.7e-05  2.8e-06      2.9e-264    4.5e-59
Cardholder          1.0e-03         2.1e-13   4.3e-05     2.5e-05    1.3e-01       NA  5.3e-06      4.8e-205    7.9e-15
                   AverageSalary Unemployment95 Unemployment96 CrimeRatio95 CrimeRatio96 EnterpreneursRatio Cardholder
status                   0.0e+00        0.0e+00        0.0e+00      0.0e+00      0.0e+00             0.0000          0
PermanentOrders          0.0e+00        0.0e+00        0.0e+00      0.0e+00      0.0e+00             0.0000          0
AvgIncome                0.0e+00        0.0e+00        0.0e+00      0.0e+00      0.0e+00             0.0000          0
AvgExpenses              0.0e+00        0.0e+00        0.0e+00      0.0e+00      0.0e+00             0.0000          0
AvgBalance               0.0e+00        0.0e+00        0.0e+00      0.0e+00      0.0e+00             0.0000          0
sex                      0.0e+00        0.0e+00        0.0e+00      0.0e+00      0.0e+00             0.0000          0
Age                      0.0e+00        0.0e+00        0.0e+00      0.0e+00      0.0e+00             0.0000          0
NoInhabitants            0.0e+00        0.0e+00        0.0e+00      0.0e+00      0.0e+00             0.0000          0
UrbanRatio               0.0e+00        0.0e+00        0.0e+00      0.0e+00      0.0e+00             0.0000          0
AverageSalary            0.0e+00        0.0e+00        0.0e+00      0.0e+00      0.0e+00             0.0000          0
Unemployment95           1.2e-71        0.0e+00        0.0e+00      0.0e+00      0.0e+00             0.0000          0
Unemployment96           4.2e-59        1.3e-06        0.0e+00      0.0e+00      0.0e+00             0.0000          0
CrimeRatio95             7.8e-25        1.9e-72        1.2e-55      0.0e+00      0.0e+00             0.0000          0
CrimeRatio96             1.7e-27        5.7e-63        2.2e-58      2.5e-19      0.0e+00             0.0000          0
EnterpreneursRatio       9.9e-46        7.1e-51        2.3e-51      1.9e-77      1.3e-63             0.0000          0
Cardholder               1.3e-08        2.7e-03        1.6e-02      5.3e-07      1.6e-13             0.0018          0
```

Figure 5-9: Correlation tests overview of the optimised dataset (display_res_azure_opt)

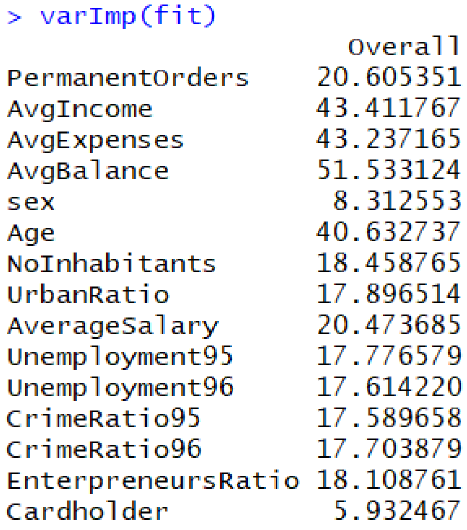
After computing and testing the correlation of the possible pairs from the original pre-processed dataset, the following R-code snippet supports the research purpose to build a cost-sensitive data model.



```
#Creates an optimized classification model using a random forest algorithms
fit <- randomForest(display_res_azure_opt$status ~ ., data=display_res_azure_opt)
# import caret library using varImp to see import variables in the fit model
library(caret)
varImp(fit)
varImpPlot(fit,type=2)
```

Figure 5-10: R-code snippet for building a cost-sensitive data model - credit scoring

The following figure 5-11 shows a table of the varImp() function results for the underlying processed dataset. The top-most variable overall is the most significant such as ‘AvgBalance’, ‘AvgIncome’, ‘AvgExpenses’ and ‘Age’. Accordingly, the higher the values, the higher the variable importance. The ranking of the variables is based on the pre-processed optimised dataset and the mean decrease is measured by the Gini Index. I have selected this measure of variable importance as the research does not deal with a time series dataset to predict suitable credit scorings for banking clients. However, there are no fixed concept criteria to measure the variables’ importance. The variables with the highest importance score overall are those that probably provide the best prediction and contribution to the credit scoring model. The predictive outcomes will be explained in the next section.



```
> varImp(fit)
                      Overall
PermanentOrders     20.605351
AvgIncome           43.411767
AvgExpenses         43.237165
AvgBalance          51.533124
sex                  8.312553
Age                 40.632737
NoInhabitants       18.458765
UrbanRatio          17.896514
AverageSalary       20.473685
Unemployment95      17.776579
Unemployment96      17.614220
CrimeRatio95        17.589658
CrimeRatio96        17.703879
EnterpreneursRatio  18.108761
Cardholder           5.932467
```

Figure 5-11: Overview of the varImp () results for the optimised dataset - 1st research project

The varImp() plot illustrates the important variables in our random forest model. It emerges that there are many variables such as 'PermanentOrders', 'AverageSalary', 'NoInhabitants', 'EnterpreneursRatio', 'UrbanRatio', 'Unemployment95', 'CrimeRatio96', 'Unemployment96', 'CrimeRatio95', 'Sex' and 'Cardholder' in the model that are less important.

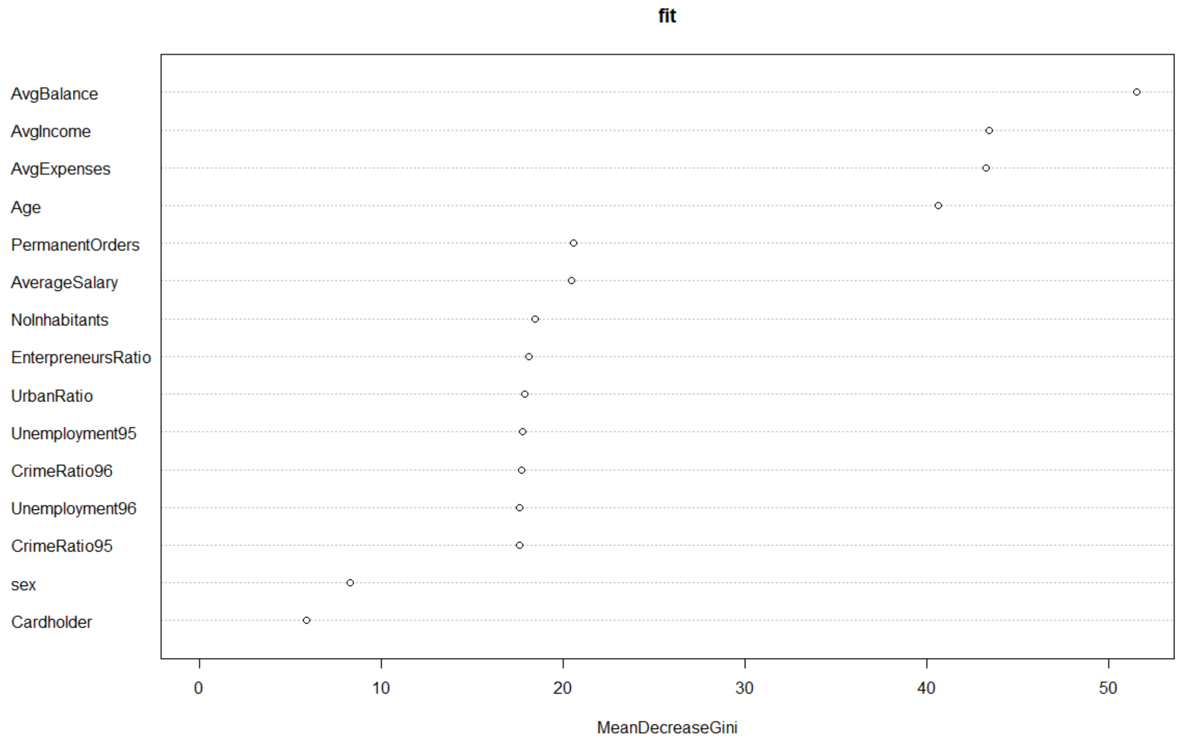
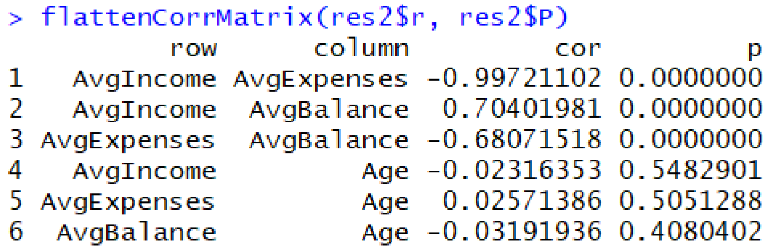


Figure 5-12: Visualisation of significant variables within the data model for the 1st research project

The graph above shows by how much the MeanDecreaseGini score increases if a variable is assigned values by random permutation. If we randomly permute the 'AvgBalance', the MeanDecreaseGini will increase by randomly 50% on average. This observation makes sense, since the overall average balance of a bank customer account has increased in recent years. The same observations can be noticed by the remaining variable such as ‘AvgIncome’, ‘AvgExpenses’ and ‘Age’. The descriptive results highlight that the top variable from our random forest model will increase the prediction power of the credit scoring model, and if we reduce one of the bottom variables such as ‘Sex’ or ‘Cardholder’, there might not be a huge impact on the prediction power of the predictive model. Regarding the prediction results – which will be dealt within the next section – it is also relevant to validate the descriptive results and their assumptions in detail by developing different forecast models for the optimised dataset concerning various subsets of mixed significant variables.

Finally, the conducted research computes a flatten correlation matrix with significance levels (p-value) for all possible pairs according to the significant variables of the optimised dataset using the function rcorr() from the R-package “Hmisc” and a self-developed function to display a flattened correlation matrix. The corresponding R-script is given in appendix E.1. The table displayed in figure 5-13 below shows the causal relationships between the selected variables. The correlation coefficient (-0.997) between ‘AvgIncome’ and ‘AvgExpenses’ shows that the increase in ‘AvgIncome’ results in a decrease of ‘AvgExpenses’. There is also a negative relationship (-0.68) between ‘AvgExpenses’ and ‘AvgBalance’, which indicates that an increase of ‘AvgExpenses’ will turn into a reduced ‘AvgBalance’. The strong positive relationship (0.704) between ‘AvgIncome’ and ‘AvgBalance’ indicates that the higher the income, the higher the balance. Regarding the p-values (p=0.000) of these relationships, there is no evidence about the significance of the associations between the analysed variables.



```
> flattenCorrMatrix(res2$r, res2$P)
          row      column         cor         p
1   AvgIncome AvgExpenses -0.99721102 0.0000000
2   AvgIncome  AvgBalance  0.70401981 0.0000000
3 AvgExpenses  AvgBalance -0.68071518 0.0000000
4   AvgIncome         Age -0.02316353 0.5482901
5 AvgExpenses         Age  0.02571386 0.5051288
6  AvgBalance         Age -0.03191936 0.4080402
```

Figure 5-13: Flattened correlation matrix of the top-most important variables for the optimised dataset - 1st research project

The following descriptive results include the visualised results of the correlation matrix by drawing a performance analytics chart highlighting the most strongly correlated variables and their relationships in the optimised dataset. Accordingly, I have displayed a chart of the correlation matrix by using the function chart.correlation() from the R-package “PerformanceAnalytics”.

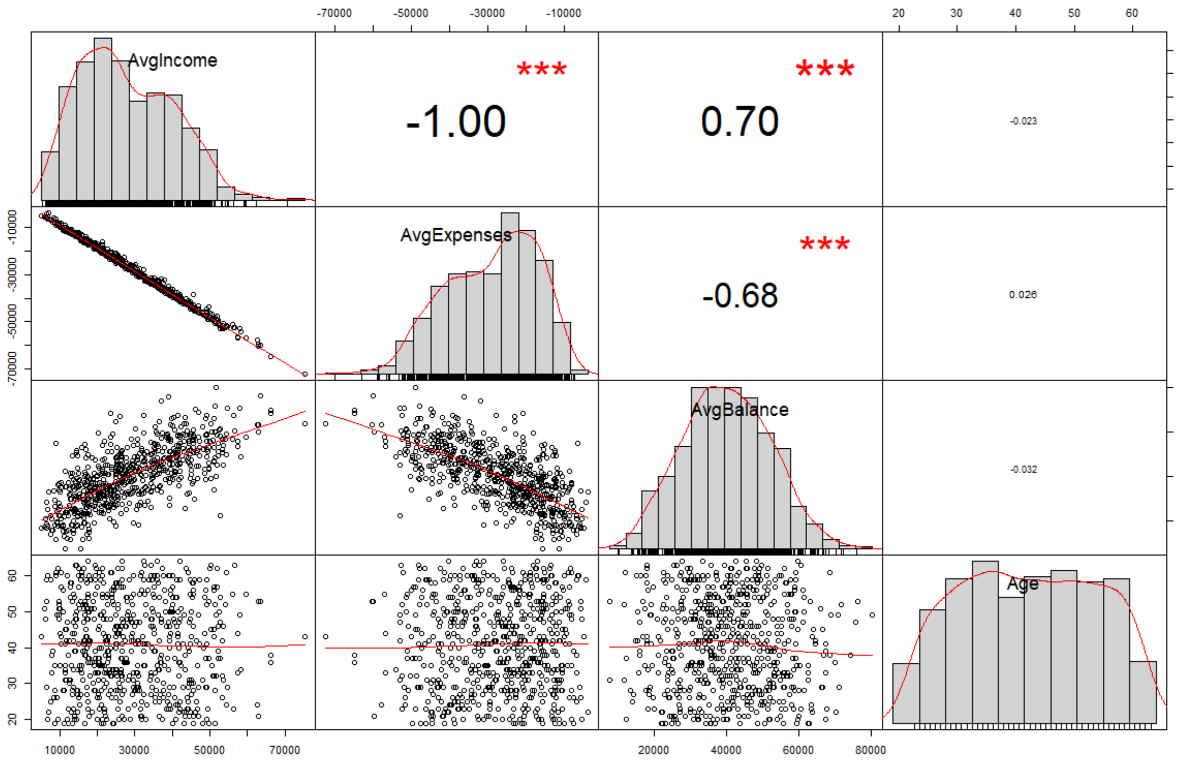


Figure 5-14: Performance analytics chart of the correlation matrix for the varImp variables – 1st research project

The diagonal of the plot above shows the distribution of every significant variable, whereby ‘AvgIncome’ is skewed to the left, ‘AvgExpenses’ is skewed to the right and ‘AvgBalance’ – which can also be summarised from both datasets – is slightly skewed to the left, while ‘Age’ is uniformly distributed around the customer base. On the top of the diagonal, the value of the correlation plus the significance level is displayed as stars. The associations between ‘AvgBalance’ vs. ‘AvgIncome’ (0.70) and ‘AvgBalance’ vs. ‘AvgExpenses’ (-0.68) are rated as highly significant with three stars (p-values of 0.001).

5.1.2. Analysis results of the cross-selling dataset

The section describes the overall evaluations and findings derived by analysing the cross-selling dataset. In the pre-processing phase, different tables were joined into single cleaned temporary tables before different target attributes could be created such as ‘sex’ and ‘age’. Independent attributes mainly comprised socio-demographics derived from the client’s district, such as the number of inhabitants, urban ratio, crime ratio, the entrepreneur’s ratio or information about the unemployment rate. Regarding the various processed intermediate steps, the calculated totals from the temporary ‘CashflowsAggregated’ table help to extract the frequency and the monetary characteristics from a transactional payment history such as average salary, average income, average balance and average expenses. For cross-selling, labelled as creditcard_azure.csv, the research work only considered 827 normalised clients.

However, van der Putten (1999) ascertained in his descriptive analysis that “credit card holders are active clients that take high cash withdrawals (factor 1.9 higher), take cash withdrawals more often (1.8) and have a higher balance (average balance 1.7, last balance 1.6)”. Credit card holders more commonly have loans (1.7) but are very good at paying them back (factor 5.1; probably a side effect of a strict credit card admission policy).

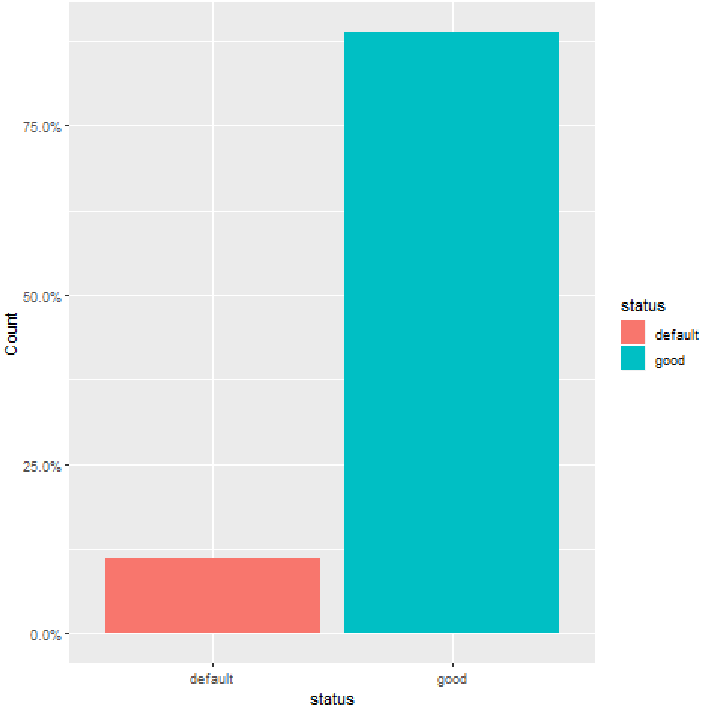


Figure 5-15: Distribution of credit cards owner vs. loan status

The illustration above shows the distribution of credit card owners against their loan status. It emerged that the majority of credit card owners present a good loan status. The imbalanced dataset must be harmonised for further analysis to ensure the best and most consistent predictive results.

The following figure 5-16 illustrates an overview of the MS Azure Mini Map implemented to realise the research experiments at a scale that is constructed in the second field of research. The mini map provides a high-level overview of the research setup, which is very powerful and useful for the research understanding.

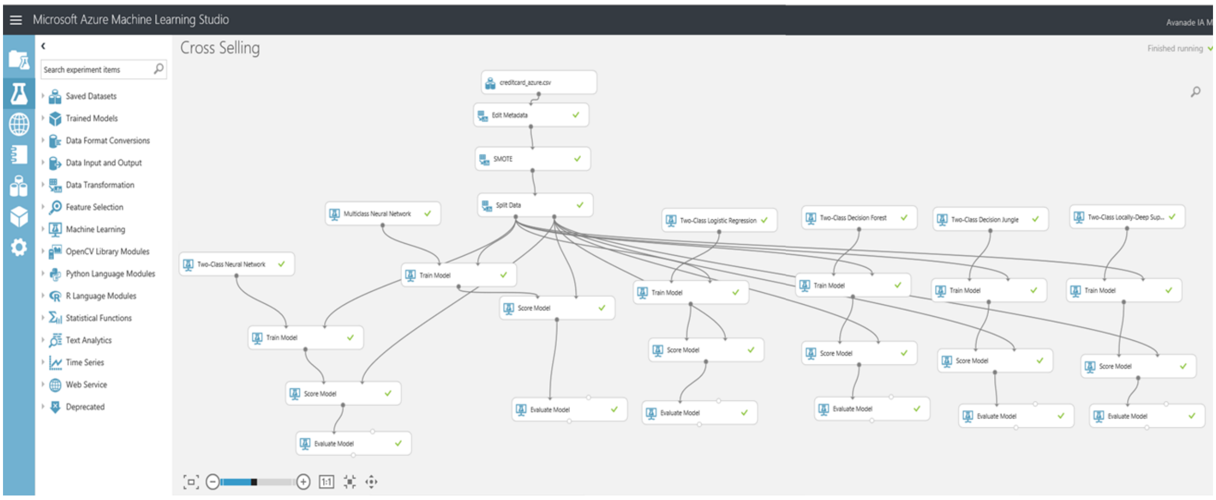


Figure 5-16: MS Azure's Mini Map overview for processed dataset (creditcard_azure.csv)

The first step was to read the pre-processed dataset “creditcard_azure.csv” and include only relevant columns such as ‘Age’, ‘AverageSalary’, ‘AvgIncome‘, ‘AvgExpenses’, ‘AvgBalance’ and ‘Cardholder‘ from the dataset. The next step was to flag the column ‘Cardholder’ as categorical and ensure that the data type remained unchanged during the processing. In the next step, I have applied the SMOTE module in Azure ML Studio due to the given imbalanced dataset (see appendix C.2). The module returns an advanced “creditcard_azure” dataset that contains the original samples, plus an additional number of synthetic minority samples with a percentage of up to 100%. I have connected the original dataset with the SMOTE module, because the targeted category ‘Cardholder’ has a rare population of credit card holders (25%). The under-represented class increased with cases in the “creditcard_azure” dataset in a balanced way. The outcome is a more general sample by increasing the percentage of only minority credit card owner cases using multiples of 100 and a number of nearest neighbours of 1. However, the applied SMOTE function does not guarantee building more accurate models, but it increases the number of low incidence examples in the cross-selling dataset using synthetic minority oversampling. The subsequent connected “Split Data” module split the data into test (20%) and training (80%) data. Finally, the last modules of the mini map show the connecting (intermediate) steps such as various applied mining algorithms for the prepared dataset with the aim to train, score and evaluate the predictive models. Detailed results will be explained in the next section.

Appendix C.2 comprises profound visual descriptions (comprising boxplot, histogram and statistics) for the cross-selling dataset “creditcard_azure.csv”. The research again initially considered interesting variables by using the visualisation functions of MS Azure ML and thereafter I sought to gain more in-depth insights while comparing exploratory variables of the pre-processed dataset. The main results of the description phase can be summarised as follows:

Figure C-21 “EDA - frequency” depicts the distribution of the amount along the variable ‘frequency’. As can be seen in the histogram, the majority of processed transactions are assigned to ‘monthly issuance’ with a percentage of over 82%, while around 13% are assigned to ‘weekly issuance’ and 5% of the entire processed transactions are assigned to ‘issuance after transactions’, which indicates late payments. Figure C-11 “EDA - sex” and figure C-23 “EDA - age” reveals that the account owner is female or male with a probability of at least 50% and with an average age of 40.1, which is evenly distributed over the customer base, assuming that Berka has provided a representative dataset. Figure C-24 “EDA - no inhabitants” illustrates that two-thirds of the account owner’s districts are associated with a district whose number of inhabitants is greater than the average of 266,252. For instance, this circumstance can be an influence factor when calculating cross-selling as a credit card promotion can be concentrated on chosen districts. Further exploratory analysis results of the socio-demographic variables such as given in figure C-25 “EDA - urban ratio”, figure C-27 “EDA - unemployment95”, figure C-28 “EDA - unemployment96”, figure C-29 “EDA - crimeratio95”, figure C-30 “EDA - crimeratio96” and figure C-31 “EDA - entrepreneur’s ratio” can also have an indirect impact by assessing the cross-selling candidates.

With the histogram for the variable ‘average salary’ (mean = 9,463), figure C-26 “EDA - average salary” shows that many banking accounts are heavily distributed to the left as more than 55% receive the same base salary volume. In conclusion, every second bank account is affected when customising the credit card promotion. Regarding the displayed histogram, four types of cross-selling offerings can be provided, whose different product offerings may vary on the underlying salary bands (1: 8.000-8.600; 2: 9.000-9.400; 3: 9.900 - 11.000; 4: 12.000-12.500). As can be seen in figure C-32 “EDA - average income”, the distribution of the values for the variable ‘average income’ is inclined to the left. From the statistical analysis (mean = 29,135) and the displayed histogram, it can be deducted that most banking clients have a lower average income compared with the entire customer base. The exploratory data analysis for the variable ‘average expenses’ in figure C-33 “EDA - average expenses” shows that the clients have on an average less expenditures (mean is -27669) than income. This can also be concluded from the histogram, which reflects that the distributed values are inclined to the right. Beyond this, the exploratory data analysis for the variable ‘average balance’ in figure C-34 “EDA - average balance” highlights that the preliminary analyses of the banking accounts showed that the distribution of the balances are slightly skewed to the left with many banking accounts. A perfect uniformly distribution of the average balance will be a moderate sign for the good healthiness of a bank account. This characteristic generally has a positive impact on the cross-selling outcome and increases the opportunity for cross-selling activities.

Taken together, the descriptive and diagnostic analysis, interesting insights and new associated knowledge can be gathered from the pre-processed dataset. The following building block based on advanced data exploration methods helps the conducted research experiments in generating more valuable customer insights. Therefore, interesting variables will be compared with each other and especially socio-demographic variables. The main results can be summarised as follows:

Figure 5-17 “EDA - average salary compared with various variable” illustrates that there is a concentration of crime between 0.01 and 0.04 within the minimum salary range of 8,100 – 9,200. This observable pattern can be highlighted in the scatterplot as a strong positive relationship. Assuming that the concentration of crime ratio indicates a high level of crime, the higher the average salary, the lower the crime. The frequency of the crime says nothing about the quality and intensity of the crime itself. Note that I have only observed a tendency in the relationship between the two variables. Regarding the urban ratio, it can be deducted from the scatterplot that there is a remarkable concentration of specific average salary bands. Bank customers with a low average salary are living in average urban areas (45 to 65). The observable pattern can be summarised that the higher the average salary of the client, the higher the urban ratio.

For instance, special cross-selling offerings can be targeted best – for clients with a high average salary – only in urban conurbations.

Average salary	to crime ratio96	to urban ratio

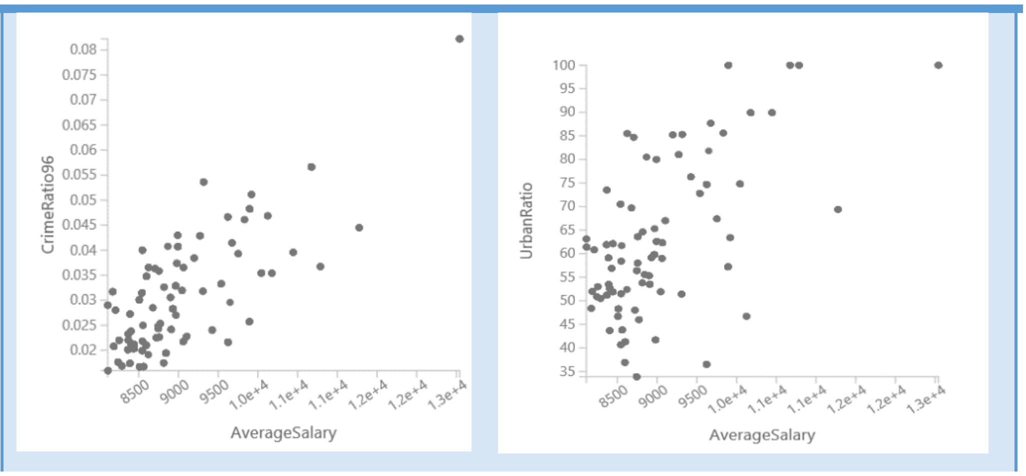
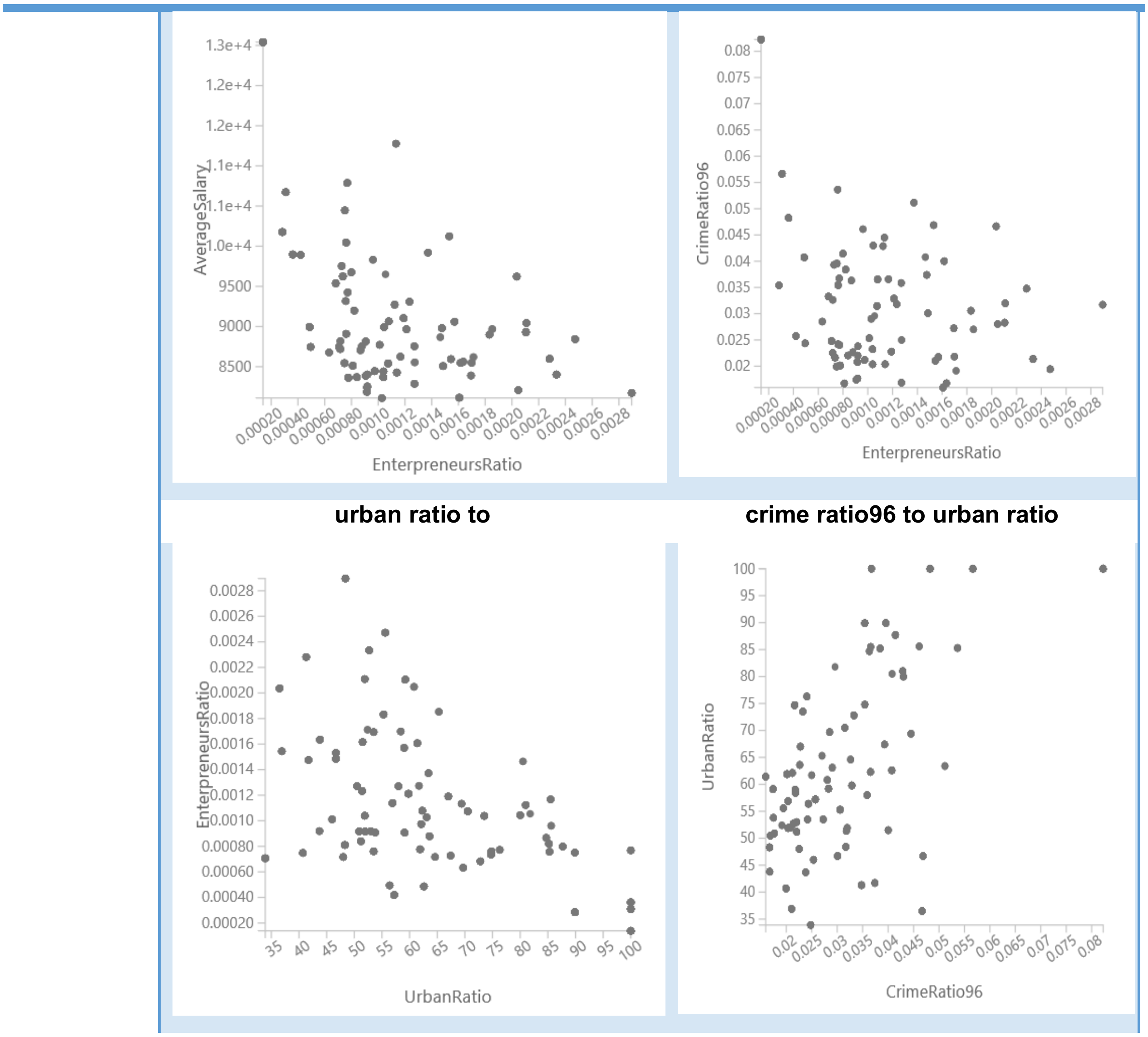


Figure 5-17: Exploratory data analysis - average salary compared with various variable

Figure 5-18 “EDA - entrepreneur’s ratio compared with various variables” describes the relationships of the entrepreneur’s ratio with other socio-demographic variables. When comparing the average salary of every bank customers with the entrepreneur’s ratio, it can be observed that the number of entrepreneurs in the lower band of average salary is higher than in the upper band. The outliers within the scatterplot should be ignored to identify any remarkable patterns. Regarding the crime ratio96, it can be highlighted that there is an obvious cluster within the range of 0.02-0.025 around the entrepreneur’s ratio of 0.0008. The second smaller cluster is centred around the range of 0.035-0.04. The scatterplot for the urban ratio compared with the entrepreneur’s ratio does not initially highlight any significant association between the two variables. Both variables will probably not have a high weighting in the data modelling process. However, the scatterplot for the variables ‘crime ratio96’ and ‘entrepreneur’s ratio’ brings truly interesting discoveries to the fore. A notable pattern is the positive relationship between the two variables. The crime ratio96 increases with the urban ratio in a balanced way. Therefore, a rapid urbanisation is associated with the growth of crimes. Both variables are expected to have a significant impact on the data modelling results. There is a slight concentration of crimes (range of 0-0.03) in average urban areas (range of 45-65). Bank product offerings may be tailored to suit the different requirements of this special clientele to avoid unnecessary contractual risks, whereby clients who live in urbanised areas might be held liable when the credit card is stolen.

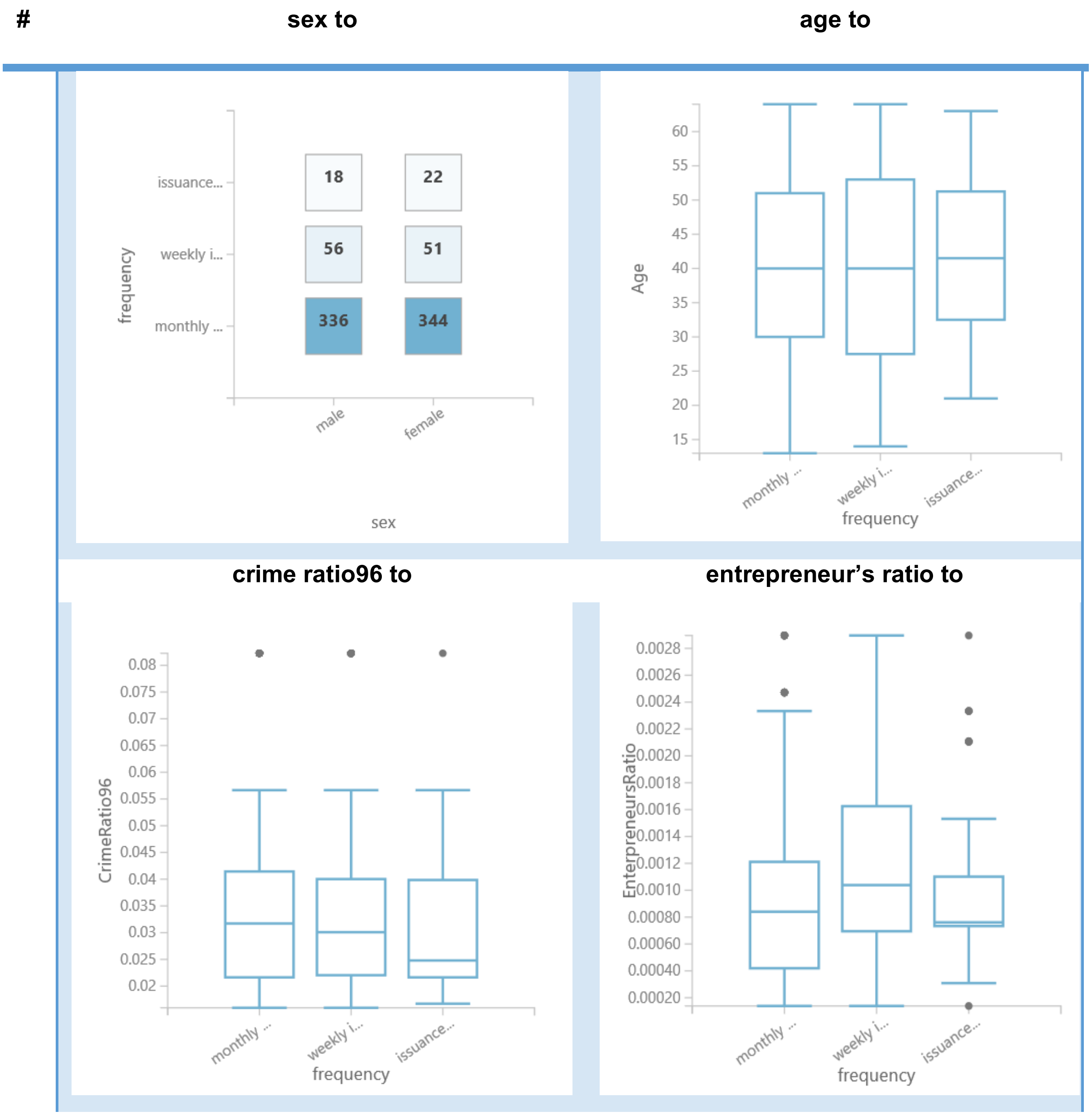
Entrepreneur's ratio **to average salary** **to crime ratio96**



urban ratio to **crime ratio96 to urban ratio**

Figure 5-18: Exploratory data analysis - entrepreneur's ratio compared with various variables

Figure 5-19 "EDA - various variables compared with frequency" summarises the relationships of different variables, always compared with the variable 'frequency'. Regarding payment transactional patterns, the crosstab for the variable 'sex' compared with 'frequency' shows that the peer group of males and females process roughly the same number of different transaction types (monthly issuance, weekly issuance or issuance after transaction). The multi-boxplots for 'age compared with frequency' illustrate that weekly transactions are processed from clients between the ages of 29 and 53 years. Banking clients with an age limit vary between 33 to 51 years have a delay in their payments. The majority of transactional payments (male: 336, female: 344) are issued on a monthly basis from clients who are 30 to 52 years old. The multi-boxplot for “crime ratio96 compared with frequency” highlights that late transactions are associated with specific crime ratio96 values. Finally, the multi-boxplot for “entrepreneur’s ratio compared with frequency” illustrates that the majority of processed transactions on a monthly or weekly basis are associated with the entrepreneur’s ratio between 0.0008 and 0.001.



#

sex to

age to

crime ratio96 to

entrepreneur’s ratio to

Figure 5-19: Exploratory data analysis - various variables compared with frequency

Figure 5-20 “EDA - unemployment95 compared with various variables” depicts the relationships between the socio-demographic variable “unemployment95” and other variables of the entire pre-processed dataset. The multi-boxplot “unemployment95 compared with age" indicates that the higher the value of 'unemployment95', the older the customer. Moreover, the data exploration for other correlations with the help of the remaining multi-boxplots does not show any reasonable outliers or clusterings. From a statistical perspective, it can be noted that the values of the variables 'average balance', 'average expenses' and 'average income' are uniformly distributed over different values of the variable 'unemployment95'.

Unemployment95 **to age** **To average balance**

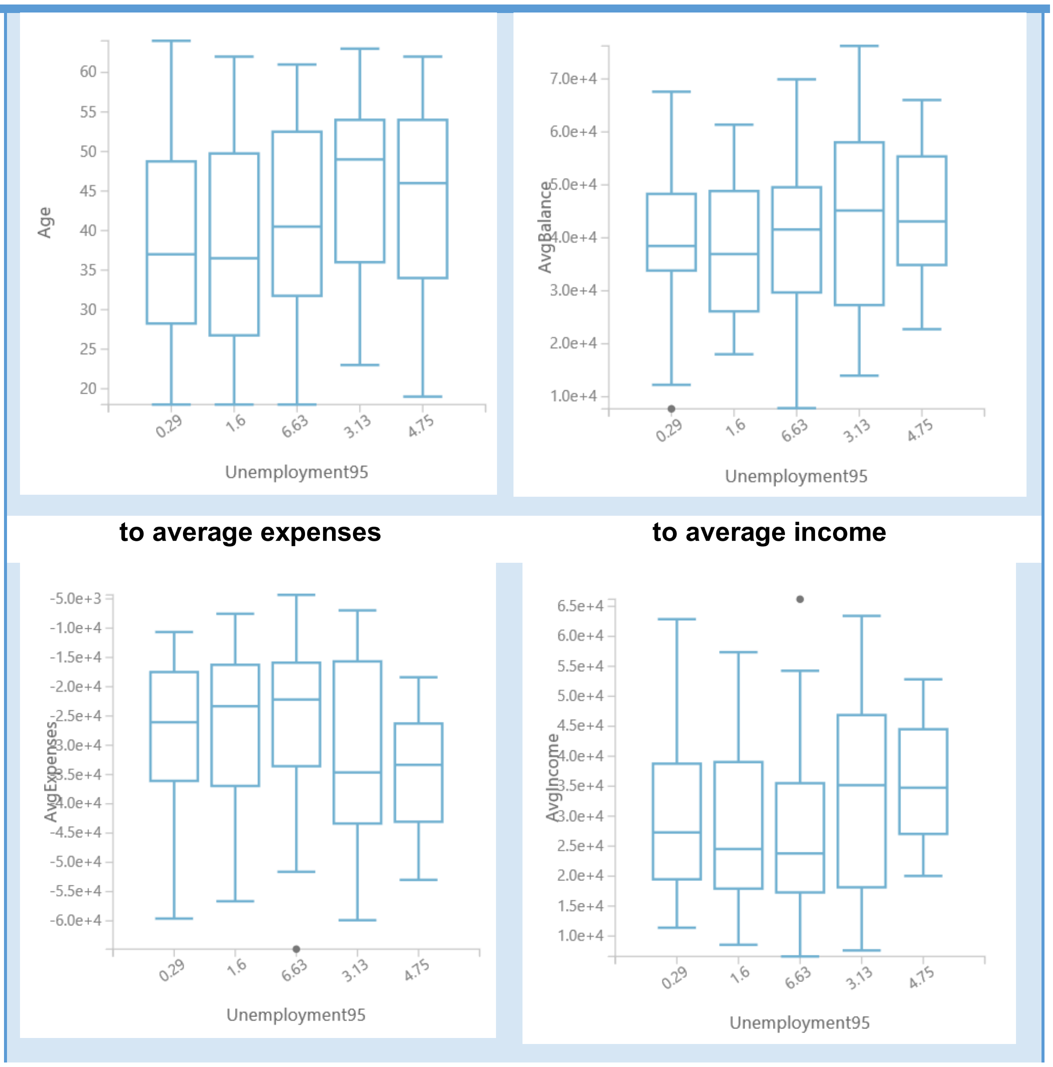
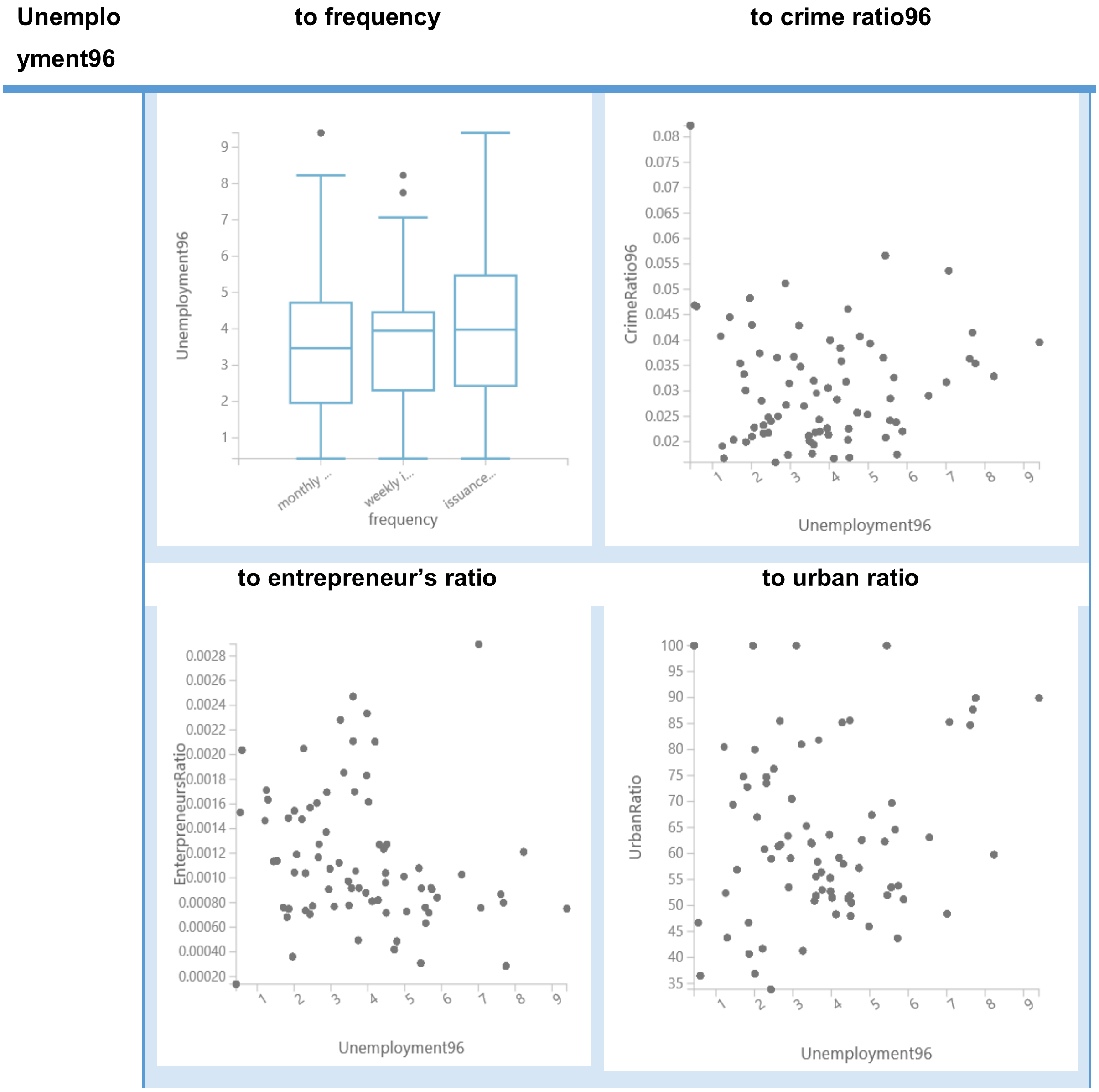

to average expenses **to average income**

Figure 5-20: Exploratory data analysis - unemployment95 compared with various variables

The following figure 5-21 "EDA - unemployment96 compared with various variables" describes results of the exploratory data analysis. Regarding the scatterplots for the urban ratio and entrepreneur's ratio, negative relationships against the unemployment95 value can be noticed. An increase in one of these variables results in a decrease of the unemployment95 values. The cross-selling activities should also be tailored based on these remarkable patterns. Clients who are living in highlyfrequented urban areas or clients who have a strong affinity with entrepreneurship will have a low probability of unemployment. This fact can be included in the customisation of dedicated product offering. For instance, those clients can receive additional discount in case of credit card promotions as their credit default risk indicator might be low.



Unemployment96 **to frequency** **to crime ratio96**

to entrepreneur's ratio **to urban ratio**

Figure 5-21: Exploratory data analysis - unemployment96 compared with various variables

The following paragraphs deal with the descriptive and diagnostic analysis results concerning the evaluation of the model-building process for optimising the predictive modelling results. Therefore, detailed predictive results will be presented in the next section. Figure 5-22 below illustrates an overview of the created MS Azure Mini Map implemented to realise the research experiments at scale for the optimised pre-processed dataset, which is constructed as the concluding part for the second field of research. The mini map provides a high-level overview of the research design, which is also aligned with the preceding research questions. The research setup ensures that the evaluations are independent and objective as the research design corresponds with the upper MS Azure Mini Map diagram excluding minor data-related adjustments to establish the comparability of the research results.

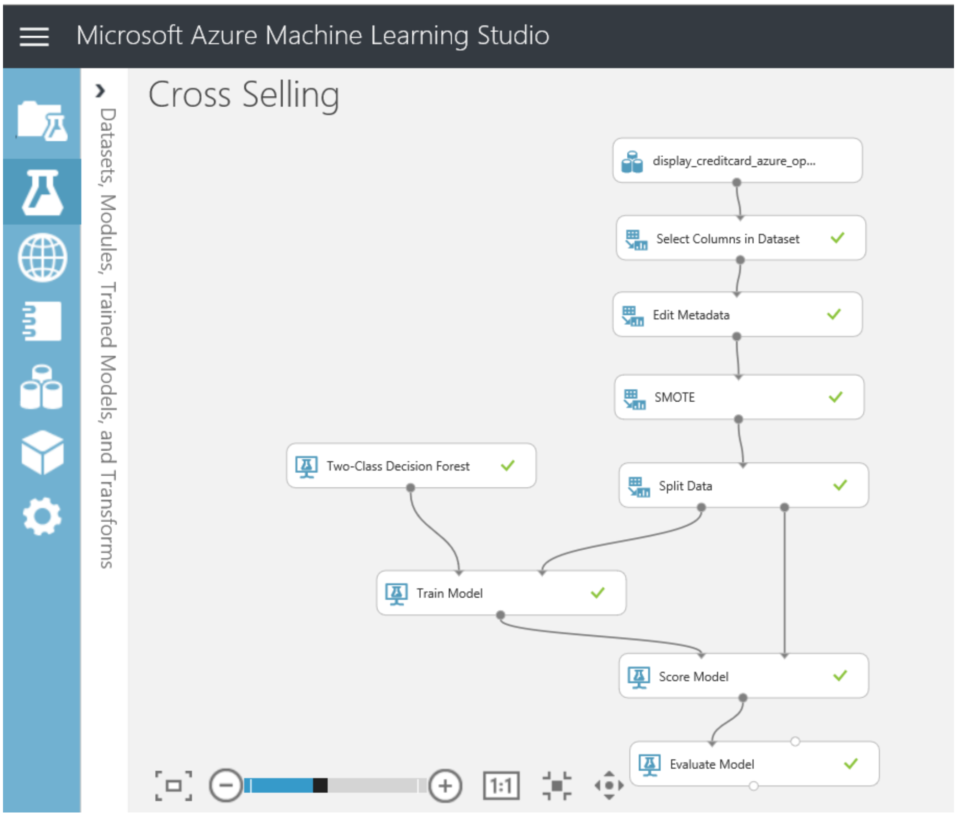

Microsoft Azure Machine Learning Studio
Cross Selling
Datasets, Modules, Trained Models, and Transforms
display_creditcard_azure_op...
Select Columns in Dataset
Edit Metadata
SMOTE
Split Data
Two-Class Decision Forest
Train Model
Score Model
Evaluate Model
1:1

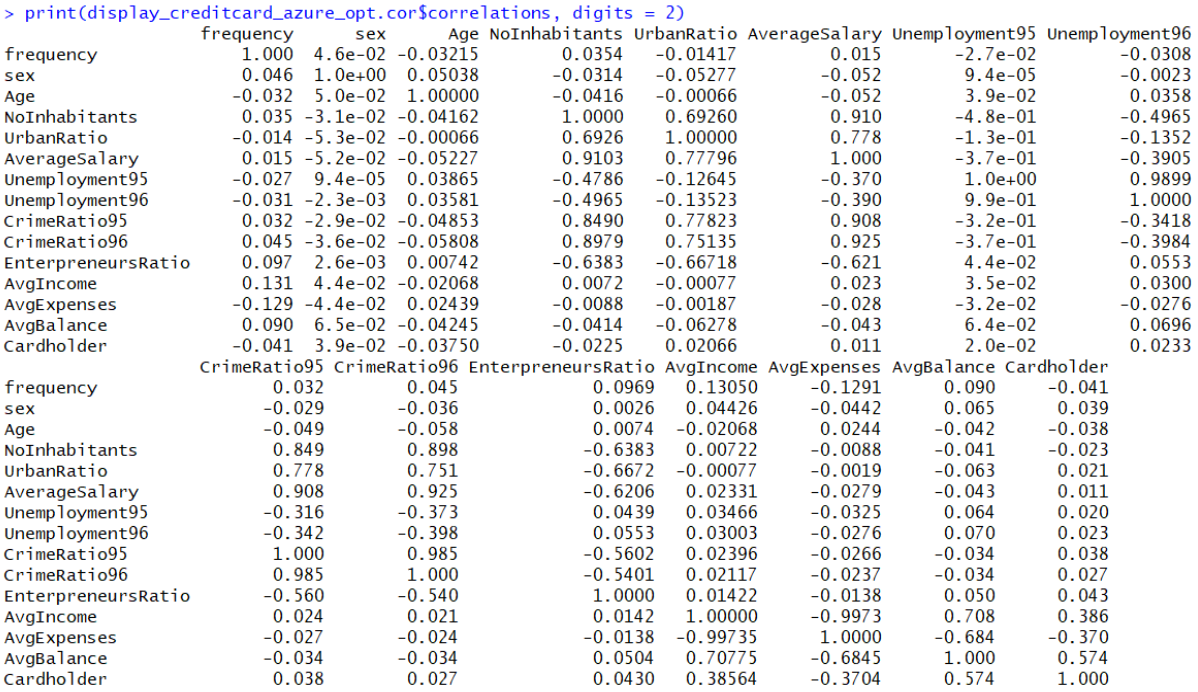
Figure 5-22: MS Azure's Mini Map overview for optimised dataset (display_creditcard_azure_opt.csv)

The first step was to read the pre-processed dataset "display_creditcard_azure_opt.csv" and include only relevant attributes such as 'Age', 'AverageSalary', 'AvgIncome', 'AvgExpenses', 'AvgBalance' and 'Cardholder' from the dataset. The selection process of the above variables is based on the statistical calculations when estimating the variable importance for the data model using the varImp() function from the R-package "caret". Detailed results will be explained later in this section. The next step was to flag the column 'Cardholder' as categorical using the connected "Edit Metadata" module in MS Azure, which ensures that the values (categorical) and the data types (unchanged) in the dataset are not altered after the previous operation. The module changes the 'Cardholder' field to a class label (target variable) as these are the values that the research experiment wants to predict. Note that the data values will not change; rather, the modules only ensure that the applied supervised learning algorithms handle the data correctly. Next up in the mini map is the applied SMOTE module in Azure ML Studio due to the given imbalanced dataset. The module returns a “display_creditcard_azure_opt” dataset, which contains the original samples, plus an additional number of synthetic minority samples with a percentage of up to 100%. The under-represented class of non-credit card holder credits (201 out of 818 banking clients) increases with cases in the “display_creditcard_azure_opt” dataset in a balanced way using multiples of 100 and a number of nearest neighbours of 1. The more general sample supports the objectivity of the evaluation results. The module “Split Data” again splits the rows of the prepared dataset into two distinct datasets with a fraction of 20% test data and 80% training data. The last steps of the mini map overview displayed in figure 5-22 above show the connecting (intermediate) steps to the applied two-class decision forest algorithms from the MS Azure ML library for the optimised dataset with the aim to train, score and evaluate the predictive models. Detailed predictive results will be explained in the next section.

As mentioned in the previous paragraph, the last sections will describe the descriptive results while extracting relevant variables from the random forest model using varImp() function from the R-package “caret” to create a cost-sensitive classification model. The function will help to assess the prediction power of the chosen random forest model. In the first step, the conducted research computes a correlation matrix using the R function cor() on the optimised dataset “display_creditcard_azure_opt” to evaluate the linear dependence between two variables. The Pearson correlation coefficient measures every possible pair of variables and correlation coefficients are shown in figure 5-23 below.

The correlation matrix below shows that the Pearson correlation between the variables ‘AvgIncome’ and ‘AvgExpenses’ is about -0.997, which indicates that there is a perfect negative relationship (value close to -1) between the variables as ‘AvgIncome’ increases and ‘AvgExpenses’ decreases. The Pearson correlation coefficient of 0.704 between ‘AvgIncome’ and ‘AvgBalance’ indicates a moderate positive relationship. Therefore, an increasing average income has a positive effect on the average balance. Another observable pattern is the correlation between the variables ‘AvgExpenses’ and ‘AvgBalance’. The coefficient of -0.684 indicates a negative (moderate) relationship, which means that a reduced average balance will result in an increased average expense. The correlation results for the variable ‘Age’ are not considerable, even if the overall varImp() results provide relatively encouraging descriptive results.

However, the conducted research does not emphasise the low Pearson correlation coefficients given in the results matrix due to the research design, in which initially significant variables were considered and receive higher priority along the research analysis. This approach does not mean that no relationship exists between the remaining variables since the variables may ultimately have a non-linear relationship or will be out of scope, like the analysis between the demographic variables (i.e. ‘NoInhabitants’ vs. ‘UrbanRatio’ results in 0.6926, etc.). The outcomes of the variable ‘Sex’, ‘Age’ or ‘Unemployment95’ add much to the underlying research proceedings since most of the correlations are close to 0, and thus have almost no linear relationship with the remaining variables, although the varImp() function shows a certain relevance for the modelling building process. Details will be shown in the following paragraphs.

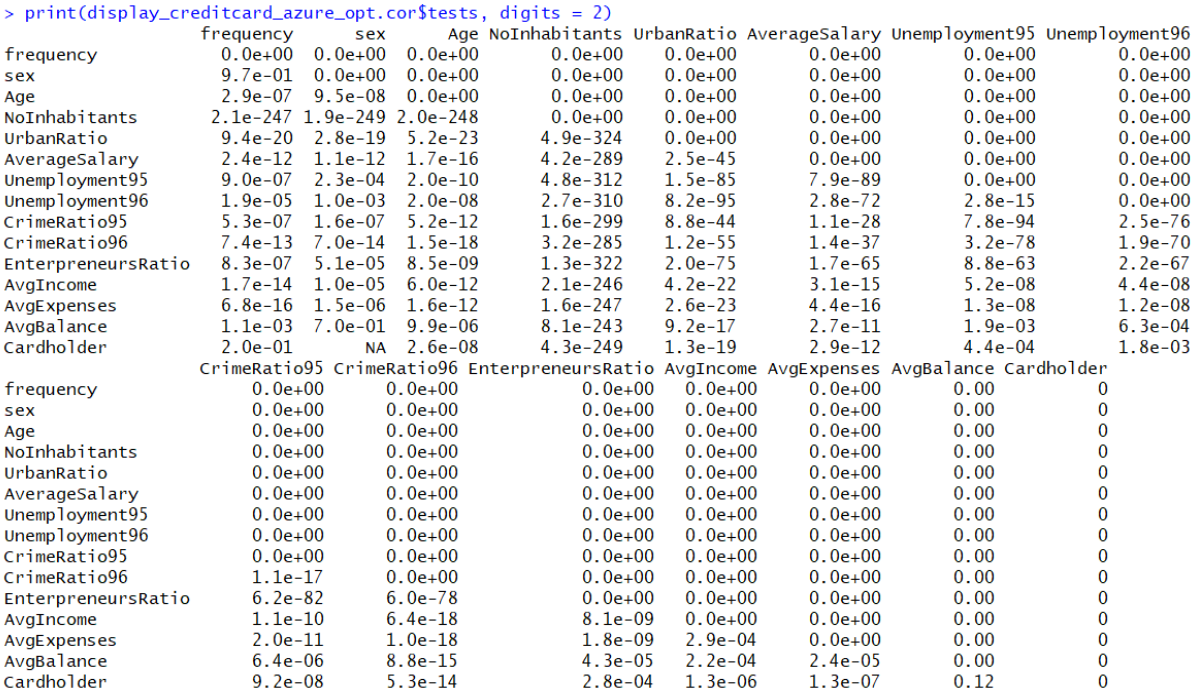


```
> print(display_creditcard_azure_opt.cor$correlations, digits = 2)
                   frequency      sex      Age NoInhabitants UrbanRatio AverageSalary Unemployment95 Unemployment96
frequency              1.000  4.6e-02 -0.03215        0.0354   -0.01417         0.015       -2.7e-02        -0.0308
sex                    0.046  1.0e+00  0.05038       -0.0314   -0.05277        -0.052        9.4e-05        -0.0023
Age                   -0.032  5.0e-02  1.00000       -0.0416   -0.00066        -0.052        3.9e-02         0.0358
NoInhabitants          0.035 -3.1e-02 -0.04162        1.0000    0.69260         0.910       -4.8e-01        -0.4965
UrbanRatio            -0.014 -5.3e-02 -0.00066        0.6926    1.00000         0.778       -1.3e-01        -0.1352
AverageSalary          0.015 -5.2e-02 -0.05227        0.9103    0.77796         1.000       -3.7e-01        -0.3905
Unemployment95        -0.027  9.4e-05  0.03865       -0.4786   -0.12645        -0.370        1.0e+00         0.9899
Unemployment96        -0.031 -2.3e-03  0.03581       -0.4965   -0.13523        -0.390        9.9e-01         1.0000
CrimeRatio95           0.032 -2.9e-02 -0.04853        0.8490    0.77823         0.908       -3.2e-01        -0.3418
CrimeRatio96           0.045 -3.6e-02 -0.05808        0.8979    0.75135         0.925       -3.7e-01        -0.3984
EnterpreneursRatio     0.097  2.6e-03  0.00742       -0.6383   -0.66718        -0.621        4.4e-02         0.0553
AvgIncome              0.131  4.4e-02 -0.02068        0.0072   -0.00077         0.023        3.5e-02         0.0300
AvgExpenses           -0.129 -4.4e-02  0.02439       -0.0088   -0.00187        -0.028       -3.2e-02        -0.0276
AvgBalance             0.090  6.5e-02 -0.04245       -0.0414   -0.06278        -0.043        6.4e-02         0.0696
Cardholder            -0.041  3.9e-02 -0.03750       -0.0225    0.02066         0.011        2.0e-02         0.0233
                   CrimeRatio95 CrimeRatio96 EnterpreneursRatio AvgIncome AvgExpenses AvgBalance Cardholder
frequency                 0.032        0.045             0.0969   0.13050     -0.1291      0.090     -0.041
sex                      -0.029       -0.036             0.0026   0.04426     -0.0442      0.065      0.039
Age                      -0.049       -0.058             0.0074  -0.02068      0.0244     -0.042     -0.038
NoInhabitants             0.849        0.898            -0.6383   0.00722     -0.0088     -0.041     -0.023
UrbanRatio                0.778        0.751            -0.6672  -0.00077     -0.0019     -0.063      0.021
AverageSalary             0.908        0.925            -0.6206   0.02331     -0.0279     -0.043      0.011
Unemployment95           -0.316       -0.373             0.0439   0.03466     -0.0325      0.064      0.020
Unemployment96           -0.342       -0.398             0.0553   0.03003     -0.0276      0.070      0.023
CrimeRatio95              1.000        0.985            -0.5602   0.02396     -0.0266     -0.034      0.038
CrimeRatio96              0.985        1.000            -0.5401   0.02117     -0.0237     -0.034      0.027
EnterpreneursRatio       -0.560       -0.540             1.0000   0.01422     -0.0138      0.050      0.043
AvgIncome                 0.024        0.021             0.0142   1.00000     -0.9973      0.708      0.386
AvgExpenses              -0.027       -0.024            -0.0138  -0.99735      1.0000     -0.684     -0.370
AvgBalance               -0.034       -0.034             0.0504   0.70775     -0.6845      1.000      0.574
Cardholder                0.038        0.027             0.0430   0.38564     -0.3704      0.574      1.000
```

Figure 5-23: Correlation overview of the optimised dataset (display_creditcard_azure_opt)

The other extreme is the correlation results of the variable ‘AverageSalary’, which shows a different strength of linear relationships with the other variables but ultimately indicates less importance than the variable ‘frequency’ when applying the varImp() function. A strong uphill linear relationship is given between ‘AverageSalary’ and ‘NoInhabitants’ (0.91), ‘AverageSalary’ and ‘UrbanRatio’ (0.77), ‘AverageSalary’ and ‘CrimeRatio95’ (0.90), ‘AverageSalary’ and ‘CrimeRatio96’ (0.92). A weak uphill linear relationship is given between ‘AverageSalary’ and ‘Unemployment95’ (-0.37) as well as ‘AverageSalary’ and ‘Unemployment96’ (-0.39). The overall diagnostic check by computing the correlation matrix also shows that the predictive results of the applied two-class logistic regression algorithm is reliable since there are not a high number of correlations given in the matrix below. Variables such as ‘Unemployment96’ and ‘Sex’ were eventually excluded in the processed cost-sensitive data model since the constructed research experiments have shown that the inclusion of the socio-demographic data does not contribute to an advanced model.

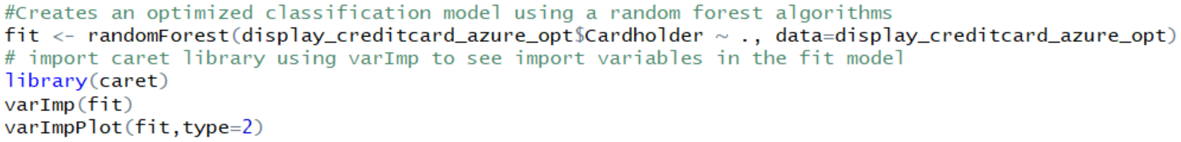
For the sake of completeness, the subsequent step involves testing the correlation results by performing a correlation test on the optimised dataset “display_creditcard_azure_opt”.



```
> print(display_creditcard_azure_opt.cor$tests, digits = 2)
                   frequency      sex      Age NoInhabitants UrbanRatio AverageSalary Unemployment95 Unemployment96
frequency            0.0e+00  0.0e+00  0.0e+00       0.0e+00    0.0e+00       0.0e+00        0.0e+00        0.0e+00
sex                  9.7e-01  0.0e+00  0.0e+00       0.0e+00    0.0e+00       0.0e+00        0.0e+00        0.0e+00
Age                  2.9e-07  9.5e-08  0.0e+00       0.0e+00    0.0e+00       0.0e+00        0.0e+00        0.0e+00
NoInhabitants       2.1e-247 1.9e-249 2.0e-248       0.0e+00    0.0e+00       0.0e+00        0.0e+00        0.0e+00
UrbanRatio           9.4e-20  2.8e-19  5.2e-23      4.9e-324    0.0e+00       0.0e+00        0.0e+00        0.0e+00
AverageSalary        2.4e-12  1.1e-12  1.7e-16      4.2e-289    2.5e-45       0.0e+00        0.0e+00        0.0e+00
Unemployment95       9.0e-07  2.3e-04  2.0e-10      4.8e-312    1.5e-85       7.9e-89        0.0e+00        0.0e+00
Unemployment96       1.9e-05  1.0e-03  2.0e-08      2.7e-310    8.2e-95       2.8e-72        2.8e-15        0.0e+00
CrimeRatio95         5.3e-07  1.6e-07  5.2e-12      1.6e-299    8.8e-44       1.1e-28        7.8e-94        2.5e-76
CrimeRatio96         7.4e-13  7.0e-14  1.5e-18      3.2e-285    1.2e-55       1.4e-37        3.2e-78        1.9e-70
EnterpreneursRatio   8.3e-07  5.1e-05  8.5e-09      1.3e-322    2.0e-75       1.7e-65        8.8e-63        2.2e-67
AvgIncome            1.7e-14  1.0e-05  6.0e-12      2.1e-246    4.2e-22       3.1e-15        5.2e-08        4.4e-08
AvgExpenses          6.8e-16  1.5e-06  1.6e-12      1.6e-247    2.6e-23       4.4e-16        1.3e-08        1.2e-08
AvgBalance           1.1e-03  7.0e-01  9.9e-06      8.1e-243    9.2e-17       2.7e-11        1.9e-03        6.3e-04
Cardholder           2.0e-01       NA  2.6e-08      4.3e-249    1.3e-19       2.9e-12        4.4e-04        1.8e-03
                   CrimeRatio95 CrimeRatio96 EnterpreneursRatio AvgIncome AvgExpenses AvgBalance Cardholder
frequency               0.0e+00      0.0e+00            0.0e+00   0.0e+00     0.0e+00       0.00          0
sex                     0.0e+00      0.0e+00            0.0e+00   0.0e+00     0.0e+00       0.00          0
Age                     0.0e+00      0.0e+00            0.0e+00   0.0e+00     0.0e+00       0.00          0
NoInhabitants           0.0e+00      0.0e+00            0.0e+00   0.0e+00     0.0e+00       0.00          0
UrbanRatio              0.0e+00      0.0e+00            0.0e+00   0.0e+00     0.0e+00       0.00          0
AverageSalary           0.0e+00      0.0e+00            0.0e+00   0.0e+00     0.0e+00       0.00          0
Unemployment95          0.0e+00      0.0e+00            0.0e+00   0.0e+00     0.0e+00       0.00          0
Unemployment96          0.0e+00      0.0e+00            0.0e+00   0.0e+00     0.0e+00       0.00          0
CrimeRatio95            0.0e+00      0.0e+00            0.0e+00   0.0e+00     0.0e+00       0.00          0
CrimeRatio96            1.1e-17      0.0e+00            0.0e+00   0.0e+00     0.0e+00       0.00          0
EnterpreneursRatio      6.2e-82      6.0e-78            0.0e+00   0.0e+00     0.0e+00       0.00          0
AvgIncome               1.1e-10      6.4e-18            8.1e-09   0.0e+00     0.0e+00       0.00          0
AvgExpenses             2.0e-11      1.0e-18            1.8e-09   2.9e-04     0.0e+00       0.00          0
AvgBalance              6.4e-06      8.8e-15            4.3e-05   2.2e-04     2.4e-05       0.00          0
Cardholder              9.2e-08      5.3e-14            2.8e-04   1.3e-06     1.3e-07       0.12          0
```

Figure 5-24: Correlation tests overview of the optimised dataset (display_creditcard_azure_opt)

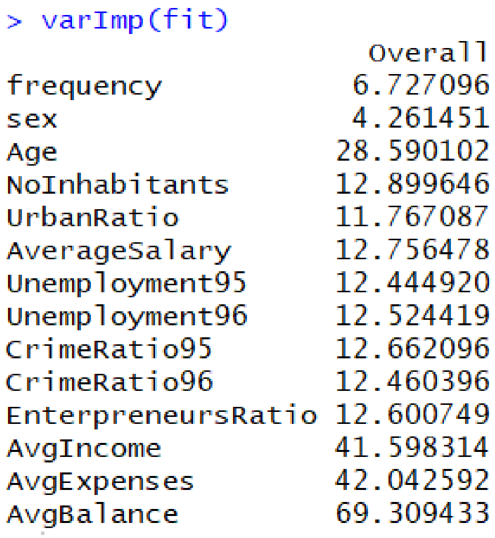
After computing and testing the correlation of the possible pairs from the original pre-processed dataset, the following R-code snippet supports the research purpose to build a cost-sensitive data model.



```
#Creates an optimized classification model using a random forest algorithms
fit <- randomForest(display_creditcard_azure_opt$Cardholder ~ ., data=display_creditcard_azure_opt)
# import caret library using varImp to see import variables in the fit model
library(caret)
varImp(fit)
varImpPlot(fit,type=2)
```

Figure 5-25: R-code snippet for building a cost-sensitive data model - cross-selling

The following figure 5-26 shows a table of the varImp() function results for the underlying processed dataset. The top-most variables overall such as ‘AvgBalance’, ‘AvgExpenses’, ‘AvgIncome’ and ‘Age’ are the most significant for the predictive model. Therefore, the variable ‘AvgBalance’ has the highest value, and is thereby the most important variable with the highest impact in our predictive model. The ranking of the variables is measured by the Gini Index. The variables ‘frequency’ and ‘sex’ are relatively irrelevant for the modelling as they have the lowest overall importance scores. Based on the varImp() results, we assume that we can exclude the variables ‘frequency’ and ‘sex’ in our random forest model without having a significant impact in predicting cross-selling candidates accurately. However, there are no fixed concept criteria to measure the variables’ importance. The variables with the highest importance scores overall are those that probably provide the best prediction and contribution to the cross-selling model. The predictive outcomes will be explained in the next section.



```
> varImp(fit)
                      Overall
frequency            6.727096
sex                  4.261451
Age                 28.590102
NoInhabitants       12.899646
UrbanRatio          11.767087
AverageSalary       12.756478
Unemployment95      12.444920
Unemployment96      12.524419
CrimeRatio95        12.662096
CrimeRatio96        12.460396
EnterpreneursRatio  12.600749
AvgIncome           41.598314
AvgExpenses         42.042592
AvgBalance          69.309433
```

Figure 5-26: Overview of the varImp () results for the optimised dataset - 2nd research project

The variable importance plot below illustrates the relevant variables in our random forest model. It emerges that there are many variables such as ‘NoInhabitants’, ‘UrbanRatio’, ‘AverageSalary’, ‘Unemployment95’, ‘Unemployment96’, ‘CrimeRatio95’, ‘CrimeRatio96’ and ‘EnterpreneursRatio’ that have nearly the same significance score and the variable ‘frequency’ and ‘sex’ will probably not have a strong impact on our predictions in case we will exclude them in the model.

The graph below shows by how much will the MeanDecreaseGini score increase if a variable is assigned values by random permutation. If we randomly permute the ‘AvgBalance’, the MeanDecreaseGini will increase by randomly 70% on average. This observation makes sense, since the overall average balance of a bank customer account has increased in recent years. The same observations can be noticed by the remaining variable such as ‘AvgExpenses’, ‘AvgIncome’ and ‘Age’. The descriptive results highlight that the top variable from our random forest model will also increase the prediction power of the cross-selling model, and if we reduce one of the bottom variables such as ‘frequency’ or ‘sex’, there might not be a huge impact on the prediction power of the predictive model. Regarding the prediction results – which will be dealt within the next section – it is also relevant to validate the descriptive results and their assumptions in detail by developing different forecast models for the cost-sensitive dataset concerning various subsets of mixed significant variables. The evaluation results of corresponding forecast models will be discussed in detail in the next section.

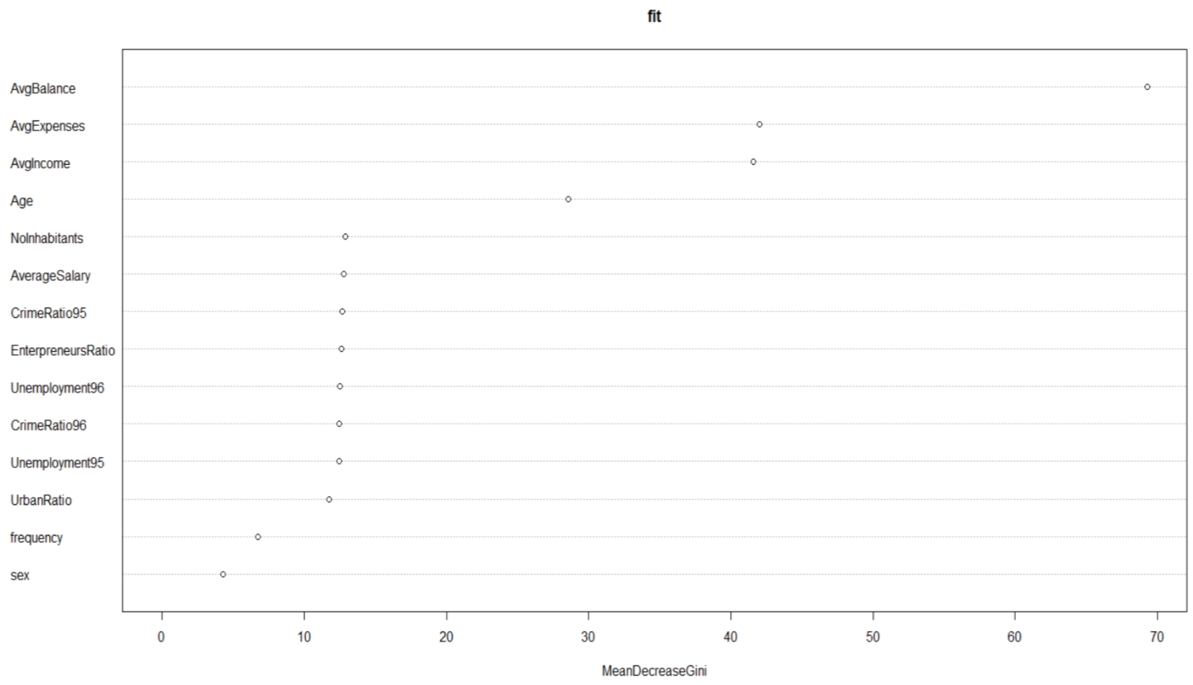
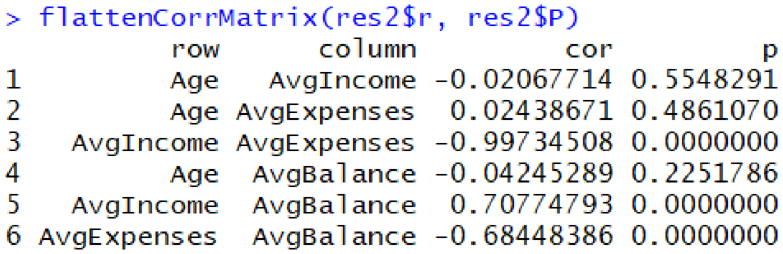


Figure 5-27: Visualisation of significant variables within the data model for the 2nd research project

Finally, I have computed a flattened correlation matrix with significance levels (p-value) for all possible pairs according to the significant variables of the optimised dataset using the function rcorr() from the R-package “Hmisc” and a self-developed function to display a flatten correlation matrix. The corresponding R-script is given in appendix E. The table displayed in figure 5-28 below shows the causal relationships between the selected variables. The correlation coefficient (-0.997) between ‘AvgIncome’ and ‘AvgExpenses’ shows that a decrease in ‘AvgExpenses’ would result in an increase of

‘AvgIncome’. There is also a negative relationship (-0.68) between ‘AvgExpenses’ and ‘AvgBalance’, which indicates that a decrease of ‘AvgBalance’ will turn into an increase of ‘AvgExpenses’. The strong positive relationship (0.707) between ‘AvgIncome’ and ‘AvgBalance’ indicates that the higher the income, the higher the balance. Regarding the p-values (p=0.000) of these relationships, there is no evidence about the significance of the associations between the analysed relationships.



```
> flattenCorrMatrix(res2$r, res2$P)
          row      column         cor         p
1         Age   AvgIncome -0.02067714 0.5548291
2         Age AvgExpenses  0.02438671 0.4861070
3   AvgIncome AvgExpenses -0.99734508 0.0000000
4         Age  AvgBalance -0.04245289 0.2251786
5   AvgIncome  AvgBalance  0.70774793 0.0000000
6 AvgExpenses  AvgBalance -0.68448386 0.0000000
```

Figure 5-28: Flattened correlation matrix of the top-most important variables for the optimised dataset - 2nd research project

The following descriptive results includes the visualised results of the correlation matrix by drawing a performance analytics chart highlighting the most strongly correlated variables and their relationships in the cost-sensitive dataset. Accordingly, I have displayed a chart of the correlation matrix by using the function chart.correlation() from the R-package “PerformanceAnalytics”.

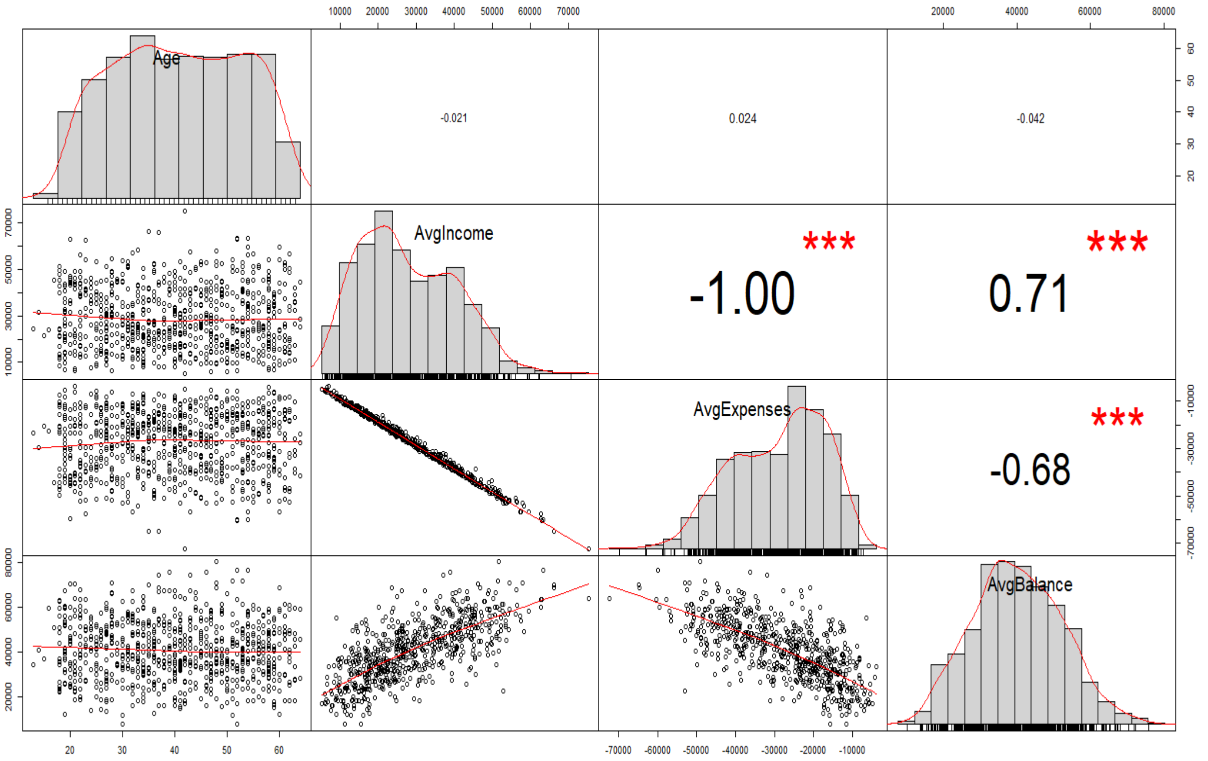


Figure 5-29: Performance analytics chart of the correlation matrix for the varImp variables – 2nd research project

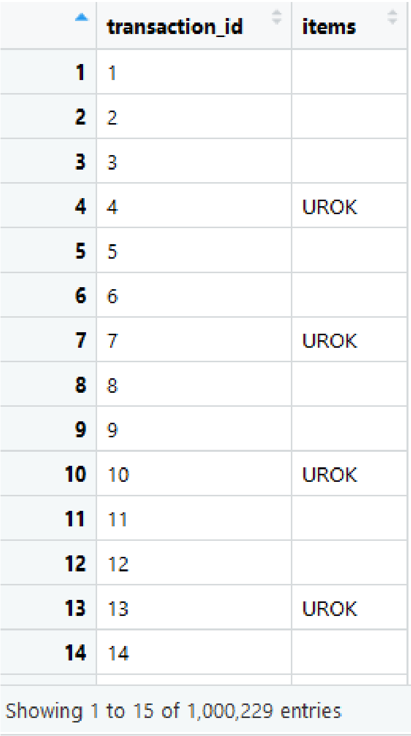
The diagonal of the plot above shows the distribution of every significant variable, whereby ‘AvgIncome’ is skewed to the left, ‘AvgExpenses’ is skewed to the right, ‘AvgBalance’ – which can also be summarised from both datasets – is slightly skewed to the left, and ‘Age’ is uniformly distributed around the customer base. On the top of the diagonal, the value of the correlation plus the significance level is displayed as stars. The associations between ‘AvgBalance’ vs. ‘AvgIncome’ (0.71) and ‘AvgBalance’ vs. ‘AvgExpenses’ (-0.68) is rated as highly significant with three stars (p-values of 0.001).

5.1.3. Analysis results of the categorised transactional dataset

The following section describes the overall evaluations and findings discovered in analysing the categorised transactional dataset. As mentioned in the previous chapter, the pre-processing phase has processed different datasets to answer the outlined research questions. Therefore, various mining methods such as frequent sequences analysis, graph-based clustering and data visualisation algorithms for analysing the transactional payment behaviour were presented. The focus of this section will be on explaining the descriptive results of the applied unsupervised learning algorithms according to the last research project. Therefore, the research analysis applies various data visualisation techniques that combine graph-based topology representation and dimensionality reduction methods to visualise the transactional payment behaviour in a low-dimensional vector space.

The following paragraphs present some practical results in analysing transactional payments data by applying frequency analysis as well as association rules to gain valuable insights from the payment data. The basic idea was to differentiate between different payment categories and describe the results of clustering based on various categorised transaction streams. Another definition of clusters has been used with a frequency analysis, whereby the aim here is to analyse frequencies in the item sets of selected categorised transactions. However, Pijls (1999) has already investigated the fields ‘operation’ and ‘k_symbol’ within the transaction file, albeit without placing the focus on the transaction field ‘k_symbol’, unlike the current research work. Within this context, Pijls examined that the most frequent mode in the ‘operation’ field is ‘withdrawal in cash’ (around 40%) and a majority have an empty string or a string of only one space. Among the transaction category ‘withdrawal in cash’, the most frequent ‘k_symbol’ string is ‘payment for statement’. The underlying research design focuses only on the transaction field ‘k_symbol’ to apply a target-oriented market basket analysis. All existing payment categories – stored in the field ‘k_symbol’ – are processed for deeper descriptive analysis, as shown in table 5-1 below.

The pre-processed table comprises over one million transactions reduced in two relevant attributes ‘transaction_id’ and ‘items’ from the original transactional table called “trans.ascii”. This intermediate step in the pre-processing phase supports the conducted research experiment to read all existing transactions in a proper basket format to convert the data into an object of the transaction class. This is the basis for mining a large-scale payment transaction dataset using the major data mining technique association rules.



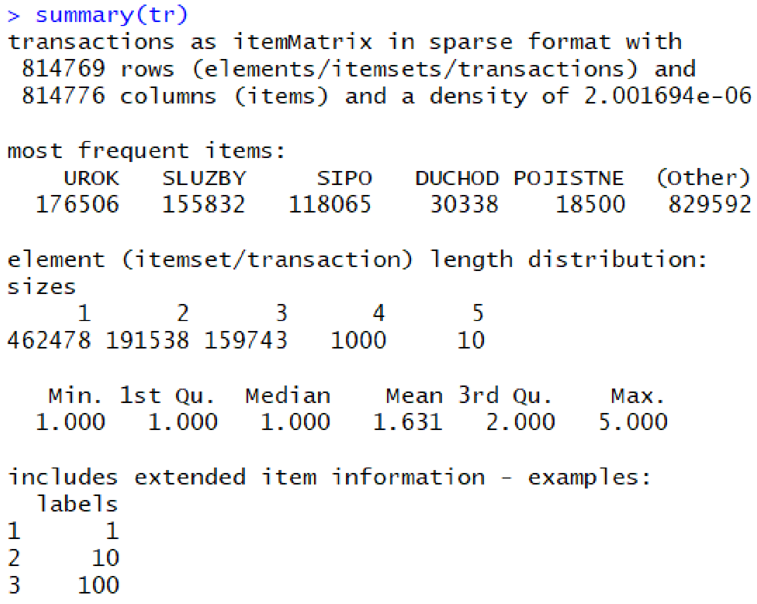
	transaction_id	items
1	1	
2	2	
3	3	
4	4	UROK
5	5	
6	6	
7	7	UROK
8	8	
9	9	
10	10	UROK
11	11	
12	12	
13	13	UROK
14	14	

Showing 1 to 15 of 1,000,229 entries

Table 5-1: View of the table - items per transaction

Pijls (1999) highlighted in his evaluation that the most frequent combinations for the fields ‘operation’ and ‘k_symbol’ within the raw dataset from Berka are ‘withdrawal in cash’ with an empty ‘k_symbol’ field (274,675 records), ‘interest credited’ (183,114 records) with empty values in ‘operation’ field, ‘credit in cash’ (156,743 records) with an empty ‘operation’ field, and ‘withdrawal in cash’ in combination with ‘payment for statement’ (155,832 records). Pijls (1999) underlines that “cash operations (credit and withdrawal in cash) and automatic operations (interest and payment for statement) are by far most frequent.”

The below output of the fetched payment transaction categories indicates that there are 814,769 transactions (rows) and 814,776 items (columns). The results of the element (itemset/transaction) length distribution show that there are 462,478 transactions for one payment category (item), 191,538 transactions for two items, 159,743 transaction for three items, 1,000 transaction for four items, and there are items with ten transactions, which are the longest. The most frequent payment category (item) in the processed dataset is ‘UROK’, which occurs in 176,506 transactions. More details about the frequency will be shown in the subsequent figures below.



```
> summary(tr)
transactions as itemMatrix in sparse format with
 814769 rows (elements/itemsets/transactions) and
 814776 columns (items) and a density of 2.001694e-06

most frequent items:
    UROK    SLUZBY      SIPO    DUCHOD POJISTNE   (Other)
  176506    155832    118065     30338    18500    829592

element (itemset/transaction) length distribution:
sizes
     1      2      3      4      5
462478 191538 159743   1000     10

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   1.000   1.000   1.631   2.000   5.000

includes extended item information - examples:
  labels
1      1
2     10
3    100
```

Figure 5-30: Summary of fetched payment transaction categories

First of all, I have generated an item frequency plot to create an item frequency bar plot for the top ten items to analyse the distribution of the payment categories based on an item matrix. Therefore, figure 5-31 below shows the numeric frequencies of each payment category independently. However, through his descriptive analysis Pijls (1999) presents some interesting information of the transaction file by analysing the ‘date’ field, which might hold interest in the context of understanding the underlying dataset. He has highlighted that more than 30% of the transactions are executed on the 30th or 31st of a month, and ‘withdrawal in cash’ combined with ‘payment for statement’ in the ‘k_symbol’ field is another frequent transaction on the last day of a month. The category ‘crediting interest’ is only performed on the last day of a month. Loan payments are only executed on the 12th of each month and only occur in combination with a remittance to other banks in the operation field (Pijls, 1999). Another significant frequency pattern occurs with the withdrawals in cash in January.

The bar plots illustrate the payment categories that are frequently transferred in this banking accounts, and it is notable that the support of even the most frequent items is relatively low. For example, the most frequent payment categories only occur in around 1.5% of transactions). I have used these insights to specify the minimum threshold when running the Apriori algorithm; for instance, the applied algorithm may return a reasonable number of rules along the research experiments once we will set the support threshold at well below 0.015. The item frequency plot below indicates that many banking clients transfer the payment categories ‘UROK’, ‘SLUZBY’ and ‘SIPO’.

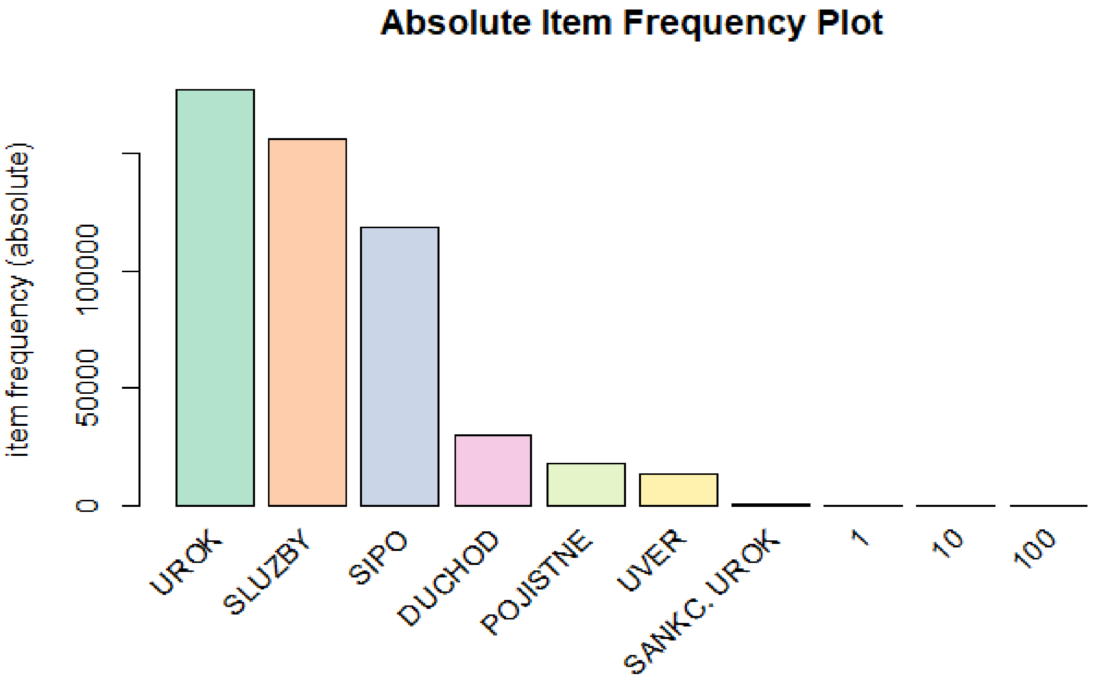


Figure 5-31: Absolute item frequency plot for the transaction field ‘k_symbol’

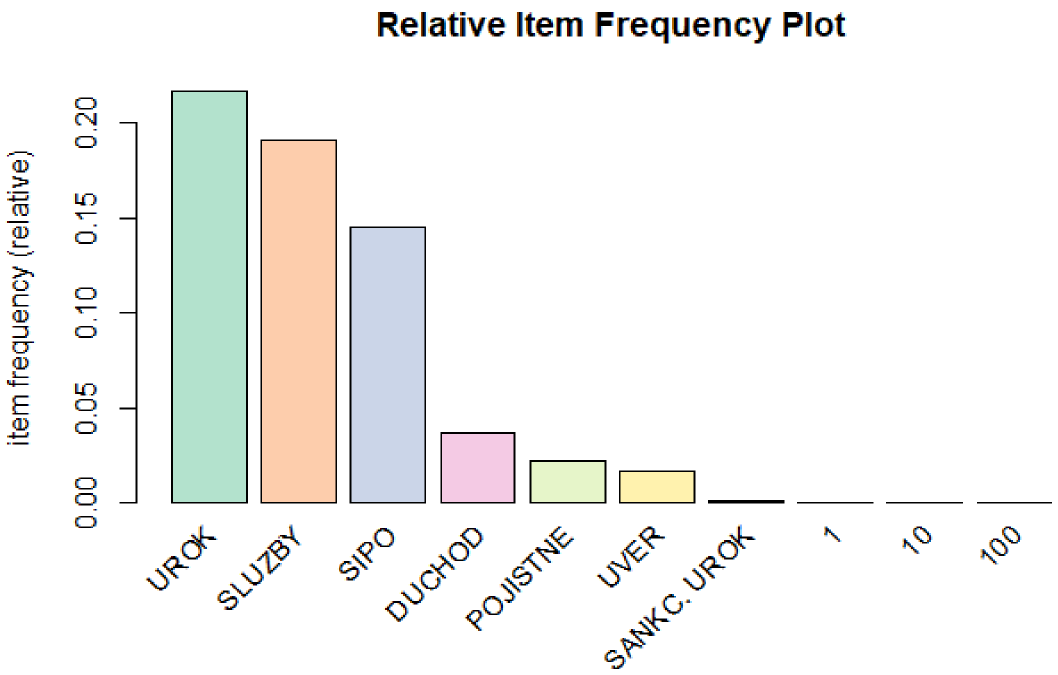
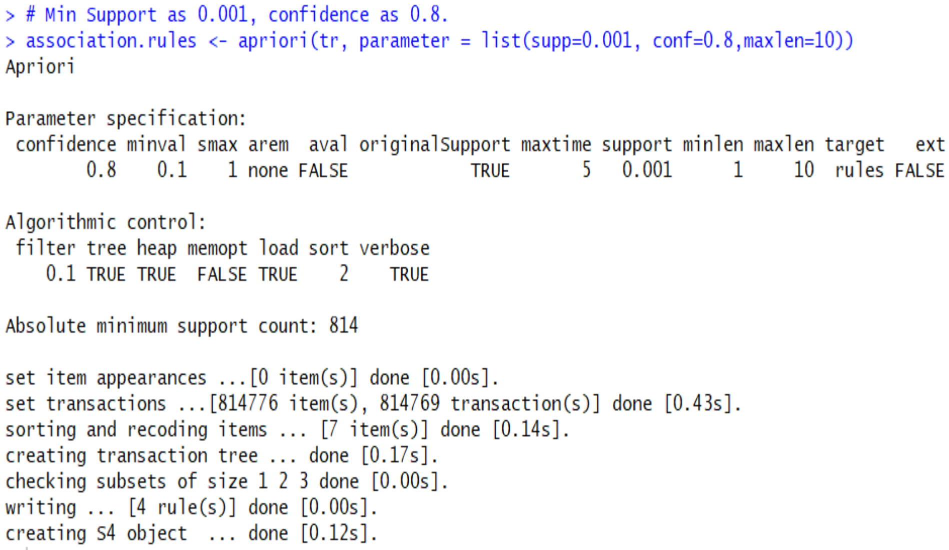


Figure 5-32: Relative item frequency plot for the transaction field ‘k_symbol’

Figure 5-32 above illustrates how many times the payment categories have appeared compared with others. The plot shows that ‘UROK’ (interest credited), ‘SLUZBY’ (payment for statement) and ’SIPU’ (household) have the most sales. Therefore, in order to increase the sale of ‘POJISTNE’ (insurance payment) or ‘DUCHOD’ (old-age pension), the banks can promote these products with ’SIPU’ (household). Note that this conclusion is not a direct or comprehensive reflection of the success of any cross-selling opportunities. A reliable forecast might be produced based upon supervised learning algorithms.

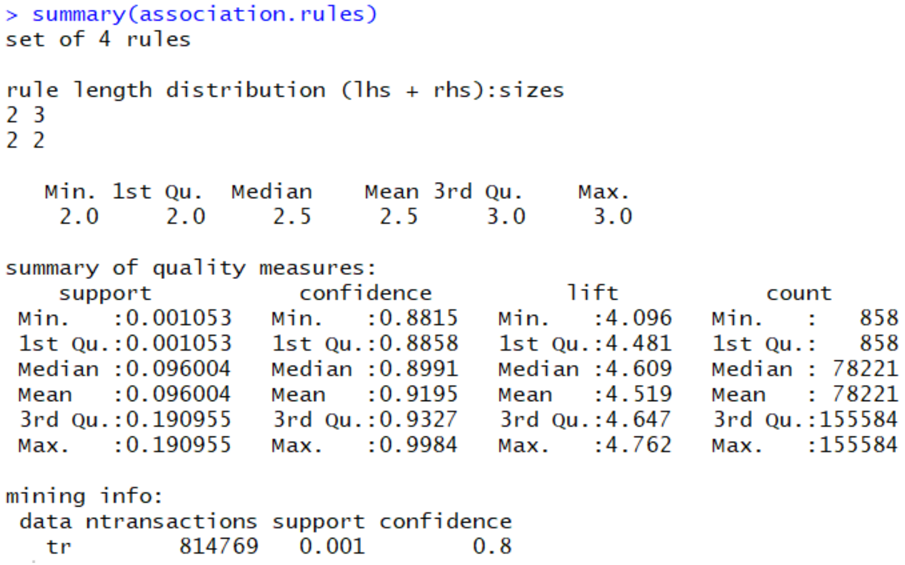
The next descriptive analysis step is to mine the rules using the Apriori algorithm implemented in the R-package “arules”. The function apriori() is applied with the parameter specification of a minimum support threshold of 0.001 and the default minimum confidence threshold of 0.8. The absolute minimum support count is 814. After cleaning up the transactional data again, the algorithm processed 814,769 transactions within 0.43s. The result is a set of our association rules. Further analysis details through the association rules generation are displayed in figure 5-33 below.



```
> # Min Support as 0.001, confidence as 0.8.
> association.rules <- apriori(tr, parameter = list(supp=0.001, conf=0.8,maxlen=10))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target   ext
        0.8    0.1    1 none FALSE            TRUE       5   0.001      1     10  rules FALSE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 814

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[814776 item(s), 814769 transaction(s)] done [0.43s].
sorting and recoding items ... [7 item(s)] done [0.14s].
creating transaction tree ... done [0.17s].
checking subsets of size 1 2 3 done [0.00s].
writing ... [4 rule(s)] done [0.00s].
creating S4 object  ... done [0.12s].
```

Figure 5-33: Association rules generation by applying Apriori algorithm

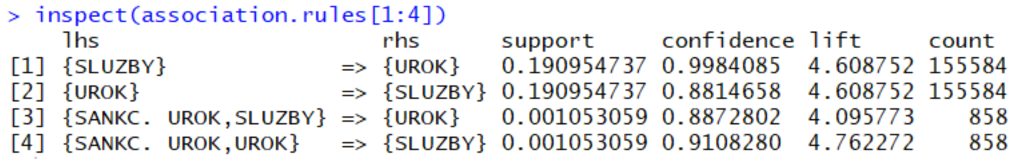
The summary overview of the generated association rules shows that a length of three payment categories (items) has the most rules (a total of two rules), and the length of two payment categories (items) has the lowest number of rules (a total of two rules).



```
> summary(association.rules)
set of 4 rules

rule length distribution (lhs + rhs):sizes
2 3
2 2

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    2.0     2.0     2.5     2.5     3.0     3.0

summary of quality measures:
    support            confidence          lift            count
 Min.   :0.001053   Min.   :0.8815   Min.   :4.096   Min.   :   858
 1st Qu.:0.001053   1st Qu.:0.8858   1st Qu.:4.481   1st Qu.:   858
 Median :0.096004   Median :0.8991   Median :4.609   Median : 78221
 Mean   :0.096004   Mean   :0.9195   Mean   :4.519   Mean   : 78221
 3rd Qu.:0.190955   3rd Qu.:0.9327   3rd Qu.:4.647   3rd Qu.:155584
 Max.   :0.190955   Max.   :0.9984   Max.   :4.762   Max.   :155584

mining info:
 data ntransactions support confidence
   tr        814769   0.001        0.8
```

Figure 5-34: Summary overview of generated association rules

The inspection of the association rules using the R-Package “arules” examines according to a measure of interestingness (e.g. support, confidence and lift) whether the result is a set of payment category groups that I would expect to be highly correlated. Appendix C.3 provides descriptive analysis results when using an interactive inspect method that creates a data table using the inspectDT () function from the R-package “arulesViz”. This technique allows me to sort rules given different interest measures, specify ranges for measures, and provides filters for the payment categories (items). Due to the applied association algorithm, the resulting associations illustrated in the figure below can express statements like if the bank client regularly made a transaction for ‘UROK’ (interest credited) AND combined this transaction with ‘SANKC.UROK’ (interest if negative balance), THEN the possibility that the transaction ‘SLUZBY’ (payment for statement) occurs will increase. Finally, the top four rules with respect to the lift measure – a popular measure of rule strength – are shown in figure 5-35 below.



```
> inspect(association.rules[1:4])
    lhs                     rhs      support     confidence lift     count
[1] {SLUZBY}             => {UROK}   0.190954737 0.9984085  4.608752 155584
[2] {UROK}               => {SLUZBY} 0.190954737 0.8814658  4.608752 155584
[3] {SANKC. UROK,SLUZBY} => {UROK}   0.001053059 0.8872802  4.095773    858
[4] {SANKC. UROK,UROK}   => {SLUZBY} 0.001053059 0.9108280  4.762272    858
```

Figure 5-35: Exploring and inspection of generated association rules

Figure 5-35 depicts further analysis results through exploring and inspection of the generated association rules. Using the inspect () function from R-Package helps the conducted research experiments to detect the rules with the highest lift. The matrix-based visualisation technique of the inspect () function results describes the number of unique itemsets in the consequent (rhs) / antecedent (lhs) in the set of the four generated rules. The applied grouping methods enables the conducted research experiments to even group antecedents containing payment categories (e.g. ‘SANC.UROK’ (interest if negative balance) and ‘SLUZBY’ (payment for statement)) which are rarely transferred together since both items will have similar dependence to the same consequents ‘UROK’ (interest credited). The antecedent group of ‘SANC.UROK’ (interest if negative balance) and ‘UROK’ (interest credited) is also statistical dependent on the same consequent ‘SLUZBY’ (payment for statement) and thus can be group together. The group of most interesting rules according to the displayed lift (the default measure) is shown in the ascending order of the table above or in the interactive table given in appendix C.3. There are two interesting rules that contain the ‘SLUZBY’ (payment for statement) item in the antecedent and the consequent is the ‘UROK’ (interest credited) and ‘UROK’ (interest credited) item in the antecedent and the consequent is ‘SLUZBY’ (payment for statement). Both single relationships indicate strong associations since the lift values are greater than 1.

The next paragraphs intend to introduce different visualisation techniques implemented with the R-Package “arulesViz”. First, I have used with a scatter plot a straight-forward visualisation technique measuring two interesting parameters (the support and confidence) on the axes. On the right hand side of the plot, the lift parameter is completing the interest measure metric. During the descriptive analysis, I have filtered only rules with a confidence greater than 0.8 or 80%, at least to remove redundant rules. Figure 5-36 below describes the scatterplot with the four selected rules. The scatterplot depicts that the most interesting rules comprises high lifts and have a relatively low support. The plot mainly provides an overview of the distribution of support and confidence border in the generated rule set. There are a ultimately two high-confidence rules in the top left corner, and high-lift rules are located close to the minimum support threshold (left corner of the plot).

The following paragraphs illustrate descriptive results using graph-based visualisations of the top 10 sub-rules for the generated association rules as the results are only viable for very small sets of rules. I have used again the R-extension package called “arulesViz” which implements several known and novel visualisation techniques to explore the association rules in detail. However, the descriptive results provide a very clear representation of the four generated rules when selecting the top 10 rules with the highest lift. Further detailed visualisation results are also given in appendix C.3. An alternative representation is the usage of a 3D bars visualisation due to the given small rule set.

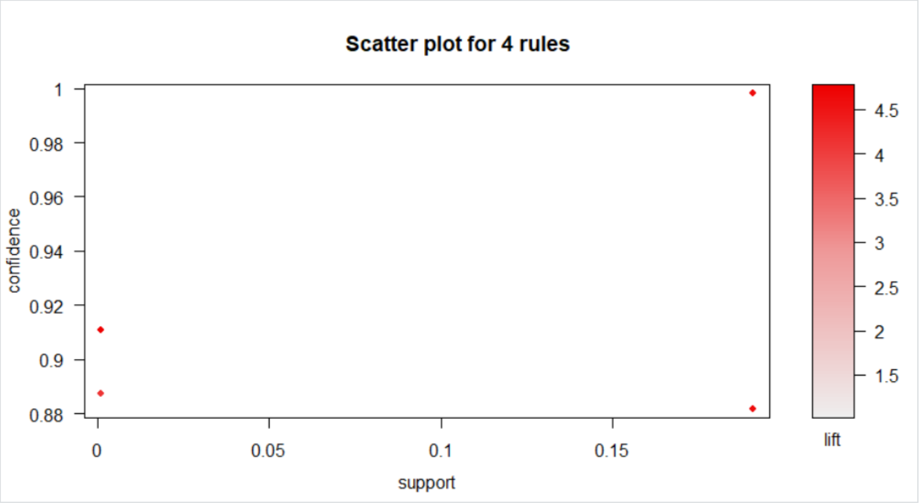


Figure 5-36: Scatter plot for the four generated association rules

Figure 5-37 below illustrates the interactive visualisation of the top 10 sub-rules. The most important rules are becoming a matter of ever greater circles to the importance, which is also highlighted in red colour. This graph-based visualisation is focusing on the relationship between the individual payment categories in the rule set.

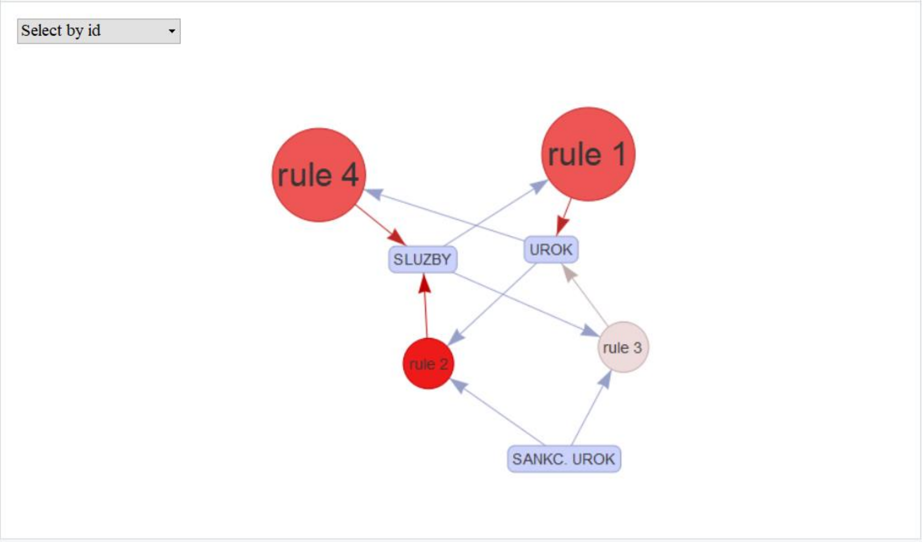


Figure 5-37: Overview of graph-based visualisations of the top 10 sub-rules

The four calculated association rules based on a market basket analysis using association rules are visualised with figure 5-37 above, and may be summarised as follows:

Rule 1 states that in the group of sequences, the consequent of the 'UROK' payment category is triggered by the antecedent 'SLUZBY' payment category. A possible interpretation of that discovery is that the outbound arrow of rule 1 is of considerable importance.

Similarly, rule 4 indicates in the group of sequences, the consequent of the 'SLUZBY' payment category is triggered by the antecedent 'UROK' payment category and the outbound arrow of rule 4 is of considerable importance. Note that assuming ever sequences are distinctive most of the identified rules are not overlapping.

Rule 3 indicates in the group of sequences that both inbound arrows (antecedent 'SLUZBY' and 'SANC.UROK') of rule 3 result in a distinct outbound arrow 'UROK' (consequent).

Rule 2 shows that the consequent of the 'SLUZBY' payment category is triggered by the antecedents 'UROK' and 'SANC.UROK'. Due to the highlighted outbound arrow of rule 2, the consequent 'SLUZBY' is of considerable importance.

In the first step, the corresponding R-script given in appendix E.1 is generating association rules for every single payment category. Following plots illustrates the outcomes of the descriptive analysis results in which I want to inspect all high-lift rules of the graph to find interesting rules. Finally, the research has employed a graph-based visualisation to gain a deeper understanding of the very small set of generated rules and payment categories (items).

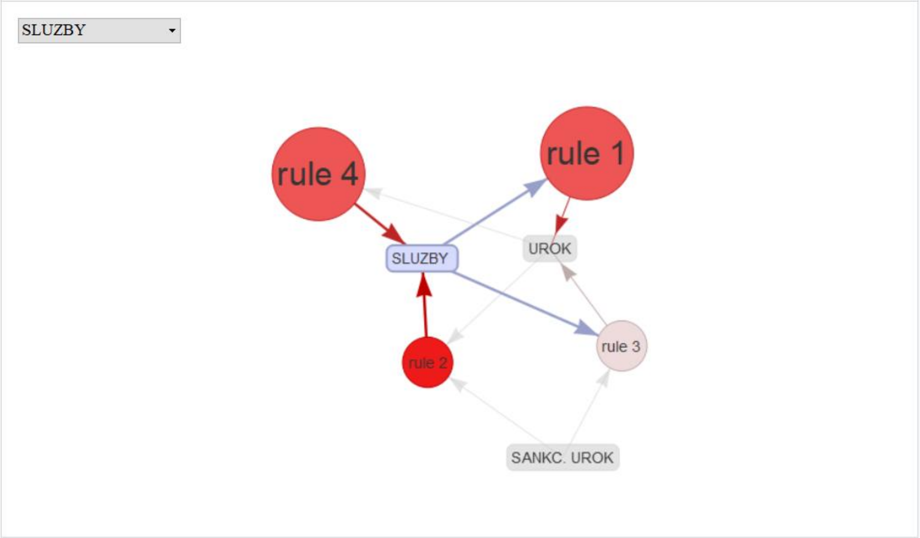


Figure 5-38: Generated association rule - payment for statement (SLUZBY)

The plot above inspects single rules that are connected to ‘SLUZBY’ (payment for statement). Figure 5-38 provides a graphical view of the individual in relationship with each of the existing payment categories. It shows that the payment category ‘SLUZBY’ is the consequent of rule 4 or rule 2, in which rule 4 is more important than rule 2. Hence, the antecedents of the payment category ‘SLUZBY’ is rule 1 or rule 3, in which rule 3 is less important than rule1.

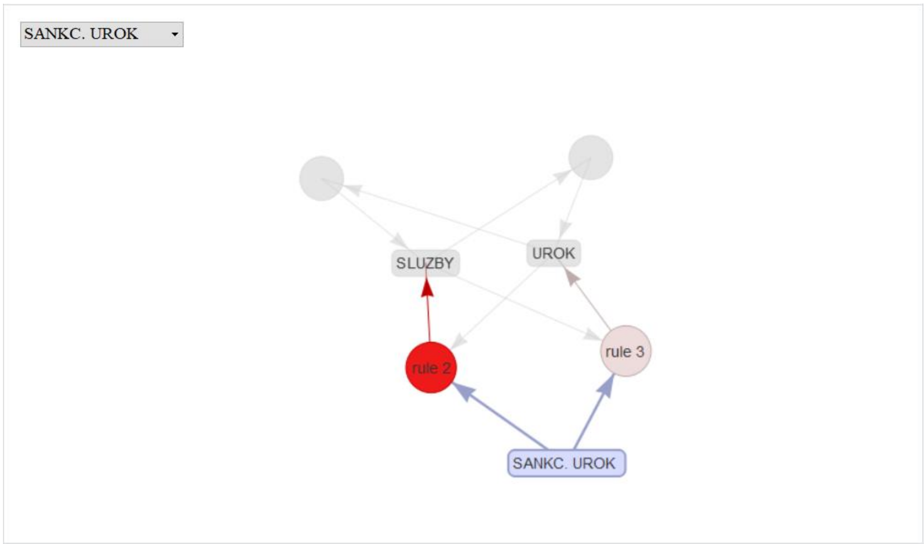


Figure 5-39: Generated association rule - interest if negative balance (SANKC.UROK)

The plot above discovers single rules that are connected to ‘SANKC.UROK’ (interest if negative balance). The graphical view shows the existing relationships around the payment category ‘SANC.UROK’ which is the antecedents of rule 2 or rule 3. The graph depicts that rule 2 is more important than rule 3 since the size and colour of the vertices are emphasised. This interest measure also underlines that there is no significant consequent from a rule to the payment category 'SANKC.UROK' available.

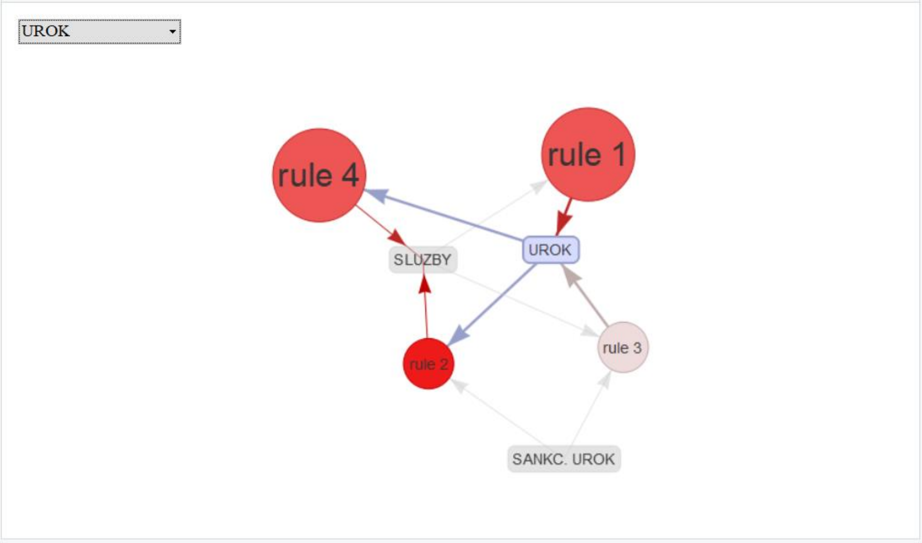
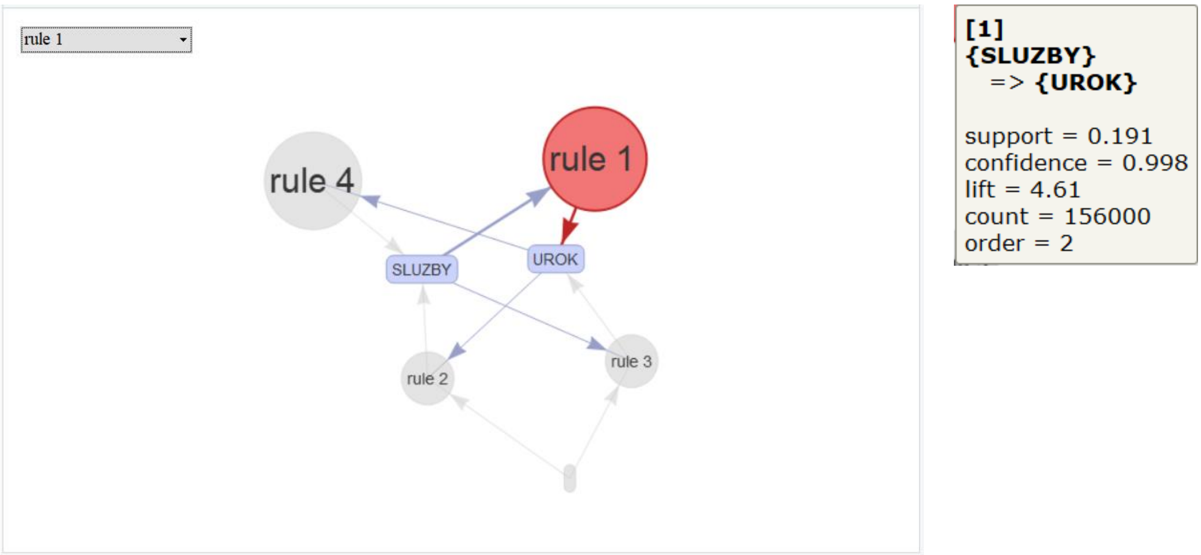


Figure 5-40: Generated association rule - interested credited (UROK)

Figure 5-40 above illustrates the relationships around the payment category 'UROK' (interested credited). The strongest interest measure is given along the inbound arrow (consequent) from rule 1 to the payment category 'UROK' through the outbound arrow (antecedent) from the payment category 'UROK' to rule 4. Hence, the other antecedents of the payment category 'UROK'' is rule 2, which is less important than the association to rule 4. The arrow pointing from rule 3 to the payment category 'UROK' is also not representing a great measure of interest.

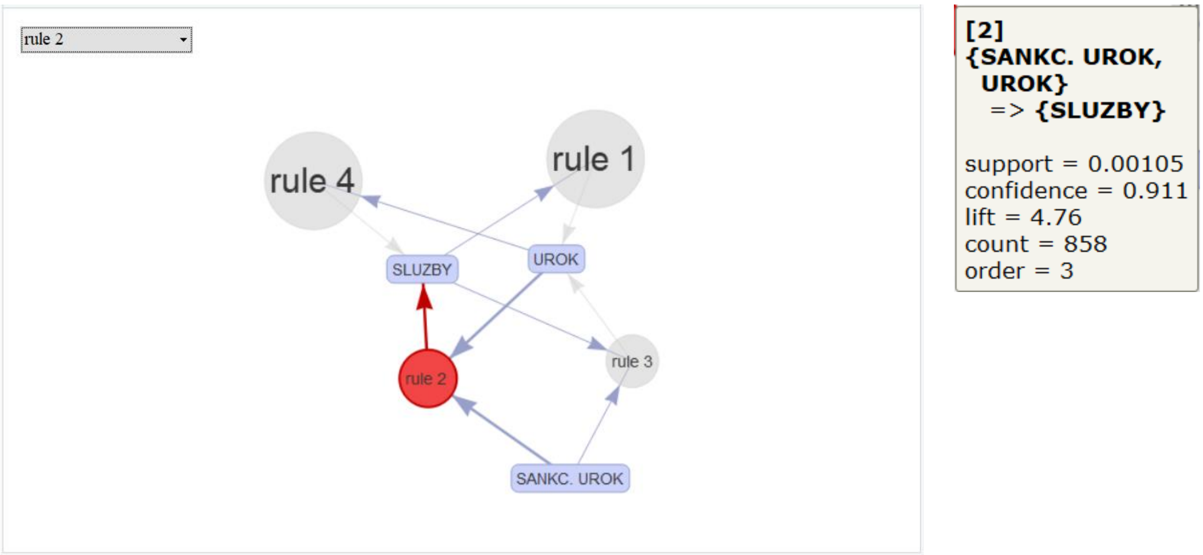
In summary, it can be ascertained that the application of graphs in clustering and visualisation has several advantages. The applied unsupervised learning technique provides a compact representation of the entire pre-processed transactional dataset as the graph edges characterises relations and weights represent similarities or distance between the existing payment categories. The following paragraphs describes clustering's and visualisations of the generated association rules based on the synergistic combination of clustering and graph-theory, which enables the research to utilise information hidden along the plotted graphs.



```
[1]
{SLUZBY}
   => {UROK}

support = 0.191
confidence = 0.998
lift = 4.61
count = 156000
order = 2
```

Figure 5-41: Summary details of rule 1 - SLUZBY association with UROK

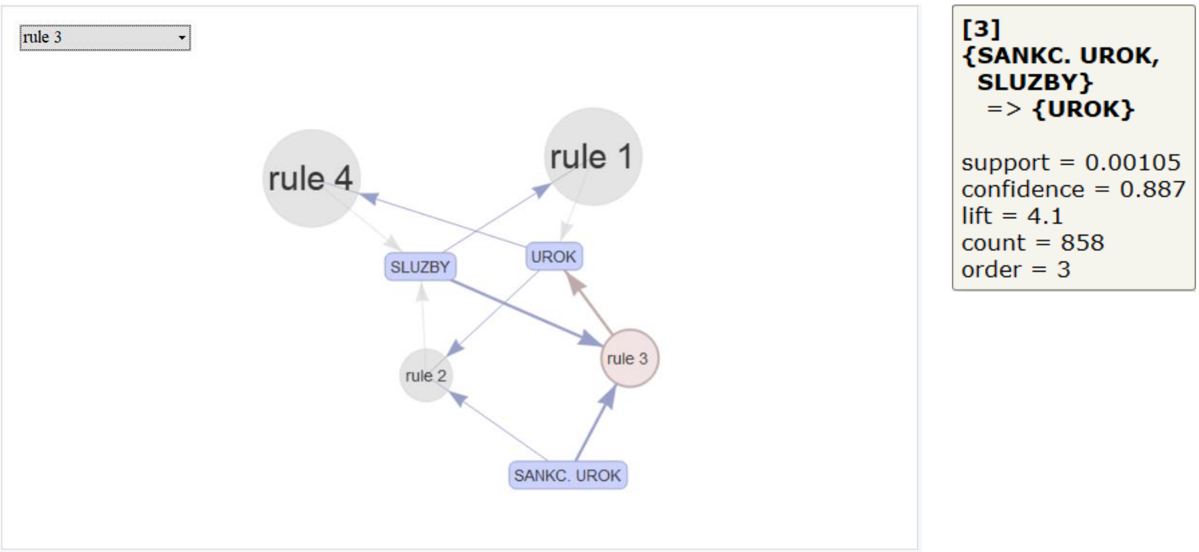
Regarding figure 5-41 above, the summary details of rule 1 shows very strong relationship (due to a high lift value) with ‘SLUZBY’ and ‘UROK’, but a rather low support of only 0.191% respectively. The rule 1 ‘SLUZBY’ => ‘UROK’ [support = 0.191, confidence = 0.998] is a strong rule since the association satisfies the minimum support of 0.001 and minimum confidence of 0.8. The correlation between the two payment categories is described with a high lift value of 4.61 and means that ‘SLUZBY’ and ‘UROK’ are dependent on each other.



```
[2]
{SANKC. UROK,
 UROK}
   => {SLUZBY}

support = 0.00105
confidence = 0.911
lift = 4.76
count = 858
order = 3
```

Figure 5-42: Summary details of rule 2 - SANKC.UROK association with SLUZBY

The summary details of rule 2 reveals that the sequence group of ‘SANKC.UROK’ and ‘UROK’ associated with ‘SLUZBY’ is a strong rule due to its high lift value of 4.76, but the relationship has a very low support of 0.00105%. The rule 2 ‘SANKC.UROK’, ‘UROK’ => ‘SLUZBY’ [support = 0.00105, confidence = 0.911] only account for a small part of relations (count of 858) between these payment categories.

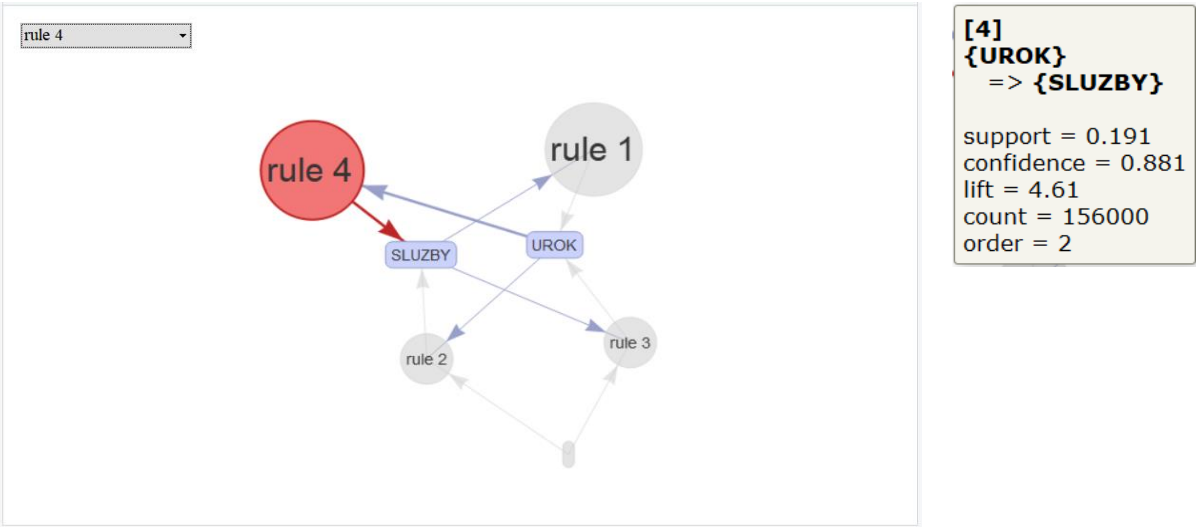


[3]
{SANKC. UROK,
SLUZBY}
=> {UROK}

support = 0.00105
confidence = 0.887
lift = 4.1
count = 858
order = 3

Figure 5-43: Summary details of rule 3 - SANKC.UROK association with UROK

The graphical representation of rule 3 shows that the sequence group of ‘SANKC.UROK’ and ‘SLUZBY’ associated with ‘UROK’ might be a strong rule at the first glance due to its relatively high lift value of 4.1, but the small size of the circles represents a small level of confidence with 0.887 and the light colour of the circle indicates a low level of the lift when comparing these descriptive results with the entire data base. The low number of existing relationships (count of 858) underlines this observation.



[4]
{UROK}
=> {SLUZBY}

support = 0.191
confidence = 0.881
lift = 4.61
count = 156000
order = 2

Figure 5-44: Summary details of rule 4 - UROK association with SLUZBY

The plot above indicates that the rule 4 with the itemset (UROK => SLUZBY) occurs more frequently (count of 156.000) due to their high support values and are likely to be applicable to a large number of future transactions. The high confidence value between the association of ‘UROK’ and ‘SLUZBY’ shows that there is a high likelihood that ‘SLUZBY’ will be transferred, and that there is a great and strong link between the two payment categories (items) due to the high lift value of 4.61. It can be noted that the larger the circle and the darker the grey the better is the detected association rule.

Taking a closer look at the lift values, figure 5-45 below shows the filtered top 10 rules with the highest lift resulting in a parallel coordinates plot for the four generated rules, which visualise the multi-dimensional data separately in each dimension on the x-axis and the y-axis. Each payment category is represented by a line connecting the values for each dimension. Arrows only span sufficient positions on the x-axis to represent all of the payment categories in the discovered rule, for instance, rules with less payment categories such as ‘UROK’ (interest credited) or ‘SLUZBY’ (payment for statement) are shorter arrows due to its higher support scores and the arrows are highlighted with a very intense red due to its high confidence scores. Looking at the top-most arrow in the plot below, it shows that when the bank client has transferred ‘UROK’ (interest credited) in his payment streams, the bank client likely transfers ‘SLUZBY’ (payment for statement) along with these as well.

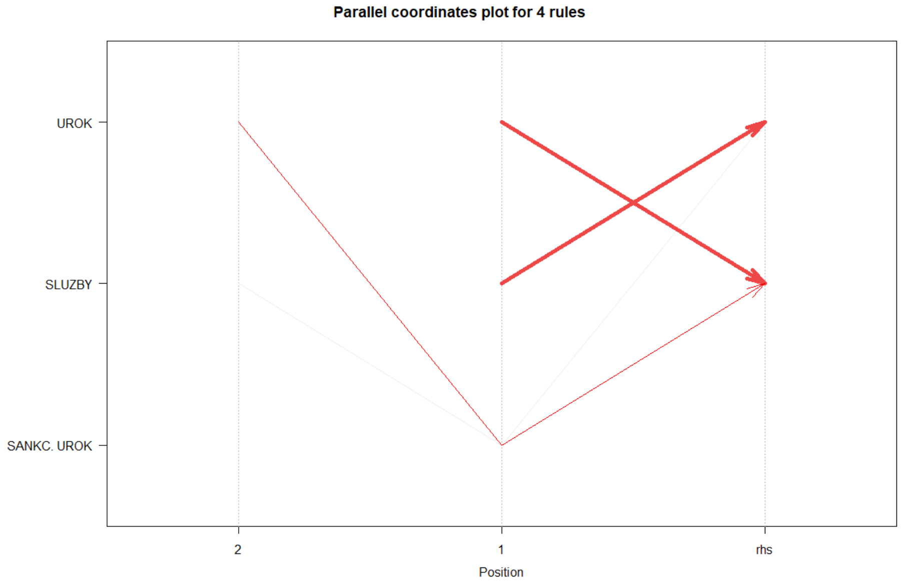
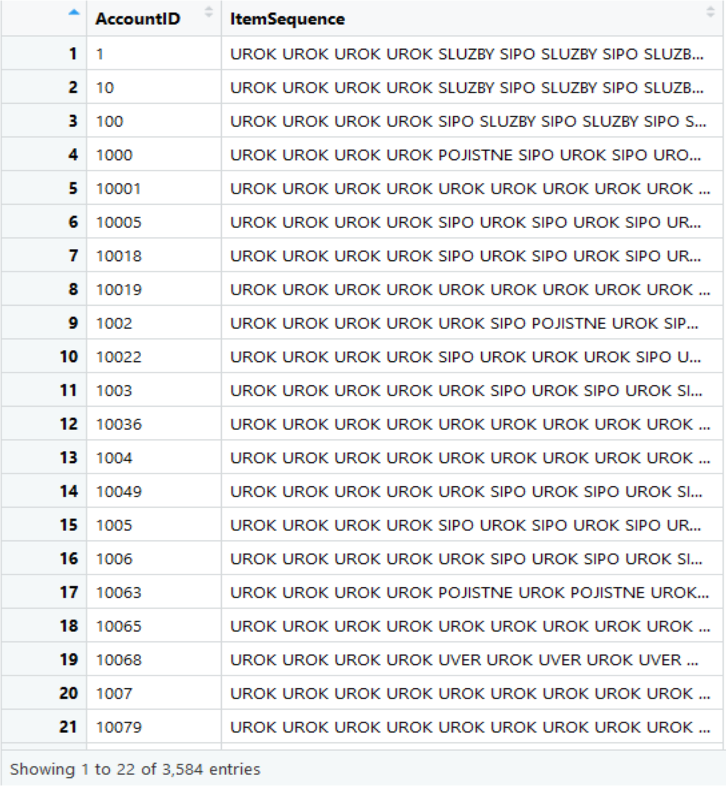


Figure 5-45: Parallel coordinates plot for four generated rules

Weber (1998) highlighted that a final evaluation of the usefulness and interestingness of the discovered rules requires deep understanding of the domain knowledge

(Miksovsky, Zelezny, Stepankova, Pechoucek, 1999; Spenke and Beilken, 1999). Therefore, drawing any final conclusions about the underlying causal dependencies leading to these results will not be serious. However, I am convinced that using unsupervised learning algorithms banking experts with tacit knowledge would detect more hidden knowledge.

The following paragraphs are dealing with the frequent sequence mining to discover sequential patterns and which of them can be applied to the underlying payment transactional dataset. Regarding the data pre-processing phase, the payment categories are cleaned up through various intermediate steps and temporary tables, then stored in a data frame illustrated in the table 5-2 below.

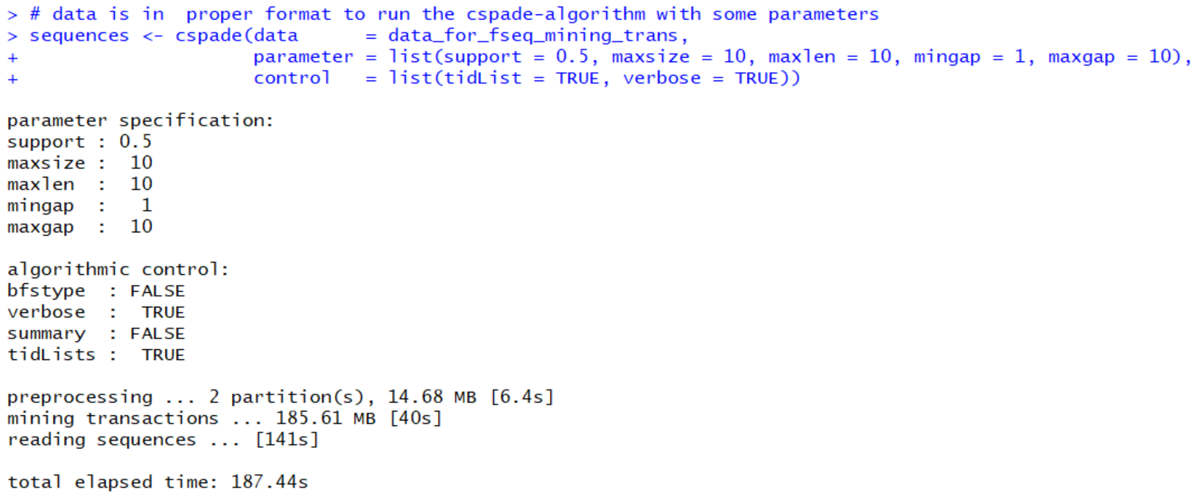


	AccountID	ItemSequence
1	1	UROK UROK UROK UROK SLUZBY SIPO SLUZBY SIPO SLUZB...
2	10	UROK UROK UROK UROK SLUZBY SIPO SLUZBY SIPO SLUZB...
3	100	UROK UROK UROK UROK SIPO SLUZBY SIPO SLUZBY SIPO S...
4	1000	UROK UROK UROK UROK POJISTNE SIPO UROK SIPO URO...
5	10001	UROK UROK UROK UROK UROK UROK UROK UROK UROK ...
6	10005	UROK UROK UROK UROK SIPO UROK SIPO UROK SIPO UR...
7	10018	UROK UROK UROK UROK SIPO UROK SIPO UROK SIPO UR...
8	10019	UROK UROK UROK UROK UROK UROK UROK UROK UROK ...
9	1002	UROK UROK UROK UROK UROK SIPO POJISTNE UROK SIP...
10	10022	UROK UROK UROK UROK SIPO UROK UROK UROK SIPO U...
11	1003	UROK UROK UROK UROK UROK SIPO UROK SIPO UROK SI...
12	10036	UROK UROK UROK UROK UROK UROK UROK UROK UROK ...
13	1004	UROK UROK UROK UROK UROK UROK UROK UROK UROK ...
14	10049	UROK UROK UROK UROK UROK SIPO UROK SIPO UROK SI...
15	1005	UROK UROK UROK UROK SIPO UROK SIPO UROK SIPO UR...
16	1006	UROK UROK UROK UROK UROK SIPO UROK SIPO UROK SI...
17	10063	UROK UROK UROK UROK POJISTNE UROK POJISTNE UROK...
18	10065	UROK UROK UROK UROK UROK UROK UROK UROK UROK ...
19	10068	UROK UROK UROK UROK UVER UROK UVER UROK UVER ...
20	1007	UROK UROK UROK UROK UROK UROK UROK UROK UROK ...
21	10079	UROK UROK UROK UROK UROK UROK UROK UROK UROK ...

Showing 1 to 22 of 3,584 entries

Table 5-2: View of the table - payment category (item) sequences per account

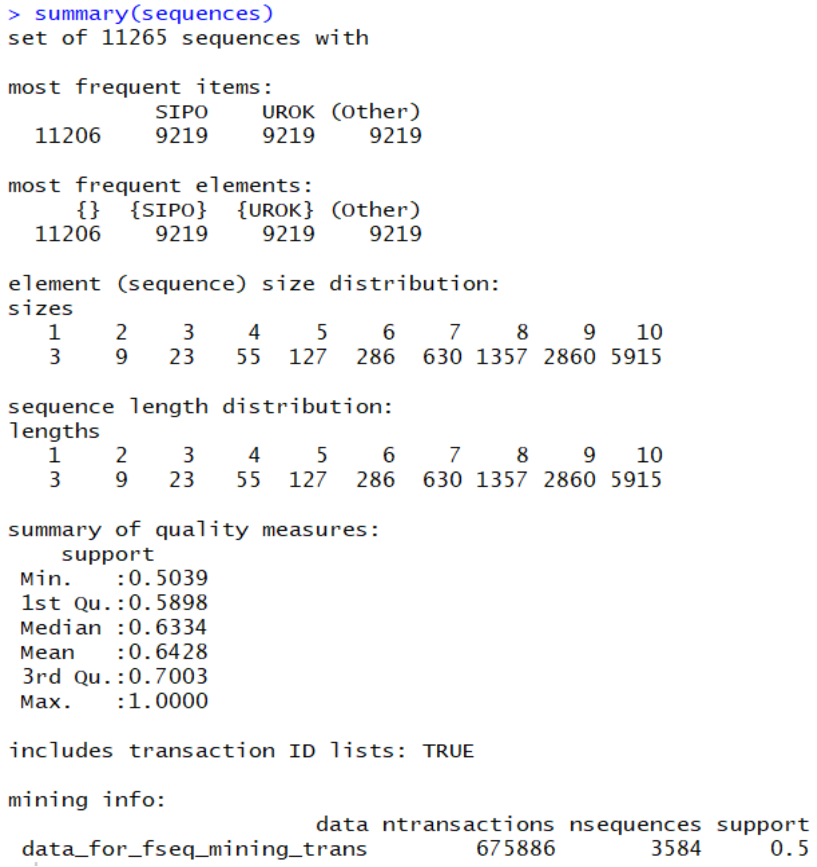
In the next step, I have split the data using the str_split () function from the R-package “stringr” and convert the data to a dataset of class 'transactions' to fulfil the prerequisite when applying the clickstream analysis method. After these prior pre-processing steps, the data was in a proper format to execute the cSPADE algorithm with the data contained in “data_for_fseq_mining_trans” object. Doing this, I have set the support parameter to 0.5 and I have also instructed the algorithm to show a verbose output. The algorithm output for fast discovery of sequential patterns will be the following:



```
> # data is in  proper format to run the cspade-algorithm with some parameters
> sequences <- cspade(data      = data_for_fseq_mining_trans,
+                     parameter = list(support = 0.5, maxsize = 10, maxlen = 10, mingap = 1, maxgap = 10),
+                     control   = list(tidList = TRUE, verbose = TRUE))

parameter specification:
support : 0.5
maxsize :  10
maxlen  :  10
mingap  :   1
maxgap  :  10

algorithmic control:
bfstype  : FALSE
verbose  :  TRUE
summary  : FALSE
tidLists :  TRUE

preprocessing ... 2 partition(s), 14.68 MB [6.4s]
mining transactions ... 185.61 MB [40s]
reading sequences ... [141s]

total elapsed time: 187.44s
```

Figure 5-46: Output of cSPADE algorithm for transaction frequent sequency analysis

The below summary of the analysed sequences shows some basic statistics of the dataset, such as the fact that the dataset comprises 11,265 sequences with the most frequent items ‘SIPO’ (household payment) and ‘UROK’ (interest credited) and the sequence size of the distribution is skewed to the right.



```
> summary(sequences)
set of 11265 sequences with

most frequent items:
             SIPO     UROK (Other)
  11206      9219     9219    9219

most frequent elements:
      {}  {SIPO}  {UROK} (Other)
  11206     9219    9219    9219

element (sequence) size distribution:
sizes
   1    2    3    4    5    6    7    8    9   10
   3    9   23   55  127  286  630 1357 2860 5915

sequence length distribution:
lengths
   1    2    3    4    5    6    7    8    9   10
   3    9   23   55  127  286  630 1357 2860 5915

summary of quality measures:
    support
 Min.   :0.5039
 1st Qu.:0.5898
 Median :0.6334
 Mean   :0.6428
 3rd Qu.:0.7003
 Max.   :1.0000

includes transaction ID lists: TRUE

mining info:
                       data ntransactions nsequences support
 data_for_fseq_mining_trans        675886       3584     0.5
```

Figure 5-47: Summary of analysed sequences

Figure 5-47 above shows a command summary with the following descriptive analysis results:

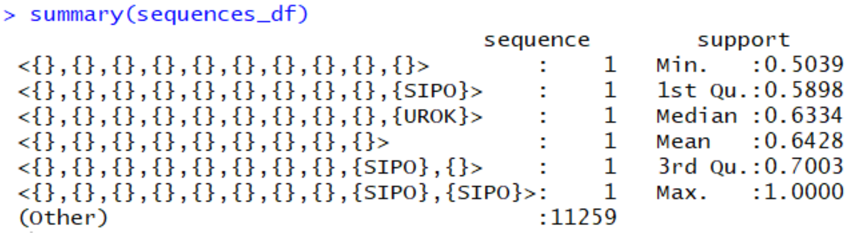
(1) the list of the most frequent isolated payment categories: blanks with 11,206 items, SIPO with 9,219 items, UROK with 9,219 items and (others) with 9,219 items.

(2) the list of the most frequent set of payment categories that occur in transactions referred to elements such as { }, {SIPO}, {UROK} and (others).

(3) the distribution of the sequence sizes of the set of payment categories: the distribution of the sequence sizes is skewed to the right. A total of 11,265 sequences can be divided into ten buckets, in which the majority of the discovered sequences are aligned to the 8, 9 and 10. For instance, bucket 1 comprises only three sequences.

(4) the distribution of the number of transactions in a sequence is referring to the following sequence length: bucket 10 comprises sequences with the highest length of 5,915 transactions.

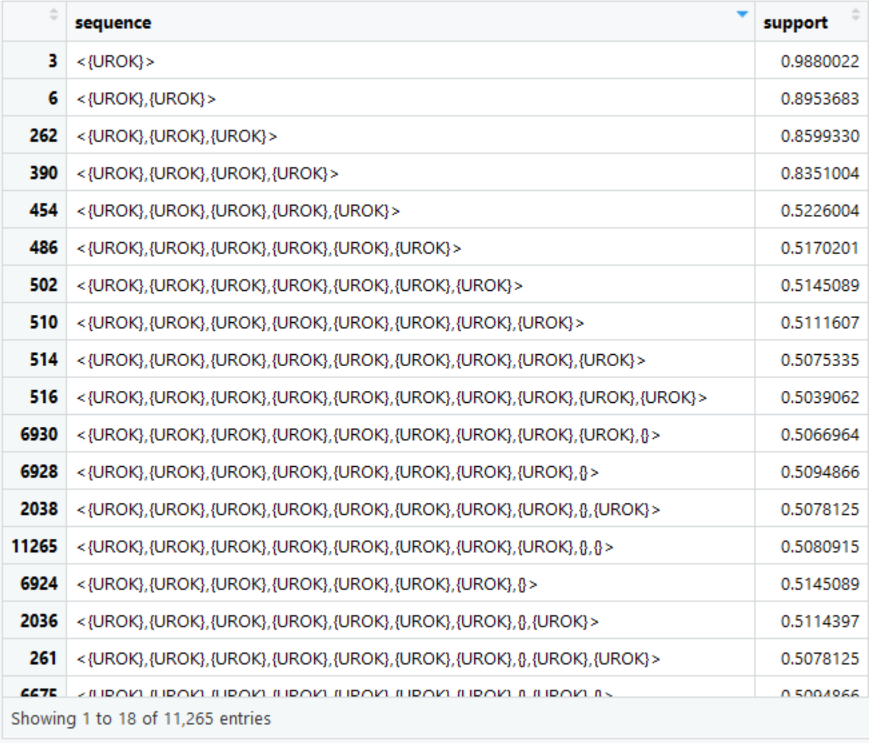
Figure 5-48 below summarises the results of the pre-processed data frame “sequences_df” and displays quality measures for the minimum, maximum, mean and median support values. It shows also the relative support of the estimated sequences.



```
> summary(sequences_df)
                                    sequence        support
 <{},{},{},{},{},{},{},{},{},{}>        :    1   Min.   :0.5039
 <{},{},{},{},{},{},{},{},{},{SIPO}>    :    1   1st Qu.:0.5898
 <{},{},{},{},{},{},{},{},{},{UROK}>    :    1   Median :0.6334
 <{},{},{},{},{},{},{},{},{}>           :    1   Mean   :0.6428
 <{},{},{},{},{},{},{},{},{SIPO},{}>    :    1   3rd Qu.:0.7003
 <{},{},{},{},{},{},{},{},{SIPO},{SIPO}>:    1   Max.   :1.0000
 (Other)                                :11259
```

Figure 5-48: Summary results of sequences and relative support

The results show that there are 11,259 unique sequences with a median of 0.6334 relative support given in the dataset. Below table 5-3 provides an overview of the calculated sequences and its relative support values in descending order.



	sequence	support
3	<{UROK}>	0.9880022
6	<{UROK},{UROK}>	0.8953683
262	<{UROK},{UROK},{UROK}>	0.8599330
390	<{UROK},{UROK},{UROK},{UROK}>	0.8351004
454	<{UROK},{UROK},{UROK},{UROK},{UROK}>	0.5226004
486	<{UROK},{UROK},{UROK},{UROK},{UROK},{UROK}>	0.5170201
502	<{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK}>	0.5145089
510	<{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK}>	0.5111607
514	<{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK}>	0.5075335
516	<{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK}>	0.5039062
6930	<{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{}>	0.5066964
6928	<{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{}>	0.5094866
2038	<{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{},{UROK}>	0.5078125
11265	<{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{},{}>	0.5080915
6924	<{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{}>	0.5145089
2036	<{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{},{UROK}>	0.5114397
261	<{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{},{UROK},{UROK}>	0.5078125
6675	<{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{UROK},{},{UROK},{}>	0.5094866

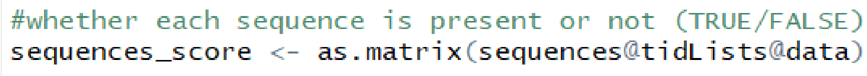
Showing 1 to 18 of 11,265 entries

Table 5-3: View of the table - sequences and relative support

The research experiment has observed that the cSPADE algorithm found many trivial sequences from customer behaviour. For example, it has found many unitary sequences, such as <{UROK}>, <{UROK},{UROK}>, <{UROK},{UROK},{UROK}>, among others. These unitary sequences are really frequently used, but they may not be useful in the particular application of identifying sustainable bank product recommendations. Our outlined research project was to suggest new products to bank customers with the objective of enhancing their cross-selling activities and enrich customer behavioural insights.

Based on the analysis outcome, I can also observe that many frequent patterns are related to 'UROK', which are also one of the most popular tags in the whole recommendation system, as can be discovered.

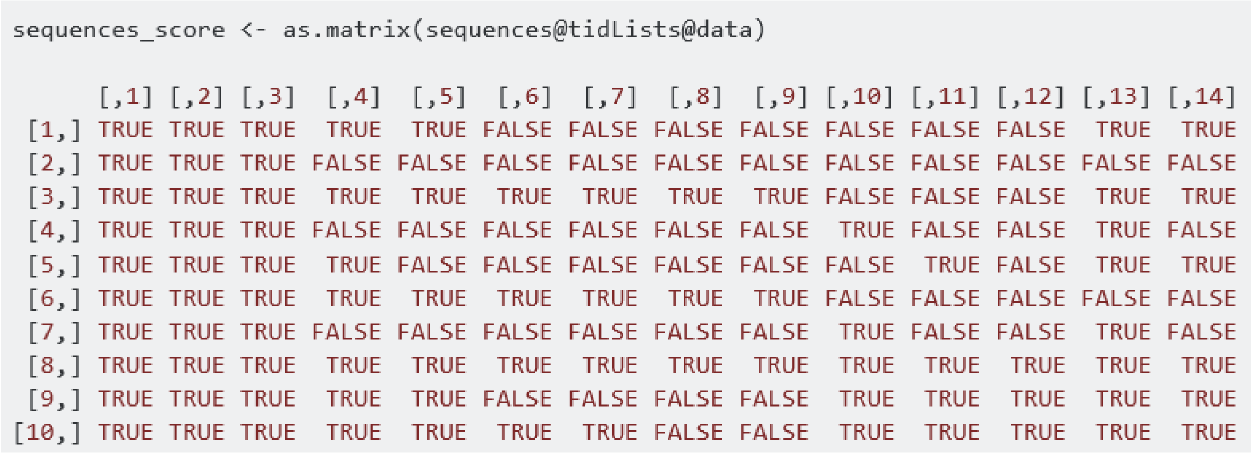
Below R-script snippet illustrate how a sequence matrix for each existing transaction, whether each sequence is present (true) or not (false), can be calculated.



```
#whether each sequence is present or not (TRUE/FALSE)
sequences_score <- as.matrix(sequences@tidLists@data)
```

Figure 5-49: R-code snippet for calculating sequences scores

A best-practice output is a visualised "tidList" matrix shown – for example – in figure 5-50 below. However, due to the extraordinary amount of the discovered sequences length, the cSPADE algorithm must be executed on a workstation with more computational power or a mainframe computer, as the search space of all 11,265 discovered sequences is extremely large to solve this complex problem with the current research setup.



```
sequences_score <- as.matrix(sequences@tidLists@data)

      [,1] [,2] [,3]  [,4]  [,5]  [,6]  [,7]  [,8]  [,9] [,10] [,11] [,12] [,13] [,14]
 [1,] TRUE TRUE TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE
 [2,] TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [3,] TRUE TRUE TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE  TRUE  TRUE
 [4,] TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE  TRUE FALSE
 [5,] TRUE TRUE TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE  TRUE
 [6,] TRUE TRUE TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE
 [7,] TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE  TRUE FALSE
 [8,] TRUE TRUE TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
 [9,] TRUE TRUE TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE
[10,] TRUE TRUE TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE
```

Figure 5-50: Summary results - Matrix of sequences

5.2. Predictive results of individual research projects

The section discusses the predictive and prescriptive evaluation results of the thesis, including a comprehensive description of the analysis results in terms of predicting future payment behaviour (e.g. credit scoring and cross-selling opportunities) based on a historical dataset from previous payment transaction streams and forecasting categorised payment transactions. The presented predictive results are built also on the outcomes of the descriptive findings from the previous section by presenting possible cost-sensitive data models and their corresponding prediction performance outcomes.

The subsequent sections provide a detailed overview of the predictive and prescriptive analysis results across the various research projects. It offers foresight into what is likely to occur (e.g. good or bad credit scores, cross-selling recommendations and unknown categorised payment transactions) using supervised, unsupervised techniques and statistical analysis introduced in the previous chapters to predict future outcomes and perform the advanced datasets for the first two research projects for optimising predictive performance and eliminating cost-intensive (unnecessary) variables within the pre-processed datasets identified during the diagnostic analysis.

Regarding the previous chapter, the following sections will also describe the predictive results of the introduced procedures for developing different forecast models. Thus, in order to achieve sustainable results, it's important to continuously test and monitor the model performance and to optimise both the design of the machine learning model and the given dataset by data wrangling and/or cleaning (e.g. dimensionality reduction, clustering) as well as development of new enhanced classification algorithms (e.g. develop fine-tuned supervised machine learning models) in case of credit scoring or cross-selling intentions.

5.2.1. Modelling results of the credit scoring case

The section describes the overall evaluations and findings made by predicting reliable credit scores for bank customers. Therefore, the following paragraphs present the detailed predictive results of the individual research project, in which the assessment of the performance measurement outcomes from the applied machine learning algorithm using the MS Azure ML library is at the forefront. One of the research objectives is to figure out the best-fitting applied machine learning algorithm(s) based on their performance outcomes and seek to further optimise their prediction results from a data-driven cost model perspective.

The following figure is an extract of the common MS Azure Mini Map, which provides a clear structured overview of the extensive range of machine learning algorithms I have applied to examine the raised research questions in detail.

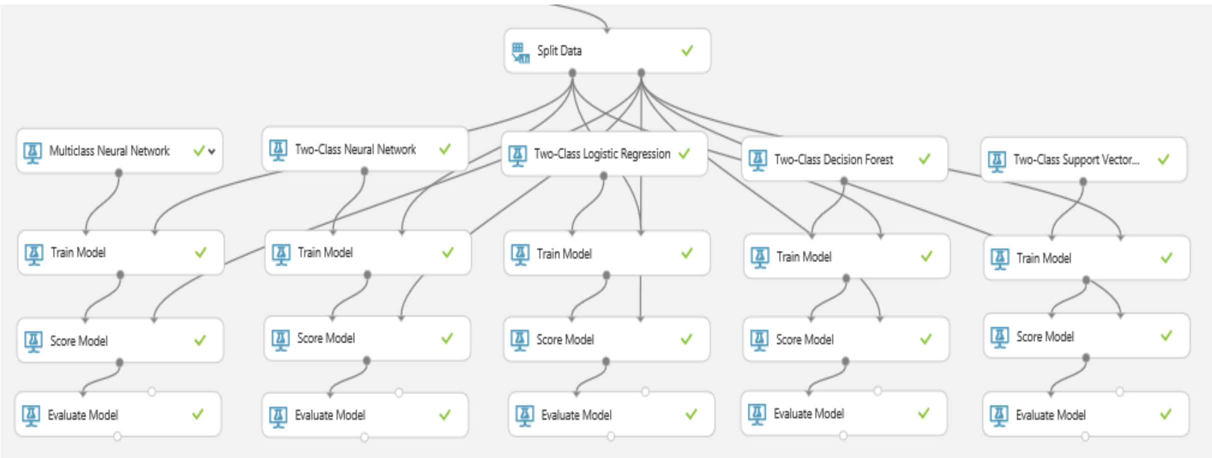


Figure 5-51: MS Azure's Mini Map overview for the applied supervised learning algorithms - extract of the credit scoring research experiment

After splitting the pre-processed data for the credit scoring case into train and test data, the individual algorithm is linked to the separate ‘Train Model’ modules to score the trained model and evaluate their performance based on a scored classification or regression model with advanced metrics. The established research experiments enable reliable statements to be made about the behaviour of the individual supervised learning algorithms when applied to the particular pre-processed dataset.

Regarding the proposed research design, the research experiments for the first research project demonstrate the usage of the ‘Multiclass Neural Network’ module to train neural network, the ‘Two-Class Neural Network’ module to train neural network, the ‘Two-Class Logistic Regression’ module to train the logistic regression model, the ‘Two-Class Decision Forest’ module to train the decision forest algorithm and the ‘Two-Class Support Vector Machine’ module to train the support vector machine algorithm. Many different machine learning algorithms have been applied to investigate their model performance, but there is no single measure available which is perfect for all credit score cases. To objectively judge the performance of an algorithm, I must be able to measure it. Therefore, MS Azure features provide a variety of performance metrics which I have used to compare the predictive results more precisely. Appendix

D.1 and D.2 contains an exhaustive list of model performance results including their visualisations deployed by MS Azure functions.

Regarding our pre-processed dataset, I also face the challenge with the dominant category 'status' (e.g. 89% of the samples are assigned to a 'good scoring' class A), pretending that all samples are belonging to a 'good scoring' class A will result in an accuracy of 89%. To achieve most accurate result, the research design considers along the data preparation phase that the underlying processed dataset is rebalanced to approximately 50/50. However, the evaluated performance is measured objectively since the managed test data was kept away from all steps of the model training. Note that there is a vast of literature describing the accuracy paradox in detail. The evaluation of the applied supervised learning algorithms has resulted in a variety of performance outcomes summarised in the table below. Thus, a more detailed view on the evaluation results for the predictive models corresponding to the table below is given in appendix D.2. Note that the aim of the research experiments is to gauge the performance of the applied algorithm with respect to the input parameters of the predictive model. In a further step towards an optimised cost-sensitive model, the best-performing algorithm will be used for advanced research experiments. Detailed analysis results will be discussed at far end of this section.

The following table 5-4 describes an objective performance comparison of the evaluation results when the various applied machine learning algorithms are measured at the 0.5% threshold. To measure and select the best-performing model, I have included the following performance characteristics into the overall metric provided by MS Azure ML:

True positive / true negative: Each value returns the number of credit applicants which will probably rate with good or bad credit scores correctly. Visualised results (lift curves) for the number of true positives plotted against positive rate are given in appendix D.2.

False positive / false negative: Each value returns the number of credit applicants which will result in business costs for the bank due to the given probability of a credit default.

The accuracy value: It describes the degree of closeness of the calculated credit scores to its actual value. The returned value shows how accurate the algorithm has differentiated (scored) between good and bad credit applicants.

The precision value: It describes how precise as well as accurate the specific credit scoring model is out of those predicted positives consisting of true positives and false positives, how many of them are actual positive. For instance, a false positive in the credit scoring model means that a credit applicant with a bad scoring (actual negative) has been identified as a credit applicant with a good scoring (predicted default credits). Banks might have credit losses if the precision of the credit scoring is not high.

The recall value: It describes how many of the actual positives the credit scoring model capture when labelling them as positive (true positive). For instance, if a credit applicant with a bad scoring (actual positive) classified as a credit applicant with a good scoring (predicted negative), the consequence can be result in significant losses for the bank.

Algorithm applied by a threshold of 0.5	True Positive	True Negative	False Positive	False Negative	Accuracy	Precision	Recall	F1 Score	AUC
Multiclass Neural Network	81.6%	87.6%	18.4%	12.4%	0.853	0.853	0.853		
Two-Class Neural Network	109	60	16	12	0.858	0.872	0.901	0.886	0.910
Two-Class Logistic Regression	105	58	18	16	0.827	0.854	0.868	0.861	0.903
Two-Class Decision Forest	**109**	**67**	**9**	**12**	**0.893**	**0.924**	**0.901**	**0.912**	**0.956**
Two-Class Support Vector Machines	102	54	22	19	0.792	0.823	0.843	0.833	0.869
Max	109	87.6	22	19	0,893	0,924	0,901	0,912	0,956
Min	81.6	54	9	12	0,792	0,823	0,843	0,833	0,869
Average	101.3	65.32	16.68	14.28	0,844	0,865	0,873	0,873	0,909

Table 5-4: Overview of the evaluation results of all applied algorithms - credit scoring

The F1-score value: It provides a balanced score / weighted average between the precision and recall measurements. Visualised results (Precision vs. Recall curves) for the precision against recall relationship are given in appendix D.2.

AUC (area under the curve): The values helps to assess the diagnostic accuracy of credit scores for the credit applicants. The closer the value of AUC is to 1, the better is the accuracy of the predicted values.

The delineated performance measures above are important to answer the arises research question about how good the predictions really are, which the research has produced through its customised research experiments. Hence, the following paragraphs are explaining the results of the different models that I have built through the conducted research experiments.

First of all, it has to be stressed that one of the best modelling results is generally shown by applying the Two-Class Decision Forest algorithm, but when taking a deeper look into the performance criteria the research ascertained that there are some fine distinctions in the evaluation results which needs to be more detailed.

The first differences become even clearer when all true positives and true negatives of every applied algorithm are observed in relation to their correctly predicted credit scores (good or bad). Regarding the predicted scorings composed of true positive and true negative values, the following performance summary may be given. The Two-Class Decision Forest model indicates to predict a maximum value of 176 cases correctly, followed by Two-Class Neural Network model with a total of 169 predicted scores, Two-Class Logistic Regression model with 163 predicted scores and Two-Class Support Vector Machines model with a number of 156 cases.

It is a fallacy to assume that a high accuracy of any applied machine learning algorithm is the baseline to assess the model is best. The accuracy is a great measure in our performance metric, but not sufficient due to our asymmetric dataset. Therefore, I am also looking at the other parameters to obtain a unified assessment of the evaluation results. For the Two-Class Decision Forest model, the research experiments have achieved the best results with 0.893 which means the model is approximately 89.3% accurate. The Two-Class Neural Network model has achieved the second-best result with 85.8% accuracy, and the poorest result is returned by Two-Class Support Vector Machines model with an accuracy of 86.9%.

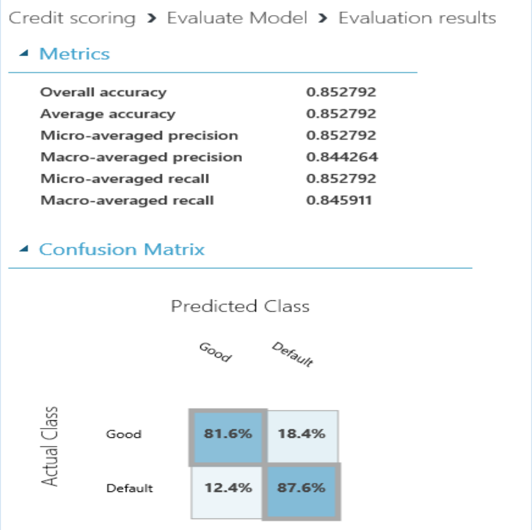
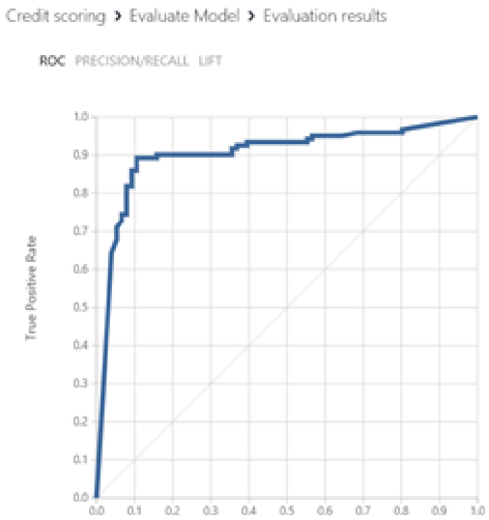
The question that the returned precision value answers is of all selected credit applicants that labelled with good credit scorings how many of them are actually good credit applicants. The Two-Class Decision Forest algorithm returns with nine false positive cases, which is the lowest positive rate compared with the remaining algorithm, one of the best results. The algorithm returns 0.924 precision which is pretty good compared with the second-best result with 0.872 by applying a Two-Class Neural Network.

The question that the returned recall value answers is of all selected credit applicants that truly rated as good, finally, how many of them did the algorithm really identified. The highest recall results with 0.901 are achieved by both algorithm the Two-Class Neural Network and Two-Class Decision Forest model, which is good as it’s at least above the threshold of 0.5. The Two-Class Logistic Regression model returns with 0.868 the second-best result.

The F1-scores provides a more useful measure than the accuracy value since all generated false positives and false negatives are very different for every applied machine learning algorithm. This is also the reason why current research has also included the precision and recall measurements into the performance metric to better figure out how good the model really has performed. It proved that the Two-Class Decision Forest model returns the best result with a F1-score of 0.912.

Following receiver operating characteristic (ROC) curves illustrates the performance outcomes when plotting the true positive rate against the false positive rate. The results of the ROC curves whether it is good or not can be determined by looking at AUC (Area Under the Curve) values which also measures the accuracy value. I have classified the accuracy based on the following measurement ranges related to the traditional academic point system: .90-1 = excellent (A); .80-.90 = good (B); .70-.80 = fair (C); .60-.70 = poor (D) and .50-.60 = fail (F). Based on that, the AUC value clearly shows that excellent measurements can be achieved with Two-Class Decision Forest model (0.956), Two-Class Neural Network model (0.910) and Two-Class Logistic Regression model (0.903). Good accuracy measurements can be generated by Two-Class Support Vector Machines model (0.869).

Finally, the confusion matrix of the multiclass neural network shows that the probability of correctly predicted positive values is 81.6% and negative values is 87.6% with an overall accuracy value of 0.852. Another conclusion that can be drawn from the prediction results is that the high percentage of the true negatives is an indicator of predicting credit applicants with bad scorings much precisely than with good scorings. The prediction accuracy is valued at 85.2%.

ROC curve algorithm	Response charts
Multiclass Neural Network	
Two-Class Neural Network	

Credit scoring › Evaluate Model › Evaluation results

Metrics

Overall accuracy	0.852792
Average accuracy	0.852792
Micro-averaged precision	0.852792
Macro-averaged precision	0.844264
Micro-averaged recall	0.852792
Macro-averaged recall	0.845911

Confusion Matrix

Predicted Class

Actual Class

	Good	Default
Good	81.6%	18.4%
Default	12.4%	87.6%

Credit scoring › Evaluate Model › Evaluation results

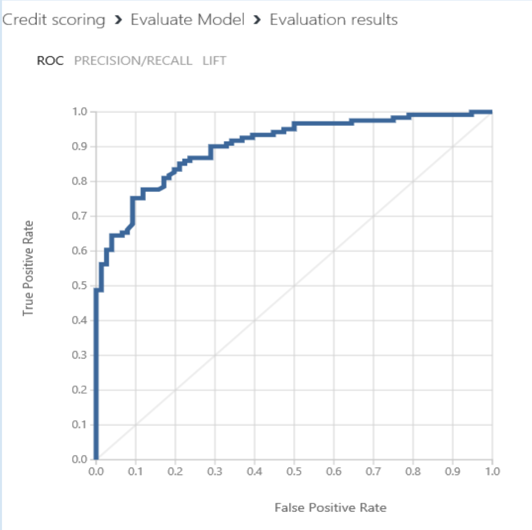
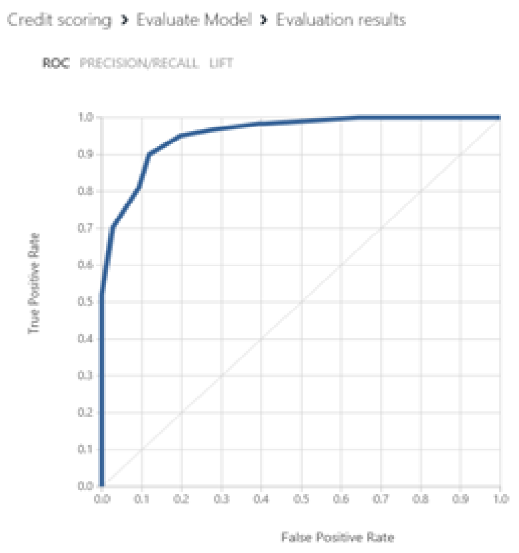
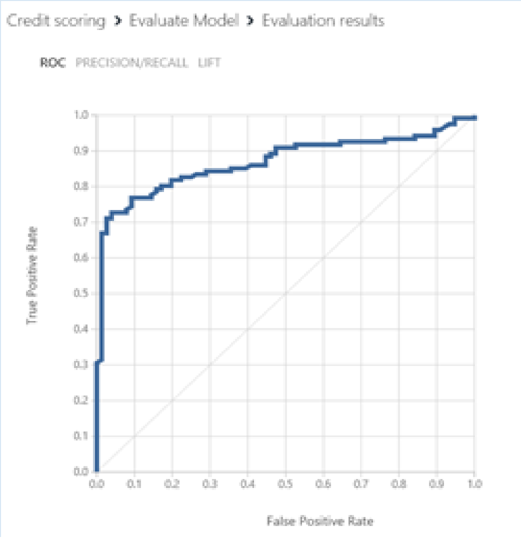
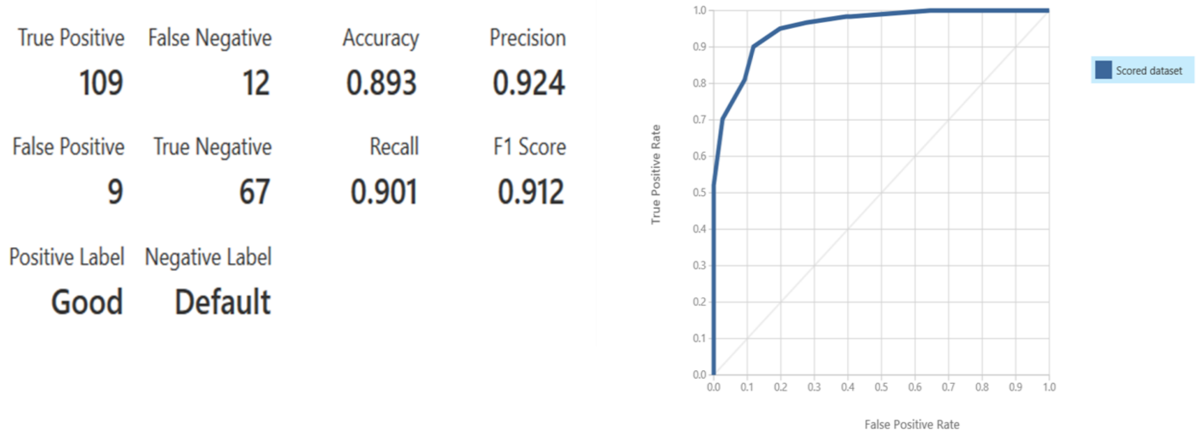
Two-Class Logistic Regression	
Two-Class Decision Forest	
Two-Class Support Vector Machines	

Table 5-5: Overview of the response charts of all applied algorithms - credit scoring

Both tables 5-4 and 5-5 above describe the performance comparisons of the applied clustering and classification algorithms. The overall analysis results show that the research is able to achieve better performance with the used supervised learning algorithm compared with the existing methods applied on the PKDD financial datasets in a different research context. The research used various data visualising features of MS Azure ML to deep dive into a range of distinct response charts for every scored dataset.

The best plotted response chart shown in the overview table above is the ROC plot of the Two-Class Decision Forest. The performance results of the Two-Class Logistic Regression are also returning an appropriate picture of the predicted values presented in the table 5-5 above. However, if we take a closer look at the AUC value, however, the accuracy measurement returns a value of 0.903. Compared with this, the response chart of the Two-Class Neural Network looks less pleasant, but the corresponding AUC value displayed with 0.910 is measurably better than the Two-Class Logistic Regression performance outcomes. The evaluation results of the Two-Class Support Vector Machine algorithm also generate a relatively good response chart including all its rough edges.

However, it can be easily observed that, ultimately, the algorithm Two-Class Random Forest plots the best ROC response chart based on their prediction results. In figure 5-52 below, the detailed evaluation results of the credit scoring model are presented on the left and their corresponding ROC response chart on the right. The visualisation helps to verify that the AUC is indeed equal to 0.956 as shown in the summary overview above. The grey line represents a completely uninformative test, which will be returned by an AUC threshold of 0.5. The curve is pulled close to the upper left corner which indicates that the applied Two-Class Decision Forest algorithm is performing best.



True Positive 109	False Negative 12	Accuracy 0.893	Precision 0.924
False Positive 9	True Negative 67	Recall 0.901	F1 Score 0.912
Positive Label Good	Negative Label Default		

Figure 5-52: Summary of the best modelling results for credit scoring achieved by Two-Class Decision Forest algorithm

The research work further studies the scalability as the conducted research experiments also changes the size of the threshold in five ways by keeping all other performance metric parameters constant. The constant performance metric parameters for all five experiments were true positive, true negative, false positive, false negative, accuracy, precision, recall and F1-score, and the threshold was varied from 0 to 1.

Table 5-6 below shows how the algorithm scales with the five different threshold parameters. For higher values of thresholds (>0.5) the predicted scorings consisting of true positive and true negative decrease with increasing maximal threshold size. This is due to the fact that the recall values are falling dramatically to a maximum of 0.521 the higher the threshold is. The consequence is also that the algorithm will discover fewer false positives and an increased number of false negatives. For lower values of the threshold, however, a larger number of predicted scorings (true positive + true negative) appears, thus the predicted scorings start to increase initially, but then decreases again when the threshold is greater than 0.5 due to the same reasons given above. The peak occurs at roughly the threshold value of 0.5 for the research experiments.

Finally, the table 5-6 below summarises the prediction results through the selected performance characteristics for the best-performing Two-Class Decision Forest algorithm applied on different thresholds.

Threshold	True Positive	True Negative	False Positive	False Negative	Accuracy	Precision	Recall	F1 Score	AUC
0	121	26	50	0	0.746	0.708	1	0.829	0.956
0.25	117	55	21	4	0.873	0.848	0.967	0.903	0.956
0.5	109	67	9	12	0.893	0.924	0.901	0.912	0.956
0.75	85	74	2	36	0.807	0.977	0.702	0.817	0.956
0.99	63	76	0	58	0.706	1	0.521	0.685	0.956

Table 5-6: Summary of the evaluation results of the scaled Two-Class Decision Forest - credit scoring

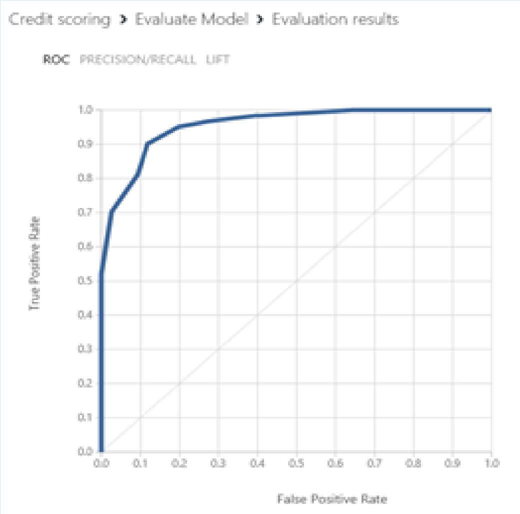
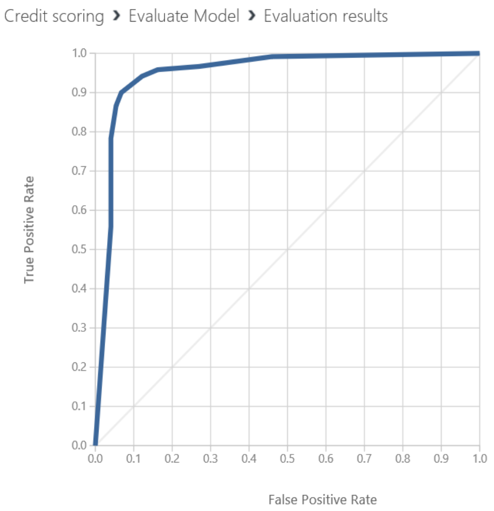
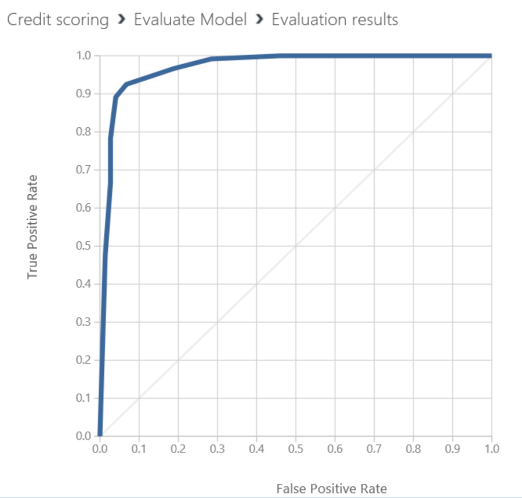
The following paragraphs are investigating the research results based on the Two-Class Decision Forest model to stress out the performance outcomes of the applied algorithm when executing the model on different (advanced) datasets. Therefore, I have instituted three different research experiments by running the model again with the relevant variables deducted from the descriptive analysis and compare the performance results as well as response charts with the original response chart (incl. all “unnecessary” input variables). Regarding the theoretical concepts of applying supervised and unsupervised learning algorithm described in chapter 2 and 4, an optimised (perfect) model fit requires finally a model with sufficient parameters and good prediction results not only depends on well-fitted models. The established research design and its analysis outcomes will provide the evidence of the statement above.

The table 5-7 below describes the evaluation results for exploring a cost-sensitive data model effected based upon five customised research experiments which are aligned to the key findings from the variable importance analysis in the previous section. First, research experiment is performed on the original pre-processed dataset, the second experiment is carried out on the original dataset excluding the variables ‘cardholder’ and ‘sex’, the third research experiment is executed on an optimized dataset which consists of the variable ‘AvgBalance’, ‘AvgIncome’, ‘AvgExpenses’ and ‘Age’, the fourth research experiment is executed on an optimized dataset which consists only of the variable ‘AvgBalance’, ‘AvgExpenses’, ‘Age’, and the last research experiment is executed on an optimised dataset which consists only of the variable ‘AvgBalance’, ‘AvgIncome’, ‘Age’.

Algorithm applied by a threshold of 0.5	True Positive	True Negative	False Positive	False Negative	Accuracy	Precision	Recall	F1 Score	AUC
Two-Class Decision Forest applied on the original dataset	109	67	9	12	0.893	0.924	0.901	0.912	0.956
Two-Class Decision Forest applied on the optimised dataset (excl. Cardholder and sex)	108	69	5	12	0.912	0.956	0.900	0.927	0.939
Two-Class Decision Forest applied on the optimised dataset (incl. AvgBalance, AvgIncome, AvgExpenses, Age)	107	71	3	13	0.918	0.973	0.892	0.930	0.967
Two-Class Decision Forest applied on the optimised dataset (incl. AvgBalance, AvgExpenses, Age)	107	71	3	13	0.918	0.973	0.892	0.930	0.967
Two-Class Decision Forest applied on the optimised dataset (incl. AvgBalance, AvgIncome, Age)	107	71	3	13	0.918	0.973	0.892	0.930	0.967

Table 5-7: Comparing evaluation results of applied Two-Class Decision Forest - original vs. optimised dataset for credit scoring

I have observed through the customised research experiments that the supervised learning algorithm Two-Class Decision Forest is using an overwhelming number of input variables, many of them trivial or useless. However, the dataset does contain all potentially useful predictor variables to generate acceptable results.

ROC curve algorithm	Response charts
Two-Class Decision Forest applied on the original dataset	 Credit scoring › Evaluate Model › Evaluation results
Two-Class Decision Forest applied on the optimised dataset (excl. Cardholder and sex)	 Credit scoring › Evaluate Model › Evaluation results
Two-Class Decision Forest applied on the optimised dataset (incl. AvgBalance, AvgIncome, AvgExpenses, Age)	 Credit scoring › Evaluate Model › Evaluation results

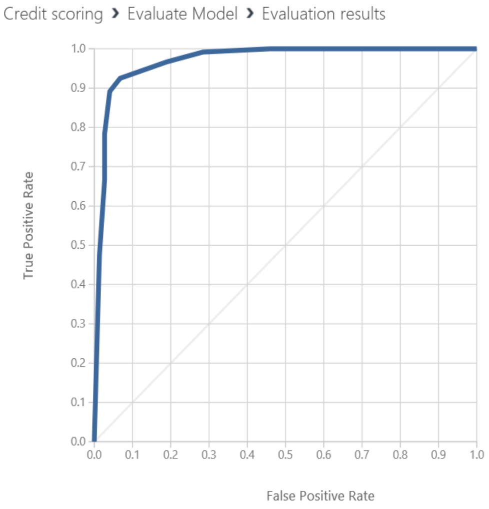
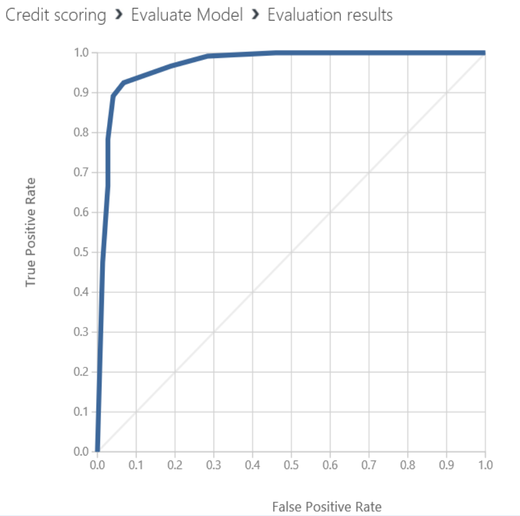
Two-Class Decision Forest applied on the optimised dataset (incl. AvgBalance, AvgExpenses, Age)	 Credit scoring › Evaluate Model › Evaluation results
Two-Class Decision Forest applied on the optimised dataset (incl. AvgBalance, AvgIncome, Age)	 Credit scoring › Evaluate Model › Evaluation results

Table 5-8: Overview of the response charts of applied Two-Class Decision Forest - original vs. optimised dataset for credit scoring

The tables 5-7 and 5-8 above reveal that the applied algorithm will generate best predictions (AUC is 0.967) when the predictive model is built only on three to four important variables ‘AvgBalance’, ‘AvgIncome’, ‘AvgExpenses’ and ‘Age’. The research experiments three, four and five are generating identical performance results, although the applied flattened correlation matrix from the previous section shows that the variable ‘AvgIncome’ and ‘AvgExpenses’ are correlating (-0.997) very strongly. Poor performance results (AUC is 0.939) will be produced in the second research experiment and the second-best performance outcomes are achieved on the original dataset.

The current research experiments show that for sophisticated model input parameters the response charts result in remarkable performance graphs. The curve for the research experiments three, four and five is pulled close to the upper left corner which indicates that the applied Two-Class Decision Forest algorithm is performing best on an optimised dataset which includes only the selected variables. However, the detailed evaluation results among the five designed research experiments underlines that some form of post-processing in case of executing the varImp() function through the data pre-processing phase is necessary to weed out the irrelevant variables and to locate the most significant input variables for the predictive model. Finally, the research results show that the best cost-sensitive data model must be deployed in an iterative process.

The research also showed that the output of the supervised learning algorithm can be used to build scalable monitors which predict failures in case of credit applicants with bad scorings in a plan before they really occur. The research was able to produce monitors with a 97.3 % precision that approximately 120 out of 194 credit applicants are actual positives (+61.85%) that occur.

5.2.2. Modelling results of the cross-selling case

The section describes the overall evaluations and findings made by predicting reliable cross-selling candidates for bank customers. For the cross-selling research project it is interesting to ascertain which supervised learning and classification algorithm performs the best predictions to identify potential credit card customer more precisely, so the research compared the performance outcomes of all applied machine learning algorithm using MS Azure ML library. Doing this, the data model also included customer with and without a credit card.

The goal of this research project was to identify the highest performing machine learning algorithm for predicting a credit card ownership. The prediction models are computing a score which represents the most likely cross-sell candidates for promoting a credit card. Van der Putten (1999) recognised that “simply predicting yes or no for ‘owns credit card’ does not suffice for most direct marketing purposes.” Therefore, the following paragraphs also present out the best-fitting applied machine learning algorithm(s) based on their performance outcomes and seek to further optimise their prediction results from a data-driven cost model perspective.

The following figure 5-53 is an extract of the common MS Azure Mini Map, which provides a clear structured overview of the extensive range of machine learning algorithms I have applied to examine the second research project in detail.

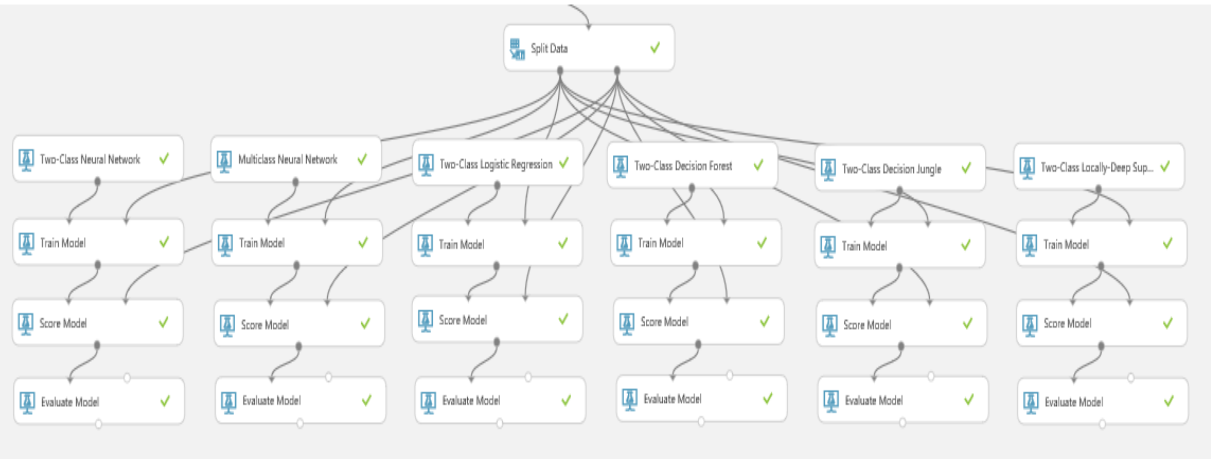


Figure 5-53: MS Azure's Mini Map overview for the applied supervised learning algorithms - extract of the cross-selling research experiment

After splitting the pre-processed data for the cross-selling case into train and test data, the individual algorithm is linked to the separate ‘Train Model’ modules to score the trained model and evaluate their performance based on a scored regression or classification model with advanced metrics. The established research experiments enable reliable statements to be made about the performance of the individual supervised learning algorithms when applied to the particular pre-processed dataset.

Regarding the proposed research design, the research experiments for the second research project demonstrates the usage of the ‘Multiclass Neural Network’ module to train neural network, the ‘Two-Class Neural Network’ module to train neural network, the ‘Two-Class Logistic Regression’ module to train the logistic regression model, the ‘Two-Class Decision Forest’ module to train the decision forest algorithm, the ‘Two-Class Decision Jungle’ module to train the decision forest algorithm and the ‘Two-Class Locally-Deep Support Vector Machine’ module to train the support vector machine algorithm. Many different machine learning algorithms have been applied to investigate their model performance, but there is no single measure available which is perfect for all cross-selling cases. To increase the objectivity of the performance results for every single algorithm requires precise measurement of the applied machine learning algorithm used. Thus, MS Azure features provide a variety of performance metrics which current research has been utilised to compare the forecasting results more precisely. Appendix D.3 and D.4 contains an exhaustive list of model performance results including their visualisations deployed by MS Azure functions.

With respect to the pre-processing phase regarding the underlying dataset, I will again have to face the challenge with an imbalanced dataset due to the slightly dominant category ‘cardholder’ (e.g. 75% of samples (621 elements) is assigned to a ‘non-cardholder’ class 0 and 25% of samples (206 elements) is assigned to a ‘cardholder’ class 1), supposing that all samples are belonging to a ‘non-cardholder’ class 0 will result in an 75% accuracy. Taken this into the account through the data pre-processing stage, I have rebalanced the dataset to achieve most accurate results which can be reliably interpreted in the context of the settled research experiments. However, the assessed performance is measured objectively since the managed test data was separated from all steps of the model training. The evaluation of the applied machine learning algorithms has resulted in a variety of performance outcomes summarised in the table 5-9 below. A more detailed view on the model performance measures for the predictive models corresponding to the table 5-9 below is given in appendix D.4. Note that the aim of the research experiments is again to analyse the performance of the used algorithm with respect to the selected input parameters of the predictive model. The best-performing algorithm will be used for advanced research experiments at a later stage when the research work wants to develop an optimised cost-sensitive model. Detailed evaluation results will be presented at far end of this section.

The table 5-9 below describes an objective performance comparison of the evaluation results when the various applied supervised learning algorithms are initially measured at the 0.5% threshold. The research applied a fixed threshold of 0.5 to secure the comparability of the evaluation results and objectively measure the best-performing model based on the following performance characteristics of the overall metric provided by MS Azure ML:

True positive / true negative: Each value returns the number of cross-selling candidates which will probably calculated correctly. Visualised results (lift curves) for the number of true positives plotted against positive rate of all applied algorithms are given in appendix D.4.

False positive/false negative: Each value returns the number of cross-selling candidates which will result in unnecessary business costs for the bank due to the increased probability of an inefficient targeted marketing campaign.

The accuracy value: Describes the degree of closeness of the computed scores for cross-selling candidates to its actual value. The returned value shows how accurate the algorithm has differentiated (scored) between existing and non-existing cross-sell opportunities.

The precision value: Describes how precise as well as accurate the specific cross-selling model is out of those predicted positives consisting of true positives and false positives, how many of them are actual positive. For instance, a false positive in the cross-selling model means that a cross-selling candidate labelled as bad (actual negative) has been identified as a cross-selling candidate labelled as good (predicted default cross-selling opportunities). Banks might have excessively marketing costs if the precision of the cross-selling score is not high. The performance characteristic is a good measure to determine, when the costs of false positive is high.

The recall value: Describes how many of the actual positives the cross-selling model capture when labelling them as good (true positive). For instance, if a cross-selling candidate labelled as bad (actual positive) classified as a cross-selling candidate labelled as good (predicted negative), the consequence can be result in tremendous marketing costs for the bank by promoting new product offerings.

Algorithm applied by a threshold of 0.5	True Positive	True Negative	False Positive	False Negative	Accuracy	Precision	Recall	F1 Score	AUC
Multiclass Neural Network	67.5%	79.0%	32.5%	21.0%	0.72	0.72	0.72		
Two-Class Neural Network	64	88	38	17	0.734	0.627	0.79	0.699	0.806
Two-Class Logistic Regression	45	106	20	36	0.729	0.692	0.556	0.616	0.814
Two-Class Decision Forest	**64**	**103**	**23**	**17**	**0.807**	**0.736**	**0.79**	**0.762**	**0.876**
Two-Class Decision Jungle	69	95	31	22	0.744	0.656	0.728	0.69	0.837
Two-Class Locally-Deep Support Vector Machines	62	98	28	19	0.733	0.689	0.765	0.725	0.816
Max	69	106	38	36	0,807	0,736	0.79	0,762	0,876
Min	45	88	20	17	0.72	0,627	0,556	0,616	0,806
Average	60.8	98	28	22.2	0,744	0,686	0,724	0,698	0,829

Table 5-9: Overview of the evaluation results of all applied algorithms - cross-selling

The F1-score value: It provides a weighted average between the precision and recall measurements. Visualised results (Precision vs. Recall curves) for the precision against recall relationship are given in appendix D.4.

AUC (area under the curve): The values helps to assess the diagnostic accuracy of cross-selling candidates for the marketing campaign. The closer the value of AUC is to 1, the better is the accuracy of the predicted values.

The outlined performance measures above are important to answer the arises research question about how good the predictions for cross-selling opportunities really are, which the research has produced through its customised research experiments.

Hence, the following covers the evaluations results of the different models that I have built through the conducted research experiments.

First of all, it should be stressed that one of the best modelling results is generally shown by using the Two-Class Decision Forest algorithm, but when taking a closer look into the introduced performance criteria the research ascertained that there are some fine distinctions in the evaluation results which needs to be deepen more clearly.

The first differences occur when all true positives and true negatives of every applied algorithm are observed in relation to their correctly predicted cross-selling candidates (yes or no). Regarding the predicted headcounts composed of true positive and true negative values, the following performance summary may be given. The Two-Class Decision Forest model indicates to predict a maximum value of 167 candidates correctly, followed by Two-Class Decision Jungle model with a total of 164 predicted candidates, Two-Class Locally-Deep Support Vector Machines model with 160 predicted candidates, Two-Class Neural Network model with 152 candidates and Two-Class Logistic Regression model with a number of 151 candidates.

The accuracy value is a suitable measure in our performance metric, although it is not sufficient due to our imbalanced dataset. Therefore, I have also included further parameters into the assessment of the prediction outcomes. For the Two-Class Decision Forest model, the research experiments have performed the best results with 0.876 which means the model is approximately 87.6% accurate. The Two-Class Decision Jungle model has achieved the second-best result with 83.7% accuracy, and the poorest result is returned by Two-Class Neural Network model with an accuracy of 80.6%.

The returned precision value answers the question whether all selected cross-selling candidates that labelled as good how many of them are actually good cross-selling candidates. The Two-Class Logistic regression algorithm returns with 20 false positive candidates, which is the lowest positive rate compared with the remaining algorithm, one of the best results regarding its false positives. However, the algorithm returns 0.692 precision which is an average value compared with the best result with 0.736 by applying a Two-Class Decision Forest.

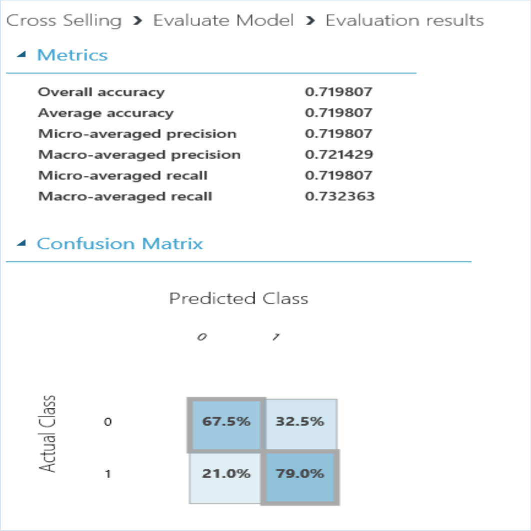
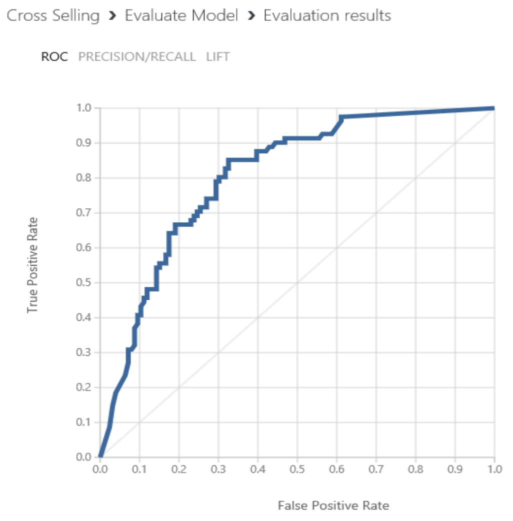
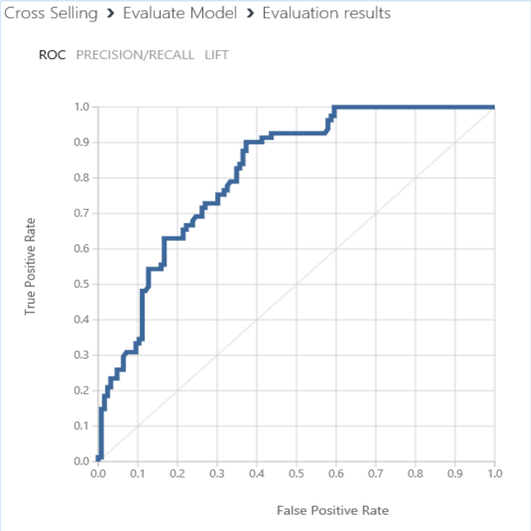
The returned recall value answers the question whether all selected cross-selling candidates that truly rated as good, finally, how many of them did the algorithm really identified. The highest recall results with 0.79 are achieved by both algorithm the Two-

Class Neural Network and Two-Class Decision Forest model, which is good as it's at least above the threshold of 0.5. The Two-Class Locally-Deep Support Vector Machines model returns with 0.765 the second-best result, followed by the Two-Class Decision Jungle with a value of 0.728. The poorest result of the recall measurement is returned by the Two-Class Logistic Regression algorithm with a value of 0.556.

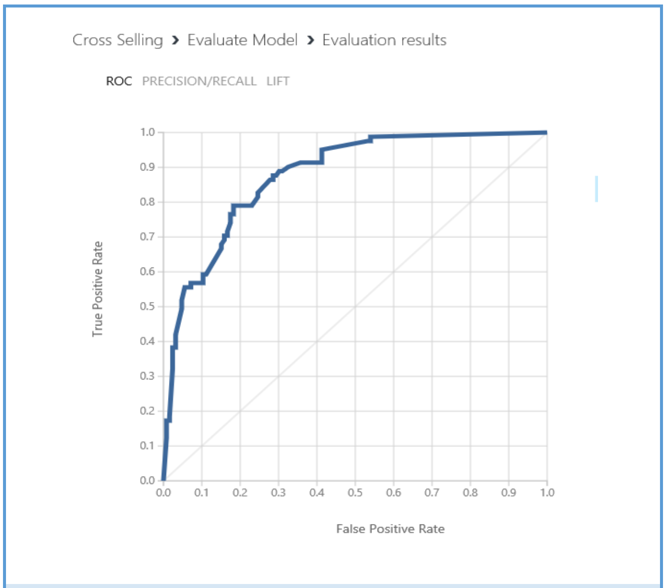
The F1-scores provides a more interesting measurement than the accuracy value since all generated false positives and false negatives are very different for every applied machine learning algorithm. The precision and recall measurements help the research work to highlight how good the model really has performed. It can be figured out that the Two-Class Decision Forest model returns the best result with a F1-score of 0.762, although the Two-Class Locally-Deep Support Vector Machines returns the second-best performance result with a weighted average (precision and recall) of 0.725. The Two-Class Logistic Regression algorithm returns the lowest F1-score, with 0.616.

The receiver operating characteristic (ROC) curves below illustrate the performance results when plotting the true positive rate against the false positive rate. The evaluation results of the ROC curves whether it is good or not can be determined by the AUC (Area Under the Curve) values which also measures the accuracy value. I have again classified the accuracy based on the following measurement ranges related to the traditional academic point system: .90-1 = excellent (A); .80-.90 = good (B); .70-.80 = fair (C); .60-.70 = poor (D) and .50-.60 = fail (F). Based on that, the AUC value clearly shows that good measurements can be achieved with Two-Class Decision Forest model (0.876), Two-Class Decision Jungle model (0.837), Two-Class Locally-Deep Support Vector Machines (0.816) and Two-Class Logistic Regression (0.814). Good accuracy with a tendency to fair measurements can be generated by Two-Class Neural Network model (0.806).

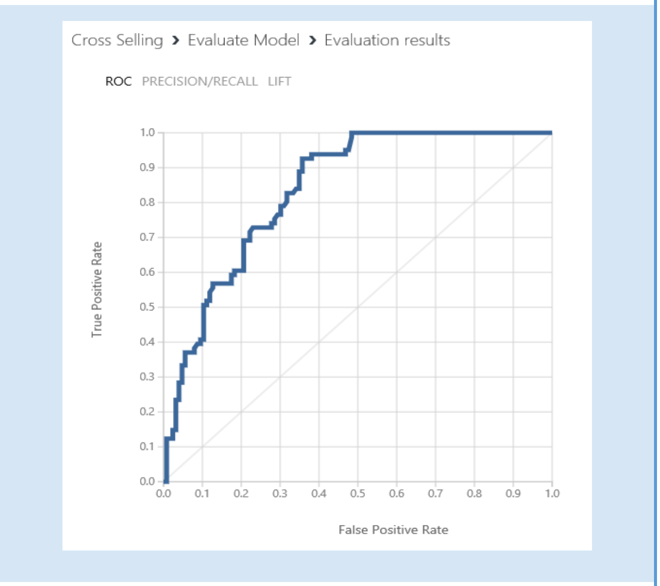
Finally, the confusion matrix of the multiclass neural network shows that the probability of correctly predicted positive values is 67.5% and negative values is 79.0% with an overall accuracy value of 0.72. Another conclusion that can be drawn from the prediction results is that the high percentage of the true negatives is an indicator of predicting more default cross-selling candidates much precisely than real (true positives) cross-selling candidates. The prediction accuracy is valued with 72%.

ROC curve algorithm	Response chart
Multiclass Neural Network	 Cross Selling › Evaluate Model › Evaluation results Metrics Overall accuracy 0.719807 Average accuracy 0.719807 Micro-averaged precision 0.719807 Macro-averaged precision 0.721429 Micro-averaged recall 0.719807 Macro-averaged recall 0.732363 Confusion Matrix Predicted Class: 0, 1 Actual Class 0: 67.5%, 32.5% Actual Class 1: 21.0%, 79.0%
Two-Class Neural Network	 Cross Selling › Evaluate Model › Evaluation results ROC PRECISION/RECALL LIFT
Two-Class Logistic Regression	 Cross Selling › Evaluate Model › Evaluation results ROC PRECISION/RECALL LIFT

Two-Class Decision Forest



Two-Class Decision Jungle



Two-Class Locally-Deep Support Vector Machines

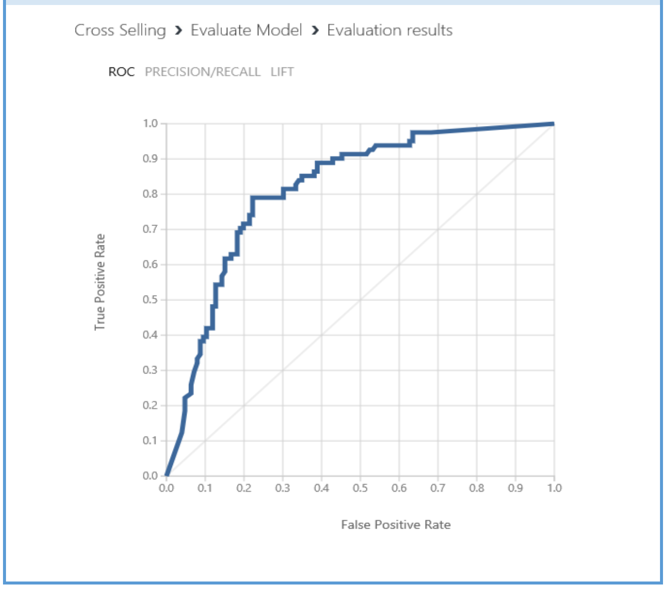
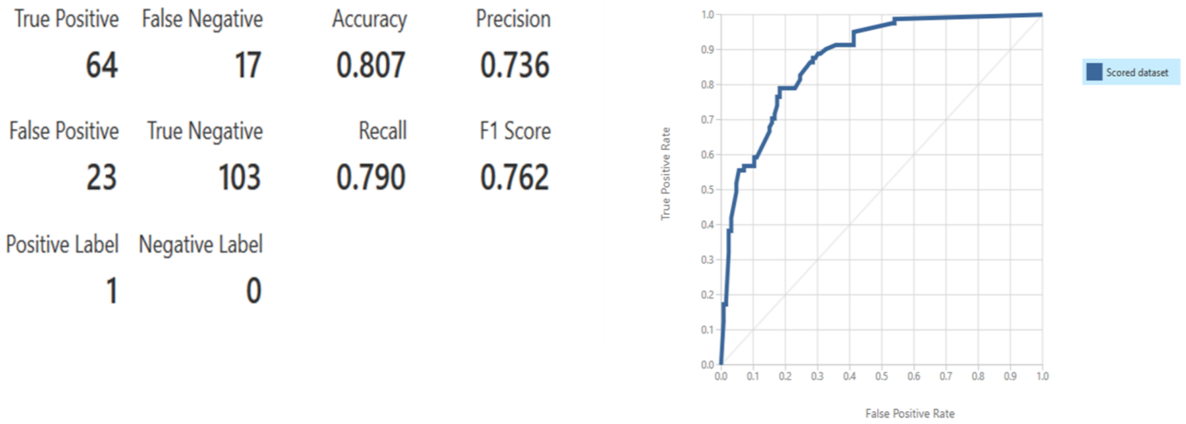


Table 5-10: Overview of the response charts of all applied algorithms - cross-selling

The tables 5-9 and 5-10 above show a detailed performance comparison of the applied clustering and classification algorithms. The overall analysis results indicate that the research outcomes from the first research project are achieving better performance results when using supervised learning algorithms than the second research project. In fact, the settled research experiments are scheduled in a different research context, but the pre-processing phase had some minor deviations. First, the predicted output of our algorithm is the categorical variable ‘cardholder’ instead of ‘status’. Additional details in the pre-processing stage which results in distinct performance outputs is explained at a later stage of this section. Regarding the evaluation, the research used various data visualising features of MS Azure ML to drill-down into a range of distinct response charts for every scored dataset.

Regarding the plotted response chart shown in the overview table 5-10 above, the ROC plot of the Two-Class Decision Forest is by far the best one. The performance results of the Two-Class Decision Jungle as well as Two-Class Locally-Deep Support Vector Machine are also returning an appropriate picture of the predicted values presented in the table 5-10 above. A detailed view at the AUC value show an accuracy measurement value of 0.837 for the Two-Class Decision Forest and an AUC value of 0.816 for the Two-Class Locally-Deep Support Vector Machine. Comparing both ROC plots with the less pleasant response chart of the Two-Class Logistic Regression, the corresponding AUC value with 0.814 is measurably not worse than the performance outcomes of the two other response charts. The evaluation results of the applied Two-Class Logistic Regression algorithm also produce a relatively good response chart with ups and down in the ROC curve.

In conclusion, it has to be noted that the algorithm Two-Class Random Forest plots the best ROC response chart based on their calculated prediction results. Figure 5-54 below presents the detailed evaluation results of the cross-selling model on the left-hand side and their corresponding ROC response chart on the right-hand side. The data visualising feature of MS Azure ML supports the current research to validate the scored dataset and showing that the AUC is indeed equal to 0.876 as described in the summary overview above. The grey line represents a completely uninformative test, which will be returned by an AUC threshold of 0.5. The core message of the displayed ROC curve is that the applied Two-Class Decision Forest algorithm is performing best since the curve is pulled close to the upper left corner of the response chart.



True Positive	False Negative	Accuracy	Precision
64	17	0.807	0.736
False Positive	**True Negative**	**Recall**	**F1 Score**
23	103	0.790	0.762
Positive Label	**Negative Label**		
1	0		

Figure 5-54: Summary of the best modelling results for cross-selling achieved by Two-Class Decision Forest algorithm

The research work further studies the scalability as the conducted research experiments also changes the size of the threshold in five ways by keeping all other performance metric parameters constant. The constant performance metric parameters for all five experiments were true positive, true negative, false positive, false negative, accuracy, precision, recall and F1-score, and the threshold was varied from 0 to 1. Regarding the research design, the experiments are aligned to the defined threshold buckets of 0; 0.25; 0.5; 0.75 and 0.99.

Table 5-11 below presents how the best-performing algorithm scales with the five different threshold parameters. For higher values of thresholds (>0.5) the predicted values consisting of true positive and true negative decrease with increasing maximal threshold size. This is due to the fact that the recall values are falling dramatically to a maximum of 0.123 the higher the threshold (+0.99) is. The consequence is also that the algorithm will discover fewer false positives (+1) and an increased number of false negatives (+71). However, for lower values of the threshold, a larger number of predicted scorings (true positive + true negative) appears, and thus the predicted scorings start to increase initially but then decrease again when the threshold is greater than 0.5 due to the same reasons given above. If we take a closer look at both thresholds 0.25 and 0.75, the research has observed that the evaluation results are nearly the same (i.e. comparing predicted scorings which indicates a total of 158 cross-selling candidates by 0.25 and 156 cross-selling candidates by 0.75 or a prediction accuracy value of 0.763 by 0.25 and 0.754 by 0.75) with some minor but major differences such as the precision and recall measurements as well as false positive and false negative values. The F1-scores of both thresholds underlines that the entire evaluation results for the threshold value of 0.25 (F1-score of 0.749) is much better than for the threshold value of 0.75 (F1-score of 0.571). The peak occurs at roughly the threshold value of 0.5 for the research experiments.

Finally, the table 5-11 below provides the prediction results through the selected performance characteristics for the best-performing Two-Class Decision Forest algorithm applied on different thresholds.

Threshold	True Positive	True Negative	False Positive	False Negative	Accuracy	Precision	Recall	F1 Score	AUC
0	80	58	68	1	0.667	0.541	0.988	0.699	0.876
0.25	73	85	41	8	0.763	0.640	0.901	0.749	0.876
0.5	64	103	23	17	0.807	0.736	0.790	0.762	0.876
0.75	34	122	4	47	0.754	0.895	0.420	0.571	0.876
0.99	10	125	1	71	0.652	0.909	0.123	0.217	0.876

Table 5-11: Summary of the evaluation results of the scaled Two-Class Decision Forest - cross-selling

The following sections are investigating the last part of the second research project in which the research study seeks to evaluate the performance outcomes of the applied Two-Class Decision Forest algorithm when executing the algorithm on different (advanced) data models. Therefore, I have designed three different research experiments by running the predictive model again with the relevant variables deducted from the descriptive analysis in the previous section and checking the performance results as well as response charts against the original response chart (incl. all "unnecessary" input variables). The research experiments also support one of the research objectives to figure out a well-fitted model which result in good prediction results including sufficient and valuable input parameters for the data model. The outcomes of the research analysis will provide the evidence of the statement above.

The table 5-12 below describes the evaluation results for exploring a cost-sensitive data model effected based upon five customised research experiments which are aligned to the key findings from the variable importance analysis through the descriptive analysis. In the first instance, the research experiment is conducted on the original pre-processed dataset, the second experiment is carried out on the original dataset excluding the variables ‘frequency’ and ‘sex’, the third research experiment is executed on an optimized dataset which consists of the variable ‘Age’, ‘AvgIncome’, ‘AvgExpenses’, ‘AvgBalance’, ‘AvgSalary’, the fourth research experiment is also executed on an optimized dataset which consists only of the variable ‘Age’, ‘AvgExpenses’, ‘AvgBalance’, ‘AvgSalary’, and the last research experiment is executed on an optimised dataset which consists only of the variable ‘Age’, ‘AvgIncome’, ‘AvgBalance’, ‘AvgSalary’.

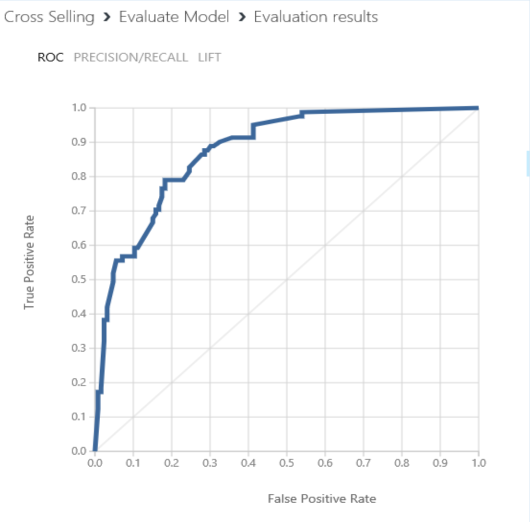
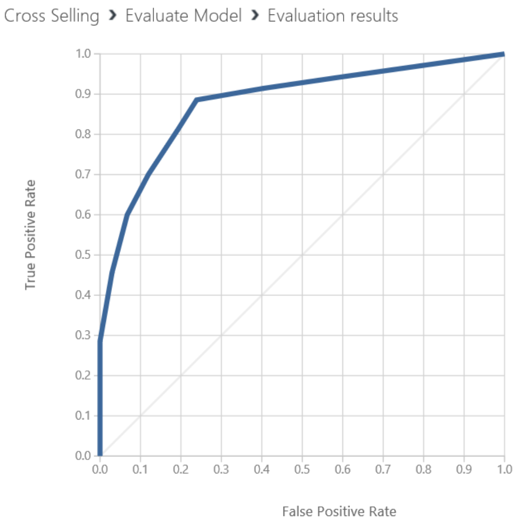
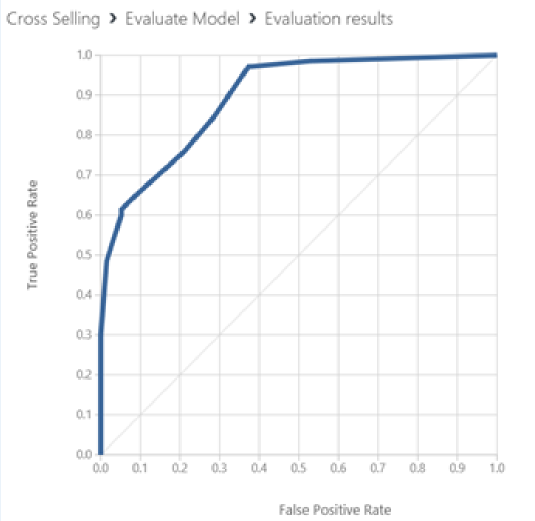
Algorithm applied by a threshold of 0.5	True Positive	True Negative	False Positive	False Negative	Accuracy	Precision	Recall	F1 Score	AUC
Two-Class Decision Forest applied on the original dataset	64	103	23	17	0.807	0.736	0.79	0.762	0.876
Two-Class Decision Forest applied on the optimised dataset (excl. frequency and sex)	49	118	16	21	0.819	0.754	0.70	0.726	0.879
Two-Class Decision Forest applied on the optimised dataset (incl. Age, AvgIncome, AvgExpenses, AvgBalance, AvgSalary)	46	121	13	24	0.819	0.78	0.657	0.713	0.895
Two-Class Decision Forest applied on the optimised dataset (incl. Age, AvgExpenses,	46	125	9	24	0.838	0.836	0.657	0.736	0.897

AvgBalance, AvgSalary)									
Two-Class Decision Forest applied on the optimised dataset (incl. Age, AvgIncome, AvgBalance, AvgSalary)	49	115	19	21	0.804	0.721	0.700	0.710	0.877

Table 5-12: Comparing evaluation results of applied Two-Class Decision Forest - original vs. optimised dataset for cross-selling

I have observed through the customised research experiments that the supervised learning algorithm Two-Class Decision Forest is using an overwhelming number of input variables, but all five pre-processed datasets consisting of potentially useful predictor variables because they produce high-performance results which are beside each other with minimal deviations resulting from the calculated false positive and false negative values. The output of the research experiments shows that in the long run, many small differences in the performance details breed a large difference in the performance quality of every single data model.

The best performance result (AUC is 0.897) was achieved by the fourth experiment whose executed cost-sensitive data model consists of only four input variables ‘Age’, ‘AvgExpenses’, ‘AvgBalance’, ‘AvgSalary’. It was surprising that the last experiment shows lower performance results (AUC is 0.877) than the remaining research experiments, and especially when comparing the results with the fourth experiment. However, the only difference in data modelling between the last two experiments was the separation of the two strongly correlating variables ‘AvgIncome’ and ‘AvgExpenses’. Detailed analysis results are provided in the flattened correlation matrix from the previous section. The third research experiment has achieved the second-best performance outcomes (AUC is 0.895) by executing an optimized dataset consists of the variable ‘Age’, ‘AvgIncome’, ‘AvgExpenses’, ‘AvgBalance’, ‘AvgSalary’. The poorest performance results (AUC is 0.876) will be produced on the original dataset in the first research experiment.

ROC curve algorithm	Response charts
Two-Class Decision Forest applied on the original dataset	
Two-Class Decision Forest applied on the optimised dataset (excl. frequency and sex)	
Two-Class Decision Forest applied on the optimised dataset (incl. Age, AvgIncome, AvgExpenses, AvgBalance, AvgSalary)	

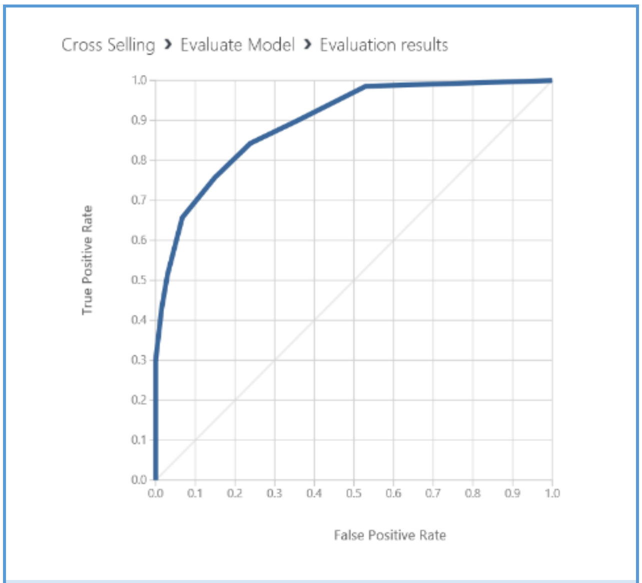
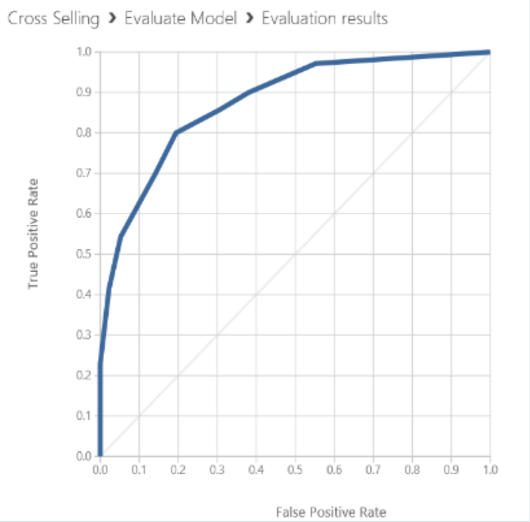
Two-Class Decision Forest applied on the optimised dataset (incl. Age, AvgExpenses, AvgBalance, AvgSalary)	
Two-Class Decision Forest applied on the optimised dataset (incl. Age, AvgIncome, AvgBalance, AvgSalary)	

Table 5-13: Overview of the response charts of applied Two-Class Decision Forest - original vs. optimised dataset for cross-selling

The ROC curves above indicate that the applied algorithm will generate best predictions (AUC is 0.897) when the predictive model is built only on the four important variables ‘Age’, ‘AvgExpenses’, ‘AvgBalance’, ‘AvgSalary’. Lower performance results will be produced in the second (AUC is 0.879) and last (AUC is 0.877) research experiment, but overall definitely a relative high-performance model was built. However, the poorest performance outcomes are achieved in the first research experiment on the original dataset as the AUC is measured with 0.876.

The current research experiments show that the applied data model generate remarkable response charts for every optimised dataset. The ROC curve for the last research experiment is pulled close to the upper left corner which indicates that the applied Two-Class Decision Forest algorithm is performing best on an advanced dataset which includes only the selected variables. However, the detailed evaluation results among the five designed research experiments underlines that some form of post-processing in case of executing the varImp() function through the data pre-processing phase is necessary to rule out the irrelevant variables and to locate the most significant input variables for the predictive model. Further observations can be drawn out of the first and second research experiments. There are no significant differences within the output of the performance metric if I will exclude the variable 'frequency' and 'sex' from the dataset. This detailed and visualised prediction results are also the evidence of the applied varImp() function as part of the descriptive analysis. In summary therefore, the research results show that the best cost-sensitive data model is again deployed in an iterative process.

The research also showed that the output of the supervised learning algorithm can be used to build scalable monitors which predict failures in case of cross-selling candidates with bad scorings in a plan before they really occur. The research was able to produce monitors with precisions up to 83,6% that approximately 70 out of 204 (+34.31%) cross-selling candidates are actual positives that occur.

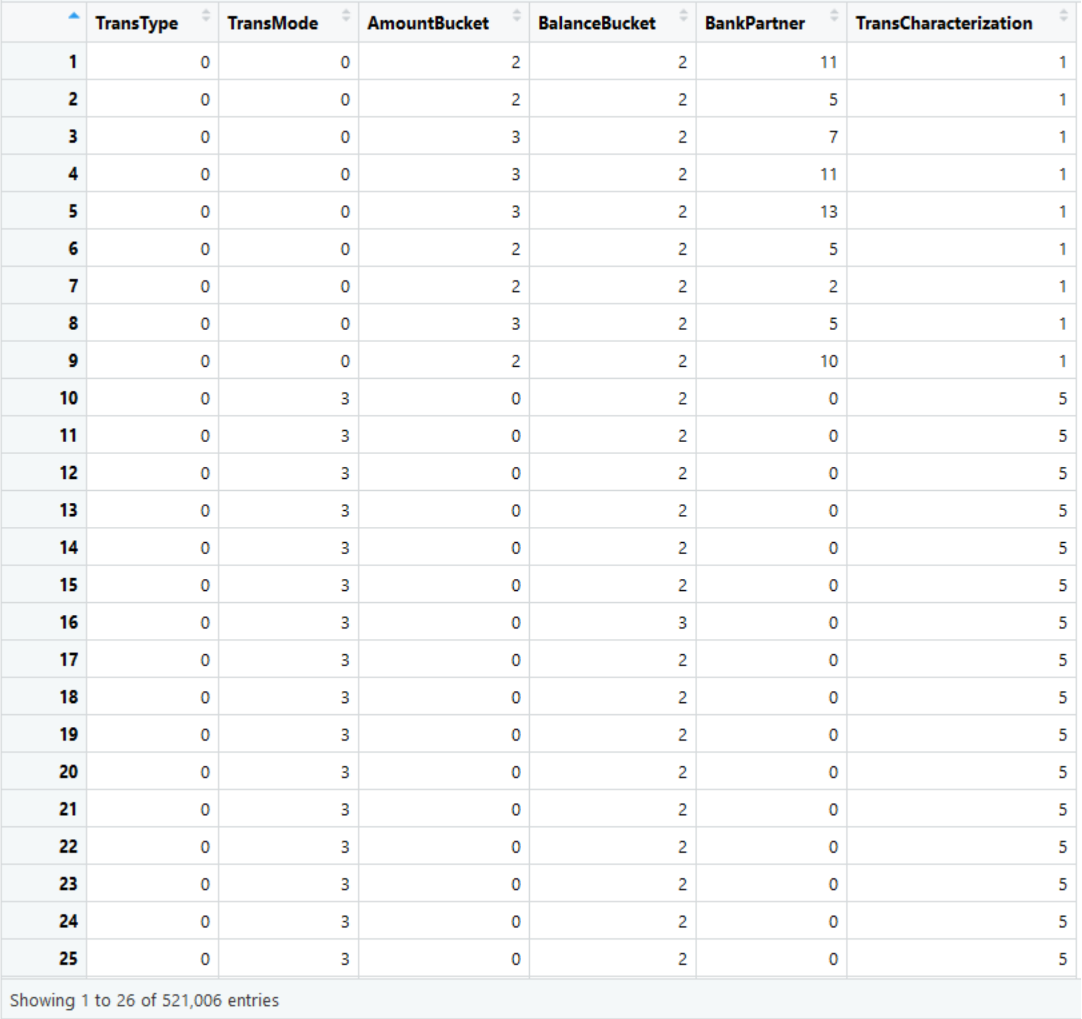
5.2.3. Modelling of the categorised transactional payment behaviours

The following section describes the overall evaluations and findings made by modelling the payment behaviour on a categorised transactional dataset. As mentioned in chapter 4, the pre-processing phase has processed different datasets and the current research work is using an objective and independent research approach to answer the outlined research questions. Therefore, various mining methods for predicting the transactional payment category were presented and applied to ensure the objectivity of the research results. The focus of this section will be on explaining and testing the predictive results of the computational system neural network according to the last research project. The detail theoretical construct of the neural network consisting of input, hidden and output layers is introduced in the previous chapters.

In the research experiment, the goal is to develop a neural network to determine if a categorised transaction type is predicted right or not. The first step of the analysis is to perform a detailed data exploration analysis in the context of a descriptive analysis, whose results are shown in appendix C.3 as bar plots, histograms and boxplots for every processed transactional payment attribute. The bar plot of the input variable ‘trans mode’ shows that the data is skewed to the right, and the boxplot illustrates that the values are distributed approximately equal. The values of the input variable ‘amount bucket’ is strongly distributed to the left as illustrated by the bar plot and histogram. However, from the boxplots applied on each attribute separately it is clear that there exist outliers in the fields ‘amount bucket’, ‘balance bucket’ and ‘bank partner’. The value distribution of the variable ‘TransCharacterization’ breaks down very unequally as shown in the bar plot and histogram figure, hence the evaluations of their predictions should be probably handled with caution.

Our independent variables within our neural network which solves the classification problem are as follows: ‘TransType’, ‘TransMode’, ‘AmountBucket’, ‘BalanceBucket’ and ‘BankPartner’. By classification, the research means ones where the transaction data is classified by categories across the ‘TransCharacterization’ field, e.g. a transaction category can be classified as insurance payment, payment for statement, interest credited, sanction interest if negative balance, household, old-age pension or loan payment.

The content of pre-processed dataset for the research experiment is illustrated in the table 5-14 below. A detailed exploratory data analysis of the entire table is provided in appendix C.3 and D.6. Before fitting a neural network, some data preparation needs to be carried out since neural networks are not that easy to train and tune.

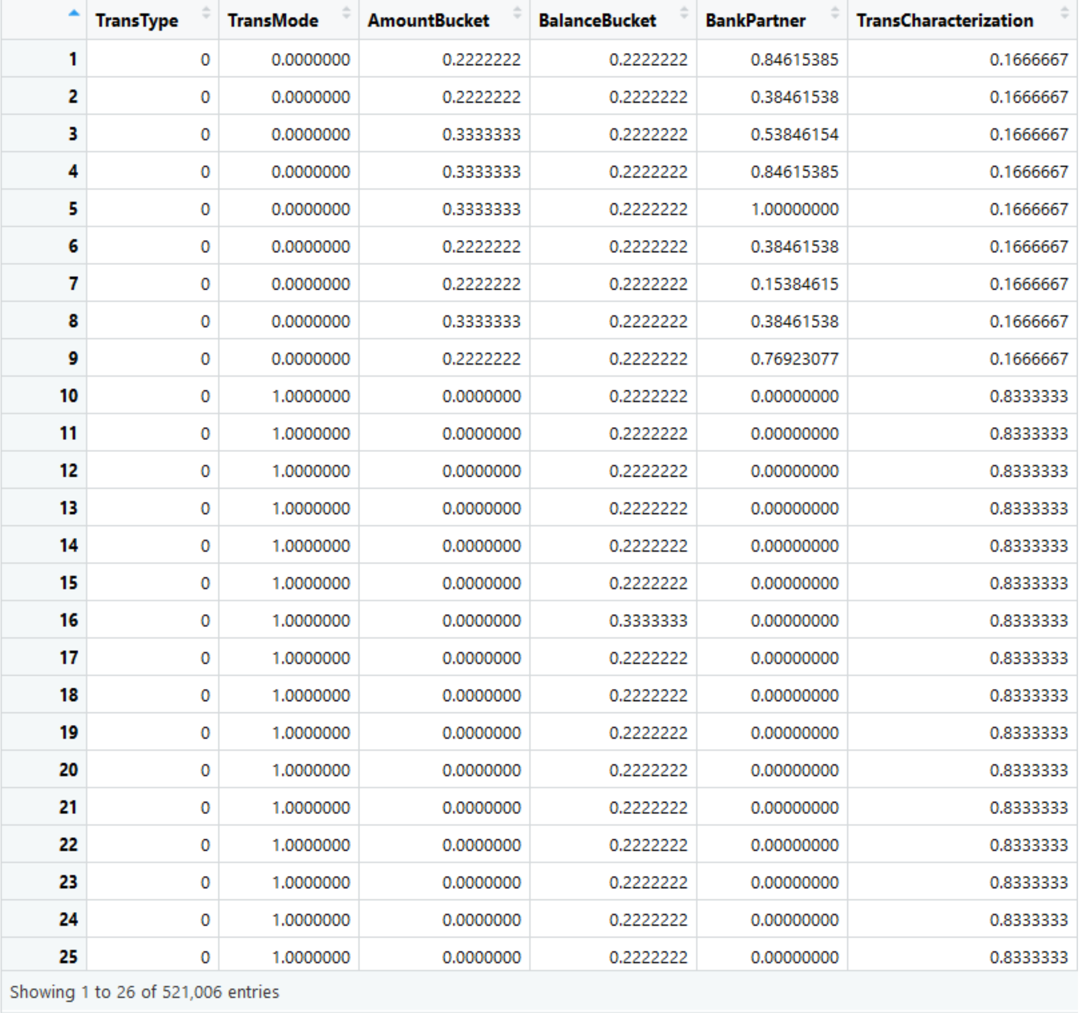


	TransType	TransMode	AmountBucket	BalanceBucket	BankPartner	TransCharacterization
1	0	0	2	2	11	1
2	0	0	2	2	5	1
3	0	0	3	2	7	1
4	0	0	3	2	11	1
5	0	0	3	2	13	1
6	0	0	2	2	5	1
7	0	0	2	2	2	1
8	0	0	3	2	5	1
9	0	0	2	2	10	1
10	0	3	0	2	0	5
11	0	3	0	2	0	5
12	0	3	0	2	0	5
13	0	3	0	2	0	5
14	0	3	0	2	0	5
15	0	3	0	2	0	5
16	0	3	0	3	0	5
17	0	3	0	2	0	5
18	0	3	0	2	0	5
19	0	3	0	2	0	5
20	0	3	0	2	0	5
21	0	3	0	2	0	5
22	0	3	0	2	0	5
23	0	3	0	2	0	5
24	0	3	0	2	0	5
25	0	3	0	2	0	5

Showing 1 to 26 of 521,006 entries

Table 5-14: View of the table - myDataTrans

The following paragraphs will introduce the major steps to construct the model for predicting categorised transactions through transaction payment streams. First step in the procedure of forming a neural network is data normalisation. This important step may lead to useful results or to an easier training process since most of the times the neural network algorithm will not converge before the number of maximum iterations allowed. The goal is to adjust the above table “myDataTrans” to a common scale in the interval [0,1] by using max-min-normalisation technique to accurately compare actual and predicted values. The scaling method usually tend to provide better results. Further details about the implementation in R Studio on how the data is normalised can be seen in appendix E.1. In general, I have used the lapply() function across the pre-processed dataset and the scaled data is coerced into a new data frame named “maxmindf”. Finally, the table 5-15 below illustrates the normalised dataset for the predictive model.



	TransType	TransMode	AmountBucket	BalanceBucket	BankPartner	TransCharacterization
1	0	0.0000000	0.2222222	0.2222222	0.84615385	0.1666667
2	0	0.0000000	0.2222222	0.2222222	0.38461538	0.1666667
3	0	0.0000000	0.3333333	0.2222222	0.53846154	0.1666667
4	0	0.0000000	0.3333333	0.2222222	0.84615385	0.1666667
5	0	0.0000000	0.3333333	0.2222222	1.00000000	0.1666667
6	0	0.0000000	0.2222222	0.2222222	0.38461538	0.1666667
7	0	0.0000000	0.2222222	0.2222222	0.15384615	0.1666667
8	0	0.0000000	0.3333333	0.2222222	0.38461538	0.1666667
9	0	0.0000000	0.2222222	0.2222222	0.76923077	0.1666667
10	0	1.0000000	0.0000000	0.2222222	0.00000000	0.8333333
11	0	1.0000000	0.0000000	0.2222222	0.00000000	0.8333333
12	0	1.0000000	0.0000000	0.2222222	0.00000000	0.8333333
13	0	1.0000000	0.0000000	0.2222222	0.00000000	0.8333333
14	0	1.0000000	0.0000000	0.2222222	0.00000000	0.8333333
15	0	1.0000000	0.0000000	0.2222222	0.00000000	0.8333333
16	0	1.0000000	0.0000000	0.3333333	0.00000000	0.8333333
17	0	1.0000000	0.0000000	0.2222222	0.00000000	0.8333333
18	0	1.0000000	0.0000000	0.2222222	0.00000000	0.8333333
19	0	1.0000000	0.0000000	0.2222222	0.00000000	0.8333333
20	0	1.0000000	0.0000000	0.2222222	0.00000000	0.8333333
21	0	1.0000000	0.0000000	0.2222222	0.00000000	0.8333333
22	0	1.0000000	0.0000000	0.2222222	0.00000000	0.8333333
23	0	1.0000000	0.0000000	0.2222222	0.00000000	0.8333333
24	0	1.0000000	0.0000000	0.2222222	0.00000000	0.8333333
25	0	1.0000000	0.0000000	0.2222222	0.00000000	0.8333333

Showing 1 to 26 of 521,006 entries

Table 5-15: View of the normalised table - myDataTrans (maxmindf)

After the data pre-processing steps, the cleaned data is fed as input to the neural network algorithm. Before training the neural network using “neuralnet” package from R Studio, the entire dataset is divided into training data (trainset) based on 80% of the observations and the test data (test set) based on the remaining 20% of observations.

The research experiments are using the “neuralnet” package to regress the dependent “TransCharacterization” variable against the other independent variables. In the following, only networks of the successful type Feedforward Multilayer Perceptron (MLP) are used. These are learning classifiers that, by presenting training data, can learn the functional relationships between input and output patterns without having in any way been given the unknown analytical dependencies. At least, in the current research experiments testing again and again is the best approach since there is no guarantee that any of these rules will fit the predictive model best.

Regarding the current research experiments, the number of hidden layers is first set to (2,1) based on the hidden = (2,1) formula. Second, the “linear.output” argument is set to FALSE because I want to solve the classification problem assuming that there is a non-linear relationship between the independent variables and the dependent variable ‘TransCharacterization’. The last parameter threshold is set to 0.01, which means that no further optimisation will be carried out by the developed model whenever the change of error is less than 1%. However, I have proceeded with try and error when adjusting the number of hidden layers in the developed neural network model to check whether the accuracy of the predictions changes whenever the number of hidden values is modified. The current research experiments can also be scaled in a way to discover at which number of hidden nodes the model is performing best. However, enhancing the developed model is not our current research focus. For instance, I have used a (2,1) configuration which ultimately yielded up to 90.7% classification accuracy for some of the designed research experiments. It proved that the current configuration is sufficient for the model to return accurate results. However, I have also recognised that a further parameter “stepmax” must be set to a high number such as 1.000.000 to allow the algorithm to take all necessary steps.

Regarding the research setup, the large-scale transactional payment dataset of +521.006 records described in the tables 5-14 and 5-15 above was divided (randomly) into eleven buckets which are also representing the realised research experiments in the area of predicting categorised payment transactions. The reason behind this approach was that I have had limited computing capacity with my recent workstation when computing a neural network model in R Studio. Therefore, rescaling the research experiments from one large-sized experiment into eleven smaller-sized datasets has consequently become necessary (divide-and-conquer principle) to develop the last research project and provide a preliminary indication of possible answers to the research question. Therefore, each developed research experiment will support the objectivity of the research outcomes. Detailed evaluation results of all developed predictive models are presented in appendix D.6. For this purpose, the results are provided as neural network plots, neural network results matrix, confusion matrix and an accuracy rate of every model. The following paragraphs will pragmatically explain, for example, the evaluation results of one of the research experiments.

Figure 5-55 below illustrates a neural network summary of the input data for the research experiment - bucket#10. The neural network plot of the input variable including their prediction results is illustrated in the graphical representation for nodes and edges. The enhanced model also includes a weighting of the edges as an additional dimension. Therefore, the MLP can be considered in every research experiment as a black box because I cannot say that much about the knowledge generating process like fitting, the weights and the model.

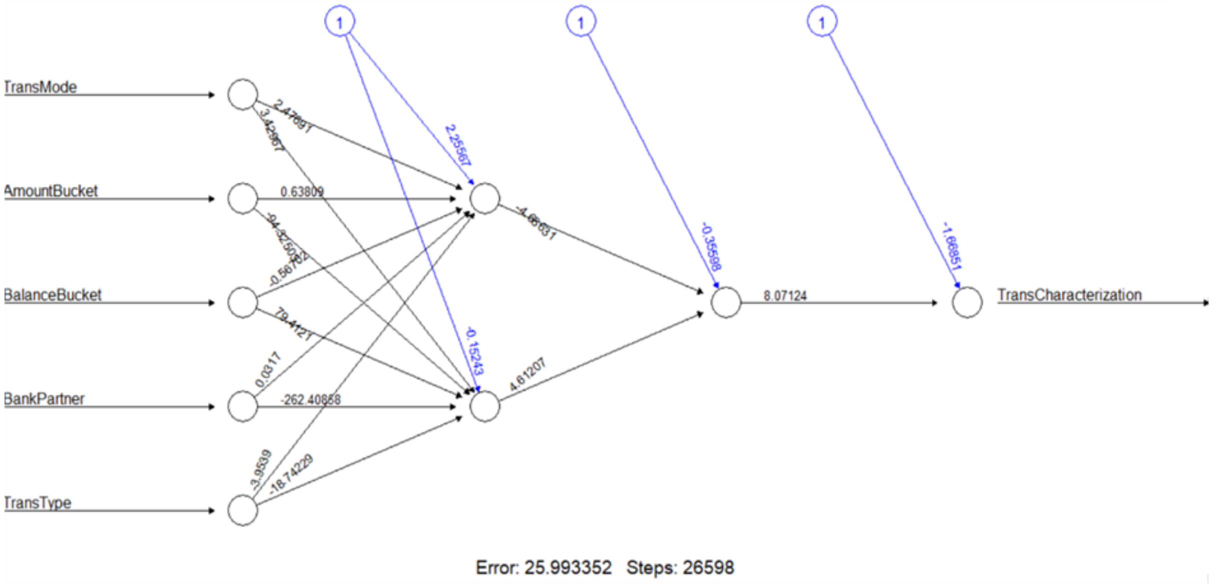
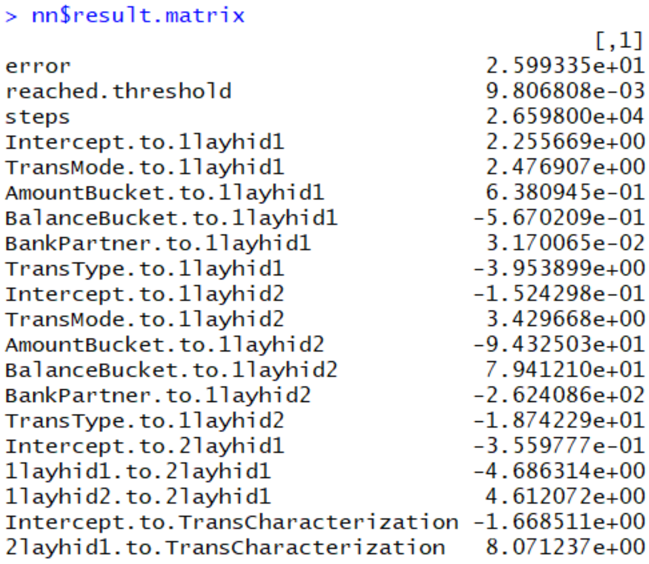


Figure 5-55: Plot of the neural network - bucket#10

Above graph obtained by plotting the results of the neural network algorithm obtained in 26,598 successive iterations are as shown also in figure 5-56 below. The black lines show the connections between each layer and the weights on each connection while the blue lines show the bias term added in each step. From both figures 5-55 and 5-56, it is clear that the training algorithm has converged and therefore the model is ready to be used.

After running the algorithm for quite a while to fit the model, the neural net algorithm has computed the below result matrix. The result matrix provides an overview of the generated error of the neural network model, along with the weights between the inputs, hidden layers, and outputs. The error value of 25.993352 is difficult to interpret. However, I want to explore the customer behaviour on payment data besides the neural nets. The black arrows will tell us how much that input variable contribute to the next node. The returning results of the model show that the variables 'BalanceBucket' (79.41) and 'TransMode' (3.42) have a greatest contribution to the following nodes, and the following both abstract nodes constitute components that the network is learning to recognise. For instance, the first hidden layer and their biases (blue lines) represents that the constructed model was learned / stored on a richer representation of the input variables. The increased variation of weights calculated with the back-propagation algorithm has generally enhanced the neural net’s guessing power in the current research experiment. In summary, the results prove that the algorithm essentially works to learn categorical payment transactions. The model learns from the information provided as input variables, which has a known categorised payment transaction type (outcome) and optimises its weights for an improved forecast, for example, in payment streams with unknown categorised payment transactions (outcomes).

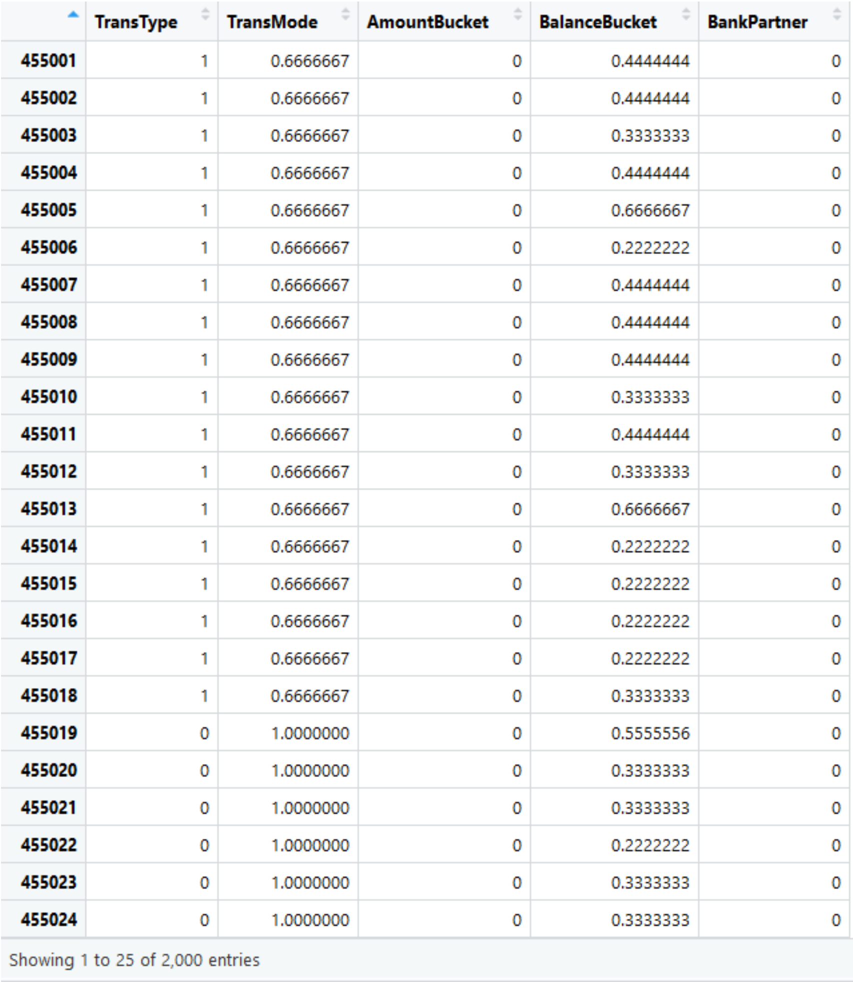


```
> nn$result.matrix
                                           [,1]
error                              2.599335e+01
reached.threshold                  9.806808e-03
steps                              2.659800e+04
Intercept.to.1layhid1              2.255669e+00
TransMode.to.1layhid1              2.476907e+00
AmountBucket.to.1layhid1           6.380945e-01
BalanceBucket.to.1layhid1         -5.670209e-01
BankPartner.to.1layhid1            3.170065e-02
TransType.to.1layhid1             -3.953899e+00
Intercept.to.1layhid2             -1.524298e-01
TransMode.to.1layhid2              3.429668e+00
AmountBucket.to.1layhid2          -9.432503e+01
BalanceBucket.to.1layhid2          7.941210e+01
BankPartner.to.1layhid2           -2.624086e+02
TransType.to.1layhid2             -1.874229e+01
Intercept.to.2layhid1             -3.559777e-01
1layhid1.to.2layhid1              -4.686314e+00
1layhid2.to.2layhid1               4.612072e+00
Intercept.to.TransCharacterization -1.668511e+00
2layhid1.to.TransCharacterization   8.071237e+00
```

Figure 5-56: Overview of the neural network results matrix - bucket#10

Apparently, the net is doing in same cases a different work compared with the other remaining research experiments at predicting categorised transactions. Once again, the research results should be interpreted very carefully because the presented result depends on the train-test split performed above for the unique research experiment “bucket#10”. Note that there are some research experiments which presents worse results due to the underlying datasets. Further down in the section, current research performs a fast-cross validation by using a more powerful tool for large-scale data processing to be more confident about the overall research results.

The following mining step is performed to test the accuracy of the predictive model. Therefore, the research experiment has been created on the test data to gauge the accuracy of the neural network forecast, and then compare them to the predictions resulting from the training data.

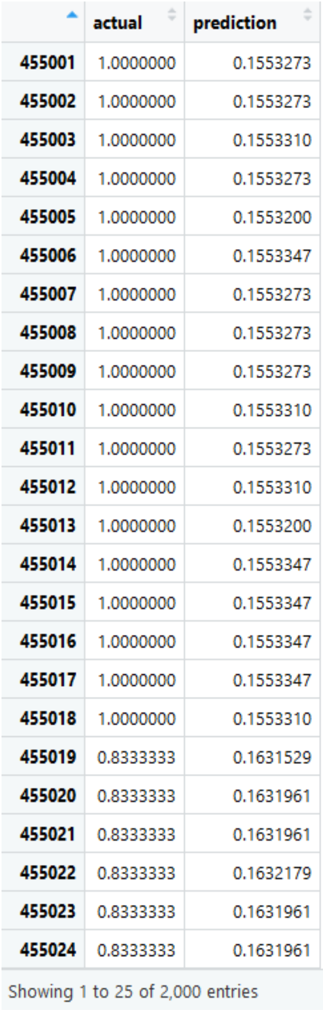


	TransType	TransMode	AmountBucket	BalanceBucket	BankPartner
455001	1	0.6666667	0	0.4444444	0
455002	1	0.6666667	0	0.4444444	0
455003	1	0.6666667	0	0.3333333	0
455004	1	0.6666667	0	0.4444444	0
455005	1	0.6666667	0	0.6666667	0
455006	1	0.6666667	0	0.2222222	0
455007	1	0.6666667	0	0.4444444	0
455008	1	0.6666667	0	0.4444444	0
455009	1	0.6666667	0	0.4444444	0
455010	1	0.6666667	0	0.3333333	0
455011	1	0.6666667	0	0.4444444	0
455012	1	0.6666667	0	0.3333333	0
455013	1	0.6666667	0	0.6666667	0
455014	1	0.6666667	0	0.2222222	0
455015	1	0.6666667	0	0.2222222	0
455016	1	0.6666667	0	0.2222222	0
455017	1	0.6666667	0	0.2222222	0
455018	1	0.6666667	0	0.3333333	0
455019	0	1.0000000	0	0.5555556	0
455020	0	1.0000000	0	0.3333333	0
455021	0	1.0000000	0	0.3333333	0
455022	0	1.0000000	0	0.2222222	0
455023	0	1.0000000	0	0.3333333	0
455024	0	1.0000000	0	0.3333333	0

Showing 1 to 25 of 2,000 entries

Table 5-16: View of the table “temptest” - myDataTrans

Above table “temptest” provides an overview of the processed test data for testing the predictive results of the developed model using a neural network. Note that the predicted class (categorised transaction type) will be scaled and it must be transformed with this intermediate step to make a real comparison with real categorised transaction types. The corresponding R-script is given in appendix E.1. I have initially used the “subset” function to eliminate the dependent variable “TransCharacterization” from the test data, and as a second step, the “compute” function was applied to create the prediction variable. Next step is summarising the predicted data with the actual data into a “results” variable to achieve comparability between the two values using visualisation techniques. Details can be seen in the table 5-17 below.



	actual	prediction
455001	1.0000000	0.1553273
455002	1.0000000	0.1553273
455003	1.0000000	0.1553310
455004	1.0000000	0.1553273
455005	1.0000000	0.1553200
455006	1.0000000	0.1553347
455007	1.0000000	0.1553273
455008	1.0000000	0.1553273
455009	1.0000000	0.1553273
455010	1.0000000	0.1553310
455011	1.0000000	0.1553273
455012	1.0000000	0.1553310
455013	1.0000000	0.1553200
455014	1.0000000	0.1553347
455015	1.0000000	0.1553347
455016	1.0000000	0.1553347
455017	1.0000000	0.1553347
455018	1.0000000	0.1553310
455019	0.8333333	0.1631529
455020	0.8333333	0.1631961
455021	0.8333333	0.1631961
455022	0.8333333	0.1632179
455023	0.8333333	0.1631961
455024	0.8333333	0.1631961

Showing 1 to 25 of 2,000 entries

Table 5-17: View of the table - predictive results vs. actual results for bucket#10

After comparing the predictive results with the actual results, a confusion matrix is then created by using the table function to compare the number of true/false positives and negatives. In doing so, the results are finally rounded up using sapply() function from the R-package “neuralnet”.

Prediction

		0	**1**
Actual	**0**	1099	34
	1	867	0

Table 5-18: Confusion matrix of the prediction results - bucket#10

Conventionally, a standard way of describing the accuracy of a diagnostic test is the two-by-two table 5-18 above. This is performed because the test results of the predictions are recorded as dichotomous outcomes (actual and prediction results). As the table 5-18 shows, the column represents the true status of transaction characterisation state that is assessed without errors by the neural network algorithm.

The confusion matrix shows that the model generates 1,099 true negatives (0's), 0 true positives (1's), while there are 867 false negatives and 34 false positives. The final outcome of the model yields to an accuracy rate of 54.95% (1099/2000) when determining if a transaction characterisation is known or not. The misclassification rate of this research experiment is 45.05% which means that every second payment category is predicted wrong.

Following table 5-19 provides an entire overview of the evaluation results for the implemented research experiments in R Studio. Further visualisations of the predictive results are given in appendix D.6. The measurements of the predictive analysis results such as generated error, accuracy rate, misclassification rate etc. reflects that neural network is well on the way to become a useful instrument to estimate the next payment transaction.

Research experiment	True Positive	True Negative	False Positive	False Negative	Accuracy	Misclassification	Error	Iteration steps	MSE
#1	-	644	-	1356	32.2%	67.8%	33.23	1089	0.37
#2	-	5	-	1996	0.25%	99.75%	24.55	21686	0.55
#3	-	1784	-	216	89.2%	10.8%	43.05	1888	0.04
#4	-	4	-	1996	0.2%	99.8%	20.22	438810	0.55
#5	-	1808	-	192	90.4%	9.6%	45.26	1971	0.04
#6	-	4	-	1996	0.25%	99.75%	46.94	2755	0.57
#7	-	1814	-	186	90.7%	9.3%	18.26	53032	0.05
#8	-	16	-	1984	0.8%	99.2%	50.23	2148	0.55
#9	-	2	-	1998	0.1%	99.9%	54.4	2794	0.58
#10	0	1099	34	867	54.95%	45.05%	25.99	26598	0.26
#11	0	406	9	1585	20.3%	79.7%	50.53	13583	0.45

Table 5-19: Summary of overall evaluation results of the neural network algorithm applied in R Studio

The research experiments indicate that the outcomes of every single model are varying due to the underlying train and test set. However, it can be simplified that the model can forecast transaction categories with an accuracy of up to 90.7% and a mean squared error (MSE) value of 0.04.

The application of neural networks is simple; nevertheless, due to their non-linear structure, they are able to recognise even complex hidden structures within the transactional payment dataset as illustrated with the reference example above. Particularly striking is the ability of the neural network to predict the categorised transaction in terms of payment data streams. The customer behaviour on a payment was learned from the previous payments. Although the dataset comprises only a few categorised transactions, the correct trend is qualitatively predicted by the network.

Finally, in the second part of the research analysis, I want to evaluate the entire dataset of +521.006 payment data records with the help of another independent approach and mining tool. As mentioned in chapter 4, the second analysis procedure is based on the Python results when using the deep-learning library TensorFlow from Google for fast numerical computing. This research setup enables us to evaluate the entire large-scale dataset of +521.006 transactions and will complete the research analysis of how categorised transaction can be predicted.

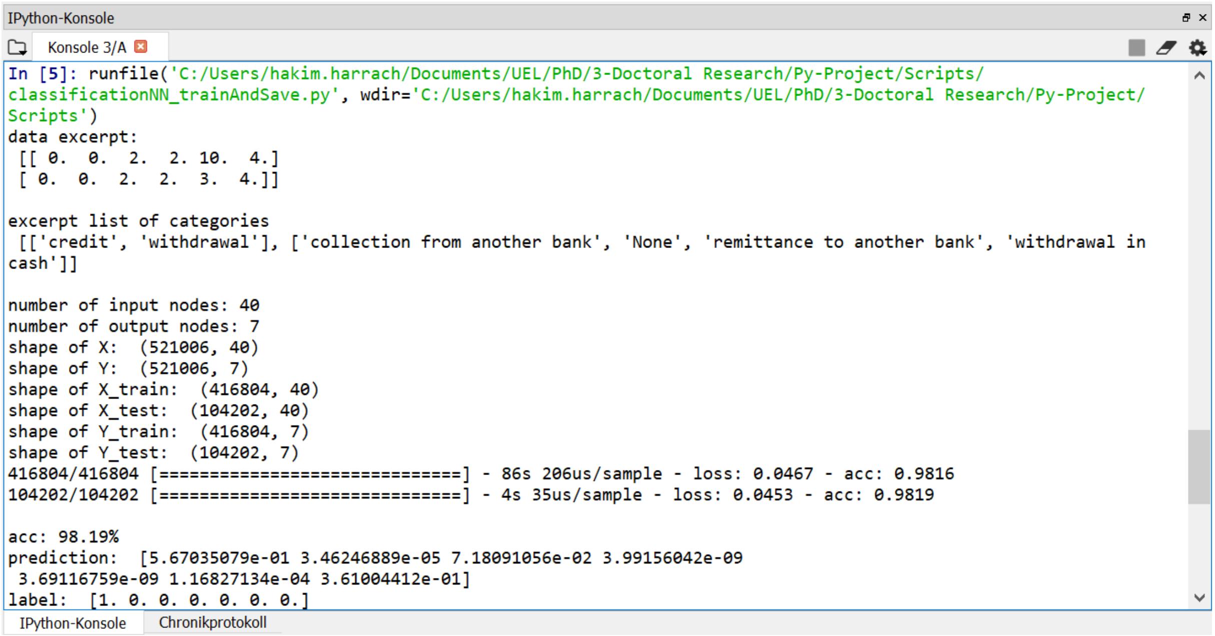
The neural network classification output from the processed dataset is given in figure 5-57 below. The details of how to obtain the introduced graph in chapter 4 are shown by the summary of the Python results. The computation is described in terms of the input parameters and operations in the structure of a directed graph. The figure shows that the computation is performed on 40 input nodes and seven output nodes. The data which moves between these nodes are known as tensors consisting of multi-dimensional arrays which are explained and implemented in the previous chapter through the data pre-processing phase.

Current research uses a standard multiclass classification problem as the basis to demonstrate the effect of predicting categorised payment transactions. The research experiment is configuring the problem via the “X” and “Y” argument as can be seen in the output screen below. Therefore, I have defined the research experiment for the abstract computation based on the input (X) attributes [['credit', 'withdrawal'], ['collection from another bank', 'None', 'remittance to another bank', 'withdrawal in cash']] and produce the output (Y) attribute ['insur. payment', 'payment for statement', 'interest credited', 'sanction interest if negative balance', 'household', 'old-age pension', 'loan payment'] during the complex operation. Both variables “X” and “Y” must be one hot encoded. This is necessary so that the constructed model can learn to predict the probability of an input example (e.g. 5 features) belonging to each of the seven classes. I have applied the OneHotEncoder() function from sklearn.preprocessing to do this. After that, the large-scale dataset was split into 80% training dataset labelled as “X_train” and “Y_train”, and 20% test dataset labelled as “X_test” and “Y_train” to evaluate the model. The train_test_split() function from sklearn.model_selection ties these elements together and returns the train and test sets in terms of the input and output elements.

Next step of the modelling process is that current research has extracted from the training dataset the number of input variables to configure the first layer by 40 and the second layer by 20, and the number of target classes to configure the output layer. The defined MLP model uses the rectified linear activation function for the first and second layer. The output layer of the model uses the softmax activation function to predict a probability for each payment category (target class). The number of nodes in the hidden layer will be provided via an argument called “len(labelElements)“. The model will be optimised using an adaptive learning rate optimisation algorithm ‘Adam’ that has been designed specifically for training deep-neural networks to update networks weights iterative based in training data, and a categorical cross-entropy loss function will be used to measure the performance of the multiclass classification model more precisely. The model will be fit for one training epoch and for a batch size of 10, then the model will be evaluated on the test dataset.

When tying these elements together, the model.evaluate() function takes the number of nodes and the pre-processed dataset as arguments and returns the history of the training loss at the end of the single epoch and the accuracy of the final model on the test dataset. The prediction score, the loss function score and the accuracy score for the training and test configuration will be printed separately, and the learning curves of training and test accuracy as well as the corresponding losses based on cross-entropy with the described configuration will be plotted. The full Python code is provided in appendix E.2.

Running the dataset samples fits the model very quickly on the CPU compared with the research experiments settled down in R Studio. The model is evaluated, reporting the classification accuracy on the train and test sets of about 98.16% and 98.19% respectively. The loss function calculates on the train and test sets values about 4.67 and 4.53. Note that the specific results may vary by the repeatedly pass through of the modelling given the stochastic nature of the training algorithm. Finally, the printed screen below shows the prediction results for every single output variable. The results lead to the conclusion that the model learn the problem perfectly compared with the outcomes we have produced with the research experiments implemented in R Studio.



```
In [5]: runfile('C:/Users/hakim.harrach/Documents/UEL/PhD/3-Doctoral Research/Py-Project/Scripts/
classificationNN_trainAndSave.py', wdir='C:/Users/hakim.harrach/Documents/UEL/PhD/3-Doctoral Research/Py-Project/
Scripts')
data excerpt:
 [[ 0.   0.   2.   2. 10.   4.]
 [ 0.   0.   2.   2.   3.   4.]]

excerpt list of categories
 [['credit', 'withdrawal'], ['collection from another bank', 'None', 'remittance to another bank', 'withdrawal in
cash']]

number of input nodes: 40
number of output nodes: 7
shape of X:  (521006, 40)
shape of Y:  (521006, 7)
shape of X_train:  (416804, 40)
shape of X_test:  (104202, 40)
shape of Y_train:  (416804, 7)
shape of Y_test:  (104202, 7)
416804/416804 [==============================] - 86s 206us/sample - loss: 0.0467 - acc: 0.9816
104202/104202 [==============================] - 4s 35us/sample - loss: 0.0453 - acc: 0.9819

acc: 98.19%
prediction:  [5.67035079e-01 3.46246889e-05 7.18091056e-02 3.99156042e-09
 3.69116759e-09 1.16827134e-04 3.61004412e-01]
label:  [1. 0. 0. 0. 0. 0. 0.]
```

Figure 5-57: Summary of the Python results - Train and Save the modelling results of the neural network classification (classificationNN_trainAndSave.py)

The resulting output of the computation from modelling the neural network classifications shows that the algorithm is printing very good forecasts of the next transaction category, which goes hand in hand with the selection of the five features. The aim of the forecast is, in particular, to anticipate outliers in the categorised cash flows to estimate the credit risks even better, or further optimise cross-sell opportunities. For this purpose, the neural network can train on exactly these connections between past payments, payment behaviour and payment type.

Crucial here is that the information about past payments, payment behaviour and payment type for the near future already exist and can be incorporated into the forecast, which also could improve the credit scoring as well as cross-selling models, evaluated through the first and second research projects.

Finally, the performance of the model on the prepared train and test sets recorded during training is graphed using a two-line plot: One for each for the learning curves of the cross-entropy loss and one for the classification accuracy. The created figure 5-58 below shows both two-line graphs. The visualised results are based on the described model configuration above (40 nodes for the first layer and 20 nodes for the second layer) where I have learned the model over the 300 training epochs. Finally, both plots of the research experiments suggest that the model has a good fit on the problem.

From the plot of accuracy, it is observable that the model should not be trained furthermore as the trend for accuracy on both datasets is still stagnating for a very wide range of epochs. However, it is also notable that the model has not yet over-learned the training dataset, showing comparable skill on both datasets. The model accuracy for both datasets is varying between 98.14% and 98.22% during the entire measurement period. However, the predicted data points for the train sets are located in the upper section of the model accuracy plot. The analysis shows that increasing the number of epochs over 150 results in peak amplitudes when learning the test dataset.

From the plot of loss, it can be noticed that the model has comparable performance on both train and validation datasets (labelled test). Since these parallel plots start to be steady and consistent, it might be a sign to stop training at an earlier epoch because we can except any significant changes in the modelling behaviour. The model cross-entropy loss for both datasets is varying between 4.4% and 5.1% during the entire measurement period. However, the predicted data points for the train sets are located in the lower section of the model loss plot. Surprisingly, test loss within the first 100 epochs shows signs of initially doing well before leaping up, thus suggesting that the learned model is likely stuck with a sub-optimal set of weights rather than over-fitting the test dataset. Regarding the peak amplitudes occurring when learning the test dataset over 150 epochs, we can observe the same model behaviour in the model loss such as found in the model accuracy.

From the both plots, the model accuracy as well as model loss, we can see that as the number of epochs is increased, there is a slightly falling trend in the model accuracy and a slightly increase in the model loss. None the less, the entire model performance is excellent when learning the training dataset, so that there is very little scope to optimise the model. The two-line plots show the direct relationship between model capacity, as pre-defined by the number of nodes in the hidden layer (e.g. 40 nodes for the first layer and 20 nodes for the second layer) and the model's ability to learn over a defined period of epochs.

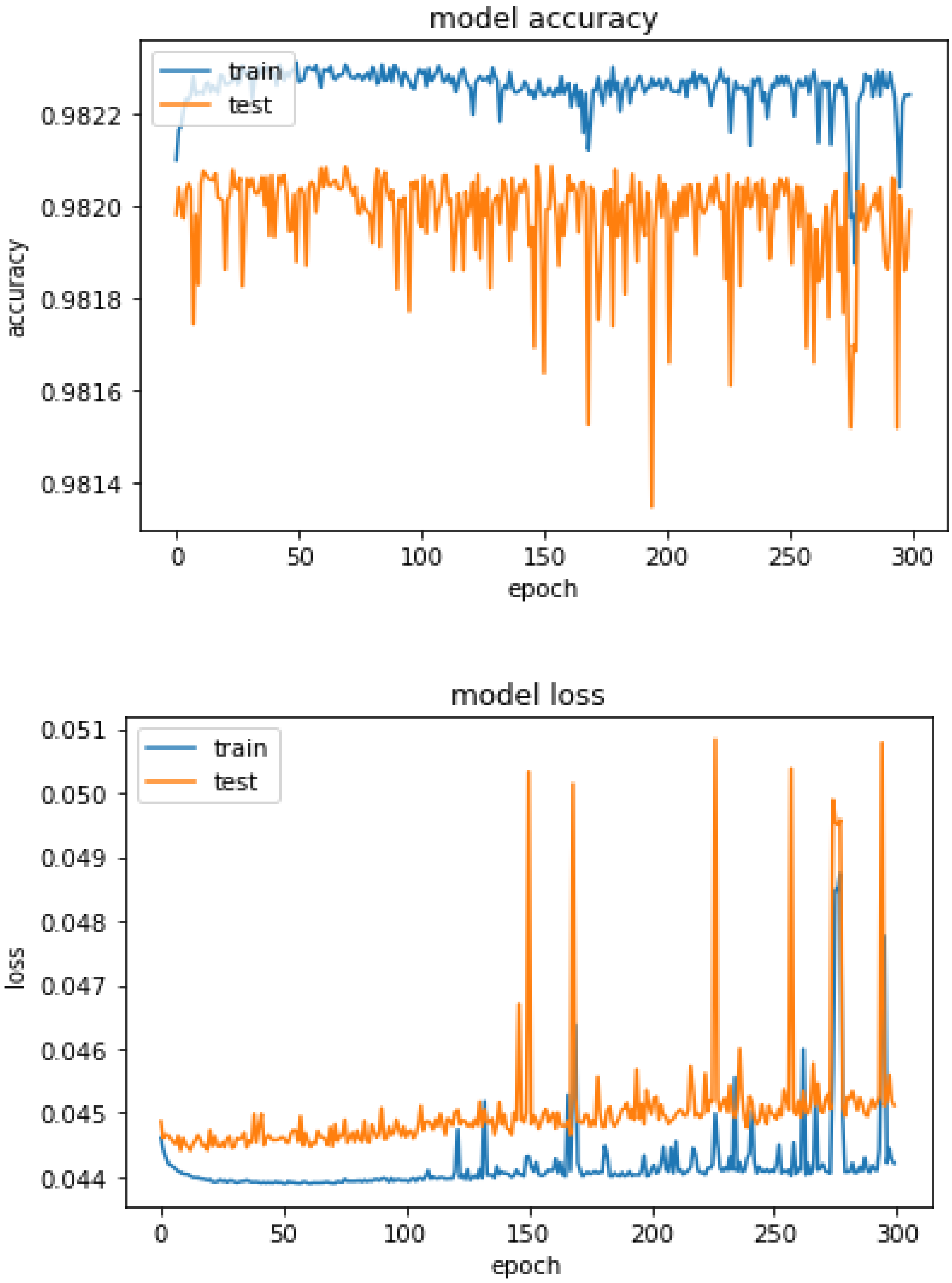
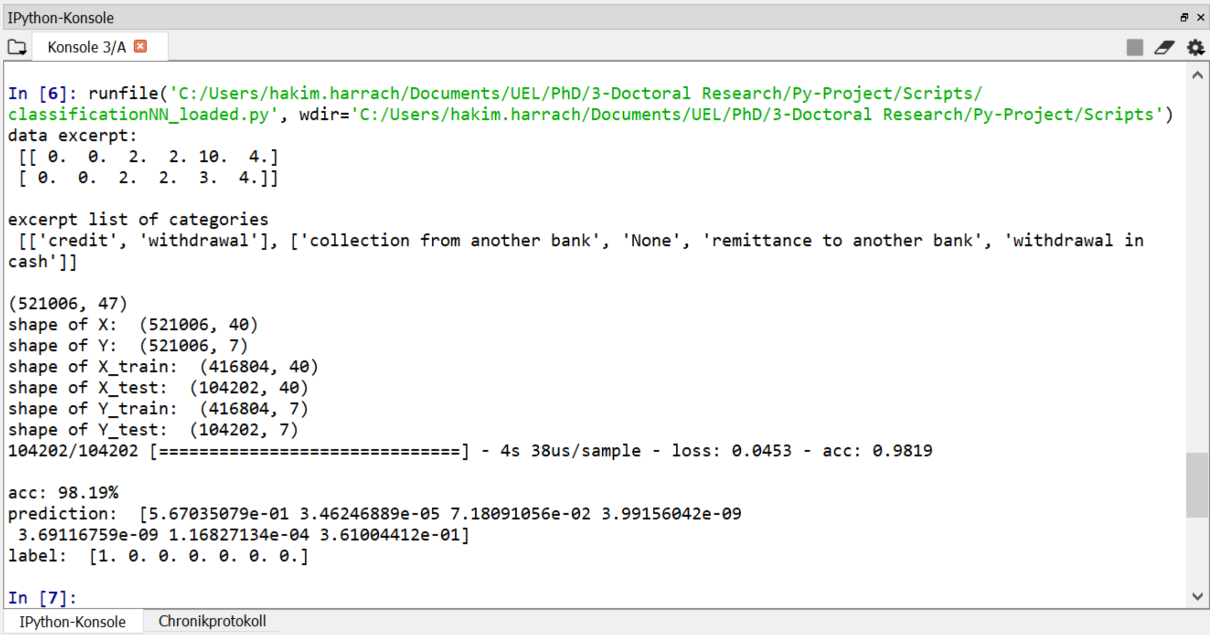


Figure 5-58: Two-line plots for learning curves of loss and accuracy of the model in predicting categorised payment transactions

The following Python output screen can be used to train and learn the model as long as it is necessary to optimise and enhance the prediction results. The model is evaluated, reporting the classification accuracy on the test set of about 98.19% respectively with a loss value on the test set about 4.53.



```
In [6]: runfile('C:/Users/hakim.harrach/Documents/UEL/PhD/3-Doctoral Research/Py-Project/Scripts/
classificationNN_loaded.py', wdir='C:/Users/hakim.harrach/Documents/UEL/PhD/3-Doctoral Research/Py-Project/Scripts')
data excerpt:
 [[ 0.   0.   2.   2. 10.   4.]
 [ 0.   0.   2.   2.   3.   4.]]

excerpt list of categories
 [['credit', 'withdrawal'], ['collection from another bank', 'None', 'remittance to another bank', 'withdrawal in
cash']]

(521006, 47)
shape of X:  (521006, 40)
shape of Y:  (521006, 7)
shape of X_train:  (416804, 40)
shape of X_test:  (104202, 40)
shape of Y_train:  (416804, 7)
shape of Y_test:  (104202, 7)
104202/104202 [==============================] - 4s 38us/sample - loss: 0.0453 - acc: 0.9819

acc: 98.19%
prediction:  [5.67035079e-01 3.46246889e-05 7.18091056e-02 3.99156042e-09
 3.69116759e-09 1.16827134e-04 3.61004412e-01]
label:  [1. 0. 0. 0. 0. 0. 0.]

In [7]:
```

Figure 5-59: Summary of the Python results - Prediction results on the trained neural network (classificationNN_loaded.py)

In summary, it emerged that for both analysis procedures developed for the last research project, apply the neural network algorithm through the pre-processed dataset using the R Studio tool or Python including TensorFlow framework, the designed research setup using Python has achieved most suitable predictions (with +98% accuracy) compared with the research setup using R Studio and its package "neuralnet". Note that the accuracy of the predictions, modelled with R Studio, is varying between 0.2% and 90.7% depending on the underlying smaller-sized dataset. One of the main advantages using Python is being able to construct powerful models on large-scale datasets and providing more accurate prediction results.

5.3. Key findings and discussion of research outcomes

This last section of the chapter presents principal findings from the three explored research projects. The findings from the current research can be divided into two groups: Descriptive and predictive evaluation results. The primary difference between descriptive and predictive research results lies in the fact that descriptive results provides insights about the customer behaviour which can be observed during a specific time of period, while the predictive results delivers insights on future transactional payment behaviour (in many cases in areas where no research in the literature existed, especially not based upon the Berka dataset). Both areas of the research results address a range of evaluation results that covers the analysis of customer behavioural patterns in transaction payment streams.

Figure 5-60 below illustrates how the research results identified through a descriptive analysis as well as predictive analysis are integrated to explore future transactional payment behaviour as a whole. According to this, the descriptive results, namely methods to identify most important variables for the predictive model or methods to discover transactional payment patterns, are placed in the predictive results, which are in the context of modelling future transactional payment behaviour through payment data streams.

There are some overlapping findings between the first and second research project; therefore, the presented findings are aligned to every single research project separately. The main results shown in this section have already been processed and evaluated, and the raw findings are presented in the previous sections as well as appendix C and D.

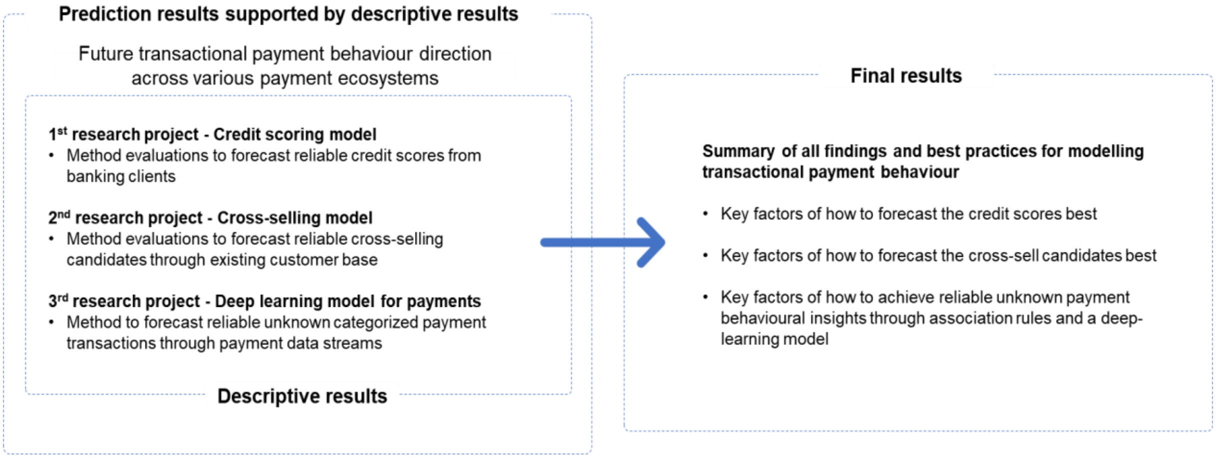

Prediction results supported by descriptive results
Future transactional payment behaviour direction across various payment ecosystems
1st research project - Credit scoring model
• Method evaluations to forecast reliable credit scores from banking clients
2nd research project - Cross-selling model
• Method evaluations to forecast reliable cross-selling candidates through existing customer base
3rd research project - Deep learning model for payments
• Method to forecast reliable unknown categorized payment transactions through payment data streams
Descriptive results
Final results
Summary of all findings and best practices for modelling transactional payment behaviour
• Key factors of how to forecast the credit scores best
• Key factors of how to forecast the cross-sell candidates best
• Key factors of how to achieve reliable unknown payment behavioural insights through association rules and a deep-learning model

Figure 5-60: Integration of descriptive results and predictive results to unleash future transactional payment behaviour

5.3.1. Summary of the main results from the 1st research project

This section summarises the key findings of the first research project and discusses the major results in case of evaluating the creditworthiness of a bank customer based on their generated transactional payment data and predict reliable credit scores efficiently. To identify the best-fitting machine learning algorithm, the research analysed various supervised classification algorithm over the derived PKDD financial dataset, so called Berka, and achieved with the Two-Class Decision Forest better accuracy for the first research project compared with the existing data mining methods in the literature.

Berka and Rauch (2007) are recommend that various conditions should be fulfilled in a successful research process: Cooperation with domain experts should remain at the forefront, usage of external data, usage of powerful pre-processing methods, apply simple machine learning models first, visualise the evaluation results to foster understandability and acceptability and finally assess the return of investment (ROI) of the deployed models. Hence, a “successful data mining project should be driven by the application needs and results should be tested quickly” (Kovalerchuk and Vityaev, 2010). All of these recommendations and requirements are in line with the underlying practical-oriented research work since both assumptions have been an essential part of the research approach. The constructed research experiments demonstrate that the applied Two-Class Decision Forest algorithm for the credit scoring case is reasonable and applicable well with good results. The scaled research experiments further underline that the approach may work well as the quality of the dataset will increase. At least, the results indicate that the research has processed a representative dataset of suitable data quality since the evaluation results are determined objectively and their findings have been traceable as well as the explanations of the key results are plausible.

Clearly, the evaluation results benefit financial institutions by highlighting issues (i.e. creditworthiness of a selected customer base), areas for improvement (i.e. providing combined product offerings), vision into likely outcomes by predicting future payment behaviour, and liquidity problems within the payment behaviour of their customers. Likewise, key findings can rely on behavioural patterns occurred in the payment dataset; for example, customers with an average of 47 years old or 37 years old receiving a bad credit scoring. Note that the patterns are interesting if they are unexpected and useful in understanding the hidden drivers of customers transactional payment behaviours.

As can be seen with the exhaustive explanation of the evaluation results for determining the best-fitting model, there is no universal measure for assessing the data model performance. At the end, the best model is the one which will maximise the benefits of any banks and the current results should be taken into account when considering how to accomplish that business goals.

To summarise, in order to rate all of the applied algorithms, a clear performance metric is required. As a result, the research has developed a range of measurement criteria which were visualised and comprehensible to all applied algorithm. The performance ranking results including the performance characteristics are presented in the table 5-20 below.

Supervised learning algorithm	Accuracy	Precision	Recall	F1 Score	AUC	**RANK**
Two-Class Decision Forest	**0.893**	**0.924**	**0.901**	**0.912**	**0.956**	**1**
Two-Class Neural Network	0.858	0.872	0.901	0.886	0.910	2
Multiclass Neural Network	0.853	0.853	0.853	-	-	3
Two-Class Logistic Regression	0.827	0.854	0.868	0.861	0.903	4
Two-Class Support Vector Machines	0.792	0.823	0.843	0.833	0.869	5

Table 5-20: Performance rank of the best-fitting machine learning algorithms - credit scoring

The research study discusses the descriptive and predictive results through the distinct research project and the meaning of every model performance characteristics. An extensive set of research experiments has been conducted to show that the applied supervised learning algorithm outperforms the best algorithm results with the ‘Two-Class Random Forest’, by a F1 score of 0.912 as well as an AUC score of 0.956, and by the original pre-processed dataset consisting of following input variables (‘status’, ‘permanentOrders’, ‘AvgIncome’, ‘AvgExpenses’, ‘AvgBalance’, ‘Sex’, ‘Age’, ‘NoInhabitants’, ‘UrbanRatio’, ‘AverageSalary’, ‘Unemployment95’, ‘Unemployment96’, ‘CrimeRatio95’, ‘CrimeRatio96’, ‘EnterpreneursRatio’). It also has provided excellent response charts with respect to the ROC, precision and recall, and lift charts given in appendix D.2.

Moreover, the research experiment provides new insights into the transactional payment behaviour of bank customers by evaluating the relationship between the selected input variables. For instance, a variety of data mining techniques were applied through the descriptive analysis to contribute a clearer understanding of customer behavioural insights.

The research also discussed the performance of the applied ‘Two-Class Decision Forest’ algorithm in detail by comparing the original dataset against the optimised more cost-sensitive dataset which can also be used in predicting a real credit scoring case. The AUC as an effective measurement of accuracy and the ROC curves plays a central role in evaluating performance ability of every applied supervised learning algorithm to assess the best-performing predictive model, finding the optimal cut of non-important input variables based on a variable importance analysis, and comparing a range of supervised learning algorithm outcomes to diagnostic the best one with the objective to perform each optimised dataset on the same best-performing algorithm, the Two-Class Decision Forest. The predictive results showed the reciprocal relationship between the credit applicants of false positive and of false negative results from the different research experiments whenever the threshold of 0.5 decreases or increases

The study demonstrates that a cost-sensitive data model can be developed in a way that the applied ‘Two-Class Random Forest’ algorithm outperforms the best prediction results by a F1 score of 0.967 instead of 0.956, and by a minimum subset of three relevant input variables (‘AvgBalance’, ‘Age’ and ‘AvgIncome’ or ‘AvgExpenses’) instead of the entire original dataset consisting of fifteen input variables. It also has excellent response charts with respect to the ROC, precision and recall, and lift charts provided in appendix D.2. During the descriptive analysis for the different research experiments, variable importance analysis has become a popular method for evaluating the statistical significance of the variable within the entire dataset with respect to filter out the best-performing predictive model on a minimum subset of input variables. The most desirable property of the varImp() results is that the importance of a variable is calculated by MeanDecreaseGini score based on a Random Forest algorithm. Finally, the prediction results also fit the theory since the predictive model applied on an optimised dataset demonstrates much better prediction power than on the original dataset. The derived performance metric to measure the model performance for identifying the most cost-sensitive data model, such as the area under the curve (AUC) determines the inherent ability of the test to discriminate between the bad and good credit applicant. Using the defined performance metric as a measure of a diagnostic performance, one can compare individual research experiments or judge whether the various research experiments can improve diagnostic model performance

Two-Class Decision Forest algorithm	Accuracy	Precision	Recall	F1 Score	AUC	**Input variable**
Applied on the original dataset	0.893	0.924	0.901	0.912	0.956	status, permanentOrders, AvgIncome, AvgExpenses, AvgBalance, Sex, Age, NoInhabitants, UrbanRatio, AverageSalary, Unemployment95, Unemployment96, CrimeRatio95, CrimeRatio96, EnterpreneursRatio
Applied on a subset of the original dataset	**0.918**	**0.973**	**0.892**	**0.930**	**0.967**	AvgBalance, AvgIncome, AvgExpenses, Age
Applied on a subset of the original dataset	**0.918**	**0.973**	**0.892**	**0.930**	**0.967**	AvgBalance, AvgExpenses, Age
Applied on a subset of the original dataset	**0.918**	**0.973**	**0.892**	**0.930**	**0.967**	AvgBalance, AvgIncome, Age

Table 5-21: Performance comparison of various cost-sensitive data models - credit scoring

The presented AUC and ROC curves hold strong interest, since they provide meaningful interpretations in case of credit applicants. The key findings from the performance graphs can be interpreted in a way that the visualised model results generally depend on the predictor variables. The other interpretation is that the more important the input variables, the better the model accuracy.

The credit defaults in designing of diagnostic research experiments concern false positive and false negative. A broad spectrum of cases is probably required to evaluate the model performance and a broad spectrum for designated predictive models. For example, the research experiments should be deducted with model performance both for existing and extending performance characteristics. Thus, the developed performance metric may include further alternative and varying measurements. The second concern is the false negatives. The values lead in a falsely low or high rates and thus results in a falsely low or high AUC. The assessment of both can be manifested in different ways. For instance, the current research work used MS Azure ML to scale the computed results to diagnose the predictions in the right way.

Despite the meaningful interpretation of the model performance results, it may still be argued that an optimised data model arises from an iterative data science process where the entire dataset initially holds strong relevance. Thus, the research design includes a range of theoretical concepts on which the results are built, but there is no other existing evidence-driven approach in the literature to analyse the depth of the defined research scope.

Finally, current research suggests that the evaluation outcomes can refer to customer payment behavioural patterns to sustain and improve current business operations in financial institutions and deliver increased efficiency and productivity, finally, in case of deploying the developed supervised learning algorithm into an enterprise landscape. Therefore, the current research work is a pioneering step towards optimising supervised learning algorithm through transactional behavioural pattern recognition.

5.3.2. Summary of the main results from the 2nd research project

This section summarises the key findings of the second research project and discusses the major results in case of evaluating cross-selling opportunities, for instance, promoting a credit card for bank customer based on a deeper level analysis of a customer’s transactional behaviour or payment practice. To identify in this context the best-fitting machine learning algorithm, the research analysed various supervised classification algorithm over the derived Berka dataset and achieved with the Two-Class Decision Forest better accuracy for the second research project compared with the existing data mining methods in the literature.

Previous literature has emphasised that unsupervised and supervised learning techniques allow banks to truly understand their customers and provide them with a personalised service with targeted product offerings. For instance, customer segmentation through cluster analysis, an unsupervised learning technique, banks can discover distinct groups in their customer base and see similarities over several dimensions. Unlike supervised learning, they do not need to define what characteristics the computer should be looking for. This way, banks can segment in ways traditional analytics would not allow.

The underlying research study pre-processed the Berka dataset to gain new behavioural knowledge about bank customers with respect to customer segmentation discoveries (e.g. the majority of the credit card owner are presenting a good loan status) through an exhaustive descriptive analysis which results the research then has been used to build predictive, supervised models. The analysis shows that algorithms can produce personalised views of the most suitable product credit card usage for each customer, which might be helpful for cross-selling and up-selling business activities. Since algorithms learn, they recognise changes in behaviour and respond in a timely manner. Consequently, the revenue can increase from successful identification of cross-sell and up-sell opportunity changes in customer preferences in real-time and therefore automatically adjust product recommendations.

Taking these into account the underlying practical-oriented research work contribute with the novel research approach to fulfil future business needs. As a result, the constructed research experiments demonstrate that the applied Two-Class Decision Forest algorithm for the cross-selling case is reasonable and applicable well with good results. The scaled research experiments further show that the approach may work well as the quality of the dataset will increase. At least, the results indicate that the research has processed a representative dataset of suitable data quality since the evaluation results are determined objectively and their findings can be addressed clearly as well as the explanations of the key results are plausible.

The results might suggest that a marketing department of banks can benefit from the novel approach by identifying promotion candidates for their products more data-driven by predicting cross-sell candidates based on their payment data streams. Likewise, key findings can rely on behavioural patterns occurred in the payment dataset; for example, the account owner is female or male with a probability of at least 50% and with an average age of 40.1, and a minority (around 5%) of the entire processed transactions are assigned to 'issuance after transactions', which indicates late payments. Clients with an age limit vary between 33 to 51 years have a delay in their payments, and a majority of these late payments are strongly associated to specific 'CrimeRatio96' values.

Personalised, improved customer offerings (e.g. providing specific marketing and promotion strategies for selected districts) and the speed of service will increase since banks would gain more deep insights about their customer base. For instance, every second bank account will be affected when customising a credit card promotion because more than 55% of the entire customer base receiving the same base salary volume.

As can be seen with the exhaustive explanation of the evaluation results for determining the best-fitting model, there is no unique assessment for measuring the data model performance. Ultimately, the best model is the one that increases the profits of any banks and the current results should be taken into account when considering how to foster these business goals.

Supervised learning algorithm	Accuracy	Precision	Recall	F1 Score	AUC	**RANK**
Two-Class Decision Forest	**0.807**	**0.736**	**0.79**	**0.762**	**0.876**	**1**
Two-Class Decision Jungle	0.744	0.656	0.728	0.69	0.837	2
Two-Class Locally-Deep Support Vector Machines	0.733	0.689	0.765	0.725	0.816	3
Two-Class Logistic Regression	0.729	0.692	0.556	0.616	0.814	4

Two-Class Neural Network	0.734	0.627	0.79	0.699	0.806	5
Multiclass Neural Network	0.72	0.72	0.72			6

Table 5-22: Performance rank of the best-fitting machine learning algorithms - cross-selling

To summarise, in order to rate all of the applied machine learning algorithms, a clear performance metric such as applied in the first research project is also required for the second research project. As a result, the research has again developed a range of measurement criteria which were visualised and comprehensible to all applied algorithm. The performance ranking results including the performance characteristics are presented in the table 5-22 above.

The research study discusses the descriptive and predictive results through the research project and the meaning of every model performance characteristics. An extensive set of research experiments has been implemented to show that the applied supervised learning algorithm outperforms the best algorithm results with the 'Two-Class Random Forest', by a F1 score of 0.762 as well as AUC score of 0.876, and by the original pre-processed dataset consisting of following input variables ('Frequency', 'Sex', 'Age', 'NoInhabitants', 'UrbanRatio', 'AverageSalary', 'Unemployment95', 'Unemployment96', 'CrimeRatio95', 'CrimeRatio96', 'EnterpreneursRatio', 'AvgIncome', 'AvgExpenses', 'AvgBalance', 'Cardholder'). It also has provided excellent response charts with respect to the ROC, precision and recall, and lift charts given in appendix D.4.

In addition, the research experiment provides new insights into the transactional payment behaviour of bank customers by evaluating the relationship between the selected input variables. For instance, a variety of data mining techniques were applied through the descriptive analysis to contribute a clearer understanding of customer behavioural insights. The research study implemented a new innovative approach in pre-processing the raw dataset to gain more valuable behavioural insights across advanced visualisation of the outcomes and by computing a reliable score as a measure for credit card ownership.

The research also discussed the performance of the applied 'Two-Class Decision Forest' algorithm in detail by comparing the original dataset against the optimised more cost-sensitive dataset which can also be used in predicting a real cross-selling case.

Therefore, the AUC as an effective measurement of accuracy and the ROC curves plays a central role in evaluating performance ability of every applied supervised learning algorithm to assess the best-performing predictive model, finding the optimal cut of non-important input variables based on a variable importance analysis, and comparing a range of supervised learning algorithm outcomes to diagnostic the best one with the objective to perform each optimised dataset on the same best-performing algorithm, the Two-Class Decision Forest. The predictive results showed the reciprocal relationship between the cross-sell candidates of false positive and of false negative results from the different research experiments whenever the threshold of 0.5 decreases or increases.

The study demonstrates that a cost-sensitive data model can be developed in a way that the applied ‘Two-Class Random Forest’ algorithm outperforms the best prediction results by a AUC score of 0.897 instead of 0.876, and by a minimum subset of four relevant input variables (‘Age’, ‘AvgExpenses’, ‘AvgBalance’, ‘AvgSalary’) instead of the entire original dataset consisting of fifteen input variables. It also has excellent response charts with respect to the ROC, precision and recall, and lift charts provided in appendix D.4. During the descriptive analysis for the different research experiments, variable importance analysis has again become a popular method for evaluating the statistical significance of the variable within the entire dataset with respect to filter out the best-performing predictive model on a minimum subset of input variables. Regarding the varImp() results, the analysis again calculated the importance of a variable by MeanDecreaseGini score which is based on a Random Forest algorithm. Finally, the prediction results also fit the theory since the predictive model applied on an optimised dataset demonstrates much better prediction power than on the original dataset.

The derived performance metric to measure the model performance for identifying the most cost-sensitive data model, such as the area under the curve (AUC) determines the inherent ability of the test to discriminate between the bad and good cross-selling candidates for credit card promotions. Using the defined performance metric as a measure of a diagnostic performance, one can compare individual research experiments or judge whether the various research experiments can improve diagnostic model performance.

The presented AUC and ROC curves hold strong interest, since they provide meaningful insights in case of cross-selling candidates. The key findings from the performance graphs can be interpreted in a way that the visualised model results again depend on the predictor variables. The other interpretation is that the more important the input variables, the better the model accuracy in general.

Two-Class Decision Forest algorithm	Accuracy	Precision	Recall	F1 Score	AUC	**Input variable**
Applied on the original dataset	0.807	0.736	0.790	0.762	0.876	Frequency, Sex, Age, NoInhabitants, UrbanRatio, AverageSalary, Unemployment95, Unemployment96, CrimeRatio95, CrimeRatio96, EnterpreneursRatio, AvgIncome, AvgExpenses, AvgBalance, Cardholder
Applied on a subset of the original dataset	0.819	0.780	0.657	0.713	0.895	Age, AvgIncome, AvgExpenses, AvgBalance, AvgSalary
Applied on a subset of the original dataset	**0.838**	**0.836**	**0.657**	**0.736**	**0.897**	Age, AvgExpenses, AvgBalance, AvgSalary
Applied on a subset of the original dataset	0.804	0.721	0.700	0.710	0.877	Age, AvgIncome, AvgBalance, AvgSalary

Table 5-23: Performance comparison of various cost-sensitive data models - cross-selling

The credit card ownerships in designing of diagnostic research experiments concern false positive and false negative. A broad spectrum of cases is probably required to evaluate the model performance and a broad spectrum for designated predictive models. For example, the research experiments should be deducted with model performance both for existing and extending performance characteristics. Thus, the developed performance metric may include also further alternative and varying measurements. The second concern is the false negatives. The values lead in a falsely low or high rates and thus results in a falsely low or high AUC. The assessment of both can be manifested in different ways. For instance, the current research work used MS Azure ML to scale the computed results to diagnose the predictions in the right way.

Despite the meaningful interpretation of the model performance results, the research work showed that an optimised data model arises from an iterative data science process where the entire dataset initially holds strong relevance. Thus, the research design includes a range of theoretical concepts on which the results are built, but there is no other existing evidence-driven approach in the literature to analyse the depth of the defined research scope.

Finally, the novelty and major contribution to research is that the study has developed a new data-driven approach through the data pre-processing stage to identifying the most likely cross-sell candidates for promoting a credit card. At the same time, I have investigated the questions such as what the best algorithm is to predict cross-selling candidates and how a cost-sensitive data model should look like in that context. However, current research suggests that the evaluation outcomes can refer to customer payment behavioural patterns gained from the first research project and combine the further gained behavioural insights from the second research project to sustain and improve current business operations in financial institutions and deliver increased efficiency and productivity, finally, in case of deploying also the developed supervised learning algorithm into an enterprise landscape. However, based on the results both research projects are complementary rather than in competition with each other; for example, someone with a bad credit scoring might be not a proper cross-sell candidate for credit card promotions. In conclusion, the current research work is a pioneering step towards optimising supervised learning algorithm through transactional payment data streams.

5.3.3. Summary of the main results from the 3rd research project

This section discusses the key findings from the last research project and relates to the evaluation results in the context along with the key outcomes of the other both research projects. The study also demonstrates how these distinct research projects can correlate to each other in case of analysing customer behaviour based on historical transactional payment data streams to create more customer value through personalised product or service offerings.

The research focus was set on categorised payment transactions occurring in the payment data streams and how the analysis can unleash more useful payment behavioural insights from this. In line with the current research objectives that Spenke and Beilken (1999) discovered some key findings from the Berka dataset in another context, such as “if at least one transaction of an account was a sanction interest (because of a negative balance) there have been problems with the loan in 90%”, and “if the average balance of an account is high, problems are rare” by having loan defaults.

However, the research experiments present preliminary work for further research, showing the promise of the approach for modelling transactional behaviour based on payment categories, but leave some questions open about the predictability of uncategorised transactional payment data. An experiment on this type of research project has mainly revealed the need for more efficient implementation: the computational and data-driven complexity is relatively high and currently precludes some meaningful comparisons to other research. Consequently, many interesting questions remain open. What is the best algorithm for this type of issue? The research proposes with a neural network, market basket analysis and frequency analysis three methods: The association rules and their graph-based visualisations are clearly more expressive than the other evaluation results, but all selected methods appear to learn well on the considered research project of clustering customer behaviour on transactional payment streams. Among these three mining methods, there was a tendency of association rules to perform somewhat better than the other algorithms in the experiments of this research project. However, further experimentations on more highly qualified datasets are necessary, even for this type of research problem formal evaluation results would be desirable. It is clear that there are hidden connections between the three different mining approaches for learning from the same structured dataset, even though the single applied approaches are not directly connected with one entire statistical learning algorithm. Current research expects that some further investigation into those approaches may advance the field of machine learning in general.

The applied association rule mining algorithms generated a small number of association rules which supports the current research to analyse the major research questions and understand the generated rules and detect comprehensive insights in transactional payment behaviour. In this research work, I have presented several visualisation techniques implemented with the help of the R-Package “arulesViz” which can be applied to explore and present key patterns in the transactional dataset. This graph-based visualisation is especially useful to present found descriptive results because it is easy to understand for non-data scientists.

Table 5-24 below summarising the key results from the applied market basket analysis (MBA) on categorised payment transactions using association rules package from R Studio. The method has generated overall certain behavioural payment patterns, which marketing departments of banks can be integrating in their advertisement campaign.

Rule	lhs	rhs	support	confidence	lift	count
1	payment for statement	interest credited	0.190	0.998	4.608	155584
2	interest credited	payment for statement	0.190	0.881	4.608	155584
3	interest if negative balance & payment for statement	interest credited	0.001	0.887	4.095	858
4	interest if negative balance & interest credited	payment for statement	0.001	0.910	4.762	858

Table 5-24: Overview of the four genereated association rules for categorised transactions

Ultimately the key outcomes from the MBA was to extract value from the payment transaction data by building up an understanding of the needs of the banking clients. However, this type of information is invaluable if the bank is interested in dedicated marketing activities such as cross-selling or targeted campaigns. Hahsler et al. (2011) emphasise that association rules become prominent as “an important exploratory method for uncovering cross-selling opportunities”. However, better results can be achieved with supervised learning algorithms than unsupervised learning algorithm since the MBA increases only a better understanding of the relationships between the payment categories that banking clients transfer. Based on that, new product pricing activities can be developed by the marketing department.

The following paragraphs also discusses the key outcomes of applying the neural network for predicting unknown categorised transactions, its implementation in R Studio as well as Python and their post training evaluations.

After introducing in chapter 2 and 4 the theoretical aspects of a neural network, the research work provides basic understanding of how machine learning techniques can be used to address the research issues. The aim is to predict the category of a payment transaction using information such as bucket of amount, type of transaction, mode of transaction etc. The significance of the variables is represented by weights of each connection. The constructed models were evaluated for accuracy and robustness. Therefore, current research starts to settle a range of research experiments in R Studio to compute the neural network and perform cross validation analysis. In cross validation, I have analysed the variation in model accuracy as the subsets of training set is changed as the research work considered various training sets across the entire dataset. For each research experiment unique samples are random picked, and the predictive model is calculated using R Studio. I have showed that the model accuracy depends on the training sets. Therefore, it emerged in case of using the model for prediction in R Studio that it is essential to check the robustness of performance through cross validation.

In addition, the research experiment shows that a neural network can generally solve the classification problem when predicting categorised payment transactions. It specifically proved that the current research work returns meaningful results after normalising the data. Moreover, using a confusion matrix to present the test results when measuring the accuracy of the developed model has been very helpful. A further research outcome was to determine the accuracy of the dependent variable "TransCharacterization".

However, the challenge with using neural networks was to set the right parameters: the number of hidden layers, the number of neurons in them, and the learning rate are only a few variables that must be set within the current research to achieve satisfactory prediction results in the first part of the research analysis. Special attention must initially be paid to the depth of the historical transaction data and the attribute selection of influencing factors and their modelling before the neural network can predict what the future transaction category will be. The optimisation of all of these parameters brings many time-consuming tests runs with it which should not be underestimated.

The work shows that the learning classifiers of the neural network are able to learn that automatically the analytically unknown connections between input and output patterns, especially in non-linear payment ecosystem, in order to be queried after training with unknown data. The classification results underline that there is no need for an identity between the taught and the requested pattern, although it is sufficient to have a similarity between the current payment data and the training data. This property of the neural networks for generalisation makes them so interesting for use in the classification and forecasting of blurred and noisy transactional payment data. The analysis results confirm that the selected input variables (e.g. 'TransType', 'TransMode', 'AmountBucket', 'BalanceBucket' and 'BankPartner') for the data model results in reasonable prediction results of unknown categorised transaction type with an accuracy rate of more than 98% learned on up to 300 epochs.

As the research results shows, explaining neural network outcome from the first part of the research analysis is much more difficult than explaining the outcome of simpler model. Therefore, due to the research design I want to take into account this factor too. The reason behind the usage of neural network is its practicability to infer meaning and detect behavioural patterns from complex transactional datasets. However, the research experiments also show that the applied neural networks in R Studio are not popular because they are computationally expensive, and they do not seem to yield better prediction results when compared with Python using the TensorFlow library from Google.

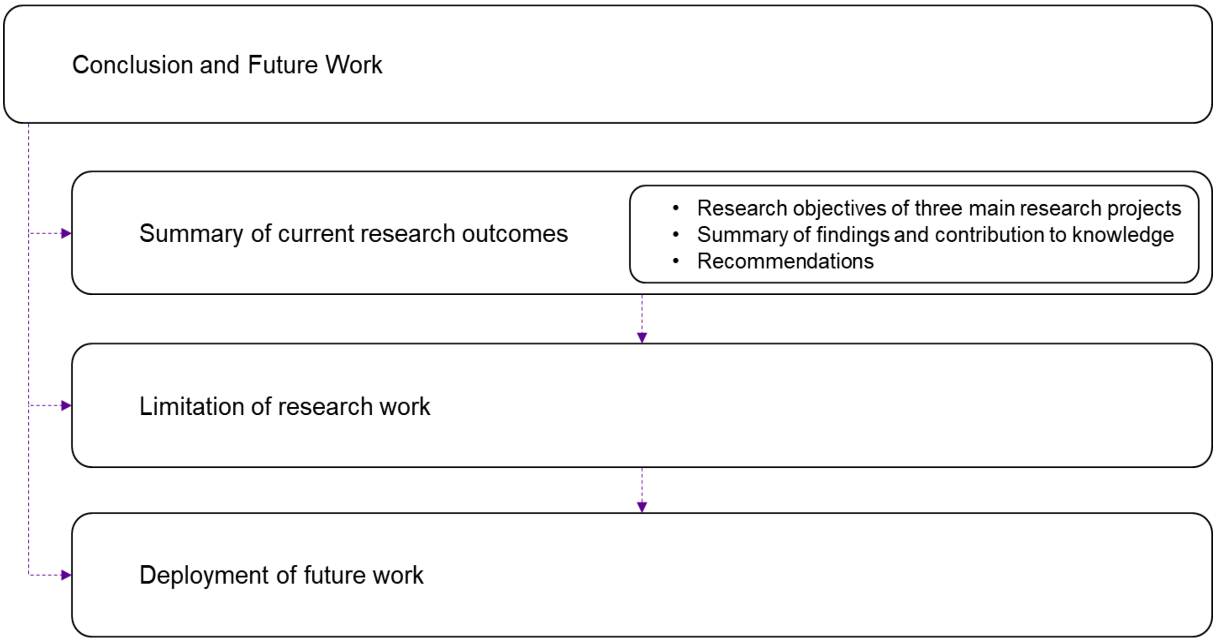
The various research experiments deployed in R Studio showed that the performance of neural network model is sensitive to training-test split and the model accuracy is dependent on the length of training set. It is an important result to note that providing predictions on larger datasets based upon Python lead to more accurate results. Finally, it must be highlighted that the current research work is a pioneering step towards transactional behavioural pattern recognition through market basket analysis, the use of unsupervised learning algorithm in analysing categorised payment transactions and developing a deep-learning model based on a neural net to solve this complex classification problem on large-scale transactional datasets

.

Chapter 6

6. Conclusion and Future Work

The last chapter deals with three main points, it summarises the key findings of the entire research including the results of every specified research project and their meanings, the limitations of the current piece of work and a possible further direction of future work.



Conclusion and Future Work
Summary of current research outcomes
• Research objectives of three main research projects
• Summary of findings and contribution to knowledge
• Recommendations
Limitation of research work
Deployment of future work

Figure 6-1: Structure of the conclusion chapter

6.1. Summary of current research findings

The current piece of work is the initial attempt to undertake a full research on how payment transactional data could be utilised for changing customer behavioural identification in case of credit applicants, cross-selling candidates and prediction of (un-)categorised transactional payments trends. The systematic literature review underlines the importance of how payment transactional data could be mined, explored and analysed to detect and predict changing customer behaviour. The research outcomes of every research project emphasise some of the causal factors as well as characteristics that influence customer behaviour.

The goal of this research study was to develop and evaluate various data mining models to classify and predict the transactional payment behaviour by modelling credit scores, cross-sell candidates and forecast categorised payment transactions as well as learn more about customer payments preferences. Therefore, a range of data mining techniques were used to develop different data models for every single research area as depict in the figure 6-2 below.

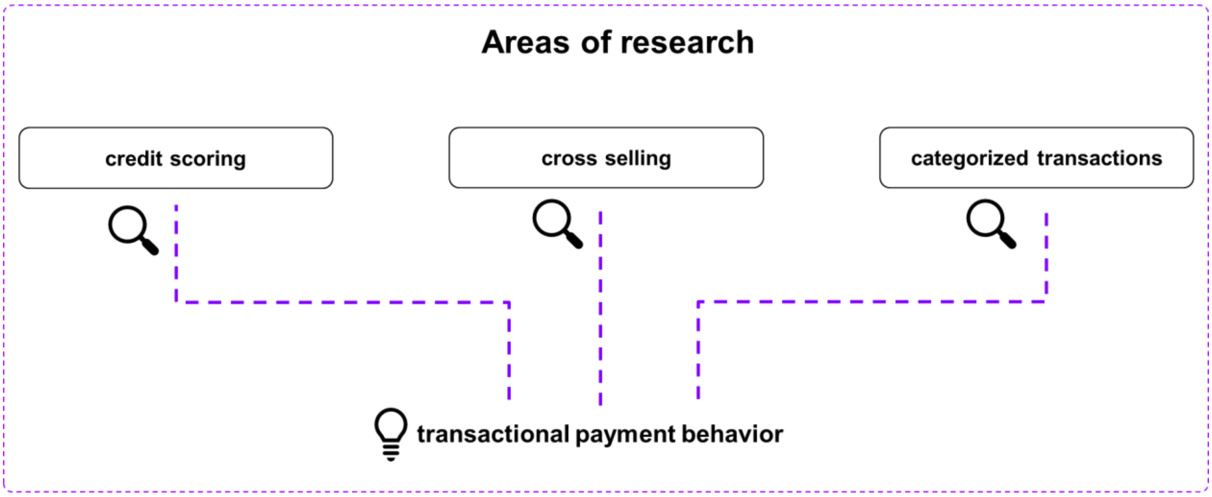


Figure 6-2: Overview of the research outcomes - transactional payment behaviour

The data science process applied in the research followed the steps suggested in the literature and was adjusted based on the CRISP-DM process. A customised data science process intensifies the understanding of the pre-processed input variables and its relationships among each other which results also in a better classification and prediction of transactional payment behaviours in general. The in-depth evaluation of the pre-processed data through a detailed descriptive data analysis has provided more succinct and precise credit scoring and cross-selling models, reduced cost-intensive features in the modelling process, and improves the prediction accuracy in the first two areas of research. As an example, the usage of the machine learning variable selection technique random forest variable importance measures has been greatly improved the model accuracy for the creditworthiness and cross-selling case. The research uses the free accessible dataset from Berka to answer the pre-defined research questions by applying a set of mathematical algorithms. Another research aim was to find insights in the pre-processed datasets and make probably greater business decisions. Both research objectives have been achieved by making sure that the developed research approach could occur new behavioural insights.

Further relevant parts of the research focused on customer behaviour analysis based on a transactional dataset combining a variety of mining methods. Therefore, the study has applied various machine learning algorithms to assess the performance outcomes for the research objectives formulated regarding the first two research projects. All evaluation results have been investigated by selected visual data mining techniques using MS Azure ML and R Studio. One accomplished goal of the developed research approach was to identify a mining algorithm that performs best predictive modelling results. MS Azure ML enables, therefore, the current research to interactively explore the underlying pre-processed dataset and to get a feeling of the contained behavorial insights.

As shown in the previous chapter, the research results clearly shows that the varImp() function based on a Random Forest Model supports the feature selection process by building a cost-sensitive data model. The presented model performance results further prove that integrating a statistical variable analysis into the modelling process can enhance the performance of credit risk and cross-sell candidates' prediction. Moreover, in order to ensure that the assessment of the various machine learning models is objectively, the research has tested the results on different customised datasets in accordance to the pre-defined data science approach. As presented in the summarised results tables in the previous chapter, the proposed cost-sensitive data model is performing significantly better than all other suggested data models. While previous research from van der Putten (1999) has focused on the feasiblity to develop a model to predict a useful score for credit card ownership, the current research results of the second research project demonstrate that the model-building process for promoting a credit card to their banking clients can be optimised from a data science perspective. In this context, the development of a cost-sensitive data model using the underlying data science approach will add new knowledge to research.

The conducted research experiments suggested at least three attributes to help predict credit scores for bank customers more accurately: 1) ‘AvgBalance’, 2) ‘Age’ and 3) ‘AvgIncome’ or 3) ‘AvgExpenses’ for the credit scoring model, and four attributes are suggested to increase the predict accuracy of cross-sell candidates for bank customers: 1) ‘Age’, 2) ‘AvgExpenses’, 3) ‘AvgBalance’, 4) ‘AvgSalary’. Together these attributes had helped to identify 91.8 % of the customers who rated with a bad or good credit score correctly, and 83.8 % who can be identified as cross-sell candidates correctly by using for both cases the Two-Class Decision Forest algorithm. Theses research findings relate to the pre-processed Berka dataset in the specific time period; to verify the generalizability of these research findings more research need to be conducted. The research also shows that preparing the data is by far the most time intensive step as it takes up to 80% of the total project duration. It proved that creating, evaluating, refining and deploying the model blocks around 20% of the time.

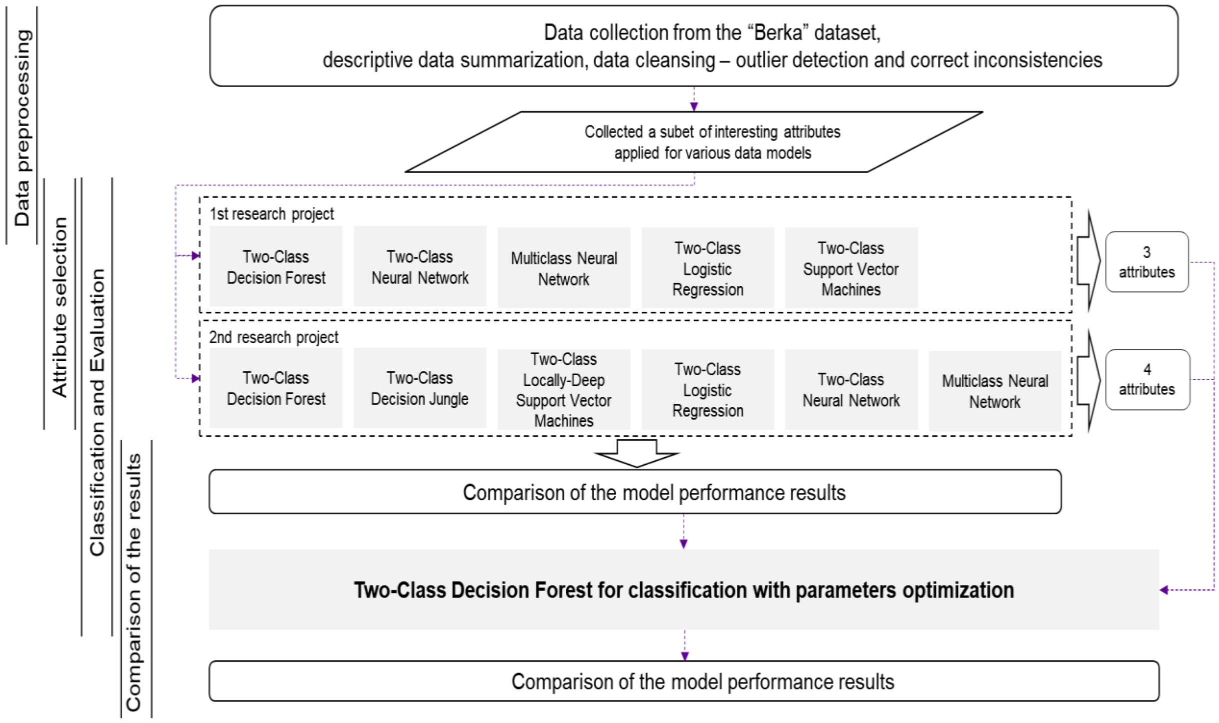

Data preprocessing
Attribute selection
Classification and Evaluation
Comparison of the results
Data collection from the “Berka” dataset,
descriptive data summarization, data cleansing – outlier detection and correct inconsistencies
Collected a subet of interesting attributes
applied for various data models
1st research project
Two-Class Decision Forest
Two-Class Neural Network
Multiclass Neural Network
Two-Class Logistic Regression
Two-Class Support Vector Machines
3 attributes
2nd research project
Two-Class Decision Forest
Two-Class Decision Jungle
Two-Class Locally-Deep Support Vector Machines
Two-Class Logistic Regression
Two-Class Neural Network
Multiclass Neural Network
4 attributes
Comparison of the model performance results
Two-Class Decision Forest for classification with parameters optimization
Comparison of the model performance results

Figure 6-3: The flowchart of the applied data science process

The above flowchart in figure 6-3 summarises the applied data science process in building a cost-sensitive data model and identifying the best-performing machine learning algorithm ‘Two-Class Decision Forest’ for the first two research projects, the credit scoring and cross-selling case. The process comprises: (1) data pre-processing with attribute selection as separate task, (2) classification and evaluation and (3) comparison of the results, as is shown in the figure 6-3 above. For dissemination purposes, it is important to analyse the evaluation results of various supervised learning and classification algorithm instead of constructing and using only one model.

In general, the research results for the last research project demonstrate the feasibility of predicting uncategorised transactions by using association rules and especially a deep-neural network algorithm. In this research thesis, for the first time the market basket analysis using association rules are applied on categorised payment transactions to discover interesting transactional payment pattern for banking clients. The research question addressed by applying a market basket analysis include how to boost the sales of a given banking product, what other banking products do discontinue a product impact, and which bank products should be shelved together. The results showed that initially four interesting association rules can be generated out of the pre-processed Berka dataset. Given to the small number of categorised transactions types the research outcomes are not significant, but the generated results might be interesting for marketing initatives. However, the applied research approach underlines the feasibility of identifying clusters of reciprocal relationships between the selected payment categories (i.e. payment for statement vs. interest credited, etc.). To the best of the gained knowledge, this study is the first time that a graph-based visualisation has been applied to analyse the transactional payment behaviour based on categorised transactions.

Moreover, the research models the transactional payment behaviour using a deep-neural network to predict prospective payment types of their banking clients for the first time. The systematic literature review does not identify any other relevant literature which discusses this special topic. The deep-learning model results demonstrates that the selected input variables (e.g. ‘TransType’, ‘TransMode’, ‘AmountBucket’, ‘BalanceBucket’ and ‘BankPartner’) for the data model results in reasonable predictions of unknown categorised transactional types by a model accuracy rate of +98% learned on up to 300 epochs. Altogether, the combination of supervised and unsupervised learning approaches aims to help banks gaining more transactional behavioural insights about their customers which might also be useful in assessing the creditworthiness of their clients or identify more suitable cross-selling candidates.

In this study, the research shows how transactional data can be transformed into reliable predictive model to score the creditworthiness of bank customers or headcount suitable cross-sell candidates for promoting credit cards to their customer, and how payment information can be extracted to recycle it to more analyzable visual representation. As mentioned in the introduction and in the previous chapter along descriptive and predictive analysis of transactional payment behaviour through payment data streams becomes increasingly important because financial institutions are collecting a massive amount of produced data from their customer. It is hoped that the “deluge of data” (Anderson, 2008) produced by the advanced analytic technologies within the “digital age” (Kitchin, 2014) could hold the key to effective decision-making within every marketing department across the financial services industry.

Finally, big data science offers valuable insights that financial institutions need to succeed in today’s increasingly competitive marketplace. However, many financial institutions do not know where to start and don’t want to have an unending big data science project. The underlying research outcomes are an excellent place to start. I believe that the various research projects enabled via a hybrid research design and methodology powered by a range of different data mining tools like R Studio, Python and MS Azure ML is a suitable choice for financial institutions that want to leverage their existing skills while ramping up their big data science use cases.

6.2. Limitation of the research work

This section summarises the most important research boundaries and further highlights some limitations of the current research, which might have direct impact to achieving the introduced research objectives.

As in any other research studies, this research is also not without its limitations. One of the potential limitations is that the pre-processed dataset was a single data source without having any further reference dataset. Further research studies in this research field of transactional behaviour analysis could include additional datasets to conduct a comparative analysis that clearly highlight the validity of the research results. For instance, the study may scrutinise the research outcomes produced by the descriptive analysis as well as the predictive models.

From the viewpoint of our data science approach the evaluation results by performing various supervised learning algorithms as a whole can be interpreted as sufficiently reliable and novel knowledge on the research universe from which the data form a random sample. However, the generalizability of the research outcomes is limited by the lack of available reference datasets. In conlusion, the presented research results can confirm only the specific sphere of the research projects.

A further limitation is that the data analysis did not consider more categorised payment transaction types due to the limited number of given payment categories within the dataset. As a result of this, the constructed deep-neural network model may produce different outcomes whenever learning the supervised model on other input datasets. Current research combined supervised and unsupervised learning algorithm to discover new transactional behavioural insights from the data by using market basket analysis with a frequency analysis and a deep-learning model along with neural networks which was applied on the entire transactional dataset from Berka. I have proposed this research approach and integrated it into the suggested data science approach without any further concerns. The problem of reengineering and optimising the presented data science approach should be discussed briefly in further studies.

The suggested adoption of the best-performing Two-Class Decision Forest algorithm can bring new ideas into the problematic of predicting credit scores and cross-sell candidates. It should be noted that most of existing research studies in the literature are focusing on neural networks. Hence, the investigation of categorised transactional payments has shown that a market basket analysis including a frequency analysis with R Studio is a suitable approach for identifying relevant and unrevealed associations. Again, further different payment categories in the Berka dataset may result in more plausible association rule generation. Beyond that, the occurred deficiencies by conduction the research can have positive development of the method itself. However, the methodological choices were constrained by the selected mining methods. Other unused mining methods can likely be of interest for the performance comparison and may result in slightly modified research outcomes.

Regarding to the research design and research methodology, following limitations must be mentioned. The data understanding is facing to some degree poor data quality regarding the analysed payment categories. Due to the current available Berka dataset, the research accessed only around seven payment categories. The research data validation process was probably sufficient but there’s no guarantee of completeness.

Concerning the data pre-processing stage, the data cleaning efforts for the accessed and pre-processed data may be insufficient. The modelling stage includes the uncertainty that the research did not cover all relevant variables when applying the appropriate algorithm. There is no further effort made to combine the Berka dataset with other free accessible and valuable datasets, which might result in additional interesting input data. However, the data enhancement and enrichment are beyond the scope of this study and the developed models are always as good as their underlying assumptions and input variables.

The evaluation results can be challenged against another comparable dataset. However, the presented research process design considered the objectivity of the research outcomes for the last research project by integrating several data mining tools such as R Studio and Python and comparing their output results.

Due to the limited computational power of my workstation, the frequent sequences analysis - conducted in the last research project - was limited. The lack of memory storage when analysing frequent sequences was restricting the research results. Probably more interesting transactional behavioural patterns can be discovered. Moreover, forecasting next categorised payment transaction in that context will be a challenging task due to the system performance of researchers' workstation. However, the presented results for that research project have generally strengthening the understanding of transaction payment behaviours within the payment ecosystem.

Previous studies concerning transactional payment behaviour have mainly focused on credit scoring models that ultimaltely based on a maximum of up to three supervised learning algorithms. Having said that, the research results generated out of the first two research projects are novel and important considering the number of applied machine learning algorithm and how the current research has visualised the performance results at scale. It should be noted that the generalizability of the visualised results is limited by the provided MS Azure ML features. The open-source software R Studio probably provide more data visualisation features.

There is no common sense about what kind of input variables we should choose for the prediction model to evaluate the creditworthiness of bank customers. More efforts should be invested in determining the most contributing features whenever credit scores or cross-selling candidates might hold strong interest. However, there are only a handful research papers, for instance published by Wang, Wang and Lai (2005) or Yu, Wang and Lai (2008), which provides a detailed overview of the attribute selection. Current research was limited by the attributes given in the Berka dataset.

Though the presented research experiments for the first two research projects show that the Two-Class Decision Forest algorithm is promising, the study reported here is far from sufficient to generate any conclusive statements about the performance of the Two-Class Decision Forest algorithm in general. Since the positive performances depends on the characteristics and data quality of the pre-processed datasets, further research is encouraged to apply the same research experiments for creditworthiness analysis and cross-selling promotion analysis to different datasets to determine whether the Two-Class Decision Forest algorithm can indeed have superior results as shown in the performance comparisons within the underlying research study.

Comparing overall performance characteristics in both datasets, the credit scoring and cross-selling case, the research clearly shows that their performances vary from another whenever the threshold of the Two-Class Decision Forest algorithm was adjusted. Especially, the Two-Class Decision Forest model achieves best performance in the optimised dataset. Therefore, the varImp() method should be chosen appropriately when the Two-Class Decision Forest method is applied to credit scoring as well as cross-selling cases. If other feature selection techniques will come up with different results, this can also be tested. Furthermore, there are several limitations that may restrict the use of machine learning models due to the over-fitting issue. The developed deep-neural network model for the last research project can be served also as an example.

In summary, this study is limited by diverse issues: a lack of other valuable datasets for comparison, evaluation, and validation of the developed model in all three research projects. There is no evidence given that the created training sets for the constructed research experiments are sufficient. Despite these limitations and research topics to consider in future research studies, the current research broadens the knowledge of transactional payment behaviour regarding the different fields of research, and to establish a successful managerial direction from a data-driven perspective.

6.3. Deployment of future work

This section provides an overview of prospective research works, which might extent the evaluated research projects with their specified research goals. Therefore, the following proposals are left for future works, for instance, using different datasets to validate the developed transactional payment behaviour models; using other data mining techniques as already applied in the current piece of work; using hybrid models, combining different techniques to further improve classification and predictive performance, to check for the existence of the identified behavioural patterns and classifications, especially for the last research project.

Another best-practice example for future work is to develop a customer-oriented API where customers themselves can check their credit scoring as well as cross-selling products. The bank can adapt the predictive models automatically when the dataset changes. To provide the basis for this deployment, the model-building journey should be pushed ahead to enable a smooth implementation into a hetereogenous system landscapes. AI and Analytics didn’t yet deliver to the promise and are now threatened by privacy laws. To succeed, the presented results needs to be embedded in an ecosystem to add new value to their customers. It is well known that transactional data combined with machine learning methods are the brains and backbone of the data-driven enterprise. As suggested in this research, transactional payment data streams could be used for product recommendations. When more payment data comes available, the potential of developing an accurate customer profile will increase and probably the next payment transaction category can be predicted for a certain client more precisely. For instance, a reward system can be built to take these measures into account and foster actual and prospective clients. Future studies should also take into account how disruptive data-driven business models benefit a vertical integration along their entire value chain.

Future research on transactional payment behaviour analysis may address – for example – the development of new algorithms and data science approaches, such as using the GUHA method (Coufal, 1999; Coufal, Holeňa and Sochorová, 1999; Petr, 2003) of automated hypotheses only for the payment categories. Furthermore, other research can investigate both pre-processed dataset for credit scoring and cross-sell candidates with the help of the GUHA method by using Fisher’s t-test. Analysing the statistical significance of the pre-processed target attributes will probably occur additional hidden pattern in both datasets. Having said that, future works may scuritinise alternative feature selection methods to identify and select out further significant attributes out of the transactional dataset and might lead to additional interesting evaluation results.

Regarding the graphical illustration of the research process (figure 4-1) in chapter 4, continuing work can be carried out in the areas of deployment of the predictive models, which could be reflect the integration into a data-driven enterprise architecture model with the goal of harmonising the process-flow of the developed data models.

Realising the entire research design and methodology on a comparable reference dataset to test current research outcomes for further benchmarks might also be a potential research work. Future research projects can be realised by using the introduced research design and methodologies. The focus can be set on the deployment; for instance, how the developed models can be integrated and adapted into everyday business or how the descriptive and predictive results should be used and who will use the transactional data.

The research questions formulated and solved in this research process can be re-used in other domains, especially with interfaces to transactional payment streams. However, in future works, it is possible to add more testing scenarios for the applied supervised learning algorithms by using different subsets of payment transaction data. Moreover, clustering can be attempted by using dedicated payment categories.

Future related work can be focus on executing the sequential pattern algorithm cSPADE for deeper frequent sequences mining. Additional work can be describing client’s experiences in applying frequency and sequence mining in a new transactional payment ecosystem to predict payment transactions before they actually happen and improve the procedures to do so efficiently. The more important aspect is how to take the results of mining the payment categories through various (un-)supervised learning algorithm and use them effectively within the banking industry. The deployment of the applied model across a heterogenous corporate system landscape was not part of the underlying research objectives, although it should be evaluated in terms of data access control and authorisation in the context of restrictive privacy laws.

Future work can develop extended research experiments for the third research project. One of the research objectives can be enhancing the neural network model and identifying the suitable threshold for the number of hidden nodes to increase the accuracy rate of the developed model. Further studies can conduct a review for the model performance by using a different subset of various nodes. The basis of the designed research experiments might also be the self-developed deep-learning model in Python using TensorFlow framework. Alternatively, other frameworks can also be applied. The research focus can be set on varying the number of nodes and observing the capacity of the model by allowing the model to better learn the training dataset. However, the research might be limited to a point by the chosen configuration for the learning algorithm (e.g. learning rate, batch size, and epochs).

Some further ideas to extend the research that other data scientist work may explore is, increasing the number of nodes (e.g. current research is configured on 40 nodes as first layer and 20 nodes as second layer) to find the point where the learning algorithm is no longer capable of learning uncategorised payment transaction types. Doing so, the research can also counter the stochastic nature of the machine learning algorithm. Other more complex work can be implemented when increasing the hidden layers on the problem that requires the increased capacity provided by increased depth to perform well whenever a deep model is trained.

The present review showed that data mining techniques can play an important role in analysing transactional payment behaviour by helping to develop machine learning models to predict bank customers future creditworthiness or cross-sell opportunities. Based on the above discussion, different classification algorithms could be used to classify customers credit scores according to their attributes. Most of the research reviewed focused on applying the favoured neural networks. The current study shows that the Two-Class Decision Forest algorithm is also a good fit which performed well for both purposes. Most of the reviewed papers addressed the topic of credit scoring, although in the majority of their studies the researchers use modified neural networks instead of the two-class decision forest algorithm. The focus was mainly on predicting creditworthiness of customers based on certain historical transactional attributes, whereas few papers considered demographic information and other payment attributes to test model accuracy. Further research is necessary to establish a feature selection process which ensures a sustainable impact on the model in case of validity, reliability, accuracy and objectivity. The review’s outcomes might lead to increased studies that focus only on the pre-processing stage in the data modelling process.

Future work can focus more on the feature selection process by performing other techniques like genetic algorithm, forward selection, information gain, gain ratio, gini index and correlation. It may be possible to achieve the same or better accuracy using a different set of features as I have suggested in the optimal cost-sensitive data models. Other studies can address the point of using a mix of machine learning variable selection techniques and comparing their results with the recently applied random forest variable importance measurement. For instance, attribute selection process can use a genetic algorithm with neural networks, forward selection, information gain and GINI index, gain ratio or correlation.

Finally, all of the findings of these proposed research reviews might help financial institutions and researchers use data mining techniques and tools to develop everyday more suitable data models to predict future transactional payment trends and/or customers at risks (i.e. clients with irregular payment transaction streams). In addition, the findings could also help financial institutions to build an early-warning system to overcome unforeseen payment behaviours and to form a knowledge-based recommender system for increasing cross-selling activities with respect to an unerring banking product marketing. This research may open the door for future comprehensive studies that apply a data mining approach to analyse more hidden transactional pattern to know the customer behaviours best. Future work can address this area by applying further data mining techniques to transactional datasets from other financial institutions to develop a predictive model, as well as ultimately identifying some additional attributes that influence the payment behaviour. Another important issue is the need for further studies that look at real-time transactional data to predict future trends more accurately.

References

Abdou, H., Pointon, J. and El-Masry, A. (2008) ‘Neural nets versus conventional techniques in credit scoring in Egyptian banking’, *Expert Systems with Applications*, 35(3), pp. 1275–1292. doi: 10.1016/j.eswa.2007.08.030.

Abdullah, A., Veltkamp, R. C. and Wiering, M. A. (2009) ‘An ensemble of deep support vector machines for image categorization’, *SoCPaR 2009 - Soft Computing and Pattern Recognition*, (January), pp. 301–306. doi: 10.1109/SoCPaR.2009.67.

Abu El-Atta, A. H., Moussa, M. I. and Hassanien, A. E. (2018) ‘Corrigendum to “Two-class support vector machine with new kernel function based on paths of features for predicting chemical activity” [Information Sciences 403–404 (2017) 42–54](S0020025517306448)(10.1016/j.ins.2017.04.003))’, *Information Sciences*. Elsevier Inc., 467, p. 618. doi: 10.1016/j.ins.2017.06.010.

Aggarwal, C. C. (2013) *Outlier analysis*, *Outlier Analysis*. doi: 10.1007/978-1-4614-6396-2.

Agrawal, R. . *et al.* (1996) ‘Fast Discovery of Association Rules, in Advances in Knowledge Discovery and Data Mining’, pp. 307–328.

Agrawal, R., Imielinski, T. and Swami, A. (1993) ‘Proceedings of the 1993 ACM SIGMOD Conference’, *Mining Association Rules between Sets of tems in Large Database*, (May), pp. 1–10. Available at: papers2://publication/uuid/091CB418-5F2E-4ED2-8B0B-CC1138668A17.

Agrawal, R., Imieliński, T. and Swami, A. (1993) ‘Mining Association Rules Between Sets of Items in Large Databases’, *ACM SIGMOD Record*, 22(2), pp. 207–216. doi: 10.1145/170036.170072.

Anand, S. S.; Patrick, A. R.; Hughes, J. G.; Bell, D. A. (1998) ‘A data mining methodology for cross-sales’, *Knowledge-Based Systems*, 10(7), pp. 449–461. doi: 10.1016/S0950-7051(98)00035-5.

Anastasiu, D. C., Iverson, J., Smith, S. and Karypis, G. (2014) ‘Big Data Frequent Pattern Mining’, *Frequent Pattern Mining*, 9783319078, pp. 225–259. doi: 10.1007/978-3-319-07821-2_10.

Anastasiu, D. C., Iverson, J., Smith, S., Karypis, G., *et al.* (2014) ‘Mining Association

Rules Between Sets of Items in Large Databases', *ACM SIGMOD Record*, 14(2), pp. 207–216. doi: 10.18637/jss.v014.i15.

Anderson, C. (2008) 'The end of theory: The data deluge makes the scientific method obsolete', *Wired Magazine*. doi: 10.1016/j.ecolmodel.2009.09.008.

Arreola, K. Z., Fehr, J. and Burkhardt, H. (2007) 'Albert-Ludwigs-Universität Freiburg - Informatik Institut für Fast Support Vector Machine Classification of very large Datasets Fast Support Vector Machine Classification of very large Datasets'.

Arvind, T. S. and Badhe, V. (2016) 'Discovery of Certain Association Rules from an Uncertain Database', in *Proceedings - 2015 International Conference on Computational Intelligence and Communication Networks, CICN 2015*, pp. 827–831. doi: 10.1109/CICN.2015.168.

Bao, T. and Chang, T. L. S. (2014) 'Why Amazon uses both the New York Times Best Seller List and customer reviews: An empirical study of multiplier effects on product sales from multiple earned media', *Decision Support Systems*. The Authors, 67, pp. 1–8. doi: 10.1016/j.dss.2014.07.004.

Bekhet, H. A. and Eletter, S. F. K. (2014) 'Credit risk assessment model for Jordanian commercial banks: Neural scoring approach', *Review of Development Finance*. University of Cairo., 4(1), pp. 20–28. doi: 10.1016/j.rdf.2014.03.002.

Benos, A. and Papanastasopoulos, G. (2007) 'Extending the Merton Model: A hybrid approach to assessing credit quality', *Mathematical and Computer Modelling*, 46(1–2), pp. 47–68. doi: 10.1016/j.mcm.2006.12.012.

Berka, P. (2006) 'NIAID Glossary of Funding and Policy Terms and Acronyms'. Available at: http://www.niaid.nih.gov/ncn/glossary/default6.htm#racial.

Berka, P. and Rauch, J. (2007) 'Lessons Learned From The ECML/PKDD Discovery Challenge on the Atherosclerosis Risk Factors Data', *Computing and Informatics*, 26(December 2006), pp. 329–344.

Berrado, A., Elfahli, S. and El Garah, W. (2013) 'Using data mining techniques to investigate the factors influencing mobile payment adoption in Morocco', *2013 8th International Conference on Intelligent Systems: Theories and Applications, SITA 2013*, pp. 1–5. doi: 10.1109/SITA.2013.6560791.

Bijak, K. and Thomas, L. C. (2012) 'Does segmentation always improve model performance in credit scoring?', *Expert Systems with Applications*. Elsevier Ltd, 39(3), pp. 2433–2442. doi: 10.1016/j.eswa.2011.08.093.

Black, G. S. (2005) 'Predictors of consumer trust: likelihood to pay online', *Marketing Intelligence & Planning*, 23(7), pp. 648–658. doi: 10.1108/MIP-01-2016-0006.

Blockeel, H. and Uwents, W. (2004) 'Using neural networks for relational learning', *Working Notes of the ICML-2004 Workshop on Statistical Relational Learning and Connections to Other Fields (SRL-2004)*, pp. 23–28.

Boyer, K. K. and Hult, G. T. M. (2006) 'Customer behavioural intentions for online purchases: An examination of fulfillment method and customer experience level', *Journal of Operations Management*, 24(2), pp. 124–147. doi: 10.1016/j.jom.2005.04.002.

Brause, R., Langsdorf, T. and Hepp, M. (1999) 'Credit card fraud detection by adaptive neural data mining', *JW GoetheUniversity Comp Sc Dep Report*, 7, p. 99.

Brereton, P. *et al.* (2007) 'Lessons from applying the systematic literature review process within the software engineering domain', *Journal of Systems and Software*. Elsevier Inc., 80(4), pp. 571–583. doi: 10.1016/j.jss.2006.07.009.

Butler, M. and Butler, R. (2015) 'Investigating the possibility to use differentiated authentication based on risk profiling to secure online banking', *Information and Computer Security*, 23(4), pp. 421–434. doi: 10.1108/ICS-11-2014-0074.

Chapman, P. *et al.* (2000) 'CRISP-DM 1.0 Step-by-step', *ASHA presentation*, p. 73. doi: 10.1109/ICETET.2008.239.

Chen, Y. L. *et al.* (2009) 'Discovering recency, frequency, and monetary (RFM) sequential patterns from customers' purchasing data', *Electronic Commerce Research and Applications*. Elsevier B.V., 8(5), pp. 241–251. doi: 10.1016/j.elerap.2009.03.002.

Chye, K. H., Chin, T. W. and Peng, G. C. (2004) 'Credit scoring using data mining techniques', *Singapore Management Review*, 26(2), p. 25.

Cortes, C. *et al.* (2016) 'Hancock: A Language for Analyzing Transactional Data Streams', 26(2), pp. 387–408. doi: 10.1007/978-3-540-28608-0_19.

Coufal, D. (1999) 'Financial data set analysis - hierarchical testing with GUHA method', *3rd European Conference on Principles and Practice of Knowledge Discovery in Databases*, pp. 1–11.

Coufal, D., Holena, M. and Sochorová, A. (1999) ‘Coping With Discovery Challenge By Guha’, *European Conference on Principles and Practice of Knowledge Discovery in Databases*, pp. 1–6.

Cristianini, N. and Shawe-Taylor, J. (2000) *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. doi: 10.1017/cbo9780511801389.

Fahrmeir, L. *et al.* (2013) *Regression: Models, methods and applications*, *Regression: Models, Methods and Applications*. doi: 10.1007/978-3-642-34333-9.

Femina, B. T. and Sudheep, E. M. (2015) ‘An efficient CRM-data mining framework for the prediction of customer behaviour’, *Procedia Computer Science*. Elsevier Masson SAS, 46(Icict 2014), pp. 725–731. doi: 10.1016/j.procs.2015.02.136.

Fournier-Viger, P. *et al.* (2012) ‘CMRules: Mining Sequential Rules’, 25, pp. 63–76.

Goodfellow, I., Bengio, Y. and Courville, A. (2015) ‘Deep Learning Book’, *Deep Learning*. doi: 10.1016/B978-0-12-391420-0.09987-X.

Gschwind, M. (2007) ‘Predicting Late Payments: A Study in Tenant Behaviour Using Data Mining Techniques’, *Journal of Real Estate Portfolio Management*, 13(3), pp. 269–288.

Haeusler, J. (2016) ‘Follow the money: Using payment behaviour as predictor for future self-exclusion’, *International Gambling Studies*, 16(2), pp. 246–262. doi: 10.1080/14459795.2016.1158306.

Hahsler, M. *et al.* (2011) ‘The arules R-Package Ecosystem: Analyzing Interesting Patterns from Large Transaction Data Sets’, *Journal of Machine Learning Research.*

Hahsler, M. and Chelluboina, S. (2011) ‘Visualizing Association Rules: Introduction to the R-extension Package arulesViz’, *R project module*, pp. 1–24. Available at: http://www.comp.nus.edu.sg/~zhanghao/project/visualization/[2010]arulesViz.pdf.

Hahsler, M., Grün, B. and Hornik, K. (2005) ‘Arules - A computational environment for mining association rules and frequent item sets’, *Journal of Statistical Software*, 14. doi: 10.18637/jss.v014.i15.

Hájek, P., Holeňa, M. and Rauch, J. (2010) ‘The GUHA method and its meaning for data mining’, *Journal of Computer and System Sciences*, 76(1), pp. 34–48. doi:

10.1016/j.jcss.2009.05.004.

Han, J. and Kamber, M. (2006) *Data mining: Concepts and Techniques, Book*, *Morgan Kaufmann Publishers, Elsevier*. doi: 10.1002/1521-3773(20010316)40:6<9823::AID-ANIE9823>3.3.CO;2-C.

Hand, D. J.; Henley, W. E. (1998) 'Statistical Classification Methods in Consumer Credit Scoring: A Review', *Journal of the Royal Statistical Society. Series A (Statistics in Society)*, 160(3), pp. 523–541.

Hastie, T; Tibshirani, R; Friedman, J. (2017) 'The Elements of Statistical Learning Second Edition', *Math. Intell.* doi: 111.

Higgins, J. P. and Green, S. (2011) 'Cochrane Handbook for Systematic Reviews of Interventions Version 5.1.0 [updated March 2011]', *The Cochrane Collaboration*. doi: 10.1002/9780470712184.ch4.

Hong, J. K. and Lee, Y. Il (2014) 'Bancassurance in East Asia: Cultural impact on customers' cross-buying behaviour', *Journal of Financial Services Marketing*. Nature Publishing Group, 19(3), pp. 234–247. doi: 10.1057/fsm.2014.16.

Hooman, A. *et al.* (2016) 'Statistical and data mining methods in credit scoring', *The Journal of Developing Areas*, 50(5), pp. 371–381. doi: 10.1353/jda.2016.0057.

Hotho, Andreas; Maedche, A. (1999) 'Efficient Discovery of Client Profiles from a Financial Database', *3rd European Conference on Principles and Practice of Knowledge Discovery in Databases*, pp. 1–7.

Hsieh, N. C. (2004) 'An integrated data mining and behavioural scoring model for analyzing bank customers', *Expert Systems with Applications*, 27(4), pp. 623–633. doi: 10.1016/j.eswa.2004.06.007.

Huang, C. L., Chen, M. C. and Wang, C. J. (2007) 'Credit scoring with a data mining approach based on support vector machines', *Expert Systems with Applications*, 33(4), pp. 847–856. doi: 10.1016/j.eswa.2006.07.007.

Huang, S. *et al.* (2018) 'Applications of support vector machine (SVM) learning in cancer genomics', *Cancer Genomics and Proteomics*, 15(1), pp. 41–51. doi: 10.21873/cgp.20063.

Hyndman, L. R. J. (2014) 'Forecasting : Principles & Practice', (September).

Islam, R. H. and Ahsan (2015) 'A data mining approach to predict prospective business sectors for lending in retail banking using decision tree', *International Journal of Data Mining & Knowledge Management Process*, 5(2), pp. 13–22.

Johnson, I. (2018) 'arulesCBA : Classification for Factor and Transactional Data Sets Using Association Rules', *Southern Methodist University,Dallas, Texas*, (MSc Dissertation).

Jose, C. *et al.* (2013) 'Local deep kernel learning for efficient non-linear SVM prediction', *30th International Conference on Machine Learning, ICML 2013*, 28(PART 2), pp. 1523–1531.

Jurafsky, D. and Martin, J. H. (2019) 'Speech and Language Processing'.

Karadag, H. and Akman, V. (2015) 'The Role Of Cross Selling In SME Banking: An Analysis From Turkey', *EMAJ: Emerging Markets Journal*, 5(1), pp. 82–92. doi: 10.5195/emaj.2015.71.

Kauffman, R. J. and Ma, D. (2015) 'Special issue: Contemporary research on payments and cards in the global fintech revolution', *Electronic Commerce Research and Applications*, 14(5), pp. 261–264. doi: 10.1016/j.elerap.2015.09.005.

Kitchin, R. (2014) 'Big Data, new epistemologies and paradigm shifts', *Big Data & Society*, 1(1), p. 205395171452848. doi: 10.1177/2053951714528481.

Kovalerchuk, B. and Vityaev, E. (2010) 'Data Mining for Financial Applications', *Data Mining and Knowledge Discovery Handbook*, pp. 1203–1224. doi: 10.1007/0-387-25465-X_57.

Kumar, S., Bhattacharyya, B. and Gupta, V. K. (2014) 'Present and Future Energy Scenario in India', *Journal of The Institution of Engineers (India): Series B*, 95(3), pp. 247–254. doi: 10.1007/s40031-014-0099-7.

Kvamme, H. *et al.* (2018) 'Predicting mortgage default using convolutional neural networks', *Expert Systems with Applications*, 102, pp. 207–217. doi: 10.1016/j.eswa.2018.02.029.

Lecun, Y., Bengio, Y. and Hinton, G. (2015) 'Deep learning', *Nature*, 521(7553), pp. 436–444. doi: 10.1038/nature14539.

Lee, Y. W. . *et al.* (2002) 'AIMQ: A methodology for information quality assessment', *Information and Management*. doi: 10.1016/S0378-7206(02)00043-5.

Levin, Boris; Meidan, Abraham; Cheskis, Alex; Gefen, Ohad; Vorobyov, I. (1999)

‘PKDD99 Discovery Challenge - Financial Domain’, *European Conference on Principles and Practice of Knowledge Discovery in Databases*, pp. 3–6.

Li, S. H. *et al.* (2012) ‘Identifying the signs of fraudulent accounts using data mining techniques’, *Computers in Human Behavior*. Elsevier Ltd, 28(3), pp. 1002–1013. doi: 10.1016/j.chb.2012.01.002.

Li, W. and Liao, J. (2011) ‘An empirical study on credit scoring model for credit card by using data mining technology’, in *Proceedings - 2011 7th International Conference on Computational Intelligence and Security, CIS 2011*, pp. 1279–1282. doi: 10.1109/CIS.2011.283.

Liu, C. H. and Cai, S. Q. (2008) ‘Customer cross-selling model based on counter propagation network’, *Direct Marketing: An International Journal*, 2(1), pp. 36–47. doi: 10.1108/17505930810863626.

Malekpour, M., Khademi, M. and Minae-bidgoli, B. (2016) ‘A hybrid Data mining method for Intrusion and Fraud Detection in E-Banking Systems A Hybrid Data Mining Method for Intrusion and Fraud Detection in E-Banking Systems’, (February). doi: 10.1166/jcies.2014.1068.

Miksovsky, P.; Zelezny, F.; Stepankova, O.; Pechoucek, M. (1999) ‘Financial Data Challenge’, *European Conference on Principles and Practice of Knowledge Discovery in Databases*, p. 38632.

Miksovsky, P., Matousek, K. and Kouba, Z. (2003) ‘Data pre-processing support for data mining’, *IEEE SMC*, pp. 1–4. doi: 10.1109/icsmc.2002.1176327.

Mohan, Lija; M., S. E. (2016) ‘A Novel Big Data Approach to Classify Bank Customers - Solution by Combining PIG, R and Hadoop’, *International Journal of Information Technology and Computer Science*, 8(9), pp. 81–90. doi: 10.5815/ijitcs.2016.09.10.

Nami, S. and Shajari, M. (2018) ‘Cost-sensitive payment card fraud detection based on dynamic random forest and k-nearest neighbors’, *Expert Systems with Applications*. Elsevier Ltd, 110, pp. 381–392. doi: 10.1016/j.eswa.2018.06.011.

Olafsson, S., Li, X. and Wu, S. (2019) ‘Operations research and data mining’, (2006). doi: 10.1016/j.ejor.2006.09.023.

Oreski, S., Oreski, D. and Oreski, G. (2012) ‘Hybrid system with genetic algorithm and artificial neural networks and its application to retail credit risk assessment’, *Expert*

Systems with Applications. Elsevier Ltd, 39(16), pp. 12605–12617. doi: 10.1016/j.eswa.2012.05.023.

Ouardighi, A. E. L., Akadi, A. E. L. and Aboutajdine, D. (2007) 'Feature Selection on Supervised Classification Using Wilk's Lambda Statistic', *3rd International Symposium on Computational Intelligence and Intelligent Informatics*, pp. 51–55.

Padillo, F., Luna, J. M. and Ventura, S. (2016) 'Subgroup discovery on big data: Pruning the search space on exhaustive search algorithms', *Proceedings - 2016 IEEE International Conference on Big Data, Big Data 2016*, pp. 1814–1823. doi: 10.1109/BigData.2016.7840799.

Pate-Cornell, M. E., Tagaras, G. and Eisenhardt, A. N. D. K. M. (1990) 'Dynamic Optimization of Cash Flow Management', 37(August), pp. 203–212.

Pavlov, Y. L. (2019) 'Random forests', *Random Forests*, pp. 1–122. doi: 10.1201/9780429469275-8.

Petr, H. (2003) 'Briefly on the GUHA method of data mining', *Journal of Telecommunications and Information Technology*, 3, pp. 1–3.

Pijls, W. (1999) 'Discovery Challenge, Financial Data', *European Conference on Principles and Practice of Knowledge Discovery in Databases*, pp. 1–4.

van der Putten, P. (1999) 'Promoting Credit Card Usage by Mining Transaction Data', *European Conference on Principles and Practice of Knowledge Discovery in Databases*, pp. 1–5.

Qiu, D., Wang, Y. and Bi, B. (2011) 'Identify Cross-Selling Opportunities via Hybrid Classifier', *International Journal of Data Warehousing and Mining*, 4(2), pp. 55–62. doi: 10.4018/jdwm.2008040107.

Salleb, A. (2000) 'An Application of Association Rules Discovery to GIS', in *Principles of Data Mining and Knowledge Discovery, 4th European Conference, PKDD 2000*, pp. 1–6.

Schetinin, V. *et al.* (2003) 'Learning multi-class neural-network models from electroencephalograms', *Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science)*, 2773 PART(Lm), pp. 155–162. doi: 10.1007/978-3-540-45224-9_23.

Schmidhuber, J. (2015) 'Deep Learning in neural networks: An overview', *Neural*

Networks, 61, pp. 85–117. doi: 10.1016/j.neunet.2014.09.003.

Schutte, J., Van Der Merwe, A. and Reyneke, F. (2017) ‘Using Data Analytics and Data Mining Methods To Determine a High Net Worth Individual’s Electronic Banking Behavior’, *Journal of Internet Banking and Commerce*, 22(3), pp. 1–39. Available at: https://search.proquest.com/docview/1992208819?accountid=10218%0Ahttps://www.ub.uni-koeln.de/openurl?url_ver=Z39.88-2004&rft_val_fmt=info:ofi/fmt:kev:mtx:journal&genre=article&sid=ProQ:ProQ%3Aabiglobal&atitle=USING+DATA+ANALYTICS+AND+DATA+MINING+METHODS+T.

Shih, C. C. *et al.* (2011) ‘Discovering cardholders’ payment-patterns based on clustering analysis’, *Expert Systems with Applications*. Elsevier Ltd, 38(10), pp. 13284–13290. doi: 10.1016/j.eswa.2011.04.148.

Shotton, J., Nowozin, S., *et al.* (2013) ‘Decision jungles: Compact and rich models for classification’, *Advances in Neural Information Processing Systems*, (January).

Shotton, J., Girshick, R., *et al.* (2013) ‘Efficient human pose estimation from single depth images’, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(12), pp. 2821–2840. doi: 10.1109/TPAMI.2012.241.

Sołdacki, P. and Protaziuk, G. (2013) ‘Discovering Interesting Rules from Financial Data’, in *Intelligent Information Systems 2002*, pp. 109–119. doi: 10.1007/978-3-7908-1777-5_11.

Spenke, M. and Beilken, C. (1999) ‘Visual interactive data mining with InfoZoom - the financial data set’, *European Conference on Principles and Practice of Knowledge Discovery in Databases*. Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.135.1288&rep=rep1&type=pdf.

Steinhaeuser, K., Chawla, N. V and Ganguly, A. R. (2015) ‘Improving Inference of Gaussian Mixtures using Auxiliary Variables’, *Statistical Analysis and Data Mining*, 8(5), pp. 497–511. doi: 10.1002/sam.

Suzuki, E. (2000) ‘Mining Financial Data with Scheduled Discovery of Exception Rules’, *European Conference on Principles and Practice of Knowledge Discovery in Databases*, pp. 1–8.

Swain, H. and Mohapatra, S. (2013) ‘A Comparative Study of Leadership Factors Affecting Public and Private Sector Banks in India: An Employee Perspective’, *Prabandhan: Indian Journal of Management*, 6(9), p. 28. doi: 10.17010//2013/v6i9/60028.

Taghva, M. R., Hosseini Bamakan, S. M. and Toufani, S. (2011) 'A data mining method for service marketing: A case study of banking industry', *Management Science Letters*, 1(3), pp. 253–262. doi: 10.5267/j.msl.2010.04.004.

Till, R. J. and Hand, D. J. (2003) 'Behavioural models of credit card usage', *Journal of Applied Statistics*, 30(10), pp. 1201–1220. doi: 10.1080/0266476032000107196.

Trubik, E. and Smith, M. (2000) 'Developing a model of customer defection in the Australian banking industry', *Managerial Auditing Journal*, 15(5), pp. 199–208. doi: 10.1108/02686900010339300.

Tsai, C.-Y. (2007) 'A credit card usage behaviour analysis framework - A data mining approach', in *ICE-B 2007: Proceedings of the Second International Conference on e-Business*, pp. 219–226.

Tsai, C. Y. (2008) 'A hybrid data mining approach for credit card usage behavior analysis', in *Communications in Computer and Information Science*, pp. 85–97. doi: 10.1007/978-3-540-88653-2-6.

Tsai, C. Y., Wang, J. C. and Chen, C. J. (2007) 'Mining usage behavior change for credit card users', *WSEAS Transactions on Information Science and Applications*, 4(3), pp. 529–536.

Tseng, F. C. (2013) 'Mining frequent itemsets in large databases: The hierarchical partitioning approach', *Expert Systems with Applications*. Elsevier Ltd, 40(5), pp. 1654–1661. doi: 10.1016/j.eswa.2012.09.005.

Vaghela, V. B.; Kalpesh H, V.; Nilesh K, M. (2014) 'Information Theory Based Feature Selection for Multi-Relational Naïve Bayesian Classifier', *Journal of Data Mining in Genomics & Proteomics*, 05(02), pp. 2–7. doi: 10.4172/2153-0602.1000155.

Vanneschi, L. *et al.* (2018) 'An artificial intelligence system for predicting customer default in e-commerce', *Expert Systems with Applications*, 104, pp. 1–21. doi: 10.1016/j.eswa.2018.03.025.

Vojtek, M. and Kocenda, E. (2006) 'Credit Scoring Methods', *Czech Journal of Economics and Finance*, 56(3–4), pp. 152–167. doi: 10.1109/TPAMI.2012.125.

Wang, G. and Ma, J. (2011) 'Study of corporate credit risk prediction based on integrating boosting and random subspace', *Expert Systems With Applications*. Elsevier Ltd, 38(11), pp. 13871–13878. doi: 10.1016/j.eswa.2011.04.191.

Wang, Yongqiao, Wang, S. and Lai, K. K. (2005) 'A new fuzzy support vector machine to evaluate credit risk', *IEEE Transactions on Fuzzy Systems*, 13(6), pp. 820–831. doi: 10.1109/TFUZZ.2005.859320.

Wang, Yongqiao;, Wang, S. and Lai, K. K. (2005) 'A new fuzzy support vector machine to evaluate credit risk', *IEEE Transactions on Fuzzy Systems*, 13(6), pp. 820–831. doi: 10.1109/TFUZZ.2005.859320.

Wang, Yongqiao, Wang, S. and Lai, K K (2005) 'A New Fuzzy Support Vector Machine to Evaluate Credit Risk', 13(6), pp. 820–831.

Weber, I. (1998) 'Discovery of interesting rules and subgroups in a financial database', *World Scientific*, 3, pp. 1–15.

Williams, N. (2014) 'Conclusions from a NAÏVE Bayes Operator Predicting the Medicare 2011 Transaction Data Set', pp. 561–565.

Xiong, T. *et al.* (2013) 'Personal bankruptcy prediction by mining credit card data', *Expert Systems with Applications*, 40(2), pp. 665–676. doi: 10.1016/j.eswa.2012.07.072.

Yan, L. *et al.* (2011) 'Group RFM analysis as a novel framework to discover better customer consumption behavior', *Expert Systems with Applications*. Elsevier Ltd, 38(10), pp. 13057–13063. doi: 10.1108/IJBM-05-2013-0048.

Yap, B. W., Ong, S. H. and Husain, N. H. M. (2011) 'Using data mining to improve assessment of credit worthiness via credit scoring models', *Expert Systems with Applications*. Elsevier Ltd, 38(10), pp. 13274–13283. doi: 10.1016/j.eswa.2011.04.147.

Yeh, I. C. and Lien, C. hui (2009) 'The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients', *Expert Systems with Applications*. Elsevier Ltd, 36(2 PART 1), pp. 2473–2480. doi: 10.1016/j.eswa.2007.12.020.

Yen, S.-J. and Chen, A. L. P. (2001) 'A Graph-Based Approach for Discovering Various Types of Association Rules', *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING*, 13(5), pp. 839–845. doi: 10.1109/TPAMI.2012.125.

Yu, L., Wang, S. and Lai, K. K. (2008a) 'Credit risk assessment with a multistage neural network ensemble learning approach', *Expert Systems with Applications*, 34(2), pp. 1434–1444. doi: 10.1016/j.eswa.2007.01.009.

Yu, L., Wang, S. and Lai, K. K. (2008b) 'Credit risk assessment with a multistage neural network ensemble learning approach', 34, pp. 1434–1444. doi: 10.1016/j.eswa.2007.01.009.

Zahir Azami, S. B., Torabi, N. and Tanabian, M. (2004) 'Modeling the customer behavior in the mobile payment on a non-connected vending machine platform', *Canadian Conference on Electrical and Computer Engineering 2004 (IEEE Cat. No.04CH37513)*, pp. 815–818. doi: 10.1109/CCECE.2004.1345239.

Zaki, M. J. ; Parthasarathy, S. ; Ogihara, M.; Li, W. (1997) 'New Algorithms for Fast Discovery of Association', pp. 283–286.

Zaki, M. J. (2001) 'An Efficient Algorithm for Mining Frequent Sequences.pdf', pp. 31–60.

Zareapoor, M. and Seeja, K. R. (2013) 'FraudMiner: A Novel Credit Card Fraud Detection Model Based on Frequent Itemset Mining', *The Scientific World Journal*, 3, pp. 342–347. doi: 10.1126/science.279.5356.1431b.

Zemke, S. (2000) 'On Developing a Financial Prediction System : Pitfalls and Possibilities'.

Zhang, D. *et al.* (2010) 'Vertical bagging decision trees model for credit scoring', *Expert Systems with Applications*. Elsevier Ltd, 37(12), pp. 7838–7843. doi: 10.1016/j.eswa.2010.04.054.

Zhao, D. (2012) 'Introduction', *SpringerBriefs in Computer Science*, (9781461432838), pp. 1–15. doi: 10.1007/978-1-4614-3284-5_1.

Appendices

Appendix A: Tabulated results of the systematic literature review study

Author / year	Mining steps	Method	Tool	Outcome	Rank
(Levin, Meidan, Cheskis, Gefen and Vorobyov, 1999)	Description	Association rules, ranking objects	WizWhy (own)	Predict yes or no for loans.	Low
(Spenke and Beilken, 1999)	Visualization, interactive exploring	Display correlations	InfoZoom (own)	Correlations	Low
(Weber, 1998)	Classification, description, interpretation	Discover interesting rules by statistical criterion "implication intensity" and subgroups		Classify good or bad loans.	Medium
(Miksovsky, Zelezny, Stepankova, Pechoucek, 1999)	Clustering, classfication	Applied the C5.0 algorithm	SQL (Delphi), ILP (PROLOG), MS Excel, ID3-based tool (C5.0)	Cluster interesting regions, classify (un-)successful loan.	Low
(van der Putten, 1999)	Classification, preprocessing, k-nearest neighbor (NN)	Descriptive profile analysis, univariate deviation detection,		Predict scores for credit card ownership.	High

		predictive models			
(Pijls, 1999)	Description, preprocessing	Classification, frequency, correlation, regression	Unix, MS Word-97	Released a new algorithm for mining frequent item sets.	Medium
(Coufal, Holena and Sochorová, 1999)	Classification using Fishers' quantifier, preprocessing	GUHA method		Classifying good or bad clients by 50 constructed hypotheses.	Low

Table A-1: Overview of the PKDD99 Discovery Challenge results

Author / year	Mining steps	Method	Tool	Outcome	Rank
(Coufal, 1999)	Design decision trees, Hypothesis testing with FIMPL quantifier	Hierarchical testing with GUHA		A set of hypotheses supporting / not supporting the fact of good or bad loan payment.	Medium
(Vaghela, Kalpesh H and Nilesh K, 2014)	Multi-classification accuracy, multi-relation data mining (MRDM)	InfoDist, Pearson's correlation, Multi-relational Naïve Bayesian Classifier		Improve classification accuracy by MRDM, good time performance, improve comprehensibility.	Low
(Hotho and Maedche, 1999)	Principal component analysis, intensional rules.	Kohonens Self-Organizing Maps based on neural networks, Decision-tree algorithm C5.0	IBM's Intelligent Miner for data, PIVOTER	Derived intensional description of the client clusters.	Low

(Mohan and M., 2016)	Classification	K-mean clustering algorithm	PIG, R and Hadoop	Classify customers on previous transactions.	High
(Suzuki, 2000)	Exception rule	Exception		Threshold scheduling in discovering interesting rules.	Low
(Blockeel and Uwents, 2004)	Classification on relational data mining.	Neural network		Present a neural networks-based approach to relational learning.	Low

Table A-2: Overview of the PKDD00 Discovery Challenge results

Author / year	Mining steps	Method	Tool	Outcome	Rank
(Xiong, Wang, Mayers and Monga, 2013)	Ordinal sequence mining techniques applied on credit card data, clustering categorical sequences	Sequence pattern extraction, support vector machine classifier, model-based k-means algorithm for binary sequences		Showed that sequence patterns are strongly predictive for personal bankruptcy; implemented a personal bankruptcy prediction system running on credit card data.	Medium
(Li and Liao, 2011)	Principle component analysis to extract comprehensible variables.	C5. 0 decision tree, neural network, chi-squared automatic interaction detector, stepwise logistic		The evaluation results for modelling the credit card behavioural usage show that the decision tree method has the	Medium

		model, classification and regression tree	best classification performance in terms of accuracy and sensitivity.	
(Brause, Langsdorf and Hepp, 1999)	Adaptive classification, association rules	Neural network	Detect and predict fraudulent transaction in credit card transactions based on GZS database.	Low
(Tsai, 2007, 2008)	Grouping customers into segments, constructing behavioural rules.	Neural network, fuzzy decision tree	Proposed an integrated data mining approach for credit card usage behavioural analysis by accessing the data from a commercial bank in Taiwan.	Low
(Malekpour, Khademi and Minae-bidgoli, 2016)	Clustering and classification of fraudulent transactions, principle component analysis (PCA) for data dimension reduction.	k-means algorithm, logistic regression, decision tree, neural networks, boosting and bagging algorithm, silhouette index	Predicted fraudulent attacks in e-Banking systems using the dataset from the PKDD Cup 99.	Medium
(Sołdacki and Protaziuk, 2013)	Mining association rules and frequent itemset.	Association rules, frequent itemset	Discovered interesting rules from financial data and presents interestingness measures given in the literature.	Medium

(Arvind and Badhe, 2016)	Mining association rules and frequent itemset.	Association rules employed on a Vague Set Theory, frequent itemset		Generated interesting association rules from a transactional database provided by a retail store.	Medium
(Wang, Wang and Lai, 2005)	t-test, classification	Fuzzy support vector machine, support vector machine, linear regression, logistic regression, artificial neural network		Developed a new fuzzy support vector machine to evaluate credit risks.	Medium
(Yu, Wang and Lai, 2008a)	Individual classification models, two hybrid classification model.	Neural network, logit regression, artificial neural network, support vector machine, neuro-fuzzy System, fuzzy SVM, bagging sampling approach, decorrelation maximization algorithm, logistic transformation, noise injection, cross-validation, t-test		Proposed a multistage neural network ensemble learning model consisting of 6 stages to evaluate credit risks at a measurement level.	Low
(Miksovsky, Matousek and Kouba, 2003)	Data pre-processing support for data mining.		Sumatra Transformation Tool	Introduced the Sumatra Transformation Tool to support	Low

				complex data pre-processing tasks for data mining and demonstrates its capabilities on a real case.	
(Agrawal, Imielinski and Swami, 1993)	Mining association rules between sets of items.	Association rules, frequent itemset, pruning function optimization		Presented an efficient algorithm which generates all significant association rules between items given in a retail dataset.	Medium
(Salleb, 2000)	Mining association rules between geographic layers.	Association rules		Proposed an algorithm to mine hierarchical multi-valued attributes in GIS for rule discovery.	Low
(Agrawal, Mannila, Srikant, Toivonen and Verkamo, 1996)	Discover association rules in large transactions.	Association rules, Apriori, AprioriTid and AprioriHybrid		Introduced and evaluated the performance of two data mining algorithm by mining association rules in large transactions.	Medium
(Padillo, Luna and Ventura, 2016)	Discover subgroups on Big Data	Supervised local pattern mining method, AprioriK-SD-OE, PFP-SD-OE	Apache Spark, MapReduce	Proposed two new efficient exhaustive search algorithms to discover subgroups on Big Data, relying on the MapReduce framework and the Spark open-	Low

				source implementation.	
(Zaki, Parthasarathy, Ogihara and Li, 1997)	Fast discovery of association rules, clustering of related transactions	Association rules, frequent itemset analysis, clustering techniques, efficient lattice traversal techniques		Presented new data mining algorithms for fast association mining in large transactional databases.	Medium
(Wang, Wang and Lai, 2005)	Classification, data mining	Support vector machine		Suggested a new fuzzy support vector machine to evaluate credit risk in order to categorize new applicants or existing customers as good or bad.	Medium
(Benos and Papanastasopoulos, 2007)	Hybrid classification	Probit regression method, econometric method, risk neutral distance		Developed a hybrid model for assessing the credit quality of firms.	Low
(Ouardighi, Akadi and Aboutajdine, 2007)	Feature selection, discriminant analysis	Wilk's Lambda Statistic		Evaluated the performance of the Wilk's Lambda method based on various real datasets.	Low

Table A-3: Overview of the related documents to the PKDD99/00 Discovery Challenge

Author / year	Mining steps	Method	Tool	Outcome	Rank
(Williams, 2014)	Classification, data mining	AI method Naïve Bayes, cross validation algorithm	Rapid Miner Open Source Data Mining Suite	Evaluated the relationship between reported and computed transaction values given in a Medicare dataset in order to develop predictive model.	Medium
(Berrado, Elfahli and El Garah, 2013)	Empirical analysis, data mining, multidimensional correlation analysis	Random forest, association rule		Evaluated the adoption of mobile payment system based on the technology acceptance model to assess the four main drivers ease of use, usefulness, risk perception and transaction fees.	Low
(Trubik and Smith, 2000)	Classification, Logistic regression analysis	Logistic regressions		Developed a predictive model to classify defecting customers and indicate customer leaving patterns.	Medium
(Oreski, Oreski and Oreski, 2012)	Classification, feature selection, forward selection, Information gain, Gain ratio, Gini	Neural network, Genetic algorithm	Rapid Miner	Proposed a genetic algorithm with neural networks technique and its application for credit scoring in retail.	High

	index and Correlation.				
(Yap, Ong and Husain, 2011)	Classification, data mining	Logistic regression, Decision tree, Predictive modeling	SAS Enterprise Miner	Developed a credit scoring model using different data mining algorithm to predict late payments of a recreational club.	Medium
(Abdou, Pointon and El-Masry, 2008)	Classification, data mining, descriptive analysis, Fisher’s test, Kruskal–Wallis test.	Probabilistic neural nets, multi-layer feed-forward nets, discriminant analysis, probit analysis, logistic regression	STATGRAPHICS Plus 5.1, SPSS 14.00 and Neural Tools software	Performance comparison of different data mining algorithm for credit scoring.	Medium
(Gschwind, 2007)	Classification, data mining	Logistic regression (logit model), decision tress, artificial neural networks	SAS Enterprise Miner	Predicting late payments to monitor tenant transactional payment behaviour	Low
(Huang, Chen and Wang, 2007)	Classification, data mining	Credit scoring, support vector machine, genetic programming, neural networks, decision tree		Proposed a new credit scoring model based on SVM and evaluate its performance against other data mining techniques.	Low
(Vojtek and Kocenda, 2006)	Classification, data mining, data visualization	Decision tree, neural network, logistic regression model	Clementine tool	Provide a brief overview of predictive modelling of credit scores including	Low

				benefits, its applications and their limitations.	
(Bekhet and Eletter, 2014)	Classification, data mining	Artificial neural network; credit scoring; logistic regression, radial basis function		Evaluated two proposed quantitative models for a credit risk assessment.	Low
(Bijak and Thomas, 2012)	Classification, data mining	Segmentation, logistic regression, classification and regression trees, chi-squared automatic interaction detection trees, logistic trees with unbiased selection		Evaluated various data mining techniques in according to segmentation to validate the performance of the developed credit scoring models.	Low
(Zhang, Zhou, Leung and Zheng, 2010)	Classification	Decision trees, bagging		Proposed a novel vertical bagging decision trees model for credit scoring which improves the classification accuracy.	Low
(Liu and Cai, 2008)	Classification, data mining	Counter propagation network based on neural network		Developed a customer cross-selling model to identify further business potentials through direct marketing.	Low

(Schutte, Van Der Merwe and Reyneke, 2017)	Customer segmenation			Developed a new segmentation model using data mining techniques to analyze mobile banking user behaviour based on transaction history, recency, frequency, monetary background.	Low
(Zareapoor and Seeja, 2013)	Pattern recognition, frequent itemset mining	Apriori, support vector machine, k-nearest neighbor classifier, naïve Bayes classifier, random forest		Constructed a credit card fraud detection model to detect fraudulent behaviour in credit card transactions.	Low
(Nami and Shajari, 2018)	Pattern recognition	Dynamic random forest, k -nearest neighbors		Proposed a cost-sensitive payment card fraud detection model to identify suspicious transactions.	Low
(Shih, Chiang, Hu and Chen, 2011)	Cluster analysis	Kohonen Feature Map (SOM),	Intelligent Miner	Examined transactional payments of credit card customers to detect behavioural patterns.	Low
(Black, 2005)	Descriptive statistics, hypothesis testing	Regression analysis		Assessment of predictors to calculate the likelihood for future online payments.	Low

(Schutte, Van Der Merwe and Reyneke, 2017)	Classification, regression, clustering, segmentation	Multinomial logit model, CHAID analysis		Determined digital banking behaviour through a mixed transactional dataset consisting of login behaviour, online access, payment and demographical data.	Low
(Hsieh, 2004)	Classification, segmentation, clustering	SOM neural network, Apriori association rule		Proposed a behavioural scoring model based on account and transaction data to assess bank customer for target marketing initiatives.	Medium
(Chen, Kuo, Wu and Tang, 2009)	Data mining, segmentation	Sequential patterns, constraint-based mining		Suggested a novel RFM sequential pattern mining algorithm to analyze purchase data from a retailer.	Low
(Yen and Chen, 2001)	Data mining, association pattern	Association rule, primitive, generalized and multiple-level association rules		Developed a graph-based approach to mine large transactions from a retailer database.	Medium
(Kvamme, Sellereite, Aas and Sjursen, 2018)	Machine learning, deep-learning	Convolutional neural network, random forest		Implemented a Mortgage default predictive model based on raw	Low

	account transactional data.

Table A-4: Overview of the transactional payment behavioural research results

Appendix B: Results based on the search strategy for the systematic literature review

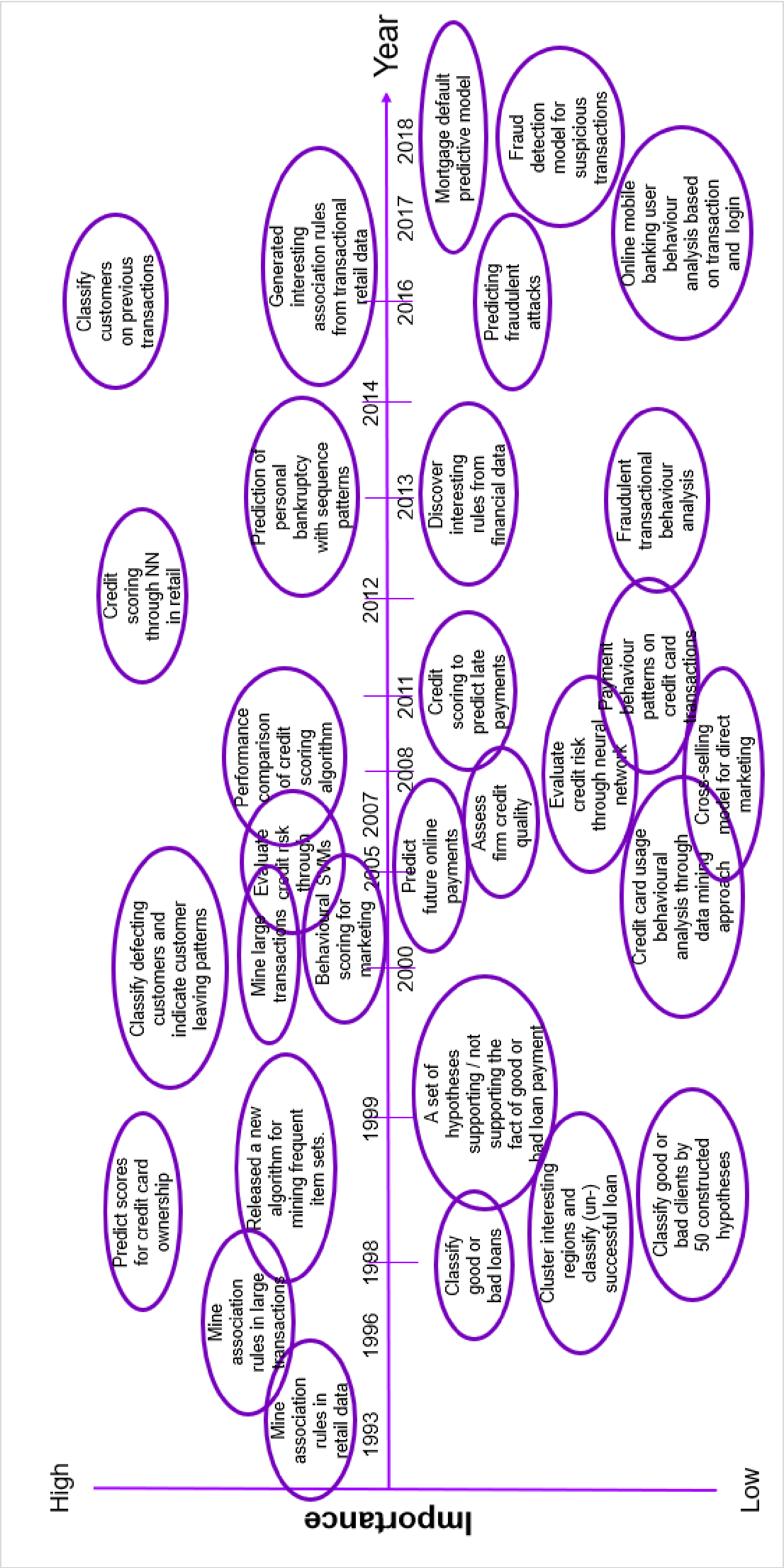
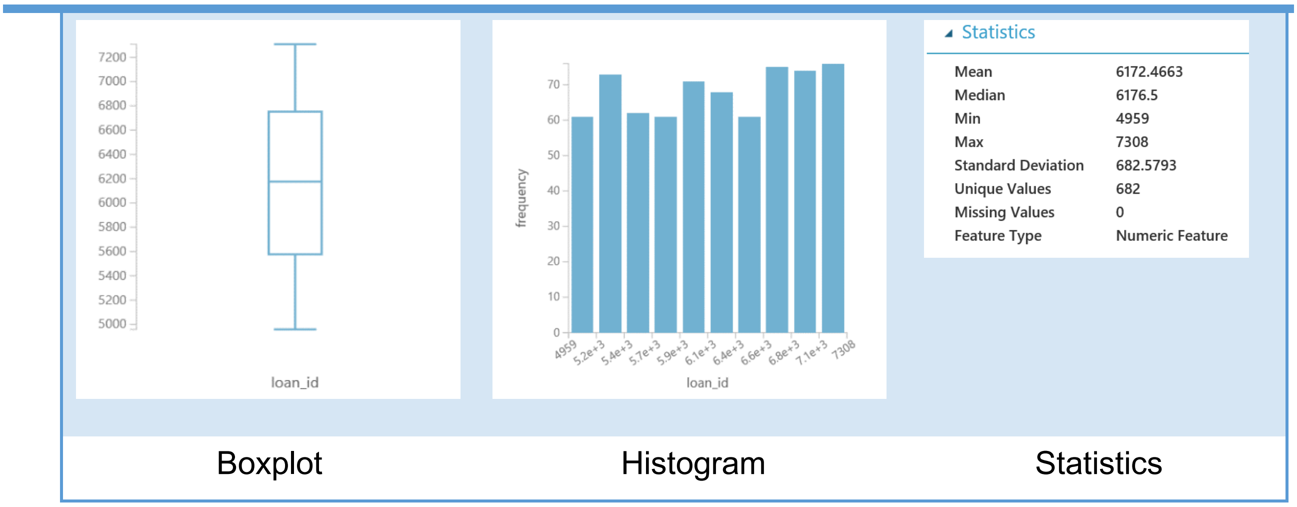

Year
High
Low
Importance
1993
1996
1998
1999
2000
2005
2007
2008
2011
2012
2013
2014
2016
2017
2018
Classify customers on previous transactions
Credit scoring through NN in retail
Classify defecting customers and indicate customer leaving patterns
Predict scores for credit card ownership
Mine association rules in large transactions
Mine association rules in retail data
Released a new algorithm for mining frequent item sets.
Mine large transactions
Evaluate credit risk through SVMs
Behavioural scoring for marketing
Performance comparison of credit scoring algorithm
Prediction of personal bankruptcy with sequence patterns
Generated interesting association rules from transactional retail data
Mortgage default predictive model
Fraud detection model for suspicious transactions
Online mobile banking user behaviour analysis based on transaction and login
Predicting fraudulent attacks
Discover interesting rules from financial data
Fraudulent transactional behaviour analysis
Credit scoring to predict late payments
Payment behaviour patterns on credit card transactions
Evaluate credit risk through neural network
Cross-selling model for direct marketing
Assess firm credit quality
Predict future online payments
Credit card usage behavioural analysis through data mining approach
A set of hypotheses supporting / not supporting the fact of good or bad loan payment
Classify good or bad loans
Cluster interesting regions and classify (un-) successful loan
Classify good or bad clients by 50 constructed hypotheses

Figure B-1: Timeline for transactional payment behaviour topics addressed in the domain of financial services

Appendix C: Descriptive analysis results - visualized results

C.1 Exploratory data analysis for the 1st research project - credit scoring (display_res_azure.csv)

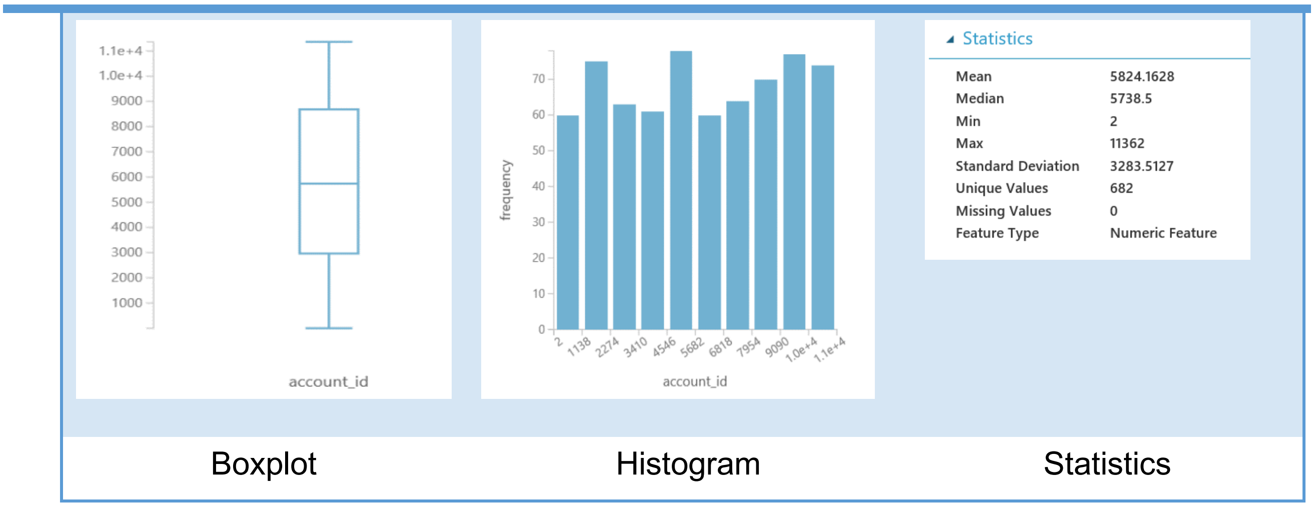
#1 **loan_id**



Mean	6172.4663
Median	6176.5
Min	4959
Max	7308
Standard Deviation	682.5793
Unique Values	682
Missing Values	0
Feature Type	Numeric Feature

Figure C-1: Exploratory data analysis of the attribute - loan_id

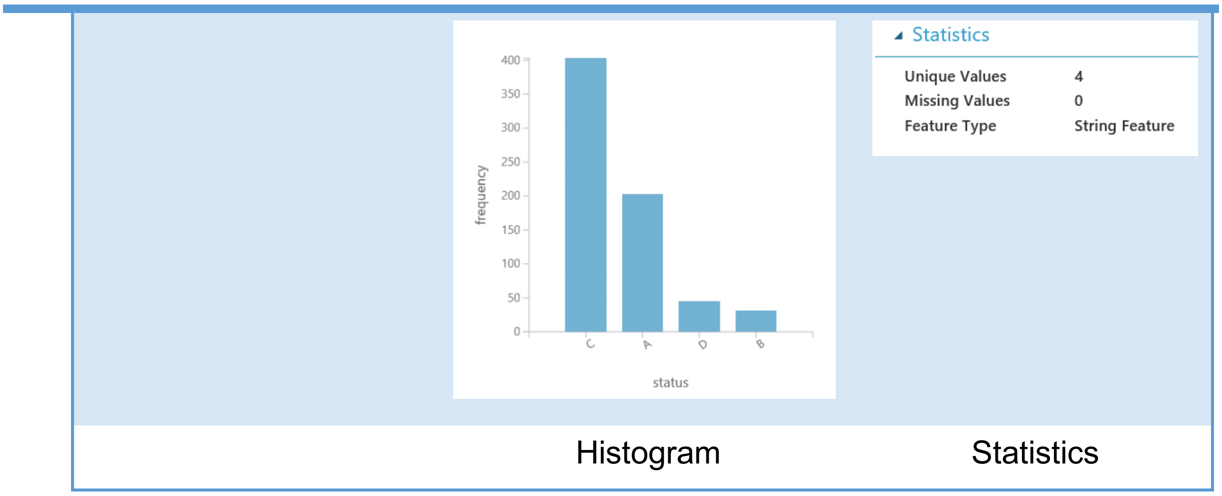
#2 **account_id**



Mean	5824.1628
Median	5738.5
Min	2
Max	11362
Standard Deviation	3283.5127
Unique Values	682
Missing Values	0
Feature Type	Numeric Feature

Figure C-2: Exploratory data analysis of the attribute - account_id

#3 **status**

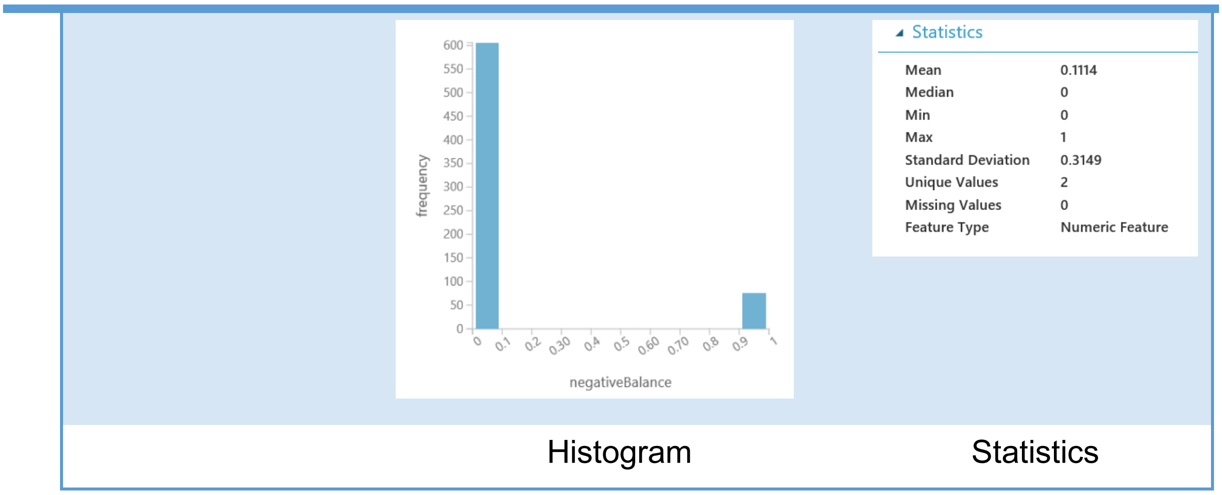


Statistics

Unique Values	4
Missing Values	0
Feature Type	String Feature

Figure C-3: Exploratory data analysis of the attribute - status

#4 **negative balance**

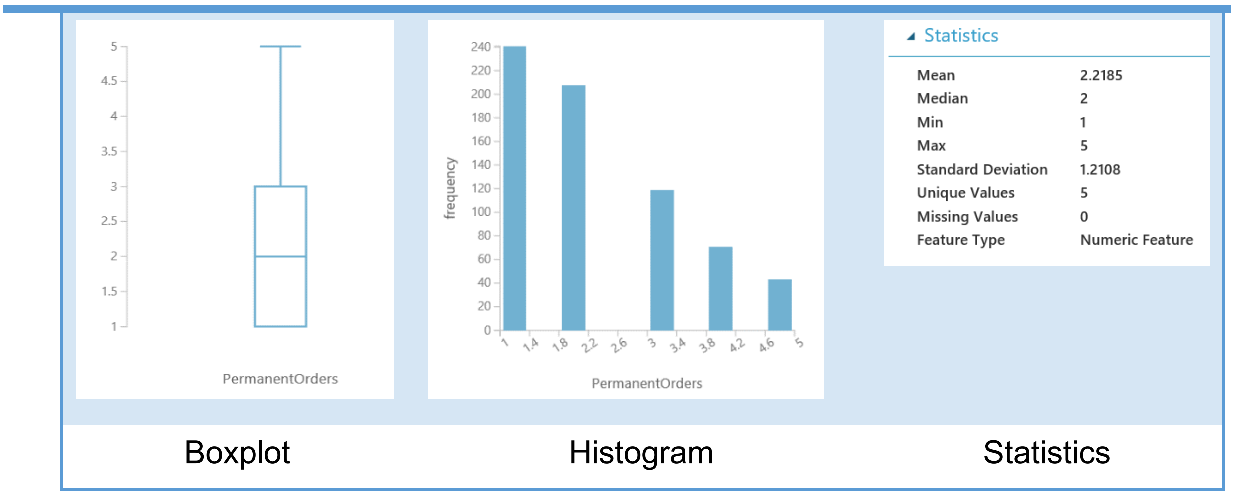


Statistics

Mean	0.1114
Median	0
Min	0
Max	1
Standard Deviation	0.3149
Unique Values	2
Missing Values	0
Feature Type	Numeric Feature

Figure C-4: Exploratory data analysis of the attribute - negative balance

#5 **permanent orders**

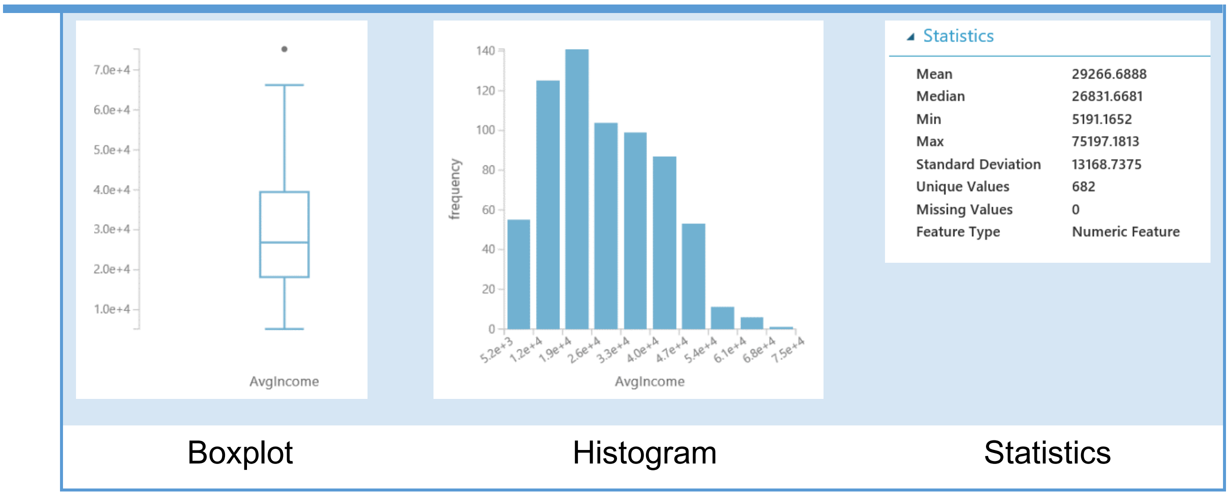


Statistics

Mean	2.2185
Median	2
Min	1
Max	5
Standard Deviation	1.2108
Unique Values	5
Missing Values	0
Feature Type	Numeric Feature

Figure C-5: Exploratory data analysis of the attribute - permanent orders

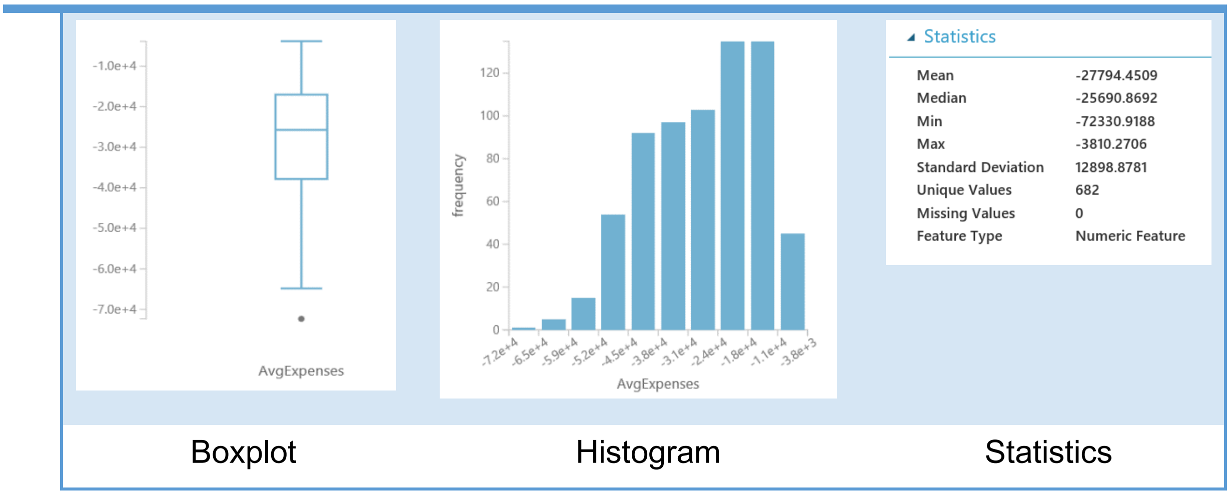
#6 **average income**



Mean	29266.6888
Median	26831.6681
Min	5191.1652
Max	75197.1813
Standard Deviation	13168.7375
Unique Values	682
Missing Values	0
Feature Type	Numeric Feature

Figure C-6: Exploratory data analysis of the attribute - average income

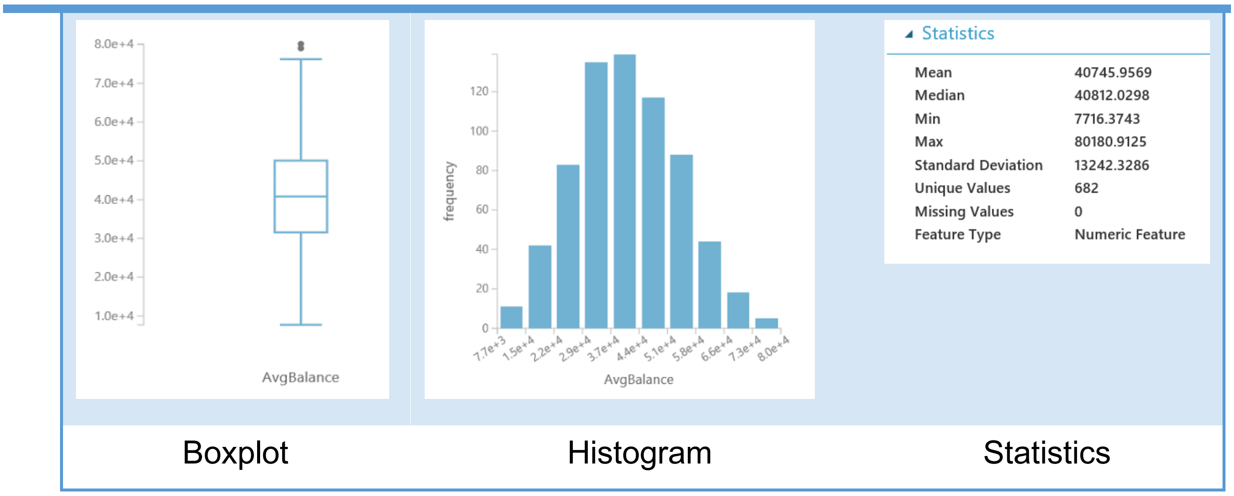
#7 **average expenses**



Mean	-27794.4509
Median	-25690.8692
Min	-72330.9188
Max	-3810.2706
Standard Deviation	12898.8781
Unique Values	682
Missing Values	0
Feature Type	Numeric Feature

Figure C-7: Exploratory data analysis of the attribute - average expenses

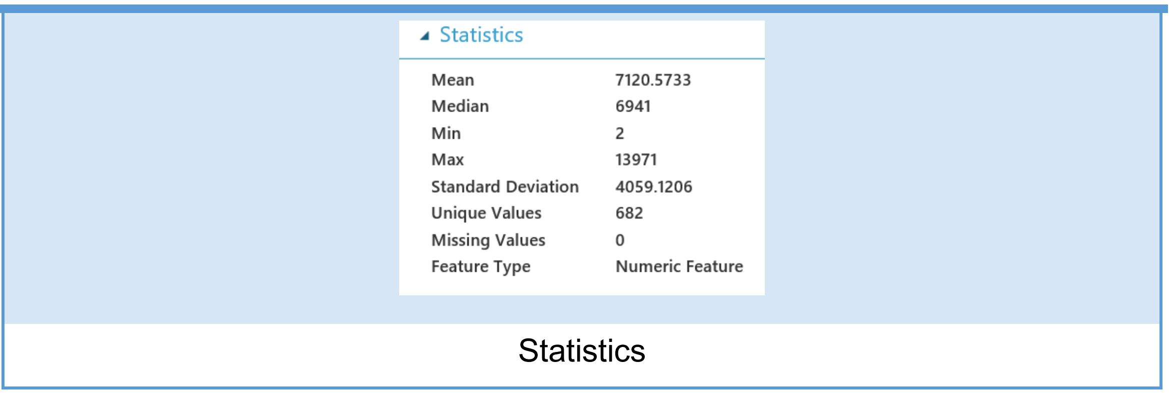
#8 **average balance**



Mean	40745.9569
Median	40812.0298
Min	7716.3743
Max	80180.9125
Standard Deviation	13242.3286
Unique Values	682
Missing Values	0
Feature Type	Numeric Feature

Figure C-8: Exploratory data analysis of the attribute - average balance

#9 **client_id**



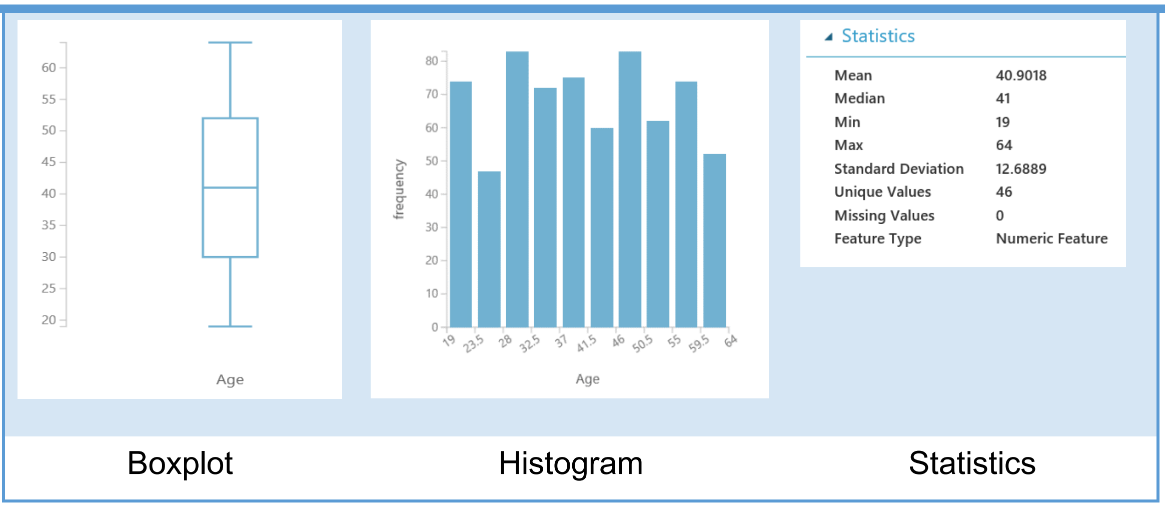
Statistics

Mean	7120.5733
Median	6941
Min	2
Max	13971
Standard Deviation	4059.1206
Unique Values	682
Missing Values	0
Feature Type	Numeric Feature

Statistics

Figure C-9: Exploratory data analysis of the attribute - client_id

#10 **Age**



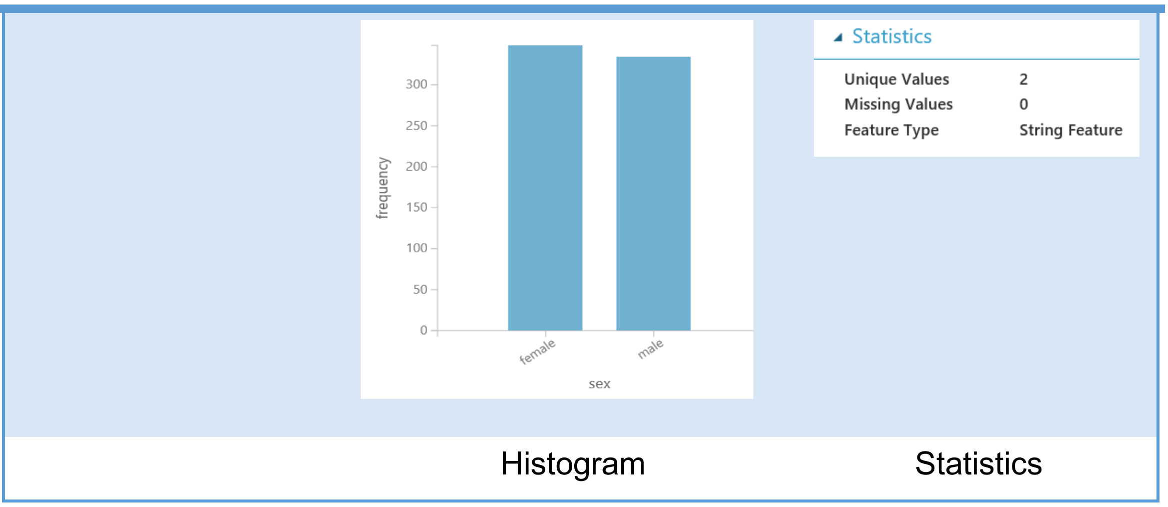
Statistics

Mean	40.9018
Median	41
Min	19
Max	64
Standard Deviation	12.6889
Unique Values	46
Missing Values	0
Feature Type	Numeric Feature

Boxplot Histogram Statistics

Figure C-10: Exploratory data analysis of the attribute - age

#11 **Sex**



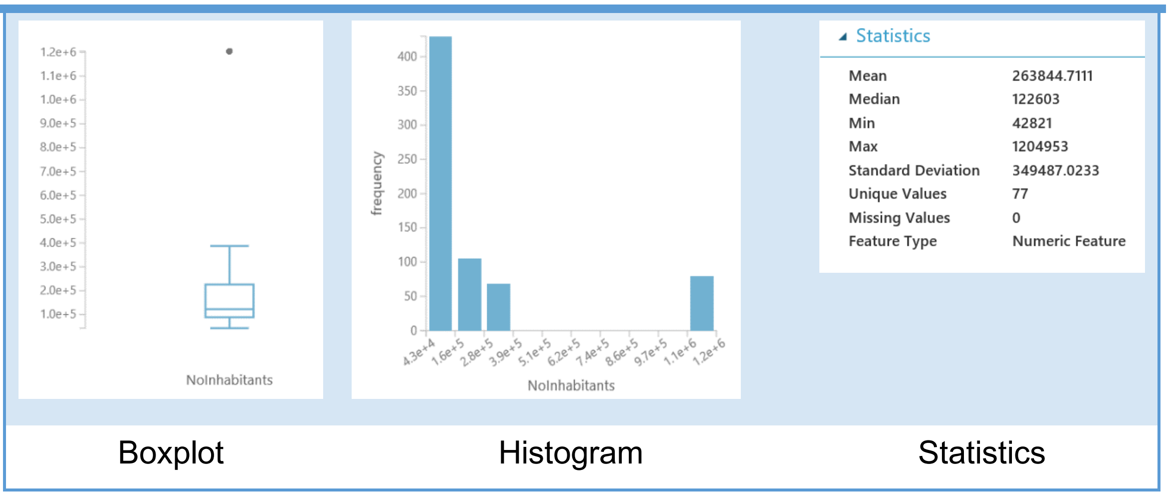
Statistics

Unique Values	2
Missing Values	0
Feature Type	String Feature

Histogram Statistics

Figure C-11: Exploratory data analysis of the attribute - sex

#12 **no inhabitant**

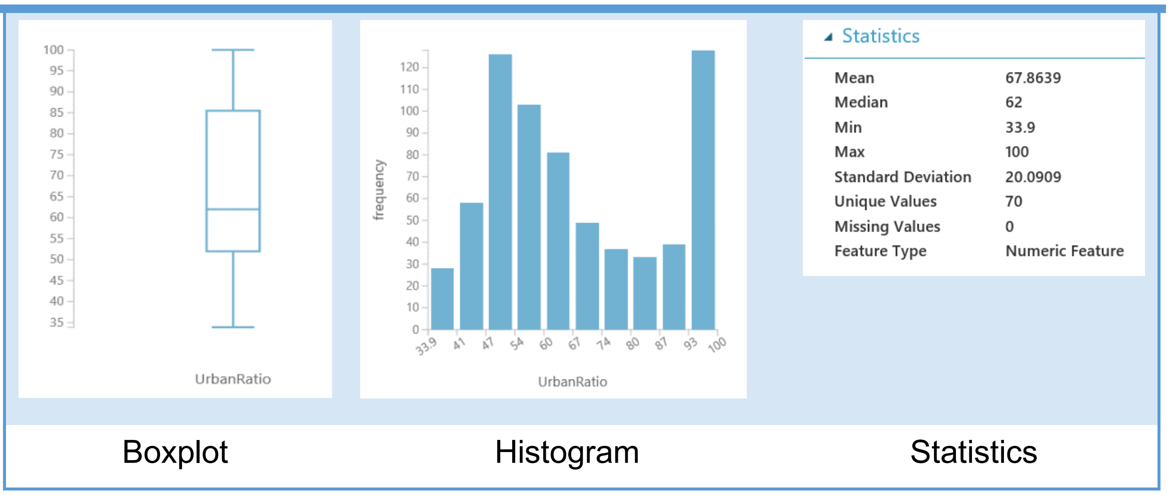


Statistics

Mean	263844.7111
Median	122603
Min	42821
Max	1204953
Standard Deviation	349487.0233
Unique Values	77
Missing Values	0
Feature Type	Numeric Feature

Figure C-12: Exploratory data analysis of the attribute - no inhabitant

#13 **urban ratio**

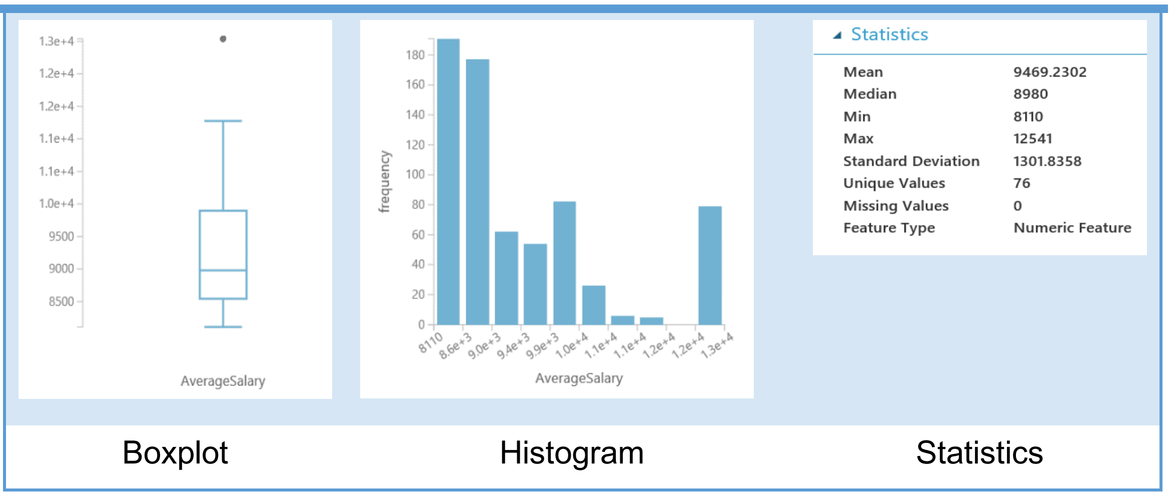


Statistics

Mean	67.8639
Median	62
Min	33.9
Max	100
Standard Deviation	20.0909
Unique Values	70
Missing Values	0
Feature Type	Numeric Feature

Figure C-13: Exploratory data analysis of the attribute - urban ratio

#14 **average salary**

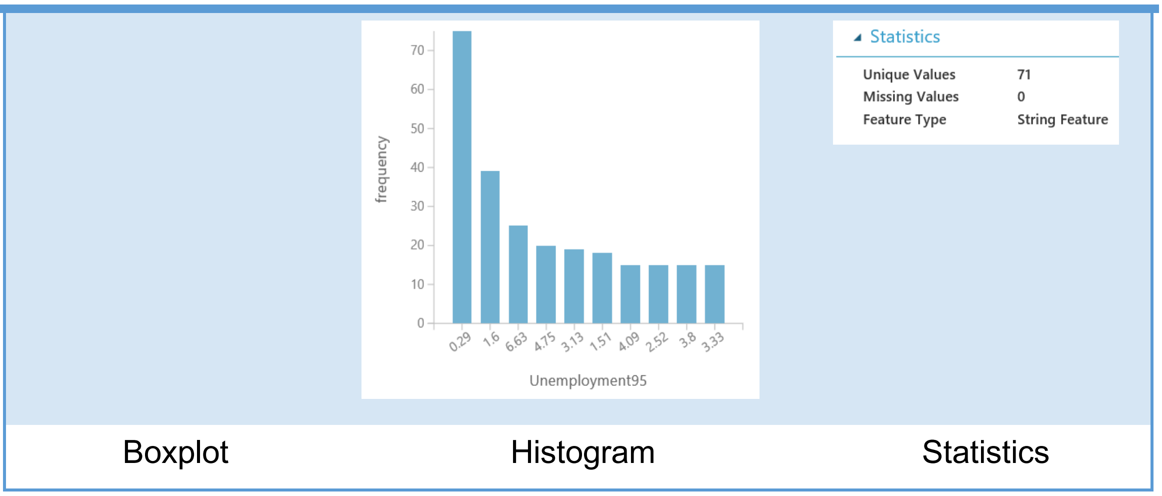


Statistics

Mean	9469.2302
Median	8980
Min	8110
Max	12541
Standard Deviation	1301.8358
Unique Values	76
Missing Values	0
Feature Type	Numeric Feature

Figure C-14: Exploratory data analysis of the attribute - average salary

#15 **unemployment95**

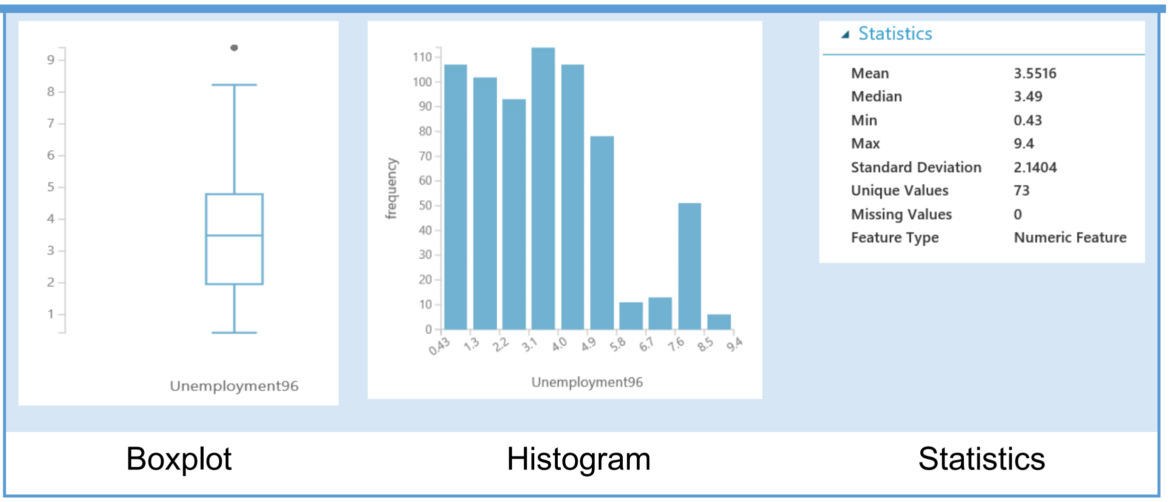


◢ Statistics

Unique Values	71
Missing Values	0
Feature Type	String Feature

Figure C-15: Exploratory data analysis of the attribute - unemployment95

#16 **unemployment96**

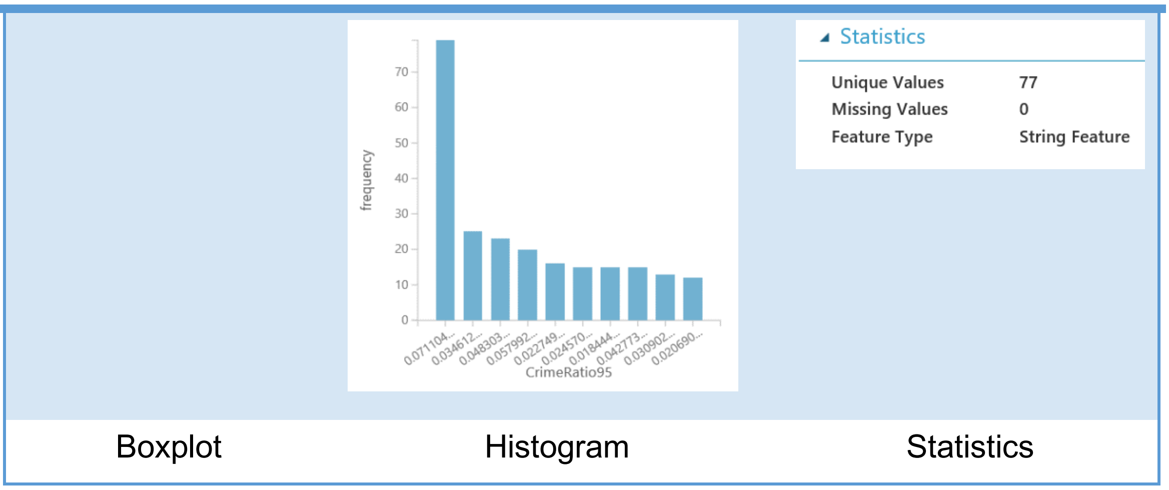


◢ Statistics

Mean	3.5516
Median	3.49
Min	0.43
Max	9.4
Standard Deviation	2.1404
Unique Values	73
Missing Values	0
Feature Type	Numeric Feature

Figure C-16: Exploratory data analysis of the attribute - unemployment96

#17 **crime ratio95**

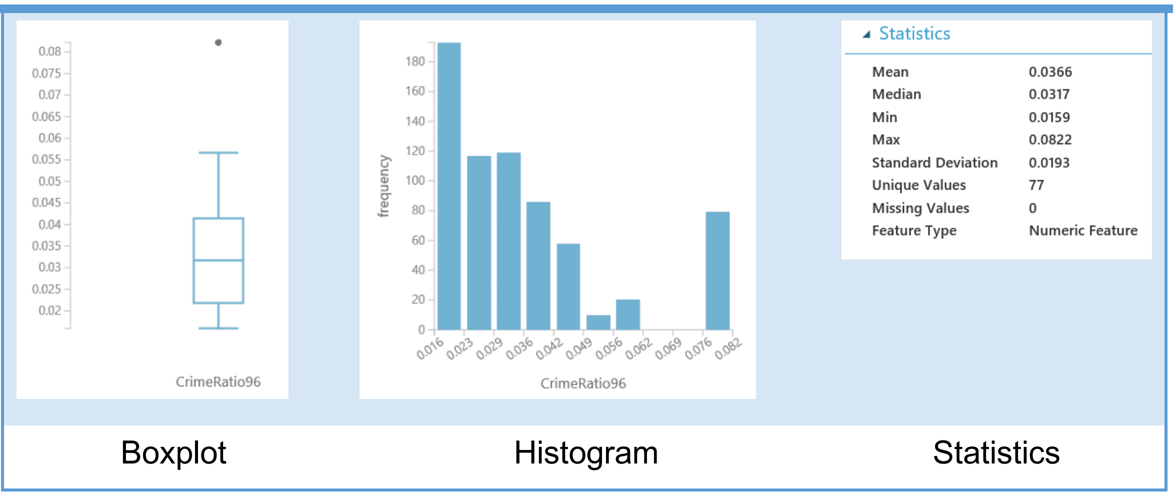


◢ Statistics

Unique Values	77
Missing Values	0
Feature Type	String Feature

Figure C-17: Exploratory data analysis of the attribute - crime ratio95

#18 **crime ratio96**

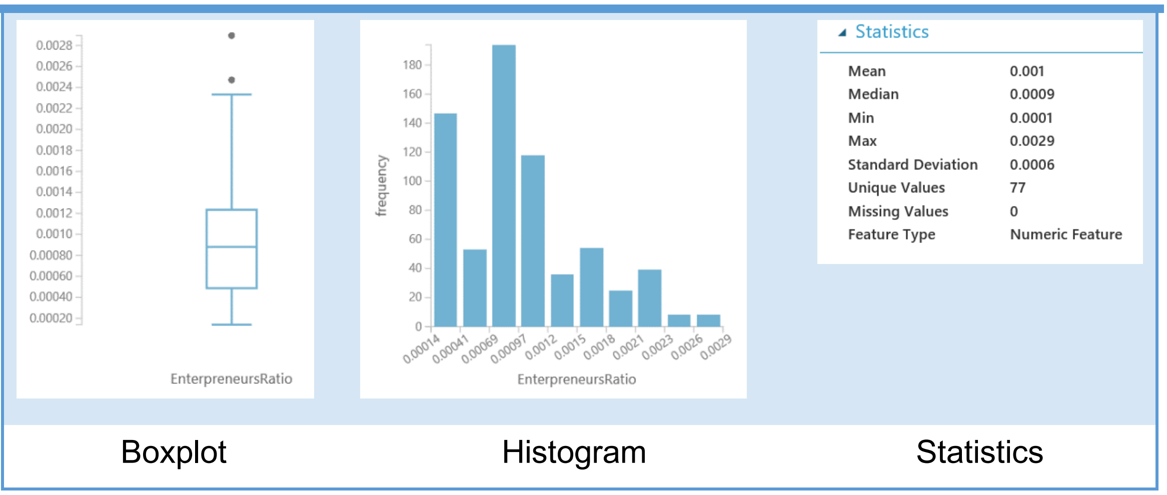


Statistics

Mean	0.0366
Median	0.0317
Min	0.0159
Max	0.0822
Standard Deviation	0.0193
Unique Values	77
Missing Values	0
Feature Type	Numeric Feature

Figure C-18: Exploratory data analysis of the attribute - crime ratio96

#19 **entrepreneur’s ratio**

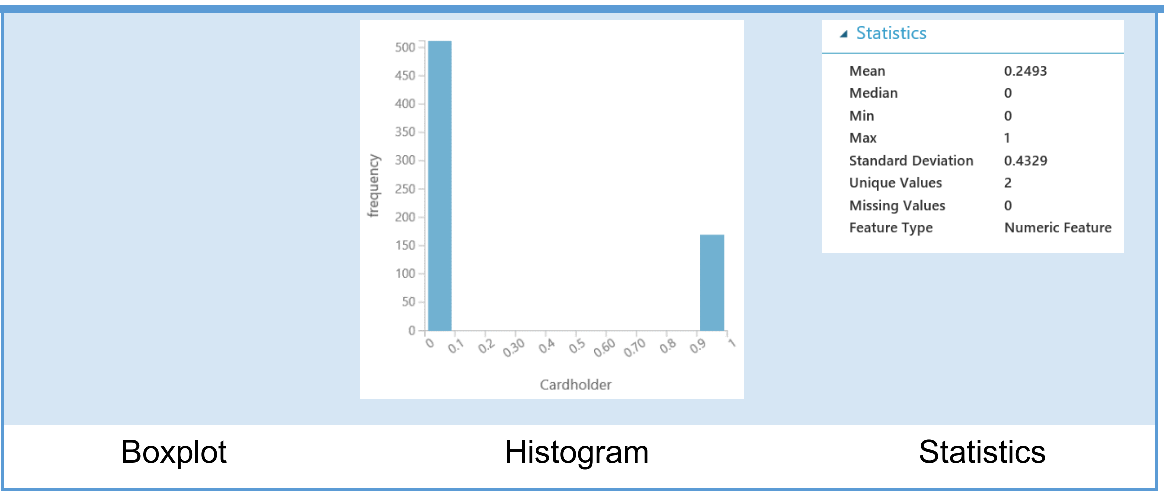


Statistics

Mean	0.001
Median	0.0009
Min	0.0001
Max	0.0029
Standard Deviation	0.0006
Unique Values	77
Missing Values	0
Feature Type	Numeric Feature

Figure C-19: Exploratory data analysis of the attribute - entrepreneur’s ratio

#20 **cardholder**



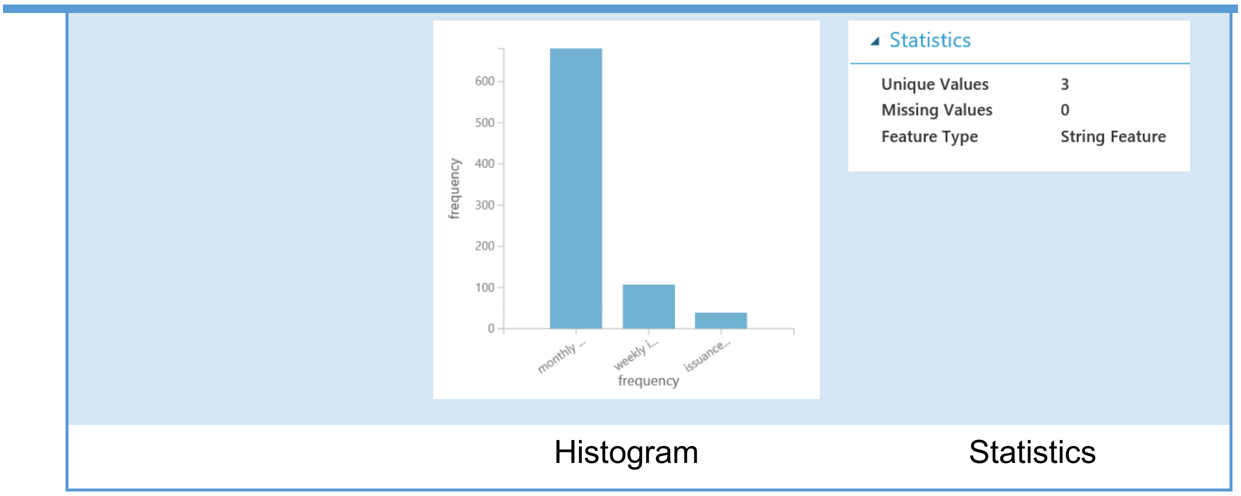
Statistics

Mean	0.2493
Median	0
Min	0
Max	1
Standard Deviation	0.4329
Unique Values	2
Missing Values	0
Feature Type	Numeric Feature

Figure C-20: Exploratory data analysis of the attribute - cardholder

C.2 Exploratory data analysis for the 2nd research project - cross-selling (display_creditcard_azure.csv)

#1 **frequency**

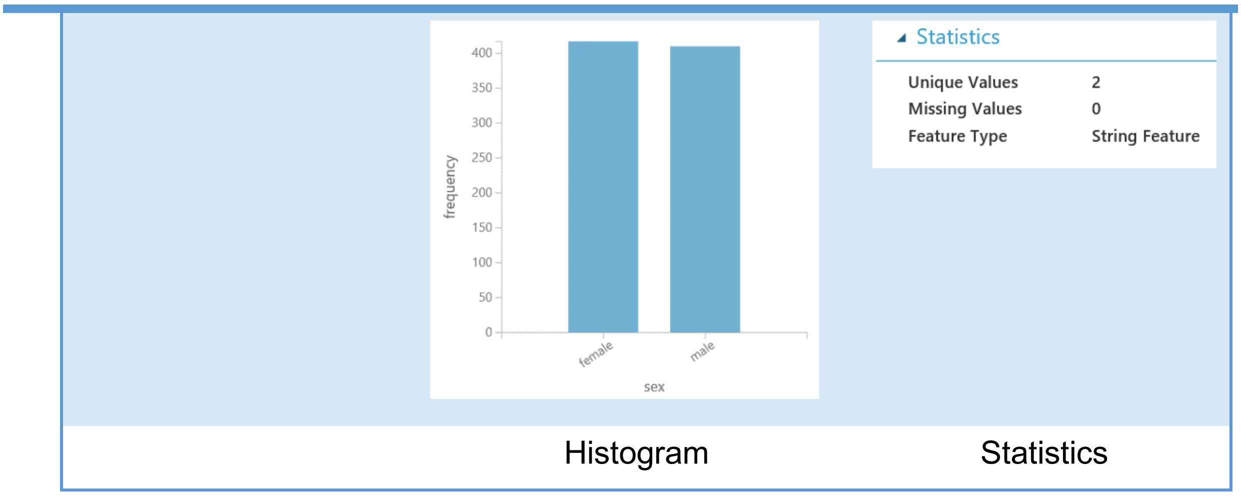


Unique Values 3
Missing Values 0
Feature Type String Feature

Histogram Statistics

Figure C-21: Exploratory data analysis of the attribute - frequency

#2 **sex**

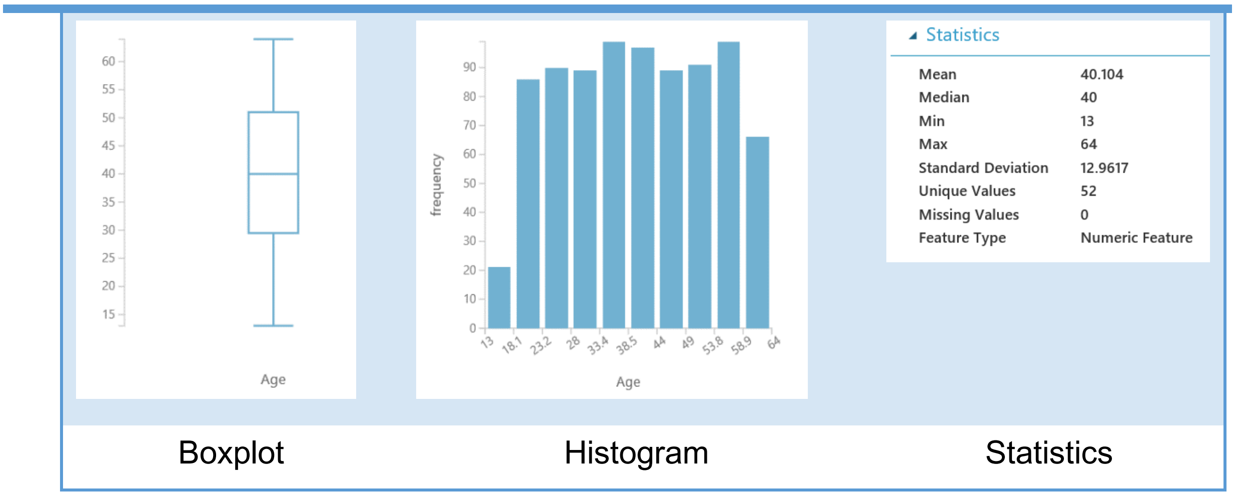


Unique Values 2
Missing Values 0
Feature Type String Feature

Histogram Statistics

Figure C-22: Exploratory data analysis of the attribute - sex

#3 **age**

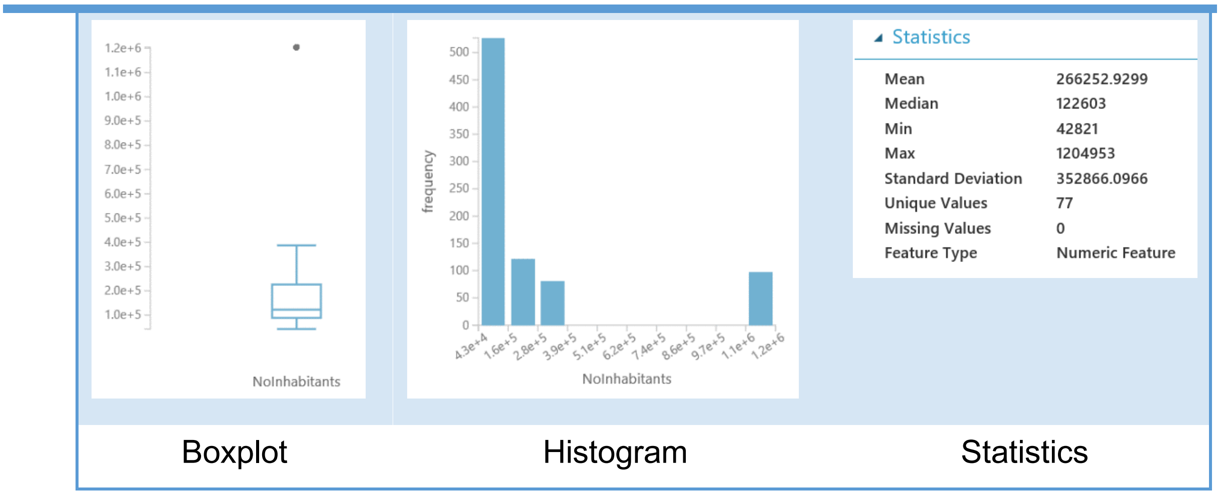


Statistics

Mean	40.104
Median	40
Min	13
Max	64
Standard Deviation	12.9617
Unique Values	52
Missing Values	0
Feature Type	Numeric Feature

Figure C-23: Exploratory data analysis of the attribute - age

#4 **no inhabitants**

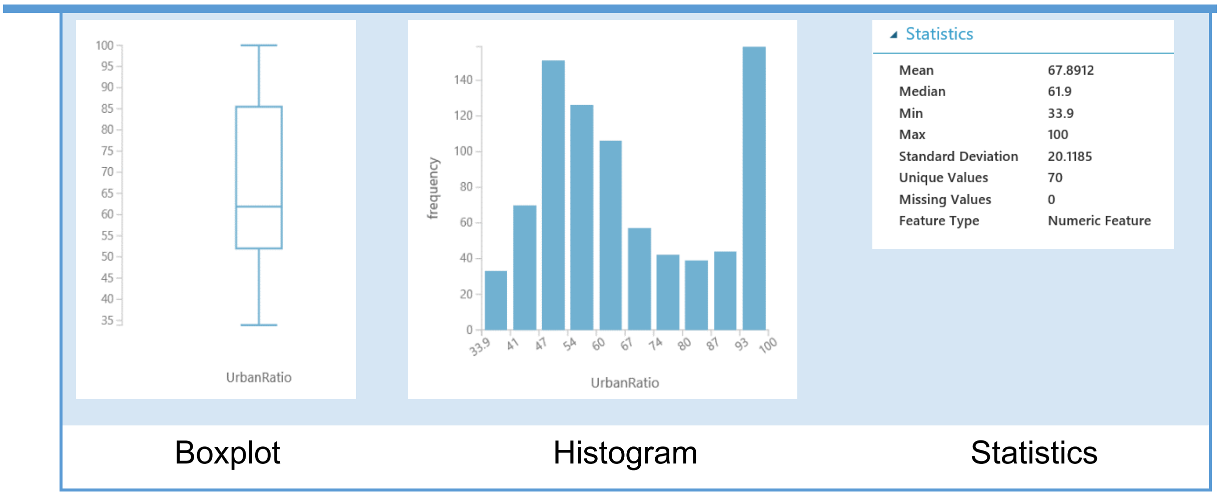


Statistics

Mean	266252.9299
Median	122603
Min	42821
Max	1204953
Standard Deviation	352866.0966
Unique Values	77
Missing Values	0
Feature Type	Numeric Feature

Figure C-24: Exploratory data analysis of the attribute - no inhabitants

#5 **urban ratio**

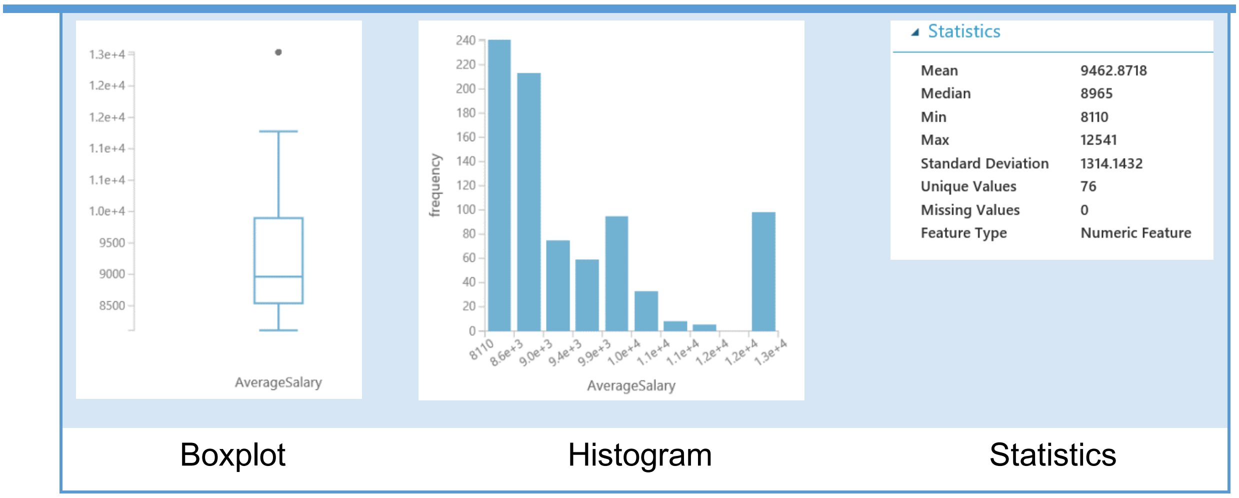


Statistics

Mean	67.8912
Median	61.9
Min	33.9
Max	100
Standard Deviation	20.1185
Unique Values	70
Missing Values	0
Feature Type	Numeric Feature

Figure C-25: Exploratory data analysis of the attribute - urban ratio

#6 **average salary**

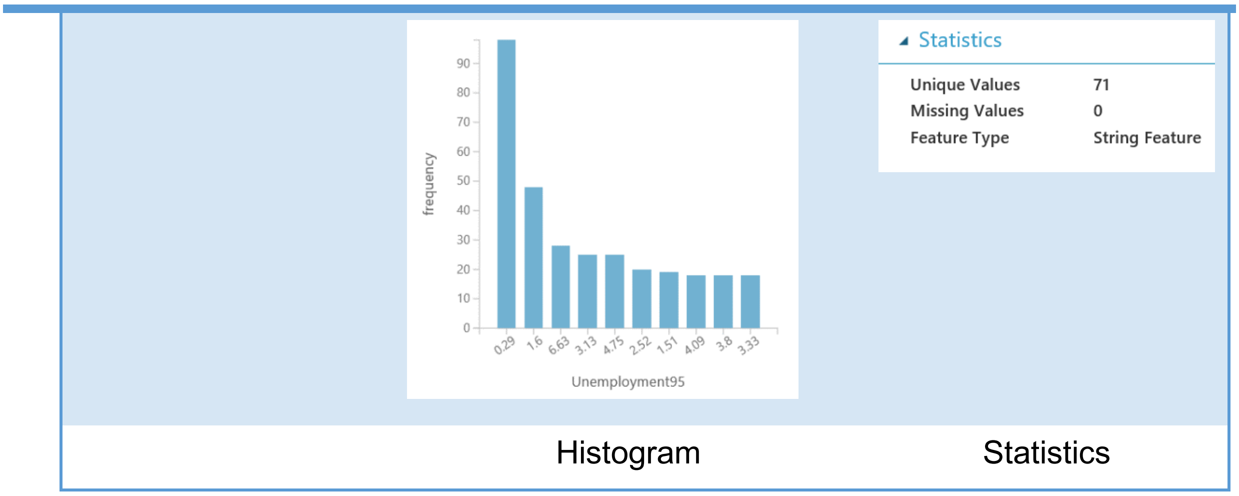


◢ Statistics

Mean	9462.8718
Median	8965
Min	8110
Max	12541
Standard Deviation	1314.1432
Unique Values	76
Missing Values	0
Feature Type	Numeric Feature

Figure C-26: Exploratory data analysis of the attribute - average salary

#7 **unemployment95**

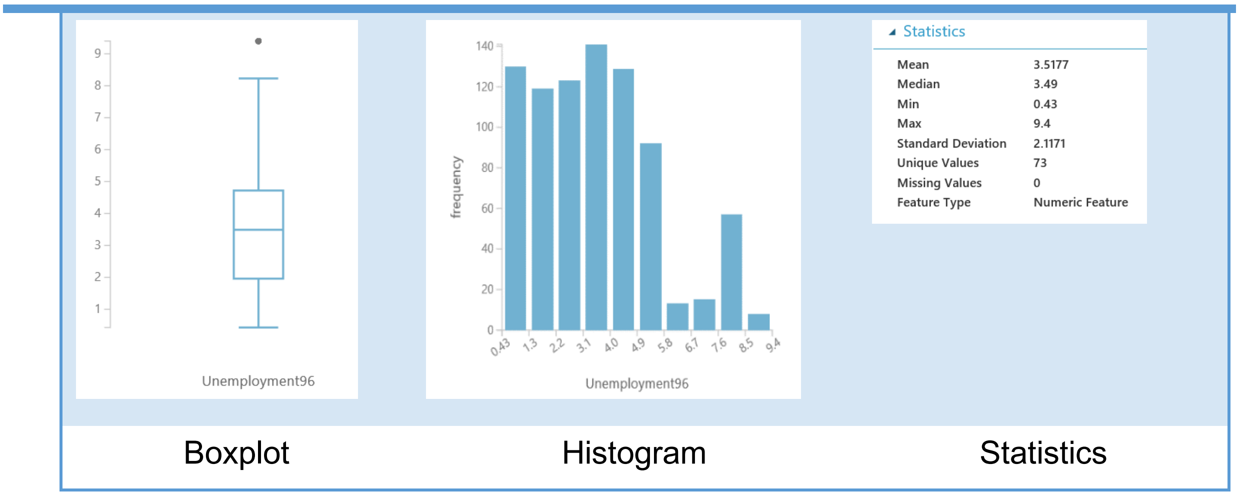


◢ Statistics

Unique Values	71
Missing Values	0
Feature Type	String Feature

Figure C-27: Exploratory data analysis of the attribute - unemployment95

#8 **Unemployment96**

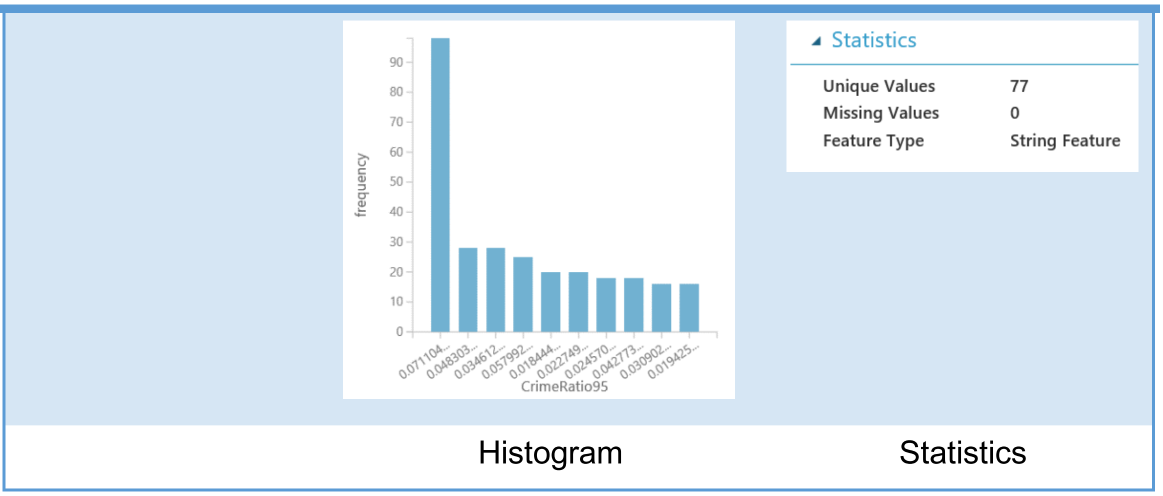


◢ Statistics

Mean	3.5177
Median	3.49
Min	0.43
Max	9.4
Standard Deviation	2.1171
Unique Values	73
Missing Values	0
Feature Type	Numeric Feature

Figure C-28: Exploratory data analysis of the attribute - unemployment96

#9 **crime ratio95**

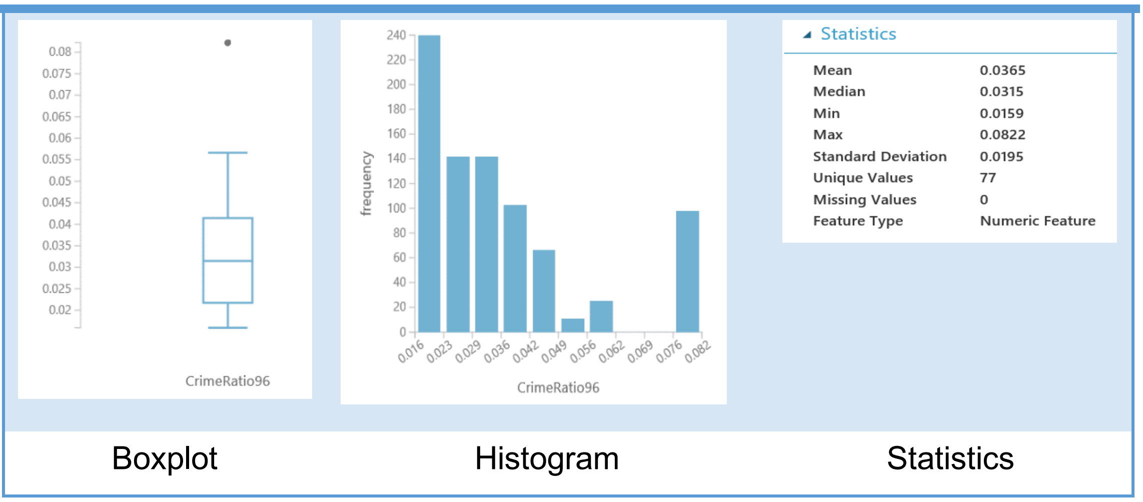


Statistics

Unique Values	77
Missing Values	0
Feature Type	String Feature

Figure C-29: Exploratory data analysis of the attribute - crime ratio95

#10 **Crime ratio96**

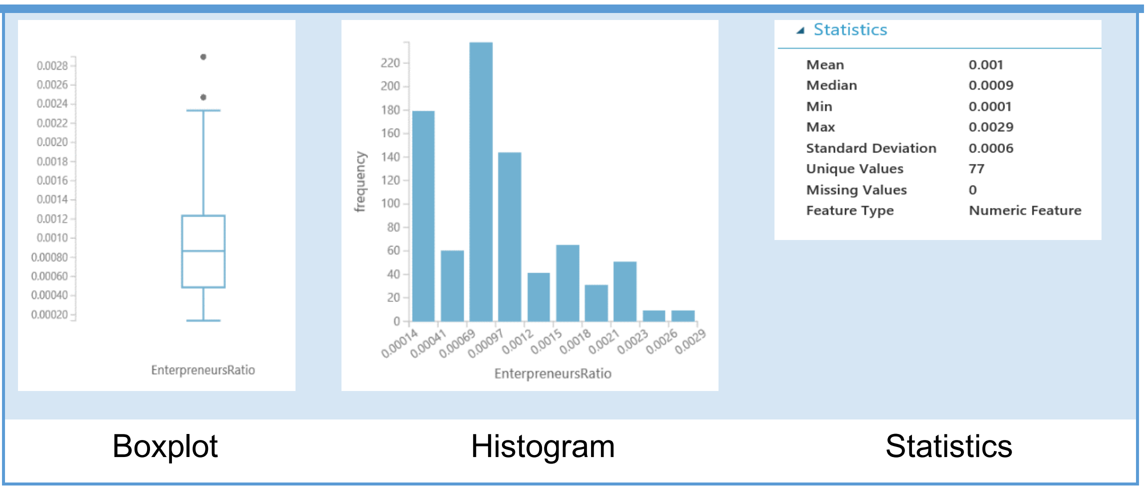


Statistics

Mean	0.0365
Median	0.0315
Min	0.0159
Max	0.0822
Standard Deviation	0.0195
Unique Values	77
Missing Values	0
Feature Type	Numeric Feature

Figure C-30: Exploratory data analysis of the attribute - crime ratio96

#11 **entrepreneur’s ratio**

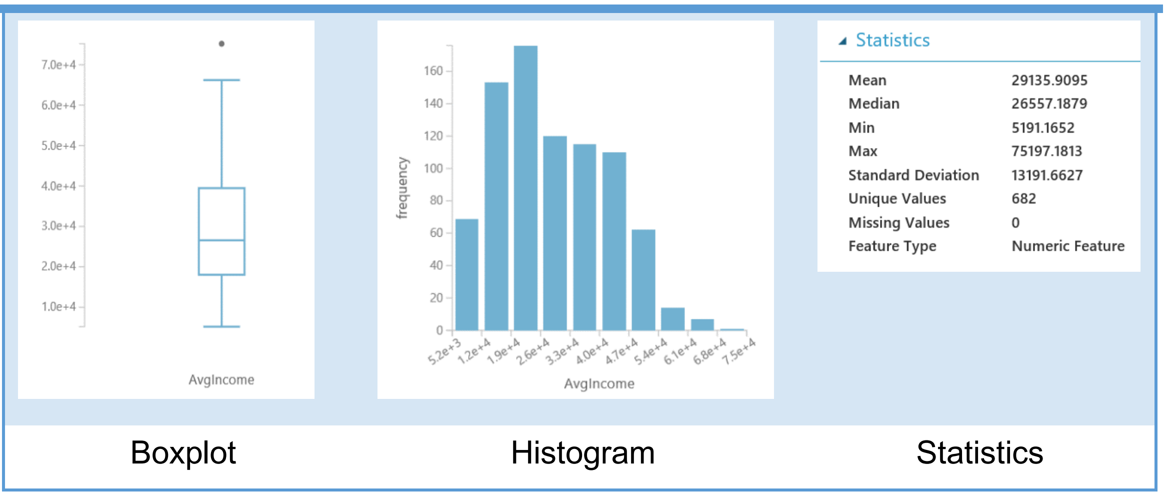


Statistics

Mean	0.001
Median	0.0009
Min	0.0001
Max	0.0029
Standard Deviation	0.0006
Unique Values	77
Missing Values	0
Feature Type	Numeric Feature

Figure C-31: Exploratory data analysis of the attribute - entrepreneur’s ratio

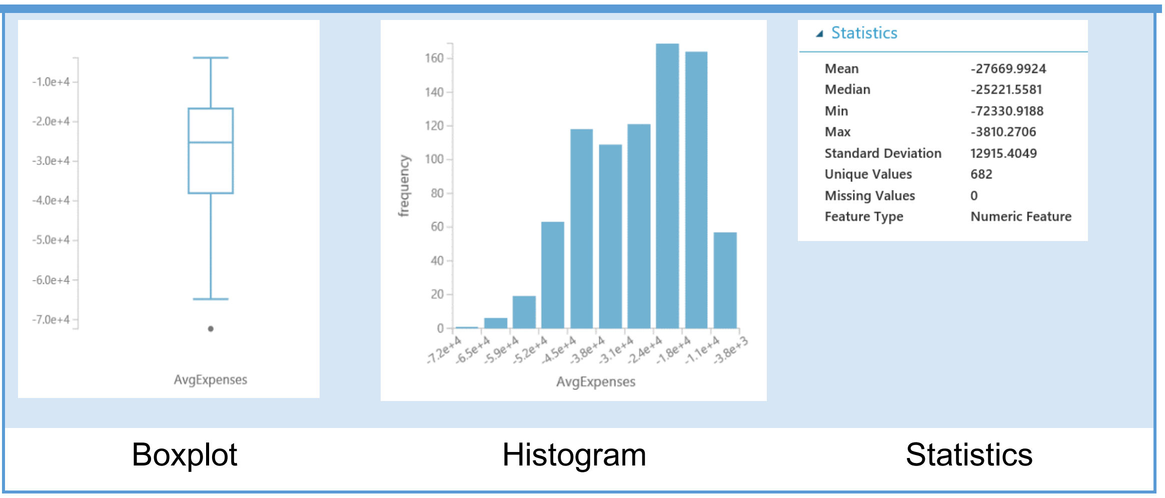
#12 **average income**



Mean	29135.9095
Median	26557.1879
Min	5191.1652
Max	75197.1813
Standard Deviation	13191.6627
Unique Values	682
Missing Values	0
Feature Type	Numeric Feature

Figure C-32: Exploratory data analysis of the attribute - average income

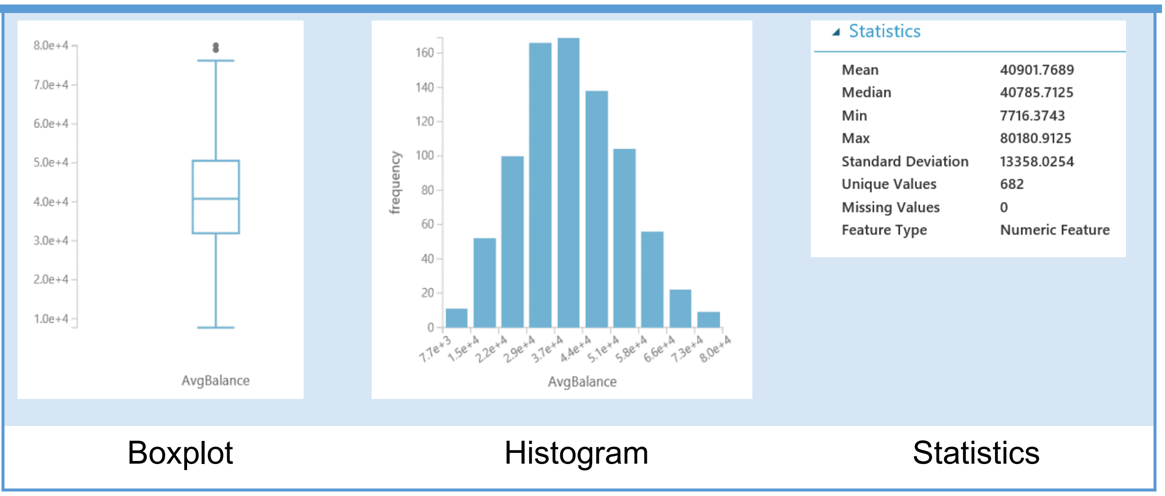
#13 **average expenses**



Mean	-27669.9924
Median	-25221.5581
Min	-72330.9188
Max	-3810.2706
Standard Deviation	12915.4049
Unique Values	682
Missing Values	0
Feature Type	Numeric Feature

Figure C-33: Exploratory data analysis of the attribute - average expenses

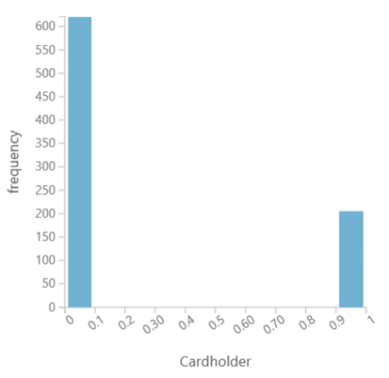
#14 **average balance**

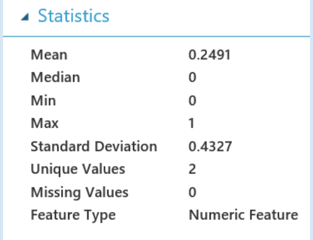


Mean	40901.7689
Median	40785.7125
Min	7716.3743
Max	80180.9125
Standard Deviation	13358.0254
Unique Values	682
Missing Values	0
Feature Type	Numeric Feature

Figure C-34: Exploratory data analysis of the attribute - average income

#15 **cardholder**



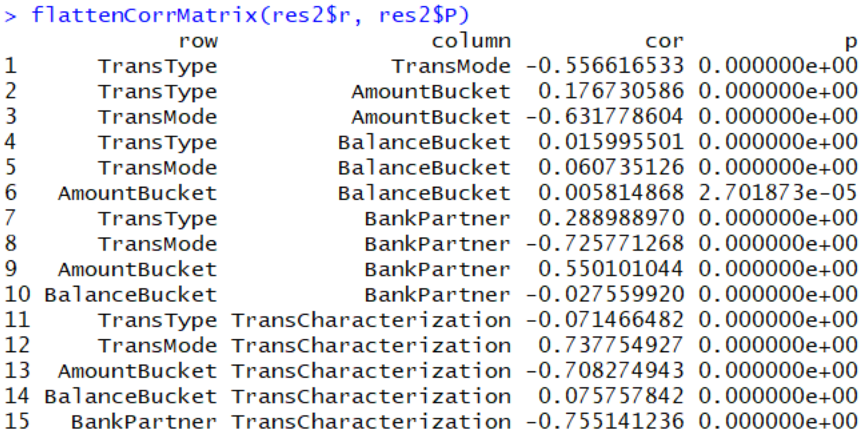

◢ Statistics

Mean	0.2491
Median	0
Min	0
Max	1
Standard Deviation	0.4327
Unique Values	2
Missing Values	0
Feature Type	Numeric Feature

Boxplot	Histogram	Statistics

Figure C-35: Exploratory data analysis of the attribute - cardholder

C.3 Exploratory data analysis for the 3rd research project - categorized transactional payment behaviour



```
> flattenCorrMatrix(res2$r, res2$P)
             row                column          cor            p
1      TransType             TransMode -0.556616533 0.000000e+00
2      TransType          AmountBucket  0.176730586 0.000000e+00
3      TransMode          AmountBucket -0.631778604 0.000000e+00
4      TransType         BalanceBucket  0.015995501 0.000000e+00
5      TransMode         BalanceBucket  0.060735126 0.000000e+00
6   AmountBucket         BalanceBucket  0.005814868 2.701873e-05
7      TransType           BankPartner  0.288988970 0.000000e+00
8      TransMode           BankPartner -0.725771268 0.000000e+00
9   AmountBucket           BankPartner  0.550101044 0.000000e+00
10 BalanceBucket           BankPartner -0.027559920 0.000000e+00
11     TransType TransCharacterization -0.071466482 0.000000e+00
12     TransMode TransCharacterization  0.737754927 0.000000e+00
13  AmountBucket TransCharacterization -0.708274943 0.000000e+00
14 BalanceBucket TransCharacterization  0.075757842 0.000000e+00
15   BankPartner TransCharacterization -0.755141236 0.000000e+00
```

s

Figure C-36: Flatten correlation matrix of the data table mydataTrans

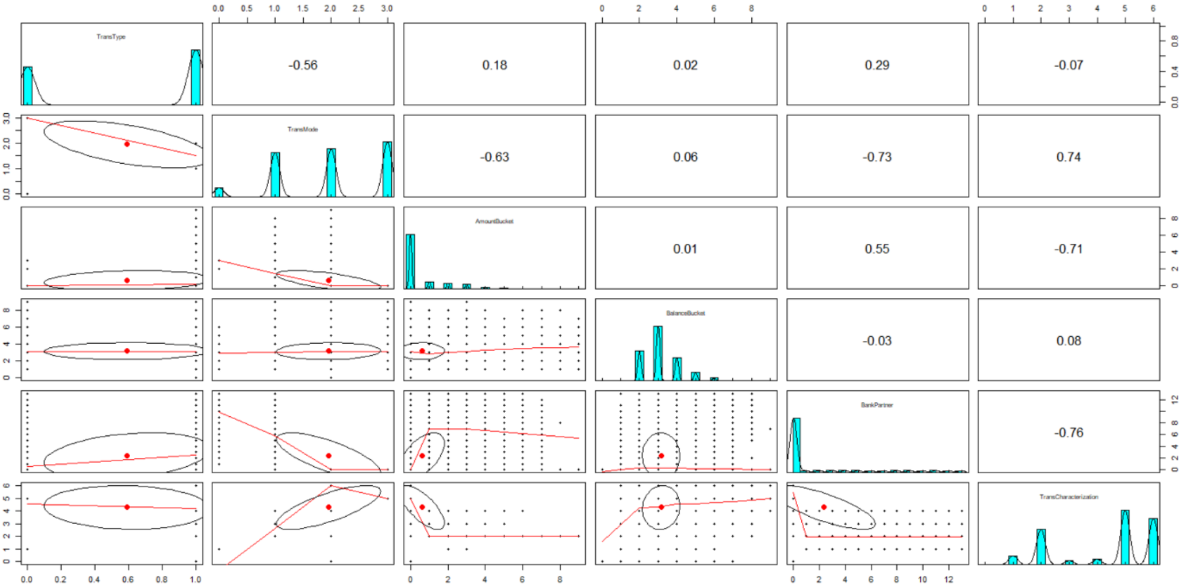
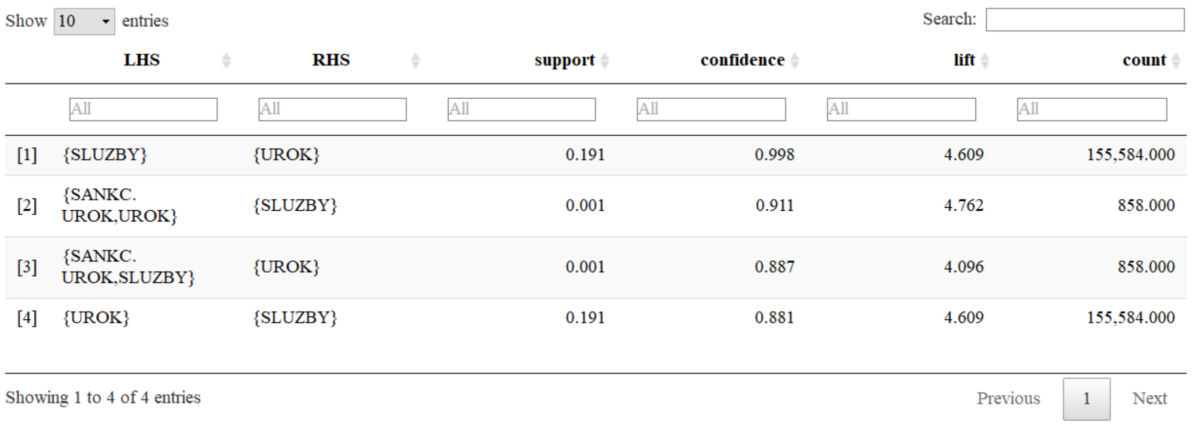


Figure C-37: Matrix of scatterplots (pairs panels) for the data table mydataTrans



Show 10 entries

Search:

	LHS	RHS	support	confidence	lift	count
	All	All	All	All	All	All
[1]	{SLUZBY}	{UROK}	0.191	0.998	4.609	155,584.000
[2]	{SANKC. UROK,UROK}	{SLUZBY}	0.001	0.911	4.762	858.000
[3]	{SANKC. UROK,SLUZBY}	{UROK}	0.001	0.887	4.096	858.000
[4]	{UROK}	{SLUZBY}	0.191	0.881	4.609	155,584.000

Showing 1 to 4 of 4 entries

Figure C-38: Interactive data table for the mined rule set

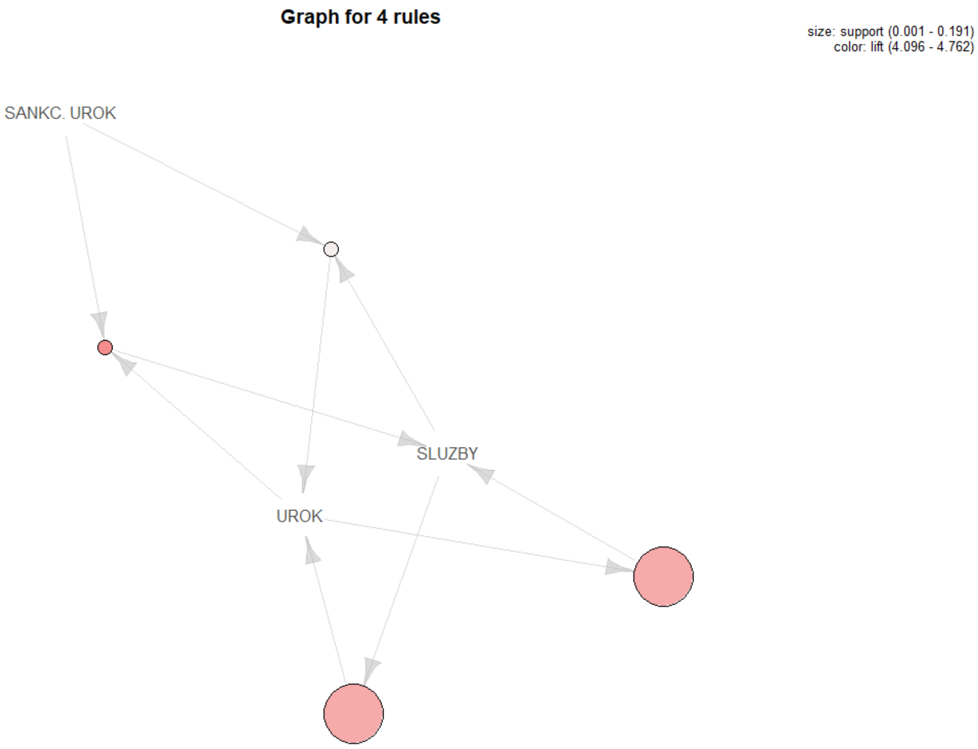


Figure C-39: Graph-based visualization of the four generated rules

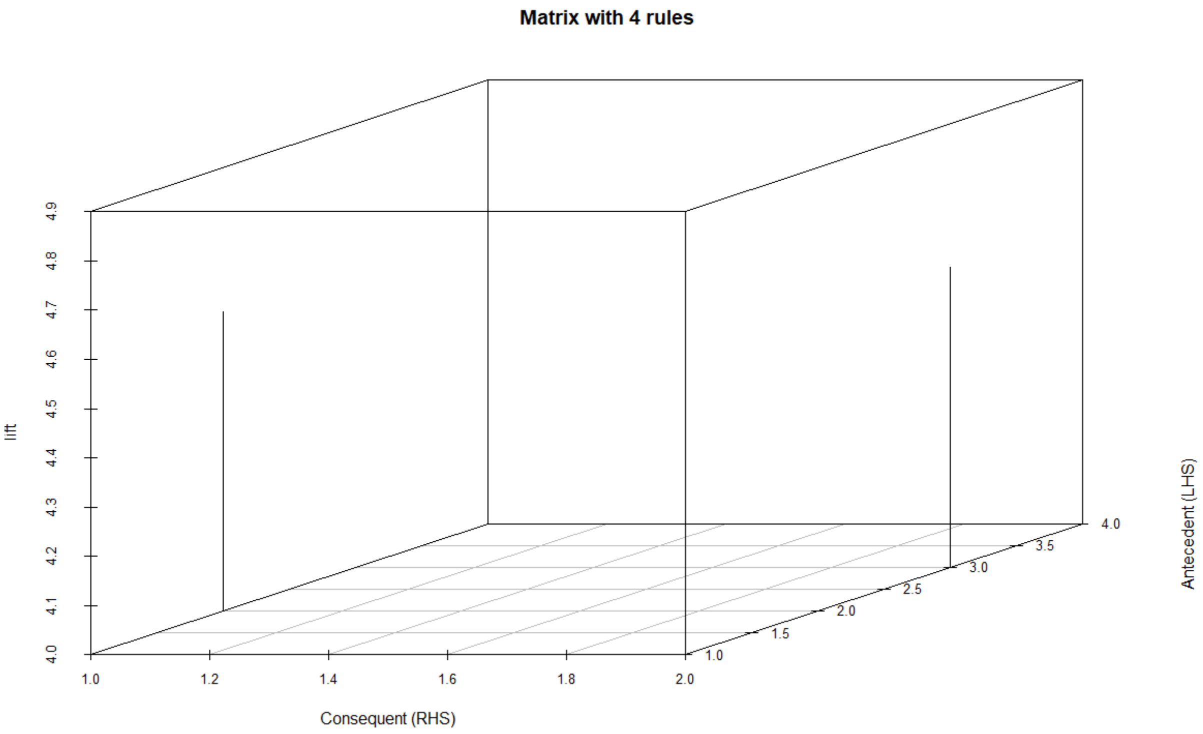
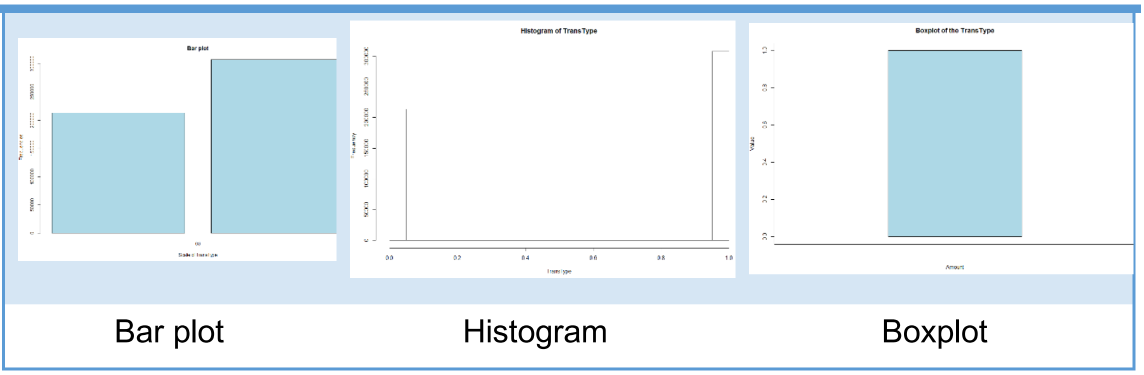


Figure C-40: Matrix-based visualization with 3D bars for the four generated rules

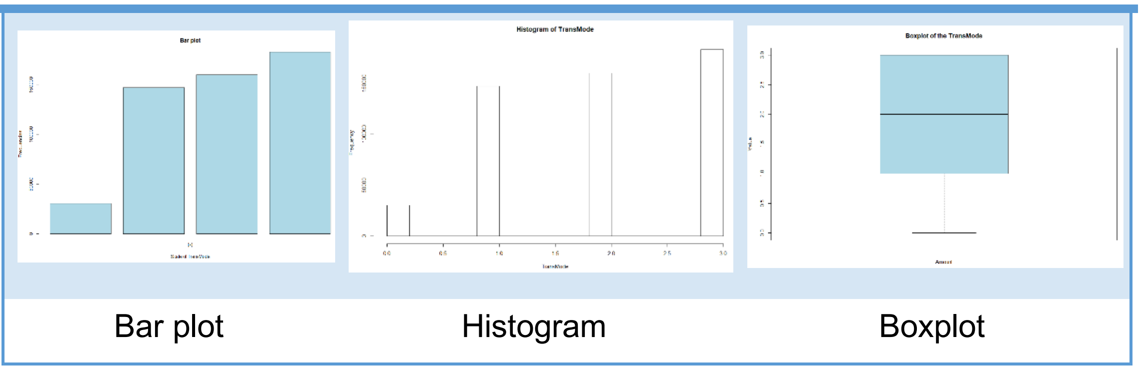
#1 **trans type**



Bar plot Histogram Boxplot

Figure C-41: Exploratory data analysis of the attribute - trans type

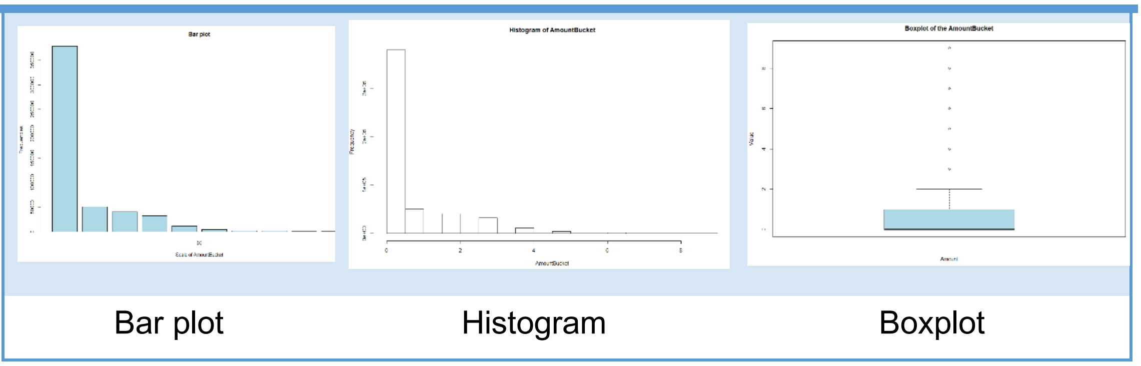
#2 **trans mode**



Bar plot Histogram Boxplot

Figure C-42: Exploratory data analysis of the attribute - trans mode

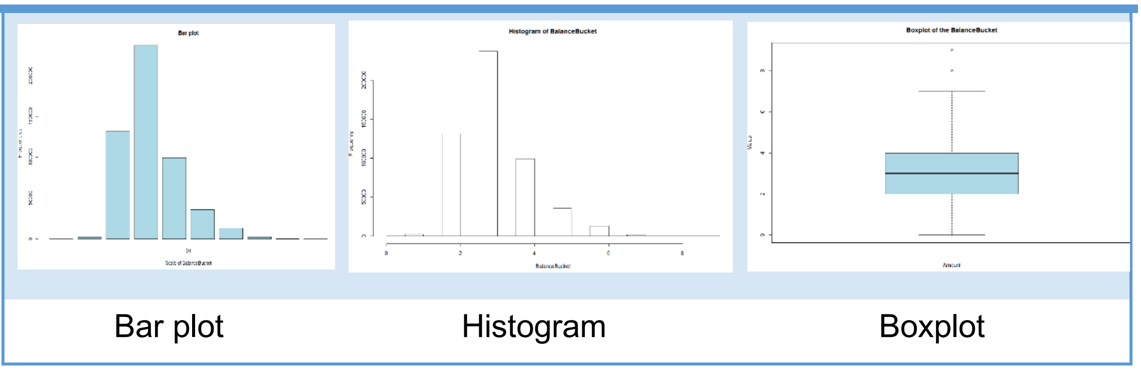
#3 **amount bucket**



Bar plot Histogram Boxplot

Figure C-43: Exploratory data analysis of the attribute - trans mode

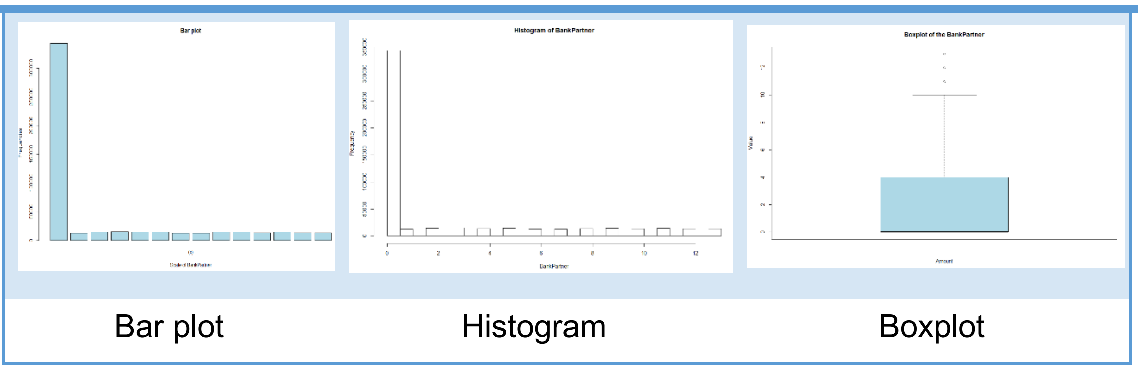
#4 **balance bucket**



Bar plot Histogram Boxplot

Figure C-44: Exploratory data analysis of the attribute - balance bucket

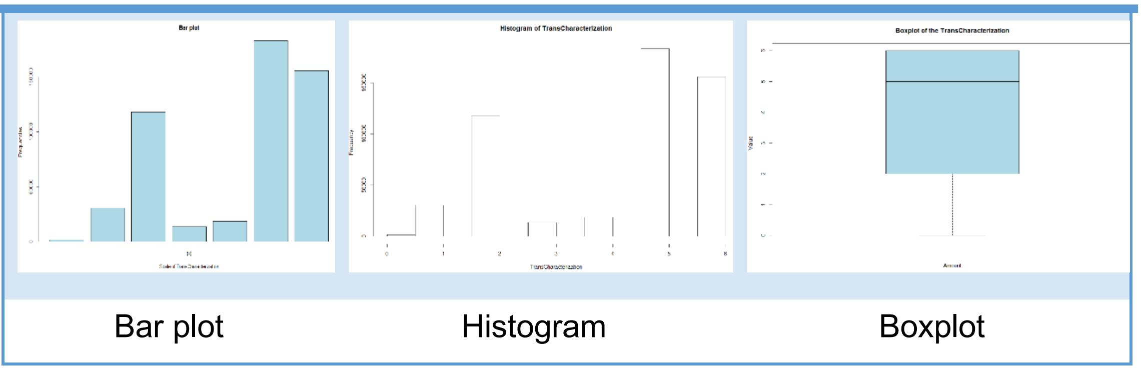
#5 **bank partner**



Bar plot Histogram Boxplot

Figure C-45: Exploratory data analysis of the attribute - bank partner

#6 **TransCharacterization**



Bar plot Histogram Boxplot

Figure C-46: Exploratory data analysis of the attribute - TransCharacterization

Appendix D: Predictive analysis results - visualized results

D.1. Credit-scoring: Mini Map overview (res_azure.csv)

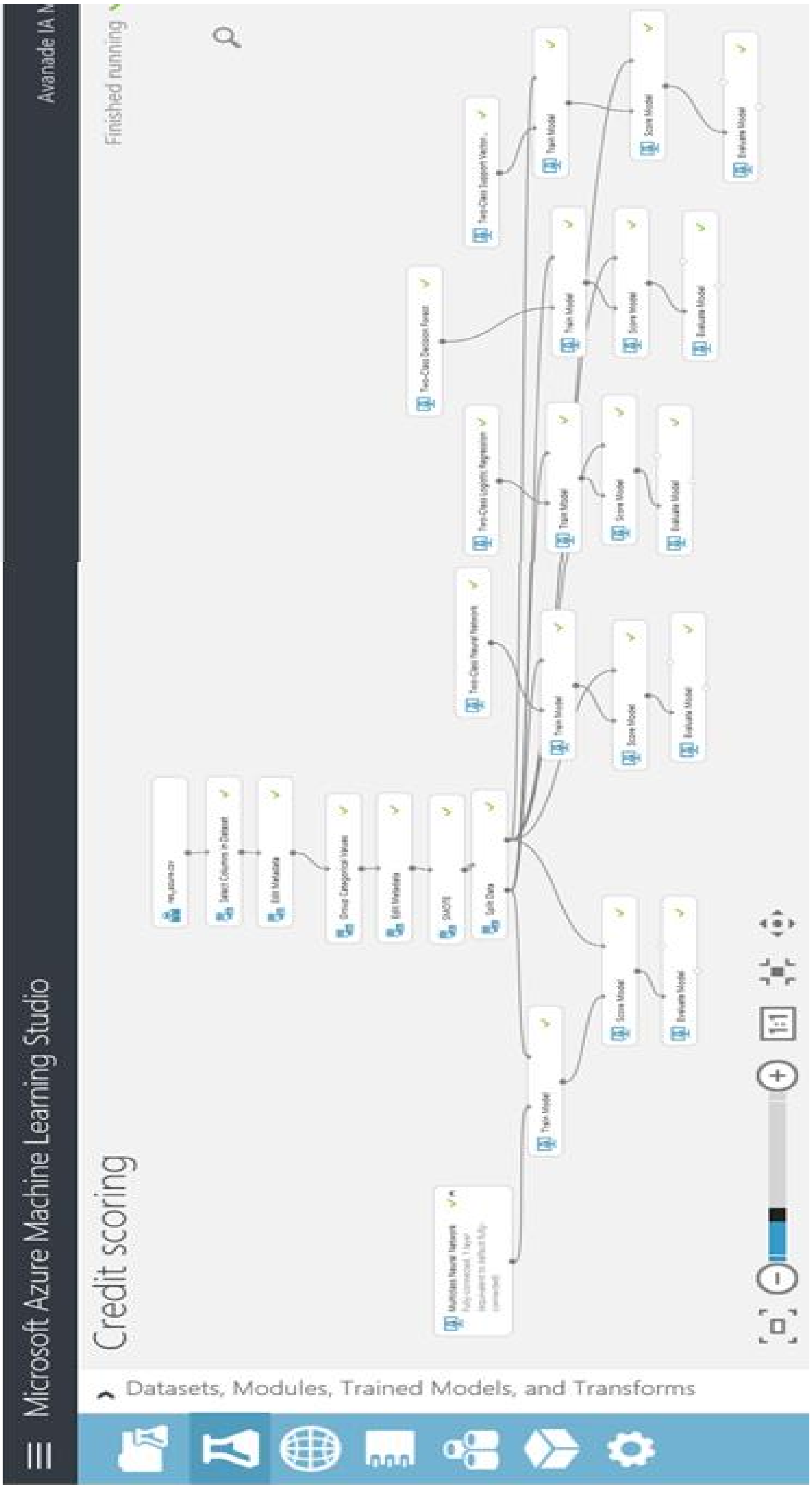
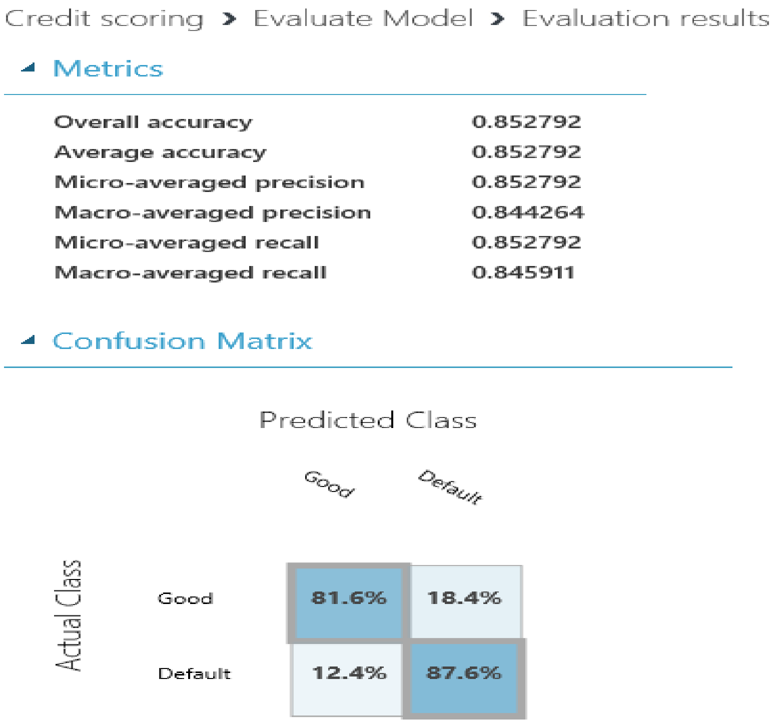

Microsoft Azure Machine Learning Studio
Avanade IA M
Credit scoring
Finished running
res_azure.csv
Select Columns in Dataset
Edit Metadata
Group Categorical Values
Edit Metadata
SMOTE
Split Data
Two-Class Support Vector...
Two-Class Decision Forest
Two-Class Logistic Regression
Two-Class Neural Network
Multiclass Neural Network
Train Model
Score Model
Evaluate Model
1:1
Datasets, Modules, Trained Models, and Transforms

Figure D-1: Credit-scoring - Mini Map overview

D.2. Evaluation results of supervised learning algorithms

D.2.1 Multiclass Neural Network



Credit scoring › Evaluate Model › Evaluation results

Metrics

Overall accuracy	0.852792
Average accuracy	0.852792
Micro-averaged precision	0.852792
Macro-averaged precision	0.844264
Micro-averaged recall	0.852792
Macro-averaged recall	0.845911

Confusion Matrix

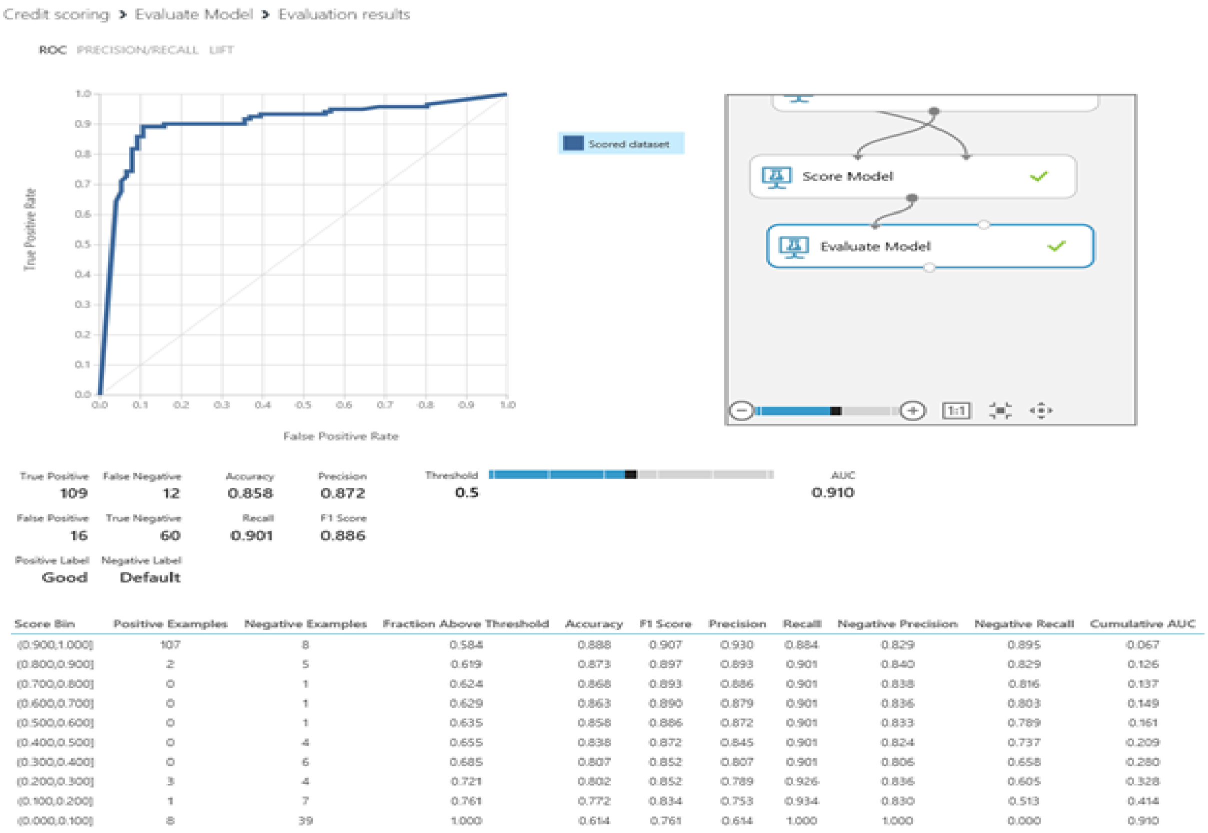
Predicted Class

Actual Class

	Good	Default
Good	81.6%	18.4%
Default	12.4%	87.6%

Figure D-2: Credit-scoring evaluation results for Multiclass Neural Network

D.2.2 Two-Class Neural Network

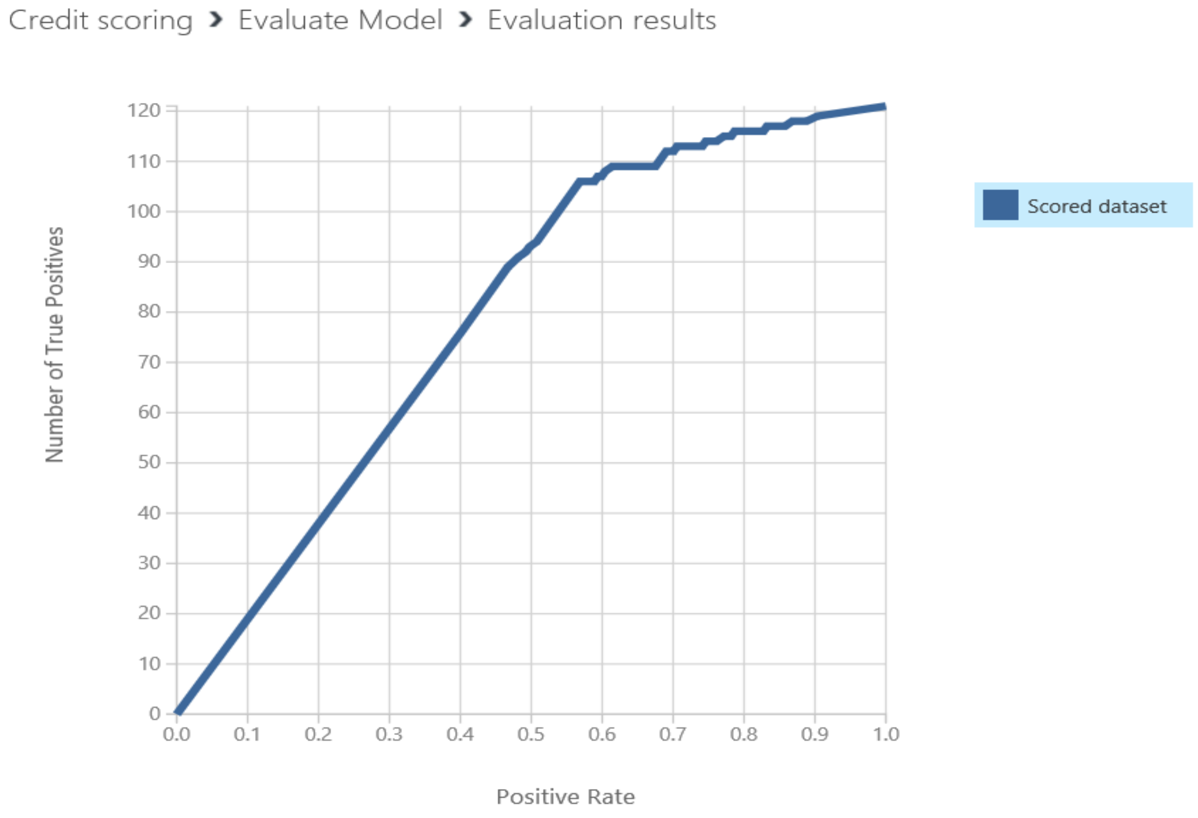


Credit scoring › Evaluate Model › Evaluation results

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
109	12	0.858	0.872	0.5	0.910
False Positive	**True Negative**	**Recall**	**F1 Score**		
16	60	0.901	0.886		
Positive Label	**Negative Label**				
Good	Default				

Score Bin	Positive Examples	Negative Examples	Fraction Above Threshold	Accuracy	F1 Score	Precision	Recall	Negative Precision	Negative Recall	Cumulative AUC
(0.900,1.000]	107	8	0.584	0.888	0.907	0.930	0.884	0.829	0.895	0.067
(0.800,0.900]	2	5	0.619	0.873	0.897	0.893	0.901	0.840	0.829	0.126
(0.700,0.800]	0	1	0.624	0.868	0.893	0.886	0.901	0.838	0.816	0.137
(0.600,0.700]	0	1	0.629	0.863	0.890	0.879	0.901	0.836	0.803	0.149
(0.500,0.600]	0	1	0.635	0.858	0.886	0.872	0.901	0.833	0.789	0.161
(0.400,0.500]	0	4	0.655	0.838	0.872	0.845	0.901	0.824	0.737	0.209
(0.300,0.400]	0	6	0.685	0.807	0.852	0.807	0.901	0.806	0.658	0.280
(0.200,0.300]	3	4	0.721	0.802	0.852	0.789	0.926	0.836	0.605	0.328
(0.100,0.200]	1	7	0.761	0.772	0.834	0.753	0.934	0.830	0.513	0.414
(0.000,0.100]	8	39	1.000	0.614	0.761	0.614	1.000	1.000	0.000	0.910

Figure D-3: Credit-scoring evaluation results - Two-Class Neural Network



Credit scoring › Evaluate Model › Evaluation results

Figure D-4: Lift response chart for Two-Class Neural Network (credit scoring)

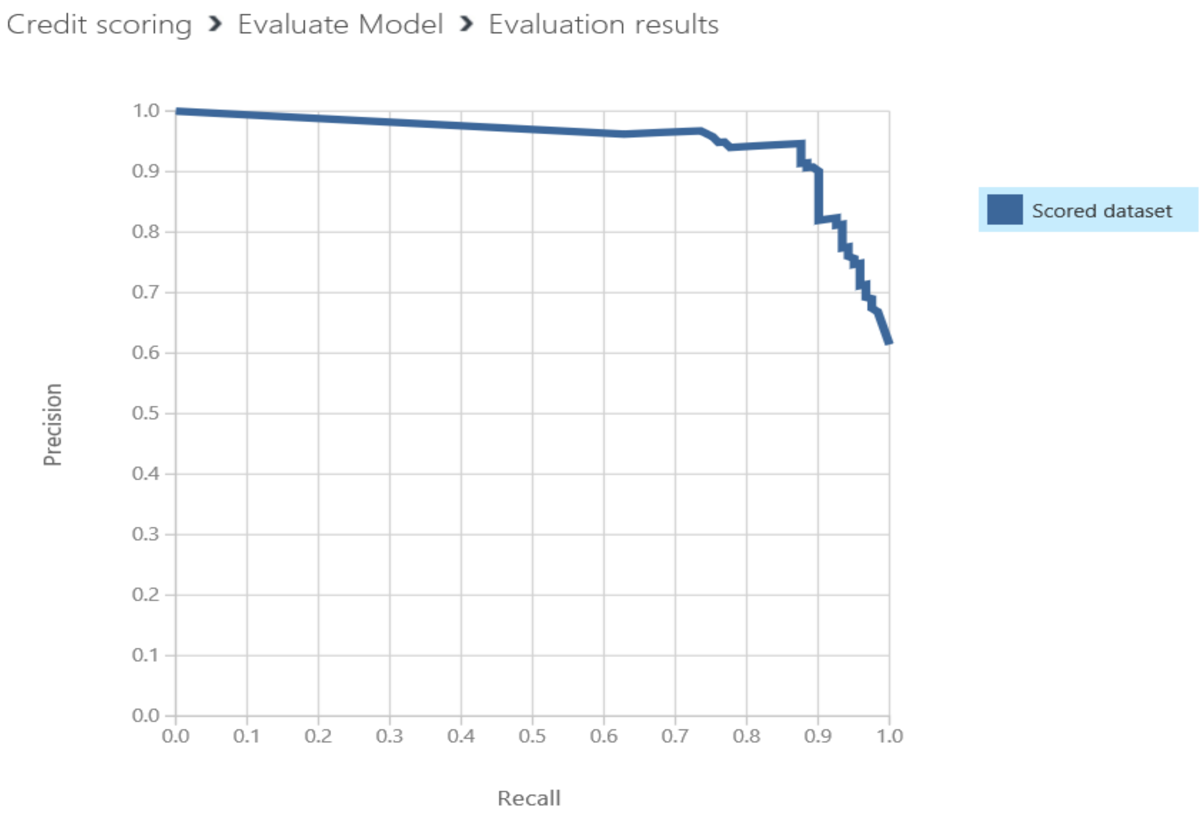
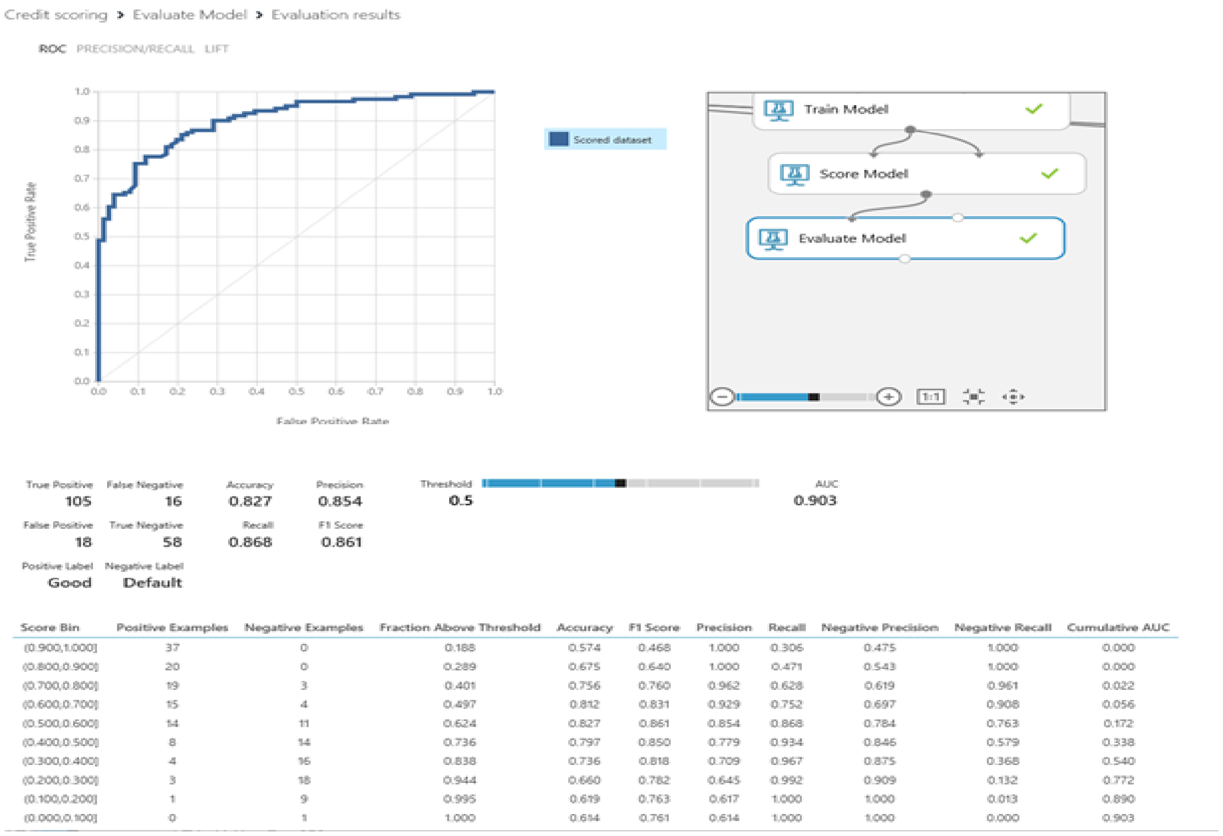


Figure D-5: Precision / Recall response chart for Two-Class Neural Network (credit scoring)

D.2.3 Two-Class Logistic Regression



True Positive	False Negative	Accuracy	Precision	Threshold	AUC
105	16	0.827	0.854	0.5	0.903

False Positive	True Negative	Recall	F1 Score
18	58	0.868	0.861

Positive Label	Negative Label
Good	Default

Score Bin	Positive Examples	Negative Examples	Fraction Above Threshold	Accuracy	F1 Score	Precision	Recall	Negative Precision	Negative Recall	Cumulative AUC
(0.900,1.000]	37	0	0.188	0.574	0.468	1.000	0.306	0.475	1.000	0.000
(0.800,0.900]	20	0	0.289	0.675	0.640	1.000	0.471	0.543	1.000	0.000
(0.700,0.800]	19	3	0.401	0.756	0.760	0.962	0.628	0.619	0.961	0.022
(0.600,0.700]	15	4	0.497	0.812	0.831	0.929	0.752	0.697	0.908	0.056
(0.500,0.600]	14	11	0.624	0.827	0.861	0.854	0.868	0.784	0.763	0.172
(0.400,0.500]	8	14	0.736	0.797	0.850	0.779	0.934	0.846	0.579	0.338
(0.300,0.400]	4	16	0.838	0.736	0.818	0.709	0.967	0.875	0.368	0.540
(0.200,0.300]	3	18	0.944	0.660	0.782	0.645	0.992	0.909	0.132	0.772
(0.100,0.200]	1	9	0.995	0.619	0.763	0.617	1.000	1.000	0.013	0.890
(0.000,0.100]	0	1	1.000	0.614	0.761	0.614	1.000	1.000	0.000	0.903

Figure D-6: Credit-scoring evaluation results - Two-Class Logistic Regression

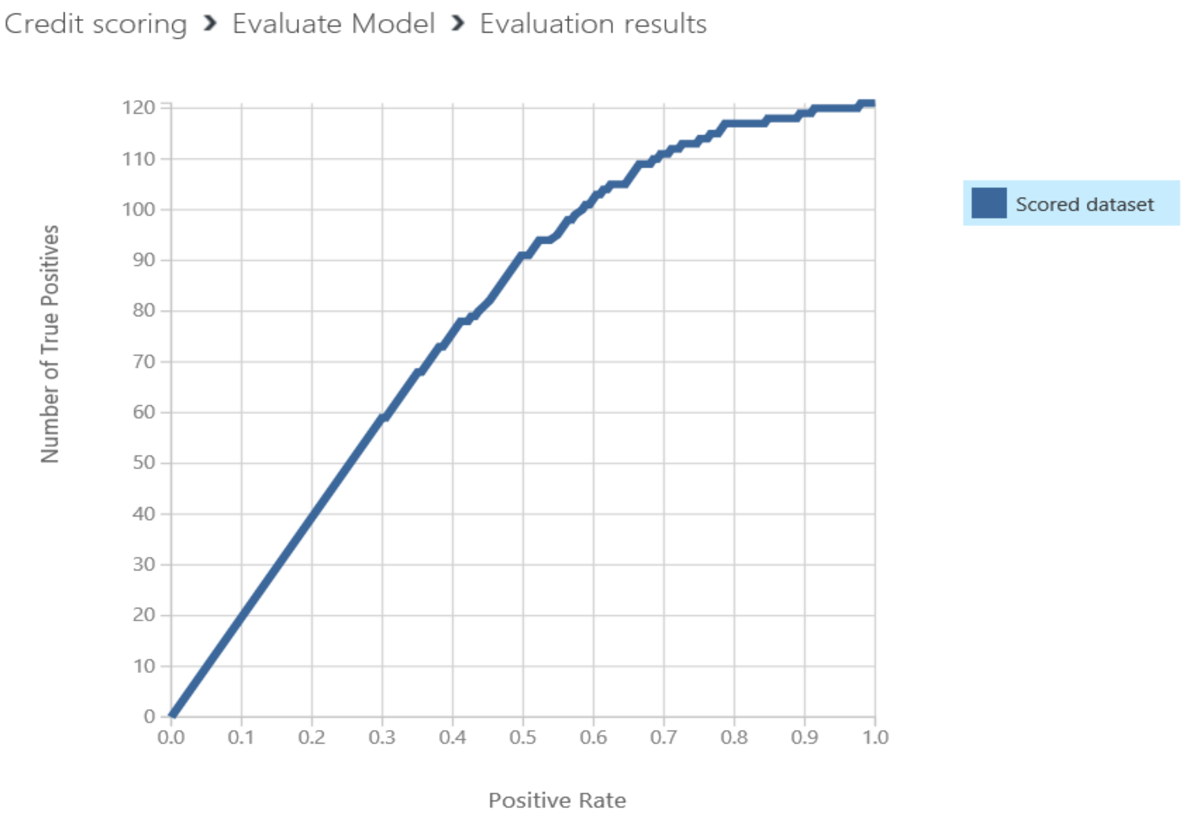


Figure D-7: Lift response chart for Two-Class Logistic Regression (credit scoring)

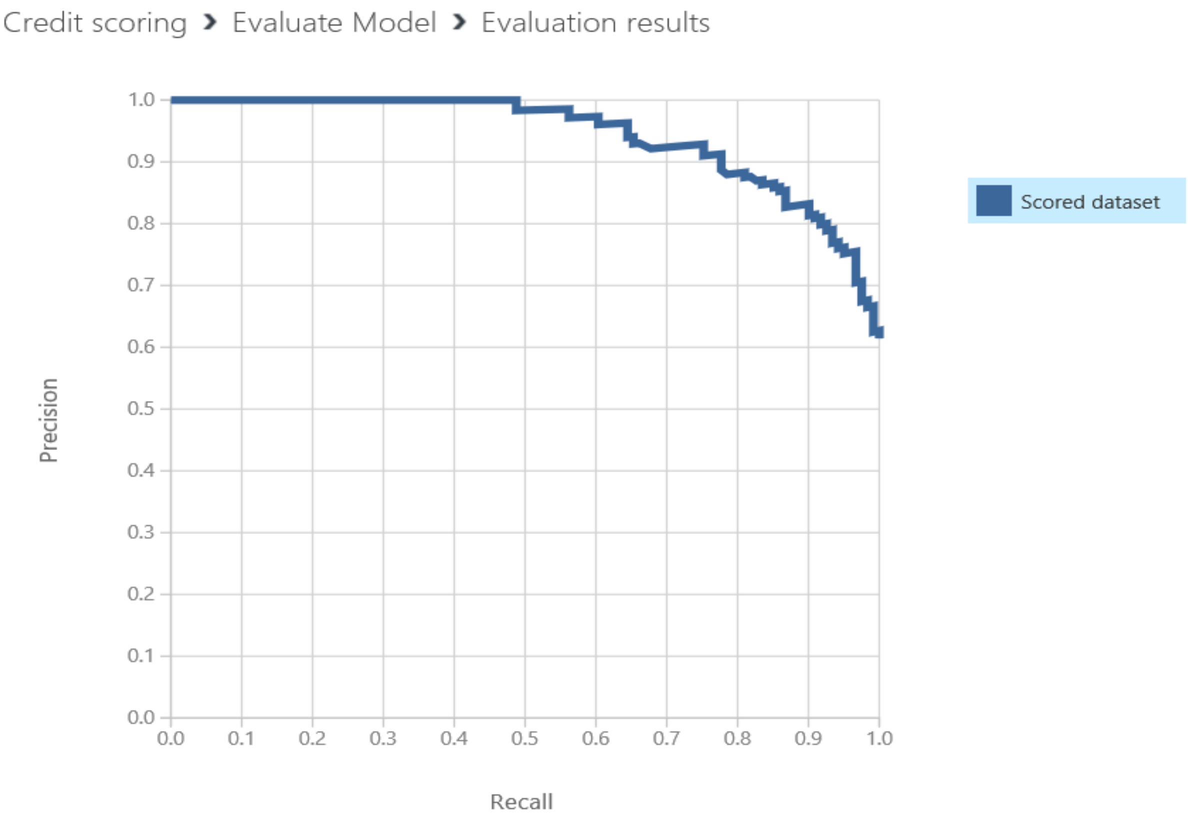
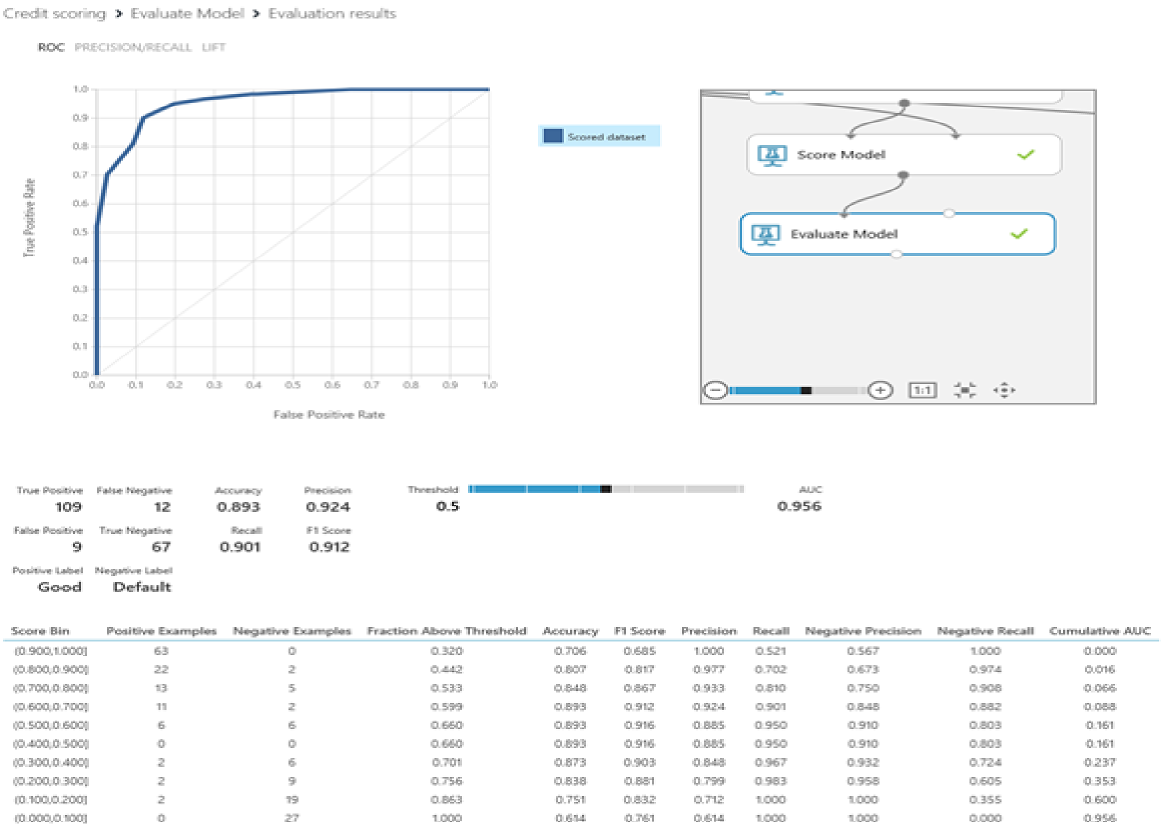


Figure D-8: Precision / Recall response chart for Two-Class Logistic Regression (credit scoring)

D.2.4 Two-Class Decision Forest


Credit scoring › Evaluate Model › Evaluation results

ROC PRECISION/RECALL LIFT

True Positive 109	False Negative 12	Accuracy 0.893	Precision 0.924	Threshold 0.5	AUC 0.956
False Positive 9	True Negative 67	Recall 0.901	F1 Score 0.912		
Positive Label Good	Negative Label Default				

Score Bin	Positive Examples	Negative Examples	Fraction Above Threshold	Accuracy	F1 Score	Precision	Recall	Negative Precision	Negative Recall	Cumulative AUC
(0.900,1.000]	63	0	0.320	0.706	0.685	1.000	0.521	0.567	1.000	0.000
(0.800,0.900]	22	2	0.442	0.807	0.817	0.977	0.702	0.673	0.974	0.016
(0.700,0.800]	13	5	0.533	0.848	0.867	0.933	0.810	0.750	0.908	0.066
(0.600,0.700]	11	2	0.599	0.893	0.912	0.924	0.901	0.848	0.882	0.088
(0.500,0.600]	6	6	0.660	0.893	0.916	0.885	0.950	0.910	0.803	0.161
(0.400,0.500]	0	0	0.660	0.893	0.916	0.885	0.950	0.910	0.803	0.161
(0.300,0.400]	2	6	0.701	0.873	0.903	0.848	0.967	0.932	0.724	0.237
(0.200,0.300]	2	9	0.756	0.838	0.881	0.799	0.983	0.958	0.605	0.353
(0.100,0.200]	2	19	0.863	0.751	0.832	0.712	1.000	1.000	0.355	0.600
(0.000,0.100]	0	27	1.000	0.614	0.761	0.614	1.000	1.000	0.000	0.956

Figure D-9: Credit-scoring evaluation results - Two-Class Decision Forest

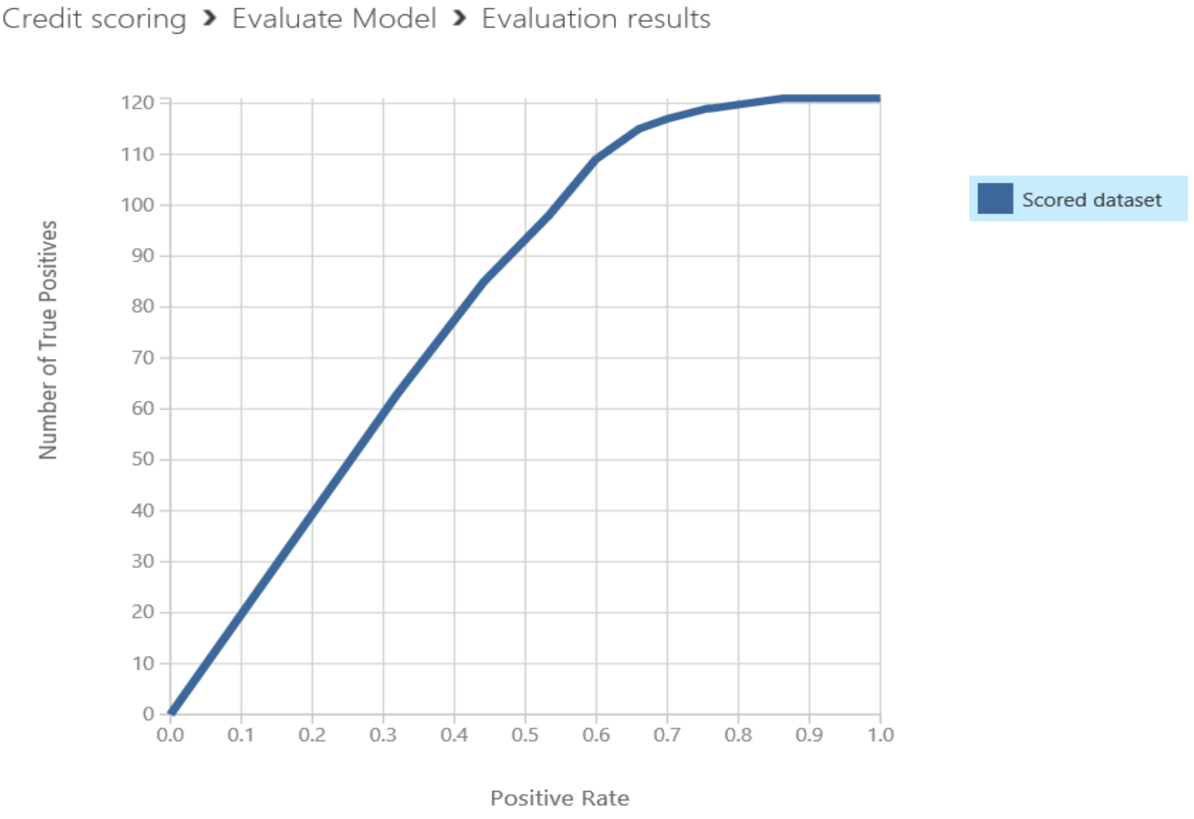

Credit scoring › Evaluate Model › Evaluation results

Figure D-10: Lift response chart for Two-Class Decision Forest (credit scoring)

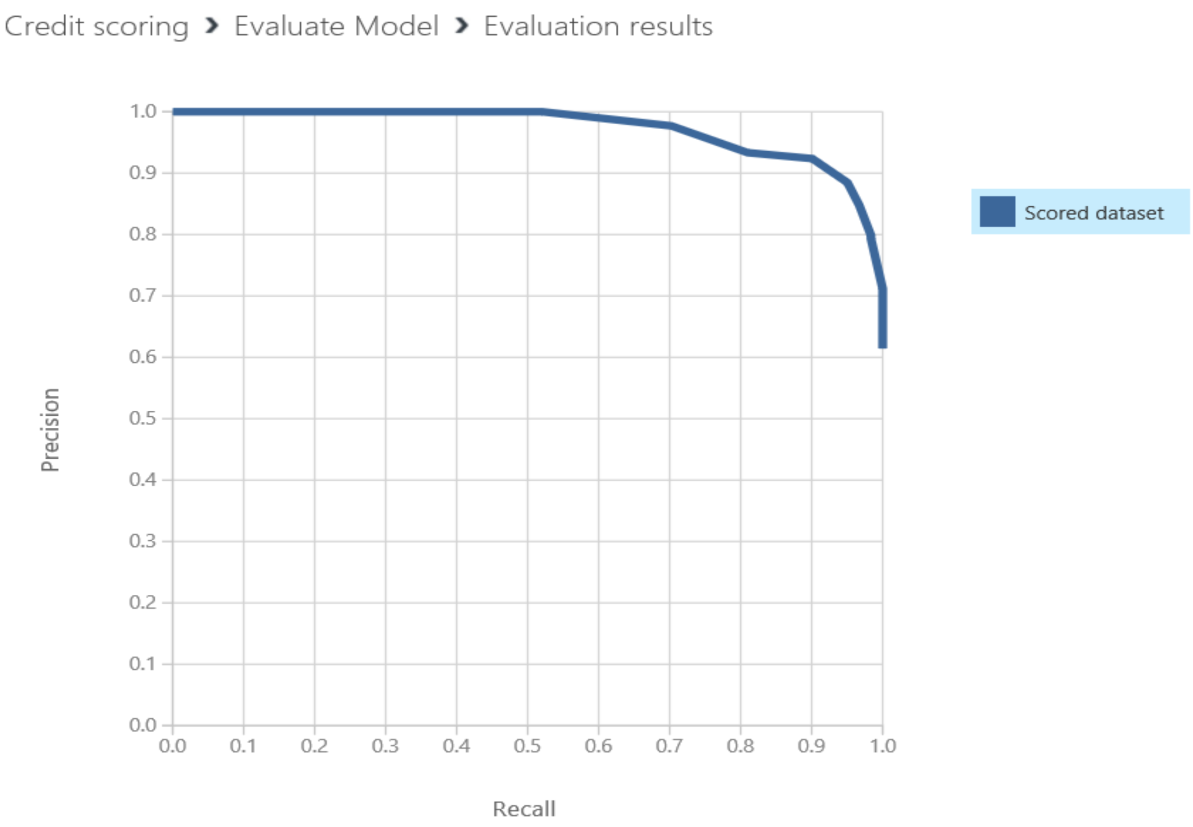
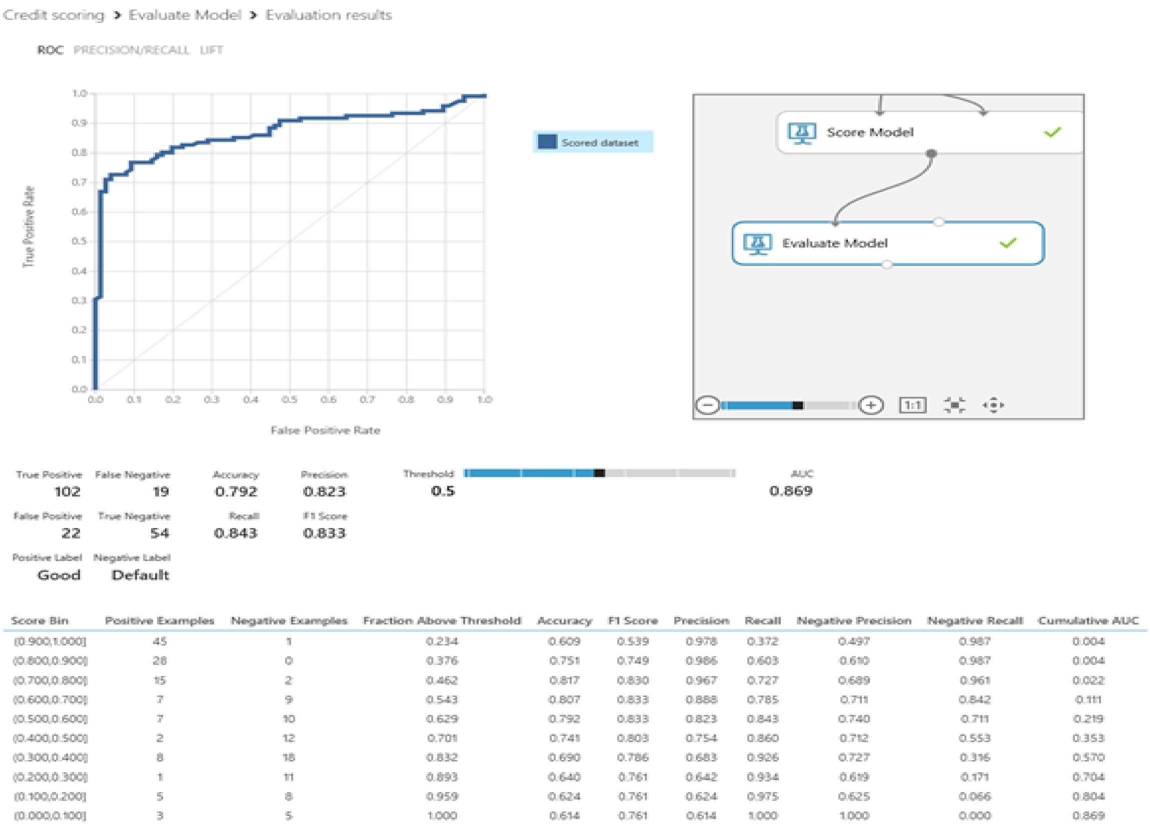

Credit scoring › Evaluate Model › Evaluation results

Figure D-11: Precision / Recall response chart for Two-Class Decision Forest (credit scoring)

D.2.5 Two-Class Support Vector Machine


Credit scoring › Evaluate Model › Evaluation results

True Positive 102
False Negative 19
Accuracy 0.792
Precision 0.823
Threshold 0.5
AUC 0.869
False Positive 22
True Negative 54
Recall 0.843
F1 Score 0.833
Positive Label Good
Negative Label Default

Score Bin	Positive Examples	Negative Examples	Fraction Above Threshold	Accuracy	F1 Score	Precision	Recall	Negative Precision	Negative Recall	Cumulative AUC
(0.900,1.000]	45	1	0.234	0.609	0.539	0.978	0.372	0.497	0.987	0.004
(0.800,0.900]	28	0	0.376	0.751	0.749	0.986	0.603	0.610	0.987	0.004
(0.700,0.800]	15	2	0.462	0.817	0.830	0.967	0.727	0.689	0.961	0.022
(0.600,0.700]	7	9	0.543	0.807	0.833	0.888	0.785	0.711	0.842	0.111
(0.500,0.600]	7	10	0.629	0.792	0.833	0.823	0.843	0.740	0.711	0.219
(0.400,0.500]	2	12	0.701	0.741	0.803	0.754	0.860	0.712	0.553	0.353
(0.300,0.400]	8	18	0.832	0.690	0.786	0.683	0.926	0.727	0.316	0.570
(0.200,0.300]	1	11	0.893	0.640	0.761	0.642	0.934	0.619	0.171	0.704
(0.100,0.200]	5	8	0.959	0.624	0.761	0.624	0.975	0.625	0.066	0.804
(0.000,0.100]	3	5	1.000	0.614	0.761	0.614	1.000	1.000	0.000	0.869

Figure D-12: Credit-scoring evaluation results - Two-Class Support Vector Machine

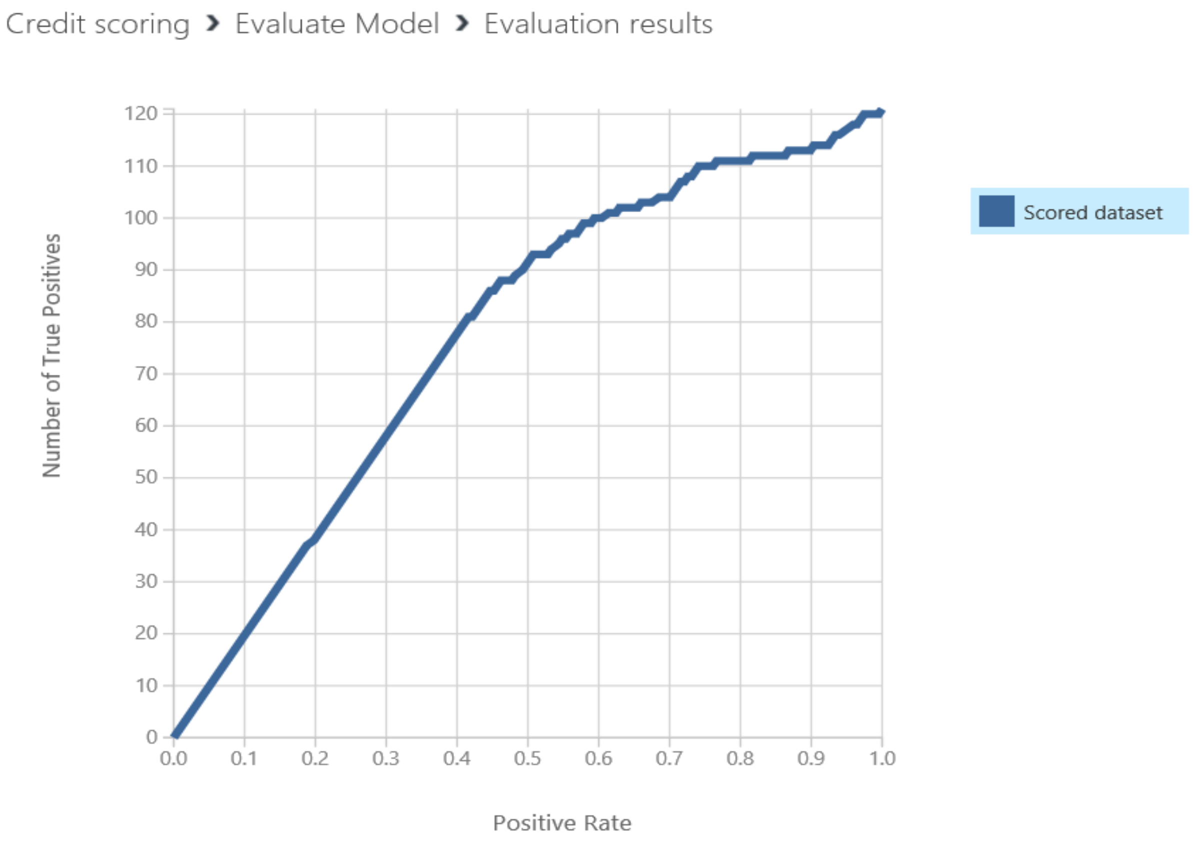


Figure D-13: Lift response chart for Two-Class Support Vector Machine (credit scoring)

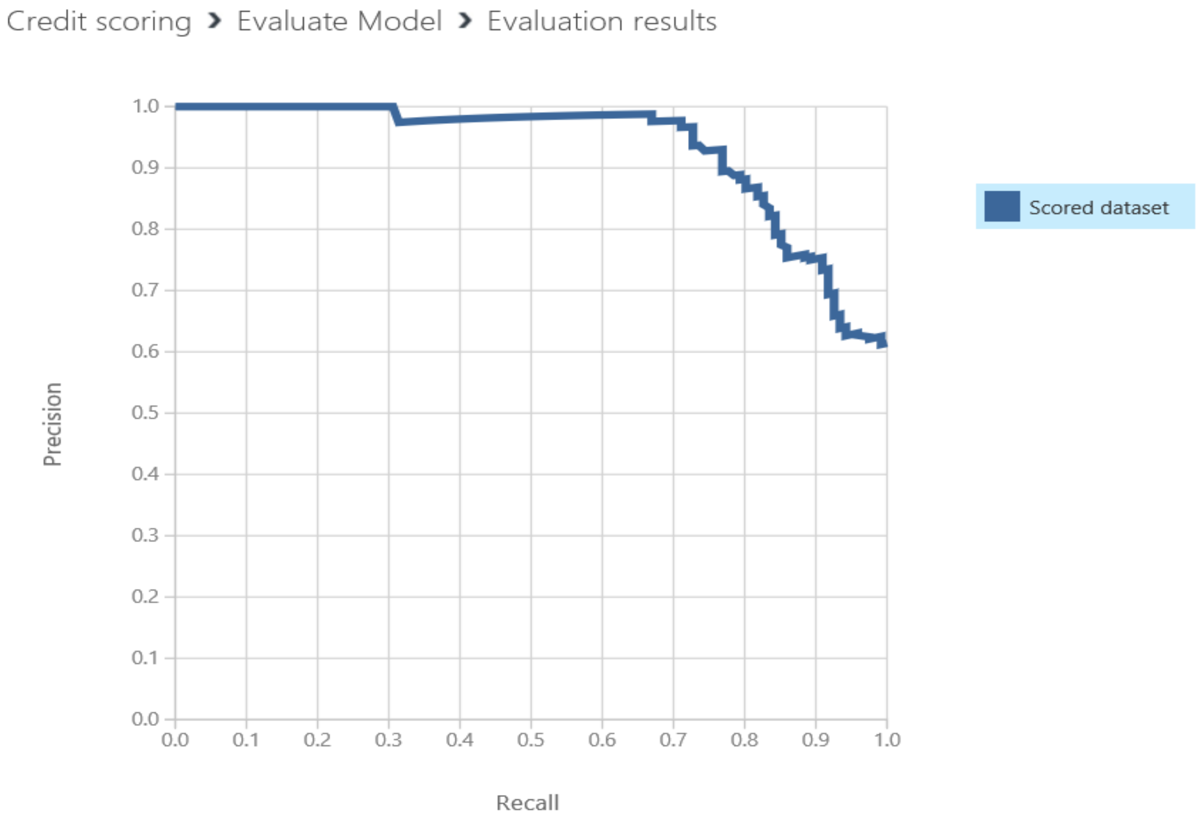


Figure D-14: Precision / Recall response chart for Two-Class Support Vector Machine (credit-scoring)

D.3. Cross-selling: Mini Map overview (creditcard_azure.csv)

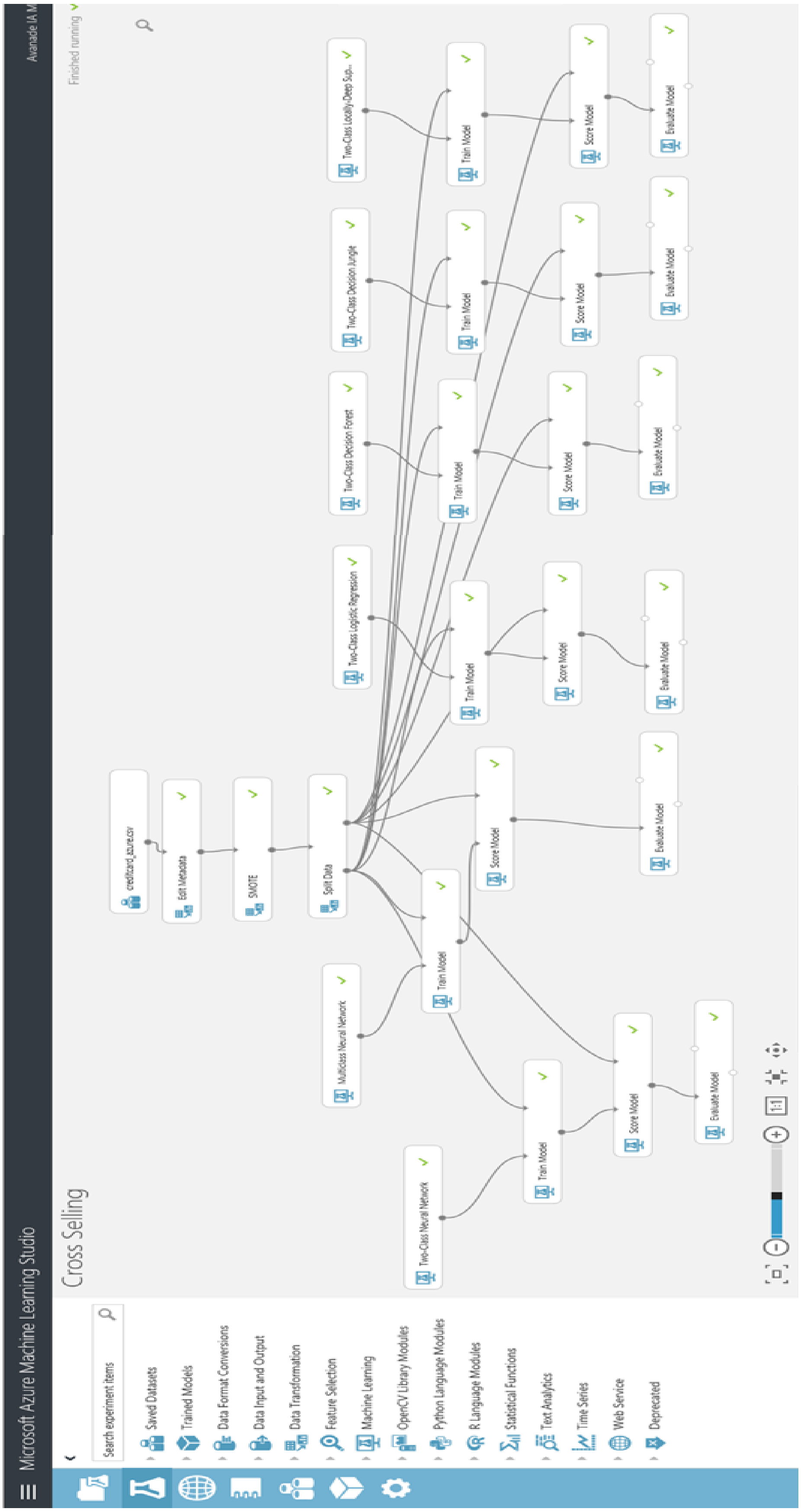
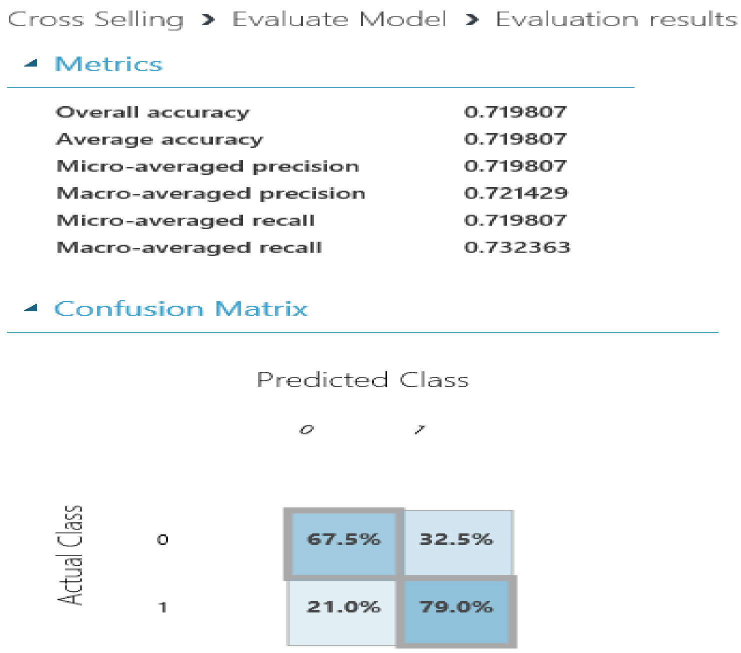


Figure D-15: Cross-selling - Mini Map overview

D.4. Evaluation results of supervised learning algorithms

D.4.1 Multiclass Neural Network



Cross Selling > Evaluate Model > Evaluation results

Metrics

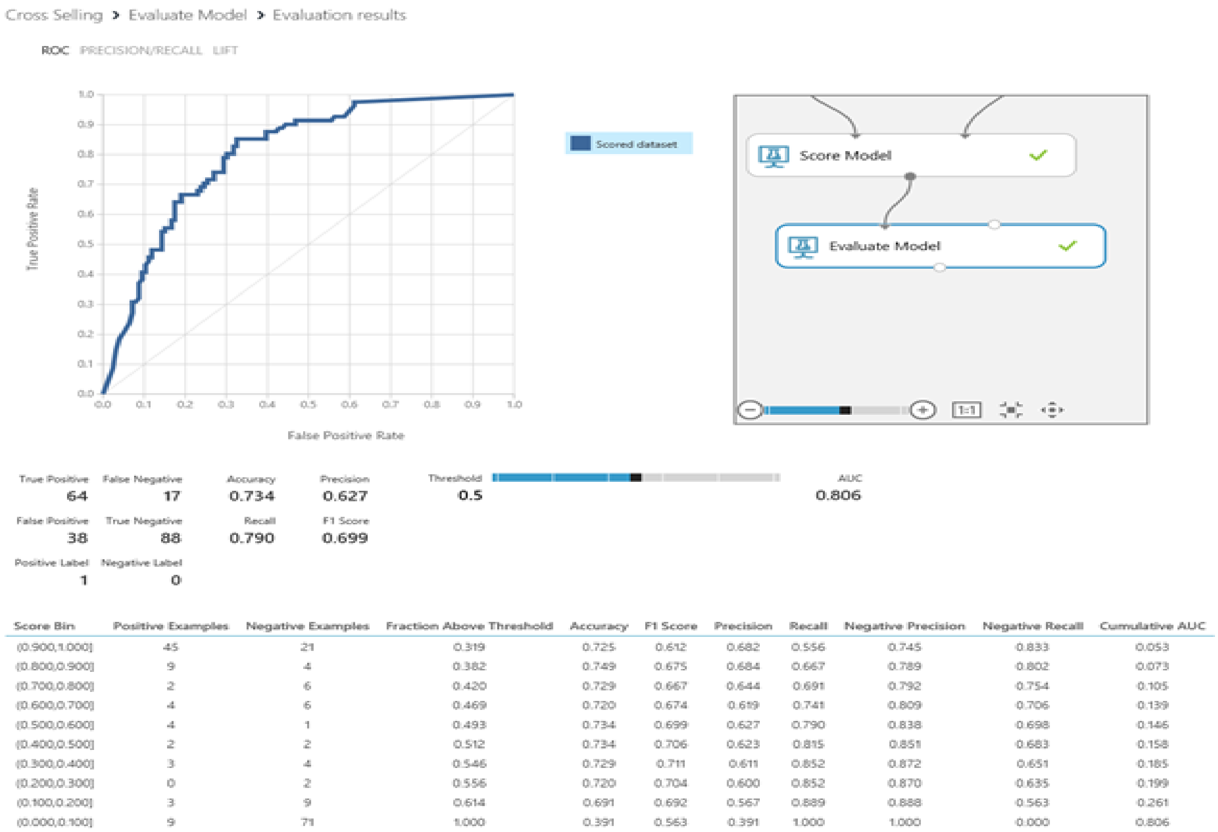
Overall accuracy	0.719807
Average accuracy	0.719807
Micro-averaged precision	0.719807
Macro-averaged precision	0.721429
Micro-averaged recall	0.719807
Macro-averaged recall	0.732363

Confusion Matrix

Actual Class \ Predicted Class	0	1
0	67.5%	32.5%
1	21.0%	79.0%

Figure D-16: Cross-selling evaluation results - Multiclass Neural Network

D.4.2 Two-Class Neural Network



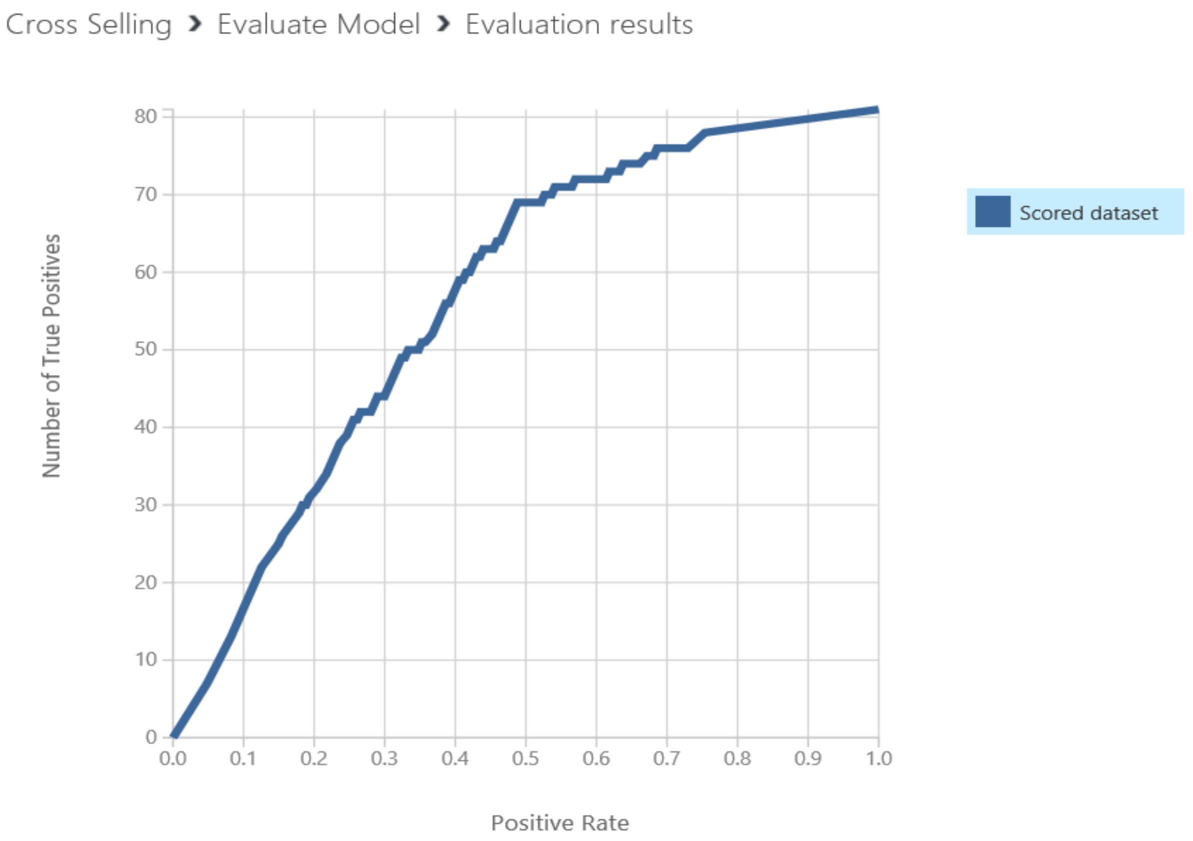
True Positive	False Negative	Accuracy	Precision	Threshold	AUC
64	17	0.734	0.627	0.5	0.806

False Positive	True Negative	Recall	F1 Score
38	88	0.790	0.699

Positive Label	Negative Label
1	0

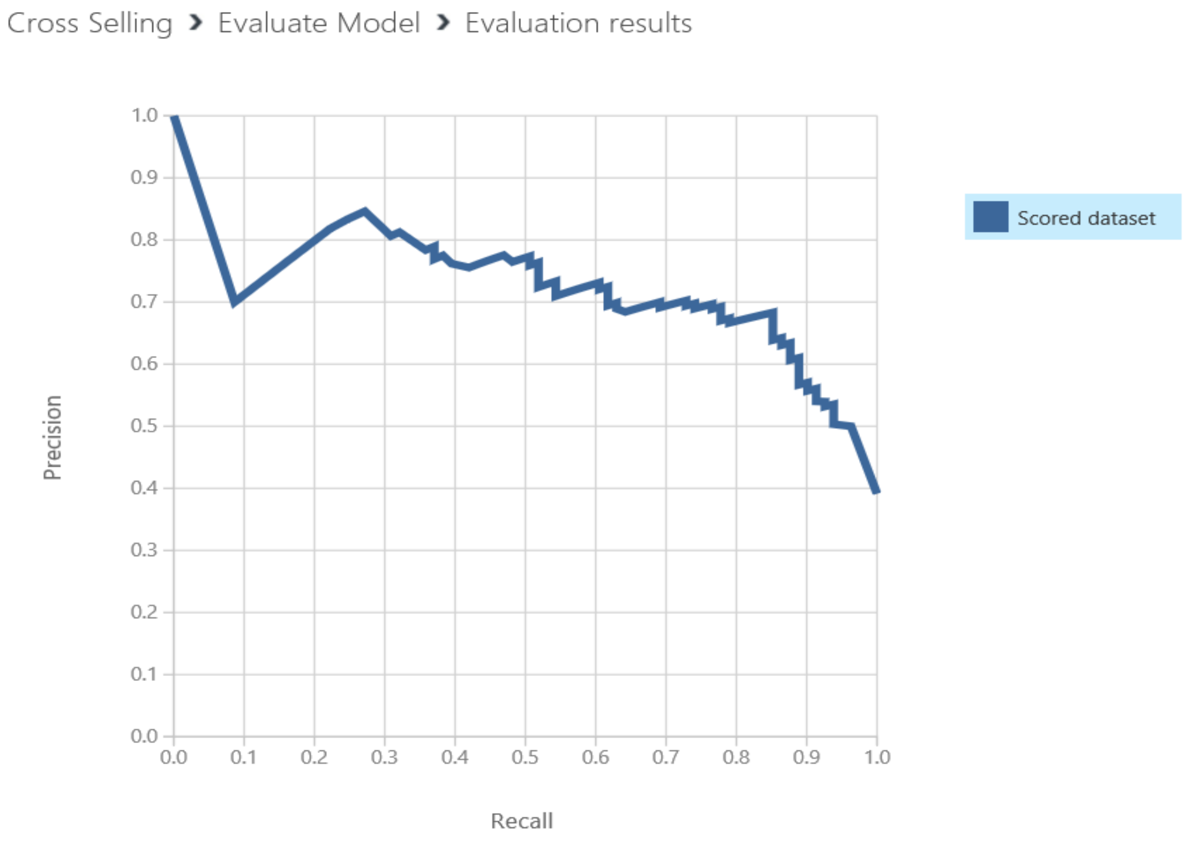
Score Bin	Positive Examples	Negative Examples	Fraction Above Threshold	Accuracy	F1 Score	Precision	Recall	Negative Precision	Negative Recall	Cumulative AUC
(0.900,1.000]	45	21	0.319	0.725	0.612	0.682	0.556	0.745	0.833	0.053
(0.800,0.900]	9	4	0.382	0.749	0.675	0.684	0.667	0.789	0.802	0.073
(0.700,0.800]	2	6	0.420	0.729	0.667	0.644	0.691	0.792	0.754	0.105
(0.600,0.700]	4	6	0.469	0.720	0.674	0.619	0.741	0.809	0.706	0.139
(0.500,0.600]	4	1	0.493	0.734	0.699	0.627	0.790	0.838	0.698	0.146
(0.400,0.500]	2	2	0.512	0.734	0.706	0.623	0.815	0.851	0.683	0.158
(0.300,0.400]	3	4	0.546	0.729	0.711	0.611	0.852	0.872	0.651	0.185
(0.200,0.300]	0	2	0.556	0.720	0.704	0.600	0.852	0.870	0.635	0.199
(0.100,0.200]	3	9	0.614	0.691	0.692	0.567	0.889	0.888	0.563	0.261
(0.000,0.100]	9	71	1.000	0.391	0.563	0.391	1.000	1.000	0.000	0.806

Figure D-17: Cross-selling evaluation results - Two-Class Neural Network



Cross Selling › Evaluate Model › Evaluation results

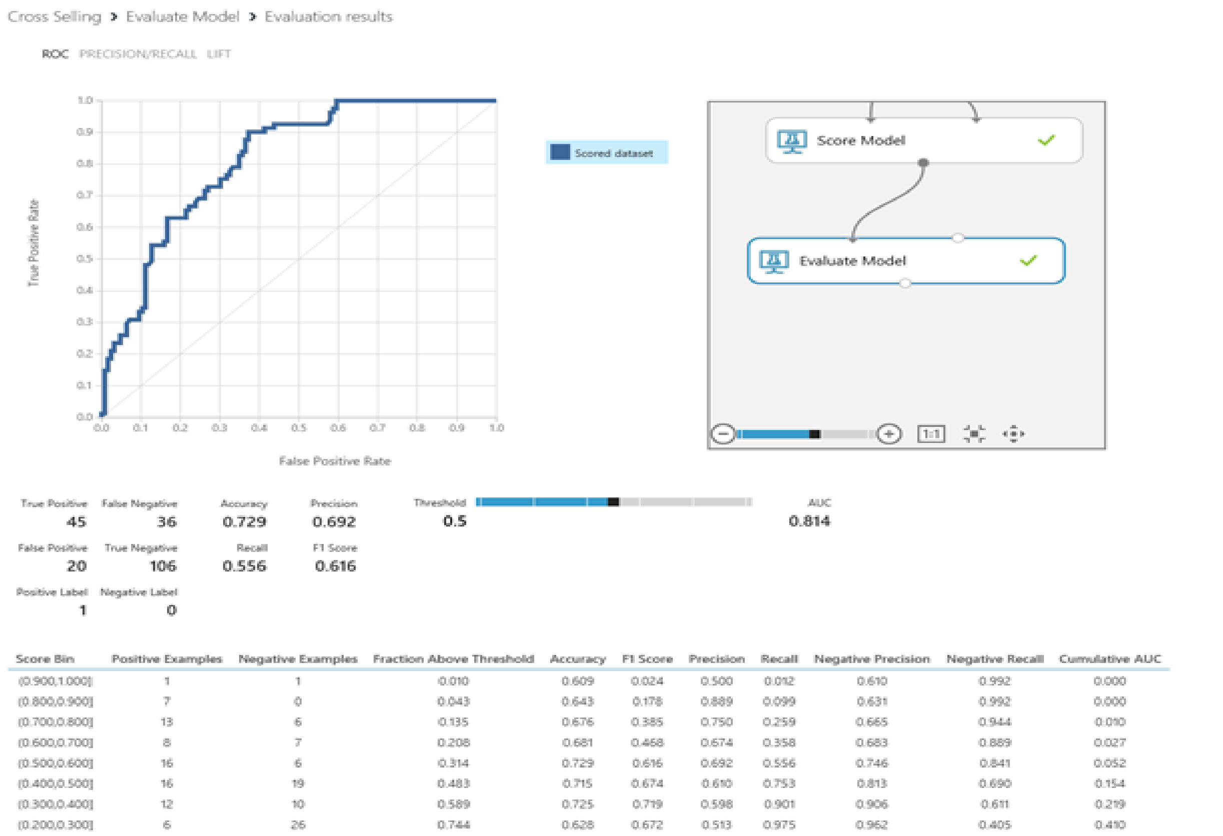
Figure D-18: Lift response chart for Two-Class Neural Network (cross-selling)



Cross Selling › Evaluate Model › Evaluation results

Figure D-19: Precision / Recall response chart for Two-Class Neural Network (cross-selling)

D.4.3 Two-Class Logistic Regression


Cross Selling › Evaluate Model › Evaluation results

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
45	36	0.729	0.692	0.5	0.814
False Positive	**True Negative**	**Recall**	**F1 Score**		
20	106	0.556	0.616		
Positive Label	**Negative Label**				
1	0				

Score Bin	Positive Examples	Negative Examples	Fraction Above Threshold	Accuracy	F1 Score	Precision	Recall	Negative Precision	Negative Recall	Cumulative AUC
(0.900,1.000]	1	1	0.010	0.609	0.024	0.500	0.012	0.610	0.992	0.000
(0.800,0.900]	7	0	0.043	0.643	0.178	0.889	0.099	0.631	0.992	0.000
(0.700,0.800]	13	6	0.135	0.676	0.385	0.750	0.259	0.665	0.944	0.010
(0.600,0.700]	8	7	0.208	0.681	0.468	0.674	0.358	0.683	0.889	0.027
(0.500,0.600]	16	6	0.314	0.729	0.616	0.692	0.556	0.746	0.841	0.052
(0.400,0.500]	16	19	0.483	0.715	0.674	0.610	0.753	0.813	0.690	0.154
(0.300,0.400]	12	10	0.589	0.725	0.719	0.598	0.901	0.906	0.611	0.219
(0.200,0.300]	6	26	0.744	0.628	0.672	0.513	0.975	0.962	0.405	0.410

Figure D-20: Cross-selling evaluation results – Two-Class Logistic Regression

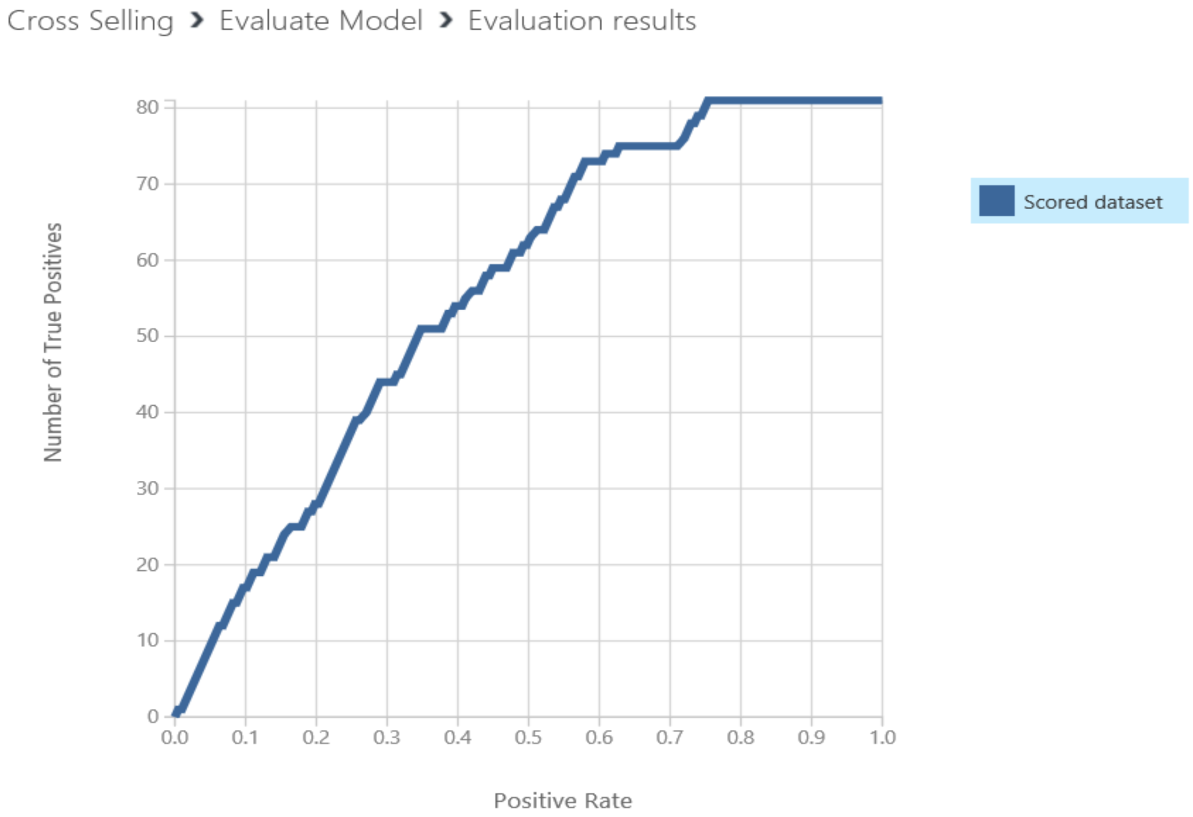
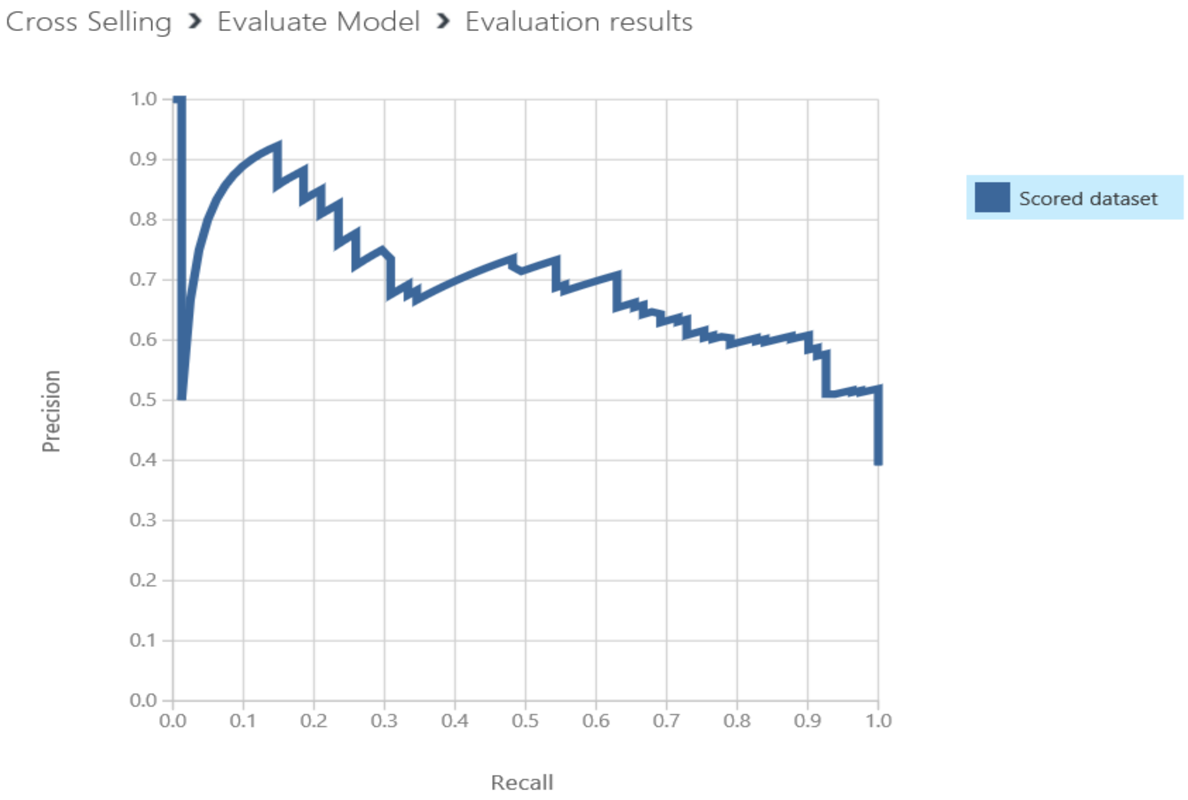

Cross Selling › Evaluate Model › Evaluation results

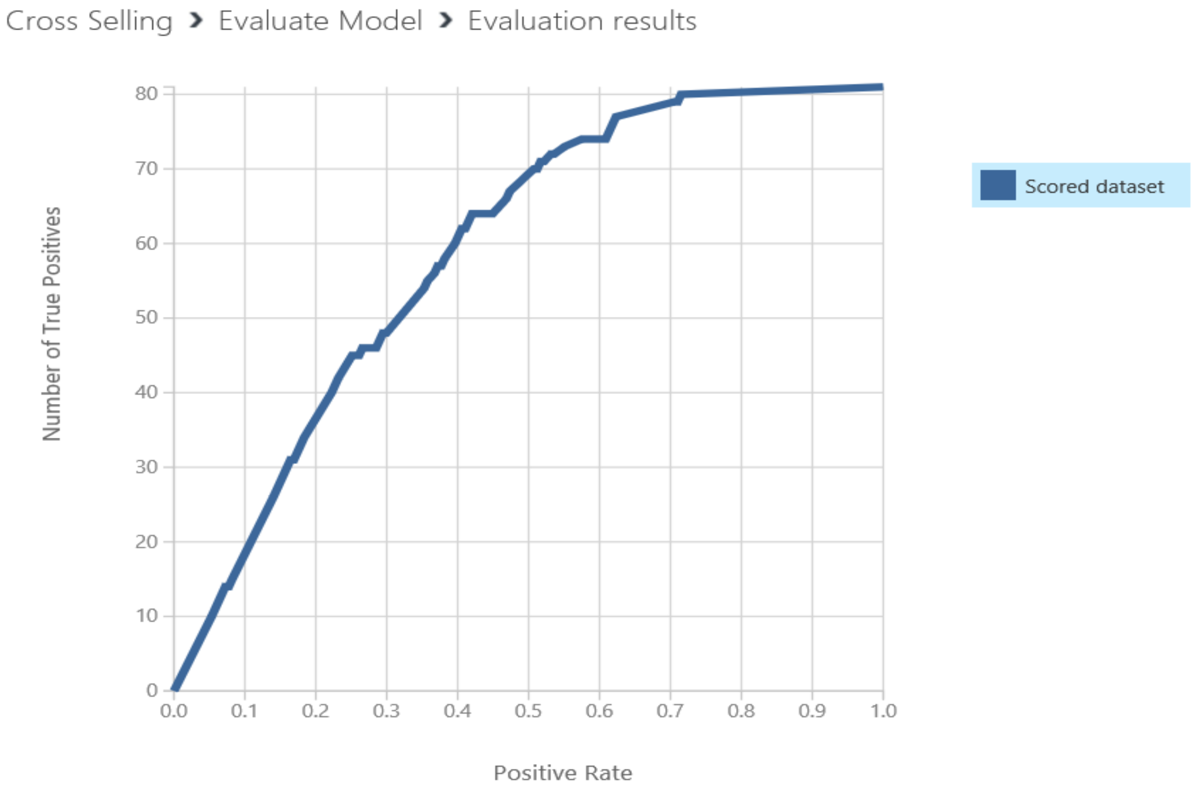
Figure D-21: Lift response chart for Two-Class Logistic Regression (cross-selling)



Cross Selling › Evaluate Model › Evaluation results

Figure D-22: Precision / Recall response chart for Two-Class Logistic Regression (cross-selling)

D.4.4 Two-Class Decision Forest



Cross Selling › Evaluate Model › Evaluation results

Figure D-23: Cross-selling evaluation results - Two-Class Decision Forest

Cross Selling › Evaluate Model › Evaluation results

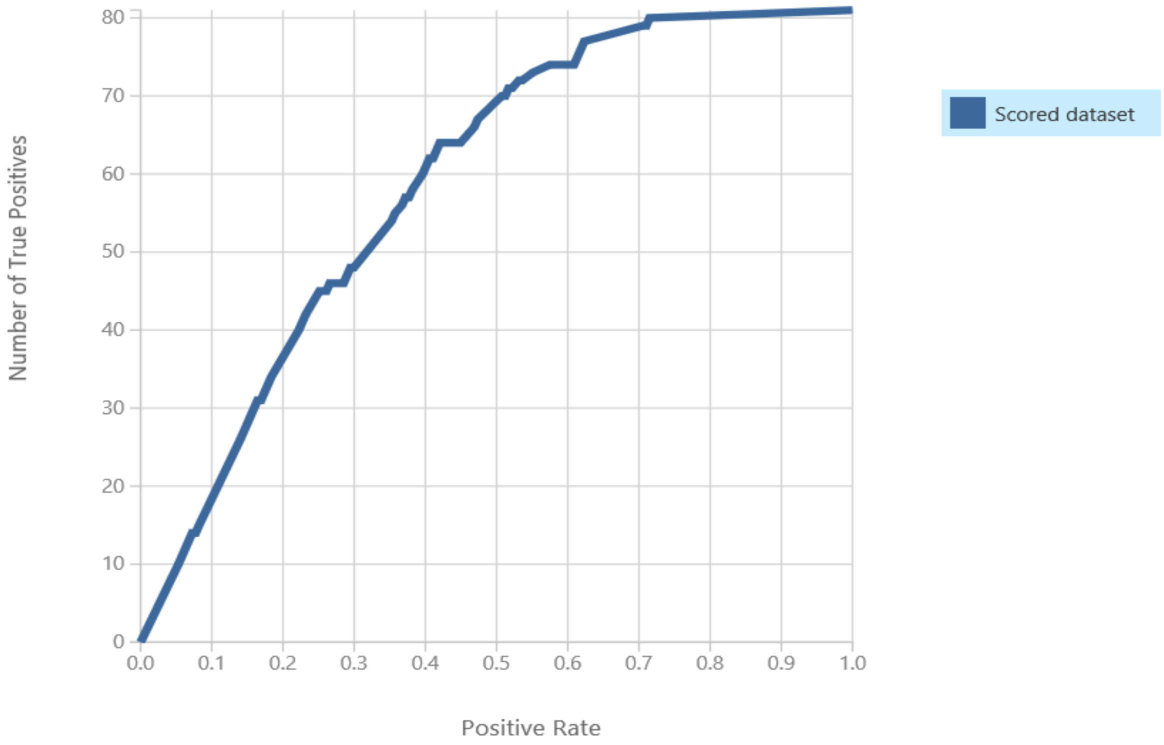


Figure D-24: Lift response chart for Two-Class Decision Forest (cross-selling)

Cross Selling › Evaluate Model › Evaluation results

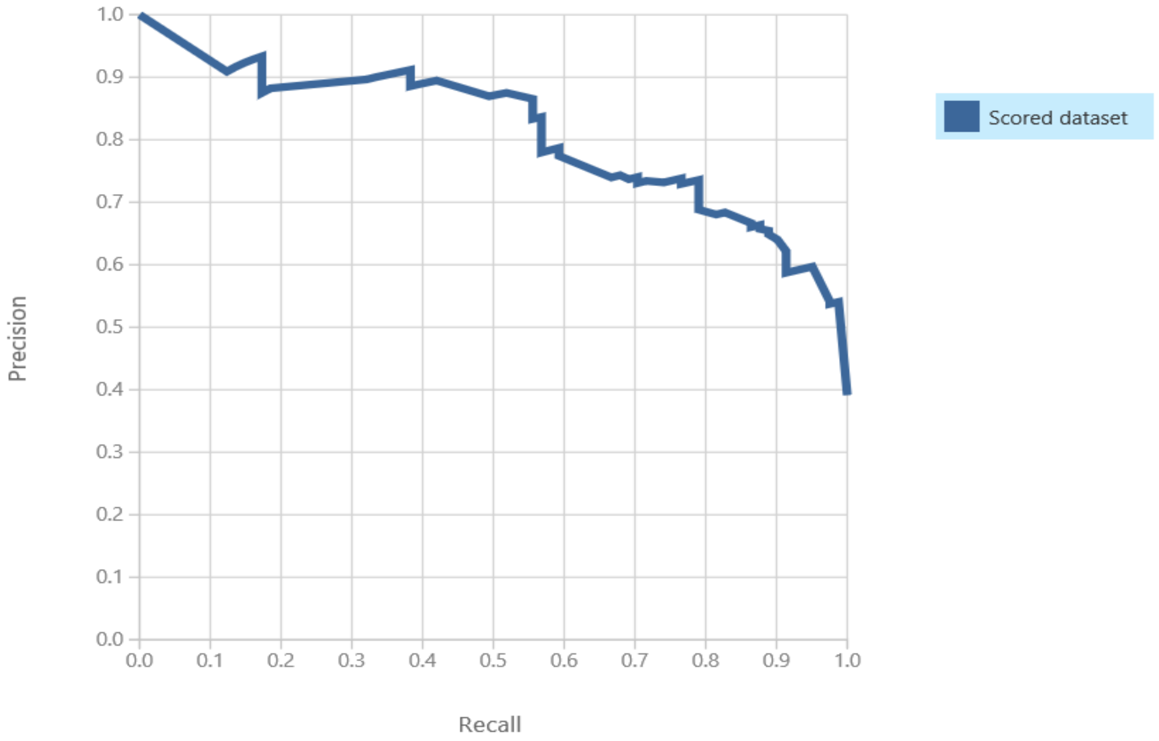
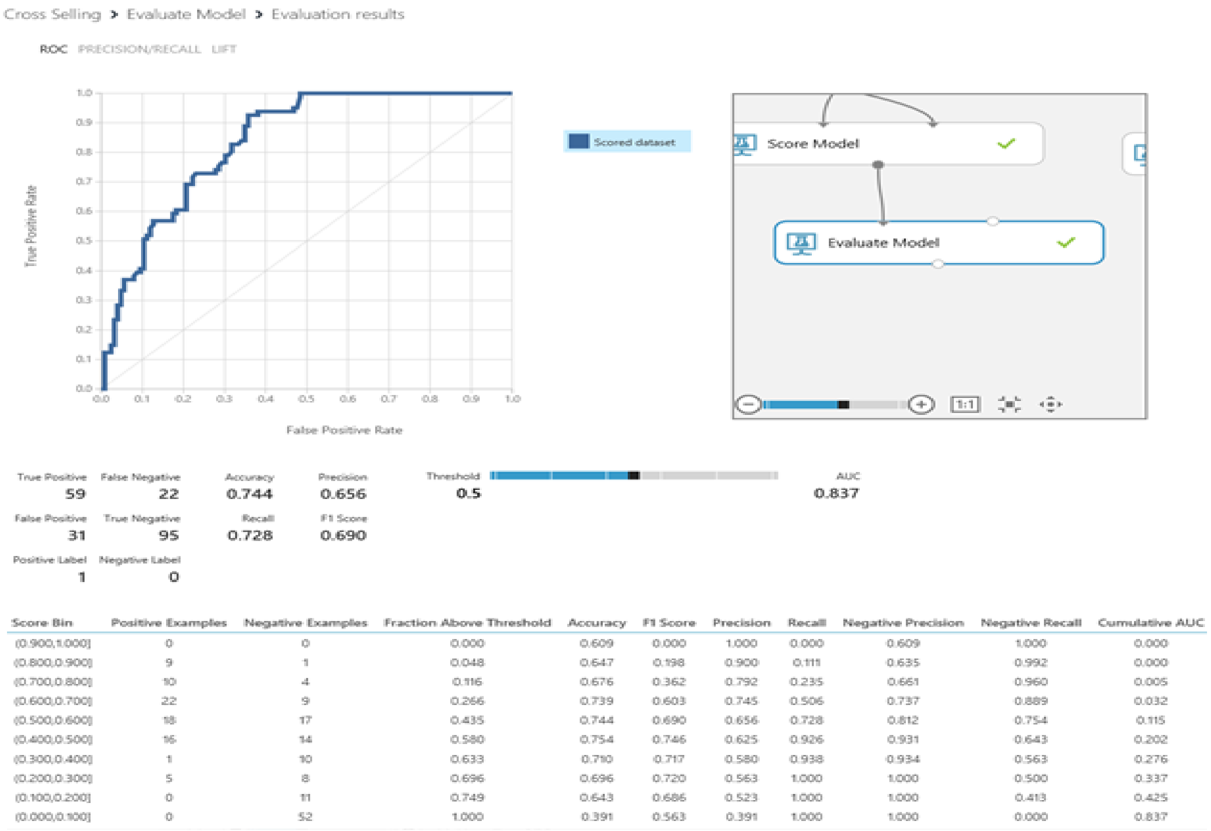


Figure D-25: Precision / Recall response chart for Two-Class Decision Forest (cross-selling)

D.4.5 Two-Class Decision Jungle



True Positive	False Negative	Accuracy	Precision	Threshold	AUC
59	22	0.744	0.656	0.5	0.837
False Positive	**True Negative**	**Recall**	**F1 Score**		
31	95	0.728	0.690		
Positive Label	**Negative Label**				
1	0				

Score Bin	Positive Examples	Negative Examples	Fraction Above Threshold	Accuracy	F1 Score	Precision	Recall	Negative Precision	Negative Recall	Cumulative AUC
(0.900,1.000]	0	0	0.000	0.609	0.000	1.000	0.000	0.609	1.000	0.000
(0.800,0.900]	9	1	0.048	0.647	0.198	0.900	0.111	0.635	0.992	0.000
(0.700,0.800]	10	4	0.116	0.676	0.362	0.792	0.235	0.661	0.960	0.005
(0.600,0.700]	22	9	0.266	0.739	0.603	0.745	0.506	0.737	0.889	0.032
(0.500,0.600]	18	17	0.435	0.744	0.690	0.656	0.728	0.812	0.754	0.115
(0.400,0.500]	16	14	0.580	0.754	0.746	0.625	0.926	0.931	0.643	0.202
(0.300,0.400]	1	10	0.633	0.710	0.717	0.580	0.938	0.934	0.563	0.276
(0.200,0.300]	5	8	0.696	0.696	0.720	0.563	1.000	1.000	0.500	0.337
(0.100,0.200]	0	11	0.749	0.643	0.686	0.523	1.000	1.000	0.413	0.425
(0.000,0.100]	0	52	1.000	0.391	0.563	0.391	1.000	1.000	0.000	0.837

Figure D-26: Cross-selling evaluation results - Two-Class Decision Jungle

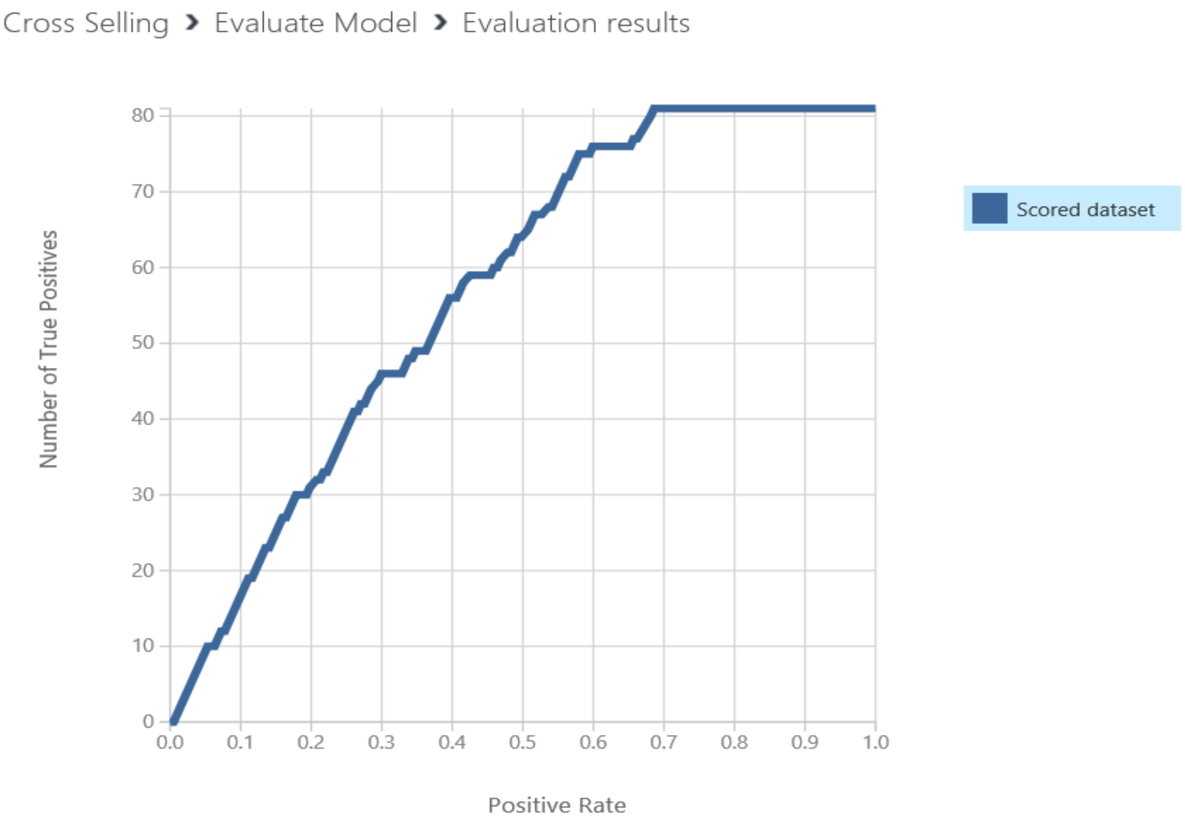


Figure D-27: Lift response chart for Two-Class Decision Jungle (cross-selling)

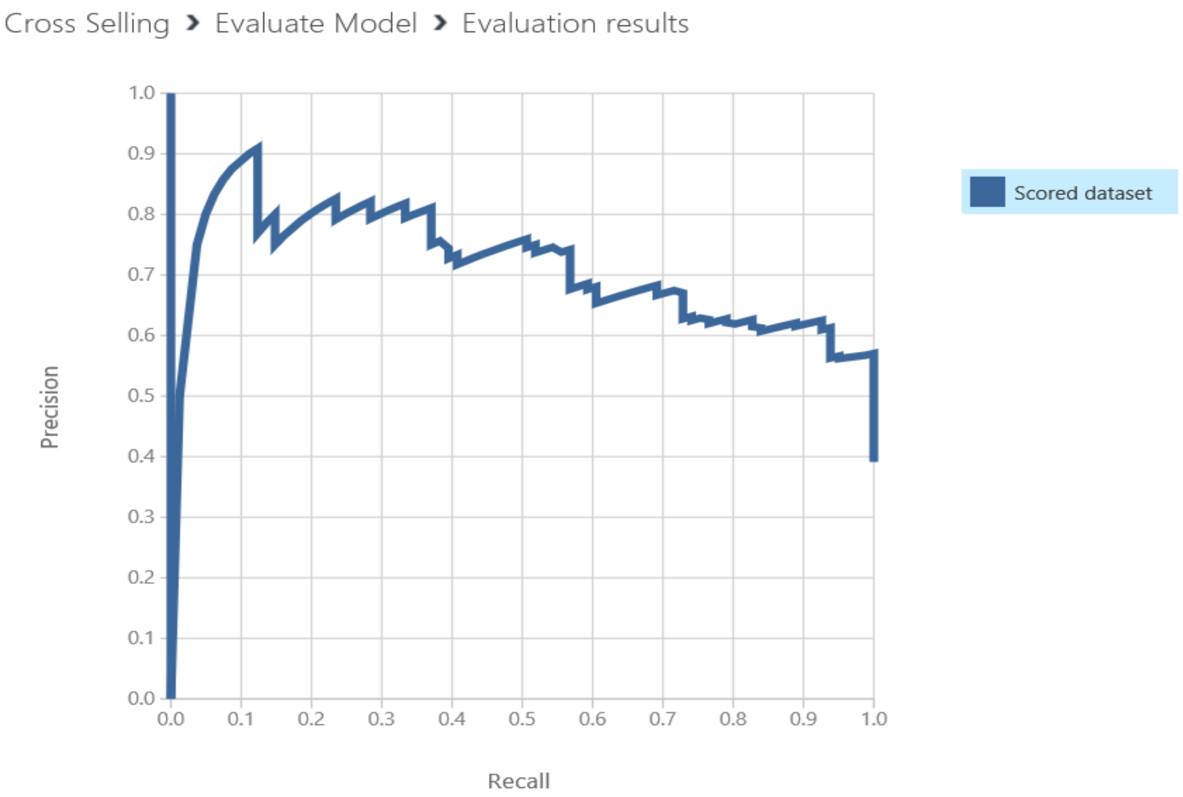
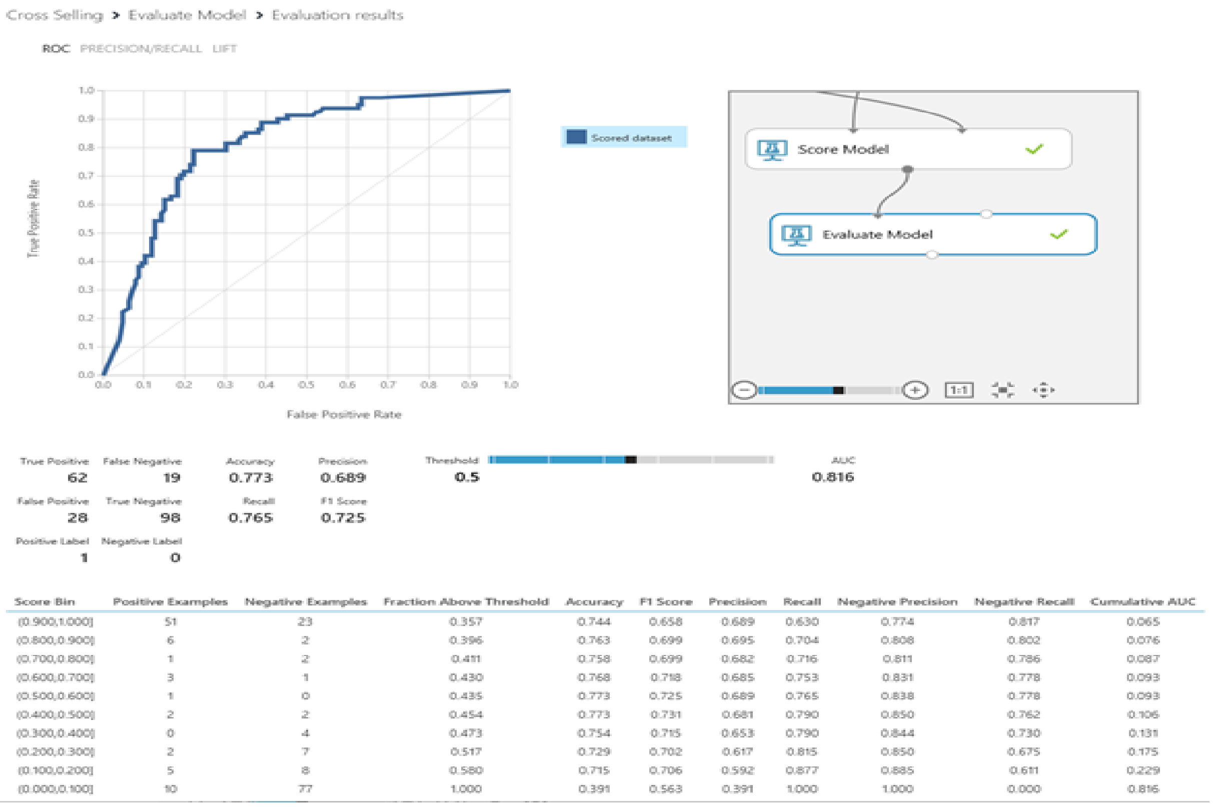


Figure D-28: Precision / Recall response chart for Two-Class Decision Jungle (cross-selling)

D.4.6 Two-Class Locally-Deep Support Vector Machine



True Positive	False Negative	Accuracy	Precision	Threshold	AUC
62	19	0.773	0.689	0.5	0.816
False Positive	True Negative	Recall	F1 Score		
28	98	0.765	0.725		
Positive Label	Negative Label				
1	0				

Score Bin	Positive Examples	Negative Examples	Fraction Above Threshold	Accuracy	F1 Score	Precision	Recall	Negative Precision	Negative Recall	Cumulative AUC
(0.900,1.000]	51	23	0.357	0.744	0.658	0.689	0.630	0.774	0.817	0.065
(0.800,0.900]	6	2	0.396	0.763	0.699	0.695	0.704	0.808	0.802	0.076
(0.700,0.800]	1	2	0.411	0.758	0.699	0.682	0.716	0.811	0.786	0.087
(0.600,0.700]	3	1	0.430	0.768	0.718	0.685	0.753	0.831	0.778	0.093
(0.500,0.600]	1	0	0.435	0.773	0.725	0.689	0.765	0.838	0.778	0.093
(0.400,0.500]	2	2	0.454	0.773	0.731	0.681	0.790	0.850	0.762	0.106
(0.300,0.400]	0	4	0.473	0.754	0.715	0.653	0.790	0.844	0.730	0.131
(0.200,0.300]	2	7	0.517	0.729	0.702	0.617	0.815	0.850	0.675	0.175
(0.100,0.200]	5	8	0.580	0.715	0.706	0.592	0.877	0.885	0.611	0.229
[0.000,0.100]	10	77	1.000	0.391	0.563	0.391	1.000	1.000	0.000	0.816

Figure D-29: Cross-selling evaluation results - Two-Class Locally-Deep Support Vector Machine

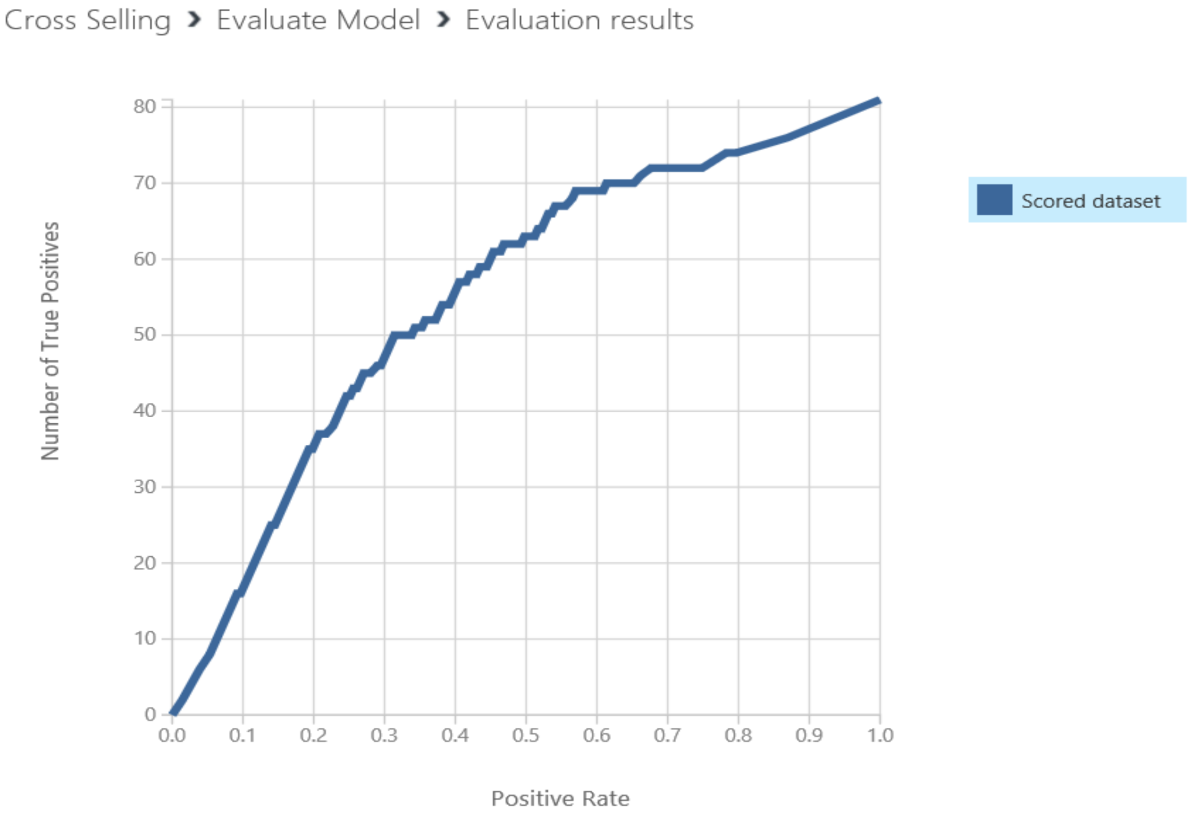


Figure D-30: Lift response chart for Two-Class Locally-Deep Support Vector Machine (cross-selling)

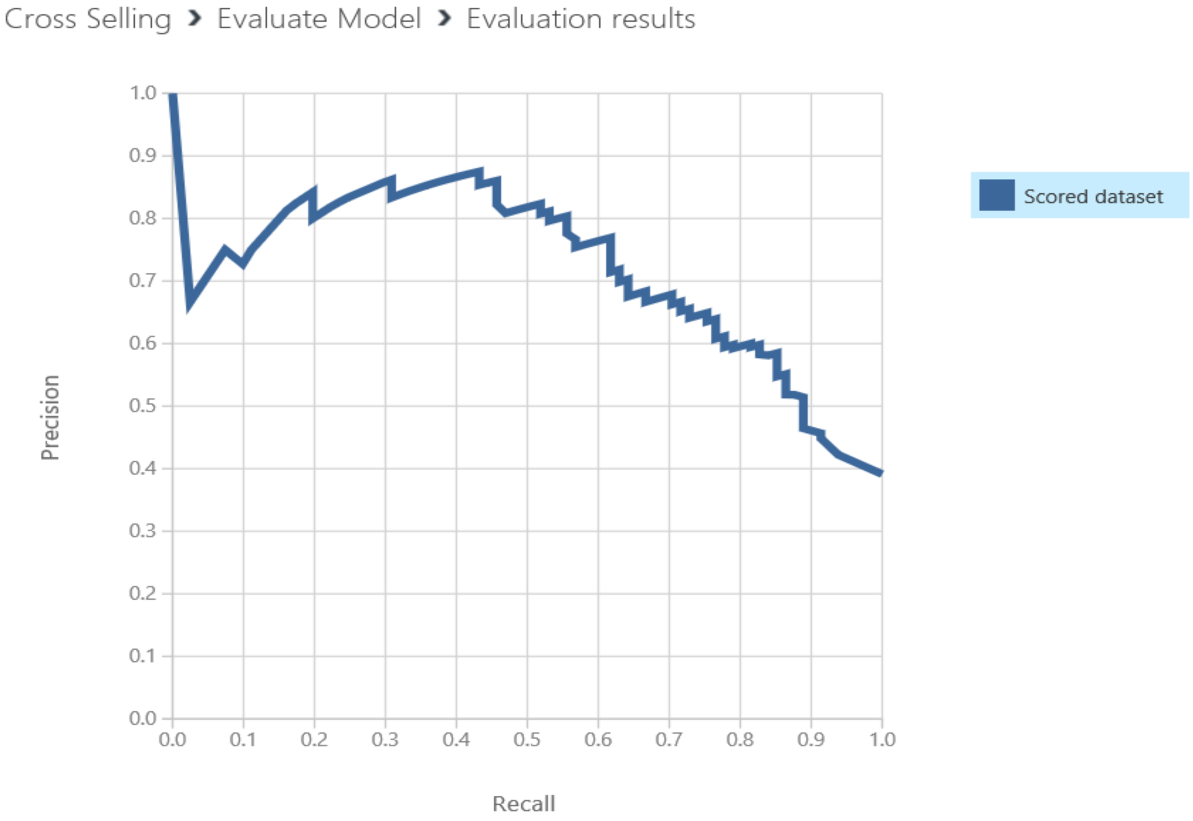
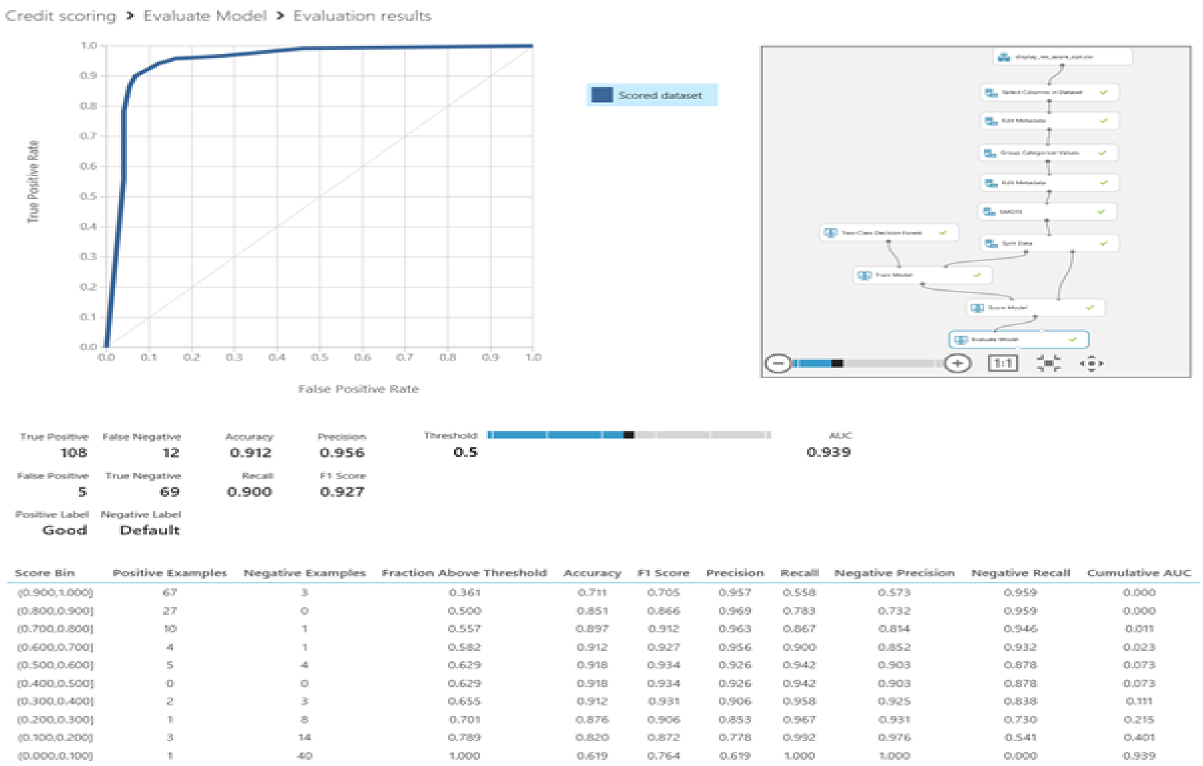


Figure D-31: Precision / Recall response chart for Two-Class Locally-Deep Support Vector Machine (cross-selling)

D.5. Evaluation results of optimized supervised learning algorithm

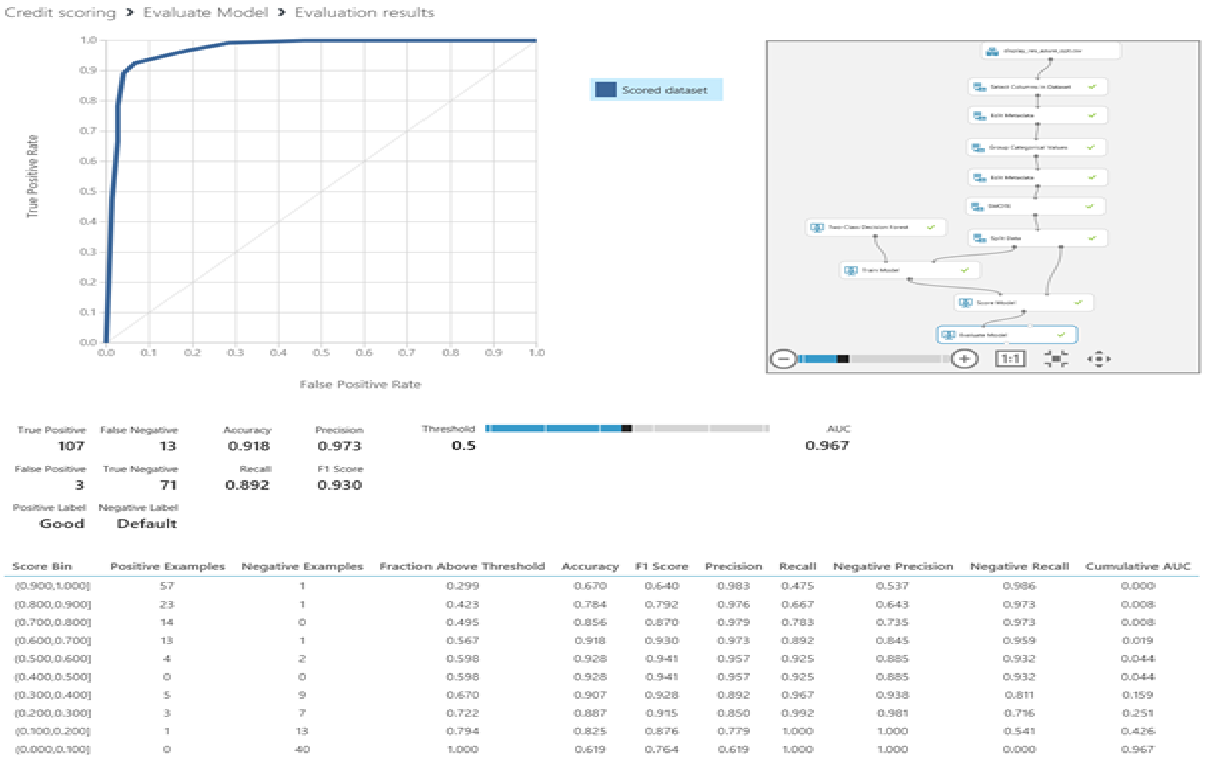
D.5.1 Two-Class Decision Forest - Credit-scoring



True Positive 108	False Negative 12	Accuracy 0.912	Precision 0.956	Threshold 0.5	AUC 0.939
False Positive 5	True Negative 69	Recall 0.900	F1 Score 0.927		
Positive Label Good	Negative Label Default				

Score Bin	Positive Examples	Negative Examples	Fraction Above Threshold	Accuracy	F1 Score	Precision	Recall	Negative Precision	Negative Recall	Cumulative AUC
(0.900,1.000]	67	3	0.361	0.711	0.705	0.957	0.558	0.573	0.959	0.000
(0.800,0.900]	27	0	0.500	0.851	0.866	0.969	0.783	0.732	0.959	0.000
(0.700,0.800]	10	1	0.557	0.897	0.912	0.963	0.867	0.814	0.946	0.011
(0.600,0.700]	4	1	0.582	0.912	0.927	0.956	0.900	0.852	0.932	0.023
(0.500,0.600]	5	4	0.629	0.918	0.934	0.926	0.942	0.903	0.878	0.073
(0.400,0.500]	0	0	0.629	0.918	0.934	0.926	0.942	0.903	0.878	0.073
(0.300,0.400]	2	3	0.655	0.912	0.931	0.906	0.958	0.925	0.838	0.111
(0.200,0.300]	1	8	0.701	0.876	0.906	0.853	0.967	0.931	0.730	0.215
(0.100,0.200]	3	14	0.789	0.820	0.872	0.778	0.992	0.976	0.541	0.401
(0.000,0.100]	1	40	1.000	0.619	0.764	0.619	1.000	1.000	0.000	0.939

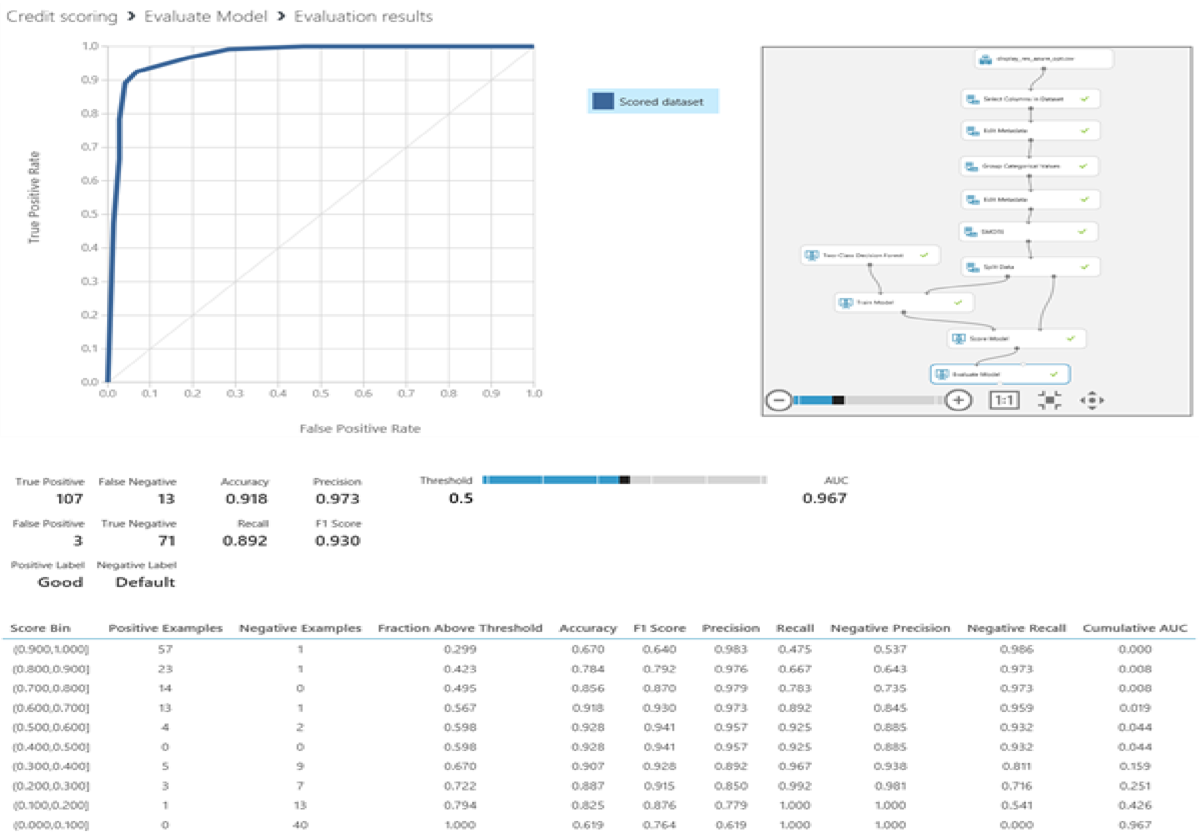
Figure D-32: Credit-scoring evaluation results using Two-Class Decision Forest algorithm based on the optimized dataset (excl. cardholder and sex)



True Positive 107	False Negative 13	Accuracy 0.918	Precision 0.973	Threshold 0.5	AUC 0.967
False Positive 3	True Negative 71	Recall 0.892	F1 Score 0.930		
Positive Label Good	Negative Label Default				

Score Bin	Positive Examples	Negative Examples	Fraction Above Threshold	Accuracy	F1 Score	Precision	Recall	Negative Precision	Negative Recall	Cumulative AUC
(0.900,1.000]	57	1	0.299	0.670	0.640	0.983	0.475	0.537	0.986	0.000
(0.800,0.900]	23	1	0.423	0.784	0.792	0.976	0.667	0.643	0.973	0.008
(0.700,0.800]	14	0	0.495	0.856	0.870	0.979	0.783	0.735	0.973	0.008
(0.600,0.700]	13	1	0.567	0.918	0.930	0.973	0.892	0.845	0.959	0.019
(0.500,0.600]	4	2	0.598	0.928	0.941	0.957	0.925	0.885	0.932	0.044
(0.400,0.500]	0	0	0.598	0.928	0.941	0.957	0.925	0.885	0.932	0.044
(0.300,0.400]	5	9	0.670	0.907	0.928	0.892	0.967	0.938	0.811	0.159
(0.200,0.300]	3	7	0.722	0.887	0.915	0.850	0.992	0.981	0.716	0.251
(0.100,0.200]	1	13	0.794	0.825	0.876	0.779	1.000	1.000	0.541	0.426
(0.000,0.100]	0	40	1.000	0.619	0.764	0.619	1.000	1.000	0.000	0.967

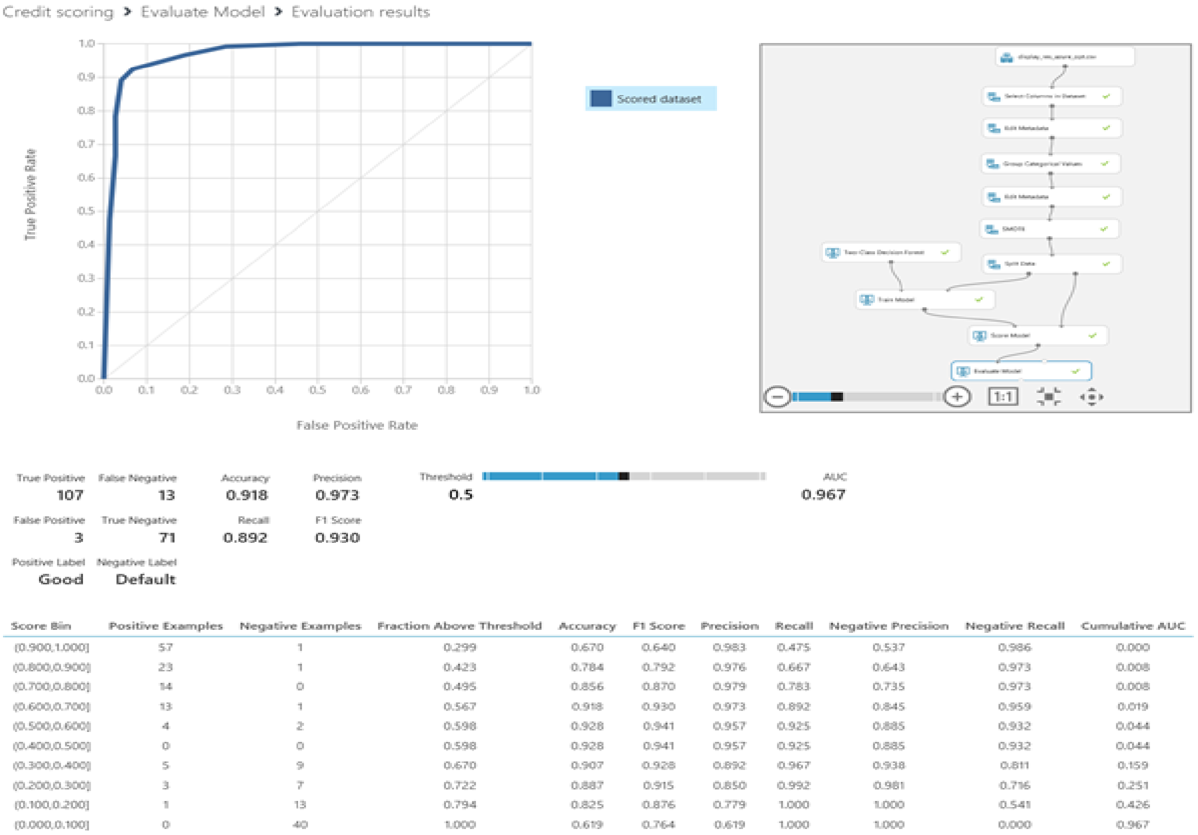
Figure D-33: Credit-scoring evaluation results using Two-Class Decision Forest algorithm based on the optimized dataset (incl. AvgBalance, AvgIncome, AvgExpenses, Age)



True Positive	False Negative	Accuracy	Precision	Threshold	AUC
107	13	0.918	0.973	0.5	0.967
False Positive	**True Negative**	**Recall**	**F1 Score**		
3	71	0.892	0.930		
Positive Label	**Negative Label**				
Good	Default				

Score Bin	Positive Examples	Negative Examples	Fraction Above Threshold	Accuracy	F1 Score	Precision	Recall	Negative Precision	Negative Recall	Cumulative AUC
(0.900,1.000]	57	1	0.299	0.670	0.640	0.983	0.475	0.537	0.986	0.000
(0.800,0.900]	23	1	0.423	0.784	0.792	0.976	0.667	0.643	0.973	0.008
(0.700,0.800]	14	0	0.495	0.856	0.870	0.979	0.783	0.735	0.973	0.008
(0.600,0.700]	13	1	0.567	0.918	0.930	0.973	0.892	0.845	0.959	0.019
(0.500,0.600]	4	2	0.598	0.928	0.941	0.957	0.925	0.885	0.932	0.044
(0.400,0.500]	0	0	0.598	0.928	0.941	0.957	0.925	0.885	0.932	0.044
(0.300,0.400]	5	9	0.670	0.907	0.928	0.892	0.967	0.938	0.811	0.159
(0.200,0.300]	3	7	0.722	0.887	0.915	0.850	0.992	0.981	0.716	0.251
(0.100,0.200]	1	13	0.794	0.825	0.876	0.779	1.000	1.000	0.541	0.426
(0.000,0.100]	0	40	1.000	0.619	0.764	0.619	1.000	1.000	0.000	0.967

Figure D-34: Credit-scoring evaluation results using Two-Class Decision Forest algorithm based on the optimized dataset (incl. AvgBalance, AvgExpenses, Age)

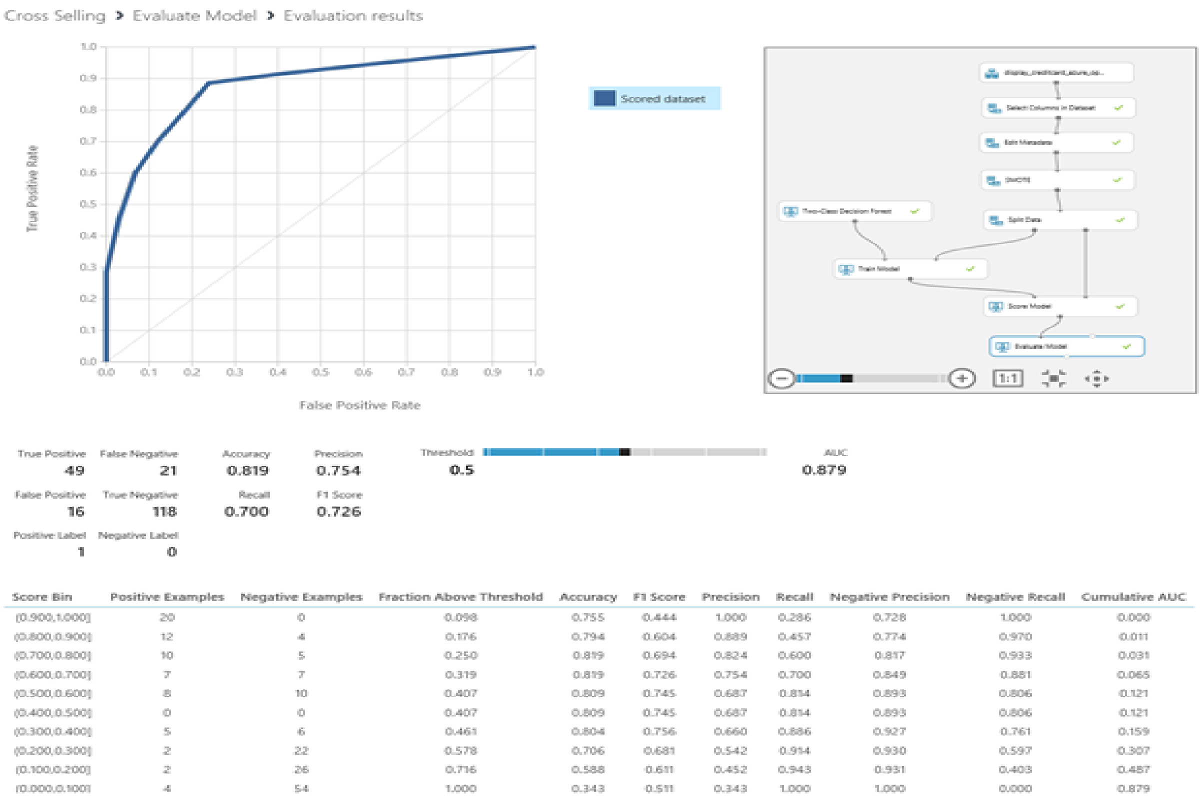


True Positive	False Negative	Accuracy	Precision	Threshold	AUC
107	13	0.918	0.973	0.5	0.967
False Positive	**True Negative**	**Recall**	**F1 Score**		
3	71	0.892	0.930		
Positive Label	**Negative Label**				
Good	Default				

Score Bin	Positive Examples	Negative Examples	Fraction Above Threshold	Accuracy	F1 Score	Precision	Recall	Negative Precision	Negative Recall	Cumulative AUC
(0.900,1.000]	57	1	0.299	0.670	0.640	0.983	0.475	0.537	0.986	0.000
(0.800,0.900]	23	1	0.423	0.784	0.792	0.976	0.667	0.643	0.973	0.008
(0.700,0.800]	14	0	0.495	0.856	0.870	0.979	0.783	0.735	0.973	0.008
(0.600,0.700]	13	1	0.567	0.918	0.930	0.973	0.892	0.845	0.959	0.019
(0.500,0.600]	4	2	0.598	0.928	0.941	0.957	0.925	0.885	0.932	0.044
(0.400,0.500]	0	0	0.598	0.928	0.941	0.957	0.925	0.885	0.932	0.044
(0.300,0.400]	5	9	0.670	0.907	0.928	0.892	0.967	0.938	0.811	0.159
(0.200,0.300]	3	7	0.722	0.887	0.915	0.850	0.992	0.981	0.716	0.251
(0.100,0.200]	1	13	0.794	0.825	0.876	0.779	1.000	1.000	0.541	0.426
(0.000,0.100]	0	40	1.000	0.619	0.764	0.619	1.000	1.000	0.000	0.967

Figure D-35: Credit-scoring evaluation results using Two-Class Decision Forest algorithm based on the optimized dataset (incl. AvgBalance, AvgIncome, Age)

D.5.2 Two-Class Decision Forest - Cross-selling



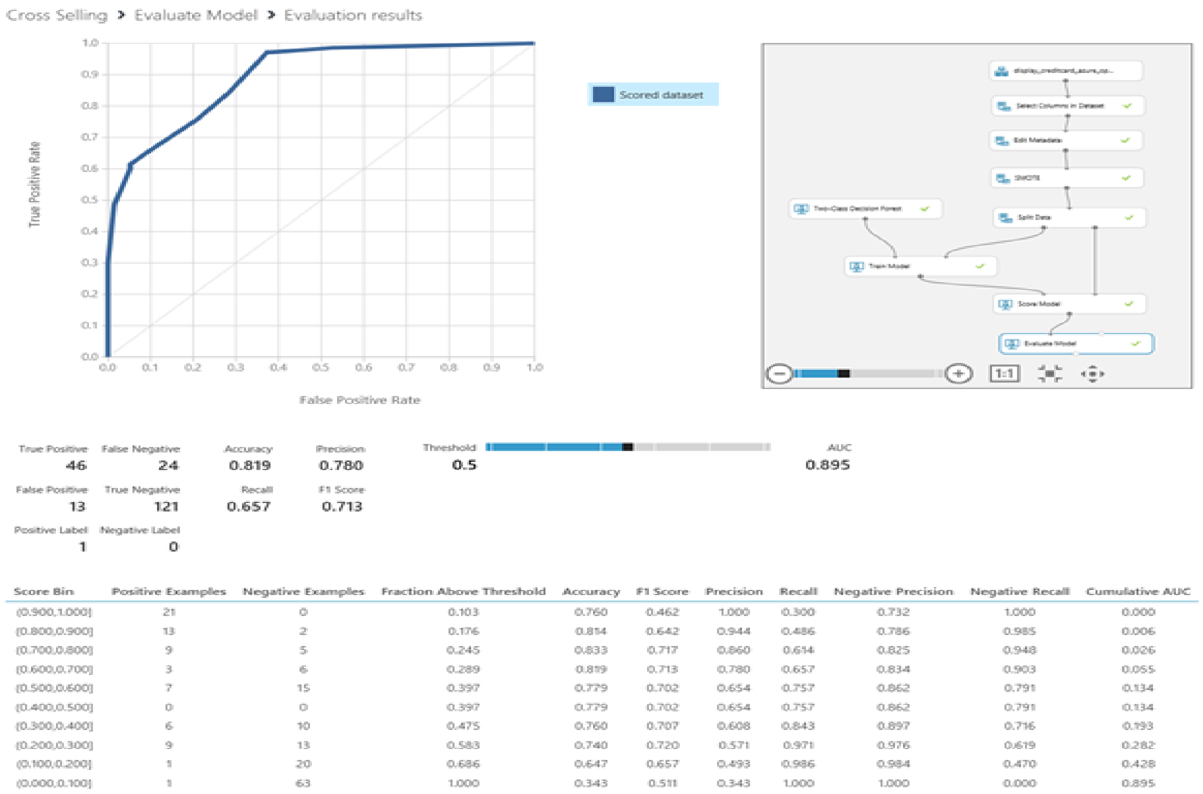
True Positive	False Negative	Accuracy	Precision	Threshold	AUC
49	21	0.819	0.754	0.5	0.879

False Positive	True Negative	Recall	F1 Score
16	118	0.700	0.726

Positive Label	Negative Label
1	0

Score Bin	Positive Examples	Negative Examples	Fraction Above Threshold	Accuracy	F1 Score	Precision	Recall	Negative Precision	Negative Recall	Cumulative AUC
(0.900,1.000]	20	0	0.098	0.755	0.444	1.000	0.286	0.728	1.000	0.000
(0.800,0.900]	12	4	0.176	0.794	0.604	0.889	0.457	0.774	0.970	0.011
(0.700,0.800]	10	5	0.250	0.819	0.694	0.824	0.600	0.817	0.933	0.031
(0.600,0.700]	7	7	0.319	0.819	0.726	0.754	0.700	0.849	0.881	0.065
(0.500,0.600]	8	10	0.407	0.809	0.745	0.687	0.814	0.893	0.806	0.121
(0.400,0.500]	0	0	0.407	0.809	0.745	0.687	0.814	0.893	0.806	0.121
(0.300,0.400]	5	6	0.461	0.804	0.756	0.660	0.886	0.927	0.761	0.159
(0.200,0.300]	2	22	0.578	0.706	0.681	0.542	0.914	0.930	0.597	0.307
(0.100,0.200]	2	26	0.716	0.588	0.611	0.452	0.943	0.931	0.403	0.487
(0.000,0.100]	4	54	1.000	0.343	0.511	0.343	1.000	1.000	0.000	0.879

Figure D-36: Cross-selling evaluation results using Two-Class Decision Forest algorithm based on the optimized dataset (excl. frequency and sex)



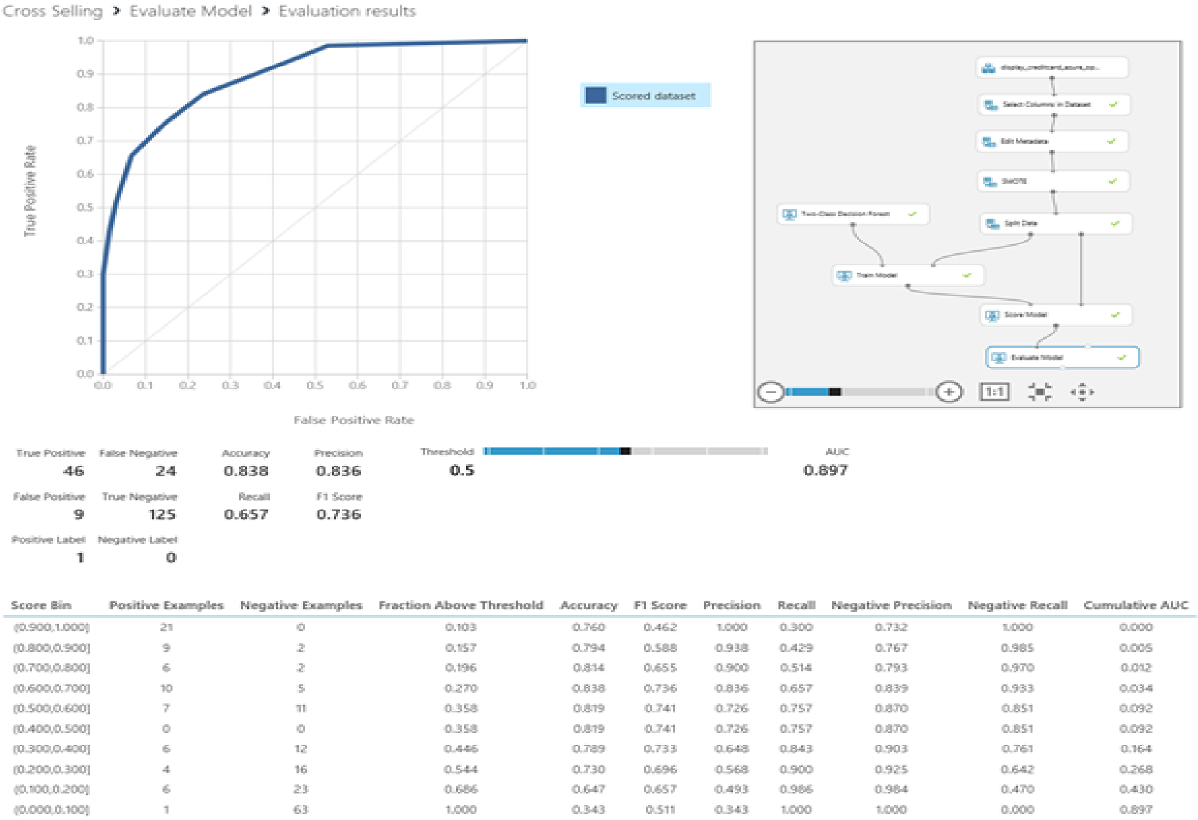
True Positive	False Negative	Accuracy	Precision	Threshold	AUC
46	24	0.819	0.780	0.5	0.895

False Positive	True Negative	Recall	F1 Score
13	121	0.657	0.713

Positive Label	Negative Label
1	0

Score Bin	Positive Examples	Negative Examples	Fraction Above Threshold	Accuracy	F1 Score	Precision	Recall	Negative Precision	Negative Recall	Cumulative AUC
(0.900,1.000]	21	0	0.103	0.760	0.462	1.000	0.300	0.732	1.000	0.000
(0.800,0.900]	13	2	0.176	0.814	0.642	0.944	0.486	0.786	0.985	0.006
(0.700,0.800]	9	5	0.245	0.833	0.717	0.860	0.614	0.825	0.948	0.026
(0.600,0.700]	3	6	0.289	0.819	0.713	0.780	0.657	0.834	0.903	0.055
(0.500,0.600]	7	15	0.397	0.779	0.702	0.654	0.757	0.862	0.791	0.134
(0.400,0.500]	0	0	0.397	0.779	0.702	0.654	0.757	0.862	0.791	0.134
(0.300,0.400]	6	10	0.475	0.760	0.707	0.608	0.843	0.897	0.716	0.193
(0.200,0.300]	9	13	0.583	0.740	0.720	0.571	0.971	0.976	0.619	0.282
(0.100,0.200]	1	20	0.686	0.647	0.657	0.493	0.986	0.984	0.470	0.428
(0.000,0.100]	1	63	1.000	0.343	0.511	0.343	1.000	1.000	0.000	0.895

Figure D-37: Cross-selling evaluation results using Two-Class Decision Forest algorithm based on the optimized dataset (incl. Age, AvgIncome, AvgExpenses, AvgBalance, AvgSalary)

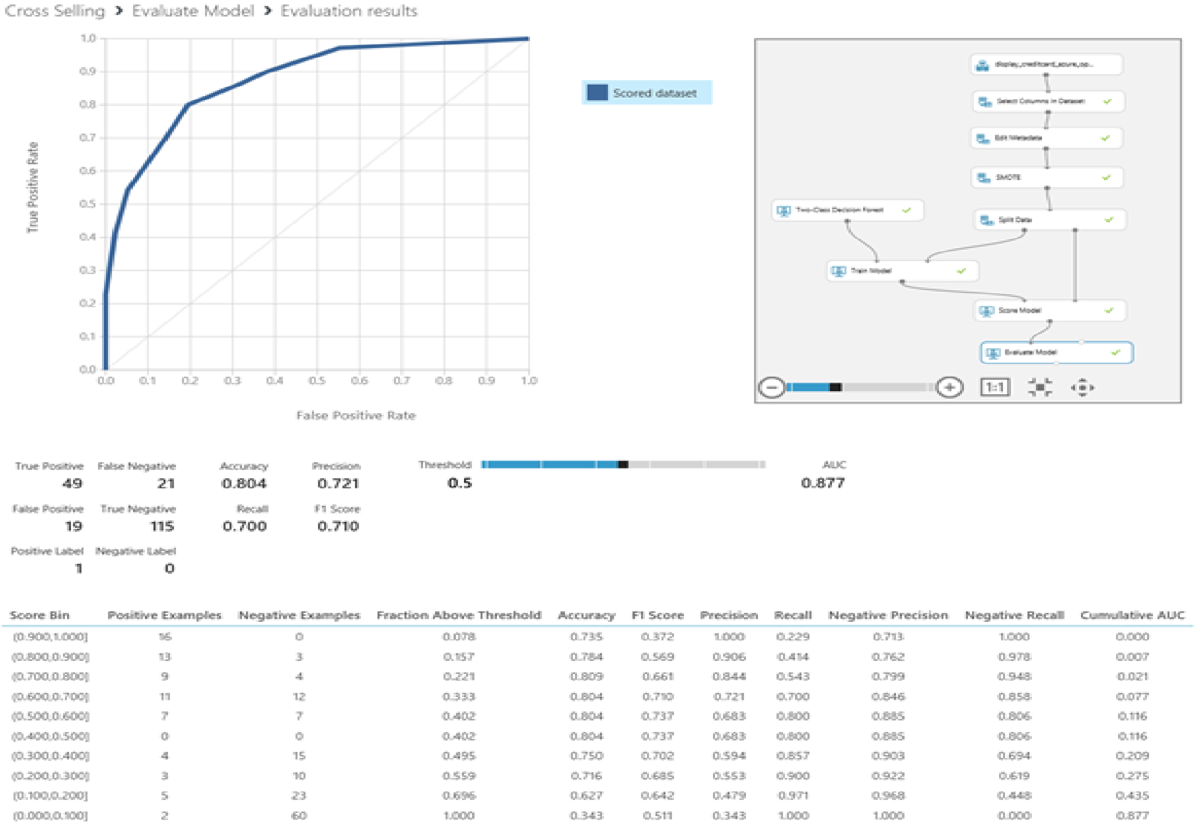


Cross Selling > Evaluate Model > Evaluation results

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
46	24	0.838	0.836	0.5	0.897
False Positive	**True Negative**	**Recall**	**F1 Score**		
9	125	0.657	0.736		
Positive Label	**Negative Label**				
1	0				

Score Bin	Positive Examples	Negative Examples	Fraction Above Threshold	Accuracy	F1 Score	Precision	Recall	Negative Precision	Negative Recall	Cumulative AUC
(0.900,1.000]	21	0	0.103	0.760	0.462	1.000	0.300	0.732	1.000	0.000
(0.800,0.900]	9	2	0.157	0.794	0.588	0.938	0.429	0.767	0.985	0.005
(0.700,0.800]	6	2	0.196	0.814	0.655	0.900	0.514	0.793	0.970	0.012
(0.600,0.700]	10	5	0.270	0.838	0.736	0.836	0.657	0.839	0.933	0.034
(0.500,0.600]	7	11	0.358	0.819	0.741	0.726	0.757	0.870	0.851	0.092
(0.400,0.500]	0	0	0.358	0.819	0.741	0.726	0.757	0.870	0.851	0.092
(0.300,0.400]	6	12	0.446	0.789	0.733	0.648	0.843	0.903	0.761	0.164
(0.200,0.300]	4	16	0.544	0.730	0.696	0.568	0.900	0.925	0.642	0.268
(0.100,0.200]	6	23	0.686	0.647	0.657	0.493	0.986	0.984	0.470	0.430
(0.000,0.100]	1	63	1.000	0.343	0.511	0.343	1.000	1.000	0.000	0.897

Figure D-38: Cross-selling evaluation results using Two-Class Decision Forest algorithm based on the optimized dataset (incl. Age, AvgExpenses, AvgBalance, AvgSalary)



Cross Selling > Evaluate Model > Evaluation results

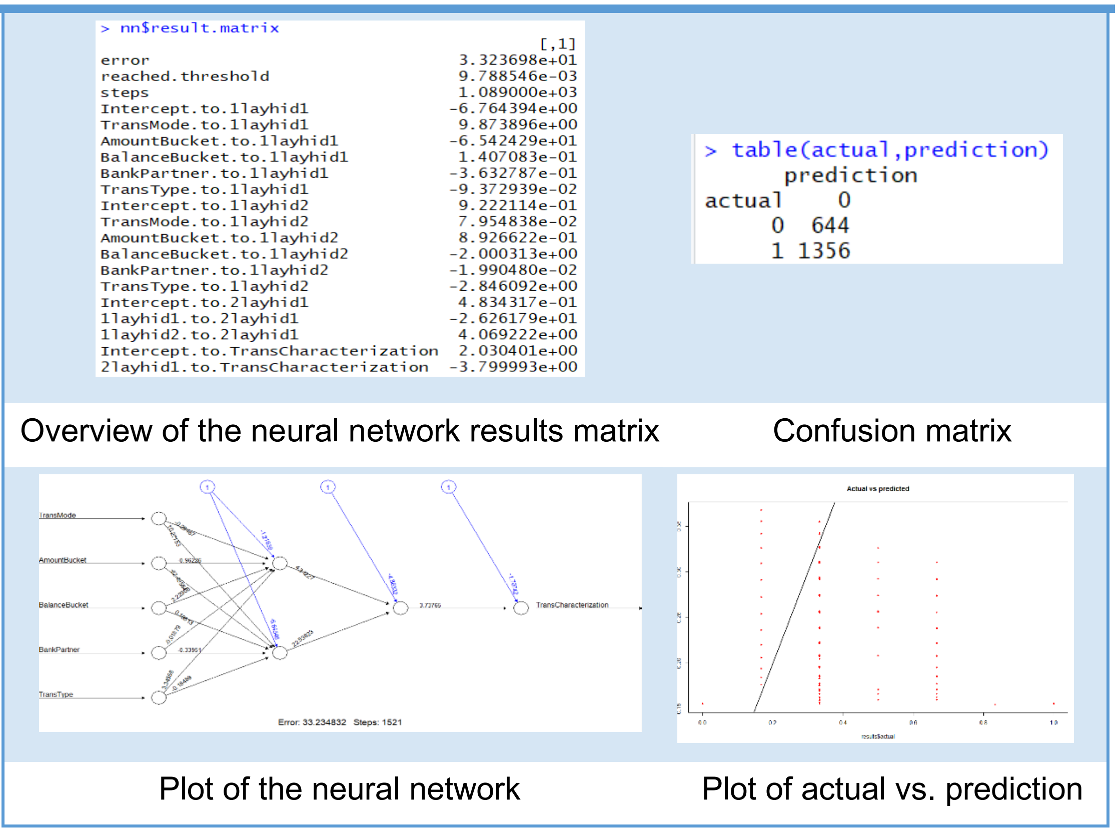
True Positive	False Negative	Accuracy	Precision	Threshold	AUC
49	21	0.804	0.721	0.5	0.877
False Positive	**True Negative**	**Recall**	**F1 Score**		
19	115	0.700	0.710		
Positive Label	**Negative Label**				
1	0				

Score Bin	Positive Examples	Negative Examples	Fraction Above Threshold	Accuracy	F1 Score	Precision	Recall	Negative Precision	Negative Recall	Cumulative AUC
(0.900,1.000]	16	0	0.078	0.735	0.372	1.000	0.229	0.713	1.000	0.000
(0.800,0.900]	13	3	0.157	0.784	0.569	0.906	0.414	0.762	0.978	0.007
(0.700,0.800]	9	4	0.221	0.809	0.661	0.844	0.543	0.799	0.948	0.021
(0.600,0.700]	11	12	0.333	0.804	0.710	0.721	0.700	0.846	0.858	0.077
(0.500,0.600]	7	7	0.402	0.804	0.737	0.683	0.800	0.885	0.806	0.116
(0.400,0.500]	0	0	0.402	0.804	0.737	0.683	0.800	0.885	0.806	0.116
(0.300,0.400]	4	15	0.495	0.750	0.702	0.594	0.857	0.903	0.694	0.209
(0.200,0.300]	3	10	0.559	0.716	0.685	0.553	0.900	0.922	0.619	0.275
(0.100,0.200]	5	23	0.696	0.627	0.642	0.479	0.971	0.968	0.448	0.435
(0.000,0.100]	2	60	1.000	0.343	0.511	0.343	1.000	1.000	0.000	0.877

Figure D-39: Cross-selling evaluation results using Two-Class Decision Forest algorithm based on the optimized dataset (incl. Age, AvgIncome, AvgBalance, AvgSalary)

D.6. Evaluation results of neural network algorithm for categorized transactions

#1 **bucket#1**



```
> nn$result.matrix
                                           [,1]
error                              3.323698e+01
reached.threshold                  9.788546e-03
steps                              1.089000e+03
Intercept.to.1layhid1             -6.764394e+00
TransMode.to.1layhid1              9.873896e+00
AmountBucket.to.1layhid1          -6.542429e+01
BalanceBucket.to.1layhid1          1.407083e-01
BankPartner.to.1layhid1           -3.632787e-01
TransType.to.1layhid1             -9.372939e-02
Intercept.to.1layhid2              9.222114e-01
TransMode.to.1layhid2              7.954838e-02
AmountBucket.to.1layhid2           8.926622e-01
BalanceBucket.to.1layhid2         -2.000313e+00
BankPartner.to.1layhid2           -1.990480e-02
TransType.to.1layhid2             -2.846092e+00
Intercept.to.2layhid1              4.834317e-01
1layhid1.to.2layhid1              -2.626179e+01
1layhid2.to.2layhid1               4.069222e+00
Intercept.to.TransCharacterization  2.030401e+00
2layhid1.to.TransCharacterization -3.799993e+00
```

```
> table(actual,prediction)
      prediction
actual    0
     0  644
     1 1356
```

Overview of the neural network results matrix

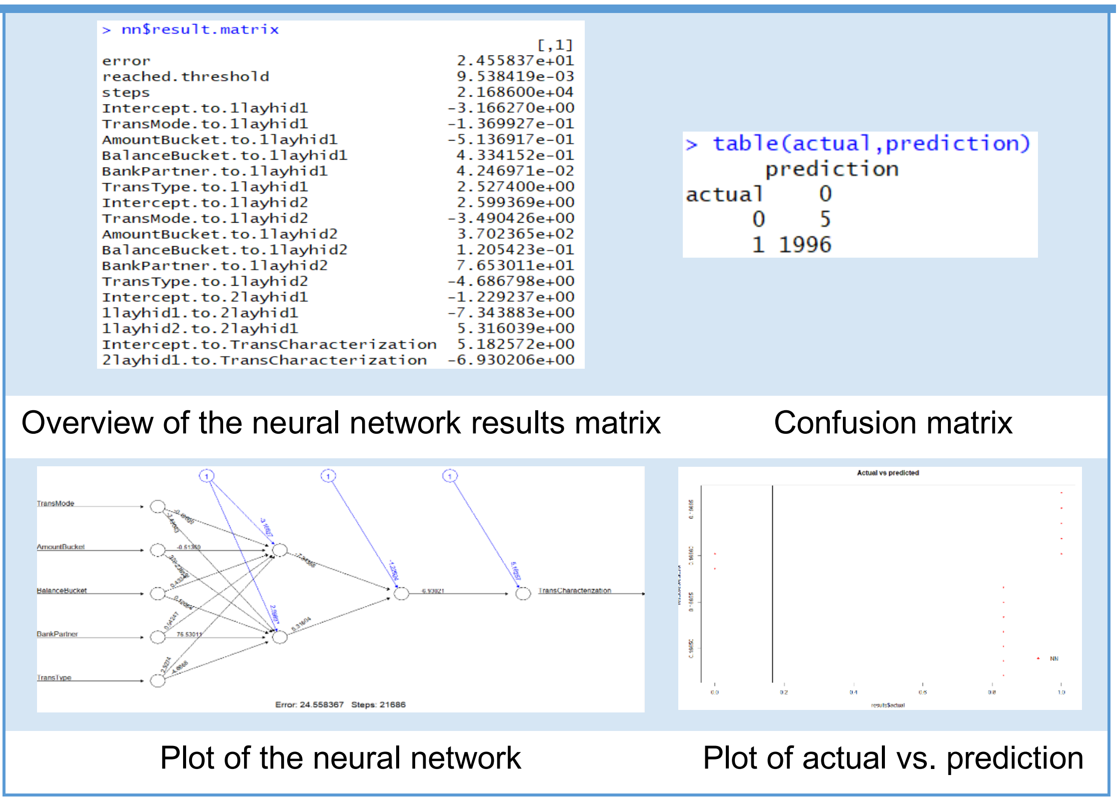
Confusion matrix

Plot of the neural network

Plot of actual vs. prediction

Figure D-40: Detailed predictive results of research experiment - bucket#1

#2 **bucket#2**



```
> nn$result.matrix
                                          [,1]
error                             2.455837e+01
reached.threshold                 9.538419e-03
steps                             2.168600e+04
Intercept.to.1layhid1            -3.166270e+00
TransMode.to.1layhid1            -1.369927e-01
AmountBucket.to.1layhid1         -5.136917e-01
BalanceBucket.to.1layhid1         4.334152e-01
BankPartner.to.1layhid1           4.246971e-02
TransType.to.1layhid1             2.527400e+00
Intercept.to.1layhid2             2.599369e+00
TransMode.to.1layhid2            -3.490426e+00
AmountBucket.to.1layhid2          3.702365e+02
BalanceBucket.to.1layhid2         1.205423e-01
BankPartner.to.1layhid2           7.653011e+01
TransType.to.1layhid2            -4.686798e+00
Intercept.to.2layhid1            -1.229237e+00
1layhid1.to.2layhid1             -7.343883e+00
1layhid2.to.2layhid1              5.316039e+00
Intercept.to.TransCharacterization  5.182572e+00
2layhid1.to.TransCharacterization -6.930206e+00
```

```
> table(actual,prediction)
      prediction
actual    0
     0    5
     1 1996
```

Overview of the neural network results matrix

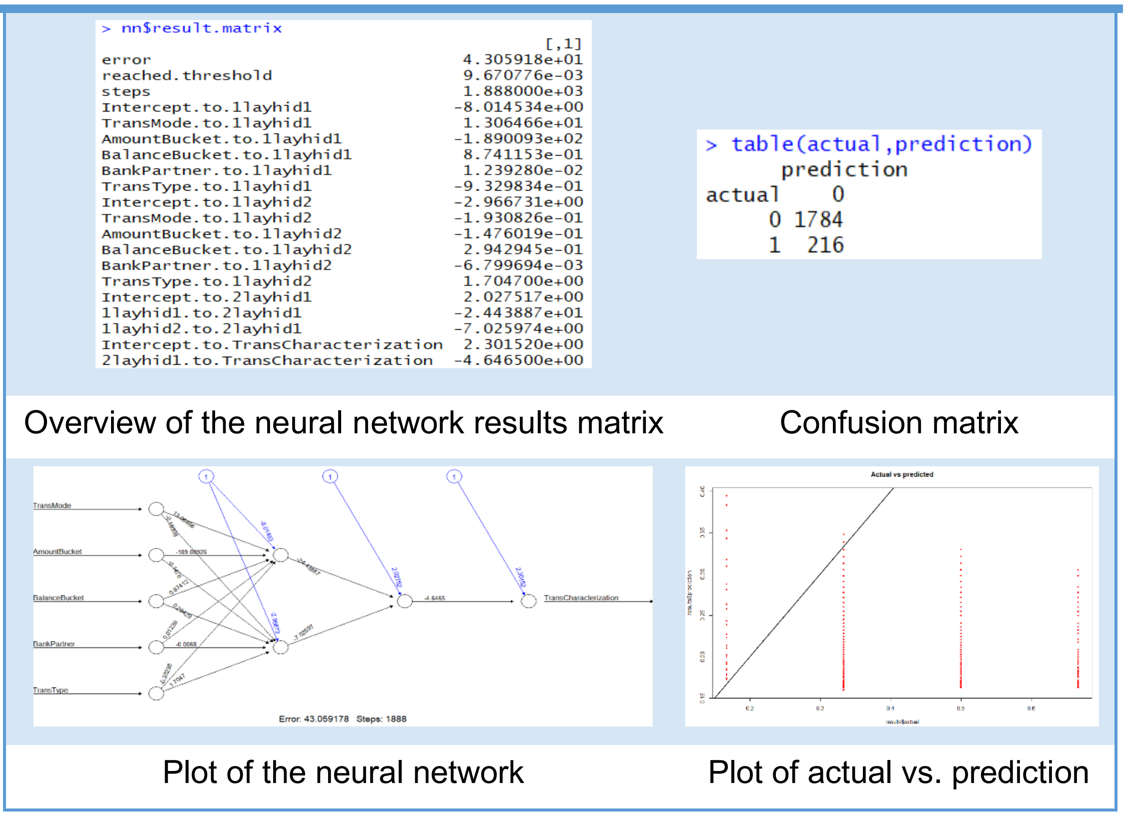
Confusion matrix

Plot of the neural network

Plot of actual vs. prediction

Figure D-41: Detailed predictive results of research experiment - bucket#2

#3 **bucket#3**



```
> nn$result.matrix
                                          [,1]
error                             4.305918e+01
reached.threshold                 9.670776e-03
steps                             1.888000e+03
Intercept.to.1layhid1            -8.014534e+00
TransMode.to.1layhid1             1.306466e+01
AmountBucket.to.1layhid1         -1.890093e+02
BalanceBucket.to.1layhid1         8.741153e-01
BankPartner.to.1layhid1           1.239280e-02
TransType.to.1layhid1            -9.329834e-01
Intercept.to.1layhid2            -2.966731e+00
TransMode.to.1layhid2            -1.930826e-01
AmountBucket.to.1layhid2         -1.476019e-01
BalanceBucket.to.1layhid2         2.942945e-01
BankPartner.to.1layhid2          -6.799694e-03
TransType.to.1layhid2             1.704700e+00
Intercept.to.2layhid1             2.027517e+00
1layhid1.to.2layhid1             -2.443887e+01
1layhid2.to.2layhid1             -7.025974e+00
Intercept.to.TransCharacterization  2.301520e+00
2layhid1.to.TransCharacterization -4.646500e+00
```

```
> table(actual,prediction)
      prediction
actual    0
     0 1784
     1  216
```

Overview of the neural network results matrix

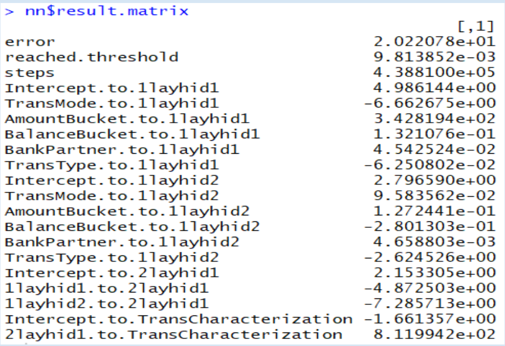
Confusion matrix

Plot of the neural network

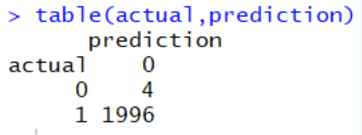
Plot of actual vs. prediction

Figure D-42: Detailed predictive results of research experiment - bucket#3

#4 **bucket#4**



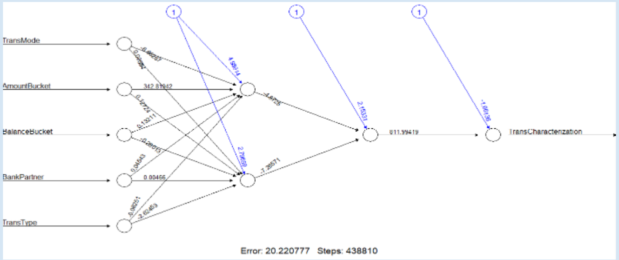
```
> nn$result.matrix
                                          [,1]
error                             2.022078e+01
reached.threshold                 9.813852e-03
steps                             4.388100e+05
Intercept.to.1layhid1             4.986144e+00
TransMode.to.1layhid1            -6.662675e+00
AmountBucket.to.1layhid1          3.428194e+02
BalanceBucket.to.1layhid1         1.321076e-01
BankPartner.to.1layhid1           4.542524e-02
TransType.to.1layhid1            -6.250802e-02
Intercept.to.1layhid2             2.796590e+00
TransMode.to.1layhid2             9.583562e-02
AmountBucket.to.1layhid2          1.272441e-01
BalanceBucket.to.1layhid2        -2.801303e-01
BankPartner.to.1layhid2           4.658803e-03
TransType.to.1layhid2            -2.624526e+00
Intercept.to.2layhid1             2.153305e+00
1layhid1.to.2layhid1             -4.872503e+00
1layhid2.to.2layhid1             -7.285713e+00
Intercept.to.TransCharacterization -1.661357e+00
2layhid1.to.TransCharacterization  8.119942e+02
```

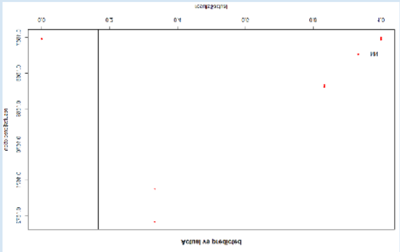


```
> table(actual,prediction)
      prediction
actual    0
     0    4
     1 1996
```

Overview of the neural network results matrix

Confusion matrix



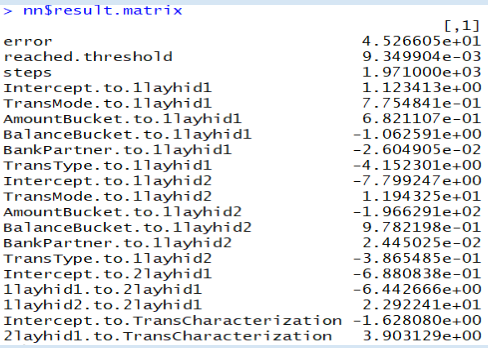


Plot of the neural network

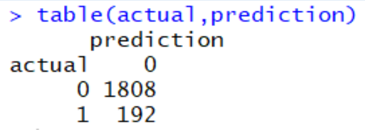
Plot of actual vs. prediction

Figure D-43: Detailed predictive results of research experiment - bucket#4

#5 **bucket#5**



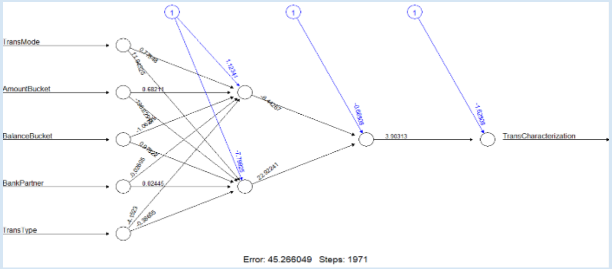
```
> nn$result.matrix
                                          [,1]
error                             4.526605e+01
reached.threshold                 9.349904e-03
steps                             1.971000e+03
Intercept.to.1layhid1             1.123413e+00
TransMode.to.1layhid1             7.754841e-01
AmountBucket.to.1layhid1          6.821107e-01
BalanceBucket.to.1layhid1        -1.062591e+00
BankPartner.to.1layhid1          -2.604905e-02
TransType.to.1layhid1            -4.152301e+00
Intercept.to.1layhid2            -7.799247e+00
TransMode.to.1layhid2             1.194325e+01
AmountBucket.to.1layhid2         -1.966291e+02
BalanceBucket.to.1layhid2         9.782198e-01
BankPartner.to.1layhid2           2.445025e-02
TransType.to.1layhid2            -3.865485e-01
Intercept.to.2layhid1            -6.880838e-01
1layhid1.to.2layhid1             -6.442666e+00
1layhid2.to.2layhid1              2.292241e+01
Intercept.to.TransCharacterization -1.628080e+00
2layhid1.to.TransCharacterization  3.903129e+00
```

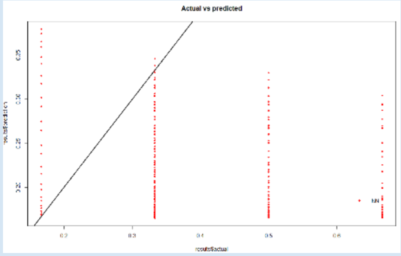


```
> table(actual,prediction)
      prediction
actual    0
     0 1808
     1  192
```

Overview of the neural network results matrix

Confusion matrix



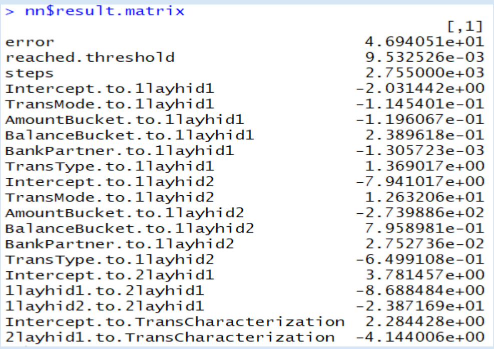


Plot of the neural network

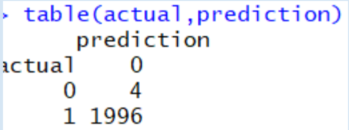
Plot of actual vs. prediction

Figure D-44: Detailed predictive results of research experiment - bucket#5

#6 **bucket#6**

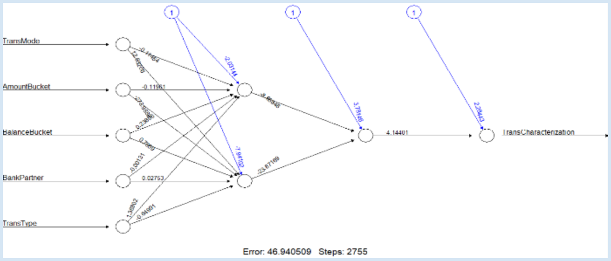


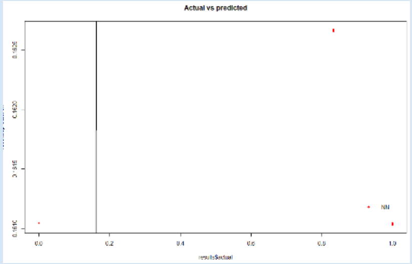
```
> nn$result.matrix
                                              [,1]
error                                 4.694051e+01
reached.threshold                     9.532526e-03
steps                                 2.755000e+03
Intercept.to.1layhid1                -2.031442e+00
TransMode.to.1layhid1                -1.145401e-01
AmountBucket.to.1layhid1             -1.196067e-01
BalanceBucket.to.1layhid1             2.389618e-01
BankPartner.to.1layhid1              -1.305723e-03
TransType.to.1layhid1                 1.369017e+00
Intercept.to.1layhid2                -7.941017e+00
TransMode.to.1layhid2                 1.263206e+01
AmountBucket.to.1layhid2             -2.739886e+02
BalanceBucket.to.1layhid2             7.958981e-01
BankPartner.to.1layhid2               2.752736e-02
TransType.to.1layhid2                -6.499108e-01
Intercept.to.2layhid1                 3.781457e+00
1layhid1.to.2layhid1                 -8.688484e+00
1layhid2.to.2layhid1                 -2.387169e+01
Intercept.to.TransCharacterization    2.284428e+00
2layhid1.to.TransCharacterization    -4.144006e+00
```



```
> table(actual,prediction)
      prediction
actual    0
     0    4
     1 1996
```

Overview of the neural network results matrix | Confusion matrix

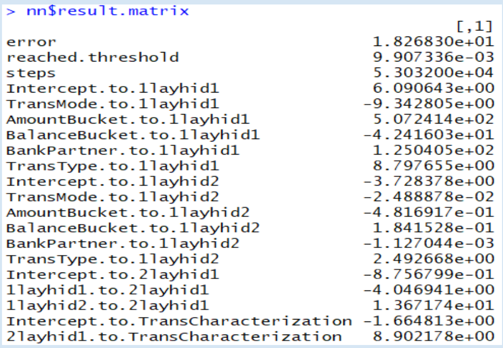




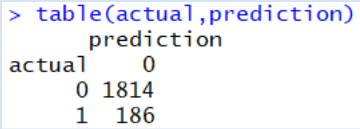
Plot of the neural network | Plot of actual vs. prediction

Figure D-45: Detailed predictive results of research experiment - bucket#6

#7 **bucket#7**

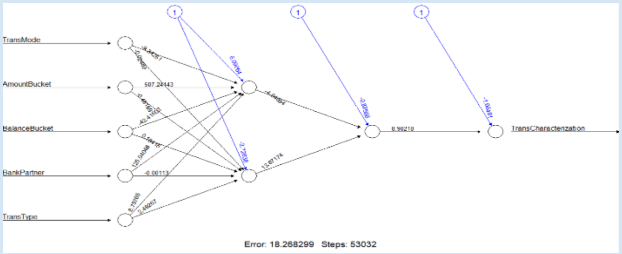


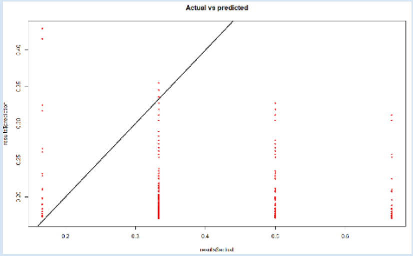
```
> nn$result.matrix
                                              [,1]
error                                 1.826830e+01
reached.threshold                     9.907336e-03
steps                                 5.303200e+04
Intercept.to.1layhid1                 6.090643e+00
TransMode.to.1layhid1                -9.342805e+00
AmountBucket.to.1layhid1              5.072414e+02
BalanceBucket.to.1layhid1            -4.241603e+01
BankPartner.to.1layhid1               1.250405e+02
TransType.to.1layhid1                 8.797655e+00
Intercept.to.1layhid2                -3.728378e+00
TransMode.to.1layhid2                -2.488878e-02
AmountBucket.to.1layhid2             -4.816917e-01
BalanceBucket.to.1layhid2             1.841528e-01
BankPartner.to.1layhid2              -1.127044e-03
TransType.to.1layhid2                 2.492668e+00
Intercept.to.2layhid1                -8.756799e-01
1layhid1.to.2layhid1                 -4.046941e+00
1layhid2.to.2layhid1                  1.367174e+01
Intercept.to.TransCharacterization   -1.664813e+00
2layhid1.to.TransCharacterization     8.902178e+00
```



```
> table(actual,prediction)
      prediction
actual    0
     0 1814
     1  186
```

Overview of the neural network results matrix | Confusion matrix

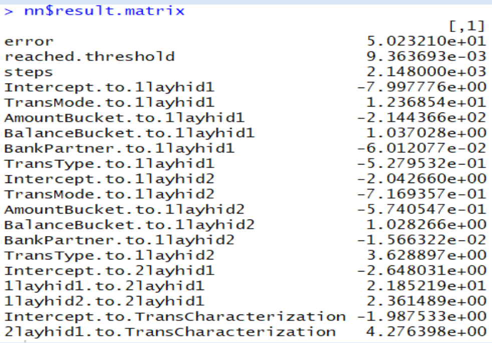




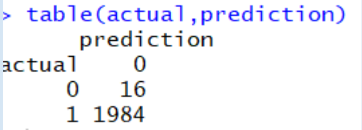
Plot of the neural network | Plot of actual vs. prediction

Figure D-46: Detailed predictive results of research experiment - bucket#7

#8 **bucket#8**

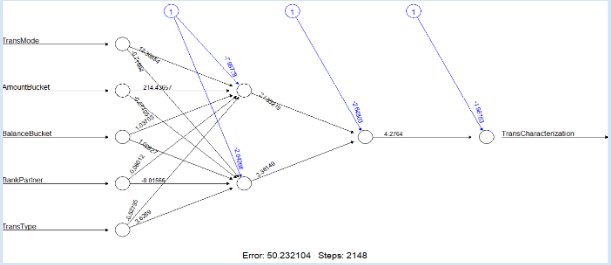


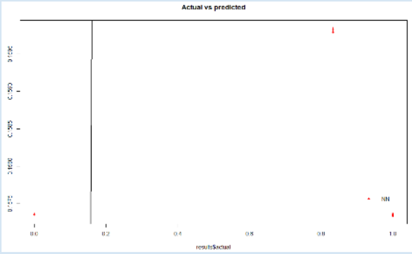
```
> nn$result.matrix
                                          [,1]
error                             5.023210e+01
reached.threshold                 9.363693e-03
steps                             2.148000e+03
Intercept.to.1layhid1            -7.997776e+00
TransMode.to.1layhid1             1.236854e+01
AmountBucket.to.1layhid1         -2.144366e+02
BalanceBucket.to.1layhid1         1.037028e+00
BankPartner.to.1layhid1          -6.012077e-02
TransType.to.1layhid1            -5.279532e-01
Intercept.to.1layhid2            -2.042660e+00
TransMode.to.1layhid2            -7.169357e-01
AmountBucket.to.1layhid2         -5.740547e-01
BalanceBucket.to.1layhid2         1.028266e+00
BankPartner.to.1layhid2          -1.566322e-02
TransType.to.1layhid2             3.628897e+00
Intercept.to.2layhid1            -2.648031e+00
1layhid1.to.2layhid1              2.185219e+01
1layhid2.to.2layhid1              2.361489e+00
Intercept.to.TransCharacterization -1.987533e+00
2layhid1.to.TransCharacterization  4.276398e+00
```



```
> table(actual,prediction)
      prediction
actual    0
     0   16
     1 1984
```

Overview of the neural network results matrix | Confusion matrix

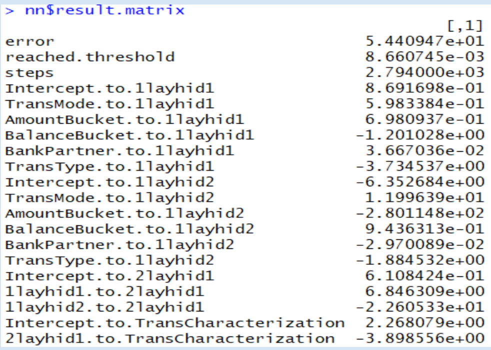




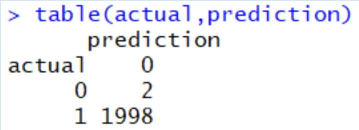
Plot of the neural network | Plot of actual vs. prediction

Figure D-47: Detailed predictive results of research experiment - bucket#8

#9 **bucket#9**

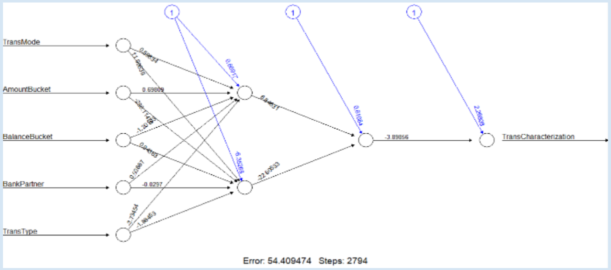


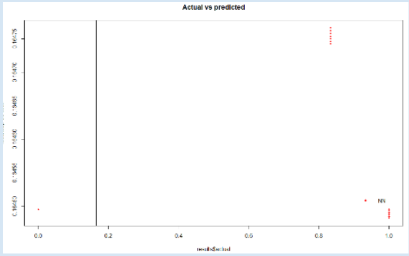
```
> nn$result.matrix
                                          [,1]
error                             5.440947e+01
reached.threshold                 8.660745e-03
steps                             2.794000e+03
Intercept.to.1layhid1             8.691698e-01
TransMode.to.1layhid1             5.983384e-01
AmountBucket.to.1layhid1          6.980937e-01
BalanceBucket.to.1layhid1        -1.201028e+00
BankPartner.to.1layhid1           3.667036e-02
TransType.to.1layhid1            -3.734537e+00
Intercept.to.1layhid2            -6.352684e+00
TransMode.to.1layhid2             1.199639e+01
AmountBucket.to.1layhid2         -2.801148e+02
BalanceBucket.to.1layhid2         9.436313e-01
BankPartner.to.1layhid2          -2.970089e-02
TransType.to.1layhid2            -1.884532e+00
Intercept.to.2layhid1             6.108424e-01
1layhid1.to.2layhid1              6.846309e+00
1layhid2.to.2layhid1             -2.260533e+01
Intercept.to.TransCharacterization  2.268079e+00
2layhid1.to.TransCharacterization -3.898556e+00
```



```
> table(actual,prediction)
      prediction
actual    0
     0    2
     1 1998
```

Overview of the neural network results matrix | Confusion matrix

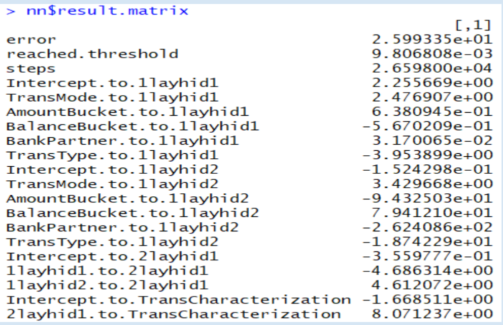




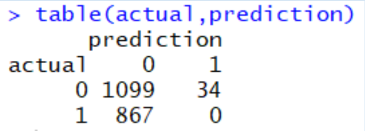
Plot of the neural network | Plot of actual vs. prediction

Figure D-48: Detailed predictive results of research experiment - bucket#9

#10 **bucket#10**



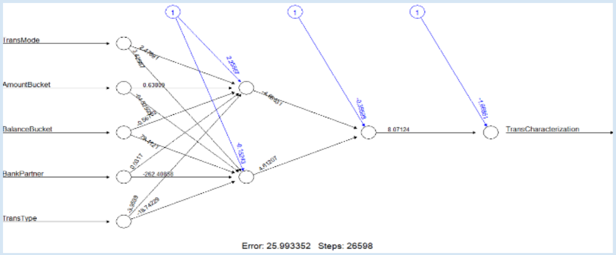
```
> nn$result.matrix
                                               [,1]
error                                  2.599335e+01
reached.threshold                      9.806808e-03
steps                                  2.659800e+04
Intercept.to.1layhid1                  2.255669e+00
TransMode.to.1layhid1                  2.476907e+00
AmountBucket.to.1layhid1               6.380945e-01
BalanceBucket.to.1layhid1             -5.670209e-01
BankPartner.to.1layhid1                3.170065e-02
TransType.to.1layhid1                 -3.953899e+00
Intercept.to.1layhid2                 -1.524298e-01
TransMode.to.1layhid2                  3.429668e+00
AmountBucket.to.1layhid2              -9.432503e+01
BalanceBucket.to.1layhid2              7.941210e+01
BankPartner.to.1layhid2               -2.624086e+02
TransType.to.1layhid2                 -1.874229e+01
Intercept.to.2layhid1                 -3.559777e-01
1layhid1.to.2layhid1                  -4.686314e+00
1layhid2.to.2layhid1                   4.612072e+00
Intercept.to.TransCharacterization    -1.668511e+00
2layhid1.to.TransCharacterization      8.071237e+00
```



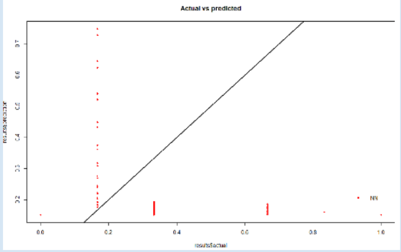
```
> table(actual,prediction)
      prediction
actual    0    1
     0 1099   34
     1  867    0
```

Overview of the neural network results matrix

Confusion matrix



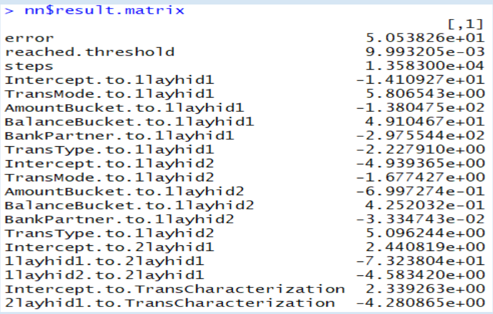
Plot of the neural network



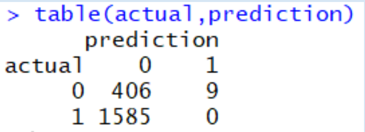
Plot of actual vs. prediction

Figure D-49: Detailed predictive results of research experiment - bucket#10

#11 **bucket#11**



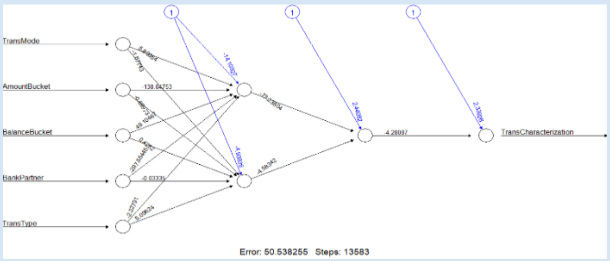
```
> nn$result.matrix
                                               [,1]
error                                  5.053826e+01
reached.threshold                      9.993205e-03
steps                                  1.358300e+04
Intercept.to.1layhid1                 -1.410927e+01
TransMode.to.1layhid1                  5.806543e+00
AmountBucket.to.1layhid1              -1.380475e+02
BalanceBucket.to.1layhid1              4.910467e+01
BankPartner.to.1layhid1               -2.975544e+02
TransType.to.1layhid1                 -2.227910e+00
Intercept.to.1layhid2                 -4.939365e+00
TransMode.to.1layhid2                 -1.677427e+00
AmountBucket.to.1layhid2              -6.997274e-01
BalanceBucket.to.1layhid2              4.252032e-01
BankPartner.to.1layhid2               -3.334743e-02
TransType.to.1layhid2                  5.096244e+00
Intercept.to.2layhid1                  2.440819e+00
1layhid1.to.2layhid1                  -7.323804e+01
1layhid2.to.2layhid1                  -4.583420e+00
Intercept.to.TransCharacterization     2.339263e+00
2layhid1.to.TransCharacterization     -4.280865e+00
```



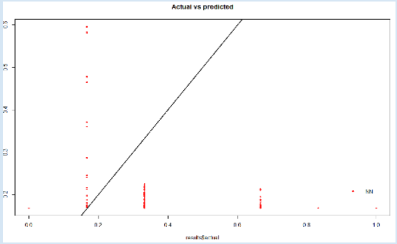
```
> table(actual,prediction)
      prediction
actual    0    1
     0  406    9
     1 1585    0
```

Overview of the neural network results matrix

Confusion matrix



Plot of the neural network



Plot of actual vs. prediction

Figure D-50: Detailed predictive results of research experiment - bucket#11

D.7. Configuration parameters of the supervised learning algorithms

D.7.1 Properties of the credit-scoring models

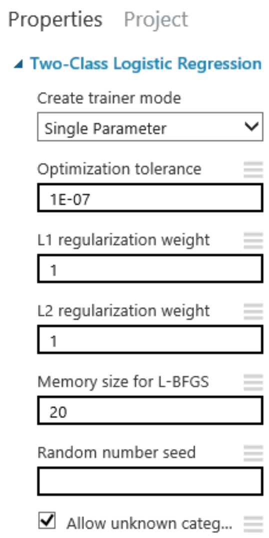
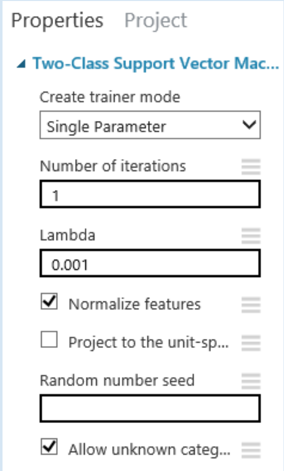
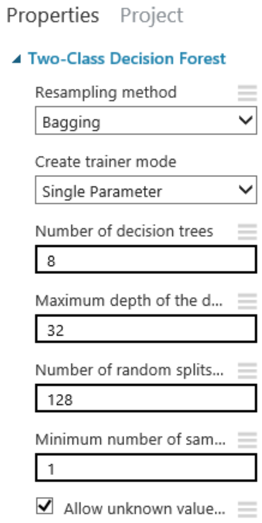
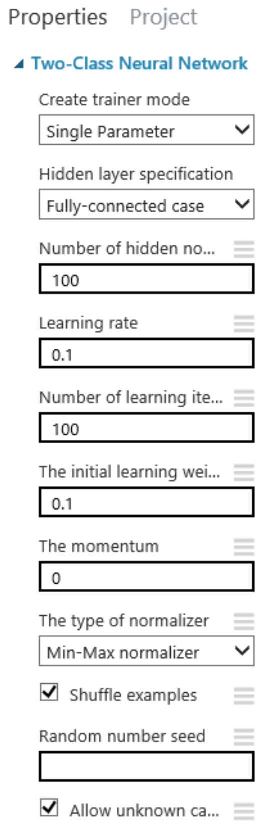
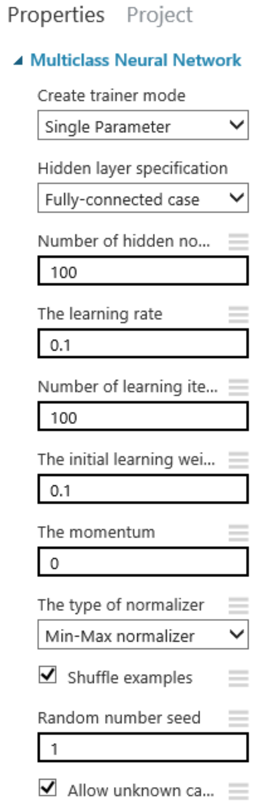
Overview of the parameterized credit scoring models				
Two-Class Logistic Regression	Two-Class Support Vector Machine	Two-Class Decision Forest	Two-Class Neural Network	Multiclass Neural Network
 Properties Project Two-Class Logistic Regression Create trainer mode Single Parameter Optimization tolerance 1E-07 L1 regularization weight 1 L2 regularization weight 1 Memory size for L-BFGS 20 Random number seed ☑ Allow unknown categ...	 Properties Project Two-Class Support Vector Mac... Create trainer mode Single Parameter Number of iterations 1 Lambda 0.001 ☑ Normalize features □ Project to the unit-sp... Random number seed ☑ Allow unknown categ...	 Properties Project Two-Class Decision Forest Resampling method Bagging Create trainer mode Single Parameter Number of decision trees 8 Maximum depth of the d... 32 Number of random splits... 128 Minimum number of sam... 1 ☑ Allow unknown value...	 Properties Project Two-Class Neural Network Create trainer mode Single Parameter Hidden layer specification Fully-connected case Number of hidden no... 100 Learning rate 0.1 Number of learning ite... 100 The initial learning wei... 0.1 The momentum 0 The type of normalizer Min-Max normalizer ☑ Shuffle examples Random number seed ☑ Allow unknown ca...	 Properties Project Multiclass Neural Network Create trainer mode Single Parameter Hidden layer specification Fully-connected case Number of hidden no... 100 The learning rate 0.1 Number of learning ite... 100 The initial learning wei... 0.1 The momentum 0 The type of normalizer Min-Max normalizer ☑ Shuffle examples Random number seed 1 ☑ Allow unknown ca...

Figure D-51: Overview of the properties of all applied machine learning algorithms for the credit scoring model

D.7.2 Properties of the cross-selling models

Overview of the parameterized cross-selling models

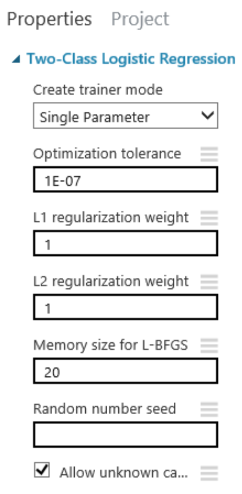
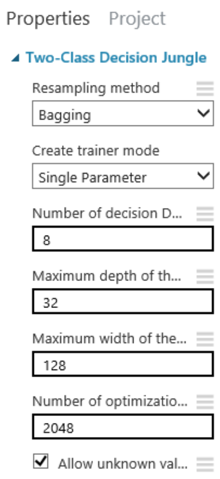
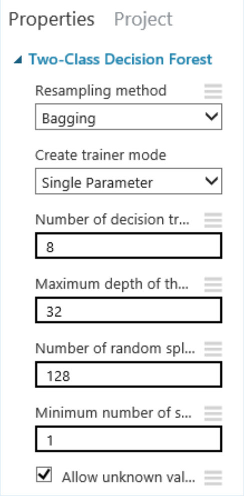
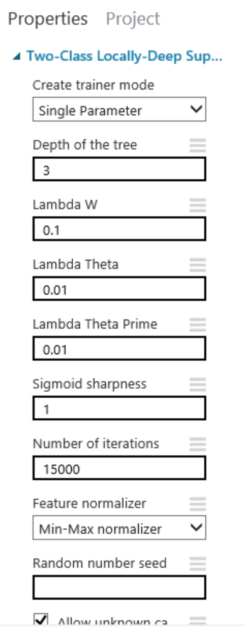
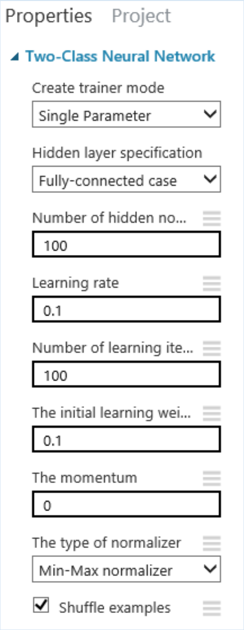
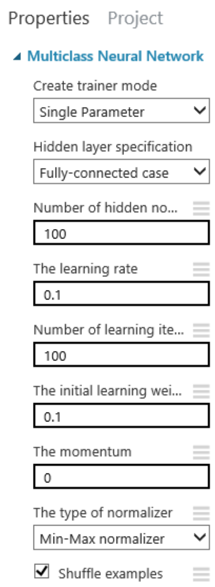
Two-Class Logistic Regression	Two-Class Decision Jungle	Two-Class Decision Forest	Two-Class Locally-Deep Support Vector Machine	Two-Class Neural Network	Multiclass Neural Network
 Properties Project Two-Class Logistic Regression Create trainer mode: Single Parameter Optimization tolerance: 1E-07 L1 regularization weight: 1 L2 regularization weight: 1 Memory size for L-BFGS: 20 Random number seed: ☑ Allow unknown ca...	 Properties Project Two-Class Decision Jungle Resampling method: Bagging Create trainer mode: Single Parameter Number of decision D...: 8 Maximum depth of th...: 32 Maximum width of the...: 128 Number of optimizatio...: 2048 ☑ Allow unknown val...	 Properties Project Two-Class Decision Forest Resampling method: Bagging Create trainer mode: Single Parameter Number of decision tr...: 8 Maximum depth of th...: 32 Number of random spl...: 128 Minimum number of s...: 1 ☑ Allow unknown val...	 Properties Project Two-Class Locally-Deep Sup... Create trainer mode: Single Parameter Depth of the tree: 3 Lambda W: 0.1 Lambda Theta: 0.01 Lambda Theta Prime: 0.01 Sigmoid sharpness: 1 Number of iterations: 15000 Feature normalizer: Min-Max normalizer Random number seed: ☑ Allow unknown ca	 Properties Project Two-Class Neural Network Create trainer mode: Single Parameter Hidden layer specification: Fully-connected case Number of hidden no...: 100 Learning rate: 0.1 Number of learning ite...: 100 The initial learning wei...: 0.1 The momentum: 0 The type of normalizer: Min-Max normalizer ☑ Shuffle examples	 Properties Project Multiclass Neural Network Create trainer mode: Single Parameter Hidden layer specification: Fully-connected case Number of hidden no...: 100 The learning rate: 0.1 Number of learning ite...: 100 The initial learning wei...: 0.1 The momentum: 0 The type of normalizer: Min-Max normalizer ☑ Shuffle examples

Figure D-52: Overview of the properties of all applied machine learning algorithms for the cross-selling model

Appendix E: Scripts - code snippets

E.1. R code

E.1.1 Gather.R

```
## This script is designed for pre-processing the cleaned data from
Parse.R script
## in order to aggregate the data and its attributes (i.e. avg monthly
income etc.)
## for applied machine learning algorithm in MS AZURE (i.e. data modeling
& visualizing the results)

install.packages("dplyr")
install.packages("zoo")
install.packages("lubridate")
install.packages("ggplot2")
install.packages("scales")

library(dplyr)
library(zoo)
library(lubridate)
library(ggplot2)
library(scales)

setwd("C:/Users/hakim.harrach/Documents/UEL/PhD/3-Doctoral Research/R-
Project/Research_Transactional_Behaviour/Dataset/final")

trans <- read.csv2("trans_complete.csv")
account <- read.csv2("account_adap.csv")
card <- read.csv2("card_adap.csv")
client <- read.csv2("client_adap.csv")
disp <- read.csv("disp.csv", sep=";")
district <- read.csv2("district_adap.csv")
loan <- read.csv2("loan_adap.csv")
order_df <- read.csv2("order_adap.csv")

trans$date <- as.Date(trans$date)
card$issued <- as.Date(card$issued)
client$birthdate <- as.Date(client$birthdate)
loan$date <- as.Date(loan$date)
trans <- trans %>% mutate(amount = ifelse(type == "withdrawal", -amount,
amount))
trans <- trans %>% mutate(Month = as.yearmon(date))

#helping dataframes
loan_orders <- order_df %>% filter(k_symbol == "loan payment")
str(loan_orders)

districtRelevantData <- district %>% mutate(CrimeRatio95 =
CommitedCrimes95/NoInhabitants,
                                        CrimeRatio96 =
CommitedCrimes96/NoInhabitants,
                                        EnterpreneursRatio =
NoEnterpreneurs/NoInhabitants) %>%
                              select(district_id, NoInhabitants,
UrbanRatio, AverageSalary,
                                     Unemployment95, Unemployment96,
CrimeRatio95, CrimeRatio96,
                                     EnterpreneursRatio)
str(districtRelevantData)
```

```
#initialize resulting dataframe
res <- loan %>% select(loan_id, account_id, status)
str(res)

#indicator whether negative balance
negativeBalance <- trans %>% filter(balance < 0) %>%
select(account_id) %>% unique()
str(negativeBalance)

res <- res %>% mutate(negativeBalance = ifelse(account_id %in%
negativeBalance$account_id, 1, 0))
res$negativeBalance <- as.factor(res$negativeBalance)
str(res)

#number of permanent orders
tmp <- order_df %>% group_by(account_id) %>%
        summarize(PermanentOrders = n()) %>% ungroup()
str(tmp)

df_permanentorders <- loan %>% inner_join(tmp,by = "account_id") %>%
                    select(account_id, PermanentOrders) %>%
                    mutate(PermanentOrders =
ifelse(is.na(PermanentOrders),0,PermanentOrders))

res <- res %>% inner_join(df_permanentorders, by = "account_id")
str(res)

#balance at end of month
balance <- trans %>% filter(account_id %in% loan$account_id) %>%
group_by(account_id,Month) %>%
            filter(date == max(date)) %>% filter(trans_id ==
min(trans_id)) %>%
            ungroup() %>% select(account_id, Month, balance)
str(balance)

#balance to loan payment

#average income and expenses
  Cashflows <- trans %>% #filter(account_id %in% loan$account_id) %>%
            group_by(account_id,Month) %>% summarize(MonthlyIncome =
sum(ifelse(amount > 0 , amount ,0)),
                  MonthlyExpenses = sum(ifelse(amount < 0, amount,
0))) %>%
          ungroup() %>% select(account_id, Month, MonthlyIncome,
MonthlyExpenses) %>%
          mutate(Saldo = MonthlyIncome + MonthlyExpenses)
str(Cashflows)

CashflowsAggregated <- Cashflows %>% group_by(account_id) %>%
      summarize(AvgIncome = mean(MonthlyIncome), AvgExpenses =
mean(MonthlyExpenses)) %>% ungroup()
str(CashflowsAggregated)

tmp <- balance %>% group_by(account_id) %>%
      summarize(AvgBalance = mean(balance)) %>% ungroup()
CashflowsAggregated <- CashflowsAggregated %>% inner_join(tmp, by =
"account_id")
str(CashflowsAggregated)

res <- res %>% inner_join(CashflowsAggregated, by = "account_id")
str(res)
```

```
#ratios to loan payment

#other loans
loans_others <- loan_orders %>% filter(!(account_id %in%
loan$account_id))

#demographic data
tmp <- disp %>% filter(account_id %in% loan$account_id & type
=="OWNER") %>%
        select(client_id, account_id)
res <- res %>% inner_join(tmp, by = "account_id")

#age
GetAge <- function(birthDate, refDate = as.Date("1999-12-31")){
  period <- as.period(interval(birthDate, refDate),
                      unit = "year")
  return(period$year)
}

tmp <- client %>% filter(client_id %in% res$client_id) %>%
      mutate(Age = GetAge(birthdate)) %>% select(client_id, district_id,
sex, Age)
tmp <- tmp %>% inner_join(districtRelevantData, by = "district_id") %>%
select(-district_id)
res <- res %>% inner_join(tmp, by = "client_id")
str(res)

#res <- res %>% mutate(IncomeToAvg = AvgIncome/AverageSalary)
#=> CAR

#number of defaults per district
NoDefaultsPerDistrict <- res %>% mutate(status =
as.character(status)) %>%
        inner_join(client, by = "client_id") %>%
        select(district_id, status) %>% filter(status == "B" | status ==
"D") %>%
        group_by(district_id) %>% summarize(DefaultsAbsolute = n()) %>%
ungroup()

NoLoansPerDistrict <- res %>% mutate(status = as.character(status)) %>%
  inner_join(client, by = "client_id") %>%
  select(district_id, status)  %>%
  group_by(district_id) %>% summarize(Count = n()) %>% ungroup()

NoDefaultsPerDistrict <- NoDefaultsPerDistrict %>%
inner_join(NoLoansPerDistrict, by = "district_id") %>%
                        mutate(DefaultsRelative = DefaultsAbsolute/Count)
str(NoDefaultsPerDistrict)
#same per region

NoDefaultsPerRegion <- res %>% mutate(status = as.character(status)) %>%
  inner_join(client, by = "client_id") %>% inner_join(district, by =
"district_id") %>%
  select(region, status) %>% filter(status == "B" | status == "D") %>%
  group_by(region) %>% summarize(DefaultsAbsolutePerRegion = n()) %>%
ungroup()
str(NoDefaultsPerRegion)

NoLoansPerRegion <- res %>% mutate(status = as.character(status)) %>%
  inner_join(client, by = "client_id") %>% inner_join(district, by =
"district_id") %>%
  select(region, status) %>%
  group_by(region) %>% summarize(CountPerRegion = n()) %>% ungroup()
str(NoLoansPerRegion)

NoDefaultsPerRegion <- NoDefaultsPerRegion %>%
inner_join(NoLoansPerRegion, by = "region") %>%
  mutate(DefaultsRelativePerRegion =
DefaultsAbsolutePerRegion/CountPerRegion)
str(NoDefaultsPerRegion)

#good client => rating, cltv

#cross selling => credit cards, loans
tmp <- card %>% inner_join(disp, by= "disp_id") %>% select(account_id)
t <- account %>% inner_join(disp, by="account_id") %>% select(-
district_id) %>%
      inner_join(client, by = "client_id") %>%
      inner_join(districtRelevantData, by = "district_id") %>%
      select(account_id, frequency, sex, Age, NoInhabitants, UrbanRatio,
AverageSalary,
             Unemployment95, Unemployment96, CrimeRatio95, CrimeRatio96,
             EnterpreneursRatio, account_id) %>%
inner_join(CashflowsAggregated, by="account_id") %>%
             mutate(Cardholder = ifelse(account_id %in%
tmp$account_id,1,0)) %>%
              select (-account_id)
res <- res %>% mutate(Cardholder = ifelse(account_id %in%
tmp$account_id,1,0))
str(res)

#credit cards
#cross-selling
str(t)

write.csv2(res,"res.csv", row.names = F)
write.csv(res,"res_azure.csv", row.names = F, sep=",")
write.csv2(t, "creditcard.csv", row.names = F)
write.csv(t,"creditcard_azure.csv", row.names = F, sep=",")

hist_res <- res %>% mutate(status = ifelse(status == "C" | status == "A",
"good", "default")) %>%
        group_by(status) %>% summarize(Count = n()) %>% ungroup() %>%
mutate(Count=Count/nrow(res))
write.csv2(hist_res, "hist_res.csv", row.names=F)

p <- ggplot(data =hist_res, aes(x = status, y = Count, fill = status)) +
   geom_bar(stat="identity") + scale_y_continuous(labels =
scales::percent)#+
  # geom_text(aes( label = scales::percent(..Count..),
  #                y= ..Count.. ), stat= "count", vjust = -.5)

png("res_count2.pgn")
print(p)
dev.off()

##############################################
##CREDIT_SCORING
#Optimizing the random forest algorithm by identifying the most relevant
attributes within modelling process
#Comparing both algorithmn - the origin vs. optimized dataset
```

```
display_res_azure <- read.csv("res.csv", sep=";", stringsAsFactors =
FALSE)
# import the cleaned dataset
display_res_azure <- read.csv("res_azure_.csv", sep=",", stringsAsFactors
= FALSE)

str(display_res_azure)
summary (display_res_azure)

install.packages("randomForest")
install.packages("caret")

library(randomForest)

# excluding loan_id,account_id,client_id from the dataset
myvars <- names(display_res_azure) %in%
c("account_id","loan_id","client_id","negativeBalance")
display_res_azure <- display_res_azure[!myvars]

# converting cardholder, sex, status into factor variables
display_res_azure$Cardholder <- as.factor(display_res_azure$Cardholder)
display_res_azure$sex <- as.factor(display_res_azure$sex)
display_res_azure$status <- as.factor(display_res_azure$status)

# removing missing values in object
display_res_azure_opt <- na.omit(display_res_azure)

str(display_res_azure_opt)
summary(display_res_azure_opt)

install.packages("polycor")
library(polycor)

display_res_azure_opt.cor <- hetcor(display_res_azure_opt)
print(display_res_azure_opt.cor$correlations, digits = 2)
print(display_res_azure_opt.cor$tests, digits = 2)

#Creates an optimized classification model using a random forest
algorithms
fit <- randomForest(display_res_azure_opt$status ~ .,
data=display_res_azure_opt)
# import caret library using varImp to see import variables in the fit
model
library(caret)
varImp(fit)
varImpPlot(fit,type=2)

# ++++++++++++++++++++++++++++
#Flatten Correlation Matrix with significance levels (p-value) for the
detected significant variables
# ++++++++++++++++++++++++++++
install.packages("Hmisc")
library("Hmisc")

# flattenCorrMatrix function
# cormat : matrix of the correlation coefficients
# pmat : matrix of the correlation p-values
flattenCorrMatrix <- function(cormat, pmat) {
  ut <- upper.tri(cormat)
  data.frame(
    row = rownames(cormat)[row(cormat)[ut]],
    column = rownames(cormat)[col(cormat)[ut]],
    cor  =(cormat)[ut],
    p = pmat[ut]
  )
}

# prepare optimized dataset by including the top most important variables
myvars <- names(display_res_azure_opt) %in%
c("AvgBalance","AvgIncome","AvgExpenses","Age")
display_res_azure_opt_varImp <- display_res_azure_opt[myvars]

res2 <- rcorr(as.matrix(display_res_azure_opt_varImp))
flattenCorrMatrix(res2$r, res2$P)

# scatterplots for the correlations
install.packages("PerformanceAnalytics")
library("PerformanceAnalytics")
chart.Correlation(display_res_azure_opt_varImp, histogram=TRUE, pch=19)


##############################################
##CROSS_SELLING
#Comparing both algorithmn - the origin vs. optimized dataset
display_creditcard_azure <- read.csv("creditcard.csv", sep=";",
stringsAsFactors = FALSE)

# import the cleaned dataset
display_creditcard_azure <- read.csv("creditcard_.csv", sep=",",
stringsAsFactors = FALSE)

str(display_creditcard_azure)
summary (display_creditcard_azure)

install.packages("randomForest")
install.packages("caret")

library(randomForest)

# excluding loan_id,account_id,client_id from the dataset
#myvars <- names(display_creditcard_azure) %in%
c("account_id","loan_id","client_id")
#display_creditcard_azure <- display_creditcard_azure[!myvars]

# converting into factor variables
display_creditcard_azure$Cardholder <-
as.factor(display_creditcard_azure$Cardholder)
display_creditcard_azure$frequency <-
as.factor(display_creditcard_azure$frequency)
display_creditcard_azure$sex <- as.factor(display_creditcard_azure$sex)

# removing missing values in object
display_creditcard_azure_opt <- na.omit(display_creditcard_azure)

str(display_creditcard_azure_opt)
summary(display_creditcard_azure_opt)

install.packages("polycor")
library(polycor)

display_creditcard_azure_opt.cor <- hetcor(display_creditcard_azure_opt)
print(display_creditcard_azure_opt.cor$correlations, digits = 2)
print(display_creditcard_azure_opt.cor$tests, digits = 2)
```

```
#Creates an optimized classification model using a random forest
algorithms
fit <- randomForest(display_creditcard_azure_opt$Cardholder ~ .,
data=display_creditcard_azure_opt)
# import caret library using varImp to see import variables in the fit
model
install.packages("caret")
install.packages("lattice")
library(caret)
library(lattice)
varImp(fit)
varImpPlot(fit,type=2)

# prepare optimized dataset by including the top most important variables
myvars <- names(display_creditcard_azure_opt) %in%
c("AvgBalance","AvgIncome","AvgExpenses","Age")
display_creditcard_azure_opt_varImp <-
display_creditcard_azure_opt[myvars]

res2 <- rcorr(as.matrix(display_creditcard_azure_opt_varImp))
flattenCorrMatrix(res2$r, res2$P)

# scatterplots for the correlations
install.packages("PerformanceAnalytics")
library("PerformanceAnalytics")
chart.Correlation(display_creditcard_azure_opt_varImp, histogram=TRUE,
pch=19)


##########################################################################
###################
###CROSS_SELLING: Creates a multiclass classification model using a
neural network algorithm

#res_azure <- read.csv("res_azure.csv", sep=";", stringsAsFactors =
FALSE)
#testdata <- res_azure

#nn <- neuralnet(res_azure, data=testdata, hidden=c(2,1),
linear.output=FALSE, threshold=0.01)
#nn$result.matrix
#plot(nn)
```

E.1.2 Parse.R

```
## This script is designed for pre-processing the cleaned data from
Parse.R ## This script is for data cleansing of the raw dataset
## and storing all adjusted datasets under the suffix "_adap"

install.packages("dplyr")
install.packages("lubridate")

library(dplyr)
library(lubridate)

setwd("C:/Users/hakim.harrach/Documents/UEL/PhD/3-Doctoral Research/R-
Project/Research_Transactional_Behaviour/Dataset/final")

GetBirthDate <- function(d){
  d <- toString(d)
  month <- substr(d,3,4) %>% as.numeric()
  year <- substr(d,1,2) %>% as.numeric()
  year <- year + 1900

  if (month <= 12){
    tmp <- paste(year, month, substr(d,5,6))
  } else{
    month <- month - 50
    if (month < 10) {
      tmp <- paste(year, "0",month, substr(d,5,6),sep="")
    } else {
      tmp <- paste(year, month, substr(d,5,6),sep="")
    }
  }
  return(tmp)
}

GetSex <- function(d){
  d <- toString(d)
  month <- substr(d,3,4) %>% as.numeric()

  if (month <= 12){
    return("male")
  } else{
    return("female")
  }
}

trans <- read.csv("trans.csv", sep=";", stringsAsFactors = FALSE)
trans$date <- as.character(trans$date)
trans$date <- as.Date(trans$date, format="%y%m%d")

trans <- trans %>% mutate(type = ifelse(type == "PRIJEM",
                                                 "credit", type))
trans <- trans %>% mutate(type = ifelse(type == "VYDAJ",
                                        "withdrawal", type))
trans <- trans %>% mutate(type = ifelse(type == "VYBER",
                                        "withdrawal", type))

trans <- trans %>% mutate(operation = ifelse(operation == "VYBER KARTOU",
                                        "credit card withdrawal",
operation))
trans <- trans %>% mutate(operation = ifelse(operation == "VKLAD",
                                             "credit in cash",
operation))
trans <- trans %>% mutate(operation = ifelse(operation == "PREVOD Z
UCTU",
                                             "collection from another
bank", operation))
trans <- trans %>% mutate(operation = ifelse(operation == "VYBER",
                                             "withdrawal in cash",
operation))
trans <- trans %>% mutate(operation = ifelse(operation == "PREVOD NA
UCET",
                                             "remittance to another
bank", operation))

trans <- trans %>% mutate(k_symbol = ifelse(k_symbol == "POJISTNE",
                                                  "insurance payment",
k_symbol))
trans <- trans %>% mutate(k_symbol = ifelse(k_symbol == "SIPO",
                                                  "household", k_symbol))
```

```
trans <- trans %>% mutate(k_symbol = ifelse(k_symbol == "SLUZBY",
                                                  "payment for
statement", k_symbol))
trans <- trans %>% mutate(k_symbol = ifelse(k_symbol == "UVER",
                                                  "loan payment",
k_symbol))
trans <- trans %>% mutate(k_symbol = ifelse(k_symbol == "UROK",
                                            "interest credited",
k_symbol))
trans <- trans %>% mutate(k_symbol = ifelse(k_symbol == "SANKC. UROK",
                                            "interest if negative
balance", k_symbol))
trans <- trans %>% mutate(k_symbol = ifelse(k_symbol == "DUCHOD",
                                            "old-age pension", k_symbol))

trans$index <- 1:nrow(trans)
write.csv2(trans %>% filter(index < (nrow(trans)/2)), "trans1.csv")
write.csv2(trans %>% filter(index >= (nrow(trans)/2)), "trans2.csv")
write.csv2(trans , "trans_complete.csv", row.names = F)
str(trans)

account <- read.csv("account.csv", sep=";", stringsAsFactors = FALSE)
account$date <- as.character(account$date)
account$date <- as.Date(account$date, format="%y%m%d")

account <- account %>% mutate(frequency = ifelse(frequency == "POPLATEK
MESICNE",
                                      "monthly issuance", frequency))
account <- account %>% mutate(frequency = ifelse(frequency == "POPLATEK
TYDNE",
                                      "weekly issuance", frequency))
account <- account %>% mutate(frequency = ifelse(frequency == "POPLATEK
PO OBRATU",
                                      "issuance after transaction",
frequency))
write.csv2(account, "account_adap.csv", row.names = F)
str(account)

card <- read.csv("card.csv", sep=";", stringsAsFactors = FALSE)
card$issued <- as.character(card$issued)
card <- card %>% mutate(issued = substr(issued,1,6))
card$issued <- as.Date(card$issued, format="%y%m%d")
write.csv2(card,"card_adap.csv", row.names = F)
str(card)

client <- read.csv("client.csv", sep=";")
client$sex <- mapply(GetSex, client$birth_number)
client$birthdate <- mapply(GetBirthDate, client$birth_number)
client$birthdate <- as.Date(client$birthdate, format ="%Y%m%d")
client <- client %>% mutate(Age = 1998 - year(birthdate))
write.csv2(client,"client_adap.csv", row.names = F)
str(client)

disp <- read.csv("disp.csv", sep=";")
str(disp)

district <- read.csv("district.csv", sep=";", stringsAsFactors = FALSE)
colnames(district) <- c("district_id", "district_name", "region",
"NoInhabitants",
                        "NoMunicipalities_0_499",
"NoMunicipalities_500_1999",
                        "NoMunicipalities_2000_9999",
"NoMunicipalities_10000",
                        "NoCities", "UrbanRatio", "AverageSalary",
"Unemployment95",
                        "Unemployment96", "NoEnterpreneurs",
"CommitedCrimes95",
                        "CommitedCrimes96")
district <- district %>% mutate(Unemployment95 =
ifelse(Unemployment95=="?",
                                                         NA,
Unemployment95))
district$Unemployment95 <- as.numeric(district$Unemployment95)
district <- district %>% mutate(CommitedCrimes95 =
ifelse(CommitedCrimes95=="?",
                                                         NA,
CommitedCrimes95))
district$CommitedCrimes95 <- as.integer(district$CommitedCrimes95)
write.csv2(district, "district_adap.csv", row.names = F)
str(district)

loan <- read.csv("loan.csv", sep=";")
loan$date <- as.character(loan$date)
loan$date <- as.Date(loan$date, format="%y%m%d")
write.csv2(loan,"loan_adap.csv", row.names = F)
str(loan)

order_df <- read.csv("order.csv", sep=";", stringsAsFactors = FALSE)
order_df <- order_df %>% mutate(k_symbol = ifelse(k_symbol == "POJISTNE",
                                      "insurance payment", k_symbol))
order_df <- order_df %>% mutate(k_symbol = ifelse(k_symbol == "SIPO",
                                                  "household payment",
k_symbol))
order_df <- order_df %>% mutate(k_symbol = ifelse(k_symbol == "LEASING",
                                                  "leasing", k_symbol))
order_df <- order_df %>% mutate(k_symbol = ifelse(k_symbol == "UVER",
                                                  "loan payment",
k_symbol))

write.csv2(order_df,"order_adap.csv", row.names = F)
str (order_df)

#############
# Data description of the processed data in python
#############
setwd("C:/Users/hakim.harrach/Documents/Personal/UEL/PhD/3-Doctoral
Research/Py-Project/Scripts/dataProcessed")
dataTrans <- read.csv("dataTrans.csv", sep=";", stringsAsFactors = FALSE)
names(dataTrans) <- c("type of transaction","mode of transaction","amount
bucket","balance bucket","bank of partner","transaction category")

str(dataTrans)
summary(dataTrans)

#Visualize boxplots

boxplot(dataTrans)
```

E.1.3 Parse4BasketAnalyze.R

```
#install and load package arules
install.packages("arules")
library(arules)
#install and load arulesViz
install.packages("arulesViz")
library(arulesViz)
#install and load tidyverse
install.packages("tidyverse")
library(tidyverse)
#install and load readxml
install.packages("readxml")
library(readxl)
#install and load knitr
install.packages("knitr")
library(knitr)
#load ggplot2 as it comes in tidyverse
library(ggplot2)
#install and load lubridate
install.packages("lubridate")
library(lubridate)
#install and load plyr
install.packages("plyr")
library(plyr)
library(dplyr)

setwd("C:/Users/hakim.harrach/Documents/UEL/PhD/3-Doctoral Research/R-
Project/Research_Transactional_Behaviour/Dataset/final")

##Data Pre-processing
trans <- read.csv("trans.csv", sep=";", stringsAsFactors = FALSE)
str(trans)

trans$date <- as.character(trans$date)
trans$date <- as.Date(trans$date, format="%y%m%d")

#install.packages("sqldf")
library(sqldf)
DF <- data.frame(sqldf('select account_id, date, k_symbol from trans
where date < "31.01.1998" group by account_id'))

#set columns of dataframe transactionData
trans$trans_id <- NULL
trans$amount <- NULL
trans$balance <- NULL
trans$bank <- NULL
trans$account <- NULL
trans$type <- NULL
trans$operation <- NULL

library(plyr)
ddply(dataframe, variables_to_be_used_to_split_data_frame,
function_to_be_applied)
#transactionData <- ddply(trans,c("account_id","date"),
#                    function(df1)paste(df1$k_symbol,
#                                         collapse = ","))
str(transactionData)
# combine all trans from that account_id and k_symbol as one row, with
each item, separated by ,
transactionData_new <- ddply(trans,c("account_id"),
                         function(df1)paste(df1$k_symbol,
                                            collapse = ","))
write.csv(transactionData,"C:/Users/hakim.harrach/Documents/Personal/UEL/
PhD/3-Doctoral Research/R-
Project/Research_Transactional_Behaviour/Dataset/final/basket_transaction
s.csv", quote = FALSE, row.names = TRUE)
#write.csv(transactionData,"C:/Users/hakim.harrach/Documents/Personal/UEL
/PhD/3-Doctoral Research/R-
Project/Research_Transactional_Behaviour/Dataset/final/basket_transaction
s_new.csv", quote = FALSE, row.names = TRUE)

transactionData <- read.csv("basket_transactions.csv")
#transactionData_new <- read.csv("basket_transactions_new.csv")

#transactionData1 <- read.csv("basket_transactions.csv")
#set column account_id of dataframe transactionData
#transactionData_new$account_id <- NULL
#set column Date of dataframe transactionData
transactionData$date <- NULL
transactionData$X <- NULL
write.csv(transactionData,"C:/Users/hakim.harrach/Documents/Personal/UEL/
PhD/3-Doctoral Research/R-
Project/Research_Transactional_Behaviour/Dataset/final/baskets4sequences.
txt", quote = FALSE, row.names = TRUE)

#Rename column to items
colnames(transactionData_new) <- c("transaction_id", "items")
#colnames(transactionData_new) <- c("items")
#colnames(transactionData) <- c("items")
#Show Dataframe transactionData
str(transactionData)
# read transactions (using arules package) which is in basket format an
convert into an object of the transaction class
tr <- read.transactions('C:/Users/hakim.harrach/Documents/UEL/PhD/3-
Doctoral Research/R-
Project/Research_Transactional_Behaviour/Dataset/final/basket_transaction
s_new.csv', format = 'basket', sep =',')
str(tr)
# convert into transaction object
trObj<-as(dataframe.dat,"transactions")

summary(tr)

#Create an item frequency plot for the top 10 items
if (!require("RColorBrewer")) {
  # install color package of R
  install.packages("RColorBrewer")
  #include library RColorBrewer
  library(RColorBrewer)
}
itemFrequencyPlot(tr,topN=10,type="absolute",col=brewer.pal(8,'Pastel2'),
main="Absolute Item Frequency Plot")

itemFrequencyPlot(tr,topN=10,type="relative",col=brewer.pal(8,'Pastel2'),
main="Relative Item Frequency Plot")

#Generating Rules!
# Next step is to mine the rules using the APRIORI algorithm. The
function apriori() is from package arules

# Min Support as 0.001, confidence as 0.8.
association.rules <- apriori(tr, parameter = list(supp=0.001,
conf=0.8,maxlen=10))
```

```
summary(association.rules)

inspect(association.rules[1:4])

#### Visualizing Association Rules
# Filter rules with confidence greater than 0.8 or 80%
subRules<-association.rules[quality(association.rules)$confidence>0.8]
#Plot SubRules
library(arulesViz)
plot(subRules)

plot(subRules,method="two-key plot")

# Graph-Based Visualizations
top10subRules <- head(subRules, n = 10, by = "confidence")
plot(top10subRules, method = "graph",  engine = "htmlwidget")
inspectDT(top10subRules)

# Filter top 10 rules with highest lift
top10subRules<-head(subRules, n=10, by="lift")
library(arulesViz)
plot(top10subRules, method="paracoord")

# Matrix3D
plot(top10subRules, method = "matrix3D", measure = "lift")
# Grouped-matrix
plot(top10subRules, method = "grouped")
# Graphs
plot(top10subRules, method = "graph")

################
install.packages("arulesSequences")
library(arulesSequences)

##TEST example
---
frequent_pattern <- cspade(tr, parameter = list(support= 0.50))
inspect(frequent_pattern)
summary(frequent_pattern)
as(frequent_pattern, "data.frame")

###############
install.packages("arules")
install.packages("arulesSequences")

library(Matrix)
library(arules)
library(arulesSequences)

install.packages("sqldf")
library(sqldf)

#Data Preprocessing
transactionData <- read.csv("basket_transactions.csv")
sqldf('select account_id, V1 from transactionData where V1 is not Null
group by account_id')

DF <- data.frame(sqldf('select account_id, group_concat (V1) from
transactionData group by account_id'))
colnames(DF) <- c("AccountID", "ItemSequence")
```

```
write.csv(DF,"C:/Users/hakim.harrach/Documents/UEL/PhD/3-Doctoral
Research/R-
Project/Research_Transactional_Behaviour/Dataset/final/baskets4sequences.
csv", quote = FALSE, row.names = TRUE)

# Data Cleansing for DF
DF_clean <- data.frame(sqldf('select AccountID, Replace (ItemSequence,
",","," ") from DF'))
str(DF_clean)
colnames(DF_clean) <- c("AccountID", "ItemSequence")
write.csv(DF_clean,"C:/Users/hakim.harrach/Documents/UEL/PhD/3-Doctoral
Research/R-
Project/Research_Transactional_Behaviour/Dataset/final/baskets4sequences.
csv", quote = FALSE, row.names = TRUE)

# Zwischentabelle 1:5
DF_clean_tmp <- data.frame(sqldf('select AccountID, ItemSequence from
DF_clean where AccountID > "1000" '))
str(DF_clean_tmp)
#split the data using the str_split function from package stringr
# install.packages("stringr")
library("stringr")
data_for_fseq_mining <- str_split(string = DF_clean_tmp$ItemSequence,
pattern = " ")

#prerequisite for using the function 'as.transactions'
names(data_for_fseq_mining) <- DF_clean_tmp$AccountID

#convert this kind of data to a dataset of class 'transactions'
install.packages("clickstream")
library(clickstream)

data_for_fseq_mining_trans <- as.transactions(clickstreamList =
data_for_fseq_mining)

#install.packages("arulesSequences")
library(arulesSequences)
# data is in  proper format to run the cspade-algorithm with some
parameters
sequences <- cspade(data      = data_for_fseq_mining_trans,
                    parameter = list(support = 0.5, maxsize = 10, maxlen
= 10, mingap = 1, maxgap = 10),
                    control   = list(tidList = TRUE, verbose = TRUE))

#Summarizing the results (sequence and relative support)
sequences_df <- cbind(sequence = labels(sequences), support =
sequences@quality)
summary(sequences_df)

#whether each sequence is present or not (TRUE/FALSE)
sequences_score <- as.matrix(sequences@tidLists@data)

## Visualization
# We can use the command summary and as to see the results:
cspade> summary(sequences)
##############
# execute the CSPADE algorithm
s1 <- cspade(x, parameter = list(support = 0.4), control = list(verbose =
TRUE))

################
## Predicting
## recommendar engine - Create a Recommender Model
# more precisely by using the frequency items - predicting the arules
sequences

#Here we can look at the frequent itemsets and we can use the eclat
algorithm rather than the apriori algorithm.
#itemFrequencyPlot(Adult, support = 0.1, cex.names=0.8);

fsets = eclat(tr, parameter = list(support = 0.05), control =
list(verbose=FALSE));

singleItems = fsets[size(items(fsets)) == 1];

singleSupport = quality(singleItems)$support;

names(singleSupport) = unlist(LIST(items(singleItems), decode = FALSE));

head(singleSupport, n = 5);

itemsetList = LIST(items(fsets), decode = FALSE);

allConfidence = quality(fsets)$support / sapply(itemsetList, function(x)

  max(singleSupport[as.character(x)]));

quality(fsets) = cbind(quality(fsets), allConfidence);


summary(fsets)
```

E.1.4 Parse4NeuralNetworkAnalyze.R

```
# Install reticulate package
install.packages("reticulate")

# Load reticulate package
library(reticulate)

##Data Pre-processing in Python

py_available()

conda_create("r-reticulate")

conda_install("r-reticulate","numpy")

## Executing py script in Spyder Notebook from Anaconda
dataProcessing.py

##########--------#########--------##########

# We firstly set our directory and load the data into the R environment
setwd("C:/Users/hakim.harrach/Documents/UEL/PhD/3-Doctoral Research/R-
Project/Research_Transactional_Behaviour/Dataset/neuralnetwork")

mydataTrans <- read.csv("dataTrans.csv", sep=";", stringsAsFactors =
FALSE)
#mydataNamesTrans <- read.csv("namesTrans.csv")
```

```
attach(mydataTrans)

colnames(mydataTrans) <-
c("TransType","TransMode","AmountBucket","BalanceBucket", "BankPartner",
"TransCharacterization")

##### Exploring and understanding data (EDA) --------------------
## Visualizing and plot the data

# Visualizing the data with simple bar plot
counts <- table(mydataTrans$TransType)
barplot(counts, main="Bar plot",
        col = "lightblue",xlab="Scale of TransType",
        ylab = "Frequencies",
        names.arg=c("(x)"))

# Visualizing the data with simple Histogram
TransType <- c(mydataTrans$TransType)
hist(TransType)

# Visualizing the data with Boxplots
boxplot(mydataTrans$TransType,
        col = "lightblue",
        main="Boxplot of the TransType",
        names=c("x1"),
        xlab="Amount", ylab="Value")
points(rep(1, length(mydataTrans$TransType, pch = 4, cex = 2)))

##
# Visualizing the data with simple bar plot
counts <- table(mydataTrans$TransMode)
barplot(counts, main="Bar plot",
        col = "lightblue",xlab="Scale of TransMode",
        ylab = "Frequencies",
        names.arg=c("(x)"))

# Visualizing the data with simple Histogram
TransMode <- c(mydataTrans$TransMode)
hist(TransMode)

# Visualizing the data with Boxplots
boxplot(mydataTrans$TransMode,
        col = "lightblue",
        main="Boxplot of the TransMode",
        names=c("x1"),
        xlab="Amount", ylab="Value")
points(rep(1, length(mydataTrans$TransMode, pch = 4, cex = 2)))

##
# Visualizing the data with simple bar plot
counts <- table(mydataTrans$AmountBucket)
barplot(counts, main="Bar plot",
        col = "lightblue",xlab="Scale of AmountBucket",
        ylab = "Frequencies",
        names.arg=c("(x)"))

# Visualizing the data with simple Histogram
AmountBucket <- c(mydataTrans$AmountBucket)
hist(AmountBucket)

# Visualizing the data with Boxplots
boxplot(mydataTrans$AmountBucket,
        col = "lightblue",
        main="Boxplot of the AmountBucket",
        names=c("x1"),
        xlab="Amount", ylab="Value")
points(rep(1, length(mydataTrans$AmountBucket, pch = 4, cex = 2)))

##
# Visualizing the data with simple bar plot
counts <- table(mydataTrans$BalanceBucket)
barplot(counts, main="Bar plot",
        col = "lightblue",xlab="Scale of BalanceBucket",
        ylab = "Frequencies",
        names.arg=c("(x)"))

# Visualizing the data with simple Histogram
BalanceBucket <- c(mydataTrans$BalanceBucket)
hist(BalanceBucket)

# Visualizing the data with Boxplots
boxplot(mydataTrans$BalanceBucket,
        col = "lightblue",
        main="Boxplot of the BalanceBucket",
        names=c("x1"),
        xlab="Amount", ylab="Value")
points(rep(1, length(mydataTrans$BalanceBucket, pch = 4, cex = 2)))

##
# Visualizing the data with simple bar plot
counts <- table(mydataTrans$BankPartner)
barplot(counts, main="Bar plot",
        col = "lightblue",xlab="Scale of BankPartner",
        ylab = "Frequencies",
        names.arg=c("(x)"))

# Visualizing the data with simple Histogram
BankPartner <- c(mydataTrans$BankPartner)
hist(BankPartner)

# Visualizing the data with Boxplots
boxplot(mydataTrans$BankPartner,
        col = "lightblue",
        main="Boxplot of the BankPartner",
        names=c("x1"),
        xlab="Amount", ylab="Value")
points(rep(1, length(mydataTrans$BankPartner, pch = 4, cex = 2)))

##
# Visualizing the data with simple bar plot
counts <- table(mydataTrans$TransCharacterization)
barplot(counts, main="Bar plot",
        col = "lightblue",xlab="Scale of TransCharacterization",
        ylab = "Frequencies",
        names.arg=c("(x)"))

# Visualizing the data with simple Histogram
TransCharacterization <- c(mydataTrans$TransCharacterization)
hist(TransCharacterization)

# Visualizing the data with Boxplots
boxplot(mydataTrans$TransCharacterization,
        col = "lightblue",
        main="Boxplot of the TransCharacterization",
        names=c("x1"),
        xlab="Amount", ylab="Value")
points(rep(1, length(mydataTrans$TransCharacterization, pch = 4, cex =
2)))

## Steps for in constructing the model
# Data Normalization to compare accurately predicted and actual values
# scaleddata<-scale(mydataTrans)
# Max-Min Normalization
normalize <- function(x) {
  return ((x - min(x)) / (max(x) - min(x)))
}

maxmindf <- as.data.frame(lapply(mydataTrans, normalize))

# Base training data (trainset) on 80% of the observations.
# The test data (testset) is based on the remaining 20% of observations.

# Training and Test Data
trainset <- maxmindf[1:416805, ]
testset <- maxmindf[416806:521006, ]

### Preparing various buckets for smaller sized research experiments due
to limited computational power
## Training and Test Data
# Bucket:1
trainset <- maxmindf[1:10000, ]
testset <- maxmindf[10001:12000, ]
# Bucket:2
trainset <- maxmindf[45000:55000, ]
testset <- maxmindf[55000:57000, ]
# Bucket:3
trainset <- maxmindf[100000:110000, ]
testset <- maxmindf[110001:112000, ]
# Bucket:4
trainset <- maxmindf[145000:155000, ]
testset <- maxmindf[155001:157000, ]
# Bucket:5
trainset <- maxmindf[200000:210000, ]
testset <- maxmindf[210001:212000, ]
# Bucket:6
trainset <- maxmindf[245000:255000, ]
testset <- maxmindf[255001:257000, ]
# Bucket:7
trainset <- maxmindf[300000:310000, ]
testset <- maxmindf[310001:312000, ]
# Bucket:8
trainset <- maxmindf[345000:355000, ]
testset <- maxmindf[355001:357000, ]
# Bucket:9
trainset <- maxmindf[400000:410000, ]
testset <- maxmindf[410001:412000, ]
# Bucket:10
trainset <- maxmindf[445000:455000, ]
testset <- maxmindf[455001:457000, ]
# Bucket:11
trainset <- maxmindf[500000:510000, ]
testset <- maxmindf[510001:512000, ]
# Training a Neural Network Model using neuralnet from R

#Neural Network
install.packages("neuralnet")
```

```
library(neuralnet)
nn <- neuralnet(TransCharacterization ~ TransMode + AmountBucket +
BalanceBucket + BankPartner + TransType, data=trainset, hidden=c(2,1),
linear.output=FALSE, threshold=0.01, stepmax = 10000000)
nn$result.matrix
plot(nn)

#Test the resulting output
temp_test <- subset(testset, select =
c("TransType","TransMode","AmountBucket","BalanceBucket","BankPartner"))
head(temp_test)
nn.results <- compute(nn, temp_test)

# The predicted results are compared to the actual results
results <- data.frame(actual = testset$TransCharacterization, prediction
= nn.results$net)

# round up the results using Confusion Matrix to compare the number of
true/false positives and negatives
roundedresults<-sapply(results,round,digits=0)
roundedresultsdf=data.frame(roundedresults)
attach(roundedresultsdf)
table(roundedresultsdf)

# Calculating the accuracy
##predicted=results$prediction * abs(diff(range(TransCharacterization)))
+ min(TransCharacterization)
##actual=results$actual * abs(diff(range(TransCharacterization))) +
min(TransCharacterization)
MSE.neuralnetModel  <- sum((results$actual -
results$prediction)^2)/nrow(testset)
MSE.neuralnetModel
plot(results$actual, results$prediction, col='red',main='Actual vs
predicted',pch=18,cex=0.7)
abline(0,1,lwd=2)
legend('bottomright',legend='NN',pch=18,col='red',  bty='n')

#predicted=results$prediction
#actual=results$actual
#comparison=data.frame(predicted,actual)
#deviation=((actual-predicted)/actual)
#comparison=data.frame(predicted,actual,deviation)
#accuracy=1-abs(mean(deviation))
#accuracy

#       prediction
#actual   0      1
#    0   36410   5457
#    1   0       62334

# The model generates 36410 true negatives (0's),
# 62334 true positives (1's), while there are 0 false negatives and
5457 ...
# Outcome: The model yield an 94.763% (98.744/104.201) accuracy rate in
determining whether a transaction type is known or not.

# The plot below displays the result of the loss function as the model is
trained.
# The objective of the model training is to minimize the result of the
loss function (Y axis),
# and we can see that as we progress in iterations (X axis), the result
of the loss function
```

```
# approaches zero.
# plot the cost
library(ggplot2)
# ggplot(data = results, aes(x = iteration, y = loss))
# +  geom_line()
```

E.2. Python code

E.2.1 Python-Script of dataPreprocessing.py

```
import pandas as pd
import numpy as np
import collections
import matplotlib.pyplot as plt
from textwrap import wrap
import pickle

data = pd.read_csv('data/trans.asc',';',header=1,
low_memory=False).values

counter = collections.Counter(data[:,7])
useableData = data[~pd.isnull(data[:,7]),:]
useableData = useableData[useableData[:,7] != " ", :]

indicesX = [3,4,5,6,8] #[1,3,4,5,6,8,9]

'''
X legend:
0 => type    +/- transaction    "PRIJEM" stands for credit, "VYDAJ"
stands for withdrawal
1 => operation    mode of transaction    {VYBER KARTOU, VKLAD,PREVOD Z
UCTU, VYBER, PREVOD NA UCET}
2 => amount of money
3 => balance    balance after transaction
4 => bank    bank of the partner    each bank has unique two-letter code
# 5 => account    account of the partner
'''
X = useableData[:, indicesX]
Y = useableData[:, 7]
numberUniques = np.empty(len(indicesX))
for i in range(len(numberUniques)):
    numberUniques[i] = len(set(X[:,i]))

maxNumberUniques = max(numberUniques)

dictTypeTranslation = dict(zip(['PRIJEM', 'VYDAJ'], ['credit',
'withdrawal']))
dictOperationTranslation = dict(zip(['VYBER KARTOU','VKLAD', 'PREVOD Z
UCTU','VYBER','PREVOD NA UCET'], ['credit card withdrawal', 'credit in
cash','collection from another bank', 'withdrawal in cash','remittance to
another bank']))
dictK_symbolTranslation = dict(zip(['POJISTNE', 'SLUZBY','UROK','SANKC.
UROK', 'SIPO','DUCHOD','UVER'], ['insur. payment','payment for
statement', 'interest credited', 'sanction interest if negative balance',
'household', 'old-age pension', 'loan payment']))

for i in range(X.shape[0]):
    X[i,0] = dictTypeTranslation.get(X[i,0])
    X[i,1] = dictOperationTranslation.get(X[i,1])
    Y[i] = dictK_symbolTranslation.get(Y[i])
```

```
X_orig = X.copy()
Y_orig = Y.copy()
'''
==============================================================
create dicts to change values of type, operation and partner bank to
floats
'''
dictType =      dict(zip(list(set(X[:,0])),
range(int(numberUniques[0]))))
dictOperation = dict(zip(list(set(X[:,1])),
range(int(numberUniques[1]))))
dictPartnerBank = dict(zip(list(set(X[:,4])),
range(int(numberUniques[4]))))
dictK_symbol = dict(zip(list(set(Y)), range(int(len(set(Y))))))

'''
==============================================================
Create Bins for amount and balance data
'''
numberBins = 10
binsAmount, binEdgesAmount = np.histogram(X[:, 2], bins=numberBins)
binsBalance, binEdgesBalance = np.histogram(X[:, 3], bins=numberBins)
for i in range(X.shape[0]):
    currentX2 = 0
    currentX3 = 0
    for j in range(numberBins):
        if X[i,2] >= binEdgesAmount[j]:
            currentX2 = j
        if X[i,3] >= binEdgesBalance[j]:
            currentX3 = j
    X[i,2] = currentX2
    X[i,3] = currentX3
    X[i,0] = dictType.get(X[i,0])
    X[i,1] = dictOperation.get(X[i,1])
    X[i,4] = dictPartnerBank.get(X[i,4])
    Y[i] = dictK_symbol.get(Y[i])

for i in range(X.shape[1]):
    counter = collections.Counter(X[:,i])

'''
==============================================================
visualization
'''
feature1 = collections.Counter(X_orig[:,0])
feature2 = collections.Counter(X_orig[:,1])
feature3 = collections.Counter(X[:,2])
feature4 = collections.Counter(X[:,3])
feature5 = collections.Counter(X_orig[:,4])
labels = collections.Counter(Y_orig)
# print(list(feature1))
keys1 = list(feature1.keys())
values1 = list(feature1.values())
keys2 = list(feature2.keys())
values2 = list(feature2.values())
keys3 = list(feature3.keys())
values3 = list(feature3.values())
keys4 = list(feature4.keys())
values4 = list(feature4.values())
keys5 = list(feature5.keys())
values5 = list(feature5.values())
```

```
keysLabels = list(labels.keys())
valuesLabels = list(labels.values())
listKeys2 = [x if x is not None else "None" for x in keys2]
listKeys5 = [str(x) for x in keys5]
listKeysLabels= [x if x is not None else "None" for x in keysLabels]

categories = np.array([keys1, listKeys2, keys3, keys4, listKeys5,
listKeysLabels], dtype=object)
print(categories.shape)
print(categories)

listKeys2 = [ '\n'.join(wrap(l, 15)) for l in listKeys2]
listKeys5 = [ '\n'.join(wrap(l, 15)) for l in listKeys5]
listKeysLabels = [ '\n'.join(wrap(l, 11)) for l in listKeysLabels]

'''
=============================================================
plot data
'''
fig = plt.figure()
titles =['type','operation','amount','balance','bank of partner',
'transaction types']
n = 6  # num sub-plots
fig.add_subplot(2, 3, 1)
for i in range (2,n+1):
    fig.add_subplot(2, 3, i)
fig.axes[0].bar(keys1, values1)
fig.axes[1].bar(listKeys2, values2)
fig.axes[2].bar(keys3, values3)
fig.axes[3].bar(keys4, values4)
fig.axes[4].bar(listKeys5, values5)
fig.axes[5].bar(listKeysLabels, valuesLabels)
for i in range(n):
    fig.axes[i].title.set_text(titles[i])
# fig.axes[0].set_xticklabels(dictTypeTranslation.values())

for tick in fig.axes[1].get_xticklabels():
    tick.set_rotation(0)
    tick.set_fontsize(8)
for tick in fig.axes[5].get_xticklabels():
    tick.set_rotation(90)
    tick.set_fontsize(8)

plt.show()

'''
=============================================================
export data to csv file (csv-file is only saved after closing the plot)
'''
fname = './dataProcessed/dataTrans.csv'
headername = 'Type transaction;mode of transaction;amount bucket;balance
bucket;bank of partner;label: characterization of the transaction'
printMatrix = np.column_stack((X,Y))
np.savetxt(fname, printMatrix, fmt='%s', delimiter=';',
header=headername)
# headername = 'Type transaction;mode of transaction;amount
bucket;balance bucket;bank of partner;label: characterization of the
transaction'
# np.savetxt(fname, categories, delimiter=';', header=headername)
fname = './dataProcessed/namesTrans.csv'

with open(fname, "w") as f:
    for lines in categories:
        f.write(str(lines) +"\n")

# df = pd.DataFrame(categories)
# print(df)
# df.to_csv(fname,  header=['0'])
print("\ndone")
```

E.2.2 classificationNN_TrainAndSave.py

```
import tensorflow as tf
import pandas as pd
from tensorflow import keras
import numpy as np
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
from sklearn.preprocessing import OneHotEncoder
from itertools import chain
# import keras

nodesFirstLayer = 40
nodesSecondLayer = 20

fname = './dataProcessed/dataTrans.csv'
data = np.loadtxt(fname, delimiter=';', skiprows=1)
print('data excerpt:\n', data[0:2,:], '\n')

fname = './dataProcessed/namesTrans.csv'
array = []
with open(fname, "r") as f:
    for line in f:
        array.append(eval(line.strip()))

print('excerpt list of categories\n', array[:2], '\n')

featureElements = list(chain.from_iterable(array[:len(array)-1]))
labelElements = array[len(array)-1]
print('number of input nodes: ' + str(len(featureElements)))
print('number of output nodes: ' + str(len(labelElements)))

enc = OneHotEncoder(handle_unknown='ignore')
enc.fit(data)
dataOneHot = enc.transform(data).toarray()

X = dataOneHot[:,:len(featureElements)]#TODO
Y = dataOneHot[:,len(featureElements):]
print('shape of X: ', X.shape)
print('shape of Y: ', Y.shape)
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.20,
random_state=42)

print('shape of X_train: ', X_train.shape)
print('shape of X_test: ', X_test.shape)
print('shape of Y_train: ', Y_train.shape)
print('shape of Y_test: ', Y_test.shape)
model = keras.Sequential([

keras.layers.Dense(nodesFirstLayer,input_shape=(len(featureElements),),
activation=tf.nn.relu),
    keras.layers.Dense(nodesSecondLayer, activation=tf.nn.relu),
    keras.layers.Dense(len(labelElements), activation=tf.nn.softmax)
])

# Compile model
model.compile(loss='binary_crossentropy', optimizer='adam',
metrics=['accuracy'])
# Fit the model
model.fit(X_train, Y_train, epochs=1, batch_size=10)
#
model.save('model_TransactionClassifier_' + str(nodesFirstLayer) + '_' +
str(nodesSecondLayer))
# keras.models.load_model('model_TransactionClassifier_40_20')

# evaluate the model
scores = model.evaluate(X_test, Y_test)
predictions = model.predict(X_test)
print("\n%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))
print("prediction: ", predictions[0])
print("label: ", Y_test[0])
```

E.2.3 classificationNN_loaded.py

```
import tensorflow as tf
import pandas as pd
from tensorflow import keras
import numpy as np
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
from sklearn.preprocessing import OneHotEncoder
from itertools import chain
#c import keras

nodesFirstLayer = 40
nodesSecondLayer = 20

fname = './dataProcessed/dataTrans.csv'
data = np.loadtxt(fname, delimiter=';', skiprows=1)
print('data excerpt:\n', data[0:2,:], '\n')

fname = './dataProcessed/namesTrans.csv'
array = []
with open(fname, "r") as f:
    for line in f:
        array.append(eval(line.strip()))

print('excerpt list of categories\n', array[:2], '\n')

featureElements = list(chain.from_iterable(array[:len(array)-1]))
labelElements = array[len(array)-1]

enc = OneHotEncoder(handle_unknown='ignore')
enc.fit(data)
dataOneHot = enc.transform(data).toarray()
print(dataOneHot.shape)

X = dataOneHot[:,:len(featureElements)]#TODO
Y = dataOneHot[:,len(featureElements):]
print('shape of X: ', X.shape)
print('shape of Y: ', Y.shape)
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.20,
random_state=42)

print('shape of X_train: ', X_train.shape)
print('shape of X_test: ', X_test.shape)
print('shape of Y_train: ', Y_train.shape)
print('shape of Y_test: ', Y_test.shape)

model = keras.models.load_model('model_TransactionClassifier_40_20')

# evaluate the model
scores = model.evaluate(X_test, Y_test)
predictions = model.predict(X_test)
print("\n%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))
print("prediction: ", predictions[0])
print("label: ", Y_test[0])
```