NTNU
Norwegian University of
Science and Technology

Early identification of high-risk credit card customers based on behavioral data

Morten Hansen Flood

Master of Science in Computer Science
Submission date: July 2017
Supervisor: Mads Nygård, IDI

Norwegian University of Science and Technology
Department of Computer Science

Abstract

Credit card banking has for a long time been one of the most profitable types of banking. The largest cost for credit card companies is customers not paying their debt. Consequently, to accurately model the risk a customer poses can provide large savings for credit card companies.

This thesis aims to determine if it is possible to identify high risk credit card customers within the first months of the customer relationship. Using a credit card dataset consisting of customers' first 18 months of data from between January 2013 and April 2017, machine learning methods are used to develop classifiers that try to predict future delinquency. Where previous work has incorporated many months of data to predict delinquency, we use only data from the first and second month of the customer relationship to do the same.

Through a number of experiments, several models are developed. In addition to predicting delinquency, the models are used to analyze behavior driving delinquency and to model credit risk.

We find that the models can not accurately identify high risk customers based on only a few months of data. The models developed reveal that the factors driving delinquency are mostly intuitive. Using the developed models to predict the probability of delinquencies, we find a strong correlation between the predicted probabilities and realized frequencies of delinquency.

Sammendrag

Kredittkortvirksomhet har lenge vært en av de mest profitable typene bankvirksomhet. Den største kostnaden for kredittkortselskaper er kunder som ikke tilbakebetaler gjelden sin. Som følge av dette vil det å kunne presist modellere risikoen en kunde utgjør kunne spare kredittkortselskaper for mye penger.

Denne oppgaven prøver å avgjøre om det er mulig å identifisere kredittkortkunder som utgjør en høy riskiko i løpet av de første månedene av kundeforholdet. Vi tar i bruk et kredittkort-datasett bestående av kunders 18 første måneder med data fra januar 2013 til april 2017, og benytter maskinlæringsmetoder til å prediktere fremtidig mislighold. Der tidligere arbeider har brukt mange måneder med data til å predikere mislighold, bruker vi bare den første og andre måneden data fra kundeforholdet til å gjøre det samme.

I løpet av flere eksperimenter utvikler vi atskillige modeller. I tillegg til å predikere mislighold er modellene brukt til å analysere atferd som medvirker til mislighold og til å modellere kredittrisiko.

Vi finner ut at vi ikke kan presist identifisere høyrisikokunder basert på bare noen få måneder med data. De utviklede modellene avslører at faktorene som i hovedsak medvirker til mislighold er intuitive. Ved å bruke de utviklede modellene til å predikere sannsynlighet for mislighold finner vi en sterk korrelasjon mellom predikerte sannsynligheter og reelle hyppigheter av mislighold.

Preface

This thesis is submitted to the Norwegian University of Science and Technology in partial fulfillment of the requirements for a Master of Science degree in Computer Science.

The work was conducted in the spring of 2017 at the Department of Computer and Information Science NTNU and is supervised by Mads Nygård. This thesis is conducted for SpareBank 1 Kredittkort, where Christian Meland has been my external supervisor.

I would like to thank my supervisor Mads Nygård for all the valuable support and guidance. Furthermore, I would like to thank Christian Meland for guidance as well as the opportunity to do this thesis. Finally, I would like to thank Hans Bystrøm at SpareBank 1 for guidance and help.

Table of Contents

List of Tables

List of Figures

Abbreviations

RF	=	Random forest
ROC	=	Receiver operating characteristic
AUC	=	Area under curve
kNN	=	k-nearest neighbor
CV	=	Cross-validation
TP	=	True positive
TN	=	True negative
FP	=	False positive
FN	=	False negative

Chapter 1

Introduction

1.1 The Credit Card Business

In today's consumer economy, credit cards has become a necessity for many consumers. This has made credit card banking one of the most profitable types of banking.

Credit card companies primarily earn their money in three ways. They charge merchants around 2% to 3% of every transaction made using their credit card. They charge customers interest on unpaid balance carried from month to month. And they charge a variety of fees, including annual and late fees. For these reasons, credit card companies earn more money the more customers they have, and are always looking for more people to use their services.

1.2 Risk

With the large potential profit in credit card banking also comes the risk of customers not paying off their credit card balance. As credit card companies seek to expand, it is important that they exercise good risk control. Although customers that carry a balance from month to month expose credit card companies to bigger risk, they can render a larger potential profit with the included interest and late fees. As such, good risk control can provide credit card companies with large savings.

An example of risk control for a credit card company would be to cut or limit customers that are likely to not pay their debt. Doing this they can avoid an increase in the balance that is likely to not be repaid. At the same time, they run the risk of cutting off or limiting customers that will eventually repay their balance, thereby potentially foregoing higher profits in addition to alienating the affected customers. For this to be a viable option, the predictions need to be reasonably accurate.

Due to the sheer amount of data and number of decision involved in the credit card business, it is necessary to rely on algorithms for both decisions as the one above as well as decisions like approving credit card applications.

It is common for credit card companies today to use algorithmic models to assess potential customers creditworthiness to determine if their credit card application should be approved or not. These models can base their decisions on data from the application form, data on the customer the credit card company is already in possession of, data available from credit bureaus, data from the customer's tax records or any combination of these.

1.3 Motivation

SpareBank 1 Kredittkort utilizes a static model like this to determine if an applicant should get a credit card or not. However, a lot of customers that pose a high risk are still approved as evidenced by the many accounts that go to collections every month.

One of SpareBank 1 Kredittkort's risk management strategies is to initially give customers relatively low credit limits and rather increase the credit limits for customers that show the need and ability to handle a larger credit limit. As such, a dynamic behavioral credit risk model that can at an early stage identify customers that pose a higher risk of future delinquency, could be a valuable tool for SpareBank 1 to reduce losses. It is in neither the customers' or SpareBank 1's interest to have customers with a higher credit limit than they can handle.

1.4 Approach

This thesis will address this problem by building predictive models to classify accounts as either high risk or not. The models will be built using several different machine learning algorithms trained on the historical credit card data from SpareBank 1's customers.

More specifically, models are trained on only the first months of customers' data trying to predict delinquency within a specified number of months after.

Machine learning is a sub-field of computer science that has the ability to find patterns, generalize and learn without being explicitly programmed. Machine learning techniques are therefore highly suitable for a problem such as this, and are already often used for building static credit scoring models (Li and Zhong, 2012).

1.5 Research Questions

To further formalize the goal of this thesis, three research questions are formed. The experiments and results presented in this thesis are designed to help answer these questions.

- At what accuracy can a high risk credit card customer be identified within the first months of the customer relationship?
- What early behavior best predicts a high risk customer?
- Can a predictive model be used as a dynamic behavioral model to make decisions regarding existing customers?

1.6 Thesis Structure

This thesis is divided into a total of seven chapters.

- **1 Introduction** introduces the problem tackled in this thesis, as well as the suggested approach to solving it.
- **2 Literature Review** is a literature review of highly relevant papers to the problem in this thesis.
- **3 Dataset** describes the dataset used.
- **4 Basic Theory** goes over the theory behind methods used in this thesis.
- **5 Implementation** describes how the dataset was prepared and how the machine learning models were implemented.
- **6 Experiments** presents the experiments and results.
- **7 Conclusion** discusses the findings from chapter 6 and tries to answer the research questions.

Chapter 2

Literature Review

This chapter serves to highlight and review work relevant to this paper. In it we look at two papers that are highly relevant to this thesis. Both use similar data and methods to predict delinquency among credit card customers. As a result, they both serve as good baseline approaches for this thesis.

This literature review is constrained to only two papers as there are few publications in this domain. There are much literature on credit and risk scoring models, but very few use machine learning methods or use credit card data. One explanation may be the lack of credit card datasets, as such data can't be published given its sensitive nature.

2.1 Risk and Risk Management in the Credit Card Industry

In 2015, Butaru et al. (2016) applied machine-learning methods to credit-card data from six major commercial U.S. banks, combined with credit-bureau and macroeconomic data to predict delinquency. The paper comes from the largest economics research organization in the United States, the National Bureau of Economic Research (NBER), an American, private nonprofit research organization. After the financial crisis of 2007-2009, where the lack of risk management from financial institutions was a big factor in the economic downturn, it became apparent that risk management was important. Still, years later, the risk management policies of these financial institutions remains mostly unknown. The paper tries to take a closer look at the practice of risk management at these institutions using credit card data provided them and machine learning techniques. The paper argues that the consumer-credit market is central to understanding risk management at large financial institutions as their credit risk management is a reflection of their risk management as a whole.

2.1.1 Dataset

The dataset Butaru et al. (2016) are using is aggregated from account-level data from six major U.S financial institutions and credit-bureau data from a large U.S credit bureau. In total, the dataset is comprised of over 500 million records over a period of six years. The account-level data consists of 106 raw data items, reported monthly. Examples of such items are month-ending balance, credit limit, payment amount, account activity, delinquency, borrower income etc. The credit-bureau data consist of 80 raw data items, reported quarterly. Examples here include items such as total credit limit, total outstanding balance on all cards, number of delinquent accounts etc. In total, the dataset consists of 186 raw items for each individual credit-card account. Because the account-level data is reported monthly while the credit-bureau data is reported quarterly, when merging the two different datasets, the credit-bureau data is repeated three times in the merged dataset.

In addition, Butaru et al. (2016) have augmented the credit card data with macroeconomic variables on both county and state level using the accounts address data (ZIP codes). These variables include statistics such as unemployment rate, average hourly wage, average weekly hours worked etc.

2.1.2 Attribute Selection

In the final models, a total of 87 attributes are used from the account-level, credit-bureau, and macroeconomic data. As a baseline, Butaru et al. (2016) try to replicate as many variables as possible from Glennon et al. (2008), a 2007 paper on credit scoring models, trusting Glennon et al. (2008)'s industry experience and institutional knowledge. The macroeconomic variables are then merged using the ZIP-code associated with the account.

2.1.3 Dependent Variable

Butaru et al. (2016) use the delinquency status of the account as the dependent variable. They define delinquency as an account equal or greater than 90 days past due. The standard accounting rule in banks is to charge off an account 180 days or more past due, but accounts are rarely recovered after being 90 days past due, so 90 days past due is often used instead.

2.1.4 Models

Butaru et al. (2016) implemented and considered three different credit-card delinquency models, - logic regression, decision tree models, and random-forest models. The open-source software package Weka is used to implement the machine learning models. For the decision three models, the C4.5 decision tree learner algorithm is used. The logistic regression models use a quadratic penalty function. Random forests are an ensemble learning method that constructs multiple decisions trees and outputs the class that is the mode or mean of the classes the multiple decision trees output individually. The paper used an ensemble of 20 trees in their implementation.

2.1.5 Model Timing

The dataset contains data from January 2009 to December 2013. A separate model is trained for every six months in that period. For every model, a 2-year rolling window is used to train and test each model. Data from the most recent quarter is combined with the data from the prior 12 months to make training samples. The training samples go that far back in time because of the lag structure of some of the variables, for example some of the macroeconomic variables only have yearly values. The forecast horizon, how far ahead in time they look to see if an account become 90+ days delinquent, is either 6, 9 or 12 months. As a result, the rolling windows can incorporate up to 24 months of information. To better understand the structure of these rolling windows, see figure 2.1.

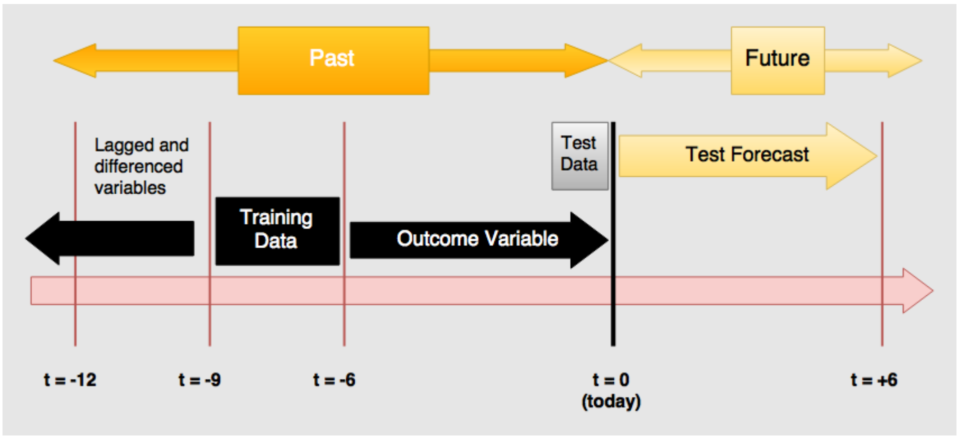

Figure 2.1: A visual representation of a rolling window with a forecast horizon of 6 months. The figure is reproduced from a presentation on the paper.

2.1.6 Measuring Performance

To measure the performance of the models, precision and recall are calculated to gauge the number of false positives and false negatives, respectively. The precision and recall are also combined to calculate the F-measure and kappa statistic, to further help evaluate the performance of the models. The kappa statistic is an interesting metric that compares the accuracy of the model with the expected accuracy of a guess, i.e. random chance.

A cost sensitive measure of performance is also introduced, by assigning costs to false negatives and false positives, and approximating the potential savings if the model was implemented. The savings can be approximated by looking at the run up in credit from when the credit line of the bad account should have been cut, to when the account goes into default. For the case of classifying good accounts as bad accounts, false positives, the authors conservatively make the assumption that the customers will pay off their remaining account balance and close their accounts, losing potential future revenue from those customers.

2.1.7 Results

The institutions included in the dataset has annual delinquency rates from 1.36% to 4.36%, which suggest the institutions have different risk management strategies and models are likely to perform differently between institutions. Individual models are trained and evaluated for each institution and forecast horizon. For the two quarter horizon forecast, the average F-Measure per bank ranges from 64% to 82%. For the three quarter forecast from 47% to 63%, and for the four quarter forecast from 39% to 52%.

Using the cost sensitive measure of performance, the value added for each model, institution and horizon has been calculated, and is represented as the percentage cost savings of each model versus passive risk management. For the two quarter horizon, the average potential percentage cost savings for each institution ranges from 47% to 75%. For the three quarter forecast between 10% and 46%, and for the four quarter forecast between -61% and 32%. For the four quarter horizon there is a less data than the shorter horizons, as well as a lot of uncertainty because of the long horizon.

The random forest and decision tree models perform about the same, both consistently outperforming the logistic regression models.

2.1.8 Attribute Analysis

To get an idea of which attributes are the more important, Butaru et al. (2016) performed attribute analysis using the learned C4.5 decision trees. From each attribute in a decision tree, three metrics were constructed:

- Log of the number of instances classified. If an attribute I used multiple times, it can be said to be more important.
- The minimum leaf number/highest node. The earlier/higher (the lower leaf number) an attribute is used in a decision tree, the more important it is, usually.
- If the attribute was selected in the model or not.

These metrics were combined to a single ranking measure. This score was then calculated for each combination of bank and forecast. Several interesting observations were made. 78 out of the total 87 attributes were used at least one time in a model. The most important, top ranking, attributes were intuitive variables like days past due, behavioral score, credit score etc., i.e. attributes one would think would be the most important. There wasn't much variation across the different time horizons in the rankings of the attributes, though there was a notable variation in attribute rankings across different institutions, likely because of different risk management strategies.

Macroeconomic attributes were shown to not be the most important, but still relatively high ranking, meaning they still can have significant impact on credit risk. The contribution of the macroeconomic attributes varied substantially across banks.

2.1.9 Closing Notes

Butaru et al. (2016) concludes that there is a substantial amount of money to be potentially saved by better risk management by the credit card issuers. There is also notable heterogeneity between banks in terms of risk factors, suggesting that models have to be fitted to the banks. The paper finally claims to provide an illustration of the potential benefits of big data, machine learning techniques and predictive analysis can bring to both consumers and banks.

2.1.10 Relevance

This paper deals with the problem of identifying high risk accounts likely to become delinquent, a problem similar to that in this thesis, albeit with the purpose of looking at the risk management strategies of different financial institutions. The dataset used share similarities to the one studied in this thesis. It provides guidelines for attribute selection, length of training window and length of forecast horizon. The paper uses the same data and methods to predict delinquency for six different banks with different results and varying degree of success for each bank. This suggest that the degree of success, using these methods, may be dependent on the existing risk management practices at the bank. Still, it shows that machine learning is viable approach to solving problems of this nature.

2.2 Consumer Credit Risk Models via Machine-Learning Algorithms

In 2010 Khandani et al. (2010) constructed non-parametric, nonlinear forecasting models for credit risk using machine learning techniques on data from a major commercial U.S bank.

Due to the large number of decisions involved in the consumer lending business, financial institutions rely heavily on algorithms to make these decisions, as opposed to human discretion. Models scoring customers on their creditworthiness are created using private information about the borrowers. While these models perform reasonably well, the measures they produce does not change much over time and the measures are therefore relatively insensitive to market condition changes. Consumer credit can deteriorate quickly, so these measures should also be able to change quickly in order to catch consumers showing signs of high-risk behavior.

2.2.1 Dataset

Khandani et al. (2010) use a dataset consisting of transaction-level, account-level and credit-bureau level for costumers of a single, unidentified U.S bank. The data spans from January 2005 to April 2009. The dataset is processed and time-aggregated to form attributes on a monthly basis.

The transaction data gives, for each transaction, information about the amount, direction (inflow or outflow), channel and category. Channel is the medium through which the transaction took place. Examples are automated teller machine (ATM), online bill payment, credit card wire transfer etc. Category refers to what the money was spent on. Examples here are restaurants, bars, grocery etc. The raw dataset had 138 different categories. The authors then further selected a subset of the 138 categories that should broadly represent the important aspects of the consumers' expenditures. Several categories were not used because of legal restrictions.

The credit bureau data provides credit score, former bankruptcies (if there are any) and "trade lines". Credit score is a number that represents a consumer's creditworthiness, usually used by lenders to evaluate the potential risk posed by lending to consumers. Credit score is throughout the paper used as a benchmark against the performance of the machine-learning models proposed in the paper. Trade lines refers to all credit and loan facilities the consumer has across financial institutions. For example, if the consumer has a mortgage with another bank, the trade lines data would contain the mortgage balance, payment history, payment status and other relevant information. Also the type of trade line. Examples here are mortgage, home loan, auto loan, credit card etc.

Finally, the transaction and credit-bureau data are matched with information about savings and checking account balances the customer has with the bank.

The various data described above is aggravated and/or collected on a monthly basis and used as input data for the models. The dependent variable is a binary value indicating whether the account has become 90 or more days delinquent within the next 3, 6, or 12 months.

2.2.2 Models and Model Timing

The data is used to train a decision tree model, using the CART-algorithm. The CART decision tree is used partly because it produces interpretable decision rules laid out as a tree. In the banking industry, this kind of transparency is appreciated as "black-box" models are viewed with skepticism and suspicion.

Khandani et al. (2010) trained and tested models in 10 consecutive periods spanning from January 2008 to April 2009. Each model was trained using input data from the first month in the period, and the delinquency data from the immediately following 3-month window. The model was then applied to the input data for the month immediately following the 3-month training window to produce forecasts of delinquencies for the following 3-month window. The forecasts were then compared against the data from the 3-month window to evaluate the model. For example, the first model was trained using input data from January 2008, and the delinquency data in the following 3-month window. Using input data from April 2008, the model produced the forecast for the following 3 months, from May 2008 to July 2008. The next period tested was then February 2008 to August 2008. This rolling-window approach was done 10 times. For the input data to model, the most recent data for that month was used for all the data except the transaction data. For the transaction data, the average values from the 6 months prior, or as many months available, were used as input.

By only training the model on data that is available at the time of forecast, and evaluating on a later time-period that is out of sample, the look-ahead-bias is minimized.

2.2.3 Results

The delinquency rate was between 2.0% to 2.5% for every period. The average predicted probability of an account going 90+ days delinquent for each period was between 59.8%

and 63.6% among customers actually going 90+ days delinquent, and between 0.6% and 1.0% among customers not going 90+ days delinquent.

The calibrated models score accounts on the probability of going 90+ days delinquent. This score has multiple uses. The most obvious being credit-line risk management. Credit lines may be increased and interest rates may be reduced for customers with a low delinquency probability. And the opposite may be true for customers with a high delinquency probability.

To classify a customer as either “good” (low-risk) or “bad” (high-risk), Khandani et al. (2010) convert a score to a binary decision by comparing it to a threshold. For example, customers scoring under 10% are classified as good, while customers scoring over 10% are classified as bad. Where to place this threshold involves a trade-off. Setting a low threshold will most likely capture most of the high-risk customers, but will also classify a lot of actual low-risk customers as high-risk. The other way around, using a high threshold, may miss many high-risk customers. Balancing false positives and false negatives (Type-I and Type-II errors) is a common problem in most, if not all, classification problems. For this problem, the authors suggest balancing this trade-off based on a cost/benefit analysis of false negatives vs. false positives, selecting a threshold that optimizes the benefit and minimizes the cost. Doing this, they get prediction accuracy of around 0.99 for all periods. Of course, since the data is heavily skewed this number does not mean much. Over the different periods the precision ranges from 0.734 to 0.85, the recall ranges from 0.65 to 0.96, the F-measure ranges from 0.73 to 0.84, and the Kappa statistic ranges from 0.73 to 0.83. These numbers suggest that their model has strong predictive power.

2.2.4 ”Value Added”

Using a measure for “value added”, practically measuring the potential money saved, Khandani et al. (2010) estimate the potential net saving to be between 6% and 25% of total losses. This measure is estimated by summing the cost savings from credit reductions of customers likely to become 90+ days delinquent and the lost revenue from false positives, customers that are misclassified as high-risk, but do not become 90+ days delinquent in reality. These potential cost savings and lost revenues are hard to quantify, so the estimate is made under a conservative set of assumptions.

Khandani et al. (2010) also argue that the forecasts for the individual accounts can be aggregated to generate macroeconomic forecasts of the credit risk in the lending business. Further it can be used as an indicator of systemic risk for consumer lending. For this purpose, it may be more appropriate to use a longer forecast horizon. Therefore, they train a new model over 6 and 12-month forecast horizons. They find that the predicted delinquencies are highly correlated with realized delinquencies.

2.2.5 Relevance

Khandani et al. (2010) tackle the problem of predicting delinquencies among credit-card-holders, customers that are likely to not pay their credit debt, to develop better consumer credit risk models. This is similar to the problem at hand in this thesis. Like the dataset from Butaru et al. (2016)’ paper, Khandani et al. (2010)’s dataset combine account and credit bureau level data, but has the difference that it also includes transaction data. This is of special interest as this thesis also uses a dataset that includes transaction data. Khandani et al. (2010) therefore provide a guideline for engineering relevant features from the transaction data.

Machine learning methods, specifically CART decision trees, are used and further shows machine learning as a viable approach.

Khandani et al. (2010) chose to use a short window for training, which is relevant and interesting to this thesis as the problem is early detection of high risk customers. Obviously a shorter training window is preferred, if viable, to identify high risk customers at the earliest.

Chapter 3

Dataset

This chapter serves to describe the dataset used in this thesis. First, the reader is introduced to the different kinds of data that is combined to form the complete dataset. The definition of high risk behavior in the data is presented, explaining the dependent variable used further in the thesis. Observations for high risk customers in the dataset are pointed out. Problems related to the dataset will then be discussed. The chapter is completed by comparing the initial credit score of the customers with the realized delinquencies in the dataset.

3.1 The Data

This thesis is written in collaboration with SpareBank 1, a large Norwegian bank, which has provided the dataset used in this thesis. The dataset consists of credit card data from over 162,000 unique customers over a period of over 3 years. Over 11 million records in total. The dataset combines account-level data, transaction-level data, and tax assessment data for the first 18 months of each account. The data spans from November 2013 to April 2017.

All individual identifying information, such as names, addresses and social security numbers, is stripped from dataset given the sensitive nature of credit card data.

3.1.1 Account Data

The main dataset consists of account-level observations for each individual credit card account, and is reported monthly from November 2013. Each monthly report is made at the end of each month, or for some of the variables, for the invoice that is due in the current month. These observations include aggregated data for the month, flags for events during the month, static data for the month, and static data for the entire 18 months.

Aggregated data includes observations such as closing balance, payment amounts, total cash withdrawals amount, number of purchases, maximum and minimum balance during the month, and similar observations. Event flags include observations such as if the account becomes overdue during the month, if the account is over the credit limit during the month, if there is a change in the credit limit during the month, and similar account status changes. Static data for the month is data such as opening balance, credit limit, and months since the account was created. Lastly, the static account data includes data such as the date the account was created, date of first transaction, and credit score at creation. In addition, the account-level dataset also includes the purchases and cash withdrawal amount for the first 14 days of the account.

The raw account-level dataset has approximately 2.2 million records, with 99 features each, across 162,000 individual accounts. This makes the account-level dataset the most important of the three datasets that are combined, as it contains the most information, both in number of accounts and features. Most importantly, the account-level dataset contains information about the status of the account, i.e if the account is overdue payment, and if it is, how much it is overdue. In section 3.2 we will use this information to define high risk behavior for later use in supervised machine learning methods.

3.1.2 Transaction Data

The transaction-level dataset consists of almost 9 million transactions from over 123,000 individual customers. The transaction dates range from January 2014 to April 2017. The discrepancy between the number of accounts in the account-level dataset and transaction-level dataset is explained by inactive accounts, i.e. credit card customers not using their credit card, therefore not creating any transactions for their account.

Out of the 33 features each transaction have, only a few of the features are interesting. Besides the account identifier, the interesting features are transaction date, transaction amount, transaction category. Each transaction has attached a category meant to broadly capture the nature of the transaction. In total, there are 18 different categories. Examples include Retail Stores, Service Providers, and Transportation.

It should be noted that some of the categories are quite a lot broader than some of the other categories, making the transactions unevenly distributed across the different categories (see figure 3.1).

For example, the two most frequent categories for purchases are Retail Stores and Miscellaneous Stores, both of which are very broad categories, while less frequent categories such as Airlines and Mail Order/Telephone Order Providers are much narrower categories. This is further discussed in section 3.3.2.

The number of transactions are varying for each customer every month. This makes the transaction data infeasible to use as is, as input for the machine learning models used in this thesis (see section 4.1). The machine learning models need the same number of inputs for each sample, meaning the transactions for each account need to be aggregated over a given time period to make samples of consistent size.

To form consistent samples from the transaction data, we use the same approach as Khandani et al. (2010) in section 2.2.1. For each category, transactions are summed and counted for every month to form a summary of activity for each month. The idea is that the categories customers spend money on can be used as a pattern to identify high risk customers.

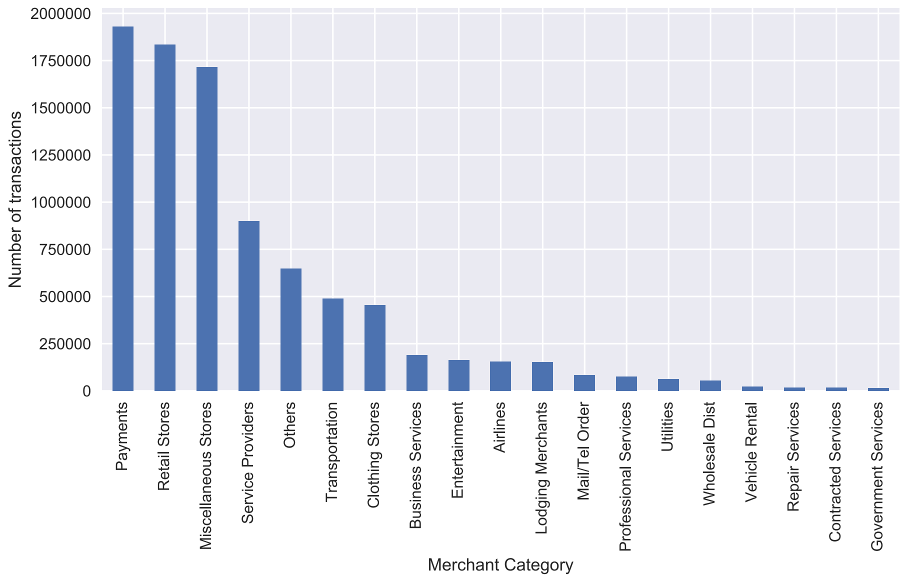

Figure 3.1: Frequency of transactions for every category. Some category names have been shortened to fit the figure.

3.1.3 Tax Data

The account-level and transaction-level datasets are complemented with tax assessment data for over 153,000 accounts. The discrepancy in number of accounts between the account-level dataset and the tax assessment dataset is because accounts opened with the assistance of a bank adviser, as opposed to through a sales channel, does not necessarily require the customer's tax assessment.

The tax dataset has 15 features per account. Examples are features such as net income, mortgage amount, other debt amount, and employment type. In section 2.1.1, we saw Butaru et al. (2016) augment their dataset with macroeconomic data to get a better understanding of each customer's financial situation. Similarly, the tax assessment data provide us with the same information, but is arguably better for that purpose, as it contains actual individual financial information for each customer.

3.2 Defining High Risk Behavior

To use the dataset to identify high risk credit card customers, high risk behavior has to first be defined in the dataset. A more precise description of high risk behavior is unwanted behavior, identifying credit card customers that, in hindsight, shouldn't have been granted a credit card in the first place. Generally speaking, from the credit card issuers point of view, unwanted behavior among credit card customers will be to not pay credit card bills on time, i.e the account is to some degree past due. However, even though overdue accounts present a higher risk, they may still be profitable. The increased risk is offset by the potential higher return when the accounts eventually pay off their debts because of the added interest and late fees. This risk-return trade-off makes it difficult to set a definite cut-off point for when a customer goes from ”good” to ”bad”, i.e when the risk outweighs the potential return.

3.2.1 Dependent variable

In this thesis, two different definitions of unwanted behavior, ”bad” customers, are used. One loose and one strict. The loose definition is a customer that has been sent to collection inside a given time frame. When an account is sent to collection, it means that an debt collection agency has been given the responsibility to collect the debt on behalf of the bank. The strict definition is a customer that has been in collection for 3 consecutive months. At that point the account is rarely recoverable, and the bank is very likely to never see the money owed them. This is called to default, and the strict definition will be referred to as this hereafter. This means the strict definition is contained in the loose definition, as an account can’t default without having been sent to collection first. These two definitions are used as dependent variables in the input for the machine learning models presented in section 4.1.

In section 2.1.3 and 2.2.1, both Butaru et al. (2016) and Khandani et al. (2010) used 90 days or more past due as the cut-off point for when an account was considered ”bad”, or unrecoverable. The definitions used in this paper are to some extent similar to those definitions. When an account is 60 days past due, the customer will receive a debt collection notice before the account is sent to collection when it becomes 66 days past due. That means, using the strict definition, an account becomes bad when it’s around 120 days or more past due.

It should be noted that accounts will not necessarily be labeled ”bad” after the same number of days. This is because customers themselves can choose their monthly credit card bill due date, and the account-level dataset only provides information about the status of the account at the end of the month, and if the account has been sent to collection during the month.

3.2.2 High risk behavior observations

The problem this thesis explores is about identifying high risk customers *early*. One question is, however, how early is necessary? Figure 3.2 shows the month the bad customers first went to collection/defaulted. We see that a large number of accounts go directly or nearly directly to collection, and of them almost half go on to default. The need is definitively to identify these accounts as early as possible, optimally already after the first month.

Another interesting observation about the bad customers, is how quick they are to take use of their new credit card. Figure 3.3 shows how many days from the account is created to first use for all customers. Figure 3.4 compares the relative frequency of number of days from an account was created to first used for normal, collection and default accounts. The plot clearly shows how bad customers are much quicker to make use of their new credit cards compared to the normal customers.

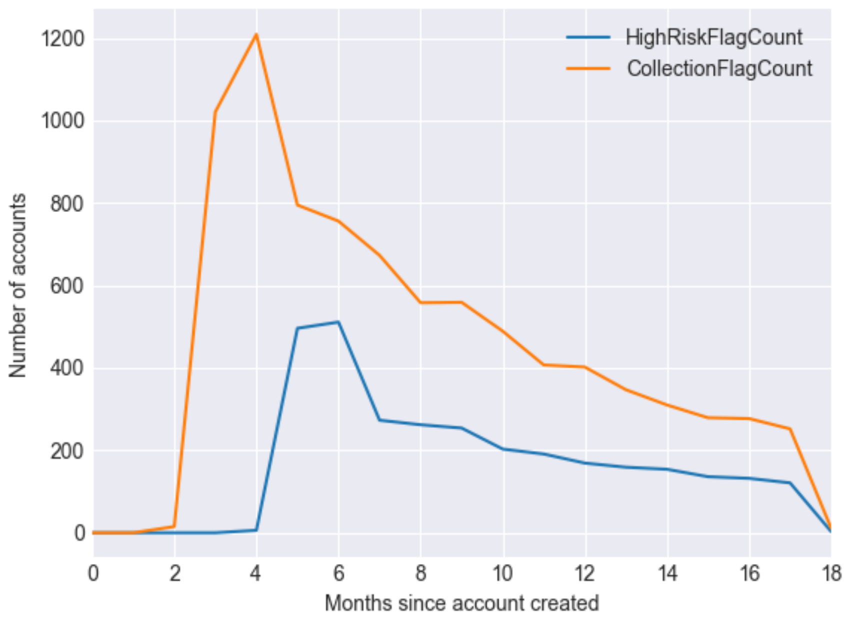

Figure 3.2: Plot showing the number of account to collection and default against the number of months since the account was created. The orange line is accounts that went to collection, while the blue line is accounts that defaulted.

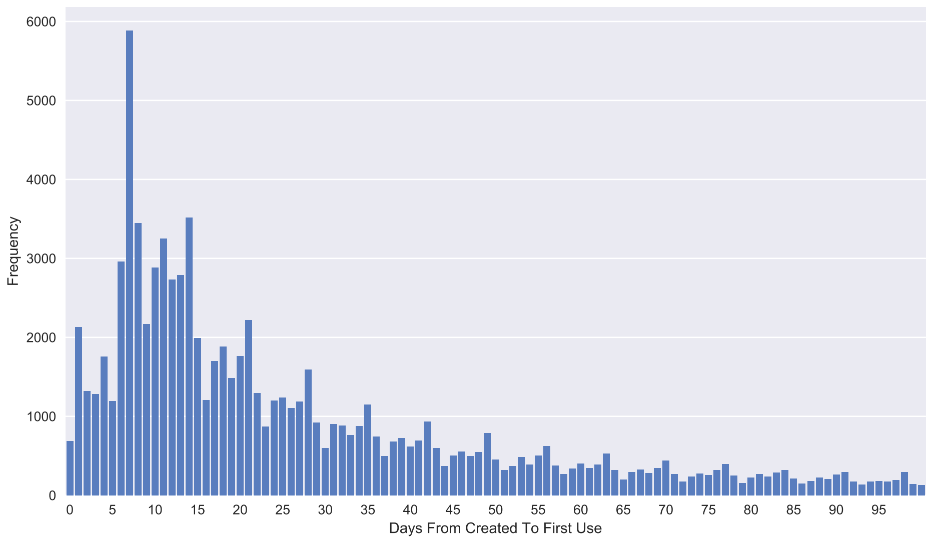

Figure 3.3: Histogram showing number of days from account first created to first use.

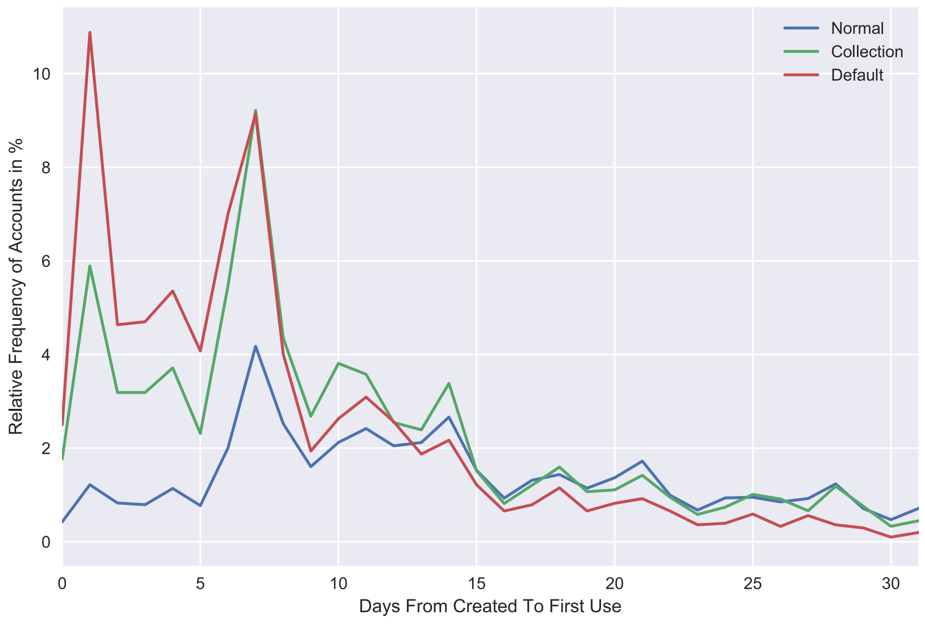

Figure 3.4: Plot comparing the normalized frequency of number of days from the account is created to it is first used for normal, collection and default accounts.

3.3 Challenges

This section describes some of the challenges involved with working with this dataset.

3.3.1 Class imbalance

In the entire dataset of 162,000 unique accounts over their first 18 months, 3067 unique accounts have defaulted, and 8360 unique accounts have been sent to collection at least once. That is 1.89 % and 5.15 %, respectively. See table 3.1 for a quick overview. The dataset is heavily skewed, imbalanced. The number of ”good” accounts vastly outnumber the ”bad” accounts. Class imbalance is a major problem for machine learning methods for classification (He and Garcia, 2009; Longadge and Dongre, 2013). The problem is that the classifiers learn that outputting the majority class result in low loss and high accuracy, meaning they do not learn to recognize the minority class. This is a reoccurring problem throughout this thesis. How to address this issue is further explored in section 4.3.

3.3.2 Transaction category imbalance

As pointed out in section 3.1.2, and seen in figure 3.1, the distribution of transactions between the 19 different categories is skewed. The transaction-level dataset is included and aggregated the way it is to extract a spending pattern for each customer. The problem

	Frequency Count	Frequency %
Default	3,067	1.89
Collection	8,360	5.15
Normal	153,914	94.85
Out of a total of 162,274 accounts		

Table 3.1: Table showing the frequency of normal, collection and default accounts.

is that the majority of purchase transactions fall within just three categories. Additionally, the account-level dataset already contains total amount and count for purchases, cash withdrawals, fund transfers and payments for each month. This means there is limited information to be gained from the transaction-level dataset. As a comparison, Khandani et al. (2010)'s raw transaction data had 138 different categories, where a subset of 40 categories was selected for the final dataset.

3.4 Credit Score

The account-level dataset includes the initial credit score for each customer. The credit score is a number between 1 and 1000 denoting the probability of the account going to collection, where a higher number means less likely to go to collection. For ordinary credit card applications, SpareBank 1 uses a cut off point for credit score of 390. That means applicants with a credit score below 390 are rejected. However, there are applications with a credit score below 390 that are approved as well, either because the application was manually approved by a bank adviser, the applicant was pre-approved, or the applicant recently was granted a mortgage.

Figure 3.5 and 3.6 shows the distribution of credit scores for the three different classes of accounts: collection accounts, defaulted accounts and normal accounts. Although the set of defaulted accounts otherwise is a subset of the collection accounts, for these two figures the defaulted accounts are not contained in the collection class, i.e they are distinct sets. This is to better highlight any differences between the two classes.

The figures show that the credit score distribution difference between collection accounts and defaulted accounts is small. However, there is a significant difference between collection and defaulted accounts, and the normal accounts. The credit score median is about 100 points larger, and the distribution is shifted much more heavily to the right.

Figure 3.7 shows the relationship between credit score and the realized frequency of collection and default. A linear regression model is fitted to the data better visualize the relationship. The plot indicates that the higher credit score, the lower the probability of collection and/or default. We see that the correlation between the credit score and realized collection/default frequency is stronger for collection suggesting there might be differences in factors that drive defaults and collections.

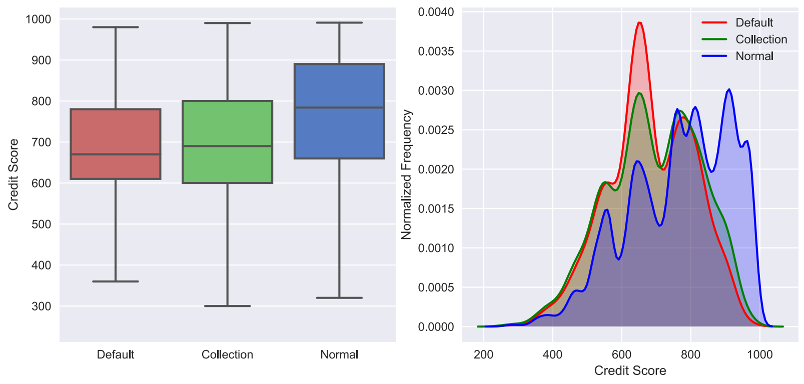

Figure 3.5: Credit score distribution comparison using box plots and a overlaying distribution plot for the three classes. The distribution plot is made using kernel density estimation.

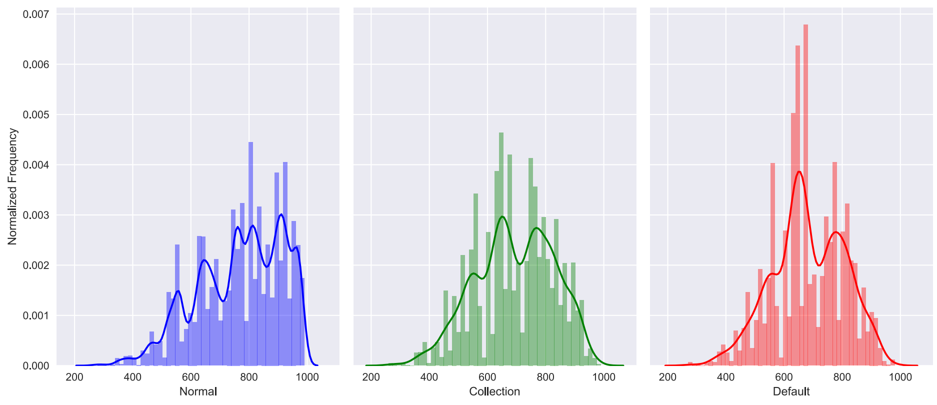

Figure 3.6: Credit score distribution comparison between the three classes. For each class the credit score histogram is combined with the estimated distribution.

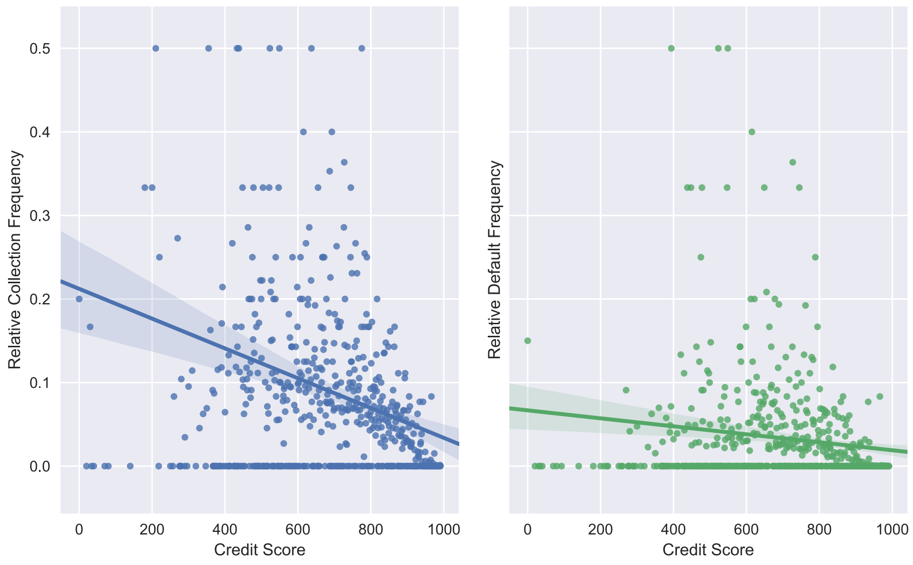

Figure 3.7: Linear approximation for the relationship between credit score and realized collections and defaults. The translucent band around each regression line is the 95% confidence interval for the estimation of the regression.

Chapter 4

Background Theory

This chapter provides the reader with an introduction to machine learning concepts, machine learning algorithms, evaluation metrics and data processing methods used throughout this thesis. It does not go in depth, but serves to give the reader the insight required to understand concepts presented in this thesis.

4.1 Machine Learning

A machine learning algorithm is a type of artificial intelligence algorithm that is able to learn from data without being explicitly programmed. Machine learning algorithms automatically detect patterns in data, and then use the uncovered patterns to predict future data (Murphy, 2012).

4.1.1 Supervised learning

Machine learning is generally divided into two main types, supervised and unsupervised. The supervised approach aim to map inputs to outputs given a labeled set of input-output pairs. The input consist of a set of features or attributes, while the output is a desired output value. Optimally, the input and output data forms a pattern that can be learned and determine class labels from new unseen instances. To do so it has to be able to generalize,

The output value can in general be anything, but is usually either a categorical variable from a finite set, or a real-valued scalar. If it is an categorical variable the problem is called classification. In the real-valued scalar case, the problem is called regression.

It is called supervised learning because models are trained under supervision. We already know the correct answers, the model iteratively makes predictions and is corrected by making updates.

4.1.2 Time series forecasting

Forecasting is about predicting the future as accurately as possible given past and present data (Hyndman and Athanasopoulos, 2013). A time series is sequence of observable data points observed at equal time intervals (Bontempi et al., 2013). Time series forecasting is simply forecasting using time series data.

Time series forecasting can be framed as a supervised learning problem by using the n previous time steps as input variables, and the next time step as the target output value. This is called one-step forecasting as it only tries to predict the next time step. When using machine learning methods to perform time series forecasting, some considerations have be done. For example, the order of the input variables can not be randomized and it is uncertain how many previous time steps the next time step is dependent on. However, machine learning has shown to be viable approach to time series forecasting (Bontempi et al., 2013; Ahmed et al., 2010).

4.1.3 Machine learning algorithms

Logistic regression

Logistic regression is a type of binary classifier that estimates the probability of a binary dependent variable given a set of explanatory values, i.e the probability a given input belongs to a certain class (Murphy, 2012; McCullagh and Nelder, 1989). Logistic regression uses the assumption that the input space can be separated by a linear boundary. As with linear regression, logistic regression computes a linear combination of the inputs before using the logistic function to produce a binary response. This makes logistic regression fast, but it assumes the input data is linearly separable. It is often used as a baseline to machine learning models.

Decision trees

A decision tree is a structure similar to flowcharts where each node represents a test on a feature, each branch represents the outcome of a test and each leaf represents a class label. See figure 4.1 for a visual example.

Creating these decision trees is called decision tree learning and is robust method for approximating discrete-valued functions (Mitchell, 1997). Although there are several decision tree algorithms, the general approach is to evaluate each instance attribute using a statistical test to determine how well it separates the training examples. The best attribute is selected as a node, and a new descendant of the node is created for every possible value of this attribute. The entire process is repeated for each new descendant using the subset of training samples associated with that descendant.

The two most common metrics to determine how well a attribute separates the training examples is *Gini impurity* and *information gain*. Gini impurity measures the probability of a random sample being classified correctly if the label is picked randomly according to the distribution in a branch. Information gain, or entropy, measures the impurity of an arbitrary collection of examples.

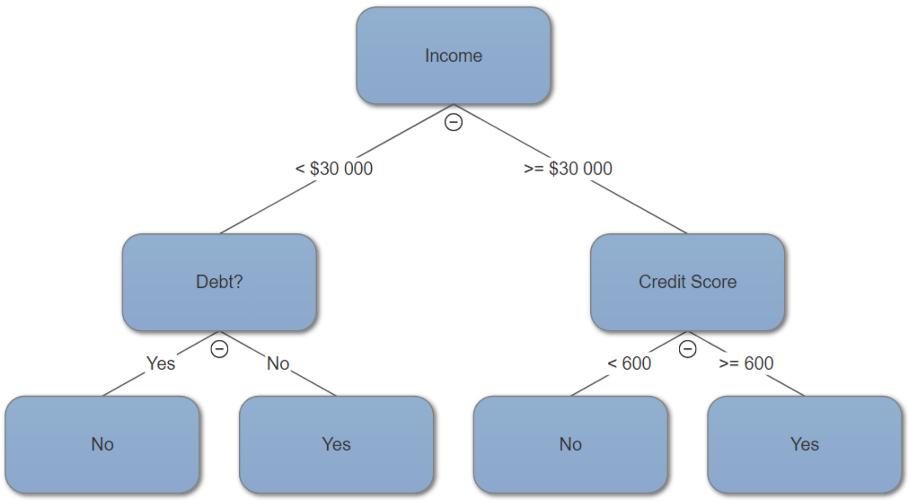

Figure 4.1: A simple example of a decision tree to decide if customer should get a credit card or not.

Ensemble methods

While ordinary machine learning approaches try to train one learner from training data, ensemble methods train multiple learners to solve the same problem and try to combine them (Zhou, 2012). The idea is that multiple learners combine their strengths and weaknesses to yield better predictive power.

An ensemble is made up of multiple base learners. Base learners are generated from training data using base learning methods such as decision trees, artificial neural networks or similar. Each learner is trained separately and predictions are combined most often by majority vote for classification and averaging for regression. Most ensemble methods use the same base learning methods for all its base learners, producing what is called a homogeneous ensemble. Base learners are often called weak learners because they individually perform just slightly better than random guessing and do not generalize well. As a contrast, the whole ensemble tend to generalize very well. An illustration of the general ensemble architecture can be seen in figure 4.2.

A popular ensemble type is bootstrap aggregating, or bagging. Each learner in the ensemble is trained on a subset of the training data obtained by sampling the training set with replacements, also called bootstrap sampling. This is done in order to promote learner variance. Bagging has a big variance reduction effect.

Another popular ensemble type is boosting. Boosting works by sequentially training a set of learners to focus on correcting the mistakes the previous learners made. Boosting primarily reduces bias, but does also reduce variance.

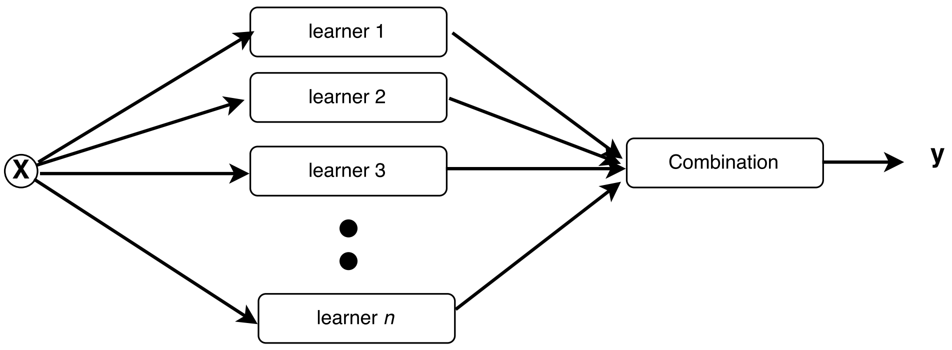

Figure 4.2: An example of a common ensemble architecture.

Random forests

Random forests is an ensemble method for classification and regression. It creates multiple decision trees and outputs the majority vote for classification, and the average for regression. It is an extension of *bagging*, where the main difference is that it uses randomized feature selection (Zhou, 2012). That is selecting a random subset of the features at each candidate split during the learning phase. The reason for this is that if one or few features are strong predictors for the dependent variable, these features will be selected for many of the trees causing the trees to become correlated.

Strengths of random forests are that it can deal with missing and imbalanced data, while still being relatively fast. As an bagging approach, it minimizes variance. It is a robust method that is not prone to overfitting.

Boosting

As mentioned in section 4.1.3, boosting is a powerful ensemble method. Two popular boosting algorithms are AdaBoost and Gradient Boosting. Both train learners to emphasize samples the previous learners misclassified, but how they do it separates them. AdaBoost emphasize misclassified training samples by, for each iteration in the training process, re-weighting them to equal the current error for that sample (Zhou, 2012). Gradient Boosting accounts for misclassified samples by fitting a new learner to the ensemble residual, that is the difference between the target outputs and the current predictions of the ensemble (Friedman, 2000).

Boosting approaches try to maximize the predictive power of the ensemble, i.e minimize the bias. The advantage of using a boosting approach is generally high predictive power, but it comes with the cost of being slow to train as each new learner is trained sequentially.

4.2 Preprocessing Data for Machine Learning Applications

Real-world data is generally noisy, incomplete and inconsistent. Many factors affect the success of using machine learning methods on a given problem, but the quality of the data may be the most important one (Kotsiantis and et al., 2006). If the data is redundant, noisy, irrelevant or unreliable, a machine learning model may not find patters during the training phase. Data preprocessing attempts to minimize this problem. Data preprocessing includes data cleaning, transformation, normalization, feature selection and more. The following subsections will discuss a few different concepts in data preprocessing.

4.2.1 Missing values

Missing values are not uncommon in datasets. A missing value is simply a value for a specific sample and feature that is missing, either because it was not recorded or was lost at one stage. The problem with missing values is that improperly handling them may introduce bias in the dataset (García et al., 2015).

Some common approaches for dealing with missing features are (Kotsiantis and et al., 2006):

- Discarding samples containing missing values in one or more features. This approach is, however, only practical if there are few samples with missing values. Does not introduce bias into the dataset.
- For categorical features, treat missing values as a category of its own.
- Select the most common feature value.
- Select the most common feature value for the class the sample belongs to.
- Substitute the value with the mean of the feature values. Alternative with the mean of the feature values for the class the sample belongs to. Median can also be used.
- Substitute the missing value with the value of the nearest neighbor.
- Develop a regression or classification model to predict the value of missing values using the complete case data of a given feature as training data, and the feature with missing values as the dependent variable.

4.2.2 Feature selection

Feature selection is the task of choosing an optimal subset of features. This means identifying the important features while discarding redundant or irrelevant features (García et al., 2015). Reasons for feature selection are many. Improving model performance, reducing storage requirements, reducing computational cost, reducing complexity etc.

Ideally, we would like to test each possible subset of features finding the one that minimizes the error rate, but that is computationally intractable. Instead, there are three main categories of feature selection algorithms: filter methods, wrapper methods, and embedded methods.

Filter methods

Filter methods, as the name suggest, filter out undesirable features before learning. They use heuristics based on the characteristics of the data to select the best feature subsets. Examples of measures used include Pearson correlation, mutual information and significance tests. Due to the relative simplicity of these methods, filter methods are able to handle big data and have low time complexity.

Wrapper methods

Wrapper methods make use of a predictive model to evaluate subsets of features. Subsets of features are used to train a model, and the model is then tested on a validation set. The subset of features with the highest predictive power is selected.

Wrapper methods are computationally expensive, but they usually select the best feature subset for a particular model.

Embedded methods

Embedded methods integrate feature selection as a part of the training process of a model. Similarly to the wrapper approach, embedded methods specifically select features for a certain machine learning model.

4.2.3 Normalization

Within a feature, there is often a large difference between the minimum and maximum value. Normalization is a feature transformation that scales down the values within a feature to a narrower range of values (Kotsiantis and et al., 2006). This is an important process for many machine learning methods such as neural networks and kNNs.

The two most common normalization methods are min-max normalization and z-score normalization.

- Min-max normalization: Scales all the numerical values of a numerical feature to a specified range.

  $$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{4.1}$$

  This type of normalization is common for learners based on distance (García et al., 2015). Min-max normalization will in those cases stop features with large differences between their max and min value dominate the distance calculations.

- Z-score normalization (or standardization): Rescales the features so that they have the properties of a standard normal distribution with mean(average) $\mu = 0$ and standard deviation from the mean $\sigma = 1$. Z-scores of the samples are calculated as follows:

  $$z = \frac{x - \mu}{\sigma} \tag{4.2}$$

  This normalization makes it robust from outliers and is important if comparing measurements with different scales.

4.3 Learning From Imbalanced Data

A imbalanced dataset is, technically speaking, any dataset where the distribution of classes is unequal (He and Garcia, 2009). More commonly, however, is it to say a dataset is imbalanced if the dataset exhibits *significant* imbalances. Examples of orders of such imbalances are 100:1, 1000:1, or even 10,000:1. More specifically, this kind of imbalance is called between-class imbalance.

Examples of imbalance in real-world datasets are many. For example, in the medical field lots of dataset exhibit significant imbalances as the number of healthy people is much bigger than the number of sick people, for most sicknesses. The cost of misclassification can also be especially large in the medical field.

The problem is that most standard machine learning algorithms consider balanced datasets, which generate good cover for the majority class, but may discard the minority class (Lopez et al., 2013). There are several reason for this. The global performance measures used to guide the learning process, such as accuracy, often favor the majority class. Patterns that identify the minority class may be highly specialized, leading to low coverage and being discarded in favor of more general patterns identifying the majority class. Clusters of minority class samples may be interpreted as noise and be wrongly discarded. Also, actual noise in the samples can make the identification of the minority class harder, as it will have fewer samples to train on.

In the next few sections, methods for dealing with the challenge that is imbalanced data will be presented. In addition to the methods discussed below, ensemble methods from section 4.1.3 are viable approaches to this problem.

4.3.1 Sampling methods

Sampling methods for imbalanced learning applications typically means modifying the dataset to provide a balanced class distribution. Although classifiers absolutely can learn from imbalanced datasets, studies have shown that, for several base classifiers, balanced datasets provide better overall performance (He and Garcia, 2009).

There are two categories of sampling methods: oversampling and undersampling. Oversampling adds data to the dataset, while undersampling removes data from the dataset. In addition, there are methods that combine the two.

Random oversampling

Random oversampling simply means randomly selecting minority examples, replicating them and adding them to the dataset. This can be done as many times necessary to reach the desired distribution. A problem with random oversampling is it may lead to overfitting as classifiers may create to specific rules after seeing the same sample multiple times. Although the training accuracy will be great, when tried on a test set, the classifier will fail to generalize and the performance will generally be far worse.

SMOTE

SMOTE, short for Synthetic Minority Over-sampling Technique, is an oversampling method that adds synthetic data points to the dataset (Chawla et al., 2002). A synthetic data point is created by taking a sample from the minority class, and then looking at its k nearest neighbors in feature space. Take the vector between the sample and a randomly chosen neighbor from the k nearest neighbors. Multiply this vector with a random number between 0 and 1, and add it to the feature vector (sample) under consideration. This creates a new synthetic data point.

ADASYN

Adaptive Synthetic Sampling approach (ADASYN) is another oversampling method that creates new synthetic data points (He et al., 2008). ADASYN weights minority class samples according to the level of difficulty in learning. The idea is to generate more synthetic data for minority class examples that are harder to learn compared to those that are easier to learn. This improves learning by reducing the bias introduced by the class imbalance and moving the classification decision boundary towards the samples that are more difficult to learn. The data points themselves are created using the same method as SMOTE.

Random undersampling

Random undersampling is randomly removing majority class samples until reaching the desired class distribution. Although appearing functionally similar to random oversampling, it is not, as it has different problems associated with it. The problem with random undersampling is that removing random majority class samples may lead to the classifier missing important concepts pertaining to the majority class, resulting in worse performance.

Informed undersampling methods

The problems with information loss for random undersampling can be be overcome by using informed undersampling methods like *EasyEnsemble* and *BalanceCascade* (Liu et al., 2009). EasyEnsemble builds an ensemble by independently sampling multiple subsets of the majority class and combining it with the minority class, and then training multiple classifiers on each subset. While EasyEnsemble explores the majority class in an unsupervised manner, BalanceCascade does so in an supervised manner. BalanceCascade develops an ensemble of classifiers to systematically select which majority class samples to undersample. It first trains a classifier on a subset of the majority class samples and all the minority class examples. Then it sees which majority class samples the classifier has correctly classified, and then removes some of them, effectively undersampling the dataset.

Both EasyEnsemble and BalanceCascade use AdaBoost ensembles as classifier, which means they create ensembles of ensembles.

Examples of other informed undersampling methods uses K-Nearest neighbor classifiers to undersample (He and Garcia, 2009). An example is the NearMiss-1 method which chooses the majority samples which have the smallest average distance to the three closest minority class samples.

Data cleaning methods

One problem with oversampling methods, especially synthetic ones, is that they may introduce overlapping. Tomek-links, a data cleaning technique, can be applied to identify and remove unwanted overlapping between classes (He and Garcia, 2009). A Tomek-link is defined as a pair of minimally distanced neighbors of opposite classes. If two examples form a Tomek-link it means that either one of them is noise or they are near a border. In both cases they should be removed to establish well-defined class clusters.

4.3.2 Cost-sensitive methods

Cost-sensitive methods are an alternative to sampling methods. While sampling methods alter the distribution of class samples in the dataset, cost-sensitive methods considers the cost associated with misclassifying a sample (Longadge and Dongre, 2013). I does so by creating cost-matrices associating cost to misclassifying any particular class sample. Usually there's no cost for correctly classifying samples, and the cost for misclassifying minority samples is larger than the other way around. The goal of any cost-sensitive method is to minimize the cost of misclassification. However, a problem with this method is that it is difficult to quantify the cost of different kinds of misclassifications.

Examples of cost-sensitive methods are cost-sensitive decision trees and cost-sensitive boosting methods, such as AdaC1, AdaC2, AdaC3, and AdaBoost.M1 (He and Garcia, 2009).

4.4 Evaluating Machine Learning Models

This section describes measures to evaluate performance in binary classification, that is when there are only two classes, two possible outcomes.

4.4.1 Confusion matrix

Supervised machine learning classifiers have several evaluation metrics to choose from. Many of them come from a confusion matrix which records correctly and incorrectly classified samples from both classes (Sokolova et al., 2006). Table 4.1 present a confusion matrix. The metrics following will make use of the confusion matrix in the definitions.

		Predicted	
	Class	**0**	**1**
Actual	**0**	True Negative	False Positive
	1	False Negative	True Positive

Table 4.1: A confusion matrix.

4.4.2 Accuracy

One of the most common metrics is accuracy, which gives the ratio of correctly classified samples to misclassified samples. Accuracy does not take class distribution into account, which makes it poor measure for evaluating performance on imbalanced data.

$$accuracy = \frac{tp + tn}{tp + tn + fp + fn} \tag{4.3}$$

4.4.3 Precision, recall and F-measure

For many classification applications, one class is often of special interest. In these cases, the class of special interest is often heavily outnumbered, the dataset is imbalanced. The class of special interest is sometimes called the positive class. Examples of problems where one class is of special interest are medical diagnoses, information retrieval, credit card fraud and the problem in this thesis.

The accuracy score is not suited for such cases. Imagine a two class dataset where the positive class is outnumbered 100:1, and the goal is to classify the samples. By simply classifying all the samples as the majority class, the classifier will have an accuracy score over 99%.

The measures below are suited for problems like these. The measures are calculated using the positive class.

$$precision = \frac{tp}{tp + fp} \tag{4.4}$$

$$recall = \frac{tp}{tp + fn} \tag{4.5}$$

$$F - measure = \frac{2 * precision * recall}{precision + recall} \tag{4.6}$$

Precision is the fraction of correctly classified samples from the positive class among all samples classified as the positive class. Recall is the fraction of correctly classified samples from the positive class among all samples from the positive class. The F-Measure is the harmonic mean between precision and recall.

4.4.4 Kappa statistic

Cohen's kappa statistic is a measure which compares an observed accuracy to the expected accuracy (Cohen, 1960). Using the confusion matrix we define it as follows:

$$kappa = \frac{P_a - P_e}{1 - P_e} \tag{4.7}$$

where

$$P_a = \frac{tp + tn}{N} \tag{4.8}$$

$$P_e = (\frac{tp + fn}{N}) * (\frac{tp + fn}{N}) \tag{4.9}$$

and N is the total number of samples.

4.4.5 Receiver operating characteristic curve

A receiver operating characteristic (ROC) curve plot is a visualization of a classifiers performance (Fawcett, 2006). Specifically, the plot is created by plotting the true positive rate (recall) against the false positive rate at various threshold levels. A ROC curve shows the trade-off between benefits (true positives) and costs (false positives). To plot a ROC curve, the classifier need to be able to yield a probability distribution. By using a threshold, a binary decision can be made. By varying the threshold for this decision, different points in the ROC space will be generated and a curve can be plotted. Figure 4.3 shows an example of a ROC curve plot.

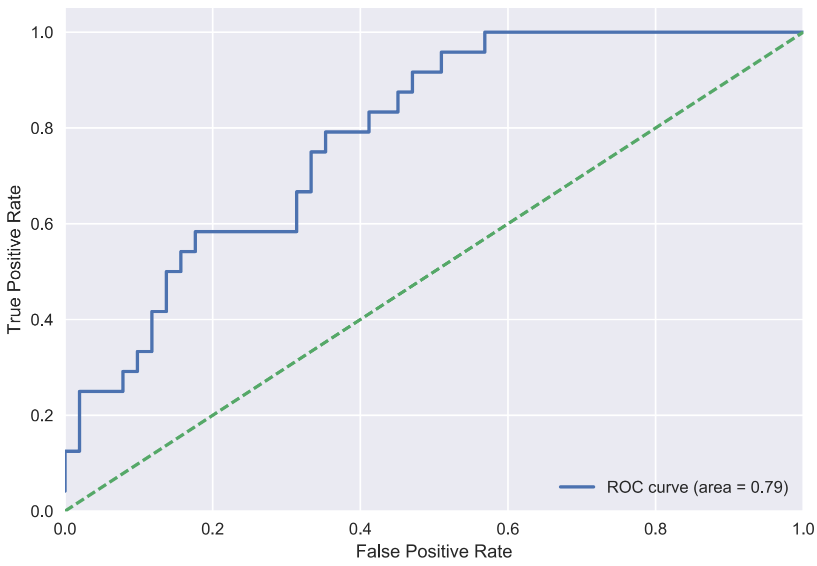

Figure 4.3: An example of a ROC plot. The green, diagonally, dotted line represents guessing randomly.

To compare classifiers using ROC curves, it would be attractive to have a single value representing ROC performance. Usually the area under the curve (AUC) is used. The AUC is a value between 0 and 1, though it should be noted that random guessing produces an AUC value of 0.5.

4.4.6 Precision-recall curve

Precision-recall curves (PRC) also visualizes the performance of a classifier. It is created by plotting the precision against the recall at various threshold settings. Although ROC is a popular measure to evaluate performance for binary classifiers, ROC plots can be misleading when used with imbalanced datasets. Studies suggest PRC is much more informative for evaluating performance of binary classifiers on imbalanced datasets (Saito and Rehmsmeier, 2015). AUC can also be calculated for PRC to compare classifiers. See figure 4.4 for an example of a PRC plot.

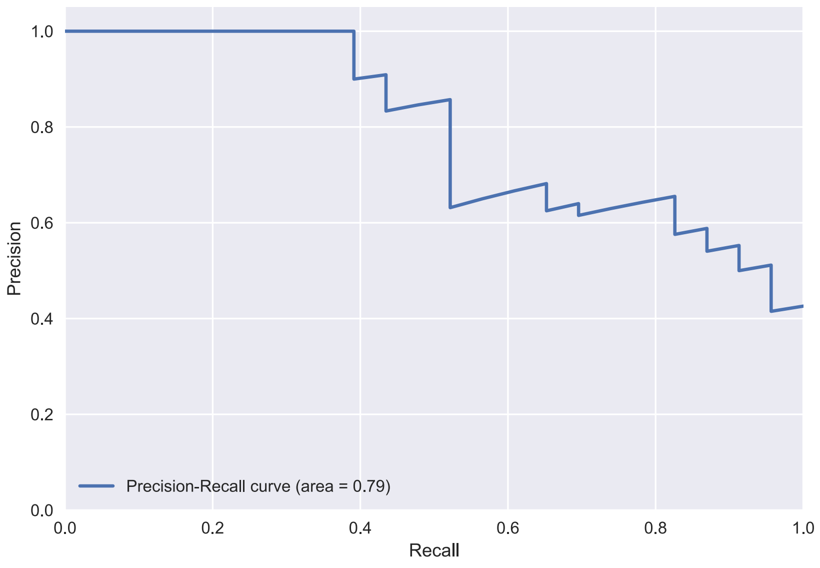

Figure 4.4: An example of a Precision-Recall curve plot.

Chapter 5

Implementation

This chapter serves the purpose to describe the experimental setup for the experiments in this thesis. The chapter describes the framework used to process the data, and build the machine learning models. It describes the structure of the datasets, how the datasets was prepossessed and how the problem in this thesis is defined as a supervised machine learning problem. Further it describes what machine learning algorithms were used, and how the resulting models were tuned and validated. Finally, the limitations in the implementation of this thesis are discussed.

5.1 Framework

For implementation, the Python programming language was used. In addition, a number of Python libraries were used for various tasks:

- All programming was done with Python 2.7.
- For machine learning and preprocessing, the Python machine learning library Scikit-Learn was used (Pedregosa et al., 2011).
- For data manipulation, exploration and preprocessing, the Python library Pandas was used (McKinney, 2010).
- To visualize data, the Python 2D plotting library Matplotlib was used (Hunter, 2007).
- Over- and undersampling was implemented using the Python library Imbalanced-learn (Lemaître et al., 2017).

5.2 Data Preparation

5.2.1 Data cleaning

One problem with the dataset was that a lot of accounts were inactive, meaning the card was not in use, or it took a long time from the account was created to the first time the card was used. In figure 3.3 we see histogram showing the amount of days customers took before using their credit cards. It should be noted that the plot only shows the frequencies for the first 100 days. Although most customers are contained in the plot, some customers took upwards of 500 days before using their cards.

Obviously we are not interested in having data from inactive accounts as it does not provide us with any information. This would only lead to worse model performance and increased computational complexity. As a result, all accounts that had not used their card within 31 days were removed. This decision almost halved the number of unique accounts in the dataset, while simultaneously removing less than 10% of the ”bad” customers, using the strict definition. The ”bad” costumers that were removed mostly became so relatively late in the customer relationship and are not the ”bad” customers primarily targeted in this thesis. What this thesis does, is focus on identifying high risk customers early in the customer relationship, optimally already after the first month. Customers that generally should not have been issued a credit card in the first place. For that reason, it made sense to remove accounts that do not become active within the first month.

An alternative could be to define the first month in the customer relationship as the the month where the card first becomes active. However, customers that took a while to activate their card and only later defaulted, may show a behavior different from the bad customers that are the primary target in this thesis, so removing them was decided to be the better choice.

5.2.2 Selection

Selecting features for this problem was not an easy task as the dataset had well over 100 features for each month of data. Having little to no domain knowledge, the decision was to leave most features in and hopefully find interesting features in the feature analysis. The rationale was that too much data is better than to little data. However, using the SpareBank 1’s domain knowledge, some features were discarded and some new features were engineered. In addition, zero-variance features were discarded as they provide no information. For a complete list of features, see the appendix.

There are a few potential problems with leaving in most features. It might lead to overfitting when training models on the data. Redundant or irrelevant features might be present, which might affect model accuracy or computational performance. Another problem is the *curse of dimensionality* (Keogh and Mueen, 2010).

5.2.3 Preprocessing

Missing values

Throughout the dataset there were some missing values for a few features. Specifically for the features *age* and *credit score*. For simplicity, the missing values for these features were substituted with the feature median.

It may have been better to substitute the missing values with the class median, but the simple solution was chosen.

Normalization

The credit card data is not entirely on the same scale as customers have different credit card limits. Because of this, normalization of the dataset may be beneficial.

Z-score standardization was tried on the dataset on proved beneficial for some machine learning algorithms. Specifically logistic regression. For ensemble approaches based on decision trees, normalization did not seem to make a difference in performance.

An approach that was considered was scaling numerical features related to transactions using a credit limit based transformation, to try to reach a common scale for all accounts. Due to time constraints, this was not done.

5.2.4 Defined as a forecasting problem

Assumption

To pose the problem in this thesis as a time series forecasting problem, a large assumption is made. We assume that the behavior patterns among high risk customers are similar for the first months of the customer relationship independent on time of year and year they first became customers. The idea is that while the underlying economic climate may change, the behavior patterns of those that do not pay their credit card bill stay the same.

As stated in chapter 3, the dataset spans from 2013 to 2017. During this time there have been some changes to the economic climate in Norway with the fall of oil prices across the globe. How much this has affected Norwegian's credit card use is difficult to say. According to the 2016 survey on Norwegian's economic situation from the National Institute for Consumer Research (SIFO), Norwegian's average credit card debt has increased in the period 2012-2016 (Torvald Tangeland, 2016). However, the percent of the population that has credit card debt has remained almost constant during this time. There is also only a slight increase in credit card use in this period. While difficult to say how this might change behavior patterns among the high risk credit card customers from year to year, the assumption is that this difference is small.

As for seasonal spending behavior, there's usually an uptick in spending around Christmas. Also tax season may affect spending. This can be a problem, for example, if a customer applies for a card around Christmas with the intent of spending a lot during the holidays and paying it back later. The behavior pattern might be the same as a bad customer, but the customer will turn out to be profitable.

To account for potential flaws in this assumption, testing is done out of sample. This will be further discussed in chapter 6.

Two class classification problem

Further, the problem is posed as a two class classification problem. The classes are good and bad customers, where bad customers are labeled as such according to the definition. This label is also associated with the month the customers fit the definition of bad customer.

Training window

Training window refers to the amount of historic data used to make a prediction, the number of previous time steps used. In this thesis, the priority is to identify high risk customers early. Looking at figure 3.2, we see that a large proportion of accounts that go to collection, do so within the first months. To identify these accounts early, the training window will have to be short. In this thesis, the training window is one and two months, depending on the dataset used.

Forecast horizon

The forecast horizon is the number of months ahead in time we look, after making the prediction, to see if the customer turned out to be a bad customer or not. Choosing the length of the forecast horizon is a balancing act between not leaving enough time for an observed pattern to manifest itself as a bad customer, and choosing a too long horizon that will include bad customers that didn't exhibit high risk behavior before after the forecast was made.

Combining knowledge from the literature review in chapter 2 and figure 3.2, a forecast horizon of 6 months is chosen as a default in this thesis. Other horizon lengths are also tested.

Figure 5.1 shows how the problem is modeled.

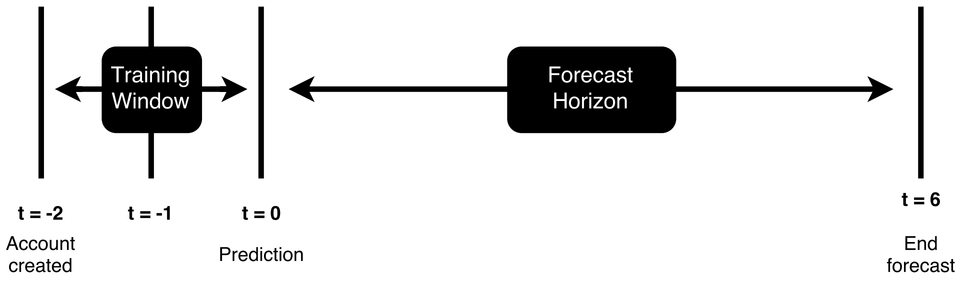

Figure 5.1: An illustration of a how a two-month training window and six-month forecast horizon are connected.

5.3 The Algorithms

5.3.1 Models

Implemented using Scikit-learn, a total of five different machine learning algorithms were tried. These were:

- Logistic regression.
- Decision trees.
- Random Forests

- AdaBoost.
- Gradient Boosting.

Linear regression is chosen as a baseline approach to compare the other algorithms to. Decision tree is a classic machine learning algorithm suitable for this type of problem. It produces decision rules that can be used for attribute analysis, and it also serves as a good comparison to the ensemble methods. Random forests, AdaBoost and gradient boosting represent both the bagging and boosting approach, and are all powerful ensemble methods that are often used on imbalanced datasets (Liu and Zhou, 2013). For those reason are these methods chosen. AdaBoost is implemented with random forests as weak learners, which have been shown to outperform other ensemble methods for some applications (Thongkam et al., 2008).

5.3.2 Dealing with imbalanced data

As previously discussed, the dataset is heavily imbalanced. To account for this, both sampling and cost-sensitive methods were used.

For the logistic regression, decision trees and random forest models, Scikit-learn offers the option to weight classes differently. For these models the classes are weighted proportionally to the inverse of their frequency.

Several over- and undersampling methods are tried for all models. Especially for the boosting models. Methods tried include random over- and undersampling, SMOTE, ADASYN, Tomek-links and Edited Nearest Neighbor. Of these methods, no method outperformed the simple random oversampling approach. For that reason, random oversampling was used with gradient boosting and AdaBoost, and also tried with random forests. A combination of informed undersamling and synthetic oversampling did perform marginally better for some of the experiments.

5.3.3 Validating models

To tune hyperparameters, assess performance of sampling and cost-sensitive methods, and pick the best models, stratified k-fold cross validation was used. Usually either 5-fold or 10-fold.

It should be noted that when validating oversampled models using cross validation, the oversampling must be done inside the cross-validation loop. For each iteration in the cross validation, the $k - 1$ subsamples get oversampled, while the single remaining validation sample is left as is. If done improperly, the validation set is highly likely to contain training data and not provide an unbiased performance measure.

5.3.4 Tuning parameters

To tune models a combination of random search, grid search and hand-tuning was used. Random search was the most used, usually for a broad search over the countless hyperparameter combinations. Grid search was less used, but did see some use for exploring more specific hyperparameter combinations. Lastly, hand-tuning was used to try to fine-tune models.

5.4 Limitations

There are several factors that affect how this thesis was conducted. This section describes limitations in this thesis that must be taken into account.

5.4.1 Limited preprocessing

Data preparation is said to be the most time consuming part of any machine learning project. Given the size of this dataset, both in terms of number of records and number of features, this is was especially true for this thesis.

Selecting an optimal subset of features for this problem was not possible in the time frame of this thesis. The number of features and combinations to consider were simply too many. Also leaving most features in probably lead to having both redundant and irrelevant features in the dataset.

With this many features to consider, we can assume that there are multiple hidden feature interactions. That is where the predictive power is larger for a certain combination of the features than the sum of the individual predictive power of each feature (Jakulin and Bratko, 2002). Uncovering these and engineering new features from them would require much more time and domain knowledge. There are probably also much insight to be gained from engineering features from several months of data, when using more than one month in the training window. For example, creating lagged features that track changes in spending. In addition, the transaction data could probably be made better use of, engineering better and more precise features from it than done in this thesis.

As mentioned earlier, another potential limitation is that relevant features were not scaled based on credit limits. A bigger effort could have been used to put features on a common scale.

5.4.2 Forecast horizon length

While the length of the forecast horizon wasn't arbitrarily chosen, and multiple lengths were tested, other horizon lengths might provide better predictions and could have been explored further.

5.4.3 Machine learning models

Only five different machine learning algorithms were tried. Literature suggested that most of them are very capable for the type of problem in this thesis, but there are also many promising approaches for this type of problem that were not pursued.

5.4.4 Hyperparameter tuning

Hyperparameter tuning is a time consuming activity. Due to time constraints, not all models had their hyperparameters fine tuned. Besides a relatively broad random search, time was only spent optimizing the best performing models.

Chapter 6

Experiments

This chapter first describes the test setup. It goes on to list the measures used to report the results. Then each experiment with accompanying results are presented. An analysis of the features used is done, describing the most important features. Lastly, the models developed are evaluated as behavioral credit risk scoring models.

6.1 Test Setup

The final processed datasets consisted of roughly 60,000 accounts, created between January 2014 and November 2016. The main differences between the datasets used are the dependent variable, training window length and forecast horizon lengths. The class balance is also different between the datasets because of the varying definition and window sizes. Only the two month training window datasets included transaction data.

The dataset is split into a training and test set, where the training set constitute about 80% of the dataset, and the test set constitutes the remaining 20%. The training and test set have roughly 48,000 and 12,000 samples each, respectively. The ratio between classes is also maintained within a reasonable deviation.

The training set includes accounts created between January 2014 and April 2016, while the test set includes accounts created between April 2016 and October 2016. This means the test set is entirely out of sample. Out of sample means that there's no samples from the training set in the test set. Because we are doing a type of forecast, it also means that the test samples are from a different, later time period than the training samples. This way the classifiers are tested as closely as possible to how they would be used in production. That the test set out of sample is very important to properly measure how well the classifiers generalize, i.e the real performance of the classifier.

As stated in chapter 5, every model is validated using cross-validation. This means that models are tuned, compared and selected using the result of cross-validation. To test the selected model, the model is retrained using the whole training set this time and then tested on the test set.

6.2 Evaluating results

To evaluate the results both during validation and testing, the following metrics are used:

- Precision
- Recall
- F-measure
- Kappa statistic
- ROC curve
- Precision-recall curve

It is difficult to choose a universal measure for this problem. Optimally we would have a profit estimation function that we could optimize. In practice, the potential profit might be estimated using a function of the precision and recall, where profit can be estimated as a trade off between true positives (profit) and false positives (cost). For that case either the F-measure or precision-recall AUC can be used as measurement to optimize. In this thesis, F-measure is chosen as the measure to optimize and results for the models with the highest F-measure are reported.

6.3 Results

This section reports the results of the experiments conducted in this thesis. A total of 6 experiments using different combinations of dependent variables, forecast horizon lengths and training window lengths have been done. The results are reported for the unseen test set performance using the best performing model for each machine learning method tried. The best performing model was decided using stratified 10-fold cross validation. Performance was evaluated with F-measure.

Every machine learning method is compared for the first four experiments. For the two last experiments, only the top two performing methods from the previous experiments are tried.

In addition to the evaluation metrics, each experiment has the ROC curve, precision-recall curve and confusion matrix plotted for the top performing model. Precision and recall is a trade-off. The precision-recall curve illustrates this and can be used to evaluate the model at different threshold values.

This section is divided in subsections based on training window length and dependent variable. The results are commented for each experiment, but any conclusions are left for chapter 7.

6.3.1 One month training window

Default as dependent variable

- **Experiment 1**: Train and compare 5 different machine learning models on a dataset with a 1 month training window, 6 month forecast horizon using default as dependent variable. Transaction data is not included.

Class	Training Set	%	Test Set	%
Good	47,584	98.72	11,889	98.66
Bad	618	1.28	162	1.34
Total	48,202	100.0	12,051	100.0

Table 6.1: Overview of the class balance in the dataset used in experiment 1.

	Precision	Recall	F-Measure	Kappa	ROC AUC	PR AUC
Logistic Regression	0.08	0.84	0.15	0.13	0.91	0.13
Decision Tree	0.09	0.65	0.16	0.14	0.83	0.11
AdaBoost	0.17	0.48	0.26	0.24	0.88	0.19
Gradient Boosting	0.24	0.42	0.31	0.29	0.86	0.17
Random Forest	0.24	0.41	0.31	0.29	0.91	0.25

Table 6.2: Results for experiment 1.

This was arguably the most difficult of the experiments as it uses the shortest training window, horizon length and the default definition of high risk behavior. As we can see in table 6.1 this dataset is the most imbalanced. This is reflected in the results. In table 6.2, we see the logistic regression and decision tree achieve relatively high recall rates, but the precision is unacceptably low. The kappa values also suggest there is not much agreement (Viera and Garrett, 2005).

The ensemble methods perform better with F-measures above 0.30. Still not very good, but the kappa statistic values suggest a fair agreement.

Gradient boosting was notoriously difficult to optimize for the datasets in these experiments. Probably because of overfitting. As there was limited time available for hyperparameters optimization, and gradient boosting is a computationally costly algorithm that can't be parallelized, there was not always the case that a random search would find good hyperparameters for it. Which is why the gradient boosting model performance varies slightly for the first 4 experiments. For this experiment it did however find relatively good hyperparameters and almost performed the best.

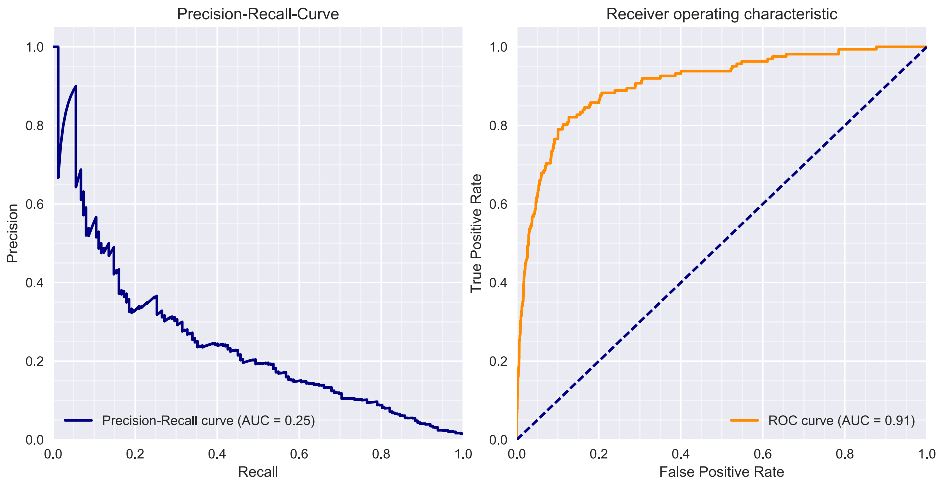

Figure 6.1: Precision-recall and receiver operating characteristic curve for the random forests model in experiment 1.

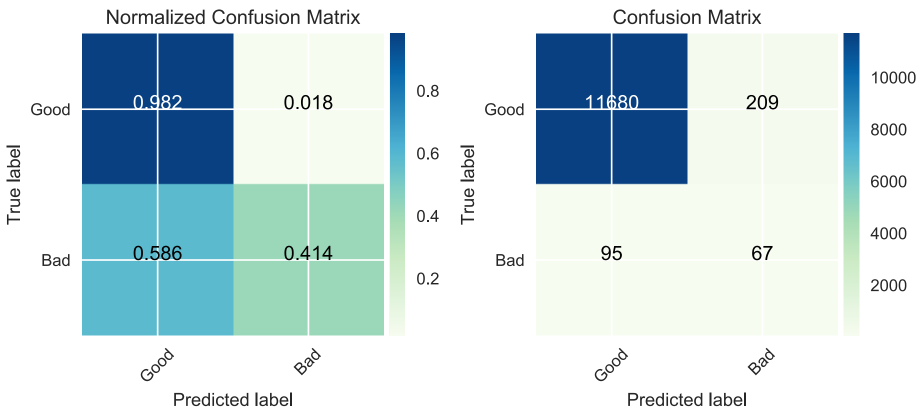

Figure 6.2: Normalized and regular confusion matrix for the random forests model in experiment 1.

- **Experiment 2**: Train and compare 5 different machine learning models on a dataset with a 1 month training window and 9 month forecast horizon using default as dependent variable. Transaction data is not included.

Class	Training Set	%	Test Set	%
Good	47,075	98.72	11,814	98.66
Bad	1,127	2.34	237	1.97
Total	48,202	100.0	12,051	100.0

Table 6.3: Overview of the class balance in the dataset used in experiment 2.

	Precision	Recall	F-Measure	Kappa	ROC AUC	PR AUC
Logistic Regression	0.09	0.83	0.17	0.13	0.90	0.22
Decision Tree	0.17	0.48	0.25	0.23	0.77	0.27
AdaBoost	0.26	0.46	0.34	0.32	0.90	0.27
Gradient Boosting	0.26	0.33	0.29	0.27	0.81	0.20
Random Forest	0.28	0.39	0.33	0.31	0.90	0.27

Table 6.4: Results for experiment 2.

This experiment is different from experiment 1 in that it uses a longer 9 month forecast horizon. As a result it includes more defaulted accounts as we can see in table 6.3. The performance is overall a little bit better, but not significantly.

The decision tree did however perform significantly better than in experiment 1. This can be explained by that the decision tree was not optimized in experiment 1. Overall, not much time was spent optimizing the decision trees as they were very susceptible of overfitting and did not generalize well.

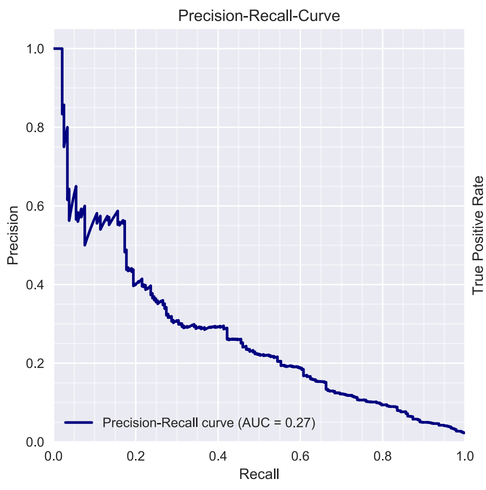

Figure 6.3: Precision-recall and receiver operating characteristic curve for the top performer in experiment 2.

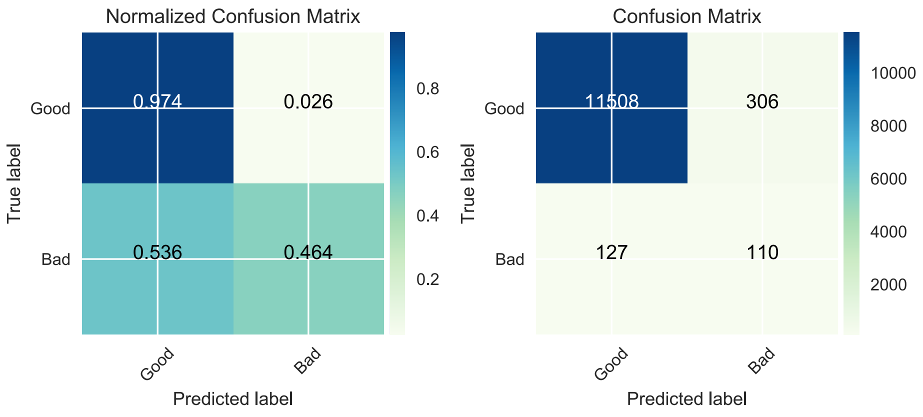

Figure 6.4: Normalized and regular confusion matrix for the top performer in experiment 2.

Collection as dependent variable

- **Experiment 3**: Train and compare 5 different machine learning models on a dataset with a 1 month training window, 6 month forecast horizon using collection as dependent variable. Transaction data is not included.

Class	Training Set	%	Test Set	%
Good	45,834	95.09	11,475	95.22
Bad	2,368	4.91	576	4.78
Total	48,202	100.0	12,051	100.0

Table 6.5: Overview of the class balance in the dataset used in experiment 3.

	Precision	Recall	F-Measure	Kappa	ROC AUC	PR AUC
Logistic Regression	0.16	0.73	0.26	0.20	0.84	0.26
Decision Tree	0.16	0.56	0.25	0.19	0.78	0.26
AdaBoost	0.31	0.42	0.36	0.32	0.84	0.29
Gradient Boosting	0.24	0.47	0.32	0.27	0.80	0.24
Random Forest	0.32	0.38	0.35	0.32	0.84	0.29

Table 6.6: Results for experiment 3.

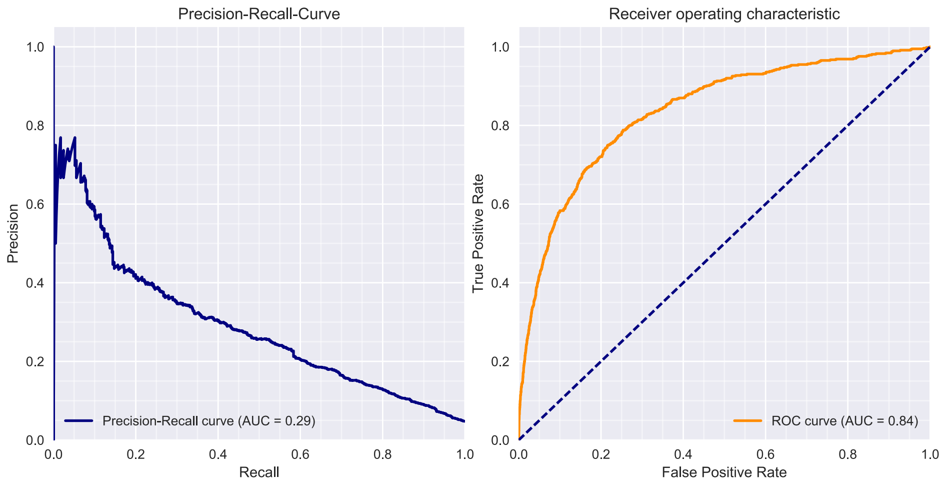

Figure 6.5: Precision-recall and receiver operating characteristic curve for the AdaBoost model in experiment 3.

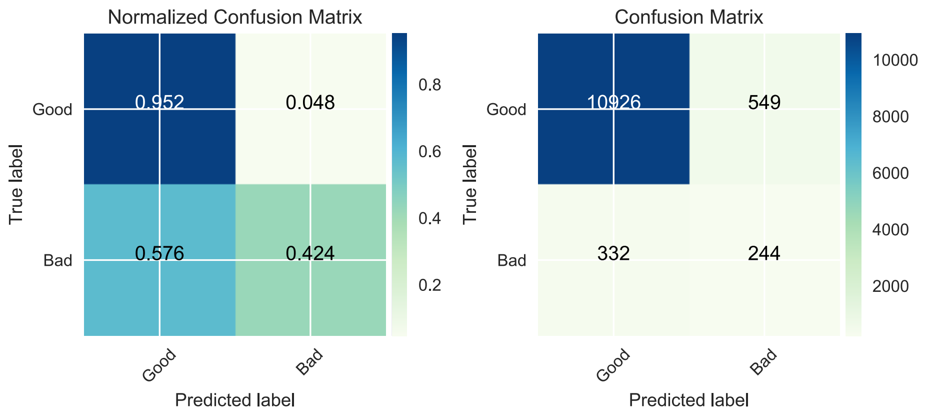

Figure 6.6: Normalized and regular confusion matrix for the AdaBoost model in experiment 3.

Experiments 3 and 4 use collection as the dependent variable and are therefore less imbalanced than experiment 1 and 2, as seen in table 6.5 and 6.7. Overall, the performance is better than experiment 1 and 2, but it is not great. The kappa statistic still only indicates a fair agreement.

As seen with experiment 1 and 2, the performance was slightly better for the longer 9 month forecast horizon.

- **Experiment 4**: Train and compare 5 different machine learning models on a dataset with a 1 month training window, 9 month forecast horizon using collection as dependent variable. Transaction data is not included.

Class	Training Set	%	Test Set	%
Good	44,726	92.23	11,313	93.88
Bad	3,476	7.77	738	6.12
Total	48,202	100.0	12,051	100.0

Table 6.7: Overview of the class balance in the dataset used in experiment 4.

	Precision	Recall	F-Measure	Kappa	ROC AUC	PR AUC
Logistic Regression	0.18	0.72	0.29	0.22	0.83	0.29
Decision Tree	0.15	0.66	0.25	0.16	0.77	0.20
AdaBoost	0.27	0.53	0.36	0.31	0.83	0.32
Gradient Boosting	0.30	0.37	0.33	0.28	0.80	0.28
Random Forest	0.31	0.46	0.37	0.32	0.84	0.31

Table 6.8: Results for experiment 4.

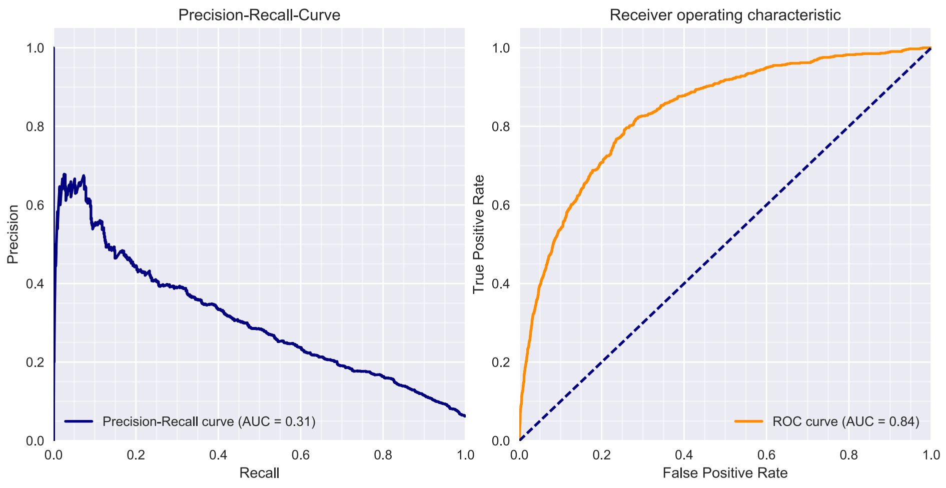

Figure 6.7: Precision-recall and receiver operating characteristic curve for the random forests model in experiment 4.

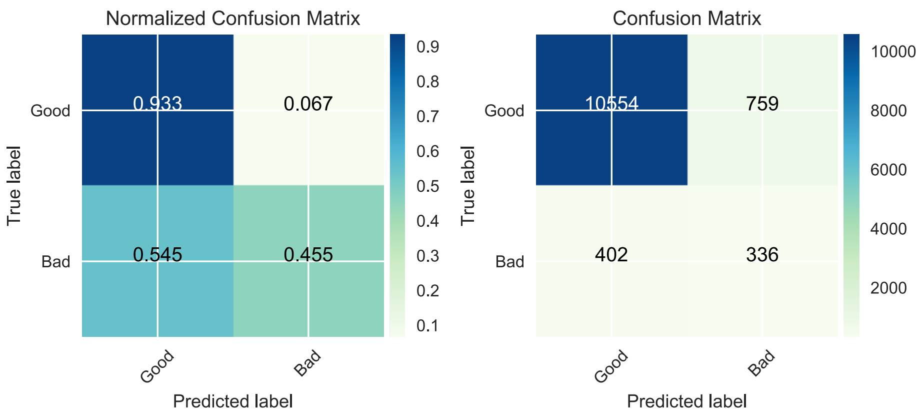

Figure 6.8: Normalized and regular confusion matrix for the random forests model in experiment 4.

6.3.2 Two month training window

For the last two experiment we use a longer training window of 2 months. In addition, the datasets also include the aggregated transaction data for both months.

From the results from the experiments in subsection 6.3.1 we have a good overview of how well the different machine learning algorithms perform.

To save time, we chose to only use the two best algorithms for the remaining two experiments. Overall, AdaBoost and random forests performed the best. In addition to this, they were also much easier to tune compared to the gradient boosting models.

Default as dependent variable

- **Experiment 5**: Train and compare 2 different machine learning models on a dataset with a 2 month training window, 6 month forecast horizon using default as dependent variable. Transaction data is included.

Class	Training Set	%	Test Set	%
Good	46,246	98.61	11,575	98.72
Bad	651	1.39	150	1.28
Total	46,897	100.0	11,725	100.0

Table 6.9: Overview of the class balance in the dataset used in experiment 5.

	Precision	Recall	F-Measure	Kappa	ROC AUC	PR AUC
AdaBoost	0.39	0.33	0.36	0.35	0.93	0.29
Random Forest	0.33	0.36	0.34	0.33	0.93	0.29

Table 6.10: Results for experiment 5.

Looking at table 6.10, we see that having two months of data is better than having just one, as the results are better. There are however not that much difference in performance. Adding transaction data doesn't seem to help much, as was previously feared. The ROC AUC is very high despite the mediocre results for the other measures used. As mentioned in chapter 3, ROC AUC can be misleading with heavily imbalanced datasets like this one. This can be explained by looking at the ROC plot in figure 6.9 and the confusion matrices in figure 6.10. We have relatively few false positives which makes the false positive rate very small because of how many 'good' samples there are compared to 'bad' samples. This makes the ROC AUC value misleadingly big.

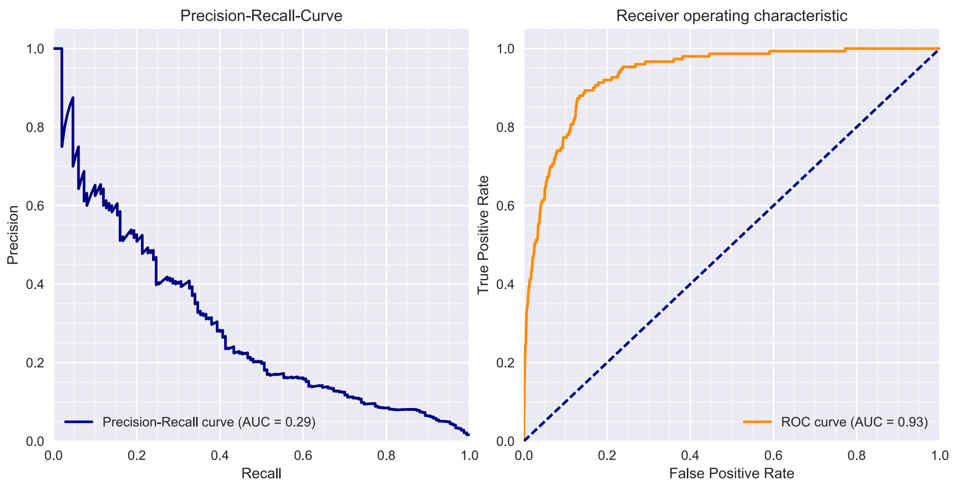

Figure 6.9: Precision-recall and receiver operating characteristic curve for the AdaBoost model in experiment 5.

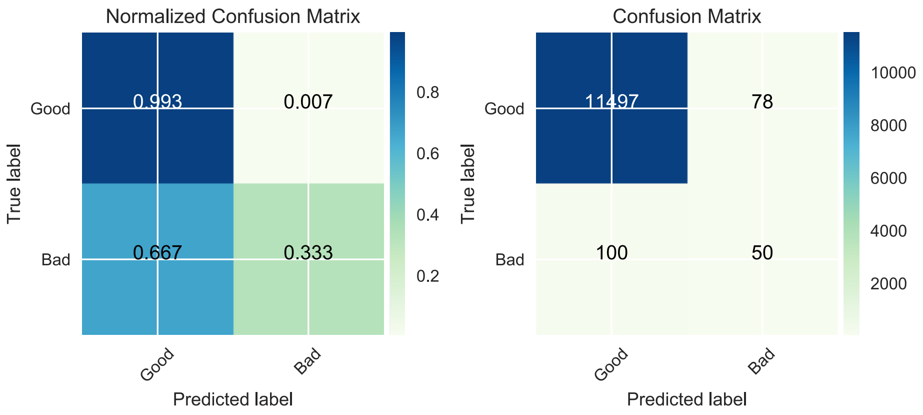

Figure 6.10: Normalized and regular confusion matrix for the AdaBoost model in experiment 5.

Collection as dependent variable

- **Experiment 6**: Train and compare 2 different machine learning models on a dataset with a 2 month training window, 6 month forecast horizon using collection as dependent variable. Transaction data is included.

Class	Training Set	%	Test Set	%
Good	44,397	94.67	11,160	95.18
Bad	2500	5.33	565	4.82
Total	46,897	100.0	11,725	100.0

Table 6.11: Overview of the class balance in the dataset used in experiment 6.

Experiment 6 is the same as experiment 5, but using collection as dependent variable this time. The performance is, as expected from seeing the results of the previous experiments, better. The kappa statistic is on the border of going from a fair agreement to a moderate agreement, using Viera and Garrett (2005) interpretation of kappa.

	Precision	Recall	F-Measure	Kappa	ROC AUC	PR AUC
AdaBoost	0.40	0.39	0.40	0.37	0.88	0.39
Random Forest	0.40	0.44	0.42	0.39	0.88	0.39

Table 6.12: Results for experiment 6.

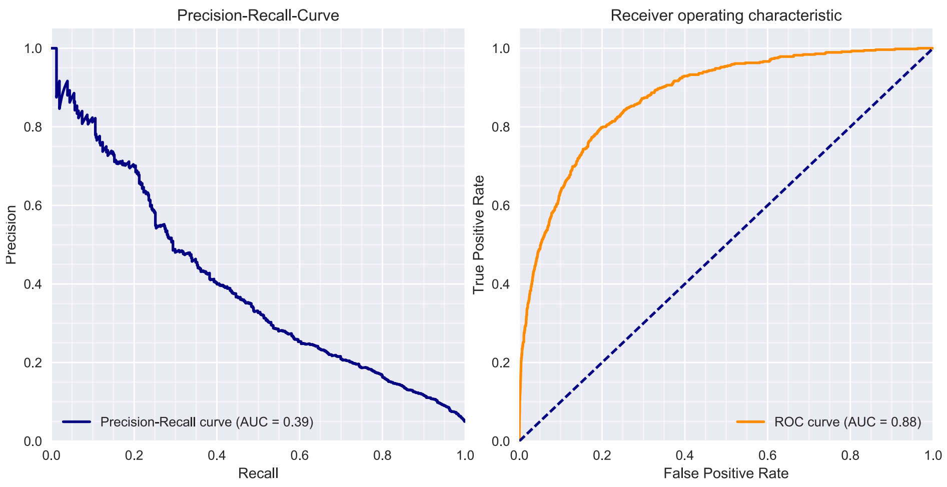

Figure 6.11: Precision-recall and receiver operating characteristic curve for the random forests model in experiment 6.

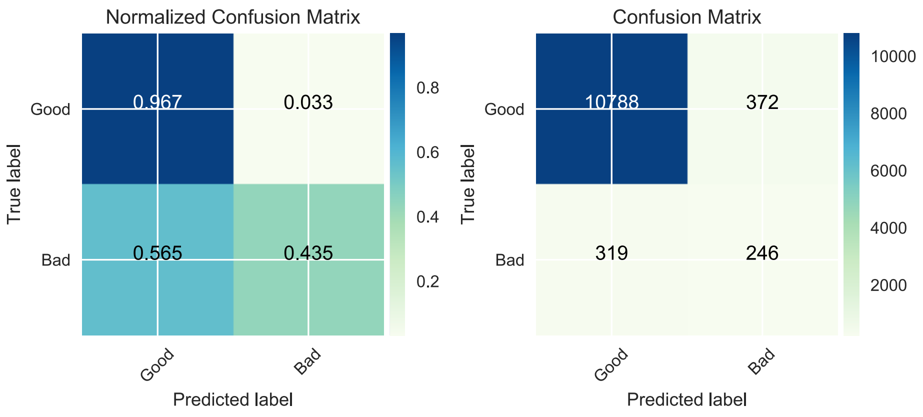

Figure 6.12: Normalized and regular confusion matrix for the random forests model in experiment 6.

6.4 Feature Analysis

Many machine learning algorithms are essentially black boxes where it is difficult to understand how decisions are made and how to interpret the results. In this thesis, we would like to know what behavior separates good and bad customers. To do this, we find the most important features, i.e the features with the highest predictive power. If we know these features, we can better understand behavior among bad customers, how to engineer better features, and how to better collect data for this purpose.

A decision tree is not black box and outputs interpretable decision rules. This has advantages. In a decision tree, features which contribute more to the decision are higher up. So the depth of a feature used as a decision node can be used to assess the relative importance of that feature. Or more precisely, the expected fraction of samples a decision node contributes to can be used as a estimate of the importance of a feature.

To reduce the variance we average this fraction of samples contributed to over thousands of randomized trees, i.e we use the random forest models to estimate feature importance.

Specifically, the feature importances are estimated using the top performing random forests models for each experiment. Those random forest models are trained with 10,000 trees and the relative feature importances are extracted averaging those expected fractions over all the trees. The feature importances are then averaged for experiments using default as the dependent variable, and collection as dependent variable. Finally, an average is provided for all experiments. Feature importances for experiments using different training window lengths are calculated separate as they are not directly comparable. The feature importances are given as a value between 0 and 1, where 0 means the feature has no predictive power, and 1 means the feature can perfectly predict the outcome alone.

Optimally, we would have liked to estimate the feature importances separately for each class, but Scikit-learn does not support that.

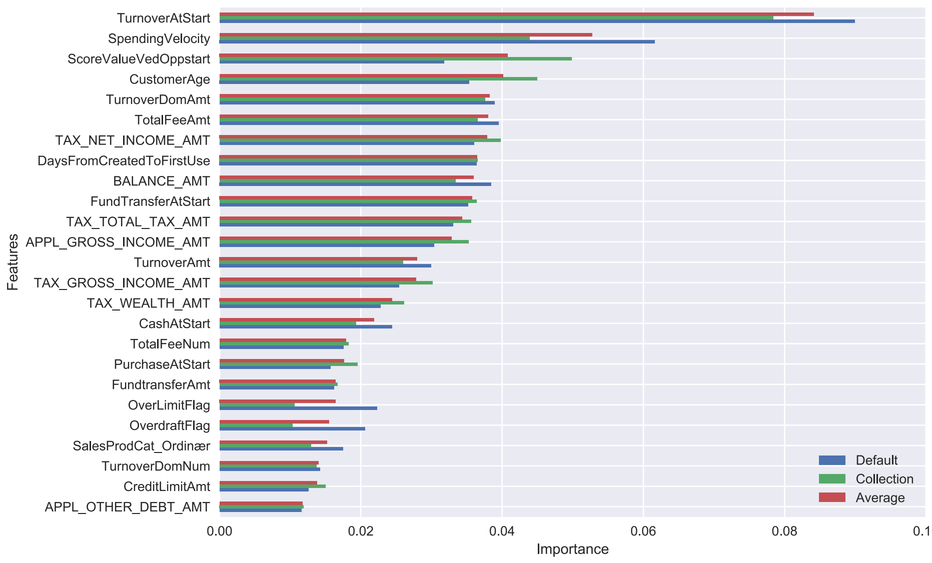

Figure 6.13: Top 25 most important features for the one month training window. The importance is a number between 0 and 1.

Figure 6.13 and 6.14 show the top 25 features for the one month and two month training window experiments, respectively. See the appendix (figure 1 and 2) for the full list of feature importances for both training window lengths.

Considering the models developed do not generalize very well, we have to exhibit some caution when analyzing the results of the feature analysis. This is moderately reflected in the feature importances as no feature importance has a value over 0.1, meaning no single feature does a good job of discriminating.

For the one month training window, figure 6.13, we see that many of the top ranking features, such as turn over at start, spending velocity, credit score (ScoreValueVedOppstart), are intuitive. It is also interesting to see that four of the engineered features, turn over at start, spending velocity (turn over amount divided by credit limit), days from created to first use, and fund transfer at start, are in the top 10 most interesting features. This suggests that there might be many hidden feature interactions, and the dataset can benefit from more feature engineering.

Must of the top ranking features have to do with spending, which is expected. Intuitively, features like over limit flag and overdraft flag should be important features, and they are as they're in the top 25, but one might expect them to be higher ranked.

There are multiple highly correlated features, such as the tax features, included in the top features. Earlier we mentioned how leaving in redundant features could lead to problems with overfitting and other problems. However, it should also be mentioned that better class separation can be obtained by leaving in presumably redundant features (Guyon and Elisseeff, 2003). It has also been shown that highly correlated features decrease the feature importance for those features (Gregorutti et al., 2013). This means that the correlated features might be more important when included by themselves.

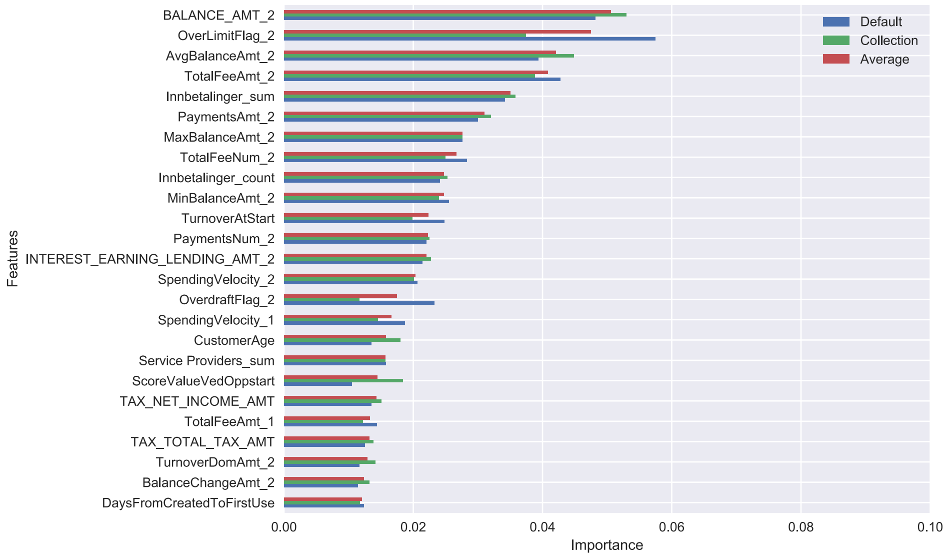

Figure 6.14: Top 25 most important features for the two month training window.

For the two month training window, figure 6.14, we see that features from the month number two, the most recent data, dominates the top ranking features. This is of course expected. Many of the same features from 6.13 are top ranked also here.

As for differences in feature importances between features predicting collection and default, there seem to be not much. For most features, the difference is negligible or within reasonable deviation.

6.5 Behavioral Credit Risk Scoring Models

While the models built in section 6.3 can not accurately classify high risk customers, the models might still be useful for dynamic behavioral credit risk scoring. The false positive rates are relatively low and the ROC AUC values are high.

The models output a score between 0 and 1 denoting the probability of a sample being a bad customer. In section 6.3 a threshold is used to classify samples as either good or bad, where samples with probability less than 0.5 are classified as good customers, and samples with probability over 0.5 are classified as bad. In this section we see if the probability values can be suitable as a credit risk score.

Because we want the models to be employed as early as possible and the performance was not that much better for models using a longer two month training window compared to one month training window, we chose to only evaluate the top models from the first four experiments. The models are also here evaluated using the test set.

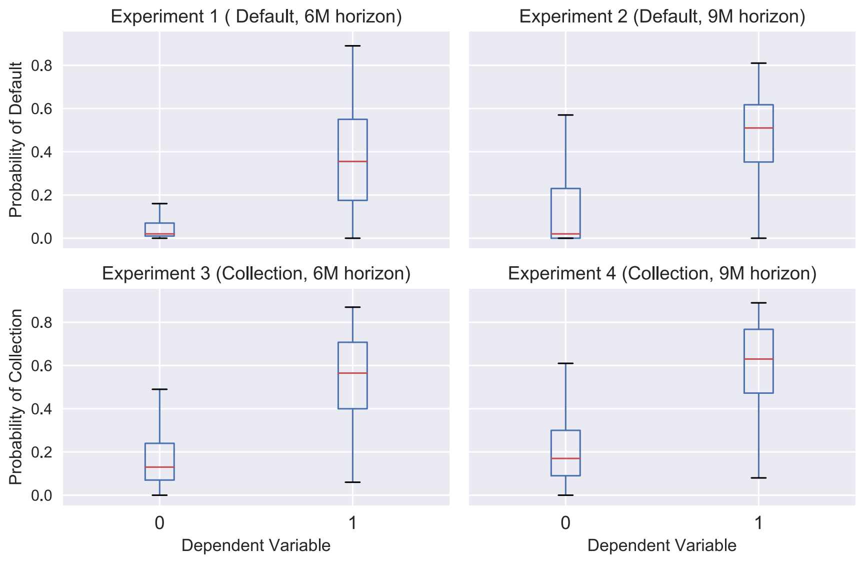

Figure 6.15: Box plots showing the distribution of behavioral credit risk score for good and bad customers for the top models from the first 4 experiments. 0 is the good customers, while 1 is the bad customers. The red line is the median and the boxes span from the 25th to the 75th percentile. The ends of the whiskers indicate the minimum and maximum value.

Figure 6.15 shows the distribution of probability values, from this point called scores, for the four first experiments. The left side of each box plot shows the distribution of scores for the good customers, and the right side for the bad customers.

The box plots seem to indicate a good separation between the classes, which is promising. We also see some interesting differences in the plots. The models using longer 9 month forecasts looks to give generally higher scores for the bad customers than the 6 month forecasts. From section 6.3 we already knew this, as the classification performance was better for longer forecast. The distribution of scores for the good customers, however, show a higher variance for the longer 9 month forecast, something we could not make out in section 6.3. Of course, this is intuitive as longer forecast involve more uncertainty.

We also see that the models predicting defaults generally give very low scores to good customers. As the fraction of defaults is low, this is expected. Especially the 6 month forecast for probability of default appears to be suitable as both the median and variance are very low for good customers.

Figure 6.16, 6.17, 6.18 and 6.19 show the probability scores plotted against the actual probability of either default or collection. The probability scores are binned to the nearest two-decimal place and the average number of defaults/collections are calculated and plotted for each bin. Which is why there are approximately 100 points in each plot instead of over 12,000. A second-order polynomial regression line is fitted to the data to show the relationship between the predicted probability and the actual probability. The translucent bands around the regression line indicates the 90% confidence interval for the regression estimate.

We see that there is a strong correlation between the predicted and actual probabilities for all four models. Table 6.13 show the linear correlation between the predicted probability and the actual probability, calculated with the Pearson correlation coefficient formula. The coefficient values further suggest there is a strong relationship between the predicted and actual probabilities. However, we can from the plots see that the relationship is not strictly linear. Because of this we have to exercise some caution when using these correlation values as the Pearson correlation coefficient can be misleading for nonlinear relationships. To account for this, the Kendall rank correlation coefficient is also provided. Kendall's τ may be a more suitable measure for the strength of the relationship considering it is non-parametric and also works with nonlinear relationships. The Kendall τ values suggest strong, positive dependence between the actual and predicted probabilities.

Looking at figure 6.16, 6.17, 6.18 and 6.19 again, we see that for all experiments the actual probability is small until the predicted probability is between 0.2 and 0.4. From there, the actual probability climbs relatively quickly. It is not optimal that the relationships behaves as such. A more linear relationship would be better.

The models predicting default seem to give scores that have a stronger relationship to the actual probabilities than the models predicting collection as can be seen in table 6.13 and from the steeper curves in figure 6.16 and 6.17, compared to the curves in figure 6.18 and 6.19. This may, however, be explained by the class distribution difference.

The differences between the 6 month and 9 month forecast models seem to be that the former's curve starts to climb a little earlier than the latter's. From the scatter plots of the the 9 month forecasts, the points seem to be more scattered than for the 6 month forecasts. This is also evidenced by the apparent larger confidence interval for the regression estimate. From the box plots in figure 6.15 this was expected as the variance was larger for the 9 month forecasts, as pointed out earlier.

Figure 6.20, shows the distribution of the predicted probabilities for each experiment using histograms. We see the models predicting accounts going to collection make more use of the probability range.

	Pearson's r	Kendall's τ
Experiment 1	0.68	0.55
Experiment 2	0.70	0.73
Experiment 3	0.63	0.62
Experiment 4	0.65	0.66

Table 6.13: The Pearson correlations and Kendall rank correlations between the predicted and actual probabilities.

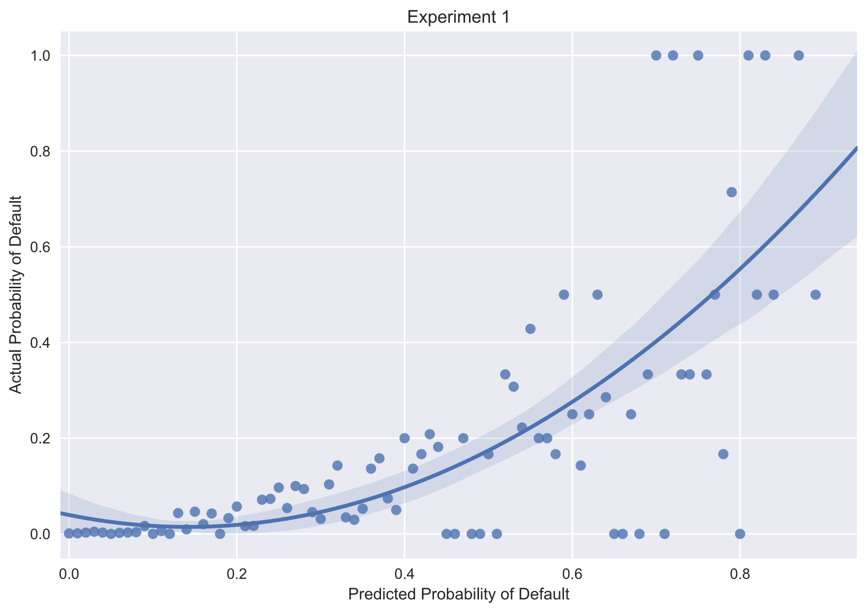

Figure 6.16: Relationship between the predicted and actual probability of default for the 6 month forecast horizon.

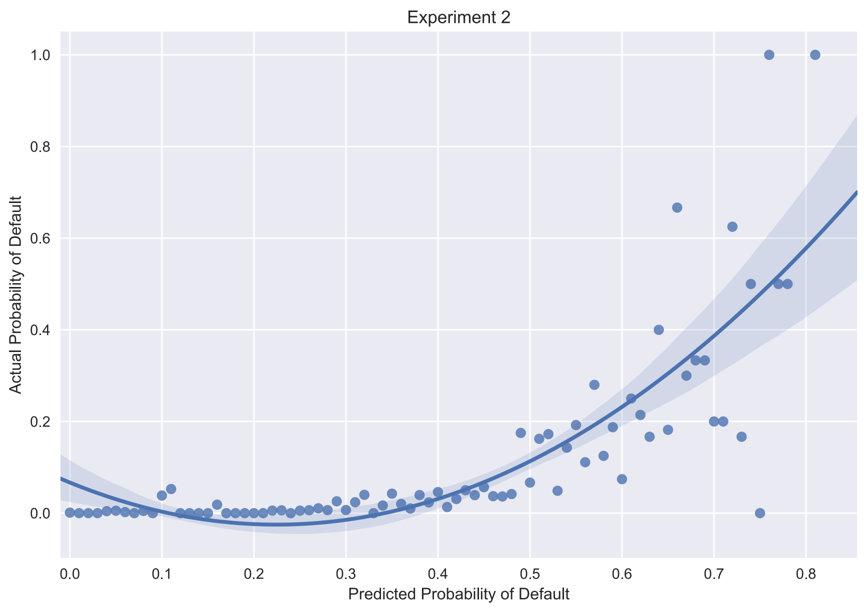

Figure 6.17: Relationship between the predicted and actual probability of default for the 9 month forecast horizon.

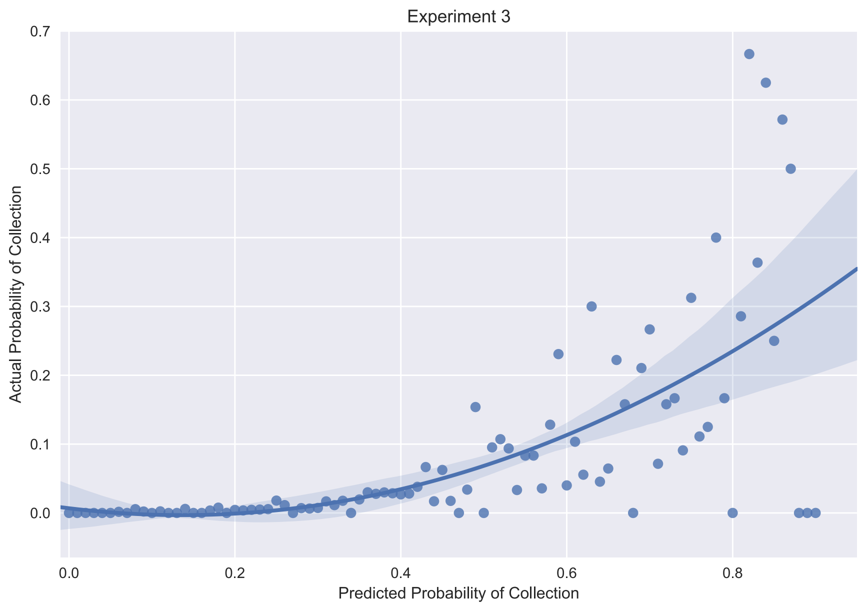

Figure 6.18: Relationship between the predicted and actual probability of collection for the 6 month forecast horizon.

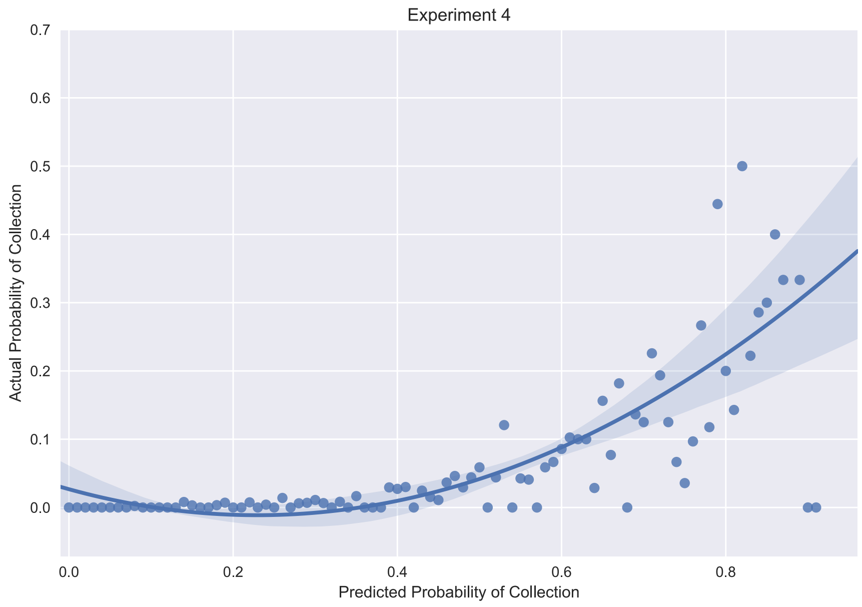

Figure 6.19: Relationship between the predicted and actual probability of collection for the 9 month forecast horizon.

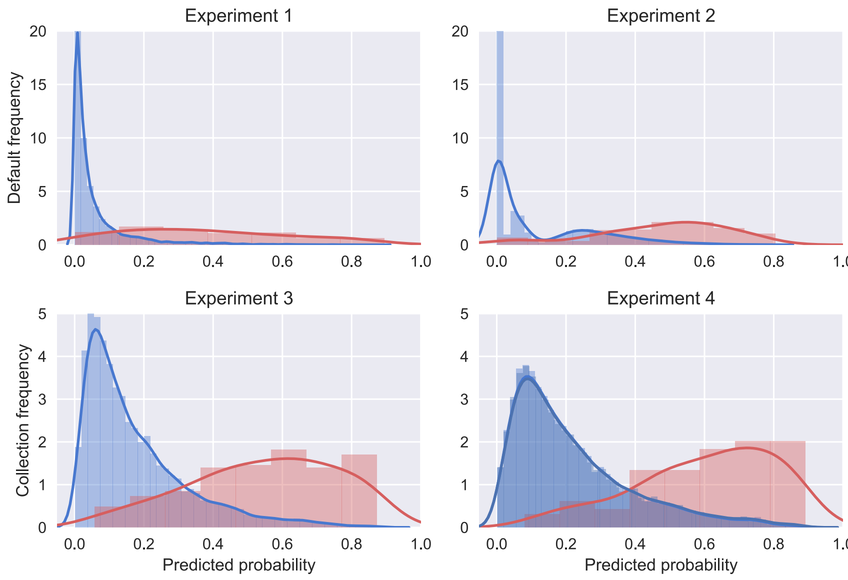

Figure 6.20: Distribution plots showing the distribution of behavioral credit risk scores for good and bad customers for experiment 1-4. Good customers are showed in blue, bad customers in red.

Chapter 7

Conclusion

This chapter discusses and interprets the results, tries to answer the research questions outlined in the introduction, points out the limitations of this thesis, and propose directions for future research.

7.1 Discussion

This is a very difficult problem. The customers we are trying to classify are new, so we have limited information about them. Within one to two months of the customer relationship, the aim is to accurately predict who will default months later. Without knowing anything about the customers' spending patterns, it is hard to separate customers that has no intention of paying their debt from the customers that are simply taking advantage of the credit and will pay back later. What constitutes a high risk customer is not a binary decision, but a sliding scale. To do binary classification, a boundary had to be set. Consequently, it is not an easy problem, and the results show that.

Butaru et al. (2016) and Khandani et al. (2010), which were reviewed in chapter 2, achieved better results. They did however base their prediction on data that spanned over a much longer time period. Butaru et al. (2016)'s training window, for example, incorporated data from up to 12 months before the prediction, including lagged variables that captured changes in the customers spending patterns. Another factor that can explain the differences in performance, is that Butaru et al. (2016) and Khandani et al. (2010) incorporate accounts that are already of different degrees delinquent. It is obviously easier to predict if an account will default if it already is 60 days past due. This is evidenced by the fact that *days past due* was found to be the by far most important feature in Butaru et al. (2016)'s feature analysis.

What could also be learned from Butaru et al. (2016)'s paper, which used credit card data from six different financial institutions, was that how well it was possible to predict future delinquency did to some degree depend on the existing risk management practices at the bank. They found it was harder to predict delinquency at banks where risk were managed well and the rate of defaults were low.

This may have contributed to the results not being better than they were. The default rate for the whole 18 months of the dataset was 2.89%. For the first 6 months it was only around 1.30%. However, this does not explain why the performance on the datasets using collection as dependent variable were not better. An explanation here might be that the behavioral patterns are weaker for this group, and therefore harder to recognize.

This imbalance in the datasets did make learning difficult for the machine learning algorithms, which is a well known problem when working with imbalanced datasets. As a result of this, one of the sub goals in this thesis became how to learn from imbalanced datasets.

For the datasets in this thesis, random oversampling and cost-sensitive methods performed the best overall with about the same performance. Without the use of these techniques the machine learning models performed extremely poorly. For some of the experiments a combination of cost-sensitive methods, informed undersampling and synthetic oversampling performed slightly better than just cost-sensitive methods or random oversampling. The assumption is that by removing overlapping samples using informed undersampling and creating new, synthetic minority class samples, the decision boundary is made more clear. Cost-sensitive methods are used as the dataset is still imbalanced after the resampling methods, as creating too many synthetic data points proved to degrade performance.

Ensemble methods proved to perform the best, as suggested by both Liu and Zhou (2013) and Butaru et al. (2016). Having few samples from the minority class combined with oversampling or cost-sensitive methods made overfitting a problem. Random forests being a method that is robust against overfitting may be why it consistently was among the top performers for all the experiments.

It is interesting that the performance somewhat correlates with class balance in each dataset, where the performance improves as the percentage of minority class samples increases. This is to some degree intuitive as more samples means more chances for the algorithms to learn the patterns that separate the classes. At the same time, one would think the performance decreases for longer forecasts as the uncertainty will be larger. This was at least the case in Butaru et al. (2016)'s results, where the performance decreased for the longer forecast horizons. For our results, this can suggest that the models recognize more generalized patterns of customers that are likely to default, and not necessarily patterns of customers that are soon going to default.

The longer two-month training window did overall lead to better predictions, but the improvement was smaller than expected. It did not seem as the machine learning algorithm managed to capture and take advantage of the increased amount of information contained in the longer training window. A way to better capture this information, might be to engineer features that model the changes from the first month to the next. For example if the balance is increasing, or the payments are decreasing or similar.

Adding transaction data to the training set did not seem to make a significant difference. In 3.3.2 this concern was addressed. The feature analysis revealed that some of the transaction features had some predictive power, but most of them were not particularly useful. Classifying transactions into more descriptive categories might make transaction data more usable.

7.2 Research Questions

- *At what accuracy can a high risk credit card customer be identified within the first months of the customer relationship?*

Depending on the definition of a high risk customer, the length of the training window and the length of the forecast horizon, the models developed have a F-measure between 0.31 and 0.42. For experiment 1 where the F-measure is 0.31, for example, this means that we identify 31% of the customers that will default with a precision of 31%. As said before, this involves a trade-off. If we look at the precision-recall curve in figure 6.1, we can shift the threshold value to identify 80% of defaults, but at the cost of having only 10% precision. Which means that for every actual default identified, 9 customers are misidentified as defaults. Or we can shift the threshold so we can identify 5% of customers that will default with a precision of 90%.

To summarize, we can't accurately identify high risk customers. That is not to say the models do not have use. This will be further discussed for research question number three.

- *What early behavior best predicts a high risk customer?*

As the classification results are not very accurate, we have to keep in mind that the feature analysis in 6.4 likely reveals more generalized behavioral patterns of high risk customers, and not necessarily specialized patterns that are exclusive to high risk customers.

The most predictive behavior, using the feature importances, seem to be spending, withdrawing and transferring large amounts right after receiving the credit card. *TurnoverAtStart*, *FundTransferAtStar*, *CashAtStart* and *PurchaseAtStart* are all features that are ranked in the top 20 most important features. These features refer to the different types of spending the first 14 days after the account is created. Related to this behavior is taking use of the card right away, which in addition to being identified as a important feature in the feature analysis, was identified in section 3.2.2 and shown in figure 3.4.

Other intuitive behavioral patterns are general high spending patterns. Having a high balance at the end of the month, large spending compared to the credit limit, high total spending, large fund transfers, spending over the credit limit and to overdraft the credit card. A similar factor is having to pay high fees, but that is generally a result of the aforementioned spending patterns.

Predictors that are not behavioral, but can be indicators of high risk customers, are low credit score, low income and having debt. Also, young men seem to be more likely to be high risk customers. It is interesting that credit limit also seem to somewhat of an indicator. This may be explained by customers in financial trouble asking for higher credit limits, but it is far from a strong indicator, however.

For longer training windows, the same patterns are found, but the more recent observations are weighted more.

- *Can a predictive model be used as a dynamic behavioral model to make decisions regarding existing customers?*

While the models devolved in this thesis are not accurate enough to make hard risk management decisions like cutting the credit line of an account, the models may be used for decisions more in line with the risk management strategy outlined in chapter 1.

As said in the answer to research question one, most high risk accounts can be identified on account of many false positives. The number of false positives is however very small compared to the total number of accounts. On the other side, some high risk accounts can be identified with high confidence. This can be usable for scoring customers.

In 6.5 we used the model outputs as credit scores and saw that the predicted probabilities of default and collection are relatively strongly correlated with the realized probabilities of those events. The distribution of predicted probabilities also seemed to have good separation. All of this makes the models to different degrees usable as credit risk scoring models.

For example, the models could be used to split customers into groups of similar risk. If we have three groups, customers could be split into a low risk group, an elevated risk group and a high risk group. Different risk management actions could then be taken for different groups. An example here could be that, if a customer wishes to have a higher credit limit, low risk customers would get automatically granted, while the elevated risk group would be reviewed or maybe denied, and the high risk group would get automatically denied or even decreased instead.

7.3 Limitations

As limitations of the implementation was discussed in section 5.4, this section will point out limitations regarding other aspects of this thesis.

It was decided from the start that this thesis would be limited to look at only the start of the customer relationship, trying to identify high risk customers at an early stage. Thus the models developed are limited to be used only after the first and second month of the customer relationship.

The dataset used in this thesis is not entirely suitable for the problem in this thesis. Because data is aggregated for the end of the month, and customers create their accounts at any time during a month, customers are not necessarily on the same scale during the first month. The transaction data could in theory have been used to address this problem, but that would require much more time.

Estimation of potential profits if implementing the models developed would have been a natural addition to this thesis. However, as the author does not have knowledge in this domain, the estimation is therefore left for the bank to look at, and is thus a limitation of this thesis.

7.4 Future Work

As noted in section 5.4, there are possibly many hidden feature interactions in the dataset that can provide better class separation if found. Engineering better features and selecting a more optimal subset of features may improve performance significantly. Related to this is finding the best combination of training window size and forecast horizon length for making the best possible credit scoring model.

As this thesis is limited to the start of the customer relationships, future works could try to make a more dynamic model not limited by the when a credit card account was created. Similar to the works by Butaru et al. (2016) and Khandani et al. (2010), but possibly shorter training window lengths while still trying to capture changes from month to month.

Future work could also look at other machine learning algorithms for this type of problem. Artificial neural networks have shown promise for imbalanced dataset problems, for instance. A recurrent neural network architecture called Long short-term memory (LSTM) could be interesting to apply to this type of forecasting problem when using multiple months for training. This type of network could also be used to develop more dynamic models. A drawback with using neural networks, however, is that they are black boxes.

Bibliography

Ahmed, N. K., Atiya, A. F., Gayar, N. E., El-Shishiny, H., 2010. An Empirical Comparison of Machine Learning Models for Time Series Forecasting. Econometric Reviews 29 (5-6), 594–621.
URL http://dx.doi.org/10.1080/07474938.2010.481556

Bontempi, G., Ben Taieb, S., Le Borgne, Y.-A., 2013. Machine Learning Strategies for Time Series Forecasting. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 62–77.
URL http://dx.doi.org/10.1007/978-3-642-36318-4_3

Butaru, F., Chen, Q., Clark, B., Das, S., Lo, A. W., Siddique, A., 2016. Risk and risk management in the credit card industry. Journal of Banking & Finance 72 (November 2016), 218–239.

Chawla, N. V., Bowyer, K. W., Hall, L. O., Kegelmeyer, W. P., Jun. 2002. SMOTE: Synthetic Minority Over-sampling Technique. J. Artif. Int. Res. 16 (1), 321–357.
URL http://dl.acm.org/citation.cfm?id=1622407.1622416

Cohen, J., 1960. A Coefficient of Agreement for Nominal Scales. Educational and Psychological Measurement 20 (1), 37–46.
URL http://dx.doi.org/10.1177/001316446002000104

Fawcett, T., Jun. 2006. An Introduction to ROC Analysis. Pattern Recogn. Lett. 27 (8), 861–874.
URL http://dx.doi.org/10.1016/j.patrec.2005.10.010

Friedman, J. H., 2000. Greedy function approximation: A gradient boosting machine. Annals of Statistics 29, 1189–1232.

García, S., Luengo, J., Herrera, F., 2015. Data Preprocessing in Data Mining. Springer International Publishing, Cham.
URL http://dx.doi.org/10.1007/978-3-319-10247-4_1

Glennon, D., Kiefer, N. M., Larson, C. E., Choi, H.-s., 2008. Development and Validation of Credit Scoring Models. Journal of Credit Risk, Forthcoming 1 (1), 1 – 70.
URL http://papers.ssrn.com/abstract=1180302

Gregorutti, B., Michel, B., Saint-Pierre, P., Oct. 2013. Correlation and variable importance in random forests. ArXiv e-prints.

Guyon, I., Elisseeff, A., Mar. 2003. An Introduction to Variable and Feature Selection. J. Mach. Learn. Res. 3, 1157–1182.
URL http://dl.acm.org/citation.cfm?id=944919.944968

He, H., Bai, Y., Garcia, E. A., Li, S., 2008. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In: IJCNN. IEEE, pp. 1322–1328.
URL http://dblp.uni-trier.de/db/conf/ijcnn/ijcnn2008.html#HeBGL08

He, H., Garcia, E. A., Sep. 2009. Learning from Imbalanced Data. IEEE Trans. on Knowl. and Data Eng. 21 (9), 1263–1284.
URL http://dx.doi.org/10.1109/TKDE.2008.239

Hunter, J. D., 2007. Matplotlib: A 2d graphics environment. Computing In Science & Engineering 9 (3), 90–95.

Hyndman, R. J., Athanasopoulos, G., 2013. Forecasting: Principles and Practice. OTexts: Melbourne, Australia.
URL http://otexts.org/fpp/

Jakulin, A., Bratko, I., 2002. Attribute Interactions in Machine Learning: Master's Thesis. A. Jakulin.

Keogh, E., Mueen, A., 2010. Curse of Dimensionality. Springer US, Boston, MA, pp. 257–258.
URL http://dx.doi.org/10.1007/978-0-387-30164-8_192

Khandani, A. E., Kim, A. J., Lo, A. W., November 2010. Consumer Credit Risk Models via Machine-Learning Algorithms. Journal of Banking & Finance 34 (11), 2767–2787.

Kotsiantis, S. B., et al., 2006. Data Preprocessing for Supervised Learning. International Journal of Computer Science 1 (1), 111–117.

Lemaître, G., Nogueira, F., Aridas, C. K., 2017. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. Journal of Machine Learning Research 18 (17), 1–5.
URL http://jmlr.org/papers/v18/16-365.html

Li, X.-L., Zhong, Y., 2012. An Overview of Personal Credit Scoring: Techniques and Future Work. International Journal of Intelligence Science 2, 181–189.

Liu, X. Y., Wu, J., Zhou, Z. H., April 2009. Exploratory undersampling for class-imbalance learning. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 39 (2), 539–550.

Liu, X.-Y., Zhou, Z.-H., 2013. Ensemble Methods for Class Imbalance Learning. John Wiley and Sons, Inc., pp. 61–82.
URL http://dx.doi.org/10.1002/9781118646106.ch4

Longadge, R., Dongre, S., 2013. Class Imbalance Problem in Data Mining Review. CoRR abs/1305.1707.
URL http://arxiv.org/abs/1305.1707

Lopez, V., Fernandez, A., García, S., Palade, V., Herrera, F., 2013. An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. Information Sciences 250, 113 – 141.

McCullagh, P., Nelder, J. A., 1989. Generalized Linear Models. Vol. 37. CRC press.

McKinney, W., 2010. Data structures for statistical computing in python. In: van der Walt, S., Millman, J. (Eds.), Proceedings of the 9th Python in Science Conference. pp. 51 – 56.

Mitchell, T. M., 1997. Machine Learning, 1st Edition. McGraw-Hill, Inc., New York, NY, USA.

Murphy, K. P., 2012. Machine Learning: A Probabilistic Perspective. The MIT Press.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12, 2825–2830.

Saito, T., Rehmsmeier, M., 03 2015. The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets. PLOS ONE 10 (3), 1–21.
URL https://doi.org/10.1371/journal.pone.0118432

Sokolova, M., Japkowicz, N., Szpakowicz, S., 2006. Beyond Accuracy, F-score and ROC: A Family of Discriminant Measures for Performance Evaluation. In: Proceedings of the 19th Australian Joint Conference on Artificial Intelligence: Advances in Artificial Intelligence. AI'06. Springer-Verlag, Berlin, Heidelberg, pp. 1015–1021.
URL http://dx.doi.org/10.1007/11941439_114

Thongkam, J., Xu, G., Zhang, Y., 2008. Adaboost algorithm with random forests for predicting breast cancer survivability. In: IJCNN. IEEE, pp. 3062–3069.
URL http://dblp.uni-trier.de/db/conf/ijcnn/ijcnn2008.html#ThongkamXZ08

Torvald Tangeland, 2016. Den Økonomiske Situasjonen for Norske Husholdninger . Fagrapport nr. 1, Forbruksforskningsinstituttet SIFO - HIOA.
URL http://www.hioa.no/extension/hioa/design/hioa/images/sifo/files/file80531_sifo_fagrapport_1_16.pdf

Viera, A., Garrett, J., 5 2005. Understanding interobserver agreement: The kappa statistic. Family Medicine 37 (5), 360–363.

Zhou, Z.-H., 2012. Ensemble Methods: Foundations and Algorithms, 1st Edition. Chapman & Hall/CRC.

Appendix

Feature Importances

Table 1: Feature importances for experiments using a one month training window. The table is sorted on average importance and is descending.

Features	Default	Collection	Average
TurnoverAtStart	0.08992	0.07836	0.08414
SpendingVelocity	0.06163	0.04389	0.05276
ScoreValueVedOppstart	0.0318	0.04981	0.04081
CustomerAge	0.03535	0.04493	0.04014
TurnoverDomAmt	0.0389	0.03759	0.03824
TotalFeeAmt	0.03947	0.03652	0.03799
TAX_NET_INCOME_AMT	0.03602	0.03977	0.0379
DaysFromCreatedToFirstUse	0.03641	0.03655	0.03648
BALANCE_AMT	0.03847	0.03342	0.03594
FundTransferAtStart	0.03516	0.03639	0.03577
TAX_TOTAL_TAX_AMT	0.03306	0.03559	0.03432
APPL_GROSS_INCOME_AMT	0.03038	0.03527	0.03282
TurnoverAmt	0.02997	0.026	0.02799
TAX_GROSS_INCOME_AMT	0.02539	0.03016	0.02778
TAX_WEALTH_AMT	0.02279	0.02612	0.02445
CashAtStart	0.02439	0.01928	0.02183
TotalFeeNum	0.01754	0.01824	0.01789
PurchaseAtStart	0.01568	0.01952	0.0176
FundtransferAmt	0.01622	0.01667	0.01644
OverLimitFlag	0.02229	0.01058	0.01644
OverdraftFlag	0.02059	0.01034	0.01547
SalesProdCat_Ordinær	0.01745	0.01294	0.01519
TurnoverDomNum	0.01423	0.01371	0.01397
CreditLimitAmt	0.01256	0.01498	0.01377
APPL_OTHER_DEBT_AMT	0.01159	0.0119	0.01175
PurchaseAmt	0.00951	0.01238	0.01094
APPL_MORTGAGES_AMT	0.01079	0.01104	0.01091
FundtransferNum	0.01002	0.0109	0.01046
TurnoverNum	0.01036	0.01026	0.01031

CashDomAmt	0.01007	0.0089	0.00949
Gender_Mann	0.00737	0.01153	0.00945
PurchaseDomAmt	0.00789	0.00959	0.00874
SalesProdCat_Kampanje	0.00992	0.00755	0.00874
PurchaseNum	0.00646	0.00822	0.00734
PaymentsAmt	0.00656	0.00701	0.00678
PurchaseDomNum	0.00633	0.00715	0.00674
CashDomNum	0.00653	0.00691	0.00672
CashAmt	0.00725	0.00609	0.00667
TurnoverIntAmt	0.00421	0.00665	0.00543
AppSalesCh_Operatørkanal	0.0052	0.00549	0.00534
CHANNEL_TYPE_INTERNAL	0.00516	0.00544	0.0053
EMPLOYMENT_TYPE_Perm employee	0.0041	0.00527	0.00468
TAX_CLASS_1E	0.00324	0.00584	0.00454
DIST_SpareBank 1 Oslo Akershus	0.00547	0.00354	0.0045
PurchaseIntAmt	0.00333	0.00537	0.00435
TAX_CLASS_1	0.00361	0.00489	0.00425
CHANNEL_TYPE_CREDITCARD_BPM	0.00432	0.00374	0.00403
AppSalesCh_Nettbank	0.00377	0.0038	0.00378
CashNum	0.0035	0.00331	0.0034
DormantFlag	0.00319	0.00346	0.00333
CHANNEL_TYPE_AUTH_WEB	0.0051	0.00149	0.0033
DIST_SpareBank 1 SR-Bank	0.00294	0.00365	0.00329
AppSalesCh_Autentisert web	0.00503	0.00148	0.00326
FullpayerFlag	0.00308	0.00343	0.00325
DIST_SpareBank 1 SMN	0.00294	0.00345	0.0032
TurnoverIntNum	0.00268	0.00359	0.00313
DIST_SpareBank 1 Nord-Norge	0.00288	0.00336	0.00312
ElectronicPurchaseAmt	0.00248	0.00375	0.00312
PaymentsNum	0.00275	0.00345	0.0031
APPL_STUDENT_LOAN_AMT	0.00279	0.00339	0.00309
AppSalesCh_Responsside	0.00268	0.00319	0.00294
PROD_SpareBank 1 MasterCard Gold	0.00217	0.0028	0.00249
PurchaseIntNum	0.00205	0.00283	0.00244
ActiveAccountFlag	0.0028	0.00194	0.00237
SumOf6ActiveTrxnFlags	0.00261	0.00186	0.00223
CHANNEL_TYPE_CRM_LANDING_PAGE	0.00191	0.00229	0.0021
ElectronicPurchaseIntAmt	0.0018	0.00239	0.0021
EMPLOYMENT_TYPE_Not set	0.00211	0.00207	0.00209
PROD_SH GOLD MC	0.00171	0.0022	0.00196
DIST_Sparebanken Hedmark	0.0017	0.00221	0.00196
SalesProdCat_Boliglån	0.00195	0.00191	0.00193
CHANNEL_TYPE_-2	0.00171	0.00195	0.00183
ElectronicPurchaseNum	0.00165	0.00184	0.00175
EMPLOYMENT_TYPE_Temp employee	0.00171	0.00165	0.00168

CashIntAmt	0.00141	0.00191	0.00166
EMPLOYMENT_TYPE_Student	0.00165	0.00166	0.00166
CHANNEL_TYPE_INTERNET_BANK	0.00139	0.0016	0.0015
ElectronicPurchaseDomAmt	0.00122	0.00162	0.00142
TAX_CLASS_nan	0.00176	0.00105	0.00141
ElectronicPurchaseIntNum	0.00131	0.00128	0.00129
CashIntNum	0.00115	0.00142	0.00128
EMPLOYMENT_TYPE_Self-employed	0.00152	0.00097	0.00124
EMPLOYMENT_TYPE_Retired	0.00104	0.00132	0.00118
DIST_SpareBank 1 Telemark	0.00094	0.00128	0.00111
HAS_ESTATEMENT_AGREEMENT_IND	0.0005	0.00154	0.00102
EMPLOYMENT_TYPE_Disability benefit	0.00107	0.00094	0.00101
DIST_SpareBank 1 Østfold Akershus	0.001	0.00101	0.001
DIST_SpareBank 1 BV	0.0009	0.0008	0.00085
SalesProdCat_Ung/Student	0.00103	0.00064	0.00083
PROD_SpareBank 1 Visa Gold	0.0006	0.00073	0.00067
TAX_CLASS_2F	0.00059	0.00067	0.00063
ElectronicPurchaseDomNum	0.00054	0.00067	0.00061
DIST_SpareBank 1 Nordvest	0.0007	0.00049	0.0006
DIST_SpareBank 1 Ringerike Hadeland	0.00044	0.0006	0.00052
DIST_SpareBank 1 Søre Sunnmøre	0.00042	0.0003	0.00036
DIST_SpareBank 1 Nøtterøy-Tønsberg	0.00043	0.00028	0.00036
TAX_CLASS_2	0.00024	0.00034	0.00029
DIST_SpareBank 1 Hallingdal Valdres	4e-05	0.00033	0.00018
CHANNEL_TYPE_OPEN_WEB	0.00033	3e-05	0.00018
AppSalesCh_Open web	0.00033	3e-05	0.00018
HasSupplementaryCard	6e-05	0.00025	0.00016
DIST_SpareBank 1 Modum	7e-05	0.0002	0.00013
EMPLOYMENT_TYPE_Other	0.00011	0.00013	0.00012
DIST_SpareBank 1 Gudbrandsdal	8e-05	0.00016	0.00012
CLOSING_BALANCE_AMT	0.00011	9e-05	0.0001
PROD_Sparebank 1 Platinum MC	2e-05	0.00017	9e-05
SalesProdCat_Platinum	2e-05	0.00016	9e-05
MINIMUM_TO_PAY_AMT	0.00011	7e-05	9e-05
EMPLOYMENT_TYPE_Not working	7e-05	5e-05	6e-05
INTEREST_EARNING_LENDING_AMT	6e-05	4e-05	5e-05
SalesProdCat_nan	5e-05	2e-05	3e-05
HAS_DIRECT_DEBIT_AGREEMENT_IND	2e-05	5e-05	3e-05
DIST_SpareBank 1 Lom og Skjåk	1e-05	4e-05	3e-05
RevolvingFlag	3e-05	2e-05	2e-05
TAX_CLASS_0	0.0	0.0	0.0
PROD_SB1 EXTRA MC	0.0	0.0	0.0
TOTAL_PAYMENTS_AMT	0.0	0.0	0.0
EMPLOYMENT_TYPE_Social security	0.0	0.0	0.0
EMPLOYMENT_TYPE_Unemployed	0.0	0.0	0.0

TAX_CLASS_2E	0.0	0.0	0.0
ClosedDuringPeriodFlag	0.0	0.0	0.0

Table 2: Feature importances for the experiments that used a two month training window. The numbers at the end of the feature names indicate month. Features without a number at the end are static features. The table is sorted on average importance and is descending.

Features	Default	Collection	Average
BALANCE_AMT_2	0.04813	0.05295	0.05054
OverLimitFlag_2	0.05744	0.03742	0.04743
AvgBalanceAmt_2	0.03932	0.04484	0.04208
TotalFeeAmt_2	0.04276	0.03883	0.04079
Innbetalinger_sum	0.03418	0.03578	0.03498
PaymentsAmt_2	0.02996	0.03199	0.03097
MaxBalanceAmt_2	0.02761	0.0276	0.0276
TotalFeeNum_2	0.02826	0.02497	0.02662
Innbetalinger_count	0.02414	0.02528	0.02471
MinBalanceAmt_2	0.02549	0.02392	0.0247
TurnoverAtStart	0.02476	0.01988	0.02232
PaymentsNum_2	0.02204	0.02246	0.02225
INTEREST_EARNING_LENDING_AMT_2	0.0214	0.02271	0.02205
SpendingVelocity_2	0.02061	0.02004	0.02033
OverdraftFlag_2	0.02328	0.01161	0.01744
SpendingVelocity_1	0.01868	0.01451	0.0166
CustomerAge	0.01351	0.01797	0.01574
Service Providers_sum	0.01575	0.01564	0.0157
ScoreValueVedOppstart	0.0105	0.01834	0.01442
TAX_NET_INCOME_AMT	0.0135	0.01508	0.01429
TotalFeeAmt_1	0.01435	0.01215	0.01325
TAX_TOTAL_TAX_AMT	0.01253	0.01382	0.01317
TurnoverDomAmt_2	0.01163	0.01415	0.01289
BalanceChangeAmt_2	0.01145	0.01321	0.01233
DaysFromCreatedToFirstUse	0.01234	0.01176	0.01205
TurnoverDomAmt_1	0.01237	0.01138	0.01187
APPL_GROSS_INCOME_AMT	0.01078	0.01191	0.01135
FundTransferAtStart	0.01156	0.00999	0.01078
TurnoverAmt_2	0.00975	0.01135	0.01055
TAX_GROSS_INCOME_AMT	0.00947	0.0113	0.01038
BALANCE_AMT_1	0.01093	0.00966	0.0103
Service Providers_count	0.01003	0.01007	0.01005
CLOSING_BALANCE_AMT_2	0.00931	0.00912	0.00921
TurnoverAmt_1	0.0093	0.00813	0.00872
RevolvingFlag_2	0.00882	0.00822	0.00852
TurnoverDomNum_2	0.007	0.00917	0.00808
PurchaseAmt_2	0.00646	0.0095	0.00798
CashDomAmt_2	0.00923	0.00667	0.00795
MINIMUM_TO_PAY_AMT_2	0.00874	0.00628	0.00751
TotalFeeNum_1	0.00744	0.00703	0.00723

TurnoverNum_2	0.00612	0.00714	0.00663
CashDomNum_2	0.00808	0.0049	0.00649
FundtransferAmt_1	0.00668	0.0061	0.00639
FundtransferAmt_2	0.00627	0.00642	0.00634
CashAmt_2	0.00725	0.00534	0.0063
PurchaseDomAmt_2	0.00489	0.00715	0.00602
TAX_WEALTH_AMT	0.00561	0.00642	0.00602
CreditLimitAmt_2	0.00536	0.00578	0.00557
PurchaseNum_2	0.00489	0.00599	0.00544
CreditLimitAmt_1	0.00517	0.00564	0.00541
Retail Stores_sum	0.00451	0.0062	0.00536
PurchaseDomNum_2	0.00454	0.00617	0.00535
TurnoverIntAmt_2	0.00396	0.00617	0.00507
Others_sum	0.00335	0.00636	0.00485
Miscellaneous Stores_sum	0.00368	0.00595	0.00481
TurnoverDomNum_1	0.00475	0.00473	0.00474
OVERDUE_AMT_2	0.0053	0.0038	0.00455
PurchaseIntAmt_2	0.0032	0.00579	0.00449
FundtransferNum_1	0.00433	0.00422	0.00428
FundtransferNum_2	0.00447	0.00403	0.00425
PurchaseAtStart	0.00342	0.00483	0.00413
OverdueFlag_2	0.00451	0.0036	0.00406
PaymentOverDueFlag_2	0.0046	0.00341	0.00401
Renter_sum	0.0049	0.00311	0.004
CashNum_2	0.00475	0.00313	0.00394
HAS_ESTATEMENT_AGREEMENT_IND_2	0.00207	0.0056	0.00383
TurnoverNum_1	0.0037	0.00384	0.00377
Retail Stores_count	0.00313	0.00427	0.0037
FullpayerFlag_2	0.00434	0.00289	0.00362
PurchaseAmt_1	0.00303	0.00378	0.0034
Others_count	0.00256	0.00425	0.0034
CashAtStart	0.00401	0.00272	0.00337
OverLimitFlag_1	0.00429	0.00216	0.00323
APPL_MORTGAGES_AMT	0.00299	0.00341	0.0032
APPL_OTHER_DEBT_AMT	0.00322	0.00306	0.00314
Miscellaneous Stores_count	0.00286	0.0033	0.00308
TurnoverIntNum_2	0.00257	0.00354	0.00306
SalesProdCat_Ordinær	0.00374	0.00236	0.00305
OverdraftFlag_1	0.0038	0.00208	0.00294
Clothing Stores_sum	0.00214	0.00358	0.00286
PurchaseDomAmt_1	0.00241	0.00307	0.00274
PurchaseIntNum_2	0.00208	0.00311	0.0026
ElectronicPurchaseAmt_2	0.00177	0.00317	0.00247
PurchaseNum_1	0.00227	0.00243	0.00235
SalesProdCat_Kampanje	0.00226	0.00219	0.00222

Transportation_sum	0.00168	0.00269	0.00218
PurchaseDomNum_1	0.0021	0.00226	0.00218
CashIntAmt_2	0.00236	0.00196	0.00216
CashDomAmt_1	0.00243	0.00188	0.00216
Gender_Mann	0.00136	0.00253	0.00194
ACC_STATUS_NUM_2	0.00228	0.0016	0.00194
Renter_count	0.00225	0.00132	0.00179
CashAmt_1	0.00195	0.00161	0.00178
TAX_CLASS_1E	0.00123	0.00224	0.00174
ElectronicPurchaseIntAmt_2	0.00124	0.00212	0.00168
CHANNEL_TYPE_INTERNAL	0.0016	0.00165	0.00162
AppSalesCh_Operatørkanal	0.00156	0.00165	0.00161
DormantFlag_1	0.00147	0.00171	0.00159
TAX_CLASS_1	0.00103	0.00209	0.00156
Utilities_sum	0.00237	0.00072	0.00155
CashDomNum_1	0.00177	0.00124	0.0015
Includes all lodging merchants_sum	0.00123	0.00174	0.00149
Clothing Stores_count	0.00112	0.00179	0.00146
FullpayerFlag_1	0.00134	0.00152	0.00143
TurnoverIntAmt_1	0.00108	0.00157	0.00133
ElectronicPurchaseNum_2	0.00121	0.00141	0.00131
CHANNEL_TYPE_CREDITCARD_BPM	0.0014	0.00116	0.00128
EMPLOYMENT_TYPE_Perm employee	0.00109	0.00141	0.00125
AppSalesCh_Nettbank	0.00116	0.00133	0.00125
CashIntNum_2	0.00126	0.0012	0.00123
Transportation_count	0.00097	0.00134	0.00116
PurchaseIntAmt_1	0.00093	0.00133	0.00113
ElectronicPurchaseDomAmt_2	0.00086	0.00135	0.0011
PaymentsAmt_1	0.00111	0.00107	0.00109
CashNum_1	0.00116	0.00085	0.00101
ElectronicPurchaseIntNum_2	0.001	0.00098	0.00099
Business Services_sum	0.00095	0.00098	0.00097
Utilities_count	0.0015	0.00035	0.00093
ElectronicPurchaseAmt_1	0.00079	0.00102	0.0009
SumOf6ActiveTrxnFlags_1	0.00087	0.00079	0.00083
Airlines_sum	0.00054	0.00111	0.00082
Amusement and Entertainment_sum	0.00074	0.00091	0.00082
AppSalesCh_Responsside	0.0007	0.0009	0.0008
TurnoverIntNum_1	0.00071	0.00084	0.00078
PROD_SpareBank 1 MasterCard Gold	0.00065	0.0009	0.00078
SumOf6ActiveTrxnFlags_2	0.00076	0.00076	0.00076
EMPLOYMENT_TYPE_Not set	0.00078	0.00064	0.00071
PurchaseIntNum_1	0.00065	0.00073	0.00069
ElectronicPurchaseIntAmt_1	0.00061	0.00068	0.00065
PROD_SH GOLD MC	0.00047	0.00076	0.00062

ElectronicPurchaseNum_1	0.00066	0.00054	0.0006
Includes all lodging merchants_count	0.00057	0.00062	0.00059
TOTAL_PAYMENTS_AMT_2	0.00058	0.00061	0.00059
CHANNEL_TYPE_-2	0.0006	0.00054	0.00057
CHANNEL_TYPE_INTERNET_BANK	0.00067	0.00043	0.00055
SalesProdCat_Boliglån	0.00046	0.00059	0.00053
Pro. Services and Organizations_sum	0.00049	0.00056	0.00053
ElectronicPurchaseIntNum_1	0.00061	0.00042	0.00052
CHANNEL_TYPE_CRM_LANDING_PAGE	0.0004	0.00063	0.00051
ElectronicPurchaseDomNum_2	0.00046	0.00055	0.00051
APPL_STUDENT_LOAN_AMT	0.00039	0.00061	0.0005
Business Services_count	0.00048	0.00047	0.00048
TAX_CLASS_nan	0.00067	0.00026	0.00047
PaymentsNum_1	0.00039	0.00052	0.00046
HAS_DIRECT_DEBIT_AGREEMENT_IND_2	0.00036	0.00053	0.00045
Amusement and Entertainment_count	0.0004	0.00046	0.00043
Airlines_count	0.00036	0.00046	0.00041
ElectronicPurchaseDomAmt_1	0.00034	0.00043	0.00038
EMPLOYMENT_TYPE_Student	0.00038	0.00032	0.00035
CHANNEL_TYPE_AUTH_WEB	0.00045	0.00022	0.00034
AppSalesCh_Autentisert web	0.00046	0.00022	0.00034
EMPLOYMENT_TYPE_Disability benefit	0.00028	0.00031	0.00029
Pro Services and Organizations_count	0.00031	0.00027	0.00029
EMPLOYMENT_TYPE_Temp employee	0.00034	0.00024	0.00029
EMPLOYMENT_TYPE_Retired	0.00025	0.0003	0.00027
DormantFlag_2	0.00019	0.00027	0.00023
CashIntAmt_1	0.00021	0.00025	0.00023
PROD_SpareBank 1 Visa Gold	0.00026	0.0002	0.00023
EMPLOYMENT_TYPE_Self-employed	0.00017	0.00028	0.00022
Wholesale Distrubutors_sum	0.00011	0.00033	0.00022
SalesProdCat_Ung/Student	0.0002	0.00019	0.00019
Mail Order / Tlf Order Providers_sum	0.0002	0.00015	0.00018
TAX_CLASS_2F	0.0002	0.00013	0.00017
CashIntNum_1	0.00014	0.00019	0.00016
ElectronicPurchaseDomNum_1	0.00017	0.00016	0.00016
Contracted Services_sum	0.00013	0.00016	0.00014
Wholesale Distrubutors_count	7e-05	0.00021	0.00014
Repair Services_sum	4e-05	0.00023	0.00014
Automobile / Vehicle Rental_sum	0.0001	0.00016	0.00013
Annet_sum	0.00018	5e-05	0.00012
Government Services_sum	4e-05	0.00018	0.00011
HAS_ESTATEMENT_AGREEMENT_IND_1	6e-05	0.00015	0.00011
Mail Order / Tlf Order Providers_count	0.00012	9e-05	0.00011
Annet_count	0.00015	3e-05	9e-05
Contracted Services_count	8e-05	8e-05	8e-05

TAX_CLASS_2	8e-05	7e-05	8e-05
Automobile / Vehicle Rental_count	7e-05	8e-05	8e-05
Repair Services_count	3e-05	0.00012	8e-05
EMPLOYMENT_TYPE_Other	2e-05	0.00013	8e-05
HasSupplementaryCard	2e-05	0.00011	6e-05
CreditLimitChangeAmt_2	7e-05	3e-05	5e-05
Government Services_count	3e-05	7e-05	5e-05
CreditLimitIncreaseFlag_2	7e-05	2e-05	5e-05
EMPLOYMENT_TYPE_Not working	8e-05	0.0	4e-05
OPENING_BALANCE_AMT_2	6e-05	2e-05	4e-05
CLOSING_BALANCE_AMT_1	5e-05	2e-05	3e-05
INTEREST_EARNING_LENDING_AMT_1	5e-05	2e-05	3e-05
MINIMUM_TO_PAY_AMT_1	5e-05	1e-05	3e-05
SalesProdCat_Platinum	1e-05	5e-05	3e-05
PROD_Sparebank 1 Platinum MC	1e-05	5e-05	3e-05
InterestPostedAmt_2	3e-05	2e-05	2e-05
SumOf6ActiveStmtFlags_2	3e-05	1e-05	2e-05
RevolvingFlag_1	3e-05	1e-05	2e-05
SumOf12ActiveStmtFlags_2	2e-05	1e-05	2e-05
CHANNEL_TYPE_OPEN_WEB	2e-05	0.0	1e-05
SalesProdCat_nan	0.0	1e-05	0.0
HAS_DIRECT_DEBIT_AGREEMENT_IND_1	0.0	1e-05	0.0
AppSalesCh_Open web	1e-05	0.0	0.0
FirstDunningFlag_2	0.0	0.0	0.0
TOTAL_PAYMENTS_AMT_1	0.0	0.0	0.0
EMPLOYMENT_TYPE_Social security	0.0	0.0	0.0
CreditLimitDecreaseFlag_2	0.0	0.0	0.0
EMPLOYMENT_TYPE_Unemployed	0.0	0.0	0.0
TAX_CLASS_0	0.0	0.0	0.0
PROD_SB1 EXTRA MC	0.0	0.0	0.0
TAX_CLASS_2E	0.0	0.0	0.0