UNIVERSITEIT UTRECHT

MASTER THESIS

Credit risk modeling using a weighted support vector machine

Author:
Jesper DE GROOT

Supervisor:
Prof. Dr. Jason FRANK
Dr. Diederik FOKKEMA

A thesis submitted in fulfillment of the requirements for the degree of Master of Science in MATHEMATICS

at the

Mathematical Institute

combined with an internship at

EY Building a better working world

September 2, 2016

Keywords: machine learning, stochastic gradient descent, support vector machine, weighted support vector machine, credit risk modeling, probability of default

Declaration of Authorship

I, Jesper DE GROOT, declare that this thesis titled, “Credit risk modeling using a weighted support vector machine” and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

UNIVERSITEIT UTRECHT

Abstract

Faculty Name
Mathematical Institute

Master of Science in MATHEMATICS

Credit risk modeling using a weighted support vector machine

by Jesper DE GROOT

The modeling of credit risk is traditionally based on approaches such as linear regression or multiple discriminant analysis. There are several limitations to these methods, in particular their inability to adapt to new data and the assumption of a certain (linear) relation between the dependent and independent variables. In this thesis we will use a new machine learning technique called *weighted support vector machine* combined with *averaged stochastic gradient descent*. Using this approach, we create a classification of a data set containing mortgage loans of Freddie Mac into several groups with increasing probabilities of default. This method shows promising results, both in terms of predictive and discriminatory power, especially when information about the monthly performance of these loans and the macro-economic situation is included. If the performance on other data sets is similar, the technique can be implemented for credit risk modeling.

Acknowledgements

At first, I would like to thank my project advisor Prof. Dr. Jason Frank of the Department of Mathematics at the Utrecht University for his continuous support and valuable comments to my master thesis.

My sincere thanks also goes to Dr. Diederik Fokkema, who was my daily advisor at EY and provided me the opportunity to join the FS Risk management team as an intern. I thank my EY colleagues, who really gave me some helpful insights and made my internship a perfect time.

Contents

Declaration of Authorship **iii**

Abstract **v**

Acknowledgements **vii**

1 Introduction **1**
1.1 Problem definition 3
1.2 Set-up of this thesis 4

2 Credit risk regulation and modeling **5**
2.1 Financial terms 5
2.2 Regulations regarding credit risk and credit risk modeling 7
2.3 Traditional credit rating approaches 8
2.4 Multivariate credit scoring models 9
2.5 Summary 12

3 Data properties **13**
3.1 Properties of the data set 13
3.2 Variables 14
3.3 Conversion of variables 17
3.4 Selection of variables 17
3.5 Data quality 22

4 Support vector machines **23**
4.1 Definition of machine learning 23
4.2 Set-up 24
4.3 Primal and dual problem 28
4.4 Extensions 32
4.5 Summary 32

5 Gradient descent techniques **33**
5.1 Introduction of the gradient descent algorithm 33
5.2 Convergence of the gradient descent algorithm 35
5.3 Subgradient descent 39
5.4 Stochastic (sub)gradient descent 41
5.5 Summary 44

6 Weighted SVM and averaged SGD **47**
6.1 Weighted SVM 47
6.2 Convergence of averaged SGD 52
6.3 Summary 58

7 Model development, results and testing **59**
7.1 Model development 59
7.2 Performance measure 60
7.3 Results for a single model 63
7.4 Overall results 69
7.5 Comparison with other models 73

8 Conclusion **75**
8.1 Opportunities 76

Bibliography **77**

A Optimization Theory **81**
A.1 Duality principle 81
A.2 Convex optimization 82
A.3 Farkas Lemma 83
A.4 Strong Duality Theorem 86
A.5 Convex dual problem 87
A.6 Karush-Kuhn-Tucker Theorem 87

List of Figures

1.1 U.S. Household debt to GDP ratio 1
1.2 Number of mortgages and average credit score in FHLMC data set (yearly observations) 2
1.3 Default percentages of FHLMC data set and U.S. mortgage delinquency rate (yearly observations) 3

2.1 Schematic illustration of multiple discriminant analysis 11

3.1 CDFs of Student's t-distributions with different degrees of freedom ν . . 20

4.1 Linear support vector machine 25
4.2 Nonlinear support vector machine 27

5.1 Gradient descent 34
5.2 Stochastic gradient descent vs. gradient descent 42
5.3 Convergence of (S)GD algorithm 45
5.4 Graph of hinge loss function 45
5.5 Convergence rate of (stochastic) subgradient descent 46

6.1 Plot of g(t) for $\alpha = 0.3$ 55
6.2 Plot of g(t) for $\alpha = 0.7$ 56
6.3 Convergence rates of exponential decay averaging scheme for different values of c 58

7.1 Schematic overview of the splitting of the data set 60
7.2 Different ROC curves 62
7.3 CAP curves without monthly performance data 65
7.4 CAP curves with monthly performance data 67
7.5 CAP curves with macro-economic variables 68
7.6 AUC values for different time horizons 69
7.7 AUC for PD-horizon of 1 year using different training set sizes 71

List of Tables

3.1 Variable definitions and types 16
3.2 Correlation between macro-economic variables 18
3.3 Correlation between numeric variables in the origination file of 1999Q1 . 18
3.4 Results of mean differences test for 1999Q1 21
3.5 Valid values for five variables in the origination file as determined by Freddie Mac 22

5.1 Convergence rates of gradient descent algorithms 44

7.1 Contingency table of binary classification 61
7.2 Classification table for PDs 62
7.3 1-year PD prediction 63
7.4 3-year PD prediction 63
7.5 5-year PD prediction 64
7.6 10-year PD prediction 64
7.7 1-year PD prediction with monthly performance variables 65
7.8 3-year PD prediction with monthly performance variables 66
7.9 5-year PD prediction with monthly performance variables 66
7.10 10-year PD prediction with monthly performance variables 66
7.11 1-year PD prediction with macro-economic variables 67
7.12 3-year PD prediction with macro-economic variables 68
7.13 5-year PD prediction with macro-economic variables 68
7.14 Test results for different data sets and intermediate time periods 70
7.15 Coefficients of the 3 hyperplanes for a 1-year PD prediction 72
7.16 Coefficients of the 4 hyperplanes for a 3-year PD prediction 72
7.17 Coefficients of the 4 hyperplanes for a 5-year PD prediction 72
7.18 Coefficients of the 4 hyperplanes for a 10-year PD prediction 72
7.19 Performance results of the credit scores method and linear regression . . 73

List of Abbreviations

ANN	**A**rtificial **N**eural **N**etwork
AUC	**A**rea **u**nder **C**urve
BCBS	**B**asel **C**ommittee on **B**anking **S**upervision
BIS	**B**ank for **I**nternational **S**ettlements
CAP	**C**umulative **A**ccuarcy **P**rofile
CDF	**C**umulative **D**istribution **F**unction
CLTV	**C**ombined **L**oan-**to**-**Value**
DTI	**D**ebt-**t**o-**I**ncome
EAD	**E**xposure **a**t **D**efault
EL	**E**xpected **L**oss
FHLMC	**F**ederal **H**ome **L**oan **M**ortgage **C**orporation
FPR	**F**alse **P**ositive **R**ate
GD	**G**radient **D**escent
IASB	**I**nternational **A**ccounting **S**tandards **B**oard
KKT	**K**arush-**K**uhn-**T**ucker
LGD	**L**oss **G**iven **D**efault
LTV	**L**oan-**to**-**V**alue
PD	**P**robability of **D**efault
ROC	**R**eceiver **O**perator **C**haracteristics
SGD	**S**tochastic **G**radient **D**escent
SVM	**S**upport **V**ector **M**achine
TPR	**T**rue **P**ositive **R**ate
UPB	**U**npaid **P**rincipal **B**alance

Chapter 1

Introduction

One of the main businesses of a bank is lending money to its customers, in particular to firms and individuals. For individuals, the loans mostly consist of mortgages that are used to buy a house. As a future house owner, one would like to choose the bank that offers a loan with the best agreements, i.e. with the smallest interest rate. On the other hand, a bank (or another financial institution issuing loans) needs to deal with the risk that a borrower cannot fulfill the repayment agreements and wishes to cover this with a risk premium. To assess this 'credit risk', a bank usually collects a huge amount of information on borrowers and the underlying property of the loan, which is the house. In most cases this results in a so called credit score, which is used to determine if a borrower will receive a mortgage loan or not. If a bank sets the interest rate too high, a customer will turn to another bank that is offering a lower interest rate. However, the downside of consistently setting the interest rate too low is that defaults may not be covered properly and the bank can obtain a serious loss on the entire portfolio of loans.

In the period 1955-2005 the U.S. household debt (which consists for a large part of mortgages) as percentage of the GDP has raised from 40% to a record high 130% (see Figure 1.1). Due to this enormous increase, it has become more and more important to assess and manage the risk of the household debt, as problems with repayment of these debts can cause bankruptcies that influence the entire economy of a country. The high number of defaults of house owners in the U.S. was one of the main reasons for the downfall of several financial institutions in September 2008 and the worldwide financial crisis of 2007-2009 (Mian and Sufi, 2014).

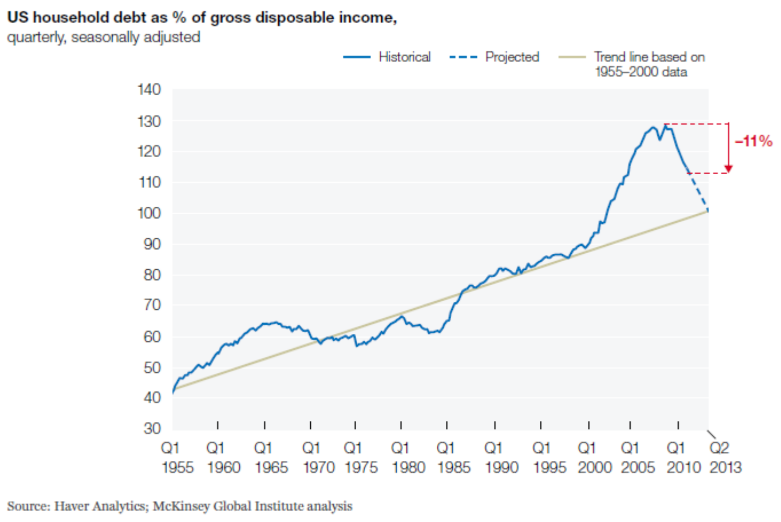

FIGURE 1.1: U.S. Household debt to GDP ratio

Since the 1970s regulators have required financial institutions to hold certain capital reserves to cover unexpected losses. These minimum capital requirements are specified in the frameworks known as Basel I (1988), Basel II (2004) and Basel III (expected implementation in 2019), and motivate banks to use more advanced models for predicting the default probability of a certain loan. Regulators such as the ECB perform stress tests for credit risk, in order to assess how banks and their outstanding loans perform under certain 'bad' scenarios. Under the new financial accounting standards (IFRS 9) financial institutions will also be required to adopt a forward-looking approach, that is based on (macro-)economic predictions, when determining credit risk.

In this thesis we are using a data set from the Federal Home Loan Mortgage Corporation (FHLMC), also known as Freddie Mac. This set contains mortgage loans originated between January 1999 and June 2015. In Figure 1.2 we see a graph with the number of mortgages issued in a certain year and the average credit scores given to these mortgages. In the period 2001-2007, prior to the crisis, we see a slight increase in the average credit scores given to the borrowers that successfully applied for a loan in these years. One would therefore predict that the default percentages for these loans are slightly smaller.

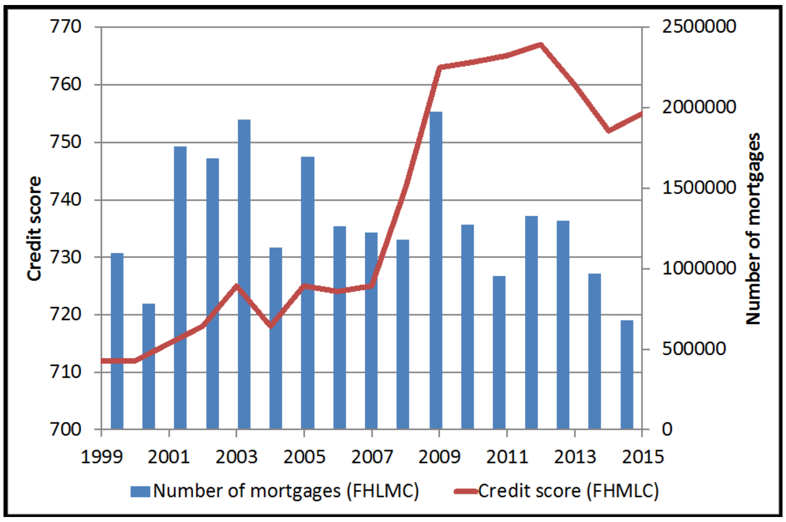

FIGURE 1.2: Number of mortgages and average credit score in FHLMC data set (yearly observations)

However, in Figure 1.3 on the next page we see in blue a drastic increase in the default rate during the period 2001-2007. Note that the defaults are assigned to the year of the origination of the defaulted loan, and are determined over the entire lifetime of the loan. Therefore we see the highest default percentage in the period 2006-2007, whereas in Figure 1.3[1] we observe in red that the period 2009-2012 contains the most actual defaults in the U.S.

We can conclude a few things from these graphs. At first we see that the slightly increasing trend in the credit scores does not coincide with the decrease in default percentage, i.e. the credit scores before the financial crisis were not correctly predicting the decreasing creditworthiness of the customers during the crisis. Furthermore, after the crisis, credit scores have increased a lot, as obtaining a mortgage is much more difficult than before.

[1]All (macro-)economic data used in this thesis is obtained from the data base of the Fred (Federal Reserve Bank of St. Louis) Economic Research.

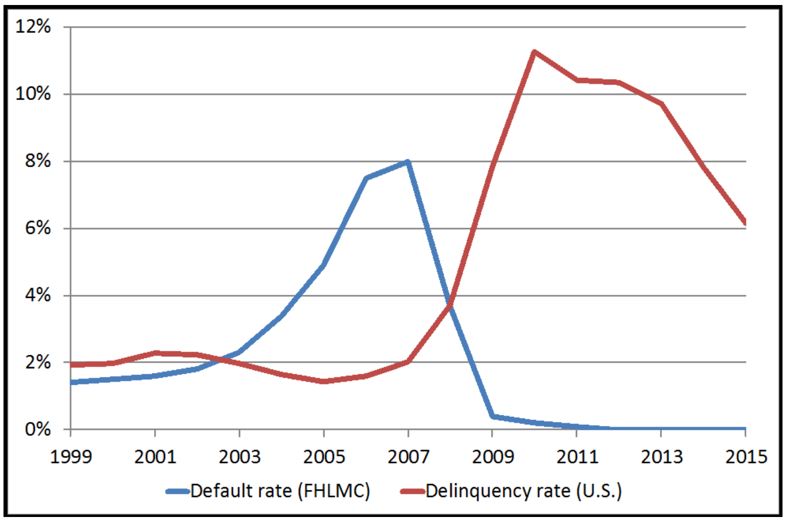

FIGURE 1.3: Default percentages of FHLMC data set and U.S. mortgage delinquency rate (yearly observations)

1.1 Problem definition

As we have seen, the credit rating system in use by Freddie Mac before the financial crisis was totally incapable of predicting the increased number of defaults after 2007. Therefore, there is need for another model for the prediction of default rates on this data set. Since the increase in the number of defaulted loans coincided with the worldwide crisis, it seems worthwhile to look at the possibility of including macro-economic variables in our model. In this thesis we will develop such a model using *machine learning*.

Machine learning is essentially teaching a computer to learn from training data and to make predictions on other data, using certain algorithms. As the amount of accessible data and computing power has increased over the past 25 years, machine learning techniques have been implemented widely and have shown many promising results, including the field of credit risk modeling.

In this thesis we look at the so called *multiclass support vector machine (SVM)*. This is a classification problem, i.e. it classifies mortgage loans into multiple classes, with different default probabilities, The classes are developed using historical data and default information. The unique feature of this technique is that it creates rating classes by itself, i.e. it does not need to have a ranking of the training data into different classes, such as AAA, AA+ etc. The optimization will be performed using a stochastic gradient descent technique. This optimization algorithm is capable of handling a large training set, and has also gained a lot of attention over the last two decades.

To summarize, the goal of this thesis is to develop a credit risk model using machine learning techniques and including macro-economic variables.

1.2 Set-up of this thesis

Chapter 2 consists of the definition of credit risk and some information about credit risk regulation and modeling. Furthermore also some very important financial terms are explained. Chapter 3 contains an introduction and analysis of the data set of FHLMC. We also perform some statistical analysis on the variables that are available in this data set. Chapter 4 consist of a description of machine learning and in particular the support vector machine. The fifth chapter introduces the gradient descent and the stochastic gradient descent and several other related methods. It also includes some statements on the convergence of these different optimization technique. Chapter 6 focuses on three new theorems that are developed and proven in this thesis, regarding a modification of the support vector machine and the averaged version of the stochastic gradient descent. Finally, Chapter 7 contains the results that are obtained after the development of a model. Also, the performance results and comparisons with other traditional models can be found in this section. Chapter 8 contains the conclusion and some opportunities for further research. For some mathematical background that is in particular needed for chapters 4 and 5, we refer to Appendix A. Especially chapters 5 and 6 are mathematically more complex, whereas chapter 2 and 3 are more based on statistics and finance.

Chapter 2

Credit risk regulation and modeling

The *Bank for International Settlements (BIS)* states the following about *credit risk* in the Principles for the Management of Credit Risk (BIS, 2000)

"Credit risk is most simply defined as the potential that a bank borrower or counterparty will fail to meet its obligations in accordance with agreed terms. The goal of credit risk management is to maximize a bank's risk-adjusted rate of return by maintaining credit risk exposure within acceptable parameters. Banks need to manage the credit risk inherent in the entire portfolio as well as the risk in individual credits or transactions. Banks should also consider the relationships between credit risk and other risks. The effective management of credit risk is a critical component of a comprehensive approach to risk management and essential to the long-term success of any banking organization."

In our concrete situation, credit risk is the potential that a borrower of a mortgage loan cannot fulfill the scheduled payments agreed on.

During the last 50 years regulators have issued different capital requirements for financial institutions to manage in particular their credit risk. They are mainly set by the *Basel Committee on Banking Supervision (BCBS)*, which is part of the BIS, or by the *International Accounting Standards Board (IASB)*. We will give a quick overview of these regulations in this chapter, but first we start with some important definitions of financial terms. Additionally, several traditional approaches are discussed that are used for modeling credit risk.

2.1 Financial terms

2.1.1 Defaults

This entire thesis is about defaults and the probability that they occur. A *default* is an event that a borrower cannot meet his contractual obligations. The exact definition of what this means differs from institution to institution. In the Basel II and Basel III framework every loan that is 3 months past due, is considered a default. The probability that a certain loan defaults in some time horizon (usually one year or the lifetime of a loan) is called the *probability of default (PD)*. The *recovery rate* denotes the percentage of the borrowed amount that can be recovered by the lender in case of a default of the borrower, e.g. by organizing an auction for the house. The *loss given default (LGD)* equals 1 minus the recovery rate, i.e. the percentage of the amount we lose in case of a default. Finally, the parameter *exposure at default (EAD)* denotes the amount that the borrower is expected to owe the lender at the time of a default. Then the *expected loss*

(EL) is defined as follows

$$\mathrm{EL} = \mathrm{PD} * \mathrm{LGD} * \mathrm{EAD} \tag{2.1}$$

Essentially, in the case of a default we expect to lose LGD $*$ EAD and in the other case we do not lose anything. The *unexpected loss* is the loss that is realised above the expected loss, which is caused by a default rate higher than the probability of default or an unexpected increase in LGD or EAD. Note that an unexpected loss can also be negative, if the actual loss is smaller than an expected loss.

2.1.2 Mortgage-specific terms

We also introduce some terms that are used in our data set of mortgages. This data set will be introduced in more detail in Chapter 3. The date when the mortgage loan is initiated is called the *origination date* and the date of the originally appointed last payment is called the *maturity date*. In this thesis we work with a data set with only *fully amortized mortgages*, which means that at the maturity date the debt is fully paid off, in contrary to mortgages with interest-only payments. We use the term *collateral* for the underlying property (house or building) that serves as a security for the repayment of a loan, i.e. if a borrower goes into default, the lender can regain at least a part of the borrowed amount by selling the collateral.

Ratios

There are four different ratios in the data set that are of high importance.

At first we have the *unpaid principle balance (UPB)*. This is the amount of the loan that has not yet been paid off to the lender. The original UPB is the amount that is initially borrowed by the borrower. In the case of an amortized mortgage, each monthly payment consists of two parts, an interest payment and a principal payment. The interest payment is the amount of interest over the last month and the principal payment the amount of the original loan that is paid off in this particular month. After this payment the UPB is decreased with only the principal payment. In the case of a fully amortized mortgage, the UPB equals zero at the maturity date of the loan.

Two also very important elements of the data set are the *loan-to-value (LTV)* and the *combined loan-to-value (CLTV)*. The precise definition of these terms differs from data set to data set. The CLTV ratio is the ratio of all loans that are secured by the property and the value of the underlying collateral. This happens for example if the borrower receives a second mortgage loan to improve a certain part of the house. The LTV ratio is the ratio of the primary mortgage loan, i.e. only the loan with the highest value, and the value of the underlying collateral.

Finally we have the *debt-to-income (DTI)* ratio, which is (at the origination date) simply the ratio of the monthly debt payments by the borrower to the lender divided by the total monthly income of the borrower specified at the date of the origination.

2.2 Regulations regarding credit risk and credit risk modeling

2.2.1 Basel I

In 1988 the BCBS, a committee of international banks, created the Basel Capital Accord (Basel I), which requires banks to divide their outstanding loans into classes with similar types of borrowers. Each class has its own fixed risk, i.e. corporate loans pose 100% risk to a bank, where mortgage loans only pose 50% risk to a bank, as there is an underlying collateral. There is no differentiation in creditworthiness of customers in the same class, which is one of the weaknesses of the Basel I framework. Basel I basically requires financial institutions to hold enough capital reserves such that their capital ratio exceed 8%, with the capital ratio defined as

$$\text{capital ratio} = \frac{\text{total capital}}{\text{total credit risk}} \tag{2.2}$$

In the Basel I framework, the total credit risk is defined as the sum of the current UPB over all classes of borrowers weighted with the risk weights for the different types of classes. The minimum capital requirement K is defined as

$$K = 0.08 * \text{total credit risk} \tag{2.3}$$

2.2.2 Basel II

As Basel I was not (sufficiently) distinguishing between borrowers with a high and a low creditworthiness, new regulations were developed. In 2004 the New Basel Capital Accord (Basel II) was issued. In this case the capital ratio is defined as

$$\text{capital ratio} = \frac{\text{total capital}}{\text{total credit risk + market risk + operational risk}} \tag{2.4}$$

Market risk and operational risk are also included in the capital ratio, but they are not in the scope of this thesis. Therefore we ignore these from now on in the expressions of the minimum capital requirement. To calculate the capital ratio, banks are now allowed to use their own best approaches to determine their capital requirements for credit risk, if these approaches are approved by the regulator. We can divide these methods into two different categories, the *standardized approach* and the *internal ratings-based (IRB) approach*, which relies on the developed experience from the bank itself in determining credit risk. There are also two types of IRB approaches, namely the *foundation IRB* and the *advanced IRB approach*. They differ in the fact that the foundation IRB approach only allows banks to provide estimates for the PD and requires them use estimates for the LGD and EAD that are provided by the BCBS, as when using the advanced IRB approach a bank is also allowed to provide its own estimates for the LGD and EAD.

In the case of the standardized approach the minimum capital requirements are given by

$$K = 0.08 * \sum_i \text{EAD}_i * \text{RW}_i \tag{2.5}$$

where EAD_i and RW_i are the exposure at default and the risk weight of class i respectively. This approach is basically as before in the Basel I framework. The risk weight is set to 35% for the class of mortgage loans and 75% for the class of other retail credit, such as credit card loans.

In the case of the IRB approach the minimum capital requirements for credit risk only on mortgage loans is given by

$$K = \mathrm{LGD} * \Phi\left(\sqrt{\frac{1}{0.85}}\Phi^{-1}(\mathrm{PD}) + \sqrt{\frac{0.15}{0.85}}\Phi^{-1}(0.999)\right) \tag{2.6}$$

Note that the EAD is not included in this expression, but it is in the capital requirements for credit risk on other loans. The IRB approach is favored and adopted by many banks, as it generally leads to less conservative, i.e. smaller, values for the minimum capital requirements. It now basically comes down on which method to use to compute the PDs. The approaches to estimate the LGD and EAD are not in the scope of this thesis.

2.2.3 IFRS 9

The new international standard for financial reporting (*IFRS 9*), developed by the International Accounting Standards Board (IASB), will become active in 2018. In the report "Financial Instruments: Expected Credit Losses" (IFRS, 2013), it is proposed to measure lifetime credit risk based on a forward-looking approach, measuring expected credit losses, instead of a backward-looking approach (based on historical data and statistics) that was in use before. To fulfill this requirement, financial institutions have to adapt their credit models, and include forecasts for several macro-economic variables to predict default events. One of the main goals for this thesis is to also include these variables in the SVM and use them to correctly predict default probabilities.

2.3 Traditional credit rating approaches

The traditional approaches to estimate the PD of a borrower can be divided into two different systems, namely expert systems and credit scoring models (Saunders and Allen, 2002 & Allen, DeLong, and Saunders, 2004).

Expert systems were initially based on the expertise of humans. As they might be inconsistent and subjective in their judgments on the creditworthiness, artificial neural networks (ANN) have been introduced to forecast PDs based on the specifications of the borrower, using historical observations of defaulted and non-defaulted loans. A neural network continuously adapts itself when new default data is passing through the network. Therefore it can incorporate changing conditions.

In the training process an ANN tries to determine coefficients for every (useful) variable in the historical data set. However, an ANN can grow very large as more and more data is presented, and therefore it can be very costly to implement and maintain it. Also, due to the fact that it can have many hidden intermediate connections, it can be very intransparent. Since the intermediate steps do not have any clear economic interpretation, these steps cannot be checked for plausibility and accuracy.

Nonetheless, the overall performance of a neural network can be very good. In 1991 a supervised version of an ANN is used for predicting bankruptcies (Kim and Scott, 1991). The system performs well for the prediction of bankruptcies in one year (87% of the defaults are predicted), but its performance declines over time to 47% over 3 years.

Secondly we have the *credit scoring models*. The most famous (external) rating systems are obviously those of Moody's, Standard & Poor's and Fitch, which make a judgement on the creditworthiness of countries and corporations. However, many banks develop their own *internal rating system* to classify a certain loan in a certain category, such as AAA or AA+.

The most widely used credit risk approaches are the classical multivariate credit scoring methods. We will investigate the following four traditional methods in the next section.

- Linear regression
- Logit model
- Probit model
- Multiple discriminant analysis model

The essential goal of these models is to identify which variables have the most discriminatory power in differentiating the defaulted loans from the non-defaulted loans. Based on this analysis, model parameters are estimated and PDs of loan applicants can be obtained.

As these credit scoring methods are relatively easy concepts, they are both easy to check for plausibility and accuracy. In the U.S., Fair Isaac and Co. Inc. (FICO) developed a credit scoring system that is widely used, also by Freddie Mac. On myfico.com a personal credit score between 300 and 850 can be obtained, based on various indicators, such as credit card debts and payment history. However, personal information, such as salary, race or religion, which can also be valuable for predicted creditworthiness, is excluded. Furthermore, another shortcoming of credit scoring models is that they usually assume some kind of linearity of the independent variables, which may be not valid.

2.4 Multivariate credit scoring models

In this section we discuss the different traditional approaches to obtain a credit score.

2.4.1 Linear regression

First we look at the *multiple linear regression* with ordinary least squares (OLS). In this case we want to predict a variable $y_i \in \{1, ..., d\}$, which is the number of the rating class to which a certain loan belongs, with a vector $x_i \in \mathbb{R}^n$ containing information about the borrower. This regression fits the following model to the training data.

$$y_i = w^T x_i + b + \epsilon \tag{2.7}$$

where w is a vector in $\mathbb{R}^n$ containing the coefficients, b is a scalar and ϵ is the residual. We know from statistics that the best estimators for this model are obtained from ordinary least squares. The following key assumptions are done when considering linear regression as a valid model.

- Linear relationship: The relation between x_i and y_i certainly needs to be (close to) linear, to obtain a useful model.

- Multivariate normality: Regression requires the independent variables to be multivariate normally distributed. Non-normally distributed variables can distort relationships and tests for significance.
- Little or no multicollinearity: The independent variables are not allowed to be too much correlated.
- No auto-correlation: The value of y_i may not depend on y_{i-1} or other observations.
- Homoscedasticity: The error term ϵ should be independent identically distributed. The expected value of ϵ^2 is constant, equal to σ^2 and does not depend on the value of x_i, i.e. the characteristics of a single observation does not influence the residual.

There is another problem with the use of this model for assessing credit risk. The dependent variable is an ordinal variable, with values between 1 and d, whereas a linear regression requires a continuous target variable. One could propose that all values between two certain values, e.g. $y_i - 0.5$ and $y_i + 0.5$, belong to class i. This is still not an ideal situation, as the 'risk' difference between two classes will be equal, i.e. the difference in PD between class 1 and 2 will be the same as the difference in PD between class $d-1$ and d. This certainly need not be the case. More information on multiple linear regression can be found in Kutner et al., 1974.

2.4.2 Ordered logit and probit regression

Both in a logit and a probit regression we fit the following model to the training data

$$y_i^* = w^T x_i + \epsilon \tag{2.8}$$

where ϵ is assumed to be $\mathcal{N}(0,1)$-distributed in the case of a probit regression and in the case of a logit regression ϵ is assumed to be distributed following the standard logistic distribution. In both regressions we have y_i^* as a continuous indicator for y_i, i.e.

$$y_i = \begin{cases} 0 & \text{if } y_i \leq 0 \\ 1 & \text{else} \end{cases} \tag{2.9}$$

In this simple case, y_i can only take the values 0 and 1, or ± 1 in other definitions. Since we want a classification in d different classes, this is not suitable.

Therefore we look at an *ordered logit* or *probit regression*. In this case we take $d-1$ values μ_j for $1 \leq j \leq d-1$ and we set

$$y_i = \begin{cases} 1 & \text{if } y_i \leq \mu_1 \\ 2 & \text{if } \mu_1 < y_i \leq \mu_2 \\ \vdots & \\ d & \text{if } \mu_{d-1} < y_i \end{cases} \tag{2.10}$$

In a logit or probit regression model we assume that the target variable is ordinal, so we do not have a problem here with our categorization. However, still a few significant assumptions are made.

- Logistic/probit relationship: The relationship between the target variable and the independent variables should be following the logistic distribution (in the ordered logit) or the inverse normal distribution (in the ordered probit).

- Independent error terms: The error terms ϵ should be independent identically distributed.
- Little or no multicollinearity: The independent variables are not allowed to be too much correlated.
- Large sample size: We need a large sample size to compute the coefficients using the maximum likelihood function.

More information on ordered probit and logit regression can be found in Draper and Smith, 1998

2.4.3 Multiple discriminant analysis

The main idea of *multiple discriminant analysis (MDA)* is to identify the variables that have the most discriminatory power in separating the classes. The procedure attempt to construct so called *discriminatory functions*, which can be seen as 'axes', from linear combinations of the original variables. Each axis is constructed such that it maximes the differences between groups and such that they are mutually orthogonal. Therefore, we first obtain the functions with a high discriminatory potential. In a sense, it is very similar to a principal component analysis. The following illustration shows how these functions can separate the different groups.

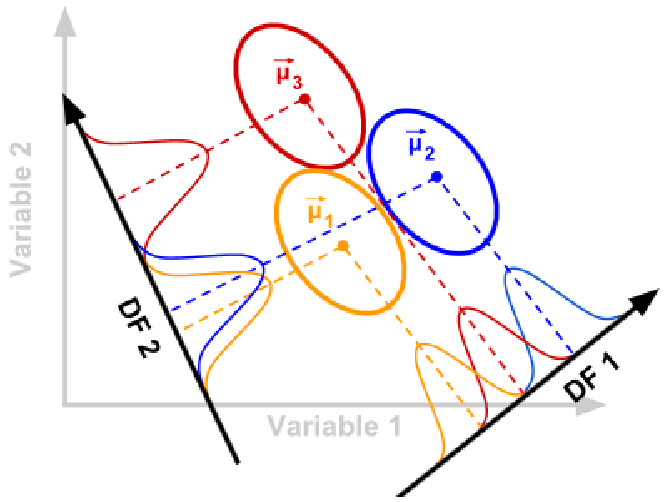

FIGURE 2.1: Schematic illustration of multiple discriminant analysis

In this case, two discriminatory functions (DF 1 and DF 2) turn out to be good in separating the three different classes.

We assign an observation x_i to class j if we have for all $k \in \{1, ..., d\} \setminus j$

$$\frac{f_j(x_i)}{f_k(x_i)} > \frac{\pi_j l(j,k)}{\pi_k l(k,j)} \tag{2.11}$$

In the above expression, f_j and f_k are the probability density functions of class j and k respectively. Furthermore, π_j and π_k are simply the probabilities of a certain element to be in class j and k, and $l(\cdot,\cdot)$ is a so called *loss function*, which gives the cost of misclassification, i.e. the cost of classifying one object in the first class, while it actually belongs to the second class. For example we can classify a certain package of loans in a low-risk class, whereas they are actually high-risk loans. This can result in a loss when a series of default occurs.

It now comes down on the choice of the probability density functions and the loss function. Usually, normal density functions are used. Again, there are several quite strong assumptions made on the data in order to perform an MDA.

- The independent variables are close to multivariate normally distributed.
- There should be little or no multicollinearity between the independent variables.
- The covariance matrices for each class should be close to equal.
- (Multivariate) linear functions should be possible to discriminate between the different classes.

More information on multiple discriminant analysis can be found in Duda, Hart, and Stork, 2001.

2.5 Summary

In all credit risk approaches that are described in the previous section, we have to make assumptions on both the relations between the independent variable and the target variables and the mutual relations between the independent variables. In the first case we have seen linear, logistic or inverse normal relations, whereas in the latter case the approaches required independence or at least little multicollinearity. This can be a huge shortcoming of these methods. In machine learning there are several approaches that do not assume any clearly defined pattern, such as the dual version of the support vector machine that we will see in Chapter 4.

Chapter 3

Data properties

The data for this thesis is acquired from the Federal Home Loan Mortgage Corporation (FHLMC), better known as *Freddie Mac*, which is a financial services corporation acting on the secondary market for mortgages. This means that Freddie Mac buys mortgages from other banks and sells them to investors as a *mortgage-backed security (MBS)*, which consists of a single mortgage or a package of mortgages. An MBS is essentially a way for smaller banks to lend money to home buyers without having to worry if the clients are able to repay the loan, as it is sold as a package to investment banks or other private investors, with the object as collateral. Such a smaller bank then acts as an intermediary between a home buyer and the investment market. Two specific types of MBS are collateralized mortgage obligations (CMOs) and collateralized debt obligations (CDOs). In the previous decade, especially for the latter type, credit ratings turned out to be completely overestimated, leading to the U.S. subprime mortgage crisis of 2007-2009 and the federal takeover of Freddie Mac in September 2008.

In this chapter we will examine the properties of the data set, particularly the variables that can be used for our model. We also deal with variables that have to be converted to numeric values and the selection of the most significant variables. Finally, we look briefly at the quality of the Freddie Mac data set.

3.1 Properties of the data set

In this thesis we use the *Single Family Loan-Level Dataset*, which is split into sets for each calender quarters, starting with the first quarter of 1999 and ending with the second quarter of 2015.[1] For each quarter there exists an origination file. This contains the initial available information for all mortgage loans originating in this quarter. There is also a monthly performance file available, containing the data on the performance of the loan, such as payments and interest rates, until the termination of the loan or the Performance Cutoff Month. The Performance Cutoff Month is the last month for which there is data available. As of August 2016, this is December 2015. The origination file consists of 25 entries for each mortgage and the monthly performance file consists of 22 entries for each month of the lifetime of a mortgage. In both files, the mortgages can be identified by their mortgage sequence number.

The data set consists of mortgages with a fixed interest rate (although the rates still can change due to a modification of the loan) and a duration of 15, 20 or 30 years.

[1] as of August 2016

3.2 Variables

We obtain our variables from three different files, i.e. the origination file, the monthly performance file (as explained in the previous section) and the file of macro-economic variables. The origination file contains variables obtained or measured at the origination date of the mortgage, which is only one date. Since the monthly performance contains observations for each month, we have to specify for each quarterly data set from which month we want to take the information. In the case that this defined month is after the month of termination, we use the information at the month of termination, since for a loan that has (or has not) gone into default, this event happened on the termination date.

We distinguish three types of variables, *numeric*, *boolean* and *nominal*. Numeric variables are variables that only take on numeric values, boolean variables only take on boolean values (± 1) and nominal variables can take on categorical values such as names or postal codes. Ordinal variables are in this thesis considered as a numeric variable, as they attain numeric values.

3.2.1 Dependent variable

The dependent variable (also called target variable) in our model is the default indicator y_i. We set $y_i = -1$ if client i has gone into default and $y_i = +1$ if this has not happened (yet). In the data set of Freddie Mac this indicator can be obtained from the *zero balance code* in the monthly performance file at the month of termination, which is a number indicating the reason for the termination of the loan. The following values can be observed in the data set.

- *01: Prepaid or matured* - This indicates that the current loan has been paid off by the client before the initial maturity date or the loan has been ended in a normal way. Note that the latter event has rarely occurred in this data set, as all mortgages have a duration of at least 15 years.
- *03: Foreclosure alternative group* - This indicates that the loan is terminated before the maturity date, due to the fact that the client is unable to make (full) payments, and sold to a party that is not the original seller of the mortgage.
- *06: Repurchase prior to Property Disposition* - This indicates that the loan is delinquent and is repurchased from Freddie Mac by the original seller. In this case Freddie Mac is compensated by the seller.
- *09: REO Disposition* - This indicates that the loan is delinquent and the underlying collateral is sold.
- *'Space': Not applicable* - This indicates that no termination event happens in this month. If a loan is still active at the Performance Cutoff Month, we see a space in the entry for the zero balance code.

The events denoted by *03* and *09* only occur for a delinquent status of the loan and are identified as defaults by Freddie Mac. Therefore we call these events 'defaults' and set $y_i = 1$ for these events. The events denoted by *01*, *06* or *'Space'* are denoted by $y_i = +1$, since no loss occurs for Freddie Mac in these cases.

Most of the times we look at a certain time horizon for the probability of default. For example, if we look at a 5-year PD with an observation period of 2 years, we mean that we first observe the loan for 2 years to gain some monthly performance information.

After these 2 years we do a prediction on the probability of default of a loan happening in 5 years. Using the monthly performance file we can find out if a default has occurred or not.

3.2.2 Independent variables

From the origination file and the monthly performance file we can extract several variables that can be used as a predictor for the default indicator. Furthermore we can use several macro-economic variables to include possible influences of the economic situation on the probability of default.

Not every column of the first two files can be used as variables, such as the *credit score*, which is a number indicating the creditworthiness of a borrowers at the initiation of the mortgage. As the creditworthiness (or the probability of default) is essentially the variable that we would like to predict, we cannot use this as an independent variable in our model. Clearly, information obtained about a termination event, such as actual losses due to a default, cannot be used either.

Table 3.1 contains a list of all variables that can be used in our model. Variables from the origination files are starting with an O, variables from the monthly performance files with a P and macro-economic variables with an M.

Code	**Variable name**	**Variable definition**	**Type**
O_1	First payment date	Number of months between end of the quarter and first scheduled payment	Numeric
O_2	First time homebuyer flag	Indicates whether the borrower is purchasing the property, will reside in the property as a primary residence and was not an owner of a property over the last three years	Boolean
O_3	Maturity date	Number of months between end of the quarter and last scheduled payment	Numeric
O_4	Metropolitan Statistical Area (MSA) or Metropolitan Division	MSA or MD code of the area in which the property is located	Nominal
O_5	Mortgage insurance percentage	Percentage of the loan, that a mortgage insurer is providing to pay back if there are any losses due to a default of the client (the percentage is set at the time of the purchase by Freddie Mac)	Numeric
O_6	Number of units	Indicates how many units a property contains	Numeric
O_7	Occupancy status	Indicates whether the property is occupied by the owner, is acquired as a second-home or as an investment	Nominal
O_8	Original combined loan-to-value (CLTV)	See Section 2.1.2	Numeric
O_9	Original debt-to-income (DTI)	See Section 2.1.2	Numeric
O_{10}	Original unpaid principal balance (UPB)	See Section 2.1.2	Numeric
O_{11}	Original loan-to-value (LTV)[2]	See Section 2.1.2	Numeric

[2]Due to a high collinearity with variable O_8 we defined O_{11} after Section 3.4.1 as CLTV - LTV.

Code	**Variable name**	**Variable definition**	**Type**
O_{12}	Original interest rate	The initial (fixed) interest rate on the mortgage	Numeric
O_{13}	Channel	Indicates whether a *broker*, *correspondent* or *not specified third party* was involved in the origination of the mortgage loan	Nominal
O_{14}	Prepayment penalty mortgage (PPM) flag	Indicates whether there is a penalty on prepaying part of the mortgage loan	Boolean
O_{15}	Property state	U.S. State where the property is located	Nominal
O_{16}	Property type	Indicates the type of property (e.g. a manufactured home or a condominium)	Nominal
O_{17}	Postal code	Postal code of the mortgage property	Nominal
O_{18}	Loan purpose	Indicates of a mortgage is used to buy the underlying property or for other specified reasons	Nominal
O_{19}	Number of borrowers	Indicates whether the amount is borrowed by one person or more than one	Boolean
O_{20}	Seller name	Name of the company selling the mortgage to Freddie Mac	Nominal
O_{21}	Servicer name	Name of the company to which a borrower pays his mortgage loan payments as of the last observed period	Nominal
P_1	Ratio actual UPB and original UPB	Ratio of actual unpaid principal balance and original unpaid principal balance	Numeric
P_2	Maximum loan delinquency	Maximum number of months a borrower has been delinquent up to and including the observed month	Numeric
P_3	Loan age	Number of months since the origination month of the mortgage	Numeric
P_4	Current interest rate	Current interest rate on mortgage, taking into account any loan modifications	Numeric
M_1	Real GDP	Percent change of the U.S. Real Gross Domestic Product from preceding period, seasonally adjusted annual rate	Numeric
M_2	Consumer Price Index (CPI)	A measure of the price of a basket of common goods for urban consumers in the U.S.	Numeric
M_3	10-year treasury constant maturity rate	The yield of the most recently auctioned 10-year U.S. Treasury notes	Numeric
M_4	Industrial Production Index (IPI)	Measures the amount of output from the manufacturing, mining, electric and gas industries in the U.S.	Numeric
M_5	Civilian unemployment rate	A measure of the number of people that are jobless and looking for a job in the U.S.	Numeric
M_6	Dollar $\setminus$ Euro exchange rate	Exchange rate between U.S. Dollar and Euro	Numeric

TABLE 3.1: Variable definitions and types

Macro-economic variables

In our model we use 6 indicators of the macro-economic situation. The choice of these indicators is quite arbitrary, but these indicators give a good view on different parts of the economy (Smith, 2011). They are all so called *lagging indicators*, variables that do not tell us where the economy is heading, but how the economy is performing at the current moment. We explicitly exclude any indicators of the housing market, as we know that there is already a high correlation between declining house prices and increasing default probabilities. It is interesting if we can do the prediction also without this indicator.

3.3 Conversion of variables

In principle, variables that are of the numeric type can be directly used as input for the model, since a support vector machine works with continuous or discrete ordered input. Nonetheless, all numeric variables are scaled such that the maximum for each variable equals $+1$ and the minimum equals -1. This is useful, since a variable that takes very high (or low) values, tend to dominate the gradient in the stochastic gradient descent. If all variables approximately take values in the same range, this effect is not observed.

Including boolean variables in our data set is therefore also not a huge difficulty, since we can set one value to equal $+1$ and one value equal to -1. Finally, to also include variable with a nominal (categorical) type, we use a ranking method. Using a not too large part of the training set, we calculate for each value (category) of this variable, the default percentage of the set of mortgages having this value. As numeric value for this category we take this default percentage. All categories of this variable that are not attained in the part of the training set, but occur in the other part of the training set or test set, will obtain a standard numeric value, such as zero or an average default probability. If certain categories continue to have a significantly higher or lower PD on the training set, it will be useful to include this variable in our model. After this conversion, the values for the nominal variable are again scaled to values with maximum $+1$ and minimum -1.

3.4 Selection of variables

As there are 31 variables available, it is useful to make a selection of the most significant variables to include in the model.

3.4.1 Multicollinearity

At first we need to observe if there are any high correlations between independent variables present in the data, to avoid *multicollinearity* between these variables. If two individual variables (predictors) are highly correlated, such that one depends linearly on the other, large standard errors in the estimated coefficients can appear in the case of a regression. Also, small changes in the data can lead to large changes in the model. This problem is often encountered in regression analysis and also poses a threat for our model, since it can lead to large (absolute) and dominating coefficients for the correlated variables in the support vector machine. More information on the effect of high correlation on predictors can be found in Mela and Kopalle, 2002.

In Table 3.2 below we see the correlation coefficients between the different macro-economic variables on the set of monthly observations between January 1999 and December 2015, which is the time period in scope of this thesis. We obtain a high negative correlation between the CPI and the 10-year treasury interest rate. As all correlations are in absolute value below the thresholds of 80% or 90% that are used a lot, we can include all macro-economic variables in our SVM.

	Real GPD	CPI	10-year interest	IPI	Unemployment	Dollar \ Euro
Real GDP	100.0%	-20.3%	23.9%	1.8%	-18.7%	-16.8%
CPI	-20.3%	100.0%	-78.3%	58.0%	56.0%	67.3%
10-year interest	23.9%	-78.3%	100.0%	-37.3%	-63.3%	-51.3%
IPI	1.8%	58.0%	-37.3%	100.0%	-26.6%	37.1%
Unemployment	-18.7%	56.0%	-63.3%	-26.6%	100.0%	52.1%
Dollar \ Euro	-16.8%	67.3%	-51.3%	37.1%	52.1%	100.0%

TABLE 3.2: Correlation between macro-economic variables

In Table 3.3 below we see the correlation coefficients for the different variables of the numeric type in the origination file of the first quarter of 1999. We see approximately the same correlation coefficients for other origination files. The only variables that are definitely highly correlated are the original CLTV (O_8) and the original LTV (O_{11}). Therefore we can substitute O_{11} by the difference between the original CLTV and the original LTV, to still capture some effect of the differences between these variables. As we can see in the table, this newly defined variable has only a little correlation with all other variables, as it has probably many values close to zero.

	First payment date	Maturity date	MI percentage	Number of units	CLTV	DTI	UPB	LTV	Interest rate	CLTV - LTV
First payment date	100.0%	11.7%	-2.1%	-0.3%	-1.6%	1.2%	2.2%	-1.6%	-0.2%	0.0%
Maturity date	11.7%	100.0%	-0.2%	0.4%	0.2%	3.0%	1.1%	0.1%	-1.3%	0.0%
MI percentage	-2.1%	-0.2%	100.0%	-4.2%	69.5%	8.7%	0.7%	69.6%	12.5%	-0.1%
Number of units	-0.3%	0.4%	-4.2%	100.0%	-4.3%	1.3%	4.5%	-4.3%	8.7%	0.0%
CLTV	-1.6%	0.2%	69.5%	-4.3%	100.0%	11.3%	10.7%	99.9%	7.7%	0.1%
DTI	1.2%	3.0%	8.7%	1.3%	11.3%	100.0%	9.4%	11.4%	3.5%	0.0%
UPB	2.2%	1.1%	0.7%	4.5%	10.7%	9.4%	100.0%	10.5%	-13.7%	0.2%
LTV	-1.6%	0.1%	69.6%	-4.3%	99.9%	11.4%	10.5%	100.0%	7.8%	-0.1%
Interest rate	-0.2%	-1.3%	12.5%	8.7%	7.7%	3.5%	-13.7%	7.8%	100.0%	-0.1%
CLTV - LTV	0.0%	0.0%	-0.1%	0.0%	0.1%	0.0%	0.2%	-0.1%	-0.1%	100.0%

TABLE 3.3: Correlation of numeric variables in the origination file of 1999Q1

For all other variables, the values for the correlation is complete depending on the choice of the set determining the numeric values for the nominal variables or on the observation period that is set, so we cannot do such a general analysis.

3.4.2 Test of mean differences

However, since we have not excluded any variables from our set in the previous section, we need another selection criterion. This selection is done again during every development of the model, as the significance of a variable may change over time. We perform a so called *test of mean differences* between the two labels. This means that we divide our data set into two different sets, one with all defaulted loans and one with all non-defaulted loans. We use a part of the training set to calculate the mean and variance of the different variables for these two sets, and choose the variables for which the means are significantly different, i.e. the variables that seem to have the largest predictive value for our model. Preferably the training set is much larger than or not including the set used for classifying the nominal variables, to avoid the influence of the already present ranking of the categorical values.

A test of mean difference on a variable is performed as follows (Whitley and Ball, 2002). Let μ_{-1} be the mean of this variable on the set of defaulted loans and μ_{+1} on the set of non-defaulted loans. The null hypothesis is that the two means are not statistically different from each other, i.e.

$$H_0 : \mu_{-1} = \mu_{+1} \tag{3.1}$$

with alternative hypothesis

$$H_a : \mu_{-1} \neq \mu_{+1} \tag{3.2}$$

For a set with observations that are i.i.d. with mean μ and standard deviation s, sample mean $\overline{x}$ and sample size m, we know from the central limit theorem that z defined as below follows a standard normal distribution.

$$z = \frac{\overline{x} - \mu}{s/\sqrt{m}} \tag{3.3}$$

We can convert this to our case with two samples. Denote the size of the sets by m_{-1} and m_{+1} and the sample means by $\overline{x_{-1}}$ and $\overline{x_{+1}}$. If the standard deviation of both sets is known (and denoted by σ_{-1} and σ_{+1}, we know that z has a standard normal distribution, where z is defined as follows.

$$z = \frac{(\overline{x_{-1}} - \overline{x_{+1}}) - (\mu_{-1} - \mu_{+1})}{\sqrt{\frac{\sigma_{-1}^2}{m_{-1}} + \frac{\sigma_{+1}^2}{m_{+1}}}} \tag{3.4}$$

The value of the z-statistic is used to compute the two-sided p-value, i.e.

$$\begin{aligned} p &= P(|z'| > |z|) = P(z' > |z|) + P(z' < -|z|) \\ &= 1 - \Phi(|z|) + \Phi(-|z|) = 2\Phi(-|z|) \end{aligned} \tag{3.5}$$

where z' assumes a standard normal distribution with CDF Φ. The two-sided p-value indicates the probability that the absolute value of z' exceeds the absolute value of z under the standard normal distribution. It is essentially the probability that, given that the two means μ_{-1} and μ_{+1} are equal, the absolute difference between the sample means $\overline{x_{-1}}$ and $\overline{x_{+1}}$ is larger than the difference we have observed. If this probability is smaller than some specified significance level α (usually 1, 5 or 10 %), the null hypothesis that the two means are equal, is rejected.

However, we are dealing with the case that the standard deviations of the sets are unknown, just as the means. Therefore we have to switch to the so called t-statistic, which is defined as follows

$$t = \frac{(\overline{x_{-1}} - \overline{x_{+1}}) - (\mu_{-1} - \mu_{+1})}{\sqrt{\frac{s_{-1}^2}{m_{-1}} + \frac{s_{+1}^2}{m_{+1}}}} \tag{3.6}$$

where s_{-1} and s_{+1} denote the population standard deviations of the sets. This is called the *two-sample t-statistic*. For the *one-sample t-statistic*, defined as

$$t = \frac{\overline{x} - \mu}{s/\sqrt{m}} \tag{3.7}$$

we know that t by definition follows a Student's t-distribution with $m-1$ degrees of freedom.

The two-sided t-statistic does not exactly follow a Student's t-distribution, as there are two (and not one single) estimated standard deviations in the statistic. However, since for $x \geq 0$ we see in Figure 3.1 that for a smaller number for the degrees of freedom, the cumulative t-distribution attains smaller values, we obtain a conservative p-value when setting the degrees of freedom ν equal to the minimum of $m_{-1} - 1$ and $m_{+1} - 1$.

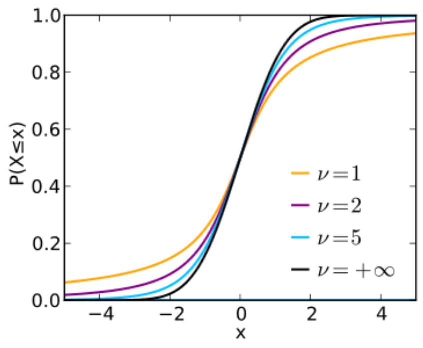

FIGURE 3.1: CDFs of Student's t-distributions with different degrees of freedom ν

The value of the two-sample t-statistic is used to compute the two-sided p-value, i.e.

$$p = P(|t'| > |t|) = P(t' > |t|) + P(t' < -|t|) = 2P(t' < -|t|) \tag{3.8}$$

where t' follows a Student's t-distribution with $\min\{m_{-1} - 1, m_{+1} - 1\}$ degrees of freedom. Again, we reject the null hypothesis that the two means μ_{-1} and μ_{+1} are equal if the p-value is smaller than some significance level.

We would like to only select the most significant variables for our model. Clearly, that would be the variables for which the mean of the set of the defaulted loans differs statistically most from the mean of the set of the non-defaulted loans, i.e. for which the two-sided p-value is the smallest. We can now select the variables in two different ways, either by only taking the variables that have a p-value below a certain significance level, or by the taking the n variables with the smallest p-values. In the first case we have an initially unknown number of variables, which may lead to unpleasant results, for example when no variables are selected. In the latter case the number of variables is fixed. Due to the high number of observations of our set, the values for m_{-1} and m_{+1} are very high, such that the t-statistic will also tend to go further away from zero, thus resulting in a p-value very close to zero. Therefore, in our case we select the variables that have the largest absolute t-statistics. Again we stress the importance that the variables are selected using a different set than the set scoring the nominal variables, as the set of defaulted loans will clearly have a higher mean for a certain nominal variable on the scoring set, so this variable is more easily selected.

Below we find in Table 3.4 the statistics of the different variables for the 1999Q1 data set, considering a 5-year default horizon after a 2-year observation period. For both the defaulted (negative) and the non-defaulted (positive) class we display the mean (upper cell) and standard deviation (lower cell) of each variable and for the test of mean differences the value of the t-statistic and the p-value. Observe that only variables from the origination and the monthly performance file are included, since the values of the macro-economic variables are the same for an entire (quarterly) data set.
If we, for example, want to have the 10 most significant variables for 1999Q1, we select the variables in the following order (with t-statistic): $O_8(-48.076)$, $P_2(-30.584)$, $O_{10}(28.624)$, $O_5(-26.204)$, $P_4(-22.761)$, $O_{11}(21.407)$, $O_{12}(-21.400)$, $O_{19}(-18.646)$, $O_9(-16.229)$ and $O_{20}(-14.936)$. Observe that for any data set this selection can be different, as some variables could have more significance in other periods.

	+ class	**- class**	**t-statistic**	**p-value**
O_1	1.1210	1.1147	0.293	0.770
	1.8162	1.0833		
O_2	-0.5270	-0.5125	-1.125	0.260
	0.6569	0.6775		
O_3	359.957	360.018	-1.860	0.063
	2.1509	1.6762		
O_4	0.0075	0.0083	-10.529	0.000
	0.0030	0.0039		
O_5	6.7820	13.6113	-26.204	0.000
	11.5917	13.3324		
O_6	0.0075	0.0075	0.006	0.995
	0.0002	0.0002		
O_7	0.0075	0.0074	10.683	0.000
	0.0002	0.0002		
O_8	75.1207	83.9616	-48.076	0.000
	15.0002	9.3440		
O_9	30.0802	33.7682	-16.229	0.000
	12.6683	11.6069		
O_{10}	126885	101035	28.624	0.000
	54459	46089		
O_{11}	0.0449	0.0057	21.407	0.000
	0.3632	0.2440		
O_{12}	6.9125	7.0914	-21.400	0.000
	0.3252	0.4281		
O_{13}	0.0075	0.0076	-7.821	0.000
	0.0006	0.0006		
O_{14}	-0.9551	-0.9567	0.312	0.754
	0.2570	0.2348		
O_{15}	0.0070	0.0075	-11.330	0.000
	0.0022	0.0025		
O_{16}	0.0075	0.0075	-8.332	0.000
	0.0003	0.0002		
O_{17}	0.0075	0.0088	-5.805	0.000
	0.0048	0.0118		
O_{18}	0.0075	0.0074	2.865	0.004
	0.0002	0.0002		
O_{19}	0.0075	0.0076	-18.646	0.000
	0.0004	0.0004		
O_{20}	0.0075	0.0081	-14.936	0.000
	0.0022	0.0022		
O_{21}	0.0075	0.0079	-12.213	0.000
	0.0016	0.0017		
P_1	1.1179	0.9750	12.049	0.000
	0.7234	0.6046		
P_2	0.1124	1.9749	-30.584	0.000
	0.4448	3.1240		
P_3	24.5947	24.5969	-0.057	0.954
	3.0894	1.9224		
P_4	6.8126	7.0746	-22.761	0.000
	0.8838	0.5855		

TABLE 3.4: Results of mean differences test for 1999Q1

3.5 Data quality

Freddie Mac indicates in their FAQ (FHMLC, 2016) that there are some data anomalies present, due to the fact that there are many data files combined into one data set, which may result in mismatches. Three other reasons are given why there may be errors in the data.

- *Reporting errors by the seller or servicer* - The data set relies on the information given by the seller and servicer of a mortgage.
- *Evolving of data quality controls* - The last 16 years the control on data quality may have been improved, such that more errors are detected and removed.
- *Loan delivery requirements* - The requirements set by Freddie Mac to a seller of a mortgage have changed over time, which can result in some missing values.

Freddie Mac already applied some analysis on invalid values for five different variables in the origination file, i.e. if the values are below or above some reasonable thresholds. The examples with the invalid values are not filtered out, but the values are set to three spaces. The values are given in the next table. Also, the percentage of 'invalid' observations in the first and last available data set are included in Table 3.5 below.

Variable	**Valid values**	**Invalid in 1999Q1**	**Invalid in 2015Q2**
Credit score	$301 \leq CS \leq 850$	0.4%	0.0%
MI percentage (O_5)	$1\% \leq MI \leq 55\%$ or 0	25.8%	0.0%
Original CLTV (O_8)	$0\% \leq CLTV \leq 200\%$	0.0%	0.0%
Original DTI (O_9)	$0\% < DTI \leq 65\%$	0.6%	0.0%
Original LTV ($O_8 + O_{11}$)	$6\% \leq LTV \leq 105\%$	0.0%	0.0%

TABLE 3.5: Valid values for five variables in the origination file as determined by Freddie Mac

Since the observations with these values are identified by Freddie Mac using three spaces in the entry of the corresponding variable, we could easily exclude these observations from our transformed data set. However, the other variables of such an observation do not need to be invalid and therefore a filter is not preferred. A better solution is to set the value of an invalid entry to a value that does not influence the coefficient for this variable in the support vector machine. Because of the fact that before the model is developed a rescaling of all variables takes place, we can set these invalid values to the average of the minimum and the maximum. Since the value is essentially (after modification) set to zero, it does not influence the value of the objective function given by Equation 6.29, hence it does not influence the gradient used for the update of the target variable.

Besides the mortgages that are only in the origination file and not in the monthly performance file (which should clearly be excluded, since we have no default information), there are mortgages for which the monthly performance file misses some reporting months between the start and termination date. In the case that we need an observation in a certain missing month, we take the last month before the missing month that is available in the data set. Since the monthly performance set contains some observations after the missing month, we certainly know that the loan has not yet gone into default. Therefore we can at least be certain about the value of the default indicator y_i.

Chapter 4

Support vector machines

During the first decades of this century machine learning became one of the most important branches of information technology and it influences an ever increasing part of our life. Self-driving cars, recommendations on Google and spam filters are all products of machine learning. As the size of data sets increase more rapidly than the computation speed, it is a challenge to find algorithms that can still handle the available data efficiently.

The principle idea of machine learning is to use training data to let the computer develop a model that can make predictions about other data. Probably the best known examples of this are *linear* and *logistic regression* (see Section 2.4). In this thesis we will look at *classification problems*, where the goal is to classify our data points correctly, i.e. to put them in the 'right' categories.

One of the most frequently used machine learning techniques is the so called *support vector machine (SVM)*. It belongs to the classification problems and is a supervised learning method. The goal of a support vector machine is to build a model, based on training examples, that assigns a category to each new example. In its simplest form, it separates two clusters of points by a gap, that is as large as possible.

In this chapter we will first introduce machine learning and then focus on the traditional set-up of an SVM, the so called hard-margin linear SVM. The soft-margin linear SVM and nonlinear SVM are introduced afterwards as generalizations of the hard-margin linear SVM. Then we switch to the primal and dual formulation of the nonlinear SVM, as the dual problem is often far more easy to solve. This chapter ends with an brief overview of some extensions, such as an unsupervised SVM and multiclass SVM.

4.1 Definition of machine learning

We cannot start our discussion of machine learning without two famous, widely quoted definitions of machine learning. The first one is by Arthur Samuel (Simon, 2013).

"Machine learning is the field of study that gives computers the ability to learn without being explicitly programmed."

The second one is a more formal definition by Tom Mitchell (Mitchell, 1997)

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E"

Machine learning can basically be divided into three categories: *supervised learning*, *unsupervised learning* and *reinforcement learning*. For supervised learning, the output of all training examples is given, whereas for unsupervised learning, the output is unknown for all training examples. Finally, in the case of reinforcement learning, a computer has to perform a certain task and reach a certain goal without being told whether it has actually come close to the goal.

Another categorization of machine learning is based on the goal of the programs. At first we have a classification problem. For this problem the output consists of labels (± 1, or 1 up to n) and the goal is to predict the label of a data point based on its input. This can be both supervised and unsupervised. In a regression, the goal is to predict a continuous output, rather than a discrete (labeled) output. This is a supervised problem. In clustering, the goal is to split a given set into k different clusters, which is unsupervised most of the times.

Famous machine learning approaches are decision tree learning, which uses decision trees as a predictive model, (artificial) neural networks, which are algorithms that are inspired by the neural networks observed in bodies, used to model complex relations between inputs and outputs, and support vector machines. The support vector machine will be the main approach to our problem of rating the creditworthiness of mortgage loans.

4.2 Set-up

We are given a data set of training examples (x_i, y_i) for $i = 1, \dots, m$, with $x_i \in \mathbb{R}^n$ and we assume for the moment that y_i can only attain two values. We denote these values by ± 1. The jth coefficient of x_i is denoted as $x_{i,j}$. The main idea of a support vector machine is to separate the data points x_i with $y_i = -1$ from the data points with $y_i = 1$.

4.2.1 Hard-margin linear SVM

We first look at linearly separable data, where we have the so called *hard-margin linear support vector machine*. The goal is to find a vector $w \in \mathbb{R}^n$ and a scalar $b \in \mathbb{R}$ such that

$$w^T x_i + b \geq 1 \qquad \text{for all } i \text{ with } y_i = 1 \tag{4.1}$$

$$w^T x_i + b \leq -1 \qquad \text{for all } i \text{ with } y_i = -1 \tag{4.2}$$

We can rewrite the above equations to one single constraint

$$y_i(w^T x_i + b) \geq 1 \quad \text{for } 1 \leq i \leq m \tag{4.3}$$

If we can find a vector w such that the above equation hold, the $(n-1)$-dimensional hyperplane given by

$$\{x \in \mathbb{R}^n : w^T x + b = 0\} \tag{4.4}$$

separates the data points. Actually we can find infinitely many separating hyperplanes, i.e. there are infinitely many of such vectors w, if the data is *linearly separable*. We call two sets linearly separable, if there exists $n+1$ real numbers $w_1, ..., w_n, b$ such that $\sum_{j=1}^n w_j x_{i,j} < b$ for all data points x_i in the first set and $\sum_{j=1}^n w_j x_{i,j} > b$ for all data points x_i in the second set. The question is now which w is the 'best' to use. Clearly, looking at Figure 4.1, we would prefer the red line H_3 to the blue line H_2, although both lines are feasible for separating the data.

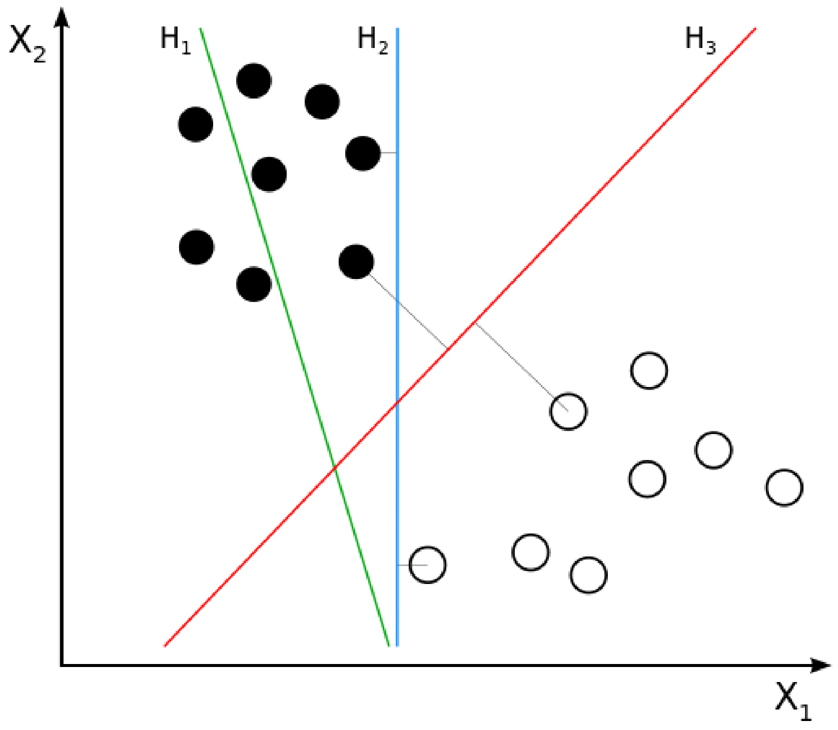

FIGURE 4.1: Linear SVM with multiple separating hyperplanes.

We can accomplish this by looking at the *support vectors* of a hyperplane. These are the points x_i such that Equation 4.3 holds with equality, i.e. the points closest to the hyperplane in normal direction. We want the Euclidean distance between the two hyperplanes $w^T x + b = 1$ and $w^T x + b = -1$ going through the support vectors, to be as large as possible. From linear algebra we recall that the Euclidean distance between two parallel hyperplanes equals $\frac{|c|}{\|w\|}$, where c is the difference between the two numbers after the equal signs, which equals 2 in this case. Maximizing this quantity is equivalent to minimizing the norm $\|w\|$ or the squared norm $\|w\|^2$, which is a convex function. We can now formulate the optimization problem for hard-margin linear SVM.

$$\begin{aligned} \min_{w\in\mathbb{R}^n, b\in\mathbb{R}} \quad & \|w\|^2 \\ \text{s.t.} \quad & y_i(w^T x_i + b) \geq 1 \quad \text{for } 1 \leq i \leq m \end{aligned} \tag{4.5}$$

4.2.2 Soft-margin linear SVM

Before we turn to algorithms for solving Problem 4.5 we take a look at the *soft-margin linear support vector machine*. This is a generalization to the hard-margin linear SVM, where the data may not necessarily be linearly separable. In this case we need to introduce a cost function, which penalizes the data points that are misclassified by the hyperplane. There are many possibilities, but the most common used is the so called *hinge loss function*, which is a convex function given by

$$f_i(w, b) = \max\{0, 1 - y_i(w^T x_i + b)\} \tag{4.6}$$

This function equals 0 if and only if Equation 4.3 holds. In this case x_i is correctly classified and 'far' enough from the separating hyperplane. For x_i close to or on the wrong side of the hyperplane, the function is positive and proportional to the Euclidean distance from the hyperplane going through the support vector(s) with the same classification as the data point (x_i, y_i). We will add the value of the hinge loss function averaged over all m training examples to our optimization objective, since we want to minimize our loss from misclassifying the training examples. We allow some of the training examples to be misclassified, so the inequality constraints do not need to hold anymore. The optimization problem changes to the following

$$\min_{w \in \mathbb{R}^n, b \in \mathbb{R}} \quad \frac{\lambda}{2}\|w\|^2 + \frac{1}{m}\sum_{i=1}^{m} \max\{0, 1 - y_i(w^T x_i + b)\} \tag{4.7}$$

The parameter λ controls the trade-off between maximizing the margin, which is achieved by minimizing $\|w\|^2$, and classifying as many data points x_i as possible correctly, which is achieved by minimizing the averaged hinge loss function. For small values of λ and linearly separable data we see that soft-margin SVM works almost identically to hard-margin SVM, and we will still obtain a separating hyperplane between the two classes.

4.2.3 Nonlinear SVM

Fortunately the use of support vector machines is not limited to the linear case. This is can be achieved by introducing a function $\phi : \mathbb{R}^n \to \mathbb{R}^d$, where usually $d > n$, which maps x_i to the vector $\phi(x_i)$. One simple example of ϕ is the mapping that maps x to all *monomials* up to a certain degree k. For example, if $n = k = 2$ this mapping is

$$\phi(x_i) = \phi\left(\begin{pmatrix} x_{i,1} \\ x_{i,2} \end{pmatrix}\right) = \begin{pmatrix} x_{i,1}^2 \\ x_{i,2}^2 \\ x_{i,1}x_{i,2} \\ x_{i,1} \\ x_{i,2} \end{pmatrix} \tag{4.8}$$

Sometimes the constant function 1 is also listed as monomial. In our case we do not include this expression, since we already have a constant intercept b that we can vary. We call ϕ the *monomial kernel mapping*. The following lemma shows that the dimension d of $\phi(x)$ rapidly increases with both n and k.

Lemma 4.1. *The number of monomials of n variables of at most degree k equals $\binom{k+n}{k}$.*

Proof. We prove this statement using the combinatorial method of *stars and bars*. We will first show that there are $\binom{m+n-1}{m}$ monomials of degree m. Consider the m powers as m stars in a line. Every power (star) can be assigned to only one variable. Then we can insert $n-1$ bars between these stars, to indicate which powers correspond to which variable. Each bar marks a cut-off, i.e. the first variable goes to the power of the number of stars before the first bar, etcetera. Obviously multiple bars can be subsequent, as a variable can be absent in a monomial. Essentially there are $n + m - 1$ spots, where we have to place m stars and $n - 1$ bars. The number of ways that this can be done equals $\binom{m+n-1}{m}$, since after putting m stars into place, the other spaces are filled up with the $n - 1$ bars.

For $k = 1$ the lemma is clearly satisfied, since the number of monomials $d = \binom{n}{1} = n$ of degree 1 equals the number of monomials up to degree 1. Now assume that the lemma is satisfied for degree $k-1$. Then the number of monomials up to degree k equals the sum of number of monomials up to degree $k-1$ and the number of monomials of degree k.

$$\begin{aligned} d &= \binom{k-1+n}{k-1} + \binom{k+n-1}{k} = \frac{(k+n-1)!}{(k-1)!\,n!} + \frac{(k+n-1)!}{k!\,(n-1)!} \\ &= \frac{k(k+n-1)! + n(k+n-1)!}{k!\,n!} = \frac{(k+n)!}{k!\,n!} = \binom{k+n}{k} \end{aligned} \tag{4.9}$$

This closes the proof. □

For example, if we have 5 variables, the number of monomials of degree up to 5 equals 252, which is already tremendously increases the computation time that is needed.

Obviously we can think of many more difficult expressions ϕ, but the idea stays the same: ϕ effectively maps the data points x_i to a (higher) dimension, on which we hope to be able to apply the soft- or hard-margin linear SVM, to find some hyperplane that is separating (most of) the data linearly, as we see in Figure 4.2.

Example 4.1. *Consider a set of data points $(x_i, y_i) = (x_{i,1}, x_{i,2}, y_i) \in \mathbb{R}^3$ where $x_{i,1}$ and $x_{i,2}$ are randomly generated with mean 0, for example from a normal distribution, and $y_i = 1$ if and only if $(x_{i,1})^2 + (x_{i,2})^2 < 1$ (see the left part of Figure 4.2 below, where the red points have $y_i = 1$). Generally, when we have enough data points, this data is not linearly separable, but only with a quadratic polynomial.*

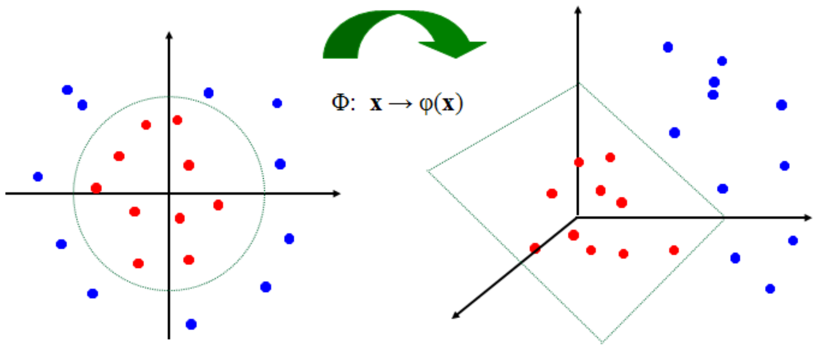

FIGURE 4.2: Transformation of the data set by ϕ

If we define the feature mapping ϕ by

$$\phi(x_i) = (x_{i,1}, x_{i,2}, (x_{i,1})^2 + (x_{i,2})^2 - 1)^T \tag{4.10}$$

we map the data points to a higher dimension ($\mathbb{R}^3$ or $\mathbb{R}^4$ if one includes the labels y_i). In this higher dimension, the data is clearly separable, since a point has a red label ($y_i = 1$) if and only if the third coordinate of $\phi(x_i)$ is smaller than 0. Therefore the plane $y_3 = 0$ separates the data points.

It is now easy to adapt our optimization problem 4.7

$$\min_{w\in\mathbb{R}^d, b\in\mathbb{R}} \quad \frac{\lambda}{2}\|w\|^2 + \frac{1}{m}\sum_{i=1}^{m} \max\{0, 1 - y_i(w^T\phi(x_i) + b)\} \tag{4.11}$$

4.3 Primal and dual problem

Our goal is to find the solution to Problem 4.11. We focus only on this general nonlinear case, since both Problem 4.7 (soft-margin linear SVM) and Problem 4.5 (soft-margin linear SVM) can be solved by setting $\phi(x) = x$ (for both cases) and $\lambda \approx 0$ (for the latter case).

4.3.1 Primal problem

We can return to an optimization problem with constraints by introducing a new scalar variable ζ_i for $1 \leq i \leq m$, and observe that Problem 4.11 is equivalent to the following convex *primal problem*

$$\begin{aligned} \min_{w \in \mathbb{R}^d, \zeta_i \in \mathbb{R}, b \in \mathbb{R}} \quad & \frac{\lambda}{2}\|w\|^2 + \frac{1}{m}\sum_{i=1}^m \zeta_i & \\ \text{s.t.} \quad & y_i(w^T\phi(x_i) + b) \geq 1 - \zeta_i & \text{for } 1 \leq i \leq m \\ & \zeta_i \geq 0 & \text{for } 1 \leq i \leq m \end{aligned} \tag{4.12}$$

Observe that both the objective function and the constraint functions are convex and continuously differentiable. The set that we are minimizing our objective function on is also clearly convex.

4.3.2 Dual problem

It is common to look at the corresponding dual problem.[1] The *Lagrangian* for our optimization problem is given by

$$\mathcal{L}(w, b, \zeta, \alpha, \beta) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^m \zeta_i - \sum_{i=1}^m \beta_i\zeta_i - \sum_{i=1}^m \alpha_i\left(y_i(w^T\phi(x_i) + b) - 1 + \zeta_i\right) \tag{4.13}$$

where $\zeta, \alpha, \beta \in \mathbb{R}^m$ are the vectors containing all $\zeta_i, \alpha_i, \beta_i$ respectively and we defined $C = \frac{1}{\lambda m}$ to simplify notation for later. The minus sign in front of the last terms are due to the fact that both inequality constraints in the primal problem are of the type $g(z) \geq 0$. To find the dual problem of this convex optimization problem, we need to minimize the Lagrangian $\mathcal{L}(w, b, \zeta, \alpha, \beta)$ with respect to our minimization variables of the primal problem w, b and ζ, while fixing α and β. We obtain

$$0 = \nabla_w \mathcal{L}(w, b, \zeta, \alpha, \beta) = w - \sum_{i=1}^m \alpha_i y_i \phi(x_i) \tag{4.14}$$

$$0 = \frac{\partial}{\partial b}\mathcal{L}(w, b, \zeta, \alpha, \beta) = -\sum_{i=1}^m \alpha_i y_i \tag{4.15}$$

$$0 = \frac{\partial}{\partial \zeta_i}\mathcal{L}(w, b, \zeta, \alpha, \beta) = C - \beta_i - \alpha_i \tag{4.16}$$

[1] For a detailed explanation of duality and convex optimization we refer to Appendix A.

From these equations we obtain the following useful expressions

$$w = \sum_{i=1}^{m} \alpha_i y_i \phi(x_i) \tag{4.17}$$

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \tag{4.18}$$

$$\alpha_i + \beta_i = C \quad \text{for all } 1 \leq i \leq m \tag{4.19}$$

When we insert Equation 4.19 into the Lagrangian of Equation **??** we see that the second term cancels out against the other terms containing ζ_i. Also, due to Equation 4.18, we observe that the last term simplifies and we obtain

$$\mathcal{L}(w, b, \zeta, \alpha, \beta) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{m} \alpha_i \left(y_i(w^T\phi(x_i)) - 1\right) \tag{4.20}$$

Now we substitute Equation 4.17 for w in the term of the Lagrangian containing $\phi(x_i)$

$$\sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j \phi(x_i)^T \phi(x_j) \tag{4.21}$$

We now define the *kernel* $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, which in fact is an inner product

$$K(x_i, x_j) = \phi(x_i)^T \phi(x_j) \tag{4.22}$$

Typically $\phi(x_i)$ has a very high dimension, and it can even be infinite. Since the kernel is essentially only a number for each pair of (i, j), it is often much more convenient to only calculate the kernel values. Combining Equations 4.21 and 4.22 we obtain

$$\sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \tag{4.23}$$

We can also substitute the expression for w in $\|w\|^2$

$$\begin{aligned} \|w\|^2 &= \left(\sum_{j=1}^{m} \alpha_j y_j \phi(x_j)\right)^T \left(\sum_{i=1}^{m} \alpha_i y_i \phi(x_i)\right) \\ &= \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j \phi(x_i)^T \phi(x_j) \\ &= \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \end{aligned} \tag{4.24}$$

We can simplify the notation still by introducing a $m \times m$-matrix Q with $Q_{ij} = y_i y_j K(x_i, x_j)$. Then we can rewrite Equation 4.24 as follows

$$\|w\|^2 = \sum_{i,j=1}^{m} \alpha_i Q_{ij} \alpha_j = \alpha^T Q \alpha \tag{4.25}$$

Therefore the Lagrangian takes the following form

$$\mathcal{L}(w, b, \zeta, \alpha, \beta) = \frac{1}{2}\alpha^T Q\alpha - \sum_{i=1}^{m} \alpha_i \tag{4.26}$$

We can now formulate the *dual problem*. We take the Lagrangian from Equation 4.26 and minimize it with respect to α. Minimizing with respect to β makes no sense, since it does not appear anymore in this expression. As usual when dealing with the dual problem, we have the constraints $\alpha_i, \beta_i \geq 0$. Combining with Equation 4.19 we can express these constraints as $0 \leq \alpha_i \leq C$ for this particular case. Together with the constraint of Equation 4.18 we arrive at the following optimization problem

$$\begin{aligned} \max_{\alpha} \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\alpha^T Q\alpha \\ \text{s.t.} \quad 0 \leq \alpha_i \leq C \quad \text{for all } 1 \leq i \leq m \\ \sum_{i=1}^{m} \alpha_i y_i = 0 \end{aligned} \tag{4.27}$$

4.3.3 Kernels

We can use different kernels for our dual problem of the support vector machine. If we for example consider the mapping ϕ of Equation 4.8, we see that the kernel $K(x_i, x_j)$ is given by

$$\begin{aligned} K(x_i, x_j) = (x_{i,1}x_{j,1})^2 + (x_{i,2}x_{j,2})^2 + x_{i,1}x_{i,2}x_{j,1}x_{j,2} \\ + x_{i,1}x_{j,1} + x_{i,2}x_{j,2} \end{aligned} \tag{4.28}$$

Obviously, one can think of kernels that are much more difficult. We list the most famous kernels:

- *Homogeneous polynomial kernel*: $K(x_i, x_j) = (x_i^T x_j)^d$ with $d \in \mathbb{N}$
- *Inhomogeneous polynomial kernel*: $K(x_i, x_j) = (x_i^T x_j + c)^d$ with $d \in \mathbb{N}, c \geq 0$
- *Gaussian kernel*: $K(x_i, x_j) = \exp\left(\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$ with $0 \neq \sigma \in \mathbb{R}$

For both polynomial kernels it is possible, but for large d very tedious, to construct the mapping ϕ explicitly. Calculating $K(x_i, x_j)$ costs $O(n)$, whereas the computation time of $\phi(x_i)$ goes as n^d, since $\phi(x_i)$ contains all monomials up to degree d, consisting of the products of elements $x_{i,1}, x_{i,2}, \ldots, x_{i,n}$. Therefore it is more convenient (if $n^d > m$) to only keep track of the kernel values.

For the Gaussian kernel it seems impossible to construct this so called *feature mapping* ϕ explicitly. Such a mapping ϕ nonetheless exists, but it maps a data point x_i to an infinite dimensional space. Therefore, it is a big problem if we have to solve the primal problem, and we have to turn to the dual problem, where only the matrix Q appears, which can be calculated using this so called Mercer kernel K. Mercer's Theorem (4.2) shows which kernels satisfy this property (Khardon, 2008).

Theorem 4.2 (Mercer)**.** *Let $K : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ be given. Then K is a* Mercer kernel *(also called a* positive definite kernel*) if and only if for any $\{x_1, \ldots x_m\}$ with $m < \infty$ the corresponding* kernel matrix *K is symmetric positive semi-definite* (PSD)*, where the (i, j)-entry of K is given by $K_{ij} = K(x_i, x_j)$.*

4.3.4 Duality gap

As usual, when we are considering the dual problem, we look at the duality gap, which is defined as the difference between the solution of the primal problem and the dual problem. If there is no duality gap present (i.e. we have *strong duality*), solving the dual problem is equivalent to solving the primal problem and in some cases much easier. In this case we have strong duality, which we can prove using the Strong Duality Theorem (Theorem A.4). If we look at the primal problem 4.12, we see that the objective function is convex in both w and each component ζ_i. Furthermore, all inequality constraints are linear and therefore convex in w, b and ζ_i. Also the set of feasible w, b and ζ is convex, because this is essentially $\mathbb{R}^{d+m+1}$. Finally, the Slater condition holds, i.e. we can find w, b, ζ_i such that the constraints hold with strict inequality, just by setting ζ_i as large as needed for every $1 \leq i \leq m$. By the *Strong Duality Theorem* A.4 we obtain that the optimal value of the primal problem p^* equals the optimal value of the dual problem d^*.

4.3.5 KKT conditions

By the Karush-Kuhn-Tucker Theorem A.5 and the following Corollary A.6 we know that if a point $(w^*, b^*, \zeta^*, \alpha^*, \beta^*)$ satisfies the KKT conditions, then (w^*, b^*, ζ^*) is a solution to the primal problem and (α^*, β^*) is a solution to the dual problem. So it seems to be a good idea to look for points satisfying these conditions. In this particular case the first condition of Equation A.6 can be represented as the following system of equations

$$0 = \nabla_w \mathcal{L}(w^*, b^*, \zeta^*, \alpha^*, \beta^*) = w^* - \sum_{i=1}^{m} (\alpha^*)_i y_i \phi(x_i) \tag{4.29}$$

$$0 = \frac{\partial}{\partial b} \mathcal{L}(w^*, b^*, \zeta^*, \alpha^*, \beta^*) = -\sum_{i=1}^{m} (\alpha^*)_i y_i \tag{4.30}$$

$$0 = \frac{\partial}{\partial \zeta_i} \mathcal{L}(w^*, b^*, \zeta^*, \alpha^*, \beta^*) = C - (\beta^*)_i - (\alpha^*)_i \tag{4.31}$$

The second condition of Equation A.6 can be represented likewise.

$$0 = \sum_{i=1}^{m} \Big((\alpha^*)_i (1 - (\zeta^*)_i - y_i (w^* \phi(x_i) + b^*) - (\beta^*)_i (\zeta^*)_i \Big) \tag{4.32}$$

Furthermore (w^*, b^*, ζ^*) and (α^*, β^*) have to be feasible for the primal and dual problem respectively, so we have another set of constraints

$$y_i(w^* \phi(x_i) + b^*) \geq 1 - (\zeta^*)_i \qquad \text{for } 1 \leq i \leq m \tag{4.33}$$

$$(\zeta^*)_i \geq 0 \qquad \text{for } 1 \leq i \leq m \tag{4.34}$$

$$(\alpha^*)_i \geq 0 \qquad \text{for } 1 \leq i \leq m \tag{4.35}$$

$$(\beta^*)_i \geq 0 \qquad \text{for } 1 \leq i \leq m \tag{4.36}$$

Together, Equations 4.29-4.36 define the KKT conditions for the problem.

4.4 Extensions

There are many extensions to the three support vector machines discussed above. At first we have *support vector clustering (SVC)*, which is in fact a technique that uses kernels to cluster unlabeled data, and therefore it is an unsupervised learning technique. The choice of the so called Gaussian kernel $K(x, x') = \exp\{-\|x - x'\|^2/\sigma^2\}$ results in nonlinear boundaries between the different categories. (Ben-Hur et al., 2001)

Support vector machines can also be used for linear regression. (Drucker et al., 1996) The minimization problem of a support vector regression (SVR) is given by

$$\begin{aligned} \min_{w\in\mathbb{R}^n, b\in\mathbb{R}} \quad & \frac{1}{2}\|w\|^2 \\ \text{s.t.} \quad & |y_i - (w^T x_i + b)| \leq \epsilon \quad \text{for } 1 \leq i \leq m \end{aligned} \tag{4.37}$$

where $\epsilon > 0$ can be arbitrarily chosen. In this way all predictions will be in a range of ϵ around the realized values y_i. Observe that y_i does not need to equal ± 1 in this case.

4.4.1 Multiclass SVM

All types of SVMs discussed above are only capable of handling two different labels, one negative and one positive. However, in many situations one would like to separate the data in three or many more different categories. There are several different methods to achieve this.

The most frequently used method is to reduce the multiclass problem into single SVMs (Hsu and Lin, 2002). This can be done in two different ways, *one-versus-all* or *one-versus-one*. The one-versus-all method consists of support vector machines that distinguish between one class (that is classified as $+1$) and all other classes together (that are classified as -1). The one-versus-one method consists of support vector machines for each pair of classes and the class of an observation is determined by a *max-wins voting strategy*. We say that a class wins a vote, if the data point is assigned to that class in an SVM. In this case a data point belongs to the class that has the most votes.

There are also some different methods that can directly classify observations in more than two groups using a single optimization problem, see for example Crammer and Singer, 2001.

4.5 Summary

In this section we have seen the three basic versions of a support vector machine, the hard-margin linear SVM, the soft-margin linear SVM and nonlinear SVM. These can all be used to classify data into two different groups, under different circumstances. The nonlinear SVM is the most general one, as it can handle both nonlinear boundaries and not separable data. If the boundary is nonlinear, it can be computationally much less expensive to look at the dual optimization problem of the SVM instead of the primal problem. The dual problem makes use of kernels that modify the observations in the data set. These kernels have to satisfy the Mercer Theorem which means that they are positive definite. There are also some extensions that can generalize the nonlinear SVM.

In the next two sections we are considering which technique to use to find the minimum of the primal or dual objective function of the support vector machine. Gradually we turn to the weighted SVM, which is a modification that can handle data sets with a large size difference between the positive and the negative class.

Chapter 5

Gradient descent techniques

Before we turn our attention to the stochastic gradient descent, it is good to first have a look at the regular (batch) *gradient descent (GD)*. The idea of the GD is to take steps in the direction of the negative of the gradient, i.e. to the place where the value of the values decreases. It is also known as the *method of steepest descent*. We will make this more formal. There are essentially three different types of gradient descent, depending on the way how the size of the steps is chosen. We will first look at gradient descent with *fixed or predetermined step size* and thereafter introduce the concepts of *backtracking line search* and *exact line search*.

The rest of this section will consist of several convergence results for in particular convex functions. Also the subgradient descent, which can be used for non-differentiable continuous convex functions and stochastic subgradient descent technique are introduced.

The goal of this section is to develop some insight on the different gradient descent techniques that can be used and come to the conclusion that the stochastic subgradient descent algorithm is the method that is the best suitable for our problem.

5.1 Introduction of the gradient descent algorithm

5.1.1 Gradient descent with fixed step size

Let $f : \mathbb{R}^n \to \mathbb{R}$ be the objective function that we want to minimize. If f is continuously differentiable in a neighborhood of some point x_1 that is not a local optimum (i.e. the gradient is nonzero), we can always take a sufficiently small step in the direction of the negative of the gradient and end up in a point x_2 for which $f(x_2) < f(x_1)$. In this case x_2 is given by

$$x_2 = x_1 - \gamma_1 \nabla f(x_1) \tag{5.1}$$

for some γ_1. Again, if x_2 is not a local optimum, we can apply the same trick to find a new point x_3 for which $f(x_3) < f(x_2)$. We hope to find a sequence $\{x_k\}_{k=1}^{\infty}$ that converges to a local minimum, or even better, to a global minimum of the function f. The procedure is visualized in Figure 5.1 on the next page, only with x_0 as the starting point.

The parameter γ_k is called the *learning rate*. Of course in the fixed step size-variant, it is set to be a constant γ. It is important to look for a good learning rate, as setting γ too high can result in overshooting a minimum, and it can even cause the series $\{x_k\}$ to diverge. Setting γ too small, can result in a very long convergence time, and for complicated functions the algorithm can get stuck in a relatively bad local minimum.

Obviously, since we only have finite computation time, we will generally not end up exactly in the desired minimum. There are essentially three ways to determine the 'end' of the optimization algorithm. The first option is to abort the algorithm if for

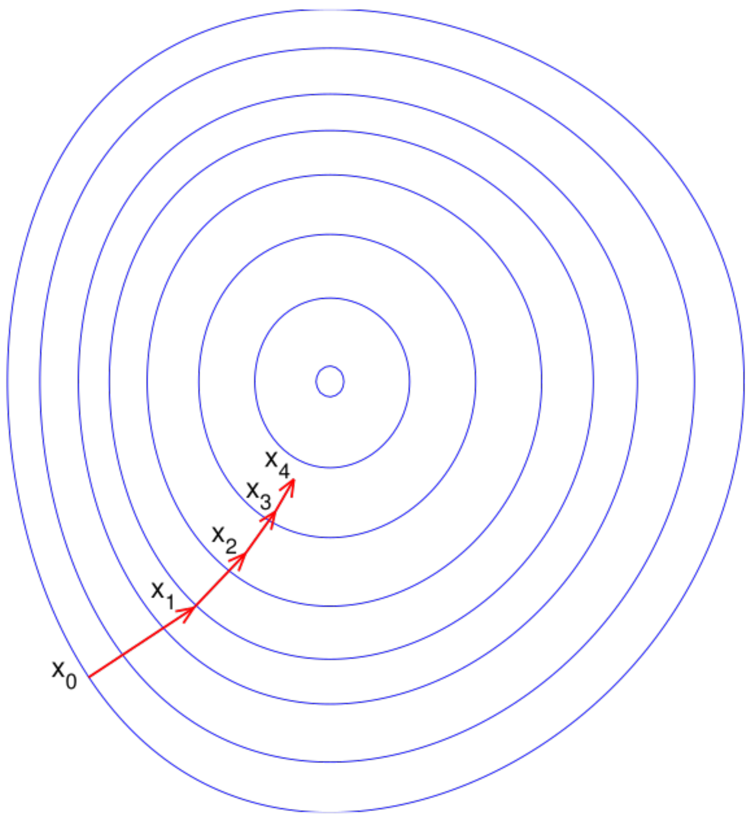

FIGURE 5.1: Gradient descent

some number of consecutive steps the function value $f(x_k)$ has decreased with at most a small $\epsilon > 0$. The second option is to terminate the algorithm after a fixed number of observations. Finally, for some specific functions f we know when we are ϵ-close to the optimum, so we can directly abort the algorithm in that case.

Example 5.1. *We can use the gradient descent algorithm to find the minimum value of $f(x) = x^2$. Note that this function is convex, so by Theorem 5.1 of the next section we know that for a small enough learning rate the sequence $\{x_k\}_{k\in\mathbb{N}}$ will converge to the global minimum 0.*

Let us take $x_1 = 2$ as starting point, and take as learning rate $\eta = 0.1$. Since $f'(x_1) = 4$, one iteration of the gradient descent algorithm will result in $x_2 = 2 - 0.4 = 1.6$. After another step we will end up in $x_3 = 1.6 - 0.32 = 1.28$, and the sequence $\{x_k\}$ will converge to the global minimum 0.

Now take a learning rate $\eta = 0.25$. One iteration of the gradient descent algorithm will result in $x_2 = 2 - 1 = 1$. After another step we will end up in $x_3 = 0.5$, converging to the global minimum 0, but at lot faster than in the previous case.

If we take a learning rate $\eta = 0.75$, our position after the first iteration will be $x_2 = 2 - 3 = -1$. The second point is $x_3 = 0.5$ and we will still end up in the global minimum, but now the values of x_k are alternating in signs, and the convergence speed is the same as with $\eta = 0.25$.

For a learning rate $\eta \geq 1$, we end up with a problem. If we take $\eta = 2$, our first two positions are $x_2 = 2 - 8 = -6$ and $x_3 = -6 + 24 = 18$, and therefore the sequence of x_k is divergent. This shows that choosing the right learning rate is very important, to guarantee both convergence and a short computation time.

5.1.2 Step sizes

There are numerous ways to choose a step size rule. The step sizes that are chosen the most can basically be divided in the following classes.

- *Constant step size*: $\gamma_k = \gamma$ independent of k.
- *Constant step length*: $\gamma_k = \frac{\gamma}{\|\nabla f(x_k)\|}$ and therefore $\|x_{k+1} - x_k\| = \gamma$ with γ constant.
- *Square summable but non-summable*: $\sum_{k=1}^{\infty} \gamma_k^2 < \infty$ but $\sum_{k=1}^{\infty} \gamma_k = \infty$, for example $\gamma_k = \frac{1}{k}$.
- *Nonsummable diminishing*: $\lim_{k\to\infty} \gamma_k = 0$ but $\sum_{k=1}^{\infty} \gamma_k = \infty$.
- *Backtracking or exact line search*: see Section 5.2.2.

5.2 Convergence of the gradient descent algorithm

Under certain conditions, we can get some nice guarantees for convergence of the gradient descent algorithm. We call a function $g : \mathbb{R}^n \to \mathbb{R}$ *K-Lipschitz* if we have for all $x, y \in \mathbb{R}^n$

$$|g(x) - g(y)| \leq K\|x - y\| \tag{5.2}$$

where $\|\cdot\|$ is the Euclidean norm on $\mathbb{R}^n$. We denote this by $g \in C_K$. The gradient $\nabla g(x)$ is in this case also bounded by K. We can see this by dividing both sides by $\|x - y\|$ and taking the limit $y \to x$.

For some specific functions, with a K-Lipschitz gradient, we have the following convergence result.[1]

Theorem 5.1. *If $\nabla f \in C_K$ and $f^* := \min_{x\in\mathbb{R}^n} f(x) > -\infty$, then the gradient descent algorithm will converge to a stationary point using a fixed learning rate $0 < \eta < \frac{2}{K}$. Furthermore*

$$\sum_{k=1}^{T} \|\nabla f(x_k)\|^2 \leq \frac{f(x_1) - f^*}{\eta\left(1 - \frac{K\eta}{2}\right)} < \infty \tag{5.3}$$

When f is a convex function, any local optimum is also a global minimum, so convergence to a global minimum is guaranteed. Observe that $\eta = \frac{1}{K}$ minimizes the fraction in Equation 5.3 and therefore it is the optimal step size to choose. Then we have

$$\sum_{k=1}^{T} \|\nabla f(x_k)\|^2 \leq 2K(f(x_1) - f^*) \tag{5.4}$$

If we define $g_T = \min_{1\leq k\leq T} \|\nabla f(x_k)\|^2$, we have $Tg_T^2 \leq 2K(f(x_1) - f^*)$ and therefore

$$g_T \leq \sqrt{\frac{2K(f(x_0) - f^*)}{T}} \tag{5.5}$$

This at least shows that the minimum value of the gradient up to iteration T has a bound that is decreasing with $T^{-\frac{1}{2}}$. But unfortunately this tell us nothing about how fast the algorithm converges to a stationary point. For (locally) convex functions this is however possible.

[1]The proofs of the lemma and theorems in this section can be found in Gordon and Tibshirani, 2012

Theorem 5.2. *Let f be convex with $\nabla f \in C_K$ and x^* be the optimal point. If $\gamma_k < \frac{1}{K}$, we have*

$$\|x_{k+1} - x^*\| \leq \|x_k - x^*\| \tag{5.6}$$

This is a nice property, but it does not tell us anything about how fast x_k goes to x^* and its rate of convergence can be arbitrarily slow. Luckily, this only depends on the choice of our learning rate γ_k.

Theorem 5.3. *Let f be convex with $\nabla f \in C_K$ and x^* be the optimal point. If $\gamma_k \leq \frac{1}{K}$ for all $k \leq T$ we have*

$$f(x_T) - f(x^*) \leq \frac{\|x_1 - x^*\|^2}{\sum_{k=1}^{T} \gamma_k \left(1 - \frac{K\gamma_k}{2}\right)} \tag{5.7}$$

We see that the denominator of Equation 5.7 is maximized if $\gamma_k = \frac{1}{K}$ and in this case $f(x_T) - f(x^*) \leq \frac{2K\|x_1 - x^*\|^2}{T}$. So we need at most

$$T = \frac{2K\|x_1 - x^*\|^2}{\epsilon} \tag{5.8}$$

iterations to obtain an ϵ-accurate solution, i.e. such that $f(x_T) - f(x^*) \leq \epsilon$.

5.2.1 Strong convexity

It would seem nice, if we could also say something about how good the Tth estimate x_T already is, i.e. we would like to have some estimate of $f(x_T) - f^*$ or $\|x_T - x^*\|$, independent of our choice of x_1. Unfortunately, convexity alone is not strong enough to guarantee some bound. Therefore we need the concept of *strong convexity*. We call f a strongly convex function on X if there is a constant $m > 0$ such that $\nabla^2 f(x) - mI$ is PSD for all $x \in X$. We denote this by $\nabla^2 f(x) \succeq mI$. A direct implication of this definition is that every eigenvalue of the Hessian of f is at least m. A useful consequence is the following

Lemma 5.4. *If f is strongly convex on X, we have for all $x, y \in X$*

$$f(y) \geq f(x) + (\nabla f(x))^T(y - x) + \frac{m}{2}\|y - x\|^2 \tag{5.9}$$

Proof. By the generalized version of the mean value theorem we have for every $x, y \in X$

$$f(y) = f(x) + (\nabla f(x))^T(y - x) + \frac{1}{2}(y - x)^T\nabla^2 f(z)(y - x) \tag{5.10}$$

where z is some point on the line segment $[x, y]$. Since f is strongly convex on X we have $(y - x)^T\nabla^2 f(z)(y - x) \geq m\|y - x\|^2$. Therefore Equation 5.9 holds. □

The independent upper bound for $f(x_k) - f^*$ and $\|x_k - x^*\|$ can be obtained using the previous lemma.

Lemma 5.5. *If f is strongly convex on X with parameter m we have for every $x \in X$*

$$f(x) - f^* \leq \frac{1}{2m}\|\nabla f(x)\|^2 \tag{5.11}$$

$$\|x - x^*\| \leq \frac{2}{m}\|\nabla f(x)\| \tag{5.12}$$

The previous lemma has an interesting consequence. For every $x \in X$ holds that the solution x^* is located within a ball of radius $\frac{2}{m}\|\nabla f(x)\|$ around x.

We can also obtain an upper bound on $f(y)$, if we also have an upper bound M on the largest eigenvalue of $\nabla^2 f(x)$ for each $x \in X$, which we denote by $\nabla^2 f(x) \preceq MI$. The following property then holds.

Lemma 5.6. *For f with $\nabla^2 f(x) \preceq MI$ and all $x, y \in X$ holds*

$$f(y) \leq f(x) + (\nabla f(x))^T(y-x) + \frac{M}{2}\|y-x\|^2 \tag{5.13}$$

Proof. As in Lemma 5.5 we have by the generalized version of the mean value theorem for every $x, y \in X$

$$f(y) = f(x) + (\nabla f(x))^T(y-x) + \frac{1}{2}(y-x)^T\nabla^2 f(z)(y-x) \tag{5.14}$$

where z is some point on the line segment $[x, y]$. Since $\nabla^2 f(x) \preceq MI$ we have $(y - x)^T\nabla^2 f(z)(y-x) \leq M\|y-x\|^2$. Therefore Equation 5.13 holds. □

Therefore, the *condition number*, which is the ratio of the largest eigenvalue and the smallest eigenvalue of $\nabla^2 f(x)$, has an upper bound $k = \frac{M}{m}$. If k is close to 1 we call the matrix *well-conditioned*. However, if k is much larger than 1, we call the matrix *ill-conditioned*.

Most of the times m and M are not known. Therefore we cannot use them for a termination criterion of the algorithm. But the idea is that if the gradient of f at x is small enough, the difference between $f(x)$ and f^* is small. More formally, if we stop the algorithm when $\|\nabla f(x_T)\| \leq \eta$, for $\eta < \sqrt{m\epsilon}$, then we have by Equation 5.11, $f(x_T) - f^* < \epsilon$.

We now arrive at the next theorem, which gives an upper bound on $f(x_T) - f^*$ in terms of $f(x_1) - f^*$ using *exact* or *backtracking line search*.

Theorem 5.7. *Let f satisfy for every $x \in X$*

$$mI \preceq \nabla^2 f(x) \preceq MI \tag{5.15}$$

Then, using gradient descent with step size determined by exact or backtracking line search, we have for some $c < 1$

$$f(x_T) - f^* \leq c^{T-1}(f(x_1) - f^*) \tag{5.16}$$

5.2.2 Gradient descent with line search

The idea of both *backtracking line search* and *exact line search* is to find an ideal learning rate γ_k for each step. For exact line search, we choose γ_k along the line $x_k - \gamma\nabla f(x_k)$ such that f is minimized, i.e.

$$\gamma_k = \arg\min_{\gamma>0} f(x_k - \gamma\nabla f(x_k)) \tag{5.17}$$

Then we do the update $x_{k+1} = x_k - \gamma_k\nabla f(x_k)$. We keep on repeating this until $\|\nabla f(x_k)\| < \epsilon$. However, this algorithm is only useful if we can compute this minimum analytically, which is not possible for most functions.

When it is not possible to use exact line search, we can turn our attention to backtracking line search, which is in general much more effective and simple. We choose two parameters $0 < \alpha < 0.5$ and $0 < \beta < 1$. Then we start with $\gamma_k = 1$ and as long as

$$f(x_k - \gamma_k \nabla f(x_k)) > f(x) - \alpha\gamma_k \|\nabla f(x)\|^2 \tag{5.18}$$

we keep on doing the update $\gamma_k = \beta\gamma_k$. Eventually, depending on the function f and the parameters α an β, we have for γ_k small enough by a Taylor expansion

$$\begin{aligned} f(x_k - \gamma_k \nabla f(x_k)) &\approx f(x_k) - \gamma_k \|\nabla f(x_k)\|^2 \\ &< f(x_k) - \alpha\gamma_k \|\nabla f(x_k)\|^2 \end{aligned} \tag{5.19}$$

since $-\nabla f(x)$ is a direction in which f decreases and $-\|\nabla f(x_k)\|^2 < 0$. After performing the update $x_{k+1} = x_k - \gamma_k \nabla f(x_k)$ we compute $\nabla f(x_{k+1})$ and repeat all steps for $k = k+1$, until $\|\nabla f(x_k)\| < \epsilon$.

The condition that ∇f is Lipschitz is a quite realistic one, however, our strong convexity condition is not easily attained. Therefore, many of the derivations on convergence bounds are not that useful in practice.

Example 5.2. *Let $f(x) = \frac{1}{2}\|z - Ax\|^2$, which can be an objective function for linear regression. Therefore*

$$\nabla f(x) = -A^T(z - Ax) \tag{5.20}$$

$$\nabla^2 f(x) = A^T A \tag{5.21}$$

Observe that for any $x, y \in X$ we have

$$\|\nabla f(x) - \nabla f(y)\| = \|A^T A(x - y)\| \leq \lambda_{max}\|x - y\| \tag{5.22}$$

where λ_{max} is the largest eigenvalue of $A^T A$, which is a PSD matrix. Therefore $\nabla f(x)$ is Lipschitz, with parameter $\lambda_{max} < \infty$. Observe that this is equivalent to $\nabla^2 f(x) \preceq \lambda_{max} I$ for all $x \in X$. However, the strong convexity condition for f requires the existence of some $\lambda_{min} > 0$ such that

$$\nabla^2 f(x) \succeq \lambda_{min} I \tag{5.23}$$

which is only the case if $A^T A$ is PD, i.e. if A is non-singular.

5.2.3 Gradient descent algorithm

We can summarize the gradient descent technique into three simple algorithmic schemes. Note that the while-condition only works if the function f is strongly convex. In all other cases we execute line 3 and 4 of the algorithm a fixed number of times.

Algorithm 5.1 Gradient descent algorithm

```
1: Start with an initial guess $x_1$, set $k = 1$ and choose a positive tolerance $\epsilon$.
2: while $\|\nabla f(x_k)\| < \sqrt{\epsilon}$ do
3:     Compute $\gamma_k$, either using the step size rules of Section 5.1.2 or determined by Algorithm 5.2 or 5.3.
4:     Update $x_{k+1} = x_k - \gamma_k \nabla f(x_k)$ and set $k = k + 1$.
5: end while
6: return $f(x_k)$
```

Algorithm 5.2 Exact line search

```
1: return $\gamma_k = \arg\min_{\gamma>0} f(x_k - \gamma \nabla f(x_k))$
```

Algorithm 5.3 Backtracking line search

```
1: Choose $0 < \alpha < 0.5$ and $0 < \beta < 1$ and set $\gamma_k = 1$.
2: while $f(x_k - \gamma_k \nabla f(x_k)) > f(x_k) - \alpha\gamma_k \|\nabla f(x_k)\|^2$ do
3:     $\gamma_k = \beta\gamma_k$
4: end while
```

5.3 Subgradient descent

Observe that the gradient descent method can only be used for differentiable convex objective functions. We can drop the differentiability assumption by looking at the so called *subgradient*. Let $f : X \mapsto \mathbb{R}$ be a convex function for $X \subset \mathbb{R}^n$ convex, then we call $v \in \mathbb{R}^n$ a subgradient of f at x_0 if

$$f(x) - f(x_0) \geq v^T(x - x_0) \tag{5.24}$$

for all $x \in X$. If $n = 1$, this means that the straight line $y = v(x - x_0)$ is everywhere under the graph of f. In the case that f is not differentiable at the point x_0, we clearly can have multiple vectors v that satisfy Equation 5.24 and we call the set of all subgradients at x_0 the *subdifferential* at x_0, which is denoted by $\partial f(x_0)$. The following lemma holds.[2].

Lemma 5.8. *Let $f : X \mapsto \mathbb{R}$ be a convex function with $X \subset \mathbb{R}^n$ a convex set. Then we have the following properties for $x_0 \in int(X)$*

- *The subdifferential of f at x_0 is a non-empty convex set.*
- *x_0 is a global minimum of the function f if and only if 0 is contained in the subdifferential of f at x_0.*

If f is differentiable at x_0, $\nabla f(x_0)$ satisfies the subgradient property. As is stated in the next theorem, this is the only element of the subdifferential at x_0.

Theorem 5.9. *Let $f : X \to \mathbb{R}$ be a convex function with $X \subset \mathbb{R}^n$ a convex set. Then f is differentiable at x_0 if and only if $\partial f(x_0) = \{\nabla f(x_0)\}$.*

Furthermore, strong convexity is also defined for convex functions without differentiability. Analogous to Lemma 5.5, a convex function f is m-strongly convex on X if we have for all $x, y \in X$ and all subgradients g of f at x

$$f(y) \geq f(x) + g^T(y - x) + \frac{m}{2}\|y - x\|^2 \tag{5.25}$$

In the gradient descent algorithm we have the update $x_{k+1} = x_k - \gamma_k \nabla f(x_k)$. Since $\nabla f(x_k)$ may not be defined if we drop the differentiability assumption, we replace this by a subgradient g_k of f at x_k, i.e.

$$x_{k+1} = x_k - \gamma_k g_k \tag{5.26}$$

[2]The proofs of the lemma and theorems in this section can be found in Boyd, Xiao, and Mutapcic, 2003 and Balder, 2010

There is a possibility that g_k is not a descent direction, for example if we have $f(x) = (|x_1|+|x_2|)$. At $x_k = 0$ we can take $g_k = (0.5, 0)^T$. Clearly $g_k \in \partial f(x_k)$, but $f(x_k - \gamma_k g_k) = |0.5\gamma_k| > 0 = f(x_k)$ for any $\gamma_k > 0$. Therefore, after every step we set

$$f_k^{(\text{best})} = \min\{f_{k-1}^{(\text{best})}, f(x_k)\} \tag{5.27}$$

to keep track of the minimal value achieved up to that moment. Clearly this will be a decreasing sequence with a limit, possibly equal to $-\infty$.

5.3.1 Convergence results for the subgradient descent algorithm

Also, in the subgradient method we can look at the convergence properties. Since we have dropped the differentiability assumption, we cannot adopt the arguments of the previous section, since we cannot assume that $\nabla f(x_k) \in C_K$. Instead of this we assume that the subgradients are uniformly bounded in norm, i.e. if for all k,

$$\|g_k\|_2 := \sup\{g | f(x) - f(x_k) \geq g(x - x_k) \quad \forall x \in X\} < G \tag{5.28}$$

This for example happens if f is K-Lipschitz, since for all $x, y \in X$ we then have

$$K\|x - y\| \geq |f(x) - f(y)| \geq \|g\|\|x - y\| \tag{5.29}$$

i.e. $\|g\|_2 \leq K$.

Theorem 5.10. *Let f be convex and assume that there is a $G < \infty$ such that $\|g_k\|_2 \leq G$ for all k. Then we obtain the following bound*

$$f_{T+1}^{(best)} - f^* \leq \frac{\|x_1 - x^*\|^2 + G^2 \sum_{k=1}^{T} \gamma_k^2}{2\sum_{k=1}^{T} \gamma_k} \tag{5.30}$$

Depending on the chosen step size we get different convergence bounds. If we consider a constant step size $\gamma_k = \gamma$ we obtain

$$f_{T+1}^{(best)} - f^* \leq \frac{\|x_1 - x^*\|^2 + G^2 T \gamma^2}{2\gamma T} \to \frac{G^2\gamma}{2} \tag{5.31}$$

The optimal bound for this expression is attained at $\gamma_k = \frac{\|x_1 - x^*\|}{G\sqrt{k}}$. In that case we obtain according to Tibshirani, 2012

$$f_k^{(best)} - f^* \leq \frac{\|x_0 - x^*\| G}{\sqrt{k+1}} \tag{5.32}$$

However, this learning rate is not realistic, as we do not know $\|x_0 - x^*\|$ in advance. Therefore we propose $\gamma_k = \frac{1}{\lambda G\sqrt{k+1}}$. This results in, for small λ,

$$f_k^{(best)} - f^* \leq \frac{\lambda\|x_0 - x^*\| + \frac{1}{\lambda}}{\frac{2\sqrt{k+1}}{G}} \to \frac{G}{2\lambda\sqrt{k+1}} \tag{5.33}$$

Finally, we call a function f strongly convex with parameter m if

$$f(y) \geq f(x) + g^T(y - x) + \frac{m}{2}\|y - x\|^2 \tag{5.34}$$

holds for all $x, y \in X$ and all subgradients $g \in \partial f(x)$.

5.3.2 Subgradient descent algorithm

We also summarize the subgradient descent technique in an algorithmic scheme. Again, the while-condition is working if we are dealing with a strongly convex function f. In all other cases we execute lines 5-7 a fixed number of times.

Algorithm 5.4 Subgradient descent algorithm

```
1: Start with an initial guess $x_0$ and set $k = 0$, and choose a positive tolerance $\epsilon$.
2: Choose a step size rule and calculate the value $\eta$ such that $f(x_k) - f^* < \epsilon$ if $\|g_k\| < \eta$.
3: Set $f_0^{(best)} = f(x_0)$.
4: while $\|g_k\| < \eta$ do
5:     Choose a subgradient $g_k$ of $f$ at $x_k$ and compute $\gamma_k$.
6:     Update $x_{k+1} = x_k - \gamma_k g_k$ and set $f_{k+1}^{(best)} = \min\{f(x_{k+1}), f_k^{(best)}\}$
7:     Set $k = k + 1$.
8: end while
9: return $f_k^{(best)}$.
```

5.4 Stochastic (sub)gradient descent

While the (sub)gradient descent algorithm is quite efficient for small data sets and smooth convex functions, things can get really nasty if the size of the data set gets huge or the function has non-global extremal points. One of the solutions to these issues is the *stochastic gradient descent*. Instead of computing the gradient of the objective function with respect to all observations, we choose one observation at random and compute the gradient with respect to this observation. Therefore the method is called stochastic.

We will make this idea more formal. Consider an objective function based on m (training) observations, which we can write as

$$f(x) = f_0(x) + \frac{1}{m}\sum_{i=1}^{m} f_i(x) \tag{5.35}$$

where f_i only depends on the value of observations i and f_0 is independent of the observations. Then the stochastic gradient restricted to observation i equals

$$\nabla_{(i)} f(x) = \nabla f_0(x) + \nabla f_i(x) \tag{5.36}$$

We call this an *individual gradient*. As in the gradient descent algorithm (Equation 5.1), the target variable x is updated using this gradient, given by

$$x_1 = x_0 - \gamma_0 \nabla_{(i)} f(x_0) \tag{5.37}$$

The choice of the learning rate γ_k is important, as it highly influences whether or not the stochastic gradient descent converges, especially when the function has many local extremal points. For a gradient descent algorithm, as the objective function approaches its global minimum, the (total) gradient obviously converges to zero. However, this does not need to happen with the individual gradients. Therefore, choosing a constant learning rate for the stochastic gradient descent is not favorable, as the objective function will continue to fluctuate a lot around the optimal value. This problem can be fixed by taking a diminishing learning rate, which causes this fluctuation to disappear eventually. As we see in Figure 5.2 on the next page, the GD algorithm proceeds a lot smoother to the optimal value than the SGD algorithm.

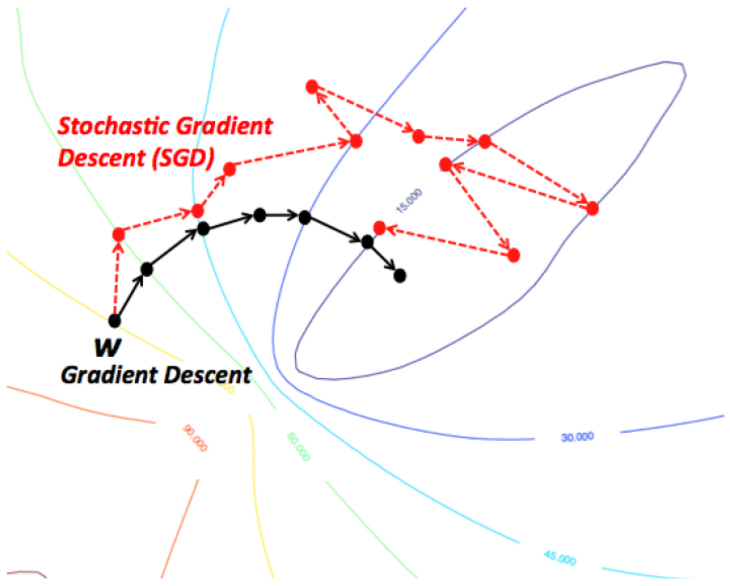

FIGURE 5.2: Stochastic gradient descent (red) vs. gradient descent (black)

There are several advantages of the stochastic gradient descent compared to the normal GD. If we have a function with many local minima, once a gradient descent algorithm has come close to a local minimum, it will stay there, since the gradient update generally forces the objective function to go the nearby minimum. The stochastic gradient descent algorithm can iterate between the diffent local minima, because the individual gradients do not need to direct the objective function to a local minimum. Depending on the characteristics of the function and the chosen learning rate, the SGD algorithm will usually end up in one of the 'better' local minima. Besides that, since the individual gradient is far more easy to compute if m is large, one iteration of the SGD algorithm is far less costly than a GD algorithm. Of course, the SGD algorithm needs many more iterations than the GD algorithm, but it is shown that for large data sets, a value close to the optimal value can be reached in a much shorter time period.

There exists also a subgradient version of the stochastic gradient descent. In this case, instead of taking a $v \in \mathbb{R}^n$ such that for fixed $x_0 \in X$ we have

$$f(x) - f(x_0) \geq v^T(x - x_0) \tag{5.38}$$

for all $x \in X \subset \mathbb{R}^n$, we take a random vector $\widehat{v} \in \mathbb{R}^n$ such that $\mathbb{E}[\widehat{v}] = v$ and v satisfies the above equation. The update rule for the subgradient descent algorithm stays the same as before, except that we do not take a minimum such as in Equation 5.27, since the stochastic (sub)gradient descent algorithm does not requires the objective function to decrease with every iteration. Furthermore the convergence rates, as given in the next section are the same for the stochastic subgradient descent algorithm as for the stochastic gradient descent algorithm, in contrary to the non-stochastic case.

In Section 6.2 the so called *averaged stochastic gradient descent* is introduced. Using that technique the output of the model is not simply the final iterate of the algorithm, but some sort of average over the previous iterations.

5.4.1 Convergence results of SGD algorithms

There are several studies discussing the convergence and convergence rate of the stochastic (sub)gradient descent algorithm. They all require some kind of convexity for the objective function f and some constraint on the norm of the (sub)gradients. The following theorem, including a proof, can be found in Shamir and Zhang, 2013 (Theorem 1).

Theorem 5.11. *Suppose f is λ-strongly convex and $\mathbb{E}[\|\hat{v}_t\|_2] \leq G^2$ for all iterates t. Consider SGD with step sizes $\eta_t = 1/\lambda t$. Then for any number of iterations $T > 1$ we have*

$$\mathbb{E}[f(x_T) - f(x^*)] \leq \frac{17G^2(1 + \log T)}{\lambda T} \tag{5.39}$$

This theorem assures us that, at least in expectation, the objective function will converge to the optimal value with rate $\log T/T$, whereas for certain strongly convex functions the gradient descent algorithm obtains a much faster looking convergence rate $1/c^T$ (Theorem 5.7). However, taking into account that the time needed for one iteration of the SGD algorithm is independent of the number of observations, whereas the time needed for one iteration of the GD algorithm grows linearly with the size of the data set, we see that the SGD algorithm may still converge faster at first. Therefore the SGD algorithm is a good alternative when the amount of time is the limiting factor. Nonetheless, as time continues, because of the exponential convergence rate, the GD algorithm will outperform the SGD algorithm regardless of the size of the data set.

The SGD algorithm is also useful for objective functions that are convex but not strongly convex, as the following theorem shows. This only works if the size of the set X of feasible x is small enough, i.e. at least bounded. The proof of this theorem can also be found in Shamir and Zhang, 2013 (Theorem 2).

Theorem 5.12. *Suppose f is convex and for some constants D, G it holds that $\mathbb{E}[\|\hat{v}_t\|_2] \leq G^2$ for all iterates t, and for all $x, x' \in X$ we have $\|x - x'\| < D$. Consider SGD with step sizes $\eta_t = c/\sqrt{t}$ where $c > 0$. Then for any number of iterations $T > 1$ we have*

$$\mathbb{E}[f(x_T) - f(x^*)] \leq \left(\frac{D^2}{c} + cG^2\right) \frac{2 + \log T}{\sqrt{T}} \tag{5.40}$$

5.4.2 Stopping criteria

For the (sub)gradient descent method, we can set a stopping criterion based on size of the norm of the (sub)gradient. However, for the stochastic gradient method, we cannot do such a thing as the norm of an individual gradient will generally not tend to zero close to a local minimum and the total gradient should not be calculated in the SGD algorithm. At first we can define a stopping criterion for the algorithm by setting a maximum number of iterations or a maximum amount of time. This is quite straightforward and does not lead to unexpected situations. Secondly, we can base our stopping criterion on some improvement threshold, i.e. the algorithm stops if the objective function f has not decreased by at least $\epsilon\%$ in the past τ iterations of the algorithm. The idea is that we have come so close to a local minimum, such that it is not that useful to perform any more updates. For a gradient descent algorithm with a strongly convex function, this would be a perfect stopping criterion, since convergence always takes place. However, since for a SGD algorithm we only have convergence in expectation, this can lead to situation in which we are getting even further away from a local minimum, but the algorithm is stopped due to the improvement threshold. Therefore, in the case of the SGD algorithm, it is better to use the stopping criterion based on a maximum number of iterations or a maximum amount of time.

5.4.3 Stochastic gradient descent algorithm

Below we see a formal scheme for the simple stochastic gradient algorithm, with a stopping criterion based on a maximum number of iterations. Note that this maximum is not bounded by the number of observations, as an observation can be used multiple times in the algorithm.

Algorithm 5.5 Stochastic gradient descent algorithm

```
1: Start with an initial guess $x_0$, set $k = 0$, take a diminishing learning rate $\gamma_k$ and choose a maximum number of
   iterations $T$.
2: for $0 \leq k \leq T-1$ do
3:     Take $i$ between 1 and $m$ randomly.
4:     Compute the learning rate $\gamma_k$ and the individual gradient $\nabla_{(i)} f(x_k)$.
5:     Update $x_{k+1} = x_k - \gamma_k \nabla_{(i)} f(x_k)$ and set $k = k+1$.
6: end for
7: return $f(x_T)$
```

The version with a subgradient is quite similar to this scheme, with the exception of line 4. For the stochastic subgradient descent algorithm, a subgradient is computed in this line.

5.5 Summary

In this chapter we have seen several algorithms to perform an optimization of some objective function, namely the (sub)gradient and the stochastic (sub)gradient descent algorithm. All algorithms are based on computing a (sub)gradient, which points into a direction of fastest increase. The algorithm consists of performing an update of the target variable into the direction of fastest **decrease**, i.e. in the direction of the negative gradient. If the objective function is (locally) convex, we can usually obtain some convergence rate for the objective function to go to a (local) minimum. The stochastic (sub)gradient descent algorithm is preferred over the (sub)gradient descent algorithm in the case of a function with many local minima, since the GD algorithm easily gets stuck in a non-optimal local minimum. However, the SGD algorithm only converges in expectation, where the GD algorithm converges absolutely.

Also many results about convergence (rates) were stated for all these algorithms. We summarized these results in Table 5.1 below. Note that for a comparison of the SGD algorithm and the GD algorithm, we have to take into account that one iteration of the GD algorithm involves calculating m individual gradients, whereas the SGD algorithm only calculates one gradient per iteration. If we use an 'easy' learning rate, such as $\gamma_k = 1/(\lambda k + \lambda_0)$, the computation of the gradients will be the main contribution to the total time used. Therefore we display the convergence rate in terms of calculations of individual gradients.

Algorithm	**Convex**	**Strongly convex**
Gradient descent	m/T	-
Gradient descent with exact line search	-	$1/c^{T/m}$
Gradient descent with backtracking line search	-	$1/c^{T/m}$
Subgradient descent	$\sqrt{m/T}$	$\sqrt{m/T}$
Stochastic (sub)gradient descent	$\log T/\sqrt{T}$	$\log T/T$

TABLE 5.1: Convergence rates of gradient descent algorithms

As we have seen, the gradient descent algorithm will turn out to perform much faster (and also converges not only in expectation) in the limit $T \to \infty$, but for small T and large m, using the stochastic gradient descent can result in an important improvement. This is illustrated in Figure 5.3 below for a λ-strongly convex function.

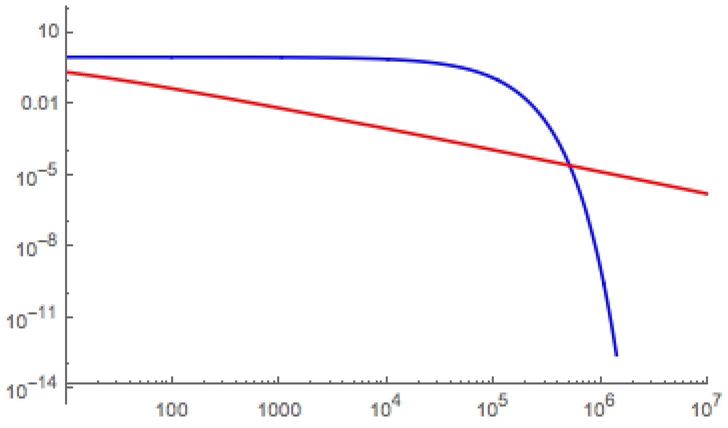

FIGURE 5.3: Convergence of SGD algorithm (red) and GD algorithm (blue)

On the x-axis we see the number of iterations, i.e. the number of computations for the individual gradients and on the y-axis we see the (expected) loss. Generally we do about 100,000 to 10,000,000 iterations for the development of our model, so if we have a λ-strongly convex differentiable function it is not clear which algorithm to use.

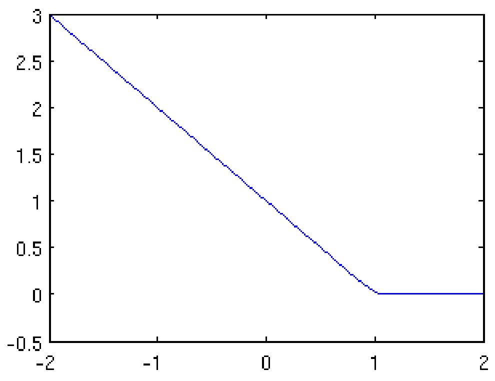

FIGURE 5.4: Graph of hinge loss function

However, since the objective function f is non-differentiable, due to the hinge loss term (see Figure 5.4 above), we definitely have to choose between the subgradient method and the stochastic subgradient method. The asymptotic convergence rate of the subgradient method equals $\frac{G}{2\lambda\sqrt{T/m}}$, with learning rate $\gamma_k = \frac{1}{\lambda G\sqrt{k}}$. For the stochastic version with learning rate $\gamma_k = \frac{1}{\lambda k}$ we have a convergence rate of $\frac{17G^2(1+\log(T))}{\lambda T}$. If we put in reasonable values for G, λ, T and m (100, 10^{-6}, 10^6 and 10^6 respectively), we observe a value of $5 * 10^7$ for the convergence rate of the subgradient method and a value of $2.52 * 10^6$ for the stochastic subgradient method. In Figure 5.5 below, we also show a plot of both convergence rates as a function of G (for $T = m = \frac{1}{\lambda} = 10^6$).

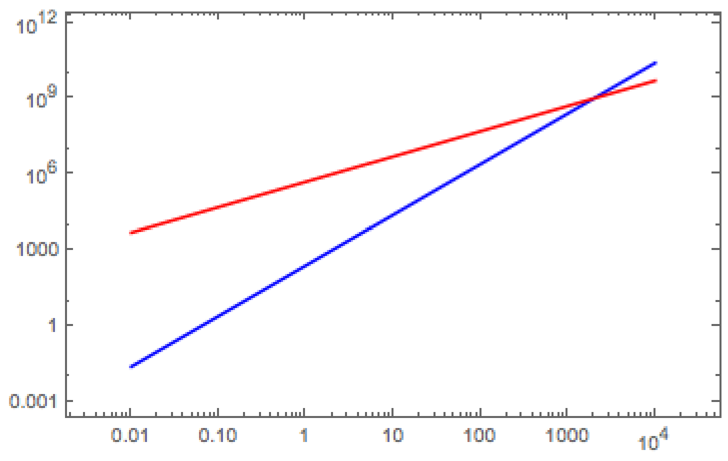

FIGURE 5.5: Convergence rates of stochastic subgradient descent (blue) and subgradient descent (red) as a function of G

In this figure above we see G on the x-axis, and the (expected) loss on the y-axis of the (stochastic) subgradient descent algorithm. Experience shows that G is between 1 and 1000, so the stochastic subgradient descent algorithm is preferred over the non-stochastic version.

Throughout the rest of the thesis we will mostly use the term *stochastic gradient descent algorithm* or *SGD algorithm* instead of the stochastic subgradient descent algorithm, as this is common in literature.

Chapter 6

Weighted SVM and averaged SGD

During the developmental period of this thesis, we have tried many different ways to implement an SVM to the data set of FHLMC, in particular the dual version with kernels. However, the so called *weighted support vector machine (wSVM)*, a modification that we initially conceived ourselves (however the idea is also present in a few other papers that can be found in Section 6.1), showed by far the most promising results.

In this chapter we will focus on this weighted SVM combined with the technique of the averaged stochastic gradient descent. We have found a proof for three new theorems concerning the performance of these two topics. They all should give us an intuition why this technique works.

6.1 Weighted SVM

For a general non-linear support vector machine, the objective function is given by

$$f(w,b) = \frac{\lambda}{2} w^T w + \frac{1}{m} \sum_{i=1}^{m} \max\{0, 1 - y_i(w^T \phi(x_i) + b)\} \tag{6.1}$$

In the case that the data is not (non-linearly) separable and one class has significantly more data points than the other, we can introduce a weighted version of SVM. This can be done by multiplying the hinge-loss function if $y_i = -1$ by a weight $z > 0$ and if $y_i = +1$ by a weight $\frac{1}{z}$. The objective function for *weighted SVM* is therefore

$$f(w,b,z) = \frac{\lambda}{2} w^T w + \frac{1}{m} \sum_{i=1}^{m} \left(\frac{1}{z}\right)^{y_i} \max\{0, 1 - y_i(w^T \phi(x_i) + b)\} \tag{6.2}$$

Clearly as $z \to 0$, the minimum value of the mapping $(w,b) \mapsto f(w,b,z)$ is attained at $(0,1)$, since the contribution of the data points in the positive class to the cost function will dominate the contribution of the data points in the negative class, i.e. all loans are predicted not to go into default. For the same reason, as $z \to \infty$, the minimum value of this mapping is attained at $(0,-1)$, i.e. all loans are predicted to go into default. These outcomes are not interesting to us, since they do not give us any division of the data in two groups with a different PD. Theorem 6.1 shows that under very mild conditions there exists some z^* for which we obtain a non-trivial solution that actually divides the data in two different groups.

We use the following definitions

$$PD = \frac{1}{m}\sum_{i=1}^{m} \mathbb{1}\{y_i = -1\} \tag{6.3}$$

$$m_- = m \cdot PD = \sum_{i=1}^{m} \mathbb{1}\{y_i = -1\} \tag{6.4}$$

$$m_+ = m \cdot (1 - PD) = \sum_{i=1}^{m} \mathbb{1}\{y_i = +1\} \tag{6.5}$$

$$G_1 = \{(x_i, y_i) : (w^*)^T \phi(x_i) + b < 0\} \tag{6.6}$$

$$G_2 = \{(x_i, y_i) : (w^*)^T \phi(x_i) + b > 0\} \tag{6.7}$$

$$PD_1 = \frac{1}{|G_1|}\sum_{i:(x_i,y_i)\in G_1} \mathbb{1}\{y_i = -1\} \tag{6.8}$$

$$PD_2 = \frac{1}{|G_2|}\sum_{i:(x_i,y_i)\in G_2} \mathbb{1}\{y_i = -1\} \tag{6.9}$$

Theorem 6.1. *Let $\{(x_i, y_i)\}_{i=1}^m$ be a data set with $0 < PD < 1$ and $\phi : \mathbb{R}^n \to \mathbb{R}^d$ be any feature mapping. Assume that for at least one coordinate j between 1 and d the total sum of $(\phi(x_i))_j$ over all data points with $y_i = -1$ is different from the total sum over all data points with $y_i = +1$, i.e.*

$$\sum_{i:y_i=-1} (\phi(x_i))_j \neq \sum_{i:y_i=+1} (\phi(x_i))_j \tag{6.10}$$

Then there exists $z^* \geq 0$, $w^* \in \mathbb{R}^d$, $b^* \in \mathbb{R}$ and $\lambda > 0$ such that

$$f(w^*, b^*, z^*) < \min\{f(0, 1, z^*), f(0, -1, z^*)\} \tag{6.11}$$

$$G_1 \neq \emptyset \neq G_2 \tag{6.12}$$

i.e. we have a non-trivial global minimum of $(w, b) \mapsto f(w, b, z^*)$ and the separating equation $(w^*)^T\phi(x_i) + b$ separates the data in two non-empty groups.

Proof. Observe that $0 < PD < 1$ implies that there exists at least one data point in each of the two classes, so $m \geq 2$. Let $z^* = \sqrt{\frac{1-PD}{PD}}$. Then we see for $b \in [-1, 1]$

$$\begin{aligned} f(0, b, z^*) &= \frac{z^*}{m}\sum_{i=1}^{m} \mathbb{1}\{y_i = -1, b \geq -1\}(1 + b) \\ &\quad + \frac{1}{z^* m}\sum_{i=1}^{m} \mathbb{1}\{y_i = +1, b \leq 1\}(1 - b) \qquad (6.13) \\ &= \frac{z^*}{m}\sum_{i=1}^{m} \mathbb{1}\{y_i = -1\}(1 + b) + \frac{1}{z^* m}\sum_{i=1}^{m} \mathbb{1}\{y_i = +1\}(1 - b) \\ &= \frac{z^*}{m} m_-(1 + b) + \frac{1}{z^* m} m_+(1 - b) \\ &= \sqrt{\frac{1 - PD}{PD}}\frac{m}{m} \cdot PD(1 + b) + \sqrt{\frac{PD}{1 - PD}}\frac{m}{m} \cdot (1 - PD)(1 - b) \\ &= 2\sqrt{PD(1 - PD)} \qquad (6.14) \end{aligned}$$

Clearly, this is independent of b. By convexity of the function $(w, b) \mapsto f(w, b, z)$, we know that we must have for $b \in [-1, 1]$

$$f(0, b, z^*) = \min_{b' \in \mathbb{R}} f(0, b', z^*) \tag{6.15}$$

Our goal is to prove the existence of a vector $w^* \in \mathbb{R}^d$ (that is necessarily non-zero by the above argument) and $b^* \in \mathbb{R}$ such that $f(w^*, b^*, z^*) < 2m\sqrt{PD(1 - PD)}$. Now we use the total sum-condition of the theorem. Assume without loss of generalization that the LHS of Equation 6.10 is less than the RHS. For notational convenience we use $x_{i,j} = (\phi(x_i))_j$ and $w_i^* = w_i$ as we set $w_j^* = 0$ for all coefficients $j \neq i$. We can also assume that we have $0 \leq x_{i,j} \leq 1$ for all i, otherwise we can modify our feature mapping ϕ to achieve this. By the same reasoning we can assume the existence of i, i' such that $x_{i,j} < 0.5$ and $x_{i',j} > 0.5$, since $m \geq 2$. Therefore, by taking $w^* = 2$ and $b^* = -1$, we obtain $-1 \leq w^* x_{i,j} + b^* \leq 1$ for all $1 \leq i \leq m$. Observe that we have in that case

$$\begin{aligned}
f(w^*, b^*, z^*) &= \frac{\lambda}{2}(w^*)^2 \\
&\quad + \frac{z^*}{m} \sum_{i=1}^{m} \mathbb{1}\{y_i = -1, w^* x_{i,j} + b^* \geq -1\}(1 + (w^* x_{i,j} + b^*)) \\
&\quad + \frac{1}{z^* m} \sum_{i=1}^{m} \mathbb{1}\{y_i = +1, w^* x_{i,j} + b^* \leq 1\}(1 - (w^* x_{i,j} + b^*)) \\
&= 2\lambda + \frac{z^*}{m} \sum_{i=1}^{m} \mathbb{1}\{y_i = -1\} 2x_{i,j} \\
&\quad + \frac{1}{z^* m} \sum_{i=1}^{m} \mathbb{1}\{y_i = +1\}(2 - 2x_{i,j}) \\
&= 2\lambda + \frac{2m_+}{z^* m} - \frac{2m_+}{z^* m} \sum_{i: y_i = +1} x_{i,j} + \frac{2z^* m_-}{m} \sum_{i: y_i = -1} x_{i,j} \\
&= 2\sqrt{PD(1 - PD)} + 2\lambda \\
&\quad - 2\sqrt{PD(1 - PD)} \left(\sum_{i: y_i = +1} x_{i,j} - \sum_{i: y_i = -1} x_{i,j} \right) \\
&< 2\sqrt{PD(1 - PD)} = f(0, -1, z) = f(0, 1, z)
\end{aligned} \tag{6.16}$$

by choosing λ such that

$$0 < \lambda < \sqrt{PD(1 - PD)} \left(\sum_{i: y_i = +1} x_{i,j} - \sum_{i: y_i = -1} x_{i,j} \right) \tag{6.17}$$

This is possible since $0 < PD < 1$ and because of the total sum-condition of $x_{i,j}$. This proves that there is a non-trivial minimum of the function $(w, b) \mapsto f(w, b, z)$ for $z = z^*$, $w_i^* = 2$ and $b = -1$. By the existence of some indices i, i' for which $x_{i,j} < 0.5$ and $x_{i',j} > 0.5$, we also obtain $G_1, G_2 \neq \emptyset$. $\square$

Theorem 6.1 is nice in the sense that our weighted SVM is guaranteed to have a nontrivial division of the data into two groups for $z^* = \sqrt{\frac{1-PD}{PD}}$. One would assume that $PD_1 > PD_2$ by the definition of G_1 and G_2. However, a very simple example shows us that this is not the case.

Example 6.1. *Consider the data set* $\{(0,1),(0.95,1),(1,1),(0.55,-1),(0.6,-1)\}$, *where the final coordinate as usual is the positive or negative label. Clearly* $PD = 0.4$ *and we have*

$$\sum_{i:y_i=-1} x_i = 1.15 < 1.95 = \frac{1}{m_+} \sum_{i:y_i=+1} x_i \tag{6.18}$$

Therefore, all conditions of Theorem 6.1 are satisfied and we have for $w^* = 2$ *and* $b = -1$

$$G_1 = \{(0,1)\} \tag{6.19}$$
$$G_2 = \{(0.95,1),(1,1),(0.55,-1),(0.6,-1)\} \tag{6.20}$$

Therefore the PD of the first set equals 0, *whereas the PD of the second set equals* 0.5.

Nonetheless, if the conditions of Theorem 6.1 hold, we can find a w^* and a b^* such that $PD_1 < PD_2$.

Theorem 6.2. *Let the conditions of Theorem 6.1 hold. Then there exist* $w^* \in \mathbb{R}^n$, $b^* \in \mathbb{R}$ *such that for* $z^* = \sqrt{\frac{1-PD}{PD}}$ *we have* $G_1, G_2 \neq \emptyset$ *and* $PD_1 > PD_2$.

Proof. We use the same assumptions and notational convenience as in Theorem 6.1, preceding Equation 6.16.

Now assume to the contrary that there does not exist a $w^* > 1$ and $b^* = -1$ for which $G_1, G_2 \neq \emptyset$ and $PD_1 < PD_2$. Sort the data points $\{(x_i, y_i)\}$ on the value of the coordinate $x_{i,j}$, such that $x_{i,j} \leq x_{i',j}$ for $i < i'$ and if $x_{i,j} = x_{i',j}$ we have $y_i \geq y_i'$, i.e. in the case of a tie the elements of the positive class come first in the sorted list. We still denote the sorted data points by $\{(x_i, y_i)\}_{i=1}^{m}$. Also let $\{(\xi_i^-, y_i)\}_{i=1}^{m_-}$ and $\{(\xi_i^+, y_i)\}_{i=1}^{m_+}$ be a sorted list of the negatively and positively classified data points respectively.

Take some $w^* > 1$, then $G_1 = \{(x_i, y_i) : x_{i,j} < \frac{1}{w^*}\}$. For some $w^* \in [1, \infty)$ this set and G_2 are non-empty, for example for $w^* = 2$. Denote $W = \{w^* \in [1,\infty) : G_1 \neq \emptyset \neq G_2\}$. For every $w^* \in W$ we have $PD_1 \geq PD_2$, i.e. $PD_1 \geq PD$. By definition we then have

$$|G_1| \geq \frac{1}{PD} \sum_{i:(x_i,y_i)\in G_1} \mathbb{1}\{y_i = -1\} \tag{6.21}$$

Define $s = \frac{1}{PD}$. Observe that, since in case of a tie the positively classified points are listed first, at least the first $\lceil s \rceil - 1$ elements of the sorted list must have $y_i = +1$, otherwise Equation 6.21 does not hold for the set G_1 containing all elements of the list up to and including the first negatively classified point. Therefore the first $\lceil s \rceil - 1$ elements of the sorted list have an average value for $x_{i,j}$ that is at most the value of the jth coordinate of the first negatively classified data point, i.e.

$$\frac{1}{\lceil s \rceil - 1} \sum_{i=1}^{\lceil s \rceil - 1} \xi_{i,j}^+ \leq \xi_{1,j}^- \tag{6.22}$$

Moreover, using the same argument, if we have k negatively classified data points in G_1 we must have at least $\lceil ks \rceil - k$ positively classified data points in G_1 to let Equation 6.21 be satisfied. By induction we see that for $K \leq m_-$

$$\sum_{k=1}^{K} \frac{1}{\omega_k} \sum_{i=\lceil (k-1)s \rceil - (k-2)}^{\lceil ks \rceil - k} \xi_{i,j}^+ \leq \sum_{k=1}^{K} \xi_{k,j}^- \tag{6.23}$$

where the coefficient ω_k is defined as

$$\begin{aligned} \omega_k &= (\lceil ks \rceil - k) - (\lceil (k-1)s \rceil - (k-1)) = (\lceil ks \rceil - \lceil (k-1)s \rceil) - 1 \\ &= \begin{cases} \lceil s \rceil - 1 & \text{if } (k-1)s \in \mathbb{Z} \\ \lceil s \rceil - 2 & \text{if } (k-1)s \notin \mathbb{Z} \end{cases} \end{aligned} \tag{6.24}$$

To obtain a contradiction with the average-value condition, we want to obtain

$$\frac{1}{m_-} \sum_{k=1}^{m_-} \xi_{k,j}^- \geq \frac{1}{m_+} \sum_{i=1}^{m_+} \xi_{i,j}^+ \tag{6.25}$$

Observe that $\lceil m_- \cdot s \rceil - m_- = m_+$, since $\lceil m_- \cdot s \rceil - m_- \neq m_+$ would imply that $m = m_+ + m_- \neq \lceil \frac{m_-}{PD} \rceil = \lceil m \rceil = m$, which leads to a contradiction. Therefore we can simply take $K = m_-$ in Equation 6.23 and divide by m_- to obtain the desired inequality.

The only problem we have to deal with now are the coefficients $\frac{1}{\omega_k}$. Write $s = \frac{p}{q}$ where p and q do not have a common prime factor. Then clearly $(k-1)s \in \mathbb{Z}$ if and only if $(k-1) \mid q$. First assume $q = 1$. Then $s \in \mathbb{Z}$ so $\omega_k = s - 1$ for all k. If we insert this into Equation 6.23 and divide both sides by $m_- = (1 - PD)m = \frac{s-1}{s} m = (s-1)m_+$ and we directly obtain Equation 6.25. For $q \geq 2$ we let $r = q - p (\text{mod } q)$, such that $\lceil s \rceil - s = \frac{r}{q}$. Now we obtain the following series of (in)equalities.

$$\begin{aligned} \sum_{i=1}^{q} \xi_{i,j}^- &\geq \sum_{k=1}^{q} \frac{1}{\omega_k} \sum_{i=\lceil (k-1)s \rceil - (k-2)}^{\lceil ks \rceil - k} \xi_{i,j}^+ \\ &= \frac{1}{\lceil s \rceil - 1} \sum_{i=1}^{\lceil s \rceil - 1} \xi_{i,j}^+ + \frac{1}{\lceil s \rceil - 2} \sum_{i=\lceil s \rceil}^{qs-q} \xi_{i,j}^+ \\ &= \frac{1}{s-1} \sum_{i=1}^{q(s-1)} \xi_{i,j}^+ + \left(\frac{1}{\lceil s \rceil - 1} - \frac{1}{s-1} \right) \sum_{i=1}^{\lceil s \rceil - 1} \xi_{i,j}^+ \\ &\quad + \left(\frac{1}{\lceil s \rceil - 2} - \frac{1}{s-1} \right) \sum_{i=\lceil s \rceil}^{q(s-1)} \xi_{i,j}^+ \\ &= \frac{1}{s-1} \sum_{i=1}^{q(s-1)} \xi_{i,j}^+ + \frac{s - \lceil s \rceil}{s-1} \frac{1}{\lceil s \rceil - 1} \sum_{i=1}^{\lceil s \rceil - 1} \xi_{i,j}^+ \\ &\quad + \frac{s - \lceil s \rceil + 1}{s-1} \frac{1}{\lceil s \rceil - 2} \sum_{i=\lceil s \rceil}^{q(s-1)} \xi_{i,j}^+ \end{aligned}$$

$$\begin{aligned}
\sum_{i=1}^{q} \xi^-_{i,j} &\geq \frac{1}{s-1} \sum_{i=1}^{q(s-1)} \xi^+_{i,j} - \frac{r}{q(s-1)} \frac{1}{\lceil s \rceil - 1} (\lceil s \rceil - 1) \xi^+_{\lceil s \rceil, j} \\
&+ \frac{1-\frac{r}{q}}{s-1} \frac{1}{\lceil s \rceil - 2} (\lceil s \rceil - 2)(q-1) \xi^+_{\lceil s \rceil, j} \\
&= \frac{1}{s-1} \sum_{i=1}^{q(s-1)} \xi^+_{i,j} + \frac{1}{q(s-1)} ((q-1)(q-r)\xi^+_{\lceil s \rceil, j} - r\xi^+_{\lceil s \rceil, j}) \\
&\geq \frac{1}{s-1} \sum_{i=1}^{q(s-1)} \xi^+_{i,j} \qquad (6.26)
\end{aligned}$$

The second inequality comes from the fact that $\{\xi^+_{i,j}\}_{i=1}^{m_+}$ is a sorted list, so $\xi^+_{i,j} \leq \xi^+_{\lceil s \rceil, j}$ for all $i \leq \lceil s \rceil - 1$ and $\xi^+_{i,j} \geq \xi^+_{\lceil s \rceil, j}$ for all $i \geq \lceil s \rceil$. The third inequality comes from the fact that $q-1 \geq r$ and $q-r \geq 1$. Since $PD = \frac{1}{s} = \frac{q}{p}$ we have $m_- = m \cdot PD = q\frac{m}{p}$. Clearly $\frac{m}{p} \in \mathbb{Z}$ since p and q do not have a common factor. Therefore we obtain by induction from Equation 6.26

$$\begin{aligned}
\frac{1}{m_-} \sum_{i=1}^{m_-} \xi^-_{i,j} &\leq \frac{1}{m_-(s-1)} \sum_{i=1}^{m_+} \xi^+_{i,j} = \frac{PD}{m_-} \frac{1}{1-PD} \sum_{i=1}^{m_+} \xi^+_{i,j} \\
&= \frac{1}{m_+} \sum_{i=1}^{m_+} \xi^+_{i,j} \qquad (6.27)
\end{aligned}$$

This closes the proof of the theorem. □

The ideal situation would be that we can calculate a z^*, so essentially a PD, such that we can control either the sizes of the sets G_1 and G_2 or the values of PD_1 and PD_2. However, since these depend entirely on the internal structure of the data, such as the clustering of points, we cannot give general results on these matters. Our solution for this problem is to split your training set into two different sets, a building set and a validation set. Using the building set we develop different models with different values for z^*, where the validation set chooses the model that obtains the value for G_1 (or PD_1 etc.) that is closest to our desired value.

The idea of a weighted SVM is not entirely new, but also proposed and investigated in some other articles (Yang, 2005 & Huang, 2005 & Qiao and Zhang, 2015 & Lapin, Hein, and Schiele, 2014). The focus of these articles is mostly to put weights on those observations that are labeled as more important, or more significant for the prediction. In some sense this model also acts the same way, i.e. the (rare) defaulted loans are regarded as more important than the non-defaulted loans, if we set z high enough.

6.2 Convergence of averaged SGD

To obtain a solution for the weighted SVM problem we use the *averaged stochastic gradient descent*. We perform an averaging over all or a subset of the iterations of w. This is needed because of the use of the weighted SVM, as a sequence of negatively labeled examples (with large weight) at the end of the development can lead to a huge deviation in the coordinates of w. This causes the SVM to predict that all examples are negatively labeled. By using an averaging scheme (with the right weights), this unfavorable event is ruled out.

As is done in most articles about SGD, we slightly modify our objective function of Equation 6.2, to remove the parts containing a b from the equation. Define $\phi' : \mathbb{R}^n \mapsto \mathbb{R}^{d+1}$ for any $\phi : \mathbb{R}^n \mapsto \mathbb{R}^d$ as $\phi'(x) = (\phi(x), 1)$. Also define $w_b \in \mathbb{R}^{d+1}$ as $w_b = (w, b)$. Then we obtain

$$w^T \phi(x_i) + b = w_b^T \phi'(x_i) \tag{6.28}$$

As we see, for m large and λ small (a value of 10^{-5} or 10^{-6} is often used) the second term of Equation 6.2 will dominate the expression, even for large values of w and b, since the second term is an average over a set of positive values. Therefore, replacing $w^T w$ with $w_b^T w_b$ in the second term will not do much harm, as they only differ a term $b^T b$ from each other. Using this we obtain a new objective function (where we renamed w_b to w and ϕ' to ϕ)

$$f(w, b) = \frac{\lambda}{2} w^T w + \frac{1}{m} \sum_{i=1}^{m} \left(\frac{1}{z}\right)^{y_i} \max\{0, 1 - y_i w^T \phi(x_i)\} \tag{6.29}$$

There are several types of averaging schemes proposed in Shamir and Zhang, 2013. At first we can look at the *α-suffix averaging*, which essentially takes the average value of w over the last αT iterates, where $\alpha \in (0, 1)$, i.e.

$$\bar{w}_T^{\alpha} = \frac{1}{\alpha T} \sum_{t=(1-\alpha)T+1}^{T} w_t \tag{6.30}$$

However, this is not pleasant if we do not know the total number of iterates T in advance, as we have to store all values of w in our memory. Therefore we could also choose to average all iterates *on-the-fly*, such that for all t

$$\bar{w}_t = \left(1 - \frac{1}{t}\right) \bar{w}_{t-1} + \frac{1}{t} w_t \tag{6.31}$$

We essentially compute the average of all iterates of w. This is proven to be suboptimal in Rakhlin and Shamir, 2012, but at least it is easy to compute.

We now analyze a new averaging scheme called *exponential-decay averaging*, which puts more weight on recent observations, but can also be computed on-the-fly, without having to store the values of w. For a given $\alpha \geq 0$ we define

$$\bar{w}_T = \frac{\sum_{t=0}^{T-1} \alpha^t w_{T-t}}{\sum_{t=0}^{T-1} \alpha^t} = \frac{1}{C} \left(w_T + \alpha w_{T-1} + \ldots + \alpha^{T-1} w_1\right) \tag{6.32}$$

If $\alpha \to 0$, then $\bar{w}_T \to w_T$, i.e. only the final observation of w is taken int account. If $\alpha = 1$, we see that we exactly obtain Equation 6.31 again.

It is interesting if this averaging scheme leads to a converging result, at least for λ-strongly convex functions. Observe that our objective function f given by Equation 6.29 is λ-strongly convex, according to the strongly convex definition for non-differentiable convex function in Equation 5.25 (see Rakhlin and Shamir, 2012). Also observe that indeed $\bar{w}_T$ can be computed only using $\bar{w}_{T-1}$ and w_T, since

$$\begin{aligned}\bar{w}_T &= \frac{\sum_{t=0}^{T-1} \alpha^t w_{T-t}}{\sum_{t=0}^{T-1} \alpha^t} = \frac{\alpha \sum_{t=0}^{T-2} \alpha^t w_{(T-1)-t} + w_T}{\sum_{t=0}^{T-1} \alpha^t} = \frac{\alpha \bar{w}_{T-1} \sum_{t=0}^{T-2} \alpha^t + w_T}{\sum_{t=0}^{T-1} \alpha^t} \\ &= \frac{\sum_{t=0}^{T-2} \alpha^t}{\sum_{t=0}^{T-1} \alpha^t} \alpha \bar{w}_{T-1} + \frac{1}{\sum_{t=0}^{T-1} \alpha^t} w^T = \alpha \frac{1-\alpha^{T-1}}{1-\alpha^T} \bar{w}_{T-1} + \frac{1-\alpha}{1-\alpha^T} w_T \end{aligned} \tag{6.33}$$

using the property of the geometric sum that $\sum_{i=0}^{n-1} r^i = (1-r^n)/(1-r)$ for $r \neq 1$. The following theorem now gives us a convergence rate for our objective function f using this exponential averaging scheme. The proof of the next theorem is partially based on the proof of Theorem 4 in Shamir and Zhang, 2013.

Theorem 6.3. *Let f be λ-strongly convex and assume $\mathbb{E}[\|g_t\|_2] \leq \beta T^c$ for some $\beta \geq 0$ and $c < 1$. Consider SGD with step size $\eta_t = 1/(\lambda t)$. Let $0 \leq \alpha < 1$ be the exponential averaging factor. Then we obtain an $\mathcal{O}(\log(T)/T^{1-c})$ convergence rate for $\mathbb{E}[f(\bar{w}_t) - f(w^*)]$.*

Proof. Observe that for any $T \geq 1$ we can write, using the properties of a geometric sum

$$\bar{w}_T = \sum_{t=0}^{T-1} \alpha^k \frac{1-\alpha}{1-\alpha^T} w_{T-t} \tag{6.34}$$

Denote the coefficient in front of w_{T-t} by a_t. Note that we have $\sum_{t=0}^{T-1} a_t = 1$ using the definition of $\bar{w}_t$ in Equation 6.32. Define $f'(w) = \mathbb{E}[f(w) - f(w^*)]$ where w^* again is the optimal value for w. Since f is a convex function we can use Jensen's inequality to obtain

$$f'(\bar{w}_T) = \mathbb{E}[f(\bar{w}_T) - f(w^*)] \leq \sum_{t=0}^{T-1} \mathbb{E}[a_t f(w_{T-t}) - f(w^*)] = \sum_{t=0}^{T-1} a_t f'(w_{T-t}) \tag{6.35}$$

Define S'_k as the unweighted sum of the function f' over the last $k+1$ iterates, i.e.

$$S'_k = \sum_{t=T-k}^{T} f'(w_t) \tag{6.36}$$

Combining Equations 6.35 and 6.36 we see

$$\begin{aligned} f'(\bar{w}_t) &\leq a_{T-1} S'_{T-1} + (a_{T-2} - a_{T-1}) S'_{T-2} + \ldots + (a_0 - a_1) S'_0 \\ &= \sum_{t=0}^{T-1} (a_t - a_{t+1}) S'_t + a_T S'_{T-1} \end{aligned} \tag{6.37}$$

In Theorem 3 of Shamir and Zhang, 2013 there is an upper bound presented on S'_k, given by

$$\begin{aligned} \frac{1}{k+1} S'_k &\leq \frac{17 \beta T^c \left(1 + \log(\frac{1}{\min\{(k+1)/T, (1+1/T)-(k+1)/T\}})\right)}{\lambda T} \\ &= \frac{17 \beta \log(\frac{Te}{\min\{k+1, T-k\}})}{\lambda T^{1-c}} \end{aligned} \tag{6.38}$$

Observe that $a_t - a_{t+1} = \frac{\alpha^t}{1-\alpha^T}(1-\alpha)^2$ and therefore

$$\begin{aligned}
f'(\bar{w}_t) &\leq \frac{17\beta(1-\alpha)^2}{\lambda T^{1-c}(1-\alpha^T)}\Big(\sum_{t=0}^{\lfloor T/2 \rfloor} \alpha^t (t+1) \log\left(\frac{Te}{t+1}\right) \\
&\quad + \sum_{t=\lceil T/2 \rceil}^{T-1} \alpha^t (t+1) \log\left(\frac{Te}{T-t}\right)\Big) + \frac{17\beta \log(Te)}{\lambda T^{1-c}} \frac{\alpha^T(1-\alpha)}{1-\alpha^T} \\
&\leq \frac{17\beta(1-\alpha)^2}{\lambda T^{1-c}(1-\alpha^T)} \sum_{t=0}^{T-1} \alpha^t (t+1) \log(Te) \\
&\quad + \frac{17\beta \log(Te)}{\lambda T^{1-c}} \frac{\alpha^T(1-\alpha)}{1-\alpha^T}
\end{aligned} \tag{6.39}$$

We now try to find a bound on the sum in the first term of Equation 6.39. Observe that for any $0 < \alpha < 1$ we see that $\lim_{t\to\infty} \alpha^t(t+1) = 0$. Also the maximum of $g(t) := \alpha^t(t+1)$ is attained at $t^* = -\frac{1}{\log(\alpha)} - 1$, for which we have

$$g(t^*) = \alpha^{t^*}(t^*+1) = -\frac{1}{\alpha^{\frac{1}{\log(\alpha)}+1}\log(\alpha)} > 0 \tag{6.40}$$

since $\alpha < 1$ so $\log(\alpha) < 0$. Hence we see that for $0 \leq t \leq t^*$ the function g is monotone increasing and for $t \geq t^*$ the function g is decreasing. Now consider two cases: $\alpha \leq \frac{1}{e}$ and $\alpha > \frac{1}{e}$. In the first case, we see that $t^* \leq 0$, so g is monotone decreasing on the entire interval $[0, \infty)$, as we see in Figure 6.1 below.

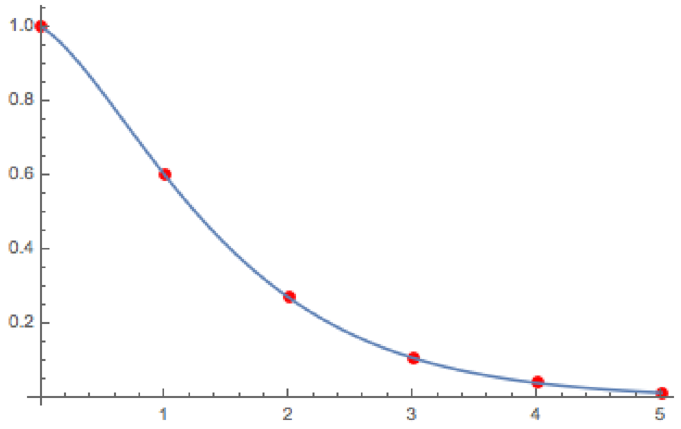

FIGURE 6.1: Plot of g(t) for $\alpha = 0.3$

Therefore we can upper bound the sum of g as follows

$$\begin{aligned}
\sum_{t=0}^{T-1} g(t) \leq g(0) + \int_0^{T-1} g(t)dt &= 1 + \left[\frac{\alpha^t((t+1)\log(\alpha) - 1))}{\log^2(\alpha)}\right]_{t=0}^{T-2} \\
&= 1 + \frac{\alpha^{T-2}(T\log(\alpha) - 1)}{\log^2(\alpha)} - \frac{\log(\alpha) - 1}{\log^2(\alpha)} \\
&= C_\alpha + \alpha^{T-2}\frac{T\log(\alpha) - 1}{\log^2(\alpha)}
\end{aligned} \tag{6.41}$$

If we plug the estimate from Equation 6.41 into Equation 6.39 we obtain

$$\begin{aligned} f'(\bar{w}_t) &\leq \frac{17\beta(1-\alpha)^2\log(Te)}{\lambda T^{1-c}(1-\alpha^T)}\left(C_\alpha + \alpha^{T-2}\frac{T\log(\alpha)-1}{\log^2(\alpha)}\right) + \frac{17\beta\log(Te)}{\lambda T^{1-c}}\frac{\alpha^T(1-\alpha)}{1-\alpha^T} \\ &= \frac{17\beta(1-\alpha)\log(Te)}{\lambda T^{1-c}(1-\alpha^T)}\left((1-\alpha)C_\alpha + (1-\alpha)\alpha^{T-2}\frac{T\log(\alpha)-1}{\log^2(\alpha)} + \alpha^T\right) \quad (6.42) \\ &= \frac{17\beta(1-\alpha)(1+\log(T))}{\lambda T^{1-c}(1-\alpha^T)}\left((1-\alpha)C_\alpha + \mathcal{O}(T\alpha^{T-2})\right) \quad (6.43) \end{aligned}$$

so we obtain a convergence rate of $\mathcal{O}(\log(T)/T^{1-c})$, just as in the case without averaging, since the $T\alpha^{T-2}$-term will tend to go much faster to zero as $T \to \infty$.

Now assume that $\alpha > \frac{1}{e}$, so g attains its maximum $g(t^*)$ on the interval $[0, \infty)$, as we see in Figure 6.2 below.

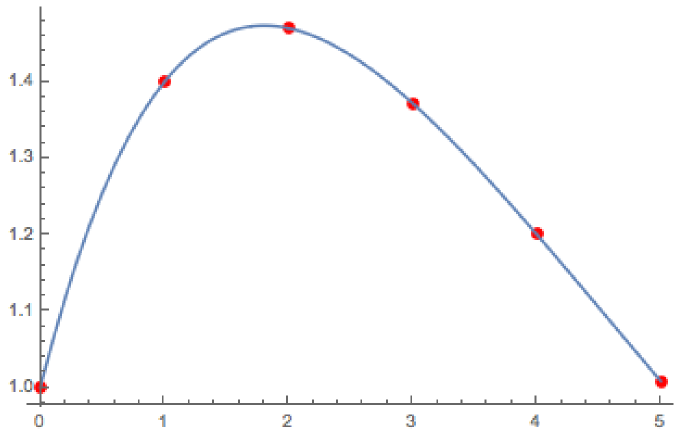

FIGURE 6.2: Plot of g(t) for $\alpha = 0.7$

We therefore split the sum in Equation 6.39 in three parts, i.e.

$$\sum_{t=0}^{T-1} g(t) = \sum_{t=0}^{\lfloor t^*\rfloor - 1} g(t) + g(\lfloor t^*\rfloor) + \sum_{t=\lceil t^*\rceil}^{T-1} g(t) \quad (6.44)$$

Observe that in the case that $t^* \geq T$ (for α very close to 1), only the first sum remains. The first sum of 6.44 is a sum on the interval where g is monotone increasing, hence we can find an upper bound

$$\sum_{t=0}^{\lfloor t^*\rfloor - 1} g(t) \leq \int_1^{\lfloor t^*\rfloor} g(t)dt \quad (6.45)$$

The second sum of 6.45 is a sum on the interval where g is monotone decreasing, hence we can again find an upper bound as in Equation 6.41

$$\sum_{t=\lceil t^*\rceil}^{T-1} g(t) = g(\lceil t^*\rceil) + \int_{\lceil t^*\rceil}^{T-2} g(t)dt \quad (6.46)$$

Combining Equations 6.44, 6.45 and 6.46 we see

$$\begin{aligned}\sum_{t=0}^{T-1} g(t) &\leq \int_{1}^{\lfloor t^* \rfloor} g(t)dt + \int_{\lceil t^* \rceil}^{T-2} g(t)dt + g(\lfloor t^* \rfloor) + g(\lceil t^* \rceil) \\ &\leq \int_{0}^{T-2} g(t)dt + g(t^*) \end{aligned} \tag{6.47}$$

as $\min\{g(\lfloor t^* \rfloor), g(\lceil t^* \rceil)\} = \min_{\lfloor t^* \rfloor \leq t \leq \lceil t^* \rceil} g(t) \leq \int_{\lfloor t^* \rfloor}^{\lceil t^* \rceil} g(t)dt$. From Equation 6.47 we obtain the following upper bound as before

$$\begin{aligned}\sum_{t=0}^{T-1} g(t) &\leq -\frac{1}{\alpha^{\frac{1}{\log(\alpha)}+1}\log(\alpha)} + \frac{\alpha^{T-2}(T\log(\alpha)-1)}{\log^2(\alpha)} - \frac{\log(\alpha)-1}{\log^2(\alpha)} \\ &= C_\alpha + \alpha^{T-2}\frac{T\log(\alpha)-1}{\log^2(\alpha)} \end{aligned} \tag{6.48}$$

Therefore we obtain Equation 6.43, also for $\alpha > \frac{1}{e}$. Summarizing, we obtain the following bound for all $0 < \alpha < 1$

$$f'(\bar{w}_t) \leq \frac{17\beta(1-\alpha)(1+\log(T))}{\lambda T^{1-c}(1-\alpha^T)} \left((1-\alpha)C_\alpha + T\alpha^{T-2}D_\alpha\right) \tag{6.49}$$

where we define

$$C_\alpha = \begin{cases} 1 + \frac{1-\log(\alpha)}{\log^2 \alpha} & \text{if } \alpha \leq \frac{1}{e} \\ -\frac{1}{\alpha^{\frac{1}{\log(\alpha)}+1}\log(\alpha)} + \frac{1-\log(\alpha)}{\log^2 \alpha} & \text{if } \alpha > \frac{1}{e} \end{cases} \tag{6.50}$$

$$D_\alpha = \frac{(1-\alpha)\log(\alpha)-1}{\log^2(\alpha)} + 1 \tag{6.51}$$

The last term in the definition of D_α is added to also include the single α^T of Equation 6.42 in the $T\alpha^{T-2}$-term, which dominates the α^T already if $T = 1$. This closes the proof of the theorem. □

Therefore, it is indeed not unfavorable to use the exponential averaging scheme instead of the last iterate w_T, looking at the performance in terms of the convergence rate.

The uniform bound on the subgradients may not seem realistic, but choosing a large β and c close to one, this should not be a problem, as we initialize our model development with $w = 0$, so that the gradients are not that large.

For $\beta = \lambda = 1$ and $\alpha = 0.5$ we see in Figure 6.3 on the next page the convergence rates for $c = 0$ (red), $c = 0.25$ (blue) and $c = 0.5$ (green), with the number of iterations on the x-axis. Obviously, as c increases the amount of time to reach some specified level of expected loss is larger.

Finally, there are some averaging schemes for the stochastic gradient descent that have a better convergence rate of $\mathcal{O}(1/T^{1-c})$, such as the polynomial averaging scheme proposed in Shamir and Zhang, 2013. However, we see for $T < 10^7$, there is no extreme difference between this scheme and our proposed exponential averaging scheme, as the polynomial averaging scheme is at most twice as 'fast' as our averaging scheme.

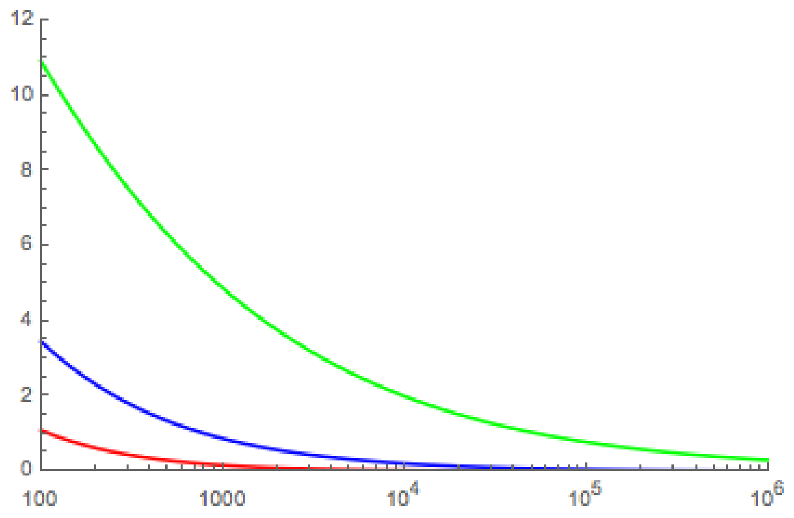

FIGURE 6.3: Convergence rates of exponential decay averaging scheme for different values of c

6.3 Summary

We summarize our weighted SVM combined with the averaged SGD in the next scheme. For simplicity of the algorithm we set $\nabla_{(i)} f(w_k) = \nabla f_0(w_k) + 0$ if $y_i w^T \phi(x_i) = 1$, i.e. where the graph of the hinge loss function is not differentiable, since already $\nabla_{(i)} f(w_k) = \nabla f_0(w_k)$ if $y_i w^T \phi(x_i) > 1$. As we removed b from the initial objective function, such as described in Section 6.2, we at least do the mapping $\phi(x_i) = (x_i, 1)$. Also, more complex feature mappings can be used that have a value of 1 for the final coordinate, e.g. if we are looking at a polynomial kernel.

Algorithm 6.1 Weighted SVM combined with averaged SGD

```
1: Start with $w_0 = 0$, specify a number of iterations $T$, a regularization parameter $\lambda$, a weight $z$, an exponential averaging factor $\alpha$ and set $\gamma_k = \frac{1}{\lambda k}$.
2: Modify the data points $x_i$ using a specified feature mapping $\phi$.
3: Define $f(w) = w^T w + \frac{1}{m} \sum_{i=1}^{m} \left(\frac{1}{z}\right)^{y_i} \max\{0, 1 - y_i w^T \phi(x_i)\}$.
4: for $0 \leq k \leq T - 1$ do
5:     Take $i$ between 1 and $m$ randomly.
6:     Compute the individual gradient $\nabla_{(i)} f(w_k)$.
7:     Update $w_{k+1} = w_k - \gamma_{k+1} \nabla_{(i)} f(w_k)$.
8:     Set $w_{k+1} = \alpha \frac{1-\alpha^k}{1-\alpha^{k+1}} w_k + \frac{1-\alpha}{1-\alpha^{k+1}} w_{k+1}$ and $k = k + 1$. (averaging step)
9: end for
10: return $w_T$
```

We will use this scheme as a basis for the model development described in the next chapter.

Chapter 7

Model development, results and testing

This is the main chapter of this thesis, as it describes how a model development takes place. Furthermore, we will look at the results that we obtain using a model. The chapter ends with a model performance test and a comparison with the results obtained by the FICO credit scoring model and a linear regression model based on these credit scores.

7.1 Model development

The model is developed as follows. First we potentially modify the data using a feature mapping ϕ. Then we define some specified number of iterations T, usually around 1,000,000. We take a regularization parameter λ of around 10^{-5} or 10^{-6} and an exponential averaging factor α between 0 and 1. The learning rate γ_k equals $\frac{1}{\lambda k}$. The next step is to define a weight set Z that contains several weights z, which are usually powers of 2, 4 or 8, for example all powers of 4 between 1/16 and 16384, i.e.

$$Z = \{16384, 4096, 1024, 256, 64, 16, 4, 1, 1/4, 1/16\} \tag{7.1}$$

Now we split the training set into three different parts. With the first part we assign values to the nominal variables. The second part is used to determine which variables are most significant. Then we perform the weighted SVM with averaged stochastic gradient descent as described in Algorithm 6.1 for all weights in the weight set z, using the first and the second part of the training set, which we call the *building set*. We now obtain for each weight z a division of the data into two groups G_1 and G_2 with two default probabilities PD_1 and PD_2. Generally, $PD_1 < PD_2$ and G_1 is a smaller set if the weight z is larger. For each weight z we also obtain the coefficients of w, the hyperplane that is separating the two groups.

Finally using the third part, called the *validation set*, we determine which weight is the best suitable for our needs. We can choose the weight that, for the validation set, splits the data into a group G_1 containing approximately 20% of the data and group G_2 containing 80%. Another option is to choose the weight that splits the data into a group G_1 with a PD_1 close to some specified value and another group G_2. However, these methods require human judgment to set the goal values. The approach that is used in this thesis takes the weight for which G_1 is non-empty and has the smallest PD. This will be class number 1.

In the case that we want to have $d > 2$ groups, we then remove the observations (of the validation set) in G_1 from our model. Then we look at all weights $z \in Z$ smaller than this specified weight, since all sets G_1 for a larger weight are generally smaller than the set with this specified weight. Again we choose the weight for which the set

G_1 (without these excluded observations) that is non-empty and has the smallest PD. We continue to do this until we have obtained $d-1$ groups. The remaining observations are in the final group, that generally has the largest probability of default. Clearly we should have $|Z| \geq d-1$, as we are taking a smaller weight in each step.

Usually the split of the training set is 10% - 60% - 30%. A schematic overview of this split can be found in Figure 7.1 below.

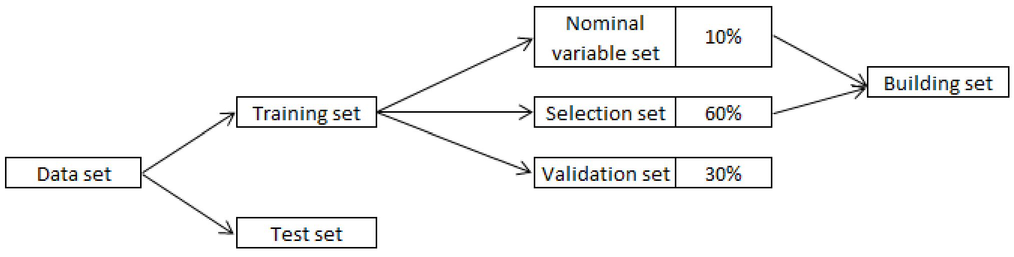

FIGURE 7.1: Schematic overview of the splitting of the data set

As we have now obtained d groups with usually an increasing probability of default on the validation set, we can apply the model with these groups, which are identified using the different created hyperplanes (so, by the values of w obtained from the weighted SVM), to our test set, to obtain also a classification in d groups there.

7.2 Performance measure

The accuracy of the developed models is tested in three ways. At first we look at the correctness of the ranking, i.e. if for the test set we obtain also a ranking of groups with an increasing probability of default. Furthermore we assess the discriminatory power of the model using a modified version of a *AUC (area-under-curve)* test, i.e. how good is this model in separating the 'good' loans from the 'bad' loans. Finally, we check if the predicted probabilities of default are in the order of the actual obtained probabilities of default. We discuss these testing methods further in this section.

7.2.1 Correctness of ranking

From our weighted SVM model we obtain a ranking into d different groups. The ranking of these groups are based on the PDs of the validation set. For the test set we observe if for each pair of successive groups indeed the group with the smaller number has a smaller PD. After these $d-1$ comparisons, we can conclude if the ranking is either *correct* or *incorrect*.

7.2.2 AUC technique

The AUC test is based on so called *ROC (receiver operator characteristics) curves*. It essentially measures the performance of binary classification functions. These are functions that classify elements as positive or negative, just as the standard SVM. For example it decides if a patient is predicted to have a certain disease or not. If an element is classified into the positive class and indeed belongs to the positive class, we call it a *true positive*. If instead the element belongs to the negative class, we call it a *false positive*. In the same way, an element from the positive class that is classified as negative is called a *false negative* and a *true negative* is an element classified as negative from the negative class. This is summarized in Table 7.1 below.

		Predicted class	
		Positive	**Negative**
True class	**Positive**	True positive	False negative
	Negative	False positive	True negative

TABLE 7.1: Contingency table of binary classification

Two important ratios are the *true positive rate (TPR)*, which is the ratio of the true positives and the total number of positive elements, and the *false positive rate (FPR)*, which is the ratio of the false positives and the total number of negative elements. The idea is to obtain as much true positives and true negatives as possible. In our case with mortgage loans, we would like to have as much defaults as possible in the last group(s). If we have different test sets for the binary classifier we can create an ROC curve, which essentially connects the points (x_i, y_i) where x_i and y_i are the FPR and the TPR of test set i respectively. The ideal situation would be that x_i is low when y_i is already high. Both x_i and y_i can attain values between 0 and 1. Obviously, if all observations are classified as negative we obtain $x_i = y_i = 0$. Likewise, if all observations are classified as positive we obtain $x_i = y_i = 1$. The linear line connecting $(0, 0)$ and $(1, 1)$ corresponds to a random guess, i.e. if we have a model where each element is classified as positive with probability equal to the relative frequency of positive elements in the data set and likewise for the negative elements. A model is performing better if it is farther above this diagonal line. The performance of models is then compared using the *area under curve (AUC)*. This means that the area under the constructed curves is computed, assuming that the curves start in $(0, 0)$ and end in $(1, 1)$. The AUC of the diagonal 'guessing' line equals 0.5. The model with the perfect discriminatory power has a FPR of 0 and a TPR of 1, and therefore it has an AUC of 1. In Figure 7.2 on the next page we see a few ROC curves.

Now we switch to our case with multiple classes. For each model with d groups we can create a curve with for every group a point (x_i, y_i) where x_i is the ratio of the (cumulative) number of defaults in group 1 up to i and the total number of defaults in the test set (FPR), and y_i is the ratio of the (cumulative) number of non-defaulted loans in group 1 up to i and the total number of non-defaulted loans in the test set (TPR). In this way we create a sequence $\{(x_i, y_i)\}_{i=0}^{n}$ where $(x_0, y_0) = (0, 0)$ and the sequence is increasing in both variables. In this way we create a curve just as in the situation above, for which we can compute the AUC. This is called a *cumulative accuracy profile (CAP) curve*.

There is no universal consensus on which AUC values are precisely classified as good or bad, since the (maixmum) performance of models also highly depends on the structure of the data. For (linearly) separable data, it is possible to construct an ideal model with an AUC of 1. However, for less structured data, an AUC of 0.7 can already be considered as good. For our case we consider an AUC of 0.8 or higher as *good*, an AUC of 0.7 or higher as *moderate* and an AUC of smaller than 0.7 as *bad*.

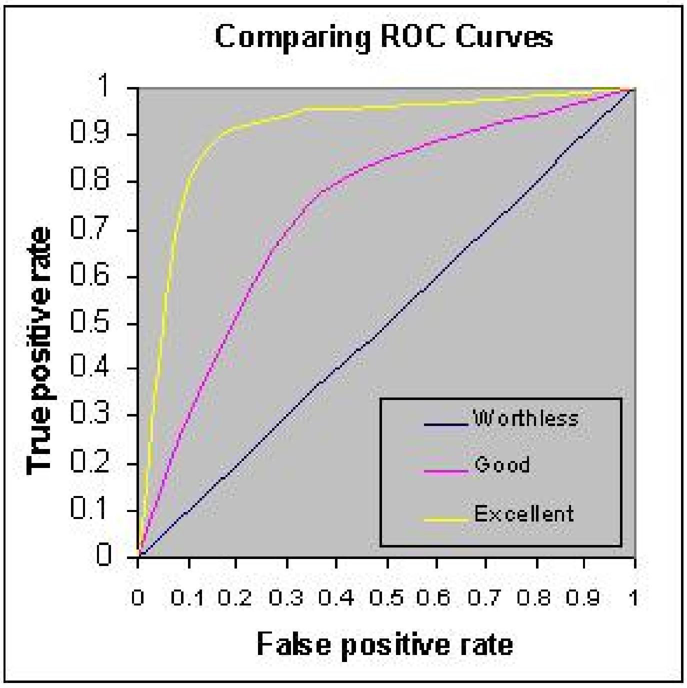

FIGURE 7.2: Different ROC curves

7.2.3 Predicted PDs

The predicted PDs are based on the number of defaults in the entire training set. We compare the predicted PD of a group with the PD of the test examples belonging to this group, based on the absolute and relative difference. The absolute difference of two PDs is simply calculated as the absolute value of the subtraction of the realized PD from the predicted PD. The relative difference of two PDs is calculated as the absolute difference divided by the maximum of the realized and the predicted PD. If both PDs are equal to zero, we also set the relative difference equal to zero. In short,

$$\text{absolute difference} = |\text{ predicted PD} - \text{realized PD }| \tag{7.2}$$

$$\text{relative difference} = \frac{\text{absolute difference}}{\max\{\text{predicted PD, realized PD}\}} \tag{7.3}$$

We call a predicted PD *good* if the absolute difference is smaller than 0.05% or the relative difference is smaller than 20%. It is ranked as *moderate* if the prediction is not good, but the absolute difference is smaller than 0.15% or the relative difference is smaller than 50%. If a prediction does not meet either of these conditions, it is classified as *bad*, as we see in the next table.

	Absolute difference		Relative difference
Good	< 0.05%	or	< 20%
Moderate	< 0.15%	or	< 50%
Bad	> 0.15%	and	> 50%

TABLE 7.2: Classification table for PDs

7.3 Results for a single model

We have applied our algorithm to several parts of the data set, first without monthly performance variables and macro-economic variables, to see how our model performs using only the origination file variables. This is the same situation as for the FICO credit scores. Secondly, we added the monthly performance variables and investigated how the model performs for different time periods between the origination date of the loan and the observation date for the performance variables. Finally we also added macro-economic variables to investigate the performance of the model using external data.

7.3.1 Without monthly performance variables

In this section we look at the results obtained with a training set containing all loans in the data sets of 2000Q1 until 2000Q4, and a test set containing some of the loans in the data sets of 2001Q1 until 2001Q4. We only use the original data set and look at the PD at a 1-, 3-, 5- and 10-year horizon. This results in the following classification of the 199,642 observations in our test set, given in Table 7.3-7.6.

1 year	# elements	# defaults	PD predicted	PD actual	Performance	TPR	FPR
Class 1	195843	46	0.011%	0.023%	Good	98.1%	74.2%
Class 2	2506	9	0.020%	0.359%	Bad	99.4%	88.7%
Class 3	1293	7	0.040%	0.541%	Bad	100.0%	100.0%

TABLE 7.3: 1-year PD prediction

Looking at the classification for the 1-year probability of default in Table 7.3, we see a classification in only three classes. The majority of the test set belongs to class 1 which has a relatively small actual PD of 0.023%. Nonetheless we filter already 2% of the 'bad' loans out, as we have a default rate that is 15-20 times as large in class 2 and class 3. This results in an AUC of 0.62. Furthermore we observe that the predicted PDs are really underestimating the actual PDs, which can be a serious problem.

3 year	# elements	# defaults	PD predicted	PD actual	Performance	TPR	FPR
Class 1	52925	54	0.043%	0.102%	Moderate	26.5%	6.9%
Class 2	122146	374	0.078%	0.307%	Bad	87.6%	55.0%
Class 3	17239	160	0.166%	0.928%	Bad	96.2%	75.6%
Class 4	7332	162	0.376%	2.209%	Bad	99.8%	96.4%
Class 5	478	28	2.032%	5.858%	Bad	100.0%	100.0%

TABLE 7.4: 3-year PD prediction

Looking at the classification for the 3-year probability of default in Table 7.4, we see a classification in five classes. The majority of the test set belongs to class 2, which has a PD of 0.307%. This is nice, as we see that we already obtain a class with less risk, containing over 25% of the data, and a default rate that is 3 times as small as the default rate of class 2. We also obtain three classes with the 'bad' performing loans, containing also 10% of the data. The AUC has increased to a value of 0.712. Again we observed that the predicted PDs are far too low.

5 year	# elements	# defaults	PD predicted	PD actual	Performance	TPR	FPR
Class 1	17154	9	0.083%	0.052%	Good	8.6%	0.7%
Class 2	102121	234	0.173%	0.229%	Moderate	60.0%	17.9%
Class 3	50740	433	0.206%	0.853%	Bad	85.4%	49.7%
Class 4	26794	533	0.554%	1.989%	Bad	98.6%	89.0%
Class 5	2833	150	2.658%	5.295%	Bad	100.0%	100.0%

TABLE 7.5: 5-year PD prediction

Looking at the classification for the 5-year probability of default in Table 7.5, we see again a classification in five classes. The majority of the test set still belongs to class 2, with an even smaller default rate compared to the default rate of class 2 in Table 7.4. This is caused by the migration of elements from the low-risk class 1 to class 2 and from class 2 to class 3, etc. Therefore all actual default rates in Table 7.5 are smaller than in Table 7.4, however the overall default rate should obviously be higher as we are considering a longer period. This is caused by the fact that the classes with a high PD contain many more elements. This has also caused the AUC to increase to 0.76. However, the predicted PDs are again too low compared to the actual PDs (expected for class 1).

10 year	# elements	# defaults	PD predicted	PD actual	Performance	TPR	FPR
Class 1	58607	59	0.068%	0.101%	Good	29.6%	2.9%
Class 2	100555	731	0.294%	0.727%	Bad	80.1%	39.1%
Class 3	35278	905	0.502%	2.565%	Bad	97.5%	84.0%
Class 4	5202	323	3.260%	6.209%	Bad	100.0%	100.0%

TABLE 7.6: 10-year PD prediction

Looking at the classification for the 10-year probability of default in Table 7.6, we see a classification in four classes. The majority of the test set still belongs to class 2. We observe again the same pattern as in the previous tables, i.e. there are significant differences between the predicted and actual PDs for most classes and also between the PDs of different classes, such that two classes are really different. In this case the AUC equals 0.757, which is slightly smaller than for the 5-year PD classification.

Using the TPR and FPR of all four time horizons, we can construct the CAP curves, which are displayed in Figure 7.3 on the next page.

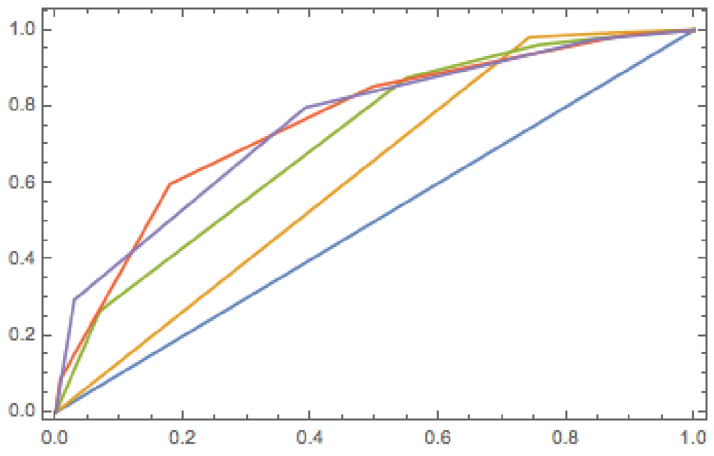

FIGURE 7.3: CAP curves without monthly performance data with PD horizons of 1 year (orange), 3 year (green), 5 year (red) and 10 year (purple)

7.3.2 Monthly performance variables

Now we have added the monthly performance variables to the set of accessible variables. The training and the test set are still the same as before. Since there is no performance of the loan known at the time of the origination, we have to add a time period, which is two years in this case, between the origination of the loan and the observation of the variables P_1 up to P_4. The results are again displayed in Table 7.7-7.10 below. Note that a PD prediction horizon of 1 year involves the 12 months after the observation date of the performance variables, not after the origination date. Therefore we have to exclude the loans from the training and test set for which we already know how they are terminated, i.e. both the defaulted loans as the prepaid loans. This results in a smaller data set.

1 year	# elements	# defaults	PD predicted	PD actual	Performance	TPR	FPR
Class 1	50522	22	0.011%	0.044%	Good	63.1%	4.1%
Class 2	18472	41	0.093%	0.222%	Moderate	86.2%	11.9%
Class 3	7567	33	0.141%	0.436%	Bad	95.6%	18.1%
Class 4	3961	435	4.956%	10.982%	Bad	100.0%	100.0%

TABLE 7.7: 1-year PD prediction with monthly performance variables

3 year	# elements	# defaults	PD predicted	PD actual	Performance	TPR	FPR
Class 1	13949	22	0.116%	0.158%	Good	17.5%	2.0%
Class 2	40467	133	0.059%	0.329%	Bad	68.3%	14.1%
Class 3	14534	171	0.266%	1.177%	Bad	86.4%	29.6%
Class 4	6041	91	0.542%	1.506%	Bad	93.9%	37.8%
Class 5	4298	231	0.911%	5.375%	Bad	99.0%	58.8%
Class 6	1233	454	26.479%	36.821%	Moderate	100.0%	100.0%

TABLE 7.8: 3-year PD prediction with monthly performance variables

5 year	# elements	# defaults	PD predicted	PD actual	Performance	TPR	FPR
Class 1	2611	1	0.000%	0.038%	Good	3.3%	0.1%
Class 2	55432	275	0.158%	0.496%	Bad	73.0%	20.2%
Class 3	9981	143	0.306%	1.433%	Bad	85.4%	30.7%
Class 4	9291	280	0.610%	3.014%	Bad	96.8%	51.2%
Class 5	1948	181	1.112%	9.292%	Bad	99.0%	64.5%
Class 6	1259	484	25.297%	38.443%	Bad	100.0%	100.0%

TABLE 7.9: 5-year PD prediction with monthly performance variables

10 year	# elements	# defaults	PD predicted	PD actual	Performance	TPR	FPR
Class 1	14740	64	0.138%	0.434%	Bad	18.7%	3.1%
Class 2	31108	330	0.344%	1.061%	Bad	58.0%	18.8%
Class 3	25491	645	0.336%	2.530%	Bad	89.6%	49.7%
Class 4	4763	211	0.634%	4.430%	Bad	95.4%	59.8%
Class 5	3074	323	1.461%	10.507%	Bad	98.9%	75.2%
Class 6	609	137	1.918%	22.496%	Bad	99.5%	81.7%
Class 7	737	382	45.448%	51.832%	Good	100.0%	100.0%

TABLE 7.10: 10-year PD prediction with monthly performance variables

We basically have the same observations as before for the different time horizons. However, there is one big difference between the two models. The PD (in particular the actual) of the final classes with the highest risk is much higher, i.e. there are far more defaults captured. This results in much better CAP curves and excellent AUC values of 0.926, 0.855, 0.824 and 0.774 respectively. This is also reflected in Figure 7.4 on the next page, containing the CAP curves of the four time horizons.

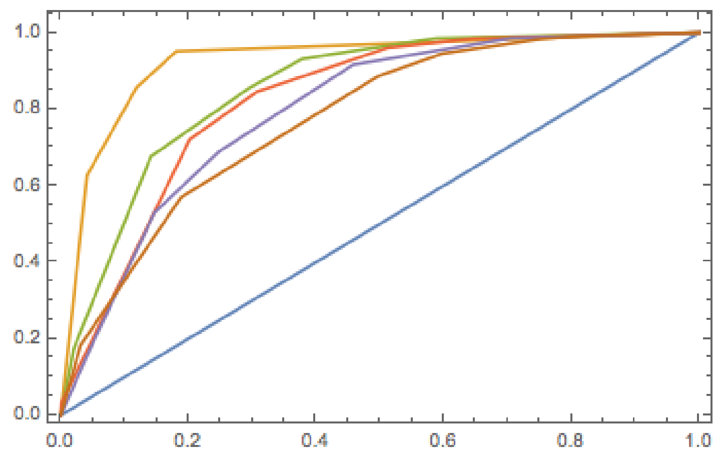

FIGURE 7.4: CAP curves with monthly performance data and PD horizons of 1 year (orange), 3 year (green), 5 year (red), 7 year (purple) and 10 year (brown)

7.3.3 Macro-economic variables

Finally the six macro-economic variables are included in the data set. The macro-economic variables are observed at the same date as the monthly performance variables. Obviously, as for many loans the observation date is the same, and some macro-economical variables only have a quarterly update, we will get many values that are (approximately) equal, which in particular can harm the performance if all examples with the same value are passed through the algorithm in succession. To avoid this, we extend our training set such that it contains loans from 2000 to 2007, such that it includes also some extreme observed values. Next, the test set equals some of the loans originating in 2008. Again we use a time period of 2 years between origination and observation of the P- and M-variables. We only consider a 1-, 3- and 5-year PD horizon, as no data is available for many loans to construct a 10-year PD horizon, as this horizon lies in our future.

1 year	# elements	# defaults	PD predicted	PD actual	Performance	TPR	FPR
Class 1	72622	223	0.192%	0.307%	Moderate	48.4%	4.1%
Class 2	17181	116	0.308%	0.675%	Bad	59.8%	11.9%
Class 3	14094	127	0.727%	0.901%	Moderate	69.1%	18.1%
Class 4	47231	974	2.195%	2.062%	Good	100.0%	100.0%

TABLE 7.11: 1-year PD prediction with macro-economic variables

3 year	# elements	# defaults	PD predicted	PD actual	Performance	TPR	FPR
Class 1	15139	26	0.193%	0.172%	Good	10.3%	0.6%
Class 2	72708	727	0.813%	1.000%	Moderate	59.3%	17.8%
Class 3	56274	1580	1.580%	2.808%	Bad	96.5%	55.3%
Class 4	7007	1887	30.430%	26.930%	Good	100.0%	100.0%

TABLE 7.12: 3-year PD prediction with macro-economic variables

5 year	# elements	# defaults	PD predicted	PD actual	Performance	TPR	FPR
Class 1	23	0	0.230%	0.000%	Bad	0.0%	0.0%
Class 2	1543	4	0.490%	0.259%	Bad	1.1%	0.1%
Class 3	47455	735	1.277%	1.549%	Moderate	33.4%	11.3%
Class 4	86717	2879	2.895%	3.320%	Good	91.4%	55.2%
Class 5	15390	2935	21.037%	19.071%	Good	100.0%	100.0%

TABLE 7.13: 5-year PD prediction with macro-economic variables

Observe that the observed PDs are now much closer to the predicted PD, especially for the larger classes, and the individual ranking still remains correct. However, we obtain fewer groups than in the case without macro-economic variables, resulting in AUC values of 0.707, 0.783 and 0.712 respectively and the following CAP curves.

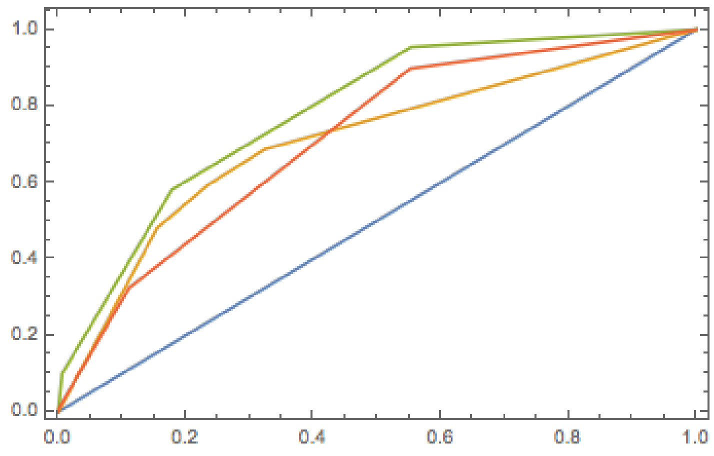

FIGURE 7.5: CAP curves with macro-economic variables and PD horizons of 1 year (orange), 3 year (green) and 5 year (orange)

7.4 Overall results

The Table 7.14 on the next page consists of various results obtained from training and testing with different parts of the FHLMC data set. Note that the first column denotes the origination year of the loans in the training set and the next column of the loans in the test set. Furthermore, in the third column one can find the intermediate period between the origination of the loan and the observation of the monthly performance variables. If this equals 0 years, it means that the monthly performance and macro-economic variables are excluded from the analysis. In column 4 up to 8 we find the AUC values for different values of the PD time horizon (see Section 7.2.2). The column with name 'Rankings' gives the percentage of class rankings for which the order of the classes is correct (see Section 7.2.1), and the next columns display for every developed model which percentage of the predicted PDs are 'good' compared to the actual observed PDs of the classes etc. (see Section 7.2.3). We performed four tests with a polynomial kernel that can be identified in the final column. The final eight tests were performed using a data set over a larger period and the final five also included macro-economic variables.

We observe a few things that are quite similar compared to the observation done in the previous section. We clearly see that the AUC for the PD with a 1-year horizon is much higher if we include monthly performance data, i.e. the model has high discriminatory power if we use a short period and information on monthly performance. This is also visible in Figure 7.6 below. Whereas the performance (in terms of the AUC) of these models decreases over time, as is expected by the fact that defaults over a longer period are more difficult to predict, we see that the performance of the models using only origination data does not change significantly over time and roughly stays between 0.55 and 0.7. However, even for a 7-year horizon including monthly performance data seems a good option.

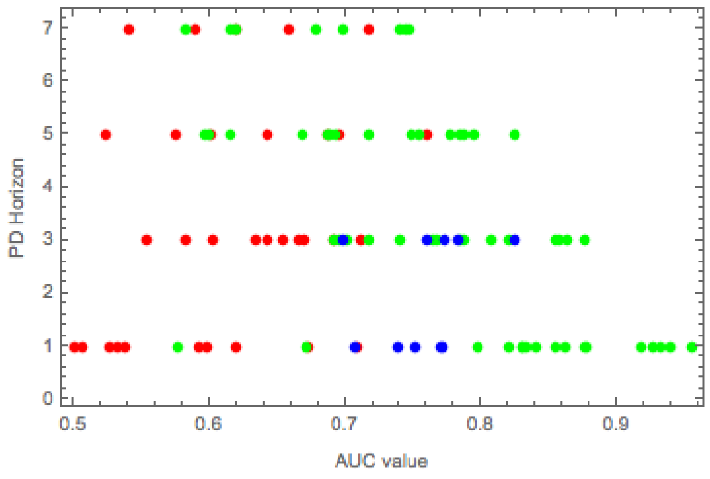

FIGURE 7.6: AUC values for different time horizons for a data set without (red) or with (green) monthly performance data and with both performance and macro-economic data (blue)

Furthermore, for the five tests with macro-economic variables, we see an increase in the percentage of good and moderate classified classes compared to all other models. However, it seems to go hand in hand with a loss of discriminatory power of the models.

			AUCs for different horizons						Rankings	PD Prediction			Kernel
			1 y	2 y	3 y	5 y	7 y	10 y		Good	Moderate	Bad	
2000	2001	0 y	0.620	■	0.712	0.760	■	0.757	100%	18%	12%	71%	
2000	2005	0 y	0.507	■	0.602	0.575	■	0.518	75%	13%	0%	88%	
2000	2008	0 y	0.527	■	0.642	0.601	0.590	■	75%	0%	7%	93%	
2000	2011	0 y	0.500	0.557	0.691	■	■	■	33%	33%	56%	11%	
2002	2004	0 y	0.593	■	0.654	0.687	0.717	■	75%	19%	5%	76%	
2002	2008	0 y	0.538	■	0.553	0.523	0.541	■	75%	11%	6%	83%	
2002	2011	0 y	0.598	0.588	0.634	■	■	■	67%	13%	13%	75%	
2005	2006	0 y	0.672	■	0.666	0.642	0.619	■	75%	28%	17%	56%	
2005	2008	0 y	0.709	■	0.670	0.696	0.659	■	100%	19%	5%	76%	
2005	2011	0 y	0.500	■	0.582	■	■	■	100%	60%	20%	20%	
2008	2011	0 y	0.532	0.643	■	■	■	■	100%	33%	0%	67%	
2000	2001	2 y	0.926	■	0.855	0.824	■	0.774	100%	14%	7%	79%	
2000	2001	5 y	0.932	■	0.864	0.795	0.747	■	100%	24%	0%	76%	
2000	2001	5 y	0.939	■	0.877	0.788	0.740	■	100%	19%	5%	76%	Degree: 2
2000	2001	5 y	0.918	■	0.857	0.784	0.744	■	100%	19%	5%	76%	Degree: 3
2000	2005	2 y	0.821	■	0.692	0.600	0.619	■	75%	5%	5%	90%	
2000	2005	2 y	0.830	■	0.701	0.597	0.583	■	75%	5%	5%	90%	Degree: 2
2000	2005	2 y	0.833	■	0.696	0.615	0.616	■	100%	5%	5%	90%	Degree: 3
2000	2005	5 y	0.862	■	0.787	0.689	■	■	67%	8%	0%	92%	
2000	2008	2 y	0.878	■	0.784	0.748	■	■	67%	23%	15%	62%	
2005	2006	2 y	0.671	■	0.740	0.687	0.679	■	100%	23%	14%	64%	
2005	2006	5 y	0.576	■	0.820	0.777	■	■	100%	0%	15%	85%	
2005	2008	2 y	0.877	■	0.764	0.669	■	■	67%	9%	36%	55%	
2005	2011	2 y	0.840	0.853	■	■	■	■	100%	29%	29%	43%	
2000	2011	2 y	0.955	0.860	■	■	■	■	100%	38%	25%	38%	
2000-2005		2 y	0.798	■	0.767	0.755	0.698	■	100%	13%	4%	83%	
2000-2005		5 y	0.831	■	0.717	0.693	■	■	100%	31%	0%	69%	
2000-2008		2 y	0.855	■	0.807	0.717	■	■	67%	15%	15%	69%	
Macro		2 y	0.707	■	0.783	0.712	■	■	100%	38%	31%	31%	
Macro		2 y	0.739	0.861	0.825	■	■	■	100%	29%	21%	50%	
Macro		2 y	0.771	0.748	0.698	■	■	■	100%	23%	23%	54%	
Macro		5 y	0.751	0.737	0.773	■	■	■	100%	43%	14%	43%	
Macro		5 y	0.770	0.809	0.760	■	■	■	100%	33%	17%	50%	

TABLE 7.14: Test results for different data sets and intermediate time periods

Also, we do not see an increase in the performance of the model using a polynomial kernel of degree 2 or 3. As the amount of time used for the development of a model increases drastically when we use second or third order monomials of the variables, it seems to be better to use only the linear version without any feature mappings.

Finally, we also did some research on the size of the training set. Certainly, if we use too few observations, we do not obtain a good model. However, in the previous case, using the loans of 2000 as training set and those of 2001 as test set, if we use only 25% of the training data, we obtain quite similar results (in terms of AUC of the 1-year PD horizon), as we can see below in Figure 7.7. Therefore, considering $T = 800,000$ observations is not entirely necessary, as we can also proceed with $T = 200,000$ observations and therefore reduce the amount of time that is used. Clearly, we also observe the linear relationship in the graph between the number of observations and the time used.

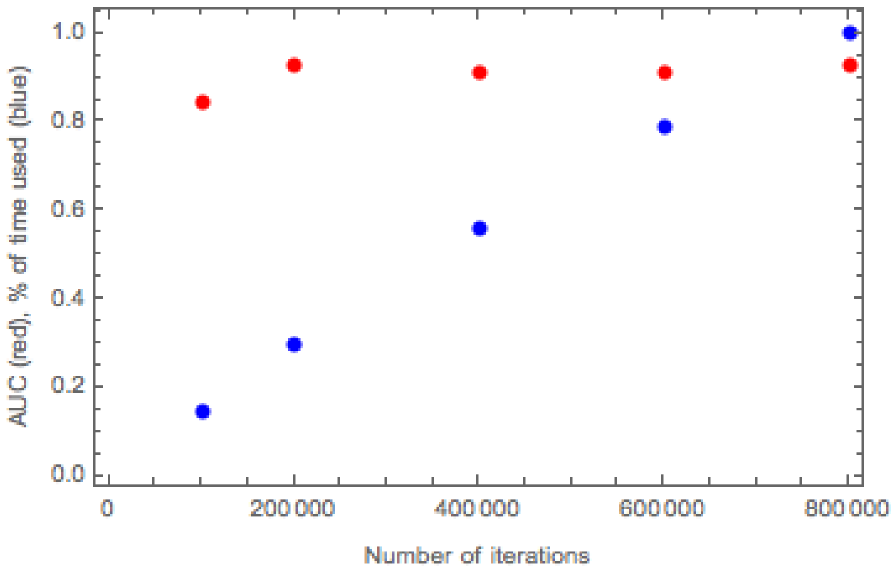

FIGURE 7.7: AUC for PD-horizon of 1 year using different training set sizes

7.4.1 Coefficient analysis

It is also interesting to look at the values of the coefficients of w in the hyperplanes and the offset b separating the different groups. Again we use the training set consisting of the loans originating in 2000. We include the monthly performance variables, but not the macro-economic variables, as before, and select 15 variables out of the 25 available. On the next page we see four tables (Table 7.15-7.18) with the 15 coefficients and the offset for each hyperplane.

	O_1	O_4	O_5	O_8	O_9	O_{12}	O_{15}	O_{16}
Plane 1	210.7	261.1	41.8	-199.4	-202.6	-117.5	15.1	-528.6
Plane 2	96.7	66.5	0.7	-44.9	2.1	-11.1	-41.9	-67.3
Plane 3	32.2	33.6	1.5	-20.4	3.1	-7.8	-8.8	1.6
	O_{17}	O_{19}	O_{20}	O_{21}	P_2	P_3	P_4	b
Plane 1	176.6	6.2	117.2	122.3	-516.8	294	-106.6	164.2
Plane 2	8.6	-4.8	92.5	46.5	-137.1	183.9	-17.4	46.2
Plane 3	1.4	-7.7	42.8	-9	-27.1	88.7	-5.1	22.4

TABLE 7.15: Coefficients of the 3 hyperplanes for a 1-year PD prediction

	O_1	O_4	O_5	O_8	O_9	O_{12}	O_{13}	O_{15}
Plane 1	8.9	-81.6	-124.3	-119	-67	-200.6	-41.8	-121.7
Plane 2	46.4	6.8	4.6	-69.8	-20.5	-80.7	11.1	-30.8
Plane 3	8	3.3	-6.6	-15.2	0.6	-19.7	0	-5.7
Plane 4	3.2	1.5	-0.5	-5.4	-0.4	-6.3	-2	-2
	O_{16}	O_{17}	O_{19}	O_{20}	O_{21}	P_2	P_4	b
Plane 1	-171.9	-117.2	-113.9	19	157.1	-313	48.6	-87.7
Plane 2	-58.3	-73.7	-33.1	39.9	-21.4	-98.8	41.9	48.6
Plane 3	-14.2	-0.3	4.3	0.8	4.8	-28.3	12.7	55
Plane 4	1.4	4.3	-0.1	-1.5	4.6	-5	3.3	28.5

TABLE 7.16: Coefficients of the 4 hyperplanes for a 3-year PD prediction

	O_1	O_4	O_5	O_8	O_9	O_{10}	O_{12}	O_{15}
Plane 1	155.3	-115.1	-6.1	-310.6	-95.9	50.9	-231.9	-86.9
Plane 2	6	21.6	0.9	-70.8	9.9	1.7	-75.3	-21.5
Plane 3	0	-5.9	-2.7	-14.1	-1.9	11.4	-30.6	-3.2
Plane 4	3.4	1.7	-0.4	-6.5	-0.7	5.5	-4.5	-2.8
	O_{16}	O_{17}	O_{19}	O_{20}	O_{21}	P_2	P_4	b
Plane 1	-717.6	-106	-43.2	-12.5	120.4	-295	181	-104
Plane 2	-119.3	-101.2	-26.6	-58.6	0.9	-112.3	35.8	16.2
Plane 3	-15.2	-16.9	-7.1	5.6	-16.1	-23	17.4	50.3
Plane 4	0.3	1	-1.3	-0.6	3.9	-6.9	3.2	28.7

TABLE 7.17: Coefficients of the 4 hyperplanes for a 5-year PD prediction

	O_1	O_4	O_5	O_8	O_9	O_{10}	O_{12}	O_{15}
Plane 1	95.4	234.6	-103.1	-154.8	70.73	92.9	-34.2	-282.2
Plane 2	-15.2	0.2	-13.6	-62.9	19.1	37.4	-73.1	-15.6
Plane 3	-5.3	-3.4	1.1	-17	0.6	5.6	-29.5	-3.7
	O_{16}	O_{17}	O_{19}	O_{20}	O_{21}	P_2	P_4	b
Plane 1	-783.4	-567.7	-147.1	29.6	-247.8	-410.5	-1.6	-395.1
Plane 2	-40.5	-28.4	-25.7	-30.6	-26.9	-119.9	32.1	35.4
Plane 3	-14.4	7.5	-5.3	-6	4.3	-27.3	20.6	37.1

TABLE 7.18: Coefficients of the 4 hyperplanes for a 10-year PD prediction

The first thing we observe is that the coefficients of plane 1 are much larger in absolute value than those of plane 3 and plane 4. This is caused by the use of larger weights to compute group G_1 as is described in the first section of this chapter, which results in larger gradient values. Furthermore we see that the selected variables slightly differ over the time horizons, but not that much.

We can also look at the variables that consistently have the largest absolute values for the coefficients, such as O_8 (CLTV), O_{12} (original interest rate), O_{16} (property type), O_{17} (postal code), O_{19} (number of borrowers), O_{21} (servicer name) and P_2 (maximum loan delinquency). These variables appear to have the most predicting power. As all variables are scaled between -1 and 1, there is no influence of the absolute size of the variables itself.

7.5 Comparison with other models

It is also useful to compare the performance of this model with some other (traditional) credit risk models as described in Chapter 2. Since the calculation of the FICO scores, which are used in the Freddie Mac database, is not transparent, we cannot directly use this model with the inclusion of macro-economic or monthly performance variables.

We have performed two comparisons with another model, again using the data set as in Secton 7.3. The first model is solely based on the *FICO credit scores* provided by Freddie Mac, whereas the second model uses a *multiple linear regression*, which can also include macro-economic variables. For both models we divide the training set in five different non-empty groups, based on their initial credit score:

- Class 1: credit score ≥ 800
- Class 2: $750 \leq$ credit score ≤ 799
- Class 3: $700 \leq$ credit score ≤ 749
- Class 4: $650 \leq$ credit score ≤ 699
- Class 5: credit score ≤ 649

For the first comparison, using only the credit scores, we divide the test set using the same procedure, and look at the default rates of these classes. For the second comparison, we perform a linear regression, as described in Section 2.4.1, using all available variables, and based on the classification of the training examples. In the second model we also include monthly performance data. The results are given in the table below.

	AUC				Ranking	PD Prediction		
	1 y	3 y	5 y	10 y		Good	Moderate	Bad
Credit scores	0.756	0.733	0.701	0.693	100%	20%	20%	60%
Linear regression	0.871	0.809	0.743	0.742	100%	33%	0%	67%

TABLE 7.19: Performance results of the credit scores method and linear regression

Both models perform well on a 1-, 3-, 5- and 10-year PD horizon. For the first model based on the FICO credit scores we obtain AUC values of 0.756, 0.733, 0.701 and 0.693 respectively. For the second model, the AUC values are significantly better for the first two time horizons, i.e. 0.871, 0.809, 0.743 and 0.742. However, when monthly performance data is included, the weighted SVM model introduced in this thesis has a much better discriminatory power. Both comparison models correctly predict the rankings for the test set, i.e. class 1 has a smaller PD than class 2 etc.. Nonetheless, for these models we see that the predicted PDs are in 60% and 67% of the cases classified as bad, therefore performing not significantly better than the support vector machine.

Overall we can conclude that the weighted SVM might have better discriminatory power, especially for short time horizons, but does not performs better in ranking the classes or predicting the PDs than the existing models.

Chapter 8

Conclusion

In this thesis we investigated the possibility of applying a support vector machine to a data set of mortgage loans by Freddie Mac, in order to develop a model for classifying these loans in classes with a successive probability of default. Furthermore this model should to be able to provide an estimate of the probability of default for the loans in these classes.

A modified version of the support vector machine (SVM) has been used in this thesis. We introduced a method called *weighted SVM*, which consists of assigning certain (higher) weights to defaults. This is motivated by the fact that defaults are fairly outnumbered by non-defaulted loans. Therefore, applying a non-weighted version of the SVM results in a model that predicts every loan to go not in default. The weighted SVM creates several different classes, such that the loans with a higher PD should be in a different class than the loans with a smaller PD. By varying the value of the weights, we can also change the size of the these classes.

Together with a weighted SVM, we use the *averaged stochastic (sub)gradient descent* to minimize the objective function obtained from the SVM. The stochastic gradient descent is a technique that minimizes the objective function by going in the direction of the negative gradient, which results in the fastest descent. The stochastic component comes from the fact that the gradient is computed with respect to one single, randomly picked observation. As a high number of successive defaults (with large weights), could cause the stochastic gradient descent to end up in a model that predicts all loans to go into default, it turned out to be better to use the averaged stochastic gradient descent. This essentially consists of averaging the iterations obtained from the model, using a particular scheme.

Our weighted SVM has turned out to be a good model to discriminate between classes of different PDs. We have built our model using different training sets and different sets of variables. Including information on the monthly performance of the loans turns out to be a big improvement in discriminatory power, as the model is able to put up to 40% of the defaults in the final class, containing less than 2% of the observations. Also, adding macro-economic variables increases the predictive power of the model.

The model is unique in the sense that it does not need an initial ranking of the training set in different credit classes. Traditional models based on linear regression or probit or logit need a classification of the training set in order to estimate the coefficients of the model. The performance with respect to the existing scoring models is comparable, however, the possibility of including both monthly performance and macro-economic data makes this new technique interesting to use in the future. The results (in terms of predictive power) are not that spectacular, as we have used a typical classification problem to predict a continuous variable. However, the discriminatory power of the model of this thesis is still quite impressive.

8.1 Opportunities

Clearly, before the model is suitable for its intended use in credit risk modeling, it has to be tested and further developed. In the next years it will be possible for this data set to look at lifetime probabilities of default, as many mortgage loans that are still need to terminate, will end after their specified 15-, 20- and 30-year period. Due to the short period that the data was available this was not possible yet. Also, if more actual (monthly performance) data is present, such as current debt-to-income ratio, we could possibly improve the model. Obviously, testing on another data set, is also necessary before we could use the weighted SVM in practice.

Also the dual version of the problem, that turned out to be much more complex than the linear weighted SVM, may deserve some attention, as it shows some promising results in credit risk. A great disadvantage of the dual version is that we can only have a small data set, since we have to store all kernel values in our memory. It would be worthwhile to investigate if the dual problem combined with some clustering, that sets all kernel values to zero for two elements from unequal clusters, can lead to good results.

Also, there are some other methods to deal with a disbalance in the number of elements in the two classes. It is worthwhile to test if these also can be used for a multiclass SVM.

Bibliography

Allen, L., G. DeLong, and A. Saunders (2004). “Issues in the Credit Risk Modeling of Retail Markets”. In: *Journal of Banking & Finance* 4, pp. 727–752. URL: http://papers.ssrn.com/sol3/papers.cfm?abstract_id=412520.

Balder, E. J. (2010). "On subdifferential calculus". Notes for Ph.D. course on Convex Analysis for Optimization. URL: http://www.staff.science.uu.nl/~balde101/cao10/cursus10_1.pdf.

Ben-Hur, A. et al. (2001). “Support vector clustering”. In: *Journal of Machine Learning Research* 2, pp. 125–137. URL: http://www.jmlr.org/papers/volume2/horn01a/rev1/horn01ar1.pdf.

Benzin, A., S. Trück, and S. T. Rachev (2003). “Approaches to Credit Risk in the New Basel Capital Accord”. In: *Credit Risk: Measurement, Evaluation and Management*. URL: http://www.pstat.ucsb.edu/research/papers/benzin_trueck.pdf.

BIS (2000). "Principles for the Management of Credit Risk". Report Basel Committee on Banking Supervision. URL: http://www.bis.org/publ/bcbs75.pdf.

Boyd, S. and L. Vandenberghe (2004). "Convex Optimization". Cambridge Press.

Boyd, S., L. Xiao, and A. Mutapcic (2003). "Subgradient Methods". Notes for EE392o, Stanford University. URL: https://web.stanford.edu/class/ee392o/subgrad_method.pdf.

Crammer, K. and Y. Singer (2001). “On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines”. In: *Journal of Machine Learning Research* 2, pp. 265–292. URL: http://jmlr.csail.mit.edu/papers/volume2/crammer01a/crammer01a.pdf.

Dikkers, H. J. (2005). “Support vector machines in ordinal classification: A revision of the ABN AMRO corporate credit rating system”. MA thesis. TU Delft.

Dobbs, R. et al. (2015). "Debt and (not much) deleveraging". Report McKinsey Global Institute. URL: http://www.mckinsey.com/global-themes/employment-and-growth/debt-and-not-much-deleveraging.

Draper, N. R. and H. Smith (1998). "Applied Regression Analysis (Wiley Series in Probability and Statistics)". John Wiley & Sons, Inc.

Drucker, H. et al. (1996). “Vector Regression Machines”. In: *Advances in Neural Information Processing Systems* 13, pp. 155–161. URL: https://papers.nips.cc/paper/1238-support-vector-regression-machines.pdf.

Duda, R. O., P. E. Hart, and D. G. Stork (2001). "Pattern Classification". John Wiley & Sons, Inc.

FHMLC (2016). "Single-Family Loan-Level Dataset Frequently Asked Questions".

Gordon, G. and R. Tibshirani (2012). "Lecture 5: Gradient Descent Revisited". Notes for 10-725: Optimization, Carnegie Mellon University. URL: https://www.cs.cmu.edu/~ggordon/10725-F12/scribes/10725_Lecture5.pdf.

Hsu, C. and C. Lin (2002). “A comparison of methods for multiclass support vector machines”. In: *IEEE Transactions on Neural Networks* 13, pp. 415–425. URL: https://www.csie.ntu.edu.tw/~cjlin/papers/multisvm.pdf.

Huang, Y. (2005). “Weighted support vector machine for classification with uneven training class sizes”. In: *International Conference on Machine Learning and Cybernetics* 7, pp. 4365–4369.

IFRS (2013). "Financial Instruments: Expected Credit Losses". Exposure Draft ED / 2013 / 3. URL: http://www.hkicpa.org.hk/file/media/section6_standards/standards/FinancialReporting/ed-pdf-2013/ed_crlost.pdf.

Khardon, R. (2008). "Lecture 18". Notes for 150 AML: Advanced Topics in Machine Learning. URL: http://web.iitd.ac.in/~sumeet/CLT2008S-lecture18.pdf.

Kim, K. S. and J. R. Scott (1991). “Prediction of Corporate Failure: an Artificial Neural Network Approach”. Southwest Missouri State University.

Kutner, M. et al. (1974). "Applied Linear Statistical Models". McGraw-Hill/Irwin.

Lapin, M., M. Hein, and B. Schiele (2014). “Learning using privileged information: SVM+ and weighted SVM”. In: *Neural Networks* 53, pp. 95–108. URL: http://www.sciencedirect.com/science/article/pii/S0893608014000306.

Mela, C. F. and P. K. Kopalle (2002). “The impact of collinearity on regression analysis: the asymmetric effect of negative and positive correlations”. In: *Applied Economics* 34, pp. 667–677. URL: https://faculty.fuqua.duke.edu/~mela/bio/papers/Mela_Kopalle_2002.pdf.

Mian, A. and A. Sufi (2014). "House of Debt: How They (and You) Caused the Great Recession, and How We Can Prevent It from Happening Again". The University of Chicago Press.

Mitchell, T. (1997). "Machine Learning". McGraw-Hill/Irwin, p. 2.

Ng, A. (2016). "Machine Learning". Lectures on Machine Learning (CS 229), Stanford University. URL: http://cs229.stanford.edu/.

Qiao, X. and L. Zhang (2015). “Distance-weighted Support Vector Machine”. In: *Statistics and Its Interface* 8, pp. 331–345. URL: https://arxiv.org/abs/1310.3003.

Rakhlin, A. and O. Shamir (2012). “Making Gradient Descent Optimal for Strongly Convex Stochastic Optimization”. In: *ICML*. URL: https://arxiv.org/pdf/1109.5647.pdf.

Saunders, A. and L. Allen (2002). "Credit Risk Management: New Approaches to Value at Risk and Other Paradigms". John Wiley & Sons, Inc.

Shamir, O. and T. Zhang (2013). “Stochastic Gradient Descent for Non-smooth Optimization: Convergence Results and Optimal Averaging Schemes”. In: *Journal of Machine Learning Research* 28. URL: http://jmlr.csail.mit.edu/proceedings/papers/v28/shamir13.pdf.

Simon, P. (2013). "Too Big to Ignore: The Business Case for Big Data". Wiley, p. 89.

Smith, K. (2011). "List of 16 Major Leading & Lagging Economic Indicators".

Still, G. (2015). "Continuous Optimization". Lectures on continuous optimization at Universtiy of Utrecht. URL: http://wwwhome.math.utwente.nl/~stillgj/conopt/.

Tape, T. G. "Interpreting Diagnostic Tests". Univeristy of Nebraska, Medical Center. URL: http://gim.unmc.edu/dxtests/Default.htm.

Tibshirani, R. (2012). "Lecture 7: September 18". Notes for 10-725: Optimization, Carnegie Mellon University.

Whitley, E. and J. Ball (2002). “Statistics review 5: Comparison of means”. In: *Crit Care* 6, pp. 424–428. URL: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC137324/.

Yang, X. (2005). “Weighted support vector machine for data classification”. In: *IEEE International Conference on Neural Networks* 2, pp. 859–864.

Yashtini, M. (2015). “On the global convergence rate of the gradient descent method for functions with Hölder continuous gradients”. In: *Optimization Letters* 9. URL: http:

//people.math.gatech.edu/~myashtini3/assets/publication/pdf/GDHolderGrad.pdf.

Appendix A

Optimization Theory

A.1 Duality principle

In this section we will have a closer look at the so called *duality principle*, the idea that an optimization problem can be viewed from two perspectives, the *primal* or the *dual problem*. The solution of the dual problem provides a lower bound for the solution of the primal problem. The generally non-zero difference between these solutions is called the *duality gap*.

We are given the following optimization problem:

$$\begin{aligned} \inf_{x \in X \subset \mathbb{R}^n} \quad & f(x) \\ \text{s.t.} \quad & g_i(x) \leq 0 \quad \text{for all } 1 \leq i \leq m \\ & h_j(x) = 0 \quad \text{for all } 1 \leq j \leq p \end{aligned} \tag{A.1}$$

Here we do not pose any restrictions on the functions f, g_i and h_j, like linearity or convexity.

In many cases we consider the minimum instead of the infimum of f. In this general case we take the infimum because we want to include the possibility that the optimal value is $-\infty$. Switching to the supremum (or maximum) can be done by considering the infimum (minimum) of the function $-f(x)$.

We now define the *Lagrangian* $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$

$$\mathcal{L}(x, \alpha, \beta) = f(x) + \sum_{i=1}^{m} \alpha_i g_i(x) + \sum_{j=1}^{p} \beta_j h_j(x) \tag{A.2}$$

Here we introduced the vectors α and β, which are called the *dual variables* or *Lagrange multiplier vectors*. Then we can write the primal problem as follows

$$\inf_{x \in X \subset \mathbb{R}^n} \sup_{\alpha_i \geq 0, \beta_j \in \mathbb{R}} \mathcal{L}(x, \alpha, \beta) \tag{A.3}$$

Note that $\sup_{\alpha_i \geq 0} \alpha_i g_i(x) = \infty$ if and only if $g_i(x) > 0$ and it equals 0 if $g_i(x) \leq 0$. Therefore also $\sup_{\beta_j \in \mathbb{R}} \beta_j h_j(x) = \infty$ if and only if $h_i(x) \neq 0$. Therefore the constraints of problem A.1 hold if and only if

$$\sup_{\alpha_i \geq 0, \beta_j \in \mathbb{R}} \left(f(x) + \sum_{i=1}^{m} \alpha_i g_i(x) + \sum_{j=1}^{p} \beta_j h_j(x) \right) = f(x) \tag{A.4}$$

Now we can introduce the so called *dual problem* by switching the infimum and the supremum

$$\sup_{\alpha_i \geq 0, \beta_j \in \mathbb{R}} \inf_{x \in X \subset \mathbb{R}^n} \mathcal{L}(x, \alpha, \beta) \tag{A.5}$$

It is a general result that taking the supremum after an infimum yields a value that it smaller than or equal to the infimum of a supremum. We denote the optimal value of the primal problem by p^* and of the dual problem by d^*. We always have *weak duality*, i.e. $d^* \leq p^*$. However, there are some conditions under which $d^* = p^*$. These conditions are given by the *Strong Duality Theorem*. Before we state this theorem, we need to obtain some knowledge about convex optimization.

A.2 Convex optimization

We call Problem A.1 a *convex optimization problem* if the set X is convex, the objective function f and constraint functions g_i are convex and the constraint functions h_j are *affine*, so both $-h_j$ and h_j are convex. A convex optimization problem has the nice property that a local minimum is also directly a global minimum.

Another interesting property of convex optimization deals with the so called *Karush-Kuhn-Tucker (KKT) conditions*. A point $x \in \mathbb{R}^n$ is said to be *feasible* if it satisfies the constraints of the (primal) optimization problem A.3. A feasible x^* (also denoted by (x^*, α^*, β^*)) satisfies the KKT conditions if there are α^*, β^* such that $\alpha_i^* \geq 0$ for all i such that

$$\nabla f(x^*) = -\sum_{i=1}^{m} \alpha_i^* \nabla g_i(x^*) - \sum_{j=1}^{p} \beta_j^* \nabla h_j(x^*) \tag{A.6}$$

$$\alpha_i^* g_i(x^*) = 0 \quad \text{for all } 1 \leq i \leq m \text{ (\textit{complementary slackness})}$$

We also introduce the so called *Slater condition*. We say that an optimization problem satisfies the Slater condition if there exists a *Slater point*, i.e. if there exists an $\widehat{x}$ in the *relative interior* of X such that

$$g_i(\widehat{x}) < 0 \quad \text{for all } i \text{ with } g_i \text{ nonlinear} \tag{A.7}$$

$$g_i(\widehat{x}) \leq 0 \quad \text{for all } i \text{ with } g_i \text{ affine} \tag{A.8}$$

$$h_j(\widehat{x}) = 0 \quad \text{for all } j \tag{A.9}$$

The relative interior of a convex set is defined as $\text{relint}(X) := \{x \in X | \text{ for all } y \in X \text{ there exist } \lambda > 1 \text{ s.t. } \lambda x + (1-\lambda)y \in X\}$. A Slater point $\widehat{x} \in X$ is called an *ideal Slater point* if

$$g_i(\widehat{x}) < 0 \quad \text{for all } i \in J_r \tag{A.10}$$

where $J_r := \{1 \leq i \leq m | g_i(x) < 0 \text{ for some } x \in \mathcal{F}\}$, where $\mathcal{F}$ is the set of feasible points. This set is called the set of *regular constraints*. We also define $J_s := J \backslash J_r = \{1 \leq i \leq m | g_i(x) = 0 \text{ for all } x \in \mathcal{F}\}$, the set of *singular constraints*. Note that for a convex optimization problem satisfying the Slater condition, the set of singular constraints can only consist of linear constraints. Two important properties are stated in the next lemma.

Lemma A.1. *If the convex optimization problem A.1 satisfies the Slater condition, it has an ideal Slater point $\widehat{x} \in \mathcal{F}$, which is in the relative interior of X.*

Proof. Let $x \in \mathcal{F}$ be a Slater point of the problem. Clearly $g_i(x) < 0$ for all nonlinear functions. Now let $g_j(x) = 0$ for some affine regular constraint. Let $y \in \mathcal{F}$ be a point such that $g_j(y) < 0$. Clearly, by convexity of the problem we have $\lambda x + (1-\lambda)y \in \mathcal{F}$ for $\lambda \in (0,1)$ and

$$g_j(\lambda x + (1-\lambda)y) \leq \lambda g_j(x) + (1-\lambda)g_j(y) < 0 \tag{A.11}$$

Repeat doing this until $g_j(x) < 0$ for all affine regular constraints. Then we see that we found an ideal Slater point $\widehat{x}$.

Note that $\widehat{x} \in \text{relint}(\mathcal{F})$ if and only if for all $y \in \mathcal{F}$ there exists a $z \in \mathcal{F}$ and a $\lambda > 1$ such that $\lambda\widehat{x} + (1-\lambda)y = z$, i.e. $\widehat{x} = \frac{1}{\lambda}z + (1-\frac{1}{\lambda})y$. Define $\mu = \frac{1}{\lambda}$. Then we see $\widehat{x} \in \text{relint}(\mathcal{F})$ if and only for every $y \in \mathcal{F}$ there exists $\mu \in (0,1)$ and $z \in \mathcal{F}$ such that

$$\widehat{x} = \mu z + (1-\mu)y \tag{A.12}$$

By construction, $\widehat{x}$ is in the relative interior of X, since it is defined as a strictly convex combination of an element of the relative interior of X and elements of $\mathcal{F} \subset X$. □

A.3 Farkas Lemma

As a preparation for the Strong Duality Theorem we prove *Farkas Lemma*.

Lemma A.2 (Farkas). *Let Problem A.1 be convex and satisfy the Slater condition, without any equality constraints. Then for $a \in \mathbb{R}$ the inequality system*

$$\begin{aligned} f(x) &< a \\ g_i(x) &\leq 0 \quad \text{for all } 1 \leq i \leq m \\ x &\in X \end{aligned} \tag{A.13}$$

has no solution if and only if there is a vector α such that

$$\begin{aligned} \alpha_i &\geq 0 \quad \text{for all } 1 \leq i \leq m \\ f(x) + \sum_{i=1}^{m} \alpha_i g_i(x) &\geq a \quad \text{for all } x \in X \end{aligned} \tag{A.14}$$

Proof. First, assume that both system A.13 and A.14 have a solution. Then, there is an $x \in X$ and $\alpha \geq 0$ such that

$$a \leq f(x) + \sum_{j=1}^{m} \alpha_i g_i(x) \leq f(x) < a \tag{A.15}$$

which yields a contradiction. Therefore at most one of the system can have a solution. Let us therefore assume that system A.13 has no solution, and show that system A.14 must have a solution. Consider the following modified system

$$\begin{aligned} f(x) &< a + u_0 \\ g_i(x) &\le u_i \quad \text{for all } i \in J_r \\ g_i(x) &= u_i \quad \text{for all } i \in J_s \\ x &\in X \end{aligned} \tag{A.16}$$

Let $u = (u_0, ..., u_m)$ and define $\mathcal{U}$ to be the set of u which are feasible for system A.16. Then clearly $0 \notin \mathcal{U}$. One can check that $\mathcal{U}$ is non-empty, by taking $u_i \to \infty$ for all $i \in J_r$ and using that the singular constraints are linear. It is also convex, since convex combinations of solutions are still clearly feasible. Observe that it is therefore possible to separate 0 and $\mathcal{U}$ by a hyperplane (due to the *Separation Theorem*), given by $\{x | \alpha^T x + b = 0\}$, where

$$\begin{aligned} &\alpha^T u \ge b \ge \alpha^T 0 \quad \text{for all } u \in \mathcal{U} \\ &\alpha^T \widehat{u} > b \quad \text{for some } \widehat{u} \in \mathcal{U} \end{aligned} \tag{A.17}$$

for some $\alpha = (\alpha_0, \ldots, \alpha_m)$ and $b \ge 0$. Now take $u \in \mathcal{U}$ and some $i \in \{0\} \cup J_r$. Clearly $u + e_i$ is still feasible, where e_i is the ith unit vector. Therefore $y^T(u + e^j) \ge 0$ by A.17 and we see $y^T e^j = y_j = 0$, for all $i \in \{0\} \cup J_r$.

Let $u_x = (f(x) - a + \lambda, g_1(x), \ldots, g_m(x))$ with $\lambda > 0$. Clearly $u_x \in \mathcal{U}$ for all $x \in X$. Therefore $\alpha^T u_x \ge b \ge 0$. By taking the limit $\lambda \to 0$ we obtain

$$\alpha_0(f(x) - a) + \sum_{i=1}^{m} \alpha_i g_i(x) \ge 0 \quad \text{for all } x \in X \tag{A.18}$$

We now show that $\alpha_0 > 0$, so that we can set $\alpha_0 = 1$ and obtain our desired result from Equation A.18. Assume to the contrary that $\alpha_0 = 0$. Then for all $x \in X$

$$0 \le \sum_{i=1}^{m} \alpha_i g_i(x) = \sum_{i \in J_r} \alpha_i g_i(x) + \sum_{i \in J_s} \alpha_i g_i(x) \tag{A.19}$$

If we take an ideal Slater point $\widehat{x}$, we have $g_i(\widehat{x}) < 0$ for all $i \in J_r$ and $\sum_{i \in J_r} \alpha_i g_i(x) \ge 0$. Using $\alpha_i \ge 0$ we obtain $\alpha_i = 0$ for $i \in J_r$. Therefore for all $x \in X$

$$\sum_{i \in J_r} \alpha_i g_i(x) \ge 0 \tag{A.20}$$

Now let $\widehat{u} \in \mathcal{U}$ be such that $\alpha^T \widehat{u} > b \ge 0$. By definition of $\mathcal{U}$ we can find an $\bar{x} \in X$ such that $\widehat{u} = g_i(\bar{x})$ for $i \in J_s$. Therefore

$$\alpha^T \widehat{u} = \sum_{i \in J_s} \alpha_i g_i(\bar{x}) > 0 \tag{A.21}$$

Because $\widehat{x}$ is an ideal Slater point, it is in the relative interior of X by Lemma A.1. As is shown in the proof of that lemma, there exists a $z \in X$ and a $\mu \in (0, 1)$ such that $\widehat{x} = \mu\bar{x} + (1 - \mu)z$. Since $g_i(\widehat{x}) = 0$ for $i \in J_s$. Therefore

$$\begin{aligned} 0 &= \sum_{i\in J_s} \alpha_i g_i(\widehat{x}) = \sum_{i\in J_s} \alpha_i g_i(\mu\bar{x} + (1-\mu)z) \\ &= \mu \sum_{i\in J_s} \alpha_i g_i(\bar{x}) + (1-\mu)\sum_{i\in J_s} \alpha_i g_i(z) > (1-\mu)\sum_{i\in J_s} \alpha_i g_i(z) \end{aligned} \tag{A.22}$$

where the third equality follows from the fact that all singular constraints are linear for a convex optimization problem satisfying the Slater condition. The inequality is due to Equation A.21. Using $1-\mu > 0$ we see

$$\sum_{i\in J_s} \alpha_i g_i(z) < 0 \tag{A.23}$$

This is a contradiction with the inequality in Equation A.20. Therefore we obtain $\alpha_0 > 0$ and we can assume $\alpha_0 = 1$. Note that we have now proven the lemma in the case that there are no singular constraints.

The final part of the proof is showing that $\alpha_i \geq 0$ for $i \in J_s$. First assume that we have only one singular constraint function $g_m(x)$. Consider the following optimization problem

$$\begin{aligned} g_m(x) &< 0 \\ g_i(x) &\leq 0 \quad \text{for all } i \in J_r \\ x &\in X \end{aligned} \tag{A.24}$$

We have now made our singular constraint the objective function and we are only left with regular constraints. It is still a convex optimization problem satisfying the Slater condition, since we only have fewer constraints than in the original problem. We can already apply Farkas Lemma to this inequality system, and see that this system has no solution if and only if the following system has a solution

$$g_m(x) + \sum_{i\in J_r} \widehat{\alpha}_i g_i(x) \geq 0 \tag{A.25}$$

where $\widehat{\alpha}_i \geq 0$ for all $i \in J_r$. We already obtained

$$f(x) - a + \sum_{i\in J_r} \alpha_i g_i(x) \geq 0 \tag{A.26}$$

where $\alpha_i \geq 0$ for all $i \in J_r$. By adding Equation A.25, possibly multiplied with a constant $\alpha_m \geq 0$, to Equation A.26 we obtain

$$f(x) - a + \sum_{i\in J_r} (\alpha_i + \alpha_m \widehat{\alpha}_i) g_i(x) + \alpha_m g_m(x) \tag{A.27}$$

By redefining $\alpha_i = \alpha_i + \alpha_m \widehat{\alpha}_i \geq 0$ we obtain again an equation in the shape of A.26. By induction on the cardinality of the set J_s we find $\alpha_i \geq 0$ for all $i \in J_s$, which closes the proof. □

It is easy to generalize Farkas Lemma to the case with equality constraints $h_j(x) = 0$. Each equality constraint can be written as two inequality constraints $h_j(x) \leq 0$ and

$-h_j(x) \leq 0$ and we can apply Farkas Lemma to the entire system of inequality constraints including the new ones. Since we obtain positive coefficients for both the inequality constraint with h_j and with $-h_j$, the (total) coefficient β_j for h_j can be any real number, instead of a positive real number. This gives the following corollary.

Corollary A.3. *Let Problem A.1 be convex and satisfy the Slater condition. Then for $a \in \mathbb{R}$ the inequality system*

$$\begin{aligned} f(x) &< a \\ g_i(x) &\leq 0 \quad \text{for all } 1 \leq i \leq m \\ h_j(x) &= 0 \quad \text{for all } 1 \leq j \leq p \\ x &\in X \end{aligned} \tag{A.28}$$

has no solution if and only if there are vectors α and β such that

$$\begin{aligned} \alpha_i \geq 0 \quad &\text{for all } 1 \leq i \leq m \\ f(x) + \sum_{i=1}^{m} \alpha_i g_i(x) \geq a \quad &\text{for all } x \in X + \sum_{j=1}^{p} \beta_j h_j(x) \end{aligned} \tag{A.29}$$

A.4 Strong Duality Theorem

The Strong Duality Theorem can now be proven using Farkas Lemma.

Theorem A.4 (Strong Duality Theorem). *Given the optimization problem A.1. Let $X \in \mathbb{R}^n$ be* non-empty convex, *let $f : X \to \mathbb{R}$ and each $g_i : \mathbb{R}^n \to \mathbb{R}$ be* convex functions *and each $h_j : \mathbb{R}^n \to \mathbb{R}$ be* affine functions, *i.e. the optimization problem is convex.*
If the Slater condition holds, and $0 \in int\ (h(X))$ where $h(X) = \{h(x) : x \in X\}$, then

$$p^* = d^* \tag{A.30}$$

Proof. Note that the Slater condition holds, i.e. there is at least one feasible point, so $p^* < \infty$. since the infimum over a set is ∞ if and only if the set is empty. If $p^* = -\infty$, we clearly have $d^* = -\infty = p^*$ by weak duality. Therefore consider the case that $p^* > -\infty$. Then let $a = p^*$ and apply the generalized version of Farkas Lemma (Corollary A.3). Since a equals the infimum of f obtained from the optimization problem, we see that the first system A.28 has no solution and therefore there exists a vector $\alpha \geq 0$ and a vector β such that

$$f(x) + \sum_{i=1}^{m} \alpha_i g_i(x) + \sum_{j=1}^{p} \beta_j h_j(x) \geq a \quad \text{for all } x \in X \tag{A.31}$$

Clearly, we recognize the term before the inequality sign as the Lagrangian $\mathcal{L}(x, \alpha, \beta)$. Therefore $\inf_{x \in X} \mathcal{L}(x, \alpha, \beta) \geq a$. Clearly

$$d^* = \sup_{\alpha \geq 0, \beta \in \mathbb{R}} \left(\inf_{x \in X} \mathcal{L}(x, \alpha, \beta) \right) \geq a = p^* \tag{A.32}$$

By weak duality we have, as usual, $d^* \leq p^*$. Therefore we have $d^* = p^*$ and strong duality holds. □

A.5 Convex dual problem

In general, when $X = \mathbb{R}^n$ the infimum of $\mathcal{L}(x, \alpha, \beta)$ is either $-\infty$ or attained at some local minimum. At the local minimum we must have $\nabla_x \mathcal{L}(x, \alpha, \beta) = 0$, i.e. the gradient with respect to the primal variable x has to equal 0.

For a convex optimization problem, it is guaranteed that any local minimum is also the global minimum, since we are minimizing a convex function f on the convex set defined by convex constraint functions. When the objective function and the constraint functions are continuously differentiable (C^1), the dual problem A.5 can be made much easier by first calculating the gradient of the Lagrangian with respect to the primal variable x and setting it to zero. This gives in fact a new optimization problem

$$\begin{aligned} \sup_{\alpha_i \geq 0, \beta_j \in \mathbb{R}} \quad & \mathcal{L}(x, \alpha, \beta) \\ \text{s.t.} \quad & \nabla_x \mathcal{L}(x, \alpha, \beta) = 0 \end{aligned} \tag{A.33}$$

A.6 Karush-Kuhn-Tucker Theorem

In Section A.2 we have introduced the KKT conditions. For a convex optimization problem satisfying the Slater condition, it is very useful to find a point satisfying the KKT conditions, since this will also be a minimizer of the problem. To prove this, we will make use of the *Karush-Kuhn-Tucker Theorem*. Note that we have to assume from now on that both the objective function f and the constraint functions g_i and h_j are differentiable functions of x, otherwise the KKT conditions do not make sense.

A point $(\widehat{x}, \widehat{\alpha}, \widehat{\beta}) \in \mathbb{R}^{n+m+p}$ is called a *saddle point* of the Lagrangian $\mathcal{L}$ if

$$\mathcal{L}(\widehat{x}, \alpha, \beta) \leq \mathcal{L}(\widehat{x}, \widehat{\alpha}, \widehat{\beta}) \leq \mathcal{L}(x, \widehat{\alpha}, \widehat{\beta}) \tag{A.34}$$

for all $x \in X$, $\alpha \geq 0$ and $\beta \in \mathbb{R}^p$. So a saddle point is a minimum of the Lagrangian with respect to x and a maximum with respect to α and β. Observe that by writing out the Lagrangian we obtain

$$\begin{aligned} f(\widehat{x}) + \sum_{i=1}^{m} \alpha_i g_i(\widehat{x}) + \sum_{j=1}^{p} \beta_j h_j(\widehat{x}) &\leq f(\widehat{x}) + \sum_{i=1}^{m} \widehat{\alpha}_i g_i(\widehat{x}) + \sum_{j=1}^{p} \widehat{\beta}_j h_j(\widehat{x}) \\ &\leq f(x) + \sum_{i=1}^{m} \widehat{\alpha}_i g_i(x) + \sum_{j=1}^{p} \widehat{\beta}_j h_j(x) \end{aligned}$$

By the first inequality we have for all $\alpha \geq 0$ and $\beta \in \mathbb{R}^p$.

$$\sum_{i=1}^{m} \alpha_i g_i(\widehat{x}) + \sum_{j=1}^{p} \beta_j h_j(\widehat{x}) \leq \sum_{i=1}^{m} \widehat{\alpha}_i g_i(\widehat{x}) + \sum_{j=1}^{p} \widehat{\beta}_j h_j(\widehat{x}) \tag{A.35}$$

The second inequality gives us for all $x \in X$

$$f(\widehat{x}) + \sum_{i=1}^{m} \widehat{\alpha}_i g_i(\widehat{x}) + \sum_{j=1}^{p} \widehat{\beta}_j h_j(\widehat{x}) \leq f(x) + \sum_{i=1}^{m} \widehat{\alpha}_i g_i(x) + \sum_{j=1}^{p} \widehat{\beta}_j h_j(x) \tag{A.36}$$

We are now ready to introduce the KKT Theorem.

Theorem A.5 (Karush-Kuhn-Tucker). *Let the convex optimization problem given by Problem A.1 satisfy the Slater condition. Then $\widehat{x}$ is a solution of this problem if and only if there exist $\widehat{\alpha} \geq 0$ and $\widehat{\beta} \in \mathbb{R}^p$, such that $(\widehat{x}, \widehat{\alpha}, \widehat{\beta})$ is a saddle point of the Lagrangian.*

Proof. First we assume that $(\widehat{x}, \widehat{\alpha}, \widehat{\beta})$ is a saddle point. Therefore Equation A.35 holds. Observe that the right hand side of this equation is fixed. If $g_i(\widehat{x}) > 0$ or $h_i(\widehat{x}) \neq 0$ for some i we can take $\alpha_i \to \infty$ or $\beta_j \to \pm\infty$ and let the left hand side go to infinity. This is not possible, so $g_i(\widehat{x}) \leq 0$ or $h_j(\widehat{x}) = 0$ for all i and j, i.e. $\widehat{x}$ is feasible. Furthermore, observe that by taking α and β both equal to 0, we see that the left hand side equals 0. Therefore we must have $\sum_{i=1}^m \widehat{\alpha}_i g_i(\widehat{x}) = 0$, using $h_j(\widehat{x}) = 0$ and $g_i(\widehat{x}) \leq 0$. Therefore, we can rewrite Equation A.36

$$f(\widehat{x}) \leq f(x) + \sum_{i=1}^m \widehat{\alpha}_i g_i(x) + \sum_{j=1}^p \widehat{\beta}_j h_j(x) \tag{A.37}$$

For feasible x the second term is at most 0 and the third term equals 0. Therefore we see that $\widehat{x}$ is a minimizer of the optimization problem

$$f(\widehat{x}) \leq f(x) \quad \text{for all } x \in \mathcal{F} \tag{A.38}$$

Now assume that we have a solution $\widehat{x}$ of the optimization problem. We introduce the following system

$$\begin{aligned} f(x) &< f(\widehat{x}) \\ g_i(x) &\leq 0 \quad \text{for all } 1 \leq i \leq m \\ h_j(x) &= 0 \quad \text{for all } 1 \leq j \leq p \\ x &\in X \end{aligned} \tag{A.39}$$

Clearly, by assumption this system has no solution. By Farkas' Lemma (see Corollary A.3), we see that there has to exist $\widehat{\alpha} \geq 0$ and $\widehat{\beta} \in \mathbb{R}^p$ such that for all $x \in X$

$$f(x) + \sum_{i=1}^m \widehat{\alpha}_i g_i(x) + \sum_{j=1}^p \widehat{\beta}_j h_j(x) \geq f(\widehat{x}) \tag{A.40}$$

If we take $x = \widehat{x}$ in the above inequality, we obtain $\sum_{i=1}^m \widehat{\alpha}_i g_i(\widehat{x}) + \sum_{j=1}^p \widehat{\beta}_j h_j(\widehat{x}) \geq 0$. Together with the fact that x is feasible for the problem we have

$$\sum_{i=1}^m \widehat{\alpha}_i g_i(\widehat{x}) + \sum_{j=1}^p \widehat{\beta}_j h_j(\widehat{x}) = 0 \tag{A.41}$$

Therefore $f(\widehat{x}) = \mathcal{L}(\widehat{x}, \widehat{\alpha}, \widehat{\beta})$. Observe that for all $\alpha > 0$ and $\beta \in \mathbb{R}^p$

$$\begin{aligned} \mathcal{L}(\widehat{x}, \alpha, \beta) &= f(\widehat{x}) + \sum_{i=1}^m \alpha_i g_i(\widehat{x}) + \sum_{j=1}^p \beta_j h_j(\widehat{x}) \\ &\leq f(\widehat{x}) = \mathcal{L}(\widehat{x}, \widehat{\alpha}, \widehat{\beta}) \end{aligned} \tag{A.42}$$

since $\widehat{x}$ is a feasible point for the optimization problem. Also by Equation A.40 we have for all $x \in X$

$$\mathcal{L}(x, \widehat{\alpha}, \widehat{\beta}) \geq f(\widehat{x}) = \mathcal{L}(\widehat{x}, \widehat{\alpha}, \widehat{\beta}) \tag{A.43}$$

Combining Equations A.42 and A.43 we see that $(\widehat{x}, \widehat{\alpha}, \widehat{\beta})$ is a saddle point, which closes the proof of the theorem. $\square$

We can use the KKT Theorem to prove the following useful corollary

Corollary A.6. *Let the convex optimization problem given by Problem A.1 satisfy the Slater condition. Then $\widehat{x}$ is a solution of this problem if and only if there are $\widehat{\alpha} \geq 0$ and $\widehat{\beta} \in \mathbb{R}^p$ such that $(\widehat{x}, \widehat{\alpha}, \widehat{\beta})$ satisfies the KKT conditions.*

Proof. By the KKT Theorem, $\widehat{x}$ is a solution of the problem if and only if there exist $\widehat{\alpha} \geq 0$ and $\widehat{\beta} \in \mathbb{R}^p$ such that $(\widehat{x}, \widehat{\alpha}, \widehat{\beta})$ is a saddle point of the Lagrangian.

Now assume that $(\widehat{x}, \widehat{\alpha}, \widehat{\beta})$ satisfies the KKT conditions. From the first condition of Equation A.6 we have $\nabla_x \mathcal{L}(\widehat{x}, \widehat{\alpha}, \widehat{\beta}) = 0$, i.e. $(\widehat{x}, \widehat{\alpha}, \widehat{\beta})$ is a minimizer of the Lagrangian with respect to x. Therefore $\mathcal{L}(\widehat{x}, \widehat{\alpha}, \widehat{\beta}) \leq \mathcal{L}(x, \widehat{\alpha}, \widehat{\beta})$ for all $x \in X$. Using the fact that $\widehat{x}$ is feasible, we see $g_i(\widehat{x}) \leq 0$ for all i and $h_j(\widehat{x}) = 0$. Then for all $\alpha \geq 0$ we have $\sum_{i=1}^m \alpha_i g_i(\widehat{x}) \leq 0$, and therefore the following holds

$$\begin{aligned}
\mathcal{L}(\widehat{x}, \alpha, \beta) &= f(\widehat{x}) + \sum_{i=1}^m \alpha_i g_i(\widehat{x}) + \sum_{j=1}^p \beta_j h_j(\widehat{x}) \\
&\leq f(\widehat{x}) = f(\widehat{x}) + \sum_{i=1}^m \widehat{\alpha}_i g_i(\widehat{x}) + \sum_{j=1}^p \widehat{\beta}_j h_j(\widehat{x}) \\
&= \mathcal{L}(\widehat{x}, \widehat{\alpha}, \widehat{\beta})
\end{aligned}$$

for all $\alpha \geq 0$ and $\beta \in \mathbb{R}^p$, using the second condition of Equation A.6. Therefore $(\widehat{x}, \widehat{\alpha}, \widehat{\beta})$ is a saddle point of the Lagrangian.

Now assume that $(\widehat{x}, \widehat{\alpha}, \widehat{\beta})$ is a saddle point of the Lagrangian. Clearly, since $\widehat{x}$ is a solution, it is feasible. Using the fact that $\widehat{x}$ minimizes the Lagrangian with respect to x we see that $\nabla_x \mathcal{L}(\widehat{x}, \widehat{\alpha}, \widehat{\beta}) = 0$, thus giving the first condition of Equation A.6. Assume that for some i we have $\widehat{\alpha}_i g_i(\widehat{x}) \neq 0$. Since $\widehat{x}$ is feasible and $\widehat{\alpha} \geq 0$ we have $g_i(\widehat{x}) < 0$ and $\alpha_i > 0$. Note that then there exists an $\overline{\alpha} \geq 0$ such that $\widehat{\alpha}_i g_i(\widehat{x}) \leq \overline{\alpha}_i g_i(\widehat{x}) = 0$, namely if $\overline{\alpha}_i = 0$. If we take all other elements of $\overline{\alpha}$ equal to the elements of $\widehat{\alpha}$ we see

$$\sum_{i=1}^m \widehat{\alpha}_i g_i(\widehat{x}) \leq \sum_{i=1}^m \overline{\alpha}_i g_i(\widehat{x}) \tag{A.44}$$

If we take $\overline{\beta}_j = \widehat{\beta}_j$ for all j we have

$$\sum_{i=1}^m \widehat{\alpha}_i g_i(\widehat{x}) + \sum_{j=1}^p \widehat{\beta}_j h_j(\widehat{x}) \leq \sum_{i=1}^m \overline{\alpha}_i g_i(\widehat{x}) + \sum_{j=1}^p \overline{\beta}_j h_j(\widehat{x}) \tag{A.45}$$

which is a contradiction with Equation A.36. Therefore $(\widehat{x}, \widehat{\alpha}, \widehat{\beta})$ satisfies the KKT conditions. $\square$