Supervised machine learning algorithms for the estimation of the probability of default in corporate credit risk

Eduard Sariev

Supervisors: Dr Guido Germano, Prof. Philip Treleaven

A DISSERTATION SUBMITTED IN PARTIAL FULFILMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
OF THE
UNIVERSITY COLLEGE LONDON

Department of Computer Science
University College London

February 19, 2021

Declaration

I, Eduard Sariev, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

.....
Eduard Sariev

Acknowledgements

Pursuing a PhD is a challenging and difficult task. I would like to overstate my gratitude to my PhD supervisors, Dr. Guido Germano and Prof. Philip Treleaven. They made this research possible with their friendly support, patience and technical expertise.

Abstract

This thesis investigates the application of non-linear supervised machine learning algorithms for estimating Probability of Default (PD) of corporate clients. To achieve this, the thesis is separated into three different experiments:

1. The first experiment investigates a wrapper feature selection method and its application on the support vector machines (SVMs) and logistic regression (LR). The logistic regression model is the most popular approach used for estimating PD in a rich default portfolio. However, other alternatives to PD estimation are available. SVMs method is compared to the logistic regression model using the proposed feature selection method.

2. The second experiment investigates the application of artificial neural networks (ANNs) for estimating PD of corporate clients. In particular ANNs are regularized and trained both with classical and Bayesian approach. Furthermore, different network architectures are explored and specifically the Bayesian estimation and regularization is compared to the classical estimation and regularization.

3. The third experiment investigates the k-Nearest Neighbours algorithm (KNNs). This algorithm is trained using both Bayesian and classical methods. KNNs could be efficiently applied to estimating PD. In addition, other supervised machine learning algorithms such as Decision trees (DTs), Linear discriminant analysis (LDA) and Naive Bayes (NB) were applied and their performance summarized and compared to that of the SVMs, ANNs, KNNs and logistic regression.

The contribution of this thesis to science is to provide efficient and at the same time applicable methods for estimating PD of corporate clients. This thesis contributes to the existing literature in a number of ways.

1. First, this research proposes an innovative feature selection method for SVMs.

2. Second, this research proposes an innovative Bayesian estimation methods to regularize ANNs.

3. Third, this research proposes an innovative Bayesian approaches to the estimation of KNNs.

Nonetheless, the objective of the research is to promote the use of the Bayesian non-linear supervised machine learning methods that are currently not heavily applied in the industry for PD estimation of corporate clients.

Impact Statement

The impact of this research is twofold. The first area of impact has mainly an academic focus. The quantitative methods used in the study cover extensively and in detail the theoretical aspect of supervised machine learning methods. In particular the research proposes innovative estimation algorithms that can be applied to estimate the model parameters of selected supervised machine learning algorithms such as neural networks and k-nearest neighbours. The application and benefits of the Bayesian approach to estimating model parameters is covered in detail. The research can be a solid starting point for other researches in the field of supervised machine learning to explore the most efficient and at the same time simple model to classify between performing and non-performing clients. The second area of impact has mainly managerial focus. The proposed quantitative models can support managers at various lending institution to optimize the lending process by employing efficient methods to accurately identify non-performing clients. In particular managers involved in corporate lending can benefit from the conducted research. Major financial drivers have been identified that contribute significantly to the probability of default. Managers have the opportunity to leverage on these default drivers and effectively monitor corporate borrowers. As a consequence this will lead to cost optimization of the lending process. Overall the impact of the research conducted covers both academic topics and topics relevant for practitioners. The work has the potential to bridge the gap between purely theoretical issues and those stemming from the industry.

Contents

1 Introduction

This chapter presents an overview of the thesis. First, explaining the motivation for this research by introducing the research topic and relevant works that have been done so far in this field. Subsequently, the chapter describes the objective of the research: estimating PD models for corporate credit risk using non-linear supervised machine learning algorithms.

1.1 Motivation

The last two decades have seen a rapid growth in both the availability and the use of credit. Until recently, the decision to grant credit was based on human judgement to assess the risk of default. The growth in the demand for credit, however, has led to a rise in the use of more formal and objective methods (generally known as credit scoring) to help credit providers decide whether to grant credit to an applicant. This approach, first introduced in the 1940s, has evolved over the years and developed significantly. In recent years, the progress in credit scoring was fuelled by increased competition in the financial industry, advances in computer technology, and the exponential growth of large databases. All these facts serve as a motivation for researches to analyse and apply new methods for credit scoring and mainly for estimating PD. Therefore, the rapid development of machine learning algorithms in the last two decades can significantly improve the credit scoring models and in particular the PD estimation of corporate entities (whole-sale clients). The estimation of the PD for the retail clients, differs from that of the whole sale clients Pluto and Tasche (2011). Many authors investigate the low default portfolio problem as a major obstacle in estimating the PD of wholesale clients Pluto and Tasche (2011). One of the main motivation of this research is to focus not on the low default portfolio problem but to examine another problem that is far more difficult to tackle than the low default portfolio issue. Given the huge data sets existing today, collecting data on defaulted wholesale clients is not such a major problem as it was decades ago. Credit rating agencies such as Moody's and S&P have acquired and created huge and accurate databases containing default information for corporate clients. Additionally, global top-tier international banks have also developed sophisticated data infrastructures that allow them to correctly monitor corporate defaults. Nonetheless, the problem of interaction between default drivers and default indicators is of vital importance to the PD estimation of wholes-sale clients. By interaction between risk drivers and default indicators, we mean the misalignment between risk drivers and default indicators. For example, corporate clients almost always disclose financial statements that lack any obvious liquidity or capital issues. Moreover, qualitative information for corporate clients such as prestige and quality of management, market position and operational structure is typically subjective and biased as well. All

these facts present a challenge for developers of PD models to properly capture the default patterns that are typical for retail/individual obligors. For instance, an individual that has defaulted on his debt, normally has either a bad credit history or his income potential has diminished significantly. The credit bureau companies collect accurate and timely data on the default history of individual clients. Therefore, the use of linear classification method such as logistic regression had become so popular for retail PD estimation. The ease of interpretation of linear logistic regression combined with its classification accuracy made possible the development of highly accurate PD models for individual clients Bailey (2006). However, the motivation of this research is to provide evidence and justification that through the use of Bayesian non-linear supervised machine learning algorithms such as SVMs, ANNs and KNNs, the hidden defaults patterns inherent to corporate clients could be more easily captured than by using linear classification methods. The motivation of this research is driven by the fact that a combination of different financial factors/ratios can be a key to properly identify the default risk of corporate clients Simkovic and Kaminetzky (2011). In order to capture interactions among key financial factors we will use non-linear machine learning methods that are capable of capturing the data non-linearity.

1.2 Research objectives

The research objective of the thesis is to offer alternatives to the PD estimation of corporate entities by using non-linear supervised learning methods. Although logistic regression is extensively used by financial institutions for predicting probability of default (PD), non-parametric models exist that allow a developer to achieve a higher accuracy in predicting PD. An important objective of this research is to provide an extensive set of supervised algorithms with different estimation and regularization approaches to improve the PD estimation of corporations. Moreover, this research investigates the impact of Bayesian estimation on the PD of corporate clients. The benefits of using Bayesian statistics to estimate PD are several. The problem of over-fitting the training set could be addressed by using the Bayesian approach. The problem of finding optimal parameters and thus improving the classification performance could also be addressed by the application of Bayesian statistics. Overall the objective of the research is to expand the current methodology on corporate PD estimation. Each method for PD estimation has its own merit and should be considered by practitioners. However, some methods are more appropriate than others for certain type of portfolios.

1.3 Research experiments

1.3.1 Experiment one

SVMs have been extensively used for classification problems in many areas such as gene, text and image recognition. However, SVMs have been rarely used for probability of default (PD) estimation in credit risk. In this experiment we propose a wrapper selection method for SVMs that is applicable to estimate the PD. The feature selection method is based on the distance of the support vectors from the separating hyperplane and on the number of support vectors. In order to assess the applicability of this feature selection method for SVMs, we compare the classification performance of SVMs and logistic regression (LR). We also make a comparison with other feature selection methods for SVMs. The results show that the proposed feature selection method is able to identify relevant features for both SVMs and LR. The data on which the variable selection method for SVMs are applied consist of annual financial statements of medium retail companies head-quartered in Eastern Europe, Poland and the method is also applied to German retail data.

1.3.2 Experiment two

ANNs have been extensively used for classification problems in many areas such as gene, text and image recognition. Although ANNs are popular methods for probability of default (PD) estimation in credit risk, ANNs have their own drawbacks that should be addressed. One major drawback of ANNs is the tendency to over fit the data. In this experiment we propose an improvement of a Bayesian estimation and regularization approach to train the ANNs. The improved Bayesian estimation and regularization is compared to the classical back-propagation algorithm for training a feed-forward network. In order to assess the applicability of the Bayesian regularization, different network architectures were investigated. Furthermore, the over fitting process was controlled by monitoring the error on the test data, while training the network on the development data set. We call this process early stopping. The results show that the applied Bayesian regularization method is able to produce a good classification accuracy by not introducing bias to the regularization parameters. The results also indicate the different sensitivity of the classical regularization and the Bayesian regularization to the early stopping procedure. The data on which the regularization algorithm is applied consist of annual financial statements of medium retail companies head-quartered in Eastern Europe, Poland and the method is also applied to German retail data.

1.3.3 Experiment three

The k-nearest neighbours (KNNs) method is extensively used for classification problems in many areas. This paper proposes a Bayesian estimation approach to train KNNs (BKNNs). The Bayesian estimation averages over the number of nearest neighbours and therefore allows to avoid the problem of specifying the number of neighbours. A Genetic algorithm (GA) and three innovative MCMC upgrades are proposed and utilised for the estimation of BKNNs. Furthermore, a family of linear and non-linear supervised methods is applied to the data and compared to BKNNs. Three data sets are used to test the classification methods: first consists of annual financial statements of East-European corporate obligors, second of Polish corporate obligors and third of German retail clients. The results show that the BKNNs with GA estimation is able to generate a reasonable default classification accuracy, when compared to other classification methods.

1.3.4 Scientific contribution

The contribution of this thesis to science is to provide efficient and at the same time applicable methods for estimating PD of corporate clients. The thesis contributes to the existing literature in a number of ways.

1. First, this research proposes an innovative feature selection method for SVMs.
2. Second, this research proposes an innovative Bayesian estimation methods to regularize ANNs.
3. Third, this research proposes an innovative Bayesian approaches to the estimation of KNNs.

Nonetheless, the objective of the research is to promote the use of the Bayesian non-linear supervised machine learning methods that are currently not heavily applied in the industry for PD estimation of corporate clients.

1.3.5 Thesis structure

This work is organized in several chapters. Each of the chapters gives information about a specific part of the work done. The chapters are listed below:

- Chapter 1 "Introduction" introduces the main research objective of the work, its motivations and its contribution to literature.
- Chapter 2 "Literature review" gives on overview of the works devoted to PD estimation of both retail and wholesale portfolios. The idea of this chapter is to provide a comprehensive description of the ideas and methods applied to PD estimation.

- Chapter 3 "Exploratory data analysis" provides information about the data used in the three experiments. Here a preliminary analysis on the data is performed. The goal of this chapter is to provide a high level descriptive statistics on the data used.
- Chapter 4 "Experiment results" contains the application of the supervised machine learning methods to the data. This chapter is divided into sub-chapters based on the experiments conducted.
- Chapter 5 "Conclusions" contains the conclusions drawn from the application of the models. This chapter explains whether the research objective has been achieved. Future areas of investigation are discussed and additional research questions are raised for further analysis.

2 Literature review

Financial organizations store, collect and distribute huge quantities of real time data. The problem of harnessing the power of hidden data patterns is of a vital importance to each financial organization. The PD estimation is by nature a data mining problem, which could be summarized as a sequence of actions aimed at finding useful relationships in the data Thuraisingham (1999). Recently, data mining serves as a pillar of many fields spanning from image recognition and digit recognition to credit scoring and customer segmentation Kamath (2009).
Here we provide a brief qualitative overview of the most popular data mining methods, focusing on their advantages and disadvantages.

2.1 Linear methods

- LDA is linear method used for classification. It was the first method applied to financial data in 1960s Altman (1968). LDA is a plug in classification method, where two normal distributions are compared to each other. A major assumption is that the distributions have the same covariance structure and that structure further meets additional requirements such as the covariance structure matrix has to be Hermitian. Given these assumptions, the LDA is able to classify between two classes. There exists a slightly different version of the LDA called "the Fisher discriminant analysis". It does not require the two distributions to be normal. An advantage of LDA is simplicity and ease of use. A disadvantage of the LDA is that it requires the independent variables to be numerical (if they are categorical then Canonical discriminant analysis should be used). Another disadvantage of LDA is that it relies on many assumptions about the distributions of the variables. In practise these assumptions are usually not met.

- LR belongs to the family of generalized linear models Cox (1958). It is the most popular approach used for estimation of PD. The link function of LR is the logit function, which allows to estimate a response that ranges from 0 to 1. There are extensions to the LR model enabling the classification of more than two classes. An advantage of the LR is its simple parametric form, allowing the LR to be applied in many different fields. LR is also straightforward to estimated, meaning the method does not take a lot of computational time. A disadvantage of LR is its ability to easily overfit the data.

- NB is a probabilistic method. The Bayes theorem is the building block of the NB method Hand and Yu (2001). One main assumption of NB is the conditional inde-

pendence of observations. One advantage of the method is its simplicity and use of computations. The main disadvantage of NB is its assumption on the conditional independence of the observations. This assumption is rarely met in practice.

2.2 Non-linear methods

2.2.1 SVMs

SVMs method maximizes the distance between two planes Theodoridis Sergios (2009). The observations that follow from either side of the separating hyper-plane are allocated to class 1 and class 2 respectively. The hyper-plane that separates the classes is based on some of the observations. These special observations are called support vectors. The original version of the SVMs method was applicable to linear classification only. However, it was extended to non-linear classification through the use of the kernel trick. The kernel allows to transfer into n-dimensional space where the observations can be linearly separable. Additionally, soft-margin is allowed which means some of the observations are not classified correctly but this is reflected in the loss function by introducing additional penalty to it. An extension of SVMs method exists where the SVMs optimization problem is formulated differently (least square SVMs). Moreover, the SVMs concept can be reformulated and used for regression problems as well. The main advantage of SVMs method is its built-in regularization feature and its flexibility trough the use of kernels. The main disadvantage is the computational difficulties that arise when the training data are huge.

SVMs can be used in calculating bank's capital requirements stipulated by the introduction of the Basel III guidelines (BCBS, 2017). The new capital requirements that banks must meet have established the necessity of an accurate risk assessment. The probability of default (PD) measure is a key estimate not only for risk assessment, but also for impairment purposes under the changes introduced by International Financial Reporting Standard 9 (IFRS9) (Onali and Ginesti, 2014). Accurate PD assessment is vital for decreasing the cost of capital (Gavalas, 2015). The estimation of PD has been a topic of extensive research for many years. A high number of different algorithms have been used to estimate the PD: ANNs, DTs, LDA, SVMs, LR. Harris (2015) provides a good general explanation of these methods. However, LR remains the most widely used PD estimation method for both corporate and retail borrowers.

Extensive research has been conducted comparing several PD estimation methods. Meyer et al. (2003) compared SVMs to 25 other methods used for PD estimation. They found that although the performance of the SVMs model is good, other methods such as ANNs

and DTs sometimes outperform SVMs. In a more general study, Mukherjee (2003) used SVMs and LR to classify traded companies on the Greek stock exchange, showing that SVMs classification was better, still without focus on the feature selection process. Another comparison between SVMs and ANNs was made by (Li et al., 2006). They showed that the SVMs model slightly outperforms ANNs and the SVMs model needs fewer features than ANNs to achieve maximum classification performance. Huang et al. (2007) compared SVMs with ANNs, genetic programming, and DTs. In this comparison the feature selection process was covered, but the LR model was not used as a comparison. Bellotti and Crook (2009b) compared LR and SVMs, but without showing the feature selection method for the LR. Bellotti et al. (2011) compared LR with SVMs, but for regression purposes, not for classification. They found that the SVMs model outperforms LR. Furthermore, Chen et al. (2011) compared LR and SVMs with regard to the feature selection process. However, the features selected for the SVMs were automatically used for LR and this way the comparison was biased toward the SVMs model: as expected, the SVMs model outperformed the LR in this case. Hens and Tiwari (2012) again focused on the comparison of SVMs with genetic programming without including LR. Lessmann et al. (2015) found that SVMs and ANNs perform better, but the performance of the LR is still relatively good. Finally, Harris (2015) compared SVMs to LR. Although this study used LR as the only alternative to SVMs, a lot of the details of this comparison were not shared; for instance, the feature selection for both models is not covered at all.

The feature selection process for SVMs is a key step in comparing SVMs to other algorithms. The existing literature indicates that some research on SVMs feature selection has been developing recently. Weston et al. (2000) proposed a method that is based upon finding those features which minimize bounds on the leave-one-out error. They show that their method is superior to some standard feature selection algorithms. Guyon and Elisseeff (2003) provided a good high-level overview of the different feature selection algorithms available in the literature. Rakotomamonjy (2003) proposed relevance criteria derived from SVMs that are based on a weight vector. He showed that the criterion based on the weight vector derivative achieves good results and performs consistently well. Chen and Lin (2006) combined SVMs and various feature selection strategies. Some of them were filter-type approaches, i.e., general feature selection methods independent of the SVMs, and some were wrapper-type methods, i.e., modifications of the SVMs which can be used to select features. Recently, variable and feature selection has become the focus of much research. Becker et al. (2009) investigated a penalized version of SVMs for feature selection. They argued that keeping a high number of features could avoid overfitting if the performance function uses an L_1 norm regularization. Huang and Huang (2010) investigated a recursive feature selection scheme in SVMs. Their results

have indicated that one-vs-one SVMs with embedded recursive feature selection outperforms other multi-class SVMs. In this context, Kuhn and Johnson (2013) presented a generalized backward feature elimination procedure for selecting a final combination of features.

2.2.2 ANNs

ANNs are a generative method. They utilize the non-linearity of the data by applying a special type of activation functions such as logistic or sigmoid functions Bishop (1995). Particularly, the derivative of these function has a very special form that allows to find an analytical solution of the optimization problem. Normally ANNs are of a feed-forward type with one or more hidden layers. The loss function of the network is usually either mean square error (MSE) or entropy function. A main advantage of ANNs is that they can solve a range of different problems such as: regression, classification and function approximation. ANNs can be applied to time series problems as well. ANNs also allow for different estimation algorithms starting from the original one: the back-propagation and going to Bayesian estimation methods. ANNs are capable of capturing the non-linearity of data by adjusting the number of neurons and hidden layers. A disadvantage of ANNs is their ability to over-fit data and ANNs training does not guarantee a global solution. To maximize the performance, ANNs need more training data then compared to other algorithms.

ANNs can be used for scoring obligors in the context of credit risk. Credit scoring became popular in the USA during the 1950s. The boosting economy in that decade and the next required the need for accessible credit and it was during this period when the methods used for automated credit scoring became more advanced. In fact the first scoring model was presented by Fisher (1936) who applied linear discriminant analysis (LDA) as a discrimination and classification technique; this was followed by Durand (1941) who possibly was the first to use multiple discriminant analysis to examine car loan applications.

Among the many options offered and investigated in the literature for credit scoring, ANNs are a flexible and rich concept to solve not only classification problems but also to offer solutions to clustering, time series and function approximation problems (Bell, 2015). The flexibility of ANNs inspired researchers to investigate their applicability to classification tasks. Recently, an extensive research has been conducted to utilize and apply ANNs for corporate credit scoring given the large amount of financial data collected. The studies of Heaton et al. (2017) and Pérez-Martín et al. (2018) advocate for extensive use of ANNs with many layers, the so-called deep learning approach. Furthermore,

Bonini and Caivano (2018) showed that artificial intelligence methods including ANNs outperform traditional statistical methods. Nonetheless the performance advantages of ANNs were questioned by Addo et al. (2018) and Kalaycıet al. (2018), who showed that ANNs underperform when compared to decision trees and logistic regression respectively.

Here we focus on the overfitting issue of ANNs. Several recent studies have been devoted to this problem. Zhang et al. (2018) investigated various ways of detecting overfitting in ANNs and advocated splitting the data into training and validation as a main way of dealing with overfitting. Using a genetic algorithm, (Nicolae-Eugen, 2016) prevented ovefitting in an ANN by encoding the weights of the ANN into binary chromosomes and applying high-probability mutation in the genetic algorithm. A different approach to reduce overfitting was proposed by Vincent et al. (2010), who applied a drop-out strategy combined with a stacked denoising autoencoder; they found that this strategy outperforms a single drop-out strategy and is computationally more efficient. One of the reasons for overfitting is the noise in the training data. In that context Hindi and Al-Akhras (2011) recommended to smooth the decision boundaries by eliminating border instances from the training set before training an ANN; this is achieved by using a variety of instance reduction techniques.

In contrast to these studies on overfitting in ANNs, we take a Bayesian approach to solve the issue. Bayesian estimation in an ANN has become applicable only since the advancement of computational power has increased enough. Initially Bayesian learning in ANNs was used to create an optimal network architecture. For example, Neal (1992) explored the difficulties related to the selection of the prior knowledge as well as the problems associated with the computation of the posterior distribution. Neal (1996) studied the effect of using different priors for the estimation of the network weights. Rasmussen (1996) investigated how to estimate the weights of a network using dynamic simulation. Furthermore, Lampinen and Vehtari (2001) applied ANNs with Bayesian learning to regression and classification. Titterington (2004) reviewed the various approaches taken to determine the network architecture, involving the use of Gaussian approximations and of non-Gaussian but deterministic approximations called variational approximations. The Bayesian estimation of an ANNs for credit scoring implies that the optimal architecture of the neural network is important to the performance because the architecture greatly impacts the estimation efficiency of the network (Heaton et al., 2017). However, in this thesis we focus on the Bayesian regularization of the network in order to avoid overfitting. We compare our approach to the classical regularization approach examined in Ashiquzzaman et al. (2017).

2.2.3 KNNs

KNNs method is a local non-linear supervised machine learning method that can be used for both regression and classification Altman (1992). The method is based on finding the neighbouring observations to the observation under investigation. The neighbours are determined by using a distance metric. The label of the observation is determined by the label of the majority of the neighbours. The advantage of KNNs method is that it is not dependent on parametric and probabilistic assumptions that are normally difficult to meet. The disadvantage of the method is it works like a black box that is hugely affected by the number of neighbours and the distance metric applied to the observations.

As discussed in 2.2.2 credit scoring emerged in the 1950s and the question of automating the credit assessment process became critical to the financial industry. Along with artificial neural networks (ANNs) Sariev and Germano (2019, 2020), k-nearest neighbours (KNN) is one of the most popular and flexible concepts to solve classification problems (Gök, 2015; Henley and Hand, 1996). Recently, extensive research has been conducted to utilize and apply KNNs for corporate credit scoring. Most of this research is oriented towards comparing KNNs to other classification algorithms with the aim of proving that KNNs achieves a similar performance as that of other popular scoring methods such as logistic regression, decision trees, SVMs, and many other methods coming from survival analysis and operational research (Gaganis et al., 2007).

Currently, several of the above methods have been applied to estimate the probability of default (PD) (Abdou et al., 2016), including time-series methods (Agosto et al., 2016). Antonakis and Sfakianakis (2009) found that classical KNN perform well when compared to other methods. The flexibility of KNNs inspired researchers to investigate its applicability to credit risk classification tasks (Huang and Wu, 2011). Brown and Mues (2012) investigated the performance of KNNs on imbalanced data, showing that it is lower than that of random forest classifiers. Faez et al. (2014) placed KNNs somewhere in the middle in terms of accuracy when compared to other classification algorithms. The overall performance of classical KNNs depends on many factors including data transformations, feature selection, performance indicator, etc.

The growth of computational power allowed the use of Bayesian estimation for KNNs. Bayesian learning in KNNs offers a solution to determine an optimal number of neighbours. Holmes and Adams (2002) explored the difficulties related to the selection of the prior knowledge as well as the problems associated with the computation of the posterior distribution. They also studied the effect of using different priors for the estimation of KNNs. Everson and Fieldsend (2004) introduced a variable metric extension to the probabilistic KNNs classifier, which permits averaging over all rotations and scalings of

the data. The results from synthetic data showed that BKNNs provide good classification accuracy. Manocha and Girolami (2007) concluded that there is no outright performance advantage of BKNNs over KNNs. Nonetheless, the main advantage of BKNNs is methodological. BKNNs provides continuous predictive probabilities in a natural manner which gives a way the need of allocating uneven miss-classification costs and further propagating these levels of predictive uncertainty as a part of further possible downstream processing. Furthermore, Villa et al. (2008) used a slightly different version of the original KNNs. The KNNs method was combined with a bootstrap procedure to provide the posterior probability of a newly classified object. He also found that the bootstrapped KNNs performs better than the classical KNNs. Su et al. (2008) also investigated the BKNNs and presented some evidence to show that BKNNs still significantly underestimates model uncertainty. They reasoned that BKNNs is unable to account for the uncertainty in the spatial locations of the neighbours. Guo and Chakraborty (2010) investigated a practical approach based on the BKNNs. In their work, the shape of the neighborhood is automatically selected according to the concentration of the data around each query point with the help of discriminants. Liu et al. (2013) applied KNNs with Bayesian learning on simulated and real data sets in order to evaluate the performances of the BKNNs. They found that the BKNNs outperforms the classical KNNs. Yoon and Friel (2015) applied the integrated nested Laplace approximation instead of MCMC to estimate the BKNNs. They concluded that the Laplace approximation resulted in an acceptable accuracy when compared to that of the MCMC application.

2.2.4 DTs

DTs are a very popular non-linear supervised machine learning algorithm. Typically a DT consists of a parent node, intermediate nodes and terminating nodes Quinlan (1999). Different algorithm for training DTs exist. One of the most popular are ID3, C4.5, CART, CHAID, MARS. To improve DTs' performance, they are combined with ensemble methods such as: Boosting, Random forests and Bagging. DTs can also be pruned to improve the classification accuracy. DTs can be used for both classification and regression. An advantage of DTs method is they are simple to understand and are easily combined with other methods. A disadvantage is they favour variables with many categories and therefore they can overfit the data set.

In addition to the above described models, linear/non-linear mathematical programming and survival analysis are also used for PD estimation. However, they are out of the scope for this thesis.

A topic of interest is the problem of identifying the best classifier among a family of classifiers. In principal there is no single winner but clearly the non-linear supervised machine algorithms are considered to have higher classification accuracy Conway and White (2012). However, the classification accuracy is only one of the aspects that should be considered when a selection of final models is to be made. Another point of consideration is the model interpretation. If a model is not well understood by model users then the model risk increases which can negatively affect the business Black (2004). The time to train models is another aspect that has to be considered. Some models take more time to be computed than others. The infrastructure where the model is implemented plays an important role as well. Sometimes models that are easily integrated are the one that are chosen as final. Nonetheless, the choice of a model is dependent on the data structure and quality, the transformations used on the data and the variable selection method. In summary, we need to highlight model selection is a subjective process. Despite that subjectivity, model selection has to be done in a way that minimizes the bias and emphasises the model quantitative performance.

3 Exploratory data analysis

Three main data set have been used in this thesis.

3.1 East-European data set

The data set contains information for 7996 observations on 34 independent variables (covariates or features) and on one binary target variable, which indicates whether a default occurred one year after the issue of the financial statement. The 34 variables are constructed based on data from the entity's financial statements. The variables are split into several groups and further analysed. The data are on an annual basis from the period 2007–2012. As it usually happens, the financial statements contain missing values. In order to tackle this problem a detailed analysis is performed on the missing patterns in the data and finally a multiple chain imputation method (Abayomi et al., 2008) is used for the East-European data, see Table 27 in Appendix A, which present the descriptive statistics before and after imputation. A sigmoid transformation is applied to all the covariates, thus bounding the covariates' value between 0 and 1 (Han and Moraga, 1995). This is a typical approach applied to variables before using them for classification purposes.The data set is not publicly available, but the author can share the data set if requested.

3.2 Polish data set

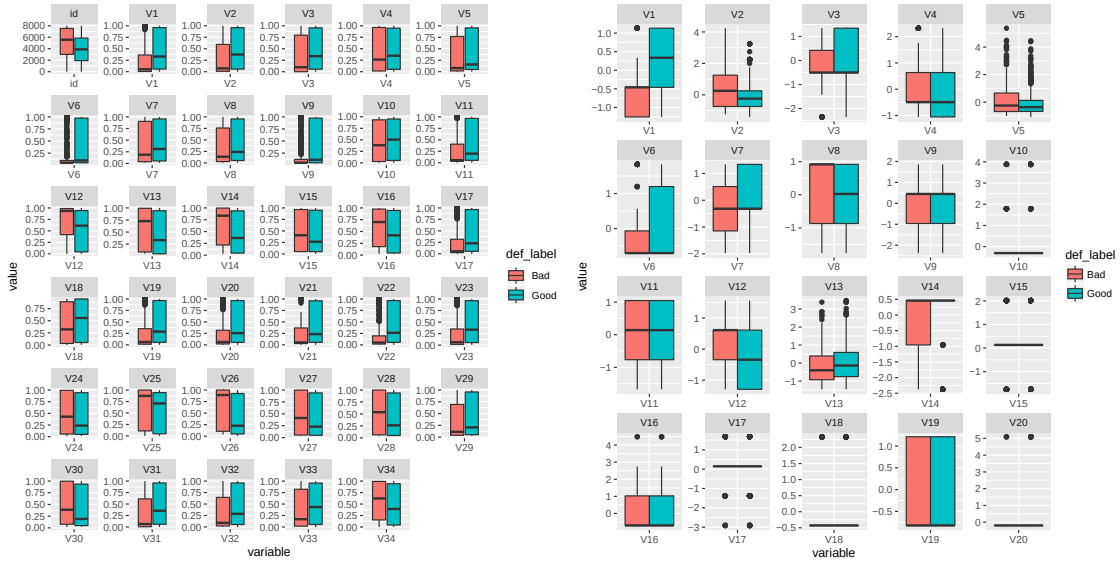
The data set is publicly available (Tomczak, 2016) and was collected from Emerging Markets Information Service, which is a database containing information on emerging markets around the world. Defaulted companies are analysed in the period 2000–2012, while still operating companies are evaluated from 2007 to 2013. The default indicator tracks the default status with a year lag. The data set has 5910 observations on 64 independent variables. Simple mean imputation is applied to the Polish data due to the low number of missing values, see Table 28 in Appendix A, which present the descriptive statistics before and after imputation. The Polish data are standardised. Standardization is a popular transformation applied in classification problems.

3.3 German data set

The data set is publicly available (Hofmann, 1994). It contains retail data for German credit borrowers with 1000 observations on 20 independent variables (covariates or features) and on one binary target variable, which indicates the presence of default with a year lag. The data set contains categorical and numerical variables. For clarity and

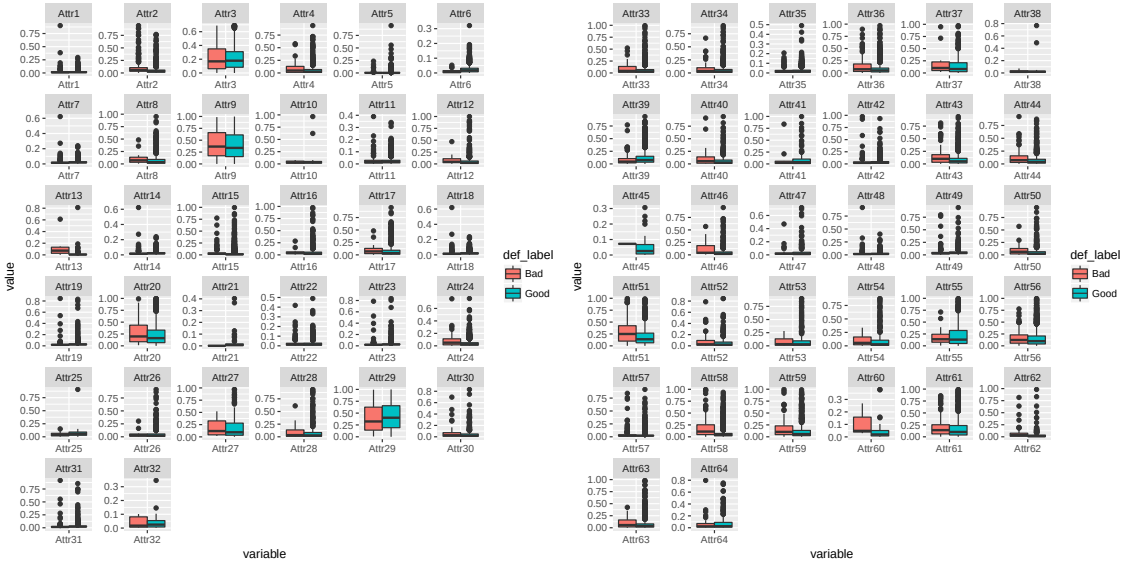
simplicity, we follow (Agresti, 2019). He argues the choice of scores for categorical variables has little impact on the final result. Thus we transform the categorical variables on a continuous scale by mapping them to integer number corresponding to the level of each category. Afterwards all variables (continuous and categorical) are standardised. Missing values are not present in the data. For variable names and variable construction refer to Appendix A, Table 26.

From Figures 1a, 1b, 2a and 2b it can be seen each variable has a significantly different summary statistics when data are split by non-default and default cases. This is promising because increases the possiblity of non-linear relation between the default drivers and PD.



(a) (b)

Figure 1: (a) Box plots on the variables in the East-European corporate data (b) Box plots on the variables in the German retail data.



(a) (b)

Figure 2: (a) Box plots on the variables from 1 to 34 in the Polish corporate data. (b) Box plots on the variables from 35 to 64 in the Polish corporate data.

4 Experiment results

4.1 Experiment 1

With respect to the above discussed studies on feature selection for SVMs, see 2.2.1, this experiment contributes to the literature firstly by proposing an innovative feature selection for SVMs and LR. Secondly by showing that on two out of three datasets SVMs model renders higher classification accuracy than logistic regression. Our findings support the findings of Bellotti et al. (2011) and Harris (2015).

The rest of the thesis is organized as follows. Section 4.1.1 presents the theoretical formulation of SVMs. Section 4.1.2 contains an empirical analysis, including the presentation of the data and the obtained results. Section 4.1.3 discusses the business rationale of the selected default drivers. Finally, Section 4.1.4 concludes the experiment, summarizes the main findings of this research, and proposes some future research directions.

4.1.1 Theoretical foundations

4.1.1.1 Support vector machines Consider a dataset of n pairs $A = \{(\boldsymbol{x}_i, y_i) \mid \boldsymbol{x}_i \in \mathbb{R}^p,\ y_i \in \{-1, +1\}\}_{i=1}^{n}$, where $\boldsymbol{x}_i$ is a p-dimensional "feature" vector and y_i is a label, i.e. a categorical variable whose value gives the class to which $\boldsymbol{x}_i$ belongs. Provided the data are linearly separable, SVMs build a hyperplane that separates the points with $y_i = +1$ from those with $y_i = -1$ maximizing the margin M, i.e. the minimum distance between the hyperplane and each point; the width of the separating band is thus $2M$. For this reason SVMs are also known as maximum margin binary classifiers. A hyperplane can be written as the set of points $\boldsymbol{x}$ satisfying the implicit equation

$$\boldsymbol{w} \cdot \boldsymbol{x} - b = 0, \tag{1}$$

where $\boldsymbol{w}$ is a normal to the hyperplane, $\cdot$ is the scalar product and $b/||w||$ is the distance between the hyperplane and the origin. Thus the objective is

$$\max_{\boldsymbol{w},b} M \quad \text{subject to} \quad y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i - b) \geq 1 \quad \text{for} \quad 1 \leq i \leq n. \tag{2}$$

This optimization problem can more conveniently be rephrased as (Kuhn and Johnson, 2013)

$$\min_{\mathbf{w},b} w \quad \text{subject to} \quad y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i - b) \geq 1 \quad \text{for} \quad 1 \leq i \leq n, \tag{3}$$

where $M = 1/||\boldsymbol{w}||$, and the distance of the hyperplane from the origin is $b/||\boldsymbol{w}||$ (Boser et al., 1992). Mathematically it is more convenient to reformulate this as a quadratic

optimization problem:

$$\arg\max_{\boldsymbol{\alpha}} \left(\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i \cdot \boldsymbol{x}_j \right) \tag{4}$$
$$\text{subject to} \quad 0 \leq \alpha_i \quad \text{for} \quad 1 \leq i \leq n \quad \text{and} \quad \sum_{i=1}^{n} \alpha_i y_i = 0,$$

where α_i are Lagrange multipliers. The solution $\boldsymbol{\alpha}_*$ determines the parameters $\boldsymbol{w}_*$ and b_* of the optimal hyperplane for the dual optimization problem. Usually, only a small number of Lagrange multipliers are positive and the corresponding vectors are in the proximity of the optimal hyperplane. The training vectors $\boldsymbol{x}_i$ corresponding to the positive Lagrange multipliers are called support vectors.

An extension of the above concept can be found in the non-separable case (Cortes and Vapnik, 1995). The problem of finding the optimal hyperplane has the expression

$$\arg\min_{\boldsymbol{w},b,\xi} \left(\frac{1}{2} w^2 + C \sum_{i=1}^{n} \xi_i \right) \tag{5}$$
$$\text{subject to} \quad y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b + \xi_i - 1) \geq 0 \quad \text{and} \quad \xi_i \geq 0,$$

where ξ is a positive "slack" variable and C is a user-defined penalty parameter. The optimization problem in Eq. (5) can be solved with the Lagrangian method Rockafellar (1993) as before, except that now $0 \leq \alpha_i \leq C$.

Non-linear SVMs map the training samples from the input space to a higher-dimensional feature space via a function $\boldsymbol{\Phi}(\boldsymbol{x}_i)$ (Cristianini and Shawe-Taylor, 2000). The use of a kernel function avoids to specify an explicit mapping:

$$\boldsymbol{\Phi}\left(\boldsymbol{x}_i\right) \cdot \boldsymbol{\Phi}\left(\boldsymbol{x}_j\right) = k\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right). \tag{6}$$

Many kernel functions have been investigated in the literature. One of the most useful Broomhead and Lowe (1988) is a radial basis function (RBF),

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{1}{2\sigma^2} \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 \right) \tag{7}$$
$$= \exp(-\gamma ||\boldsymbol{x}_i||) \exp(-\gamma ||\boldsymbol{x}_j||^2) \exp(2\gamma \boldsymbol{x}_i \cdot \boldsymbol{x}_j), \tag{8}$$

where $\gamma = 1/\sigma^2$ is the scaling parameter. The kernel generalization of the decision

function for each $\boldsymbol{z}$ is

$$f(\boldsymbol{x}_i, \boldsymbol{z}, \boldsymbol{\alpha}_*, b_*) = \operatorname{sgn}\left(\sum_{i=1}^{n} \alpha_i y_i k\left(\boldsymbol{x}_i, \boldsymbol{z}\right) + b\right), \tag{9}$$

where $\boldsymbol{z} \in \mathbb{R}^p$ is a vector containing the number of features of a new observation.

One of the less investigated areas of SVMs is the width of the hyperplane that separates the labels (Chang and Lin, 2011). The average distance of the support vectors from the hyperplane is called hyperplane width:

$$\bar{D} = \frac{1}{s}\sum_{l=1}^{s} D_l. \tag{10}$$

The distance D_l of support vector l from the hyperplane is

$$D_l = \frac{1}{w}|f(\boldsymbol{x}_l, \boldsymbol{\alpha}_*, b_*)|, \tag{11}$$

where

$$w = \sqrt{\sum_{l=1}^{s}\sum_{m=1}^{s} y_l y_m \alpha_l \alpha_m k(\boldsymbol{x_l}, \boldsymbol{x_m})}, \tag{12}$$

where s is the total number of support vectors.

Instead of predicting a label y_i, many applications require a posterior class probability $P(y_i = 1|\boldsymbol{x}_i)$. The transformation of class labels to PD estimates is done with Platt's method (Platt, 1999).

4.1.1.2 Data transformations The comparison of different models depends on how the data are transformed. This is another aspect that is rarely discussed when model performance is assessed. From a practical point of view, data transformations play a pivotal role in every statistical model (Box and Cox, 1964). A truncated sigmoid transformation was applied to the East-European data, see 3.1, prior to modelling the default probabilities. The sigmoid function is a popular practical choice that allows to diminish the outliers' effect and to bound the feature values between 0 and 1 (Balaji and Baskaran, 2013):

$$f(x) = \begin{cases} 0 & \text{if} \quad |x - x_0| \geq 100 \\ \frac{1}{1+\mathrm{e}^{-k(x-x_0)}} & \text{if} \quad |x - x_0| < 100, \end{cases} \tag{13}$$

where $x_0 = (\max x - \min x))/2$ is the midpoint and $k = 2.95/(\max x - x_0)$ gives the steepness of the curve. The choice of k has been discussed in Strasburger (2001). Based

on his study we explore different choices of the numerator 2.95 of k so that k is in the range from 1 to 10 as suggested by Strasburger (2001). We note that the above sigmoid transformation has been applied to East-European data only. For illustrative purposes Figure 3 below shows the sigmoid transformation on simulated data between 0 and 1. As one can see the curvature of the function can vary by changing the value of k.

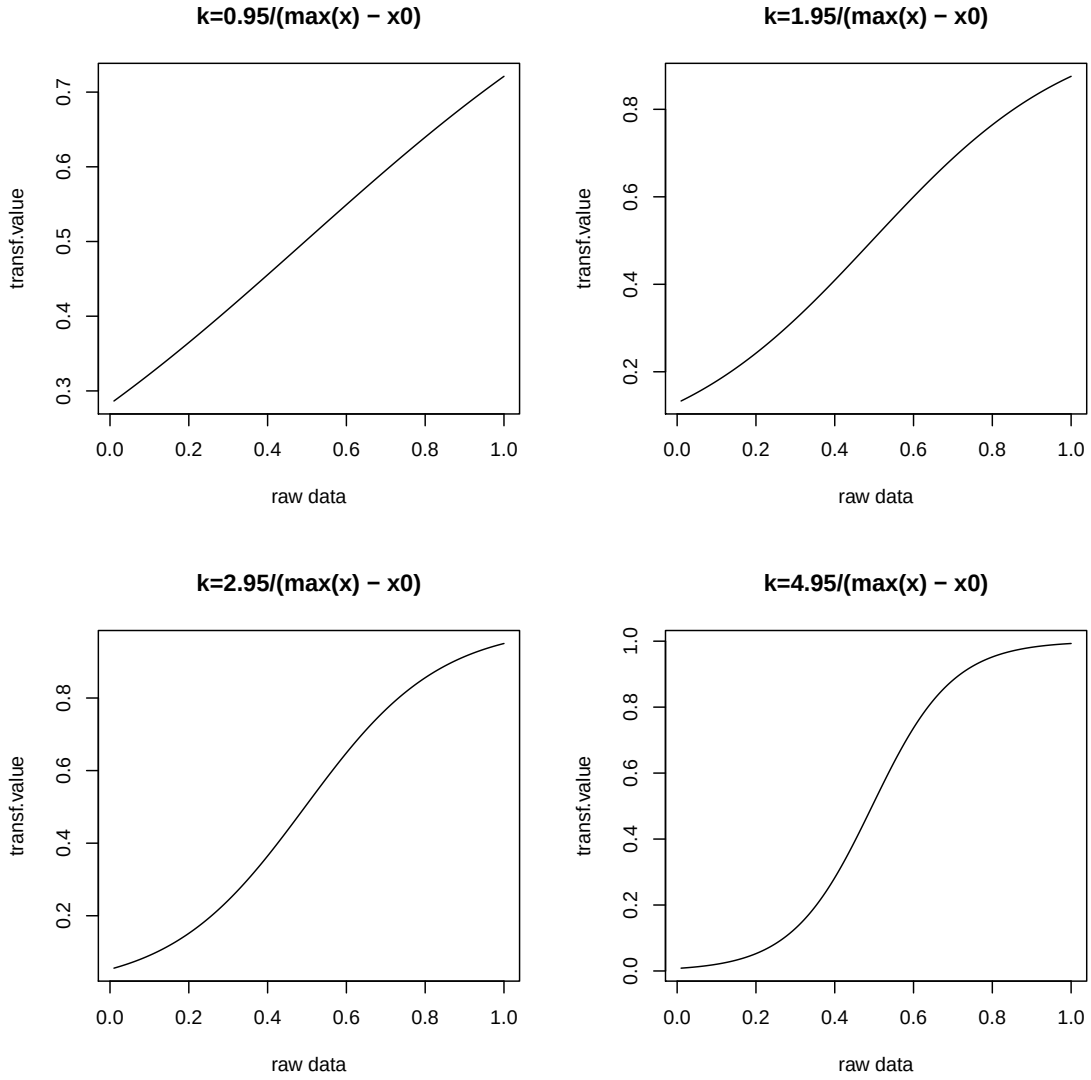


Figure 3: Sigmoid function for different values of k that range between 1 and 10.

Table 1 presents a robustness check on how the AUC changes on real test data compared to the base scenario of $k = 2.95/(\max x - x_0)$. In all cases the AUC deviation is between 1% and 2%, which is considered not significant.

Table 1: AUC deviation in % from base scenario $k = 2.95/(\max x - x_0)$ per different values of slope parameter k.

k	SVM	LR
$0.95/(\max x - x_0)$	1%	2%
$1.95/(\max x - x_0)$	1%	2%
$2.95/(\max x - x_0)$	0%	0%
$4.95/(\max x - x_0)$	2%	2%

4.1.2 Empirical analysis

4.1.2.1 Feature selection The objective of variable selection is threefold: improve the prediction performance of the predictors, provide faster and more cost-effective predictors, and provide a better understanding of the underlying process that generates the data.

The statistical literature offers many approaches for feature selection (Guyon and Elisseeff, 2003). However, there is no proven methodology that works for each data set. Based on previous experience on the selection of appropriate features for different models, we decided that an automatic script shall be written that overcomes many of the drawbacks of a manual feature selection process. An univariate analysis on the features is the most common approach used for feature selection: those features that exhibit good performance based on a specific measure, for example the F-score (Güneş et al., 2010), are selected for further analysis. Nevertheless, there are some negative aspects of this approach (Quanquan et al., 2011):

1. Some variables cannot discriminate well on a standalone basis but show better explanatory power in a combination with other factors.
2. Often the modeller selects a combination of factors that is highly correlated and even though they have a strong performance on a univariate level, it is difficult to select a combination of factors with a low multicollinearity.

In order to avoid the above drawbacks of the simpler methods for variable selection, we propose an innovative variable selection method that we apply to the three data sets described above. The applied feature selection algorithm consists of the following steps:

1. Initialization: set F = initial set of n features, D = development sample, V = validation sample and S = selected set of features, where $S \subseteq F$. Define f_S^k = set of all feature combinations at k, where $k \in \{1, \ldots, n\}$ is a generation index for a feature combination $S = \{i, j, \ldots, z\}$ with cardinality $l \leq n$. Set $P^k \subseteq S$ = final approved combinations of features for generation k.

2. **for** $k = 1, \ldots, N$

    1. Create generation k of feature combinations $S^k = \{i, j, \ldots, z\} \implies f_S^k$ where $i \neq j \neq \ldots \neq z$. The number of different feature combinations is $\binom{n}{r} = \frac{n!}{r!(n-r)!}$, where r is the cardinality of S^k and n is the total number of features.

    2. For each $\{i, j, \ldots, z\}$ of generation k compute:
    if model $==$ SVMs **then**

        1. $\bar{D}^k_{\{i,j,\ldots,z\}}$ = hyperplane width for f_S^k.

        2. $s^k_{\{i,j,\ldots,z\}}$ = number of support vectors for f_S^k.

        3. $\mathrm{AUC}^k_{\{i,j,\ldots,z\}}$ = area under the curve (AUC) for f_S^k on V^k, where V^k is a validation sample for a feature combination from generation k.

    end if
    if model $==$ LR **then**

        1. p-value$^k_{\{i,j,\ldots,z\}}$ = a p-value for f_S^k

        2. $\mathrm{AIC}^k_{\{i,j,\ldots,z\}}$ = an Akaike information criterion (AIC) for f_S^k

        3. $\mathrm{BIC}^k_{\{i,j,\ldots,z\}}$ = a Bayes information criterion (BIC) for f_S^k on V^k, where V^k is a validation sample for a feature combination from generation k

    end if
    On D^k compute the $l \times l$ feature correlation matrix $\boldsymbol{A}$, where l is the cardinality of $\{i, j, \ldots, z\}$.

    3. For each $\{i, j, \ldots, z\}$ of generation k, given a predefined AUC threshold AUC_t test:
    if $\mathrm{AUC}^k_{\{i,j,\ldots,z\}} \geq \mathrm{AUC}_t$ **and** maximum element of $\boldsymbol{A} \leq 60\%$ **then** accept $P^k \subseteq S^k$ for $\{i, j, \ldots, z\}$
    end if

    4. Given all accepted feature combinations (P^k) from generation k, increase the cardinality of the set $\{i, j, \ldots, z\}$ by 1 until $k = n$.

    end for

3. Test the performance of the model on test data on all accepted feature combinations (P^k) from each generation k.
    if model $==$ SVMs **then**

1. Select the l feature combinations with the highest AUC, distance to the hyperplane and the lowest number of support vectors on the test data in that order.

end if
if model $==$ LR **then**

1. Select the l feature combinations with the highest AUC, AIC and BIC on the test data in that order.

end if

For the data sets under investigation the algorithm explained above is run under the following conditions:

1. The initial number of features is equal to n, i.e. to the total number of variables in each data set for both models.

2. The first generation $k = 1$ contains only 2 features for both models. It is assumed that including more than 5 features can result in overfitting the data especially for the logistic regression. SVMs method has an embedded regularization, i.e., it introduces additional information in order to prevent overfitting Fan et al. (2012), but overfitting is still possible.

3. The AUC threshold in Step 2.3 is set to 60% on the validation sample. We set the threshold of 60% based on Mukaka (2012) and Schober et al. (2018) who show that a moderate to high correlation could be assumed around the range 60%-80%. We set the correlation to the lower bound of 60% in order to increase the computational efficiency of the algorithm. However, we can increase the threshold in order to apply a more exhaustive search of different variable combinations. Table 2 presents with how many percentages the AUC deviates from the base scenario of 60% threshold. Changes are considered to be not significant. We note that increasing the variable correlation in LR could undermine the statistical inference in LR, therefore using low correlated variables is preferable.

Table 2: AUC deviation in % from base scenario (60% correlation threshold).

Variable correlation threshold	SVM	LR
60%	0%	0%
70%	1%	2%
80%	1%	2%

4. The feature correlation matrix in Step 2.2 is estimated using the Pearson product-

moment correlation coefficient.

5. The number of final selected feature combinations l on Step 3 of the algorithm is set to 5 for the SVMs and LR.

6. To improve the computational efficiency of the algorithm, the total number of variables is reduced by randomly sampling 10 variables out of n without replacement and running the algorithm 10 times on different random sub-samples of n.

7. The SVMs model is run with an RBF kernel with parameter $\gamma = \frac{1}{k+1}$. The penalty parameter C is kept constant across the iterations and the feature combinations. This allows a direct comparison of the number of support vectors for each combination. The number of support vectors is also affected by the number of features in the model. However the effect is not significant and therefore this factor is ignored when comparing the number of support vectors. The expectation is that the lower the number of support vectors the better the model. Nonetheless, we have to point out that the number of support vectors is affected by several factors:

    1. the size of the data (the number of observations for the validation sample and the training sample is constant for each iteration, only the content is different);

    2. the cost C of constraints violation;

    3. the RBF kernel.

4.1.2.2 Selection of the best performing LR models on test data Table 3 presents the output from the feature selection method on the German training data, see 3.3. The calibration data are split into training set and test set. The feature selection method is run on the training data and the performance is measured on the test (validation) data. The columns of Table 3 show the BIC, AIC and AUC on the test data. The algorithm selects the five feature combinations with the lowest BIC, AIC and with the highest AUC on the German test data. Table 4 and Table 5 present the output of the feature selection method on the East-European, see 3.1 and Polish data, see 3.2.

Table 3: Final feature combinations for LR, German retail data. AUC, AIC and BIC on test (validation) data.

Feature combination	BIC	AIC	AUC
1, 2, 7, 9, 19	250.22	270.01	77.94%
1, 2, 7, 14, 19	245.43	265.22	77.71%
1, 2, 7, 8, 19	249.26	269.05	77.70%
1, 2, 7, 18, 19	250.28	270.07	77.49%
1, 2, 7, 19, 20	249.26	269.05	77.48%

Table 4: Final feature combinations for LR, East-European corporate data. AUC, AIC and BIC on test (validation) data.

Feature combination	BIC	AIC	AUC
1, 8, 14, 25, 30, 32	1019.43	1053.20	77.92%
1, 14, 25, 30, 32, 33	1024.97	1058.74	77.84%
1, 13, 14, 25, 30, 32	1024.67	1058.45	77.80%
8, 9, 14, 26, 29, 30	997.45	1031.22	77.58%
9, 11, 14, 25, 29, 30	958.74	992.51	77.55%

Table 5: Final feature combinations for LR, Polish corporate data. AUC, AIC and BIC on test (validation) data.

Feature combination	BIC	AIC	AUC
2, 21, 26, 34, 39	462.11	486.06	84.85%
2, 21, 34, 39, 45	466.38	490.34	84.33%
2, 11, 21, 34, 39	464.94	488.89	83.93%
6, 32, 43, 55, 56	473.58	497.53	83.69%
2, 11, 34, 39, 45	465.78	489.73	83.59%

4.1.2.3 Selection of the best performing SVMs models on test data Table 6 presents the output from the feature selection method on the German training data. The calibration data are split into a training set and a test set. The feature selection method is run on the training data and the performance is measured on the test data. The columns of Table 6 show the distance to the hyperplane, the number of support vectors and the AUC on the German test data. The algorithm selects the five feature combinations with the highest distance to the hyperplane, the lowest number of support vectors and the highest AUC on the test data. Table 7 and Table 8 present the output of the feature selection method on the East-European and Polish data.

Table 6: Final feature combinations for SVMs, German retail data. AUC, distance to the hyperplane and number of support vectors on test (validation) data.

Feature combination	Distance	Number of SV	AUC
1, 11, 13, 14, 15	0.119	153	79.45%
1, 10, 13, 14, 15	0.077	152	79.34%
1, 4, 10, 13, 14	0.062	148	79.13%
1, 4, 13, 14, 19	0.055	152	78.98%
1, 2, 6, 11, 17	0.072	151	78.90%

Table 7: Final feature combinations for SVMs, East-European corporate data. AUC, distance to the hyperplane and number of support vectors on test (validation) data.

Feature combination	Distance	Number of SV	AUC
9, 14, 25, 30	0.437	554	75.09%
14, 25	0.159	588	74.95%
1, 6, 30	0.155	568	74.82%
9, 25, 30	0.061	588	74.26%
1, 25, 30	0.066	568	74.10%

Table 8: Final feature combinations for SVMs, Polish corporate data. AUC, distance to the hyperplane and number of support vectors on test (validation) data.

Feature Combination	Distance	Number of SV	AUC
2, 26, 39	0.196	309	83.79%
2, 11, 39	0.177	310	83.74%
2, 39, 45	0.181	310	83.72%
2, 21, 39	0.181	310	83.72%
2, 26, 34, 39, 55	0.272	309	83.71%

4.1.2.4 Out-of sample results The final feature combinations selected from the LR are further tested on out-of sample data. The results are shown in Table 9, Table 10 and Table 11. The columns of the tables below show the percentage of overall correctly classified obligors, the percentage of correctly classified good obligors, the percentage of the correctly classified bad olbiogrs and the AUC on the out-of-sample data for LR.

The data have been split into training, test and validation. The validation data (on which the out-of sample results are calculated) consist of 100 defaulted and 100 non-defaulted observations for each of the three data sets explored in this thesis. The validation data

(out-of sample data) have been randomly sampled 100 times, each time sampling 90%. Based on that sampling we calculated a standard deviation for the total number of correctly classified observations. The standard deviation is in the range of 0.7% to 1.4%, which is expected. The training and test data are equal in size and are based on the total number of defaults-100 for each data set. We exclude 100 because these were allocated to the validation data. Additionally the training and test data are balanced, have equal number of defaulted and non-defaulted observations.

Table 9: Final feature combinations for LR, out-of-sample German retail data. Percentage of correctly classified (All), percentage of the correctly classified bad obligors (Bad), percentage of the correctly classified good obligors (Good), AUC.

Feature combination	All	Good	Bad	AUC
1, 2, 7, 9, 19	70%(1.1)	70%	70%	77%
1, 2, 7, 14, 19	68%(1.0)	69%	66%	75%
1, 2, 7, 8, 19	70%(1.0)	68%	72%	78%
1, 2, 7, 18, 19	67%(1.1)	68%	65%	75%
1, 2, 7, 19, 20	69%(1.0)	69%	68%	75%

Table 10: Final feature combinations for LR, out of sample East-European corporate data. Percentage of correctly classified (All), percentage of the correctly classified bad obligors (Bad), percentage of the correctly classified good obliogors (Good), AUC.

Feature combination	All	Good	Bad	AUC
1, 8, 14, 25, 30, 32	62%(1.3)	60%	63%	69%
1, 14, 25, 30, 32, 33	61%(1.4)	63%	58%	67%
1, 13, 14, 25, 30, 32	60%(1.4)	62%	58%	67%
8, 9, 14, 26, 29, 30	65%(1.1)	58%	71%	67%
9, 11, 14, 25, 29, 30	62%(1.2)	55%	70%	69%

Table 11: Final feature combinations for LR, out of sample Polish corporate data. Percentage of correctly classified (All), percentage of the correctly classified bad obligors (Bad), percentage of the correctly classified good obliogors (Good), AUC.

Feature combination	All	Good	Bad	AUC
2, 21, 26, 34, 39	87%(0.7)	85%	89%	93%
2, 21, 34, 39, 45	88%(0.7)	87%	88%	93%
2, 11, 21, 34, 39	87%(0.7)	86%	88%	92%
6, 32, 43, 55, 56	71%(0.7)	80%	61%	81%
2, 11, 34, 39, 45	81%(0.7)	83%	79%	89%

The final feature combinations selected from the SVR are further tested on out-of sample data. The results are shown in Table 12, Table 13 and Table 14. The columns of the tables are analogous to those of Tables 9–11.

Table 12: Final feature combinations for SVMs, out-of-sample German retail data. Percentage of correctly classified (All), percentage of the correctly classified bad obligors (Bad), percentage of the correctly classified good obliogors (Good), AUC.

Feature combination	All	Good	Bad	AUC
1, 11, 13, 14, 15	70%(1.1)	64%	75%	70%
1, 10, 13, 14, 15	69%(1.2)	58%	79%	71%
1, 4, 10, 13, 14	71%(1.3)	59%	83%	72%
1, 4, 13, 14, 19	73%(1.1)	65%	81%	74%
1, 2, 6, 11, 17	76%(1.0)	78%	74%	77%

Table 13: Final feature combinations for SVMs, East-European corporate out-of-sample data. Percentage of correctly classified (All), percentage of the correctly classified bad obligors (Bad), percentage of the correctly classified good obliogors (Good), AUC.

Feature combination	All	Good	Bad	AUC
9, 14, 25, 30	70%(1.1)	70%	69%	69%
14, 25	64%(1.4)	52%	76%	64%
1, 6, 30	70%(1.2)	65%	74%	70%
9, 25, 30	66%(1.2)	68%	64%	66%
1, 25, 30	70%(1.1)	72%	67%	70%

Table 14: Final feature combinations for SVMs, Polish corporate out-of-sample data. Percentage of correctly classified (All), percentage of the correctly classified bad obligors (Bad), percentage of the correctly classified good obliogors (Good), AUC.

Feature combination	All	Good	Bad	AUC
2, 26, 39	79%(0.7)	83%	74%	80%
2, 11, 39	79%(0.7)	83%	74%	80%
2, 39, 45	79%(0.7)	83%	74%	82%
2, 21, 39	83%(0.7)	83%	83%	84%
2, 26, 34, 39, 55	81%(0.7)	76%	85%	81%

The results based on one out-of-sample data set indicate that in terms of AUC the logistic regression should out-perform the SVMs on all data sets. For the German data the AUC

of the LR ranges from 75% to 78%, whereas the AUC of the SVMs ranges from 70–77%. For the East-European data the AUC of the LR ranges from 67–69%, whereas the AUC of the SVMs ranges from 64–70%. For the Polish data the AUC of the LR ranges from 81–93%, whereas the AUC of the SVMs ranges from 80–84%. However, the percentage of the overall correctly classified obligors is a better measure of classification accuracy, whereas the AUC is a rank-ordering measure. In terms of correctly classified obligors the SVMs out-performs the LR on two out of three data sets (the German and the East-European data), see column "All" in Tables 12–14.

For that reason the final feature combinations selected from the LR and the SVMs models are further tested 100 times with different out-of-sample data sets (subsets of the main out-of-sample data set). Figures 4–9 show the results. SVMs performers better than LR in two out of three data sets.

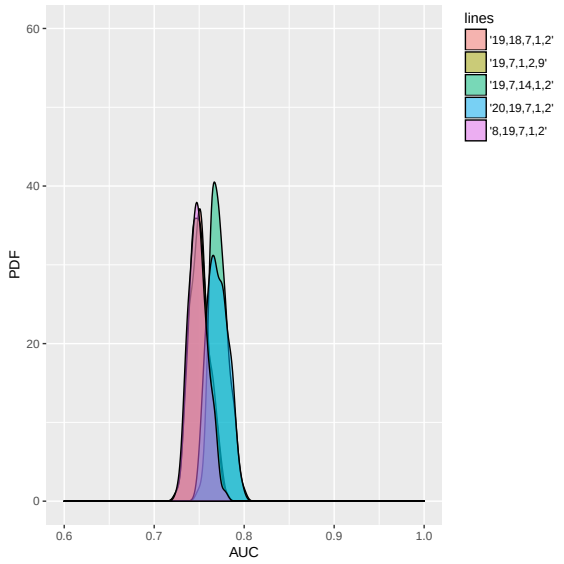


Figure 4: AUC distribution on out-of-sample German retail data, LR.

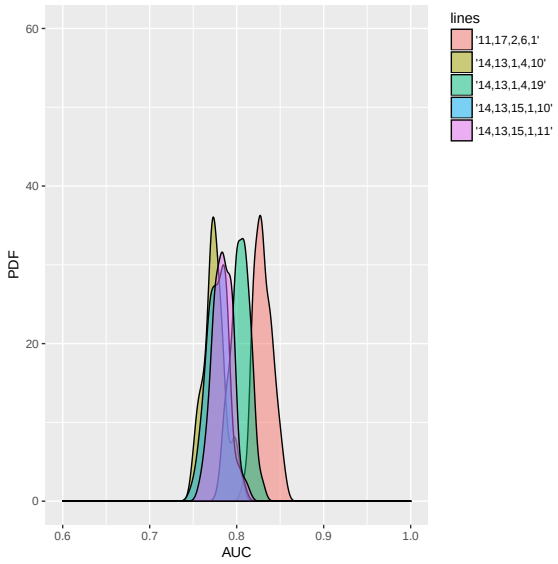


Figure 5: AUC distribution on out-of-sample German retail data, SVMs.

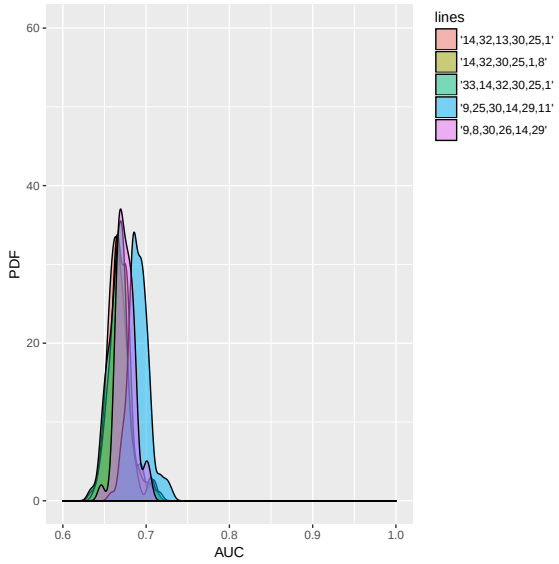


Figure 6: AUC distribution on out-of-sample East-European corporate data, LR.

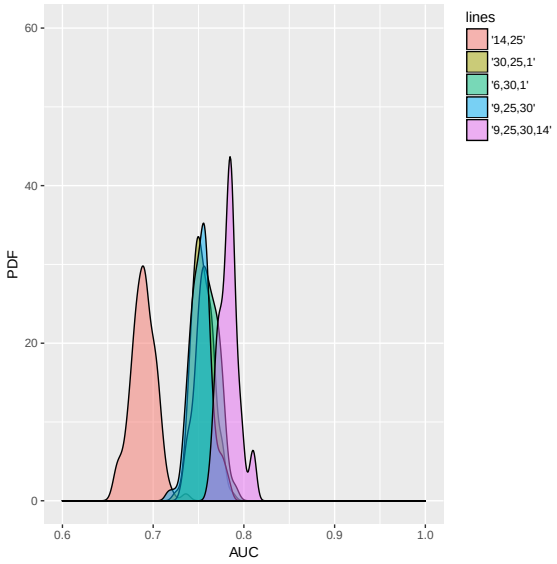


Figure 7: AUC distribution on out-of-sample East-European corporate data, SVMs.

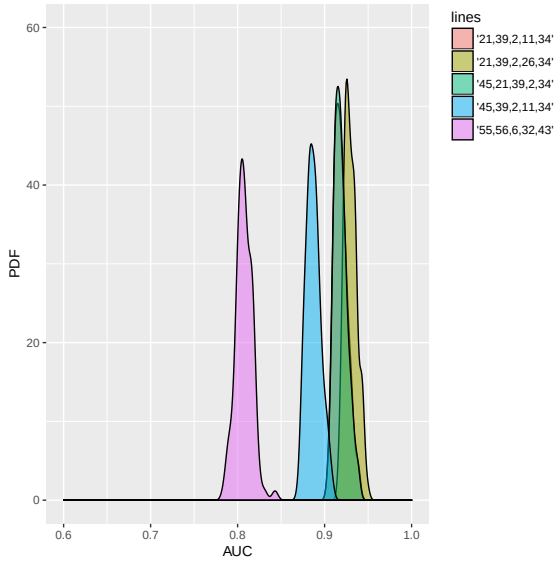


Figure 8: AUC distribution on out-of-sample Polish corporate data, LR.

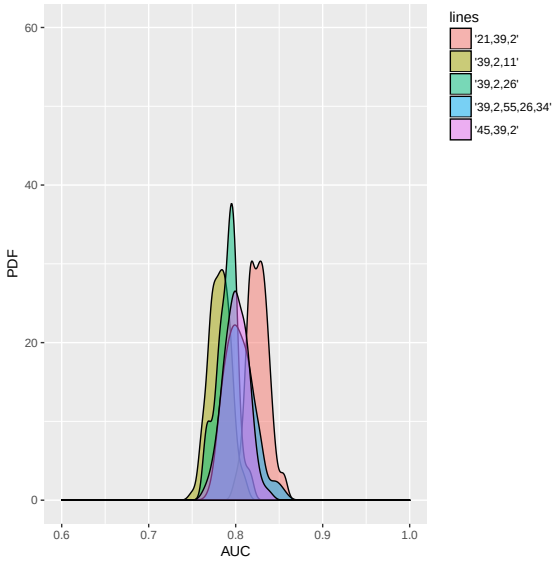


Figure 9: AUC distribution on out-of-sample Polish corporate data, SVMs.

4.1.2.5 Comparison of the variable selection method to an alternative variable selection methods

Table 15 presents the output of the sequential variable selection method implemented in MATLAB. Hira and Gillies (2015) provide a comprehensive discussion on feature selection methods. The results show that the proposed variable selection method performs similarly to the challenger selection method on the out-of sample data. On the Polish data the proposed method outperforms significantly the alternative variable selection method.

Table 15: Final feature combinations; challenger feature selection method applied to the out-of-sample data. Percentage of correctly classified (All), percentage of the correctly classified bad obligors (Bad), percentage of the correctly classified good obliogors (Good), AUC.

Data Set	Method	Feature combination	All	Good	Bad	AUC
German retail data	LR	1, 2, 3, 10, 12, 19	73%	74%	71%	77%
German retail data	SVMs	1, 2, 3, 10, 12, 19	77%	81%	73%	78%
East-European data	LR	6, 9, 21, 22, 25, 30	65%	60%	69%	70%
East-European data	SVMs	6, 9, 21, 22, 25, 30	72%	74%	69%	72%
Polish data	LR	1, 28, 32, 47, 62	79%	84%	73%	86%
Polish data	SVMs	1, 28, 32, 47, 62	77%	86%	67%	80%

We further test the performance of the challenger sequential variable selection method 100 times with different out-of-sample data sets (subsets of the main out-of-sample data set). In the case we show that the performance of the proposed selection method works well for the SVMs when it is based on the distance to the hyperplane. The SVMs distance to the hyperplane method outperforms the challenger method on all data sets as can be seen by comparing Figures 5, 7 and 9 with Figures 11, 13 and 15. In the case of LR, where we do not use the distance to the hyperplane and the number of support vectors (this is possible only for SVMs), the proposed method has similar performance and LR outperforms the challenger only on the Polish data as can be seen by comparing Figures 4, 6 and 8 with Figures 10, 12 and 14. However, this is due to the fact that in general LR is a more suitable method for that data set as can be concluded when compared to the SVMs.

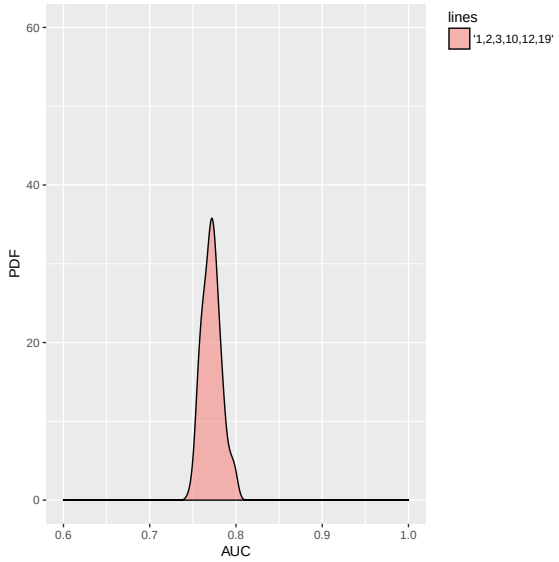


Figure 10: AUC distribution on out-of-sample German retail data, LR.

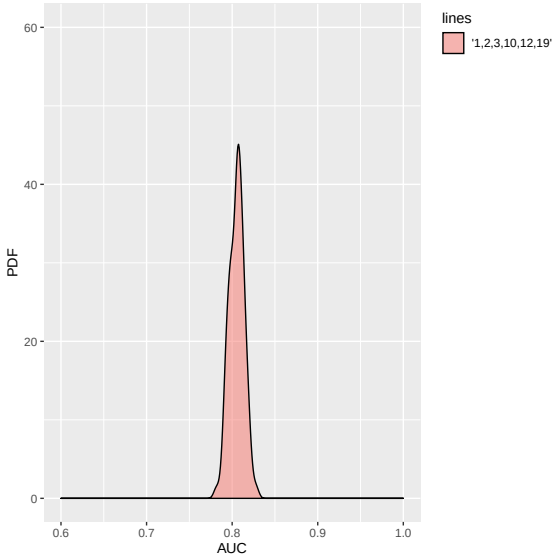


Figure 11: AUC distribution on out-of-sample German retail data, SVMs.

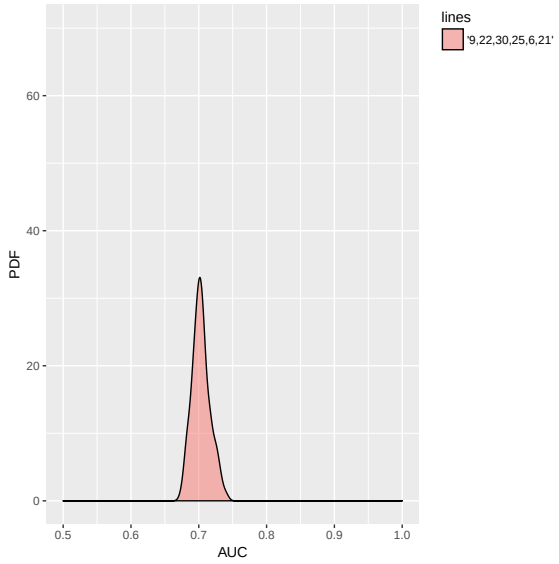


Figure 12: AUC distribution on out-of-sample East-European corporate data, LR.

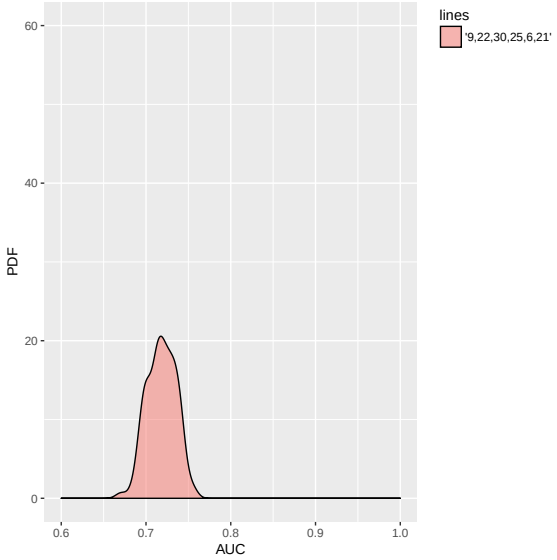


Figure 13: AUC distribution on out-of-sample East-European corporate data, SVMs.

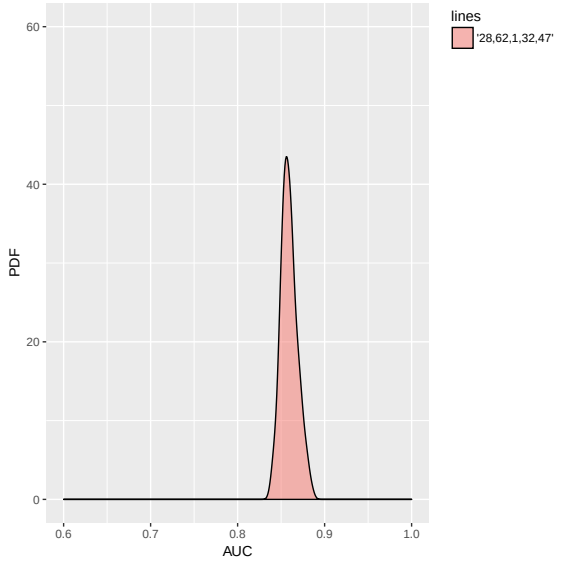


Figure 14: LR, AUC distribution on out-of-sample Polish corporate data.

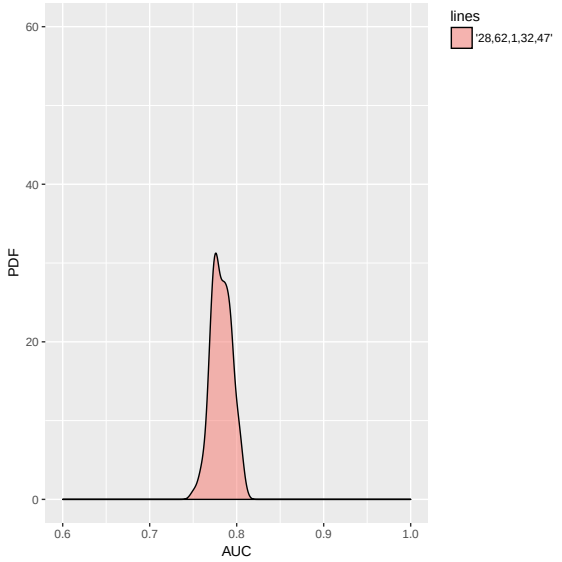


Figure 15: SVMs, AUC distribution on out-of-sample Polish corporate data.

4.1.3 Managerial insights

The economic interpretation of the final results is important. For that reason we identify the most frequent default drivers in each data set. Referring back to tables Tables 9–15 and counting the occurrence of variables in both models (LR and SVMs) we present in Table 16 the occurrence of each feature in each data set. Then we compare the most frequent variables from the proposed variable selection method to the ones given by the challenger variable selection method. If possible, we identify the common features between the two methods considering only those variables from the proposed method that appear at least 6 times (in 50% of the cases, we have 10 final models for each data set). For the Polish data set there is no common frequent variables between the two methods and therefore we further discuss the variables from the proposed method only.

Table 16: Selection of the most frequent variables on the test (validation) data across the three different data sets: German (G), East-European (E), Polish (P). The last three columns are based on the challenger variable selection method applied to the data sets: German (CV_G), East-European (CV_E), Polish (CV_P); Id colums show the variable id in a given data set, Freq columns show the number of times a variable appears in all the final variable combination (maximum can be 10, 5 models for LR and 5 models for SVMs).

Id (G)	Freq (G)	Id (E)	Freq (E)	Id (P)	Freq (P)	Id (CV_G)	Id (CV_E)	Id (CV_P)
1	10	30	9	2	9	1	6	1
2	6	25	8	39	9	2	9	28
19	6	14	7	34	5	3	21	32
7	5	1	5	21	4	10	22	47
14	5	9	4	11	3	12	25	62
13	4	32	3	26	3	19	30	
4	2	8	2	45	2			
10	2	11	2	55	2			
11	2	29	2	6	1			
15	2	6	1	43	1			
6	1	13	1	56	1			
8	1	26	1					
9	1	33	1					
17	1							
18	1							
20	1							

Following the logic described above we have identify the following common variables:

1. For the German retail data the most common variables across the two selection methods are: status of existing checking account, duration of the account in months and phone number availability.
2. For the East-European corporate data the most common variables across the two selection methods are: earnings on operating income and total assets.
3. For the Polish corporate data the most common variables are: total liabilities/total assets and profit on sales/sales.

The results are shown in Table 17.

Table 17: Selected most frequent variables for each data set, based on proposed and challenger variable selection methods.

Data set	Variable ID	Variable name
German	1	status of existing checking account
German	2	duration in months of the account
German	19	telephone availability
East-European	25	earnings on operating income
East-European	30	total assets
Polish	2	total liabilities/total assets
Polish	39	profit on sales/sales

One explanation for the total assets to significantly affect the PD is that the change in total assets is related to business growth. If a business grows substantially in terms of assets, this means that large long-term investments were made in that business. All other factors being equal, the long-term investments will result in higher profit if the company keeps the same level of operational risk. In the retail, the final variables that appear most are the "status of existing checking account" and the "duration in months of the checking account".

The above difference in the most frequent ratios across the models and the datasets shows that model selection is not only a function of the best performing model but also a function of the business goals and the business environment of the lending institution.

4.1.3.1 Reference to the findings of other authors Bellotti and Crook (2009b) found that one of the most important factors for default estimation are "home owner status" and the "time with bank". We also found that the time spent with the bank is a main indicator of default risk. However, Bellotti and Crook (2009b) found other significant indicators of default such as "total outstanding balance excluding mortgages on all active CAIS accounts" and "total number of credit searches in last 6 months". In contrast, we did not identify similar variables to appear frequently as default risk drivers. One reason is the fact the we kept the total number of variables down to five, whereas Bellotti and Crook (2009b) used as many as eleven variables in their final model.

A study on wholesale data was done by (Chen et al., 2011). They found that the variable "account payable turnover" is a significant factor in measuring credit risk. The other seven variables proposed by Chen et al. (2011) were mainly based on the total assets and sales. Another interesting study is by (Hammer et al., 2012). They evaluated the creditworthiness of banks using statistical, as well as combinatorics-optimization logic-based

methodologies. In their study the Fitch risk ratings of banks were reversed-engineered using ordered logistic regression, SVMs, and Logical Analysis of Data (LAD). They also indicated that total assets and liabilities play an important role in differentiating between good and bad obligors. This solidifies our findings and shows that although the individual factors can be slightly different, the major components of these factors are the same in both studies. This is also consistent with the findings of (Tian et al., 2015). The business intuition is that the amount of the total assets relative to the liquid assets or other balance sheet items such as net profit provide a clear picture of how efficient the utilization of those assets by a particular obligor is. Minimizing the amount of total assets and maximizing the net profit is the objective of every private company. Another common default driver is the short-term (current) liabilities. This is consistent with the findings of Gök (2015). The business intuition is that current liabilities is a significant indicator of short-term debt. Companies with high levels of current liabilities in relation to other balance sheet items such as cash and sales are riskier and therefore they have a higher default probability. Finally we stress on that fact that although some differences exist between the Polish obligors and those of East-European obligors, most of the default drivers are the same, namely total assets, total liabilities and sales. This is consistent with the findings of Hosaka and Takata (2016).

4.1.4 Conclusion

The findings of this research experiment yield promising insights into the potential of SVMs to estimate the probability of default (PD) of corporate and retail clients. Our work is consistent with the findings of Bellotti and Crook (2009b) with respect to the usefulness of SVMs for credit scoring.

Furthermore, we apply a wrapper approach for feature selection based on the distance of the support vectors from the separating hyperplane. We show that a combination of a wider hyperplane and fewer support vectors leads to a higher discrimination power for SVMs.

From a financial point of view, the most frequently applied variables for PD estimation are total assets, total liabilities and sales in the corporate segment. In the retail segment the variables that appear most are current account status and duration of the current account.

Future work may include more experiments on estimating other Basel measures such as loss-given default (LGD) and exposure at default (EAD). Supervised non-linear machine learning methods can be successfully applied for the estimation of PD, LGD and EAD in a way that accounts for their correlations. The collateral prices and their evolution, which are an important aspect of the capital calculations under the Basel guidelines, can also be

modelled with non-linear machine learning methods.

Overall, the SVMs model proposed here shows promising results. Practically, this could save time and effort and will lead to making better-informed credit risk decisions.

4.2 Experiment 2

With respect to the above discussed studies on ANNs, see 2.2.2, this experiment contributes to the literature first by proposing an update to the estimation of the regularization parameters and secondly by exploring classical and Bayesian regularization in the estimation of a network with different architectures.

The rest of the thesis is organized as follows. Section 4.2.1 presents the theoretical formulation of an ANNs in a classical and in a Bayesian framework. Section 4.2.2 presents the results from the regularized networks. Section 4.2.3 discusses the policy implications of the selected default factors and their business intuition. Finally, Section 4.2.4 concludes the experiment by summarizing the main findings.

4.2.1 Theoretical foundations

In theory, there are several neural network architectures. In practice, most researchers (Hagan et al., 2014) focus on three main types: feed-forward, competitive and recurrent networks. While competitive and recurrent networks are definitely an interesting area of research, in this experiment we explore the most popular kind of network architecture, the feed-forward network. It is called a feed-forward network because data moves in forward direction only: initially the data input is processed in the first layer of the network, then it is pushed forward to the next layer until it reaches the final output layer. In a feed-forward network, data are not fed back from a layer to the previous, which instead happens in a recurrent network. A detailed description of a feed-forward network is given in the next subsection.

4.2.1.1 Feed-forward neural network architecture In this section we briefly introduce the most basic theoretical concepts behind an ANNs. A detailed discussion is given by Kim et al. (1996). A multilayer ANNs can be described as a system with the following elements:

1. An input data vector $\mathbf{x} \in \mathbb{R}^p$ and a categorical variable $y \in \{0, 1\}$.
2. An output $\widehat{y} = P(Y = 1|\mathbf{X} = \mathbf{x})$.
3. Layers $k = 1, \ldots, l$ with m units per layer; the layers with $k < l$ are hidden, the layer l is the output layer. Each layer has a bias $b^k \in \mathbb{R}$ and each unit has an activation $h_i^k \in \mathbb{R}$. The units in layer k are connected to those in the previous layer by weights $w_{ij}^k \in \mathbb{R}$, $i, j = 1, \ldots, m,\ k = 1, \ldots, l$.
4. A previous layer is defined as layer $k - 1$ in respect to layer k.

5. The individual inputs $\mathbf{x} \in \mathbb{R}^p$ are each weighted by weights w_{ij}^k. Each neuron i is weighted in each layer k.

6. The final output has a bias $b^{l+1} \in \mathbb{R}$ and is connected to the units of the output layer by weights $w_j^{l+1} \in \mathbb{R}$.

7. An activation function $s_i^k(.)$ for layer k and unit i. An activation function determines how each node reacts in an artificial neural network and what output each node generates. This output is then used as input for the next node in an iterative procedure until the estimation process converges to a local or global optimum. The most popular choices of activation functions are the logistic sigmoid and the hyperbolic tangent (Farhadi, 2017).

Below we present the sequence in which the estimation of the network weights is performed. The first step in the estimation process of the network weights w is to feed data into the first layer of the network. The unit activations of the first layer are computed from the input data as

$$h_i^1 = s_i^1\Big(b^1 + \sum_{j=1}^{p} w_{ij}^1 x_j\Big), \tag{14}$$

where $s_i^1(\cdot)$ is an activation function. After receiving the output from the first layer, we can proceed with the estimation of the second layer activation functions. The unit activations of the next layer are computed from those of the previous layer as

$$h_i^{k+1} = s_i^{k+1}\Big(b^{k+1} + \sum_{j=1}^{m} w_{ij}^{k+1} h_i^k\Big). \tag{15}$$

After reaching the final output by sequentially moving through each hidden layer k, the output probability is estimated as

$$\widehat{y} = b^{l+1} + \sum_{j=1}^{m} w_j^{l+1} h_i^l. \tag{16}$$

In the estimation process described above the activation function $s(.)$ plays a vital role. In our analysis we apply the logistic function which is the most common non-linear activation function,

$$s(x) = \frac{1}{1 + \exp(-x)}. \tag{17}$$

The above estimation process can be described as a learning process where the weights of the network are estimated through learning from the data. In particular the weights w_{ij}^k for layer k and neuron i in the neural network are estimated sequentially and itera-

tively. Afterwards, the network performance with weights learned from the training data is monitored on a test data.

In order to estimate the network weights a cost function is required. The purpose of the cost function is to serve as an objective to be minimized during the learning process. A typical choice of a cost function is the mean squared error (MSE) that can be written as

$$E = \frac{1}{N}\sum_{i=1}^{N}(y_i - \widehat{y}_i)^2, \tag{18}$$

where N is the number of observations, i.e. the number of input data vectors and categorical variables. Another popular cost function is the cross entropy (CE)

$$S = -\sum_{i=1}^{N} p_i \log q_i, \tag{19}$$

where p_i and q_i, $i = 1, \ldots, N$, are the probability masses of two discrete probability distributions.

A common issue in estimating network weights is the overfitting of the network in which the network cannot generalize well and subsequently the network performance on new data is poor. When overfitting occurs the network weights are calculated in way that maximizes the network performance on the training data but this is achieved through significantly decreasing the performance on the test data. The most common way of solving the overfitting issue that occurs in the estimation process is applying regularization during the estimation (Deng et al., 2014). Regularization can be applied to penalize the cost function with the squared sum of the weights so that the generalization performance of the network is maintained. For the MSE cost function this can be written as

$$E_{\text{reg}} = \gamma \sum_{k=1}^{l}\sum_{i,j=1}^{m}(w_{ij}^{k})^2 + (1-\gamma)E = \gamma E_w + (1-\gamma)E, \tag{20}$$

where $\gamma \in (0, 1)$ is a regularization constant. Usually the backpropagation algorithm (Dreyfus, 1990) is used to estimate the weights. A common optimization algorithm used for reaching a convergence of the estimation procedure is the gradient descent algorithm.

Although classical regularization as described above works adequately, in this experiment, we recommend a Bayesian approach to regularization which we describe in the next-subsection. We advocate that the Bayesian approach to regularization allows for more flexibility by reducing the bias inherent to classical regularization (through the choice of regularization constant) and therefore leading to a higher performance.

4.2.1.2 A Bayesian approach for feed-forward neural networks

After we have explained what a feed-forward network is, in this section we present the theory behind our proposed approach to regularization. The networks are trained using supervised learning, with a training data set of inputs and targets $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_N, y_N)\}$. We choose an interpolating functions of the form

$$g(\mathbf{x}) = \sum_{h=1}^{k} w_h \phi_h(\mathbf{x}), \tag{21}$$

where $\phi_h(\mathbf{x})$ are basis functions and w_h are coefficients inferred from the data. We assume that the targets are generated by

$$y_i = g(\mathbf{x}_i) + \epsilon_i, \tag{22}$$

where $g(\mathbf{x}_i)$ is an unknown function and ϵ_i are independent Gaussian random variables with average zero and variance σ^2. The initial objective of the training process is to minimize the sum of squared errors

$$E_D = \sum_{i=1}^{N} \frac{1}{2}(y_i - \widehat{y_i})^2, \tag{23}$$

where $\widehat{y_i}$ represents the neural network response for observation i.

An extensive work on Bayesian estimation and regularization has been done by MacKay (1992). In summary the Bayesian regularization requires the Hessian matrix of the objective function. For the MSE cost function and regularization by the sum of squared weights, it follows that the Hessian matrix is a quadratic function and it can be approximated by using the Levenberg-Marquardt algorithm (Gill and Murray, 1978). The objective function becomes

$$F = \alpha E_W + \beta E_D, \tag{24}$$

where E_W was defined in Eq. (20), and α and β are objective function parameters.

In the Bayesian framework (Foresee and Hagan, 1997) the weights of the network are considered random variables. Given the data, the probability density function of an array $\mathbf{w}$ of network weights is

$$f(\mathbf{w}|D, \alpha, \beta, M) = \frac{f(D|\mathbf{w}, \beta, M) f(\mathbf{w}|\alpha, M)}{f(D|\alpha, \beta, M)}, \tag{25}$$

where M is the particular neural network model used; $f(\mathbf{w}|\alpha, M)$ is the prior density,

which represents our knowledge of the weights before any data are collected; $f(D|\mathbf{w}, \beta, M)$ is the likelihood function, which is the probability of the data occurring given the weights; $f(D|\alpha, \beta, M)$ is a normalization factor, which guarantees that the total probability is 1.

Under the assumption of Gaussian noise, the probability of the data given the parameters $\mathbf{w}$ is

$$f(D|\mathbf{w}, \beta, M) = \frac{\exp(-\beta E_D)}{Z_D(\beta)}, \tag{26}$$

where $Z_D(\beta) = \left(\frac{2\pi}{\beta}\right)^{\frac{N}{2}}$, $\beta = \frac{1}{\sigma^2}$. The density of the prior can be written as

$$f(\mathbf{w}|\alpha, M) = \frac{\exp(-\alpha E_W)}{Z_W(\alpha)}, \tag{27}$$

where $Z_W(\alpha) = \int \exp(-\alpha E_W) d\mathbf{w}$. If Eq. (26) and Eq. (27) are substituted into Eq. (25), we obtain

$$f(\mathbf{w}|D, \alpha, \beta, M) = \frac{\exp(-(\beta E_D + \alpha E_W))}{Z_W(\alpha) Z_D(\beta)} = \frac{\exp(-F(\mathbf{w}))}{Z_F(\alpha, \beta)}. \tag{28}$$

where $Z_F(\alpha, \beta) = \int \exp(-F) d\mathbf{w}$. In this Bayesian framework, the optimal weights should maximize the posterior probability.

4.2.1.3 Optimizing the regularization parameters After we showed that the weights are a function of the parameters α and β, we optimize these parameters using Bayes' theorem,

$$f(\alpha, \beta|D, M) = \frac{f(D|\alpha, \beta, M) f(\alpha, \beta|M)}{f(D|M)}. \tag{29}$$

If a uniform prior density $f(\alpha, \beta|M)$ is taken for the regularization parameters α and β, then maximizing the posterior is achieved by maximizing the likelihood function $f(D|\alpha, \beta, M)$. However, note that this likelihood function is the normalization factor in Eq. (25). Since all probabilities have a Gaussian form, the posterior can expressed as

$$f(D|\alpha, \beta, M) = \frac{f(D|\mathbf{w}, \beta, M) f(\mathbf{w}|\alpha, M)}{f(\mathbf{w}|D, \alpha, \beta, M)} = \frac{Z_F(\alpha, \beta)}{Z_W(\alpha) Z_D(\beta)}. \tag{30}$$

$Z_D(\beta)$ and $Z_W(\alpha)$ are known from from Eq. (26) and Eq. (27). $Z_F(\alpha, \beta)$ can be expressed as a Taylor series expansion. Since the objective function has a quadratic shape in the surrounding of the minimum, we can expand $Z_F(\mathbf{w})$ around the minimum point of the posterior density $\mathbf{w}_{\mathrm{MP}}$, where the gradient is zero. We refer to $\mathbf{w}_{\mathrm{MP}}$ as the most probable

interpolant and therefore F can be written as

$$F = F(\mathbf{w}_{\mathrm{MP}}) + \frac{1}{2}(\mathbf{w} - \mathbf{w}_{\mathrm{MP}})^{\mathsf{T}}\mathbf{H}(\mathbf{w} - \mathbf{w}_{\mathrm{MP}}), \tag{31}$$

where $\nabla^2 E_D = \mathbf{B}$, $\nabla^2 E_W = \mathbf{C}$, $\mathbf{H} = \alpha\mathbf{C} + \beta\mathbf{B}$, $\mathbf{w}_{\mathrm{MP}} = \mathbf{H}^{-1}\mathbf{B}\mathbf{w}_{\mathrm{ML}}$. It follows that Z_F is a Gaussian integral that can be expressed as

$$Z_F = e^{-F(\mathbf{w}_{\mathrm{MP}})}(2\pi)^{\frac{k}{2}}(\det \mathbf{H})^{-\frac{1}{2}}. \tag{32}$$

Thus we can rewrite the log evidence for α and β as

$$\log f(D|\alpha, \beta, A, R) = -\alpha E_W - \beta E_D + \frac{k}{2}\log(2\pi) - \frac{1}{2}\log \det\mathbf{H} - \log Z_W(\alpha) - \log Z_D(\beta). \tag{33}$$

Notice that this expression contains the logarithm of the Occam factor $(2\pi)^{\frac{k}{2}}(\det \mathbf{H})^{-\frac{1}{2}}/Z_W(\alpha)$, which can control the overfitting. Substituting Z_D from Eq. (26) and Z_W from Eq. (27),

$$\log f(D|\alpha, \beta, A, R) = -\alpha E_W - \beta E_D - \frac{1}{2}\log \det\mathbf{H} + \frac{k}{2}\log\alpha + \frac{N}{2}\log\beta. \tag{34}$$

We differentiate the log evidence with respect to α and β to find the condition that is satisfied at the maximum. Differentiating with respect to α and setting the result equal to zero gives

$$\alpha_{\mathrm{MP}} = \frac{\gamma}{2E_W(\mathbf{w}_{\mathrm{MP}})}; \tag{35}$$

differentiating with respect to β gives and setting the result equal to zero gives

$$\beta_{\mathrm{MP}} = \frac{N - \gamma}{2E_D(\mathbf{w}_{\mathrm{MP}})}. \tag{36}$$

One step of the calculation is $\frac{\partial}{\partial\alpha}\log\det\mathbf{H} = \mathrm{tr}\left(\mathbf{H}^{-1}\frac{\partial\mathbf{H}}{\partial\alpha}\right) = \mathrm{tr}\,(\mathbf{H}^{-1}\mathbf{I}) = \mathrm{tr}\,\mathbf{H}^{-1} = (\mathrm{tr}\,\mathbf{H})^{-1}$, where $\nabla\nabla^{\mathsf{T}}E_W = \mathbf{I}$. Here $\gamma = k - 2\alpha_{\mathrm{MP}}\,\mathrm{tr}\,\mathbf{H}_{\mathrm{MP}}^{-1}$ is the effective number of parameters and k is the total number of parameters in the network. The parameter γ is a measure of how many parameters in the neural network are effectively used in reducing the error function; it can range from zero to k.

Summarizing, the steps required for the Bayesian optimization of the regularization parameters with a quadratic approximation of the Hessian matrix are:

1. Initialize the parameters α, β and the weights $\mathbf{w}$.
2. Take one step of the Levenberg-Marquardt algorithm to minimize the objective function $F(\mathbf{w}) = \alpha E_W + \beta E_D$.

3. Compute the effective number of parameters $\gamma = k - 2\alpha \operatorname{tr} \mathbf{H}^{-1}$ using the Gauss-Newton approximation of the Hessian available in the Levenberg-Marquardt training algorithm, $\mathbf{H} = \nabla^2 F(\mathbf{w}) \approx 2\beta \mathbf{J}^\mathsf{T}\mathbf{J} + 2\alpha \mathbf{I}_k$, where $\mathbf{J}$ is the Jacobian matrix of the training set errors.

4. Compute new estimates for the objective function parameters $\alpha = \frac{\gamma}{2E_W(\mathbf{w})}$, $\beta = \frac{N-\gamma}{2E_D(\mathbf{w})}$.

5. Iterate steps (ii) through (iv) until convergence.

4.2.1.4 Markov chain Monte Carlo estimation for α and β We propose an improvement on the estimation of the regularization parameters developed by MacKay (1992). We advocate applying a Markov chain Monte Carlo (MCMC) scheme to estimate α and β rather than approximating $Z_F(\alpha, \beta)$ and consequently approximating the Hessian matrix to estimate the parameters α, β. Collecting these paramenters into the two-dimensional vector $\mathbf{x}$ and indicating their estimate with $\mathbf{X}$, the MCMC method (Gelfand and Smith, 1990) can be described as

1. Choose the target distribution on $\mathbf{X}$ with density $\pi(\mathbf{x})$.

2. Choose the proposal distribution q: for any $\mathbf{x} \in \mathbb{R}^2_+$ we have $q(\mathbf{x}|\mathbf{x}) \geq 0$, $\int q(\mathbf{x}|\mathbf{x})d\mathbf{x} = 1$.

3. Starting with $\mathbf{X}_1$, for $t = 2, 3, \ldots, M$, sample $\mathbf{X}_* \sim q(\cdot|\mathbf{X}_{t-1})$.

4. Compute $\alpha(\mathbf{X}_*|\mathbf{X}_{t-1}) = \min\left\{1, \frac{\pi(\mathbf{X}_*)q(\mathbf{X}_{t-1}|\mathbf{X}_*)}{\pi(\mathbf{X}_{t-1})q(\mathbf{X}_*|\mathbf{X}_{t-1})}\right\}$.

5. Sample $U \sim U_{(0,1)}$. If $U < \alpha(\mathbf{X}_*|\mathbf{X}_{t-1})$, set $\mathbf{X}_t = \mathbf{X}_*$, otherwise set $\mathbf{X}_t = \mathbf{X}_{t-1}$.

We apply a standard normal prior distribution to the MCMC scheme.

4.2.2 Application of neural networks to financial data

As discussed in the literature, neural networks are a powerful concept that can be applied to different problems ranging from function approximation to clustering. There are many studies devoted to the comparison of neural networks to each other or to other algorithms. Specht (1990) investigated probabilistic neural networks; Wang and Peng (2000) explored vector-quantization networks; Stallkamp et al. (2012) compared convolutional neural networks with linear discriminant analysis and decision trees. In contrast to the above studies, in our analysis we focus on the concept of regularization and how it is applied in the context of neural networks. We test the performance of feed-forward networks with and without regularization. We report our results as an average performance over a range of different network architectures, i.e. combinations of layers and neurons.

Overfitting is one of the main challenges faced by statisticians today. The volume and the complexity of the data increase every year, which requires special attention to not overfit the classification algorithm. In this work, we use a combination of different network architectures combined with early stopping (ES) and regularization to tackle the problem of overfitting. We define ES as the process where we monitor the test error in n consecutive runs, while training the network. If the test error increases n times then the training of the network is terminated. We used $n = 6$, which is a typical choice for most classification problems. Furthermore, we combine the regularization with ES. However, the main focus of the analysis is on the Bayesian approach to regularization for neural networks. In contrast to the classical approach to regularization, in the Bayesian approach the regularization parameters are inferred from the data. We propose an improvement of the Bayesian estimation over the one suggested by MacKay (1992). Our estimation approach provides objectivity to the estimation and reduces the bias. MacKay (1992) proposed a Gauss-Newton approximation to the posterior distribution of the regularization parameters. In this Gauss-Newton approximation an objective function with parameters α and β is maximized. MacKay (1992) proposed an iterative solution for α and β by applying the Levenberg-Marquardt algorithm. On the other hand we apply a MCMC scheme to estimate the regularization parameters. In our approach α and β are considered random variables and are based on the mean of a posterior distribution. Finally, we compare the improved Bayesian regularization approach to the classical regularization and to the Bayesian regularization based on the Gauss-Newton approximation.

We now apply the methodology proposed in Section 4.2.1 to three different data sets on 1. corporate obligors based in Eastern Europe, see 3.1, 2. corporate obligors based in Poland, see 3.2, and 3. retail obligors based in Germany, see 3.3. We use data on corporates from Poland and Eastern Europe because these are developing markets where the relations between the risk factors and the default event are not yet well investigated. In a developing market the group of default drivers could be significantly different from what we observe in a developed market. By using data from developing markets we try to find out whether the default drivers in these markets are significantly different from the default drivers in developed markets. Finally, we examine retail data from a developed market to check whether the default identification of our proposed algorithm is adequate on a data set that is not corporate.

4.2.2.1 Feature selection The literature offers a variety of algorithms for variable selection such as filter and wrapper methods. However, the main goal of our analysis is to examine the effect of Bayesian regularization on ANNs. Therefore, we apply a simple approach to variable selection based on the 80% percentile of the vector containing the

absolute value of the correlation with the target variable. We select only variables whose correlation is equal or above the 80% percentile of the vector containing the absolute value of the correlation with the target variable. This leads to a balanced number of variables that are shown in Table 18. Nonetheless, not to bias our results based on a single combination of variables, we report our results for different numbers of variables by changing the percentile value from 0% to 90%; see appendix B. This is consistent with the principle applied in Sariev and Germano (2019), where model performance is assessed comparing a model on a different set of variables.

Table 18: Selected variables by data set based on the 80% percentile of the correlation to the target variable.

Data set	Selected variables
East-European data	payables turnover, return on assets, cash ratio, income from sales/total assets, liquid assets/total assets, interest coverage
Polish data	total costs/total sales, (sales − cost of products sold)/sales, profit on sales/sales, working capital, sales(n)/sales($n-1$), sales/inventory, working capital/total assets, sales/receivables, short-term liabilities/total assets, total liabilities/total assets, sales/total assets, logarithm of total assets
German data	duration in months of the account, credit history, checking account status

4.2.2.2 Results Table 19 presents eight different feed-forward neural network architectures. Prior to applying the networks on the data we need to make a choice on the number of neurons and the number of layers for each network. Determining the number of neurons and layers is driven by many factors such as: the number of variables in the model, the number of data points and etc. In order to avoid reporting biased result we run each network on a range of different combinations of layers and neurons. This allows us to monitor the performance of the network over different network architectures and allows us to summarize the network performance. Although there is no clear rule on selecting the number of neurons and layers, we follow Hagan et al. (2014) who argue that the number of neurons should be lower than the number of variables used in the network. Further, the number of hidden layers should not be more than 2-3 because most problems are tackled even with one hidden layer. Adding many hidden layers on data sets that are not big (more than one million observations) does not result in a better performance. Therefore, we decide to report the performance of the network on a combination of: neurons that range from 1 to 25 and hidden layers that range from 1 to 3. We investigate combinations

with more neurons than Hagan et al. (2014) suggests, so that our results are more comprehensive. However, we show that increasing the number of layers does not lead to a higher performance; see Section 4.2.2.3.

We acknowledge that our decision on the number of neurons and layers is subjective, but we aim to cover a wide enough range of neurons and layers so that our results are less biased than using just a single combination of hidden layers and neurons. For the classical regularization, the regularization parameter should be determined before applying the network to the data. Based on 10 fold cross-validation we estimated the regularization parameter (γ, see Eq. (20)) for classical regularization to be 0.05. All networks are trained with MSE loss function. One third of the data are left for testing the networks, two-thirds of the data are allocated for training and validating.

Following the above logic we report our results in Table 19. The first column in Table 19 shows the architecture type; the second column shows the regularization type used: classical, Bayesian and Bayesian based on MCMC; the third column indicates whether ES was applied or not. The fourth to seventh columns give the mean percentage of correctly classified observations, the percentage of non-defaulted obligors, the percentage of defaulted obligors and the Gini coefficient on the grid of 1 to 25 neurons and 1 to 3 hidden layers. All results are reported on test data. Finally, the eight column presents the average CPU time in seconds to compute a network on an Intel Celeron N2840 with 2.16GHz.

Based on Table 19, we observe that:

1. The percentage of overall correctly classified obligors is the highest for a network architecture where the regularization parameters are estimated by the Bayesian approach with the proposed MCMC estimation rather than the Gaussian approximation.

2. In some cases the ES procedure can lead to a better performance but in other cases ES undermines the network performance.

3. The computational time needed for the MCMC estimation of the network is significantly higher than for the other networks but the Bayesian estimation automatically estimates the regularization thus reducing the bias.

The above observations are valid for all data sets. Below we examine the results for each data set separately.

1. For the East-European data, see 3.1, Bayesian regularization with MCMC leads to the highest overall performance. The improvement in performance is 4%, which is high enough to make a difference from a practical point of view. In terms of

identification of bad obligors and Gini coefficient, Bayesian regularization with MCMC performs similarly to classical regularization.

2. For the Polish data, see 3.2, Bayesian regularization with MCMC leads to the highest overall performance. The improvement in performance is 1%, which can be ignored for practical purposes. However, in terms of identification of bad obligors and Gini coefficient, Bayesian regularization with MCMC performs significantly better than the other methods.

3. For the German data, see 3.3, Bayesian regularization with MCMC leads to the highest overall performance. The improvement in performance is 2%, which makes a difference in situation where overall performance is of utmost importance. However, in terms of identification of bad obligors and Gini coefficient, Baeysian regularization with MCMC under-performs compared to the other methods.

The results in Table 19 are based on the 80% percentile of the correlation to the target variable. In Appendix B one can see the results for the other combinations of variables but the conclusion stays the same. In all cases Bayesian regularization with MCMC overall outperforms the other methods. The orginial data set has been randomly sampled 10 times, each time sampling 60% of it, ensuring defaulted and non-defaulted observations are equal in numbers. Afterwards, a training, test and validation sets have been generated based on the sampled data, in the following proportions: 70%-training data,15%-validation data, 15%-test data. Finally we computed a standard deviation based on the 10 runs. In all cases the standard deviation is in the range of 0.1% to 1.4%, which is in the expected range.

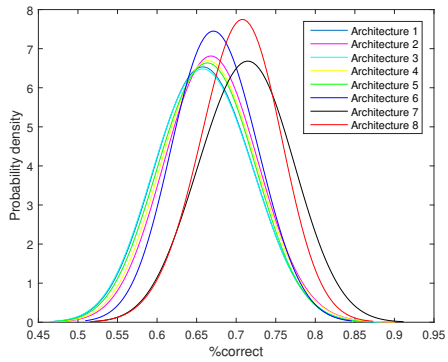
Finally, in Table 29 in Appendix B we apply a two-sample t-test on the overall performance of our proposed method namely Architecture 7 and 8. The two-sample t-test is a parametric test that compares the location parameter of two independent data samples. The test statistics of the test follow a Student's t distribution. The null hypothesis states that the means of two populations are equal. In Table 29, 1 indicates a rejection of the null hypothesis. Therefore, we can claim our results are statistically different for each data set.

Table 19: Performance of the ANNs on the East-European, Polish and German test data when using factors based on the 80% percentile of the correlation to the target variable. Correct: the percentage of overall correctly classified obligors; Good: the percentage of correctly classified good obligors; Bad: the percentage of correctly classified bad obligors; Gini: the Gini coefficient; CPU time/s: the CPU time in seconds needed for one run of the network.

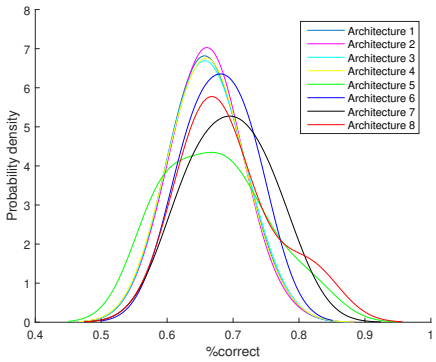
Architecture	Regularization	ES	Correct	Good	Bad	Gini	CPU time/s
East-European data							
1	No	No	66%(0.7)	59%	73%	59%	1.2
2	No	Yes	67%(0.8)	58%	75%	61%	0.9
3	Classical	No	66%(0.7)	58%	73%	59%	0.9
4	Classical	Yes	67%(0.8)	58%	75%	60%	0.9
5	Bayesian	No	66%(1.3)	59%	73%	55%	1.6
6	Bayesian	Yes	67%(1.4)	73%	62%	50%	1.7
7	Bayesian MCMC	No	71%(0.4)	69%	74%	61%	19.1
8	Bayesian MCMC	Yes	70%(0.5)	70%	70%	57%	18.3
Polish data							
1	No	No	67%(0.3)	75%	57%	52%	0.8
2	No	Yes	65%(0.3)	75%	56%	52%	0.8
3	Classical	No	67%(0.5)	79%	54%	53%	0.9
4	Classical	Yes	65%(0.4)	76%	55%	52%	0.8
5	Bayesian	No	63%(0.8)	68%	52%	55%	1.5
6	Bayesian	Yes	64%(0.6)	87%	39%	53%	1.6
7	Bayesian MCMC	No	68%(0.5)	75%	64%	52%	17.1
8	Bayesian MCMC	Yes	68%(0.7)	66%	69%	56%	16.4
German data							
1	No	No	68%(0.7)	63%	72%	60%	1.2
2	No	Yes	67%(0.1)	63%	71%	61%	1.0
3	Classical	No	68%(0.1)	61%	74%	61%	1.3
4	Classical	Yes	67%(0.8)	61%	74%	61%	1.0
5	Bayesian	No	66%(0.3)	67%	65%	57%	2.0
6	Bayesian	Yes	61%(1.0)	43%	75%	55%	2.3
7	Bayesian MCMC	No	68%(0.3)	65%	70%	57%	20.6
8	Bayesian MCMC	Yes	70%(0.4)	72%	67%	58%	19.4

Figure 16 presents distributions of the overall correctly classified obligors per data set and per network architecture that are shown in Table 19. We can see from Figure 16 that the

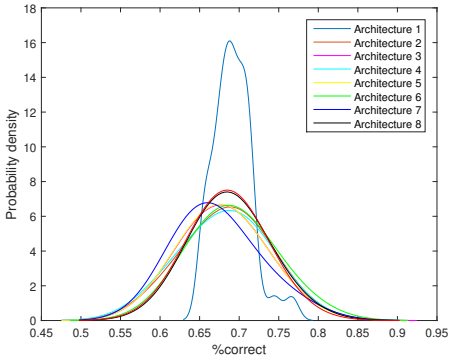
distribution of the overall correctly classified obligors for Architectures 7 and 8 is right skewed for the East-European and Polish data. The results in Figure 16 are based on the 80% percentile of the correlation to the target variable. In Appendix B one can see the results for the other combinations of variables but the conclusion stays the same. In all cases Bayesian regularization with MCMC overall outperforms the other methods.



(a) East-European data



(b) Polish data



(c) German data

Figure 16: Distribution of the overall correctly classified obligors for the East-European (a), Polish (b) and German (c) data. Results are based on the 80% percentile of the correlation to the target variable.

4.2.2.3 Neural network performance on increasing the number of layers Inspired by the flexibility of the deep neural network paradigm, we tried to increase the number of layers with the goal of increasing the performance on the test data. However, contrary to our expectations the generalization power of the network decreased for each data set, as can be seen from Figure 17. The decrease in performance is different for each data set. For the Polish and German the decrease of performance is not significant but for the East-European data the decrease is significant. The reason is that our data sets are not big enough to allow the application of many layers. The networks with more than 4 layers

significantly overfit the data.

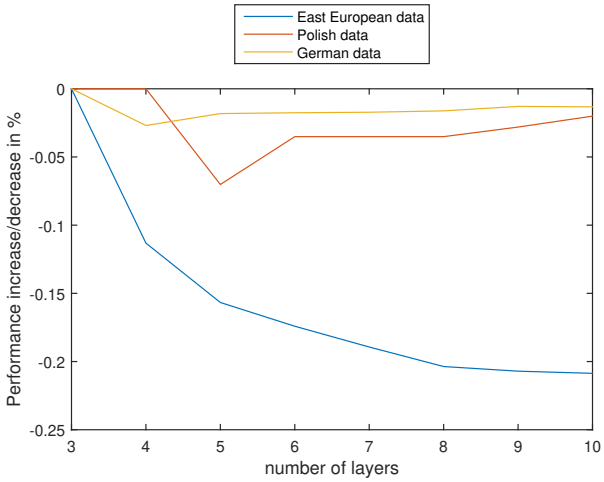


Figure 17: Percentage increase/decrease per number of layers for each data set.

4.2.2.4 Comparison to other classification algorithms The main objective of our research is to analyse the effect of Bayesian regularization and compared it to classical regularization for ANNs. We report our results as an average over different combinations of layers and neurons. Therefore, we do not report the maximum classification accuracy that can be achieved rather we aim to present the effect of Bayesian regularization with MCMC over different network architectures and advocate that on average our proposed approach leads to higher performance when compared to other regularization approaches for ANNs. However, for the purpose of completeness we apply two other non-linear classification methods to the three data sets. The first is SVMs and the second is KNNs. The results in terms of classification accuracy are shown in Table 20. Overall the performance is similar to our proposed method. On the Polish corporate data SVMs and KNNs outperform but as we emphasised before the results we report in Table 19 are averaged over a grid of different neurons and layers and therefore are not directly comparable to the results in Table 20. Therefore, the performance reported in Table 19 is not the highest that could be achieved using Bayesian regularization but this average performance is close to the maximum performance we achieve when we apply SVMs and KNNs to the data.

Table 20: Overall accuracy (percent of classified obligors) by SVMs, KNNs and ANNs. The results are shown per variable selection combination based on the 0%, 50%, 60%, 70%, 80% and 90% percentile of the correlation to the target variable.

Percentile	East-European data			Polish data			German data		
	SVMs	KNNs	ANNs	SVMs	KNNs	ANNs	SVMs	KNNs	ANNs
0%	0.62	0.63	0.69	0.70	0.71	0.64	0.49	0.70	0.67
50%	0.65	0.63	0.70	0.73	0.74	0.67	0.63	0.62	0.70
60%	0.69	0.63	0.71	0.69	0.73	0.66	0.69	0.67	0.70
70%	0.67	0.62	0.69	0.69	0.73	0.67	0.71	0.63	0.69
80%	0.66	0.62	0.70	0.72	0.74	0.68	0.65	0.57	0.70
90%	0.66	0.61	0.68	0.72	0.69	0.74	0.62	0.65	0.68

4.2.3 Policy implications

Identifying a classification method to estimated the PD is an important factor but equally important is deriving business intuition from the selected default factors. Typically PD models are used by non-technical audience and the interpretation of the default factors from an industry prospective is of utmost importance. For that reason we split the selected ratios into three categories[1]

1. Leverage category — ratios that signal how much debt and debt related costs a company utilizes against company's equity or assets. Effectively this category indicates the level of indebtedness of a company.

2. Profitability category — ratios that signal the ability of a company to generate income relative to its equity or assets. Effectively this category indicates how efficiently a company utilizes it assets.

3. Liquidity category — ratios that signal a company ability to meet the current liabilities when they become due with its current assets. Effectively this category indicates the ability of a company to pay off its short-term obligations.

Table 21 presents the allocation of the selected ratios from the variable selection method on the two corporate data sets (East-European and Polish) to each of the above three categories.

[1] payables turnover = supply payables$\times$360/cost of goods sold from the East-European data and working capital/total assets as well as logarithm of total assets from the Polish data cannot be allocated to these three groups

Table 21: Selected ratios based on the 80% percentile of the correlation to the target, allocated into three main financial categories: leverage, profitability and liquidity on the East-European (E) and on the Polish (P) data.

Category	Ratio
Leverage ratios	interest coverage (E), short-term liabilities/total assets (P), total liabilities/total assets (P), total costs/total sales (P)
Profitability ratios	return on assets (E), income from sales/total assets (E), sales/total assets (P), sales/inventory (P), sales/receivables(P), sales(n)/sales($n-1$) (P), profit on sales/sales (P), (sales $-$ cost of products sold)/sales(P)
Liquidity ratios	cash ratio (E), liquid assets/total assets (E), working capital (P)

As can be seen from Table 21 the default risk in the Polish data set is driven mainly by the profitability ratios, followed by the leverage ratios. Liquidity ratios don't play an important role in determining the default risk of the Polish obligors. We compare our approach with that of Liang et al. (2016) where they split the financial ratios into several groups namely: solvency(leverage) ratios, profitability ratios, capital structure ratios, cash flow ratios, ownership ratios, turnover ratios ratios, growth ratios. They found that leverage and profitability ratios are the most important categories in identifying defaults. Interestingly they have used data from the Taiwan Stock Exchange. The fact that their findings align with ours proofs the significance and the universality of the leverage and profitability ratios. Another study by Al-Kassar and Soileau (2014) also indicates the importance of profitability and leverage ratios through the use of factor analysis. However, they advocate non-financials data are also important in identifying and measuring default risk. Furthermore a study by Chen et al. (2011) emphasise the role of the profitability and leverage ratios. The analysis is done on 20000 solvent and 1000 insolvent companies. Their study applies SVMs on German companies and shows the importance of profitability and leverage in identifying defaults.

Similarly the default risk in the East-European data set is driven by profitability and leverage ratios but it is also driven by liquidity ratios. We compare our approach with that of Marilena and Alina (2015) where liquidity and leverage ratios are identified as a major default driver. Their work applies multiple discriminant analysis, logistic regression analysis, and artificial neural networks analysis. The data used are from the Bucharest Stock Exchange principal market. Moreover a study by Tian et al. (2015) also indicates the importance of liquidity and leverage ratios. They use North American financial data on corporate obligors and apply the LASSO method for variable selection. Finally, Tian et al.

(2015) claim their approach is superior to the one given by the popular distance to default model proposed by Merton (1974).

We note that the major difference in default drivers between the East-European data and the Polish data is the higher importance of liquidity ratios for the former, the reason being that Polish Companies are in general bigger and liquidity is not a major indication of default risk. Practically, larger companies have access to cheaper funding, whereas smaller companies incur higher funding costs.

Due to the low number of selected features in the German retail data set we are not able to allocate them into different groups. However, most of the variables in the German retail data are based on the status and duration of the current account. This is aligned with the study of Barrell et al. (2010), which shows evidence that the status of the current account is a major predictor of mortgage defaults.

4.2.4 Discussion and conclusions

In this experiment we propose an improvement of a Bayesian approach to regularize feed-forward neural networks. The Bayesian approach is attractive because it provides automatic determination of the regularization parameters. Moreover, we demonstrate that the improved Bayesian approach performs well when compared to the classical regularization approach for neural networks. We find that using a MCMC scheme to estimate the Bayesian regularization parameters leads to a higher performance than using a Gauss-Newton approximation. Furthermore, the application of ES on the network does not guarantee higher performance.

We analysed three data sets; two are corporate and one retail. From a policy prospective three groups of financial ratios are identified as major drivers of default risk: profitability ratios, leverage ratios and liquidity ratios. The effect of liquidity ratios is higher on the East-European data and the of profitability ratios is higher on the Polish data.

The findings of this experiment yield promising insights into the potential of Bayesian regularization to efficiently estimate the network weights. Practically, this leads to making better-informed and less biased credit risk decisions.

4.3 Experiment 3

With respect to the above discussed studies on KNNs, see 2.2.3, this experiment contributes to the literature first by first using a GA algorithm to estimate BKNNs and second by applying innovative MCMC proposal schemes to estimate the BKNNs.

The rest of the thesis is organized as follows. Section 4.3.1 presents the theoretical formulation of KNNs in a Bayesian framework. Section 4.3.2 presents the results from the BKNN and the results from other classification methods that are used as a benchmark. Section 4.3.3 discusses the business intuition of the proposed default drivers. Finally, section 4.3.4 concludes the experiment by summarizing the main findings of this research.

4.3.1 Theoretical foundations

Consider a data set $D = \{(\mathbf{Y}_1, \mathbf{X}_1), (\mathbf{Y}_2, \mathbf{X}_2)\}$ with $n = n_1 + n_2$ observations, where $(\mathbf{Y}_1, \mathbf{X}_1) = \{(y_{11}, \mathbf{x}_{11}), (y_{12}, \mathbf{x}_{12}), \ldots, (y_{1n_1}, \mathbf{x}_{1n_1})\}$ is the training set with n_1 observations, and $(\mathbf{Y}_2, \mathbf{X}_2) = \{(y_{21}, \mathbf{x}_{21}), (y_{22}, \mathbf{x}_{22}), \ldots, (y_{2n_2}, \mathbf{x}_{2n_2})\}$ is the test set with n_2 observations. $\mathbf{X}_1 = (\mathbf{x}_{11}, \ldots, \mathbf{x}_{1n_1})$ and $\mathbf{X}_2 = (\mathbf{x}_{21}, \ldots, \mathbf{x}_{2n_2})$ are $n_1 \times p$ and $n_2 \times p$ data matrices for training and test, respectively. The subscript 1 indicates the training set, whereas the subscript 2 indicates the test set. $\mathbf{Y}_1$ is a $1 \times n_1$ vector of known class labels for the training set, whereas $\mathbf{Y}_2$ is a $1 \times n_2$ unknown vector and must be predicted. The dimension of the predictors is p and n is the total number of observations. Suppose that there are Q classes, $y_{li} \in 1, 2, \ldots, Q, \;\; l = 1, 2, \;\; i = 1, 2, \ldots, n_l$. We wish to classify the points in the test set by assigning them to one of the Q classes, based on the known information in the training set. In short, we need to predict the unknown class memberships $\mathbf{Y}_2$. In this case the likelihood function is

$$f(\mathbf{Y}|\mathbf{X}, \beta, k) = \prod_{i=1}^{n} \frac{\exp\left(\frac{\beta}{k} \sum\limits_{j \overset{k}{\sim} i} \delta_{y_{1i} y_{1j}}\right)}{\sum\limits_{q=1}^{Q} \exp\left(\frac{\beta}{k} \sum\limits_{j \overset{k}{\sim} i} \delta_{q y_{1j}}\right)}, \tag{37}$$

where β acts as the parameter for the strength of association among nearest neighbours and k is the neighbourhood size required to construct a KNNs classifier. The δ is the Kronecker delta, defined as $\delta_{ab} = 1$ if $a = b$ and zero otherwise. In the likelihood, the term $\frac{1}{k} \sum\limits_{j \overset{k}{\sim} i} \delta_{q y_{1j}}$ calculates the proportion of training points among the k-nearest neighbours of

$\mathbf{x}_i$ belonging to class q. The predictive distribution for a new observation is

$$f(y_{n+1}|\mathbf{x}_{n+1}, \mathbf{Y}, \mathbf{X}, \beta, k) = \frac{\exp\left(\frac{\beta}{k} \sum\limits_{j \overset{k}{\sim} n+1} \delta_{y_{n+1} y_{1j}}\right)}{\sum\limits_{q=1}^{Q} \exp\left(\frac{\beta}{k} \sum\limits_{j \overset{k}{\sim} n+1} \delta_{q y_{1j}}\right)}, \tag{38}$$

so the most probable class for y_{n+1} is given by the most common class found among its k-nearest neighbours. Treating β and k as known and fixed a priori is an unrealistic component of uncertainty in the model. To accommodate for this we assign prior probabilities to β and k, leaving the marginal predictive distribution as

$$f(y_{n+1}|\mathbf{x}_{n+1}, \mathbf{Y}, \mathbf{X}) = \sum_k \int f(y_{n+1}|\mathbf{x}_{n+1}, \mathbf{Y}, \mathbf{X}, \beta, k) f(\beta, k|\mathbf{Y}, \mathbf{X}) d\beta, \tag{39}$$

where

$$f(\beta, k|\mathbf{Y}, \mathbf{X}) \propto f(\mathbf{Y}, \mathbf{X}|\beta, k) f(\beta, k). \tag{40}$$

We have little prior knowledge about the likely values of β and k, other than the fact that β should be positive. Hence we adopt independent default probability densities

$$f(k) = f_u(1, \ldots, k_{\max}) = \frac{1}{k_{\max}}, \;\; k_{\max} = s \tag{41}$$

$$f(\beta) = c I_{\mathbb{R}_+}(\beta), \tag{42}$$

where f_u is a uniform discrete probability function, c and s are constants and I is an indicator function prior so that the prior on β is uniform on $\mathbf{R}_+$. A proper prior could also be applied in the form of

$$f(\beta) = 2N(0, \sigma^2) I_{\mathbb{R}_+}(\beta), \tag{43}$$

where $\sigma^2 = c$. We propose Markov chain Monte Carlo (MCMC) to draw M samples from $f(\beta, k|\mathbf{Y}, \mathbf{X})$ and then to approximate the distribution by

$$f(y_{n+1}|\mathbf{x}_{n+1}, \mathbf{Y}, \mathbf{X}) \approx \frac{1}{M} \sum_{i=1}^{M} f(y_{n+1}|\mathbf{x}_{n+1}, \mathbf{Y}, \mathbf{X}, \beta^{(i)}, k^{(i)}), \tag{44}$$

where $\beta^{(i)}, k^{(i)}$ represent the sample i in the converged chain. We use a single joint proposal with

$$\widehat{k} = k \pm U_{(0,4)}, \tag{45}$$

$$\widehat{\beta} = \beta + N(0, \sigma^2), \tag{46}$$

where $\sigma^2 = c = 1/2$. We impose a reflection at the boundaries of the prior range in density, so that if $\widehat{\beta} < 0$ it is reset to $|\widehat{\beta}|$. The MCMC method can be described as

1. Choose the target distribution on $\mathbf{X}$ with density $\pi(\mathbf{x})$.
2. Choose the proposal distribution q: for any $\mathbf{x}, \in \mathbb{R}_+$ we have $q(\mathbf{x}|\mathbf{x}) \geq 0$, $\int q(\mathbf{x}|\mathbf{x})d\mathbf{x} = 1$.
3. Starting with $\mathbf{X}^{(1)}$, for $t = 2, 3, \ldots, M$, sample $\mathbf{X}^* \sim q(\cdot|\mathbf{X}^{(i-1)})$.
4. Compute $\alpha(\mathbf{X}^*|\mathbf{X}^{(i-1)}) = \min\left\{1, \frac{\pi(\mathbf{X}^*)q(\mathbf{X}^{(i-1)}|\mathbf{X}^*)}{\pi(\mathbf{X}^{(i-1)})q(\mathbf{X}^*|\mathbf{X}^{(i-1)})}\right\}$.
5. Sample $U \sim U_{(0,1)}$. If $U < \alpha(\mathbf{X}^*|\mathbf{X}^{(i-1)})$, set $\mathbf{X}^{(i)} = \mathbf{X}^*$, otherwise set $\mathbf{X}^{(i)} = \mathbf{X}^{(i-1)}$.

In the Metropolis-Hastings algorithm, pick $q(\mathbf{X}^*|\mathbf{X}) = g(\mathbf{X}^* - \mathbf{X})$ with g a symmetric distribution, e.g. a zero-mean multivariate normal or t-Student; thus

$$\mathbf{X}^* = \mathbf{X} + \epsilon, \tag{47}$$

where ϵ has a distribution g. The acceptance probability becomes

$$\alpha(\mathbf{X}^*|\mathbf{X}^{(i-1)}) = \min\left\{1, \frac{\pi(\mathbf{X}^*)}{\pi(\mathbf{X}^{(i-1)})}\right\}. \tag{48}$$

Applying MCMC to the target in Eq (40) and using the uniform priors in Eq (42), the proposals are then accepted with probability

$$\alpha(\mathbf{X}^*|\mathbf{X}^{(i-1)}) = \min\left\{1, \frac{p(\mathbf{Y}|\mathbf{X}, \widehat{\beta}, \widehat{k})}{p(\mathbf{Y}|\mathbf{X}, \beta, k)}\right\}. \tag{49}$$

In determining the neighbourhood of the point $\mathbf{x}_i$, the following distances are considered:

1. the Euclidean distance

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^\mathsf{T}(\mathbf{x}_i - \mathbf{x}_j)}; \tag{50}$$

2. the Mahalanobis distance

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^\mathsf{T}\mathbf{C}^{-1}(\mathbf{x}_i - \mathbf{x}_j)}, \tag{51}$$

where $\mathbf{C} = \frac{1}{n}(\mathbf{X} - E(\mathbf{X}))^\mathsf{T}(\mathbf{X} - E(\mathbf{X}))$ is the $p \times p$ covariance matrix of the vectors $\mathbf{x}_i,\ i = 1, \ldots, n$, forming the columns of the $p \times n$ matrix $\mathbf{X}$;

3. the cosine distance

$$d(\mathbf{x}_i, \mathbf{x}_j) = 1 - \widehat{\mathbf{x}}_i^{\mathsf{T}} \widehat{\mathbf{x}}_j, \tag{52}$$

where $\widehat{\mathbf{x}}_i = \mathbf{x}_i / x_i$ and $x_i = \|\mathbf{x}\| = d(\mathbf{x}_i, \mathbf{0}) = \sqrt{\mathbf{x}_i^{\mathsf{T}} \mathbf{x}_i}$ (i.e., the Euclidean norm of $\mathbf{x}_i$).

4.3.1.1 Proposed MCMC updates Inspired by the work of Atchade (2006), we explore alternative MCMC algorithms, where the proposal distribution was changed in one of three ways:

1. Use a combined rather than a single proposal distribution for β. The combined random variable that we applied can be written as

$$\widehat{\beta} = \sum_{k=1}^{n} \beta + N(0, \sigma_k^2), \tag{53}$$

where σ_k^2 is selected based on cross-validation. We investigated different ranges of σ_k^2 and then set the final range for σ_k^2 to [1,10]. This is done with the aim of exploring distributions with significantly different variances so that the mixing of the chain is improved. We highlight that our analysis is biased in terms of selecting a prior distribution and its hyperparameters. However, subjectivity can be interpreted as awareness of multiple perspectives and context dependence (Gelman and Hennig, 2017). Applying Bayesian analysis to our problem requires a certain amount of subjectivity, which can be used to spur more research questions rather than to invalidate the results of the analysis.

2. Randomize the variance of the normal distribution $N(0, \sigma^2)$ used to sample the parameter β, drawing it e.g. from a standard uniform variate, i.e. $\sigma^2 \sim U(0, 1)$. If $\widehat{\beta}_i \sim N(0, \sigma_i^2)$ was accepted on iteration i, then on successive iteration $i + 1$, the variance stays the same, i.e. $\sigma^2_{(i+1)} = \sigma_i^2$ and thus $\widehat{\beta}_{(i+1)} = N(0, \sigma_i^2)$.

3. Sample $\widehat{\beta} = \beta + N(0, 1) + \frac{\partial L(\beta)}{\partial \beta}$, where $L(\beta) = \exp\left(\frac{\beta}{k} \sum_{j \overset{k}{\sim} n} \delta_{y_{1i} y_{1j}}\right)$. Introducing the derivative of the numerator in Eq (37) can increase the mixing of the chain.

4.3.1.2 GA algorithm In addition to the MCMC updates suggested in 4.3.1.1, a GA algorithm is proposed to estimate β and k from Equation 37. The GA algorithm for mixed-integer problems is based on the following equations (Deb, 2000), (Deep et al., 2009).

$$F(\mathbf{x_i}) = f(\mathbf{x_i}) + \sum_{j=1}^{J} R_j \phi_j(\mathbf{x_i})^2, \tag{54}$$

where $F(\mathbf{x_i})$ is defined as the sum of the objective function $f(\mathbf{x_i})$, J-number of constraints, and a penalty term which depends on the constraint violation $\phi_j(\mathbf{x_i})$. The fitness function to be minimized $Fit(\mathbf{x_i})$ then is written as

$$Fit(\mathbf{x_i}) = \begin{cases} f(\mathbf{x_i}), & \text{if } F(\mathbf{x_i}) \text{ is fesible} \\ F_w + \sum_j^J \phi_j(\mathbf{x_i}) \ , & \text{otherwise} \end{cases}, \tag{55}$$

where, F_w is the objective function value of the worst feasible solution currently available in the population. Thus, the fitness of an infeasible solution not only depends on the amount of constraint violation, but also on the population of solutions at hand. However, the fitness of a feasible solution is always fixed and is equal to its objective function value. $\phi_j(\mathbf{x_i})$ refers to value of the left hand side of the inequality constraints (equality constraint are also transformed to inequality constraints using a tolerance).

The logical steps of the GA are outlined below (Goldberg, 1989) .

1. The algorithm begins by creating a random initial population.
2. The algorithm then creates a sequence of new populations. At each step, the algorithm uses the individuals in the current generation to create the next population. To create the new population, the algorithm performs the following steps
    1. Scores each member of the current population by computing its fitness value. These values are called the raw fitness scores.
    2. Scales the raw fitness scores to convert them into a more usable range of values. These scaled values are called expectation values.
    3. Selects members, called parents, based on their expectation.
    4. Some of the individuals in the current population that have best fitness are chosen as elite. These elite individuals are passed to the next population.
    5. Produces children from the parents. Children are produced either by making random changes to a single parent—mutation—or by combining the vector entries of a pair of parents—crossover.
    6. Replaces the current population with the children to form the next generation.
3. The algorithm stops when the average cumulative change in value of the fitness

function over the pre-specified number of generations is less than a predefined constant.

4.3.2 Application of BKNNs to financial data

We apply a sequential feature selection to choose a set of default predictors (Aha and Bankert, 1996). Starting from an empty feature set, sequential feature selection creates candidate feature subsets by sequentially adding each of the features not yet selected. For each candidate feature subset, sequential feature selection performs 10-fold cross-validation by repeatedly estimating the misclassification error with different training subsets

Going forward the final set of predictors is selected to be the same for each classification method. The list of final variables for the East-European data, for the Polish data and for the German data can be found in Table 25 in section 4.3.3.

The original data set is split into training and test data. The results below refer to the test data. As discussed in the literature, KNNs method is a flexible and popular method for classification problems. In this work we examine KNNs from both a classical and a Bayesian prospective. The MCMC of the posterior distribution of $\widehat{\beta}$ in the Bayesian estimation of KNNs is checked for convergence using a time series plot and an autocorrelation plot for $\widehat{\beta}$. The plots are produced for each MCMC version of the BKNNs: original sampling, combined proposal, randomized variance, derivative approach. The plots are shown in Appendix C.

The GA algorithm applied to BKNN is run with the following specifications.

1. Population size is set to 50.
2. Number of generations (iterations) is unbounded.
3. Penalty parameter is set to 10.

We observed that GA algorithm is stable with the respect of the above specifications see Figure 33 in Appendix E. Additionally, we tested the GA algorithm in regards to its pseudo-parameters such as population size, number of iterations and penalty parameter. In all cases the parameters produced by the algorithm did not change significantly see Table 38 in Appendix E. The GA algorithms converged in all cases.

The list of algorithms examined here does not pretend to be exhaustive but it contains the most popular methods for PD estimation. Other PD estimation methods exist (Bellotti and Crook, 2009a), but their exploration is not discussed here. We compare the BKNNs algorithm to the following family of popular classification algorithms: LR, LDA, naive

Bayes, DTs, ANNs SVMs. The flexibility of decision trees allows two different versions of it to be applied on the data: boosted D Ts and classical DTs method without any adjustments. The boosting is based on the Logitboost algorithm where the binomial deviance is minimized. We apply different architectures and number of neurons for the ANNs. Based on performance and simplicity, we use a feed-forward network with 1 hidden layer and 10 neurons using cross entropy as a loss function and logistic regression as an activation function. We apply a linear kernel to the SVMs.

The split between train and test sets for the East Europe data set, see 3.1, is 70% train and 30% test data. Both sets test and train are balanced, have equal number of defaulted and non-defaulted observations.

The split between train and test sets for the Polish data, see 3.2, is 75% train and 25% test data. Both sets test and train are balanced, have equal number of defaulted and non-defaulted observations.

The split between train and test sets for the German data, see 3.3, is 70% train and 30% test data. Both sets test and train are balanced, have equal number of defaulted and non-defaulted observations.

The train and test data are randomly sampled 10 times. Based on that sampling we calculated a standard deviation for the total number of correctly classified observations. The standard deviation for most algorithms is in the range of 0.4% to 3.3%, which is an acceptable level of variation based on updating the development sample on each trial.

4.3.2.1 Results on East-European corporate data A bootstrap procedure with replacement is applied on the data with the aim of constructing ten different train/test samples. Table 22 presents the mean output from the bootstrapping. Table 22 shows the percentage of overall correctly classified observations, the percentage of correctly classified observations for non-defaulted obligors, the percentage of correctly classified observations for defaulted obligors and the CPU time/s in seconds required for the estimation of each classification method.

Based on Table 22 several conclusions can be drawn:

1. Estimation of BKNN with a GA algorithm leads to the same results as MCMC estimation.
2. The classical KNNs method is under-performing when compared to the BKNNs.
3. The BKNN method in all their versions outperform all other benchmark methods aside from boosted DTs.

Table 22: Performance of the classification methods on the East-European test data. % correct: the percentage of overall correctly classified obligors; % good: the percentage of correctly classified good obligors; % bad: the percentage of correctly classified bad obligors; CPU time/s: the CPU time in seconds needed for one run of the method.

Method	% correct (std. in %)	% good	% bad	CPU
BKNNs, GA	70%(2.1)	70%	70%	251.02
BKNNs	70%(1.9)	69%	71%	43.70
BKNNs, randomized variance	70%(1.8)	71%	69%	414.7
BKNNs, combined proposal	70%(1.9)	68%	72%	76.80
BKNNs, derivative	70%(1.7)	68%	72%	44.30
KNNs	67%(1.9)	69%	65%	0.17
DTs	62%(2.7)	63%	61%	0.08
DTs, boost	70%(2.5)	70%	69%	5.06
ANNs	65%(1.3)	63%	67%	0.90
SVMs	68%(2.8)	68%	68%	0.22
LDA	69%(2.6)	65%	73%	0.19
NB	68%(2.8)	68%	68%	0.07
Logistic	69%(2.6)	66%	72%	1.20

The distribution density of the overall correctly classified observations and that of the percentage of correctly classified observations for defaulted obligors are plotted in Figure 18 and Figure 19 respectively. The figures are split into three panels. The upper panel shows the density for the linear models: LDA, naive Bayes and LR. The middle panel shows the density for the non-linear models: decision trees and ANNs and the bottom panel shows the density for the KNNs and the BKNNs.

Figure 18 reconfirms the findings from Table 22 where we observed that BKNNs and boosted DTs outperform the other algorithms in terms of overall classification accuracy. The BKNNs lead to the highest performance in terms of overall classification accuracy. Likewise, Figure 19 reconfirms the findings from Table 22 where we observe that LDA, naive Bayes and ANNs also provide high accuracy in identifying defaulted obligors.

Nonetheless, we stress that BKNN with GA estimation, leads to the same level of accuracy as BKNN with MCMC estimation. Moreover, due to the potential issues with the autocorrelation in MCMC, see Figure 30, Appendix C. We observe that only in case of MCMC with derivative adjustment there is a decrease in the autocorrelation of the chain. Although BKNN with MCMC estimation has the same level of accuracy as BKNN with GA estimation, the letter is a preferable choice due to its stability, see Figure 33 and Table

38.

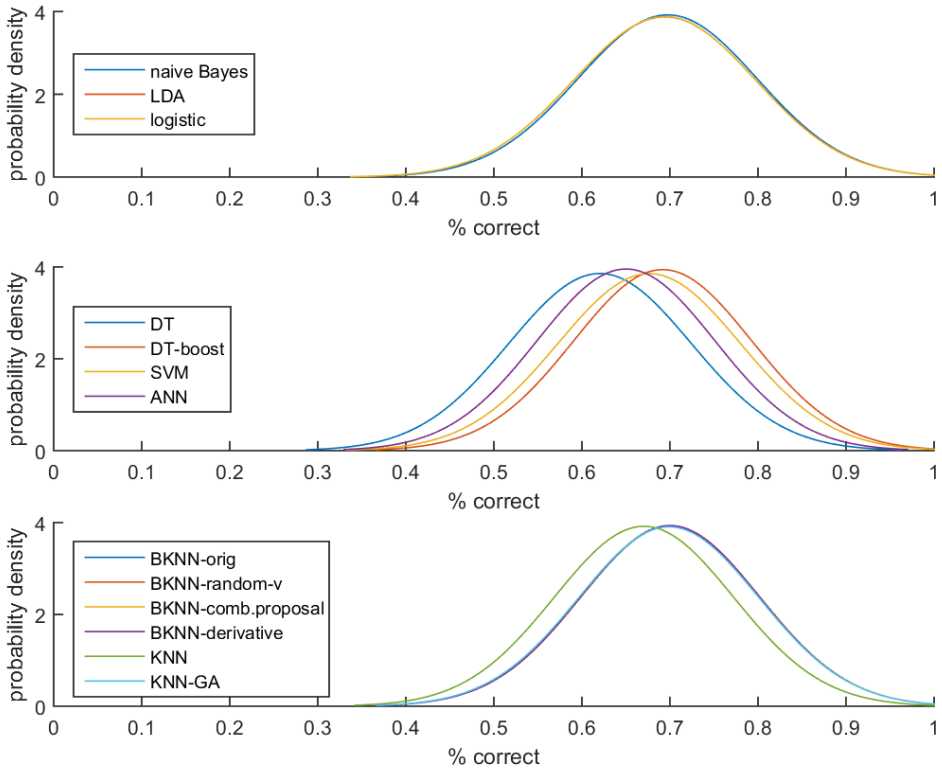


Figure 18: Density plots of the overall classification accuracy per classification algorithm; upper panel: density for the linear models: LDA, naive Bayes and LR; middle panel: density for the non-linear models: decision trees and ANNs; bottom panel: density for the KNNs and the BKNNs; East-European data.

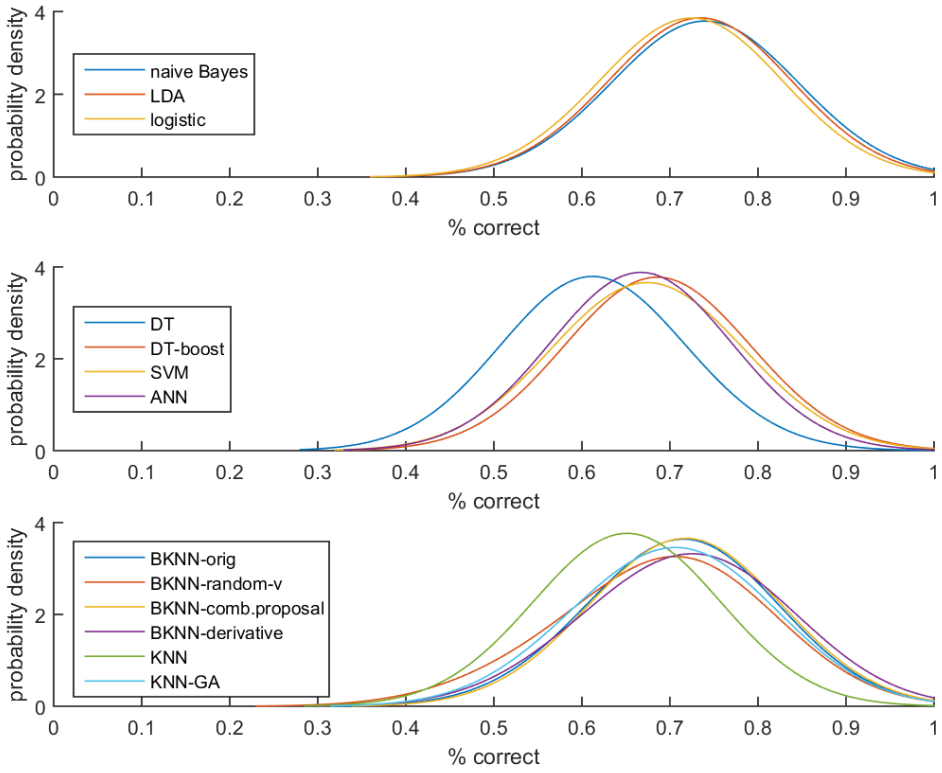


Figure 19: Density plots of the default obligors classification accuracy per algorithm; upper panel: density for the linear models: LDA, naive Bayes and LR; middle panel: density for the non-linear models: decision trees and ANNs; bottom panel: density for the KNNs and the BKNNs; East-European data.

Table 35 in Appendix D provides the results from testing whether there is a statistical difference in the mean of the percentage of overall correctly classified observations based on the bootstrapping results. When compared to classical KNN, all BKNN have statistically higher results. In Table 35, the number 1 indicates that there is a statistically significant difference in the mean, 0 indicates the opposite. Results are based on a two sample t-test (5% conf.interval), where the samples are derived using bootstrapping. We observe that the performance of the BKNNs methods is statistically different from that of classical KNNs.

4.3.2.2 Results on Polish corporate data Similarly to the East-European data study we apply the classification methods to the Polish data. The information contained in Table 23 is exactly the same as in Table 22 from the previous section.

Based on Table 23 the following conclusions can be drawn:

1. Estimation of BKNN with a GA algorithm leads to the same results as MCMC estimation.
2. The classical KNNs method is under-performing when compared to the BKNNs.
3. The BKNN method in all their versions outperform all other benchmark methods aside from boosted DTs.

Table 23: Performance of the classification methods on the Polish test data. % correct: the percentage of overall correctly classified obligors; % good: the percentage of correctly classified good obligors; % bad: the percentage of correctly classified bad obligors; CPU time/s: the CPU time in seconds needed for one run of the method.

Method	% correct (std. in %)	% good	% bad	CPU
BKNNs, GA	75%(2.6)	84%	66%	139.53
BKNNs	75%(2.6)	85%	66%	41.90
BKNNs, randomized variance	75%(2.6)	84%	66%	327.5
BKNNs, combined proposal	75%(2.4)	84%	66%	56.82
BKNNs, derivative	75%(2.6)	85%	66%	46.43
KNNs	72%(2.6)	78%	66%	0.18
DTs	74%(0.5)	77%	68%	0.12
DTs, boost	75%(2.4)	85%	63%	5.54
ANNs	57%(0.9)	56%	58%	1.09
SVMs	74%(2.8)	88%	59%	0.19
LDA	68%(5.4)	92%	28%	0.22
NB	57%(0.6)	96%	53%	0.16
Logistic	72%(8.6)	88%	21%	0.85

The distribution density of the overall correctly classified observations and that of the percentage of correctly classified observations for defaulted obligors are plotted in Figure 20 and Figure 21 respectively. These two figures present exactly the same information as Figure 18 and Figure 19 but for the Polish corporate data.

Figure 20 reconfirms the findings from Table 23 where we observe that BKNNs and DTs outperform the other algorithms in terms of overall classification accuracy. Figure 21 reconfirms the findings from Table 23 where we observed that LR, LDA and naive Bayes provide a significantly lower accuracy in identifying defaulted obligors than the other algorithms.

Nonetheless, we stress that BKNN with GA estimation, leads to the same level of accuracy as BKNN with MCMC estimation. Moreover, due to the potential issues with the

autocorrelation in MCMC, see Figure 31, Appendix C. Although BKNN with MCMC estimation has the same level of accuracy as BKNN with GA estimation, the letter is a preferable choice due to its stability, see Figure 33 and Table 38.

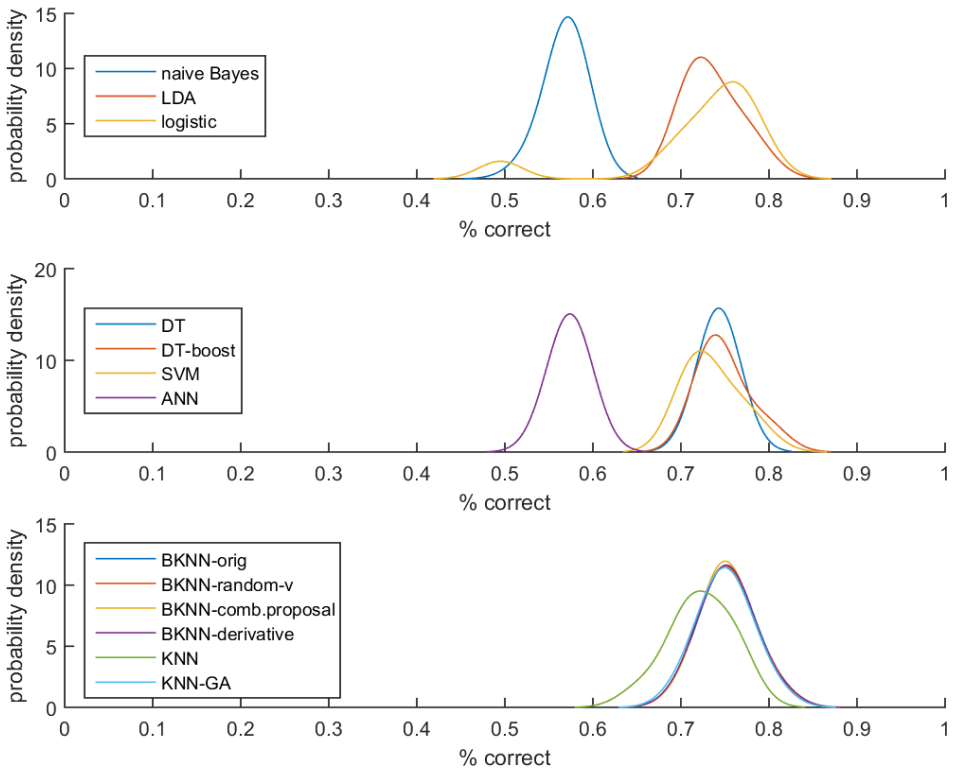


Figure 20: Density plots of the overall classification accuracy per classification algorithm; upper panel: density for the linear models: LDA, naive Bayes and LR; middle panel: density for the non-linear models: decision trees and ANNs; bottom panel: density for the KNNs and the BKNNs; Polish data.

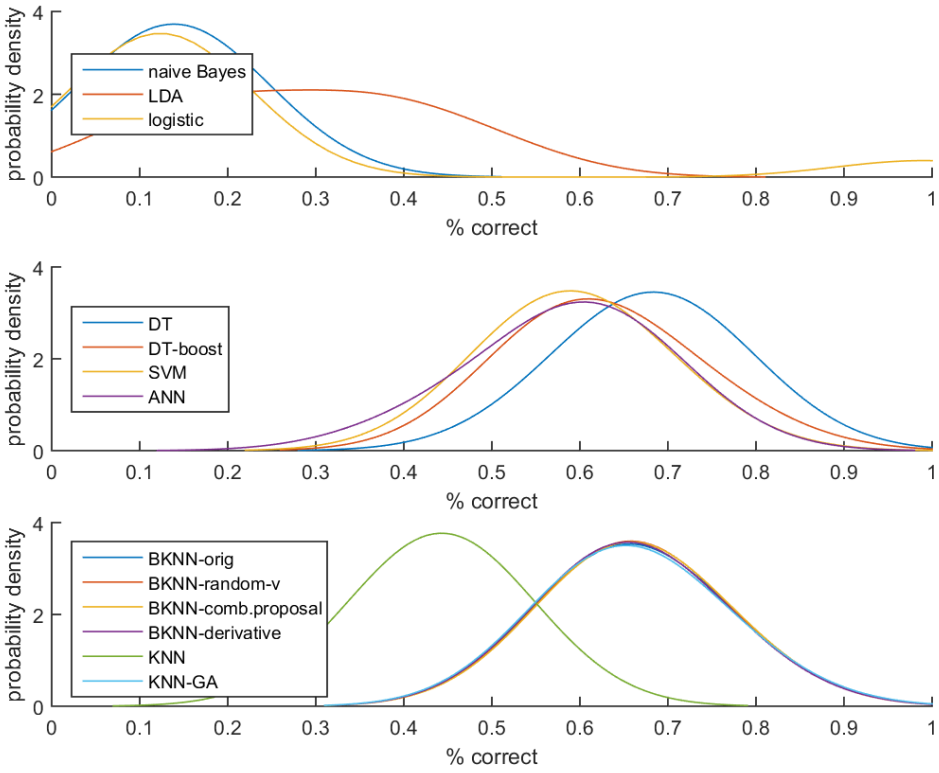


Figure 21: Density plots of the overall classification accuracy per classification algorithm;upper panel: density for the linear models: LDA, naive Bayes and LR; middle panel: density for the non-linear models: decision trees and ANNs; bottom panel: density for the KNNs and the BKNNs; Polish data.

Table 36 in Appendix D provides the results from testing weather there is a statistical difference in the mean of the percentage of overall correctly classified observations based on the bootstrapping results. When compared to classical KNN, all BKNN have statistically higher results. In Table 36 one indicates that there is a statistically significant difference in the mean, zero indicates the opposite. Results are based on a two sample t-test (5% conf.interval), where the samples are derived using bootstrapping.

4.3.2.3 Results on German corporate data Similarly to the East-European data study we apply the classification methods to the German retail data. The information contained in Table 24 is exactly the same as in Table 22 from the previous section.

Based on Table 24 several conclusions can be drawn:

1. The classical KNNs method is under-performing when compared to the Bayesian approach in terms of correctly classified observations.

2. The BKNNs shows the highest performance in terms of correctly classified observations.

3. LR, LDA and naive Bayes provide similar performance to DTs and BKNNs.

Table 24: Performance of the classification methods on the German test data. % correct: the percentage of overall correctly classified obligors; % good: the percentage of correctly classified good obligors; % bad: the percentage of correctly classified bad obligors; CPU time/s: the CPU time in seconds needed for one run of the method.

Method	% correct (std. in %)	% good	% bad	CPU
BKNNs, GA	65%(3.1)	66%	64%	65.99
BKNNs	66%(2.9)	67%	65%	33.45
BKNNs, randomized variance	66%(2.7)	71%	60%	250.6
BKNNs, combined proposal	66%(2.6)	70%	61%	46.81
BKNNs, derivative	67%(2.7)	66%	68%	32.18
KNNs	63%(2.4)	77%	50%	0.15
DTs	58%(2.2)	61%	52%	0.08
DTs, boost	65%(1.9)	66%	60%	4.17
ANNs	65%(1.1)	64%	76%	0.82
SVMs	66%(3.3)	67%	65%	0.12
LDA	65%(3.0)	66%	56%	0.17
NB	65%(5.7)	74%	49%	0.11
Logistic	66%(4.5)	67%	65%	0.64

The distribution density of the overall correctly classified observations and that of the percentage of correctly classified observations for defaulted obligors are plotted in Figure 22 and Figure 23 respectively. These two figures present exactly the same information as Figure 18 and Figure 19 but for the German data.

Figure 22 reconfirms the findings from Table 24 where we observe that BKNNs outperform the other algorithms in terms of overall classification accuracy. Among the different versions of BKNNs the one with a derivative adjustment shows the highest performance in terms of overall classification accuracy. Figure 23 confirms the findings that LDA and naive Bayes provide a relatively good accuracy in identifying defaulted obligors.

Nonetheless, we stress that BKNN with GA estimation, leads to slightly lower level of accuracy as BKNN with MCMC estimation. Due to the potential issues with the autocorrelation in MCMC, see Figure 32, Appendix C. Although BKNN with MCMC estimation has slightly higher level of accuracy when compared to BKNN with GA estimation, the

letter is a preferable choice due to its stability, see Figure 33 and Table 38.

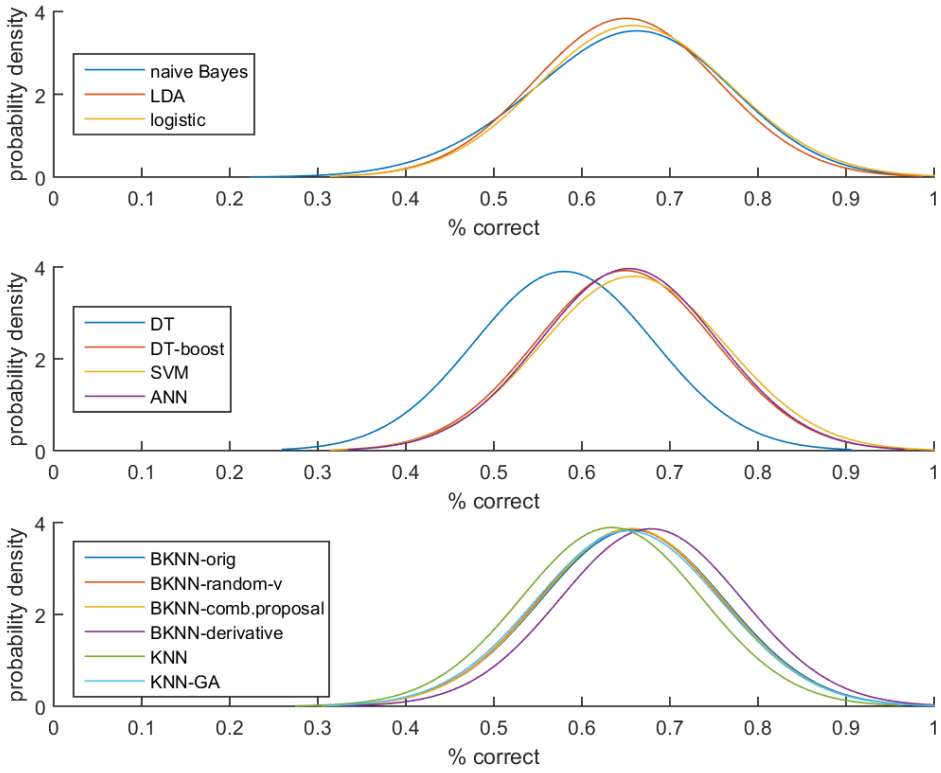


Figure 22: Density plots of the overall classification accuracy per classification algorithm; upper panel: density for the linear models: LDA, naive Bayes and LR; middle panel: density for the non-linear models: decision trees and ANNs; bottom panel: density for the KNNs and the BKNNs; German data.

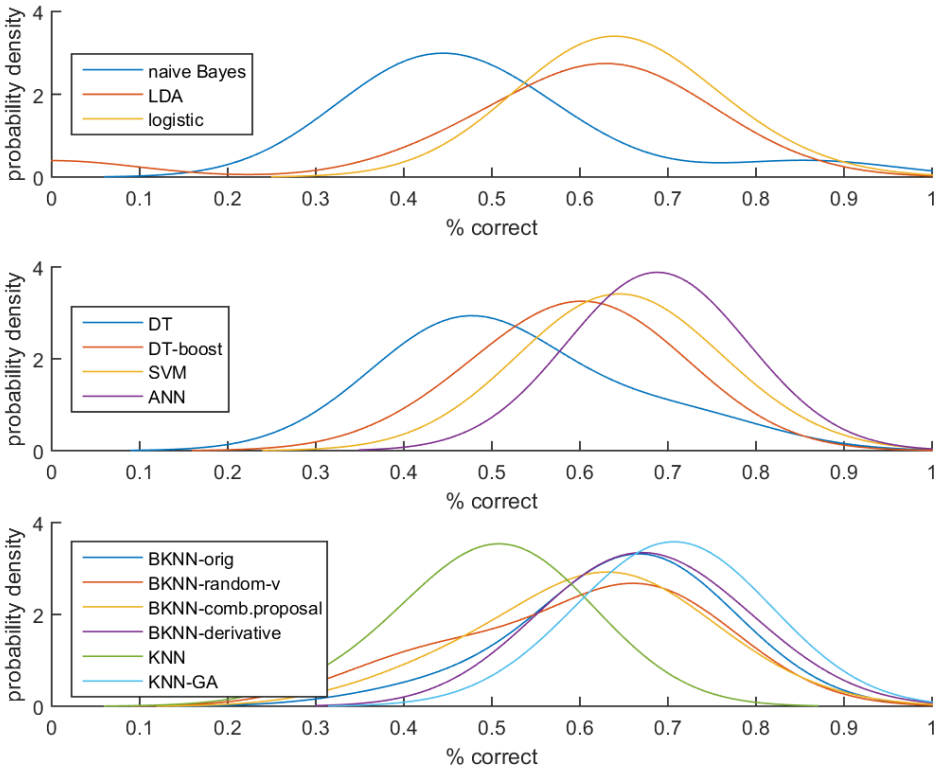


Figure 23: Density plots of the overall classification accuracy per classification algorithm;upper panel: density for the linear models: LDA, naive Bayes and LR; middle panel: density for the non-linear models: decision trees and ANNs; bottom panel: density for the KNNs and the BKNNs; German data.

Table 37 in Appendix D provides the results from testing weather there is a statistical difference in the mean of the percentage of overall correctly classified observations based on the bootstrapping results. When compared to classical KNN, most BKNN have statistically higher results. In Table 37 one indicates that there is a statistically significant difference in the mean, zero indicates the opposite. Results are based on a two sample t-test (5% conf.interval), where the samples are derived using bootstrapping.

4.3.3 Business intuition of the default drivers

Identifying a classification method to estimated the PD is an important factor but equally important is deriving business intuition from the final default factors. Typically PD models are used by non-technical audience and the interpretation of default factors from an industry prospective is of utmost importance. We analyse two data sets of corporate obligors and identify a group of variables for each data set that drive the default risk. The analysis performed on the Polish data set shows that six ratios significantly contribute to

the PD, see Table 25. The analysis performed on the East-European data of corporate clients shows elven drivers of defaults, see Table 25. Instead of focusing our analysis on a single ratio structure, we highlight the most frequent components of the selected default drivers. This way we generalize the default indicators rather than base our conclusions on certain type of financial ratios. Accordingly in the analysis below we discuss the main components of the selected default drivers.

We start with the similarities among default factors in the two datasets. We observe that total assets are a common default driver component for both data sets. This is consistent with the findings of Tian et al. (2015). The business intuition is that the amount of total assets relative to liquid assets or other balance sheet items such as net profit provide a clear picture of how efficient the utilization of these assets by a particular obligor is. Minimizing the amount of total assets and maximizing the net profit is an objective of every private company. Another common default driver component is the short-term (current) liabilities. This is consistent with the findings of Dragos et al. (2008). The business intuition is that current liabilities are a significant indicator of sort-term debt. Companies with high levels of current liabilities in relation to other balance sheet items such as cash and sales are riskier and therefore they have a higher default probability. Nevertheless, a default driver component present only in the East-European data is the cash variable. Clearly the amount of cash is more indicative measure of default for the smaller companies in East-Europe. The default rate in the Polish economy due to its scale and level of development is driven by more complex financial ratio structures than that of the East-European economies. Although some differences in default factor components exist between the Polish obligors and the East-European obligors, most of the default driver components are the same, namely total assets (the size of the company) and current liabilities. This is consistent with the findings of Hosaka and Takata (2016).

The analysis performed on the German retail data set shows that nine ratios significantly contribute to the PD; see Table 25. In this data set, most of the variables are based on the status and duration of the current account and the obligor's credit history. This is aligned with the study of Barrell et al. (2010), which shows evidence that the status of the current account is a major predictor of mortgage defaults.

Table 25: Final default drivers (based on the variable selection method) for the Polish data, East-European data and German data.

Polish data	East-European data	German data
net profit / total assets	interest coverage	status of existing checking account
net profit / inventory	cash ratio = (total cash and cash equivalents) / current liabilities	duration in months of the account
gross profit / short-term (current) liabilities	total liabilities / total assets	credit history
gross profit / sales	equity / total liabilities	other debtors/guarantors
(current liabilities * 365) / cost of products sold	return on operating income	telephone availability
inventory$\times$365 / cost of products sold	inventory turnover = (average inventory * 360) / cost of goods sold	credit amount
	payables turnover = supply payables $\times$ 360 / cost of goods sold	present employment since
	supply payables / total assets	present residence since
	relative annual change in total sales	property indicator
	relative annual change in total assets	
	income from sales / total assets	

4.3.4 Conclusion

In this experiment, we propose three innovative MCMC schemes to estimate the model parameters of a BKNNs as well as a GA algorithm that can be applied to estimate BKNN. We demonstrate that the Bayesian approach provides an automatic determination of the number of neighbours and outperforms the classical KNNs as well as many other classification algorithms. In particular, the BKNNs estimation based on GA algorithm leads to high performance and it has low dependence on potential autocorrelation issues typical for MCMC estimations. On the two corporates data sets: Polish and Eastern European,

the BKNN GA estimation has same level of accuracy as BKNN MCMC but its more stable. On the German retail data, BKNN GA estimation has slightly lower accuracy than BKNN MCMC but still sufficiently high. Results reported in this experiment provide evidence that BKNN GA estimation enhances corporate PD estimation. From a policy prospective the total assets and current liabilities are identified as main drivers of default for both Polish and East European corporate obligors. For the German retail data the main default drivers refer to the status of the current account and the obligor's credit history. Based on the above findings, we conclude that in principle the application of non-linear models to corporate PD estimation should be widely used in practice.

5 Conclusions

In this thesis we aim to provide evidence that Bayesian non-linear supervised machine learning algorithms stand as a superior alternative to their linear and non-linear counterparts. Extensive evidence has been provided that Bayesian non-linear supervised machine learning methods result in higher accuracy when compared to the linear and non-linear alternatives. In particular, we proposed a new estimation approach on the Bayesian regularization of ANNs. It has been shown that the new estimation provides higher classification accuracy for most data sets in this thesis. Moreover, we proposed a new estimation updates on the learning of BKNNs and managed to show the BKNNs estimated in this new way result in higher classification accuracy. Last but not least, we proposed an innovative variable selection method for SVMs that is based on the specific structure of SVMs. We showed the variable selection method outperforms other variable selection methods for most data sets used.

With respect to the above contributions of our research, we also highlight the limitations of our study. First, the proposed estimation updates on ANNs and KNNs are applied to classification problems only. It would be interesting to see their relevance in other contexts such as clustering and regression problems. Second, the superior performance of the proposed estimation and variable selection is benchmarked to several algorithms but this is not an exhaustive list of classification benchmarks. There are some classification methods that have not been covered but are also used in practice. Third, the proposed estimation principles can be applied to some of the linear classification methods and perhaps offer another superior alternative to the methods currently used in the industry.

Overall, in our research we aimed to open a new horizon for exploring classification methods and applying them to real data. Although this work could be expanded further, we believe in its current for, it presents enough evidence in support of Bayesian non-linear classification methods and we advocate their use is increased both in the academia and in the industry.

Appendices

Appendix A

Table 26: Summary statistics for all ratios, German retail data. The median and the mean are shown before standardization of the variable.

Ratio number and name	Median	Mean
1 status of existing checking account	2	2.58
2 duration in months of the account	18	20.9
3 credit history	3	3.6
4 credit purpose	2	2.9
5 credit amount	2320	3271
6 savings account/bonds	1	2.1
7 present employment since	3	3.9
8 installment rate in percentage of disposable income	3	2.973
9 personal status and sex	3	2.7
10 other debtors/guarantors	1	1.2
11 present residence since	3	2.845
12 property indicator	2	2.4
13 age in years	33	35.55
14 other installment plans	3	2.7
15 housing indicator	2	1.9
16 number of existing credits at this bank	1	1.41
17 job status	3	2.9
18 number of people being liable to provide maintenance for	1	1.2
19 telephone availability	1	1.4
20 foreign worker indicator	1	1

Table 27: Summary statistics for all ratios, East-European data. Mean, mean_i, median and median_i are the mean and median before and after imputation; % missing is the percentage of missing values.

Ratio name	Mean	Median	Mean_i	Median_i	% Missing
1 return on assets (ROA)	0.13	0.08	0.13	0.08	0.00

Continued on next page

Table 27 — continued from previous page

Ratio name	Mean	Median	Mean_i	Median_i	% Missing
2 ROA before financial expenses	0.18	0.13	0.18	0.12	0.00
3 return on operating income	-0.07	0.08	-0.07	0.08	0.28
4 return on sales income	-0.01	0.11	-0.01	0.11	0.44
5 return on investment	0.06	0.03	0.06	0.03	0.00
6 cash ratio	0.45	0.01	0.45	0.01	4.24
7 quick ratio	2.02	0.50	2.06	0.50	4.24
8 operating cash flow ratio	4.04	1.14	4.21	1.16	4.24
9 liquid assets/total assets	0.04	0.00	0.03	0.00	0.00
10 working capital/total assets	0.49	0.48	0.49	0.48	0.00
11 financial autonomy	6.67	0.64	14.25	20.34	0.00
12 total funding ratio	0.83	0.76	0.83	0.00	0.00
13 long term funding ratio	0.39	0.20	0.39	0.00	0.00
14 total liabilities/total assets	0.39	0.23	0.39	0.22	0.00
15 supply payables/total assets	0.16	0.09	0.16	0.09	0.00
16 financial liabilities/total liabilities	0.39	0.35	0.39	0.35	0.00
17 equity/total liabilities	2.01	0.29	2.04	0.29	1.88
18 short term funding ratio	0.62	0.68	0.62	0.68	1.88
19 total liabilities coverage	1.35	0.17	1.37	0.17	1.88
20 financial liabilities coverage	12.15	0.40	11.45	0.41	20.40
21 current financial liabilities coverage	7.30	0.87	NA	NA	84.54
22 interest coverage	47.44	4.17	99.86	4.43	16.17
23 income from sales/total assets	1.74	1.00	1.73	1.00	0.00
24 employees' expense/sales income	0.13	0.06	0.13	0.06	0.44
25 earnings on operating income	1.10	0.95	1.10	0.94	0.44
26 payables turnover	243.32	39.27	263.44	39.30	0.89
27 inventory turnover	248.54	66.59	251.29	66.41	0.89
28 receivables turnover	96.08	20.27	97.17	20.27	0.44
29 total sales income	5349	524	5348.68	524.00	0.00
30 total assets	3365	531	3365.22	531.00	0.00
31 relative annual change in total sales	2.37	0.12	4.62	0.14	33.00

Continued on next page

Table 27 — continued from previous page

Ratio name	Mean	Median	Mean_i	Median_i	% Missing
32 relative annual change in total assets	1.13	0.16	1.34	0.15	32.94
33 relative annual change in profit from main activities	4.33	-0.09	11.18	-0.08	34.28
34 absolute annual change in total liabilities	0.03	0.00	0.03	0.00	32.94

Table 28: Summary statistics for all ratios, Polish data. Mean, mean_i, median and median_i are the mean and median before and after imputation; % missing is the percentage of missing values.

Ratio name	Mean	Median	Mean_i	Median_i	% Missing
1 net profit/total assets	-0.02	0.05	-0.02	0.05	0.00
2 total liabilities/total assets	0.47	0.45	0.47	0.45	0.00
3 working capital/total assets	0.19	0.22	0.19	0.22	0.00
4 current assets/short-term liabilities	4.89	1.65	4.89	1.66	0.00
5 (cash + short - term securities + receivables - short-term liabilities)/(operating expenses-depreciation)$\times$365	19.41	0.49	19.41	0.57	0.00
6 retained earnings/total assets	0.02	0.00	0.02	0.00	0.00
7 EBIT/total assets	-0.11	0.06	-0.11	0.06	0.00
8 book value of equity/total liabilities	5.74	1.15	5.74	1.16	0.00
9 sales/total assets	1.59	1.14	1.59	1.14	0.00
10 equity/total assets	0.55	0.52	0.55	0.52	0.00
11 (gross profit + extraordinary items + financial expenses)/total assets	-0.01	0.07	-0.01	0.07	0.00
12 gross profit/short-term liabilities	1.07	0.17	1.07	0.17	0.00
13 (gross profit + depreciation)/sales	0.35	0.07	0.35	0.07	0.00

Continued on next page

Table 28 — continued from previous page

Ratio name	Mean	Median	Mean_i	Median_i	% Missing
14 gross profit + interest)/total assets	-0.11	0.06	-0.11	0.06	0.00
15 (total liabilities×365)/(gross profit + depreciation)	1033.62	872.16	1033.62	875.25	0.00
16 (gross profit + depreciation)/total liabilities	1.19	0.24	1.19	0.24	0.00
17 total assets/total liabilities	6.83	2.21	6.83	2.21	0.00
18 gross profit/total assets	-0.10	0.06	-0.10	0.06	0.00
19 gross profit/sales	-0.09	0.04	-0.09	0.04	0.00
20 (inventory×365)/sales	56.67	38.62	56.67	38.62	0.00
21 sales(n)/sales(n-1)	2.46	1.12	2.46	1.12	2.00
22 profit on operating activities/total assets	-0.00	0.06	-0.00	0.06	0.00
23 net profit/sales	-0.10	0.03	-0.10	0.03	0.00
24 gross profit(in 3 years)/total assets	0.14	0.16	0.14	0.16	2.00
25 (equity - share capital)/total assets	0.38	0.42	0.38	0.42	0.00
26 (net profit + depreciation)/total liabilities	1.09	0.21	1.09	0.21	0.00
27 profit on operating activities/financial expenses	463.64	0.98	463.64	1.15	7.00
28 working capital/fixed assets	10.23	0.53	10.23	0.55	2.00
29 logarithm of total assets	4.15	4.17	4.15	4.17	0.00
30 (total liabilities,cash)/sales	0.85	0.22	0.85	0.22	0.00
31 (gross profit + interest)/sales	-0.07	0.04	-0.07	0.04	0.00
32 (current liabilities * 365)/cost of products sold	2111.59	81.13	2111.59	81.91	1.00
33 operating expenses/short-term liabilities	8.34	4.47	8.34	4.50	0.00
34 operating expenses/total liabilities	5.01	1.71	5.01	1.72	0.00
35 profit on sales/total assets	-0.01	0.06	-0.01	0.06	0.00
36 total sales/total assets	2.05	1.56	2.05	1.56	0.00

Continued on next page

Table 28 — continued from previous page

Ratio name	Mean	Median	Mean_i	Median_i	% Missing
37 (current assets,inventories)/long-term liabilities	114.03	3.66	67.02	5.00	43.00
38 constant capital/total assets	0.65	0.62	0.65	0.62	0.00
39 profit on sales/sales	0.02	0.04	0.02	0.04	0.00
40 (current assets,inventory/receivables)/short-term liabilities	2.21	0.18	2.21	0.18	0.00
41 total liabilities/((profit on operating activities + depreciation)×(12/365))	2.19	0.09	2.19	9.00	1.00
42 profit on operating activities/sales	-0.02	0.04	-0.02	0.04	0.00
43 rotation receivables + inventory turnover in days	155.56	106.41	155.56	106.41	0.00
44 (receivables×365)/sales	98.88	58.79	98.88	58.79	0.00
45 net profit/inventory	66.63	0.26	66.63	0.29	5.00
46 (current assets-inventory)/short-term liabilities	4.01	1.07	4.01	1.07	0.00
47 (inventory×365)/cost of products sold	137.42	41.99	137.42	42.35	1.00
48 EBITDA/total assets	-0.09	0.02	-0.09	0.02	0.00
49 EBITDA/sales	-0.07	0.01	-0.07	0.01	0.00
50 current assets/total liabilities	4.17	1.29	4.17	1.29	0.00
51 short-term liabilities/total assets	0.43	0.33	0.43	0.33	0.00
52 (short-term liabilities×365)/cost of products sold)	0.73	0.22	0.73	0.22	1.00
53 equity/fixed assets	11.20	1.28	11.20	1.30	2.00
54 constant capital/fixed assets	12.11	1.43	12.11	1.45	2.00
55 working capital	10817	1803	10817	1803	0.00
56 (sales, cost of products sold)/sales	0.06	0.05	0.06	0.05	0.00

Continued on next page

Table 28 — continued from previous page

Ratio name	Mean	Median	Mean_i	Median_i	% Missing
57 (current assets-inventory-short-term liabilities)/(sales-gross profit-depreciation)	-0.26	0.11	-0.26	0.11	0.00
58 total costs/total sales	0.96	0.95	0.96	0.95	0.00
59 long-term liabilities/equity	0.28	0.01	0.28	0.01	0.00
60 sales/inventory	911.03	9.04	911.03	9.45	5.00
61 sales/receivables	10.94	6.20	10.94	6.21	0.00
62 (short-term liabilities×365)/sales	241.98	73.78	241.98	73.78	0.00
63 sales/short-term liabilities	9.13	4.93	9.13	4.94	0.00
64 sales/fixed assets	65.28	4.10	65.28	4.22	2.00

Appendix B

Table 29: Statistical significance of Bayesian MCMC (Architectures 7 and 8) on the overall accuracy per data set: East-European(E), Polish (P), German (G). 1 indicates statistical difference, 0 indicates no statistical difference. MCMC1 and MCMC2 stand for Architectures 7 and 8 in Table 19

	E MCMC1	E MCMC2	P MCMC1	P MCMC2	G MCMC1	G MCMC2
Architecture	7	8	7	8	7	8
1	1	1	1	1	0	1
2	1	1	1	1	1	1
3	1	1	1	1	0	1
4	1	1	1	1	1	1
5	1	1	1	1	1	1
6	1	1	1	1	1	1
7	0	1	0	0	0	1
8	1	0	0	0	1	0

Table 30: Performance of the ANNs on the East-European, Polish and German test data when using factors based on the 90% percentile of the correlation to the target variable. Correct: the percentage of overall correctly classified obligors; Good: the percentage of correctly classified good obligors; Bad: the percentage of correctly classified bad obligors; Gini: the Gini coefficient.

Architecture	Regularization	ES	Correct	Good	Bad	Gini
East-European data						
1	No	No	0.67	0.56	0.77	0.58
2	No	Yes	0.67	0.57	0.77	0.60
3	Classical	No	0.67	0.56	0.77	0.58
4	Classical	Yes	0.67	0.57	0.77	0.60
5	Bayesian	No	0.66	0.54	0.77	0.59
6	Bayesian	Yes	0.66	0.51	0.80	0.61
7	Bayesian MCMC	No	0.67	0.58	0.77	0.62
8	Bayesian MCMC	Yes	0.68	0.66	0.70	0.58
Polish data						
1	No	No	0.67	0.73	0.60	0.52
2	No	Yes	0.67	0.66	0.67	0.59
3	Classical	No	0.67	0.70	0.64	0.52

Table 30 – continued from previous page

4	Classical	Yes	0.67	0.68	0.66	0.59
5	Bayesian	No	0.65	0.82	0.46	0.54
6	Bayesian	Yes	0.68	0.66	0.70	0.56
7	Bayesian MCMC	No	0.74	0.74	0.71	0.61
8	Bayesian MCMC	Yes	0.74	0.73	0.75	0.66
		German data				
1	No	No	0.68	0.63	0.72	0.60
2	No	Yes	0.67	0.63	0.71	0.61
3	Classical	No	0.68	0.61	0.74	0.61
4	Classical	Yes	0.67	0.61	0.74	0.61
5	Bayesian	No	0.66	0.67	0.65	0.57
6	Bayesian	Yes	0.61	0.43	0.75	0.55
7	Bayesian MCMC	No	0.68	0.65	0.70	0.57
8	Bayesian MCMC	Yes	0.68	0.64	0.72	0.58

Table 31: Performance of the ANNs on the East-European, Polish and German test data when using factors based on the 70% percentile of the correlation to the target variable. Correct: the percentage of overall correctly classified obligors; Good: the percentage of correctly classified good obligors; Bad: the percentage of correctly classified bad obligors; Gini: the Gini coefficient.

Architecture	Regularization	ES	Correct	Good	Bad	Gini
		East-European data				
1	No	No	0.66	0.63	0.69	0.58
2	No	Yes	0.66	0.59	0.73	0.60
3	Classical	No	0.66	0.62	0.70	0.58
4	Classical	Yes	0.66	0.60	0.73	0.60
5	Bayesian	No	0.67	0.55	0.80	0.55
6	Bayesian	Yes	0.67	0.63	0.71	0.54
7	Bayesian MCMC	No	0.70	0.71	0.68	0.58
8	Bayesian MCMC	Yes	0.69	0.72	0.66	0.56
		Polish data				
1	No	No	0.66	0.78	0.55	0.51
2	No	Yes	0.64	0.76	0.53	0.54
3	Classical	No	0.66	0.76	0.57	0.50
4	Classical	Yes	0.65	0.77	0.52	0.54
5	Bayesian	No	0.66	0.71	0.59	0.53

Table 31 – continued from previous page

6	Bayesian	Yes	0.65	0.56	0.66	0.49
7	Bayesian MCMC	No	0.69	0.73	0.62	0.51
8	Bayesian MCMC	Yes	0.67	0.67	0.66	0.54
		German data				
1	No	No	0.68	0.68	0.66	0.59
2	No	Yes	0.67	0.65	0.68	0.59
3	Classical	No	0.68	0.69	0.66	0.59
4	Classical	Yes	0.67	0.65	0.69	0.59
5	Bayesian	No	0.68	0.67	0.69	0.52
6	Bayesian	Yes	0.60	0.65	0.53	0.54
7	Bayesian MCMC	No	0.70	0.68	0.71	0.59
8	Bayesian MCMC	Yes	0.69	0.70	0.69	0.60

Table 32: Performance of the ANNs on the East-European, Polish and German test data when using factors based on the 60% percentile of the correlation to the target variable. Correct: the percentage of overall correctly classified obligors; Good: the percentage of correctly classified good obligors; Bad: the percentage of correctly classified bad obligors; Gini: the Gini coefficient.

Architecture	Regularization	ES	Correct	Good	Bad	Gini
		East-European data				
1	No	No	0.67	0.63	0.70	0.59
2	No	Yes	0.65	0.57	0.73	0.60
3	Classical	No	0.66	0.61	0.71	0.58
4	Classical	Yes	0.65	0.57	0.73	0.60
5	Bayesian	No	0.68	0.61	0.74	0.53
6	Bayesian	Yes	0.67	0.62	0.70	0.55
7	Bayesian MCMC	No	0.70	0.69	0.70	0.59
8	Bayesian MCMC	Yes	0.71	0.67	0.75	0.63
		Polish data				
1	No	No	0.66	0.74	0.57	0.51
2	No	Yes	0.62	0.79	0.45	0.53
3	Classical	No	0.66	0.76	0.56	0.51
4	Classical	Yes	0.63	0.82	0.42	0.54
5	Bayesian	No	0.63	0.64	0.59	0.53
6	Bayesian	Yes	0.63	0.59	0.64	0.48
7	Bayesian MCMC	No	0.67	0.72	0.64	0.53

Table 32 – continued from previous page

8	Bayesian MCMC	Yes	0.66	0.68	0.64	0.52
German data						
1	No	No	0.70	0.66	0.73	0.59
2	No	Yes	0.68	0.67	0.68	0.58
3	Classical	No	0.70	0.66	0.72	0.59
4	Classical	Yes	0.68	0.67	0.68	0.58
5	Bayesian	No	0.69	0.73	0.65	0.52
6	Bayesian	Yes	0.62	0.52	0.68	0.50
7	Bayesian MCMC	No	0.69	0.69	0.69	0.58
8	Bayesian MCMC	Yes	0.70	0.71	0.69	0.59

Table 33: Performance of the ANNs on the East-European, Polish and German test data when using factors based on the 50% percentile of the correlation to the target variable. Correct: the percentage of overall correctly classified obligors; Good: the percentage of correctly classified good obligors; Bad: the percentage of correctly classified bad obligors; Gini: the Gini coefficient.

Architecture	Regularization	ES	Correct	Good	Bad	Gini
East-European data						
1	No	No	0.66	0.62	0.69	0.59
2	No	Yes	0.66	0.61	0.71	0.59
3	Classical	No	0.66	0.62	0.69	0.59
4	Classical	Yes	0.66	0.61	0.71	0.59
5	Bayesian	No	0.66	0.58	0.74	0.54
6	Bayesian	Yes	0.67	0.74	0.60	0.56
7	Bayesian MCMC	No	0.69	0.72	0.66	0.58
8	Bayesian MCMC	Yes	0.70	0.73	0.67	0.58
Polish data						
1	No	No	0.68	0.78	0.57	0.51
2	No	Yes	0.64	0.73	0.54	0.54
3	Classical	No	0.67	0.77	0.57	0.51
4	Classical	Yes	0.64	0.75	0.53	0.54
5	Bayesian	No	0.64	0.45	0.81	0.53
6	Bayesian	Yes	0.65	0.57	0.73	0.49
7	Bayesian MCMC	No	0.69	0.74	0.56	0.52
8	Bayesian MCMC	Yes	0.67	0.67	0.68	0.53

Table 33 – continued from previous page

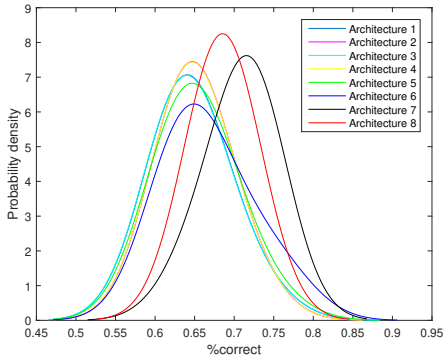
German data						
1	No	No	0.66	0.65	0.67	0.56
2	No	Yes	0.66	0.67	0.65	0.59
3	Classical	No	0.66	0.65	0.67	0.56
4	Classical	Yes	0.66	0.68	0.64	0.58
5	Bayesian	No	0.68	0.65	0.70	0.55
6	Bayesian	Yes	0.61	0.44	0.72	0.50
7	Bayesian MCMC	No	0.69	0.70	0.68	0.59
8	Bayesian MCMC	Yes	0.70	0.69	0.70	0.61

Table 34: Performance of the ANNs on the East-European, Polish and German test data. when using factors based on the 0% percentile of the correlation to the target variable. Correct: the percentage of overall correctly classified obligors; Good: the percentage of correctly classified good obligors; Bad: the percentage of correctly classified bad obligors; Gini: the Gini coefficient.

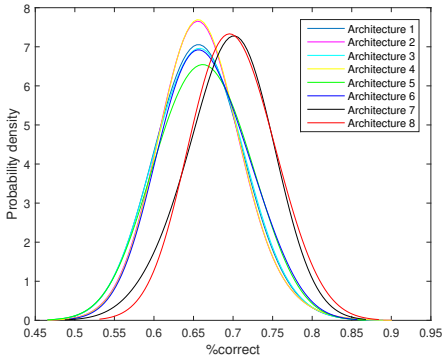
Architecture	Regularization	ES	Correct	Good	Bad	Gini
East-European data						
1	No	No	0.65	0.60	0.70	0.60
2	No	Yes	0.65	0.60	0.71	0.60
3	Classical	No	0.65	0.60	0.70	0.60
4	Classical	Yes	0.65	0.60	0.71	0.60
5	Bayesian	No	0.65	0.65	0.66	0.55
6	Bayesian	Yes	0.67	0.66	0.67	0.56
7	Bayesian MCMC	No	0.71	0.67	0.75	0.61
8	Bayesian MCMC	Yes	0.69	0.68	0.70	0.60
Polish data						
1	No	No	0.65	0.76	0.52	0.51
2	No	Yes	0.62	0.74	0.49	0.54
3	Classical	No	0.66	0.80	0.50	0.51
4	Classical	Yes	0.62	0.78	0.44	0.54
5	Bayesian	No	0.63	0.67	0.55	0.51
6	Bayesian	Yes	0.64	0.87	0.39	0.53
7	Bayesian MCMC	No	0.67	0.79	0.56	0.52
8	Bayesian MCMC	Yes	0.64	0.65	0.60	0.51
German data						
1	No	No	0.65	0.68	0.62	0.56

Table 34 – continued from previous page

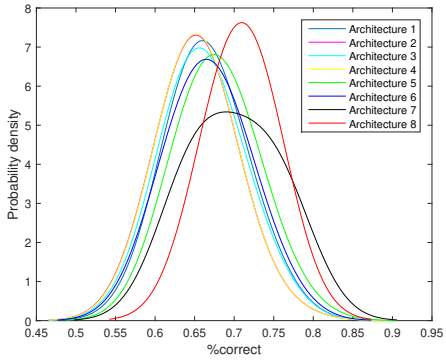
	2	No	Yes	0.66	0.68	0.64	0.57
	3	Classical	No	0.65	0.67	0.62	0.56
	4	Classical	Yes	0.66	0.68	0.64	0.57
	5	Bayesian	No	0.68	0.76	0.59	0.53
	6	Bayesian	Yes	0.67	0.67	0.62	0.54
	7	Bayesian MCMC	No	0.68	0.66	0.69	0.61
	8	Bayesian MCMC	Yes	0.67	0.62	0.72	0.58



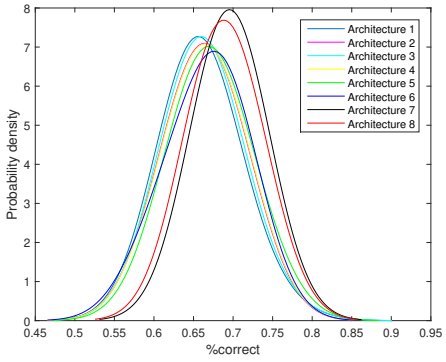
(a) 0% percentile



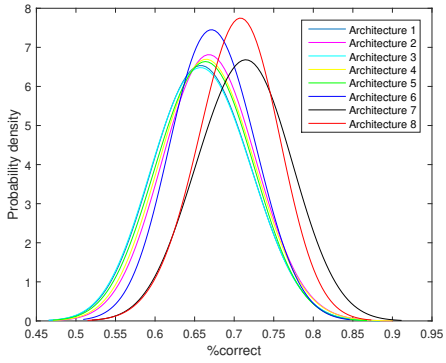
(b) 50% percentile



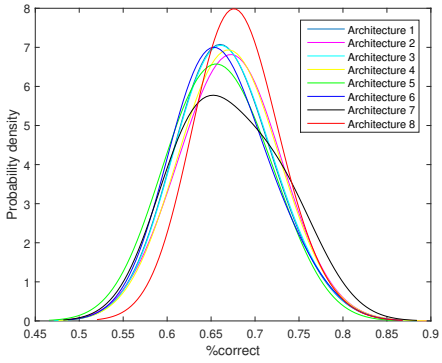
(c) 60% percentile



(d) 70% percentile

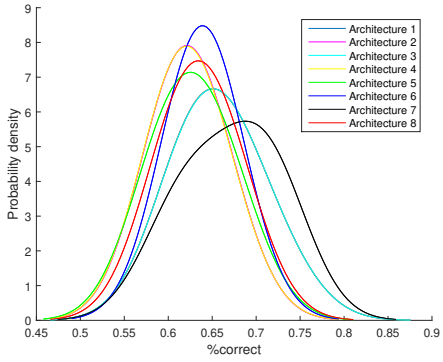


(e) 80% percentile

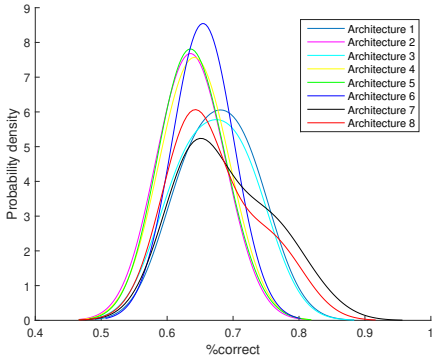


(f) 90% percentile

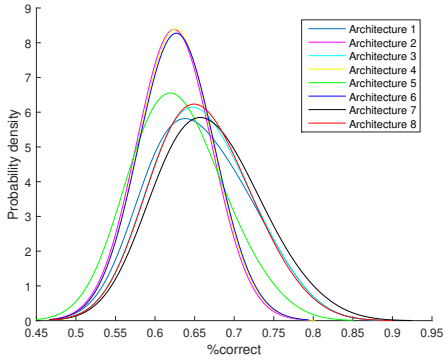
Figure 24: Distribution of the overall correctly classified obligors for the East-European data, based on the 0%, 50%, 60%, 70%, 80% and 90% percentile of the correlation to the target variable.



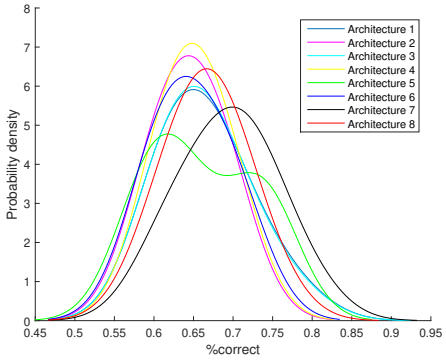
(a) 0% percentile



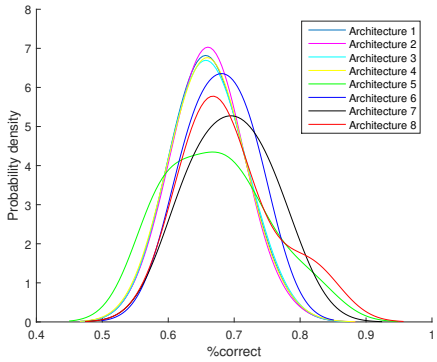
(b) 50% percentile



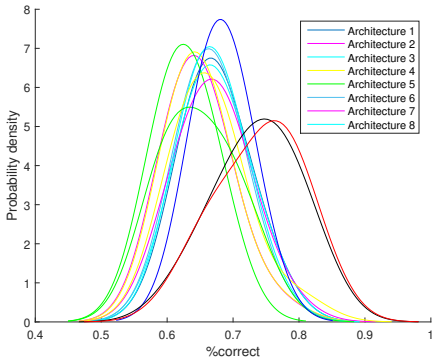
(c) 60% percentile



(d) 70% percentile

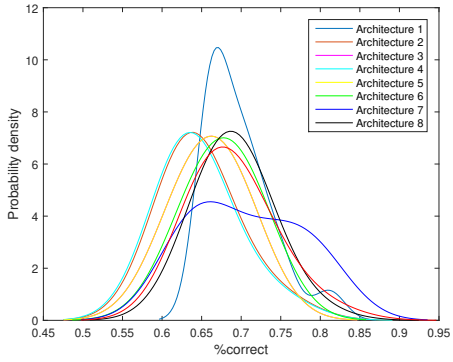


(e) 80% percentile

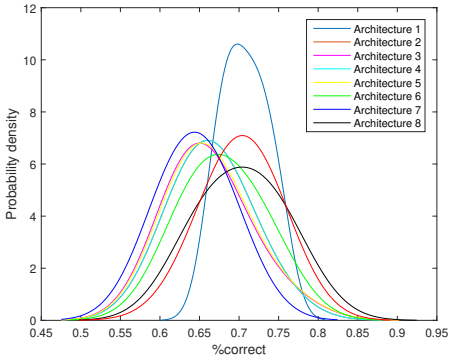


(f) 90% percentile

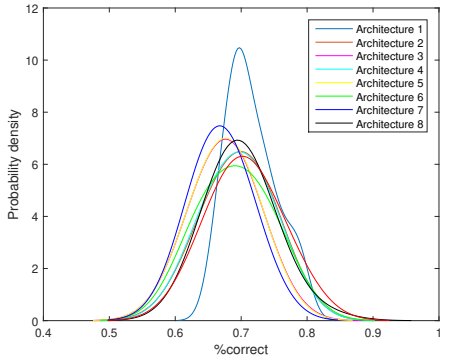
Figure 25: Distribution of the overall correctly classified obligors for the Polish data, based on the 0%, 50%, 60%, 70%, 80% and 90% percentile of the correlation to the target variable.



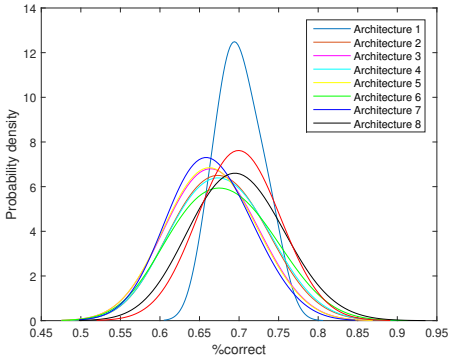
(a) 0% percentile



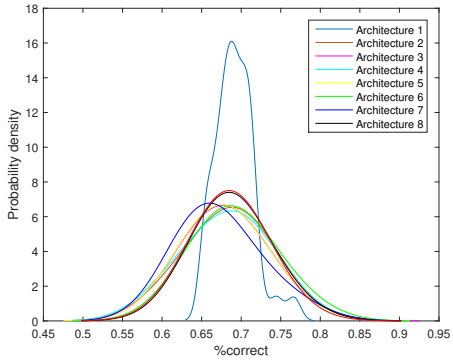
(b) 50% percentile



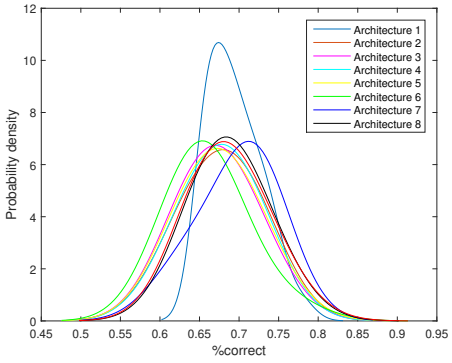
(c) 60% percentile



(d) 70% percentile



(e) 80% percentile



(f) 90% percentile

Figure 26: Distribution of the overall correctly classified obligors for the German data, based on the 0%, 50%, 60%, 70%, 80% and 90% percentile of the correlation to the target variable.

Appendix C

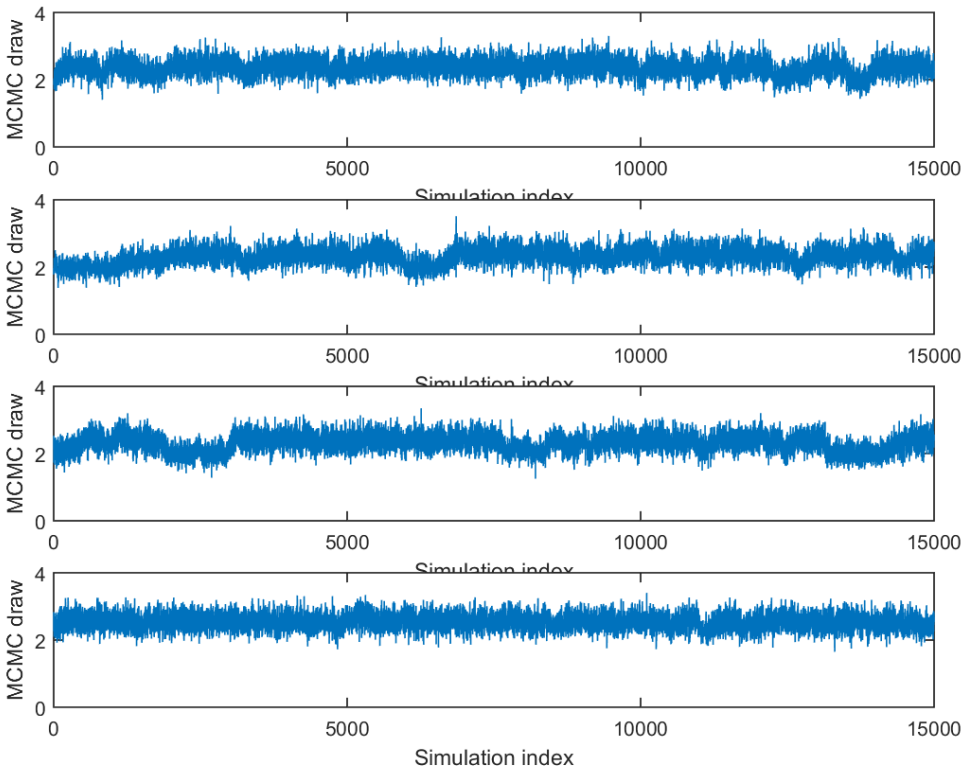


Figure 27: Trace plots on the BKNNs for the East-European data. Upper panel: original sampling; middle panels: randomized variance and combined proposal; bottom panel: derivative adjustment. All plots show that the estimation of the β parameter is stable.

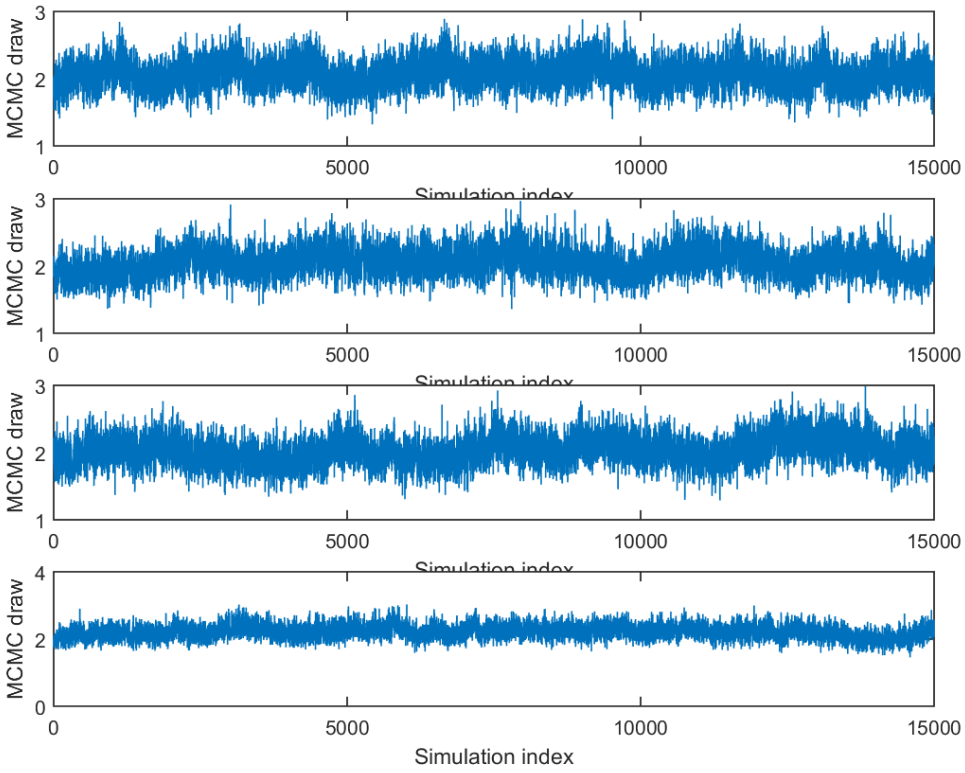


Figure 28: Trace plots on the BKNNs for the Polish data. Upper panel: original sampling; middle panels: randomized variance and combined proposal; bottom panel: derivative adjustment. All plots show that the estimation of the β parameter is stable.

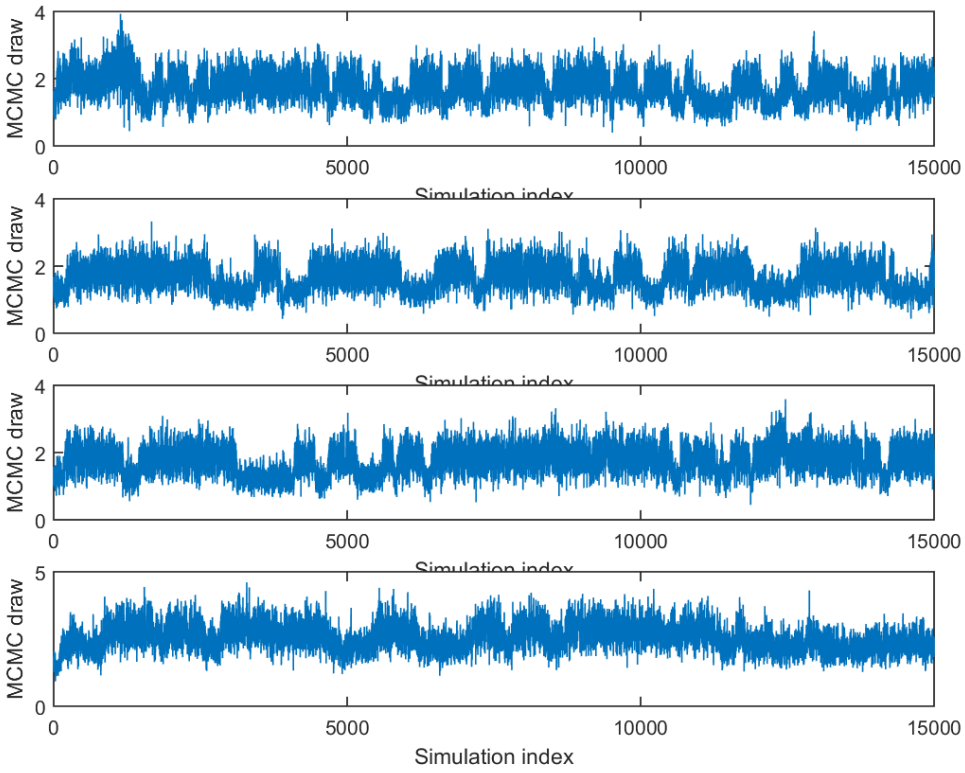


Figure 29: Trace plots on the BKNNs for the German data. Upper panel: original sampling; middle panels: randomized variance and combined proposal; bottom panel: derivative adjustment. All plots show that the estimation of the β parameter is stable.

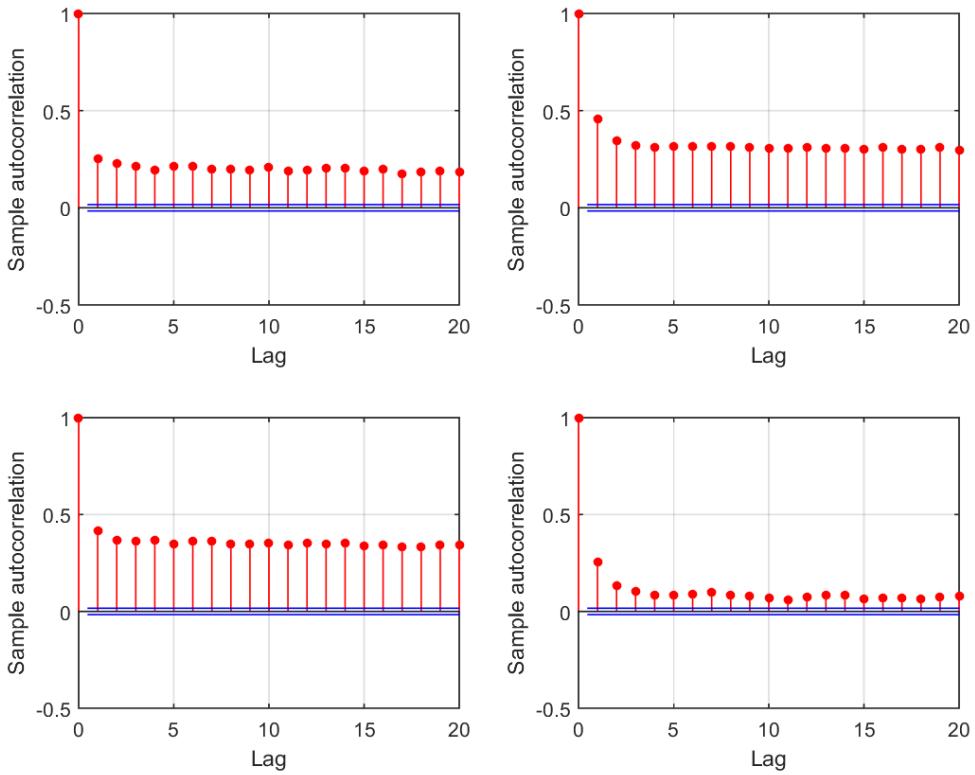


Figure 30: Sample autocorrelation functions on the BKNNs for the East-European data. Top left: original sampling; top right: randomized variance; bottom left: combined proposal; bottom right: derivative adjustment. All plots show that autocorrelation gradually decreases and therefore provide evidence of low autocorrelation in the chain.

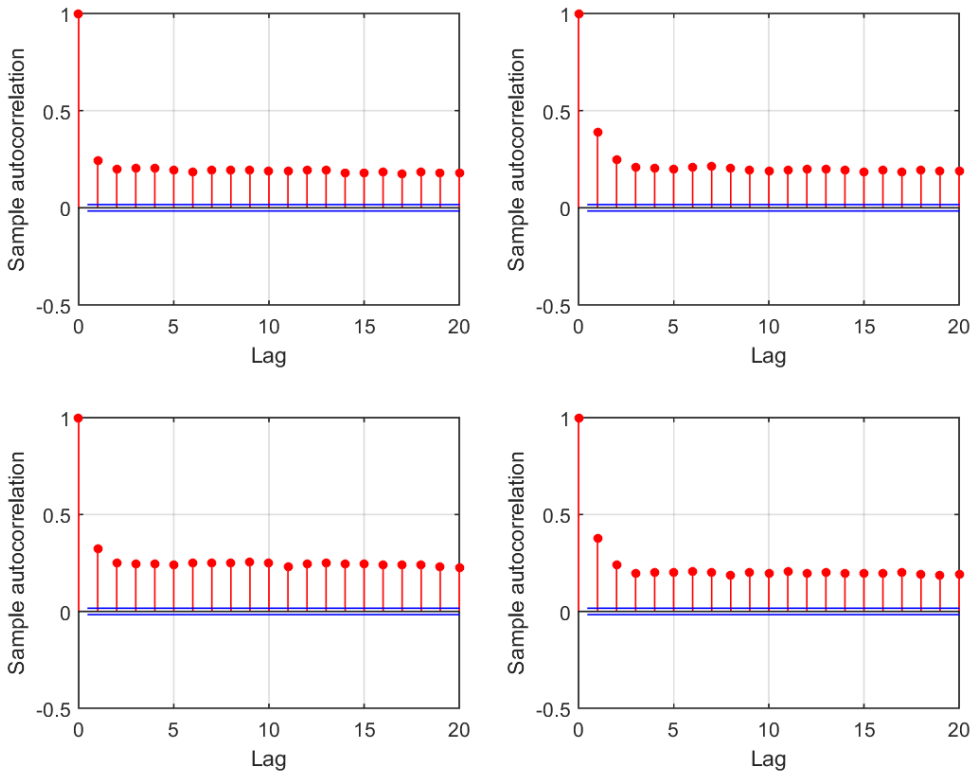


Figure 31: Sample autocorrelation functions on the BKNNs for the Polish data. Top left: original sampling; top right: randomized variance; bottom left: combined proposal; bottom right: derivative adjustment. All plots show that autocorrelation gradually decreases and therefore provide evidence of low autocorrelation in the chain.

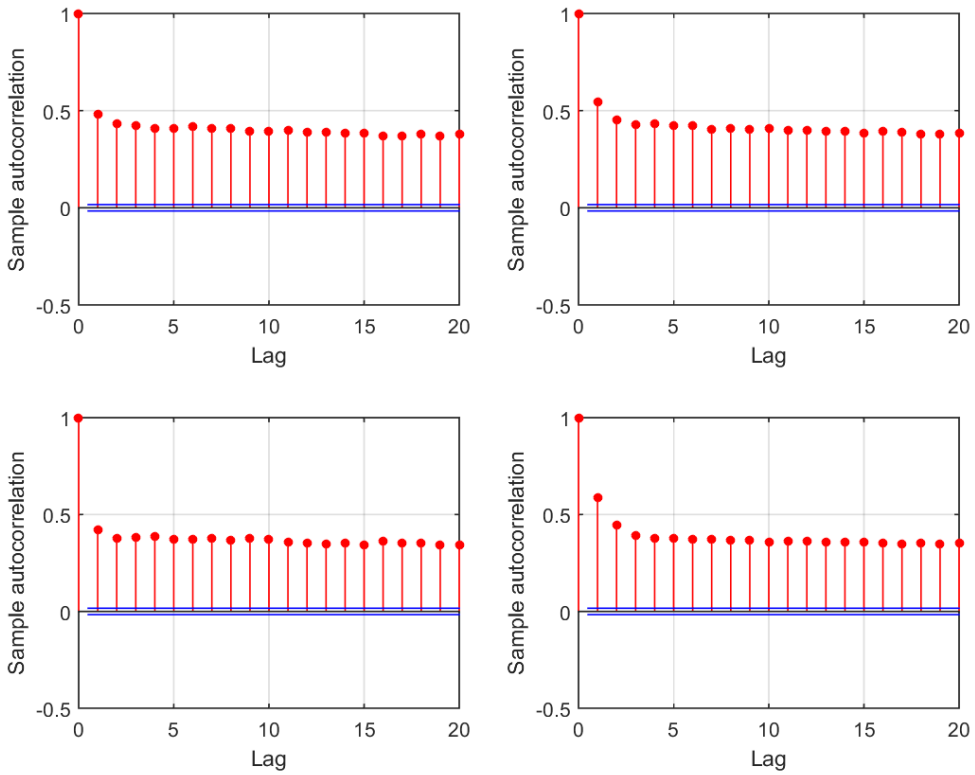


Figure 32: Sample autocorrelation functions on the BKNNs for the German data. Top left: original sampling; top right: randomized variance; bottom left: combined proposal; bottom right: derivative adjustment. All plots show that autocorrelation gradually decreases and therefore provide evidence of low autocorrelation in the chain.

Appendix D

Table 35: Two sample t-test results on the overall classification accuracy for the East-European data. 1 indicates that the mean difference is significant.

N	Method	1	2	3	4	5	6
1	BKNNs, GA	-	0	0	0	0	1
2	BKNNs	0	-	0	0	0	1
3	BKNNs, randomized variance	0	0	-	0	0	1
4	BKNNs, combined proposal	0	0	0	-	0	1
5	BKNNs, derivative	0	0	0	0	-	1
6	KNNs	1	1	1	1	1	-
7	DTs	1	1	1	1	1	1
8	DTs, boost	0	0	0	0	0	1
9	ANNs	1	1	1	1	1	1
10	SVMs	1	1	1	1	1	0
11	LDA	0	0	0	0	0	1
12	NB	0	0	0	0	0	1
13	Logistic	0	0	0	0	0	1

Table 36: Two sample t-test results on the overall classification accuracy for the Polish data. 1 indicates that the mean difference is significant.

N	Method	1	2	3	4	5	6
1	BKNNs, GA	-	0	0	0	0	1
2	BKNNs	0	-	0	0	0	1
3	BKNNs, randomized variance	0	0	-	0	0	1
4	BKNNs, combined proposal	0	0	0	-	0	1
5	BKNNs, derivative	0	0	0	0	-	1
6	KNNs	1	1	1	1	1	-
7	DTs	0	0	0	0	0	0
8	DTs, boost	0	0	0	0	0	1
9	ANNs	1	1	1	1	1	1
10	SVMs	0	1	1	1	1	0
11	LDA	0	1	1	1	1	0
12	NB	1	1	1	1	1	1
13	Logistic	0	0	0	0	0	0

Table 37: Two sample t-test results on the overall classification accuracy for the German data. 1 indicates that the mean difference is significant.

N	Method	1	2	3	4	5	6
1	BKNNs, GA	-	0	0	0	1	0
2	BKNNs	0	-	0	0	1	1
3	BKNNs, randomized variance	0	0	-	0	1	1
4	BKNNs, combined proposal	0	0	0	-	1	1
5	BKNNs, derivative	1	1	1	1	-	1
6	KNNs	0	1	1	1	1	-
7	DTs	1	1	1	1	1	1
8	DTs, boost	0	0	0	0	1	0
9	ANNs	0	0	0	0	0	1
10	SVMs	0	0	0	0	0	0
11	LDA	0	0	0	0	0	0
12	NB	0	0	0	0	0	0
13	Logistic	0	0	0	0	0	0

Appendix E

Table 38: Sensitivity to GA parameters, measured in terms of standard deviations for each parameter and each data set. Some very small deviation is observed when the population size is small (10 cases).

	German			Polish			East-European		
Sensitivity in respect to	β	k	F.V.	β	k	F.V.	β	k	F.V.
number of iterations	0	0	0	0	0	0	0	0	0
population size	0.46	0	0.14	0.05	11.6	0.13	0	0	0
penalty parameter	0	0	0	0	0	0	0	0	0

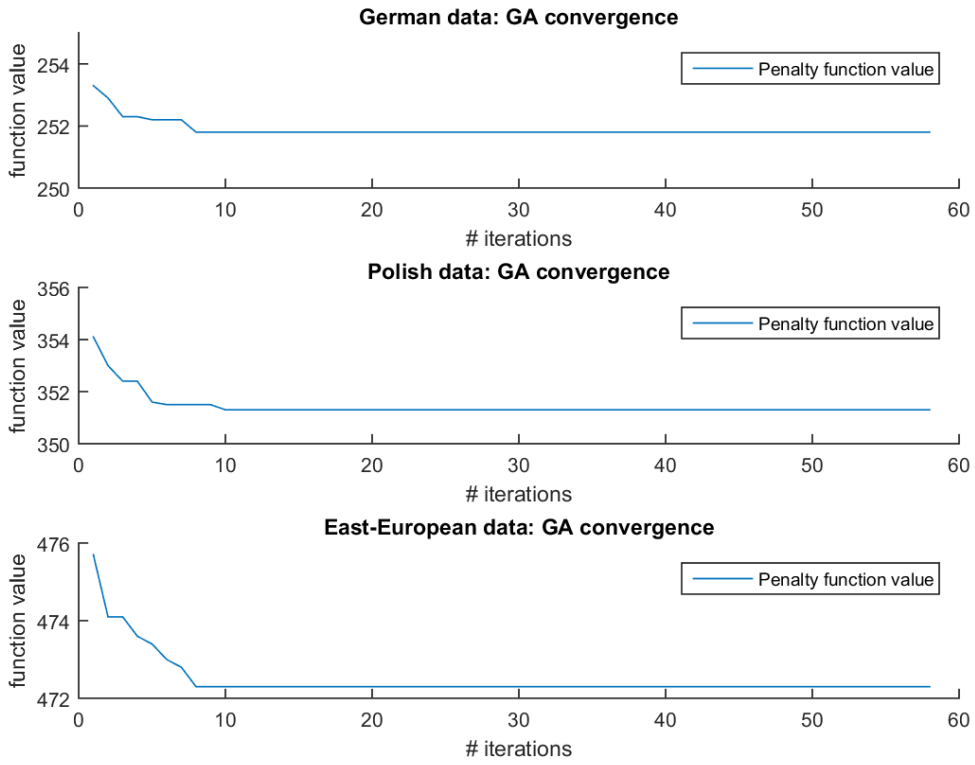


Figure 33: Penalty function value per GA iteration for each data set. After first 10 iterations the value of the penalty function stabilizes.

Bibliography

Abayomi, K., Gelman, A., and Levy, M. (2008). Diagnostics for multivariate imputations. *Applied Statistics*, 57(3):273–291. https://doi.org/10.1111/j.1467-9876.2007.00613.x.

Abdou, H. A., Dongmo, M. D., Collins, T., Ntim, G., and Baker, R. D. (2016). Predicting creditworthiness in retail banking with limited scoring data. *Knowledge-Based Systems*, 103:89–103. https://doi.org/10.1016/j.knosys.2016.03.023.

Addo, P. M., Guégan, D., and Hassani, B. (2018). Credit risk analysis using machine and deep learning models. Technical Report 18003, Université Panthéon-Sorbonne (Paris 1), Centre d'Economie de la Sorbonne. https://ideas.repec.org/p/mse/cesdoc/18003.html.

Agosto, A., Cavaliere, G., Kristensen, D., and Rahbek, A. (2016). Modeling corporate defaults: Poisson autoregressions with exogenous covariates (PARX). *Journal of Empirical Finance*, 38(Part B):640–663. https://doi.org/10.1016/j.jempfin.2016.02.007.

Agresti, A. (2019). *An Introduction to Categorical Data Analysis*, chapter Chapter 2, pages 25–56. Wiley, Hoboken, NJ, 3rd edition edition. http://users.stat.ufl.edu/~aa/.

Aha, D. W. and Bankert, R. L. (1996). A comparative evaluation of sequential feature selection algorithms. In Fisher, D. and Lenz, H.-J., editors, *Learning from Data, Lecture Notes in Statistics, vol 112*, pages 199–206, New York. Springer. https://doi.org/10.1007/978-1-4612-2404-4_19.

Al-Kassar, T. A. and Soileau, J. S. (2014). Financial performance evaluation and bankruptcy prediction (failure). *Arab Economic and Business Journal*, 9(2):147–155. https://doi.org/10.1016/j.aebj.2014.05.010.

Altman, E. I. (1968). Financial Ratios, Discriminant Analysis and the Prediction of Corporate Bankruptcy. *Journal of Finance*, XXIII Sep:189–209. https://doi.org/10.1111/j.1540-6261.1968.tb00843.x.

Altman, N. S. (1992). An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185. https://doi.org/10.2307/2685209.

Antonakis, A. C. and Sfakianakis, M. E. (2009). Assessing naive Bayes as a method

for screening credit applicants. *Journal of Applied Statistics*, 36(5):537–545. https://doi.org/10.1080/02664760802554263.

Ashiquzzaman, A., Tushar, A. K., Islam, M. R., Shon, D., Im, K., Park, J.-H., Lim, D.-S., and Kim, J. (2017). Reduction of overfitting in diabetes prediction using deep learning neural network. In Kim, K. J., Kim, H., and Baek, N., editors, *IT Convergence and Security 2017*, pages 35–43, Singapore. Springer Singapore. https://doi.org/10.1007/978-981-10-6451-7_5.

Atchade, Y. (2006). An adaptive version for the Metropolis adjusted Langevin algorithm with a truncated drift. *Methodology and Computing in Applied Probability*, 8(2):235–254. https://doi.org/10.1007/s11009-006-8550-0.

Bailey, M. (2006). *Practical Credit Scoring: Issues and Techniques*. White Box Publishing, New York, USA. https://www.amazon.ca/Practical-Credit-Scoring-Issues-Techniques/dp/095400535X.

Balaji, S. A. and Baskaran, K. (2013). Design and development of an artificial neural networking (ANN) system using sigmoid activation function to predict annual rice production in Tamilnadu. *International Journal of Computer Science, Engineering and Information Technology*, 3(1):13–31. https://doi.org/10.5121/ijcseit.2013.3102.

Barrell, R., Davis, E., Karim, D., and Liadze, I. (2010). The impact of global imbalances: Does the current account balance help to predict banking crises in oecd countries? NIESR Discussion Paper 351. https://www.niesr.ac.uk/sites/default/files/publications/dp351_0.pdf.

BCBS (December, 2017). *Basel III: Finalizing post-crisis reforms*. Bank for International Settlements. https://www.bis.org/bcbs/publ/d424.pdf.

Becker, N., Werft, W., Toedt, G., Lichter, P., and Benner, A. (2009). Penalized SVM: a R-package for feature selection SVM classification. *Bioinformatics*, 25(13):1711–1712. https://doi.org/10.1093/bioinformatics/btp286.

Bell, J. (2015). Artificial neural networks. In Long, C., editor, *Machine Learning: Hands-On for Developers and Technical Professionals*, pages 91–117, Indianapolis, IN. Wiley. https://doi.org/10.1002/9781119183464.

Bellotti, T. and Crook, J. (2009a). Credit scoring with macroeconomic variables using survival analysis. *Journal of the Operational Research Society*, 60(12):1699–1707. https://doi.org/10.1057/jors.2008.130.

Bellotti, T. and Crook, J. (2009b). Support vector machines for credit scoring and dis-

covery of significant features. *Expert Systems with Applications*, 36(2):3302–3308. https://doi.org/10.1016/j.eswa.2008.01.005.

Bellotti, T., Matousek, R., and Stewarti, C. (2011). A note comparing support vector machines and ordered choice models' predictions of international banks' ratings. *Decision Support Systems*, 51(3):682–687. https://doi.org/10.1016/j.dss.2011.03.008.

Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press, Oxford, UK. https://www.oupcanada.com/catalog/9780198538646.html.

Black, K. H. (2004). *Managing a Hedge Fund*. McGraw-Hill Professional, New York, USA. https://books.google.co.uk/books?id=vmsKEqZkFBYC.

Bonini, S. and Caivano, G. (2018). Probability of default modeling: A machine learning approach. In Corazza, M., Durbán, M., Grané, A., Perna, C., and Sibillo, M., editors, *Mathematical and Statistical Methods for Actuarial Sciences and Finance*, pages 173–177, Cham. Springer. https://doi.org/10.1007/978-3-319-89824-7_32.

Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, pages 144–152, New York. Association for Computing Machinery. https://doi.org/10.1145/130385.130401.

Box, G. E. P. and Cox, D. R. (1964). An analysis of transformations. *Journal of the Royal Statistical Society, Series B*, 26(2):211–252. http://www.jstor.org/stable/2984418.

Broomhead, D. and Lowe, D. (1988). Multivariable functional interpolation and adaptive networks. *Complex Systems*, 35(2):321–355. https://www.bibsonomy.org/bibtex/24ef3a0adaabe7e13dcdeee339068f840/mcdiaz.

Brown, I. and Mues, C. (2012). An experimental comparison of classification algorithms for imbalanced credit scoring data sets. *Expert Systems with Applications*, 39(3):3446–3453. https://doi.org/10.1016/j.eswa.2011.09.033.

Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27. https://www.csie.ntu.edu.tw/~cjlin/papers/libsvm.pdf.

Chen, S., Härdle, W. K., and Moro, R. A. (2011). Modeling default risk with support vector machines. *Quantitative Finance*, 11(1):135–154. https://doi.org/10.1080/14697680903410015.

Chen, Y.-W. and Lin, C.-J. (2006). Combining svms with various feature selection

strategies. In Guyon, I., Nikravesh, M., Gunn, S., and Zadeh, L. A., editors, *Feature Extraction: Foundations and Applications*, pages 315–324. Springer, Berlin. https://www.springer.com/gp/book/9783540354871.

Conway, D. and White, J. M. (2012). *Machine Learning for Hackers*. Oreilly Media Inc, USA. http://shop.oreilly.com/product/0636920018483.do.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 30(3):273–297. https://doi.org/10.1023/A:1022627411411.

Cox, D. (1958). The regression analysis of binary sequences (with discussion). *Journal of the Royal Statistical Society B*, 20(2):215–242. http://www.jstor.org/stable/2983890.

Cristianini, N. and Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Chapter 3. Cambridge University Press, Cambridge. https://doi.org/10.1017/CBO9780511801389.

Deb, K. (2000). An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering*, 186(2-4):311–338. https://doi.org/10.1016/S0045-7825(99)00389-8.

Deep, K., Singh, K. P., Kansal, M., and Mohan, C. (2009). A real coded genetic algorithm for solving integer and mixed integer optimization problems. *Applied Mathematics and Computation*, 212(2):505–518. https://doi.org/10.1016/j.amc.2009.02.044.

Deng, W., Smirnov, E., Timoshenko, D., and Andrianov, S. (2014). Comparison of regularization methods for ImageNet classification with deep convolutional neural networks. *AASRI Procedia*, 6:89–94. https://doi.org/10.1016/j.aasri.2014.05.013.

Dragos, C., Dragos, S., and Dumitru, A. (2008). Financial scoring: a literature review and experimental study. *Economic and Business Review for Central and South-Eastern Europe*, 10(1):53–68. https://www.researchgate.net/publication/284026030_Financial_scoring_a_literature_review_and_experimental_study.

Dreyfus, S. (1990). Artificial neural networks, back propagation, and the kelley-bryson gradient procedure. *Journal of Guidance Control and Dynamics*, 13(5):926–928. https://doi.org/10.2514/3.25422.

Durand, D. (1941). *Risk Elements in Consumer Instalment Financing*. National Bureau of Economy Research, Cambridge, MA. https://www.nber.org/books/dura41-2.

Everson, R. M. and Fieldsend, J. E. (2004). A variable metric probabilistic k-nearest-

neighbours classifier. In Yang, Z. R., Yin, H., and Everson, R. M., editors, *Intelligent Data Engineering and Automated Learning — IDEAL 2004: 5th International Conference, Exeter, UK. August 25-27, 2004. Proceedings*, pages 654–659, Berlin. Springer. https://doi.org/10.1007/978-3-540-28651-6_96.

Faez, A., Ahmadvand, Z., Mirzaei, M., and Ahmadvand, S. (2014). Neural network, decision tree and k–nearest neighbor methods for classification of credit customers of loans from bank. *Advances in Environmental Biology*, 8:1909–1914. http://www.aensiweb.com/old/aeb/2014/1909-1914.pdf.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J. (2012). LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874. http://dl.acm.org/citation.cfm?id=1390681.1442794.

Farhadi, F. (2017). Learning activation functions in deep neural networks. Diploma thesis, Department of Mathematics, Polytechnique Montréal. https://publications.polymtl.ca/2945/1/2017_FarnoushFarhadi.pdf.

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2):179–188. https://doi.org/10.1111/j.1469-1809.1936.tb02137.x.

Foresee, F. and Hagan, M. (1997). Gauss-Newton approximation to Bayesian learning. In *1997 International Joint Conference on Neural Networks*, volume 3, pages 1930–1935, New York, NY. IEEE. https://doi.org/10.1109/ICNN.1997.614194.

Gaganis, C., Pasiouras, F., Spathis, C., and Zopounidis, C. (2007). A comparison of nearest neighbours, discriminant and logit models for auditing decisions. *Intelligent Systems in Accounting, Finance & Management*, 15(1–2):23–40. https://doi.org/10.1002/isaf.283.

Gavalas, D. (2015). How do banks perform under Basel III? Tracing lending rates and loan quantity. *Journal of Economics and Business*, 81(September–October):21–37. https://EconPapers.repec.org/RePEc:eee:jebusi:v:81:y:2015:i:c:p:21-37.

Gelfand, A. and Smith, A. (1990). Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85(410):398–409. https://doi.org/10.1080/01621459.1990.10476213.

Gelman, A. and Hennig, C. (2017). Beyond subjective and objective in statistics. *Journal of the Royal Statistical Society: Series A*, 180(4):967–1033. https://doi.org/10.1111/rssa.12276.

Gill, P. and Murray, W. (1978). Algorithms for the solution of the nonlinear least-squares problem. *SIAM Journal on Numerical Analysis*, 15(5):977–992. https://doi.org/10.1137/0715063.

Gök, M. (2015). An ensemble of k-nearest neighbours algorithm for detection of Parkinson's disease. *International Journal of Systems Science*, 46(6):1108–1112. https://doi.org/10.1080/00207721.2013.809613.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Boston. https://books.google.bg/books?id=3_RQAAAAMAAJ.

Güneş, S., Polat, K., and Yosunkaya, Ç. (2010). Multi-class f-score feature selection approach to classification of obstructive sleep apnea syndrome. *Expert Systems with Applications*, 37(2):998–1004. https://doi.org/10.1016/j.eswa.2009.05.075.

Guo, R. and Chakraborty, S. (2010). Bayesian adaptive nearest neighbor. *Statistical Analysis and Data Mining*, 3(2):92–105. https://doi.org/10.1002/sam.10067.

Guyon, I. and Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3(1):1157–1182. http://dl.acm.org/citation.cfm?id=944919.944968.

Hagan, M. T., Demuth, H. B., Beale, M. H., and De Jesús, O. (2014). *Neural Network Design*. Martin Hagan, 2nd ed. edition. https://books.google.co.uk/books?id=4EW9oQEACAAJ.

Hammer, P. L., Kogan, A., and Lejeune, M. A. (2012). A logical analysis of banks' financial strength ratings. *Expert Systems with Applications*, 39(9):7808–7821. https://doi.org/10.1016/j.eswa.2012.01.087.

Han, J. and Moraga, C. (1995). The influence of the sigmoid function parameters on the speed of backpropagation learning. In Mira, J. and Sandoval, F., editors, *From Natural to Artificial Neural Computation*, Proceedings of the International Workshop on Artificial Neural Networks, Malaga-Torremolinos, Spain, 7–9 June, 1995, pages 195–201, Berlin. Springer. https://doi.org/10.1007/3-540-59497-3_175.

Hand, D. J. and Yu, K. (2001). Idiot's Bayes — not so stupid after all? *International Statistical Review*, 69(3):385–399. http://doi.org/10.1111/j.1751-5823.2001.tb00465.x.

Harris, T. (2015). Credit scoring using the clustered support vector machine. *Expert Sys-*

tems with Applications, 42(2):741–750. https://doi.org/10.1016/j.eswa.2014.08.029.

Heaton, J. B., Polson, N. G., and Witte, J. H. (2017). Deep learning in finance: deep portfolios. *Applied Stochastic Models in Business and Industry*, 33(1):3–12. https://onlinelibrary.wiley.com/doi/abs/10.1002/asmb.2209.

Henley, W. E. and Hand, D. J. (1996). A k-nearest-neighbour classifier for assessing consumer credit risk. *Journal of the Royal Statistical Society. Series D (The Statistician)*, 45(1):77–95. https://www.jstor.org/stable/2348414.

Hens, A. B. and Tiwari, M. K. (2012). Computational time reduction for credit scoring: An integrated approach based on support vector machine and stratified sampling method. *Expert Systems with Applications*, 39(8):6774–6781. https://doi.org/10.1016/j.eswa.2011.12.057.

Hindi, K. E. and Al-Akhras, M. (2011). Smoothing decision boundaries to avoid overfitting in neural network training. *Neural Network World*, 21(4):311–325. http://www.nnw.cz/doi/2011/NNW.2011.21.019.pdf.

Hira, Z. M. and Gillies, D. F. (2015). A review of feature selection and feature extraction methods applied on microarray data. *Advances in Bioinformatics*, 2015:198363:1–13. https://doi.org/10.1155/2015/198363.

Hofmann, H. (1994). Statlog German credit data set. https://archive.ics.uci.edu/ml/datasets/Statlog+(German+Credit+Data).

Holmes, C. C. and Adams, N. M. (2002). A probabilistic nearest neighbour method for statistical pattern recognition. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 64(2):295–306. http://www.jstor.org/stable/3088801.

Hosaka, T. and Takata, Y. (2016). Corporate bankruptcy forecast using RealAdaBoost. *Information : an international Interdisciplinary journal*, 19(6B):2285–2298. https://ci.nii.ac.jp/naid/40020905300/en/.

Huang, C.-L., Chen, M.-C., and Wang, C.-J. (2007). Credit scoring with a data mining approach based on support vector machines. *Expert Systems with Applications*, 33(4):847–856. https://doi.org/10.1016/j.eswa.2006.07.007.

Huang, S.-C. and Huang, M.-H. (2010). Using SVM with embedded recursive feature selections for credit rating forecasting. *Journal of Statistics and Management Systems*, 13(1):165–177. https://doi.org/10.1080/09720510.2010.10701462.

Huang, S.-C. and Wu, C. F. (2011). Customer credit quality assessments using

data mining methods for banking industries. *South African Journal of Business Management*, 5(11):4438–4445. https://pdfs.semanticscholar.org/2786/4edc69a79d4dfce0a05f1efd4ecc4dc76059.pdf.

Kalaycı, S., Kamasak, M., and Arslan, S. (2018). Credit risk analysis using machine learning algorithms. In *26th Signal Processing and Communications Applications Conference (SIU 2018)*, pages 821–824, Red Hook, NY. IEEE/Curran Associates. https://doi.org/10.1109/SIU.2018.8404353.

Kamath, C. (2009). *Scientific Data Mining : A Practical Perspective*. Society for Industrial and Applied Mathematics,U.S., New York, United States. https://doi.org/10.1137/1.9780898717693.

Kim, H. B., Jung, S. H., Kim, T. G., and Park, K. H. (1996). Fast learning method for back-propagation neural network by evolutionary adaptation of learning rates. *Neurocomputing*, 11(1):101–106. https://doi.org/10.1016/0925-2312(96)00009-4.

Kuhn, M. and Johnson, K. (2013). *Applied Predictive Modeling*. Springer, New York. https://doi.org/10.1007/978-1-4614-6849-3.

Lampinen, J. and Vehtari, A. (2001). Bayesian approach for neural networks – Reviews and case studies. *Neural Networks*, 14(3):257–274. https://doi.org/10.1016/S0893-6080(00)00098-8.

Lessmann, S., Baesens, B., Seow, H.-V., and Thomas, L. C. (2015). Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. *European Journal of Operational Research*, 247(1):124–136. https://doi.org/10.1016/j.ejor.2015.05.030.

Li, S.-T., Shiue, W., and Huang, M.-H. (2006). The evaluation of consumer loans using support vector machines. *Expert Systems with Applications*, 30(4):772–782. https://doi.org/10.1016/j.eswa.2005.07.041.

Liang, D., Lu, C.-C., Tsai, C.-F., and Shih, G.-A. (2016). Financial ratios and corporate governance indicators in bankruptcy prediction: A comprehensive study. *European Journal of Operational Research*, 252(2):561–572. https://doi.org/10.1016/j.ejor.2016.01.012.

Liu, Z., Pan, Q., and Dezert, J. (2013). A newbelief-based k-nearest neighbor classification method. *Pattern Recognition*, 46(3):834–844. https://doi.org/10.1016/j.patcog.2012.10.001.

MacKay, D. (1992). Bayesian interpolation. *Neural Computing*, 4(3):415–447. https://doi.org/10.1162/neco.1992.4.3.415.

Manocha, S. and Girolami, M. A. (2007). An empirical analysis of the probabilistic k-nearest neighbour classifier. *Pattern Recognition Letters*, 28(13):1818–1824. https://doi.org/10.1016/j.patrec.2007.05.018.

Marilena, M. and Alina, T. (2015). The significance of financial and non-financial information in insolvency risk detection. *Procedia Economics and Finance*, 26:750–756. https://doi.org/10.1016/S2212-5671(15)00834-5.

Merton, R. C. (1974). On the pricing of corporate debt: The risk structure of interest rates. *Journal of Finance*, 29(2):449–470. https://doi.org/10.1111/j.1540-6261.1974.tb03058.x.

Meyer, D., Leisch, F., and Hornik, K. (2003). The support vector machine under test. *Neurocomputing*, 55(1–2):169–186. https://doi.org/10.1016/S0925-2312(03)00431-4.

Mukaka, M. (2012). Statistics corner: A guide to appropriate use of correlation coefficient in medical research. *Malawi medical journal : the journal of Medical Association of Malawi*, 24(3):69–71. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3576830/.

Mukherjee, S. (2003). Classifying microarray data using support vector machines. In Berrar, D. P., Dubitzky, W., and Granzow, M., editors, *A Practical Approach to Microarray Data Analysis*, pages 166–185, New York. Springer. https://doi.org/10.1007/0-306-47815-3_9.

Neal, R. (1992). Bayesian training of backpropagation networks by the hybrid Monte Carlo method. Technical Report CRG-TR-92-1, Department of Computer Science, University of Toronto. ftp://www.cs.toronto.edu/pub/radford/bbp.pdf.

Neal, R. M. (1996). *Bayesian Learning for Neural Networks*. Springer-Verlag, Berlin, Heidelberg. https://dl.acm.org/citation.cfm?id=525544.

Nicolae-Eugen, C. (2016). Lowering evolved artificial neural network overfitting through high-probability mutation. In Davenport, J., Negru, V., Ida, T., Jebelean, T., Petcu, D., Watt, S., and Zaharie, D., editors, *SYNASC 2016 — 18th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing*, pages 325–329. IEEE, Los Alamitos, CA. https://doi.org/10.1109/SYNASC.2016.059.

Onali, E. and Ginesti, G. (2014). Pre-adoption market reaction to IFRS 9: A cross-

country event-study. *Journal of Accounting and Public Policy*, 33(6):628–637. https://doi.org/10.1016/j.jaccpubpol.2014.08.004.

Pérez-Martín, A., Pérez-Torregrosa, A., and Vaca, M. (2018). Big data techniques to measure credit banking risk in home equity loans. *Journal of Business Research*, 89:448–454. http://www.sciencedirect.com/science/article/pii/S0148296318300833.

Platt, J. (1999). Probabilistic outputs for support vector machines and comparison to regularize likelihood methods. In Smola, A., Bartlett, P., Schoelkopf, B., and Schuurmans, D., editors, *Advances in Large Margin Classifiers*, pages 61–74. MIT Press. http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.41.1639.

Pluto, K. and Tasche, D. (2011). Estimating Probabilities of Default for Low Default Portfolios. In Engelmann, B. and Rauhmeier, R., editors, *The Basel II Risk Parameters*, pages 201–213. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-16114-8_5.

Quanquan, G., Zhenhui, L., and Jiawei, H. (2011). Generalized fisher score for feature selection. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, pages 266–273, Arlington, Virginia, United States. AUAI Press. http://dl.acm.org/citation.cfm?id=3020548.3020580.

Quinlan, J. R. (1999). Simplifying decision trees. *International Journal of Human-Computer Studies*, 51(2):497–510. https://doi.org/10.1006/ijhc.1987.0321.

Rakotomamonjy, A. (2003). Variable selection using SVM-based criteria. *Journal of Machine Learning Research*, 3:1357–1370. http://dl.acm.org/citation.cfm?id=944919.944977.

Rasmussen, C. E. (1996). A practical Monte Carlo implementation of Bayesian learning. In Touretzky, D. S., Mozer, M. C., and Hasselmo, M. E., editors, *Advances in Neural Information Processing Systems 8*, pages 598–604. MIT Press, Cambridge, MA. http://dl.acm.org/citation.cfm?id=2998828.2998913.

Rockafellar, R. T. (1993). Lagrange multipliers and optimality. *SIAM Review*, 35(2):183–238. https://doi.org/10.1137/1035044.

Sariev, E. and Germano, G. (2019). An innovative feature selection method for support vector machines and its test on the estimation of the credit risk of default. *Review of Financial Economics*, 37(3):404–427. https://doi.org/10.1002/rfe.1049.

Sariev, E. and Germano, G. (2020). Bayesian regularized artificial neural networks for the

estimation of the probability of default. *Quantitative Finance*, 20(2):311–328. https://doi.org/10.1080/14697688.2019.1633014.

Schober, P., Boer, C., and Schwarte, L. (2018). Correlation coefficients: Appropriate use and interpretation. *Anesthesia & Analgesia*, 126(5):1763–1768. https://doi.org/10.1213/ANE.0000000000002864.

Simkovic, M. and Kaminetzky, B. (2011). Leveraged Buyout Bankruptcies, the Problem of Hindsight Bias, and the Credit Default Swap Solution. *Columbia Business Law Review*, 1:118. http://ssrn.com/abstract=1632084.

Specht, D. (1990). Probabilistic neural networks. *Neural Networks*, 3(1):109–118. https://doi.org/10.1016/0893-6080(90)90049-Q.

Stallkamp, J., Schlipsing, M., Salmen, J., and Igel, C. (2012). Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, 32:323–332. https://doi.org/10.1016/j.neunet.2012.02.016.

Strasburger, H. (2001). Converting between measures of slope of the psychometric function. *Perception & Psychophysics*, 63:1348–1355. https://doi.org/10.3758/BF03194547.

Su, W., Chipman, H., and Zhu, M. (2008). On the underestimation of model uncertainty by Bayesian k-nearest neighbors. arXiv:0804.1325. https://arxiv.org/abs/0804.1325.

Theodoridis Sergios, K. K. (2009). *Pattern Recognition*. Academic Press, Stockholm, 4 edition. https://www.bokus.com/bok/9781597492720/pattern-recognition-4th-edition/.

Thuraisingham, B. M. (1999). *Data Mining Technologies, Techniques, Tools and Trends*. Taylor and Francis Inc, Bosa Roca, United States. https://books.google.co.uk/books?id=UX9yMMpbLFkC.

Tian, S., Yu, Y., and Guo, H. (2015). Variable selection and corporate bankruptcy forecasts. *Journal of Banking & Finance*, 52(Supplement C):89–100. https://doi.org/10.1016/j.jbankfin.2014.12.003.

Titterington, D. M. (2004). Bayesian methods for neural networks and related models. *Statistical Science*, 19(1):128–139. https://projecteuclid.org/euclid.ss/1089808278.

Tomczak, S. (2016). Polish companies bankruptcy data set. https://archive.ics.uci.edu/ml/datasets/Polish+companies+bankruptcy+data.

Villa, J. L., Boqué, R., and Feré, J. (2008). Calculation of the probability of correct classification in probabilistic bagged k-nearest neighbours. *Chemometrics and Intelligent Laboratory Systems*, 94(1):51–59. https://doi.org/10.1016/j.chemolab.2008.06.007.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., and Manzagol, P. (2010). Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 10:3371–3408. http://www.jmlr.org/papers/volume11/vincent10a/vincent10a.pdf.

Wang, J.-H. and Peng, C.-Y. (2000). A novel self-creating neural network for learning vector quantization. *Neural Processing Letters*, 11(2):139–151. https://doi.org/10.1023/A:1009626513932.

Weston, J., Mukherjee, S., Chapelle, O., Pontil, M., Poggio, T., and Vapnik, V. (2000). Feature selection for SVMs. *Advances in Neural Information Processing Systems*, 13:668–674. http://hdl.handle.net/1721.1/102484.

Yoon, J. W. and Friel, N. (2015). Efficient model selection for probabilistic k nearest neighbour classification. *Neurocomputing*, 149(Part B):1098–1108. https://doi.org/10.1016/j.neucom.2014.07.023.

Zhang, C., Vinyals, O., Munos, R., and Bengio, S. (2018). A study on overfitting in deep reinforcement learning. arXiv:1804.06893. http://arxiv.org/abs/1804.06893.