University of Economics in Prague

Faculty of Finance

Department of Banking and Insurance

Field of study: Financial Engineering

# Support Vector Machines for Credit Scoring

Author of the Master Thesis: Bc. Michal Haltuf

Supervisor of the Master Thesis: doc. RNDr. Jiří Witzany, Ph.D.

Year of Defence: 2014

# The Declaration of Authorship

I hereby declare that I carried out the master thesis "Support Vector Machines in Credit Scoring" independently, using only the resources and literature properly marked and included in the bibliography.

Prague, 24th August, 2014 signature of the author

# Acknowledgments

I would like to thank to doc. RNDr. Jiří Witzany, Ph.D. for his comments and valuable feedback, my family and my partner for her continuous moral support.

**Abstract:**
Quantitative methods to assess the creditworthiness of the loan applicants are vital for the profitability and the transparency of the lending business. With the total loan volumes typical for traditional financial institutions, even the slightest improvement in credit scoring models can translate into substantial additional profit. Yet for the regulatory reasons and due to the potential model risk, banks tend to be reluctant to replace the logistic regression as an industrial standard with the new algorithms. This does not stop researchers from examining such new approaches, though. This thesis discusses the potential of the support vector machines, to become an alternative to logistic regression in credit scoring. Using the real-life credit data set obtained from the P2P lending platform Bondora, the scoring models were built to compare the discrimination power of support vector machines against the traditional approach. The results of the comparison were ambiguous. The linear support vector machines performed worse than logistic regression and their training consumed much more time. On the other hand, support vector machines with non-linear kernel performed better than logistic regression and the difference was statistically significant at 95% level. Despite this success, several factors prevent SVM from the widespread applications in credit scoring, higher training times and lower robustness of the method being two of the major drawbacks. Considering the alternative algorithms which became available in the last 10 years, support vector machines cannot be recommended as a standalone method for credit risk models.

**Keywords:**
support vector machines, logistic regression, credit scoring, peer to peer lending

**Abstrakt:**
Využití kreditních modelů v rozhodování o přidělení půjček retailovým zákazníkům je dnes ve finančním sektoru již běžnou záležitostí a představuje důležitou složku pro udržení ziskovosti i transparentnosti celého procesu. Při objemech, s nimiž poskytovatelé úvěrů běžně pracují, představuje i sebenepatrnější zlepšení účinnosti používaných modelů významné dodatečné zisky. Finanční instituce však (z důvodů regulatorních pravidel i z opatrnosti kvůli možnému modelovému riziku) preferují pro tyto účely logistickou regresi před novými a potenciálně účinnějšími metodami. To však neznamená, že by výzkum nových přístupů měl ustat. Podpůrné vektorové stroje (SVM) patří k těmto alternativním přístupům. Tato práce zkoumá možnost jejich uplatnění při tvorbě kreditních modelů. Srovnává výkonnost modelů založených na SVM oproti tradičnímu přístup pomocí logistické regrese, a to na reálných kreditních datech získaných z platformy zaměřené na P2P půjčky. Lineární verze podpůrných vektorových strojů byla v rozlišení dobrých a špatných dlužníků méně úspěšná než klasická logistická regrese. Naopak SVM model s nelinární jádrovou funkcí byl schopen logistickou regresi překonat a tento rozdíl ve výkonnosti byl statisticky významný. Navzdory tomuto dílčímu úspěchu se ale praktické využití podpůrných vektorových strojů v tomto oboru pojí s celou řadou obtíží, mezi něž patří dlouhá doba vývoje modelu a jeho nižší robustnost. Vezmeme-li v úvahu nové algoritmy, s nimiž odborná literatura přišla v posledních 10 letech a které jsou schopny soustavně dosahovat lepších výsledků, nelze metodu podpůrných vektorových strojů pro využití v kreditních modelech doporučit.

**Klíčová slova:**
logistická regrese, podpůrné vektorové stroje, svm, kreditní skóring, p2p půjčky

# Contents

**List of Tables** ix

**List of Figures** x

**Abreviations** xi

**Notation** xii

**Introduction** 1

**1 Linear classifiers** 3
- 1.1 Geometric interpretation of the data points . . . 6
- 1.2 Classification formalized . . . 7
- 1.3 Logistic regression . . . 10

**2 Support vector machines** 13
- 2.1 Maximum margin hyperplane . . . 13
- 2.2 Lagrangian methods for optimization . . . 16
  - 2.2.1 Primal problem . . . 17
  - 2.2.2 Dual problem . . . 18
  - 2.2.3 Application to maximum margin hyperplane . . . 19
- 2.3 Kernels and kernel trick . . . 21
  - 2.3.1 Linear kernel . . . 24
  - 2.3.2 Polynomial kernel . . . 24
  - 2.3.3 Gaussian kernel (Radial basis function) . . . 25
  - 2.3.4 Other kernels . . . 25
- 2.4 Soft margin classifiers . . . 26
- 2.5 Algorithm implementation . . . 29

**3 Model building** 31
- 3.1 Data preprocessing . . . 31
  - 3.1.1 Categorical variables . . . 31
  - 3.1.2 Scaling and standardizing . . . 32
  - 3.1.3 Grid search . . . 32
  - 3.1.4 Missing values . . . 34
- 3.2 Feature selection . . . 34
  - 3.2.1 Forward selection . . . 35
  - 3.2.2 Backward selection . . . 36
  - 3.2.3 F-score . . . 36

3.2.4 Genetic algorithm . . . 37
3.3 Model selection . . . 38
3.3.1 Hold-out cross validation . . . 39
3.3.2 k-fold cross validation . . . 39
3.3.3 Leave-one-out cross validation . . . 39
3.4 Model evaluation . . . 40

**4 Support vector machines in Credit risk** **42**

**5 Application** **47**
5.1 Peer to Peer Lending . . . 47
5.2 Data description . . . 51
5.3 Benchmark model: Logistic regression . . . 54
5.4 SVM with Linear kernel . . . 55
5.4.1 Optimal cost parameter search . . . 57
5.5 SVM with Gaussian kernel . . . 58
5.6 Model results . . . 59
5.7 Quantifying the edge . . . 66

**Conclusion** **70**

**Bibliography** **72**

**A Appendix** **I**

# List of Tables


1 The confusion matrix . . . . . 40
2 List of explanatory variables . . . . . 53
3 Backward selection, variables significance . . . . . 55
4 Results of Logistic regression model . . . . . 60
5 Results of SVM with the Linear kernel . . . . . 61
6 Results of SVM with the Gaussian kernel . . . . . 62
7 Comparison of models . . . . . 64
8 Backtesting simulation results . . . . . 68
9 LR: Analysis of effect (dummy variables) . . . . . I
10 LR: Maximum likelihood estimates (dummy variables) . . . . . III
11 LR: Analysis of effects (woeised) . . . . . III
12 LR: Maximum likelihood estimates (woeised) . . . . . IV

# List of Figures


1 An example of a linearly separable binary classification problem . 5
2 Linear decision functions of LR and SVM on the same data set . . 12
3 Geometric margin and maximum-margin hyperplane . . . . . 15
4 Transformation into the higher dimension space. Kernel trick . . . 23
5 Slack variable in the soft margin . . . . . 27
6 Example of two-step grid search . . . . . 33
7 SVM training times . . . . . 59
8 Comparison of the ROC curves . . . . . 65
9 Histogram of profit/loss after 1 year . . . . . 69

# Abbreviations

| | |
|---|---|
| AUC | Area Under Curve. |
| DTI | Debt to Income. |
| GA | Genetic Algorithm. |
| KKT | Karush-Kuhn-Tucker. |
| kNN | k-Nearest Neighbours. |
| LC | Lending Club. |
| LDA | Linear Discriminant Analysis. |
| LR | Logistic Regression. |
| MLE | Maximum likelihood estimate. |
| P2P | Peer to Peer, in this context as peer to peer lending. |
| RBF | Radial Basis Function. |
| RF | Random Forests. |
| ROC | Receiver Operation Characteristic. |
| ROI | Return on Investment. |
| SMO | Sequential Minimal Optimization. |
| SVM | Support Vector Machines. |
| VaR | Value at Risk. |
| VC | Vapnik-Chervonenkis. |
| WoE | Weight of Evidence. |

# Notation

| | |
|---|---|
| $b$ | Bias, along with weight vector defines the hyperplane. |
| $C$ | Cost parameter in the soft-margin algorithm. |
| $K$ | Number of sets in the K-fold cross validation. |
| $K(\boldsymbol{x},\boldsymbol{z})$ | Kernel function to transform data from the original space to the feature space. |
| $m$ | Number of observations in the training set. |
| $n$ | Dimension of the input space. |
| $\boldsymbol{w}$ | The vector of weights. Vector normal to the separating hyperplane. |
| $\boldsymbol{x}$ | Input. The vector of properties of one observation. |
| $y$ | Binary output (+1/-1). |
| $\boldsymbol{\alpha}$ | Vector of Lagrangian multipliers. |
| $\boldsymbol{\alpha}^*,b^*$ | Optimal values, the optimization results. |
| $\gamma$ | Shape parameter of the radial basis function (Gaussian kernel). |
| $\delta$ | Geometric margin. |
| $\hat{\delta}$ | Functional margin. |
| $\xi$ | Slack variable in the soft-margin algorithm. |

# Introduction

Credit scoring is a standard method of assessing the creditworthiness of the loan applicants in the banking industry. Over time, objective quantitative tools have been developed and adopted to suppress the subjective ad-hoc elements in the decision process, strengthen the accountability and substitutability and reduce the chances for corruption.

Various statistical and computer science methods can be used to build an objective model to differentiate the loan applicants and to estimate the probability of their default. Support Vector Machines derived and proved by Vapnik is an example of such a method.

One of the principal goal of my thesis is to examine the performance of support vector machines (SVM) in credit scoring and to compare them with the logistic regression (LR) which still remains the industry standard in banking. In the first chapter, I will show that SVM and LR share some common properties and that they both belong to a wide family of linear classifiers.

The second chapter is dedicated to the mathematical derivation of the support vector machines as it was developed throughout 30 years of their evolution: from the the hard maximum-margin classifier, which works only for the linearly separable data, to the current form using the soft margin and the kernel trick to cover non-linear and noisy data.

The third chapter is focused on the practical problems of the credit scoring models development: how to measure the performance, how to prepare the data set, how to evaluate different models and choose the best one among all possibilities.

The fourth chapter summarizes the current state of knowledge in the credit scoring, based on the peer-reviewed literature with a special focus on the support vector machines.

The banking industry is, due to the understandable carefulness and the model risk, reluctant to adopt new approaches in the credit scoring, unless it introduced a major performance advantage. Therefore, I concentrate on the application of SVMs in the newly emerging sector of the loan business: the peer-to-peer lending, which as a new phenomenon is described in detail the fifth chapter.

The P2P lending is characteristic for the prevailing presence of the amateur investors, whose strategy is based either on a blind diversification or on a naïve and rather empirical credit scoring that embodies one or two factors. Credit scoring based on the objective and quantitative methods could under such conditions lead to the excess returns when applied by a concerned investor.

The last chapter describes the procedures and methods used to build a support vector machine for credit scoring on a real data set obtained from one of the leading P2P platforms in the Europe. Performance of the model is discussed as well as the comparison with the industry-standard logistic regression.

As the chosen P2P platform is open for small investors from all over the Europe, my findings could be directly applied in practice by anyone willing to build an investment strategy on the top of them. This thesis is, however, written solely for the educational purposes and such proceeding is therefore not recommended.

# 1. Linear classifiers

The main task of the credit scoring is simple: build a model, which will be able to distinguish between a good creditor and a bad one. Or put in other words: take the past credit data set and use them to *learn* the rules *generally* applicable for the future discrimination between the two creditor classes, with minimal number of mistakes both in false positives and false negatives.

The task to assign an observation to its respective class is called *Classification*.

Since for the purpose of the credit scoring, there are only two possible categories (e.g. 0 = good creditor, 1 = bad creditor), this specific setting of the classification is referred as *Binary classification*. Apart from the credit scoring, the binary classification has a wide variety of applications; no wonder there have been several different approaches developed in the literature.

No fewer than two scientific fields are intersecting in this task: Statistics and Machine Learning.[1] While the methods and scope of these fields may be different in general, it is practically impossible to draw a clear line between them in the context of the data classification problem.[2]

Machine Learning accents the ability to make a correct prediction, while Statistics accents the ability to describe the relationships among the model variables and to quantify the contribution of each one of them.

---

[1] being part of a more general Computer Science field

[2] Consider for example the Logistic Regression, which is frequently used for classification. Logistic regression is widely considered to be a statistical tool and is still a standard performance benchmark for the credit scoring in the real-life banking [16]. On the other hand, Support Vector Machines (SVMs), which are the main topic of this thesis, are commonly referred as a machine learning tool. Yet they are actually very similar to each other: they both have a same goal and purpose and they both can be expressed as a hyperplane separating the space into two subspaces. The main practical difference between them lies in the terminology. Due to the key focus of my thesis, I will prefer the machine learning terminology, where appropriate.

The classification problem is sometimes called *supervised learning*, because the method operates under supervision [58]. As mentioned above, the learning process in the credit scoring usually begins with some historical, labelled[3] observations. These observations are often referred as the *training set*.

The goal of the supervised learning is to derive a mapping (function) which not only can correctly describe the data in the training set, but more importantly is able to *generalize* from the training set to the unobserved situations. Since the main application of the classifier models is the prediction, the generalization ability of the learning algorithms are by far the most important property that distinguishes a good classifier from the bad one. [28]

The function derived from the supervised learning introduces a border between two classes - we say it forms a *decision boundary* or *decision surface*. Depending on the shape of this decision boundary we distinguish *linear classifiers* and *non-linear classifiers*. The linear classifiers represent a wide family of algorithms, whose common characteristic is that the decision is based on the linear combination of the input variables [51, p. 30]. All the classifiers described and used in this thesis, namely Logistic regression and Support vector machines, can be regarded as members of this wide family.

Fisher in his classic paper [23] introduced his linear discriminant, which was probably the first linear classifier. The Iris Data set [2], published in the same paper to demonstrate the power of his linear discriminant, became eventually the most cited classification data set, used widely for educational and explanation purposes. A subset of this data set is presented at Figure 1. It demonstrates a simple example of the binary classification problem with 2 explanatory variables and 100 observations. The goal of the classification is to draw a line that could serve as a borderline between the two different classes. Since the data are clearly *linearly separable*, it does not appear to be much a challenging task.

**Definition 1.1.** Two subsets X and Y of $\mathbb{R}^d$ are said to be **linearly separable** if there exists a hyperplane of $\mathbb{R}^d$ such that the elements of X and those of Y lie on opposite sides of it. [20] □

[3] i.e. we know, whether the loan applicant turned to be good or bad

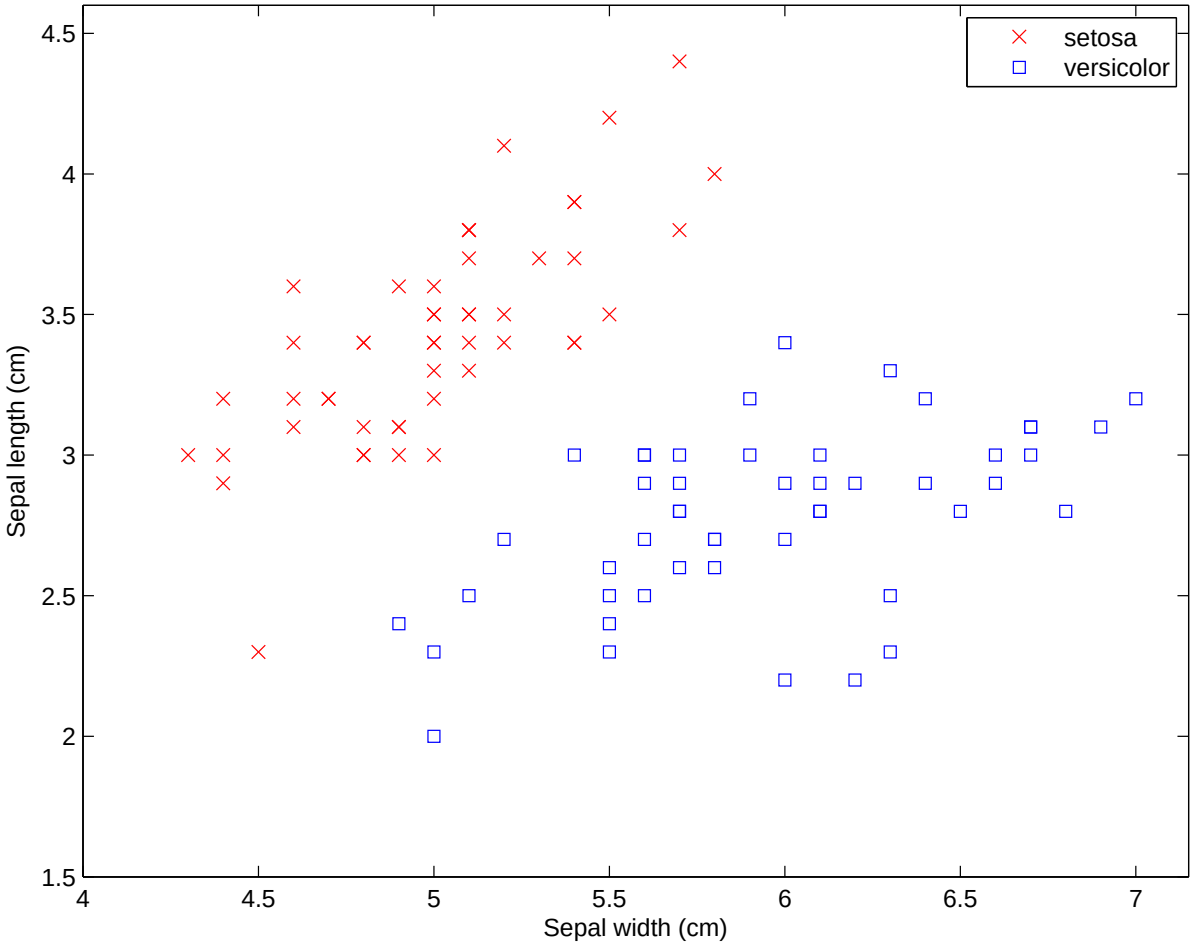


*Figure 1: An example of a linearly separable binary classification problem based on the Fisher Iris Data (citation and detail in the text). In principle there is an infinite number of lines that could serve as a decision boundary perfectly separating the two classes.*

In principle, there is an infinite number of lines capable of performing this task. However, we can intuitively feel, that not all of them are equally good at it and that there are lines which accomplish this task better than others. Before we define what "good" and "better" means in this context, let's have a closer look at the Iris dataset.

## 1.1 Geometric interpretation of the data points

As already mentioned, there are 100 data points (observations) with 2 input variables (sepal width and sepal length in cm) and one binary class variable (setosa/versicolor). The mere fact that, in Figure 1, we chose to display the data set as a two-dimensional scatter plot implicitly assumes, that each property can be regarded as a dimension in a coordinate system. Each data point can therefore be interpreted geometrically, as a point in a coordinate system.

This can be taken even further, though. Each data point can be visualised as a *vector* rooted in the origin of a coordinate system.[4] [28, p. 33]

**Definition 1.2. Vector** is a directed line segment, that has both length and direction. [40, p. 355] □

**Definition 1.3.** Given the components of vector, $\boldsymbol{a} = (a_1, a_2, \ldots, a_n)$, the **length** or **norm** of a vector $\boldsymbol{a}$, written as $\|\boldsymbol{a}\|$, is defined by:

$$\|\boldsymbol{a}\| = \sqrt{a_1^2 + a_2^2 + \cdots + a_n^2} \tag{1.1}$$

□

The geometric interpretation of vectors is conceptually very advantageous. Effectively, we have converted our data universe into a vector space. By allowing this, we can compute new objects using algebraic vector operations. More, it is possible to calculate the *dot products* of two vectors which can then serve as a measure of similarity between two observations.[5] [28, p. 34]

**Definition 1.4.** Given two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ from $\mathbb{R}$, the **dot product** or **inner product** of $\boldsymbol{a} \cdot \boldsymbol{b}$ is:

$$\boldsymbol{a} \cdot \boldsymbol{b} = \sum_{i=1}^{n} a_i b_i \tag{1.2}$$

□

**Remark 1.1.** There is a relation between the length of a vector and the dot product of a vector with itself:

$$\|\boldsymbol{a}\| = \sqrt{\boldsymbol{a} \cdot \boldsymbol{a}} \tag{1.3}$$

---

[4] (0,0) in this case

[5] The dot product of two vectors computes a single scalar value, zero for orthogonal vectors, which can be interpreted as no similarity at all.

To construct a decision boundary in this example, one has to find a *line* such that it will separate one class of the training data set from the other one.

**Definition 1.5.** A set of points $(x_1, x_2) \in \mathbb{R}^2$ satisfying equation

$$w_1 x_1 + w_2 x_2 + b = 0 \tag{1.4}$$

is called a **line**. □

**Remark 1.2.** Note that we may express the equation (1.4) in the vector form as

$$\boldsymbol{w} \cdot \boldsymbol{x} + b = 0 \tag{1.5}$$

The beauty of the vector form (1.5) lies in the fact, that it is valid in general for any number of dimensions, i.e. for the data sets with more than just two explanatory variables.

In the three-dimensional settings, if we let $\boldsymbol{w}, \boldsymbol{x} \in \mathbb{R}^3$, that is $\boldsymbol{w} = (w_1, w_2, w_3)$ and $\boldsymbol{x} = (x_1, x_2, x_3)$, the equation (1.5) will not change and the subset of all points satisfying this equation is called a *plane*.

More generally, in $n$-dimensional space, we call the subset of all points satisfying (1.5) a *hyperplane*. A hyperplane is an affine subspace of dimension $n - 1$ which divides the space into two half spaces. [14, p. 10]

The notation $\boldsymbol{w}$ and $b$ was chosen with an agreement of the majority of the SVM literature; $\boldsymbol{w}$ standing for *weight vector* and $b$ for *bias*, both terms coming originally from the perceptron research, which has historically much common with the support vector machines [14, p. 10]. The statistics literature usually refers both as *parameters*.

## 1.2 Classification formalized

The classification problem can be formalized as follows:

- Let the dot product space $\mathbb{R}^n$ be the data universe with vectors $\boldsymbol{x} \in \mathbb{R}^n$ as objects
- Let $S$ be a sample set such that $S \subset \mathbb{R}^n$
- Let $f : \mathbb{R}^n \to \{+1, -1\}$ be the target function

- Let $D = \{(\boldsymbol{x}, y) \mid \boldsymbol{x} \in S \wedge y = f(\boldsymbol{x})\}$ be the training set

Find a function $\hat{f} : \mathbb{R}^n \to \{+1, -1\}$ using $D$ such that

$$\hat{f}(\boldsymbol{x}) \approx f(\boldsymbol{x}) \tag{1.6}$$

for all $\boldsymbol{x} \in \mathbb{R}^n$. Depending on $n$, we construct line, plane or hyperplane, that separates classes $+1$ and $-1$ as best as possible. The resulting *linear decision surface* is then used to assign new objects to the classes. [55, p. 1], [28, p. 50]

Vectors $\boldsymbol{x}$ are usually referred in the literature as *patterns*, *cases*, *instances* or *observations*; values $y$ are called *labels*, *targets* or *outputs*. [55, p. 1]

As already mentioned, we call this problem *binary classification problem*. It is caused by the fact, that the output domain $Y = \{+1, -1\}$ consists of two elements. In more general cases, for $Y = \{1, 2, \ldots, m\}$, the task is called m-class classification. For $Y \subseteq \mathbb{R}$ we speak about regression. [14, p. 11] While the support vector machines can also be used, with some simple modifications, for the m-class classification and for the regression, these modifications hardly have any use in the context of the credit scoring and are not a subject for this thesis.

Having the normal vector $\boldsymbol{w}$ and bias $b$ from (1.5), the task to classify new observation $\boldsymbol{a}$ is quite simple. Should the observation belong to the $+1$ class, the point will lie *above* the decision surface and the value of (1.5) will be positive. On the contrary, all observations belonging to the $-1$ class will lie *below* the decision surface and the value of (1.5) will be negative. The decision function $\hat{f}$ can be constructed as follows [28, p. 50-51]

$$\hat{f}(\boldsymbol{x}) = \begin{cases} +1 & \text{if } \boldsymbol{w} \cdot \boldsymbol{x} + b \geq 0 \\ -1 & \text{if } \boldsymbol{w} \cdot \boldsymbol{x} + b < 0 \end{cases} \tag{1.7}$$

That is, by defining the *sign function* for all $k \in \mathbb{R}$ as

$$sgn(k) = \begin{cases} +1 & \text{if } k \geq 0 \\ -1 & \text{if } k < 0 \end{cases} \tag{1.8}$$

the linear decision function for $\boldsymbol{w}, \boldsymbol{x} \in \mathbb{R}^n$, $b \in \mathbb{R}$ can be constructed as

$$\hat{f}(\boldsymbol{x}) = sgn(\boldsymbol{w} \cdot \boldsymbol{x} + b) \tag{1.9}$$

The situation is clearly demonstrated in Figure 2.

The key problem lies in the determination of the appropriate weight vector $\boldsymbol{w}$ and bias $b$. There are basically two approaches to this job.

The *generative learning* aims to learn the joint probability distribution over all variables within a problem domain (inputs $x$ and outputs $y$) and then uses the Bayes rule to calculate the conditional distribution $p(y|x)$ to make its prediction. Methods like naive Bayes classifier, Fisher's discriminant[6], hidden Markov models, mixtures of experts, sigmoidal belief networks, Bayesian networks and Markov random fields belong to this class of models. [38]

The *discriminative learning* does not attempt to model the underlying probability distributions of the variables and aims only to optimize a mapping from the inputs to the desired outputs, i.e. the model learns the conditional probability distribution $p(y|x)$ directly. These methods include perceptron, traditional neural networks, logistic regression and support vector machines. [38]

Discriminative models are almost always to be preferred in the classification problem. [47]

All these methods generate a linear decision boundary. That is, the boundary between the classes in the classification problem can be expressed in the form of the equation (1.5).

For example, the first and simplest linear classifier, the aforementioned Fisher's discriminant, attempts to find the hyperplane $(\boldsymbol{w}, b)$ on which the projection of the training data is maximally separated. That is to maximize the cost function

$$F = \frac{(m_+ - m_-)^2}{\sigma_+^2 + \sigma_-^2} \tag{1.10}$$

where $m_i$ is the mean and $\sigma_i$ is the standard deviation of the projection of the *positive* and *negative* observations. [14, p. 20] The decision boundary is then perpendicular to this hyperplane.

[6] Also called as Linear discriminant analysis

## 1.3 Logistic regression

Although its name would indicate otherwise, the logistic regression also belongs to the linear classifiers. Thanks to its very good performance and a long history of successful utilization, it is widely acknowledged that the logistic regression sets the standard against which other classifications methods are to be compared.[7] [22] This thesis will not be an exception and it also uses the logistic regression as an etalon.

Unlike the classical linear regression where the response variable can attain any real number, the logistic regression builds a linear model based on a transformed target variable that can attain only values in the interval between zero and one [58]. Thus the transformed target variable can be interpreted as a probability of belonging to the specific class. For binary classification problem, the model is especially simple, comprising just single linear function [33].

$$\log \frac{P(Y = +1 \mid X = \boldsymbol{x})}{P(Y = -1 \mid X = \boldsymbol{x})} = \boldsymbol{w} \cdot \boldsymbol{x} + b \tag{1.11}$$

The maximum likelihood estimation (MLE) method is usually used to find the parameters $\boldsymbol{w}$ and $b$, using the conditional likelihood of Y given the X. [33]

The left-hand side of (1.11) is called the *log-odds ratio*. Note, that in the situation when both probabilities, the one in the numerator as well as the one in the denominator, will be equal to 0.5, the log-odds ratio is zero. The equation will then simplify to (1.5). This is the equation describing the decision boundary in the logistic regression model; the set of points where the logistic regression is indecisive.

The difference in performance among the various linear classifiers arises from its different distribution assumptions or from the method used to the estimation of the parameters of the decision boundary $(\boldsymbol{w}, b)$. When we return to the geometrical interpretation of the input data, we can realize, that:

- If established on reasonable assumptions, the boundary lines generated by different linear classifiers should be quite similar. An example of two decision boundaries on our example data set is presented in the Figure 2.

[7] This opinion, however, has recently been challenged in [42]. Other, more powerful classifiers were proposed as a norm for the performance comparison in future research instead.

- There is an upper bound to the performance of ANY linear classifier. In every binary classification problem, there exists a single line that best separates the two classes of data.

The reason, why some classifiers systematically outperform others in applications using the real data sets, depends on 3 key points. How well does the classifier handle:

1. the linearly inseparable data.

2. the random noise in the data and the outliers.

3. the non-linear relationships in the data.

As will be shown in the next chapter, the strength of the Support vector machines when compared to other linear classifiers lies in the fact that (1) it can separate linearly inseparable data, (2) it is almost immune against outliers, (3) it can fit non-linear data thanks to the technique generally referred as Kernel trick. The logistic regression, for comparison, lacks the last ability.

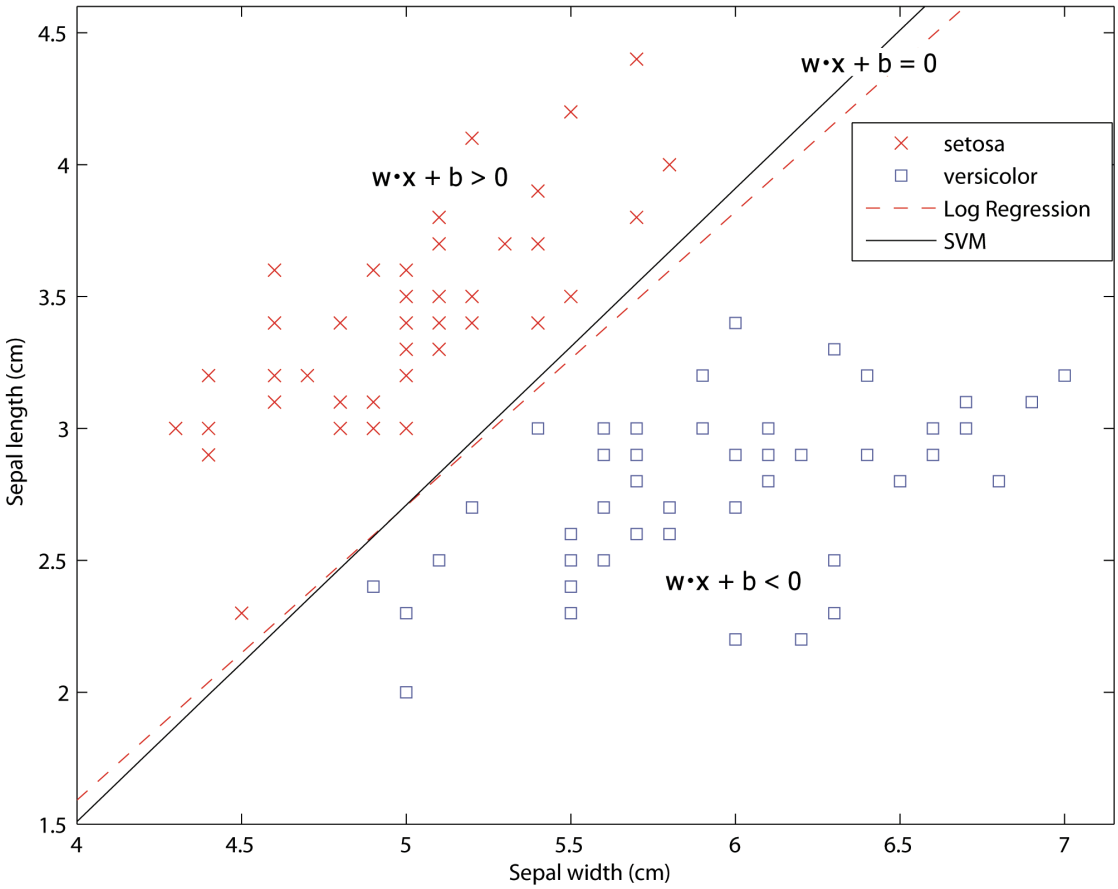


*Figure 2: The sample data set with two possible linear decision functions - one obtained with the logistic regression, the second one with the support vector machines. Although the lines are not identical, they both could be regarded as satisfactory models to classify the iris flowers. It can be proved from the Vapnik-Chervonenkis theory that the decision surface obtained with the support vector machines is the optimal linear classifier under the condition that "optimal" is defined in terms of structure risk minimization.*

# 2. Support vector machines

Support vector machines present a unique combination of several key concepts [57]:

1. Maximum margin hyperplane to find a linear classifier through optimization
2. Kernel trick to expand up from linear classifier to a non-linear one
3. Soft-margin to cope with the noise in the data.

This chapter describes the derivation of the support vector machines as the maximum margin hyperplane using the Lagrangian function to find a global extreme of the problem. First, hard-margin SVM applicable only to linearly separable data is derived. Primal and dual form of the Lagrangian optimization problem is formulated. The reason, why the dual representation poses significant advantage for further generalization is explained. Further, a non-linear generalization using kernel functions is described and the "soft-margin" variation of algorithm, allowing for noise, errors and misclassification, is finally derived.

## 2.1 Maximum margin hyperplane

Let us start with the assumption, that the training data set is linearly separable. We are looking for the hyperplane parameters $(\boldsymbol{w}, b)$, so that the distance between the hyperplane and the observations is maximized. We shall call the Euclidean distance between the point $\boldsymbol{x}_i$ and the hyperplane as *geometric margin*.

**Definition 2.1.** The **geometric margin** of an individual observation $\boldsymbol{x}_i$ is given by the equation

$$\delta_i = \frac{y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b)}{\|\boldsymbol{w}\|} \tag{2.1}$$

□

Note, that the equation (2.1) really does correspond to the perpendicular distance of the point,[8] only the multiplier $y_i$ seems to be superfluous. But recall that $y_i$ assigns either +1 or -1. Its only purpose in the equation is to assure the distance to be a non-negative number. [44]

[8] See [48, p. 161] for details and proof.

**Definition 2.2.** Given the training set D, the **geometric margin of a hyperplane** $(\boldsymbol{w}, b)$ with respect to D is the smallest of the geometric margins on the individual training observations: [44]

$$\delta = \min_{i=1...m} \delta_i \tag{2.2}$$

□

If we are looking for the optimal separating hyperplane, our goal should be to maximize the geometric margin of the training - i.e. to place the hyperplane such that its distance from the nearest points will be maximized. However, this is hard to do directly as such an optimization problem is a non-convex one. [44] Therefore, the problems needs to be reformulated before we can move forward.

**Definition 2.3.** The **functional margin** of an individual observation $\boldsymbol{x}_i$ is given by

$$\hat{\delta}_i = y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \tag{2.3}$$

□

**Definition 2.4.** Given a training set D, the **functional margin of** $(\boldsymbol{w}, b)$ with respect to D is the smallest of the functional margins on the individual training observations. [44]

$$\hat{\delta} = \min_{i=1...m} \hat{\delta}_i \tag{2.4}$$

□

Note, that there is a relationship between geometric margin and the functional margin:

$$\delta = \frac{\hat{\delta}}{\|\boldsymbol{w}\|} \tag{2.5}$$

Also note, that the geometric margin is scaling invariant - replacing the parameters $(\boldsymbol{w}, b)$ with $(2\boldsymbol{w}, 2b)$, for instance, will not change the results. The opposite is true for the functional margin. Therefore we may impose an arbitrary scaling constraint on $\boldsymbol{w}$ conveniently to suit our needs. We will set the scaling so that the functional margin is 1 and maximize the geometric margin under this constraint. [44]

$$\hat{\delta} = 1, \tag{2.6}$$

When we want to maximize the geometric margin, from (2.5) it follows that we need to maximize the term $\frac{1}{\|\boldsymbol{w}\|}$. Such a task is equal to the minimization of $\|\boldsymbol{w}\|$ in the denominator. Yet this task is still quite a difficult one to solve, as by (1.1) it would involve the terms in a square root.

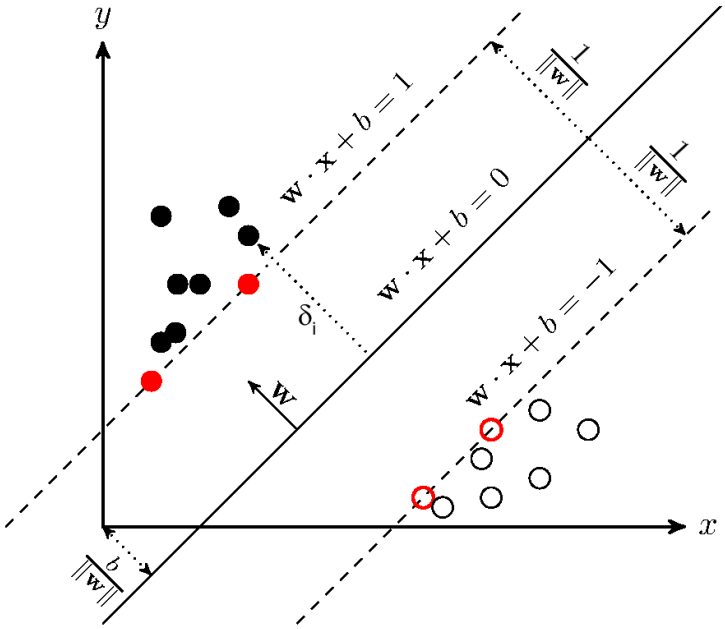


*Figure 3: Geometric margin and maximum margin hyperplane (line in 2-D). The solid line separates two classes (filled and empty circles) and is fully defined by the vector of weights* $\boldsymbol{w}$ *which is perpendicular to the line and bias* $b$*, which determines the position of the line with respect to the origin. The distance between each point and the line is called geometric margin* $\delta_i$ *and defined in* (2.1)*. There will always be some points that are closest to the line (marked red). Using the appropriate scaling* (2.6)*, their distance can be set to* $\frac{1}{\|\boldsymbol{w}\|}$*. In further steps, the goal is to maximize this distance. Credits: Yifan Peng, 2013 [49].*

Nevertheless, it is possible to replace this term by $\frac{1}{2}\|w\|^2$ without changing the optimal solution. [9] The multiplier $\frac{1}{2}$ is added for the mathematical convenience in later steps and again, does not affect the position of the extreme.

Finally, applying (1.3), the objective function to be minimized is

$$\Phi(\boldsymbol{w}, b) = \frac{1}{2}\, \boldsymbol{w} \cdot \boldsymbol{w} \tag{2.7}$$

It is necessary to ensure that the equation (2.4) holds when the objective function is minimized. The functional margin of each individual observation must be equal

[9] Optimization over the positive values $\|\boldsymbol{w}\|$ is invariant under the transformation with the square function. In general, $x^2$ is a monotonic function for $x \geq 0$. Therefore optimizing over $x^2$ is same as optimizing over $x$. [28, p. 81]

to or larger than the functional margin of the whole dataset.

$$y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq \hat{\delta}, \ \forall i \tag{2.8}$$

By substituting (2.6), we have

$$y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq 1, \ \forall i \tag{2.9}$$

This inequality poses a constraint to the optimization problem (2.7). [44]

The solution of this problem will be a linear decision surface with the maximum possible margin (given the training dataset). The objective function is clearly a convex function which implies that we are able to find its global maximum. The entire optimization problem consists of a quadratic objective function (2.7) and linear constraints (2.9) and is therefore solvable directly by the means of the *quadratic programming*. [28, p. 82]

## 2.2 Lagrangian methods for optimization

Since the means of the quadratic programming are usually computationally inefficient when dealing with large data sets[10], we will continue solving this optimization problem using the Lagrangian methods.[11]

Let us start more generally and then apply the mathematical theory to the problem of finding the maximum-margin hyperplane.

---

[10] thousands or tens of thousands of observations

[11] This whole section of my work is based mostly on the online videolecture of professor Andrew Ng from the Stanford University [45] who (along with many other lecturers from the most prominent universities) made his knowledge freely available under the Creative Commons license to people from all over the world. For that I express him (them) my deepest gratitude.

### 2.2.1 Primal problem

Consider the optimization problem

$$\begin{aligned} \min_{\boldsymbol{w}} \quad & f(\boldsymbol{w}) \\ s.t. \quad & g_i(\boldsymbol{w}) \leq 0 \\ & h_i(\boldsymbol{w}) = 0 \end{aligned} \tag{2.10}$$

The minimization is constrained by an arbitrary number of equality and inequality constraints.[12] One method to solve such a problem consists in formulating a derivative problem, that has the same solution as the original one. Define the **generalized Lagrangian** function of (2.10) as

$$\mathcal{L}(\boldsymbol{w}, \alpha, \beta) = f(\boldsymbol{w}) + \sum_i \alpha_i g_i(\boldsymbol{w}) + \sum_i \beta_i h_i(\boldsymbol{w}) \tag{2.11}$$

Further define

$$\theta_P(\boldsymbol{w}) = \max_{\alpha,\beta;\alpha_i \geq 0} \mathcal{L}(\boldsymbol{w}, \alpha, \beta) \tag{2.12}$$

This allows as to formulate the so called *primal optimization problem*, denoted by subscript $P$ on that account:

$$p^* = \min_{\boldsymbol{w}} \theta_P(\boldsymbol{w}) = \min_{\boldsymbol{w}} \max_{\alpha,\beta;\alpha_i \geq 0} \mathcal{L}(\boldsymbol{w}, \alpha, \beta) \tag{2.13}$$

Note that the function $\theta_P$ value assumes plus infinity, in case any constraint from (2.10) does not hold. If $g_i(\boldsymbol{w}) > 0$ for any $i$, (2.12) can be maximized by simply setting $\alpha = +\infty$. If on the other hand $h_i(\boldsymbol{w}) \neq 0$, (2.12) is maximized by setting $\beta$ to plus or minus infinity (depending on the sign of the function $h_i$).

If all the constraints hold, $\theta_P$ takes the same value as the objective function in the original problem, $f(\boldsymbol{w})$. Provided both constraints hold, the third equation term in (2.11) is always zero and the second term is also maximized when it is set to zero by appropriate setting of weights $\alpha_i$. Thus we have

---

[12]Note, that (2.10) embodies a very broad class of problems. For example, any maximization problem can be transformed to the minimization, since $\max x = \min -x$ or $\max x = \min \frac{1}{x}$, if defined. Also any constrained in the form $g(x) \geq 0$ can be converted to the form of (2.10) multiplying by (-1) and every constraint in the form $h(x) = b$ can be transformed by subtracting $b$ to the desirable form.

$$\theta_P(\boldsymbol{w}) = \begin{cases} f(\boldsymbol{w}) & \text{if constraints are satisfied} \\ +\infty & \text{otherwise} \end{cases} \tag{2.14}$$

It is obvious that (2.13) is, indeed, equal to the original problem (2.10).

### 2.2.2 Dual problem

We could solve the primal problem, but when applied to our task, the properties of the algorithm would not bring much benefits. Instead, for the sake of the forthcoming solutions, let us define a different optimization problem by switching the order of minimization and maximization.

$$\theta_D(\alpha, \beta) = \min_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{w}, \alpha, \beta) \tag{2.15}$$

The subscript $D$ stands for *dual* and it will be be used to formulate the so called *dual optimization problem*:

$$d^* = \max_{\alpha,\beta;\alpha_i \geq 0} \min_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{w}, \alpha, \beta) = \max_{\alpha,\beta} \theta_D \tag{2.16}$$

The value of the primal problem $p^*$ and the dual problem $d^*$ is not necessarily equal. There is a generally valid relationship between maximum of minimization and minimum of maximization

$$d^* \leq p^* \tag{2.17}$$

The difference between primal and dual solution, $p^* - d^*$ is called the *duality gap*. As follows from (2.17), the duality gap is always greater than or equal to zero.

Certain conditions have been formulated that (if met) guarantee the duality gap to be zero, i.e. under such conditions, the solution of the dual problem equals to the solution of the primal one.

**Definition 2.5.** Suppose $f$ and all functions $g_i$ are convex and all $h_i$ are affine. Suppose that the constraints are strictly feasible (i.e. there exists some $\boldsymbol{w}$ so that $g_i(\boldsymbol{w}) < 0$ for all $i$). Then there exist $\boldsymbol{w}^*$, $\alpha^*$, $\beta^*$ so that $\boldsymbol{w}^*$ is the solution to the primal problem, $\alpha^*$, $\beta^*$ is the solution to the dual problem, $p^* = d^* = \mathcal{L}(\boldsymbol{w}^*, \alpha^*, \beta^*)$ and $\boldsymbol{w}^*, \alpha^*, \beta^*$ satisfy following **Karush-Kuhn-Tucker (KKT) conditions** [44]

$$\frac{\partial \mathcal{L}}{\partial w_i} = 0, \quad i = 1, \ldots, n \tag{2.18}$$

$$\frac{\partial \mathcal{L}}{\partial \beta_i} = 0, \quad i = 1, \ldots, n \tag{2.19}$$

$$\alpha_i^* g_i(\boldsymbol{w}^*) = 0, \quad i = 1, \ldots, n \tag{2.20}$$

$$g_i(\boldsymbol{w}^*) \leq 0, \quad i = 1, \ldots, n \tag{2.21}$$

$$\alpha_i^* \geq 0, \quad i = 1, \ldots, n \tag{2.22}$$

□

Remark: For its specific importance, KKT condition (2.20) is often referred as *complementary condition*.

### 2.2.3 Application to maximum margin hyperplane

Let us return to our original task defined by equations (2.7), (2.9). The constraint can be rewritten in order to match the form of (2.10)[13]

$$g_i(\boldsymbol{w}) = -y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) + 1 \leq 0 \tag{2.23}$$

The Lagrangian of the objective function (2.7) subject to (2.9) will be, according to (2.11)

$$\mathcal{L}(\boldsymbol{w}, \boldsymbol{\alpha}, b) = \frac{1}{2}\boldsymbol{w} \cdot \boldsymbol{w} - \sum_{i=1}^{m} \alpha_i \left(y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) - 1\right) \tag{2.24}$$

The dual can be found quite easily. To do this, we need to first minimize $\mathcal{L}$. [44] By (2.18) we take the partial derivatives of $\mathcal{L}$ with respect to $\boldsymbol{w}$ and $b$

$$\boldsymbol{w} = \sum_{i=1}^{m} \alpha_i y_i x_i \tag{2.25}$$

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \tag{2.26}$$

Substituting these equations into (2.24) and simplifying the result

$$\mathcal{L}(\boldsymbol{w}, \boldsymbol{\alpha}, b) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \ \boldsymbol{x}_i \cdot \boldsymbol{x}_j \tag{2.27}$$

[13]Note, that there are no equality constraints in this task, so there will be no $h_i$ terms.

This equation was obtained by minimizing $\mathcal{L}$ with respect to $\boldsymbol{w}$ and $b$. [44] According to (2.16), this expression needs to be maximized now. Putting it together with (2.22) and (2.26), the dual optimization problem is

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \left( \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \ \boldsymbol{x}_i \cdot \boldsymbol{x}_j \right) \\ s.t. \quad & \sum_{i=1}^{m} \alpha_i y_i = 0 \\ & \alpha_i \geq 0, \ i = 1, \ldots, l \end{aligned} \tag{2.28}$$

It can be shown, that KKT conditions are met and that (2.28) really presents a solution to the original problem. [28]

One interesting property should be pointed out at the moment. The constraint (2.9) actually enforces the fact that the functional margin of each observation should be equal to or greater than the functional margin of the dataset (which we set by (2.6) to one).

It will be exactly one only for the points closest to the separating hyperplane. All other points will lie further and therefore their functional margin will be higher than one. By the *complementary condition* (2.20) the corresponding Lagrangian multipliers $\alpha_i$ for such points must be equal to zero. [14, p. 97]

Therefore the solution of the problem, the weight vector $\boldsymbol{w}$ in (2.25), depends only on a few points with non-zero $\alpha_i$ that lie closest to the separating hyperplane. These points are called **support vectors**. [14, p. 97]

The solution of the dual problem (2.28) leads to an optimal vector $\boldsymbol{\alpha}^*$, which can be immediately used to calculate the weight vector of the separating hyperplane $\boldsymbol{w}$ using 2.25. Since most of the $\alpha_i$ coefficients will be zero, the weight vector will be a linear combination of just a few support vectors.

Having calculated the optimal $\boldsymbol{w}^*$, the bias $b^*$ must be found making use of the primal constraints, because the value $b$ does not appear in the dual problem: [14, p. 96]

$$b^* = -\frac{\max_{y_i=-1} \boldsymbol{w}^* \cdot x_i + \min_{y_i=+1} \boldsymbol{w}^* \cdot x_i}{2} \tag{2.29}$$

From (1.9) this leads to a decision function that can be used to classify new

out-of-sample observation $\boldsymbol{x}$:

$$\hat{f}(\boldsymbol{x}) = sgn\left(\sum_{i=1}^{m} \alpha_i^* y_i \boldsymbol{x}_i \cdot \boldsymbol{x} - b^*\right) \tag{2.30}$$

As stated above, this classifier is determined completely only by the support vectors. Because of this property, it is called a **support vector machine**. To be more precise, a *linear support vector machine*, since it is based on a linear decision surface. [28, p. 102] It is able to classify linearly separable data only. On the other hand, unlike some other classifiers, it is guaranteed to converge and find the hyperplane with the maximum margin. Since the optimization problem is a convex and KKT conditions are met, there is always a unique solution and it is a global extreme.[14]

There is one remarkable property in the decision function (2.30) that is worthy of our special attention. The observations only appear there as dot products between the input vectors. This opens a door to the use of a special technique that researcher usually refer as "kernel trick". Owing to it, the linear support vector machine derived in this section can be extended and generalized to fit almost any non-linear decision surface.

## 2.3 Kernels and kernel trick

The problem with the real life data sets is that the relationship among the input variables are rarely simple and linear. In most cases some non-linear decision surface would be necessary to correctly separate the observations into two groups. This necessity, on the other hand, noticeably increases the complexity and computational difficulty of such a task.

Instead, the support vector machines implement another conceptual idea: the input vectors are mapped from an original *input space* into a high-dimensional *feature space* through some non-linear mapping function chosen a priori. The linear decision surface is then constructed in this feature space. [12]

The underlying justification can be found in Cover's theorem [13], which may be qualitatively expressed as:

[14]Even some advanced classification algorithms, like neural networks, must face the problem that they may get "stuck" in the local extreme and find only a suboptimal solution.

> A complex pattern-classification problem, cast in a high-dimensional space nonlinearly, is more likely to be linearly separable than in a low-dimensional space, provided that the space is not densely populated. [34, p. 231]

Because now we want to operate in the feature space rather than in the original input space, we can slightly modify the original equation (2.28) and replace the plain dot products $\boldsymbol{x}_i \cdot \boldsymbol{x}_j$ with $\phi(\boldsymbol{x}_i) \cdot \phi(\boldsymbol{x}_j)$. By $\phi(\cdot)$ we denote some appropriately chosen mapping function.

Consider that the transformation $\phi(\boldsymbol{x})$ may be very complex, complicated. The results could belong to high dimensional, in some cases even infinitely dimensional spaces. Such transformations would be computationally very expensive or even impossible to do.

The key idea of this concept is contained in the fact, that to find the maximum margin hyperplane in the feature space, one actually need not to calculate the transformed vectors $\phi(\boldsymbol{x_i}), \phi(\boldsymbol{x_j})$ explicitly, since it is present only as a dot product in the training algorithm.

It would be very wasteful to spend a precious computational time to perform these complex transformations only to aggregate them immediately in a dot product, one single number.

It appears that for our needs it is sufficient to use a so called *kernel function* which is defined as a dot product of the two transformed vectors and which may be calculated very inexpensively. [44]

**Definition 2.6.** Given an appropriate mapping $\phi : \mathbb{R}^n \to \mathbb{R}^m$ with $m \geq n$, the functions of the form

$$K(\boldsymbol{x}, \boldsymbol{z}) = \phi(\boldsymbol{x}) \cdot \phi(\boldsymbol{z}) \tag{2.31}$$

where $\boldsymbol{x}, \boldsymbol{z} \in \mathbb{R}^n$ are called *kernel functions* or just *kernels*. [28, p. 107] □

The process of mapping the data from the original input space to the feature space is called *kernel trick*. [28, p. 103] The kernel trick is rather universal approach, its application is not solely limited to the support vector machines. In general, any classifier that depends merely on the dot products of the input data may capitalize on this technique.

Let us rephrase the support vector machines using the kernel functions. The

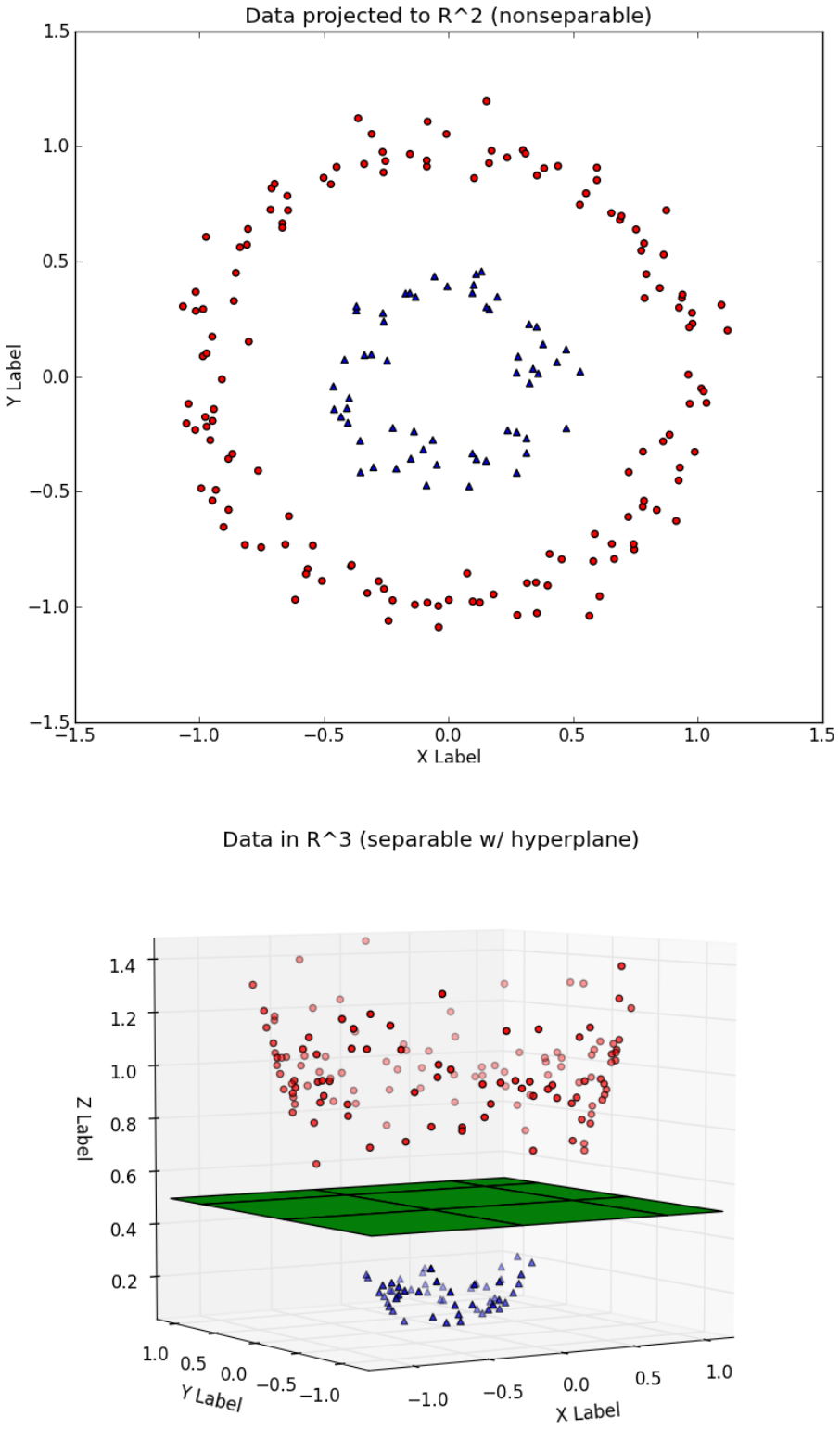


*Figure 4: An example of the data set, that is not linearly separable in the original space* $\mathbb{R}^2$ *(Top). The very same data set can be however separated linearly in the transformed space obtained by the transformation:* $[x_1, x_2] = [x_1, x_2, x_1^2 + x_2^2]$ *(Bottom). Credits: Eric Kim, 2013 [39].*

training algorithm (2.28) can be now expressed as

$$\begin{aligned} \boldsymbol{\alpha}^* = \underset{\boldsymbol{\alpha}}{\operatorname{argmax}} \quad & \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ s.t. \quad & \sum_{i=1}^{m} \alpha_i y_i = 0 \\ & \alpha_i \geq 0, \ i = 1, \ldots, l \end{aligned} \tag{2.32}$$

while the decision surface (2.30) in terms of kernel function is defined as

$$\hat{f}(\boldsymbol{x}) = sgn \left( \sum_{i=1}^{m} \alpha_i^* y_i K(\boldsymbol{x}_i, \boldsymbol{x}) - b^* \right) \tag{2.33}$$

### 2.3.1 Linear kernel

There are several well known kernel functions that are commonly used in connection with support vector machines.

The first, rather trivial, example is a *linear kernel*. It is defined simply as

$$K(\boldsymbol{x}, \boldsymbol{z}) = \phi(\boldsymbol{x}) \cdot \phi(\boldsymbol{z}) = \boldsymbol{x} \cdot \boldsymbol{z} \tag{2.34}$$

In this case, the feature space and the input space are same and we get back to the solution derived in the previous Section 2.2.3, where we discussed linear support vector machines.

I mention the linear kernel here just to emphasize the fact, that the substitution with the kernel function does not change the original algorithm entirely. It merely generalizes it.

### 2.3.2 Polynomial kernel

The *polynomial kernels* are another example of frequently used kernel functions. When Vapnik and Cortes in their essential paper [12] achieved a great shift in the optical character recognition of handwritten numbers, it was the polynomial kernel that showed its strength for this particular task.

The homogeneous polynomial kernel of degree $d$ ($d \geq 2$) is defined as

$$K(\boldsymbol{x}, \boldsymbol{z}) = (\boldsymbol{x} \cdot \boldsymbol{z})^d \tag{2.35}$$

The more general non-homogeneous polynomial kernel ($d \geq 2, c > 0$) is

$$K(\boldsymbol{x}, \boldsymbol{z}) = (\boldsymbol{x} \cdot \boldsymbol{z} + c)^d \tag{2.36}$$

Note, that the latter maps the original input space to the feature space of the dimensionality $\binom{n+d}{d}$, where $d$ is the degree of the kernel and $n$ is the dimension of the original input space. However, despite working in this $O(n^d)$-dimensional space, computation of $K(\boldsymbol{x}, \boldsymbol{z})$ consumes only $O(n)$ time. [44]

### 2.3.3 Gaussian kernel (Radial basis function)

The *gaussian kernel* is probably the mostly used kernel function and is defined as

$$K(\boldsymbol{x}, \boldsymbol{z}) = exp\ \left\{ -\frac{\|\boldsymbol{x} - \boldsymbol{z}\|^2}{2\sigma^2} \right\} \tag{2.37}$$

It maps the input space to the infinitely dimensional feature space. Thanks to this property, it is very flexible and is able to fit a tremendous variety of shapes of the decision border. The gaussian kernel is also often referred as radial basis function (RBF). In some instances is parametrized in a slightly different manner:

$$K(\boldsymbol{x}, \boldsymbol{z}) = exp\ \left\{ -\gamma \|\boldsymbol{x} - \boldsymbol{z}\|^2 \right\} \tag{2.38}$$

### 2.3.4 Other kernels

There are many other well researched kernel functions. An extensive, although probably not exhaustive, list of other commonly used kernel functions is available in [17]. Most of them are, however, used only for research purposes or under the special circumstances and have no use in the credit scoring models.

For any function K to be a valid kernel, a necessary and sufficient condition has been derived. This condition is usually referred as *Mercer's theorem*, which is formulated in a slightly simplified form below [44].

**Definition 2.7.** Let a finite set of $m$ points be given $\{x_1, \ldots, x_m\}$ and let a square $m$-by-$m$ matrix $K$ be defined so that $(i, j)$ entry is given by $K_{ij} = K(x_i, x_j)$. Then such a matrix is called **Kernel matrix**. □

**Theorem 2.1** (Mercer's theorem)**.** Let $K : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ be given. Then for $K$ to be a valid kernel, it is necessary and sufficient that for any $\{x_1, \dots x_m\}, m < \infty$, the corresponding kernel matrix is symmetric positive semi-definite.

There are no general rules concerning the selection of a suitable kernel. It highly depends on the character of the task being solved, the data set and the relationships among the variables play a major role, too. In case of simple jobs with plain linear relationships, the linear kernel may be sufficient and other kernels will not bring any improvement.

In other applications, where the data are complex and interdependent, more advanced kernels may bring a major leap in the classification performance. This may be the case in the aforementioned optical character recognition (OCR), face recognition, natural language processing or genetics.

The credit scoring lies kind of in the middle of these extremes. Some papers suggest, that the consumer credit scoring data tend to be linear or just mildly non-linear. Therefore, the linear support vector machines predominate in the literature, often accompanied with the Gaussian kernel to seek for the non-linearities in the data (see [42] for comprehensive list of papers). The polynomial kernels are rarely represented in the credit scoring literature (see [5], [56] for examples) as they seem to give the mixed results.

## 2.4 Soft margin classifiers

Until now we have assumed that our training data are linearly separable - either directly in the input space or in the feature space. In other words we have supposed only perfect data sets. In reality the data sets are often far from perfect. The real data are noisy, contain misclassified observations, random errors etc.

We need to generalize the algorithm to be able to deal with the non-separable data. Surprisingly enough, the basic idea of the generalization is very simple. We just need to allow some observations to end up on the "wrong" side of the separating hyperplane.

The required property is accomplished by introducing a *slack variable* $\xi$ and rephrasing the original constraint (2.9)

$$y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq 1 - \xi_i, \ i = 1, \dots, l \tag{2.39}$$

where $\xi_i \geq 0$. This way we allow the functional margin of some observations to be less than 1.

The slack variable works like a corrective equation term here. For the observations located on the correct side of the hyperplane, the respective $\xi_i = 0$ and the (2.39) is then same as (2.9). For noisy or misclassified observations appearing on the wrong side of the hyperplane, the respective $\xi_i > 0$, as shown in Figure 5.

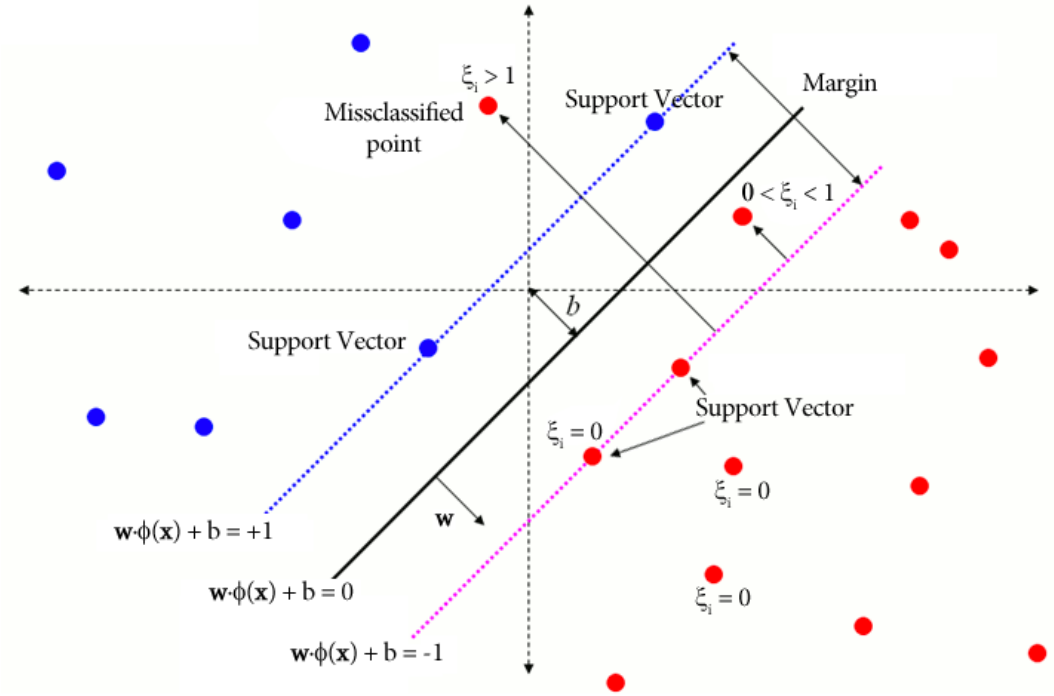


*Figure 5: The purpose of the slack variable explained through the simple sketch. The respective slack variable $\xi_i$ is zero if the observation is located on the correct side of the hyperplane and nothing is changed. The $\xi_i$ is greater than zero if its distance from the separating hyperplane is lower than the distance of support vectors. Credits: Stephen Cronin, 2010, slightly modified. [15].*

It is necessary that with each misclassification we pay some price, since in general we want to maximize the margin while simultaneously minimize the adverse effect of the wrong observations. Otherwise the trivial solutions where all data are marked as wrong would always prevail. This is why we have to expand the original objective function by the penalty term

$$\min_{\boldsymbol{w},\boldsymbol{\xi},b} \left( \frac{1}{2}\boldsymbol{w} \cdot \boldsymbol{w} + C \sum_{i=1}^{m} \xi_i \right) \tag{2.40}$$

where $C > 0$ is the *cost parameter*, which determines the importance of wrong observations. Small values of C lead to more benevolent solutions while the larger values of C would lead to solutions similar to the hard margin classifier from the previous sections.

This time due to the limited space I will not go again through the whole process of deriving the dual of this problem (for the step by step solution see [28, p. 118-121]). It turns out, that the dual of the soft margin classifier simplifies to a very convenient form:

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \left( \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \right) \\ s.t. \quad & \sum_{i=1}^{m} \alpha_i y_i = 0 \\ & C \geq \alpha_i \geq 0, \qquad\qquad i = 1, \ldots, l \end{aligned} \tag{2.41}$$

Note that the newly inserted cost term does not appear in the objective function and poses only a restriction for the values of the individual $\alpha$ coefficients.

Thanks to this property and combined with the kernel trick, the soft margin support vector machines are very easily implemented and solved efficiently on the computers without requiring a tremendous computational power.

The results, again, will be the decision function (2.30). This function produces results on the simple yes/no scale. In credit scoring we often need more than that: we need to quantify the quality of the debtor as a real-valued number which is then called *score* in this context. This may come handy in tasks like calculating the *area under curve* or finding the optimal *cut-off value*, which both will be discussed later in the text.

For this purpose, the functional margin of the observation can be used. Thus, the score $s$ of the observation $\boldsymbol{x}$ will be simply defined as

$$s(\boldsymbol{x}) = \sum_{i=1}^{m} \alpha_i^* y_i \boldsymbol{x}_i \cdot \boldsymbol{x} - b^* \tag{2.42}$$

where $\boldsymbol{\alpha}^*$, $b^*$ are the parameters of the optimal hyperplane calculated by the support vector machines, $\boldsymbol{x}_i$, $y_i$ are the individual observations from the training set and $m$ is the number of observations in the training data set.

## 2.5 Algorithm implementation

The simplest approach to solve a convex optimization problem numerically on computers is the gradient ascent, sometimes known as the steepest ascent algorithm. The method starts at some arbitrary initial vector (e.g. rough estimate of the possible solution) and proceeds iteratively, updating the vector in each step in the direction of the gradient of the objective function at the point of the current solution estimate. [14, p. 129]

But using the gradient methods to solve (2.41) brings problems which are hard to be circumvented. The main trouble is caused by the fact, that the constraint $\sum_{i=1}^{m} \alpha_i y_i = 0$ must hold on every iteration. From this constraint it follows, that the elements of vector of solutions, $\boldsymbol{\alpha}$ are interconnected and cannot be updated independently. It also implies, that the smallest number of multipliers that can be optimised at each step is 2. [14, p. 137]

This key fact is used by the algorithm Sequential Minimal Optimisation (SMO) which, in the course of time, was established as the optimal way to solve the problem (2.41). At each iteration, the SMO algorithm chooses exactly 2 elements of the optimised vector, $\alpha_i$ and $\alpha_j$, finds the optimal values for those elements given that all others are fixed and updates the vector $\boldsymbol{\alpha}$ accordingly. [14, p. 137]

In each step, the vector $\alpha_i$ and $\alpha_j$ are chosen by the heuristics described in [50] by Platt, who first introduced the SMO algorithm; the optimisation itself is however solved analytically in every iteration. The introduction of SMO removed the main barrier for support vector machines to be widely adopted by the researchers, as it sped up the training process at least 30 times against the methods used before. [19]

Since then, the SMO efficiency was further improved and it quickly became a standard for the support vector machines training.

Because of the fact, that the actual process of implementation of the training algorithm is rather tedious, prone to errors and represents more just a technicality rather than actual research value, the standardized packages and libraries to support the SVM training and evaluation were created.

Among them, LIBSVM [9] is the most prominent one, with the longest history, continuous updates, associated research paper and data sets and widest support for different development environments (IDEs). Other libraries, like SVM*light*[15], mySVM[16] or the specialised toolbox for R[17], Torch[18] or Weka[19] do exist.

Matlab, which was used for the practical part of my thesis, has its own support for SVM. However, its performance and the possibilities it offers are not astonishing. For that reason, I decided to use the LIBSVM library as an extension for the standard Matlab functions.

[15]http://svmlight.joachims.org/

[16]http://www-ai.cs.uni-dortmund.de/SOFTWARE/MYSVM/index.html

[17]SVM in R, http://cran.r-project.org/src/contrib/Descriptions/e1071.html

[18]SVMTorch, http://www.idiap.ch/learning/SVMTorch.html

[19]http://www.cs.waikato.ac.nz/ml/weka/

# 3. Model building

In this chapter, some practical issues that need to be addressed when one builds a credit scoring model (or any other classification model) are discussed. While some of these issues are rather general and not solely limited to the support vector machines, the text was adjusted to the fact, that SVM will be my primary training algorithm in the practical part of this thesis.

## 3.1 Data preprocessing

### 3.1.1 Categorical variables

Support vector machines assume that each input observation is represented as a vector of real numbers. In some instances, and this is true especially for the credit risk applications, many features are not real numbers. They are either ordinal or categorical variables (e.g. education, gender, home ownership, ...). Hence for the use of SVM algorithm, these variables need to be transformed into numerical ones.

One common approach is to express $r$-category variable as a zeros-one vector of length $r$. Thus for example gender male, female can be represented as (1,0) and (0,1), respectively. The increased number of attributes should not negatively affect the performance of the SVM. [35]

In fact, there is one dimension redundant in this approach. Merely $(r-1)$-dimensional vector of zeros and ones is sufficient to encode categorical variables with $r$ values. This is not an issue with SVM as the learning algorithm is able to ignore the excess dimension with no negative effect on the resulting model and it may be the reason, why most of the SVM literature recommends the aforementioned technique.

The extra dimension does, however, cripple the Logistic regression usability when full model with constant is sought. In such cases, the coefficient matrix is linearly dependent and the problem with multicollinearity arises, thus making the coefficient search task unsolvable. For these reasons, I use only $(r-1)$-dimensional encoding for each categorical variable, so that it is solvable by SVM as well as Logistic regression algorithms.

The conversion of the categorical variables to the real-valued numerical variables poses another approach to solve this problem. One commonly used technique takes advantage of calculation the Weight of Evidence (WoE). For each value of one categorical variable $c$, the WoE is defined as [59, p. 56]:

$$WoE(c) = ln\ P\left(c \mid nondefault\right) - ln\ P\left(c \mid default\right) \tag{3.1}$$

Unlike previous technique, which leads to the increase of the data set dimensionality, replacing the categorical variables with WoE keeps the number of dimensions intact. Therefore, it is advantageous particularly in cases, when the dimensionality of the problem poses a potential problem (e.g. due to high computational costs or algorithm weaknesses). While this is not in theory a case of the SVM algorithm, it can still be beneficial to work only with real-valued numerical inputs.

### 3.1.2 Scaling and standardizing

Another very important preprocessing step is the data scaling. It prevents the attributes in greater numeric ranges to dominate those in smaller ranges. It has also positive effect on the convergence of SMO algorithm that is most common computer implementation of Support Vector Machines at the moment. [35]

Most often, the numerical variables are scaled linearly to the range [-1,+1] or [0,1]. Another approach is standardizing, i.e. subtracting the mean and dividing by the standard deviation. A very detailed description on scaling and standardizing can be found in [53]. It is important to use the same scaling factors on the training as well as on the testing set. [35]

### 3.1.3 Grid search

Depending on the chosen kernel, there are always some free parameters to be set or chosen. In the case of the most used non-linear kernel, the radial basis function (2.38), there are two parameters: $C$ (the soft-margin cost parameter) and $\gamma$ (the shape parameter). The optimal $(C, \gamma)$ is not known and therefore there must be some parameters search procedure during the model selection. In the course of the time, a procedure called "grid search" became established as a standard approach to this task.

The guide from the authors of the LIBSVM library [35] recommends using exponentially growing sequence of parameters ($C = 2^{-5}, 2^{-3}, \ldots, 2^{15}$; $\gamma = 2^{-15}, 2^{-13}, \ldots, 2^{3}$) in combination with the k-fold cross validation to find the optimal pair of the parameters. For bigger data sets, rough scale may be used for finding the best performing area of parameters on the grid at first. Then the finer grid resolution is applied to the best performing region. After finding the optimal ($C$, $\gamma$), the model should be retrained using the whole data set.

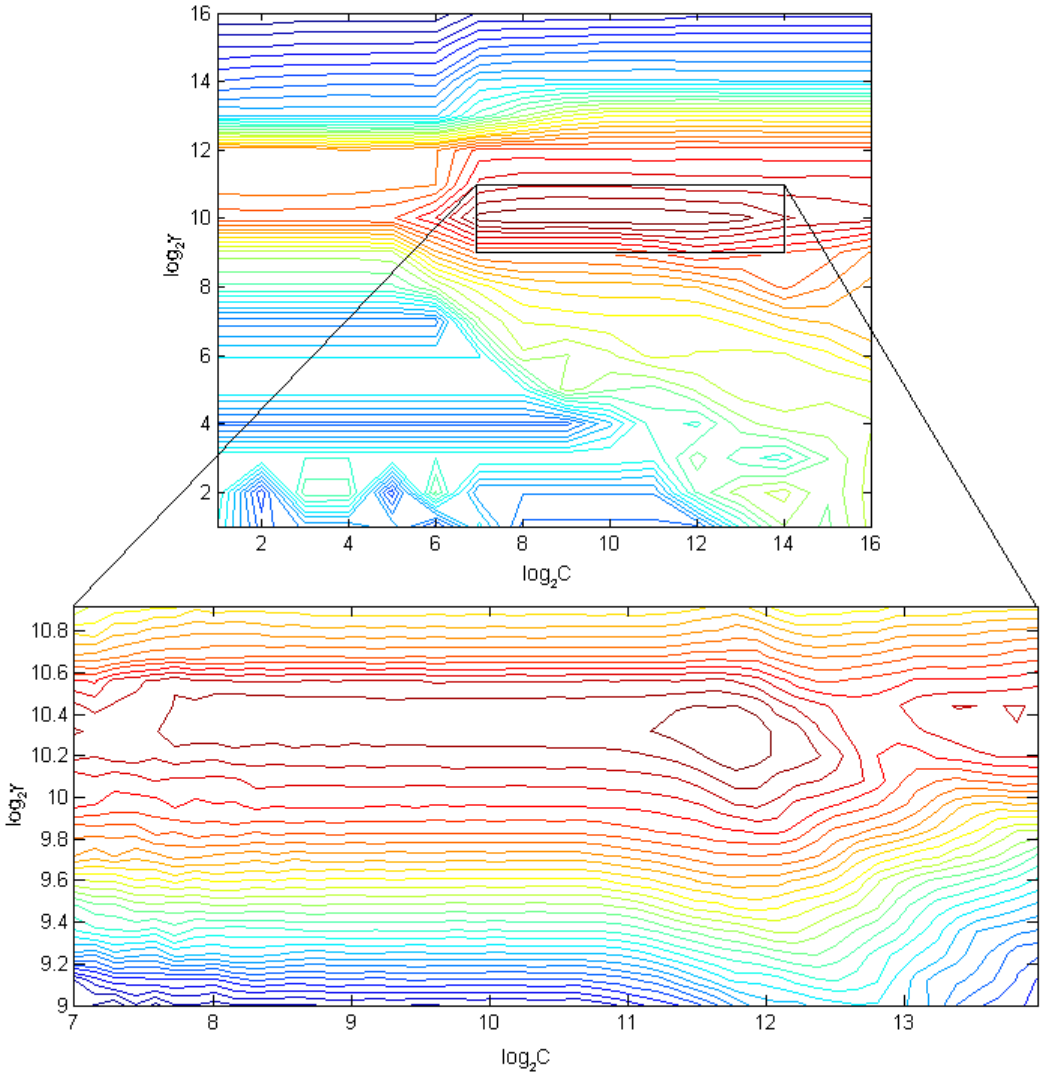


*Figure 6: The example of the two-step grid search process is shown on this contour graph. Darker contours in red represent higher performance of the support vector machine on the testing data set. First, the region with highest performance is identified on the rough grid scale. Then the finer resolution is used to find the optimal parameter values. In this example, $C = 2^{11.8}$, $\gamma = 2^{10.3}$ would be used to train the best performing SVM model.*

### 3.1.4 Missing values

Another problem that arises with the real world data sets are the missing values. This can be caused either by the fact, that some data were not collected from the beginning and started to be recorded at some later point. Or there might be some information that just could not be obtained from the applicant.

If such cases occur only rarely, we can just ignore those few incomplete points with a clear conscience. Naturally, a different approach must be taken if the missing values concern a big part of the total observations in the data set, as in such case it is not possible to simply disregard them.

For categorical variables, it is common among practitioners, that a new category "N/A" or "missing" is created. The other way how to deal with this problem is replacing the missing values with the mode value in its category. Missing values in numerical variables on the other hand can be replaced by the mean or median value.

While this technique may seem quite arbitrary on the first sight, it actually makes sense from the pragmatic point of view. Most of the learning algorithms cannot handle the data sets containing the missing values. If we still need to use such observations, it is best to assume, that the missing parts attain the most probable values. The real-world interpretation of this technique would be: if you do not know some information about the applicant, assume the most common value.

## 3.2 Feature selection

Many data sets contain a high number (hundreds, thousands or even much more in some study areas) of input features albeit only couple of them can be relevant to the examined classification problem. Credit scoring is not a typical example of high-dimensional problems, because there are rarely data sets containing more than like 100 input features in total.

Regardless, decreasing the dimensionality of the task can be beneficial both thanks to lower computational costs as well as increasing the performance of the resulting model. The irrelevant input features can actually bring additional noise into the model.

This is why some feature filtering or selection is often performed before the actual model training can begin. There are several established methods for this task, most of them are used in general and not limited only to the support vector machines context.

It should be noted, that feature selection techniques must be based on some kind of heuristic algorithms, since selecting the optimal set of the input features is not as simple job as it might seem at first sight. Suppose we have a data set with $n$ input features. Each of them can be found in one of the two possible states: included or not included in the final model. Thus there are $2^n$ possibilities to choose from.

The feature selection techniques are often combined with the model selection techniques. Each variant is then evaluated using the out-of-sample data set, which prevents to choose such features that perform well on the training subset (e.g. just by chance), but are not actually relevant to the classification task.

### 3.2.1 Forward selection

Forward selection is a sequential iterative procedure to select appropriate set of relevant input features. We start with an empty set of included features, i.e. all $n$ features are in the *not included* state at the beginning. Now we iterate over these features and use exactly one at a time, thus creating $n$ different models in the first step.

All these models are evaluated using the model selection procedure and the input feature performing best on the testing subset is chosen and included in the model.

In the next step, the remaining $n-1$ features produce a new set of $n-1$ models, each of them incorporating two input features - the one that was selected in the first step combined with one from the remaining features. Yet again the models' performance is evaluated and the best performing feature is added to the set of included features.

This process goes on as long as it is necessary. In the support vector machines, there is no clear way how to decide, when the forward selection algorithm should stop. A researcher can for example set the reasonable amount of input features a priori. Or the stopping condition can be formulated as the minimum performance improvement required for each step. If in any given step the performance difference between the previous model and the new one is lower then some a priori set value, the Forward selection algorithm stops. The threshold value can either be zero (meaning to stop as soon as the newly added features start decreasing the performance of the model) or any other positive number.

### 3.2.2 Backward selection

The backward selection algorithm is a slight modification of the previous one. The main difference is that all of the input features are included in the model at the beginning. Then at each step, one worst performing feature is taken away in the same iterative manner. The stopping condition can be again some a priori chosen number of included input features or some performance threshold, as in the previous case.

Despite the fact, that forward and backward selection are essentially same, there are no guarantees that they end up with the same sets of the included input features.

### 3.2.3 F-score

A rather more quantitative and theoretically better justified approach to the feature selection is using of the F-score. Application of this method for the support vector machines was proposed by the LIBSVM authors in [10] and is used in many subsequent research papers on support vector machines in the credit scoring accordingly.

Given the training vectors $\boldsymbol{x}_k, k = 1, \ldots, m$, the number of positive and negative classes $m^+$ and $m^-$, respectively, and the averages of the $i$-th feature $\bar{x}_i$, $\bar{x}_i^+$, $\bar{x}_i^-$, respectively, then the F-score of the $i$-th input feature is defined by the following equation [10]:

$$F(i) = \frac{\left(\bar{x}_i^+ - \bar{x}_i\right)^2 + \left(\bar{x}_i^- - \bar{x}_i\right)^2}{\frac{1}{m^+-1}\sum_{k=1}^{n^+}\left(x_{k,i}^+ - \bar{x}_i^+\right)^2 + \frac{1}{m^--1}\sum_{k=1}^{n^-}\left(x_{k,i}^- - \bar{x}_i^-\right)^2} \tag{3.2}$$

The higher the F-score the higher is the discriminative power of the respective attribute. The F-score is therefore a quite easily computable metric for judging the importance of each input feature although it has some drawbacks, too. The major disadvantage is that the F-score does not assess the relationships among the input features itself.

After the F-score is computed for all the input features some threshold has to be determined. Features with the F-score above this threshold will then be included in the model. The selection of the appropriate threshold can be done by iterating over some reasonable range of possible threshold values and minimizing the validation error of the corresponding models. [10]

### 3.2.4 Genetic algorithm

Genetic algorithms (GA) are biologically inspired optimization methods. They are very powerful in the optimization situations, where unattainable number of possibilities exist and the landscape of the problem is too complicated to use analytical solution.

Genetic algorithms mimic the process of the natural selection. The optimization problem must be translated in a suitable expression that plays the role of the DNA throughout the optimization - usually the binary encoding of the solution is used.

After that, some random set (i.e. population) from the set of all possible solutions is generated and evaluated. Subsequently, the best performing solutions are used to generate a new population through the combination of inheritance and several genetic operators: mutation and crossover. The process is repeated until some convergence criterion is met.

Although the genetic algorithms are not guaranteed to find a global extreme in all cases, it turns out that in many applications they are capable to find *good enough* solution in a satisfactory low number of steps. Concerned reader is referred to the relevant literature [26] for more information on this truly interesting topic.

In the association with the support vector machines, genetic algorithms are sometimes used for the feature selection. The problem is binary encoded in the chain of 0-1 numbers of length $n$ (the number of all features in the data set), whereas 0 representing the features that are not included and 1 representing the included features. E.g. for $m = 3$, the chain 010 represents a solution, where only 2nd feature is included in the model, while 111 represents the model, where all features are considered relevant for the model.

A set of random chains is generated and the respective models are evaluated according to the model selection techniques. After the evolution is simulated in the sufficient number of steps, the chains representing near-optimal selection of input features will prevail in the entire population. Either the best performing model is directly chosen or the performance in combination with the number of features is taken into the consideration.

The experiments on the credit data sets show, that the genetic algorithms are effective for the feature selection, since they end up with as little as half of relevant features when compared to other techniques without sacrificing the performance of the models [36]. But the magic always comes with a price. In this case, the extreme requirements for the computer power represent such a cost.

## 3.3 Model selection

In practice, we have usually many model candidates to choose from and it may not be clear which one to use optimally for a given task. In the context of the support vector machines, the proper kernel function must be picked and then its optimal parameters found. Although most researchers use either linear or radial basis kernels for the credit scoring application, we need some general approach how to determine the optimal model from the class of all possibilities.

In principle, there are 3 basic methods that differ in the precision and the required computing resources.

### 3.3.1 Hold-out cross validation

The idea of this approach is rather simple: split the data set $S$ randomly into two subsets $S_{train}$ and $S_{test}$, use only $S_{train}$ subset for training each model, and measure its performance on the $S_{test}$ subset. The model that performs best on $S_{test}$ is chosen as the optimal. Optionally, the selected model can be then retrained using the whole dataset $S$ in the end.

The advantage of the approach is low computational cost, while the disadvantage is that only the part of the dataset is used in the model selection process. This may be issue in cases, when the data is scarce or its acquisition was costly [46]. The split ratios 70:30 or 50:50 are most commonly used in the literature.

### 3.3.2 k-fold cross validation

During this process, dataset $S$ is randomly split into $k$ subsets $S_1, \ldots, S_k$. One of these subsets is left as a testing one and the rest $k - 1$ are used to train the model. This process repeats $k$ times, so that each subset is left out as a testing one. Thus we obtain $k$ performance measures in the end. These measures are averaged for each model and the model with the best performance is chosen.

The advantage of the k-fold cross validation is more efficient use of the dataset, the disadvantage are higher computational costs since each model has to be trained $k$ times instead just once as in previous case. Usually, $k = 10$ or $k = 5$ are used in the literature as the default values. [46]

### 3.3.3 Leave-one-out cross validation

This is the special case of the previous approach when $k = m$, i.e. each subset contains one observation. The computational costs are extreme so that the method can be used only on small datasets.

## 3.4 Model evaluation

When a best model is to be chosen from the set of several alternatives, a criterion to evaluate each model's performance needs to be established. The simplified confusion matrix of the classification problem is given by Table 1.

| | | PREDICTED | |
|---|---|---|---|
| | | -1 | +1 |
| ACTUAL | -1 | True Negative (TN) | False Positive (FP) |
| | +1 | False Negative (FN) | True Positive (TP) |

*Table 1: The confusion matrix of the classification model. Each cell of the matrix contains a number of observations suiting the conditions in the respective heading. [42]*

Simple criteria can be derived from this confusion matrix:

- **Accuracy (ACC)** $= (TP + TN)/(TP + TN + FP + FN)$
- **Classification error** $= (FP + FN)/(TP + TN + FP + FN)$
- **Sensitivity** $= TP/(TP + FN)$, also called true positive rate, hit rate or recall
- **Specificity** $= TN/(FP + TN)$, also called true negative rate

The support vectors machine algorithm leads to the separating hyperplane $(\boldsymbol{w}, b)$ which geometrically is, by default, positioned exactly in the middle between the support vectors of the $+1$ and $-1$ classes. Such treatment implicitly assumes that the negative consequences coming from the wrongly classified $+1$ class are of same seriousness as from the wrongly classified $-1$ class. This may not be always the case.[20]

[20] As an extreme example, consider the classification in medicine, where False positive test leads to emotional distress and the need for repeating the test while False negative can lead to the patient's death.

From his previous experience, the lender may know, that giving the loan to a bad creditor (falsely classified as a good one) is worse than missing the opportunity to fund a good creditor (falsely classified as a bad one). Such an information should be included in the model. To achieve this, we must shift the separating hyperplane closer to one or another class. Mathematically speaking, it means tweaking the parameter $b$ or, equivalently, changing the decision function so that the *threshold* score (2.42) for classification will be different from the default value 0.

In the logistic regression, this means to find a threshold $\tau$ other then the default value and then to classify an observation as $+1$ class when $P(Y = +1|\boldsymbol{x}) > \tau$ and $-1$ otherwise.

The disadvantage of the metrics derived from the Table 1 rests in the fact that they represent a static picture, which is valid only for one choice of the threshold value.

A more sophisticated approach is represented by the *Receiver Operating Characteristic* (ROC) curve, which is a plot of the *Sensitivity* on the y-axis against $(1 - Specificity)$ on the x-axis across all possible thresholds. [42]

An aggregated single number, the area under the receiver operating characteristics curve (simply called as *Area Under Curve* or AUC), can be then calculated. AUC then serves as a metric for evaluating the model performance. The value of AUC = 0.5 corresponds to the random classification, while 1.0 indicates a perfect classification. An AUC is well-established in credit scoring and is explicitly mentioned in the Basel II accord. [42]

Formal statistical test has been developed by DeLong & DeLong to compare two or more areas under correlated ROC curves [18]. It can be used to test the hypothesis that the AUC of two different models (as measured on the same data set) is same. The method was later reformulated to the more accessible form in [21, p. 11] and applied specifically to the credit scoring problems.

# 4. Support vector machines in Credit risk

This chapter describes the current literature on the support vector machines in credit scoring with a special focus on the comparison of the SVM performance with other classifiers, especially with the logistic regression.

Over time, couple of standardized data sets have been used for the purpose of comparability of different models' performance in the space and time. Notable examples include German data set [31] and Australian data set [3]. I will refer to them in the following lines.

The original idea to separate the linearly separable observations with a hyperplane and to choose such a hyperplane that maximizes the minimal distance from the observation dates back to 1962 [60]. The concept remained unnoticed by most researchers until the pivotal work of Vapnik and Cortes [12] wasn't published in 1995. This paper formulated support vector machines (then referred as support vector networks) in the current form (soft margin, kernel function) and showed the superiority of SVM over the state-of-the art classification algorithms of that time on the example of the handwritten numbers recognition.

This work had a boosting impact on the subsequent advance of interest in SVMs towards the close of the 20th and on the beginning of the 21st century. Its effect could be compared with consequences of the pioneering paper [52] about the back-propagation algorithm that started the late 80s interest in the neural networks, which were neglected and underestimated till then.

Over time, papers examining the performance of the SVMs in the credit scoring and other financial applications started to appear, especially after 2000.

Baesens in 2003 [4] applied SVMs and other classifiers to several credit data sets. He concludes that SVMs perform well when compared with other algorithms; they do not however always result in the best performing model. He also notes the specifics for the credit data which are typically hardly separable by any decision surface. It's given by the fact that the data cannot capture the complexities of an individual's life. It is quite common, that the misclassification on credit data reach around 20 % or 30 %.

Li in 2004 [43] studied the SVM performance on 1 000 credit records of a Chinese commercial bank. He concludes, that SVM performs noticeably better (by more than 50 %) in the terms of hit rate against the credit scoring methodology the bank used at the time. However, he did not in any way specify, which methodology did the bank use then and this mere fact degrades the significance of the whole paper.

Schebesch and Stecking (2005) [54] apply SVM to a database of applicants for building and loan credit. They conclude that SVMs perform slightly better than LR, but not significantly so.

In Huang et al (2007) [36] the SVM performance on German and Australian credit data set is compared against other data mining methods (back-propagation neural network, genetic programming and decision trees). For the support vector machines, various feature selection techniques are tested: unrestricted model, features selection by F-score and features selection by GA-approach. The GA approach used significantly lower number of features than the other methods, with a slightly higher hit rate. The authors conclude that SVM is competitive method for the credit scoring when compared against other commonly used data mining algorithms but is not significantly more accurate than other methods.

Bellotti et al (2009) [5] used the data set consisting the records of 25 000 credit card users and compared the performance of SVM with LR, LDA and kNN. They found that the non-linear kernels in SVM do not perform better than the simple linear SVM. Especially the polynomial kernel performed poorly and authors attribute this fact to the possible over-fitting. The method for selecting significant features in the data using the square of weights on features output proposed by Guyon et al (2002) [27] was used and enhanced. The paper concluded that LR and SVM tend to select same features as the most important ones. Generally, the SVM performed slightly better than the logistic regression, according to Bellotti.

Ghodselahi (2011) [25] utilized 10 SVM classifiers as the members on an ensemble models and conclude that its performance is significantly better than of any individual SVM model or logistic regression. The training was performed on the standardized German data set. However the reported AUC for the logistic regression is worse than the AUC reported by others on the very same data set (e.g. [42]) which casts a shadow of doubts on the conclusions of this paper. Chances are, that the conclusion stands on the simple fact, that authors were not able to fully utilize the strengths of the LR approach. This paper is nevertheless a representative of a research trend of the recent years when ensemble models are preferred to the individual ones.

There is usually high imbalance in credit data sets, as the bad cases occur significantly less often than the good ones. Brown (2012) in [7] examined the effect of the data imbalance on several data mining algorithms. Support vector machines tended to perform poorly, when the percentage of bad cases in the training data set decreased and the data imbalance increased.

The solution of the data imbalance is not as simple as it may appear. Simple *undersampling*[21] does not necessarily solve the problem. On the contrary, it may lead to the solutions that are further from the ideal solution, as [1] shows and proves. This paper suggests *oversampling* instead: the technique to generate new artificial bad cases from the actual ones, so that their total number matches the number of good cases. Authors achieved positive results with their method on 10 different data sets[22]. Despite this fact, I find the proposed idea kind of "spooky" and not suitable for the credit scoring.

The problem of the unbalanced data in combination with Support vector machines remains unresolved.

Recently, there were some doubts regarding the AUC as an indicator for comparing different classifiers performance due to its fundamental incoherence [30]. Several alternative measures have been proposed in the literature, one of them being H-measure as per [29]. However, according to the extensive empirical classifier comparison in [42] there is a high correlation between the AUC and H-measure. In other words, both measures give same conclusions in the vast majority of cases. Authors conclude that using AUC to evaluate credit scoring performance remains "safe" from the empirical point of view.

Probably the most in-depth analysis of the current state-of-the-art machine learning algorithms was performed by Baesens et al (2013) [42], exactly 10 years after their first comparative paper [4] was published.

---

[21] randomly take away good cases from the training data set, so that the final ratio of good and bad cases will be 50:50

[22] none of them is credit scoring, however

Authors compared the performance of different algorithms with many different settings (1141 models in total) on 7 real-life credit data sets (German and Australian were also among them) using 4 different performance measures to evaluate the models (percentage correctly classified, AUC, H-measure and Brier Score). The models from 3 wide families were examined: individual classifiers (mostly linear classifiers), homogeneous ensemble and heterogeneous ensemble classifiers. [42]

The results are not favourable to the SVM. The performance of the SVM models with linear kernel was moderate when compared with other individual classifiers. The Gaussian kernel performed slightly better. Logistic regression noticeably outperformed all other linear classifiers and ended up as second best individual algorithm, beaten only by the artificial neural networks by a small margin. It turns out that SVM can perform really great on some datasets, but the overall results on different data sets are average at best, according to their paper. [42]

Authors remark that the overall performance of the "individual classifiers" did not significantly improve over the last decade. In their opinion it implies that the possible limits were reached with this approach. Ensemble classifiers (homogeneous as well as heterogeneous) performed markedly better on the 99 % significance level: Random forests (RF) as being the best of homogeneous and HCES-Bag the best of heterogeneous ensembles (and the best overall). [42]

Authors propose to use Random forests as the benchmark for the future research on new classification algorithms in credit scoring. Despite it ended up as second best, it is (unlike HCES-Bag) easily available and implemented in many standard data mining software. [42]

The authors argue against the current common practice of using logistic regression as the only classifier for the purpose of comparing a newly proposed classifiers, since beating the logistic regression is no longer a challenge. While outperforming random forests can be considered as a signal for a methodological advancement. [42]

The most recent progress in the machine learning in general were reached using Deep learning, a set of algorithms enhanced and derived from the neural networks, though with a significantly more complex architecture.

The pioneers of this approach accomplished some noticeably achievements in the computer vision, beating other existing algorithms by a large margin and achieving a human-competitive performance on major benchmarks.[23] [11]

Although the idea is not new, it did not gain an attention of the researchers until recently with new applications being developed and explored, like speech recognition, natural language processing, facial recognition and others. When a new breakthrough machine learning algorithm is developed, its performance is usually tested on tasks like these, at first.

Credit scoring is not usually attractive enough among the engineers and technically oriented researchers who are the originators of such new advancements. Consequently, the research of their applications on credit scoring frequently comes with a delay of several years behind the latest trends. Up to now I was not able to find any relevant paper on Deep learning for credit scoring and I believe this might present an interesting topic for further research.

[23]See Deep Learning Wins 2012 Brain Image Segmentation Contest at `http://www.idsia.ch/~juergen/deeplearningwinsbraincontest.html`

# 5. Application

This part of my work dwells on the application of the support vector machines for building the predictive model using the real world credit data set. I used my own code written in the MATLAB environment with the help of the LIBSVM library [9] that became a standard routines package for the support vector machines during the years thanks to its availability, multi-platform support and continuing optimization. While MATLAB itself contains a built-in support for SVMs, it is not comparably efficient in the sense of the computational time needed.

The data set I used for the research purposes was obtained from the Estonian peer-to-peer lending platform IsePankur.ee/Bondora.com[24], that are accessible online thanks to the transparency policy promoted by the platform. [37]

## 5.1 Peer to Peer Lending

Financial sector, despite being one of the most regulated and supervised economic sectors [32], has recently experienced several important and possibly game-changing disruptive innovations like crowdfunding or peer-to-peer lending, among others.

Since peer to peer lending is such a new phenomenon and there are very few peer-reviewed papers studying it from the perspective of the credit scoring, it might be useful to describe its basic principles and examine the common and distinct characteristics of the P2P lending and the traditional banking lending.

[24] In the course of writing this thesis, IsePankur rebranded and renamed to Bondora.com due to the ongoing expansion to other European markets; the original name was not very catchy and easy to pronounce for non-Estonians and this was probably the main reason for rebranding.

I chose Bondora as my data source among other possibilities[25] because it's one of the few platforms that allows foreign investors to participate[26] and because I have the personal experience with this platform for more than 15 months at the moment.

While the principles of P2P lending are same, the details of implementation can vary from platform to platform (from country to country). Since these details can influence the credit data collection and the usability of the data set, I will primarily focus on the description of the Bondora specifics and details.

Peer to peer lending (P2P Lending, sometimes also Social Lending) denotes the practice of matching the individual lenders and borrowers for the unsecured consumer loans [24]. This is accomplished with the means of electronic, automatic or semi-automatic, brokers' systems without traditional intermediaries and thus can be regarded as one of the symptoms of the ongoing *disintermediation* [6].[27]

There are 3 main factors that account for the growing popularity of the P2P lending: low barriers, automation and liquidity.

---

[25]Open access to the credit data is an industry standard in the peer to peer lending, apart from Bondora some other big platforms like Lending Club, Prosper or Zopa could have been used.

[26]The heavily regulated US platforms are restricted only to US citizens, moreover residents only from some US states.

[27]Although there are reports, that there is a growing presence of the banks, specialized funds and other "professionals" among the investors on the greatest P2P lending markets and the platform operators are reportedly trying to attract these types of investors to keep the liquidity on their markets. Rather than disruptive competition, the P2P-banks relationship can turn out to be a symbiotic one in the future.

**Low barriers:** each loan application is divided into small financial amounts and there are usually very low minimum investment requirements. At Bondora, it is possible to invest with as little as €5 in each loan. This implies there are usually hundreds of people financing one loan. And on the other hand it is possible to build a relatively well diversified portfolio of the consumer loans with an amount that is affordable to many non-affluent investors. The operators are regulated by the financial authorities[28] and can be therefore considered as an alternative for the individual yield-seeking investors.[29]

**Automation:** The proper diversification of one's investment portfolio can be time-consuming. This is the reason, why all the platforms adopt some form of automation of the investment process. Each platform has some kind of its own internal scoring system, that might not be perfect in the sense of the discrimination power, but its performance is good enough to allow for creating the automatic or semi-automatic investment plans.

An investor sets the level of the maximum riskiness and the minimum interest rate required, optionally sets some other limits[30] and these plans then invest small amounts of his total account to each individual loan that meets the criteria. At Bondora, the internal rating consists of an information about disposable income (borrowers are sorted into three categories: A,B,C) and payment history (6 groups: 1000, 900, 800, 700, 600, 500, the higher the better).

The resulting interest rate of the loan is either determined on the auction principle or fixed by the loan applicant or the platform operator. Bondora allowed both of these models in the past.[31]

---

[28]Bondora is supervised by the Estonian central bank, ZOPA by the Financial Conduct Authority and the Lending Club and Prosper come under the authority of U.S. Securities and Exchange Commission - to name the 4 largest marketplaces at the moment.

[29]Indeed, in United States it is possible to include these investments to one's IRA account under 401k, make the P2P lending part of the retirement investments and realize the tax advantages on the top of that [41].

[30]For example minimum/maximum age, education, DTI ratio and other parameters

[31]Recent changes in the system abandoned the auction principle leaving only the fixed-rate loans; my data set nevertheless comes from the time when both these models were available.

In the auction, borrower sets the amount requested and the maximum interest rate he is willing to accept. In the predefined time period, investors then bid the amount and minimum interest rate required. When the time is up and the total amount of bids is at least equal to the amount requested, the loan is assigned to the borrower. The interest rate is determined by the Reverse Auction principle: only the lowest bidding investors are granted the right to finance the loan and interest rate is set as the highest rate that was sufficient to become a financing investor (with respect to the amounts requested by the borrower and offered by the investors).

In the fixed-rate setting, the applicant suggests the interest rate and investors either bid on the loan with some amount of money or pass this opportunity and wait for other loan applications. If enough money is offered by investors, the loan is approved. If not, the borrower can repeat the process with higher interest rate.

From the investor's point of view, there is a considerable level of automation in the following steps, too. The instalments are automatically credited to investors' account (proportionally to their amount invested in the loan) as soon as the operator receives the money from the debtor. Automatic remainders are sent in case the borrower is late with his payment.

Default is defined as missing part of the payments by more than 60 days. If default occurs, the whole balance of loan turns into an immediately collectible claim and the operator initiates the debt recovery process using the legal enforcement tools available in the country of the debtor's residency. The workout process is again fully administered by the operator without any need for the cooperation from the individual investors. The operator is motivated to perform well in this process by two forces. First, he earns part of the recovered money. Second, thanks to the data openness, the global default rates and recovery rates can be monitored on the almost real-time basis. It is in his best interest to keep these numbers low so as not to drive investors out of the market.

**Liquidity:** With most of the P2P lending platforms, the loans are more or less securitized and the secondary market is established to trade the cash flow claims from the loans among the investors. Bondora allows to resell any non-defaulted loan investment and the buyers have available all the resources to make the informed decision - not only the information from the loan application but also pre-existing course of the loan performance: payment plan, date and amount of each payments, remainders sent, missed payments etc. The liquidity on the secondary market is sufficiently high and the transaction costs represent 1.5 % of the principal amount traded, which allows investor to liquidate their positions quite comfortably, if needed.

Default risk aside, the main danger in the P2P lending represents the operator risk. Indeed, there were already some bankruptcies in the past and this risk should be borne in mind when considering investments in this new and still rather experimental product. At Bondora, the loans are legally binding contracts signed between lender and borrower (or between lenders in case of reselling).[32] The existence of the claims are not entirely dependent on the existence of the market operator.[33] The enforceability of a portfolio of a huge amount of tiny loans would be, however, rather questionable.

## 5.2 Data description

The data set obtained from the Bondora Data Export [37] contains 6818 loans provided between 28th February 2009 and 16th April 2014. Since my goal is to build a model predicting the probability of default on a 1-year horizon, only the loans provided between 28th February 2009 and 16th April 2013 were used, 2932 data points in total.

---

[32]Estonia is well known for its pioneering attitude to the eGovernment including the electronically concluded contracts and their effective enforcement at the courts.

[33]US platforms, on the other hand, take part in the truly securitization schemes where pass-through notes are emitted for each loan. Investors do not (de iure) invest in the loans itself - they buy (and sell) these notes. In all cases, the property of the investors is usually separated from the operator's assets.

Information available for each loan are quite thorough, include the personal as well as behavioural information about the borrower and cover also the performance of each loan and the recovery process in case the loan defaulted. The complete list of the data set values are summarized in the Table 2.

All the numerical data were normalized using the z-score. All the categorical variables were encoded using the technique of the dummy variables described in Section 3.1.1. The missing values were replaced by the newly created "N/A" category in case of the categorical variables and by the mean value in case of the numerical variables. No effort has been made to decrease the number of categories in the categorical variables (e.g. to join several categories with similar probability of default into one).

In the alternative approach, the categorical variables were replaced by the respective Weight of Evidence of each category as defined by equation (3.1). My goal was to learn, if and how will the treating of the categorical variables affect the performance of different models.

The binary class variable "1-YR DEFAULT" was calculated, being 1 in cases where the default occurred less than 365 after the loan origination date and 0 in cases when the default occurred later or no default was observed at all. The loan is considered to be in default, when borrower missed the scheduled payment by more than 60 days.

Of the 2932 loans, 656 defaulted by the end of the first year (or 22.37 percent of all loans granted). In total 2 096 352 EUR was lent; the total Exposure at Default for the defaulted loans was 281 946 EUR (or 13.45 percent of the amount). These numbers demonstrate a high risk-exposure in the P2P lending portfolios even when compared with the credit cards delinquency rates. On the other hand, the average recovery rates of the defaulted loans reach 60-70 percent and are noticeably higher than in the credit cards loans portfolios. [8]

| | | | |
|---|---|---|---|
| ApplicationSignedHour | n | | Hour of signing the loan application. |
| ApplicationSignedWeekday | n | | Weekday of signing the loan application. |
| VerificationType | c | 3 | Method of application data verification. |
| language_code | c | 3 | Language settings of the borrower. |
| Age | n | | Age of the borrower (years). |
| Gender | c | 2 | Gender of the borrower. |
| credit_score | c | 6 | Payment history (1000-500). |
| CreditGroup | c | 3 | Disposable income (A,B,C). |
| TotalNumDebts | n | | Total number of debts. |
| TotalMaxDebtMonths | n | | The longest period when loans were in debt. |
| AppliedAmount | n | | Amount applied. |
| Interest | n | | Maximum interest acceptable for borrower. |
| LoanDuration | n | | The loan term in months. |
| UseOfLoan | c | 9 | Use of loan as declared by the borrower. |
| ApplicationType | c | 2 | Auction or fixed-interest rate. |
| education_id | c | 5 | Education of the borrower. |
| marital_status_id | c | 5 | Current marital status of the borrower. |
| nr_of_dependants | n | | Number of children or other dependants. |
| employment_status_id | c | 5 | Current employment status. |
| Employment_Duration | n | | Years with the current employer. |
| work_experience | n | | Work experience in total (years). |
| occupation_area | c | 19 | Occupation area, economic sectors. |
| home_ownership | c | 10 | Homeownership status. |
| income_total | n | | Total income. |
| DebtToIncome | n | | Debt to income ratio (DTI). |
| NewLoanMonthlyPayment | n | | New loan monthly payment. |
| AppliedAmountToIncome | n | | Applied amount to income, %. |
| FreeCash | n | | Discretionary income after monthly liabilities. |
| LiabilitiesToIncome | n | | Liabilities to income, %. |
| NewPaymentToIncome | n | | New payment to income, %. |
| NoOfPreviousApplications | n | | Number of previous loan applications. |
| AmountOfPreviousApplications | n | | Value of previous loan applications. |
| NoOfPreviousLoans | n | | Number of previous loans approved. |
| AmountOfPreviousLoans | n | | Value of previous loans |
| PreviousRepayments | n | | Previous repayments |
| PreviousLateFeesPaid | n | | Previous late charges paid |

*Table 2: List of the data set features. Second column: c = categorical, n = numerical variable. Number of categories with non-zero observations are stated for the categorical variables.*

## 5.3 Benchmark model: Logistic regression

Logistic regression as the industry standard for the credit scoring was used to build a benchmark model for the purpose of the performance comparison with the Support vector machines. The backward selection algorithm with the threshold p-value = 0.05 was used to identify the statistically significant features in the data set. The Chi-Square Wald statistics was used as a criterion, which allows for evaluating the contribution of categorical variable as a whole. The following analysis of variables significance is inspired by the approach taken in [5].

I performed 10 independent runs of the backward selection algorithm, each time with the different random training and testing subsets. The resulting models differed, sometimes quite significantly, in the number of variables selected as statistically significant. They were however very similar in terms of the performance measured by the AUC.

Table 3 shows the results of the feature selection process for two different handling of the categorical variables. We can clearly see, that there are 11 explanatory variables that are consistently selected on every run while the significance of some others seems to be unreliable and data-dependent. They may survive the elimination process on some of the random data subsets but drop out as insignificant on the other ones.

Another remarkable conclusion is that the process of handling the categorical variables HAS an effect on the variable selection process. Although the results are similar, they are not same - there are couple of variables in the woeised dataset, that does not appear in the dummy variables dataset (like ApplicationSignedHour). This may be caused by the fact, that MLE algorithm used in Matlab for the logistic regression parameters estimate had some issues with the dummy dataset. Sometimes, it was not able to reach results during a satisfactory number of iterations. Woeised dataset did not exhibit such troubles and the parameters estimation was fast and efficient.

On average, 15 explanatory variables stayed in the model after the feature selection process. There were minimum 13 and maximum 17 variables in the model on the 0.05 significance level.

All the variables that occurred in the absolute majority of trials were then used to build a final logistic regression model. The detail description, analysis and maximum likelihood estimates are given in Appendix A.

| Dummy variables | | | WOEisation | | |
|---|---|---|---|---|---|
| Variable | # | Rank | Variable | # | Rank |
| credit_score | 10 | 1.00 | credit_score | 10 | 1.00 |
| Interest | 10 | 2.00 | Interest | 10 | 2.00 |
| PreviousRepayments | 10 | 4.30 | ApplicationSignedHour | 10 | 3.90 |
| Age | 10 | 4.70 | PreviousRepayments | 10 | 4.80 |
| AmountOfPreviousLoans | 10 | 4.90 | home_ownership_type | 10 | 5.00 |
| marital_status_id | 10 | 6.50 | AmountOfPreviousLoans | 10 | 5.50 |
| language_code | 10 | 6.60 | language_code | 10 | 5.80 |
| NewLoanMonthlyPayment | 10 | 9.43 | marital_status_id | 10 | 8.90 |
| AppliedAmount | 10 | 11.80 | Age | 10 | 9.90 |
| nr_of_dependants | 10 | 12.70 | occupation_area | 10 | 10.70 |
| occupation_area | 10 | 13.00 | employment_status | 10 | 11.00 |
| employment_status | 9 | 11.89 | NewPaymentToIncome | 9 | 11.11 |
| VerificationType | 8 | 7.69 | NewLoanMonthlyPayment | 9 | 12.22 |
| NewPaymentToIncome | 8 | 11.25 | ApplicationSignedWeekday | 7 | 15.00 |
| ApplicationType | 7 | 14.43 | AppliedAmount | 6 | 13.83 |
| UseOfLoan | 6 | 11.50 | ApplicationType | 5 | 14.80 |
| income_total | 5 | 15.40 | nr_of_dependants | 4 | 14.25 |
| Employment_Duration | 4 | 17.50 | LiabilitiesToIncome | 2 | 14.50 |
| home_ownership_type | 3 | 10.00 | UseOfLoan | 2 | 15.50 |
| FreeCash | 3 | 15.00 | education_id | 1 | 12.00 |

*Table 3: Summary of the variables selected by the Backward selection algorithm with the threshold p-value = 0.05 for two different categorical variables handling. The column # presents number of independent runs in which the variable was selected as statistically significant. The column Rank shows the average rank of each variables when sorted by it's p-value in each model.*

## 5.4 SVM with Linear kernel

Unlike logistic regression, there are no standardized and generally applicable procedures to judge the statistical significance of the explanatory variables and to select the most important ones. The techniques used all over the relevant literature are rather heuristics or ad-hoc ideas that could not be relied upon in general.[34]

I have extensively experimented with the procedure described in section 3.2.2 with the AUC as decision criterion, using the chi-square statistic test to distinguish the real performance improvement from the differences caused by the pure chance.

[34]See [35] as an example.

Unfortunately, the feature selection was very unstable. Except the most significant features (credit_score, Interest), the selected final models seemed to be affected more by random fluctuations and no clear pattern could be identified. This is true for SVM with linear kernel as well as with the Gaussian kernel. After many attempts, I have discarded this approach as unreliable and abandoned the idea.

The main challenge arises from the fact, that there are no formal statistical tests similar to the Wald statistics used in Logistic Regression which could be used to evaluate the contribution of the given variable to the performance of the whole model in case of Support Vector Machines. Indeed, there are some proposals, in the literature, but nothing that could be compared in terms of formal definition and theoretical background.

Other quantitative approaches (like F-Score mentioned in section 3.2.3) can play only subsidiary role in the feature selection process due to the strong assumptions they depend on (e.g. the independence of features).

Guyon in [27] does propose some methods for the feature selection, namely the square of the weights from the hyperplane generated by the support vector machines, $(w_i)^2$. Although theoretically justified, this can be used only as the ordinal criterion to rank the explanatory variables. It does not tell us, whether the specific value of the square of the weight is high enough to be considered as statistically significant. Another difficulty arises with the categorical variables in the dummy approach - how should be the square of weights relevant to the categorical variable aggregated so that it can be compared with one single number obtained from the numerical variable?

Bellotti in [5] cunningly bypasses these problems: "We set a threshold of 0.1 [for weights on explanatory variable] and all features with weights greater than this will be selected as significant features. This threshold level is chosen since we found it yields approximately the same number of features as the LR method."

The quotation emphasizes the greatest obstacle for the wider application of the support vector machines for credit scoring. As Bellotti&Crook wrote in the introduction of their work [5]:

> Financial institutions are primarily interested in determining which consumers are more likely to default on loans. However, they are also interested in knowing which characteristics of a consumer are most likely to affect their likelihood to default. (...) This information allows credit modellers to stress test their predictions.

So our approach to feature selection could be as follows:

- Choose the statistically significant features using Logistic regression and the formal statistical tests (e.g. Wald statistics).
- Train support vector machines using these features.

Since the goal of my thesis is to compare performance between SVM and LR, this approach may be deemed reasonable. As a consequence, SVM cannot be regarded as a stand-alone method in such a framework.

Genetic algorithm for feature selection problem was proven to be superior to the above mentioned approaches [36]. Genetic algorithm approach, on the other hand, brings up the black box problem. The resulting models may be better and more robust in terms of the discriminative power. But with its inability to quantify the contribution of each variable, one cannot expect that such framework could be adopted by conservative and highly regulated financial institutions, no matter how perfect the resulting models would be, compared to traditional ones. For this exactly reason, I did not consider genetic algorithm for feature selection as a viable option for this purpose.

### 5.4.1 Optimal cost parameter search

Support vector machines bring another hitch to get over - before the actual training begins, the optimal cost parameter C needs to be found. I have used the technique described in the section 3.1.3.

The values of $log_2 C$ from the interval $(-25; 10)$ were examined. Since the default value recommended by the LIBSVM manual [9], $log_2 C = -1$, was among the best performing possibilities, I have selected that value as the optimal one to proceed.

One very important conclusion regarding the categorical variables follows from the optimal cost parameter search. While the optimal $C$ was clear and easy to find in the case of "woeised" data set, the exact opposite is true in the case of "dummy variables" data set.

In the latter case, the performance exhibited a chaotic behaviour. No matter how fine the division of the possible values was, even the slightest change of $C$ led to the great and unpredictable changes in the observed AUC on the testing sample. Such phenomenon is rather disturbing and indicates a lower-than-expected robustness of the linear SVM in combination with the dummy variables used to express categorical values.

Another issue arose with values $log_2 C \geq 2$ and was common for both the data sets. The time needed for the SVM algorithm to find an optimal hyperplane rises exponentially for the higher values of $C$ (see Figure 7). The increased computational demandingness does not translate into higher performance, though. Based on the numerous experiments carried out for the sake of this thesis, I cannot recommend using values higher than $log_2 C \geq 5$, as they lead to unbearably long training times even on smaller data set comprising of just couple of hundreds of data points.

## 5.5 SVM with Gaussian kernel

Radial basis function as defined in (2.38) is probably the most used non-linear kernel in connection with the support vector machines. The Gaussian kernel maps the data set from the original space with $n$ dimensions to the feature space with infinite number of dimensions. Despite this fact, it has only one parameter. Thus, RBF unites the sensible computational costs with the robustness and flexibility of the separating hyperplane shapes.

Together, there are 2 parameters to be found: cost parameter $C$ and shape parameter $\gamma$. The recommended 2-D grid search technique as described in Section 3.1.3 and depicted at Figure 6 was used for this task.

Once again, the stability and robustness of the search process was by far higher in the case of the woeised data set.

The optimal parameters $log_2 \gamma = -3$ and $log_2 C = 2.5$ were found to maximize the AUC on the test sample.

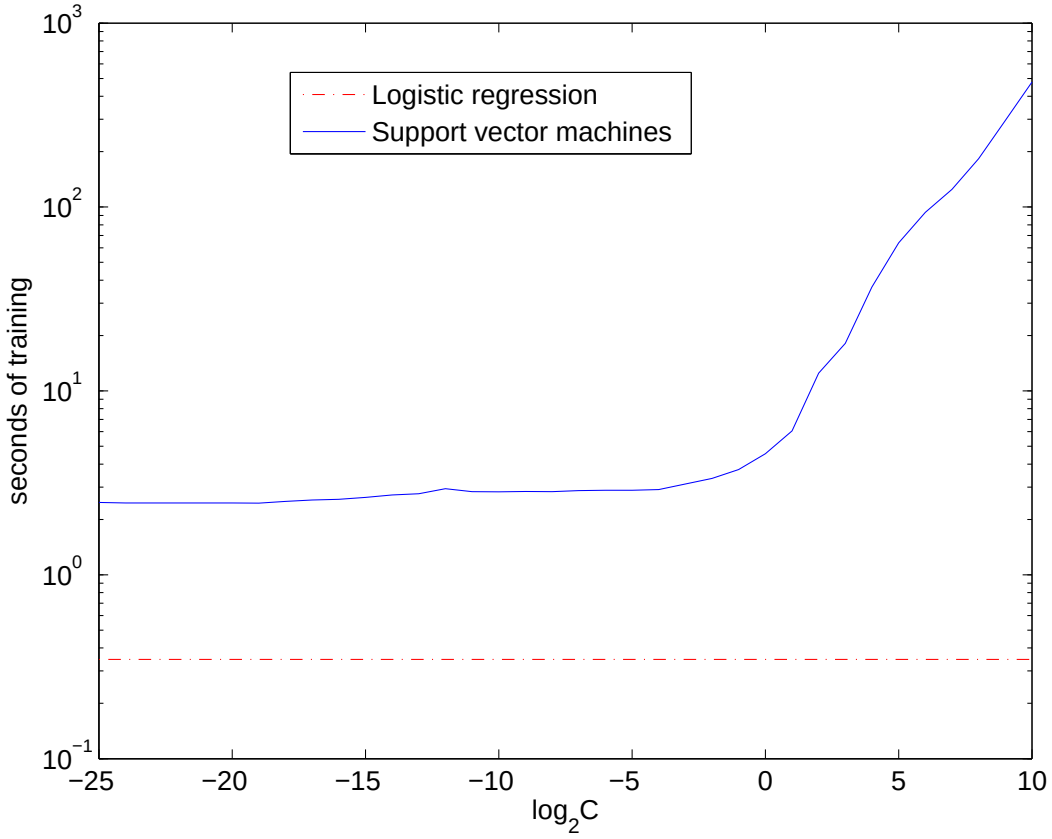


*Figure 7: The time necessary to train the support vector machines with linear kernel as a function of the cost parameter $C$. The time on the y-axis is in the logarithmic scale. For values $log_2 C > 0$, the dependence becomes roughly exponential and the training time rises quickly to the unbearable values. The training time of the logistic regression on the very data set is shown for comparison ($t = 0.347$ seconds).*

## 5.6 Model results

There are three questions I was looking the answer for:

- How is the model performance affected by the number of explanatory variables included in the model?
- Does the feature selection process bring some added value or will the performance of the unrestricted model be satisfactory enough to skip this time-consuming task with no remorse?
- How is the model performance affected by the way the categorical variables are treated?

| Dummy variables | | WOEisation | | |
|---|---|---|---|---|
| Model | AUC | Model | AUC | Diff. |
| credit_score | 0.641 | credit_score | 0.650 | +0.009 * |
| Interest | 0.717 | Interest | 0.722 | +0.006 ** |
| PreviousRepayments | 0.727 | ApplicationSignedHour | 0.735 | +0.008 |
| Age | 0.729 | PreviousRepayments | 0.744 | +0.015 ** |
| AmountOfPreviousLoans | 0.735 | home_ownership_type | 0.753 | +0.018 ** |
| marital_status_id | 0.747 | AmountOfPreviousLoans | 0.758 | +0.011 |
| language_code | 0.750 | language_code | 0.761 | +0.011 |
| NewLoanMonthlyPayment | 0.752 | marital_status_id | 0.765 | +0.013 * |
| AppliedAmount | 0.752 | Age | 0.767 | +0.015 ** |
| nr_of_dependants | 0.754 | occupation_area | 0.770 | +0.016 ** |
| occupation_area | 0.753 | employment_status | 0.772 | +0.019 ** |
| employment_status | 0.755 | NewPaymentToIncome | 0.774 | +0.019 ** |
| VerificationType | 0.760 | NewLoanMonthlyPayment | 0.775 | +0.015 ** |
| NewPaymentToIncome | 0.762 | ApplicationSignedWeekday | 0.775 | +0.014 * |
| ApplicationType | 0.762 | AppliedAmount | 0.777 | +0.015 ** |
| UseOfLoan | 0.760 | ApplicationType | 0.775 | +0.015 ** |
| income_total | 0.761 | nr_of_dependants | 0.776 | +0.015 ** |
| Employment_Duration | 0.762 | LiabilitiesToIncome | 0.775 | +0.014 * |
| home_ownership_type | 0.761 | UseOfLoan | 0.775 | +0.014 ** |
| FreeCash | 0.761 | education_id | 0.774 | +0.014 * |
| Unrestricted | 0.750 | Unrestricted | 0.768 | +0.018 ** |

*Table 4: The summary of the performance of logistic regression with respect to the number of variables included in the model. The order of the inclusion was done according to the Table 3, i.e. first row model contains only one variable (credit_score), second row model contains two variables (credit_score and Interest) etc. The column Diff. calculates the difference between AUC of the woeized model against AUC of the model with dummy variables. The difference marked with * is significant at 95 % level, marked with ** is significant at 99 % level. The statistical significance of the difference between the models was tested by the DeLong&DeLong test. [18]*

The following steps were taken to ascertain the answers to these questions and to deduce some conclusions: With the statistically significant variables identified in the previous section, I started with the most important one (credit_score). Gradually, one by one, other variables were added to the model according to the Table 3. The performance of each models was evaluated using the K-fold cross validation (K = 10) and the respective AUC and its standard deviations were obtained. The performance was then compared with the performance of the unrestricted model.

The results are summarized in Table 4 for the benchmark logistic regression model, Table 5 for SVM with linear kernel and in Table 6 for SVM with gaussian kernel.

| Dummy variables | | WOEisation | | |
|---|---|---|---|---|
| Model | AUC | Model | AUC | Diff. |
| credit_score | 0.627 | credit_score | 0.664 | +0.038 ** |
| Interest | 0.645 | Interest | 0.689 | +0.044 ** |
| PreviousRepayments | 0.644 | ApplicationSignedHour | 0.701 | +0.056 ** |
| Age | 0.683 | PreviousRepayments | 0.701 | +0.017 |
| AmountOfPreviousLoans | 0.707 | home_ownership_type | 0.702 | -0.005 |
| marital_status_id | 0.702 | AmountOfPreviousLoans | 0.729 | +0.026 * |
| language_code | 0.726 | language_code | 0.728 | +0.002 |
| NewLoanMonthlyPayment | 0.726 | marital_status_id | 0.744 | +0.019 * |
| AppliedAmount | 0.730 | Age | 0.748 | +0.018 * |
| nr_of_dependants | 0.729 | occupation_area | 0.750 | +0.021 ** |
| occupation_area | 0.717 | employment_status | 0.755 | +0.039 ** |
| employment_status | 0.725 | NewPaymentToIncome | 0.754 | +0.029 ** |
| VerificationType | 0.730 | NewLoanMonthlyPayment | 0.758 | +0.028 ** |
| NewPaymentToIncome | 0.735 | ApplicationSignedWeekday | 0.758 | +0.023 ** |
| ApplicationType | 0.734 | AppliedAmount | 0.758 | +0.025 ** |
| UseOfLoan | 0.735 | ApplicationType | 0.760 | +0.025 ** |
| income_total | 0.735 | nr_of_dependants | 0.761 | +0.026 ** |
| Employment_Duration | 0.735 | LiabilitiesToIncome | 0.761 | +0.027 ** |
| home_ownership_type | 0.736 | UseOfLoan | 0.762 | +0.026 ** |
| FreeCash | 0.736 | education_id | 0.762 | +0.027 ** |
| Unrestricted | 0.722 | Unrestricted | 0.756 | +0.034 ** |

*Table 5: Performance of the support vector machines model with the linear kernel with the cost parameter $C = 2^{-1}$. The arranging and meaning of the columns are exactly same as in Table 4.*

The benchmark logistic regression model shows a mild benefit from the woeisation of the categorical variables. It is also clear, that the feature selection may be beneficial for LR, as the performance of the unrestricted model as well as the performance of the model with many explanatory variables underperform the models with fewer variables in both cases.

As already mentioned earlier, I had a big trouble trying to make the linear SVM work, especially on the dummy data set. The performance of the model exhibits a chaotic behaviour, it is extremely sensitive to the value of C parameter. This behaviour made me study the problem further, using the artificially created data sets in the controlled environment. From that it seems to me that the support vector machines with linear kernel might be negatively affected with the unbalanced classes, i.e. the erratic/chaotic behaviour appears, when one class has more prominent representation than the other one. This issue was also discussed in Chapter 4.

| Dummy variables | | WOEisation | | |
|---|---|---|---|---|
| Model | AUC | Model | AUC | Diff. |
| credit_score | 0.584 | credit_score | 0.568 | -0.016 |
| Interest | 0.658 | Interest | 0.615 | -0.044 |
| PreviousRepayments | 0.662 | ApplicationSignedHour | 0.617 | -0.044 |
| Age | 0.659 | PreviousRepayments | 0.669 | +0.010 |
| AmountOfPreviousLoans | 0.677 | home_ownership_type | 0.681 | +0.004 |
| marital_status_id | 0.697 | AmountOfPreviousLoans | 0.703 | +0.006 |
| language_code | 0.710 | language_code | 0.718 | +0.008 |
| NewLoanMonthlyPayment | 0.706 | marital_status_id | 0.717 | +0.012 |
| AppliedAmount | 0.725 | Age | 0.726 | +0.001 |
| nr_of_dependants | 0.730 | occupation_area | 0.755 | +0.025 |
| occupation_area | 0.782 | employment_status | 0.743 | -0.039 * |
| employment_status | 0.791 | NewPaymentToIncome | 0.736 | -0.056 ** |
| VerificationType | 0.803 | NewLoanMonthlyPayment | 0.739 | -0.064 ** |
| NewPaymentToIncome | 0.800 | ApplicationSignedWeekday | 0.729 | -0.071 ** |
| ApplicationType | 0.804 | AppliedAmount | 0.729 | -0.075 ** |
| UseOfLoan | 0.791 | ApplicationType | 0.732 | -0.060 ** |
| income_total | 0.792 | nr_of_dependants | 0.751 | -0.041 * |
| Employment_Duration | 0.794 | LiabilitiesToIncome | 0.755 | -0.039 * |
| home_ownership_type | 0.795 | UseOfLoan | 0.775 | -0.021 |
| FreeCash | 0.796 | education_id | 0.769 | -0.028 |
| Unrestricted | 0.831 | Unrestricted | 0.806 | -0.025 * |

*Table 6: Performance of the support vector machines model with the Gaussian kernel with parameters $C = 2^{2.5}, \gamma = 2^{-}3$. The arranging and meaning of the columns are exactly same as in Table 4.*

Despite the fact, that linear SVM and logistic regression are in principle alike and should lead to similar hyperplanes, the logistic regression has almost always beaten the linear support vector machines in terms of AUC performance. My results contradict the theoretical conclusion coming from the Vapnik–Chervonenkis theory, that the support vector machine gives the best linear classifier. While the theory itself is mathematically proven, it seems that the algorithm does not cope well with the negative frictions and data imbalance in the inputs when applied to the real data sets.

My findings are in agreement with the comparative papers of Baesens et al., [42] and [4], which conclude that, on average, the support vector machines with linear kernel perform slightly worse than logistic regression when applied to the real data sets - although they may perform slightly better in some specific cases.

My experiments with the artificial data also support this conclusion: the maximum likelihood algorithm used for the estimation of the logistic regression parameters gives systematically better results than SVM with linear kernel, in cases when the data show high level of random noise and the classes are imbalanced. The separating hyperplanes provided by the LR was in most cases closer to the ideal solution than the hyperplane generated by the linear support vector machines. The difference descends when the data are well separated (i.e. lower level of random noise) and the classes are equally represented; the superiority of the LR results remains, though.

The linear support vector machines seem to be less affected by the number of explanatory variables included in the model, although the difference is not high to distinguish whether the observed phenomenon is real or caused just by a chance. There is a slight benefit in woeisation of the categorical variables, especially in cases where just a few variables are presented in the model. I have a reason to believe that this is not an artefact; when other categorical variables were used as the only explanatory variable, the woeised version was always doing better than the dummy version.

The true power of support vector machines appears in cases, when the input data exhibit some non-linear behaviour. This is probably the only instance, when SVM can beat the logistic regression as it can produce the separating hyperplane of the shapes that logistic regression simply cannot.

According to my results, this appears to be the case of the Bondora credit scoring data set. The performance of the SVM with RBF was evidently higher than the performance of the benchmark model, with a slightly preference of the dummy variables to the woeised ones.

Again, the AUC increases with the number of explanatory variables and the unrestricted model shows the highest performance among other models. Should it be a general issue, the feature selection process would actually do more harm than good, but I did not perform enough experiments with other data sets to make such a strong conclusion.

My results in the case of non-linear kernel kind of contradict the prevailing conclusions in the literature, namely [5], [42]. Gaussian kernel is seldom reported as being significantly better than the linear kernel and/or the logistic regression, pointing to the logical conclusion that the credit data are usually linear or just mildly non-linear. However, according to Table 6, the superiority of the RBF to the linear classifiers persists without regard to the variables included to the model, the data subset or the time subset of the original inputs and seem to be a real issue in this specific case. Also, my finding is in agreement with the results in [56] where the author used SVM (with linear and non-linear kernel) and LR to evaluate a credit data set from other peer-to-peer lending platform, Lending Club.

| | | LR | | SVM-L | | SVM-R | |
|---|---|---|---|---|---|---|---|
| | | Dummy | Woe | Dummy | Woe | Dummy | Woe |
| LR | Dummy | | +0.015<br>6.1844<br>(.0129) | -0.027<br>1.9807<br>(.1593) | -0.002<br>1.1057<br>(.2930) | +0.069<br>16.9649<br>(.0000) | +0.044<br>5.4346<br>(.0197) |
| | Woe | | | -0.042<br>12.109<br>(.0005) | -0.017<br>3.4913<br>(.0617) | +0.054<br>6.2367<br>(.0125) | +0.029<br>0.8783<br>(.3487) |
| SVM-L | Dummy | | | | +0.025<br>4.8055<br>(.0284) | +0.096<br>22.1021<br>(.0000) | +0.071<br>8.2049<br>(.0042) |
| | Woe | | | | | +0.071<br>9.2264<br>(.0024) | +0.046<br>2.5450<br>(.1106) |
| SVM-R | Dummy | | | | | | -0.025<br>4.3432<br>(.0372) |
| | Woe | | | | | | |

*Table 7: The statistical tests of the differences in performance among the 6 models considered in this thesis. Each cell contains 3 rows: the difference between AUC of the model specified in the heading minus AUC of the model specified on the left-hand side of the table, the respective $\chi^2$ statistics and the p-value. Statistical test from [18] was used.*

The differences among the models is summarized in Table 7, for each of the 6 model classes, the best performing model with lowest possible variables was selected (15 variables for Logistic regression, 16 variables for linear SVM and unrestricted models for SVM with Gaussian kernel). Two main conclusions can be done according to this table:

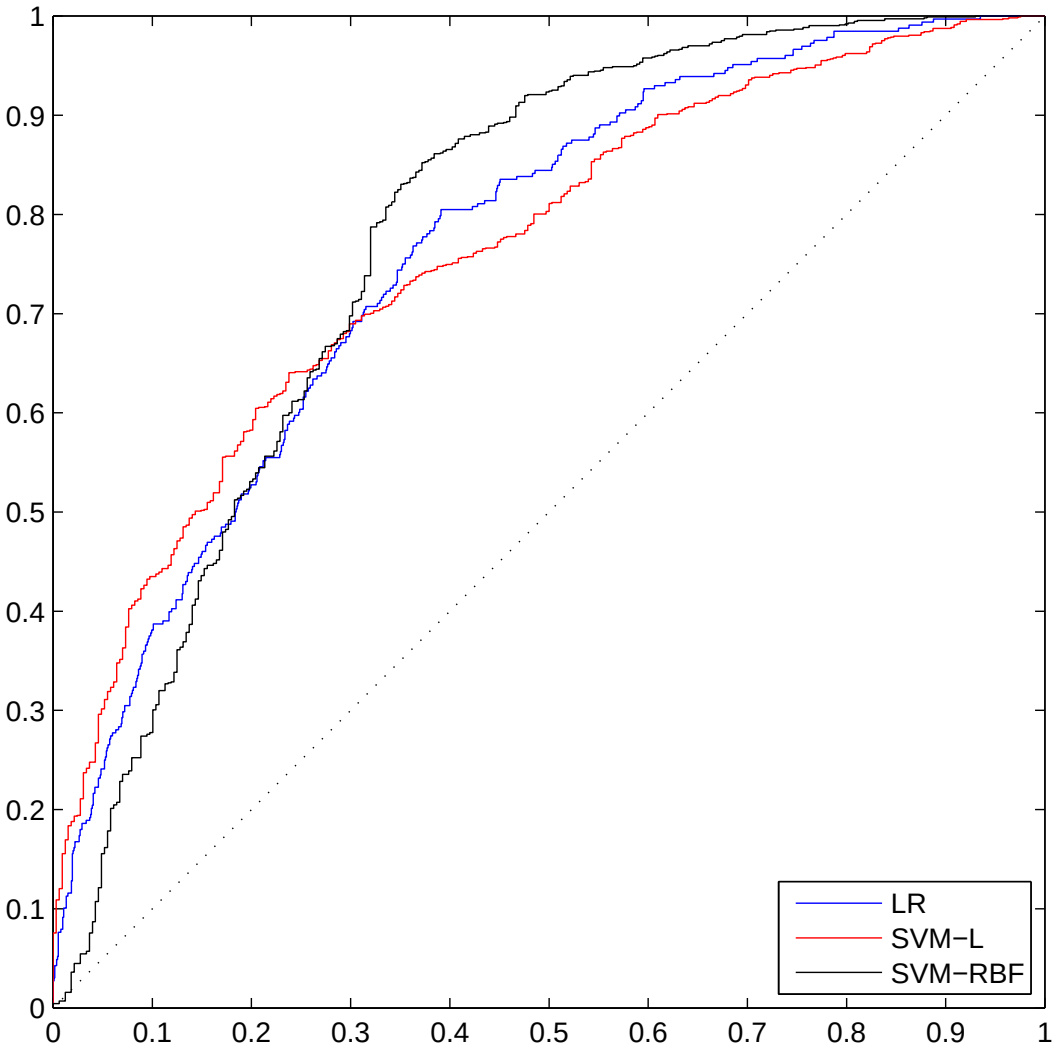


*Figure 8: The comparison of the ROC curves for all the three model classes.*

1. The way how the categorical variables are treated does have influence on the models' performance, in all cases the difference is statistically significant at 95 % level. The direction of this influence, however, is not stable. One cannot say that woeization brings a performance boost against the dummy variables encoding (or vice versa).

2. The SVM-R model with Dummy variables outperforms all the other models at 95 % significance level; apart from woeized LR model it outperforms all of them at 99 % significance level, too.

## 5.7 Quantifying the edge

The mission of this section is to quantify the benefits of using advanced statistical models in investing into the P2P loans. Can the excess performance brought by the now-developed models justify the troubles underwent and time spent building them? Or would be an average investor better off blindly diversifying or counting on the common sense when investing? The backtesting approach was chosen to answer these questions.

Let us focus on three different types of investors:

- RANDOM - Randomly chooses his investments, relying on nothing than diversification effect of a large portfolio of loans and the initial screening process done by the operator of the platform itself.

- NAIVE - The investments are chosen by looking at a couple of simple attributes of the debtor, based on common sense rather than strict mathematical model. This investor will go only for loans, that have: Credit Group A, Credit Score 900-1000, Debt-to-income ratio $\leq 59\%$ and are willing to pay 22 percent interest, or higher. This investor mimics my own behaviour in the beginning of the platform, when no historical data were available.

- FORMAL - Investments will be chosen based on the best mathematical model available. In this case it will be the Support vector machines with Gaussian kernel, unrestricted, with the categoricals encoded using the dummy variables. The model is trained for each run using a subset of the data. Then it can invest only in the loans set aside as a test sample. Logistic regression model as a benchmark is used, too.

The backtest was performed using the Monte Carlo simulation approach with 10 000 iterations. The initial investment of 1 000 EUR was allocated in different loans, 10 EUR each. The ROI observed after 1 year of investing and the risk profile of the portfolio (measured as Value at Risk, VaR) were used to evaluate successfulness of the aforementioned strategies. For the sake of the feasibility of this experiment, some simplifying assumptions were made:

- All investments are available for bidding at the beginning and the initial amount can be spread among loans immediately. This was not true when the platform started, but it is a reasonable assumption for the recent months and years.

- The data set is homogeneous enough with respect to the time (i.e. the cohort of loans provided in 2009 has same properties and behaviour as the cohort from 2013). Although this may sound as a strong assumption, my informal analysis does not indicate the presence of sudden changes in the data set and even the default probability was stable throughout the years.

- The investor is in the position of the pure market taker - his investments do not influence the final interest rate of the loan resulting from the auction and have no effect on whether the loan will be provided or fails due to the lack of total money gathered from the investors. Given the low amount per one loan, this assumption is reasonable.

- The defaulted loan is sold on the secondary market right before the default occurs, with the 40 % discount from the current principal. Given the general recovery ratio and my experience, this assumption is also reasonable. The money is then reinvested into another loan according to the respective investment strategy.

- The revenues are reinvested at the end of each month to other loans that satisfy the strategy rules.

- After a year, all cash flows are summed up, the current loans are valued at the remaining principal and the return on investment based on the internal rate of return is calculated.

Table 8 shows the results of the backtesting. The amount of loans in portfolio (100 at the beginning, and rising as the revenues are further reinvested) seems to provide a sufficient diversification against a potential loss. The Random strategy is the only one that shows negative VaR(1yr, 1%).

There is a clear pattern in the table: with more sophisticated strategy, there is a higher expected return (profit) and lower risk undertaken. While the Naive strategy outperforms the Random strategy in terms of yield and risk, the formalized approach using either LR model or SVM model tends to bring additional risk-adjusted returns.

Despite the fact that process of development of such model is laborious[35] The differences between the strategies are, however, rather small as an absolute value. Therefore, the extra effort could be profitable only to those, who are investing or plan to invest higher amounts of money in absolute terms.

One has to bear in mind, that the Table 8 provides a result of simulations based on simplified assumptions. These numbers can not in any case be used as an extrapolation of future. Personally, I expect that the average return will be decreasing for the next couple of years, as the "market" inside the platform will become more efficient and the operator of the platform will push for higher profits for them, thus wiping off the "early adopters premium"[36] which could have been realized at Bondora in the past. Such trends could have been observed at older p2p platforms (like Lending Club) in the previous years, too.

| | P/L after 1 year (EUR) | | | | |
|---|---|---|---|---|---|
| Strategy | Mean | St.dev. | $VaR_{0.05}$ | $VaR_{0.01}$ | ROI |
| Random | 45.63 | 24.56 | 5.05 | -12.61 | 4.74 % |
| Naive | 105.50 | 19.13 | 73.98 | 60.49 | 10.97 % |
| LR | 146.77 | 15.04 | 121.60 | 111.29 | 15.25 % |
| SVM-R | 172.41 | 13.97 | 149.17 | 139.60 | 17.91 % |

*Table 8: The profit and ROI on 1 year horizon with the initial investment 1 000 EUR spread into 10 EUR loans with different strategies as per the Monte-Carlo simulation with 10 000 independent runs. The standard deviation and the 1-year 5% and 1% Value at Risk as the measures of risk are calculated.*

[35]We are talking about months rather than weeks under amateur investors' conditions without previous experience in this field.

[36]as I use to call it

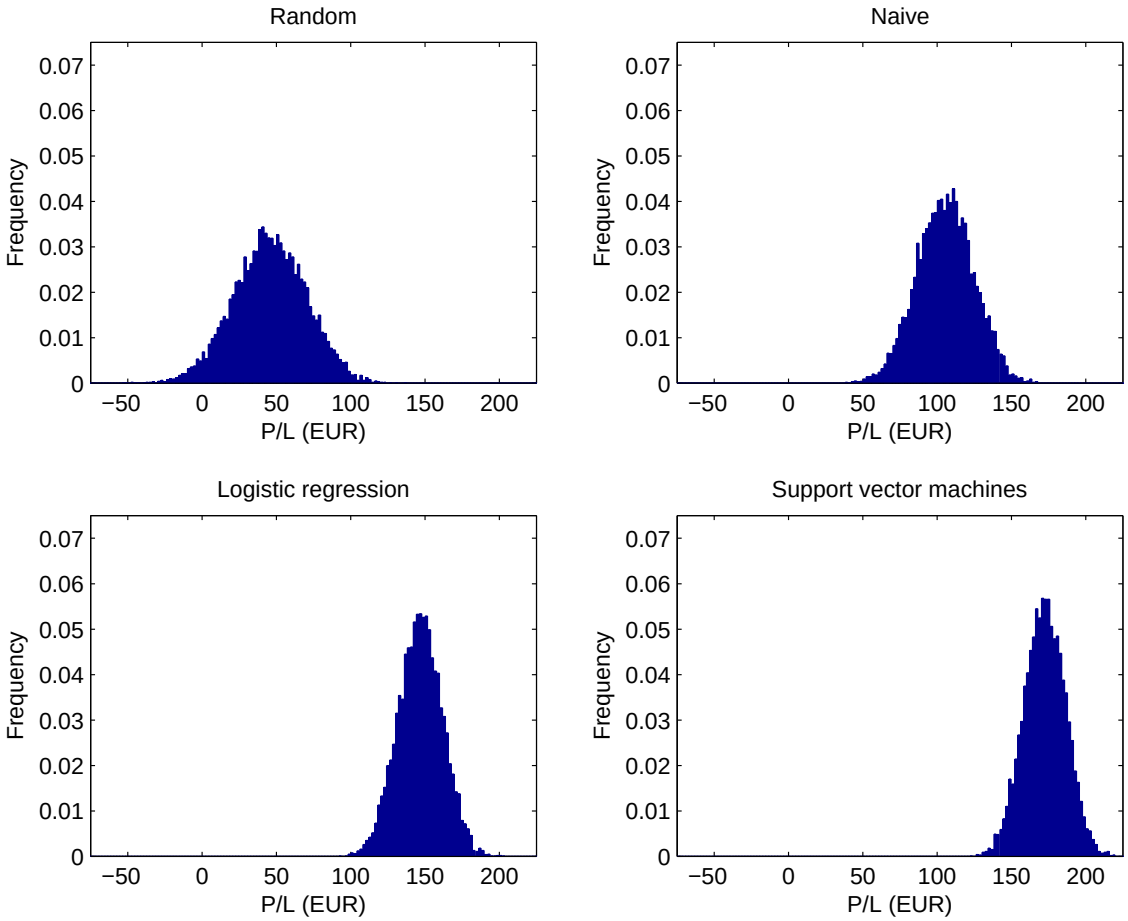


*Figure 9: The distribution of the P/L in EUR after 1 year, according to the 10 000 backtesting simulations.*

# Conclusion

With the advance of new technologies and investment possibilities, the statistical or machine learning methods, once reserved exclusively to the professional financial institutions, can be also beneficial to the amateur investors.

The method of support vector machines as an alternative to the conservative logistic regression models was studied and its performance compared on the real credit data sets. Especially in combination with the non-linear kernel, SVM proved itself as a competitive approach and provided a slight edge on top of the logistic regression model.

The cost for this is much higher computational time, which was needed for the finding of the optimal parameters of the kernel function in particular. The process of model development was time consuming, as well. Partly because of the necessity to study the subject thoroughly, since SVM is not as notorious method as LR. Partly because of lower support of support vector machines in the environment I chose for my models' development.

The extra performance brought by the support vector machines can not be considered as an argument for replacing the well established logistic regression. The professional institutions are bound by the strict regulatory rules and the extra performance is not high enough to outweigh the potential model risk: there is simply not enough incentive for regulators and traditional institutions to replace a model that worked so well for decades.

Since the amateur investors and private funds are not bound by the regulatory rules, my results could indicate that SVM may be an interesting alternative for them. Nevertheless, as the background research from the cited scientific papers shows, support vector machines' performance tend to be less stable and reliable in general. While it performs well on some data sets (as in my case), it gets beaten by the logistic regression on average. [42]

There are other arguments against the real-life application of the support vector machines in credit scoring. The massively cited comparison paper of Baesens et al. from 2013 clearly shows that individual classifiers (where LR along with SVM belong to) have reached their performance limits years ago and are no longer at the center of attention of the researchers. Baesens' team goes even further, when they recommend to stop using logistic regression as an etalon in the future scientific research and replace it by the new generation of algorithms (namely Random Forests).

One can therefore argue, that private subjects will tend to adopt the current state-of-the art classifications algorithms. Some of them were briefly discussed in the Chapter 4 as the potentially interesting topics for further research.

Other, more serious argument against SVM, comes out from the fact that it is very hard to use it as a standalone method. Despite a serious effort, I was not able to implement a reliable feature selection process according to the techniques recommended in the literature (which were summarized in Section 3.2). Eventually, I had to divert to the alternative approach, using the rigorous statistical tests in connection with the logistic regression, to select the most statistically significant explanatory variables to build the model. More importantly, this is the only method explicitly mentioned in the literature available to me, to *quantify* the effect of each variable in support vector machines model. This disadvantage is slightly balanced by the fact, that SVM tend to perform better in the form of the unrestricted model.

Despite all the facts against it, support vector machines remain an important concept from the educational and theoretical point of view. They also formed a history of machine learning, as it was the first method which was able to compete with human in the recognition of the handwritten numbers and they inspired many subsequent research. Their use in credit scoring specifically is not, however, without problems and cannot be recommended in real applications, unless another major breakthrough further increases its performance or reliability.

# Bibliography


[1] Akbani, R., Kwek, S., and Japkowicz, N. Applying support vector machines to imbalanced datasets. In *Machine Learning: ECML 2004*. Springer, 2004, pp. 39–50.

[2] Bache, K., and Lichman, M. Iris data set, 2013. Available at `http://archive.ics.uci.edu/ml/datasets/Iris`.

[3] Bache, K., and Lichman, M. Statlog (australian credit approval) data set, 2013. Available at `http://archive.ics.uci.edu/ml/datasets/Statlog+%28Australian+Credit+Approval%29`.

[4] Baesens, B., Van Gestel, T., Viaene, S., Stepanova, M., Suykens, J., and Vanthienen, J. Benchmarking state-of-the-art classification algorithms for credit scoring. *Journal of the Operational Research Society 54*, 6 (2003), 627–635.

[5] Bellotti, T., and Crook, J. Support vector machines for credit scoring and discovery of significant features. *Expert Systems with Applications 36*, 2 (2009), 3302–3308.

[6] Berger, S., and Gleisner, F. Emergence of financial intermediaries in electronic markets: The case of online p2p lending. *BuR Business Research Journal 2*, 1 (2009).

[7] Brown, I., and Mues, C. An experimental comparison of classification algorithms for imbalanced credit scoring data sets. *Expert Systems with Applications 39*, 3 (2012), 3446–3453.

[8] Canals-Cerdá, J. J., and Kerr, S. Forecasting credit card portfolio losses in the great recession: a study in model risk. Tech. rep., 2014.

[9] Chang, C.-C., and Lin, C.-J. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology 2* (2011), 27:1–27:27. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[10] Chen, Y.-W., and Lin, C.-J. Combining svms with various feature selection strategies. In *Feature Extraction*. Springer, 2006, pp. 315–324.

[11] Ciresan, D., Meier, U., and Schmidhuber, J. Multi-column deep neural networks for image classification. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on* (2012), IEEE, pp. 3642–3649.

[12] Cortes, C., and Vapnik, V. Support-vector networks. *Machine learning 20*, 3 (1995), 273–297.

[13] Cover, T. M. Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *Electronic Computers, IEEE Transactions on*, 3 (1965), 326–334.

[14] Cristianini, N., and Shawe-Taylor, J. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, 2000.

[15] Cronin, S. Modern analytics: A look at the smo. part 1 of a week long series., September 2010.

[16] Crook, J. N., Edelman, D. B., and Thomas, L. C. Recent developments in consumer credit risk assessment. *European Journal of Operational Research 183*, 3 (2007), 1447–1465.

[17] de Souza, C. R. Kernel functions for machine learning applications, March 2010.

[18] DeLong, E. R., DeLong, D. M., and Clarke-Pearson, D. L. Comparing the areas under two or more correlated receiver operating characteristic curves: a nonparametric approach. *Biometrics* (1988), 837–845.

[19] Dumais, S., Platt, J., Heckerman, D., and Sahami, M. Inductive learning algorithms and representations for text categorization. In *Proceedings of the seventh international conference on Information and knowledge management* (1998), ACM, pp. 148–155.

[20] Elizondo, D. The linear separability problem: Some testing methods. *Neural Networks, IEEE Transactions on 17*, 2 (2006), 330–344.

[21] Engelmann, B., Hayden, E., and Tasche, D. Measuring the discriminative power of rating systems. Tech. rep., Discussion paper, Series 2: Banking and Financial Supervision, 2003.

[22] Finlay, S. *Credit Scoring, Response Modeling, and Insurance Rating: A Practical Guide to Forecasting Consumer Behavior*. Palgrave Macmillan, 2012.

[23] Fisher, R. A. The use of multiple measurements in taxonomic problems. *Annals of eugenics 7*, 2 (1936), 179–188.

[24] Freedman, S., and Jin, G. Z. Do social networks solve information problems for peer-to-peer lending? evidence from prosper. com. Tech. rep., com .NET Institute Working Paper, 2008.

[25] Ghodselahi, A. A hybrid support vector machine ensemble model for credit scoring. *International Journal of Computer Applications 17* (2011).

[26] Goldberg, D. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Artificial Intelligence. Addison-Wesley, 1989.

[27] Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. Gene selection for cancer classification using support vector machines. *Machine learning 46*, 1-3 (2002), 389–422.

[28] Hamel, L. H. *Knowledge discovery with support vector machines*, vol. 3. John Wiley & Sons, 2011.

[29] Hand, D. J. Evaluating diagnostic tests: the area under the roc curve and the balance of errors. *Statistics in Medicine 29*, 14 (2010), 1502–1510.

[30] Hand, D. J., and Anagnostopoulos, C. When is the area under the receiver operating characteristic curve an appropriate measure of classifier performance? *Pattern Recognition Letters 34*, 5 (2013), 492–495.

[31] Hans, H. Statlog (german credit data) data set, 2013. Available at `http://archive.ics.uci.edu/ml/datasets/Statlog+%28German+Credit+Data%29`.

[32] Hardy, D. *Regulatory Capture in Banking*. IMF working paper. INTERNATIONAL MONETARY FUND, 2006.

[33] Hastie, T., Tibshirani, R., and Friedman, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition*. Springer Series in Statistics. Springer, 2009.

[34] Haykin, S. *Neural Networks and Learning Machines*. No. sv. 10 in Neural networks and learning machines. Prentice Hall, 2009.

[35] Hsu, C.-W., Chang, C.-C., Lin, C.-J., et al. A practical guide to support vector classification, 2003.

[36] Huang, C.-L., Chen, M.-C., and Wang, C.-J. Credit scoring with a data mining approach based on support vector machines. *Expert Systems with Applications 33*, 4 (2007), 847–856.

[37] isePankur AS. Bondora.com data export. Available at `https://www.bondora.com/en/invest/statistics/data_export`.

[38] Jebara, T. *Machine Learning: Discriminative and Generative*. The Springer International Series in Engineering and Computer Science. Springer US, 2004.

[39] Kim, E. Everything you wanted to know about the kernel trick (but were too afraid to ask). `http://www.eric-kim.net/eric-kim-net/posts/1/kernel_trick.html`, September 2013.

[40] Kreyszig, E. *Advanced Engineering Mathematics*. John Wiley & Sons, 2010.

[41] Lending Club. Individual retirement accounts iras can provide tax advantaged growth, 2014. Available online at `https://www.lendingclub.com/public/individual-retirement-accounts.action`.

[42] Lessmann, S., Seowb, H.-V., Baesens, B., and Thomas, L. C. Benchmarking state-of-the-art classification algorithms for credit scoring: A ten-year update. In *Credit Research Centre, Conference Archive* (2013).

[43] Li, J., Liu, J., Xu, W., and Shi, Y. Support vector machines approach to credit assessment. In *Computational Science-ICCS 2004*. Springer, 2004, pp. 892–899.

[44] Ng, A. Lecture notes. CS 229: Machine learning. *Stanford University* (2003).

[45] Ng, A. Machine Learning (Stanford) — Lecture 07. `https://www.youtube.com/watch?v=s8B4A5ubw6c`, 2008.

[46] Ng, A. Machine Learning (Stanford) — Lecture 11. https://www.youtube.com/watch?v=sQ8T9b-uGVE, 2008.

[47] Ng, A. Y., and Jordan, M. I. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. *Advances in neural information processing systems 2* (2002), 841–848.

[48] Pender, W., Saddler, D., Shea, J., and Ward, D. *Cambridge 2 Unit Mathematics Year 11 Enhanced Version PDF Textbook*. Cambridge Secondary Maths (Australia) Series. Cambridge University Press, 2012.

[49] Peng, Y. Tikz example – svm trained with samples from two classes, September 2013.

[50] Platt, J. C. Fast training of support vector machines using sequential minimal optimization. In *Advances in kernel methods* (1999), MIT press, pp. 185–208.

[51] Rao, K., and Koolagudi, S. *Emotion Recognition using Speech Features*. SpringerBriefs in Electrical and Computer Engineering. Springer, 2012.

[52] Rummelhart, D. Learning representations by back-propagating errors. *Nature 323*, 9 (1986), 533–536.

[53] Sarle, W. S. Neural Network FAQ, part 2 of 7: Learning. ftp://ftp.sas.com/pub/neural/FAQ2.html.

[54] Schebesch, K. B., and Stecking, R. Support vector machines for classifying and describing credit applicants: detecting typical and critical regions. *Journal of the Operational Research Society 56*, 9 (2005), 1082–1088.

[55] Schölkopf, B., and Smola, A. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Adaptive computation and machine learning. MIT Press, 2002.

[56] Tvaroh, T. Using data mining methods in the analysis of credit risk data. Master's thesis, University of Economics in Prague, 2014. [in Czech language].

[57] Williams, C. *Support Vector Machines*. School of Informatics, University of Edinburgh, October 2008.

[58] Witten, I., Frank, E., and Hall, M. *Data Mining: Practical Machine Learning Tools and Techniques: Practical Machine Learning Tools and Techniques*. The Morgan Kaufmann Series in Data Management Systems. Elsevier Science, 2011.

[59] Witzany, J. *Credit Risk Management and Modeling*. Oeconomica, 2010.

[60] Xia, F. Support vector machine, 2008.

# A. Appendix

## Logistic regression: dummy variables

| Variable | DF | Wald $\chi^2$ | p | Gini |
|---|---|---|---|---|
| Credit score | 6 | 108.8197 | 0.0000 | 0.3205 |
| Interest | 1 | 57.1851 | 0.0000 | 0.4137 |
| PreviousRepayments | 1 | 24.3019 | 0.0000 | 0.1604 |
| Age | 1 | 23.3031 | 0.0000 | 0.1534 |
| AmountOfPreviousLoans | 1 | 22.3896 | 0.0000 | 0.0207 |
| Marital status | 4 | 27.6078 | 0.0000 | 0.0941 |
| VerificationType | 2 | 20.4638 | 0.0000 | 0.2248 |
| NewLoanMonthlyPayment | 1 | 15.5639 | 0.0001 | 0.2772 |
| NewPaymentToIncome | 1 | 11.4037 | 0.0007 | 0.2615 |
| nr_of_dependants | 1 | 8.7477 | 0.0031 | 0.0058 |
| UseOfLoan | 8 | 21.1369 | 0.0068 | 0.0658 |
| Occupation | 19 | 36.4364 | 0.0093 | 0.1249 |
| Employment | 5 | 14.5231 | 0.0126 | 0.1214 |
| AppliedAmount | 1 | 5.7889 | 0.0161 | 0.1952 |
| ApplicationType | 1 | 4.2583 | 0.0391 | 0.0773 |

*Table 9: The final logistic regression model built from 15 explanatory variables. Categorical variables were encoded using the dummy variables. The Wald $\chi^2$ statistics and the p-value of the total effect of the variable are summarized in 3rd and 4th columns. The last column contains the Gini coefficient for each variable from the univariate analysis.*

| Variable | DF | Estimate | SE | Wald $\chi^2$ | p |
|---|---|---|---|---|---|
| Intercept | 1 | -0.6409 | 0.7997 | 0.6423 | 0.4229 |
| Interest | 1 | 0.4643 | 0.0614 | 57.1851 | 0.0000 |
| PreviousRepayments | 1 | -0.9138 | 0.1854 | 24.3019 | 0.0000 |
| Age | 1 | -0.3274 | 0.0678 | 23.3031 | 0.0000 |
| AmountOfPreviousLoans | 1 | 0.6262 | 0.1323 | 22.3896 | 0.0000 |
| NewLoanMonthlyPayment | 1 | -0.4786 | 0.1213 | 15.5639 | 0.0001 |
| AppliedAmount | 1 | 0.2147 | 0.0893 | 5.7889 | 0.0161 |
| nr_of_dependants | 1 | -0.1837 | 0.0621 | 8.7477 | 0.0031 |
| NewPaymentToIncome | 1 | 0.3794 | 0.1123 | 11.4037 | 0.0007 |
| Credit score = 1000 | 1 | -0.4408 | 0.2070 | 4.5362 | 0.0332 |
| Credit score = 800 | 1 | -0.1349 | 0.2813 | 0.2299 | 0.6316 |
| Credit score = 700 | 1 | 0.6146 | 0.2740 | 5.0317 | 0.0249 |
| Credit score = 600 | 1 | -0.0502 | 0.2588 | 0.0376 | 0.8462 |
| Credit score = 500 | 1 | 0.9316 | 0.2273 | 16.7974 | 0.0000 |
| Credit score = empty | 1 | 0.1902 | 0.7848 | 0.0588 | 0.8085 |
| Marital status = Married | 1 | -1.4145 | 0.3538 | 15.9883 | 0.0001 |
| Marital status = Cohabitant | 1 | -1.8084 | 0.3672 | 24.2555 | 0.0000 |
| Marital status = Single | 1 | -1.5172 | 0.3744 | 16.4220 | 0.0001 |
| Marital status = Divorced | 1 | -1.4623 | 0.3785 | 14.9280 | 0.0001 |
| Occupation = empty | 1 | -1.0717 | 0.4980 | 4.6318 | 0.0314 |
| Occupation = Other | 1 | 0.0767 | 0.2483 | 0.0954 | 0.7574 |
| Occupation = Telecom | 1 | -0.1106 | 0.2926 | 0.1430 | 0.7053 |
| Occupation = Finance | 1 | 0.3766 | 0.3166 | 1.4142 | 0.2344 |
| Occupation = Real-estate | 1 | 0.6374 | 0.6939 | 0.8437 | 0.3583 |
| Occupation = Research | 1 | -0.7243 | 0.6831 | 1.1243 | 0.2890 |
| Occupation = Administrative | 1 | 0.3894 | 0.5114 | 0.5798 | 0.4464 |
| Occupation = Civil service & military | 1 | 0.7060 | 0.2994 | 5.5589 | 0.0184 |
| Occupation = Education | 1 | -0.2217 | 0.3096 | 0.5127 | 0.4740 |
| Occupation = Healthcare | 1 | 0.5409 | 0.3292 | 2.6990 | 0.1004 |
| Occupation = Art/entertainment | 1 | 0.3496 | 0.4096 | 0.7285 | 0.3934 |
| Occupation = Agriculture | 1 | 0.5361 | 0.3406 | 2.4771 | 0.1155 |
| Occupation = Mining | 1 | 2.3646 | 1.2185 | 3.7660 | 0.0523 |
| Occupation = Processing | 1 | 0.0232 | 0.2770 | 0.0070 | 0.9331 |
| Occupation = Energy | 1 | -0.3527 | 0.4661 | 0.5726 | 0.4492 |
| Occupation = Utilities | 1 | -0.3805 | 0.8587 | 0.1964 | 0.6577 |
| Occupation = Construction | 1 | 0.1950 | 0.2997 | 0.4234 | 0.5153 |
| Occupation = Retail/wholesale | 1 | -0.0209 | 0.2822 | 0.0055 | 0.9410 |
| Occupation = Transport | 1 | 0.1994 | 0.3157 | 0.3990 | 0.5276 |
| VerificationType = Phone | 1 | 1.0354 | 0.2337 | 19.6264 | 0.0000 |
| VerificationType = Income verified | 1 | 0.8578 | 0.2434 | 12.4161 | 0.0004 |
| ApplicationType = Timed funding | 1 | 0.2747 | 0.1331 | 4.2583 | 0.0391 |
| UseOfLoan = Loan consolidation | 1 | 0.3533 | 0.2468 | 2.0488 | 0.1523 |
| UseOfLoan = Real estate | 1 | 0.0374 | 0.3792 | 0.0097 | 0.9215 |
| UseOfLoan = Home improvement | 1 | 0.6109 | 0.2430 | 6.3216 | 0.0119 |
| UseOfLoan = Business | 1 | 0.5974 | 0.3791 | 2.4837 | 0.1150 |
| UseOfLoan = Education | 1 | 0.9645 | 0.3086 | 9.7692 | 0.0018 |

| Variable | DF | Estimate | SE | Wald $\chi^2$ | p |
|---|---|---|---|---|---|
| UseOfLoan = Travel | 1 | 0.1772 | 0.3745 | 0.2240 | 0.6360 |
| UseOfLoan = Vehicle | 1 | 0.6574 | 0.2500 | 6.9123 | 0.0086 |
| UseOfLoan = Other | 1 | 0.2459 | 0.2291 | 1.1520 | 0.2831 |
| Employment = empty | 1 | -0.4164 | 0.6641 | 0.3932 | 0.5306 |
| Employment = Partially employed | 1 | -0.4944 | 0.6663 | 0.5506 | 0.4581 |
| Employment = Fully employed | 1 | -0.5997 | 0.6310 | 0.9034 | 0.3419 |
| Employment = Self-employed | 1 | -0.8576 | 0.7325 | 1.3708 | 0.2417 |
| Employment = Entrepreneur | 1 | -1.7734 | 0.7127 | 6.1908 | 0.0128 |

*Table 10: Maximum likelihood estimates for the model from Table 9. Missing values of the categorical variables are: Credit score=900, Marital status=Widowed, Occupation=Hospitality and catering, Verification type=Income and expenses verified, Application type=Quick funding, UseOfLoan=Health, Employment=Retiree - the estimates for these values can be derived from the estimates stated in the table.*

# Logistic regression: woeised categorical variables

| Variable | DF | Wald $\chi^2$ | p | Gini |
|---|---|---|---|---|
| Credit score | 1 | 100.9309 | 0.0000 | 0.3431 |
| Interest | 1 | 49.7779 | 0.0000 | 0.4137 |
| Home ownership | 1 | 31.2436 | 0.0000 | 0.2412 |
| ApplicationSignedHour | 1 | 27.1686 | 0.0000 | 0.0830 |
| PreviousRepayments | 1 | 23.4446 | 0.0000 | 0.1604 |
| Language code | 1 | 22.5242 | 0.0000 | 0.0942 |
| AmountOfPreviousLoans | 1 | 21.7680 | 0.0000 | 0.0207 |
| Marital status | 1 | 16.6359 | 0.0000 | 0.0941 |
| Age | 1 | 12.2566 | 0.0005 | 0.1534 |
| Employment status | 1 | 10.8514 | 0.0010 | 0.1214 |
| Occupation | 1 | 10.3915 | 0.0013 | 0.2022 |
| NewLoanMonthlyPayment | 1 | 8.0873 | 0.0045 | 0.2772 |
| NewPaymentToIncome | 1 | 7.1446 | 0.0075 | 0.2615 |
| AppliedAmount | 1 | 4.2468 | 0.0393 | 0.1952 |
| ApplicationSignedWeekday | 1 | 4.0817 | 0.0434 | 0.0732 |

*Table 11: The final logistic regression model built from 15 explanatory variables. Categorical variables were transformed to the real-valued variables using the Weight of Evidence calculation.*

| Variable | DF | Estimate | SE |
|---|---|---|---|
| Intercept | 1 | -1.6810 | 0.0628 |
| Interest | 1 | 0.3915 | 0.0555 |
| PreviousRepayments | 1 | -0.8528 | 0.1761 |
| AmountOfPreviousLoans | 1 | 0.5789 | 0.1241 |
| Age | 1 | -0.1808 | 0.0516 |
| NewPaymentToIncome | 1 | 0.2598 | 0.0972 |
| NewLoanMonthlyPayment | 1 | -0.3149 | 0.1107 |
| AppliedAmount | 1 | 0.1689 | 0.0820 |
| Credit score | 1 | -0.4709 | 0.0469 |
| ApplicationSignedHour | 1 | -0.5922 | 0.1136 |
| Home ownership | 1 | -0.5077 | 0.0908 |
| Language code | 1 | -0.7404 | 0.1560 |
| Marital status | 1 | -0.2080 | 0.0510 |
| Occupation | 1 | -0.1656 | 0.0514 |
| Employment | 1 | -0.1916 | 0.0582 |
| ApplicationSignedWeekday | 1 | -0.1017 | 0.0504 |

*Table 12: Maximum likeliehood estimates of the model from Table 11.*