Report of graduation internship

Université de Technologie de Compiègne

Implementing Machine Learning Methods In Credit Risk Management

Author: Zinan ZHOU

Supervisor: Prof Dritan NACE

Tutor: Prof Nikolaos LIMNIOS

Acknowledgements

I would like to express my deepest appreciation to all those who provided me the possibility to complete this report. Special gratitude to my project manager, Prof.NACE Dritan[1], for his patient guidance, enthusiastic encouragement and constructive suggestions during the planning and development of this research work. His willingness to give his time so generously has been very much appreciated.

Furthermore, I would also like to acknowledge with appreciation the crucial role of Prof.LEGER Jean-Benoist[2], who gave valuable technical support and suggestions for programming. A special thanks goes to my team mate, Ms.FEJZA Doris[3] and Mr.ZHANG Haifei[4], who helped me to assemble the parts and gave suggestions. I have to appreciate the guidance given by other professors in our project presentation that has improved our presentation skills thanks to their comments and advice. Finally, thanks to the data provided by the bank out of France.

[1]Professeur, HEUDIASYC - UMR CNRS 7253 Université de technologie de Compiègne
[2]Professeur, HEUDIASYC - UMR CNRS 7253 Université de technologie de Compiègne
[3]Intern, HEUDIASYC - UMR CNRS 7253 Université de technologie de Compiègne
[4]Intern, HEUDIASYC - UMR CNRS 7253 Université de technologie de Compiègne

Laboratory Heudiasyc

My graduation internship is in Laboratory Heudiasyc. It is created in 1981 and associated with the CNRS since its creation. The Heudiasyc unit (Heuristics and Diagnosis of Complex Systems, UMR-CNRS 7253) is attached to INS2I (Institute of Information Sciences and their interactions) and operates in the field of information and digital sciences, in particular computer science, automation, robotics and artificial intelligence.

The scientific project developed within Heudiasyc is based on the synergy between upstream research and finalized research, to respond to major societal challenges: mobility, transportation, communication and security.

Heudiasyc's scientific activity is organized around three teams:

- **CID: Knowledge, Uncertainties, Data**
  CID's research focuses on statistical learning, uncertainty management and knowledge engineering with applications to knowledge capitalization, path recommendation, and scripting of virtual environments.
- **SCOP: Safety, Communication, Optimization**
  Motivated by the increasingly complex interactions between systems, SCOP's work focuses on the design and optimization of logistics or network systems and safe and secure systems. The scientific themes of the team revolve around issues of (operational) safety, security, optimization and their interactions. The field of application concerns complex systems and systems of systems.
- **SyRI: Robotic Systems in Interaction**
  The objective of the research carried out in the SyRI team is to develop embedded systems to increase the autonomy capacities of mobile robots, evolving in open and complex environments, which can be in interaction with human operators or in mutual interaction with d 'other robots. The flagship applications are mini aerial drones and autonomous cars.

The subject of my project involves both data processing and machine learning, as well as optimization theory. My project manager Prof.NACE Dritan, he is a member of the SCOP team, he gave me strong support on optimization aspects. At the same time, the discussions with Prof.LEGER Jean-Benoist of the CID team provided us with invaluable help in data engineering and model evaluation.

Abstract

The need for controlling and effectively managing credit risk has developed the various quantitative models by financial institutions and consulting companies. Hence, the growing number of academic studies about credit scoring shows a variety of classification methods applied to discriminate good and bad clients.

This report presents the literature review relating theory and application of binary classification techniques for credit scoring financial analysis. Then, we set out to compare several techniques that can be used in the analysis of imbalanced credit scoring data sets. In a credit scoring context, imbalanced data sets frequently occur as the number of defaulting loans in a portfolio is usually much lower than the number of observations that do not default. As well as using traditional classification techniques such as logistic regression, neural networks and decision trees, this report also explores the Support Vector Machine (SVM) and Random Forest (RF) for loan default prediction, including five SVM-based models (Fuzzy SVM, Bilateral Fuzzy SVM, Least Square Fuzzy SVM, Weighted Least Square SVM and Least Square Bilateral Fuzzy SVM) and three RF-based models (Weighted RF, Balanced RF and RF with SMOTE).

Two real-world credit scoring data sets are used to build classifiers and test their performance. The performance criteria chosen to measure the effect are the precision of the majority class and the minority class, the total accuracy and the receiver operating characteristic curve (AUC).

The results from this empirical study indicate that the random forest classifiers (especially balanced random forest) perform very well in a credit scoring context and are able to cope comparatively well with pronounced class imbalances. We also found that, applying the Principle Component Analysis (PCA) algorithm before SVM classifiers improve the precision of the minority class for these data sets. Among the above five SVM-based models, Least Square Fuzzy SVM and Weighted Least Square SVM have higher precision, faster calculation speed and more robust. However, when faced with a large class imbalance, SVM and SVM-based models perform significantly worse than the Random Forest classifiers.

Contents

Chapter 1

An analytic study of credit risk model and machine learning techniques

1.1 Introduction

Banking sector faces every day a variety of risks such as Strategic risk, Cyber Security risk, Market risk, Liquidity risk, Credit Risk or Operational risk. Out of these, crediting is the biggest risk, especially for commercial banks. Considering the economic importance of the subject, there have been many studies concerning Credit Risk Management, mainly in developing a Credit Scoring system by implementing different Machine Learning (ML) methods. The aim of this system is to classify the individual consumers into good and bad clients as well as to predict the lending capability of companies.

1.2 Background

1.2.1 Credit score

The credit score is a numeric expression measuring people's creditworthiness. In various situations, service suppliers need to evaluate customers' credit history first and then decide whether they will provide the service or not. Traditional credit scores may incorporate a dozen variables. With the rise of the big data, a customer might have hundreds or thousands of information to be considered. However, it is time-consuming to check the entire personal portfolios and generate a credit report manually. Then, alternative methods for credit score using a deeper data analysis has been developed. These methods are in the focus of this work.

1.2.2 Machine learning based credit model

In order to be able to consider more variables, lenders need new algorithms that are able to handle them and Machine Learning offers a way through that problems. Machine learning benefits the credit lending industry in meanly two ways: improve operational efficiency and make use of new data sources for predicting credit score[1]. Machine learning can find effective information in a large data set to predict the applicant's ability to repay the loan, which is more efficient than traditionally checking personal credit history information. ML-based credit models can consider unknown data points to furthermore fully predict the likelihood that the borrower will repay the loan.

[1]Simon Kostadinov: My Analysis from 50+ papers on the Application of ML in Credit Lending. http://towardsdatascience.com/my-analysis-from-50-papers-on-the-application-of-credit-lending-b9b810a3f38

With machine learning predictions, lenders can lend to applicants with incomplete information without increasing risk. On the other hand, ML-based credit models can detect the more nuanced gap between borrowers, enable more accurate risk-based pricing. As a result, the lender can provide different interest rates for different borrowers to increase the lender's income. For unknown applicants and young people with no history credits, they will have a greater chance to apply for loans.

Machine Learning algorithms are used in various ways. The best performing classification algorithms on the credit scoring problem vary based on the specific data set, features selection and more. Support Vector Machines (SVM), Gradient Boosting, Tree-based algorithms (e.g. Random Forest) are recognized as good methods. However, applying ML algorithm to determine credit score has some limitations. Rather than the algorithm's poor performance or lack of sophistication, the limitations meanly come from biased data, interpretability, regulations and scalability [2].

1.3 Literature review

1.3.1 Techniques

In this section we describe various methods used over the years for modeling the credit scoring problem, analyzing both the traditional statistical techniques and the more advanced methods. The main reason for using these advanced methods is their potentiality of modeling exceptionally complex functions, in contrast with traditional models. We have listed their advantages and disadvantages and presented several comparison between them, using results found on the reviewed literature.

Techniques used for credit risk analysis may be categorized in the following groups:

- Statistical methods
  There are many statistical models for credit scores. Linear discriminant analysis, logistic regression, probit regression, k-nearest neighbour, classification tree, etc. Among them, Logistic regression models are commonly used in theoretical research and practical applications because of their many advantages, such as excluding the effects of outliers, suitable for continuous or categorical independent variables, and strong interpretability of calculation results, etc.

- Mathematical Programming methods
  Mathematical Programming methods include linear programming, quadratic programming, integer programming, etc. When the incorporation of a side condition becomes necessary, or a small sample size is available, or a large number of explanatory variables is present, or the data set is heavily contaminated. In these conditions, MP methods might perform better than the statistic discriminant analysis models [43]. The essence of SVM is solving quadratic programming problems.

- Artificial Intelligence techniques
  Artificial intelligence techniques include artificial neural networks, support vector machines, genetic algorithm and genetic programming, rough set, etc. For credit scoring systems, the advantage of AI technique is that they can mine hidden relationships between variables which are not always obvious to traditional credit scoring systems. In addition, as new data is entered into the system, the AI-based credit score model can continuously improve itself. However, the more complex the algorithm, the worse the interpretability from a financial perspective. So we should make a trade off between algorithm choice and interpretability.

[2]McKinsey Quarterly: What AI can and can't do (yet) for your business. https://www.mckinsey.com/business-functions/mckinsey-analytics/our-insights/what-ai-can-and-cant-do-yet-for-your-business

- Hybrid approaches
  Hybrid methods combine two or more different machine learning techniques in order to improve the performance of the model and to avoid weaknesses. These techniques can be data reduction techniques, clustering or classification techniques. For example a hybrid model may combine a data reduction technique and a classifier (rough set and artificial neural network), two classifiers (artificial neural network and support vector machines), etc. Hybrid approaches are also widely used in credit scoring. [25] present a credit scoring model using the hybrid neural discriminant technique.

- Ensemble methods
  Ensemble methods refer to techniques used for combining classifiers. Different authors propose ensemble methods in order to generate more powerful systems [37]. Three popular ensemble methods used are Bagging, Boosting, and Stacking.
  The idea of Bagging is to generate different subsets of the training set, and to build models on each subset separately. The final result is determined by the votes of each model, and the weight of the votes of each model is the same.
  Contrarily, in Boosting the models are built sequentially, each base model depending on the previous ones because with each iteration of boosting, the new base model is updated from the errors of the previous learners.
  Stacking includes two levels, level-0 model and level-1 model. The level-0 model consists in two or more base classifiers (base-learners) for learning and fitting. And the output values of these base classifiers as the input to the next classifier (Meta-learner) for predicting, which is level-1 model. The final results is the output value of the Meta learner.

Discriminant Analysis

Discriminant Analysis is one of the most primitive classification methods used for credit scoring. It was first introduced by Fisher [15] as a technique for solving two class classification problems. Later, it was extended on situation of multi class classification (as known today Linear Discriminant Analysis). This technique projects a p-dimensional feature vector onto a hyperplane that divides the space into two half-spaces, each half-space representing a class [12].

(Artis, 1994) [31] presents a model for decision making in the process of crediting, based on Discriminant Analysis. This model calculates for each client the probability of paying or not paying the money back and transforms it into a score between 50 and 80. This probability is calculated using Bayes' rule. As a result, with the increase of the minimum score allowed for granting a loan, decreases the number of total applications accepted, consequently the number of bad applications accepted.

The main problem with DA is the low classification precision, as a result of the fact that it is primarily developed for detecting linear relationships among variables. [13] describes this and many other problems in applying DA in credit scoring model. Despite of all these problems, discriminant analysis is still a very common used technique in credit scoring [11].

Regression

Regression analysis is a mathematical method used for estimating how an independent variable is related to one or more dependent variables. This method finds applications in all fields and its most popular algorithm is Linear Regression.

- Linear Regression
  Linear Regression has been used in credit scoring even though it is a two class problem. The relationship between the characteristics of one client and the output y is set using this equation:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_i x_i + \epsilon,$$

where β variables are estimated using ordinary least squares and ϵ is the random error. Numerous authors have applied linear regression model in credit scoring [22], [3].

- Logistic Regression
  Contrarily, logistic regression is not a regression but a classification learning algorithm considering the fact that the target output, y, can take only discrete values for given set X of inputs x_i. Its name comes from the similarity of its mathematical formulation with that of Linear Regression. Comparatively as Linear Regression presumes that the data follows a linear function, Logistic Regression uses the sigmoid function for modeling the data:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{1.1}$$

  Logistic Regression has a wide range of applications in many research areas like social science, biomedical studies, marketing as well as financial sector. One application of this model in financial domain is predicting the creditworthiness of clients. If we consider the group of input variables X=$\{x_1, x_2, , , x_i\}$ and the output variable Y= $\{y_1, y_2\}$ the logistic regression method involves estimating a linear combination between X and Y. The main difference between using a logistic regression model and a linear regression model in credit scoring is that the outcome of the credit applicant in logistic regression is binary (not continuous). Some studies using Logistic Regression for credit scoring worth mentioning are: [23], [3], [1]

Decision Trees

Decision Trees are also a classification technique widely used for credit scoring. They are based on construction of tree-like using a set of if-then logical conditions in order to classify the cases. [18] proved that results obtained through Recursive Partitioning Algorithm are better than those obtained using Discriminant Analysis when evaluating credit risk. Decision Trees are easy to understand and implement, however they have a restriction in generalization of results and are not very suitable when handling large amount of variables.

K-nearest-neighbor (KNN)

KNN is a non-parametric statistical technique that is often used as a measure for more complex classifiers. This algorithm consists in predicting a new point as the average of its K nearest neighbors, which are chosen from the training data. Its main disadvantage in credit scoring is that for a better performance it is necessary to have an equal number of good and bad clients. That's why optimized versions of this algorithm are proposed. [30] uses Weighted K Nearest Neighbor, a modified version of KNN, for predicting new clients.
For modeling credit scoring problem, apart from traditional statistical methods like LR or DA, there are also implemented advanced methods like neural networks and genetic programming.

Artificial Neural Networks

Artificial Neural Networks (ANN) are a category of nonlinear regression and discrimination methods. Different models of ANN are implemented in credit scoring, mentioning here radial basis function, multilayer perceptron, learning vector quantization or mixture-of-experts. [39] studies some of these models concluding that Moduls of Elasticity (MOE) and Radial Basis Function (RBF) should be considered as potential techniques for credit scoring applications. Other application of ANN in credit scoring could be found in [25] and [32]. Even though ANN models have reported to perform better credit scoring accuracy than those using discriminant analysis and logistic regression, ANN has, however, the disadvantage of its long training process for the obtainment of the optimal network's topology. Another important drawback of ANN is the lack of the interpretability, which is one of the main requirements of bank regulations. All these problems limit the capability of this method in handling general classification and credit scoring problems.

	German credit data[b]			Australian credit data[b]		
	Good credit	Bad credit	Overall	Good credit	Bad credit	Overall
Neural models[a]						
MOE	0.1428	0.4775	0.2434	0.1457	0.1246	0.1332
RBF	0.1347	0.5299	0.2540	0.1315	0.1274	0.1286
MLP	0.1352	0.5753	0.2672	0.1540	0.1326	0.1416
LVQ	0.2493	0.4814	0.3163	0.1710	0.1713	0.1703
FAR	0.4039	0.4883	0.4277	0.2566	0.2388	0.2461
Parametric models						
Linear discriminant	0.2771	0.2667	0.2740	0.0782	0.1906	0.1404
Logistic regression	0.1186	0.5133	0.2370	0.1107	0.1409	0.1275
Non-parametric models						
K nearest neighbor	0.2257	0.5533	0.3240	0.1531	0.1332	0.1420
Kernel density	0.1557	0.6300	0.3080	0.1857	0.1514	0.1666
CART	0.2063	0.5457	0.3044	0.1922	0.1201	0.1562

Table 1 - Credit scoring error for average case, source: (West, 2000)

Genetic Programming

Genetic Programming is another technique which has been applied for predicting client's creditworthiness. This technique comes from optimization problems and finds many applications in credit scoring [16]. However, comparing with ANN it shows a lack of precision. In order to solve this problem and ANN deficiencies [42] proposes a combination of both GA and ANN for credit scoring approach.

SVM

SVM is a classification algorithm widely used for solving two-group classification problems. It consists in finding the optimal hyperplane that separates the data points in two classes.

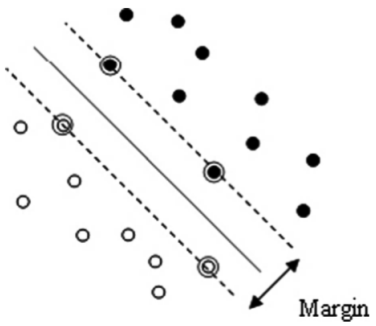

Figure 1.1: *Hyperplane with maximal margin*

SVM has been applied successfully in various classification problems such as image recognition, text categorization as well as credit scoring. SVM was first introduced in 1992 [5], but its applications in financial field , especially in credit scoring started to appear after 2000. (B.Baesens, 2003) [2] implemented SVMs and other classifiers techniques in different credit data sets. In conclusion, SVMs perform well when compared with other methods; however they do not give always the best performance. Baesens also emphasizes that particularities of credit data are extremely connected with decision making. Commonly, the misclassification rate on credit data varies around 20% or 30%.

[34] applies SVM to an amount of credit applicants. They conclude that SVMs perform slightly better than Logistic Regression, but not considerably therefore.

[20] compares the SVM performance on German and Australian credit data set against other methods like BPN (back-propagation neural network), decision trees, and genetic programming. In terms of classification accuracy SVM achieves identical classificatory accuracy. The authors propose a hybrid Genetic Programming-SVM model that can perform at the same time the selection of features and the optimization of parameters.

[7] studied the impact of the imbalanced data on several algorithms. As from the results, SVM performance decreases when the level of imbalance in the data set increases.

Optimized versions of SVM

SVM works very well with balanced datasets but in case of imbalanced data, it is not such effective. This is very important for credit scoring, considering the fact that the "good clients" class is in majority comparing to the "bad clients" class. There exist CIL methods (Class Imbalance Learning) that may solve the problem of imbalanced data but SVM is also very sensitive to noise and outliers. This problem may be reduced by using Fuzzy SVM (FSVM) firstly proposed in [26]. FSVM is an extension of SVM that takes into account the importance of the class training example belongs to. This is realized by assigning a fuzzy membership value to each training example of the dataset. [4] proposes FSVM-CIL to solve both problems of imbalanced data and outliers/noises. FSVM-CIL shows better results than normal FSVM and than normal SVM combined with existing CIL methods like Oversampling, Undersampling, SMOTE (Synthetic Minority Over-sampling Technique).

FSVM has been proved to be very convenient for credit risk evaluation offering both generalization capability and insensitivity to outliers. On the other hand, this method has high computational complexity, making it more difficult to implement. [40] proposes the least square method to reduce this complexity and introduces a new classification method: LS-FSVM. In this model, inequality constraints are replaced with equality constraints transforming the QP problem from where FSVM final solution is derived into a linear problem. LS-FSVM produces good classification results and a lower CPU-time of computations.

[41] proposes a new form of fuzzy SVM called Bilateral-Weighted fuzzy SVM. This method constructs two instances from the original instance, one for the positive class and one for the negative class assigning them with different memberships. For example, the instances detected as outliers are treated as a member of the class they belong to with a large membership and at the same time they are treated as a member of the contrary class they fall in with a small membership. This model offers better generalization ability and more efficient use of the training sample.

Random Forest

Random forests are defined as a group of un-pruned classification or regression trees, trained on the training data using random feature selection in the process of tree generation. After a large number of trees have been generated, each tree votes for the most popular class[6]. The result of Random Forest is the majority class of Decision Tree.

A decision tree is a tree-like structure (fig 1.2) which divides a set of input samples based on some characteristics of their attributes into several smaller sets. The algorithm of the Decision Tree goes as follows:

- Feature selection. Filter out features that are highly correlated with classification results. The selection criterion is Gini coefficient which measure the degree or probability of a particular variable being wrongly classified when it is randomly chosen.
- Decision tree generation. Calculating the Gini coefficient of the features with the formula (1.2), where p_i is the probability of an object being classified to a particular class. The least

Gini index as the root node. Sub-nodes or branches are generated in the same way until the gain rate maximum or there are no features to choose from.

$$\text{Gini } = 1 - \sum_{i=1}^{n} (p_i)^2 \tag{1.2}$$

- Decision tree pruning. The main purpose of pruning is to avoid "overfitting" and reduce the risk of overfitting by actively removing some branches.

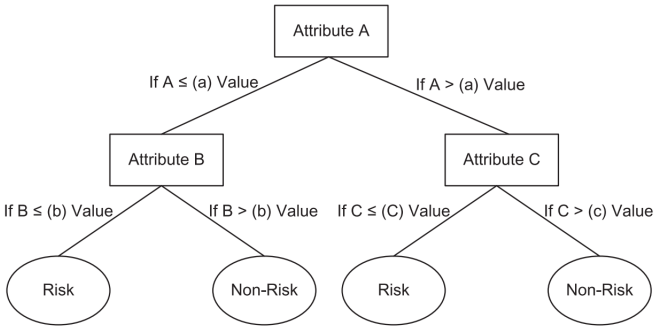

Figure 1.2: *An example of decision tree* [38]

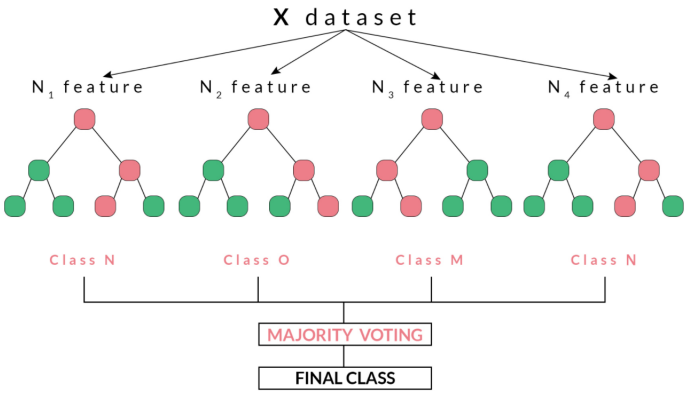

Figure 1.3: *An example of Random Forest*

For the Random forests three main hyperparameters require tuning. These are the number of trees, the number of attributes used to grow each tree (the node size) and the number of samples for each tree.

1.3.2 Data pre-processing

Data pre-processing is a data mining technique which consists in the transformations of the data before being fed to the algorithm. The purpose of pre-processing is to improve the quality of the data, thus improving the prediction and generalization capabilities of the model. The main steps of data pre-processing are divided into: data cleaning, data transformation and data reduction.

Data cleaning

Data cleaning makes the data correct and usable by handling missing values and outliers. In the process of obtaining information and data, there are various reasons that cause data loss and vacancies. Therefore, we need to detect the missing proportion of variables, then analyze the characteristics (continuous or discrete) and distribution of the variables, and use different methods to fill. The library *sklearn.impute* provides some imputation techniques:

- Imputation by the constant value 0.
- imputation by the mean, the median or the most frequent value.
- K Nearest Neighbor imputation.
- Iterative imputation

If the missing rate of the variable is high (greater than 80%) and the importance is low, the variable can be deleted directly.

Outliers are noise that is outside a specific distribution, which can spoil and mislead the training process resulting in longer training times, less accurate models and ultimately poorer results. Box plot is one of the simplest methods for detecting the data outliers. A box plot is a graphical display for describing the distribution of the data by using the median and the lower and upper quartiles. We can increase the sensitivity of the model to data by deleting some outliers that are far away from the median data.

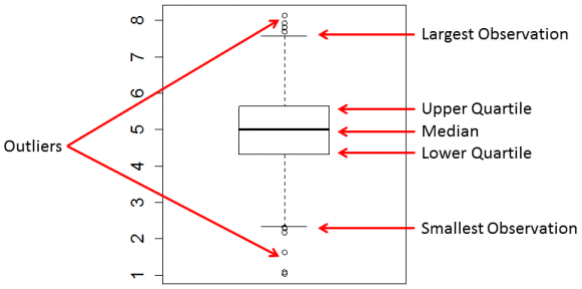

Figure 1.4: *The explanation of box plot*

Data transformation

This step is taken in order to transform the data in appropriate forms suitable for mining process. This involves following ways:

- Normalisation. The object is to improve algorithm accuracy and accelerate algorithm convergence speed. Two common methods are Min-Max Normalization and Z-score Standardization.
  - **Min-Max Normalization:** $x' = \frac{x - \min(x)}{\max(x) - \min(x)}$
  - **Z-score Standardization:** $x' = \frac{x - \mu}{\sigma}$

    μ: mean of the data, σ: standard deviation of the data
- Encoding method. Many classifiers like SVM and its derived versions only take numerical inputs. In credit scoring, data is often categorical (The German credit data set has 21 variables out of which 14 are qualitative variables and the remaining 7 are quantitative), consequently it is necessary to implement encoding techniques, like One-hot encoding [24], Ordinal encoding and Additive encoding, which provides the transformation of categorical features into binary features.

– **One-hot encoding:** This method maps each category to a vector that contains 1 and 0 denoting the presence or absence of the feature. The number of vectors depends on the number of categories for features.
– **Ordinal encoding:** For the ordinal variables, each category is assigned a value from 1 through N (N is the number of categories for the feature) and retains the ordinal nature of the variable. For the unordered qualitative features, use one-hot encoding.
– **Additive encoding:** For the ordinal variables, we consider the ordinal nature of the variable and map each category to a vector that contains 1 and 0 denoting the satisfied or not satisfied the feature.

For example (Fig 1.5), if v is a variable qualitative, $v_x = \{\mathrm{R}, \mathrm{G}, \mathrm{B}\}$, $\mathrm{R} \leq \mathrm{G} \leq \mathrm{B}$

	v
1	B
2	R
3	B
4	G
5	R

	R?	G?	B?
1	0	0	1
2	1	0	0
3	0	0	1
4	0	1	0
5	1	0	0

	v
1	3
2	1
3	3
4	2
5	1

	$\geq R$	$\geq G$	$\geq B$
1	1	1	1
2	1	0	0
3	1	1	1
4	1	1	0
5	1	0	0

Figure 1.5: *Sub1: An example of the variable qualitative. Sub2.3.4: the transformations using One-hot encoding, Ordinal encoding and Additive encoding*

Data reduction

While the data with a huge volume, data reduction technique could increase storage efficiency and reduce data storage and analysis costs. The most commonly used technique for the dimensionality reduction is PCA (Principal Component Analysis). PCA is an unsupervised technique that reduces dimensionality without significant loss of information.

[21] shows the importance of using PCA in the banking sector, especially for the loan granting where many factors are analyzed and there has to be an effective decision on whether which factors are more important than others. In this study from 15 initial variables of the dataset used, there were extracted 7 variables containing more than 60% of the original information.

PCA may be integrated with classification algorithms for better results by reducing data dimensionality before applying the classification algorithm. [14] proposes a hybrid model PCA-SVM for credit scoring. Using real-world German and UK credit datasets, this study demonstrates the effectiveness of the PCA-SVM model comparing to normal PCA and PCA-Logical Regression. Moreover, [29] proves that combining PCA and FSVM gives better classification accuracy than the other applied methods (SVM and Back Propagation neural networks).

1.4 Comparison of Methods

1.4.1 Review of comparison studies

Over the years, multiple comparison studies of Machine Learning methods have been published for credit scoring. (B.Baesens, 2003)[2] provides valuable insight in various classification techniques. Apart from the well-known traditional algorithms used for credit scoring (Linear Regression, Logistic Regression, Linear Discriminant Analysis), the decision tree, the nearest neighbor K-NN, the support vector machine (SVM) and the neural network have been applied to analyze credit risk. As seen from the table 2, for the Baesens study, SVM performs better.

	Boyle et al. (1992)	Desai et al. (1997)	West (2000)[1]	Lee et al. (2002)	Malhotra & Malhotra (2003)	Baesens (2003)[3]	Ong et al. (2005)[4]
Linear regression or LDA	**77.5**	66.5	79.3	71.4	69.3	79.3	80.8
Logistic regression		**67.3**	81.8	73.5		79.3	
Decision tree	75.0		77.0			77.0	78.4
Math programming	74.7					79.0	
Neural nets		66.4	**82.6**	73.7 (**77.0**)[2]	**72.0**	**79.4**	81.7
Genetic programming							**82.8**
K-nearest neighbours			76.7			78.2	
Support vector machines						**79.7**	

Table 2 - A comparison of overall accuracy from published research, source: (Crook, 2007) [3]

Lessman article (Lessmann, 2015) as an update of Baesens article, consists in an analysis of all credit scoring studies realized from 2003 to 2014. A comparison of 41 classifiers (including SVM) is made in terms of six performance measures using 8 real world datasets. Six indicators considered for evaluating the predictive accuracy of the models are: the area under the receiver operating characteristics curve (AUC), the performance-oriented congestion control (PCC), the brier score (BS), the H-measure, a partial gini index (PG) and the kolmogorov–smirnov statistic (KS). According to the results of the comparison, heterogeneous ensembles perform better overall, while SVM has an average performance. They combine multiple classification models, creating these models with different classification algorithms. Some examples are stacking, AvgS (Simple average ensemble) or AvgW (Weighted average ensemble).

According to current references, the vast majority consider that networks of neurons and other methods of machine learning are better than traditional statistical methods of credit scoring [35]. [28] exhaustively compares the performance of neural networks, NN, SVM, LR and other machine learning methods in credit scoring, and finally concludes that neural networks, the support vector machine method and the Fuzzy method are the methods that have relatively superior performance.

This article studies the performance of advanced kernel-based classifiers such as support vector machines (SVMs) and least-squares support vector machines (LS-SVMs) which have been proved to perform well in the classification of customers. In [41], Yu makes a comparison of methods using the following metrics: accuracy, interpretability, simplicity and flexibility. The results are shown in table 3. We can note that SVM performs with high accuracy and flexibility and has better interpretability than Neural Networks.

[41] also makes a comparison of 32 studied articles implementing Machine Learning methods for credit scoring. Most of the articles in this comparison analyse German and Australian datasets, but there are also some studies using different datasets like (Galindo and Tamayo 2000) that analyze data from Mexican banks. As seen from table 4 and the table 5, among the top methods, SVMs perform with the best accuracy.

Furthermore, there are some recent papers comparing multiple algorithms for imbalanced credit scoring data sets like [10], Random forest and extreme gradient boosting perform well under conditions of extremely unbalanced data. But they take into consideration only the main SVM algorithm

Methods	Accuracy	Interpretability	Simplicity	Flexibility
LDA,LOG, PR	★★	★★★	★★★	★
DT	★★	★★★	★★	★
KNN	★	★★★	★★★	★
LP	★	★★★	★★	★★★
NN	★★★	★	★	★
EA	★★	★	★	★
RS	★★	★	★★	★
SVM	★★★	★★	★	★★★
Hybrid/ensemble	★★★	★	★	★★

Table 3 - Comparisons of different credit risk models based on different criteria, source : (Lean YU) [41]

No	Author & Year	LDA	LOG	PR	DT	K-NN	LP	NN	ET	RS	SVM	HY
1	Baesens et al. 2005a		68.60-78.24					66.93-78.58				
2	Baesens et al. 2003a	72.2-88.6	72.0-89.2		67.1-90.4	66.7-89.5	71.2-89.5	72.4-89.4			71.2-89.5	
3	Baesens et al. 2003b				70.03-74.25			71.85-77.84				67.24-77.25
4	Bedingfield and Smith 2003								70.5			
5	Chen and Huang 2003	86.09			87.78			87.92				
6	Daubie et al. 2002				87.50					76.78		
7	Desai et al. 1997	66.53	67.30					66.38	65.70			
8	Desai et al. 1996	82.35	82.67					83.19				
9	Galindo and Tamayo 2000			84.87	91.7	85.05		89.0				
10	Gao et al. 2006							84.7	81.6			85.3
11	Van Gestel et al. 2006	86.49	86.49								89.34	89.34
12	Harmen and Leon 2005		78.4								75.0	
13	Hsieh 2004							96.16				
14	Hsieh 2005											97.99-98.46-
15	Huang et al. 2007				73.60-85.90			77.83-86.83	78.10-87.00			77.92-86.90
16	Huang et al. 2006		76.42-86.19		70.59-87.06	68.43-68.42		75.51-87.93	77.34-88.27			79.49-89.17

Table 4 - Overall Accuracy Comparison of different quantitative methods, source : (Lean YU) [41]

No	Author & Year	LDA	LOG	PR	DT	K-NN	LP	NN	ET	RS	SVM	HY
17	Huang et al. 2006				82.80			82.88				
18	Huysmans et al. 2005				68.9-69.8							71.3-71.9
19	Jagielska and Jaworski 1996							82.0				
20	Jiang and Yuan 2007							92.63				94.14
21	Lai et al. 2006c		60.66					78.36			83.49	93.27
22	Lee and Chen 2005	75.49	76.08					87.58				87.36
23	Lee et al. 2006	69.00	70.90		77.95			73.85				
24	Li et al. 2006							73.17			84.34	
25	Martens et al. 2007		85.70-96.40		80.20-94.6						85.70-97.00	82.00-96.20
26	Mirta et al. 2005		76.32		65.79			84.21				
27	Ong et al. 2005		75.40-86.19		74.57-87.06			75.51-87.93	77.34-88.27	74.57-83.72		
28	Schebesch and Stecking 2007										75.08	
29	Schebesch and Stecking 2005a										72.64	
30	Sun and Yang 2006							82.50			95.00	
31	Wang et al. 2005		64.18-82.53					62.13-81.45			64.54-78.8	66.17-83.94
32	West 2000		76.30-87.25		69.56-84.38	67.60-85.80		74.60-87.14				

Table 5- Continuation of table 4 : (Lean YU) [41]

and not the optimized versions of this algorithm like Fuzzy SVM[26], Fuzzy SVM for class imbalance learning[4], Weighted-LSSVM[36], or LS-Fuzzy SVM[40].

1.4.2 Comparison using Scikit-learn

We have also made a comparison of methods using Scikit-learn library. The data set is German Bank Credit, in which there are 300 bad clients and 700 good clients. In scikit-learn, the classifier methods are Linear Regression, Linear Discriminant Analysis, Quadratic Discriminant Analysis, K–nearest Neighbors Classifier, Decision Tree Classifier, MLP Classifier(neural network), Random Forest Classifier, AdaBoost Classifier, Gaussian Naive Bayes, XGBoost, and Support Vector Machine. The SVM-based methods are Fuzzy SVM, bilateral-weighted fuzzy SVM, least square SVM, least square fuzzy SVM, Weighted least square SVM, SVM bagging, FSVM bagging and LSFSVM bagging. For overall accuracy, the WLSSVM performs the best of all listed approaches, followed by DT and XGBOOST is the worst. The methods which are SVM-based performs good.

Methode	Bad precision	Good precision	Type I	Type II	Total accuracy	AUC
LR	0.475	0.887	0.636	0.801	0.765	0.681
LDA	0.525	0.872	0.633	0.815	0.77	0.699
QDA	0.559	0.816	0.559	0.816	0.74	0.687
KNN	0.424	0.887	0.61	0.786	0.75	0.655
DT	0.39	0.773	0.418	0.752	0.66	0.581
MLP	0.339	0.908	0.606	0.766	0.74	0.623
RF	0.288	0.957	0.739	0.763	0.76	0.623
AdaBoost	0.525	0.851	0.596	0.811	0.755	0.688
GaussianNB	0.458	0.851	0.562	0.789	0.735	0.654
XGBOOST	0.508	0.745	0.784	0.455	0.675	0.627
SVM (rbf)	0.441	0.922	0.703	0.798	0.78	0.681
LinearSVC	0.492	0.879	0.63	0.805	0.765	0.685
FSVM (rbf)	0.678	0.794	0.58	0.855	0.76	**0.736**
BFSVM (rbf)	0.661	0.816	0.6	0.852	0.77	**0.738**
LSSVM (rbf)	0.61	0.837	0.61	0.837	0.77	**0.723**
LSFSVM (rbf)	0.483	0.921	0.725	0.806	**0.79**	0.702
WLSSVM (rbf)	0.627	0.879	0.685	0.849	**0.805**	**0.753**
SVM_bagging (rbf)	0.508	0.894	0.667	0.813	0.78	0.701
FSVM_bagging (linear)	0.644	0.78	0.551	0.84	0.74	0.712
LSFSVM_bagging (linear)	0.492	0.879	0.63	0.809	0.765	0.685

Table 6 - Comparision of several classifiers in Scikit-learn on german dataset

1.5 Performance Measurement

In learning extremely imbalanced data, the overall classification accuracy is often not an appropriate measure of performance. A trivial classifier that predicts every case as the majority class can still achieve very high accuracy. We use metrics such as bad precision, good precision, Type I, Type II, Total accuracy and AUC to evaluate the performance of learning algorithms on imbalanced data. These metrics (except AUC) are functions of the confusion matrix as shown in Table 7. The rows of the matrix are actual classes, and the columns are the predicted classes.

The Area under the Curve (AUC) measures the percentage of the box that is under the Receiving Operating Characteristic (ROC) curve. The higher the percentage, the more accurate the

classifier. It is calculated by integrating the ROC curve lower bounded by 0.5. ROC curve is a graphical plot of the True Positive Rate ($\frac{\text{TP}}{\text{FN + TP}}$) versus the False Positive Rate ($\frac{\text{TN}}{\text{TN + FP}}$) for every threshold or cut-off.

Based on Table 7, the performance metrics are defined as:

$$\text{bad precision} = \frac{\text{TN}}{\text{TN + FP}} = \frac{\text{number of both actual bad and predict bad}}{\text{number of actual bad}}$$

$$\text{good precision (Recall)} = \frac{\text{TP}}{\text{FN + TP}} = \frac{\text{number of both actual good and predicted good}}{\text{number of actual good}}$$

$$\text{Type I} = \frac{\text{TN}}{\text{TN + FN}} = \frac{\text{number of both actual bad and predicted bad}}{\text{number of predicted bad}}$$

$$\text{Type II} = \frac{\text{TP}}{\text{TP + FP}} = \frac{\text{number of both actual good and predicted good}}{\text{number of predicted good}}$$

$$\text{Total accuracy} = \frac{\text{TN + TP}}{\text{TN + TP + FP + FN}} = \frac{\text{number of correct classification}}{\text{the number of evaluation sample}}$$

AUC: The quality of predictive models

	Predicted Negative class	Predicted Positive class
Actual Negative Class	TN (True Negative)	FP (False Positive)
Actual Positive Class	FN (False Negative)	TP (True Positive)

Table 7 - Confusion matrix

1.6 Conclusions

Comparing the models studied in different kinds of literature is not as easy as it may seem. This depends on the performance metrics used, as well as on the type and size of the data set analyzed. Each model has its own advantages and disadvantages. Which model is more convenient for us depends on the most important criteria fixed for the solution to our problem. For example, in several studies, AI methods perform better than traditional statistical methods in terms of accuracy, but they have the drawbacks of high complexity and lack of interpretability. These methods may fit perfectly well in many real-life problems, but for credit scoring problems they are not appropriate. That's why we can't say that there exists the best model for all situations.

In conclusion of the literature review and of all analyses realized in this chapter, for credit risk modeling, SVM and SVM-based give a good research direction. Selecting SVM for credit risk analysis is due to three reasons. Firstly, SVM is a class of data-driven, self-adaptive, and nonlinear methods that do not require specific assumptions (e.g., normal distribution in statistics) on the underlying data generating process. Secondly, SVM performs a nonlinear mapping from an original input space into a high dimensional feature space, in which it can construct a linear discriminant function to replace the nonlinear function in the original low dimensional input space. This characteristic also solves the curse of dimensionality problem because its computational complexity is not dependent on the sample dimension. Thirdly, SVM implements a structural risk minimization strategy to separate hyperplanes by using margin maximization principle, therefore possessing good generalization ability [41]. Furthermore SVM offers both interpretability of results and the possibility of outputting the probability that a client belongs to the bad or the good clients class.

However, the number of good clients is ten times or even dozens of times that of bad clients in reality. So the impact of the unbalanced data on the classification models has to be considered. [7] pointed out through experiments that, at the extreme class split (99% good, 1% bad) gradient boosting and random forest classifiers yield a very good performance, whereas the Lin LS-SVM sees a reduction in performance as a larger class imbalance is introduced.

Area under the receiver operating characteristic curve (AUC) results on test set data sets.

	30% Bad Friedman test statistic = 31.86 ($p < 0.005$)						15% Bad Friedman test statistic = 29.23 ($p < 0.005$)						10% Bad Friedman test statistic = 26.37 ($p < 0.005$)					
	Bene1	Bene2	Germ	Aus	Behav	AR	Bene1	Bene2	Germ	Aus	Behav	AR	Bene1	Bene2	Germ	Aus	Behav	AR
LOG	79.6	78.7	76.7	90.6	63.4	5.4	79.4	78.0	74.0	91.8	67.8	4.5	78.1	78.8	76.6	50.0	65.4	4.4
C4.5	71.4	71.0	71.2	91.8	61.9	9.1	69.7	60.9	65.2	91.6	61.6	8.2	64.7	64.0	64.1	91.9	50.3	8.4
NN	78.6	78.1	72.7	92.1	72.1	5.7	75.5	77.6	70.1	92.1	70.0	5.7	75.1	76.4	72.4	89.7	68.8	5.8
Gradient boosting	78.2	81.2	77.2	94.9	72.1	3.7	79.8	80.3	75.0	94.8	70.7	2.3	78.0	80.2	75.3	93.8	63.3	3.2
LDA	79.2	78.0	79.1	94.4	75.6	3.8	78.6	77.4	76.0	93.8	76.6	3.2	77.9	77.3	74.2	94.5	70.1	3.2
QDA	75.4	73.7	71.8	85.5	63.0	8.5	68.4	72.5	59.7	65.4	51.4	9.2	67.2	70.8	52.8	84.9	50.7	8.4
Random forests	78.5	79.0	80.0	93.7	76.2	3.2	77.9	78.0	76.9	94.1	76.5	2.7	78.6	76.9	77.2	93.2	74.7	2.2
k-NN10	76.2	71.0	75.0	92.8	61.8	7.7	75.5	68.1	71.7	90.3	58.7	7.9	70.4	64.4	68.8	92.5	56.3	6.8
k-NN100	75.4	73.9	79.3	93.0	56.0	6.7	75.8	73.6	78.1	92.6	62.9	4.6	75.3	72.9	78.5	92.3	61.7	4.6
Lin LS-SVM	80.3	80.6	81.9	95.1	82.9	1.2	50.0	54.4	75.0	91.0	90.0	6.7	50.0	50.0	76.8	90.6	50.0	8.0

	5% Bad Friedman test statistic = 26.29 ($p < 0.005$)						2.5% Bad Friedman test statistic = 27.43 ($p < 0.005$)						1% Bad Friedman test statistic = 30.86 ($p < 0.005$)					
	Bene1	Bene2	Germ	Aus	Behav	AR	Bene1	Bene2	Germ	Aus	Behav	AR	Bene1	Bene2	Germ	Aus	Behav	AR
LOG	75.0	75.4	75.7	50.0	50.0	5.5	72.7	73.9	55.1	50.0	50.0	6.3	50.0	64.7	50.0	50.0	50.0	7.7
C4.5	58.6	64.9	56.5	75.4	55.0	7.6	65.8	67.9	61.4	58.7	53.9	7.0	50.0	55.5	64.2	50.0	50.0	6.9
NN	68.4	70.7	68.3	89.4	64.4	5.0	71.2	70.2	59.2	70.0	62.3	4.6	50.0	62.5	54.2	86.7	54.0	5.6
Gradient boosting	70.8	78.0	76.6	93.1	52.7	3.4	68.1	74.7	71.4	88.3	55.6	2.8	58.1	69.1	59.4	74.5	51.0	3.5
LDA	74.1	76.1	73.8	93.5	63.5	2.6	75.7	72.2	62.6	81.8	60.6	3.0	50.2	69.0	58.3	86.8	54.6	3.4
QDA	63.8	72.0	50.0	59.7	50.5	7.9	66.5	65.3	50.0	51.6	50.5	8.3	50.0	50.0	50.0	52.0	50.7	7.9
Random forests	73.2	75.8	75.2	93.2	63.1	3.2	69.2	71.3	69.1	87.9	68.7	3.0	61.9	67.4	67.1	90.1	60.0	1.6
k-NN10	65.2	62.0	67.1	88.6	53.5	7.0	59.0	56.3	59.3	72.8	54.7	7.0	52.5	52.3	54.8	67.2	50.0	6.5
k-NN100	74.7	71.3	75.8	92.3	59.8	3.6	70.6	68.8	69.3	87.8	58.3	3.8	67.2	63.2	63.6	90.0	51.0	3.1
Lin LS-SVM	50.0	50.0	50.0	87.8	50.0	9.2	50.0	50.0	50.0	65.2	50.0	9.2	50.0	50.0	50.0	50.0	50.0	8.8

Table 7 - AUCs of all ten classifiers on the five credit scoring data sets at varying degrees of class imbalance [7]

Table 7 reports the AUCs of all ten classifiers on the five credit scoring data sets at varying degrees of class imbalance. At this original 70/30% split, the linear LS-SVM is the best performing classification technique with an average rank (AR) value of 1.2. However at the extreme class split (99% good, 1% bad) Random Forests provides the best average ranking across the five data sets (Random Forests also ranks first on the 10% data set). Therefore, we also implement Random Forest models for the real bank credit data set in Chapter 4.

Chapter 2

Support Vector Machines Learning Theory

2.1 SVM and SVM-based version

In this chapter we will introduce the principle of the Support Vector Machine model and 5 SVM-based version models. Fuzzy set is calculated based on the distance between the samples which are on the high dimension and the hyperplane, similar to the weight of each sample. Least Square SVM is transform the SVM optimal problem from a quadratic processing to a linear processing, so that all samples can determine hyperplane, not just support vectors.

2.1.1 SVM

The goal of the SVM learning algorithm is to find a separating hyperplane that separates these data points into two classes. In order to find a better separation of classes, the data are first transformed into a higher dimensional feature space by a mapping function Φ. Then, a possible separating hyperplane, which resides in the higher dimensional feature space, can be represented by:

$$w \cdot \Phi(x) + b = 0 \tag{2.1}$$

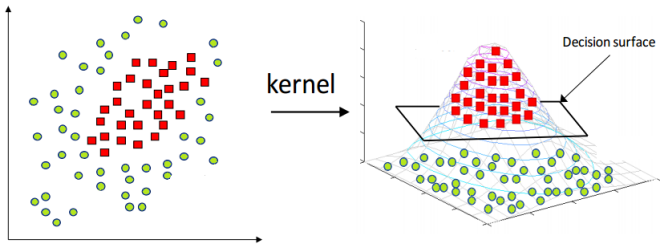

Figure 2.1: *Kernel Trick converts non-linear classification to linear classification by shifting the lower dimension space to higher dimensional space.*

When the datasets are not completely linearly separable, they are mapped into a higher dimensional feature space. Some points should be misclassified, but these misclassifications should be as few as possible. The constrains in the aforementioned optimization problem are relaxed by introducing a slack variable $\xi_i \geq 0$, and then, the soft-margin optimization problem is formulated as follows:

$$\begin{aligned} &\text{Minimize} \quad \Psi\left(w, b, \xi_i, \mu_i\right) = \frac{1}{2} w^T w + C \sum_{i=1}^{N} \xi_i \\ &\text{Subject to:} \quad y_i \left(w^T \phi\left(x_i\right) + b\right) \geq 1 - \xi_i \quad \text{for } i = 1, \ldots, N \\ &\qquad\qquad \xi_i \geq 0 \quad \text{for } i = 1, \ldots, N \end{aligned} \tag{2.2}$$

The slack variables $\xi_i \geq 0$ hold for misclassified examples, therefore, $\sum_{i=1}^{l} \xi_i$ can be considered as a measure of the amount of misclassifications. In this new objective function, it has to trade off the maximizing of the margin and minimizing of the error of the misclassification. The parameter C controls the tradeoff. When C is infinite, it becomes a hard-spaced state and erroneous points are not allowed at all; when C is smaller, more points that are misclassified are allowed. This quadratic-optimization problem can be solved by constructing a Lagrangian representation and transforming it into the following dual problem:

$$\begin{aligned} &\text{Maximize } W(\alpha) = -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j K\left(x_i, x_j\right) + \sum_{i=1}^{N} \alpha_i \\ &\text{Subject to:} \qquad \sum_{i=1}^{N} \alpha_i y_i = 0 \\ &\qquad\qquad 0 \leq \alpha_i \leq C \qquad \text{for} i = 1, \ldots, N \end{aligned} \tag{2.3}$$

in which $K\left(x_i, x_j\right) = \Phi(x_i) * \Phi(x_j)$. Solving the minimization problem involves taking the partial derivatives of (2.2), w can be recovered as follows

$$w = \sum_{i=1}^{l} \alpha_i y_i \Phi\left(x_i\right) \tag{2.4}$$

α_i are the Lagrange multipliers, in which $\alpha_i = 0$ called support vectors. Support vectors determine the hyperplane. Finally, the SVM decision function is given by

$$f(x) = \text{sign}(w \cdot \Phi(x) + b) = \text{sign}\left(\sum_{i=1}^{l} \alpha_i y_i K\left(x_i, x\right) + b\right) \tag{2.5}$$

2.1.2 Fuzzy SVM

Fuzzy SVM is an extension of an SVM which takes into account the variable meaning of the training samples. For FSVM, each training sample is associated with a membership value $\{\mu_i\}_{i=1}^{N} \in [0, 1]$. The membership value μ_i reflects the confidence level of the data points. The higher the value, the higher the confidence level of its class label.

- Attribution of Fuzzy-Membership Values

  1) μ_i is based on the distance from the class center proper: In this method, μ_i is defined relative to d_i^{cen}, which is the distance between x_i and its own class center. Examples closer to the class center are treated as more informative and given a higher x_i value, while examples far from the center are treated as outliers or noise and given a x_i lower value.

  Here we use two separate decomposition functions of d_i^{cen}:

$$d_{\text{lin}}^{\text{cen}}\left(x_i\right) = 1 - \frac{d_i^{\text{cen}}}{\max\left(d_i^{\text{cen}}\right) + \Delta} \tag{2.6}$$

Δ is a small positive value, which avoids problems when $d_{\text{lin}}^{\text{cen}}$ equals to zero.

$$d_{\text{exp}}^{\text{cen}}(x_i) = \frac{2}{1+\exp(\beta d_i^{\text{cen}})} \qquad \beta \quad \in [0,1] \tag{2.7}$$

where β determines the slope of the decay. $d_i^{\text{cen}} = \|x_i - \bar{x}\|^{1/2}$ is the Euclidean distance to x_i from its own class center $\overline{x}$.

2) μ_i is based on the distance from the real hyperplane: In this method, we define μ_i based on the distance between the real separation hyperplane and x_i, which is found by training a conventional SVM and LS-SVM model on the dataset.

When we train a normal SVM and LS-SVM model with the original dataset, we look for the functional margin $d_i^{\text{hyp}} = y_i(w \cdot \Phi(x_i) + b)$, then:

$$d_{\text{lin}}^{\text{hyp}}\ (x_i) = 1 - \frac{d_i^{\text{hyp}}}{\max\left(d_i^{\text{hyp}}\right) + \Delta} \tag{2.8}$$

$$d_{\text{exp}}^{\text{hyp}}(x_i) = \frac{2}{1+\exp\left(\beta d_i^{\text{hyp}}\right)} \quad \beta \quad \in [0,1] \tag{2.9}$$

Similar to SVM, the optimization problem of the Fuzzy SVM is also to maximize the geometric distance of the training samples, formulated as follows:

$$\begin{aligned} \text{Minimize} \quad & \Psi(w,b,\xi_i,\mu_i) = \frac{1}{2}w^T w + C\sum_{i=1}^{N}\mu_i\xi_i \\ \text{Subject to:} \quad & y_i\left(w^T\phi(x_i)+b\right) \geq 1-\xi_i \quad \text{for } i=1,\ldots,N \\ & \xi_i \geq 0 \quad \text{for } i=1,\ldots,N \end{aligned} \tag{2.10}$$

The solution of Fuzzy SVM is obtained from the quadratic programming problem (QP) above. Note that the error term ξ_i is scaled by membership value μ_i. The membership values used to weight the term soft penalty C reflect the relative confidence of the training samples.

It is transformed into the dual:

$$\begin{aligned} \text{Maximize } W(\alpha) = & -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j K(x_i,x_j) + \sum_{i=1}^{N}\alpha_i \\ \text{Subject to:} \quad & \sum_{i=1}^{N}\alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq \mu_i C \qquad \text{for} i=1,\ldots,N \end{aligned} \tag{2.11}$$

2.1.3 Bilateral-Weighted Fuzzy SVM

Bilateral-Weighted Fuzzy SVM is an extension of Fuzzy SVM. It is based on the idea of Fuzzy SVM that if one sample is detected as an outlier, its membership will decrease, consequently its contribution to total error term also decreases. In addition of Fuzzy SVM, this method also considers the sample as an input of the opposite class with higher membership.

Given the Lagrange multipliers: α_k, β_k, u_k, v_k the Lagrange function of the model is as below:

$$\begin{aligned}
&\max_{a_k,\beta_i,\mu_i,y_i,w,b_k,b_k} J\left(w,b,\xi_k,\eta_k;\alpha_k,\beta_k,\mu_k,v_k\right) \\
&= \tfrac{1}{2}w^r w + C\sum_{k=1}^{N} m_k\xi_k + c\sum_{k=1}^{N}\left(1-m_k\right)\eta_k \\
&- \sum_{k=1}^{N}\alpha_k\left[w\phi\left(x_k\right)+b-1+\xi_k\right) \\
&+ \sum_{k=1}^{N}\beta_k\left(w^*\phi\left(x_k\right)+b+1-\eta_k\right) - \sum_{k=1}^{N}u_k\xi_k - \sum_{k=1}^{N}v_k\xi_k
\end{aligned} \tag{2.12}$$

Differentiating with w, b, ξ_k and η_k,

$$\begin{aligned}
\tfrac{d}{dw}J = w - \sum_{k=1}^{N}\alpha_k\phi\left(x_k\right) + \sum_{k=1}^{N}\beta_k\phi\left(x_k\right) = 0 \\
\tfrac{d}{db}J = -\sum_{k=1}^{N}\alpha_k + \sum_{k=1}^{N}\beta_k = 0 \\
\tfrac{d}{d\xi_k}J = Cm_k - \alpha_k - \mu_k = 0 \text{ for } k = 1,\ldots,N \\
\tfrac{d}{d\eta_k}J = C\left(1-m_k\right) - \beta_k - v_k = 0 \text{ for } k = 1,\ldots,N
\end{aligned} \tag{2.13}$$

The following conditions are satisfied from the Kuhn-Tucker Theorem :

$$\begin{aligned}
\alpha_k\left(w^T * \phi\left(x_k\right)+b-1+\xi_k\right) = 0 \text{ for } k = 1,\ldots,N \\
\beta_k\left(w^T * \phi\left(x_k\right)+b+1-\eta_k\right) = 0 \text{ for } k = 1,\ldots,N \\
u_k\xi_k = 0 \text{ for } k = 1,\ldots,N \\
v_k\xi_k = 0 \text{ for } k = 1,\ldots,N
\end{aligned} \tag{2.14}$$

$$\alpha_k \geq 0; \beta_k \geq 0; \mu_k \geq 0; v_k \geq 0; \xi_k \geq 0; \eta_k \geq 0 \text{ for } k = 1,\ldots,N \tag{2.15}$$

After transformations based on the above conditions, a new formulation of the model is obtained:

$$\begin{cases}
\max_{\alpha_i/\beta_i} \sum_{k=1}^{N}\alpha_k + \sum_{k=1}^{N}\beta_k - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\left(\alpha_i-\beta_i\right)\left(\alpha_j-\beta_j\right)\phi\left(x_i\right)\phi\left(x_j\right) \\
\text{s.t.} \quad \sum_{k=1}^{N}\alpha_k = \sum_{k=1}^{N}\beta_k \\
0 \leq \alpha_k \leq Cm_k \text{ for } k = 1,\cdots,N \\
0 \leq \beta_k \leq C\left(1-m_k\right) \text{ for } k = 1,\cdots,N
\end{cases} \tag{2.16}$$

After transformed into a quadratic programming problem, the above optimization becomes:

$$\begin{cases}
\min_{\beta_k,\eta_k} \sum_{k=1}^{N}\gamma_k + \sum_{k=1}^{N}2\beta_k - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\gamma_i\gamma_j K\left(x_i,x_j\right) \\
\text{s.t.} \quad \sum_{k=1}^{N}\gamma_k = 0 \\
0 \leq \beta_k + \gamma_k \leq Cm_k \quad \text{for } k = 1,\cdots,N \\
0 \leq \beta_k \leq C\left(1-m_k\right) \quad \text{for } k = 1,\cdots,N
\end{cases} \tag{2.17}$$

After solving this and substituting $w = \sum_{k=1}^{N}\left(\alpha_k-\beta_k\right)\phi\left(x_k\right)$ into the original classification problem, the following classifier is obtained:

$$y(x) = \text{sign}\left(w^T\phi(x)+b\right) = \text{sign}\left(\sum_{k=1}^{N}\left(\alpha_k-\beta_k\right)K\left(x,x_k\right)+b\right) \tag{2.18}$$

2.1.4 Least Square Fuzzy SVM

Least Square Fuzzy SVM transforms the traditional inequality constraints of FuzzySVM into equality constraints and uses the error and squared loss function as the empirical loss of the learning set. Consequently, the solution to the problem of quadratic programming is solved in a problem of linear equation, which improves the speed of the solution and the precision of convergence.

$$\begin{aligned} &\text{Minimize} \quad \varphi\left(w, b, \xi_i, \mu_i\right)=\frac{1}{2} w^T w+\frac{C}{2} \sum_{i=1}^N \mu_i \xi_i^2 \\ &\text{Subject to: } y_i\left(w^T \phi\left(x_i\right)+b\right)=1-\xi_i \qquad \text{for } i=1, \ldots, N \end{aligned} \tag{2.19}$$

The Lagrangian function is defined as:

$$\begin{aligned} \min_{w, b, \xi_i} L\left(w, b, \xi_i ; \alpha_i\right) =& \frac{1}{2} w^T w+\frac{C}{2} \sum_{i=1}^N \mu_i \xi_i^2 \\ &-\sum_{i=1}^N \alpha_i\left[y_i\left(w^T \phi\left(x_i\right)+b\right)-1+\xi_i\right] \end{aligned} \tag{2.20}$$

where α_i is the ith Lagrange Multiplier, unlike the Fuzzy SVM Lagrange Multiplier, which can be positive or negative (Appendix A), due to equality constraints in accordance with the KKT conditions. So LSFSVM lacks clarity. Clarity is defined as the number of zeros in the parameter or the data. The more zeros, the more sparse the parameter or data. Lack of clarity makes LSFSVM possible to select and interpret more the variables. Since most of the Lagrangian multipliers are not zero, which means more Lagrangian multipliers determine the result of the classification.

Dierentiating (2.4) with w, b, and ξ_i:

$$\begin{aligned} \tfrac{d}{dw} L\left(w, b, \xi_i ; \alpha_i\right) &= w-\textstyle\sum_{i=1}^N \alpha_i y_i \phi\left(x_i\right)=0 \\ \tfrac{d}{db} L\left(w, b, \xi_i ; \alpha_i\right) &= -\textstyle\sum_{i=1}^N \alpha_i y_i=0 \\ \tfrac{d}{d \xi_i} L\left(w, b, \xi_i ; \alpha_i\right) &= \mu_i C \xi_i-\alpha_i=0 \\ \tfrac{d}{d \alpha_i} L\left(w, b, \xi_i ; \alpha_i\right) &= y_i\left[w^T \phi\left(x_i\right)+b\right]-1+\xi_i=0 \end{aligned} \tag{2.21}$$

From (2.5), we can obtain the following equations:

$$\begin{aligned} w &= \textstyle\sum_{i=1}^N \alpha_i y_i \phi\left(x_i\right) \\ \textstyle\sum_{i=1}^N \alpha_i y_i &= 0 \\ \alpha_i &= \mu_i C \xi_i \\ y_i\left[w^T \phi\left(x_i\right)+b\right]-1+\xi_i &= 0 \end{aligned} \tag{2.22}$$

Using a matrix form, the optimal conditions in (3.6) can be expressed by:

$$\left[\begin{array}{cc} \boldsymbol{\Omega} & \boldsymbol{Y} \\ \boldsymbol{Y^T} & 0 \end{array}\right]\left[\begin{array}{c} \boldsymbol{\alpha} \\ b \end{array}\right]=\left[\begin{array}{c} \mathbf{1} \\ 0 \end{array}\right] \tag{2.23}$$

where Ω_{ij},$\mathbf{Y}$,$\mathbf{1}$ represent:

$$\begin{aligned} \Omega_{ij} &= y_i y_j \phi\left(x_i\right)^T \phi\left(x_j\right)+\left(\mu_i C\right)^{-1} I \\ \mathbf{Y} &= \left(y_1, y_2, \ldots, y_N\right)^T \\ \mathbf{1} &= (1,1, \ldots, 1)^T \end{aligned} \tag{2.24}$$

So the linear equations of (7):

$$\begin{bmatrix} 1 & K(x_1,x_1)+\frac{1}{c} & K(x_1,x_2)+\frac{1}{c} & \cdots & K(x_1,x_l)+\frac{1}{c} \\ 1 & K(x_2,x_1)+\frac{1}{c} & K(x_2,x_2)+\frac{1}{c} & \cdots & K(x_2,x_l)+\frac{1}{c} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & K(x_l,x_1)+\frac{1}{c} & K(x_l,x_2)+\frac{1}{c} & \cdots & K(x_l,x_l)+\frac{1}{c} \\ 0 & 1 & 1 & \cdots & 1 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_l \\ b \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_l \\ 0 \end{bmatrix} \tag{2.25}$$

Therefore, LS-SVM obtains the threshold b and the optimal solution by solving the linear equations of the equation $\boldsymbol{\alpha} = (a_1, \alpha_2, \cdots, \alpha_l)^{\mathrm{T}}$.

The code for FSVM and LSFSVM is in the Appendix B.

2.1.5 Least Square Bilateral Fuzzy SVM

This method combines Bilateral-Weighted Fuzzy SVM and Least Square Fuzzy SVM, profiting in this way the advantages of both methods (better generalization ability and reduced computational complexity).
Using this method the classification problem is formulated as follows:

$$\min_{w,\xi,\eta} J_5(w,\xi,\eta) = \frac{1}{2}w^T w + \frac{c}{2}\sum_{i=1}^{N}\left[\mu_i \xi_i^2 + (1-\mu_i)\,\eta_i^2\right] \tag{2.26}$$

subject to

$$\begin{cases} w^T\phi(x_i) + b = 1 - \xi_i & i = 1,\ldots,N \\ w^T\phi(x_i) + b = -1 + \eta_i & i = 1,\ldots,N \\ \xi_i \geq 0 & i = 1,\ldots,N \\ \eta_i \geq 0 & i = 1,\ldots,N \end{cases} \tag{2.27}$$

The Lagrangian function is constructed as:

$$\begin{aligned} L(w,b,\xi,\eta,\alpha,\beta) = J_5(w,\xi,\eta) - \sum_{i=1}^{N}\alpha_i\left[w^T\phi(x_i) + b - 1 + \xi_i\right] \\ - \sum_{i=1}^{N}\beta_i\left[w^T\phi(x_i) + b + 1 - \eta_i\right] \end{aligned} \tag{2.28}$$

where $\{\alpha_i\}_{i=1}^{N} \geq 0, \{\beta_i\}_{i=1}^{N} \geq 0$ are the Lagrangian multipliers of (2.23)
From differentiating are obtained:

$$\begin{aligned} \frac{\partial L}{\partial w} = 0 \quad &\Rightarrow \quad w - \sum_{i=1}^{N}(\alpha_i + \beta_i)\,\phi(x_i) = 0 \\ \frac{\partial L}{\partial b} = 0 \quad &\Rightarrow \quad \sum_{i=1}^{N}(\alpha_i + \beta_i) = 0 \\ \frac{\partial L}{\partial \xi_i} = 0 \quad &\Rightarrow \quad c\mu_i\xi_i - \alpha_i = 0 \\ \frac{\partial L}{\partial \eta_i} = 0 \quad &\Rightarrow \quad c(1-\mu_i)\,\eta_i + \beta_i = 0 \\ \frac{\partial L}{\partial \alpha_i} = 0 \quad &\Rightarrow \quad w^T\phi(x_i) + b = 1 - \xi_i \\ \frac{\partial L}{\partial \beta_i} = 0 \quad &\Rightarrow \quad w^T\phi(x_i) + b = -1 + \eta_i \end{aligned} \tag{2.29}$$

Considering $K(x_i,x_j) = \phi(x_i)^T\phi(x_j)$, the following linear equations are obtained by simple substitutions:

$$\begin{cases} \sum_{i=1}^{N} (\alpha_i + \beta_i) = 0 \\ \sum_{j=1}^{N} (\alpha_j + \beta_j) K(x_i, x_j) + b = 1 - \frac{\alpha_i}{c\mu_i} \\ \text{for } i = 1, \ldots, N \\ \sum_{j=1}^{N} (\alpha_j + \beta_j) K(x_i, x_j) + b = -1 - \frac{\beta_i}{c(1-\mu_i)} \\ \text{for } i = 1, \ldots, N \end{cases} \tag{2.30}$$

Finally the classifier for this method is as follows:

$$z(x) = \text{sign}\left[\sum_{i=1}^{N} (\alpha_i + \beta_i) K(x_i, x) + b\right] \tag{2.31}$$

2.1.6 Weighted Least Square SVM

This model is based upon the LS-SVM solution of the first step. In the subsequent step, one can weight the error variables $e_k = \alpha_k/\gamma$ by weighting factors v_k. The optimal problem:

$$\min_{w^\star, b^\star, e^\star} J(w^\star, e^\star) = \frac{1}{2} w^{\star \mathrm{T}} w^\star + \frac{1}{2}\gamma \sum_{k=1}^{N} v_k e_k^{\star 2} \tag{2.32}$$

And the Lagrangian becomes:

$$L(w^\star, b^\star, e^\star; \alpha^\star) = J(w^\star, e^\star) - \sum_{k=1}^{N} \alpha_k^\star \left\{ w^{\star \mathrm{T}} \varphi(x_k) + b^\star + e_k^\star - y_k \right\} \tag{2.33}$$

$\star$ symbol represent the unkown variables. And the KTT system:

$$\left[\begin{array}{c|c} 0 & 1_v^{\mathrm{T}} \\ \hline 1_v & \Omega + V_\gamma \end{array}\right] \left[\begin{array}{c} b^\star \\ \hline \alpha^\star \end{array}\right] = \left[\begin{array}{c} 0 \\ \hline y \end{array}\right] \tag{2.34}$$

where the diagonal matrix V_γ is given by:

$$V_\gamma = \text{diag}\left\{ \frac{1}{\gamma v_1}, \ldots, \frac{1}{\gamma v_N} \right\} \tag{2.35}$$

The choice of the weights v_k is determined based upon the error variables $e_k = \alpha_k/\gamma$ from the (unweighted) LS-SVM case. Robust estimates are obtained then by taking

$$v_k = \begin{cases} 1 & \text{if } |e_k/\hat{s}| \leqslant c_1 \\ \frac{c_2 - |e_k/\hat{s}|}{c_2 - c_1} & \text{if } c_1 \leqslant |e_k/\hat{s}| \leqslant c_2 \\ 10^{-4} & \text{otherwise} \end{cases} \tag{2.36}$$

where $\hat{S}$ is a robust estimate of the standard deviation of the LS-SVM error variables e_k:

$$\hat{s} = \frac{\text{IQR}}{2 \times 0.6745} \tag{2.37}$$

The interquartile range IQR is the difference between the 75th percentile and 25th percentile.

2.2 Probabilistic Outputs for Support Vector Machines

We consider to have the probability of the classification. We used the Platt's probabilistic outputs for Support Vector Machines, which is proposed by John C.Platt in 1999 [33]. This has been later improved by Hsuan-Tien Lin and al. [27] who have shown that the method theoretically converges and avoids numerical difficulties.

Platt's Probabilistic Outputs for Support Vector Machines

The formula of the output probability value is:

$$\Pr(y=1|x) \approx P_{A,B}(f) = \frac{1}{1+\exp(Af+B)}, \text{ where } f = f(x) \tag{2.38}$$

in which parameters A and B to adjust the size of the mapping value. $f(x)$ is the support vector machine decision function:

$$f(x) = \left(w^T \phi(x) + b\right) = \sum_{i=1}^{l} y_i \alpha_i K(x_i, x) + b \tag{2.39}$$

$\sum_{i=1}^{l} y_i \alpha_i K(x_i, x) + b$ is proportional to the distance of the hyperplane. The idea of this algorithm is that the closer to the hyperplane, the smaller the possibility of pairing; the further away from the hyperplane, the greater the possibility of pairing.
In [27], each f_i is an estimate of $f(x_i)$. The best parameter setting $z* = (A*, B*)$ is determined by solving the following regularized maximum likelihood problem (with $N+$ of the y_i's positive, and $N-$ negative:

$$\begin{aligned} &\min_{z=(A,B)} \quad F(z) = -\sum_{i=1}^{l} \left(t_i \log(p_i) + (1-t_i)\log(1-p_i)\right) \\ &\text{for} \qquad p_i = P_{A,B}(f_i), \text{ and } t_i = \begin{cases} \frac{N_+ +1}{N_+ +2} & \text{if } y_i = +1 \\ \frac{1}{N_- +2} & \text{if } y_i = -1 \end{cases}, i = 1, \ldots, l \end{aligned} \tag{2.40}$$

Platt (2000) gives a pseudo code for solving (2.40), we got parameters A and B, bringing them into (2.38), we got the probabilistic output of each test example x.

Chapter 3

Implementation of SVM based methods on the german credit data set

3.1 Introduction

The credit dataset used in this chapter is German credit dataset, which is provided by Professor Dr. Hans Hofmann of the University of Hamburg and is obtained from UCI Machine Learning Repository (http://www.ics. uci.edu/ mlearn/databases/statlog/german/). The total number of instance is 1000 including 700 creditworthy cases and 300 default cases. For each applicant, 20 kinds of attribute are available, such as account balance, credit history, loan purpose, credit amount, employment status, personal status and sex, age, housing, telephone status and job.
We will use this data set to test the SVM based models.

3.2 Features of German data set

3.2.1 Weight of Evidence (WOE) and Information Value (IV)

weight of evidence (WOE) and information value (IV) evolved from the logistic regression technique and has been applied in credit scoring for a long time. They are used as a benchmark to screen variables in the credit risk modeling projects.

WOE tells the predictive power of an independent variable in relation to the dependent variable. And IV helps to rank variables on the basis of their importance. The WOE is calculated using the following formula:

$$WOE = \ln\left(\frac{\text{Distribution of goods}}{\text{Distribution of bads}}\right)$$

Positive WOE means the percentage of good clients is more than the percentage of bad clients in a particular group. Negative WOE means less than.

The formula of IV is:

$$IV = \sum(\%\text{ of goods } - \%\text{ of bads }) * WOE$$

By convention the values of the IV statistic for variable selection can be used as follows:

Ranking shows how important features are to the result of prediction, or to the classification of clients.

	Attribute	Type
1	Status of existing checking account	qualitative
2	Duration in month	numerical
3	Credit history	qualitative
4	Purpose	qualitative
5	Credit amount	numerical
6	Savings account/bonds	qualitative
7	Present employment since	qualitative
8	Installment rate in percentage of disposable income	numerical
9	Personal status and sex	qualitative
10	Other debtors / guarantors	qualitative
11	Present residence since	numerical
12	Property	qualitative
13	Age in years	numerical
14	Other installment plans	qualitative
15	Housing	qualitative
16	Number of existing credits at this bank	numerical
17	Job	qualitative
18	Number of people can provide maintenance	numerical
19	Telephone	qualitative
20	Foreign worker	qualitative

Table 3.1: Detailed information on data attributes

Information Value	Variable Predictiveness
≤ 0.5	Suspicious relationship
0.3 to 0.5	Predictor has a strong relationship to the Goods/Bads odds ratio
0.1 to 0.3	Predictor has a medium relationship to the Goods/Bads odds ratio
0.02 to 0.1	Predictor has a weak relationship to the Goods/Bads odds ratio
Less than 0.02	Not useful for prediction

Table 3.2: Rules of Information Value

3.2.2 The judgmental approach

The judgmental approach is a qualitative, expert-based approach. This approach makes decision about the credit risk based on the business experience and common sense. Commercial banks adopt the "6C" principle for loan review.The “6C” analysis method is a traditional credit risk measurement method for commercial banks. Character, capacity, capital, collateral, condition, and continuity in business this six factors to assess its credibility.

- Character, which is the most important. It reflects each credit transaction in the speed and amount of payment.
  (Attribut : credit amount, credit history, credits in this bank, personal status and sex, present employment since, telephone, present residence since...)
- Capacity, including the client's operating ability, management ability and solvency.
  (Attribut : checking account status, savings account, other debtors, other installment plans, purpose, job...)
- Capital, refers to the financial strength and financial condition of the client, indicating the

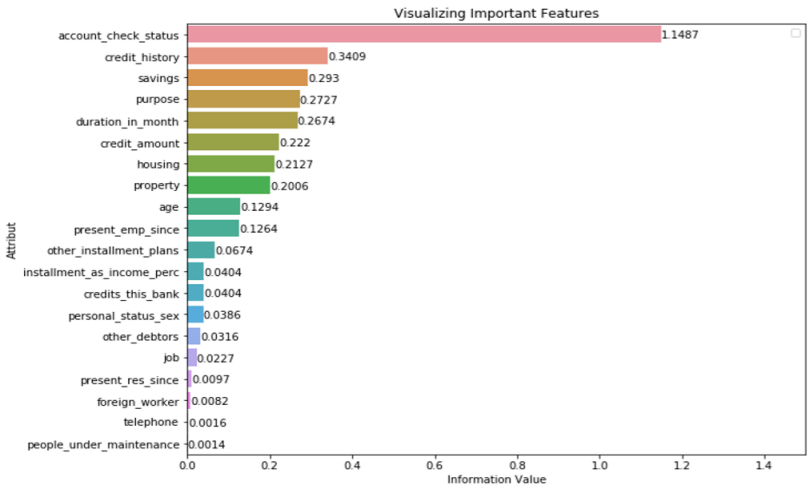

Figure 3.1: Chart showing the importance of each feature

background in which the client may repay the debt.
(Attribut : checking account status, savings account, other installment plans, other debtors, property, people under maintenance, housing...)

- Collateral. It is an asset that is used as collateral when a customer refuses to pay or is unable to pay. This is more important for customers who do not know the details or have disputed credit status.
(Attribut : property, housing...)

- Condition. The operating environment mainly refers to the internal and external environment in which the customer operates. When these environments change, will the customer's ability to pay back be affected.
(Attribut : property, job, duration in month, installment rate in percentage of disposable income, purpose...)

- Continuity in business refers to the possibility of continuous operation of the customer, which requires a comprehensive evaluation from the customer's internal financial situation, product replacement, and scientific and technological development.
(Attribut : checking account status, savings account, purpose, job...)

3.2.3 Analyse of combining IV and judgmental scoring

In the figure 3.1 checking account status presents the most important attribute to classifier the client. Checking accounts are usually used for daily consumption. Checking account is totally liquid, as opposed to less-liquid savings accounts. These two accounts can reflect the customer's financial situation. Credit history and credit amount are important factors that reflects the quality of customers: the customer's historical credit better, then the possibility of repayment higher. Purpose is the purpose of the consumer loan, what customers will use this loan to do. The results show that loans for retraining and education purposes have a higher repayment rate than loans for car or domestic appliance purchases. Duration measures how long it takes, in month, for an investor to be repaid the bond's price by the bond's total cash flows. The duration shorter, the higher probability to repayment.

Based on the rules of Information Value, the ranking shows that present residents since, foreign worker, telephone and number of people under maintenance are not useful.

3.3 Performance evaluation

German dataset, there are a total of 1,000 samples, of which 300 are bad customers and belong to a minority class, and 700 are good customers and belong to a majority class. During the data preprocessing, 13 qualitative variables are converted into quantitative variables using a "one-hot-encoder". The number of quantitative variables becomes 61. Then, the analysis of the main components was used to reduce the dimension. Finally, we keep 30 main components for comparison.

3.3.1 FuzzySVM and LS-FuzzySVM with different method of calculating Fuzzy membership value

Fuzzy SVM :

Méthode	bad precision	good precision	Type I	Type II	Total accuracy	AUC
d_{lin}^{cen}	0.566	**0.842**	0.626	0.813	**0.756**	**0.704**
d_{exp}^{cen}	0.573	0.801	0.554	0.813	0.733	0.687
d_{lin}^{hyp}	0.572	0.817	0.584	0.815	0.743	0.695
d_{exp}^{hyp}	**0.584**	0.806	0.563	0.820	0.740	0.695

LS-Fuzzy SVM :

Méthode	bad precision	good precision	Type I	Type II	Total accuracy	AUC
d_{lin}^{cen}	0.503	**0.921**	0.742	0.806	**0.792**	0.712
d_{exp}^{cen}	0.492	0.908	0.701	0.805	0.782	0.700
d_{lin}^{hyp}	0.511	0.911	0.722	0.812	0.790	0.711
d_{exp}^{hyp}	**0.534**	0.894	0.682	0.819	0.787	**0.714**

According to the table above, for the German dataset, μ_i is based on the distance from the proper class center in linear giving good results. For LS-FuzzySVM, μ_i is based on the distance from the real hyperplane in exponential has given good results.

In addition, we can also change $\beta \in [0, 1]$ in the exponential decay functions to get better results.

3.3.2 Comparison of different Kernel

Fuzzy SVM :

Kernel	bad precision	good precision	Type I	Type II	Total accuracy	AUC
rbf	0.566	**0.842**	0.626	0.813	**0.756**	**0.704**
lin	0.570	0.784	0.517	0.820	0.723	0.677
poly	**0.652**	0.764	0.541	0.838	0.731	0.708

LS-Fuzzy SVM :

Kernel	bad precision	good precision	Type I	Type II	Total accuracy	AUC
rbf	0.503	**0.921**	0.742	0.806	**0.792**	**0.712**
lin	0.467	0.856	0.571	0.798	0.744	0.662
ploy	**0.545**	0.864	0.631	0.818	0.769	0.704

According to the table above, Kernel = 'rbf' works better.

$$K\left(x_i, x_j\right) = \exp\left(-\left\|x_i - x_j\right\|^2 / \left(2\sigma^2\right)\right), i = 1, 2, \cdots, l \tag{3.1}$$

σ is kernel width, which determines the complexity of the example entity subspace distribution. The larger σ, the more support vectors there are, which will affect predictions.

3.3.3 Comparison of methods

$d_{\text{lin}}^{\text{cen}}$, kernel='rbf'.

Méthode	bad precision	good precision	Type I	Type II	Total accuracy	AUC
Fuzzy SVM	0.566	0.842	0.626	0.813	0.756	0.704
LS-Fuzzy SVM	0.503	0.921	0.742	0.806	0.792	0.712
FuzzySVM&bagging	**0.580**	0.917	0.756	0.833	0.815	**0.749**
LS-FuzzySVM&bagging	0.544	**0.943**	0.810	0.824	**0.821**	0.744

For overall accuracy, LS-FuzzySVM & bagging has the highest result. However, for minority predictions, Fuzzy SVM & bagging works better.

3.3.4 LS-Fuzzy SVM and LS-FuzzySVM with data balancing

$d_{\text{lin}}^{\text{cen}}$, kernel='rbf'.

Méthode	Méthode Sampling	bad precision	good precision	Total accuracy	AUC
Fuzzy SVM	origine	0.566	0.842	0.756	0.704
	OverSampling	0.553	0.788	0.719	0.670
	UnderSampling	0.736	0.626	0.662	0.681
LS-Fuzzy SVM	origine	0.503	0.921	0.792	0.712
	OverSampling	0.569	0.806	0.737	0.687
	UnderSampling	0.788	0.640	0.687	0.714
FuzzySVM&bagging	origine	0.580	0.917	0.815	0.749
	OverSampling	0.595	0.918	0.822	0.756
	UnderSampling	**0.822**	0.66	0.709	0.741
LS-FuzzySVM&bagging	origine	0.544	**0.943**	0.821	0.744
	OverSampling	0.641	0.916	**0.835**	**0.778**
	UnderSampling	0.759	0.763	0.763	0.761

origin: Do nothing after analyzing the main components

OverSampling: Using SVM-SMOTE, which uses an SVM algorithm to detect the sample to be used to generate new synthetic samples. Increase the number of minority classes to the same level as for the majority class.

UnderSampling: Randomly sample the majority class until the number of majority classes is the same as that of the minority class.

Oversampling and undersampling are primarily aimed at balancing the data and improving accuracy. With oversampling and undersampling, minority class accuracy can be improved, and the method of using undersampling is more obvious. Using oversampling in bagging can improve the overall accuracy and predictive power of the model.

3.3.5 Conclusion

Method	bad precision	good precision	Total accuracy	AUC	Robustness	Execution Time
Fuzzy SVM	**	*	*	*	*	**
LS-Fuzzy SVM	*	***	**	**	*	****
FuzzySVM&bagging	****	**	***	****	***	*
LS-FuzzySVM&bagging	***	****	****	***	***	***

bad precision, good precision, Total accuracy, AUC: low *$\cdots$****high

Robustness: poor *$\cdots$**** good

Execution time: slow *$\cdots$**** fast

The code is executed 10 times for each method and reports the average. The standard deviation represents the degree of dispersion from the mean. A larger standard deviation means less robustness.

In short, LS-FuzzySVM & bagging combine the advantages of FuzzySVM and integrated learning with speed, high precision and great durability. They perform better.

Chapter 4

Study on a real bank credit data set

4.1 Introduction

The data are retrieved from the loan inventory of a bank out of France. Firstly, we have analyzed and checked the data to identify incomplete, inaccurate or irrelevant records from the database and removed them. Secondly, we have considered the borrowers once because there are cases when a borrower might have other exposures within the bank. Thirdly, for each borrower we have observed over one-year performance horizon. It considered a default (or bad) loan if a borrower for the first time is overdue more than +90 days and it is denoted in the database with value -1 and 1 for the rest of the borrowers (good).

In this chapter, we have first performed pre-processing data set, then use SVM, SVM-based models and Random Forest models to classify new data. We have compared the performance of different models and at the end selected the best models for this data.

4.2 Data processing

4.2.1 Data analysis

First of all, we analyse the whole data. There are 10215 samples in this data, of which 9737 samples are considered as good clients and 478 samples are considered as bad clients.

The features include 18 variables, in which 14 variables quantitative and 11 variables qualitative. Among ordinal encoding, additive encoding and one hot encoding three coding methods, we have chosen one hot encoding.

The ratio of the good and bad clients is about 22 times, hence the data set is heavily imbalanced. This will make the supervised learning algorithm pay too much attention to the majority class, and reduce the classification performance. Thus, before the model training, we will use some oversampling or undersampling methods to process the data respectively.

Description of features:

Variables	Explanation	Type
Approved loan	The amount of loans has been approved	numerical
Duration (in months)	The duration of the loan (if loan)	numerical
Currency	Local currency or Euros	qualitative
Age	Age	numerical
Gender	Male or Female	qualitative
Marital Status	Single, Married, Widowed, Separated, Divorced	qualitative
Number of family members	Number of family members	numerical
Education Type	Education level	qualitative
Current address	Number of years at the current address	numerical
Employment Type	The type of jobs	qualitative
Current years at job	Current years at job	qualitative
Residential Status	Rent, Co-owner, Owner, Others, Live with parents	qualitative
Credit Registry Information	The stats of credit registry information	numerical
Monthly Installment of the loan	Monthly Installment of the loan	numerical
Other installments of other loans	Other installments of other loans	numerical
Monthly Income	Monthly Income	numerical
Reference	the reference whether from the bank staff	qualitative
Type of Loan	Loan or overdraft	qualitative
Saving account	The state of saving account	numerical
Current account	The state of checking account	numerical
Purpose	The purpose of loan	qualitative
Collateral type	Residential, Commercial, Land, Deposits pledge, Others	qualitative
Other debitors/guarantors	Have other debitors/ guatantors or not	numerical
Credit history	Number of existing credits at this bank	numerical
Foreign work	Have foreign work or not	numerical

4.2.2 Consistency and outlier analysis

The purpose of the consistency of data in the information system is to ensure that the recorded data conforms to the rules defined by "business" and there are no inconsistencies. In the consistency analysis, we have removed 33 clients whose total installments are more than their respective monthly income. We have removed also 16 clients whose loan type is mortgage but with no collateral provided.

Outliers are values that deviate significantly from the large majority of the sampled data. To find outliers in the sample population, the outlier detection method is usually adopted. The boxplot is a standardized way of displaying the distribution of data based on a five number summary ("minimum", first quartile, median, third quartile, and "maximum"). The outliers are (for a normal distribution) 0.7% of the data. According to the box plot below, we can see that for some of the qualitative variables exist outliers [Figure 4.1].

After removing the unreasonable samples and the outliers, 10114 samples were left. There are 9652 good customers and 462 bad customers.

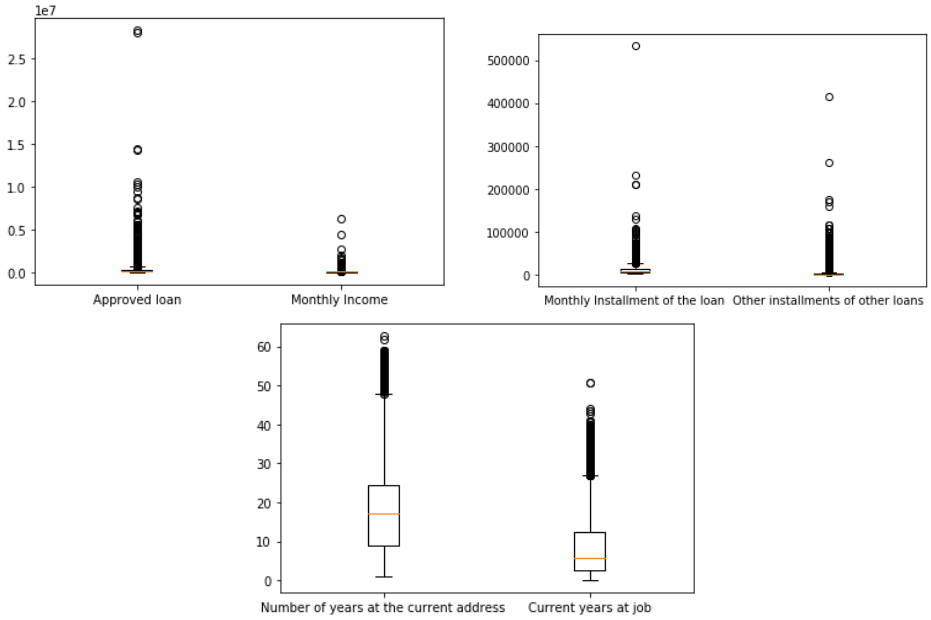

Figure 4.1: Box plot of the variables qualitative (the circles are outliers

4.2.3 Feature engineering

Principle Component Analysis (PCA)

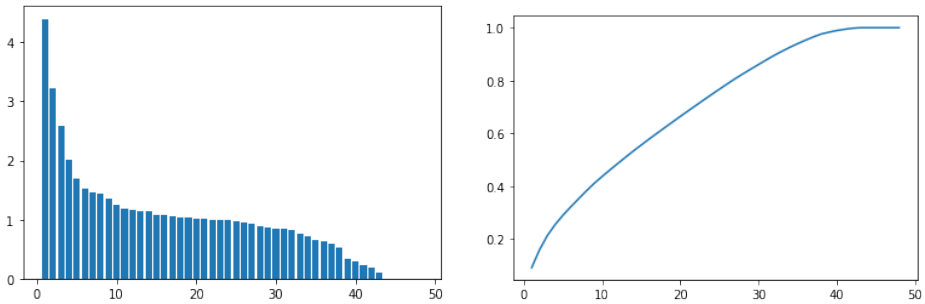

Figure 4.2: The explained variance and the cumulative sum of explained variance ratio

The screen plot suggest to choose 43 components. The cumulative sum of explained variance ratio with more than 80% of the explained variance, shows that taking only 30 components is enough.

Table 1. Results of Linear Least Square Fuzzy SVM model with and without PCA

Methode	bad precision	good precision	Type I	Type II	Total accuracy	AUC
linear LSFSVM without PCA	0.002	0.998	0.01	0.954	0.953	0.5
linear LSFSVM with PCA	0.344	0.897	0.145	0.964	0.87	0.62

Table 2. Results of Balanced Random Forest model with and without PCA

Methode	bad precision	good precision	Type I	Type II	Total accuracy	AUC
balanced RF without PCA	0.815	0.768	0.148	0.988	0.77	0.792
balanced RF with PCA	0.68	0.668	0.091	0.977	0.668	0.674

PCA helps in reducing number of interested dimensions in the data space. At the same time, it may affect the performance of modules by changing the data-space drastically. For LSFSVM model, the data

structure transformed by PCA makes LSFSVM perform better.

Nevertheless, this is not the case for Random Forest model for which we decided not to apply PCA in our experiments.

4.3 Performance evaluation

In this section, first we implement the statistical models and mathematical programming models in the library 'sklearn'. Then we implement some versions based on the SVM and RF.

The data set is randomly split with shuffle into two parts, 70% of the training subset for training the models and 30% of the test subset for prediction. The performance measures show the results of testing subset, which also present the performance of the model.

4.3.1 The performance of the classification models

Methode	bad precision	good precision	Type I	Type II	Total accuracy	AUC
Linear Regression	0.634	0.744	0.101	0.978	0.739	0.689
Linear Discriminant Analysis	0.634	0.745	0.102	0.978	0.74	0.69
Quadratic Discriminant Analysis	0.961	0.16	0.05	0.989	0.195	0.561
K Nearest Neighbor	0.488	0.85	0.129	0.973	0.834	0.669
Multilayer Perceptron	0.453	0.828	0.11	0.971	0.811	0.64
Decision Tree	0.611	0.786	0.118	0.978	0.778	0.698
Random Forest	**0.632**	**0.898**	0.211	0.983	0.887	**0.765**
Adaboost	0.456	0.918	0.201	0.974	0.898	0.687
Gaussian Naive Bayes	0.952	0.191	0.051	0.988	0.224	0.572
Support Vector Machine	0.283	0.995	0.716	0.968	**0.964**	0.639
LinearSVM	**0.661**	0.766	0.114	0.98	0.762	0.714
Gradient Boost	0.384	0.959	0.298	0.972	0.934	0.672

In general, each method has its advantages and disadvantages to build credit risk evaluation model. Total accuracy is not the unique criterion for measuring performance of the model, bad precision and good precision are also important criterion. If the bad precision is very low, the bank will loan to the bad clients, resulting in loss of profits. At the same time, improving the bad precision should not sacrifice the good precision. Because the good clients' class is the majority class, then, even a fraction of a percent decrease in good precision is still significant. The bank will lose the opportunity to benefit from good clients.

As we can see from the table above, Support Vector Machine classifier and Random Forest classifier have higher total accuracy, the bad precision and the good precision are acceptable. Therefore, we will study in detail some model versions based on SVM and RF.

4.3.2 Comparison of encoding methods

In this part, we compare the 3 encoding models (one-hot encoding, ordinal encoding and additive encoding) with different SVM-based models.

With respect to the three encoding methods, the accuracy of FBSVM methods is lower in one hot encoding and additive encoding. It means that the above method is more effective in the condition of variables qualitative with ordinal encoding. LSSVM, LSFSVM, WLSSVM and LSBFSVM perform well in the above 3 encoding methods. Among them, LSFSVM has the highest accuracy. However, considering the type I and type II, the high accuracy of LSFSVM largely comes from paying attention to the majority class, and reducing the precision of the minority class.

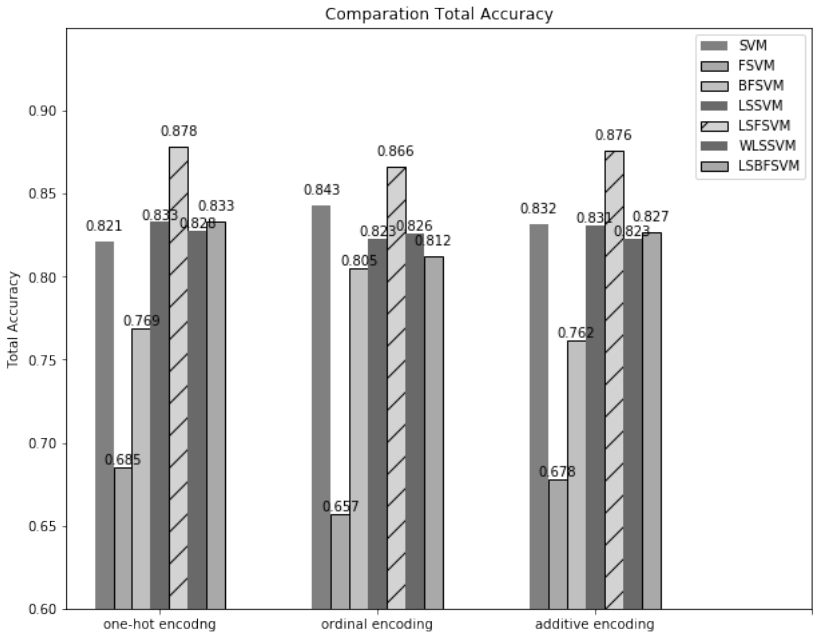

Figure 4.3: Comparison of encoding model. The training models are SVM, FSVM, BFSVM, LSSVM, LSFSVM, WLSSVM, LSBFSVM.

LSFSVM gives different weights to samples and WLSSVM gives different weights to classes, which increase the accuracy of minority class while maintaining the high accuracy of majority class. In total accuracy, LSFSVM performs greatly better than WLSSVM (about 4% higher than WLSSVM). However, considering the bad precision, WLSSVM performs slightly better than LSFSVM (about 1% higher than LSFSVM).

4.3.3 The performance of SVM and SVM based models

Methode	bad precision	good precision	Type I	Type II	Total accuracy	AUC
SVM	0.294	0.887	0.129	0.956	0.855	0.59
LinearSVM	0.51	0.818	0.138	0.967	0.801	0.664
Fuzzy SVM	**0.625**	0.716	0.106	0.973	0.711	0.67
Bilateral Fuzzy SVM	**0.549**	0.781	0.126	0.968	0.769	0.665
LSSVM	0.392	0.859	0.137	0.961	0.833	0.625
LS-Fuzzy SVM	0.37	**0.891**	0.148	0.966	**0.866**	0.63
Weighted LSSVM	0.412	0.852	0.137	0.962	0.828	0.632
LS Bilateral Fuzzy SVM	**0.444**	**0.845**	0.126	0.968	0.826	0.645
SVM&bagging	0.333	0.888	0.145	0.959	0.858	0.611
FuzzySVM&bagging	0.927	0.315	0.058	0.99	0.342	0.621
LS-FuzzySVM&bagging	0.213	0.879	0.085	0.955	0.846	0.546

SVM and LinearSVM model are classification models in library 'sklearn' and SVM-based models concludes Fuzzy SVM (FSVM), Bilateral Fuzzy SVM (BFSVM), Least Square SVM (LSSVM), Least Square Fuzzy SVM (LSFSVM), Weighted Least square SVM (WLSSVM), Least Square Bilateral Fuzzy SVM (LSBFSVM), SVM & bagging, Fuzzy SVM & bagging and Least Square Fuzzy SVM & bagging. The encoding method is one-hot encoder.

From the table above, we can observe that FSVM and BFSVM have higher bad precision, but the good precision is low. LSSVM, LSFSVM, WLSSVM and LSBFSVM have higher total accuracy. Considering the trade-off between the bad precision and good precision, LSFSVM and LSBFSVM perform better.

4.3.4 The performance of Random Forest and RF based models

Weighted Random Forest

Weighted Random Forest (WRF) is an optimized version of Random Forest for imbalanced data. This approach follows the idea of cost sensitive learning. Since the RF classifier tends to be biased towards the majority class, we shall place a heavier penalty on misclassifying the minority class. Weighted Random Forest assign a weight to each class, with the minority class given larger weight (i.e., higher misclassification cost).

Bad clients weight=4; Good clients weight=0.2.

Balanced Random Forest

Balanced Random Forest (BRF) randomly under-samples each bootstrap sample, artificially altering the class distribution so that classes are represented equally in each tree.

BRF is computationally more efficient with large imbalanced data, since each tree only uses a small portion of the training set to grow, while WRF needs to use the entire training set. WRF assigns a weight to the minority class, possibly making it more vulnerable to noise (mis-labeled class) than BRF. A majority case that is mislabled as belonging to the minority class may have a larger effect on the prediction accuracy of the majority class in WRF than in BRF.

SMOTE and Random Forest

The Synthetic Minority OverSampling Technique (SMOTE) is a method for sampling data. The principle is: for a minorty class sample x, randomly choose a K nearest neighbor sample x_k (same class with x). Generate the new sample with $x_{new} = x + \lambda * (x_k - x)$, where λ is a random value between 0 and 1.

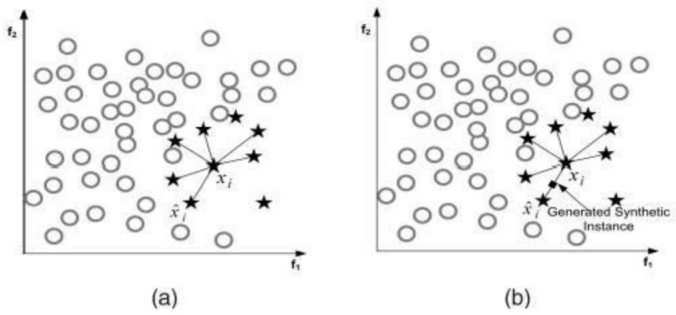

Figure 4.4: An example of SMOTE

Borderline SMOTE is an improved SMOTE method which divides the minority samples into 3 categories, namely Safe, Danger and Noise. Then the method consists of giving these three categories different weights and generating the different numbers of samples.

Methode	bad precision	good precision	Type I	Type II	Total accuracy	AUC
Random Forest	0.792	0.815	0.166	0.988	0.814	0.804
Balanced Random Forest	**0.829**	0.796	0.152	0.99	0.808	**0.807**
Weighted Random Forest	0.79	0.82	0.17	0.988	0.819	0.805
Borderline SMOTE RF	0.526	0.894	0.188	0.976	**0.878**	0.71

Random Forest classifiers perform better in bad precision and AUC. Balanced Random Forest (BRF) has higher bad precision than Weighted Random Forest (WRF). Compared with Random Forest, Borderline SMOTE RF improve the good precision.

BRF has advantages for extremely imbalanced data, since each tree only uses a small portion of the training set to grow, draw a bootstrap sample from the minority class. Randomly draw the same number of cases, with replacement, from the majority class. BRF has the highest bad precision but Type I is very low. That means many good clients are mis-classified as bad clients. In the next subsection, we design a model to try to solve this problem by combining prediction probability and threshold of BRF.

4.3.5 Graphic Distribution

Graphic distribution of the actual performance (good and bad) across the predicted probabilities of default is a way to visually evaluate the performance of a model. The more separated these distributions are, the more accurate the model.

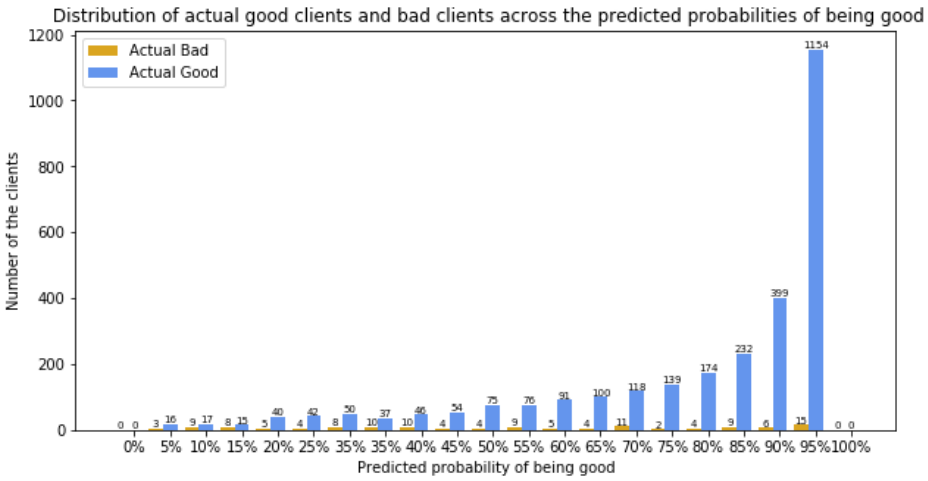

Figure 4.5: *Distribution of actual good clients and bad clients in* ***Least Square Fuzzy SVM model***

Compared Figure 4.5 and Figure 4.6, we observe that LSFSVM model does not clearly separates the two classes. Most of the actual good clients are predicted close to 95%. However the actual bad clients are predicted in the entier probability interval. In this case, the data set with Least Square Fuzzy SVM does not perform good.

In the Figure 4.6, the actual bad clients are predicted mainly distributed in the interval [25%, 50%], and there are no actual bads beyond a predicted default of 70%. The actual good clients are predicted mainly distributed in the interval [45%, 75%]. Therefore, we could set the clients predicted probability [0, 70%] uncertain.

We consider now on Balanced Random Forest Model. Improved Balanced Random Forest model is inspired by the graphic distribution. After exceeding a threshold, almost all the actual good clients in this interval are correctly predicted. Therefore, we could remove the samples in this part ([threshold,1]) who is certain. And Re-enter the remaining samples into model to classify.

Figure 4.6: *Distribution of actual good clients and bad clients in* ***Balanced Random Forest model***

This model includes 3 Balanced Random Forest models. After the first classifier, remove some predicted good clients with high predicted probability. The remaining samples enter the second classifier, also remove some predicted good clients with high predicted probability. The remaining samples of the second classifier enter the third classifier. The final results is the mean of these three classifiers.

The object is gradually increasing the ratio of good clients and bad clients by removing predicted good clients with high predicted probability and improving the accuracy of the model. The training set (70% of data set) trains the model, and the threshold is selected according to the graphic distribution of the training set. The clients counter below represents the change in the number of classes when the test set (30% of data set) passes through three classifiers.

Classifier 1 Clients Counter(Good clients: 2896, Bad clients: 139)
Classifier 2 Clients Counter(Good clients: 1572, Bad clients: 137)
Classifier 3 Clients Counter(Good clients: 1459, Bad clients: 136)

Figure 4.7: *Distribution of actual good clients and bad clients in* ***Balanced Random Forest model of Classifier 1***

Figure 4.8: *Distribution of actual good clients and bad clients in* ***Balanced Random Forest model of Classifier 2***

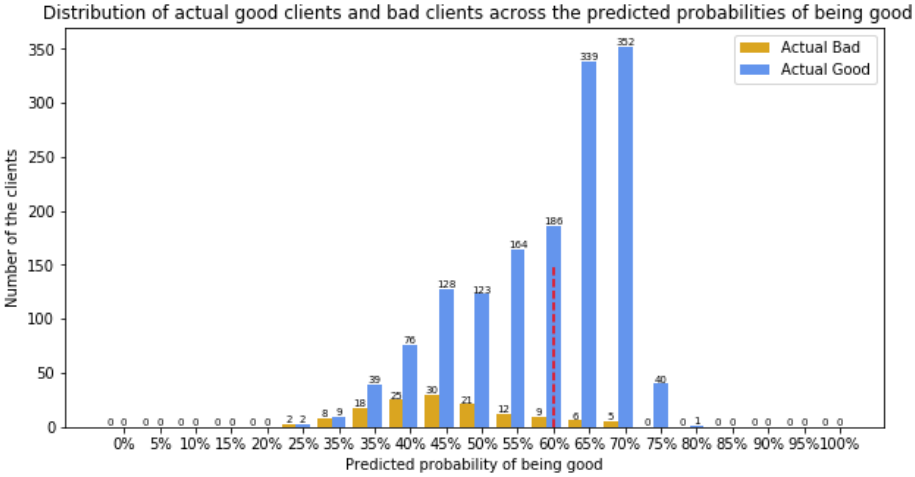

Figure 4.9: *Distribution of actual good clients and bad clients in* ***Balanced Random Forest model of Classifier 3***

Methode	bad precision	good precision	Type I	Type II	Total accuracy	AUC
Benchmark	0.827	0.769	0.15	0.99	0.772	0.805
Model	0.775	0.845	0.186	0.986	0.841	0.802

Confusion matrix:

Banchmark		Model	
113	20	105	28
633	2269	470	2432

Figure 4.7, 4.8, 4.9 corresponds to the distribution after three classifications. The red dotted lines in classifier1 and classifier2 indicate the threshold. They are at 70%. If the predicted probability of the samples are greater than 70%, we consider these samples as good clients. The remaining samples (uncertain part) whose predicted probabilities are less than 70% enter to the next classifier. Figures 4.7 and 4.8 are not much different, it means that the uncertain part will not continue to be separated as the number of classifications increases.

The confusion matrix shows that the Improved Balanced Random Forest greatly improves the accuracy of good clients (2432) and less mis-classify the good clients as bad clients (470).

4.3.6 Performance of other models

According to models mentioned in [8], the random forest and gradient boosting classifiers perform very well in a credit scoring context and are able to cope comparatively well with pronounced class imbalances in these data sets.

So we compare the result of gradient boosting classification model, XGboost algorithm model and random forest classification model. Gradient boosting classifier [17] is the classification class of gradient boosting decision tree. eXtreme Gradient Boosting (XGboost) [9] algorithm model is a kind of gradient boosting decision tree model, which integrates many tree models together to form a strong classifier. The tree model used is the CART regression tree model. Random forest [19] is a classifier that contains multiple decision trees, and its output category is determined by the mode of the category output by individual trees. Directly call XGboost model and the package of RandomForestClassifier, GradientBoostingClassifier in sklearn.

Methode	bad precision	good precision	Type I	Type II	Total accuracy	AUC
Gradient Boost	0.39	0.955	0.292	0.97	0.929	0.673
XGBoost	0.414	0.948	0.278	0.971	0.924	0.681
Random Forest	0.77	0.814	0.16	0.987	0.812	0.792

According to the above table Gradient boost, XGboost and Random forest, Gradient boost model has the highest accuracy rate, followed by XGboost. For bad precision, Random Forest performs the best. However, Type I of the Random Forest needs to be improved. Since Random Forest predicts more the good clients into bad clients. Comparing with the results of SVM-based models in section 4.3.3 and RF-based models in 4.3.4, tree-based models like random forest could be the best choice.

4.4 Conclusion

We preprocessed the data by removing the outliers, transforming qualitative variables with one-hot encoding. Then we implement SVM-based models and tree-based models for classifying.

The difference over the encoding methods (one-hot encoding, ordinal encoding and additive encoding) is not high for SVM-based models. Still, when making the distance calculation between features more reasonable, one-hot encoding is better. One-hot encoding has the capacity of expanding the features. But when the number of categories is large, the feature space becomes very large. In this case, PCA is used to reduce the dimensionality to make the SVM-based model perform better.

The SVM-based models are Fuzzy SVM, Bilateral Fuzzy SVM, Least Square SVM, Least Square Fuzzy SVM, Weighted Least Square SVM, and Least Square Bilateral Fuzzy SVM. After applying over sampling for balancing data, LSSVM, LSFSVM, and WLSSVM performs good in total accuracy. FSVM and BFSVM have higher bad precision than SVM in the library. The combined model of SVM-based model and Bootstrap aggregation depends largely on the choice of training set and the results perform more robust. But the precision of minority class has no significant improvement.

Because of the extremely unbalanced characteristics of this data, Random Forest models perform better than SVM models. Both the minority class and the majority class achieve an accuracy rate of about 80%. Balanced Random Forest model has the highest bad precision. We also compared gradient boosting model, eXtreme gradient boosting model and random forest model which are tree-based models. Among them, Random Forest has the highest bad precision 77% and the lowest total accuracy 81%. Gradient Boost has the highest total accuracy 93%. XGBoost improves the precision of minority class, while maintaining the precision of majority class.

However, considering the interpretability characteristics of financial data, the choice of model is based on the needs of the bank. If the bank wants accurately predict the bad clients, so that the bank loan will be repaid in time, RF is a good choice. If the bank wants to accurately predict good clients and increase the bank's profits, it can consider gradient boosting model, eXtreme gradient boosting model. For further research, tree-based model could be a research direction of financial data.

Chapter 5

Implementation and integration of Machine Learning model

5.1 Implementation into a web application

We have implemented our Machine Learning model into a simple web application. The architecture and the main functionalities of this application are shown in the image below.

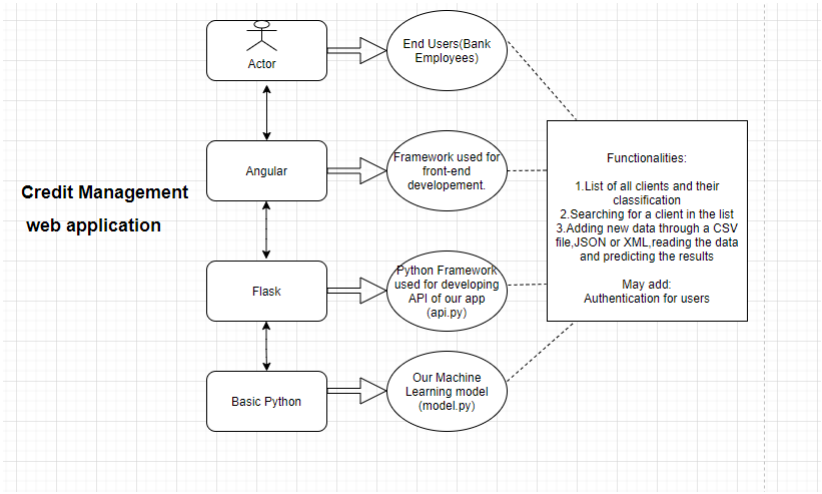

Figure 5.1: *Architecture and functionalities*

5.1.1 Architecture

The architectural style used for developing the web application is Representational State Transfer (REST). The main project contains two sub projects. One project contains the frontend of our application and the other contains the backend services, representing respectively the client who requests the resources and the server who has the resources.

REST

REST, or REpresentational State Transfer, is an architectural style for providing standards between computer systems on the web, making it easier for systems to communicate with each other. REST-compliant systems, often called RESTful systems, are characterized by how they are stateless and separate the concerns of client and server.

In the REST architectural style, the implementation of the client and the implementation of the server can be done independently without each knowing about the other. This means that the code on the client side can be changed at any time without affecting the operation of the server, and the code on the server side can be changed without affecting the operation of the client.
As long as each side knows what format of messages to send to the other, they can be kept modular and separate. Separating the user interface concerns from the data storage concerns, we improve the flexibility of the interface across platforms and improve scalability by simplifying the server components. Additionally, the separation allows each component the ability to evolve independently.
By using a REST interface, different clients hit the same REST endpoints, perform the same actions, and receive the same responses.

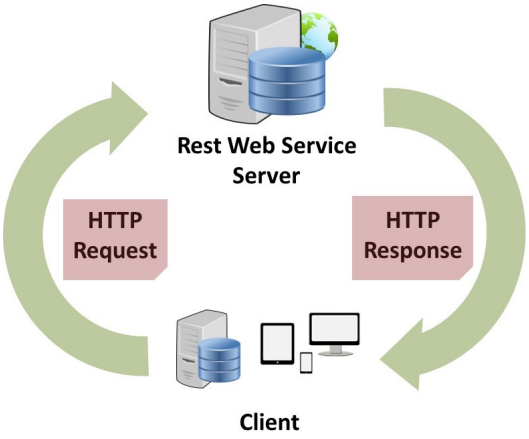

Figure 5.2: *Rest architecture*

Angular framework

Angular is a development platform and application design framework used for developing sophisticated and efficient single-page apps, while Angular Material is an UI component library which offers a variety of ready-to-use components. For developing the frontend of our app we have used some Material components such as forms and tables.

Flask framework

Flask is a web framework which provides tools, libraries and technologies for building web applications. It is written in Python and it is considered as a microframework for not requiring particular tools or libraries. We have used Flask for developing the backend of our project. Flask handles Http Requests coming from the front-end project, while communicating with basic python files which contain our Machine Learning model.

Pickle module

To avoid training our model every time we shut down our Python session, we need to save the classifier we trained and built. For this purpose we have used Python's in-built pickle module which allows us to serialize and deserialize Python objects to compact byte code.

5.1.2 Functionalities

Our app contains 4 basic elements: some forms, a file chooser, a table and a chart. The forms are used for choosing the criteria of predicting while the file chooser for selecting the new data to add (mainly in csv format).

The table lists all the clients, their classification as a good or a bad client and the probability of being a good client, while the chart shows the importance of each feature in deciding whether a client is a good or a bad client. For the chart component we have used Ng2-charts library.

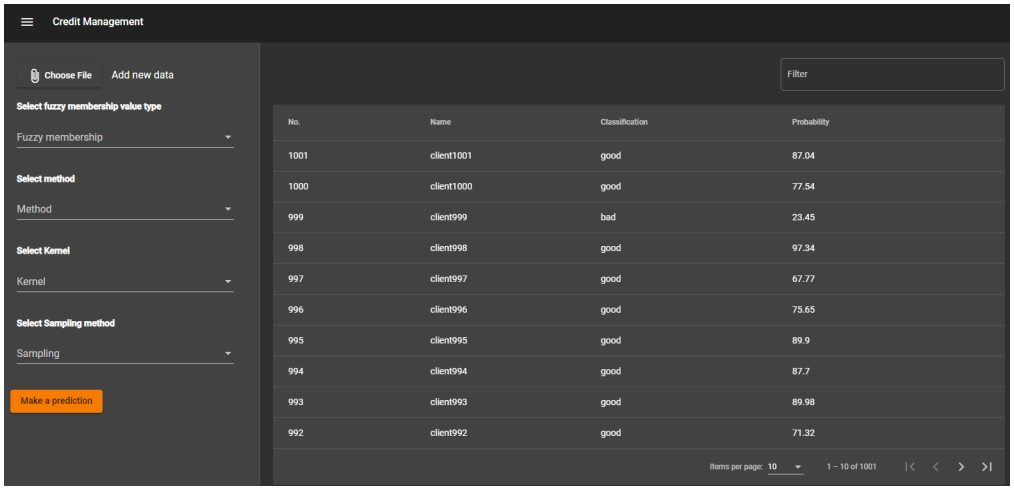

Figure 5.3: *Frontend of the application*

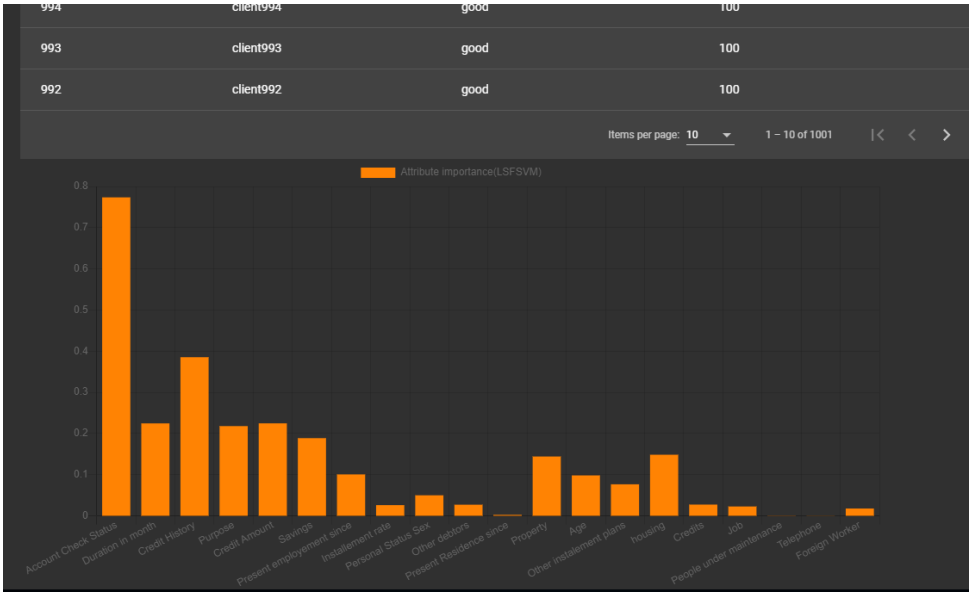

Figure 5.4: *Chart showing the importance of each feature*

5.2 Integration with banking software

For managing credits many banks still use software that can not be integrated with web applications. That's why we have developed a simple algorithm for integrating the machine learning model within the bank software. The algorithm is as follows:

PROGRAM

START

READ client data
SET array client_data with data read

Function Data_Transformations:
Pass in:client_data
TRANSFORM client_data categorical values into binary values
READ file PCA.csv //contains coefficients for reducing data dimensions
SET array means with the last column of file
SET matrix components_T with the rest of the columns of file
READ file Scaler.csv
SET array scaler_scale with the first column of file

SET array scaler_min with the second column of file CALCULATE:
transformed_data=((client_data - means)x(components_T))*scaler_scale+scaler_min
Pass out:transformed_data
EndFunction

Function predict_client:
Pass in:transformed_data
READ file LSFSVM.csv //contains coefficients of the algorithm
used for classification
SET array alpha with alpha column of the file
SET array y with Y column of the file
SET b variable with the value of b column
SET A variable with the value of A column
SET B variable with the value of B column
SET sigma variable with the value of K.Sigma column
READ file X_train.csv(containing all data already trained)
SET matrix xTrain with all the data read from the file
CALCULATE:
//.T refering to transposed vector
K0=(xTrain*xTrain+1)+(transformed_data*transformed_data+1).T-2*(transformed_data X xTrain.T)
A=alpha*y
prediction=b+power(exp(-1.0 / (2*sigma$\hat{2}$)), K0) X A
probability= 1/(1+exp(A*prediction+B))
Pass out: sign of prediction(-1 for bad clients and 1 for good clients) and probability
EndFunction

END

Bibliography

[1] Ali Al-Aradi. Credit Scoring via Logistic Regression. 2014.

[2] Bart Baesens, Tony Van Gestel, Stijn Viaene, Maria Stepanova, Johan Suykens, and Jan Vanthienen. Benchmarking state-of-the-art classification algorithms for credit scoring. *Journal of the operational research society*, 54(6):627–635, 2003.

[3] John Banasik, Jonathan Crook, and Lyn Thomas. Sample selection bias in credit scoring models. *Journal of the Operational Research Society*, 54(8):822–832, 2003.

[4] Rukshan Batuwita and Vasile Palade. Fsvm-cil: fuzzy support vector machines for class imbalance learning. *IEEE Transactions on Fuzzy Systems*, 18(3):558–571, 2010.

[5] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152, 1992.

[6] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

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

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

[9] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[10] Leopoldo Soares de Melo Junior, Franco Maria Nardini, Chiara Renso, and José Antônio Fernandes de Macêdo. An empirical comparison of classification algorithms for imbalanced credit scoring datasets. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 747–754. IEEE, 2019.

[11] Gheorghita Dinca and Madalina Bociu. Using discriminant analysis for credit decision. *Bulletin of the Transilvania University of Brasov. Economic Sciences. Series V*, 8(2):277, 2015.

[12] Richard O Duda, Peter E Hart, and David G Stork. *Pattern classification*. John Wiley & Sons, 2012.

[13] Robert A Eisenbeis. Problems in applying discriminant analysis in credit scoring models. *Journal of Banking & Finance*, 2(3):205–219, 1978.

[14] M A H Farquad, V Ravi, and G Praveen. Credit Scoring using PCA-SVM hybrid model. In *International Conference on Advances in Communication, Network, and Computing*, pages 249–253. Springer, 2011.

[15] Ronald A Fisher. The use of multiple measurements in taxonomic problems. 1(1):1–8, 1936.

[16] David J Fogarty. Using genetic algorithms for credit scoring system maintenance functions. *International Journal of Artificial Intelligence & Applications*, 3(6):1, 2012.

[17] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[18] Halina Frydman, Edward I Altman, and Duen-Li Kao. Introducing recursive partitioning for financial classification: the case of financial distress. *The Journal of Finance*, 40(1):269–291, 1985.

[19] Tin Kam Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE, 1995.

[20] Cheng-Lung Huang, Mu-Chen Chen, and Chieh-Jen Wang. Credit scoring with a data mining approach based on support vector machines. *Expert systems with applications*, 33(4):847–856, 2007.

[21] Irina Ioniţă and Daniela Şchiopu. Using principal component analysis in loan granting. *Buletinul Universităţii Petrol-Gaze din Ploieşti*, 62(1.2010), 2010.

[22] Dimitris Karlis and Mohieddine Rahmouni. Analysis of defaulters' behaviour using the Poisson-mixture approach. *IMA Journal of Management Mathematics*, 18(3):297–311, 2007.

[23] Erkki K Laitinen and Teija Laitinen. Bankruptcy prediction: Application of the Taylor's expansion in logistic regression. *International review of financial analysis*, 9(4):327–349, 2000.

[24] Brett Lantz. *Machine learning with R*. Packt Publishing Ltd, 2013.

[25] Tian-Shyug Lee, Chih-Chou Chiu, Chi-Jie Lu, and I-Fei Chen. Credit scoring using the hybrid neural discriminant technique. *Expert Systems with applications*, 23(3):245–254, 2002.

[26] Chun Fu Lin and Sheng De Wang. Fuzzy support vector machines. *IEEE Transactions on Neural Networks*, 13(2):464–471, 2002.

[27] Hsuan-Tien Lin, Chih-Jen Lin, and Ruby C Weng. A note on platt's probabilistic outputs for support vector machines. *Machine learning*, 68(3):267–276, 2007.

[28] Francisco Louzada, Anderson Ara, and Guilherme B. Fernandes. Classification methods applied to credit scoring: Systematic review and overall comparison. *Surveys in Operations Research and Management Science*, 21(2):117–134, 2016.

[29] Zhao Min. Credit risk assessment based on fuzzy svm and principal component analysis. In *2009 International Conference on Web Information Systems and Mining*, pages 125–127. IEEE, 2009.

[30] M A Mukid, T Widiharih, A Rusgiyono, and A Prahutama. Credit scoring analysis using weighted k nearest neighbor. In *Journal of Physics: Conference Series*, volume 1025, page 12114. IOP Publishing, 2018.

[31] M Artís Ortuño, Montserrat Guillén, and JOSÉ Ma Martínez. A model for credit scoring: an application of discriminant analysis. *Qüestiió: quaderns d'estadística i investigació operativa*, 18(3), 1994.

[32] Su-Lin Pang, Yan-Ming Wang, and Yuan-Huai Bai. Credit scoring model based on neural network. In *Proceedings. International Conference on Machine Learning and Cybernetics*, volume 4, pages 1742–1746. IEEE, 2002.

[33] John Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.

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

[35] Kyung Shik Shin, Talk Soo Lee, and Hyun Jung Kim. An application of support vector machines in bankruptcy prediction model. *Expert Systems with Applications*, 28(1):127–135, 2005.

[36] J. A.K. Suykens, J. De Brabanter, L. Lukas, and J. Vandewalle. Weighted least squares support vector machines: Robustness and sparce approximation. *Neurocomputing*, 48(1-4):85–105, 2002.

[37] Gang Wang, Jian Ma, Lihua Huang, and Kaiquan Xu. Two credit scoring models based on dual strategy ensemble trees. *Knowledge-Based Systems*, 26:61–68, 2012.

[38] Gang Wang, Jian Ma, Lihua Huang, and Kaiquan Xu. Two credit scoring models based on dual strategy ensemble trees. *Knowledge-Based Systems*, 26:61–68, 2012.

[39] David West. Neural network credit scoring models. *Computers & Operations Research*, 27(11-12):1131–1152, 2000.

[40] Lean Yu. Credit risk evaluation with a least squares fuzzy support vector machines classifier. *Discrete Dynamics in Nature and Society*, 2014, 2014.

[41] Lean Yu, Shouyang Wang, Kin Keung Lai, and Ligang Zhou. *Bio-inspired credit risk analysis: Computational intelligence with support vector machines.* Number 71473155. 2008.

[42] Li Zhan, Xu Ji-sheng, and Xu Min. ANN-GA approach of credit scoring for mobile customers. In *IEEE Conference on Cybernetics and Intelligent Systems, 2004.*, volume 2, pages 1148–1153. IEEE, 2004.

[43] Houshmand A Ziari, David J Leatham, and Calum G Turvey. Application of Mathematical Programming Techiniques in Credit Scoring of Agricultural Loans. Technical report, 1994.

Appendix A

Lagrange Multiplier comparison

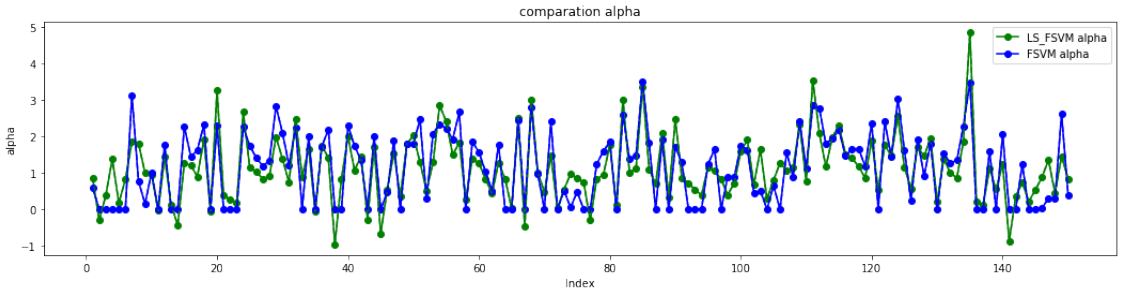

Figure A.1: The section of Lagrangian Multiply of Least Square Fuzzy SVM and Fuzzy SVM

Compared to Fuzzy SVM (FSVM) and Least Square Fuzzy SVM (LSFSVM), the Lagrangian multipliers of FSVM are all greater than or equal to zero, and many of them are equal to zero. The Lagrangian multiplier of LSFSVM has positive and negative numbers.

Appendix B

The code in Python

B.1 Fuzzy SVM

```
import cvxopt

def rbf_kernel(x,sigma=1.0):
   return np.exp((-linalg.norm(x - x) ** 2) / (2 * (sigma ** 2)))

Kernel[i, j] = rbf_kernel(X[i], X[j], sigma)

n_samples, n_features = X.shape

P = cvxopt.matrix(np.outer(y, y) * Kernel)
q = cvxopt.matrix(np.ones(n_samples) * -1)
A = cvxopt.matrix(y, (1, n_samples))
b = cvxopt.matrix(0.0)

tmp1 = np.diag(np.ones(n_samples) * -1)
tmp2 = np.identity(n_samples)
G = cvxopt.matrix(np.vstack((tmp1, tmp2)))
tmp1 = np.zeros(n_samples)
tmp2 = np.ones(n_samples) * membership_value * C
h = cvxopt.matrix(np.hstack((tmp1, tmp2)))

solution = cvxopt.solvers.qp(P,q,G,h,A,b)
alpha = np.ravel(solution['x'])

for i in range(n_samples):
    sv = np.logical_and(alpha < membership_value * C, alpha > 1e-5)

sv_alpha = alpha[sv]
sv_x = X[sv]
sv_y = y[sv]

b = 0  #calculate b
for n in range(len(alpha)):
    b += self.sv_y[n]
    b -= np.sum(sv_alpha * sv_y * Kernel[ind[n], sv])
b /= len(alpha)

# decision function
w_phi = np.zeros(len(X))
for i in range(len(X)):
    w_phi[i] = 0
    w_phi[i] += sv_alpha * sv_y * rbf_kernel(X[i], sv, sigma)

z = np.sign(w_phi + b)

```

B.2 Least Square Fuzzy SVM

```
from numpy import linalg

H = np.multiply(np.dot(np.matrix(y).T, np.matrix(y)), Kernel)
Omiga_ij = H + np.eye(X.shape[0]) / (C * membership_value)
Col_L = np.concatenate((omiga, np.matrix(Y).T), axis=0)
Col_R = np.concatenate((np.matrix(Y), np.matrix(0)), axis=0)

A = np.concatenate((Col_L, Col_R), axis=1)
B = np.ones(X.shape[0] + 1)
B[-1] = 0

solution = linalg.solve(A, B)
alpha = solution[:-1]
b = solution[-1]

# decision function
w_phi = np.zeros(len(X))
for i in range(len(X)):
    w_phi[i] = 0
    w_phi[i] += alpha * y * rbf_kernel(X[i], sigma)

z = np.sign(w_phi + b)

```