brought to you by CO
provided by Aaltodoc Publication Arc

Machine Learning for Corporate Bankruptcy Prediction

Qi Yu

Aalto University

DOCTORAL DISSERTATIONS

Aalto University publication series
DOCTORAL DISSERTATIONS 90/2013

Machine Learning for Corporate Bankruptcy Prediction

Qi Yu

A doctoral dissertation completed for the degree of Doctor of Science to be defended, with the permission of the Aalto University School of Science, at a public examination held at the lecture hall AS2 in TUAS building in Otaniemi campus on 24 May 2013 at 12.

Aalto University
School of Science Aalto University
Department of Information and Computer Science

Supervising professor
Prof. Olli Simula

Thesis advisors
Dr. Amaury Lendasse
Prof. Eric Severin

Preliminary examiners
Prof. Dominique Haughton, Bentley University, USA
Dr. Ignacio Rojas, University of Granada, Spain

Opponents
Prof. Christophe Biernacki, University of Lille 1, France

Aalto University publication series
DOCTORAL DISSERTATIONS 90/2013

© Qi Yu

ISBN 978-952-60-5186-4 (printed)
ISBN 978-952-60-5187-1 (pdf)
ISSN-L 1799-4934
ISSN 1799-4934 (printed)
ISSN 1799-4942 (pdf)
http://urn.fi/URN:ISBN:978-952-60-5187-1

Unigrafia Oy
Helsinki 2013

Finland

441 697
Printed matter

Aalto University

Abstract

Aalto University, P.O. Box 11000, FI-00076 Aalto www.aalto.fi

Author
Qi Yu

Name of the doctoral dissertation
Machine Learning for Corporate Bankruptcy Prediction

Publisher School of Science

Unit Information and Computer Science Department

Series Aalto University publication series DOCTORAL DISSERTATIONS 90/2013

Field of research Information and Computer Science

Manuscript submitted 10 March 2013 **Date of the defence** 24 May 2013

Permission to publish granted (date) 16 April 2013 **Language** English

☐ **Monograph** ☒ **Article dissertation (summary + original articles)**

Abstract

Corporate bankruptcy prediction has long been an important and widely studied topic, which is of a great concern to investors or creditors, borrowing firms or governments. Especially due to the recent change in the world economy and as more firms, large and small, seem to fail now more than ever. The prediction of the bankruptcy, is then of increasing importance.

There has been considerable interest in using financial ratios for predicting financial distress in companies since the seminal works of Beaver using an univariate analysis and Altman approach with multiple discriminant analysis. The big amount of financial ratios makes bankruptcy prediction a difficult high-dimensional classification problem. So this dissertation presents a way for ratio selection which determines the parsimony and economy of the models and thus the accuracy of prediction. With the selected financial ratios, this dissertation explores several Machine Learning methods, aiming at bankruptcy prediction, which is addressed as a binary classification problem (bankrupt or non-bankrupt companies). They are OP-KNN (Publication I), Delta test-ELM (DT- ELM) (Publication VII) and Leave-One-Out-Incremental Extreme Learning Machine (LOO-IELM) (Publication VI). Furthermore, soft classification techniques (classifier ensembles and the usage of financial expertise) are used in this dissertation. For example, Ensemble K-nearest neighbors (EKNN) in Publication V, Ensembles of Local Linear Models in Publication IV, and Combo and Ensemble model in Publication VI. The results reveal the great potential of soft classification techniques, which appear to be the direction for future research as core techniques that are used in the development of prediction models.

In addition to selecting ratios and models, the other foremost issue in experiments is the selection of datasets. Different studies have used different datasets, some of which are publicly downloadable, some are collected from confidential resources. In this dissertation, thanks to Prof. Philippe Du Jardin, we use a real dataset built for French retails companies. Moreover, a practical problem, missing data, is also considered and solved in this dissertation, like the methods shown in Publication II and Publication VIII.

Keywords Bankruptcy Prediction, Machine Learning, Extreme Learning Machine, Variable Selection

ISBN (printed) 978-952-60-5186-4 **ISBN (pdf)** 978-952-60-5187-1

ISSN-L 1799-4934 **ISSN (printed)** 1799-4934 **ISSN (pdf)** 1799-4942

Location of publisher Espoo **Location of printing** Espoo, Finland **Year** 2013

Pages 111 **urn** http://urn.fi/URN:ISBN:978-952-60-5187-1

Preface

This dissertation is carried out at Department of Information and Computer Science (ICS), Aalto University and funded by Helsinki Graduate School in Computer Science and Engineering (Hecse), Adaptive Information and Research Center (AIRC), ICS department, as well as Research training scholarship, from Helsinki University of Technology (Aalto University). Thanks for all these organizations and more specially to Erkki Oja and Pekka Orponen, who offered sufficient financial support to this work.

I would like to thank my supervisor Olli Simula, who accepted me as his student and guided me for all these years. I am very grateful to his patience and thoughtful consideration which helped me a lot, especially in those tough moments. I would also like to thank my two instructors: Amaury Lendasse and Eric Séverin. I won't have chances to graduate without their generous helps and guidances. Many thanks to Philippe Du Jardin, who provided to me valuable resources and ideas.

In addition, many thanks to all the coauthors of my publications, I really appreciated the experience cooperating with them. It was memorable time I spent in the ICS department, I also want to thanks for everybody there, especially the members in Environmental and Industrial Machine Learning (EIML) Group.

I thank Christophe Biernacki for the honor of having him as an opponent, and also the pre-examiners Dominique Haughton and Ignacio Rojas. Thanks for all the help they offered to this dissertation.

Finally, I would like to thank my family for all their love and encouragement. For my parents who supported me in every possible ways and believed in me long after I lost belief in myself. For my daughter, Yuqing Hu (Sunny), who is so far my greatest achievement that nothing can be compared with. She brings me the joys much beyond my expecta-

tion. And most of all for my loving, supportive and encouraging husband, Zhongliang Hu, who is my most enthusiastic cheerleader and my best friend. I feel lucky and happy to have him accompanying on the whole journey.

Espoo, May 4, 2013,

Yu Qi

Contents

List of Publications

This thesis consists of an overview and of the following publications which are referred to in the text by their Roman numerals.

I Qi Yu, Yoan Miche, Antti Sorjamaa, Alberto Guillén, Amaury Lendasse and Eric Séverin. OP-KNN: Method and Applications. *Advances in Artificial Neural Systems*, Volume 2010, 6 pages, February 2010.

II Qi Yu, Yoan Miche, Email Eirola, Mark van Heeswijk, Eric Séverin and Amaury Lendasse. Regularized Extreme Learning Machine for Regression with Missing Data . *Neurocomputing*, Volume 102, pages 45-51, June 2012.

III Dusan Sovilj, Antti Sorjamaa, Qi Yu, Yoan Miche and Eric Séverin. OPELM and OPKNN in long-term prediction of time series using projected input data . *Neurocomputing*, Volume 73, pages 1976-1986, June 2010.

IV Laura Kainulainen, Yoan Miche, Emil Eirola, Qi Yu, Benoit Frénay, Eric Séverin and Amaury Lendasse. Ensembles of Local Linear Models for Bankruptcy Analysis and Prediction. *Case Studies in Business, Industry and Government Statistics*, Volume 4, November 2011.

V Qi Yu, Amaury Lendasse and Eric Séverin. Ensemble KNNs for Bankruptcy Prediction. In *CEF 09, 15th International Conference: Computing in Economics and Finance*, Sydney, Australia, pages 78-81, June 15-17 2009.

VI Qi Yu, Yoan Miche, Eric Severin and Amaury Lendasse. Bankruptcy Prediction using Extreme Learning Machine and Financial Expertise. Accepted for publication in *Neurocomputing*, January 2013.

VII Qi Yu, Mark van Heeswijk, Yoan Miche, Rui Nian, He Bo, Eric Séverin and Amaury Lendasse. Ensemble Delta test- Extreme Learning Machine (DT-ELM) For Regression. Accepted for publication in *Cognitive Computation*, February 2013.

VIII Qi Yu, Yoan Miche, Amaury Lendasse and Eric Séverin. Bankruptcy Prediction with Missing Data. In *DMIN11: International Conference on Data Mining*, Las Vegas, USA, pages 279-285, July 18-21 2011.

Author's Contribution

Publication I: "OP-KNN: Method and Applications"

This journal paper presents a methodology named Optimally Pruned K-nearest neighbor (OP-KNN) which has the advantage of competing with state-of-the-art methods while remaining fast. It builds a one hidden-layer feedforward neural network using K-nearest neighbor as kernels to perform regression. Multiresponse Sparse Regression (MRSR) is used in order to rank each kth nearest neighbor and finally Leave-One-Out estimation is used to select the optimal number of neighbors and to estimate the generalization performances. Since computational time of this method is small, this paper also presents a strategy for Variable Selection using OP-KNN. The author carried out the writing and experiments work for this publication. The other authors provided useful suggestions and corrections to the paper.

Publication II: "Regularized Extreme Learning Machine for Regression with Missing Data "

This journal paper proposes a method which is the advanced modification of the original extreme learning machine with a new tool for solving the missing data problem. It uses a cascade of L1 penalty (LARS) and L2 penalty (Tikhonov regularization) on ELM (called TROP-ELM [70]) to regularize the matrix computations and on the other hand, it estimates the expected pairwise distances between samples directly on incomplete data. The entire algorithm shows its significant advantages: fast computational speed, no parameter need to be tuned, good generalization performance and the ability to solve missing data problem. The experiments

and writing of this paper were carried out mainly by the author, with the help of Yoan Miche and Emil Eirola who are the original developers of OPELM and Pairwise distance estimation.

Publication III: "OPELM and OPKNN in long-term prediction of time series using projected input data "

This journal paper presents a methodology that uses input processing before building the model for Long-term time series prediction. Input processing aims is to reduce the number of input variables or features and in this publication, the combination of the Delta test and the genetic algorithm is used to obtain two aspects of reduction: scaling and projection. After input processing, two fast models are used to make the predictions: Optimally Pruned Extreme Learning Machine (OP-ELM) and Optimally Pruned K-nearest neighbor (OP-KNN). Both models have fast training times, which makes them suitable choice for direct strategy for long-term prediction. This publication was a joint work and the author did the part of the experiments and writing related to Financial data.

Publication IV: "Ensembles of Local Linear Models for Bankruptcy Analysis and Prediction"

This journal paper presents an ensemble methodology, aiming for bankruptcy prediction. It builds ensembles of locally linear models using a forward variable selection technique and provides information about the importance of the variables. Therefore, the main advantage of the method is that the variable selection embedded into the method provides good interpretability of the results. The author contributed about 40% to the publication, the experiment and writing work was mainly done by Laura Kainulainen.

Publication V: "Ensemble KNNs for Bankruptcy Prediction"

A global methodology for bankruptcy prediction is presented in this conference paper. The computational time of this method is extremely small since it uses K-nearest neighbors (k-NN) to build several classifiers, each of the classifiers uses different nearest neighbor on different subset of in-

put variables and try to minimize the mean square error. Finally a linear combination of these classifiers is calculated to get even better performance. On the other hand, this method is robust because the ensemble of classifiers has smaller variance than each single classifier. The result from real dataset confirms the robustness of the method as well as its good performance and high learning speed. The author carried out the experiments and wrote the article. The instructors Amaury Lendasse and Eric Séverin provided very useful advices and helped correcting the original paper.

Publication VI: "Bankruptcy Prediction using Extreme Learning Machine and Financial Expertise"

In this journal paper, Leave-One-Out-Incremental Extreme Learning Machine (LOO-IELM) is explored for the task of bankruptcy prediction. LOO-IELM operates in an incremental way to avoid inefficient and unnecessary calculations and stops automatically with the neurons of which the number is unknown. Moreover, Combo method and further Ensemble model are investigated based on different LOO-IELM models and the specific ratios on which special strategies are used according to the financial expertise. The entire process has shown its good performance with a very fast speed, and also helps to interpret the model and the special ratios. The original idea was from Amaury Lendasse and the author carried out the experiments and writing of the paper.

Publication VII: "Ensemble Delta test- Extreme Learning Machine (DT-ELM) For Regression"

This journal paper proposes a method called Delta test-ELM (DT-ELM), which operates in an incremental way to avoid inefficient and unnecessary calculations and stops automatically with the number of neurons which is unknown beforehand. It uses Bayesian Information Criterion (BIC) to restrict the search as well as to consider the size of the network and Delta test (DT) to further prevent overfiting. Moreover, ensemble modeling is used on different DT-ELM models and it shows good test results in Experiments Section. The experiments and writing were carried out by the author and Amaury Lendasse helped a lot improving the quality of the paper.

Publication VIII: "Bankruptcy Prediction with Missing Data"

This conference paper proposes a classification method Ensemble Nearest Neighbors (ENN) to solve bankruptcy prediction problem. ENN uses different nearest neighbors as ensemble classifiers, then make a linear combination of them. Instead of choosing k in original K-nearest neighbor algorithm, ENN provides weights to each classifier which makes the method more robust. Moreover, using a adapted distance metric, ENN can be used directly for incomplete data. In a word, ENN is a robust and a comparatively simple model while providing good performance with missing data. The author carried out the experiments and wrote the article. Amaury Lendasse provided very useful advices and helped correcting the original paper.

List of Abbreviations

BIC	Bayesian information criterion
BPNN	Backpropagational neural network
DT-ELM	Ensemble Delta test-ELM
ECM	Expectation conditional maximization algorithm
ELM	Extreme learning machine
EKNN	Ensemble K-nearest neighbor
FFNN	Feedforward neural network
$k-$NN	K-nearest neighbor
LARS	Least angle regression
LDA	Linear discriminant analysis
LL	Lazy learning
LOO-CV	Leave-One-Out Cross-validation
LOO-IELM	Incremental ELM with Leave-One-Out
LPM	Linear probability model
MAR	Missing at random
MCAR	Missing completely at random
MD	Missing data
MDA	Multiple discriminant analysis
MNAR	Missing not at random
MRSR	Multiresponse sparse regression
NNE	Nonparametric noise estimation
NNLS	Non-Negative constrained Least-Squares
OLS	Ordinary least square
OP-KNN	Optimal pruned K-nearest neighbors
PCA	Principal component analysis
PDE	Pairwise distance estimation
PRESS	PREdiction sum of squares

SLFN	Single-Layer feedforward neural network
SOM	Self-organizing map
SVM	Support vector machine
TROP-ELM	Double-regularized ELM

List of Tables

List of Figures

1. Introduction

1.1 Scope of the dissertation

Machine Learning has come a long way over the last decade. There are more and more application fields relying on Machine Learning for data analysis. In this dissertation, we explore Machine Learning methods for financial fields, especially aiming at corporate bankruptcy prediction.

Corporate bankruptcy has always been widely studied due to its severe consequences. The accurate prediction of bankruptcy has been an important topic in the accounting and finance field for a long time. Therefore, several important issues about bankruptcy prediction are studied and discussed in this dissertation.

Firstly, there has been considerable interest in using financial ratios for predicting financial distress in companies since the seminal work of Beaver [10] using univariate analysis and Altman approach with multiple discriminant analysis [5]. The big amount of ratios makes the bankruptcy prediction a very different high-dimensional classification problem. So data preprocessing for the selection of ratios is the important area in which prediction performance has to be improved. Thus, this area is covered in this dissertation.

Secondly, several new Machine Learning methods are explored aiming at bankruptcy prediction in this dissertation. They are Optimally Pruned K-nearest neighbors (OP-KNN) (Publication I), Delta test-ELM (DT-ELM) (Publication VII) and Leave-One-Out-Incremental Extreme Learning Machine (LOO-IELM) (Publication VI). Moreover, soft classification techniques (classifier ensembles and the usage of financial expertise) are used in this dissertation. For example, Ensemble K-nearest neighbors (EKNN) (Publication V), Ensembles of Local Linear Models (Publication IV), and

Combo and Ensemble model in Publication VI. This reveals not only the great potential of soft classification techniques, which appear to be the direction for future research as core techniques that are used in the development of prediction models. In addition, missing data issue is also considered and solved in this dissertation, like the methods shown in Publication II and Publication VIII.

Thirdly, in addition to selecting ratios and models, the other foremost issue in experiments is the selection of datasets. Different studies have used different datasets, some of which are publicly downloadable, some collected data from very limited local companies. In this dissertation, we use a dataset collected from French retails SME companies. It is always important to carefully select the datasets for experiments.

Finally, to conduct a reliable experiment, the n-fold cross-validation strategy should be considered. In addition, a cross test in 6.5.2 is also used in this dissertation. These methods eliminate variability in samples, which may influence the performance of prediction models and minimize the effect of bias. And then be able to provide a better understanding of the performance of the classifiers and provide more reliable conclusions.

1.2 Scientific contributions of the dissertation

The present dissertation contains the following scientific contributions:

- A new Machine Learning model is proposed, Optimally Pruned $K-$nearest neighbors (OP-KNN) (Publication I). It builds a one hidden-layer feed-forward neural network using K-nearest neighbor (k-NN) as kernels to perform regression. Multiresponse Sparse Regression (MRSR) [88] is used in order to rank each kth nearest neighbor and finally Leave-One-Out (LOO) estimation is used to select the optimal number of neighbors and to estimate the generalization performances. OP-KNN gains a good generalization performance for both regression and classification problems. Moreover, thanks to its extremely fast learning speed, OP-KNN can be used recurrently for variable selection.

- Aiming at bankruptcy prediction, several methods have been developed based on different specific requirements. (1) Earlier work on Ensemble K-nearest neighbors (EKNN) (Publication V) uses K-nearest neighbors (k-NN) to build several classifiers, each of the classifiers uses different

nearest neighbor on different subset of input variables and tries to minimize the mean square error. Finally a linear combination of these classifiers is calculated to get even better performance. The learning speed of EKNN is extremely small, but the classification accuracy still can be improving. (2) Thus, Ensembles of Local Linear Models (Publication IV) has been developed. It builds ensembles of locally linear models using a forward variable selection technique and provides information about the importance of the variables. Therefore, the main advantage of the method is that the variable selection embedded into the method provides good interpretability of the results. (3) Another branch of methods referred to Extreme Learning Machine (ELM) is brought to bankruptcy prediction. ELM originally proposed by Huang [40] is based on random projections and Artificial Neural Networks. We have developed two algorithms using ELM: Delta test-ELM (DT-ELM) (Publication VII) and Leave-One-Out-Incremental Extreme Learning Machine (LOO-IELM) (Publication VI). DT-ELM uses Bayesian Information Criterion (BIC) to restrict the search as well as to consider the size of the network and Delta test (DT) to further prevent overfiting. LOO-IELM operates in an incremental way to avoid inefficient and unnecessary calculations and stops automatically with the neurons of which the number is unknown. Especially LOO-IELM was used with the combination of financial expertise for better prediction. This reveals the great potential of the combination with Machine Learning methods and financial preknowledge.

- Missing data (Missing value) is a common problem when collecting the companies' financial data. In this dissertation, two methods are developed to contribute on this issue. One is to use Ensemble Nearest Neighbors (ENN) to solve bankruptcy prediction problem and meanwhile using an adapted distance metric which can be used directly for incomplete data (Publication VIII). Another one is to estimate the expected pairwise distances between samples directly on incomplete data and to use TROP-ELM [70] to regularize the matrix computations (Publication II). These tools for missing data problem make our methods more practical for real world data.

1.3 Structure of the dissertation

This dissertation is organized as follows: Chapter 1 gives an overall introduction of this dissertation. The main body of this dissertation is expounded in three parts. The first part (containing Chapter 2) gives a brief overview of the field of bankruptcy prediction, including the development of modeling solutions. The goal is to better present the motivation and the contribution of this dissertation.

The second part of this dissertation (containing Chapter 3 to 5) proposes firstly a review of Machine Learning field in Chapter 3, then presents several new methods developed by us in the following two chapters. The third part of this dissertation (containing Chapter 6) utilizes a real French companies' data, aiming to test all the methods mentioned above and make the analysis based on all resources including financial expertise. This dataset is well collected from year 2002 (a smoothly developing year for France) and 2003 (a difficult year with financial distress in France), which makes possible to test the robustness of the model whether be capable to cover extreme financial situations. On the other hand, this dataset is built on SME (Small and medium enterprises) and on a specific sector (retail), aiming to focus on the accuracy and pertinency of the model. Several articles like [45, 60] have also used this data so that our results are able to compare with them.

The dissertation is finally concluded in chapter 7.

2. Basics of Bankruptcy Prediction

Bankruptcy prediction has long been an important and widely studied topic, which is of a great concern to investors or creditors, borrowing firms and governments. For example, banks need to predict the possibility of default of a potential counterpart before they extend a loan. This can lead to wiser lending decisions, and therefore result in significant savings. Bankruptcy can happen to any organization because the business environment is increasingly undergoing uncertainty and competition these days. Therefore, assessment of bankruptcy offers invaluable information by which governments, investors, shareholders or the management can make their financial decisions in order to prevent possible losses. Especially due to the recent changes in the world economy and as more firms, large and small, seem to fail now more than ever. The prediction of the bankruptcy, is then of increasing importance.

This section briefly introduces the history and development of bankruptcy prediction study. In latter sections of this dissertation, new Machine Learning methods are applied for solving this problem.

2.1 Background and Financial ratios analysis

The bankruptcy prediction analysis traces its history back to two centuries ago. At first, potential corporate distresses were assessed based on some qualitative information, which were very subjective [11, 57, 23]. Surprisingly, these recommendations could still be considered in many existing investment decisions. Later, early in the 20th century, the analysis of companies' financial conditions has moved forward to the analysis of financial statement data, more particularly, to the univariate ratio analysis.

The use of financial ratios to make qualitative statements about the

going concern of the firm has a long tradition. However, the generality of constructed ratios are controversial. There is unfortunately no textbook of corporate finance emphasizing the fact that benchmark values are not directly comparable over different industries. Financial ratios must thus be evaluated in conjunction with additional information related to the nature of the firm and the market. Moreover, measuring financial ratios is not equivalent with observing "real characteristics", but should rather be considered as "surrogate measures" of the relevant aspects.

One thing that appears to have influence on the predictive abilities of models is the number of factors considered in the model. The number of factors considered in relevant articles ranges from one to 57 [26]. Table 2.1 lists the 42 factors that are considered popular in these studies. The factor most common to multiple studies is the ratio of Net Income to Total Assets (Return on Assets). There has also been some fluctuation in the number of factors used over the last 40 years; however, the average has remained fairly constant around eight to ten factors.

Another thing has to be pointed out is that the unsuccessful business has been defined in different ways. The definition of "bankruptcy" itself is a complex story, which is not covered in this dissertation. The most frequently used terms found in literature are: failure, bankruptcy, insolvency, and default. In this dissertation, we collect the data when the companies go into voluntary liquidation. After the data has collected, bankruptcy prediction is treated as a binary classification problem in most of the studies. The target (output) variable of the models is commonly a dichotomous variable where "firm filed for bankruptcy" is set to 1 and "firm remains solvent" is set to -1.

2.2 Earlier methods

As mentioned previously, the early studies concerning ratio analysis for bankruptcy prediction are known as the univariate studies. These studies consisted mostly of analyzing individual ratios, and sometimes, of comparing ratios of failed companies to those of successful firms. However, in Mid-60s, Beaver [10] studied the predictive ability of accounting data as predictors. His work was intended to be a benchmark for future investigations. Beaver found that a number of indicators could discriminate between matched samples of bankrupt and non-bankrupt firms for as long as five years prior to failure. In a real sense, his univariate analy-

sis of a number of bankruptcy predictors set the stage for the development of multivariate analysis models. Two years later, the first multivariate study was published by Altman (1968) [5]. With the well-known "Z-score", which is a multiple discriminant analysis (MDA) model, Altman demonstrated the advantage of considering the entire profile of characteristics common to the relevant firms, as well as the interactions of these properties. Specifically, the usefulness of a multivariate model taking combinations of ratios that can be analyzed together in order to consider the context or the whole set of information at a time compared to univariate analysis that study variables one at a time and tries to gather most information at once. Consequently to this discriminatory technique, Altman was able to classify data into two distinguished groups: bankrupt and non-bankrupt (health) firms.

2.3 Development of statistical techniques

Altman's works were then followed by subsequent studies that implemented comparable and complementary models. Meyer & Pifer (1970) employed a linear probability model (LPM) [55]. This is a special case of ordinary least square (OLS) regression. Deakin (1972) compared Beaver's and Altman's methods using the same sample [24]. Finally Deakin's findings were in favor of the discriminant analysis, which compared to the univariate analysis, is a better classifier for potential bankrupt firms. The same year, Edmister (1972) tested a number of methods of analyzing financial ratios to predict small business failures and Edmister recommended using at least three consecutive year's financial statement to predict [28]. Altman *et al.* (1977) constructed a new bankruptcy classification model called the "Zeta model" to update the "Z-score" [27]. Altman obtained good results with a classification accuracy: above 95% (training accuracy) one period prior to bankruptcy and above 70% prior to five annual reporting periods. Martin (1977) also presented a logistic regression model to predict probabilities of failure of banks [63] . Martin was then followed by Ohlson (1980) who developed a logistic regression model, logit model or logit analysis (LA), to predict bankruptcies [75]. Zmijewski (1984) denounced that estimating models on nonrandom samples can result in biased parameter and probability estimates if appropriate estimation techniques are not used [102]. West (1985) used the combination of factor analysis (FA) and logit estimation as a new approach to measure

the condition of individual institutions and to assign each of them a probability of being a problem bank [95]. Karels & Prakash (1987) underlined the fact that it would be better to use linear discriminant analysis (LDA) than quadratic discriminant analysis (QDA), which is too sensitive to the loss of the normality assumption [49].

Altman (1993) adapted his "Z-score" to private firms' application and moreover, Altman (1995) applied a further adaptation of the original "Z-score" to non-manufacturers and emerging markets' firms [7]. Few years later, Shumway (2001) developed a dynamic logit for forecasting bankruptcy [87]. Jones & Hensher (2004) developed a mixed logit model for financial distress prediction [37]. Canbas *et al.* (2005) combined four different statistical techniques (PCA, DA, LA, and PA) to develop the integrated early warning system (IEWS) that can be used in prediction of bank failures [15]. Results were in favor of the utilization of such a combination of four parametric approaches to the banking sector and more generally, they should be extended to other business sectors for failure prediction. Philosophov *et al.* (2007) also investigates a new type of predictive information. Bayesian-type forecasting rules are developed that jointly use the financial ratios and maturity schedule factors [78].

Recently, Altman, Fargher, & Kalotay (2011) estimated the likelihood of default inferred from equity prices, using accounting-based measures, firm characteristics and industry-level expectations of distress conditions [4]. In order to improve the analysis performance of logit model, Li, Lee, Zhou, & Sun (2011) presented a combined random subspace approach (RSB) with binary logit model (L) to generate a so called RSB-L model that takes into account different decision agents' opinions as a matter to enhance results [56]. J. Sun & Li (2011) tested the feasibility and effectiveness of dynamic modeling for financial distress prediction (FDP) based on the Fisher discriminant analysis model [91].

2.4 Recent techniques

Recently, the bankruptcy prediction models can be divided into two main streams. The first one is based on statistical methods which discussed in previous section.

The second one is employing artificial intelligence (AI) methods and data mining methods, and a number of studies have applied them to bankruptcy prediction problem from 1990's. AI methods include decision tree

(Frydman *et al.* [34], Marais *et al.* [62]), fuzzy set theory (Zimmermann *et al.* [101]), case-based reasoning (Bryant [14] and Park *et al.* [77]), genetic algorithm (Shin and Lee *et al.* [86] and Varetto [93]), support vector machine (Min & Lee *et al.* [71]), data envelopment analysis (Cielen & Vanhoof *et al.* [18]), rough sets theory (Dimitras *et al.* [25], McKee [65] and McKee [66]), and several kinds of neural networks such as BPNN (back propagation trained neural network) (Atiya, 2001, Bell, 1997, Lam, 2004, Leshno and Spector, 1996, Salchenberger *et al.*, 1992, Swicegood and Clark, 2001, Tam, 1991 and Wilson and Sharda, 1994), PNN (probabilistic neural networks) (Yang & Platt *et al.* [97]), SOM (self-organizing map) (Kaski *et al.* [50], Kiviluoto [51]), Cascor (cascade correlation neural network) (Lacher, Coats, Sharma,& Fantc *et al.* [52]).

It is rather complicated to compare these Machine Learning methods, aiming at bankruptcy prediction. However, next section commences the illumination of these techniques, aiming to compare and analyze the advantages and drawbacks of them.

Net Income/ Total Assets
Current Ratio
Working Capital/ Total Assets
Retained earnings/ Total Assets
Earnings before interest and taxes/ Total Assets
Sales/ Total Assets
Quick Ratio
Total Debt/ Total Assets
Current Assets/Total Assets
Net Income/ Net worth
Total Liabilities/ Total Assets
Cash/Total assets
Market value of equity/ Book value of total debt
Cash flow from operations/ Total assets
Cash flow from operations/ Total liabilities
Current liabilities/ Total assets
Cash flow from operations/ Total debt
Quick assets/ Total assets
Current assets/ Sales
Earnings before interest and taxes/ Interest
Inventory/ Sales
Operating income/ Total assets
Cash flow from operations/ Sales
Net income/ Sales
Long-term debt/ Total assets
Net worth/ Total assets
Total debt/ Net worth
Total liabilities/ Net worth
Cash/ Current liabilities
Cash flow from operations/ Current liabilities
Working capital/ Sales
Capital/ Assets
Net sales/ Total assets
Net worth/ Total liabilities
No-credit interval
Total assets (log)
Cash flow (using net income)/ Debt
Cash flow from operations
Operating expenses/ Operating income
Quick assets/ Sales
Sales/ Inventory
Working capital/ Net worth

Table 2.1. Popular factors in prediction models

3. A brief review on Machine Learning

Machine Learning (ML), as a broad concept which covers techniques from so many different fields, is very difficult to define precisely. Coarsely speaking, Machine Learning is a way to teach the computers to "learn" from the "data" automatically. Therefore, in this section, we focus on summarizing Machine Learning methods that are most relevant to the Financial cases, especially Bankruptcy prediction issues, and also how we prepare the "data" that can be learned.

Besides, we propose a procedure in general how to estimate the Machine Learning model, so that to know if the Machine "learns" well from the "data".

3.1 Machine Learning Basics

3.1.1 What is Machine Learning

Machine Learning (ML) is considered as a subfield of Artificial Intelligence and it is concerned with the development of techniques and methods which enable the computer to learn. Zoologists and psychologists study learning in animals and human beings, but here, Machine Learning aims to mimic intelligent abilities of humans and animals by machines.

Machine Learning borrows techniques from so many different fields. Many problems in Machine Learning can be phrased in different but equivalent ways. In the following, we list several classic and significant real-world applications as one measure of progress in Machine Learning.

- *Web page ranking*. Most readers will be familiar with the concept of web page ranking. That is, the process of submitting a query to a search engine, which then finds webpages relevant to the query and which re-

turns them in their order of relevance [98]. To achieve this goal, a search engine needs to 'know' which pages are relevant and which pages match the query. Such knowledge can be gained from several sources: the link structure of webpages, their content, the frequency with which users will follow the suggested links in a query, or from examples of queries in combination with manually ranked webpages. Machine Learning is used to *automate* the process of designing a good search engine.

- *Face recognition*. That is, given the photo (or video recording) of a person, recognize who this person is. In other words, the system needs to *classify* the faces into one of many categories (Alice, Bob, Charlie, ...) or decide that it is an unknown face [9]. A similar, yet conceptually quite different problem is that of verification. Here the goal is to verify whether the person in question is who he claims to be. Note that differently to before, this is now a yes/no question. To deal with different lighting conditions, facial expressions, whether a person is wearing glasses, hairstyle, etc., it is desirable to have a system which *learns* which features are relevant for identifying a person.

- *Automatic translation*. Automatic translation of documents becomes more and more important in the modern business. At one extreme, we could aim at fully understanding a text before translating it using a curated set of rules crafted by a computational linguist well versed in the two languages we would like to translate. This is a rather arduous task, in particular given that text is not always grammatically correct, nor is the document understanding part itself a trivial one. Instead, we could simply use examples of translated documents, such as the proceedings of the Canadian parliament or other multilingual entities (United Nations, European Union, Switzerland) to learn how to translate between the two languages. In other words, we could use examples of translations to *learn* how to translate. This Machine Learning approach proved quite successful.

- *Robot control*. Machine learning methods have been successfully used in a number of robot systems. For example, several researchers have demonstrated the use of Machine Learning to acquire control strategies for stable helicopter flight and helicopter aerobatics. The recent Darpa competition involving a robot driving autonomously for over 100 miles in

the desert was won by a robot that used Machine Learning to refine its ability to detect distant objects (training itself from self-collected data consisting of terrain seen initially in the distance, and seen later up close).

Overall, Machine Learning Techniques play a key role in the world of computer science, within an important and growing niche. So the questions become to how machines can learn or how the learning process can be automated. Before further discussing on that, let us consider different types of learning problems.

In this dissertation, we consider that a problem is described by a data set, which takes the form of a matrix x, called inputs (or input data). The typical formulation uses the rows of x as samples (examples of the observed phenomenon, being different firms in the Bankruptcy Prediction case), and columns as variables (or features, or indicators in the Financial case). The data set is usually acquired from a specific source, for example to predict the status of the firms in this dissertation, company data are collected from balance sheet, annual reports, etc. Then we could define the learning problem as following.

3.1.2 Some types of Machine Learning problems

Learning is, of course, a very wide domain. Consequently, the field of Machine Learning has branched into several subfields dealing with different types of learning tasks. And it is useful to characterize learning problems according to the type of data.

Complete vs. Missing data

On the research point of view, all the data set can be treated as complete and reliable resource for modeling, however, in practice data can never be collected well organized. Missing data (MD) is unfortunately a part of almost all practical research, and researchers have to decide how to deal with it from time to time. When confronting the Missing Data, the common question you may ask is why and how they are distributed. Well, the nature of Missing Data can be categorized into three main types [58].

- *Missing completely at random (MCAR)* [36] When we say that data are missing completely at random, we mean that the probability that an observation (x_i) is missing is unrelated to the value of x_j or to the value

of any other variables. Thus, a nice feature of data which are MCAR is the analysis remains unbiased. We may lose power for our design, but the estimated parameters are not biased by the absence of data.

- *Missing at random (MAR)* Often data are not missing completely at random, but they may be classifiable as missing at random if the missingness does not depend on the value of x_i after controlling for another variable. The phraseology MAR is a bit awkward because we tend to think of randomness as not producing bias, and thus might well think that Missing at Random is not a problem. Unfortunately it is a problem, although we have ways of dealing with the issue so as to produce meaningful and relatively unbiased estimates [61].

- *Missing Not at Random (MNAR)* If data are not missing at random or completely at random then they are classed as Missing Not at Random (MNAR). When we have data that are MNAR we have a problem. The only way to obtain an unbiased estimate of parameters is to model missingness. In other words we would need to write a model that accounts for the missing data. Therefore, MNAR is not covered in this dissertation. This dissertation focuses on developing the method to solve the MD problem using Extreme Learning Machine, rather than to analyze the data of any specific field or MD for any specific reasons.

By far the most common approach is to simply omit those observations with missing data and to run the analysis on what remains. This is so called listwise deletion. Although listwise deletion often results in a substantial decrease in the sample size available for the analysis, it does have important advantages. In particular, under the assumption that data are missing completely at random, it leads to unbiased parameter estimates.

Another branch of approach is imputation, meaning to substitute the missing data point with a estimated value. A once common method of imputation was Hot-deck imputation where a missing value was imputed from a randomly selected similar record [33]. Besides, Mean substitution method uses the idea of substituting a mean for the missing data [19], etc.

There are also some advanced methods such as Maximum Likelihood and Multiple Imputation [82, 83]. There are a number of ways to obtain maximum likelihood estimators, and one of the most common is called the Expectation-Maximization algorithm (EM). This idea is further ex-

tended in Expectation conditional maximization (ECM) algorithm [67]. ECM replaces each M-step with a sequence of conditional maximization (CM) steps in which each parameter θ_i is maximized individually, conditionally on the other parameters remaining fixed. In this dissertation, a distance estimation method is presented based on ECM.

Binary classification, Multi-class Classification, Regression, Detection...
The range of learning problems is clearly large, as we saw when discussing applications. That said, researchers have identified an ever growing number of templates which can be used to address a large set of situations. It is those templates which make deployment of Machine Learning in practice easy and our discussion will largely focus on a choice set of such problems. We now give a by no means complete list of templates.

- *Binary Classification* is probably the most frequently studied problem in Machine Learning and it has led to a large number of important algorithmic and theoretic developments over the past century. In its simplest form it reduces to the question: given a sample x_i from input data $x \in \mathbb{R}^d$, estimate which value an associated binary random variable $y_i \in \pm 1$ will assume. For instance, given pictures of apples and oranges, we might want to state whether the object in question is an apple or an orange. Equally well, we might want to predict whether a home owner might default on his loan, given income data, his credit history, or whether a given e-mail is a spam or ham. The ability to solve this basic problem already allows us to address a large variety of practical settings. As to the Bankruptcy prediction problems, it is always treated as a binary classification one. Each sample of the data belongs to a group of predefined classes (Bankrupt or Non-bankrupt) and the objective is to try to separate one class from the other with the minimum amount of error.

- *Multi-class Classification* is the logical extension of binary classification. The main difference is that now $y \in 1, ..., n$ may assume a range of different values. For instance, we might want to classify a document according to the language it was written in (English, French, German, Spanish, Hindi, Japanese, Chinese,...). The main difference to before is that the cost of error may heavily depend on the type of error we make. For instance, in the problem of assessing the risk of cancer, it makes a

significant difference whether we mis-classify an early stage of cancer as healthy (in which case the patient is likely to die) or as an advanced stage of cancer (in which case the patient is likely to be inconvenienced from overly aggressive treatment). Similar situations happen on Bankruptcy Prediction. Investors (Or Bank) will more concentrate on the error of mis-classifying the Bankruptcy firms to Health ones.

- *Regression* is another prototypical application. Here the goal is to estimate a real-valued variable (output) $y \in \mathbb{R}$ given a pattern x. For instance, we might want to estimate the value of a stock the next day, the yield of a semiconductor factory given the current process, the iron content of ore given mass spectroscopy measurements, or the heart rate of an athlete, given accelerometer data. One of the key issues in which regression problems differ from each other is the choice of a loss. For instance, when estimating stock values our loss for a put option will be decidedly one-sided. On the other hand, a hobby athlete might only care that our estimate of the heart rate matches the actual on average.

- *Novelty Detection* is a rather ill-defined problem. It describes the issue of determining "unusual" observations given a set of past measurements. Clearly, the choice of what is to be considered unusual is very subjective. A commonly accepted notion is that unusual events occur rarely. Hence a possible goal is to design a system which assigns to each observation a rating as to how novel it is. Readers familiar with density estimation might contend that the latter would be a reasonable solution. However, we neither need a score which sums up to 1 on the entire domain, nor do we care particularly much about novelty scores for typical observations. The application of Novelty Detection is not covered in this dissertation.

Unsupervised Learning vs. Supervised Learning

Since learning involves an interaction between the learner and the environment, one can divide learning tasks according to the nature of that interaction. The first distinction to note is the difference between supervised and unsupervised learning.

As an illustrative example, consider the task of learning to detect spam email versus the task of anomaly detection. For the spam detection task, we consider a setting in which the learner receives training emails for

which the label spam or non spam is provided. Based on such training the learner should figure out a rule for labeling a newly arriving email message. In contrast, for the task of anomaly detection, all the learner gets as training is a large body of email messages and the learner's task is to detect "unusual" messages.

More abstractly, viewing learning as a process of 'using experience to gain expertise', supervised learning describes a scenario in which the 'experience', a training example, contains significant information that is missing in the 'test example' to which the learned expertise is to be applied (say, the Spam/no-Spam labels). In this setting, the acquired expertise is aimed to predict that missing information for the test data. In such cases, we can think of the environment as a teacher that 'supervises' the learner by providing the extra information (labels). In contrast with that, in unsupervised learning, there is no distinction between training and test data. The learner processes input data with the goal of coming up with some summary, or compressed version of that data. Clustering a data set into subsets of similar objects is a typical example of such a task.

There is also an intermediate learning setting in which, while the training examples contain more information than the test examples, the learner is required to predict even more information for the test examples. For example, one may try to learn a value function, that describes for each setting of a chess board the degree by which White's position is better than the Black's. Such value functions can be learned based on a data base that contains positions that occurred in actual chess games, labeled by who eventually won that game. Such learning frameworks are mainly investigated under the title of 'reinforcement learning'.

From a theoretical point of view, supervised and unsupervised learning differ only in the causal structure of the model. In supervised learning, the model defines the effect one set of observations, called inputs, has on another set of observations, called outputs. In other words, the inputs are assumed to be at the beginning and outputs at the end of the causal chain. The models can include mediating variables between the inputs and outputs.

In unsupervised learning, all the observations are assumed to be caused by latent variables, that is, the observations are assumed to be at the end of the causal chain. In practice, models for supervised learning often leave the probability for inputs undefined. This model is not needed as long as the inputs are available, but if some of the input values are missing,

it is not possible to infer anything about the outputs. If the inputs are also modeled, then missing inputs cause no problem since they can be considered latent variables as in unsupervised learning.

Besides, with unsupervised learning it is possible to learn larger and more complex models than with supervised learning. This is because in supervised learning one is trying to find the connection between two sets of observations. The difficulty of the learning task increases exponentially in the number of steps between the two sets and that is why supervised learning cannot, in practice, learn models with deep hierarchies. In unsupervised learning, the learning can proceed hierarchically from the observations into ever more abstract levels of representation. Each additional hierarchy needs to learn only one step and therefore the learning time increases (approximately) linearly in the number of levels in the model hierarchy.

In this dissertation, Bankruptcy problem is treated as a supervised binary classification problem, like in most of related articles.

3.2 Some Machine Learning Models

For classification and regression problem, there are different choices of Machine Learning Models each of which can be viewed as a black box that solve the same problem. However, each model come from a different algorithm approaches and will perform differently under different data set.

The following subsections provide a brief summary of some underlying algorithmic models which related with this dissertation and also hope it can give a sense of whether they are a good fit for your particular problem.

3.2.1 Linear Regression based Models

The basic assumption is that the output variable y (a numeric value) can be expressed as a linear combination (weighted sum) of a set of input variables $x_1, ..., x_d$ (which is also numeric value). $y = \omega_1 x_1 + \omega_1 x_1 + ... + \omega_d x_d + b$, b is a constant term

The whole objective of the training phase is to learn the weights ω_1, ω_2,...,ω_d and b by minimizing the error function lost. Gradient descent is the classical technique of solving this problem with the general idea of adjusting the parameters along the direction of the maximum gradient of

the loss function.

To avoid overfitting, regularization technique ($L1$ and $L2$) is used to penalize large values of the weights. $L1$ is by adding the absolute value of weights into the loss function while $L2$ is by adding the square of ω_1 into the loss function. $L1$ has the property that it penalizes redundant features or irrelevant features (with very small weight) and is a good tool to select highly influential features.

The strength of Linear model is that it has very high performance in both scoring and learning. The Stochastic gradient descent-based learning algorithm is highly scalable and can handle incremental learning. The weakness of linear model is linear assumption of input features, which is often false.

3.2.2 Decision Tree based Models

The fundamental learning approach is to recursively divide the training data into buckets of homogeneous members through the most discriminative dividing criteria. The measurement of "homogeneity" is based on the output label; when it is a numeric value, the measurement will be the variance of the bucket; when it is a category, the measurement will be the entropy or gini index of the bucket. During the learning, various dividing criteria based on the input will be tried (using in a greedy manner); when the input is a category (Mon, Tue, Wed ...), it will first be turned into binary (isMon, isTue, isWed ...) and then use the true/false as a decision boundary to evaluate the homogeneity; when the input is a numeric or ordinal value, the lessThan, greaterThan at each training data input value will be used as the decision boundary. The training process stops when there is no significant gain in homogeneity by further splitting the Tree. The members of the bucket represented at leaf node will vote for the prediction; majority wins when the output is a category and member average when the output is a numeric.

The good part of Tree is that it is very flexible in terms of the data type of input and output variables which can be categorical, binary and numeric value. The level of decision nodes also indicate the degree of influences of different input variables. The limitation is each decision boundary at each split point is a concrete binary decision. Also the decision criteria only consider one input attribute at a time but not a combination of multiple input variables. Another weakness of Tree is that once learned it cannot be updated incrementally. When new training data arrives, you have to

throw away the old tree and retrain every data from scratch.

However, Tree when mixed with Ensemble methods (e.g. Random Forest, Boosting Trees) addresses a lot of the limitations mentioned above. For example, Gradient Boosting Decision Tree consistently beat the performance of other Machine Learning Models in many problems and is one of the most popular method these days.

3.2.3 *K*-nearest neighbor

We are not learning a model at all. The idea is to find K similar data point from the training set and use them to interpolate the output value, which is either the majority value for categorical output, or average (or weighted average) for numeric output. K is a tunable parameter which needs to be cross-validated to pick the best value.

Nearest Neighbor requires the definition of a distance function which is used to find the nearest neighbor. For numeric input, the common practice is to normalize them by subtracting the mean and dividing by the standard deviation. Euclidean distance is commonly used when the inputs are independent, otherwise mahalanobis distance (which account for correlation between pairs of input features) should be used instead. For binary attributes, Jaccard distance can be used.

The strength of *K*-nearest neighbor [20] is its simplicity as no model needs to be trained. Incremental learning is automatic when more data arrives (and old data can be deleted as well). Data, however, needs to be organized in a distance-aware tree such that finding the nearest neighbor is O(logN) rather than O(N). On the other hand, the weakness of k-NN is it doesn't handle high dimensionality well. Also, the weighting of different factors needs to be hand tuned (by cross-validation on different weighting combination) and can be a very tedious process.

3.2.4 Neural Networks

Neural Networks (NNs) are typically organized in layers. Layers are made up of a number of interconnected "nodes" which contain an "activation function". Patterns are presented to the network via the "input layer", which communicates to one or more "hidden layers" where the actual processing is done via a system of weighted "connections". The hidden layers then link to an "output layer" where the answer is output. This multi-layer model enables Neural Network to learn non-linear rela-

tionship between input x and output y.

Most NNs contain some form of "learning rule" which modifies the weights of the connections according to the input patterns that it is presented with. For example, the most common classes of NNs called "Feedforward Neural Networks" (FFNNs) and "Backpropagational neural networks" (BPNNs). Here in this dissertation, we demonstrate the case Single-Layer Feedforward Neural Network (SLFN) for simplicity.

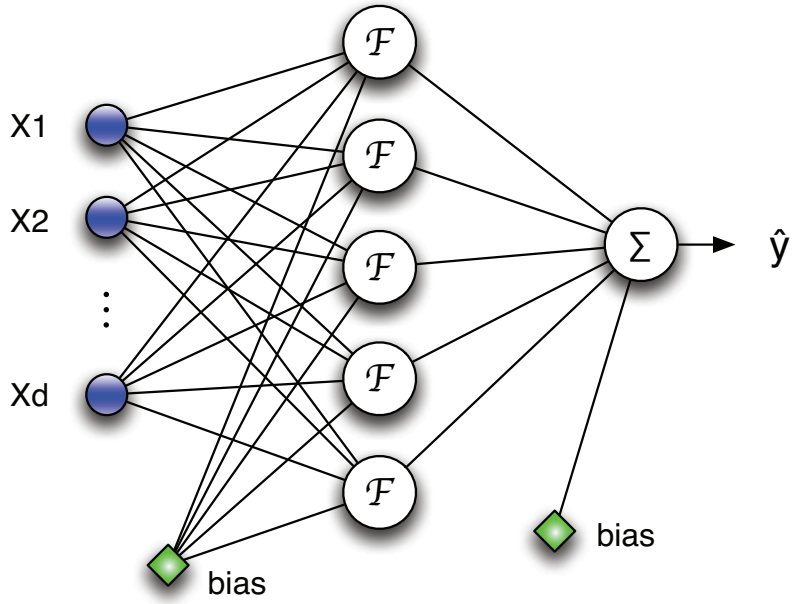


Figure 3.1. A Single Hidden Layer Feedforward Neural Network with m neurons

Figure 3.1 illustrates the case of a SLFN, with the input layer being the input sample $x_1, x_2, ..., x_d$ and the estimated output $\hat{y}$. The hidden layer contains m neurons, each of which performed by a function φ. Most of the cases, the function which calculates the output is considered as linear.

$$\hat{y} = \sum_{i=1}^{m} \alpha_i F(\sum_{j=1}^{d} \omega_i x_i + b_j^1) + b_i^2 \tag{3.1}$$

where b^1 and b^2 are the biases and the common choices of activation function F are the standard sigmoid function and hyperbolic tangent function that looks like this

$$Sigmoid(\chi) = \frac{1}{1 + e^{-\chi}} \tag{3.2}$$

$$tanh(\chi) = \frac{e^{2\chi - 1}}{e^{2\chi + 1}} \tag{3.3}$$

Taking the application of Bankruptcy Prediction for example, suppose 4 financial indicators are chosen for the predictor: Working Capital / Total Assets (WC/TA), Retained Earnings / Total Assets (RE/TA), Earnings Before Interest and Taxes / Total Assets (EBIT/TA) and Market Value of Equity / Total Liabilities (ME/TL). If Single Hidden Layer Feedforward Neural Network is used to solve this binary classification problem, it should perform as following Fig 3.2.

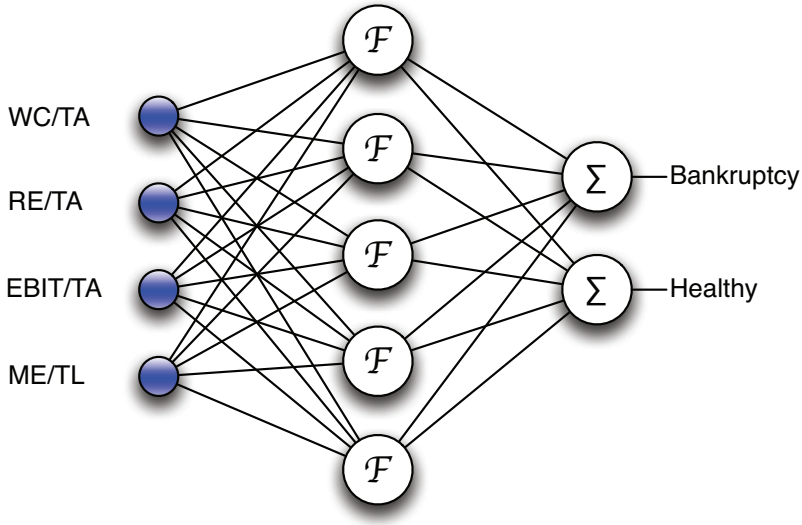


Figure 3.2. A Single Hidden Layer Feedforward Neural Network For Bankruptcy prediction

In a word, Neural networks offer a number of advantages, including requiring less formal statistical training, ability to implicitly detect complex nonlinear relationships between input and output variables, etc. On the other hand, it also has its "black box" nature, relatively longer computational time and proneness to overfitting.

3.2.5 Support Vector Machines

Support Vector Machines (SVMs) were first suggested by Vapnik [21] in the 1960s for classification and have recently become an area of intense research owing to developments in the techniques and theory coupled with extensions to regression and density estimation. In this dissertation, we focus on SVMs for binary class classification, the classes being P, N for $y_i = +1, -1$ respectively.

If the training data are linearly separable then there exists a pair (ω, b) such that

$$\omega^T x_i + b \geq 1, \text{ for all} x_i \in P \tag{3.4}$$

$$\omega^T x_i + b \leq -1, \text{ for all} x_i \in N \tag{3.5}$$

ω is termed the weight vector and b the bias. The learning problem is hence reformulated as: minimize $\|\omega\|^2 = \omega^T \omega$ subject to the constraints of linear separability. This is equivalent to maximizing the distance, normal to the hyperplane, between the convex hulls of the two classes; this distance is called the margin. The optimization is now a convex quadratic programming (QP) problem

$$\text{Minimize } \Phi(\omega) = \frac{1}{2}\|\omega\|^2 \tag{3.6}$$

$$\text{subject to } y_i(\omega^T x_i + b) \geq 1, i = 1, ..., l \tag{3.7}$$

This problem has a global optimum; thus the problem of many local optima in the case of training e.g. a neural network is avoided. This has the advantage that parameters in a QP solver affects only the training time, and not the quality of the solution.

So far we have restricted ourselves to the case where the two classes are noise-free. In the case of noisy data, forcing zero training error will lead to poor generalization. This is because the learned classifier is fitting the idiosyncrasies of the noise in the training data. To take account of the fact that some data points may be misclassified we introduce a vector of slack variables ${\xi_1, ..., \xi_l}^T$ that measure the amount of violation of the constraints (Equation 3.4). The problem can then be written

$$\text{Minimize } \Phi(\omega, b, \xi) = \frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{l} \xi_i^k \tag{3.8}$$

$$\text{subject to } y_i(\omega^T \phi(x_i) + b) \geq 1 - \xi_i, \xi_i \geq 0, i = 1, ..., l \tag{3.9}$$

where C and k are specified beforehand. C is a regularization parameter that controls the trade-off between maximizing the margin and minimizing the training error term. Thus, the SVM learns the optimal separating hyperplane in some feature space, subject to ignoring certain points which become training misclassification. The learned hyperplane is an expansion on a subset of the training data known as the support vectors. By use of an appropriate kernel function the SVM can learn a wide range of classifiers including a large set of RBF networks and neural networks.

The flexibility of the kernels does not lead to overfitting since the space of hyperplanes separating the data with large margin has much lower capacity than the space of all implementable hyperplanes.

3.3 Some Remarks of solving problems using ML

Before explaining the details of the procedure, an important concept, also a common problem, needs to be mentioned, *Overfitting* and *Underfitting*. Overfitting refers to the situation in which the algorithm generates a model which perfectly fits the data but loses the capability of generalizing to samples not presented during modeling. In other words, instead of learning, the overfitting model just memorizes the samples used to build the model. Underfitting, on the other hand, refers to the situation that the algorithm works poorly with the data set, the model memorizes not enough information of the samples. Figure 3.3 shows these phenomenons.

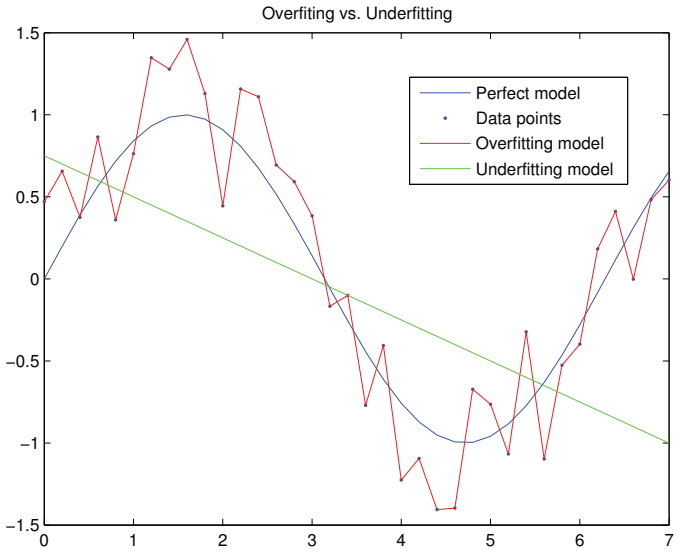


Figure 3.3. Overfitting and Underfitting example

There are many ways to prevent overfitting. In the following sections, several methods are introduced for this purpose.

3.3.1 Data Pre-processing

Pre-processing typically constitutes the initial (and possibly one of the most important) step in the analysis of data from any practical problems.

In most of the cases, pre-processing shouldn't be ignored or treated as a black box. Generally speaking, data pre-processing consists of data exploration, background correction, normalization and sometimes quality assessment, all of which are all interlinked steps. In this dissertation, we focus more on the following three steps:

Missing Data Missing data are a part of almost all practical research, and we all have to decide how to deal with it from time to time. In the first Chapter of this dissertation, some discussions have done on why data is missing and the nature of missing data. The last section of this chapter introduces some alternative ways of dealing with missing data. In some cases, missing data problems are solved as a separate pre-processing step and in some cases, missing data are solved combining with the modeling process. These techniques are discussed in more details later.

Outliers Outlier problem is one of the typical problems in an incomplete data based Machine Learning system. The definition of an outlier is an observation that lies an abnormal distance from other values in a random sample from a population. It is a pattern that was either mislabeled in the training data, or inherently ambiguous and hard to recognize, therefore, it usually brings extra trouble for a learning task, either in debasing the performance or leading the learning process to be more complicated. In a sense, the definition leaves it up to the analyst (or a consensus process) to decide what will be considered abnormal.

Outlier Detection aims to separate a core of regular observations from some polluting ones, called "outliers". One common way of performing outlier detection is to assume that the regular data come from a known distribution (e.g. data are Gaussian distributed). From this assumption, we generally try to define the "shape" of the data, and can define outlying observations as observations which stand far enough from the fit shape.

Outlier Substitution can be completed by many ways. The most common way is to remove the outliers but it causes various problems. For example, the removing operation may lose important information especially if the data set is quite limited. Or the reason should be taken into account, why the outlier becomes abnormal. Thus, in this dissertation, outliers are treated as missing data and substitute them using imputation methods.

Normalization The term normalization is used in many contexts, with distinct, but related, meanings. Basically, normalizing means transforming so as to render normal. When data are seen as vectors, normalizing

means transforming the vector so that it has unit norm. When data are though of as random variables, normalizing means transforming to normal distribution. When the data are hypothesized to be normal, normalizing means transforming to unit variance.

Let us consider input data $x : x \in N \times d$ as a matrix where each row $(x_1, ..., x_N)$ corresponds to an observation (a data element), and each column $x^1, ..., x^d$ corresponds to a variable (an attribute of the data). Let us furthermore assume that each data element has a response value $y : y \in N \times 1$ (target) associated to it. (In this dissertation we focus on supervised learning.)

Why *column normalization*? The simple answer is so that variables can be compared fairly in terms of information content with respect to the target variable. This issue is most important for algorithms and models that are based on some sort of distance, such as the Euclidean distance. As the Euclidean distance is computed as a sum of variable differences, its result greatly depends on the ranges of the variables. In practice, the way the normalization is handled depends on the hypotheses made. As to the data sets covered in this dissertation, variables are supposed normally distributed with distinct means and variances.

In such case, the idea is to center all variables so they have a zero mean and divide them by their standard deviation so that they all express unit variance. The transformed variables are then what are called 'z-scores' in the statistical literature. They are expressed in 'number of standard deviations' in the original data. The transformed values lie within the [-1, 1] interval.

Why *row normalization*? While column normalization can be applied to any data table, row normalization makes sense only when all variables are expressed in the same unit. This is not the case in this dissertation.

Why *target normalization*? Because building a model between the data observations and their targets is made easier when the set of values to predict is rather compact. So when the distribution of the target variable is skewed, that is there are many lower values and a few higher values, it is preferable to transform the variable to a normal one by computing its logarithm. Then the distribution becomes more even. Or for the binary classification problem (like in bankruptcy prediction in this dissertation), the target y is transformed either 1 or -1.

Therefore, normalization is a procedure followed to bring the data closer to the requirements of the algorithms, or at least to pre-process data so

as to ease the algorithm's job. Moreover, for some specific cases, data may need a *cleaning process* at the beginning (Removal of redundancies, errors, etc.). Or other operations like *discretization*: continuous values to a finite set of discrete values; *abstraction*: merge together categorical values; *aggregation*: summary or aggregation operations, such minimum value, maximum value etc.

Dimensionality Reduction Dimensionality Reduction is also an important issue in Machine Learning, especially when the number of observations (samples) is relatively small compared to the number of input variables. It has been the subject in application domains like pattern recognition, time series modeling and econometrics. There are some methods for this task, like Mutual Information measure, Principal Component Analysis (PCA), etc.

Feature (Variable) Selection is a particular case of Dimensionality Reduction. It selects the most important features to build the model, according to the target. There are also various ways to achieve this. In this dissertation, a new method called Nonparametric Noise Estimation (NNE) is investigated for feature selection and is introduced later.

3.3.2 Model Selection and its Criterion

After preparing the data set, it's time to take into account modeling process. We could generally name this process Model Selection. It contains several steps: defining the model types, setting the model parameters and evaluating the performance of the model.

Model Type is the first item being required for the fixed problem. Different learning problems (unsupervised learning, classification or regression problems, etc.) and different data sets (highly correlated, continuous features, high dimensionality or big amount of samples, etc.) lead to different types of models. Some models are designed and proved to be more appropriate for some specific problems. As well as the enterprise of specific application gives a priority to some candidate models. In the next chapter of the dissertation, few examples are given on how to choose the suitable model class, especially in the field of Bankruptcy Prediction.

Choosing *Model hyper-parameters*, in other words, choosing the *Model Structure* is related with the model design choice. For example, the number of hidden layers and the weights for each neuron in Neural Network, the depth and number of leaves in Decision Tree, etc. Moreover, this pro-

cess is intensively related with the selection *Error criterion*.

Again we suppose the data structured like $x : x \in N \times d$ with the target value $y : y \in N \times 1$. Model $\mathcal{M}$, which contains a certain number l of *hyper-parameters* $(\theta_1, ..., \theta_l)$, is chosen for these data. Thus, the remaining problem transformed to determine the optimal set of l parameters according to the dataset. The goal is to find the smallest possible *error* for the model $\mathcal{M}(x, \theta)$, regarding to the output y.

Error criterion in general quantifies how close the expected output $\hat{y}$ from the original output y, for the specific model $\mathcal{M}(x, \theta)$ with chosen hyper-parameters. However, error criterion differs in accordance with different learning problems. For example, the regression problem of a single output typically uses *Mean Square Error* as criteria, which is

$$Error_{reg}(\theta) = \varepsilon_{MSE} = \frac{N}{1}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 = \frac{N}{1}\sum_{i=1}^{N}(y_i - \mathcal{M}(x, \theta))^2 \quad (3.10)$$

Therefore, the model learning is a process to find the optimal set of θ

$$\theta^* = \arg\min_{\theta} Error_{reg}(\theta) \quad (3.11)$$

In this dissertation, *Normalized Mean Square Error (NMSE)* is also defined and used as the following formulation,

$$Error_{reg} = \varepsilon_{NMSE} = \frac{\varepsilon_{MSE}}{Var(y)} \quad (3.12)$$

As to the binary classification problem, error criteria basically concentrates on the four numbers: True positive (Tp), False positive (Fp), True negative (Tn) and False negative (Fn). In this dissertation, we use the accuracy of both classes to build the model,

$$Accuracy_c = \frac{Tp + Tn}{Tp + Tn + Fp + Fn} \quad (3.13)$$

Then the corresponding error is calculated as: $Error_c = 1 - Accuracy_c$. The reason for choosing this criteria is the balanced dataset we used. In other cases, more calculations are defined, like the common probabilities: Recall $(= \frac{Tp}{Tp+Fn})$ and Precision $(= \frac{Tp}{Tp+Fp})$ [76]. Most of the regression models or techniques can be used for classification, some with minor modifications. We introduce more details later on our specific application.

Model learning process equals to optimization problems which can be hard to solve. Right choice of the model class and an error function makes a difference. However, the model built on some certain data may not perform well on the new data. To solve this, model evaluation is needed.

3.3.3 Model Evaluation: Training, Validation and Testing

Many methods, such as recursive partitioning and neural networks, are extremely sensitive to the sample of data being mined. How do you know if you are creating a model that would be useful for predicting future outcomes?

A classic way is to *split* the original data into three parts (N samples): *training data* (N_{train} samples), *validation data* (N_{val} samples) and *testing data* (N_test samples), where $N = N_{train} + N_{val} + N_{test}$. So that the data are assigned for two tasks, learning the model (training part and validation part) and testing the model.

The learning scheme operates in two stages: building the basic structure of the model and optimizing parameter settings; some learning algorithms combine the two stages as an integration. Generally, the larger the training data, the better the model is; the larger the test data, the more accurate the error estimates. Thus, how to find the balanced point splitting the data, and how to make good use of the data become an important issue. In such cases, cross-validation is created and used.

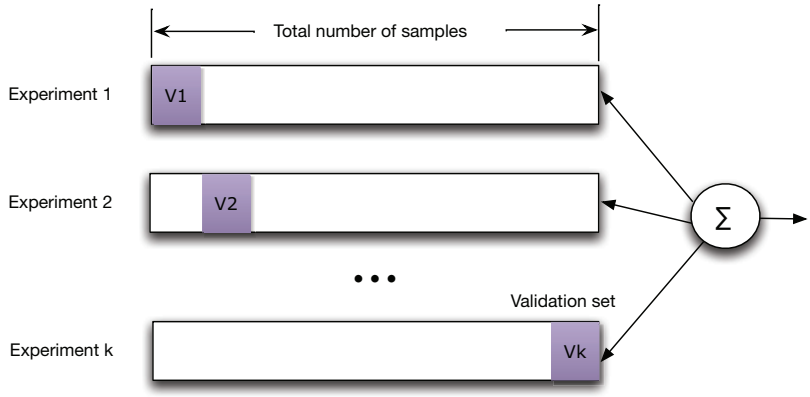


Figure 3.4. Example of k-folder cross-validation

Cross-validation, showed in Fig 3.4, is usually performed in a K-fold way, where the data set is divided into k subsets, and the learning process is repeated k times. Each time, one of the k subsets is used as the validation set and the other $k-1$ subsets are put together to form a training set.

Then the average error across all k trials is computed. The advantage of this method is that it matters less how the data gets divided. Every data point gets to be in a validation set exactly once, and gets to be in a training set $k-1$ times. The variance of the resulting estimate is reduced as k is increased. The disadvantage of this method is that the training algorithm has to be rerun from scratch k times, which means it takes k times as much computation to make an evaluation.

One particular case of Cross-Validation is when $k = N$, the number of folders equals exactly the number of samples. It is called Leave-One-Out Cross-Validation (LOO-CV), where each sample has a chance to validate the model and the learning process repeated N times.

With a large number of folds, the bias of the true error estimator will be small and the computational time will be very large; while with a small number of folds, the computation time is reduced but the bias of the estimator will be large. In practice, the choice of the number of folds depends on the size of the dataset and the application field. For very sparse datasets, we may have to use Leave-One-Out in order to train on as many examples as possible. In the case of no prior knowledge on the dataset, the common choice for CV is $k = 10$.

As to the final testing phase, it is important that the test data is not used in any way to build the model and the test data can't be used for parameter tuning neither. This can be also seen in the Fig 3.4. The model should not be further tuned after assessing the final model with the test set.

Fig 3.5 illustrates the entire procedure of the modeling process. Generally, it contains the following steps:

1. Preprocess the data. (Fixing Missing Data, Outliers problem, and Normalize the data)

2. Divide the data into training, validation and testing set.

3. Decide the model type and the corresponding hyper-parameters.

4. Train the model using the training set.

5. Evaluate the model using the validation set.

6. Repeat steps 4 and 5 for each of the k folders if Cross-Validation is used.

7. Select the best model structure and its optimal set of parameters

8. Assess the final model using the test set.

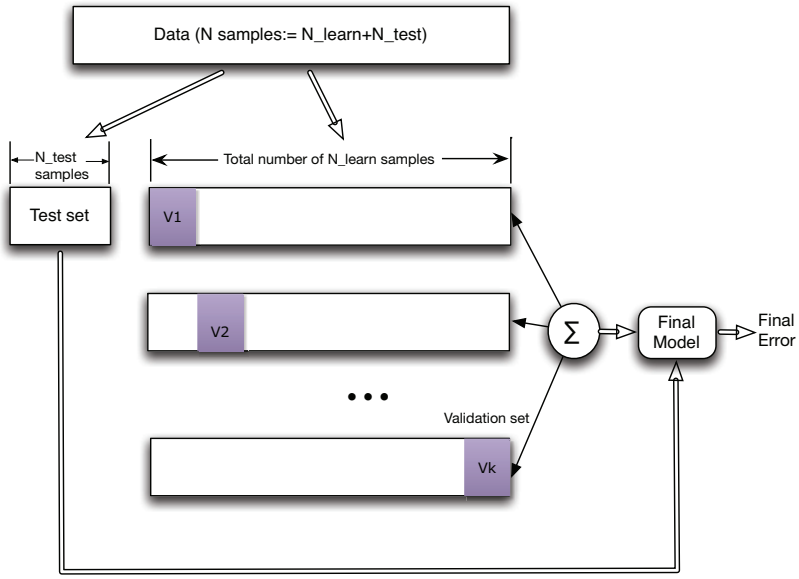


Figure 3.5. General data modeling process

In addition, a cross test method 6.5.2 is used in this dissertation. Cross test works in a way that when the data is prepared after step 1 and 2, all the processes are repeated k times. Thus, the final results is the average of these k trials. The goal is to get more general performance of the model and meantime, reduce the errors from biased data splitting for training and testing. Since the repetitions are randomly based, it is also called "Monte Carlo" cross test. All the experiments shown in this dissertation is done using Monte Carlo cross test.

4. Optimal Pruned *K*-Nearest Neighbors (OP-KNN)

In this section, we propose a new Machine Learning model Optimal pruned K-nearest neighbors (OP-KNN) (Publication I) which builds a single-hidden layer feedforward neural networks (SLFN) using k-NN as the kernel. The most significant characteristic of this method is that it tends to provide as good generalization performance as SVM, or even better for some cases and with an extremely high learning speed.

4.1 Motivation of OP-KNN

High dimensional data appears in more and more application fields. Taking Bankruptcy prediction for example, more financial ratios are taken into account as indicators instead of five for classic Z-score method in Altman's article [6]. The necessary size of the data set increases exponentially with the number of features. In theory, the more samples learned, the more accurate the model is. When taking into account the computational time, it simply leads to disaster eventually. Especially if the model containing a number of hyper-parameters, the optimization process will reach a dramatical growth on computational time.

For example, Support Vector Machine (SVM) is one of the most popular techniques now in Machine Learning, which was initially developed to classification tasks and lately has been extended to the domain of regression. Different from Multiple Layer Perception (MLP), the nonlinear classification and model regression are solved with convex optimization with a unique solution, which avoid the problem of local minima of MLP [22]. However, there are still some limitations of SVM that weakens its performance: the biggest one lies in the choice of the kernels, following is the speed and size, both in training and testing. Even from a practical point of view perhaps the most serious problem with SVM is the high algorithmic

complexity and extensive memory requirements of the required quadratic programming in large-scale tasks.

Thus, model complexity as well as data complexity raise challenges of Machine learning: fast and less-parameters models are needed. Therefore, another group of methods like K-nearest neighbor (k-NN), or Lazy Learning (LL) [89, 12] is taken into account. The key idea behind k-NN is that similar training samples have similar output values and it keeps avoiding the local minima problem as SVM, but performs more simple and fast.

4.2 Algorithm Structure of OP-KNN

The three main steps of the OP-KNN are summarized in Figure 4.1.

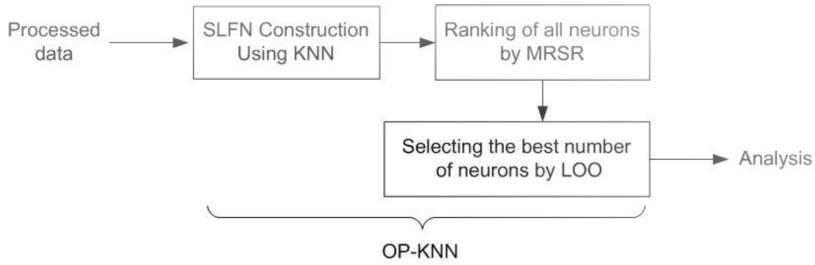


Figure 4.1. The three steps of the OP-KNN algorithm

4.2.1 Single-hidden Layer Feedforward Neural Networks (SLFN)

The first step of the OP-KNN algorithm is building a single-layer feedforward neural network. This is similar as the core of OP-ELM, which has been proposed by Yoan Miche *et al.* in [70]. The difference is that OP-KNN is deterministic, rather than randomly chooses hidden nodes like in OP-ELM.

In the context of a single hidden layer perceptron network, let us denote the weight vectors between the hidden layer and the output by $\mathbf{b}$. Activation functions used with the OP-KNN differ from the original SLFN choice since the original sigmoid activation functions of the neurons are replaced by the K-nearest neighbor (k-NN), hence the name OP-KNN. For the output layer, the activation function remains as a linear function, meaning the relationship between hidden layer and output layer is linear.

A theorem proposed in [40] states that the activation functions, output

weights $\mathbf{b}$ can be computed from the hidden layer output matrix $\mathbf{H}$: the columns $\mathbf{h_i}$ of $\mathbf{H}$ are the corresponding output of the *K*-nearest neighbor. Finally, the output weights $\mathbf{b}$ are computed by $\mathbf{b} = \mathbf{H}^{\dagger}\mathbf{y}$, where $\mathbf{H}^{\dagger}$ stands for the Moore-Penrose inverse [80] and $\mathbf{y} = (y_1, \ldots, y_M)^T$ is the output.

The only remaining parameter in this process is the initial number of neurons N of the hidden layer.

4.2.2 *K*-nearest neighbor (k-NN)

The *K*-nearest neighbor (k-NN) model is a very simple, but powerful tool. It has been used in many different applications and particularly in classification tasks. The key idea behind the k-NN is that similar training samples have similar output values. In OP-KNN, the approximation of the output is the weighted sum of the outputs of the *K*-nearest neighbor. The model introduced in the previous section becomes:

$$\hat{y}_i = \sum_{j=1}^{k} b_j y_{P(i,j)} \tag{4.1}$$

where $\hat{y}_i$ represents the output estimation, $P(i,j)$ is the index number of the j^{th} nearest neighbor of sample $\mathbf{x}_i$ and b is the results of the Moore-Penrose inverse introduced in the previous Section.

In this sense, for each different neuron, we use different nearest neighbors, in another word, the only remaining parameter we have to choose N is the neighborhood size we want to use K. Besides this, k-NN is a method with no parameters, as well as OP-KNN.

4.2.3 Multiresponse Sparse Regression (MRSR)

For the removal of the useless neurons of the hidden layer, the Multiresponse Sparse Regression proposed by Timo Similä and Jarkko Tikka in [88] is used. It is an extension of the Least Angle Regression (LARS) algorithm [29] and hence is actually a variable ranking technique, rather than a selection one. The main idea of this algorithm is the following: denote by $\mathbf{T} = [\mathbf{t}_1 \ldots \mathbf{t}_p]$ the $n \times p$ matrix of targets, and by $\mathbf{X} = [\mathbf{x}_1 \ldots \mathbf{x}_m]$ the $n \times m$ regressors matrix. MRSR adds each regressor one by one to the model $\mathbf{Y}^k = \mathbf{X}\mathbf{W}^k$, where $\mathbf{Y}^k = [\mathbf{y}_1^k \ldots \mathbf{y}_p^k]$ is the target approximation by the model. The $\mathbf{W}^k$ weight matrix has k nonzero rows at kth step of the MRSR. With each new step a new nonzero row, and a new regressor to the total model, is added.

An important detail shared by the MRSR and the LARS is that the ranking obtained is exact in the case where the problem is linear. In fact, this is the case, since the neural network built in the previous step is linear between the hidden layer and the output. Therefore, the MRSR provides the exact ranking of the neurons for our problem.

Details on the definition of a cumulative correlation between the considered regressor and the current model's residuals and on the determination of the next regressor to be added to the model can be found in the original dissertation about the MRSR [88].

MRSR is hence used to rank the kernels of the model: the target is the actual output y_i while the "variables" considered by MRSR are the outputs of the *K*-nearest neighbor.

4.2.4 Leave-One-Out (LOO)

Since the MRSR only provides a ranking of the kernels, the decision over the actual best number of neurons for the model is taken using a Leave-One-Out method. One problem with the LOO error is that it can get very time consuming if the dataset tends to have a high number of samples. Fortunately, the PRESS (or PREdiction Sum of Squares) statistics provide a direct and exact formula for the calculation of the LOO error for linear models. See [72] for details on this formula and implementations:

$$\epsilon^{\text{PRESS}} = \frac{y_i - \mathbf{h}_i\mathbf{b}}{1 - \mathbf{h}_i\mathbf{P}\mathbf{h}_i^T}, \tag{4.2}$$

where $\mathbf{P}$ is defined as $\mathbf{P} = (\mathbf{H}^T\mathbf{H})^{-1}$ and $\mathbf{H}$ the hidden layer output matrix defined in subsection 5.1.

The final decision over the appropriate number of neurons for the model can then be taken by evaluating the LOO error versus the number of neurons used (properly ranked by MRSR already).

4.3 Variable Selection using OP-KNN

Whether using k-NN, OP-KNN, SVM, LS-SVM or some other regression methods, a metric is needed to do Variable Selection. In fact, there are many ways to deal with the Variable Selection problem, a common one is using the generalization error estimation. In this methodology, the set of features that minimizes the generalization error are selected using Leave-One-Out, Bootstrap or other resampling technique [29, 94]. But these

approaches are very time consuming and may lead to an unacceptable computational time. However, there are other approaches. In this dissertation, Variable Selection is performed using OP-KNN as the metric since OP-KNN is extremely fast.

Wrapper method As we know, Variable Selection can be roughly divided into two board classes: filter method and wrapper method. As the name implies, our strategy belongs to the wrapper methods which means the variables are selected according to the criterion directly from the training algorithm.

In other word, our strategy is to selected the input subset that can give the best OP-KNN result. Once the input subset is fixed, OP-KNN is repeated to build the model. Furthermore, for the training set and test set, we do the selection procedure on the training set, and then use OP-KNN on the selected variables of the test set. In this dissertation, the input subset is selected by means of Forward Selection algorithm.

Forward Selection This algorithm starts from the empty set S which represents the selected set of the input variables. Then the best available variable in added to the set S one by one until running of all the variables.

To make more clear about Forward selection, suppose we have a set of inputs $X_i, i = 1, 2, ..., M$ and the output Y, then the algorithm is as follows:

1. Set F to be the initial set of the original M input variables, and S to be the empty set like mentioned before.

2. Find:
$$X_S = \arg\min_{x_i} \{Opknn(S \cup X_i)\} \quad x_i \in F \tag{4.3}$$
where X_S represents the selected variable, save the OP-KNN results and move X_S from F to S

3. Continue the same procedure, till the size of S is M.

4. Compare the OP-KNN values for all the sizes of the sets S, the final selection result is the set S which the corresponding OP-KNN give the smallest value.

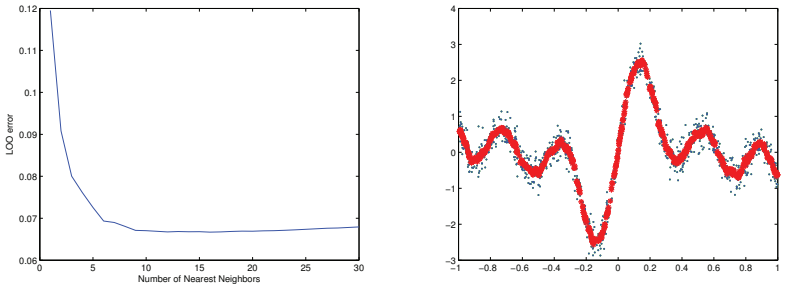
4.4 Experiments

This section demonstrates the speed and accuracy of the OP-KNN method, as well as the strategy we introduced before, using several different regression data sets. For the comparison, we provides also the performances using a well-known Support Vector Machine (SVM) implementations which is widely identified as a standard methods recently.

Following subsection shows a toy example to illustrate the performance of OP-KNN on a simple case that can be plotted.

4.4.1 Sine example

In this toy example, a set of 1000 training points are generated (and represented in Fig. 4.2 (b)), the output is a sum of two sines. This single dimension example is used to test the method without the need for variable selection beforehand.



(a) (b)

Figure 4.2. Sine Toy example using OPKNN

The Fig. 4.2 (a) shows the LOO error for different number of nearest neighbors and the model built with OP-KNN using the original dataset. This model approximates the dataset accurately, using 18 nearest neighbors; and it reaches a LOO error close to the noise introduced in the dataset which is 0.0625. The computational time for the whole OP-KNN is one second (using Matlab© implementation).

Thus, in order to have a very fast and still accurate algorithm, each of the three presented steps have a special importance in the whole OP-KNN methodology. The K-nearest neighbor ranking by the MRSR is one of the fastest ranking methods providing the exact best ranking, since the model is linear (for the output layer), when creating the neural network using k-

NN. Without MRSR, the number of nearest neighbor that minimizes the Leave-One-Out error is not optimal and the Leave-One-Out error curve has several local minima instead of a single global minimum. The linearity also enables the model structure selection step using the Leave-One-Out, which is usually very time-consuming. Thanks to the PRESS statistics formula for the LOO error calculation, the structure selection can be done in a small computational time.

4.4.2 UCI datasets

Ten data sets from UCI machine learning repository [1] are used. They are chosen for their heterogeneity in terms of problem, number of variables, and sizes. Eight data sets are for regression problem and two are for classification problem. The specification of the 10 selected data sets can be found in 4.1.

For the comparison of the OP-KNN and SVM, each data set is divided into two sets, train and test sets. The train set includes two thirds of the data, selected randomly without replacement, and the test set one third.

Table 4.1 shows some key information about the data sets and number of variables selected using OP-KNN, while Table 4.2 illustrates the Test error and Computational time for both methods.

Table 4.1. Specification of the selected UCI data sets and the their variable Selection results. For classification problem, both two data sets contain two classes

	Data			Variables
Regression	Train	Test	#Variable	Selected by OP-KNN
Abalone	2784	1393	8	6
Ailerons	4752	2377	5	2
Elevators	6344	3173	6	3
Auto Price	106	53	15	13
Servo	111	56	4	3
Breast Cancer	129	65	32	8
Bank	2999	1500	8	6
Stocks	633	317	9	8
Classification				
Wisconsin Cancer	379	190	30	9
Indians Diabetes	512	256	8	5

Table 4.2. Test error and Computational time comparison

	Test Error		Computational Time (second)	
Regression	SVM	OP-KNN	SVM	OP-KNN
Abalone	4.3	4.7	4.76E+04	2.15E+02
Ailerons	2.63E-08	3.22E-08	8.70E+04	2.79E+02
Elevators	2.89E-06	2.46E-06	7.72E+05	7.56E+02
Auto Price	3.78E+06	3.25E+06	4.92E+02	1.66
Servo	4.24E-01	1.503	8.63E+02	0.17
Breast Cancer	8.93E+02	7.15E+02	6.45E+02	8.88
Bank	2.21E-03	1.27E-03	6.54E+05	2.34E+02
Stocks	2.25E-01	7.96E-01	2.19E+03	1.16E+01
Classification				
Wisconsin Cancer	9.41E-01	9.65E-01	1.08E+03	1.24E-01
Indians Diabetes	7.54E-01	7.16E-01	1.72E+02	5.82E-02

From Tables 4.1 and Table 4.2 we can see that in general, the OP-KNN is on the same performance level than Support Vector Machine method. On some data sets, OP-KNN performs worse and on some, better than the SVM. On all the data sets, however, the OP-KNN method is clearly the fastest, with several orders of magnitude. For example, in the Abalone data set using the OP-KNN is more than 200 times faster than the SVM.

On the other hand, the extremely fast speed is not the only advantage of OP-KNN. Since we selected the most significant input variables, this operation highly simplifies the final model, and moreover, make the data and model more interpretable. For example, we select 8 variables from the original 32.

4.5 Summary

As we know, it is usual to have very long computational time for training a feedforward network using existing classic learning algorithms even for simple problems, especially when the number of observations (samples) is relatively small compared to the numbers of input variables. Thus, this dissertation presents OP-KNN method as well as a strategy using OP-KNN to do Variable Selection. This algorithm has several notable achievements:

- keeping good performance while being simpler than most learning algorithms for feedforward neural network,

- using k-NN as the deterministic initialization,

- the computational time of OP-KNN being extremely low (lower than OP-ELM or any other algorithm).

- Variable Selection highly simplifies the final model, and moreover, make the data and model more interpretable.

In the experiment section, we have demonstrated the speed and accuracy of the OP-KNN methodology in ten real applications. Comparing to well-known SVM method, we achieves roughly the same level of accuracy with several orders of magnitude less calculation time. That exactly proves our main goal, which is to show that the method provides very accurate results very fast. This makes it a valuable tool for applications.

5. Extreme Learning Machine based Methods

5.1 Extreme Learning Machine

The Extreme Learning Machine (ELM) algorithm is proposed by Huang *et al.* in [40] as an original way of building a single Hidden Layer Feedforward Neural Network (SLFN).

Given a set of N observations $(x_i, y_i), i \leq N.$ with $x_i \in \mathbf{R}^p$ and $y_i \in \mathbf{R}$. A SLFN with m hidden neurons in the hidden layer can be expressed by the following sum:

$$\sum_{i=1}^{m} \beta_i f(\omega_i x_j + b_i), \quad 1 \leq j \leq N \tag{5.1}$$

where β_i are the output weights, f be an activation function, ω_i the input weights and b_i the biases. Suppose the model perfectly describes the data, the relation can be written in matrix form as $\mathbf{H}\beta = \mathbf{y}$, with

$$\mathbf{H} = \begin{pmatrix} f(\omega_1 x_1 + b_1) & \dots & f(\omega_m x_1 + b_m) \\ \vdots & \ddots & \vdots \\ f(\omega_1 x_n + b_1) & \dots & f(\omega_m x_n + b_m) \end{pmatrix} \tag{5.2}$$

$\beta = (\beta_1, ..., \beta_m)^T$ and $\mathbf{y} = (y_1, ..., y_n)^T$. The ELM approach is thus to initialize randomly the ω_i and b_i and compute the output weights $\beta = \mathbf{H}^{\dagger}\mathbf{y}$ by a Moore-Penrose pseudo-inverse [80]. The essence of ELM is that the hidden layer needs not to be iteratively tuned [39, 40], and moreover, the training error $\parallel \mathbf{H}\beta - \mathbf{y} \parallel$ and the norm of the weights $\parallel \beta \parallel$ are minimized.

The significant advantages of ELM are its extremely fast learning speed, and its good generalization performance while being a simple method [40]. There has been recent advances based on the ELM algorithm, to improve its robustness (OP-ELM [69], TROP-ELM [70], CS-ELM [53]), or make it a batch algorithm, improving at each iteration (EM-ELM [31], EEM-ELM [100]).

5.2 Regularized ELM for Missing Data

5.2.1 Double-Regularized ELM: TROP-ELM

Miche *et al.* in [70] proposed a double regularized ELM algorithm, which uses a cascade of two regularization penalties: first a L_1 penalty to rank the neurons of the hidden layer, followed by a L_2 penalty on the regression weights (regression between hidden layer and output layer). This method is introduced briefly here and used in the next chapter for Missing Data problem.

L_1 penalty: Least absolute shringkage and selection operator (Lasso)
An important part in ELM is to minimize the training error $\parallel \mathbf{H}\beta - \mathbf{y} \parallel$, which is an ordinary regression problem. One technique to solve this is called Lasso, for 'least absolute shrinkage and selection operator' proposed by Tibshirani [92].

Lasso solution minimizes the residual sum of squares, subject to the sum of the absolute value of the coefficients being less than a constant, that's why it is also called 'L_1 penalty'. The general form which Lasso works on is

$$\min_{\lambda,\omega} \left(\sum_{i=1}^{N} (y_i - \mathbf{x_i}\omega)^2 + \lambda \sum_{j=1}^{p} |\omega_j| \right) \tag{5.3}$$

Because of the nature of the constant, Lasso tends to produce some coefficients that are exactly 0 and hence give interpretable models. The shrinkage is controlled by parameter λ. The smaller λ is, the more ω_j coefficients are zeros and hence less variables are retained in the final model.

Computation of Lasso solution is a quadratic programming problem, and can be tackled by standard numeral analysis algorithms. However, a more efficient computation approach is developed by Efron *et al.* in [29], called Least Angle Regression (LARS). LARS is similar to forward stepwise regression, but instead of including variables at each step, the estimated parameters are increased in a direction equiangular to each one's correlations with the residual. Thus, it is computationally just as fast as forward selection. If two variables are almost equally correlated with the response, then their coefficients should increase at approximately the same rate. The algorithm thus behaves as intuition would expect, and

also is more stable. Moreover, LARS is easily modified to produce solutions for other estimators, like the Lasso, and it is effective when the number of dimensions is significantly greater than the number of samples [29].

The disadvantages of the LARS method is that it has problem with highly correlated variables, even though this is not unique to LARS. This problem is discussed in detail by Weisberg in the discussion section of the article [29]. To overcome this, next paragraph introduces Tikhonov Regularization method.

L_2 penalty: Tikhonov Regularization

Tikhonov regularization, named for Andrey Tychonoff, is the most commonly used method of regularization [38]. In statistics, the method is also known as ridge regression.

The general form of Tikhonov regularization is to minimize:

$$\min_{\lambda,\omega} \left(\sum_{i=1}^{N} (y_i - \mathbf{x_i}\omega)^2 + \lambda \sum_{j=1}^{p} \omega_j^2 \right) \tag{5.4}$$

The idea behind of Tikhonov regularization is at the heart of the "bias-variance tradeoff" issue, thanks to it, the Tikhonov regularization achieves better performance than the traditional OLS solution. Moreover, it outperforms the Lasso solution in cases that the variables are correlated. One advantage of the Tikhonov regularization is that it tends to identify/isolate groups of variables, enabling further interpretability.

One big disadvantage of the ridge-regression is that it doesn't have sparseness in the final solution and hence, it doesn't give an easily interpretable result. Therefore, a new idea is created to use a cascade of the two regularization penalties, which is introduced in the next paragraph.

OP-ELM and TROP-ELM

Miche *et al.* in [70] proposed a method OP-ELM, which uses LARS to rank the neurons of the hidden layers in ELM and select the optimal number of neurons by Leave-One-Out (LOO). One problem with LOO error is that it can be very time consuming, especially when the data has large number of samples.

Fortunately, the PREdiction Sum of Squares (PRESS) statistics provide a direct and exact formula for the calculation of the LOO error for linear models, the expression has been shown in 4.2.4. The main drawback of this approach lies in the use of a pseudo-inverse in the calculation, which can lead to numeral instabilities if the data set X is not full rank. This

happens very often in the real world data. Thus, a Tikhonov-Regularized version of PRESS (TR-PRESS) is created:

$$\epsilon^{PRESS}(\lambda) = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{y_i - x_i(X^TX + \lambda I)^{-1}x^Ty}{1 - x_i(X^TX + \lambda I)^{-1}x_i^T}\right)^2 \tag{5.5}$$

This new modified version uses the Singular Value Decomposition (SVD) approach [35] of X to avoid computational issues, and introduces the Tikhonov regularization parameter in the calculation of the pseudo-inverse by the SVD. In practice, the optimization of λ in this method is performed by a Nelder-Mead [73] minimization approach, which converges quickly on this problem.

In general, TROP-ELM is an improvement of original ELM. It first constructs a SLFN like ELM, then ranks the best neurons by LARS (L_1 regularization), finally selects the optimal number of neurons by TR-PRESS (L_2 regularization).

5.2.2 Pairwise Distance Estimation

Pairwise Distance Estimation efficiently estimates the expectation of the squared Euclidean distance between observations in datasets with missing data [30]. Therefore, in general, it can be embedded into any distance-based method, like *K*-nearest neighbor (k-NN), Support Vector Machine (SVM), Multidimensional scaling (MDS), etc., to solve Missing data problem.

Given two samples x and y with missing values, in a d dimensional space. Denote by $M_x, M_y \subseteq [d] = 1, ..., d$ the indexes of the missing components in the two samples. We use x_{obs} and y_{obs} to presents the existing variables of the two samples, which they containing no missing value. Here we assume the data are MCAR or MAR, that is, the missing value can be modeled as random variables, $X_i, i \in M_x$ and $Y_i, i \in M_y$. Thus,

$$x_i' = \begin{cases} E[X_i|x_{obs}] & \text{if} \quad i \in M_x, \\ x_i & \text{otherwise} \end{cases} \tag{5.6}$$

$$y_i' = \begin{cases} E[Y_i|y_{obs}] & \text{if} \quad i \in M_y, \\ y_i & \text{otherwise} \end{cases} \tag{5.7}$$

Where x' and y' is the imputed version of x and y where the missing value has been replaced by its conditional mean. The corresponding conditional variance becomes:

$$\sigma_{x,i}^2 = \begin{cases} Var[X_i|x_{obs}] & \text{if} \quad i \in M_x, \\ 0 & \text{otherwise} \end{cases} \tag{5.8}$$

$$\sigma_{y,i}^2 = \begin{cases} Var[Y_i|y_{obs}] & \text{if} \quad i \in M_y, \\ 0 & \text{otherwise} \end{cases} \tag{5.9}$$

Then, the expectation of the squared distance can be expressed as:

$$E[\| x - y \|^2] = \sum_i ((x_i' - y_i')^2 + \sigma_{x,i}^2 + \sigma_{y,i}^2) \tag{5.10}$$

or, equivalently,

$$E[\| x - y \|^2] = \| x' - y' \|^2 + \sum_{i \in M_x} \sigma_{x,i}^2 + \sum_{i \in M_y} \sigma_{y,i}^2 \tag{5.11}$$

According to Eirola [30], covariance matrix can be achieved through the ECM (Expectation Conditional Maximization) method provided in the MATLAB Financial Toolbox [64]. It is possible to calculate the conditional means and variances of the missing elements using ECM method [67] with some improvements by [85]. Therefore, each pairwise squared distance can be calculated with the missing values replaced by their respective conditional means and by adding the sum of the conditional variances of the missing values respectively.

Since this algorithm is suitable for methods which rely only on the distance between samples, in this dissertation, we use this estimation algorithm embedded Extreme Learning Machine to solve missing data problem.

5.2.3 The Entire Methodology

In this section, the general methodology is presented as well as the details of the implementation steps.

Fig 5.1 illustrates the main components of the whole algorithm, and how they are connected. Therefore, when confronting a regression problem with incomplete data, there are several steps to follow in order to implement this method:

- First of all, it is necessary to replace the missing values with their respective conditional means mentioned in Section 5.2.2. This is a so called 'imputation' step. The reason of this move is because we want to make the whole method more robust.

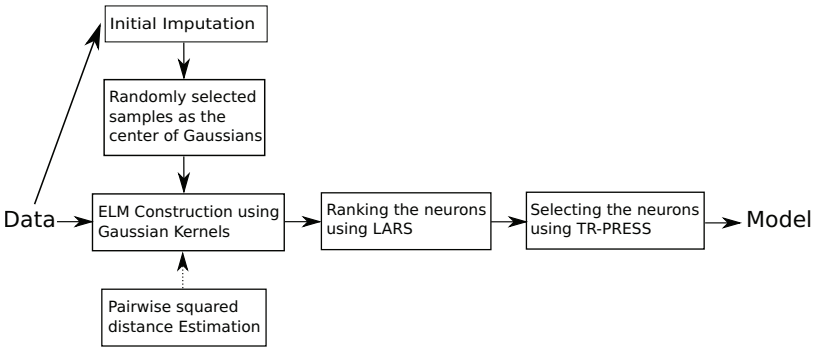


Figure 5.1. The framework of the proposed regularized ELM for missing data

Thus, the accuracy of the distances calculated afterwards is not really based on these imputed values. The main purpose here is to make it possible to use Gaussians as the active function in ELM. Next step explains more about why the imputation is done at the beginning.

- Secondly, we decide to use Gaussian as the active function of the hidden node to build the Single layer feedforward network. Then, m samples are randomly selected from original N samples ($m \leq N$) as the center of Gaussians, that's why the imputation is done in the first step. Choosing the randomly selected samples as the center could anyway guarantee the neural network built here adjoin the data. Therefore, when calculating the output of each neuron, the squared distance between each sample and the selected ones are needed, which are exactly the same thing the Pairwise squared distance estimation method achieved. The hidden node parameters (σ^2, μ) are randomly generated, which remains the advantage of ELM that the parameters in hidden layer need not to be tuned. More specifically, parameter σ^2 is chosen from a interval (20% to 80%) of the original random generations, to further make sure that the model surrounds the data.

- When the distance matrix is ready (by Pairwise distance estimation), with the random generated parameter (σ^2, μ), it is easy to compute the outputs of all the neurons in the hidden layer. The next step would be to figure out the weights (β) between hidden layer and the output of the data (Y).

- The assumption to use LARS is that the problem to be solved should be linear. In fact, this is exactly the case when the neural network built in

previous step, the relationship between the hidden layer and the output in ELM is linear. Therefore, LARS is used to rank neurons according to the output.

- Finally, as mentioned in previous Section 5.2.1, TR-PRESS is used to select the optimal number of neurons, mean square error is minimized through the optimization of parameter λ in Equation 5.5.

The entire algorithm inherits most of the advantage of original ELM, fast computational speed, no parameter need to be tuned, comparatively high generalization performance, etc. Moreover, it perfects ELM with a new tool to solve missing data problem and offers more stable and accurate results with double regularization method.

5.2.4 Experiments

In order to evaluate the method, we also use the UCI database [1] which is the same resources as testing OP-KNN in the previous section.

Table 5.1 shows the specifications of the 5 selected data sets here.

Regression	# Attributes	# Training data	# Testing data
Ailerons	5	4752	2377
Elevators	6	6344	3173
Bank	8	2999	1500
Stocks	9	633	317
Boston Housing	13	337	169

Table 5.1. Specification of the 5 tested regression data sets

These five datasets are split into training, validating and testing sets in a same way as done for OP-KNN experiments. Again, we only need to separate training and testing set because Leave-One-Out validation is used with the training set, i.e. the error we get from the training set is actually the validation error.

Generating the Missing points There is no missing value originally in these 5 datasets. Therefore, missing data is artificially created in each dataset, in order to test the performance on incomplete data with the method. More precisely, the missing data is created (same as deleting the existing data) at randomly position once $1/200$ of the total points till only half data points left. For example, if we have training set with N observations and d features ($N \times d$ data point totally), missing data is cre-

ated $(N \times d)/200$ at a time, and continue 100 times till there is only half data points left ($(N \times d) * 100/200$). Thus, the model is trained and tested 100 times which is so called one round of the experiments.

Monte-Carlo methods [74] refer to various techniques. In this dissertation, Monte-Carlo methods are used to preprocess the data, aiming to two tasks. Firstly, training set is drawn randomly about two thirds of the whole data sets, the rest one third leaves for test set. Secondly, this Monte-Carlo preprocessing is repeated many times for each dataset independently. Therefore, after these rounds of training and testing, an average test error is computed to represent the more general performance of the method.

Other methods used in this dissertation For comparison, mean imputation and 1-nearest neighbor (1-NN) imputation [17, 44] combined with TROP-ELM are tested in this dissertation. Specifically, in the mean imputation method, the mean of corresponding variable is calculated based on the existed samples to replace the missing data; in the 1-NN imputation method, the missing data is replaced by the corresponding variable of its first nearest neighbor whose value is the not missing. Therefore, Pairwise Distance Estimation (PDE), Mean imputation (Mean) and 1-nearest neighbor imputation (1-NN) are used as three different tools here for TROP-ELM to solve the MD problem.

Moreover, this dissertation also tests all the incomplete datasets using TROP-ELM without any MD tools, that means, those samples which contain missing variables are removed (deleted) in order to perform normal TROP-ELM. The main drawback of this method is the huge loss of the training samples. Since the data is missing at random, so when the number of missing points is larger than the sample size, the worst case may happen that no samples left for training. Especially when the percentage of the missing data in the training sets continues to increase, this may happen more and more often. This kind of phenomenon can be seen in the following experiments results.

For each dataset, the same experiment procedure is done to evaluate the method. Firstly, Monte-Carlo split is performed for hundreds of rounds, then for each Monte-Carlo split, missing values are added to training part set by set for 100 times till half of the training values are missing. Once the new missing values are added, the model is trained and tested respectively. Thus, LOO and test results are calculated 100 times with different amount of missing value. In other words, for each different amount of

missing value, the mean LOO errors and test errors are recorded for hundreds of rounds from those Monte-Carlo splits. All the results shown here are the normalized results.

Take the Bank data for instance. There are 4499 samples and 8 variables originally in this data, and one output. For each Monte-Carlo split, 2999 samples are randomly selected for training, and the rest for testing. As to the training set, $(2999 \times 8)/200 \approx 120$ data points are added continuously for 100 times, meaning models are trained and tested for 100 times. Figure 5.2 illustrates the Boston Housing data results. x axis represents the percentage of the missing data from 0% to 50%, while the y axis represents the mean error of the 500 rounds of Monte-Carlo split. More specifically, the results are compared with mean imputation, 1-NN imputation and without any MD tool which are shown in the same figure.

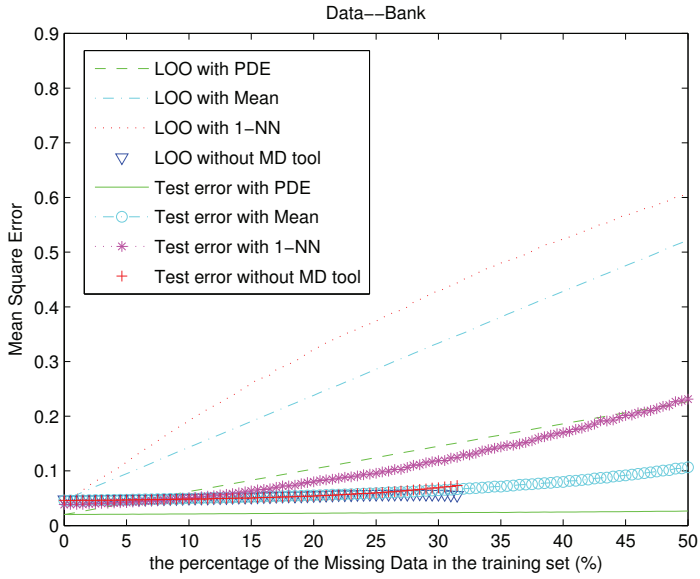


Figure 5.2. Normalized MSE for the dataset - Bank

From Bank figure, we can see that it is risky not to use any MD tool. If the amount of missing data is very small, removing samples may work in some case even it sacrifices much information. But when the amount of missing data increases, there is no reason to take this risk. Like the Bank data, there are not enough samples left to run TROP-ELM when the percentage of MD reaches around 32%. Figure 5.2 also illustrates that PDE tool generally performs better than both mean imputation and 1-NN imputation. Moreover, we can see the LOO error and test error (with PDE) start from a very low value 0.03, then arise smoothly with the increasing

number of missing data. When the amount of missing data reaches as high as half of the whole training set, LOO error is just 0.19 which is still acceptable. As to the test error (with PDE), it performs smaller than LOO error since the beginning. After adding 50% of the Missing data, test error remains on a stable level, around 0.03, which is a significant result we are looking forward to. The results demonstrate the efficiency and stability of the model. On the other hand, test error line vibrates a lot due to the randomness of MD emergences. Nevertheless, the tendency of both LOO and test error keeps the same, and more smoothness can be expected from more rounds of Monte-Carlo test.

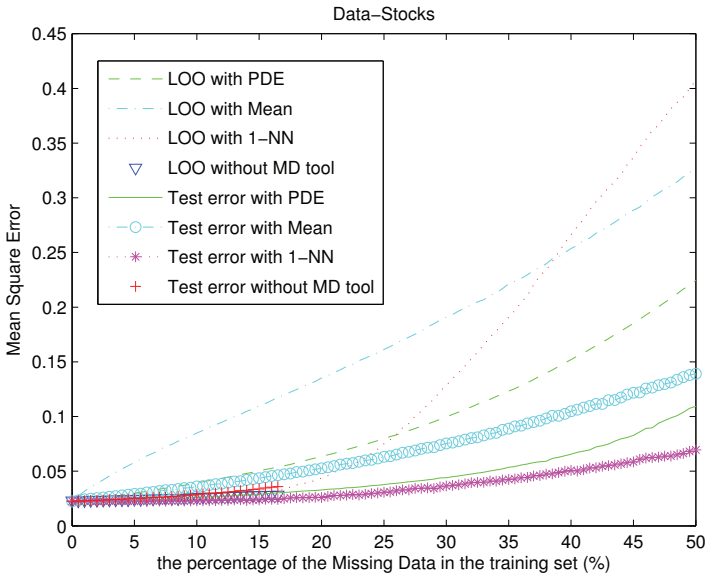


Figure 5.3. Normalized MSE for the dataset - Stock

Figure 5.3, Figure 5.4 and Figure 5.5 show the results for the other four data sets. The results are quite similar with the Bank Data. From both of these four Data results, PDE tools performs better than mean imputation and 1-NN imputation, test errors are less than LOO error sfrom the beginning, and much less vibrations. These prove that models are more stable and reliable.

5.2.5 Conclusions

Briefly speaking, this method is an advanced modification of the original Extreme Learning Machine with a new tool to solve the missing data problem. It uses a cascade of L_1 penalty (LARS) and L_2 penalty (Tikhonov

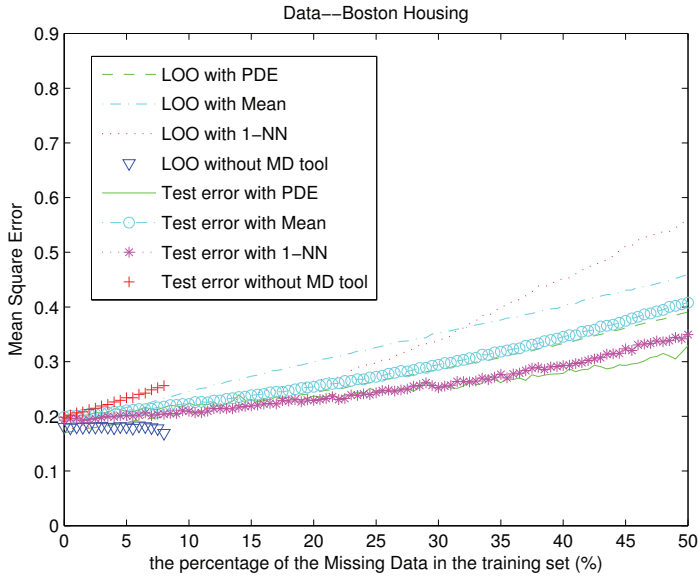


Figure 5.4. Normalized MSE for the dataset - Boston

regularization) on ELM to regularize the matrix computations and hence make the MSE computation more reliable, and on the other hand, it estimates the expected Pairwise distances directly on incomplete data so that it offers the ELM a solution to solve the missing data issues.

According to the experiments of the 5 data sets with hundreds of times Monte-Carlo tests, the method shows its significant advantages: it inherits most of the features of original ELM, fast computational speed, no parameter need to be tuned, etc., and it appears to be more stable and reliable generalization performance by the two penalties. Moreover, according to the the results from our proposed methods which perform much better than TROP-ELM without any missing tool, our method completes ELM with a new tool to solve missing data problem even though the half of the training data are missing as the extreme case.

5.3 Ensemble Delta Test-ELM (DT-ELM)

5.3.1 Bayesian information criterion (BIC)

The Bayesian information criterion (BIC) is one of the most widely known and pervasively used tools in statistical model selection, also known as Schwarz's information criterion (SIC) [84]. It is based, in part, on the

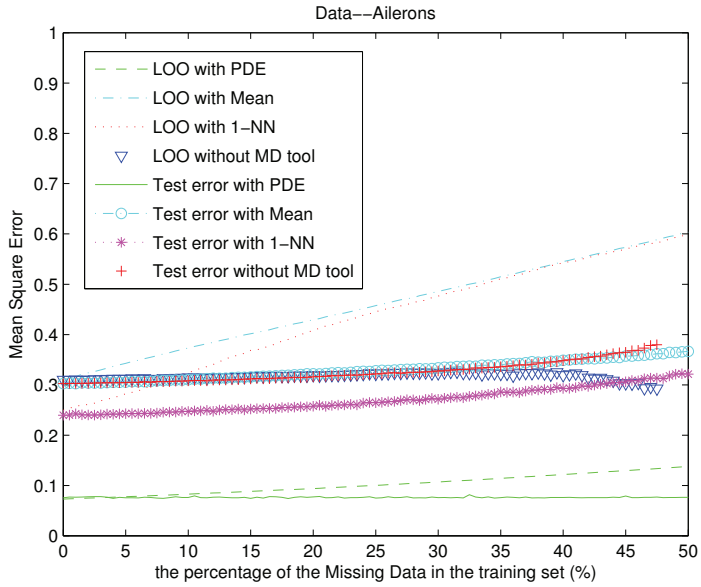


Figure 5.5. Normalized MSE for the dataset - Ailerons

likelihood function, and it is closely related to Akaike information criterion (AIC) [3].

When fitting models, it is possible to increase the likelihood by adding parameters, but doing so may result in overfitting. The BIC resolves this problem by introducing a penalty term for the number of parameters in the model.

In brief, BIC is defined as:

$$\mathrm{BIC} = -2 \cdot \ln L + m \ln(N) \tag{5.12}$$

where,

- N–the number of observations, or equivalently, the sample size;
- m–the number of degrees of freedom remaining after fitting the model (free parameters to be estimated), with smaller value representing the better fits. If the estimated model is a linear regression, m is the number of regressors, including the intercept;
- L–the maximized value of the likelihood function for the estimated model.

Under some assumptions of model errors, BIC becomes the following formula for practical calculations [79]:

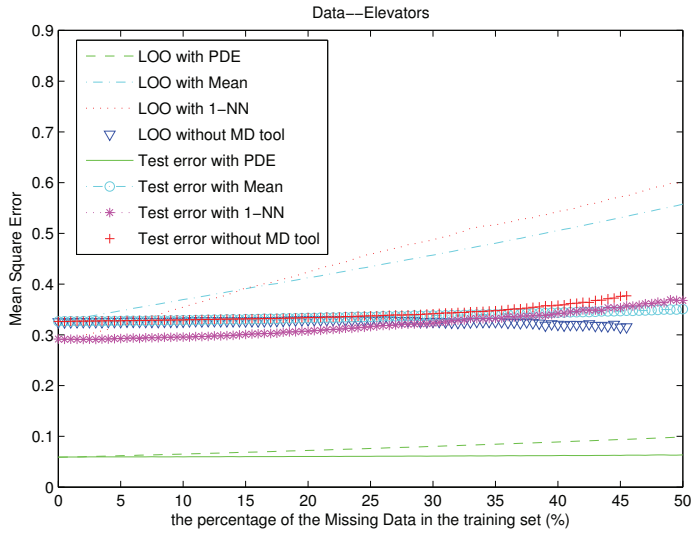


Figure 5.6. Normalized MSE for the dataset - Elevators

$$\mathrm{BIC} = N \cdot \ln(\widehat{\sigma_e^2}) + m \cdot \ln(N) \tag{5.13}$$

where $\widehat{\sigma_e^2}$ is the error variance.

Because BIC includes an adjustment for sample size, the BIC often favors a simpler model. In this dissertation, BIC is used to selected neurons incrementally in ELM, which are randomly generated and tested cluster by cluster. Therefore, BIC is calculated like:

$$\mathrm{BIC} = N \cdot \ln(MSE) + m \cdot \ln(N) \tag{5.14}$$

where N continues to be the number of samples, MSE represents the Mean Square error for the regression problem, and m is the number of neurons used in current model.

However, BIC is in theory designed only for data set of an infinite sample size, and in practice, it is really difficult to find the balance point between smaller error and not overfitting, even though BIC is used instead of least squares in the proposed method. Therefore, only BIC couldn't offer sufficient restrictions to ELM, and Delta test (DT) is used with BIC in this dissertation. DT is introduced in next section.

5.3.2 Nonparametric Noise Estimator (NNE): Delta Test

Delta test (DT) is a non-parametric technique based on nearest neighbors principle. It is a fast scalable algorithm for estimating the noise variance

presented in a data set modulo the best smooth model for the data, regardless of the fact that this model is unknown [2]. A useful overview and general introduction to the method and its various applications is given in [48]. The evaluation of the NNE is done using the DT estimation introduced by Stefansson [90].

In the standard DT analysis, we consider vector-input/scalar-output data sets of the form

$$(x_i, y_i | 1 \leq i \leq M) \tag{5.15}$$

where the input vector $x_i \in \mathbb{R}^d$ is confined to some closed bounded set $C \subset R^d$. The relationship between input and output is expressed by $y_i = f(x_i) + r_i$, where f is the unknown function and r is the noise. The Delta test estimates the variance of the noise r.

The Delta test works by exploiting the hypothesized continuity of the unknown function f. If two points x and x' are close together in input space, the continuity of f implies that the points $f(x)$ and $f(x')$ will be close together in output space. Alternatively, if the corresponding output values y and y' are not close together in output space, this can only be due to the influence of noise on $f(x)$ and $f(x)$.

Let us denote the first nearest neighbor of the point x_i in the set $\{x_1, \ldots, x_N\}$ by x_{NN}. Then the Delta test, δ is defined as:

$$\delta = \frac{1}{2N} \sum_{i=1}^{N} \left| y_{NN(i)} - y_i \right|^2 \tag{5.16}$$

where $y_{NN(i)}$ is the output of $x_{NN(i)}$. For the proof of the convergence of the Delta test, see [48].

In a word, the Delta test is useful for evaluating relationship between two random variables, namely, input and output pairs. The DT has been introduced for model selection but also for variable (feature) selection: the set of inputs that minimizes the DT is the one that is selected. Indeed, according to the DT, the selected set of variables (features) is the one that represents the relationship between variables and output in the most deterministic way.

In this dissertation, Delta test is used between the output of the hidden layer and the real output, following the BIC criterion, to further validate the selection of the ELM neurons.

5.3.3 Delta test ELM: DT-ELM

In this section, the general frame of the DT-ELM methodology is presented as well as the details of the implementation steps.

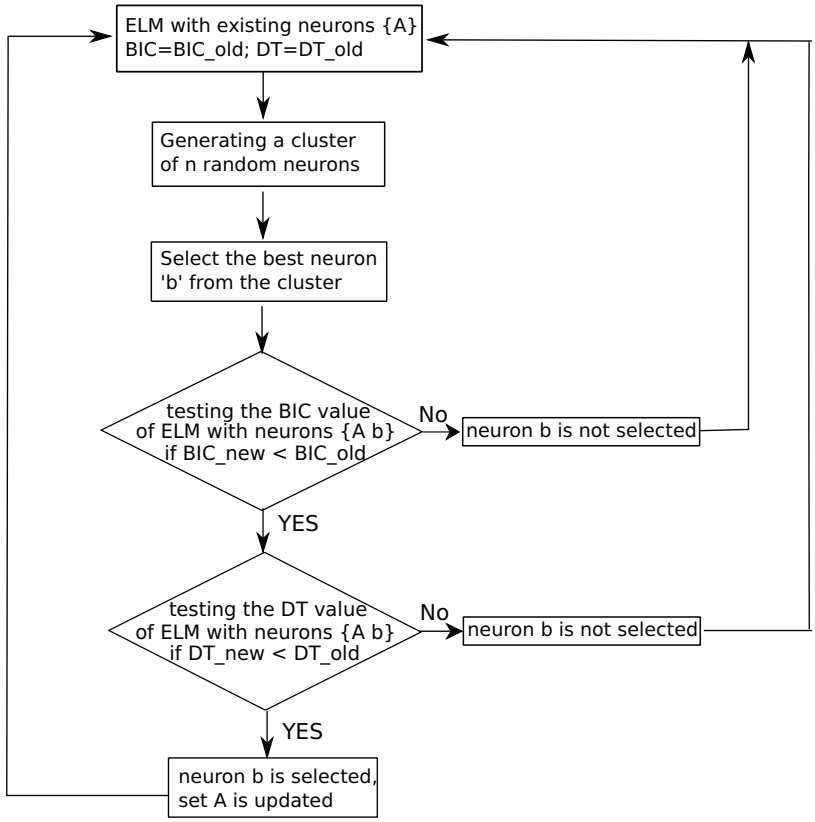


Figure 5.7. The framework of the proposed DT-ELM method

Fig 5.7 illustrates the main procedures of DT-ELM, and how they interact. In general, DT-ELM is a robust method where no parameters need to be tuned. Unlike most of other ELM related methods, DT-ELM has the ability to run without setting expected training error or the maximum number of neurons beforehand. It will reach the balance point automatically. The algorithm of DT-ELM can be summarized as follow:

Given a training set $(x_i, y_i)|x_i \in R^d, y_i \in R, i = 1, 2, ..., d$, activation function $f(x)$. Each trial cluster contains n neurons.

Initialization step: Let the number of hidden neurons to be zero at the very beginning, then the neurons could be chosen progressively later on

by DT-ELM. Set the initial BIC and DT value to be infinite, so that the following steps are always trying to add neurons to DT-ELM to minimize BIC and DT results.

Learning step:

- Randomly generate a cluster of n neurons. n is optional that can be configured according to the different computer power or different data sets. It saves computational time to test neurons cluster by cluster, instead of one by one.

- Construct ELM using the combination of each neuron and the existing selected neurons.(For the first round, it means to construct ELM with each neuron separately). Test the BIC value for each new ELM, find the neuron that gives the smallest BIC.

- Check whether the smallest BIC is smaller than the previous BIC value. If so, continue to next step; otherwise, stop current trial and repeat the learning step. In practice, the value of BIC decreases easily and fast at the beginning, but becomes more and more difficult with the increasing number of neurons.

- Calculate the DT value between the hidden layer and the output for the ELM with the existing neuron and the neuron found in previous step. If the DT results get decreased, this new neuron is added; otherwise stop the current round and repeat the learning step. It is similar with BIC value at the beginning, DT decreased quite fast, but with the increase number of neurons, it becomes extremely difficult to find a new satisfying neuron.

Stop criterion

One advantage of DT-ELM is that no parameter needs to be set beforehand, the number of neurons is chosen automatically according to the algorithm. Therefore, when to stop finding new neurons becomes an issue for this method. In this dissertation, the default setting is 200 extra clusters. As we mentioned that the neurons are tested cluster by cluster, instead of one by one in other incremental learning algorithm. Therefore, this means DT-ELM stops training if DT values doesn't decrease for continuous 4000 new neurons (here each cluster contains $n = 20$ neurons).

Example Take data set Bank (more details in Experiment Section) for example, Bank has 8 attributions (variables) and 4499 samples, from which 2999 samples are randomly selected for training and the rest 1500 for test.

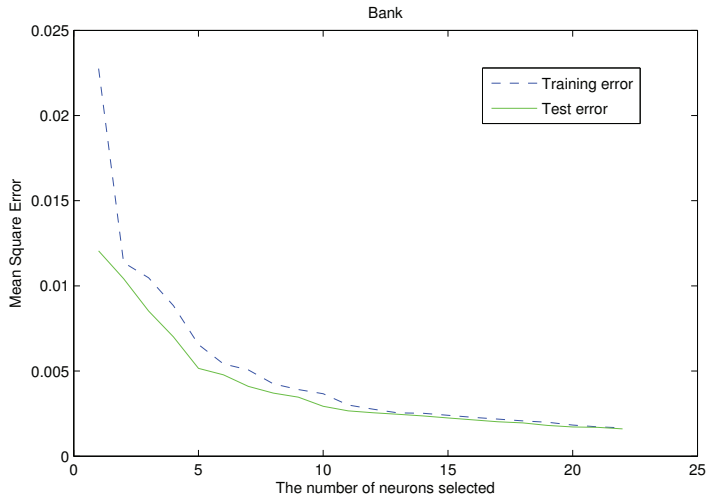


Figure 5.8. Mean Square Error for Bank, versus the number of Neurons

Fig 5.8 illustrates the results of training and testing on bank data using DT-ELM. For this trial, 369 clusters are generated and tested for selection, and 23 neurons are selected eventually.

5.3.4 Ensemble modeling

No guideline is always correct. No single method is always the best. This has lead to the idea of trying to combine models into an ensemble rather than selecting among them [16]. The idea seems to work well as demonstrated by many practical applications [8, 68].

It is stated [8] that a particular method for creating an ensemble can be better than the best single model. Therefore, how to combine the models becomes an issue. There are several ways to achieve this. One example is using Non-Negative constrained Least-Squares (NNLS) algorithm [54]. In this dissertation, we use the equalized weights for all the ensemble models and it works well as shown in the Experiments.

The ensemble error can be calculated between $y_{Ensemble}$ and y, where $y_{Ensemble} = \sum_{i=1}^{k} \omega_i \hat{y}_i$ is the weighted sum of the output of each i individual models, ω_i is the weighted assigned to the output of the ith model; these weights satisfy $\sum_i \omega_i = 1$; y is the real output of the data and $\hat{y}_i$ is ensem-

ble target. In this dissertation, we want to build k models and more particularly, $\omega_i = \frac{1}{k}$. Thus, the final output we obtain is $y_{Ensemble} = \sum_{i=1}^{k} \frac{1}{k} \hat{y}_i$.

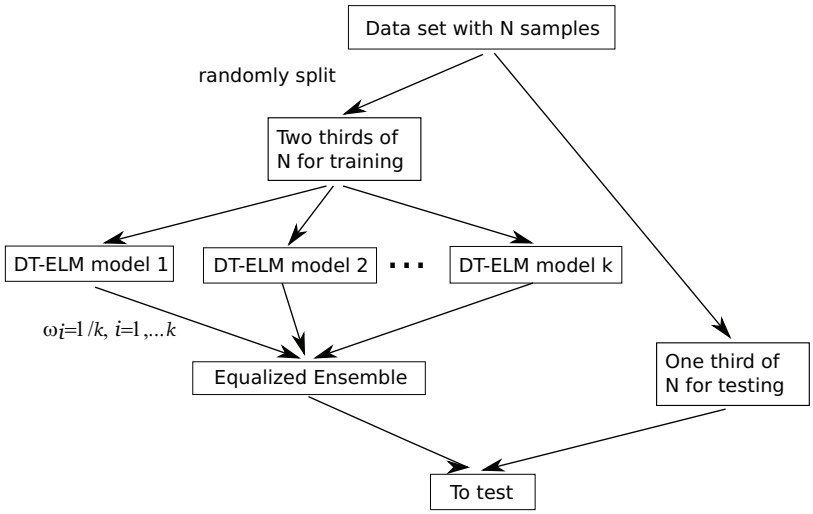


Figure 5.9. The framework of the proposed Ensemble DT-ELM method

Fig 5.9 shows more details on how the ensemble DT-ELM works in this dissertation. As we know that DT-ELM is based all on randomness, so that even using the same training samples, the model built varies time by time. Therefore, for each training set, 50 models (DT-ELM) are constructed, aiming to acquire more stable results. Thanks to the fast construction speed of ELM, 50 doesn't bring too much computational time. On the other hand, other even bigger numbers have been tested but no consequent improvement. So we choose 50 models. Then the ensemble step assigns the same weights $\omega = \frac{1}{50}$ to each output of the model y_i. So the training result of the Ensemble DT-ELM is $y_{train} = \frac{1}{50} \sum_{i=1}^{50} y_i$.

5.3.5 Experiments

Eight data sets from UCI machine learning repository [1] are used. They are chosen for their heterogeneity in terms of problem, number of variables, and sizes. Six data sets are for regression problem and two are for classification problem. The specification of the 8 selected data sets can be found in 4.4.2.

The data sets have all been processed in the same way: for each data set, 10 different random permutations are taken without replacement; for

each permutation, two thirds are taken for the training set, and the remaining third for the test set (see Table 1). Training sets are then normalized (zero-mean and unit variance) and test sets are also normalized using the very same normalization factors than for the corresponding training set. The results presented in the following are hence the average of the 10 repetitions for each data set.

The regression performance of the Ensemble DT-ELM is compared with OP-ELM, ELM and other well-known machine learning methods like Support Vector Machine (SVM), Multilayer Perception (MLP), and Gaussian process for Machine Learning (GPML). Here, the OP-ELM was using a maximum number of 100 neurons.

Firstly, the mean square errors for the six algorithms tested are reported in Table 5.2. Table 5.3 and Table 5.4 then illustrate the computational time and the number of neurons selected, respectively. As seen from these tables, the test results of Ensemble DT-ELM performs at least as good as OP-ELM, with relatively similar computational time, but much simpler model eventually. The number of neurons E.DT-ELM selected is smaller than half of the number with OP-ELM, for some cases, like Ailerons and Elevators, E.DT-ELM uses around 4 neurons instead of about 70 neurons of OP-ELM.

	Regression						Classification	
	Ailerons	Elevators	Servo	Bank	Stocks	Boston	Cancer	Diabetes
E. DT-ELM	3.2e-8	2.1e-6	5.3e-1	1.4e-3	6.6e-1	1.6e+1	96.8%	75.8%
	(1. 0e-7)	(5.0e-6)	(1.9)	(3.2e-3)	(1.1)	(49)	(8.2e-3)	(2.1e-2)
OP-ELM	2.8e-7	2.0e-6	8.0e-1	1.1e-3	9.8e-1	1.9e+1	95.6%	74.9%
	(1.5e-9)	(5.4e-8)	(3.3e-1)	(1.0e-6)	1.1e-1	(2.9)	(1.3e-2)	(2.4e-2)
ELM	3.3e-8	2.2e-6	7.1	6.7e-3	3.4e+1	1.2e+2	95.6%	72.2%
	(2.5e-9)	(7.0e-8)	(5.5)	(7.0e-4)	(9.35)	(2.1e+1)	(1.2e-2)	(1.9e-2)
GP	2.7e-8	2.0e-6	4.8e-1	8.7e-4	4.4e-1	1.1e+1	97.3%	76.3%
	(1.9e-9)	(5.0e-8)	(3.5e-1)	(5.1e-5)	(5.0e-2)	(3.5)	(9.0e-3)	(1.8e-2)
MLP	2.7e-7	2.6e-6	2.2e-1	9.1e-4	8.8e-1	2.2e+1	95.6%	75.2%
	(4.4e-9)	(9.0e-8)	(8.1e-2)	(4.2e-5)	(2.1e-1)	(8.8)	(1.9e-2)	(1.9e-2)
SVM	1.3e-7	6.2e-6	6.9e-1	2.7e-2	5.1e-1	3.4e+1	91.6%	72.7%
	(2.6e-8)	(6.8e-7)	(3.3e-1)	(8.0e-4)	(9.0e-2)	(3.1e+1)	(1.7e-2)	(1.5e-2)

Table 5.2. Mean Square Error results for comparison. Standard derivations in brackets. For classification, the showing results are the correct classification rate.

5.3.6 Summary

Ensemble DT-ELM assembles from a number of DT-ELM models trained with the same training set. BIC and DT is applied into the algorithm with

	Ailerons	Elevators	Servo	Bank	Stocks	Boston	Cancer	Diabetes
E. DT-ELM	5.9	8.7	6.4e-1	3.1e+1	2.12+1	1.0e+1	1.25	8.1e-1
OP-ELM	16.8	29.8	2.1e-1	8.03	1.54	7.0e-1	2.8	1.7e+2
ELM	9.0e-1	1.6	3.9e-2	4.7e-1	1.1e-1	7.4e-2	1.2e-1	1.7e-1
GP	2.9e+3	6.5e+3	2.2	1.7e+3	4.1e+1	8.5	6.1	5.8
MLP	3.5e+3	3.5e+3	5.2e+2	2.7e+3	1.2e+3	8.2e+2	2.3e+3	6.0e+2
SVM	4.2e+2	5.8e+2	1.3e+2	1.6e+3	2.3e+3	8.5e+2	1.1e+3	6.8e+2

Table 5.3. Computational times (in seconds) for comparison

	Ailerons	Elevators	Servo	Bank	Stocks	Boston	Cancer	Diabetes
E. DT-ELM	3.3	3.7	9.6	19.4	43	33.8	6	13
OP-ELM	75	74	36	98	100	66	43	56

Table 5.4. Average (over the ten repetitions) on the number of neurons selected for the final model for both OP-ELM and Ensemble DT-ELM

the penalty of the number of the neuron and the estimated variance of the noise between hidden layer and the output. So that DT-ELM method adds neurons incrementally and stops when couldn't decrease both BIC and DT values.

The significant advantages of this method are its robustness and the sparsity of the model. There is no parameter needed to be tuned and it constructs much more sparse model. As we know that the less hidden nodes used, the more interpretable of the model. On the other hand, ensemble DT-ELM maintains the fast speed even it stops after 4000 unsuccessful test of neurons. These are also proved by the experiments. In the experiments section, six real regression data sets have been tested, and the results show that DT-ELM maintains the fast computational time, the good performance, but constructs much sparse models. (The number of hidden nodes selected is much less than OP-ELM).

5.4 Incremental Extreme Learning Machine with Leave-One-Out (LOO-IELM)

Many methods have been exploited recently trying to choose the most suitable network structure of ELM and to further reduce the number of neurons without affecting the generalization performance. Pruning methods are one type of algorithms to address this problem. For example, Rong *et al.* in [81] proposed a pruned ELM (P-ELM) for classification, and Miche *et al.* in [69, 70] presented a method called optimally pruned ELM (OP-ELM). But pruning methods in general are rather inefficient since most of

the time they are dealing with a network structure larger than necessary. On the other hand, some researchers manage to solve the problems via incremental learning. Like the Incremental extreme learning machine (I-ELM) [41] which adds randomly generated hidden nodes one-by-one to the hidden layer until achieving an expected training accuracy or reaching the maximum number of hidden nodes. There are also some modifications made to I-ELM, like shown in [42, 32, 43]. However, these methods need to set the expected training error or maximum number of neurons in advance.

Thus, another method we would like to propose here is called Leave-One-Out-Incremental Extreme Learning Machine (LOO-IELM). It is operated in an incremental way but stops automatically based on the stop criteria. In general, the method can be operated as the following: Fig 5.10

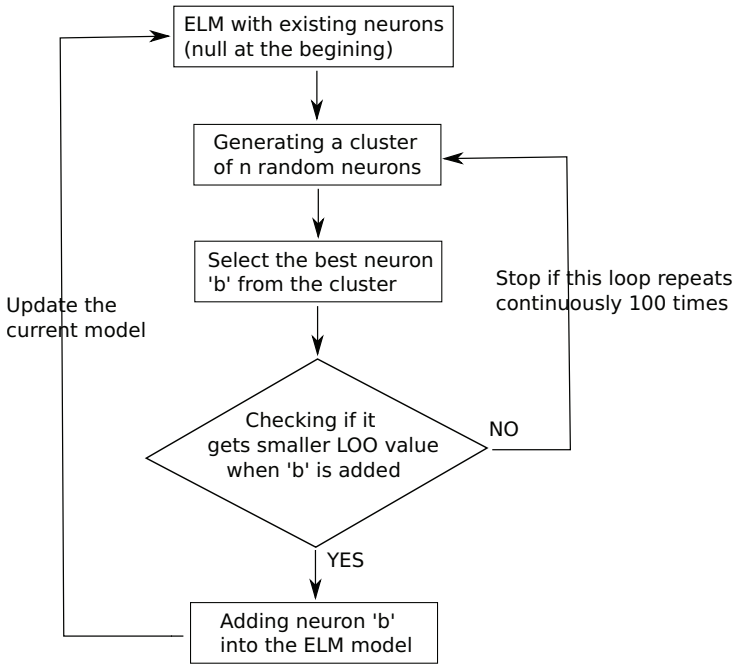


Figure 5.10. The framework of the LOO-IELM method

illustrates the main procedures of LOO-IELM, and how they interact. The neurons of the ELM are selected and added incrementally till no smaller LOO value can be found. Next paragraphs concentrate on more details of this method.

Given a training set $(x_i, y_i)|x_i \in \mathbf{R}^d, y_i \in \mathbf{R}, i = 1, 2, ..., d$, activation function $f(x)$. Each trial cluster contains n neurons.

5.4.1 Incremental strategy

Initialization step

Let the number of hidden neurons to be zero at the very beginning, then the neurons could be chosen progressively later on by LOO-IELM.

Learning step:

- Randomly generate a cluster of n neurons. n is optional that can be configured according to the different computer power or different data sets. It saves computational time to test neurons cluster by cluster, instead of one by one. In this dissertation, n is chosen to be 20.

- Construct ELM using the combination of each of the n neurons and the existing selected neurons. That means ELM models are build 20 times in this step. (For the first round, it means to construct ELM with each neuron separately). Test the LOO value for each of these ELMs, find the neuron "b" that gives the smallest LOO.

- Check whether the LOO value with neuron "b" is smaller than previous one. If so, continue to next step; otherwise, stop current trial and repeat the learning step.

Stop criterion

One advantage of LOO-IELM is that no parameter needs to be set beforehand, the number of neurons is chosen automatically according to the algorithm. Therefore, when to stop finding new neurons becomes an issue for this method. In this dissertation, the default setting is 100 extra clusters. As we mentioned that the neurons are tested cluster by cluster, instead of one by one in other incremental learning algorithm. Therefore, this means LOO-IELM stops training if LOO value doesn't decrease for continuous 2000 new neurons (here each cluster contains $n = 20$ neurons).

5.4.2 Summary

In a word, LOO-IELM uses PRESS statistics to calculate LOO, in order to select the best neurons in an incremental way. Therefore, LOO-IELM has an advantage that it achieves an optimal set of neurons while no parameters need to be tuned. Moreover, it maintains the fast computational speed as original ELM. The performance of LOO-IELM is evaluated and

discussed in next section, along with some strategies designed for the specific dataset.

6. Real cases of French Retails companies

In most of the studies, bankruptcy prediction is treated as a binary classification problem. The target (output) variable of the models is commonly a dichotomous variable where “firm filed for bankruptcy” is set to 1 and “firm remains solvent” is set to 0. The reference (input) variables contain information about liquidity, solvency, leverage, profitability, asset composition, firm size, growth, cash flow information and other features of interest that include information on macroeconomic, industry specific, location or spatial. There is no agreement on which information are necessary for the prediction, and the selection of the indicators itself is a very difficult topic. Recently, financial ratios are quite popular in many articles for bankruptcy prediction and the choice of ratios varies in accordance with different algorithms and different goals (for companies, for bank, etc.). To continue the discussion, we choose the datasets originally collected by Philippe du Jardin [45] for experiments in this dissertation.

6.1 The Dataset Summary

Philippe du Jardin collected and built this data from French retail companies for year 2002 and 2003. The dataset of 2002 comprises companies that have accounting data from the year 2002 and net equity data from the year 2001. The bankruptcy decisions, or more accurately, decisions of reorganization or liquidation, are from the year 2003. The dataset of 2003 was constructed similarly. In both datasets, the proportion of healthy and bankrupted companies is balanced. In total, there are 500 and 520 samples, respectively. The companies are all from the trade sector and they have a similar structure, juridically and from the point of view of the assets. In addition, the healthy companies were still running in 2005, and had activities at least during four years. The ages of the companies were

also considered, in order to obtain a good partition of companies of different ages [45].

Both of the datasets have 41 input variables originally, which were divided into six groups; The first represents the performance of the firms (such as for instance EBITDA/Total assets), the second the efficiency (such as for instance Value added/Total sales), the third the financial distress (such as for instance financial expenses/Total sales), the fourth the financial structure (such as for instance Total debt/Total equity), the fifth the liquidity (such as for instance quick ratio) and the sixth the rotation (such as for instance Accounts payable/Total sales). The labels of the variables are presented in Table 6.1. The output variable (target variable) is labeled with -1 or 1, where -1 indicates a healthy result and 1 indicates its bankruptcy ending after one year.

6.2 Practical operation on the Outliers

As we briefly mentioned in previous chapter, an outlier is an observation that appears to deviate markedly from other observations in the sample. Outliers may otherwise adversely lead to model misspecification, biased parameter estimation and incorrect results. It is therefore important to identify them prior to modeling and analysis [96, 59]. In this dissertation, we use financial expertise of some experts to detect the outliers.

6.2.1 Outliers Detection using Financial Expertise

There are many existing methods for outlier detection based on different assumptions. Outlier Detection itself is a wide and complex topic, which needs to be explored in depth. However, outlier problems are not the main issue here and our goal is to uses it as a tool without making an exhaustive study. Therefore, this dissertation uses financial expertise to achieve this task. Intervals are given for each variable (ratio) by some skilled experienced experts, so that those values outside the boundaries are intolerant.

Table 6.2 illustrates the tolerant intervals for each variable. The values which do not lie inside the corresponding range, are considered as outliers. Outliers here are defined by common sense, which means the thresholds (intervals) are tolerant enough to cover some extreme cases.

X1	Profit before Tax/Shareholders' Funds
X2	Net Income/Shareholders' Funds
X3	EBITDA/Total Assets
X4	EBITDA/Permanent Assets
X5	EBIT/Total Assets
X6	Net Income/Total Assets
X7	Value Added/Total Sales
X8	Total Sales/Shareholders' Funds
X9	EBIT/Total Sales
X10	Total Sales/Total Assets
X11	Gross Trading Profit/Total Sales
X12	Operating Cash Flow/Total Assets
X13	Operating Cash Flow/Total Sales
X14	Financial Expenses/Total Sales
X15	Labor Expenses/Total Sales
X16	Shareholders' Funds/Total Assets
X17	Total Debt/Shareholders' Funds
X18	Total Debt/Total Assets
X19	Net Operating Working Capital/Total Assets
X20	Long Term Debt/Total Assets
X21	Long Term Debt/Shareholders' Funds
X22	(Cash + Marketable Securities)/Total Assets
X23	Cash/Total Assets
X24	(Cash + Marketable Securiti es)/Total Sales
X25	Quick Ratio
X26	Cash/Current Liabilities
X27	Current Assets/Current Liabilities
X28	Quick Assets/Total Assets
X29	Current Liabilities/Total Assets
X30	Quick Assets/Total Assets
X31	EBITDA/Total Sales
X32	Financial Debt/Cash Flow
X33	Cash/Total Debt
X34	Cash/Total Sales
X35	Inventory/Total Sales
X36	Net Operating Working Capital/Total Sales
X37	Accounts Receivable/Total Sales
X38	Accounts Payable/Total Sales
X39	Current Assets/Total Sales
X40	Change in Equity Position
X41	Change in Other Debts

Table 6.1. The variables used in the du Jardin datasets. EBITDA = Earnings Before Interest, Taxes, Depreciation and Amortization.

6.2.2 Outliers treatment

Let us first look at Fig 6.1 which shows the distribution of the outliers. A common procedure for data processing is to remove the outliers. In our case, that means to remove all the samples containing the outliers, or to

X1	[-30, 30]	X2	[-10, 10]	X3	[-2, 2]	X4	[-10, 10]
X5	[-1, 1]	X6	[-1, 1]	X7	[-0.2, 1]	X8	[-100, 100]
X9	[-0.5, 0.5]	X10	[0.5, 10]	X11	[-1, 1]	X12	[-2, 2]
X13	[-0.5, 0.5]	X14	[0, 0.1]	X15	[0, 1]	X16	[0, 1]
X17	[$-$ inf, 10]	X18	[0, 10]	X19	[$-$ inf, 1]	X20	[0, 1]
X21	[-10, 10]	X22	[0, 1]	X23	[-1, 1]	X24	[0, 0.5]
X25	[0, 10]	X26	[-1, 5]	X27	[0, 10]	X28	[0, 1]
X29	[0, 1]	X30	[0, 1]	X31	[-0.5, 0.5]	X32	[0, 30]
X33	[-1, inf]	X34	[-1, 1]	X35	[0, 1]	X36	[-1, 1]
X37	[0, 1]	X38	[0, 1]	X39	[0, inf]	X40	None
X41	None						

Table 6.2. Tolerant intervals for each variables

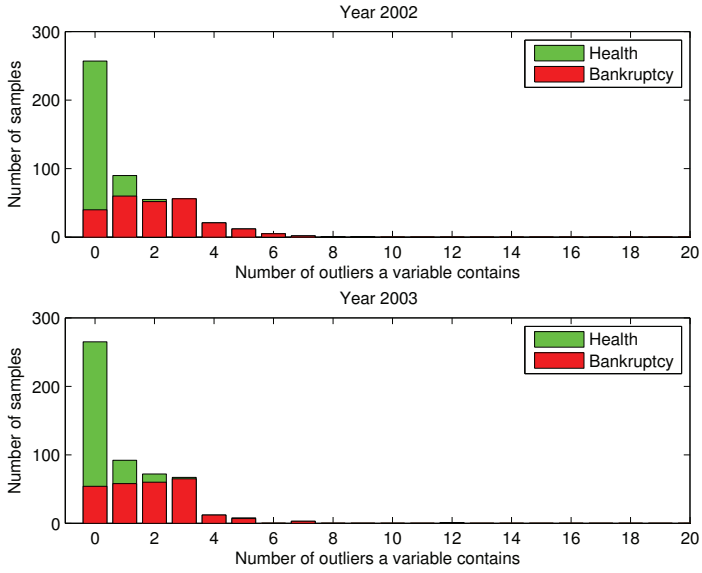


Figure 6.1. Outlier distribution for data from year 2002 and 2003

remove all the variables containing the outliers. Although outliers are often considered as an error or noise, they may carry important information, let along remove other normal values. Thus, in this dissertation, each outlier is considered to be a missing value of the data.

Two imputation methods are proposed here for the missing data problems. One method is to replace the missing value using the threshold. For example, the tolerant interval given for a margin indicator (Value Added/Total Sales) is $[-0.2, 1]$ by experts. If a sample (company) has a value of this indicator -5 which is smaller than the lower bound, this value is replaced by threshold -0.2; if its value is 4 which is larger than the higher bound, it is replaced by the threshold 1. In this way, the size of

the dataset maintains the same as originally and moreover, the abnormal values remain in an extreme level while not misleading the model.

Another imputation method we used is TROP-ELM with Pairwise distance estimation, which has been introduced in Section 5.2. In the following experiment part, the performance as well as comparison of these treatments on outliers are illustrated and analyzed.

6.3 Feature (Variable) Selection or not, or how

Variable Selection has several important advantages if it is performed properly. It helps to decrease the redundancy of the original data, it can also reduce the complexity of the modeling process. Moreover, it contributes to the interpretability of the input variables which is very useful to analyze the data and even the model.

Forty-one variables are obviously too much to build the model, especially these bring difficulties to interpret the model. As we can see from Table 6.1, many variables are correlated because they are financial ratios by cross calculations. Thus, it is important to select variables to reduce the redundancy as much as possible and meanwhile to retain the necessary information for prediction.

6.3.1 Financial Preknowledge versus Algorithmic Selection

Generally speaking, there are two ways to select the variables, either by automatic black-box variable selection or by financial expertise.

For the algorithmic modeling aspect, there is great variety in bankruptcy prediction models from how many and which factors are considered to what methods are employed to develop the model. For example, Altman's model [5] is a five-factor multivariate discriminant analysis model while Boritz and Kennedy's model [13] is a 14-factor neural network. The number of factors considered in other models ranges from one to 57 factors. In this dissertation, 9 variables are chosen for the French datasets, referring to the results of Publication IV on the same data. In Publication IV, ensembles of locally linear models are built using a forward variable selection technique. The 9 best variables are selected, which are X1, X2, X3, X4, X5, X6, X16, X17 and X18 in Table 6.1.

On the other hand, according to some financial experts (Du Jardin and Séverin [46]), 12 variables are chosen based on their experience. They are

X3, X7, X14, X25, X26, X27, X32, X34, X35, X36, X37 and X38.

For these two sets of variables (9V and 12V respectively), the goal of this dissertation is to contrast their performance and furthermore, make use of advantages of both two sets.

6.4 Combo method

6.4.1 Outliers concentrates on several variables

Table 6.3 and Table 6.4 illustrate the intervals and the number of outliers for each selected variables on year 2002 and 2003.

Index	X1	X2	X3	X4	X5	X6	X16	X17	X18
Interval	[-30,30]	[-10,10]	[-2,2]	[-10,10]	[-1,1]	[-1,1]	[0,1]	$[-\infty,10]$	[0,10]
Year 2002									
Number of the outliers	9	13	0	8	11	13	139	1	0
Number of the Bankrupt outliers	7B	13B	0B	8B	11B	13B	139B	0B	0B
Year 2003									
Number of the outliers	2	8	0	1	8	9	158	0	0
Number of the Bankrupt outliers	2B	7B	1B	7B	8B	9B	145B	0B	0B

Table 6.3. The 9 variables selected by locally linear models. B: Bankruptcy, 7B: among the 9 samples containing outliers of X1, there are 7 samples lead to bankruptcy.

As shown in the Table 6.3, most of the outliers concentrate on the variable X16, which is Shareholders' Funds/Total Assets. Especially, the samples containing outliers of X16 mostly lead to bankruptcy (all 139 samples for 2002 and 145 out of 158 samples for 2003). This feature could create an independent classifier (indicator) X16, which operates as: if the value of X16 is normal (inside the interval [0,1]), the prediction decision is healthy, otherwise, the sample is predicted as bankruptcy. And the accuracy can achieve as high as $139/139 := 100\%$ for 2002 and $145/158 := 91.77\%$ for 2003. In the following ensemble section, this indicator will be combined with other classifiers to further improve the performance.

The similar status occurs in the 12 variables shown in the Table 6.4. In this case, the special variable turns to be X32 instead of X16 in the 9 variables set. X32 defined as Financial Debt/Cash Flow and most of its outliers indicate bankruptcy (175 out of 187 samples for 2002 and 161 out

Index	X3	X7	X14	X25	X26	X27	X32	X34	X35	X36	X37	X38
Interval	[-2,2]	[-0.2,1]	[0,0.1]	[0,10]	[-1,5]	[0,10]	[0,30]	[-1,1]	[0,1]	[-1,1]	[0,1]	[0,1]
Year 2002												
Number of the outliers	0	5	29	0	1	10	187	0	0	1	1	3
Number of the Bankrupt outliers	0B	5B	17B	0B	0B	0B	175B	0B	0B	1B	1B	3B
Year 2003												
Number of the outliers	1	4	0	1	2	1	194	0	1	2	0	0
Number of the Bankrupt outliers	1B	4B	0B	0B	0B	0B	161B	0	1B	2B	0B	0B

Table 6.4. The 12 variables selected by financial experts for 2002.
B: Bankruptcy, 161B: among the 194 samples containing outliers of X32, there are 161 sample leads to bankruptcy.

of 194 samples for 2003). Of course, this is not a perfect classifier with the maximum accuracy $175/187 := 93.58\%$ and $161/194 := 82.99\%$, however, it can be improved by combining other classifiers.

6.4.2 Some strategies on the datasets (Combo method)

Take the 9 variables of year 2002 for example. From the original 500 samples, two thirds (375 samples) are taken for training the model and the rest (125 samples) for testing. And all the training and testing processes are repeated for 100 times to acquire more general performance. The following Table 6.5 illustrates the confusion matrix on average of 100 repetitions using LOO-IELM.

		Predicted Classes	
		Health	Bankruptcy
Actual Classes	Health	60.06	1.94
	Bankruptcy	6.56	56.44

Table 6.5. Confusion matrix for the 9V of 2002, using LOO-IELM. (Average number of classified companies on 100 repetitions)

According to Table 6.5, the accuracy of healthy companies is 96.87% and the accuracy of bankrupt companies is 89.59%. As we discussed in the previous subsection, the accuracy of bankrupt companies containing outliers of variable X16 is $139/139 := 100\%$, which is better than 89.59%. Therefore, one strategy of separating X16 as an independent classifier is applied here. The particular rule for this strategy is:

- First split the total samples into training and testing set, they occupy two thirds and one third of samples respectively.

- As to the training samples, separate them into two groups according to the outliers of variable X16. For the samples containing no X16 outliers, build the LOO-IELM model, while for the samples with abnormal X16 values, we predict all of them bankruptcy.

- Similar situation for the testing part, when the samples have strange values of X16, the final predicted results are bankruptcy; otherwise, the samples are tested with the model built in the second step.

With the special strategy, we call it Combo method in this dissertation,

the results are improved as shown in the Table 6.6.

		Predicted Classes	
		Health	Bankruptcy
Actual Classes	Health	59.87	2.13
	Bankruptcy	5.34	57.66

Table 6.6. Confusion matrix for the 9V of 2002, using Combo method. (Average number of classified companies on 100 repetitions)

For the Combo method, the accuracy of healthy companies is 96.56%, which remains nearly the same level with the normal LOO-IELM method (96.87%). However, on the other hand, the bankruptcy accuracy improved from 89.59% to 91.52%. Therefore, this compromise helps to increase the total accuracy 0.008%. Then the question becomes how to increase the bankrupt accuracy without sacrificing the healthy part. Next section, ensemble modeling is introduced to accomplish this aim.

6.5 Ensemble modeling

In 5.3.4, the idea of ensemble modeling was explained. Here we present more details on how the ensemble is done for this specific case.

6.5.1 Combining different models into ensembles

It is stated [8] that a particular method for creating an ensemble can be better than the best single model. Therefore, how to combine the models becomes an issue. There are several ways to achieve this. One example is using Non-Negative constrained Least-Squares (NNLS) algorithm [99, 54]. In this dissertation, we use the decision table for the ensemble. The decision table is made in accordance with the accuracy of different individual classifiers and its results are shown in the experience section.

Fig 6.2 illustrates the main procedures of the ensemble method, and how they interact.

Also take 9 variables of 2002 for example, Table 6.7 presents how we make the final decision whether the companies predicted health or bankruptcy. As we can see from the table, the final decision is the same as the results of Combo method except the last second line, when the Combo method indicates to be bankruptcy while the normal LOO-IELM points to be different. So if this situation occurs, the decision is corrected to be Health, instead of the bankruptcy from Combo method.

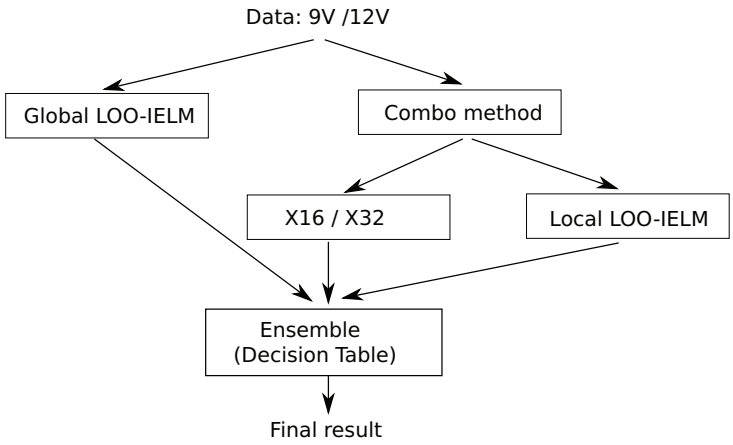


Figure 6.2. The framework of the Ensemble method

Global	Combo method		Final
LOO-IELM	X16 (abnormal)	Local LOO-IELM	Decision
B	B		B
B		B	B
B 89.59%		H 96.56%	H
H 89.59%	B 100%		B
H 96.87%		**B** 91.52%	**H**
H		H	H

Table 6.7. Decision Table of 9 Variables for 2002

According to the decision table, results are improved as shown in the following Table 6.8. The total result is 94.00% eventually.

		Predicted Classes	
		Health	Bankruptcy
Actual Classes	Health	60.99	1.01
	Bankruptcy	6.49	56.51

Table 6.8. Confusion matrix for the 9V of 2002, using Ensemble modeling. (Average number of classified companies on 100 repetitions)

6.5.2 Estimating the performance of the ensemble

The main idea in estimating the performance of the method is to divide the data set into training, validation and testing sets. The models are

built in the training phase based on the information that the training set contains. The results are validated and the best model chosen. Finally, the model is tested in a test set that is not used for building the model.

However, the LOO-IELM we proposed can automatically construct the neural network with incremental neurons and minimizes the Leave-One-Out error. When building the model with the training set, it combines validation set already. Therefore, in this dissertation, the data set is only divided into training and test set. More particularly, the data set is split randomly ten times with the same proportion to train and test LOO-IELM and the final results is the average of the 10 repetitions. In this way, we are able to obtain more general performance of the method, the computational time and the standard derivation is also recorded and illustrated in next Section.

6.6 Final Results

As we presented previously, Combo and Ensemble method work well on the 9 variables of year 2002. In this section, more results are listed and compared. Table 6.9 brings a summary on the two groups of data from both 2002 and 2003, using the proposed method.

The data sets have all been processed in the same way: for each data set, 10 different random permutations are taken without replacement; for each permutation, two thirds are taken for the training set, and the remaining third for the test set. Training sets are then normalized (zero-mean and unit variance) and test sets are also normalized using the very same normalization factors than for the corresponding training set. The results presented in the following are hence the average of the 10 repetitions for each data set.

To demonstrate the classification power of LOO-IELM and Ensemble method, a comparison is made with Ensembles of local linear models (E-LL), Linear Discriminant Analysis (LDA) and Extreme Learning Machine Support Vector Machines (ELM-SVM) which are shown in Publication IV. According to the experiment results in Publication IV on data 2002, the best method is E-LL with the classification accuracy 93.4%, followed by ELM-SVM method with the accuracy 93.2%. LDA is incompetent here as it can only classify 86.5% of the companies correctly. As to the data of 2003, the performances are in general less good than 2002. This tendency appears in nearly all the articles containing this data set, like [45, 60].

	LOO-IELM			Combo method			Ensemble method			LDA	E-LL	E-SVM
	H (%)	B (%)	Total (%)	H (%)	B (%)	Total (%)	H (%)	B (%)	Total (%)	Total (%)	Total (%)	Total (%)
2002_9V	96.87	89.59	93.23	96.56	91.52	94.04	98.39	88.89	93.84	86.5	93.4	93.2
2003_9V	84.09	80.32	82.21	82.35	82.95	82.65	93.85	76.92	85.39	82.4	81.9	83.0
2002_12V	87.89	82.67	85.28	88.18	87.98	88.08	88.18	87.98	88.08			
2003_12V	79.14	80.52	79.83	73.25	82.98	78.12	75.38	87.69	81.54			

Table 6.9. Results comparison

In spite of this, E-LL has the accuracy of 81.9% and ELM-SVM performs the best for 2003, with the accuracy 83%. The main drawback of E-LL is it is very time consuming. In any case, the proposed method in this dissertation is better than the others and the most significant advantages are its robustness and its fast speed.

6.7 Summary

According to the preknowledge of financial expertise, there is an interesting phenomenon that the companies containing the outliers of some specific variables tend to go bankrupt. Therefore, Combo method utilizes this phenomenon as well as the LOO-IELM model, so that it improves the classification results. Furthermore, an Ensemble method is also investigated. It ensembles from a LOO-IELM model and a Combo method trained with the same training set. The ensemble process is accomplished by decision table (in Section 4) and the entire algorithm performs a good prediction as shown in the experiments and it helps to interpret the model.

Moreover, the mentioned methods are tested on both two sets (9V and 12V) respectively. In general, the 9 variables (9V) selected by automatic black-box variable selection performs better than the 12 variables (12V) chosen by financial expertise. However, even though the choice of 12V couldn't compel conviction, the financial ratios, the intervals used in Combo method play an important role in this dissertation.

7. Conclusion

In this dissertation, we have addressed bankruptcy prediction as a classification problem. We use several Machine Learning methods as well as some financial expertise in order to improve the prediction accuracy. On the other hand, a practical problem, missing data issue, is also considered and solved in this dissertation.

According to the experimental results in Chapter 6, it can be noted that variable (ratio) selection is effective and necessary in terms of bankruptcy prediction. However, among numerous studies which have been devoted to the bankruptcy prediction models, only 23% have undertaken to explore variable selection [26]. This dissertation highlights to reduce the gap between modeling techniques and variable selection in order to improve the accuracy. Of course the variables selected will be diverse aiming at different background of data. In this dissertation, data used is focusing on small and medium enterprise (SME) in French retail sector. So strictly speaking, the financial ratios selected in this dissertation can be only guaranteed working in this specific area. Any other changes on the data, like different size of the company, different country, etc, will cause different ratios options. However, in general, the methods and the strategies proposed in this dissertation for variable selection is an automatic solution, like forward selection using OP-KNN (Publication I). If luckily some financial expertise is acquired, we also have Combo and Ensemble method (Publication VI) to use it for reference. On the other hand, based on the results in Chapter 6, the comparison between the ratios selected by a specific Machine Learning method (Publication IV) and the ones selected by some financial experts, the former performs a little better than the latter. This at least gives me a great motivation to continue working on Machine Learning methods for bankruptcy prediction in the future.

Among the methods proposed for bankruptcy prediction in this disserta-

tion, Ensemble K-nearest neighbors (EKNN) (Publication V), Ensembles of Local Linear Models (Publication IV), Delta test-ELM (DT-ELM) (Publication VII) and Leave-One-Out-Incremental Extreme Learning Machine (LOO-IELM) (Publication VI), ELM based methods in general give better performance. DT-ELM uses Bayesian Information Criterion (BIC) to restrict the search as well as to consider the size of the network and Delta Test (DT) to further prevent overfiting. LOO-IELM operates in an incremental way to avoid inefficient and unnecessary calculations and stops automatically with the neurons of which the number is unknown. Especially LOO-IELM was used with the combination of financial expertise for better prediction. This reveals the great potential of the combination with Machine Learning methods and financial preknowledge.

As to the Missing data problem, it is commonly confronted when collecting the companies' accounting data. Instead of bringing errors into the model by imputation, this dissertation presents two algorithms estimating distances between incomplete samples. These two algorithms could be combined with various distance based Machine Learning methods. This dissertation shows two examples how to predict bankruptcy with incomplete data. One is to use Ensemble Nearest Neighbors (ENN) to solve bankruptcy prediction problem and meanwhile using an adapted distance metric which can be used directly for incomplete data (Publication VIII). Another one is to estimate the expected pairwise distances between samples directly on incomplete data and to use TROP-ELM [70] to regularize the matrix computations (Publication II). These tools for missing data problem make our methods more practical for real world data.

In the future, if the work achieved is to be pursued, there are several directions to explore, to possibly improve the results and the performance, as well as to widen the view and approach. First of all, new data are precious and longing to verify the proposed methods, which could further improve the methods, even though the fact is financial data are mainly costly and partially confidential. There is no doubt that it is extremely important to carefully select the dataset for experiments. Secondly, it is still questionable that how to select the most important ratios and reduce the dimension redundancy. The author will continue the way on variable selection (VS) because she believes VS can improve the prediction performance. Thirdly, like done in this dissertation, both Machine Learning methods and financial expertise are used together for prediction, this kind of combination will be improved and developed more in the future

work. In addition, soft classification techniques (classifier ensembles and hybrid classifiers.) appear to be another direction for future research as core techniques that are used in the development of prediction models. Finally, the author is willing to broaden the prediction horizon to long-term (+2 years). In fact, most of the studies predict bankruptcy in a period of 1 or 2 years, which is hard to reference when banks have to make a long term loan. Thus, in the future, new research will turn to improve of the ability of long-term forecast. In this framework, variable selection will still be able to improve the prediction capacity [47].

Bibliography

[1] *URL http://archive.ics.uci.edu/ml/datasets.html*.

[2] D. Sovilj M. G. Arenas L. J. Herrera H. Pomares I. Rojas A. Guillén, M. van Heeswijk. Variable selection in a gpu cluster using delta test. In *International Work Conference on Artificial Neural Networks (IWANN)*.

[3] H. Akaike. Information theory and an extension of the maximum likelihood principle. In *Second International Symposium on Information Theory*, pages 267–281, Budapest, 1973.

[4] E. Altman, N. Fargher, and E. Kalotay. A simple empirical model of equity-implied probabilities of default. *Journal of Fixed Income*, 20(3):71–85, 2011.

[5] E. I. Altman. Financial ratios, discriminant analysisi and the prediction of corporation bankruptcy. *The Journal of Finance*, 23:589–609, 1968.

[6] E. I. Altman. Financial ratios, discriminant analysisi and the prediction of corporation bankruptcy. *The Journal of Finance*, 23:589–609, 1968.

[7] E. I. Altman. A yield premium model for the high-yield debt market. *Financial Analysts Journal*, 1:49–56, 1995.

[8] S. V. Barai and Y. Reich. Ensemble modelling or selecting the best model: Many could be better than one. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, pages 377–386, 1999.

[9] M. S. Bartlett, G. Littlewort, M. Frank, C. Lainscsek, I. Fasel, and J. Movellan. Recognizing facial expression: machine learning and application to spontaneous behavior. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

[10] W. H. Beaver. Financial ratios as predictors of failure. *Journal of Accounting Research*, 4:71–111, 1966.

[11] J. Bellovary, D. Giacomino, and M. Akers. A review of bankruptcy prediction studies: 1930 to present. *Journal of Financial Education*, 33:87–114, 2007.

[12] G. Bontempi, M. Birattari, and H. Bersini. Recursive lazy learning for modeling and control. In *European Conference on Machine learning*, pages 292–303, 1998.

[13] J. E. Boritz and D. B. Kennedy. Effectiveness of neural network types for prediction of business failure. *Expert Systems with Applications*, 9(4):95–112, 1995.

[14] S. M. Bryant. A case-based reasoning approach to bankruptcy prediction modeling. *Intelligent Systems in Accounting, Finance and Management*, 6(3):195–214, 1997.

[15] S. Canbas, A. Cabuk, and S. B. Kilic. Prediction of commercial bank failure via multivariate statistical analysis of financial structure: The turkish case. *European Journal of Operational Research*, 1:528–546, 2005.

[16] J. Cao, Z. Lin, G.-B. Huang, and N. Liu. Voting based extreme learning machine. *Information Sciences*, 185(1):66–77, 2012.

[17] J. Chen and J. Shao. Nearest neighbor imputation for survey data. *Journal of Official Statistics*, 16(2):113–131, 2000.

[18] A. Cielen, L. Peeters, and K. Vanhoof. Bankruptcy prediction using a data envelopment analysis. *European Journal of Operational Research*, 154(2):526–532, 2004.

[19] J. Cohen and P. Cohen. Applied multiple regression/correlation analysis for the behavioral sciences (2nd ed.). pages 185–207, 2003.

[20] D. Coomans and D. L. Massart. Alternative k-nearest neighbour rules in supervised pattern recognition : Part 1. k-nearest neighbour classification by using alternative voting rules. *Analytica Chimica Acta*, 136:15–27, 1982.

[21] C. Cortes and V. N. Vapnik. Support-vector networks. *Machine Learning*, 20:15–21, 1995.

[22] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press.

[23] J. de Andrés, M. Landajo, and P. Lorca. Bankruptcy prediction models based on multinorm analysis: An alternative to accounting ratios. *Knowledge-Based Systems*, 30:67–77, 2012.

[24] E. B. Deakin. A discriminant analysis of predictors of business failure. *Journal of Accounting Research*, 10(1):167–179, 1972.

[25] A. I. Dimitras, R. Slowinski, R. Susmaga, and C. Zopounidis. Business failure prediction using rough sets. *European Journal of Operational Research*, 114(2):263–280, 1999.

[26] P. du Jardin. Bankruptcy prediction models: How to choose the most relevant variables? *Bankers, Markets & Investors*, (98):39–46, 2009.

[27] R. G. Haldeman E. Altman and P. Narayanan. Zeta analysis: A new model to identify bankruptcy risk of corporations. *Journal of Banking and Finance*, 1:29–35, 1977.

[28] Robert O. Edmister. An empirical test of financial ratio analysis for small business failure prediction. *The Journal of Financial and Quantitative Analysis*, 7(2):1477–1493, 1972.

[29] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32(2):407–499, 2004.

[30] E. Eirola, G. Doquire, M. Verleysen, and A. Lendasse. Distance estimation in numerical data sets with missing values. *Information Science, accepted for publication*, 2013.

[31] G. Feng, G. B. Huang, Q. Lin, and R. Gay. Error minimized extreme learning machine with growth of hidden nodes and incremental learning. *IEEE Transactions on Neural Networks*, 20(8):1352–1357, 2009.

[32] Guorui Feng, Guang-Bin Huang, Qingping Lin, and Robert Gay. Error minimized extreme learning machine with growth of hidden nodes and incremental learning. *IEEE Transactions on Neural Networks*, 20(8):1352–1357, 2009.

[33] B. L. Ford. An overview of hot-deck procedures, in: Incomplete data in sample surveys. In *Academic Press*, pages 185–207, New York, USA, 1983.

[34] H. Frydman, E.I. Altman, and D. Kao. Introducing recursive partitioning for financial classification: The case of financial distress. *Journal of Finance*, 40(1):269–291, 1985.

[35] G. H. Golub, M. Heath, and G.Wahba. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21(2):215–223, 1979.

[36] D. F. Heitjan and S. Basu. Distinguishing "missing at random" and "missing completely at random". *The American Statistician*, 50(3):207–213, 1996.

[37] D. A. Hensher and S. Jones. Forecasting corporate bankruptcy: Optimizing the performance of the mixed logit model. *Abacus*, 43(3):241–364, 2007.

[38] A. E. Hoerl and R. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[39] G. B. Huang, Q.-Y. Zhu, and C.-K. Siew. Extreme learning machine: Theory and applications. In *2004 International joint conference on Neural Networks (IJCNN'2004)*.

[40] G. B. Huang, Q.Y. Zhu, and C. K. Siew. Extreme learning machine: Theory and applications. *Neurocomputing*, 70(1), 2006.

[41] Guang-Bin Huang and Lei Chen. Convex incremental extreme learning machine. *Neurocomputing*, 70:3056–3062, 2007.

[42] Guang-Bin Huang and Lei Chen. Enhanced random search based incremental extreme learning machine. *Neurocomputing*, 71:3460–3468, 2008.

[43] Guang-Bin Huang, Ming-Bin Li, Lei Chen, and Chee-Kheong Siew. Incremental extreme learning machine with fully complex hidden nodes. *Neurocomputing*, 71:576–583, 2008.

[44] J. Van Hulse and T. M. Khoshgoftaar. Incomplete-case nearest neighbor imputation in software measurement data. *Information Sciences*, 1, 2011.

[45] P. Du Jardin. Prévision de la défaillance et réseaux de neurones: léapport des méthodes numériques de sélection de variables. *PhD thesis, Université de Nice-Sophia-Antipolis*, pages 58–63.

[46] P. Du Jardin and E. Séverin. Dynamic analysis of the business failure process: a study of bankruptcy trajectories. In *Portuguese Finance Network Conference*.

[47] P. Du Jardin and E. Séverin. Forecasting financial failure using a kohonen map: A comparative study to improve model stability over time. *European Journal of Operational Research*, 221(2):378–396, 2012.

[48] A. J Jones. New tools in non-linear modeling and prediction. *Computational Management Science*, 1(2):109–149, 2004.

[49] G. V. Karels and A. J. Prakash. Multivariate normality and forecasting of business bankruptcy. *Journal of Business Finance and Accounting*, 1:573–593, 1987.

[50] S. Kaski, J. Sinkkonen, and J. Peltonen. Bankruptcy analysis with self-organizing maps in learning metrics. *IEEE Transactions on Neural Networks*, 12(4):936–947, 2001.

[51] K. Kiviluoto. Predicting bankruptcies with the self-organizing map. *Neurocomputing*, 21(1):191–201, 1998.

[52] R. C. Lacher, P. K. Coats, S. C. Sharma, and L.F. Fant. A neural network for classifying the financial health of a firm. *European Journal of Operational Research*, 85:53–65, 1995.

[53] Y. Lan, Y.Chai Soh, and G. B. Huang. Constructive hidden nodes selection of extreme learning machine for regression. *Neurocomputing*, 73(16), 2010.

[54] C. L. Lawson and R. J. Hanson. Solving least squares problems. *CLassics in Applied Mathematics*, pages 245–286, 1995.

[55] L.-F. Lee. Fully recursive probability models and multivariate log-linear probability models for the analysis of qualitative data. *Journal of Econometrics*, 16(1):51–69, 1981.

[56] Hui Li, Young-Chan Lee, Yan-Chun Zhou, and Jie Sun. The random subspace binary logit (rsbl) model for bankruptcy prediction. *Knowledge-Based System*, 1:1380–1388, 2011.

[57] Ming-Yuan Leon Li and Peter Miu. A hybrid bankruptcy prediction model with dynamic loadings on accounting-ratio-based and market-based information: A binary quantile regression approach. *Empirical Finance*, 17:818–833, 2010.

[58] R. J. A. Little and D. B. Rubin. Statistical analysis with missing data (second ed.). pages 138–149, Wiley, NJ, USA, 2002.

[59] H. Liu, S. Shah, and W. Jiang. On-line outlier detection and data cleaning. *Computers and Chemical Engineering*, 28:CSIRO Technical Report CMIS–02/102, 2004.

[60] A. Lourme and C. Biernacki. Simultaneous t-model-based clustering for data differing over time period: Application for understanding companies financial health. *Case Studies in Business, Industry and Government Statistics (CSBIGS)*, 4(2):73–82, 2011.

[61] G. Lu and J. Copas. Missing at random, likelihood ignorability and model completeness. *Annals of Statistics*, 32(2):754–765, 2004.

[62] M. L. Marais, J. Patel, and M. Wolfson. The experimental design of classification models: An application of recursive partitioning and bootstrapping to commercial bank loan classifications. *Journal of Accounting Research*, 22:87–114, 1984.

[63] D. Martin. Early warning of banking failure. *Journal of Banking and Finance*, 7:249–276, 1977.

[64] MathWorks. Matlab financial toolbox: ecmnmle. *URL http://www.mathworks.com/help/toolbox/finance/ecmnmle.html*, 2010.

[65] T. E. Mckee. Developing a bankruptcy prediction model via rough sets theory. *Intelligent Systems in Accounting, Finance and Management*, 9(3):159–173, 2000.

[66] T. E. Mckee. Rough sets bankruptcy prediction models versus auditor signaling rates. *Journal of Forecasting*, 22(8):569–586, 2003.

[67] X. Meng and D. B. Rubin. Maximum likelihood estimation via the ecm algorithm: A general framework. *Biometrika*, 80(2):267–278, 1993.

[68] Y. Miche, E. Eirola, P. Bas, O. Simula, C. Jutten, A. Lendasse, and M. Verleysen. Ensemble modeling with a constrained linear system of leave-one-out outputs. In *In proceedings of 18th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, April.

[69] Y. Miche, A. Sorjamaa, and A. Lendasse. Op-elm: Theory, experiments and a toolbox. In *Artificial Neural Networks - ICANN 2008*.

[70] Y. Miche, M. van Heeswijk, P. Bas, O. Simula, and A. Lendasse. Trop-elm: a double-regularized elm using lars and tikhonov regularization. *Neurocomputing*, 74(16):2413–2421, 2011.

[71] J. H. Min and Y. C. Lee. Bankruptcy prediction using support vector machine with optimal choice of kernel function parameters. *Expert Systems with Applications*, 28(4):603–614, 2005.

[72] R. H. Myers. *Classical and modern regression with applications*. Boston: PWS-KENT, 1990.

[73] J. A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.

[74] T. E. Nichols and A. P. Holmes. Nonparametric permutation tests for functional neuroimaging: A primer with examples. *Human Brain Mapping*, 15(1):1–25, 2001.

[75] James A. Ohlson. Financial ratios and the probabilistic prediction of bankruptcy. *Journal of Accounting Research*, 18(1):109–131, 1980.

[76] D. L. Olson and D. Delen. Advanced data mining techniques. *The American Statistician*, page 138, 2008. ISBN:3-540-76916-1.

[77] Cheol-Soo Park and Ingoo Han. A case-based reasoning with the feature weights derived by analytic hierarchy process for bankruptcy prediction. *Expert Systems with Applications*, 23:255–264, 2002.

[78] L. V. Philosophov, J. A. Batten, and V. L. Philosophov. Predicting the event and time horizon of bankruptcy using financial ratios and the maturity schedule of long-term debt. *EFA 2005 Moscow Meetings Paper*, 2007.

[79] Maurice Bertram Priestley. Spectral analysis and time series. *Academic Press*, 1, 1981.

[80] C. Radhakrishna Rao and Sujit Kumar Mitra. Generalized inverse of matrices and its applications. *New York: John Wiley & Sons*, page 240, 1971.

[81] H. J. Rong, Y. S. Ong, A. H. Tan, and Z. Zhu. A fast pruned-extreme learning machine for classification problem. *Neurocomputing*, 72:359–366, 2008.

[82] J. L. Schafer. Multiple imputation: a primer. *Statistical Methods in Medical Research*, 8(1):3–15, 1999.

[83] F. Scheuren. Multiple imputation: How it began and continues. *The American Statistician*, 59:315–319, 2005.

[84] G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.

[85] J. Sexton and A. R. Swensen. Ecm algorithms that converge at the rate of em. *Biometrika*, 87(3):651–662, 2011.

[86] K. S. Shin and Y. J. Lee. A genetic algorithm application in bankruptcy prediction modeling. *Expert Systems with Applications*, 23:312–328, 2002.

[87] T. Shumway. Forecasting bankruptcy more accurately: A simple hazard model. *Journal of Business*, 1:573–593, 1987.

[88] T. Similä and J. Tikka. Multiresponse sparse regression with application to multidimensional scaling. In *Artificial Neural Networks: Formal Models and Their Applications-ICANN*, volume 3697, pages 97–102, 2005.

[89] A. Sorjamaa, A. Lendasse, and M. Verleysen. Pruned lazy learning models for time series prediction. In *In Proceedings of ESANN 2005*, pages 509–514, 2005.

[90] A. Stefánsson, N. Koncar, and A. J. Jones. A note on the gamma test. *Neural Computing and Applications*, 5(3):131–133, 1997.

[91] J. Sun and H. Li. Dynamic financial distress prediction using instance selection for the disposal of concept drift. *Expert Systems with Applications*, 38(3):2566–2576, 2011.

[92] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, 58(1):267–288, 1996.

[93] F. Varetto. A genetic algorithm application in bankruptcy prediction modeling. *Journal of Banking and Finance*, 22(10):1421–1439, 1998.

[94] M. Verleysen. Learning high-dimensional data. In *NATO Advanced Research Workshop on Limitations and Future Trends in Neural Computing*, pages 22–24, Italy, 2001.

[95] R. C. West. A factor analytic approach to bank condition. *Journal of Banking and Finance*, 9:253–266, 1985.

[96] G. J. Williams, R. A. Baxter, H. X. He, S. Hawkins, and L. Gu. A comparative study of rnn for outlier detection in data mining. In *IEEE International Conference on Data-mining (ICDM'02)*, pages 185–207, Maebashi City, Japan, 2002.

[97] Z. R. Yang, M. B. Platt, and H. D. Platt. Probabilistic neural networks in bankruptcy prediction. *Journal of Business Research*, 44(2):67–74, 1999.

[98] S. L. Yong, M. Hagenbuchner, and A. C. Tsoi. Ranking web pages using machine learning approaches. In *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*.

[99] Qi Yu, Amaury Lendasse, and Eric Séverin. Ensemble knns for bankruptcy prediction. In *in CEF 09, 15th International Conference: Computing in Economics and Finance*, Sydney, Austrilia, 2009.

[100] L. Yuan, S. Y.Chai, and G. B. Huang. Random search enhancement of error minimized extreme learning machine. In *European Symposium on Artificial Neural Networks (ESANN) 2010*, pages 327–332, Bruges, Belgium, 2010.

[101] H. J. Zimmermann. Fuzzy set theory and its applications. *Kluwer Academic Publishers*, pages 298–319, 1996.

[102] M. E. Zmijewski. Methodological issues related to the estimation of financial distress prediction models. *Journal of Accounting Research*, 22:59–82, 1984.

DISSERTATIONS IN INFORMATION AND COMPUTER SCIENCE

Aalto-DD117/2012 Huopaniemi, Ilkka
Multivariate Multi-Way Modelling of Multiple High-Dimensional Data Sources. 2012.

Aalto-DD137/2012 Paukkeri, Mari-Sanna
Language- and domain-independent text mining. 2012.

Aalto-DD133/2012 Ahlroth, Lauri
Online Algorithms in Resource Management and Constraint Satisfaction. 2012.

Aalto-DD158/2012 Virpioja, Sami
Learning Constructions of Natural Language: Statistical Models and Evaluations. 2012.

Aalto-DD20/2013 Pajarinen, Joni
Planning under uncertainty for large-scale problems with applications to wireless networking. 2013.

Aalto-DD29/2013 Hakala, Risto
Results on Linear Models in Cryptography. 2013.

Aalto-DD44/2013 Pylkkönen, Janne
Towards Efficient and Robust Automatic Speech Recognition: Decoding Techniques and Discriminative Training. 2013.

Aalto-DD47/2013 Reyhani, Nima
Studies on Kernel Learning and Independent Component Analysis. 2013.

Aalto-DD70/2013 Ylipaavalniemi, Jarkko
Data-driven analysis for natural studies in funtional brain imaging. 2013.

Aalto-DD61/2013 Kandemir, Melih
Learning Mental States from Biosignals. 2013.

Bankruptcy prediction is of a great concern to investors, creditors, borrowing firms and governments. For example, banks need to predict the possibility of default of a potential counterpart before they extend a loan. This could lead to wiser lending decisions, and therefore result in significant savings. Bankruptcy can happen to any organizations because the business environment is increasingly undergoing uncertainty and competition these days. Especially due to the recent changes in the world economy and as more firms, large and small, seem to fail now more than ever. The prediction of the bankruptcy, is then of increasing importance. This dissertation offers a way to predict the bankruptcy possibilities which hopefully could contribute in the real world.

Aalto-DD 90/2013

ISBN 978-952-60-5186-4
ISBN 978-952-60-5187-1 (pdf)
ISSN-L 1799-4934
ISSN 1799-4934
ISSN 1799-4942 (pdf)

Aalto University
School of Science
Department of Information and Computer Science
www.aalto.fi

BUSINESS + ECONOMY

ART + DESIGN + ARCHITECTURE

SCIENCE + TECHNOLOGY

CROSSOVER

DOCTORAL DISSERTATIONS