이학석사 학위논문

Credit Rating Prediction by Using Machine Learning Methods

(기계 학습을 이용한 신용 등급 예측)

2014년 8월

서울대학교 대학원

수리과학부

김 도 영

Credit Rating Prediction by Using Machine Learning Methods

(기계 학습을 이용한 신용 등급 예측)

지도교수 최 형 인

이 논문을 이학석사 학위논문으로 제출함

2014년 4월

서울대학교 대학원

수리과학부

김 도 영

김 도 영의 이학석사 학위논문을 인준함

2014년 6월

위 원 장 _____ (인)

부 위 원 장 _____ (인)

위 원 _____ (인)

Credit Rating Prediction by Using Machine Learning Methods

A dissertation

submitted in partial fulfillment

of the requirements for the degree of

Master of Science

to the faculty of the Graduate School of
Seoul National University

by

Do-young Kim

Dissertation Director : Professor Hyeong In Choi

Department of Mathematical Sciences
Seoul National University

August 2014

Abstract

In this thesis, we discuss the credit ratings of companies. Our purpose is to make a credit rating prediction rule that gives each company a credit which is as correct as possible to the actual rank. We describe three representative machine learning algorithms, which are ordinal logistic regression, neural networks and support vector machine. In addition, we try to analyze their performance and correctness and compare them to determine which method is the most efficient in machine learning to decide ratings. We deal with two different data sets of experiments which consist of true credit rating of companies in 2009 and 2013 and financial information in the previous year.

Key words: credit rating, machine learning, logistic regression, neural networks, support vector machine

Student Number: 2011-23201

Contents

Abstract **i**

1 Introduction **1**

2 Machine learning methods **3**

2.1 Ordinal logistic regression 3

2.1.1 Ordinal regression 6

2.2 Neural networks 7

2.3 Support vector machine 10

3 Data and experiment **14**

3.1 Data and features 14

3.2 Experment 16

4 Results **18**

5 Conclusion **24**

Abstract (in Korean) **27**

Acknowledgement (in Korean) **28**

Chapter 1

Introduction

Credit ratings have been widely used by bond investors, debt issuers, and governmental officials as a measure of riskiness of the companies and bonds. They are important determinants of risk premiums and the marketability of bonds[1]. Credit ratings are determined by rating agencies such as Standard & Poor' s, Moody' s and Fitch all over the world. In case of Korea, there are three representative rating companies, which are KIS, Korea Ratings and NICE. They invest great amount of time and human resources to perform deep and accurate analysis of the company' s credit risk based on various aspects ranging from strategic competitiveness to operational details[1]. Accordingly, we need to pay high cost to get ratings from professional rating company. Also, since ratings are estimated yearly, they are not updated frequently. Even though subjective judgment affects a lot to decide ratings, we try to predict a credit rating based on financial statements and ratios that are relatively easy to access.

In the past, credit rating prediction was researched by using statistical

methods such as Ordinary Least Squares(OLS) and Multiple Discriminant Analysis(MDA)[4][5][6][7]. In the recent years, many people are focusing on machine learning techniques, which are Neural Networks, Support vector machine and Expectation Maximization algorithms etc[8][9][10]. Machine learning aims for figuring out the pattern that data have commonly and constructing a new predicting structure based on the data. Most of researches have the conclusion that machine learning methods performed better than conventional statistical methods.

In this thesis, we find the rule that forecast credit ratings by using machine learning methods.

In Chapter 2, we introduce typical machine learning methods : logistic regression, neural networks and support vector machine. We present how they operate to train and classify the data.

In Chapter 3, we deal with data, features and experimental procedures.

In Chapter 4, we show the results and detailed confusion matrices of each method.

In Chapter 5, we discuss conclusions.

Chapter 2

Machine learning methods

In chapter 2, we introduce representative machine learning methods. Machine learning methods basically divide into unsupervised learning and supervised learning. Unsupervised learning deals with problem related to unlabeled data. It is used for clustering, hidden markov models and dimensionality reduction etc. On the other hand, supervised learning is a classifier based on labeled data. It trains labeled data, figures out pattern and make classifying structures. We present three supervised learning methods : ordinal logistic regression, neural networks and support vector machine.

2.1 Ordinal logistic regression

Logistic regression is a conventional binary classifier when dependent variables can be representable as discrete response such as 0 or 1. We call independent variables features.

We have the following situation.

- Input data $\mathbf{X} = (\mathbf{x}(0), \mathbf{x}(1), \mathbf{x}(2), \mathbf{x}(3), \cdots, \mathbf{x}(d))$ where d is the number of features, $\mathbf{x}(i)$ is a N-dimensional vector for each i and $\mathbf{x}(0) = (1, 1, 1, \cdots, 1)^T$
- Output data $y = (y_1, y_2, y_3, \cdots, y_N)$ where N is the number of data and $y_i \in \{0, 1\}$

Let $\mathbf{x}_i$ denote the i-th row vector of the matrix $\mathbf{X}$ and $\mathscr{D} = \{\mathbf{x}_i, y_i\}_{i=1,2,\cdots,N}$ be a data set. We train a machine in order to construct a predicting rule by inserting $\mathscr{D}$ into it . First of all, we introduce a sigmoid function $g(t)$ that is often used for machine learning algorithms.

$$g(t) = \frac{1}{1 + e^{-t}}$$

The best thing of a sigmoid function is that it satisfies the equation $g'(t) = g(t)(1 - g(t))$. This will be useful to derive optimization problem for logistic regression.

We assume for model that there exists $\omega = (\omega_0, \omega_1, \cdots, \omega_d) \in \mathbb{R}^{d+1}$ and a function $h_\omega(\mathbf{x}_i) = g(\omega^T \mathbf{x}_i)$ such that $P(y_i = 1|\mathbf{x}_i) = h_\omega(\mathbf{x}_i)$. Then, we define a prediction rule for given each $\mathbf{x}_i$ as follows.

$$\text{The predicted } y_i^P = = \begin{cases} 0 & \text{if } h_\omega(\mathbf{x}_i) < 0.5 \\ 1 & \text{if } h_\omega(\mathbf{x}_i) \geq 0.5 \end{cases}$$

Since y_i is a binary variable, $P(y_i = 0|\mathbf{x}_i) = 1 - h_\omega(\mathbf{x}_i)$. In that case, $P(y|\mathbf{x})$ follows Bernollui distribution as $P(y|\mathbf{x}) = h_\omega(\mathbf{x})^y (1 - h_\omega(\mathbf{x}))^{1-y}$. Let us derive a cost function for the logistic regression from the log-likelihood function of $P(y|\mathbf{x}_i)$. The log-likelihood function $l(\omega)$ of $P(y|\mathbf{x}_i)$ is

$$l(\omega) = \sum_{i=1}^{N} \{y_i \log(h_\omega(\mathbf{x}_i)) + (1 - y_i) \log(1 - h_\omega(\mathbf{x}_i))\}$$

Then, we define a cost function $J(\omega) = -l(\omega)$. To minimize a cost function $J(\omega)$, which means to maximize the log-likelihood function $l(\omega)$, we have to choose the appropriate parameter ω. In the first place, let us check the uniqueness of the minimum of $J(\omega)$. We show the convexity of $J(\omega)$ for all ω.

$$\begin{aligned} \frac{\partial^2 J(\omega)}{\partial \omega_k \partial \omega_r} &= \sum_i g'(\omega^T \mathbf{x}_i) x_{ir} x_{ik} \\ &= (\mathbf{X}^T D \mathbf{X})_{kr} \end{aligned} \tag{2.1.1}$$

where D is a diagonal matrix with diagonal element $d_i = g'(\omega^T \mathbf{x}_i)$ and x_{ik} is element of $\mathbf{X}$. Then, the Hessian matrix of $J(\omega)$ is $\mathbf{X}^T D \mathbf{X}$. Since the sigmoid function $g(t)$ is strictly increasing , $g'(t) > 0$. Therefore, for any $v \in \mathbb{R}^{d+1}, v^T \mathbf{X}^T D \mathbf{X} v = \sum_i d_i (\mathbf{X}v)_i^{\,2} > 0$. Accordingly, $\mathbf{X}^T D \mathbf{X}$ is a positive semi-definite, which means that the second derivative term of the cost function $J(\omega)$ is always positive. In this way, the cost function $J(\omega)$ is convex for all ω, which means it has an unique minimum.

Let us find ω that minimizes $J(\omega)$. For each $k \in \{0, 1, 2, \cdots, d\}$,

$$\begin{aligned} \frac{\partial J(\omega)}{\partial \omega_k} &= \sum_{i=1}^{N} x_{ik} \{ y_i (1 - g(\omega^T \mathbf{x}_i)) - (1 - y_i) g(\omega^T \mathbf{x}_i) \} \\ &= \sum_{i=1}^{N} x_{ik} \{ y_i - g(\omega^T \mathbf{x}_i) \} \end{aligned} \tag{2.1.2}$$

We should get a parameter ω which makes $\dfrac{\partial J(\omega)}{\partial \omega_k} = 0$ for each k.

We use a gradient descent algorithm to solve this optimization problems. Gradient descent algorithm is to find proper ω changing the parameters, selecting the gradient and updating parameters. We denote $\omega_j^{(o)}$ old parameter

and $\omega_j^{(n)}$ new parameter induced by gradient descent algorithms. Then, for each k,

$$\begin{aligned} \omega_j^{(n)} &= \omega_j^{(o)} - \alpha \frac{\partial J(\omega)}{\partial \omega_j} \\ &= \omega_j^{(o)} - \alpha \sum_{i=1}^{N} \{y_i - h_{\omega^{(o)}}(\mathbf{x}_i)\} x_{ij} \end{aligned} \tag{2.1.3}$$

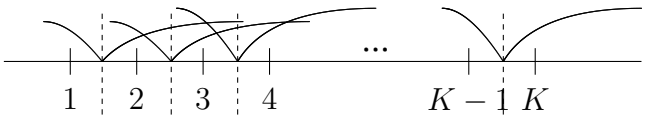
where α is a small real number enough to converge.

Note that we should update simultaneously for every k.

2.1.1 Ordinal regression

When we classify credit ratings, it is important to make parallel hyperplanes since there are orders. To solve this, we assume that given data make us derive parallel hyperplanes.

Suppose that we have a K-class ordinal regression problem. The idea to solve the ordinal regression problem is to break it into a set of binary regression problems. Let $\mathscr{D} = \{\mathbf{x}_i, y_i\}_{i=1,2,\cdots,N}$ be a data set. we define new data set $D(l)$ for each $l = 1, 2, 3, \cdots, K-1$ as follows.



$$D(l) = \{(\mathbf{x}_i, z_i^l)\}_{i=1,2,\cdots,n} \quad \text{where } z_i^l = \begin{cases} 0 & \text{if } y_i = 1, 2, \cdots, l \\ 1 & \text{if } y_i = (l+1), \cdots, K \end{cases} \tag{2.1.4}$$

For each binary regression problem, we can apply logistic regression algorithm introduced. Let $\omega = (\alpha_l, \beta)$ be a parameter that we find where $\alpha_l \in \mathbb{R}$ and $\beta = (\beta_1, \beta_2, \cdots, \beta_d) \in \mathbb{R}^d$. We need only one optimization problem so that we get the same β. Therefore, combined optimization problem of ordinal regression is to minimize the cost function given by

$$\sum_{l=1}^{k-1} \sum_{i=1}^{N} \{z_i^l \log g(\alpha_l + \beta^T \mathbf{x}_i) + (1 - z_i^l) \log(1 - g(\alpha_l + \beta^T \mathbf{x}_i)\} \tag{2.1.5}$$

where $g(t)$ is a sigmoid function.

We also use the gradient decent algorithm introduced before to get the appropriate ω.

2.2 Neural networks

In this section, we introduce neural networks. They are learning models motivated by basic cell of humans' nervous system, neuron, which is able to learn and recognize pattern. Neural networks are mainly used for highly non-linear data. A logistic unit of neural networks is a *perceptron*. See figure2.1.

For $\mathbf{x} = (x_1, x_2, \cdots, x_d) \in \mathbb{R}^d$, $\mathbf{a}$ is a linear combination of the inputs with $x_0 = 1$ where the coefficients of the linear combination are proper parameters. The output z is $g(\mathbf{a})$ where g is *activation function*. Tangent hyperbolic or sigmoid function are mainly used for activation function.

Neural networks comprises a number of layers which consist of several perceptrons. The first and last layer are called input layer and output layer, respectively. The layers in the middle are hidden layers. See figure2.2.

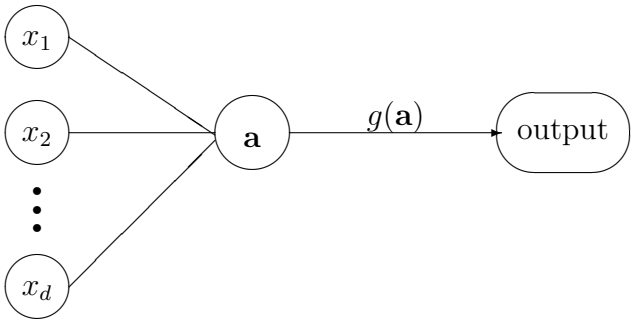


Figure 2.1: Perceptron

Let $\mathscr{D} = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1,2,\cdots,N}$ be a data set. For multiclass neural networks, $\mathbf{y}_i$ represents as a vector. For instance, $\mathbf{y}_i$ can be one of $(0,0,1)^T, (0,1,0)^T$ and $(0,0,1)^T$ when we have 3 classes. For j-th unit of l-th layer $\mathbf{a}_j^{(l)}$,

$$\mathbf{z}_j^{(l)} = g(\mathbf{a}_j^{(l)}) \tag{2.2.1}$$

$$\mathbf{a}_j^{(l+1)} = \sum_i \omega_{ij}^{(l)} \mathbf{z}_j^{(l)} \tag{2.2.2}$$

where $\omega_{ij}^{(l)} z_j^{(l)}$ is a weighted sum of units in l-th layer and g is an activation function.

Determining $\omega_{ij}^{(l)}$ for each layer plays an important role to make powerful neural networks. Our approach to solve this problem is to minimize sum of square errors. We define the cost function $J(\omega)$ as follows.

$$J(\omega) = \frac{1}{2} \sum_{i=1}^{N} ||\mathbf{y}_i^P - \mathbf{y}_i||^2$$

where $\mathbf{y}_i^P$ is a predicted K-dimensional output vector coming through the

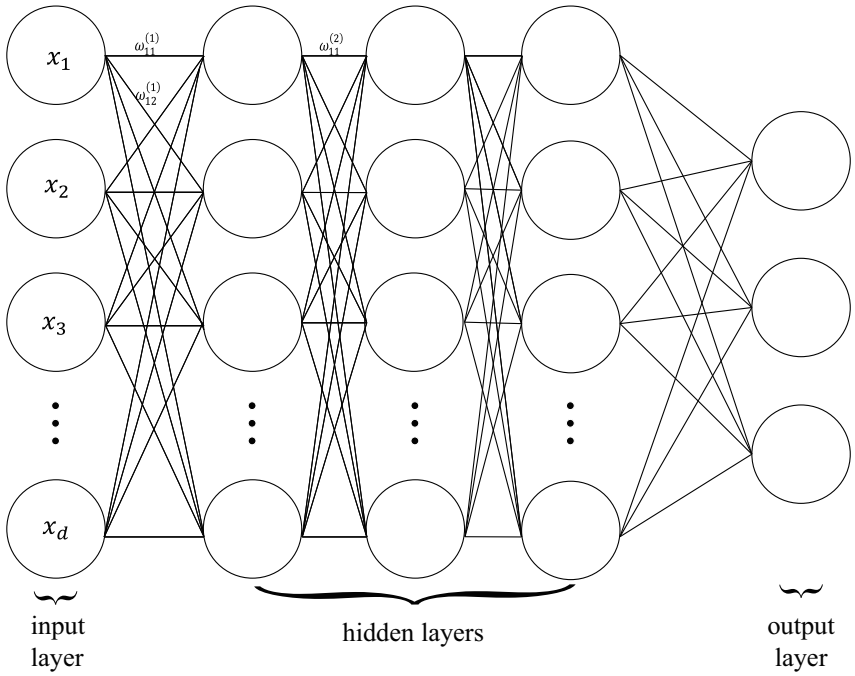


Figure 2.2: Structure of nerual networks

neural networks.

We need to find parameters ω that minimizes $J(\omega)$. To calculate derivatives of $J(\omega)$, we introduce backpropagation algorithms. It is an effective technique for evaluating the gradient. For a data point $(\mathbf{x}_n, \mathbf{y}_n)$,

$$J_n(\omega) = ||\mathbf{y}_n^P - \mathbf{y}_n||^2 = \frac{1}{2}\sum_{k=1}^{K}(\mathbf{y}_{nk}^P - \mathbf{y}_{nk})^2$$

where $\mathbf{y}_{nk}$ is k-th element of vector $\mathbf{y}_n$.

By chain rule,

$$\frac{\partial J_i(\omega)}{\partial \omega_{ij}^{(l)}} = \frac{\partial J_i(\omega)}{\partial \mathbf{a}_j^{(l+1)}} \cdot \frac{\partial \mathbf{a}_j^{(l+1)}}{\partial \omega_{ij}^{(l)}} \tag{2.2.3}$$

where $\omega_{ij}^{(l)}$ is a weight used for from i-th unit in the l-the layer to the j-th unit in the next layer.

Let us denote $\dfrac{\partial J_n(\omega)}{\partial \mathbf{a}_j^{(l+1)}}$ by $\delta_j^{(l+1)}$. By (2.2.1) and (2.2.3),

$$\frac{\partial J_i(\omega)}{\partial \omega_{ij}^{(l)}} = \delta_j^{(l+1)} \mathbf{z}_{ij}^{(l+1)}$$

At the first onset, for the output layer, we have

$$\delta_j^{(L)} = \mathbf{y}_{nj}^P - \mathbf{y}_{nj} = \mathbf{a}_j^{(L)} - \mathbf{y}_j$$

where L is the number of layers.

By moving back through the network from the output layer to input layer, we can compute all partial derivatives.

2.3 Support vector machine

Support Vector Machine(SVM) is famous for the most powerful supervised learning algorithms. The intuition of support vector machine comes from finding the best classifier to make the distance from the decision boundary to the data point the largest. In advance of starting support vector machine, we reparametrize ω to weights part $\mathbf{w}$ and bias part b, that is $\omega = (\mathbf{w}, b)$. For given data $\{\mathbf{x}_i, y_i\}_{i=1,2,\cdots,N}$ where $y_i \in \{-1, 1\}$, the margin r_i is equal to $\dfrac{|\mathbf{w}^T \mathbf{x}_i + b|}{||\mathbf{w}||^2}$ However, we do not check every margin for each data point. All we have to consider is the closest point to the decision boundary. We call these points *support vectors*. In the figure2.3, the support vectors are indicated by circles.

Thus, our problem is

$$\arg\max_{\mathbf{w},b} \{\frac{1}{||\mathbf{w}||} \min_i y_i(\mathbf{w}^T \mathbf{x}_i + b)\}$$

Adding constraint $y_i(\mathbf{w}^T\mathbf{x_i} + \mathbf{b}) \geq 1, \quad i = 1, 2, 3, \cdots, n$, our optimization problem changes to

$$\arg\min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||^2$$

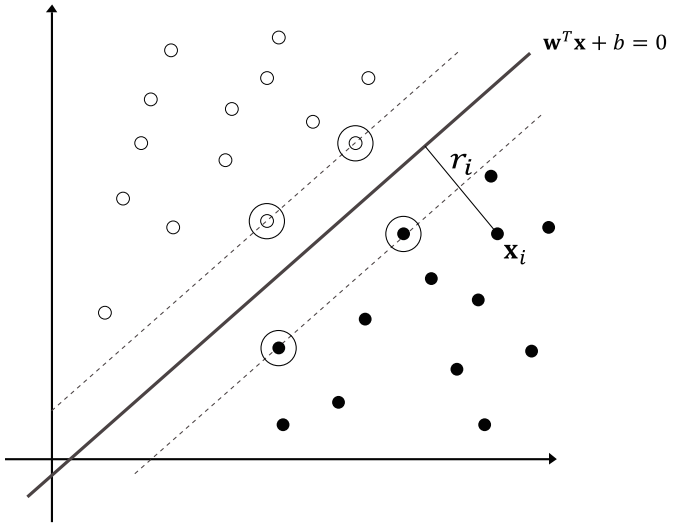


Figure 2.3: Support vector machine

To solve this problem with constraints, we adopt Lagrange multiplier $\alpha_i \geq 0$, giving the Lagrangian function

$$\mathscr{L}(\mathbf{w}, b, \alpha) = \frac{1}{2}||\mathbf{w}||^2 - \sum_i \alpha_i\{y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1\} \tag{2.3.1}$$

Since the derivatives of $\mathscr{L}(\mathbf{w}, b, \alpha)$ with respect to $\mathbf{w}$ and b have to be zero, the following conditions are also obtained.

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i, \quad \sum_{i=1}^{n} \alpha_i y_i \tag{2.3.2}$$

Combining (2.3.1) and (2.3.2) gives us the dual representation of the maximization problem, which is

$$\widetilde{\mathscr{L}}(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \tag{2.3.3}$$

subject to the constraints

$$\sum_i \alpha_i y_i = 0, \quad \alpha_i \geq 0 \quad i = 1, 2, 3, \cdots, n$$

In case of the overlapping data, we allow some errors by adding slack variable $\xi_i \geq 0$. Then constraint changes to $y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i$ and our problem becomes

$$\frac{1}{2}||\mathbf{w}||^2 + C \sum_{i=1}^{n} \xi_i$$

where C>0 controls the slack variable.

The following Lagrangian function is given by

$$\mathscr{L}(\mathbf{w}, b, \xi, \alpha, \beta) = \frac{1}{2}||\mathbf{w}||^2 + C \sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \alpha_i \{y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 + \xi_i\} - \sum_{i=1}^{n} \beta_i \xi_i \tag{2.3.4}$$

where α_i and β_i are Lagrangian multipliers. The corresponding Karush-Kuhn-Tucker conditions are given by

$$\begin{aligned}
\alpha_i &\geq 0 && (2.3.5)\\
y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 + \xi_i &\geq 0 && (2.3.6)\\
\alpha_i \{y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 + \xi_i\} &= 0 && (2.3.7)\\
\beta_i &\geq 0 && (2.3.8)\\
\xi_i &\geq 0 && (2.3.9)\\
\beta_i \xi_i &= 0 && (2.3.10)
\end{aligned}$$

The derivatives of $\mathscr{L}(\mathbf{w}, b, \xi, \alpha, \beta)$ with the respect to the $\mathbf{w}$, b and ξ_i are equal to zero. It gives us

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i, \quad \sum_{i=1}^{n} \alpha_i y_i = 0, \quad \alpha_i = C - \beta_i \tag{2.3.11}$$

Substituting (2.3.11) into (2.3.4), our goal is to find $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_n)$ that maximize the Lagrangian dual problem given by

$$\widetilde{\mathscr{L}}(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \tag{2.3.12}$$

subject to the constraints

$$\sum_{i=1}^{n} \alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C \quad i = 1, 2, 3, \cdots, n \tag{2.3.13}$$

When data lie very complicatedly, it is hard to decide boundary. In this situation, we can map original data space to much higher dimensional space where the data can be separable. We apply *the kernel function* ϕ for data $\mathbf{x}_i$.

Finally, (2.3.12) changes to

$$\widetilde{\mathscr{L}}(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$$

with constraint (2.3.13).

Sequential Minimal Optimization(SMO) algorithms are taken for the most common technique to solve this quadratic optimization problem[12].

Chapter 3

Data and experiment

3.1 Data and features

We have two data sets.

2009		2013	
AAA	6	AAA	7
AA	24	AA	46
A	45	A	50
BBB	42	BBB	21
under BBB	51	under BBB	8
Total	168	Total	132

Table 3.1: Data distribution

One is a set of credit ratings of companies in 2013. The other is 2009 data set. The reason we choose 2009 data is that its distribution differed from recent data because of 2008 financial crisis arising from USA. Data

sets consists of stock and KOSDAQ listed manufacturing companies. 'under BBB' contains the companies whose ratings are BB, B, C, $\cdots$, etc.

We have 30 features. They are based on financial statements and ratios in 2008 and 2012. We forecast the credit ratings by using the financial information in the previous year. Chosen features are usually indication of the stability of companies.

	Feature
X1	Assets
X2	Liabilities
X3	Long-term borrowings
X4	Liabilities ratio
X5	Operating income to operating capital
X6	Earning per share(EPS)
X7	Net sales
X8	Cash flows from operating activities
X9	Non-operating income
X10	Net incom for the year
X11	Net income to capital stock
X12	Interest expense to operating income
X13	Net worth growth rate
X14	Operating income to total capital
X15	Net income to stockholder's equity
X16	Cash flow per share(CPS)
X17	Book-value per share(BPS)
X18	Reserve ratio

	Features
X19	Current assets to non-current assets
X20	Stockholder's equity to total assets
X21	Cash ratio
X22	Current liabilities ratio
X23	Short-term borrowings to total borrowings
X24	Net working capital to total assets
X25	Total borrowings & bonds payable to total assets
X26	Cash flow to liabilities
X27	Cash flow to total borrowings
X28	Cash flow to total assets
X29	Cash flow to net sales
X30	Gross value-added to total assets

Table 3.2: Features

3.2 Experment

We conduct experiment to forecast credit ratings by utilizing machine learning algorithms introduced in chapter 2 : ordinal logistic regression, neural networks and support vector machine.

We divide the data 90% training set and 10% testing set by 10-cross validation. It gives each data point index from 1 to 10 randomly. Each indexed subset is reserved for testing and remaining subsets are used for training in rotation.

When training neural networks, the output $\mathbf{y}_i$ is a 5-dimensional vector since we have 5 classes and 15 hidden layers are used. The activation function

g is hyperbolic tangent function.

In case of the support vector machine, it is fundamentally a two-class classifier. Thus, we propose two approaches to construct multiclass support vector machines. First of all, we build them by using Directed Acyclic Graph, so-called *DAGSVM*[11]. They are one-against-one classifiers. For K classes, the DAGSVM has $K(K-1)/2$ binary classification SVMs. Since we have 5 classes, we construct 10 SVMs. See Figure3.1.

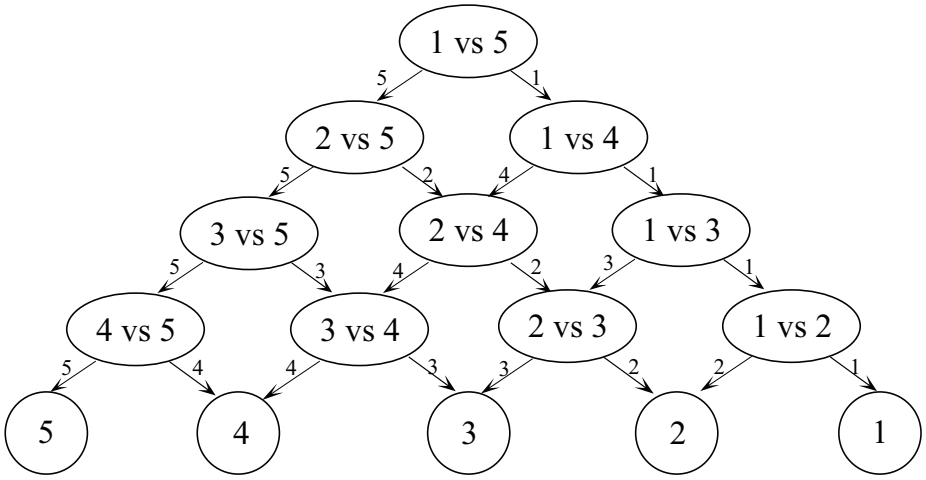


Figure 3.1: DAGSVM

Secondly, we use an ordinal regression algorithm. By (2.1.3) in chapter 2, we divide our data set into 4 binary classification data sets and derive 4 decision boundaries. We denote this method SVM2. The kernel function we take in the support vector machine is Gaussian kernel, which is

$$\phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j) = \exp(-\gamma||\mathbf{x}_i - \mathbf{x}_j||^2)$$

Chapter 4

Results

We try to compare the performance of three machine learning methods. The entire results are the same following Table 4.1.

	OLR[1]	NNs[2]	SVM1[3]	SVM2[4]
2009 data	61.30 %	63.09 %	66.67 %	60.11 %
2013 data	60.61 %	57.58 %	62.12 %	62.12 %
Time	4 $\sim$ 5 sec	15 $\sim$ 20sec	1$\sim$ 2 sec	1 $\sim$ 2sec

[1] Ordinal logistic regression.

[2] Neural networks with 15 hidden layers.

[3] DAGSVM

[4] Support vector machine with ordinal regression algorithm.

Table 4.1: Results

SVM1(DAGSVM) has the highest accuracy in the both data sets. SVM2 and neural networks hold the lowest accuracy in the 2009 and 2013, respectively. Support vector machine is the fastest learning method and neural

networks take long time to train and classify the data in comparison with others.

Prior to dealing with detailed results, we introduce measures used in a confusion matrix. *within-1-class accuracy* is the probability that allows the predictions within one class away from the true rank[1]. For each class i, *Recall* is ratio of the number of companies predicted correctly to the total number of the companies whose rank is i. *Precision* is ratio of the number of companies predicted correctly to the total number of companies whose predicted rating is i.

The following tables from Table 4.2 to Table 4.5 are confusion matrices of 2009 data set. The results of 2009 data hold low recall and precision of AAA in every method. Since we have few AAA companies in 2009, every method tends to show insignificant accuracies when it comes to having a right decision for AAA.

Likewise, in case of 2013 data, it shows same drawbacks. As they have a small number of under BBB and AAA companies, there are difficulties to decide these ranks. Most of methods have the bad recall and precision of AAA and under BBB. See tables from Table 4.6 to Table 4.9.

2009 data						
Ordinal Logistic Regression(accuracy = 61.30%)						
Actual rating	predicted rating					
	AAA	AA	A	BBB	Under BBB	Recall
AAA	2	2	1	0	1	0.3333
AA	3	11	8	1	1	0.4583
A	1	5	26	12	1	0.5778
BBB	0	0	7	30	5	0.7143
Under BBB	0	1	4	12	34	0.6667
Precision	0.3333	0.5789	0.5652	0.5455	0.8095	0.6130
within-1-class accuracy : 93.45 %						

Table 4.2: 2009 data : Ordinal logistic regression

2009 data						
Neural networks(accuracy = 63.09%)						
Actual rating	predicted rating					
	AAA	AA	A	BBB	Under BBB	Recall
AAA	0	4	1	0	1	0.0000
AA	2	13	8	1	0	0.5417
A	0	10	24	10	1	0.5333
BBB	0	2	10	26	4	0.6190
Under BBB	0	0	2	6	43	0.8431
Precision	0	0.4483	0.5333	0.6047	0.8775	0.6309
within-1-class accuracy : 95.24%						

Table 4.3: 2009 data : Neural networks

2009 data						
SVM1(accuracy = 66.67%)						
Actual rating	predicted rating					
	AAA	AA	A	BBB	Under BBB	Recall
AAA	3	1	1	0	1	0.5000
AA	2	15	7	0	0	0.6250
A	2	6	31	4	2	0.6889
BBB	0	2	12	23	4	0.5476
Under BBB	0	1	3	7	40	0.7843
Precision	0.4286	0.6	0.5636	0.6764	0.8511	0.6667
within-1-class accuracy : 92.86%						

Table 4.4: 2009 data : SVM1

2009 data						
SVM2(accuracy = 60.12%)						
Actual rating	predicted rating					
	AAA	AA	A	BBB	Under BBB	Recall
AAA	3	2	1	0	0	0.5000
AA	4	14	6	0	0	0.5833
A	3	4	26	10	2	0.5778
BBB	0	3	6	24	9	0.5714
Under BBB	0	4	5	8	34	0.6667
Precision	0.3000	0.5185	0.5909	0.5714	0.7556	0.6012
within-1-class accuracy : 89.29%						

Table 4.5: 2009 data : SVM2

2013 data						
Ordinal Logistic Regression(accuracy = 60.61%)						
Actual rating	predicted rating					
	AAA	AA	A	BBB	Under BBB	Recall
AAA	4	1	2	0	0	0.5714
AA	5	27	12	1	1	0.5870
A	0	9	36	4	1	0.7200
BBB	0	1	6	11	3	0.5238
Under BBB	0	0	1	5	2	0.2500
Precision	0.4444	0.7105	0.6316	0.5238	0.2857	0.6061
within-1-class accuracy : 94.7						

Table 4.6: 2013 data : Ordinal logistic regression

2013 data						
Neural networks(accuracy = 57.58%)						
Actual rating	predicted rating					
	AAA	AA	A	BBB	Under BBB	Recall
AAA	3	2	2	0	0	0.4286
AA	1	34	9	2	0	0.7391
A	1	9	27	12	1	0.5400
BBB	0	0	9	8	4	0.3809
Under BBB	0	0	1	3	4	0.5000
Precision	0.6000	0.7556	0.5625	0.3200	0.4444	0.5758
within-1-class accuracy : 94.7%						

Table 4.7: 2013 data : Neural networks

2013 data						
SVM1(accuracy = 62.12%)						
Actual rating	predicted rating					
	AAA	AA	A	BBB	Under BBB	Recall
AAA	4	3	0	0	1	0.5714
AA	2	38	6	0	0	0.8261
A	1	19	25	5	0	0.5000
BBB	0	3	3	14	1	0.6667
Under BBB	0	4	0	3	1	0.1250
Precision	0.5714	0.5671	0.7329	0.6364	0.5000	0.6212
within-1-class accuracy : 93.94%						

Table 4.8: 2013 data : SVM1

2013 data						
SVM2(accuracy = 62.12%)						
Actual rating	predicted rating					
	AAA	AA	A	BBB	Under BBB	Recall
AAA	6	1	0	0	0	0.8571
AA	3	33	10	0	0	0.7174
A	1	11	30	8	0	0.6000
BBB	0	1	4	11	5	0.5238
Under BBB	0	1	2	3	2	0.2500
Precision	0.6000	0.7021	0.6522	0.5000	0.2857	0.6212
within-1-class accuracy : 96.21%						

Table 4.9: 2013 data : SVM2

Chapter 5

Conclusion

Our results shows that the support vector machine with Directed Acyclic Graph is the best classifier in three methods. Now that within-1-class accuracy of three methods is close to 100%, we can guess a credit rating of company roughly based on only financial information by using machine learning methods. However, their accuracy is approximately 58∼66 %, which is relatively low. To improve the performance of machine learning. we may consider industrial characteristic or external environmental factors apart from financial statements.

Also, we use 30 typical financial information without any refinement. After feature selection such as Principal Component Analysis, stepwise and Analysis of variance, we might anticipate better performance.

Finally, we need to deal with the problem when data is extremely imbalanced. Both data shows bad precision and recall because we have few AAA companies in 2009 and under BBB companies in 2013. Prediction rule has difficulty in deciding their ratings due to imbalanced data. If we adjust balance of data, we could obtain better results.

Bibliography

[1] Z. Huang, H. Chen, C.-J Hsu, W.-H Chen and S. Wu, *Credit rating analysis with support vector machine and neural networks : a market comparative study*,Decision Support Systems, 37, 543-558, 2004.

[2] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer-Verlag, 2006.

[3] Y. Son, D.-J. Noh, J. Lee, *Forecasting trends of high-frequency KSOPI200 index data using learning classifiers*, Expert Systems with Applications **39** (2012), 11607-11615.

[4] L. Fisher, *Determinants of risk premiums on corporate bonds*, Journal of Political Economy, 1959, 217-237.

[5] J. O. Horrigan, *The determination of long term credit standing with financial ratios*, Journal of Accounting Research, **4** (1966), 44-62.

[6] G. E. Pinches, K. A. Mingo, *Multivariate analysis of industrial bond ratings*, Journal of finance 28 **1** (1973), 1-18.

[7] T. F. Pogue, R. M, Soldofsky, *What's in a bond rating?*, Journal of Financial and Quantitative Analysis **4**(1969), 201-228.

BIBLIOGRAPHY

[8] K. S. Kim, *Predicting bond rating using publicly available information*, Expert Systems with Application **29** (2005), 75-81.

[9] J. J. Maher and T. K. Sen, *Predicting Bond Ratings Using Neural Networks: A comparison with Logisitc Regression*, Finance and management **6** (1997), 59-72.

[10] L. Cao, L. K. Guan and Z. Jingquing, *Bond rating using support vector machine*, Intelligent Data Analysis **10** (2006), 285-296.

[11] J. C. Platt, N. Cristianini and J. Shawe-Taylor, L. K. Guan and Z. Jingquing, *Large Margin DAGs for multiclass classification*, Advances in Neural Information Processing Systems **12** (2000), 547-553, MIT Press.

[12] J. C. Platt, *Sequential Minimal optimization: A Fast Algorithm for Training Support Vector Machine*, Technical Report MSR-TR-98-14, Microsoft Research, 1998.

[13] A. Ng, *CS229: Machine learning, Lecture notes*, Stanford University, San Francisco, CA, Retrieved from http://cs229.stanford.edu/materials.html, 2012.

[14] A. Lee, *Default Prediction by Using Machine Learning Methods*, Seoul National University, 2014.

국문초록

이 논문에서는 기업의 신용 등급에 대해 논한다. 가장 정확하게 신용 등급을 예측하는 규칙을 찾는 것이 목표이며 세 가지의 대표적인 기계 학습 방법 로지스틱 회귀분석, 뉴럴 네트워크, 서포트 벡터 머신을 소개하고 비교하여 어떤 것이 가장 효과적인지 알아본다. 또한, 그것들의 정확성과 효율성을 분석한다. 2009년과 2013년도의 실제 신용 등급 자료와 그 전년도 재무제표 자료를 바탕으로 실험하였다.

주요어휘: 신용 등급, 기계 학습, 로지스틱 회귀분석, 뉴럴 네트워크, 서포트 벡터 머신

학번: 2011-23201