CREDIT RISK MODELING USING MACHINE LEARNING TECHNIQUES

by

Murat Emre Kaya

B.S., Computer Engineering, Boğaziçi University, 2003

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering
Boğaziçi University
2006

ACKNOWLEDGEMENTS

I want to thank to Fikret Gürgen first for his patience during the development of this thesis. His positive and helpful character played a very crucial role. Also, I want to thank to Levent Akın and Nesrin Okay for all their crucial contributions.

My friend Taskin, I won't forget you of course. You helped me a lot for the thesis since I was not in Istanbul. Thank you very much. By the way, you won the bet. I will offer the dinner.

Also, lots of thanks to Manikanth and Greg. Thank you for all your help.

Special thanks to Necmettin and Yavuz Abi for opening their house to me and their hospitality.

This thesis may not have been completed unless I came to Dubai. So special thanks to 3 people: Mrs. Bayrakgil, Evren and Attiq.

Of course, thank you mom and dad, for all your support as always.

ABSTRACT

CREDIT RISK MODELING USING MACHINE LEARNING TECHNIQUES

In this thesis, credit scoring ability of several machine learning techniques were investigated such as multi-layer perceptron (MLP), radial basis function (RBF), k-nearest neighbor (k-NN) and support vector machines (SVM). Statistical technique logistic regression was also used for the purpose of comparison. In the second part, a two layer cascading model methodology called SVM-Reject was proposed which does not classify the instances under a threshold with its first layer model SVM. Experiments were performed by using on German credit data set and model comparisons are based on accuracy, error cost and ROC Analysis. Results show that SVM is a good option for credit scoring applications and SVM-Reject is the most accurate model. In the last part of this study, PD (probability of default) model building by using machine learning techniques was discussed in a comparative manner with logistic regression. This part is mostly from a bank's point of view and includes practical information as well.

ÖZET

KREDİ RİSKİ MODELLEMESİNDE MAKİNA ÖĞRENMESİ TEKNİKLERİ

Bu tezde yapay sinir ağları, destek vektör makineleri (DVM) ve k-en yakın komşu sınıflandırma algoritması gibi çesitli makina öğrenmesi tekniklerinin kredi skorlamadaki yeteneği araştırıldı. İstatistiksel bir teknik olan lojistik regresyon da karşılaştırma amacıyla kullanıldı. İkinci bölümde, belli eşik değerinin altındaki örnekleri sınıflandırmayan iki katmanlı DVM-Reddet adında bir metodoloji önerildi. Deneyler Alman (German) kredi data kümesini kullanarak ve karşılaştırmalar doğruluk, hata maliyeti ve ROC analizinden faydalanarak yapıldı. Sonuçlar gösteriyor ki, DVM kredi skorlaması için iyi bir alternatif ve DVM-Reject bu çalışmadaki en başarılı model. Çalışmanın en son kısmında makina öğrenmesi teknikleri kullanılarak temerrüt olasılığı (PD) modeli oluşturulması da tartışıldı. Bu kısım daha çok bir bankanın bakış açısıyla, pratik bilgiler de verecek şekilde tasarlandı.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS/ABBREVIATIONS

C	Penalty parameter of the error term
C_{bad}	Cost of error type bad
C_{good}	Cost of error type good
e_{bad}	Error rate of error type bad
e_{good}	Error rate of error type good
$K(x_i, x_j)$	Kernel function
t	Threshold value for SVM-Reject
x_i	N Dimensional data vector
y_i	Label corresponding to the data vector
γ	Parameter kernel function
ϵ_{bad}	Error type bad
ϵ_{good}	Error type good
ϕ	Mapping function
EAD	Exposure at default
EL	Expected loss
kNN	k-nearest neighbor
LGD	Loss given default
MLP	Multi-layer perceptron
PD	Probability of default
RBF	Radial basis function
SVM	Support vector machines

1. INTRODUCTION

1.1. What is Credit Scoring?

Credit scoring is used by most financial organizations to help them to decide whether to grant credit to an applicant or not. A score is generated for the applicant by using the relevant details of the application. Scorecards can be based on a simple template or a very sophisticated machine learning technique. Logistic regression (which is a statistical model), is one of the most commonly used models in the credit industry. If a scorecard is performing well, it should assign high scores to good borrowers and low scores to bad borrowers. In other words, the average scores of good borrowers should be higher than the average score of bad ones.

Credit scoring is important due to several reasons. Firstly, for retail markets, the number of customers applying for a credit product is large. Therefore, it is impossible to evaluate them one by one by a specialist which brings the necessity of automated evaluation process. At this point, credit scoring comes into place and reduces evaluation time significantly. Secondly, credit scoring applies the same criteria to all applicants regardless of their gender, nationality or other factors. This is also the case for commercial credit applications where applicants are mostly the companies. Although the number of commercial applications in a financial organization is not as many as retail applications and it is possible to evaluate them by credit specialists, credit scoring has always the advantage of objective decision making.

1.2. History of Credit Scoring

While the history of credit is nearly 5000 years, the history of credit risk scoring is only 50 years old. Credit scoring is a way to understand different groups in a population such as good borrowers and bad borrowers.

After World War 2 started, all the lender companies had difficulties with credit

management. Since credit analysts were being drafted into military service, it was very difficult to find an expert of credit risk. Therefore, lending companies used the rules written down by the credit analysts to decide whether to give the loan or not. Some of these were numerical scoring systems and others were the conditions which were needed to be satisfied.

In 1950s, statistical techniques were developed for credit risk scoring and being used for the purpose of lending decisions. After the arrival of credit cards in 1960s, banks and other credit card issuers understood the usefulness of credit risk scoring. The amount of people applying for a credit card was huge and so it was impossible to make a judgemental credit decision for each application. The solution was to automate the credit decision which was succeeded by using credit risk scoring. Credit risk scoring not only decreased the amount of time spent on one application, but also it reduced the default rates by 50 percent.

In 1980s, after the success of credit scoring techniques in credit cards, banks started using the same techniques also for the other products such as personal loans, home loans, etc. In these years, logistic regression and linear programming were introduced. More recently, expert systems and neural networks were also used.

Today, the aim is not just to identify which borrowers are good and which are bad. Now, the lenders are trying to maximize their profit from customer. Instead of rejecting the credit application when the borrower is risky, the firm can apply more margin on the loan. By this way non-risky customers get good interest rates on their loans contrary to the risky ones.

1.3. Literature Review

The credit industry has experienced two decades of rapid growth with significant increases in installment credit, single-family mortgages, auto-financing, and credit card debt. Credit scoring models have been widely used by the financial industry during this time to improve cash flow and credit collections. The advantages of credit scoring

include reducing the cost of credit analysis, enabling faster credit decisions, closer monitoring of existing accounts, and prioritizing collections [1], [2].

With the growth in financial services there have been mounting losses from delinquent loans. In response, many organizations in the credit industry are developing new models to support the credit decision. The objective of these new credit scoring models is increased accuracy, which means more creditworthy applicants are granted credit thereby increasing profits; non-creditworthy applicants are denied credit thus decreasing losses.

Linear discriminant analysis (LDA), a simple parametric statistical model, was one of the first credit scoring models. The appropriateness of LDA for credit scoring has been questioned because of the categorical nature of the credit data and the fact that the covariance matrices of the good and bad credit classes are not likely to be equal. The credit data is usually not normally distributed, although Reichert reports this may not be a critical limitation [3]. More sophisticated models are being investigated today to overcome some of the deficiencies of the LDA model. Henley [4] explores a logistic regression model for credit scoring applications. Other researchers are currently investigating nonparametric statistical models like k nearest neighbor [5], classification trees [6], [7] and neural network models for credit scoring applications [8], [9], [10].

2. MODELS FOR CREDIT SCORING

In this chapter, theoretical background is provided on five models (logistic regression, MLP, RBF, k-NN and SVM) which were used for credit scoring in this thesis.

2.1. Logistic Regression

Logistic regression is part of a category of statistical models called generalized linear models.

Logistic regression allows one to predict a discrete outcome, such as group membership, from a set of variables that may be continuous, discrete, dichotomous, or a mix of any of these. Generally, the dependent or response variable is dichotomous, such as presence/absence or success/failure. Discriminant analysis is also used to predict group membership with only two groups. However, discriminant analysis can only be used with continuous independent variables. Thus, in instances where the independent variables are a categorical, or a mix of continuous and categorical, logistic regression is preferred.

The dependent variable in logistic regression is usually dichotomous, that is, the dependent variable can take the value 1 with a probability of success p, or the value 0 with probability of failure 1-p. This type of variable is called a Bernoulli (or binary) variable.

As mentioned previously, the independent or predictor variables in logistic regression can take any form. That is, logistic regression makes no assumption about the distribution of the independent variables. They do not have to be normally distributed, linearly related or of equal variance within each group.The relationship between the predictor and response variables is not a linear function in logistic regression, instead,

the logistic regression function is used, which is the logit transformation of p:

$$p = \frac{e^{(a+b_1x_1+b_2x_2+...+b_ix_i)}}{1+e^{(a+b_1x_1+b_2x_2+...+b_ix_i)}} \tag{2.1}$$

This equation can also be written as follows:

$$p = \frac{1}{1+e^{-(a+b_1x_1+b_2x_2+...+b_ix_i)}} \tag{2.2}$$

where a is the constant of the equation and, b is the coefficient of the predictor variables.

The goal of logistic regression is to correctly predict the category of outcome for individual cases. To accomplish this goal, a model is created that includes all predictor variables that are useful in predicting the response variable. Several different options are available during model creation. Variables can be entered into the model in the order specified by the researcher or logistic regression can test the fit of the model after each coefficient is added or deleted, called stepwise regression.

2.2. Neural Networks

A neural network is a massively parallel distributed processor that has a natural propensity for storing experiential knowledge and making it available for classification [11]. Multi-layer perceptron (MLP) is a type of feed-forward networks. The learning rule is back-propagation [12]. The feed-forward network has three layers: input layer, hidden layer, and output layer. The hidden layer is between the input and the output layers. For example, a simple feed-forward network is shown in Figure 2.1, where there are two nodes in the input layer, two nodes in the hidden layer, and one node in the output layer. Weights are incorporated into the connections from input nodes to hidden nodes and from hidden nodes to the output node.

For linear processes, binary thresholding can be used in the hidden and output units. For nonlinear processes, activation functions are used rather than thresholding.

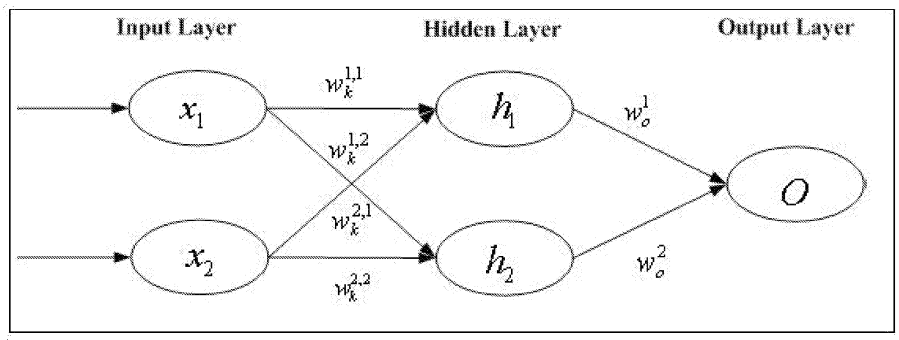

Figure 2.1. The architecture of MLP

Sigmoid function can be used as the activation function:

$$h_j = sigmoid(\sum w_k^{i,j} x_{i}{}_{i=1}^{N_I}) \tag{2.3}$$

$$O = sigmoid(\sum w_0^j h_{j}{}_{j=1}^{N_H}) \tag{2.4}$$

where x_i is the input feature at the input node, $w_h^{i,j}$, is the weight connecting the input node with the hidden node, w_0^j is the Weight connecting the j th hidden node with the output node, and O is the output value. The sigmoid function is defined as:

$$sigmoid(x) = \frac{1}{(1+e^{-x})} \tag{2.5}$$

The back-propogation training is conducted to obtain the correct weights. Mean square error (MSE) can be computed between the actual output and the desired output for the given input in the training set. The error function E can be obtained by:

$$E = (T-O)^2 \tag{2.6}$$

where T is the value of the desired target. To reduce the mean square error, it is necessary to calculate the gradient of the error function with respect to each weight. One may then move each weight slightly in the opposite direction to the gradient. The

gradient function for the weights in the output layer is shown below:

$$\delta_j = \frac{\partial E}{\partial W_0^j} = \frac{\partial (T-O)^2}{\partial W_0^j} \tag{2.7}$$

$$W_0^j + 1 = W_0^j - \eta \delta_j \tag{2.8}$$

From Equation 2.8, the new values for the network weights are calculated by multiplying the negative gradient with a step size of parameter η (called the learning rate). The weights in the hidden layer are updated using the same procedure. After all correct weights are computed, neural network is completely constructed.

2.3. Radial Basis Function

Design of a neural network can be viewed as a curve-fitting (approximation) problem in a high-dimensional space. From this viewpoint, learning is accomplished by finding a surface in a multi-dimensional space. This surface is used to interpolate the test data. Radial basis function (RBF) network is a fully connected network and generally is used as a classification tool. In a RBF model, the layer from input nodes to hidden neurons is unsupervised and the layer from hidden neurons to output nodes is supervised [13]. The transformation from the input to the hidden space is nonlinear, and the transformation from the hidden to the output space is linear. The hidden neurons provide a set of functions that constitute an arbitrary basis for the input patterns. These are the functions known as called radial basis functions. Through careful design, it is possible to reduce a pattern in a high-dimensional space at input units to a low-dimensional space at hidden units.

In Figure 2.2, the network forms the following equation:

$$y_k(x) = \sum w_{kj} h_j(x) + w_{k0} \tag{2.9}$$

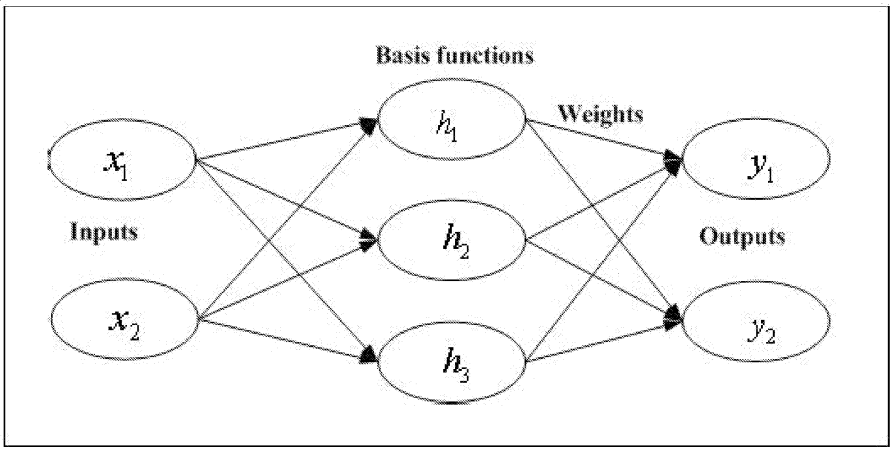

Figure 2.2. The architecture of RBF

where $h_j(x)$ is a Gaussian function typically, and w_{k0} is the bias or threshold. The Gaussian basis function is used as an activation function. That is

$$h_j(x) = e^{\frac{-||x-\mu_j||^2}{2\sigma_j^2}} \tag{2.10}$$

where x is the d -dimensional input vector with element x_i , and μ_j and σ_j are respectively the center and the standard deviation of the Gaussian basis function. Since the first and second layers of RBF network are unsupervised and supervised respectively, a two-stage training procedure is used for training the RBF model. In the first stage, the input data set is used to obtain the parameters of the activation functions (like μ_j and σ_j). In the second stage, the optimal weights between hidden neurons and output nodes are obtained by minimizing a sum-of-square error function. The following procedures describe the steps of obtaining the optimal weights.

By absorbing the bias parameter, w_{k0}, into the weights, Equation 2.9 can be revised as

$$y_k(x) = \sum w_{kj} h_j(x) \tag{2.11}$$

where $h_j(x)$ is an extra basis function with the activation value 1. Equation 2.11 can

be rewritten using the matrix notation as

$$Y(x) = W\Phi \tag{2.12}$$

where $W = [w_{kj}]$ and $\Phi = [h_j(x)]$.

The sum-of-square error function, E , can be described as

$$E = 1/2 \sum \sum [y_k(x^n) - t_k^n]^2 \tag{2.13}$$

where x^n is the input data set, and t_k^n is the target value for the output unit k.

By differentiating E with respect to w_{kj} and setting derivative to zero, the optimal weights can be obtained. The solutions of weights can be expressed using the matrix notation as:

$$(\Phi^{'T}\Phi)W^{'T} = \Phi^{'T}T \tag{2.14}$$

where $W = [w_{kj}]$ and $\Phi = [h_j(x)]$.

By multiplying $(\Phi^T\Phi)^{-1}$ to Equation 2.14, the solution for the weights is given as:

$$W^T = (\Phi^T\Phi)^{-1}\Phi^T T \tag{2.15}$$

$$W^T = \Phi^+ T \tag{2.16}$$

where $\Phi^+ = (\Phi^T\Phi)^{-1}\Phi^T$ being the pseudo-inverse of Φ. After computing the optimal weights, the RBF network can be used as a classifier to segment the test data into the corresponding classes, with -1 indicating a non-flare state and 1 indicating a flare state.

RBF is strongly dependent on the quality of the employed learning strategy and the quantity of training images. The aim of an adaptive learning RBF network is to reduce the required knowledge of the system parameters with a minimum amount of performance loss. The RBF network requires knowledge of three different parameters per neuron:

- The center vector μ_j
- The weights w_{kj}
- The radius σ_j

One unsupervised learning strategy is the self-organizing feature map. When the algorithm has converged, prototype vectors which corresponding to nearby points on the feature map grid have nearby locations in input space. However, the imposition of the topographic property, particularly if the data is not intrinsically two-dimensional, may lead to a sub-optimal placement of vectors.

Here K-Means unsupervised learning strategy is used as follows: let μ_j be the mean of the data points in set S_j given by:

$$\mu_j = 1/N_j \sum_{n \epsilon S_j} x^n \tag{2.17}$$

The initial centers are randomly chosen from the data points, and the nearest μ_j is updated using:

$$\Delta\mu_j = \eta(x^n - \mu_j) \tag{2.18}$$

where η is the learning rate parameter.

The second parameter of the RBF network is the weight w_k of the output layer. It can be done by using the LMS algorithm. The LMS algorithm was originally developed by Widrow and Hoff in 1960 and is also known as the Widrow-Hoff rule, see [14]. The

weights can be summarized as follows:

$$\Delta W = 2\eta(y_k(x^n) - t_k^n)h(x^n) \tag{2.19}$$

where η is the learning rate of LMS, and $y_k(x^n)$ and t_k^n are the responses of the RBF network and the desired response. The vector $h(x^n)$ contains the un-weighted responses of all neurons in the hidden layer.

The last parameter of the RBF network is the radius or spread of the radial function. An average of the center spread of all RBFs can be used to calculate the radius. After the centers μ_j are established, σ^2 can be derived from the center as:

$$\sigma^2 = 1/M \sum_{j=1}^{M} ||y_k(x^n) - \mu_j||^2 \tag{2.20}$$

where M is the number of hidden nodes.

2.4. k-Nearest Neighbor Algorithm

In pattern recognition, the k-nearest neighbour algorithm (k-NN) is a method for classifying phenomena based upon observable features, similar to the nearest neighbor classification method.

The difference lies in the fact that rather than assigning a classification based upon the classification of the nearest neighbor (the nearest neighbor is normally calculated using a distance measure such as the Euclidean distance) the algorithm selects a set which contains the k nearest neighbors and assigns the class label to the new data point based upon the most numerous class with the set. The best choice of k depends upon the data; generally, larger values of k reduce the effect of noise on the classification, but make boundaries between classes less distinct.

2.5. Support Vector Machine

Support Vector Machine (SVM) is generation learning system based on advances in statistical learning theory. It has been successfully applied in text categorization, hand-written character recognition, image classification, biosequences analysis, etc.

Suppose l observations are given. Each observation consists of a pair: a vector $x_i \epsilon R^n, i = 1, ..., l$. and the associated truth y_i , given to us by a trusted source. It is assumed that there exists density distribution $p(x, y)$ from which these data are drawn, i.e., the data are assumed independently drawn and identically distributed. Now suppose there is a machine to learn the mapping $x_i \rightarrow y_i$. The machine is actually defined by a set of possible mappings $x_i \rightarrow f(x, \alpha)$, where the function $f(x, \alpha)$ is labeled by the adjustable parameter α. For example, a neural network with a fixed architecture and with a corresponding to the weights and biases, is a learning machine in this sense. $f(x, \alpha)$ function could represent a set of Radial Basis functions or MLPs with a certain number of hidden neurons [15]. The expectation of the test error for a trained machine is therefore:

$$R(\alpha) = \int 1/2|y - f(x, \alpha)|p(x, y)dxdy \tag{2.21}$$

where $R(\alpha)$ is called the expected risk.

The empirical risk $R_{emp}(\alpha)$ is defined to be just the measured mean error rate on the training set (for a fixed, finite number of observations).

$$R_{emp}(\alpha) = \frac{1}{2l} \sum_{i=1}^{l} |y_i - f(x_i, \alpha)| \tag{2.22}$$

The empirical risk minimization (ERM) inductive principle is to minimize the risk functional (20) on the basis of empirical data x_i, y_i. The ERM principle is based on the law of large numbers converges in probability to the expected risk (strategy followed by MLP and RBF neural networks). By choosing some η such that $0 \leq \eta \leq 1$,

the following bound holds).

$$R_{emp}(\alpha) = R_{emp}(\alpha) + \sqrt{\frac{h(log(2l/h) + l) - log(\eta/4)}{l}} \tag{2.23}$$

where h is a non-negative integer called the Vapnik Chervonenkis (VC) dimension. The right hand side of Equation 2.23 is the bound on the risk, and the second term on the right hand side is called the Confidence interval. When l/h is large, the Confidence interval is small. The actual risk is then close to the value of the empirical risk one. In this case, a small value of the empirical risk guarantees a small value of the expected risk. When l/h is small, a small $R_{emp}(\alpha)$ does not guarantee a small value of the actual risk. In this case, to minimize the actual risk has to minimize the confidence interval. To minimize the right-hand side of the bound risk in Equation 2.23, one has to make the VC dimension a controlling variable.

In the previous section, a classical neural network was considered, which implement the first strategy: Keep the confidence interval fixed and minimize the empirical risk. Below there is a new type of universal learning machine, the Support Vector Machine, implements the second strategy: Keep the value of the empirical risk fixed and minimize the confidence interval [16]. SVM is based on the structural risk minimization (SRM) inductive principle. The SRM principle is intended to minimize the risk functional with respect to the both terms, the empirical risks, and the confidence interval [17]. Therefore, SVM is a better strategy than classical neural network such as MLP neural network. The basic idea of a support vector machine is to separate the fixed given input pattern vectors into two classes using a hyperplane in the high dimensional space [18]. This means the VC-dimension is always incremented h by one if the input pattern vectors cannot be separated. The process continues until the separation is satisfied.

SVM intends to separate the input pattern using the minimum VC-dimension h , which minimizes the total risk by considering both empirical risk and confidence interval. Figure 2.3 shows the SVM with a linear hyperplane.

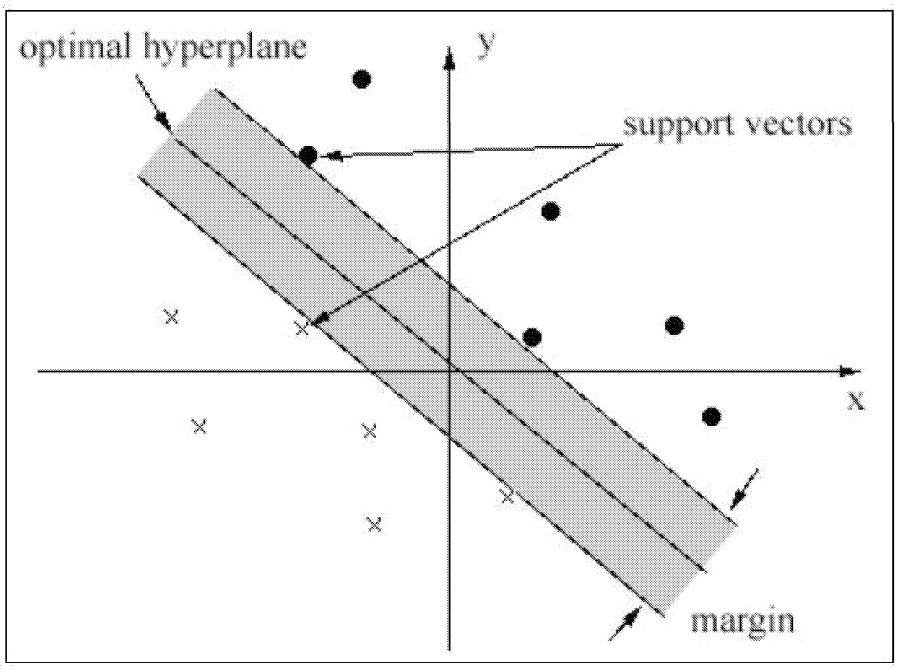

Figure 2.3. Binary SVM classification

Consider a two-class classification problem where patterns are represented as an N-dimensional vector x. For each corresponding vector, there exists a value $y\epsilon\{-1,+1\}$.

$$x = \{x_1, x_2, x_3, ..., x_m\}\epsilon R^N \tag{2.24}$$

$$y = \{y_1, y_2, y_3, ..., y_m\}\epsilon\{-1,+1\} \tag{2.25}$$

The problem is to find a decision function $g(x)$ such that the class label y of any example x can be accurately predicated. Using mathematical terms, $g : R \rightarrow \{-1,+1\}$ is obtained.

For linear support vector classifier

$$f(x) = (w.x) + b \tag{2.26}$$

where $w\epsilon R^N$ and $b\epsilon R$.

the set of labeled training patterns

$$(y_1, x_1), ..., (y_l, x_l), y_i \epsilon \{-1, +1\} \tag{2.27}$$

It is linearly separable if there exist a vector w and a scalar b such that the following inequalities are valid for all elements of the training set.

$$w.x_i + b \geq 1 \quad if y_i = 1, \tag{2.28}$$

$$w.x_i + b \leq -1 \quad if y_i = -1. \tag{2.29}$$

The optimal hyperplane is the unique one which separates the training data with a maximal margin. The distance $\rho(w, b)$ is given by

$$\rho(w, b) = min_{x:y=1}(\frac{x.w}{||w||}) - max_{x:y=-1}(\frac{x.w}{||w||}) \tag{2.30}$$

Because the optimal hyperplane (w_0, b_0) is the arguments that maximize the distance $\rho(w, b)$, Equation 2.31 is obtained from 2.28, 2.29 and 2.30.

$$\rho(w, b) = \frac{2}{||w||} = \frac{2}{\sqrt{w_0.w_0}} \tag{2.31}$$

It means that the optimal hyperplane is the unique factor that minimizes $w.w$ under the constraints 2.28 and 2.29. This problem is usually solved by means of the classical method of Lagrange multipliers.

If the N nonnegative Lagrange multipliers are denoted as $\alpha = (\alpha_1, \alpha_2, ..., \alpha_n)$,

the solution is equivalent to determining the saddle point of the function.

$$L(w, b, \alpha) = 1/2w.w - \sum_{i=1}^{N} \alpha_i (y_i(w.x_i + b) - 1) \tag{2.32}$$

At the saddle point, L has a minimum for $w = w_0$ and $b = b_0$ and a maximum for $\alpha = \alpha^0$.

$$\frac{\partial L(w_0, b_0, \alpha^0)}{\partial b} = \sum_{i=1}^{N} y_i \alpha_i^0 = 0 \tag{2.33}$$

$$\frac{\partial L(w_0, b_0, \alpha^0)}{\partial w} = w_0 - \sum_{i=1}^{N} \alpha_i^0 y_i x_i = 0 \tag{2.34}$$

Substituting Equations 2.33 and 2.34 into the right hand side of 2.32, it is reduced to the maximization of the function

$$W(\alpha) = \sum_{i=1}^{N} \alpha_i - 1/2 \sum_{i,j}^{l} \alpha_i a_j y_i y_j (x_i.x_j) \tag{2.35}$$

From 2.34 it follows that

$$w_0 = \sum_{i=1}^{N} \alpha_i^0 y_i x_i \tag{2.36}$$

The parameter b_0 can be obtained from the Kuhn-Tucker condition.

$$b_0 = y_j - w_0.x_j \tag{2.37}$$

The problem of classifying a new data point x is now simply solved by looking at

the sign of

$$w_0.x + b_0 \tag{2.38}$$

For the nonlinear case, kernel functions are used to transfer input data to feature space. The nonlinear kernel functions are given following:

Polynomial support vector classifier:

$$k(x, x^{'}) = < x, x^{'} >^{d} \tag{2.39}$$

Gaussian RBF kernel classifier:

$$k(x, x^{'}) = \exp(-\frac{||x - x^{'}||^2}{2\sigma_j^2}) \tag{2.40}$$

3. EXPERIMENTS AND MODEL COMPARISON

3.1. Dataset Description

In this thesis German credit data set from UCI Machine Learning Repository (http://www.ics.uci.edu/mlearn/databases/statlog/german/) was used which is a retail credit data set. It contains observations on 20 variables for 1000 past applicants for credit. Each applicant is rated either as good credit (700 cases) or bad credit (300 cases). Variables in the data set are shown in table 3.1.

3.2. Experiment Design and Parameter Selection

10-fold cross validation was used to test model performances. The data set was divided into ten partitions, and the holdout method was repeated ten times. Each time, one of the ten partitions was used as the test set and the other nine partitions were put together to form a training set. Then the average error across all ten trials was calculated. The advantage of cross validation is that it minimizes the affect of how the data gets divided. Every data point is included in a test set exactly once.

Five different models were tested in this study: SVM, MLP, RBF, kNN and logistic regression. For MLP and RBF, number of hidden layers is crucial. If there are too few hidden layers, there will be high training and generalization errors due to under-fitting and high statistical bias. If there are too many hidden layers, there will be low training error but still have high generalization error due to over-fitting and high variance. For these models, number of hidden layers was determined experimentally and 20 hidden layers were used for both of them. Another important point about MLP and RBF is that, they have a stochastic training process and each run of 10-fold cross-validation gives different error rates. Therefore, 10-fold cross validation was applied ten times to these two models and error rate was calculated by taking the average of ten runs. By this way, stochastic variability of the neural network training process is reduced since the average is considered. Moreover, for kNN, several k values were

Table 3.1. German Credit Data

Variable #	**Variable Name**	**Variable Type**
1	Status of checking account	Qualitative
2	Duration in month	Numerical
3	Credit history	Qualitative
4	Purpose	Qualitative
5	Credit amount	Numerical
6	Savings account/bonds	Qualitative
7	Present employment	Qualitative
8	Instalment rate	Numerical
9	Personal status and sex	Qualitative
10	Other debtors/guarantors	Qualitative
11	Present residence since	Numerical
12	Property	Qualitative
13	Age in years	Numerical
14	Other instalment plans	Qualitative
15	Housing	Qualitative
16	Number of existing credits	Numerical
17	Job	Qualitative
18	Number of people being liable	Numerical
19	Telephone	Qualitative
20	Foreign worker	Qualitative

tested and k was selected $k = 3$ which gave the best accuracy. Finally, grid search was used to determine the best values of parameters C and γ of SVM and this will be described in detail in the following section.

3.2.1. Grid Search

In this study, RBF kernel is used whose formula is given in equation 3.1 which has parameter γ. But, the kernel parameter is not the only parameter whose value should be selected for SVM model. There is also a penalty parameter C in the optimization problem. Our aim is to find out the best pairs of (C, γ) values to obtain best 10-fold cross validation accuracy.

$$K(x_i, x_j) = \exp(-\gamma||x_i - x_j||^2),\ \gamma > 0. \tag{3.1}$$

Grid search is used for this purpose which tries different (C, γ) values with 10-fold cross validation within a specified value range. The pair which gives maximum accuracy is picked. Search ranges for the parameters were:

$$C = 2^{-5}, 2^{-3}, ..., 2^{15} \tag{3.2}$$

$$\gamma = 2^{-15}, 2^{-13}, ..., 2^{3} \tag{3.3}$$

In order to perform this search, a Python script called "grid.py" was used which is available in LIBSVM website [19]. Finally C value of $C = 32.768$ and γ value of $\gamma = 0.000120703125$ were used to train the model. Figure 3.1 shows the accuracies of (C, γ) values tried.

3.3. Model Comparison

In this section, the results of the experiments are discussed. Models are compared according to accuracy and error cost criteria, moreover ROC Analysis is performed.

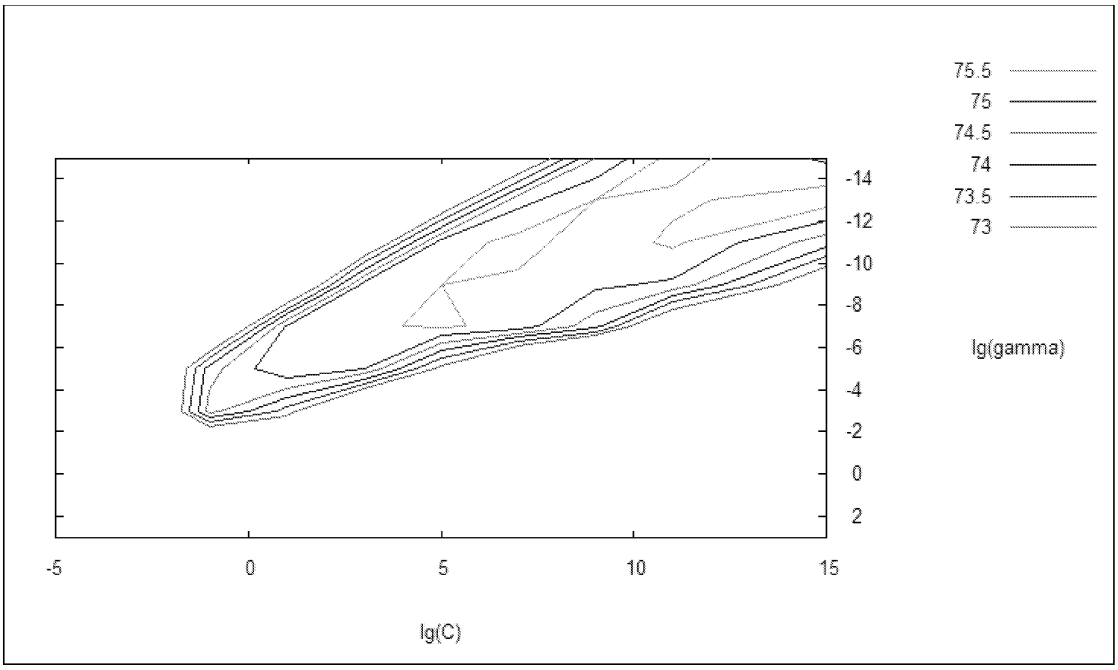

Figure 3.1. Grid Search

3.3.1. Accuracy

This comparison is simply based on the average accuracy of ten independent hold-out partitions used in 10-fold cross validation. Accuracy is defined as the number of correct classifications over total number of classifications and accuracy of the 10-fold cross validations is calculated by dividing the summation of accuracies of each hold-out partition by ten. As described in the previous section, 10-fold cross validation is applied ten times on MLP and RBF due to the stochastic nature of their training process, so accuracy calculation of these models is slightly different. First, accuracy of each cross validation run and then average of these ten runs are calculated. The results are given in table 3.2. As seen from the results, support vector machines and logistic regression are the most accurate two models. Accuracy of support vector machines is slightly better than logistic regression.

3.3.2. Error cost criterion

There are two possible erroneous classifications for credit scoring. First, an actual good borrower may be rejected, ϵ_{good}. Second, an actual bad borrower can be granted

Table 3.2. Accuracy comparison of models (10-fold cross validation)

Model	**Accuracy**
SVM	% 75.8
RBF	% 75.0
MLP	% 74.1
KNN	% 63.4
Logistic Regression	% 75.6

credit, ϵ_{bad}. If the loss due to a default customer is significant for the application, ϵ_{bad} is more critical for the lender. This is mostly the case for commercial credit applications where the amount of loan is big. It can also be valid for some special type of retail portfolio where customers have high limits. However, for most of the retail credit applications, both error costs have the same importance and even error ϵ_{good} may be more critical which causes actual good borrowers to be rejected. In this section, error cost are calculated for the following two pairs of (C_{good}, C_{bad}) values which represent the cost of error ϵ_{good} and the cost of error ϵ_{bad}:

1. (C_{good}, C_{bad}) = (1,1) {represents retail credit applications}
2. (C_{good}, C_{bad}) = (1,5) {represents commercial credit applications}

The formula of our error cost is:

$$Error\ cost = C_{bad} * e_{bad} + C_{good} * e_{good} \tag{3.4}$$

where e_{good} is the rate of error ϵ_{good} and is the rate of error ϵ_{bad}.

Table 3.3 shows error costs of all models. Since German credit data set is a retail credit data set, SVM seems to be a better option with $cost = 0.242$. However, as stated before, some retail credit applications (especially the ones with high limits) may give importance to having the rate of error ϵ_{bad} as least as possible (like commercial credit applications). If that is the case, then logistic regresion is more appropriate with

$cost = 0.892$.

Table 3.3. Error cost comparison of models (10-fold cross validation)

Model	$(C_{good},C_{bad}) = (1,1)$	$(C_{good},C_{bad}) = (1,5)$
SVM	0.242	0.902
RBF	0.250	0.952
MLP	0.259	0.935
KNN	0.366	1.254
Logistic Regression	0.244	0.892

3.3.3. ROC Analysis

In this section, ROC (receiver operating curve) Analysis is performed to compare model performances. Receiver operating curve is a graphical plot of the true positive rate (sensitivity) vs. false positive rate (1 - specificity) for a binary classifier system. Here are the criteria to evaluate the performances of the models based on ROC curve:

1. If model A and model B has same TP (true positive) rate but FP (false positive) rate of model A is lower, then model A is better than model B.
2. If model A and model B has same FP rate but TP rate of model A is higher, then model A is better than model B.
3. A convex hull is drawn conceptually by stretching a rubber band around the points so that all of the points lie within the graph. Models which lie under the convex hull are sub-optimal models and models which lie on the convex hull are the optimal ones.
4. In the case of multiple models on the convex hull, best model is selected according to the application. If FP rate is very critical for the application, then the model with least FP rate should be selected. However, if it is crucial to obtain high TP rate, then the model with highest TP rate should be selected.

In table 3.4, TP and FP rates of models are given and figure 3.2 shows the ROC convex hull for five models used for credit scoring on German Credit data set. As seen in the figure, there are two models on the convex hull: SVM and logistic regression. The other three models (MLP, RBF, kNN) are sub-optimal models and eliminated. In our research, TP classification means correct prediction of a good borrower (granting credit to an actual good borrower) and FP classification means false prediction of a good borrower (granting credit to an actual bad borrower).

The ROC curve shows TP and FP rates for different models, but there is no general rule for choosing the best (optimal) model. What we can generalize is only this: "When the ROC convex hull is drawn, points below the hull are sub-optimal models while points on the hull are optimal.". The nature of the application should be considered when selecting the most appropriate model among the optimal ones. For example, if commercial credit is taken into account, the amount of loans are significantly higher. Default of a borrower (which might be a company) may cause big losses for the lender. Therefore,the lender might prefer granting credit to actual bad borrowers as low as possible, even at the cost of loosing some good customers. This is translated to a selection of the optimal model with lowest FP rate. However, if you consider retail credits, the situation might reverse. A retail lender may want to have as many good borrowers as possible because the more good customers are granted credit, lender will earn more revenue. In this case, the lender might prefer choosing a model with a higher TP rate. Therefore, the nature of the application is very critical in deciding the final and most optimum model to use.

According to the results, SVM has higher TP rate than logistic regression (which is the other optimal model) so it seems to be more suitable for the case in this study, since German credit data set is a retail credit data set.

Table 3.4. TP and FP Rates (10-fold cross validation)

Model	**TP rate**	**FP rate**
SVM	% 89.0	% 55.0
RBF	% 89.3	% 58.5
MLP	% 87.2	% 56.3
kNN	% 79.4	% 74.0
Logistic Regression	% 88.3	% 54.0

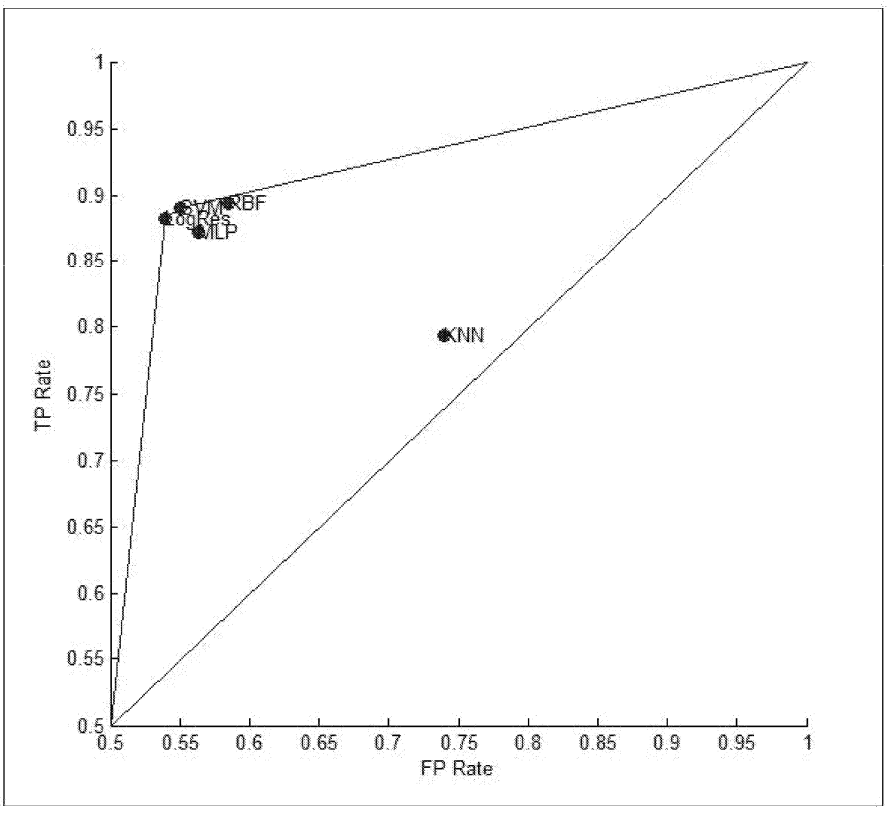

Figure 3.2. ROC Analysis (10-fold cross validation)

4. SVM-REJECT

4.1. Idea and Theory

After finding the maximum separating hyper-plane in high dimensional space, the classification rule for SVM is as follows:

$$c = \begin{cases} 1 & \text{if } w.\phi(x) + b \geq 0 \\ -1 & \text{if } w.\phi(x) + b \leq 0 \end{cases}$$

In order to increase the accuracy of classification on German Credit data set, a new methodology is proposed in this section which originates from the following question: "What percent of the misclassified instances lie close to the separating hyper-plane". If this percentage is high, it is not worth to classify these instances. A critical region is defined as the space between two boundary hyperlanes (see equations 4.1, 4.2). Figure 4.1 gives a better understanding of the critical region.

$$w.\phi(x) + b = t \quad \{boundary\ hyperplane\ 1\} \tag{4.1}$$

$$w.\phi(x) + b = -t \quad \{boundary\ hyperplane\ 2\} \tag{4.2}$$

To answer this question, an experiment was performed. German credit data was divided into training and validation data sets which contained 490 good borrowers, 210 bad borrowers and 210 good borrowers, 90 bad borrowers sequentially. By this way, thirty percent of bad borrower ratio in the complete data set was also obtained in both training and validation data sets. SVM model was formed by using the training data, critical region was defined by using $t = 0.4$ and model tested on validation data. t

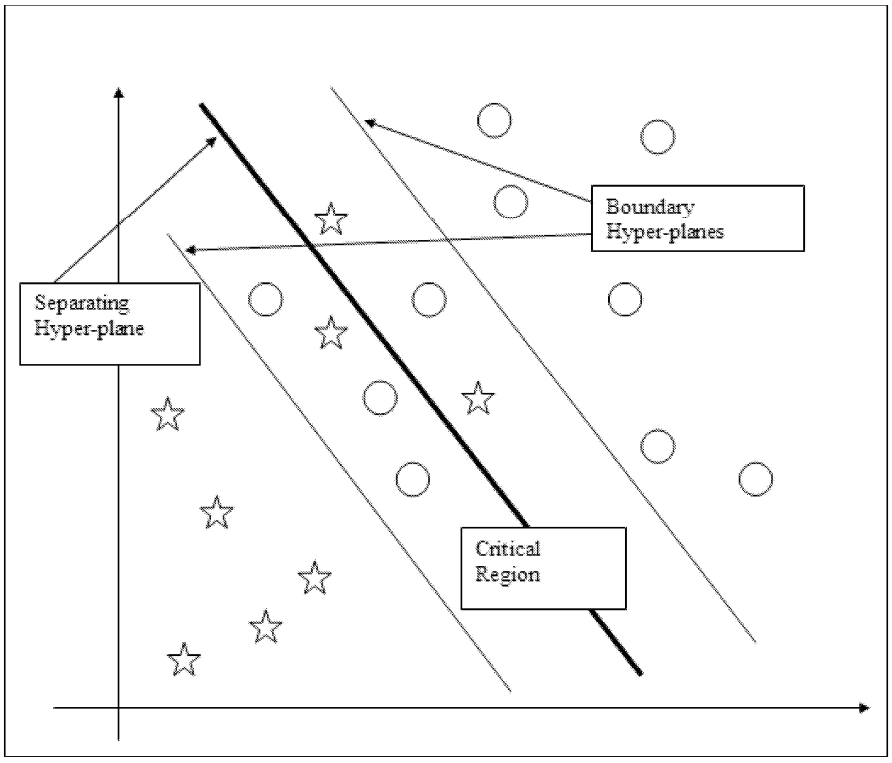

Figure 4.1. Critical Region

value was determined experimentally by trying several values and it is specific to this application. Results can be seen in table 4.1.

Table 4.1. Region Comparison (in-CR and out-CR)

Region type	# False	# True	# Instances	Accuracy	Error rate
CR	35	27	62	43.5 %	56.5 %
non-CR	37	201	238	84.5 %	15.5 %
Overall	72	228	300	76.0 %	24.0 %

As shown in table 4.1, nearly half of the incorrectly predicted data instances (35 of 72) lie in the critical region (CR) which is defined as the area between boundary hyper-planes. Accuracy in the critical region (43.5%) is also very low, however accuracy in the non-critical (non-CR) region is pretty decent with a value of 84.5%. That is, most of the predictions in the critical region are erroneous (56.5 %), so it is risky to trust on these results. With rejecting to classify 20.7 % percent of data instances (62

of 300 instances which lie in CR), 84.5 % of accuracy can be obtained on the classified instances in the non-critical region.

SVM-Reject is a two-layer cascading model which uses SVM in the first layer. SVM classifies the instances which are in the non-critical region and others (which lie in the critical region) are not classified and forwarded to the second layer, logistic regression (see figure 4.2). By this way, rejected instances are also classified by the second layer. In the next section, results of SVM-Reject is compared with SVM and logistic regression.

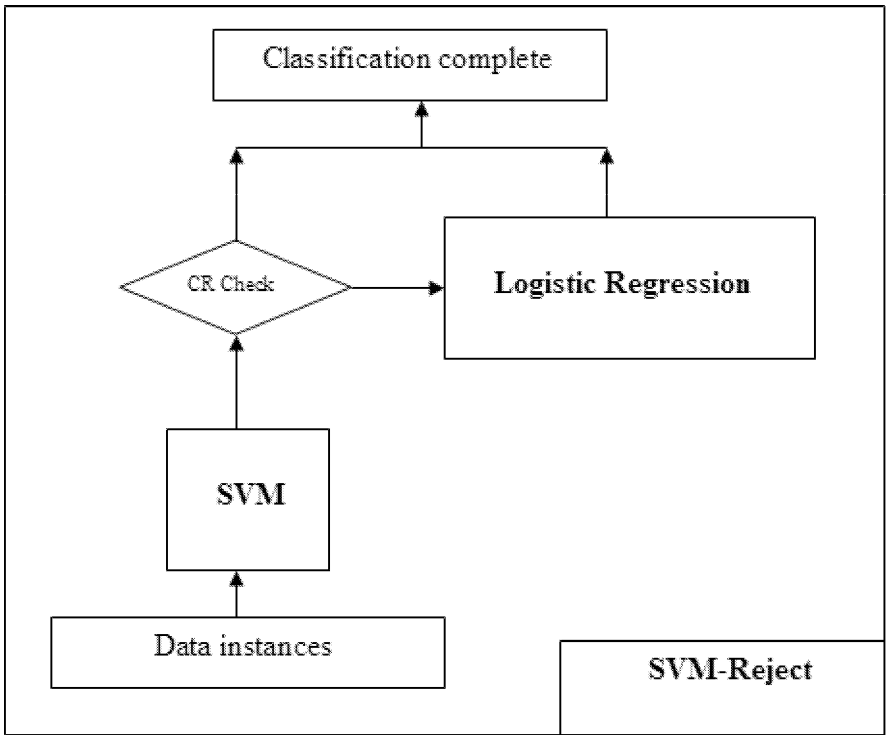

Figure 4.2. SVM-Reject Architecture

4.2. Comparison and Results

4.2.1. Accuracy

Table 4.2 shows the accuricies of three models and SVM-Reject appears as the best accurate model.

Table 4.2. Accuracy comparison of models (SVM-Reject comparison)

Model	**Overall**
SVM-Reject	% 80.3
SVM	% 76.0
Logistic regression	% 78.7

4.2.2. Error Cost

In table 4.3 error cost values of three models are given and SVM-Reject has lower cost value than support vector machines and logistic regression. Error cost was calculated as described in previous chapter.

Table 4.3. Error cost comparison of models (SVM-Reject comparison)

Model	**(C_{good},C_{bad}) = (1,1)**	**(C_{good},C_{bad}) = (1,5)**
SVM-Reject	0.197	0.703
SVM	0.240	0.920
Logistic Regression	0.213	0.747

4.2.3. ROC Analysis

ROC Analysis also shows that SVM-Reject is the optimal model, see figure 4.3. Support vector machines and logistic regression appear as sub-optimal models which lie below the convex hull. TP and FP rates are also available in table 4.4. SVM-Reject has same TP rate as SVM but its FP rate is lower than SVM which makes it the only optimal model.

4.2.4. Results

Three criteria were also used in this part to compare the three models. As a result, it can be stated that SVM-Reject gives better results than SVM and logistic

Table 4.4. TP and FP Rates (SVM-Reject comparison)

Model	**TP rate**	**FP rate**
SVM-Reject	90.0 %	42.2 %
SVM	90.0 %	56.7 %
Logistic Regression	88.6 %	44.0 %

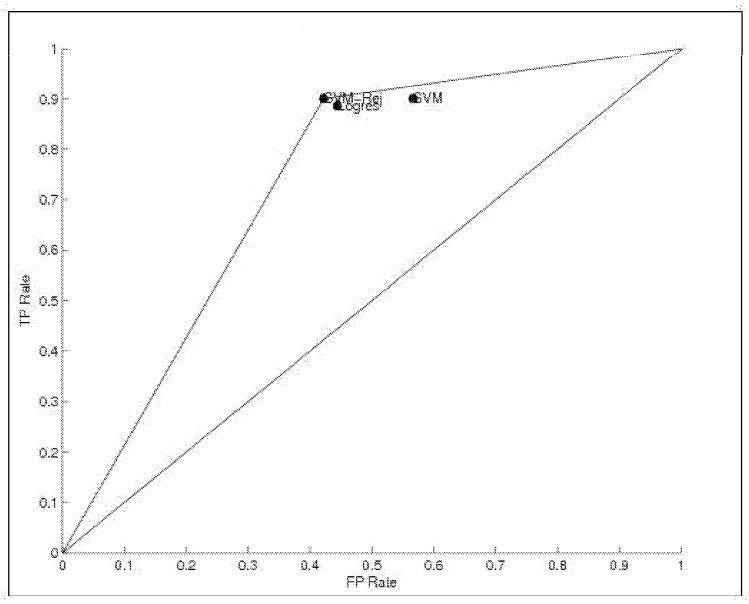

Figure 4.3. ROC Analysis (SVM-Reject comparison)

regression according to all of them. It does not take the risk due to classification of instances which lie in the critical region. These are forwarded to the second layer, logistic regression. There are two reasons for using logistic regression as the second layer. First, it was one of the most accurate models (with SVM) in the first part of this research. Secondly, if the layers of the cascading model come from different theories and algorithms, they will complement each other as stated in [20].

5. CREDIT SCORING: A BANK POINT OF VIEW

5.1. Basel 2 and PD Model

The Basel Committee on Banking Supervision is responsible for proposing capital requirements for internationally active banks. Typically, regulators around the world adopt the guidelines put forth by the committee, even if they are not from one of the 13 nations represented on the committee. The committee first proposed the Basel New Capital Accord, also known as Basel II (see [21]), in December 2001, with revisions in July 2002 and April 2003. More revisions are likely before the final adoption of the accord. By year-end 2006, Basel II is expected to replace the original Basel Accord, which was implemented in 1992. The proposals allow banks to choose among several approaches to determine their capital requirements to cover credit risk. The standardized approach allows less sophisticated banks to use external credit ratings to classify the banks assets into risk classes. Over time, banks are expected to evolve to the internal ratings-based approaches (foundation and advanced), which rely on the banks own experience in determining the risk characteristics of various asset classes. For example, the foundation IRB approach for corporate, sovereign, and bank exposures allows banks to provide estimates of PD but requires banks to use supervisory estimates of LGD, EAD, and maturity. The advanced IRB app roach for such exposures allows banks to provide estimates of PD, LGD, and EAD and requires banks to provide estimates of maturity. The focus of this chapter is to describe the PD model building process and its importance for the banks.

5.2. PD Model Building

Figure 5.2 shows the steps for PD model building. First step is obviously the data collection. Historical data of the bank's customers is extracted from the databases of its systems. There can be many features available for customers, so the number of features (columns) in the data should be decreased to a reasonable number. Several techniques can be used in this stage. Factors can be analyzed in terms of their predictive ability

and the ones which are predictive or intuitive may be selected. Moreover, principal component analysis (PCA) can also be used, but the disadvantage of PCA is that, since components replace the factors (features), it becomes impossible to understand which factors are used in the model which is not intuitive.

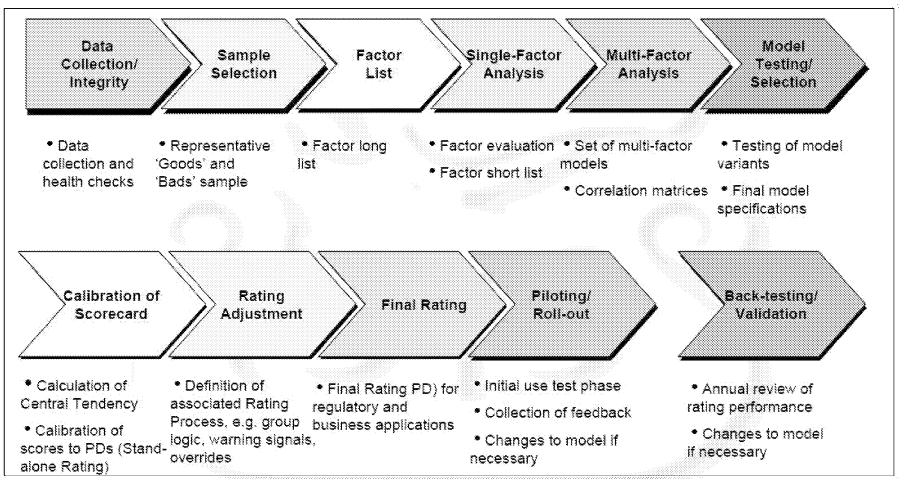

Figure 5.1. Methodology

After reducing the long list of factors into a short list (single factor analysis), correlations of the factors are analyzed. Instead of correlation analysis, cluster analysis may also be performed in order to understand the correlated factors so that they will not be incorporated into a model together. All models with several techniques (SVM, logistic regression, etc.) and different sets of variables are formed in this step (multi-factor analysis). These models are tested and the best one to use is determined. Then the model is calibrated which is calculating PD values from scores by taking the central tendency of the portfolio into account.

5.3. PD Model building with Logistic Regression

The difference of the PD model from the classification is that it assigns a default probability rather than assigning 0 (bad customer) or 1 (good customer). This is the basic reason why logistic regression is so common in PD model building, because it gives the probability directly. As described in previous chapter, logistic regression

model has the following form:

$$logit(p) = \alpha + \beta_1 x_{1,i} + ... + \beta_k x_{k,i} \tag{5.1}$$

After the model outputs the score, it is very easy to obtain the PD of the instance (which corresponds to the customer) by using the following formula:

$$Pr(Y_i = 0|X) = \frac{1}{1 + e^{\alpha+\beta_1 x_{1,i}+...+\beta_k x_{k,i}}} \tag{5.2}$$

5.4. PD Model building with Machine Learning Techniques

Logistic regression is directly related to the probability, so it is very straightforward to form a PD model by using it. This is the main reason of the fact that most of the banks use logistic regression model to form their PD models. However, the output of machine learning models are not directly related to a PD. But, this does not imply that it is impossible to obtain a PD model by using them. With a transformation, these outputs can be turned into a PD value. This is based on the following assumption: "The higher the output, the low the PD will be". In order to check the validity of this assumption, scores of SVM model on test data (obtained in previous chapter) were ranked in ascending order and inserted into buckets. Each bucket consists of 30 instances and the total number of instances is 300. The graph which shows the default rate of each bucket is shown below: Figure 5.2 proves the validity of the assumption.

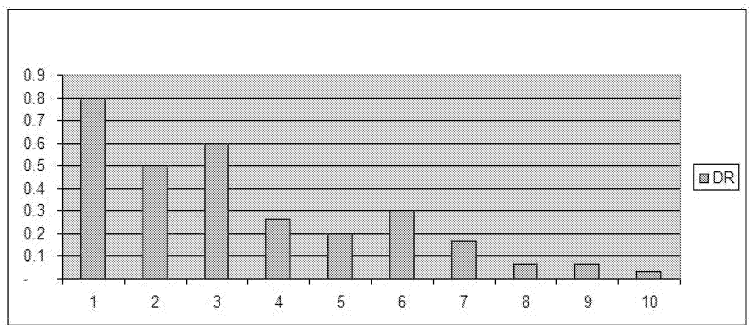

Figure 5.2. Score Buckets vs Default Rate (DR) Histogram

As seen, when the score bucket number increases , default rate of the bucket decreases. At this point, the aim is to find a function which will transform the scores into a PD value between 0 and 1 satisfying the following condition: "PD should decrease when the scores increase". In figure 5.2, PD seems to decrease exponentially with the score, therefore the following function was used for transformation:

$$PD = e^{-ay} + b \tag{5.3}$$

In this thesis, values of a and b were determined experimentally, but generally more advanced techniques are used (such as gini coefficient) for transformation which also considers the central tendency of the portfolio. If value of a is used as $a = 0.2$ and value of b is used as $b = -0.5$, all the outputs are transformed into the interval [0,1] with the following minimum and maximum values: Figure 5.3 shows scores versus

Table 5.1. Output Transformation into PD

Output Type	**Output**	**PD**
min	-1.707182	0.906967107
max	2.717254	0.08074077

PD value distribution of data instances. Also in figure 5.4, both default rate and PD of score buckets are shown where it is clearly seen that the shape of the PD curve is similar to the default rate histogram.

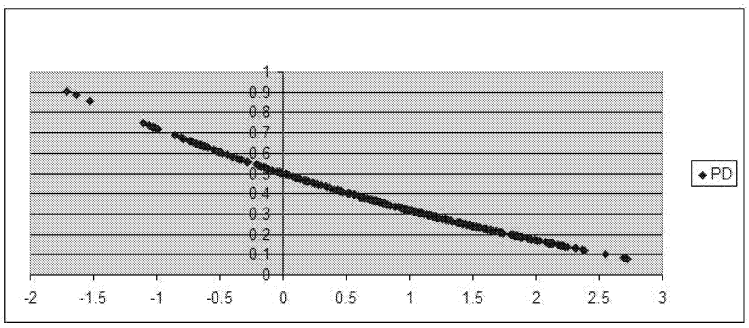

Figure 5.3. Score vs PD Distribution

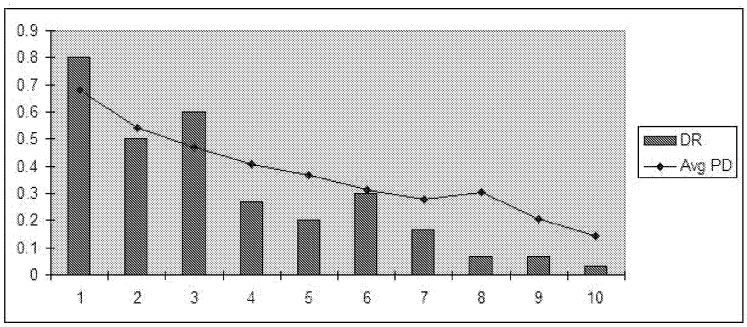

Figure 5.4. Bucket DR (histogram) vs Bucket PD (line) Distribution

5.5. Model Back-testing

PD model is important not just for Basel 2 capital calculations but also for the market share and the profitability of the bank. Banks adjust their margins on the loans according to the PD of their customers. While low PD customers pay low interest rates on their loans, high PD ones pay higher rates. It is important to have an accurate PD model, because if it is overestimating the actual PD of the customers, the market share of the bank may come down due to the rejected customers. On the other hand, if the model is underestimating the actual PD's of its customers, default rates will increase. Therefore, having an accurate PD model is very crucial for the bank.

After some time it is built, the scorecard may not evaluate the risk of borrowers correctly. One of the most important reasons is the change is the business cycle. The business cycle or economic cycle refers to the ups and downs seen somewhat simultaneously in most parts of an economy. The cycle involves shifts over time between periods of relatively rapid growth of output (recovery and prosperity), alternating with periods of relative stagnation or decline (contraction or recession). These fluctuations are often measured using the real gross domestic product. If the model was formed in recession period, the model may overestimate the PD of customers. In the reverse case, if it was formed in expansion period, it will underestimate the PD of borrowers. Therefore, whenever the economic situation changes, the model should be re-calibrated according to the new default rate of the portfolio. Sometimes calibration itself is not enough to revalidate the model. Predictive power of some factors also may change and a predictive factor may become ineffective when determining the PD. So predictive

ability of factors should also be tested. Accordingly, some factors may be taken out and new ones may be incorporated.

As a result, model building is not a one time job. Maintaining the model is more crucial because model performance has important effects on the market share of the bank and the calculation of minimum capital requirements. Therefore, models should be re-calibrated and and factors should be re-analyzed periodically in order to incorporate the changes in the economic situation and human behavior into the model.

6. CONCLUSION

6.1. Summary

This research consisted of three parts. In the first part, the performance of SVM was compared with the other famous machine learning and statistical techniques in terms of credit scoring ability. In the comparison, several metrics were used such as accuracy, error cost and ROC analysis. ROC Analysis is a very useful tool for the comparison of model performances and it showed that SVM and logistic regression are the two optimal models. Model selection for retail and commercial credit applications was also discussed based on the results of ROC analysis. Retail and commercial credit have different characteristics and these should be considered when deciding the correct model to use. In commercial credit applications, default of a borrower might result in significant loss so it is crucial to have a low FP rate (frequency of incorrectly granting credit to a bad borrower) for the model. However, most of the retail applications give their first priority to high revenue; so the model with high TP rate (frequency of correctly granting credit to a good borrower) is more preferible. In our case, SVM appears to be a better model than logistic regression for retail credit since its TP rate is higher and logistic regression is more suitable for commercial credit applications with lower FP rate.

In the second part, a new methodology based on SVM was proposed called SVM-Reject. It is a two-layer cascading model whose first layer is SVM and the second layer is logistic regression. SVM do not classify the instances which are close to the separating hyper-plane (which lie in the defined critical region) and these are forwarded to logistic regression. According to the experiments, SVM-Reject is better than SVM and logistic regression (which appeared as the two optimal models in the first part) in terms of both accuracy and error cost. ROC analysis also shows that SVM-Reject is the optimal model leaving SVM and logistic regression as sub-optimal ones.

Finally, in the third part, credit scoring was discussed from a banks point of view.

The need and the benefits of a PD model were pointed. Maintaining the model is more crucial because model performance has important effects on the market share of the bank and the calculation of minimum capital requirements. Moreover, the steps of the PD model building were also described. In this part, the basic aim was to emphasize that, it is not impossible to obtain a PD model by using techniques other than logistic regression. Machine learning techniques are not directly related to the concept of probability, but by means of a transformation, the PD value can be obtained. The basic reason behind the wide usage of logistic regression in banks is that it is very easy to obtain the PD directly. It was also mentioned that, the model building is not a one time job and requires validation periodically. Changes in the business cycle may cause the model to produce inaccurate results. If the model is overestimating or underestimating the actual default probability of the customers, market share of the bank may decrease or number of defaults may increase.

REFERENCES

1. Brill, J., "The importance of credit scoring models in improving cash flow and collections", *Business Credit*, Vol. 1, pp. 16-17, 1998.

2. Mester, L.J., "What's the point of credit scoring?", *Business Review-Federal Reserve Bank of Philadelphia*, pp. 3-16, Sept/Oct 1997.

3. Reichert, A.K., C.C. Cho and G.M. Wagner, "An examination of the conceptual issues involved in developing credit-scoring models", *Journal of Business and Economic Statistics*, Vol. 1, pp. 101-114, 1983.

4. Henley, W.E., *Statistical aspects of credit scoring*, Dissertation, The Open University, Milton Keynes, UK, 1995.

5. Henley, W.E. and D.J. Hand, "A k-nearest neighbor classifier for assessing consumer credit risk", *Statistician*, Vol 44, pp. 77-95, 1996.

6. Tam, K.Y. and M.Y. Kiang, "Managerial applications of neural networks: the case of bank failure predictions", *Management Science*, Vol. 38, pp. 926-947, 1992.

7. Frydman, H.E., E.I. Altman and D. Kao, "Introducing recursive partitioning for financial classification: the case of financial distress", *Journal of Finance*, Vol. 40, No. 1, pp. 269-91, 1985.

8. Altman, E.I., "Corporate distress diagnosis: comparisons using linear discriminant analysis and neural networks (the Italian experience)", *Journal of Banking and Finance*, Vol. 18, pp. 505-529, 1994.

9. Coats, P.K. and L.F. Fant, "Recognizing Financial distress patterns using a neural network tool", *Financial Management*, pp. 142-155, Autumn 1993.

10. Salchenberger, L.M., E.M. Cinar and N.A. Lash, "Neural networks: a new tool for

predicting thrift failures", *Decision Sciences*, Vol. 23, pp. 899-916, 1992.

11. Haykin, S., *Neural Networks*, Macmillan Publishing Company, NY, USA, 1994.

12. Ripley, B.D., *Pattern Recognition and Neural Networks*, Cambridge University Press, Cambridge, 1996.

13. Bishop, C.M., *Neural Networks for Pattern Recognition*, Oxford University Press, Oxford, 1995.

14. Mitra, U. and H.V. Poor, "Neural Network Techniques for Adaptive Multiuser Demodulation", *IEEE Journal on Selected Areas in Communications*, Vol. 12, No. 9, pp. 1460-1470, 1994.

15. Vapnik, N.V., *Estimation of Dependences Based on Empirical Data*, Springer-Verlag, New York, 1982.

16. Vapnik, N.V., *The Nature of Statistical Learning Theory*, Springer, New York, 1995.

17. Vapnik, N.V. and A.J. Chervonenkis, *Theory of Pattern Recognition*, Nauka, Moscow, 1997.

18. Vapnik, V.N. and A.J. Chervonenkis, "The necessary and sufficient conditions for consistency in the empirical risk minimization method", *Pattern Recognition and Image Analysis*, Vol. 1, No. 3, pp. 283-305, 1991.

19. Chang, C.C. and C.J. Lin, *LIBSVM: a library for support vector machines*, http://www.csie.ntu.edu.tw/cjlin/libsvm.

20. Alpaydin, E., *Introduction to Machine Learning*, The MIT Press, London, England, 2004.

21. *BIS: The new Basel capital accord: Consultative document*, Basel Committee on

Banking Supervision, April 2000.