International Journal of Information Technology & Decision Making
Vol. 8, No. 4 (2009) 677–696

World Scientific
www.worldscientific.com

CREDIT SCORING MODELS WITH AUC MAXIMIZATION BASED ON WEIGHTED SVM

LIGANG ZHOU* and KIN KEUNG LAI†

Department of Management Sciences
City University of Hong Kong, Kowloon Tong, Hong Kong
**mszhoulg@cityu.edu.hk*
†*mskklai@cityu.edu.hk*

JEROME YEN

Department of Finance
Hong Kong University of Science and Technology
Hong Kong

Credit scoring models are very important tools for financial institutions to make credit granting decisions. In the last few decades, many quantitative methods have been used for the development of credit scoring models with focus on maximizing classification accuracy. This paper proposes the credit scoring models with the area under receiver operating characteristics curve (AUC) maximization based on the new emerged support vector machines (SVM) techniques. Three main SVM models with different features weighted strategies are discussed. The weighted SVM credit scoring models are tested using 10-fold cross validation with two real world data sets and the experimental results are compared with other six traditional methods including linear regression, logistic regression, k nearest neighbor, decision tree, and neural network. Results demonstrate that weighted 2-norm SVM with radial basis function (RBF) kernel function and t-test feature weighting strategy has the overall better performance with very narrow margin than other SVM models. However, it also consumes more computational time. In considering the balance of performance and time, least squares support vector machines (LSSVM) with RBF kernel maybe a better choice for large scale credit scoring applications.

Keywords: Credit scoring; AUC; SVM; features weighting.

1. Introduction

In recent financial crisis, many financial institutions suffered high losses from a steady increase in customers' defaults on loan. In USA, general credit cards' issuers wrote off $27.19 billion of debts as losses in 1997 and this figure had increased to $31.91 billion in 2006.[1] In addition, the recent subprime mortgage crisis in USA has caused some companies to lose billions of dollars due to customers' default and caused some credit granting institutions to financial distress. However, the credit

*Corresponding author.

granting institutions cannot refuse credit applications to the growing market solely to avoid credit risk. Therefore, effective credit risk assessment has become a crucial factor for gaining competitive advantages in credit market.

Credit scoring is the most widely used techniques that help lenders to make credit granting decisions. Credit scoring's main idea is to estimate the probability of applicants' default, according to the characteristics recorded in the application form or from credit bureau, by a quantitative model that is built on the basis of information of past applicants.

Many quantitative methods and techniques from different disciplines have been used for building credit scoring models, such as linear discriminant analysis, logistic regression, decision tree, bayes network from statistics; linear programming; artificial neural network, support vector machines from artificial intelligence and some hybrid approaches. Some previous researches have provided summaries about various techniques for credit scoring.[2–4] However, most of the previous researches focus on improving the predictive accuracy, i.e. the percentage of correctly classified testing samples, which is also the most frequently used and simple performance measure for classifiers. In addition, quantitative methods are also used for build scoring models for other applications, such as R&D projects scoring,[5] stock performance scoring,[6] and credit card holders' behavior.[7]

Although predictive accuracy is simple and commonly used to compare different classifiers, it is always based on a single threshold value. However, in practice, the decision makers need a measure that can reflect the relationship between threshold value and classification accuracy of different class for the model that they can adjust their granting strategies in terms of market situation. Moreover, the misclassification cost of classifying a bad customer as good is more than that of classifying a good customer as bad, since the credit granting institutions may lose the whole loan granted to the bad customer and lose only the interest from a missing good customer. The decision makers should be able to choose the proper threshold value for the model to make the balance between possible profit and loss. For above reasons, the area under the Receiver Operating Characteristics (ROC) curve, abbreviated to AUC, is an alternative measure other than predictive accuracy. ROC can provide a set of classifiers from selection of various threshold values and thus the decision makers will be able to choose the appropriate one in terms of their market view or knowledge of misclassification cost on good and bad customers. Then, the credit scoring models should be optimized by maximizing AUC instead of single classification accuracy based on a fixed threshold value.

Support vector machines have become powerful classification methods with a wide range of business application, including credit scoring. Baesens *et al.*[8] mainly discussed the benchmarking study of various classification techniques on eight real-life credit scoring datasets, used SVM and least square SVM (LSSVM) with linear and RBF kernels, and adopted a grid search mechanism to tune the hyperparameters. The experiments show that RBF–LSSVM classifier can yield a good performance in terms of classification accuracy. Huang[9] demonstrates that the SVM-based

model is very competitive to back-propagation neural network (BPN), Genetic programming (GP), and decision tree on the measure of predictive accuracy. Klaus and Ralf[10] considered an adjustment of support vector machines for credit scoring to a set of non-standard situations to practitioner. They mainly focused on the selection of cut-off with considering prior class weights and the parameters in SVM were optimized with regard to expected classification accuracy other than AUC. However, there are reports about some new methods that outperform single SVM model on some datasets testing, such as Multi-criteria convex quadratic programming model,[11] neural network ensemble model,[12] etc.

However, the performance of SVM models is sensitive to the selection of parameters and input variables (features). Chapelle *et al.*[13] discussed the problem of automatically tuning multiple parameters for pattern recognition support vector machines by minimizing some estimated generalization errors of SVMs, using a gradient descent algorithm over the set of parameters, and pointed out that since many real-world databases contain attributes of different natures, there may exist more appropriate scaling factors that assign the right weight to the right feature. Huang *et al.*[9] proposed a hybrid GA–SVM strategy to simultaneously perform the feature selection task and model parameters optimization. Feature selection can be viewed as a special case of feature weighting; it only assigns weights that are either 1 (keep the feature) or 0 (eliminate the feature). Finally selected features for the SVM model can be thought of as important features, while others that have been eliminated are useless features. This strategy classifies the features into only two classes, which cannot show the relative importance of features from the same class, i.e. we cannot tell which feature is the most important, among the selected features, and which feature is the most useless, among the eliminated features. However, the features weighting strategy can overcome this shortcoming.

In this study, genetic algorithm is introduced to optimize the parameters and weights of features in typical SVM models with AUC maximization. This paper is organized as follows: Sec. 2 gives an introduction to weighted SVM models and ROC curve. In Sec. 3 different strategies about features weighting will be introduced. In Sec. 4, empirical study about the proposed models on two real-world data sets will be reported. Finally Sec. 5 is a short conclusion and discussion.

2. Weighted SVM Models

2.1. *Weighted 1-norm SVM*

Given a training set $\{(\boldsymbol{x_1}, y_1), (\boldsymbol{x_2}, y_2), \ldots, (\boldsymbol{x_k}, y_k), \ldots, (\boldsymbol{x_N}, y_N)\}$, where $\boldsymbol{x_k} \in R^m$ and $y_k \in \{-1, +1\}$, and each element $(\boldsymbol{x_k}, y_k)$ is corresponding to a point in a high-dimensional space. Suppose all the points can be separated by a hyperplane $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b = 0$; then it is natural to construct a linear classifier as follows[14]:

$$y(\boldsymbol{x}) = \mathrm{sign}(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b) \tag{1}$$

If all the data of the two classes are separable, one can say that

$$\begin{cases} \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x_k} + b \geq +1, & \text{if } y_k = +1 \\ \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x_k} + b \leq -1, & \text{if } y_k = -1 \end{cases} \tag{2}$$

Above set of inequalities can be transformed into following compact form:

$$y_k(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x_k} + b) \geq 1, \quad k = 1, \ldots, N. \tag{3}$$

The traditional SVM finds an optimal separating hyperplane to maximize the margin subject to the condition that all training data points need to be correctly classified, and it gives the following optimization problem:

$$\begin{aligned} &\underset{\boldsymbol{w},b}{\mathrm{Min}}\, Z = \frac{1}{2}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w} \\ &\text{subject to: } y_k(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x_k} + b) \geq 1, \quad k = 1, \ldots, N. \end{aligned} \tag{4}$$

Usually, for most real-life problems, it is difficult to find a hyperplane that can separate all the data points correctly. Thus, an extension of linear SVM to a non-separable case is to introduce an additional slack variable $\xi_k \geq 0$, which indicates the misclassification errors. The problem of finding an optimal separating hyperplane combining two objectives, maximizing the margin and minimizing the classification errors is formulated as follows:

$$\begin{aligned} &\underset{\boldsymbol{w},b,\xi}{\mathrm{Min}}\, Z = \frac{1}{2}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w} + C\sum_{k=1}^{N}\xi_k \\ &\text{subject to: } y_k(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x_k} + b) \geq 1 - \xi_k, \quad k = 1, \ldots, N \\ &\qquad\qquad\qquad\qquad\qquad\qquad \xi_k \geq 0, \quad k = 1, \ldots, N \end{aligned} \tag{5}$$

where C is a penalty parameter on the training error, which is a positive real constant.

The above linear SVM model has been extended to a nonlinear model which was viewed as an important progress in SVM theory, made by Vapnik. The main idea is to map the training samples in input space to a high dimensional feature space by a nonlinear mapping function $\boldsymbol{\varphi}(\boldsymbol{x})$ and then construct the linear separating hyperplane in this high dimensional feature space. The problem to find optimal hyperplane is as follows:

$$\begin{aligned} &\underset{\boldsymbol{w},b,\xi}{\mathrm{Min}}\, Z = \frac{1}{2}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w} + C\sum_{k=1}^{N}\xi_k \\ &\text{subject to: } y_k[\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\varphi}(\boldsymbol{x_k}) + b] \geq 1 - \xi_k, \quad k = 1, \ldots, N \\ &\qquad\qquad\qquad\qquad\qquad\qquad \xi_k \geq 0, \quad k = 1, \ldots, N \end{aligned} \tag{6}$$

When the number of samples for training from different classes is unbalanced, to obtain robust classifiers, Osuna *et al.*[15] proposed to use different penalty parameters

for different classes in the above SVM formulation. It gives the following form:

$$\begin{aligned} &\underset{\boldsymbol{w},b,\xi}{\text{Min}}\, Z = \frac{1}{2}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w} + C_+\sum_{k=1}^{N_+}\xi_k + C_-\sum_{k=1}^{N_-}\xi_k \\ &\text{subject to: } y_k[\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\varphi}(\boldsymbol{x_k}) + b] \geq 1 - \xi_k, \quad k = 1, \ldots, N \\ &\qquad\qquad\qquad\qquad\qquad\qquad \xi_k \geq 0, \quad k = 1, \ldots, N \end{aligned} \tag{7}$$

where N_+ is the number of samples with $y = +1$ and N_- is the number of samples with $y = -1$, and $N_+ + N_- = N$.

If we think the input space is heterogeneous, i.e. the importance of different input features, for classification is different, and then we need to assign a different feature weight to each feature. If a feature is useless for the classification, its weight is likely to become small. In an extreme case, if the feature weight is small enough, it means that the corresponding feature can be removed without affecting the classification performance. The weighted SVM (WSVM) model is shown as follows:

$$\begin{aligned} &\underset{\boldsymbol{w},b,\xi}{\text{Min}}\, Z = \frac{1}{2}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w} + C_+\sum_{k=1}^{N_+}\xi_k + C_-\sum_{k=1}^{N_-}\xi_k \\ &\text{subject to: } y_k[\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\varphi}(\boldsymbol{\beta}^{\mathrm{T}} \otimes \boldsymbol{x_k}) + b] \geq 1 - \xi_k, \quad k = 1, \ldots, N \\ &\qquad\qquad\qquad\qquad\qquad\qquad \xi_k \geq 0, \quad k = 1, \ldots, N \end{aligned} \tag{8}$$

where $\boldsymbol{\beta}^{\mathrm{T}} = \{\beta_1, \beta_2, \ldots, \beta_m\}$ is the features weighting vector, $\boldsymbol{\beta}^{\mathrm{T}} \otimes \boldsymbol{x_k} = \{\beta_1 x_{1k}, \beta_2 x_{2k}, \ldots, \beta_m x_{mk}\}$.

To solve the problem of Eq. (8), the following Lagrangian function can be constructed:

$$\begin{aligned} \min_{\boldsymbol{w},b,\xi} L(\boldsymbol{w}, b, \xi; \alpha, \lambda) = {} & \frac{1}{2}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w} + C_+\sum_{k=1}^{N_+}(\xi_k | y_k = 1) + C_-\sum_{k=1}^{N_-}(\xi_k | y_k = -1) \\ & - \sum_{k=1}^{N}\alpha_k\{y_k[\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\varphi}(\boldsymbol{\beta}^{\mathrm{T}} \otimes \boldsymbol{x_k}) + b] - 1 + \xi_k\} - \sum_{k=1}^{N}\lambda_k\xi_k \end{aligned} \tag{9}$$

where $\alpha_k, \lambda_k, k = 1, \ldots, N$are non-negative Lagrange multipliers.

The parameters that minimize the Lagrangian must satisfy the following conditions:

$$\begin{cases} \dfrac{\partial L}{\partial \boldsymbol{w}} = 0 \\ \dfrac{\partial L}{\partial b} = 0 \\ \left(\dfrac{\partial L}{\partial \xi_k}\middle| y_k = 1\right) = 0 \\ \left(\dfrac{\partial L}{\partial \xi_k}\middle| y_k = -1\right) = 0 \end{cases} \tag{10}$$

From the above equations, one can derive:

$$\begin{cases} \boldsymbol{w} - \sum_{k=1}^{N} \alpha_k y_k \boldsymbol{\varphi}(\boldsymbol{\beta}^{\mathrm{T}} \otimes \boldsymbol{x_k}) = 0 \rightarrow \boldsymbol{w} = \sum_{k=1}^{N} \alpha_k y_k \boldsymbol{\varphi}(\boldsymbol{\beta}^{\mathrm{T}} \otimes \boldsymbol{x_k}) \\ \sum_{k=1}^{N} \alpha_k y_k = 0 \\ C_+ - \alpha_k - \lambda_k = 0 | (y_k = 1, \lambda_k \geq 0, \alpha_k \geq 0) \\ \quad \rightarrow 0 \leq \alpha_k \leq C_+, \quad \text{if } y_k = +1 \\ C_- - \alpha_k - \lambda_k = 0 | (y_k = -1, \lambda_k \geq 0, \alpha_k \geq 0) \\ \quad \rightarrow 0 \leq \alpha_k \leq C_-, \quad \text{if } y_k = -1 \end{cases} \tag{11}$$

Then, we can get the dual form of problem Eq. (8):

$$\begin{aligned} &\max L(\alpha_k) = -\frac{1}{2} \sum_{k,l=1}^{N} \alpha_k \alpha_l y_k y_l K(\boldsymbol{\beta}^{\mathrm{T}} \otimes \boldsymbol{x_k}, \boldsymbol{\beta}^{\mathrm{T}} \otimes \boldsymbol{x_l}) + \sum_{k=1}^{N} \alpha_k \\ &\text{Subject to} \\ &\quad \sum_{k=1}^{N} \alpha_k y_k = 0 \\ &\quad 0 \leq \alpha_k \leq C_+, \quad \text{if } y_k = +1 \\ &\quad 0 \leq \alpha_k \leq C_-, \quad \text{if } y_k = -1 \end{aligned} \tag{12}$$

where function $K(u, v) = \boldsymbol{\varphi}(u)^{\mathrm{T}} \cdot \boldsymbol{\varphi}(v)$ is the kernel function. Based on the solution of the above quadratic programming problem, the nonlinear SVM classifier is constructed as follows:

$$y(\boldsymbol{x}) = \operatorname{sign}\left[\sum_{k=1}^{N} \alpha_k y_k K(\boldsymbol{\beta}^{\mathrm{T}} \otimes x, \boldsymbol{\beta}^{\mathrm{T}} \otimes \boldsymbol{x_k}) + b\right] \tag{13}$$

It is widely acknowledged that parameters selection is a very important factor that affects the performance of SVM. For the WSVM, there are multiple parameters that need tuning at the same time, so it will cause a problem of curse of the dimensionality for the DOE and direct search methods. Thus the DOE or direct search may not be a good choice, because of the huger computational time required. From the point of theoretical analysis, if the performance of the SVM classifier can be formulated explicitly with these parameters, the optimal parameters can be defined exactly. However, it is always a challenging problem to get explicit performance function of the parameters. Chapelle *et al.*[13] proposed a method that cannot only find the hyperplane which maximizes the margin but also the values of the mapping parameters that yield lowest generalization errors, by a standard gradient descent approach. Some other studies on parameters selection focus on searching strategy, which is based on performance feedback.[9,16]

2.2. *Weighted 2-norm SVM*

The weighted 2-norm SVM model is defined as follows:

$$\begin{aligned} &\underset{\boldsymbol{w},b,\xi}{\text{Min}}\, Z = \frac{1}{2}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w} + \frac{C_+}{2}\sum_{k=1}^{N_+}\xi_k^2 + \frac{C_-}{2}\sum_{k=1}^{N_-}\xi_k^2 \\ &\text{subject to: } y_k[\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\varphi}(\boldsymbol{\beta}^{\mathrm{T}}\otimes\boldsymbol{x_k}) + b] \geq 1 - \xi_k, \quad k = 1, \ldots, N \end{aligned} \tag{14}$$

when compared with 1-norm SVM, it can be noted that the condition $\xi_k \geq 0$ is removed, as if $\xi_k < 0$, it can be set to zero and the objective function is further optimized.

The primal Lagrangian for weighted 2-norm SVM problem is as follows:

$$\begin{aligned} \min_{w,b,\xi} L(\boldsymbol{w}, b, \xi; \alpha, \lambda) = {} & \frac{1}{2}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w} + \frac{C_+}{2}\sum_{k=1}^{N_+}(\xi_k^2\,|y_k = 1) + \frac{C_-}{2}\sum_{k=1}^{N_-}(\xi_k^2\,|y_k = -1) \\ & - \sum_{k=1}^{N}\alpha_k\{y_k[\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\varphi}(\boldsymbol{\beta}^{\mathrm{T}}\otimes\boldsymbol{x_k}) + b] - 1 + \xi_k\} \end{aligned} \tag{15}$$

The paddle point must meet the following conditions:

$$\begin{cases} \dfrac{\partial L}{\partial \boldsymbol{w}} = 0 \rightarrow \boldsymbol{w} = \displaystyle\sum_{k=1}^{N}\alpha_k y_k \boldsymbol{\varphi}(\boldsymbol{\beta}\otimes\boldsymbol{x_k}) \\ \dfrac{\partial L}{\partial b} = 0 \rightarrow \displaystyle\sum_{k=1}^{N}\alpha_k y_k = 0 \\ \dfrac{\partial L}{\partial \xi_{k|y_{k=1}}} = 0 \rightarrow \alpha_{k|y_{k=1}} = C_+\xi_{k|y_{k=1}} \\ \dfrac{\partial L}{\partial \xi_{k|y_{k=-1}}} = 0 \rightarrow \alpha_{k|y_{k=-1}} = C_-\xi_{k|y_{k=-1}} \end{cases} \tag{16}$$

Substituting the above equations into Eq. (15), the dual form can be obtained as follows:

$$\begin{aligned} \max L(\alpha_k) = {} & -\frac{1}{2}\sum_{k,l=1}^{N}\alpha_k\alpha_l y_k y_l K(\boldsymbol{\beta}^{\mathrm{T}}\otimes\boldsymbol{x_k}, \boldsymbol{\beta}^{\mathrm{T}}\otimes\boldsymbol{x_l}) + \sum_{k=1}^{N}\alpha_k \\ & -\frac{1}{2C_+}\sum_{k=1}^{N_+}\alpha_k^2 - \frac{1}{2C_-}\sum_{k=1}^{N_-}\alpha_k^2 \\ & \begin{cases} \text{Subject to} \quad \alpha_k \geq 0 \\ \displaystyle\sum_{k=1}^{N}\alpha_k y_k = 0 \end{cases} \end{aligned} \tag{17}$$

The above quadratic programming problem can be solved to get α_k and the classifier is in the same form as 1-norm.

2.3. *Weighted least square SVM*

The least square support vector machine was proposed by Suykens *et al.*[17] The optimization problem to find the optimal hyperplane that can separate the two different classes with maximum margin in the feature spaces for WLSSVM is as follows:

$$\begin{aligned} &\underset{\boldsymbol{w},b,\xi}{\text{Min}}\, Z = \frac{1}{2}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w} + \frac{C}{2}\sum_{k=1}^{N} v_k \xi_k^2 \\ &\text{subject to: } y_k\left[\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\varphi}(\boldsymbol{x_k}) + b\right] = 1 - \xi_k, \quad k = 1, \ldots, N \end{aligned} \tag{18}$$

where v_k is the weighting factor for sample k on the training error, which is a positive real constant.

To solve it, the following Lagrangian function can be constructed:

$$L(\boldsymbol{w}, b, \xi; \alpha) = \frac{1}{2}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w} + \frac{C}{2}\sum_{k=1}^{N} v_k \xi_k^2 - \sum_{k=1}^{N} \alpha_k \left\{ y_k \left[\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\varphi}(\boldsymbol{x_k}) + b\right] - 1 + \xi_k \right\} \tag{19}$$

where α_k's are the Lagrange multipliers. The following conditions for optimality as following should be met:

$$\begin{cases} \dfrac{\partial L}{\partial w} = 0 \rightarrow \boldsymbol{w} = \displaystyle\sum_{k=1}^{N} \alpha_k y_k \boldsymbol{\varphi}(\boldsymbol{x_k}) \\ \dfrac{\partial L}{\partial b} = 0 \rightarrow \displaystyle\sum_{k=1}^{N} \alpha_k y_k = 0 \\ \dfrac{\partial L}{\partial \xi_k} = 0 \rightarrow \alpha_k = C v_k \xi_k, \quad k = 1, \ldots, N \\ \dfrac{\partial L}{\partial \alpha_k} = 0 \rightarrow y_k \left[\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\varphi}(\boldsymbol{x_k}) + b\right] - 1 + \xi_k = 0, \quad k = 1, \ldots, N. \end{cases} \tag{20}$$

Letting

$$\begin{gathered} \boldsymbol{\alpha}^{\mathrm{T}} = \{\alpha_1, \ldots, \alpha_N\}, \quad \boldsymbol{y}^{\mathrm{T}} = \{y_1, \ldots, y_N\}, \quad \boldsymbol{\xi} = \{\xi_1, \ldots, \xi_N\}, \\ \boldsymbol{E}^{\mathrm{T}} = \underbrace{\{1, \ldots, 1\}}_{N}, \quad \boldsymbol{U}^{\mathrm{T}} = \left\{\boldsymbol{\varphi}(\boldsymbol{x_1})^{\mathrm{T}} y_1, \ldots, \boldsymbol{\varphi}(\boldsymbol{x_N})^{\mathrm{T}} y_N\right\}, \\ V_C = \begin{Bmatrix} \frac{1}{Cv_1} & 0 & \cdots & 0 \\ 0 & \frac{1}{Cv_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{Cv_N} \end{Bmatrix} \end{gathered}$$

and eliminating $\boldsymbol{w}$ and ξ, one can get the following linear system[17]:

$$\begin{bmatrix} 0 & \boldsymbol{y}^{\mathrm{T}} \\ \boldsymbol{y} & \boldsymbol{\Omega} + V_{\mathrm{C}} \end{bmatrix} \begin{Bmatrix} b \\ \boldsymbol{\alpha} \end{Bmatrix} = \begin{Bmatrix} 0 \\ \boldsymbol{E} \end{Bmatrix} \tag{21}$$

where $\boldsymbol{\Omega} = \boldsymbol{U}\boldsymbol{U}^{\mathrm{T}}$ with element

$$\Omega_{kl} = y_k y_l \boldsymbol{\varphi}(\boldsymbol{x_k})^{\mathrm{T}} \boldsymbol{\varphi}(\boldsymbol{x_l}) = y_k y_l K(\boldsymbol{x_k}, \boldsymbol{x_l}), \quad k, l = 1, \ldots, N \tag{22}$$

and $K(\cdot, \cdot)$ is the kernel function.

The classifier of WLSSVM in the dual space is as follows:

$$\begin{aligned} f(\boldsymbol{x}) &= \sum_{k=1}^{N} \alpha_k y_k K(\boldsymbol{x}, \boldsymbol{x_k}) + b \\ y(\boldsymbol{x}) &= \operatorname{sign}\left[f(\boldsymbol{x})\right] \end{aligned} \tag{23}$$

2.4. *Area under ROC curve*

Formally, for credit risk assessment problem, each instance I is mapped to one element of the set $\{G, B\}$ of good and bad class labels. The weighted LS-SVM classification model produces a score which indicates the probability that an instance will be good. Different thresholds can be applied to the score to predict class membership. Let the labels $\{g, b\}$ illustrate the class predictions produced by a given classifier, and given an instance, the four possible outcome of the classifier can be shown with a matrix in Fig. 1.

Usually, there are three common performance measures for classifier:

$$\text{Sensitivity} = \frac{Gg}{Gg + Gb}$$

$$\text{Specificity} = \frac{Bb}{Bg + Bb}$$

$$\text{Predictive accuracy} = \frac{Gg + Bb}{Gg + Gb + Bg + Bb}$$

The ROC curve is a two-dimensional measure of binary classification performance[18,19] in which sensitivity is plotted on the Y axis and 1-Specificity is plotted on the X axis. Figure 2 shows an example of ROC curve and the thresholds curve. The diagonal line corresponds to the ROC curve of a classifier that predicts the class at random and the performance improves the curve further near to the upper left corner of the plot. AUC is the area under ROC curve; if it is equal to 1, the classifier achieves perfect accuracy, which means that the sensitivity and

		Observed class	
		G	*B*
Predicted class	*g*	*Gg*	*Bg*
	b	*Gb*	*Bb*

If the instance is good and it is classified as good, it is counted as a Gg; similarly for Gb, Bg and Bb.

Fig. 1. Confusion matrix.

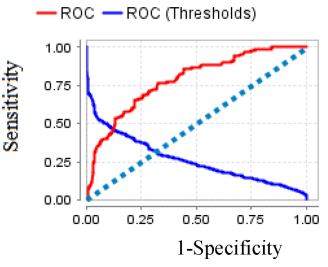

Fig. 2. Example of ROC curve.

the specificity of the model is equal to 1 if the threshold is correctly chosen; if it is equal to 0.5, it means that the classifier has no discriminative power at all. Each point on the ROC curve corresponds to a threshold value which makes a balance between the performance of sensitivity and specificity.

AUC depicts a general behavior of the classifier since it is independent to the threshold used for obtaining a class label. For the weighted LS-SVM classifier, the AUC can be calculated using the following formula[18]:

$$\text{AUC} = \frac{\sum_{i=1}^{n^+} \sum_{j=1}^{n^-} 1_{f(x_i^+) > f(x_j^-)}}{n^+ n^-} \tag{24}$$

where x^+ and x^-, respectively, denote the positive and negative samples and n^+ and n^- are respectively, the number of positive and negative examples.

$$1_{f(x_i^+) > f(x_j^-)} = \begin{cases} 1, & \text{if } f(x_i^+) > f(x_j^-) \\ 0, & \text{otherwise} \end{cases} \tag{25}$$

3. Features Weighting Strategies

There are many different strategies to conduct features weighting, such as traditional GA, t-test and entropy based features weighting and new method based on least squares support feature machine introduced by Li *et al.*[20] In this study, only the traditional strategies are employed.

3.1. *Optimized by genetic algorithm*

Suppose the number of input features of the training dataset is m, then the features of weight vector is $\beta^{\mathrm{T}} = \{\beta_1, \beta_2, \ldots, \beta_m\}$. Let the number of control parameters in kernel function $K_\theta(\cdot, \cdot)$ be n_p, where $\theta = \{\theta_1, \ldots, \theta_{n_p}\}$. For the WSVM, the parameters that need to be determined are β, θ, C_+ and C_-, the number of the parameters is $m + n_P + 2$. For the WLSSVM, the parameters that need to be determined are $\beta, \theta, C, v_1, \ldots, v_N$, the number of the control parameters is $m + n_P + 1 + N$, where N is the number of training samples. In the WLSSVM model, if all $m + n_P + 1 + N$ parameters are optimized in terms of a chosen performance

measure, the model will easily get over the problem of over fitting. In this study, parameters $v_1, \ldots, v_N$ will be excluded from the set of parameters that need to be optimized, and will be determined by analysis of the training dataset or risk manager's personal experience.

There are few literatures on credit scoring modeling with GA based SVM. Huang *et al.*[9] proposed a hybrid GA-SVM model that can simultaneously perform features selection and parameters optimization for credit scoring modeling, but the relative importance of the features cannot be determined by the GA-SVM model directly. To show the relative importance of the selected features, they use the frequency of the selected features, based on the 10-fold cross validation. The features' relative importance vector obtained from this method cannot reflect the importance of features in the SVM model precisely since the selected features vectors for each cross validation may vary widely with the vector of frequency of the selected features. In our model, the relative importance is found directly by the GA process, which is guided by the performance of SVM model with selected business objectives on the credit dataset.

Several key issues about the GA for optimizing parameters in weighted SVM models (GA-WSVM) are discussed as follows:

(1) ***The structure of chromosomes***

In our GA-WSVM model, the binary genetic algorithm will be adopted. In the binary GA, each variable that needs to be optimized is denoted by a series of binary bits with value 1 or 0. Each variable corresponds to a gene and the set of all variables forms the chromosome. The binary GA works with bits. However, the fitness function in GA is always defined on the variables, hence, there must be a way to convert the chromosome into a continuous variable, and vice versa. The length of a gene is determined by the precision requirement of its corresponding variable. A gene can be denoted by $[b_1 b_2 \cdots b_{N_g}]$, where $b_i \in \{0, 1\}$ and N_g is the length of the gene. For example, if the precision requirement for one real variable is 0.1 and the maximum value is 100, there should be at least four bits to express the decimal part and seven bits to express the integer part of the real number, and thus the total length of the gene of this variable should be at least 11.

In GA-WSVM model, each feature weight will have the same precision and the length of the gene for each feature is N_{gf}. The length of genes of all parameters in the SVM model, such as error penalty constant C and σ in RBF kernel function, are set at the same value N_{gp}, with precision requirement of $1/2^{N_{\mathrm{gd}}}(N_{\mathrm{gd}} < N_{\mathrm{gp}})$. Thus the length of bits that express the integer part of the parameters is $N_{\mathrm{gp}} - N_{\mathrm{gd}}$. The chromosome is the array of gene. The structure of the chromosome in GA-WSVM is as follows:

$$\underbrace{\Big[\underbrace{b_1^1 b_2^1 \cdots b_{N_{\mathrm{gf}}}^1}_{\mathrm{gene}_{F1}} \underbrace{b_1^2 b_2^2 \cdots b_{N_{\mathrm{gf}}}^2}_{\mathrm{gene}_{F2}} \cdots \underbrace{b_1^{n_F} b_2^{n_F} \cdots b_{N_{\mathrm{gf}}}^{n_F}}_{\mathrm{gene}_{Fn_F}}}_{m \text{ features}} \underbrace{\underbrace{\dot{b}_1^1 \dot{b}_2^1 \cdots b_{N_{\mathrm{gp}}}^1}_{\mathrm{gene}_{p1}} \cdots \underbrace{\dot{b}_1^{n'_p} \dot{b}_2^{n'_p} \cdots \dot{b}_{N_{\mathrm{gp}}}^{n'_p}}_{\mathrm{gene}_{Pn'_p}} \Big]}_{n'_p \text{ parameters}}$$

where n'_p is the number of total parameters in the support vector machines model, m is the number of input features, b_i^k is the ith bit of gene of the kth feature, and $\dot{b}_i^k$ is the ith bit of gene of the kth parameter. The value of the feature weight is constrained in the range of $[0, 1]$; it is decoded from the chromosome as follows:

$$\beta_i = \frac{b_1^i \times 2^{(N_{\text{gf}}-1)} + b_2^i \times 2^{(N_{\text{gf}}-2)} + \cdots + b_{N_{\text{gf}}}^i \times 2^0}{2^{N_{\text{gf}}} - 1}$$

The parameters are decoded for genes as follows:

$$\theta_i = \dot{b}_1^i \times 2^{(N_{\text{gp}}-N_{\text{gd}}-1)} + \dot{b}_2^i \times 2^{(N_{\text{gp}}-N_{\text{gd}}-2)} + \cdots + \dot{b}_{(N_{\text{gp}}-N_{\text{gd}})}^i \times 2^0 + \dot{b}_{(N_{\text{gp}}-N_{\text{gd}}+1)}^i \times 2^{-1} + \cdots + \dot{b}_{N_{\text{gp}}}^i \times 2^{-N_{\text{gd}}}$$

For example, let $N_{\text{gf}} = 5$ and the gene of fourth feature will be [10101], and the weight is $(1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0)/(2^5 - 1) \approx n'_p 0.677$. If $N_{\text{gp}} = 10$ and $N_{\text{gd}} = 5$, and the gene of the parameter is [1011001010], then the value of this parameter is 21.3125. The parameters in the model have the same mechanism to code and decode the gene.

(2) *Fitness function*

Fitness function is a function of a chromosome whose value is always used to measure the performance of a group of parameters that the chromosome corresponds to. Fitness function can be chosen in terms of needs of business objectives. The fitness function is the classification error of the SVM model with corresponding parameters in a chromosome if the business objective is to maximize the classification accuracy. In addition, to consider the objective to minimize the total cost of misclassifications, i.e. costs from classifying bad as good and good as bad, the fitness function can be defined as the cost of classification on a set of validation samples by the SVM model with parameters defined in chromosome. In this study, the fitness function is defined by the average AUC on 5-fold cross validation on training data set.

(3) *Stop criteria*

The stop criteria are used to determine when the evolutionary process will be stopped. The common stop criteria include[21]: (1) when the number of generations that the GA produces exceeds the predetermined number MAXGEN; (2) when the interval of computational time during which there is no improvement in the best fitness value, in seconds, exceeds specified value TNOIMPROVE; (3) when weighted average change in the fitness function value, over STALLGEN generations, is less than the specified function tolerance FTOL. There are some other stop criteria which can be defined on some indicators from the fitness function. The stop criteria control the computational time and efficiency of the genetic algorithm. Most of this parameters in this criteria are chosen by users' experience. Fortunately, performance of the GA is not sensitized to small or even large deviations of these parameters, in a wide value range.

With above setting, the GA-WSVM algorithm can be described as follows:

Algorithm GA-WSVM

Suppose the samples set is $\boldsymbol{S} = \{(xo, y_1), (xt, y_2), \ldots, (xN, y_N)\}$, and the size of the samples set is N. Then the algorithm for building a hybrid classifier based on GA and WSVM is as follows:

1. Randomly divide N samples into k subsets evenly for k-fold cross validation, after each class of samples is evenly divided, each subset is denoted by I_m, $m = 1, \ldots, k$, let $r = 1$.
2. While $r \leq k$, divide the samples set into two parts, training set and test set, sample indices set for test S_{test} is I_r and sample indices set for training set S_{training} is $\bar{I}_r = I - I_r$, then do feature weighting and parameters optimization with GA as follows:
   - 2.1. Initialize the populations of chromosomes which encode the weight for each input feature, and the parameters for the SVM model;
   - 2.2. For each chromosome in the population,
     - 2.2.1. Decode the feature weights and parameters for SVM model, from the chromosome, and get β, θ, C_+, and C_-, training sample indices set $\bar{I}_r$ is divided into k' subsets evenly for k'-fold cross validation, for constructing the fitness function with SVM model, each subset of sample indices is denoted by $\bar{I}_r^h$, $h = 1, \ldots, k'$, let $r' = 1$;
     - 2.2.2. While $r' \leq k'$,
       select all samples with index in set $\bar{I}_r^{r'}$ to construct the validation sample set $S'_{\text{validation}}$, with all the samples with index in set $\bar{I}_r - \bar{I}_r^{r'}$ for forming the training sample set S'_{training}, using S'_{training} to train the SVM model with parameters β, θ, C and using $S'_{\text{validation}}$ to validate the performance of the SVM model, and denote the performance by $f'_{r'}$;
       $$r' = r' + 1.$$
     - 2.2.3. Average value of the performance of $f'_i, ; i = 1, \ldots, k'$ is taken as the fitness of the chromosome.
   - 2.3. Evolutionary processing: select the elite chromosomes having higher fitness values and pass them to the next generation directly, perform mutation and crossover operation on the population, and generate a new population.
   - 2.4. Go back to Step 2.2, until the stop criteria are met.
3. Optimization parameters are $\beta^r = \beta, \theta^r = \theta, C^r = C$, from training set S_{training}, performance of SVM with these optimization parameters on S_{test} is f_r, $r = r+1$, go to Step 2.
4. Overall performance value is $\sum_{i=1}^{k} f_i/k$, the relative importance of each feature can be evaluated as $\bar{\beta} = \sum_{i=1}^{k} \beta^i/k$.

3.2. *T-test based features weighting*

Suppose $v_j^{+1} = \text{var}(x_{ij}|y_i = +1)$, $v_j^{-1} = \text{var}(x_{ij}|y_i = -1)$, where $\text{var}(\cdot)$ is the variance of a group of values, $m_j^{+1} = \text{mean}(x_{ij}|y_i = +1)$, $m_j^{-1} = \text{mean}(x_{ij}|y_i = -1)$, where $\text{mean}(\cdot)$ is the mean of a group of values, features weighting strategy based on t-test on the training dataset is defined as following[22]:

$$z_j = \frac{\left|m_j^{+1} - m_j^{-1}\right|}{\sqrt{\frac{v_j^{+1}}{n^+} + \frac{v_j^{-1}}{n^-}}} \tag{26}$$

Then, the weight for feature j is defined as follows:

$$w_j = \frac{z_j}{\sum_{k=1}^{m} z_k}, \quad j = 1, 2, \ldots, m \tag{27}$$

3.3. *Entropy based features weighting*

Features weight strategy based on entropy is defined as following[23]:

$$z_j = \frac{\frac{v_j^{+1}}{v_j^{-1}} + \frac{v_j^{-1}}{v_j^{+1}} - 2}{2} + \frac{(m_j^{+1} - m_j^{-1})^2\left(\frac{1}{v_j^{+1}} + \frac{1}{v_j^{-1}}\right)}{2} \tag{28}$$

Then the weight for each feature j is defined by Eq. (27).

4. Empirical Study

In this section, the experimental results on two credit data sets from UCI Machine Learning Repository database will be presented. The experiments are to make a comparison among different SVM models and six commonly used models.

One data set is from German, it consists of 700 instances of creditworthy applicants and 300 instances of noncreditworthy applicants. For each instance has 20 features including status of existing account, credit history, loan purpose, credit amount, employment status, etc. All nominal variables are transferred into binary variables and ordinal and continuous variables are kept. The original dataset with 19 attributes denoted by symbols is transformed into a numerical data set with 24 numerical variables.

Another one is Australian credit dataset, each instance contains 14 attributes. To protect confidentiality of data, names and values have been transferred to meaningless symbolic data, including the observed class of the instances. There are 383 instances of class 1, which is supposed to be non-default applicants here and 307 instances of class 2, which is supposed to be default applicants. In the experiments, all symbolic data are transferred into numerical data for computation.

All input attributes are linear scaled to [0, 1] as to avoid the dominance of attributes with greater numerical values over those with small values.

$$x_{ij} = \frac{x_{ij} - \min_{k\in\{1,\ldots,N\}} x_{kj}}{\max_{k\in\{1,\ldots,N\}} x_{kj} - \min_{k\in\{1,\ldots,N\}} x_{kj}}, \quad i = 1, \ldots, N, \quad j = 1, \ldots, m \tag{29}$$

The parameters in GA searching process mentioned in GA-SVM are set as follows: TNOIMPROVE = 50; MAXGEN = 50; STALLGEN = 50, other parameters without special setting are set to the default value in matlab.[18]

Linear regression (LR) and logistic regression (LOG) need no parameters tuning. Decision tree (DT) uses CART model and the minimum number of observations in impure nodes that need to be further split into 10 and pruned after tree construction. The structure of BP neural network (BPNN) is designed by trial and error. The number of iterations in training BP neural network is set to be 200. The number of nodes in MLPSig, in the hidden layer, is 4. Adaboost is implemented with modest AdaBoost algorithm proposed by Vezhnevets and Vezhnevets[24] in GML AdaBoost Matlab Toolbox. For 1-norm SVM, we use the toolbox LibSVM,[25] and for least square SVM, the LSSVM proposed by Suykens *et al.*[17] is adopted. The parameters in SVMs are optimized by the GA method, with fitness function defined on AUC of 5-fold cross validation on the training dataset. For the KNN10 model set the k to be 10.

The results of 10-fold cross validation on German and Australian datasets from different methods are shown as Tables 1 and 2, respectively. The number in bold means it is the largest one among its corresponding column and that with underline means it is the largest one among its corresponding row. From the results, it can be observed that with the same features weighting strategy, SVM model with RBF kernel function perform better than linear kernel function in most cases with a very small gap no more than 1%. For both the two datasets, it seems that 2-Norm SVM and LS-SVM model with RBF kernel stably keep good performance, however, 2-Norm SVM consumer more time since it needs to solve a large scale quadratic programming problem while LS-SVM only needs to solve linear equations. For some types of SVM model, features weighing strategy seems to be able to improve its performance, but it seems that no feature weighting strategy can improve all

Table 1. Test set AUS on German credit data set.

Methods	Features Weighting Method			Equal Weights
	GA	*t*-Test	Entropy	
WSVM-1N-lin	78.88 ± 4.60	79.17 ± 4.63	78.63 ± 4.75	$\underline{79.76 \pm 4.46}$
WSVM-1N-rbf	$\underline{79.50 \pm 4.96}$	78.75 ± 4.67	78.42 ± 3.70	79.46 ± 4.31
WSVM-2N-lin	79.42 ± 4.92	79.66 ± 4.54	$\underline{79.70 \pm 4.52}$	79.70 ± 4.52
WSVM-2N-rbf	$\mathbf{79.58 \pm 3.92}$	$\underline{\mathbf{80.07 \pm 4.06}}$	79.69 ± 4.43	79.28 ± 4.31
WSVM-LS-lin	79.46 ± 4.46	$\underline{79.85 \pm 4.43}$	79.46 ± 3.60	79.70 ± 4.27
WSVM-LS-rbf	79.20 ± 4.10	$\underline{79.79 \pm 4.81}$	$\mathbf{79.71 \pm 4.36}$	$\mathbf{79.77 \pm 4.57}$
LR	74.07 ± 5.76			
LOR	79.66 ± 4.64			
BPNN	75.80 ± 3.91			
DT	63.06 ± 3.39			
KNN10	70.90 ± 6.19			
Adaboost	59.40 ± 2.01			

Table 2. Test set AUS on Australian credit data set.

Methods	Features Weighting Method			Equal Weights
	GA	*t*-Test	Entropy	
WSVM-1N-lin	92.48 ± 2.55	92.39 ± 3.30	93.11 ± 3.15	92.39 ± 2.71
WSVM-1N-rbf	91.23 ± 4.16	92.71 ± 3.14	93.28 ± 3.20	92.98 ± 3.20
WSVM-2N-lin	92.87 ± 2.41	92.85 ± 2.66	93.00 ± 2.62	92.88 ± 2.54
WSVM-2N-rbf	93.37 ± 2.57	$\mathbf{93.45 \pm 2.76}$	$\mathbf{93.31 \pm 2.90}$	$\mathbf{93.38 \pm 2.51}$
WSVM-LS-lin	92.79 ± 3.23	92.99 ± 2.89	93.07 ± 2.85	93.09 ± 2.89
WSVM-LS-rbf	$\mathbf{93.45 \pm 3.02}$	93.14 ± 2.56	93.17 ± 2.92	93.26 ± 3.09
LR	90.78 ± 2.79			
LOR	92.90 ± 2.74			
BPNN	91.31 ± 4.84			
DT	65.00 ± 9.51			
KNN10	61.46 ± 7.91			
Adaboost	87.46 ± 3.19			

kinds of SVM model. Moreover, the improvement is slight on both data sets. One reason for this may be that the performance of WSVM is mainly determined by the selection of parameters in SVM model (such as C and σ in RBF kernel). In these experiments, all parameters in SVM model are optimized by GA and thus the improvement space of performance is small. When compared with six traditional models, WSVM significantly outperform all except LOG model. In addition, WSVM models have stable good performance on both data sets.

Figure 3 shows the ROC curve with maximum and minimum AUC by WSVM-2N-rbf on the test sets of German credit data set among the 10-fold cross validations. Figure 4 shows ROC curve by WSVM-LS-rbf on Australian credit data set. From both the groups of figures, it can be observed that the difference of the ROC curve is slight for different feature weighting strategies; however, the difference of ROC curve with maximum AUC is significantly different from that with minimum AUC. For German data set, the maximum AUC is about 0.886, while the minimum AUC is 0.889 for Australian data set. In practice, to develop credit scoring models, the selection of training and testing samples are very important to measure the performance of model. However, what is the optimal way to select the samples is still open.

Sensitivity, specificity and overall accuracy are common measures for the performance of classifiers which is on basis of a selected value of threshold. While from the ROC curve, the decision makers can select the proper threshold value to make a balance between sensitivity and specificity of the model. Suppose the threshold value is denoted by t, then the sensitivity and specificity are two different functions of t, denoted by $e(t)$ and $p(t)$ separately. The coordinate of each point on the ROC curve is $(1 - p(t), e(t))$. The ROC curve is drawn within t value range. As shown in Fig. 3(a), the decision maker can choose a proper threshold to accept 80% good customers (sensitivity = 0.8) only at cost to accept no more than 20% bad customers (specificity $= 1 - 0.2$) from the ROC curve obtained from one-fold

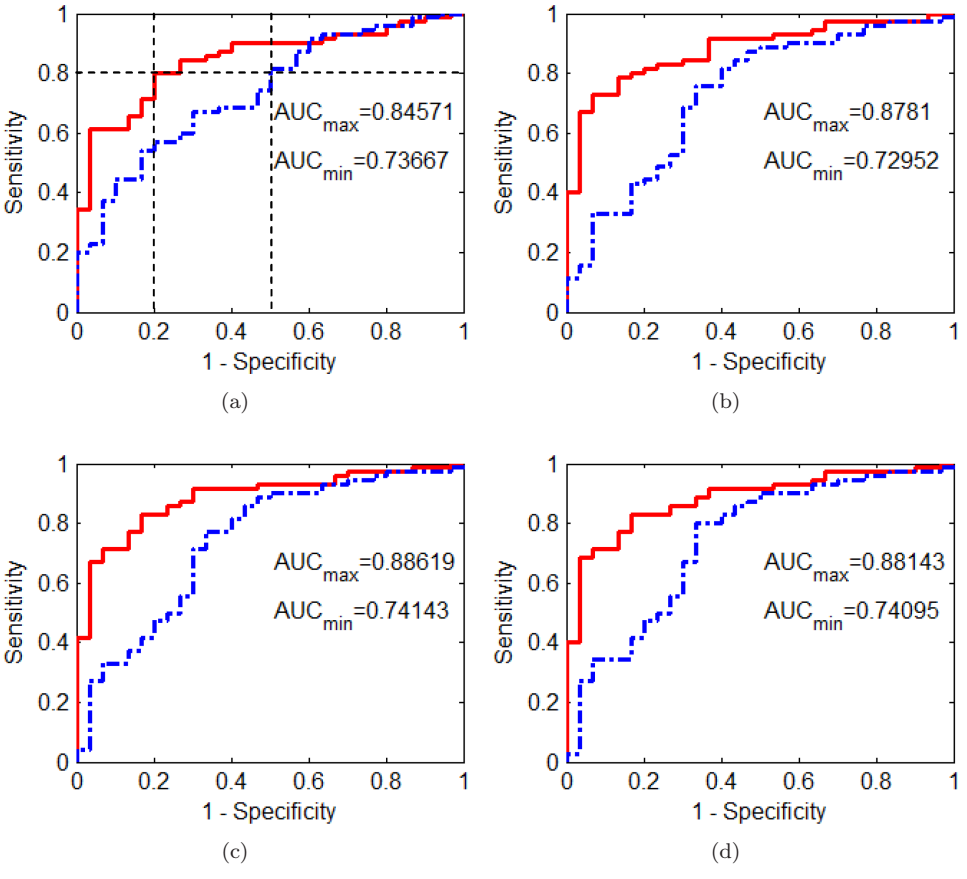

Fig. 3. ROC curves with max and min AUC by WSVM-LS-rbf model on test sets of German credit data set: (a) GA (b) *t*-test (c) entropy (d) equal weights.

iteration with maximum AUC value. While on the curve with minimum AUC value obtained from another fold iteration, when to achieve the goal of accepting 80% good customers, it will suffer the cost of accepting about 50% bad customers. From another viewpoint, if the decision maker want to improve specificity to reduce the risk of accepting more bad customers, the model rejects more bad customers and at the same time it also rejects good customers, thus the sensitivity is reduced. The strategy to make the balance of sensitivity and specificity is mainly dependent on two factors, one is the ratio of cost of misclassifying bad customer as good to that of misclassifying good customer as bad, another one is the population size of the two classes: good customers and bad customers. The classifier with larger AUC has more choice to get the higher specificity and sensitivity at the same time. Taking Figs. 3(a) and 4(a) as an example, the model for Australia sample has larger AUC than that of German sample, then the decision maker can get a model for Australia testing sample with specificity = 0.8 while keeping sensitivity greater than 0.9, but

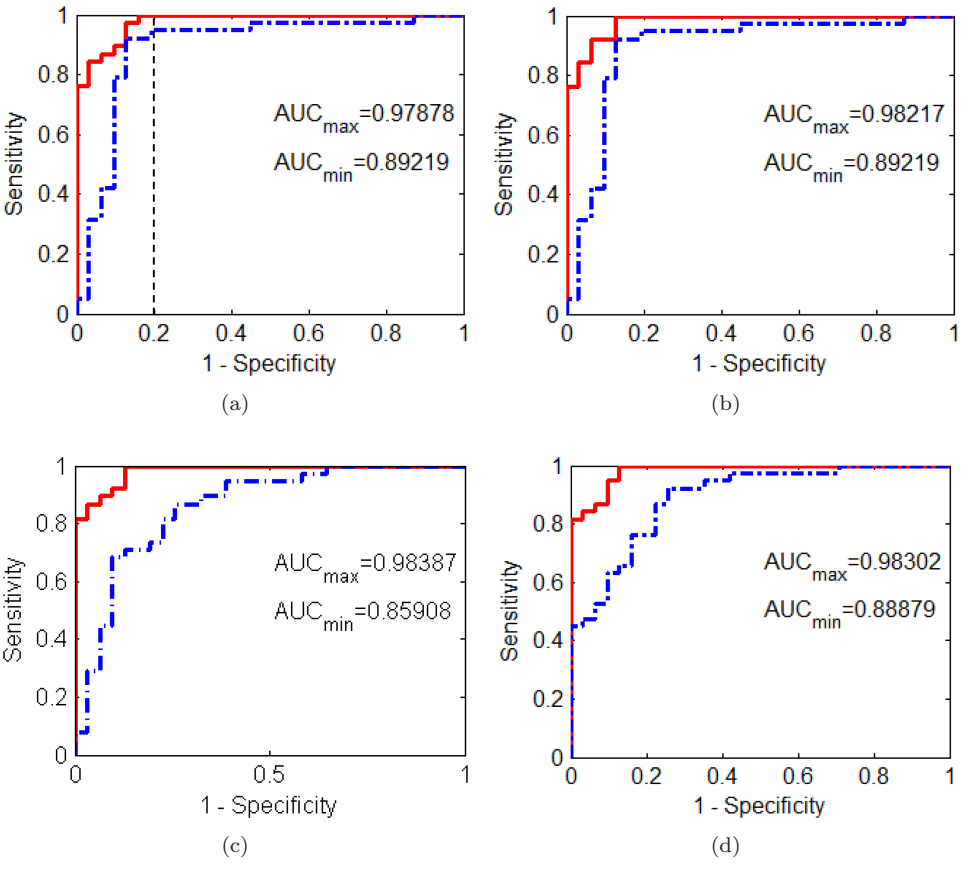

Fig. 4. ROC curves with max and min AUC by WSVM-2N-rbf model on test sets of Australian credit data set: (a) GA (b) t-test (c) entropy (d) equal weights.

for German testing sample, when keeping specificity equal to 0.8, the maximum sensitivity value is no more than 0.8.

5. Conclusion

This paper proposes several different credit scoring models with AUC maximization based on the new emerged SVM techniques and compares them with other six traditional methods. The experimental results on two real world data sets suggest that 2-Norm WSVM with RBF kernel and t-test features weighting can achieve slight improvement in AUC performance, however, it require more computational time. When the sample size is large in practice, LSSVM with parameters optimized by GA is good alternative. When compared with traditional methods, the AUC performance from WSVM model is significantly better than that from linear regression, decision tree, neural network, $k$nn and adaboost, but its slight outperformance to

logistic regression is not significant. Thus for credit scoring model with AUC maximization, logistic regression is also a good alternative.

In this research, the credit scoring models are developed with the objective to maximize AUC, however, in practice, the business objectives may focus on maximizing profit or minimizing loss. How to build SVM credit scoring models with practical business objectives will be our future research.

Acknowledgments

This study is supported by the grant from the NSFC of China and RGC of Hong Kong Joint Research Scheme (Project No. N_CityU110/07) and a Strategic Research Grant from City University of Hong Kong (Project No. 7008023). The authors would like to thank two anonymous reviewers for their valuable comments.

References

1. *The Nilson Report* (Oxnard, California, March 2007).
2. J. N. Crook, D. B. Edelman and L. C. Thomas, Recent developments in consumer credit risk assessment, *Eur. J. Oper. Res.* **183** (2007) 1447–1465.
3. E. Rosenberg and A. Gleit, Quantitative methods in credit management: a survey, *Operations research* **42** (1994) 589–613.
4. L. C. Thomas, A survey of credit and behavioural scoring: forecasting financial risk of lending to consumers, *Int. J. Forecasting* **16** (2000) 149–172.
5. A. D. P. Henriksen and S. W. Palocsay, An excel-based decision support system for scoring and ranking proposed R&D projects, *Int. J. Inform. Technol. Decision Making* **7** (2008) 529–546.
6. C.-T. B. Ho, D. D. Wu and D. L. Olson, A risk scoring model and application to measuring internet stock performance, *Int. J. Inform. Technol. Decision Making* **8** (2009) 133–149.
7. K.-J. Tseng, Y.-H. Liu and J.-F. Ho, An efficient algorithm for solving a quadratic programming model with application in credit card holders' behavior, *Int. J. Inform. Technol. Decision Making* **7** (2008) 421–430.
8. B. Baesens, T. V. Gestel, S. Viaene, M. Stepanova, J. Suykens and J. Vanthienen, Benchmarking state-of-the-art classification algorithms for credit scoring, *Journal of the Operational Research Society* **54** (2003) 627–635.
9. C.-L. Huang, M.-C. Chen and C.-J. Wang, Credit scoring with a data mining approach based on support vector machines, *Expert Systems with Applications* **33** (2007) 847–856.
10. S. Klaus and S. Ralf, *Support Vector Machines for Credit Scoring: Extension to Non Standard Cases* (2005).
11. Y. Peng, G. Kou, Y. Shi and Z. Chen, A multi-criteria convex quadratic programming model for credit data analysis, *Decision Support Systems* **44** (2008) 1016–1030.
12. L. Yu, S. Wang and K. K. Lai, Credit risk assessment with a multistage neural network ensemble learning approach, *Expert Systems with Applications* **34** (2008) 1434–1444.
13. O. Chapelle, V. Vapnik, O. Bousquet and S. Mukherjee, Choosing multiple parameters for support vector machines, *Machine Learning* **46** (2002) 131–159.
14. V. N. Vapnik, *Statistical Learning Theory* (Springer-Verlag, New York, 1998).
15. E. Osuna, R. Freund and F. Girosi, *Support Vector Machines: Training and Applications* (Massachusetts Institute of Technology, 1997).

16. S.-W. Lin, Z.-J. Lee, S.-C. Chen and T.-Y. Tseng, Parameter determination of support vector machine and feature selection using simulated annealing approach, *Applied Soft Computing* **8** (2008) 1505–1512.
17. J. A. K. Suykens, T. V. Gestel, J. D. Brabanter, B. D. Moor and J. Vandewalle, *Least Squares Support Vector Machines* (World Scientific, Singapore, 2002).
18. A. Rakotomamonjy, Optimizing area under ROC curve with SVMs. *Workshop on ROC Analysis in Artificial Intelligence* (2004).
19. T. Fawcett, An introduction to ROC analysis, *Pattern Recognition Letters* **27** (2006) 861–874.
20. J. Li, Z. Chen, L. Wei, W. Xu and G. Kou, Feature selection via least squares support feature machine, *Int. J. Inform. Technol. Decision Making* **6** (2007) 671–686.
21. MATHWORKS, Genetic algorithm and direct search toolbox: user's guide (2006).
22. I. Guyon, S. Gunn, M. Nikravesh and L. A. Zadeh, *Feature Extraction: Foundations and Applications* (Springer, 2006).
23. S. Theodoridis and K. Koutroumbas, *Patt. Recognit.* (Academic Press, 2006).
24. A. Vezhnevets and V. Vezhnevets, *'Modest AdaBoost' — Teaching AdaBoost to Generalize Better* (Graphicon-2005, Novosibirsk Akademgorodok, Russia, 2005).
25. C. C. Chang and C. J. Lin, *LIBSVM: A Library for Support Vector Machines* (2001).