ISSN : 0974 - 7435 Volume 8 Issue 7

BioTechnology
An Indian Journal

FULL PAPER

BTAIJ, 8(7), 2013 [923-932]

Ensemble-SVM-based Model of credit rating system in electronic commerce

Yuqiang Qin[1,2]*, Yudong Qi[1]

[1]College of Business Administration, Capital University of Economics and Business, Beijing, 100070, (CHINA)

[2]College of Economics and Management, Taiyuan University of Science and Technology, Taiyuan, 030024,Shanxi, (CHINA)

E-mail : qinyuqiang@126.com.

ABSTRACT

In this paper, a four-stage ensemble support vector machine (ESVM) based on multi-agent learning approach is proposed for credit rating system in electronic commerce. In the first stage, the initial credit dataset is divided into two independent subsets: training credit subset (in-sample data) and testing credit subset (out-of-sample data) for training and verification purposes. In the second stage, different ESVM learning paradigms with much dissimilarity are constructed as intelligent agents for credit rating evaluation. In the third stage, multiple individual ESVM agents are trained using training rating subsets and the corresponding rating results are also obtained. In the final stage, all individual rating results produced by ESVM in the previous stage are aggregated into an ensemble rating result. In particular, the impact of the diversity of individual intelligent agents on the generalization performance of the ESVM-based multi-agent learning way is examined and analyzed. For illustration, one corporate credit rating dataset is used to verify the effectiveness of the ESVM-based multiagent learning system.

KEYWORDS

Ensemble Support vector machine (ESVM);
Credit rating system; Multi-agent learning approach;
Multi-agent technology;
Credit dataset.

INTRODUCTION

The ongoing subprime mortgage crisis event originated from United States is attracting considerable concerns about credit rating analysis and evaluation. Generally, multiagent learning can be divided into two categories: competitive learning, where agents work asynchronously on the same problem and the result of the best agent is the final output result, and cooperative learning, where the final output result is a fusion or aggregation of the individual results of some agents. However, past studies have revealed that an effective multiagent learning system may not be an individual model by a single learning agent, but the combination or ensemble of some agents. Note that in the context of multiagent learning systems, an agent is usually defined as an independent learning unit participating as a member of the multiagent learning system. Usually, ensemble learning models outperform single learning models, whose performance is limited by the imperfection of feature extraction, learning algorithms, and the inadequacy of training data. Another reason supporting this

proposition is that different single learning models have their inherent drawbacks. Aggregating them may thus lead to a better model with a high generalization capability. From the above descriptions, we can conclude that there are two essential requirements

in a multiagent ensemble learning system.

To achieve high performance, this paper attempts to utilize a highly competitive machine learning tool – support vector machine (SVM), first proposed by Vapnik– as a generic ensemble learning agent for credit risk evalution. The main reasons of selecting SVM as an ensemble learning agent are three-fold. First of all, SVM requires less prior assumptions about the input data, such as normal distribution and continuity. This is different from traditional statistical models. Second, SVM can perform a nonlinear mapping from an original input space into a high dimensional feature space, in which it constructs a linear discriminant function to replace the nonlinear function in the original low-dimension input space. This characteristic also solves any dimension disaster problem because its computational complexity is not dependent on the sample dimension. Third, SVM attempts to learn the separating hyperplane to maximize the margin, therefore implementing structural risk minimization (SRM) and realizing good generalization capability. These important characteristics will also make SVM popular in many practical application problems.

The basic procedure of using SVM as a generic ensemble learning agent to construct a multiagent learning system consists of four stages. In the first stage, an initial dataset is divided into two independent subsets: training subset and testing subset. In some necessary situations, the third subset, validation subset may be produced to overcome overfitting problem. In the second stage, diverse SVM models are created to formulate the generic ensemble learning agents (i.e., SVM agents) through using some diversity strategies. In the third stage, the diverse SVM models are trained by the training subset to produce different results. In the final stage, these different SVM agents are integrated into an aggregated output using a specific ensemble strategy.

The main aims of this paper are to construct a SVM-based multiagent ensemble learning system for credit risk evaluation and to investigate the effect of both diversity strategy and ensemble strategy on the generalization performance of the SVM-based multiagent ensemble learning system.

METHODOLOGY FORMULATION

In this section, a four-stage SVM-based multiagent ensemble learning approach is proposed for credit risk evaluation problems. First of all, the initial dataset is divided into two independent subsets: training subset (in-sample data) and testing subsets (out-of-sample data) for learning and testing purposes. A validation subset is also included in the training subset by k-fold cross validation approach. Then, diverse SVM learning paradigms are used as intelligent agents to analyze and evaluate credit risk. Subsequently, multiple individual SVM agents are trained using training subset and the corresponding evaluation results are also obtained. Finally, all individual results produced by multiple single SVM agents in the previous stage are aggregated into a final output result.

Data partition

In applications of SVM on multiagent ensemble learning, data partition is a necessary step. Some research results shown that the data division can have a significant impact on the results obtained. In previous similar studies, data division was carried out in all cases. Generally, data partition is carried out on an arbitrary basis, and the statistical properties of the respective datasets are seldom considered. In most cases, two subsets, i.e., training subset and testing subset, are used. In this paper, the overall dataset is also divided into two subsets. Eighty percent of the data are used as the training subset and the remainder is used as the testing subset. Note that k -fold cross validation is used in the paper to achieve a more reliable result.

Diverse SVM agent creation

In order to capture the implicit patterns hidden in the dataset from different perspectives, diverse SVM agents should be used. Generally, an effective multiagent ensemble learning system consisting of diverse models with much disagreement is more likely to have a good generalization performance in terms of the principle of bias-variance trade-off. Therefore, how to generate the diverse models is a crucial factor for constructing an

effective multiagent ensemble learning system. For SVM models, several diversity strategies have been investigated for the generation of ensemble members making different errors. Such diversity strategies basically relied on varying the training samples and parameters related to the SVM design. In particular, some main diversity strategies can be categorized into the following three aspects:

Data diversity

Because different data often contain different information, these different data can generate diverse models with dissimilarity. Usually, in intelligent learning models, training dataset is used to construct a concrete model and thus different training dataset will be an important source of diversity if the training dataset at hand is functionally or structurally divisible into some distinct training subsets. In order to achieve diverse models, some typical data sampling approaches, such as bootstrap aggregating (bagging) proposed by and noise injection, are used to create diversity. In this paper, the bagging algorithm is adopted as a data sampling tool. The bagging approach is a widely used data sampling method in machine learning. Given that the size of the original data set DS is P, the size of new training data is N, and the number of new training data points is m, that is, each new training subset TR has m data points, $TR_1, TR_2, ..., TR_m$. Accordingly the bagging algorithm for generating new training subsets can be shown in algorithm:.

Input: original data set DS

Output: Thegenerated new training subsets $(TR_1, TR_2, ..., TR_m)$

```
For t=1 to m
  For i=1 to N
    RandRow=P*rand()
     If RandRowf" P
     P_i(i, AllColumns)=DS(RandRow,AllColumns)
     End If
    Next i
Next t
```

Output the final training subsets $(\mathbf{TR_1}, \mathbf{TR_2}, ..., \mathbf{TR_m})$

The bagging algorithm is very efficient in constructing a reasonably sized training subset when the size of the original data set is small due to the feature of its random sampling with replacement. Therefore, the bagging is a useful data sampling method for machine learning. In this paper, we use the bagging algorithm to generate different training subsets.

Parameter diversity

In an SVM model, there are two classes of typical parameters: regulation parameter and kernel parameters. By changing the SVM model parameters, different SVM models with high levels of disagreement can be produced. In this paper, we will investigate the impacts of different kernel parameters on SVM generalization performance.

Kernel diversity

In an SVM model, the kernel function has an important effect on the generalization performance of SVM. Hence, using different kernel functions in the SVM models can also create diverse SVM models. In the SVM model, the polynomial function, Sigmoidal function, and RBF function, are several typical kernel functions. Similarly, this paper will also examine the influences of different kernel functions on the SVM generalization performance.

Single SVM agent learning

After determining diverse SVM models, the next step is to train the different SVM agents to produce different output results using training subsets. In our paper, the standard SVM proposed by Vapnik is used as the intelligent learning agent. The generic idea of SVM is to maximize the margin hyperplane in the feature space. Similar to other supervised learning methods, an underlying theme of the SVM is to learn from data, which adopts the structural risk minimization (SRM) principle from computational learning theory. Usually, SVM can be used as regression and classification. In this paper, we focus on the classification problem.

Let $\mathbf{x}$ be an input vector as $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$, where $\mathbf{x}_i$ $(\mathbf{i = 1,2,..., N})$ is the ith element of $\mathbf{x}$ in the training subset. Let $\mathbf{y}$ be an output vector as $\mathbf{y} = \{\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_N\}$, where $\mathbf{y}_i$ $(\mathbf{i = 1,2,..., N})$ is the ith element of . is the number of training data points. In SVM classification, $\mathbf{y}_i$ is the class value of the training data point $\mathbf{x}_i$. Using $\mathbf{x}_i$ and $\mathbf{y}_i$, the SVM classification prob-

FULL PAPER

lem can be represented in the following optimization problem:

$$\begin{cases} \min J(\mathbf{a},b,\xi) = (1/2)\mathbf{a}^T\mathbf{a} + C\sum_{i=1}^{N}\xi_i \\ s.t.\, y_i\left[\varphi(\mathbf{x}_i)\cdot\mathbf{a}_i + b\right] \geq 1-\xi_i, \\ \xi_i \geq 0, i = 1,2,...,N \end{cases} \quad (1)$$

where $\mathbf{a}$ is a normal vector of the hyperplane as $\mathbf{a}=\{a_1,a_2,...,a_N\}$, where $a_i\,(i=1,2,...,N)$ is the ith element of $\mathbf{a}$. b is a bias that is a scalar $\phi(\mathbf{x}_i)$. is a nonlinear mapping function for the ith input data $\mathbf{x}_i$ from a low-dimension space to a high-dimension space. ξ is a tolerable misclassification error vector as $\xi=\{\xi_1,\xi_2,...,\xi_N\}$, where $\xi_i\,(i=1,2,...,N)$ is the ith misclassification error of ξ C . is a regulation parameter controlling the trade-off between the maximal margin and the tolerable misclassification errors. When C is large, the error term will be emphasized. Small C means that the large classification margin is encouraged. Vapnik found that training a SVM model will lead to a quadratic programming (QP) problem with bound constraints and linear equality constraints, as shown in Eq. (2).

$$\begin{cases} \max J(\alpha) = \sum_{i=1}^{N}\alpha_i - (1/2) \\ \sum_{i,j=1}^{N}\alpha_i\alpha_j y_i y_j \varphi(\mathbf{x}_i)^T\varphi(\mathbf{x}_j) \\ = \sum_{i=1}^{N}\alpha_i - (1/2)\sum_{i,j=1}^{N}\alpha_i\alpha_j y_i y_j K(\mathbf{x}_i,\mathbf{x}_j) \\ s.t.\sum_{i=1}^{N}\alpha_i y_i = 0, 0 \leq \alpha_i \leq C, i = 1,2,...,N \end{cases} \quad (2)$$

where α is a vector of Lagrange multipliers as $\alpha=\{\alpha_1,\alpha_2,...,\alpha_N\}$, where $\alpha_i\,(i=1,2,...,N)$ is the *i* th Lagrange multiplier of α corresponding to *i* th inequality constraint defined in Eq. (1). $K(\mathbf{x}_i,\mathbf{x}_j)(i,j=1,2,...,N,i\neq j)$ is defined as the kernel function with $K(\mathbf{x}_i,\mathbf{x}_j)=\varphi(\mathbf{x}_i)^T\varphi(\mathbf{x}_j)$. The elegance of using the kernel function is that one can deal with feature spaces of arbitrary dimensionality without having to compute the nonlinear map function $\varphi(\mathbf{x})$ explicitly, where $\mathbf{x}=\{\mathbf{x}_i,\mathbf{x}_2,...,\mathbf{x}_N\}$. Any function that satisfies Mercer's condition can be used as the kernel function for SVM models. Typical examples of the kernel function are the polynomial kernel ÿradial basis function $K_{poly}(\mathbf{x}_i,\mathbf{x}_j)=(\mathbf{x}_i^T\mathbf{x}_j+1)^d$ (RBF) kernel: $K_{rbf}(\mathbf{x}_i,\mathbf{x}_j)=\exp(-|\mathbf{x}_i-\mathbf{x}_j|^2/\sigma^2)$, and sigmoid kernel $K_{sig}(\mathbf{x}_i,\mathbf{x}_j)=\tanh(\rho\mathbf{x}_i^T\mathbf{x}_j+\theta)$, where d,σ,P and are the kernel parameters.

From the implementation point of view, training SVM is actually equivalent to solving the linearly constrained QP problem. By solving the QP problem, the final output function $\mathbf{f(z)}$ of the SVM model can be represented as

$$\mathbf{f(z)} = \sum_{l=1}^{N_L}\alpha_l^{(r)}y_l^{(r)}K(z,\mathbf{x}_l^{(r)}) + b \quad (3)$$

where $\alpha_l^{(r)}(l=1,2,...,N_L)$ is the *l* th element of a support vector $\alpha^{(r)}$, $\alpha^{(r)}=\{\alpha_1^{(r)},\alpha_2^{(r)},...,\alpha_{N_L}^{(r)}\}$, which is selected from α, the vector of Lagrange multipliers, defined in Eq. (2). N_L is the number of elements of the support vector $\alpha^{(r)}$. Generally, the N_L is less than the number of training data points N in many practical problems, namely, $N_L \leq N$ $\mathbf{x}_l^{(r)}(l=1,2,...,N_L)$. is the lth element of a vector $\mathbf{x}^{(r)}$ corresponding to the *l* th element of support vector $\alpha^{(r)}$. The vector $\mathbf{x}^{(r)}$ is a part of the input vector $\mathbf{x}$ as $\mathbf{x}^{(r)}=\{\mathbf{x}_1^{(r)},\mathbf{x}_2^{(r)},...,\mathbf{x}_{N_L}^{(r)}\}$ selected from the input $\mathbf{x}$ vector satisfying the conditions shown in Eq. (2). $y_l^{(r)}(l=1,2,...,N_L)$ is the th element of corresponding to the *l* th element $\mathbf{y}^{(r)}$ of support vector $\alpha^{(r)}$. The vector $\mathbf{y}^{(r)}$ is a part of output vector $\mathbf{y}$ as $\mathbf{y}^{(r)}=\{y_1^{(r)},y_2^{(r)},...,y_{N_L}^{(r)}\}$ selected from the output vector satisfying the conditions shown in Eq. (2). z is a testing data vector as $\mathbf{z}=\{z_1,z_2,...,z_m\}$, where *m* is the number of testing data points and *b* is defined in Eq. (1).

Through training SVM, model parameters can be determined and accordingly the SVM classifier *F(z)* can be represented as

$$F(z) = \text{sign}(f(z))$$
$$= \text{sign}\left(\sum_{l=1}^{N_L} \alpha_l^{(r)} y_l^{(r)} K(z, x_l^{(r)}) + b\right) \quad (4)$$

where **sign** is an indicator function and other symbols are identical to Eq. (3).

With the above SVM classifier, some results for a specific problem can be achieved.

Multiagent ensemble learning

When individual SVM agents are generated, each SVM agent can output its own computational results for a specific problem. Although one agent based on a single SVM model may have good performance, it is sensitive to samples and parameter settings, i.e. each single SVM agent may have some biases. One effective way to reduce the bias is to integrate these SVM agents into an aggregated output for final results. FIgure 2 shows the structure of the multiagent ensemble learning system, where p is the number of learning agents.

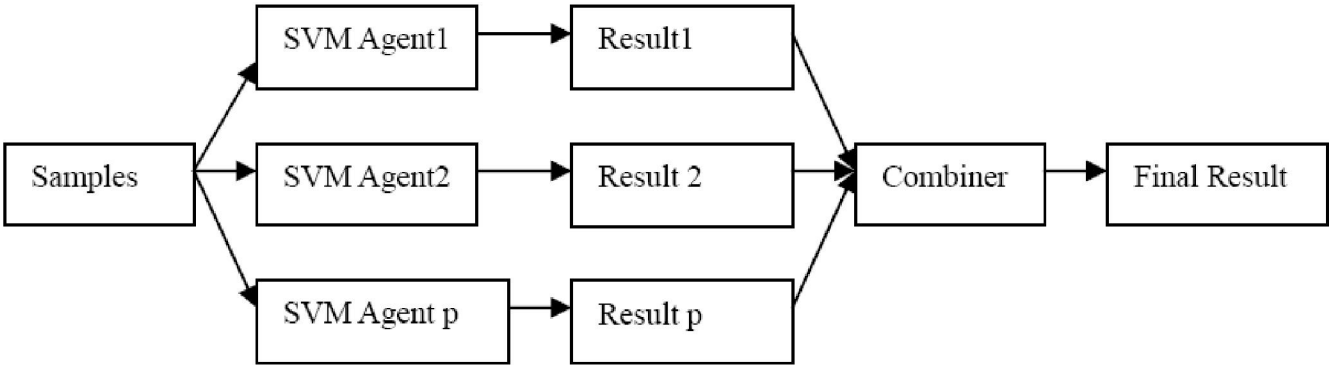

FIgure 2 : The structure of multiagent ensemble learning system.

From the previous descriptions and FIgure 2, it is easy to find that how to construct the combiner, and how to create diversity among the multiple agents, are the two main factors in building an effective multiagent ensemble learning system. Because the previous sections have introduced some diverse SVM agents by using different diversity strategies, how to construct an efficient combiner has been a key factor in constructing an effective multiagent ensemble learning system.

However, before integrating these SVM agents, strategies for selecting SVM agents must be noted. Generally, these strategies can be divided into two categories: '$producing an exact number of SVM learning agents; and a$overproducing SVM learning agents and then selecting a subset of these overproduced SVM agents.

In the first strategy, several common ensemble approaches, e.g., boosting (Schapire, 1990), can be employed to generate the exact number of diverse learning agents for ensemble purpose. That is, no selection process will be used in this strategy, and all the learning agents produced will be combined into an aggregated output. In the second strategy, the main aim is to create a large number of agent candidates and then choose some most diverse agents for the ensemble. The selection criterion includes some error diversity measures, such as mean squares errors (MSE), which is introduced in detail by Partridge & Yates (1996). Because the first strategy is based upon the idea of creating diverse SVMs at the early stage of ensemble design, it is more effective than the second strategy; especially for some conditions where some powerful computing resources are restricted. However, the main reason is that the second strategy is not preferred is because it cannot avoid occupying large amounts of computing time and storage space while creating a large number of ensemble agent candidates, some of which are to be later discarded.

Generally, there are some ensemble strategies to take into account different results in the existing literatures. Typically, majority voting and weighted averaging are two popular ensemble

strategies. The main objective of majority voting strategy is to determine the vote of the majority of the population of learning agents. To our knowledge, majority voting is one of the most widely used ensemble strategies due to its easy implementation. In the majority voting strategy, each agent has the same weight and the voting of the ensemble agents will determine the final result. Usually, it takes over half the ensemble agents

FULL PAPER

to agree in order for a result to be accepted as the final output of the ensemble, regardless of the diversity and accuracy of each agent's generalization. In the mathematic form, the result of the final ensemble output G(z) can be represented as

$$G(z) = \text{sign}(\sum_{k=1}^{p} F_k(z) / p) \quad (5)$$

where z is a testing data vector as $z = \{z_1, z_2, ..., z_m\}$ defined in Eq.(3), $F_k(z)$ is the output result of the th learning agent presented in Eq. (4), using the testing data vector *z*,*m* is the number of testing data points and *p* is the number of learning agents. Although this strategy is easy to be used and implemented, a serious problem associated with majority voting is that it ignores the fact that some learning agents that lie in a minority sometimes produce the more accurate results. At the ensemble stage, it ignores the existence of diversity that is the motivation for a multi-agent ensemble learning system.

Weighted averaging is where the final output result is calculated in terms of single agents' performance, and a weight is attached to each single agent's output. The sum of the total weight is one and each agent is entitled to a portion of this total weight based on their performance. A typical binary classification example is to use validation examples to determine the performance weight. Suppose that represents the number of observed Class instances that are mistakenly classified as Class *B* by an agent, *AA* denotes the number of correctly classified Class *A* instances that belong to Class *A*; *AB* represents the number of observed Class *B* instances that are mistakenly classified as Class *A*, while *BB* represents the number of correctly classified Class *B* instances that belong to Class *B*. Some common measures used to evaluate the performance of the agent are defined below:

$$\text{TypeIAccuracy} = \text{Specificity} = \frac{BB}{BB + BA} \quad (6)$$

$$\text{TypeIIAccuracy} = \text{Sensitivity} = \frac{AA}{AA + AB} \quad (7)$$

$$\text{TotalAccuracy(TA)} = \frac{AA + BB}{AA + AB + BB + BA} \quad (8)$$

Each SVM agent is trained by a training dataset and is verified by the validation samples. Assume that the total accuracy of validation samples of agent k is denoted by TA_k, the weight of SVM agent k denoted by w_k can be calculated as

$$w_k = TA_k \Big/ \sum_{k=1}^{p} TA_k \quad (9)$$

Then the final ensemble output value *G(z)* of this strategy is shown as follows:

$$G(z) = \text{sign}(\sum_{k=1}^{p} w_k \cdot F_k(z)) \quad (10)$$

where w_k is the k th weight for k th agent and other symbols are identical to Eq. (5).

The above TA -based weight averaging ensemble strategy is a typical class of ensemble strategy, the determination of the weight is heavily dependant on the validation samples. When there is no validation subset in data division, this ensemble strategy will be useless.

DATA DESCRIPTION AND EXPERIMENT DESIGN

In this section, a British credit card application approval dataset is used. It is from a financial service company in England, obtained from the accessory CDROM of Thomas, Edelman, Crook. The dataset includes detailed information of 1225 applicants, including 323 observed bad applicants. In the 1225 applicants, the number of good cases (902) is nearly three times that of bad cases (323). To make the numbers of the two classes near equal, we triple the number of bad cases, i.e. we add two copies of each bad case. Thus the total dataset grows to 1871 cases. The purpose of doing this is to avoid having too many good cases or too few bad cases in the training samples. Then we randomly draw 1000 cases comprising 500 good cases and 500 bad cases from the total of 1871 cases as the training samples and treat the rest as testing samples (i.e. 402 good applicants and 469 bad applicants). In these cases, each case is characterized by 14 decision attributes, which are described as follows:

(1) Year of birth.
(2) Number of children.
(3) Number of other dependents.
(4) Is there a home phone.
(5) Applicant's income.

(6) Applicant's employment status.
(7) Spouse's income.
(8) Residential status.
(9) Value of home.
(10) Mortgage balance outstanding.
(11) Outgoings on mortgage or rent.
(12) Outgoings on loans.
(13) Outgoings on hire purchase.
(14) Outgoings on credit cards.

Using this dataset and previous descriptions, we can construct a practical SVM-based multiagent ensemble learning system for credit risk evaluation. The basic purpose of the multiagent ensemble learning model is to make full use of knowledge and intelligence of each agent in the multiagent ensemble learning system. In this multiagent ensemble learning system, agents are some intelligent techniques or intelligent agents.

Subsequently, in order to investigate the impacts of diversity strategy and ensemble strategy on the performance of a multiagent ensemble learning system, we perform different experiments with diverse diversity strategies and ensemble strategies. When testing the effects of diversity strategies on the performance of a multiagent ensemble learning system, the ensemble strategy is assumed to be fixed. Similarly, the diversity strategy will be fixed if the impacts of ensemble strategies on the performance of the multiagent ensemble learning system are evaluated.

In the testing of diversity strategies, a majority voting based ensemble strategy is adopted. For data diversity, a bagging algorithm is utilized. In this paper, we use five different training scenarios for SVM learning. For example, we use 500 training sets (i.e. P = 1871, N = 1000, and m = 500) to create 500 different evaluation results for each intelligent agent. In this case, the SVM model with a RBF kernel is used as an intelligent learning agent. The parameters of SVM including regulation parameter and kernel parameters are determined by a grid search.

For parameter diversity, we use SVM models with the RBF kernel as intelligent learning agents. Accordingly, the regulation parameter C and RBF kernel parameter r2 will be changed. The training data used is the initial training data produced by data partition.

For kernel diversity, three typical kernel functions, polynomial function, RBF function and sigmoid function are used. In this case, the parameters of SVM are determined by a grid search. The training data used is the initial training data produced by data partition.

In the testing of ensemble strategies, bagging-based diversity strategy is used. In this case, the SVM models with the RBF kernel are used as intelligent learning agents. The parameters of SVM are determined by a grid search.

EXPERIMENTAL RESULTS

Using the previous experiment design, three types of different experiments were conducted. The first was the diversity strategy testing experiment, the second was the ensemble strategy testing experiment, and the final one was the performance comparison of different models.

Diversity strategy testing

In this subsection, three different diversity strategies, data diversity, parameter diversity and kernel diversity, were used to investigate the impact on the multiagent ensemble learning model in group decision making.

In the data diversity testing, the main goal was to examine the impact of the number of training samples on the generalization performance of the multiagent ensemble learning model. For this purpose, we adopted the bagging algorithm to create 100, 500, 1000, 2000 and 5000 different training samples (5 different scenarios), as mentioned previously. Using these different training samples, different single SVM models with dissimilarities were produced. With the trained SVM models, some classification results were achieved. Using the majority voting strategy, the final multiagent ensemble

TABLE 1 : Performance comparisons of SVM ensemble learning with data diversity.

Number of training samples	Type I(%)	Type II(%)	Total accuracy(%)
100	66.94[3.58]	63.45[3.93]	65.12[3.77]
500	69.38[3.23]	66.36[3.71]	67.82[3.52]
1000	73.44[2.81]	67.93[3.26]	70.54[3.03]
2000	73.97[2.76]	68.14[3.15]	70.91[2.87]
5000	73.65[2.73]	68.72[3.29]	71.06[2.85]

results were obtained from the multiagent ensemble learning system. Accordingly the final computational results are shown in TABLE 1. Note that the values in brackets are standard deviations.

Based on the results in TABLE 1, two conclusions can be easily drawn. On the one hand, as the number of training samples increases, the performance improvements are increased from the perspective of total accuracy. On the other hand, the degree of performance improvement from 100 datasets to 1000 datasets is larger than the degree of performance improvement from 1000 datasets to 5000 datasets, which indicates that the degree of performance improvement is not fully dependant on the number of training samples.

In the parameter diversity testing, the SVM models with the RBF kernel are used as intelligent learning agents. Therefore, different regulation parameters and kernel parameters are used to create diversity. In particular, the regulation parameter varies from 10 to 100 with a step size of 10 and the kernel parameter changes from 10 to 100 with a step size of 10. Some computational results are shown in TABLE 2.

From TABLE 2, we can observe that different parameter settings can lead to different generalization performance, but the differences are insignificant. More specifically, the classification performance on the testing data increases when C increases from 10 to 80, but the classification performance decreases when C increases from 90 to 100. Thus a suitable regulation parameter is of utmost importance to SVM learning. The main reason is that the regulation parameter C is an important trade-off parameter between margin maximization and tolerable classification errors. If the selection of regulation parameter C is inappropriate, it might lead to unexpected generalization results. But these results partly support the conclusions of Kim (2003) & Tay & Cao (2001). However, for kernel parameter r2, it is difficult to find similar results. The possible reason is worth exploring further in the future research.

Similarly, we can also use different kernel functions

TABLE 2 : Performance comparisons of SVM ensemble learning with different parameters.

C	δ^2	Type I(%)	Type II(%)	Total accuracy(%)
10	10,20,…,100	67.48[4.01]	63.75[3.54]	65.32[3.82]
20	10,20,…,100	69.34[3.84]	64.89[3.78]	66.81[3.94]
30	10,20,…,100	68.96[3.33]	65.05[3.66]	66.88[3.53]
40	10,20,…,100	69.58[4.21]	64.98[4.08]	67.14[4.14]
50	10,20,…,100	70.24[3.57]	65.46[4.14]	67.51[3.88]
60	10,20,…,100	69.87[4.01]	66.63[4.26]	68.22[4.12]
70	10,20,…,100	71.32[3.82]	67.89[4.01]	69.44[3.96]
80	10,20,…,100	70.45[3.51]	69.34[3.83]	69.53[3.73]
90	10,20,…,100	70.08[4.14]	69.08[4.54]	69.39[4.31]
100	10,20,…,100	69.87[4.76]	67.85[5.04]	68.57[4.89]
10,20,…,100	10	69.65[4.25]	67.42[3.96]	68.93[4.15]
10,20,…,100	20	71.93[3.92]	66.38[4.33]	68.67[4.18]
10,20,…,100	30	68.02[4.61]	66.42[3.89]	67.14[4.27]
10,20,…,100	40	69.98[4.83]	67.02[4.22]	68.49[4.54]
10,20,…,100	50	68.69[3.95]	64.87[4.45]	66.78[4.29]
10,20,…,100	60	69.65[3.32]	66.22[3.86]	67.54[3.66]
10,20,…,100	70	67.43[4.47]	66.38[3.92]	66.76[4.23]
10,20,…,100	80	68.52[4.81]	65.15[3.74]	66.82[4.43]
10,20,…,100	90	66.49[4.09]	65.41[4.45]	65.78[6.09]
10,20,…,100	100	67.56[3.85]	66.24[4.08]	66.78[3.97]

to create a multiagent SVM ensemble learning system for credit risk evaluation. For this purpose, three different kernel functions: polynomial function, sigmoid function and RBF function are used for testing. In order to construct a diverse ensemble learning system, 1000 copies of the same training dataset are replicated. In the 1000 training data points, different kernel functions are hybridized to create the kernel diversity. Note that in every experiment only one kernel function dominates the multiagent ensemble learning system. Detailed configuration of the kernel functions and corresponding computational results are shown in TABLE 3.

From TABLE 3, several interesting results should be noted. First of all, the multiagent ensemble learning system dominated by the RBF-type kernel function produces the best classification results, followed by the sigmoid kernel and the polynomial kernel. Second, in terms of the total accuracy, there is a significant difference at the 10% level in the two-tail t-test between RBF kernel function and polynomial kernel function. Third, although the multiagent ensemble learning system dominated by the RBF kernel is advantageous to the ensemble learning system dominated by the sigmoid kernel, the robustness of the former is slightly worse than that of the latter (determined from the standard deviation measure). The possible reason for this is unknown, and worth exploring further.

TABLE 3 : Performance comparisons of SVM ensemble learning with kernel diversity.

No.	Kernel functions	Type I(%)	Type II(%)	Total accuracy(%)
1	600Poly+200Sig+200RBF	64.76[4.76]	61.89[5.05]	63.25[4.94]
2	200Poly+600Sig+200RBF	68.14[4.28]	65.51[4.69]	66.68[4.55]
3	200Poly+200Sig+600RBF	71.32[4.93]	68.13[4.71]	68.59[4.87]

CONCLUSIONS

In this paper, a multi-stage SVM-based multi-agent ensemble learning system was proposed for credit risk evaluation problems. For the purposes of verification, one publicly available credit dataset was used in order to test effectiveness and classification power. In particular, multiple different experiments were conducted to test the impact of various diversity strategies and ensemble strategies on the performance of a SVM-based ensemble learning system. All results reported in the experiments clearly show that the proposed SVM-based multi-agent ensemble learning approach can consistently outperform the other comparable models, including the other

two ensemble learning models and the five single agent learning systems. The obtained results reveal that the proposed SVM-based multi-agent ensemble learning model can provide a promising solution to credit risk evaluation problem and implies that the proposed SVM-based multi-agent ensemble learning technique has a great potential in its application to other classification problems.

ACKNOWLEDGEMENTS

This work is sponsored by The National Natural Science Foundation of China (NO.61072087), Shanxi International Cooperation Foundation of Science and Technology of China (NO.2011081047), Shanxi Natural Science Foundation of China (NO. 2010011020-1) and The PhD Start-up Foundation of Taiyuan University of Science and Technology (W20122006)

REFERENCES

[1] T.Bellotti, J.Crook; Support vector machines for credit scoring and discovery of significant features. Expert Systems with Applications, **36(2)**, 3302–3308 **(2009)**.

[2] Yuqiang Qin, Xueying Zhang;. Fuzzy Support Vector Machine-Based Emotional Optimal Algorithm in Spoken Chinese[J]. Journal of Computational and Theoretical Nanoscience, **9(10)**, 1715–1719 **(2012)**.

[3] H.Abdou, J.Pointon, A.El-Masry; Neural nets versus conventional techniques in credit scoring in Egyptian banking. Expert Systems with Applications, **35(3)**, 1275–1292 **(2008)**.

[4] W.Chen, C.Ma, L.Ma; Mining the customer credit using hybrid support vector machine technique.

Expert Systems with Applications, **36(4)**, 7611–7616 **(2009)**.

[5] R.J.Elliott, A.Filinkov; A self tuning model for risk estimation. Expert Systems with Applications, **34(3)**, 1692–1697 **(2008)**.

[6] QI Yu-dong, Fan He-Jun; Deregulation: Reform Trend of China's Natural Monopoly Industries[J]. China Industrial Economics, **21(4)**, 62-72 **(2009)**.

[7] J.T.S.Quah, M.Sriganesh; Real-time credit card fraud detection using computational intelligence. Expert Systems with Applications, **35(4)**, 1721–1732 **(2008)**.

[8] L.Yu, S.Y.Wang, K K.Lai; Credit risk assessment with a multistage neural network ensemble learning approach. Expert Systems with Applications, **34(2)**, 1434–1444 **(2008)**.

[9] L.Yu, S.Y.Wang, K.K.Lai, L.G.Zhou; Bio-inspired credit risk analysis computational intelligence with support vector machines. Berlin: Springer-Verlag, **(2008)**.

[10] Yuqiang Qin, Xueying Zhang; MSF-Based Speaker Automatic Emotional Recognition in Continuous Chinese Mandarin[J]. **Procedia Engineering, 15(11)**, 2229–2233 **(2011)**.