Knowledge-Based Systems 76 (2015) 67–78

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

# Support vector machine-based optimized decision threshold adjustment strategy for classifying imbalanced data

Hualong Yu [a,b,c], Chaoxu Mu [a,b], Changyin Sun [a,b,*], Wankou Yang [a,b], Xibei Yang [c], Xin Zuo [c]

[a] *School of Automation, Southeast University, Nanjing 210096, China*
[b] *Key Lab of Measurement and Control of Complex Systems of Engineering, Ministry of Education, Southeast University, Nanjing 210096, China*
[c] *School of Computer Science and Engineering, Jiangsu University of Science and Technology, Zhenjiang, Jiangsu 212003, China*

ARTICLE INFO

*Article history:*
Received 3 May 2014
Received in revised form 4 December 2014
Accepted 5 December 2014
Available online 13 December 2014

*Keywords:*
Class imbalance
Support vector machine
Decision threshold adjustment
Optimization search
Ensemble learning

ABSTRACT

Class imbalance problem occurs when the number of training instances belonging to different classes are clearly different. In this scenario, many traditional classifiers often fail to provide excellent enough classification performance, i.e., the accuracy of the majority class is usually much higher than that of the minority class. In this article, we consider to deal with class imbalance problem by utilizing support vector machine (SVM) classifier with an optimized decision threshold adjustment strategy (SVM-OTHR), which answers a puzzled question: how far the classification hyperplane should be moved towards the majority class? Specifically, the proposed strategy is self-adapting and can find the optimal moving distance of the classification hyperplane according to the real distributions of training samples. Furthermore, we also extend the strategy to develop an ensemble version (EnSVM-OTHR) that can further improve the classification performance. Two proposed algorithms are both compared with many state-of-the-art classifiers on 30 skewed data sets acquired from Keel data set Repository by using two popular class imbalance evaluation metrics: F-measure and G-mean. The statistical results of the experiments indicate their superiority.

## 1. Introduction

In the past decade, the class imbalance problem has received considerable attention in several fields, such as artificial intelligence [1], machine learning [2] and data mining [3,4]. A data set is said to be imbalanced when and only when the instances of some classes are obviously much more than that in other classes. The problem is important due to it widely emerges in many real-world applications, including financial fraud detection [5], network intrusion detection [6], spam filtering [7], video monitoring [8], medical diagnosis [9], Bioinformatics [10], etc. Generally, in these applications, we are more interested in the pattern represented by the examples of the minority class. However, majority traditional classification algorithms pursuing the minimal training errors would heavily damage the recognition accuracy of the minority class, thus it is necessary to adopt some bias correction techniques before/after constructing a classifier.

The bias correction techniques can be roughly divided into four major categories as follows:

1. Resampling the original training set until all the classes are approximately equally represented. Resampling includes oversampling [11–13], undersampling [14,15] and hybrid sampling [16].
2. Cost-sensitive learning, which is also called instances weighting method, assigns different weights for the training instances belonging to different classes so that the misclassification of the minority class can be highlighted [17–19].
3. Moving the decision boundary (decision threshold adjustment) towards the majority class in order to remedy the bias caused by skewed sample distributions [20,21]. Unlike the other correction techniques, decision threshold adjustment strategy runs after modeling a classifier.
4. Ensemble learning that provides a framework to incorporate resampling strategy, weighting strategy or decision threshold adjustment strategy, usually produces better and more balanced classification performance [22–29].

Among those correction techniques mentioned above, decision threshold adjustment is regarded as a potential solution for dealing with class imbalance in recent studies [20,21]. However, the existing decision threshold adjustment approaches generally give the moving distance of classification boundary empirically, thus fail

* Corresponding author at: School of Automation, Southeast University, No. 2 Sipailou, Nanjing 210096, China. Tel./fax: +86 25 83794974.
*E-mail address:* cysun@seu.edu.cn (C. Sun).

http://dx.doi.org/10.1016/j.knosys.2014.12.007
0950-7051/© 2014 Elsevier B.V. All rights reserved.

to answer a significant question: how far the classification hyperplane should be moved towards the majority class? This study solves this puzzle in the context of support vector machine (SVM) [30]. SVM is a robust classifier and is relatively insensitive to class imbalance in comparison with many other classification algorithms, because its classification hyperplane only associates with a few support vectors [31].

In this paper, we first investigate the reason that the classification performance of SVM can be destroyed by skewed classification data in theory, and then we analyze the merits and drawbacks of some existing SVM-based bias correction techniques. Next, the computational formula of the moving distance in SVM-THR algorithm proposed by Lin and Chen [21] is intensively modified to lead to one optimized version (SVM-OTHR). Furthermore, we incorporate SVM-OTHR into Bagging ensemble learning framework and present a novel classification algorithm named EnSVM-OTHR. In particular, to avoid overfitting and to guarantee the diversity of different individuals, a small random perturbation term is inserted into each SVM-OTHR to disturb the final position of classification hyperplane. Finally, we compare the two proposed classification algorithms with many state-of-the-art imbalanced classifiers on 30 data sets acquired from Keel data set Repository via non-parametrical statistical testing [32,33], indicating their superiority.

The rest of this paper is organized as follows. In Section 2, we introduce SVM theory and explain the reason that the performance of SVM can be damaged by imbalanced classification data. Section 3 briefly reviews some existing SVM-based class imbalance correction techniques and indicates their pros and cons. In Section 4, one optimized SVM decision threshold adjustment strategy (SVM-OTHR) and its extended version based on ensemble learning (EnSVM-OTHR) are described in detail. Experimental results and discussions are presented in Section 5. Finally in Section 6, the main contributions of this study are summarized.

## 2. Why SVM can be damaged by class imbalance ?

Support vector machine (SVM), which comes out of the theory of structure risk minimization, has several merits as follows: high generalization capability, absence of local minima and adaptation for high-dimension and small sample data [31,34].

Given some training data $D$, a set of $m$ points of the form: $D = \{(x_i, y_i) | x_i \in \mathbb{R}^p, y_i \in \{-1, 1\}_{i=1}^m\}$, where $y_i$ is either 1 or −1, indicating the class to which the point $x_i$ belongs. Each $x_i$ is one $p$-dimensional real vector. SVM is used to find the maximum margin hyperplane that divides the points having $y_i = 1$ from those having $y_i = -1$. The decision function of SVM is described as:

$$h(x) = \langle w, \phi(x) \rangle + b \tag{1}$$

where $\phi(x)$ represents a mapping of sample $x$ from the input space to high-dimensional feature space, $\langle \cdot, \cdot \rangle$ denotes the dot product in the feature space, $w$ denotes the weight vector for learned decision hyperplane and $b$ is the model bias. We can optimize the values of $w$ and $b$ by solving the following optimization problem:

$$\begin{aligned} &\text{minimize}: \ g(w, \xi) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}\xi_i \\ &\text{subject to}: \ y_i(\langle w, \phi(x_i) \rangle + b) \geqslant 1 - \xi_i, \quad \xi_i \geqslant 0 \end{aligned} \tag{2}$$

where $\xi_i$ is the $i$th slack variable and $C$ is regularization parameter (penalty factor) which is used to regulate the relationship between training accuracy and generalization. Then the minimization problem in formula (2) can be transformed to a dual form and be rewritten as:

$$\begin{aligned} &\text{maximize}: \ W(\alpha) = \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} y_i y_j \alpha_i \alpha_j K(x_i, x_j) \\ &\text{subject to}: \ \sum_{i=1}^{m} y_i \alpha_i = 0, \quad \forall i: \ 0 \leqslant \alpha_i \leqslant C \end{aligned} \tag{3}$$

where $\alpha_i$ is the sample $x_i$'s lagrange multiplier, $K(\cdot, \cdot)$ is a kernel function that maps the input vectors into a suitable feature space:

$$K(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle \tag{4}$$

Some previous work has found that radial basis kernel function (RBF) generally provides better classification accuracy than many other kernel functions [31,34]. RBF kernel is presented as follows:

$$K(x_i, x_j) = \exp\left\{ -\frac{\|x_i - x_j\|^2}{2\sigma^2} \right\} \tag{5}$$

where $\sigma$ is the width of RBF kernel.

Although previous work found that SVM is more robust to class imbalance than many other machine learning methods as its classification hyperplane only associates with a few support vectors, it can be still hurt by skewed class distributions to some extent. We try to analyze its reason in theory.

After training an SVM classifier, lagrange multiplier $\alpha_i$ can be divided into three categories as follows:

**Case 1:** $\alpha_i = 0$, it means the instance $x_i$ is classified accurately.
**Case 2:** $0 < \alpha_i < C$, the corresponding instance $x_i$ is called a normal support vector which is exactly on one of the margin hyperplanes.
**Case 3:** $\alpha_i = C$, $x_i$ is called a boundary support vector that lies between margins. The percentage of boundary support vectors reflects the error rate of SVM to some extent.

Suppose $N^+$ and $N^-$ represent the number of instances belonging to the positive class (minority class) and the negative class (majority class), respectively. $N_{sv}^+$ and $N_{sv}^-$ are the number of support vectors (including normal and boundary support vectors) in two classes, while $N_{\text{boundary}}^+$ and $N_{\text{boundary}}^-$ represent the number of boundary support vectors in two classes, respectively. According to formula (3), we can get:

$$\sum_{i=1}^{m}\alpha_i = \sum_{y_i=+1}\alpha_i + \sum_{y_i=-1}\alpha_i \tag{6}$$

$$\sum_{y_i=+1}\alpha_i = \sum_{y_i=-1}\alpha_i \tag{7}$$

Because $\alpha_i$'s value is $C$ at most, it can deduce two inequalities as follows:

$$\sum_{y_i=+1}\alpha_i \geqslant N_{\text{boundary}}^+ \times C \tag{8}$$

$$\sum_{y_i=+1}\alpha_i \leqslant N_{sv}^+ \times C \tag{9}$$

By integrating formula (8) and (9), we get:

$$N_{sv}^+ \times C \geqslant \sum_{y_i=+1}\alpha_i \geqslant N_{\text{boundary}}^+ \times C \tag{10}$$

Similarly, it is not difficult to get the following inequality:

$$N_{sv}^- \times C \geqslant \sum_{y_i=-1}\alpha_i \geqslant N_{\text{boundary}}^- \times C \tag{11}$$

Suppose $\sum_{y_i=+1}\alpha_i = \sum_{y_i=-1}\alpha_i = M$, if formula (10) and (11) respectively divide by $N^+ \times C$ and $N^- \times C$, we get:

$$\frac{N_{sv}^{+}}{N^{+}} \geqslant \frac{M}{N^{+} \times C} \geqslant \frac{N_{\text{boundary}}^{+}}{N^{+}} \tag{12}$$

$$\frac{N_{sv}^{-}}{N^{-}} \geqslant \frac{M}{N^{-} \times C} \geqslant \frac{N_{\text{boundary}}^{-}}{N^{-}} \tag{13}$$

where $\frac{N_{\text{boundary}}^{+}}{N^{+}}$ and $\frac{N_{\text{boundary}}^{-}}{N^{-}}$ approximately reflect the upper bounds of error rates in the minority class and the majority class, respectively. It is clear that $\frac{M}{N^{+}\times C}$ is larger than $\frac{M}{N^{-}\times C}$ as $N^{+}$ is smaller than $N^{-}$. Therefore, it is easy to draw a conclusion that the error rate of the minority class is usually larger than that of the majority class. Based on the analysis above, we find that the minority class often sacrifices more in the process of modeling SVM classifier, thus the impartiality of SVM can be destroyed by class imbalance.

## 3. A brief review of SVM-based class imbalance correction techniques

In previous work, some class imbalance correction strategies for SVM classifier have been proposed, including resampling [11,13], weighting [17,19] and decision threshold adjustment [21].

Resampling can be accomplished either by oversampling the minority class or undersampling the majority class. In fact, resampling repairs the difference between $\frac{M}{N^{+}\times C}$ and $\frac{M}{N^{-}\times C}$ by either increasing $N^{+}$ or reducing $N^{-}$. However, both techniques have their advantages and disadvantages. Oversampling makes the classifier overfitting and increases the time of modeling, while undersampling often causes information loss [35]. Random oversampling (ROS) and random undersampling (RUS) are the simplest resampling approaches [11]. Akbani et al. [13] found that combining SVM classifier and SMOTE oversampling method which proposed by Chawla et al. [12] can acquire better classification results than ROS and RUS.

Different from resampling, the weighting strategy (CS-SVM) assigns different weights to the instances in different classes, where the weight reflects in the penalty factor $C$ [17]. That means for the samples in the minority class, the penalty factor is assigned as $C^{+}$, while the penalty factor in the majority class is assigned as $C^{-}$. To guarantee $\frac{M}{N^{+}\times C^{+}} = \frac{M}{N^{-}\times C^{-}}$, we should make $\frac{C^{+}}{C^{-}} = \frac{N^{-}}{N^{+}}$, then the formula (2) can be rewritten as:

$$\begin{aligned} &\text{minimize:}\;\; g(w,\xi) = \frac{1}{2}\|w\|^{2} + C^{+}\sum_{y_i=+1}\xi_i + C^{-}\sum_{y_i=-1}\xi_i \\ &\text{subject to:}\;\; y_i(\langle w, \phi(x_i)\rangle + b) \geqslant 1-\xi_i, \quad \xi_i \geqslant 0 \end{aligned} \tag{14}$$

z-SVM can be regarded as another weighting strategy of SVM [19]. Unlike CS-SVM that assigns different penalty factors for different classes, z-SVM directly rewrites the final decision function (formula (1)) by assigning a larger weight for the positive class. According to Ref. [19], formula (1) can be rewritten as:

$$h(x) = z \times \langle w, \phi(x_{\text{p}})\rangle + \langle w, \phi(x_{\text{N}})\rangle + b \tag{15}$$

where $x_{\text{p}}$ and $x_{\text{N}}$ denote the positive and negative support vectors in one learned SVM model, and $z$ is weighted factor for the positive class. In Ref. [19], G-mean is adopted as the evaluation measure to determine the optimal $z$ value $z^{*}$. Meanwhile, as one univariate unconstrained optimization problem, golden section optimization technology is adopted into the optimization process of $z^{*}$.

The third method is decision threshold adjustment that is used after classifier modeling. Actually, the decision threshold adjustment method improves the classification accuracy of the minority class by directly moving the classification hyperplane towards the majority class. Lin and Chen [21] suggest to use the following function to calculate the moving distance of decision threshold:

$$\theta = \frac{N^{-} - N^{+}}{N^{+} + N^{-} + 2} \tag{16}$$

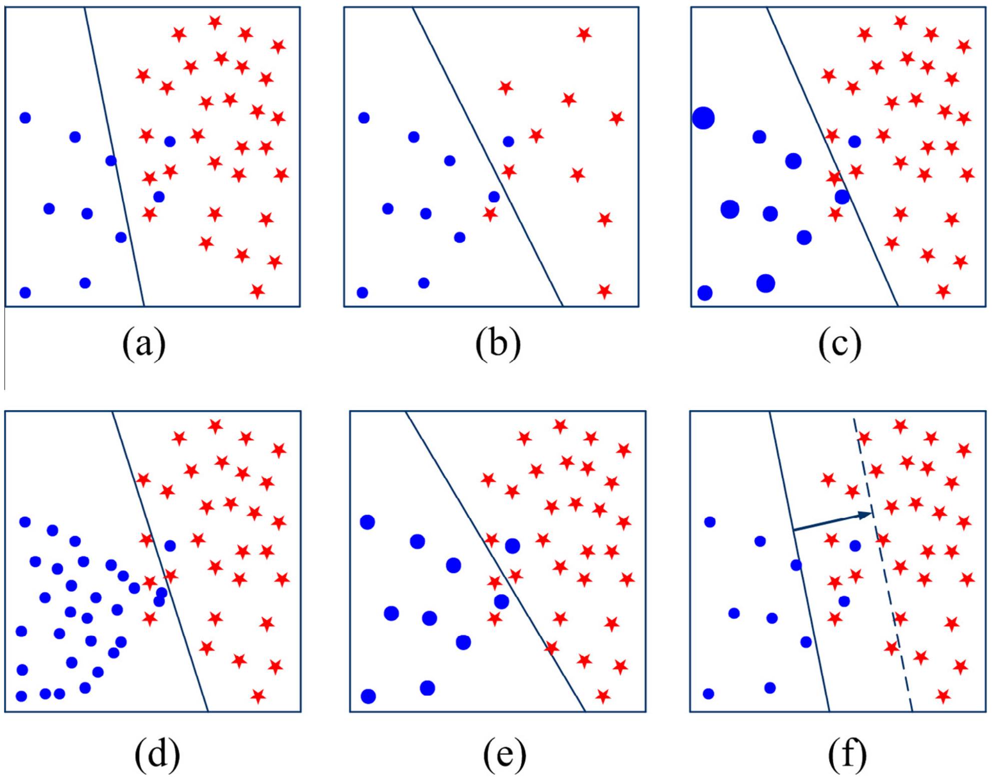


**Fig. 1.** Graphical representations of original SVM and SVMs based on five different correction technologies for class imbalance problem, where (a) original SVM modeling; (b) SVM-RUS modeling; (c) SVM-ROS modeling; (d) SVM-SMOTE modeling; (e) CS-SVM/z-SVM modeling and (f) SVM-THR modeling. The circle points denote positive samples and the asterisk points represent negative examples, respectively. The size of each circle point indicates its weight.

For each test instance $x$, it can be divided into the minority class or the majority class by the adjusted decision function as follows:

$$h'(x) = h(x) + \theta \tag{17}$$

where $h(x)$ and $h'(x)$ represent the original decision function and the adjusted decision function, respectively. They named this adjustment strategy as SVM-THR and found it often performs better than many other correction techniques by experiments on several Bioinformatics data sets. However, the formula (16) was given empirically and the authors failed to provide its theoretical foundation. Then a confusing question emerges: how to find the optimal moving distance for the decision threshold? In Section 4, we will exhibit an novel optimized method (SVM-OTHR) which may be a potential answer for the question above.

Fig. 1 gives the graphical representations of standard SVM, SVM-RUS, SVM-ROS, SVM-SMOTE, CS-SVM/z-SVM and SVM-THR, respectively. It can be seen that each correction technique can alleviate the affection of class imbalance more or less.

## 4. Optimized decision threshold adjustment strategy and its evolving version in ensemble environment

To adjust decision threshold (classification boundary) to the optimal position, the concept of *optimization* should be primarily defined. Obviously, for balanced classification tasks, SVM can spontaneously find the optimal position of classification boundary which is a tradeoff between classification margin and classification accuracy. But when the classification task is skewed, classification accuracy generally gives bias evaluation, thus some other specific evaluation metrics, such as F-measure and G-mean, are needed to evaluate the classification performance of a learner. F-measure and G-mean can be regarded as functions of the confusion matrix as shown in Table 1.

They are calculated as follows:

$$\text{F-measure} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{18}$$

**Table 1**
Confusion matrix.

| | Predicted positive class | Predicted negative class |
|---|---|---|
| Actual positive class | TP (True Positive) | FN (False Negative) |
| Actual negative class | FP (False Positive) | TN (True Negative) |

$$\text{G-mean} = \sqrt{\text{TPR} \times \text{TNR}} \tag{19}$$

where Precision, Recall, TPR and TNR are further defined as:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{20}$$

$$\text{Recall} = \text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{21}$$

$$\text{TNR} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{22}$$

From the formula (19), it obviously expresses that G-mean reflects the balance between the accuracy of the minority class and that of the majority class. Therefore, G-mean can be regarded as an excellent criterion to determine the optimal position of the classification hyperplane. Unfortunately, it is difficult to find this optimal hyperplane taking advantage of direct optimization technology by rewriting the formula (2). Here, we give an iterative algorithm that can be also seen as an exhaustive search algorithm, to search the optimal position for the classification hyperplane. Considering each misclassified minority class instance in the training set, undoubtedly, its decision function (see formula (1)) output value will be negative. Therefore, we can acquire all candidate positions for the optimal classification hyperplane according to the output values of misclassified instances. By comparing G-mean values of these candidate positions, it is easy to find the optimal adjusted position. Furthermore, to maintain the generalization ability of the classifier to some extent, we profit from the idea of SVM to adjust each candidate position to the middle between the corresponding misclassified minority class instance $x_i$ and its nearest neighbor instance belonging to the majority class that lies far from the original classification hyperplane, denoting as $nx_i$. Then the adjusted distance $\theta_i$ can be calculated with the following formula:

$$\theta_i = \frac{-h(x_i) - h(nx_i)}{2} \tag{23}$$

where $h$ is the original decision function. Then we compare G-mean values of all candidate positions to find the optimal position and its corresponding adjusted distance, representing as $\theta_{\text{optimal}}$. Finally, the decision function is adjusted as:

$$h'(x) = h(x) + \theta_{\text{optimal}} \tag{24}$$

where $h(x)$ and $h'(x)$ are original and adjusted decision functions, respectively. Fig. 2 gives the schematic diagram of the search process.

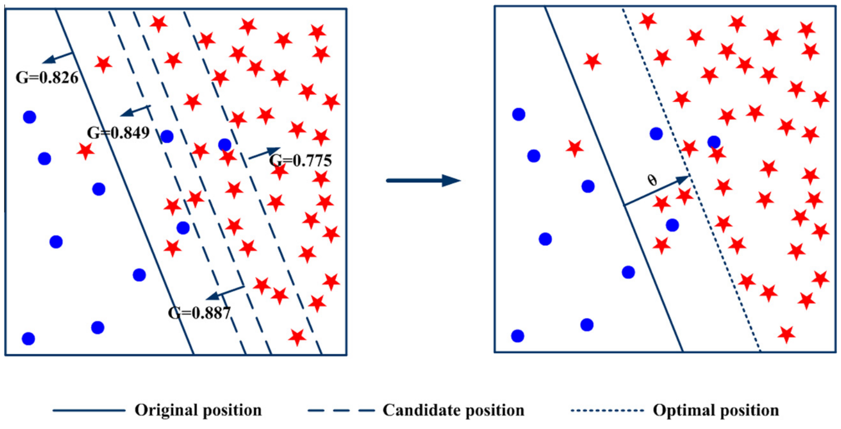


**Fig. 2.** Schematic diagram of search process for the optimal classification hyperplane, where the circle points denote positive samples and the asterisk points represent negative examples, respectively. In left subgraph, the G-mean value of each candidate position is calculated. By ranking G-mean values in descending order, the optimal position can be found and it is given in right subgraph, where $\theta$ represents the moving distance from the original position of hyperplane to the optimal position.

```
Input: Minority class training set $S^+$; Majority class training set $S^-$; Test instance $x'$.
Training Process:
1.  Integrating $S^+$ and $S^-$ to train a SVM classifier $I$;
2.  Compute G-mean value of the training set $G_0$ using classifier $I$;
3.  Find all misclassified instances in $S^+$ and record its number as $M$, then
    rank them in descending order based on decision function output values;
4.  for $i$=1:$M$
  4.1  Extract the $i$th misclassified minority class sample $x_i$;
  4.2  Find its nearest neighbor of majority class $nx_i$ at the side far from the
       original classification hyperplane;
  4.3  Calculate the $i$th adjustment distance by the following formula:
                 $\theta_i=(-h(x_i)-h(nx_i))/2$
       where $h$ is the original decision function;
  4.4  Compute G-mean value of the training set $G_i$ for the $i$th adjusted
       classification hyperplane;
    end for
5.  Rank $G_0, G_1,\ldots,G_M$ in descending order;
6.  Find $\theta_{\text{optimal}}$ corresponding to the highest G-mean value;
7.  The final decision threshold $h'$ can be calculated by:
                 $h' = h+\theta_{\text{optimal}}$
Testing Process:
1.  Put $x'$ into the classifier $I$ and get decision output value $h(x')$;
2.  Compute the adjusted output value $h'(x')= h(x')+ \theta_{\text{optimal}}$;
3.  According to the value of $h'(x')$ to determine the class label of the test
    instance $x'$;
Output: $y'$ which is the predicted class label for test instance $x'$.
```

**Fig. 3.** Pseudo code description of SVM-OTHR algorithm.

According to the idea above, we also present a novel SVM decision threshold adjustment algorithm that is named SVM-OTHR. The pseudo code description of SVM-OTHR algorithm is provided in Fig. 3.

Fig. 3 indicates that the time complexity of SVM-OTHR is merely correlated with the number of misclassified instances belonging to the minority class, thus it will not increase computational burden to a large extent. Of course, from Fig. 3, we can also observe one obvious drawback of SVM-OTHR that is the appearance of overfitting and/or the decrease of generalization ability. The algorithm can be only local optimization for the limited training samples, as well the adjusted classification hyperplane cannot guarantee to minimize the structural risk.

To overcome the defects described above, an extended ensemble learning version of SVM-OTHR named as EnSVM-OTHR is proposed. As indicated in previous work [36,37], ensemble learning can help repair the bias of single classifier and improve the generalization ability of classification results. Here, we select Bagging ensemble learning framework [38] to develop EnSVM-OTHR algorithm. Bagging adopts Bootstrap strategy to generate multiple diverse training subsets and majority voting rule to make the final decision. EnSVM-OTHR simply inherits both Bootstrap strategy and majority voting rule from Bagging. To alleviate the impact of class imbalance, we carry out SVM-OTHR algorithm on each training subset. In addition, to further promote the generalization ability and guarantee the diversity of different base classifiers, a small random perturbation term is inserted into each SVM-OTHR to disturb the final position of classification hyperplane. It is notable that the value of random perturbation term should be moderate due to a too small value is helpless to increase the diversity among base classifiers and a too large value can destroy the performance of each base classifier. According to the feedback from a mass of experimental results, we empirically assign each random perturbation term in the range of $[-0.1 \times \theta_{\text{optimal}}, 0.1 \times \theta_{\text{optimal}}]$, then the actual moving distance $\theta'$ can be calculated as:

$$\theta' = \theta_{\text{optimal}} \times (1 + 0.1 \times \text{rand}) \tag{25}$$

where rand denotes one random number whose value is in the range of $[-1,1]$. The schematic diagram and pseudo code

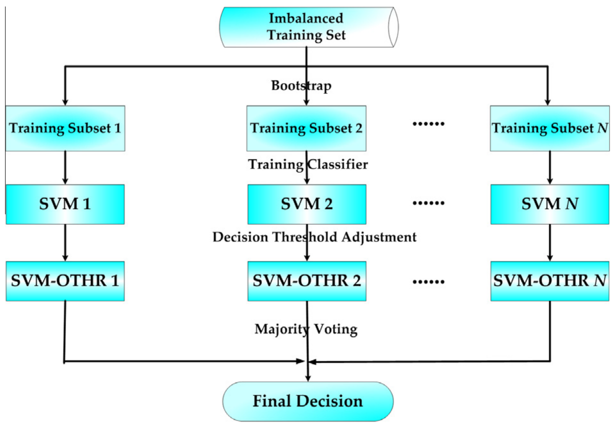


**Fig. 4.** Schematic diagram of EnSVM-OTHR algorithm.

```
Input: Training set $S$; Number of base classifiers $N$; Test instance $x$'.
Training Process:
For $i$=1:$N$
    Use Bootstrap technology on $S$ to generate training subset $S_i$;
    Train the $i$th SVM classifier SVM$_i$;
    Call SVM-OTHR algorithm to adjust the classification hyperplane of SVM$_i$ to form SVM-OTHR$_i$;
end
Testing Process:
For $i$=1:$N$
    Put $x$' into the classifier SVM-OTHR$_i$ to predict its class label $y_i$';
end
Use majority voting to get the final class label $y$';
Output: $y$' which is the predicted class label for test instance $x$'.
```

**Fig. 5.** Pseudo code description of EnSVM-OTHR algorithm.

description of EnSVM-OTHR algorithm are presented in Figs. 4 and 5, respectively.

## 5. Results and discussions

We tested the proposed algorithms on 30 Keel data sets [39]. Similar to much previous work, we focused on binary-class imbalanced problem, too. Information about these data sets is summarized in Table 2.

Statistical analysis is an efficient means to detect the effectiveness of results in machine learning, e.g., Wang et al. [40] used statistical analysis results of instances to validate concept matching technologies, acquiring excellent performance. Therefore, to compare multiple classification algorithms on lots of data sets, we also provided statistical analysis results. Specifically, Friedman test [32,33] is used to detect statistical differences among a group of results, and the Holm post hoc test [33] is adopted to examine whether the proposed algorithm is distinctive among a $1 \times N$ comparison. The post hoc procedure allows us to know whether a hypothesis of comparison of means could be rejected at a specified level of significance $\alpha$. The adjusted $p$-value (APV) is also computed to denote the lowest level of significance of a hypothesis that results in a rejection. Furthermore, we consider the average rankings of the algorithms in order to measure how good a method is with respect to its partners. This ranking is obtained by assigning a position to each algorithm depending on its performance on each data set. The algorithm which achieves the best accuracy on a specific data set will have the first ranking (value 1); then, the algorithm with the second best accuracy is assigned rank 2, and so forth. This task is carried out for all data sets and finally an average ranking is calculated. The procedure of our statistical tests is the

**Table 2**
Data sets used in this article.

| Dataset | Number of instances | Number of attributes | Imbalance ratio |
|---|---|---|---|
| glass1 | 214 | 9 | 1.82 |
| wisconsin | 683 | 9 | 1.86 |
| pima | 768 | 8 | 1.87 |
| haberman | 306 | 3 | 2.78 |
| vehicle1 | 846 | 18 | 2.9 |
| new-thyroid1 | 215 | 5 | 5.14 |
| yeast3 | 1484 | 8 | 8.1 |
| ecoli3 | 336 | 7 | 8.6 |
| vowel0 | 988 | 13 | 9.98 |
| yeast-1_vs_7 | 459 | 7 | 14.3 |
| ecoli4 | 336 | 7 | 15.8 |
| abalone9-18 | 731 | 8 | 16.4 |
| shuttle-c2-vs-c4 | 129 | 9 | 20.9 |
| yeast4 | 1484 | 8 | 28.1 |
| yeast5 | 1484 | 8 | 32.73 |
| abalone19 | 4174 | 8 | 129.44 |
| ecoli-0-3-4_vs_5 | 200 | 7 | 9 |
| ecoli-0-6-7_vs_3-5 | 222 | 7 | 9.09 |
| yeast-0-3-5-9_vs_7-8 | 506 | 8 | 9.12 |
| yeast-0-3-5-9_vs_7-8 | 1004 | 8 | 9.14 |
| ecoli-0-1_vs_2-3-5 | 244 | 7 | 9.17 |
| ecoli-0-6-7_vs_5 | 220 | 6 | 10 |
| led7digit-0-2-4-5-6-7-8-9_vs_1 | 443 | 7 | 10.97 |
| ecoli-0-1_vs_5 | 240 | 6 | 11 |
| shuttle-6_vs_2-3 | 230 | 9 | 22 |
| flare-F | 1066 | 11 | 23.79 |
| winequality-red-4 | 1599 | 11 | 29.17 |
| shuttle-2_vs_5 | 3316 | 9 | 66.67 |
| poker-8-9_vs_5 | 2075 | 10 | 82 |
| poker-8_vs_6 | 1477 | 10 | 85.88 |

same as Ref. [41]. Additional information and software about these statistical tests can be found on the web: http://sci2s.ugr.es/sicidm/.

First, we tested the proposed SVM-OTHR algorithm in comparison with seven other single classification algorithms based on SVM, including:

1. Standard SVM (SSVM) [30]: SVM classifier without any class imbalance correction technologies.
2. SVM with random undersampling (SVM-RUS): It firstly adopts RUS [11] to preprocess the original training set, then trains SVM classifier using the balanced training data.
3. SVM with random oversampling (SVM-ROS): It firstly adopts ROS [11] to preprocess the original training set, then trains SVM classifier using the balanced training data.
4. SVM with SMOTE [12] (SVM-SMOTE): It firstly adopts SMOTE algorithm to oversampling the minority class instances and to make the data set balance, then trains SVM classifier using the balanced training data. Based on the recommendation in Ref. [12], the number of neighbors $K$ is assigned as 5.
5. Weighted SVM [17] (CS-SVM): It assigns different values for the penalty factor $C$ belonging to different classes, to guarantee the fairness of classifier modeling, we make $C^+/C^- = N^-/N^+$, where $C^-$ is tuned by grid search.
6. z-SVM [19]: This is the algorithm proposed by Imam et al. [19]. Golden section search algorithm was used to find $z^*$ that is the optimal $z$ value which corresponds to the maximal G-mean value on the training set.
7. SVM with decision threshold adjustment [21] (SVM-THR): This is the algorithm proposed by Lin and Chen [21]. We simply used their default function (see formula (16)) to move the classification hyperplane of SVM towards the majority class.

In the experiments, for each data set, we performed a fivefold cross validation. In view of the randomness of classification results, we randomly repeated the cross-validation process for ten times and provided the results in the form of mean ± standard deviation. In addition, all algorithms were implemented in Matlab 2013a running environment and SVM was realized by libSVM toolbox with the version of 3.17 [42]. Specifically, for SVM, RBF kernel function was adopted, as well the penalty factor $C$ and the width parameter $\sigma$ of RBF kernel function were tuned by using grid search with fivefold cross-validation, where $C \in [2^{-2}, 2^{-1}, \ldots, 2^{10}]$, $\sigma \in [2^{-6}, 2^{-5}, \ldots, 2^{5}]$. We evaluated these eight classification algorithms by two widely used evaluation criterions in imbalanced classification tasks: F-measure and G-mean. The F-measure and G-means values of each algorithm on each data set are summarized in Tables 3 and 4, respectively. The statistical results are provided in Table 5.

From Tables 3 and 4, we observed that almost all bias correction strategies can promote F-measure and G-mean values compared with standard SVM except SVM-RUS on F-measure evaluation criterion. We consider that the classification hyperplane of RUS can be exceedingly adjusted towards the majority class, consequently causing one low Precision value. Also, we found that two oversampling technologies often perform better on those datasets with relatively high class imbalance ratio, while RUS shows better performance on those ones with relatively low imbalance ratio. The same phenomenon have been observed in Refs. [35,43]. Actually, for highly imbalanced data sets, RUS tends to lose much information, while for relatively balanced data sets, oversampling is apt to be overfitting.

Another interesting phenomenon observed from Tables 3 and 4 is that F-measure and G-mean values not just associate with class imbalance ratio, e.g., on shuttle-2_vs_5 dataset which has a imbalance ratio of 66.67, most classifiers including standard SVM can acquire 100% classification accuracy. As indicated in Refs. [35,44], the destroy of class imbalance should be estimated by examining several different factors, including class overlapping, the size of training instances, class imbalance ratio, noisy level and small disjuncts, etc. In our previous work [45], one pre-estimated strategy

**Table 3**
F-measure values of eight compared single classifiers on 30 Keel data sets, where bold indicates the best result on each data set.

| Dataset | SSVM | SVM-RUS | SVM-ROS | SVM-SMOTE | CS-SVM | z-SVM | SVM-THR | SVM-OTHR |
|---|---|---|---|---|---|---|---|---|
| glass1 | .6615 ± .0318 | .6374 ± .0284 | .6543 ± .0175 | .6408 ± .0241 | .6227 ± .0219 | .6578 ± .0299 | .6690 ± .0184 | **.6715 ± .0182** |
| wisconsin | .9411 ± .0047 | .9441 ± .0033 | .9420 ± .0032 | .9413 ± .0059 | .9435 ± .0045 | .9431 ± .0047 | **.9461 ± .0039** | .9442 ± .0052 |
| pima | .5687 ± .0145 | .5829 ± .0209 | .5578 ± .0243 | .5628 ± .0125 | .5432 ± .0148 | .5672 ± .0217 | .5796 ± .0147 | **.5982 ± .0166** |
| haberman | .2687 ± .0336 | .4238 ± .0230 | .4048 ± .0234 | .4048 ± .0332 | .4227 ± .0175 | **.4507 ± .0318** | .3638 ± .0381 | .4432 ± .0297 |
| vehicle1 | .6424 ± .0211 | .6449 ± .0149 | .6421 ± .0135 | .6453 ± .0168 | .6458 ± .0183 | .6628 ± .0179 | **.6674 ± .0167** | .6506 ± .0209 |
| new-thyroid1 | .9194 ± .0228 | .8806 ± .0395 | .9301 ± .0418 | .9101 ± .0250 | .9124 ± .0220 | .9228 ± .0375 | .9349 ± .0232 | **.9351 ± .0269** |
| yeast3 | **.7363 ± .0157** | .6334 ± .0149 | .6735 ± .0125 | .7019 ± .0095 | .6684 ± .0103 | .7085 ± .0107 | .7183 ± .0149 | .7106 ± .0091 |
| ecoli3 | .6125 ± .0511 | .5216 ± .0254 | .5720 ± .0315 | .5984 ± .0532 | .5644 ± .0380 | .6127 ± .0268 | .5977 ± .0200 | **.6345 ± .0226** |
| vowel0 | .9985 ± .0049 | .8691 ± .0257 | .9979 ± .0052 | .9994 ± .0018 | .9995 ± .0017 | **1.00 ± 0.00** | .9971 ± .0058 | **1.00 ± 0.00** |
| yeast-1_vs_7 | **.4173 ± .0734** | .1887 ± .0223 | .2511 ± .0558 | .2308 ± .0360 | .2913 ± .0414 | .3018 ± .0469 | .2991 ± .0429 | .2714 ± .0330 |
| ecoli4 | .7257 ± .1258 | .5943 ± .0573 | .6931 ± .0791 | **.7502 ± .0658** | .7046 ± .1228 | .6923 ± .1066 | .6401 ± .0754 | .7058 ± .0718 |
| abalone9-18 | .4044 ± .0649 | .2512 ± .0358 | .3438 ± .0466 | .3434 ± .0256 | .3490 ± .0363 | .3476 ± .0398 | .3896 ± .0502 | **.4167 ± .0546** |
| shuttle-c2-vs-c4 | .9931 ± .0021 | .9949 ± .0033 | .9924 ± .0025 | **.9942 ± .0030** | **.9942 ± .0021** | .9898 ± .0075 | .9918 ± .0010 | .9924 ± .0023 |
| yeast4 | .3303 ± .0237 | .1958 ± .0133 | .3455 ± .0289 | .3157 ± .0252 | .3517 ± .0316 | .3557 ± .0222 | **.3772 ± .0241** | .3635 ± .0182 |
| yeast5 | .6462 ± .0339 | .4660 ± .0444 | **.7057 ± .0249** | .6907 ± .0415 | .6950 ± .0390 | .6825 ± .0312 | .6344 ± .0293 | .6823 ± .0355 |
| abalone19 | 0.00 ± 0.00 | .0307 ± .0046 | **.0507 ± .0136** | .0389 ± .0119 | .0461 ± .0205 | .0438 ± .0255 | .0499 ± .0109 | .0479 ± .0074 |
| ecoli-0-3-4_vs_5 | .8280 ± .0549 | .6395 ± .0770 | **.8290 ± .0613** | .7789 ± .0584 | .8220 ± .0741 | .8147 ± .0751 | .6866 ± .0499 | .8246 ± .0664 |
| ecoli-0-6-7_vs_3-5 | .6756 ± 0778 | .5497 ± .0345 | .6795 ± .0538 | .6196 ± .0882 | .7078 ± .0772 | .6927 ± .0743 | .6772 ± .0583 | **.7110 ± .0720** |
| yeast-0-3-5-9_vs_7-8 | .2949 ± .0421 | .2605 ± .0246 | .2899 ± .0373 | .2792 ± .0430 | .3165 ± .0401 | **.3221 ± .0377** | .3142 ± .0247 | .3179 ± .0302 |
| yeast-0-2-5-7-9_vs_3-6-8 | **.7982 ± .0140** | .5952 ± .0271 | .6901 ± .0150 | .7090 ± .0205 | .6869 ± .0126 | .6848 ± .0374 | .7608 ± .0177 | .7091 ± .0179 |
| ecoli-0-1_vs_2-3-5 | .6693 ± .0708 | .5344 ± .0505 | .6413 ± .0821 | **.6813 ± .0465** | .6186 ± .0524 | .6431 ± .0603 | .6142 ± .0512 | .6654 ± .0621 |
| ecoli-0-6-7_vs_5 | .7213 ± .0360 | .4952 ± .0308 | .7411 ± .0497 | .6483 ± .0823 | .7226 ± .0732 | .7398 ± .0802 | .7228 ± .0485 | **.7590 ± .0714** |
| led7digit-0-2-4-5-6-7-8-9_vs_1 | .6477 ± .0249 | .6299 ± .0373 | .6389 ± .0421 | **.7109 ± .0293** | .6582 ± .0324 | .6635 ± .0411 | .6903 ± .0428 | .6962 ± .0942 |
| ecoli-0-1_vs_5 | .8074 ± .0602 | .6766 ± .0308 | .7956 ± .0585 | .7515 ± .0386 | .8108 ± .0738 | .8055 ± .0628 | .6673 ± .0393 | **.8258 ± .0365** |
| shuttle-6_vs_2-3 | .7806 ± .1590 | .5582 ± .0792 | .7105 ± .1515 | **.8770 ± .0598** | .8200 ± .1677 | .8442 ± .1468 | .7699 ± .1073 | .8175 ± .1539 |
| flare-F | .1117 ± .0310 | .1990 ± .0166 | .1759 ± .0311 | .1946 ± .0310 | .1628 ± .0285 | .1952 ± .0260 | .2216 ± .0413 | **.2454 ± .0331** |
| winequality-red-4 | .0978 ± .0362 | .1071 ± .0082 | .1635 ± .0397 | .1392 ± .0333 | .1585 ± .0332 | .1495 ± .0308 | .1562 ± .0234 | **.1685 ± .0336** |
| shuttle-2_vs_5 | **1.00 ± 0.00** | .9980 ± .0063 | **1.00 ± 0.00** | **1.00 ± 0.00** | **1.00 ± 0.00** | **1.00 ± 0.00** | .9534 ± .0134 | **1.00 ± 0.00** |
| poker-8-9_vs_5 | .0161 ± .0301 | .0353 ± .0042 | .0447 ± .0351 | .0923 ± .0532 | .0788 ± .0248 | .0819 ± .0454 | **.0934 ± .0240** | .0797 ± .0389 |
| poker-8_vs_6 | .3432 ± .1021 | .2514 ± .0330 | .3738 ± .0642 | **.6464 ± .0820** | .4515 ± .1059 | .4244 ± .1152 | .3532 ± .0150 | .3818 ± .0703 |

**Table 4**
G-mean values of eight compared single classifiers on 30 Keel data sets, where bold indicates the best result on each data set.

| Dataset | SSVM | SVM-RUS | SVM-ROS | SVM-SMOTE | CS-SVM | z-SVM | SVM-THR | SVM-OTHR |
|---|---|---|---|---|---|---|---|---|
| glass1 | .7357 ± .0222 | .7143 ± .0240 | .7323 ± .0174 | .7160 ± .0217 | .7056 ± .0209 | .7328 ± .0198 | .7104 ± .0128 | **.7376 ± .0139** |
| wisconsin | .9592 ± .0035 | .9602 ± .0028 | .9592 ± .0029 | .9599 ± .0055 | .9612 ± .0038 | .9517 ± .0031 | .9649 ± .0027 | **.9660 ± .0036** |
| pima | .6611 ± .0122 | .6701 ± .0186 | .6516 ± .0189 | .6578 ± .0102 | .6386 ± .0125 | .6627 ± .0174 | .6695 ± .0114 | **.6781 ± .0137** |
| haberman | .4034 ± .0521 | .5332 ± .0202 | .5657 ± .0199 | .5676 ± .0292 | .5249 ± .0208 | .5411 ± .0278 | .5166 ± .0339 | **.6063 ± .0269** |
| vehicle1 | .7486 ± .0168 | .7820 ± .0108 | .7466 ± .0096 | .7499 ± .0127 | .7507 ± .0163 | .7478 ± .0129 | **.7970 ± .0136** | .7865 ± .0170 |
| new-thyroid1 | .9611 ± .0187 | .9584 ± .0172 | .9613 ± .0333 | .9570 ± .0197 | .9545 ± .0166 | .9602 ± .0235 | **.9811 ± .0110** | .9627 ± .0185 |
| yeast3 | .8333 ± .0125 | .8413 ± .0123 | .8638 ± .0122 | .8659 ± .0084 | .8702 ± .0073 | .8891 ± .0129 | .8850 ± .0088 | **.8917 ± .0079** |
| ecoli3 | .7692 ± .0476 | .8403 ± .0221 | .7961 ± .0266 | .7802 ± .0742 | .7929 ± .0426 | .8288 ± .0546 | .8263 ± .0209 | **.8512 ± .0221** |
| vowel0 | .9985 ± .0047 | .9833 ± .0048 | .9979 ± .0050 | .9994 ± .0018 | .9999 ± .0002 | .9962 ± .0012 | .9934 ± .0020 | **1.00 ± 0.00** |
| yeast-1_vs_7 | .5529 ± .1050 | **.6309 ± .0471** | .5270 ± .0914 | .5393 ± .0870 | .6237 ± .0743 | .6269 ± .0530 | .6156 ± .0628 | .6127 ± .0615 |
| ecoli4 | .7927 ± .1323 | **.9158 ± .0305** | .8071 ± .0819 | .8572 ± .0487 | .7904 ± .1348 | .8427 ± .0633 | .8477 ± .0776 | .8671 ± .0734 |
| abalone9-18 | .5574 ± .0656 | **.7315 ± .0381** | .5921 ± .0564 | .5925 ± .0258 | .5968 ± .0417 | .6658 ± .0561 | .7226 ± .0604 | .7238 ± .0377 |
| shuttle-c2-vs-c4 | .9957 ± .0005 | .9979 ± .0020 | .9953 ± .0009 | .9962 ± .0015 | .9956 ± .0008 | .9951 ± .0016 | **.9994 ± .0000** | .9960 ± .0007 |
| yeast4 | .5097 ± .0308 | **.7763 ± .0127** | .6970 ± .0341 | .6831 ± .0430 | .6777 ± .0310 | .7268 ± .0415 | .7266 ± .0285 | .7595 ± .0258 |
| yeast5 | .8629 ± .0203 | **.9431 ± .0174** | .8968 ± .0200 | .8814 ± .0293 | .8773 ± .0241 | .8752 ± .0103 | .9074 ± .0285 | .8987 ± .0237 |
| abalone19 | 0.00 ± 0.00 | **.6441 ± .0625** | .3694 ± .0809 | .3461 ± .1027 | .2534 ± .0712 | .3277 ± .0961 | .2770 ± .0380 | .3487 ± .0715 |
| ecoli-0-3-4_vs_5 | .8782 ± .0540 | .8598 ± .0688 | .8631 ± .0660 | .8657 ± .0683 | .8511 ± .0757 | **.9165 ± .0882** | .8532 ± .0153 | .8688 ± .0660 |
| ecoli-0-6-7_vs_3-5 | .7677 ± .0770 | .8465 ± .0248 | .7615 ± .0667 | .7621 ± .0992 | .7856 ± .0859 | .7742 ± .0526 | **.8764 ± .0451** | .7877 ± .0714 |
| yeast-0-3-5-9_vs_7-8 | .4862 ± .0437 | .5702 ± .0350 | .5557 ± .0483 | .5428 ± .0499 | .5921 ± .0481 | .5417 ± .0648 | .6155 ± .0224 | **.6365 ± .0338** |
| yeast-0-2-5-7-9_vs_3-6-8 | .8662 ± .0086 | .8400 ± .0115 | .8461 ± .0105 | .8489 ± .0132 | .8452 ± .0127 | .8647 ± .0158 | **.8886 ± .0114** | .8842 ± .0072 |
| ecoli-0-1_vs_2-3-5 | .7808 ± .0630 | .7907 ± .0545 | .7621 ± .0726 | **.8209 ± .0245** | .7474 ± .0661 | .7602 ± .0527 | .7574 ± .0382 | .7969 ± .0552 |
| ecoli-0-6-7_vs_5 | .8066 ± .0486 | .8304 ± .0388 | .8296 ± .0543 | .7607 ± .1048 | .8082 ± .0712 | .8550 ± .0468 | **.8684 ± .0271** | .8568 ± .0526 |
| led7digit-0-2-4-5-6-7-8-9_vs_1 | .8469 ± .0213 | .8308 ± .0219 | .8159 ± .0545 | .8435 ± .0167 | .8398 ± .0258 | .8318 ± .0326 | .8057 ± .0174 | **.8508 ± .1073** |
| ecoli-0-1_vs_5 | .8530 ± .0671 | .8683 ± .0299 | .8490 ± .0616 | .8659 ± .0501 | .8637 ± .0707 | .8568 ± .0410 | .8587 ± .0253 | **.8943 ± .0210** |
| shuttle-6_vs_2-3 | .7868 ± .1606 | .7514 ± .1122 | .7198 ± .1492 | **.8871 ± .0641** | .8312 ± .1704 | .8068 ± .1070 | .8193 ± .1109 | .8249 ± .1525 |
| flare-F | .2294 ± .0510 | **.7694 ± .0236** | .5006 ± .0837 | .4643 ± .0455 | .4729 ± .0618 | .4791 ± .0254 | .5012 ± .0877 | .6396 ± .0585 |
| winequality-red-4 | .2127 ± .0808 | .4520 ± .0324 | .3818 ± .0695 | .3939 ± .0680 | .3682 ± .0629 | **.5602 ± .0680** | .5038 ± .0430 | .5401 ± .0701 |
| shuttle-2_vs_5 | **1.00 ± 0.00** | **1.00 ± 0.00** | **1.00 ± 0.00** | **1.00 ± 0.00** | **1.00 ± 0.00** | **1.00 ± 0.00** | .9992 ± .0002 | **1.00 ± 0.00** |
| poker-8-9_vs_5 | .0335 ± .0591 | .2192 ± .0580 | .3917 ± .0788 | **.5946 ± .1340** | .5111 ± .0552 | .4936 ± .0896 | .4973 ± .1077 | .5019 ± .0795 |
| poker-8_vs_6 | .3958 ± .1165 | .4247 ± .0978 | .5124 ± .0657 | **.6689 ± .0809** | .4893 ± .1978 | .5573 ± .0778 | .6151 ± .0970 | .6309 ± .0869 |

by scatter matrix-based class separability measure for estimating the harmfulness of class imbalance had been proposed.

Tables 3 and 4 also show that our proposed SVM-OTHR algorithm outperforms SVM-THR algorithm that was proposed by Lin and Chen [21] on majority data sets. Specifically, SVM-OTHR obtains the best F-measure and G-mean values on 12 and 11 data sets, respectively. While SVM-THR only acquires the best F-measure and G-mean values on 4 and 6 data sets. Actually, the experimental results have indicated that SVM-THR is quite competitive with the other bias correction strategies. SVM-OTHR, however, could find better position of decision hyperplane than SVM-THR as the adoption of priori knowledge (original distributions of training instances).

Table 5 shows the average ranking computed for all algorithms according to F-measure and G-mean, respectively. In addition, Table 5 also presents that the APV computed by the Holm post hoc test [33] reported between our algorithm and the other algorithms. We can observe that SVM-OTHR has obtained the lowest values in both ranking$_F$ and ranking$_G$, thus it is the best algorithm. Moreover, since all APVs are lower than a standard used level of significance of $\alpha = 0.05$, the null-hypothesis of equality is rejected in all cases, which supports the conclusion that SVM-OTHR outperforms the remaining algorithms. From Table 5, we also observed that z-SVM has acquired lower ranking$_F$ and ranking$_G$ values than CS-SVM. That means one optimized weight can be more advisable than one empirical weight in cost-sensitive learning.

Then we compared the running time of eight single classification algorithms, the results are summarized in Fig. 6 which shows that the proposed SVM-OTHR algorithm generally consumes a little more training time than SSVM, CS-SVM and SVM-THR, but much fewer in comparison with two oversampling strategies (SVM-ROS and SVM-SMOTE). As we know, the training time of SVM has the closest relationship with the number of training instances, thus it is not difficult to understand why oversampling often consumes more training time. Meanwhile, it is observed that SVM-SMOTE is often more time-consuming than SVM-ROS as SMOTE needs to compute lots of distances for determining the neighborhood relationship, while ROS only needs to duplicate the positive instances. In addition, it is notable that the time complexity of z-SVM is usually a little higher than that of SVM-OTHR because z-SVM needs to find the optimal *z* value by one optimization technology, while the training time of SVM-OTHR is merely dependent on the number of initially misclassified instances of the minority class. Moreover, we found that when there are a few training samples, the difference of running time of various algorithms can be ignored. With the increase of training instances, however, the gap will be gradually enlarged.

Next, we tested the performance of EnSVM-OTHR whose all parameters stay the same as that of SVM-OTHR and the number of base classifiers is empirically designated as 40. We compared the performance of EnSVM-OTHR and SVM-OTHR, then provided the percentage of performance increment on each data set in Fig. 7 which shows that on majority data sets, EnSVM-OTHR performs better than SVM-OTHR. Specifically, EnSVM-OTHR improves F-measure values on 24 data sets and G-mean values on 27 data

**Table 5**
Average Friedman rankings and APVs of various single classifiers using Holm's procedure in F-measure and G-mean, where ranking$_F$ denotes average rankings on F-measure, ranking$_G$ denotes average rankings on G-mean measure, APV$_F$ denotes adjusted *p*-value using Holm's procedure in F-measure and APV$_G$ denotes adjusted *p*-value using Holm's procedure in G-mean.

| Algorithms | ranking$_F$ | APV$_F$ | ranking$_G$ | APV$_G$ |
|---|---|---|---|---|
| SVM-OTHR | 2.517 | – | 2.100 | – |
| SSVM | 4.783 | 0.0017 | 6.050 | $2.9572 \times 10^{-9}$ |
| SVM-RUS | 6.600 | $7.5097 \times 10^{-10}$ | 3.867 | 0.0104 |
| SVM-ROS | 4.917 | $8.8682 \times 10^{-4}$ | 5.583 | $2.1821 \times 10^{-7}$ |
| SVM-SMOTE | 4.717 | 0.0020 | 4.467 | $5.4762 \times 10^{-4}$ |
| CS-SVM | 4.400 | 0.0087 | 5.400 | $9.0551 \times 10^{-7}$ |
| z-SVM | 3.800 | 0.0424 | 4.667 | $1.978 \times 10^{-4}$ |
| SVM-THR | 4.267 | 0.0113 | 3.867 | 0.0104 |

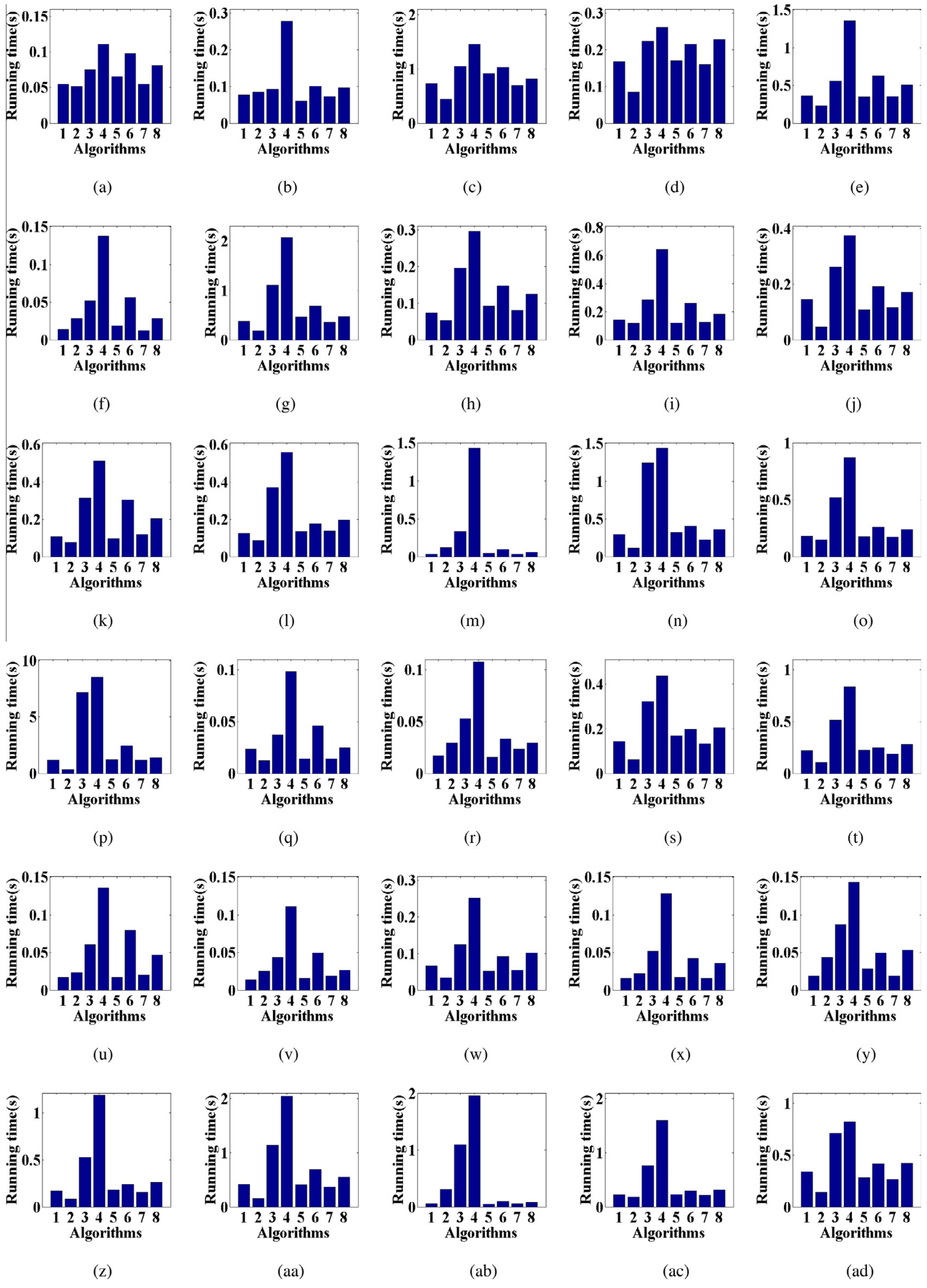
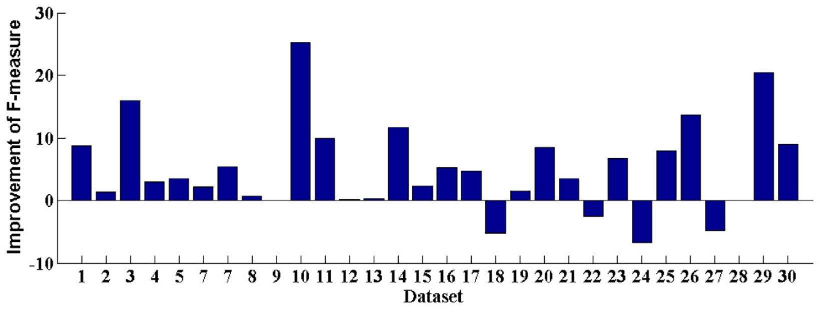


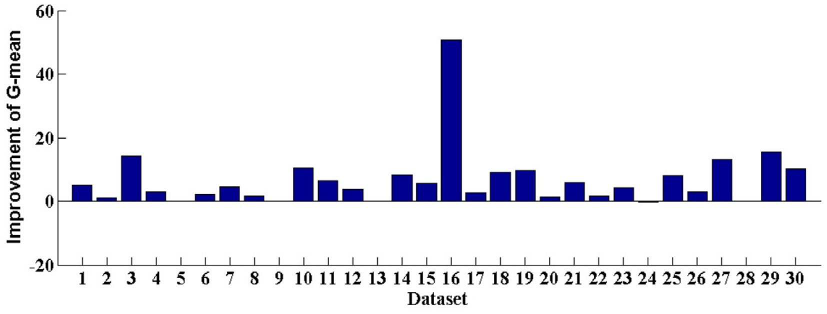
**Fig. 6.** Comparison of running time for eight single classifiers on 30 used data sets. In horizontal axis, 1–8 represent seven different algorithms, respectively, where 1: SSVM; 2: SVM-RUS; 3: SVM-ROS; 4: SVM-SMOTE; 5: CS-SVM; 6: z-SVM; 7: SVM-THR; 8: SVM-OTHR. Subgraphs (a)–(ad) denote 30 used datasets, respectively, where (a) glass1; (b) Wisconsin; (c) pima; (d) haberman; (e) vehicle1; (f) new-thyroid1; (g) yeast3; (h) ecoli3; (i) vowel0; (j) yeast-1_vs_7; (k) ecoli4; (l) abalone9-18; (m) shuttle-c2-vs-c4; (n) yeast4; (o) yeast5; (p) abalone19; (q) ecoli-0-3-4_vs_5; (r) ecoli-0-6-7_vs_3-5; (s) yeast-0-3-5-9_vs_7-8; (t) yeast-0-3-5-9_vs_7-8; (u) ecoli-0-1_vs_2-3-5; (v) ecoli-0-6-7_vs_5; (w) led7digit-0-2-4-5-6-7-8-9_vs_1; (x) ecoli-0-1_vs_5; (y) shuttle-6_vs_2-3; (z) flare-F; (aa) winequality-red-4; (ab) shuttle-2_vs_5; (ac) poker-8-9_vs_5; (ad) poker-8_vs_6.

sets. The increment of the performance are often higher than 10%. On abalone19 data set, the increment of G-mean value can be even higher than 50%. The experimental results indicate that the combination of SVM-OTHR and Bagging ensemble learning framework is feasible as it can further improve classification performance of the single classifier.

Finally, we compared the performance of EnSVM-OTHR algorithm in comparison with that of six well-known class



(a) Improvement of F-measure



(b) Improvement of G-mean

**Fig. 7.** Percentage of performance increment for EnSVM-OTHR in comparison with SVM-OTHR. In horizontal axis, 1–30 represent thirty used data sets, respectively, where 1: glass1; 2: Wisconsin; 3: pima; 4: haberman; 5: vehicle1; 6: new-thyroid1; 7: yeast3; 8: ecoli3; 9: vowel0; 10: yeast-1_vs_7; 11: ecoli4; 12: abalone9-18; 13: shuttle-c2-vs-c4; 14: yeast4; 15: yeast5; 16: abalone19; 17: ecoli-0-3-4_vs_5; 18: ecoli-0-6-7_vs_3-5; 19: yeast-0-3-5-9_vs_7-8; 20: yeast-0-3-5-9_vs_7-8; 21: ecoli-0-1_vs_2-3-5; 22: ecoli-0-6-7_vs_5; 23: led7digit-0-2-4-5-6-7-8-9_vs_1; 24: ecoli-0-1_vs_5; 25: shuttle-6_vs_2-3; 26: flare-F; 27: winequality-red-4; 28: shuttle-2_vs_5; 29: poker-8-9_vs_5; 30: poker-8_vs_6.

imbalance ensemble classifiers, including asymmetric Bagging (abbreviated as asBagging) [24], SMOTEBagging [26], RUSBoost [23], SMOTEBoost [22], EasyEnsemble (abbreviated as EasyE) [25] and BalanceCascade (abbreviated as BalanceC) [25]. To guarantee the fairness of the compared results, all ensemble classifiers share one common parameter, i.e., the number of base classifiers was designated as 40. The other parameters adopted the default best ones according to the corresponding Refs. [22–26]. For each

**Table 6**
F-measure values of seven compared ensemble classifiers on 30 Keel data sets, where bold indicates the best result on each data set.

| Dataset | asBagging | SMOTEBagging | RUSBoost | SMOTEBoost | EasyE | BalanceC | EnSVM-OTHR |
|---|---|---|---|---|---|---|---|
| glass1 | .6470 ± .0201 | .6564 ± .0296 | .6991 ± .0117 | .6732 ± .0198 | .7219 ± .0065 | **.7355 ± .0227** | .7299 ± .0166 |
| wisconsin | .9498 ± .0034 | .9420 ± .0055 | .9522 ± .0036 | .9490 ± .0068 | .9565 ± .0061 | **.9595 ± .0040** | .9575 ± .0056 |
| pima | .6168 ± .0155 | .5720 ± .0166 | .6785 ± .0148 | .6122 ± .0136 | .6600 ± .0170 | .6421 ± .0175 | **.6937 ± .0148** |
| haberman | .4266 ± .0399 | .4020 ± .0243 | .4242 ± .0358 | **.4648 ± .0285** | .4628 ± .0206 | .4259 ± .0344 | .4565 ± .0300 |
| vehicle1 | **.6749 ± .0101** | .6436 ± .0209 | .6423 ± .0144 | .6304 ± .0123 | .6381 ± .0161 | .6348 ± .0138 | .6731 ± .0171 |
| new-thyroid1 | .9222 ± .0335 | .9176 ± .0184 | .9210 ± .0157 | .9490 ± .0238 | .9063 ± .0139 | .9191 ± .0387 | **.9550 ± .0199** |
| yeast3 | .6915 ± .0092 | .7017 ± .0077 | .7278 ± .0096 | .7088 ± .0092 | .7132 ± .0100 | **.7594 ± .0119** | .7486 ± .0079 |
| ecoli3 | .5507 ± .0200 | .5989 ± .0462 | .5946 ± .0357 | .6231 ± .0419 | .5777 ± .0274 | .5978 ± .0220 | **.6386 ± .0403** |
| vowel0 | .9432 ± .0126 | **1.00 ± 0.00** | .9377 ± .0253 | **1.00 ± 0.00** | .8475 ± .0180 | .9369 ± .0217 | **1.00 ± 0.00** |
| yeast-1_vs_7 | .2218 ± .0164 | .1994 ± .0361 | **.4206 ± .0558** | .3455 ± .0373 | .2735 ± .0206 | .2661 ± .0261 | .3398 ± .0536 |
| ecoli4 | .6267 ± .0324 | .7236 ± .0811 | .6355 ± .0211 | **.7887 ± .0735** | .5717 ± .0357 | .7069 ± .0405 | .7756 ± .0594 |
| abalone9-18 | .3106 ± .0150 | **.4231 ± .0323** | .3044 ± .0359 | .3978 ± .0344 | .2540 ± .0215 | .2581 ± .0289 | .4175 ± .0546 |
| shuttle-c2-vs-c4 | .9957 ± .0021 | .9926 ± .0029 | .9928 ± .0032 | **1.00 ± 0.00** | .9909 ± .0039 | .9896 ± .0039 | .9953 ± .0023 |
| yeast4 | .2234 ± .0103 | .3166 ± .0373 | .3698 ± .0206 | .3455 ± .0324 | .2400 ± .0132 | .2686 ± .0156 | **.4059 ± .0276** |
| yeast5 | .5002 ± .0133 | .6893 ± .0408 | .5843 ± .0263 | .6678 ± .0482 | .4982 ± .0228 | .5621 ± .0260 | **.6982 ± .0355** |
| abalone19 | .0398 ± .0030 | .0388 ± .0040 | .0431 ± .0038 | .0397 ± .0042 | .0377 ± .0039 | .0198 ± .0027 | **.0479 ± .0175** |
| ecoli-0-3-4_vs_5 | .6832 ± .0538 | .8263 ± .0414 | .6564 ± .0419 | .8479 ± .0307 | .6728 ± .0506 | .7401 ± .0608 | **.8629 ± .0286** |
| ecoli-0-6-7_vs_3-5 | .5541 ± .0471 | .6413 ± .0578 | .5742 ± .0467 | .6226 ± .0522 | .5098 ± .0449 | .5992 ± .0446 | **.6734 ± .0821** |
| yeast-0-3-5-9_vs_7-8 | .3055 ± .0185 | **.3406 ± .0372** | .3001 ± .0178 | .3120 ± .0205 | .3079 ± .0221 | .3120 ± .0177 | .3226 ± .0255 |
| yeast-0-2-5-7-9_vs_3-6-8 | .6712 ± .0183 | **.7689 ± .0202** | .7029 ± .0098 | .7445 ± .0211 | .6966 ± .0176 | .7445 ± .0171 | .7688 ± .0218 |
| ecoli-0-1_vs_2-3-5 | .5657 ± .0319 | .6726 ± .0540 | .5994 ± .0518 | .6633 ± .0412 | .5791 ± .0342 | .6171 ± .0608 | **.6883 ± .0717** |
| ecoli-0-6-7_vs_5 | .5188 ± .0589 | **.7460 ± .0659** | .6880 ± .0718 | .7085 ± .0462 | .5876 ± .0249 | .7045 ± .0526 | .7395 ± .0836 |
| led7digit-0-2-4-5-6-7-8-9_vs_1 | .6576 ± .0219 | **.7472 ± .0282** | .6406 ± .0364 | .7227 ± .0367 | .5538 ± .0463 | .6030 ± .0500 | .7426 ± .0309 |
| ecoli-0-1_vs_5 | .7419 ± .0352 | .7805 ± .0413 | .7548 ± .0412 | **.7905 ± .0623** | .6530 ± .0464 | .7390 ± .0378 | .7709 ± .0598 |
| shuttle-6_vs_2-3 | .8463 ± .1626 | .7006 ± .1637 | .8764 ± .1417 | .7650 ± .0828 | .8800 ± .1095 | **.8914 ± .1018** | .8818 ± .0950 |
| flare-F | .2322 ± .0051 | .2569 ± .0225 | .2197 ± .0252 | .2458 ± .0310 | .2513 ± .0129 | .2444 ± .0155 | **.2734 ± .0447** |
| winequality-red-4 | .1286 ± .0063 | **.1939 ± .0278** | .1418 ± .0088 | .1370 ± .0210 | .1290 ± .0089 | .0992 ± .0044 | .1603 ± .0437 |
| shuttle-2_vs_5 | **1.00 ± 0.00** | **1.00 ± 0.00** | .9764 ± .0124 | .9982 ± .0156 | .6684 ± .0625 | .6787 ± .0686 | **1.00 ± 0.00** |
| poker-8-9_vs_5 | .0397 ± .0038 | .0987 ± .0513 | .0468 ± .0065 | **.1188 ± .0329** | .0419 ± .0068 | .0345 ± .0034 | .0911 ± .0258 |
| poker-8_vs_6 | .2385 ± .0182 | **.7074 ± .0934** | .2198 ± .0444 | .5961 ± .0757 | .0339 ± .0036 | .0326 ± .0038 | .4161 ± .0536 |

**Table 7**
G-mean values of seven compared ensemble classifiers on 30 Keel data sets, where bold indicates the best result on each data set.

| Dataset | asBagging | SMOTEBagging | RUSBoost | SMOTEBoost | EasyE | BalanceC | EnSVM-OTHR |
|---|---|---|---|---|---|---|---|
| glass1 | .7246 ± .0168 | .7430 ± .0239 | .7308 ± .0205 | .7576 ± .0144 | .7887 ± .0088 | **.7987 ± .0198** | .7746 ± .0113 |
| wisconsin | .9679 ± .0029 | .9604 ± .0051 | .9642 ± .0032 | **.9766 ± .0078** | .9715 ± .0046 | .9733 ± .0042 | .9751 ± .0038 |
| pima | .6992 ± .0132 | .6640 ± .0136 | .7428 ± .0122 | .6637 ± .0163 | .7354 ± .0148 | .7221 ± .0151 | **.7738 ± .0130** |
| haberman | .5882 ± .0430 | .5648 ± .0251 | **.6257 ± .0311** | .5602 ± .0366 | .6208 ± .0212 | .5849 ± .0324 | .6250 ± .0282 |
| vehicle1 | **.8052 ± .0092** | .7482 ± .0172 | .7981 ± .0125 | .7568 ± .0201 | .7799 ± .0137 | .7699 ± .0132 | .7872 ± .0140 |
| new-thyroid1 | .9835 ± .0059 | .9492 ± .0123 | **.9844 ± .0107** | .9566 ± .0123 | .9728 ± .0119 | .9731 ± .0245 | .9829 ± .0145 |
| yeast3 | .9132 ± .0048 | .8659 ± .0060 | .9078 ± .0026 | .8598 ± .0194 | .9220 ± .0064 | .9213 ± .0083 | **.9314 ± .0065** |
| ecoli3 | .8550 ± .0213 | .7903 ± .0536 | .8642 ± .0362 | .7902 ± .0618 | **.8796 ± .0158** | .8730 ± .0191 | .8647 ± .0495 |
| vowel0 | .9938 ± .0012 | **1.00 ± 0.00** | .9968 ± .0026 | **1.00 ± 0.00** | .9748 ± .0092 | .9864 ± .0045 | **1.00 ± 0.00** |
| yeast-1_vs_7 | .6947 ± .0358 | .4938 ± .0811 | .7200 ± .0544 | .5652 ± .0767 | **.7440 ± .0310** | .7353 ± .0242 | .6776 ± .0829 |
| ecoli4 | .9182 ± .0204 | .8340 ± .0764 | .9128 ± .0158 | .8626 ± .0610 | **.9309 ± .0097** | .9196 ± .0310 | .9226 ± .0164 |
| abalone9-18 | **.7828 ± .0228** | .6014 ± .0385 | .7305 ± .0413 | .6177 ± .0497 | .7316 ± .0329 | .7218 ± .0399 | .7504 ± .0625 |
| shuttle-c2-vs-c4 | .9988 ± .0021 | .9958 ± .0019 | .9952 ± .0020 | **1.00 ± 0.00** | .9924 ± .0003 | .9893 ± .0002 | .9964 ± .0021 |
| yeast4 | .8034 ± .0175 | .6814 ± .0600 | .7754 ± .0322 | .7452 ± .0597 | .8194 ± .0196 | .8196 ± .0190 | **.8223 ± .0193** |
| yeast5 | **.9636 ± .0109** | .8862 ± .0294 | .9447 ± .0202 | .9108 ± .0365 | .9494 ± .0106 | .9404 ± .0148 | .9493 ± .0244 |
| abalone19 | **.7290 ± .0357** | .3532 ± .0461 | .6858 ± .0449 | .3306 ± .0362 | .7085 ± .0729 | .4909 ± .0593 | .5260 ± .0870 |
| ecoli-0-3-4_vs_5 | .8803 ± .0543 | .8905 ± .0271 | .8742 ± .0463 | .8919 ± .0257 | .8848 ± .0647 | .8831 ± .0682 | **.8931 ± .0223** |
| ecoli-0-6-7_vs_3-5 | .8479 ± .0327 | .7476 ± .0675 | .8242 ± .0372 | .7765 ± .0629 | .7850 ± .0837 | .8337 ± .0291 | **.8604 ± .0883** |
| yeast-0-3-5-9_vs_7-8 | .6969 ± .0208 | .5696 ± .0457 | .6972 ± .0378 | .5628 ± .0567 | .6881 ± .0310 | .6933 ± .0212 | **.6988 ± .0335** |
| yeast-0-2-5-7-9_vs_3-6-8 | .8792 ± .0128 | **.9013 ± .0093** | .8858 ± .0072 | .8997 ± .0112 | .8981 ± .0095 | .8949 ± .0098 | .8963 ± .0108 |
| ecoli-0-1_vs_2-3-5 | .8432 ± .0549 | .8057 ± .0604 | .8269 ± .0743 | .7946 ± .0827 | **.8586 ± .0567** | .8351 ± .0831 | .8449 ± .0687 |
| ecoli-0-6-7_vs_5 | .8000 ± .1047 | .8607 ± .0468 | .8491 ± .0773 | .8650 ± .0656 | .8760 ± .0200 | **.8951 ± .0361** | .8713 ± .0876 |
| led7digit-0-2-4-5-6-7-8-9_vs_1 | .8381 ± .0198 | .8507 ± .0125 | .8442 ± .0260 | .8775 ± .0197 | .8630 ± .0211 | .8751 ± .0217 | **.8883 ± .0222** |
| ecoli-0-1_vs_5 | .8840 ± .0507 | **.8954 ± .0599** | .8762 ± .0622 | .8853 ± .0419 | .8898 ± .0406 | .8797 ± .0589 | .8907 ± .0553 |
| shuttle-6_vs_2-3 | .8553 ± .1618 | .7130 ± .1619 | .8742 ± .1228 | .8266 ± .0970 | .8800 ± .1095 | **.9123 ± .1037** | .8912 ± .0958 |
| flare-F | **.8258 ± .0099** | .7471 ± .0536 | .7663 ± .0657 | .8164 ± .0272 | .8235 ± .0208 | .7933 ± .0248 | .6583 ± .0863 |
| winequality-red-4 | .5852 ± .0249 | **.6670 ± .0597** | .5961 ± .0348 | .6482 ± .0527 | .5869 ± .0315 | .5197 ± .0153 | .6208 ± .0475 |
| shuttle-2_vs_5 | **1.00 ± 0.00** | **1.00 ± 0.00** | .9894 ± .0047 | .9972 ± .0008 | .9920 ± .0011 | .9925 ± .0017 | **1.00 ± 0.00** |
| poker-8-9_vs_5 | .3238 ± .0594 | **.6109 ± .1049** | .3488 ± .0614 | .5622 ± .0979 | .3786 ± .0665 | .3927 ± .0734 | .5797 ± .1142 |
| poker-8_vs_6 | .5151 ± .0724 | **.7335 ± .0926** | .4347 ± .0761 | .6720 ± .0819 | .3194 ± .0881 | .3527 ± .0605 | .6959 ± .0863 |

algorithm, fivefold cross validation is randomly repeated for 10 times, and then the experimental results are provided in the form of mean ± standard deviation. The F-measure and G-mean values of these seven ensemble classification algorithms are described in Tables 6 and 7, respectively. Table 8 shows the corresponding statistical results.

Tables 6 and 7 show that EnSVM-OTHR performs best on 12 data sets referring to F-measure criterion and obtains the highest values on 9 data sets according to G-mean evaluation measure. As for six other ensemble learning algorithms, they present quite different classification performance on different types' data sets. Specifically speaking, two SMOTE-based ensemble learning algorithms often outperform the others on those highly imbalanced data sets, while four undersampling-based algorithms usually performs better on those data sets with low imbalance ratio. From the results of these two tables, we can deduce a conclusion that for highly imbalanced classification data, SMOTEBagging can be considered as an excellent candidate irrespective of the acquirement of running time.

Table 8 shows the average rankings and APVs computed for all seven ensemble learning algorithms following the same schema of Table 5. We can observe that the proposed EnSVM-OTHR algorithm has obtained the lowest ranking$_F$ and ranking$_G$ values, indicating it is the best one in these seven algorithms. In this case, all APVs except APV$_G$ of EasyE algorithm, are lower than a standard used level of significance of $\alpha = 0.05$, indicating EnSVM-OTHR obviously outperforms 5 other algorithms in statistics. Meanwhile, it indicates that there is no obvious difference between EnSVM-OTHR and EasyE on G-mean evaluation measure.

**Table 8**
Average Friedman rankings and APVs of various ensemble classifiers using Holm's procedure in F-measure and G-mean, where ranking$_F$ denotes average rankings on F-measure, ranking$_G$ denotes average rankings on G-mean measure, APV$_F$ denotes adjusted *p*-value using Holm's procedure in F-measure and APV$_G$ denotes adjusted *p*-value using Holm's procedure in G-mean.

| Algorithms | ranking$_F$ | APV$_F$ | ranking$_G$ | APV$_G$ |
|---|---|---|---|---|
| EnSVM-OTHR | 1.967 | – | 2.433 | – |
| asBagging | 5.100 | $9.6828 \times 10^{-8}$ | 4.000 | 0.0149 |
| SMOTEBagging | 3.333 | 0.0286 | 4.967 | $3.3455 \times 10^{-5}$ |
| RUSBoost | 4.333 | $6.6147 \times 10^{-5}$ | 4.467 | 0.0011 |
| SMOTEBoost | 3.183 | 0.0292 | 4.667 | $3.1138 \times 10^{-4}$ |
| EasyE | 5.400 | $4.4925 \times 10^{-9}$ | 3.467 | 0.0639 |
| BalanceC | 4.683 | $4.4514 \times 10^{-6}$ | 4.000 | 0.0149 |

## 6. Summary

In this article, we firstly analyzed the reason why the classification performance of SVM can be damaged by class imbalance in theory. To our best knowledge, it is the first time to theoretically study the influence of class imbalance towards SVM. We also indicated the drawbacks of the existing decision threshold adjustment algorithm SVM-THR, transformed it as an optimization problem, as well as designed an exhaustive search algorithm named SVM-OTHR to automatically find the optimal position of decision hyperplane. Furthermore, we indicated that SVM-OTHR would often lead to the loss of generalization ability, then inserted a small random perturbation term into each SVM-OTHR, finally integrated multiple SVM-OTHRs into Bagging ensemble learning framework to present a improved version of algorithm called EnSVM-OTHR. By a mass of experiments, we found that both proposed algorithms could not only acquire more balanced classification results than SVM-THR algorithm without obviously increased time costs, but also outperform many state-of-the-art class imbalance learning algorithms.

In future work, we wish to further optimize the proposed algorithms to lower their time and space complexity. Additionally, the possibility of combining SVM-OTHR and Boosting ensemble learning framework will be investigated, too.

## Acknowledgments

The work was supported in part by National Natural Science Foundation of China under Grant No. 61305058, No. 61473086, No. 61375001, No. 61100116 and No. 61005008, Natural Science

Foundation of Jiangsu Province of China under Grant No. BK20130471 and No. BK2011492, China Postdoctoral Science Foundation under Grant No. 2013M540404, Jiangsu Planned Projects for Postdoctoral Research Funds under Grant No. 1401037B, Nature Science Foundation of the Jiangsu Higher Education Institutes of China under Grant No. 12KJB520003, open fund of Key Laboratory of Measurement and Control of Complex Systems of Engineering, Ministry of Education under Grant No. MCCSE2013B01, and open fund of the Jiangsu Key Laboratory of Image and Video Understanding for Social Safety (Nanjing University of Science and Technology) under Grant No. 30920130122006.

## References

[1] N. Japkowicz, Workshop on learning from imbalanced data sets, in: Proceedings of the 17th American Association for Artificial Intelligence, Austin, Texas, USA, 2000.
[2] N.V. Chawla, N. Japkowicz, A. Kolcz, Workshop on learning from imbalanced data sets II, in: Proceedings of the 20th International Conference of Machine Learning, Washington, USA, 2003.
[3] N.V. Chawla, N. Japkowicz, A. Kolcz, Editorial: special issue on learning from imbalanced data sets, ACM SIGKDD Explor. Newslett. 6 (2004) 1–6.
[4] N. Chawla, N. Japkowicz, Z.H. Zhou, Workshop on data mining when classes are imbalanced and errors have costs, in: Proceedings of the 13th Pacific-Asia Knowledge Discovery and Data Mining Conference, Bangkok, Thailand, 2009.
[5] W. Wei, J. Li, L. Cao, Y. Ou, J. Chen, Effective detection of sophisticated online banking fraud on extremely imbalanced data, World Wide Web 16 (2013) 449–475.
[6] C. Thomas, Improving intrusion detection for imbalanced network traffic, Secur. Commun. Netw. 6 (2013) 309–324.
[7] Y. Tang, S. Krasser, P. Judge, Fast and effective spam sender detection with granular SVM on highly imbalanced mail server behavior data, in: Proceedings of the 2nd IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing, Atlanta, Georgia, USA, 2006, pp. 1–6.
[8] C.C. Loy, T. Xiang, S. Gong, Stream-based active unusual event detection, in: Proceedings of the 10th Asian Conference on Computer Vision, Queenstown, New Zealand, 2010, pp. 161–175.
[9] M. Khalilia, S. Chakraborty, M. Popescu, Predicting disease risks from highly imbalanced data using random forest, BMC Med. Inform. Decis. 11 (2011) 51.
[10] N.G. Pedrajas, J.P. Rodriguez, M.G. Pedrajas, D. Ortiz-Boyer, C. Fyfe, Class imbalance methods for translation initiation site recognition in DNA sequences, Knowl.-Based Syst. 25 (2012) 22–34.
[11] C. Ling, C. Li, Data mining for direct marketing problems and solutions, in: Proceedings of the 4th ACM SIGKDD International Conference of Knowledge Discovery and Data Mining, New York, USA, 1998, pp. 73–79.
[12] N.V. Chawla, K.W. Bowyer, L.O. Hall, SMOTE: synthetic minority over-sampling technique, J. Artif. Intell. Res. 16 (2002) 321–357.
[13] R. Akbani, S. Kwek, N. Japkowicz, Applying support vector machine to imbalanced datasets, in: Proceedings of the 15th European Conference on Machine Learning, Pisa, Italy, 2004, pp. 39–50.
[14] S.J. Yen, Y.S. Lee, Cluster-based under-sampling approaches for imbalanced data distributions, Expert Syst. Appl. 36 (2009) 5718–5727.
[15] H.L. Yu, J. Ni, J. Zhao, ACOSampling: an ant colony optimization-based undersampling method for classifying imbalanced DNA microarray data, Neurocomputing 101 (2013) 309–318.
[16] C. Seiffert, T.M. Khoshgoftaar, J.V. Hulse, Hybrid sampling for imbalanced data, Integr. Comput.-Aided Eng. 16 (2009) 193–210.
[17] E. Osuna, R. Freund, F. Girosit, Training support vector machines: an application to face detection, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Juan, Puerto Rico, 1997, pp. 130–136.
[18] B. Zadrozny, J. Langford, N. Abe, Cost-sensitive learning by cost-proportionate example weighting, in: Proceedings of the 3rd International Conference of Data Mining, Melbourne, Florida, USA, 2003, pp. 435–442.
[19] T. Imam, K.M. Ting, J. Kamruzzaman, z-SVM: an SVM for improved classification of imbalanced data, in: Proceedings of the 19th Australian Joint Conference on Artificial Intelligence, Hobart, Australia, 2006, pp. 264–273.
[20] Z.H. Zhou, X.Y. Liu, Training cost-sensitive neural networks with methods addressing the class imbalance problem, IEEE Trans. Knowl. Data Eng. 18 (2006) 63–77.
[21] W.J. Lin, J.J. Chen, Class-imbalanced classifiers for high-dimensional data, Brief. Bioinform. 14 (2013) 13–26.
[22] N.V. Chawla, A. Lazarevic, L.O. Hall, K.W. Bowyer, SMOTEBoost: improving prediction of the minority class in boosting, in: Proceedings of 7th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'03), 2003, pp. 107–119.
[23] C. Seiffert, T.M. Khoshgoftaar, J. Van Hulse, A. Napolitano, RUSBoost: a hybrid approach to alleviating class imbalance, IEEE Trans. Syst., Man, Cybern. A 40 (1) (2010) 185–197.
[24] D. Tao, X. Tang, X. Li, Asymmetric Bagging and random subspace for support vector machines-based relevance feedback in image retrieval, IEEE Trans. Pattern Anal. Mach. Intell. 28 (2006) 1088–1099.
[25] X.Y. Liu, J. Wu, Z.H. Zhou, Exploratory undersampling for class-imbalance learning, IEEE Trans. Syst., Man, Cybern. Part B 39 (2009) 539–550.
[26] S. Wang, X. Yao, Diversity analysis on imbalanced data sets by using ensemble models, in: Proceedings of the 2009 IEEE Symposium on Computational Intelligence and Data Mining (CIDM'09), 2009, pp. 324–331.
[27] T.M. Khoshgoftaar, J.V. Hulse, A. Napolitano, Comparing boosting and Bagging techniques with noisy and imbalanced data, IEEE Trans. Syst., Man, Cybern. Part B 41 (2011) 552–568.
[28] H.L. Yu, J. Ni, An Improved ensemble learning method for classifying high-dimensional and imbalanced biomedicine data, IEEE/ACM Trans. Comput. Biol. Bioinform. 11 (2014) 657–666.
[29] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, F. Herrera, A review on ensembles for class imbalance problem: Bagging, boosting and hybrid based approaches, IEEE Trans. Syst., Man, Cybern. – Part C: Appl. Rev. 42 (4) (2012) 463–484.
[30] V. Vapnik, Statistical Learning Theory, Wiley Publishers, New York, USA, 1998.
[31] A. Anand, G. Pugalenthi, G.B. Fogel, P.N. Suganthan, An approach for classification of highly imbalanced data using weighting and undersampling, Amino Acids 39 (2010) 1385–1391.
[32] J. Demsar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.
[33] S. Garcia, A. Fernandez, J. Luengo, F. Herrera, Advanced non-parametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power, Inf. Sci. 180 (2010) 2044–2064.
[34] M.J. Abdi, S.M. Hosseini, M. Rezghi, A novel weighted support vector machine based on particle swarm optimization for gene selection and tumor classification, Comput. Math. Methods Med. (2012), http://dx.doi.org/10.1155/2012/320698.
[35] N. Japkowicz, S. Stephen, The class imbalance problem: a systematic study, Intell. Data Anal. 6 (2002) 429–450.
[36] A. Krogh, J. Vedelsby, Neural network ensembles, cross validation, and active learning, Adv. Neural Inform. Process. Syst. 7 (1995) 231–238.
[37] T. Ho, The random subspace method for constructing decision forests, IEEE Trans. Pattern Anal. Mach. Intell. 20 (1998) 832–844.
[38] L. Breiman, Bagging predictors, Mach. Learn. 24 (1996) 123–140.
[39] J. Alcala-Fdez, A. Fernandez, J. Luengo, J. Derrac, S. Garcia, L. Sanchez, F. Herrera, KEEL data-mining software tool: data set repository, integration of algorithms and experimental analysis framework, J. Multiple-Valued Logic Soft Comput. 17 (2–3) (2011) 255–287.
[40] C. Wang, J. Lu, G. Zhang, Integration of ontology data through learning instance matching, in: Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence (WI'06), 2006, pp. 536–539.
[41] S. Garcia, J. Derrac, I. Triguero, C.J. Carmona, F. Herrera, Evolutionary-based selection of generalized instances for imbalanced classification, Knowl.-Based Syst. 25 (2012) 3–12.
[42] C.C. Chang, C.J. Lin, LIBSVM: a library for support vector machines, ACM Trans. Intell. Syst. Technol. 2 (2011) 27.
[43] V. Garcia, J.S. Sanchez, R.A. Mollineda, On the effectiveness of preprocessing methods when dealing with different levels of class imbalance, Knowl.-Based Syst. 25 (2012) 13–21.
[44] V. Lopez, A. Fernandez, S. Garcia, V. Palade, F. Herrera, An insight into classification with imbalanced data: empirical results and current trends on using data intrinsic characteristics, Inform. Sci. 250 (2013) 113–141.
[45] H.L. Yu, J. Ni, S. Xu, B. Qin, H.R. Jv, Estimating harmfulness of class imbalance by scatter matrix based class separability measure, Intell. Data Anal. 18 (2014) 203–216.