Pattern Recognition Letters 125 (2019) 325–332

Contents lists available at ScienceDirect

Pattern Recognition Letters

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

Kernel modified optimal margin distribution machine for imbalanced data classification[☆]

Xiaogang Zhang[a], Dingxiang Wang[a], Yicong Zhou[b], Hua Chen[c,*], Fanyong Cheng[d], Min Liu[a]

[a] College of Electrical and Information Engineering, Hunan University, Changsha 410082, China
[b] Department of Computer and Information Science, University of Macau, Macau 999078, China
[c] College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China
[d] Fujian Provincial Key Laboratory of Information Processing and Intelligent Control, Minjiang University, Fuzhou 350108, China

ARTICLE INFO

Article history:
Received 9 November 2018
Available online 8 May 2019

Keywords:
Margin distribution
Imbalanced data classification
Kernel modification
Balanced detection rate
Generalization performance

ABSTRACT

Although the optimal margin distribution machine (ODM) has better generalization performance in pattern recognition than traditional classifiers, ODM as well as traditional classifiers often suffers from data imbalance. To address this, this paper proposes a kernel modified ODM (KMODM) to eliminate the side effect of imbalanced data. According to the mechanism of ODM, a novel conformal function is designed to scale the kernel matrix of ODM, this can increases the separability of the training data in the feature space. In addition, to eliminate the skew of the separator toward minority class, KMODM introduces two free parameters in conformal function to balance the influence of different training data on separating hyperplane. Experimental results on two-dimensional visualization data show that KMODM can alleviate the skew of the separating hyperplane caused by imbalanced data. For most of ten UCI data sets, KMODM can broad the margin of the minority class and achieve the highest average G-mean and F1 score. This means that KMODM has more balanced detection rate and better generalization performance compared to other baseline methods, especially in presence of heavily imbalanced training data.

1. Introduction

In data mining and pattern recognition tasks, data imbalance is a common problem that the examples of a certain class are outnumbered by another class or all other classes [1–3]. In this case, if adopting the conventional classification algorithms such as the support vector machine (SVM), the detection rate of a majority class is much higher than that of a minority class. The conventional classifiers aim at improving the overall classification accuracy even in presence of imbalanced data. The majority classes are treated equally with the minority class. This leads to the separating hyperplane skewed toward minority classes [4], and results in a decrease of the detection accuracy of the minority class and deterioration of generalization performance of classifiers [5]. Therefore, one of the key issues is how to balance the detection accuracy of each class and obtain better generalization performance of classifier simultaneously in imbalanced data classification. At present, the commonly used methods to balance the detection rate can be simply divided into four categories: kernel modification, data resampling, cost sensitive and post-processing methods. A brief introduction of these methods is given below.

To obtain a balanced training data, the resampling methods either oversample the minority classes [6-7], or undersample the majority classes [8], or synthesize them both. The difference among these methods is that the strategies used in the resampling process. For example, to improve the synthetic minority oversampling technique (SMOTE) [6], a weighted kernel-based SMOTE (WKSMOTE) was proposed in [7] to increase the number of minority examples by oversampling the minority examples in kernel space. In [8], clustering technique was introduced to balance the number of training data in the undersampling process. However, the data resampling process is unavoidable to introduce noise (oversampling) to or eliminate some useful information (undersampling) from the training data. It thus leads to the performance degradation of the classifier. Instead of balancing the number of training data directly, the cost-sensitive methods balance the influence of imbalanced data by assigning different misclassification penalty factors to the training data with different labels [9–12]. Base on cost-sensitive strategy, a lot of classifiers have been proposed including Cost-Sensitive SVM (CSSVM) [9-10], Cost-Sensitive Extreme Learning Machine (CSELM) [11] and Large Cost-Sensitive

[☆] We confirm that the manuscript has been read and approved by all named authors and that there are no other persons who satisfied the criteria for authorship but are not listed. We further confirm that the order of authors listed in the manuscript has been approved by all of us.
[*] Corresponding author.
E-mail address: anneychen@126.com (H. Chen).

https://doi.org/10.1016/j.patrec.2019.05.005
0167-8655/© 2019 Elsevier B.V. All rights reserved.

Margin Distribution Machine (LCSDM) [5], which have achieved great successes. In particular, the LCSDM proposed in [5] combines cost-sensitive method with the latest LDM classifier, which can obtain a more balanced detection rate by increasing the margin distribution of the minority class. The cost sensitive based methods are aimed at minimize the misclassification costs without changing the spatial distribution of the training data, it determines the influence weight of training data by adjusting the Lagrange multiplier α of the classifier. In fact, the effect of this method is limited because the penalty factor is used only as the upper bound of α according to the KKT conditions [13]. This increase of the misclassification penalty does not necessarily affect α. Recently, a post-processing method [14] is proposed to eliminate the skewness of separator toward minority class by adjusting the trained bias according to the imbalance rate of training data. Since this method neither introduces new parameters nor changes the optimization process of the original classifier, it is relatively easy to implement. However, once the separator is determined, this method controls the separator position instead of optimizing separator shape.

Unlike the above-mentioned techniques, the kernel modification methods change the distribution of training data in the feature space to eliminate the skewness of the separator by modifying the kernel function or the kernel matrix of the classifier. The conformal transformation of the kernel function is one of the most common kernel modification methods [15-16], because it can increase the space resolution in the area of the class boundary and improve separability of training data in the feature space. In recent years, various revisions of conformal functions were proposed to improve the kernel modification method for imbalanced data classification [4,13,17-18]. These conformal functions can enhance the mapping ability of kernel function. For example, a simple form of conformal function which only considers the margin and imbalance rate of training data is proposed in [17]. Consequently, it has excellent performance for imbalanced data classification when combined with SVM. The kernel modification method avoids the operations that may decrease the overall performance and computation efficiency of classifiers. These operations include resampling training data and assigning misclassification penalty factors. Thus, it is more suitable for improving classifiers for imbalanced data.

However, since the conformal function of kernel modification method generally designed for conventional classifiers like SVM. From the margin theory [19-20], only learn the minimum margin without optimizing the margin distribution always leads to the poor generalization performance of SVM. Thus, the optimal margin distribution machine (ODM) [21–23] is proposed to optimize the margin distribution. It was proved that ODM has better generalization performance in the balanced data classification than conventional classifiers. However, ODM still fails to deal with imbalanced data and the conformal function of kernel scaling method designed for SVM is not suitable for ODM.

To improve the performance of ODM on imbalanced data classification, this paper proposes a novel ODM algorithm based on kernel modification (KMODM). We design a new conformal function to improve the kernel modification methods for ODM. Tuning two free parameters in the new conformal function, KMODM is able to increase the space resolution of class boundary and the influence of the minority class on final separator. For simplicity, we discuss only the binary classifier for imbalanced data in this paper. This paper is organized as follows. The related works are reviewed in Section 2, including a brief introduction to the ODM and kernel scaling method. The proposed conformal function and KMODM are described in Section 3. The experiment results of KMODM and comparison with other methods are presented in Section 4. In Section 5, conclusions are drawn and some suggestions of the future work are given.

2. Related works

The ODM classifier and kernel scaling method will be briefly reviewed in this section.

2.1. Optimal margin distribution machine (ODM)

For the nonlinear separable case, the conventional classifiers like SVM, introduce a kernel function $K(x_i, x_j) = \varphi(x_i)\varphi(x_j)$ to map the training data in the input space to a high-dimensional feature space for better data separability, where φ is a mapping function. The core of conventional classifiers (such as SVM) is to maximize the minimum margin. For a given classifier $y = \omega^T\varphi(x)$, where ω is a linear classifier, the margin between example $(x_i,\ y_i)$ and the separator is defined as:

$$f(x_i) = \gamma_i = y_i\omega^T\varphi(x_i), \forall i = 1, 2, \ldots, m, \tag{1}$$

where, m is the number of training examples. The object function of SVM is given as:

$$\begin{aligned} &\min_{\omega,\xi} \frac{1}{2}\omega^T\omega + C\sum_{i=1}^{m}\xi_i \\ &s.t.\quad y_i\omega^T\varphi(x_i) \geq 1 - \xi_i \\ &\qquad\ \ \xi_i \geq 0, i = 1, 2, \ldots, m \end{aligned} \tag{2}$$

Where ξ is the slack variable denotes the loss of examples, C is misclassification penalty factor. Since the SVM optimizes only the minimum margin and ignores the margin distribution of training data, it may have poor generalization performance and cannot obtain the optimal separating hyperplane. According to the margin theory in [20], optimizing the margin distribution is more important than optimizing the minimum margin for improving the performance of a classifier. According to the margin definition in Eq. (1), the margin mean $\bar{\gamma}$ and margin variance $\hat{\gamma}$ can be formulated as:

$$\begin{aligned} \bar{\gamma} &= \sum_{i=1}^{m} y_i\omega^T\varphi(x_i) = \frac{1}{m}(Xy)^T\omega \\ \hat{\gamma} &= \frac{1}{m}\sum_{i=1}^{m}\left(y_i\omega^T\varphi(x_i) - \bar{\gamma}\right)^2 \\ &= \frac{1}{m}\omega^T XX^T\omega - \frac{1}{m^2}\omega^T Xyy^T X^T\omega \end{aligned} \tag{3}$$

where $X = [\varphi(x_1), \ldots, \varphi(x_m)], y = [y_1, \ldots, y_m]^T$. To reach better generalization performance, optimal margin distribution machine (ODM) [22] minimizes the margin variance while maximizing the margin mean to obtain the optimal margin distribution. Its objective function is formulated as:

$$\begin{aligned} &\min_{\omega,\xi_i,\varepsilon_i} \frac{1}{2}\omega^T\omega + \frac{1}{m}\sum_{i=1}^{m}\left(C_1\xi_i^2 + C_2\varepsilon_i^2\right) \\ &s.t.\quad y_i\omega^T\phi(x_i) \geq 1 - S - \xi_i \\ &\qquad\ \ y_i\omega^T\phi(x_i) \leq 1 + S + \varepsilon_i \\ &\qquad\ \ \xi_i \geq 0, \varepsilon_i \geq 0, i = 1, 2, \ldots, m \end{aligned} \tag{4}$$

Where ξ_i, ε_i represent the deviation between the margin and margin mean, S is the sparse parameter determining the number of support vectors, C_1 and C_2 are the parameters for trading-off the margin variance of the support vectors located in different areas. The margin mean of ODM is set to 1 by scaling $\|\omega\|$ in Eq. (4). Introducing sparse parameter S and optimizing margin distribution allows ODM to be better generalization performance than SVM. However, ODM still fails to consider the influence of imbalanced training data. This paper aims to address this defeat.

2.2. Kernel scaling

The kernel function of classifier can also be written as $K(x_i, x_j) = \varphi(x_i)\varphi(x_j)$. Using the mapping function φ, the training data in the input space I are embedded into a curved Riemannian manifold S in a high dimensional feature space F [4]. The Riemannian metric of S is defined as:

$$g_{ij}(x) = \left(\frac{\partial^2 K(x, x')}{\partial x_i \partial x'_j} \right)_{x=x'}, \tag{5}$$

$g_{ij}(x)$ represents the local volume expansion coefficient of the adjacent area of example (x, y) in the feature space. It can be seen in Eq. (5) that the expansion coefficient $g_{ij}(x)$ is closely related to the kernel function $K(x, x')$.$g_{ij}(x)$ can be changed indirectly by adjusting $K(x, x')$. A new kernel $\tilde{K}(x, x')$ can be obtained using the conformal transformation and the original kernel function. In order to increase the separability of train data in kernel space, it is expected to assign a larger $g_{ij}(x)$ to the boundary examples to magnify the spatial resolution of the class boundary [4]. The conformal transformation of kernel function can be written as [4]:

$$\tilde{K}(x, x') = D(x)D(x')K(x, x') \tag{6}$$

The conformal function $D(x)$ reaches its maximum near the separating hyperplane. In the case of using RBF as the kernel function, [4] proposed a conformal function to adjust the kernel of SVM according to the distance between the example and support vectors:

$$D(x) = \sum_{k \in SV} \exp\left(-\frac{|x - x_k|}{\eta \tau_k^2} \right) \tag{7}$$

where SV represents the support vector set, η is the parameter which reflects the imbalance rate of training data. Parameter τ_k^2 reflects the relative position of support vectors in the feature space. It can be calculated by:

$$\tau_k^2 = \underset{i \in \{\|\varphi(x_i) - \varphi(x_k)\|^2 < M, y_i \neq y_k\}}{AVG} \left(\|\varphi(x_i) - \varphi(x_k)\|^2 \right) \tag{8}$$

In which M is the average distance between support vectors and $\varphi(x_k)$. We can see that the conformal function in Eq. (7) is complicated and sensitive to the distribution of support vectors. Therefore, a simple form of the conformal function was proposed in [18]:

$$D(x) = e^{-kf(x)^2} \tag{9}$$

Where $f(x)$ is the margin of example (x, y) defined in Eq. (1), k is a positive constant which controls the value of $D(x)$. Since the value of$D(x)$ decreases with the increase of the margin $f(x)$, the conformal function in Eq. (9) can amplify $g_{ij}(x)$ in the area near to the separator while reducing it in the area far away from the separator, thus can magnify the spatial resolution of class boundary.

3. Kernel modification of ODM

This section proposes a novel classifier called kernel modified ODM (KMODM) to inherit the excellent generalization performance of ODM while effectively handling imbalanced data.

3.1. Construction of the conformal function

The basic idea of the kernel scaling method is to adjust the kernel function according to the margin between the training data and initial separator in the feature space. It ensures that the data close to the initial separator have a larger $g_{ij}(x)$, increasing the influence of such data to the final classification result. In the implementation of kernel scaling method, conformal function is the key factor to determine whether the algorithm is effective or not.

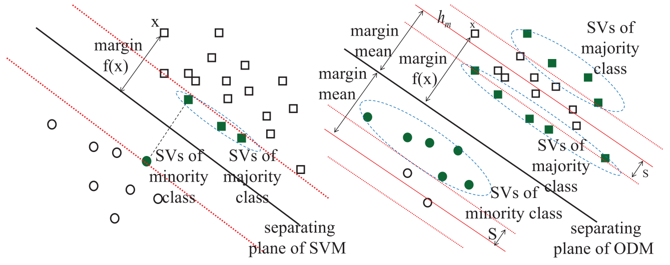

Fig. 1. Illustration of support vectors of (a) SVM. (b) ODM. green examples represent support vectors, h_m represents the margin mean hyperplane.

The conformal function in Eq. (7) and Eq. (9) were proposed for modifying the kernel function of SVM. In these functions, the distance between the example and initial separating hyperplane or support vectors is considered since the separator learned by SVM is determined by the examples with the smallest margin (support vectors, SVs). Unlike SVM, in order to optimize the margin variance, ODM treats the data whose deviation from margin mean is larger than S as a support vector to determine the separating hyperplane, as shown in Fig. 1. Therefore, the conformal functions in Eq. (7) and Eq. (9) are not suitable to modify the kernel function of ODM.

From Fig. 1(b), we can see that the margin mean hyperplane is used as a baseline to define support vectors of ODM. Since the space of ODM's support vectors is not continuous, it can be divided into two subspaces $f(x) \le (1 - S)m$ and $f(x) \ge (1 + S)m$ according to the margin $f(x)$. The purpose of kernel modification for ODM is to gather the training data to h_m. In other words, $D(x)$ should ensure that the examples in $f(x) \le (1 - S)m$ have a larger expansion coefficient and that the examples located in $f(x) \ge (1 + S)m$ have a smaller one. Thus, in this paper, we construct a piecewise conformal function as follows:

$$D(x) = \begin{cases} e^{-K_n/(m - f(x))} & , f(x) \le (1 - S)m \\ e^{-K_f(f(x) - m)} & , f(x) \ge (1 + S)m \\ e^{-K_f \cdot m} & , else \end{cases} \tag{10}$$

where S is the sparse parameter of KMODM which is the same as that of ODM in Eq. (4), $f(x)$ and m are the margin and margin mean of training examples obtained by ODM respectively, K_n and K_f are parameters to control the volume expansion coefficients of different areas in feature space. When the training data x is located in the area near to the separating hyperplane $(f(x) \le (1 - S)m)$, the value of $D(x)$gradually reduces from 1 to $\exp(-K_n/(1 - S)m)$ with the margin of x approaching $(1 - S)m$. However, when x is located in the area far away from the separating hyperplane $(f(x) \ge (1 + S)m)$, $D(x)$ gradually reduces from $\exp(-K_f(1 + S)m)$ to 0 with the margin of x approaching ∞. Because the separator obtained by ODM skews toward the minority class, the margin of most minority examples are smaller than the margin mean. That is, minority examples are generally located in $f(x) \le (1 - S)m$. In order to obtain a large expansion coefficient in the area in which minority examples are located, K_f should be greater thanK_n. In this paper, grid search and cross validation are used to search the optimal values of these two parameters.

Generally speaking, The advantages of using $D(x)$ constructed in Eq. (10) to adjust the ODM kernel function are as follows:

(i) $D(x)$ amplifies the spatial resolution of the area near to the class boundary while improving the separability of the training data in feature space. The examples with a small margin are generally located at the class boundary. $D(x)$ can increase the expansion coefficient of the area around these examples. That is, the spatial resolution of the class boundary can be increased indirectly. The data separability can be improved effectively.

(ii) There is no need to consider the problem of data imbalance. The separator learned by ODM skews toward the minority class, resulting in that the margin of minority examples is smaller than the margin mean. Optimizing parameters K_n and K_f in $D(x)$, the expansion coefficients of minority examples will be set to large values automatically, increasing the influence of minority examples on the final separator.

3.2. KMODM

According to Eqs. (6) and (10), a new kernel function can be obtained. The original kernel matrix is denoted as $\mathbf{K} = (k_{ij})$, the new kernel matrix obtained by the conformal transformation is:

$$\tilde{\mathbf{K}} = (\tilde{k}_{ij}) = D(x_i) \times D(x_j) \times k_{ij} \tag{11}$$

According to the Mercer theorem, to prove that $\tilde{\mathbf{K}}$ is a valid kernel function, we simply prove that the kernel matrix $\tilde{\mathbf{K}}$ is positive (semi) definite. The rationality of our work can be verified by the following corollary.

Corollary 1. When given a valid kernel matrix $\mathbf{K}$, the new kernel matrix $\tilde{\mathbf{K}}$ defined in Eqs. (10) and (11) is positive (semi) definite and a valid kernel matrix.

Proof. From Eq. (10), since the range of conformal function is $D(x) \geq 0$, we have $D(x_i)D(x_j) = D(x_j)D(x_i)$. It is a symmetric function. For training data $x_1, x_2, \ldots, x_n \in X$ and $\alpha_1, \alpha_2, \ldots, \alpha_n \in R$, we have:

$$\begin{aligned}\sum_{i,j=1}^{n} \alpha_i \alpha_j D(x_i)D(x_j) &= \sum_{i=1}^{n} \alpha_i D(x_i) \sum_{j=1}^{n} \alpha_j D(x_j) \\ &= \left(\sum_{i=1}^{n} \alpha_i D(x_i) \right)^2 \geq 0\end{aligned}$$

Thus $D(x_i)D(x_j)$ is also a positive (semi) definite function. Denoting $\mathbf{d} = (d_1, d_2, \ldots, d_n)^T$ as an n-dimensional vector with $d_i = D(x_i)$, $i = 1, 2, \ldots, n$. According to the training data set and positive (semi) definite function $D(x_i)D(x_j)$, a matrix $\mathbf{dd}^T$ is also a positive (semi) definite matrix. The new kernel matrix $\tilde{\mathbf{K}}$ in Eq. (11) can be rewritten as the Hadamard product of $\mathbf{dd}^T$ and $\mathbf{K}$:

$$\tilde{\mathbf{K}} = \mathbf{dd}^T * \mathbf{K}.$$

$\mathbf{K}$ is a valid kernel matrix, which means a positive (semi) definite matrix. Then according to Schur product theorem, $\tilde{\mathbf{K}}$ is a positive (semi) definite matrix, and is thus a valid kernel matrix.

The details of the KMODM algorithm are given in Fig. 2. We first apply ODM to the training data to obtain their margin and the margin mean. The corresponding *D(x)* of each training example is calculated by Eq. (10) to adjust the original kernel matrix $\mathbf{K}$. The new kernel matrix $\tilde{\mathbf{K}}$ is then obtained. $\tilde{\mathbf{K}}$ is used as the new kernel matrix of ODM to learn the final separator, resulting in the KMODM classifier.

```
Input :
$X_{train}$, $\mathbf{K}$;
Output :
$OC$; /* output classifier */
Variables :
$S$; /* sparse parameter of ODM */
$x$; /* a train example */
$m$; /* margin mean of $X_{train}$ */
$K_n$, $K_f$; /* trade-off parameter of $D(x)$ */
Function :
ODMTrain($X_{train}$, $\mathbf{K}$); /* train classifier $C$ */
Extractf ($X$, $C$); /* extract margin of $X$ under $C$ */
ComputeM($f(x)$); /* compute margin mean */
ComputeD($f(x)$, $S$, $m$, $K_n$, $K_f$); /* compute $D(x)$ of example $x$ */
Begin
1) $C \leftarrow$ ODMTrain($X_{train}$, $\mathbf{K}$);
2) $f(x) \leftarrow$ Extractf ($X_{train}$, $C$);
3) $m \leftarrow$ ComputeM($f(x)$);
4)    for each $x \in X_{train}$ {
5)        $D(x) \leftarrow$ ComputeD($f(x)$, $S$, $m$, $K_n$, $K_f$)}
6)    for each $k_{ij}$ in $\mathbf{K}${
7)        $\tilde{k}_{ij} = D(x_i) \times D(x_j) \times k_{ij}$;}
8) $OC \leftarrow$ ODMTrain($X_{train}$, $\tilde{\mathbf{K}}$);
9) return $OC$
End
```

Fig. 2. The KMODM algorithm.

4. Experiments

The experiments mainly focus on two parts: (1) classifier visualization on two-dimensional data; (2) performance comparisons with baseline algorithms on UCI data sets. It is worth noting that, for simplicity, all kernel functions in this section adopt the RBF function. In fact, when using other kernel functions, similar conclusions can also be reached.

4.1. Assessment metrics and experiment data

We usually denote the minority class as a positive class, and majority class as a negative class. The purpose of a classifier designed for imbalanced data is to balance the detection rate of positive and negative classes. G-mean is generally used to synthetically evaluate the performance of these classifier, it can be formulated as:

$$\mathrm{G-Mean} = \sqrt{\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} \times \frac{\mathrm{TN}}{\mathrm{TN}+\mathrm{FP}}}$$

It can also reflect the balance degree of the detection rates of different classes. In addition, F1 score is widely used to compare the performance of different classifiers on imbalanced data, and it is formulated as follows:

$$recall = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}, \quad pre = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}$$

$$F_1 = \frac{2pre \times recall}{pre + recall}$$

This paper uses G-mean and F1 score to compare the performance of KMODM and other classifiers.

The standard UCI data sets used in our experiments are shown in Table 1. The number after the name of the data set represents the positive class that we select in the original data set. #exam is the number of examples in whole data set; #attri represents the data dimension; #posi and #nega denote the numbers of positive and negative examples, respectively; and IR indicates the imbalanced rate of the data set. According to the imbalance rate, all UCI data sets in Table 1 are divided into two categories: the lightly imbalanced data set and heavily imbalanced data set.

Table 1
UCI data sets.

	Dataset	# exam	#attri	# posi / # nega	IR
Lightly imbalanced	breast	683	9	239/444	1.86
	german	1000	24	300/700	2.33
	haberman	306	3	81/225	2.78
	glass123	214	9	51/163	3.20
	glass7	214	9	29/185	6.38
	glass3	214	9	17/197	11.60
Heavily imbalanced	ecoli6	336	7	20/316	15.8
	car3	1728	24	69/1659	24.04
	yeast	1484	8	51/1433	28.10
	abalone19	4177	8	32/4145	129.53

4.2. Visualization experiments on two-dimensional data

In order to illustrate the optimization effect of KMODM to the separating hyperplane, we generate two-dimensional data with the normal distribution to construct imbalanced training examples. Moreover, to verify that the *D(x)* proposed in this paper is more effective for the ODM algorithm, the two existing conformal functions in Eqs. (7) and (9) are combined with the ODM respectively, named as ODM-7 and ODM-9. Fig. 3 gives a comparison of the separating hyperplanes learned by ODM [22], ODM-7, ODM-9, CSSVM [9], AKSSVM [17], WKSMOTE [7], LCSDM [5] and KMODM when the imbalanced rate of training data is IR=10.

As can be seen from Fig. 3, the separating hyperplane of ODM seriously skews toward the minority class, resulting in a large number of the misclassified positive examples. Several other baseline algorithms adopt different technologies to optimize the conventional classifier to alleviate the skew of the separating hyperplane in a certain extent. However, due to the shortcomings of conventional classifiers and the technologies their adopted, the performances of these classifiers are not as good as expected. For example, the conformal functions of ODM-7 and ODM-9 are originally designed for SVM, in which the difference between the mechanism of SVM and ODM has not been considered. Further more, the conformal function of ODM-9 does not take into account the effects of data imbalance, thus the two separators in Fig. 3(b) and (c) are worse than our method. In Fig. 3(f), the separating hyperplane of WKSMOTE [7] seems to be ideal, but WKSMOTE is sensitive to the distribution of training data and its performance is not stable during the experiment. LCSDM [5] is to learn the classifier and to optimize the margin distribution based on the least misclassification loss. The skew of its separating hyperplane toward the minority classes still exists. In Fig. 3(h), the proposed KMODM inherits the advantages of the kernel scaling and ODM and alleviates the skew of the classifier. Compared with ODM and other baseline algorithms, KMODM achieves better generalization performance and higher classification accuracy of positive classes.

Fig. 4(b)–(e) compares KMODM, ODM and LCSDM on imbalanced data with different IRs. Fig. 4(a) shows the separator learned by ODM when training data are balanced (IR=1). It can be seen that the separating hyperplane of ODM can correctly detect most of the data except for several data located in the overlapped area between two classes. By under sampling the positive class in Fig. 4(a), we can get a series of imbalanced train data with different IR. Therefore, we consider the ODM on balanced data in Fig. 4(a) as an approximated ideal classifier. When the IR of the training data increases, the closer the separator approaches the ideal classifier, the better robustness the separator is. Fig. 4(b)–(e) compare the separators of ODM, LCSDM and KMODM when IR changes from 5 to 30. It is obvious that, with the increase of IR, the separating hyperplane of KMODM is closer to that of ideal classifier than that of other classifiers. This means that KMODM is more robust to the change of IR of training data compared to other methods. Similar conclusions can be obtained by the change of the G-mean score when IR increases as shown in Fig. 4(f). The G-mean is the average result of 10 repeated experiments.

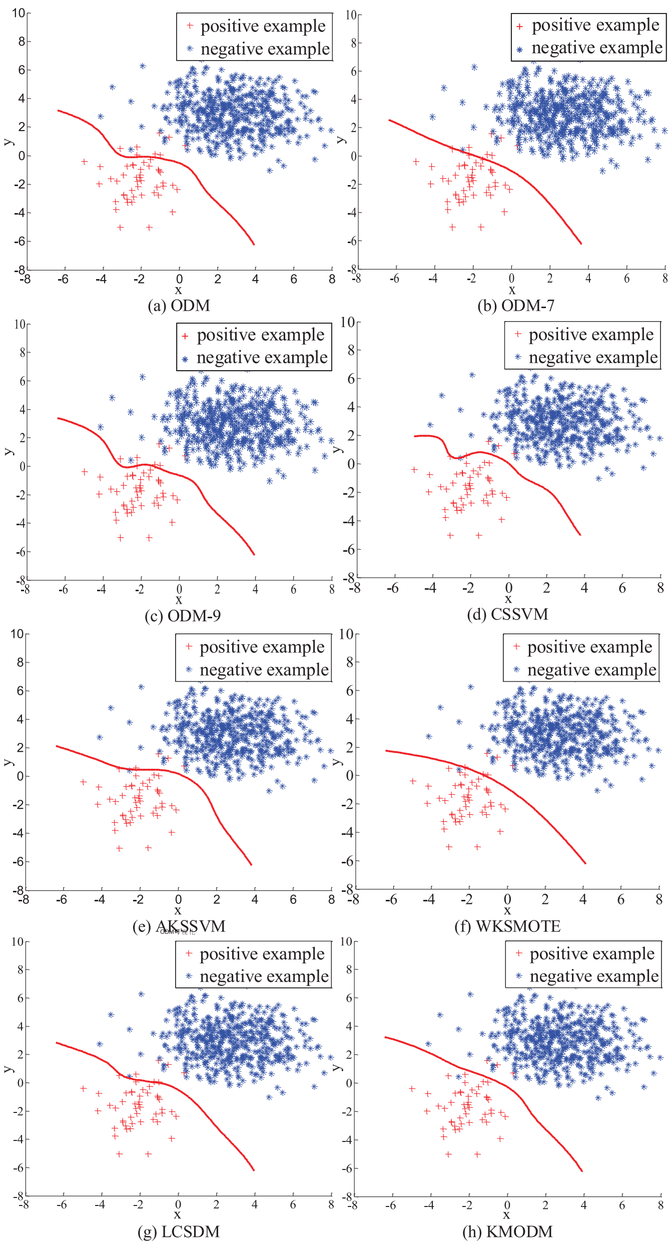

Fig. 3. Generated imbalanced data and visualized separators (IR=10).

4.3. Experiments on UCI data sets

Before reporting the experiment results, all features were normalized into the interval of [0, 1]. For each UCI data set, half examples are randomly selected to compose the training data, and the rest is used as the test data. It is guaranteed that the proportion of negative and positive examples is equal to IR in both the training data and test data.

4.3.1. Comparative study

In order to verify the performance of KMODM, all classifiers mentioned in Section 4.2, are selected to compare with KMODM. For ODM, ODM-7, ODM-9 and KMODM, the regularization parameters C_1, C_2 are selected from $[2^0, \ldots, 2^{10}]$ while their sparse param-

Table 2
Means and standard deviations of G-mean with RBF kernel.

	ODM[22]	ODM-7	ODM-9	CSSVM[9]	AKSSVM[17]	WKSMOTE[7]	LCSDM[5]	KMODM
breast	61.2 ± 4.0	65.3 ± 3.1	62.5 ± 2.0	65.8 ± 4.0	65.5 ± 1.5	66.7 ± 2.1	66.3 ± 3.5	$\mathbf{68.9 \pm 2.0}$
german	73.5 ± 3.1	75.4 ± 2.4	74.1 ± 2.9	74.2 ± 1.6	76.0 ± 2.0	75.9 ± 1.9	75.4 ± 1.4	$\mathbf{77.6 \pm 1.1}$
haberman	64.7 ± 2.4	67.5 ± 1.7	64.5 ± 2.3	66.2 ± 2.0	66.5 ± 0.9	65.8 ± 2.2	68.1 ± 1.9	$\mathbf{71.8 \pm 1.6}$
glass123	87.0 ± 1.6	89.0 ± 2.1	87.7 ± 1.6	88.5 ± 1.7	89.3 ± 5.4	$\mathbf{92.7 \pm 1.6}$	89.5 ± 2.7	91.9 ± 2.0
glass7	89.8 ± 4.1	92.2 ± 1.5	90.0 ± 1.9	91.2 ± 3.1	93.4 ± 1.1	91.8 ± 2.3	91.1 ± 1.5	$\mathbf{95.3 \pm 1.4}$
glass3	21.0 ± 2.5	57.4 ± 2.8	25.1 ± 3.1	48.2 ± 2.3	75.1 ± 6.7	$\mathbf{76.5 \pm 2.0}$	68.8 ± 4.9	76.2 ± 3.9
ecoli6	64.9 ± 2.7	83.1 ± 2.5	73.6 ± 2.3	78.2 ± 3.1	96.5 ± 2.5	95.5 ± 3.1	98.0 ± 1.6	$\mathbf{98.9 \pm 1.1}$
car3	98.5 ± 1.9	98.0 ± 2.3	97.4 ± 1.1	98.2 ± 1.5	98.6 ± 2.1	$\mathbf{99.2 \pm 1.3}$	98.7 ± 3.3	99.1 ± 2.1
yeast	56.9 ± 3.1	78.4 ± 1.8	71.3 ± 2.2	68.2 ± 2.4	81.5 ± 3.2	78.9 ± 1.7	79.3 ± 1.4	$\mathbf{83.6 \pm 1.8}$
abalone19	5.1 ± 3.2	53.2 ± 2.2	13.3 ± 4.1	54.2 ± 3.0	63.9 ± 8.1	59.7 ± 5.2	68.1 ± 5.2	$\mathbf{72.9 \pm 6.3}$

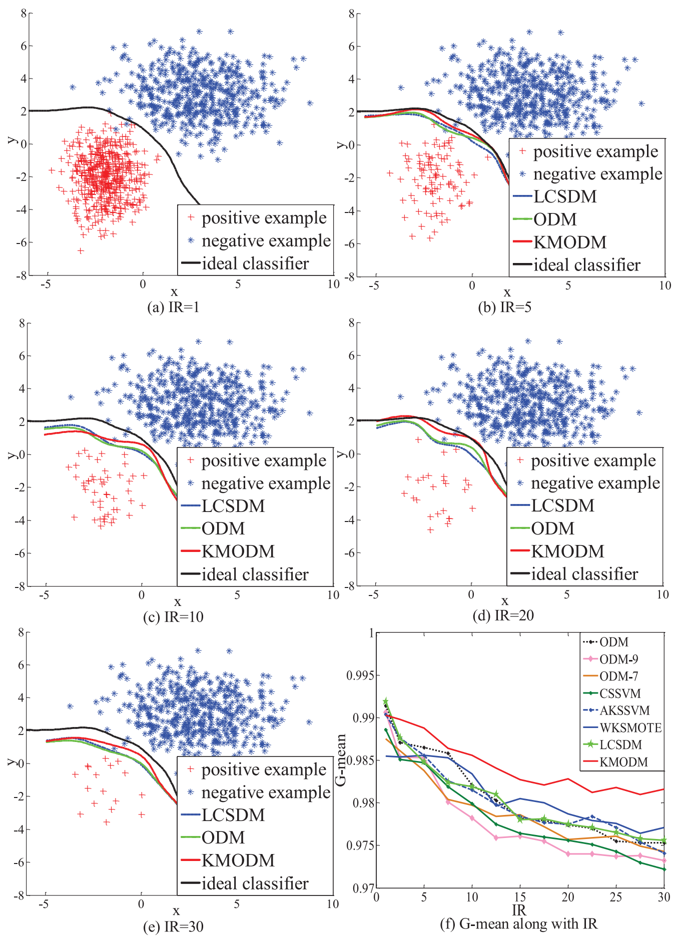

Fig. 4. The influence of different IR of generated data on classifier.

eters are fixed to 0.2. The parameters that control the space expansion coefficients of KMODM are optimized by the grid search and 5-fold cross validation from $[10^{-5}, \ldots, 10^{0}]$. The number of synthetic examples of WKSMOTE is equal to the difference between the number of the majority class and minority class. The balance point of LCSDM is fixed to 0.6. Other parameters are set as the same values as [5]. The parameter selection methods of CSSVM and AKSSVM are as the same as [9] and [17], respectively. The width of the RBF kernel for all methods is selected by 5-fold cross validation from $[2^{-10}, \ldots, 2^{5}]$. The experiments on each data set are repeated 30 times and the means and standard deviations of G-mean and F1 score are reported in Tables 2 and 3. The bolded data are the best results.

From the experiment results, we can see that KMODM has the highest average G-mean and F1 score on breast, Haberman, glass7, ecoli6, yeast and abalone19. Although the best results are not obtained from other data sets, these results are suboptimal in most cases. This indicates that KMODM is superior to other methods on most data sets. On the heavily imbalanced data sets such as ecoli6, car3, yeast, and abalone19, KMODM obtains the significantly improved performance compared with other methods except WKSMOTE. It is worth noting that on abalone 19, since the IR of training examples reached 129.53, it is difficult to detect minority examples when using ODM, this caused a lower G-mean score of it. The experimental results show that when RBF is used as a kernel function, KMODM has better performance than other classifiers in most data sets, especially on heavily imbalanced data sets. In other words, KMODM has higher classification accuracy and higher balanced detection rates than other classifiers in most cases.

4.3.2. *Margin distribution*

Fig. 5 shows the cumulative margin distributions of all competing classifiers with the RBF kernel on German. The similar results can be obtained on other data sets as well. In Fig. 5, MD represents the margin distribution of total examples, PMD represents the margin distribution of positive examples, and the intersection point of the curve and x axis represents the corresponding minimum margin. The larger value of the minimum margin and the more right the curve represent the better generalization performance of a classifier.

As we can see from Fig. 5, compared with ODM, KMODM suffers from a certain loss of overall generalization performance, but it has better positive generalization performance. This is because the proposed kernel modification method increases the spatial resolution of the regions where positive examples are located. It also results in the separating hyperplane offset to negative class. Compared with the margin distributions of other algorithms in Fig. 5(b)-(g), it is obvious that the generalization performance of KMODM is better. In Fig. 5(g), the LCSDM algorithm also broads the margin of the minority class. Since the LDM algorithm adopted in LCSDM also optimizes the margin distribution like ODM, LCSDM and KMODM have a slight difference in generalization performance.

4.3.3. *CPU time cost*

All experiments in this paper are performed with matlab2014a on the computer with 2×3.3 GHz CPUs and 4GB memory. We compared CPU time of ODM, ODM-7, ODM-9, CSSVM, AKSSVM, WKSMOTE, LCSDM and KMODM on different UCI data sets. Without the time of grid search and five-fold cross-validation to search the optimal value of parameters, the average time cost of each algorithm is shown in Fig. 6. Since ODM is executed two times in KMODM, the computation cost of KMODM is higher than that of ODM. However, compared with other baseline algorithms, especially ODM-7, CSSVM and LCSDM, KMODM shows a significantly low computation cost. These results also illustrate that KMODM is computationally efficient.

Table 3
Means and standard deviations of F1 score with RBF kernel.

Dataset	ODM[22]	ODM-7	ODM-9	CSSVM[9]	AKSSVM[17]	WKSMOTE[7]	LCSDM[5]	KMODM
breast	72.1 ± 2.3	78.2 ± 1.1	72.0 ± 3.0	69.4 ± 2.1	74.2 ± 2.1	76.6 ± 2.5	75.4 ± 2.4	**79.7 ± 2.0**
german	74.6 ± 2.4	76.9 ± 2.5	73.2 ± 2.1	76.3 ± 1.8	77.8 ± 2.3	**78.4 ± 1.9**	78.2 ± 2.7	77.4 ± 2.2
haberman	65.7 ± 2.2	68.7 ± 1.8	66.0 ± 2.4	67.2 ± 2.0	69.5 ± 1.8	69.4 ± 2.5	70.3 ± 1.7	**71.6 ± 1.9**
glass123	83.4 ± 1.5	86.6 ± 2.3	84.5 ± 2.3	85.6 ± 2.8	86.8 ± 2.4	86.4 ± 2.2	**88.1 ± 2.3**	87.7 ± 2.5
glass7	88.0 ± 3.1	92.7 ± 2.2	89.8 ± 2.7	88.3 ± 2.1	92.5 ± 2.3	93.1 ± 2.0	92.9 ± 1.3	**94.7 ± 2.0**
glass3	68.4 ± 2.6	75.1 ± 1.7	74.4 ± 3.2	73.1 ± 1.9	76.8 ± 1.2	77.1 ± 3.2	76.0 ± 2.9	**78.3 ± 3.1**
ecoli6	73.5 ± 2.7	95.2 ± 2.4	82.8 ± 2.5	90.9 ± 2.7	92.6 ± 2.4	**98.4 ± 2.5**	95.6 ± 1.9	98.3 ± 1.4
car3	83.9 ± 3.3	88.4 ± 3.6	82.7 ± 1.4	85.0 ± 1.6	84.3 ± 3.4	92.4 ± 2.6	91.1 ± 2.1	**93.7 ± 1.9**
yeast	71.7 ± 2.5	87.9 ± 2.1	80.3 ± 2.2	82.4 ± 2.1	87.0 ± 1.6	89.1 ± 1.9	89.3 ± 2.4	**91.7 ± 2.7**
abalone19	47.5 ± 2.3	56.5 ± 2.4	53.9 ± 2.1	52.7 ± 2.2	58.7 ± 2.7	60.1 ± 2.4	65.5 ± 2.0	**68.7 ± 3.2**

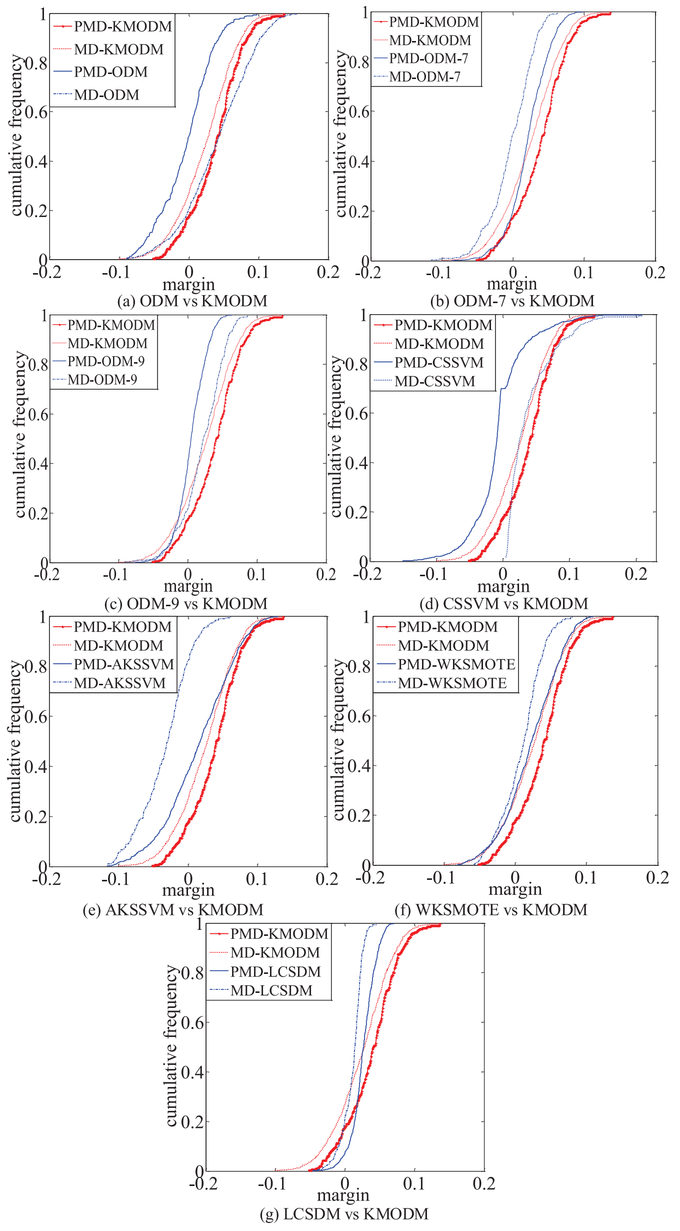

Fig. 5. Cumulative frequency (*y*-axis) with respect to margin (*x*-axis) of ODM, ODM-7, ODM-9, CSSVM, AKSSVM, WKSMOTE, LCSDM and KMODM on German data set using RBF kernel.

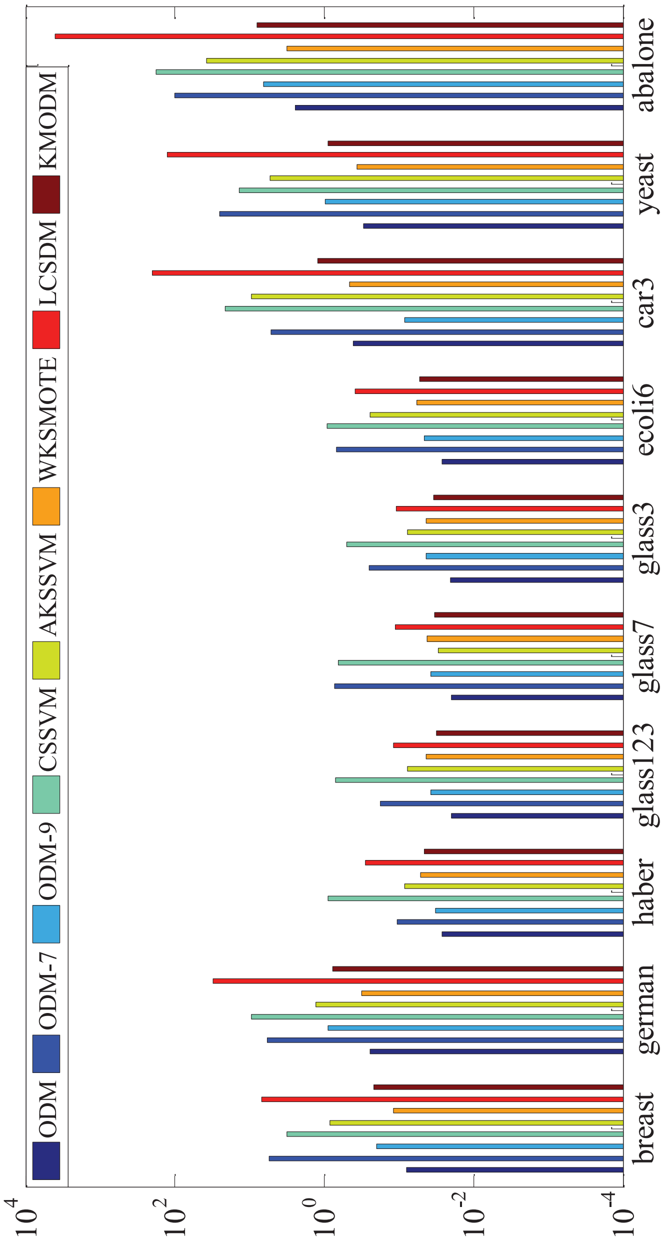

Fig. 6. CPU time cost on UCI data sets.

5. Conclusions

In this paper, we proposed a classification method named KMODM based on optimal distribution learning for imbalanced data. The kernel scaling method was introduced to modify the kernel function of KMODM, and a new conformal transform function was constructed to improve kernel scaling method for KMODM. In the conformal function, two free parameters were introduced to adjust the spatial expansion coefficients of different areas, thus eliminating the influence of imbalanced data on the performance of KMODM. The experiment results show that, compared with the baseline classifiers, the proposed KMODM not only has the higher detection accuracy of the minority class and more balanced detection rate, but also inherits the better generalization performance and higher computation efficiency of ODM. Especially facing heavily imbalanced data, KMODM has obvious superior performance.

The relationship between two free parameters in conformal function and training data as well as its estimation methods are two directions in the future study. Data imbalance is more common in multi classification tasks, it will be worth in the future to using kernel scaling method to improve multi-class ODM.

Acknowledgements

This work was supported in part by National Key R&D Program of China (2018YFB1305900), the National Natural Science Foundation of China under Grant 61672216 and 61673162, the Scientific Research Program of Outstanding Young Talents in Universities of Fujian Province, the youth project of Fujian Education Department (JAT170457), the Fujian Natural Science Foundation (2018J01806) and the Open Project of Electronic Information and Control University Engineering Research Center of Fujian Province (EIC1705).

References

[1] Francisco J. Castellanos, et al., Oversampling imbalanced data in the string space, *Pattern Recognit. Lett.* 103 (2018) 32–38.
[2] Xiaohui Yuan, L. Xie, M. Abouelenien, A regularized ensemble framework of deep learning for cancer detection from multi-class, imbalanced training data, *Pattern Recognit.* 77 (2018) 160–172.
[3] Roozbeh Razavi-Far, M. Farajzadeh-Zanjani, M. Saif, An integrated class-imbalance learning scheme for diagnosing bearing defects in induction motors, *IEEE Trans. Ind. Inf.* 13 (6) (2017) 2758–2769.
[4] G. Wu, E.Y. Chang, KBA: kernel boundary alignment considering imbalanced data distribution[J], IEEE Trans. Knowl. Data Eng. 17 (6) (2005) 786–795.
[5] Cheng Fanyong, et al., Large cost-sensitive margin distribution machine for imbalanced data classification, Neurocomputing 224 (2017) 45–57.
[6] Nitesh V. Chawla, et al., SMOTE: synthetic minority over-sampling technique, *J. Artif. Intell. Res.* 16 (1) (2002) 321–357.
[7] J. Mathew, C.K. Pang, M. Luo, et al., Classification of imbalanced data by oversampling in kernel space of support vector machines[J], IEEE Trans. Neural Netw. Learn. Syst. 99 (2017) 1–12.
[8] W.C. Lin, C.F. Tsai, Y.H. Hu, et al., Clustering-based undersampling in class-imbalanced data[J], Inf. Sci. 409-410 (2017) 17–26.
[9] K. Veropoulos, C. Campbell, N. Cristianini, Controlling the sensitivity of support vector machines[C], in: International Joint Conference on Ai., 1999, pp. 55–60.
[10] S. Dhar, V. Cherkassky, Development and evaluation of cost-sensitive universum-SVM[J], IEEE Trans. Cybern. 45 (4) (2017) 806–818.
[11] L. Zhang, D. Zhang, Evolutionary cost-sensitive extreme learning machine[J], IEEE Trans. Neural Netw. Learn. Syst. 28 (12) (2017) 3045–3060.
[12] H. Masnadi-Shirazi, N. Vasconcelos, Cost-sensitive boosting, *IEEE Trans. Pattern Anal. Mach. Intell.* 33 (2) (2010) 294–309.
[13] G. Wu, E.Y. Chang, Adaptive feature-space conformal transformation for imbalanced-data learning[C], in: Machine Learning, Proceedings of the Twentieth International Conference, DBLP, 2003, pp. 816–823.
[14] Haydemar Núñez, L. Gonzalez-Abril, C. Angulo, Improving SVM classification on imbalanced datasets by introducing a new bias, J. Classif. 34 (4) (2017) 427–443.
[15] Wu Amari, Improving support vector machine classifiers by modifying kernal functions[J], Neural Netw. 12 (6) (1999) 783–789.
[16] W. Si, S.I. Amari, Conformal transformation of kernel functions: a data-dependent way to improve support vector machine classifiers[J], Neural Process. Lett. 15 (1) (2002) 59–67.
[17] A. Maratea, A. Petrosino, M. Manzo, Adjusted F-measure and kernel scaling for imbalanced data learning[J], Inf. Sci. 257 (2) (2014) 331–341.
[18] P. Williams, S. Li, J. Feng, et al., Scaling the kernel function to improve performance of the support vector machine[C, in: Advances in Neural Networks - Isnn 2005, Second International Symposium on Neural Networks, Chongqing, China, May 30 - June 1, 2005, Proceedings, DBLP, 2005, pp. 831–836.
[19] Lev Reyzin, R.E. Schapire, How boosting the margin can also boost classifier complexity, in: International Conference, DBLP, 2006, pp. 753–760.
[20] Wei Gao, Z.H. Zhou, On the doubt about margin explanation of boosting, *Artif. Intell.* 203 (5) (2010) 1–18.
[21] Teng Zhang, Z.H. Zhou, Large margin distribution machine, in: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2014, pp. 313–322.
[22] Zhang, T. and Zhou, Z.-H. Optimal margin distribution machine. *CoRR, abs/1604.03348*, 2016.
[23] Teng Zhang, Zhi-Hua Zhou, Multi-class optimal margin distribution machine, in: Proceedings of the 34th International Conference on Machine Learning, 70, PMLR, 2017, pp. 4063–4071.