2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

An Asymmetric Stagewise Least Square Loss Function for Imbalanced Classification

Guibiao Xu, Bao-Gang Hu and Jose C. Principe

***Abstract*— In this paper, we present an asymmetric stagewise least square (ASLS) loss function for imbalanced classification. While keeping all the advantages of the stagewise least square (SLS) loss function, such as, better robustness, computational efficiency and sparseness, the ASLS loss extends the SLS loss by adding another two parameters, namely, ramp coefficient and margin coefficient. Therefore, asymmetric ramps and margins can be formed which makes the ASLS loss be more flexible and appropriate for processing class imbalance problems. A reduced kernel classifier of the ASLS loss is also developed which only uses a small part of the dataset to generate an efficient nonlinear classifier. Experimental results confirm the effectiveness of the ASLS loss in imbalanced classification.**

I. Introduction

In this paper, we consider the problem of binary classification. In classification, the quality of a classifier $f(\boldsymbol{x})$ is measured by a problem dependent loss function $l\big(f(\boldsymbol{x}),t\big)$, where $t \in \{\pm 1\}$ is the true label of pattern $\boldsymbol{x}$. $l\big(f(\boldsymbol{x}),t\big)$ can also be written as $l(z)$, where $z = tf(\boldsymbol{x})$ is the margin variable and can be used to measure the confidence of classification. Given a training set $\{(\boldsymbol{x}_i, t_i)\}_{i=1}^N$, where each training pattern $\boldsymbol{x}_i \in \mathbb{R}^d$, the classifier $f(\boldsymbol{x})$ can be found by empirical risk minimization of $l(z)$. Misclassification error rate (**0-1 loss**) $l_{0-1} = ||(-z)_+||_0$, where $(\cdot)_+$ denotes the positive part and $||\cdot||_0$ denotes the L_0 norm, is the most appealing loss function for classification because it relates to the misclassification probability directly. However, the noncontinuity and nonconvexity of the 0-1 loss make its optimization NP-hard [1]. Therefore, researchers apply various convex upper bounds of the 0-1 loss to alleviate this computational problem [2], such as the hinge loss function $l_{hinge}(z) = [(1-z)_+]^q (q = 1 \text{ or } 2)$, the logistic loss function $l_{log}(z) = \log[1 + \exp(-z)]$, the least square (**LS**) loss function $l_{ls}(z) = (1-z)^2$, and the exponential loss function $l_{exp}(z) = \exp(-z)$ (see Fig.1). These convex surrogate loss functions are popular because of their virtues of convex optimization like unique optima, abundant convex optimization tools and theoretic generalization error bound analysis [3]. However, these convex loss functions are poor approximation to the 0-1 loss and less robust. Despite of the disadvantages of nonconvex loss functions, various algorithms of nonconvex loss functions are studied in [4], [5] which prove that nonconvex loss functions have higher generalization ability, better scalability and better robustness. The successful applications of deep neural networks further shows the promising future of nonconvex loss functions [6]. In [7], Yang and Hu innovatively proposed a stagewise least square (**SLS**) loss function that gradually approximates a nonconvex squared ramp loss function by adaptively updating the targets (the details are in Section II-B). SLS loss inherits the advantages from both convex and nonconvex loss functions. Correntropy loss function (**C-loss**) $l_C(z) = \beta[1 - \exp(-\frac{(1-z)^2}{2\sigma^2})]$, where $\beta = \frac{1}{1-\exp(-\frac{1}{2\sigma^2})}$ and σ is the correntropy window width, is another nonconvex loss function that was proposed in [8]. One of the appealing advantages of C-loss is that it is more robust to overfitting compared with other loss functions. Both the SLS loss and the C-loss are also shown in Fig.1.

Imbalanced classification is another key problem in classification. Because all the above loss functions assume that the class distributions and misclassification costs are balanced, classifiers based on these assumptions tend to classify all the patterns to be negatives[1] when they run into imbalanced datasets. The objective of imbalanced classification is trying our best to separate positives from negatives, and it usually costs more if we classify positives to be negatives than otherwise. Hence, a variety of class imbalance learning methods [9], [10] have been developed which could be broadly divided into external methods and internal methods [20], [21]. External methods are about data pre-processing so as to balance the classes, while internal methods focus on algorithmic modifications in order to reduce their sensitiveness to class imbalance. MetaCost [11], one-side selection [12]

[1]In this paper, we use +1 (positive) to represent the minority class and -1 (negative) to represent the majority class.

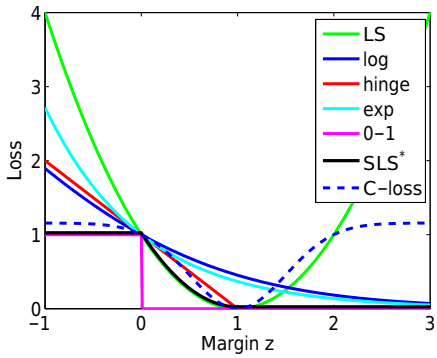


Fig. 1. Loss functions in classification ($\sigma = 0.5$ for C-loss).

Guibiao Xu and Bao-Gang Hu are with the NLPR, Institute of Automation, Chinese Academy of Sciences, Beijing, China (email: {guibiao.xu, hubg}@nlpr.ia.ac.cn).

Jose C. Principe is with the CNEL, Department of Electrical & Computer Engineering, University of Florida, Gainesville, FL 32611, USA (email: principe@cnel.ufl.edu).

This work was supported by NSFC grants #61075051, #61273196 and China Scholarship Council.

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

TABLE I

THE METHODS FOR SVMS TO DEAL WITH IMBALANCED CLASSIFICATION

No.	Method	Principle
1	Active Learning [17]	**Use** active learning to balance the classes
2	Granular SVM-Repetitive Undersampling (**GSVM-RU**) [18]	**Use** granular computing theories to undersample the negatives
3	Different Error Costs (**DEC**) [19]	**Apply** different misclassification costs to different classes
4	Margin Calibration [20]	**Combine** margin compensation with DEC
5	Fuzzy SVM-Class Imbalance Learning (**FSVM-CIL**) [21]	**Assign** different fuzzy-membership values to different classes
6	Total Margin-based Adaptive Fuzzy SVM (**TAF-SVM**) [22]	**Incorporate** fuzzification and DEC into total margin-based SVM
7	Proximal SVM with DEC (**PSVM-DEC**) [23]	**Apply** DEC to proximal SVM
8	Kernel Boundary Alignment (**KBA**) [24]	**Adjust** the class boundary by adaptive conformal transformation
9	Scaling the Kernel Function [25]	**Use** the prior knowledge obtained in a primary training to conformally rescale the kernel function
10	Kernel-Target Alignment [26]	**Use** kernel-target alignment to measure the degree of agreement between a kernel and a learning task
11	SMOTE with Different Costs (**SDC**) [27]	**Combine** SMOTE with DEC
12	Hybrid Kernel Machine Ensemble (**HKME**)[28]	**Combine** discriminative SVM with one-class SVM
13	One-Class SVM [29]	**Only** use the positives to train a recognition-based one-class SVM
14	Asymmetric SVM (**ASVM**) [30]	**Maximize** the class-margin and core-margin simultaneously
15	z-SVM [31]	**Orient** the trained decision boundary of SVM by z

and SMOTE [13] are typical external methods. Cost-sensitive learning [14] and asymmetric surrogate loss [15] are mainly used in internal methods. Support vector machines (**SVMs**) [16] are popular classifiers because of their remarkable generalization performance. But their performance is also greatly reduced when they are applied to imbalanced datasets for the reason that the negatives push the decision boundaries closer to the positives [9], [24], [27]. Table I systematically lists the ways for SVMs to deal with imbalanced classification. All of them try to apply data pre-processing, cost-sensitive learning, or both to recover the decision boundary skewness.

For the aforementioned SLS loss, it also gives equal penalties to all the training patterns. Intuitively, it is sensitive to class imbalance (see Fig.3(c)). As the SLS loss is convex within each stage and finally approximates the squared ramp loss (see Fig.1), it naturally combines the merits of convex and nonconvex losses. Thus, it is hopeful to extend the SLS loss to imbalanced cases so that its application is expanded. In this paper, we combine the ideas of DEC [19] and Margin Calibration [20] and propose an asymmetric stagewise least square (**ASLS**) loss function which is also a kind of asymmetric surrogate loss [15]. On one hand, we apply different penalties to different classes which is widely used in imbalanced classification. On the other hand, positives usually lie further from the ideal boundary because of the problem of sampling and training data ratio [27], balanced margins may hinder class imbalance learning methods from totally recovering the boundary bias, which is the bias introduced by the model. Thus, we place larger positive margin than negative margin in order to further help recover the boundary bias. Experimental results show that the boundary bias can be effectively recovered if we apply this ASLS loss in the imblanced datasets. In addition, we also develop a reduced [32] kernel classifier of ASLS loss in order to further improve the scalability as well as the training and testing speed. Experiments on several benchmark datasets confirm the effectiveness of the reduced kernel classifier of ASLS loss.

The rest of the paper is organized as follows: Section II presents the ASLS loss function; in Section III, we use ASLS loss to build both linear and kernel classifiers. Illustrative examples and experimental results are given in Section IV. Finally, Section V summarizes the whole paper.

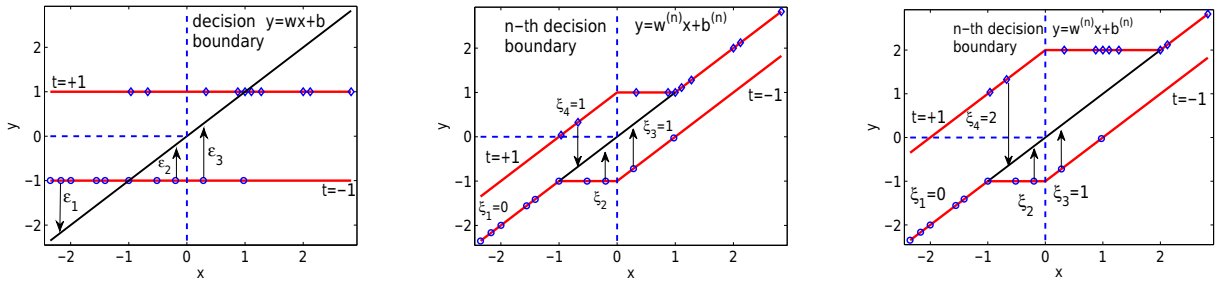
II. ASYMMETRIC STAGEWISE LEAST SQUARE LOSS

A. Least Square Loss

Firstly, we briefly introduce the LS loss and its properties so as to make clear that why the SLS loss is introduced. The LS loss $l_{ls}(z) = (1-z)^2$ is the simplest convex loss which possesses the computational advantage, and it is popular in regression problems. But due to the differences between regression and classification problems, its shortcomings are obvious when it is applied to classification problems:

1) In regression problems, LS loss is the optimal loss function when the noise in the dataset is Gaussian noise. However, in the classification scenario, t is clamped onto two discrete values -1 and +1. As a result, Gaussian distribution is no longer proper to describe the residue variable $\varepsilon = t - y$ ($y = f(\boldsymbol{x})$). From Fig.2(a), we can easily see that ε tends to be large. Thus, the resulting classifier may be poor in performance.
2) Different from other loss functions, LS loss is not monotonically decreasing and it even penalizes the patterns with large margins $z > 1$ which, from the perspective of statistical learning, can be classified with high confidence [1]. In fact, it can be seen from Fig.1 that LS loss encourages a margin of exactly one.
3) LS loss is boundless and is sensitive to outliers.
4) Kernel classifiers of LS loss, such as Proximal SVM (**PSVM**) [23] and least square SVM (**LSSVM**) [33], lose the important property of sparseness. Because the VC-complexity of kernel classifiers, the training and testing speed all are closely related to the number of support vectors (**SVs**) [16], nonsparseness limits the application scope of LS loss in kernel classifiers [34].

Considering these limitations of LS loss, Yang and Hu [7] succeeded to solve the above problems by carefully updating



(a) (b) (c)

Fig. 2. Classification in the 1-D feature space: (a) regularized least square(ridge regression), patterns locate on the two red axis-aligned tracks, as a result, the residues are large and their p.d.f. isn't like the Gaussian distribution any more; (b) SLS-based classification [7], patterns at the $n+1$-th stage are translated to $\{(x_i, t_i\tau_i^{(n)})\}_{i=1}^N$ which locate on the two new symmetric red tracks; (c) ASLS-based classification and $(r_+, m_+) = (4, 2)$, patterns at the $n+1$-th stage are translated to $\{(x_i, t_i\tau_i^{(n)})\}_{i=1}^N$ which locate on the two new asymmetric red tracks.

the targets τ and forming an LS-style loss function $(\tau - z)^2$ stagewisely, which is called the SLS loss function.

B. Stagewise Least Square Loss

The key idea of SLS loss is to use stagewisely updated targets τ to solve a series of LS problems so that it can finally approximate the squared ramp loss function (see Fig.1). At the n-th stage, the SLS loss is [7]:

$$l_{SLS}^{(n)}(z) = (\tau^{(n)} - z)^2, \tag{1}$$

where the targets are updated as follows:

$$\begin{aligned} \tau^{(0)} &= 1, \\ \tau^{(n+1)} &= z^{(n)} + S(1 - z^{(n)}), \end{aligned} \tag{2}$$

where $z^{(n)} = ty^{(n)}$, $y^{(n)} = f^{(n)}(\boldsymbol{x})$ [2] and $S(v) = \max(0, \min(1, v))$. Note that at the 0-th stage, the SLS loss is actually the original LS loss. Fig.2(b) shows an illustrative example of the SLS loss. According to (2), the updating rules are: **1)** if $z^{(n)} > 1$, we are confident about these predictions, then $\tau^{(n+1)} = z^{(n)}$ so that the objective function can emphasis less on these patterns at the next stage; **2)** if $z^{(n)} \in [0, 1]$, these patterns are correctly classified but with insufficient margins, then the SLS loss sets $\tau^{(n+1)} = 1$ in order to increase these patterns' margins at the next stage; **3)** if $z^{(n)} < 0$, these patterns are misclassified, then the SLS loss sets $\tau^{(n+1)} = z^{(n)} + 1$ so as to penalize these misclassifications with moderate losses at the next stage.

When the SLS loss converges, the SLS loss becomes the squared ramp loss (see Fig.1) [7]:

$$l_{SLS}^*(z) = \begin{cases} 0, & \text{if} \quad z > 1, \\ (1-z)^2, & \text{if} \quad z \in [0, 1], \\ 1, & \text{otherwise.} \end{cases} \tag{3}$$

The SLS loss has several desirable properties: 1) it is in an LS-style within each stage, then it is convex at each stage and can be easily optimized; 2) it naturally results in a more sparse kernel classifier (see Section III-B); 3) when converges, the SLS loss approximates the nonconvex squared ramp loss (3) which satisfies the requirements of a robust loss [35], as a result, both generalization ability and robustness are improved. However, from (1) and (2), we can easily find that the SLS loss is sensitive to class imbalance, and this is the problem that we settle in this paper.

[2] $f^{(n)}(\boldsymbol{x})$ denotes the output of a classifier at the n-th stage.

C. Asymmetric Stagewise Least Square Loss

The SLS loss (1) places equal penalties on all the training patterns, as a result, when it is applied to imbalanced datasets, it produces unfavorable classification results for the positives. For example, in Fig.3(c), the positives are uniformly distributed in the upper squared plane, the negatives are uniformly distributed in the lower squared plane and the black dash line between the two squared planes is the ideal decision boundary. However, in a training set, the class ratio of positives over negatives is 1:10. It is easy to verify from Fig.3(c) that the linear decision boundary of the SLS loss [7] is far from the ideal one, which causes a high incidence of false negatives which usually have higher cost. In order to improve the performance of imbalanced classification of the SLS loss, we propose to place different penalties on different classes under the principle of cost-sensitive learning. In Fig.3(c), the positives lie farther from the ideal boundary than the negatives, thus we think that placing balanced margins may increase the bias of the model. Considering this issue and the viewpoint of asymmetric surrogate loss [15], we suggest the margin of positives should be larger in imbalanced classification so as to help improve the classification of positives. In all, we combine the ideas of DEC [19] and Margin Calibration [20], and introduce the asymmetric stagewise least square (**ASLS**) loss function in this paper:

$$l_{ASLS}^{(n)}(z) = \frac{r}{m^2}(\tau^{(n)} - z)^2, \tag{4}$$

where we call r the ramp coefficient and m the margin coefficient. The targets τ are updated as follows:

$$\begin{aligned} \tau^{(0)} &= 1, \\ \tau^{(n+1)} &= z^{(n)} + S(m - z^{(n)}), \end{aligned} \tag{5}$$

where $S(m - z^{(n)}) = \max(0, \min(m, m - z^{(n)}))$. We use the pair (r, m) to represent the pair of ramp and margin

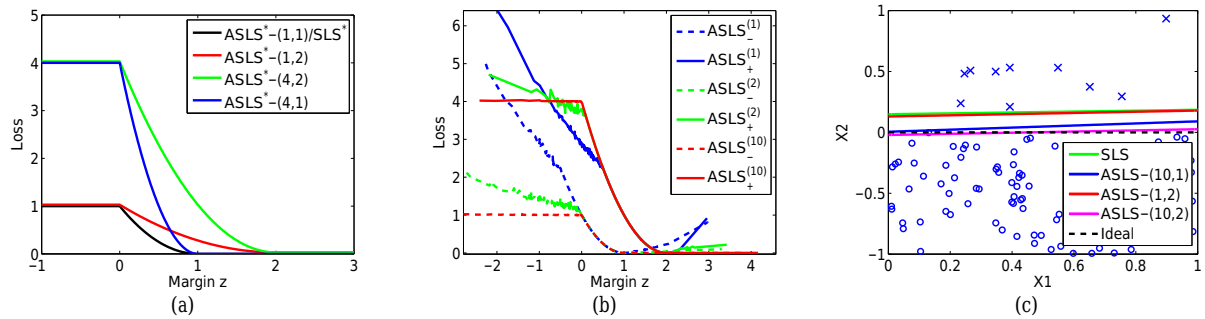


Fig. 3. (a) The ASLS loss functions with various pairs of (r, m); (b)The ASLS loss of each pattern at the 1st, 2nd and 10th stage on the Pima dataset, where $(r_+, m_+) = (4, 2)$; (c) Linear decision boundaries of the SLS loss and various ASLS losses on a training set whose class ratio is 1:10.

coefficients in the ASLS loss. We can easily find that the SLS loss is a particular case of the ASLS loss where $(r, m) = (1, 1)$. When converges, the ASLS loss becomes:

$$l^*_{ASLS}(z) = \begin{cases} 0, & \text{if} \quad z > m, \\ \frac{r}{m^2}(m-z)^2, & \text{if} \quad z \in [0, m], \\ r, & \text{otherwise.} \end{cases} \tag{6}$$

Fig.3(a) shows $l^*_{ASLS}(z)$ with various pairs of (r, m), from which we can find that different pairs of (r, m) lead to different ramp penalties and margins. In this paper, for imbalanced classification, we always set $(r_-, m_-) = (1, 1)$ for the negatives and only modify (r_+, m_+) for the positives. Note that both r_+ and m_+ should not be smaller than 1 in imbalanced classification. Fig.2(b) and Fig.2(c) intuitively illustrate the differences between the SLS and ASLS losses at the n-th stage in the 1-D feature space. For the positives, it begins to penalize when $z < 2$ while for the negatives, it begins to penalize when $z < 1$. That is to say, the positives and negatives tend to have the different margins. Besides, the positives ($r_+ = 4$) finally have larger penalties than the negatives ($r_- = 1$). In this way the ASLS loss shows the idea of cost-sensitive leaning. Fig.3(b) shows that on the Pima dataset, the positives and negatives gradually converge to the different squared-ramp losses in several stages. The linear boundary of the ASLS loss with $(r_+, m_+) = (10, 2)$ in Fig.3(c) approximates the ideal one perfectly, while neither the linear boundary of ASLS-$(10, 1)$ nor ASLS-$(1, 2)$ can approximate the ideal boundary perfectly. This phenomenon in Fig.3(c) proves that combining the class-dependent penalties with the class-dependent margins can help further improve the efficiency of class imbalance learning methods. In Section IV-A and IV-B, we experimentally discuss the effects of ramp and margin coefficients in detail.

III. Classifiers based on the ASLS Loss

A. Linear Classifier

Assume that $\boldsymbol{e}$ is a vector of all ones; X is the data matrix where each row of X is a training pattern $\boldsymbol{x}'$[3]; $\bar{X} = (X, \boldsymbol{e})$ is the augmented data matrix; Λ is an $N \times N$ diagonal matrix, where $\Lambda_{ii} = 1$ if $t_i = -1$ and $\Lambda_{ii} = \frac{r_+}{m_+^2}$ if $t_i = +1$, $i = 1, 2, \cdots, N$; and I is the identity matrix. Define the linear classifier is $y = f(\boldsymbol{x}) = \boldsymbol{\omega}'\boldsymbol{x} + b$, $\bar{\boldsymbol{\omega}} = (\boldsymbol{\omega}', b)'$ and $\Gamma = (t_1\tau_1, t_2\tau_2, \cdots, t_N\tau_N)'$. A linear ASLS classifier (**LASLS**) minimizes the following objective at the n-th stage[4]:

$$\min \frac{1}{2}\bar{\boldsymbol{\omega}}'\bar{\boldsymbol{\omega}} + \frac{\nu}{2}(\bar{X}\bar{\boldsymbol{\omega}} - \Gamma)'\Lambda(\bar{X}\bar{\boldsymbol{\omega}} - \Gamma), \tag{7}$$

where ν is the regularization number. The closed-form solution to (7) is:

$$\bar{\boldsymbol{\omega}} = (\frac{1}{\nu}I + \bar{X}'\Lambda\bar{X})^{-1}\bar{X}'\Lambda\Gamma. \tag{8}$$

As d is usually not very large, LASLS is efficient for massive datasets. However, if $d \gg N$, computing (8) directly will be time-consuming. Fortunately, we can use the following equation to reduce the computational complexity [1]:

$$(\frac{1}{\nu}I + \bar{X}'\Lambda\bar{X})^{-1} = \nu I - \nu^2\bar{X}'(\Lambda^{-1} + \nu\bar{X}\bar{X}')^{-1}\bar{X}. \tag{9}$$

Note that $(\frac{1}{\nu}I + \bar{X}'\Lambda\bar{X})^{-1}$ needs to be done only once and the training complexity of LASLS is $O(\min(N, d)^3)$ [7].

B. Kernel Classifier

The kernel ASLS classifier (**KASLS**) can be obtained by applying the kernel trick $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = <\phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j)>$. Suppose K is the Gram matrix where $K_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j), i, j = 1, 2, \cdots, N$ and $\Psi = (\phi(\boldsymbol{x}_1), \phi(\boldsymbol{x}_2), \cdots, \phi(\boldsymbol{x}_N))$. At the n-th stage, the objective of KASLS is:

$$\begin{aligned} \min \quad & \frac{1}{2}(\boldsymbol{\omega}'\boldsymbol{\omega} + b^2) + \frac{\nu}{2}\boldsymbol{\varepsilon}'\Lambda\boldsymbol{\varepsilon}, \\ s.t. \quad & \Psi'\boldsymbol{\omega} + \boldsymbol{e}b - \Gamma = \boldsymbol{\varepsilon}. \end{aligned} \tag{10}$$

The KKT condition of its Lagrangian [16] is:

$$\begin{aligned} \boldsymbol{\omega} &= \Psi\boldsymbol{\alpha}, \quad & b &= \boldsymbol{e}'\boldsymbol{\alpha}, \\ \boldsymbol{\alpha} &= -\nu\Lambda\boldsymbol{\varepsilon}, \quad & \boldsymbol{\varepsilon} &= \Psi'\boldsymbol{\omega} + \boldsymbol{e}b - \Gamma, \end{aligned} \tag{11}$$

where $\boldsymbol{\alpha}$ is a vector of Lagrange multipliers and is given by:

$$\boldsymbol{\alpha} = (K + \boldsymbol{e}\boldsymbol{e}' + \frac{1}{\nu}\Lambda^{-1})^{-1}\Gamma. \tag{12}$$

The decision boundary of KASLS is:

$$y = f(\boldsymbol{x}) = \sum_{i=1}^{N}\alpha_i k(\boldsymbol{x}, \boldsymbol{x}_i) + b = 0. \tag{13}$$

[3]The superscript $'$ denotes the transpose of vectors or matrices.

[4]Without confusion, the superscript (n) of the stage number is omitted here.

In contrast to PSVM [23] and LSSVM [33] whose sparseness is totally lost, KASLS retains the sparseness to some extent [7]. Because part of ε_i's are close to zero (see Fig.2(c)), their corresponding α_i's are also close to zero based on their relationship (11). Thus, the sparseness of KASLS can be gained by setting an off-training filtering procedure with a small threshold θ, which is exactly the same as KSLS [7]. The computational complexity of (12) is $O(N^3)$ which is also computed only once. It is time-consuming when N becomes large. If conjugate gradient is applied to solve (10), the computational complexity can be reduced to $O(hN^2)$ where h is the rank of matrix $(K+\boldsymbol{e}\boldsymbol{e}'+\frac{1}{\nu}\Lambda^{-1}-I)$ [33]. In the next section, we develop a reduced kernel classifier for the ASLS loss in order to further reduce the computational complexity.

C. Reduced Kernel Classifier

KASLS is time-consuming when N is large. Besides, the SVs of KASLS may still be redundant, which causes over-fitting problem [7]. In order to improve the training and testing speed as well as the scaling property of the resulting classifier, we derive a reduced version of KASLS (**$\mathbf{KASLS_R}$**) which only uses a small portion of the training set to constitute the nonlinear decision boundary.

Reduced SVM was proposed by Lee et al. [32], [36]. Although it only uses a small randomly portion of the training set to its explicit evaluation, it has comparable performance to the conventional SVM. What's more, it also reduces the size of the quadratic program. Here, we change the inequality constraints in the reduced SVM to equality constraints and form the KASLS_R which is exactly an RBF network [1]. Assume that $X_R \in \mathbb{R}^{N_R \times d}$ is a small randomly selected portion of the training set; $\eta = \frac{N_R}{N}$ is the reduced rate; and $K_R \in \mathbb{R}^{N \times N_R}$ is the reduced Gram matrix where $K_{Rij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, $\boldsymbol{x}_i \in X, i = 1, 2, \cdots, N$ and $\boldsymbol{x}_j \in X_R, j = 1, 2, \cdots, N_R$. The decision boundary of KASLS_R is:

$$y = f(\boldsymbol{x}) = \sum_{\boldsymbol{x}_j \in X_R} u_j k(\boldsymbol{x}, \boldsymbol{x}_j) + b = 0, \tag{14}$$

where $\boldsymbol{u} \in \mathbb{R}^{N_R}$ is also a weight vector. The optimization problem of KASLS_R is:

$$\begin{array}{ll} \min & \frac{1}{2}(\boldsymbol{u}^T\boldsymbol{u} + b^2) + \frac{\nu}{2}\boldsymbol{\varepsilon}'\Lambda\boldsymbol{\varepsilon}, \\ s.t. & K_R\boldsymbol{u} + \boldsymbol{e}b - \Gamma = \boldsymbol{\varepsilon}. \end{array} \tag{15}$$

The KKT condition of its Lagrange is:

$$\begin{array}{ll} \boldsymbol{u}=K_R'\boldsymbol{\alpha}, & b=\boldsymbol{e}'\boldsymbol{\alpha}, \\ \boldsymbol{\alpha}=-\nu\Lambda\boldsymbol{\varepsilon}, & \boldsymbol{\varepsilon}=K_R\boldsymbol{u} + \boldsymbol{e}b - \Gamma. \end{array} \tag{16}$$

And the solutions of Lagrange multipliers $\boldsymbol{\alpha}$ are:

$$\boldsymbol{\alpha} = (K_R K_R' + \boldsymbol{e}\boldsymbol{e}' + \frac{1}{\nu}\Lambda^{-1})^{-1}\Gamma = (GG' + \frac{1}{\nu}\Lambda^{-1})^{-1}\Gamma, \tag{17}$$

where $G = (K_R, \boldsymbol{e})$. Computing $(GG' + \frac{1}{\nu}\Lambda^{-1})^{-1}$ directly is time-consuming. Fortunately, like (9), we can also compute it as follows:

$$(GG' + \frac{1}{\nu}\Lambda^{-1})^{-1} = \nu\Lambda - \nu^2\Lambda G(I + \nu G'\Lambda G)^{-1}G'\Lambda. \tag{18}$$

TABLE II
CHARACTERISTICS OF THE EXPERIMENTAL DATASETS

Dataset	# of Samples	# of Positive Samples(%)	# of Features
Pima	768	268 (34.90)	8
Phoneme	5404	1586 (29.35)	5
Vehicle-1	846	212 (25.06)	18
KC1	2109	326 (15.46)	21
Satimage-4	6435	626 (9.73)	36
Vowel-0	990	90 (9.09)	10
Yeast-me2	1484	51 (3.44)	8
Mammography	11180	260 (2.33)	6
Abalone-19	4177	32 (0.77)	8

Since there are only N_R training patterns used for explicit evaluation in KASLS_R, the memory requirement of KASLS_R is $O(NN_R)$ and its computational complexity is approximately $O(N_R^3)$, i.e. $O(\eta^3 N^3)$. If we set η around 0.3, as in most cases the rank h of $(K+\boldsymbol{e}\boldsymbol{e}'+\frac{1}{\nu}\Lambda^{-1}-I)$ is much larger than $0.027N$, the computational complexity of KASLS_R is much smaller than KASLS even when conjugate gradient is applied. The SVs ratio of KASLS_R is just η.

Note that all the above derivation is for the n-th stage. After the n-th stage is finished, we need to update the targets Γ according to (5) for the next stage. We know from the experimental results that the update usually converges in around 10 stages.

IV. EXPERIMENTS

In this section, we present the experimental results of the proposed approaches to show their properties and performance. Table II shows the characteristics of the datasets, among which KC1[5] is about the software engineering measurements, Mammography is generously provided by Dr. Nitesh Chawla [13] and the other seven are from the UCI data repository. If a dataset has more than two classes, we carefully convert all but a certain class into negatives so as to form a binary classification problem and increase the skewness.

A. The Effect of Asymmetric Margins

Cost-sensitive learning is the most popular method to deal with imbalanced classification which has been used in many classifier models and its effectiveness has been broadly proved [10]. Because positives usually lie further from the ideal boundary, placing balanced margins is likely to be a bias introduced by the model. In this section, we are going to show how asymmetric margins help improve the imbalanced classification. Firstly, we generate a set of patterns shown in Fig.4(a) and apply LASLS with different pairs of (r_+, m_+) to it. During this process, we always set the regularization number $\nu = 100$ so that the differences are only caused by the margin coefficient of the ASLS loss. Fig.4(a) shows that the positive margin of LASLS varies with m_+, and the ratio of positive margin over negative margin is $\frac{m_+}{1} = m_+$. This reflects the fact that the margin coefficient is in charge of the

[5]http://promise.site.uottawa.ca/SERepository/datasets-page.html

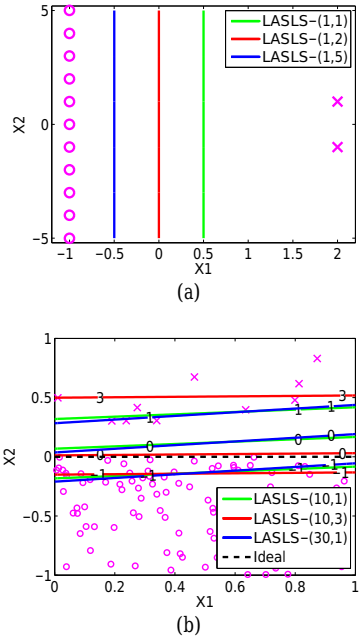


Fig. 4. (a) Asymmetric margins of LASLS on a synthetic dataset; (b) Decision boundaries of LASLS-(10,1), LASLS-(10,3) and LASLS-(30,1) on an artificial dataset.

margins of different classes. Secondly, we generate another set of patterns like Fig.3(c) whose class ratio is also 1:10. We apply LASLS with $(r_+, m_+) = (10,1), (10,3), (30,1)$ to this dataset respectively. Fig.4(b) shows the decision boundaries of these LASLS's. The decision boundary of LASLS-(10,1) doesn't approximate the ideal boundary well. The decision boundary of LASLS-(30,1) is almost the same with LASLS-(10,1) although it has higher ramp coefficient. This phenomenon reflects the fact that only applying the class-dependent penalties sometimes can't fully recover the boundary bias, because the margins of positives and negatives to the ideal boundary are also imbalanced. Thus, we should also consider the margin imbalance in order to fully recover the boundary bias. And that is the reason, we believe, that the decision boundary of LASLS-(10,3) approximates the ideal one well. Therefore, introducing asymmetric margins to imbalanced classification can help improve the classification results as margin imbalance may also exist in the imblanced datasets.

B. Comparison of Different Pairs of Parameters

We investigate how ramp and margin coefficients affect the results of imbalanced classification in this section. We perform LASLS and KASLS with different pairs of (r_+, m_+) on the Vehicle dataset. We don't perform off-training filtering for KASLS here. 10-fold cross validation is applied and Gaussian kernel $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-||\boldsymbol{x}_i - \boldsymbol{x}_j||^2/\sigma^2)$ is used in KASLS. We use *G-mean* [9] to evaluate the imbalanced classification results. And Table III shows the comparison results. Firstly, it is LSLS or KSLS [7] when $(r_+, m_+) = (1,1)$. Compared with LSLS and KSLS, LASLS and KASLS do help to improve the imbalanced classification results a lot if (r_+, m_+) are properly set. Secondly, ramp coefficient contributes more to the improvement of imbalanced classification when compared with margin coefficient. The reason is that positives are much fewer than negatives. If $r_+ = r_-$, positives contribute much less to the objectives (7) or (10) compared with negatives even when all the positives are used. Thus, we have to use class-dependent penalties to balance their contribution. Thirdly, margin coefficient doesn't begin to help improve imbalanced classification until ramp coefficient is large enough and this phenomenon is more severe in LASLS. This is an interesting and confusing phenomenon which we will further investigate in the future. The margin coefficient is not always the larger the better and it should depend on the classifier outputs of positives, because if m_+ is too large, all the positive targets will be m_+ which makes no differences among the positives. Above all, both the ramp and margin coefficients can help improve imbalanced classification if they are set properly.

TABLE III

COMPARISON AMONG DIFFERENT PAIRS OF (r_+, m_+) ON THE VEHICLE DATASET. **CR**=2.99 IS THE CLASS RATIO OF VEHICLE DATASET. IN EACH CELL, THE UPPER VALUE IS FOR LASLS, THE LOWER VALUE IS FOR KASLS. THE BEST VALUES ARE IN BOLD.

m_+ \ r_+	1	0.5CR	CR	1.5CR	2CR	2.5CR	3CR
1	67.26 76.52	73.43 77.49	77.60 82.02	77.73 82.64	77.35 82.95	77.64 83.03	75.54 83.21
1.4	58.32 75.84	70.45 78.11	77.42 82.42	77.96 83.47	77.47 83.62	77.51 84.01	76.19 **84.16**
1.8	48.43 74.53	65.94 78.30	77.19 83.15	77.45 83.95	**78.71** 83.09	77.67 83.64	76.67 83.46
2.2	34.10 73.34	59.19 76.87	76.99 82.15	77.99 83.30	78.38 83.24	77.76 82.88	76.63 83.24
2.6	23.14 71.83	52.36 76.07	74.75 80.92	78.22 82.59	78.63 82.70	78.35 82.71	77.29 83.25
3	16.11 70.91	41.16 74.99	73.52 80.79	77.09 82.30	78.35 82.36	78.47 82.19	77.62 82.13

C. The Effectiveness of $KASLS_R$

We compare $KASLS_R$ with KASLS on the Vowel dataset in order to show the effectiveness of $KASLS_R$. Gaussian

TABLE IV

COMPARISON BETWEEN KASLS AND $KASLS_R$ ON THE VOWEL DATASET. (MEAN(STD))

η \ $KASLS_R$	G-mean (%) Time (s)	η \ $KASLS_R$	G-mean (%) Time (s)
0.05	93.85(2.69) 0.0230(0.0008)	0.1	96.02(1.31) 0.0246(0.0004)
0.2	97.00(1.17) 0.0295(0.0005)	0.3	97.75(1.10) 0.0356(0.0003)
0.4	98.03(0.89) 0.0440(0.0004)	0.5	97.98(0.85) 0.0545(0.0006)
0.6	98.20(0.95) 0.0658(0.0007)	0.7	98.48(1.00) 0.0794(0.0008)
0.8	98.49(0.84) 0.0954(0.0014)	0.9	98.54(0.93) 0.1123(0.0005)
1	98.65(0.93) 0.1330(0.0048)		
KASLS	G-mean (%) Time (s)	Ratio of SVs	
	98.28(2.76) 0.1424(0.0198)	0.7490(0.0109)	

TABLE V

COMPARISON OF LINEAR CLASSIFICATION (MEAN(STD)). THE ASTERISK (*) DENOTES A SIGNIFICANT DIFFERENCE BETWEEN LASLS AND OTHER METHODS. THE BEST VALUES ARE IN BOLD. BELOW THE NAME OF DATASET IS THE RATIO OF CLASSES.

Dataset		G-mean(%)	F-Measure(%)	AUC(%)
Pima (1:1.866)	LASLS	**75.10(5.13)**	**68.06(6.47)**	82.92(4.60)
	PSVM-DEC	73.92(6.70)	66.18(7.22)	**83.02(4.74)**
	SVM-DEC	73.72(6.24)	65.49(6.69)	82.81(4.80)
	SVM-SDC	74.54(5.00)	66.82(5.43)	82.52(4.72)
Phoneme (1:2.407)	LASLS	**76.10(1.98)**	**65.32(2.47)**	**81.30(2.21)**
	PSVM-DEC	$74.38(2.40)*$	$62.93(2.79)*$	81.16(2.18)
	SVM-DEC	$74.81(2.34)$	$63.43(2.71)*$	81.05(2.28)
	SVM-SDC	74.97(2.32)	$61.59(1.63)*$	81.05(2.32)
Vehicle (1:2.99)	LASLS	**78.08(3.41)**	**63.69(4.89)**	85.65(3.68)
	PSVM-DEC	77.14(4.44)	63.45(4.76)	85.01(3.98)
	SVM-DEC	77.80(3.53)	63.19(4.29)	85.61(3.70)
	SVM-SDC	78.04(3.47)	62.74(4.50)	**85.91(3.74)**
Yeast (1:28.10)	LASLS	**81.22(8.89)**	**28.76(9.60)**	87.19(8.82)
	PSVM-DEC	78.98(11.23)	26.64(7.63)	**87.59(8.61)**
	SVM-DEC	80.75(10.44)	28.59(7.82)	87.08(8.67)
	SVM-SDC	78.70(12.34)	27.36(8.45)	87.34(8.53)
Abalone (1:129.53)	LASLS	72.67(12.81)	**6.11(3.06)**	85.79(6.81)
	PSVM-DEC	**76.73(8.19)**	5.40(2.12)	**86.07(6.30)**
	SVM-DEC	73.61(9.60)	4.86(1.59)	85.02(6.96)
	SVM-SDC	74.89(13.28)	4.95(1.75)	84.80(7.13)

kernel, 10-fold cross validation and *G-mean* are all used here. And we set $\theta = 6.3 \times 10^{-2}$ for the off-training filtering of KASLS. The reduced rate $\eta = \frac{N_R}{N}$ ranges from 0.05 to 1. Table IV shows the comparison results. On one hand, the differences of *G-mean* among KASLS_R's are small and we can get good classification results even using only 10 percent of the training set. On the other hand, the differences of *G-mean* between KASLS and KASLS_R's are also small. The SVs ratio of KASLS_R is η. We can see from Table IV that KASLS_R with $\eta = 0.4$ has comparable performance to KASLS whose SVs ratio is 0.7490 and besides, the former training time is much less than the latter. Thus, KASLS_R is comparable to KASLS even though only a small part (X_R) of the training set is used in the training and testing phases. We use KASLS_R in the following experiments.

D. Comparison with Other Class Imbalance Learning Methods

To demonstrate the performance of the ASLS loss in imbalanced classification, we report experimental results on the real world datasets along with other class imblance learning methods. Except for *G-mean*, we also use *F-measure* and AUC [37] to evaluate the classification results. For comparison, PSVM-DEC [23], SVM-DEC [19] and SVM-SDC [27] are selected as baselines, among which SVM-SDC involves a popular data pre-processing method named SMOTE [13]. Libsvm [38] is used to implement SVM-DEC and SVM-SDC. For hyper-parameters like regularization number ν, ramp coefficient r_+ and margin coefficient m_+ etc., we determine them by 5-fold cross-validation on the training set. We run three 10-fold cross-validation throughout the experiments. In addition to reporting the average results, we also perform paired t-test [1] comparing other class imbalance learning methods to the ASLS-based classifiers at the 5% significance level.

TABLE VI

COMPARISON OF NONLINEAR KERNEL CLASSIFICATION (MEAN(STD)). THE ASTERISK (*) DENOTES A SIGNIFICANT DIFFERENCE BETWEEN KASLS_R AND OTHER METHODS. THE BEST VALUES ARE IN BOLD. BELOW THE NAME OF DATASET ARE THE RATIO OF CLASSES AND THE REDUCED RATE η.

Dataset		G-mean(%)	F-Measure(%)	AUC(%)
Phoneme (1:2.41) ($\eta = 0.3$)	KASLS_R	**88.68(1.55)**	82.41(2.82)	94.45(1.28)
	PSVM-DEC	87.95(1.27)	81.71(1.94)	**94.82(0.86)**
	SVM-DEC	88.23(1.15)	**82.74(1.89)**	93.99(1.20)
	SVM-SDC	$86.10(1.47)*$	$79.63(1.61)*$	$92.51(0.93)*$
Vehicle (1:2.99) ($\eta = 0.7$)	KASLS_R	**83.78(4.60)**	**71.49(6.55)**	**90.65(3.51)**
	PSVM-DEC	81.70(5.02)	68.48(5.79)	89.86(2.94)
	SVM-DEC	81.85(4.31)	69.08(6.13)	90.46(3.11)
	SVM-SDC	81.56(4.23)	69.00(4.82)	90.13(3.13)
KC1 (1:5.47) ($\eta = 0.3$)	KASLS_R	**72.10(4.99)**	**46.05(6.23)**	80.10(4.02)
	PSVM-DEC	$68.38(4.34)*$	45.98(5.95)	**80.62(4.06)**
	SVM-DEC	71.43(3.98)	44.12(7.28)	78.40(4.23)
	SVM-SDC	$68.06(5.89)*$	$43.09(5.74)*$	$78.17(5.02)*$
Satimage (1:9.28) ($\eta = 0.3$)	KASLS_R	**91.87(1.08)**	**73.97(4.06)**	96.73(0.89)
	PSVM-DEC	91.15(1.67)	73.92(4.50)	**97.21(0.80)***
	SVM-DEC	$90.16(2.18)*$	$70.45(4.69)*$	96.88(0.84)
	SVM-SDC	$89.20(2.12)*$	$69.81(3.54)*$	$95.53(0.95)*$
Yeast (1:28.10) ($\eta = 0.7$)	KASLS_R	**81.69(9.48)**	**44.35(15.05)**	88.61(9.05)
	PSVM-DEC	$71.67(11.66)*$	35.59(19.08)	88.67(8.31)
	SVM-DEC	$75.82(9.40)*$	35.68(11.55)	**91.51(5.16)**
	SVM-SDC	$74.21(12.74)*$	38.41(10.73)	89.16(6.74)
Mammo. (1:42) ($\eta = 0.3$)	KASLS_R	**91.40(4.38)**	**67.72(9.12)**	94.61(3.25)
	PSVM-DEC	90.81(4.50)	$54.70(5.75)*$	**95.24(3.81)**
	SVM-DEC	$86.99(6.05)*$	$53.11(6.81)*$	94.02(3.48)
	SVM-SDC	$82.04(5.59)*$	$56.08(5.89)*$	91.91(4.62)

TABLE VII

THE AVERAGE SVS RATIOS OF KERNEL CLASSIFIERS OVER ALL THE DATASETS.

	KASLS_R	PSVM-DEC	SVM-DEC	SVM-SDC
Ratio	0.4654	1.0000	0.3274	0.3489

The average linear classification results are reported in Table V. LASLS generally performs better than the other class imbalance learning methods in terms of *G-mean* and *F-measure*. Especially on the Phoneme dataset, LASLS is statistically better at the 5% significance level when *F-measure* is used to evaluate the classification reuslts. On the Abalone dataset, PSVM is a little better. In all, there are no statistical differences in AUC among all the methods and all the datasets. From Table V, we know that LASLS has comparable or better performance compared with the other state-of-the-art linear imbalanced classification methods.

Table VI shows the average results of nonlinear kernel classification. Gaussian kernel is used in all the kernel classifiers and the reduce rates of KASLS_R are listed below the name of datasets. Like linear classification results, KASLS_R generally performs better than the other methods in terms of *G-mean* and *F-measure*, particularly on the highly imbalanced datasets like Satimage, Yeast and Mammography, where KASLS_R is statistically better in most of the cases. The performance of AUC of all the methods are comparable on all the datasets. Table VII shows the average SVs ratios of the four kernel classifiers over all the datasets. We easily

find that PSVM-DEC loses the sparseness, while $KASLS_R$ possesses the property of sparseness although its sparseness is not comparable to SVMs. In all, $KASLS_R$ is generally an effective imbalanced classification method.

V. Conclusion

We propose an asymmetric stagewise least square loss function for imbalanced classification in this paper. Its ramp and margin coefficients are in command of the penalty and margin of the ASLS loss respectively which make the ASLS loss be more flexible. Experimental results show the effectiveness of asymmetric margins in recovering the decision boundary bias in imbalanced classification. Besides, compared with the LS loss, it naturally inherits the advantages of the SLS loss like higher generalization, better robustness, and better sparseness [7]. In order to further improve the training and testing speed, we also develop the $KASLS_R$ algorithm which has comparable performance to KASLS. Overally, experimental results confirm the effectiveness of the ASLS loss in imbalanced classification. In the future, we will further study the optimization and sparseness problems of the ASLS loss and compare the ASLS loss to the emerging C-loss function [8] and cost-free leaning [39], [40].

References

[1] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.

[2] P. L. Bartlett, M. I. Jordan and J. D. McAuliffe, "Convexity, classification, and risk bounds", *Journal of the American Statistical Association*, Vol. 101, No. 473, pp. 138–156, Mar. 2006.

[3] T. Zhang, "Statistical behavior and consistency of classification methods based on convex risk minimization", *Annals of Statistics*, Vol. 32, No. 1, pp. 56–85, Feb. 2004.

[4] X. T. Shen, G. C. Tseng, X. G. Zhang and W. H. Wong, "On ψ-learning", *Journal of the American Statistical Association*, Vol. 98, No. 463, pp. 724–734, Sep. 2003.

[5] R. Collobert, F. Sinz, J. Weston and L. Bottou, "Trading convexity for scalability", *Proc. of Int'l Conf. on Machine learning (ICML)*, 2006.

[6] Bengio, Yoshua, "Learning deep architectures for AI", *Foundations and Trends® in Machine Learning*, Vol. 2, No. 1, pp. 1–127, 2009.

[7] S.-H. Yang and B.-G. Hu, "A stagewise least square loss function for classification", *Proc. of SIAM Int'l Conf. on Data Mining (SDM)*, 2008.

[8] A. Singh, R. Pokharel and J. Principe, "The C-loss function for pattern classification", *Patten Recognition*, Vol. 47, No. 7, pp. 441-453, Jan. 2014.

[9] H. B. He and E. A. Garcia, "Learning from imbalanced data", *IEEE Trans. on Knowledge and Data Engineering*, Vol. 21, No. 9, pp. 1263–1284, Sep. 2009.

[10] H.B. He and Y.Q. Ma, *Imbalanced Learning: Foundations, Algorithms, and Applications*, Wiley, 2013.

[11] P. Domingos, "MetaCost: a general method for making classifiers cost-sensitive", *Proc. of ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (SIGKDD)*, 1999.

[12] M. Kubat and S. Matwin, "Addressing the curse of imbalanced training sets: one-sided selection", *Proc. of Int'l Conf. on Machine Learning (ICML)*, 1997.

[13] N. V. Chawla, K. W. Bowyer, T. E. Moore and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique", *Journal of Artificial Intelligence Research*, Vol. 16, pp. 321–357, June 2002.

[14] C. Elkan, "The foundations of cost-sensitive learning", *Proc. of Int'l Joint Conf. on Artificial Intelligence (IJCAI)*, 2001.

[15] C. Scott, "Calibrated asymmetric surrogate losses", *Electronic J. of Statistics*, Vol. 6, pp. 958–992, 2012.

[16] V. Vapnik, *The Nature of Statistical Learning Theory (1st Edition)*, Springer-Verlag New York, 1995.

[17] S. Ertekin, J. Huang, L. Bottou and L. Giles, "Learning on the border: active learning in imbalanced data classification", *Proc. of ACM Conf. on Information and Knowledge Management (CIKM)*, 2007.

[18] Y. C. Tang, Y. Q. Zhang, N. V. Chawla and S. Krasser, "SVMs modeling for highly imbalanced classification", *IEEE Trans. on Systems, Man, and Cybernetics, Part B: Cybernetics*, Vol. 39, No. 1, pp. 281–288, Feb. 2009.

[19] K. Veropoulos, C. Campbell and N. Cristianini, "Controlling the sensitivity of support vector machines", *Proc. of Int'l Joint Conf. on Artificial Intelligence (IJCAI)*, 1999.

[20] C.-Y. Yang, J.-S. Yang and J.-J. Wang, "Margin calibration in SVM class-imbalanced learning", *Neurocomputing*, Vol. 73, No. 1, pp. 397–411, Aug. 2009.

[21] R. Batuwita and V. Palade, "FSVM-CIL: fuzzy support vector machines for class imbalance learning", *IEEE Trans. on Fuzzy Systems*, Vol. 18, No. 3, pp. 558–571, June 2010.

[22] Y.-H. Liu and Y.-T. Chen, "Face recognition using total margin-based adaptive fuzzy support vector machines", *IEEE Trans. on Neural Networks*, Vol. 18, No. 1, pp. 178–192, Jan. 2007.

[23] G. M. Fung and O. L. Mangasarian, "Multicategory proximal support vector machine classifiers", *Machine Learning*, Vol. 59, No. 1, pp. 77–97, 2005.

[24] G. Wu and E. Y. Chang, "KBA: kernel boundary alignment considering imbalanced data distribution", *IEEE Trans. on Knowledge and Data Engineering*, Vol. 17, No. 6, pp. 786–795, June 2005.

[25] P. Williams, S. Li, J. F. Feng and S. Wu, "A geometrical method to improve performance of the support vector machine", *IEEE Trans. on Neural Networks*, Vol. 18, No. 3, pp. 942–947, May 2007.

[26] J. Kandola and J. Shawe-Taylor, "Refining kernels for regression and uneven classification problems", *Proc. of Int'l Workshop on Artificial Intelligence and Statistics*, 2003.

[27] R. Akbani, S. Kwek and N. Japkowicz, "Applying support vector machines to imbalanced datasets", *Proc. of European Conf. on Machine Learning (ECML)*, 2004.

[28] P. Li, K. L. Chan and W. Fang, "Hybrid kernel machine ensemble for imbalanced data sets", *Proc. of Int'l Conf. on Pattern Recognition (ICPR)*, 2006.

[29] B. Raskutti and A. Kowalczyk, "Extreme re-balancing for SVMs: a case study", ACM SIGKDD Explorations Newsletter, Vol. 6, No. 1, pp. 60–69, 2004.

[30] S. Wu, K. Lin, H. Chien, C. Chen and M. Chen, "On generalizable low false-positive learning using asymmetric support vector machines", *IEEE Trans. on Knowledge and Data Engineering*, Vol. 25, No. 5, pp. 1083–1096, May 2013.

[31] T. Imam, K. M. Ting and J. Kamruzzaman, "z-SVM: an SVM for improved classification of imbalanced data", *Advances in Artificial Intelligence*, 2006.

[32] Y.-J. Lee and O. L. Mangasarian, "RSVM: Reduced support vector machines", *Proc. of the SIAM Int'l Conf. on Data Mining (SDM)*, 2001.

[33] T. V. Gestel, J. A. K. Suykens, B. Baesens, S. Viaene, J. Vanthienen, G. Dedene, B. D. Moor and J. Vandewalle, "Benchmarking least squares support vector machine classifiers", *Machine Learning*, Vol. 54, No. 1, pp. 5–32, 2004.

[34] P. B. Nair, A. Choudhury and A. J. Keane, "Some greedy learning algorithms for sparse regression and classification with mercer kernels", *Journal of Machine Learning Research*, Vol. 3, pp. 781–801, Dec. 2002.

[35] H. Masnadi-Shirazi, V. Mahadevan and N. Vasconcelos, "On the design of robust classifiers for computer vision", *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[36] O. L. Mangasarian, "Generalized support vector machines", *Advances in Large Margin Classifiers*, pp. 135–146, MIT Press, 2000.

[37] T. Fawcett, "ROC graphs: Notes and practical considerations for researchers", Technical Report HPL-2003–4, HP Laboratories, Palo Alto, CA, USA, 2003.

[38] C. C. Chang, C. J. Lin, "LIBSVM: a library for support vector machines", *ACM Trans. on Intelligent Systems and Technology*, Vol. 2, No. 3, pp. 1–27, April 2011.

[39] X. W. Zhang and B.-G. Hu, "A new strategy of cost-free learning in the class imbalance problem", *IEEE Trans. on Knowledge and Data Engineering*, Vol. PP, No. 99, Mar 2014 (accept).

[40] G. B. Xu and B.-G. Hu, "Cost-free learning for support vector machines with a reject option", *IEEE Int'l Conf. on Data Mining Workshops (ICDMW)*, 2013.