Pattern Recognition Letters 80 (2016) 107–112

Contents lists available at ScienceDirect

Pattern Recognition Letters

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

Cost-Sensitive Large margin Distribution Machine for classification of imbalanced data☆

Fanyong Cheng[a,b,*], Jing Zhang[a], Cuihong Wen[a]

[a] College of Electrical and Information Engineering, Hunan University, Changsha 410082, China
[b] Fujian Provincial Key Laboratory of Information Processing and Intelligent Control, Department of Computer Science, Minjiang University, Fuzhou 350121, China

ARTICLE INFO

Article history:
Received 9 November 2015
Available online 21 June 2016

Keywords:
Minimum margin
Margin distribution
Imbalanced training data
Cost-sensitive learning
Balanced detection rate

ABSTRACT

This paper proposes a new method to design a balanced classifier on imbalanced training data based on margin distribution theory. Recently, Large margin Distribution Machine (LDM) is put forward and it obtains superior classification performance compared with Support Vector Machine (SVM) and many state-of-the-art methods. However, one of the deficiencies of LDM is that it easily leads to the lower detection rate of the minority class than that of the majority class on imbalanced data which contradicts to the needs of high detection rate of the minority class in the real application. In this paper, Cost-Sensitive Large margin Distribution Machine (CS-LDM) is brought forward to improve the detection rate of the minority class by introducing cost-sensitive margin mean and cost-sensitive penalty. Theoretical and experimental results show that CS-LDM can gradually improve the detection rate of the minority class with the increasing of the cost parameter and obtain a balanced classifier when the cost parameter increases to a certain value. CS-LDM is superior to some popular cost-sensitive methods and can be used in many applications.

1. Introduction

As one of the most popular classification algorithms, Support Vector Machine (SVM) can be seen as a learning approach trying to maximize the minimum margin on the training data. It acquires a good theoretical foundation for the generalization performance from margin theory [8]. What is noteworthy for the margin theory is that it not only plays a guidance role for SVM, but also has been extended to interpret the strong generalization performance of many other learning approaches [4]. However, according to margin theory, Breiman [2] designed Arc-gv, which is able to maximize the minimum margin but has a weak generalization performance. Therefore, he doubted about the margin theory, and this almost sentenced margin theory to death. After ignoring the issue for years, Reyzin and Schapire [3] found that although Arc-gv directly maximizes the minimum margin, it suffers from a weak margin distribution. Thus, they conjectured that the margin distribution is more crucial than the minimum margin to the generalization performance. Gao and Zhou [1] proved this conjecture and revealed that rather than simply considering a single margin, margin distribution is really significant for the generalization performance. Consequently, Large margin Distribution Machine (LDM) which maximizes the margin mean and minimizes the margin variance is proposed by Zhang and Zhou [6,7]. Though LDM is superior to SVM and many state-of-the-art methods whether on theoretical results or many experimental results [7], it is not satisfactory when the training data is imbalanced.

We find that LDM generally makes the separator incline to the minority class to obtain a larger margin mean, and this leads to the problem that the minority class examples are more easily misclassified than the majority class examples. Generally speaking, real-world data sets are composed of the majority class examples with only a small percentage of the minority class examples. The minority class is usually more interesting or costly. For example, the mammography data set which might contain 98% normal examples and 2% abnormal examples. A simple default strategy of guessing the majority class would give a predictive accuracy of 98%, but miss all abnormal examples. However, the nature of the application requires a high detection rate of the minority class allowing a certain error rate in the majority class. Therefore, Cost-Sensitive Large margin Distribution Machine (CS-LDM) is proposed to address the imbalanced detection rate between two classes by adjusting margin weight for each class in the margin mean. CS-LDM increases

☆ "This paper has been recommended Ananda S Chowdhury."

* Corresponding author at: College of Electrical and Information Engineering, Hunan University, Changsha 410082, China. Tel.: +86 18673197062.
E-mail address: b12090031@hnu.edu.cn (F. Cheng).

http://dx.doi.org/10.1016/j.patrec.2016.06.009
0167-8655/

the margin weights of the minority class examples in the margin mean to force the separator to move towards the majority class examples, while increasing the misclassification penalty of the minority class. It is trained and tested on some UCI data sets, and experimental results show that CS-LDM can effectively improve the detection rate of the minority class to achieve a balanced detection rate, while has a strong generalization performance.

2. Related works

To deal with the imbalanced detection rate, various techniques have been proposed. These techniques can be mainly divided into three basic types: data-preprocessing, algorithmic approach, and boosting approach. The first type tries to increase the number of the minority examples (over-sampling) [22] or decrease the number of the majority class examples (under-sampling) [23] in different ways. Batista [24] combined these two methods to acquire good detection results for the minority class. The second type adjusts the cost of error or decision threshold in classification on the imbalanced data and tries to control the detection rate of the minority class. For example, Elkan [17], Seiffert et al. [25] decreased the proportion of the majority examples in training data set to train an optimal classification decision, and many methods were proposed to improve the prediction performance by adjusting the weight (cost) for each class [5,18]. Huang et al. [19] proposed the BMPM algorithm to build up biased classifier. The third type uses the cost of misclassifications to update the training distribution on successive boosting rounds [20,21,26]. In addition, there are some other combined ways to address this problem. For example, RUSBoost [25] was proposed by combining data sampling and fast boosting. Although these algorithms mentioned above obtained good results, there is a lack of consideration of the cost of the margin distribution which is crucial to generalization performance. In the following, LDM and CS-LDM will be introduced in detail and compared.

3. Large margin Distribution Machine

We denote sample space by $\boldsymbol{x} \in R^d$ (d is the dimension of each example or feature), and the label set by $y = \{+1, -1\}$. A training set of size m $S = \{(\boldsymbol{x_1}, y_1), (\boldsymbol{x_2}, y_2), \ldots (\boldsymbol{x_m}, y_m)\}$ is drawn identically and independently according to an unknown distribution Γ over $\boldsymbol{x} \times y$. The final goal is to learn a linear function $f(\boldsymbol{x}) = \boldsymbol{\omega}^T \phi(\boldsymbol{x})$ with strong generalization performance to predict an unlabeled example, where $\boldsymbol{\omega}$ is a weight vector and $\phi(\boldsymbol{x})$ is a feature transform of x induced by a kernel k, i.e., $k(\boldsymbol{x_i}, \boldsymbol{x_j}) = \phi(\boldsymbol{x_i})^T \phi(\boldsymbol{x_j})$. In the light of [8–11], the margin of example $(\boldsymbol{x_i}, y_i)$ is formulated as

$$\gamma_i = y_i \boldsymbol{\omega}^T \phi(\boldsymbol{x_i}), \forall i = 1, \ldots, m.$$

The margin mean is formulated as

$$\bar{\gamma} = \sum_{i=1}^{m} y_i \boldsymbol{\omega}^T \phi(\boldsymbol{x_i}) = \frac{1}{m} (X\boldsymbol{y})^T \boldsymbol{\omega},$$

where $X = [\phi(\boldsymbol{x_1}), \ldots, \phi(\boldsymbol{x_m})]$ represents a matrix ($\phi(\boldsymbol{x_i})$ is the ith column of the matrix) and $\boldsymbol{y} = [y_1, \ldots, y_m]^T$ is denoted as a vector. The margin variance is formulated as

$$\begin{aligned} \hat{\gamma} &= \frac{1}{m^2} \sum_{i=1}^{m} \sum_{j=1}^{m} (y_i \boldsymbol{\omega}^T \phi(\boldsymbol{x_i}) - y_j \boldsymbol{\omega}^T \phi(\boldsymbol{x_j})^2 \\ &= \frac{2}{m^2} (m \boldsymbol{\omega}^T X X^T \boldsymbol{\omega} - \boldsymbol{\omega}^T X \boldsymbol{y} \boldsymbol{y}^T X^T \boldsymbol{\omega}) \end{aligned} \quad (1)$$

LDM maximizes the margin mean and minimizes the margin variance simultaneously. The optimal object function of LDM is formulated as

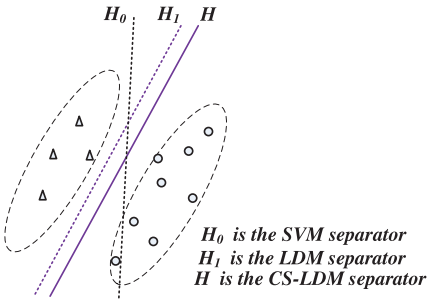

Fig. 1. Simple illustration of the separators with different algorithms. Triangles represent positive examples; circles represent negative examples.

$$\min_{w,\xi} \frac{1}{2} \boldsymbol{\omega}^T \boldsymbol{\omega} + \lambda_1 \hat{\gamma} - \lambda_2 \bar{\gamma} + C \sum_{i=1}^{m} \xi_i.$$

$$s.t. \quad y_i \boldsymbol{\omega}^T \phi(\boldsymbol{x_i}) \geq 1 - \xi_i,$$
$$\xi_i \geq 0, i = 1, \ldots, m.$$

Where $C \geq 0$, $\lambda_1 \geq 0$ and $\lambda_2 \geq 0$ are real trade-off parameters for the training error, the margin variance, the margin mean and the model complexity.

4. Cost-Sensitive Large margin Distribution Machine

The idea is that we increase the margin weight of the minority class in the margin mean and the misclassification penalty of the minority class to address the problem that the separator of LDM (H_1 in Fig. 1) inclines to the minority class to get a larger margin mean as shown in Fig. 1.

4.1. Formulation

Considering that the positive training example size is m_+, and the negative training example size is m_- ($m_+ < m_-$), the separator of LDM (H_1 in Fig. 1) inclines to positive examples to increase the margin mean as shown in Fig. 1. To resist the trend of LDM, we introduce cost-sensitive learning method. We define the cost of margin as

$$\theta_i = \begin{cases} (m_-/m_+)^\rho & i \in I_+, \\ (m_+/m_-)^\rho & i \in I_-, \end{cases} \quad (2)$$

where $\rho \geq 0$ is a real cost parameter, $I_+ \equiv \{i | y_i = 1\}$, and $I_- \equiv \{i | y_i = -1\}$. We can find that the cost of the positive class (the minority class) is more than or equal to that of the negative class (the majority class) for all possible value of ρ. The cost-sensitive margin mean is formulated as

$$\bar{\bar{\gamma}} = \sum_{i=1}^{m} \theta_i y_i \boldsymbol{\omega}^T \phi(\boldsymbol{x_i}) = \frac{1}{m} (XD\boldsymbol{y})^T \boldsymbol{\omega} \quad (3)$$

where D is a diagonal matrix with diagonal elements $\theta_1, \ldots, \theta_m$. For simplicity, the separable case without outliers is considered. If ρ increases, the cost-sensitive margin mean of the minority class will increase and the cost-sensitive margin mean of the majority class will decrease. This induces the CS-LDM separator (H in Fig. 1) incline to the majority class examples (circles) to get a larger cost-sensitive margin mean as shown in Fig. 1.

To prevent the separator towards one class immoderately, the margin variance is also introduced into CS-LDM. Besides that, Fig. 2 shows a more complicated case where there are outliers or noisy examples. If we only optimize the minimum margin, the separator may be dominated by the outlier or noisy example. However, if

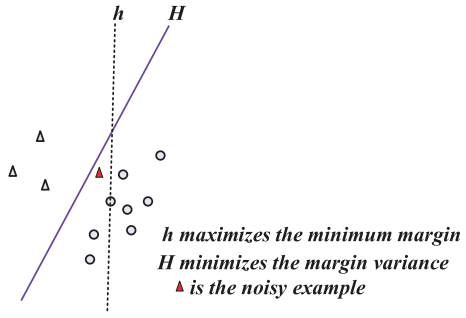

Fig. 2. Simple illustration of the separators existing outlier or noisy instance. Red triangle represents outlier or noisy data.

we try to minimize the margin variance instead, the unfavorable effect of the outliers or noisy examples will diminish automatically. Taken together, the objective function of CS-LDM is expressed as

$$\min_{w,\xi} \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} + \lambda_1\hat{\gamma} - \lambda_2\bar{\bar{\gamma}} + C\sum_{i=1}^{m}\theta_i\xi_i. \tag{4}$$
$$s.t. \quad y_i\boldsymbol{\omega}^T\phi(\boldsymbol{x_i}) \geq 1-\xi_i,$$
$$\xi_i \geq 0, i = 1,\ldots,m.$$

Where $C \geq 0$, $\lambda_1 \geq 0$ and $\lambda_2 \geq 0$ are trade-off parameters. We can find that when m_+ is equal to m_- or ρ is equal to 0, D becomes the identity matrix I and CS-LDM is degenerated to LDM.

4.2. Optimization

In this part, a dual coordinate descent method is presented to solve kernel CS-LDM, and then an average stochastic descent (ASGD) method is adopted to solve large scale linear CS-LDM.

4.2.1. Kernel CS-LDM

By substituting (1), (3), and (4) turns to the following quadratic programming problem,

$$\min_{w,\xi} \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} + \frac{2\lambda_1}{m^2}(m\boldsymbol{\omega}^TXX^T\boldsymbol{\omega} - \boldsymbol{\omega}^TX\mathbf{y}\mathbf{y}^TX^T\boldsymbol{\omega})$$
$$-\frac{\lambda_2}{m}(XD\mathbf{y})^T\boldsymbol{\omega} + C\sum_{i=1}^{m}\theta_i\xi_i. \tag{5}$$
$$s.t. \quad y_i\boldsymbol{\omega}^T\phi(\boldsymbol{x_i}) \geq 1-\xi_i,$$
$$\xi_i \geq 0, i = 1,\ldots,m.$$

Generally, (5) is unrealistic due to the high or infinite dimensionality of $\phi(.)$. Fortunately, in light of Zhang and Zhou [7], the optimal solution $\boldsymbol{\omega}$ can be represented by $\phi(\boldsymbol{x_i})$

$$\boldsymbol{\omega} = \sum_{i=1}^{m}\alpha_i\phi(\boldsymbol{x_i}) = X\boldsymbol{\alpha}, \tag{6}$$

where $\boldsymbol{\alpha} = [\alpha_1,\ldots,\alpha_m]$. According to (6), we have

$$X^T\boldsymbol{\omega} = X^TX\boldsymbol{\alpha} = G\boldsymbol{\alpha},$$
$$\boldsymbol{\omega}^T\boldsymbol{\omega} = \boldsymbol{\alpha}^TX^TX\boldsymbol{\alpha} = \boldsymbol{\alpha}^TG\boldsymbol{\alpha},$$

where $G = X^TX$ is the kernel matrix. The ith column of G is represented by G_i, then (5) can be expressed as

$$\min_{\alpha,\xi} \frac{1}{2}\boldsymbol{\alpha}^TQ\boldsymbol{\alpha} + P^T\boldsymbol{\alpha} + C\sum_{i=1}^{m}\theta_i\xi_i \tag{7}$$
$$s.t. \quad y_i\boldsymbol{\alpha}^TG_i \geq 1-\xi_i$$
$$\xi_i \geq 0, i = 1,\ldots,m,$$

where $Q = 4\lambda_1(mG^TG - (G\mathbf{y})(G\mathbf{y})^T)/m^2 + G$ is a symmetric matrix, and $P = -\lambda_2GDy/m$. Through the first and the second constraints introducing the lagrange multipliers $\boldsymbol{\mu} = [\mu_1,\ldots,\mu_m]^T$ and $\boldsymbol{\eta} = [\eta_1,\ldots,\eta_m]^T$ respectively, the lagrangian function of (7) is formulized as

$$L(\boldsymbol{\alpha},\boldsymbol{\xi},\boldsymbol{\mu},\boldsymbol{\eta}) = \frac{1}{2}\boldsymbol{\alpha}^TQ\boldsymbol{\alpha} + P^T\boldsymbol{\alpha} + C\sum_{i=1}^{m}\theta_i\xi_i$$
$$-\sum_{i=1}^{m}\mu_i(y_i\boldsymbol{\alpha}^TG_i - 1 + \xi_i) - \sum_{i=1}^{m}\eta_i\xi_i \tag{8}$$

According to KKT conditions, we have

$$\frac{\partial L}{\partial \boldsymbol{\alpha}} = Q\boldsymbol{\alpha} + P - \sum_{i=1}^{m}\mu_iy_iG_i = 0 \tag{9}$$

$$\frac{\partial L}{\partial \xi_i} = C\theta_i - \mu_i - \eta_i = 0, i = 1,\ldots,m, \tag{10}$$

$$\mu_i(y_i\boldsymbol{\alpha}^TG_i - 1 + \xi_i) = 0, i = 1,\ldots,m, \tag{11}$$

$$\eta_i\xi_i = 0, \mu_i \geq 0, \eta_i \geq 0, i = 1,\ldots,m. \tag{12}$$

By substituting (9) and (10) into (8), the lagrangian dual function of (7) can be deduced as

$$\min_{\mu} f(\boldsymbol{\mu}) = \frac{1}{2}\boldsymbol{\mu}^TH\boldsymbol{\mu} + \left(\frac{\lambda_2}{m}HD\boldsymbol{e} - \boldsymbol{e}\right)^T\boldsymbol{\mu} \tag{13}$$
$$s.t. \quad 0 \leq \mu_i \leq C\theta_i, i = 1,\ldots,m,$$

where $H = YGQ^{-1}GY$ is a symmetric matrix, Q^{-1} is the inverse matrix of Q, Y is a $m \times m$ diagonal matrix with diagonal elements $y_1,\ldots,y_m$, and $\boldsymbol{e}$ represents the all one vector. The dual coordinate descent method can efficiently solve (13) [12]. In a dual coordinate descent method [13], one of the variables is chosen to minimize while the other variables are treated as constants at each iteration, and a closed-form solution can be derived at each iteration. Specifically, to minimize μ_i by treating the other $\mu_{j\neq i}$ as constants, we need to solve the following subproblem,

$$\min_{t} f(\boldsymbol{\mu} + t\boldsymbol{e}_i) \tag{14}$$
$$s.t. \quad 0 \leq \mu_i + t \leq C\theta_i,$$

where $\boldsymbol{e}_i$ denotes the vector whose ith component is 1 and the others are all 0. Let $H = [h_{ij}]_{i,j=1,\ldots,m}$, we deduce the following equation

$$f(\boldsymbol{\mu} + t\boldsymbol{e}_i) = \frac{1}{2}h_{ii}t^2 + [\nabla f(\boldsymbol{\mu})]_it + f(\boldsymbol{\mu})$$
$$\left([\nabla f(\boldsymbol{\mu})]_i = \boldsymbol{e_i}^TYG\left[Q^{-1}GY\left(\frac{\lambda_2}{m}D\boldsymbol{e} + \boldsymbol{\mu}\right)\right] - 1\right), \tag{15}$$

where $[\nabla f(\boldsymbol{\mu})]_i$ is the ith component of the gradient $[\nabla f(\boldsymbol{\mu})]$. Note that $f(\boldsymbol{\mu})$ is a constant about t in (15), thus it can be removed. We can easily find that (14) has an optimum at $t = 0$ if and only if

$$[\nabla^P f(\boldsymbol{\mu})]_i = 0, \tag{16}$$

where $[\nabla^P f(\boldsymbol{\mu})]_i$ [13] is represented by

$$[\nabla^P f(\boldsymbol{\mu})]_i = \begin{cases} [\nabla f(\boldsymbol{\mu})]_i & \text{if } 0 < \mu_i < C\theta_i, \\ min(0,[\nabla f(\boldsymbol{\mu})]_i) & \text{if } \mu_i = 0, \\ max(0,[\nabla f(\boldsymbol{\mu})]_i) & \text{if } \mu_i = C\theta_i. \end{cases}$$

If (16) holds, we move to update the next element of $\boldsymbol{\mu}$ without updating μ_i. Otherwise, we must find the optimal solution of (14). In view of $f(\boldsymbol{\mu} + t\boldsymbol{e}_i)$ is a simple quadratic function about t, and the box constraint $0 \leq \mu_i + t \leq C\theta_i$. If $h_{ii} \neq 0$, a closed-form solution of (14) can be deduced as,

$$\mu_i^{new} = min\left(max\left(\mu_i - \frac{[\nabla f(\boldsymbol{\mu})]_i}{h_{ii}}, 0\right), C\theta_i\right). \tag{17}$$

If $h_{ii} = 0$, we set $1/h_{ii} = \infty$ in (17). To predict unlabeled example, in accordance to (9), we can easily deduce the coefficient $\boldsymbol{\alpha}$ from the optimal $\boldsymbol{\mu}^*$ as

$$\boldsymbol{\alpha} = Q^{-1}GY\left(\frac{\lambda_2}{m}De + \boldsymbol{\mu}^*\right).$$

The pseudo-code of kernel CS-LDM is summarized in Algorithm 1.

Algorithm 1 Kernel CS-LDM.

Input: Data set $X, \theta_i, \lambda_1, \lambda_2, C$;
Output: $\boldsymbol{\alpha}$
Initialize $\mu = 0, \boldsymbol{\alpha} = \frac{\lambda_2}{m}Q^{-1}GD\boldsymbol{y}, A = Q^{-1}GY, h_{ii} = \boldsymbol{e}_i^T YGQ^{-1}GY\boldsymbol{e}_i$;
while $\boldsymbol{\mu}$ not converge **do**
for $i = 1, \ldots, m$ **do**
$[\nabla f(\boldsymbol{\mu})]_i \leftarrow \boldsymbol{e}_i^T YG\boldsymbol{\alpha} - 1$;

$$[\nabla^P f(\boldsymbol{\mu})]_i = \begin{cases} [\nabla f(\boldsymbol{\mu})]_i & \text{if } 0 < \mu_i < C\theta_i \\ min(0, [\nabla f(\boldsymbol{\mu})]_i) & \text{if } \mu_i = 0 \\ max(0, [\nabla f(\boldsymbol{\mu})]_i) & \text{if } \mu_i = C\theta_i \end{cases}$$

if $|[\nabla^P f(\boldsymbol{\mu})]_i| \neq 0$ **then**
$\mu_i^{old} \leftarrow \mu_i$
$\mu_i \leftarrow min(max(\mu_i - \frac{[\nabla f(\boldsymbol{\mu})]_i}{h_{ii}}, 0), C\theta_i)$;
$\boldsymbol{\alpha} \leftarrow \boldsymbol{\alpha} + (\mu_i - \mu_i^{old})A\boldsymbol{e}_i$;
end if
end for
end while

Hence for unlabeled example $\boldsymbol{z}$, its label can be calculated by

$$sgn\left(\boldsymbol{\omega}^T\phi(\boldsymbol{z})\right) = sgn\left(\sum_{i=1}^{m}\alpha_i k(\boldsymbol{x_i}, \boldsymbol{z})\right).$$

4.2.2. Large scale linear CS-LDM

In the previous section, kernel CS-LDM can be efficiently solved by the Algorithm 1. Nevertheless, the algorithm takes the time complexity of $O(m^2)$ to compute the kernel matrix, which might be too large to compute for large scale problems. To improve the learning speed, a fast linear CS-LDM for large scale problems is presented by adopting the average stochastic gradient descent (ASGD) method [14]. For linear CS-LDM, (5) can be reshaped as

$$\begin{aligned}\min_{\boldsymbol{\omega}} g(\boldsymbol{\omega}) = {} & \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} + \frac{2\lambda_1}{m^2}(m\boldsymbol{\omega}^T XX^T\boldsymbol{\omega} - \boldsymbol{\omega}^T X\boldsymbol{y}\boldsymbol{y}^T X^T\boldsymbol{\omega}) \\ & - \frac{\lambda_2}{m}(XD\boldsymbol{y})^T\boldsymbol{\omega} + C\sum_{i=1}^{m}\theta_i max(0, 1 - y_i\boldsymbol{\omega}^T\boldsymbol{x_i}) \end{aligned} \quad (18)$$

where $X = [\boldsymbol{x_1}, \ldots, \boldsymbol{x_m}]$ is a matrix. For large scale problems, the gradient of (18) is computationally expensive because its computation incorporates all the training examples. Stochastic gradient descent (SGD) samples a subset of the training examples to compute an unbiased estimation of the gradient instead of the actual gradient [15]. Therefore, SGD has a fast learning speed and achieved good performance for large scale linear SVM problems [16]. In accordance with the proof way of 2 in [7], we can deduce the following theorem which gives a method to obtain an unbiased estimation of $\nabla g(\boldsymbol{\omega})$.

Theorem 1. *If two examples $(\boldsymbol{x_i}, y_i)$ and $(\boldsymbol{x_j}, y_j)$ are randomly selected from the training set, then*

$$\begin{aligned}\nabla g(\boldsymbol{\omega}, \boldsymbol{x_i}, \boldsymbol{x_j}) = {} & 4\lambda_1\boldsymbol{x_i}\boldsymbol{x_i}^T\boldsymbol{\omega} - 4\lambda_1 y_i\boldsymbol{x_i}y_j\boldsymbol{x_j}^T\boldsymbol{\omega} + \boldsymbol{\omega} \\ & - \lambda_2\theta_i y_i\boldsymbol{x_i} - mC\begin{cases}\theta_i y_i\boldsymbol{x_i} & i \in I, \\ 0 & otherwise,\end{cases}\end{aligned} \quad (19)$$

where $I \equiv \{i | y_i\boldsymbol{\omega}^T\boldsymbol{x_i} < 1\}$ is an unbiased estimation of $\nabla g(\boldsymbol{\omega})$.

According to Theorem 1, the update method of SGD can be formed as

$$\boldsymbol{\omega}_{t+1} = \boldsymbol{\omega}_t - \tau_t\nabla g(\boldsymbol{\omega}, \boldsymbol{x_i}, \boldsymbol{x_j}), \quad (20)$$

where τ_t is an appropriate step-size parameter at the tth iteration. To improve the robustness, besides executing the norm stochastic gradient update method (20), we also calculate

$$\bar{\boldsymbol{\omega}}_t = \frac{1}{t - t_0}\sum_{i=t_0+1}^{t}\boldsymbol{\omega}_i,$$

where t_0 is the starting iteration number of the average process. This average can be calculated efficiently using a recursive formula:

$$\bar{\boldsymbol{\omega}}_{t+1} = \bar{\boldsymbol{\omega}}_t + \nu_t(\boldsymbol{\omega}_{t+1} - \bar{\boldsymbol{\omega}}_t),$$

where $\nu_t = 1/(t + 1 - t_0)$.

The pseudo-code of large scale linear CS-LDM is summarized in Algorithm 2.

Algorithm 2 Large scale linear CS-LDM.

Input: Data set $X, \theta_i, \lambda_1, \lambda_2, C$;
Output: $\bar{\boldsymbol{\omega}}$
Initialize $\boldsymbol{\omega} = 0, \bar{\boldsymbol{\omega}} = 0, T = 5$;
for $t = 1, \ldots, Tm$ **do**
Randomly sample two training examples $(\boldsymbol{x_i}, y_i)$ and $(\boldsymbol{x_j}, y_j)$;
Compute $\nabla g(\boldsymbol{\omega}, \boldsymbol{x_i}, \boldsymbol{x_j})$ as in(19);
$\boldsymbol{\omega} \leftarrow \boldsymbol{\omega} - \tau_t\nabla g(\boldsymbol{\omega}, \boldsymbol{x_i}, \boldsymbol{x_j})$;
$\bar{\boldsymbol{\omega}} \leftarrow \bar{\boldsymbol{\omega}} + \nu_t(\boldsymbol{\omega} - \bar{\boldsymbol{\omega}})$;
end for

Hence for unlabeled example $\boldsymbol{z}$, its label can be estimated by $sgn(\bar{\boldsymbol{\omega}}^T\boldsymbol{z})$.

4.3. Analysis

The idea is that we try to decrease the number of positive examples whose μ_i is greater than or equal to $C\theta_i$ to improve the detection rate of the positive class. If $0 \le \mu_i < C\theta_i$ in (10), then $\eta_i > 0$, we can have $\xi_i = 0$ according to (12). Furthermore, we can have $y_i\boldsymbol{\omega}^T\phi(\boldsymbol{x_i}) = y_i\boldsymbol{\alpha}^T G_i \ge 1 - \xi_i = 1$. Namely, $(\boldsymbol{x_i}, y_i)$ will always be classified correctly. If $\mu_i = C\theta_i$ in (10), then $\eta_i = 0$, we can have $\xi_i \ge 0$ according to (12), namely, $(\boldsymbol{x_i}, y_i)$ may be misclassified. Therefore, we can have

$$E[R(\boldsymbol{\mu})] \le E\left[\frac{|I_2|}{m}\right] \quad (21)$$

where $\boldsymbol{\mu}$ denotes the optimal solution of (13), $E[R(\boldsymbol{\mu})]$ is the expected frequency of error over random samples of size m, $I_2 \equiv \{i | \mu_i = C\theta_i\}$, and $|I_2|$ is the number of elements in I_2. Intuitively, to improve the positive detection rate, $\mu_i (i \in I_+)$ should be as small as possible.

For CS-LDM, the closed-form optimal solutions of (13) without restraints can be obtained by the following formulas.

$$\begin{aligned}\frac{\partial f}{\partial \boldsymbol{\mu}} &= H\boldsymbol{\mu} + \frac{\lambda_2}{m}HD\boldsymbol{e} - \boldsymbol{e} = 0, \\ \boldsymbol{\mu} &= H^{-1}\boldsymbol{e} - \frac{\lambda_2}{m}D\boldsymbol{e}, \mu_i = e_i^T H^{-1}\boldsymbol{e} - \frac{\lambda_2}{m}\theta_i.\end{aligned} \quad (22)$$

Similarly, for LDM, we can obtain

$$\mu_i^l = e_i^T H^{-1}\boldsymbol{e} - \frac{\lambda_2}{m}, \quad (23)$$

where μ_i^l is the optimal solution of LDM. For a positive example $(\boldsymbol{x_i}, y_i)$ $(i \in I_+)$, θ_i is greater than 1 when $\rho > 0$. In the light of (22) and (23), we can easily determine $\mu_i < \mu_i^l$. Furthermore, $C\theta_i$

Table 1
Characteristics of experimental data sets.

Scale	Dataset	#instance	#feature	#minority	(#maj/#min)
Regular	Heart	270	9	120	1.25
	Breast	277	10	88	2.15
	Wdbc	569	14	212	1.68
	Pima	768	8	268	1.85
Large	Adult-a	32,561	123	7841	3.15
	w8a	49,749	300	1933	32.47
	Real-sim	72,309	20,958	22,238	2.25
	Ijcnn1	141,691	22	13,565	9.45

is greater than C. According to (17), we can have $|I_1|(I_1 \equiv \{i|0 \le \mu_i < C\theta_i\})$ is generally more than $|I_1^l|(I_1^l \equiv \{i|0 \le \mu_i^l < C\})$, and $|I_2|$ is generally less than $|I_2^l|$ $(I_2^l \equiv \{i|\mu_i^l = C\})$ when restraints are added. According to (21), we can deduce that CS-LDM generally has higher positive detection rate compared with LDM. Similarly, For CS-LDM, the positive detection rate gradually increases with the increasing of θ_i or ρ.

5. Experiments

In our experiments, LDM is treated as CS-LDM with $\rho = 0$ for convenience. We evaluate the performance of CS-LDM on some imbalanced data sets. We first introduce performance metrics, then describe experimental setup, and finally compare CS-LDM with LDM and two popular cost-sensitive methods.

5.1. Performance metrics

To evaluate the performance of the classifier, four metrics are given as:

$$\mathrm{Acc} = \frac{\mathrm{tp}+\mathrm{tn}}{\mathrm{tp}+\mathrm{fp}+\mathrm{fn}+\mathrm{tn}}, \mathrm{TPR} = \frac{\mathrm{tp}}{\mathrm{tp}+\mathrm{fn}}, \mathrm{TNR} = \frac{\mathrm{tn}}{\mathrm{tn}+\mathrm{fp}},$$

$$G-mean = \sqrt{\mathrm{TPR} * \mathrm{TNR}},$$

where tp indicates the amount of true positives, tn indicates the amount of true negatives, fn indicates the amount of false negatives,and fp indicates the amount of false positives. In the context of balanced data, it is reasonable to use the Acc to evaluate classification performance. In the presence of imbalanced data with unequal error costs, G-mean [23] is adopted to evaluate the classifier performance. The balanced classifier can be derived when TPR = TNR. In other words, we need to make the detection rate between two classes balanced. We define a special value of ρ which can make the detection rate between two classes balanced as the balance point. Once the balance point is determined, the balanced classifier can be derived. The balance point can be determined by experiments.

5.2. Experimental setup

We evaluate the performance of CS-LDM both on regular and large scale data sets from UCI data sets. Table 1 summarizes the statistics of these data sets. The ratio between two classes of sample sizes is ranged from 1.25 to 32.47. For each data set, half of the examples are randomly selected as the training data , and the remaining examples are used as the test data. The regularization parameter C, λ_1, λ_2 are selected via 5-fold cross validation from the set $[0, 2^{-10}, \ldots, 2^{10}]$ on the training data at the first random data partition. The parameters η_t, ν_t and t_0 are set as [14], and T is fixed to 5. CS-LDM is compared with LDM, and two popular cost-sensitive methods, that is, cost-sensitive SVM (CS-SVM)[5] and SVM combined with SMOTE [22](SM-SVM). Experiments are repeated for 10 times with random data partitions, and the average metrics as well as the standard deviations are recorded.

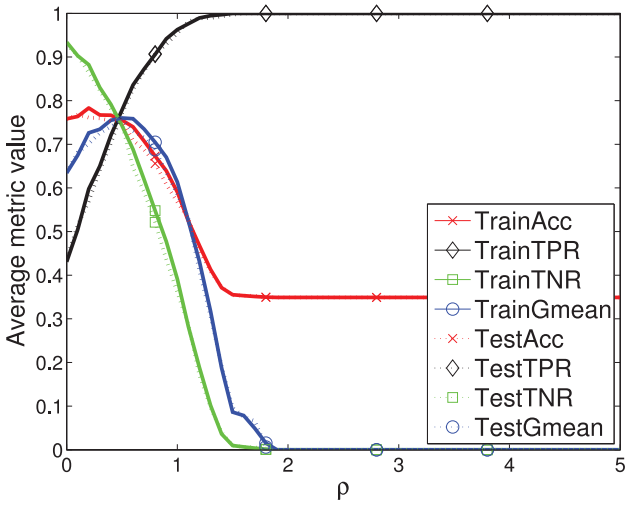

Fig. 3. Average metric curves using linear kernel on pima data set.

Table 2
Regular scale G-mean comparison using linear kernel (the bolded indicates the G-mean is the best in this row).

Dataset	LDM	CS-SVM	SM-SVM	CS-LDM
Heart	0.816 ± 0.031	0.828 ± 0.025	0.823 ± 0.027	**0.842 ± 0.016**
Breast	0.437 ± 0.062	0.651 ± 0.032	0.643 ± 0.036	**0.659 ± 0.026**
Wdbc	0.953 ± 0.011	**0.963 ± 0.011**	0.954 ± 0.012	0.961 ± 0.010
Pima	0.594 ± 0.030	0.735 ± 0.016	0.742 ± 0.015	**0.756 ± 0.010**
Ave. G	0.700	0.794	0.791	**0.805**

Table 3
Regular scale G-mean comparison using RBF kernel.

Dataset	LDM	CS-SVM	SM-SVM	CS-LDM
Heart	0.816 ± 0.031	0.817 ± 0.028	0.816 ± 0.026	**0.826 ± 0.018**
Breast	0.554 ± 0.062	**0.647 ± 0.033**	0.621 ± 0.039	0.645 ± 0.026
Wdbc	0.960 ± 0.015	0.954 ± 0.015	0.956 ± 0.017	**0.964 ± 0.008**
Pima	0.641 ± 0.025	0.735 ± 0.019	0.739 ± 0.019	**0.756 ± 0.014**
Ave. G	0.743	0.788	0.783	**0.798**

5.3. Results on regular scale data sets

Fig. 3 plots the metric curves on pima data set using linear kernel, and the curves for other data sets are similar. Note that TrainAcc marks the Acc curve on the training data, and TestAcc represents the Acc curve on the test data. The other curves are marked similarly. As can be seen, for LDM ($\rho = 0$), the TPR is generally lower than the TNR. CS-LDM can gradually improve the detection rate of the positive class with the increasing of ρ. Four metric curves on the training data intersect at the point defined as the balance point which is around 0.5. The balance point is similar to the point of maximum G-mean. Furthermore, each pair of curves on the training and test data have similar shapes and the balance points. In other word, CS-LDM has a strong generalization performance on regular scale data sets. Table 2 summarizes the results on regular scale data sets using linear kernel. Note that the value of CS-LDM is sampled at the balance point. We can find that the G-mean of CS-LDM is superior or highly competitive to LDM and other compared methods. As linear kernel, Table 3 shows that the G-mean of CS-LDM is also superior or highly competitive to LDM and other compared methods when RBF kernel is used.

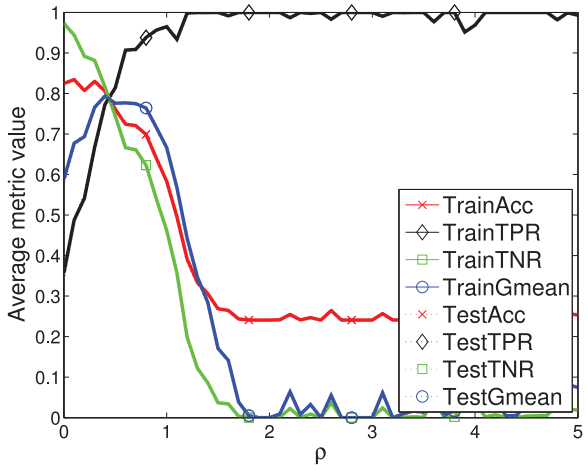

Fig. 4. Average metric curves on adult-a data set.

Table 4
Large scale *G*-mean comparison.

Dataset	LDM	CS-SVM	SM-SVM	CS-LDM
Adult-a	0.594 ± 0.057	0.764 ± 0.004	0.774 ± 0.014	**0.792 ± 0.007**
w8a	0.652 ± 0.025	0.875 ± 0.016	0.852 ± 0.018	**0.878 ± 0.006**
Real-sim	0.884 ± 0.021	0.912 ± 0.020	0.918 ± 0.011	**0.941 ± 0.008**
Ijcnn1	0.536 ± 0.076	**0.828 ± 0.026**	0.821 ± 0.034	0.821 ± 0.032
Ave. *G*	0.667	0.845	0.841	**0.858**

Table 5
Time cost comparison (s) obtained from the same PC.

Dataset	Adult-a	w8a	Real-sim	Ijcnn1
LDM	0.15	0.43	12.32	0.76
CS-LDM	0.16	0.46	12.43	0.77

5.4. Results on large scale data sets

Fig. 4 plots the metric curves on adult-a data set, and the curves for other data sets are similar. It illustrates that the TPR of LDM ($\rho = 0$) is generally lower than TNR and the TPR of CS-LDM gradually rises as ρ increases. The balance point determined by the intersection is around 0.5. Moreover, each pair of curves on the training data and test data have similar shapes and the balance points. Namely, CS-LDM has a strong generalization performance on large scale data sets. Table 4 shows that the *G*-mean of CS-LDM is superior or highly competitive to LDM and other compared methods on large scale data sets. In summary, it is concluded that CS-LDM can effectively improve the TPR of the minority class and get balanced detection rate at the balance point. Meanwhile, for a special value of ρ, CS-LDM is as efficient as LDM as shown in Table 5.

6. Summary and conclusions

In this paper, we propose cost-sensitive large margin distribution machine (CS-LDM) to address the lower detection rate of the minority class and obtain a balanced classifier. CS-LDM including the cost-sensitive margin mean and penalty, tries to maximize the cost-sensitive margin mean, while minimizing the margin variance and cost-sensitive penalty. Theoretical results indicate that CS-LDM subsumes LDM when the majority class size is equal to the minority class size and CS-LDM can gradually increase the detection rate of the minority class with the increasing of the cost parameter. Experimental results show that CS-LDM can effectively improve the detection rate of the minority class and obtain more balanced classifier when the parameter is at the balance point. In the future it will be interesting to study cost-sensitive margin variance and the relationship between the balance point and the ratio.

Acknowledgments

This work is financed by the China National Natural Science Foundation (Approval no. 61573299) and the Natural Science Foundation of Fujian Province (Grant no. 2014J05079).

References

[1] W. Gao, Z.H. Zhou, On the doubt about margin explanation of boosting, Artif. Intell. 203 (5) (2013) 1–18.
[2] L. Breiman, Prediction games and arcing classifiers, Neural Comput. 11 (7) (1999) 1493–1517.
[3] L. Reyzin, R.E. Schapire, How boosting the margin can also boost classifier complexity, in: Proceedings of the 2006 International Conference on Machine Learning, 2006, pp. 753–760.
[4] R.E. Schapire, et al., Boosting the margin: A new explanation for the effectives of voting methods, in: Proceedings of the 1997 International Conference on Machine Learning, 1997, pp. 322–330.
[5] H.G. Chew, et al., Target detection in radar imagery using support vector machines with training size biasing, in: Proceedings of the 2000 International Conference on Control, Automation, Robotics, and Vision (ICARCV), 2000.
[6] Z.H. Zhou, Large margin distribution learning, in: Proceedings of the Sixth IAPR TC3 International Workshop, ANNPR 2014, Montreal, QC, Canada, 2014, pp. 1–11.
[7] T. Zhang, Z.H. Zhou, Large margin distribution machine, in: Proceedings of the Twentieth ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), New York, NY, 2014, pp. 313–322.
[8] C. Cortes, V. Vapnik, Support-vector networks, Mach. Learn. 20 (5) (1995) 273–297.
[9] V. Vapnik, The Nature of Statistical Learning Theory, Springer-Verlag, NewYork, NY, 1995.
[10] S. Theodoridis, K. Koutroumbas, Pattern Recognition, fourth ed., Academic Press, American, 2008.
[11] F. Aiolli, G.S. Martino, A. Sperduti, A kernel method for the optimization of the margin distribution, in: Proceedings of the Eighteenth International Conference on Artificial Neural Networks, Prague, Czech, 2008, p. 305C314.
[12] G.X. Yuan, C.H. Ho, C.J. Lin, Recent advances of large-scale linear classification, Proc. IEEE 100 (2012) 2584–2603.
[13] C.J. Hsieh, K.W. Chang, C.J. Lin, A dual coordinate descent method for large-scale linear SVM, Proceedings of the Twenty-fifth International Conference on Machine Learning, ACM, 2008, pp. 1369–1398.
[14] W. Xu, Towards optimal one pass large scale learning with averaged stochastic gradient descent, in: Computer Science, 2001.
[15] L. Bottou, Large-scale machine learning with stochastic gradient descent, in: Proceedings of COMPSTAT'2010, Physica-Verlag HD, 2010, pp. 177–186.
[16] Y. Singer, N. Srebroet, Pegasos: Primal estimated sub-gradient solver for SVM, Math. Program. 127 (1) (2007) 3–30.
[17] C. Elkan, The foundations of cost-sensitive learning, Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, 2001, pp. 973–978.
[18] T. Joachims, Transductive inference for text classification using support vector machines, Proceedings of the Sixteenth International Conference on Machine Learning, Morgan Kaufmann Publishers Inc., 1999, pp. 200–209.
[19] K. K. Huang, et al., Learning classifiers from imbalanced data based on biased minimax probability machine, in: Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2, 2013, pp. 558–563.
[20] W. Fan, et al., AdaCost: Misclassification cost-sensitive boosting, Proceedings of the Sixteenth International Conference on Machine Learning, Morgan Kaufmann Publishers Inc., 1999, pp. 97–105.
[21] Y. Sun, et al., Cost-sensitive boosting for classification of imbalanced data, Pattern Recognit. 40 (12) (2007) 3358–3378.
[22] N.V. Chawla, et al., SMOTE: Synthetic minority over-sampling technique, J. Artif. Intell. Res. 16 (1) (2011) 321–357.
[23] M. Kubat, S. Matwin, Addressing the curse of imbalanced training sets: One-sided selection, in: Proceedings of the Fourteenth International Conference on Machine Learning, 2000, pp. 179–186.
[24] G.E.A.P.A. Batista, et al., A study of the behavior of several methods for balancing machine learning training data, ACM SIGKDD Explor. Newsl. 6 (1) (2004) 20–29.
[25] C. Seiffert, et al., RUSBoost: A hybrid approach to alleviating class imbalance, IEEE Trans. Syst. Man Cybern. Part A Syst. Hum. 40 (1) (2010) 185–197.
[26] M. Galar, et al., A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches, IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. 42 (4) (2012) 463–484.