Physica A 509 (2018) 422–438

Contents lists available at ScienceDirect

Physica A

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

A biased least squares support vector machine based on Mahalanobis distance for PU learning

Ting Ke*, Hui Lv, Mingjing Sun, Lidong Zhang

Department of Mathematics, College of Science, Tianjin University of Science and Technology, Tianjin, 300457, China

HIGHLIGHTS

- We propose a new Mahalanobis distance-based least squares support vector machines (MD-BLSSVM) classifier, in which two Mahalanobis distances are constructed according to the covariance matrices of two class data for PU learning.
- Excellent kernel technique can be introduced to solve the linear non-separable problem in a reproducing kernel Hilbert space after making certain linear transformation ingeniously.
- MD-BLSSVM not only possesses faster learning speed, but also obtains better generalization than BLSSVMs and other methods.

ARTICLE INFO

Article history:
Received 29 January 2018
Received in revised form 11 May 2018
Available online 31 May 2018

Keywords:
Positive and unlabeled learning
Least squares support vector machine
Mahalanobis distance
Regularization

ABSTRACT

In many domains, the presence of both positive and negative examples is not satisfied and only one class of examples is available. This special case of binary classification is called as PU (positive and unlabeled) learning in short. At present, many classification algorithms have been introduced for PU learning, such as BLSSVM, BSVM and so on. However, all of these classical approaches were measured by Euclidean distance, which did not take into account the correlative information of each class and the fluctuation of various attributions. In order to reflect this information, we propose a new Mahalanobis distance-based least squares support vector machines (MD-BLSSVM) classifier, in which two Mahalanobis distances are constructed according to the covariance matrices of two class data for optimizing the hyper-planes. Actually, MD-BLSSVM has a special case of BLSSVMs when the covariance matrices are degenerated to the identity matrix. The merits of MD-BLSSVM are (1) Mahalanobis distance of two classes can measure more suitable distance with certain weights on attributions; (2) Excellent kernel technique can be introduced in a reproducing kernel Hilbert space after making certain linear transformation ingeniously; (3) A solution is obtained simply by solving the system of linear equations. In all, MD-BLSSVM is appropriate to many real problems, especially for the case that the distribution and correlation of two classes of data are obviously different. The experimental results on several artificial and benchmark datasets indicate that MD-BLSSVM not only possess faster learning speed, but also obtains better generalization than BLSSVMs and other methods.

* Corresponding author.
E-mail address: kk.ting@163.com (T. Ke).

https://doi.org/10.1016/j.physa.2018.05.128
0378-4371/

1. Introduction

Training binary classifiers on positive and unlabeled data is referred to as PU learning [1]. PU learning has no training negative data because negative training data is either hard to obtain or even not available at all in some domains. Fig. 1 shows the distribution of PU data in two-dimensional space. In Fig. 1, each example has two features, $[x]_1$ and $[x]_2$. "Red $+$" represents positive data and "blue circle" represents unlabeled data, where "blue $\oplus$" and "blue $\ominus$" describe positive and negative data in unlabeled examples respectively. How to identify the label of unlabeled data (blue circle) or any new data point effectively is our goal in this paper.

To cope with this setting, a theoretical study of probably approximately correct (PAC) learning from positive and unlabeled examples was first done in [2]. The study concentrated on the computational complexity of learning and showed that function classes learnable under the statistical queries model. Recently, learning from positive examples was also studied theoretically in [3] within a Bayesian framework where the distribution of functions and examples were assumed known. Roughly speaking, most research works in this area are divided into three types of methods.

The first type is the two-step strategy, which selects possible negative or positive examples from unlabeled examples, and then builds classifiers using positive examples and negative examples. The popular used techniques for extracting possible negative or positive examples included spy, Rocchio and 1-DNF [4]. After extracting possible negative or positive examples, standard machine learning methods such as Naïve Bayes and SVM [5] are used to train classifiers. At present, the two-step strategy is a widely used method, such as S-EM [6], ROC-SVM [7], 1-DNF, PNLH [8], LCLC [9], Pulce [10,11] etc. Pulce took advantage of the intrinsic relationships between attribute values and exceeded the independence assumption made by Naïve Bayes. In fact, leverages on the statistical properties of the data to learn a distance metric employed during the classification task.

The second type of methods is the probability estimation approach [12,13]. In [12], it took each unlabeled example as both positive and negative example with weights pre-computed by an additional classifier for protein record identification. Luigi Cerulo et al. made use of the approach in [14] to reconstruct gene regulatory networks without negative examples.

The third type of methods is a one-step method. It includes one-class SVM [15,16], BSVM [17], WL [18,19], BLSSVM [20] LUHC [21]. One class SVM is effective only for the case where the number of positive examples is large enough to capture the characteristics of the positive class, and their performance would be rather poor when this number is very small [22]. BSVM is built by giving appropriate weights to the positive examples and unlabeled examples which are regarded as negative examples with noise, respectively. Experimental results indicated that the performance is better than most of two-step strategies. WL used Logistic Regression technique after weighting the negative class. BLSSVM utilized all training data to learn a biased least squares SVM classifier. It reduces to be more stable and effective than BSVM. Moreover, the time complexity of BLSSVM is lower than that of BSVM, where BLSSVM only needs to solve linear equations and BSVM is a quadratic programming. LUHC proposed a Laplacian unit hyper-plane classifier which adding a manifold regularizer to make the predicted labels and the initial labels sufficiently close on the labeled points.

As an excellent state-of-the-art tools for classical binary classifier in machine learning [23,24], support vector machines (SVMs) has already outperformed most of other learning algorithms. One of the important reasons for the successfulness of SVMs is the employment of kernel technique. However, the kernels used in SVMs are based on Euclidean distance. That is, SVMs assume data points are more likely distributed within a hyper-spherical region. However, data points in two classes are more likely distributed within two different hyper-ellipsoidal regions. For this more general case, Mahalanobis distance, which was introduced by P. C. Mahalanobis in 1936 [25], takes into account the correlations of the dataset and is scale-invariant, is a better choice [26]. Many efforts were made toward classifying data based on Mahalanobis distance in a reproducing kernel Hilbert space [27–31]. Correspondingly, as one of the most effective technique for PU learning, BLSSVM and BSVM are still measured by Euclidean distance. This is inappropriate for certain data distributions.

Therefore, a more suitable distance measure for different distribution of data points for PU learning is introduced in this paper. We present a Mahalanobis distance-based biased least squares support vector machine (MD-BLSSVM) classifier for PU learning. In MD-BLSSVM, a pair of Mahalanobis distance-based inner products in a reproducing kernel Hilbert space is first introduced according to the empirical covariance matrices of the two classes of data. In fact, MD-BLSSVM has a special case of BLSSVM when the covariance matrices of two classes of data in a reproducing kernel Hilbert space are both degenerated to the identity matrix. Compared to other Mahalanobis distance-based methods, the idea in this paper is not only simpler and more natural, but also considers the Mahalanobis distance-based kernels for the two classes of data respectively. MD-BLSSVM successfully inherits the merit of BLSSVM, i.e., fast learning speed and good adaptability and robustness for PU learning. In addition, MD-BLSSVM effectively combines the covariance matrix information of two classes of data in the prediction stage. As a simple illustration on the MD-BLSSVM significance, Fig. 2 shows the learning result of the linear MD-BLSSVM. It can be found that the linear MD-BLSSVM obtains the much better separating hyper-plane than the linear BLSSVM. Computational comparisons on BLSSVM, MD-BLSSVM in terms of generalization performance and learning speed have been made on several artificial and benchmark datasets, indicating MD-BLSSVM is not only fast, but also shows comparable generalization.

The rest of this paper is organized as follows. Section 2, briefly dwells on distance measures and previous PU algorithm, such as BLSSVM. The proposed MD-BLSSVM is presented in Section 3. Then in Section 4, experiments on both synthetic and real datasets are reported. Finally, we conclude this paper in Section 5.

2. Theories

Before we go into the detail of related works, let us give the definition of PU learning firstly. Throughout this paper, we suppose that positive training dataset is represented as $X_1 = \left(x_1, \quad x_2, \quad \ldots, \quad x_p\right)^T$ and unlabeled training dataset is $X_2 = \left(x_{p+1}, \quad x_{p+2}, \quad \ldots, \quad x_{p+u}\right)^T$, let $X = \left(X_1^T, X_2^T\right)^T$, where p and u are the number of positive and unlabeled examples respectively, $x_i \in \mathbb{R}^n$ is a training example input and $y_i = 1$ is the positive class label of $x_i, (i = 1, \ldots, p)$, the label of $x_j, j = p+1, \ldots, p+u$ is unknown, but it equals $+1$ or -1. The objective of PU learning is to find a real function $f(x)$ in $\mathbb{R}^n$ such that the output value of y for any x can be predicted by the sign function of $f(x)$ as

$$\text{sign}(f(x)). \tag{1}$$

2.1. Distance measures

The training data is assumed to be independently sampled from a multivariate normal distribution $N_n\left(\mu, \sum\right)$, where $\sum$ is the population covariance matrix. A general measure of squared distance from an observation $x_i = (x_{i1}, \ldots, x_{in})^T \in \mathbb{R}^n$ to the other data point $x_j = \left(x_{j1}, \ldots, x_{jn}\right)^T \in \mathbb{R}^n$ can be written as follows:

$$d\left(x_i, x_j\right) = \left(x_i - x_j\right)^T M \left(x_i - x_j\right) \tag{2}$$

where $M_{n\times n}$ is a real symmetrical matrix. It can be easily noted that formula (2) reduces squared Euclidean distance if $M_{n\times n}$ is the identity matrix. If $M = \sum^{-1}$, the squared Mahalanobis distance is obtained as:

$$d_M\left(x_i, x_j\right) = \left(x_i - x_j\right)^T \sum^{-1} \left(x_i - x_j\right) \tag{3}$$

If the covariance matrix is diagonal, then the resulting distance measure is called a standardized Euclidean distance:

$$\sum_{k=1}^{n} \left(\frac{x_{ik} - x_{jk}}{\sigma_k}\right)^2 \tag{4}$$

where σ_k is the standard deviation of the x_{ik} and x_{jk} over the sample set.

Mahalanobis distance is an effective method for computing the similarity of two unknown sample sets. It is generally known that real symmetrical covariance matrix reflects the linear correlations among random variables. It is generally known that $\sum$ can be decomposed into the product of a lower triangular matrix and its conjugate transpose utilizing Cholesky decomposition or Cholesky factorization [32]. It means that

$$\sum = T^T \Lambda T = T^T \Lambda^{\frac{1}{2}} \left(T^T \Lambda^{\frac{1}{2}}\right)^T = AA^T \tag{5}$$

where Λ is a diagnose matrix, the diagnosed entry are eigenvalues. T is an orthogonal matrix, where the column vectors are composed by eigenvectors.

Actually, Mahalanobis distance is the Euclidean distance after a linear mapping. This linear mapping goes through two steps. The first step is decorrelation about n dimension random variables. Namely, we compute the covariance matrix of TX after linear transformation $X \to TX$:

$$\text{cov}\left(TX, TX\right) = T\ \text{cov}\left(X, X\right) T^T = T \sum T^T = \Lambda = \begin{pmatrix} \sigma_1^2 & 0 & 0 & 0 \\ 0 & \sigma_2^2 & 0 & 0 \\ & & \ddots & \\ 0 & 0 & & 0 \\ 0 & 0 & 0 & \sigma_n^2 \end{pmatrix} \tag{6}$$

Formula (6) illustrates the new random variables are uncorrelated after the linear transformation and variance of ith random variable is $\sigma_i^2, i = 1, \ldots, n$. The second step is standardizing for every random variable. Namely, we compute the covariance matrix of $\Lambda^{-\frac{1}{2}} TX$ after linear transformation $TX \to \Lambda^{-\frac{1}{2}} TX$:

$$\begin{aligned} &\text{cov}\left(\Lambda^{-\frac{1}{2}} TX, \Lambda^{-\frac{1}{2}} TX\right) = \Lambda^{-\frac{1}{2}} T\ \text{cov}\left(X, X\right) \left(\Lambda^{-\frac{1}{2}} T\right)^T \\ &= \Lambda^{-\frac{1}{2}} T \sum T^T \Lambda^{-\frac{1}{2}} = \Lambda^{-\frac{1}{2}} \Lambda \Lambda^{-\frac{1}{2}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ & & \ddots & \\ 0 & 0 & & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{aligned} \tag{7}$$

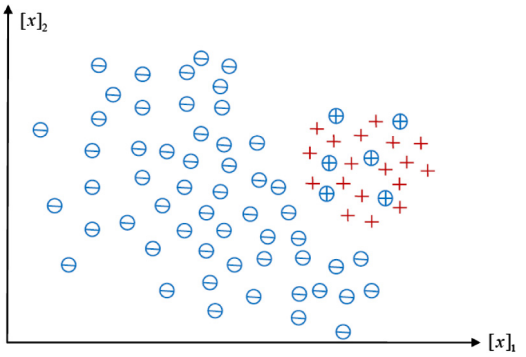


Fig. 1. Two-dimensional data distribution for PU learning.

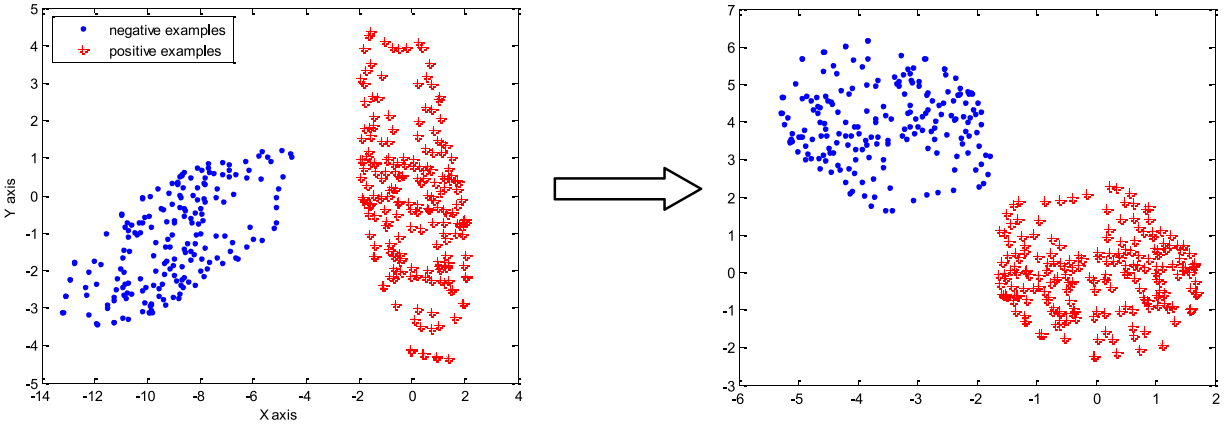


Fig. 2. The interpretation of the difference between Euclidean distance and Mahalanobis distance on the distribution of 2-dimension data points.

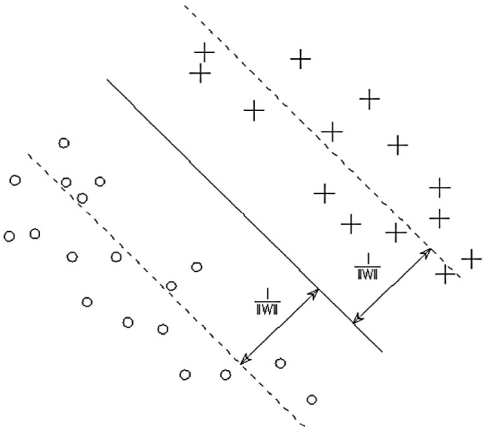


Fig. 3. An interpretation of two-dimensional LSSVM.

Thus, Euclidean distance of $\Lambda^{-\frac{1}{2}}Tx_i and \Lambda^{-\frac{1}{2}}Tx_j$ is

$$\begin{aligned} &d\left(\Lambda^{-\frac{1}{2}}Tx_i, \Lambda^{-\frac{1}{2}}Tx_j\right) = \left(\Lambda^{-\frac{1}{2}}Tx_i - \Lambda^{-\frac{1}{2}}Tx_j\right)^T \left(\Lambda^{-\frac{1}{2}}Tx_i - \Lambda^{-\frac{1}{2}}Tx_j\right) \\ &= \left(x_i - x_j\right)^T T^T \Lambda^{-1} T \left(x_i - x_j\right) = \left(x_i - x_j\right)^T \sum^{-1} \left(x_i - x_j\right) = d_M\left(x_i, x_j\right) \end{aligned} \tag{8}$$

The above formula illustrates the connection and difference between the two distance measures. Actually, Euclidean distance treats all random variables as equally important and is regardless of the correlation between random variables. Nevertheless, Mahalanobis distance gives different weights for different variances of random variables. i.e., larger variance

gives less weight of attributes, Fig. 2 shows the above theory. The positive and negative data points have different elliptic distributions respectively. There are different variances on two random variables in the left figure of Fig. 2. Thus, it is unsuitable for Euclidean distance which is deemed as equal weight on two dimensions. Data points after linear transformation as formula (8) show the circular distribution as the right figure in Fig. 2. Now, it is more suitable that the right data points are measured by Euclidean distance. In other words, Euclidean distance of right data points is just Mahalanobis distance of left data points.

2.2. Review of biased least squares SVM

The least squares support vector machine (LSSVM) is a promising classification technique proposed firstly by Suykens et al. [33–35]. LSSVM attempts to seek an optimal separating hyper-plane $f(x) = \langle w, \varphi(x)\rangle + b = 0, w \in \mathbb{R}^n, b \in \mathbb{R}$ that maximizes the margin between two classes while simultaneously to minimize the least squares error based on given training examples $\{x_i, y_i\}, i = 1, \ldots, l$, where x_i is a vector in the input space $\mathbb{R}^n$ and y_i denotes the class label taking a value of $+1$ or -1. $\varphi(\cdot)$ is a nonlinear function which is used to map the input space into a higher dimensional space. Therefore, the maximum-margin classifier is obtained by solving

$$\begin{aligned} &\min_{(w,b,\xi)} \quad \frac{1}{2}\|w\|^2 + \frac{C}{2}\sum_{i=1}^{l}\xi_i^2 \\ &s.t.\ \ y_i(\langle w, \varphi(x_i)\rangle + b) = 1 - \xi_i, \quad i = 1, 2, \ldots, l \end{aligned} \tag{9}$$

The geometric interpretation of the above problem (9) with $x \in \mathbb{R}^2$ for linearly separable case is shown in Fig. 3, where minimizing $\frac{1}{2}\|w\|^2$ realizes the maximal margin between the straight lines $\langle w, \varphi(x)\rangle + b = 1$ and $\langle w, \varphi(x)\rangle + b = -1$, minimizing $\sum_{i=1}^{l}\xi_i^2$ makes the two straight lines proximal to all inputs of positive ("+") and negative ("∘") examples respectively.

It is worth mentioning that the superiority of LSSVM is simpler and the computation time is lower than SVM. This is because LSSVM needs to solve a quadratic programming with only equality constraints, or equivalently a linear system of equations, whereas SVM needs to solve a quadratic programming with inequality constraints.

For a special learning problem-PU learning, a biased least squares SVM classifier (BLSSVM) was constructed in [20] by giving two different penalty parameters to weight the misclassification errors of positive and unlabeled examples respectively, where unlabeled examples can be regarded as negative examples with noise. i.e., $y_i = -1, i = p+1, \ldots, p+u$. Thus, we seek the decision function so that the classification hyper-plane separates both positive examples and unlabeled examples with the maximum margin.

$$\begin{aligned} &\min_{(w,b,\xi)} \quad \frac{1}{2}\|w\|^2 + \frac{C_p}{2}\sum_{i=1}^{p}\xi_i^2 + \frac{C_n}{2}\sum_{i=p+1}^{p+u}\xi_i^2 \\ &s.t.\ \ y_i(\langle w, x_i\rangle + b) = 1 - \xi_i, \quad i = 1, 2, \ldots, p+u \end{aligned} \tag{10}$$

where equality constraint in optimization problem (10) and $C_p\sum_{i=1}^{p}\xi_i^2 + C_n\sum_{i=p+1}^{p+u}\xi_i^2$ imply that all positive examples approximate to straight line $\langle w, \varphi(x)\rangle + b = 1$ and unlabeled examples approximate to $\langle w, \varphi(x)\rangle + b = -1$ respectively. Extensive empirical comparisons show that this classifier is able to obtain good performance on many problems with a fast training speed.

3. A biased least squares support vector machine based on Mahalanobis distance

In formula (11), the maximum margin is measured by Euclidean distance. However, Mahalanobis distance can offer more reasonable measure than Euclidean distance for most data distributions as depicted in Section 2. Generally, if we substitute Mahalanobis distance for classical Euclidean distance, the following Mahalanobis distance-based inner product is $x_i^T\sum^{-1}x_j$. Since the distributions of data points are unknown, we consider the empirical covariance matrices on two class datasets. Now, centered at $\bar{x}$ for concrete training data $x_i, i = 1, \ldots, p+u$, the covariance matrix $\sum$ can be calculated as:

$$\sum = \frac{1}{p+u-1}\sum_{i}^{p+u}(x_i - \bar{x})(x_i - \bar{x})^T \tag{11}$$

Remark that, for many real-world classification problems, different classes of points have different distributions, and then the covariance matrices of these classes of data are also different. Thus, it is necessary to simultaneously consider the covariance matrices of the two classes, which are

$$\begin{aligned} \sum\nolimits_1 &= \frac{1}{p-1}\sum_{i=1}^{p}(x_i - \overline{x_1})(x_i - \overline{x_1})^T, \\ \sum\nolimits_2 &= \frac{1}{u-1}\sum_{i=p+1}^{p+u}(x_i - \overline{x_2})(x_i - \overline{x_2})^T. \end{aligned} \tag{12}$$

where $\overline{x_i}$, $i = 1, 2$ are the means of positive and unlabeled data. Based on the above discussion on the Mahalanobis distance-based inner products for different classes, we seek the decision function

$$g(x) = w^T \sum^{-1} x + b,\ w \in \mathbb{R}^n,\ b \in \mathbb{R} \tag{13}$$

so that the classification hyper-plane separates both positive examples and unlabeled examples with the maximum margin, resulting in the primal optimization problem by introducing Mahalanobis distance of each class:

$$\begin{aligned} &\min_{w,b,\xi_i} \quad \frac{1}{2} w^T \sum^{-1} w + \frac{C_p}{2} \sum_{i=1}^{p} \xi_i^2 + \frac{C_n}{2} \sum_{i=p+1}^{p+u} \xi_i^2 \\ &s.t.\ \ y_i(w^T \sum_1^{-1} x_i + b) = 1 - \xi_i,\ i = 1, 2, \ldots, p \\ &\qquad y_i(w^T \sum_2^{-1} x_i + b) = 1 - \xi_i,\ i = p+1, p+2, \ldots, p+u \end{aligned} \tag{14}$$

Note that the inverses of $\sum, \sum_1, \sum_2$ need to be computed in formula (14). It is worth mentioning that the covariance matrices $\sum, \sum_1, \sum_2$ are irreversible when the number of samples is smaller than the dimension. To solve this problem, we add a small positive number υ to the diagonal element. That is, we actually compute the inverses of $\sum_i + \upsilon I$, $i = 1, 2$. Setting

$$\begin{aligned} Y_1 &= diag(y_1, y_2, \ldots, y_p), \\ Y_2 &= diag(y_{p+1}, y_{p+2}, \ldots, y_{p+u}), \\ C_1 &= diag\,(\underbrace{C_p, \ldots, C_p,}_{p}), \\ C_2 &= diag\,(\underbrace{C_n, \ldots, C_n}_{u}), \\ \xi_P &= (\xi_1, \xi_2, \ldots, \xi_p)^T, \\ \xi_U &= (\xi_{p+1}, \xi_{p+2}, \ldots, \xi_{p+u})^T. \end{aligned} \tag{15}$$

Rewriting the optimization model (14) as the matrix format:

$$\begin{aligned} &\min_{w,b,\xi_P,\xi_U} \quad \frac{1}{2} w^T \sum^{-1} w + \frac{1}{2} \xi_P^T C_1 \xi_P + \frac{1}{2} \xi_U^T C_2 \xi_U \\ &s.t.\ \ Y_1 \left(X_1 \sum_1^{-1} w + e_1 b \right) + \xi_P = e_1 \\ &\qquad Y_2 \left(X_2 \sum_2^{-1} w + e_2 b \right) + \xi_U = e_2 \end{aligned} \tag{16}$$

where e_1, e_2 are p-dimension and u-dimension vector of ones.

Clearly, the minimizing problem (16) is a quadratic programming with only equality constraints. Introducing the Lagrange multiplier $\alpha_P = (\alpha_1, \ldots, \alpha_p)^T$, $\alpha_U = (\alpha_{p+1}, \ldots, \alpha_{p+u})^T$ for optimization problem (16), we obtain

$$\begin{aligned} L(w, b, \xi, \alpha) \ = \ & \frac{1}{2} w^T \sum^{-1} w + \frac{1}{2} \xi_P^T C_1 \xi_P + \frac{1}{2} \xi_U^T C_2 \xi_U \\ & - \alpha_P^T Y_1 \left(X_1 \sum_1^{-1} w + e_1 b \right) - \alpha_P^T \xi_P + \alpha_P^T e_1 \\ & - \alpha_U^T Y_2 \left(X_2 \sum_2^{-1} w + e_2 b \right) - \alpha_U^T \xi_U + \alpha_U^T e_2 \end{aligned} \tag{17}$$

According to Karush–Kuhn–Tucker (KKT) necessary and sufficient optimality conditions, taking the partial derivatives of (17) with respect to w, b, ξ_P, ξ_U, α_P, α_U and equating them to zero, we receive the following optimal condition

$$\begin{aligned} &\frac{\partial L}{\partial w} = \sum^{-1} w - (\sum_1^{-1})^T X_1^T Y_1^T \alpha_P - (\sum_2^{-1})^T X_2^T Y_2^T \alpha_U = 0 \\ &\qquad \Rightarrow w = ((\sum_1^{-1})^T X_1^T Y_1^T \alpha_P + (\sum_2^{-1})^T X_2^T Y_2^T \alpha_U) \sum \end{aligned} \tag{18}$$

$$\frac{\partial L}{\partial b} = e_1^T Y_1^T \alpha_P + e_2^T Y_2^T \alpha_U = 0 \tag{19}$$

$$\frac{\partial L}{\partial \xi_P} = C_1 \xi_P - \alpha_P = 0 \Rightarrow \xi_P = C_1^{-1} \alpha_P \tag{20}$$

$$\frac{\partial L}{\partial \xi_U} = C_2 \xi_U - \alpha_U = 0 \Rightarrow \xi_U = C_2^{-1} \alpha_U \tag{21}$$

$$\frac{\partial L}{\partial \alpha_P} = Y_1\left(X_1\sum\nolimits_1^{-1} w + e_1 b\right) + \xi_P - e_1 = 0 \tag{22}$$

$$\frac{\partial L}{\partial \alpha_U} = Y_2\left(X_2\sum\nolimits_2^{-1} w + e_2 b\right) + \xi_U - e_2 = 0 \tag{23}$$

Substituting (18) and (20) into (22) and substituting (18) and (21) into (23), we can solve the resulting equations for α_P, α_U and b

$$\begin{cases} Y_1X_1\sum\nolimits_1^{-1}\sum(\sum\nolimits_1^{-1})^T X_1^T Y_1^T\alpha_P + Y_1X_1\sum\nolimits_1^{-1}\sum(\sum\nolimits_2^{-1})^T X_2^T Y_2^T\alpha_U + C_1^{-1}\alpha_P + Y_1e_1b = e_1 & \text{(a)} \\ Y_2X_2\sum\nolimits_2^{-1}\sum(\sum\nolimits_1^{-1})^T X_1^T Y_1^T\alpha_P + Y_2X_2\sum\nolimits_2^{-1}\sum(\sum\nolimits_2^{-1})^T X_2^T Y_2^T\alpha_U + C_2^{-1}\alpha_U + Y_2e_2b = e_2 & \text{(b)} \\ e_1^T Y_1^T\alpha_P + e_2^T Y_2^T\alpha_U = 0 & \text{(c)} \end{cases} \tag{24}$$

Now, the Eqs. (24)(a–c) are equivalent to the following format:

$$\begin{pmatrix} Y_1X_1\sum\nolimits_1^{-1}\sum(\sum\nolimits_1^{-1})^T X_1^T Y_1^T + C_1^{-1} & Y_1X_1\sum\nolimits_1^{-1}\sum(\sum\nolimits_2^{-1})^T X_2^T Y_2^T & Y_1e_1 \\ Y_2X_2\sum\nolimits_2^{-1}\sum(\sum\nolimits_1^{-1})^T X_1^T Y_1^T & Y_2X_2\sum\nolimits_2^{-1}\sum(\sum\nolimits_2^{-1})^T X_2^T Y_2^T + C_2^{-1} & Y_2e_2 \\ e_1^T Y_1^T & e_2^T Y_2^T & 0 \end{pmatrix} \begin{pmatrix} \alpha_P \\ \alpha_U \\ b \end{pmatrix} = \begin{pmatrix} e_1 \\ e_2 \\ 0 \end{pmatrix} \tag{25}$$

Thus, solving the system of equations, the decision function is given by

$$\begin{aligned} f(x) &= \operatorname{sign}(g(x) = w^T\sum\nolimits^{-1} x + b) \\ &= \operatorname{sign}(\alpha_P^T Y_1X_1\sum\nolimits_1^{-1} x + \alpha_U^T Y_2X_2\sum\nolimits_2^{-1} x + b) \end{aligned} \tag{26}$$

Next, substituting Eq. (5) into formula (25),

$$\begin{pmatrix} Y_1\left(X_1\sum\nolimits_1^{-1}A\right)\left(X_1\sum\nolimits_1^{-1}A\right)^T Y_1^T + C_1^{-1} & Y_1\left(X_1\sum\nolimits_1^{-1}A\right)\left(X_2\sum\nolimits_2^{-1}A\right)^T Y_2^T & Y_1e_1 \\ Y_2\left(X_2\sum\nolimits_2^{-1}A\right)\left(X_1\sum\nolimits_1^{-1}A\right)^T Y_1^T & Y_2\left(X_2\sum\nolimits_2^{-1}A\right)\left(X_2\sum\nolimits_2^{-1}A\right)^T Y_2^T + C_2^{-1} & Y_2e_2 \\ e_1^T Y_1^T & e_2^T Y_2^T & 0 \end{pmatrix} \begin{pmatrix} \alpha_P \\ \alpha_U \\ b \end{pmatrix} = \begin{pmatrix} e_1 \\ e_2 \\ 0 \end{pmatrix} \tag{27}$$

Comparing the algorithm of LSSVM and ours, the difference lies in the linear transformation of the raw data. i.e., $X_i \to X_i\sum_i^{-1}A = X_iA_i, i = 1, 2$. Such a mapping is more useful for classification. Moreover, it is natural to find that coefficient matrix in (27) contains an inner product operation. It is well known that inner product is a kind of kernel or some distance measure, such as linear kernels, RBF kernels and so on. This kind of transformation plays an important role in feature selection and dimension decrease. In conclusion, these new insights prompt us to extend the above linear separable case to nonlinear separable case. For nonlinearly separable case, we also introduce a nonlinear function $\Phi(X_iA_i), i = 1, 2$ mapping the input feature space to the higher dimensional space. Naturally, Inner product $\Phi(X_iA_i)\Phi(X_jA_j)^T, i = 1, 2$ is deemed as a kernel function (matrix), denoted as $K(X_iA_i, X_jA_j) = \Phi(X_iA_i)\Phi(X_jA_j)^T, i = 1, 2$. Substituting $X_iA_i(X_jA_j)^T$ for $K(X_iA_i, X_jA_j)$ will receive the following linear equations

$$\begin{pmatrix} Y_1K(X_1A_1, X_1A_1)Y_1^T + C_1^{-1} & Y_1K(X_1A_1, X_2A_2)Y_2^T & Y_1e_1 \\ Y_2K(X_2A_2, X_1A_1)Y_1^T & Y_1K(X_2A_2, X_2A_2)Y_2^T + C_2^{-1} & Y_2e_2 \\ e_1^T Y_1^T & e_2^T Y_2^T & 0 \end{pmatrix} \begin{pmatrix} \alpha_P \\ \alpha_U \\ b \end{pmatrix} = \begin{pmatrix} e_1 \\ e_2 \\ 0 \end{pmatrix} \tag{28}$$

Analogy from the formula (27), a nonlinear classifier appears and classifies any new example x as positive class or negative class by the following decision function

$$f(x) = \operatorname{sign}(\alpha_P^T Y_1K(X_1A_1, A_1^T x) + \alpha_U^T Y_2K(X_2A_2, A_2^T x) + b) \tag{29}$$

Right now, we establish the complete learning framework. Our classifier is constructed using Mahalanobis distance to measure the maximum margin hyper-plane of least squares SVM. So it is called MD-BLSSVM for short. The solving process of algorithm for MD-BLSSVM is shown in Table 1. MD-BLSSVM not only inherits all merits of BLSSVM, but also has some own strengths, such as more reasonable distance measure and the ingenious introduction of kernel function in MD-BLSSVM.

4. Experiments

To evaluate the performance of MD-BLSSVM, we perform experiments on three artificial datasets, six UCI benchmark datasets [36] and one USPS handwritten image dataset in this paper. More concretely, we compare MD-BLSSVM with the following state-of-the-art algorithms:

Table 1
The solving algorithm for MD-BLSSVM.

- **Input:** A group of parameters C_p, C_n, kernel parameter, a threshold $\varepsilon=10^{-3}$, $\upsilon=0.01$, Training dataset X_1, X_2, class label matrix of training data $Y=diag(Y_1, Y_2)$; Test dataset Q;
- Selecting the kernel function;
- If $\sum, \sum_1, \sum_2$ are irreversible,
- $\sum_i = \sum_i + \upsilon I, i = 1,2, \sum = \sum + \upsilon I,$
  End
  Computing $A_i = \sum_i^{-1} A, i, j = 1,2;$
- Solve the system of equations in (28) :

If $\left| \det \begin{pmatrix} Y_1K(X_1A_1, X_1A_1)Y_1^T + C_1^{-1} & Y_1K(X_1A_1, X_2A_2)Y_2^T & Y_1e_1 \\ Y_2K(X_2A_2, X_1A_1)Y_1^T & Y_1K(X_2A_2, X_2A_2)Y_2^T + C_2^{-1} & Y_2e_2 \\ e_1^T Y_1^T & e_2^T Y_2^T & 0 \end{pmatrix} \right| > \varepsilon$

Variable vector

$$\begin{pmatrix} \alpha_p^* \\ \alpha_u^* \\ b^* \end{pmatrix} = \begin{pmatrix} Y_1K(X_1A_1, X_1A_1)Y_1^T + C_1^{-1} & Y_1K(X_1A_1, X_2A_2)Y_2^T & Y_1e_1 \\ Y_2K(X_2A_2, X_1A_1)Y_1^T & Y_1K(X_2A_2, X_2A_2)Y_2^T + C_2^{-1} & Y_2e_2 \\ e_1^T Y_1^T & e_2^T Y_2^T & 0 \end{pmatrix}^{-1} \begin{pmatrix} e_1 \\ e_2 \\ 0 \end{pmatrix};$$

Predict the class labels of test dataset Q using (29);

Else
The system of equations in (29) has no solutions;
Exit;
End

- **Output:** decision function and the class labels of test data Q .

1. BLSSVM is a classical least squares support vector machine. It uses positive examples and unlabeled examples which can be negative examples with noise to build a Euclidean distance-based least squares SVM classifier.
2. Pulce takes into account data dependencies and learns a distance model from attribute relationships to train a KNN-like classifier.
3. LUHC (unit-hyper-plane classifier) determines a decision unit-hyper-plane by solving a quadratic programming problem.
4. NB is another PU learning algorithm, which takes 'selected completely at random' assumption and estimates the prior probability of positive, then learn classifier model.
5. S-EM uses spy technique to identify reliable negative examples from the unlabeled examples, and uses the Expectation Maximization (EM) algorithm to build a NB classifier.

All the classifiers are implemented in MATLAB 2014a environment on a PC with Intel Core(TM) core i3 CPU (2.4 GHz) with 2 GB RAMS. In both classification datasets, we use LIBSVM [37] to build a classifier for BSVM, LPU package [38] to implement system for NB and S-EM.

4.1. Performance metric

We use the popular *F* score on the positive class as the evaluation measure. *F* score takes into account of both recall and precision

$$F = \frac{2pr}{p + r} \tag{30}$$

Where

$$p = \frac{TP}{TP + FP}$$

And

$$r = \frac{TP}{TP + FN}.$$

Table 2
Parameter selection algorithm for MD-BLSSVM.

- Set ranges for parameters
  $C_n \in \{C_{\min},\cdots, C_{\max}\}, C_p = 2C_n,$
  $\sigma \in \{\sigma_{\min},\cdots, \sigma_{\max}\}, N_{fold};$
- Partition the training dataset into N_{fold} partitions: $(P_i,U_i), i=1,\cdots,N_{fold};$
- Perform N_{fold} fold cross validation method to optimize parameters:

For each group of parameters $C_p, C_n, \sigma,$

For $i = 1 : N_{fold}$

Substitute (P_i,U_i) into the system of equations (28) ;

Solve the system of equations (28), and compute the F value $\hat{F}_i$ using (31) ;

End

Compute the average F value $\overline{F} = \left(\sum_{i=1}^{N_{fold}} \hat{F}_i\right) / N_{fold};$

End

- Find the optimal parameters vector C_p^*, C_n^*, σ^* that maximizes the average F value using the predictions.

TP, *TN*, *FP* and *FN* are the number of true positive, true negative, false positive and false negative, respectively. A high precision ensures that the identified positives are predominantly true positives, and a high recall ensures that most of the positives are identified. *F* score captures the average effect of both precision and recall, and is therefore suitable for our purpose. In addition, the accuracy of classification is also showed in this paper to evaluate the performance of MD-BLSSVM

4.2. The selection of the parameters

F score cannot be computed on the validation set during the training process because there is no negative example. An approximate computing method [9] is used to evaluate the performance by

$$\hat{F} = \frac{r_p^2}{P(f(X) = 1)} \tag{31}$$

where X is the random variable representing the input vector, $P(f(X) = 1)$ is the probability of an input example X classified as positive example, r_p is the recall for positive set P in the validation set. Therefore, parameter selection algorithm for MD-BLSSVM is shown in Table 2.

Concretely, hyper-parameters are set by five-fold cross validation from some grids introduced in the following.

- MD-BLSSVM. RBF kernel parameter σ, C_n are chosen from $\{2^{-6}, 2^{-5}, \ldots, 2^6\}$ and C_p is searched from $2C_n$,
- LUHC. RBF kernel parameter σ and λ are also searched from $\{2^{-6}, 2^{-5}, \ldots, 2^6\}$ and ν is selected from $\{0.1, 0.2, \ldots, 0.9\}$.
- BSVM. RBF kernel parameter σ, C_n are also chosen from the set $\{2^{-6}, 2^{-5}, \ldots, 2^6\}$ and C_p is searched from $2C_n$.
- Pulce. Parameter k is varied over the set of values {1,3,7}.

The final decision function is obtained by the optimal parameters by selecting the best *F* score in Eq. (30).

4.3. Experiment results

4.3.1. Artificial datasets

Consider two types of two-dimensional synthetic 'elliptic distributions' datasets with a few randomly labeled positive examples. These two types of datasets contain 400 and 800 examples respectively, where red star represents positive examples and blue dot represent negative examples. In order to demonstrate the classification performance for PU learning, 10% and 20% proposition of positive examples are labeled randomly and residual data points are taken as unlabeled examples for these two types of datasets, which are marked black triangle in Figs. 4–7(a). In order to reflect the advantages of Mahalanobis distance measure in our algorithm, we compare MD-BLSSVM with BLSSVM only in this experiment. The RBF kernel is used by all these methods. In Figs. 4–7(b) and (c), black line represent classification hyper-plane. More specifically, we can draw the following conclusions:

(1) No matter how many samples and what ratio of labeled positive samples is, our approach always identifies all positive examples, i.e., the accuracy and *F* score of MD-BLSSVM is 100%. Not only that, the direction of classification hyper-plane is more suitable for the distribution of positive examples.

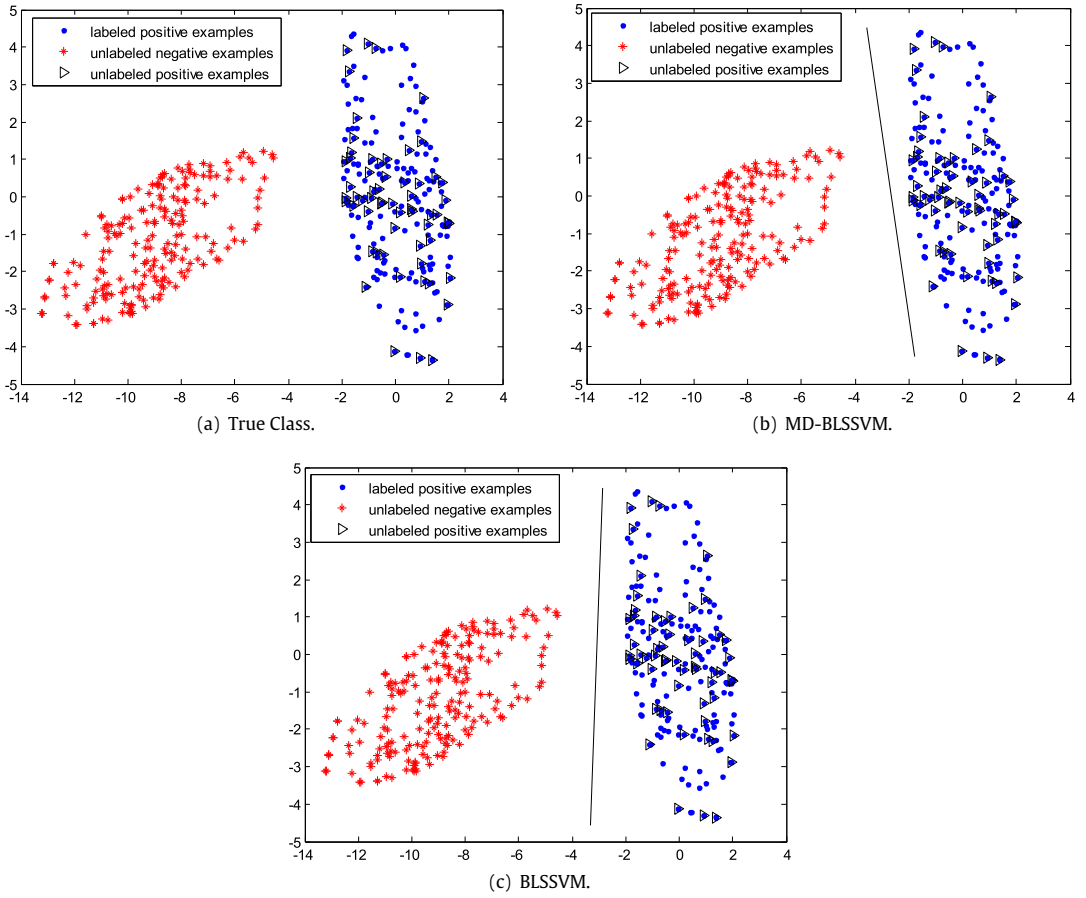


Fig. 4. Comparison results of our MD-BLSSVM and BLSSVM on artificial datasets when the number of labeled examples is 400 and the proportions of positive examples is 10%. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

(2) BLSSVM also has good classification effective and all unlabeled examples can be classified accurately. Even so, the hyperplane is disadvantageous to the direction of the data distribution, especially for the positive examples.

All in all, we summary an important conclusion, that is Mahalanobis distance measure plays a key role in classification for PU learning.

4.3.2. UCI datasets

To better illustrate the effective of MD-BLSSVM, we also provide the numerical results of MD-BLSSVM and other classical methods on six UCI datasets. More concretely, our experiments are set up in the following way: firstly, each dataset is divided into two subsets: 50% for training and 50% for testing; Then, we randomly select 10%, 20% and 30% of the training set as positive set, and use the remainder as unlabeled data; Finally, we transform them into PU tasks. Each experiment runs 10 times independently, and records the average F value and classification accuracy. Table 3 shows the details of the UCI datasets. Instance: Total number of examples, Feature: Number of features, Test: Number of test examples, Train (10%): Number of training examples, where 10% positive examples are selected as positive set and the rest as the unlabeled set, Train (20%): Number of training examples, where 20% positive examples are selected as positive set and the rest as the unlabeled set, Train (30%): Number of training examples, where 10% positive examples are selected as positive set and the rest as the unlabeled set

The values in Table 4 are the mean F value and Table 5 shows the average accuracy when 10%, 20% and 30% of the training set are selected as the labeled positive examples, respectively. The best F value and accuracy are depicted by bold figures. From these tables, we can also observe the superiority of MD-BLSSVM. Compared with six classical methods, BLSSVM, LUHC, Pulce, BSVM, NB and S-EM, MD-BLSSVM has the best performance on German, CMC and Ionosphere datasets. However, we have to point out with great regret that the performance of MD-LSSVM on Australian, Hearts and Hepatitis is unsatisfactory. In other words, BLSSVM has the best results on Australian and Hearts, while Pulce has the largest F value on Hepatitis dataset. The causes include (1): the number of Hepatitis's training samples is always less than its dimension when 10%, 20% and 30%

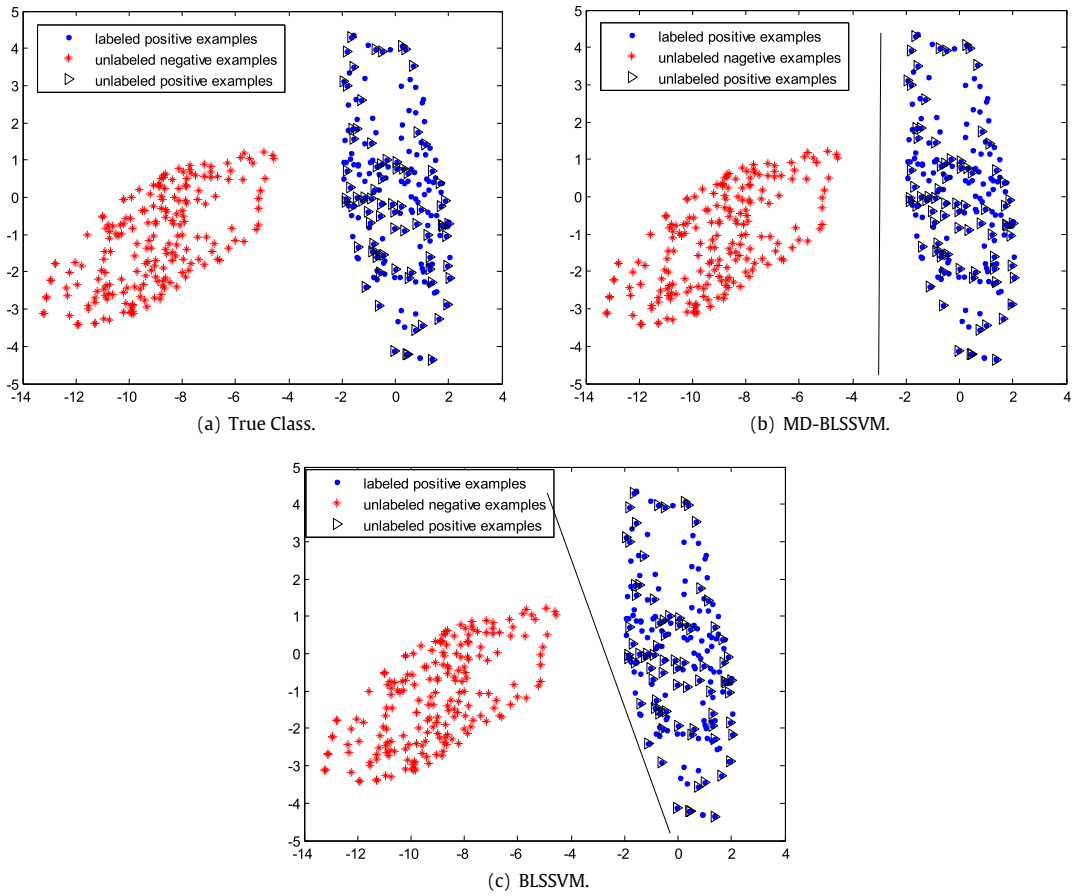


Fig. 5. Comparison results of our MD-BLSSVM and BLSSVM on artificial datasets when the number of labeled examples is 400 and the proportions of positive examples is 20%. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 3
Details of the six UCI datasets.

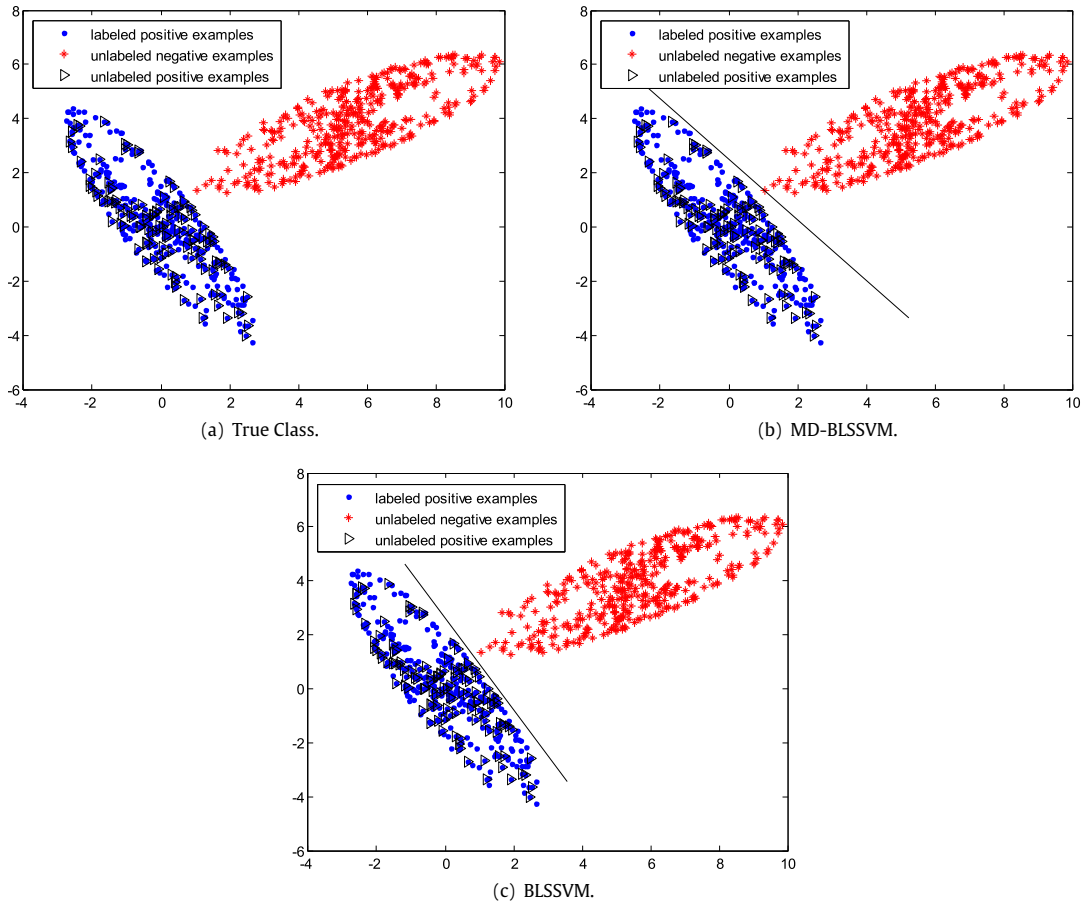
Datasets	#Instance	#Feature	#Train(10%)	#Train(20%)	#Train(30%)	#Test
German	1000	24	70/430	140/360	210/290	500
Australian	690	14	31/304	62/283	93/252	345
Hepatitis	155	19	4/74	7/71	10/68	77
Hearts	270	14	15/120	30/105	45/90	135
CMC	1473	9	114/623	228/509	342/395	736
Ionosphere	351	34	13/163	26/150	38/138	175

of the training were set as positive set on, which leads to a noninvertible covariance matrix. Obviously, the consequences of adding that disturbance factor would reduce the classification performance. (2): After careful investigation, the variance's fluctuations of each feature on these datasets lead to covariance instability slightly on training stage. That may affect the outcomes on test dataset. But anyway, MD-LSSVM is still robust in most datasets.

In order to contrast more obvious, we also provide the change trend of average F value and average classification accuracy for different ratio of labeled positive examples in Figs. 8 and 9. In Figs. 8 and 9, x-axis represents the percentage of randomly labeled positive points; y-axis is the average classification F value and accuracy. It is evident from experimental results in Figs. 8 and 9 that the performance of MD-BLSSVM is effective on most datasets. F value and accuracy reaches to 0.7–0.85. In addition, the performance of our MD-BLSSVM is more stable than other methods with the percentage of labeled positive examples changing.

4.3.3. Handwritten image

In the following, we conduct the performance evaluation on big data, USPS. The USPS [39] database consists of grayscale handwritten digit images from 0 to 9, as shown in Fig. 10. Each digit contains 1100 images, and the size of each image is 16∗16 pixels with 256 gray levels. Here we select four pair wise digits of varying difficulty for classification. Fig. 11(a–c) gives the



(a) True Class. (b) MD-BLSSVM.

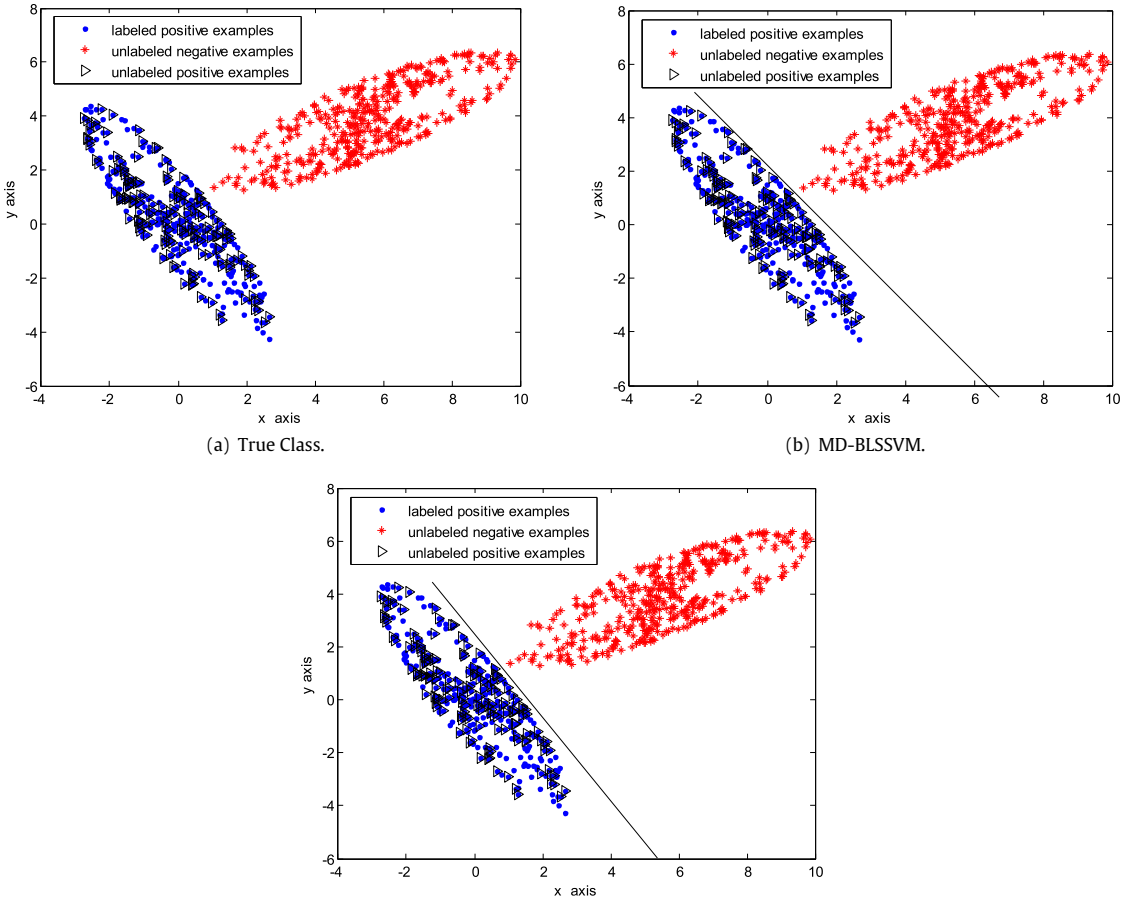
(c) BLSSVM.

Fig. 6. Comparison results of our MD-BLSSVM and BLSSVM on artificial datasets when the number of labeled examples is 800 and the proportions of positive examples is 10%. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 4
Average F value of MD-BLSSVM, BLSSVM, LUHC, Pulce, BSVM, NB and S-EM on UCI datasets at the test set with 10%, 20%, 30% of labeled positive examples.

% pos	Datasets	MD-BLSSVM	BLSSVM	LUHC	Pulce	BSVM	NB	S-EM
10%	German	**0.8245**	0.8021	0.548	0.508	0.536	0.480	0.522
20%	German	**0.8445**	0.8342	0.584	0.544	0.561	0.474	0.550
30%	German	**0.8551**	0.8527	0.577	0.555	0.519	0.486	0.562
10%	Australian	0.7235	**0.8656**	0.671	0.671	0.562	0.510	0.621
20%	Australian	0.7895	**0.8848**	0.656	0.691	0.583	0.498	0.631
30%	Australian	0.7938	**0.814**	0.663	0.701	0.607	0.547	0.644
10%	Hepatitis	0.3721	0.3529	0.481	**0.742**	0.571	0.355	0.425
20%	Hepatitis	0.4500	0.4348	0.548	**0.819**	0.514	0.449	0.532
30%	Hepatitis	0.4507	0.4167	0.620	**0.833**	0.538	0.489	0.608
10%	Hearts	0.7212	**0.8333**	0.537	0.605	0.516	0.475	0.546
20%	Hearts	0.7692	**0.8369**	0.592	0.653	0.563	0.499	0.588
30%	Hearts	0.7843	**0.8609**	0.607	0.679	0.582	0.471	0.595
10%	CMC	**0.8732**	0.8674	0.482	0.498	0.366	0.399	0.453
20%	CMC	**0.8742**	0.8716	0.514	0.536	0.479	0.384	0.527
30%	CMC	**0.8752**	0.8708	0.531	0.542	0.592	0.402	0.514
10%	Ionosphere	**0.7813**	0.7788	0.518	0.741	0.527	0.569	0.386
20%	Ionosphere	**0.8281**	0.8268	0.533	0.754	0.549	0.544	0.501
30%	Ionosphere	**0.8482**	0.831	0.531	0.776	0.523	0.539	0.584

average F value of digit 1 versus 7, 2 versus 5 and 0 versus 8 with 5%, 10%, 20%, 30%, 40% and 50% ratio of the training sets are considered as labeled positive examples, respectively. In this subsection, we choose to compare MD-BLSSVM with the approaches BLSSVM, LUHC, Pulce and BSVM. We use linear kernel for our MD-BLSSVM, LUHC and BSVM. From Fig. 11(a–c), it can be noticed that all methods exhibit an increment trend in performance when the number of labeled positive examples



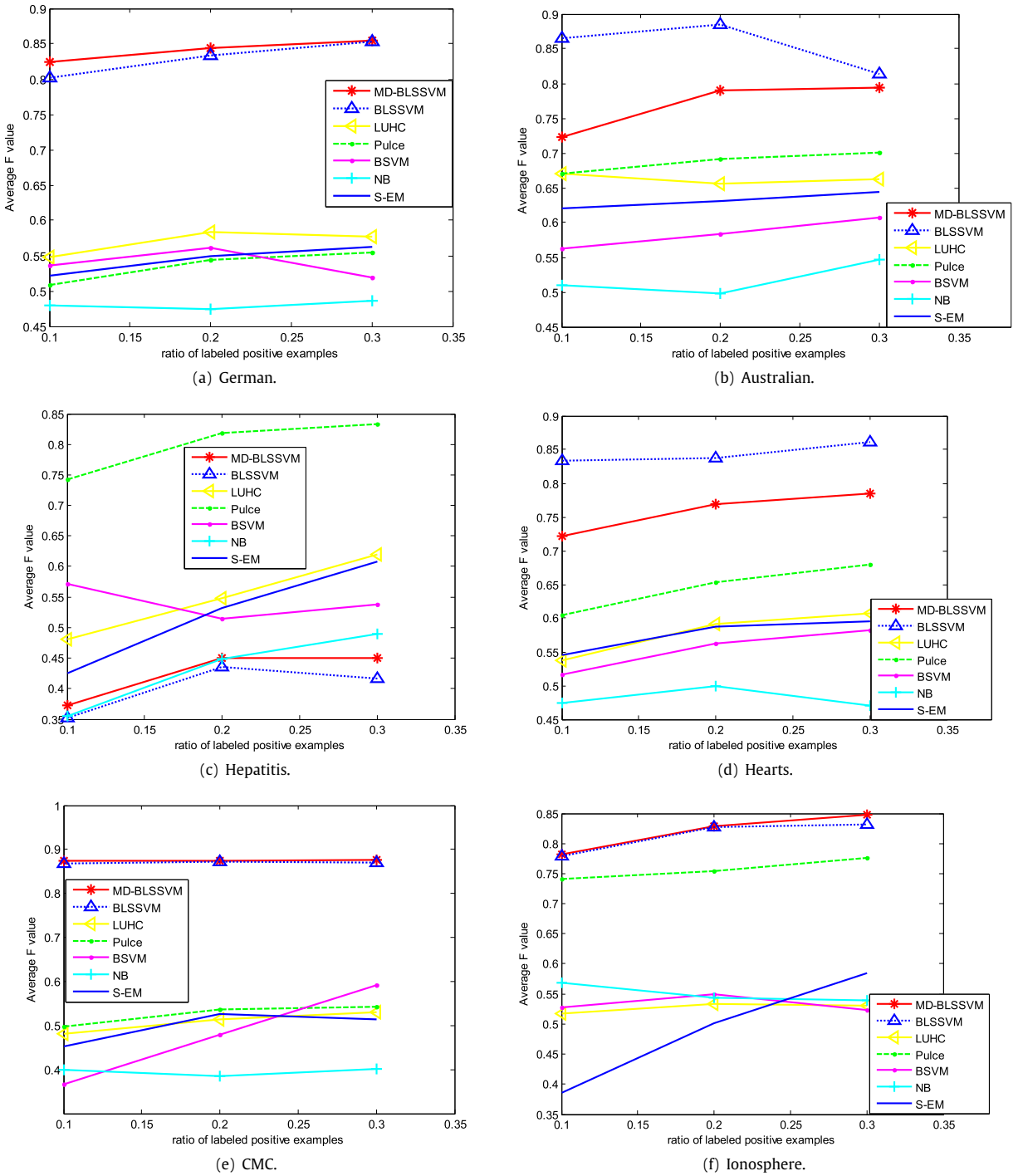
(a) True Class. (b) MD-BLSSVM.

(c) BLSSVM.

Fig. 7. Comparison results of our MD-BLSSVM and BLSSVM on artificial datasets when the number of labeled examples is 800 and the proportions of positive examples is 20%. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 5
Average accuracy of MD-BLSSVM, BLSSVM, LUHC, Pulce, BSVM, NB and S-EM on UCI datasets at the test set with 10%, 20% and 30% of labeled positive examples.

% pos	Datasets	MD-BLSSVM	BLSSVM	LUHC	Pulce	BSVM	NB	S-EM
10%	German	**0.723**	0.6998	0.683	0.627	0.662	0.588	0.668
20%	German	**0.740**	0.7128	0.698	0.686	0.646	0.568	0.674
30%	German	**0.788**	0.7541	0.692	0.692	0.654	0.588	0.703
10%	Australian	0.574	0.7265	**0.761**	0.723	0.641	0.668	0.734
20%	Australian	0.682	**0.7761**	0.758	0.764	0.674	0.635	0.764
30%	Australian	0.763	0.7654	0.763	**0.773**	0.709	0.675	0.751
10%	Hepatitis	**0.808**	0.7902	0.718	0.804	0.581	0.548	0.626
20%	Hepatitis	0.795	0.7769	0.748	**0.837**	0.686	0.603	0.696
30%	Hepatitis	0.795	0.7312	0.753	**0.858**	0.695	0.643	0.743
10%	Hearts	0.600	0.7928	**0.730**	0.623	0.659	0.626	0.663
20%	Hearts	0.637	0.8018	**0.764**	0.702	0.703	0.646	0.716
30%	Hearts	0.637	0.8269	0.752	0.724	0.711	0.697	**0.759**
10%	CMC	**0.779**	0.7643	0.632	0.622	0.522	0.539	0.579
20%	CMC	**0.779**	0.7596	0.658	0.674	0.584	0.524	0.617
30%	CMC	**0.780**	0.7761	0.664	0.677	0.602	0.561	0.636
10%	Ionosphere	0.790	0.7994	0.674	**0.795**	0.702	0.682	0.648
20%	Ionosphere	**0.875**	0.8245	0.704	0.804	0.719	0.716	0.692
30%	Ionosphere	**0.841**	0.8132	0.725	0.829	0.732	0.695	0.742



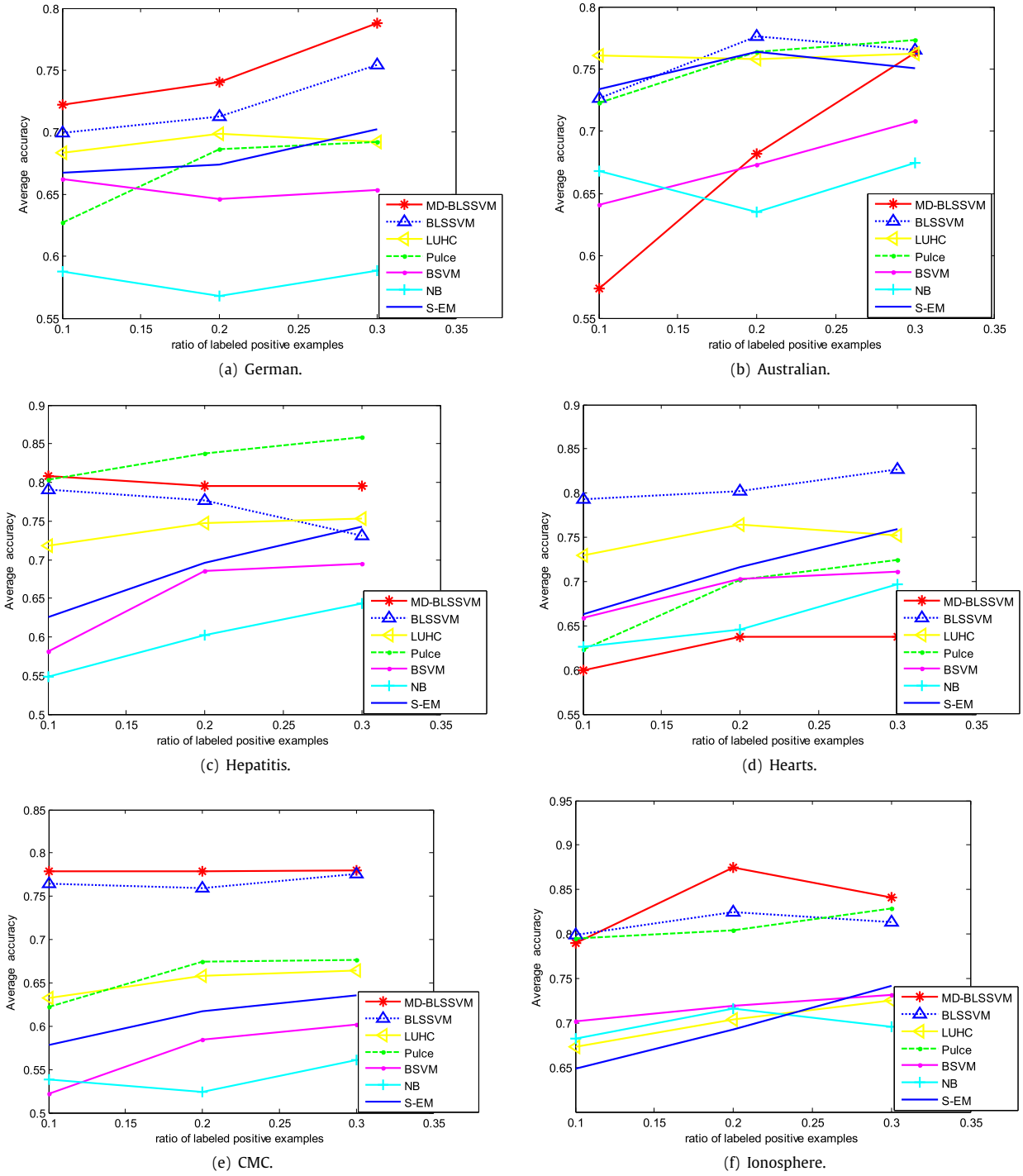
(a) German. (b) Australian.

(c) Hepatitis. (d) Hearts.

(e) CMC. (f) Ionosphere.

Fig. 8. Comparison the mean F value of our MD-BLSSVM, BLSSVM, LUHC, Pulce, BSVM, NB and S-EM on six UCI datasets.

grows from 5% to 50%. More remarkably, our MD-BLSSVM shows excellent results no matter what the ratio of labeled positive examples is. In Fig. 11(a), F value of digit 1 versus 7 is more than 0.9 whatever the ratio of labeled positive examples is.



(a) German. (b) Australian.

(c) Hepatitis. (d) Hearts.

(e) CMC. (f) Ionosphere.

Fig. 9. Comparison the mean accuracy of our MD-BLSSVM, BLSSVM, LUHC, Pulce, BSVM, NB and S-EM on six UCI datasets.

5. Conclusions

In this paper we have put forward a novel classifier based M distance, named MD-BLSSVM for short. In theory, we have proved that our MD-BLSSVM not only has very low computational time but also is more reasonable for some data distribute. Experiments on artificial datasets, six UCI datasets and handwritten image classification have shown that MD-BLSSVM is

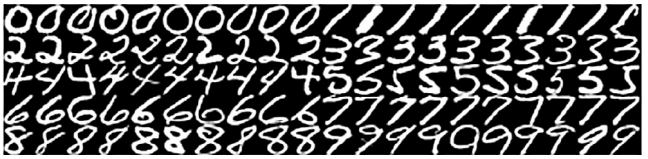
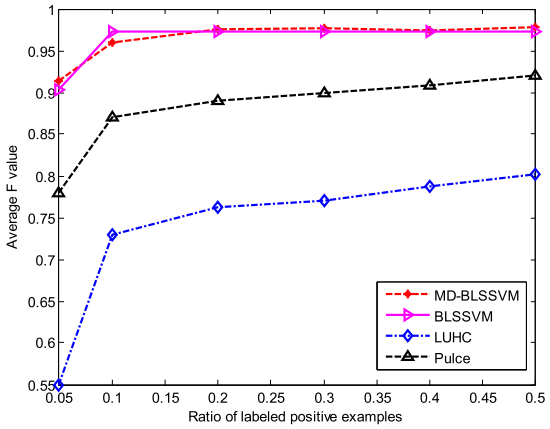
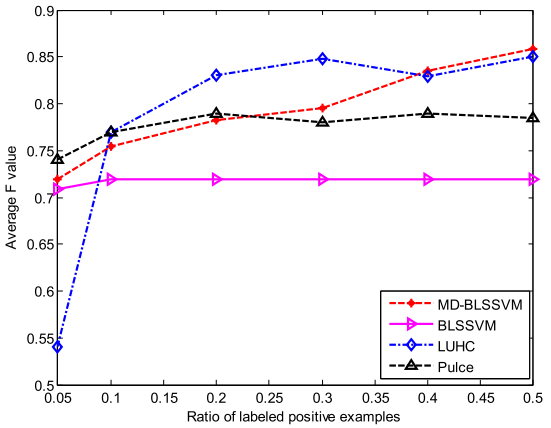


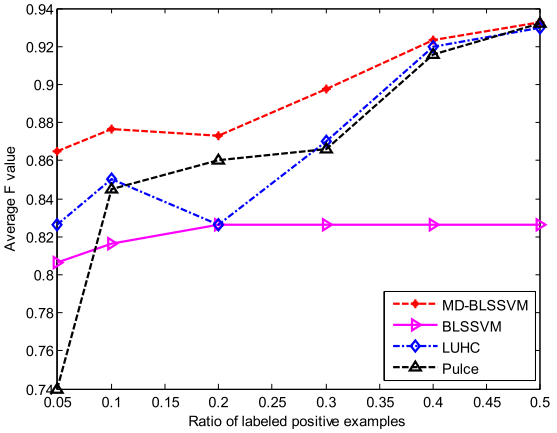
Fig. 10. Representation of Handwritten image on USPS.



(a) 1VS7.



(b) 2VS5.



(c) 8VS0.

Fig. 11. *F* values of MD-BLSSVM-lin, BLSSVM-lin, LUHC-lin, Pulce on handwritten image datasets, where *x*-axis is the ratio of positive labeled examples.

more effective than other popular method for PU learning on most datasets. In brief, MD-BLSSVM has stronger discriminative power for positive and unlabeled learning.

Acknowledgment

This work is supported by the youth innovative Foundation of Tianjin University of Science & Technology (2016LG30, 2017LG07).

References

[1] A. Fujino, N. Ueda, M. Nagata, Adaptive semi-supervised learning on labeled and unlabeled data with different distributions, Knowl. Inf. Syst. 37 (1) (2013) 129–154.
[2] F. Denis, PAC learning from positive statistical queries, Lecture Notes in Comput. Sci. 1501 (1998) 112–126.
[3] S. Muggleton, Learning from the positive data. Machine learning inductive logic programming, Lecture Notes in Comput. Sci. 1314 (1997) 358–376.
[4] H. Yu, J. Han, K.C.C. Chang, PEBL: Web page classification without negative examples, IEEE Trans. Knowl. Data Eng. 16 (1) (2004) 70–81.

[5] V.N. Vapnik, The Nature of Statistical Learning Theory, Springer Verlag, Berlin, 1995.
[6] M. Christoffe, D. Plessis, M. Sugiyama, Semi-supervised learning of class balance under class-prior change by distribution matching, Neural Netw. 50 (2014) 110–119.
[7] X.L. Li, B. Liu, Learning to classify text using positive and unlabeled data, in: Proceedings of the 18th International Joint Conference on Artificial Intelligence, Acapulco, Mexico, 2003, pp. 587–594.
[8] G.P.C. Fung, J.X. Yu, H. Lu, P.S. Yu, Text classification without negative examples revisit, IEEE Trans. Knowl. Data Eng. 18 (1) (2006) 6–20.
[9] M.N. Nguyen, X.L. Li, S.K. Ng, Positive unlabeled learning for time series classification, in: Proceedings of International Joint Conference on Artificial Intelligence, IJCAI, 2011, pp. 1421–1426.
[10] D. Ienco, R.G. Pensa, Positive and unlabeled learning in categorical data, Neurocomputing 196 (2016) 113–124.
[11] B. Liu, W.S. Lee, P.S. Yu, et al., Partially supervised classification of text documents, in: Proceedings of the 19th International Conference on Machine Learning, 2002, pp. 387–394.
[12] D. Zhang, W.S. Lee, A simple probabilistic approach to learning from positive and unlabeled examples, in: Proceedings of the 5th Annual UK Workshop on Computational Intelligence, UKCI, 2005, pp. 83–87.
[13] C. Elkan, K. Noto, Learning classifiers from only positive and unlabeled data, in: Proceedings of the 14th International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, 2008, pp. 213–220.
[14] C. Luigi, E. Charles, C. Michele, Learning gene regulatory networks from only positive and unlabeled data, Bioinformatics 11 (1) (2010) 228–240.
[15] B. Schkopf, C.P. John, S. John, J. Alex, C. Robert, Williamson, Estimating the support of a high-dimensional distribution, Neural Comput. 13 (7) (2001) 1443–1471.
[16] F. Zhu, N. Ye, W. Yu, S. Xu, G.B. Li, Boundary detection and sample reduction for one-class support vector machines, Neurocomputing 123 (2014) 166–173.
[17] B. Liu, Y. Dai, X.L. Li, W.S. Lee, P.S. Yu, Building text classifiers using positive and unlabeled examples, in: Proceedings of the 3rd IEEE International Conference on Data Mining, Melbourne, Florida, United States, 2003, pp. 179–188.
[18] W.S. Lee, B. Liu, Learning with positive and unlabeled examples using weighted logistic regression, in: Proceedings of the 20th International Conference on Machine Learning, Washington DC, United States, 2003, pp. 448–455.
[19] T. Ke, B. Yang, J.Y. Tan, L. Jing, Building high-performance classifiers on positive and unlabeled examples for text classification, Advances in Neural Networks ISNN 2012, in: Lecture notes in Computer Science, vol. 7368, 2012, pp. 187–195.
[20] T. Ke, Yang, B. Song L.J, X.B. Zhao, L. Jing, Building a biased least squares support vector machine classifier for positive and unlabeled learning, J. Softw. 9 (6) (2014) 1494–1502.
[21] Y.H. Shao, W.J. Chen, L.M. Liu, N.Y. Deng, Laplacian unit-hyper-plane learning from positive and unlabeled examples, Inform. Sci. 314 (2015) 152–1687.
[22] K. Zhou, X. Gui-Rong, Q. Yang, Y. Yu, Learning with positive and unlabeled examples using topic-sensitive. plsa, IEEE Trans. Knowl. Data Eng. (1) (2010) 46–58.
[23] N.Y. Deng, Y.J. Tian, C.H. Zhang, Support Vector Machines: Optimization-Based Theory, Algorithms, and Extensions, CRC Press, 2013.
[24] Y.H. Shao, W.J. Chen, N.Y. Deng, Nonparallel hyper-plane support-vector machine for binary classification problems, Inform. Sci. 263 (2014) 22–35.
[25] Prasanta Chandra Mahalanobis, On the generalised distance in statistics, Proc. Natl. Inst. Sci. India 2 (1) (1936) 49–55.
[26] R.D. Maesschalck, D. Jouan-Rimbaud, D.L. Massart, The Mahalanobis distance, Chemometrics Intell. Lab. Syst. 50 (2000) 1–18.
[27] R. Chatpatanasiri, T. Korsrilabutr, P. Tangchanachaianan, B. Kijsirikul, A new kernelization framework for Mahalanobis distance learning algorithms, Neurocomputing 73 (10–12) (2010) 1570–1579.
[28] B. Haasdonk, E. Pekalska, Classification with kernel Mahalanobis distance classifiers, in: Advances in Data Analysis, Data Handling and Business Intelligence Studies in Classification, Data Analysis, and Knowledge Organization, 2010, Part 5, 2010, pp. 351–361.
[29] K. Huang, H. Yang, I. King, M.R. Lyu, Maxi–min margin machine-learning large margin classifiers locally and globally, IEEE Trans. Neural Netw. 19 (2008) 260–272.
[30] C. Zhang, F. Nie, S. Xiang, A general kernelization framework for learning algorithms based on kernel PCA, Neurocomputing 4 (6) (2010) 959–967.
[31] X.J. Peng, D. Xu, Twin Mahalanobis distance-based support vector machines for pattern recognition, Inform. Sci. 16 (10) (2007) 2551–2564.
[32] Dariusz Dereniowski, Marek Kubale, Cholesky factorization of matrices in parallel and ranking of graphs, in: 5th International Conference on Parallel Processing and Applied Mathematics, vol. 3019, 2004, pp. 985–992.
[33] J.A.K. Suykens, J. Vandewalle, Least squares support vector machine classifiers, Neural Process. Lett. 9 (3) (1999) 293–300.
[34] J.A.K. Suykens, Least squares support vector machines for classification and nonlinear modeling, Neural Netw. World 10 (1–2) (2000) 29–47.
[35] J.A.K. Suykens, T.V. Gestel, J.D. Brabanter, B.D. Moor, J. Vandewalle, Least Squares Support Vector Machines, World Scientific, Singapore, 2002.
[36] C.L. Blake, C.J. Merz, UCI repository for machine learning databases, 1998. http://www.ics.uci.edu/mlearn/MLRepository.html.
[37] Z.R. Lin, LIBSVM, 2016. http://www.csie.ntu.edu.tw/~cjlin/libsvm.
[38] B. Liu, LPU package, 2008. http://www.cs.uic.edu/~liub/LPU/LPU-download.html.
[39] USPS, 1998. USPS database. http://www.cs.nyu.edu/roweis/data.html.