Kobe University Repository : Kernel

PDF issue: 2025-12-04

Unconstrained large margin distribution machines

Abe, Shigeo

(Citation)
Pattern Recognition Letters,98:96-102

(Issue Date)
2017-10

(Resource Type)
journal article

(Version)
Accepted Manuscript

(Rights)
© 2017 Elsevier B.V.
This manuscript version is made available under the CC-BY-NC-ND 4.0 license
http://creativecommons.org/licenses/by-nc-nd/4.0/

(URL)
https://hdl.handle.net/20.500.14094/90004613

Pattern Recognition Letters
journal homepage: www.elsevier.com

Unconstrained large margin distribution machines

Shigeo Abe[a,**]

[a]*Kobe University, Kobe, Japan*

ABSTRACT

Large margin distribution machines (LDMs) maximize the margin mean and minimize the margin variance, and show good generalization performance compared to support vector machines (SVMs). But because two additional hyperparameters are necessary, model selection needs more time. In this paper we propose unconstrained large margin distribution machines (ULDMs). In the ULDM, the objective function is the sum of the margin mean (a linear term), the margin variance (a quadratic term), and the weighted regularization term (a quadratic term), and constraints are not included. Therefore, the solution is expressed by a set of linear equations with one hyperparameter for the regularization term. Theoretical analysis proves that the decision boundary between two classes passes through the mean of all mapped training data if the numbers of training data of both classes are the same. The case where the numbers are different is analyzed for a one-dimensional input and how the decision boundary is determined is clarified. Using benchmark data sets, we show that the generalization performance of ULDMs is comparable to or better than that of SVMs.

© 2016 Elsevier Ltd. All rights reserved.

1. Introduction

In a support vector machine (SVM) the separating hyperplane is determined so that the minimum distance between the hyperplane and the training data are maximized [1, 2]. The distance between the separating hyperplane and the training data is called margin. The separating hyperplane thus obtained is called optimal separating hyperplane and is proved to be optimal under the assumption that the data obey unknown but fixed distribution.

If *a priori* knowledge on the distribution is available, the optimal separating hyperplane may not be optimal. For instance, if data of one class have a large variance but those of the other class have a small one, it may not be good to place the hyperplane at the equal distances from the data of opposite classes.

To cope with this problem, several approaches have been proposed. One way is to use the Mahalanobis distance, instead of the Euclidean distance. There are two ways to incorporate the Mahalanobis distance into SVMs; one is to reformulate SVMs so that the margin is measured by the Mahalanobis distance [3, 4, 5, 6], and the other is to use Mahalanobis kernels, which calculate the kernel value according to the Mahalanobis distance between the associated two argument vectors [7, 8, 9, 10, 11].

The central idea of SVMs, maximizing the minimum margin, has been influenced to improve generalization performance of other classifiers such as neural networks, fuzzy systems, and multiple classifier systems. Especially for AdaBoost [12], which is a typical multiple classifier system, the margin distribution, instead of the minimum margin, has been known to be of importance to improving the generalization ability [13, 14]. In [14], the margin mean and the margin variance are shown to be two important statistics to define the margin distribution.

There are several approaches to control the margin distribution in SVM-like classifiers [15, 16, 17, 18, 19, 20]. In [15], for each data point, a cost function is defined, in which exponential weights are assigned for misclassified data, and the objective function that is the sum of the cost functions is minimized by the gradient descent method. In [16], instead of maximizing the minimum margin, the margin mean for the training data is maximized without slack variables. In [17], game theory is used to optimize a trade-off between maximizing the minimum margin and maximizing the margin mean. In [19, 20], in addition to maximizing the margin mean, the margin variance is minimized and the classifier is called large margin distribution

**Corresponding author: Tel.: +81-78-994-3432; fax: +81-78-994-3432;
e-mail: abe@kobe-u.ac.jp (Shigeo Abe)

machine (LDM). The LDM is formulated based on the soft margin SVM, namely the slack variables are introduced to penalize the training data that are misclassified. Thus, two additional hyperparameters are introduced compared to the SVM. This is a serious problem in model selection. According to the computer experiments in [19], the generalization ability of the LDM is the best among the SVM and the classifiers proposed in [15, 16, 17] for the benchmark data sets tested.

In this paper, we reformulate the LDM and propose a unconstrained LDM (ULDM), whose number of hyperparameters is the same as that of SVM. As in the LDM, the margin mean is maximized and the margin variance is minimized. Because the objective function is expressed by the weighted sum of the margin mean in a linear term and the margin variance in a quadratic term, the solution is given by solving a set of linear equations and the weight works as a scale factor of the hyperplane. Therefore, the change of the weight value does not change the decision boundary. Thus, determination of the weight value is not necessary. To control the generalization ability of the ULDM, we introduce a quadratic regularization term. Thus, one hyperparameter associated with the regularization term is introduced into the ULDM.

To clarify the characteristics of the ULDM, we analyze the decision boundary when the numbers of training data of both classes are the same. For the case where the numbers are different, we analyze the decision boundary for a classification problem with one-dimensional input.

We evaluate the proposed ULDM for two-class and multiclass problems and compare performance of the ULDM with that of the SVM.

In Section 2, we summarize L1 SVMs, LS (least squares) SVMs, and LDMs. And in Section 3, we propose variants of LDMs: LS LDMs and ULDMs, and discuss the decision boundary of ULDMs. In Section 4, we compare performance of the proposed ULDMs with that of L1 and LS SVMs using two-class and multiclass problems.

2. Support Vector Machines and Large Margin Distribution Machines

In this section we summarize the L1 SVM, one of its variants, the LS SVM, and the LDM.

2.1. L1 Support Vector Machines

In the SVM, nonlinear separation is realized by using the nonlinear vector function $\boldsymbol{\phi}(\mathbf{x})$ that maps the m-dimensional input vector $\mathbf{x}$ into the l-dimensional feature space. In the feature space, we determine the decision function that separates Class 1 data from Class 2 data:

$$f(\mathbf{x}) = \mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x}) + b, \tag{1}$$

where $\mathbf{w}$ is the l-dimensional vector, $\top$ denotes the transpose of a vector (matrix), and b is the bias term.

Assume that we have M training input-output pairs $\{\mathbf{x}_i, y_i\}$ $(i = 1, \ldots, M)$, where $\mathbf{x}_i$ are m-dimensional training inputs and belong to Class 1 or 2 and the associated labels are $y_i = 1$ for Class 1 and -1 for Class 2. The margin of $\mathbf{x}_i$, δ_i, which is the distance from the hyperplane, is given by

$$\delta_i = y_i\, f(\mathbf{x}_i)/\|\mathbf{w}\|. \tag{2}$$

Let δ be the minimum margin among δ_i $(i = 1, \ldots, M)$, and assume that $\delta\,\|\mathbf{w}\| = 1$. Then maximizing δ is equivalent to minimizing $\|\mathbf{w}\|$, and $\mathbf{x}_i$ must satisfy $y_i\, f(\mathbf{x}_i) \geq 1$ so that the margin for $\mathbf{x}_i$ is larger than or equal to 1. Allowing misclassification, the L1 SVM is formulated in the primal form as follows:

$$\text{minimize} \quad Q(\mathbf{w}, b, \boldsymbol{\xi}) = \frac{1}{2}\mathbf{w}^\top \mathbf{w} + C \sum_{i=1}^{M} \xi_i \tag{3}$$

$$\text{subject to} \quad y_i\, f(\mathbf{x}_i) \geq 1 - \xi_i \quad \text{for} \quad i = 1, \ldots, M, \tag{4}$$

where $Q(\mathbf{w}, b, \boldsymbol{\xi})$ is the objective function, C is the margin parameter that controls the trade-off between the training error and the generalization ability, $\xi_i\ (\geq 0)$ are the slack variables for $\mathbf{x}_i$, and $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_M)^\top$.

Usually we solve the following dual form, instead of solving (3) and (4):

$$\text{maximize} \quad Q(\boldsymbol{\alpha}) = \sum_{i=1}^{M} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{M} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{5}$$

$$\text{subject to} \quad \sum_{i=1}^{M} y_i\, \alpha_i = 0,\ 0 \leq \alpha_i \leq C \ \text{for} \quad i = 1, ..., M, \tag{6}$$

where α_i are Lagrange multipliers associated with $\mathbf{x}_i$ and $K(\mathbf{x}, \mathbf{x}') = \boldsymbol{\phi}^\top(\mathbf{x})\, \boldsymbol{\phi}(\mathbf{x}')$ is a kernel function. Using a kernel function, we can avoid treating the feature space directly.

One of the advantages of the L1 SVM is that the solution is sparse, i.e., some of α_i are zero and only non-zero α_i are necessary to represent the solution.

We use the following radial basis function (RBF) kernels, which are often used for pattern classification:

$$K(\mathbf{x}, \mathbf{x}') = \exp(-\gamma \|\mathbf{x} - \mathbf{x}'\|^2 / m), \tag{7}$$

where m is the number of inputs for normalization and γ is to control a spread of a radius.

2.2. Least Squares Support Vector Machines

The LS SVM [21] is a variant of the L1 SVM. The linear sum of slack variables in (3) is changed to the square sum and the inequality constraints in (4) are changed to equality constraints as follows:

$$\text{minimize} \quad Q(\mathbf{w}, b, \boldsymbol{\xi}) = \frac{1}{2}\mathbf{w}^\top \mathbf{w} + \frac{C}{2} \sum_{i=1}^{M} \xi_i^2 \tag{8}$$

$$\text{subject to} \quad y_i\, f(\mathbf{x}_i) = 1 - \xi_i \qquad \text{for} \ \ i = 1, \ldots, M. \tag{9}$$

Solving the equation in (9) for ξ_i and substituting it to the objective function in (8), we obtain the unconstrained optimization problem:

$$\text{minimize} \quad Q(\mathbf{w}, b) \quad = \quad \frac{1}{2}\mathbf{w}^\top \mathbf{w} + \frac{C}{2} \sum_{i=1}^{M} (1 - y_i\, f(\mathbf{x}_i))^2. \tag{10}$$

The solution of the LS SVM can be obtained by solving a set of linear equations and generalization performance is known to be comparable to the L1 SVM [2]. One of the disadvantages of the LS SVM is that unlike the L1 SVM, the solution is not sparse.

2.3. Large Margin Distribution Machines

In the LDM, the margin mean is maximized and the margin variance is minimized.

As in [19], instead of (2), we consider the following relation as the margin:

$$\delta_i = y_i\, f(\mathbf{x}_i). \tag{11}$$

Then the margin mean $\bar{\delta}$ and margin variance $\hat{\delta}$ are given, respectively, by

$$\bar{\delta} = \frac{1}{M}\sum_{i=1}^{M}\delta_i, \tag{12}$$

$$\hat{\delta} = \frac{1}{M}\sum_{i=1}^{M}\left(\delta_i - \bar{\delta}\right)^2 = \frac{1}{M}\sum_{i=1}^{M}\delta_i^2 - \bar{\delta}^2. \tag{13}$$

According to [19], the LDM is formulated based on the L1 SVM given by (3) and (4) as follows:

$$\text{minimize } Q(\mathbf{w}, b, \boldsymbol{\xi}) = \frac{1}{2}\mathbf{w}^\top\mathbf{w} - \lambda_1\,\bar{\delta} + \frac{1}{2}\lambda_2\,\hat{\delta} + C\sum_{i=1}^{M}\xi_i \tag{14}$$

$$\text{subject to } y_i\, f(\mathbf{x}_i) \geq 1 - \xi_i \qquad \text{for } i = 1, \ldots, M, \tag{15}$$

where λ_1 and λ_2 are parameters to control maximization of the margin mean and minimization of the margin variance, respectively. Compared with the L1 SVM, in the objective function, the second and the third terms are added.

According to the computer experiments in [19], the LDM is statistically better than or comparable to that of the L1 SVM. However, the LDM is at a disadvantage in sparsity and model selection. Because in the LDM, all the training data are used in calculating the margin mean and the margin variance, the solution is dense. In addition, because four parameter values (including one kernel parameter value), instead of two, need to be determined by model selection, model selection requires more time.

3. Variants of Large Margin Distribution Machines

In this section, first we formulate the LS LDM instead of the L1 SVM based LDM and then reformulate the LDM to reduce the number of hyperparameters.

3.1. Least Squares Large Margin Distribution Machines

Similar to the LDM, the LS LDM that maximizes the margin mean and minimizes the margin variance is given by

$$\text{minimize } Q(\mathbf{w}, b, \boldsymbol{\xi}) = \frac{1}{2}\mathbf{w}^\top\mathbf{w} - \lambda_1\,\bar{\delta} + \frac{1}{2}\lambda_2\,\hat{\delta} + \frac{C}{2}\sum_{i=1}^{M}\xi_i^2 \tag{16}$$

$$\text{subject to } y_i\, f(\mathbf{x}_i) = 1 - \xi_i \qquad \text{for } i = 1, \ldots, M. \tag{17}$$

Solving the equation in (17) for ξ_i and substituting it to the objective function in (16), we obtain

$$Q(\mathbf{w}, b) = \frac{1}{2}\mathbf{w}^\top\mathbf{w} - \lambda_1\,\bar{\delta} + \frac{1}{2}\lambda_2\,\hat{\delta} + \frac{C}{2}\sum_{i=1}^{M}(\delta_i - 1)^2. \tag{18}$$

In the above objective function, the last term is the variance of margins around 1, i.e., around the minimum margin. Therefore, it works similarly to the variance of margin around the margin mean, $\hat{\delta}$. Similarly, the first term is the reciprocal of the square minimum margin and the second term is the margin mean. Therefore, by this formulation, maximization of the minimum margin and the margin mean and minimization of the variances around 1 and around the margin mean are mixed and their trade off is determined by selecting the parameter values of λ_1, λ_2, and C. This is not a good formulation because very similar measures are mixed in the model. Therefore, in the following we consider a simpler model.

3.2. Unconstrained Large Margin Distribution Machines

Deleting the first and the fourth terms in (18), we obtain:

$$\text{minimize} \quad Q(\mathbf{w}, b) = -\bar{\delta} + \frac{1}{2}\lambda_0\,\hat{\delta}, \tag{19}$$

where λ_0 is a trade-off parameter between maximizing the margin mean and minimizing the margin variance.

As will be shown later, because $\bar{\delta}$ is linear and $\hat{\delta}$ is quadratic, λ_0 works as a scale factor of the decision function and does not control the generalization ability. Thus to control the generalization ability, we add a regularization term to (19). But to do this, first we derive the solution of (19).

Because (19) is a quadratic optimization problem without inequality or equality constraints, it can be trained by solving a set of linear equations.

The coefficient vector $\mathbf{w}$ is in the empirical feature space spanned by $\{\boldsymbol{\phi}(\mathbf{x}_1), \ldots, \boldsymbol{\phi}(\mathbf{x}_M)\}$ [2]. Let $\{\mathbf{z}_1, \ldots, \mathbf{z}_N\}$ be a subset of $\{\mathbf{x}_1, \ldots, \mathbf{x}_M\}$, where $N \leq M$ and let $\{\boldsymbol{\phi}(\mathbf{z}_1), \ldots, \boldsymbol{\phi}(\mathbf{z}_N)\}$ span the empirical feature space. Then, $\mathbf{w}$ can be expressed by

$$\mathbf{w} = \sum_{i=1}^{N}\alpha_i\,\boldsymbol{\phi}(\mathbf{z}_i), \tag{20}$$

where α_i are constants. If a linear kernel is used, and if $m < N$, to improve sparsity, we use the following Euclidean coordinates:

$$\begin{aligned} \mathbf{z}_1 &= \{1, 0, \ldots, 0\} \\ &\vdots \\ \mathbf{z}_m &= \{0, \cdots, 0, 1\}, \end{aligned} \tag{21}$$

and use the identity mapping: $\boldsymbol{\phi}(\mathbf{x}) = \mathbf{x}$.

Substituting (20) into (11), we obtain

$$\delta_i = y_i\left(\sum_{j=1}^{N}\alpha_j\,K_{ij} + b\right), \tag{22}$$

where $K_{ij} = K(\mathbf{x}_i, \mathbf{z}_j) = \boldsymbol{\phi}^\top(\mathbf{x}_i)\,\boldsymbol{\phi}(\mathbf{z}_j)$ and $K(\mathbf{x}, \mathbf{z})$ is a kernel function. Expressing (22) in a vector form, we obtain

$$\delta_i = y_i\,(K_i\,\boldsymbol{\alpha} + b) = y_i\begin{pmatrix} K_i & 1 \end{pmatrix}\boldsymbol{\beta}, \tag{23}$$

where $K_i = (K_{i1}, \ldots, K_{iN})$, $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_N)^\top$, and $\boldsymbol{\beta} = (\boldsymbol{\alpha}^\top\ b)^\top$.

Then, substituting (23) into (12) gives

$$\bar{\delta} = \frac{1}{M}\sum_{i=1}^{M} y_i \begin{pmatrix} K_i & 1 \end{pmatrix} \boldsymbol{\beta} = \begin{pmatrix} \overline{K^y} & \bar{y} \end{pmatrix} \boldsymbol{\beta} = \mathbf{h}\boldsymbol{\beta}, \tag{24}$$

where

$$\mathbf{h} = \begin{pmatrix} \overline{K^y} & \bar{y} \end{pmatrix}, \quad \overline{K^y} = \frac{1}{M}\sum_{i=1}^{M} y_i K_i, \quad \bar{y} = \frac{1}{M}\sum_{i=1}^{M} y_i. \tag{25}$$

Substituting (23) and (24) into (13),

$$\hat{\delta} = \boldsymbol{\beta}^\top K \boldsymbol{\beta} \tag{26}$$

where

$$K = \begin{pmatrix} \overline{K^2} - \overline{K^y}^\top \overline{K^y} & \bar{K}^\top - \bar{y}\,\overline{K^y}^\top \\ \bar{K} - \bar{y}\,\overline{K^y} & 1 - \bar{y}^2 \end{pmatrix}, \tag{27}$$

$$\overline{K^2} = \frac{1}{M}\sum_{i=1}^{M} K_i^\top K_i, \qquad \bar{K} = \frac{1}{M}\sum_{i=1}^{M} K_i. \tag{28}$$

Matrix K is positive semi-definite because $\hat{\delta} \geq 0$.

To regularize the solution of (19) we add a regularization term as follows:

$$\text{minimize} \quad Q(\boldsymbol{\beta}) = \frac{1}{2} C \boldsymbol{\beta}^\top \boldsymbol{\beta} - \bar{\delta} + \frac{1}{2}\lambda_0 \hat{\delta}, \tag{29}$$

$$= \frac{1}{2}\boldsymbol{\beta}^\top (C\, I_N + \lambda_0 K)\boldsymbol{\beta} - \mathbf{h}\boldsymbol{\beta}, \tag{30}$$

where $C\,(> 0)$ is a margin parameter and I_N is the $N \times N$ unit matrix. We use the same C as in L1 and LS SVMs. In the above formulation, as C approaches zero, the accuracy for the training data set increases and as C becomes large, the accuracy decreases. This is opposite to that in L1 and LS SVMs.

In Section 4, to normalize the C value, we use

$$C \max_{i=1,\ldots,N} K_{ii} I_N \tag{31}$$

instead of CI_N in (29).

By the introduction of the regularization term, the above optimization problem always has the optimal solution by solving

$$(C\, I_N + \lambda_0 K)\boldsymbol{\beta} = \mathbf{h}^\top, \tag{32}$$

namely

$$\boldsymbol{\beta} = (C\, I_N + \lambda_0 K)^{-1}\mathbf{h}^\top. \tag{33}$$

If we multiply the positive parameter to solution $\boldsymbol{\beta}$, the separating hyperplane $f(\mathbf{x}) = 0$ does not change. Therefore, if $C = 0$, the solution does not change even if we change the λ_0 value. In addition, the solutions are the same if we change parameter values fixing the ratio of C and λ_0 values. Therefore, we can set

$$\lambda_0 = 1. \tag{34}$$

Substituting (34) into (30), we obtain

$$\text{minimize} \quad Q(\boldsymbol{\beta}) = \frac{1}{2}\boldsymbol{\beta}^\top (C\, I_N + K)\boldsymbol{\beta} - \mathbf{h}\boldsymbol{\beta}. \tag{35}$$

We call (35) unconstrained LDM (ULDM).

If we use the Euclidean coordinates given by (21) and the identity mapping function, $K_i = \mathbf{x}^\top$, this is equivalent to solving (33) in the input space and $\boldsymbol{\alpha} = \mathbf{w}$.

If N is small we can solve (33) by matrix inversion. If N is large, we can solve it by the coordinate descent method [22] because the coefficient matrix is positive definite.

The procedure of training the ULDM is as follows:

1. Calculate K in (27) and $\mathbf{h}$ in (25).
2. Setting $\lambda_0 = 1$, solve (32) for $\boldsymbol{\beta}$ either by matrix inversion as in (33) or by the coordinate descent method.
3. Obtain the decision function (1) substituting b in $\boldsymbol{\beta}$ and $\mathbf{w}$ given by (20) with α_i in $\boldsymbol{\beta}$ into (1).

3.3. *Analysis of ULDMs*

We analyze characteristics of the solution of the ULDM.

3.3.1. *Relations between ULDMs and LDMs*

We consider the relations between the ULDM given by (35) and the LDM given by (14) and (15).

In the LDM, the bias term is not included [19, 20]. Usually, kernel functions include a constant term, which works to generate an implicit bias term. Even if they do not, we can easily add a constant term [2]. Although the implicit bias term is not equivalent to the explicit bias term, in that they give the same solution, the difference will not make a large difference in the generalization performance.

In the ULDM given by (35), because the value of C is set to a small value as will be discussed in the following section, the solution is determined by the second and third terms.

The LDM given by (14) and (15) is the combination of the L1 SVM and the ULDM; If $\lambda_1 = 0$ and $\lambda_2 = 0$, the LDM is equivalent to the L1 SVM, and the second and the third terms are the same as the major terms of the ULDM.

Therefore, as the values of λ_1 and λ_2 approach 0, the solution of the LDM becomes similar to that of the L1 SVM. Likewise, as the values of λ_1 and λ_2 approach ∞, the solution becomes similar to that of the ULDM. When λ_1 and λ_2 have values no so large and not so small, the solution is a combination of those of the L1 SVM and the ULDM.

Thus, the performance difference between the ULDM and LDM is considered to be small.

3.3.2. *Analysis for $\bar{y} = 0$*

For $\bar{y} = 0$ the following two theorems hold.

Theorem 1. *If K is nonsingular and $\bar{y} = 0$, the decision boundary of the ULDM passes through $\bar{K}^\top/(1 + C)$. Furthermore, if $C = 0$, the decision boundary goes through the center of the mapped training data in the empirical feature space.*

Proof. From (32) the solution satisfies

$$\left(\overline{K^2} - \overline{K^y}^\top \overline{K^y} + C\right)\boldsymbol{\alpha} + \left(\bar{K}^\top - \bar{y}\,\overline{K^y}^\top\right) b = \overline{K^y}^\top \tag{36}$$

$$\left(\bar{K} - \bar{y}\,\overline{K^y}\right)\boldsymbol{\alpha} + (1 - \bar{y}^2 + C)\, b = \bar{y}. \tag{37}$$

From (37),

$$b = \frac{\bar{y} - \left(\bar{K} - \bar{y}\,\overline{K^y}\right)\boldsymbol{\alpha}}{1 - \bar{y}^2 + C}. \tag{38}$$

If the numbers of training data for Classes 1 and 2 are the same, i.e., $\bar{y} = 0$,

$$b = -\frac{\bar{K}}{1 + C}\,\boldsymbol{\alpha}. \tag{39}$$

If K is non-singular, the set of linear equations given by (36) and (37) has a unique solution. Therefore, the solution satisfies (39). The decision boundary is given by

$$\boldsymbol{\alpha}^\top \boldsymbol{\phi} + b = \mathbf{0}, \tag{40}$$

where $\boldsymbol{\phi} = (\phi^\top(\mathbf{z}_1), \ldots, \phi^\top(\mathbf{z}_N))^\top$. Substituting (39) into (40),

$$\boldsymbol{\alpha}^\top \left(\boldsymbol{\phi} - \frac{\bar{K}^\top}{1 + C}\right) = \mathbf{0}. \tag{41}$$

Because $\boldsymbol{\alpha}$ is a unique, non-zero vector, the decision boundary passes through $\bar{K}^\top/(1 + C)$. If $C = 0$, it passes through the center vector of the mapped training data in the empirical feature space. ■

Theorem 2. *For $\bar{y} = 0$, $\overline{K^y}\,\boldsymbol{\alpha} \geq 0$ is satisfied, where the equality holds when $C = 0$ and $\boldsymbol{\alpha}$ is in the null space of $\overline{K^y}$.*

Proof. Because $\bar{y} = 0$, $\bar{K}$ is the mean of Classes 1 and 2 means.

Multiplying $\boldsymbol{\beta}^\top$ from the left-hand side of (32) with (34), we obtain

$$\boldsymbol{\beta}^\top (K + C\,I_N)\boldsymbol{\beta} = \mathbf{h}\boldsymbol{\beta}. \tag{42}$$

Because K is positive semi-definite and $K + C\,I_N$ is positive definite for $C > 0$,

$$\mathbf{h}\boldsymbol{\beta} = \overline{K^y}\,\boldsymbol{\alpha} + \bar{y}\,b = \overline{K^y}\,\boldsymbol{\alpha} \geq 0. \tag{43}$$

The equality holds when $C = 0$ and $\boldsymbol{\alpha}$ is in the null space of $\overline{K^y}$. ■

We define the means of the mapped training data for Classes 1 and 2, respectively, by

$$\bar{K}_+ = \frac{1}{M_+}\sum_{i=1, y_i=1}^{M} K_i, \quad \bar{K}_- = \frac{1}{M_-}\sum_{i=1, y_i=-1}^{M} K_i, \tag{44}$$

where M_+ and M_- are respectively the numbers of training data for Classes 1 and 2, and $M_+ = M_-$. Because $\overline{K^y} = (\bar{K}_+ - \bar{K}_-)/2$, $\overline{K^y}$ is the vector from $\bar{K}_-/2$ to $\bar{K}_+/2$. Therefore, the above theorem means that the angle between $\overline{K^y}$ and $\boldsymbol{\alpha}$ is less than $\pi/2$.

Example 1. We show the change of the decision boundary for the change of the C value using the iris data set, which consists of 150 data with 3 classes and 4 features [23]. For the illustration purpose, we use Classes 2 and 3 and Features 3 and 4. For each class the first 25 data are used for training and the remaining data are used for testing. We use linear kernels with the Euclidean coordinates given by (21).

Figure 1 shows the decision boundaries for $C = 10^{-6}, 0.1, 1$. "Mean" denotes the line segment connecting the means for Classes 2 and 3 with the mean $\bar{K}$ in between. As Theorem 1 shows, for $C = 10^{-6}$, which is near zero, the decision boundary passes through $\bar{K}$. The angle between $\boldsymbol{\alpha}$, which is orthogonal to the decision function, and $\overline{K^y}$ is smaller than 90 degrees, which exemplifies Theorem 2.

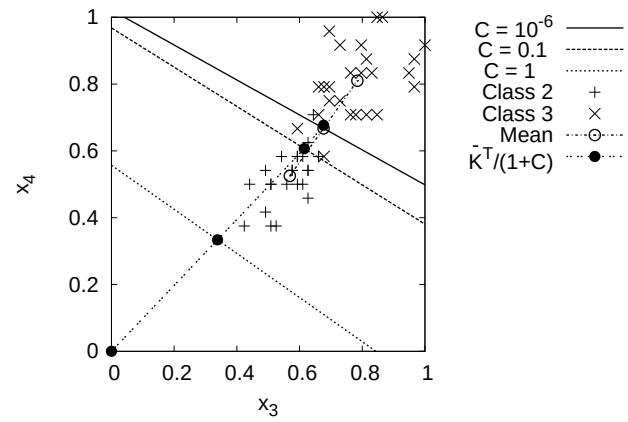

Fig. 1. Decision boundaries for the iris data set

In accordance with Theorem 1, the decision boundary goes through $\bar{K}/(1 + C)$ as the C value is changed to 0.1 and 1. For $C = 1$, all the training data are classified into Class 2. Therefore, in the iris data set, a large C value is inadequate. The recognition rates for the test data are 94% (3 data misclassified), 92% (4 data), and 50% (25 data) for $C = 10^{-6}, 0.1$, and 1, respectively.

3.3.3. Analysis for a one-dimensional case

We discuss the case where $\bar{y} \neq 0$. To make discussions simpler, we consider a one-dimensional case in the input space. We assume that the decision function is given by

$$f(x) = x + b. \tag{45}$$

This implies that the region for Class 1 satisfies $x + b > 0$.

The margin for x_i is given by $\delta_i = y_i\,(x_i + b)$ and the optimal b satisfies

$$\frac{\mathrm{d}\,Q}{\mathrm{d}\,b} = -\bar{y} + \bar{K} - \bar{y}\,\overline{K^y} + (1 - \bar{y}^2 + C)\,b = 0 \tag{46}$$

Therefore, b is given by

$$b = -\frac{\bar{K} - \bar{y}\,\overline{K^y} - \bar{y}}{1 - \bar{y}^2 + C}. \tag{47}$$

Solving $x + b = 0$ for x, the boundary is given by

$$x = \frac{\bar{K} - \bar{y}\,\overline{K^y} - \bar{y}}{1 - \bar{y}^2 + C}. \tag{48}$$

As shown in Theorem 1, for $\bar{y} = 0$, $x = \bar{K}/(1 + C)$. Assume that $\bar{y} \neq 0$ and the numerator of (48) is not zero. Then, as the value of C is changed from 0 to ∞, the decision boundary goes to zero. Suppose the input of the training data are scaled into

a non-negative region. Then, as the value of C becomes larger, data tend to be classified into the single class. Therefore, the value of C needs to be small.

In the following we analyze the boundary for $C = 0$, i.e.,

$$x = \frac{\bar{K} - \bar{y}\overline{K^y}}{1 - \bar{y}^2} - \frac{\bar{y}}{1 - \bar{y}^2}. \tag{49}$$

The first term in (49) reduces to

$$\begin{aligned} \frac{\bar{K} - \bar{y}\overline{K^y}}{1 - \bar{y}^2} &= \frac{1}{1+\bar{y}}\frac{M_+}{M}\bar{K}_+ + \frac{1}{1-\bar{y}}\frac{M_-}{M}\bar{K}_- \\ &= \frac{1}{2}(\bar{K}_+ + \bar{K}_-), \end{aligned} \tag{50}$$

which implies that the first term is the mean of the Classes 1 and 2 means. If the distribution of Class 1 is larger than that of Class 2, the boundary is nearer to Class 1, irrespective of the numbers of data for each class.

The second term in (49) is negative if $M_+ > M_-$, and non-negative, otherwise. Therefore, if $M_+ > M_-$, the boundary x shifts towards $\bar{K}_-$.

According to the above analysis, the first terms in (49) set the boundary in the middle of the means for two classes. The second term works to shift the boundary towards $\bar{K}_-$ for $M_+ > M_-$; the increase of the Class 1 data works to widen the Class 1 region.

3.3.4. Analysis for a general case

It is difficult to analyze the general case where $\bar{y} \neq 0$ theoretically. In the following example, we show the change of the class boundary for the change of the C value using the iris data used in Example 1. The difference of the data set is that in the following example, only the first 10 training data are used for Class 3. Other conditions are the same as those in Example 1.

Example 2. Figure 2 shows the decision boundaries for $C = 10^{-6}, 10^{-2}, 0.1$. Unlike for $\bar{y} = 0$, the decision boundary for $C = 10^{-6}$ does not pass through the average of Classes 2 and 3 means. As the value of C is increased, the decision boundary moves upward, which is opposite to that for $\bar{y} = 0$.

For $C = 0.1$, all the training data are classified into Class 2. Therefore, in this case also a large C value is inadequate. The recognition rates for the test data set are 90% (5 data misclassified), 90% (5 data), and 50% (25 data) for $C = 10^{-6}, 10^{-2}$ and 0.1, respectively.

4. Performance Evaluation

According to the computer experiments in [19], LDM performed best among SVM, MDO [15], MAMC [16], and KM-OMD [17]. From the standpoint of the average accuracy, SVM performed the second best. Therefore, in this experiment, we compare the ULDM with the L1 SVM and the LS SVM, whose architecture is similar to that of the ULDM using two-class and multiclass problems. Because most of the benchmark data sets used in [19] are not publicly available, we compared the ULDM with the LDM using some of the data sets that, we consider, are the same as those used in [19].

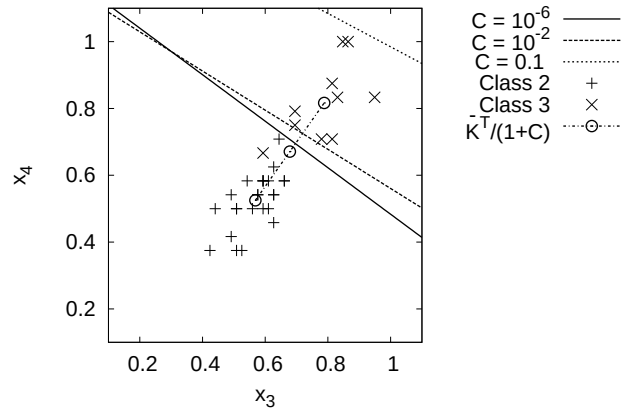

Fig. 2. Decision boundaries for the iris data set with reduced training data

4.1. Two-class Problems

The two-class data sets [24] used in our study is listed in Table 1. In each row of the "Data" column, the numerals in the parentheses list the numbers of inputs, training data, test data, and data set pairs of a classification problem. Each data set pair consists of the training data set and the test data set. We trained classifiers using the training data set and evaluated the performance using the test data set. Then we calculated the average accuracy and the standard deviation for all the data set pairs. We determined the parameter values by fivefold cross-validation. We selected the γ value from {0.01, 0.1, 0.5, 1, 5, 10, 15, 20, 50, 100, 200}. For the C value, we selected from {0.1, 1, 10, 50, 100, 500, 1000, 2000} for the L1 and LS SVMs, and for the ULDM, from $\{10^{-12}, 10^{-10}, 10^{-8}, 10^{-6}, 10^{-4}, 10^{-3}, 10^{-2}, 0.1\}$.

For the ULDM, instead of selecting the independent data, we used all the training data, namely N in (20) is M.

We trained the ULDM and LS SVM by matrix inversion. For the L1 SVM, we used SMO-NM [25], which fuses SMO (Sequential minimal optimization) and NM (Newton's method).

Tables 1 shows the average accuracies and their standard deviations of the three classifiers.

Among three classifiers the best average accuracy is shown in bold and the worst average accuracy is underlined. The "Average" row shows the average accuracy of the 13 average accuracies and the three numerals in the parentheses show the numbers of the best, the second best, and worst accuracies in the order. We performed Welch's t test with the confidence intervals of 95%. The "W/T/L" row shows the results; W, T, and L denote the numbers that the ULDM shows statistically better than, the same as, and worse than the LS (L1) SVM, respectively.

From the table, from the standpoint of the average accuracy, the ULDM performed best but from the standpoint of statistical analysis the three classifiers are comparable.

We measured the average CPU time per data set including model selection by fivefold cross-validation, training a classifier, and classifying the test data by the trained classifier. We used a personal computer with 3.4GHz CPU and 16GB memory. Table 2 shows the results. In average, the LS SVM is the

Table 1. Accuracy comparison of the two-class problems for RBF kernels

Data	ULDM	LS SVM	L1 SVM
Banana (2/400/4900/100)	89.12±0.69	**89.17**±0.66	**89.17**±0.72
Breast cancer (9/200/77/100)	**73.70**±4.42	73.13±4.68	73.03±4.51
Diabetes (8/468/300/100)	**76.51**±1.95	76.19±2.00	76.29±1.73
Flare-solar (9/666/400/100)	66.28±2.05	66.25±1.98	**66.99**±2.12
German (20/700/300/100)	**76.12**±2.30	76.10±2.10	75.95±2.24
Heart (13/170/100/100)	82.57±3.64	82.49±3.60	**82.82**±3.37
Image (18/1300/1,010/20)	97.16±0.68	**97.52**±0.54	97.16±0.41
Ringnorm (20/400/7,000/100)	98.16±0.35	**98.19**±0.33	98.14±0.35
Splice (60/1000/2,175/20)	**89.16**±0.53	88.98±0.70	88.89±0.91
Thyroid (5/140/75/100)	95.15±2.27	95.08±2.55	**95.35**±2.44
Titanic (3/150/2051/100)	**77.46**±0.91	77.39±0.83	77.39±0.74
Twonorm (20/400/7000/100)	97.41±0.26	**97.43**±0.27	97.38±0.26
Waveform (21/400/4600/100)	**90.18**±0.54	90.05±0.59	89.76±0.66
Average	**85.31** (6/6/1)	85.23 (4/5/4)	85.26 (4/3/6)
W/T/L	—	0/13/0	1/11/1

Table 2. Execution time comparison of the two-class problems (in seconds)

Data	ULDM	LS SVM	L1 SVM
Banana	27.00	12.03	**4.92**
B. cancer	3.18	**1.69**	7.08
Diabetes	43.67	**20.30**	22.96
Flare-solar	194.46	**67.28**	218.67
German	348.92	**98.72**	776.53
Heart	1.94	**1.12**	1.75
Image	4347.34	1826.86	**56.70**
Ringnorm	27.71	13.15	**12.57**
Splice	1792.23	740.76	**30.71**
Thyroid	1.16	0.69	**0.33**
Titanic	1.34	**0.75**	21.25
Twonorm	31.79	13.33	**10.46**
Waveform	31.08	**13.64**	35.61
B/S/W	0/5/8	7/6/0	6/2/5

fastest and the ULDM the slowest. Because the ULDM and LS SVM were trained by solving the sets of linear equations, slower training by the ULDM was due to more complex calculation in setting the coefficients of the linear equations.

4.2. Multiclass problems

We evaluated the ULDM using nine multiclass data sets listed in Table 3. For each problem, there is one training data set and one test data set. We trained the classifiers using fuzzy pairwise classification [2]. The table lists the parameter values determined by fivefold cross-validation. For each problem, the three classifiers took on the similar γ values and the LS and L1 SVM took on the similar C values. The C values for the ULDM were small.

Table 4 shows the accuracy of the three classifiers using RBF kernels. From the standpoint of the average accuracy the L1 SVM was the highest and the ULDM was the second highest. But from the best/second best/worst accuracies, the ULDM and the LS SVM were the best and the L1 SVM was the worst.

Table 5 shows the execution time of the three classifiers. The time includes the cross-validation for the selected γ value, training for the selected parameter values, and classifying the test data. For all the cases, the L1 SVM was the fastest and the ULDM the slowest except one case.

Table 3. Selected parameter values for the multiclass problems (C, γ)

Data	ULDM	LS SVM	L1 SVM
Numeral (12/10/810/820) [2]	10^{-4}, 15	1000, 0.01	10, 5
Thyroid (21/3/3,772/3,428) [26]	10^{-10}, 50	2000, 50	2000, 10
Blood cell (13/12/3,097/3,100) [2]	10^{-12}, 0.5	500, 5	100, 5
Hiragana-50 (50/39/4,610/4,610) [2]	10^{-10}, 5	100, 10	100, 5
Hiragana-13 (13/38/8,375/8,356) [2]	10^{-10}, 10	2000, 15	1000, 15
Hiragana-105 (105/38/8,375/8,356) [2]	10^{-6}, 10	2000, 15	10, 10
Satimage (36/6/4,435/2,000) [26]	10^{-10}, 5	10, 200	10, 200
USPS (256/10/7,291/2,007) [27]	10^{-6}, 50	500, 5	100, 10
Letter (16/26/16,000/4,000) [26]	10^{-6}, 50	50, 50	10, 200

Table 4. Accuracy comparison of the multiclass problems

Data	ULDM	LS SVM	L1 SVM
Numeral	99.39	99.15	**99.76**
Thyroid	95.57	95.39	**97.26**
Blood cell	**94.61**	94.23	93.16
Hiragana-50	98.92	**99.48**	99.00
Hiragana-13	**99.90**	99.87	99.79
Hiragana-105	**100.00**	**100.00**	**100.00**
Satimage	**92.25**	91.95	91.90
USPS	95.42	**95.47**	95.27
Letter	97.75	**97.88**	97.85
Average	97.09	97.04	**97.11**
B/S/W	4/3/2	4/3/2	3/2/4

Training of the ULDM by matrix inversion becomes slow when the number of training data is large. To speed up training of the LDM, in [19], the coordinate descent method [22] was used. Because the coefficient matrix of the ULDM is positive definite, the coordinate descent method is also applicable to the ULDM. But because of space limitation we do not discuss it here.

4.3. Comparison with LDMs

We compared the ULDM with the LDM using the two-class data sets listed in Table 6, which are included in Table 1 of [19]. Because the other regular-scale data sets in the table were different in the number of data and/or the number of features from the data sets available in the Internet, we did not use them. As in [19], we randomly split the original data set into the training data set and the test data set with equal sizes and generated 30 pairs.

Table 7 lists the results using the RBF and linear kernels. The accuracies in "LDM" and "SVM" columns were cited from [19]. Because the average accuracies and their standard deviations were calculated using 30 files, the effect of different split on them is considered to be small. "SVM" in the Table is the L1 SVM.

From the last row of the table, the ULDM is slightly better than the L1 SVM or the SVM and comparable to the LDM. Therefore, the simplification of the LDM architecture did not affect much on the generalization ability.

5. Conclusions

To reduce the number of hyperparameters of the large margin distribution machine (LDM), we proposed a unconstrained

Table 5. Execution time comparison of the multiclass problems (in seconds)

Data	ULDM	LS SVM	L1 SVM
Numeral	10.12	9.31	**0.70**
Thyroid	27497.00	14383.67	**34.60**
Blood cell	828.07	476.51	**10.90**
Hiragana-50	1517.76	1524.47	**61.01**
Hiragana-13	9028.28	6315.22	**86.16**
Hiragana-105	12229.78	11762.44	**225.58**
Satimage	12018.75	6428.43	**455.57**
USPS	33052.67	17212.67	**728.87**
Letter	114348.06	50594.73	**11015.95**
B/S/W	0/1/8	0/8/1	9/0/0

Table 6. Benchmark data specification for two-class problems for comparison

Data	Inputs	Train	Test	Sets
Australian [28]	14	345	345	30
Breast-cancer [26]	9	138	139	30
Fourclass [28]	2	431	431	30
Sonar [26]	60	104	104	30

Table 7. Accuracy comparison with LDMs (l: linear kernel, r: RBF kernel)

Data	ULDM	L1 SVM	LDM [19]	SVM [19]
Austrarian (l)	**85.95**±1.11	85.27±1.11	85.9±1.5	85.7±1.3
Austrarian (r)	**85.82**±1.12	85.51±1.29	85.7±1.4	85.3±1.3
Breast-cancer (l)	**73.19**±2.56	71.99±2.41	72.5±2.7	71.7±3.3
Breast-cancer (r)	73.31±2.56	72.97±2.72	**75.3**±2.7	72.9±3.0
Fourclass (l)	75.55±1.65	**76.78**±1.43	72.3±1.4	72.4±1.4
Fourclass (r)	**99.92**±0.30	99.89±0.16	99.8±0.3	99.8±0.3
Sonar (l)	73.53±3.99	**74.42**±3.54	73.6±3.6	72.5±3.9
Sonar (r)	83.11±2.96	83.88±4.17	**84.6**±3.2	84.2±3.4
Average	81.31	81.34	81.21	80.56
W/T/L	—	2/5/1	1/6/1	2/6/0

LDM (ULDM) which has the same number of hyperparameters as the L1 SVM. As in the LDM, the ULDM maximizes the margin mean (linear term) and minimizes the margin variance (quadratic term). Optimizing the objective function, which is the weighed sum of the linear and quadratic terms results in solving a set of linear equations and the weight works as scaling and does not change the separating hyperplane.

We showed that the decision boundary of the ULDM passes through the mean of the mapped total training data if the numbers of the Classes 1 and 2 data are the same. We also derived the decision boundary for a one-dimensional input.

According to the computer experiments using two-class and multiclass problems, we showed that the ULDM trained by matrix inversion has comparable generalization ability with the L1 SVM and LS (least squares) SVM. But because matrix inversion is slow for a large number of training data, we need to speed up training for large problems.

References

[1] V. N. Vapnik. Statistical Learning Theory. John Wiley & Sons, 1998.

[2] S. Abe. Support Vector Machines for Pattern Classification. Springer, second edition, 2010.

[3] G. R. G. Lanckriet, L. El Ghaoui, C. Bhattacharyya, and M. I. Jordan. A robust minimax approach to classification. Journal of Machine Learning Research, 3:555–582, 2002.

[4] P. K. Shivaswamy and T. Jebara. Ellipsoidal kernel machines. In Proc. 11th Int. Conf. Artificial Intelligence and Statistics (AISTATS 2007), 2007.

[5] H. Xue, S. Chen, and Q. Yang. Structural regularized support vector machine: A framework for structural large margin classifier. IEEE Trans. Neural Networks, 22(4):573–587, 2011.

[6] X. Peng and D. Xu. Twin Mahalanobis distance-based support vector machines for pattern recognition. Information Sciences, 200:22–37, 2012.

[7] Y. Grandvalet and S. Canu. Adaptive scaling for feature selection in SVMs. In Advances in Neural Information Processing Systems 15, pages 569–576. MIT Press, 2003.

[8] S. Abe. Training of support vector machines with Mahalanobis kernels. In Artificial Neural Networks: Formal Models and Their Applications—Proc. 15th Int. Conf., Part II, pages 571–576. Springer, 2005.

[9] X. Liang and Z. Ni. Hyperellipsoidal statistical classifications in a reproducing kernel Hilbert space. IEEE Trans. Neural Networks, 22(6):968–975, 2011.

[10] M. Fauvel, J. Chanussot, J.A. Benediktsson, and A. Villa. Parsimonious Mahalanobis kernel for the classification of high dimensional data. Pattern Recognition, 46(3):845–854, 2013.

[11] T. Reitmaier and B. Sick. The responsibility weighted Mahalanobis kernel for semi-supervised training of support vector machines for classification. Information Sciences, 323:179–198, 2015.

[12] Y. Freund and R. E. Schapire. A decision-theoretic generalization of online learning and an application to boosting. Journal of Computer and System Sciences, 55(1):119–139, 1997.

[13] L. Reyzin and R. E. Schapire. How boosting the margin can also boost classifier complexity. In Proc. 23rd Int. Conf. Machine learning, pages 753–760. ACM, 2006.

[14] W. Gao and Z.-H. Zhou. On the doubt about margin explanation of boosting. Artificial Intelligence, 203:1–18, 2013.

[15] A. Garg and D. Roth. Margin distribution and learning. In Machine Learning, Proc. the 20th Int. Conf., 2003, pages 210–217, 2003.

[16] K. Pelckmans, J. Suykens, and B. D. Moor. A risk minimization principle for a class of Parzen estimators. In Advances in Neural Information Processing Systems 20, pages 1137–1144. Curran Associates, Inc., 2008.

[17] F. Aiolli, G. Da San Martino, and A. Sperduti. A kernel method for the optimization of the margin distribution. In Artificial Neural Networks—Proc. 18th Int. Conf., Part I, pages 305–314. Springer, 2008.

[18] L. Zhang and W.-D. Zhou. Density-induced margin support vector machines. Pattern Recognition, 44(7):1448–1460, 2011.

[19] Z.-H. Zhou T. Zhang. Large margin distribution machine. In 20th ACM SIGKDD Conf. Knowledge Discovery and Data Mining, pages 313–322, 2014.

[20] Z.-H. Zhou. Large margin distribution learning. In Artificial Neural Networks in Pattern Recognition, pages 1–11. Springer, 2014.

[21] J. A. K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle. Least Squares Support Vector Machines. World Scientific Publishing, 2002.

[22] C.-J. Hsieh, K.-W. Chang, C.-J. Lin, S. S. Keerthi, and S. Sundararajan. A dual coordinate descent method for large-scale linear SVM. In Proc. 25th Int. Conf. Machine Learning, pages 408–415. ACM, 2008.

[23] J. C. Bezdek, J. M. Keller, R. Krishnapuram, L. I. Kuncheva, and N. R. Pal. Will the real iris data please stand up? IEEE Trans. Fuzzy Systems, 7(3):368–369, 1999.

[24] G. Rätsch, T. Onoda, and K.-R. Müller. Soft margins for AdaBoost. Machine Learning, 42(3):287–320, 2001.

[25] S. Abe. Fusing sequential minimal optimization and Newton's method for support vector training. Int. Journal of Machine Learning and Cybernetics, 7 (3):345–364, 2016.

[26] A. Asuncion and D. J. Newman. UCI machine learning repository, http://www.ics.uci.edu/∼mlearn/MLRepository.html. 2007.

[27] USPS Dataset. http://www-i6.informatik.rwth-aachen.de/˜keysers/usps.html.

[28] C.-C. Chang and C.-J. Lin. LIBSVM–A library for support vector machines: http://www.csie.ntu.edu.tw/˜cjlin/libsvm/.