INFORMATICA, 2007, Vol. 18, No. 1, 137–157

Oblique Support Vector Machines *

Chih-Chia YAO
Department of Computer Science and Information Engineering
Chaoyang University of Technology
Wufong, Taichung 41349, Taiwan
e-mail: cedric@nkc.edu.tw

Pao-Ta YU
Department of Computer Science and Information Engineering
National Chung Cheng University
Ming-Hsiung, Chiayi 62102, Taiwan
e-mail: csipty@cs.ccu.edu.tw

Received: September 2005

Abstract. In this paper we propose a modified framework of support vector machines, called Oblique Support Vector Machines(OSVMs), to improve the capability of classification. The principle of OSVMs is joining an orthogonal vector into weight vector in order to rotate the support hyperplanes. By this way, not only the regularized risk function is revised, but the constrained functions are also modified. Under this modification, the separating hyperplane and the margin of separation are constructed more precise. Moreover, in order to apply to large-scale data problem, an iterative learning algorithm is proposed. In this iterative learning algorithm, three different schemes for training can be found in this literature, including pattern-mode learning, semi-batch mode learning and batch mode learning. Besides, smooth technique is adopted in order to convert the constrained nonlinear programming problem into unconstrained optimum problem. Consequently, experimental results and comparisons are given to demonstrate that the performance of OSVMs is better than that of SVMs and SSVMs.
Key words: SVMs, rotate, orthogonal vector, pattern-mode, batch-mode.

1. Introduction

Support vector machines (SVMs), a classification algorithm for the machine learning community, have attracted much attention in recent years (Schölkopf and Smola, 2002; Vapnik, 1998). In many applications, SVMs have been shown to provide higher performance than traditional learning machines (Chapelle *et al.*, 1999; Guo *et al.*, 2001).

The principle of SVMs is based on approximating structural risk minimization (Vapnik, 1998). The idea of SVMs originates from finding an optimal separating hyperplane in order to separate the largest possible fraction of training set of the same class on the

*This work is supported by National Science Council of the Republic of China under Grant NSC94-2213-E-194-043

same side, while maximizing the distance from either class to the separating hyperplane. According to Vapnik (Vapnik, 1998), this hyperplane minimizes the risk of misclassifying not only the examples in the training set, but also the unseen examples of the test set.

Although there are numerous researches concerning about SVMs, some drawbacks still exist on SVMs. Fig. 1 depicts the geometric construction of separating hyperplane and support hyperplanes by SVMs. Two support hyperplanes which represent the borders of margin of separation are used to generate the separating hyperplane. In Fig. 1, the separating hyperplane is the maximal margin hyperplane with geometric margin d_1. Can the separating hyperplane achieve the optimal solution? Fig. 2 shows an example to illustrate that the validity of classifying by the separating hyperplane is worse to the testing samples. In Fig. 2, the training samples appeared in Fig. 1 are replaced with • and the testing samples are represented with symbols 'x' and 'o', respectively. Obviously, in Fig. 2 the solution which is provided by SVMs is worse in this case. It is because the margin of separation constructed by SVMs doesn't fit the optimal margin of separation.

In this paper we propose a new model of the support vector machines, which is called oblique support vector machines (OSVMs). The idea of oblique support vector machines originates from joining an orthogonal vector into the weight vector in order to rotate the support hyperplanes. Under this modification, the support hyperplanes can fit the borders of optimal margin of separation. After joining an orthogonal vector, these two support hyperplanes become $(w_1 + w_2)^T x + (b_1 - b_2) = 0$ and $(w_1 - w_2)^T x + (b_1 + b_2) = 0$. This new model is shown in Fig. 3. On the new architecture, OSVMs are adaptable for any cases and have excellent performance on pattern classification problem.

As we know, the major problem of SVMs is the dependence of the nonlinear separating surface on the entire dataset which creates unwieldy storage problems. Some researches are proposed to overcome this problem, such as SMO, PCGC, RSVM, etc (Gill *et al.*, 1981; Lee and Mangasarian, 2001b; Schölkopf and Smola, 2002). For overcoming this problem, in this paper we propose an interactive learning algorithm which can be im-

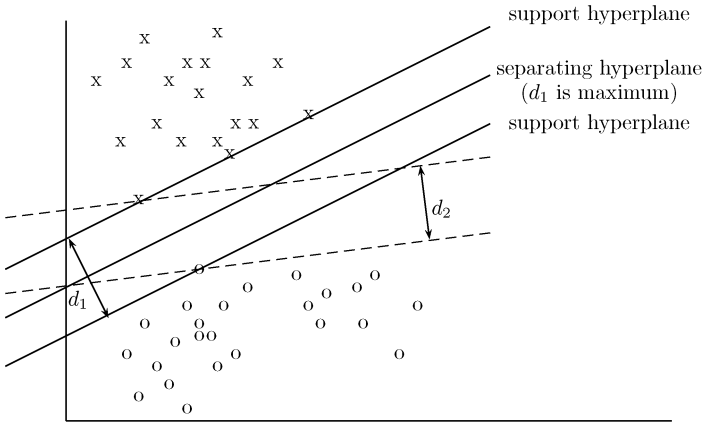

Fig. 1. The illustration of constructing optimal hyperplane by SVMs.

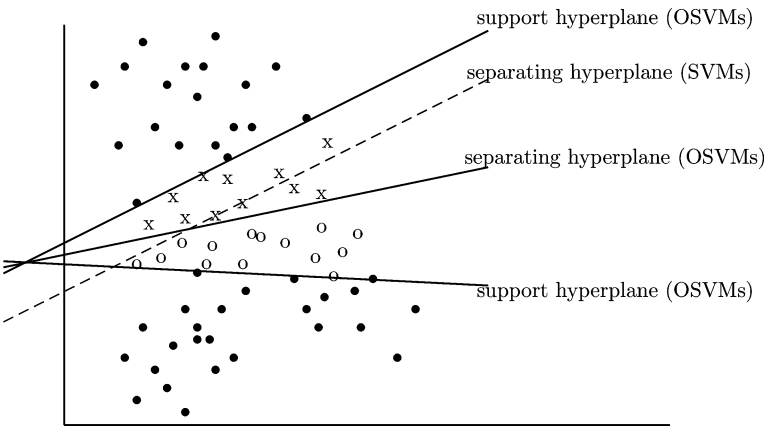

Fig. 2. The illustration of difference between SVMs and OSVMs.

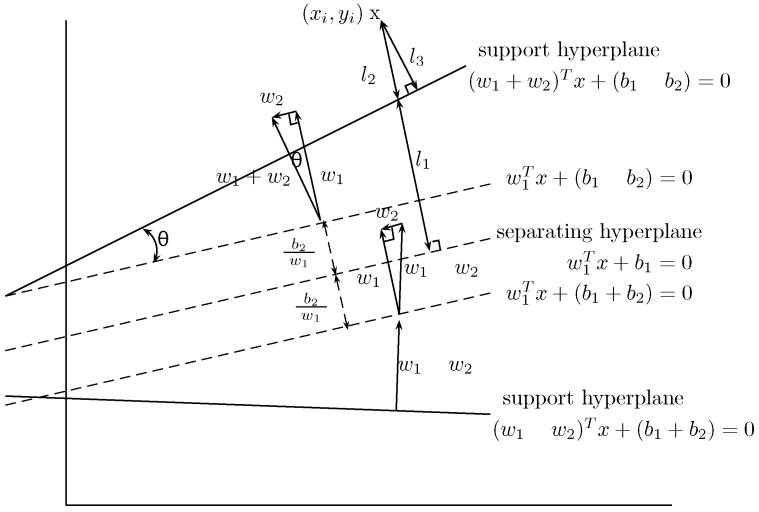

Fig. 3. Illustration of the idea of Oblique Support Vector Machines.

plemented on pattern-mode learning and batch-mode learning (Heskes and Wiegerinck, 1996; Qin *et al.*, 1992). On the learning algorithm, smooth technique is adopted in order to convert the constrained nonlinear programming problem into unconstrained optimum problem (Lee and Mangasarian, 2001a), so that conjugate gradient method can be used to speed up the convergent rate. Batch-mode learning is completely deterministic but requires additional storage for each parameters which may be inconvenient for large dataset problem. Pattern-mode learning, is less demanding on the memory side, but the arbitrariness of the order in which the patterns are presented makes them stochastic. Further discussion concerning pattern learning and batch learning can be found in (Heskes and Wiegerinck, 1996; Qin *et al.*, 1992). They mentioned that pattern learning is approximating to batch learning when the learning rate is small. This conclusion is shown in our

simulation results.

To demonstrate the efficiency of OSVMs, OSVMs are compared with Smooth Support Vector Machines (SSVMs) (Lee and Mangasarian, 2001a). The principle of SSVMs is introduced a smoothing technique into SVMs. By this way the optimal problem with constrained function can be converted into unconstrained optimal problem which possesses the property of convexity and infinitely differentiability. SSVMs have been compared to possess better performance than RLP, FSV, 1-norm approach ($\mathrm{SVM}_{\|\cdot\|_1}$) and the classical support vector machines ($\mathrm{SVM}_{\|\cdot\|_2^2}$) (Bennett and Mangasarian, 1992; Bradley and Mangasarian, 1998; Cherkassky and Mulier, 1998; Vapnik, 1998). And for massive dataset, the iterative learning algorithm is compared with RSVMs which are comparable than SMO, SOR and $\mathrm{SVM}^{\mathrm{light}}$ (Joachims, 1999; Lee and Mangasarian, 2001b; Platt, 1999). The experimental results demonstrate that OSVMs are better than SSVMs.

We briefly outline the contents of the paper now. In Section 2 we review the basic concept of SVMs. In Section 3 the basic idea of OSVMs is introduced. In Sections 4 and 5 we conduct the principle of noniterative and iterative algorithm for OSVMs. Last, in Section 6 some experimental results are discussed.

2. Basic Concepts

2.1. *Support Vector Machines for Linearly Case*

Consider the training samples $\{(x_i, y_i)\}_{i=1}^N$, where x_i is the input pattern for the ith sample and y_i is the corresponding desired response; $x_i \in R^m$ and $y_i \in \{-1, 1\}$. The objective is to define a separating hyperplane which divides the set of examples such that all the points with the same class are on the same sides of the hyperplane.

Let w_o and b_o denote the optimum values of the weight vector and bias, respectively. Correspondingly, the optimal separating hyperplane, representing a multidimensional linear decision surface in the input space, is given by

$$w_o^T x + b_o = 0. \tag{1}$$

The set of vectors is said to be optimally separated by the hyperplane if it is separated without error and the margin of separation is maximal. Then, the separating hyperplane $w^T x + b = 0$ must satisfy the following constraints:

$$y_i(w^T x_i + b) \geqslant 1, \quad i = 1, 2, \ldots, N. \tag{2}$$

Two support hyperplanes $w^T x_i + b = 1$ and $w^T x_i + b = -1$, which define the two borders of margin of separation, are specified on (2). According to (2), the optimal separating hyperplane is the maximal margin hyperplane with the geometric margin $\frac{2}{\|w\|}$. Hence the optimal separating hyperplane is the one that satisfies (2) and minimizes the cost function,

$$\Phi(w) = \frac{1}{2} w^T w. \tag{3}$$

Since the cost function is a convex function, a Lagrange function can be used to minimize the constrained optimization problem:

$$L(w, b, \alpha) = \frac{1}{2} w^T w - \sum_{i=1}^{N} \alpha_i [y_i (w^T x_i + b) - 1], \tag{4}$$

where $\alpha_1, \alpha_2, \ldots,$ and α_N are the Lagrange multipliers.

Once the solution $\alpha^o = (\alpha_1^o, \alpha_2^o, \ldots, \alpha_N^o)$ of (4) has been found, the optimal weight vector is given by,

$$w^o = \sum_{i=1}^{N} \alpha_i^o y_i x_i. \tag{5}$$

Classical Lagrangian duality enables the primal problem to be transformed to its dual problem. The dual problem of (4) is reformulated as

$$Q(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j x_i^T x_j, \tag{6}$$

with $\alpha_i \geqslant 0$, constrained by $\sum_{i=1}^{N} y_i \alpha_i = 0$.

2.2. *Nonlinear Support Vector Machines*

Input data can be mapped onto an alternative, higher-dimensional space, called *feature space* through a replacement to improve the representation.

$$x_i \cdot x_j \to \varphi(x_i)^T \varphi(x_j). \tag{7}$$

The functional form of the mapping $\varphi(\cdot)$ does not need to be known since it is implicitly defined by selected kernel function: $K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$, such as polynomials, splines, radial basis function networks or multilayer perceptrons (Schölkopf and Smola, 2002). A suitable choice of kernel can make the data separable in feature space despite being non-separable in the original input space.

When x_i is replaced by its mapping $\varphi(x_i)$, (6) becomes,

$$Q(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j K(x_i, x_j). \tag{8}$$

3. Oblique Support Vector Machines for Classification

This section proposes a new framework for classification, called OSVMs. The new framework is based on the theory of SVMs, but uses a different way to construct the margin

of separation. Accordingly, the borders of margin of separation and the separating hyperplane are constructed precisely.

Figs. 1 and 2 depict different approaches to construct the optimal separating hyperplane in a two-dimensional input space. In these two figures, two support hyperplanes which define the borders of margin of separation are used to construct the separating hyperplane. Fig. 1 depicts the geometric construction of an optimal separating hyperplane with maximum margin based on SVMs. In contrast to Fig. 1, Fig. 2 depicts the geometric construction of OSVMs. The idea of oblique support vector machines originates from finding two support hyperplanes to fit the optimal borders of margin of separation, and then choosing the hyperplane which is located on the middle of these two support hyperplanes as the separating hyperplane. Suppose the weight vector of the separating hyperplane is w_1, these two support hyperplanes can be formed by adding or subtracting an orthogonal vector. Fig. 3 shows the idea of oblique support vector machines. Based on SVMs, the design goal is to maximize the distance between two support hyperplanes. The varying of the distance can be viewed as moving the two support hyperplanes from the separating hyperplane in a direction opposite to each other. Suppose that, after moving away from the separating hyperplane, the distance between the support hyperplane with $w_1^T x + b_1 = 0$ is $\frac{b_2}{\|w_1\|}$; then, the upper support hyperplane is given by, $w_1^T x + (b_1 - b_2) = 0$.

Next, an orthogonal vector w_2 is added into weight vector w_1. This causes the upper support hyperplane to be rotated θ. Under this modification, the new upper support hyperplane is $(w_1 + w_2)^T x + (b_1 - b_2) = 0$. Then the angle between hyperplane $w_1^T x + (b_1 - b_2) = 0$ and $(w_1 + w_2)^T x + (b_1 - b_2) = 0$ is

$$\cos\theta = \frac{w_1^T (w_1 + w_2)}{\|w_1\|\|w_1 + w_2\|}. \tag{9}$$

In contrast to the upper support hyperplane, the lower support hyperplane is also moved away from the separating hyperplane but in opposite direction and subtracted the weight vector w_1 by w_2. The moving distance is also equal to $\frac{b_2}{\|w_1\|}$. Finally the lower support hyperplane is $(w_1 - w_2)^T x + (b_1 + b_2) = 0$.

4. Architecture of Oblique Support Vector Machines

4.1. *Oblique Support Vector Machines in Linearly Case*

In Fig. 3, the separating hyperplane $w_1^T x + b_1 = 0$ is the median hyperplane between the two hyperplanes $(w_1 + w_2)^T x + (b_1 - b_2) = 0$ and $(w_1 - w_2)^T x + (b_1 + b_2) = 0$, where w_1 is orthogonal to w_2.

Since the input data $x_i, 1 \leqslant i \leqslant N$, is located outside the bounded region, first, (x_i, y_i) is supposed to be above the hyperplane $(w_1 + w_2)^T x + (b_1 - b_2) = 0$. Then the distance between x_i and the separating hyperplane is,

$$l_1 + l_2 = \frac{|w_1^T x_i + b_1|}{\|w_1\|}. \tag{10}$$

And that between x_i and the hyperplane $(w_1 + w_2)^T x + (b_1 - b_2) = 0$ is,

$$l_3 = \frac{|(w_1 + w_2)^T x_i + (b_1 - b_2)|}{\|w_1 + w_2\|}. \tag{11}$$

From Fig. 3,

$$l_2 = \frac{l_3}{cos\theta} = l_3 \times (\frac{w_1^T(w_1 + w_2)}{\|w_1\|\|w_1 + w_2\|})^{-1} = \frac{|(w_1 + w_2)^T x_i + (b_1 - b_2)|}{\|w_1\|}. \tag{12}$$

Since x_i is supposed to be located outside the margin of separation bounded by $(w_1 + w_2)^T x + (b_1 - b_2) = 0$ and $(w_1 - w_2)^T x + (b_1 + b_2) = 0$. So we can conclude:

$$\begin{cases} (w_1 - w_2)^T x_i + (b_1 + b_2) < 0 \Rightarrow w_1^T x_i + b_1 < 0, \\ (w_1 + w_2)^T x_i + (b_1 - b_2) > 0 \Rightarrow w_1^T x_i + b_1 > 0. \end{cases} \tag{13}$$

Eq. 13 is applied to (12) to yield,

$$l_1 = \frac{|w_1^T x_i + b_1|}{\|w_1\|} - l_2 = \frac{|w_2^T x_i - b_2|}{\|w_1\|}. \tag{14}$$

On SVMs, the design goal of the classifier is to determine maximal margin hyperplane. However, this design goal is hard to achieve on oblique support vector machines because the margin of separating is not bounded by two parallel hyperplane. Rather than maximizing the geometric margin, on oblique support vector machines the distances l_1 on each training samples are taken into consideration. OSVMs maximize the summation of l_1 for each training samples, so the problem is stated as follows:

$$\text{maximize} \quad \frac{1}{2} \sum_{i=1}^{N} \frac{(w_2^T x_i - b_2)^2}{\|w_1\|_2^2} \tag{15}$$

$$\begin{aligned} \text{subject to} \quad & y_i \left((w_1 + w_2)^T x_i + (b_1 - b_2) \right) \geqslant 0, \quad i = 1, 2, \ldots, N, \\ & y_i \left((w_1 - w_2)^T x_i + (b_1 + b_2) \right) \geqslant 0, \quad i = 1, 2, \ldots, N, \\ & w_1^T w_2 = 0. \end{aligned}$$

Another, constraint: $k\|w_2\|_2^2 \leqslant \|w_1\|_2^2$ is applied to increase tolerance. From Fig. 3 and (9), the angle θ is determined by the ratio of $\|w_1\|$ to $\|w_2\|$. If k is approaching to

∞, then the oblique support vector machines are like support vector machines. On the contrary, if k is approaching to zero, the support hyperplanes are almost orthogonal to the optimal hyperplane and are useless for classification. Such a case may arise when few training samples are available. The lower bound on k should be set to avoid this case.

Let $\underline{X} = [x_1, x_2, \ldots, x_N]^T$, $e_1 = [1, 1, \ldots, 1]^T$, $e_0 = [0, 0, \ldots, 0]^T$, $\underline{X} \in R^{N \times m}$, $e_1, e_0 \in R^N$. Matrix $\underline{Y}$ is an $N \times N$ diagonal matrix with y_i as the ith diagonal element. Then, (15) can be expressed as vector form. Hence, the problem is stated as follows:

$$\text{maximize} \quad \frac{1}{2}\frac{\|\underline{X}w_2 - b_2 e_1\|_2^2}{\|w_1\|_2^2} \tag{16}$$

$$\begin{aligned} \text{subject to} \quad & \underline{Y}\,(\underline{X}(w_1 + w_2) + (b_1 - b_2)e_1) \geqslant e_0, \\ & \underline{Y}\,(\underline{X}(w_1 - w_2) + (b_1 + b_2)e_1) \geqslant e_0, \\ & w_1^T w_2 = 0, \\ & k\|w_2\|_2^2 \leqslant \|w_1\|_2^2, \end{aligned}$$

where k is a user-specified positive parameter.

Accordingly, the model of OSVMs have been reformed to nonlinear optimization problem. For simplistically, on (16) b_2 can be set to 1 because $\frac{b_2}{\|w_1\|}$ represents the widening distance between the support hyperplane and separating hyperplane and is used to set to $\frac{1}{\|w_1\|}$ on SVMs. Besides, $\|\underline{X}w_2 - b_2 e_1\|_2^2$ can be set as a constant t ($t > 0$) if it is satisfied with some constraints which are discussed later. Then the above problem is equivalent to,

$$\text{minimize} \quad \frac{1}{2}\|\hat{w_1}\|_2^2 \tag{17}$$

$$\begin{aligned} \text{subject to} \quad & \underline{Y}\left(\underline{X}(\hat{w_1} + \hat{w_2}) + (\hat{b_1} - \hat{b_2})e_1\right) \geqslant e_0, \\ & \underline{Y}\left(\underline{X}(\hat{w_1} - \hat{w_2}) + (\hat{b_1} + \hat{b_2})e_1\right) \geqslant e_0, \\ & \hat{w_1}^T \hat{w_2} = 0, \\ & k\|\hat{w_2}\|_2^2 \leqslant \|\hat{w_1}\|_2^2, \end{aligned}$$

where k is a user-specified positive parameter and $\hat{w_1} = \frac{w_1}{\|\underline{X}w_2 - b_2 e_1\|}$, $\hat{w_2} = \frac{w_2}{\|\underline{X}w_2 - b_2 e_1\|}$, $\hat{b_1} = \frac{b_1}{\|\underline{X}w_2 - b_2 e_1\|}$.

The reason why $\|\underline{X}w_2 - b_2 e_1\|_2^2$ can be set as a constant is discussed as follows.

First, $\|\underline{X}w_2 - b_2 e_1\|_2^2$ can be expressed as a quadratic form. Let $\vec{A} = [\underline{X},\ -e_1]$ and $\vec{w_2} = [w_2^T,\ b_2]^T$. Then

$$\|\underline{X}w_2 - b_2 e_1\|_2^2 = \|\vec{A}\vec{w_2}\|_2^2 = (\vec{A}\vec{w_2})^T(\vec{A}\vec{w_2}) = \vec{w_2}^T \vec{A}^T \vec{A}\vec{w_2}, \tag{18}$$

where ${\vec{w}_2}^T \vec{A}^T \vec{A} \vec{w}_2$ is a quadratic form with respect to $\vec{w}_2$. In (18), $\vec{A} \in R^{N\times(m+1)} (N \gg m)$ and $\vec{A}^T \vec{A} \in R^{(m+1)\times(m+1)}$.

Next, $\vec{A}^T \vec{A}$ is a positive semidefinite matrix. As we know, a symmetric matrix Q is positive semidefinite if and only if $x^T Q x \geqslant 0$, where $Q \in R^{n\times n}$, $\forall x \in R^{n\times 1}$ (Kreysig, 1999). Trivially, $\|\underline{X} w_2 - b_2 e_1\|_2^2 = {\vec{w}_2}^T \vec{A}^T \vec{A} \vec{w}_2$, $\forall \vec{w}_2 \in R^{(m+1)\times 1}$, ${\vec{w}_2}^T \vec{A}^T \vec{A} \vec{w}_2 \geqslant 0$. Note that $\vec{A}^T \vec{A}$ is a symmetric matrix.

Lemma 1 (Horn and Johnson, 1990). *If $A \in R^{m\times n}$, then $r(A) = r(A^T A) = r(AA^T)$, where $r(A)$ denotes the rank of A.*

Lemma 2. *The rank of $\vec{A}^T \vec{A}$ is m.*

Proof. See Appendix.

Since $\vec{A}^T \vec{A}$ is a real symmetric matrix, $\exists P$, $P^T P = PP^T = I$, $\ni$ $\vec{A}^T \vec{A} = PDP^{-1} = PDP^T$ where D is a diagonal matrix and let D be

$$D = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 & 0 \\ 0 & \lambda_2 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & \lambda_m & 0 \\ 0 & 0 & \dots & 0 & 0 \end{bmatrix} = \begin{bmatrix} D_1 & O_{m\times 1} \\ O_{1\times m} & O_{1\times 1} \end{bmatrix}. \tag{19}$$

Note that $\lambda_i > 0$, $i = 1, 2, \dots, m$. Also, the orthogonal matrix P can be let as $P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix}$, where $P_{11} \in R^{m\times m}$, $P_{12} \in R^{m\times 1}$, $P_{21} \in R^{1\times m}$, $P_{22} \in R^{1\times 1}$. Then

$$\begin{aligned} PDP^T &= \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \begin{bmatrix} D_1 & O_{m\times 1} \\ O_{1\times m} & O_{1\times 1} \end{bmatrix} \begin{bmatrix} P_{11}^T & P_{21}^T \\ P_{12}^T & P_{22} \end{bmatrix} \\ &= \begin{bmatrix} P_{11} D_1 & O_{m\times 1} \\ P_{21} D_1 & O_{1\times 1} \end{bmatrix} \begin{bmatrix} P_{11}^T & P_{21}^T \\ P_{12}^T & P_{22} \end{bmatrix} = \begin{bmatrix} P_{11} D_1 P_{11}^T & P_{11} D_1 P_{21}^T \\ P_{21} D_1 P_{11}^T & P_{21} D_1 P_{21}^T \end{bmatrix}. \end{aligned} \tag{20}$$

Lemma 3. *If t satisfies below constraints:*

(1) $H = t - b_2 P_{21} D_1 P_{21}^T b_2 + \frac{b_2^2 u_1^2}{\lambda_1} + \dots + \frac{b_2^2 u_m^2}{\lambda_m} > 0$.

(2) $\sqrt{\left(\frac{b_2 u_1}{\lambda_1}\right)^2 + \left(\frac{b_2 u_2}{\lambda_2}\right)^2 + \dots + \left(\frac{b_2 u_m}{\lambda_m}\right)^2} < \min\left(\sqrt{\frac{H}{\lambda_1}}, \sqrt{\frac{H}{\lambda_2}}, \dots, \sqrt{\frac{H}{\lambda_m}}\right)$,

where $P_{21} D_1 = [u_1, u_2, \dots, u_m]$, then ${\vec{w}_2}^T \vec{A}^T \vec{A} \vec{w}_2 = t$ is an m-dimensional ellipsoid with the origin of coordinate inside this ellipsoid.

Proof. See Appendix.

On the design of OSVMs, w_2 is the key factor to control the degree of rotating. If w_2 can be guaranteed to point to any direction in R^m, then the optimal solution is possibly

obtained by running the nonlinear programming. For instance, in R^2, Fig. 4(a) shows a case of origin inside the ellipse (ellipsoid in R^2 is called ellipse). We can find that the vector on this ellipse can point to any direction in R^2. That is, the vector on this ellipse can be rotated to any degree between 0^o to 360^o. In Fig. 4(b), we can find that the vector on the ellipse only can be rotated between 0^o to 90^o due to the origin not inside the ellipse.

Therefore, a theorem can be concluded as follows.

Theorem 1. *In* (16), $\|\underline{X}w_2 - b_2e_1\|_2^2$ *can be set as a constant* t $(t > 0)$ *and the optimal solution can be found by running nonlinear programming without loss accuracy if* t *is satisfied conditions stated in Lemma* 3.

Theorem 1 reveals that the optimal solution can be obtained if $\|\underline{X}w_2 - b_2e_1\|_2^2$ is set as a constant t $(t > 0)$ under two constraints stated in Lemma 3. However, it is hard to run the nonlinear programming by adding these two extra conditions into (17). To solve the nonlinear programming problem with these two constraints is still an open problem and is left as a future work. In this paper, a heuristic algorithm is proposed to determine the value of t such that the optimal solution can be found with high probability sense.

PROPOSITION 1. The nonlinear optimization problem which is stated in (17) can be solved by introducing the Lagrangian function.

Proof. See Appendix.

In addition, the model of OSVMs also suffers with another problem. Equality constraint $w_1^T w_2 = 0$ causes the optimization problem is not convex (Bertsekas and Hardcover, 1999). Since the nonlinear programming problem with equality constraint is hard to solve, an approximate approach can be got by rewriting the cost function and equality constraint. After rewriting the cost function and equality constraint, the problem is stated as follows:

$$\text{minimize} \quad \frac{1}{2}\|\hat{w_1}\|_2^2 + C_\varsigma \varsigma^2 \tag{21}$$

$$\text{subject to} \quad \underline{Y}\left(\underline{X}(\hat{w_1} + \hat{w_2}) + (\hat{b_1} - \hat{b_2})e_1\right) \geqslant e_0,$$

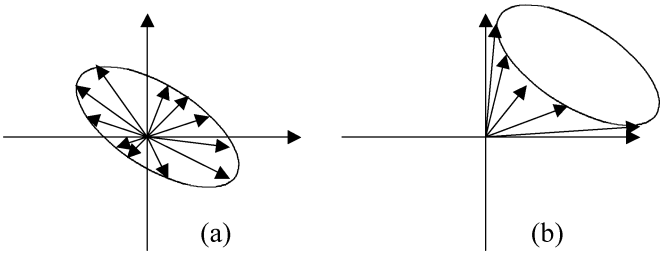

Fig. 4. Two possible cases for w_2 can be pointed to any direction or not.

$$\underline{Y}\left(\underline{X}(\hat{w_1} - \hat{w_2}) + (\hat{b_1} + \hat{b_2})e_1\right) \geqslant e_0,$$
$$0 \leqslant \hat{w_1}^T \hat{w_2} \leqslant \varsigma,$$
$$k\|\hat{w_2}\|_2^2 \leqslant \|\hat{w_1}\|_2^2,$$

where k is a user-specified positive parameter. In (21), when C_ς is big enough, $\hat{w_1}^T \hat{w_2}$ will approximate to zero.

The Lagrangian function is introduced to solve the above problem. Let

$$\begin{aligned} L(\hat{w_1}, \hat{w_2}, \hat{b_1}, \hat{b_2}, \varsigma) = & \frac{1}{2}\|\hat{w_1}\|_2^2 + C_\varsigma \varsigma^2 - \alpha^T \left[\underline{Y}\left(\underline{X}(\hat{w_1} + \hat{w_2}) + (\hat{b_1} - \hat{b_2})e_1\right)\right] \\ & - \beta^T \left[\underline{Y}\left(\underline{X}(\hat{w_1} - \hat{w_2}) + (\hat{b_1} + \hat{b_2})e_1\right)\right] - \gamma_1(\varsigma - \hat{w_1}^T \hat{w_2}) \\ & - \gamma_2(\hat{w_1}^T \hat{w_2}) - \gamma_3\left(\|\hat{w_1}|_2^2 - k\|\hat{w_2}\|_2^2\right), \end{aligned} \tag{22}$$

where the nonnegative variables α, β, γ_1, γ_2 and γ_3 are Lagrange multipliers where $\alpha, \beta \in R^{N\times 1}$. The optimal value is obtained by setting the results of differentiating $L(\hat{w_1}, \hat{w_2}, \hat{b_1}, \hat{b_2}, \varsigma)$ with respect to $\hat{w_1}, \hat{w_2}, \hat{b_1}, \hat{b_2}, \varsigma$ equal to zero. Thus, (23)–(26) are yielded.

$$\alpha^T \underline{Y} e_1 = \beta^T \underline{Y} e_1 = 0, \tag{23}$$
$$2C_\varsigma \varsigma = \gamma_1, \tag{24}$$
$$\hat{w_2} = \frac{\gamma_1 - \gamma_2 + 2\gamma_3 - 1}{(\gamma_1 - \gamma_1)^2 - 2k\gamma_3 + 4k\gamma_3^2}\underline{X}^T \underline{Y}^T \alpha + \frac{\gamma_1 - \gamma_2 - 2\gamma_3 + 1}{(\gamma_1 - \gamma_2)^2 - 2k\gamma_3 + 4k\gamma_3^2}\underline{X}^T \underline{Y}^T \beta, \tag{25}$$
$$\hat{w_1} = \frac{\gamma_1 - \gamma_2 - 2k\gamma_3}{(\gamma_1 - \gamma_2)^2 - 2k\gamma_3 + 4k\gamma_3^2}\underline{X}^T \underline{Y}^T \alpha - \frac{\gamma_1 - \gamma_2 + 2k\gamma_3}{(\gamma_1 - \gamma_2)^2 - 2k\gamma_3 + 4k\gamma_3^2}\underline{X}^T \underline{Y}^T \beta. \tag{26}$$

Then the decision surface $w_1^T x + b_1 = 0$ becomes

$$\left(\frac{\gamma_1 - \gamma_2 - 2k\gamma_3}{(\gamma_1 - \gamma_2)^2 - 2k\gamma_3 + 4k\gamma_3^2}\underline{X}^T \underline{Y}^T \alpha - \frac{\gamma_1 - \gamma_2 + 2k\gamma_3}{(\gamma_1 - \gamma_2)^2 - 2k\gamma_3 + 4k\gamma_3^2}\underline{X}^T \underline{Y}^T \beta\right)^T x + \hat{b_1} = 0. \tag{27}$$

The value of $\hat{b_1}$ and $\hat{b_2}$ don't appear in the above equations, but it can be found by making use of the primal constraints:

$$\hat{b_1} = -\frac{\min_{y_i=1}(\hat{w_1} + \hat{w_2})^T x_i + \max_{y_i=-1}(\hat{w_1} - \hat{w_2})^T x_i}{2}, \tag{28}$$
$$\hat{b_2} = -\frac{\min_{y_i=1}(\hat{w_1} + \hat{w_2})^T x_i - \max_{y_i=-1}(\hat{w_1} - \hat{w_2})^T x_i}{2}. \tag{29}$$

The linearly nonseparable case can be obtained by adding variable ξ^+, ξ^- into (17) and proceeding in a manner to that described in this subsection.

4.2. *Nonlinear Oblique Support Vector Machines*

The input data are mapped into a high-dimensional feature space by some nonlinear mapping chosen a priori. Let φ denote a set of nonlinear transformations from the input space R^m to a higher dimensional feature space. Then (27) becomes,

$$\Big(\frac{\gamma - 2k\gamma_3}{\gamma^2 - 2k\gamma_3 + 4k\gamma_3^2}\varphi(\underline{X})^T\underline{Y}^T\alpha - \frac{\gamma + 2k\gamma_3}{\gamma^2 - 2k\gamma_3 + 4k\gamma_3^2}\varphi(\underline{X})^T\underline{Y}^T\beta\Big)^T\varphi(x) + \hat{b_1} = 0,$$

where $\gamma = \gamma_1 - \gamma_2$.

The inner product of the two vectors induced in the feature space can be replaced by the inner-product kernel denoted by $K(x, x_i)$ and defined by $K(x, x_i) = \varphi(x) \cdot \varphi(x_i)$. Once a kernel $K(x, x_i)$, which satisfies Mercer's condition has been selected, the decision function becomes,

$$\Big(\frac{\gamma - 2k\gamma_3}{\gamma^2 - 2k\gamma_3 + 4k\gamma_3^2}\alpha^T\underline{Y}K(\underline{X}, x) - \frac{\gamma + 2k\gamma_3}{\gamma^2 - 2k\gamma_3 + 4k\gamma_3^2}\beta^T\underline{Y}K(\underline{X}, x)) + \hat{b_1} = 0. \quad (30)$$

Using the method of Lagrange multipliers and proceeding in a manner similar to that described in this subsection, the dual problem for nonlinearly nonseparable case can be gotten.

5. Iterative Learning Algorithm for Oblique Support Vector Machines

A number of optimization techniques can be directly applied to the case of SVMs, including Newton method, conjugate gradient method, primal dual interior-point methods, and others. However, several of these require that the kernel matrix is stored in memory, implying that the space complexity is quadratic in the sample size. This section proposes an iterative learning algorithm to overcome this problem with a large dataset.

5.1. *Iterative Learning Algorithm with Linear Kernel*

The preceding section established that the constrained optimal problem can be transformed into a Lagrangian function and solved by Newton's method. In this section SSVMs are adopted to remove the constraints (Lee and Mangasarian, 2001a). Hence, the nonlinear programming problem is converted into unconstrained optimization and conjugate gradient method can be used to get the solution.

On linearly nonseparable case, the square of the 2-norm of the variable ξ^+, ξ^- is included. Then the problem is formulated as,

$$\text{minimize} \quad \frac{1}{2}\|\hat{w_1}\|_2^2 + C_\varsigma\varsigma^2 + \frac{C_\xi}{2}\left(\|\xi^+\|_2^2 + \|\xi^-\|_2^2\right) \quad (31)$$

$$\text{subject to} \quad \underline{Y}\left(\underline{X}(\hat{w_1} + \hat{w_2}) + e_1(\hat{b_1} - \hat{b_2})\right) + \xi^+ \geqslant e_0, \quad (32)$$

$$\underline{Y}\left(\underline{X}(\hat{w_1} - \hat{w_2}) + e_1(\hat{b_1} + \hat{b_2})\right) + \xi^- \geqslant e_0, \tag{33}$$

$$\xi^+,\ \xi^- \geqslant 0, \tag{34}$$

$$k\|\hat{w_2}\|_2^2 \leqslant \|\hat{w_1}\|_2^2, \tag{35}$$

$$0 \leqslant \hat{w_1}^T \hat{w_2} \leqslant \varsigma. \tag{36}$$

Eqs. (32), (33) and (35) can be rewritten as

$$\xi^+ = \left(e_0 - \underline{Y}\left(\underline{X}(\hat{w_1} + \hat{w_2}) + e_1(\hat{b_1} - \hat{b_2})\right)\right)_+, \tag{37}$$

$$\xi^- = \left(e_0 - \underline{Y}\left(\underline{X}(\hat{w_1} - \hat{w_2}) + e_1(\hat{b_1} + \hat{b_2})\right)\right)_+, \tag{38}$$

$$\tau = \left(k\|\hat{w_2}\|_2^2 - \|\hat{w_1}\|_2^2\right)_+, \tag{39}$$

where $(\cdot)_+$ replaces negative components of a vector by zeros.

Smoothing technique is introduced to replace x_+ by a given $p(x, \mu)$, where

$$p(x, \mu) = x + \frac{1}{\mu}\log(1 + e^{-\mu x}), \quad \mu > 0. \tag{40}$$

Consequently, (31) is rewritten as

$$\begin{aligned}\text{minimize} \quad & \frac{1}{2}\|\hat{w_1}\|_2^2 + C_\varsigma \varsigma^2 + \frac{C_\xi}{2}\left(\left\|p\Big(e_0 - \underline{Y}\big(\underline{X}(\hat{w_1} + \hat{w_2}) + e_1(\hat{b_1} - \hat{b_2})\big), \mu\Big)\right\|_2^2\right. \\ & \left. + \left\|p\Big(e_0 - \underline{Y}\big(\underline{X}(\hat{w_1} - \hat{w_2}) + e_1(\hat{b_1} + \hat{b_2})\big), \mu\Big)\right\|_2^2\right) \\ & + \left\|p(k\|\hat{w_2}\|_2^2 - \|\hat{w_1}\|_2^2), \mu)\right\|_2^2. \end{aligned} \tag{41}$$

Besides, constraint $0 \leqslant \hat{w_1}^T \hat{w_2} \leqslant \varsigma$ can also be removed by rewriting (41).

$$\begin{aligned}\text{minimize} \quad & \frac{1}{2}\|\hat{w_1}\|_2^2 + C_\varsigma(\hat{w_1}^T \hat{w_2})^2 + \frac{C_\xi}{2}\left(\left\|p\Big(e_0 - \underline{Y}\big(\underline{X}(\hat{w_1} + \hat{w_2}) \right.\right. \\ & + e_1(\hat{b_1} - \hat{b_2})\big), \mu\Big)\Big\|_2^2 + \left\|p\Big(e_0 - \underline{Y}\big(\underline{X}(\hat{w_1} - \hat{w_2}) \right. \\ & \left.\left. + e_1(\hat{b_1} + \hat{b_2})\big), \mu\Big)\right\|_2^2\right) + \left\|p\big(k\|\hat{w_2}\|_2^2 - \|\hat{w_1}\|_2^2\big), \mu\big)\right\|_2^2 \end{aligned}$$

Now the nonlinear programming problem has been converted into unconstrained optimization problem as follows:

Given the training samples $\{(x_i, y_i)\}_{i=1}^N$, *find optimal* $\hat{w_1}, \hat{w_2}, \hat{b_1}, \hat{b_2}$ *such that minimize the cost function:*

$$\frac{1}{2}\|\hat{w_1}\|_2^2 + C_\varsigma(\hat{w_1}^T \hat{w_2})^2 + \frac{C_\xi}{2}\left(\left\|p\Big(e_0 - \underline{Y}\big(\underline{X}(\hat{w_1} + \hat{w_2})\right.\right.$$

$$+ e_1(\hat{b_1} - \hat{b_2})), \mu\Big)\Big\|_2^2 + \Big\|p\Big(e_0 - \underline{Y}\big(\underline{X}(\hat{w_1} - \hat{w_2}) + e_1(\hat{b_1} + \hat{b_2})), \mu\Big)\Big\|_2^2\Big) + \Big\|p\big(k\|\hat{w_2}\|_2^2 - \|\hat{w_1}\|_2^2\big), \mu\big)\Big\|_2^2 \quad (42)$$

Eq. 42 is a quadratic function. An optimization problem with a quadratic function is well known to be solvable by adopting the conjugate gradient algorithm (Bertsekas and Hardcover, 1999; Hagan *et al.*, 1996). When the conjugate gradient algorithm is adopted, (43) is used to obtain optimal solutions.

$$(\hat{w_1}(k+1), \hat{w_2}(k+1), \hat{b_1}(k+1)) = (\hat{w_1}(k), \hat{w_2}(k), \hat{b_1}(k)) + \alpha(k)P(k). \quad (43)$$

Here, α_k is a step length obtained by one-dimensional search for the direction of descent, the set of vectors $\{P_j\}$ gives the conjugate directions (Bertsekas and Hardcover, 1999).

In supervised learning, the solution to (42) can be solved by adopting batch-mode learning or pattern-mode learning. When batch-mode learning is adopted, all input samples are taken as a whole batch and the parameters are not updated until the entire batch of data is processed. (42) is a typical batch model. On the other hand, if pattern-mode learning is chosen, the parameters are adapted immediately after each input signal is fed in. Pattern-mode learning can be implemented by choosing the subset of the dataset given by S samples of the original N data samples, where $1 \leqslant S \leqslant N$. Herein, $\vec{X}_i$ is used to represent the variant structure matrix of $\underline{X}$ and $\vec{X}_i = \{x_i, x_{i+1}, \ldots, x_{i+S-1}\}$.

Theoretically, batch-mode learning is more precise and efficient than other methods of learning. However, it may be unrealizable when it requires huge memory. On the other hand, pattern-mode learning is easily implemented and performed in the same way batch-mode learning (Heskes and Wiegerinck, 1996; Qin *et al.*, 1992). Based on these properties, the proposed iterative algorithm can be adapted to datasets of various sizes.

Iterative Learning Algorithm for OSVMs:

In this algorithm, j represents the training step for the jth input signal. In (42) matrix $\underline{X}$ is replaced by $\vec{X}_j$. $\vec{X}_j$ is an $S \times m$ matrix and $\underline{Y}$ is an $S \times S$ matrix, where S is the number of input signals that are included in the j'th training.

Step 1. Start with any $\hat{w_1}(0), \hat{w_2}(0), \hat{b_1}(0)$, where $\hat{w_1}(0), \hat{w_2}(0) \in R^n$ and $\hat{b_1}(0) \in R$.

Step 2. In jth training step, using conjugate gradient algorithm to get the optimal weight vector $\hat{w_1}^{opt}, \hat{w_2}^{opt}, \hat{b_1}^{opt}$.

Step 3. Use the jth optimal weight vector that is generated in Step 2 to update the global optimal weight vector according to (44).

$$(\hat{w_1}^{j+1}, \hat{w_2}^{j+1}, \hat{b_1}^{j+1}) = (1 - \mu_i)(\hat{w_1}^{j}, \hat{w_2}^{j}, \hat{b_1}^{j}) + \mu_i(\hat{w_1}^{opt}, \hat{w_2}^{opt}, \hat{b_1}^{opt}). \quad (44)$$

5.2. *Iterative Learning Algorithm with Nonlinear Kernel*

A nonlinear kernel is used to obtain the nonlinear oblique support vector machines and construct a nonlinear classifier. After converting into the dual problem, the objection

function of the nonlinear OSVMs model is stated as follows:

$$\begin{aligned}\text{minimize} \quad & \frac{1}{2}\|\frac{\gamma_1-\gamma_2-2k\gamma_3}{(\gamma_1-\gamma_2)^2-2k\gamma_3+4k\gamma_3^2}\underline{X}^T\underline{Y}^T\alpha \\ & -\frac{\gamma_1-\gamma_2+2k\gamma_3}{(\gamma_1-\gamma_2)^2-2k\gamma_3+4k\gamma_3^2}\underline{X}^T\underline{Y}^T\beta\|_2^2 \\ & +C_\varsigma(\frac{\gamma_1}{2C_\varsigma})^2+\frac{C_\xi}{2}(\|\xi^+\|_2^2+\|\xi^-\|_2^2). \end{aligned} \tag{45}$$

Constrained functions are got by applying (25) and (26) to (32), (33), (35) and (36). Moreover, the smooth technique is introduced to converted the constrained linear programming problem into unconstrained optimization problem. Last, the product term $\underline{XX}^T$ is replaced with kernel function $K(\underline{X}, \underline{X})$. After that, optimal solution of can be got by adopting conjugate gradient method.

Another, one of important factors for constructing an iterative nonlinear classifier is to find representative samples from the entire database. Herein, $\vec{B}$ is used to represent the representative samples and the representative samples can be determined by partition clustering method, such as Forgy Algorithm (Anderberg, 1973). After replacing the entire database with representative samples, the size of the quadratic problem is greatly reduced and the characterization of the nonlinear separating surface is simplified by making use kernel $K(\vec{X}_i, \vec{B})$. The algorithm of determining representative samples is described as follows:

(i) According to the distribution of each class in the whole dataset, randomly choosing S samples as seed points.
(ii) Allocate each data unit to the cluster with the nearest seed point. The seed points remain fixed for a full cycle through the entire data set.
(iii) Compute new seed points as the centroids of the clusters of data units.
(iv) Alternate Steps (ii) and (iii) until no data units change their cluster membership. If this alternation does not end, stop after R alternations.

6. Experimental Results

This section demonstrates the effectiveness of Oblique Support Vector Machines. The experimental's results that pertain to OSVMs are compared to those for classical support vector machines, SSVMs and RSVMs to evaluate the effectiveness of the OSVMs, respectively (Lee and Mangasarian, 2001a; Lee and Mangasarian, 2001b; Vapnik, 1998).

In this work, six publicly available datasets are used as experimental samples. They include: two Wisconsin Prognostic Breast Cancer databases and four datasets, "Ionosphere", "Cleveland Heart Problem", "Pima Indians" and "BUPA Liver", from the Irvine Machine Learning Database Repository (Murphy and Aha, 1992).

At beginning, training samples and testing samples are normalized. The number of t is chosen by heuristics but must be satisfied those two constraints stated in Lemma 3. Obviously, t is the factor to control the length of radius and is independent to u_i and

λ_i. Accordingly, if t is chosen big enough, the optimal solution can be obtained. So a heuristic algorithm for choosing t is stated as follows:

1. Randomly selecting m training samples and setting t as the summation of m training samples' 1-norm.
2. Recording the experimental result.
3. Increasing the value of t and then training and testing again.
4. Comparing the experimental result. If the results are almost same, then stop. Otherwise, repeat Step 3.

First, tenfold cross-validation is performed on each dataset to evaluate how well each algorithm generalizes to future data (Stone, 1974). The method of tenfold cross-validation involves extracting a certain proportion, typically 10%, of the training set as the tuning set, which is a surrogate of the testing set. For each training, OSVMs are applied to the rest of the training data to obtain a classifier and the tuning set correctness of this classifier is computed. All parameters in the OSVMs algorithm are selected to optimize the performance on the tuning set, a surrogate for a test set. Table 1 lists the experimental's results. Clearly, in Table 1 the correctness of the classification on testing samples by OSVMs is better than that by SSVMs and SVMs. This result indicates that the ability of prediction of OSVMs is better than the others.

Second, a checkerboard dataset is used as testing data to determine the effectiveness of the OSVMs in generating a highly nonlinear separating hyperplane. Fig. 5 shows the results for OSVMs.

Finally, Adult dataset is used to demonstrate the correctness and efficient about iterative learning algorithm of OSVMs in solving larger problems (Murphy and Aha, 1992). In this experiment Gaussian kernel is adopt and smooth parameter μ is set to 5. In our

Table 1

Tenfold cross-validation correctness results on six moderately sized datasets using three different methods

Tenfold Training Correctness, % Tenfold Testing Correctness, %			
Dataset size	OSVM	SSVMs	SVM
WPBC (24 months)	86.59	86.16	81.94
155×32	91.61	83.47	82.02
WPBC (60 months)	77.5	80.20	80.91
110×32	77.8	68.18	61.83
Ionosphere	94.91	94.12	92.96
351×34	89.72	89.63	89.17
Cleveland	87.95	87.32	72.05
297×13	88	86.13	72.12
Pima Indians	75.40	78.11	77.92
768×8	80.37	78.12	77.07
BUPA Liver	70.8	70.37	70.57
345×6	71	70.33	69.86

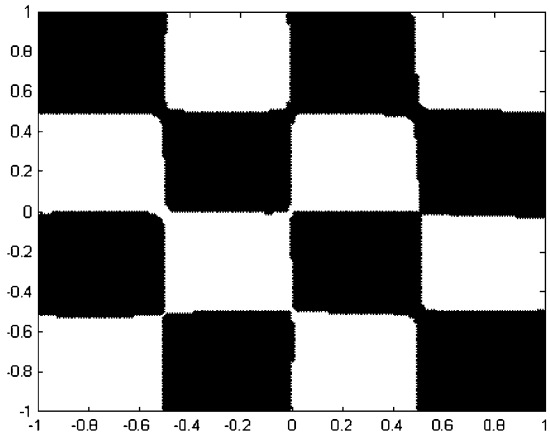

Fig. 5. Gaussian kernel separation of the checkerboard dataset.

Table 2
Computational results of OSVMs and RSVMs on Adult dataset

representative sample number	$K(\vec{A}_i, \vec{B})$		$K(\vec{B}, \vec{B})$	
	OSVMs	RSVMs	OSVMs	RSVMs
	Testing %	Testing %	Testing %	Testing %
81	84.58	84.29	78.81	77.93
114	84.39	83.88	78.88	74.64
160	84.61	84.56	77.81	77.74
192	84.61	84.55	79.01	76.93
210	84.65	84.47	79.12	77.03
242	84.77	84.90	79.96	75.45
284	85.07	85.31	79.99	76.73
326	85.07	85.07	80.12	76.95

algorithm, $S \times S$ storage capacity is required for kernel matrix rather than $N \times S$ storage capacity on RSVMs. Table 2 shows the correctness of pattern-mode learning with different numbers of input signals in each training epoch and compares the experimental results of OSVMs with RSVMs. The results indicate that the correctness of the classification by OSVMs is better than that by RSVMs and also indicate that, independently input signals in each training epoch, the same stable classification rate is approached.

7. Conclusion

In this paper a modified framework of support vector machines, called OSVMs, has been proposed to improve the capability of classification. For achieving an optimal solution, an orthogonal vector is joined into the weight vector in order to rotate the support hy-

perplanes. In constrast to the model of classical SVMs, the optimal separating hyperplane and the margin of separation are constructed more precisely. Six publicly available datasets are used as training and testing samples for simulation. Consequently, our experimental results definitely reveal the excellent ability of classification.

In addition, an iterative learning algorithm has been proposed to process the large-scale data problem. In this iterative learning algorithm, three different schemes for training can be found in this literature including pattern mode, batch mode and semi-batch mode learning. The main advantage of this iterative learning algorithm is greatly saving the storage capacity. In this iterative learning, the storage capacity for kernel matrix is only $H \times H$, whereas on RSVMs the storage capacity is $N \times H$, $H \ll N$. Experimental results have been given to demonstrate that the performance of OSVMs is better than that of RSVMs.

Appendix

Proof of Lemma 2

Due to $N \gg m$, without loss of generality, we can select the training samples $x_1, x_2, \ldots, x_N$ such that $dim(Span\{x_1, x_2, \ldots, x_N\}) = m$. Note that $Span(S)$ denote the subspace spanned by the set S (Leon, 2002). Therefore, the rank of $\underline{X}$ is equal to m. Note that again, $\underline{X} = \begin{bmatrix} x_1 & x_2 & \ldots & x_N \end{bmatrix}^T$. Consequently, $r(\vec{A}) = m$, where $\vec{A} = [\underline{X} \quad -e_1] = \begin{bmatrix} x_1 & x_2 & \ldots & x_N \\ -1 & -1 & \ldots & -1 \end{bmatrix}^T$. By Lemma 1, the rank of $\vec{A}^T\vec{A}$ is equal to m, where the structure of $\vec{A}^T\vec{A}$ can be specified as follows.

$$\vec{A}^T\vec{A} = \begin{bmatrix} \underline{X}^T\underline{X} & -\underline{X}^T e_1 \\ -e_1^T\underline{X} & e_1^T e_1 \end{bmatrix} = \begin{bmatrix} \underline{X}^T\underline{X} & -\sum_{i=1}^N x_i \\ -\sum_{i=1}^N x_i^T & N \end{bmatrix}. \tag{46}$$

Proof of Lemma 3

$$\begin{aligned} \vec{w_2}^T\vec{A}^T\vec{A}\vec{w_2} &= \vec{w_2}^T PDP^T\vec{w_2} = [w_2^T \quad b_2] \begin{bmatrix} P_{11}D_1P_{11}^T & P_{11}D_1P_{21}^T \\ P_{21}D_1P_{11}^T & P_{21}D_1P_{21}^T \end{bmatrix} \begin{bmatrix} w_2 \\ b_2 \end{bmatrix} \\ &= w_2^T P_{11}D_1P_{11}^T w_2 + w_2^T P_{11}D_1P_{21}^T b_2 + b_2 P_{21}D_1P_{11}^T w_2 + b_2 P_{21}D_1P_{21}^T b_2 \end{aligned}$$

After the coordinate is translated, the new coordinate $w_2' = P_{11}^T w_2$, where $w_2' = [w_{21}', w_{22}', \ldots, w_{2m}']^T$. Then

$$\vec{w_2}^T\vec{A}^T\vec{A}\vec{w_2} = t \quad \Rightarrow \quad w_2^{'T} D_1 w_2' + 2b_2 P_{21}D_1 w_2' + b_2 P_{21}D_1P_{21}^T b_2 = t. \tag{47}$$

Suppose $P_{21}D_1 = [u_1, u_2, \ldots, u_m]$, then (47) becomes

$$\begin{aligned} &\lambda_1 w_{21}^{'2} + \lambda_2 w_{22}^{'2} + \ldots + \lambda_m w_{2m}^{'2} + 2b_2(u_1 w_{21}' + u_2 w_{22}' + \ldots + u_m w_{2m}') \\ &\quad = t - b_2 P_{21}D_1P_{21}^T b_2 \end{aligned}$$

$$\Rightarrow \lambda_1(w'_{21} + \frac{b_2u_1}{\lambda_1})^2 + \ldots + \lambda_m(w'_{2m} + \frac{b_2u_m}{\lambda_m})^2$$
$$= t - b_2P_{21}D_1P_{21}^Tb_2 + \frac{b_2^2u_1^2}{\lambda_1} + \ldots + \frac{b_2^2u_m^2}{\lambda_m}. \quad (48)$$

The geometric form of (48) is an m-dimensional ellipsoid. Let $H = t - b_2P_{21}D_1P_{21}^Tb_2 + \frac{b_2^2u_1^2}{\lambda_1} + \ldots + \frac{b_2^2u_m^2}{\lambda_m}$, then (48) become,

$$\frac{(w'_{21} + \frac{b_2u_1}{\lambda_1})^2}{\frac{H}{\lambda_1}} + \frac{(w'_{22} + \frac{b_2u_2}{\lambda_2})^2}{\frac{H}{\lambda_2}} + \ldots + \frac{(w'_{2m} + \frac{b_2u_m}{\lambda_m})^2}{\frac{H}{\lambda_m}} = 1. \quad (49)$$

We can find that $(-\frac{b_2u_1}{\lambda_1}, -\frac{b_2u_2}{\lambda_2}, \ldots, -\frac{b_2u_m}{\lambda_m})(= O_1)$ is the center of the ellipsoid and $\sqrt{\frac{H}{\lambda_i}}$ is the length of radius in the direction ν_i where ν_i is the eigenvector corresponding to λ_i. In order to guarantee the origin O inside the ellipsoid, the distance between O and O_1 must be less than the length of any radius. Therefore, when t satisfies

(1) $t - b_2P_{21}D_1P_{21}^Tb_2 + \frac{b_2^2u_1^2}{\lambda_1} + \ldots + \frac{b_2^2u_m^2}{\lambda_m} > 0$,

(2) $\sqrt{(\frac{b_2u_1}{\lambda_1})^2 + (\frac{b_2u_2}{\lambda_2})^2 + \ldots + (\frac{b_2u_m}{\lambda_m})^2} < \min\left(\sqrt{\frac{H}{\lambda_1}}, \sqrt{\frac{H}{\lambda_2}}, \ldots, \sqrt{\frac{H}{\lambda_m}}\right)$,

the origin of coordinate is located inside the ellipsoid.

Proof of Proposition 1

(1) In (17), the constraint gradients are linearly independent.

(2) The Lagrangian function for (17) is listed as follow:

$$L(\hat{w_1}, \hat{w_2}, \hat{b_1}, \hat{b_2}) = \frac{1}{2}\|\hat{w_1}\|_2^2 - \alpha^T\left[\underline{Y}(\underline{X}(\hat{w_1} + \hat{w_2}) + (\hat{b_1} - \hat{b_2})e_1)\right]$$
$$- \beta^T\left[\underline{Y}(\underline{X}(\hat{w_1} - \hat{w_2}) + (\hat{b_1} + \hat{b_2})e_1)\right] + \gamma_1({\hat{w_1}}^T\hat{w_2})$$
$$- \gamma_2\left(\|\hat{w_1}\|_2^2 - k\|\hat{w_2}\|_2^2\right)$$
$$\Rightarrow \nabla^2 L(\hat{w_1}, \hat{w_2}, \hat{b_1}, \hat{b_2}) = \begin{bmatrix} (1-2\gamma_2)I & \gamma_1 I & 0 & 0 \\ \gamma_1 I & 2k\gamma_2 I & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$
$$\Rightarrow y^T\nabla^2 L(\hat{w_1}, \hat{w_2}, \hat{b_1}, \hat{b_2})y = (1-2\gamma_2)y_1^Ty_1 + 2\gamma_1y_1^Ty_2 + 2k\gamma_2y_2^Ty_2$$
$$= (1-2\gamma_2)\left\|y_1 + \frac{\gamma_1}{1-2\gamma_2}y_2\right\|^2 + 2k\gamma_2^2y_2^Ty_2 - \frac{\gamma_1^2}{1-2\gamma_2}y_2^Ty_2.$$

If the twice differential function is semi-positive, then the solution which is provided by Lagrangian function is global minimum. Hence, when the two conditions are satisfied: (1) $1 - 2\gamma_2 > 0$, (2) $2k\gamma_2^2 \geqslant \frac{\gamma_1^2}{1-2\gamma_2}$.

The solution of (17) which is provided by Lagrangian function is global minimum solution.

References

Anderberg, M.R. (1973). *Cluster Analysis for Applications*. Academic Press, New York.

Bennett, K.P., and O.L. Mangasarian (1992). Robust linear programming discrimination of two linearly inseparable sets. *Optimization Methods and Software*, **1**, 23–34.

Bertsekas, D.P., and Hardcover (1999). *Nonlinear Programming*. Massachusetts, Athena Scientific.

Bradley, P.S., and O.L. Mangasarian (1998). Feature selection via concave minimization and support vector machines. In J. Shavlik (Ed.), *Machine Learning Proceedings of the Fifteenth International Conference* (ICML'98). Morgan Kaufmann, San Francisco, CA, pp. 82–90.

Chapelle, O., P. Haffner, and V.N. Vapnik (1999). Support vector machines for histogram-based image classification. *IEEE Trans. on Neural Network*, **10**(5), 1055–1064.

Cherkassky, V., and F. Mulier (1998). *Learning from Data-Concepts, Theory and Methods*. John Wiley and Sons, New York.

Gill, P.E., W. Murray and M.H. Wright (1981). *Practical Optimization*. Academic Press, London.

Guo, G., S.Z. Li and K.L. Chan (2001). Support vector machines for face recognition. *Image and Vision Computing*, **19**, 631–638.

Hagan, M.T., H.B. Demuth and M.H. Beale (1996). *Neural Network Design*. Thomson, Boston.

Heskes, T., and W. Wiegerinck (1996). A theoretical comparison of batch-mode, on-line, cyclic, and almost-cyclic learning. *IEEE Trans. on Neural Networks*, **7**(4), 919–925.

Horn, R.A., and C.R. Johnson (1990). *Matrix Analysis*. Cambridge, New York.

Joachims, T. (1999). Making large-scale support vector machine learning practical. In B. Schölkopf, Ch.J.C. Burges and A.J. Smola (Eds.), *Advances in Kernel Methods – Support Vector Learning*. MIT Press, Camridge, MA, pp. 169–184.

Kreysig, E. (1999). *Advanced Engineering Mathematics*. John Wiley, New York.

Lee, Y.-J., and O.L. Mangasarian (2001a). SSVM: a smooth support vector machine for classification. *Computational Optimization and Application*, **20**, 5–22.

Lee, Y.-J., and O.L. Mangasarian (2001b). RSVM: Reduced support vector machines. In *Proceedings of the First SIAM International Conference on Data Mining*. Chicago, pp. 5–7.

Leon, S.J. (2002). *Linear Algebra with Applications*. Prentice-Hall, New Jersey.

Murphy, P.M., and D.W. Aha (1992). UCI repository of machine learning databases. `www.ics.uci.edu/ mlearn/MLRepository.html`

Platt, J. (1999). Sequential minimal optimization: a fast algorithm for training support vector machines. In B. Schölkopf, Ch.J.C. Burges, A.J. Smola (Eds.), *Advances in Kernel Methods – Support Vector Learning*. MIT Press, Camridge, MA, pp. 185–208.

Qin, S.Z., H.T. Su and T.J. McAvoy (1992). Comparison of four neural net learning methods for dynamic system identification. *IEEE Trans. on Neural Networks*, **3**(1), 122–130.

Schölkopf, B., and A.J. Smola (2002). *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, Mass: MIT Press, London.

Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions, *Journal of the Royal Statistical Society*, **36**, 111–147.

Vapnik, V.N. (1998). *Statistical Learning Theory*. Wiley, New York.

C.-C. Yao received his BS degree in computer science and information engineering from National Chiao Tung University, the MS degree in computer science and information engineering from National Cheng Kung University, and the PhD degree in computer science and information engineering from Nation Chung Cheng University, Chiayi, Taiwan. Currently, he is an assistant professor with the Department of Computer Science and Information Engineering, Chaoyang University of Technology, Taichung Country, Taiwan. His research interests include possibility reasoning, machine learning, data mining, and fuzzy inference system.

P.-T. Yu received the BS degree in mathematics from National Taiwan Normal University in 1979, the MS degree in computer science from National Taiwan University, Taipei, Taiwan, in 1985, and the PhD degree in electrical engineering from Purdue University, West Lafayette, IN, USA, in 1989. Since 1990, he has been with the Department of Computer Science and Information Engineering at National Chung Cheng University, Chiayi, Taiwan, R.O.C., where he is currently a professor. His research interests include e-learning, neural networks and fuzzy systems, nonlinear filter design, intelligent networks, and XML technology.

Pakrypusio atraminio vektoriaus klasifikatorius

Chih-Chia YAO, Pao-Ta YU

Šiame straipsnyje klasifikavimo galimybių pagerinimui pasiūlyta modifikuota atraminio vektoriaus klasifikatoriaus schema, pavadinta pakrypusio atraminio vektoriaus klasifikatoriais (OSVMs). Šio klasifikatoriaus principas yra ortogonalaus vektoriaus įtraukimas į svorių vektorių atraminių hiperplokštumų pasukimui. Tokiu būdu ne tik rizikos funkcija yra pataisoma, bet ir ribojimų funkcijos yra modifikuojamos. Po šio modifikavimo, skiriančioji hiperplokštuma ir skiriamasis kraštas yra tiksliau sudaromi. Be to iteracinis mokymo algoritmas yra pasiūlytas taikymui dideliems duomenims. Pateikti eksperimentiniai rezultatai ir palyginimai, parodantys, kad OSVMs yra pajėgesni už SVMs ir SSVMs.