Machine Learning, , 1–21 (2004)

Multicategory Proximal Support Vector Machine Classifiers

GLENN M. FUNG glenn.fung@siemens.com
Computer-Aided Diagnosis & Therapy Solutions
Siemens Medical Solutions, Inc
51 Valley Stream Parkwa
Malvern, PA 19355

O. L. MANGASARIAN olvi@cs.wisc.edu
Computer Sciences Department
University of Wisconsin
Madison, WI 53706
Department of Mathematics
University of California at San Diego
La Jolla, CA 92093

Received July 2001; Revised October 2002 and November 2003

Editor: Shai Ben-David

Abstract. Given a dataset, each element of which labeled by one of k labels, we construct by a very fast algorithm, a k-category proximal support vector machine (**PSVM**) classifier. Proximal support vector machines and related approaches [13, 32] can be interpreted as ridge regression applied to classification problems [11]. Extensive computational results have shown the effectiveness of PSVM for two-class classification problems where the separating plane is constructed in time that can be as little as two orders of magnitude shorter than that of conventional support vector machines. When PSVM is applied to problems with more than two classes, the well known one-from-the-rest approach is a natural choice in order to take advantage of its fast performance. However, there is a drawback associated with this one-from-the-rest approach. The resulting two-class problems are often very unbalanced, leading in some cases to poor performance. We propose balancing the k classes and a novel Newton refinement modification to PSVM in order to deal with this problem. Computational results indicate that these two modifications preserve the speed of PSVM while often leading to significant test set improvement over a plain PSVM one-from-the-rest application. The modified approach is considerably faster than other one-from-the-rest methods that use conventional SVM formulations, while still giving comparable test set correctness.

keywords: multicategory data classification, support vector machines, proximal classifiers

1. Introduction

Standard support vector machines (SVMs) [36, 8, 4, 6, 23], which are powerful tools for data classification, classify 2-category points by assigning them to one of two disjoint halfspaces in either the original input space of the problem for linear classifiers, or in a higher dimensional feature space for nonlinear classifiers [36, 8, 23]. Recently [32, 13] much simpler classifiers, the least squares and the proximal support vector machine (PSVM), were implemented wherein each class of points is assigned to the closest of two parallel planes (in input or feature space) that are pushed apart as far as possible. This formulation, which can also be interpreted as regularized least squares [34] or in the more general context of regularized networks [11], leads to an extremely fast and simple algorithm for generating a linear or nonlinear classifier that is obtained by solving a single system of linear equations. For a comprehensive approach to the related least squares support vector machines see [30], where geometric and statistical interpretations as well as the link with the Fischer discriminant analysis are given. It is the purpose of this work to apply this simple 2-class PSVM classifier to k-category classification by using a one-from-rest (OFR) separation for each class [3]. However, due to the fact that the number of points belonging to one class is usually much smaller than the number of points in the union of the remaining classes, the resulting two-class problems are very unbalanced. PSVM fits each class with one of two distant parallel planes and errors in both classes are penalized similarly in the objective function. Because of the unbalanced classes, PSVM tends to fit better the the class with more data points and it underestimates the overall error of the class with fewer data points. This often results in a poor PSVM performance. In order to override this difficulty, we propose a balanced modification of PSVM which weights each class equally no matter how many points are in each class. In addition, we propose a very fast Newton refinement algorithm, which is applicable to any SVM classification approach, and which leads to a better classifier. Experimental results show that incorporation of these two modifications into a plain PSVM one-from-the-rest approach, improves significantly test set correctness while maintaining its speed.

In contrast, other one-from-the-rest and SVM k-class classifiers [3, 2, 5] require the solution of either a large single or k smaller quadratic or linear programs that need specialized optimization codes such as CPLEX [9]. On the other hand, obtaining a linear or nonlinear PSVM classifier as we propose here, requires nothing more sophisticated than solving k systems of linear equations. Efficient and fast linear equation solvers are freely available [1] or are part of standard commercial packages such as MATLAB [24], and can solve very large systems. We note that in [33, 31], multiclass least squares formulations are proposed that explicitly require Mercer's positive definiteness condition [36, 8] on the kernels used which is not needed here. Another way to avoid the need for Mercer's condition is to use the product of an arbitrary kernel with its transpose as was proposed in [23, Problem (8.10)]. In addition, the problem in [33] is formulated as single large constrained optimization problem in contrast to the k smaller uncoupled and unconstrained OFR approach used here. Various multiclass schemes are investigated in [15, 37]. We also note that, in concept, PSVM can be interpreted as ridge regression [17] which is essentially regularized least squares [34]. However, ridge regression in its general form lacks the geometric justification and interpretation of PSVM which consists of constructing two parallel planes, each proximal to one of two classes of data points, while simultaneously pushing these plane as far apart as possible. A ridge regression application similar to PSVM is given in [35], which however uses a variation of the EM-algorithm to solve the classification problem, whereas we use a straightforward solution of the normal equations of regularized least squares. Interesting numerical comparison of multiclass methods is given in [18].

We summarize the contents of the paper now. In Section 2 we briefly review the 2-category proximal linear support vector machine [13] and then introduce our multicategory PSVM (MPSVM). MPSVM for a k-class problems consists of solving k systems of nonsingular linear eqautions. We then give the linear MPSVM algorithm. In Section 3 we introduce the nonlinear MPSVM with nonlinear separating surfaces in the input space and give the corresponding nonlinear MPSVM algorithm. In Section 2.3 we describe a simple 2-dimensional Newton refinement of the algorithms presented in Sections 2 and 3. Section 4 contains numerical test results on six public data sets for both the linear and nonlinear MPSVM. These tests show a speedup of as high as 477-times, for our nonlinear MPSVM over conventional SVM, with comparable or better test set correctness (Table 2, Segment Dataset). These tests also show that a linear MPSVM with balancing and a Newton refinement can improve tenfold test set correctness over a plain MPSVM from 83.3% to 97.3% (Table 1, Iris Dataset). Simple and short MATLAB [24] codes, very similar to those of PSVM [13], underly the proposed MPSVM algorithms. Finally, we give a 2-dimensional visual example that demonstrates the effectiveness of our balancing and Newton refinements on a nonlinear classifier for a 3-class dataset, and exhibit the computed classifiers in Figures 5 and 6.

A word about our notation and background material. All vectors will be column vectors unless transposed to a row vector by a prime superscript $'$. For a vector x in the n-dimensional real space R^n, the plus function x_+ defines a vector function of x with all negative components of x set to zero, while the step function x_* defines a vector function of x with all positive components set to 1 and nonpositive components of x set to zero. The scalar (inner) product of two vectors x and y in the n-dimensional real space R^n will be denoted by $x'y$ and the 2-norm of x will be denoted by $\|x\|$. For a matrix $A \in R^{m\times n}$, A_i is the ith row of A which is a row vector in R^n, while $A_{\cdot j}$ is the jth column of A. A column vector of ones of arbitrary dimension will be denoted by e. For $A \in R^{m\times n}$ and $B \in R^{n\times k}$, the kernel $K(A, B)$ maps $R^{m\times n} \times R^{n\times k}$ into $R^{m\times k}$. In particular, if x and y are column vectors in R^n then, $K(x', y)$ is a real number, $K(x', A')$ is a row vector in R^m and $K(A, A')$ is an $m \times m$ matrix. The base of the natural logarithm will be denoted by ε. We will make use of the following Gaussian kernel [36, 8, 23] that is frequently used in the SVM literature:

$$K(A, B) = \varepsilon^{-\mu\|A_i{}' - B_{\cdot j}\|^2},\ i = 1\ldots, m,\ j = 1\ldots, k, \tag{1}$$

where $A \in R^{m\times n}$, $B \in R^{n\times k}$ and μ is a positive constant. The identity matrix of arbitrary dimension will be denoted by I. For a numerical function $f(x)$ of $x \in R^n$, the gradient $\nabla f(x)$ denotes the $n \times 1$ vector of first partial derivatives of f, while $\partial^2 f(x)$ denotes the generalized Hessian $n \times n$ matrix of second partial derivatives of f if they exist, else each row of the generalized Hessian matrix is a subgradient [26, 27] of the corresponding row element of the gradient vector $\nabla f(x)$ [21, 19].

2. The Linear Multicategory Proximal Support Vector Machine (MPSVM)

2.1. Two-Category Proximal Support Machine Formulation

To motivate our MPSVM we begin with a brief description of the 2-category proximal support machine formulation [13]. We consider the problem, depicted in Figure 1, of classifying m points in the n-dimensional real space R^n, represented by the $m \times n$ matrix A, according to membership of each point A_i in the class $A+$ or $A-$ as specified by a given $m \times m$ diagonal matrix D with plus ones or minus ones along its diagonal. For this problem, the proximal support vector machine [13] with a linear kernel is given by the following quadratic program with parameter $\nu > 0$ and linear *equality* constraint:

$$\begin{array}{rl} \displaystyle\min_{(w,\gamma,y)\in R^{n+1+m}} & \frac{\nu}{2}\|y\|^2 + \frac{1}{2}\left\|\begin{bmatrix} w \\ \gamma \end{bmatrix}\right\|^2 \\ \text{s.t.} & D(Aw - e\gamma) + y \;=\; e, \end{array} \tag{2}$$

where e is a vector of ones. As depicted in Figure 1, $\begin{bmatrix} w \\ \gamma \end{bmatrix}$ is normal to the *proximal planes*:

$$\begin{array}{rcl} x'w - 1\cdot\gamma &=& +1, \\ x'w + 1\cdot\gamma &=& -1, \end{array} \tag{3}$$

which are proximal to points belonging to the sets $A+$ and $A-$ respectively. The error variable y in (2) is a measure of the distance from the plane $x'w - 1\cdot\gamma = +1$ of points of class $A+$ points and from the plane $x'w - 1\cdot\gamma = -1$ of points of class $A-$. Consequently, the plane:

$$x'w - 1\cdot\gamma = 0, \tag{4}$$

midway between and parallel to the proximal planes (3), is a *separating plane* that approximately separates $A+$ from $A-$ as depicted in Figure 1. (The separation is only approximate, here and in general, because no plane can separate all points of $A+$ from those of $A-$ when their convex hulls intersect.) The second term in the quadratic objective function of (2), which is twice the reciprocal of the square of the 2-norm distance $\frac{2}{\|\begin{bmatrix} w \\ \gamma \end{bmatrix}\|}$ between the two proximal planes of (3) (see Figure 1), maximizes this distance, often called the "margin". Maximizing the margin enhances the generalization capability of a support vector machine [36, 8]. The approximate separating plane (4) as depicted in Figure 1, acts as a linear classifier as follows:

$$x'w - \gamma \begin{cases} > 0, & \text{then } x \in A+, \\ < 0, & \text{then } x \in A-, \\ = 0, & \text{then } x \in A+ \text{ or } x \in A- \end{cases} \tag{5}$$

We note that the PSVM formulation (2) can be also interpreted as a regularized least squares solution [34] of the system of linear *equations* $D(Aw - e\gamma) = e$, that

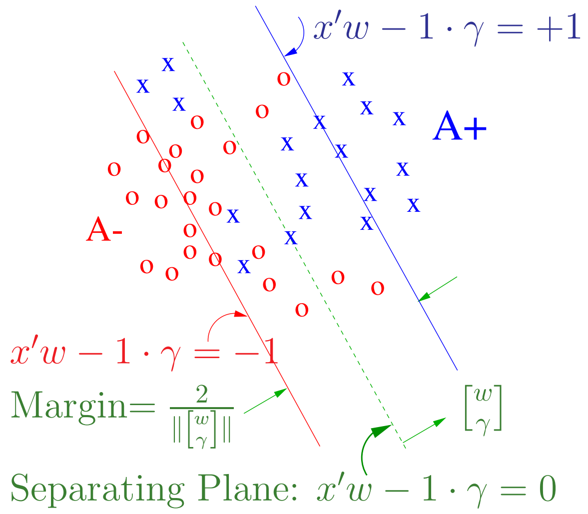

Figure 1. **The Proximal Support Vector Machine Classifier: The *proximal planes* $x'w - \gamma = \pm 1$ around which points of the sets $A+$ and $A-$ cluster and which are pushed apart by the optimization problem (2).**

is finding an approximate solution (w, γ) to $D(Aw - e\gamma) = e$, with least 2-norm. PSVM can also be considered as a very special case of regularization networks [11].

Substituting for y in terms of w and γ from the linear constraint in the objective function of (2) gives the *unconstrained* minimization problem:

$$\min_{(w,\gamma) \in R^{n+1}} \frac{\nu}{2} \|D(Aw - e\gamma) - e\|^2 + \frac{1}{2} \left\| \begin{bmatrix} w \\ \gamma \end{bmatrix} \right\|^2, \tag{6}$$

Setting the gradient with respect to w and γ to zero and noting that $D^2 = I$ gives the necessary and sufficient optimality conditions for (6):

$$\begin{aligned} \nu A'(Aw - e\gamma - De) + w &= 0, \\ \nu e'(-Aw + e\gamma + De) + \gamma &= 0. \end{aligned} \tag{7}$$

2.2. PSVM Modification for Unbalanced Classes

In order to improve PSVM performance when one of classes has many more data points than the other one, which is usually the case in the two-class subproblems that the OFR approach generates, we propose the following simple balancing approach. A similar balancing approach was proposed in [14].

Let m_1 and m_2 be the number of points in classes 1 and -1 respectively. We first define an $m \times m$ diagonal matrix N as follows:

$$N_{ii} = \begin{cases} \frac{1}{m_1}, & \text{if } d_{ii} = 1, \\ \frac{1}{m_2}, & \text{if } d_{ii} = -1. \end{cases} \tag{8}$$

We then formulate the following *balanced* PSVM problem:

$$\min_{(w,\gamma)\in R^{n+1}} \frac{\nu}{2}(D(Aw - e\gamma) - e)'N(D(Aw - e\gamma) - e) + \frac{1}{2}\left\|\begin{bmatrix} w \\ \gamma \end{bmatrix}\right\|^2. \tag{9}$$

Setting the gradient with respect to w and γ equal to zero and noting that $D^2 = I$ and $DND = N$ we obtain the following necessary and sufficient optimality conditions for (9):

$$\begin{array}{rcl} \nu A'N(Aw - e\gamma - De) + w & = & 0, \\ \nu e'N(-Aw + e\gamma + De) + \gamma & = & 0. \end{array} \tag{10}$$

We describe now a computational enhancement to PSVM which is also applicable to other SVM classifiers as well.

2.3. **Newton Refinement**

The simple computational refinement that we have implemented, and which is applicable to any type of SVM classifier, consists of taking a solution obtained by either a linear or nonlinear classifier, say for simplicity a solution $\begin{bmatrix} \bar{w} \\ \bar{\gamma} \end{bmatrix}$ to the PSVM problem (6), which generates a separating plane $x'\bar{w} - 1 \cdot \bar{\gamma} = 0$ as shown in Figure 1. The idea here is to move this plane parallel to itself in such a way to improve the separation of the two sets $A+$ and $A-$. One way to measure such improvement is by counting the number of misclassified points as was done in [7]. A simpler way is to slightly alter the objective function of (6) so that the first term is zero if all the points are correctly classified by the separating plane. This is easily achieved by setting nonnegative components of $D(Aw - e\gamma) - e$, which correspond to correctly classified points, equal to zero, that is: $(-D(Aw-e\gamma)+e)_+ = 0$, where as defined in the Introduction, $(z)_+ = \max\{0, z\}$. Thus the minimization problem (6) becomes:

$$\min_{(w,\gamma)\in R^{n+1}} \frac{\nu}{2}\|((-D(Aw - e\gamma) + e)_+\|^2 + \frac{1}{2}\left\|\begin{bmatrix} w \\ \gamma \end{bmatrix}\right\|^2, \tag{11}$$

which is the optimization problem underlying the smooth support vector machine algorithm [21]. Since we are only interested in merely refining our solution while maximizing the margin $\begin{bmatrix} \bar{w} \\ \bar{\gamma} \end{bmatrix}$ of (6), we replace w by $\lambda\bar{w}$ in (11) and obtain our refinement problem:

$$\min_{(\lambda,\gamma)\in R^{n+1}} f(\lambda,\gamma) = \frac{\nu}{2}\|((-D(\lambda A\bar{w} - e\gamma) + e)_+\|^2 + \frac{1}{2}\left\|\begin{bmatrix} \lambda\bar{w} \\ \gamma \end{bmatrix}\right\|^2. \tag{12}$$

This is a simple strongly convex problem in the 2-dimensional space of (λ, γ), that is its objective function has a positive definite generalized Hessian [16, 22] which can be very quickly minimized by a fast Newton method. The quadratic convergence and effectiveness of a Newton method for such a problem has been established in [21] for the full problem (11) in the $n+1$ dimensional space (w,γ). We briefly describe the approach proposed here for minimizing (12) now. We first need the expressions for the gradient and generalized Hessian matrix [12, 16] of $f(\lambda,\gamma)$ as follows. We first define:

$$d(\lambda,\gamma) = (-D(\lambda A\bar{w} - e\gamma) + e), \tag{13}$$

then the 2×1 gradient and the 2×2 generalized Hessian matrix of $f(\lambda,\gamma)$ are given by:

$$\nabla f(\lambda,\gamma) = \begin{bmatrix} -\nu\bar{w}'A'D(d(\lambda,\gamma))_+ + \parallel \bar{w} \parallel^2 \lambda \\ \nu e'D(d(\lambda,\gamma))_+ + \gamma \end{bmatrix}, \tag{14}$$

and,

$$\partial^2 f(\lambda,\gamma) = \begin{bmatrix} \nu\bar{w}'A'EA\bar{w} + \parallel \bar{w} \parallel^2 & -\nu\bar{w}'A'Ee \\ -\nu e'EA\bar{w} & \nu e'Ee + 1 \end{bmatrix}, \tag{15}$$

where E is the diagonal matrix:

$$E = Ddiag((d(\lambda,\gamma))_*)D = diag((d(\lambda,\gamma))_*), \tag{16}$$

and the $(\cdot)_*$ is the step function defined in the Introduction and which is taken here as a specific subgradient [27, 26] of the plus function $(\cdot)_+$ and is used to generate the generalized Hessian matrix in the same manner as in [21, 19].

A key difference between PSVM and SVM, is that with PSVM the conventional concept of support vectors (the data points corresponding to the positive multipliers) does not hold [13]. However, it is interesting to note that after this refinement is applied to the PSVM solution, the concept of support vectors applies to the new solution. If the pair (λ^*,γ^*) is the solution obtained by (12), then the corresponding dual multipliers associated with this problems are given by [21]:

$$u = (-D(\lambda^* A\bar{w} - e\gamma^*) + e)_+. \tag{17}$$

Then, the support vectors for the problem (12) are the data points of A corresponding to positive components of u.

The Newton refinement procedure can then be summarized as follows, where the iteration maximum of 30 and the tolerance of $\leq 10^{-3}$ are empirically arrived at.

Algorithm 1 Newton Refinement *Given a solution* $\begin{bmatrix}\bar{w}\\ \bar{\gamma}\end{bmatrix}$ *to the PSVM 2-class problem (6) refine it as follows:*

(i) Start with $\lambda^0 = 1$ *and* $\gamma^0 = \bar{\gamma}$.

(ii) Iterate (iii) until either $j = 30$ *or:*

$$\left\| \begin{bmatrix}\lambda^j\\ \gamma^j\end{bmatrix} - \begin{bmatrix}\lambda^{j+1}\\ \gamma^{j+1}\end{bmatrix} \right\| \leq 10^{-3}, \tag{18}$$

in which case $\begin{bmatrix}w\\ \gamma\end{bmatrix} = \begin{bmatrix}\lambda^{j+1}\bar{w}\\ \gamma^{j+1}\end{bmatrix}$ *is the refined solution to (6).*

(iii) Calculate the new iterates:

$$\begin{bmatrix}\lambda^{j+1}\\ \gamma^{j+1}\end{bmatrix} = \begin{bmatrix}\lambda^{j}\\ \gamma^{j}\end{bmatrix} - \nabla^2 f(\lambda^j,\gamma^j)^{-1}\nabla f(\lambda^j,\gamma^j). \tag{19}$$

With obvious modifications this algorithm can be applied to refine a solution $\left[\begin{smallmatrix}\bar{u}\\ \bar{\gamma}\end{smallmatrix}\right]$ of the nonlinear PSVM (25) as well.

In order to illustrate the proposed modifications we generated a small unbalanced artificial two-dimensional two-class dataset. The dataset consist of 100 points, 85 of which are in class $A+$ and 15 points in class $A-$. When the problem is solved using plain PSVM (6), the influence of the 85 points in class $A+$ prevails over that of the much smaller set of data points in $A-$. As a result, 14 out of 15 points in class $A-$ are misclassified. The total training set correctness is 86%, with only 6.6% correctness for the smaller class $A-$ and 100% correctness for the larger class $A+$. The resulting separating plane is shown in Figure 2. When a balanced PSVM (9) is used we can see an improvement over the plain PSVM, in the sense that a separating plane is obtained that correctly classifies *all* the points in class $A-$. However due to the significant difference in the cardinality of the two classes and the distribution of their points, a subset of 16 points in class $A+$ is now misclassified. The total training set correctness is 84%, with 100% correctness for $A-$ points and 81.2% correctness for $A+$ points. The resulting separating plane is shown in Figure 3. If now in addition to balancing, the Newton refinement is also applied, we obtain a separating plane that misclassifies only two points. The total training set correctness is 98%. The resulting separating plane is shown in Figure 4.

To extend this formulation to k classes, all we need is to redefine the following for separating class r from the rest:

$$\begin{aligned} A &= \begin{bmatrix} A^1\\ \vdots\\ A^k\end{bmatrix},\\ A+ &= A^r,\\ A- &= \begin{bmatrix} A^1\\ \vdots\\ A^{r-1}\\ A^{r+1}\\ \vdots\\ A^k\end{bmatrix},\\ r &\in \{1,\ldots,k\}, \end{aligned} \tag{20}$$

where, $A^r \in R^{m^r\times n}$ represents the m^r points in class r. We then define for $m = m^1 + \ldots + m^k$ the $m \times m$ diagonal matrix D of ± 1 as follows:

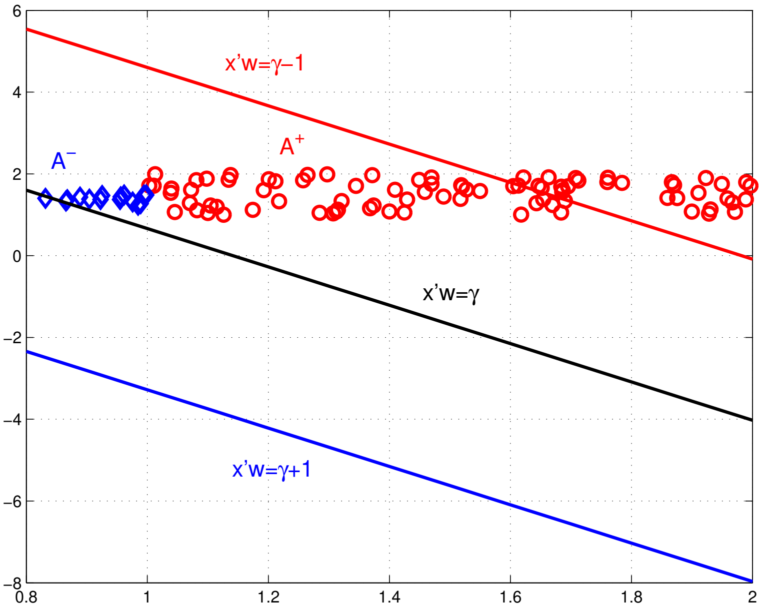

Figure 2. **An unbalanced dataset consisting of 100 points, 85 of which in class $A+$ represented by hollow circles, and 15 points of which in class $A-$ represented by hollow diamonds. The separating plane is obtained by using a plain PSVM (6). The class $A-$ is practically ignored by the solution. The total training set correctness is 86% with 6.6% correctness for $A-$ and 100% correctness for $A+$.**

$$\begin{aligned} D_{ii} &= 1 \text{ for } A_i \in A^r, \\ D_{ii} &= \text{-1 for } A_i \notin A^r, \\ r &\in \{1, \ldots, k\}. \end{aligned} \tag{21}$$

We note that since the multicategory classification problem, $A-$ has many more rows than $A+$, a normalization is usually carried out by dividing the error vector y_i by m^r for $A_i \in A^r$ and by $(m - m^r)$ for $A_i \notin A^r$. Here, m^r is the number of points in class r which is represented by the matrix $A^r \in R^{m^r \times n}$.

Once the k minimization problems (6) are solved (with A and D defined as in (20) and (21)) by solving the linear system of equations (10), k unique separating planes are generated:

$$x'w^r - \gamma^r = 0, \quad r = 1, \ldots, k. \tag{22}$$

A given new point $x \in R^n$ is assigned to class s, depending on which of the k halfspaces generated by the k planes (22) it lies deepest in, that is:

$$x'w^s - \gamma^s = \max_{r=1,\ldots,k} x'w^r - \gamma^r. \tag{23}$$

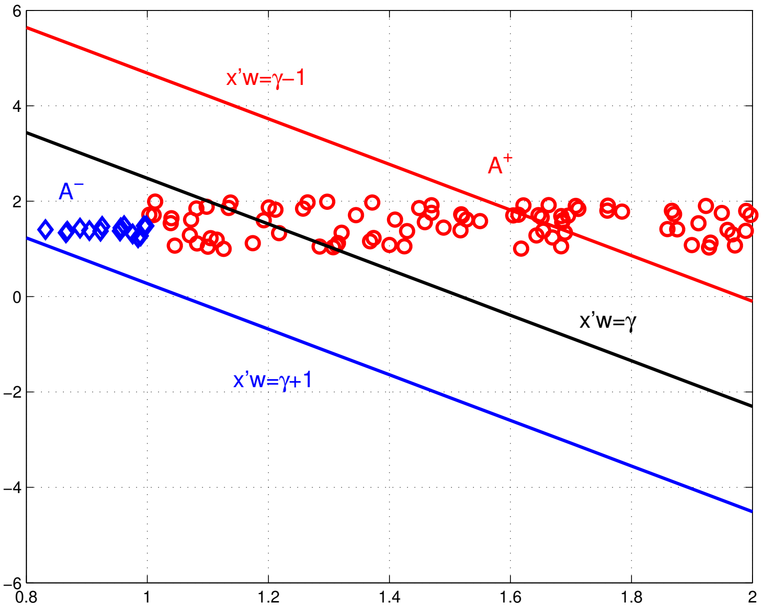

Figure 3. **Linear classifier improvement by balancing is demonstrated on the same dataset of Figure 2. The separating plane is obtained by using a balanced PSVM (9). Even though the class** $A-$ **is correctly classified in its entirety, the overall performance is still rather unsatisfactory due to significant difference in the distribution of points in each of the classes. Total training set correctness is 84%.**

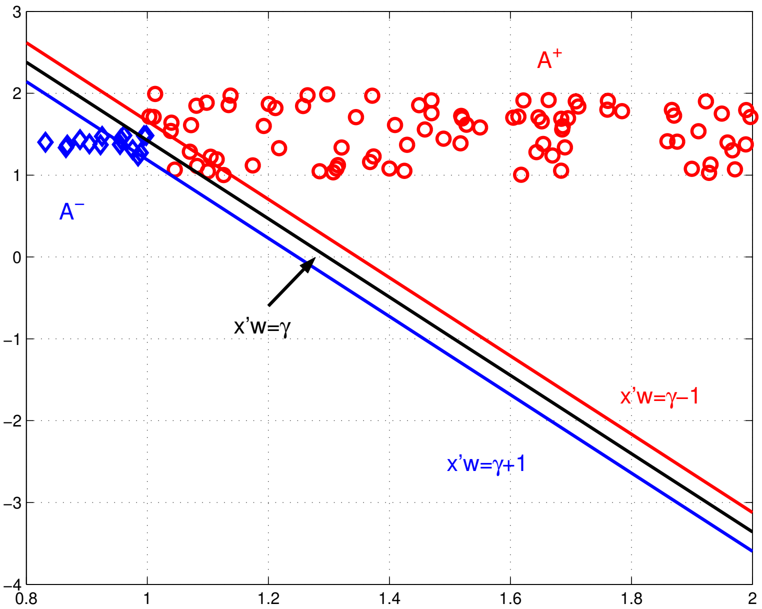

Figure 4. **Very significant linear classifier improvement as a consequence of balancing and the use of the Newton refinement is demonstrated on the same dataset of Figures 2 and 3. The separating plane is obtained using both modifications to PSVM: balancing and Newton refinement . The total training set correctness is now 98% compared to 86% for plain PSVM and 84% for balanced PSVM.**

For concreteness we explicitly state our multicategory PSVM algorithm.

Algorithm 2 Linear Multicategory Proximal SVM *Given m data points in R^n, each belonging to one of k classes and represented by k matrices A^r of order $m^r \times n$, $r = 1, \ldots, k$, with $m^1 + \ldots + m^k = m$, we generate the linear classifier (23) as follows:*

(i) Solve k independent nonsingular systems of $(n+1)$ linear equations (10) in $(n+1)$ unknowns, with A and D defined as in (20) and (21), for some positive value of ν. (Typically ν is chosen by means of a tuning set.)

(ii) Apply the Newton Refinement 1 to each solution $(\bar{w}^r, \bar{\gamma}^r)$, $(r = 1 \ldots k)$ obtained on step (i) to get the refined solutions (w^r, γ^r) .

(iii) The point x belongs to class s as determined by the criterion (23).

We extend now the above results to nonlinear proximal support vector machines that result in nonlinear proximal surfaces instead of planes in the input space.

3. Nonlinear Proximal Support Vector Machines

To obtain our nonlinear proximal classifier we modify our proximal minimization problem (6) as in [23, 13] by first replacing the primal variable w by its dual equivalent , $w = A'Du$, and modifying the last term of the objective function to be the norm of the new dual variable u and γ. This is based on the duality theory underlying support vector machines described in [23]. We obtain then the following problem:

$$\min_{(u,\gamma)\in R^{m+1}} \frac{\nu}{2}\|D(AA'Du - e\gamma) - e\|^2 + \frac{1}{2}\left\|\begin{bmatrix} u \\ \gamma \end{bmatrix}\right\|^2 . \tag{24}$$

If we now replace the linear kernel AA' by a nonlinear kernel $K(A, A')$, as defined in the Introduction, we obtain:

$$\min_{(u,\gamma)\in R^{m+1}} \frac{\nu}{2}\|D(K(A,A')Du - e\gamma) - e\|^2 + \frac{1}{2}\left\|\begin{bmatrix} u \\ \gamma \end{bmatrix}\right\|^2 . \tag{25}$$

As in the linear kernel case, we extend the above two category case to k categories by redefining A and D as in (20) and (21) to obtain k minimization problems. Setting the gradient with respect to u and γ to zero and noting again that $D^2 = I$ gives the following necessary and sufficient optimality conditions for (25):

$$\begin{aligned} \nu D(K(A,A')'K(A,A')Du - e\gamma - De) + u &= 0, \\ \nu e'(-K(A,A')Du + e\gamma + De) + \gamma &= 0. \end{aligned} \tag{26}$$

Once the k minimization problems (25) are solved (with A and D defined as in (20) and (21)) by solving the k independent linear systems of equations (26), k unique proximal surfaces are generated:

$$K(x', A')Du^r - \gamma^r = 0, \quad r = 1, \ldots, k. \tag{27}$$

A given new point $x \in R^n$ is assigned to class s depending on which of the k nonlinear halfspaces generated by the k surfaces (27) it lies deepest in, that is:

$$K(x', A')Du^s - \gamma^s = \max_{r=1,\ldots,k} K(x', A')Du^r - \gamma^r. \tag{28}$$

For concreteness we explicitly state our multicategory nonlinear PSVM algorithm.

Algorithm 1 Nonlinear Multicategory Proximal SVM *Given m data points in R^n, each belonging to one of k classes and represented by k matrices A^r of order $m^r \times n$, $r = 1, \ldots, k$, with $m^1 + \ldots + m^k = m$, we generate the nonlinear classifier (28) as follows:*

(i) Solve k independent nonsingular systems of $(m+1)$ linear equations (26) in $(m+1)$ unknowns, with A and D defined as in (20) and (21), for some positive value of ν. (Typically ν is chosen by means of a tuning set.)

(ii) Apply the Newton refinement algorithm 1 to each solution $(\bar{u}^r, \bar{\gamma}^r)$, $(r = 1 \ldots k)$ obtained on step (i) to get the refined solutions (u^r, γ^r) .

(ii) The point x belongs to class s as determined by the criterion (28).

When each of the k subproblems become large enough so as not to fit in memory, then the $m \times m$ kernel $K(A, A')$ is replaced by the considerably smaller $m \times \bar{m}$ rectangular kernel $K(A, \bar{A}')$, where $\bar{A}$ consists of as little as 15% of randomly chosen rows of A. This leads to the extremely fast and effective Reduced Support Vector Machine (RSVM) algorithm as described in [20] and presented in Algorithm 2 below. The RSVM approach can be interpreted as a random projection approach [10]. Other related reduction approaches are given in [14, 28, 29, 38].

Algorithm 2 RSVM Algorithm

(i) Choose a random subset matrix $\bar{A} \in R^{\bar{m} \times n}$ of the original data matrix $A \in R^{m \times n}$. Typically $\bar{m}$ is 1% to 15% of m, and $\bar{A}$ consists of the union of random samples of each class that maintain the original relative sizes of the k classes.

(ii) Solve the following modified version of the PSVM (25) where A' **only** *is replaced by $\bar{A}'$ with corresponding $\bar{D} \subset D$:*

$$\min_{(\bar{u}, \gamma) \in R^{m+1}} \frac{\nu}{2} \|D(K(A, \bar{A}')\bar{D}\bar{u} - e\gamma) - e\|^2 + \frac{1}{2} \left\| \begin{bmatrix} \bar{u} \\ \gamma \end{bmatrix} \right\|^2, \tag{29}$$

which is equivalent to solving (25) with A' **only** *replaced by $\bar{A}'$.*

The separating k surface is given by (27) with A' replaced by $\bar{A}'$ as follows:

$$K(x', \bar{A}')\bar{D}\bar{u}^r = \gamma^r, \tag{30}$$

where $(\bar{u}, \gamma) \in R^{\bar{m}+1}$ is the unique solution of (29), and $x \in R^n$ is a free input space variable of a new point.

We turn now to our numerical results.

4. Numerical Implementation and Comparisons

All our computations were performed on the University of Wisconsin Data Mining Institute "locop1" machine, which utilizes a 400 Mhz Pentium II and allows a maximum of 2 Gigabytes of memory for each process. This computer runs on Windows NT server 4.0, with MATLAB 6 installed. Even though "locop1" is a multiprocessor machine, only one processor was used for all the experiments since MATLAB is a single threaded application and does not distribute any load across processors [24]. Our algorithms require the solution of k square systems of linear equations, where k is the number of classes to be classified. Each one of the linear systems of equations involved is of the size of the number of input attributes n plus one in the linear case, and of the size of the number of data points m plus one in the nonlinear case. When using a rectangular kernel [21], the size of the problem can be reduced from m to $\bar{m}$ with $\bar{m} < m$ for the nonlinear case.

The real life datasets used for our numerical tests are the following:

- Four publicly available datasets from the UCI Machine Learning Repository [25]: Wine, Glass, Iris, Vowel, with 3, 6, 3 and 11 categories respectively.
- Two publicly available datasets from the Statlog Project Databases, also available from UCI [25]: Vehicle and Segment, with 4 and 7 categories respectively.

Properties of each dataset such as number of points, number of features and number of classes are given in Table 1.

4.1. Numerical experiments using linear classifiers

We compared the performances of the methods described below.

- **Linear OFRQP**: One-From-Rest Quadratic Programming classifier using a standard support vector machine formulation for each subproblem and solved using a MATLAB-CPLEX interface [9]. CPLEX is a state of the art software widely employed to solve linear and quadratic programs that uses a finitely terminating pivotal method of solution.
- **Linear MPSVM** : Multicategory Proximal SVM One-From-Rest classifier using a Linear Proximal support vector machine (PSVM) for each subproblem. Usually, each one-from-rest problem is an unbalanced two-class classification problem. This means that the number of points $m-$ in $A-$ is much larger than the number of points $m+$ in $A+$. In order to address this problem, we apply balancing, which is, a weight factor added to each error term in the objective function of (6) that is inversely proportional to the number of points in each class. We call this MPSVM modification Balanced MPSVM (**B-MPSVM**) and is given in (9). In order to further improve the performance of B-PSVM, for each two-class classification subproblem we use the Newton Refinement 1. Although the refinement step is very simple and fast, in almost all the tested

cases this refinement combined with the balancing procedure improved test set correctness of the MPSVM by as much as 16.8% (Table 1, Iris). We called this MPSVM modification Balanced and Refined MPSVM (**BR-MPSVM**). The underlying method consists of solving a nonsingular system of linear equations.

The value of the parameter ν in each of these methods was chosen by using a tuning set extracted from the training set. In order to find an optimal value for ν the following tuning procedure was employed on each fold:

- A random tuning set of the the size of 10% of the training data was chosen and separated from the training set.
- Several SVMs were trained on the remaining 90% of the training data using values for ν equal to 2^i, where $i = 0, 1, \ldots, 25$.
- The value of ν that gave the best SVM correctness on the tuning set was chosen.
- A final SVM was trained using the chosen value of ν and all the training data. The resulting SVM was tested on the testing data.

The linear BR-MPSVM running time was often one order of magnitude less than the standard OFRQP time. Furthermore, there was no a significant statistical difference between both methods as far test set correctness was concerned. This is shown by the p-values obtained using a 95% confidence interval t-test for the tenfold test set correctness. Experiments indicated that both modifications, balancing and refinement achieved significant accuracy improvements over the plain MPSVM, while maintaining relatively fast performances. Testing set correctness, training set correctness, CPU times and p-values are given in Table 1.

4.2. Numerical experiments using nonlinear classifiers

For the nonlinear case, we compared again nonlinear OFRQP and nonlinear PSVM and its modifications. In all experiments, a Gaussian kernel was used. In order to find an optimal value for ν and the Gaussian kernel parameter μ, a tuning procedure similar to that employed for the linear case was employed. Values for ν where taken equal to 2^i, where $i = 5, 6, \ldots, 35$. Values for μ where taken equal to 2^i, where $i = -7, -6, \ldots, 1$. Since the difference between the plain MPSVM and the modified MPSVM was not significant, Table 2 shows comparisons between the following methods only:

- **Nonlinear OFRQP**: One-From-Rest Quadratic Programming classifier using a standard nonlinear support vector machine for each subproblem which is solved by a MATLAB-CPLEX that uses a finitely terminating pivotal method of solution.
- **Nonlinear BR-MPSVM** : Balanced Refined Multicategory PSVM One-From-Rest classifier using a nonlinear PSVM including both modifications, balancing and Newton refinement. The underlying method consists of solving a nonsingular system of linear equations.

Table 1. **OFRQP, MPSVM,B-MPSVM,BR-MPSVM linear classifier training correctness, tenfold testing correctness and running times. Execution times include tenfold training. Best results are in bold. The p-values were calculated with respect to OFRQP for tenfold testing correctness, using a t-test with 95% confidence interval.**

Data Set $m \times n$ # of Classes	OFRQP Train Test Time (Sec.)	MPSVM Train Test Time (Sec.) p-value	B-MPSVM Train Test Time (Sec.) p-value	BR-MPSVM Train Test Time (Sec.) p-value
Wine 178×13 3	**100.0%** 96.1 % 1.39	**100.0 %** 98.9 % **0.02** 0.20	99.9% 98.9% **0.02** 0.80	**100.0%** **99.4%** 0.11 0.10
Glass 214×9 6	**72.9 %** **67.2 %** 1.80	66.5 % 60.6 % **0.02** 0.19	68.29 % 61.6 % 0.03 0.28	68.9% 63.0 % 0.14 0.35
Iris 150×4 3	**98.7 %** **98.0 %** 0.73	85.6% 83.3% **0.02** $1.2e-6$	86.9 % 86.7 % **0.02** $2.0e-4$	97.6% 97.3% 0.11 0.66
Vowel 528×10 11	**68.7 %** 57.2 % 5.56	54.6% 45.5 % **0.05** $9.9e-3$	56.1% 47.0% **0.05** $1.8e(-2)$	64.5% **57.6%** 0.14 0.93
Vehicle 846×18 4	**83.3 %** **79.0 %** 2.88	79.1% 76.2 % **0.11** $8.8e-2$	81.0% 77.4% **0.11** 0.33	81.1 % 77.5 % 0.34 0.30
Segment 2310×19 7	**93.0 %** **91.9 %** 18.57	85.5% 84.8 % **0.22** $7.5e-7$	90.3% 90.1% 0.31 $2.2e(-2)$	91.3% 90.8% 0.67 0.14

Table 2. **Nonlinear OFRQP and Nonlinear BR-MPSVM training correctness, tenfold testing correctness and running times. Execution times include tenfold training. The p-values were calculated with respect to OFRQP for tenfold testing correctness, using a t-test with 95% confidence interval. For the Vehicle dataset, RSVM [20] with an 85% kernel reduction was used for the nonlinear MPSVM classifier here in order to obtain a smaller rectangular kernel problem that would fit in memory (2310×350 instead of 2310×2310). Similarly for the Segment dataset, RSVM with 85% kernel reduction was used to obtain a smaller rectangular kernel (2310×350 instead of 2310×2310.)**

Data Set $m \times n$ # of Classes	Nonlinear OFRQP Train Test Time (Sec.)	Nonlinear BR-MPSVM Train Test Time (Sec.) p-value
Wine 178×13 3	99.2 % 97.7 % 5.39	**100.0 %** **100.0 %** **0.45** $2.5e-2$
Glass 214×9 6	**88.5%** **70.0 %** 9.05	78.09% 69.1% **0.59** 0.84
Iris 150×4 3	98.1 % 98.0 % 3.01	**99.5%** **98.7%** **0.31** 0.62
Vowel 528×10 11	**100.0%** 94.3 % 221.34	**100.0%** **98.5%** **6.62** 0.67
Vehicle 846×18 4	**89.5%** 80.5 % 148.01	88.6% **82.2%** **1.17** 0.78
Segment 2310×19 7	**99.9 %** 96.1% 5562.31	98.3% **97.0%** 11.65 0.16

On the larger datasets (Vehicle, Segment) a rectangular kernel [20] was used on both methods in order to reduce even more the computational time while maintaining the correctness achieved by using the full kernel.

The nonlinear BP-MPSVM classifier was obtained in shorter time than the nonlinear OFRQP classifier in all the datasets tested. Furthermore, the BR-MPSVM algorithm was statistically better or equal to the nonlinear OFRQP on test set correctness. CPU times and p-values are given in Table 2.

In order to show graphically for the nonlinear case that BP-MPSVM can produce significant improvement over MPSVM, we created an artificial 2-dimensional example where this improvement can be visually observed. The example consists of 500 data points in 2 dimensions belonging to one of three classes. Class 1 consists of 400 points, class 2 consists of 50 points and class 3 consists of 50 points. Figure 5 depicts a nonlinear classification obtained using MSPVM without any modifications using a Gaussian kernel. Since the classes are unbalanced, we observe that the majority of the x class is misclassified by the algorithm leading to 91.8% training set correctness. On the other hand, Figure 6 depicts a nonlinear classification obtained by BP-MPSVM that utilizes balancing and Newton refinement which gives a significantly improved 98.8% training set correctness.

5. Concluding Remarks

We have proposed an extremely simple and fast procedure for generating linear and nonlinear multicategory classifiers. The one-from-the-rest approach is based on proximity of each class to one of two parallel planes that are pushed as far apart as possible. This procedure, a multicategory proximal support vector machine (MPSVM) with balancing and Newton refinement, requires nothing more sophisticated than solving k simple systems of linear equations, for either a linear or nonlinear classifier, where k is the number of classes. In contrast, standard one-from-the-rest support vector machine classifiers require the more costly solution of a linear or quadratic program. For a linear classifier, all that is needed by MPSVM is the solution of k nonsingular systems of linear equations of the order of the input space dimension, typically of 100 or less, even if there are millions of data points to classify. For a nonlinear classifier, a reduction method using rectangular kernels such as [20] is utilized and k linear systems of the order of as small as 15% of the data points are solved. Our computational results demonstrate that MPSVM classifiers obtain test set correctness comparable to that of standard one-from-the-rest SVM classifiers at a fraction of the time, often orders of magnitude less.

We have also proposed a novel Newton refinement algorithm that can improve classification accuracy for any two-class kernel classifier. This refinement is very fast, since it is a minimization problem in only two variables and is easy to implement. Future research plans include applying this refinement to other linear and nonlinear kernel-based classification algorithms. We have also addressed the problem of unbalanced datasets, which often occurs in one-from-rest classification approaches, by applying a very simple balanced version of PSVM together with a Newton refinement.

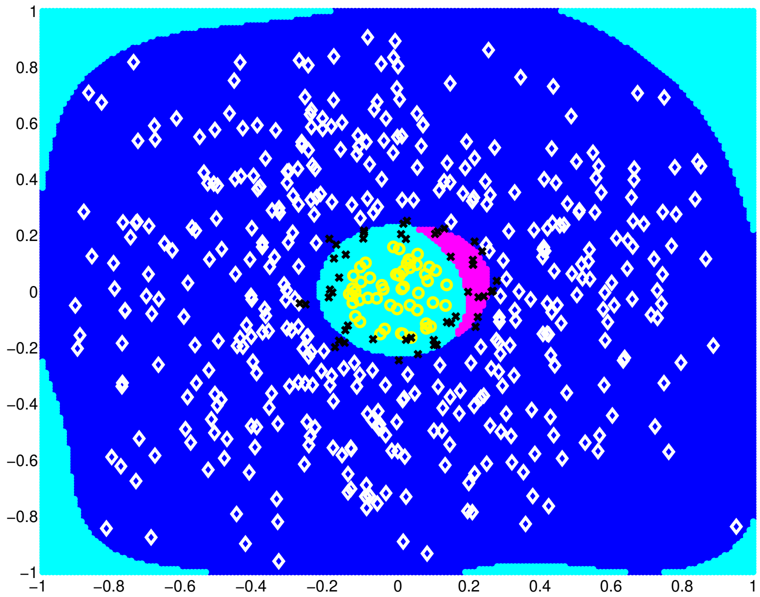

Figure 5. **Example consisting of 500 data points in 2 dimensions belonging to one of three classes. Nonlinear Gaussian kernel classifiers using MSPVM without balancing or Newton refinement generated a torus containing mostly white diamonds, a crescent containing black x's, and an ellipse cotaining mostly yellow circles. Since the classes are unbalanced, most of the x class is misclassified by the algorithm and resulting in a 91.8% overall training set correctness.**

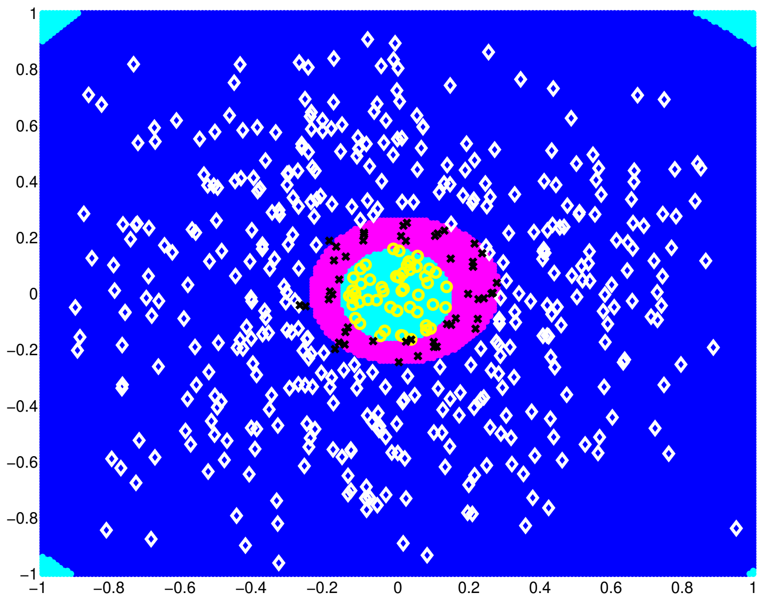

Figure 6. **The same example as that of Figure 5 classified by a nonlinear BR-MPSVM which uses MSPVM plus balancing and Newton refinement. This resulted in a torus containing mostly white diamonds, another torus containing black x's and an ellipse containing mostly yellow circles. Overall training set correctness is 98.8%.**

A promising avenue for future research is that of incremental classification for large scale multicategory datasets. This appears particularly promising in view of the very simple explicit solutions and for the linear and nonlinear MPSVM classifiers that can be updated incrementally as new data points come streaming in.

To sum up, the principal contribution of this work, is a very efficient classifier that requires no specialized software. MPSVM can be easily incorporated into all sorts of data mining and machine learning applications such as incremental and online learning that require a fast, simple and effective multicategory classifier.

Acknowledgments

The research described in this Data Mining Institute Report 01-06, July 2001, was supported by National Science Foundation Grants CCR-9729842, CCR-0138308 and CDA-9623632, by Air Force Office of Scientific Research Grant F49620-00-1-0085 and by the Microsoft Corporation.

References

1. E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK User's Guide*. SIAM, Philadelphia, Pennsylvania, third edition, 1999. http://www.netlib.org/lapack/.
2. K. P. Bennett and O. L. Mangasarian. Multicategory separation via linear programming. *Optimization Methods and Software*, 3:27–39, 1993.
3. L. Bottou, C. Cortes, J. Denker, H. Drucker, I. Guyon, L. Jackel, Y. LeCun, U. Muller, E. Sackinger, P. Simard, and V. Vapnik. Comparison of classifier methods: A case study in handwriting digit recognition. In *International Conference on Pattern Recognition*, pages 77–87. IEEE Computer Society Press, 1994.
4. P. S. Bradley and O. L. Mangasarian. Massive data discrimination via linear support vector machines. *Optimization Methods and Software*, 13:1–10, 2000. ftp://ftp.cs.wisc.edu/math-prog/tech-reports/98-03.ps.
5. E. J. Bredensteiner and K. P. Bennett. Multicategory classification by support vector machines. *Computational Optimization and Applications*, 12:53–79, 1999.
6. C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.
7. Chunhui Chen and O. L. Mangasarian. Hybrid misclassification minimization. *Advances in Computational Mathematics*, 5(2):127–136, 1996. ftp://ftp.cs.wisc.edu/math-prog/tech-reports/95-05.ps.
8. V. Cherkassky and F. Mulier. *Learning from Data - Concepts, Theory and Methods*. John Wiley & Sons, New York, 1998.
9. CPLEX Optimization Inc., Incline Village, Nevada. *Using the CPLEX(TM) Linear Optimizer and CPLEX(TM) Mixed Integer Optimizer (Version 2.0)*, 1992.
10. Sanjoy Dasgupta. Experiments with random projection. In *Uncertainty in Artificial Intelligence: Proceedings of the Sixteenth Conference (UAI-2000)*, pages 143–151, San Francisco, CA, 2000. Morgan Kaufmann Publishers.
11. T. Evgeniou, M. Pontil, and T. Poggio. Regularization networks and support vector machines. In A. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 171–203, Cambridge, MA, 2000. MIT Press.
12. F. Facchinei. Minimization of SC^1 functions and the Maratos effect. *Operations Research Letters*, 17:131–137, 1995.
13. G. Fung and O. L. Mangasarian. Proximal support vector machine classifiers. In F. Provost and R. Srikant, editors, *Proceedings KDD-2001: Knowledge Discovery and Data Mining,*

August 26-29, 2001, San Francisco, CA, pages 77–86, New York, 2001. Asscociation for Computing Machinery. ftp://ftp.cs.wisc.edu/pub/dmi/tech-reports/01-02.ps.

14. T. S. Furey, N. Duffy, N. Cristianini, D. Bednarski, M. Schummer, and D. Haussler. Support vector machine classification and validation of cnacer tissue samples using microarray expression data. *Bioinformatics*, 16(10):906–914, 2000.
15. T. Van Gestel, J. Suykens, G. Lanckriet, A. Lambrechts, B. De Moor, and J. Vandewalle. Multiclass ls-svms: moderated outputs and coding-decoding schemes. *Neural Processing Letters*, 15(1):45–48, 2002.
16. J.-B. Hiriart-Urruty, J. J. Strodiot, and V. H. Nguyen. Generalized hessian matrix and second-order optimality conditions for problems with C^{L1} data. *Applied Mathematics and Optimization*, 11:43–56, 1984.
17. A. E. Hoerl and R. W. Kennard. Biased estimation for nonorthogonal problems. *Technometrics*, 12:55–67, 1952.
18. C.-W. Hsu and C.-J. Lin. A comparison on methods for Multi-Class support vector machines, 2001. http://www.csie.ntu.edu.tw/ cjlin/papers.html.
19. C. Kanzow, H. Qi, and L. Qi. On the minimum norm solution of linear programs. Preprint, University of Hamburg, Hamburg, 2001. http://www.math.uni-hamburg.de/home/kanzow/paper.html. Journal of Optimization Theory and Applications, to appear.
20. Y.-J. Lee and O. L. Mangasarian. RSVM: Reduced support vector machines. In *Proceedings of the First SIAM International Conference on Data Mining, Chicago, April 5-7, 2001, CD-ROM*. ftp://ftp.cs.wisc.edu/pub/dmi/tech-reports/00-07.ps.
21. Y.-J. Lee and O. L. Mangasarian. SSVM: A smooth support vector machine. *Computational Optimization and Applications*, 20:5–22, 2001. Data Mining Institute, University of Wisconsin, Technical Report 99-03. ftp://ftp.cs.wisc.edu/pub/dmi/tech-reports/99-03.ps.
22. O. L. Mangasarian. *Nonlinear Programming*. SIAM, Philadelphia, PA, 1994.
23. O. L. Mangasarian. Generalized support vector machines. In A. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 135–146, Cambridge, MA, 2000. MIT Press. ftp://ftp.cs.wisc.edu/math-prog/tech-reports/98-14.ps.
24. MATLAB. *User's Guide*. The MathWorks, Inc., Natick, MA 01760, 1994-2001. http://www.mathworks.com.
25. P. M. Murphy and D. W. Aha. UCI machine learning repository, 1992. www.ics.uci.edu/∼mlearn/MLRepository.html.
26. B. T. Polyak. *Introduction to Optimization*. Optimization Software, Inc., Publications Division, New York, 1987.
27. R. T. Rockafellar. *Convex Analysis*. Princeton University Press, Princeton, New Jersey, 1970.
28. B. Schölkopf, S. Mika, C. J. C. Burges, P. Knirsch, K.-R. Müller, G. Rätsch, and A. J. Smola. Input space versus feature space in kernel-based methods. *IEEE Transactions on Neural Networks*, 10:1000–1017, 1999.
29. A. J. Smola and B. Schölkopf. Sparse greedy matrix approximation for machine learning. In *Proc. 17th International Conf. on Machine Learning*, pages 911–918. Morgan Kaufmann, San Francisco, CA, 2000.
30. J. A. K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle. *Least Squares Support Vector Machines*. World Scientific Publishing Co., Singapore, 2002.
31. J. A. K. Suykens, L. Lukas, P. Van Dooren, B. De Moor, and J. Vandewalle. Least squares support vector machine classifiers: a large scale algorithm. In *European Conference on Circuit Theory and Design, ECCTD'99*, pages 839–842, Stresa, Italy, 1999.
32. J. A. K. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural Processing Letters*, 9(3):293–300, 1999.
33. J. A. K. Suykens and J. Vandewalle. Multiclass least squares support vector machines. In *Proceedings of IJCNN'99*, pages CD–ROM, Washington, DC, 1999.
34. A. N. Tikhonov and V. Y. Arsenin. *Solutions of Ill–Posed Problems*. John Wiley & Sons, New York, 1977.

35. V. Roth V. and V. Steinhage. Nonlinear discriminant analysis using kernel function. In S.A. Solla, T.K. Leen, and K.-R. Mueller, editors, *Advances in Neural Information Processing Systems–NIPS*99*, pages 568–574, 1999.
36. V. N. Vapnik. *The Nature of Statistical Learning Theory.* Springer, New York, second edition, 2000.
37. J. Weston and C. Watkins. Multi-class support vector machines. Technical report csd-tr-98-04, Royal Holloway, University of London, Surrey, England, 1998.
38. C. K. I. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. *Advances in Neural Information Processing Systems (NIPS2000)*, 2000. http://www.kernel-machines.org.