Chapter 1

OPTIMIZATION METHODS IN MASSIVE DATASETS

P. S. Bradley
Microsoft Research
One Microsoft Way
Redmond, WA 98052
bradley@microsoft.com

O. L. Mangasarian
Computer Sciences Dept.
University of Wisconsin
1210 West Dayton St.
Madison, WI 53706
olvi@cs.wisc.edu

D. R. Musicant
Computer Sciences Dept.
University of Wisconsin
1210 West Dayton St.
Madison, WI 53706
musicant@cs.wisc.edu

Abstract We describe the role of generalized support vector machines in separating massive and complex data using arbitrary nonlinear kernels. Feature selection that improves generalization is implemented via an effective procedure that utilizes a polyhedral norm or a concave function minimization. Massive data is separated using a linear programming chunking algorithm as well as a successive overrelaxation algorithm, each of which is capable of processing data with millions of points.

1. INTRODUCTION

We address here the problem of classifying data in n-dimensional real (Euclidean) space R^n into one of two disjoint finite point sets (i.e. classes).

The support vector machine (SVM) approach to classification [57, 2, 25, 58, 13, 54, 55] attempts to separate points belonging to two given sets in R^n by a nonlinear surface, often only implicitly defined by a kernel function. Since the nonlinear surface in R^n is typically linear in its parameters, it can be represented as a linear function (plane) in a higher, often much higher, dimensional space, say R^k. Also, the original points of the two given sets can also be mapped into this higher dimensional space. If the two sets are linearly separable in R^k, then it is intuitively plausible to generate a plane mid-way between the furthest parallel planes apart that bound the two sets. Using a distance induced by the kernel generating the nonlinear surface in R^n, it can be shown [57] that such a plane optimizes the generalization ability of the separating plane. If the two sets are not linearly separable, a similar approach can be used [15, 57] to maximize the distance between planes that bound each set with certain minimal error.

The feature selection problem addressed here is that of discriminating between the two point sets in R^n by a separating plane utilizing as few of the n original problem features as possible. We focus on two approaches. The first approach is motivated by formulating the feature selection problem as the minimization of a concave function over a polyhedral region. A stationary point to this problem is efficiently computed by solving a sequence of linear programs in a successive linearization algorithm. The second approach results from an investigation of the SVM problem formulated as maximizing the margin of separation measured in the 1-norm and ∞-norm.

We also consider a linear programming approach [6, 1] to SVMs [57, 58, 15] for the discrimination between two possibly massive datasets. A proposed approach consists of a novel method for solving linear programs with an extremely large number of constraints that is proven to be monotonic and finite. In the standard SVM formulation [50, 49, 52] very large quadratic programs are solved. In contrast, the formulation here consists of solving a linear program which is considerably less difficult. For simplicity, our results are given here for a linear discriminating surface, i.e. a separating plane. However, extension to nonlinear surfaces such as quadratic [32] or more complex surfaces [12] is straightforward.

We next consider the successive overrelaxation (SOR) method for solving massive quadratic programming SVMs. A conventional SVM in its dual formulation contains bound constraints, as well as an equality constraint that requires special treatment in iterative procedures. A very simple convex quadratic program with bound constraints *only* can be obtained by taking the dual of the quadratic program associated with an SVM that maximizes the margin (distance between bounding separating planes) with respect to *all* the parameters determining the bounding planes [41]. This quadratic program can be solved for massive datasets by successive overrelaxation to obtain a linear or nonlinear separating surface [42].

1.1 NOTATION AND BACKGROUND

- All vectors will be column vectors unless transposed to a row vector by a prime superscript $'$. The scalar (inner) product of two vectors x and y in R^n will be denoted as $x'y$.

- For a vector x in the n-dimensional real space R^n, $|x|$ will denote a vector of absolute values of the components $x_i,\ i = 1, \ldots, n$ of x.

- For a vector x in R^n, x_+ denotes the vector in R^n with components $\max\{0, x_i\}$.

- For a vector x in R^n, x_* denotes the vector in R^n with components $(x_*)_i$ equal 1 if $x_i > 0$ and 0 otherwise (i.e. x_* is the result of applying the step function to the components of x).

- The base of the natural logarithm will be denoted by ε, and for a vector $y \in R^n$, ε^{-y} will denote a vector in R^n with components $\varepsilon^{-y_i},\ i = 1, \ldots, n$.

- For $x \in R^n$ and $1 \leq p < \infty$, the norm $\|x\|_p$ will denote the p-norm:

$$\|x\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{\frac{1}{p}},$$

  and

$$\|x\|_\infty = \max_{1 \leq i \leq n} |x_i|.$$

- For a general norm $\|\cdot\|$ on R^n, the *dual norm* $\|\cdot\|'$ on R^n is defined as

$$\|x\|' = \max_{\|y\|=1} x'y.$$

The 1-norm and ∞-norm are dual norms, and so are a p-norm and a q-norm for which $1 \leq p,\ q \leq \infty$ and $\frac{1}{p} + \frac{1}{q} = 1$.

- The notation $A \in R^{m \times n}$ will signify a real $m \times n$ matrix. For such a matrix A' will denote the transpose of A, A_i will denote the i-th row of A, and $A_{\cdot j}$ will denote the jth column of A.

- A vector of ones in a real space of arbitrary dimension will be denoted by e. A vector of zeros in a real space of arbitrary dimension will be denoted by 0. The identity matrix in a real space of arbitrary dimension will be denoted by I.

- The notation $\arg \min\limits_{x \in S} f(x)$ will denote the set of minimizers of $f(x)$ on the set S.

- We shall employ the MATLAB "dot" notation [44] to signify application of a function to all components of a matrix or a vector. For example if $A \in R^{m \times n}$, then $A^2_{\bullet} \in R^{m \times n}$ will denote the matrix of elements of A squared.

2. GENERALIZED SUPPORT VECTOR MACHINES FOR MASSIVE DATA DISCRIMINATION

We start with a nonlinear separating surface (1.1), implicitly defined by some chosen kernel and by some linear parameters $u \in R^m$ to be determined. These parameters turn out to be closely related to some dual variables. Based on this surface we derive a general convex mathematical program (1.5) that attempts separation via the nonlinear surface (1.1) while minimizing some function f of the parameters u. The function f which attempts to suppress u can be interpreted as minimizing the number of support vectors, or under more conventional assumptions as maximizing the distance between the separating planes in R^k. The choice of f leads to various SVMs. We consider two classes of such machines based on whether f is quadratic or piecewise linear. If we choose f to be a quadratic function generated by the kernel defining the nonlinear surface (1.1), then we are led to the conventional dual quadratic program (1.9) associated with an SVM which requires positive definiteness of this kernel. However the quadratic function choice for f can be divorced from the kernel defining the separating surface and this leads to other convex quadratic programs such as (1.10) *without* making any assumptions on the kernel. Another class of SVMs is generated by choosing a piecewise linear convex function for f and this leads to linear programs such as (1.11) and (1.12), both of which make no assumptions on the kernel.

We begin by defining a general kernel function as follows.

Definition 1 *Let $A \in R^{m\times n}$ and $B \in R^{n\times \ell}$. The* **kernel** $K(A, B)$ *maps $R^{m\times n} \times R^{n\times \ell}$ into $R^{m\times \ell}$.*

In particular if x and y are column vectors in R^n, then $K(x', A')$ is a row vector in R^m, $K(x', y)$ is a real number and $K(A, A')$ is an $m \times m$ matrix. Note that for our purposes here $K(A, A')$ need not be symmetric in general. Examples of kernels follow, where $a \in R^m$, $b \in R^\ell$, $\mu \in R$ and d is an integer.

Example: Polynomial Kernel (degree d) $(AB + \mu ab')^d_\bullet$

Example: Neural Network Kernel $(AB + \mu ab')_{\bullet *}$

Example: Radial Basis Kernel $\varepsilon^{-\mu\|A_i'-B_{\cdot j}\|^2}$, $i, j = 1, \ldots, m$, $\ell = m$.

Note that our approach allows discontinuous kernels such as the neural network kernel with a discontinuous step function without the need for a smoothing approximation such as the sigmoid or hyperbolic tangent approximation as is usually done [57, 13].

2.1 GSVM: THE GENERAL SUPPORT VECTOR MACHINE

We consider a given set $\mathcal{A}$ of m points in the n-dimensional real feature space R^n represented by the matrix $A \in R^{m\times n}$. Each point A_i, $i = 1, \ldots, m$, belongs to class 1 or class -1 depending on whether D_{ii} is 1 or -1, where $D \in R^{m\times m}$ is a given diagonal matrix of plus or minus ones. We shall attempt to discriminate between the classes 1 and -1 by a nonlinear *separating surface*, induced by some kernel $K(A, A')$, as follows:

$$K(x', A')Du = \gamma, \tag{1.1}$$

where $K(x', A') \in R^m$, according to Definition 1. The parameters $u \in R^m$ and $\gamma \in R$ are determined by solving a mathematical program, typically quadratic or linear. A point $x \in R^n$ is classified in class 1 or -1 according to whether the *decision function*

$$(K(x', A')Du - \gamma)_*, \tag{1.2}$$

yields 1 or 0 respectively. The kernel function $K(x', A')$ implicitly defines a nonlinear map from $x \in R^n$ to some other space $z \in R^k$ where k

may be much larger than n. In particular if the kernel K is an inner product kernel under Mercer's condition [16, pp 138-140],[57, 13, 12] (an assumption that we will not make) then for x and y in R^n:

$$K(x, y) = h(x)'h(y), \tag{1.3}$$

and the separating surface (1.1) becomes:

$$h(x)'h(A')Du = \gamma, \tag{1.4}$$

where h is a function, not easily computable, from R^n to R^k, and $h(A') \in R^{k\times m}$ results from applying h to the m columns of A'. The difficulty in computing h and the possible high dimensionality of R^k have been important factors in using a kernel K as a generator of an implicit nonlinear separating surface in the original feature space R^n but which is linear in the high dimensional space R^k. Our separating surface (1.1) written in terms of a kernel function retains this advantage and is linear in its parameters, u, γ. We now state a mathematical program that generates such a surface for a general kernel K as follows:

$$\begin{array}{rrcl} \displaystyle\min_{u,\gamma,y} & \nu e'y + f(u) & & \\ \text{s.t.} & D(K(A, A')Du - e\gamma) + y & \geq & e \\ & y & \geq & 0. \end{array} \tag{1.5}$$

Here f is some convex function on R^m, typically some norm or seminorm, and ν is some positive parameter that weights the separation error $e'y$ versus suppression of the separating surface parameter u. Suppression of u can be interpreted in one of two ways. We interpret it here as minimizing the number of support vectors, i.e. constraints of (1.5) with positive Lagrange multipliers. A more conventional interpretation is that of maximizing some measure of the distance or margin between the bounding parallel planes in R^k, under appropriate assumptions, such as f being a quadratic function induced by a positive definite kernel K as in (1.9) below. As is well known, this leads to improved generalization by minimizing an upper bound on the VC dimension [57, 54].

We term a solution of the mathematical program (1.5) and the resulting decision function (1.2) a *generalized support vector machine* (GSVM). In the following section we derive a number of special cases, including the standard SVM. First, however, we note that the mathematical program (1.5) has a solution whenever f is a piecewise-linear or quadratic function bounded below on R^m [38, Proposition 2.1]. Note that no convexity of f is required for this existence result. However in applications where duality theory will be invoked, f will need to be convex.

2.2 QUADRATIC PROGRAMMING SUPPORT VECTOR MACHINES

We consider specific formulations of the SVM problem, including the standard ones [57, 13, 12] and those obtained by setting f of (1.5) to be a convex quadratic function $f(u) = \frac{1}{2}u'Hu$, where $H \in R^{m\times m}$ is some symmetric positive definite matrix. The mathematical program (1.5) becomes the following convex quadratic program:

$$\begin{array}{rl} \min\limits_{u,\gamma,y} & \nu e'y + \frac{1}{2}u'Hu \\ \text{s.t.} & D(K(A,A')Du - e\gamma) + y \geq e \\ & y \geq 0. \end{array} \tag{1.6}$$

The Wolfe dual [59, 34] of this convex quadratic program is:

$$\begin{array}{rl} \min\limits_{r\in R^m} & \frac{1}{2}r'DK(A,A')DH^{-1}DK(A,A')'Dr - e'r \\ \text{s.t.} & e'Dr = 0 \\ & 0 \leq r \leq \nu e. \end{array} \tag{1.7}$$

Furthermore, the primal variable u is related to the dual variable r by:

$$u = H^{-1}DK(A,A')'Dr. \tag{1.8}$$

If we assume that the kernel $K(A,A')$ is symmetric positive definite and let $H = DK(A,A')D$, then our dual problem (1.7) degenerates to the dual problem of the standard SVM [57, 13, 12] with $u = r$:

$$\begin{array}{rl} \min\limits_{u\in R^m} & \frac{1}{2}u'DK(A,A')Du - e'u \\ \text{s.t.} & e'Du = 0 \\ & 0 \leq u \leq \nu e. \end{array} \tag{1.9}$$

The positive definiteness assumption on $K(A,A')$ in (1.9) can be relaxed to positive *semi*definiteness while maintaining the convex quadratic program (1.6), with $H = DK(A,A')D$, as the direct dual of (1.9) without utilizing (1.7) and (1.8). The symmetry and positive semidefiniteness of the kernel $K(A,A')$ for this version of an SVM is consistent with the SVM literature. The fact that $r = u$ in the dual formulation (1.9), shows that the variable u appearing in the original formulation (1.6) is also the dual multiplier vector for the first set of constraints of (1.6). Hence the quadratic term in the objective function of (1.6) can be thought of as suppressing as many multipliers of support vectors as possible. This is another interpretation of the standard SVM that is usually interpreted as maximizing the margin or distance between parallel separating planes.

This leads to the idea of using other values for the matrix H other than $DK(A, A')D$ that will also suppress u. One particular choice is interesting because it puts no restrictions on K: no symmetry, no positive definiteness or semidefiniteness and not even continuity. This is the choice $H = I$ in (1.6) which leads to a dual problem (1.7) with $H = I$ and $u = DK(A, A')'Dr$ as follows:

$$\begin{array}{rrcl} \min\limits_{r \in R^m} & \multicolumn{3}{l}{\frac{1}{2}r'DK(A, A')K(A, A')'Dr - e'r} \\ \text{s.t.} & e'Dr & = & 0 \\ 0 \leq & r & \leq & \nu e. \end{array} \tag{1.10}$$

We note immediately that $K(A, A')K(A, A')'$ is positive semidefinite with no assumptions on $K(A, A')$, and hence the above problem is an always solvable convex quadratic program for any kernel $K(A, A')$. In fact by [38, Proposition 2.1] the quadratic program (1.6) is solvable for *any* symmetric positive definite matrix H, and by quadratic programming duality so is its dual problem (1.7), the solution r of which can be immediately used to generate a decision function (1.2). Thus we are free to choose any symmetric positive definite matrix H to generate an SVM. Experimentation determines the most appropriate choices for H.

We turn our attention to linear programming SVMs.

2.3 LINEAR PROGRAMMING SUPPORT VECTOR MACHINES

In this section we consider problems generated from the mathematical program (1.5) by using a piecewise linear function f in the objective function thus leading to linear programs.

An obvious choice for f is the 1-norm of u, which leads to the following linear programming formulation:

$$\begin{array}{rrcl} & \min\limits_{u,\gamma,y,s} & \multicolumn{2}{l}{\nu e'y + e's} \\ \text{s.t.} & D(K(A, A')Du - e\gamma) + y & \geq & e \\ & -s \leq u & \leq & s \\ & y & \geq & 0. \end{array} \tag{1.11}$$

A solution (u, γ, y, s) to this linear program for a chosen kernel $K(A, A')$ will provide a decision function as given by (1.2). This linear program parallels the quadratic programming formulation (1.10) that was obtained as the dual of (1.5) by setting $f(u)$ therein to half the 2-norm squared of u whereas $f(u)$ is set to the 1-norm of u in (1.11). Another linear programming formulation that somewhat parallels the quadratic programming formulation (1.9), which was obtained as the dual of (1.5) by setting $f(u)$ therein to half the 2-norm squared of $K(A,A')^{\frac{1}{2}}Du$, is obtained by setting f to be the 1-norm of $K(A,A')Du$. This leads to the following linear program:

$$\begin{array}{rl} \min\limits_{u,\gamma,y,s} & \nu e'y + e's \\ \text{s.t.} & D(K(A,A')Du - e\gamma) + y \;\geq\; e \\ & -s \;\leq\; K(A,A')Du \;\leq\; s \\ & y \;\geq\; 0. \end{array} \tag{1.12}$$

No assumptions of symmetry or positive definiteness on $K(A,A')$ are needed in either of the above linear programming formulations as was the case in the quadratic program (1.9).

It is interesting to note that if the linear kernel $K(A,A') = AA'$ is used in the linear program (1.11) we obtain the high-performing 1-norm linear SVM proposed in [11] and utilized successfully in [10, 3, 6]. Hence, if we set $w = A'Du$ in (1.11) we obtain (1.26) and (1.28).

This linear programming SVM was implemented in [42] with nonlinear kernels. A quadratic kernel resulted in better testing set results than a linear kernel did on the liver-disorders dataset from the UCI repository [45]. Figure 1.1 depicts a rather complex "checkerboard" example for which no linear classifier can give good separation. Figure 1.2 shows the sharp nonlinear separation obtained by a nonlinear polynomial kernel of degree 6 ($d = 6$):

Polynomial Kernel: $K(A,A') = ((\frac{A}{\lambda} - \rho)(\frac{A}{\lambda} - \rho)' - \mu)^d_\bullet$

The parameter values for λ, ρ, and μ are shown in the caption for Figure 1.2.

We next focus on the feature selection problem in classification.

3. FEATURE SELECTION VIA CONCAVE MINIMIZATION AND SUPPORT VECTOR MACHINES

We consider in this section the problem of discriminating between two finite point sets in R^n by a separating plane that utilizes as few of the n features as possible. Two approaches are described.

The first approach [37, 9], described in Section 3.1, involves the minimization of a concave function on a polyhedral set and is based on the following considerations. A plane is constructed such that a weighted sum of distances of misclassified points to the plane is minimized, utilizing as few dimensions of the original feature space R^n as possible.

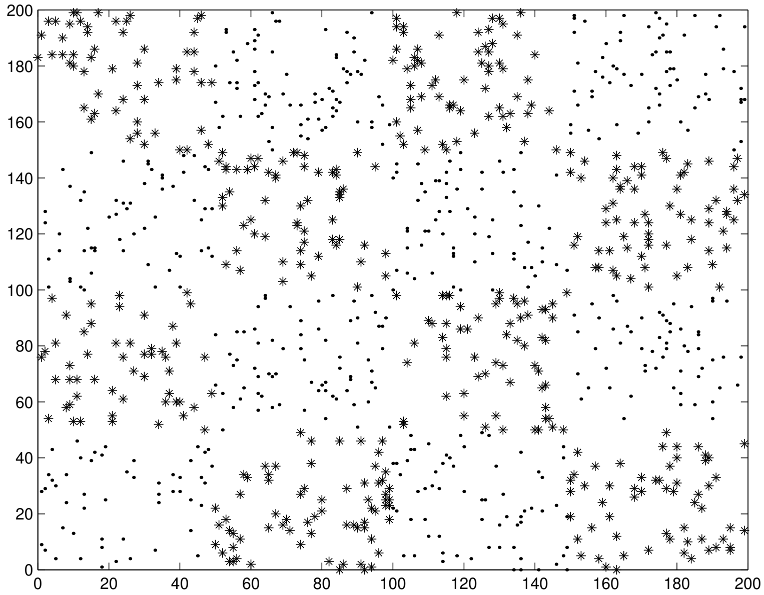

Figure 1.1 **Checkerboard training dataset**

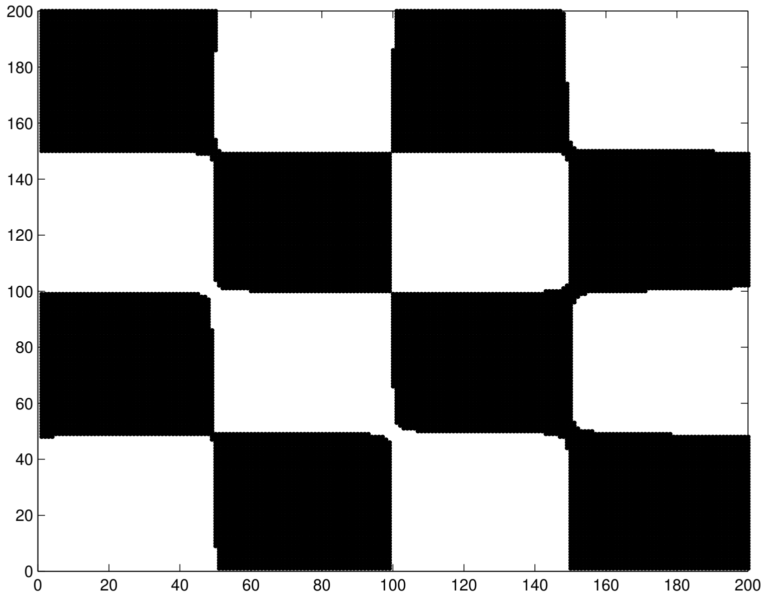

Figure 1.2 **Indefinite sixth degree polynomial kernel separation of the checkerboard dataset ($\nu = 10,000,\ \lambda = 100,\ \rho = 1,\ d = 6,\ \mu = 0.5$)**

This is achieved by constructing two parallel bounding planes, utilizing a minimal number of dimensions. The two planes determine two opposite halfspaces. Each halfspace mostly contains points of one set. These planes are determined so as to minimize the sum of weighted distances to the bounding plane from points lying in the wrong halfspace. This leads to the minimization of a concave function on a polyhedral set (problems (1.19) and (1.21) below) for which a stationary point can be obtained by solving a few (typically 5 to 7) linear programs in a successive linearization algorithm (Algorithm 2 below). The final separating plane is taken midway between the two bounding parallel planes.

The second approach, that of an SVM [57, 2, 25, 58] (described in Section 3.2), constructs the two parallel bounding planes in the full n-dimensional space R^n as in the first approach outlined above, but in addition attempts to push these planes as far apart as possible. The justification for this is to improve the VC dimension [57], which in turn improves generalization. The typical formulation of the SVM problem uses the 2-norm to measure the distance between the two bounding planes and leads to a quadratic programming problem [57]. We use here instead the 1-norm and the ∞-norm that lead to the single linear programs (1.25) and (1.26) below, respectively. Although improved generalization is the primary purpose of the SVM formulation, it turns out that the linear program (1.26) resulting from employing the ∞-norm to measure the distance between the two bounding planes leads also to a feature selection method, whereas the linear program (1.25) resulting from the use of the 1-norm does not lead to a feature selection method. Note that the norms on w used in (1.25) and (1.26) are dual to those used to measure the distance between the separating planes [39].

3.1 FSV: FEATURE SELECTION VIA CONCAVE MINIMIZATION

We consider a feature selection procedure that has been effective in medical and other applications [9, 37].

Given m points in R^n represented by the $m \times n$ matrix A and the $m \times m$ diagonal matrix D with $D_{ii} = 1$ if the i-th data point A_i belongs to class 1 and $D_{ii} = -1$ if the i-th data point belongs to class -1, we wish to discriminate between them by a separating plane:

$$P = \{x \mid x \in R^n, x'w = \gamma\}, \tag{1.13}$$

with normal $w \in R^n$ and 1-norm distance to the origin of $\dfrac{|\gamma|}{\|w\|_\infty}$ [39]. We shall attempt to determine w and γ so that the separating plane P defines two open halfspaces $\{x \mid x \in R^n, x'w > \gamma\}$ containing mostly points

A_i from class 1, and $\{x \mid x \in R^n, x'w < \gamma\}$ containing mostly points A_i from class -1. Hence we wish to satisfy

$$\begin{array}{ll} A_i'w > \gamma & \text{if } D_{ii} = 1, \\ A_i'w < \gamma & \text{if } D_{ii} = -1, \end{array} \tag{1.14}$$

to the extent possible. Upon normalization, these inequalities can be equivalently written as:

$$D(Aw - e\gamma) \geq e. \tag{1.15}$$

Conditions (1.14) or equivalently (1.15) can be satisfied if and only if the convex hulls of the data points belonging to class 1 and the data points belonging to class -1 are disjoint. This is not the case in many real-world applications. Hence, we attempt to satisfy (1.15) in some "best" sense, for example, by minimizing some norm of the violations of (1.15) such as

$$\min_{w,\gamma} f(w,\gamma) = \min_{w,\gamma} \| \left(D(Aw - e\gamma) - e\right)_+ \|_1. \tag{1.16}$$

Recall that for a vector x, x_+ denotes the vector with components $\max\{0, x_i\}$. Two principal reasons for choosing the 1-norm in (1.16) are:

(i) Problem (1.16) is then reducible to a linear program (1.17) with many important theoretical properties making it an effective computational tool [4].

(ii) The 1-norm is less sensitive to outliers such as those occurring when the underlying data distributions have pronounced tails, hence (1.16) has a similar effect to that of robust regression [27],[26, pp 82-87].

The formulation (1.16) is equivalent to the following robust linear programming formulation (RLP) proposed in [4] and effectively used to solve problems from real-world domains [43]:

$$\begin{array}{rl} \displaystyle\min_{w,\gamma,y} & e'y \\ \text{s.t.} & D(Aw - e\gamma) + y \geq e \\ & y \geq 0. \end{array} \tag{1.17}$$

The linear program (1.17) or, equivalently, the formulation (1.16) define a separating plane P that approximately satisfies the conditions (1.15) in the following sense. For each positive value of y_i, the corresponding data point A_i is in error. Thus, if $D_{ii} = 1$, then the data point A_i lies on the wrong side of the bounding plane $x'w = \gamma + 1$ for class 1; if $D_{ii} = -1$, the data point A_i lies on the wrong side of the bounding plane $x'w = \gamma - 1$ for class -1. Hence the objective function of the linear program (1.17) minimizes the sum of distances, weighted by $\|w\|'$, of misclassified points to their respective bounding planes. The separating plane P (1.13) is midway between the two bounding planes and parallel to them.

We now introduce a feature selection idea [37, 9] by attempting to suppress as many components as possible of the normal vector w to the separating plane P that is consistent with obtaining an acceptable separation between the points with class 1 and class -1. We achieve this by introducing an extra term into the objective of (1.17) while weighting the original objective by $\nu > 0$ as follows:

$$\begin{array}{rl} \displaystyle\min_{w,\gamma,y} & \nu e'y + e'|w|_* \\ \text{s.t.} & D(Aw - e\gamma) + y \geq e \\ & y \geq 0. \end{array} \tag{1.18}$$

Note that the vector $|w|_* \in R^n$ has components which are equal to 1 if the corresponding components of w are nonzero and components equal to zero if the corresponding components of w are zero. Recall that e is a vector of ones and $e'|w|_*$ is simply a count of the nonzero elements in the vector w. Problem (1.18) balances the error in separating the points from class 1 and -1 (the term $e'y$), and the number of nonzero elements of w (the term, $e'|w|_*$). Further, note that if an element of w is zero, the corresponding feature is removed from the problem.

By introducing the variable v we are able to eliminate the absolute value from problem (1.18) which leads to the following equivalent parametric program for $\nu > 0$:

$$\begin{array}{rl} \displaystyle\min_{w,\gamma,y,v} & \nu e'y + e'v_* \\ \text{s.t.} & D(Aw - e\gamma) + y \geq e \\ & -v \leq w \leq v \\ & y \geq 0. \end{array} \tag{1.19}$$

Since v appears positively weighted in the objective and is constrained by $-v \leq w \leq v$, it effectively models the vector $|w|$. This feature selection problem will be solved for a value of $\nu > 0$ for which the resulting classification obtained by the midway separating plane (1.13) generalizes best, estimated by a cross-validation tuning procedure. Typically this will be achieved in a feature space of reduced dimensionality, that is $e'v_* < n$ (i.e. the number of features used is less than n).

Because of the discontinuity of the step function term $e'v_*$, we approximate it by a concave exponential on the nonnegative real line [37]. The approximation of the step vector v_* of (1.19) by the concave exponential:

$$v_* \approx t(v, \alpha) = e - \varepsilon^{-\alpha v}, \alpha > 0, \tag{1.20}$$

leads to the smooth problem (**FSV**:Feature Selection Concave), for $\nu > 0$:

$$\begin{array}{rl} \min\limits_{w,\gamma,y,v} & \nu e'y + e'\left(e - \varepsilon^{\alpha v}\right) \\ \text{s.t.} & D(Aw - e\gamma) + y \geq e \\ & -v \leq w \leq v \\ & y \geq 0. \end{array} \tag{1.21}$$

It can be shown [8, Theorem 2.1] that for a finite value of α (appearing in the concave exponential) the smooth problem (1.21) generates an exact solution of the nonsmooth problem (1.19). We note that this problem is the minimization of a concave objective function over a polyhedral set. Even though it is difficult to find a global solution to this problem, a fast successive linear approximation (SLA) algorithm [9, Algorithm 2.1] terminates finitely (usually in 5 to 7 steps) at a stationary point which satisfies the minimum principle necessary optimality condition for problem (1.21) [9, Theorem 2.2]. This solution also leads to a sparse w with good generalization properties. We state the SLA algorithm below.

Algorithm 2 Successive Linearization Algorithm (SLA) for FSV (1.21). *Choose $\nu > 0$. Start with a random $(w^0, \gamma^0, y^0, v^0)$. Having $(w^i, \gamma^i, y^i, v^i)$ determine the next iterate by solving the linear program:*

$$(w^{i+1}, \gamma^{i+1}, y^{i+1}, v^{i+1}) \in$$

$$\begin{array}{rl} \arg \mathit{vertex} \min\limits_{w,\gamma,y,v} & \nu e'y + \alpha(\varepsilon^{-\alpha v^i})'(v - v^i) \\ s.t. & D(Aw - e\gamma) + y \geq e \\ & -v \leq w \leq v \\ & y \geq 0. \end{array} \tag{1.22}$$

Stop when

$$\nu\left(e'(y^{i+1} - y^i)\right) + \alpha(\varepsilon^{-\alpha v^i})'(v^{i+1} - v^i) = 0. \tag{1.23}$$

Comment: It was empirically determined that a value of $\alpha = 5$ produced satisfactory solutions. The parameter ν is usually chosen so as to maximize the generalization ability of the resulting classifier (as measured by cross-validation [56], for instance).

3.2 FEATURE SELECTION VIA SUPPORT VECTOR MACHINES

The SVM idea [57, 2, 25, 58], although not originally intended as a feature selection tool, does in fact indirectly suppress components of the normal vector w to the separating plane P (1.13) when an appropriate norm is used for measuring the distance between the two parallel bounding planes for the sets being separated. The SVM approach consists of adding another term, $\|w\|'$, to the objective function of the RLP (1.17) in a similar manner to the appended term $e'|w|_*$ of problem (1.18). Here, $\|\cdot\|'$ is the dual of some norm on R^n used to measure the distance between the two bounding planes. The justification for this term is as follows. The separating plane P (1.13) generated by the RLP linear program (1.17) lies midway between the two parallel planes $w'x = \gamma + 1$ and $w'x = \gamma - 1$. The distance, measured by some norm $\|\cdot\|$ on R^n, between these planes is precisely $\frac{2}{\|w\|'}$ [39, Theorem 2.1]. The appended term to the objective function of the RLP (1.17), $\|w\|'$, is twice the reciprocal of this distance, which has the effect of driving the distance between these two planes up to obtain better separation. This results then in the following mathematical programming formulation for the SVM problem:

$$\begin{array}{rl} \min\limits_{w,\gamma,y} & \nu e'y + \|w\|' \\ \text{s.t.} & D(Aw - e\gamma) + y \geq e \\ & y \geq 0. \end{array} \tag{1.24}$$

Points A_i appearing in active constraints of the linear program (1.24) with positive dual variables constitute the *support vectors* of the problem. These points are the *only* data points that are relevant for determining the optimal separating plane. Their number is usually small and it is proportional to the generalization error of the classifier [50].

If we use the 1-norm to measure the distance between the planes, then the dual to this norm is the ∞-norm and accordingly $\|w\|' = \|w\|_\infty$ in (1.24) which leads to the following linear programming formulation:

$$\begin{array}{rl} \min\limits_{w,\gamma,y,\sigma} & \nu e'y + \sigma \\ \text{s.t.} & D(Aw - e\gamma) + y \geq e \\ & -e\sigma \leq w \leq e\sigma \\ & y \geq 0. \end{array} \tag{1.25}$$

Similarly if we use the ∞-norm to measure the distance between the planes, then the dual to this norm is the 1-norm and accordingly $\|w\|' = \|w\|_1$ in (1.24) which leads to the following linear programming formulation:

$$\begin{array}{rl} \min\limits_{w,\gamma,y,s} & \nu e'y + e's \\ \text{s.t.} & D(Aw - e\gamma) + y \geq e \\ & -s \leq w \leq s \\ & y \geq 0. \end{array} \tag{1.26}$$

We note that the first paper on the multisurface method on pattern separation [33] also proposed and implemented the idea of separating the bounding planes, just as the SVM approach does.

Usually the SVM problem is formulated using the 2-norm in the objective [57, 2]. Since the 2-norm is dual to itself, it follows that the distance between the parallel planes defining the separating surface is also measured in the 2-norm when this formulation is used.

An experimental comparison between the linear classifiers obtained by solving the FSV problem (1.21), the SVM formulation with a 1-norm minimization of w (1.26), the SVM formulation with an ∞-norm minimization of w (1.25) and the RLP (1.17) is given in [6]. The FSV (1.21) and the SVM 1-norm (1.26) were the only methods tested that computed classifiers utilizing fewer than the original number of problem features. There was no significant difference in the generalization performance of these classifiers on 6 publicly available datasets [45].

Classifiers obtained by solving the SVM ∞-norm (1.25) suppressed none of the original problem features for all but the smallest values of ν (near 0), which in general is of little use because it is often accompanied by poor set separation.

4. MASSIVE DATA DISCRIMINATION VIA LINEAR SUPPORT VECTOR MACHINES

We consider in this section the problem of discriminating between two massive finite point sets in R^n by a linear programming approach to SVMs. In the standard SVM formulation [50, 49, 52] large quadratic programs are solved. In contrast, the formulation here consists of solving a linear program which is considerably less difficult. The algorithm creates a succession of sufficiently small linear programs that separate chunks of the data at a time. The key idea is that a small number of support vectors, corresponding to linear programming constraints with positive dual variables, are carried over between the successive small linear programs, each of which contains a chunk of the data. We prove that this procedure is monotonic and terminates in a finite number of steps at an exact solution that leads to an optimal separating plane for the entire dataset.

4.1 LSVM: THE LINEAR SUPPORT VECTOR MACHINE

Given m points in R^n represented by the matrix $A \in R^{m\times n}$ with the class label for data point A_i given by the corresponding diagonal element D_{ii} of the diagonal matrix $D \in R^{m\times m}$, we wish to discriminate between points A_i with $D_{ii} = 1$ from those A_i with $D_{ii} = -1$. A separating plane may be efficiently computed by solving the robust linear program (RLP) (1.17) [4]:

$$
\begin{array}{rl}
\min\limits_{w,\gamma,y} & e'y \\
\text{s.t.} & D(Aw - e\gamma) + y \geq e \\
 & y \geq 0.
\end{array} \tag{1.27}
$$

The objective function of (1.27) minimizes the sum of distances, weighted by $\|w\|'$, of misclassified points to the bounding planes: $x'w = \gamma + 1$ for points A_i with $D_{ii} = 1$ and $x'w = \gamma - 1$ for points A_i with $D_{ii} = -1$. The linear support vector machine (LSVM) consists of parametrically adding another term, $\|w\|'$, to the objective function of (1.27). Sections 1 and 2 provide detailed justification for this additional term.

Thus, appending $\|w\|_1$ to ν times the objective function of (1.27) with $\nu > 0$ leads to the following LSVM where $e's = \|w\|_1$ at optimality:

$$
\begin{array}{rl}
\min\limits_{w,\gamma,y,s} & \nu e'y + e's \\
\text{s.t.} & D(Aw - e\gamma) + y \geq e \\
 & -s \leq w \leq s \\
 & y \geq 0.
\end{array} \tag{1.28}
$$

4.2 LPC: LINEAR PROGRAM CHUNKING

We consider a general linear program

$$\min_x \left\{ c'x \,|\, Hx \geq b \right\}, \tag{1.29}$$

where $c \in R^n$, $H \in R^{m\times n}$ and $b \in R^m$. We state now our chunking algorithm and establish its finite termination for the linear program (1.29) where m may be orders of magnitude larger than n. In its dual form our algorithm can be interpreted as a block-column generation method related to column generation methods of Gilmore-Gomory [24], Dantzig-Wolfe [18], [14, pp 198-200,428-429] and others [46, pp 243-248], but it differs from active set methods [30, pp 326-330] in that it does not require the satisfaction of a working set of constraints as equalities.

Algorithm 3 LPC: Linear Programming Chunking Algorithm for (1.29) *Let $[H \quad b]$ be partitioned into ℓ blocks, possibly of different sizes, as follows:*

$$\left[\begin{array}{cc} H & b \end{array}\right] = \left[\begin{array}{cc} H^1 & b^1 \\ \vdots & \vdots \\ H^\ell & b^\ell \end{array}\right].$$

Assume that (1.29) and all subproblems (1.30) below, have vertex solutions. At iteration $j = 1, \ldots$ compute x^j by solving the following linear program:

$$x^j \in arg\ vertex\ min \left\{ c'x \;\middle|\; \begin{array}{lcl} H^{(j\ mod\, \ell)}\, x & \geq & b^{(j\ mod\, \ell)} \\ \bar{H}^{(j\ mod\, \ell)-1}\, x & \geq & \bar{b}^{(j\ mod\, \ell)-1} \end{array} \right\}, \quad (1.30)$$

where $[\bar{H}^0 \quad \bar{b}^0]$ is empty and $[\bar{H}^j \quad \bar{b}^j]$ is the set of active constraints (that is all inequalities of (1.29) satisfied as equalities by x^j) with positive optimal Lagrange multipliers at iteration j. Stop when $c'x^j = c'x^{j+\tau}$ for some input integer τ. Typically $\tau = 4$.

Theorem 4 Finite Termination of LPC Algorithm *The sequence $\{x^j\}$ generated by the LPC Algorithm 3 has the following properties:*

(i) The sequence $\{c'x^j\}$ of objective function values is nondecreasing and is bounded above by the minimum of $\min_x \{c'x \,|Hx \geq b\}$.

(ii) The sequence of objective function values $\{c'x^j\}$ becomes constant, that is: $c'x^{j+1} = c'x^j$ for all $j \geq \bar{j}$ for some $\bar{j} \geq 1$.

(iii) For $j \geq \bar{j}$, active constraints of (1.30) at x^j with positive multipliers remain active for iteration $j+1$.

(iv) For all $j \geq \tilde{j}$, for some $\tilde{j} \geq \bar{j}$, x^j is a solution of the linear program (1.29) provided all active constraints at x^j have positive multipliers for $j \geq \bar{j}$.

This theorem is significant not only for the support vector application presented here, but also as a fundamental computational approach for handling linear programs with massive constraints for which the subproblems (1.30) of the LPC Algorithm 3 have vertex solutions. To establish its validity we first prove a lemma.

Lemma 5 *If $\bar{x}$ solves the linear program (1.29) and $(\bar{x}, \bar{u}) \in R^{n+m}$ is a Karush-Kuhn-Tucker (KKT) [34] point (i.e. a primal-dual optimal pair)* *such that $\bar{u}_I > 0$ where $I \subset \{1, \dots, m\}$ and $\bar{u}_J = 0,\ J \subset \{1, \dots, m\}$, $I \cup J = \{1, \dots, m\}$, then*

$$\bar{x} \in arg\min\{c'x \,|\, H_I x \geq b_I\} \tag{1.31}$$

where H_I consists of rows $H_i,\ i \in I$ of H, and b_I consists of elements $b_i,\ i \in I$.

Proof The KKT conditions [34] for (1.29) satisfied by $(\bar{x}, \bar{u})$ are:

$$c = H'\bar{u},\ \bar{u} \geq 0,\ \bar{u}'(H\bar{x} - b) = 0,\ H\bar{x} - b \geq 0,$$

which under the condition $\bar{u}_I > 0$ imply that

$$H_I\bar{x} = b_I,\ \bar{u}_J = 0,\ H_J\bar{x} \geq b_J.$$

We claim now that $\bar{x}$ is also a solution of (1.31) because $(\bar{x}, \bar{u}_I)$ satisfy the KKT sufficient optimality conditions for (1.31):

$$c = H_I{}'\bar{u}_I,\ \bar{u}_I > 0,\ H_I\bar{x} = b_I.\diamond$$

Proof of Theorem 4

(i) By Lemma 5, $c'x^j$ is a lower bound for $c'x^{j+1}$. Hence the sequence $\{c'x^j\}$ is nondecreasing. Since the the constraints of (1.30) form a subset of the constraints of (1.29), it follows that $c'x^j \leq \min\limits_x \left\{c'x \,|\, Hx \geq b\right\}$.

(ii) Since there are a finite number of (feasible and infeasible) vertices to the original linear program (1.29) as well as of the subproblems (1.30), it follows that from a certain $\bar{j}$ onward, a finite subset of these vertices will repeat infinitely often. Since a repeated vertex gives the same value for $c'x$, it follows, by the nondecreasing property of $\{c'x^j\}$ just established, that all vertices between repeated vertices also have the same objective value $c'x$ and hence: $c'x^j = c'x^{j+1} \leq \min\limits_x \left\{c'x \,|\, Hx \geq b\right\},\ \forall j \geq \bar{j}$.

(iii) Let $\bar{j}$ be as defined in the theorem. Let the index $t \in \{1, \dots, m\}$ be that of some active constraint at iteration $\bar{j}$ with positive multiplier ($H_t x^{\bar{j}} = b_t,\ u_t^{\bar{j}} > 0$) which has become inactive at the next step, that is: $H_t x^{\bar{j}+1} > b_t$. We then obtain the following contradiction by part (ii) above and the KKT saddlepoint condition:

$$0 = c'x^{\bar{j}+1} - c'x^{\bar{j}} \geq u^{\bar{j}\prime}(\bar{H}^{\bar{j}}x^{\bar{j}+1} - \bar{b}^{\bar{j}}) \geq u_t^{\bar{j}}(H_t x^{\bar{j}+1} - b_t) > 0.$$

(iv) By (ii) a finite number of vertices repeat infinitely for $j \geq \bar{j}$ all with constant $c'x^j$. Since active constraints with positive multipliers at iteration j remain active at iteration $j+1$ by (iii) and hence have a positive multiplier by assumption of part (iv), the set of active constraints with positive multipliers will remain constant for $j \geq \tilde{j}$, for some $\tilde{j} \geq \bar{j}$ (because there are a finite number of constraints) and hence x^j will remain a fixed vertex $\bar{x}$ for $j \geq \tilde{j}$. The point $\bar{x}$ will satisfy all the constraints of problem (1.29) because all constraints are eventually imposed on the infinitely repeated vertex $\bar{x}$. Hence $c'\bar{x}$ which lower-bounds the minimum of (1.29) is also a minimum of (1.29) because $\bar{x}$ is feasible. Hence the algorithm can be terminated at $j = \tilde{j}$. $\diamond$

Remark 6 *We have not excluded the possibility (never observed computationally) that the objective function remains constant over a finite number of iterations then increases. Theorem 4 asserts that eventually the objective function will be constant and equal to the minimum for all iterates $j \geq \bar{j}$. Of course, one can check the satisfaction of the KKT conditions for optimality, although this does not seem to be needed in practice.*

Remark 7 *In order to handle degenerate linear programs, i.e. those with active constraints with zero multipliers at a solution, we have modified the computational implementation of the LPC Algorithm 3 slightly by admitting into $[\bar{H}^j \quad \bar{b}^j]$ all active constraints at iteration j even if they have zero multipliers. We note that the assumption of part (iv) of Theorem 4 is required to prove that x^j is a solution of (1.29) for all $j > \bar{j}$.*

The LPC Algorithm 3 was experimentally evaluated in [7] and some results are summarized here. Applied to a separation task of 200,000 points in R^{32}, the LPC algorithm computed a solution in 1.75 hours while 6.94 hours were required to solve the full dataset linear program. The chunk size corresponding to this running time was 25,000 rows (1/8 total dataset size). Separation of 500,000 data points via LPC required 25.91 hours and separation of 1 million data points in R^{32} required 231.32 hours.

We note that there exists a chunk size which is empirically best in the sense that it balances the computational overhead of constructing the LPC subproblems (1.30) with the computational overhead of solving the LPC subproblems. For small chunk sizes, the computational burden is dominated by subproblem construction since there are many and solving them is not expensive. For large chunk sizes, the computational burden is dominated by solving large subproblem LPs.

5. SUCCESSIVE OVERRELAXATION FOR SUPPORT VECTOR MACHINES

Successive overrelaxation, originally developed for the solution of large systems of linear equations [48, 47] has been successfully applied to mathematical programming problems [17, 35, 36, 51, 40, 29], some with as many 9.4 million variables [20]. By taking the dual of the quadratic program associated with an SVM [57, 13] for which the margin (distance between bounding separating planes) has been maximized with respect to *both* the normal to the planes as well as their location, we obtain a very simple convex quadratic program with bound constraints *only*. This problem is equivalent to a symmetric mixed linear complementarity problem (i.e. with upper and lower bounds on its variables [21]) to which SOR can be directly applied. This corresponds to solving the SVM dual convex quadratic program for one variable at a time, that is computing one multiplier of a potential support vector at a time.

We again consider the problem of discriminating between 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 classes 1 or -1 as specified by a given $m \times m$ diagonal matrix D with ones or minus ones along its diagonal. For this problem the standard linear SVM with a linear kernel AA' [57, 13] is given by the following for some $\nu > 0$:

$$\begin{array}{rrcl} \min\limits_{w,\gamma,y} & \nu e'y + \frac{1}{2}w'w & & \\ \text{s.t.} & D(Aw - e\gamma) + y & \geq & e \\ & y & \geq & 0. \end{array} \tag{1.32}$$

Note that this is a special case of problem (1.6), with $w = A'Du$, $H = DK(A, A')D$, and $K(A, A') = AA'$. Here w is the normal to the bounding planes:

$$\begin{array}{rcccl} x'w & - & \gamma & = & +1 \\ x'w & - & \gamma & = & -1. \end{array} \tag{1.33}$$

The first plane above bounds the class 1 points and the second plane bounds the class -1 points, if the two classes are linearly separable and $y = 0$. If the classes are linearly inseparable then the two planes bound the two classes with a "soft margin" determined by a slack variable $y \geq 0$, that is:

$$\begin{array}{rcccccccl} x'w & - & \gamma & + & y_i & \geq & +1, & \text{for } x = A_i \text{ and } D_{ii} = +1, \\ x'w & - & \gamma & - & y_i & \leq & -1, & \text{for } x = A_i \text{ and } D_{ii} = -1. \end{array} \tag{1.34}$$

The one-norm of the slack variable y is minimized with weight ν in (1.32). The quadratic term in (1.32), which is twice the reciprocal of the square of the 2-norm distance $\frac{2}{\|w\|_2}$ between the two planes of (1.33) in the n-dimensional space of $w \in R^n$ for a *fixed* γ, maximizes that distance. In our approach here, which is similar to that of [5, 23, 22], we measure the distance between the planes in the $(n+1)$-dimensional space of $[w; \gamma] \in R^{n+1}$ which is $\frac{2}{\|[w;\gamma]\|_2}$, instead of $\frac{2}{\|w\|_2}$. Using this measure of distance instead results in our modification of the SVM problem:

$$\begin{array}{lrcl} \min\limits_{w,\gamma,y} & \nu e'y + \frac{1}{2}(w'w + \gamma^2) & & \\ \text{s.t.} & D(Aw - e\gamma) + y & \geq & e \\ & y & \geq & 0. \end{array} \tag{1.35}$$

The Wolfe duals [34, Section 8.2] to the quadratic programs (1.32) and (1.35) are as follows.

$$\begin{array}{c} \max\limits_{u} \; -\dfrac{1}{2}u'DAA'Du + e'u, \text{ s.t. } e'Du = 0, \; 0 \leq u \leq \nu e \\ (w = A'Du). \end{array} \tag{1.36}$$

$$\begin{array}{c} \max\limits_{u} \; -\dfrac{1}{2}u'DAA'Du - \dfrac{1}{2}u'Dee'Du + e'u, \text{ s.t. } 0 \leq u \leq \nu e \\ (w = A'Du, \; \gamma = -e'Du, \; y = (e - D(Aw - e\gamma))_+). \end{array} \tag{1.37}$$

The principal difference between these duals is that (1.37) has simple bound constraints only, whereas (1.36) has in addition a computationally complicating equality constraint. We also note that the variables (w, γ, y) of the primal problem (1.35) can be directly computed from the solution u of its dual (1.37) as indicated. However, only the variable w of the primal problem (1.32) can be directly computed from the solution u of its dual (1.36) as shown. The remaining variables (γ, y) of (1.32) can be computed by setting $w = A'Du$ in (1.32), where u is a solution of its dual (1.36), and solving the resulting linear program for (γ, y). Alternatively, γ can be determined by minimizing the expression for $e'y = e'(e - D(Aw - e\gamma))_+$ as a function of the single variable γ after w has been expressed as a function of the dual solution u as indicated in (1.36), that is:

$$\min_{\gamma \in R} e'(e - D(AA'Du - e\gamma))_+. \tag{1.38}$$

We note that formulations (1.36) and (1.37) can be extended to a general nonlinear kernel $K(A,A') : R^{m\times n} \times R^{n\times m} \rightarrow R^{m\times m}$ by replacing AA' by the kernel $K(A,A')$. (For more details about kernel function choices, see Section 2.) Thus the conventional SVM with a positive semidefinite kernel $K(A,A')$ [57, 13] is given by the following dual convex quadratic program for some $\nu > 0$ which is obtained from (1.36) by replacing AA' by $K(A,A')$:

$$\begin{array}{ll} \min\limits_{u\in R^m} & \frac{1}{2}u'DK(A,A')Du - e'u \\ & \text{s.t.} \quad \begin{array}{rcl} e'Du & = & 0 \\ 0 \leq u & \leq & \nu e. \end{array} \end{array} \tag{1.39}$$

A solution u of this quadratic program leads to the nonlinear separating surface

$$K(x',A')Du = \gamma, \tag{1.40}$$

where γ is defined in a similar manner to (1.38) above by a solution of:

$$\min_{\gamma\in R} e'(e - D(K(A,A')Du - e\gamma))_{+}. \tag{1.41}$$

Note that the explicit definition of the nonlinear surface (1.40) in R^n in terms of u and γ obviates the need for computing w, since w is defined only in the higher dimensional space in which the nonlinear surface (1.40) is mapped into a plane.

By a similar approach we obtain the following quadratic dual with bound constraints only and a nonlinear kernel $K(A,A')$ by replacing AA' in (1.37) by $K(A,A')$:

$$\begin{array}{c} \min\limits_{u} \dfrac{1}{2}u'D[K(A,A') + ee']Du - e'u, \text{ s.t. } 0 \leq u \leq \nu e \\ \\ (\gamma = -e'Du), \end{array} \tag{1.42}$$

with an explicit expression for γ and the same separating surface (1.40). The formulation (1.42) allows a direct use of the SOR algorithm to solve very large problems.

We mention in passing that another possible change in (1.39) allows the use of possibly indefinite kernels. One particular formulation motivated by (1.10) is the following one which, as in (1.42), incorporates the equality $e'Du = 0$ into the objective function:

$$\min_u \frac{1}{2}u'D[K(A,A')K(A,A')' + ee']Du - e'u, \text{ s.t. } 0 \leq u \leq \nu e \\ (\gamma = -e'Du), \tag{1.43}$$

with a separating surface different from (1.40):

$$K(x', A')K(A, A')'Du = \gamma. \tag{1.44}$$

Note that the kernel $K(A, A')$ in the formulation (1.43) is completely arbitrary and need not satisfy any positive semidefiniteness condition in order for the objective function of (1.43) to be convex. This makes the separating surface (1.44) quite general.

It is interesting to note that for a linear kernel, the standard SVM problem (1.36) and our modification (1.37) frequently give the same w. For $1,000$ randomly generated problems with $A \in R^{40\times 5}$ and the same ν, only 34 cases had solution vectors w that differed by more than 0.001 in their 2-norm.

The main reason for introducing our modification (1.35) of the SVM is that its dual (1.37) does not contain an equality constraint as does the dual (1.36) of (1.32). This enables us to apply in a straightforward manner the effective matrix splitting methods such as those of [35, 36, 31] that process one constraint of (1.35) at a time through its dual variable, without the complication of having to enforce an equality constraint at each step on the dual variable u. This permits us to process massive data without bringing it all into fast memory. We thus apply the SOR algorithm to the nonlinear SVMs (1.42) and (1.43). Note that the linear SVM is simply the special case of (1.42) where $K(A, A') = AA'$. These problems can be stated as:

$$\min_u \frac{1}{2}u'Mu - e'u, \text{ s.t. } u \in S = \{u \mid 0 \leq u \leq \nu e\}, \tag{1.45}$$

with the symmetric matrix M defined as $D(K(A, A') + ee')D$ and $D(K(A, A')K(A, A')' + ee')D$ respectively for these two problems. Thus M will be positive semidefinite if we assume that $K(A, A')$ is positive semidefinite in the former case and under no assumptions in the latter case. If we decompose M as follows:

$$M = L + E + L', \tag{1.46}$$

where $L \in R^{m\times m}$ is the strictly lower triangular part of the symmetric matrix M, and $E \in R^{m\times m}$ is the positive diagonal of M, then a necessary and sufficient optimality condition for (1.45) for positive semidefinite M is the following gradient projection optimality condition [53, 31]:

$$u = (u - \omega E^{-1}(Mu - e))_{\#},\ \omega > 0, \tag{1.47}$$

where $(\cdot)_{\#}$ denotes the 2-norm projection on the feasible region S of (1.45), that is:

$$((u)_{\#})_i = \left\{ \begin{array}{l} 0 \ \text{ if } u_i \leq 0 \\ u_i \ \text{ if } 0 < u_i < \nu \\ \nu \ \text{ if } u_i \geq \nu \end{array} \right\},\ i = 1, \ldots, m. \tag{1.48}$$

Our SOR method, which is a matrix splitting method that converges linearly to a point $\bar{u}$ satisfying (1.47), consists of splitting the matrix M into the sum of two matrices as follows:

$$M = \omega^{-1}E(B + C),\ \text{s.t. } B - C \text{ is positive definite.} \tag{1.49}$$

For our specific problem we take:

$$B = (I + \omega E^{-1}L),\ C = ((\omega - 1)I + \omega E^{-1}L'),\ 0 < \omega < 2. \tag{1.50}$$

This leads to the following linearly convergent [31, Equation (3.14)] matrix splitting algorithm:

$$u^{i+1} = (u^{i+1} - Bu^{i+1} - Cu^i + \omega E^{-1}e)_{\#}, \tag{1.51}$$

for which

$$B + C = \omega E^{-1}M,\ B - C = (2 - \omega)I + \omega E^{-1}(L - L'). \tag{1.52}$$

Note that for $0 < \omega < 2$, the matrix $B + C$ is positive semidefinite and matrix $B - C$ is positive definite. The matrix splitting algorithm (1.51) results in the following easily implementable SOR algorithm once the values of B and C given in (1.50) are substituted in (1.51).

Algorithm 8 SOR Algorithm *Choose $\omega \in (0, 2)$. Start with any $u^0 \in R^m$. Having u^i compute u^{i+1} as follows:*

$$u^{i+1} = (u^i - \omega E^{-1}(Mu^i - e + L(u^{i+1} - u^i)))_{\#}, \tag{1.53}$$

until $\|u^{i+1} - u^i\|$ is less than some prescribed tolerance.

Remark 9 *The components of u^{i+1} are computed in order of increasing component index. Thus the SOR iteration (1.53) consists of computing u_j^{i+1} using $(u_1^{i+1}, \ldots, u_{j-1}^{i+1}, u_j^i, \ldots, u_m^i)$. That is, the latest computed* *components of u are used in the computation of u_j^{i+1}. The strictly lower triangular matrix L in (1.53) can be thought of as a substitution operator, substituting $(u_1^{i+1}, \dots, u_{j-1}^{i+1})$ for $(u_1^i, \dots, u_{j-1}^i)$.*

We have immediately from [31, Proposition 4.21] the following linear convergence result.

Theorem 10 SOR Linear Convergence *The iterates $\{u^i\}$ of the SOR Algorithm 8 converge R-linearly to a solution of $\bar{u}$ of the dual problem (1.42), and the objective function values $\{f(u^i)\}$ of (1.42) converge Q-linearly to $f(\bar{u})$. That is for $i \geq \bar{i}$ for some $\bar{i}$:*

$$\begin{array}{rcl} \|u^i - \bar{u}\| & \leq & \mu\delta^i, \text{ for some } \mu > 0,\ \delta \in (0,1), \\ f(u^{i+1}) - f(\bar{u}) & \leq & \tau(f(u^i) - f(\bar{u})), \text{ for some } \tau \in (0,1). \end{array} \tag{1.54}$$

Remark 11 *A significant simplification can be made when a linear kernel is used in (1.45) with $M = D(AA' + ee')D$. The matrix M can be rewritten as HH', where $H = D[A \quad -e]$. Even though our SOR iteration (1.53) is written in terms of the full $m \times m$ matrix HH', it can easily be implemented one row at a time without bringing all of the data into memory as follows for $j = 1, \dots, m$:*

$$u_j^{i+1} = (u_j^i - \omega E_{jj}^{-1}(H_j(\sum_{\ell=1,\, j>1}^{j-1} H_\ell u_\ell^{i+1} + \sum_{\ell=j}^{m} H_l u_\ell^i - 1))_{\#}. \tag{1.55}$$

A simple interpretation of this step is that one component of the multiplier u_j is updated at a time by bringing one constraint of (1.35) at a time.

We benchmarked the SOR algorithm with a linear kernel against the SMO [52] and SVM^{light} [28] algorithms using subsets of the UCI Adult dataset [45]. For larger versions of the dataset, SOR ran almost twice as fast as SMO, and more than an order of magnitude faster than SVM^{light}. SOR produced similar test set accuracies to the other two algorithms. See [41] for more details.

SOR was also used with a linear kernel on synthetic highly separable massive datasets [41]. The algorithm terminated in 9.7 hours on a one million point dataset, and reached 95% of true separability on a 10 million point dataset in only 14.3 hours. Finally, SOR with a *nonlinear* kernel was used on a synthetic Gaussian dataset [42]. Both a linear and quadratic kernel were used, with the quadratic kernel showing improved test set accuracy over the linear kernel. The quadratic kernel yielded a test set accuracy of 93.4%, whereas the linear kernel produced a test set accuracy of only 81.7% under tenfold cross-validation.

6. CONCLUSION

We have proposed a direct mathematical programming framework for general SVMs that makes essentially no or few assumptions on the kernel employed. We have derived new kernel-based linear programming formulations (1.11) and (1.12), and a new quadratic programming formulation (1.10) that require no assumptions on the kernel K. These formulations can lead to different but equally satisfactory decision functions as that obtained by the quadratic programming formulation (1.9) for a conventional SVM that requires symmetry and positive definiteness of the kernel. Even for negative definite kernels these new formulations can generate decision functions that separate the given points whereas the conventional SVM does not.

Feature selection that improves generalization was achieved by parametrically minimizing a polyhedral norm or a concave function that suppresses as many components of the features as possible, while generating a separating surface that discriminates effectively between the points of a given set containing two categories of points.

We have described a linear programming chunking algorithm for discriminating between massive datasets. The algorithm, significant in its own right as a linear programming decomposition algorithm, is very effective for discrimination problems with large datasets that may not fit in machine memory and for problems taking excessive time to process. The algorithm uses support vector ideas by keeping only essential data points needed for determining a separating plane. The algorithm can handle extremely large datasets because it deals with small chunks of the data at a time and is guaranteed to terminate in a finite number of steps. Although we have not discussed parallelization here, this can be easily implemented by splitting the data among processors and sharing only support vectors among them. This would allow one to handle problems with extremely large datasets on a network of workstations or PCs.

We have described a powerful iterative method, successive overrelaxation, for the solution of extremely large discrimination problems using SVMs. The method converges considerably faster than other methods that require the presence of a substantial amount of the data in memory. We have solved problems that cannot be directly handled by conventional methods of mathematical programming. The proposed method scales up with no changes and can be parallelized by using techniques already implemented [19, 20]. We have also described how use successive overrelaxation in conjunction with nonlinear kernels to generate nonlinear separating surfaces.

Acknowledgements

The research described in this Data Mining Institute Report 99-01, June 1999, was supported by National Science Foundation Grants CCR-9729842 and CDA-9623632, by Air Force Office of Scientific Research Grant F49620-97-1-0326 and by the Microsoft Corporation.

References

[1] K. P. Bennett. Combining support vector and mathematical programming methods for induction. Unpublished manuscript, Department of Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, NY 12180, 1998.

[2] K. P. Bennett and J. A. Blue. A support vector machine approach to decision trees. In *Proceedings of IJCNN'98*, pages 2396–2401, Anchorage, Alaska, 1997. http://www.math.rpi.edu/∼bennek/.

[3] K. P. Bennett, D. Hui, and L. Auslender. On support vector decision trees for database marketing. Department of Mathematical Sciences Math Report No. 98-100, Rensselaer Polytechnic Institute, Troy, NY 12180, March 1998. http://www.math.rpi.edu/∼bennek/.

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

[5] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In D. Haussler, editor, *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pages 144–152, Pittsburgh, PA, July 1992. ACM Press.

[6] P. S. Bradley and O. L. Mangasarian. Feature selection via concave minimization and support vector machines. In J. Shavlik, editor, *Machine Learning Proceedings of the Fifteenth International Conference(ICML '98)*, pages 82–90, San Francisco, California, 1998. Morgan Kaufmann. ftp://ftp.cs.wisc.edu/math-prog/tech-reports/98-03.ps.Z.

[7] P. S. Bradley and O. L. Mangasarian. Massive data discrimination via linear support vector machines. Technical Report 98-05,

Computer Sciences Department, University of Wisconsin, Madison, Wisconsin, May 1998. *Optimization Methods and Software*, to appear. ftp://ftp.cs.wisc.edu/math-prog/tech-reports/98-03.ps.Z.

[8] P. S. Bradley, O. L. Mangasarian, and J. B. Rosen. Parsimonious least norm approximation. *Computational Optimization and Applications*, 11(1):5–21, October 1998. ftp://ftp.cs.wisc.edu/math-prog/tech-reports/97-03.ps.Z.

[9] P. S. Bradley, O. L. Mangasarian, and W. N. Street. Feature selection via mathematical programming. *INFORMS Journal on Computing*, 10(2):209–217, 1998. ftp://ftp.cs.wisc.edu/math-prog/tech-reports/95-21.ps.Z.

[10] E. J. Bredensteiner. *Optimization Methods in Data Mining and Machine Learning.* PhD thesis, Department of Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, NY, 1997.

[11] E. J. Bredensteiner and K. P. Bennett. Feature minimization within decision trees. *Computational Optimizations and Applications*, 10:111–126, 1998.

[12] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[13] V. Cherkassky and F. Mulier. *Learning from Data - Concepts, Theory and Methods.* John Wiley & Sons, New York, 1998.

[14] V. Chvátal. *Linear Programming.* W. H. Freeman and Company, New York, 1983.

[15] C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20:273–279, 1995.

[16] R. Courant and D. Hilbert. *Methods of Mathematical Physics.* Interscience Publishers, New York, 1953.

[17] C. W. Cryer. The solution of a quadratic programming problem using systematic overrelaxation. *SIAM Journal on Control and Optimization*, 9:385–392, 1971.

[18] G. B. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Operations Research*, 8:101–111, 1960.

[19] R. De Leone, O. L. Mangasarian, and T.-H. Shiau. Multi-sweep asynchronous parallel successive overrelaxation for the nonsymmetric linear complementarity problem. *Annals of Operations Research*, 22:43–54, 1990.

[20] R. De Leone and M. A. Tork Roth. Massively parallel solution of quadratic programs via successive overrelaxation. *Concurrency: Practice and Experience*, 5:623–634, 1993.

[21] S. P. Dirkse and M. C. Ferris. MCPLIB: A collection of nonlinear mixed complementarity problems. *Optimization Methods and Software*, 5:319–345, 1995. ftp: //ftp.cs.wisc.edu/tech-reports/reports/94/tr1215.ps.

[22] T.-T. Friess. Support vector neural networks: The kernel adatron with bias and soft margin. Technical report, Department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield, England, 1998. Revised Version: www.brunner-edv.com/friess/.

[23] T.-T. Friess, N. Cristianini, and C. Campbell. The kernel-adatron algorithm: A fast and simple learning procedure for support vector machines. In Jude Shavlik, editor, *Machine Learning Proceedings of the Fifteenth International Conference (ICML'98)*, pages 188–196, San Francisco, 1998. Morgan Kaufmann. http://svm.first.gmd.de/papers/FriCriCam98.ps.gz.

[24] P. C. Gilmore and R. E. Gomory. A linear programming approach to the cutting stock problem. *Operations Research*, 9:849–859, 1961.

[25] F. Girosi. An equivalence between sparse approximation and support vector machines. *Neural Computation*, 10(6):1455–1480, 1998. http://www.ai.mit.edu/people/girosi/home-page/svm.html.

[26] M. H. Hassoun. *Fundamentals of Artificial Neural Networks.* MIT Press, Cambridge, MA, 1995.

[27] P. J. Huber. *Robust Statistics.* John Wiley, New York, 1981.

[28] T. Joachims. Making large-scale support vector machine learning practical. In Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 169–184. MIT Press, 1999.

[29] W. Li. Remarks on convergence of matrix splitting algorithm for the symmetric linear complementarity problem. *SIAM Journal on Optimization*, 3:155–163, 1993.

[30] D. G. Luenberger. *Linear and Nonlinear Programming.* Addison–Wesley, second edition, 1984.

[31] Z.-Q. Luo and P. Tseng. Error bounds and convergence analysis of feasible descent methods: A general approach. *Annals of Operations Research*, 46:157–178, 1993.

[32] O. L. Mangasarian. Linear and nonlinear separation of patterns by linear programming. *Operations Research*, 13:444–452, 1965.

[33] O. L. Mangasarian. Multi-surface method of pattern separation. *IEEE Transactions on Information Theory*, IT-14:801–807, 1968.

[34] O. L. Mangasarian. *Nonlinear Programming.* McGraw–Hill, New York, 1969. Reprint: SIAM Classic in Applied Mathematics 10, 1994, Philadelphia.

[35] O. L. Mangasarian. Solution of symmetric linear complementarity problems by iterative methods. *Journal of Optimization Theory and Applications*, 22(4):465–485, August 1977.

[36] O. L. Mangasarian. On the convergence of iterates of an inexact matrix splitting algorithm for the symmetric monotone linear complementarity problem. *SIAM Journal on Optimization*, 1:114–122, 1991.

[37] O. L. Mangasarian. Machine learning via polyhedral concave minimization. In H. Fischer, B. Riedmueller, and S. Schaeffler, editors, *Applied Mathematics and Parallel Computing - Festschrift for Klaus Ritter*, pages 175–188. Physica-Verlag A Springer-Verlag Company, Heidelberg, 1996. ftp://ftp.cs.wisc.edu/math-prog/tech-reports/95-20.ps.Z.

[38] O. L. Mangasarian. Generalized support vector machines. Technical Report 98-14, Computer Sciences Department, University of Wisconsin, Madison, Wisconsin, October 1998. ftp://ftp.cs.wisc.edu/math-prog/tech-reports/98-14.ps.Z. NIPS*98 Workshop on Large Margin Classifiers, Breckenridge, Colorado, December 4-5, 1998. “Advances in Large Margin Classifiers”, A. Smola, P. Bartlett, B. Schölkopf and D. Schuurmans (editors), MIT Press, Cambridge, Massachusetts 1999, to appear.

[39] O. L. Mangasarian. Arbitrary-norm separating plane. *Operations Research Letters*, 24(1-2), 1999. ftp://ftp.cs.wisc.edu/math-prog/tech-reports/97-07r.ps.Z.

[40] O. L. Mangasarian and R. De Leone. Parallel gradient projection successive overrelaxation for symmetric linear complementarity problems. *Annals of Operations Research*, 14:41–59, 1988.

[41] O. L. Mangasarian and David R. Musicant. Successive overrelaxation for support vector machines. Technical Report 98-18, Computer Sciences Department, University of Wisconsin, Madison, Wisconsin, October 1998. ftp://ftp.cs.wisc.edu/math-prog/tech-reports/98-18.ps.Z. IEEE Transactions on Neural Networks, to appear.

[42] O. L. Mangasarian and David R. Musicant. Data discrimination via nonlinear generalized support vector machines. Technical Report 99-03, Computer Sciences Department, University of Wisconsin, Madison, Wisconsin, March 1999. ftp://ftp.cs.wisc.edu/math-prog/tech-reports/99-03.ps.Z.

[43] O. L. Mangasarian, W. N. Street, and W. H. Wolberg. Breast cancer diagnosis and prognosis via linear programming. *Operations Research*, 43(4):570–577, July-August 1995.

[44] MATLAB. *User's Guide.* The MathWorks, Inc., Natick, MA 01760, 1992.

[45] P. M. Murphy and D. W. Aha. UCI repository of machine learning databases, 1992. www.ics.uci.edu/∼mlearn/MLRepository.html.

[46] K. G. Murty. *Linear Programming.* John Wiley & Sons, New York, 1983.

[47] J. M. Ortega. *Numerical Analysis, A Second Course.* Academic Press, 1972.

[48] J. M. Ortega and W. C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables.* Academic Press, 1970.

[49] E. Osuna, R. Freund, and F. Girosi. Improved training algorithm for support vector machines. In *Proceedings of IEEE NNSP'97, Amelia Island, FL, September 1997, 276-285*, New York, 1997. IEEE Press. http://www.ai.mit.edu/people/girosi/home-page/svm.html.

[50] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: An application to face detection. In *IEEE Conference on Computer Vision and Pattern Recognition, Puerto Rico, June 1997, 130-136*, 1997. http://www.ai.mit.edu/people/girosi/home-page/svm.html.

[51] J.-S. Pang. More results on the convergence of iterative methods for the symmetric linear complementarity problem. *Journal of Optimization Theory and Applications*, 49:107–134, 1986.

[52] J. Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. In Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 185–208. MIT Press, 1999. http://www.research.microsoft.com/∼jplatt/smo.html.

[53] B. T. Polyak. *Introduction to Optimization*. Optimization Software, Inc., Publications Division, New York, 1987.

[54] B. Schölkopf. *Support Vector Learning*. R. Oldenbourg Verlag, Munich, 1997.

[55] A. J. Smola. *Learning with Kernels*. PhD thesis, Technische Universität Berlin, Berlin, Germany, 1998.

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

[57] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.

[58] G. Wahba. Support vector machines, reproducing kernel Hilbert spaces and the randomized GACV. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 69–88, Cambridge, MA, 1999. MIT Press. ftp://ftp.stat.wisc.edu/pub/wahba/index.html.

[59] P. Wolfe. A duality theorem for nonlinear programming. *Quarterly of Applied Mathematics*, 19:239–244, 1961.