Generalized Support Vector Machines

O. L. Mangasarian
Computer Sciences Department
University of Wisconsin
1210 West Dayton Street
Madison, WI 53706
email: olvi@cs.wisc.edu

Abstract

By setting apart the two functions of a support vector machine: separation of points by a nonlinear surface in the original space of patterns, and maximizing the distance between separating planes in a higher dimensional space, we are able to define indefinite, possibly discontinuous, kernels, not necessarily inner product ones, that generate highly nonlinear separating surfaces. Maximizing the distance between the separating planes in the higher dimensional space is surrogated by support vector suppression, which is achieved by minimizing any desired norm of support vector multipliers. The norm may be one induced by the separation kernel if it happens to be positive definite, or a Euclidean or a polyhedral norm. The latter norm leads to a linear program whereas the former norms lead to convex quadratic programs, all with an arbitrary separation kernel. A standard support vector machine can be recovered by using the same kernel for separation and support vector suppression. On a simple test example, all models perform equally well when a positive definite kernel is used. When a negative definite kernel is used, we are unable to solve the nonconvex quadratic program associated with a conventional support vector machine, while all other proposed models remain convex and easily generate a surface that separates all given points.

1 Introduction

Support vector machines [18, 1, 13, 19, 7, 16, 17] attempt to separate points belonging to two given sets in n-dimensional real (Euclidean) space 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 [18] 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 [8, 18] to maximize the distance between planes that bound each set with certain minimal error.

In this paper we start with a nonlinear separating surface (1), implicitly defined by some chosen kernel and by some linear parameters $u \in R^m$, to be determined, that turn out to be closely related to some dual variables. Based on this surface we derive a general convex mathematical program (5) that attempts separation via the nonlinear surface (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 support vector machines. 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), then we are led to the conventional dual quadratic program (9) associated with a support vector machine 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 (10) *without* making any assumptions on the kernel. Another class of support vector machines are generated by choosing a piecewise linear convex function for f and this leads to linear programs such as (11) and (12), both of which make no assumptions on the kernel. In Section 5 we give some simple applications of all four formulations to the Exclusive-Or (XOR) problem using first a positive definite second-order polynomial kernel and then a negative definite third-order polynomial kernel. For the positive definite kernel all four convex formulations are easily solved and the resulting nonlinear surface separates all points in all cases. However, for the negative definite kernel, a powerful state-of-the-art package fails to solve the nonconvex quadratic program associated with the conventional support vector machine, whereas all other three convex formulations are easily solved and lead to complete separation of the data by the nonlinear surface.

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 step function x_* of $x \in R^n$ is defined as a vector of ones and zeros in R^n, with ones corresponding to positive components of x and zeros corresponding to nonpositive components. The scalar (inner) product of two vectors x and y in the n-dimensional real space R^n will be denoted by $x'y$. For an $m \times n$ matrix A, A_i will denote the ith row of A and $A_{\cdot j}$ will denote the jth column of A. The identity matrix in a real space of arbitrary dimension will be denoted by I, while a column vector of ones of arbitrary dimension will be denoted by e. We shall employ the MATLAB "dot" notation [15] 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. The base of the natural logarithm will be denoted by

ε.

We begin by defining a general kernel function as follows.

Definition 1.1 *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 1.2 Polynomial Kernel $(AB + \mu ab')^d_\bullet$

Example 1.3 Neural Network Kernel $(AB + \mu ab')_{\bullet *}$

Example 1.4 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 [18, 7].

2 GSVM: The General Support Vector Machine

We consider a given set $\mathcal{A}$ of m points in real n-dimensional space of features 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:

$$K(x', A')Du = \gamma, \tag{1}$$

where $K(x', A') \in R^m$, according to Definition 1.1. 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*

$$(K(x', A')Du - \gamma)_*, \tag{2}$$

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 [9, pp 138-140],[18, 7, 6] (an assumption that we will not make in this paper) then for x and y in R^n:

$$K(x, y) = h(x)'h(y), \tag{3}$$

and the separating surface (1) becomes:

$$h(x)'h(A')Du = \gamma, \tag{4}$$

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) 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:

$$\begin{array}{rrcl} \min\limits_{u,\gamma,y} & \nu e'y + f(u) & & \\ \text{s.t.} & D(K(A,A')Du - e\gamma) + y & \geq & e \\ & y & \geq & 0. \end{array} \tag{5}$$

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 (5) with positive 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 (9) below. As is well known, this leads to improved generalization by minimizing an upper bound on the VC dimension [18, 16].

We term a solution of the mathematical program (5) and the resulting decision function (2) a *generalized support vector machine*, GSVM. In the following sections of the paper we derive a number of special cases, including the standard support vector machine. First, however, it is important to state under what conditions does the mathematical program (5) have a solution.

Proposition 2.1 Existence of a GSVM *For any given $A \in R^{m\times n}$, any $D \in R^{m\times m}$, $\nu > 0$ and any kernel K, the mathematical program (5) has a solution whenever f is a piecewise-linear or quadratic function bounded below on R^m.*

Proof The feasible region of (5) is always nonempty. Just take: $u = 0$, $\gamma = 0$ and $y = e$. When f is piecewise-linear, existence follows from the standard linear programming result, that a feasible linear program with a bounded objective has a solution. Just apply this result to each piece of the objective on its polyhedral region. For a quadratic f the result is a direct consequence of the Frank-Wolfe existence result for quadratic programming [12]. $\diamondsuit$

We note that no convexity of f was needed for this existence result. However in our specific applications where duality theory will be invoked, f will need to be convex.

3 Quadratic Programming Support Vector Machines

We consider in this section support vector machines that include the standard ones [18, 7, 6] and which are obtained by setting f of (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 (5) becomes the following convex quadratic program:

$$
\begin{array}{rrcl}
\min\limits_{u,\gamma,y} & \nu e'y + \frac{1}{2}u'Hu & & \\
\text{s.t.} & D(K(A,A')Du - e\gamma) + y & \geq & e \\
 & y & \geq & 0.
\end{array}
\tag{6}
$$

The Wolfe dual [20, 14] of this convex quadratic program is:

$$
\begin{array}{rrcl}
\min\limits_{r\in R^m} & \frac{1}{2}r'DK(A,A')DH^{-1}DK(A,A')'Dr - e'r & & \\
\text{s.t.} & e'Dr & = & 0 \\
 & 0 \leq r & \leq & \nu e.
\end{array}
\tag{7}
$$

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

$$
u = H^{-1}DK(A,A')'Dr. \tag{8}
$$

If we assume that the kernel $K(A,A')$ is symmetric positive definite and let $H = DK(A,A')D$, then our dual problem (7) degenerates to the dual problem of the standard support vector machine [18, 7, 6] with $u = r$:

$$
\begin{array}{rrcl}
\min\limits_{u\in R^m} & \frac{1}{2}u'DK(A,A')Du - e'u & & \\
\text{s.t.} & e'Du & = & 0 \\
 & 0 \leq u & \leq & \nu e.
\end{array}
\tag{9}
$$

The positive definiteness assumption on $K(A,A')$ in (9) can be relaxed to positive *semi*definiteness while maintaining the convex quadratic program (6), with $H = DK(A,A')D$, as the direct dual of (9) without utilizing (7) and (8). The symmetry and positive semidefiniteness of the kernel $K(A,A')$ for this version of a support vector machine is consistent with the support vector machine literature. The fact that $r = u$ in the dual formulation (9), shows that the variable u appearing in the original formulation (6) is also the dual multiplier vector for the first set of constraints of (6). Hence the quadratic term in the objective function of (6) can be thought of as suppressing as many multipliers of support vectors as possible and thus minimizing the number of such support vectors. This is another interpretation of the standard support vector machine 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 (6) which leads to a dual problem (7) with $H = I$ and $u = DK(A, A')'Dr$ as follows:

$$\begin{array}{lrcl} \min\limits_{r \in R^m} & \frac{1}{2} r'DK(A, A')K(A, A')'Dr - e'r & & \\ \text{s.t.} & e'Dr & = & 0 \\ & 0 \leq r & \leq & \nu e. \end{array} \tag{10}$$

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 Proposition 2.1 the quadratic program (6) is solvable for *any* symmetric positive definite matrix H, and by quadratic programming duality so is its dual problem (7), the solution r of which can be immediately used to generate a decision function (2). Thus we are free to choose any symmetric positive definite matrix H to generate a support vector machine. Experimentation will be needed to determine what are the most appropriate choices for H.

We turn our attention to linear programming support vector machines.

4 Linear Programming Support Vector Machines

In this section we consider problems generated from the mathematical program (5) by using a piecewise linear function f in the objective function thus leading to linear programs.

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

$$\begin{array}{lrcl} \min\limits_{u,\gamma,y,s} & \nu e'y + e's & & \\ \text{s.t.} & D(K(A, A')Du - e\gamma) + y & \geq & e \\ & s \geq u & \geq & -s \\ & y & \geq & 0. \end{array} \tag{11}$$

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 (2). This linear program parallels the quadratic programming formulation (10) that was obtained as the dual of (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 (11). Another linear programming formulation that somewhat parallels the quadratic programming formulation (9), which was obtained as the dual of (5) by setting $f(u)$ therein to half the 2-norm squared of $K(A, A')^{\frac{1}{2}}Du$, is obtained setting f to be the 1-norm of $K(A, A')Du$. This leads to the following linear program:

$$\begin{array}{lrcl} \min\limits_{u,\gamma,y,s} & \nu e'y + e's & & \\ \text{s.t.} & D(K(A, A')Du - e\gamma) + y & \geq & e \\ & s \geq K(A, A')Du & \geq & -s \\ & y & \geq & 0. \end{array} \tag{12}$$

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 (9).

It is interesting to note that if the linear kernel $K(A, A') = AA'$ is used in the linear program (11) we obtain the high-performing 1-norm linear SVM proposed in [5] and utilized successfully in [4, 2, 3]. Hence, if we set $w = A'Du$ in (11) we obtain [3, Equation (13)].

5 A Simple Illustrative Example

We first demonstrate the workings and sometimes different, yet equally effective, decision surfaces obtained by the various proposed mathematical programming formulations, for a positive definite symmetric kernel. We then show that for a negative definite symmetric kernel, the conventional support vector machine fails to generate a decision function that correctly separates the given points, whereas all the new formulations do succeed in generating a decision surface that correctly separates all the given points.

For our positive definite kernel we use a polynomial kernel of order 2, based on Example 1.2 with $B = A'$, $\mu = 1$, $a = b = e$ and $d = 2$, and apply it to the classical Exclusive-Or (XOR) problem. We thus have:

$$A = \begin{bmatrix} 1 & 1 \\ 1 & -1 \\ -1 & -1 \\ -1 & 1 \end{bmatrix},\ D = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}. \tag{13}$$

Hence with the MATLAB [15] "dot" notation signifying componentwise exponentiation we get that:

$$K(A, A') = (AA' + ee')^2_\bullet = \begin{bmatrix} 9 & 1 & 1 & 1 \\ 1 & 9 & 1 & 1 \\ 1 & 1 & 9 & 1 \\ 1 & 1 & 1 & 9 \end{bmatrix}, \tag{14}$$

and

$$K(x', A') = (x'A' + e')^2_\bullet = [x_1 + x_2 + 1 \;\; x_1 - x_2 + 1 \;\; -x_1 - x_2 + 1 \;\; -x_1 + x_2 + 1]^2_\bullet. \tag{15}$$

Solution of the linear program (11) with $\nu \geq 1$ gives:

$$u = \frac{1}{8}e,\ \gamma = 0,\ y = 0,\ s = \frac{1}{8}e,\ \nu e'y + e's = \frac{1}{2}. \tag{16}$$

Note that the $y = 0$ in the above solution means that the decision surface correctly classifies all given points represented by A and D. Solution of either quadratic program (9) or (10) with the same kernel and for $\nu \geq 1$ also yields $u = \frac{1}{8}e$. Substitution of this u in (6) and solving the resulting linear program gives the same γ, y as in (16). Thus all mathematical programs (9), (10) and (11) yield exactly the same decision surface (2):

$$(K(x', A')Du - \gamma)_* = ((x'A' + e')^2_\bullet Du - \gamma)_* = (x_1x_2)_*, \tag{17}$$

a step function of the quadratic function x_1x_2, which correctly classifies the two categories class 1 and class -1 and is in agreement with the solution obtained in [7, pages 372-375] for the conventional support vector machine (9). Note that neither mathematical program (10) or (11) required positive definiteness of $K(A, A')$, whereas (9) does.

However, it is rather interesting to observe that the linear programming solution (16) is not unique. In fact another solution is the following:

$$u = \begin{bmatrix} 0 \\ \frac{1}{4} \\ 0 \\ \frac{1}{4} \end{bmatrix}, \ \gamma = -\frac{3}{2}, \ y = 0, \ s = \begin{bmatrix} 0 \\ \frac{1}{4} \\ 0 \\ \frac{1}{4} \end{bmatrix}, \ \nu e'y + e's = \frac{1}{2}. \tag{18}$$

For this solution the decision surface (2) turns out to be:

$$(K(x', A')Du - \gamma)_* = ((x'A' + e')^2_\bullet Du - \gamma)_* = \frac{1}{2}(2 - (x_1 - x_2)^2)_*. \tag{19}$$

This decision surface is rather different from that of (17), but it does separate the two classes correctly and in fact it consists of two parallel lines separating R^2 into 3 regions, whereas (17) separates R^2 into four quadrants each pair of which contains one class. Both of these decision functions are depicted in Figures 1 and 2.

Solution of the linear program (12) with $\nu > 1$ yields:

$$u = \begin{bmatrix} \frac{1}{24} \\ \frac{5}{24} \\ \frac{1}{24} \\ \frac{5}{24} \end{bmatrix}, \ \gamma = -1, \ y = 0, \ s = \begin{bmatrix} 0 \\ 2 \\ 0 \\ 2 \end{bmatrix}, \tag{20}$$

which gives the decision surface:

$$((x'A' + e')^2_\bullet Du - \gamma)_* = \frac{1}{3}(2 + x_1x_2 - (x_1 - x_2)^2)_*. \tag{21}$$

This decision function divides R^2 into three regions by a pair of "square root" curves that correctly classify the two classes as depicted in Figure 3.

Finally in order to show that positive definiteness of the kernel $K(A, A')$ is not essential in any of our new mathematical programming formulations (10), (11) or (12), whereas it is in the conventional quadratic programming formulation (9), we consider the following negative definite kernel:

$$K(A, A') = (-AA' - ee')^3_\bullet, \tag{22}$$

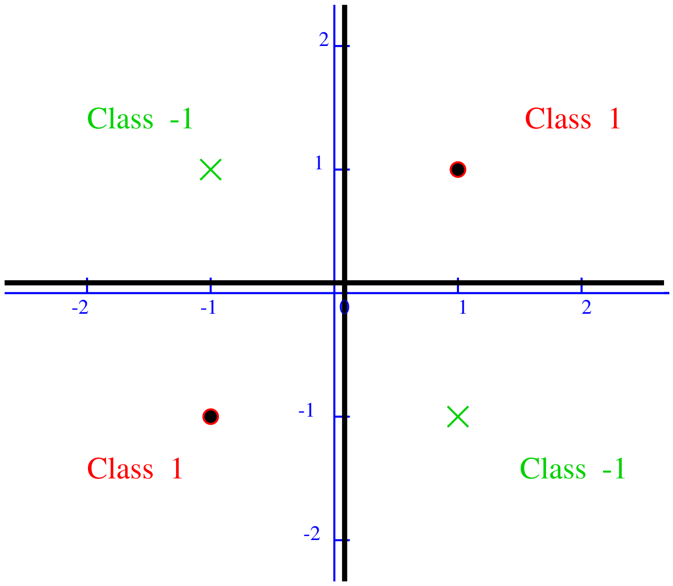

Figure 1: **XOR discrimination by a step function of a quadratic function: $(x_1x_2)_*$ obtained by the linear program (11) and the quadratic programs (9) and (10).**

and attempt to solve the mathematical programs (9),(10), (11) and (12) with this kernel and with $\nu = 1$. The powerful PATH mathematical programming package [11, 10] failed to solve the nonconvex quadratic programming formulation (9) for the conventional support vector machine. In contrast, the same package solved the quadratic program (10) giving $r = \frac{1}{576}e$ and a corresponding $u = DK(A, A')'Dr = -\frac{1}{24}e$. Substitution of this u in the quadratic program (6) and solving the resulting linear program gives: $\gamma = 0,\ y = 0$. The solution $y = 0$ indicates that all points represented by A have been correctly classified, which is corroborated by the resulting decision surface $(x_1x_2)_*$, the same as that of (17). This indicates the effectiveness of the quadratic program (10) in its ability to extract from the negative definite cubic kernel just the required quadratic term to achieve correct separation. Similarly both linear programs (11) and (12) gave $y = 0$ thus also achieving complete separation with this negative definite kernel.

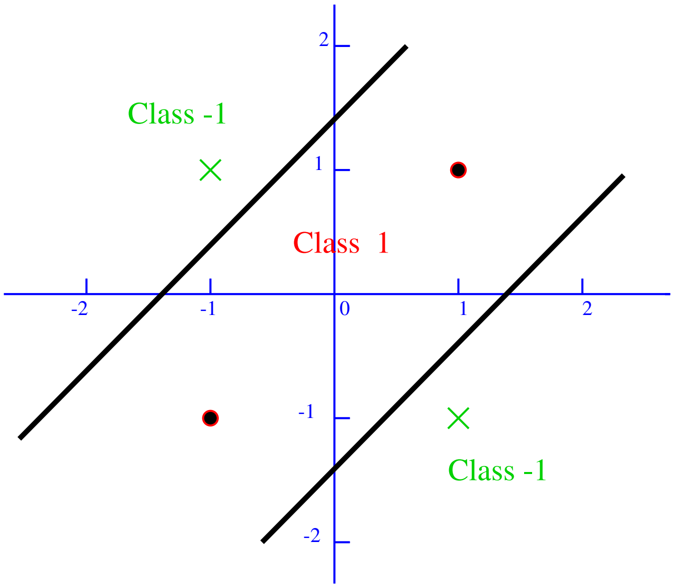

Figure 2: **XOR discrimination by a step function of a quadratic function: $(2-(x_1-x_2)^2)_*$ obtained by another solution of the linear program (11).**

6 Conclusion

We have proposed a direct mathematical programming framework for general support vector machines that makes essentially no or few assumptions on the kernel employed. We have derived new kernel-based linear programming formulations (11) and (12), and a new quadratic programming formulation (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 (9) for a conventional support vector machine 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 support vector machine does not. This leads us to suggest that further testing and experimentation with mathematical programming formulations such as (11), (12) and (10) and others are worthwhile. These formulations may open the way for a variety of of support vector machines that could be tested computationally against each other and against existing ones. Furthermore, broad classes of serial and parallel optimization algorithms can be brought to bear on these different formulations exploiting their structure in order to permit the processing of massive databases.

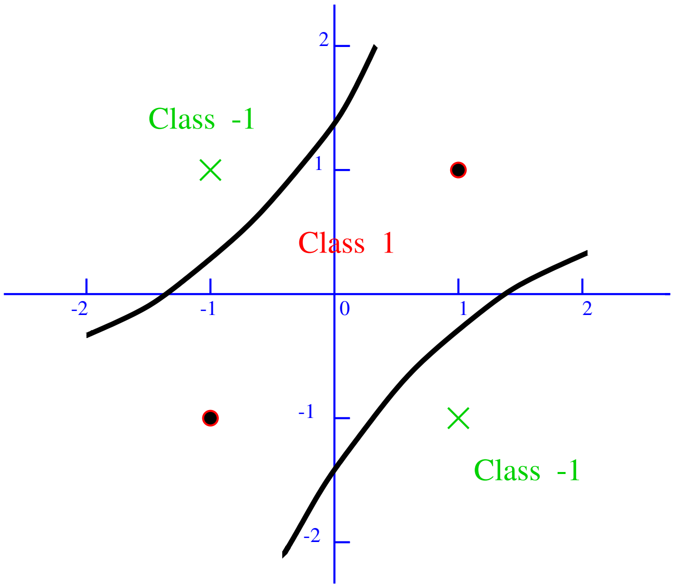

Figure 3: **XOR discrimination by a step function of a quadratic function:** $(2 + x_1x_2 - (x_1 - x_2)^2)_*$ **obtained by the linear program (12).**

Acknowledgements

I am grateful to my colleague Grace Wahba for very helpful discussions on kernel theory, to my PhD student David Musicant and to Paul Bradley for important comments on the paper.

This research is supported by National Science Foundation Grants CCR-9322479, CCR-9729842 and CDA-9623632, and by Air Force Office of Scientific Research Grant F49620-97-1-0326 as Mathematical Programming Technical Report 98-14, October 1998.

References

[1] K. P. Bennett and J. A. Blue. A support vector machine approach to decision trees. Department of Mathematical Sciences Math Report No. 97-100, Rensselaer Polytechnic Institute, Troy, NY 12180, 1997. http://www.math.rpi.edu/~bennek/.

[2] 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/.

[3] P. S. Bradley and O. L. Mangasarian. Feature selection via concave minimization and support vector machines. In J. Shavlik, editor, *Ma-*

chine 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.

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

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

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

[7] V. Cherkassky and F. Mulier. *Learning from Data - Concepts, Theory and Methods.* wiley, New York, 1998.

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

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

[10] S. P. Dirkse and M. C. Ferris. The PATH solver: A non-monotone stabilization scheme for mixed complementarity problems. *Optimization Methods and Software*, 5:123–156, 1995. ftp://ftp.cs.wisc.edu/tech-reports/reports/93/tr1179.ps.

[11] M. C. Ferris and T. S. Munson. Interfaces to PATH 3.0: Design, implementation and usage. *Computational Optimization and Applications, forthcoming*, 1998. ftp: //ftp.cs.wisc.edu/math-prog/tech-reports/97-12.ps.

[12] M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3:95–110, 1956.

[13] F. Girosi. An equivalence between sparse approximation and support vector machines. A.I. Memo 1606, Artificial Intelligence Laboratory, MIT, Cambridge, Massachusetts, 1997. http://www.ai.mit.edu/people/girosi/home-page/svm.html.

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

[15] MATLAB. *User's Guide.* The MathWorks, Inc., 1992.

[16] B. Schölkopf. *Support Vector Learning.* PhD thesis, Technischen Universität Berlin, Berlin, Germany, 1997.

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

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

[19] 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*, Cambridge, MA, 1998. MIT Press. ftp://ftp.stat.wisc.edu/pub/wahba/index.html, Department of Statistics, University of Wisconsin, Madison, Technical Report No. 984rr, July 1998.

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