Support Vector Machines: Hype or Hallelujah?

Kristin P. Bennett
Math Sciences Department
Rensselaer Polytechnic Institute
Troy, NY 12180
bennek@rpi.edu

Colin Campbell
Department of Engineering Mathematics
Bristol University
Bristol BS8 1TR, United Kingdom
C.Campbell@bristol.ac.uk

ABSTRACT

Support Vector Machines (SVMs) and related kernel methods have become increasingly popular tools for data mining tasks such as classification, regression, and novelty detection. The goal of this tutorial is to provide an intuitive explanation of SVMs from a geometric perspective. The classification problem is used to investigate the basic concepts behind SVMs and to examine their strengths and weaknesses from a data mining perspective. While this overview is not comprehensive, it does provide resources for those interested in further exploring SVMs.

Keywords

Support Vector Machines, Kernel Methods, Statistical Learning Theory.

1. INTRODUCTION

Recently there has been an explosion in the number of research papers on the topic of Support Vector Machines (SVMs). SVMs have been successfully applied to a number of applications ranging from particle identification, face identification, and text categorization to engine knock detection, bioinformatics, and database marketing. The approach is systematic, reproducible, and properly motivated by statistical learning theory. Training involves optimization of a convex cost function: there are no false local minima to complicate the learning process. SVMs are the most well-known of a class of algorithms that use the idea of kernel substitution and which we will broadly refer to as *kernel methods*. The general SVM and kernel methodology appears to be well-suited for data mining tasks.

In this tutorial, we motivate the primary concepts behind the SVM approach by examining geometrically the problem of classification. The approach produces elegant mathematical models that are both geometrically intuitive and theoretically well-founded. Existing and new special-purpose optimization algorithms can be used to efficiently construct optimal model solutions. We illustrate the flexibility and generality of the approach by examining extensions of the technique to classification via linear programming, regression and novelty detection. This tutorial is not exhaustive and many approaches (e.g. kernel PCA[56], density estimation [67], etc) have not been considered. Users interested in actually using SVMs should consult more thorough treatments such as the books by Cristianini and Shawe-Taylor [14], Vapnik's books on statistical learning theory [65][66] and recent edited volumes [50] [56]. Readers should consult these and web resources (e.g. [14][69]) for more comprehensive and current treatment of this methodology. We conclude this tutorial with a general discussion of the benefits and shortcomings of SVMs for data mining problems.

To understand the power and elegance of the SVM approach, one must grasp three key ideas: *margins*, *duality*, and *kernels*. We examine these concepts for the case of simple linear classification and then show how they can be extended to more complex tasks. A more mathematically rigorous treatment of the geometric arguments of this paper can be found in [3][12].

2. LINEAR DISCRIMINANTS

Let us consider a binary classification task with datapoints x_i ($i=1,\ldots,m$) having corresponding labels $y_i=\pm 1$. Each datapoint is represented in a d dimensional input or attribute space. Let the classification function be: $f(x)=sign(w\cdot x-b)$. The vector w determines the orientation of a discriminant plane. The scalar b determines the offset of the plane from the origin. Let us begin by assuming that the two sets are linearly separable, i.e. there exists a plane that correctly classifies all the points in the two sets. There are infinitely many possible separating planes that correctly classify the training data. Figure 1 illustrates two different separating planes. Which one is preferable? Intuitively one prefers the solid plane since small perturbations of any point would not introduce misclassification errors. Without any additional information, the solid plane is more likely to generalize better on future data. Geometrically we can characterize the solid plane as being "furthest" from both classes.

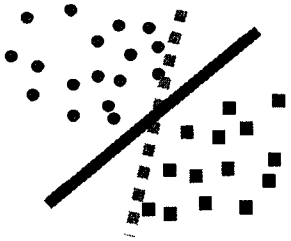


Figure 1 - Two possible linear discriminant planes

How can we construct the plane "furthest'' from both classes? Figure 2 illustrates one approach. We can examine the *convex hull* of each class' training data (indicated by dotted lines in Figure 2) and then find the closest points in the two convex hulls (circles labeled d and c). The convex hull of a set of points is the smallest convex set containing the points. If we construct the plane that bisects these two points ($w=d-c$), the resulting classifier should be robust in some sense.

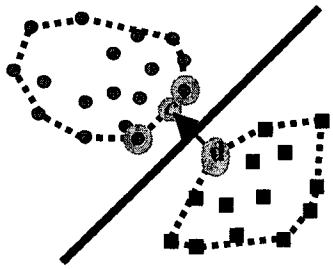


Figure 2 – Best plane bisects closest points in the convex hulls

The closest points in the two convex hulls can be found by solving the following quadratic problem.

$$\begin{aligned} \min_{\alpha} \quad & \tfrac{1}{2}\|c-d\|^2 \\ & c=\sum_{y_i\in Class1}\alpha_i x_i \quad d=\sum_{y_i\in Class-1}\alpha_i x_i \\ s.t. \quad & \sum_{y_i\in Class1}\alpha_i=1 \qquad \sum_{y_i\in Class-1}\alpha_i=1 \\ & \alpha_i\geq 0 \qquad i=1,\ldots,m \end{aligned} \tag{1}$$

There are many existing algorithms for solving general-purpose quadratic problems and also new approaches for exploiting the special structure of SVM problems (See Section 7). Notice that the solution depends only on the three boldly circled points.

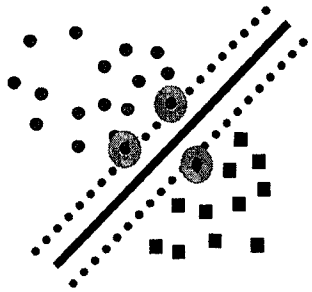


Figure 3 - Best plane maximizes the margin

An alternative approach is to maximize the margin between two parallel supporting planes. A plane supports a class if all points in that class are on one side of that plane. For the points with the class label +1 we would like there to exist w and b such that $w\cdot x_i > b$ or $w\cdot x_i - b > 0$ depending on the class label. Let us suppose the smallest value of $|w\cdot x_i - b|$ is κ, then $w\cdot x_i - b \geq \kappa$. The argument inside the decision function is invariant under a positive rescaling so we will implicitly fix a scale by requiring $w\cdot x_i - b \geq 1$. For the points with the class label -1 we similarly require $w\cdot x_i - b \leq -1$. To find the plane furthest from both sets, we can simply maximize the distance or **margin** between the support planes for each class as illustrated in Figure 3. The support planes are "pushed" apart until they "bump" into a small number of data points (the **support vectors**) from each class. The support vectors in Figure 3 are outlined in bold circles.

The distance or **margin** between these supporting planes $w\cdot x = b+1$ and $w\cdot x = b-1$ is $\gamma = 2/\|w\|_2$. Thus maximizing the margin is equivalent to minimizing $\|w\|_2/2$ in the following quadratic program:

$$\begin{aligned} \min_{w,b} \quad & \tfrac{1}{2}\|w\|^2 \\ s.t \quad & w\cdot x_i \geq b+1 \quad y_i \in Class\,1 \\ & w\cdot x_i \leq b-1 \quad y_i \in Class\,-1 \end{aligned} \tag{2}$$

The constraints can be simplified to $y_i(w\cdot x_i - b)\geq 1$.

Note that the solution found by "maximizing the margin between parallel supporting planes" method (Figure 3) is identical to that found by "bisecting the closest points in the convex hull method" (Figure 2). In the maximum margin method, the supporting planes are pushed apart until they bump into the **support vectors** (boldly circled points), and the solution only depends on these support vectors. In Figure 2, these same support vectors determine the closest points in the convex hull. It is no coincidence that the solutions are identical. This is a wonderful example of the mathematical programming concept of **duality**. The Lagrangian dual of the supporting plane QP (2) yields the following dual QP (see [66] for derivation):

$$\begin{aligned} \min_{\alpha} \quad & \tfrac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} y_i y_j \alpha_i \alpha_j x_i \cdot x_j - \sum_{i=1}^{m}\alpha_i \\ s.t. \quad & \sum_{i=1}^{m} y_i \alpha_i = 0 \\ & \alpha_i \geq 0 \quad i=1,..,m \end{aligned} \tag{3}$$

which is equivalent modulo scaling to the closest points in the convex hull QP (1) [3]. We can choose to solve either the primal QP (2) or the dual QP (1) or (3). They all yield the same normal to the plane $w=\sum_{i=1}^{m} y_i \alpha_i x_i$ and threshold b determined by the support vectors (for which $\alpha_i > 0$).

Thus we can choose to solve either the primal supporting plane QP problem (2) or dual convex hull QP problem (1)or (3) to give the same solution. From a mathematical programming perspective, these are relatively straightforward problems from a well-studied class of convex quadratic programs. There are many effective robust algorithms for solving such QP tasks. Since the QP problems are convex, any local minimum found can be identified as the global minimum. In practice, the dual formulations (2) (3) are preferable since they have very simple constraints and they more readily admit extensions to nonlinear discriminants using kernels as discussed in later sections.

3. THEORETICAL FOUNDATIONS

From a statistical learning theory perspective these QP formulations are well-founded. Roughly, statistical learning proves that bounds on the generalization error on future points not in the training set can be obtained. These bounds are a function of the misclassification error on the training data and terms that measure the complexity or capacity of the classification function. For linear functions, maximizing the margin of separation as discussed above reduces the function capacity or complexity. Thus by explicitly maximizing the margin we are minimizing bounds on the generalization error and can expect better generalization with high probability. The size of the margin is not

directly dependent on the dimensionality of the data. Thus we can expect good performance even for very high-dimensional data (i.e., with a very large number of attributes). In a sense, problems caused by overfitting of high-dimensional data are greatly reduced. The reader is referred to the large volume of literature on this topic, e.g. [14][65][66], for more technical discussions of statistical learning theory.

We can gain insight into these results using geometric arguments. Classification functions that have more capacity to fit the training data are more likely to overfit resulting in poor generalization. Figures 4 and 5 illustrate how a linear discriminant that separates two classes with a small margin has more capacity to fit the data than one with a large margin. In Figure 4, a "skinny" plane can take many possible orientations and still strictly separate all the data. In Figure 5, the "fat" plane has limited flexibility to separate the data. In some sense a fat margin is less complex than a skinny one. So the complexity or capacity of a linear discriminant is a function of the margin of separation. Usually we think of complexity of a linear function as being determined by the number of variables. But if the margin is fat, then the complexity of a function can be low even if the number of variables is very high. Maximizing the margin regulates the complexity of the model.

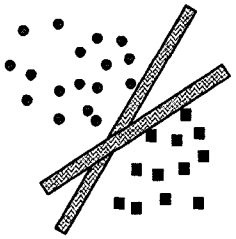


Figure 4 - Many possible "skinny" margin planes

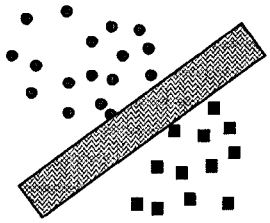


Figure 5 - Few possible "fat" margin planes

4. LINEARLY INSEPARABLE CASE

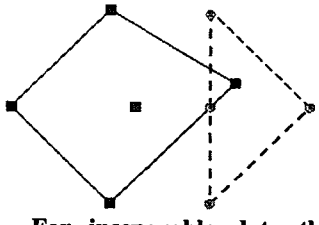


Figure 6 – For inseparable data the convex hulls intersect

So far we have assumed that the two datasets are linearly separable. If this is not true, the strategy of constructing the plane that bisects the two closest points of the convex hulls will fail. As illustrated in Figure 6, if the points are not linearly separable than the convex hulls will intersect. Note that if the single bad square is removed, then our strategy would work again. Thus we need to restrict the influence of any single point. This can be accomplished by using the *reduced* convex hulls instead of the usual definition of convex hulls [3]. The influence of each point is restricted by introducing an upper bound D<1 on the multiplier for that point. Formally the reduced convex hull is defined as:

$$d = \sum_{y_i \in Class1} \alpha_i x_i$$
$$\sum_{y_i \in Class1} \alpha_i = 1 \qquad (4)$$
$$0 \le \alpha_i \le D$$

For D sufficiently small the reduced convex hulls will not intersect. Figure 7 shows the reduced convex hulls (for *D=1/2*) and the separating plane constructed by bisecting the closest points in the two *reduced* convex hulls. The reduced convex hulls for each set are indicated by dotted lines.

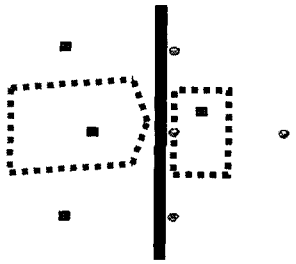


Figure 7 - Best plane bisects the reduced convex hulls

To find the two closest points in the convex hulls we modify the quadratic program for the separable case by adding an upper bound D on multiplier for each constraint to yield:

$$\min_{\alpha} \ \tfrac{1}{2} \| \sum_{y_i \in 1} \alpha_i x_i - \sum_{y_i \in -1} \alpha_i x_i \|^2$$
$$s.t. \quad \sum_{y_i \in 1} \alpha_i = 1 \quad \sum_{y_i \in -1} \alpha_i = 1 \qquad (5)$$
$$0 \le \alpha_i \le D$$

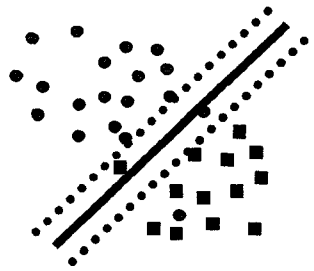


Figure 8 - Select plane to maximize margin and minimize error

For the linearly inseparable case, the primal supporting plane method will also fail. Since the QP task (2) is not feasible for the linearly inseparable case, the constraints must be relaxed. Consider the linearly inseparable problem shown in Figure 8. Ideally we would like no points to be misclassified and no points to fall in the margin. But we must relax the constraints that insure that each point is on the appropriate side of its supporting plane. Any point falling on the wrong side of its supporting plane is

considered to be an error. We want to simultaneously maximize the margin and minimize the error.

This can also be accomplished through minor changes in the supporting plane QP problem (2). A nonnegative slack or error variable z_i is added to each constraint and then added as a weighted penalty term in the objective as follows:

$$\begin{aligned} \min_{w,b,z} \quad & \tfrac{1}{2}\|w\|^2 + C\sum_{i=1}^{\ell} z_i \\ s.t \quad & y_i(w\cdot x_i - b) + z_i \geq 1 \\ & z_i \geq 0 \;\; i = 1,..,m \end{aligned} \tag{6}$$

Once again we can show that the primal relaxed supporting plane method is equivalent to the dual problem of finding the closest points in the reduced convex hulls. The Lagrangian dual of the QP task (6) is:

$$\begin{aligned} \min_{\alpha} \quad & \tfrac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} y_i y_j \alpha_i \alpha_j x_i \cdot x_j - \sum_{i=1}^{m} \alpha_i \\ s.t. \quad & \sum_{i=1}^{m} y_i \alpha_i = 0 \\ & C \geq \alpha_i \geq 0 \quad i = 1,..,m \end{aligned} \tag{7}$$

See [11][66] for the formal derivation of this dual. This is the most commonly used SVM formulation for classification. Note that the only difference between this QP (7) and that for the separable case QP (3) is the addition of the upper bounds on α_i. Like the upper bounds in the reduced convex hull QP (5), these bounds limit the influence of any particular data point. Analogous to the linearly separable case, the geometric problem of finding the closest points in the reduced convex hulls QP (5) has been shown to be equivalent to the QP task in (7) modulo scaling of α_i and D by the size of the optimal margin [3][12].

Up to this point we have examined linear discrimination for the linearly separable and inseparable cases. The basic principle of SVM is to construct the maximum margin separating plane. This is equivalent to the dual problem of finding the two closest points in the (reduced) convex hulls for each class. By using this approach to control complexity, SVMs can construct linear classification functions with good theoretical and practical generalization properties even in very high-dimensional attribute spaces. Robust and efficient quadratic programming methods exist for solving the dual formulations. But if the linear discriminants are not appropriate for the data set, resulting in high training set errors, SVM methods will not perform well. In the next section, we examine how the SVM approach has been generalized to construct highly nonlinear classification functions.

5. NONLINEAR FUNCTIONS VIA KERNELS

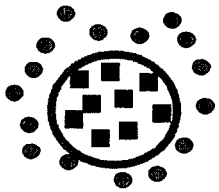


Figure 9 - Example requiring a quadratic discriminant

Consider the classification problem in Figure 9. No simple linear discriminant function will work well. A quadratic function such as the circle pictured is needed. A classic method for converting a linear classification algorithm into a nonlinear classification algorithm is to simply add additional attributes to the data that are nonlinear functions of the original data. Existing linear classification algorithms can be then applied to the expanded dataset in feature space producing nonlinear functions in the original input space. To construct a quadratic discriminant in a two dimensional vector space with attributes r and s, simply map the original two dimensional input space $[r,s]$ to the five dimensional feature space $[r,s,rs,r^2,s^2]$ and construct a linear discriminant in that space. Specifically, define: $\theta(x): R^2 \rightarrow R^5$ then

$$x = [r,s]$$
$$w\cdot x = w_1 r + w_2 s$$
$$\downarrow$$
$$\theta(x) = [r,s,rs,r^2,s^2]$$
$$w\cdot\theta(x) = w_1 r + w_2 s + w_3 rs + w_4 r^2 + w_5 s^2$$

The resulting classification function,

$$\begin{aligned} f(x) &= sign(w\cdot\theta(x) - b) \\ &= sign(w_1 r + w_2 s + w_3 rs + w_4 r^2 + w_5 s - b), \end{aligned}$$

is linear in the mapped five-dimensional feature space but it is quadratic in the two-dimensional input space.

For high-dimensional datasets, this nonlinear mapping method has two potential problems stemming from the fact that dimensionality of the feature space explodes exponentially. The first problem is that overfitting becomes a problem. SVMs are largely immune to this problem since they rely on margin maximization, provided an appropriate value of parameter C is chosen. The second concern is that it is not practical to actually compute $\theta(x)$. SVMs get around this issue through the use of **kernels**.

Examine what happens when the nonlinear mapping is introduced into QP (7). Let us define: $\theta(x): R^n \rightarrow R^{n'} \quad n' >> n$ We need to optimize:

$$\begin{aligned} \min_{\alpha} \quad & \tfrac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j \theta(x_i)\cdot\theta(x_j) - \sum_{i=1}^{l} \alpha_i \\ s.t. \quad & \sum_{i=1}^{l} y_i \alpha_i = 0 \\ & C \geq \alpha_i \geq 0 \quad i = 1,..,m \end{aligned} \tag{8}$$

Notice that the mapped data only occurs as an inner product in the objective. Now we apply a little mathematically rigorous magic known as Hilbert-Schmidt Kernels, first applied to SVMs in [11]. By Mercer's Theorem, we know that for certain mappings θ and any two points u and v, the inner product of the mapped points can be evaluated using the kernel function without ever explicitly knowing the mapping, e.g. $\theta(u)\cdot\theta(v) \equiv K(u,v)$. Some of the

more popular known kernels are given below. New kernels are being developed to fit domain specific requirements.

$\theta(u)$	$K(u,v)$
Degree d polynomial	$(u \cdot v+1)^d$
Radial Basis Function Machine	$\exp\left(-\frac{\|u-v\|^2}{2\sigma}\right)$
Two-Layer Neural Network	$sigmoid(\eta(u \cdot v)+c)$

Table 1- Examples of Kernel Functions

Substituting the kernel into the Dual SVM yields:

$$\min_{\alpha} \quad \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} y_i y_j \alpha_i \alpha_j K(x_i,x_j) - \sum_{i=1}^{m}\alpha_i$$

$$s.t. \quad \sum_{i=1}^{l} y_i \alpha_i = 0 \qquad (9)$$

$$C \geq \alpha_i \geq 0 \quad i=1,..,m$$

To change from a linear to nonlinear classifier, one must only substitute a kernel evaluation in the objective instead of the original dot product. Thus by changing kernels we can get different highly nonlinear classifiers. No algorithmic changes are required from the linear case other than substitution of a kernel evaluation for the simple dot product. All the benefits of the original linear SVM method are maintained. We can train a highly nonlinear classification function such as a polynomial or a radial basis function machine, or a sigmoidal neural network using robust, efficient algorithms that have no problems with local minima. By using kernel substitution a linear algorithm (only capable of handling separable data) can be turned into a general nonlinear algorithm.

6. SUMMARY OF SVM METHOD

The resulting SVM method (in its most popular form) can be summarized as follows

1. Select the parameter C representing the tradeoff between minimizing the training set error and maximizing the margin. Select the kernel function and any kernel parameters. For example for the radial basis function kernel, one must select the width of the gaussian σ.
2. Solve Dual QP (9) or an alternative SVM formulation using an appropriate quadratic programming or linear programming algorithm.
3. Recover the primal threshold variable b using the support vectors
4. Classify a new point x as follows:

$$f(x) = sign(\sum_i y_i \alpha_i K(x,x_i) - b) \qquad (10)$$

Typically the parameters in Step 1 are selected using cross-validation if sufficient data are available. However, recent model selection strategies can give a reasonable estimate for the kernel parameter without use of additional validation data [13][10]. As an example, we consider a recent scheme proposed by Joachims [30]. In this approach the number of leave-one-out errors of an SVM is bounded by $|\{i:(2\alpha_i B^2+z_i) \geq 1\}|/m$ where α_i are the solutions of the optimization task in (9) and B^2 is an upper bound on $K(x_i,x_i)$ with $K(x_i,x_j) \geq 0$ (we can determine z_i from $y_i(\sum_j \alpha_j K(x_j,x_i)-b) \geq 1-z_i$). Thus, for a given value of the kernel parameter, the leave-one-out error is estimated from this quantity (the system is *not* retrained with datapoints left out: the bound is determined using the α_i^0 from the solution of (9)). The kernel parameter is then incremented or decremented in the direction needed to lower this bound. Model selection approaches such as this scheme are becoming increasingly accurate in predicting the best choice of kernel parameter without the need for validation data.

This basic SVM approach has now been extended with many variations and has been applied to many different types of inference problems. Different mathematical programming models are produced but they typically require the solution of a linear or quadratic programming problem. The choice of algorithm used to solve the linear or quadratic program is not critical for the quality of the solution. Modulo numeric differences, any appropriate optimization algorithm will produce an optimal solution, though the computational cost of obtaining the solution is dependent on the specific optimization utilized, of course. Thus we will briefly discuss available QP and LP solvers in the next section.

7. ALGORITHMIC APPROACHES

Typically an SVM approach requires the solution of a QP or LP problem. LP and QP type problems have been extensively studied in the field of mathematical programming. One advantage of SVM methods is that this prior optimization research can be immediately exploited. Existing general-purpose QP algorithms such as quasi-Newton methods and primal-dual interior-point methods can successfully solve problems of small size (thousands of points). Existing LP solvers based on simplex or interior points can handle of problems of moderate size (ten to hundreds of thousands of data points). These algorithms are not suitable when the original data matrix (for linear methods) or the kernel matrix needed for nonlinear methods no longer fits in main memory. For larger datasets alternative techniques have to be used. These can be divided into three categories: techniques in which kernel components are evaluated and discarded during learning, *decomposition* methods in which an evolving subset of data is used, and new optimization approaches that specifically exploit the structure of the SVM problem.

For the first category the most obvious approach is to sequentially update the α_i and this is the approach used by the Kernel Adatron (KA) algorithm [23]. For some variants of SVM models, this method is very easy to implement and can give a quick impression of the performance of SVMs on classification tasks. It is equivalent to Hildreth's method in optimization theory. However, it is not as fast as most QP routines, especially on small datasets. In general, such methods have linear convergence rates and thus may require many scans of the data.

Chunking and decomposition methods optimize the SVM with respect to subsets. Rather than sequentially updating the α_i the alternative is to update the α_i in parallel but using only a subset or *working set* of data at each stage. In *chunking* [41], some QP

optimization algorithm is used to optimize the dual QP on an initial arbitrary subset of data. The support vectors found are retained and all other datapoints (with α_i=0) discarded. A new working set of data is then derived from these support vectors and additional datapoints that maximally violate the storage constraints. This *chunking* process is then iterated until the margin is maximized. Of course, this procedure may still fail because the dataset is too large or the hypothesis modeling the data is not sparse (most of the α_i are non-zero, say). In this case *decomposition* methods provide a better approach: these algorithms only use a fixed-size subset of data called the working set with the remainder kept fixed. A much smaller QP or LP is solved for each working set. Thus many small subproblems are solved instead of one massive one. There are many successful codes based on these decomposition strategies. SVM codes available online such as SVMTorch [15] and SVMLight [32] use these working set strategies. The LP variants are particularly interesting. The fastest LP methods decompose the problem by rows and columns and have been used to solve the largest reported nonlinear SVM regression problems with up to to sixteen thousand points with a kernel matrix of over a billion elements [6][36].

The limiting case of decomposition is the Sequential Minimal Optimization (SMO) algorithm of Platt [43] in which only two α_i are optimized at each iteration. The smallest set of parameters that can be optimized with each iteration is plainly two if the constraint $\sum_{i=1}^{m} \alpha_i y_i = 0$ is to hold. Remarkably, if only two parameters are optimized and the rest kept fixed then it is possible to derive an analytical solution that can be executed using few numerical operations. This eliminates the need for a QP solver for the subproblem. The method therefore consists of a heuristic step for finding the best pair of parameters to optimize and use of an analytic expression to ensure the dual objective function increases monotonically. SMO and improved versions [33] have proven to be an effective approach for large problems.

The third approach is to directly attack the SVM problem from an optimization perspective and create algorithms that explicitly exploit the structure of the problem. Frequently these involve reformulations of the base SVM problem that have proven to be just as effective as the original SVM in practice. Keerthi et al [34] proposed a very effective algorithm based on the dual geometry of finding the two closest points in the convex hulls such as discussed in Section 2. These approaches have been particularly effective for linear SVM problems. We give some examples of recent developments for massive Linear SVM problems. The Lagrangian SVM (LSVM) method reformulates the classification problem as an unconstrainted optimization problem and then solves the problem using an algorithm requiring only solution of systems of linear equalities. Using an eleven line Matlab code, LSVM solves linear classification problems for millions of points in minutes on a Pentium III [37]. LSVM uses a method based on the Sherman-Morrison-Woodbury formula that requires only the solution of systems of linear equalities. This technique has been used to solve linear SVMs with up to 2 million points. The interior-point [22] and Semi-Smooth Support Vector Methods [21] of Ferris and Munson are out-of-core algorithms that have been used to solve linear classification problems with up to 60 million data points in 34 dimensions. Overall, rapid progress is being made in the scalability of SVM approaches. The best algorithms for optimization of SVM objective functions remains an active research subject.

8. SVM EXTENSIONS

One of the major advantages of the SVM approach is its flexibility. Using the basic concepts of maximizing margins, duality, and kernels, the paradigm can be adapted to many types of inference problems. We illustrate this flexibility with three examples. The first illustrates that by simply changing the norm used for regularization, i.e., how the margin is measured, we can produce a linear program (LP) model for classification. The second example shows how the technique has been adapted to do the unsupervised learning task of novelty detection. The third example shows how SVMs have been adapted to do regression. These are just three of the many variations and extensions of the SVM approach to inference problems in data mining and machine learning.

8.1 LP Approaches to Classification.

A common strategy for developing new SVM methods with desirable properties is to adjust the error and margin metrics used in the mathematical programming formulation. Rather than using quadratic programming it is also possible to derive a kernel classifier in which the learning task involves *linear programming* (LP) instead. Recall that the primal SVM formulation (6) maximizes the margin between the supporting planes for each class where the distance is measured by the 2-norm. The resulting QP does this by minimizing the error and minimizing the 2-norm of w. If the model is changed to maximize the margin as measured by the infinity norm, one minimizes the error and minimizes the 1-norm of w (the sum of the absolute values of the components of *w*), e.g.,

$$\begin{aligned} \min_{w,b,z} \quad & \|w\|_1 + C\sum_{i=1}^{\ell} z_i \\ s.t \quad & y_i(x_i \cdot w - b) + z_i \geq 1 \\ & z_i \geq 0 \;\; i = 1,..,m \end{aligned} \tag{11}$$

This problem is easily converted into a LP problem solvable by simplex or interior point algorithms. Since the 1-norm of *w* is minimized the optimal *w* will be very sparse. Many attributes will be dropped since they receive no weight in the optimal solution. Thus this formulation automatically performs feature selection and had been used in that capacity [4].

To create nonlinear discriminants the problem is formulated directly in the kernel or feature space. Recall that in the original SVM formulation the final classification was done as follows: $f(x) = sign(\sum_i y_i \alpha_i K(x, x_i) - b)$. We now directly substitute this function into LP (11) to yield:

$$\begin{aligned} \min_{\alpha,b,z} \quad & \|\alpha\|_1 + C\sum_{i=1}^{\ell} z_i \\ s.t \quad & y_i\left(\sum_{j=1}^{m} y_j \alpha_j K(x_i, x_j) - b\right) + z_i \geq 1 \\ & z_i \geq 0 \; \alpha_i \geq 0 \; i = 1,..,m \end{aligned} \tag{12}$$

By minimizing $\|\alpha\|_1 = \sum_{i=1}^{m} \alpha_i$ we obtain a solution which is *sparse,* i.e. relatively few datapoints will be support vectors. Furthermore, efficient simplex and interior point methods exist for solving linear programming problems so this is a practical alternative to conventional QP. This linear programming approach evolved independently of the QP approach to SVMs and, as we will see, linear programming approaches to regression and novelty detection are also possible.

8.2 Novelty Detection

For many real-world problems the task is not to classify but to detect novel or abnormal instances. Novelty or abnormality detection has potential applications in many problem domains such as condition monitoring or medical diagnosis. One approach is to model the *support* of a data distribution (rather than having to find a real-valued function for estimating the density of the data itself). Thus, at its simplest level, the objective is to create a binary-valued function that is positive in those regions of input space where the data predominantly lies and negative elsewhere.

One approach is to find a hypersphere with a minimal radius R and center **a** which contains most of the data: novel test points lie outside the boundary of this hypersphere. The technique we now outline was originally suggested by Tax and Duin [62][63] and used by these authors for real life applications. The effect of outliers is reduced by using slack variables z to allow for datapoints outside the sphere. The task is to minimize the volume of the sphere and the distance of the datapoints outside, i.e.

$$\begin{aligned} \min_{R,z,a} \quad & R^2 + \frac{1}{mv}\sum_{i=1}^{m} z_i \\ s.t. \quad & (x_i - a)^T (x_i - a) \le R^2 + z_i \\ & z_i \ge 0 \;\; i = 1,..,m \end{aligned} \tag{13}$$

Using the same methodology as explained above for SVM classification, the dual Lagrangian is formed and kernel functions are substituted to produce the following dual QP task for novelty detection:

$$\begin{aligned} \min_{\alpha} \quad & -\sum_{i=1}^{m} \alpha_i K(x_i, x_i) + \sum_{i,j=1}^{m} \alpha_i \alpha_j K(x_i, x_j) \\ & \sum_{i=1}^{m} \alpha_i = 1 \\ s.t. \quad & \frac{1}{mv} \ge \alpha_i \ge 0 \;\; i = 1,..,m \end{aligned} \tag{14}$$

If mv > 1 then *at bound* examples will occur with $\alpha_i = 1/mv$ and these correspond to outliers in the training process. Having completed the training process a test point **v** is declared novel if:

$$K(v,v) - 2\sum_{i=1}^{m} \alpha_i K(v, x_i) + \sum_{i,j=1}^{m} \alpha_i \alpha_j K(x_i, x_j) - R^2 \ge 0$$

where R^2 is first computed by finding an example which is *non-bound* and setting this inequality to an equality.

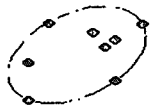
An alternative approach has been developed by Schölkopf et al [51]. Suppose we restricted our attention to RBF kernels: in this case the data lie in a region on the surface of a hypersphere in feature space since $\theta(x)\cdot\theta(x) = K(x,x) = 1$. The objective is therefore to separate off this region from the surface region containing no data. This is achieved by constructing a hyperplane which is maximally distant from the origin with all datapoints lying on the opposite side from the origin, such that $w \cdot x_i - b \ge 0$. After kernel substitution the dual formulation of the learning task involves minimization of:

$$\begin{aligned} \min_{\alpha} \quad & \sum_{i,j=1}^{m} \alpha_i \alpha_j K(x_i, x_j) \\ & \sum_{i=1}^{m} \alpha_i = 1 \\ s.t. \quad & \frac{1}{mv} \ge \alpha_i \ge 0 \;\; i = 1,..,m \end{aligned} \tag{15}$$

To determine b we find an example, k say, which is non-bound (α_i and β_i are nonzero and $0 < \alpha_i < 1/mv$) and determine b from: $b = \sum_{j=1}^{m} \alpha_j K(x_j, x_k)$. The support of the distribution is then modeled by the decision function:

$$f(z) = sign\left(\sum_{j=1}^{m} \alpha_j K(x_j, v) - b\right) \tag{16}$$

In the above models the parameter v has a neat interpretation as an upper bound on the fraction of outliers and a lower bound of the fraction of patterns that are support vectors. Schölkopf et al. [51] provide good experimental evidence in favor of this approach including the highlighting of abnormal digits in the USPS handwritten character dataset.



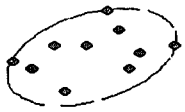


Figure 10 - Novelty detection using (17): points outside the boundary are viewed as novel.

For the model of Schölkopf et al. the origin of feature space plays a special role. It effectively acts as a prior for where the class of abnormal instances is assumed to lie. Rather than repelling away from the origin we could consider attracting the hyperplane onto datapoints in feature space. In input space this corresponds to a surface that wraps around the data clusters (Figure 10) and can be achieved through the following linear programming task [9]:

$$\min_{w,b,z} \quad \sum_{i=1}^{m}\left(\sum_{j=1}^{m}\alpha_j K(x_i,x_j)-b\right)+\lambda\sum_{i=1}^{t}z_i$$
$$s.t \quad \left(\sum_{j=1}^{m}\alpha_j K(x_i,x_j)-b\right)+z_i \geq 1 \qquad (17)$$
$$z_i \geq 0\ \alpha_i \geq 0\ i=1,..,m$$

The parameter b is just treated as an additional parameter in the minimization process, though unrestricted in sign. Noise and outliers are handled by introducing a soft boundary with error z. This method has been successfully used for detection of abnormalities in blood samples and detection of faults in the condition monitoring of ball-bearing cages [9].

8.3 Regression

SVM approaches for real-valued outputs have also been formulated and theoretically motivated from statistical learning theory [66]. SVM regression uses the ε–*insensitive* loss function shown in Figure 11. If the deviation between the actual and predicted value is less than ε, then the regression function is not considered to be in error. Thus mathematically we would like $-\varepsilon \leq w\cdot x_i - b - y_i \leq \varepsilon$. Geometrically, we can visualize this as a band or tube of size 2ε around the hypothesis function f(x) and any points outside this tube can be viewed as training errors (see Figure 12).

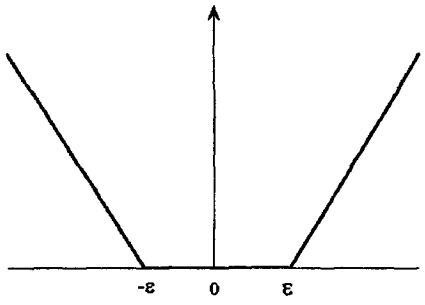


Figure 11- A piecewise linear ε-insensitive loss function

As before we minimize $\|w\|$ to penalize overcomplexity. To account for training errors we also introduce slack variables z and $\hat{z}_i$ for the two types of training error. The first computes the error for underestimating the function. The second computes the error for overestimating the function. These slack variables are zero for points inside the tube and progressively increase for points outside the tube according to the loss function used. This general approach is called ε-SV regression and is the most common approach to SV regression. For a *linear ε–insensitive loss function* the task is therefore to optimize:

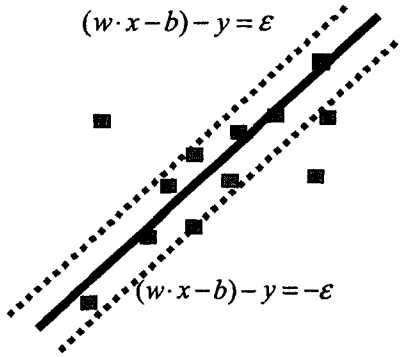


Figure 12 - Plot of wx-b versus y with ε-insensitive tube. Points outside of the tube are errors.

$$\min_{w,b,z,\hat{z}} \quad C\sum_{i=1}^{t}\left(z_i+\hat{z}_i\right)+\frac{1}{2}\|w\|^2$$
$$s.t \quad \left(w\cdot x_i-b-y_i\right)+z_i \geq \varepsilon \qquad (18)$$
$$\left(w\cdot x_i-b-y_i\right)-\hat{z}_i \leq -\varepsilon$$
$$z_i,\hat{z}_i \geq 0\ \ i=1,..,m$$

The same strategy of computing the Lagrangian dual and adding kernels functions is then used to construct nonlinear regression functions.

Apart from the formulations given here it is possible to define other loss functions giving rise to different dual objective functions. In addition, rather than specifying ε *a priori* it is possible to specify an upper bound ν ($0 \leq \nu \leq 1$) on the fraction of points lying outside the band and then find ε by optimizing [48][49]. As for classification and novelty detection it is possible to formulate a linear programming approach to regression e.g.:

$$\min_{\alpha,\hat{\alpha},b,z,\hat{z}} \quad \sum_{i=1}^{m}\alpha_i+\sum_{i=1}^{m}\hat{\alpha}_i+C\sum_{i=1}^{m}z_i+C\sum_{i=1}^{m}\hat{z}_i$$
$$y_i-\varepsilon-\hat{z}_i \leq \left(\sum_{j=1}^{m}(\hat{\alpha}_j-\alpha_j)K(x_i,x_j)-b\right) \qquad (19)$$
$$s.t \qquad \leq y_i+\varepsilon-z_i$$
$$z_i,\hat{z}_i,\alpha_i,\hat{\alpha}_i \geq 0\ i=1,..,m$$

Minimizing the sum of the α_i approximately minimizes the number of support vectors. Thus the method favors sparse functions that smoothly approximate the data.

9. SVM APPLICATIONS

SVMs have been successfully applied to a number of applications ranging from particle identification [1], face detection [40] and text categorization [17][19][29][31] to engine knock detection [46], bioinformatics [7][24][28][71][38] and database marketing [5]. In this section we discuss three successful application areas as illustrations: machine vision, handwritten character recognition,

and bioinformatics. These are rapidly changing research areas so more contemporary accounts are best obtained from relevant websites [27].

9.1 Applications to Machine Vision

SVMs are very suited to the classsification tasks that commonly arise in machine vision. As an example we consider an application involving face identification [20]. This experiment used the standard ORL dataset [39] (consisting of 10 images per person from 40 different persons). Three methods were tried: a direct SVM classifier that learned the original images directly (apart from some local rescaling), a classifier that used more extensive pre-processing involving rescaling, local sampling and local principal component analysis, and an invariant SVM classifier that learned the original images plus a set of images which have been translated and zoomed. For the invariant SVM classifier the training set of 200 images (5 per person) was increased to 1400 translated and zoomed examples and an RBF kernel was used. On the test set these three methods gave generalization errors of 5.5%, 3.7%, and 1.5% respectively. This was compared with a number of alternative techniques with the best result among the latter being 2.7%. Face and gender detection have also been successfully achieved. 3D object recognition [47] is another successful area of application including 3D face recognition, pedestrian recognition [44], etc.

9.2 Handwritten digit recognition

The United States Postal Service (USPS) dataset consists of 9298 handwritten digits each consisting of a 16×16 vector with entries between -1 and 1. An RBF network and a SVM were compared on this dataset. The RBF network had spherical Gaussian RBF nodes with the same number of Gaussian basis functions as there were support vectors for the SVM. The centers and variances for the Gaussians were found using classical k-means clustering. Gaussian kernels were used and the system was trained with a soft margin (with C=10.0). A set of one-against-all classifiers was used since this is a multi-class problem. With a training set of 7291 and test set of 2007, the SVM outperformed an RBF network on all digits [55]. SVMs have also been applied to the much larger NIST dataset of handwritten characters consisting of 60,000 training and 10,000 test images each with 400 pixels. Recently DeCoste and Scholkopf [16] have shown that SVMs outperform all other techniques on this dataset.

9.3 Applications to Bioinformatics: functional interpretation of gene expression data.

The recent development of DNA microarray technology is creating a wealth of gene expression data. In this technology RNA is extracted from cells in sample tissues and reverse transcribed into labeled cDNA. Using fluorescent labels, cDNA binding to DNA probes is then highlighted by laser excitation. The level of expression of a gene is proportional to the amount of cDNA that hybridizes with each DNA probe and hence proportional to the intensity of fluorescent excitation at each site.

As an example of gene expression data we will consider a recent ovarian cancer dataset investigated by Furey et al. [24]. The microarray used had 97,802 DNA probes and 30 tissue samples were used. The task considered was binary classification (ovarian cancer or no cancer). This example is fairly typical for current datsets: it has a very high dimensionality with comparatively few examples. Viewed as a machine learning task the high dimensionality and sparsity of datapoints suggest the use of SVMs since the good generalization ability of SVMs doesn't depend on the dimensionality of the space but on maximizing the margin. Also the high-dimensional feature vector $\mathbf{x}_i$ is absorbed in the kernel matrix for the purposes of computation, thus the learning task follows the reduced dimensionality of the example set size rather than the number of features. By constrast a neural network would need 97,802 input nodes and a correspondingly large number of weights to adjust. A further motivation for considering SVMs comes from the existence of the model selection bounds mentioned in Section 6 which may be exploited to achieve effective feature selection [69] thereby highlighting those genes which have the most significantly different expression levels for cancer.

In the study by Furey et al. [24] three cancer datasets were considered: the ovarian cancer dataset mentioned above, a colon tumor dataset and datasets for acute myeloid leukemia (AML) or acute lymphoblastic leukemia (ALL). For ovarian cancer it was possible to get perfect classification using leave-one-out testing for one choice of the model parameters [24]. For the colon cancer expression levels from 40 tumor and 22 normal colon tissues were determined using a DNA microarray and leave-one-out testing gave six incorrectly labelled tissues.

For the leukemia datasets [24][38] the training set consisted of 38 examples (27 ALL and 11 AML) and the test set consisted of 34 examples (20 ALL and 14 AML). A weighted voting scheme correctly learned 36 of the 38 instances and a self-organizing map gave two clusters: one with 24 ALL and 1 AML and the other with 10 AML and 3 ALL [25]. The SVM correctly learned all the training data. On the test data the weighted voting scheme gave 29 of 34 correct, declining to predict on 5. For the SVM, results varied according to the different configurations that achieved zero training error. 30 to 32 of test instances were correctly labeled except for one choice with 29 correct and the 5 declined by the weighted voting scheme classified incorrectly.

SVMs have been successfully applied to other bioinformatics tasks. In a second successful application they have been used for protein homology detection [28] to determine the structural and functional properties of new protein sequences. Determination of these properties is achieved by relating new sequences to proteins with known structural features. In this application the SVM outperformed a number of established systems for homology detection for relating the test sequence to the correct families. As a third application we also mention the detection of translation initiation sites (the points on nucleotide sequences where regions encoding proteins start). SVMs performed very well on this task using a kernel function specifically designed to include prior biological information [71].

10. DISCUSSION

Support Vector Machines have many appealing features.

1. SVMs are a rare example of a methodology where geometric intutition, elegant mathematics, theoretical guarantees, and practical algorithms meet.
2. SVMs represent a general methodology for many types of problems. We have seen that SVMs can be applied to a wide range of classification, regression, and novelty detection tasks but they can also be applied to other areas we have not

covered such as operator inversion and unsupervised learning. They can be used to generate many possible learning machine architectures (e.g., RBF networks, feedforward neural networks) through an appropriate choice of kernel. The general methodology is very flexible. It can be customized to meet particular application needs. Using the ideas of margin/regularization, duality, and kernels, one can extend the method to meet the needs of a wide variety of data mining tasks.

3. The method eliminates many of the problems experienced with other inference methodologies like neural networks and decision trees.
    a. There are no problems with local minima. We can construct highly nonlinear classification and regression functions without worrying about getting stuck at local minima.
    b. There are few model parameters to pick. For example if one chooses to construct a radial basis function (RBF) machine for classification one need only pick two parameters: the penalty parameter for misclassification and the width of the gaussian kernel. The number of basis functions is automatically selected by the SVM algorithm.
    c. The final results are stable, reproducible, and largely independent of the specific algorithm used to optimize the SVM model. If two users apply the same SVM model with the same parameters to the same data, they will get the same solution modulo numeric issues. Compare this with neural networks where the results are dependent on the particular algorithm and starting point used.
4. Robust optimization algorithms exist for solving SVM models. The problems are formulated as mathematical programming models so state-of-the-art research from that area can be readily applied. Results have been reported in the literature for classification problems with millions of data points.
5. The method is relatively simple to use. One need not be a SVM expert to successfully apply existing SVM software to new problems.
6. There are many successful applications of SVM. They have proven to be robust to noise and perform well on many tasks.

While SVMs are a powerful paradigm, many issues remain to be solved before they become indispensable tools in a data miner's toolbox. Consider the following challenging questions and SVMs progress on them to date.

1. Will SVMs always perform best? Will it beat my best hand-tuned method on a particular dataset? Though one can always anticipate the existence of datasets for which SVMs will perform worse than alternative techniques, this does not exclude the possibility that they perform best on the average or outperform other techniques across a range of important applications. As we have seen in the last section, SVMs do indeed perform best for some important application domains. But SVMs are no panacea. They still require skill to apply them and other methods may be better suited for particular applications.
2. Do SVMs scale to massive datasets? The computational costs of an SVM approach depends on the optimization algorithm being used. The very best algorithms to date are typically quadratic and involved multiple scans of the data. But these algorithms are constantly being improved. The latest linear classification algorithms report results for 60 million data points. So progress is being made.
3. Do SVMs eliminate the model selection problem? Within the SVM method one must still select the attributes to be included in the problems, the type of kernel (including its parameters), and model parameters that trade-off the error and capacity control. Currently, the most commonly used method for picking these parameters is still cross-validation. Cross-validation can be quite expensive. But as discussed in Section 6 researchers are exploiting the underlying SVM mathematical formulations and the associated statistical learning theory to develop efficient model selection criteria. Eventually model selection will probably become one of the strengths of the approach.
4. How does one incorporate domain knowledge into SVM?
   Right now the only way to incorporate domain knowledge is through the preparation of the data and choice/design of kernels. The implicit mapping into a higher dimensional feature space makes use of prior knowledge difficult. An interesting question is how well will SVM perform against alternative algorithmic approaches that can exploit prior knowledge about the problem domain.
5. How interpretable are the results produced by a SVM? Interpretability has not been a priority to date in SVM research. The support vectors found by the algorithms provide limited information. Further research into producing interpretable results with confidence measures is needed.
6. What format must the data be in to use SVMs? What is the effect of attribute scaling? How does one handle categorical variables and missing data? Like neural networks, SVMs were primarily developed to apply to real-valued vectors. So typically data is converted to real-vectors and scaled. Different methods for doing this conversion can affect the outcome of the algorithm. Usually categorical variables are mapped to numeric values. The problem of missing data has not been explicitly addressed within the methodology so one must depend on existing preprocessing techniques. There is however potential for SVMs to handle these issues better. For example, new types of kernels could be developed to explicitly handle data with graphical structure and missing values.

Though these and other questions remain open at the current time, progress in the last few years has resulted in many new insights and we can expect SVMs to grow in importance as a data mining tool.

11. ACKNOWLEDGMENTS

This work was performed with the support of the National Science Foundation under grants 970923 and IIS-9979860.

12. REFERENCES

[1] Barabino N., Pallavicini M., Petrolini A., Pontil M. and Verri A. Support vector machines vs multi-layer perceptrons in particle identification. In *Proceedings of the European Symposium on Artifical Neural Networks '99* (D-Facto Press, Belgium), p. 257-262, 1999.

[2] Bennett K. and Bredensteiner E. Geometry in Learning, in *Geometry at Work*, C. Gorini Editor, Mathematical Association of America, Washington D.C., 132-145, 2000.

[3] Bennett K. and Bredensteiner E. Duality and Geometry in SVMs. In P. Langley editor, *Proc. of 17th International Conference on Machine Learning*, Morgan Kaufmann, San Francisco, 65-72, 2000

[4] Bennett K., Demiriz A. and Shawe-Taylor J. A Column Generation Algorithm for Boosting. In P. Langley editor, *Proc. of 17th International Conference on Machine Learning*, Morgan Kaufmann, San Francisco, 57-64, 2000.

[5] Bennett K., Wu D. and Auslender L. On support vector decision trees for database marketing. Research Report No. 98-100, Rensselaer Polytechnic Institute, Troy, NY, 1998.

[6] Bradley P., Mangasarian O. and Musicant, D. Optimization in Massive Datasets. To appear in Abello, J., Pardalos P., Resende, M (eds) , *Handbook of Massive Datasets*, Kluwer, 2000.

[7] Brown M., Grundy W., D. Lin, N. Cristianini, C. Sugnet, T. Furey, M. Ares Jr. D. Haussler. Knowledge-based Analysis of Microarray Gene Expression Data using Support Vector Machines. *Proceedings of the National Academy of Sciences*, 97 (1), p. 262-267, 2000.

[8] Burges C. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2, p. 121-167, 1998.

[9] Campbell C. and Bennett K. A Linear Programming Approach to Novelty Detection. To appear in *Advances in Neural Information Processing Systems* 14 (Morgan Kaufmann, 2001).

[10] Chapelle O. and Vapnik V. Model selection for support vector machines. To appear in *Advances in Neural Information Processing Systems*, 12, ed. S.A. Solla, T.K. Leen and K.-R. Muller, MIT Press, 2000.

[11] Cortes C. and Vapnik V. Support vector networks. *Machine Learning* 20, p. 273-297, 1995.

[12] Crisp D. and Burges C. A geometric interpretation of v-svm classifiers. *Advances in Neural Information Processing Systems*, 12, ed. S.A. Solla, T.K. Leen and K.-R. Muller, MIT Press, 2000.

[13] Cristianini N., Campbell C. and Shawe-Taylor, J. Dynamically adapting kernels in support vector machines. *Advances in Neural Information Processing Systems*, 11, ed. M. Kearns, S. A. Solla, and D. Cohn, MIT Press, p. 204-210, 1999.

[14] Cristianini N. and Shawe-Taylor J. *An Introduction to Support Vector Machines and other Kernel-based Learning Methods*. Cambridge University Press, 2000. www.support-vector.net.

[15] Collobert R. and Bengio S. SVMTorch web page, http://www.idiap.ch/learning/SVMTorch.html

[16] DeCoste D. and Scholkopf B. Training Invariant Support Vector Machines. To appear in *Machine Learning* (Kluwer, 2001).

[17] Drucker H., with Wu D. and Vapnik V. Support vector machines for spam categorization. *IEEE Trans. on Neural Networks*, 10, p. 1048-1054. 1999.

[18] Drucker H., Burges C., Kaufman L., Smola A. and Vapnik V. Support vector regression machines. In: M. Mozer, M. Jordan, and T. Petsche (eds.). *Advances in Neural Information Processing Systems*, 9, MIT Press, Cambridge, MA, 1997.

[19] Dumais S., Platt J., Heckerman D. and Sahami M. Inductive Learning Algorithms and Representations for Text Categorization. *7th International Conference on Information and Knowledge Management*, 1998.

[20] Fernandez R. and Viennet E. Face identification using support vector machines. *Proceedings of the European Symposium on Artificial Neural Networks* (ESANN99), (D.-Facto Press, Brussels) p.195-200, 1999

[21] Ferris, M. and Munson T. Semi-smooth support vector machines. Data Mining Institute Technical Report 00-09, Computer Sciences Department, University of Wisconsin, Madison, Wisconsin, 2000.

[22] Ferris M. and Munson T. Interior point methods for massive support vector machines. Data Mining Institute Technical Report 00-05, Computer Sciences Department, University of Wisconsin, Madison, Wisconsin, 2000.

[23] Friess T.-T., Cristianini N. and Campbell, C. The kernel adatron algorithm: a fast and simple learning procedure for support vector machines. *15th Intl. Conf. Machine Learning*, Morgan Kaufman Publishers, p. 188-196, 1998.

[24] Furey T., Cristianini N., Duffy N., Bednarski D., Schummer M. and Haussler D. Support Vector Machine Classification and Validation of Cancer Tissue Samples using Microarray Expression Data. *Bioinformatics* 16 p. 906-914, 2000.

[25] Golub T., Slonim D., Tamayo P., Huard C., Gassenbeek M., Mesirov J., Coller H., Loh M., Downing J., Caligiuri M., Bloomfield C. and Lander E. Modecular Classification of cancer: Class discovery and class prediction by gene expression monitoring. Science, 286 p. 531-537, 1999.

[26] Guyon I., Matic N. and Vapnik V. Discovering informative patterns and data cleaning. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, MIT Press, p. 181--203, 1996.

[27] Guyon, I Web page on SVM Applications, http://www.clopinet.com/isabelle/Projects/SVM/applist.html

[28] Jaakkola T., Diekhans M. and Haussler, D. A discriminative framework for detecting remote protein homologies. MIT Preprint, 1999.

[29] Joachims, T. Text categorization with support vector machines: learning with many relevant features. *Proc. European Conference on Machine Learning (ECML)*, 1998.

[30] Joachims, T. Estimating the Generalization Performance of an SVM efficiently. In *Proceedings of the 17th International Conference on Machine Learning*, Morgan Kaufmann,. 431-438, 2000.

[31] Joachims, T. Text categorization with support vector machines: learning with many relevant features. *Proc. European Conference on Machine Learning (ECML)*, 1998.

[32] Joachims, T. Web Page on SVMLight: http://www-ai.cs.uni-dortmund.de /SOFTWARE/SVM_LIGHT/svm_light.eng.html

[33] Keerthi S., Shevade S., Bhattacharyya C. and Murthy, K. Improvements to Platt's SMO algorithm for SVM classifier design. Tech Report, Dept. of CSA, Banglore, India, 1999.

[34] Keerthi S., Shevade, S., Bhattacharyya C. and Murthy, K. A. Fast Iterative Nearest Point Algorithm for Support Vector Machine Classifier Design, Techical Report TR-ISL-99-03, Intelligent Systems Lab, Dept of Computer Science and Automation, Indian Institute of Science, Bangalore, India, (accepted for publication in *IEEE Transaction on Neural Networks*) 1999.

[35] Luenberger, D. *Linear and Nonlinear Programming*. Addison-Wesley, 1984.

[36] Mangasarian, O. and Musicant D. Massive Support Vector Regression Data mining Institute Technical Report 99-02, Dept of Computer Science, University of Wisconsin-Madison, August 1999.

[37] Mangasarian, O. and Musicant D. Lagrangian Support Vector Regression Data mining Institute Technical Report 00-06, June 2000.

[38] Mukherjee S., Tamayo P., Slonim D., Verri A., Golub T., Mesirov J. and Poggio T. Support Vector Machine Classification of Microarray Data, MIT AI Memo No. 1677 and MIT CBCL Paper No. 182.

[39] ORL dataset: Olivetti Research Laboratory, 1994,. http://www.uk.research.att.com/facedatabase.html

[40] Osuna E., Freund R. and Girosi F. Training Support Vector Machines: an Application to Face Detection. *Proceedings of CVPR'97*, Puerto Rico, 1997

[41] Osuna E., Freund R. and Girosi F. *Proc. of IEEE NNSP*, Amelia Island, FL p. 24-26, 1997.

[42] Osuna E. and Girosi F. Reducing the Run-time Complexity in Support Vector Machines. In B. Scholkopf, C.Burges and A. Smola (ed.), *Advances in Kernel Methods: Support Vector Learning*, MIT press, Cambridge, MA, p. 271-284, 1999.

[43] Platt J. Fast training of SVMs using sequential minimal optimization. In B. Scholkopf, C.Burges and A. Smola (ed.), *Advances in Kernel Methods: Support Vector Learning*, MIT press, Cambridge, MA, p. 185-208, 1999.

[44] Papageorgiou C., Oren M. and Poggio, T. A General Framework for Object Detection. *Proceedings of International Conference on Computer Vision*, p. 555-562, 1998.

[45] Raetsch G., Demiriz A., and Bennett K. Sparse regression ensembles in infinite and finite hypothesis space. NeuroCOLT2 technical report, Royal Holloway College, London, September, 2000.

[46] Rychetsky M., Ortmann, S. and Glesner, M. Support Vector Approaches for Engine Knock Detection. *Proc. International Joint Conference on Neural Networks (IJCNN 99)*, July, 1999, Washington, USA

[47] Roobaert D. Improving the Generalization of Linear Support Vector Machines: an Application to 3D Object Recognition with Cluttered Background. Proc. Workshop on Support Vector Machines at the 16th International Joint Conference on Artificial Intelligence, July 31-August 6, Stockholm, Sweden, p. 29-33 1999.

[48] Scholkopf B., Bartlett P., Smola A. and Williamson R. Support vector regression with automatic accuracy control. In L. Niklasson, M. Boden and T. Ziemke, editors, *Proceedings of the 8th International Conference on Artificial Neural Networks*, Perspectives in Neural Computing, Berlin, Springer Verlag, 1998.

[49] Scholkopf B., Bartlett P., Smola A., and Williamson R. Shrinking the Tube: A New Support Vector Regression Algorithm. To appear in: M. S. Kearns, S. A. Solla, and D. A. Cohn (eds.), *Advances in Neural Information Processing Systems*, 11, MIT Press, Cambridge, MA, 1999.

[50] Scholkopf B., Burges C. and Smola A. *Advances in Kernel Methods: Support Vector Machines*. MIT Press, Cambridge, MA. 1998.

[51] Scholkopf B., Platt J.C., Shawe-Taylor J., Smola A.J., Williamson R.C. Estimating the support of a high-dimensional distribution. Microsoft Research Corporation Technical Report MSR-TR-99-87, 1999.

[52] Scholkopf B., Shawe-Taylor J., Smola A. and Williamson R. Kernel-dependent support vector error bounds. *Ninth International Conference on Artificial Neural Networks*, IEE Conference Publications No. 470, p. 304 - 309, 1999.

[53] Scholkopf B., Smola A., and Muller, K.-R.. Kernel principal component analysis. In B. Scholkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods: Support Vector Learning*. MIT Press, Cambridge, MA, 1999b. 327 -- 352.

[54] Scholkopf B., Smola A., Williamson R., and Bartlett P. New support vector algorithms. To appear in *Neural Computation*, 1999.

[55] Scholkopf, B., Sung, K., Burges C., Girosi F., Niyogi P., Poggio T. and Vapnik V. Comparing Support Vector Machines with Gaussian Kernels to Radial Basis Function Classifiers. *IEEE Transactions on Signal Processing*, 45, p. 2758-2765, 1997.

[56] Smola A., Bartlett P., Scholkopf B. and Schuurmans C. (eds), *Advances in Large Margin Classifiers*, Chapter 2, MIT Press, 1999.

[57] Shawe-Taylor J. and Cristianini N. Margin distribution and soft margin. In A. Smola, P. Barlett, B. Scholkopf and C. Schuurmans (eds), *Advances in Large Margin Classifiers*, Chapter 2, MIT Press, 1999.

[58] Smola A. and Scholkopf B. A tutorial on support vector regression. NeuroColt2 TR 1998-03, 1998.

[59] Smola A. and Scholkopf B. From Regularization Operators to Support Vector Kernels. In: M. Mozer, M. Jordan, and T. Petsche (eds). *Advances in Neural Information Processing Systems*, 9, MIT Press, Cambridge, MA, 1997.

[60] Smola A., Scholkopf B. and Muller K.-R.. The connection between regularisation operators and support vector kernels. *Neural Networks*, 11 p. 637-649, 1998.

[61] Smola A., Williamson R., Mika S., and Scholkopf B. Regularized principal manifolds. In Computational Learning Theory: 4th European Conference, volume 1572 of *Lecture Notes in Artificial Intelligence* (Springer), p. 214-229, 1999.

[62] Tax D. and Duin R. Data domain description by Support Vectors. In *Proceedings of ESANN99*, ed. M Verleysen, D. Facto Press, Brussels, p. 251-256, 1999.

[63] Tax D., Ypma A., and Duin R.. Support vector data description applied to machine vibration analysis. In: M. Boasson, J. Kaandorp, J.Tonino, M. Vosselman (eds.), *Proc. 5th Annual Conference of the Advanced School for Computing and Imaging* (Heijen, NL, June 15-17), 1999, 398-405.

[64] http://www.ics.uci.edu/mlearn/MLRepository.html

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

[66] Vapnik, V. *Statistical Learning Theory*. Wiley, 1998.

[67] Weston, J. Gammerman, A., Stitson, M., Vapnik, V., Vovk, V. and Watkins, C. Support Vector Density Estimation. In B. Scholkopf, C. Burges and A. Smola. Advances in Kernel Methods: Support Vector Machines. MIT Press, cambridge, M.A. p. 293-306, 1999.

[68] Vapnik, V.and Chapelle, O. Bounds on error expectation for Support Vector Machines. Submitted to *Neural Computation*, 1999

[69] Weston J., Mukherjee, Chapelle, Pontil M., Poggio T., and Vapnik V. Feature Selection for SVMs. To appear in *Advances in Neural Information Processing Systems* 14 (Morgan Kaufmann, 2001).

[70] http://kernel-machines.org/

[71] Zien A., Ratsch G., Mika S., Scholkopf B., Lemmen C., Smola A., Lengauer T. and Muller K.-R. Engineering Support Vector Machine Kernels That Recognize Translation Initiation Sites. Presented at the German Conference on Bioinformatics, 1999.

About the authors:

Kristin P Bennett is an associate professor of mathematical sciences at Rensselaer Polytechnic Institute. Her research focus on support vector machines and other mathematical programming based methods for data mining and machine learning and their application to practical problems such as drug discovery, properties of materials, and database marketing. She recently returned from being a visiting researcher at Microsoft Research and has consulted for Chase Manhattan Bank, Kodak and Pfizer. She earned a Ph.D. from the Computer Sciences Department at University of Wisconsin – Madison. (http://www.rpi.edu/~bennek).

Colin Campbell gained a BSc degree in Physics from Imperial College, London and a PhD in Applied Mathematics from the Department of Mathematics, King's College, University of London. He was appointed to the Faculty of Engineering, Bristol University in 1990. His interests include neural computing, machine learning, support vector machines and the application of these techniques to medical decision support, bioinformatics and machine vision. (http://lara.enm.bris.ac.uk/cig).