Machine Learning, 48, 115–136, 2002

Statistical Properties and Adaptive Tuning of Support Vector Machines

YI LIN yilin@stat.wisc.edu (http://stat.wisc.edu/~yilin)
GRACE WAHBA wahba@stat.wisc.edu (http://stat.wisc.edu/~wahba)
HAO ZHANG hzhang@stat.wisc.edu (http://stat.wisc.edu/~hzhang)
YOONKYUNG LEE yklee@stat.wisc.edu (http://stat.wisc.edu/~yklee)
Department of Statistics, University of Wisconsin, Madison, 1210 West Dayton Street, Madison, WI 53706-1685, USA

Editors: Yoshua Bengio and Dale Schuurmans

Abstract. In this paper we consider the statistical aspects of support vector machines (SVMs) in the classification context, and describe an approach to adaptively tuning the smoothing parameter(s) in the SVMs. The relation between the Bayes rule of classification and the SVMs is discussed, shedding light on why the SVMs work well. This relation also reveals that the misclassification rate of the SVMs is closely related to the generalized comparative Kullback-Leibler distance (*GCKL*) proposed in Wahba (1999, Scholkopf, Burges, & Smola (Eds.), Advances in Kernel Methods—Support Vector Learning. Cambridge, MA: MIT Press). The adaptive tuning is based on the generalized approximate cross validation (*GACV*), which is an easily computable proxy of the *GCKL*. The results are generalized to the unbalanced case where the fraction of members of the classes in the training set is different than that in the general population, and the costs of misclassification for the two kinds of errors are different. The main results in this paper have been obtained in several places elsewhere. Here we take the opportunity to organize them in one place and note how they fit together and reinforce one another. Mostly the work of the authors is reviewed.

Keywords: support vector machine, classification, Bayes rule, *GCKL*, *GACV*

1. The binary classification problem and the Bayes rule

Consider a training set of n subjects from a certain probability distribution (the sampling distribution). For each subject $i, i = 1, 2, \dots, n$, in the training data set, we observe an explanatory vector $x_i \in \mathcal{X}$, where $\mathcal{X}$ is an arbitrary index set, and a label y_i indicating one of the two given classes ($\mathcal{A}$ or $\mathcal{B}$) to which the subject belongs. The observations (x_i, y_i) are assumed to be i.i.d. from an (unknown) sampling distribution $P(x, y)$. The task of classification is to derive from the training set a good classification rule, so that once we are given the x value of a new subject from a target population, we can assign a class label to the subject. The distribution of this target population may or may not be the same as the sampling distribution. Most studies in the literature assume that the two distributions are identical, and the cost of misclassifying subject in class $\mathcal{A}$ to class $\mathcal{B}$ is the same as that of misclassifying subject in class $\mathcal{B}$ to class $\mathcal{A}$. We will refer to this case as the standard case. We will consider the standard case first, and then extend the results to the nonstandard case as commonly arises in practice, where the fraction of members of the classes in the sampling distribution may be different than that in the target population, and the costs of misclassification for the two kinds of errors may be different.

If we knew the underlying sampling distribution and the distribution of the target population, we could derive the optimal classification rule with respect to any given loss function. This is the classification rule that minimizes the expected cost, and is usually called the Bayes rule. In practice, however, we do not know the underlying distributions, and need to get the necessary information from the training samples to approximate the Bayes rule.

From now on until the last section, we will concentrate on the standard case. In such situation the expected cost is equivalent to the expected misclassification rate. Let

$$p_0(x) = Pr\{Y = \mathcal{A} \mid X = x\},$$

where (X, Y) is a generic pair of random variables with distribution $P(x, y)$. It is well known that the Bayes rule for minimizing the expected misclassification rate is

$$\phi^*(x) = \begin{cases} \mathcal{A} & \text{if } p_0(x) > 1/2, \\ \mathcal{B} & \text{otherwise.} \end{cases} \tag{1}$$

We can see the conditional probability function $p_0(x)$ completely determines the Bayes rule. It is for this reason many statistical methods try to estimate $p_0(x)$, or equivalently, the log odds ratio defined as

$$f_0(x) = \log\left[\frac{p_0(x)}{1 - p_0(x)}\right].$$

One example of this is the penalized log likelihood estimation method, of which we will give a brief description for purpose of illustration. This method estimates f_0 by minimizing the penalized negative log likelihood. Coding the label of class $\mathcal{A}$ as 1 and the label of class $\mathcal{B}$ as 0, the probability distribution for $Y \mid p$ is

$$p^y(1-p)^{1-y} = \begin{cases} p & \text{if } y = 1, \\ 1 - p & \text{if } y = 0, \end{cases}$$

or equivalently, $\exp[yf - log(1 + e^f)]$, where $f(x) = \log[\frac{p(x)}{1-p(x)}]$. Therefore the negative log likelihood is

$$l_n(f) = \frac{1}{n}\sum_{i=1}^{n}\left[-y_i f(x_i) + \log\left(1 + e^{f(x_i)}\right)\right]$$

In the penalized log likelihood estimation method, we try to find $f(x) = b + h(x)$ with $h \in \mathcal{H}_K$ to minimize

$$l_n(f) + \lambda \|h\|^2_{\mathcal{H}_K}, \tag{2}$$

where $\mathcal{H}_K$ is the reproducing kernel Hilbert space (*RKHS*) with reproducing kernel $K(s, t)$, $s, t \in \mathcal{X}$, $\|\cdot\|_{\mathcal{H}_K}$ is the norm on the *RKHS*, and λ is the smoothing parameter to be chosen.

The penalized log likelihood estimation method is only one example of a class of estimation methods called regularization methods or smoothing spline methods. The regularization methods have been studied extensively in the statistics literature. See Wahba (1990) and the reference therein. Other examples include penalized least square regression, penalized density estimation, and regularization procedures used in more general nonlinear inverse problems. Cox and O'Sullivan (1990) provided a general framework for studying regularization methods. As in (2), the method of regularization always has two components: a data fit functional component and a regularization penalty component. The data fit functional component ensures that the estimate should fit the training data well, whereas the regularization penalty component is used to guard against overfitting. The data fit component usually approaches a limiting functional as $n \to \infty$. The target function that the regularization method is supposed to estimate is the minimizer of this limiting functional. Under various general conditions, the estimate from the regularization method approaches the target function as $n \to \infty$.

For example, for the penalized log likelihood estimation, the limiting functional of the data fit component is

$$l(f) = E\left[-Yf(X) + \log\left(1 + e^{f(X)}\right)\right], \tag{3}$$

and with some simple calculations it is easy to check that f_0 is the minimizer of (3).

2. Support vector machines with reproducing kernel

The support vector machine methodology was introduced in Boser, Guyon, and Vapnik (1992). See also Cortes and Vapnik (1995), Vapnik (1995). Support vector machines have proved highly successful in a number of classification studies. The linear SVMs are motivated by the geometric interpretation of maximizing the margin, and the nonlinear SVMs are characterized by the use of reproducing kernels. (The reproducing kernel is sometimes called kernel in SVM literature, not to be confused with the kernel estimators used in nonparametric statistics). For a tutorial on SVMs for classification, see Burges (1998).

Now code the label of class $\mathcal{A}$ as 1 and the label of class $\mathcal{B}$ as -1. It has been shown that the SVM with reproducing kernel K is equivalent to a regularization problem in the *RKHS* $\mathcal{H}_K$. See Wahba (1999), Poggio and Girosi (1998). The SVM with reproducing kernel K finds the minimizer of

$$\frac{1}{n}\sum_{i=1}^{n}[1 - y_i g(x_i)]_+ + \lambda\|h\|^2_{\mathcal{H}_K} \tag{4}$$

over all functions of the form $g(x) = h(x) + b$, and $h \in \mathcal{H}_K$. Here

$$(\tau)_+ = \begin{cases} \tau & \text{if } \tau > 0, \\ 0 & \text{otherwise.} \end{cases}$$

From this we can see that the SVM methodology is a regularization method with a particular loss function $[1 - yg(x)]_+$ (sometimes called the 'Hinge-loss'). Once the minimizer $\hat{g}$ is found, the SVM classification rule is

$$\hat{g}(x) > 0 \rightarrow \mathcal{A}$$
$$\hat{g}(x) < 0 \rightarrow \mathcal{B}$$

3. Support vector machines and the Bayes rule

In this section we review the relation between the SVM classification rule and the Bayes rule. These ideas have been developed in some detail in Lin (1999). Theoretical details and further numerical results may be found there. The connection between the SVM classification rule and the Bayes rule is of special interest to us, since it will give us insight on how SVMs work, and why SVMs work well. To establish this connection we need to know what SVMs are doing by minimizing (4). The following observation is the key to this question.

Lemma 3.1 (Lin, 1999). *The minimizer of* $E[(1 - Yg(X))_+]$ *is* $g_0(x) = sign[p_0(x) - 1/2]$, *or equivalently*, $sign[f_0(x)]$.

Proof: Notice $E[1 - Yg(X)]_+ = E\{E\{[1 - Yg(X)]_+ \mid X\}\}$. We can minimize $E[1 - Yg(X)]_+$ by minimizing $E\{[1 - Yg(X)]_+ \mid X = x\}$ for every fixed x.

For any fixed x, we have

$$E\{[1 - Yg(X)]_+ \mid X = x\} = p_0(x)[1 - g(x)]_+ + [1 - p_0(x)][1 + g(x)]_+. \tag{5}$$

We can see the minimizing $g(x)$ has to be in $[-1, 1]$, since it is easy to check that for any $g(x)$ outside $[-1, 1]$, the right hand side of (5) will achieve a smaller value with $sign[g(x)]$ than with $g(x)$. So we can restrict our search of the minimizing $g(x)$ in $[-1, 1]$. For $g(x)$ in $[-1, 1]$, (5) becomes $p_0(x)[1 - g(x)] + [1 - p_0(x)][1 + g(x)] = 1 + g(x)[1 - 2p_0(x)]$, and is obviously minimized by $g(x) = 1$ if $1 - 2p_0(x) < 0$ and $g(x) = -1$ otherwise. That is, the minimizing $g(x)$ is $sign[p_0(x) - 1/2]$. □

As a regularization method, the data fit component of the SVM is $\frac{1}{n}\sum_{i=1}^{n}[1 - y_i g(x_i)]_+$, which has a limiting functional $E[(1 - Yg(X))_+]$. Lemma 3.1 gives the minimizer of this limiting functional, thus identifies the target function of the SVM as $sign[p_0(x) - 1/2]$, or equivalently, $sign[f_0(x)]$. Notice this is exactly the Bayes rule. Thus the SVM with kernel tries to implement the Bayes rule by approximating the function $sign[f_0(x)]$ directly. If the RKHS is rich enough to contain functions that approximate $sign[p_0(x) - 1/2]$, the solution to the SVM problem (4) approaches the function $sign[p_0(x) - 1/2]$ as $n \rightarrow \infty$ when the smoothing parameter is chosen appropriately.

A variety of reproducing kernels have been used successfully in practical applications, including polynomial kernels, Gaussian kernels, and Sobolev Hilbert space kernels (spline kernels). The RKHS' for the latter two types of reproducing kernels are of infinite dimension. For a review on the spline kernels, see Wahba (1990). Roughly speaking, the *RKHS* induced by the spline kernel of order m contains all the functions that have mth order derivatives. The requirement that the RKHS is rich enough to contain functions that approximate the sign function is important for our result. Our result only applies to rich RKHS's such as those induced by spline kernels, Gaussian kernels and high order polynomial kernels. For example, our result does not cover the linear support vector machines.

Lemma 3.1 also has important implication on the tuning of the smoothing parameter. For a general function $g(x)$, consider the classification rule

$$g(x) > 0 \to 1;$$
$$g(x) < 0 \to -1.$$

The expected misclassification rate of this classifier can be expressed as $E[-Yg(X)]_*$, where

$$(\tau)_* = \begin{cases} 1 & \text{if } \tau > 0, \\ 0 & \text{otherwise.} \end{cases}$$

Following Wahba, Lin, and Zhang (1999), we will call the quantity $E[(1-Yg(X))_+]$ the generalized comparative Kullback-Leibler distance (*GCKL*). It is easy to check that *GCKL* is an upper bound of the expected misclassification rate $E[-Yg(X)]_*$ for any function g. For functions that only take values in $\{-1, 1\}$, we can see the *GCKL* is exactly 2 times the expected misclassification rate. We will consider choosing the smoothing parameter with the *GCKL*. This can be motivated by the following consideration. In SVMs, when the data size is large and the smoothing parameter λ is in the neighborhood of the optimal level, the solution $\hat{g}(x)$ to (4) is close to $g_0(x) = sign[f_0(x)]$ which only takes value in $\{-1, 1\}$, therefore the *GCKL* of $\hat{g}(x)$ is approximately 2 times the expected misclassification rate. Thus in the neighborhood, choosing λ in the SVM to minimize the *GCKL* is approximately the same as choosing λ to minimize the expected misclassification rate. This approximation gets better as the size of the training sample grows. Since the function $(\cdot)_*$ is discontinuous and not convex, whereas $(\cdot)_+$ is continuous and convex, it is advantageous to choose λ with *GCKL* rather than with the expected misclassification rate.

In the following we review the computation of the SVM solution, and use a simulation study to illustrate the results in this section. In the simulation GCKL is used to choose λ. Notice the GCKL depends on the unknown underlying probability distribution of the population, and is not directly computable. In the simulation in this section, we pretend that we know the underlying probability distribution when we compute GCKL. This is just to illustrate our result that the solution of the SVM approaches $sign[p_0(x) - 1/2]$ when the smoothing parameter is chosen as the minimizer of GCKL. In practice, we need to find a proxy of the GCKL based on the training data. We will postpone the discussion of this proxy to the next section.

For a chosen reproducing kernel K, by the theory of *RKHS* (see Wahba, 1990), the minimizer of (4) must have the form

$$g(\cdot) = \sum_{i=1}^{n} c_i K(\cdot, x_i) + b.$$

Letting $e = (1, \dots, 1)'$, $y = (y_1, y_2, \dots, y_n)'$, $c = (c_1, c_2, \dots, c_n)'$, and with some abuse of notation, letting $g = (g(x_1), g(x_2), \dots, g(x_n))'$ and K now be the $n \times n$ matrix with ijth entry $K(x_i, x_j)$, we have

$$g = Kc + eb, \tag{6}$$

and the regularization problem (4) becomes: find (c, b) to minimize

$$\frac{1}{n}\sum_{i=1}^{n}[1 - y_i g(x_i)]_+ + \lambda c'Kc,$$

or equivalently, find (c, b) to minimize

$$\frac{1}{n}\sum_{i=1}^{n}\xi_i + \lambda c'Kc,$$

subject to constraints:

$$y_i g(x_i) \geq 1 - \xi_i, \forall i; \quad \xi_i \geq 0, \forall i.$$

This is exactly the standard setup of the SVM with reproducing kernel. A standard way of solving this problem is to consider its dual problem. Let Y be the $n \times n$ diagonal matrix with y_i in the iith position, and let $H = \frac{1}{2n\lambda}YKY$. The dual problem has the form

$$\max L = -\frac{1}{2}\alpha' H\alpha + e'\alpha \tag{7}$$

subject to

$$0 \leq \alpha_i \leq 1, \ \forall i = 1, 2, \dots, n, \quad \text{and} \quad y'\alpha = 0. \tag{8}$$

Here $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_n)'$. Once we get the α's, we get c's by

$$c = \frac{1}{2n\lambda}Y\alpha, \tag{9}$$

and b can be computed robustly by

$$b = [e'A(I - A)(y - Kc)]/[\alpha'(e - \alpha)]. \tag{10}$$

as long as there exists an i for which $0 < \alpha_i < 1$. Here A is the $n \times n$ diagonal matrix with α_i in the iith position.

For easy visualization, we conducted the simulation in one dimension. The simulation has been reported in Lin (1999), and more numerical results can be found there. We took n equidistant points on the interval [0, 1]. That is, $x_i = i/(n - 1)$, $i = 0, 1, \dots, n - 1$. Let $p_0(x) = Pr(Y = 1 \mid X = x) = 1 - |1 - 2x|$, and randomly generate y_i to be 1 or -1 with probability $p_0(x_i)$ and $1 - p_0(x_i)$. It is easy to see that $sign[p_0(x) - 1/2] = 1$, if $x \in (0.25, 0.75)$, and -1, otherwise. We concentrate on the SVM with spline kernels here.

The spline kernel of order 2 on [0, 1] that we used was

$$K(s,t) = 1 + k_1(s)k_1(t) + k_2(s)k_2(t) - k_4(|s-t|),$$

where $k_1(\cdot) = \cdot - 0.5$, $k_2 = (k_1^2 - 1/12)/2$, and $k_4 = (k_1^4 - k_1^2/2 + 7/240)/24$.

We choose the smoothing parameter with *GCKL*. In our simulation here, we can calculate *GCKL* directly for any g, since we know what $p_0(x)$ is. In reality, we do not know $p_0(x)$, hence we can not calculate the *GCKL* directly, but we can estimate the *GCKL* with a tuning data set, or develop a proxy of the *GCKL* based on the training data. See Section 4.

We ran the simulation for $n = 33, 65, 129, 257$. In each case the smoothing parameter was chosen so that the *GCKL* for $\hat{g}_\lambda$ was minimized. The result is shown in figure 1. To illustrate how the smoothing parameter influences the solution, we give the solutions to (4) in the case $n = 257$ with smoothing parameters λ such that $n\lambda = 2^{-j}$, $j = 1, 2, \ldots, 25$. The results are shown in figures 2 and 3. Figure 2 shows how the SVM solutions change with the smoothing parameter, and that for a range of choices of λ, the solutions are close to $sign[p_0(x) - 1/2]$. We can see in figure 3 that the minimizer of GCKL coincides with a local minimum point of the expected misclassification rate. It is our experience that this local

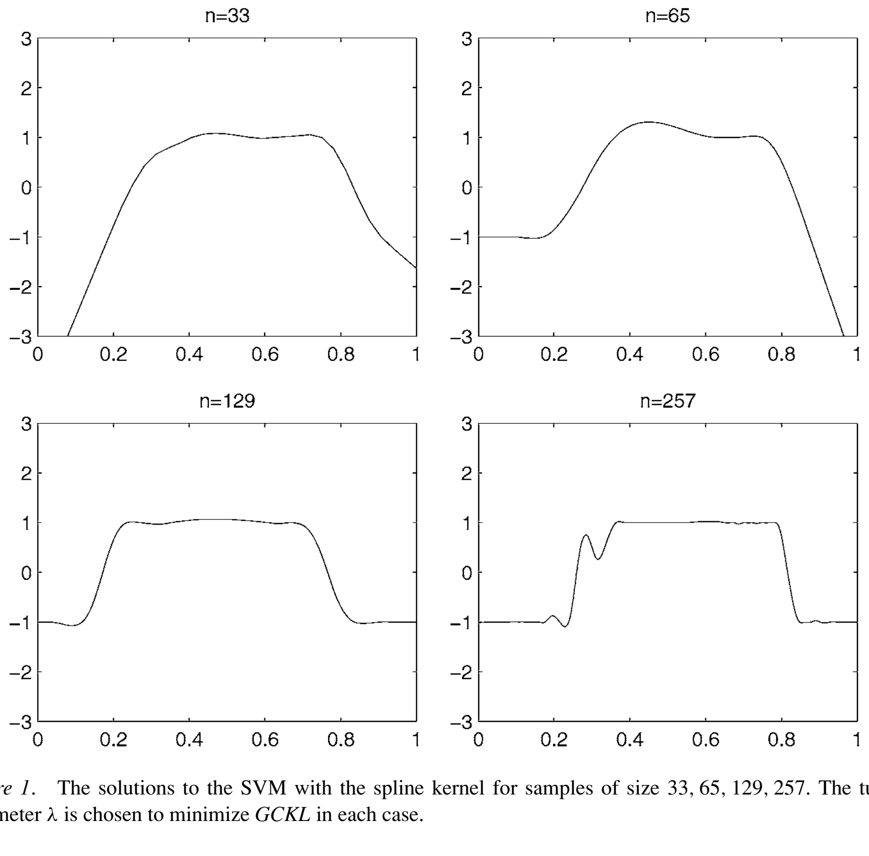

Figure 1. The solutions to the SVM with the spline kernel for samples of size 33, 65, 129, 257. The tuning parameter λ is chosen to minimize *GCKL* in each case.

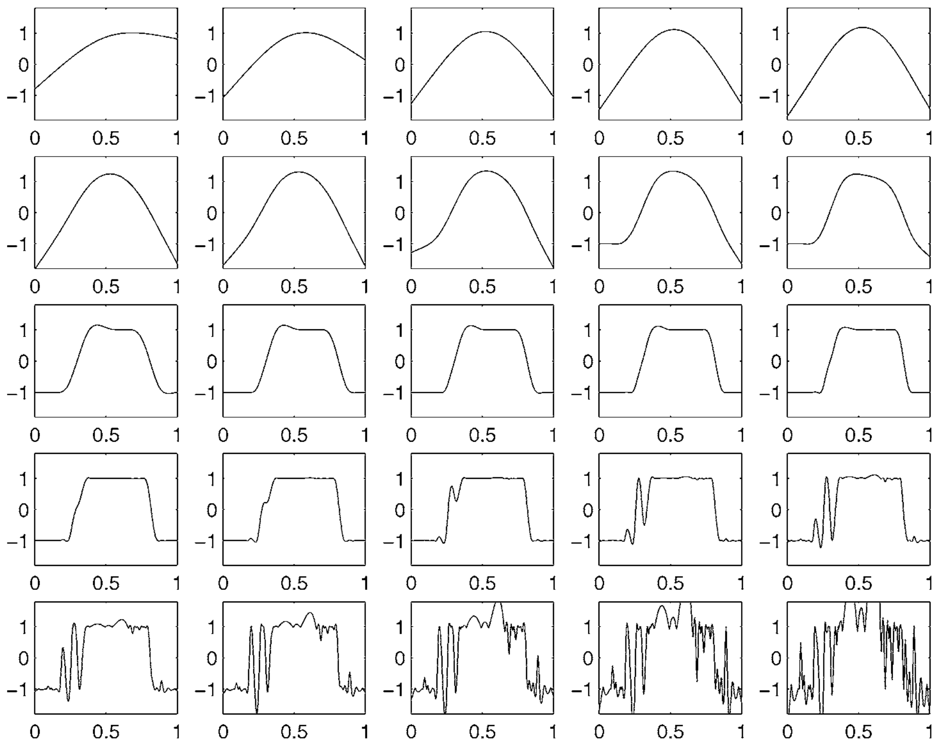

Figure 2. For a fixed sample with $n = 257$, the solutions to the SVM with the spline kernel for $n\lambda = 2^{-1}$, $2^{-2}, \ldots, 2^{-25}$.

minimum of the expected misclassification rate may or may not be the global minimum, but the value of the local minimum is close to the value of the global minimum.

4. Adaptive tuning of the smoothing parameter(s)

The choice of the smoothing parameter(s) is a problem of great practical importance. We develop a proxy of the unobservable *GCKL* by using the leaving-out-one cross-validation. The ideas originally come from Wahba, Lin, and Zhang (1999) and more details can be found there. Let $g_\lambda^{[-k]}$ be the minimizer of

$$\frac{1}{n}\sum_{\substack{i=1\\ i\neq k}}^{n}[1 - y_i g(x_i)]_+ + \lambda\|h\|^2_{\mathcal{H}_K} \tag{11}$$

over all the functions of the form $g(x) = h(x) + b$, and $h \in \mathcal{H}_K$. Then the leaving-out-one function $V_0(\lambda)$ is defined as

$$V_0(\lambda) = \frac{1}{n}\sum_{k=1}^{n}\left[1 - y_k g_\lambda^{[-k]}(x_k)\right]_+.$$

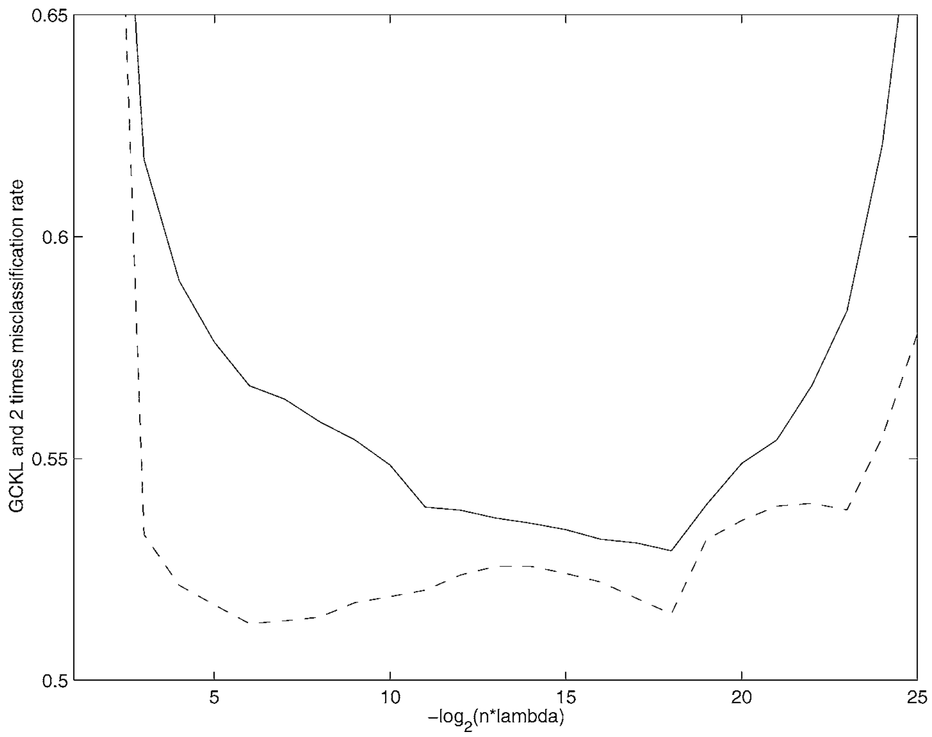

Figure 3. *GCKL* (solid line) and two times misclassification rate (dashed line) of $\hat{g}_\lambda$ with varying λ for the same sample as in figure 2, where $\hat{g}_\lambda$ is the solution to the SVM with spline kernel. Notice the x-axis is $-\log_2(n\lambda)$. (Larger values of λ correspond to the points on the left.)

In theory this quantity can be computed from the training data, but it is impractical to compute this quantity directly, since it requires n implementations of the SVM algorithm for each λ. Therefore, we need to make some approximation that is easier to compute.

Define

$$OBS(\lambda) = \frac{1}{n}\sum_{k=1}^{n}[1 - y_k\hat{g}_\lambda(x_k)]_+,$$

then $OBS(\lambda)$ is easily computable. Let $D(\lambda) = V_0(\lambda) - OBS(\lambda)$. It can be shown (see Wahba, Lin, & Zhang, 1999) that

$$D(\lambda) \approx \frac{1}{n}\left[\sum_{y_i\hat{g}_\lambda(x_i)<-1} 2\frac{\partial\hat{g}_\lambda(x_i)}{\partial y_i} + \sum_{y_i\hat{g}_\lambda(x_i)\in[-1,1]} \frac{\partial\hat{g}_\lambda(x_i)}{\partial y_i}\right]. \tag{12}$$

Here y_i is treated as a continuous variable.

We need a further approximation for $\frac{\partial g_\lambda(x_i)}{\partial y_i}$. Equations (6) and (9) motivate the use of $\frac{\alpha_i}{2n\lambda}K(x_i, x_i)$ as a quick interpretation of this quantity. By doing so we are ignoring the dependence between α_i and y_i.

To summarize, we will approximate $D(\lambda)$ with the easily computable quantity

$$\hat{D}(\lambda) = \frac{1}{n}\left[2\sum_{y_i\hat{g}_\lambda(x_i)<-1}\frac{\alpha_i}{2n\lambda}K(x_i, x_i) + \sum_{y_i\hat{g}_\lambda(x_i)\in[-1,1]}\frac{\alpha_i}{2n\lambda}K(x_i, x_i)\right].$$

Define the generalized approximate cross validation as

$$GACV(\lambda) = OBS(\lambda) + \hat{D}(\lambda),$$

then *GACV* is a proxy for the *GCKL*.

We give two simple examples to illustrate the property of *GACV*. We use a Gaussian kernel in both examples. The Gaussian kernel is

$$K(s, t) = \exp\left[-\frac{\|s - t\|^2}{2\sigma^2}\right],$$

where σ is a smoothing parameter that needs to be chosen jointly with λ. See also Wahba, Lin, and Zhang (1999).

Example 1. The explanatory variable x is two dimensional, and is generated uniformly in the square $[-1, 1] \times [-1, 1]$ as depicted in figure 4. The points outside the larger circle were randomly assigned to be $+1$ with probability 0.95, and -1 with probability 0.05. The points between the outer and inner circles were assigned $+1$ with probability 0.5, and the points inside the inner circle were assigned $+1$ with probability 0.05. Figure 5 plots $\log_{10}(GACV)$ and $\log_{10}(GCKL)$ as a function of $\log_{10}(\lambda)$, with $\log_{10}(\sigma)$ fixed at -1. Figure 6 gives the corresponding plot as a function of $\log_{10}(\sigma)$ with $\log_{10}(\lambda)$ fixed at -2.5, which was the minimizer of $\log_{10}(GACV)$ in figure 5. Figure 7 shows the level curve $\hat{g} = 0$ for $\log_{10}(\sigma) = -1$ and $\log_{10}(\lambda) = -2.5$, which was the minimizer of $\log_{10}(GACV)$ over the two plots. This can be compared to the theoretically optimal classifier boundary, which is any curve between the two circles.

Example 2. The explanatory variable x is two dimensional, and is generated uniformly in the square $[-1, 1] \times [-1, 1]$ as depicted in figure 8, with $p_0 = 0.95$, 0.5, and 0.05 respectively in the three regions, starting from the top. Figures 9 and 10 are analogous to figures 5 and 6. Figure 11 shows the level curve for $\log_{10}(\sigma) = -1.25$ and $\log_{10}(\lambda) = -2.5$, which was the minimizer of $\log_{10}(GACV)$. This can again be compared to the theoretically optimal classifier, which would be any curve falling between the two sine waves of figure 8.

It can be seen in these two examples that the minimizer of $\log_{10}(GACV)$ is a good estimate of the minimizer of $\log_{10}(GCKL)$.

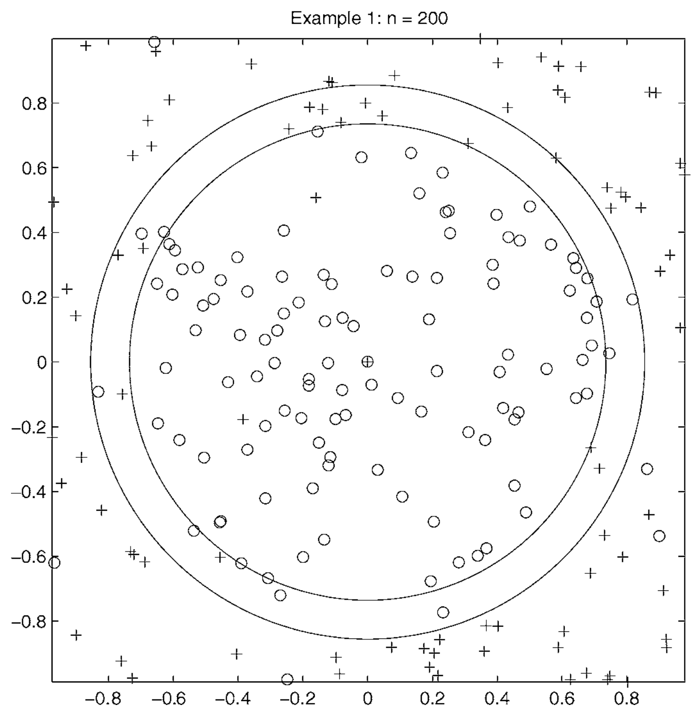

Figure 4. Data for Example 1, with regions of constant (generating) probability.

5. Extension to nonstandard case

Nonstandard situations arise in practice frequently, and are receiving more and more attention. Again code class $\mathcal{A}$ as 1 and class $\mathcal{B}$ as -1. The most common nonstandard situations are:

1. Different types of misclassification may have different costs. One type of misclassification is often more serious than another. For example misclassifying a subject from class $\mathcal{A}$ to class $\mathcal{B}$ (false negative) might be more costly than misclassifying a class $\mathcal{A}$ subject to class $\mathcal{B}$ (false positive). This should be taken into account when constructing the classification rules. In particular, the expected cost of future misclassification, rather than the expected misclassification rate, should be used to measure the performance of the classifier.
2. The target population may not have the same distribution as the sampling distribution because of the so called "sampling bias". For example, in some situations, the smaller class in the target population is over-sampled and the bigger class down-sized in an attempt to have a more "balanced" sample. In this situation, a certain proportion of the examples are randomly selected from class $\mathcal{A}$, and the rest of the examples are randomly selected from class $\mathcal{B}$. These proportions do not reflect the actual proportions of class $\mathcal{A}$ and class $\mathcal{B}$ subjects in the target population.

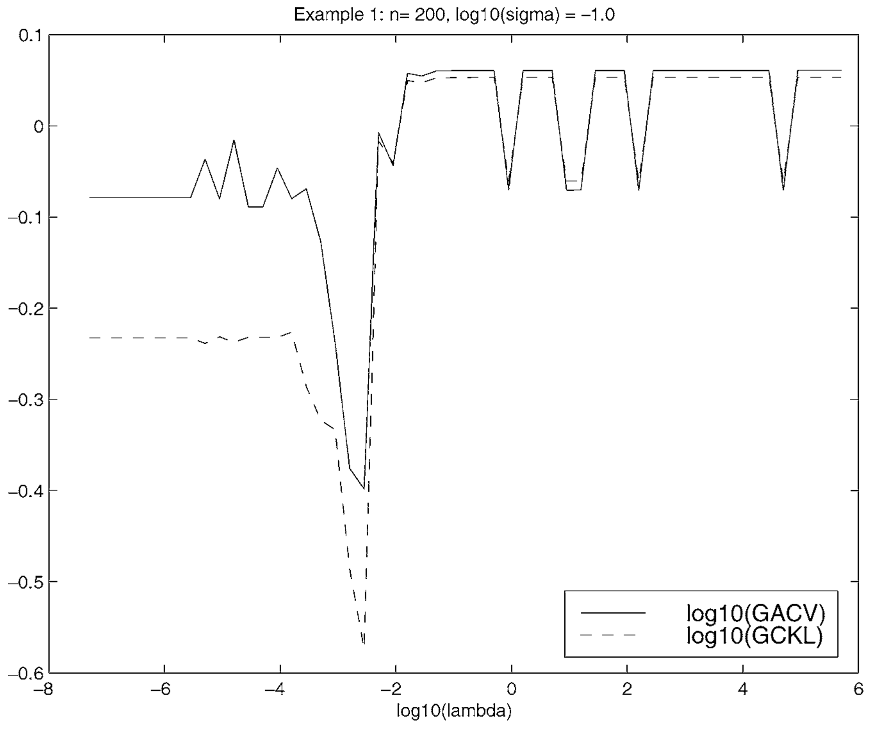

Figure 5. Plot of $\log_{10}GACV$ and $\log_{10}GCKL$ as a function of $\log_{10}\lambda$ for $\log_{10}\sigma = -1.0$.

Hand (1997) gives a nice explanation for these issues. Brown et al. (2000) considered a nonstandard situation. However, they dealt with the problem in a different way than ours.

The results in the earlier sections can be extended to the nonstandard situation by incorporating the two possibilities. The ideas originally come from Lin, Lee, and Wahba (2002), and more details can be found there. Let the cost of false positive be c^+, and the cost of false negative be c^-. The costs may or may not be equal. It is clear that we do not need exact numbers for c^+ and c^-, what matters is the ratio of the two.

Let the probability distribution of the target population be $P^t(x, y)$. Let (X_t, Y_t) be a generic random pair from the distribution $P^t(x, y)$. Let $d^+(x)$ be the probability density of X_t for the positive population (class $\mathcal{A}$), i.e., the conditional density of X_t given $Y_t = 1$. Let $d^-(x)$ be similarly defined. The unconditional ("prior") probabilities of the positive class and negative class in the target population are denoted by π_t^+ and π_t^- respectively.

Let $p_t(x) = Pr(Y_t = 1 \mid X_t = x)$ be the conditional probability of a randomly chosen subject from the target population belonging to the positive class given $X_t = x$. We obviously have $p_t(x) = 1 - Pr(Y_t = -1 \mid X_t = x)$. By Bayes formula, we have

$$p_t(x) = \frac{\pi_t^+ d^+(x)}{\pi_t^+ d^+(x) + \pi_t^- d^-(x)} \tag{13}$$

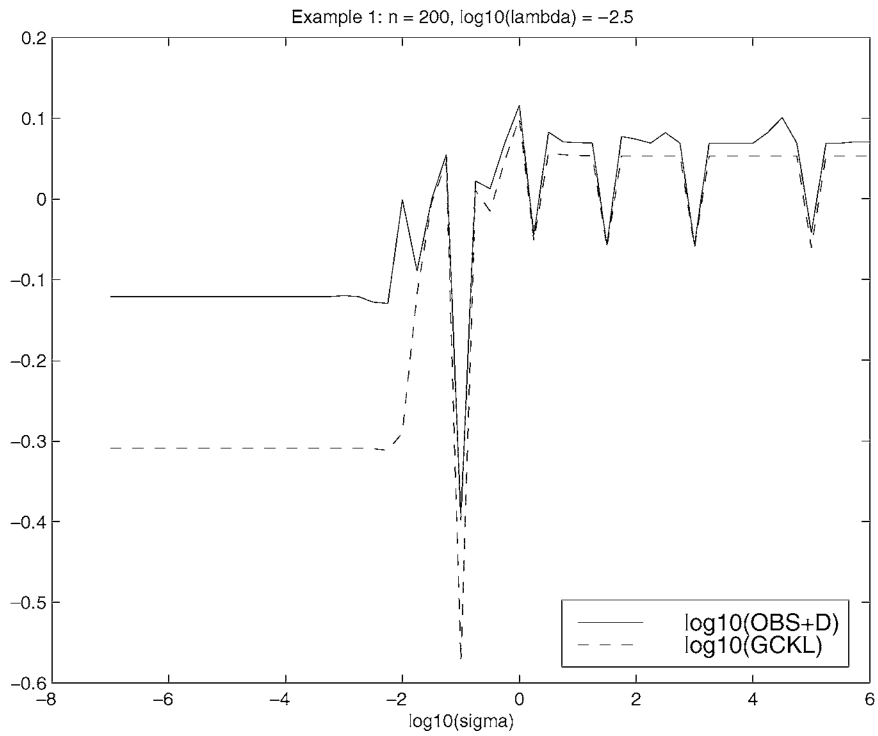

Figure 6. Plot of $\log_{10} GACV$ and $\log_{10} GCKL$ as a function of $\log_{10}\sigma$ for $\log_{10}\lambda = -2.5$.

For a classification rule ϕ that assigns a class label to any subject based on the input vector $x \in \mathcal{X}$, (that is, a mapping from $\mathcal{X}$ to $\{-1, 1\}$), the inaccuracy of ϕ when applied to the target population is characterized by the expected (or average) cost

$$E\{c^+[1 - p_t(X_t)]1(\phi(X_t) = 1) + c^- p_t(X_t)1(\phi(X_t) = -1)\} \tag{14}$$

Here $1(\cdot)$ is the indicator function: it assumes the value 1 if its argument is true, and 0 otherwise. The Bayes rule that minimizes (14) is given by

$$\phi_B(x) = \begin{cases} +1 & \text{if } \dfrac{p_t(x)}{1 - p_t(x)} > \dfrac{c^+}{c^-} \\ -1 & \text{otherwise} \end{cases} \tag{15}$$

Notice (15) reduces to (1) when we have the standard case.

Since in general we do not know $p_t(x)$, we have to learn from the training set a classification rule. As explained earlier, it is sometimes the case that a pre-specified proportion of positive examples and negative examples are included in the sample, whereas within each class the examples are randomly chosen. Denote these pre-specified proportions for

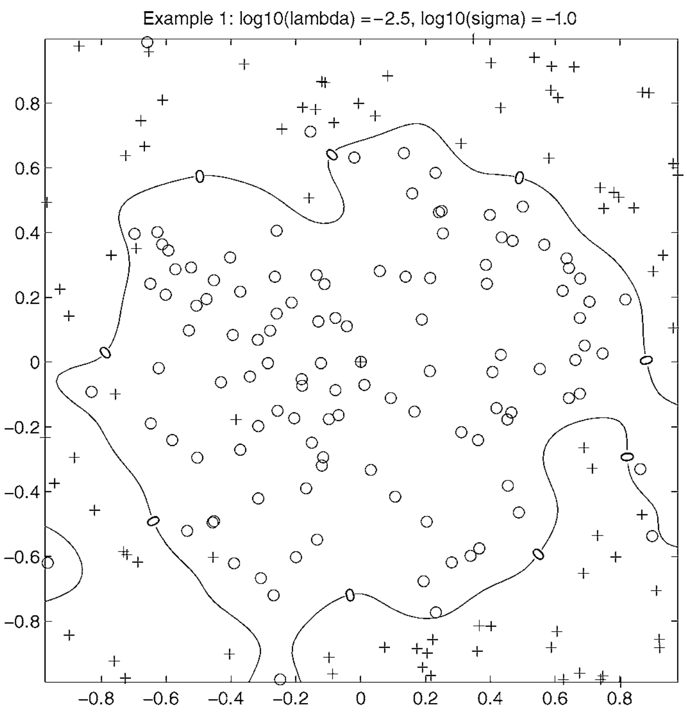

Figure 7. Level curve for $f_\lambda = 0$.

the training sample by π^+ and π^-. These may not match the population proportions π_t^+ and π_t^-. Again let (X, Y) be a generic random pair from the sampling distribution $P(x, y)$. The conditional probability of a subject in the sample belonging to the positive class given $X = x$ is

$$p_0(x) = \frac{\pi^+ d^+(x)}{\pi^+ d^+(x) + \pi^- d^-(x)} \tag{16}$$

Since the training sample only gives information about the sampling distribution where the training sample comes from, it is advantageous for us to express the Bayes rule (15) in terms of $p_0(x)$ instead of $p_t(x)$. By (13), (15), (16), we get

$$\phi_B(x) = \begin{cases} +1 & \text{if } \dfrac{p_0(x)}{1 - p_0(x)} > \dfrac{c^+}{c^-}\dfrac{\pi^+}{\pi^-}\dfrac{\pi_t^-}{\pi_t^+} \\ -1 & \text{otherwise} \end{cases} \tag{17}$$

For notational purpose, define a function L on the two element set $\{-1, 1\}$:

$$L(-1) = c^+\pi^+\pi_t^- \quad \text{and} \quad L(1) = c^-\pi^-\pi_t^+. \tag{18}$$

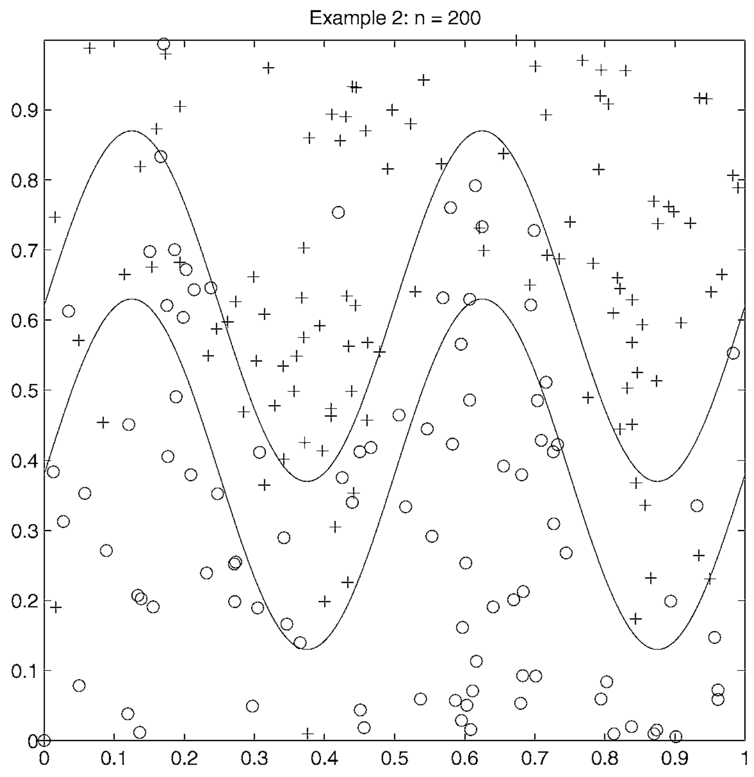

Figure 8. Data for Example 2, and regions of constant (generating) probability.

Then the Bayes rule can be expressed as

$$\phi_B(x) = \begin{cases} +1 & \text{if } p_0(x) - \dfrac{L(-1)}{L(-1)+L(1)} > 0 \\ -1 & \text{otherwise} \end{cases} \tag{19}$$

Notice $\frac{L(-1)}{L(-1)+L(1)}$ reduces to $1/2$ in the standard case. For any procedure that implements the Bayes rule by giving an estimate of the conditional probability $p_0(x)$, the adjustment from standard case to nonstandard case is fairly simple: The estimation of $p_0(x)$ remains the same, but in the nonstandard case we should compare the estimate with $\frac{L(-1)}{L(-1)+L(1)}$ instead of $1/2$. This is not the case for other methods such as the support vector machines with reproducing kernels. As we have shown, the standard support vector machine with kernel does not yield an estimate of the conditional probability $p_0(x)$, but instead estimates $sign[p_0(x) - 1/2]$, the Bayes rule in the standard situation. Since the Bayes rule in the nonstandard case can be quite different from $sign[p_0(x) - 1/2]$, the standard SVM can not perform optimally in the nonstandard situation. Also, it is impossible to estimate the Bayes rule (19) in the nonstandard case from the estimate of $sign[p_0(x) - 1/2]$, which is given

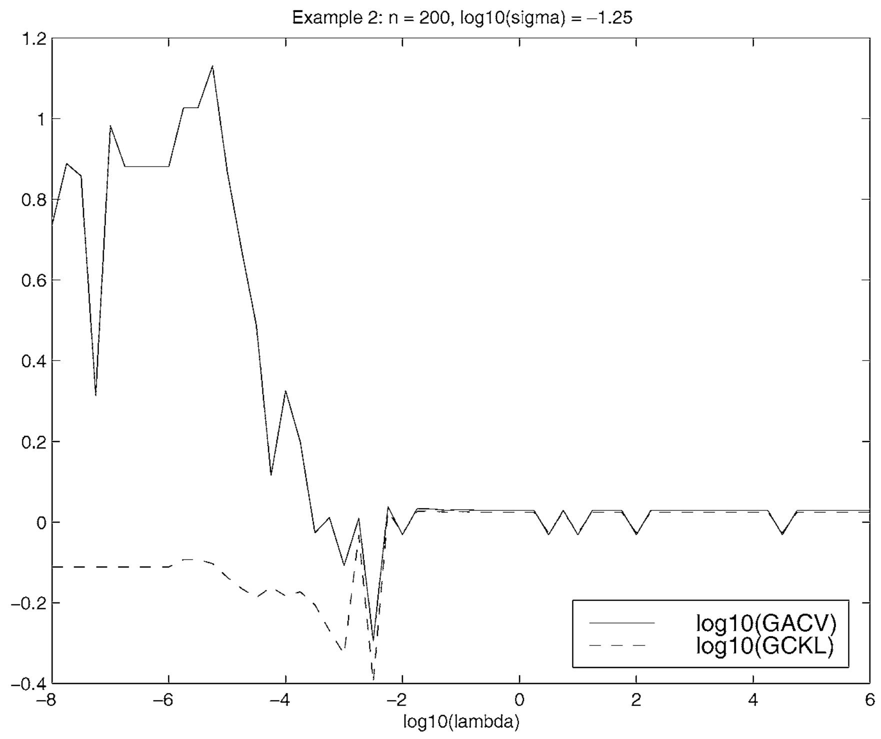

Figure 9. Plot of $\log_{10} GACV$ and $\log_{10} GCKL$ as a function of $\log_{10}\lambda$ for $\log_{10}\sigma = -1.25$.

by the standard SVM. Therefore in the nonstandard case, we need to modify the estimation procedure of support vector machines to get a correctly estimated Bayes rule.

In the nonstandard situation, $sign(p_0 - 1/2)$ is actually the classification rule that minimizes the expected misclassification rate in the sampling distribution, and we will call it the "unweighted" Bayes rule later on.

In order to extend the support vector machine methodology to the nonstandard situation, we modify the support vector machine with reproducing kernel K (4) to minimizing

$$\frac{1}{n}\sum_{i=1}^{n} L(y_i)[1 - y_i g(x_i)]_+ + \lambda \|h\|^2_{\mathcal{H}_K} \tag{20}$$

over all the functions of the form $g(x) = h(x) + b$, and $h \in \mathcal{H}_K$. Again when the minimizer $\hat{g}$ is found, the SVM classification rule is

$$\hat{g}(x) > 0 \rightarrow \mathcal{A}$$
$$\hat{g}(x) < 0 \rightarrow \mathcal{B}$$

We remark that we do not need the exact values for $L(1)$ and $L(-1)$. What we really need is the ratio of the two.

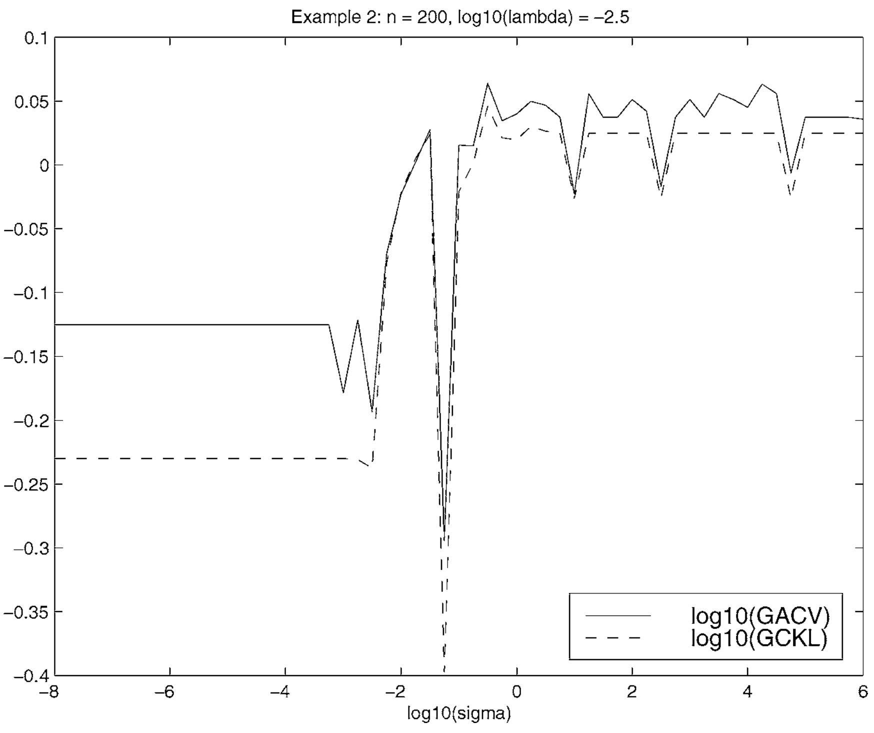

Figure 10. Plot of $\log_{10} GACV$ and $\log_{10} GCKL$ as a function of $\log_{10}\sigma$ for $\log_{10}\lambda = -2.5$.

For the regularization problem (20), the limiting functional of the data fit component is $EL(Y)[1 - Yg(X)]_+$. Thus the following lemma identifies the target function of (20).

Lemma 5.1. *The minimizer of* $EL(Y)[1 - Yg(X)]_+$ *is* $sign[p_0 - \frac{L(-1)}{L(-1)+L(1)}]$.

The proof of this lemma is similar to that of Lemma 3.1, and is given in Lin, Lee, and Wahba (2002).

Therefore the classifier induced by the general SVM (20) approximates the Bayes rule in the nonstandard case. It is easy to see that the general SVM reduces to the standard SVM in the standard case.

The calculation of the solution to (20) can be carried out in the same way as that of the standard SVM. In fact, it is easy to check that the dual formulation of (20) is almost identical to (6), (7), (8), (9), and (10), with the only changes being that (8) is replaced by

$$0 \le \alpha_i \le L(y_i),\ \forall i = 1, 2, \ldots, n, \quad \text{and} \quad y'\alpha = 0, \tag{21}$$

and (10) is replaced by

$$b = \frac{\sum_{i=1}^n \alpha_i (L(y_i) - \alpha_i)\big(y_i - \sum_{j=1}^n c_j K(x_i, x_j)\big)}{\sum_{i=1}^n \alpha_i (L(y_i) - \alpha_i)} \tag{22}$$

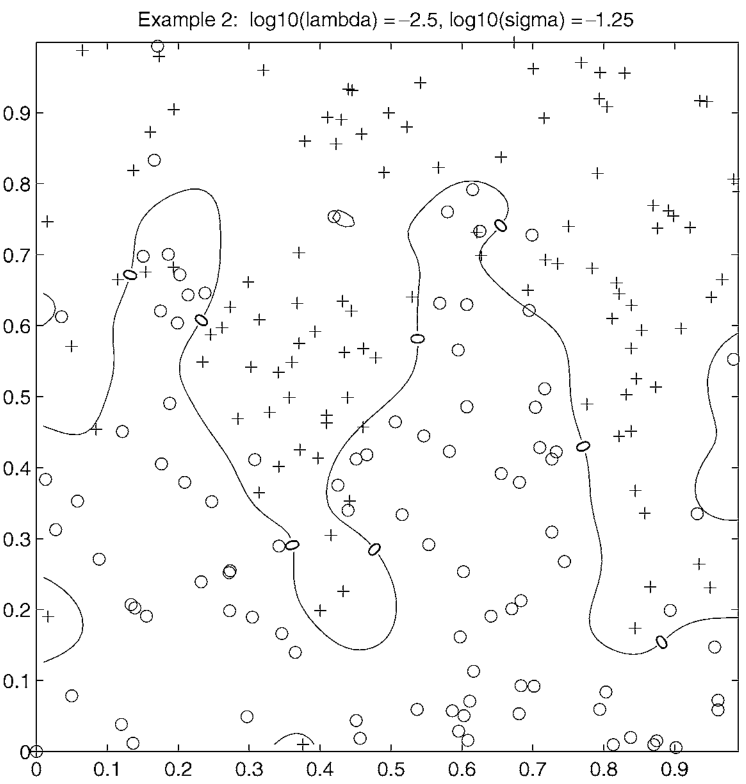

Figure 11. Level curve for $f_\lambda = 0$.

Note the whole computation procedure is very similar to that of the standard support vector machine: in that case $L(y_i)$ is replaced by 1. As a result, any algorithm for implementing the standard support vector machine can be easily modified to carry out this general procedure.

A simple simulation (shown in Lin, Lee, & Wahba, 2002) was carried out to illustrate the effectiveness of the modified support vector machine in a nonstandard situation. Consider a target population consisting of two classes $\mathcal{A}$ and $\mathcal{B}$. Class $\mathcal{A}$ has a bivariate normal distribution with mean $(0, 0)'$ and covariance matrix $diag(1, 1)$; Class $\mathcal{B}$ has a bivariate normal distribution with mean $(2, 2)'$ with covariance $diag(2, 1)$. The target population is highly unbalanced: Class $\mathcal{A}$ and $\mathcal{B}$ account for 10% and 90% of the total population, respectively. Assume the cost of false negative is twice the cost of false positive.

Suppose we have 400 examples available, 40% of which are randomly chosen from class $\mathcal{A}$, and 60% of which are randomly chosen from class $\mathcal{B}$. We use the standard and modified support vector machine to derive classification rules based on the sample. We use a Gaussian kernel in this simulation.

In order to be able to choose the smoothing parameters λ and σ, we randomly split the examples into a training set and a tuning set, each with 200 examples.

We first apply the standard support vector machine to derive a classification rule. The estimation is done with the training set and the smoothing parameters λ and σ are chosen by minimizing the misclassification rate in the tuning set. This is typical for the standard support vector machine.

We then apply the modified support vector machine. In this case $L(-1) = 0.36$, and $L(1) = 0.12$. We tune the smoothing parameters by minimizing

$$\frac{1}{200}\sum_{i=1}^{200} L(y_i')1[y_i'\hat{g}_{\lambda,\sigma}(x_i') < 0],$$

where (x_i', y_i'), $i = 1, 2, \ldots, 200$, are the examples in the tuning set, and $\hat{g}_{\lambda,\sigma}$ is the solution to (20) with smoothing parameters λ and σ. Since the tuning set comes from the same population as the training set, the above is the empirical estimate based on the tuning set of

$$EL(Y)1[Y\hat{g}_{\lambda,\sigma}(X) < 0]. \tag{23}$$

Recall that (X, Y) is a random vector from the sampling distribution. It can be shown by direct calculation that minimizing (23) is equivalent to minimizing the expected cost in the target population given by

$$E\{c^+[1 - p_t(X_t)]1[\hat{g}_{\lambda,\sigma}(X_t) > 0] + c^- p_t(X_t)1[\hat{g}_{\lambda,\sigma}(X_t) < 0]\}, \tag{24}$$

where (X_t, Y_t) is a random vector from the target population.

We ran the simulation a number of times. The results are similar to the plots in figure 12. We can see in figure 12 the decision surface of our modified support vector machine is close to that of the Bayes rule minimizing the expected cost in the target population, whereas the decision surface of the standard support vector machine follows closely that of the "unweighted" Bayes rule minimizing the expected misclassification rate in the sampling population. The points in figure 12 are the examples in the training set.

In order to be able to choose the smoothing parameters without a tuning set, we extend the concepts of *GCKL* and *GACV* to the nonstandard case. In the nonstandard situation, we can define

$$GCKL(\lambda, \sigma) = E\{c^+[1 - p_t(X_t)](1 + \hat{g}_{\lambda,\sigma}(X_t))_+ + c^- p_t(X_t)(1 - \hat{g}_{\lambda,\sigma}(X_t))_+\}.$$

It is easy to see the *GCKL* is an upper bound of the expected cost in the target population given by (24). But, to the extent that $\hat{g}_{\lambda,\sigma}$ tends to $sign[p_0 - \frac{L(-1)}{L(-1)+L(1)}]$, the *GCKL* tends to two times (24). Thus, when $\hat{g}_{\lambda,\sigma}(x)$ is close to $sign[p_0(x) - \frac{L(-1)}{L(-1)+L(1)}]$, minimizing the *GCKL* is similar to minimizing the expected cost (24). In general, the global minimum of *GCKL* is easier to identify than the global minimum of (24), since the dependence of *GCKL* on $\hat{g}_{\lambda,\sigma}$ is continuous and convex, while the dependence of (24) on $\hat{g}_{\lambda,\sigma}$ is discontinuous and not convex. Our experience is that the *GCKL* usually has a unique minimum, whereas (24) can have several local minima.

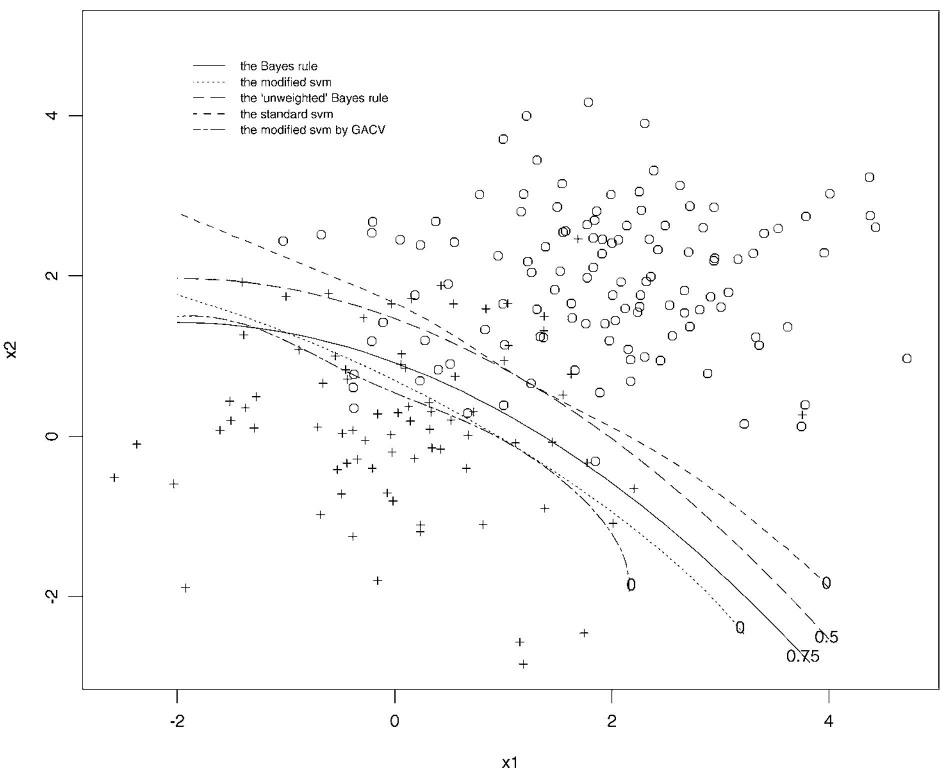

Figure 12. Decision surfaces given by the modified and standard support vector machines, the Bayes rule, and the "unweighted" Bayes rule. The modified support vector machine is implemented both with a tuning set and with the GACV.

Direct calculation shows the *GCKL* has an equivalent form: Let (X, Y) be a random vector from the sampling distribution, then

$$GCKL(\lambda, \sigma) = \frac{1}{\pi^+\pi^-} E[L(Y)(1 - Y\hat{g}_{\lambda,\sigma}(X))_+]$$

When we choose the smoothing parameter with the *GCKL*, all we care about is the minimizer of the *GCKL*. Hence we can also take the *GCKL* as $E[L(Y)(1 - Y\hat{g}_{\lambda,\sigma}(X))_+]$.

Similar to the standard case, we can derive a formula for $GACV$ as a computable proxy of the *GCKL* (see Lin, Lee, & Wahba, 2002).

$$GACV(\lambda, \sigma) = \frac{1}{n}\sum_{i=1}^{n} L(y_i)(1 - y_i\hat{g}_{\lambda,\sigma}(x_i))_+ + \hat{D}(\lambda, \sigma)$$

where

$$\hat{D}(\lambda, \sigma) = \frac{1}{n}\left[2 \sum_{y_i\hat{g}_{\lambda,\sigma}(x_i)<-1} L(y_i)\frac{\alpha_i}{2n\lambda}K(x_i, x_i) + \sum_{y_i\hat{g}_{\lambda,\sigma}(x_i)\in[-1,1]} L(y_i)\frac{\alpha_i}{2n\lambda}K(x_i, x_i)\right],$$

where α_i's are defined as in (7). The dependence of α_i's on the smoothing parameters are suppressed here to ease notation. For the Gaussian kernel, we have $K(s, s) = 1$ for any s, and the expression of *GACV* can be simplified a bit.

We use *GACV* to choose λ and σ jointly. For the training set in figure 12, the decision surface obtained by using *GACV* is also plotted in figure 12. We can see in this example the *GACV* does a decent job. Note we only need the training set for the whole estimation. Figures 13 and 14 show how *GACV* and *GCKL* vary with the λ and σ around the minimizing λ and σ. The λ and σ chosen by the *GACV* in this example are 0.0025 and 1 respectively.

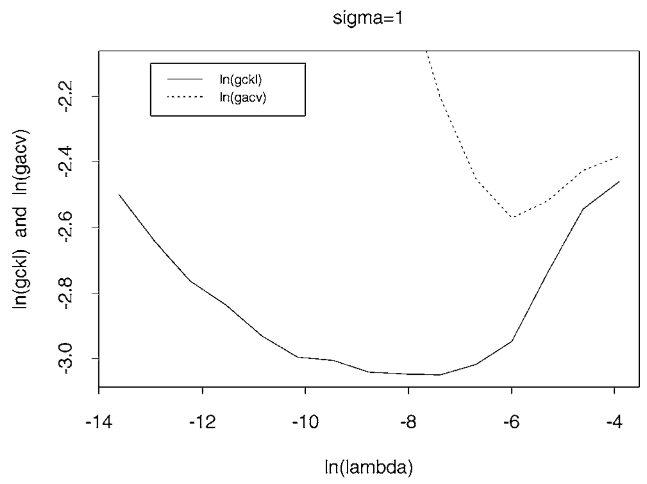

Figure 13. *GCKL* and *GACV* plot as a function of λ when σ is fixed at 1. We can see the minimizer of *GACV* is a decent estimate of the minimizer of *GCKL*.

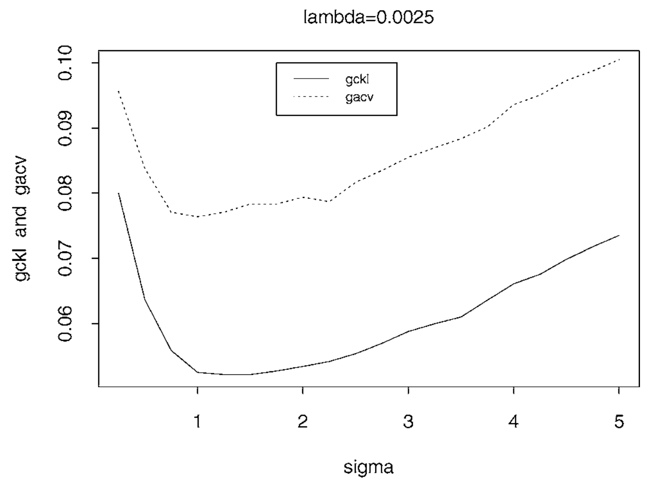

Figure 14. *GCKL* and *GACV* plot as a function of σ when λ is fixed at 0.0025. We can see the minimizer of *GACV* is a decent estimate of the minimizer of *GCKL*.

Limited experience with the *GACV* shows that it is a promising technique for picking out good choices of smoothing parameters. However, it seems to be quite variable and further investigation for possible improvement is in order.

Acknowledgment

This work was partly supported by Wisconsin Alumni Research Foundation, NSF under Grant DMS-9704758 and NIH under Grant R01 EY09946.

References

Brown, M. P. S., Grundy, W. N., Lin, D., Cristianini, N., Sugnet, C. W., Furey, T. S., Ares, M., Jr., & Haussler, D. (2000). Knowledge-based analysis of microarray gene expression data by using support vector machines. In *Proceedings of the National Academy of Sciences*, *97:1*, 262–267.

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In D. Haussler (Ed.), In *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*. Pittsburgh, PA: ACM Press.

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

Cox, D. D., & O'Sullivan, F. (1990). Asymptotic analysis of penalized likelihood and related estimates. *The Annals of Statistics*, *18:4*, 1676–1695.

Cortes, C., & Vapnik, V. N. (1995). Support vector networks. *Machine Learning*, *20*, 273–297.

Hand, D. J. (1997). *Construction and assessment of classification rules*. Chichester, England: John Wiley & Sons.

Lin, Y. (1999). Support vector machines and the Bayes rule in classification. *Data Mining and Knowledge Discovery*, to appear.

Lin, Y., Lee, Y., & Wahba, G. (2002). Support vector machines for classification in nonstandard situations. *Machine Learning, 46*, 191–202.

Poggio, T., & Girosi, F. (1998). A sparse representation for function approximation. *Neural Computation*, *10*, 1445–1454.

Vapnik, V. N. (1995). The nature of statistical learning theory. New York: Springer Verlag.

Wahba, G. (1990). Spline models for observational data. Philadelphia, PA: Society for Industrial and Applied Mathematics.

Wahba, G. (1999). Support vector machines, reproducing kernel Hilbert spaces and the randomized GACV. In B. Scholkopf, C. J. C. Burges, & A. J. Smola (Eds.), *Advances in kernel methods—Support vector learning*. Cambridge, MA: MIT Press.

Wahba, G., Lin, Y., & Zhang, H. (2000). *GACV* for support vector machines, or, another way to look at margin-like quantities. In A. J. Smola, P. Bartlett, B. Scholkopf, & D. Schurmans (Eds.), *Advances in large margin classifiers*. Cambridge, MA & London, England: MIT Press.

Received September 1, 2000
Revised January 18, 2001
Accepted January 23, 2001
Final manuscript January 26, 2001