Machine Learning, 46, 21–52, 2002

Bayesian Methods for Support Vector Machines: Evidence and Predictive Class Probabilities

PETER SOLLICH peter.sollich@kcl.ac.uk
Department of Mathematics, King's College London, Strand, London WC2R 2LS, UK

Editor: Nello Cristianini

Abstract. I describe a framework for interpreting Support Vector Machines (SVMs) as maximum a posteriori (MAP) solutions to inference problems with Gaussian Process priors. This probabilistic interpretation can provide intuitive guidelines for choosing a 'good' SVM kernel. Beyond this, it allows Bayesian methods to be used for tackling two of the outstanding challenges in SVM classification: how to tune hyperparameters—the misclassification penalty C, and any parameters specifying the kernel—and how to obtain predictive class probabilities rather than the conventional deterministic class label predictions. Hyperparameters can be set by maximizing the evidence; I explain how the latter can be defined and properly normalized. Both analytical approximations and numerical methods (Monte Carlo chaining) for estimating the evidence are discussed. I also compare different methods of estimating class probabilities, ranging from simple evaluation at the MAP or at the posterior average to full averaging over the posterior. A simple toy application illustrates the various concepts and techniques.

Keywords: Support vector machines, Gaussian processes, Bayesian inference, evidence, hyperparameter tuning, probabilistic predictions

1. Introduction

Support Vector Machine (SVM) classifiers have recently been the subject of intense research activity within the neural networks and machine learning community; for tutorial introductions and overviews of recent developments see (Burges, 1998; Smola & Schölkopf, 1998; Schölkopf et al., 1998; Cristianini & Shawe-Taylor, 2000). One of the open questions that remains is how to set the tunable parameters ('hyperparameters' in Bayesian terminology) of an SVM algorithm, such as the misclassification penalty C and any parameters specifying the kernel function (for example the width of an RBF kernel); this can be thought as the model selection problem for SVMs. It has been tackled by optimizing generalization error bounds (see e.g. Schölkopf et al., 1999; Cristianini, Campbell, & Shawe-Taylor, 1999; Chapelle & Vapnik, 2000); approximate cross-validation errors (Wahba, 1998) have also been proposed as criteria for this purpose. None of these methods, however, addresses a second important issue: the estimation of class probabilities for the predictions of a trained SVM classifier. In many applications, these would obviously be rather more desirable than just deterministic class labels.

It is thus natural to look for a probabilistic interpretation of SVM classifiers: This lends itself naturally to the calculation of class probabilities, and also allows hyperparameters to be tuned by maximization of the so-called evidence (type-II likelihood). I will describe a framework for such an interpretation in this paper. Its main advantage, compared to other recent probabilistic approaches to SVM classification (Seeger, 2000; Opper & Winther, 2000; Herbrich, Graepel, & Campbell, 1999; Kowk, 1999a, 1999b)—which are reviewed briefly below—is that it clarifies the somewhat subtle issue of normalization of the probability model. I hope to show that this is important if Bayesian methods are to be applied to SVMs in a principled way.

In Section 2, I set out the basic ingredients of the probabilistic framework. The SVM kernel is shown to define a Gaussian process prior over functions on the input space. This interpretation avoids the need to think in terms of high-dimensional feature spaces, and relates kernels to prior assumptions about the kind of classification problem at hand. As explained in Section 3, it may thus help with the choice of a 'good' kernel if such prior knowledge is available. In Section 4, I define the evidence for SVMs; because the probability model is a joint one for inputs and outputs, there are two possible choices for such a definition, a joint and a conditional evidence. Methods for approximating the evidence analytically or estimating it numerically are discussed in Section 5, while Section 6 deals with the definition and evaluation of predictive class probabilities. Section 7 then illustrates the ideas and concepts introduced up to that point by means of a simple toy application. Finally, in Section 8, I discuss the relationship between the work presented in this paper and other recent probabilistic approaches to SVM classification, and outline perspectives for future work. A brief summary of some of the above results can be found in the conference proceedings (Sollich, 2000); an earlier version of the probabilistic framework (Sollich, 1999) disregarded the issue of normalization and should therefore be regarded as superseded.

At the end of this introduction, I would like to stress the philosophy behind the present paper. There is clearly a large community of SVM users, and there have been many successful applications of SVM classifiers. My approach here aims to make standard Bayesian methods available for these applications, while leaving as much as possible of the standard SVM framework intact. Hence the insistence, for example, on creating a probability model whose maximum a posteriori solution yields the standard SVM classifier. I do *not* address the wider question of whether there are "better" classification models which combine a straightforward probabilistic justification with the benefits of SVMs (fast training, sparse solutions); but this would certainly be an exciting topic for future research.

2. Support vector machines: A probabilistic framework

I focus on two-class classification problems. Suppose we are given a set D of n training examples (x_i, y_i) with binary outputs $y_i = \pm 1$ corresponding to the two classes[1]. The basic SVM idea is to map the inputs x to vectors $\phi(x)$ in some high-dimensional feature space; ideally, in this feature space, the problem should be linearly separable. Suppose first that this is true. Among all decision hyperplanes $\mathbf{w} \cdot \phi(x) + b = 0$ which separate the training examples (i.e. which obey $y_i(\mathbf{w} \cdot \phi(x_i) + b) > 0$ for all $x_i \in X$, X being the set of training inputs), the SVM solution is chosen as the one with the largest *margin*, i.e. the largest minimal distance from any of the training examples. Equivalently, one specifies the margin to be equal to 1 and minimizes the squared length of the weight vector $\|\mathbf{w}\|^2$ (Cristianini & Shawe-Taylor, 2000), subject to the constraint that $y_i(\mathbf{w} \cdot \phi(x_i) + b) \geq 1$ for all i.

[The quantities $\gamma_i = y_i(\mathbf{w}\cdot\phi(x_i)+b)$ are again called margins, although for an unnormalized weight vector they no longer represent geometrical distances (Cristianini & Shawe-Taylor, 2000)]. This leads to the following optimization problem: Find a weight vector $\mathbf{w}$ and an offset b such that $\frac{1}{2}\|\mathbf{w}\|^2$ is minimized, subject to the constraint that $y_i(\mathbf{w}\cdot\phi(x_i)+b)\geq 1$ for all training examples.

If the problem is not linearly separable (or if one wants to avoid fitting noise in the training data[2]) 'slack variables' $\xi_i \geq 0$ are introduced which measure how much the margin constraints are violated; one thus writes $y_i(\mathbf{w}\cdot\phi(x_i)+b)\geq 1-\xi_i$. To control the amount of slack allowed, a penalty term $C\sum_i \xi_i$ is then added to the objective function $\frac{1}{2}\|\mathbf{w}\|^2$, with a penalty coefficient C. Training examples with $y_i(\mathbf{w}\cdot\phi(x_i)+b)\geq 1$ (and hence $\xi_i = 0$) incur no penalty; the others contribute $C[1-y_i(\mathbf{w}\cdot\phi(x_i)+b)]$ each. This gives the SVM optimization problem: Find $\mathbf{w}$ and b to minimize

$$\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i l(y_i[\mathbf{w}\cdot\phi(x_i)+b]) \tag{1}$$

where $l(z)$ is the (shifted) 'hinge loss', also called soft margin loss,

$$l(z) = (1-z)H(1-z) \tag{2}$$

The Heaviside step function $H(1-z)$ (defined as $H(a)=1$ for $a\geq 0$ and $H(a)=0$ otherwise) ensures that this is zero for $z>1$.

To interpret SVMs probabilistically, one can regard (1) as defining a negative log-posterior probability for the parameters $\mathbf{w}$ and b of the SVM, given a training set D. The conventional SVM classifier is then interpreted as the maximum a posteriori (MAP) solution of the corresponding probabilistic inference problem. The first term in (1) gives the prior $Q(\mathbf{w}, b)\propto \exp(-\frac{1}{2}\|\mathbf{w}\|^2 - \frac{1}{2}b^2B^{-2})$. This is a Gaussian prior on $\mathbf{w}$; the components of $\mathbf{w}$ are uncorrelated with each other and have unit variance. I have also chosen a Gaussian prior on b with variance B^2. The flat prior implied by (1) can be recovered by letting $B\to\infty$, but it actually often makes more sense to keep B finite in a probabilistic setting (see below).

Because only the 'latent function' values $\theta(x) = \mathbf{w}\cdot\phi(x)+b$—rather than $\mathbf{w}$ and b individually—appear in the second, data dependent term of (1), it makes sense to express the prior directly as a distribution over these. The $\theta(x)$ have a joint Gaussian distribution because b and the components of $\mathbf{w}$ do, with covariances given by

$$\langle\theta(x)\theta(x')\rangle = \langle(\phi(x)\cdot\mathbf{w})(\mathbf{w}\cdot\phi(x'))\rangle + B^2 = \phi(x)\cdot\phi(x') + B^2$$

The SVM prior is therefore simply a *Gaussian process* (GP) over the functions θ, with zero mean; its covariance function is

$$K(x,x') = \tilde{K}(x,x') + B^2, \qquad \tilde{K}(x,x') = \phi(x)\cdot\phi(x') \tag{3}$$

and (except for the additive term B^2, which arises here because I have incorporated the offset b into $\theta(x)$) is called the kernel in the SVM literature. To understand the role of B, note that one can write $\theta(x) = \tilde{\theta}(x) + b$, where $\tilde{\theta}(x)$ is a zero mean GP with covariance function $\tilde{K}(x, x')$ and b is a Gaussian distributed offset with mean zero and standard deviation B. Consider now the case where $\tilde{K}(x, x)$ is independent of x; an example would be the RBF kernel discussed below. Then a typical sample from the prior will have values of $\tilde{\theta}(x)$ in a range of order $[\tilde{K}(x, x)]^{1/2}$ around zero. If $B \gg [\tilde{K}(x, x)]^{1/2}$, then in $\theta(x) = \tilde{\theta}(x) + b$ the second term will typically be dominant, implying that $\theta(x)$ will very likely have the same sign for all inputs (namely, the sign of b); this becomes true with probability one for $B \to \infty$. The prior then assigns nonzero probability only to two classifiers, those which return the same label (either $+1$ or -1) for all inputs. It is in order to avoid this pathological situation that I suggest keeping B finite in a probabilistic context. Note that the above argument does not apply to kernels where $\tilde{K}(x, x)$ is strongly x-dependent. An example would be the linear SVM, with $\tilde{K}(x, x') = x \cdot x'$. Even in such cases, however, one may want to work with finite B, but treating its actual value as an adjustable hyperparameter. For data sets that require a large offset b to achieve a good fit, evidence maximization (see below) should then automatically yield a large B.

The above link between SVMs and GPs has been pointed out by a number of authors, e.g. (Seeger, 2000; Opper & Winther, 2000). It can be understood from the common link to reproducing kernel Hilbert spaces (Wahba, 1998), and can be extended from SVMs to more general kernel methods (Jaakkola & Haussler, 1999). For connections to regularization operators see also Smola et al. (1998).

Before discussing the probabilistic interpretation of the second (loss) term in (1), let us digress briefly to the SVM regression case. There, the training outputs y_i are real numbers, and each training example contributes $Cl_\epsilon(y_i - [\mathbf{w} \cdot \phi(x_i) + b]) = Cl_\epsilon(y_i - \theta(x_i))$ to the overall loss, where $l_\epsilon(z) = (|z| - \epsilon)H(|z| - \epsilon)$ is Vapnik's ϵ-insensitive loss function. This can be turned into a negative log-likelihood if one defines the probability of obtaining output y for a given input x as

$$Q(y \mid x, \theta) = \kappa(C, \epsilon) \exp[-Cl_\epsilon(y - \theta(x))]$$

As the notation indicates, this probability of course depends on the latent function θ (through its value $\theta(x)$ at x); in the feature space notation possibly more familiar to readers, one would write the same equation as $Q(y \mid x, \mathbf{w}, b) = \kappa(C, \epsilon) \exp\{-Cl_\epsilon(y - [\mathbf{w} \cdot \phi(x_i) + b])\}$. The constant $\kappa(C, \epsilon)$ is chosen to normalize $Q(y \mid x, \theta)$ across all possible values of y; it is explicitly given by $[\int dy \exp[-Cl_\epsilon(y)]]^{-1}$ and is independent of $\theta(x)$. The above interpretation was advanced by (Pontil, Mukherjee, & Girosi, 1998), who also showed that $Q(y \mid x, \theta)$ can be represented as a superposition of Gaussians with a distribution of means and variances.

Returning to the present classification scenario, the second term in (1) similarly becomes a (negative) log-likelihood if we define the probability of obtaining output y for a given x (and θ) as

$$Q(y = \pm 1 \mid x, \theta) = \kappa(C) \exp[-Cl(y\theta(x))] \tag{4}$$

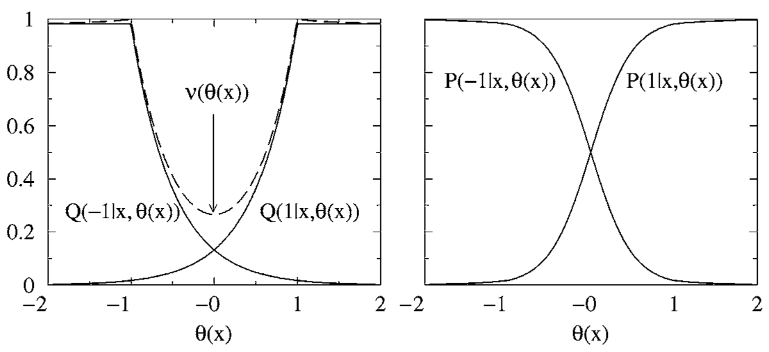

Figure 1. Left: The unnormalized class probabilities $Q(y = \pm 1 \mid x, \theta(x))$ as a function of the value $\theta(x)$ of the latent function, and their sum $\nu(\theta(x))$. Right: The normalized class probabilities $P(y = \pm 1 \mid x, \theta(x))$.

Here I set $\kappa(C) = 1/[1 + \exp(-2C)]$ to ensure that the probabilities for $y = \pm 1$ never add up to a value larger than one. The likelihood for the complete data set is then

$$Q(D \mid \theta) = \prod_i Q(y_i \mid x_i, \theta) Q(x_i)$$

with some input distribution $Q(x)$ which remains essentially arbitrary at this point. However, in contrast to the regression case, this likelihood function is *not normalized*, because

$$\begin{aligned}\nu(\theta(x)) &= Q(1 \mid x, \theta) + Q(-1 \mid x, \theta)\\ &= \kappa(C)\{\exp[-Cl(\theta(x))] + \exp[-Cl(-\theta(x))]\} < 1\end{aligned}$$

except when $|\theta(x)| = 1$ (see figure 1). Correspondingly, the sum of the data set likelihood $Q(D \mid \theta)$ over all possible data sets D with the given size n,

$$\sum_D Q(D \mid \theta) \equiv \sum_{\{x_i, y_i\}} Q(D \mid \theta) = \left(\sum_x Q(x)\nu(\theta(x))\right)^n \tag{5}$$

is also less than one in general. Here the unrestricted sum over x runs over all possible inputs; this notation will be used throughout. I assume here that the input domain is discrete. This avoids mathematical subtleties with the definition of determinants and inverses of operators (rather than matrices), while maintaining a scenario that is sufficiently general for all practical purposes: A continuous input space can always be covered with a sufficiently fine grid, at least conceptually, or discretized in some other way.

It may not be apparent to the reader why the issue of the normalization of the likelihood is important; in fact, I disregarded this problem myself in my first attempt at a probabilistic interpretation of SVMs (Sollich, 1999). However, it turns out that if one wants to construct a quantity like the evidence—the likelihood of the data, given the hyperparameters defining the SVM classification algorithm—and use it to tune these hyperparameters, then normalization of the probability model is crucial if one wants to have even the most modest guarantee that this procedure is sensible. I defer a detailed discussion of this issue to Section 4, which deals with the definition of the evidence for SVM classification.

If we accept for now that the probability model needs to be normalized, a naive approach that comes to mind is to replace $Q(y \mid x, \theta)$ by $Q(y \mid x, \theta)/\nu(\theta(x))$. While this certainly solves the normalization problem, it also destroys the property that the conventional SVM is obtained as the MAP solution: Because $\nu(\theta(x))$ depends on $\theta(x)$, the log-likelihood $\ln[Q(y \mid x, \theta)/\nu(\theta(x))]$ would no longer be proportional to the hinge loss (2). Instead, I write the normalized probability model as

$$P(D, \theta) = Q(D \mid \theta) Q(\theta)/\mathcal{N}(D) \tag{6}$$

Its posterior probability $P(\theta \mid D) = Q(\theta \mid D) \propto Q(D \mid \theta) Q(\theta)$ is independent of the normalization factor $\mathcal{N}(D)$, and thus equal to that of the unnormalized model. By construction, the MAP value of θ therefore remains the SVM solution. I adopt the simplest choice of $\mathcal{N}(D)$ that normalizes $P(D, \theta)$. This consists in taking $\mathcal{N}$ as D-independent; its value then follows from (5) as

$$\mathcal{N}(D) = \mathcal{N} = \int d\theta\, Q(\theta) N^n(\theta), \quad N(\theta) = \sum_x Q(x)\nu(\theta(x)). \tag{7}$$

Conceptually, this corresponds to the following procedure of sampling from $P(D, \theta)$: First, sample θ from the GP prior $Q(\theta)$. Then, for each data point, sample x from $Q(x)$. Assign outputs $y = \pm 1$ with probability $Q(y \mid x, \theta)$, respectively. With the *remaining* probability $1 - \nu(\theta(x))$ (the 'don't know' class probability introduced in (Sollich, 1999) to avoid the normalization problem), restart the whole process by sampling a new θ. Because $\nu(\theta(x))$ is smallest[3] inside the 'gap' $|\theta(x)| < 1$, functions θ with many values in this gap are less likely to 'survive' until a data set of the required size n is built up.

If we calculate the prior and likelihood of the normalized probability model by taking marginals of (6), we find results entirely consistent with this sampling interpretation. The prior

$$P(\theta) \propto Q(\theta) N^n(\theta) \tag{8}$$

has an n-dependent factor which reflects the different 'survival probabilities' of different θ. The likelihood

$$P(D \mid \theta) = \prod_i P(y_i \mid x_i, \theta) P(x_i \mid \theta) \tag{9}$$

is a product of the likelihoods of the individual training examples, as expected. The conditional likelihood for the output y,

$$P(y \mid x, \theta) = Q(y \mid x, \theta)/\nu(\theta(x)) \tag{10}$$

is now normalized over $y = \pm 1$ as it should be; explicitly, one has

$$\begin{aligned} P(y \mid x, \theta) &= \frac{1}{1 + e^{-2Cy\theta(x)}} \quad \text{for} \ \ |\theta(x)| \leq 1 \\ &= \frac{1}{1 + e^{-Cy[\theta(x) + \mathrm{sgn}(\theta(x))]}} \quad \text{for} \ |\theta(x)| > 1 \end{aligned} \tag{11}$$

This shows (see also figure 1) that the class probabilities have a roughly sigmoidal dependence on $\theta(x)$, with a change of slope at $\theta(x) = \pm 1$. The misclassification penalty parameter C is proportional to the slope of the curves at the origin, and so $1/C$ can be interpreted as a noise level which measures how stochastic the outputs are. Finally, the input density

$$P(x \mid \theta) \propto Q(x)\nu(\theta(x)) \tag{12}$$

of the normalized probability model is influenced by the function θ itself; it is reduced in the gap $|\theta(x)| < 1$ where $\nu(\theta(x))$ tends to be smaller. The model thus implicitly assumes that regions of data with larger input density and well-determined outputs (large $|\theta(x)|$) are separated by gap regions ($|\theta(x)| < 1$) with lower input density and more uncertain outputs.

To summarize, Eqs. (8–12) define a probabilistic inference model whose MAP solution $\theta^* = \mathrm{argmax}\ P(\theta \mid D)$ for a given data set D is identical to a standard SVM. The prior $P(\theta)$ is a GP prior $Q(\theta)$ modified by a data set size-dependent factor.[4] This feature of a data set-dependent prior is not unique to the approach adopted here; compare for example the recently proposed Relevance Vector Machine (Tipping, 2000), whose prior depends not only on the size of the data set but even on the location of the training inputs. The likelihood (9, 10, 12) defines not just a conditional output distribution for each latent function θ, but also an input distribution (relative to some arbitrary $Q(x)$). So we really have a joint input-output model. All relevant properties of the feature space are encoded in the underlying GP prior $Q(\theta)$, with covariance function equal to the kernel $K(x, x')$. The log-posterior of the model

$$\ln P(\theta \mid D) = -\frac{1}{2}\sum_{x,x'} \theta(x)K^{-1}(x, x')\theta(x') - C\sum_i l(y_i\theta(x_i)) + \text{const} \tag{13}$$

(where $K^{-1}(x, x')$ are the elements of the inverse of $K(x, x')$, viewed as a matrix) is just a transformation of (1) from $\mathbf{w}$ and b to θ. By construction, its maximum $\theta^*(x)$ gives the conventional SVM. This is easily verified explicitly: Differentiating (13) w.r.t. the $\theta(x)$ for *non-training inputs* implies

$$\sum_{x'} K^{-1}(x, x')\theta^*(x') = 0$$

at the maximum. One can therefore write θ^* in the form

$$\theta^*(x) = \sum_i \alpha_i y_i K(x, x_i) \tag{14}$$

where the factors y_i have been separated out to retain the standard SVM notation. Maximizing w.r.t. the remaining $\theta(x_i) \equiv \theta_i$ divides the training input set X into three parts: Depending on whether $y_i\theta_i^* > 1$, $=1$ or <1, one has $\alpha_i = 0, \alpha_i \in [0, C]$ or $\alpha_i = C$. I will call these training examples easy (correctly classified), marginal, and hard, respectively. The last two classes form the support vectors (with $\alpha_i > 0$). Note that the quantities $\gamma_i = y_i\theta_i^*$ are simply the margins of the various training points. We see here that these also can be visualized directly in the input (rather than the feature) space: They are simply the values θ_i^* of the latent function at the training inputs, multiplied by the sign of the corresponding training output.

Note finally that the sparseness of the SVM solution (often the number of support vectors is $\ll n$) comes from the fact that the hinge loss $l(z)$ is constant for $z > 1$. This contrasts with other uses of GP models for classification (see e.g. Barber & Williams, 1997; Williams, 1998), where instead of the likelihood (4) a sigmoidal 'transfer function' is used. A logistic transfer function, for example, corresponds to replacing the hinge loss $l(z)$ by $l_{GP}(z) = C^{-1}\ln(1+e^{-Cz})$, which has nonzero slope everywhere. Moreover, in the noise free limit $C \to \infty$, the transfer function $P(y \mid x, \theta(x)) = \exp[-Cl_{GP}(y\theta(x))] = 1/[1+e^{-Cy\theta(x)}]$ becomes a step function $H(y\theta(x))$, and the MAP values θ^* will tend to the trivial solution $\theta^*(x) = 0$. This illuminates from an alternative point of view why the margin (the 'shift' in the hinge loss) is important for SVMs.

3. Understanding kernels

Within the probabilistic framework outlined in the previous section, the main effect of the kernel in SVM classification is to change the properties of the underlying GP prior $Q(\theta)$ in $P(\theta) \propto Q(\theta)N^n(\theta)$. The fact that the kernel simply corresponds to the covariance function of a GP makes it easy to understand its role in SVM inference. Recall that the covariance function of a GP encodes in an easily interpretable way prior assumptions about what kind of functions θ are likely to occur (see e.g. Williams, 1998). Smoothness is controlled by the behaviour of $K(x, x')$ for $x' \to x$: The Ornstein-Uhlenbeck (OU) covariance function $K(x, x') \propto \exp(-|x - x'|/l)$, for example, produces very rough (non-differentiable) functions, while functions sampled from the radial basis function (RBF) prior with $K(x, x') \propto \exp[-|x - x'|^2/(2l^2)]$ are infinitely often differentiable. Figure 2 shows how this translates into smoothness of the decision boundaries. The 'length scale' parameter l corresponds directly to the distance in input space over which we expect the function θ to vary significantly. It therefore determines the typical size of the decision regions of SVM classifiers sampled from the prior (see figure 2). The amplitude of the covariance function also has an intuitive meaning. It determines the typical values of $|\theta(x)|$, which are of order $A(x) = [K(x, x)]^{1/2}$. Using the fact that $\theta(x)$ varies from typically $+A(x)$ to $-A(x)$ over a length l, one then estimates that the function $\theta(x)$ traverses the gap $|\theta(x)| < 1$ over distances of order $l/A(x)$. This distance therefore gives the typical size of the gap regions which separate regions of higher input density. For large kernel amplitudes $A(x)$ this separation is much less than l, the typical size of the decision regions itself. Kernel amplitudes of order unity or less, on the other hand, correspond to the prior belief that the decision regions are separated by wide gap regions; see again figure 2.

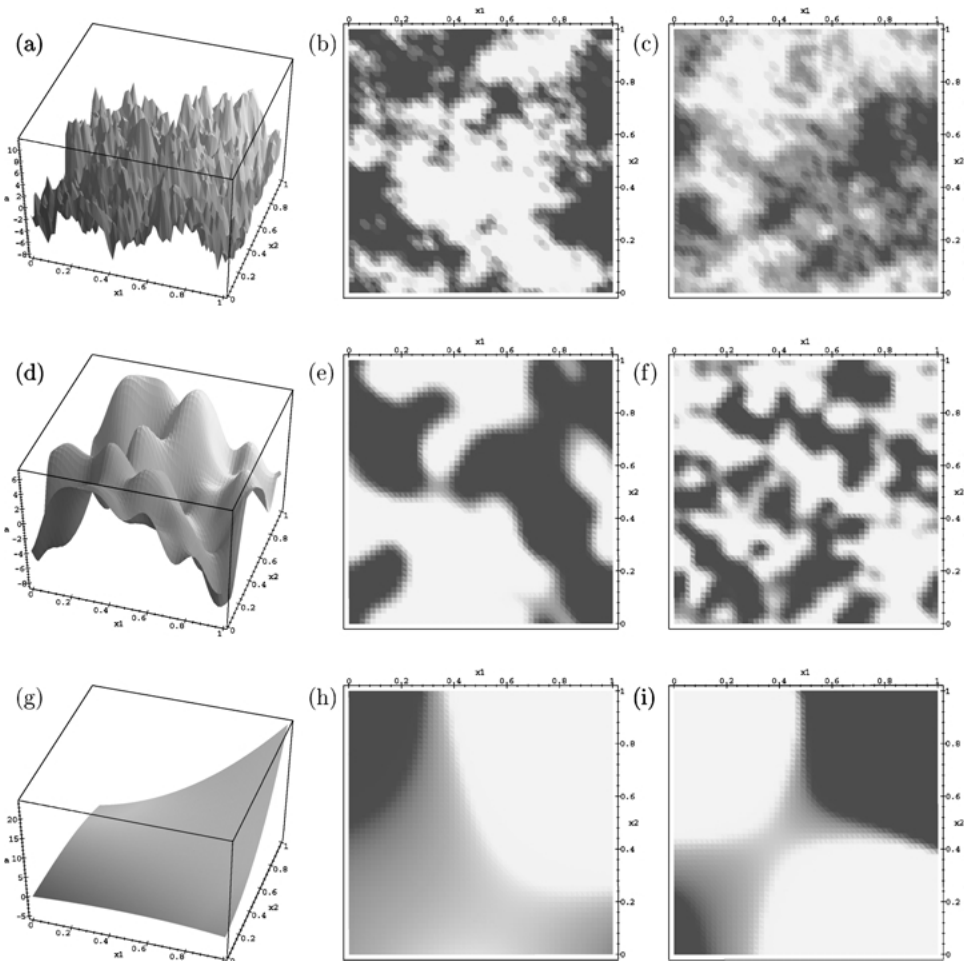

Figure 2. Samples from SVM priors; the input space is the unit square $[0, 1]^2$. 3d plots are samples $\theta(x)$ from the underlying Gaussian process prior $Q(\theta)$. 2d greyscale plots represent the output distributions obtained when $\theta(x)$ is used in the likelihood model (10) with $C = 2$; the greyscale indicates the probability of $y = 1$ (black: 0, white: 1). (a,b) Exponential (Ornstein-Uhlenbeck) kernel/covariance function $K_0 \exp(-|x - x'|/l)$, giving rough $\theta(x)$ and decision boundaries. Length scale $l = 0.1$, $K_0 = 10$. (c) Same with $K_0 = 1$, i.e. with a reduced amplitude of $\theta(x)$; note how, in a sample from the prior corresponding to this new kernel, the grey gaps (given roughly by $|\theta(x)| < 1$) between regions of definite outputs (black/white) have widened. (d, e) As first row, but with radial basis function (RBF) kernel $K_0 \exp[-(x - x')^2/(2l^2)]$, yielding smooth $\theta(x)$ and decision boundaries. (f) Changing l to 0.05 (while holding K_0 fixed at 10) and taking a new sample shows how this parameter sets the typical length scale for decision regions. (g, h) Polynomial kernel $(1 + x \cdot x')^p$, with $p = 5$; (i) $p = 10$. The absence of a clear length scale and the widely differing magnitudes of $\theta(x)$ in the bottom left ($x = [0, 0]$) and top right ($x = [1, 1]$) corners of the square make this kernel less plausible from a probabilistic point of view.

The above discussion shows that prior knowledge about (1) the smoothness of decision region boundaries, (2) the typical size of decision regions, or (3) the size of the gap regions between them, when it is available, has clear implications for the choice of an appropriate SVM kernel. To conclude this section, I also show in figure 2 a sample from the popular polynomial SVM kernel, $K(x, x') = (1 + x \cdot x')^p$. In contrast to the OU and RBF kernel, this is not translationally (but still rotationally) invariant; it also does not define a scale for the size of decision regions. In addition, $K(x, x)$ is no longer spatially uniform. In fact, for the unit square shown in the figure, the values of $K(x, x)$ differ by a factor of 3^p between the bottom left and top right corner; for the cases $p = 5$, 10 shown, this means that the typical amplitudes of $\theta(x)$ at these two points will differ by factors of 15.6 and 243, respectively. Unless justified by strong prior knowledge, such a large variation is likely to make a polynomial kernel suboptimal for most problems. One may conjecture that kernels constructed from Chebychev polynomials (which are bounded) would not suffer from the same problems; but they are likely to be more costly computationally.

Note that all the samples in figure 2 are from $Q(\theta)$, rather than from the effective prior $P(\theta)$ of the normalized probability model. One finds, however, that the n-dependent factor $N^n(\theta)$ does not change the properties of the prior qualitatively. Quantitative changes arise because function values with $|\theta(x)| < 1$ are 'discouraged' for large n; this tends to increase the size of the decision regions and narrow the gap regions between them. Figure 3 shows

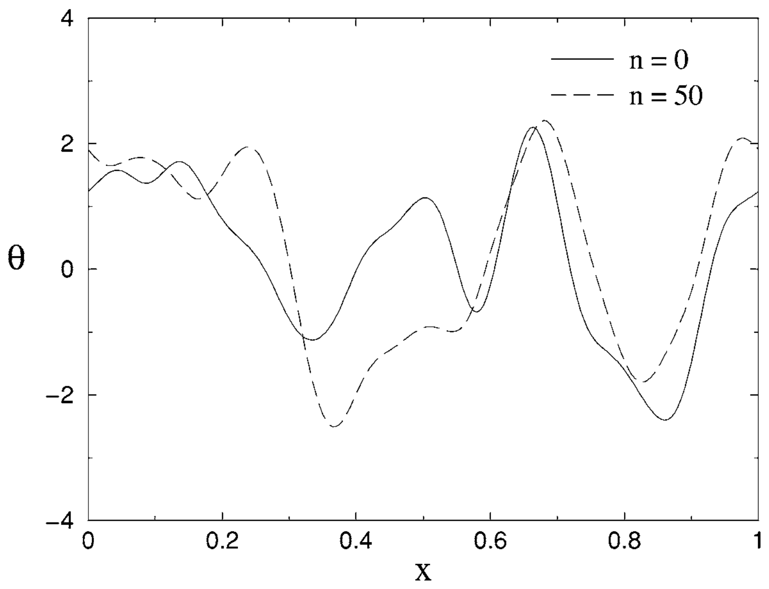

Figure 3. The effect of n on the prior $P(\theta) \propto Q(\theta)N^n(\theta)$ of the normalized probability model. For the RBF kernel $K(x, x') = K_0 \exp[-(x - x')^2/(2l^2)]$ with $K_0 = 1.25$ and $l = 0.05$, the solid line shows a sample $\theta(x)$ from the prior $Q(\theta)$ of the unnormalized model, which is equivalent to $P(\theta)$ for $n = 0$. A sample from $P(\theta)$ for $n = 50$ (dashed line) is qualitatively similar. As expected, quantitative differences arise because the sample for the larger value of n has more of a tendency to 'avoid' the gap $|\theta(x)| < 1$.

an example: I chose an RBF kernel $K(x, x') = K_0 \exp[-(x - x')^2/(2l^2)]$ with $K_0 = 1.25$ and $l = 0.05$, and $C = 2$, and took a large number of samples from $Q(\theta)$ (corresponding to $P(\theta)$ for $n = 0$) and from $P(\theta)$ for $n = 50$. The figure displays two exemplary samples and shows that there are no qualitative differences between them. Sampling experiments for a range of other kernel parameters and shapes (not shown) confirm this conclusion.

4. Defining the evidence

Beyond providing intuition about SVM kernels, the probabilistic framework introduced in Section 2 also makes it possible to apply more quantitative Bayesian methods to SVMs. In this section, we turn to the question of how to define the evidence. This is the likelihood $P(D)$ of the data D, given the model as specified by some hyperparameters; Bayes theorem $P(\theta \mid D) = P(D \mid \theta)P(\theta)/P(D)$ shows that it can also be interpreted as the normalization factor for the posterior. In our case, the hyperparameters are C and any parameters appearing in the definition of the kernel $K(x, x')$; we denote these collectively by λ. Explicitly, the evidence is obtained from (6) by integrating over the 'parameters' $\theta(x)$ (which correspond to the weights in a neural network classifier) of the SVM. Using (6,7), one finds

$$P(D) = Q(D)/\mathcal{N}, \quad Q(D) = \int d\theta\, Q(D \mid \theta) Q(\theta). \tag{15}$$

where $d\theta$ is a shorthand for $\prod_x d\theta(x)$. The factor $Q(D)$ is the 'naive' evidence derived from the unnormalized likelihood model; the correction factor $\mathcal{N}$ ensures that $P(D)$ is normalized over all data sets (of the given size n).

One of the main uses of the evidence is for tuning the hyperparameters λ: If we have no prior preference for any particular hyperparameter values, corresponding to a flat (uninformative) prior $P(\lambda)$, the posterior probability of λ given the data is

$$P(\lambda \mid D) \propto P(D \mid \lambda)P(\lambda) \propto P(D \mid \lambda) \tag{16}$$

The most probable values of the λ are then those that maximize the evidence. (In a fully Bayesian treatment, the hyperparameters should be integrated over, with a weight proportional to the evidence; maximizing the evidence can be seen as an approximation to this which assumes that the evidence is sharply peaked around its maximum (MacKay, 1992)). Note that in (16), I have made the conditioning of the evidence on the hyperparameters explicit by writing $P(D \mid \lambda)$ instead of $P(D)$.

We are now in a position to understand why normalization of the probability model is important. Let us assume that there exist 'true' values of the hyperparameters in the sense that $P(D \mid \lambda^*) = P_{\text{true}}(D)$ for some unique λ^*, where $P_{\text{true}}(D)$ is the probability distribution over data sets which 'nature' generates. One would then hope that the maximum of the evidence—or, equivalently, the log-evidence—would lie close to λ^*, at least on average.

Averaging the log-evidence over all data sets, one has

$$\sum_D P_{\text{true}}(D) \ln P(D \mid \lambda) = \sum_D P_{\text{true}}(D) \ln P_{\text{true}}(D) - \sum_D P_{\text{true}}(D) \ln \frac{P_{\text{true}}(D)}{P(D \mid \lambda)} \tag{17}$$

Now, *because* $P(D \mid \lambda)$ *is normalized* over data sets, i.e., $\sum_D P(D \mid \lambda) = 1$ for all λ, the second term on the r.h.s. can be identified as a negative KL-divergence or cross-entropy. Since the KL-divergence is always non-negative and achieves its unique minimum (zero) when $P(D \mid \lambda) = P_{\text{true}}(D)$, it follows that the average log-evidence has its global maximum at $\lambda = \lambda^*$. So *for the normalized probability model*, maximizing the evidence gives the true hyperparameters at least in an average sense. (I am grateful to Manfred Opper for bringing this elegant argument to my attention).

If, on the other hand, we consider the unnormalized evidence, related to $P(D \mid \lambda)$ by $Q(D \mid \lambda) = P(D \mid \lambda)\mathcal{N}(\lambda)$, the log-average over data sets becomes

$$\begin{aligned}\sum_D P_{\text{true}}(D) \ln Q(D \mid \lambda) = {} & \sum_D P_{\text{true}}(D) \ln P_{\text{true}}(D) - \sum_D P_{\text{true}}(D) \ln \frac{P_{\text{true}}(D)}{P(D \mid \lambda)} \\ & + \sum_D P_{\text{true}}(D) \ln \mathcal{N}(\lambda)\end{aligned} \tag{18}$$

The presence of the last term now means that there is no guarantee that this average will be maximized at the true values of the hyperparameters; in fact, the position of the maximum will normally be different from λ^* except in the trivial case where the normalization factor $\mathcal{N}(\lambda)$ is independent of λ. So maximization of the naive, unnormalized evidence leads to suboptimal results even in this ideal scenario (where unique true hyperparameter values are assumed to exist, and only optimality on average is asked for). It is thus difficult to justify theoretically. Nevertheless, it may of course still be useful as a heuristic procedure in some circumstances; see Opper and Winther (2000) and the discussion in Section 7.

So far, we have discussed the evidence for the complete data set D, consisting of the set of training inputs X and the set of training outputs (which we denote by Y from now on). To emphasize this fact, let us rewrite (15) as

$$P(X, Y) = Q(X, Y)/\mathcal{N} \tag{19}$$

where

$$Q(X, Y) = Q(Y \mid X) Q(X) \tag{20}$$

$$Q(Y \mid X) = \int d\theta\, Q(\theta) \prod_i Q(y_i \mid x_i, \theta) \tag{21}$$

$$Q(X) = \prod_i Q(x_i) \tag{22}$$

and I have dropped the explicit conditioning on the hyperparameters again. The evidence $P(X, Y)$ contains information both on the distribution of training inputs and on the training outputs. One could, however, take the view that for the purposes of classification the training inputs X should be regarded as fixed, and thus consider the evidence $P(Y \mid X)$ for the training outputs Y, conditional on X. In models which specify only the conditional distribution of outputs given inputs, $P(X, Y)$ and $P(Y \mid X)$ are related by a trivial constant factor; but for our joint input-output model, this is not the case. To find $P(Y \mid X)$, we use $P(Y \mid X) = P(X, Y)/P(X)$ and work out $P(X)$ from (6):

$$P(X) = \frac{1}{\mathcal{N}} \sum_{\{y_i\}} \int d\theta\, Q(\theta) \prod_i Q(y_i \mid x_i, \theta) Q(x_i) = \frac{\mathcal{N}(X)}{\mathcal{N}} Q(X)$$

where

$$\mathcal{N}(X) = \int d\theta\, Q(\theta) \prod_i \nu(\theta(x_i)) \tag{23}$$

Using (19, 20), we thus have for the evidence conditioned on training inputs

$$P(Y \mid X) = \frac{P(X, Y)}{P(X)} = \frac{Q(X, Y)}{\mathcal{N}(X) Q(X)} = \frac{Q(Y \mid X)}{\mathcal{N}(X)} \tag{24}$$

and $\mathcal{N}(X)$ is recognized as the relevant conditional normalization factor.

What are the relative merits of the two types of evidence defined above? As explained previously, the conditional evidence $P(Y \mid X)$ regards the training inputs as fixed and only considers the likelihood of the training outputs. This is the quantity that is conventionally defined as the evidence (MacKay, 1992); it disregards all information about the input space. In our scenario, this is reflected in the fact that—as can be seen from Eqs. (21, 23, 24)—$P(Y \mid X)$ is independent of the assumed input distribution $Q(x)$ of the unnormalized probability model. The joint evidence $P(X, Y)$, on the other hand, also considers the likelihood of the observed training inputs. This appears desirable: As we saw, our normalized probability model makes implicit assumptions about the distribution of inputs (which should be found predominantly outside the gap regions defined by $|\theta(x)| < 1$), and the joint evidence should allow us to pick hyperparameters for which the structure of this assumed distribution matches the observed one. The drawback of the joint evidence is that it depends on the unknown distribution $Q(x)$, which also needs to be estimated. This is in principle a full density estimation problem (although simple proxies may be useful heuristically—see below), which one would like to avoid, in line with Vapnik's advice that "when solving a given problem, try to avoid solving a more general problem as an intermediate step" (Vapnik, 1995). From this point of view, then, the conditional evidence $P(Y \mid X)$ seems preferable to the joint evidence $P(X, Y)$.

Note that both types of evidence that we have defined in general depend on the inverse noise level C and the kernel $K(x, x')$ *separately*. This is in contrast to the conventional SVM solution: The latter is found by maximizing the log-posterior (13), and the position of this maximum clearly only depends on the product $CK(x, x')$. This is an important point:

It implies that properties of the conventional SVM alone—generalization error bounds, test error or cross-validation error, for example—can never be used to assign an unambiguous value to C. *Since C determines the class probabilities* (10), this also means that well-determined class probabilities for SVM predictions cannot be obtained in this way. The intuition behind this observation is simple: If C is varied while $CK(x, x')$ is kept fixed (which means changing the amplitude of the kernel in inverse proportion to C), then the position of the maximum of the posterior $P(\theta \mid D)$, i.e., the conventional SVM solution, remains unchanged. The shape of the posterior, on the other hand, does vary in a nontrivial way, being more peaked around the maximum for larger C; the evidence is sensitive to these changes in shape and so depends on C.[5]

I conclude this section by discussing a simple limit of SVM classification in which both the joint and the conditional evidence defined above reduce to the corresponding quantities for noise free Gaussian process classification. This limit consists of considering a sequence of kernel functions $K(x, x') = K_0 M(x, x')$ with amplitude $K_0 \to \infty$, while keeping C and the shape $M(x, x')$ of the kernel fixed. In this limit, one can replace $Q(y \mid x, \theta(x))$ by $\kappa(C)H(\theta(x))$ in all integrals over the prior $Q(\theta)$. This is because the typical values of the $\theta(x)$ in such integrals are proportional to $\sqrt{K_0}$ (remember that the kernel is equivalent to a covariance function), and because $\exp[-Cl(z\sqrt{K_0})] \to H(z)$ for $K_0 \to \infty$ at any fixed z. A similar argument shows that $\nu(\theta(x))$ can be replaced by the constant $\kappa(C)$. One thus finds for the conditional evidence

$$P(Y \mid X) = \int d\theta\, Q(\theta) \prod_i H(y_i \theta(x_i))$$

while the joint evidence only differs by a trivial factor, $P(X, Y) = P(Y \mid X)Q(X)$. These expressions are identical to those for noise free Gaussian process classification, as claimed; note that they are independent of the value of C. One can easily check that different results obtain in the limit $C \to \infty$ at constant K_0. The conventional SVM classifier, on the other hand, is the same in both limits, because both imply $CK_0 \to \infty$. This reinforces the point that the evidence is sensitive to C and K_0 separately while the conventional SVM is not.

5. Evaluating the evidence

We now turn to the question of how to evaluate or estimate the two types of evidence defined in the previous section, $P(Y \mid X)$ and $P(X, Y)$. Beginning with the former (the conditional evidence), we see from (24) that we need to find the naive conditional evidence $Q(Y \mid X)$ and the normalization factor $\mathcal{N}(X)$. The naive evidence (21), being an average of a product of n likelihood factors over the prior, is expected to be exponential in n; the same is true for the normalization factor $\mathcal{N}(X)$ and all other evidence-like quantities. I will thus consider the normalized logarithms of these quantities; Eq. (24) then becomes

$$E(Y \mid X) = E_{\mathrm{nv}}(Y \mid X) - \frac{1}{n} \ln \mathcal{N}(X), \tag{25}$$

where

$$E(Y \mid X) = \frac{1}{n} \ln P(Y \mid X), \qquad E_{\mathrm{nv}}(Y \mid X) = \frac{1}{n} \ln Q(Y \mid X)$$

and the subscript in E_{nv} indicates that this is the (normalized log-) naive evidence. Similarly, for the joint evidence,

$$E(X, Y) = E_{\mathrm{nv}}(Y \mid X) + E_{\mathrm{nv}}(X) - \frac{1}{n} \ln \mathcal{N} \tag{26}$$

For brevity, I will simply refer to $E(Y \mid X)$ and $E(X, Y)$ as conditional and joint evidence, dropping the attribute 'normalized log'.

To obtain $E(Y \mid X)$, we need to find $E_{\mathrm{nv}}(Y \mid X)$ and $\frac{1}{n} \ln \mathcal{N}(X)$. Both of these are averages (over the prior) of functions which only depend on the values $\theta(x_i) \equiv \theta_i$ of the function θ at the training inputs x_i. All other $\theta(x)$ can be integrated out directly; with the shorthands $\boldsymbol{\theta} = (\theta_1 \ldots \theta_n)$ and $d\boldsymbol{\theta} = \prod_i d\theta_i$, we can thus write

$$e^{nE_{\mathrm{nv}}(Y \mid X)} = \int d\boldsymbol{\theta} Q(\boldsymbol{\theta}) \prod_i Q(y_i \mid x_i, \theta_i) \tag{27}$$

$$\mathcal{N}(X) = \int d\boldsymbol{\theta} Q(\boldsymbol{\theta}) \prod_i \nu(\theta_i) \tag{28}$$

Here $Q(\boldsymbol{\theta})$ is the marginal prior distribution over the θ_i. This is an n-dimensional Gaussian distribution with zero means; its covariance matrix $\mathbf{K}$ (the 'Gram matrix') has the entries $K(x_i, x_j) \equiv K_{ij}$.

Turning now to the joint evidence $E(X, Y)$ as given by (26), we note that a simplification similar to (28) is a priori not possible for the calculation of $\mathcal{N}$, which from (7) is an average of a function of all $\theta(x)$. However, as noted above, the underlying input distribution $Q(x)$ is not normally known in practice. A sensible proxy is the empirically observed input distribution.[6] While this is expected to introduce some bias, the effect should be rather benign because $\mathcal{N}$ does not depend on training *outputs* (unlike, for example, the generalization error, for which a similar approximation—which yields the training error as an estimate—is normally disastrous). This leads to the approximation for $\mathcal{N}$

$$\hat{\mathcal{N}} = \int d\boldsymbol{\theta} Q(\boldsymbol{\theta}) \left(\frac{1}{n} \sum_i \nu(\theta_i) \right)^n \tag{29}$$

which now again only requires integration over the θ_i. Within this approximation, the overall naive likelihood of the set of training inputs becomes just a constant, $Q(X) = 1/n^n$. Dropping this from $E(X, Y)$, we thus arrive at the following proxy for the joint evidence

$$\hat{E}(X, Y) = E_{\mathrm{nv}}(Y \mid X) - \frac{1}{n} \ln \hat{\mathcal{N}} \tag{30}$$

in close correspondence with (25). One easily sees that both $E(Y \mid X)$ and $\hat{E}(X, Y)$ reduce to a sensible 'random guessing' baseline value of $-\ln 2$ in the limit $C \to 0$, where from (11) the model assumes that class labels are randomly distributed independently of the latent function $\theta(x)$. Both $E(Y \mid X)$ and $\hat{E}(X, Y)$ are also trivially bounded from below by the (conditional) naive evidence $E_{\mathrm{nv}}(Y \mid X)$; this follows from the fact that the normalization factors obey $\mathcal{N}(X) \leq 1$ and $\hat{\mathcal{N}} \leq 1$ (because $\nu(\theta(x)) \leq 1$).

As they stand, the expressions (27, 28, 29) for the naive evidence and the required normalization factors are still n-dimensional integrals and therefore in general intractable. I therefore next discuss a simple analytical approximation for the naive evidence, before moving on to methods for obtaining numerical estimates. Writing out the integral for $Q(Y \mid X)$ explicitly, one has

$$Q(Y \mid X) = \kappa^n(C) \int \frac{d\boldsymbol{\theta}}{\sqrt{\det(2\pi\mathbf{K})}} \exp\left[-\frac{1}{2}\sum_{ij}\theta_i(\mathbf{K}^{-1})_{ij}\theta_j - C\sum_i l(y_i\theta_i)\right]$$

To approximate the integral, one can expand the exponent around its maximum θ_i^*, which just corresponds to the conventional SVM. For all θ_i corresponding to non-marginal training inputs (i.e., for which $y_i\theta_i^* \neq 1$), this is unproblematic; the expansion around the maximum is quadratic to leading order, and one has a classical Laplace approximation. For the θ_i corresponding to marginal training inputs, on the other hand, the exponent has a 'kink' at its maximum (the partial derivative $\partial/\partial\theta_i$ has a discontinuity there), and so linear terms have to be taken into account as well. This leads to

$$\begin{aligned} Q(Y \mid X) \approx{}& \kappa^n(C)\exp\left[-\frac{1}{2}\sum_{ij}\theta_i^*(\mathbf{K}^{-1})_{ij}\theta_j^* - C\sum_i l(y_i\theta_i^*)\right] \\ &\times \int \frac{d\boldsymbol{\theta}}{\sqrt{\det(2\pi\mathbf{K})}} \exp\Bigg[-\frac{1}{2}\sum_{ij}\Delta\theta_i(\mathbf{K}^{-1})_{ij}\Delta\theta_j \\ &\quad - \sum_i{}' \Delta\theta_i \sum_j (\mathbf{K}^{-1})_{ij}\theta_j^* - C\sum_i{}'(-y_i\Delta\theta_i)H(-y_i\Delta\theta_i)\Bigg] \end{aligned}$$

where the primed sums run over the marginal training inputs only, and $\Delta\theta_i = \theta_i - \theta_i^*$ are the deviations in the θ_i from the maximum. We denote the prefactor of the integral by $Q^*(Y \mid X)$. It can be regarded as a zeroth-order approximation to the evidence, obtained by assuming that the posterior is Gaussian with mean at the true MAP and curvature induced by the prior only. (In other words, $Q^*(Y \mid X)$ neglects all contributions of the log-likelihood to the *shape* of the posterior, retaining only its effect on the *position* of the posterior maximum). The integration over all the non-marginal $\Delta\theta_i$ is straightforward (a simple Gaussian marginalization), so that

$$\begin{aligned} Q(Y \mid X) \approx{}& Q^*(Y \mid X) \int \frac{d\boldsymbol{\theta}'}{\sqrt{\det(2\pi\mathbf{K}_m)}} \exp\Bigg[-\frac{1}{2}\sum_{ij}{}'\Delta\theta_i(\mathbf{K}_m)^{-1}_{ij}\Delta\theta_j \\ &- \sum_i{}'\sum_j \Delta\theta_i(\mathbf{K}^{-1})_{ij}\theta_j^* - C\sum_i{}'(-y_i\Delta\theta_i)H(-y_i\Delta\theta_i)\Bigg] \end{aligned}$$

where $\mathbf{K}_m$ is the submatrix of the Gram matrix corresponding to the marginal inputs. Only an integration over the θ_i for marginal inputs (indicated by $d\boldsymbol{\theta}'$) is now left, and the leading order terms in $\Delta\theta_i$ in the exponent are linear. Discarding the quadratic terms as sub-dominant, the integral factorizes and can be evaluated explicitly. Noting also that, from the form (14) of the conventional SVM, $\sum_j (\mathbf{K}^{-1})_{ij}\theta_j^* = y_i\alpha_i$, one finds in this way

$$Q(Y \mid X) \approx Q^*(Y \mid X)\frac{1}{\sqrt{\det(2\pi\mathbf{K}_m)}}\prod_i{}'\left(\frac{1}{C-\alpha_i}+\frac{1}{\alpha_i}\right)$$

This can be written in a more compact form by defining $\mathbf{L}_m$ as a diagonal matrix (of size equal to the number of marginal inputs) with entries $2\pi[\alpha_i(C-\alpha_i)/C]^2$. Taking logs and normalizing, this gives

$$E_{\text{nv}}(Y \mid X) \approx \frac{1}{n}\ln Q^*(Y \mid X) - \frac{1}{2n}\ln\det(\mathbf{L}_m\mathbf{K}_m) \tag{31}$$

Note that the second term only contains information about the *marginal* training inputs, through the matrices $\mathbf{L}_m$ and $\mathbf{K}_m$. Given the sparseness of the SVM solution, these matrices should be reasonably small, making their determinants amenable to numerical computation or estimation (Williams, 1998). Equation (31) diverges when $\alpha_i \to 0$ or $\to C$ for one of the marginal training inputs; the approximation of retaining only linear terms in the log integrand then breaks down. I therefore adopt the simple heuristic of replacing $\det(\mathbf{L}_m\mathbf{K}_m)$ by $\det(\mathbf{I}+\mathbf{L}_m\mathbf{K}_m)$, which prevents these spurious singularities (where $\mathbf{I}$ is the identity matrix). It is easy to show that this has the added benefit of producing an approximation which is continuous when training inputs move in or out of the set of marginal inputs as hyperparameters are varied. In summary, the final generalized Laplace approximation to the naive evidence is

$$E_{\text{nv}}(Y \mid X) \approx \frac{1}{n}\ln Q^*(Y \mid X) - \frac{1}{2n}\ln\det(\mathbf{I}+\mathbf{L}_m\mathbf{K}_m) \tag{32}$$

Here the first term can be rewritten as (using the form of the SVM solution (14))

$$\frac{1}{n}\ln Q^*(Y \mid X) = -\frac{1}{2n}\sum_i y_i\alpha_i\theta_i^* - \frac{C}{n}\sum_i l(y_i\theta_i^*) + \ln\kappa(C)$$

to simplify the numerical evaluation. Note that the final form (32) of the approximation actually has quite a simple interpretation: It can be seen as the result of an ordinary Laplace approximation (MacKay, 1992) which estimates the contribution of the log-likelihood to the Hessian (curvature matrix) of the log-posterior at its maximum by the matrix $\mathbf{L}_m$.

For the normalization factors $\mathcal{N}(X)$ and $\hat{\mathcal{N}}$, no similar analytical approximation is possible: In contrast to the naive evidence, the integrals (28, 29) defining them are in general multimodal, ruling out a Laplace-like approximation. I will therefore estimate these numerically; also for the naive evidence itself it is useful to have numerical estimates to compare with the above approximation. All three quantities $Q(Y \mid X)$, $\mathcal{N}(X)$ and $\hat{\mathcal{N}}$ are defined by averages over the prior distribution $Q(\boldsymbol{\theta})$ of the θ_i. But direct sampling from the prior would produce estimates with extremely large fluctuations. To see this, consider for example the naive evidence. The integral (21) defining it has as its integrand simply an unnormalized version of the posterior. In general, the latter has most of its 'mass' in a very different region of $\boldsymbol{\theta}$-space than the prior; sampling from the prior will therefore normally take a large (exponential in n) number of samples before capturing most of this mass. Instead, I use Monte Carlo chaining (Neal, 1993; Barber & Bishop, 1997). The idea is to construct a chain $k = 0 \cdots m$ of distributions $R_k(\boldsymbol{\theta}) \propto Q(\boldsymbol{\theta}) w_k(\boldsymbol{\theta})$ defined by weight functions $w_k(\boldsymbol{\theta})$. If $w_0(\boldsymbol{\theta}) = 1$ and $w_m(\boldsymbol{\theta}) = \prod_i Q(y_i \mid x_i, \theta_i)$, so that the chain of distributions interpolates between the prior and the posterior, then the naive evidence can be rewritten as

$$
\begin{aligned}
Q(Y \mid X) &= \int d\boldsymbol{\theta} Q(\boldsymbol{\theta}) \prod_i Q(y_i \mid x_i, \theta_i) = \prod_{k=0}^{m-1} \frac{\int d\boldsymbol{\theta} Q(\boldsymbol{\theta}) w_{k+1}(\boldsymbol{\theta})}{\int d\boldsymbol{\theta} Q(\boldsymbol{\theta}) w_k(\boldsymbol{\theta})} \\
&= \prod_{k=0}^{m-1} \left\langle \frac{w_{k+1}(\boldsymbol{\theta})}{w_k(\boldsymbol{\theta})} \right\rangle_{R_k(\boldsymbol{\theta})}
\end{aligned}
$$

This is simply a product of averages over the interpolating distributions; if successive weight functions are not too dissimilar, these averages can be estimated reliably. A natural choice for the weight functions in our case is

$$
w_k(\boldsymbol{\theta}) = \prod_{i=1}^{ka} Q(y_i \mid x_i, \theta_i)
$$

where a is some integer that divides n and the chain has $m = n/a$ links; this means that at each link in the chain a further a training examples are included in the calculation of the evidence. A similar approach can be applied to the estimation of $\mathcal{N}(X)$ and $\hat{\mathcal{N}}$; for these, I chose $w_k(\boldsymbol{\theta}) = \prod_{i=1}^{ka} \nu(\theta_i)$ and $w_k(\boldsymbol{\theta}) = [(1/n) \sum_i \nu(\theta_i)]^{ka}$, respectively.

For each link in the chain, the actual sampling from the interpolating distributions was done by a Monte Carlo method (Neal, 1993). Each Monte Carlo run requires a number of steps; at each step, a change in the θ_i is proposed and accepted or rejected depending on the relative probabilities of the old and new values. This requires evaluation of the prior $Q(\theta)$ and the weight function $w_k(\boldsymbol{\theta})$ at each step. For small data sets, to simplify the evaluation of the prior, one can represent the θ_i as $\boldsymbol{\theta} = \mathbf{M}\boldsymbol{z}$ in terms a vector $\boldsymbol{z}$ of independent unit variance Gaussian random variables z_i. The prior is then simply $\propto \exp(-\|z\|^2/2)$. The matrix $\mathbf{M}$ has to obey $\mathbf{M}\mathbf{M}^{\mathbf{T}} = \mathbf{K}$ and can be found from the eigenvalues and eigenvectors of $\mathbf{K}$. (If the eigenvalues of $\mathbf{K}$ decay sufficiently fast, one can also discard the z_i corresponding to negligibly small eigenvalues to speed up the method). Random Gaussian changes in the z_i are then suitable proposals for the Monte Carlo method; the change in the prior is trivial to evaluate, while for the likelihood the changes in the θ_i also need to be found from $\boldsymbol{\theta} = \mathbf{M}\boldsymbol{z}$.

This method (used in the toy application below) becomes expensive for large data sets; already the required initial calculation of the eigenvalues and eigenvectors of $\mathbf{K}$ becomes infeasible for large n. One may then think of proposing Monte Carlo steps directly in terms of the θ_i. However, the evaluation of the prior probability

$$Q(\boldsymbol{\theta}) \propto \exp\left[-\frac{1}{2}\sum_{ij} \theta_i (\mathbf{K}^{-1})_{ij}\theta_j \right]$$

then still requires the inversion of the matrix $\mathbf{K}$, an $O(n^3)$ process which one would like to avoid. A better option is to express the θ_i as

$$\theta_i = \sum_j K_{ij} y_j \beta_j$$

where the β_i generalize the variables α_i specifying the conventional SVM solution. The advantage is that the prior in terms of the new variables β_i is $\propto \exp(-\frac{1}{2}\sum_{ij}\beta_i K_{ij}\beta_j)$, no longer requiring a matrix inversion.[7] If at each step, a Gaussian change in a randomly chosen β_j is proposed, the change in the prior probability can readily be found with $O(1)$ operations, and the same is true for the changes in the θ_i. A maximum of n (all of the θ_i) updates need to be made at each update, but when the kernel elements K_{ij} between sufficiently distant input points x_i and x_j are negligibly small, the number of updates may be rather less than n. A full sweep through all the α_i would then take between $O(n)$ and $O(n^2)$ operations. With a number of links in the chaining method of $O(n)$, one thus estimates a total of $O(zn^2)$ to $O(zn^3)$ operations for estimating the naive evidence and the normalization factors, where z is the number of sweeps through the α_i required at each link in the chain to get a reliable estimate. In some preliminary experiments, I found that z was of the order of 10^3 or larger, so the overall procedure is still relatively computationally intensive; work is in progress to improve this.

Once the joint $[\hat{E}(X, Y)]$ or conditional $[E(Y \mid X)]$ evidence has been found by the above method—where for the contribution from the naive evidence one can use either the generalized Laplace approximation or a numerical estimate—it can be used to set hyperparameter values. An exhaustive search over hyperparameter values will generally be too expensive, so a greedy search technique could be employed instead. More elegantly, one could try to estimate gradients of the evidence rather than its absolute values. As is well known from applications of Monte Carlo methods in statistical physics (Neal, 1993), such gradients are much cheaper to evaluate because they do not require the use of chaining methods. For the gradient of the naive evidence, for example, one finds that gradients with respect to the hyperparameters can be found as simple averages over the posterior distribution (which can be estimated using a single Monte Carlo run). The same is true for $\mathcal{N}(X)$ and $\hat{\mathcal{N}}$, with averages over distributions proportional to $\prod_i \nu(\theta_i)Q(\boldsymbol{\theta})$ and $[(1/n)\sum_i \nu(\theta_i)]^n Q(\boldsymbol{\theta})$, respectively. This therefore seems a promising direction for the numerical implementation of evidence maximization for SVMs, which I leave for future work.

6. Predictive class probabilities

We now turn to the issue of how to define class probabilities for SVM predictions. From the Bayesian point of view, the procedure is clear: One averages the required class probability

$P(y \mid x, \theta(x))$ for a test input x over the posterior distribution of the latent function value $\theta(x)$:

$$P(y \mid x, D) = \int d\theta(x) P(y \mid x, \theta(x)) P(\theta(x) \mid D) \tag{33}$$

To find the posterior distribution of $\theta(x)$, one writes

$$P(\theta(x) \mid D) = \int d\boldsymbol{\theta} P(\boldsymbol{\theta}, \theta(x) \mid D) \tag{34}$$

Now, by construction of the model (6), the posterior probability $P(\boldsymbol{\theta}, \theta(x) \mid D)$ is equal to that obtained from the unnormalized model,

$$P(\boldsymbol{\theta}, \theta(x) \mid D) = Q(\boldsymbol{\theta}, \theta(x) \mid D) \propto Q(\boldsymbol{\theta}, \theta(x)) \prod_i Q(y_i \mid x_i, \theta_i) \tag{35}$$

So one can obtain samples from the posterior distribution of $\theta(x)$ by sampling from the joint posterior of $\boldsymbol{\theta} = (\theta_1 \cdots \theta_n)$ and $\theta(x)$ and then simply dropping the $\boldsymbol{\theta}$ components. The sampling from $Q(\boldsymbol{\theta}, \theta(x) \mid D)$ can be done by a Monte Carlo procedure exactly analogous to that described above for the posterior $Q(\boldsymbol{\theta} \mid D)$; one only has to bear in mind that there is no likelihood factor for $\theta(x)$ because the corresponding output has not been observed.

In practice, however, running a Monte Carlo estimate for each prediction to be carried out is likely to be too computationally expensive. A cheaper alternative would be to approximate (33) by evaluating the class probabilities at the posterior mean of $\theta(x)$, rather than averaging the probabilities themselves:

$$P(y \mid x, D) \approx P(y \mid x, \bar{\theta}(x)) \tag{36}$$

where

$$\bar{\theta}(x) = \int d\theta(x) \theta(x) P(\theta(x) \mid D)$$

Using (34, 35) one can write

$$P(\theta(x) \mid D) = \int d\boldsymbol{\theta} Q(\theta(x) \mid \boldsymbol{\theta}) Q(\boldsymbol{\theta} \mid D)$$

Now the conditional prior $Q(\theta(x) \mid \boldsymbol{\theta})$ of $\theta(x)$ is Gaussian, with mean $\sum_{ij} K(x, x_i)(\mathbf{K}^{-1})_{ij}$ θ_j; this follows from the fact that the joint prior of $\theta(x)$ and $\boldsymbol{\theta}$ is Gaussian. Writing the $\theta_i = \sum_j K_{ij} y_j \beta_j$ in terms of the auxiliary variables β_j introduced in the previous section, this mean can be expressed as $\sum_i y_i \beta_i K(x, x_i)$. The posterior average of $\theta(x)$ is therefore simply a linear combination of the posterior averages of the β_j:

$$\bar{\theta}(x) = \sum_i y_i \bar{\beta}_i K(x, x_i)$$

This approach to estimating class probabilities thus only requires that the posterior averages of the $\bar{\beta}_i$ be evaluated once and for all; this can be done by Monte Carlo sampling as explained in the previous section. If even this is considered to be expensive, one could approximate the class probabilities even further by simply evaluating them at the maximum of the posterior, i.e., the conventional SVM (14):

$$P(y \mid x, D) \approx P(y \mid x, \theta^*(x)) \tag{37}$$

We thus have three possibilities for evaluating class probabilities, involving decreasing computational effort as one moves from (33) to (36) to (37). The toy application in the next section will give us a chance to compare them.

7. A toy application

In this section, I illustrate the evaluation of the evidence and of class probabilities using a simple toy application. I decided to generate synthetic data from the probability model described in Section 2, rather than use a standard benchmark data set. This has the advantage that both the true values of the hyperparameters and the true class probabilities are known, and so the performance of the evidence in locating these true hyperparameters and the quality of the various approximations to the class probabilities can be analysed in detail.

As the input space I chose the unit interval [0, 1] with a uniform input distribution $Q(x)$. The inverse noise level C was set to $C^* = 2$, and the true kernel was of RBF form, $K^*(x, x') = K_0^* \exp\{-(x - x')^2/[2(l^*)^2]\}$ with amplitude $K_0^* = 1.25$ and length scale $l^* = 0.05$. With these parameters defined, I chose—by sampling from the probability model (6)—a target latent function $\theta(x)$ and a data set of $n = 50$ training examples (see figure 4). This data set was then fed into a standard SVM classification algorithm; the kernel of the classifier was chosen to have the correct shape (RBF), but with unknown amplitude K_0 and length scale l; the parameter C was also left free. In figure 5, the results for the evidence are shown as a function of l and K_0, for C fixed to the true value $C^* = 2$; other choices of C lead to qualitatively similar results. The left and right columns of the figure differ only in the way the contribution from the naive evidence was calculated (either by Monte Carlo chaining, or from the generalized Laplace approximation (32)). The shapes of the resulting surfaces are quite similar (although the absolute values of the evidence are not), at least for K_0 not too large. While this could suggest that the—computationally much cheaper—generalized Laplace approximation may be useful for optimizing hyperparameters, preliminary results on benchmark data sets (discussed below) with a larger number of hyperparameters show that this is probably not so: The evidence as estimated by the Laplace approximation is a rather "rough" function of the hyperparameters as soon as there are several of them, and has a number of local optima among which the global optimum is difficult to locate.

Comparing the three rows of the figure, we also see that the naive (unnormalized) evidence as well as the joint and conditional evidence of the normalized probability model all have their maxima at values of l close to the truth ($l^* = 0.05$). For determining this hyperparameter, therefore, the normalization of the probability model does not seem to be crucial. However, if we plot the evidence as a function of C and K_0 (having fixed l to its true

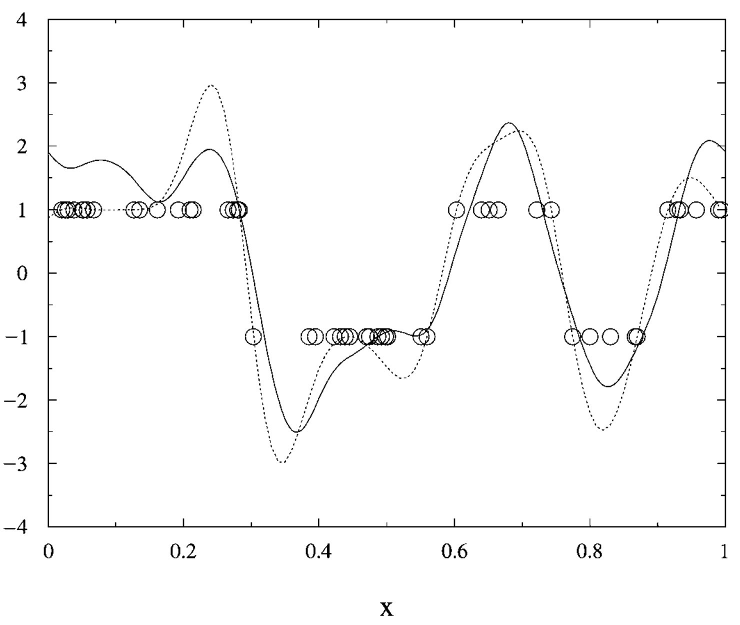

Figure 4. The target latent function $\theta(x)$ for the synthetic data set (solid line), and the $n = 50$ training examples with their output labels (circles). The dotted line shows the 'noise-free' SVM fit ($CK_0 \to \infty$) with an RBF kernel of lengthscale $l = 0.05$.

value $l^* = 0.05$), the importance of normalization becomes apparent (figure 6): The naive evidence is maximized for a value of C significantly smaller than the true one, $C^* = 2$; along the line $CK_0 = 2.5$, for example (corresponding to $\log_{10} CK_0 \approx 0.398$) the maximum is attained at $C \approx 1.2$. Both the joint and the conditional evidence of the normalized model, on the other hand, are maximized rather closer to the true value of C (at $C \approx 1.9$ along $CK_0 = 2.5$).

So far, we have seen that maximization of the evidence (joint or conditional) of the normalized probability model gives values of l and C close to the true values with which the training data were generated. With respect to K_0, on the other hand, figure 6 suggests that the optimal choice is in fact to make K_0 arbitrarily large. As explained at the end of Section 4, in this limit we essentially recover a deterministic Gaussian process classifier, and the evidence appears to prefer this noise-free interpretation of the data. This may seem puzzling at first, given that the model generating the data had finite values of C and K_0 and thus did not produce deterministic outputs. However, a look back at the training data in figure 4 shows that there is no 'visible' corruption in this particular data set, and the SVM solution in the noise free limit ($CK_0 \to \infty$), shown as the dotted line, actually looks rather plausible. This emphasizes that the theoretical justification for evidence maximization given in Section 4 is based on an average over all possible data sets of a given size; there is no absolute guarantee that for any *particular* data set the evidence maximum will be located

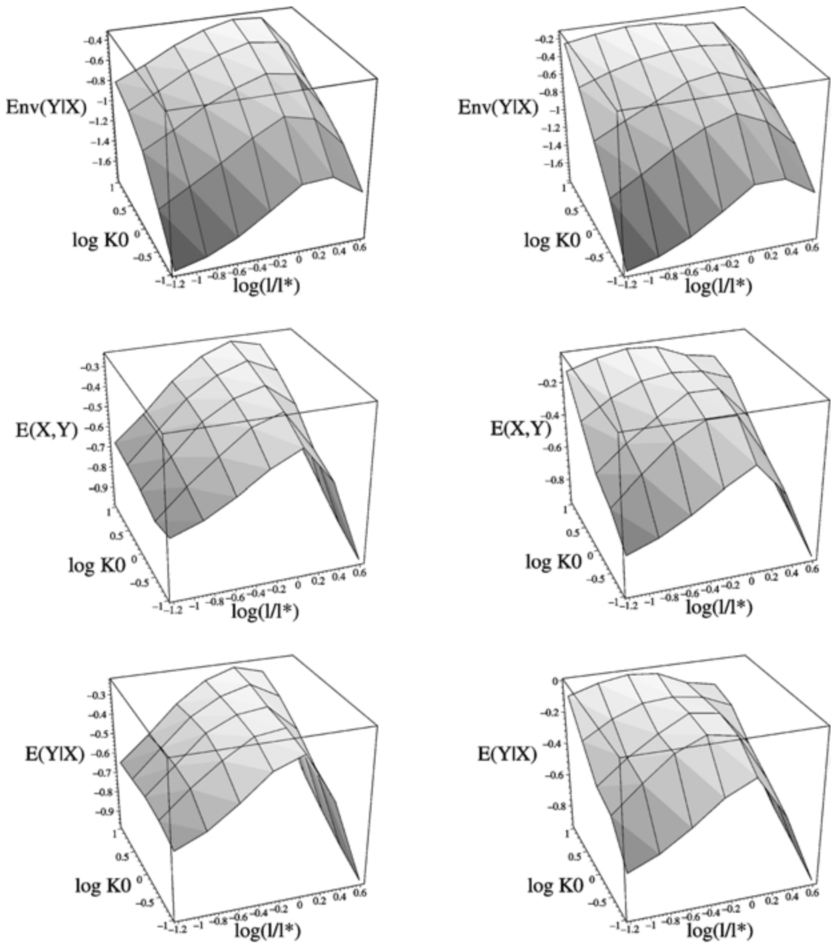

Figure 5. The evidence for an SVM with an RBF kernel trained on the data set of figure 4, as a function of the lengthscale l and the amplitude K_0 of the kernel; $C = 2$ was kept fixed. The three rows show the naive (unnormalized) evidence $E_{\mathrm{nv}}(Y \mid X)$ and the joint $[\hat{E}(X, Y)]$ and conditional $[E(Y \mid X)]$ evidence of the normalized model. The two columns differ in how the naive evidence was evaluated: By Monte Carlo chaining on the left and from the generalized Laplace approximation (32) on the right. The normalization factors $\hat{\mathcal{N}}$ and $\mathcal{N}(X)$ contributing to $\hat{E}(X, Y)$ and $E(Y \mid X)$, respectively, were always estimated by Monte Carlo chaining. Note that the lengthscale is displayed as $\log_{10}(l/l^*)$, which equals zero when l equals the true value $l^* = 0.05$. The K_0-scale is also logarithmic.

close to the true hyperparameter values, certainly when n is small.[8] To further illustrate this point, I show in figure 7 a second sample from the same model as considered above, and the evidence as a function of K_0 for $C = 2$ and $l = 0.05$. Consistent with the fact that this second data set has one 'visibly' corrupted training output (for training input $x_i \approx 0.37$), the evidence now has a maximum at a finite K_0.

It should be stressed that the observed lack of an evidence maximum at finite K_0 for the first data set (figure 4) is not peculiar to SVM classification with its somewhat subtle

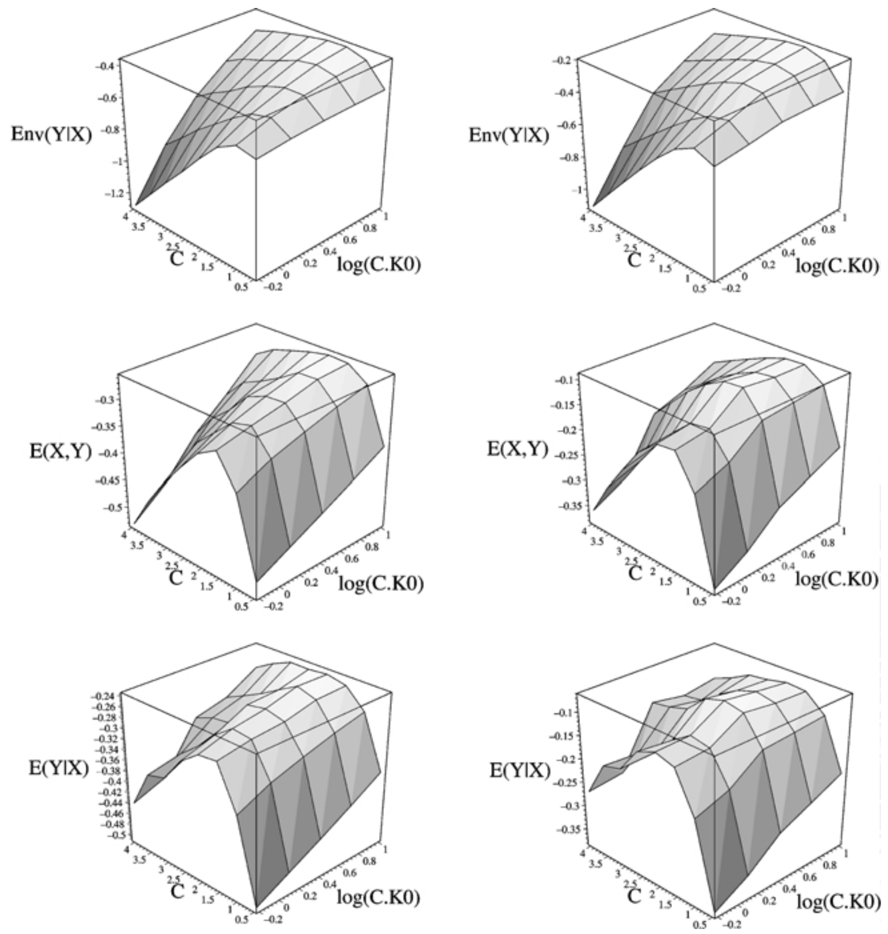

Figure 6. The evidence for an SVM with an RBF kernel trained on the data set of figure 4, as a function of the noise parameter C and the kernel amplitude K_0; the lengthscale $l = 0.05$ was kept fixed. The rows and columns show the same quantities as in figure 5. Along the K_0 axis, $\log_{10} CK_0$ is shown rather than just $\log_{10} K_0$, because a constant value of CK_0 corresponds to the same conventional SVM (maximum a posteriori) solution, independently of C. Note that the same is not true of the evidence.

normalization of the probability model. To demonstrate this, I show in figure 8 the evidence for a Gaussian process classifier with RBF kernel (with $l = 0.05$ and $C = 2$) as a function of K_0; just as for SVM classification, this has no maximum at finite K_0.

Finally I consider the estimation of predictive class probabilities for the data set of figure 4. Three approaches to this problem were discussed in the previous section: a full average (33) of the class probabilities over the posterior; evaluation at the posterior average $\bar{\theta}(x)$ of the latent function $\theta(x)$ (see (36)); and evaluation (37) at the maximum θ^* of the posterior, the conventional SVM solution. Figure 9 compares the results of these methods to each other and to the true class probabilities corresponding to the target function shown in

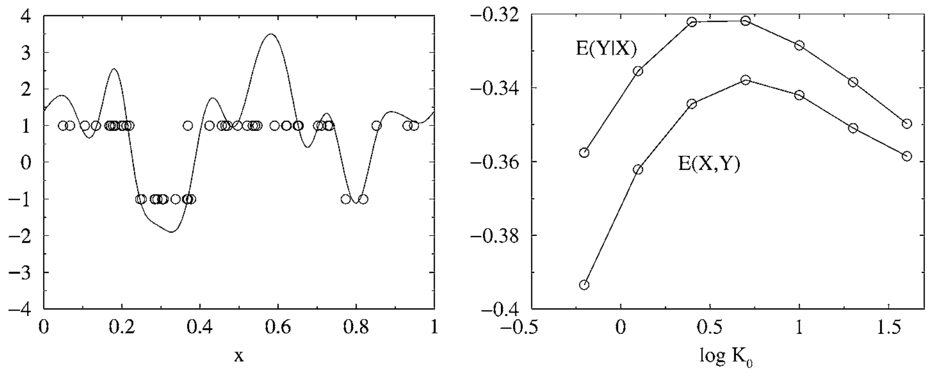

Figure 7. Left: A second sample (target latent function $\theta(x)$ and $n = 50$ training examples) from the same probability model used to generate figure 4. Note the 'visibly' corrupted training output (for training input $x_i \approx 0.37$). Consistent with this, both the joint and the conditional evidence (right) now show a maximum at finite rather than infinite kernel amplitude K_0. (The kernel lengthscale $l = 0.05$ and the noise parameter $C = 2$ were kept fixed to their true values; the naive evidence was evaluated by Monte Carlo chaining).

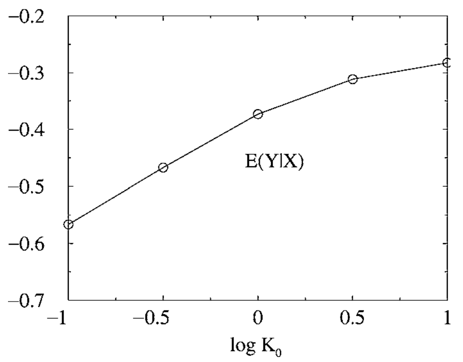

Figure 8. The evidence for a Gaussian process classifier [corresponding to a logistic loss function $(1/C)\ln(1 + e^{-Cz})$ rather than the SVM hinge loss $l(z)$] trained on the data set of figure 4, with an RBF kernel of lengthscale $l = 0.05$, as a function of kernel amplitude K_0. Note that just as for the SVM classifier (see figure 6), the evidence increases with K_0 and appears to have no maximum at finite K_0. (The value of C was chosen to be $C = 2$, as in figure 6. This is largely immaterial here because the evidence for a GP classifier depends on K_0 and C only through the combination $C\sqrt{K_0}$; see footnote 5. The results shown were obtained by Monte Carlo chaining. Note that because the GP classification model is automatically normalized, all three definitions of the evidence (naive, and joint/conditional) trivially coincide here).

figure 4. The classifier considered had all its hyperparameters set to their true values ($C = 2$, $l = 0.05$, $K_0 = 1.25$). As expected, the full posterior average gives the most 'moderated' class probabilities, moving away most quickly from confident predictions (probabilities close to 0 or 1) at the edges of the decision regions. The first approximation, evaluating class probabilities at the posterior average $\bar{\theta}(x)$ of the latent function, seems to lead to rather too over-confident predictions. The cheapest approximation—evaluating class probabilities at the SVM solution—on the other hand, provides quite a reasonable estimate of the class

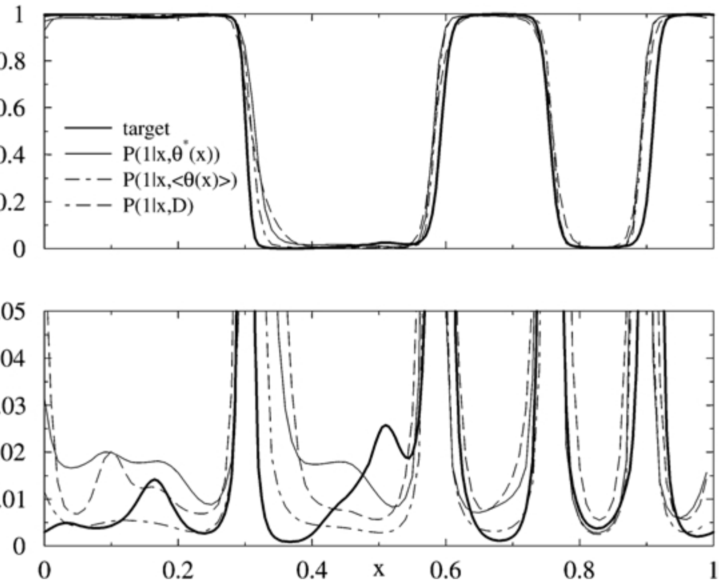

Figure 9. Class probabilities for the SVM classifier trained on the data set of figure 4, with hyperparameter values $C = 2$, $l = 0.05$, $K_0 = 1.25$. Shown are the probabilities for the class corresponding to output $y = 1$, evaluated at the maximum $\theta^*(x)$ of the posterior (the conventional SVM, thin solid line), the posterior average $\bar{\theta}(x)$ of the latent function (dot-dashed line), and by a full average over the posterior distribution of $\theta(x)$ (dashed line). The full solid line shows the true class probabilities generated by the target in figure 4. In the bottom half, $\min(P(y = 1), 1 - P(y = 1))$ is shown to allow a close-up up of the various predictions for class probabilities close to 0 or 1.

probabilities calculated from the full posterior average. This observation can be understood as follows: As shown in figure 10, the posterior average $\bar{\theta}(x)$ is generally larger in modulus than the MAP value $\theta^*(x)$. This suggests that the posterior $P(\theta(x) \mid D)$ is generally skewed (rather than symmetric) around its maximum, with more of its mass towards larger values of $|\theta(x)|$; histograms from the Monte Carlo simulations confirm this. Class probabilities evaluated at $\bar{\theta}(x)$ neglect fluctuations of $\theta(x)$ into the region of small $|\theta(x)|$ where the class probabilities are close to 0.5, and thus give over-confident predictions relative to the full posterior average. The further approximation of replacing the posterior mean by the SVM solution $\theta^*(x)$ then partially compensates for this because the modulus of $\theta^*(x)$ is generally smaller than that of $\bar{\theta}(x)$; in this sense, the fact that the SVM solution does not represent the posterior mean well may actually be beneficial. The above conclusions about the relative merits of the various methods of determining class probabilities should, however, be regarded as tentative until they have been tested on a wide range of real-world and synthetic data sets; work in this direction is in progress (Gold & Sollich, 2001).

In conclusion of this section, I would like to point out that the toy application presented above must be regarded as a rather ‘benign’ test of the probabilistic framework that I have described, because it used data generated from precisely the ‘right’ kind of probabilistic

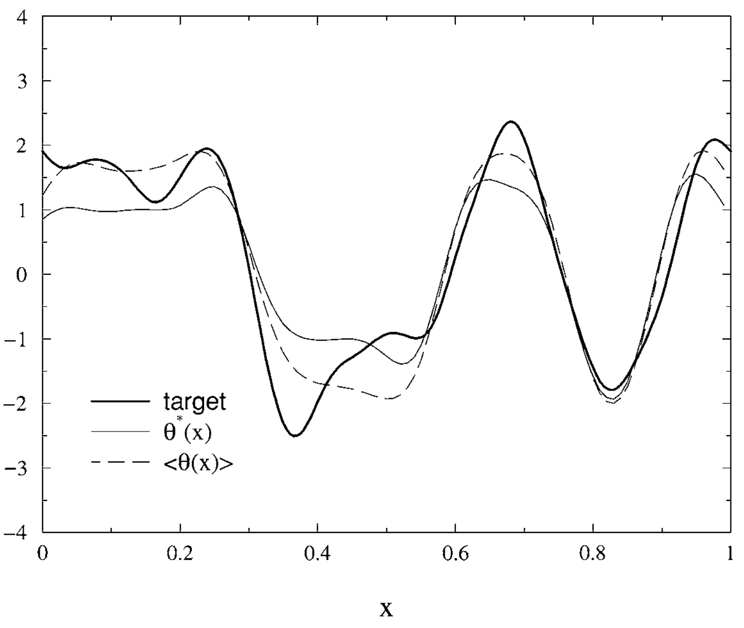

Figure 10. Comparison of the conventional SVM solution $\theta^*(x)$ and the posterior average $\bar{\theta}(x)$ of the latent function, for the SVM classifier trained on the data set of figure 4, with hyperparameter values $C = 2$, $l = 0.05$, $K_0 = 1.25$. Note that $\bar{\theta}(x)$ (dashed line) is generally larger in modulus than the MAP value $\theta^*(x)$ (thin solid line); this is because the posterior $P(\theta(x) \mid D)$ is generally skewed, with more of its mass towards larger values of $|\theta(x)|$.

model. As stated earlier, the motivation for this was to have true values for the hyperparameters available for comparison with those that are obtained by maximizing the evidence. Similarly, knowing the probability model that generated the data allowed us to compare two approximations for the predictive class probabilities to their optimal Bayesian counterpart for the given data set and to the true class probabilities. While I do believe that this approach yields some information about the probabilistic framework and its usefulness, it clearly leaves open a number of open questions. One would like to know, for example, how hyperparameter tuning by evidence maximization performs on real-world data sets, which are obviously not generated from the kind of probabilistic model that I have described. No 'true' hyperparameter values then exist, and one would instead primarily be interested in whether the hyperparameters that optimize the evidence actually yield competitive generalization performance. In this context, a systematic comparison with other methods for hyperparameter tuning for SVMs (see introduction) would also have to be performed.

It is clear that the above questions go rather beyond the scope of the present paper. However, work in this direction is in progress (Gold & Sollich, 2001). On the benchmark Pima and Crabs data sets (Ripley, 1996), for example, we have used evidence maximization methods to determine the lengthscales l_i (one per input component) of an RBF kernel. The generalized Laplace approximation yielded a rather 'rough' evidence surface as a function of the l_i, which made numerical optimization difficult; the Laplace evidence values also showed only limited correlation with the generalization error. (Wahba's generalized approximate cross validation error (Wahba, 1998) had rather similar properties). Gradient ascent on the (naive) evidence, on the other hand, using Monte Carlo estimates of the relevant gradients, yielded robust (initialization independent) values of the hyperparameters. Together with an optimally chosen value of C, the resulting generalization performance for the Pima data set, for example, was 18.9%, which is among the best results reported in the literature (see e.g. Barber & Williams, 1997; Opper & Winther, 2000; Seeger, 2000). Details of these and other results will be given in a forthcoming publication (Gold & Sollich, 2001).

8. Conclusion

In summary, I have described a probabilistic framework for SVM classification. It gives an intuitive understanding of the effect of the kernel, which is seen to correspond to the covariance function of a Gaussian process prior. More importantly, it also allows a properly normalized evidence to be defined; from this, optimal values of hyperparameters such as the noise parameter C, and corresponding class probabilities, can be derived.

The toy application shown in Section 7 suggested that evidence maximization can be a useful procedure for finding optimal hyperparameter values; synthetic data sets were considered to ensure that 'true' values of the hyperparameters were known. In the case of the one hyperparameter (the kernel amplitude K_0) where the maximum was far from the true value used for generating the data set, the preference of the evidence for a noise-free interpretation was actually quite plausible. For finding the correct value of C, the normalization of the probability model turned out to be important: the unnormalized (naive) evidence tended to underestimate C. Amongst the two possible definitions (joint or conditional) of the evidence for the *normalized* model, no clear preference emerged from the example: Both had very similar dependences on the hyperparameters. In the estimation of class probabilities, the most straightforward approach—evaluation at the conventional SVM solution—appeared to provide a reasonable approximation to the (rather more costly) full average over the posterior which a Bayesian treatment in principle requires.

It is appropriate at this point to compare the approach described above to other recent work on probabilistic approaches to SVMs. As pointed out in the introduction, the issue of normalization of the probability model has generally been disregarded up to now; as such, all existing work that I am aware of deals essentially with what I have called the naive evidence. Opper and Winther (Opper & Winther, 2000; Csató et al., 2000) have derived a number of elegant approximations and bounds for this quantity, using techniques such as the cavity method borrowed from statistical mechanics. They found encouraging performance results on some benchmark data sets by using these approximations to set hyperparameter values. Seeger (Seeger, 2000) used a Gaussian variational approach to estimate the naive evidence, which again seemed to perform well in practice. Finally, Kwok (Kwok, 1999a) used a Laplace approximation to approximate the naive evidence. His approach relied, however, on a rather ad-hoc smoothing of the hinge loss (by replacing the Heaviside step function $H(1-z)$ in the definition (2) with a sigmoid $1/[1+e^{-\eta(1-z)}]$, with some arbitrary η). The evaluation of the evidence also requires the calculation of determinants of large matrices (of size $n \times n$, or $d \times d$ if the kernel only has a finite number d of nonzero eigenvalues and $n > d$). The same approach can be used to obtain a Gaussian approximation of the posterior distribution $P(\theta(x) \mid D)$ required for the calculation of class probabilities. Note, however, that in (Kwok, 1999a) it was proposed to average the unnormalized class probabilities $Q(y \mid x, \theta(x))$ over this distribution and then normalize; from (33), the approach of averaging the normalized class probabilities appears to have a better theoretical foundation.

All three approaches reviewed so far share with mine the idea of treating the conventional SVM as the maximum a posteriori solution to an inference problem. In Herbrich, Graepel, and Campbell (1999), a different approach is taken: The authors consider essentially a Gaussian process classification model, with the relatively minor change of replacing the Gaussian prior on the weight vector **w** with a spherical one. Normalization of the probability model is then unproblematic, but the interpretation of the traditional SVM solution becomes different: It is viewed as an approximation to the Bayes optimal predictor. (The latter is defined as predicting, for each test input, the label $y = \pm 1$ which has the higher posterior probability. In general, this predictor can only be approximated—but not represented exactly—by a single weight vector **w** or, equivalently, a single latent function $\theta(x)$). The authors suggest a 'Bayes point machine' as an improvement on SVMs; this corresponds to making predictions on the basis of the posterior mean of the latent function $\theta(x)$.

A number of possible avenues for future work suggest itself. Some of these (testing the different approximations to the evidence on a number of benchmark data sets, and comparing with other approaches to hyperparameter tuning for SVM classification) have already been touched on in the previous section. The methods for calculating predictive class probabilities also need further testing; in particular, one would like to know whether the computationally attractive method of evaluating class probabilities at the conventional SVM solution can generally give a good approximation to the full average over the posterior.

The example in Section 7 shows that maximization of the naive (unnormalized) evidence is not likely to give reliable settings for *all* hyperparameters. But it may still be a useful heuristic for some classes of hyperparameters; in the example, this was the case for the kernel lengthscale l, in qualitative agreement with the findings of Opper and Winther (2000) and Seeger (2000). More work is obviously needed to find out whether there are general classes of such 'normalization-insensitive' hyperparameters.

For the hyperparameters that do need to be set using the normalized rather than the naive evidence, computationally cheap approximations for the factors $\hat{\mathcal{N}}$ and $\mathcal{N}(X)$ would obviously be desirable; the cavity method of Opper & Winther (2000) would be an interesting candidate for this. Further work is also required to understand in detail the difference between the joint evidence (for training inputs *and* outputs) and the conditional evidence (for training outputs *conditional* on the training inputs). I suspect that in situations where there is strong clustering in the input domain, the joint evidence will be preferable, but this remains to be seen.

Acknowledgments

It is a pleasure to thank Tommi Jaakkola, Manfred Opper, Matthias Seeger, Amos Storkey, Chris Williams and Ole Winther for interesting comments and discussions, and the Royal Society for financial support through a Dorothy Hodgkin Research Fellowship. I am also very grateful to Carl Gold for performing the numerical experiments briefly touched on at the end of Section 7.

Notes

1. I apologize for deviating from the notation in some of the other articles in this issue, where the size of the training set is denoted by l. I wish to reserve this symbol for the lengthscales occurring in the kernels later on.
2. Note that for infinite dimensional kernels—such as the RBF kernel—linear separability is actually the generic case: The feature space is then infinite dimensional and the images $\phi(x_i)$ of the training inputs in the feature space generically span an n-dimensional subspace. With n linearly independent vectors in an n-dimensional space, the two classes are then always linearly separable. So even a very noisy data set *can* generically be fitted without slack variables by using an RBF kernel; but the resulting generalization performance would be expected to be very poor, so the use of slack variables and a finite value of C would still be advised.
3. This is true for $C > \ln 2$. For smaller C, $\nu(\theta(x))$ is actually higher in the gap, and the model makes less intuitive sense.
4. This implies that, if one constructs the model for a data set of size $n + 1$ and then marginalizes over x_{n+1} and y_{n+1}, one does not obtain the same model as if one had assumed from the start that there were only n data points. While this may seem objectionable on theoretical grounds, in most applications one has to deal with a single, given, data set—and thus a single value of n—and then this objection is irrelevant. For sequential learning problems, on the other hand, where n grows dynamically, the present framework may be inappropriate.
5. Mathematically, this corresponds to the fact that the log-likelihood in the SVM case cannot be written as a function of $C\theta(x)$ alone, in contrast to, for example, Gaussian process regression with a logistic transfer function. Explicitly, the argument for the SVM case is as follows: The kernel amplitude K_0 can be absorbed into a rescaling of the latent function, $\theta(x) \to \sqrt{K_0}\theta(x)$. Once this has been done, C and K_0 affect only the contribution each example makes to the log-likelihood, which is $-Cl(\sqrt{K_0}y_i\theta(x_i))$. Because of the presence of the shift in the hinge loss, this—and hence the evidence—depends separately on C and K_0. For a Gaussian process classifier, on the other hand, the same rescaling argument shows that each example contributes $-\ln(1 + e^{-C\sqrt{K_0}y_i\theta(x_i)})$ to the log-likelihood, so that the evidence only depends on the combination $C\sqrt{K_0}$ (and hence either C or K_0 can be fixed to some constant without loss of generality).
6. An alternative motivation for this approximation is as follows: $Q(x)$ is an unknown (functional) hyperparameter specifying the probability model. One could thus try to set its value by maximizing the (joint) evidence $P(X, Y)$. It is easy to show that this leads to an expression of the form $Q(x) = \sum_i w_i\delta_{x,x_i}$, i.e., the empirical input distribution modified by weights w_i; the w_i obey a complicated nonlinear equation. The proxy for $Q(x)$ that I use is then obtained by the simple approximation that all weights are equal, $w_i = 1/n$.
7. A disadvantage is that along eigendirections of the Gram matrix $\mathbf{K}$ with small eigenvalues the β_i are basically unconstrained and will perform essentially random walks, causing undesirably large fluctuations in a Monte Carlo simulation. This can be countered by adding a small diagonal term to $\mathbf{K}$.
8. For small n, fluctuations in the evidence between different data sets generated from the same underlying distribution $P_{\text{true}}(D)$ are expected to be largest. For large data sets, on the other hand, these fluctuations will tend to be small, making it more likely that close to optimal hyperparameter values can be found by maximizing the evidence for the single given data set. An interesting question for future research is whether these statements could be formalized by extending methods from VC-theory.

References

Barber, D. & Bishop, C. M. (1997). Bayesian model comparison by Monte Carlo chaining. In M. C. Mozer, M. I. Jordan, & T. Petsche (Eds.). *Advances in neural information processing systems NIPS 9* (pp. 333–339). Cambridge, MA: MIT Press.

Barber, D. & Williams, C. K. I. (1997). Gaussian processes for Bayesian classification via hybrid Monte Carlo. In M. C. Mozer, M. I. Jordan, & T. Petsche (Eds.). *Advances in neural information processing systems NIPS 9* (pp. 340–346). Cambridge, MA: MIT Press.

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

Chapelle, O. & Vapnik, V. N. (2000). Model selection for support vector machines. In S. A. Solla, T. K. Leen, & K.-R. Müller (Eds.). *Advances in neural information processing systems 12* (pp. 230–236). Cambridge, MA: MIT Press.

Cristianini, N., Campbell, C., & Shawe-Taylor, J. (1999). Dynamically adapting kernels in support vector machines. In M. Kearns, S. A. Solla, & D. Cohn (Eds.). *Advances in neural information processing systems 11* (pp. 204–210). Cambridge, MA: MIT Press.

Cristianini, N. & Shawe-Taylor, J. (2000). *An introduction to support vector machines*. Cambridge: Cambridge University Press.

Csató, L., Fokoué, E., Opper, M., Schottky, B., & Winther, O. (2000). Efficient Approaches to Gaussian Process Classification. In: S. Solla, T. Leen, & K.-R. Müller (Eds.). *Advances in neural information processing systems 12* (pp. 251–257). Cambridge, MA: MIT Press.

Gold, C. & Sollich, P. (2001). In preparation.

Herbrich, R., Graepel, T., & Campbell, C. (1999). Bayesian learning in reproducing kernel Hilbert spaces. Technical Report TR 99-11, Technical University Berlin. http://stat.cs.tu-berlin.de/∼ralfh/publications.html.

Jaakkola, T. & Haussler, D. (1999). Probabilistic kernel regression models. In D. Heckerman & J. Whittaker (Eds.). *Proceedings of the Seventh International Workshop on Artificial Intelligence and Statistics*. San Francisco, CA.

Kwok, J. T. Y. (1999a). Integrating the evidence framework and the support vector machine. In *Proc. European Symp. Artificial Neural Networks (Bruges 1999)* (pp. 177–182). Brussels.

Kwok, J. T. Y. (1999b). Moderating the outputs of support vector machine classifiers. *IEEE Trans. Neural Netw.*, *10:5*, 1018–1031.

MacKay, D. J. C. (1992). The evidence framework applied to classification networks. *Neural Computation*, *4*, 720–736.

Neal, R. M. (1993). Probabilistic inference using Markov Chain Monte Carlo methods. Technical Report CRG-TR-93-1, University of Toronto.

Opper, M. & Winther, O. (2000). Gaussian process classification and SVM: Mean field results and leave-one-out estimator. In A. J. Smola, P. Bartlett, B. Schölkopf, & D. Schuurmans (Eds.). *Advances in large margin classifiers* (pp. 43–65). Cambridge, MA: MIT Press.

Pontil, M., Mukherjee, S., & Girosi, F. (1998). On the noise model of support vector machine regression. Technical Report CBCL Paper 168, AI Memo 1651, Massachusetts Institute of Technology, Cambridge, MA.

Ripley, B. D. (1996). *Pattern recognition and neural networks*. Cambridge: Cambridge University Press.

Schölkopf, B., Bartlett, P., Smola, A., & Williamson, R. (1999). Shrinking the tube: a new support vector regression algorithm. In M. Kearns, S. A. Solla, & D. Cohn (Eds.). *Advances in neural information processing systems 11* (pp. 330–336). Cambridge, MA: MIT Press.

Schölkopf, B., Burges, C., & Smola, A. J. (1998). *Advances in kernel methods: support vector machines*. Cambridge, MA: MIT Press.

Seeger, M. (2000). Bayesian model selection for support vector machines, Gaussian processes and other kernel classifiers. In S. Solla, T. Leen, & K.-R. Müller (Eds.). *Advances in neural information processing systems 12* (pp. 603–609). Cambridge, MA.

Smola, A. J. & Schölkopf, B. (1998). A tutorial on support vector regression. Neuro COLT Technical Report TR-1998-030; Available from http://svm.first.gmd.de/.

Smola, A. J., Schölkopf, B., & Müller, K. R. (1998). The connection between regularization operators and support vector kernels. *Neural Networks*, *11:4*, 637–649.

Sollich, P. (1999). Probabilistic interpretation and Bayesian methods for support vector machines. In *ICANN99—Ninth International Conference on Artificial Neural Networks* (pp. 91–96). London.

Sollich, P. (2000). Probabilistic methods for support vector machines. In S. A. Solla, T. K. Leen, & K.-R. Müller (Eds.). *Advances in neural information processing systems 12* (pp. 349–355). Cambridge, MA: MIT Press.

Tipping, M. E. (2000). The relevance vector machine. In S. Solla, T. Leen, & K.-R. Müller (Eds.). *Advances in neural information processing systems 12* (pp. 652–658). Cambridge, MA: MIT Press.

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

Wahba, G. (1998). Support vector machines, reproducing kernel Hilbert spaces and the randomized GACV. In B. Schölkopf, C. Burges, & A. J. Smola (Eds.). *Advances in kernel methods: support vector machines* (pp. 69–88). Cambridge, MA: MIT Press.

Williams, C. K. I. (1998). Prediction with Gaussian processes: from linear regression to linear prediction and beyond. In M. I. Jordan (Ed.). *Learning and inference in graphical models* (pp. 599–621). Dordrecht: Kluwer Academic.

Received March 24, 2000
Revised January 26, 2001
Accepted February 6, 2001
Final Manuscript July 20, 2001