Probabilistic interpretations and Bayesian methods for Support Vector Machines

Peter Sollich

Department of Mathematics, King's College London
Strand, London WC2R 2LS, U.K.; email: peter.sollich@kcl.ac.uk

Abstract

Support Vector Machines (SVMs) can be interpreted as maximum a posteriori solutions to inference problems with Gaussian Process (GP) priors and appropriate likelihood functions. Focussing on the case of classification, I show first that such an interpretation gives a clear intuitive meaning to SVM kernels, as covariance functions of GP priors; this can be used to guide the choice of kernel. Second, a probabilitistic interpretation allows Bayesian methods to be used for SVMs: Using a local approximation of the posterior around its maximum (the standard SVM solution), I discuss how the evidence for a given kernel and noise parameter can be estimated, and how approximate error bars for the classification of test points can be calculated.

1 SVMs

Support Vector Machines (SVMs) have been the subject of intense research activity from within the neural networks community over the last few years; for tutorial introductions and overviews of recent developments see [1, 2, 3]. One of the open questions that remains is how to set the 'tunable' parameters of an SVM algorithm: While methods for choosing the width of the kernel function and the noise parameter (which controls how closely the training data are fitted) have recently been proposed [4, 5], the effect of the overall shape of the kernel function remains imperfectly understood [1]. In this paper I suggest that a *probabilistic* interpretation of SVMs might be useful in tackling this problem. It has two main benefits: First, it clarifies the role of the kernel as specifying an SVM 'prior' over functions on the input space, avoiding the need to think in terms of an abstract feature space. I illustrate this with samples from some typical SVM priors and discuss general guidelines for the choice of kernels that emerge. Second, a probabilistic interpretation also allows Bayesian methods to be applied to SVMs. As an example, I sketch how the log-likelihood (or 'evidence') for an SVM model – specified by a kernel and noise parameter – can be evaluated approximately; this could be a useful alternative to generalization error bounds in guiding the search for optimal kernels. I also outline how approximate error bars for the predictions of a trained SVM can be obtained. These could be of use in safety-critical applications, for example, where the conventional purely deterministic SVM predictions are undesirable.

I will focus mainly on (two-class) classification problems. Suppose we are given a set D of n training examples, each of the form (x, y_x) with a binary output $y_x = \pm 1$ classifying to which of the two possible classes the input x belongs. The basic SVM idea is then to map the inputs x onto vectors $\boldsymbol{\phi}(x)$ 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 $\mathbf{w}\cdot\boldsymbol{\phi}(x) + b = 0$ which separate the training examples (i.e., which obey $y_x(\mathbf{w}\cdot\boldsymbol{\phi}(x) + b) > 0$ for all $x \in X$, X being the set of training inputs), the SVM solution is then chosen as the one with the largest *margin*, i.e., the largest minimal distance from any of the training examples. In practice, it is easier to specify the margin instead (the conventional value is unity) and minimize the squared length of the weight vector $||\mathbf{w}||^2$ [1]. 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 $y_x(\mathbf{w}\cdot\boldsymbol{\phi}(x)+b) \geq 1$ for all training examples.

What happens if the problem is not linearly separable, even in the high-dimensional feature space? Then 'slack variables' $\xi_x \geq 0$ are introduced which measure how much the margin constraint is violated for training input x; one writes $y_x(\mathbf{w}\cdot\boldsymbol{\phi}(x)+b) \geq 1-\xi_x$. To control the amount of slack allowed (which determines how closely the training data are fitted), a penalty term must then be added to the objective function $\frac{1}{2}||\mathbf{w}||^2$. The most widely used choice is $C\sum_x \xi_x$, with a penalty coefficient C. Training examples for which $y_x(\mathbf{w}\cdot\boldsymbol{\phi}(x)+b) \geq 1$ (and hence $\xi_x = 0$) then incur no penalty, while for the other examples there is a penalty of $C[1-y_x(\mathbf{w}\boldsymbol{\phi}(x)+b)]$ each. Altogether, the SVM optimization problem can therefore be expressed as follows: Find $\mathbf{w}$ and b to minimize

$$\frac{1}{2}||\mathbf{w}||^2 + C\sum_{x\in X} l(y_x[\mathbf{w}\cdot\boldsymbol{\phi}(x)+b]) \qquad (1)$$

where $l(z)$ is the (shifted) 'hinge loss', $l(z) = (1-z)\Theta(1-z)$.

The probabilistic interpretation of SVMs that I discuss below hinges on the fact that (1) can be regarded as defining a (negative) log-posterior probability for the parameters $\mathbf{w}$ and b of the SVM, given a training set D. The first term gives us the prior $P(\mathbf{w}) \sim \exp(-\frac{1}{2}||\mathbf{w}||^2)$. This is a Gaussian prior on $\mathbf{w}$; the components of $\mathbf{w}$ are uncorrelated with each other and have unit variance. The prior on b is flat (uninformative, improper) and I will not write it explicitly. The weights $\mathbf{w}$ occur in the second term of (1) – to be identified as the log-likelihood shortly – only through $a(x) = \mathbf{w}\cdot\boldsymbol{\phi}(x)$. It is therefore useful to express the prior directly as a distribution over the functions a. The function values $a(x)$ have a joint Gaussian distribution because the components of $\mathbf{w}$ do, and their covariances are given by $\langle a(x)a(x')\rangle = \langle(\boldsymbol{\phi}(x)\cdot\mathbf{w})(\mathbf{w}\cdot\boldsymbol{\phi}(x'))\rangle = \boldsymbol{\phi}(x)\boldsymbol{\phi}(x')$. In other words, the SVM prior is simply a *Gaussian process* (GP) prior over the functions a, with covariance function $K(x,x') = \boldsymbol{\phi}(x)\cdot\boldsymbol{\phi}(x')$ (and zero mean).

To interpret the second term in (1) as a (negative) log-likelihood, one only has to define the probability of obtaining output y for a given x (and a, b) as

$$P(y=\pm 1|x,a,b) = \kappa(C)e^{-Cl(y[a(x)+b])} \quad (2)$$

The proportionality constant $\kappa(C)$ here needs to be chosen such that the probabilities for $y = \pm 1$ never add up to a value larger than one; it is sensible to choose the largest value that achieves this, which is

$$\kappa(C) = 1/[1+\exp(-2C)]. \qquad (3)$$

For a generic value of $a(x)+b$, however, the probabilities $P(y=\pm 1|x,a,b)$ then still add to a value < 1. It is therefore necessary to introduce a 'don't-know' class (labelled by $y = 0$, say), with probability

$$P(y=0|x,a,b) = 1 - \sum_{y=\pm 1} P(y=\pm 1|x,a,b). \qquad (4)$$

While this may seem surprising at first sight, it has the appealing feature that the probability for the 'don't-know' class is largest in the 'gap', $-1 < a(x)+b < 1$, where one would intuitively expect the output of the SVM to be least certain[1]. Up to the trivial additive constant $n\ln(1+e^{-2C})$, the second term in (1) can now be identified as the negative log-likelihood of the observed data D.

To summarize, the task of training an SVM for classification can be reformulated as follows: Given a zero-mean Gaussian process prior over functions a with covariance function $K(x,x')$, a flat prior over offsets b, and the likelihood function defined by (2–4), find the maximum a posteriori (MAP) values of a and b. (These are the values that maximize $P(a,b|D) \sim P(a)P(D|a,b)$.) The feature space has disappeared entirely from this formulation[2]; all its relevant properties are encoded in the GP prior, specified by the kernel $K(x,x')$. There is nothing profound about this correspondence between SVMs and GPs. It is related to the common link to reproducing kernel Hilbert spaces [7], and can be extended from SVMs to more general

[1] This is based on the reasonable assumption that C is not too small ($C \geq 1$, roughly). Very small values of C would give a large range of values of $a(x)+b$ where the probabilities for $y \pm 1$ are comparable, producing essentially random outputs; this is an unrealistic situation.

[2] The case of SVM regression is completely analogous. The only modification is in the likelihood function, which then determines the probability of a real-valued output, $P(y|x,a,b) \sim \exp\{-Cl_\epsilon(y-[a(x)+b])\}$, in terms of the ϵ-insensitive loss function $l_\epsilon(z) = (|z|-\epsilon)\Theta(|z|-\epsilon)$.

kernel methods [8]; for connections to regularization operators see also [9].

In what follows, I will consider the input space as discretized for simplicity, with sums over all possible values of x replacing integrals. This avoids technical subtleties, but does not restrict the scope of the treatment (the discretization 'grid' can be made arbitrarily fine). The (unnormalized log-) posterior is then given by

$$\begin{aligned} f(a,b) &= \ln P(a)P(D|a,b) = \qquad (5)\\ &-\frac{1}{2}\sum_{x,x'} a(x)K^{-1}(x,x')\,a(x') - \frac{1}{2}\ln\det K \\ &- C\sum_{x\in X} l(y_x[a(x)+b]) - n\ln(1+e^{-2C}) \end{aligned}$$

The standard SVM algorithm finds the maximum a^*, b^* of $f(a,b)$. For input points x which are not in the training set X, this implies $\sum_{x'} K^{-1}(x,x')a^*(x') = 0$, so one can write a^* in the form

$$a^*(x') = \sum_{x\in X} \alpha_x y_x K(x,x');$$

the factors y_x are separated out here to retain the standard SVM notation. Maximizing $f(a,b)$ w.r.t. the remaining $a(x)_{x\in X}$ yields a natural division of the training set X into three parts,

$$X_0 \;:\; y_x(a^*(x)+b^*) > 1, \quad \alpha_x = 0 \qquad (6)$$
$$X_m \;:\; y_x(a^*(x)+b^*) = 1, \quad \alpha_x \in [0,C] \qquad (7)$$
$$X_C \;:\; y_x(a^*(x)+b^*) < 1, \quad \alpha_x = C \qquad (8)$$

I will call these, respectively, the easy (correctly classified), marginal, and hard (inside the 'gap') training examples; the last two classes form the support vectors (SVs; $\alpha_x > 0$). The sparseness of the SVM solution (often the number of SVs 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 that have been proposed; there a sigmoidal (often logistic) 'transfer function' normally replaces the likelihood (2) (see *e.g.* [10]). Its log has nonzero gradient everywhere, and so the MAP solution will not be sparse. Moreover, in the noise free limit, the sigmoidal transfer function becomes a step function, and the MAP values of a and b will tend to the trivial solution $a(x) = b = 0$. This illuminates from an alternative point of view why the margin (the 'shift' in the hinge loss) is important for SVMs.

2 Understanding kernels

We saw above that SVMs correspond to maximum a posteriori solutions of inference problems with zero mean Gaussian process (GP) priors; the relevant covariance function is the kernel $K(x,x') = \langle a(x)a(x')\rangle$. This makes it easy to understand the role of the kernel in SVM inference. Recall that the covariance function of a GP encodes in an easily interpretable way prior assumptions about what kind of functions a are likely to occur (see *e.g.* [10]). Smoothness, for example, is controlled by the behaviour of $K(x,x')$ for $x' \to x$: The Ornstein-Uhlenbeck (OU) covariance function $K(x,x') \propto \exp(-|x-x'|/l)$ produces very rough (non-differentiable) functions, while functions sampled from the squared exponential (SE) prior with $K(x,x') \propto \exp(-|x-x'|^2/(2l^2))$ are infinitely differentiable. Fig. 1 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 a to vary significantly. It therefore determines the typical size of the decision regions of SVMs sampled from the prior (see Fig. 1). The amplitude of the covariance function also has an intuitive meaning. It determines the typical values of $|a(x)|$, which are of order $A(x) = [K(x,x)]^{1/2}$. If this quantity is much larger than unity, then from the likelihood function (2–4) the probability of either $y = +1$ or $y = -1$ will typically be close to unity and the likelihood of 'don't know' outputs ($y = 0$) will be small. In other words, decision regions will have only small 'gaps of uncertainty' between them. Using the fact that $a(x)$ varies from typically $+A(x)$ to $-A(x)$ over a length l, the size of the 'gap' (defined by $|a(x)| < 1$, if we set $b = 0$ for simplicity) can in fact be estimated as $l/A(x)$. For large kernel amplitudes $A(x)$ this 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 gaps of uncertainty; see again Fig. 1.

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 uncertainty gaps 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 Fig. 1 a sample from the popular polynomial SVM kernel, $K(x,x') = (1 + x\cdot x')^p$. In contrast to the OU and SE 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 $a(x)$ 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 much more costly computationally.

3 Bayesian methods

Beyond providing intuition about SVM kernels, the probabilistic interpretation discussed above also opens the door to the application of Bayesian methods to SVMs. I give two examples there: The estimation of the 'evidence' for a specific kernel and noise parameter from the training data, and the calculation of error bars. In this context, it is easiest to replace the improper prior on the offset b by a Gaussian prior with zero mean and a large standard deviation B. The functions $\tilde{a}(x) = a(x) + b$ then come from a GP prior with covariance function $\tilde{K}(x,x') = K(x,x') + B^2$, and b does not need to be considered separately. I work with this formulation in the following, but drop the tildes on $\tilde{a}(x)$ and $\tilde{K}(x,x')$ again to save notation.

The evidence[3] is simply the likelihood of the data for a given model (specified by a kernel $K(x,x')$ and a noise parameter C), obtained by integration over the model parameters $a(x)$ (see *e.g.* [10] for a discussion in the GP context):

$$P(D) = \int da\, P(a) P(D|a) \tag{9}$$

The conditioning on the model $\{K(x,x'), C\}$ is implicit here. Inserting (6) leads to an integral which is intractable analytically. Progress can be made, however, by expanding the log-integrand around the SVM solution a^*, to second order in the deviations δa. For all input points $x \notin X_m$ which are *not* marginal training inputs, the linear terms in $\delta a(x)$ are zero and the method is equivalent to Laplace's approximation. Carrying out the resulting Gaussian integrals leaves an integral over $\delta a(x)_{x \in X_m}$. The log-integrand now contains *linear* terms in these variables, whose prefactors depend on the sign of the integration variables. Discarding quadratic terms as subdominant, the integral then factorizes and can be carried out explicitly. The end result of this calculation, which will be described in more detail elsewhere, is $\ln P(D) \approx$

$$\begin{aligned}
&-\frac{1}{2} \sum_{x,x' \in X_C \cup X_m} y_x \alpha_x K(x,x')\, y_{x'} \alpha_{x'} \\
&-C \sum_{x \in X} l(y_x a^*(x)) - n \ln(1 + e^{-2C}) \\
&-\frac{1}{2} \ln\det(2\pi K_m) + \sum_{x \in X_m} \ln \frac{C}{\alpha_x (C - \alpha_x)}
\end{aligned}$$

The first three terms represent the contribution from the maximum of the integrand in (9), while the last two terms arise from the integration over the $\delta a(x)$. The matrix K_m is the submatrix of $K(x,x')$ corresponding to the marginal training inputs, $x, x' \in X_m$. Given the sparseness of the SVM solution, this should be reasonably small, making $\det(K_m)$ amenable to numerical computation or estimation [10]. Finally, note that the approximate $P(D)$ diverges when $\alpha_x \to 0$ or $\to C$ for one of the marginal training inputs; the approximation of retaining only linear terms in $\delta a(x)_{x \in X_m}$ in the log integrand then breaks down. Diagonal quadratic terms (those that don't couple different $\delta a(x)_{x \in X_m}$) can be retained to remedy this, leading to an only slightly more complicated final expression. Alternatively, a simple heuristic 'regularization' of the approximate evidence can be used [11].

To calculate error bars on SVM predictions, one requires the distribution of $a(z)$

[3] The question of (average case) optimality of the log evidence for choosing hyperparameters is subtle in the SVM case, because the 'don't know' class never occurs in actual training data. See Ref. [11].

for a training input z, given the data set D; integrating over this distribution in the likelihood model (2–4) then gives values for the class probabilities $P(y=-1,0,1|z,D)$. This leaves the problem of determining the distribution of $a(z)$. The calculation is very similar to the one outlined above for the evidence, with $a(z)$ now held fixed. Expanding again to second order in $\delta a(z)$ leads to a Gaussian approximation for the distribution of $a(z)$; explicit formulae for the resulting class probabilities will be given elsewhere. Here I only note that the results depend, in addition to the quantities affecting the evidence, on the prior correlations $K(z,x)_{x\in X_m}$ between the test point and the marginal training inputs, and of course on the maximum a posteriori (SVM) value $a^*(z)$.

4 Conclusion

More work is obviously required to test the above ideas in practical applications. It will be interesting to see how good a guide the approximate evidence can be for the optimization of SVM kernels and noise parameters; and whether the approximate error bars can assign sensible confidence levels to the (conventionally deterministic) output of SVMs. Comparisons with other approximations for the evidence, based on Gaussian variational [12] and cavity field methods [13] should provide additional insight.

Beyond these more tangible applications, I hope that the probabilistic interpretation of SVMs that I have emphasized will add a valuable angle from which the workings of this powerful inference tool can be understood.

Acknowledgements: I would like to thank Manfred Opper for stimulating discussions, and the Royal Society for financial support through a Dorothy Hodgkin Research Fellowship.

References

[1] C J C Burges. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

[2] A J Smola and B Schölkopf. A tutorial on support vector regression. 1998. Neuro COLT Technical Report TR-1998-030; available from http://svm.first.gmd.de/.

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

[4] B Schölkopf, P Bartlett, A Smola, and R Williamson. Shrinking the tube: a new support vector regression algorithm. In [6].

[5] N Cristianini, C Campbell, and J Shawe-Taylor. Dynamically adapting kernels in support vector machines. In [6].

[6] M Kearns, S A Solla, and D Cohn, editors, *Advances in Neural Information Processing Systems 11*, Cambridge, MA. MIT Press. In press.

[7] G Wahba. Support vector machines, reproducing kernel Hilbert spaces and the randomized GACV. Technical Report 984, University of Wisconsin, 1997. See also pages 69–88 in [3].

[8] T S Jaakkola and D Haussler. Probabilistic kernel regression models. In *Proc. 7th Int. Workshop on AI and Statistics.* To appear.

[9] A J Smola, B Schölkopf, and K R Müller. *Neural Networks*, 11:637–649, 1998.

[10] C K I Williams. In M I Jordan, editor, *Learning and Inference in Graphical Models*, pages 599–621. Kluwer Academic, 1998.

[11] P Sollich. Probabilistic methods for Support Vector Machines. Submitted to *NIPS'99*.

[12] M Seeger. Bayesian model selection for Support Vector machines, Gaussian processes and other kernel classifiers. Submitted to *NIPS'99*.

[13] M Opper and O Winther. Gaussian process classification and SVM: Mean field results and leave-one-out estimator. In *Advances in Large Margin Classifiers.* MIT Press. To appear.

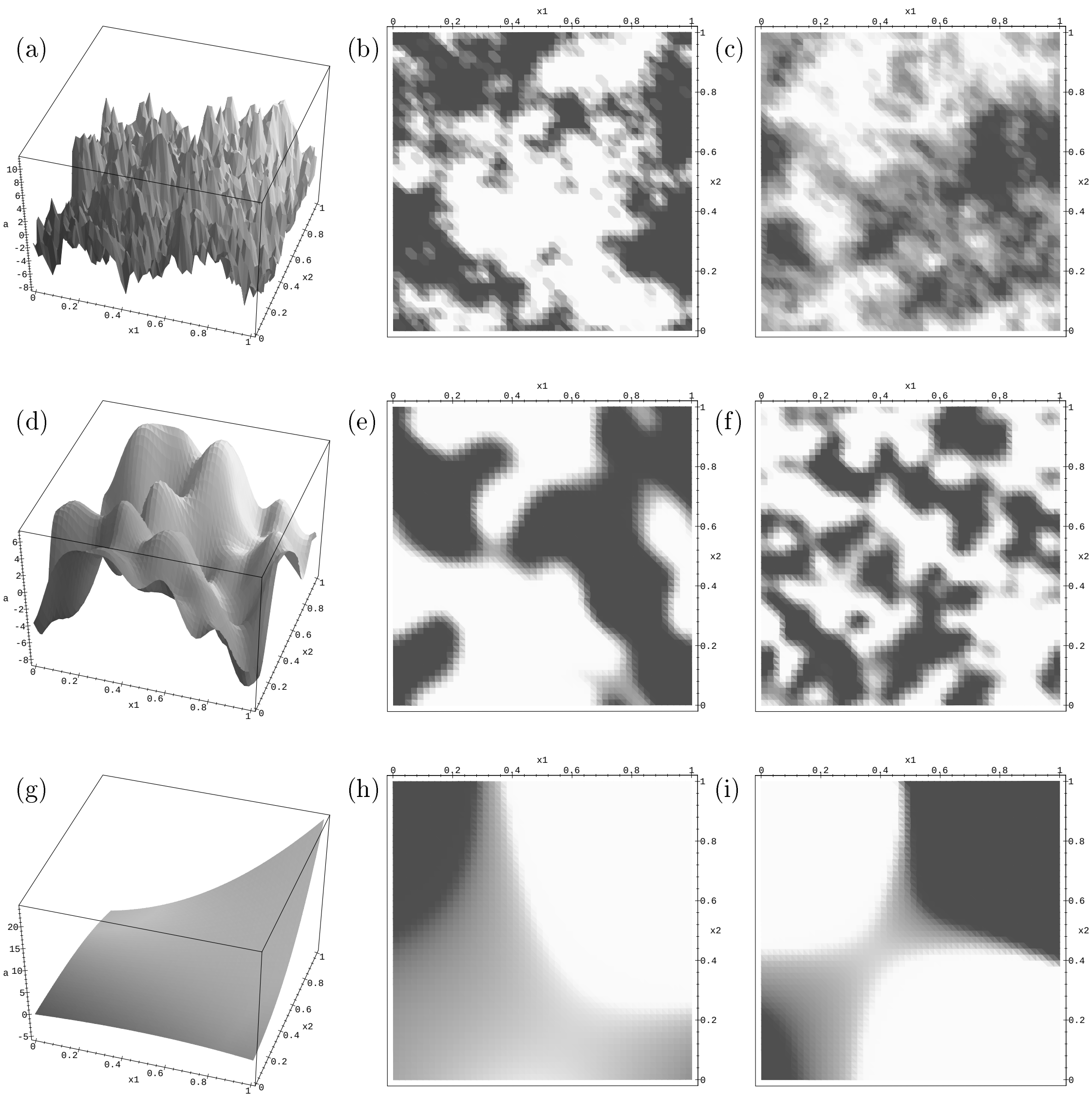

Figure 1: Samples from SVM priors; the input space is the unit square $[0,1]^2$. 3d plots are samples $a(x)$ from the underlying Gaussian process priors. 2d greyscale plots represent the output distributions obtained when $a(x)$ is used in the likelihood model (2–4), with $b = 0$ and $C = 2$; the greyscale indicates the value of $\langle y \rangle = P(y = +1) - P(y = -1)$ (black: -1, white: $+1$). (a,b) OU kernel/covariance function $K_0 \exp(-|x - x'|/l)$, giving rough $a(x)$ and decision boundaries. Length scale $l = 0.1$, $K_0 = 10$. (c) Same with $K_0 = 1$; note how the grey 'uncertainty gaps' between regions of definite outputs (black/white) have widened. (d,e) As first row, but with SE kernel $K_0 \exp[-(x - x')^2/(2l^2)]$, yielding smooth $a(x)$ and decision boundaries. (f) Changing l to 0.05 (while holding K_0 fixed at 10) 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) $p = 10$. Note the absence of a clear length scale, and the widely differing magnitudes of $a(x)$ in the bottom left ($x = [0,0]$) and top right ($x = [1,1]$) corners of the square.