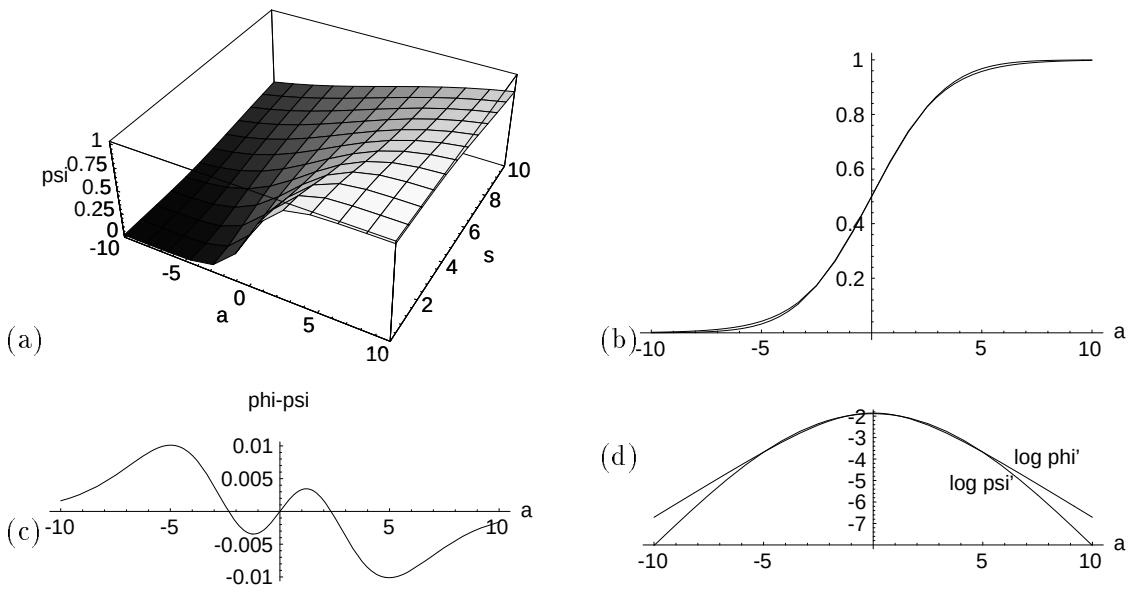

Figure 5.1: **Approximation to the moderated probability**
(a) The function $\psi(a, s^2)$, evaluated numerically. In (b) the functions $\psi(a, s^2)$ and $\phi(a, s^2)$ defined in the text are shown as a function of a for $s^2 = 4$. In (c), the difference $\phi - \psi$ is shown for the same parameter values. In (d), the breakdown of the approximation is emphasised by showing $\log \phi'$ and $\log \psi'$ (derivatives with respect to a). The errors become significant when $a \gg s$.

moderated outputs will make a difference if a more sophisticated penalty function is involved. In the following demonstration the performance of a classifier's outputs is measured by the value of G achieved on a test set.

A model classification problem with two input variables and two possible classes is shown in figure 5.2a. Figure 5.2b illustrates the output of a typical trained network, using its *most probable* parameter values. Figure 5.2c shows the *moderated* outputs of the same network. Notice how the moderated output is similar to the most probable output in regions where the data are dense. In contrast, where the data are sparse, the moderated output becomes significantly less certain than the most probable output; this can be seen by the widening of the contours. Figure 5.2d shows the correct posterior probability for this problem given the knowledge of the true class densities.

Several hundred neural networks having two inputs, one hidden layer of sigmoid units and one sigmoid output unit were trained on this problem. During optimisation, the second weight decay scheme of chapter 3 was used, using independent decay rates for each of three weight classes: hidden weights, hidden unit biases, and output weights and biases. This corresponds to the prior which models the weights in each class as coming from a gaussian; the scale of the gaussians for different classes are independent and are specified by regularising constants α_c. Each regularising constant is optimised on line by intermittently updating it to its *most probable* value as estimated within the 'evidence' framework.

The prediction abilities of a hundred networks using their 'most probable' outputs and using the moderated outputs suggested above are compared in figure 5.3. It can be seen that the predictions given by the moderated outputs are in nearly all cases superior. The improvement is most substantial for underdetermined networks with relatively poor performance. In a small fraction of the solutions however, especially among the best solutions,

In regression problems, it is also important to calculate error bars on outputs, but the problem is more acute in the case of classification because, on account of the non–linear output, the mean output over the posterior distribution is not equal to the most probable network's output. To obtain an output representative of the posterior ensemble of networks around $\mathbf{w}_{\rm MP}$, we need to *moderate* the output of the most probable network in relation to the error bars on $\mathbf{w}_{\rm MP}$.

Of course this idea of averaging over the hidden parameters is not new: marginalisation goes back to Laplace. More recently, and in a context closer to the present one, the same message can be found for example in [68]. But it seems that most practitioners of adaptive classification do not currently use marginalisation.

I suggest that any classifier should have two sets of outputs. The first set would give the usual class probabilities corresponding to $\mathbf{w}_{\rm MP}$, $y(\mathbf{x};\mathbf{w}_{\rm MP})$; these outputs would be used for learning, *i.e.* for calculating the error signals for optimisation of $\mathbf{w}_{\rm MP}$. The second set would be the moderated outputs $y(\mathbf{x};P(\mathbf{w}|D)) = \int d^k\mathbf{w}\, y(\mathbf{x};\mathbf{w})P(\mathbf{w}|D)$; these outputs would be used for all other applications, *e.g.* prediction, evaluation of test error, and for evaluating the utility of candidate data points (section 5.4). Let us now discuss how to calculate the moderated outputs. It will then be demonstrated that these outputs do indeed make better predictions.

Calculating the moderated outputs

If we assume a locally gaussian posterior probability distribution[3] over $\mathbf{w} = \mathbf{w}_{\rm MP} + \Delta\mathbf{w}$, $P(\mathbf{w}|D) \simeq P(\mathbf{w}_{\rm MP})\exp(-\frac{1}{2}\Delta\mathbf{w}^{\rm T}\mathbf{A}\Delta\mathbf{w})$, and if we assume that the activation $a(\mathbf{x};\mathbf{w})$ is a locally linear function of $\mathbf{w}$ with $\partial a/\partial\mathbf{w} = \mathbf{g}$, then for any $\mathbf{x}$, the activation a is approximately gaussian distributed:

$$P(a(\mathbf{x})|D) = \text{Normal}(a^{\rm MP}, s^2) = \frac{1}{\sqrt{2\pi s^2}}\exp\left(-\frac{(a-a^{\rm MP})^2}{2s^2}\right), \tag{5.6}$$

where $a^{\rm MP} = a(\mathbf{x};\mathbf{w}_{\rm MP})$ and $s^2 = \mathbf{g}^{\rm T}\mathbf{A}^{-1}\mathbf{g}$. This means that the moderated output is:

$$P(t=1|\mathbf{x},D) = \psi(a^{\rm MP}, s^2) \equiv \int da\, f(a)\,\text{Normal}(a^{\rm MP}, s^2). \tag{5.7}$$

This is to be contrasted with the most probable network's output, $y(\mathbf{x};\mathbf{w}_{\rm MP}) = f(a^{\rm MP})$. The integral of a sigmoid times a gaussian cannot be solved analytically; here I suggest a simple numerical approximation to it:

$$\psi(a^{\rm MP}, s^2) \simeq \phi(a^{\rm MP}, s^2) \equiv f(\kappa(s)a^{\rm MP}) \tag{5.8}$$

with $\kappa = 1/\sqrt{1+\pi s^2/8}$. This approximation is not globally accurate over $(a^{\rm MP}, s^2)$, (for large $s^2 > a$ the function should tend to an error function, not a logisitic) but it breaks down gracefully. The value of κ was chosen so that the approximation has the correct gain at $a^{\rm MP} = 0$, as $s^2 \to \infty$. A representative of this approximation is given in figure 5.1 which compares ϕ and ϕ' with numerical evaluations of ψ and ψ'. A similar approximation in terms of the error function is suggested in [68].

If the output is immediately used to make a (0/1) decision, then the use of moderated outputs will make no difference to the performance of the classifier (unless the costs associated with error are assymetrical), since both functions pass through 0.5 at $a^{\rm MP} = 0$. But

[3] Conditioning variables such as $\mathcal{A}, \mathcal{R}, \{\alpha_c\}$ will be omitted in this section, since the emphasis is not on model comparison.

to β must appear, expressing the precision of the teacher's estimates. This constant would correspond to the effective number of observations on which the teacher's opinion is based.

The calculation of the gradient and Hessian of G is as easy as for a quadratic E_D, if the output unit's activation function is the traditional logistic $f(a) = 1/(1+e^{-a})$, or the generalised 'softmax' in the case of more than two classes [10]. The appropriateness of a logisitic output function for a classifier is well known: it is the function that converts a log probability ratio a into a probability $f(a)$.

Gradient: If $y(\mathbf{x}^{(m)}) = f(a(\mathbf{x}^{(m)}))$ as defined above, the gradient of G is

$$\nabla G = \sum_m (t_m - y)\mathbf{g}_{(m)}, \tag{5.4}$$

where $\mathbf{g}_{(m)} = \partial a/\partial \mathbf{w}|_{\mathbf{x}=\mathbf{x}^{(m)}}$.

Hessian: The Hessian can be analytically evaluated [8], but a useful approximation neglecting terms in $\partial^2 a/\partial^2 \mathbf{w}$ is:

$$\nabla\nabla G \simeq -\sum_m f' \mathbf{g}_{(m)} \mathbf{g}_{(m)}^{\mathrm{T}}. \tag{5.5}$$

This approximation is expected to be adequate for the evaluation of error bars, for use in data selection and for the evaluation of the number of well–determined parameters γ. A more accurate evaluation of the Hessian is probably needed for estimation of the evidence. In this chapter's demonstrations, the Hessian is evaluated using second differences.

Validity of approximations

On account of the central limit theorem, we expect the posterior distribution to converge to a set of locally gaussian peaks with increasing quantities of data. However, the quadratic approximation to G is expected to converge more slowly than the quadratic approximation to E_D, the error function for regression models, because (a) G is not a quadratic function even for a linear model (a model for which $a = \sum w_h \phi_h(\mathbf{x})$): each term in G has the large scale form of a ramp function; and (b) only inputs which fall in the 'bend' of the ramp contribute curvature to G. If we have the opportunity for active data selection we could improve the convergence of this quadratic approximation by selecting inputs that are expected to contribute maximal curvature. A related data selection criterion is derived in section 5.4.

5.2 Every classifier should have two sets of outputs

Consider a classifier with output $y(\mathbf{x};\mathbf{w}) = f(a(\mathbf{x};\mathbf{w}))$. Assume that we receive data D and infer the posterior probability of the parameters $\mathbf{w}$ (*i.e.* we perform 'learning'). Now if we are asked to make predictions with this classifier, it is common for the most probable parameter vector $\mathbf{w}_{\mathrm{MP}}$ to be used as the sole representative of the posterior distribution. This strategy seems unwise, however, since there may be regions in input space where the posterior ensemble is very uncertain about what the class is; in such regions the output of the network should be $y \simeq 0.5$ (assuming equiprobable classes *a priori*), whereas typically the network with parameters $\mathbf{w}_{\mathrm{MP}}$ will give a more extreme, unrepresentative and overconfident output. *The error bars on the parameters should be taken into account when predictions are made.*

2. The alternative models are ranked in the light of the data.

3. The expected utility of alternative new data points is estimated for the purpose of 'query learning' or 'active data selection.'

This chapter will present Bayesian ideas for these three tasks. Other aspects of classifiers use such as prediction of generalisation ability are not addressed.

First let us review the framework for supervised adaptive classification.

Derivation of the objective function $G = \sum t \ln p$

The same notation and conventions will be used as in chapters 2 and 3. Let the data set be $D = \{\mathbf{x}^{(m)}, t_m\}$, $m = 1 \ldots N$. In a classification problem, each target t_m is a binary (0/1) variable (more than two classes can also be handled [10]), and the activity of the output of a classifier is viewed as an estimate of the probability that $t = 1$. It is assumed that the classification problem is noisy, that is, repeated sampling at the same $\mathbf{x}$ would produce different values of t with certain probabilities; those probabilities, as a function of $\mathbf{x}$, are the quantities that a discriminative classifier is intended to model. It is well known that the natural objective function in this case is an information–based distance measure, rather than the sum of squared errors [10, 27, 28, 67].

A classification model $\mathcal{H}$ consists of a specification of its architecture $\mathcal{A}$ and the regulariser $\mathcal{R}$ for its parameters $\mathbf{w}$. When a classification model's parameters are set to a particular value, the model produces an output $y(\mathbf{x}; \mathbf{w}, \mathcal{A})$ between 0 and 1, which is viewed as the probability $P(t = 1|\mathbf{x}, \mathbf{w}, \mathcal{A})$. The likelihood, *i.e.* the probability of the data[2] as a function of $\mathbf{w}$, is then:

$$\begin{aligned} P(D\,|\mathbf{w}, \mathcal{A}) &= \prod_m y^{t_m}(1-y)^{1-t_m} \\ &= \exp G(D\,|\mathbf{w}, \mathcal{A}) \end{aligned}$$

where

$$G(D\,|\mathbf{w}, \mathcal{A}) = \sum_m t_m \log y + (1 - t_m)\log(1 - y). \tag{5.1}$$

This is the probabilistic motivation for the cross–entropy objective function $\sum p \log \frac{q}{p}$. Now if we assign a prior to alternative network configurations $\mathbf{w}$ (see chapter 3),

$$P(\mathbf{w}|\{\alpha_c\}, \mathcal{A}, \mathcal{R}) = \frac{\exp(-\sum_c \alpha_c E_W^{(c)})}{Z_W}, \tag{5.2}$$

we obtain a posterior:

$$P(\mathbf{w}|D, \{\alpha_c\}, \mathcal{A}, \mathcal{R}) = \frac{\exp(-\sum_c \alpha_c E_W^{(c)} + G)}{Z_M}, \tag{5.3}$$

where Z_W and Z_M are the appropriate normalising constants. Thus the identical framework is obtained to that in chapter 3 with $-G$ replacing the term βE_D. Note that in contrast to the framework in chapter 3 there is now no free parameter β and no $Z_D(\beta)$. If however a teacher supplies probability estimates t instead of binary targets, then a constant equivalent

[2] Strictly this is the probability of $\{t_m\}$ given $\{\mathbf{x}^{(m)}\}, \mathbf{w}, \mathcal{A}$; the density over $\{\mathbf{x}\}$ is not modelled by the 'discriminative' classifiers discussed in this chapter.

Chapter 5

The Evidence Framework applied to Classification Networks

Abstract

Three Bayesian ideas are presented for supervised adaptive classifiers. First, it is argued that the output of a classifier should be obtained by marginalising over the posterior distribution of the parameters; a simple approximation to this integral is proposed and demonstrated. This involves a 'moderation' of the most probable classifier's outputs, and yields improved performance. Second, it is demonstrated that the Bayesian framework for model comparison described for regression models in chapters 2 and 3 can also be applied to classification problems. This framework successfully chooses the magnitude of weight decay terms, and ranks solutions found using different numbers of hidden units. Third, an information–based data selection criterion is derived and demonstrated within this framework.

5.1 Introduction

A quantitative Bayesian framework has been described for learning of mappings in feedforward networks in chapters 2 and 3. It was demonstrated that this 'evidence' framework could successfully choose the magnitude and type of weight decay terms, and could choose between solutions using different numbers of hidden units. The framework also gives quantified error bars expressing the uncertainty in the network's outputs and its parameters. In chapter 4 information–based objective functions for active learning were discussed within the same framework.

These three chapters concentrated on interpolation (regression) problems. Neural networks can also be trained to perform classification tasks.[1] This chapter will show that the Bayesian framework for model comparison can be applied to these problems too.

Assume that a set of candidate classification models are fitted to a data set, using standard methods. Three aspects of the use of classifiers can then be distinguished:

1. The individual classification models are used to make predictions about new targets.

[0] Chapter 5 of PhD thesis 'Bayesian Methods for Adaptive Models' by David MacKay, California Institute of Technology, submitted December 10 1991.

[1] In regression the target variables are real numbers, assumed to include additive errors; in classification the target variables are discrete class labels.