Nonstationary Logistic Regression

William D. Penny and Stephen J. Roberts

Technical Report,
Neural Systems Research Group,
Department of Electrical and Electronic Engineering,
Imperial College of Science, Technology and Medicine,
London SW7 2BT., U.K.
w.penny@ic.ac.uk, s.j.roberts@ic.ac.uk

May 5, 1999

Abstract

We propose an online learning algorithm for training a logistic regression model on nonstationary classification problems. The nonstationarity is captured by modelling the weights in a logistic regression classifier as evolving according to a first order Markov process. The weights are updated using the extended Kalman filter formalism and nonstationarities are tracked by inferring a time-varying state noise variance parameter. We describe an algorithm for doing this based on maximising the evidence of model predictions. The algorithm is illustrated on a number of synthetic problems.

1 Introduction

This paper proposes an online learning algorithm for training a logistic regression model on nonstationary classification problems. By nonstationary[1] we mean that the statistics of each class may vary with time or, equivalently from the classification perspective, that the optimal decision boundary changes with time.

The simplest online algorithm for training a logistic regression model is steepest descent. This algorithm has a learning rate parameter which we would wish to be small in regions of stationarity, to allow convergence, and large in regions of nonstationarity, to allow tracking. Our problem therefore, can be seen as equivalent to estimating a time-varying learning rate.

In this paper the weights are modelled as evolving according to a first order Markov process. This state-space approach (where the state is the weight vector) possesses an implicit learning rate which decreases as more data is seen from the same stationary regime and increases when a new regime is encountered. The ability to adapt to a new regime is based on inferring a state-noise variance parameter. We present an algorithm for doing this based on maximising the evidence of model predictions.

In section 2 we briefly review steepest descent and Bayesian algorithms for stationary logistic regression. In section 3 we describe how the Bayesian formalism is extended to the case of nonsatationary logistic regression and in section 4 describe algorithms for inferring the state noise variance. In section 5 we present results from numerical examples. Throughout the paper we consider only the case of binary logistic regression although the methods are readily extended to multiple classes.

[1] We assume the usual technical definition of nonstationarity (see eg. [3]) which for a sequence of input-output pairs, $\{\boldsymbol{x}_t, z_t\}$, is that the joint density $p(\boldsymbol{x}_t, z_t; \boldsymbol{x}_{t+1}, z_{t+1}; ...\boldsymbol{x}_{t+T} z_{t+T})$ may vary with time, t.

2 Stationary Logistic Regression

In a logistic regression model the predicted class label, $\hat{Z}_t$, is generated according to

$$y_t = P(\hat{Z}_t = 1 \mid \boldsymbol{w}) = g(\boldsymbol{w}^T \boldsymbol{x}_t) \tag{1}$$

where $g(a_t)$ is the logistic function

$$g(a_t) = \frac{\exp(a_t)}{1 + \exp(a_t)} \tag{2}$$

and a_t is the 'activation' at time t, where $\boldsymbol{w}$ is a column vector of weights and $\boldsymbol{x}_t$ is a column vector of inputs. Note that this is a stationary model because the weight vector $\boldsymbol{w}$ is not time varying.

2.1 Steepest Descent Learning

The weights in a logistic regression model can be learnt using a steepest descent algorithm which maximises the likelihood, $L(z_t)$, of the class labels where

$$L(z_t) = P(\hat{Z}_t = z_t \mid \boldsymbol{w}) = y_t^{z_t}(1 - y_t)^{1-z_t} \tag{3}$$

Maximising the likelihood can be achieved by minimising the negative log likelihood or *cross entropy*

$$G(z_t, y_t) = -\ln L(z_t) = -[z_t \ln y_t + (1 - z_t)\ln(1 - y_t)] \tag{4}$$

The corresponding steepest descent learning rule is [2]

$$\hat{\boldsymbol{w}}_t = \hat{\boldsymbol{w}}_{t-1} + \alpha \boldsymbol{x}_t (z_t - y_t) \tag{5}$$

where α is the learning rate and z_t is the true class label. A problem with steepest descent learning for stationary logistic regression is that there are two conflicting requirements on the learning rate parameter; at the start of learning we wish α to be large so that learning is fast but at the end of learning α must be small to ensure that the learning process converges. Whilst it is true that α can be set to be proportional to $1/t$, methods based on this approach are entirely *ad hoc* and have practical difficulties e.g. how is the constant of proportionality chosen ?

This problem can be overcome by taking a Bayesian approach instead of a maximum likelihood approach. The Bayesian algorithm, as we shall see, posseses an implicit learning rate parameter which naturally decreases as learning progresses.

2.2 Bayesian Learning

In Bayesian learning we consider the evolution of the distribution of weight values, $p(\hat{\boldsymbol{w}})$. This distribution is often approximated as a Gaussian. Before data point $(\boldsymbol{x}_t, z_t)$ arrives the weights are distributed according to the prior distribution $N(\hat{\boldsymbol{w}}_{t-1}, \boldsymbol{\Sigma}_{t-1})$ where $N(\boldsymbol{m}, \boldsymbol{C})$ is a normal distribution with mean $\boldsymbol{m}$ and covariance $\boldsymbol{C}$. After arrival of the new data point this prior is updated to a posterior, $N(\hat{\boldsymbol{w}}_t, \boldsymbol{\Sigma}_t)$, according to Bayes rule.

The mean and covariance of this posterior can be evaluated using a variational approximation due to Jaakkola and Jordan [6], by Newton's method as described by Spiegelhalter and Lauritzen [15] or by the Extended Kalman Filter as described by Niranjan [12]. In this paper, we consider the latter two methods.

In Bayesian learning the prior weight distribution (ie. before the class label is observed and the weights are updated) implies a prior distribution on the activation $P(a_t) = N(\bar{a}_t, s_t^2)$ where $\bar{a}_t = \hat{\boldsymbol{w}}_{t-1}^T \boldsymbol{x}_t$ is the mean activation and s_t^2 is the variance of the activation given by

$$s_t^2 = \boldsymbol{x}_t^T \boldsymbol{\Sigma}_{t-1} \boldsymbol{x}_t \tag{6}$$

This gives rise to the concept of a 'moderated' output, $\tilde{y}_t$, defined as

$$\tilde{y}_t = P(\hat{Z}_t = 1) = \int P(\hat{Z}_t = 1 \mid a_t) p(a_t) da_t \tag{7}$$

This integral can be accurately approximated by [10]

$$\tilde{y}_t = g(K(s_t)\bar{a}_t) \tag{8}$$

where

$$K(s_t) = \left(1 + \frac{\pi s_t^2}{8}\right)^{-1/2} \tag{9}$$

The unmoderated output, y_t, is given by $y_t = g(\bar{a}_t)$. The 'moderation' changes the actual output, y_t, to a moderated output, $\tilde{y}_t$, which is nearer to 0.5 by an amount which is proportional to the prior uncertainty on the model parameters. Moderated outputs are typically better than unmoderated outputs in terms of the likelihood of predictions [10].

If the posterior distribution is approximated by a Gaussian, its mean and covariance can be found via Newton's method as formulated by Spiegelhalter and Lauritzen [15] and described in Appendix A. The update equations are

$$\boldsymbol{\Sigma}_{\boldsymbol{t}} = \boldsymbol{\Sigma}_{t-1} - \frac{y_t(1-y_t)}{1 + y_t(1-y_t)s_t^2} (\boldsymbol{\Sigma}_{t-1}\boldsymbol{x}_t)(\boldsymbol{\Sigma}_{t-1}\boldsymbol{x}_t)^T \tag{10}$$

$$\hat{\boldsymbol{w}}_t = \hat{\boldsymbol{w}}_{t-1} + \boldsymbol{\Sigma}_t \boldsymbol{x}_t (z_t - y_t) \tag{11}$$

A comparison of equation 11 with equation 5 shows that the *effective* learning rate is equal to the covariance matrix $\boldsymbol{\Sigma}_t$. We can therefore evaluate an average learning rate as $\frac{1}{p} Tr(\boldsymbol{\Sigma}_t)$ where p is the number of weights. The learning rate is therefore adaptive; as the classifier sees more data from the same stationary regime the covariance reduces which, in turn, reduces the learning rate.

Alternatively, we could use the Extended Kalman Filter (EKF) paradigm to approximate the posterior distribution, as described in Appendix B. This produces an update equation for the covariance which is identical to that obtained from Newton's method. But the update for the mean is different and is given by

$$\hat{\boldsymbol{w}}_t = \hat{\boldsymbol{w}}_{t-1} + \frac{\boldsymbol{\Sigma}_{t-1}}{1 + y_t(1-y_t)s_t^2} \boldsymbol{x}_t (z_t - y_t) \tag{12}$$

This is different to equation 11 in that learning will be slower for larger activation uncertainty, s_t^2, or output uncertainty, $y_t(1-y_t)$.

3 Nonstationary Logistic Regression

We now consider a nonstationary logistic regression model where the weights evolve dynamically and the classifications are made according to the usual logistic function

$$\boldsymbol{w}_t = \boldsymbol{w}_{t-1} + \boldsymbol{n}_t \tag{13}$$

$$y_t = g(\boldsymbol{w}_t^T \boldsymbol{x}_t) \tag{14}$$

where $\boldsymbol{n}_t$ is 'state' noise generated from a zero mean normal distribution with isotropic covariance matrix $q_t\boldsymbol{I}$. The important difference between this nonstationary model and the stationary model of equation 1 is that the weights $\boldsymbol{w}_t$ are now time-varying.

We now take a Bayesian-learning approach to the problem of estimating the weights online. The situation is exactly the same as for the stationary case except that the prior distribution is now $N(\hat{\boldsymbol{w}}_{t-1}, \boldsymbol{\Sigma}_{t-1} + q_t\boldsymbol{I})$. This is because we are adding state noise before the new observation is made (see equation 13). If we use the Newton updates to estimate the posterior distribution then the mean and covariance are given by equations 10, 11 and 6 except that $\boldsymbol{\Sigma}_{t-1}$ is replaced by $\boldsymbol{\Sigma}_{t-1} + q_t\boldsymbol{I}$. That is, our dynamic nonstationary learning rule is

$$\boldsymbol{\Sigma_t} = (\boldsymbol{\Sigma}_{t-1} + q_t\boldsymbol{I}) - \frac{y_t(1-y_t)}{1 + y_t(1-y_t)s_t^2}[(\boldsymbol{\Sigma}_{t-1} + q_t\boldsymbol{I})\boldsymbol{x}_t][(\boldsymbol{\Sigma}_{t-1} + q_t\boldsymbol{I})\boldsymbol{x}_t]^T \tag{15}$$

$$\hat{\boldsymbol{w}}_t = \hat{\boldsymbol{w}}_{t-1} + \boldsymbol{\Sigma}_t\boldsymbol{x}_t(z_t - y_t) \tag{16}$$

where s_t^2 is the variance of the activation and is given by

$$s_t^2 = \boldsymbol{x}_t^T(\boldsymbol{\Sigma}_{t-1} + q_t\boldsymbol{I})\boldsymbol{x}_t \tag{17}$$

The EKF learning rule is identical except that equation 16 is replaced by equation 12. The use of an isotropic covariance matrix for the state noise, $q_t\boldsymbol{I}$, assumes that the weight changes are of similar magnitude. Whilst this may be a reasonable assumption if we consider just the input weights, it may not be so reasonable if we consider the input weights *and* the bias weight. However, by scaling the input data to have zero mean the bias weights will fluctuate around zero. Alternatively, one could use a separate noise variance parameter for the bias weight or, indeed, have separate noise variance parameters for every weight as discussed in the context of prediction problems by DeFreitas and Niranjan [4].

3.1 Effective learning rate

The effective learning rate of the nonstationary rule is given by the covariance matrix $\boldsymbol{\Sigma}_t$. An average effective learning rate can therefore be evaluated as

$$< \alpha >= (1/p)Trace(\boldsymbol{\Sigma}_t) \tag{18}$$

where p is the number of weights. An alternative way of expressing the Newton update formula for $\boldsymbol{\Sigma}_t$ is

$$\boldsymbol{\Sigma_t}^{-1} = (\boldsymbol{\Sigma}_{t-1} + q_t\boldsymbol{I})^{-1} + y_t(1-y_t)\boldsymbol{x}_t\boldsymbol{x}_t^T \tag{19}$$

Equation 15 is related to this equation via the Sherman-Morrison-Woodbury matrix inversion formula (see, for example, page 50 in [5]). Inspection of the above equation shows that increasing q_t increases $\boldsymbol{\Sigma}_t$ which will, in turn, increase the learning rate.

4 Estimating state noise

We now describe an online method for updating the state noise parameter, q_t, based on maximising the 'evidence' of the observations. This approach has been used in the context of Kalman Filters and Extended Kalman Filters and was developed originally by Jazwinski [8] and has since been re-discovered by DeFreitas and Niranjan and placed in the context of regularization in neural networks [4].

We note, however, that maximising the evidence of the observations in a classification context, as opposed to a prediction context, may not be the optimal strategy. This is because, in the logistic regression model, the observations are class labels which are *binary random variables* (drawn from a binomial distribution with a 'hit' probability given by the output of the logistic unit). Each class label therefore contains much less information (at most 1 bit) than the equivalent observation in the prediction context (a Gaussian random variable). The resulting estimates of state noise variance are commensurately noisier.

We therefore also investigate an alternative strategy: maximising the evidence of *model predictions* which, we believe, is a more appropriate criterion for classification problems.

The state noise variance parameter, q_t, which maximise these criteria can be found by a line search procedure, which is also described in this section. A gradient ascent approach, described in Appendix E, has also been investigated but we do not use it further in this paper.

4.1 Maximising Evidence of Class Labels

The *evidence* for class label z_t is defined as the likelihood of class label z_t after the parameters of the model have been integrated out. That is

$$Ev(z_t) = P(\hat{Z}_t = z_t) = \int P(\hat{Z}_t = z_t \mid \boldsymbol{w}_t) p(\boldsymbol{w}_t) d\boldsymbol{w}_t \tag{20}$$

This may be re-written as

$$Ev(z_t) = P(\hat{Z}_t = z_t) = \tilde{y}_t^{z_t}(1 - \tilde{y}_t)^{(1-z_t)} \tag{21}$$

where $\tilde{y}_t = g(K(s_t)\bar{a}_t)$, $s_t^2 = \boldsymbol{x}_t^T(\boldsymbol{\Sigma}_{t-1} + q\boldsymbol{I})\boldsymbol{x}_t$ and $\bar{a}_t = \boldsymbol{w}_{t-1}^T\boldsymbol{x}_t$. The evidence is therefore seen to be a function of q and q can therefore be chosen so as to maximise the evidence (for example, by line search, as described in section 4.3). The evidence can be expressed in an alternative expanded form as shown in Appendix D which is used in the derivative of the gradient ascent approach.

Just as maximising the likelihood can be achieved by minimising the cross entropy between the targets and the model outputs (see section 2.1), maximising the evidence can be achieved by minimising the cross entropy between the targets and the *moderated* model outputs. For a window of independent samples $\{\boldsymbol{x}_t, z_t\}$ of length N the total moderated cross entropy is given by

$$G_N(z, \tilde{y}) = -\sum_{t=1}^{N}[z_t \ln \tilde{y}_t + (1 - z_t)\ln(1 - \tilde{y}_t)] \tag{22}$$

where s_t is given by equation 17 and $\bar{a}_t = \hat{\boldsymbol{w}}_{t-1}^T\boldsymbol{x}_t$.

4.2 Maximising Evidence of Updated Predictions

As the class label z_t is a "noisy" quantity, better estimation of state noise can be obtained if, instead of maximising the evidence of the new class label, we maximise the evidence of the model trained on the new class label. This can be achieved by minimising

$$G_N(\hat{y}, \tilde{y}) = -\sum_{t=1}^{N}[\hat{y}_t \ln \tilde{y}_t + (1 - \hat{y}_t)\ln(1 - \tilde{y}_t)] \tag{23}$$

where $\hat{y}_t = g(\boldsymbol{w}_t^T\boldsymbol{x}_t)$. Note that this 'updated prediction' is based on the updated weights whereas the moderated prediction is based on the weights from the previous time step (if this were not the case, the optimal value of q would be $q = 0$).

This criterion may also be understood from a 'variance-matching' perspective. The maximum evidence criterion corresponds to setting the state noise so that the observed prediction variance is matched by the estimated prediction variance. In the prediction context (see, for example, section 5.3.1 in [4]) these variances parameterise a Gaussian distribution whereas in the classification context the variances parameterise a binomial distribution. The evidence of *updated predictions* is maximised when the following variance matching criterion is minimised

$$V(\hat{y}, \tilde{y}) = \sum_{t=1}^{N} \hat{y}_t(1 - \hat{y}_t) - \tilde{y}_t(1 - \tilde{y}_t) \tag{24}$$

Note that the corresponding criterion for maximising the evidence of *class labels* contains the term $z_t(1 - z_t)$ which is zero. This provides another explanation of why we should maximise the evidence of updated predictions rather than class labels.

4.3 Line Search Method

The value of q which minimises $G_N(\hat{y}, \tilde{y})$ can be found with a line search algorithm. Line searches are iterative schemes which, for example, fit a quadratic polynomial to a cost function evaluated at three distinct points and the new parameter estimate is given as the minimum of that quadratic function. The next iteration uses this new point and two of the previous points. In this paper we use Brent's line search algorithm ([13], page 402).

The state noise can be estimated by moving a window along one sample at a time and performing a new line search. This is, however, rather computationally intensive and may be unnecessary as q is likely to evolve on a slower time-scale than the weights. We therefore choose to move the window along N samples at a time and then to re-estimate q ie. $q_t = q_{t-1}$ unless t is exactly divisible by N in which case q_t is re-estimated using line search.

From Appendix F we note that if the window size, N, is set too small then it is possible for the state noise estimates to become infinite. This therefore suggests both that reasonably large values of N should be used and that there may be a need to impose a maximum state noise estimate q_{max}.

In this paper we use the following heuristic for setting N and q_{max}. Firstly, different segments of the time series are used to train different logistic regression models using a batch method (eg. Bayesian logistic regression using the evidence formalism [10]). This allows weight vectors and covariance matrices to be estimated for each segment. We then calculate the difference in weight vector between consecutive segments and form an outer product. This acts as an estimate of the empirical covariance. We then subtract the estimated covariance from the empirical covariance and infer that whatever covariance remains is due to state noise (and not just parameter uncertainty from having finite data sets). The average diagonal entries correspond to estimated q values and we set q_{max} to the median value (a conservative but robust estimate). The window length N is then initially set to a small value, say, 10 and is increased if the line search estimate of q is greater than q_{max}.

In practice, we find that, as expected, maximising the evidence of *updated predictions* is much more robust to choices of N and q_{max} than is maximising the evidence of *class labels* and find that the above heuristic scheme is often not necessary. All of the numerical examples in this paper estimate the state noise variance by maximising the evidence of updated predictions.

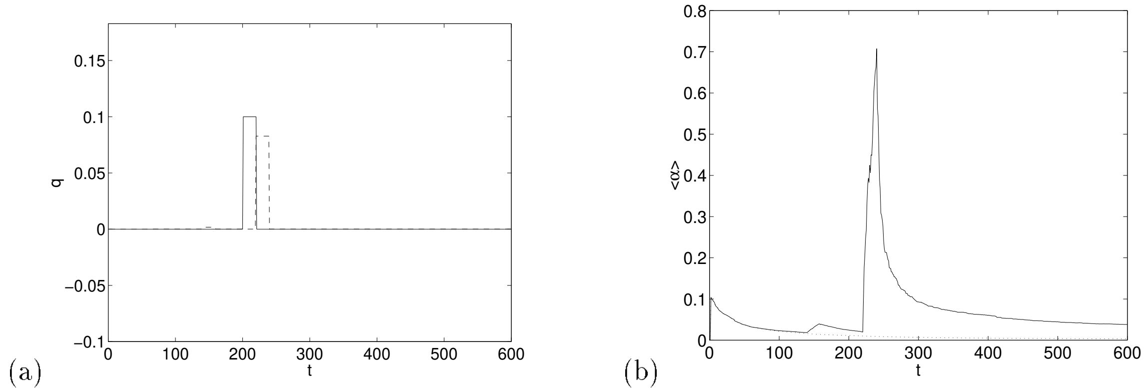

Figure 1: *(a) Estimate of state noise, q, and (b) effective learning rate of nonstationary rule (solid line) and stationary rule (dotted line). The nonstationary rule increases the effective learning rate in response to the change in classification regime at $t = 200$, whereas the effective learning rate for the stationary rule decreases monotonically as it believes that all data points belong to the same classification regime.*

5 Results

5.1 Continuous drift

We generated a 'continuous drift' data set as follows. Two Gaussians were first defined as having means $\boldsymbol{m}_1 = [1, 1]^T$ and $\boldsymbol{m}_2 = [-1, -1]^T$ and covariances $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma} = diag(v, v)$ and $N_D = 300$ data points were drawn from each Gaussian. The data were then randomly ordered to create a set of 600 input vectors, $\{\boldsymbol{x}_t\}$. The relations in Appendix C were then used to initialise the weights in a logistic regression model (assuming class 1 data points came from Gaussian 1 and class 2 from Gaussian 2). The variance v was chosen so that the 70% of the data could be correctly classified (the appropriate value of v was found via Monte Carlo simulation). The weights were then allowed to evolve according to the diffusion in equation 13 where the state noise at time t has isotropic variance, $q(t)$. Various profiles for $q(t)$ were tried, the one we report here being zero except between $t = 200$ and $t = 220$ where $q(t) = 0.1$. The true posterior probability at time t was then evaluated by passing the input at time t through a logistic node, $y_t = g(\boldsymbol{w}_t^T \boldsymbol{x}_t)$. Class labels, z_t, were then generated from this posterior probability.

We then trained logistic regression models using stationary, nonstationary and steepest descent learning algorithms. In the nonstationary model the state noise was estimated using the line search method to maximise the evidence of updated predictions.

Figure 1(a) shows the true state noise and the estimated state noise trajectories and Figure 1(b) shows the corresponding learning rates for the stationary and nonstationary algorithms. Figure 2 shows the true weights and the weights as estimated by the nonstationary rule and Figure 3 compares the cumulative prediction error (as measured by the cross entropy between moderated predictions and true class labels) with that obtained by the stationary and steepest descent algorithms.

The reason why nonstationary learning performs so much better is that the learning rate is increased when the new classification regime is encountered (see Figure 1(b)). The EKF approximation was preferred to Newton's method as it was less sensitive to overestimates of state noise (due to the denominator terms in equation 12).

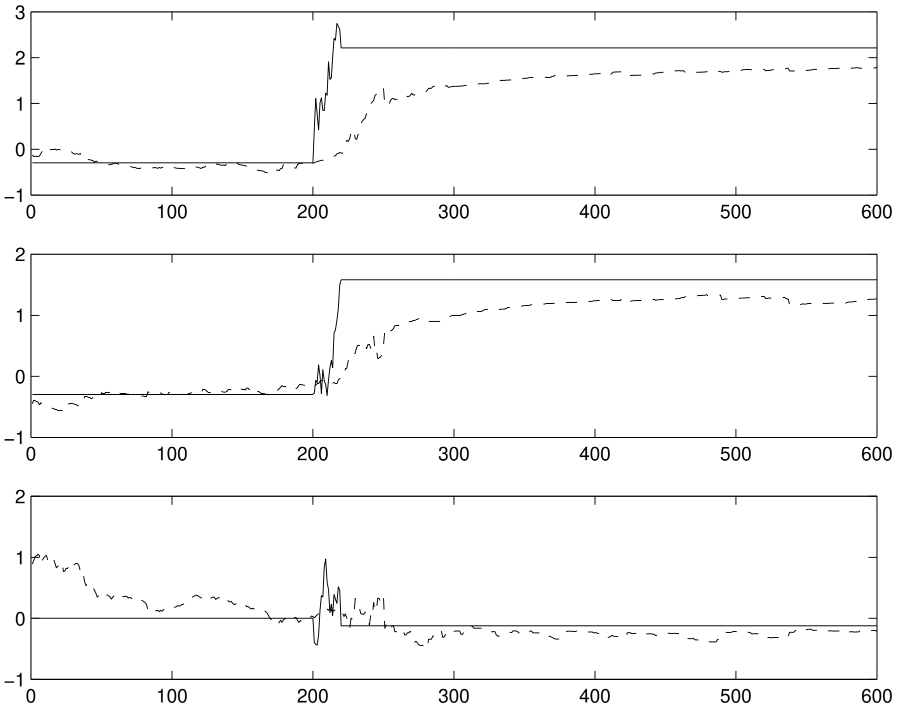

Figure 2: *Estimation of weights using nonstationary learning rule with the state noise estimated by the line search method; true weights (solid line), estimated weights (dashed line).*

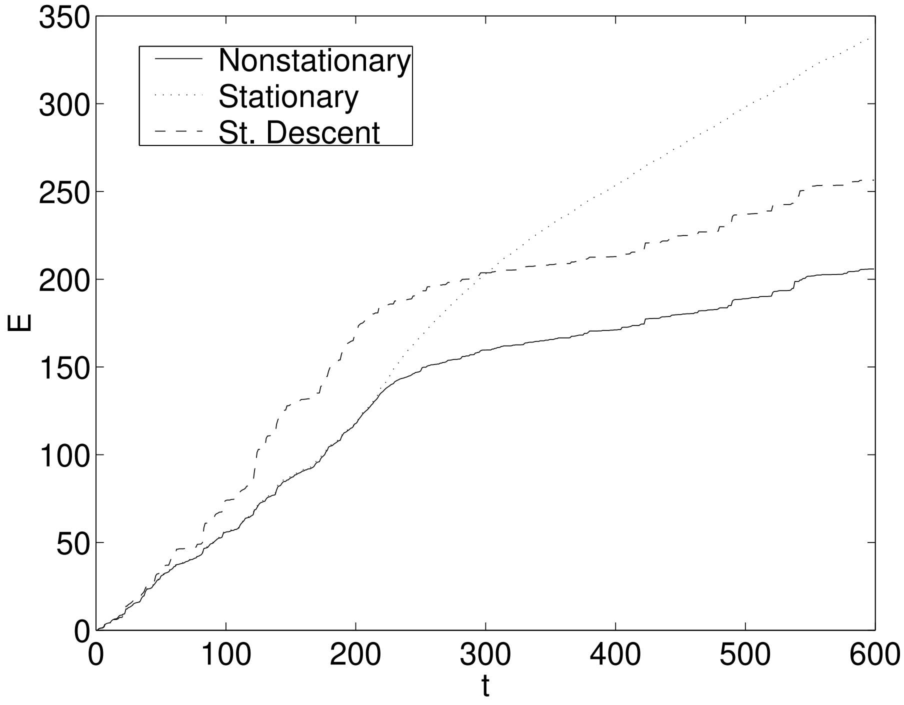

Figure 3: *Cumulative prediction error, E, of nonstationary, stationary and steepest descent learning rules on a data set which has a slow transition between classification regimes starting at $t = 200$.*

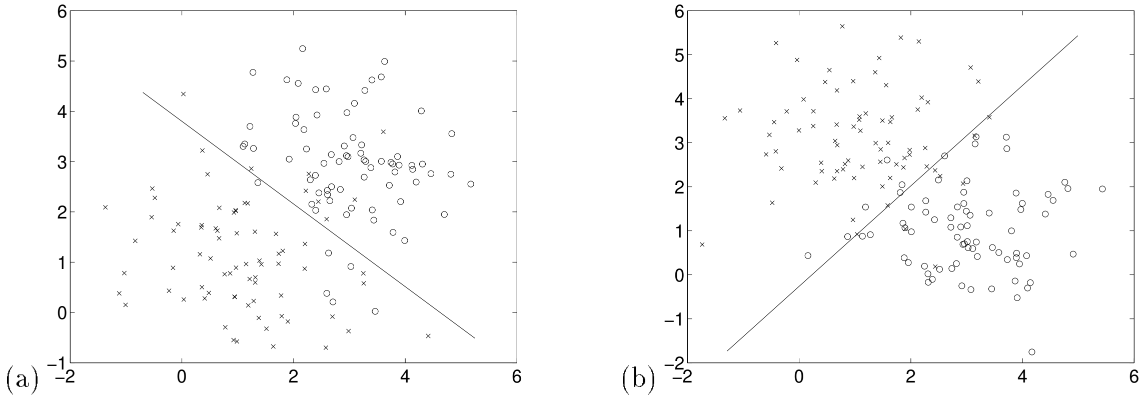

Figure 4: *Two data sets (a) A and (b) B used to form the discrete change data set.*

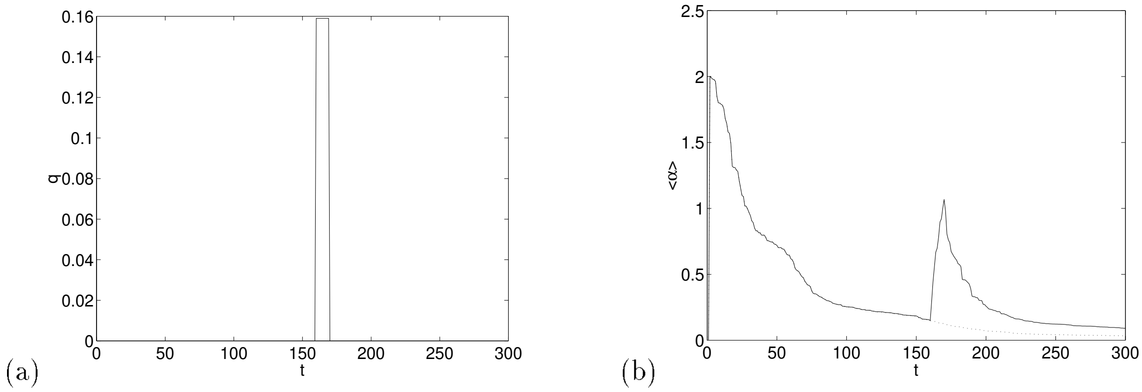

Figure 5: *(a) Estimate of state noise, q, and (b) effective learning rate of the nonstationary rule (solid line) and the stationary rule (dotted line). The nonstationary rule increases the effective learning rate in response to the change in classification regime at $t = 150$, whereas the effective learning rate for the stationary rule decreases monotonically as it believes that all data points belong to the same classification regime.*

5.2 Discrete change

We generated data sets A and B, shown in Figure 4. Each data set consists of examples drawn from one of two isotropic Gaussians corresponding to one of two class labels. Each data set is optimally separated by a linear decision boundary shown in the Figure. A data set was then formed by drawing 150 points from data set A followed by 150 points from data set B. Thus at $t = 150$ the optimal decision boundary switches from that shown in Figure 4(a) to that in Figure 4(b). Note that if the points in the overall data set were randomly ordered then it could only be classified with an accuracy of 50% ie. the chance rate. If, however, the temporal ordering is preserved near perfect classification is possible.

Figure 5(a) shows the estimated state noise trajectory and Figure 5(b) shows the corresponding learning rates for the stationary and nonstationary algorithms. Figure 6 compares the cumulative prediction error with that obtained by the stationary and steepest descent algorithms. Again, the nonstationary algorithm is seen to be the superior method as it is able to increase the implicit learning rate when a new classification regime is encountered.

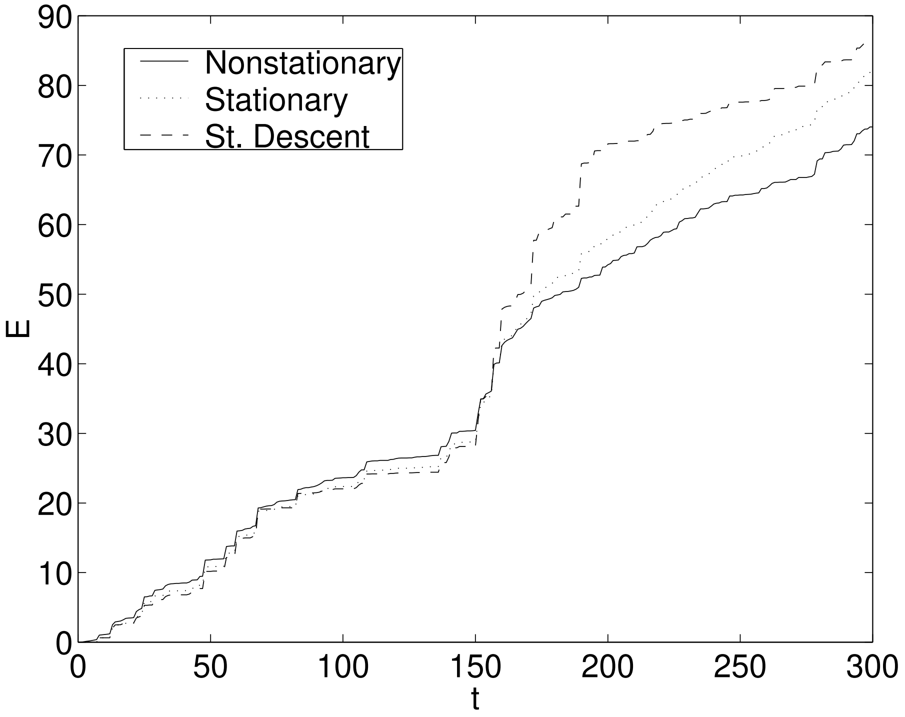

Figure 6: *Cumulative prediction error, E, of nonstationary, stationary and steepest descent learning rules on a data set which has a discrete transition between classification regimes at* $t = 150$.

6 Discussion

We have investigated an online learning algorithm for training a logistic regression model on nonstationary classification problems. The nonstationarity is captured by modelling the weights in a logistic regression classifier as evolving according to a first order Markov process. The weights were updated using the extended Kalman filter formalism. Similar approaches exist in the statistics literature and are known as Dynamic Generalised Linear Models (DGLIMs) [17, 16].

The nonstationarities were tracked by inferring a time-varying state noise variance parameter using a maximum evidence algorithm widely used for nonstationary prediction problems [7, 4] but adapted to the classification context by choosing to maximise the evidence of updated predictions rather than of observations.

An alternative approach to estimating state-noise parameters is outlined in Li and Bar-Shalom [9] where a mixture of models, each having a fixed state-noise value is employed. In its simplest incarnation two models would be used; one with zero state noise and one with a 'maximum' state noise. Estimates of state noise are then obtained by estimating mixing coefficients.

Our observation that state-noise estimation is easier if the targets are 'updated predictions', rather than binary class labels, suggests the following heuristic estimation scheme; if the model is tracking the data at all, then the estimated weights at time t should be closer to the true weights at time $t - N$ than are the estimated weights at time $t - N$. Therefore, for the purposes of estimating state noise, the model at time t could be used to generate pseudo-targets for previous time steps ie. $z_{t-N} \approx y(\boldsymbol{x}_{t-N}, \boldsymbol{w}_t)$. Experimentation with such an approach shows that good results can be obtained as long as some initial tracking ability is present in the model. This therefore suggests a multiple-pass algorithm where, firstly, a weight trajectory is estimated and then, secondly, a state noise trejectory is estimated and these two steps are iterated until a consistent solution is reached. Such a scheme is reminiscent of the forward and backward recursions involved in the EM (Expectation-Maximisation) estimation [14] of model parameters and seems a promising avenue for further research. However such 'smoothing' approaches (as opposed to filtering approaches) mean that the algorithms are no longer 'online'.

References

[1] Y. Bar-Shalom and X.R. Li. *Estimation and Tracking.* Artech House, Boston, MA., 1993.

[2] C. M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, Oxford, 1995.

[3] C. Chatfield. *The Analysis of Time Series: An Introduction.* Chapman and Hall, 1996.

[4] J.F.G. DeFreitas, M. Niranjan, and A.H. Gee. Hierarchical Bayesian-Kalman Models for Regularisation and ARD in Sequential Learning. Technical Report CUED/F-INFENG/TR 307, Department of Engineering, Cambridge University, 1998. Also avaiable from http://svr-www.eng.cam.ac.uk/ jfgf/publications.html.

[5] G. H. Golub and C. F. Van Loan. *Matrix Computations.* John Hopkins University Press, 3rd edition, 1996.

[6] T. S. Jaakkola and M. I. Jordan. A variational approach to Bayesian logistic regression models and their extensions. Technical Report 9702, MIT Computational Cognitive Science, January 1997.

[7] A. Jazwinski. *Stochastic Processes and Filtering Theory.* Academic Press, 1970.

[8] A.H. Jazwinski. Adaptive Filtering. *Automatica*, 5:475–485, 1969.

[9] X.R. Li and Y. Bar-Shalom. A recursive multiple model approach to noise identification. *IEEE Transactions on Aerospace and Electronic Systems*, 30(3):671–684, 1994.

[10] D.J.C. Mackay. The evidence framework applied to classification networks. *Neural Computation*, 4(5):720–736, 1992.

[11] R.J. Meinhold and N.D. Singpurwalla. Understanding the Kalman Filter. *The American Statistician*, 37(2):123–127, May 1983.

[12] M. Niranjan. Risk prediction in pregnancy. Talk at the Isaac Newton Institute, November 1997.

[13] W. H. Press, S.A. Teukolsky, W.T. Vetterling, and B.V.P. Flannery. *Numerical Recipes in C, second edition.* Cambridge, 1992.

[14] S. Roweis and Z. Ghahramani. A unifying review of linear gaussian models. *Neural Computation*, 11(2):305–346, 1999.

[15] D. J. Spiegelhalter and S. L. Lauritzen. Sequential updating of conditional probabilities on directed graphical structures. *Networks*, 20:579–605, 1990.

[16] M. West and J. Harrison. *Bayesian Forecasting and Dynamic Models.* Springer, 2nd edition, 1997.

[17] M. West, J. Harrison, and H.S. Migon. Dynamic Generalized Linear Models and Bayesian Forecasting. *Journal of the American Statistical Association*, 80(389):73–83, 1985.

[18] Z. Zhang. Parameter Estimation Techniques: A Tutorial with Application to Conic Fitting. *Image and Vision Computing Journal*, 15(1):59–76, 1997.

7 Acknowledgements

W. D. Penny was supported by the UK Engineering and Physical Sciences Research Council (EPSRC), grant number GR/K79062. This research has been strongly motivated by the 1998 paper by DeFreitas and Niranjan [4]. We would also like to acknowledge many useful conversations with Richard Everson about nonstationary models and with Dirk Husmeier about line search optimisation and Bayesian approaches.

A Newton's method

The Spiegelhalter-Lauritzen (SL) learning rule is derived by approximating the posterior distribution with a Gaussian. This amounts to approximating the log posterior with a quadratic around $\hat{\boldsymbol{w}}_{t-1}$. The new covariance matrix is given by the negative of the inverse Hessian (second derivative of the log posterior) at $\hat{\boldsymbol{w}}_{t-1}$. The weights are then updated according to Newton's method (see, for example, [13], page 426). The resulting formulae are [15]

$$\boldsymbol{\Sigma_t} = \boldsymbol{\Sigma}_{t-1} - \frac{y_t(1-y_t)}{1+y_t(1-y_t)s_t^2}(\boldsymbol{\Sigma}_{t-1}\boldsymbol{x}_t)(\boldsymbol{\Sigma}_{t-1}\boldsymbol{x}_t)^T \tag{25}$$

$$\hat{\boldsymbol{w}}_t = \hat{\boldsymbol{w}}_{t-1} + \boldsymbol{\Sigma}_t\boldsymbol{x}_t(z_t - y_t) \tag{26}$$

B The Extended Kalman Filter

Consider the following nonlinear dynamical system

$$\boldsymbol{w}_t = h(\boldsymbol{w}_{t-1}) + \boldsymbol{n}_t \tag{27}$$

$$y_t = f(\boldsymbol{w}_t) + e_t \tag{28}$$

where, generally, h and f are nonlinear functions, $\boldsymbol{n}_t$ is Gaussian noise with covariance matrix $\boldsymbol{Q}$ and e_t is Gaussian noise with variance v_t. If we restrict the system such that h is the identity function and we linearize the observation function according to a first order Taylor series expansion

$$f(\boldsymbol{w}_t) = f(\boldsymbol{w}_{t-1}) + \boldsymbol{G}_t\boldsymbol{w}_t \tag{29}$$

where $\boldsymbol{G}_t = \frac{\partial f}{\partial \boldsymbol{w}_t} |_{\boldsymbol{w}_{t-1}}$ then the extended Kalman filter update equations are [1]

$$\boldsymbol{\Sigma}_t = (\boldsymbol{\Sigma}_{t-1} + \boldsymbol{Q}_t) - \boldsymbol{K}_t\boldsymbol{G}_t^T(\boldsymbol{\Sigma}_{t-1} + \boldsymbol{Q}_t) \tag{30}$$

$$\hat{\boldsymbol{w}}_t = \hat{\boldsymbol{w}}_{t-1} + \boldsymbol{K}_t(z_t - \hat{y}_t) \tag{31}$$

where

$$\boldsymbol{K}_t = (\boldsymbol{\Sigma}_{t-1} + \boldsymbol{Q}_t)\boldsymbol{G}_t\left[v_t + \boldsymbol{G}_t^T(\boldsymbol{\Sigma}_{t-1} + \boldsymbol{Q}_t)\boldsymbol{G}_t\right]^{-1} \tag{32}$$

and the prediction at time t is given by $\hat{y}_t = f(\hat{\boldsymbol{w}}_{t-1})$, $\boldsymbol{K}_t$ is referred to as the Kalman gain matrix. Note that if h and f are linear then the standard Kalman filter updates [11] suffice.

Niranjan [12] notes that a version of the Extended Kalman Filter (EKF) may also be used for online learning in a logistic regression model. The learning algorithm can be derived by making the appropriate substitutions in the above EKF equations. If the state noise is set to zero ($\boldsymbol{Q}_t = 0$), f is set to the logistic function and the observation noise v_t is set to $y_t(1-y_t)$ (we comment more on this later) then

$$\boldsymbol{G}_t = y_t(1-y_t) \tag{33}$$

and

$$\boldsymbol{K}_t = \frac{\boldsymbol{\Sigma}_{t-1}}{1 + y_t(1-y_t)s_t^2}\boldsymbol{x}_t \tag{34}$$

where

$$s_t^2 = \boldsymbol{x}_t^2 \boldsymbol{\Sigma}_{t-1}\boldsymbol{x}_t \tag{35}$$

and the update equation for the covariance matrix is identical to equation 10. The corresponding EKF update equation for the weights is

$$\hat{\boldsymbol{w}}_t = \hat{\boldsymbol{w}}_{t-1} + \frac{\boldsymbol{\Sigma}_{t-1}}{1 + y_t(1-y_t)s_t^2}\boldsymbol{x}_t(z_t - y_t) \tag{36}$$

The EKF uses a first order Taylor series expansion of the observation function about the old weight values. A modified version of the EKF called the Iterated EKF (IEKF) (see, for example, [18]) aims to iteratively improve the accuracy of this expansion by first calculating the derivative of the observation function, f', at the old weight values and then re-calculating it at the new weight values. This procedure continues until convergence. Note also that the iteration process is used for the weight updates and not for the covariance updates. For our case, where f is the logistic function, the component of the derivative which enters our weight update equation occurs in the denominator of equation 36 as $y_t(1-y_t)$. The corresponding component in an IEKF scheme would converge to a value $z_t(1-z_t)$ which equals zero (as the class labels are all either one or zero). Hence the IEKF update equation for the weights is

$$\hat{\boldsymbol{w}}_t = \hat{\boldsymbol{w}}_{t-1} + \boldsymbol{\Sigma}_{t-1}\boldsymbol{x}_t(z_t - y_t) \tag{37}$$

Hence, the IEKF learning rule for logistic regression is identical to the SL learning rule except that the weight updates in IEKF use the old covariance matrix whereas SL uses the new covariance matrix (compare equations 11 and 37).

As we remarked earlier, EKF/IEKF can be reduced to to an online learning rule for logistic regression if it is assumed that the observation noise is $y_t(1-y_t)$. This make sense as $y_t(1-y_t)$ is the variance of the binomial prediction, y_t and the true classification may, in some sense, be regarded as the prediction corrupted by this binomial noise.

To reduce the EKF/IEKF to online logistic regresssion we also assumed there was no state noise, i.e. $\boldsymbol{Q}_t = 0$. If, however, we assume a non-zero $\boldsymbol{Q}_t$ then we can apply the EKF/IEKF to the case of dynamic logistic regression (see next section).

C Logistic Regression for Gaussian inputs

Bishop [2] pp. 82-83 notes that if each class (in a two-class classification problem) is assumed Gaussian and the Gaussians have means $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$ and the *same* covariance matrix $\boldsymbol{\Sigma}$ then the weights of a logistic regression model may be set analytically as follows

$$\boldsymbol{w} = \boldsymbol{\Sigma}^{-1}(\boldsymbol{m}_1 - \boldsymbol{m}_2) \tag{38}$$

and

$$w_0 = -\boldsymbol{m}_1^T \boldsymbol{\Sigma}^{-1} \boldsymbol{m}_1 + \boldsymbol{m}_2^T \boldsymbol{\Sigma}^{-1} \boldsymbol{m}_2 + \ln \frac{P(C_1)}{P(C_2)} \tag{39}$$

where $\boldsymbol{w}$ are the weights connected to inputs, w_0 is the bias weight and $P(C_1)$ and $P(C_2)$ are the prior probabilities of class 1 and 2.

We now ask what effect a change in mean or a change in covariance has on the corresponding logistic regression model.

C.1 Change of mean

If we assume that $\boldsymbol{\Sigma}$ and $\boldsymbol{m}_2$ remain fixed but $\boldsymbol{m}_1$ changes as follows

$$\boldsymbol{m}_1(t) = \boldsymbol{m}_1(t-1) + \delta \mathbf{1} \tag{40}$$

where $\mathbf{1}$ is a vector of 1's then, for the input weights,

$$\boldsymbol{w}(t) = \boldsymbol{w}(t-1) + \delta \boldsymbol{\Sigma}^{-1} \mathbf{1} \tag{41}$$

and for the bias weight

$$w_0(t) = w_0(t-1) - \frac{1}{2} \delta^2 \mathbf{1}^T \boldsymbol{\Sigma}^{-1} \mathbf{1} \tag{42}$$

That is, the input weights increase but the bias weight decreases. The signs are opposite for an increase in $\boldsymbol{m}_2$ or a decrease in $\boldsymbol{m}_1$.

C.2 Change of covariance

If we assume that $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$ remain fixed but $\boldsymbol{\Sigma}$ changes as follows

$$\boldsymbol{\Sigma}(t) = \boldsymbol{\Sigma}(t-1) + k\boldsymbol{I} \tag{43}$$

and that $\boldsymbol{u}_i$ and λ_i are the eigenvectors and eigenvalues of $\boldsymbol{\Sigma}(t-1)$ we can write

$$\boldsymbol{m}_1 = \sum_i a_i \boldsymbol{u}_i \tag{44}$$

$$\boldsymbol{m}_2 = \sum_i b_i \boldsymbol{u}_i \tag{45}$$

$$\boldsymbol{m}_1 - \boldsymbol{m}_2 = \sum_i (a_i - b_i) \boldsymbol{u}_i \tag{46}$$

We can then write the input weights in the corresponding logistic regression model as

$$\boldsymbol{w}(t-1) = \sum_i \frac{a_i - b_i}{\lambda_i} \boldsymbol{u}_i \tag{47}$$

$$\boldsymbol{w}(t) = \sum_i \frac{a_i - b_i}{\lambda_i + k} \boldsymbol{u}_i \tag{48}$$

$$\tag{49}$$

and the bias weights as

$$w_0(t-1) = \sum_i \frac{b_i^2 - a_i^2}{\lambda_i} \tag{50}$$

$$w_0(t) = \sum_i \frac{b_i^2 - a_i^2}{\lambda_i + k} \tag{51}$$

$$\tag{52}$$

By using a power series approximation truncated to first order

$$\frac{1}{\lambda+k} = \frac{1}{\lambda}(1+\frac{k}{\lambda})^{-1} \approx \frac{1}{\lambda} - \frac{k}{\lambda^2} \tag{53}$$

which is valid for a small relative increase in variance (small k/λ) we can write the following recursions

$$\begin{aligned} \boldsymbol{w}(t) &\approx \boldsymbol{w}(t-1) - k\sum_i \frac{a_i - b_i}{\lambda_i^2}\boldsymbol{u}_i && (54)\\ w_0(t) &\approx w_0(t-1) - k\sum_i \frac{b_i^2 - a_i^2}{\lambda_i^2} && (55)\\ & && (56) \end{aligned}$$

That is, a small increase in variance (small k) leads to a decrease in weights in the corresponding logistic regression model.

Note that even for isotropic (same in each direction) changes in mean and covariance the resulting changes in weight values are not isotropic.

If, however, we restrict ourselves to looking at the input weights and the the covariance is isotropic ($\lambda_i = \lambda$ and u_i are the unit vectors) then isotropic changes in mean lead to isotropic changes in weights. If we make the further assumption that the weights are also isotropic (then $a_1 = a_2$, $b_1 = b_2$) then isotropic changes in covariance will cause isotropic changes in weights.

Even so, changes in the bias weight are not of the same magnitude as changes in input weights.

D Likelihood and evidence

Because $P(\hat{Z}_t = 1) = y_t$ and $P(\hat{Z}_t = 0) = 1 - y_t$, the *likelihood* of class label z_t can be written as

$$P(\hat{Z}_t = z_t \mid \boldsymbol{w}) = \frac{\exp(z_t \boldsymbol{w}^T \boldsymbol{x}_t)}{1+\exp(\boldsymbol{w}^T \boldsymbol{x}_t)} \tag{57}$$

The *evidence* for class label z_t is defined as the likelihood of class label z_t after the parameters of the model have been integrated out. That is

$$Ev(z_t) = P(\hat{Z}_t = z_t) = \int P(\hat{Z}_t = z_t \mid \boldsymbol{w}) p(\boldsymbol{w}) d\boldsymbol{w} \tag{58}$$

This can be re-written as an integral over the corresponding activations

$$Ev(z_t) = P(\hat{Z}_t = z_t) = \int P(\hat{Z}_t = z_t \mid a_t) p(a_t) da_t \tag{59}$$

which can be accurately approximated by [10]

$$Ev(z_t) = \frac{\exp(z_t K(s_t)\bar{a}_t)}{1+\exp(K(s_t)\bar{a}_t)} \tag{60}$$

where $\bar{a}_t = \boldsymbol{w}_{t-1}^T \boldsymbol{x}_t$ and $s_t^2 = \boldsymbol{x}_t^T(\boldsymbol{\Sigma}_{t-1} + q_t \boldsymbol{I})\boldsymbol{x}_t$. The log evidence is given by

$$\log Ev(z_t) = z_t K(s_t)\bar{a}_t - \log[1+\exp(K(s_t)\bar{a}_t)] \tag{61}$$

E Gradient ascent estimation of state noise

By the chain rule

$$\frac{\partial \log Ev(z_t)}{\partial q_t} = \frac{\partial \log Ev(z_t)}{\partial K(s_t)} \frac{\partial K(s_t)}{\partial s_t} \frac{\partial s_t}{\partial q_t} \tag{62}$$

We can simply write $K'(s) = \frac{\partial K(s_t)}{\partial s_t}$ and the other derivatives can be evaluated as

$$\frac{\partial \log Ev(z_t)}{\partial K(s_t)} = (z_t - \tilde{y}_t)\bar{a}_t \tag{63}$$

where $\tilde{y}_t$ is the 'moderated' prediction given by

$$\tilde{y}_t = \frac{\exp(K(s)\bar{a}_t)}{1 + \exp(K(s)\bar{a}_t)} \tag{64}$$

and differentiating equation 17 gives

$$\frac{\partial s_t}{\partial q_t} = \frac{\boldsymbol{x}_t^T \boldsymbol{x}_t}{2 s_t^2} \tag{65}$$

Combining terms gives

$$\frac{\partial \log Ev(z_t)}{\partial q_t} = \frac{(z_t - \tilde{y}_t)\bar{a}_t K'(s) \boldsymbol{x}_t^T \boldsymbol{x}_t}{2 s_t^2} \tag{66}$$

If we substitute s_t^2 from equation 17 we get

$$\frac{\partial \log Ev(z_t)}{\partial q_t} = \frac{(z_t - \tilde{y}_t)\bar{a}_t K'(s) \boldsymbol{x}_t^T \boldsymbol{x}_t}{2 \boldsymbol{x}_t^T (\boldsymbol{\Sigma}_{t-1} + q_t \boldsymbol{I}) \boldsymbol{x}_t} \tag{67}$$

Now, $K'(s)$ is given by

$$K'(s) = -\frac{\pi s}{8} K(s)^3 \tag{68}$$

which is always negative and $\boldsymbol{x}_t^T \boldsymbol{x}_t$ is always positive. Therefore, $(z_t - \tilde{y}_t)\bar{a}_t$ must be negative if the gradient is to be positive. This can only happen if the new data point is misclassified. Therefore, if the new data point is misclassified $\frac{\partial \log Ev(z_t)}{\partial q_t} = 0$ at $q_t = +\infty$.

However, as noted above, if a data point is misclassified the maximum evidence value of q_t is $+\infty$. This gives rise to a moderated output of 0.5 (the closest it can be to the true class label given a misclassification). A maximum evidence update scheme would therefore set q_t to $+\infty$ every time there is a misclassification which is clearly undesirable.

A better approach would be to change the state noise so that the evidence of the new data point is *increased* rather than maximised. This can be achieved by a gradient ascent approach

$$q_t = q_{t-1} + \rho \frac{\partial \log Ev(z_t)}{\partial q_t} \tag{69}$$

where ρ is a learning rate parameter for the state noise. By substituting the gradient derived above we get

$$q_t = q_{t-1} + \rho \frac{(z_t - \tilde{y}_t)\bar{a}_t K'(s) \boldsymbol{x}_t^T \boldsymbol{x}_t}{2 s_t^2} \tag{70}$$

The state noise is increased if there is a misclassification. Also, we note that the state noise can be decreased for a correct classification. These effects are shown in Figure 7. Alternatively, we can estimate the gradient over a series of N samples

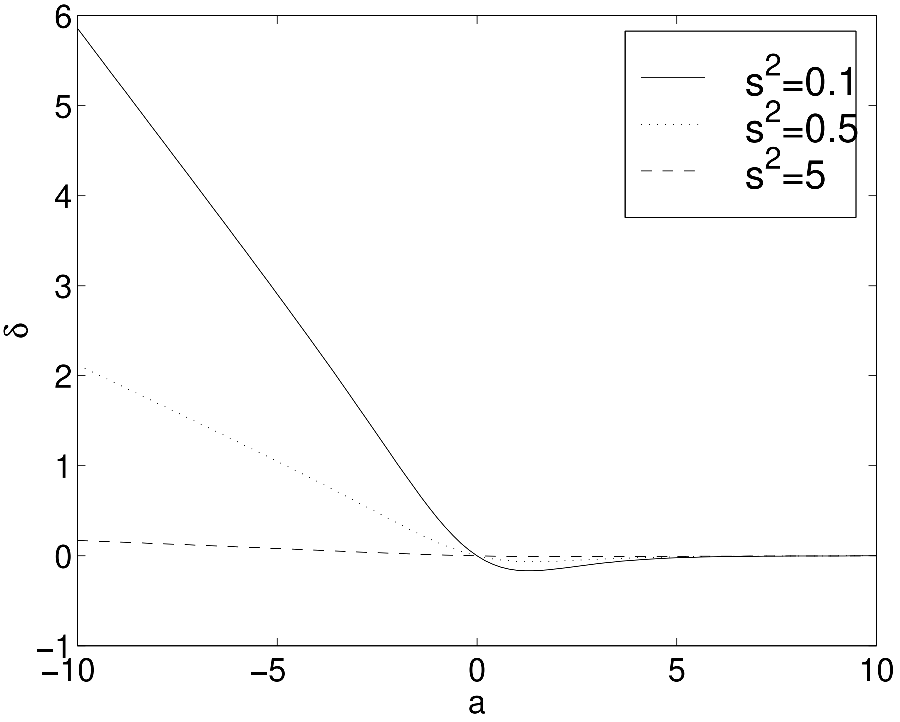

Figure 7: The figure shows $\delta = \frac{\partial LogEv}{\partial q_t}$ versus the activation, $\bar{a}$, for three different values of activation variance, s^2. The true class label is $z = 1$ and $\boldsymbol{x}_t^T \boldsymbol{x}_t = 1$. Only if the new data point is misclassified, i.e. $\bar{a} < 0$, is the gradient positive and the state noise increased. This increase is larger if the present activation variance is small, ie. there is currently little moderation of network predictions. The interval $0 \leq \bar{a} \leq 2$ shows that if the data point is correctly classified the state noise can be decreased.

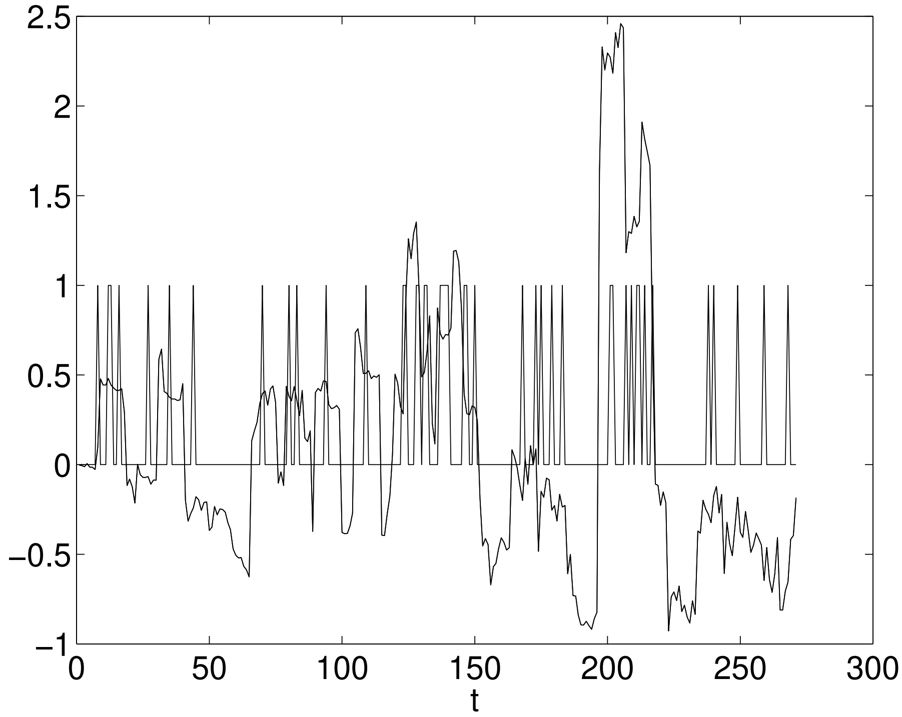

Figure 8: The spiky line shows misclassifications as 1's and correct classifications as 0's. The other line is the gradient $\delta = \frac{\partial LogEv}{\partial q_t}$ calculated over $N = 50$ samples centred at t. The gradient becomes more positive as the number of misclassifications increases. In a gradient ascent scheme this will cause the state noise, q, to increase which will, in turn, increase the learning rate.

$$\frac{\partial \log Ev}{\partial q} = \sum_{t=1}^{N} \frac{\partial \log Ev(z_t)}{\partial q_t} \tag{71}$$

and then update using

$$q_t = q_{t-1} + \rho \frac{\partial \log Ev}{\partial q} \tag{72}$$

Figure 8 shows the gradient calculated on a synthetic data set. It shows that where there are a large number of misclassifications the gradient ascent scheme will increase the state noise.

But gradient ascent approaches have a problem. How do we set ρ ? An alternative procedure is to set q to a value which will maximise the evidence of a number of data points using a line search method thus obviating the need for ρ.

A second reason for using line search rather than gradient ascent is that Mackay's approximation (equation 8) breaks down for large values of activation and the breakdown is more critical for the derivative than for the value itself (see for example plots 1(c) and 1(d) in [10]). We have also verified this ourselves using Monte Carlo (MC) simulation and central differences (CD); MC evaluation of the integral in equation 8 does not improve the estimation of q using the line search method but MC and CD evaluation of the gradient *does* improve the estimation of dEv/dq. This is because $K(s)$ is obtained from a series approximation and $K'(s)$ is therefore estimated from one less term in the series (this suggests using a better analytic approximation for $K'(s)$).

F Line search estimation of state noise

F.1 Choice of window size

We now consider the case of $N = 2$ to illustrate a possible problem with the line search method. If both examples are correctly classified then maximum evidence is obtained for zero moderation i.e. $q = 0$. If, however, both examples are misclassified then, as was the case with $N = 1$, maximum evidence is obtained for moderated outputs of 0.5, i.e. $q = \infty$. Further, if one example is correctly classified and the other is misclassified then it is also possible for the maximum evidence value to be $q = \infty$. We now give an example to show how this can occur. Firstly, we note that

$$\frac{\partial \log Ev(z_t)}{\partial s_t} = (z_t - \tilde{y}_t)\bar{a}_t K'(s_t) \tag{73}$$

If we assume that $z_1 = z_2 = 0$, $\boldsymbol{x}_1^T \boldsymbol{\Sigma}_1 \boldsymbol{x}_1 = \boldsymbol{x}_2^T \boldsymbol{\Sigma}_2 \boldsymbol{x}_2 = u$ and $\boldsymbol{x}_1^T \boldsymbol{x}_1 = \boldsymbol{x}_2^T \boldsymbol{x}_2 = v$ then $s_t = u + qv$ and for maximum evidence

$$-\tilde{y}_1 \bar{a}_1 - \tilde{y}_2 \bar{a}_2 = 0 \tag{74}$$

Hence,

$$\frac{\tilde{y}_1}{\tilde{y}_2} = \frac{-\bar{a}_2}{\bar{a}_1} \tag{75}$$

We now assume that the first example is incorrectly classified but the second example is correctly classified (obviously the same reasoning applies if it were the other way around). Because $z_1 = z_2 = 0$ this means that $\bar{a}_1 > 0$ and $\bar{a}_2 < 0$. Therefore, for maximum evidence we require

$$\frac{\tilde{y}_1}{\tilde{y}_2} = \frac{\mid \bar{a}_2 \mid}{\mid \bar{a}_1 \mid} \tag{76}$$

Note that $\bar{a}_1 > 0$ and $\bar{a}_2 < 0$ also implies $y_1 > y_2$. Now, if (a) we are as confident about the correct classification as we are about the incorrect classification then $\mid \bar{a}_2 \mid = \mid \bar{a}_1 \mid$. This results in $\tilde{y}_1 = \tilde{y}_2 = 0.5$ ie. $q = \infty$. But if (b) we are less confident about the correct classification than we are about the incorrect classification then $\mid \bar{a}_2 \mid < \mid \bar{a}_1 \mid$. Hence, for maximum evidence we require $\tilde{y}_1 < \tilde{y}_2$. But because $y_1 > y_2$, $\tilde{y}_1$ cannot be made smaller than $\tilde{y}_2$ for any value of q. The best that can be achieved is $\tilde{y}_1 = \tilde{y}_2$ which is given by $q = \infty$. A third scenario (c) where we are more confident about the correct classification than we are about the incorrect classification leads to reasonable estimates of q.

For $N > 2$, if the misclassified examples have large activations in proportion to the correctly classified examples then it is also possible for the maximum evidence value to be $q = \infty$. However, as N is increased the likelihood of this occuring is reduced and scenario (c) dominates. That is, for some reasonable N the line search method should give good estimates of q.

The important conclusion of this section is that if N is too small it is possible that the esimated value of q will be ∞. Therefore, if the estimates of q are infinite we should increase N. A second precaution is to impose a maximum value on q, q_{max}, during the line search.

These problems are less severe if updated predictions, $\hat{y}_t$, are used instead of class labels, z_t (see section 4.2).