Machine Learning, 50, 159–173, 2003

Classification with Bayesian MARS

C.C. HOLMES c.holmes@ic.ac.uk
D.G.T. DENISON d.denison@ic.ac.uk
Department of Mathematics, Imperial College, London, SW7 2BZ, UK

Editors: Nando de Freitas, Christophe Andrieu, Arnaud Doucet

Abstract. We present a new method for classification using a Bayesian version of the Multivariate Adaptive Regression Spline (MARS) model of J.H. Friedman (*Annals of Statistics*, *19*, 1–141, 1991). Special attention is paid to the use of Markov chain Monte Carlo (MCMC) simulation to gain inference under the model. In particular we discuss three important developments in MCMC methodology. First, we describe the reversible jump MCMC algorithm of P.J. Green (*Biometrika, 82*, 711–732, 1995) which allows inference on a varying dimensional, possibly uncountable, model space. This allows us to consider MARS models of differing numbers and positions of splines. Secondly, we discuss marginalisation which is used to reduce the effective dimension of the parameter space under consideration. Thirdly, we describe the use of latent variables to improve the MCMC computation. Our methods are generic and can be applied to any basis function model including, wavelets, artificial neural nets and radial basis functions. We present examples to show that the Bayesian MARS classifier is competitive with other approaches on a number of benchmark data sets.

Keywords: Bayesian MARS, Markov chain Monte Carlo, latent variables, probit regression

1. Introduction

The multivariate adaptive regression spline (MARS) model of Friedman (1991) is a popular method for non-linear regression which provides a direct competitor to artificial neural networks (ANNs). Empirical evidence suggests it is very competitive with other methods such as ANNs on a variety of different data sets (Rasmussen, 1996). The original MARS model was designed for regression problems, however, extensions to classification problems are well documented, see for example Kooperberg, Bose, and Stone (1997).

In this article we present a Bayesian version of MARS, extending the work of Denison, Mallick, and Smith (1998), to deal with two class classification problems. Our Bayesian framework places a prior distribution on the whole of MARS model space where we treat the number of splines, as well as all other parameters, as unknown. Class decision boundaries are formed by integrating over the posterior model space which produces a form of model averaging. We show that this averaging has the effect of producing smooth decision boundaries from the individual non-smooth MARS models.

The integration takes place over a high dimensional model space and hence approximation techniques are required for inference. In particular we make use of Markov chain Monte Carlo (MCMC) methods which are used to draw samples of models from the posterior distribution. One of the aims of this paper, alongside the introduction of the Bayesian MARS classifier, is to highlight efficient MCMC procedures for implementing our method.

Specifically, we make use of the reversible jump MCMC algorithm of Green (1995) which allows inference on the varying dimensional, uncountable, model space of MARS. We formulate the model through the use of latent variables and the Probit link function that dramatically improves the efficiency of the MCMC computation. An important role is played by marginalising over the spline coefficients (or output weights) which is used to reduce the effective dimension of the model space. Our methods are generic and can be applied to any basis function model including, wavelets, artificial neural nets and radial basis functions. A Matlab version of the algorithm is available from the first author on request.

A key feature of our approach is that we accommodate uncertainty in the number of spline basis functions that make up the MARS model. Related work in the literature for regression problems includes the papers of Smith and Kohn (1996), Holmes and Mallick (1998), and Andrieu, de Freitas, and Doucet (2000). For classification, the use of smoothing splines with latent variables and probit link function is reported in Wood and Kohn (1998).

The rest of the paper is as follows. In Section 2 we present the Bayesian formulation of MARS and discuss implementation details for the binary classification problem. In Section 3 we provide examples that illustrate the relative performance of the model and support our argument for the use of latent variables. A brief discussion is given in Section 4.

2. Bayesian MARS for the 2 class classification problems

The conventional MARS model for regression makes predictions, $\hat{y}_i$, on a set of p predictor variables, $\boldsymbol{x}_i = \{x_{i1}, \ldots, x_{ip}\}$, using a linear combinations of tensor product splines

$$\hat{y}_i = \beta_0 + \sum_{j=1}^{k} \beta_j \prod_{l=1}^{z_j} \left(x_{id_{jl}} - \theta_{jl}\right)_{q_{jl}}, \tag{1}$$

where k is the number of spline bases, $\beta = \{\beta_1, \ldots, \beta_k\}$ are a set of spline coefficients (or output weights), z_j is the interaction level (or order) of the jth spline, θ_{jl} is a spline knot point, d_{jl} indicates which of the p predictors enters into the lth interaction of the jth spline, $d_{jl} \in \{1, \ldots, p\}$, and q_{jl} determines the orientation of the spline components, $q_{jl} \in \{+, -\}$, where $(a)_+ = \max(a, 0)$, $(a)_- = \min(a, 0)$. It will be useful in what follows to write the model in matrix notation,

$$\hat{Y} = \Theta(\boldsymbol{X})\boldsymbol{\beta}, \tag{2}$$

where $\boldsymbol{X}$ is the data set of $n \times p$ predictor variables, $\hat{Y}$ is the n vector of predictions, $\boldsymbol{\beta} = \{\beta_0, \ldots, \beta_k\}$ is the set of spline coefficients and $\Theta(\boldsymbol{X})$ is the $n \times (k+1)$ matrix of spline basis outputs,

$$\Theta = \begin{pmatrix} 1, & \prod_{l=1}^{z_1} \left(x_{1d_{1j}} - \theta_{1j}\right)_{q_{1j}}, & \cdots & \prod_{l=1}^{z_k} \left(x_{1d_{kl}} - \theta_{kl}\right)_{q_{kl}} \\ 1, & \prod_{l=1}^{z_1} \left(x_{2d_{1j}} - \theta_{1j}\right)_{q_{1j}}, & \cdots & \prod_{l=1}^{z_k} \left(x_{2d_{kl}} - \theta_{kl}\right)_{q_{kl}} \\ \vdots & \vdots & \ddots & \vdots \\ 1, & \prod_{l=1}^{z_1} \left(x_{nd_{1j}} - \theta_{1j}\right)_{q_{1j}}, & \cdots & \prod_{l=1}^{z_k} \left(x_{nd_{kl}} - \theta_{kl}\right)_{q_{kl}} \end{pmatrix}, \tag{3}$$

where the first column of $\Theta(\boldsymbol{X})$ corresponds to the intercept term β_0 and we define an individual spline basis function to represent one column of $\Theta(\boldsymbol{X})$. In matrix notation it becomes clear that MARS, just like any other basis function model, is simply a linear regression in a non-linear spline base. From now on we shall simply write $\Theta(\boldsymbol{X})$ as Θ.

In two class classification problems, $y_i \in \{0, 1\}$, and an invertible link function, g, is introduced that maps the real valued predictor onto the range (0, 1),

$$\begin{aligned} p(y_i = 1 \mid \boldsymbol{x}_i) &= g(\eta_i) \\ \eta_i &= \beta_0 + \sum_{j=1}^{k} \beta_j \prod_{l=1}^{z_j} \left(x_{i d_{ij}} - \theta_{jl}\right)_{\pm}, \end{aligned} \tag{4}$$

and in matrix form,

$$\begin{aligned} p(Y = 1 \mid \boldsymbol{X}) &= g(\boldsymbol{\eta}) \\ \boldsymbol{\eta} &= \Theta\boldsymbol{\beta}. \end{aligned} \tag{5}$$

By convention, researchers have tended to use the sigmoidal link function, $g(a) = \frac{1}{1+\exp(-a)}$, however, in the next section we show that from a Bayesian perspective considerable computational advantages are made by using the equally valid probit link function, $g(a) = \Phi(a)$, where Φ is the cumulative distribution of the standard normal $\Phi(a) = \int_{-\infty}^{a} \frac{1}{2\pi} \exp(-a^2/2)$.

In non-Bayesian MARS, the model parameters, $\mathcal{M} = \{k, \boldsymbol{\beta}, \boldsymbol{z}, \boldsymbol{d}, \boldsymbol{q}, \boldsymbol{\theta}\}$, are set to single optimal values, where $\boldsymbol{z} = \{z_1, \ldots, z_k\}$, $\boldsymbol{d} = \{d_{11}, \ldots, d_{kz_k}\}$, $\boldsymbol{q} = \{q_{11}, \ldots, q_{kz_k}\}$ and $\boldsymbol{\theta} = \{\theta_{11}, \ldots, \theta_{kz_k}\}$.

This optimisation is typically achieved by stepwise spline selection on the spline bases functions with the spline coefficients set by least squares. In conventional MARS, higher order spline interactions, $z_j > 1$, can only be included if the corresponding lower order splines are already present. We do not impose this constraint in our model, though we note that it would be possible to do so if the practitioners prior beliefs accorded. The cost function that is optimised in the stepwise selection is commonly taken to be a penalised likelihood that balances goodness of fit against model complexity, see Friedman (1991) and Kooperberg, Bose, and Stone (1997). In the next section we present an alternative Bayesian formulation.

2.1. Bayesian formulation

The fixing of $\mathcal{M}$ to a single set of values fails to capture a key component of uncertainty in the modelling process which can lead to over confident predictions (Draper, 1995). In a Bayesian framework prior distributions are placed on all unknown quantities including the number of splines k and a parameter that governs the prior on $\boldsymbol{\beta}$. It is useful to write the prior in factorised form which highlights conditional dependencies,

$$\begin{aligned} p(\boldsymbol{\theta}, \boldsymbol{q}, \boldsymbol{d}, \boldsymbol{z}, \boldsymbol{\beta}, v, k) = {} & p(\boldsymbol{\theta} \mid \boldsymbol{z}, \boldsymbol{d}, k) p(\boldsymbol{q} \mid \boldsymbol{z}, k) p(\boldsymbol{d} \mid \boldsymbol{z}, k) \\ & \times p(\boldsymbol{z} \mid k) p(\boldsymbol{\beta} \mid k, V) p(v \mid k) p(k). \end{aligned} \tag{6}$$

The specific form that we adopt for these prior distributions is as follows. We take the prior on the individual knot locations θ_{jl} to be uniform at the n data points $p(\theta_{jl} \mid d_{jl}) = U(x_{1d_{jl}}, x_{2d_{jl}}, \ldots, x_{nd_{jl}})$, where d_{jl} indicates which predictor is being used and $p(d_{jl})$ is uniform on the p predictors, $p(d_{jl}) = U(1, \ldots, p)$. The prior on the orientation is uniform, $p(q_{jl} = +) = p(q_{jl} = -) = 0.5$. The interaction level in each spline has the prior, $p(z_j) = U(1, \ldots, z_{\max})$, where $z_{\max}$ is the maximum level to be set by the user. The prior on the spline coefficients is taken to be normal with mean zero and variance v, $p(\boldsymbol{\beta}) = N(0, vI)$, where I is the identity matrix of suitable dimensions. To reduce sensitivity to the prior specification of v we adopt a conjugate hyper-prior distribution on v which we take to be an Inverse-Gamma density, $p(v^{-1}) = \mathit{Gamma}(.001, .1)$, which is sufficiently diffuse with a mean of 0.01 for v^{-1}. Finally, $p(k)$ is assigned an improper prior, $p(k) = U(1, \ldots, \infty)$ which indicates ignorance on the number of splines needed a priori. Hence the model has just one user set parameter, $z_{\max}$, for which we recommend a default setting in Section 3.

The prior distributions on the parameters are combined with information in the data via Bayes theorem and the likelihood function, which for the two class classification problem is taken to be Bernoulli, $p(Y \mid \mathcal{M}) = \prod_{i=1}^{n} g(\eta_i)^{y_i}[1 - g(\eta_i)]^{(1-y_i)}$, where $g(\eta_i)$ is from (4). This leads to marginal predictive distributions for new data being given as,

$$p(y_{\text{new}} = 1 \mid \boldsymbol{x}_{\text{new}}) = \sum_{k=1}^{\infty} \int p(y_{\text{new}} = 1 \mid \boldsymbol{x}_{\text{new}}, \mathcal{M}_k) p(\mathcal{M}_k \mid Y)\, d\mathcal{M}_k, \tag{7}$$

where $p(\mathcal{M}_k \mid Y)$ is the posterior probability and $\mathcal{M}_k$ indicates a MARS model with k splines. Clearly the integral in (7) is computationally and analytically intractable and some approximation method is required if we are to proceed. An elegant solution is provided by MCMC simulation which allows one to draw samples, $\{\mathcal{M}^{(1)}, \mathcal{M}^{(2)}, \ldots, \mathcal{M}^{(m)}\}$, from the full posterior distribution $p(\mathcal{M} \mid Y)$ and then approximate (7) by

$$p(y_{\text{new}} = 1 \mid \boldsymbol{x}_{\text{new}}) \approx \frac{1}{m} \sum_{j=1}^{m} p\big(y_{\text{new}} = 1 \mid \boldsymbol{x}_{\text{new}}, \mathcal{M}^{(j)}\big). \tag{8}$$

The approximation (8) converges to the true value (7) as $m \to \infty$. An immediate advantage of the MCMC approach is that only a single collection of samples is required to in order to make predictions at any data point $\boldsymbol{x}_{\text{new}}$. The MCMC procedure that we use is described in the next section.

2.2. *Implementation via Markov chain Monte Carlo simulation*

It is fair to say that the resurgence of interest in Bayesian methods is in no small part due to the introduction of MCMC simulation and the availability of fast computing devices. MCMC is an extremely general computer intensive method for generating samples from any distribution, however complex. Moreover, the procedures are very simple. To begin, one constructs a Markov chain which has a stationary (steady state) distribution that matches the distribution of interest. There are simple rules on how to do this. Then one runs an iterative computer simulation of the chain for a long enough time, generating a new sample at each

iteration. An initial portion of samples must be discarded as being unrepresentative of the steady state distribution and the rest are used to make inference on quantities of interest, such as the predictive distributions in (8) although much more information is available in the samples. For example, all of the marginal parameter distributions, such as $p(k \mid Y)$, are approximated by the values that appear in the retained samples.

One of the simplest MCMC procedures is the Metropolis-Hastings algorithm (Metropolis et al., 1953; Hastings, 1970). In this method, at each iteration of the simulation, the current state of the chain, $\mathcal{M}^{(t)}$, is updated using a proposal distribution $R(\cdot)$, that might be based on the current state. This produces a new model $\mathcal{M}^{(*)}$ which is then accepted with probability Q,

$$Q = \min\left\{1, \frac{p\left(\mathcal{M}^{(*)} \mid Y\right) S\left(\mathcal{M}^{(t)}, R, \mathcal{M}^{(*)}\right)}{p\left(\mathcal{M}^{(t)} \mid Y\right) S\left(\mathcal{M}^{(*)}, R, \mathcal{M}^{(t)}\right)}\right\}, \tag{9}$$

where $S(\mathcal{M}^{(a)}, R, \mathcal{M}^{(b)})$ is the probability of generating the state $\mathcal{M}^{(a)}$ given the current state $\mathcal{M}^{(b)}$ and proposal density R. If the state is accepted then we set the sample $\mathcal{M}^{(t+1)} = \mathcal{M}^{(*)}$ else the current state is retained, $\mathcal{M}^{(t+1)} = \mathcal{M}^{(t)}$. It is straightforward to show that the above procedure results in samples appearing in the chain with probability $p(\mathcal{M} \mid Y)$ as desired (Gilks, Richardson, & Spiegelhalter, 1996).

The theory of MCMC places few restrictions on the form of the proposal density R. However, dramatic variations in performance occur for different choices of R and it is vital that 'good' proposal distributions can be found in order to get reasonable results within realistic computer time (Gilks, Richardson, & Spiegelhalter, 1996).

For our purposes, we wish to sample from the posterior density $p(\mathcal{M} \mid Y)$ in order to make inference about marginal densities of interest such as predictive distributions. Conventional MCMC methods, such as the Metropolis-Hastings (MH) algorithm, are not applicable as we do not know the number of splines a priori. Therefore we use the variable dimension reversible jump sampler outlined in Green (1995). Reversible jump MCMC is an extension of the MH algorithm to variable dimensions. In our method, at each iteration the sampler updates the current state of the chain, which equates to a MARS model, by choosing one of the following proposals with equal probability

1. Add a new spline basis function to the model.
2. Remove one of the k existing spline bases from the model.
3. Alter an existing spline basis in the model (by altering one of it's knot points).

Following each move an update is made to the spline coefficients $\boldsymbol{\beta}$. The three move steps above are equivalent to adding, removing and altering a column of Θ in (3). The exact algorithm is listed in the Appendix and we note that the removal step is not attempted when $k = 0$. Using these update proposals the acceptance probability, following Green (1995), is of the same form as (9). Following the update to the model a new value for v, the prior variance of $\boldsymbol{\beta}$, is drawn from its conditional distribution, see the Appendix.

In RJMCMC methods for basis function models, such as MARS, the update to $\boldsymbol{\beta}$ is the critical step which determines the efficiency of the algorithm. A poor proposal distribution for $\boldsymbol{\beta}$ will cause the generated state $\mathcal{M}^{(*)}$ to have low posterior probability and hence acceptance rates can be very low. For example, consider the procedure of removing a spline

basis from the model, which is equivalent to removing a column of Θ in (3). If we do not alter the remaining $\boldsymbol{\beta}$ parameters then the acceptance probability will tend to be very low, as the remaining parameters are now ill-tuned to the data due to the strong collinearity that exists between the spline bases. The most efficient procedure would be to update $\boldsymbol{\beta}$ using the conditional distribution $p(\boldsymbol{\beta} \mid Y, k^*, \boldsymbol{\theta}^*, \boldsymbol{d}^*, \boldsymbol{q}^*, V^*)$, as in Denison, Mallick and Smith (1998) and Holmes and Mallick (2000). However, for classification models, unlike the regression models in Denison, Mallick, and Smith (1998) and Holmes and Mallick (2000), this conditional distribution is unknown. This then leaves us with a problem. How can one find 'good' proposal distributions to update the spline coefficients $\boldsymbol{\beta}$ given changes to the spline bases.

The solution we propose is to use latent variables, following the approach of Albert and Chib (1993). The concept behind latent variables in MCMC is to introduce an extra set of parameters into the model that leave the marginal (original) model distribution unchanged. The reason for introducing the extra variables is to improve the overall efficiency of the sampling algorithms.

For our method, we now augment the original model space with a extra set of n real valued parameters $\{\omega_1, \ldots, \omega_n\}$, one for each data point. We then redefine the model in (4) to be,

$$\begin{aligned} p(y_i = 1 \mid \omega_i) &= \begin{Bmatrix} 1 & \text{if } \omega_i > 0 \\ 0 & \text{otherwise} \end{Bmatrix} \\ \omega_i &= \beta_0 + \sum_{j=1}^{k} \beta_j \prod_{l=1}^{z_j} \left(x_{d_{jl}} - \theta_{jl}\right)_{q_{jl}} + \epsilon_i, \end{aligned} \tag{10}$$

where ϵ_i is a standard normal random variable $\epsilon_i \sim N(0, 1)$. It is then straightforward to show that marginally, y_i, is equivalent for the two models: (4) with the probit link and (10) with the latent variables ϵ. This is clearly seen as in (10),

$$p(y_i = 1) = p(\omega_i > 0) = p(\epsilon_i > -m) = \Phi(m), \tag{11}$$

where $\Phi(\ )$ is the cdf of the standard normal and $m = \beta_0 + \sum_{j=1}^{k} \beta_j \prod_{l=1}^{z_j} (x_{d_{jl}} - \theta_{jl})_{q_{jl}}$. Hence (10) is equivalent to (4) once we marginalise over ω as $\Phi(m)$ is just the probit link function.

Now, it appears that not much has been gained in that we have introduced an extra n parameters and we still have the same (marginal) model. However, examining the second line in (10), we observe that it has the form of a linear regression model of ω in the spline base Θ, with a known noise variance $\epsilon_i \sim N(0, 1)$. Furthermore, in the Bayesian linear model with normal priors on $\boldsymbol{\beta}$ we know the exact form of the posterior distribution of $\boldsymbol{\beta}$ given the other parameters. This leads to the following efficient MCMC procedure.

Iterate until enough samples have been generated:

- Sample the variables $\Omega = \{\omega_1, \ldots, \omega_n\}$, conditional on the MARS parameters $\mathcal{M}$.
- Sample the MARS parameters $\mathcal{M}$ conditional on Ω.

The sampling distribution of the latent variables Ω, given the current model, is given in the Appendix. The updates to the spline bases are as before but now the updated values for $\boldsymbol{\beta}$,

conditional on the change to the spline base and the current value of Ω, follows from the Bayes linear model theory (O'Hagan, 1994),

$$p(\boldsymbol{\beta} \mid \Omega, \Theta, v) = N(\hat{\boldsymbol{\beta}}, \hat{V}) \tag{12}$$

where $\hat{\boldsymbol{\beta}}$ is the posterior mean $\hat{\boldsymbol{\beta}} = (\Theta'\Theta + v^{-1}I)^{-1}\Theta'\Omega$, $\hat{V}$ is the posterior variance $\hat{V} = (\Theta'\Theta + v^{-1}I)^{-1}$ and as before Θ is the output of the k spline basis functions plus intercept at the n data points.

With $\boldsymbol{\beta}$ updated using (12) it is straightforward to show that the acceptance probability of the MARS update step, conditional on Ω, is just,

$$Q = \min\left\{1, \frac{v^{k/2}|\hat{V}^*|^{1/2}}{v^{k^*/2}|\hat{V}|^{1/2}} \exp\left(\frac{a}{a^*}\right)\right\}, \tag{13}$$

where, v determines the prior variance, $|\hat{V}|$ is the determinant of the posterior variance covariance matrix in (12), the superscripts $*$ refer to the parameters of the proposed update model and a is an error term,

$$a = \Omega'\Omega - \hat{\boldsymbol{\beta}}'\hat{V}^{-1}\hat{\boldsymbol{\beta}}. \tag{14}$$

The acceptance ratio in (13) is just the ratio of marginal likelihoods, as if the model was a linear regression in Ω with predictors Θ. The ratio of marginal likelihoods is known as the Bayes factor (Kass & Raftery, 1995), or sometimes the evidence ratio (MacKay, 1992). It is well known that Bayesian methods contain a natural penalty against over complex models and hence we would not expect the model to over fit by selecting many spline bases even though the prior on k contains no dimension penalty. Note that the ratio in (13) makes no reference to the actual value of $\boldsymbol{\beta}$ that was drawn. In effect the spline coefficients $\boldsymbol{\beta}$ have been marginalised out in the acceptance probability. This marginalisation effectively reduces the dimension of the model space, to leave the marginal probability of the spline base Θ given the latent variables Ω.

We reiterate the fact that the models (4) and (10) are equivalent. The introduction of the latent variables Ω is a computational procedure used to improve the MCMC. The information contained in Ω allow for $\boldsymbol{\beta}$ to be updated in an informed manner when we make effectively large changes to the model by removing, adding or altering a spline base.

3. Examples

We tested the method on the three two class data sets that appeared in Husmeier, Penny, and Roberts (1999) and on two larger data sets obtained from the consumer credit research group within Imperial College, London University. In Husmeier, Penny, and Roberts (1999) they present an empirical comparison of Bayesian neural network models that are described in Neal (1996). The credit data sets have previously be analysed in Denison et al. (2001) and Kelly (1998) where they compare a number of machine learning and statistical algorithms. A brief overview of the data sets is given below and also in Table 1. We ran our method on each

Table 1. Training data set characteristics.

Data set	Observations	Predictors (irrelevant)	Real/Synthetic (1/0)
Ripley	250	4 (2)	0
Tremor	178	4 (2)	1
Ionosphere	250	34 (?)	1
Collections	3488	8 (?)	1
UPL	4000	12 (?)	1

data set using a maximum of 2 interactions per spline, $z_{\max} = 2$. The choice of $z_{\max} = 2$ is made on two grounds; first that the resulting model is interpretable, being additive in a series of one and two dimensional curves and surfaces, and secondly to accurately uncover any higher order interactions requires a very a large amount of data. The algorithm was run for 20,000 iterations which takes around 5 minutes in Matlab on our DEC Alpha workstations for the fist three data sets in Table 1 and about 30 minutes for the larger collections data. The first 10,000 samples were discarded as a burn in. The burn in period was chosen by examining plots of k and of the log likelihood, both of which had completely settled down by this point.

3.1. Ripley's synthetic data

This data set is taken from Ripley (1994). The data set is a two class classification problem where each population is an equal mixture of two two-dimensional normal distributions. A training set of 250 points is used and the model is tested on a set of 1000 points. The data can be obtained from `www.stats.ox.ac.uk/pub/PRNN/`. Following Husmeier, Penny, and Roberts (1999) we add 2 irrelevant predictors to the data set by drawing random points uniform on [0, 1].

3.2. Arm tremor

The task in this data set is to try and predict the presence or absence of Parkinson's disease in 357 individuals based on the two measurements of arm tremor. The original data is taken from Spyers-Ashby (1996). The original data is partitioned into a training set of 178 points and a test set of 179 points. Again, as in Husmeier, Penny, and Roberts (1999) we add 2 irrelevant predictors to the data set by drawing random points uniform on [0, 1].

3.3. Ionosphere data

This data was consists of 'good' and 'bad' radar returns from a system of 16 high-frequency antennas set up in Goose Bay, Labrador to monitor free electrons in the ionosphere. Good returns are those signals showing evidence of structure, while bad returns are those that do not. The data is pre-processed to obtain 17 pulse numbers each of which is described by 2 attributes to produce 34 predictors. The training set size was 250 points with 150 points retained for testing.

3.4. *Collections data*

This example is taken from a retail credit problem concerning the repayment of bank loans. Accounts that are deemed to be unsatisfactory are entered into a process called "collections", where a more vigorous effort is made to correct the problem. The task is to try to predict those customers that spend less than 30 days in collections, based on a series of 8 measurements taken from the original application form and from features of the customer's conduct during repayment. A total of 6976 observations are available, which are randomly split into equally sized training and test sets.

3.5. *Unsecured personal loan (UPL) data*

This example is a retail credit problem relating to the forecasting of bad debtors of unsecured personal loans. An individual is classed as having defaulted on their loan if they are more than 3 months late on their repayment plan. The problem is to predict defaulters using 12 predictors taken from the loan application form and from the customers bank record. A total of 8,000 records are available (4,000 of which are defaulters) and these are randomly split into two equally sized training and test sets.

3.6. *Results*

The results in terms of misclassification error are shown in Table 2. As a benchmark we include results using the Real Adaboost algorithm (Schapire & Singer, 1999) with decision stumps run for 200 rounds of boosting. Also included are the best reported model in Husmeier, Penny, and Roberts (1999) for the Bayesian neural networks with Automatic Relevance Determination (ARD) and from Denison et al. (2001) and Kelly (1998). From Table 2 we see that the Bayesian MARS method appears competitive with the other models on these data sets.

The predictive distribution for the arm tremor data set is shown in figure 1. Notice how the model has selected essentially a hard linear decision boundary at the top of the figure. The predictive contours in the bottom of figure 1 appear smooth even though the individual MARS models have axis parallel non-smooth contours. This is an artifact of the Bayesian

Table 2. Test errors on data sets for the Bayesian MARS (BMARS) model.

Data set	BMARS	Real Adaboost	Best reported result
Ripley	0.091	0.131	0.088[a]
Tremor	0.146	0.201	0.163[a]
Ionosphere	0.053	0.080	0.073[a]
Collections	0.255	0.266	0.263[b]
UPL	0.33	0.33	0.36[c]

Also included are results using Real Adaboost with stumps (Schapire et al., 1998) and the best results found in the papers of [a]Husmeier, Penny, and Roberts (1999), [b]Denison et al. (2001) and [c]Kelly (1998).

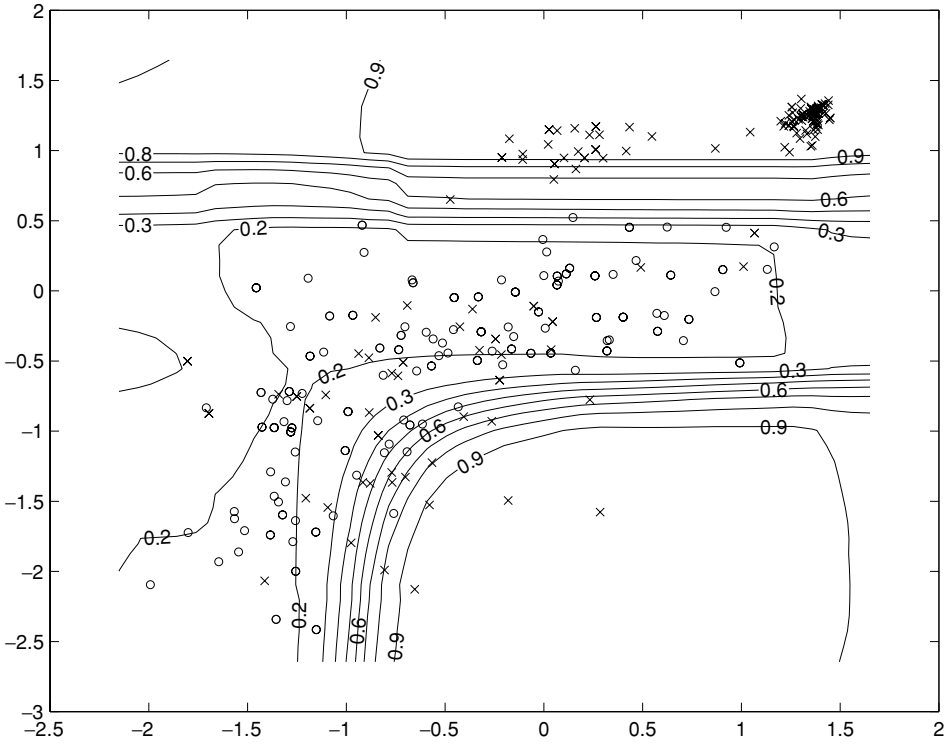

Figure 1. Arm tremor data set with class probability contours showing $p(y_i = 1)$ under the Bayesian MARS model.

approach where the process of marginalisation in (7) has the tendency to smooth the non-smooth models.

In all of the examples the use of the latent variables produces excellent acceptance rates for the reversible jump MCMC update proposals with all three proposals (addition, deletion, alteration) showing around a 30% acceptance rate. In contrast, when we tried the algorithm without latent variables the acceptance rates were much poorer, down to under 2% when using a Metropolis-Hastings approach. The marginal densities for the number of splines used, $p(k \mid Y)$, for the examples are shown in figure 2 as histograms using 10,000 MCMC samples. We can see that the number of splines used by the model adapts to the problem at hand and that a range of values for k have support under the data which suggests that fixing a value for the number of splines is inappropriate.

4. Discussion

We have presented a Bayesian version of MARS for performing two class classification that mixes over the space of models with differing numbers and position of splines. On a number of problems the model appears competitive with other methods reported in the

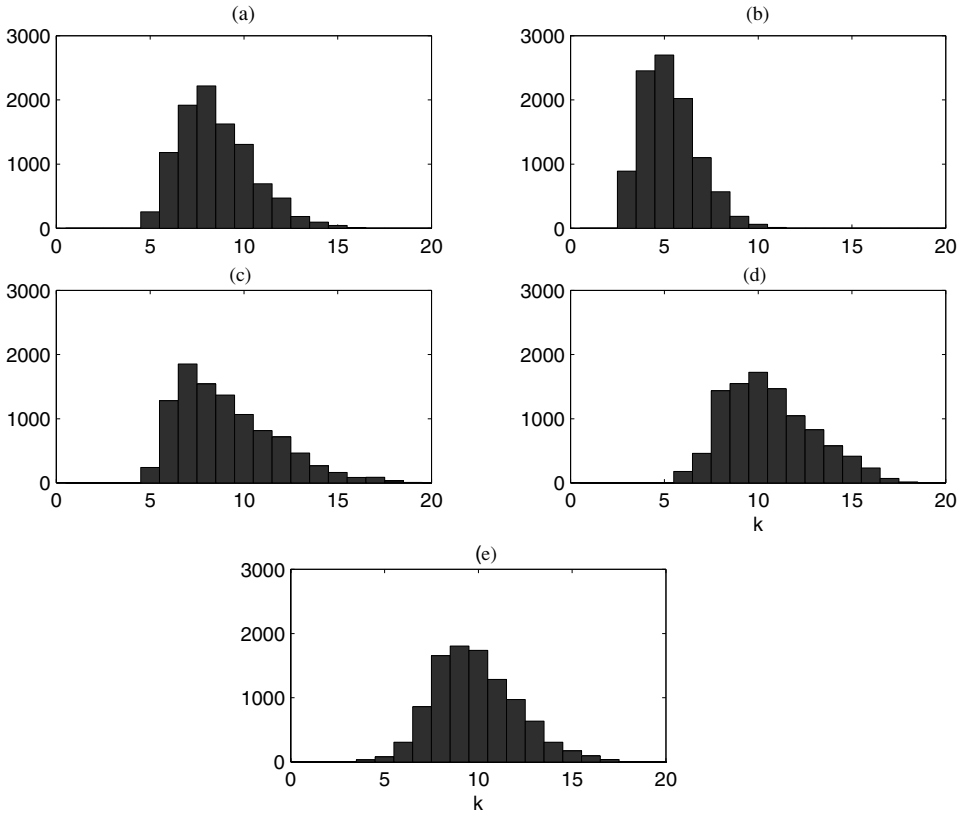

Figure 2. Plots of $p(k \mid Y)$, the posterior distribution of the number of spline basis functions, for the four data examples, using 10,000 samples from the MCMC output. (a) $p(k \mid Y)$ for Ripley's data; (b) $p(k \mid Y)$ for the arm tremor data; (c) $p(k \mid Y)$ for the ionosphere data; (d) $p(k \mid Y)$ for the collections data; (e) $p(k \mid Y)$ for the UPL data.

literature. The use of latent variables allowed us to define efficient MCMC procedures for the computation. Multi-class problems can be tackled in much the same way as we describe here; see the extension to polychotomous classification in Albert and Chib (1993). The Bayesian MARS method should scale well to larger problems as the major computational burden in the algorithm is inverting the $k \times k$ posterior variance-covariance matrix $\hat{v}$ in (12) which is not dependent on the number of data points or the number of predictors. The procedures we describe are generic and applicable to any basis function method including wavelets, artificial neural networks and radial basis functions.

Appendix

The pseudo-code for the algorithm is as follows:

- Start the model with just an intercept term and no spline basis functions, i.e. $k = 0$ and $\Theta = (1, \ldots, 1)'$.

- Initialise the latent variables $\omega_i = 1$ if $y_i = 1$, $\omega_i = -1$ otherwise, for $i = 1, \ldots, n$.
- Draw β_0 using the procedure for updating β given below.
- Iterate until enough samples have been generated:
  - Redraw the latent variables Ω given the current model.
  - Update the prior variance v using the procedure described below.
  - Update Θ using one of the following moves chosen with equal probability:
    - Add a spline basis function.
    - Delete a spline basis function.
    - Alter a spline basis function.
  - Redraw $\boldsymbol{\beta}$.
  - Accept the changes to Θ and $\boldsymbol{\beta}$ with probability

$$Q = \min\left\{1, \frac{v^{k/2}|\hat{V}^*|^{1/2}}{v^{k^*/2}|\hat{V}|^{1/2}} \exp\left(\frac{a}{a^*}\right)\right\}, \tag{15}$$

    where, v determines the prior variance, $|\hat{V}|$ is the determinant of the posterior variance covariance matrix in (12), the superscripts $*$ refer to the parameters of the proposed update model and a is an error term,

$$a = \Omega'\Omega - \hat{\boldsymbol{\beta}}'\hat{V}^{-1}\hat{\boldsymbol{\beta}}. \tag{16}$$

  - Else keep the current model

The procedures for adding, deleting and altering the spline base and for updating $\boldsymbol{\beta}$ are given below.

Adding a spline

When adding a basis function to the model we perform the following procedure

1. Draw the interaction level of the spline from the prior $z_j \sim U[1, \ldots, z_{\max}]$.
2. Draw the z_j elements $\{d_{j1}, \ldots, d_{jz_j}\}$ from $\{1, \ldots, p\}$ without replacement.
3. For each of the z_j interactions that make up the jth spline: select a data point at random from the data set, say $\boldsymbol{x}_i$ and set the corresponding knot point θ_{jl} to be $\theta_{jl} = x_{id_{jl}}$. Then draw the orientation q_{jl} from uniform $\{+, -\}$.
4. Update the values for $\boldsymbol{\beta}$. See below.

Deleting a spline

Simply choose one of the k splines at random and remove it from the model. Then update the values of $\boldsymbol{\beta}$. See below.

Altering a spline

When altering a spline

1. Select one, say the jth, spline uniformly at random.
2. Select one, say the lth, of the z_j interactions at random and reset the knot point θ_{jl} by randomly drawing a data point $\boldsymbol{x}_i$ from the data set and fixing the value of θ_{jl} to be $x_{id_{jl}}$.
3. Update the values for $\boldsymbol{\beta}$. See below.

Updating the latent variables Ω

The latent variables ω conditional on the current model parameters $\mathcal{M}$ and the data Y, can be seen to follow a truncated normal distribution (Albert & Chib, 1993),

$$\omega_i \sim \begin{Bmatrix} N(m_i, 1)I(\omega_i > 0) & \text{if } y_i = 1 \\ N(m_i, 1)I(\omega_i < 0) & \text{if } y_i = 0 \end{Bmatrix}, \tag{17}$$

where I is the identity function and $m_i = \beta_0 + \sum_{j=1}^{k} \beta_j \prod_{l=1}^{z_j} (x_{id_{jl}} - \theta_{jl})_{q_{jl}}$. Efficient algorithms for generating from truncated normal distributions exist, see for example Robert (1995).

Updating $\boldsymbol{\beta}$ *given changes to the spline base and the latent variables* Ω

Given the set of latent variables Ω and the model (10), we can make use of Bayesian linear model theory to update the spline coefficients using their posterior distribution, following changes to the spline base. New values for $\boldsymbol{\beta}^*$ are drawn from a multivariate normal distribution,

$$\boldsymbol{\beta}^* \sim N(m^*, V^*) \tag{18}$$

where,

$$\begin{aligned} m^* &= V^*(\Theta^*)'\Omega, \\ V^* &= [(\Theta^*)'\Theta^* + v^{-1}I]^{-1} \end{aligned} \tag{19}$$

where Θ^* is the $n \times (k^* + 1)$ matrix of outputs from the k^* splines plus intercept (3) and v determines the variance of the prior $p(\boldsymbol{\beta}) = N(0, vI)$.

Updating the prior variance v *given the model*

Using the conjugate (Gamma) hyper-prior for v^{-1} with parameters, $p(v^{-1}) = Gamma(0.001, 0.1)$ we can draw new values for v using the conditional posterior

distribution which is,

$$v^{-1} \sim Gamma\left(0.001 + \frac{k}{2}, 0.1 + \frac{\beta'\beta}{2}\right)$$

where k is the number of basis functions and β are the regression coefficients.

References

Albert, J. H., & Chib, S. (1993). Bayesian analysis of binary and polychotomous response data. *Journal of the American Statistical Association, 88*, 669–679.

Andrieu, C., de Freitas, J. F. G., & Doucet, A. (2000). Robust full Bayesian methods for neural networks. In S. A. Solla, T. K. Leen, & K. Muller (Eds.), *Advances in neural information processing systems (NIPS 12)* (Vol. 12, pp. 379–385). MIT Press.

Denison, D. G. T., Adams, N. A., Holmes, C. C., & Hand, D. J. (2001). Bayesian partition modelling. *Computational Statistics and Data Analysis*, to appear.

Denison, D. G. T., Mallick, B. K., & Smith, A. F. M. (1998). Bayesian MARS. *Statistics and Computing, 8*, 337–346.

Draper, D. (1995). Assessment and propogation of model uncertainty (with discussion). *Journal of the Royal Statistical Society series B, 57*, 45–97.

Friedman, J. H. (1991). Multivariate adaptive regression splines (with discussion). *Annals of Statistics, 19*, 1–141.

Gilks, W. R., Richardson, S., & Spiegelhalter, D. J. (1996). *Markov chain Monte Carlo in practice*. London: Chapman and Hall.

Green, P. J. (1995). Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika, 82*, 711–732.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika, 57*, 97–109.

Holmes, C. C., & Mallick, B. K. (1998). Bayesian radial basis functions of variable dimension. *Neural Computation, 10*, 1217–1233.

Holmes, C. C., & Mallick, B. K. (2000). Bayesian wavelet networks for nonparametric regression. *IEEE Transactions on Neural Networks, 11*, 27–35.

Husmeier, D., Penny, W. D., & Roberts, S. J. (1999). An empirical evaluation of Bayesian sampling with hybrid Monte Carlo for training neural network classifiers. *Neural Networks, 12*, 677–705.

Kass, R. E., & Raftery, A. E. (1995). Bayes factors. *Journal of the American Statistical Association, 90*, 773–795.

Kelly, M. (1998). Tackling change and uncertainty in credit scoring. PhD Thesis, The Open University.

Kooperberg, C., Bose, S., & Stone, C. J. (1997). Polychotomous regression. *Journal of the American Statistical Association, 93*, 117–127.

MacKay, D. J. C. (1992). Bayesian interpolation. *Neural Computation, 4*, 415–447.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E. (1953). Equations of state calculations by fast computing machines. *Journal of Chemical Physics, 21*, 1087–1091.

Neal, R. M. (1996). *Bayesian learning for neural networks*. New York: Springer-Verlag.

O'Hagan, A. (1994). *Kendall's advanced theory of statistics: Bayesian inference* (Vol. 2b). Cambridge: Arnold.

Rasmussen, C. E. (1996). Evaluation of Gaussian processes and other methods for non-linear regression. PhD Thesis, University of Toronto.

Robert, C. P. (1995). Simulation of truncated normal variables. *Statistics and Computing, 5*, 121–125.

Schapire, R. E., Freund, Y., Bartlett, P., & Lee, W. S. (1998). Boosting the margin: A new explanation for the effectiveness of voting methods. *Annals of Statistics, 26*, 1651–1686.

Schapire, R. E., & Singer, Y. (1999). Improved boosting algorithms using confidence-rated predictions. *Machine Learning, 37*, 297–336.

Smith, M., & Kohn, R. (1996). Nonparametric regression using Bayesian variable selection. *Journal of Econometrics, 75*, 317–344.

Spyers-Ashby, J. M. (1996). The recording and analysis of tremor in neurological disorders. PhD Thesis, Imperial College, London University.

Wood, S., & Kohn, R. (1998). A Bayesian approach to robust binary nonparametric regression. *Journal of the American Statistical Association, 93*, 203–213.

Received July 19, 2000
Revised July 9, 2001
Accepted September 7, 2001
Final manuscript October 1, 2001