



ELSEVIER

Computational Statistics & Data Analysis 38 (2002) 475–485

COMPUTATIONAL
STATISTICS
& DATA ANALYSIS

www.elsevier.com/locate/csda

Bayesian partition modelling

D.G.T. Denison*, N.M. Adams, C.C. Holmes, D.J. Hand

Imperial College of Science, Technology and Medicine, London SW7 2BU, UK

Abstract

This paper reviews recent ideas in Bayesian classification modelling via partitioning. These methods provide predictive estimates for class assignments using averages of a sample of models generated from the posterior distribution of the model parameters. We discuss modifications to the basic approach more suitable for problems when there are many predictor variables and/or a large training sample. © 2002 Elsevier Science B.V. All rights reserved.

Keywords: Bayesian model averaging; Classification; Markov chain Monte Carlo; Partitioning

1. Introduction

The general classification problem involves predicting the class conditional probabilities at any location $\mathbf{x}$ within some domain of interest $\mathcal{X}$. This is nearly always achieved by “training” a model with a set of historic data pairings, $\mathcal{D} = \{y_i, \mathbf{x}_i\}_1^n$. Adopting a Bayesian approach involves determining the posterior predictive distribution, $p(y=k|\mathbf{x}, \mathcal{D})$, for $\mathbf{x} \in \mathcal{X}$, where $k \in \{1, \dots, K\}$ and denotes the class label. A model, indexed by parameters $\boldsymbol{\theta}$, is often introduced to capture our beliefs about the underlying probability surface. This allows us to write the (predictive) distribution of interest as

$$\begin{aligned} p(y|\mathbf{x}, \mathcal{D}) &= \int_{\Theta} p(y|\mathbf{x}, \mathcal{D}, \boldsymbol{\theta}) p(\boldsymbol{\theta}|\mathcal{D}) d\boldsymbol{\theta} \\ &\propto \int_{\Theta} p(y|\mathbf{x}, \boldsymbol{\theta}) p(\mathcal{D}|\boldsymbol{\theta}) p(\boldsymbol{\theta}) d\boldsymbol{\theta}, \end{aligned} \tag{1}$$

where Θ is the space defined by all the allowable models.

* Corresponding author.

In essence, predictive Bayesian modelling is straightforward. Once a set of models to perform the classification is identified, all that is required is a prior for θ and the definition of a likelihood $p(\mathcal{D}|\theta)$. Further, the likelihood is usually defined naturally from the context of the problem so the model prior is very important. Hence, the definition of the model space, Θ , is critical to the performance of the data analysis. Even though (1) compensates to some degree for model misspecification, through integration out of the model parameters, if the model space has no support over models close to the truth poor out-of-sample predictive results will ensue. Thus, taking Θ to cover a wide variety of possible models is desirable but this often means that the integral in (1) cannot be performed analytically. Approximations to the integral are readily available by simulation from $p(\theta|\mathcal{D})$, most commonly with Markov chain Monte Carlo methods (e.g. Gelfand and Smith, 1990). However, depending on the form of Θ , the model space can be difficult to sample from, often due to correlation in the parameters of θ . As efficient estimation of $p(y|x, \mathcal{D})$ relies on an approximation to the posterior $p(\theta|\mathcal{D})$, forms for θ which allow efficient sampling should be preferred. One such type of model is the Bayesian partition model of Holmes et al. (1999).

In this paper, we think of partition models as those which explain the data by fitting local distributions to local areas of the predictor space $\mathcal{X}$. Further, we make the extra restriction that the parameters of the local distributions are mutually independent. Partition models of this type are well established in the literature dating back at least to the work on recursive partitioning by Morgan and Sonquist (1963), which was further developed by Friedman (1979), leading to the well-known book on classification and regression trees by Breiman et al. (1984). Classification trees involve determining partitions of the data which lead to groupings of the datapoints into terminal nodes which, ideally, are dominated by one class. Classification trees are highly interpretable models but suffer from relatively poor out-of-sample predictive performance. This is because the partitions of the predictor space are forced to be made on a single predictor, so are necessarily axis-parallel, and the partitioning proceeds greedily, so successive “best” splits to the data are made at each step. The machine learning community, in particular, has continued to research generalisations of (recursive) partitioning models, most notably improvements in the space of possible splits (e.g. Murthy et al., 1994).

The Bayesian partition model (BPM) is motivated by the same basic principles of earlier models of the same type, i.e. points close by in predictor space come from the same local distribution. However, a more flexible partitioning scheme is proposed. Further, Bayesian model averaging (Raftery et al., 1996) ideas allow for prediction using averages (or ensembles) of models, rather than single ones which have a somewhat unappealing discontinuous nature. As well as describing the BPM of Holmes et al. (1999) in detail, we comment on both its strengths and weaknesses. In particular, we introduce a new model which is based on the partition model but is more suited to data-mining applications when there are a large number of training data points and/or a large number of predictor variables (or features). This new model, known as the product partition model (PPM), utilises a modified form of standard naive Bayes formula (e.g. Domingos and Pazzini, 1997) which

assumes

$$p(y|\mathbf{x}) \propto p(\mathbf{x}|y) \times p(y) = \prod_{i=1}^P p(x_i|y) \times p(y), \quad (2)$$

where $\mathbf{x} = (x_1, \dots, x_P)$ is a general feature vector assuming P predictors. The main advantage of this model is the speed with which the MCMC algorithm can be run, allowing fast computation of the posterior. This greater speed comes at the cost of slightly poorer predictive power but this degradation appears acceptable for the large datasets that are typically encountered in modern data-mining. Further, the method allows the user to identify important variables and gives an idea as to how the class assignment is affected by each predictor.

The paper is arranged as follows. In the next section, we review the BPM of Holmes et al. (1999) and Section 3 then introduces the PPM. Section 4 contains a comparison of the two approaches in both classification error and computational time and lastly Section 5 contains a brief discussion.

2. Bayesian partition models

Bayesian partition models are motivated by the basic premise that points nearby in predictor space come from the same local distribution. This idea is well known in many statistical models including local polynomial regression (e.g. Fan and Gijbels, 1996) and classification trees (Breiman et al., 1984). Using this idea, we partition the design space $\mathcal{X}$ into a number of disjoint regions, where within each region the data are assumed exchangeable. We choose to construct these regions through the Voronoi tessellation (Green and Sibson, 1978). Given a set of M points $\mathbf{T} = (\mathbf{t}_1, \dots, \mathbf{t}_M)$ with each $\mathbf{t}_i \in \mathcal{X}$, the Voronoi tessellation defines M non-overlapping regions $R_1, \dots, R_M$ such that the points within R_i are closer to $\mathbf{t}_i$ than any of the other $\mathbf{t}_j (j \neq i)$, i.e. $R_i = \{\mathbf{x} \in \mathcal{X} : \|\mathbf{x} - \mathbf{t}_i\| < \|\mathbf{x} - \mathbf{t}_j\| \text{ for all } j \neq i\}$. We shall take $\|(x_1, \dots, x_P)\|^2 = \sum_1^P w_i^2 x_i^2$, where $\sum_i w_i^2 = 1$. In particular note that when we take all the w_i equal to $1/\sqrt{P}$ then we have the usual Euclidean distance metric (see Fig. 1 for an example of this). Also, as $\mathbf{t}_i$ defines the middle of the region R_i , according to the distance metric used, we shall refer to the $\mathbf{t}_i$ as “centres” and write $\mathbf{T} = (\mathbf{t}_1, \dots, \mathbf{t}_M)$.

Taking a general region, R_i , we only need to count the number of observations in each class lying in this region, given by $n_{i1}, \dots, n_{iK}$, to determine the likelihood of the data in that partition. Further, the complete likelihood just involves multiplying together the individual multinomial likelihoods in each region, $R_i (i = 1, \dots, M)$, so that

$$\ell(\boldsymbol{\phi}, \mathbf{T}, \mathbf{w}, M | \mathcal{D}) = \prod_{i=1}^M \left\{ \frac{n_i!}{\prod_{k=1}^K n_{ik}!} \prod_{k=1}^K \phi_{ik}^{n_{ik}} \right\}, \quad (3)$$

where $\boldsymbol{\phi} = (\phi_1, \dots, \phi_M)$ with the $\phi_i = (\phi_{i1}, \dots, \phi_{iK})$ giving the class probability assignments in each region, and $n_i = \sum_k n_{ik}$. Note that $\sum_k \phi_{ik} = 1$ so this K -vector

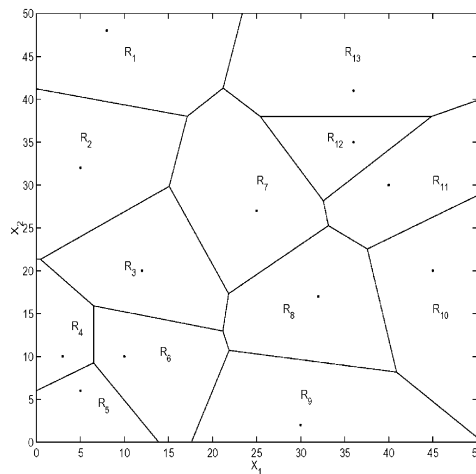


Fig. 1. An example of a BPM with two predictor variables. The lines give the boundaries of the disjoint regions which cover the space of the predictor variables and the dots give the “centres” of the regions.

only contains $K - 1$ random parameters with the obvious restrictions (i.e. they lie between zero and one and their sum is less than, or equal to, one).

2.1. Prior specification

The model described above contains $(K + P)M + 1$ (scalar) random parameters so using analytic integration to obtain the posterior distribution of interest will be impossible except in the very simplest of problems with severe restrictions on the possible values of M . Hence, we must resort to simulation techniques to evaluate the posterior distribution of interest. However, as noted in the introduction, the model is only introduced as a way of writing down the conditional dependence between y and x and the model parameters will, eventually, be integrated out. For this reason it would appear appropriate to specify priors, where possible, that allow us to simplify the posterior by analytic integration of some of the random variables.

For classification problems of this type we know that the prior distribution on the class assignment probabilities that allows analytic integration of these parameters is the Dirichlet distribution. In the Bayesian literature this is known as a conjugate prior. Note that the probability density function of the d -dimensional Dirichlet distribution, $\text{Di}_d(x_1, \dots, x_d | \alpha_1, \dots, \alpha_{d+1})$, is given by

$$f(x_1, \dots, x_d) = \frac{\Gamma(\sum_{i=1}^{d+1} \alpha_i)}{\prod_{i=1}^{d+1} \Gamma(\alpha_i)} \left\{ \prod_{i=1}^d x_i^{\alpha_i - 1} \right\} \left(1 - \sum_{i=1}^d x_i \right)^{\alpha_{d+1} - 1},$$

for $0 \leq x_1, \dots, x_d \leq 1$, $\sum_{i=1}^d x_i \leq 1$ and where $\Gamma(\cdot)$ denotes, the usual Gamma function. In our model, we choose to use the uniform prior specification on class assignments and assume independence between the ϕ_i in different regions, i.e., we take all

the α_i 's, equal to one and the prior on ϕ is given by

$$p(\phi|M) = \prod_{i=1}^M \text{Di}_{K-1}(\phi_{i1}, \dots, \phi_{i,K-1} | \mathbf{1}),$$

where $\mathbf{1}$ represents a K -dimensional vector of ones. The next subsection gives more details on how the use of this prior aids us in the calculation of the posterior distribution, $p(y|\mathbf{x}, \mathcal{D})$.

The remaining prior distributions, $p(\mathbf{T}|M)$ and $p(M)$, are more straight-forward to define. The general strategy we use is to assign uniform priors that have little effect on the resulting posterior as, in most tasks, little is known about these parameters *a priori*.

The prior for M is take uniform on $\{1, \dots, n\}$, where n is the total number of datapoints. The prior for each element t_{ij} of a given centre, $\mathbf{t}_i$, is assumed to be drawn uniformly from the marginal predictor values present for the j th predictor variable. The final prior is on $\mathbf{w}$ which we take to be drawn uniformly from the P -dimensional unit hypercube.

2.2. Prior to posterior analysis

The use of a conjugate prior for ϕ allow analytic integration of these parameters within regions such that

$$\begin{aligned} p(\mathcal{D} | \mathbf{T}, \mathbf{w}, M) &= \int_{\phi} p(\mathcal{D} | \phi, \mathbf{T}, \mathbf{w}, M) p(\phi|M) d\phi \\ &= \{(K-1)!\}^M \prod_{i=1}^M \left\{ \frac{\prod_k n_{ik}!}{(n_i + K - 1)!} \right\}, \end{aligned} \quad (4)$$

using the standard result that when m is a positive integer $\Gamma(m+1) = m!$. Thus, we can assume that the model is indexed only by the parameter vector $\theta = (\mathbf{T}, \mathbf{w}, M)$, rather than needing to integrate over the extra $(K-1)M$ terms in ϕ as well. This reduces the model space over which numerical methods need to be applied, vastly improving the efficiency of the simulation algorithm we will need to employ.

We find that the predictive density of a general point in region R_i (Denison et al., 2000) is given by

$$p(y=k | \mathbf{x} \in R_i, \mathbf{T}, \mathbf{w}, M) = \frac{n_{ik} + 1}{n_i + K}, \quad (5)$$

for $k=1, \dots, K$. This approaches the maximum likelihood estimate for the class assignment probabilities in R_i as n_{ik} and n_i become large, but for smaller values the class probabilities are shrunk towards the prior mean value of $1/K$.

Although the technical details employed to approximate $p(\theta|\mathcal{D})$ are outside the scope of this paper the general idea has two basic steps:

1. Generate a sample of N models from the posterior distribution of the model, $p(\boldsymbol{\theta}|\mathcal{D})$.
2. Approximate $p(y|\mathbf{x}, \mathcal{D})$ assuming that the generated sample of models is in fact the true posterior distribution. That is, assume that the posterior distribution is discrete with weight $1/N$ on each of the N models so that

$$p(y|\mathbf{x}, \mathcal{D}) = \frac{1}{N} \sum_{i=1}^N p(y|\mathbf{x}, \mathcal{D}, \boldsymbol{\theta}^{(i)}),$$

where $\boldsymbol{\theta}^{(i)}$ is the i th model in the generated sample.

Using this approach there are many questions that need to be addressed, most notably how to efficiently sample the models, how many to collect and how do we know we are sampling from the correct distribution. for more information on these, so-called Markov chain Monte Carlo, techniques in general see Gilks et al. (1996) and, in particular, for the methodology involved in running these algorithms for partition models see Holmes et al. (1999).

3. Product partition models

Bayesian partition models are particularly effective when the number of predictors is small but can suffer both from (relatively) poor predictive results as well as being computationally expensive algorithm when either the number of predictors, or the number of training data examples, becomes large. A number of factors are likely to account for this including:

- (a) for every extra predictor variable the dimension of $\boldsymbol{\theta}$ increases by M leading to more computational effort being required to simulate the random sample of models as well as worse approximations to $p(y|\mathbf{x}, \mathcal{D})$;
- (b) the Voronoi tessellation relies on a distance metric which becomes expensive to compute in high dimensions;
- (c) when P is large, points become more spread out in predictor space (the curse of dimensionality) so the disjoint regions become less “local” and hence they cannot capture small-scale features so well.

However, it is worth remembering that in smaller-scale studies the BPM is still very effective as shown by the results in Holmes et al. (1999).

Data-mining applications invariably involve a large number of training examples and predictor variables so fitting a BPM becomes a less attractive option. Instead, we now outline the Bayesian PPM, inspired by the model described in Hand and Adams (2000) for consumer credit problems. In this area, banks often need to make classifications that can be explained. This has led them to favour simple classification rules, such as logistic regression and naive Bayes (Hand and Henley, 1997). These methods have the advantage of yielding a model that is additive in the variables, and so a decision is made simply by summing scores. The work of

Denison et al. (2000) showed how to modify naive Bayes to find the optimal positions for a fixed number of partitions on continuous variables.

The PPM uses a refinement of (2) to build up a model which is a product of simple descriptions of the influence each predictor has on class membership. In particular, we choose to partition each predictor in a number of regions where within each the data are assumed exchangeable, and the parameters across regions are taken to be independent. This has obvious similarities to the BPM but now the partitioning takes place on individual predictors, not jointly as in the BPM. Further, the simple partitioning scheme means that the computational effort required to determine the regions that a datum resides in is greatly reduced for large P or with large datasets.

The definition of this new model is very similar to the BPM, however, we do not now need the weighting vector $\mathbf{w}$. We think of the partition of the v th predictor variable as being defined by the vector $\mathbf{T}_v$, which gives the M_v split locations. Hence, in this case, the model we must simulate from is defined by $\boldsymbol{\theta} = (\mathbf{T}_v, M_v)_{v=1}^P$. We could fix the number of split locations for each predictor but we prefer to keep it unknown so the model can essentially exclude unimportant predictors by setting $M_v = 0$ for these variables.

For this model, we use a likelihood motivated by (2) so that the log-likelihood is given by

$$L(\boldsymbol{\phi}, \boldsymbol{\theta} | \mathcal{D}) = \sum_{v=1}^P \frac{M_v + 1}{M} \sum_{i=1}^{M_v+1} \sum_{k=1}^K n_{vik} \log \phi_{vik} + \text{constant}, \quad (6)$$

where n_{vik} is the number of class k in the i partition of the v th predictor variable and $M = \sum_v (M_v + 1)$ is the total number of partitions in the model. If we had used (2) exactly to give us the likelihood the $(M_v + 1)/M$ factor would not have been present in (6). However, this leads to the likelihood making overconfident predictions as the naive Bayes formula leads to the effective size of the dataset being increased P -fold. This is because each single datum in the standard Bayes formula, say $(y_i, \mathbf{x}_i)$ appears in the naive Bayes formula P times as $(y_i, x_{i1}), \dots, (y_i, x_{iP})$. A simple way to avoid this problem is to weight each dimension in the log-likelihood by $1/P$ so each variable is assumed equally important and the effective size of the dataset is unchanged. However, we wish to upweight the predictions made by those variables for which there are many splits as these variables are likely to aid classification rather than the ones with few, or no splits at all. This is why the $(M_v + 1)/M$ fraction is included.

Using this likelihood, we can still use a conjugate Dirichlet prior on $\boldsymbol{\phi}$ and we take this prior to be

$$p(\boldsymbol{\phi} | M_1, \dots, M_P) = \prod_{v=1}^P \prod_{i=1}^{M_v} \text{Di}_{K-1}(\phi_{vi1}, \dots, \phi_{vi,K-1} | \boldsymbol{\alpha}_v), \quad (7)$$

where $\boldsymbol{\alpha}_v$ is a K -dimensional vector of α_v , where we take $\alpha_v = (M_v + 1)/M$. As before, we can determine the predictive density for each class as

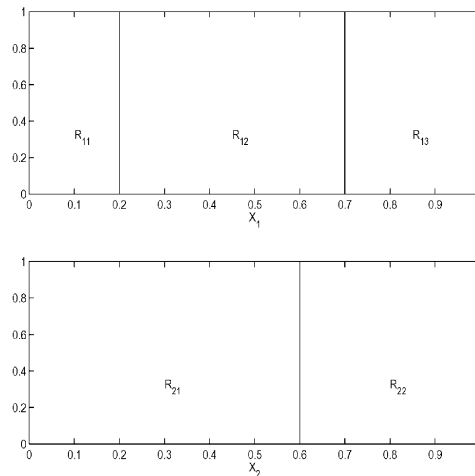


Fig. 2. An example of a PPM with two predictor variables. Each variable is split into a random number of disjoint regions with random split points.

$$p(y=k|\mathbf{x}, \boldsymbol{\theta}, \mathcal{D}) = \frac{1}{Z} \prod_{v=1}^P \frac{\Gamma((n_{vi_v k} + 2)/\alpha_v)}{\Gamma((n_{vi_v k} + 1)/\alpha_v)} \frac{\Gamma((n_{vi_v \cdot} + K)/\alpha_v)}{\Gamma((n_{vi_v \cdot} + K + 1)/\alpha_v)}, \quad (8)$$

for $k = 1, \dots, K$, where $n_{vi \cdot} = \sum_{k=1}^K n_{vik}$ and the i_v give the index of the partition that the v th component of $\mathbf{x}$ is located in. Note that Z is a normalising constant ensuring that (8) is a distribution.

It is easier to visualise the distribution defined by (8) when the n_{vik} are large (as in data-mining examples). Then we find that, by using Stirling's formula, this distribution can be approximated by

$$p(y=k|\mathbf{x}, \boldsymbol{\theta}, \mathcal{D}) \approx \frac{1}{\tilde{Z}} \prod_{v=1}^P \left(\frac{n_{vi_v k} + 1}{n_{vi_v \cdot} + K} \right)^{\alpha_v}.$$

for $k = 1, \dots, K$ and where $\tilde{Z}$ is the normalising constant for this distribution. This looks far more like the corresponding density for the BPM given in (5). In effect the predictions are weighted geometric means of the proportions of the classes in each of the partitions a given datum is located in. Variables which are important (i.e. have a higher α_v) are upweighted and those which are less important have less influence on the overall prediction. It is this consideration which led us to the particular form of the prior we chose for this model.

In Figs. 1 and 2, we compare the BPM and PPM. The main difference is that the BPM defines regions in as many dimensions as there are predictor variables, unlike the PPM which ensures all the regions are one-dimensional. Also, the BPM uses only the single region the test point is located in to determine the class assignment probabilities whereas the PPM uses the “bin” the test point is located in for each predictor variable.

Now, we have outlined the PPM model, we proceed in an analogous way to make predictions using a sample of generated models, and in the next section, we compare the methodologies introduced.

4. Collections data example

This example is concerned with retail credit, in particular, the conduct of a bank loan which is monitored throughout its lifetime. Accounts that are deemed to be conducted badly enter a process called collections, where more vigorous effort is made to correct the problem. The organisation that provided this data set are especially interested in being able to identify customers who spend less than 30 days in collections. Thus, we have a two class classification problem. The bank has a vast number of variables available to build such a classification rule, including application variables (data from the original application form) and behavioural variables (determined by the customer's conduct during repayment). Bank experts selected a subset of eight variables for the purpose of constructing a collections classifier. The data consists of a total of 6976 observations, which were randomly split into equal size training and test sets. Note that the observed proportion of class 1 (longer than 30 days in collections) accounts was 32%.

We ran both Bayesian methods (BPM and PPM) for 100,000 iterations with the first 50,000 discarded as burn-in iterations. Every tenth model visited after the burn-in period was taken to be in the generated sample so we used ($N=$)5000 models to approximate the posterior distribution. Also, we constrain the overall complexity of the model by allowing no more than 10 splits on any one variable.

Table 1 reports the results of the two Bayesian models, as well as logistic regression, boosting, and a tree-based approach. Performance is compared using error rate. The classification tree was constructed using the `tree` function in S-plus (Clark and Pregibon, 1992), and pruned on the basis of a cross-validated estimate of deviance. The logistic regression model was fitted using the S-Plus function `glm`. Finally, we quote the performance of the boosting algorithm (e.g. Freund, 1995) using the AdaboostM1 implementation in the WEKA software package (Witten and Frank, 1999) based on 1000 boosting trials with decision stumps. Note that both Bayesian methods do acceptably well on this dataset with respect to misclassification rate when compared to these other methods (see Table 1).

Table 1
The misclassification rates and for a variety of methods
on the Collections data

Method	% Misclassified
PPM	26.3
BPM	27.2
Tree	29.1
Logistic	28.6
Boosting	26.6

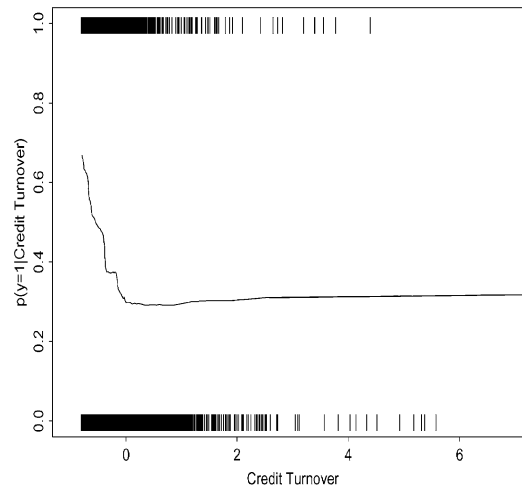


Fig. 3. The effect the credit turnover variable has on the probability of being in class 1. The ticks give the location of datapoints of each class and the other variables were set to their median values.

Comparing the results obtained using BPM and PPM, we find that the PPM had a lower error rate and it only took 2 min to run the algorithm, rather than the 80 min for the BPM (both on a 250 MHz DEC Alpha). However, we found evidence that the simulation algorithm for the BPM had not reached its stationary distribution when we started collecting the generated sample. This led to the relatively poor results found, as these improved to the same levels as PPM when more iterations were performed.

The PPM model favours many splits on the fifth predictor (credit turnover) with relatively few splits on the others, yet this very simple model performs well. In Fig. 3 we explicitly show how credit turnover affects the probability of being in class 1. This also shows how the effect does not look too jagged due to the posterior smoothing that takes place by making predictions using 5000 models, rather than just a single PPM.

5. Discussion

In this paper, we have outlined two methods for performing classification using models that partition the predictor space into disjoint regions. These models try to capture local features in the data to aid prediction. The BPM model has been shown to be unsuitable when there are many predictors and/or a large number of datapoints so a new model has been introduced which alleviates some of these problems. This PPM has been shown to be competitive with other standard classification procedures on the dataset analysed but more investigation needs to be undertaken to see for which type of datasets it is most appropriate. By its very nature it will not be particularly good at classifying data when there are strong interactions between variables but it is well-suited to situations where there are many irrelevant predictors.

Acknowledgements

The work of the first author was supported by a grant from the Nuffield Foundation.

References

- Breiman, L., Friedman, J.H., Olshen, R., Stone, C.J., 1984. *Classification and Regression Trees*. Wadsworth, Blemont, CA.
- Clark, L.A., Pregibon, D., 1992. Tree based models, *Statistical Models in S*. Wadsworth, Pacific Grove.
- Denison, D.G.T., Adams, N.A., Hand, D.J., Holmes, C.C., 2000. Product partitioned models. Technical Report, Department of Mathematics, Imperial College.
- Domingos, P., Pazzini, M., 1997. On the optimality of the simple bayesian classifier under zero-one loss. *Mach. Learning* 29, 103–130.
- Fan, J., Gijbels, I., 1996. *Local Polynomial Modelling and its Applications*. Chapman & Hall, London.
- Freund, Y., 1995. Boosting a weak learning algorithm by majority. *Inform. Comput.* 121, 256–285.
- Friedman, J.H., 1979. A tree-structured approach to nonparametric multiple regression. In: Gasser, T.H., Rosenblatt, M. (Eds.), *Smoothing Techniques for Curve Estimation*. Springer, Berlin, New York.
- Gelfand, A.E., Smith, A.F.M., 1990. Sampling-based approaches to calculating marginal densities. *J. Amer. Statist. Assoc.* 85, 398–409.
- Gilks, W.R., Richardson, S., Spiegelhalter, D.J., 1996. *Markov chain Monte Carlo in practice*. Chapman & Hall, London.
- Green, P.J., Sibson, R., 1978. Computing Dirichlet tessellations in the plane. *Comput. J.* 21, 168–173.
- Hand, D.J., Adams, N.M., 2000. Choosing attributes for scorecard construction. *J. Appl. Statist.* 27, 527–540.
- Hand, D.J., Henley, W.E., 1997. Statistical classification methods in consumer credit scoring: a review. *J. Roy. Statist. Soc. A* 160, 523–541.
- Holmes, C.C., Denison, D.G.T., Mallick, B.K., 1999. Bayesian partitioning for classification and regression. Technical Report, Imperial College, London.
- Morgan, J.N., Sonquist, J.A., 1963. Problems in the analysis of survey data and a proposal. *J. Amer. Statist. Assoc.* 58, 415–434.
- Murthy, S.K., Kasif, S., Salzberg, S., 1994. A system for induction of oblique decision trees. *J. Artificial Intell.* 2, 1–32.
- Raftery, A.E., Madigan, D., Volinsky, C.T., 1996. Accounting for model uncertainty in survival analysis improves predictive performance (with discussion). In: Bernardo, J.M., Berger, J.O., Dawid, A.P., Smith, A.F.M. (Eds.), *Bayesian Statistics V*. Clarendon Press, Oxford, pp. 323–349.
- Witten, I.H., Frank, F., 1999. *Data mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, Los Alios, CA.