Classification using distance nearest neighbours

N. FRIEL
School of Mathematical Sciences, University College Dublin, Ireland.
and
A.N. PETTITT
Discipline of Mathematical Sciences, Queensland University of Technology, Australia.

Abstract

This paper proposes a new probabilistic classification algorithm using a Markov random field approach. The joint distribution of class labels is explicitly modelled using the distances between feature vectors. Intuitively, a class label should depend more on class labels which are closer in the feature space, than those which are further away. Our approach builds on previous work by Holmes and Adams (2002, 2003) and Cucala *et al* (2009). Our work shares many of the advantages of these approaches in providing a probabilistic basis for the statistical inference. In comparison to previous work, we present a more efficient computational algorithm to overcome the intractability of the Markov random field model. The results of our algorithm are encouraging in comparison to the $k-$nearest neighbour algorithm.

1 Introduction

This paper is concerned with the problem of supervised classification, a topic of interest in both statistics and machine learning. Hastie *et al* (2001) gives a description of various classification methods. We outline our problem as follows. We have a collection of training data $\{(x_i, y_i), i = 1, \ldots, n\}$. The values in the collection $\boldsymbol{x} = \{x_1, \ldots, x_n\}$ are often called features and can be conveniently thought of as covariates. We denote the class labels as $\boldsymbol{y} = \{y_1, \ldots, y_n\}$, where each y_i takes one of the values $1, 2, \ldots, G$. Given a collection of incomplete/unlabelled test data $\{(x_i, y_i), i = n+1, \ldots, n+m\}$, the problem amounts to predicting the class labels for $\boldsymbol{y}^* = \{y_{n+1}, \ldots, y_{n+m}\}$ with corresponding feature vectors $\boldsymbol{x}^* = \{x_{n+1}, \ldots, x_{n+m}\}$.

Perhaps the most common approach to classification is the well-known $k-$nearest neighbours ($k-$nn) algorithm. This algorithm amounts to classifying an unlabelled y_{n+i} as the most common class among the k nearest neighbours of x_{n+i} in the training set $\{(x_i, y_i), i = 1, \ldots, n\}$. While this algorithm is easy to implement, and often gives good performance, it can be criticised since it does not allow any uncertainty to be associated to the test class labels, and to the value to k. Indeed the choice of k is crucial to the performance of the algorithm. The value of k is often chosen on the basis of leave-one-out cross-validation.

There has been some interest in extending the $k-$nearest neighbours algorithm to allow for uncertainty in the test class labelling, most notably by (Holmes and Adams 2002), (Holmes and Adams 2003) and more recently (Cucala *et al* 2009). Each of these probabilistic variants of the $k-$nearest neighbour algorithm, is based on defining a neighbourhood of each point x_i, consisting of the k nearest neighbours of x_i. But

moreover, each of these neighbouring points has equal influence in determining the missing class label for y_i, regardless of distance from x_i. In this article we present a class of models, the *distance nearest neighbour* model, which shares many of the advantages of these probabilistic approaches, but in contrast to these approaches, the relative influence of neighbouring points depends on the distance from x_i. Formally, the distance nearest neighbour model is a discrete-valued Markov random field, and, as is typical with such models, depends on an intractable normalising constant. To overcome this problem we use the exchange algorithm of Murray *et al.* (2006) and illustrate that this provides a computationally efficient algorithm with very good mixing properties. This contrasts with the difficulties encountered by Cucala *et al.* (2009) in their implementation of the sampling scheme of Møller *et al* (2006).

This article is organised as follows. Section 2 presents a recent overview of recent probabilistic approaches to supervised classification. Section 3 introduces the new distance nearest neighbour model and outlines how it compares and contrasts to previous probabilistic nearest neighbour approaches. We provide a computationally efficient framework for carrying out inference for the distance nearest neighbour model in Section 4. The performance of the algorithm is illustrated in Section 5 for a variety of benchmark datasets, as well as challenging high-dimensional datasets. Finally, we present some closing remarks in Section 6.

2 Probabilistic nearest neighbour models

Holmes and Adams (2003) attempted to place the $k-$nn algorithm in a probabilistic setting therefore allowing for uncertainty in the test class labelling. In their approach the full-conditional distribution for a training label is written as

$$\pi(y_i|\boldsymbol{y}_{-i}, \boldsymbol{x}, \beta, k) \propto \exp\left(\beta \sum_{j \sim^k i} I(y_i = y_j)/k\right),$$

where the summation is over the k nearest neighbours of x_i and where $I(y_i = y_j)$ is an indicator function taking the value 1 if $y_i = y_j$ and 0 otherwise. The notation, $j \sim^k i$ means that x_j is one of the k nearest neighbours of x_i. However, as pointed out in (Cucala *et al* 2009), there is a difficulty with this formulation, namely that there will almost never be a joint probability for $\boldsymbol{y}$ corresponding to this collection of full-conditionals. The reason is simply because the $k-$nn neighbourhood system is usually asymmetric. If x_i is one of the k nearest neighbours of x_j, then it does not necessarily follow that x_j is one of the k nearest neighbours of x_i.

Cucala *et al.* (2009) corrected the issue surrounding the asymmetry of the k-nn neighbourhood system. In their probabilistic k-nn (pk-nn) model, the full-conditional for class label y_i appears as

$$\pi(y_i|\boldsymbol{y}_{-i}, \boldsymbol{x}, \beta, k) \propto \exp\left(\beta/k \left\{\sum_{j \sim^k i} I(y_i = y_j) + \sum_{i \sim^k j} I(y_i = y_j)\right\}\right), \qquad (1)$$

and this gives rise to the joint distribution

$$\pi(\boldsymbol{y}|\boldsymbol{x},\beta,k) \propto \exp\left(\beta/k \sum_{i=1}^{n} \sum_{j\sim^k i} I(y_i = y_j)\right).$$

Therefore under this model, following (1), mutual neighbours are given double weight, with respect to non-mutual neighbours and for this reason the model could be seen, perhaps, as an ad-hoc solution to this problem.

It is important to also note that both Holmes and Adams (2002) and Cucala *et al.* (2009) allow the value of k to be a variable. Therefore the neighbourhood size can vary. Holmes and Adams (2002) argue that allowing k to vary has a certain type of smoothing effect.

3 Distance nearest neighbours

Motivated by the work of Holmes and Adams (2002) and Cucala *et al.* (2009) our interest focuses on modelling the distribution of the training data as a Markov random field. Similar to these approaches, we consider a Markov random field based approach, but in contrast our approach explicitly models and depends on the distances between points in the training set. Specifically, we define the full-conditional distribution of the class label y_i as

$$\pi(y_i|\boldsymbol{y}_{-i},\boldsymbol{x},\beta,\sigma) \propto \exp\left(\beta \sum_{j=1,j\neq i}^{n} w_j^i I(y_j = y_i)\right).$$

Positive values of the Markov random field parameter β encourage aggregation of the class label. When $\beta = 0$, the class labels are uncorrelated. In contrast to the pk-nn model, here the neighbourhood set of x_i is constructed to be

$$\boldsymbol{x} \setminus \{x_i\} = \{x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n\}$$

and is therefore of maximal size. We consider three possible models depending on how the collection of weights $\{w_i^j\}$ for $j = 1, \ldots, i-1, i+1, \ldots, n$ are defined.

1. $d-\mathrm{nn}_1$:

$$w_i^j \propto \exp\left\{-\frac{d(x_i,x_j)^2}{2\sigma^2}\right\}, \text{ for } j = 1, \ldots, i-1, i+1, \ldots, n,$$

   where d is a distance measure such as Euclidean.

2. $d-\mathrm{nn}_2$:

$$w_i^j \propto \epsilon + (1-\epsilon) I(d(x_i,x_j) < \sigma), \text{ for } j = 1, \ldots, i-1, i+1, \ldots, n,$$

   again where, I is an indicator function taking value 1, if $d(x_i,x_j) < \sigma$ and 0, otherwise. Further, $\epsilon \in (0,1)$ is defined as a constant, and is set to a value close to 0. (Throughout this paper we assign the value $\epsilon = 10^{-10}$.) A non-zero value of ϵ guarantees that if there are no features within a distance σ of x_i then the class of y_i is modelled using the marginal proportions of the class labels.

3. $d-\text{nn}_3$:

$$w_i^j \propto \exp\left\{-d(x_i, x_j)\sigma\right\}, \text{ for } j = 1, \ldots, i-1, i+1, \ldots, n.$$

Clearly the neighbour system for both models is symmetric, and so the Hammersley-Clifford theorem guarantees that the joint distribution of the class labels is a Markov random field. This joint distribution is written as

$$\pi(\boldsymbol{y}|\boldsymbol{x}, \beta, \sigma) = \frac{q(\boldsymbol{y}|\beta, \sigma, \boldsymbol{x})}{z(\beta, \sigma)} = \frac{\exp\left(\beta \sum_i \sum_{j=1, j\neq i}^{n} w_j^i I(y_j = y_i)\right)}{z(\beta, \sigma)}. \tag{2}$$

As usual, the normalising constant of such a Markov random field is difficult to evaluate in all but trivial cases. It appears as

$$z(\beta, \sigma) = \sum_{y_1} \cdots \sum_{y_n} \exp\left(\beta \sum_i \sum_{j=1, j\neq i}^{n} w_j^i I(y_j = y_i)\right). \tag{3}$$

Some comments:

1. The $k-$nn algorithm and its probabilistic variants always contain neighbourhoods of size k, regardless of how far each of neighbouring points are from the center point, x_i. Moreover, each neighbouring point x_j has equal influence, regardless of distance from x_i. It could therefore be argued that these algorithms are not sensitive to outliers. By contrast the distance nearest neighbour models deal with outlying points in a more robust manner, since if a point x_j lies further away from other neighbours of x_i, then it will have a relatively smaller weight, and consequently less influence in determining the likely class label of y_i.

2. The formulation of distance nearest neighbour models includes every training point in the neighbourhood set, but the value of σ determines the relative influence of points in the neighbourhood set. For the $d-\text{nn}_1$ model, small values of σ imply that only those points with small distance from the centre point will be influential, while for large values of σ, points in the neighbourhood set are more uniformly weighted. Similarly, for the $d-\text{nn}_2$ model, points within a σ radius of the center point are weighted equally, while those outside a σ radius of the center point will have relatively little weight, when ϵ is very close to 0. By contrast, for the $d-\text{nn}_3$ model, large values of the parameter σ imply that points close to the centre point will be influential.

3. For the $d-\text{nn}_2$ model, if there are no features in the training set within a distance σ of x_i, then

$$\pi(y_i = j|\boldsymbol{y}_{-i}, \boldsymbol{x}, \beta, \sigma) \propto \exp\left(\beta p_j^i\right), \text{ for } j = 1, \ldots, G,$$

where p_j^i denotes the proportion of class labels j in the set $\boldsymbol{y} \setminus \{y_i\}$. The parameter β determines the dependence on the class proportions. A large value of β typically predicts the class label to be the class with the largest proportion, whereas a small value of β results in a prediction which is almost uniform over all possible classes. Conversely, if there any feature vectors within a radius σ of x_i, then the class labels for these features will most influence the class label of y_i.

4. As $\beta \to \infty$, the most frequently occurring training label in the neighbourhood of a test point will be chosen with increasing large probability. The β parameter can be thought of, in a sense, as a tempering parameter. In the limit as $\beta \to \infty$, the modal class label in the neighbourhood set has probability 1.

There has been work on extending the $k-$nearest neighbours algorithm to weight neighbours within the neighbourhood of size k. For example, (Dudani 1976) weighted neighbours using the distance in a linear manner while standardizing weights to lie in $[0, 1]$.

A model similar to the $d-\text{nn}_1$ model appeared in (Zhu and Ghahramani 2002), but it does not contain the β Markov random field parameter to control the level of aggregation in the spatial field. Moreover, the authors outline some MCMC approaches, but note that inference for this model is challenging. The aim of this paper is to illustrate how this model may be generalised and to illustrate an efficient algorithm to sample from this model. We now address the latter issue.

4 Implementing the distance-nearest neighbours algorithm

Throughout we consider a Bayesian treatment of this problem. The posterior distribution of test labels and Markov random field parameters can be expressed as

$$\pi(\boldsymbol{y}^*, \beta, \sigma | \boldsymbol{y}, \boldsymbol{x}, \boldsymbol{x}^*) \propto \pi(\boldsymbol{y}, \boldsymbol{y}^* | \beta, \sigma, \boldsymbol{x}, \boldsymbol{x}^*) \pi(\beta) \pi(\sigma),$$

where $\pi(\beta)$ and $\pi(\sigma)$ are prior distributions for β and σ, respectively. Note, however that the first term on the right hand side above depends on the intractable normalising constant (3). In fact, the number of test labels is often much greater than the number of training labels, and so the resulting normalising constant for the distribution $\pi(\boldsymbol{y}, \boldsymbol{y}^* | \beta, \sigma, \boldsymbol{x}, \boldsymbol{x}^*)$ involves a summation over G^{n+m} terms, where as before n, m and G are the number of test data points, training data points and class labels, respectively. A more pragmatic alternative is to consider the posterior distribution of the unknown parameters for the training class labels,

$$\pi(\beta, \sigma | \boldsymbol{x}, \boldsymbol{y}) \propto \pi(\boldsymbol{y} | \beta, \sigma, \boldsymbol{x}) \pi(\beta) \pi(\sigma),$$

where now the normalising constant depends on G^n terms. Test class labels can then be predicted by averaging over the posterior distribution of the training data,

$$\pi(y_{n+i} | x_{n+i}, \boldsymbol{x}, \boldsymbol{y}) = \int \pi(y_{n+i} | x_{n+i}, \boldsymbol{x}, \boldsymbol{y}, \beta, \sigma) \pi(\beta, \sigma | \boldsymbol{x}, \boldsymbol{y}) d\beta d\sigma.$$

Obviously, this assumes that the test class labels, $\boldsymbol{y}^*$ are mutually independent, given the training data, which will typically be an unreasonable assumption. The training class labels are modelled as being mutually independent. Clearly, this is not ideal from the Bayesian perspective. Nevertheless, it should reduce the computational complexity of the problem dramatically.

In practice, we can estimate the predictive probability of y_{n+i} as an ergodic average

$$\pi(y_{n+i}|x_{n+i}, \boldsymbol{x}, \boldsymbol{y}) \approx \frac{1}{J}\sum_{j=1}^{J} \pi(y_{n+i}|x_{n+i}, \boldsymbol{x}, \boldsymbol{y}, \beta^{(j)}, \sigma^{(j)}),$$

where $\beta^{(j)}, \sigma^{(j)}$ are samples from the posterior distribution $\pi(\beta, \sigma|\boldsymbol{x}, \boldsymbol{y})$.

4.1 Pseudolikelihood estimation

A standard approach to approximate the distribution of a Markov random field is to use a pseudolikelihood approximation, first proposed in (Besag 1974). This approximation consists of a product of easily normalised full-conditional distributions. For our model, we can write a pseudolikelihood approximation as

$$\pi(\boldsymbol{y}|\boldsymbol{x}, \beta, \sigma) \approx \prod_{i=1}^{n} \pi(y_i|\boldsymbol{y}_{-i}, \boldsymbol{x}, \beta, \sigma) = \prod_{i=1}^{n} \frac{\exp\left(\beta \sum_{j=1, j\neq i}^{n} w_j^i I(y_j = y_i)\right)}{\sum_{k=1}^{G} \exp\left(\beta \sum_{j=1, j\neq i}^{n} w_j^i I(y_j = k)\right)}.$$

This approximation yields a fast approximation to the posterior distribution, however it does ignore dependencies beyond first order.

4.2 The exchange algorithm

The main computational burden is sampling from the posterior distribution

$$\begin{aligned} \pi(\beta, \sigma|\boldsymbol{x}, \boldsymbol{y}) &\propto \pi(\boldsymbol{y}|\beta, \sigma, \boldsymbol{x})\pi(\beta)\pi(\sigma) \\ &= \frac{q(\boldsymbol{y}|\beta, \sigma, \boldsymbol{x})}{z(\beta, \sigma)}\pi(\beta)\pi(\sigma). \end{aligned}$$

A naive implementation of a Metropolis-Hastings algorithm proposing to move from (β, σ) to (β', σ') would require calculation of the following ratio at each sweep of the algorithm

$$\frac{q(\boldsymbol{y}|\beta', \sigma', \boldsymbol{x})\pi(\beta')\pi(\sigma')}{q(\boldsymbol{y}|\beta, \sigma, \boldsymbol{x})\pi(\beta)\pi(\sigma)} \times \frac{z(\beta, \sigma)}{z(\beta', \sigma')}. \tag{4}$$

The intractability of the normalising constants, $z(\beta, \sigma)$ and $z(\beta', \sigma')$, makes this algorithm unworkable. There has been work which has tackled the problem of sampling from such complicated distributions, for example, (Møller *et al* 2006). The algorithm presented in this paper overcomes the problem of sampling from a distribution with intractable normalising constant, to a large extent. However the algorithm can result in an MCMC chain with poor mixing among the parameters. The algorithm in (Møller *et al* 2006) has been extended and improved in (Murray, Ghahramani and MacKay 2006).

The algorithm samples from an augmented distribution

$$\pi(\beta', \sigma', \boldsymbol{y}', \sigma, \beta|\boldsymbol{y}, \boldsymbol{x}) \propto \pi(\boldsymbol{y}|\beta, \sigma, \boldsymbol{x})\pi(\beta)\pi(\sigma)h(\beta', \sigma'|\beta, \sigma)\pi(\boldsymbol{y}'|\beta', \sigma', \boldsymbol{x}), \tag{5}$$

where $\pi(\boldsymbol{y}'|\beta', \sigma', \boldsymbol{x})$ is the same distance nearest-neighbour distribution as the training data $\boldsymbol{y}$. The distribution $h(\beta', \sigma'|\beta, \sigma)$ is any arbitrary distribution for the augmented

variables (β', σ') which might depend on the variables (β, σ), for example, a random walk distribution centred at (β, σ). It is clear that the marginal distribution of (5) for variables σ and β is the posterior distribution of interest.

The algorithm can be written in the following concise way:

1. Gibbs update of $(\beta', \sigma', \boldsymbol{y}')$:
   (i) Draw $(\beta', \sigma') \sim h(\cdot, \cdot|\beta, \sigma)$.
   (ii) Draw $\boldsymbol{y}' \sim \pi(\cdot|\beta', \sigma', \boldsymbol{x})$.
2. Propose to move from $(\beta, \sigma, \boldsymbol{y}), (\beta', \sigma', \boldsymbol{y}')$ to $(\beta', \sigma', \boldsymbol{y}), (\beta, \sigma, \boldsymbol{y}')$. (Exchange move) with probability

$$\min\left(1, \frac{q(\boldsymbol{y}'|\beta, \sigma, \boldsymbol{x})\pi(\beta')\pi(\sigma')h(\beta, \sigma|\beta', \sigma')q(\boldsymbol{y}|\beta', \sigma', \boldsymbol{x})}{q(\boldsymbol{y}|\beta, \sigma, \boldsymbol{x})\pi(\beta)\pi(\sigma)h(\beta', \sigma'|\beta, \sigma)q(\boldsymbol{y}'|\beta', \sigma', \boldsymbol{x})} \times \frac{z(\beta, \sigma)z(\beta', \sigma')}{z(\beta, \sigma)z(\beta', \sigma')}\right).$$

Notice in Step 2, that all intractable normalising constants cancel above and below the fraction. The difficult step of the algorithm in the context of the $d-$nn model is Step 1 (ii), since this requires a draw from $\pi(\boldsymbol{y}'|\beta', \sigma', \boldsymbol{x})$. Perfect sampling (Propp and Wilson 1996) is often possible for Markov random field models, however a pragmatic alternative is to sample from $\pi(\cdot|\beta', \sigma', \boldsymbol{x})$ by standard MCMC methods, for example, Gibbs sampling, and take a realisation from a long run of the chain as an approximate draw from the distribution. Note that this is the approach that Cucala *et al.* (2009) take. They argue that perfect sampling is possible for the $pk-$nn algorithm for the case where there are two classes, but that the time to coalescence can be prohibitively large. They note that perfect sampling for more than two classes is not yet available.

Note that this algorithm has some similarities with Approximate Bayesian Computation (ABC) methods (Sisson, Fan and Tanaka 2007) in the sense that ABC algorithms also rely on drawing exact values from analytically intractable distributions. By contrast however, ABC algorithms rely on comparing summary statistics of the auxiliary data to summary statistics of the observed data. Finally, note that the Metropolis-Hastings ratio in step 2 above, after re-arranging some terms, and assuming that $h(\beta, \sigma|\beta', \sigma')$ is symmetric can be written as

$$\frac{q(\boldsymbol{y}|\beta', \sigma', \boldsymbol{x})\pi(\beta')\pi(\sigma')q(\boldsymbol{y}'|\beta, \sigma, \boldsymbol{x})}{q(\boldsymbol{y}|\beta, \sigma, \boldsymbol{x})\pi(\beta)\pi(\sigma)q(\boldsymbol{y}'|\beta', \sigma', \boldsymbol{x})}.$$

Comparing this to (4), we see that the ratio of normalising constants, $z(\beta, \sigma)/z(\beta', \sigma')$, is replaced by $q(\boldsymbol{y}'|\beta, \sigma, \boldsymbol{x})/q(\boldsymbol{y}'|\beta', \sigma', \boldsymbol{x})$, which itself can be interpreted as an importance sampling estimate of $z(\beta, \sigma)/z(\beta', \sigma')$, since

$$\mathbf{E}_{\boldsymbol{y}'|\beta', \sigma'}\left[\frac{q(\boldsymbol{y}'|\beta, \sigma, \boldsymbol{x})}{q(\boldsymbol{y}'|\beta', \sigma', \boldsymbol{x})}\right] = \int \frac{q(\boldsymbol{y}'|\beta, \sigma, \boldsymbol{x})}{q(\boldsymbol{y}'|\beta', \sigma', \boldsymbol{x})} \frac{q(\boldsymbol{y}'|\beta', \sigma', \boldsymbol{x})}{z(\beta', \sigma')}\, d\boldsymbol{y}' = \frac{z(\beta, \sigma)}{z(\beta', \sigma')}.$$

5 Results

The performance of our algorithm is illustrated in a variety of settings. We begin by testing the algorithm on a collection of benchmark datasets and follow this by

exploring two real datasets with high-dimensional feature vectors. Matlab computer code and all of the datasets (test and training) used in this paper can be found at `mathsci.ucd.ie/∼nial/dnn/`.

5.1 Benchmark datasets

In this section we present results for our model and in each case we compare results with the $k-$nn algorithm for well known benchmark datasets. A summary description of each dataset is presented in Table 1.

	G	F	N
Pima	2	8	532
Forensic glass	4	9	214
Iris	3	4	150
Crabs	4	5	200
Wine	3	13	178
Olive	3	9	572

Table 1: Summary of the benchmark datasets: G, F, N correspond to the number of classes, the number of features and the overall number of observations, respectively.

In all situations, the training dataset was approximately 25% of the size of the overall dataset, thereby presenting a challenging scenario for the various algorithms. Note that the sizes of each dataset ranges from quite small in the case of the iris dataset, to reasonably large in the case of the forensic dataset. In all examples, the data was standardised to give transformed features with zero mean and unit variance. In the Bayesian model, non-informative $N(0, 50^2)$ and $U(0, 100)$ priors were chosen for β and σ, respectively. Each $d-$nn algorithm was run for $20,000$ iterations, with the first $10,000$ serving as burn-in iterations. The auxiliary chain within the exchange algorithm was run for $1,000$ iterations. The $k-$nn algorithm was computed for values of k from 1 to half the number of features in the training set. In terms of computational run time, the $d-$nn algorithms took, depending on the size of the dataset, between 1 to 12 hours to run using Matlab code on a 2GHz desktop machine.

A summary of misclassification error rates is presented in Table 2 for various benchmark datasets. In almost all of the situations $d-\text{nn}_1$ and $d-\text{nn}_3$ performs at least as well as $k-$nn and often considerably better. In general, $d-\text{nn}_1$ and $d-\text{nn}_3$ performed better than $d-\text{nn}_2$. A possible explanation for this may be due to the cut-off nature of the weight function in the $d-\text{nn}_2$ model, since if a point x_i has no neighbours inside a ball of radius σ, then w_i^j is uniform over the entire test set, and consequently there is no effect of distance. By contrast, both the $d-\text{nn}_1$ and $d-\text{nn}_3$ models, have weight functions which depend on distance, and smoothly converge to a uniform distribution as $\sigma \to \infty$ and $\sigma \to 0$, respectively.

	$k-$nn	$d-\text{nn}_1$	$d-\text{nn}_2$	$d-\text{nn}_3$
Pima	30%	29%	32%	30%
Forensic glass	35%	33%	39%	31%
Iris	6%	5%	5%	6%
Crabs	16%	16%	23%	16%
Wine	6%	4%	6%	4%
Olive	1%	3%	4%	2%

Table 2: Misclassification error rates for various benchmark dataset. The value of k in the $k-$nn algorithm was chosen as the value that minimises the leave-one-out cross-validation error rate. (In the case of a tie, the smallest value of k was selected.)

5.2 Classification with large feature sets: food authenticity

Here we consider two datasets concerned with food authentication. The first example involves samples of Greek olive oil from 3 different regions, and the second example involves samples of 5 different types of meat. In both situations each sample was analysed using near infra-red spectroscopy giving rise to 1050 reflectance values for wavelengths in the range $400 - 2098$nm. These 1050 reflectance values serve as the feature vector for each sample. The objective in both examples is to authenticate a test sample based on a training set of complete data (both reflectance values and class labels). Details of how both datasets were collected appear in (McElhinney *et al* 1999), and were analysed using a model-based clustering approach in (Dean *et al* 2006).

5.2.1 Classifying meat samples

Here 231 samples of meat were collected. The aim of this study was to see if these measurements could be used to classify each meat sample according to whether it is chicken, turkey, pork, beef or lamb. The data were randomly split into 60 training samples and 171 test samples. The respective number of samples in each class is given in the table below.

	Training	Test
Chicken	15	40
Turkey	20	35
Pork	13	42
Beef	11	21
Lamb	11	23

Table 3: Number of samples within each class for both the training and test datasets

As before, non-informative normal, $N(0, 50^2)$ and uniform $U(0, 10)$ priors were chosen for β and σ, respectively. In the exchange algorithm, the auxiliary chain was run for 1000 iterations, and the overall chain ran for $20,000$ of which the first $10,000$ were discarded as burn-in iterations. The overall acceptance rate for the exchange algorithm was around 25% for each of the $d-$nn models.

The misclassification error rate for leave-one-out cross-validation on the training dataset is minimised for $k = 3$ and $k = 4$. See Figure 1 (a). At both of these values, the $k-$nn algorithm yielded a misclassification error rate of 35% and 39%, respectively, for the test dataset. See Figure 1 (b). By comparison, the $d-\text{nn}_1$, $d-\text{nn}_2$ and $d-\text{nn}_3$ models achieved misclassification error rates of 29%, 33% and 27%, respectively, for the test dataset. This example further illustrates the value of the $d-$nn models.

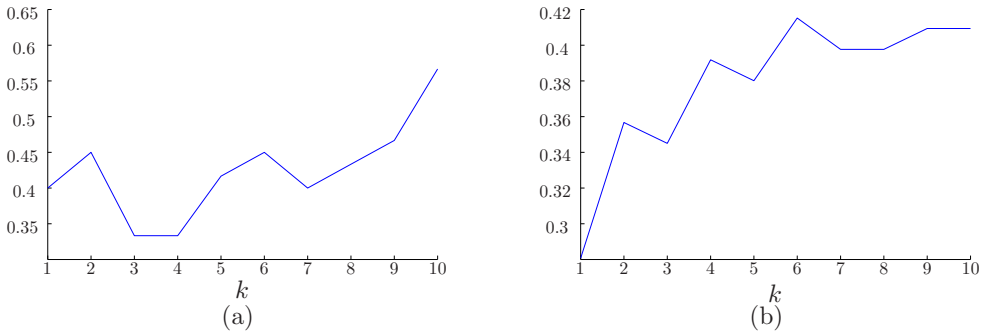

Figure 1: Meat dataset: (a) Training data: misclassification rates of leave-one-out cross-validation for $k-$nn algorithm for varying values of k. (b) Test data: misclassification rates for $k-$nn algorithm for varying values of k.

5.2.2 Classifying Greek olive oil

This example concerns classifying Greek oil samples, again based on infra-red spectroscopy. Here 65 samples of Greek virgin olive-oil were collected. The aim of this study was to see if these measurements could be used to classify each olive-oil sample to the correct geographical region. Here there were 3 possible classes (Crete (18 locations), Peloponnese (28 locations) and other regions (19 locations).

In our experiment the data were randomly split into a training set of 25 observations and a test set of 40 observations. In the training dataset the proportion of class labels was similar to that in the complete dataset.

In the Bayesian model, non-informative $N(0, 50^2)$ and $U(0, 100)$ priors were chosen for β and σ. In the exchange algorithm, the auxiliary chain was run for 1000 iterations, and the overall chain ran for $50,000$ of which the first $20,000$ were discarded as burn-in iterations. The overall acceptance rate for the exchange algorithm was around 15% for each of the Markov chains.

The $d-\text{nn}_1$, $d-\text{nn}_2$ and $d-\text{nn}_3$ models achieved misclassification rates of 20%, 26% and 20%, respectively. In terms of comparison with the $k-$nn algorithm, leave-one-out cross-validation was minimised for $k = 3$ for the training dataset. See Figure 2 (a). The misclassification rates at this value of k was 29% for the test dataset. See Figure 2 (b).

It is again encouraging that the $d-$nn algorithms yielded improved misclassification rates by comparison.

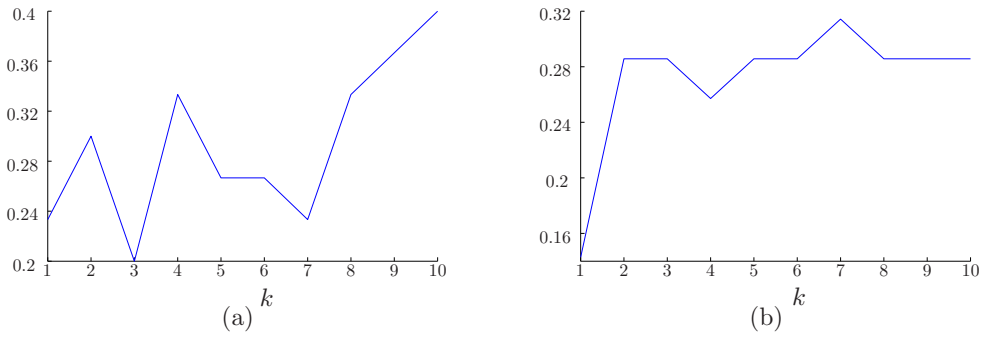

Figure 2: Olive oil dataset: (a) Training data: misclassification rates of leave-one-out cross-validation for $k-$nn algorithm for varying values of k. (b) Test data: misclassification rates for $k-$nn algorithm for varying values of k.

6 Concluding remarks

In terms of providing a probabilistic approach to a Bayesian analysis of supervised learning, our work builds on that of Cucala *et al* (2009) and shares many of the advantages of the approach there, providing a sound setting for Bayesian inference. The most likely allocations for the test dataset can be evaluated and also the uncertainty that goes with them. So this makes it possible to determine regions where allocation to specific classes is uncertain. In addition, the Bayesian framework allows for an automatic approach to choosing weights for neighbours or neighbourhood sizes.

The present paper also addresses the computational difficulties related to the well-known issue of the intractable normalising constant for discrete exponential family models. While Cucala *et al* (2009) demonstrated that MCMC sampling is a practical alternative to the perfect sampling scheme of Møller *et al* (2006), there remain difficulties with their implementation of the approach of (Møller *et al* 2006), namely the choice of an auxiliary distribution. To partially overcome the difficulties of a poor choice, Cucala *et al* (2009) use an adaptive algorithm where the auxiliary distribution is defined by using historical values in the Monte Carlo algorithm. We use an alternative approach based on the exchange algorithm which avoids this choice or adaptation and has very good mixing properties and therefore also has computational efficiency.

An issue with the neighbourhood model of Cucala *et al* (2009), which is an Ising or Boltzmann type model, is that it is necessary to define an upper value for the association parameter β. This parameter value arises from the phase change of the model and which is known for a regular neighbourhood structure but has to be investigated empirically for the probabilistic neighbourhood model. Our distance nearest neighbour models avoid this difficulty.

Our approach is robust to outliers whereas the nearest neighbour approaches will always have an outlying point having neighbours and therefore classified according to assumed independent distant points which are the nearest neighbours.

Acknowledgements: Nial Friel was supported by a Science Foundation Ireland Research Frontiers Program grant, 09/RFP/MTH2199. Tony Pettitt's research was sup-

ported by an Australian Research Council Discovery Grant.

References

Besag, J. E. (1974), Spatial interaction and the statistical analysis of lattice systems (with discussion). *Journal of the Royal Statistical Society, Series B* **36**, 192–236

Cucala, L., J.-M. Marin, C. P. Robert and D. M. T. Titterington (2009), A Bayesian reassessment of nearest-neighbour classification. *Journal of the American Statistical Association* **104**, 263–273

Dean, N., T. B. Murphy and G. Downey (2006), Using unlabelled data to update classification rules with applications in food authenticity studies. *Applied Statistics* **55**, 1–14

Dudani, S. A. (1976), The distance-weighted k-nearest neighbor rule. *IEEE Trans. Syst. Man. Cybern.,* **SMC-6**, 325–327

Hastie, T., R. Tibshirani and J. Friedman (2001), *The Elements of Statistical Learning, Data Mining, Inference, and Prediction*. Springer

Holmes, C. C. and N. M. Adams (2002), A probabilistic nearest-neighbour algorithm for statistical pattern recognition. *Journal of the Royal Statistical Society, Series B* **64**, 295–306

Holmes, C. C. and N. M. Adams (2003), Likelihood inference in nearest-neighbour classification models. *Biometrika* **90**, 99–112

McElhinney, J., G. Downey and T. Fearn (1999), Chemometric processing of visible and near infrared reflectance spectra for species identification in selected raw homogenised meats. *Journal of Near Infrared Spectroscopy* **7**, 145–154

Møller, J., A. N. Pettitt, R. Reeves and K. K. Berthelsen (2006), An efficient Markov chain Monte Carlo method for distributions with intractable normalising constants. *Biometrika* **93**, 451–458

Murray, I., Z. Ghahramani and D. MacKay (2006), MCMC for doubly-intractable distributions. In *Proceedings of the 22nd Annual Conference on Uncertainty in Artificial Intelligence (UAI-06)*, Arlington, Virginia, AUAI Press

Propp, J. G. and D. B. Wilson (1996), Exactly sampling with coupled Markov chains and applications to statistical mechanics. *Random Structures and Algorithms* **9**, 223–252

Sisson, S., Y. Fan and M. Tanaka (2007), Sequential Monte Carlo without likelihoods. *Proceedings of the National Academy of Sciences of the United States of America* **104**(6), 1760–1765

Zhu, X. and Z. Ghahramani (2002), Towards semi-supervised classification with Markov random fields. Technical Report CMU-CALD-02-106, Carnegie Mellon University