Available online at www.sciencedirect.com
ScienceDirect
Pattern Recognition Letters 28 (2007) 1818–1824
Pattern Recognition Letters
www.elsevier.com/locate/patrec

An empirical analysis of the probabilistic *K*-nearest neighbour classifier

S. Manocha [a,1], M.A. Girolami [b,*]

[a] *Department of Computer Science and Engineering, Indian Institute of Technology, Kharagpur, India*
[b] *Department of Computing Science, University of Glasgow, Glasgow, UK*

Received 13 July 2006; received in revised form 19 March 2007
Available online 8 June 2007

Communicated by F. Roli

Abstract

The probabilistic nearest neighbour (PNN) method for pattern recognition was introduced to overcome a number of perceived shortcomings of the nearest neighbour (NN) classifiers namely the lack of any probabilistic semantics when making predictions of class membership. In addition the NN method possesses no inherent principled framework for inferring the number of neighbours, *K*, nor indeed associated parameters related to the chosen metric. Whilst the Bayesian inferential methodology underlying the PNN classifier undoubtedly overcomes these shortcomings there has been to date no extensive systematic study of the performance of the PNN method nor any comparison with the standard non-probabilistic approach. We address this issue by undertaking an extensive empirical study which highlights the essential characteristics of PNN when compared to a cross-validated *K*-NN.

Keywords: *K*-nearest neighbour; Non-parametric classification; Bayesian classification; Monte Carlo estimation

1. Introduction

The *K*-nearest neighbour (KNN) method for classification is one of the most straightforward approaches to classifying objects which are represented as points defined in some feature space. Despite the simplicity of KNN the performance it achieves on a number of pattern recognition tasks indicates that it remains competitive as a classification method (Ripley, 1996). Indeed asymptotic analysis of the nearest neighbour rule shows that the error rate will be no greater than twice the Bayes error rate (see Chapter 6 of Ripley, 1996 for a review of such results). However, from a methodological perspective there are a number of shortcomings with the KNN rule which make further development and enhancements of the method difficult without resorting to ad hoc solutions. The main shortcoming of KNN is the lack of any probabilistic semantics which would allow posterior predictive probabilities to be employed in, for example, assigning variable losses in a consistent manner. In addition the selection of the value of *K*, the number of nearest neighbours, is not straightforward without resorting to cross-validation (CV). However, whilst one can employ CV to select a single value of *K* inferring the metric of similarity, or parameters associated with the metric require additional approaches some of which have been proposed in (Hastie and Tibshirani, 1996; Pardes and Vidal, 2000, 2006a; Shakhnarovish et al., 2005).

If a probabilistic model can be defined for KNN then issues surrounding obtaining properly calibrated continuous predictive probabilities and obtaining parameters related to the model (i.e. *K* and possible metric parameters) are resolved consistently and naturally within the Bayesian inferential framework. In a landmark paper (Holmes and Adams, 2002) a probabilistic KNN method for pattern recognition was introduced. By defining an approximate joint

* Corresponding author. Fax: +44 141 330 8627.
E-mail addresses: manochaiitkgp@gmail.com (S. Manocha), girolami@dcs.gla.ac.uk (M.A. Girolami).
[1] This work was undertaken whilst S. Manocha was visiting the Department of Computing Science, University of Glasgow.

0167-8655/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2007.05.018

distribution the authors of Holmes and Adams (2002) show that posterior inference over K can be performed in a relatively straightforward manner employing standard Markov Chain Monte Carlo (MCMC) methods. The inferential framework now available for probabilistic KNN (PKNN) has been exploited in (Everson and Fieldsend, 2004) where posterior inference over the similarity metric has been demonstrated by further developing the MCMC procedure of Holmes and Adams (2002). The work of Everson and Fieldsend (2004) illustrates the natural way in which the inferential framework for PKNN can be extended to deal with metric learning without having to resort to introducing any ancillary criteria or learning objectives.

However, despite the clear methodological benefits of the PKNN method it is unclear whether these advantages come at a cost, when compared to standard KNN, in terms of eventual classification accuracy, computational overhead, or indeed if increased accuracy of PKNN can be consistently observed over a diverse range of classification problems and data sets. In this communication we address the issue by conducting an extensive series of experiments to compare the performance of PKNN and KNN under the 0–1 classification loss. This is measured over a diverse range of classification problems with various dimensions of feature representation, size of available data sample, and number of classes. In addition we provide an illustrative study of the differences in the underlying mechanisms for choosing the number of nearest neighbours employing CV for KNN and the posterior inference employed by PKNN.

2. Probabilistic *K*-nearest neighbours

Consider a finite data sample $\{(t_1, \boldsymbol{x}_1), \ldots, (t_N, \boldsymbol{x}_N)\}$ where each $t_n \in \{1, \ldots, C\}$ denotes the class label associated with the D-dimensional feature vector $\boldsymbol{x}_n \in \mathbb{R}^D$ and the feature space $\mathbb{R}^D$ has an associated metric with parameters $\boldsymbol{\theta}$ denoted as $\mathscr{M}_{\boldsymbol{\theta}}$. To define a probabilistic representation of the KNN method an approximate conditional joint likelihood is defined in (Holmes and Adams, 2002) such that

$$p(\boldsymbol{t}|\boldsymbol{X}, \beta, k, \boldsymbol{\theta}, \mathscr{M}) \approx \prod_{n=1} \frac{\exp\left\{\frac{\beta}{k}\sum_{j\sim n|k}^{\mathscr{M}_{\boldsymbol{\theta}}} \delta_{t_n t_j}\right\}}{\sum_{c=1}^{C} \exp\left\{\frac{\beta}{k}\sum_{j\sim n|k}^{\mathscr{M}_{\boldsymbol{\theta}}} \delta_{c t_n}\right\}}, \tag{1}$$

where we define the $N\times 1$ dimensional vector $\boldsymbol{t}$ as $[t_1, \ldots, t_N]^{\mathrm{T}}$ and the $N\times D$ dimensional matrix $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N]^{\mathrm{T}}$, $\mathscr{M}$ denotes the metric employed in the feature space and $\boldsymbol{\theta}$ are the associate parameters. The number of nearest neighbours is k and β defines a scaling variable. The expression

$$\sum_{j\sim n|k}^{\mathscr{M}_{\boldsymbol{\theta}}} \delta_{t_n t_j}$$

denotes the number of the nearest k neighbours of $\boldsymbol{x}_n$, as measured under the metric $\mathscr{M}_{\boldsymbol{\theta}}$ within $N-1$ samples from $\boldsymbol{X}$ remaining when $\boldsymbol{x}_n$ is removed which we denote as $\boldsymbol{X}_{-n}$, and have the class label value of t_n, whilst each of the terms in the summation of the denominator provides a count of the number of the k neighbours of $\boldsymbol{x}_n$ which have class label equaling c.

It should be noted that the right-hand side is a product of terms of the form $p(t_n|\boldsymbol{x}_n, \boldsymbol{X}_{-n}, \boldsymbol{t}_{-n}, \beta, k, \boldsymbol{\theta}, \mathscr{M})$ which can be viewed as a Leave-One-Out (LOO) predictive likelihood, where $\boldsymbol{t}_{-n}$ denotes the vector $\boldsymbol{t}$ with the nth element removed, and as such the approximate joint likelihood above provides an overall measure of the LOO predictive likelihoods which we would anticipate should exhibit some resiliance to overfitting due to the LOO nature of the approximate likelihood.

Full posterior inference will follow by obtaining the parameter posterior distribution $p(\beta, k, \boldsymbol{\theta}|\boldsymbol{t}, \boldsymbol{X}, \mathscr{M})$ and subsequent predictions of the target class label t_* of a new datum $\boldsymbol{x}_*$ are made by posterior averaging such that

$$p(t_*|\boldsymbol{x}_*, \boldsymbol{t}, \boldsymbol{X}, \mathscr{M}) = \sum_k \int p(t_*|\boldsymbol{x}_*, \boldsymbol{t}, \boldsymbol{X}, \beta, k, \boldsymbol{\theta}, \mathscr{M}) p(\beta, k, \boldsymbol{\theta}|\boldsymbol{t}, \boldsymbol{X}, \mathscr{M}) \mathrm{d}\beta \, \mathrm{d}\boldsymbol{\theta}$$

as the required posterior takes an intractable form an MCMC procedure is proposed in (Holmes and Adams, 2002) and extended in (Everson and Fieldsend, 2004) to enable metric inference so that the following Monte Carlo estimate is employed

$$\hat{p}(t_*|\boldsymbol{x}_*, \boldsymbol{t}, \boldsymbol{X}, \mathscr{M}) = \frac{1}{N_s}\sum_{s=1}^{N_s} p(t_*|\boldsymbol{x}_*, \boldsymbol{t}, \boldsymbol{X}, \beta^{(s)}, k^{(s)}, \boldsymbol{\theta}^{(s)}, \mathscr{M}),$$

where each $\beta^{(s)}$, $k^{(s)}$, $\boldsymbol{\theta}^{(s)}$ are samples obtained from the full parameter posterior $p(\beta, k, \boldsymbol{\theta}|\boldsymbol{t}, \boldsymbol{X}, \mathscr{M})$ using a Metropolis style sampler.

As the standard KNN method has no straightforward way to learn the metric we restrict this study to posterior inference over k and β[2] and fix the metric to the standard Euclidean metric for both KNN and PKNN. In this way CV can be employed to select an optimal value K[3] for the number of nearest neighbours in KNN classification. Whilst metric learning for standard KNN has been developed in for example (Pardes and Vidal, 2006a) our main focus is to seek an understanding of the underlying mechanisms of PKNN and KNN employing CV and the added complexities of metric learning and inference will only add unnecessary complications to our proposed experimental analysis. We therefore adopt the Metropolis scheme detailed in (Holmes and Adams, 2002) and obtain samples from the posterior $p(\beta, k, |\boldsymbol{t}, \boldsymbol{X}, \mathscr{M})$ and employ Monte

[2] The random variable β is a scaling coefficient used in producing the required $p(t_n|\boldsymbol{x}_n, \boldsymbol{X}_{-n}, \boldsymbol{t}_{-n}, \beta, k, \boldsymbol{\theta}, \mathscr{M})$ terms. As there is no comparable variable in the KNN and given that β is of secondary importance to the probabilities we will not consider it in any great detail within the paper.

[3] We will employ K to denote the optimal number of nearest neighbours identified using 10CV and k to denote the random integer in the PKNN method.

Carlo estimates $\hat{p}(t_*|\boldsymbol{x}_*, \boldsymbol{t}, \boldsymbol{X}, \mathscr{M}) = \frac{1}{N_s}\sum_{s=1}^{N_s} p(t_*|\boldsymbol{x}_*, \boldsymbol{t}, \boldsymbol{X}, \beta^{(s)}, k^{(s)}, \mathscr{M})$ in the following experimental section.

3. Experiments

As a first example we consider the toy synthetic binary classification problem devised in (Ripley, 1996) and employed in (Holmes and Adams, 2002) to demonstrate the PKNN method. We draw 100,000 samples from the posterior $p(\beta, k|\boldsymbol{t}, \boldsymbol{X}, \mathscr{M})$ and employ these samples in obtaining the required Monte Carlo estimates of the predictive posteriors for each point in the independent test set. A hard classification decision is made by thresholding the posteriors at the value of 0.5 and the percentage prediction error under a 0–1 loss is estimated for PKNN. We achieve a test error rate of 8.4% which is identical to that reported in (Holmes and Adams, 2002). In addition we employ tenfold cross-validation on the training data to select an optimal value of K which will be employed in the KNN classifier when making predictions on the test set. The top bar chart in Fig. 1 shows a histogram of the marginal posterior for k based on the 100,000 samples drawn using a bin width of 10. Again it is clear that the three 'modes' for k which are clearly visible reflect the results of Holmes and Adams (2002). It is interesting to consider the evolution of the overall percentage test error as the Monte Carlo estimates of the predictive probabilities improve with MCMC sampling. Fig. 3 shows the percentage test error evolution over the first 500 samples drawn from the Metropolis sampler, it is clear that convergence to the optimal test error is rapid after of the order of 300 samples.

Consider the case where the available training set varies in size. Smaller available training sets may make the choice of an optimal value for K difficult when using cross-validation, in such a situation the Bayesian formalism of PKNN may yield improved performance. To assess this we sub-sample the available training data for the Ripley synthetic data set at various sizes of sub-sample ranging from 25 to 250 in step of 25. Each sub-sampling was carried out 50 times and the classifier performance for KNN and PKNN was obtained for each of the 50 training sets at each sub-sample size. Fig. 2 shows the mean percentage error (and

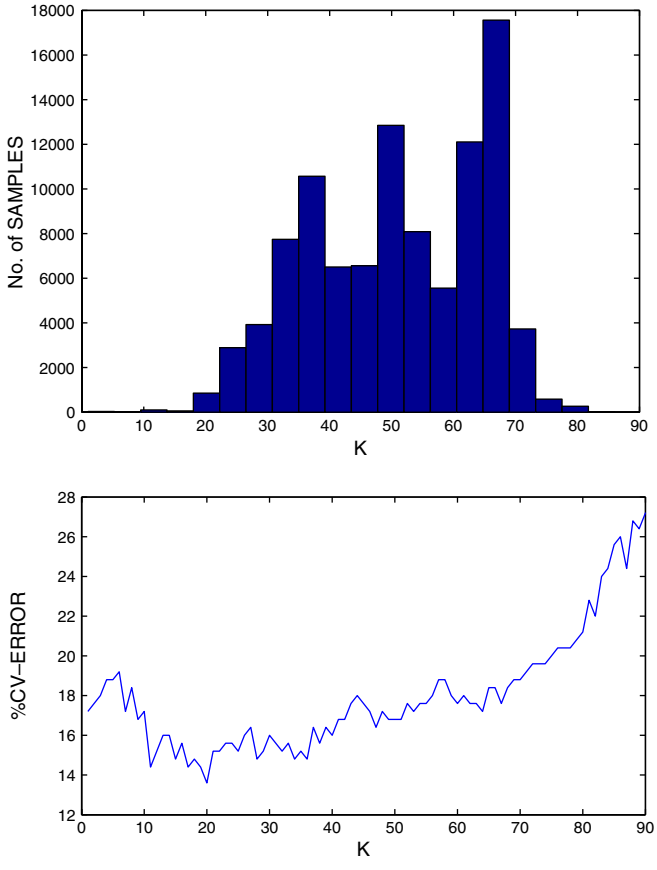

Fig. 1. The top graph shows a histogram of the marginal posterior for K on the synthetic Ripley data set and the bottom shows the 10CV error against the value of K.

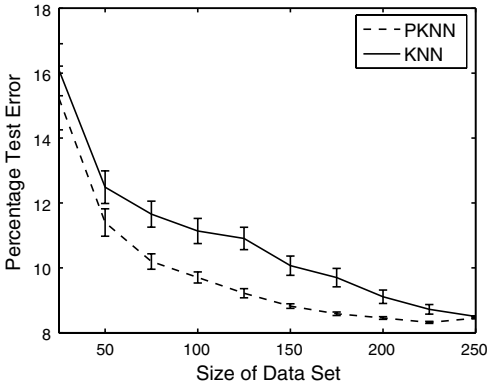

Fig. 2. The percentage test error obtained with training sets of varying size from 25 to 250 data points. For each sub-sample size, 50 random subsets were sampled and each of these used to obtain a KNN and PKNN classifier which were then used to make predictions on the 1000 independent test points. The mean percentage performance and associated standard error obtained for each training set are shown in the above figure for each classifier.

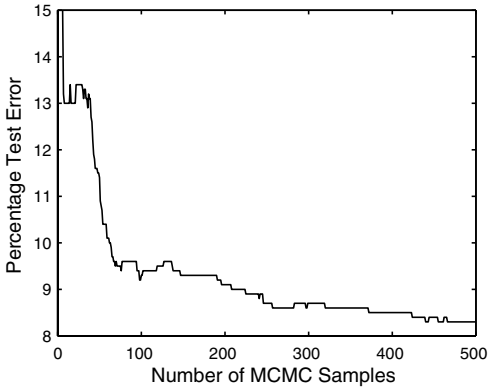

Fig. 3. The evolution of the total percentage test error on the synthetic Ripley data set. The predictive posteriors are estimated using all the samples drawn from the Markov chain from 1 to 500, it is clear that after around 400 samples the PKNN is operating at close to the peak achievable performance.

standard error) for both PKNN and KNN which illustrates that the Bayesian averaging of the PKNN makes it less prone to the loss of performance incurred by the CV tuned KNN classifier, at least for this data set.

If we now consider the estimated prediction error using tenfold cross-validation on the training set for a range of values of K from $K = 1$ to $K = 90$, Fig. 1 bottom chart, we see that the estimated expected 0–1 loss over the range of values of K considered is somewhat pessimistic with the estimated error being around twice as high as that obtained on the test set sample which was also 8.4% when $K = 20$ was selected. So we see that selecting a single value of K using tenfold cross-validation (10CV) and employing this in the standard KNN classifier yields the same 0–1 error rate as that achieved by PKNN on this particular data set. Of course this is an incomplete picture as we only consider 0–1 loss (KNN is non-probabilistic) and in addition an assessment of the differences between the two approaches should be made on a diverse range of data collections with varying size of samples, dimensionality of feature space and number of classes.

The following section now presents further experiments to address this issue.

3.1. Experimental procedure

In this paper, we have tested the classification performance of the two methods viz. KNN and PKNN on a diverse collection of data sets. These have been selected in such a way that the total number of samples per data set, the feature dimensionality and the number of classes cover an appropriate range of values which will provide a comprehensive comparison of the two classification methods. The data sets were obtained from the UCI Machine Learning Repository[4] with the exception of Crabs and Pima[5] and the summary table (Table 1) details the number of samples (N), number of associated classes (C) and the dimensionality of the feature representations (D).

Each data set was randomly permuted with no normalisation of the features, with the exception of Wine, Balance, Heart and Soybean where we have standardised the feature vectors i.e. shifting and scaling to zero mean and unit variance. This was primarily due to the mismatch in scale of certain features within these data sets which affected the similarity measure induced by the Euclidean metric employed. Clearly learning the metric by for example inferring appropriate length scales would obviate the requirement for such normalisation which is not linked to optimising the predictive capability of the classifier.

To obtain an estimate of the 0–1 loss classification error we have employed 10CV on both KNN and PKNN. In addition, to obtain an optimal value for the number of nearest neighbours in KNN for each fold we have further used 10CV to estimate the predictive error for each value of K ranging from $K = 1$ to $K = N_f$ where N_f is the number of data points in the current fold i.e. if $N = 100$ then each $N_f = 90$. Once the 10CV errors for each K are obtained we select the smallest value of K which yields the minimum 10CV error within the specific fold. In the case of multiple minima we select the smallest value of K which yields this minimum. This means that each fold may have a different optimal value for the number of nearest neighbours.

Table 1
Summary table of the data sets employed in the experiments undertaken

Data	N	C	D
Glass	214	6	9
Iris	150	3	4
Crabs	200	4	5
Pima	200	2	7
Soybean	48	4	35
Wine	178	3	13
Balance	625	3	4
Heart	270	2	13
Liver	345	2	6
Diabetes	768	2	8
Vehicle	846	4	18

N, C, D are respectively the total number of data points, the number of classes and the number of attributes in each data set.

For PKNN we also employed 10CV to obtain our estimate of the predictive error (under 0–1 loss). Within each fold we sampled from the posterior using the MCMC method outlined in (Holmes and Adams, 2002). The proposal distributions were tuned such that an acceptance rate of 35–40% for the Metropolis sampler was achieved and then 100,000 samples were drawn. These were then employed in the Monte Carlo estimate of the predictive likelihood for each test point in the fold and a classification was made based on the maximum of the class posterior. This sampling and Monte Carlo estimation was run for each of the tenfolds to obtain the overall estimate of classification error.

3.2. Analysis of experimental results

Let us consider first the overall classification performance of both KNN and PKNN. The results are listed in Table 2 and are summarised as a percentage mean error and standard deviation computed over the tenfolds. In the fourth column the P-value obtained from performing a Wilcoxon Rank-Sum non-parametric test of difference in medians is listed[6] this gives the probability that there is no detectable difference in the overall performance of the two classifiers (KNN and PKNN) for the particular data set.

Studying the mean errors for each data set and the corresponding P-values it is clear that over all data sets KNN

[4] http://www.ics.uci.edu/~mlearn/MLSummary.html.
[5] http://www.stats.ox.ac.uk/pub/PRNN/.
[6] A t-test was considered inappropriate in this case as the distribution of errors over the folds, in many of the data sets, did not conform to a normal distribution and as such a non-parametric test was selected.

Table 2
Comparison of Results for KNN and PKNN

Data	KNN	PKNN	*P*-value
Glass	29.91 ± 9.22	26.67 ± 8.81	0.517
Iris	5.33 ± 5.25	4.00 ± 5.62	0.537
Crabs	15.00 ± 8.82	19.50 ± 6.85	0.240
Pima	27.00 ± 8.88	24.00 ± 14.68	0.645
Soybean	14.50 ± 16.74	4.50 ± 9.56	0.155
Wine	3.922 ± 3.77	3.37 ± 2.89	0.805
Balance	11.52 ± 2.99	10.23 ± 3.02	0.324
Heart	15.18 ± 5.91	15.18 ± 4.43	1.000
Liver	33.60 ± 6.98	36.26 ± 12.93	0.705
Diabetes	25.91 ± 7.15	25.25 ± 8.11	0.970
Vehicle	36.28 ± 5.16	37.22 ± 4.53	0.732

The tenfold cross-validated results for KNN and PKNN reported as mean and standard deviation. The *P*-value reported is obtained from a Wilcoxon rank-based test.

and PKNN perform equally well with no set of results achieving a significance level of say 5%. There are two cases where there is weak evidence to suggest superior performance of PKNN (Soybean with a *P*-value of 15%) and KNN (Crabs with a *P*-value of 24%). However, these *P*-values are insufficient to reject the null-hypothesis that both PKNN and KNN are of equal performance. For the remaining data sets the performance for both KNN and PKNN are indistinguishable.

In summary, over a diverse set of classification problems there is no evidence to suggest that KNN, where an optimal K value is selected by cross-validation, or PKNN perform in terms of classification error any differently from each other. This in itself is an interesting result as PKNN does not have to resort to any form of out of sample validation to avoid any overfitting. However, assessing averaged performance of both methods does not provide us with an understanding of how both methods differ in operation and so in the following section we will consider examples of where the performance of (a) PKNN outperforms KNN, (b) KNN outperforms PKNN, and (c) where both KNN and PKNN have similar performance.

3.3. An example of situations where PKNN outperforms KNN

We have selected the results from a specific fold (number 3) from the Pima data set where the fold error for KNN is 40% whilst for PKNN it is 0%. We now consider how this difference in performance has occurred. Fig. 4 (top plot) shows the unnormalised histogram of the marginal posterior $p(k|\boldsymbol{t}_f, \boldsymbol{X}_f, \mathscr{M})$ (where subscript f denotes the specific fold of the data) and the bottom plot shows how the 10CV within-fold error varies with K. The first thing to note is that as an automated form of selecting an optimal K is being employed here 10CV selects $K = 11$, however, we can see that this is a rather unstable value and any perturbation from the selected K value will incur an increase in classification error of between 2% and 4%. A wiser choice of K based on 10CV inter-fold error would be in the range

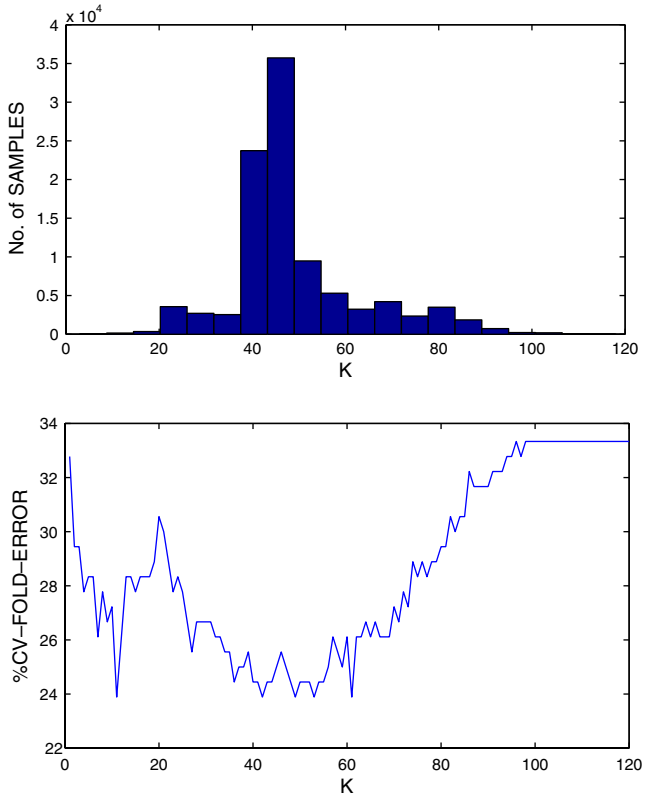

Fig. 4. The top plot shows the histogram of the marginal posterior for k and the bottom plot shows the 10CV error versus the value of k for the third fold of the Pima data set.

of $K = 40$ to $K = 60$. On the other hand if we consider the marginal posterior shown in the top plot of Fig. 4 it is clear that there is no posterior mass at the low values of k where the unstable minimum occurs and indeed the posterior mass is concentrated at the values of k where the minimum inter-fold 10CV errors occur i.e. $k = 40$ to $k = 60$. Now as the predictive probabilities $p(t_*|\boldsymbol{x}_*, \boldsymbol{t}, \boldsymbol{X}, \mathscr{M})$ are averaged with respect to the posterior we can observe how the posterior smoothing avoids the unstable local minimum and averages over the stable range of k values. This is a nice illustration of how the Bayesian averaging operates in this method and how it can improve on naive selection of a single optimal value.

3.4. An example of situations where KNN outperforms PKNN

From Fig. 5 and graphs of the other folds which exhibit similar characteristics (which are not presented here) we observe that when there is a single distinct minimum in the graph of K versus 10CV error for KNN at $K = 1$ i.e. the 10CV error is strictly increasing then the majority of posterior samples for PKNN are localised at the correspondingly appropriate value of $k = 1$. However, due to the posterior sampling there are regions of the posterior which are explored where the likelihood is quite low, corresponding to a large 10CV error, then we observe that the

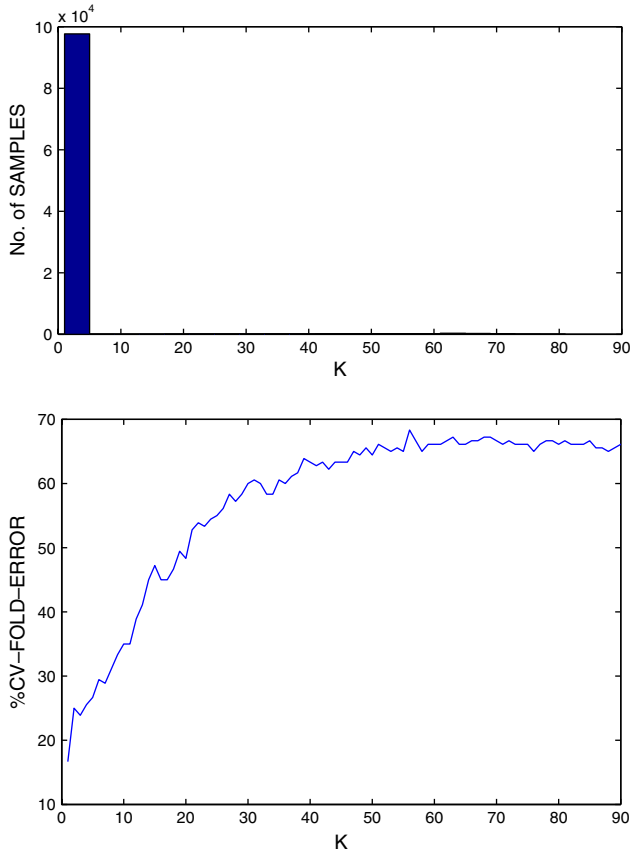

Fig. 5. The top plot shows the histogram of the marginal posterior for k and the bottom plot shows the 10CV error versus the value of k for the third fold of the Crabs data set.

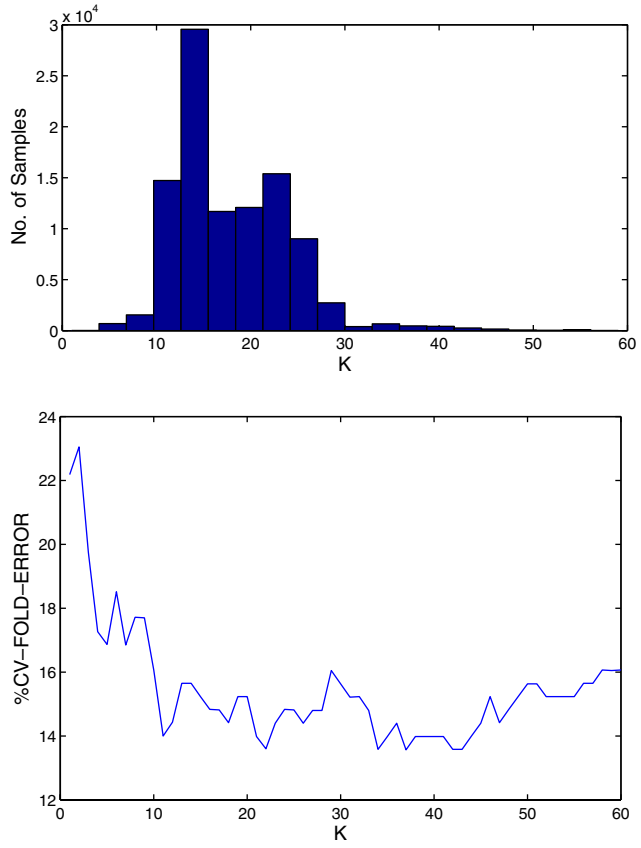

Fig. 6. The top plot shows the histogram of the marginal posterior for k and the bottom plot shows the 10CV error versus the value of k for the first fold of the Heart data set.

0–1 error performance of KNN is better than PKNN. From the 100,000 samples obtained from the posterior there are of the order of 2000 samples drawn for $k = 50$ which yields a high 10CV error and the remaining samples from the posterior are unable to compensate for this induced error. Of course if we were to increase the number of samples drawn from the posterior then this negative effect would be smoothed out. However, this may require a much larger number of samples and it is unclear how, in a practical manner, this number could be selected.

3.5. An example of where PKNN and KNN errors are equal

Fig. 6 shows the posterior and 10CV error versus K for the first fold of the Heart data set and here the performance of KNN and PKNN is exactly the same i.e. 18.5% fold error. This is due to a region of the 10CV error response which is relatively stable and is insensitive to the values of K.

On the whole KNN tends to perform better when there is a unique value at $K = 1$ and this is due to the finite sampling effects of PKNN. However, when the values of K yielding lowest 10CV errors are spread over an unstable region of the error response then in general the Bayesian averaging of PKNN has a significant beneficial effect.

4. Computational cost of PKNN and KNN

In terms of computational cost it may be suggested that full MCMC inference has unacceptably high overhead. We have monitored the raw compute times, on a 1.6 GHz Intel based Personal Computer with 2 Gb RAM, to obtain the overall 10CV estimated error for both KNN[7] and PKNN[8] for each data set and have listed these in Table 3. Inspection of these times clearly indicates that PKNN takes between five and ten times as long as KNN[9] does in obtaining the 10CV errors.[10] The data sets which have a large number of samples KNN (with no fast search) takes slightly longer in some cases (Balance, Diabetes, Vehicle)

[7] Employing 10CV in each fold to obtain a single fold-specific optimal value of K.

[8] Where MCMC sampling and averaging using 100,000 posterior samples for each fold is employed. We should note that the tuning of the proposal distributions to obtain the required acceptance rates has not been included here.

[9] It should be noted that this study has used a standard naive implementation of KNN which does not take advantage of any fast search techniques.

[10] We should note that in (Holmes and Adams, 2002) the point is made that quadrature could be employed when estimating the predictive posteriors.

Table 3
The table showing the running times for different data sets for KNN and PKNN

Data	KNN	PKNN
Glass	39.55	243.52
Iris	7.58	91.8
Crabs	21.99	156.30
Pima	24.10	103.60
Soybean	1.16	38.38
Wine	27.9	144.90
Balance	609.86	555.72
Heart	96.11	145.22
Liver	116.71	189.73
Diabetes	1643.09	567.03
Vehicle	4226.69	1063.13

The running times (s) for KNN with no fast search and PKNN (using Monte Carlo) to compute the tenfold cross-validation errors.

than PKNN and this is due to the exhaustive computation of 10CV error for each value of K from 1 to the number of samples in the nine 'training' folds. In our experimental protocol the maximum value which k could be sampled from is the same as that for KNN. However, whilst in KNN each value of K up to the maximum will be tested only a small number of these possible values end up being explored in PKNN. In summary, it is clear that there is an additional computational overhead incurred of PKNN however, for this particular classification problem this is not such a significant issue. However, we should note that when sampling metric specific parameters i.e. (Everson and Fieldsend, 2004) then computation of all distances have to be re-computed as does the selection of the k nearest neighbours for each values of k and β sampled and in this case the time required for PKNN will be significantly greater.

5. Conclusions and discussion

In this contribution, we have undertaken an experimental analysis of the PKNN classification method as proposed in (Holmes and Adams, 2002) and evaluated its performance and characteristics with a cross-validated KNN classifier. Employing the overall miss-classification rate under a 0–1 loss it is found that there is no significant statistical evidence to suggest that either method is a more accurate classifier than the other based on the selected collection of data sets studied. Whilst there is no outright performance advantage of PKNN over KNN the main advantage of PKNN is methodological. PKNN provides continuous predictive probabilities in a natural manner which gives a way of allocating uneven miss-classification costs and further propagating these levels of predictive uncertainty as a part of further possible downstream processing. In addition the Bayesian inferential framework allows extension of the levels of inference to for, example, metric learning as has been demonstrated in (Everson and Fieldsend, 2004) and can be, in a most straightforward manner, extended to accommodate class-specific metrics and indeed combinations of metrics. It is also possible to consider data-condensation by the introduction of a binary indicator variable for each data point within the available training set and obtaining the posterior for each of these variables. This posterior would indicate which data points would be retained within the data set and which would be removed.

The form of the pseudo-likelihood defined for PKNN is reminiscent of a predictive 'Leave-One-Out' likelihood and the Bayesian posterior averaging in many cases provides a means of protection from 'over-fitting' without the need for any form of explicit cross-validation to select the number of nearest neighbours. The analysis of the experimental results have shown specific cases where both PKNN and KNN may achieve superior predictive performances and this has provided insight into the underlying mechanisms governing the performance of each method.

Whilst it would be of possible value to further consider evaluations of performance with variable metric PKNN (Everson and Fieldsend, 2004) and methods such as Hastie and Tibshirani (1996), Pardes and Vidal (2000, 2006b) the main contribution of this paper has clarified what can be reasonably expected in terms of relative performance of both probabilistic and non-probabilistic approaches as well as highlighting certain data characteristics which may have an impact on their levels of performance.

Acknowledgements

S. Manocha is grateful for the funding provided by the Department of Computing Science Research Committee in partial support of his visit to the University of Glasgow.

References

Everson, Richard M., Fieldsend, Jonathan E., 2004. A variable metric probabilistic k-nearest-neighbours classifier. In: Intelligent Data Engineering and Automated Learning-IDEAL 2004, pp. 654–659.

Hastie, T., Tibshirani, R., 1996. Discriminant adaptive nearest neighbour classification and regression. Adv. Neural Inf. Process. Systems 8, 409–415.

Holmes, C.C., Adams, N.M., 2002. A probabilistic nearest neighbour method for statistical pattern recognition. J. Roy. Statist. Soc. Ser. B 64 (2), 295–306.

Pardes, R., Vidal, E., 2000. A class-dependent weighted dissimilarity measure for nearest neighbour classification problems. Pattern Recognition Lett. 21, 1027–1036.

Pardes, R., Vidal, E., 2006a. Learning weighted metrics to minimize nearest-neighbour classification error. IEEE Trans. Pattern Anal. Machine Learn. 28 (7).

Pardes, R., Vidal, E., 2006b. Learning prototypes and distances: A prototype reduction technique based on nearest neighbour error minimisation. Pattern Recognition 39 (2), 180–188.

Ripley, B.D., 1996. Pattern Recognition and Neural Networks. Cambridge University Press.

Shakhnarovich, G., Darrell, T., Indyk, P., 2005. Nearest-Neighbor Methods in Learning and Vision. The MIT Press.