Bayesian Kernel Projections for Classification of High Dimensional Data

May 5, 2009

Katarina Domijan and Simon P. Wilson

Abstract

A Bayesian multi-category kernel classification method is proposed. The hierarchical model is treated with a Bayesian inference procedure and the Gibbs sampler is implemented to find the posterior distributions of the parameters. The practical advantage of the full probabilistic model-based approach is that probability distributions of prediction can be obtained for new data points, which gives a more complete picture of classification. Large computational savings and improved classification performance can be achieved by a projection of the data to a subset of the principal axes of the feature space. The algorithm is aimed at high dimensional data sets where the dimension of measurements exceeds the number of observations. The applications considered in this paper are microarray, image processing and near-infrared spectroscopy data.

1 Introduction

Supervised learning for classification can be formalized as the problem of inferring a function $f(\mathbf{x})$ from a set of n training samples $\mathbf{x}_i \in \mathbb{R}^J$ and their corresponding class labels $\mathbf{y}_i$. The model developed in this paper is aimed at multi-category classification problems. Of particular interest is classification of high dimensional data, where each sample is defined by hundreds or thousands of measurements, usually concurrently obtained. Such data arise in many application domains, for example, the genomic and proteomic technologies, and their rapid emergence in the last decade has generated much interest in the statistical community, as analysis of such data requires novel statistical techniques. The applications considered in this paper are microarray, image processing and near-infrared (NIR) spectroscopy data where the dimension of the variables J exceeds ten to twenty - fold the number of samples n.

In this paper we present the Bayesian Kernel Projection Classifier (BKPC), a multi-category classification method based on the reproducing kernel Hilbert spaces (RKHS) theory. The proposed classifier performs classification of high dimensional data without any pre-processing steps to reduce the number of variables. RKHS methods allow for non-linear generalization of linear classifiers by implicitly mapping the classification problem into a high dimensional feature space where the data is thought to be linearly separable. Due to the reproducing property of the RKHS, the classification is actually carried out in the subspace of the feature space which is of dimension $n << J$. Kernel classifiers

function through the kernel matrix $\mathbf{K}$ whose dimension is only dependent on the number of observations n. If they work well, they condense the information in the large variable space to this smaller dimension. Therefore, kernel methods are ideally suited for high dimensional data, such as the data sets considered in this paper. The main difference between the BKPC and other kernel classifiers is that BKPC is constructed so that it performs the classification of the projections of the data to the principal axes of the feature space. Large computational savings and improved classification performance can be achieved if the underlying structure of the feature space can be adequately summarized by a small subset of the principal axes.

Kernel methods were first introduced into statistical learning by (1) and later re-introduced by (2) who constructed the Support Vector Machine, a generalization of the optimal hyperplane algorithm for binary classification. Bayesian treatments of this popular deterministic statistical learning method were motivated by the need to overcome the problem of quantifying uncertainty of SVM predictions, as Bayesian framework allows for probabilistic outputs to be obtained from the predictive distribution. Statistical learning models usually have complex structure and contain parameters that need to be tuned, which is often done via cross-validation. In can be argued, see for example (3), that the Bayesian framework is a natural setting for statistical learning algorithms, as decisions on the complexity of structure and parameter settings can be approached by specifying prior distributions, which formalizes the prior beliefs about which inputs are relevant, what a distribution of a parameter is or how smooth a function is.

Many Bayesian treatments of deterministic kernel methods have been developed, but only a subset of most relevant approaches are discussed here. (4; 5; 6; 7; 8) use Gaussian process priors to SVM classification models. For other basis function models that have been fitted in Bayesian framework via Gaussian processes see (3; 9; 10; 11).

The Relevance Vector Machine (RVM) (12) is an alternative Bayesian formulation of SVM, developed for both classification and regression with the aim of obtaining a sparse solution. The sparseness is induced in the model through the prior structure; see (13) for an in-depth discussion on the sparsity in RVM. Following the work of (14), (12) re-cast the SVM as regularization problem where the aim is to minimize a loss function L subject to a penalty term over a set of regression coefficients β:

$$\min_{\beta} \left[L(\mathbf{y}, \mathbf{K}\beta) + \tau \beta^T \mathbf{K} \beta \right]. \tag{1}$$

The model function is a linear combination of the reproducing kernels and is in the dual form. (12) use a binary logistic likelihood to model loss and assume a relatively standard prior structure for regression coefficients. (15) proposed a similar model to the RVM, but uses a probit likelihood for binary classification and places double exponential priors on regression coefficients, which are known to promote sparseness (16; 17). Note that the RVM model can be viewed as an implicit formulation of the Gaussian process, where the prior is a Gaussian process over then model functions f expressed in the primal form, i.e. as a (possibly infinite) linear combination of the feature space bases. For a more detailed discussion see (11).

The approach of (15) obtains MAP estimates for the model parameters via expectation maximization algorithm. The RVM (12) employs the empirical Bayes approach. (18) adopt the same model construction and prior structure as the RVM, however, rather than estimating the hyperparameters, they assign distributions to them and employ an MCMC sampling algorithm. In addition to the binary logistic likelihood, (18) also consider a stochastic version of the SVM likelihood. The practical advantage of the full probabilistic

approach is that probability distributions of prediction can be obtained for new observations, which gives a more complete picture of classification. By assigning priors to the hyperparameters, the binary classifier of (18) accounts for the uncertainty due to their estimation. A practical disadvantage of such an approach is the relative slow convergence rate caused by the block updating of regression parameters which requires computations involving matrices of dimension $n \times n$, where n is the number of training samples, at each iteration of the MCMC algorithm. In addition, due to over-parameterization, the algorithms exhibit poor mixing even though good classification rates are reported.

In this paper we extend the Bayesian classification model of (18) to a multi-category kernel classifier. A related approach is the Bayesian Multi-category Support Vector Machines (MSVM) (19) who also follow the model construction and choice of prior architecture of (18), however, they use a stochastic multi-class hinge loss function. The disadvantage of this model is that the MSVM likelihood may involve some regression parameters in the normalizing constant which requires a more complex parameterization than the multinomial logistic regression model.

The central contribution of the paper is the proposal of the Bayesian kernel projection classifier, where, by reducing the number of principal axes from the analysis, the dimensionality of the pseudo-design matrix is reduced, which makes BKPC a more efficient algorithm than the above mentioned approaches. In addition, we observed that the sparser set of uncorrelated principal axes contained sufficient class discrimination information so that improved classification rates were obtained for the data sets considered.

The paper is organized as follows; Sect. 2 describes a Bayesian multi-category kernel classifier (BMKC) which is a multi-category extension of the model of (18). The likelihood is modeled through the multinomial logistic regression model and the relatively standard hierarchical prior structure for Bayesian generalized linear models is assumed. The Gibbs sampler is implemented to find the posterior distributions of the parameters and the practical aspects of this implementation are discussed. Bayesian Kernel Projection Classifier (BKPC) is presented in Sect. 3. The sparsity from the projection step and the implementation advantages of this algorithm are outlined. Sect. 4 gives a brief description of the data sets used. The classification results are presented in Sect. 5 and the concluding remarks are given in Sect. 6.

2 Bayesian Multi-category Kernel Classifier (BMKC)

The training data are n samples $(\mathbf{x}_1, \mathbf{y}_1)$,...,$(\mathbf{x}_n, \mathbf{y}_n)$ where the predictors $\mathbf{x}_i = (x_{i1}, ..., x_{iJ})$ are real valued J- dimensional vectors of feature values and $\mathbf{y}_i = (y_{i1}, ..., y_{iK})$ are K-dimensional categorical response variables with $y_{ik} = 1$ if $\mathbf{x}_i$ belongs to a class k and 0 otherwise. A standard approach to this classification problem is the multinomial logistic regression model given by:

$$\mathbb{P}(\mathbf{y}|\mathbf{z}) = \prod_{i=1}^{n} \prod_{k=1}^{K} \mathbb{P}(y_{ik} = 1|z_{ik})^{y_{ik}}, \tag{2}$$

where $\mathbb{P}(y_{ik} = 1|z_{ik})$ is defined as:

$$\mathbb{P}(y_k = 1|\mathbf{x}) = \frac{\exp(z_k)}{\sum_{l=1}^{K} \exp(z_l)}, \tag{3}$$

and z_{ik} are linear combinations of the kernel functions:

$$z_{ik}(\mathbf{x}_i, \beta_k, \theta) = \sum_{l=1}^{n} \beta_{kl} K(\mathbf{x}_i, \mathbf{x}_l|\theta) = \mathbf{K}_i \beta_k, \tag{4}$$

for $i = 1, ..., n$ where β_k are regression parameters $\beta_k = [\beta_{1k}, \beta_{2k}, ..., \beta_{nk}]$ corresponding to class k, for $k = 1, ..., K-1$. $\mathbf{K}_i$ is the i^{th} row of matrix $\mathbf{K}$. In this application only Gaussian kernels are considered:

$$K(\mathbf{x}_i, \mathbf{x}_l|\theta) = \exp(-\theta||x_i - x_l||^2). \tag{5}$$

This is essentially the multi-category extension of the binary kernel classifier presented by (18).

2.1 Prior Specification

In a Bayesian inference approach, priors are assigned to the model parameters. The prior model is specified as:

$$\begin{aligned} z_{ik} &\sim N(\mathbf{K}_i \beta_k, \sigma^2), \\ \beta_k &\sim MVN(0, \sigma^2 \mathbf{T}_k^{-1}), \\ \sigma^2 &\sim IG(\gamma_1, \gamma_2), \\ \tau_{ik} &\sim G(\gamma_3, \gamma_4). \end{aligned}$$

$\mathbf{T}_k$ is a matrix with diagonal entries $\tau_{1k}, ..., \tau_{nk}$. G denotes a gamma prior, IG an inverse gamma and MVN is a multivariate normal of dimension n.

Note that this is a relatively standard hierarchical prior structure for generalized linear models and is used by (18) for binary classification. In order to improve the mixing and convergence of the MCMC algorithm, the latent variables are given a normal prior with means $\mathbf{K}_i \beta_k$ and standard deviation σ^2. This allows for direct block updating of regression coefficients from the joint conditional density (18; 20; 21).

2.2 Inference

A Metropolis-within-Gibbs algorithm was used for sampling from the posterior. The output from the MCMC is a set of samples ($\beta^{(m)}$, $\mathbf{z}^{(m)}$, $\sigma^{2\,(m)}$, $\tau^{(m)}$), for $m = 1, ..., M$ iterations, obtained from the joint posterior distribution after a period of 'burn-in' iterations. The joint posterior distribution is given by:

$$\begin{aligned} \mathbb{P}(\beta, \mathbf{z}, \tau, \sigma^2|\mathbf{y}) &\propto \mathbb{P}(\mathbf{y}|\mathbf{z}, \beta, \tau, \sigma^2)\mathbb{P}(\mathbf{z}|\beta, \sigma^2) \\ &\times \mathbb{P}(\beta|\tau, \sigma^2)\mathbb{P}(\tau)\mathbb{P}(\sigma^2). \end{aligned} \tag{6}$$

The full conditional distributions that were sampled from for each parameter in the model are given in Appendix A.

The MCMC algorithm is implemented so that it iterates through block updates of the parameters starting with $\mathbf{z}$. Each $\mathbf{z}_i = [z_{i1}...z_{i(K-1)}]$ is proposed to be updated conditionally on the rest of the parameters including the matrix $\mathbf{z}$ without the ith element. The proposal density for $\mathbf{z}_i$ is a random walk and is sampled using a Metropolis step within the Gibbs algorithm. Subsequently, parameters β, σ^2 and τ are block updated directly from their conditionals via Gibbs steps.

2.3 Practical Aspects of Implementation

The MCMC algorithm was implemented in the C programming language. The most time consuming aspect involves inverting square matrices with dimension equal to the number of observations. The fact that the matrices are symmetric can be exploited to make the computation easier by using Cholesky decomposition, which runs in time proportional to n^3, see (22) or (23). The Cholesky decomposition of matrix $\mathbf{V}_k = \mathbf{L}\mathbf{L}^T$ is used to compute the determinant of $\mathbf{V}_k$, which is the square of the product of the diagonal elements of $\mathbf{L}$ and to generate vector valued samples from $MVN_{(n)}(\mathbf{m}_k, \sigma^2\mathbf{V}_k)$. If ε is a vector of components that are i.i.d. $N(0,1)$ then $\beta_k^{(m)} = \mathbf{m}_k + \sigma\mathbf{L}\varepsilon$.

2.4 Prediction

A new observation $\mathbf{x}^*$ is classified in class $k^* = \arg\max_k \mathbb{P}(k|\mathbf{x}^*, \mathbf{x}, \mathbf{y})$. This is given by the usual Monte Carlo integration approximations:

$$\mathbb{P}(k|\mathbf{x}^*, \mathbf{x}, \mathbf{y}) \approx \frac{1}{M}\sum_{m=1}^{M} \frac{\exp(\mathbf{K}^*\beta_k^{(m)})}{1 + \sum_{l=1}^{K-1}\exp(\mathbf{K}^*\beta_l^{(m)})}, \tag{7}$$

$\forall k = 1, ..., K-1$, where $\mathbf{K}^* = [K(\mathbf{x}^*, \mathbf{x}_1|\theta), K(\mathbf{x}^*, \mathbf{x}_2|\theta), ..., K(\mathbf{x}^*, \mathbf{x}_n|\theta)]$ and $\beta_K^{(m)} = 0$.

2.5 Modeling the Latent Spatial Dependence

The response data has a latent spatial dependence, since neighbouring features are likely to have the same class label. In the multinomial regression model used, the responses are independent given the latent variables, see equation (2). However, this dependence is implicitly modelled, as the latent variables are spatially dependent on each other and have a spatial structure through the kernel. Intuitively, for a two-dimensional data set, one can think of the linear combination of the kernel functions as a nonlinear generalization of a plane. This two-dimensional spline surface is essentially the kernel density, weighted by the regression coefficients β_l. The regression coefficients β_l place more weights on the kernels of the corresponding sample points $\mathbf{x}_l$ from the training data $\mathbf{x}_l \in \mathbb{R}^2$, $\forall l = 1, ..., n$. The implied spatial dependence of the classification probability obtained for a two dimensional, two class, Ripley's synthetic data (24) can be seen in Figure 1. The classification probability $\mathbb{P}(y = 1|\mathbf{x}^*, \mathbf{x}, \mathbf{y}, \beta^{(m)})$ can be obtained for some parameter realizations $\beta^{(m)}$ from the full posterior distribution and can be evaluated for a particular state $\mathbf{x}^* \in \mathbb{R}^2$. The plot in Figure 1 shows the classification probability surface across the domain of the training data.

2.6 Advantages of the Full Probabilistic Approach

In Ripley's synthetic data (24) each class is set to be a mixture of two Gaussians with the optimal error rate of 0.08. There are 200 training and 1,000 testing samples. The proposed classifier allows for posterior distributions to be obtained through simulation, as opposed to just MAP estimates, which gives a more complete picture of classification. Thus, for each new observation $\mathbf{x}^*$, the probability

$$\mathbb{P}(k|\mathbf{x}^*, \mathbf{x}, \mathbf{y}, \beta^{(m)}) = \frac{\exp(\mathbf{K}_*\beta_k^{(m)})}{1 + \sum_{l=1}^{K-1}\exp(\mathbf{K}_*\beta_l^{(m)})} \tag{8}$$

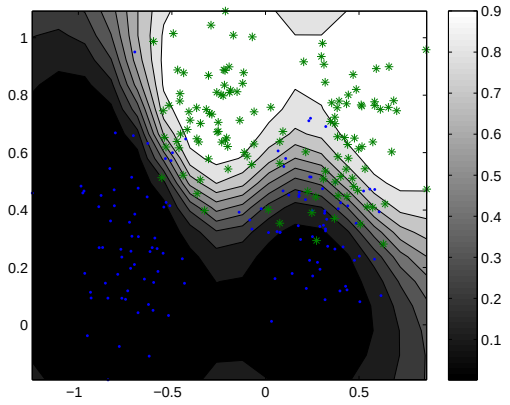

Figure 1: The marginal probability of classification for the domain of the data for Ripley's two-dimensional synthetic data set.

is calculated for each class $k = 1, ..., K-1$ for sets of samples $\beta^{(m)}$ from $m = 1, ..., M$ iterations of the parameters from the joint posterior taken after a 'burn-in' number of iterations. The MAP estimate can be obtained from the Monte Carlo Integration. Figure 2 displays histograms of realizations from the posterior distributions $\mathbb{P}(y = 1|\mathbf{x}^*, \mathbf{x}, \mathbf{y}, \beta^{(m)})$ of predictions for four test observations from Ripley's synthetic data set. Note that this information can be particularly useful for examining borderline observations.

The result of a classification of this two-dimensional data set can be graphically displayed. The multinomial regression model obtains a classification probability surface across the domain of the training data. However, the full probabilistic approach results in a set of realizations of the classification probability surfaces from the posterior density. From these realizations, it is possible to estimate the MAP classification probability surface and information about the certainty of this estimate is available. Whereas it is difficult to plot a set of overlaid surfaces $\mathbb{P}(y = 1|\mathbf{x}^*, \mathbf{x}, \mathbf{y}, \beta^{(m)})$, for some samples $m \in \{1, ..., M\}$, Figure 3 shows the classification boundary, i.e. $\mathbb{P}(y = 1|\mathbf{x}^*, \mathbf{x}, \mathbf{y}, \beta^{(m)}) = 0.5$ obtained for 25 samples of β from the posterior and the mean boundary curve.

The classification results of the BMKC are good, obtained was the error rate of 0.098 which is comparable to results reported by (12), and (16) who obtain 0.093 and 0.095 respectively.

3 Bayesian Kernel Projection Classifier (BKPC)

In this section, the Bayesian Kernel Projection Classifier (BKPC) is proposed. This is a modification to BMKC, but instead of working with the data mapped to some feature space via $\Phi(\mathbf{x})$, the classification is performed in the space spanned by the principal axes of the feature space. The general hope of this approach is that the underlying structure of the feature space can be adequately summarized by a small subset of the principal axes. The mapping of the data and the eigen-decomposition of the covariance matrix $Cov(\Phi(\mathbf{x}))$ is carried out implicitly via the kernel matrix. This is also the mechanism behind the Kernel Principal Components Analysis (KPCA) of (25) and the data projections to the principal axes are the kernel principal components.

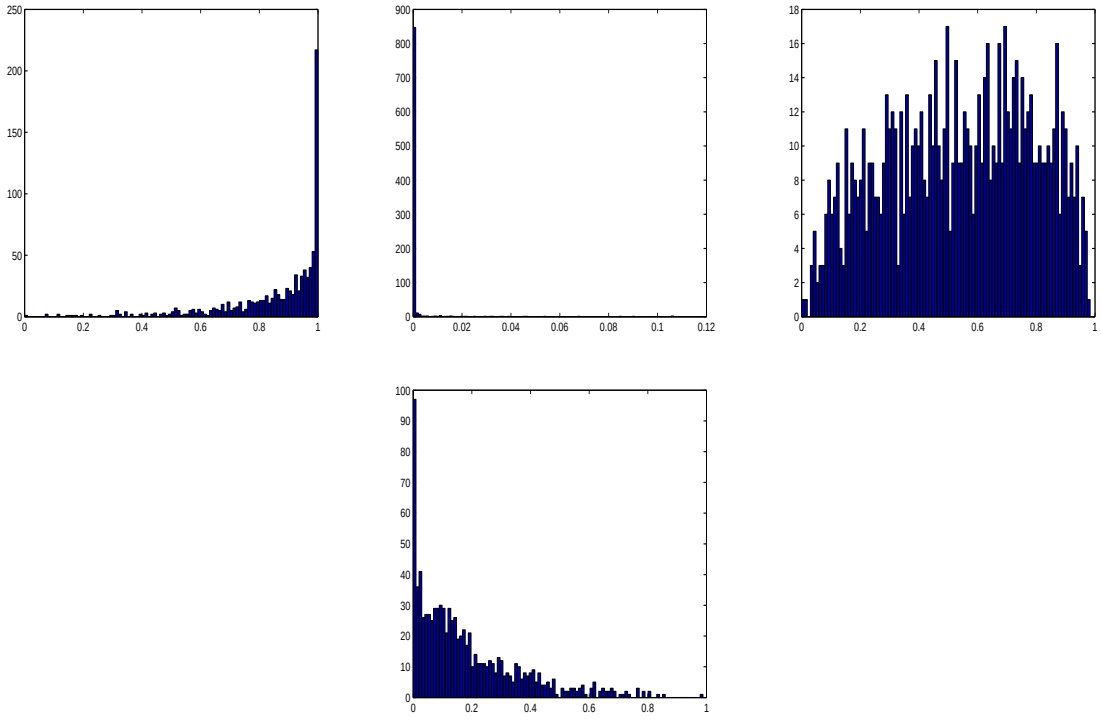

Figure 2: Histograms of realizations from the posterior distribution of predictions $\mathbb{P}(y = 1|\mathbf{x}^*, \mathbf{x}, \mathbf{y}, \beta^{(m)})$ calculated at some $m \in \{1, ..., M\}$ for four observations from Ripley's test data set. The range of values $\mathbb{P}(y = 1|\mathbf{x}^*, \mathbf{x}, \mathbf{y}, \beta^{(m)})$ can take is between 0 and 1. The first MAP estimate for the first observations will place it in class 1, the second observation will be placed in class 2 etc.

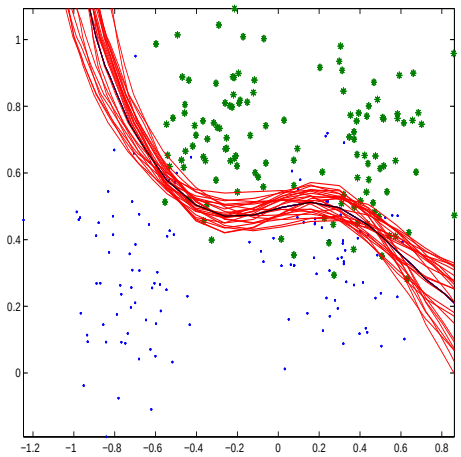

Figure 3: Twenty-five classification boundaries from the posterior distribution, including the posterior mean boundary for the two-dimensional synthetic data set.

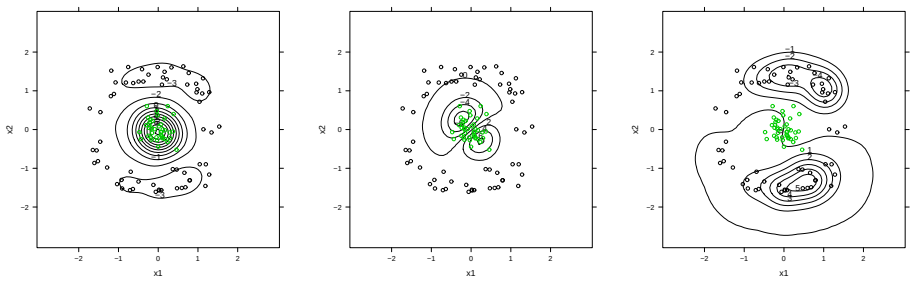

Figure 4: A simulated dataset with the lines of constant principal component value for the first three eigenvectors (given from left to right). A Gaussian kernel with bandwidth $\theta = 5$ was used.

KPCA maps the data $\mathbf{x}_i \in \mathbb{R}^J$ into a high dimensional feature space and then projects the mapped data $\Phi(\mathbf{x})$ to a subspace of the feature space. In the KPCA literature, the vector $\mathbf{x}_i$ is often referred to as the *pre-image* of $\Phi(\mathbf{x}_i)$. Note that, typically, the KPCA subspace will not have a pre-image in the input space. Techniques have been proposed for finding approximate pre-images of data projected on a subset of the eigenvectors, see for example (26; 27).

(25) and (28) note that the first few eigenvectors of the KPCA can be used for separating clusters in two dimensional data, see, for example, the simulated data in Figure 4. They suggest extracting nonlinear principal components and then training a support vector machine, thus constructing a multi-layer SVM. The multi-layer formulation evades pre-image reconstruction, but the evident disadvantage of this algorithm is loss of interpretability as the data are mapped to a feature space twice.

The Bayesian Kernel Projection Classifier is a somewhat different approach to using KPCA to aid classification. It follows the model construction of BMKC, however, the kernel matrix $\mathbf{K}$ is replaced with the matrix of kernel principal components:

$$\underline{\mathbf{K}} = (n\Lambda)^{-1/2}\tilde{\mathbf{K}}\mathbf{U}, \tag{9}$$

where $\tilde{\mathbf{K}}$ is a kernel matrix of the 'centered' mapping, given by:

$$\tilde{\mathbf{K}} = \mathbf{K} - \mathbf{A}\mathbf{K} - \mathbf{K}\mathbf{A} + \mathbf{A}\mathbf{K}\mathbf{A} \tag{10}$$

where $\mathbf{A}$ is a $n \times n$ matrix with all entries equal to $1/n$ (25) and $\mathbf{U}$ and $n\Lambda$ are matrices of eigenvector and eigenvalues obtained from:

$$\tilde{\mathbf{K}} = \mathbf{U}n\Lambda\mathbf{U}^T. \tag{11}$$

Thus, the latent variables z_{ik} become:

$$z_{ik} \sim N(\underline{\mathbf{K}}_i\beta_k, \sigma^2), \quad i = 1, ..., n, \tag{12}$$

where $\underline{\mathbf{K}}_i$ is the i^{th} row of matrix $\underline{\mathbf{K}}$.

Consider the two dimensional, two class 'circle data' from Figure 4. BMKC from Sect. 2 fits a logistic regression model to the data mapped to a high dimensional feature space. By application of the kernel trick, the algorithm is working in the small subspace of the full feature space, which is spanned by the reproducing kernels $\{K(\mathbf{x}_i, \cdot|\theta)\}_{i=1}^n$ and whose dimension is at most equal to the number of observations. The first nine

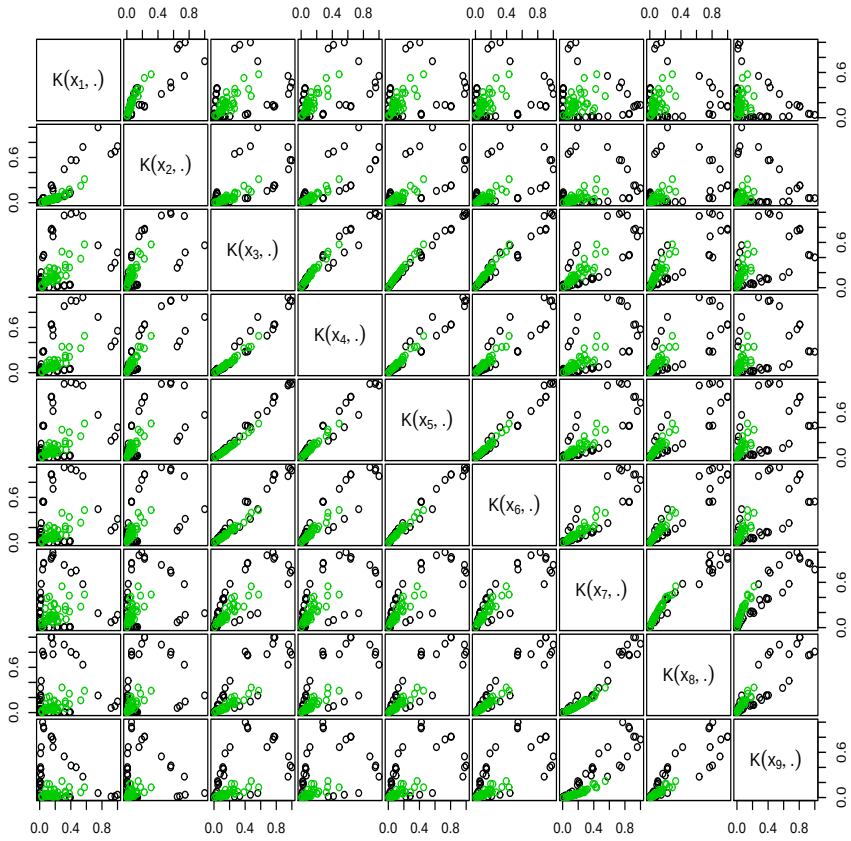

Figure 5: The 'circle' data are mapped to the feature space spanned by reproducing kernels. Only the first 9 reproducing kernels are plotted.

reproducing kernels of the 'circle data' are plotted in Figure 5. The graph shows that the reproducing kernels are highly correlated and only a subset is needed for a good classification model. The Bayesian Kernel Projection Classifier, however, fits the logistic model to the projections of the data to the principal axes of the feature space. Figure 6 shows the first three bases of the KPCA subspace for the 'circle data'.

Note that the proposed model does not require pre-image calculations as the classification is performed in the same feature space as the PCA. This is the main difference between the BKPC and the multilayer formulations suggested by (25) and (28).

3.1 Sparsity from the Projection Step

A kernel classifier with a *sparse* solution depends only on the kernel function evaluated at a subset of training points. An example is the Support Vector Machine where only a set of support vectors is retained in order to make predictions for new inputs. For the Bayesian kernel classifier described in Sect. 2, sparsity can be achieved by setting a subset of the regression parameters to zero, i.e. setting $\beta_{lk} = 0$ for $l = n', ..., n$, $k = 1, ..., K-1$. Obvious advantages of a solution with a fewer number of basis functions are that it is easier to interpret and will generalize better.

For highly correlated mapped data, the diagonalization of the kernel matrix will yield many eigenvalues $n\lambda_l$ equal to zero. In that case, the corresponding principal axes can be

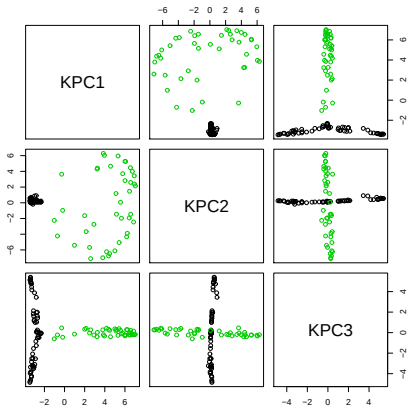

Figure 6: First three KPCs of circular simulated data are plotted. Note that the first eigenvector separates the two classes of observations.

removed from the analysis For example, Figure 6 indicates that only the first principal axis is needed in order to achieve linear discrimination in the mapped space for the 'circle data'. In the BKPC, setting parameters $\beta_l = 0$ removes the l^{th} KPCA basis from the model.

The degree of 'sparseness' can be regulated through a threshold parameter t, that is specified prior to running the algorithm. The largest n' eigenvalues $n\lambda_l$ and their corresponding eigenvectors are included in the model, where n' is chosen so that it is the smallest number that satisfies the condition below:

$$\sum_{l=1}^{n'} \frac{\lambda_l}{\sum_{i=1}^{n} \lambda_i} \geq t, \qquad 0 < t \leq 1. \tag{13}$$

Thus the parameters included in the model are σ^2, z_{ik},τ_{kI} and β_{kI}, $\forall k = 1, ..., K-1$ and $I = \{l = 1, ..., n'\}$.

3.2 Inference for Sparse Model

Consider a sparse model where some regression parameters β_l are set equal to 0. Let $I = \{l = 1, ..., n' | \beta_l \neq 0\}$ and $\bar{I} = \{l = n', ..., n | \beta_l = 0\}$. The conditional distribution for $\beta_I | \beta_{\bar{I}} = 0$ is given by:

$$\mathbb{P}(\beta_I | \beta_{\bar{I}} = 0, \mathbf{z}, \tau, \sigma^2) = \prod_{k=1}^{K-1} MVN_{(n')}(\tilde{\mathbf{m}}_k, \sigma^2 \tilde{\mathbf{V}}_k), \tag{14}$$

where $\tilde{\mathbf{m}}_k = \mathbf{m}_{kI} - \mathbf{V}_{k2}\mathbf{V}_{k4}^{-1}\mathbf{m}_{k\bar{I}}$ is of dimension $n' \times 1$, $\tilde{\mathbf{V}}_k = (\mathbf{V}_{k1} - \mathbf{V}_{k2}\mathbf{V}_{k4}^{-1}\mathbf{V}_{k3})$, is of dimension $n' \times n$. Note that $\mathbf{m}_{kI}$ and $\mathbf{m}_{k\bar{I}}$ are block components of

$\mathbf{m}_k = \begin{pmatrix} \mathbf{m}_{kI} \\ \mathbf{m}_{k\bar{I}} \end{pmatrix}$ with sizes $\begin{pmatrix} n' \times 1 \\ (n - n') \times 1 \end{pmatrix}$

and $\mathbf{V}_{k1}$, $\mathbf{V}_{k2}$, $\mathbf{V}_{k3}$ and $\mathbf{V}_{k4}$ are block components of

$\mathbf{V}_k = \begin{pmatrix} \mathbf{V}_{k1} & \mathbf{V}_{k2} \\ \mathbf{V}_{k3} & \mathbf{V}_{k4} \end{pmatrix}$ with sizes $\begin{pmatrix} n' \times n' & n' \times (n - n') \\ n' \times (n - n') & (n - n') \times (n - n') \end{pmatrix}$, where $\mathbf{m}_k = \mathbf{V}_k \underline{\mathbf{K}}^T \mathbf{z}_k$ and $\mathbf{V}_k = (\underline{\mathbf{K}}^T \underline{\mathbf{K}} + \mathbf{T}_k)^{-1}$

The conditional distributions of the other model parameters are given in Appendix B.

3.3 Implementation Issues in Sparse Classifiers

Implementation of the BKPC algorithm involves spectral decomposition of $\tilde{\mathbf{K}}$, the kernel matrix of the 'centered' mapping, prior to the MCMC run. The parameter updates require calculating $\mathbf{m}_k = \mathbf{V}_k \underline{\mathbf{K}}^T \mathbf{z}_k$ and $\mathbf{V}_k = (\underline{\mathbf{K}}^T \underline{\mathbf{K}} + \mathbf{T}_k)^{-1}$, and subsequently decomposing them to block components in order to get: $\tilde{\mathbf{m}}_k = \mathbf{m}_{kI} - \mathbf{V}_{k2}\mathbf{V}_{k4}^{-1}\mathbf{m}_{k\bar{I}}$ and $\tilde{\mathbf{V}}_k = (\mathbf{V}_{k1} - \mathbf{V}_{k2}\mathbf{V}_{k4}^{-1}\mathbf{V}_{k3})$.

Let:

$\underline{\mathbf{K}} = \begin{pmatrix} \mathbf{K}_1 & \mathbf{K}_2 \\ \mathbf{K}_3 & \mathbf{K}_4 \end{pmatrix}$ and $\mathbf{T}_k = \begin{pmatrix} \mathbf{T}_{kI} & 0 \\ 0 & \mathbf{T}_{k\bar{I}} \end{pmatrix}$

both with sizes $\begin{pmatrix} n' \times n' & n' \times (n-n') \\ n' \times (n-n') & (n-n') \times (n-n') \end{pmatrix}$.

It can be shown using Shur complement that:

$$(\mathbf{V}_{k1} - \mathbf{V}_{k2}\mathbf{V}_{k4}^{-1}\mathbf{V}_{k3})^{-1} = \mathbf{K}_1^T\mathbf{K}_1 + \mathbf{K}_3^T\mathbf{K}_3 + \mathbf{T}_{kI}, \tag{15}$$

and

$$\begin{aligned} \mathbf{m}_{kI} - \mathbf{V}_{k2}\mathbf{V}_{k4}^{-1}\mathbf{m}_{k\bar{I}} &= (\mathbf{K}_1^T\mathbf{K}_1 + \mathbf{K}_3^T\mathbf{K}_3 + \mathbf{T}_{kI})^{-1} \\ &\times \mathbf{K}_1^T\mathbf{z}_{kI} + \mathbf{K}_3^T\mathbf{z}_{k\bar{I}}, \end{aligned} \tag{16}$$

where $\mathbf{z}_k = \begin{pmatrix} \mathbf{z}_{kI} \\ \mathbf{z}_{k\bar{I}} \end{pmatrix}$ with sizes $\begin{pmatrix} n' \times 1 \\ (n-n') \times 1 \end{pmatrix}$.

Thus it is possible to work with matrices $\underline{\mathbf{K}}$ whose columns corresponding to $\bar{I} = \{l = n', ..., n | \beta_l = 0\}$ are deleted. It follows that Cholesky decomposition and other computationally demanding operations of the proposed algorithm BKPC are only applied to matrices of dimension $n' \times n'$, hence large computational gains can be achieved for very sparse models.

3.4 Prediction

For test points $\mathbf{x}_i^* \in \mathbb{R}^J$, where $i = 1, ..., n^*$, the $n^* \times n$ inner product kernel matrix is given by:

$$\mathbf{K}_{il}^* = K(\mathbf{x}_i^*, \mathbf{x}_l | \theta), \forall i = 1, ..., n^*, \forall l = 1, ..., n. \tag{17}$$

Similar to (10), inner product matrix of the test observations centered in the feature space can be expressed in terms of $\mathbf{K}^*$:

$$\tilde{\mathbf{K}}^* = \mathbf{K}^* - \mathbf{A}^*\mathbf{K} - \mathbf{K}^*\mathbf{A} + \mathbf{A}^*\mathbf{K}\mathbf{A}, \tag{18}$$

where $\mathbf{A}^*$ is a $n^* \times n$ matrix with all entries equal to $1/n$. The new observation is projected on the principal axes of the mapping $\Phi(\mathbf{x}^*)$ by:

$$\underline{\mathbf{K}}_l^* = (n\lambda_l)^{-1/2}\tilde{\mathbf{K}}^*\mathbf{u}_l, \tag{19}$$

where $l = 1, ..., n'$ and $\underline{\mathbf{K}}_l^*$ denotes the l^{th} column of the $n \times n'$ matrix $\underline{\mathbf{K}}^*$. The observation $\mathbf{x}^*$ is classified in class $k^* = \arg\max_k \mathbb{P}(k|\mathbf{x}^*, \mathbf{x}, \mathbf{y})$ by employing the usual Monte Carlo integration approximations:

$$\mathbb{P}(k|\mathbf{x}^*, \mathbf{x}, \mathbf{y}) \approx \frac{1}{M}\sum_{m=1}^{M} \frac{\exp(\underline{\mathbf{K}}^*\beta_k^{(m)})}{1 + \sum_{l=1}^{K-1}\exp(\underline{\mathbf{K}}^*\beta_l^{(m)})} \tag{20}$$

$\forall k = 1, ..., K-1$.

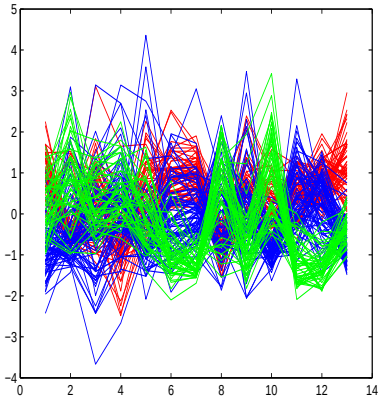

Figure 7: Individual observations in the wine data are plotted and coloured by groups. The graphs shows that there is a systematic difference between the three groups in different parts of the feature space.

4 Application: High Dimensional Data

4.1 Wine Data

The data are the results of a chemical analysis aimed at classifying wines of three different origins. The wines were grown in the same region but come from different cultivars. The analysis determined the quantities of thirteen constituents found in each of the three types of wines. 178 samples were tested. The data set is available from UCI repository of machine learning databases (29). A plot of the data is given in Figure 7.

4.2 Microarray Data

(30) describe gene expression profile data consisting of eighty-three mRNA microarray slides. Each microarray slide corresponds to an individual suffering from one of four tumour types (EWS, BLC, NB and RMS). The total of 2308 genes profiles are reported for each slide. This corresponds to a four category classification problem with a large number of features ($J = 2308$) and small number of observations ($n = 83$). The aim of the analysis is to classify the slides into one of four tumour types on the basis of the gene profiles.

4.3 NIR Spectroscopy Data

The data come from a food authenticity study (31): analysis of spectra of raw homogenized meat samples recorded over the visible and near infra-red wavelength range ($400 - 2498$ at intervals of 2 nm, so recorded are 1050 reflectance values) in order to classify samples into five individual species (chicken, turkey, pork, beef and lamb). A four class problem where chicken and turkey are grouped together into a 'poultry' class is also considered for the purposes of classification. Altogether, there are 1050 features and 231 samples in the study A plot of the data is given in Figure 8. Each meat sample is plotted across the feature space and coloured according to its classification group. The plot shows the most apparent differences between the groups in the visible range of the spectra, which

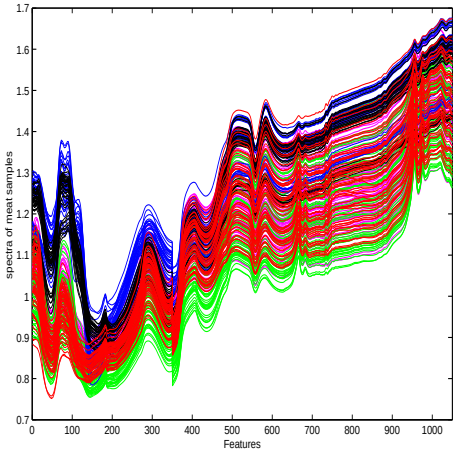

Figure 8: Individual observations in the NIR spectroscopy data are plotted and coloured by groups: blue and black correspond to the red meat, green is pork and magenta and red correspond to poultry. The visible range of the spectra corresponds to the range [0,150] in this graph.

corresponds to the [0,150] section of the feature space. Note that these wavelengths differentiate the colour of the samples so the segregation is between the red and white meat groups.

4.4 Animal Categorization Data

Object recognition is a widely studied problem which has been tackled by a variety of different models. The long term aim of such research is to achieve human levels of recognition accuracy across a large number of object classes in images varying in location, scale, orientation, illumination and subject to occlusions. Animals in natural scenes constitute a challenging problem due to large intra-class variability in terms of shape, texture, size, pose, location in the scene, number of animals etc.

The data set is made up of images that are a subset of the Corel database, which contains 59,795 images of a wide variety of scenes, 8,114 of which are of animals. Four classes of animals were considered: tiger, elephant, goat and lion. 100 images from each class were randomly selected.

The success of the classification depends on the quality of the features summarizing the images. For this task local features which form the 'bag of keypoints' histogram with order of 3013 features were considered. This set of features was obtained by first detecting the areas of high interest in each image and then extracting the colour, texture and structure information from each area. This information is combined into a histogram of frequencies of the occurrence of certain structures in the image. The data was scaled to have equal standard error across the features.

5 Results

The BKPC was used to fit the data sets described in Sect. 4. For all of the data sets, ten random splits into training and testing data were used. In each case, the algorithm was run for 100,000 iterations, of which the first 9,000 were discarded as 'burn-in'.

Table 1: Average misclassification error in the test set obtained from ten random splits of the data sets. Standard deviations are given in brackets. The results are given for runs of the BMKC algorithm proposed in Sect. 2 and the BKPC algorithm described in Sect. 3. The corresponding threshold t setting and the number of largest projections included in the analysis for the given threshold are given. The results are given for runs of the proposed algorithm and multi-category SVM (mSVM) with one-against-one technique and the Gaussian processes (GPs) for classification.

Data set	J	n	Average n'	t	BMKC	BKPC	mSVM	GPs	Better?
Images	3013	200	10.1	0.1	0.37 (0.06)	0.27 (0.03)	0.27 (0.05)	0.37 (0.06)	$\surd$
Microarray	2308	43	3.75	0.2	0.06 (0.04)	0.05 (0.04)	0.14 (0.05)	0.17 (0.08)	$\surd$
NIR (4 groups)	1050	117	7	0.99	0.1 (0.03)	0.05 (0.03)	0.11 (0.03)	0.11 (0.02)	$\surd$
NIR (5 groups)	1050	117	7	0.99	0.24 (0.04)	0.19 (0.04)	0.22 (0.04)	0.23 (0.04)	$\surd$
Wine	13	90	8.1	0.75	0.02 (0.01)	0.02 (0.02)	0.02 (0.01)	0.02 (0.01)	$\surd$

The algorithm requires that the threshold for the percent variance explained by the included KPCs, i.e. t from (13), is set prior to running the algorithm. The optimal t for classification will depend on the nature of the features of each data set and is therefore difficult to preempt. The algorithm was run for t = 0.1,0.2,0.75, 0.9, 0.99 for each data set and the threshold with the best average misclassification rate was reported. The results, along with the threshold value and the corresponding average number of included KPCs is given in Table 1. For comparison, the results of classification with BMKC are also given. In addition, the results of the proposed method were compared with a multi-category SVM with one-against-one technique that fits all the binary sub-classifications and finds the correct class by a voting mechanism implemented in library(e1071) (32) and the Gaussian processes for classification (10) implemented in library(kernlab) (33), R package version 2.6.1 (34).

In the proposed method the empirical estimate $(10/max(\mathbf{K}))$ is used as a starting value for θ. The same estimate is used for the Gaussian kernel bandwidth of the mSVM and the GPs.

The BMKC algorithm proposed in Sect. 2 suffers from over - parameterization, as all of the reproducing kernel basis functions are utilized by the model. By projecting the data onto the principal axes of the feature space, one is hoping that the underlying structure of the feature space can be summarized by a much smaller subset of the principal axes, i.e. so that $n' << n$. After introducing the projection step and reducing a number of components large classification improvements were achieved for the high dimensional NIR spectroscopy and animal image categorization data sets. This makes intuitive sense, as both of these data sets contain many features which are highly correlated. Another advantage is improved data visualization; since BMKC performs the classification in the feature space spanned by reproducing kernels, the number of bases n is usually too large for a matrix plot. However, is possible to visualize a small number of principal component bases of the feature space that the BKPC works in. Figure 9 shows the KPCs, i.e. the ordered columns of matrix $\underline{\underline{\mathbf{K}}}$, with the largest eigenvalues for the NIR spectroscopy data. The first seven KPCs account for 99% of the variation in the data and were used for the classification for both four and five class problem. The graphs shows that the differences within the white meat classes are still difficult to distinguish. The image of the matrix $\underline{\underline{\mathbf{K}}}$ can be seen in Figure 10.

A 'sparser' model of uncorrelated principal components is likely to achieve better

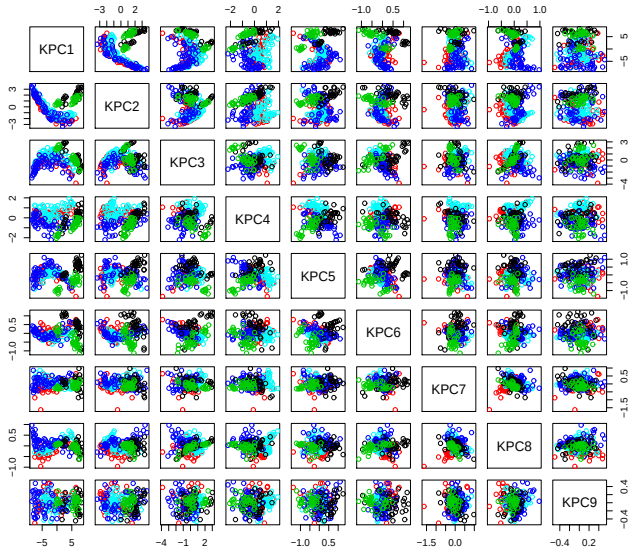

Figure 9: The first nine KPCs for the NIR spectroscopy data. The colours correspond to the meat type (red=chicken, cyan=turkey, blue=pork, black=beef and green=lamb). Only the first seven KPCs were used for the classification.

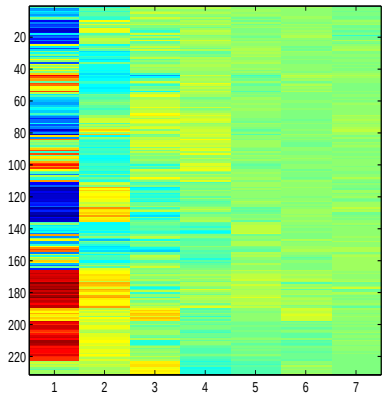

Figure 10: The image of the matrix of projections $\underline{\mathbf{K}}$ is plotted. Only the first seven KPCs were included in the analysis. The sections of the matrix correspond to: $1-55$ chicken, $55-110$ turkey, $110-165$ pork, $166-197$ beef and $198-231$ lamb.

Table 2: Average misclassification error for different starting values of β obtained from running the chain ten times on the same random split of the data set. Standard deviations are given in brackets.

Data set	Misclassific. rate
Images	0.28 (0.0)
Microarray	0.038 (0.052)
NIR (4 groups)	0.04 (0.016)
NIR (5 groups)	0.17 (0.02)
Wine	0.011 (0.0)

mixing and faster convergence. The number of regression coefficients in this model is $7 \times (K-1) = 28$, as opposed to 468 in the full model.

The computational speed gain depends on the data set, as the most computationally demanding operations run in time proportional to n'^3, which, even for the same setting of the threshold t, varies from data to data. However, for illustration purposes, consider the NIR spectroscopy data, where $n = 117$ and only the first 7 principal axes were needed to obtain an improved misclassification rate. The 100,000 iterations of MCMC took 1.92 minutes for $n' = 7$ as opposed to 110 minutes for the full algorithm with $n' = 117$.

Multiple chains for different initial values of parameters were run and the classification algorithm was shown to yield similar misclassification error rates. To examine the impact of Monte Carlo error on correct classification rates, ten chains with different initial values for the regression coefficients were run for the single split of the five data sets into a training and testing data. The regression coefficients were initially either set equal to 1, or were randomly drawn from normal and uniform distributions. Average misclassification rate for the ten runs is given in Table 2. The results are comparable to those obtained by multiple runs of the chain with the same initial values, but with different random splits seen in Table 1. This shows relative insensitivity of the algorithm to the starting values of these parameters and indicates that 'convergence' to a good classification algorithm has been reached.

6 Discussion

A multicategory RHKS-based classification method was proposed. The classifier performed well with high dimensional data sets without any pre-processing steps to reduce the number of variables. The proposed classifier obtains probability distributions of prediction for new data points rather than just MAP estimates as is the case with the approaches of (12) and (15).

The training times of the Bayesian methods are relatively long compared to the deterministic classification methods such as SVMs, however, this is offset by the avoidance of cross validation runs to choose the model complexity parameters.

The proposed classifier is a genuine multi-category extension of the Bayesian binary kernel classifier, which is a more efficient and more principled approach to multi-class classification then the pairwise methods often used in the support vector machines.

We show that the proposed projection step can be used for dimension reduction of the pseudo-design matrix in the Bayesian kernel classifier which was able to achieve large computational time savings at no cost for the prediction accuracy. Furthermore, for the high

dimensional data sets considered, the sparser set of uncorrelated principal axes adequately summarized the underlying structure of the feature space and improved classification rates were observed. The drawback of the approach is that some of the interpretability of the regression coefficients is lost. In the original model, β_l can be thought of as a weight placed on the similarity to observation $\mathbf{x}_l$. Thus, setting $\beta_l = 0$ is equivalent to removing observation $\mathbf{x}_l$ from the training data set for the purposes of class prediction of future data points $\mathbf{x}^*$. In the proposed algorithm, β_l is a weight of a projection of the mapped data to l^{th} principal axis. Hence, setting $\beta_l = 0$ removes a linear combination from the training data, however, all of the observations are retained for the purposes of class prediction.

An important consideration to keep in mind is that the data projections with the largest variance are not necessarily the most useful for class discrimination. More appropriate methods for choosing a sparse subset of β then the selection criteria in (13) could be considered for future research. In the Bayesian setting, sparsity is often approached by choosing appropriate priors for the parameters. In the context of BKPC, one could consider exponential priors for hyperparameters τ_i, as this hierarchical prior construction is equivalent to placing Laplacian priors on the regression coefficients β_i, which is analogous to the Bayesian formulation of the lasso (35). Another possible option is Jeffrey's independence prior $\mathbb{P}(\tau_i) \propto \tau_i^{-1}$ as it was noted in (16) that it has the effect of shrinking some coefficients to zero, however, (18) found that this prior performed worse in terms of misclassification error for their applications.

Future work could also involve exploring other prior structures, for example, in the current construction, both the mean and variance of the latent variables depend on σ^2. Whereas, this is a standard assumption for a normal-gamma model which is widely used for tractability in the posterior model, it would be worth exploring the relaxation of this dependence. Furthermore, the inverse-gamma (γ_1, γ_2) distribution is the most common prior distribution used for variance parameters, but it is well recognized that the inverse-gamma priors can be problematic (36; 37). Instead of the standard (38; 39) uninformative prior $\sigma^2 \sim IG(\gamma_1 = 0.001, \gamma_2 = 0.001)$ on the variance parameter, it is possible to use a truncated prior, or as (37) suggests a proper uniform prior.

Efficient computation of eigen-decomposition was not considered in this section as it is only performed once, while Cholesky decomposition and other matrix calculations are computed at each iteration of the MCMC algorithm, hence make more impact on the overall computation time. Furthermore, eigen-decomposition of a $n \times n$ matrix $\tilde{\mathbf{K}}$ requires that the data sets be relatively small; for large n approximations are needed.

Note that the idea of performing the classification on the projection of the data to the subset of the principal axes in the feature space is can be generalized to other formulations of the kernel classifier.

A Conditional Distributions for Parameters of BMKC

The conditional distributions for the parameters are given by:

$$\mathbb{P}(\tau|\beta) = \prod_{i=1}^{n}\prod_{k=1}^{K-1} G(\gamma_3 + \frac{1}{2}, \gamma_4 + \frac{\beta_{ik}^2}{2\sigma^2}), \tag{21}$$

$$\mathbb{P}(\beta|\mathbf{z}, \theta, \tau, \sigma^2) = \prod_{k=1}^{K-1} MVN_{(n)}(\mathbf{m}_k, \sigma^2\mathbf{V}_k), \tag{22}$$

$$\mathbb{P}(\sigma^2|\beta,\mathbf{z},\theta,\tau) \;=\; IG(\gamma_1+n(K-1),\tilde{\gamma}_2), \tag{23}$$

where $\mathbf{m}_k=\mathbf{V}_k\mathbf{K}^T\mathbf{z}_k$, $\mathbf{V}_k=(\mathbf{K}^T\mathbf{K}+\mathbf{T}_k)^{-1}$ and $\tilde{\gamma_2}=\gamma_2+\frac{1}{2}\sum_{k=1}^{K-1}(\mathbf{z}_k^T\mathbf{z}_k-\mathbf{m}_k^T\mathbf{V}_k^{-1}\mathbf{m}_k)$,

$$\begin{aligned}\mathbb{P}(\mathbf{z}_i|\mathbf{z}_{-i},\mathbf{y},\beta,\theta,\tau,\sigma^2) \propto \exp\Bigg[&\sum_{k=1}^{K-1} y_{ik}z_{ik}- \\ -\;&\log\sum_{k=1}^{K}\exp(z_{ik})-\sum_{k=1}^{K-1}\frac{1}{2\sigma^2}(z_{ik}-\mathbf{K}_i\beta_k)^2\Bigg].\end{aligned} \tag{24}$$

B Conditional Distributions for Parameters of BKPC

The conditional distributions for the parameters are given by:

$$\mathbb{P}(\beta_I|\beta_{\bar{I}}=0,\mathbf{z},\theta,\tau,\sigma^2)=\prod_{k=1}^{K-1} MVN_{(n')}(\tilde{\mathbf{m}}_k^{(m)},\sigma^{2(m)}\tilde{\mathbf{V}}_k^{(m)}), \tag{25}$$

$$\mathbb{P}(\mathbf{z}_i|\mathbf{z}_{-i},\mathbf{y},\beta,\theta,\tau,\sigma^2) \propto \exp\left[\sum_{k=1}^{K-1} y_{ik}z_{ik}-\log\sum_{k=1}^{K}\exp(z_{ik})-\sum_{k=1}^{K-1}\frac{1}{2\sigma^2}(z_{ik}-\underline{\mathbf{K}}_i\beta_{kI})^2\right], \tag{26}$$

$$\mathbb{P}(\sigma^2|\beta,\mathbf{z},\theta,\tau)=IG(\gamma_1+n'(K-1),\gamma_2+\frac{1}{2}\sum_{k=1}^{K-1}(\mathbf{z}_k^T\mathbf{z}_k-\tilde{\mathbf{m}}_k^T\tilde{\mathbf{V}}_k^{-1}\tilde{\mathbf{m}}_k)), \tag{27}$$

$$\mathbb{P}(\tau_I|\beta,\tau_{\bar{I}}=0)=\prod_{l=1}^{n'}\prod_{k=1}^{K-1}\sum_{G}(\gamma_3+\frac{1}{2},\gamma_4+\frac{(\beta_{kl})^2}{2\sigma^2}), \tag{28}$$

where $I=\{l=1,...,n'\}$, $\tilde{\mathbf{V}}_k^{(m)}=(\underline{\mathbf{K}}^T\underline{\mathbf{K}}+\mathbf{T}_{kI}^{(m-1)})^{-1}$ and $\tilde{\mathbf{m}}_k^{(m)}=\tilde{\mathbf{V}}_k^{(m)}\underline{\mathbf{K}}^T\mathbf{z}_k^{(m)}$.

References

[1] M. Aizerman, E. Braverman, L. Rozonoer, Automation and Remote Control **25**, 821 (1964)

[2] B.E. Boser, I.M. Guyon, V.N. Vapnik, in *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, ed. by D. Haussler (ACM Press, Pittsburgh, PA, 1992), pp. 144–152

[3] R.M. Neal, *Bayesian Learning for Neural Networks* (Springer Verlag, New York, 1996)

[4] P. Sollich, Machine Learning **46**(1-3), 21 (2002)

[5] M. Seeger, in *Advances in Neural Information Processing Systems*, vol. 12, ed. by T.L. S.A. Solla, K.R. Müller (MIT Press, 2000), vol. 12, pp. 603–609

[6] M. Opper, O. Winther, in *Advances in Large Margin Classifiers*, ed. by A.J.Smola, P. Bartlett, B. Schölkoph, D. Schuurmans (Cambridge, MA, 2000), pp. 43–65

[7] R. Herbrich, T. Graepel, C. Campbell, Bayesian learning in reproducing kernel hilbert spaces – the usefulness of the bayes point. Tech. Rep. TR-99-11, Technical University Berlin (1999)

[8] J. Kwok, IEEE Transactions on Neural Networks **5**, 1018 (1999)

[9] R.M. Neal, in *Bayesian Statistics 6*, ed. by J.M. Bernardo, et al. (Oxford University Press, 1998), pp. 475–501

[10] C.K.I. Williams, D. Barber, IEEE Transactions on Pattern Analysis and Machine Intelligence **20**(12), 1342 (1998)

[11] C.E. Rasmussen, Evaluation of Gaussian processes and other methods for non-linear regression. Phd, Dept. of Computer Science, University of Toronto (1996)

[12] M.E. Tipping, in *Advances in Neural Information Processing Systems*, vol. 12, ed. by S.A. Solla, T.K. Leen, K.R. Müller (MIT Press, 2000), vol. 12, pp. 652–658

[13] M.E. Tipping, Journal of Machine Learning Research **1**, 211 (2001)

[14] G. Wahba, *Spline models for observational data* (SIAM [Society for Industrial and Applied Mathematics], 1990)

[15] M. Figueiredo, IEEE Trans. Pattern Anal. Machine Intell. **PAMI-25**(9), 1150 (2003)

[16] M. Figueiredo, in *Advances in Neural Information Processing Systems 14*, ed. by T.G. Dietterich, S. Becker, Z. Ghahramani (MIT Press, Cambridge, MA, 2002), pp. 697–704

[17] C.M. Bishop, M.E. Tipping, in *UAI '00: Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence* (Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2000), pp. 46–53

[18] B.K. Mallick, D. Ghosh, M. Ghosh, J. Royal Statistical Soc. B **67**, 219 (2005)

[19] Z. Zhang, M.I. Jordan, in *In Uncertainty in Artificial Intelligence (UAI), Proceedings of the Twenty-Second Conference* (2006)

[20] C. Holmes, L. Held, Bayesian Analysis **1**, 145 (2005)

[21] D. Denison, C. Holmes, B. Mallick, A. Smith, *Bayesian Methods for Nonlinear Classification and Regression* (John Wiley and Sons, Chichester, 2002)

[22] R.A. Thisted, *Elements of Statistical Computing* (Chapman and Hall, New York, 1988)

[23] W.H. Press, B.P. Flannery, S.A. Teukolsky, W.T. Vetterling, *Numerical recipes: the art of scientific computing* (Cambridge University Press, New York, 1986)

[24] B.D. Ripley, *Pattern Recognition and Neural Networks* (Cambridge University Press, Cambridge, 1996)

[25] B. Schölkopf, A. Smola, K.R. Müller, Neural Computation **10**, 1299 (1998)

[26] B. Schölkopf, S. Mika, C.J.C. Burges, et al., IEEE Trans. on Neural Networks **10(5)**, 1000 (1999)

[27] G. Bakir, J. Weston, B. Schölkopf, in *Advances in Neural Information Processing Systems*, vol. 16, ed. by S. Thrun, L. Saul, B. Schölkopf (MIT Press, Cambridge, MA, 2004), vol. 16, pp. 449–456

[28] B. Schölkopf, A. Smola, *Learning with Kernels- Support Vector Machines, Reproducing Kernel Hilbert Spaces , Regularization, Optimization and Beyond* (Cambridge, MA: MIT Press, 2002)

[29] C. Merz, P. Murphy. UCI repository of machine learning databases (1998). URL www.ics.uci.edu/mlearn/MLRepository.html

[30] J. Khan, J.S. Wei, M. Ringnr, L.H. Saal, M. Ladanyi, F. Westermann, F. Berthold, M. Schwab, C.R. Antonescu, C. Peterson, P.S. Meltzer, Nature Medicine **7**(6), 673 (2001)

[31] N. Dean, T. Murphy, G. Downey, J. Roy. Statist. Soc. C **55**(1), 1 (2006)

[32] E. Dimitriadou, K. Hornik, F. Leisch, D. Meyer, A. Weingessel. e1071: Miscellaneous functions of the department of statistics (e1071), TU-Wien, version 1.5-11. (2005). URL http://CRAN.R-project.org/

[33] A. Karatzoglou, A. Smola, K. Hornik, A. Zeileis, Journal of Statistical Software **11** (2004)

[34] R Development Core Team, *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria (2008). URL http://www.R-project.org. ISBN 3-900051-07-0

[35] R. Tibshirani, J. Royal. Statist. Soc. B. **58**, 267 (1996)

[36] P.C. Lambert, Bayesian Analysis **1**(3), 543 (2006)

[37] A. Gelman, Bayesian Analysis **1**(3), 515 (2006)

[38] D.J. Spiegelhalter, A. Thomas, N. Best, W.R. Gilks, *BUGS Examples, Volume 1, Version 0.5* (Cambridge: MRC Biostatistics Unit., 1996)

[39] D.J. Spiegelhalter, A. Thomas, N. Best, W.R. Gilks, *BUGS Examples, Volume 2, Version 0.5* (Cambridge: MRC Biostatistics Unit., 1996)