Feature Extraction for Nonlinear Classification

Anil Kumar Ghosh and Smarajit Bose

Theoretical Statistics and Mathematics Unit,
Indian Statistical Institute,
203, B. T. Road, Calcutta 700108, India
anilkghosh@rediffmail.com, smarajit@isical.ac.in

Abstract. Following the idea of neural networks, multi-layer statistical classifier [3] was designed to capture interactions between measurement variables using nonlinear transformation of additive models. However, unlike neural nets, this statistical method can not readjust the initial features, and as a result it often leads to poor classification when those features are not adequate. This article presents an iterative algorithm based on backfitting which can modify these features dynamically. The resulting method can be viewed as an approach for estimating posterior class probabilities by projection pursuit regression, and the associated model can be interpreted as a generalized version of the neural network and other statistical models.

1 Introduction

In high-dimensional classification problems, often a smaller number of features (linear combination of measurement variables) contain most of the information about class separability. Proper identification of these features helps to reduce the dimensionality of the problem which in turn reduces the computational complexity and storage requirement. If we restrict ourselves to linear classification, results for feature extraction are available in the literature [6], [12], [9]. However in practice, linear classifiers are often found to be inadequate and one needs to extract relevant features for nonlinear classification. In regression, projection pursuit model [7] was developed to compensate for this inadequacy, and artificial neural network models [14] provide much more flexibility for classification. Both of them have their own strengths and limitations. In this article, we try to combine these two approaches in order to develop an iterative feature selection (IFS) method, which is more flexible and statistically meaningful.

2 Nonlinear Classification and IFS

In classification problems, one aims to achieve maximum accuracy in assigning p-dimensional observations $\mathbf{x}$ into one of J competing classes. The optimal Bayes rule [1] assigns an observation to the class with the largest posterior probability. In practice, one estimates these unknown probabilities using the available training data. Many nonparametric classification methods like neural nets [14]

S.K. Pal et al. (Eds.): PReMI 2005, LNCS 3776, pp. 170–175, 2005.

and classification using splines (CUS) [2] put the classification problem in a regression framework and regress the indicator variables $Y_1, \ldots, Y_J$ (defined for J classes) on the measurement space to estimate $E(Y_j \mid \mathbf{x}) = p(j \mid \mathbf{x})$, the posterior probabilities. CUS uses additive models to estimate these posteriors and associated class boundaries. But in practice, these boundaries may be more complex, and one needs to find suitable classification algorithms in such cases. Bose [3] proposed the multi-layer statistical classifier (MSC), which uses the estimated posterior probabilities obtained by CUS as new features and repeats the CUS procedure with them to get the final posterior probability estimates. In the process, some interactions are indirectly introduced in the model which helps to improve the performance of CUS. However, unlike neural nets, MSC can not re-adjust the features selected by CUS. As a result, the improvement over CUS is often not that significant. On the other hand, neural network training algorithms try to re-adjust these features iteratively, but this popular method lacks meaningful statistical interpretations.

This article presents an iterative feature selection (IFS) algorithm, which tries to estimate the best possible features from the data and then performs CUS with the extracted features to estimate the posterior probabilities. Backfitting is used for dynamic feature adjustment, which makes the method more flexible. IFS can be viewed as an approach to estimate the posterior probabilities $p(j \mid \mathbf{x})$ by projection pursuit regression model $\psi_j(\mathbf{x}) = \phi_{j0} + \sum_{i=1}^{p} \phi_{ji}(\boldsymbol{\alpha}_i'\mathbf{x})$, where ϕ_{j0} is a constant, $\boldsymbol{\alpha}_i$'s are feature directions and ϕ_{ji}'s are smooth univariate functions. IFS uses cubic splines as univariate functions ϕ_{ji} and estimates $\boldsymbol{\alpha}_i$, ϕ_{j0} and ϕ_{ji} iteratively. Note that the additive model used in CUS is a special case of IFS when $\boldsymbol{\alpha}_i$'s are unit vectors along the co-ordinate axes. If sigmoidal transformations are used as the smooth univariate functions, this model turns out to be the popular perceptron model with one hidden layer. Since, our aim is to extract low dimensional features for visualizing class separability, instead of starting with too many features, here we restrict this number to the dimension of the original measurement space.

3 Description of IFS Algorithm

Like neural nets, IFS regresses the indicators $Y_1, Y_2, \ldots, Y_J$ on the measurement variables. We take the eigen vectors of $W^{-1}(W+B)$ (where B and W are between class and within class sum of square matrices respectively) as the initial feature directions $\boldsymbol{\alpha}_i$ $(i = 1, 2, \ldots, p)$ and place K knots (based on order statistics) on the range of each feature $Z_i = \boldsymbol{\alpha}_i'\mathbf{x}$ to construct the basis functions. Following the idea of Breiman [4], we use the set of power basis functions $\{1, Z_i, (Z_i - t_k)_+^3, \quad k = 1, 2, \ldots, K; \ i = 1, 2, \ldots, p\}$ for fitting cubic splines. We regress Y_j $(j = 1, 2, \ldots, J)$ on these basis functions to get the initial estimates for $\phi_{j0}, \phi_{j1}, \ldots, \phi_{jp}$. Posterior probability estimates $\widehat{Y}_{j_n} = \psi_j(\mathbf{x}_n)$ and residuals $r_{j_n} = Y_{j_n} - \widehat{Y}_{j_n}$, $n = 1, 2, \ldots, N$ (N being the training sample size) are computed to find the residual sum of squares $(RSS) = \sum_{j=1}^{J} \sum_{n=1}^{N} r_{j_n}^2$.

Backfitting is used to re-adjust the features $Z_1, Z_2, \ldots, Z_p$ iteratively. At any stage $\boldsymbol{\alpha}_i$ is adjusted by a factor $\boldsymbol{\delta}_i$, where $\boldsymbol{\delta}_i$ is obtained by minimizing $RSS \simeq \sum_{j=1}^{J}\sum_{n=1}^{N}[r_{j_n} - \boldsymbol{\delta}_i^{'}\eta_{j_n}^{(i)}]^2$, where $\eta_{j_n}^{(i)} = \frac{\partial \phi_{ji}}{\partial \boldsymbol{\alpha}_i}|_{\mathbf{x}=\mathbf{x}_n}$ is computed using the current estimates of $\boldsymbol{\alpha}_i$ and ϕ_{ji}. This new estimate of $\boldsymbol{\alpha}_i$ is normalized and Z_i is recomputed. Knots are placed on its range to find the new basis functions. $Y_j - \sum_{k\neq i}\phi_{jk}(Z_k)$ is then regressed on these basis functions to get new estimates of ϕ_{j0} and ϕ_{ji} $(j = 1, 2, \ldots, J)$. In this manner, all the p features $(Z_1, Z_2, \ldots, Z_p)$ are updated one by one. This step is repeated until no significant improvement in RSS is observed over some consecutive iterations. At the end, Y_j is regressed on the resulting basis functions to get the final estimates for $\phi_{j0}, \phi_{j1}, \ldots, \phi_{jp}$ $(j = 1, 2, \ldots, J)$. Like CUS, IFS puts no restriction on ψ_j $(j = 1, 2, \ldots, J)$ to ensure that they are in $[0, 1]$. Imposing positivity restrictions by any manner results in much more complicated but not necessarily better classification [13]. However, inclusion of intercept terms ϕ_{j0} $(j = 1, 2, \ldots, J)$ in the set of basis functions guarantees the additivity constraint $(\sum_{j=1}^{J}\psi_j(\mathbf{x}_n) = 1,\ \forall n = 1, 2, \ldots, N)$.

After fitting the initial model using large number of knots, backward deletion is used to delete the knots and hence the basis functions one by one. At any stage, we delete the basis function whose deletion leads to least increase in RSS. However, the linear basis function Z_i $(i = 1, 2, \ldots, p)$ is not considered as a candidate for deletion until all knots on that variable get deleted. This backward deletion process is done by using a modified Gaussian sweep algorithm [4]. It generates a sequence of nested models indexed by their dimensionalities. To arrive at the final and parsimonious model, we adopt the cost complexity criterion proposed in [5], where the ideal cost parameter is selected by 10-fold cross-validation technique [14]. To explain local patterns of the measurement space, the number of initial knots should be reasonably large. However, use of too many knots may make the algorithm numerically unstable. Our empirical experience suggests that $10 - 12$ knots per variable is enough for a moderately large sample size and the final result is not too sensitive on this choice.

4 Experimental Results

For illustration, we present the result of a two-class problem in five dimension, where the first two measurement variables in the two classes are distributed as $\boldsymbol{\Pi}_1 : (X_1, X_2) \sim N_2(0, 0, 1, 1, 1/2)$ and $\boldsymbol{\Pi}_2 : (X_1, X_2) \sim U[-5, 5] \times U[-5, 5]$. Other three variables are i.i.d. $N(0, 1)$ in both classes, and these variables are used to add to the noise. Clearly, the optimal class boundary is of the form $X_1^2 + X_2^2 - X_1X_2 = C_0$ for some constant C_0. This can be re-written (not uniquely) as $A\{(X_1 - 0.5X_2)\}^2 + BX_2^2 = C$, where A, B and C are appropriate constants. Thus, we see that only two linear features contain the information about class separability. When we ran IFS algorithm on different sets of 500 observations generated from these two-classes, the final model contained exactly two features in most of the cases. Figure 1 shows the estimated features and class boundary for one such data set. Clearly, IFS could obtain appropriate estimates of features and that of the optimal class boundary in this case.

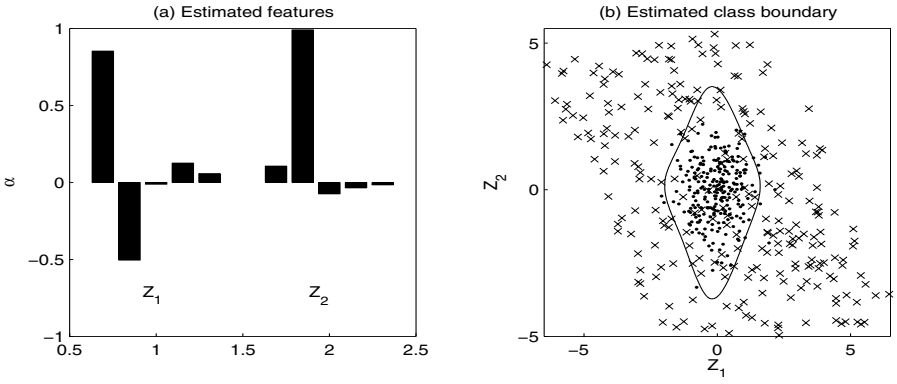

Fig. 1. Estimated features and class boundary

To compare the performance of IFS with other classification techniques like linear discriminant analysis (LDA), quadratic discriminant analysis (QDA), kernel discriminant analysis (KDA) [11], CUS, MSC and neural nets (as implemented in [8]), we report the misclassification rates of these methods for this and two other data sets. For kernel discriminant analysis, we standardize the observations in a class using the usual moment based estimate of the class dispersion matrix to make the data cloud more spherical and then use same bandwidth in all directions to find out the density estimate for the standardized data vector. Density estimate at the original data point can be obtained from that using a simple transformation formula when the measurement variables undergo a linear transformation. Least square cross validation [15] is used to find the bandwidths that minimize the mean integrated square errors of kernel density estimates of different classes and those bandwidths are used for density estimation and classification.

In our second simulated example, each class is an equal mixture of two bivariate normal populations. The location parameters for components of class-1 are $(1, 1)$ and $(3, 3)$, whereas those for class-2 are $(2, 2)$ and $(4, 4)$. All components have the same scatter matrix $0.25\mathbf{I}$. For both of these examples, we used training sets of size 500 and test sets of size 3000 taking equal number of observations from each class. In each case, we repeated the experiment 10 times, and the average test set error rates over these 10 trails are reported in Table-1 along with their corresponding standard errors. Since these are simulated examples, one can also plug-in the true densities of the two populations in the Bayes rule to classify the observations. In Table-1, we also present the average misclassification rates of this optimal Bayes classifier for these two examples with their corresponding standard errors. For the third example, we consider a benchmark data set related to a vowel recognition problem (available at http://www.ics.uci.edu), where we have 10 dimensional observations from 11 different classes. This data set has specific training and test sets, and we report the test set error rates of different methods in Table-1. In Example-2 and Example-3, IFS led to the best performance among the classifiers considered here. Misclassification rates for different

Table 1. Misclassification (in %) rates for different classification methods

	Example-1	Example-2	Example-3
Optimal	10.74 (0.14)	12.08 (0.19)	—
LDA	49.98 (0.33)	45.23 (0.57)	55.63
QDA	11.61 (0.17)	45.44 (0.54)	52.81
KDA	12.43 (0.16)	12.68 (0.23)	62.12
CUS	14.30 (0.27)	18.35 (0.41)	51.30
MSC	14.02 (0.28)	18.54 (0.40)	45.67
Neural Net	12.48 (0.19)	12.58 (0.21)	48.92
IFS	11.88 (0.22)	12.38 (0.18)	43.29

types of kernel methods and other nonparametric classifiers on vowel recognition data are available in [12] and [10]. Results of our proposed method is better than those reported error rates. Since the optimal class boundary in Example-1 is quadratic, QDA is expected to perform well, but IFS could nearly match the performance of QDA.

It is difficult to find out the expression for computation complexity of IFS algorithm. Since IFS is an iterative algorithm, in addition to depending on the sample size, number of classes and number of basis functions, the computing time depends on the closeness of the initial and final solution and the convergence criterion as well. However, as a nonparametric classifier IFS works reasonably fast. When 10 knots are used on each of the 10 variables in vowel recognition data, it took nearly 30 minutes on a pentium 4 machine for model selection and classification. Note that we had to run IFS algorithm 11 times including 10 times for cross-validation. We terminated the iteration in IFS if the relative change in RSS is less than 0.1% over 10 consecutive iterations. Of course, one can relax this convergence criterion to cut down the computing time. The overall performance of IFS turned out to be excellent which should be explored and tested further.

References

1. Anderson, T. W. (1984) *An Introduction to Multivariate Statistical Analysis*. Wiley, New York.
2. Bose, S. (1996) Classification using splines. *Comp. Statist. Data Anal.*, **22**, 505-525.
3. Bose, S. (2003) Multilayer Statistical Classifiers. *Comp. Statist. Data Anal.*, **42**, 685-701.
4. Breiman, L. (1993) Fitting additive models to regression data: diagnostics and alternating views. *Comp. Statist. Data Anal.*, **15**, 13-46.
5. Breiman, L., Friedman, J. H., Olshen, R. A. and Stone, C. J. (1984) *Classification and Regression Trees*. Wadsworth and Brooks Press, Monterrey, California. .
6. Duda, R., Hart, P. and Stork, D. G. (2000) *Pattern classification*. Wiley, New York.
7. Friedman, J. and Stuetzle, W. (1981) Projection Pursuit Regression. *J. Amer. Statist. Assoc.*, **76**, 817-823.
8. Ghosh, A. K. and Bose, S. (2004) Backfitting neural networks. *Computational Statistics*, **19**, 193-210.

9. Ghosh, A. K. and Chaudhuri, P. (2005) Data depth and distribution free discriminant analysis using separating surfaces. *Bernoulli*, **11**, 1-27.
10. Ghosh, A. K., Chaudhuri, P. and Sengupta, D. (2005) Multi-scale kernel discriminant analysis. *Proceedings of Fifth International Conference on Advances in Pattern Recognition (ICAPR-03)* (Ed. D. P. Mukherjee and S. Pal), Allied Publishers, Kolkata, pp. 89-93.
11. Hand, D. J. (1982) *Kernel Discriminant Analysis*. Wiley, Chichester.
12. Hastie, T., Tibshirani, R. and Friedman, J. H. (2001) *The elements of statistical learning : data mining, inference and prediction*. Springer Verlag, New York.
13. Kooperberg, C., Bose, S. and Stone, C. J. (1997) Polychotomus regression. *J. Amer. Statist. Assoc.*, **92**, 117-127.
14. Ripley, B. D. (1996) *Pattern recognition and neural networks*. CUP, Cambridge.
15. Silverman, B. W. (1986) *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, London.