The Korean Communications
in Statistics Vol. 7, No. 2, 2000
pp. 585-596

Prediction and Classification Using Projection Pursuit Regression with Automatic Order Selection[1)]

Heon Jin Park[2)], Daewoo Choi[3)] and Ja-Yong Koo[4)]

Abstract

We developed a macro for prediction and classification using projection pursuit regression based on Friedman (1984b) and Hwang, *et al.* (1994). In the macro, the order of the Hermite functions can be selected automatically. In projection pursuit regression, we compare several smoothing methods such as super smoothing, smoothing with the Hermite functions. Also, classification methods applied to German credit data are compared.

Keyword : backfitting, Hermite polynomial, supersmoother

1. Introduction

Projection pursuit regression proposed by Friedman and Stuetzle (1981) is a statistical learning procedure for multivariate data analysis. This procedure is a tool for finding the most interesting lower dimensional feature of high dimensional data and optimization with respect to this projection direction.

The projection pursuit regression method is based on a generalized form of a linear model as follows.

$$Y_j = \beta_0 + \sum_{m=1}^{M} \beta_m f_m(\alpha_m^T X_j) + e_j, \quad j = 1, \cdots, n, \qquad (1)$$

where Y_j are q dimensional observed response vectors, X_j are p dimensional observed predictor vectors, and $\{e_j\}$ is a noise process. Also, β_0 and β_m are q dimensional unknown vectors and f_m are unknown functions. For model identification, the following restrictions are imposed.

1) The authors thank Seong-Yun Kim for his help in programming.
2) Associate Professor, Department of Statistics, Inha University, Inchon, 402-751, Korea.
E-mail : hjpark@anova.inha.ac.kr
3) Assistant Professor, Hankook University of Foreign Studies, Yongin, 449-791, Korea.
4) Professor, Department of Statistics, Hallym University, Chunchon, 200-702, Korea.

$$E[f_m]=0$$
$$E[f_m^2]=1$$
$$\alpha_m^T\alpha_m=1$$

The model in (1) is a function of projections of X_j, $\alpha_m^T X_j$, instead of predictor variables X_j themselves. Therefore, the number of terms in the right hand side in (1) can be reduced and simplified. As a result, overfitting problem can be avoided. That is, projection pursuit regression is to bypass the "curse of dimensionality" caused by the fact that it becomes difficult to collect enough samples for high-dimensional functions. The *projection* part of the term *projection pursuit* indicates that p-dimensional predictor vector X_j is projected onto direction vectors $\{\alpha_m, 1\leq m\leq M\}$ to get the lengths $\alpha_m^T X_j$ of the projections, for $1\leq m\leq M$, and the *pursuit* part indicates that the optimization technique is used to find *good* direction vectors, α_m, for $1\leq m\leq M$.

In projection pursuit regression, a universal approximator is used for any continuous function. In the universal approximator, for any function $g(X)$ and any positive ε, there exists number of terms, M, such that $||g(X)-\beta_0-\sum_{m=1}^{M}\beta_m f_m(\alpha_m^T X)|| < \varepsilon$ for every X. See DeVore (1991) or Cherkassky and Mulier (1998) for details. A function form of f_m in (1) is not given and the function is predicted in projection pursuit regression, which allows more flexible results than methods with fixed forms of functions such as neural network. In projection pursuit regression, estimate of f_m is given as nonparametric regression or a linear combination with Hermite functions.

Projection pursuit regression can be compared with neural network with one hidden layer. Neural network has same model as (1) except a function form of f_m. While neural network uses a fixed function form such as a logistic function and hyperbolic tangent function, projection pursuit regression uses flexible nonlinear ridge functions such as a nonparametric function or a linear combination of the Hermite functions. Even though both of neural network and projection pursuit regression are difficult to interpret, projection pursuit regression is known to give more flexible and stable result than neural network. (See Donoho and Johnstone (1989) or Zhao and Atkeson (1992) for details.)

We developed a SAS macro for prediction and classification using projection pursuit regression. The SAS macro is programmed with SAS/IML and SAS/Macro. The algorithm in the macro is based on Friedman (1984b), which uses super smoothing technique for function smoothing. Also, the SAS macro uses Hermite functions as a function smoother, which is proposed by Hwang *et al.* (1994). Hwang *et al.* (1994) fixes number of order in Hermite functions, but we designed to select number of order automatically using F-test.

We compare super smoothing, smoothing with Hermite functions with fixed order and automatically selected order. Also, we adapt projection pursuit regression with German credit

data to show a procedure for classification.

2. Algorithm for projection pursuit regression

2.1 Estimation for prediction in projection pursuit regression

Parameter estimation procedure for projection pursuit regression in this section is based on Friedman (1984b). We consider continuous response variables first. Assuming that there are q continuous response variables, parameter estimates for prediction in projection pursuit regression are obtained by minimizing

$$L_2 = \sum_{i=1}^{q} \sum_{j=1}^{n} (Y_{ij} - \beta_{i0} - \sum_{m=1}^{M} \beta_{im} f_m(\alpha_m^T X_j))^2, \qquad (2)$$

where Y_{ij} is the i-th element of observation Y_j. In (2), there are three sets of parameters, coefficient parameters of functions $\{\beta_{im}, 1 \le i \le q,\ 1 \le m \le M\}$, projection directions $\{\alpha_{jm}, 1 \le j \le p,\ 1 \le m \le M\}$, and unknown functions $\{ f_m, 1 \le m \le M\}$ for a given number of terms, M. Since it is impossible to estimate all parameters at a time, parameters are estimated iteratively using backfitting algorithm. The backfitting algorithm in projection pursuit regression is given as follows.

(growing procedure)

(1) Set k to be 0.

(2) Let the estimate of β_{i0} be $\overline{Y_i}$, the i-th element of mean vector of Y_{ij} and $R_{ij}^{(0)} = Y_{ij} - \overline{Y_i}$.

(3) $k = k+1$.

(4) Obtain estimates, $\hat{\beta}_{ik}$, $\hat{f}_k$, and $\hat{\alpha}_k$ from the model with one term

$$R_{ij}^{(k-1)} = \beta_{ik} f_k(\alpha_k^T X_j) + e_{ij}^{(k)},\ i = 1, \cdots, q,\ j = 1, \cdots, n$$

and obtain residuals

$$R_{ij}^{(k)} = R_{ij}^{(k-1)} - \hat{\beta}_{ik}\, \hat{f}_k(\hat{\alpha}_k^T X_j)$$

(5) Repeat (3) and (4) until k reaches the maximum number of terms or decrease of absolute and relative error sum of squares is very small.

(pruning procedure)

(4) For the k-th term, fix other terms and obtain residual

$$R_{ij(-k)} = Y_{ij} - \overline{Y_i} - \sum_{l \ne k} \hat{\beta}_{il}\, \hat{f}_l(\hat{\alpha}_l^T X_j)$$

and obtain the new parameter estimates by fitting model

$$R_{ij(-k)} = \beta_{ik} f_k(\alpha_k^T X_j) + e_{ij(-k)}.$$

For each term, parameter estimates are adjusted iteratively and iteration is repeated

until convergence.

(5) Delete a term with smallest $\hat{\beta}_m$ in absolute value and redo (4).

(6) Repeat (4) and (5) until number of terms reaches the designated number.

In parameter estimation in the model with one term, the estimate for one set is obtained by fixing other two sets. For the m-th term, $\{\beta_{im}, 1 \le i \le q\}$ is estimated with $\{\alpha_{jm}, 1 \le j \le p\}$ and f_m fixed. When α_{jm} and f_m are fixed, the structure for β_{im} is linear and estimates of β_{im} is given by least squares estimation. And, then, $\{\alpha_{jm}, 1 \le j \le p\}$ is obtained with f_m fixed and β_{im} updated for $1 \le i \le q$. For α_{jm}, the object function is nonlinear and, therefore, the estimates are obtained using Gauss-Newton method. Also, with α_{jm} and β_{im} updated, f_m is obtained through smoothing procedure in Section 2.2. These procedures are repeated until convergence. In projection pursuit regression, two convergence criteria are used. One is the convergence criterion of the relative error sum of square and the other is convergence criterion for the absolute error sum of square. The initial values are given internally.

2.2 Smoothing methods - super smoothing and smoothing with Hermite functions

The super smoothing technique proposed by Friedman (1984a) is generalization of the running line smoother. The super smoothing technique is performed as follows. For detail of super smoothing, see Friedman (1984a).

1. Perform the running line smoother with spans 0.02, 0.2 and 0.5 and obtain the predicted values and the cross-validated absolute residuals for each data point and for each span.
2. Perform the running line smoother for the cross-validated absolute residuals with span 0.2. Then, we have a set of smoothed residuals for each span.
3. Choose the span with minimum of smoothed residuals in Step 2 for each data point.
4. Perform the running line smoother for the spans chosen in Step 3 with span 0.2. Then, we obtain a smoothed span for each observation.
5. For the smoothed span for each observation in Step 4, perform interpolation with predicted values and spans in Step 1.
6. Finally, perform the running line smoother for the predicted values in Step 5 with span 0.02.

In the super smoothing, the function form is not obtained and only the predicted value for each observation is generated. It may be uncomfortable, but the super smoother covers more general forms in the model than other smoothing technique. A defect of super smoothing is that a closed form of derivative at each data point can not be obtained. In projection pursuit regression, the derivative at a data point is given as a slope of two nearest data points in the right hand side and left hand side of each data point.

Smoothing technique with Hermite functions is proposed by Hwang *et al.* (1994). The Hermite polynomials are constructed recursively as follows.

$$H_0(z)=1,$$
$$H_1(z)=2z,$$
$$H_r(z)=2(zH_{r-1}(z)-(r-1)H_{r-2}(z)),\quad r=2,3,\cdots.$$

The Hermite polynomials are orthogonal on $(-\infty,\infty)$ with respect to the square of the standard normal density function. That is, for $i\neq j$

$$\int_{-\infty}^{\infty} H_i(z)H_j(z)\phi^2(z)\,dz=0,$$

where

$$\phi(z)=\frac{1}{\sqrt{2\pi}}e^{-z^2/2}.$$

Since

$$\int_{-\infty}^{\infty} H_r^2(z)\phi^2(z)\,dz=r!\,\pi^{-1/2}2^{r-1},$$

we can construct the orthogonal Hermite function

$$h_r(z)=(r!)^{-1/2}\pi^{1/4}2^{-(r-1)/2}H_r(z)\phi(z),$$

so that

$$\int_{-\infty}^{\infty} h_i(z)h_j(z)\,dz=\begin{cases}1 & \text{for } i=j\\ 0 & \text{for } i\neq j\end{cases}.$$

Then, usually the smoothing function based on the Hermite polynomial of order R is defined as

$$f(z)=\sum_{r=0}^{R} c_r h_r(z).$$

The coefficients c_r is estimated by the least squares method.

Hwang *et al.* (1994) uses the same order R in all functions. But, the high order of these functions may not contribute to a specific term. Also, in projection pursuit regression, same order for all terms is not effective. That is, using high order has high risk of overfitting.

To avoid this problem, the order in Hermite functions needs to be adjusted for each term. In the macro we designed, order of Hermite function can be determined using backward elimination with F-test. The coefficient c_r is estimated for maximum number of order and the most insignificant estimate is eliminated until all estimates are significant.

Smoothing with the Hermite functions gives a parametric model and provides smooth interpolation. Also, an accurate derivative is obtained in a smoothing function with the Hermite functions. And the smoothing with the Hermite functions tends to be smoother than super smoothing. Also, the smoothing with the Hermite functions is faster and need less memory space.

2.3 Classification in projection pursuit regression

Suppose that the response variable has q possible categories, $c_1, \cdots, c_q$. Then, for a observed response Y, dummy variables are given as follows.

$$H_i = 1 \text{ if } Y \text{ belongs to category } c_i$$
$$= 0 \text{ otherwise.}$$

For classification projection pursuit regression, dummy variables $H_1, \cdots, H_q$ are used as response variables. Estimates may be obtained in order to minimize the expected loss of misclassification, but this approach produces a non-convex object function in parameter estimation and it is not desirable in practice. See Breiman, Friedman, Olshen and Stone (1983) for details. Instead, classification with projection pursuit regression produces estimates by minimizing

$$\sum_{i=1}^{q} \sum_{j=1}^{n} \left(H_{ij} - \beta_{i0} - \sum_{m=1}^{M} \beta_{im} f_m(\alpha_m^T X_j) \right)^2$$

where H_{ij} are the dummy variables described above for the j-th observation Y_j.
Then the decision rule is to classify an observation into the category where

$$\beta_{i0} + \sum_{m=1}^{M} \beta_{im} f_m(\alpha_m^T X)$$

is maximized over i.

3. SIMULATION AND COMPARISON

We developed a SAS macro for prediction and classification using projection pursuit regression. The SAS macro is programmed with SAS/IML and SAS/Macro. The algorithm is based on Friedman (1984b), which uses super smoothing technique for function smoothing. Also the SAS macro uses the Hermite functions as a function smoother, which is proposed by Hwang *et al.* (1994). In addition to their algorithm, using backward elimination we add automatic order selection in Hermite functions with maximum order given. Also, the SAS macro provides several selection criteria for term selection to help users choose number of terms in model (1).

We compare three smoothing methods in projection pursuit regression - super smoothing proposed by Friedman (1984b), smoothing with Hermite functions with order 7 proposed by Hwang *et al.* (1994) and smoothing with Hermite functions with automatic order selection. These comparisons are made on the learning accuracy that is the fraction of variance unexplained (FVU) of independent test data. Also, we show an example of classification with German credit data.

3.1 Simulation study with nonlinear regression functions

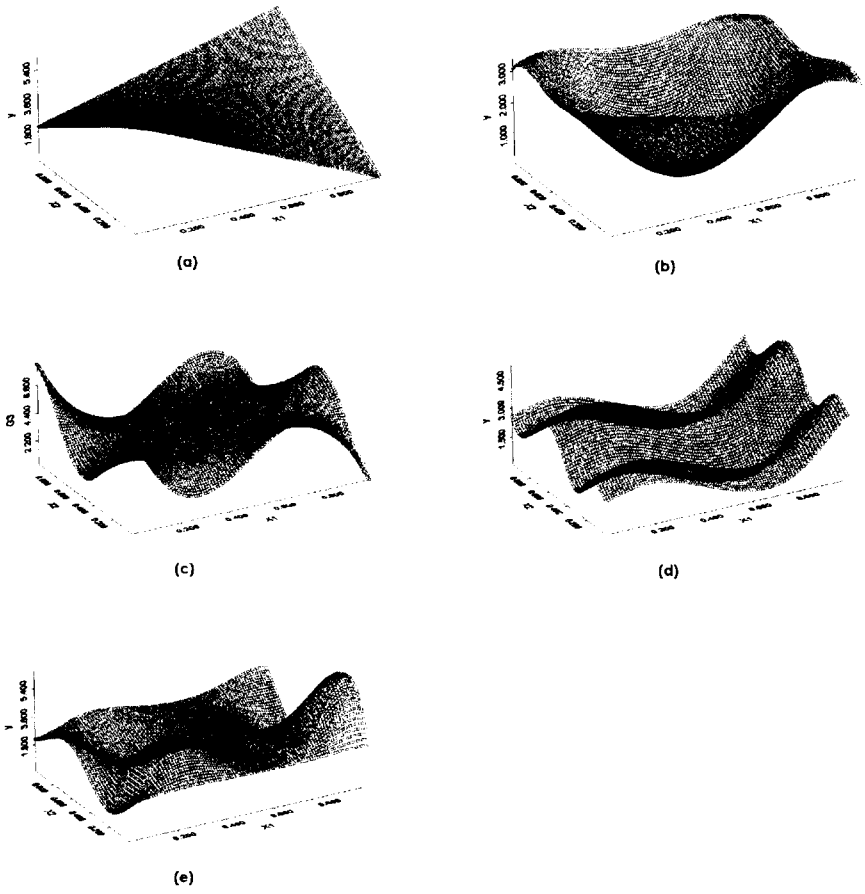

Figure 1. (a) simple function, (b) radial function, (c) harmonic function, (d) additive function, (e) complicated function.

We investigate performance in five nonlinear functions used in Hwang *et al.* (1994). These functions are as follows.

(1) Simple function

$$f(x_1, x_2) = 10.391((x_1 - 0.4) \times (x_2 - 0.6) + 0.36)$$

(2) Radial function

$$f(x_1, x_2) = 24.234(r^2(0.75 - r^2)),\ r^2 = (x_1 - 0.5)^2 + (x_2 - 0.5)^2$$

(3) Harmonic function

$$f(x_1, x_2) = 42.659(0.1 + x_1^*(0.05 + x_1^{*4} - 10\, x_1^{*2} \cdot x_2^{*2} + 5\, x_2^{*4}))$$

$$\text{with}\ \ x_1^* = x_1 - 0.5,\ \ x_2^* = x_2 - 0.5$$

(4) Additive function

$$f(x_1, x_2) = 1.3356(1.5(1-x_1) + e^{2x_1-1}\sin(3\pi(x_1-0.6)^2) + e^{3(x_2-0.5)}\sin(4\pi(x_2-0.9)^2))$$

(5) Complicated function

$$f(x_1, x_2) = 1.9(1.35 + e^{x_1}\sin(13(x_1-0.6)^2) \times e^{-x_2}\sin(7x_2))$$

Shapes of these functions are illustrated in Figure 1.

We generate independent two data sets for each of five functions. The first data set, called training data set, is for model fitting. Two independent predictor variables (x_{1j}, x_{2j}) that are generated from the uniform distribution $U([0,1]^2)$ and the response variable y_j is generated with a noise as follows.

$$y_j = f(x_{1j}, x_{2j}) + 0.25\varepsilon_j.$$

Here $\varepsilon_j \sim N(0,1)$. 225 observations are generated for model fitting.

The second data set, called test data set, is for performance assessment by comparing the fitted values with the true value. The test data set is generated on an equally spaced grid on $[0,1]^2$ and the size of test data set is 10000, i.e. with

$$x_{li} = \frac{(2l-1)}{200}, \quad \text{for } l = 1, \cdots, 100, \quad i = 1, 2.$$

The training data sets were generated 50 times. For each training set, the model was fitted and assessed with the test data set. The same test data set was used for 50 training data set.

We assess each algorithm with 3 and 5 terms. For the models with 3 terms, we increase the number of terms up to 5, and then, decrease the number of terms to 3. We set 7 to be the maximum number of terms for the model with 5 terms. We use the fraction of variance unexplained (FVU) of independent test data for evaluation. FVU is obtained by residual sum of squares divided by corrected sum of squares of the response variables. Means and standard deviations for FVU's for 50 independent training data are given in Table 1.

As shown in Table 1, for the simple function and the additive function, 3 terms provides less FVU in each smoothing method and 5 terms are appropriate for the radial function, harmonic function and complicate function. In comparison of smoothing methods, smoothing with the Hermite functions with order of 7 is better in cases of the simple function, the radial function and the harmonic functions. Super smoothing provides smaller FVU in the additive function and smoothing with the Hermite functions with internal selection of order has smaller FVU for the complicate function. This shows that fixed order in the Hermite function provides better result in relatively simple form of functions. And super smoothing is appropriate for corrugated form of functions. If a function form is complicate, internal selection of the order of the Hermite function looks reasonable.

3.2 Classification in German credit data

		3 terms FVU(S.D)	5 terms FVU(S.D)
simple function	Hermite function (Order=7)	0.012 (0.009)	0.019 (0.028)
	super smoothing	0.152 (0.988)	0.023 (0.039)
	Hermite function (automatic order)	0.019 (0.018)	0.056 (0.071)
radial function	Hermite function (Order=7)	0.029 (0.016)	0.018 (0.029)
	super smoothing	0.040 (0.015)	0.024 (0.008)
	Hermite function (automatic order)	0.027 (0.094)	0.038 (0.094)
harmonic function	Hermite function (Order=7)	0.106 (0.031)	0.019 (0.009)
	super smoothing	0.258 (0.073)	0.191 (0.063)
	Hermite function (automatic order)	0.127 (0.051)	0.037 (0.028)
additive function	Hermite function (Order=7)	0.025 (0.016)	0.033 (0.026)
	super smoothing	0.018 (0.030)	0.022 (0.020)
	Hermite function (automatic order)	0.028 (0.021)	0.045 (0.057)
complicate function	Hermite function (Order=7)	0.121 (0.064)	0.058 (0.027)
	super smoothing	0.120 (0.064)	0.063 (0.022)
	Hermite function (automatic order)	0.106 (0.055)	0.030 (0.014)

Table 1. The average FVU of the test data set

We investigate performance of smoothing methods for German credit data, which was used in StatLog project. German credit data, provided by Professor Dr. Hans Hofmann at University Hamburg, contain 1000 observations and 13 attributes with some categorical attributes.

We added several indicator variables for categorical attributes. This preprocessed data set had 51 numerical attributes, for predictor, and a categorical attribute for response variable with 2 possible categories. The response variable of German credit data is to show that credit is good or bad for each customer. And we call a concerned category of the response variable as a target class. "Bad" category is a target class in German credit data. We use 10-fold

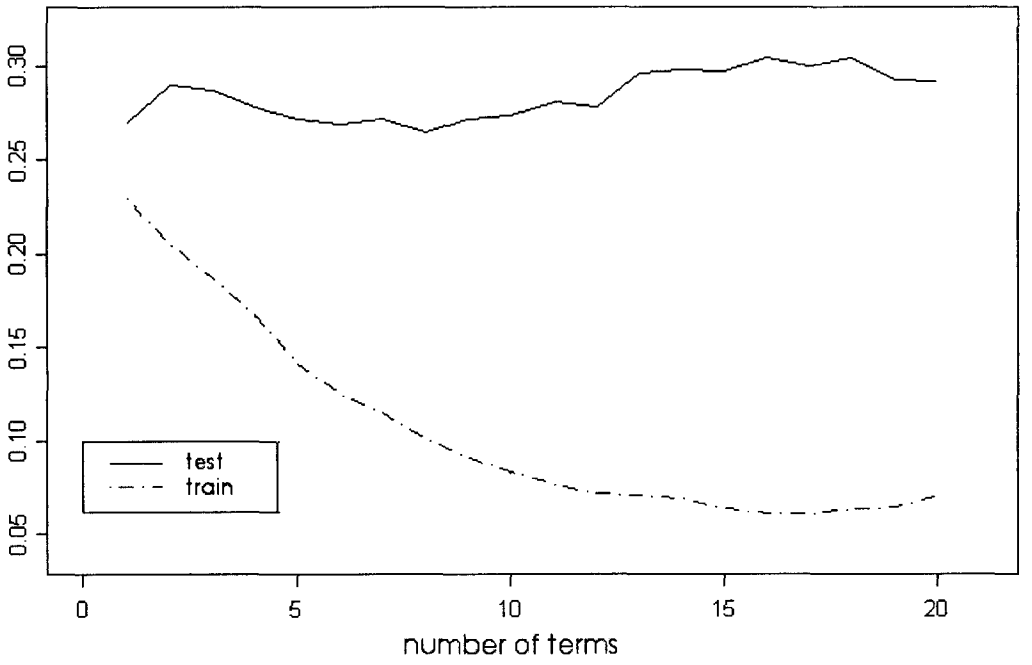

Figure 2. Average misclassification error rate for 10-fold cross-validation

cross-validation for term selection and performance measure. 10-fold cross-validation was used as follows.

1. Randomly divide data into 10 disjoint sets with roughly equal size, $Z_1, Z_2, \cdots, Z_{10}$.
2. For each data set Z_i of size n_i,
   (a) Use the remaining data sets, $Z_t = \bigcup_{j \neq i} Z_j$ for model estimation.
   (b) Apply the estimated model from (a) to data set Z_i and obtain misclassification error rate (MER), the ratio of misclassified observations in Z_i.
3. Compute an average and a standard deviation of 10 MER's and use the average as a measure of performance.
4. With a new observation, apply the new observation to 10 estimated models in 2 and classify the observation to the class of which more models are in favor of.

For classification, we need to decide number of terms in model (1). For choice of number of terms, we apply 10-fold cross validation for the model with number of terms varying 1 to 20. Figure 2 shows average of MER's for data sets of model estimation (train data set) and a data set not in model estimation and used for model performance (test data set). The average of MER's for test data sets is more reasonable for model selection because the test data set is irrelevant to model estimation. In Figure 2, model with 8 terms gives the smallest average of MER's for test data sets and, therefore, we chooses 8 to be the number of terms.

	LDA	super smoothing	Hermite function with order=7	Hermite function with automatic order
Average of MER	0.348	0.302	0.284	0.241
S.D of MER	0.100	0.038	0.046	0.044

Table 2. The average misclassification error rate

We compare four classification methods - linear discriminant analysis (LDA), classification with projection pursuit regression with super smoothing and classification with projection pursuit regression with the Hermite functions with order equal to 7 and with automatic order selection. Table 2 shows that classification with projection pursuit regression with the Hermite functions with automatic order overperforms other methods. Also, classification with projection pursuit regression with super smoothing is more stable because the standard deviation is smaller than others.

4. Concluding remarks

As a tool for high dimensional data analysis, we examined algorithms for projection pursuit regression and developed a macro program for prediction and classification using projection pursuit regression in SAS. In the macro, the order of the Hermite function can be adjusted automatically with F-test. Also, we evaluated the performance for prediction and classification. For learning speed, the macro with super smoothing takes a lot of time. But, the macro with the Hermite function takes much less time. Smoothing with the Hermite function with order of 7 looks better in relatively simpler form of functions and smoothing with the Hermite functions with automatic order selection outperforms in a complicate form of functions.

REFERENCE

[1] Breiman, L., Friedman, J. H., Olshen, R. A. and Stone, C. J. (1983). *Classification and regression trees*, Wadsworth International, Belment, CA.

[2] DeVore, R. A. (1991). Degree of nonlinear approximation, *Approximation Theory, VI*, C. K. Chui, L. L. Schumaker, and D. J. Ward (eds.), 175-201. Academic Press, New York.

[3] Donoho, D. L. and Johnstone, I. M. (1989). Projection-based approximation and a duality with kernel methods, *The Annals of Statistics*, 17, 58-106.

[4] Friedman, J. H. (1984a). *A variable span smoother,* Laboratory for Computational Statistics Technical Report No. 5, Dept. of Statistics, Stanford University.

[5] Friedman, J. H. (1984b). *SMART users' guide,* Laboratory for Computational Statistics Technical Report No. 1, Dept. of Statistics, Stanford University.

[6] Friedman, J. H. and Stuetzle, W. (1981). Projection pursuit regression, *Journal of the American Statistical Association* 76, 817-823.

[7] Hwang, J.-N., Lay, S.-R., Maechler, M., Martin, D. and Schimert, J. (1994). Regression modeling in back-propagation and projection pursuit learning, *IEEE Transactions on Neural Networks* 5, 342-353.

[8] Cherkassky, Vladimir S. and Mulier, Filip M. (1998). *Learnig From Data,* John Wiley & Sons Inc.

[9] Zhao, Y. and Atkeson, C. G. (1992). Some approximation properties of projection pursuit learning networks, *NIPS4*, pp. 936-943.