Bootstrapping Nonparametric Feature Selection Algorithms for Mining Small Data Sets

Jen-Lun Yuan

Abstract

This paper presents feature selection algorithms based on nonparametric feature ranking indices and demonstrates for small data sets that by bootstrapping feature ranking indices one uniformly (over various data sets and different ranking indices) improves the performance of correct detection of true features (i.e. probability of the top ranking features matching the true ones). [1]

I. Introduction

Selection of input variables (features) is often necessary in applying neural networks to data mining. The problem is to select from a large number (hundreds or thousands) of candidate variables in a "feature pool" a few features used as the input to the neural network. The selection process is necessary for several reasons: i) the size of training data (number of samples) may be limited and therefore a parsimonious model with less complexity is preferred, ii) even the training size is "sufficiently large" (depending on the problem several hundred thousands observations may be considered as large enough for moderate-size neural nets) trimming redundant or irrelevant features helps accelerate the training process and improve generalization (testing) performance, and iii) the analyst (data modeler) may not have *a priori* knowledge of the underlying application to determine the "true" features of the statistical model and can only "guess" which feature might to certain extent effect the output (target variable).

To avoid exhaustive search of all possible combinations of candidates, one strategy is to rank each feature by giving a score or "figure of merit" which assesses the "importance" of each candidate and the selection algorithm chooses the top ranked ones. This ranking approach can be applied recursively to a hierarchy of groups of "similar" features to avoid exclusive choice of similar high-scored features [Yuan and Fine, 1998].

[1] 4402 Stearns Hill Road, Waltham, MA 02451. Email: jlyuan@hotmail.com

One problem for the index-based feature selection approach is the variability (features selected vary significantly from one sample of training set to another) of the selected feature set [Gong, 1982]. In this paper, we propose to bootstrap the feature selection index to improve the stability of the selected feature set while achieving a higher feature detection rate.

In Section II, we present the feature ranking index used in this paper. Section III presents the bootstrapping algorithm and Section IV demonstrates the simulation results.

II. Feature Ranking Indices

A feature ranking (scoring) index maps one (candidate) feature x_k into a real number s which represents the score for feature x_k. Depending upon the interpretation of the index, either the higher or lower the score, the more likely a feature is to be incorporated into the selected set. In supervised training, the target (output) information y can be utilized in calculating s.

The training data is denoted by $(x_i, y_i), i = 1, ..., n$, n is the number of observations, x_i is m-dimensional and each feature is represented by a column vector $x_i(k), k = 1, ..., m$ where m is the number of candidates.

II.1 Difference-based variance estimates (the I_n statistics).

A difference-based variance estimation I_n proposed in [Hall, 1990] quantifies the "scatterness" of a scatterplot of $(x_i(k), y_i)$ (the less scatter the scatterplot is, the sharper, and therefore the better a feature is) by calculating a sum of squared difference of adjacent "concomitants" [Goel and Hall, 1994] $y_{(i)}$ and $y_{(i+1)}$ of order statistics $x_{(i)}$ and $x_{(i+1)}$:

$$I_n = \frac{1}{2n} \sum_{i=1}^{n-1} (y_{(i+1)} - y_{(i)})^2. \quad (1)$$

This estimate and a higher order generalization of this index and their asymptotic properties were established by [Hall, 1990] and were used for feature selection in [Yuan and Fine, 1998].

The estimator defined in (1) takes the first order difference. Higher order difference can be found by finding the

0-7803-5529-6/99/$10.00 ©1999 IEEE

coefficients associated with each $Y_{(i)}$'s (see [Hall, 1990]) in:

$$\min_{\sum_{j=0}^{d_j=0} d_j^2=1} \sum_{i=1}^{n-m} \left(\sum_{j=0}^{m} d_j Y_{(i+j)} \right)^2 .$$

This paper uses the calculated 3rd-order I_n^3:

$$I_n^3 = \sum_{i=1}^{n-3} \big(.194Y_{(i)} + .281Y_{(i+1)} + .383Y_{(i+2)} - .858Y_{(i+3)}\big)^2$$

and the 5_th-order I_n^5 :

$$I_n^5 = \sum_{i=1}^{n-5} \big(.906Y_{(i)} - .26Y_{(i+1)} - .217Y_{(i+2)} - 177Y_{(i+3)} - .142Y_{(i+4)} - .11Y_{(i+5)}\big)^2$$

II.2 The Kolmogorov-Smirnov(K-S) statistics.

The K_n index measures the deviation of a distribution function (CDF) from that of an uniform one - the least informative distribution function which implies the more deviation, the better a feature). We used the index K_n taking the maximum/average vertical distance between the empirical distribution and the CDF of an uniform (0,1) distribution within each "slice" of y_i's (see Step 2. below):

1. Sort on $\{y_i\}$ (getting $\{y_i'\}$) and replace x_i by $x_i' = \text{rnk}(x_i)$ where $\text{rnk}(x_i)$ is the rank of x_i (scaled to the interval $[0,1]$) in the data $\{x_i\}_{i=1}^N$.
2. (Slice) Divide the range of $\{y_i'\}$ into H intervals, $I_1, .., I_H$, and within each interval I_h, $\text{rnk}_h(x_i')$ is taken as the (scaled) rank of x_i' inside $\{x_i'\}$ where $y_i' \in I_h$.
3. Calculate various K_n index by taking combinations of the maximum or average distance among all slices $I_1, .., I_H$,:

$$K_n^1 = \text{avg}_h \max_{i:y_i' \in I_h} \{|\text{rnk}_h(x_i') - x_i'|\}, \quad (2)$$

$$K_n^2 = \max_h \max_{i:y_i' \in I_h} \{|\text{rnk}_h(x_i') - x_i'|\}, \quad (3)$$

$$K_n^3 = \text{avg}_h \text{avg}_{i:y_i' \in I_h} \{|\text{rnk}_h(x_i') - x_i'|\}. \quad (4)$$

II.3 χ^2 statistics.

The χ^2 statistics is capable of detecting nonlinear dependency relationship between two variables besides those typically characterized by using correlation coefficients. The χ^2 statistics can be implemented by constructing a $S \times T$ contingency table through sorting and slicing for both x and y into S and T slices respectively. A selection index is then calculated using the Pearson formula:

$$I_{\chi^2} = \sum_{s=1}^{S} \sum_{t=1}^{T} \frac{(N_{st} - N\theta_{st})^2}{N\theta_{st}} \quad (5)$$

where N_{st} is the number of data pairs $(x_i(k), y_i)$ falling into cell (s,t) of the contingency table and θ_{st} is the probability of occurrence which is equal to $1/(ST)$ for uniformly distributed X and Y. Better (scored) features are identified by those with large I_{χ^2} values.

II.4 Mutual information measure.

Using mutual information measure for scoring / selecting features have been proposed in [Battiti, 1994]. A pair-wise mutual information measure can be efficiently implemented using the $S \times T$ contingency table similar to the χ^2-based index where the join density function defining a mutual information measure $I(x;y)$ is replaced by counting the number of data points (N_{st}) falling into each cell of the contingency table:

$$I(x;y) = \sum_{s=1}^{S} \sum_{t=1}^{T} \frac{N_{st}}{N} \log \frac{N_{st} N}{\sum_{s=1}^{S} N_{st} \sum_{t=1}^{T} N_{st}} \quad (6)$$

III. The Bootstrapping Algorithm

We propose to bootstrap the feature score to obtain a sampling distribution of the score through a Monte Carlo procedure of sampling (with replacement) of the training data. The algorithm is easy to implement and is applicable to any scoring function besides those proposed in Section II. This simplicity is achieved through ignoring the error structure since it does not resample directly with respect to the error term in the regression model (see [Mooney, 1993] p.16).

```
Input: training data X = (x_i(k), y_i),
       number of resampling B.
Output: scores s(k) for feature k.
       for k=1,...,m
          for b=1,...,B
             X1 ⟵ random_sample(X);
             s(k,b) ⟵ S(X1(k),Y);
          end b
          s(k) ⟵ avg_b(s(k,b));
       end k
```

Besides the parameter required for feature scoring function s(.), the only parameter required by this algorithm is the re-sampling number B. Practical number of $B = 50$–200 has been suggested by Mooney and Duval [Mooney, 1993]. B=100 is used throughout this paper in the simulation to match the number of recalculating the feature score for those without using bootstrapping.

IV. Simulation Results

We carried out an experiment conducted in [Yuan, 1996] for detecting a set of "genuine features" from a feature pool with added "forgery features" (features irrelevant to the target y). All features (genuine + forgery) are generated independently of each other.

We used six regression model each of which has K genuine features $x_{t1}, x_{t2}, .., x_{tK}$ (e.g. $K = 4$ for model 1 and $K = 2$ for model 2,3,...,6):

$$\begin{aligned}
&\text{model 1} \quad y_t = x_{t1} + \cdots + x_{t4} + \epsilon_t,\\
&\text{model 2} \quad y_t = .8x_{t1} + .2x_{t2} + \epsilon_t,\\
&\text{model 3} \quad y_t = x_{t1}(x_{t1} + x_{t2} + 1) + \epsilon_t,\\
&\text{model 4} \quad y_t = \frac{x_{t1}}{0.5 + (x_{t2} + 1.5)^2} + \epsilon_t,\\
&\text{model 5} \quad y_t = x_{t1}^2 + x_{t2}^2 + \epsilon_t,\\
&\text{model 6} \quad y_t = x_{t1}x_{t2} + \epsilon_t.
\end{aligned}$$

The data x_{ti} and ϵ_t are generated i.i.d. following the Gaussian N(0,1) distribution. Note that model 1 and 2 are linear (in $\beta_1^T \underline{x}_t$) while model 3 to 6 are nonlinear (in $\beta_1^T \underline{x}_t$ and $\beta_2^T \underline{x}_t$).

The input vector $\underline{x}_t = (x_{t1}, ..., x_{tK}; x_{t,K+1}, ..., x_{td})$ consists of "genuine features" $x_{t1}, ..., x_{tK}$ and "forgery features" $x_{t,K+1}, ..., x_{td}$. The input dimension $d = 10$. For each model, we generated N=200 i.i.d. samples of $(\underline{x}_t, y_t)$ and calculated an index for each feature x_{ti}. The $d = 10$ indices are then sorted and the largest (or smallest based on the feature scoring convention) K features are checked for matching the set of index $\{1, 2, ..., K\}$. If the two are identical, the features have been successfully detected by the algorithm.

This process is repeated for $R = 100$ times to estimate the detection rate. The whole process of calculating the detection rate for each model by each scoring algorithm is then repeated for a total of $M = 100$ times to estimate the mean and standard deviation of the detecting rate. Table IV.1 lists mean and standard deviation of the detection rates.

model	I_n	I_n^3	I_n^5	K_n^1	K_n^2	K_n^3
1	.44	.69	.74	.94	.81	.98
	(.05)	(.04)	(.04)	(.02)	(.04)	(.01)
2	.16	.17	.18	.26	.22	.30
	(.04)	(.04)	(.03)	(.04)	(.04)	(.05)
3	.33	.35	.41	.57	.42	.63
	(.05)	(.05)	(.05)	(.05)	(.05)	(.04)
4	.42	.59	.49	.50	.58	.55
	(.05)	(.05)	(.05)	(.05)	(.05)	(.05)
5	.90	.94	.94	.77	.36	.76
	(.03)	(.02)	(.02)	(.04)	(.05)	(.04)
6	.79	.92	.94	.16	.11	.13
	(.04)	(.02)	(.02)	(.04)	(.03)	(.03)

model	I_{χ^2}	$I(x;y)$
1	.97	.94
	(.02)	(.03)
2	.28	.25
	(.04)	(.04)
3	.45	.43
	(.05)	(.05)
4	.39	.36
	(.04)	(.05)
5	.98	.97
	(.01)	(.02)
6	.19	.15
	(.04)	(.03)

Table IV.1. Mean/(Std. Dev.) of feature detection rates for six artificial data models before bootstrapping.

We next bootstrapped the feature scores using the algorithm described in Section III. The total number of bootstrapping resampling B is set to equal to the total number of simulations $M = 100$ for the non-bootstrapped case in order to compare detecting rates on the basis of the same amount of computation. Table IV.2 gives results for comparing with the corresponding columns in Table IV.1:

model	I_n	I_n^3	I_n^5	K_n^1	K_n^2	K_n^3	I_{χ^2}	$I(x;y)$
1	.65	.71	.79	1.0	.99	1.0	1.0	.96
2	.18	.25	.24	.42	.39	.40	.45	.39
3	.31	.32	.36	.59	.53	.65	.59	.55
4	.52	.63	.54	.62	.75	.69	.48	.48
5	.93	.94	.93	.93	.82	.93	1.0	.99
6	.90	.93	.93	.21	.20	.14	.37	.25

Table IV.2. Feature detection rates of each algorithm for six artificial data models after bootstrapping.

By comparing Table IV.1 with Table IV.2 one clearly sees that the detection rate improves uniformly over different data sets and over various selection indices.

V. Conclusion

In this paper, it is demonstrated that by bootstrapping, one uniformly enhances the performance of feature selection algorithms based on several nonparametric feature scoring/ranking indices. The selection indices are capable of detecting nonlinear relationship between input and output variables. The current implementation was applied to a small data set with only 200 observations. However, with the current trend of microprocessor speed and cost, we expect computational intensive bootstrapping-based approach to become a practical solution to the problem of feature selection for neural networks.

References

[1] Battiti, Using Mutual Information for Selecting Features in Supervised Neural Net Learning, *IEEE Transaction on Neural Networks,* Vol.5, No.4, July 1994.

[2] Belue, L. M. and K. W. Bauer, Determining Input Features for Multilayer Perceptrons, *Neurocomputing,* 7:111–121, 1995.

[3] Gasser, T., L. Stroka, and C. Jennen-Steinmetz, Residual Variance and Residual Pattern in Nonlinear Regression, *Biometrica,* 73(3):625–633, 1986.

[4] Goel, P. K. and P. Hall, On Average Difference Between Concomitants and Order Statistics, *Annals of Probability,* 22:126–144, 1994.

[5] Gong, G., Some Ideas on Using the Bootstrap in Assessing Model Variability, Proceedings of the 14th Symposium on Interfacing, pp. 169–173, Computer Science and Statistics, 1982.

[6] Hall, P., J. W. Kay, and D. M. Titterington, Asymptotic Optimal Difference-based Estimation of Variance in Nonparametric Regression, *Biometrica,* 77(3):521–528, 1990.

[7] Mooney, C. Z. and R. D. Duval, *Bootstrapping: A Nonparametric Approach to Statistical Inference,* Sage University Press, Newbury Park, CA, 1993.

[8] Yuan, J.-L., Feature Selection using Slicing Inverse Regression, International Symposium on Multitechnology Information Processing, Hsing-Chu, Taiwan, 1996.

[9] Yuan, J.-L., and T. L. Fine, Neural Network Design for Small Training Sets of High Dimension, *IEEE Transaction on Neural Networks,* 9(2):266–280, March 1998.