Selection of Binary Variables and Classification by Boosting

JUNYONG PARK[1], JAYSON D. WILBUR[2], JAYANTA K. GHOSH[3], CINDY H. NAKATSU[4], AND CORINNE ACKERMAN[4]

[1]Department of Mathematics and Statistics, Univeristy of Maryland Baltimore County, Baltimore, Maryland, USA
[2]Department of Mathematical Sciences, Worcester Polytechnic Institute, Worcester, Massachusetts, USA
[3]Department of Statistics, Purdue University, West Lafayette, Indiana, USA
[4]Department of Agronomy, Purdue University, West Lafayette, Indiana, USA

We adopt boosting for classification and selection of high-dimensional binary variables for which classical methods based on normality and non singular sample dispersion are inapplicable. Boosting seems particularly well suited for binary variables. We present three methods of which two combine boosting with the relatively classical variable selection methods developed in Wilbur et al. (2002). Our primary interest is variable selection in classification with small misclassification error being used as validation of proposed method for variable selection. Two of the new methods perform uniformly better than Wilbur et al. (2002) in one set of simulated and three real life examples.

Keywords Boosting; Cross validation; DNA fingerprints; High-dimensional data; Multivariate binary data; Thresholding; Variable selection.

Mathematics Subject Classification Primary 62H30; Secondary 62P10.

1. Introduction

Wilbur et al. (2002) (henceforth abbreviated as [W]) have explored the problem of identifying treatment dependent microbial populations within a community by comparative analysis of community DNA fingerprints. A community DNA

Received August 18, 2005; Accepted January 19, 2007

Address correspondence to Junyong Park, Department of Mathematics and Statistics, Univeristy of Maryland Baltimore County, Baltimore, MD 21250, USA; E-mail: junpark@math.umbc.edu

fingerprint may be regarded as high-dimensional multivariate binary data, the value of each variable indicating the presence or absence of a band (i.e., the presence or absence of a specific microbial population). The fingerprint is generated after extracting DNA from the environment of interest and using polymerase chain reaction (PCR) to amplify the target gene. Typically for community analysis, the small subunit rRNA gene is targeted. Separation of the PCR products by gel electrophoresis produces a genetic fingerprint of the populations comprising a community (Nakatsu et al., 2000). See Fig. 1 of [W]. The aim was to identify a subset of variables (i.e., bands) that may help explain the major source of variation between communities of interest. In natural systems each variable contributes to the community and the desire is to identify the minimal number of variables (i.e., bands) within a treatment so that they can then be studied biologically in greater detail. The main example of [W] concerned four populations associated with a single crop species, namely corn. We re-examine that data set as well as two new data sets involving soybean.

In [W], certain methods of selection of variable were proposed, and applied to DNA fingerprint data. The methods were evaluated by examining how well one could classify samples based on selected variables, the idea being that only selection of important variables will lead to good classification. We continue the study initiated in [W] by exploring new methods of variable selection, depending on boosting, which appears to work more successfully than those of [W]. Boosting is an algorithmic approach to classification which has been studied thoroughly both by computer scientists and statisticians (Breiman, 2004; Freund and Schapire, 1997; Friedman et al., 2000). We use a specific version of boosting, namely, Adaboost.

In the present article, some of the ideas of [W] are combined with boosting and evaluated through simulated and real examples. Three methods of boosting are proposed. In the first, a simple stopping rule is used to avoid overfitting and a threshold value prunes out some of the variables appearing in the classifier. In the second, a method from [W] is coupled with boosting at each iteration. The third method is the first method preceded by a method from [W] (i.e., we first use methods in [W] to select a set of variables and then apply the first method).

We evaluate the methods as in [W] by their misclassification probabilities, the smaller the errors the better are the methods. The first method is quite successful and works more successfully than those in [W] in the sense that misclassification errors of the first method are much smaller than those in [W]. The second method does not perform well compared with the other two methods in the sense that it does not improve the misclassification error or reduce the number of variables. The third method combines the strength of the method in [W] and boosting. It controls the misclassification error with a relatively small number of variables.

We also evaluate the performance of the Bayes rule, which serves as an oracle and provides an unattainable lower bound to the misclassification error of all methods in classification problems. If a method has misclassification probability close to that of the oracle, then no other method can perform much better. The primary object of the article is to show through examples that boosting can be an effective algorithm for variable selection in high-dimensional problems.

2. Notations and Variable Selection Methods in [W]

Let $\boldsymbol{X}$ denote a $n \times d$ matrix of binary indicators, the rows of which are the n sample vectors $\boldsymbol{x}_{ij} = (x_{ij}^1, \ldots, x_{ij}^k, \ldots, x_{ij}^d)$; $i = 1, \ldots, t$, $j = 1, \ldots n_i$, $k = 1, \ldots, d$

and $\sum_{i=1}^{t} n_i = n$ such that $x_{ij}^k = 1$ if the kth microbial population is present in the jth sample from the ith class (i.e., ith treatment). Marginally, we model $X_{ij}^k \sim$ Bernoulli (p_{ik}) and define

$$\boldsymbol{S_W} = \frac{1}{n-1} \sum_{i=1}^{t} \sum_{j=1}^{n_i} (\boldsymbol{x}_{ij} - \bar{\boldsymbol{x}}_{i.})'(\boldsymbol{x}_{ij} - \bar{\boldsymbol{x}}_{i.})$$

$$\boldsymbol{S_B} = \frac{1}{n-1} \sum_{i=1}^{t} n_i (\bar{\boldsymbol{x}}_{i.} - \bar{\boldsymbol{x}}_{..})'(\bar{\boldsymbol{x}}_{i.} - \bar{\boldsymbol{x}}_{..})$$

where $\bar{\boldsymbol{x}}_{i.}$ is the ith class mean vector and $\bar{\boldsymbol{x}}_{..}$ is the grand mean vector.

Normality fails for binary variables. Moreover, Fisher's linear discriminant function can be rather inefficient if $d \approx n$ or larger than n. Because of its near singularity (or singularity), $\boldsymbol{S}_W^{-1}$ is a poor estimate of the inverse of the population covariance matrix. Reducing the dimension (i.e., d) via variable selection, is one way to address this. We consider two different variable selection methods that are suggested in [W]. Define

$$SS_W[l, m] = \sum_{i=1}^{t} \sum_{j=1}^{n_i} (x_{ij}^l - \bar{x}_{i.}^l)(x_{ij}^m - \bar{x}_{i.}^m) \tag{1}$$

$$SS_B[l, m] = \sum_{i=1}^{t} n_i (\bar{x}_{i.}^l - \bar{x}_{..}^l)(\bar{x}_{i.}^m - \bar{x}_{..}^m) \tag{2}$$

One method of [W] is to consider variables individually based on the statistic

$$D^2(k) = \frac{SS_B[k, k]}{SS_W[k, k]} = \frac{\sum_{i=1}^{t} n_i (\bar{x}_{i.}^k - \bar{x}_{..}^k)^2}{\sum_{i=1}^{t} n_i \bar{x}_{i.}^k (1 - \bar{x}_{i.}^k)}. \tag{3}$$

A variable X^k is selected if $D^2(k)$ exceeds empirical threshhold C_α^k. The threshholds C_α^k are computed such that relative to a reference distribution generated by a permutation scheme which assumes there is no difference between classes (1) the probability of incorrectly selecting at least one variable is α and (2) any particular variable has the same probability of being selected (incorrectly). That is to say,

$$P\Big(\bigcup_{k=1}^{d} \{D^2(k) > C_\alpha^k\} \Big) = \alpha$$

and

$$P(D^2(1) > C_\alpha^2) = P(D^2(1) > C_\alpha^2) = \cdots = P(D^2(d) > C_\alpha^d).$$

In a second method of [W], based on Fisher's linear discriminant functions (LDF), let $\mathbf{W}$ be the $d \times d$ matrix with (i, j) element,

$$W[i, j] = \frac{S_B[i, j]}{\sqrt{S_W[i, j] S_W[i, j]}}. \tag{4}$$

$\mathbf{W}$ is considered as an estimate of $S_W^{-1} S_B$. The coefficienct vectors for the discriminant functions $\boldsymbol{f_h}$, $h = 1, \ldots, q \leq \min(t-1, d)$ are given by the eigenvectors of $\mathbf{W}$, normalized so that $\boldsymbol{F}'\boldsymbol{S}_W\boldsymbol{F} = \boldsymbol{I}_q$. In this procedure, a variable X^k is selected if $f_{hk} \notin (C^k_{h,\alpha/2}, C^k_{h,1-\alpha/2})$ for at least one $h = 1, \ldots, q$. As in the first method of [W], the threshholds are computed such that relative to a reference distribution generated by a permutation scheme which assumes there is no difference between classes the probability of incorrectly selecting at least one variable is α and any particular variable has the same probability of being selected.

In [W], variable selection methods are evaluated via a classification rule. The posterior probability of the treatment group g given $\boldsymbol{X}_{ij} = \boldsymbol{x}_{ij}$ is

$$P(i = g \mid \boldsymbol{X}_{ij} = \boldsymbol{x}_{ij}) = \frac{P(\boldsymbol{X}_{ij} = \boldsymbol{x}_{ij} \mid i = g)P(i = g)}{\sum_{i=1}^{t} P(\boldsymbol{X}_{ij} = \boldsymbol{x}_{ij}|i)P(i)}$$

and $\boldsymbol{x}_{ij}$ is classified into g if

$$P(i = g^* \mid \boldsymbol{X}_{ij} = \boldsymbol{x}_{ij}) = \max_{g} P(i = g \mid \boldsymbol{X}_{ij} = \boldsymbol{x}_{ij}).$$

As $P(\boldsymbol{X}_{ij} = \boldsymbol{x}_{ij} \mid i = g)$, [W] consider a conditionally independent Bernoulli model and a conditionally independent logistic regression. Conditional independent Bernoulli model can be specified as

$$P(\boldsymbol{X}_{ij} = \boldsymbol{x}_{ij} \mid i = g) = \prod_{k \in K} p_{gk}^{x^k_{ij}}(1 - p_{gk})^{1-x^k_{ij}} \tag{5}$$

where K is an index set for the selected variables, with parameters p_{gk} estimated by $\bar{x}^k_{g.}$. Conditionally independent logistic regression is specified by

$$logit(P(i = g \mid \boldsymbol{X}_{ij} = \boldsymbol{x}_{ij})) = \alpha_{g0} + \sum_{k \in K} \alpha_{gk} x^k_{ij} + \epsilon_{gij} \tag{6}$$

where $g = 1, \ldots, t$, $i = 1, \ldots, t$, and $j = 1, \ldots, n_i$. Maximum likelihood estimates for α are computed using iteratively weighted least squares. Cross-validation and test error (i.e., estimate of misclassification probability calculated from an independent sample) are used to evaluate the subsets of variables selected.

3. Boosting

Friedman et al. (2000) argued that Adaboost is equivalent to finding a base classifier that minimizes the empirical risk with revised weights at each iteration. These base classifiers are combined to form classifier. Denote by T_m a base classifier at the mth iteration. Let $X = (X^1, \ldots, X^d)$ be the d dimensional random vector consisting of random variables. We create $2d$ base classifiers, $\pm(I(X^k = 1) - I(X^k = 0)) = \pm(2X^k - 1)$ for $1 \le k \le d$. The d dimensional vector with binary co-ordinate leads to $2d$ base classifiers.

If a co-ordniate, X^k is continuous valued rather than binary, it is usually converted into a base classifier, a tree with two nodes, namely $\pm(I(X^k \ge t) - I(X^k < t))$, where t is usually an observed point. Hence the total number of base classifiers may be quite large for continuous variables.

With the base classifiers, $\pm(2X^k - 1)$ for $1 \le k \le d$, we now define the Adaboost algorithm in Freund and Schapire (1997) and Friedman et al. (2000). Suppose we have n samples $\{(\boldsymbol{x}_j, y_j)\}_{j=1}^n$ where $y_j \in \{-1, 1\}$ which represents two classes and $\boldsymbol{x}_j$ is d-dimensional vector.

1. Start with weights $w_j = \frac{1}{n}$, $j = 1, \ldots, n$
2. Repeat for $m = 1, 2, \ldots, M$:
   (a) Select a base classifier $T_m(X) \in \{-1, 1\}$ minimizing $\sum_{j=1}^n w_j I(y_j \neq T_m(\boldsymbol{x}_j))$ from the training data with weights w_j.
   (b) Compute $e_m = \frac{\sum_{j=1}^n w_j I(y_j \neq T_m(\boldsymbol{x}_j))}{\sum_{j=1}^n w_j}$ and $c_m = \log((1 - e_m)/e_m)$
   (c) Set $w_j \leftarrow w_j \exp[c_m \cdot I_{(y_j \neq T_m(\boldsymbol{x}_j))}]$, $j = 1, 2, \ldots, n$ and renormalize so that $\sum w_j = 1$ where $I(\cdot)$ is an indicator function.
3. The final classifier after M iterations is $\mathrm{sign}\big[\sum_{m=1}^M c_m T_m(X)\big]$

Clearly, misclassified units are given higher weights so as to choose base classifier that reclassify most of them correctly. If e_m is close to 1/2, then w_j in (c) will not change much. There is some debate as to whether boosting would overfit if it is continued up to this stage. See this connection in Freund and Schapire (2004) who hold a different view. Our stopping procedure, explained below, leads to stopping much earlier.

Since a base classifier is $T_m(X) = \pm(2X^l - 1)$, the linear combination of base classifiers can be re-expressed as a linear combination of original variables. In other words, a linear classifier, $f(X)$, is

$$f(X) = \sum_{m=1}^{M} c_m T_m(X) = \sum_{l \in B_M} c_l^*(2X^l - 1) \tag{7}$$

where B_M is the set of selected variables until M iteration. We rearrange the classifier with respect to $2X^l - 1$ since the same variables may appear again during the whole M iterations. Here c_l^* is the coefficient of $2X_l - 1$ after rearranging. For the notational convenience, we use c_l as a coefficient of $2X^l - 1$ instead of c_l^*.

Various authors have pointed out that boosting may overfit. See Breiman (2004), Jiang (2004), Lugosi and Vayatis (2004), and Zhang (2004). In other words, boosting tends to attain too small a risk for the training set or in cross-validation by selecting more than the optimal number of variables. But as refree pointed out, some authors such as Bartlett et al. (1998) and Freund and Schapire (2004) have different aspect that boosting does not overfit data. But in our case, we observe overfitting problem, so we consider regularization of boosting. There are several regularization methods for avoiding this. These include: (1) early stopping, and (2) employing a norm constraint. Norm constraint puts a restriction on the l_1 norm $\sum |c_m|$. Norm constraint regularization is more flexible in finding a classifier, but numerically more intensive than early stopping. In Rosset et al. (2004), it is also argued that l_1 regularization can be done directly, and naturally, by early stopping of boosting. In this present work, we will consider early stopping as a regularization of boosting. Our stopping rule is explained below.

If there are enough data, a subset of data may be separated into a test set for stopping rule. Otherwise, cross-validation error is used. We employ the stopping rule which minimizes cross-validation error rate in this article. We first iterate a large number of times, say, M_0, and then for each $M \leq M_0$, we estimate the misclassification error by cross-validation. If the estimated misclassification error is minimized at $M_{min} \leq M_0$, then M_{min} is our stopping time. The variables selected by boosting stopped at M_{min} are kept as the final result, the remaining iterations are ignored. As cross-validation, K-fold cross-validation is common which means that data set is divided into K subsets. Then $K-1$ subsets are used to estimate error of the remaining set and this is repeated for all K subsets. We use $K = 10$ and $K = n$, each of which is called ten-fold and leave-one-out cross-validation respectively, vide Hastie et al. (2001). Ten-fold cross-validation is used in the simulated example and leave-one-out cross-validation is used in three real data sets in Sec. 5. For the corn data of [W], the algorithm stopped after about fifty iterations however, the soybean data required several hundred iterations.

4. Variable Selection Methods with Boosting

Simulations in the next section show that even after boosting is stopped early, it may have many variables (i.e., a high-dimensional model is chosen). But many of the variables have small coefficients, so they do not greatly affect classification.

In our first modification, we exclude the variables with small coefficients with only a small increase in misclassification error. More precisely, we exclude the variables with the coefficients smaller than a threshold parameter, c. Finding an appropriate threshold value is important to maintain balance between the number of variables and performance of the classifier. We try this for several different threshold values and select one for which cross-validation error is minimized. We extend this idea of thresholding to the problem of multiclass classification. If there are t classes, then we need t classifiers, each of which is a classifier for one class against the other classes. In other words, a classifier for ith class is computed when $y_j = 1$ for ith class and $y_j = -1$ for the other classes in boosting algorithm. After M iterations, we have ith classifier, $f_i(X) = \sum_{l \in B_M} c_{il}(2X^l - 1)$, for $i = 1, \ldots, t$ where c_{il} is the coefficient of $2X^l - 1$ of the ith classifier and B_M is the set of the selected variables by t classifiers. If the coefficients of $2X^l - 1$ for $1 \leq i \leq t$ are similar, then $2X^l - 1$ does not greatly affect the classification. With this idea, if the range of c_{il} for $1 \leq i \leq t$ is small, which means the coefficients are similar, then $2X^l - 1$ can be neglected. Define $R_l = \max_{1 \leq i \leq t} c_{il} - \min_{1 \leq i \leq t} c_{il}$, the range of the coefficients of $2X^l - 1$ for $1 \leq i \leq t$.

Method 1

1. Apply boosting to original data. Determine stopping time by cross-validation.
   (a) In these classifiers, remove the lth variable if $R_l \leq c$.
   (b) Apply boosting again with only the selected variables.
2. Repeat (a) and (b) for different threshold values c and choose c minimizing cross-validation error.

In (b), after variables are selected, we construct a classifier again with the selected variables. This is for determining the best model in the subspace induced by the selected variables. Method 1 takes care of this by boosting with a single variable at a time.

As refree pointed out, some pruning methods were already suggested, for example, by Margineantu and Dietterich (1997). They also suggested early stopping, but not exactly the same as ours. Our stopping rule consists of double regularizations: early stopping and hardthresholding in (b). Since we focus on the variable selection, we remove the variables with small coefficients even if they are included after early stopping. So we adopt hardthresholding to remove more variables in step (b). After the variables are selected, we apply adaboost again to only the selected variables. Briefly speaking, we considered more steps to select a smaller subset of variables and achieve more accurate model with the selected variables than the old stopping rules.

The second variable selection methods is a combination of boosting and variable selection methods in [W]. Let SS_W and SS_B in (1) and (2) be redefined with revised weights as

$$SS_W^w[l,m] = \sum_{i=1}^{t}\sum_{j=1}^{n_i} w_{ij}(x_{ij}^l - \bar{x}_{..}^l)(x_{ij}^m - \bar{x}_{..}^m) \tag{8}$$

$$SS_B^w[l,m] = \sum_{i=1}^{t} w_{i.}(\bar{x}_{i.}^l - \bar{x}_{..}^l)(\bar{x}_{i.}^m - \bar{x}_{..}^m) \tag{9}$$

where $w_{i.} = \sum_{j=1}^{n_i} w_{ij}$.

In each iteration of boosting, the empirical distribution of samples changes. Based on these facts, we define the variable selection combining variable selection in [W] and revised empirical distributions obtained at each iteration. It was expected that the changed empirical distribution will reduce misclassification due to the increased weight for misclassified units. This did not occur, most likely because Method 2 selects a full classifier rather than a weak base classifier at each iteration.

Method 2

1. Start with weights $w_{ij} = \frac{1}{n}$ where $i = 1, \ldots, t$ and $j = 1, \ldots, n_i$.
2. Repeat the following: $m = 1, \ldots, M$
   - (a) Select variables based on weighted samples.
   - (b) Construct classification rule, f_m, based on selected variables such that $f_m(X) = \mathrm{argmax}_{1 \le i \le t} P(i \mid X)$.
   - (c) Update weights of samples by boosting. $w_{ij} \leftarrow w_{ij} \exp(c_m I(f_m(\boldsymbol{x}_{ij}) \neq i))$ where $c_m = \log \frac{1-e_m}{e_m}$ and $e_m = \frac{\sum_i^t \sum_{j=1}^{n_i} w_{ij} I(f_m(x_{ij}) \neq i)}{\sum_{i=1}^{n} \sum_{j=1}^{n_i} w_{ij}}$.
   - (d) Update SS_W^w and SS_B^w using (8) and (9).
3. Select f_m which minimizes cross-validation error.

When $m = 1$, variable selection in (a) is the same as the methods in [W]. From $m \geq 2$, the empirical distribution changes, causing the variable selection in (a) to work differently. In (b), $P(i \mid \boldsymbol{x}_{ij})$ can be (5) or (6).

Method 3

1. Select variables.
2. Use Method 1 with the selected variables in 1.

In the second step of Method 3, variables are reselected from the variables selected in the first step. With Method 3, we select a small subset of variables compared to Method 1 and Method 2 since we select variables twice in the first and the second step. Boosting with preselected variables is also discussed in Dudoit et al. (2002). Genes (i.e., variables) are selected on the basis of their between-group to within-group sums of squares (BW ratio) which is the same as D^2 in (3). However, Dudoit et al. (2002) chose the genes corresponding to the 50 highest values of D^2, whereas in [W] this is based on a permutation test taking into account the dependencies. Method 3 also tries to reduce the number of variables by thresholding as Method 1. Method 3 starts with a much smaller subset of variables than that in Method 1, so its performance depends on the quality of the preselected variables.

5. Examples

In this section, we present examples to see how the three methods work in simulations and in the analysis of DNA fingerprints from plant rhizosphere microbial communities. They are compared with the original boosting and the methods in [W]. Also, as a lower bound the Bayes classifier is considered in the simulated examples.

5.1. *Multivariate Binary Data with Exchangeable Correlation Matrix*

In this section, we generate correlated multivariate binary data and compute optimal Bayes error via multivariate normal distribution with an exchangeable correlation matrix, namely,

$$\Sigma_\rho = (1-\rho)I + \rho J.$$

Here J is a $d \times d$ matrix of 1's and I is a $d \times d$ identity matrix. We relate binary variables X_{ij}^k and the normal variables Y_{ij}^k through $X_{ij}^k = I(Y_{ij}^k \geq 0)$. Suppose we want the marginal probability of $X_{ij}^k = 1$ to be p_{ik}. This is ensured by taking the mean of Y_{ij}^k to be $\Phi^{-1}(p_{ik})$, $k = 1, 2, \ldots, d$.

Thus we define $\mathbf{p}_i = (p_{i1}, p_{i2}, \ldots, p_{id})$ and $\Phi^{-1}(\mathbf{p}_i) = (\Phi^{-1}(p_{i1}), \Phi^{-1}(p_{i2}), \ldots, \Phi^{-1}(p_{id}))$ for $i = 1, \ldots, t$ and $j = 1, \ldots, n_i$ where $\Phi(\cdot)$ is inverse of standard normal distribution function. Let $\boldsymbol{X}_{ij} = (X_{ij}^1, X_{ij}^2, \ldots, X_{ji}^d)$ and $X_{ij}^k = I(Y_{ij}^k \geq 0)$ for $1 \leq k \leq d$ where

$$\boldsymbol{Y}_{ij} = (Y_{ij}^1, \ldots, Y_{ij}^d) \sim N(\Phi^{-1}(\mathbf{p}_i), \Sigma_\rho). \tag{10}$$

Then X_{ij}^k, $k = 1, \ldots, d$, are Bernoulli (p_{ik}) and correlated. In this simulation, we set $t = 4$, $n_1 = n_2 = n_3 = n_4 = 25$ and $d = 250$.

Let $P(\boldsymbol{X}_{ij} \mid i = g)$ be the conditional probability of the vector $\boldsymbol{X}_{ij}$ given it came from the gth class. For example, for $\mathbf{X}_{ij} = (1, 0, \ldots, 1)$,

$$\begin{aligned} P(\boldsymbol{X}_{ij} \mid i = g) &= P(X_{ij}^1 = 1, X_{ij}^2 = 0, \ldots, X_{ij}^d = 1 \mid i = g) \\ &= P(Y_{ij}^1 \geq 0, Y_{ij}^2 < 0, \ldots, Y_{ij}^d \geq 0 \mid i = g) \end{aligned}$$

which requires the computation of a d-dimensional multivariate integral. We followed the algorithm in Genz (1992). Since our d is much larger than the values studied by Genz (1992), we recalculate the integral in a few cases in two other ways.

In both of them, we used the algorithm in Ihm (1959) to reduce the problem to a one dimensional integral,

$$P(\mathbf{X}_{ij} \mid i = g) \\ = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \prod_{k=1}^{d} \Phi\left(\frac{-\Phi(p_{ik}) + \sqrt{\rho}z}{\sqrt{1-\rho}}\right)^{1-x_{ij}^k} \left[1 - \Phi\left(\frac{-\Phi(p_{ik}) + \sqrt{\rho}z}{\sqrt{1-\rho}}\right)\right]^{x_{ij}^k} e^{-\frac{1}{2}z^2}\, dz$$

and then either used numerical integration or Laplace approximation (See Barndorff-Nielson and Cox, 1989). Computations by the two methods agree well. Based on this computation, we estimate Bayes error from the following simulation and three computations gave fairly consistent values of Bayes error. We assume prior probabilities are uniform (i.e., $P(i) = \frac{1}{t}$ for $1 \leq i \leq t$).

Monte Carlo simulation for Bayes error

1. Repeat the following for $g = 1, \ldots, t$.
   (a) Given class g, generate N samples, $\boldsymbol{x}_{gj}$, where $1 \leq j \leq N$ by generating $\boldsymbol{y}_{gj}$ from (10).
   (b) For each $\boldsymbol{x}_{gj}$, compute conditional probability, $p(\boldsymbol{x}_{gj} \mid i)$ for $i = 1, \ldots, t$.
   (c) If $g \neq \text{argmax}_{1 \leq i \leq t} p(\boldsymbol{x}_{gj} \mid i)$, then $\boldsymbol{x}_{gj}$ is misclassified.
2. Misclassification error is approximated by the proportion of misclassified samples:

$$\text{Misclassification error} = \frac{\text{the number of misclassified samples}}{\text{the total number of samples}}.$$

For a large N, we expect the above misclassification error is close to Bayes error.

Optimal Bayes error depends on the combination of treatment probabilities $\mathbf{p}_i$, $i = 1, \ldots, 4$ and ρ. We studied the cases $\rho = 0.0, 0.3, 0.6, 0.9$ and two different

Table 1

Comparison of boosting with the methods in [W] for low Bayes errors. LDF is variable selection using **W** in (4) and D^2 is in (3). We use $\alpha = 0.05$ and 10,000 permutations in D^2 and LDF. Each number represents the mean misclassification errors or numbers of selected variables for 100 test sets, and $(\cdot)$ represents standard deviation of them. I.B. represents Independent Bernoulli rule in (5)

ρ	0	0.3	0.6	0.9
		Misclassification error		
Boosting	0.0126 (0.0148)	0.0143 (0.0145)	0.0262 (0.0213)	0.0516 (0.0224)
I.B. (LDF)	0.0684 (0.0486)	0.0678 (0.0438)	0.0624 (0.0524)	0.0681 (0.0660)
I.B. (D^2)	0.1518 (0.0668)	0.1614 (0.0710)	0.1493 (0.0794)	0.1262 (0.0944)
Bayes error	0.0000	0.0005	0.0081	0.0340
		Number of selected variables		
Boosting	15.17 (2.38)	17.81 (3.32)	16.76 (2.82)	14.19 (2.82)
LDF	24.39 (2.90)	16.42 (2.40)	17.50 (2.96)	20.85 (4.92)
D^2	73.85 (3.00)	4.92 (3.97)	73.53 (4.37)	74.73 (9.51)

combinations of probability vectors $\{\mathbf{p}_i, 1 \le i \le 4\}$. If the probability vectors in $\{\mathbf{p}_i, 1 \le i \le 4\}$ are close to each other, then classification becomes difficult, indicating Bayes errors are large. From simulation, Bayes errors of one combination $\{\mathbf{p}_i, 1 \le i \le 4\}$ are from 0–0.034 for different ρ's. But another combination of $\{\mathbf{p}_i, 1 \le i \le 4\}$ produces large Bayes errors which vary from 0.035–0.24.

We generate 100 training data sets for each ρ and consider ten-fold cross-validation. Based on ten-fold cross-validation, we compare the results of boosting with those in Wilbur (2002). We use them on a test set, which is possible for a simulated example.

From Table 1, in the case of the low Bayes errors (i.e., Bayes errors from 0–0.034), it is clear that boosting has a much lower misclassification error than the methods in [W] as indicated by the fact that misclassification errors of boosting are much closer to Bayes error than those of the methods in [W]. The number of variables selected by boosting is smaller than that by D^2, but similar to that of LDF for each ρ. Considering misclassification errors and the number of selected variable, boosting performs better than D^2 and LDF for the case of low Bayes error. Since boosting performs well both with respect to control of misclassification error and the number of selected variables, we did not try out any of the three suggested methods.

Table 2 shows the misclassification errors and the number of selected variables for the high Bayes error case (i.e., Bayes errors from 0.035–0.24). The misclassification errors of boosting are smallest among all methods, but the largest number of variables are selected. Method 1 removes half the variables of boosting

Table 2

Comparison boosting with the methods in [W] when high Bayes errors. LDF is variable selection using $\mathbf{W}$ in (4) and D^2 is in (3). In D^2 and LDF, we use $\alpha = 0.05$ for selection of significant variables and 10,000 permutations for reference distribution. Each number represents the mean number of misclassification errors or mean numbers of selected variables for 100 test sets, and $(\cdot)$ represents standard deviation of them. I.B. represents Independent Bernoulli rule in (5)

ρ	0	0.3	0.6	0.9
	Misclassification error			
I.B. (LDF)	0.3120 (0.1271)	0.6263 (0.1448)	0.6683 (0.1415)	0.6300 (0.1865)
I.B. (D^2)	0.2589 (0.0565)	0.2727 (0.0574)	0.3607 (0.0463)	0.4465 (0.0395)
Boosting	0.1363 (0.0362)	0.1413 (0.0381)	0.2469 (0.0423)	0.3563 (0.0234)
Method 1	0.1712 (0.0421)	0.1794 (0.0454)	0.2837 (0.0478)	0.3682 (0.0273)
Method 2 (D^2)	0.2597 (0.0565)	0.2755 (0.0603)	0.3607 (0.0463)	0.4465 (0.0395)
Method 3 (D^2)	0.2453 (0.0582)	0.2436 (0.0588)	0.2963 (0.0619)	0.3926 (0.0495)
Method 3 (LDF)	0.4419 (0.0904)	0.6056 (0.0979)	0.6586 (0.0913)	0.6977 (0.0716)
Bayes error	0.035	0.051	0.096	0.242
	Number of selected variables			
LDF	8.58 (3.25)	2.28 (1.99)	1.55 (1.72)	2.55 (4.31)
D^2	31.59 (3.39)	31.85 (6.30)	33.34 (9.41)	36.14 (14.06)
Boosting	62.97 (9.71)	61.35 (8.40)	53.00 (9.66)	40.64 (8.23)
Method 1	33.10 (7.52)	31.55 (6.80)	27.00 (7.01)	22.13 (5.88)
Method 2 (D^2)	31.16 (3.49)	31.36 (6.47)	33.34 (9.41)	36.14 (14.06)
Method 3 (D^2)	28.96 (6.62)	29.67 (7.71)	31.93 (10.91)	34.71 (15.13)
Method 3 (LDF)	8.42 (3.30)	2.15 (1.71)	1.44 (1.59)	1.79 (2.85)

with only small increase of misclassification errors. Compared with D^2, Method 1 selects a similar number of variables, but achieves a lower misclassification error than that of D^2. However, Method 2 (D^2) and Method 3 (D^2) do not greatly change the misclassification error and the number of variables of D^2. The misclassification error based on LDF is very large since LDF selects too small a number of variables, Method 3 (LDF) also does not achieve satisfactory misclassification error for the same reason.

5.2. *Three Real Data Sets*

Data 1, 2, and 3 are generated from the type of DNA fingerprints which have the same type, (i.e., binary data), but the plant rhizpsphere from which samples were collected differs from one another. Data set 1 in [W] represents corn rhizosphere communities, whereas data sets 2 and 3 are soybean-associated communities. The details of the study from which data set 1 were generated are discussed in greater detail in [W]. Briefly, the objective of the study is to determine if rhizosphere microbial communities differ in plants grown under four different agronomic treatments; combinations of tilling (or not) and rotation of crops (or not). If differences are found, then the objective is to identify members of the community (variables in these data sets) that best explain these differences. The study of significant members of the microbial community under these different treatments may provide insight into the link between agronomic management practice and the plant-associated microbial community. The objective of the study for data sets 2 and 3 are the same with the exception of the crop plants (soybean instead of corn) which are being used. Table 3 indicates that for data from [W], Method 1 correctly classifies most samples (86) based on leave-one-out cross-validation. Figure 1 shows

Table 3
Number of correctly classified samples by three methods and methods for data in [W]. I.B. (Independent Bernoulli) and Logistic represent classification rules using (5) and (6), respectively

Rule	Number of variables	Correct samples
I.B. (Full)	84	72
Logistic (Full)	84	52
I.B. (LDF)	3	68
Logistic (LDF)	3	65
I.B. (D^2)	19	79
Logistic (D^2)	19	72
Boosting	37	84
Method 1	26	86
Method 2 (D^2)	23	83
Method 2 (LDF)	3	68
Method 3 (D^2)	16	81
Method 3 (LDF)	3	66
Total		89

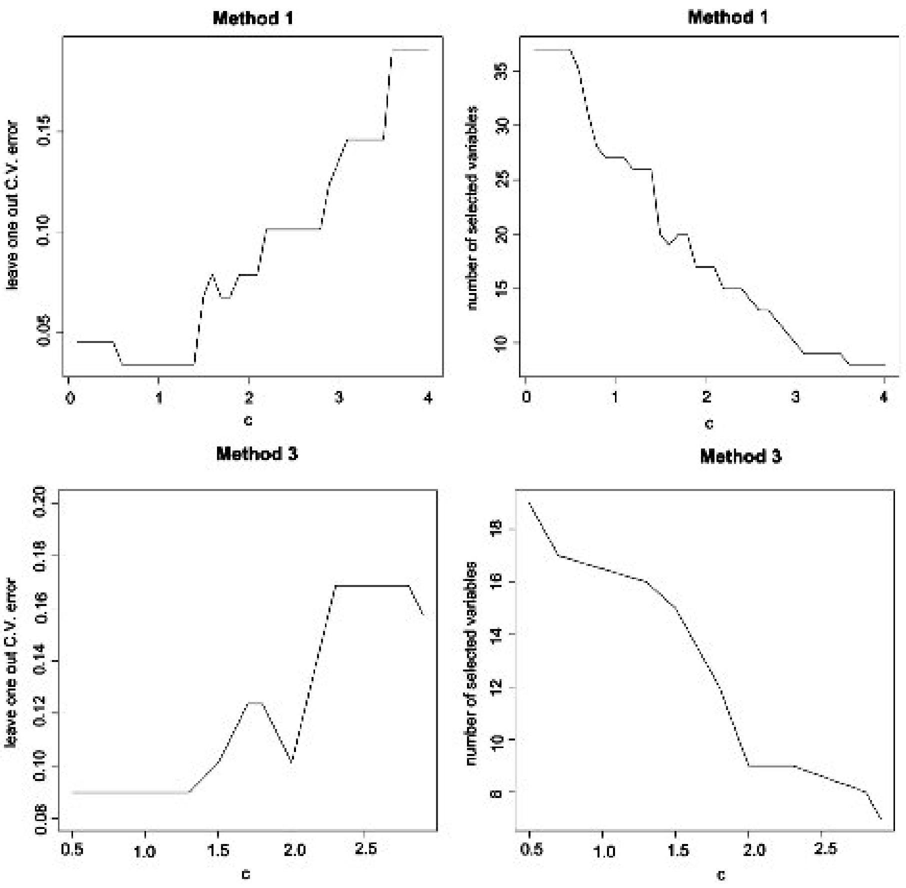

Figure 1. Graphs of cross-validation errors (i.e., misclassification error estimated by cross-validation) and the number of selected variables for data in [W]. Upper two graphs represent Method 1, lower two graphs represent Method 3 with D^2. Graphs on the left-hand side represent cross-validation error against thresholding parameter, c, those on the right represent the number of variables against c.

how cross-validation error and the number of selected variables change as a function of the threshold parameter for Methods 1 and 3.

Application of the methods to the new data sets, 2 and 3, demonstrated similar trends. The data represent samples collected in two different years, where data set 2 is from 1999 and data set 3 from 2000. In total there were 336 samples, which were divided into a training set and test set, 252 (4×63) and 84 (4×21) samples, respectively.

In application to these data, the LDF method does not select any variable, so Methods 2 and 3 with LDF are not considered. The D^2 method selects only a small number of variables. Due to the small size of these subsets, employing Method 3 is not affected by thresholding, it is better than Method 2 but inferior to Method 1. Figure 2 shows how Method 1 works with a threshold parameter. The number of variables initially selected by boosting alone decreases quickly as the threshold parameter increases, but the test and cross-validation error rates do not change

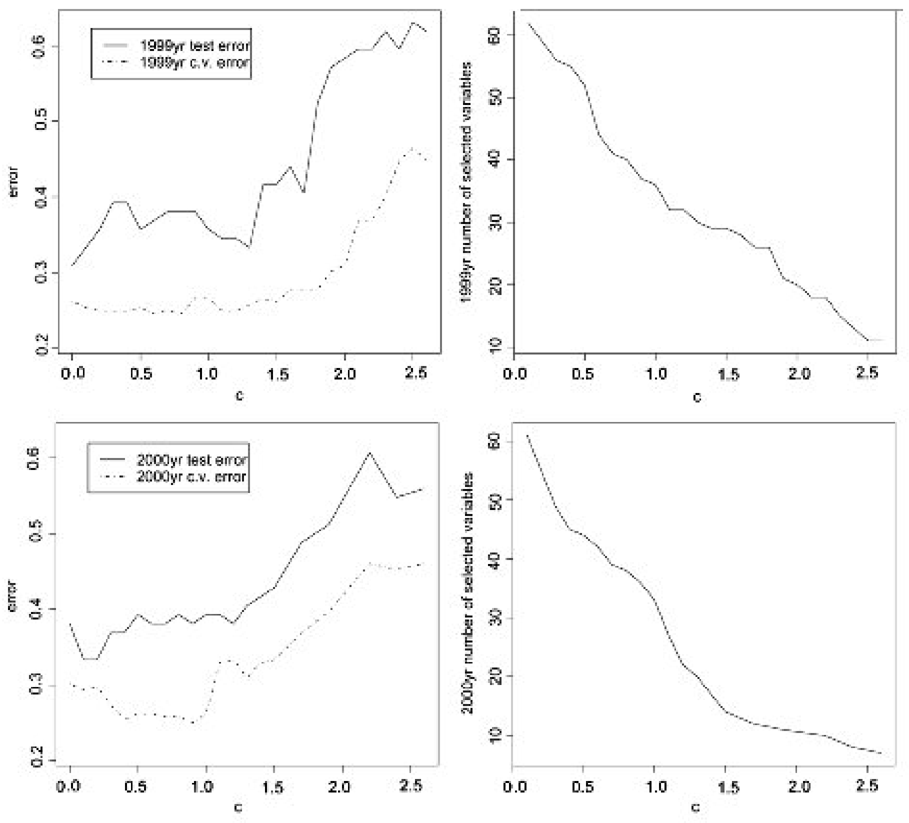

Figure 2. Number of selected variables, misclassification errors (i.e., misclassification errors for test sample), and cross-validation errors of Method 1 for data from each year. (Upper left) Represents the number of selected variables against threshold c for data from the 1999 season. (Upper right) Represents the misclassification error and cross-validation error of Method 1 for data from the 1999 season. The two lower graphs represent data from the 2000 season.

substantially for $0 < c < 1$. Thus, a large number of variables can be excluded by Method 1 with little change in the misclassification error rate.

The results using the data sets 2 and 3 had higher cross-validation errors than data set 1. These differences likely arise from the original data sources. Other measurements made from the two crops indicated there was less treatment effect on growth parameters for soybean relative to corn. Thus the statistical analysis reflects the biological trends that were noted.

6. Discussion

We have investigated boosting and its three methods along with the methods in [W] for classification and selection of binary variables in high dimensional real, and simulated examples. Boosting generally achieves low misclassification error in all our cases, but it tends to use many variables. To reduce the number of variables, Method 1 is effective for variable selection with small change of misclassification error. Method 2 does not greatly improve the results of methods in [W]. This

Table 4
Summary of D^2, boosting and three methods for data from 1999 and 2000 seasons

	c.v.	Misclassification error	Number of variables
		1999 yr	
Logistic (D^2)	0.4920	0.5476	12
I.B. (D^2)	0.4405	0.7619	12
Boosting	0.2619	0.3095	62
Method 1	0.2500	0.3452	29
Method 2 (D^2)	0.4206	0.7142	11
Method 3 (D^2)	0.4405	0.5595	12
		2000 yr	
Logistic (D^2)	0.4682	0.4404	13
I.B. (D^2)	0.4444	0.6786	13
Boosting	0.3016	0.3810	61
Method 1	0.2500	0.3810	36
Method 2 (D^2)	0.4642	0.7142	11
Method 3 (D^2)	0.4008	0.5000	13

means that change of empirical distribution alone may not improve non-boosting type methods such as [W]. The performance of Method 3 depends greatly on the preselected variables.

The thresholding method can be extended to selection of continuous variables.

As in [W], we have tried to validate selection of variables by examining the performance of a classifier using these variables. While our method of thresholding coupled with validation reduced the numbers of variables substantially, the selected subsets were probably still too large from the standpoint of identifying a few of the variables in the real life examples as scientifically important. We believe further progress in this direction would require better understanding of functions and effects of different microbial communities.

Acknowledgment

We thank an anonymous refree whose careful reading and suggestions have led to an improvement presentation.

References

Bahadur, R. R. (1961). A representation of the joint distribution of responses to *n* dichotomous items. In: *Studies in Item Analysis and Prediction*. Solomon, H., ed. Stanford Mathematical Studies in the Social Sciences VI. Stanford University Press.

Barndorff-Nielson, O., Cox. D. R. (1989). *Asymptotic Techniques for Use in Statistics*. Chapman and Hall.

Breiman, L. (2004). Population theory for boosting ensembles. *Ann. Statist.* 32:1–11.

Dudoit, S., Fridlyand, J., Speed, T. (2002). Comparison of discrimination methods for the classification of tumors using gene expression data. *J. Amer. Stat. Assoc.* 97:77–87.

Emrich, L. J., Piedmonte, M. R. (1991). A method for generating high-dimensional multivariate binary variates. *Amer. Statist.* 45:302–304.

Freund, Y., Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comp. Syst. Sci*. 119–139.

Freund, Y., Schapire, R. E. (2004). Discussion of consistency in boosting. *Annals Stat*. 32:113–117.

Friedman, J. H., Hastie, T., Tibshirani, R. (2000). Additive logistic regression: A statistical view of boosting. *Annals Statist*. 28:337–407.

Genz, A. (1992). Numerical computation of multivariate normal probabilities. *J. Comp. Graph. Stat*. 1:141–150.

Hastie, T., Tibshirani, R., Friedman, J. (2001). *The Elements of Statistical Learning: Data Mining, Inference and Prediction*. Spring-Verlag.

Ihm, P. (1959). Numerical evaluation of certain multivariate normal integrals. *Sankhya* 21:363–366.

Jiang, W. (2004). Process consistency for Adaboost. *Annals Statist.* 32:13–29.

Bartlett, P., Freund, Y., Lee, W. S., Schapire, R. E. (1998). Boosting the margin: A new explanation for the effectiveness of voting methods. *Annals Stat*. 26:1651–1686.

Lugosi, G., Vayatis, N. (2004). On the Bayes-risk consistency of regularization boosting methods. *Annals Stat*. 32:30–55.

Margineantu, D. D., Dietterich, T. G. (1997). Pruning adaptive boosting. In: Proceedings of the 14th International Conference on Machine Learning, pp. 211–218.

Nakatsu, C. H., Torsvik, V., Ovreas, L. (2000). Soil community analysis using denaturing gradient gel electrophoresis (DGGE) profiles of 16S rDNA PCR products. *Soil. Sci. Soc. Amer. J*. 64:1382–1388.

Rosset, S., Zhu, J., Hastie, T. (2004). Boosting as a regularized path to a maximum margin classifier. *J. Machine Learning Research* 5:941–973.

Westfall, P. H., Young, S. S. (1993). *Resampling-Based Multiple Testing: Examples and Methods for p-value Adjustment*. Wiley.

Wilbur, J. D. (2002). Variable Selection Methodology for High-Dimensional Multivariate Binary Data with Application to Microbial Community DNA Fingerprint Data. PhD thesis. Purdue University.

Wilbur, J. D., Ghosh, J. K., Nakatsu, C. H., Brouder, S. M., Doerge, R. W. (2002). Variable selection in high-dimensional multivariate binary data with application to the analysis of microbial community DNA fingerprints. *Biometrics* 58:378–386.

Zhang, T. (2004). Statistical behavior and consistency of classification methods based on convex risk minimization. *Annals Stat.* 32:56–85.