The Korean Communications in Statistics
Vol. 13 No. 2, 2006, pp. 449-466

Pruning the Boosting Ensemble of Decision Trees

Young Joo Yoon[1] and Moon Sup Song[2]

Abstract

We propose to use variable selection methods based on penalized regression for pruning decision tree ensembles. Pruning methods based on LASSO and SCAD are compared with the cluster pruning method. Comparative studies are performed on some artificial datasets and real datasets. According to the results of comparative studies, the proposed methods based on penalized regression reduce the size of boosting ensembles without decreasing accuracy significantly and have better performance than the cluster pruning method. In terms of classification noise, the proposed pruning methods can mitigate the weakness of AdaBoost to some degree.

Keywords : AdaBoost; Penalized regression; Cluster pruning; LASSO; SCAD; Pruning ensemble.

1. Introduction

Boosting is one of the most powerful methods for combining classifiers to improve prediction accuracy. The adaptive boosting algorithm AdaBoost (Freund and Schapire, 1997) with the decision-tree algorithm C4.5 (Quinlan, 1993) or CART (Breiman et al., 1984) has been extensively studied. The AdaBoost algorithm is an ensemble method which generates a set of classifiers and votes for the prediction by combining the generated classifiers. The idea of boosting is to construct classifiers sequentially by focusing the base learning algorithm on the training examples that have been misclassified by previous classifiers. Then boosting combines the "weak" classifiers to produce a "strong" committee. The boosting algorithm AdaBoost has been called the best off-the-shelf classifier because of its superior performance (see, e.g. Breiman, 1998, and Tamon and Xiang, 2000).

Despite its good performance, AdaBoost with decision trees requires a large amount of memory space to store the constructed trees. For example, Margineantu and Dietterich (1997) observed that for the Frey-Slate letter recognition dataset, AdaBoost requires about 200 trees to achieve a reasonable accuracy. However, each tree requires 295 Kbytes of memory, and so the final ensemble of 200 trees requires

1) Ph.D, Department of Statistics, Seoul National University, Seoul 151-742, Korea.
Correspondence : youngjoo@freechal.com
2) Professor, Department of Statistics, Seoul National University, Seoul 151-742, Korea.

59 Mbytes. They investigated if it is possible to discard some of the constructed trees while obtaining nearly the same levels of performance as the entire set. The results show that the ensemble produced by AdaBoost can be radically pruned in some datasets. They called this "pruning the ensemble."

Margineantu and Dietterich (1997) proposed five methods of pruning the boosting ensemble. Among the five methods, the kappa pruning method has shown good performance. The idea behind this method is based on the assumption that boosting works by constructing a diverse, yet accurate, collection of classifiers. To choose diverse classifiers they used the kappa statistics to measure the disagreement of two classifiers. The kappa pruning method continues to choose pairs of classifiers with the lowest agreement until a pre-specified number of classifiers are chosen. Tamon and Xiang (2000) proposed a slight modification to the kappa pruning method called weight shifting. They shifted the voting weights of pruned trees onto its unpruned neighbors.

Lazarevic and Obradovic (2001) proposed a method for pruning classifiers from ensembles based on a clustering approach. This method partitions the set of classifiers into subsets containing similar classifiers by using the standard k -means clustering algorithm. Then redundant classifiers from each cluster are eliminated according to their accuracy on the validation set or according to their disagreement with the retained ones.

Treating the trees in an ensemble as variables in a linear regression model, we may apply the variable selection techniques to choose some trees in the ensemble. There are a number of variable selection methods in linear regression, and we are mainly interested in regularization or penalized regression methods which include ridge regression, LASSO (Least Absolute Shrinkage and Selection Operator) and SCAD (Smoothly Clipped Absolute Deviation). Increasing the regularization parameter, the solution to the LASSO or SCAD problem tends to be sparse. That is, only a small fraction of all possible trees will be relevant in approximating the prediction rule.

This paper consists of three main parts : a review of AdaBoost and penalized regression, pruning the boosting ensemble of decision trees, and a summary and concluding remarks. In Section 2, we briefly introduce AdaBoost. Also we review some well-known penalized regression methods such as the LASSO and SCAD methods. In Section 3, we propose pruning boosting ensemble methods via penalized regression. To compare the proposed methods with other methods, experiments were performed on simulated datasets and also on real datasets. The performances of various methods are compared in terms of test error and the number of selected classifiers. The test errors of the proposed methods are better than or similar to those of AdaBoost and cluster pruning. But the number of selected trees is significantly smaller than those of the boosting ensemble and cluster pruning in many cases. Especially with classification noise the pruning methods based on penalized regression can significantly improve the weakness of AdaBoost. Finally, a summary and concluding remarks are provided in Section 4.

2. AdaBoost and Penalized Regression

2.1 AdaBoost

Boosting is one of the most powerful learning algorithms. The boosting algorithm applies a "weak" or "base" learning algorithm repeatedly to a different distribution (or a newly weighted training set) to produce a new weak classifier, and combines these weak classifiers to construct a "strong" prediction rule. The most popular boosting algorithm is AdaBoost proposed by Freund and Schapire (1997).

In this paper we consider only a two class problem.Let $\{(x_{1,}y_1),(x_2,y_2),\cdots,(x_{n,}y_n)\}$ be a training set, where x is a vector of predictor variables and y is a two-class response variable such that $y\in\{-1,1\}$. A weak classifier (learner) is one whose error rate is slightly better than random guessing. We sequentially apply the weak classification algorithm repeatedly to the training set with modified weights, producing a sequence of weak classifiers $h_t(x), t=1,2,\cdots,T$. The final prediction rule is defined as follows:

$$H(x)=\mathrm{sign}(\sum_{t=1}^{T}\alpha_t h_t(x)),$$

where $\alpha_1,\alpha_2,\cdots,\alpha_T$ are the weights assigned to the weak classifiers h_t. Thus, each α_t indicates the importance of h_t. For each iteration $t=1,2,\cdots,T$, the weights assigned to examples are individually modified and the classification algorithm is reapplied to the updated training examples. At each iteration the weights of examples misclassified by h_t are increased and those of correctly classified examples are decreased. Thus each classifier is forced to concentrate on the training examples that are misclassified by the previous classifier. <Figure 1> shows the details of AdaBoost.M1. In this paper AdaBoost.M1 is referred to AdaBoost for simplicity.

```
1. Initialize the observation weights $w_i = 1/n,\ i = 1,2,\cdots,n$.
2. For $t$=1 to $T$
   (a) Fit a classifier $h_t(x)$ to the training data using weights $w_i$.
   (b) Compute $\epsilon_t = (\sum_{i=1}^{n} w_i I(y_i \neq h_t(x_i))) / \sum_{i=1}^{n} w_i$.
   (c) Compute $\alpha_t = \log((1-\epsilon_t)/\epsilon_t)$.
   (d) Set $w_i \leftarrow w_i \cdot \exp[\alpha_t I(y_i \neq h_t(x_i))],\ i = 1,2,\cdots,n$
3. Output $G(x) = \mathrm{sign}\left[\sum_{t=1}^{T} \alpha_t h_t(x)\right]$
```

<Figure 1> AdaBoost.M1 algorithm

2.2 Penalized Regression

Variable selection is an important topic in linear regression analysis. By retaining a subset of the predictors and discarding the rest (for example, stepwise forward selection or backward elimination), the reduced regression model is more interpretable and sometimes reduces the prediction error.

Although subset selection methods are practically useful, they have several drawbacks. The most severe drawback is their lack of stability. Since the variables are either retained or discarded, subset selection methods often show high variance and do not reduce the prediction error of the full model. Shrinkage methods are useful to overcome these difficulties, since they do not suffer as much from high variability (Hastie et al. 2001, Chapter 3).

2.2.1 LASSO

Consider the linear regression model

$$y = X\beta + \epsilon,$$

where y is an $n \times 1$ vector and X is an $n \times p$ matrix.

The LASSO (Least Absolute Shrinkage and Selection Operator) was proposed by Tibshirani (1996). The LASSO is a shrinkage method like the ridge regression, using an L_1 penalty instead of the L_2 penalty in the ridge regression. Thus the LASSO estimator is defined by

$$\hat{\beta}^{\mathrm{LASSO}} = \mathrm{argmin}_{\beta} \left\{ \sum_{i=1}^{n} (y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\}.$$

Because of the L_1 penalty the solution is not linear in y, and usually quadratic programming methods are used to calculate the LASSO estimate. As the regularization parameter λ increases, some of the coefficients tend to be exactly zero.

2.2.2 SCAD

Fan and Li (2001) proposed a variable selection method based on a SCAD (Smoothly Clipped Absolute Deviation) penalty function. As a motivation for the SCAD penalty they claimed that a good penalty function should result in an estimator with the following three properties:

- *Unbiasedness* : The resulting estimator is nearly unbiased when the true unknown parameter is large to avoid unnecessary modelling bias.
- *Sparsity* : The resulting estimator is a thresholding rule, which automatically sets small estimated coefficients to zero to reduce model complexity.
- *Continuity* : The resulting estimator is continuous in data to avoid instability in model prediction.

For simplicity, we only consider the following penalized least square problem

$$\frac{1}{2}(z-\theta)^2 + p_\lambda(|\theta|).$$

To satisfy the conditions for unbiasedness, sparsity and continuity, Fan and Li (2001) proposed the SCAD penalty function defined by

$$p_\lambda(|\theta|) = \lambda \begin{cases} |\theta| & , \ 0 \le |\theta| \le \lambda \\ \dfrac{1}{2(a-1)\lambda}(\theta^2 - 2a\lambda|\theta| + \lambda^2), & \lambda \le |\theta| \le a\lambda \\ \dfrac{1}{2}(a+1)\lambda & , \ |\theta| > a\lambda \end{cases}$$

for some $a > 2$. The derivative of the SCAD function is given by

$$p_\lambda{}'(\theta) = \lambda\left\{ I(\theta \le \lambda) + \frac{(a\lambda - \theta)_+}{(a-1)\lambda} I(\theta > \lambda) \right\} \text{ for some } a > 2 \text{ and } \theta > 0,$$

where $(t)_+ = 0$ for $t < 0$ and $(t)_+ = t$ for $t \ge 0$. The solution to the SCAD penalty is given by

$$\hat{\theta}=\begin{cases} sign(z)(|z|-\lambda)_{+} & , |z| \leqq 2\lambda \\ \frac{1}{a-2}((a-1)z-sign(z)a\lambda) & , 2\lambda < |z| \leqq a\lambda \\ z & , |z| > a\lambda \end{cases}$$

Fan and Li (2001) also proposed an algorithm for minimizing penalized general loss via local quadratic approximations. Denote a loss function by $l(\beta)$. Then the penalized general loss can be written in a unified form as

$$l(\beta)+n\sum_{j=1}^{p}p_{\lambda}(|\beta_j|).$$

Under some mild conditions, both terms can be locally approximated by a quadratic function (see Fan and Li, 2001). Therefore the minimization problem of the unified form can be reduced to a quadratic minimization problem and the Newton-Raphson algorithm can be used.

3. Pruning the Boosting Ensemble of Decision Trees

3.1 Proposed Penalized Methods

To formulate the penalized methods, let $h_1, h_2, \cdots, h_T$ be the classifiers produced by AdaBoost based on a training set $\{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$. Let α_t be the weight corresponding to h_t, $t=1,\cdots,T$. Then the final prediction rule is given by

$$H(x) = sign\left(\sum_{t=1}^{T}\alpha_t h_t(x)\right).$$

To estimate the coefficients $\beta=(\beta_1, \cdots, \beta_T)^T$ we want to minimize a given loss function with a penalty. That is, the penalized estimate $\hat{\beta}$ is defined by

$$\hat{\beta}= \text{argmin}_{\beta}\left\{L(y,F)+n\sum_{t=1}^{T}p_{\lambda}(|\beta_t|)\right\},$$

when $L(y,F(x))$ is a loss function and $p_{\lambda}(\cdot)$ is a penalty function. We are interested in the following loss functions.

$L(y,F)=\exp(-yF)$	exponential loss
$L(y,F)=\log(1+\exp(-yF))$	binomial loglikelihood loss

We know that LASSO and SCAD are good penalty functions to choose a subset of coefficients without sacrificing the prediction error too much. Thus we propose to use these two penalty functions in pruning the ensemble. The unified algorithm suggested by Fan and Li (2001), compute $\hat{\beta}$ for SCAD. The estimate $\hat{\beta}$ based on LASSO can be computed by using the forward stagewise algorithm proposed by Rosset et al. (2004). The weight α_t is expected to be a good initial value β_{t0} for each $t=1,\cdots,T$. The algorithms of the proposed penalized methods are briefly summarized in <Figure 2> - <Figure 3>.

```
1. Perform the Adaboost with a base learner.
2. Let $H=(h_1,\cdots,h_T)$ where $h_i=(h_i(x_1),h_i(x_2),\cdots h_i(x_n))^T$,
   $i=1,\cdots,T, h_i\in\{-1,1\}$.
3. Repeat
   (a) Initialize $\hat{\beta}_0=(\alpha_1,\cdots,\alpha_T)^T$.
   (b) Until ($\hat{\beta}_j$ is converged)
       $\hat{\beta}_j=\hat{\beta}_{j-1}-\{\nabla^2L(y,F)+n\Sigma_\lambda(\hat{\beta}_{j-1})\}^{-1}\{\nabla L(y,F)+nU_\lambda(\hat{\beta}_{j-1})\}$
       where $\nabla L(y,F)=\frac{\partial L(y,F)}{\partial\hat{\beta}_{j-1}},\nabla^2L(y,F)=\frac{\partial^2L(y,F)}{\partial\hat{\beta}_{j-1}\partial\hat{\beta}_{j-1}^T}$,
       $\Sigma_\lambda(\hat{\beta}_{j-1})=\mathrm{diag}\{p'_\lambda(|\hat{\beta}_{j-1,1}|),\cdots,p'_\lambda(|\hat{\beta}_{j-1,T}|)\}$,
       $U_\lambda(\hat{\beta}_{j-1})=\Sigma_\lambda(\hat{\beta}_{j-1})\hat{\beta}_{j-1}$, $\hat{\beta}_{j-1}=(\hat{\beta}_{j-1,1},\cdots,\hat{\beta}_{j-1,T})^T$.
```

<Figure 2> SCAD method for classifier pruning of AdaBoost

```
1. Perform the Adaboost with a base learner.
2. Let $H=(h_1,\cdots,h_T)$ where $h_i=(h_i(x_1),h_i(x_2),\cdots h_i(x_n))^T$
   $i=1,\cdots,T, h_i\in\{-1,1\}$.
3. $\beta_0=0$, where $\beta_0=(\beta_{0,1},\cdots,\beta_{0,T})^T$
4. For $m=1$ to $M$
   (a) Let $F=H\beta_{m-1}$ where $F=(F_1,\cdots,F_n)^T$
   (b) Set $w_i=\frac{\partial L(y_i,F_i)}{\partial F_i}$, $i=1,2,\cdots,n$.
   (c) Identify $j_0$ which maximizes $|\sum_i w_ih_j(x_i)|$.
   (d) Set $\beta_{t,j_0}=\beta_{t-1,j_0}-\epsilon\cdot\mathrm{sign}(\sum_i w_ih_{j_0})$ and $\beta_{t,k}=\beta_{t-1,k}$, $k\neq j_0$.
```

<Figure 3> LASSO penalty method for classifier pruning of AdaBoos

3.2 Simulation Studies

To compare the proposed methods with other methods, numerical experiments are performed with simulated data sets.

3.2.1 Data Sets Description

For numerical comparison, the following data sets that were introduced in Breiman (1998) are used.

- Twonorm : This is 20-dimension, 2-class data. Each class is drawn from a multivariate normal distribution with unit covariance matrix. Class 1 has mean $(a, a, \cdots, a)$ and class 2 has mean $(-a, -a, \cdots, -a)$, where $a = 2/\sqrt{20}$.

- Threenorm : This is 20-dimension, 2-class data. Class 1 is drawn with equal probability from a unit multivariate normal with mean $(a, a, \cdots, a)$ and from a unit multivariate normal with mean $(-a, -a, \cdots, -a)$. Class 2 is drawn from a unit multivariate normal with mean $(a, -a, a, -a, \cdots, -a)$ where $a = 2/\sqrt{20}$.

The size of the training set for each model is 200. We generate a validation set of size 200 to determine the regularization (tuning) parameter which minimizes the validation error. To estimate the accuracy of the pruned ensemble we use a test set of size 1000.

3.2.2 Base Learning Algorithm and Selection Methods

In simulation studies, we used CART as the base learner without pruning. We constructed classification trees of depth 1 and 3. Since the simulation results are almost the same as those of depth 3 for larger trees, we report only two cases. The number of AdaBoost iterations is 200, that is, we use 200 classification trees in each ensemble. The Monte Carlo simulation was performed 100 times to compute the average test error and average size of the pruned ensemble.

The pruning ensemble methods based on LASSO and SCAD are implemented with exponential loss. We also implement penalized methods with binomial loglikelihood loss functions. But the test errors of exponential loss and binomial loglikelihood loss make little difference. So we use only penalized methods with exponential loss. The unsupervised cluster pruning method needs a validation set to compute the accuracy of each classifier in order to eliminate redundant classifiers in each cluster. Thus 20% of the training set was used as a validation set, which is different from the validation set of size 200 to determine the regularization parameter. The simulations were performed on a personal computer with a Pentium IV 1.7GHz processor using R (version 1.9.0).

3.2.3 Results of Simulation Studies

Numerical comparison results are summarized in Table 1 - Table 2. The pruning methods reduce the size of classifiers, while achieving a similar level of performance as the entire boosting ensemble. In these limited numerical comparisons, the pruning methods are effective in terms of the size of classifiers and test error. The pruning methods via penalized regression are better than cluster pruning in most cases. Especially for the decision trees with depth equal to 1, the penalized regression methods are much better than cluster pruning. For twonorm data, SCAD performs slightly better than LASSO. Average test errors are similar, but the sizes of classifiers of SCAD are slightly smaller when compared with LASSO. However, in the case of threenorm data, these trends are reversed, i.e. the LASSO method is better than SCAD. The SCAD method slightly increases test errors.

<Table 1> Results for twonorm data

Method	Ave. Test Error	Ave. Size
	Depth=1	
AdaBoost	0.0556 (0.0083)*	200.00
SCAD	0.0597 (0.0111)	57.65
LASSO	0.0608 (0.0095)	70.14
Cluster Pruning Method	0.0637 (0.0129)	174.99
	Depth=3	
AdaBoost	0.0467 (0.0082)	200.00
SCAD	0.0545 (0.0079)	63.81
LASSO	0.0580 (0.0103)	95.80
Cluster Pruning Method	0.0558 (0.0114)	77.02

*The values in parentheses indicate standard errors of test error.

<Table 2> Results for threenorm data

Method	Ave. Test Error	Ave. Size
	Depth=1	
AdaBoost	0.2247 (0.0180)*	200.00
SCAD	0.2605 (0.0154)	61.25
LASSO	0.2345 (0.0166)	49.54
Cluster Pruning Method	0.2455 (0.0232)	181.78

<Table 2> Continued

	Depth=3	
AdaBoost	0.1946 (0.0152)*	200.00
SCAD	0.2418 (0.0341)	72.25
LASSO	0.2008 (0.0173)	100.05
Cluster Pruning Method	0.1937 (0.0152)	164.20

*The values in parentheses indicate standard errors of test error.

3.3 Real Data Analysis

The real datasets used in this section were obtained from the University of California at Irvine (UCI) Machine Learning Repository (Blake and Merz, 1998). The datasets of this repository are often used by the machine learning community for empirical analyses.

3.3.1 Data

We used four datasets for our comparative study. The datasets are the Wisconsin Breast Cancer dataset, the Credit Card Application Approval dataset, the Ionosphere dataset, and the Sonar dataset. We summarize these datasets as follows:

- Wisconsin Breast Cancer Dataset (BREAST): Each observation (example) has one of 2 possible classes : benign, malignant. By using 10 integer predictors (e.g. clump thickness, uniformity of cell size and so on), they will attempt to decide whether the class is benign or malignant. The number of observations is 699, and the number of predictor variables is 10. Originally, this dataset contains 16 observations with a single missing value. However, we remove missing values in our experiments. Therefore the number of observations used in our analysis is 683.
- Credit Card Application Approval Dataset (CRX): This dataset concerns the approval of a credit card's issue. All attribute names and values have been changed to meaningless symbols to protect confidentiality of the data. The class values are + (credit card approval) and - (rejection). This dataset has 15 predictors. 37 observations have one or more missing values, and those observations with missing values are removed. Thus, we analysed this dataset using 653 observations.
- Ionosphere Dataset (IONO): In this dataset, we classify each observation by whether it is a "good" radar or not. There are 34 continuous predictors. There are no missing observation. The number of observations in this dataset is 351.
- Sonar Dataset (SONAR): This is the dataset for the classification of sonar signals. It contains 60 predictors which represent the energy within a particular

frequency band, integrated over a certain period of time. These predictors are used for classification of sonar signals ("R" is a rock, "M" a mine). 208 observations are contained in this dataset.

<Table 3> Summary of dataset

			Variables	
Dataset	Size	Classes	Numerical	Categorical
BREAST	683	2	10	0
CRX	653	2	6	9
IONO	351	2	34	0
SONAR	208	2	60	0

<Table 3> shows a summary of the four datasets. To compare the performances of various algorithms for these datasets, we used 5-fold cross validation. By using this method, regularization parameters or tuning parameters (e.g. threshold value in cluster pruning method) which minimize the validation error are decided.

3.3.2 Base Learning Algorithm and Selection Methods

The setting is similar to the one in Subsection 3.2. We used CART without pruning as the base learning algorithm. Instead we restrict the depth of the tree, to be 1 or 3. The number of AdaBoost iterations is 200. The proposed LASSO and SCAD methods with exponential loss function are compared with the cluster pruning method.

3.3.3 Results

The results on the datasets are given in <Table 4> - <Table 7>. The results on real datasets are similar to the results on simulated datasets. The pruning methods prune many (more than half) of the classifiers. Especially for the BREAST data with depth=1, the sizes are reduced to 1/10 of the 200 boosting classifiers without increasing the CV errors. For the other datasets, the ensemble sizes are reduced significantly while maintaining the same level of performances on the whole. For the SONAR dataset with depth=1, the SCAD method increases the CV errors while reducing the sizes. Also for the same dataset with depth=3, the size is not reduced very much. But the size is about 60% of the AdaBoost size, and the CV error is reduced, the proposed methods are acceptable. These results show that the pruning methods based on penalized regression are successful, therefore it is useful to apply the pruning methods to the AdaBoost procedure.

<Table 4> Results for BREAST dataset

	Depth=1		Depth=3	
Method	CV Error	Size	CV Error	Size
AdaBoost	0.0425 (0.0077)*	200	0.0366 (0.0072)	200
SCAD	0.0337 (0.0069)	23	0.0337 (0.0069)	43
LASSO	0.0366 (0.0072)	17	0.0351 (0.0070)	58
Cluster Pruning Method	0.0425 (0.0077)	183	0.0371 (0.0072)	101

*The values in parentheses indicate standard errors of the CV error.

<Table 5> Results for CRX dataset

	Depth=1		Depth=3	
Method	CV Error	Size	CV Error	Size
AdaBoost	0.1532 (0.0141)*	200	0.1516 (0.0140)	200
SCAD	0.1286 (0.0131)	17	0.1501 (0.0140)	45
LASSO	0.1317 (0.0132)	29	0.1363 (0.0134)	82
Cluster Pruning Method	0.1317 (0.0132)	193	0.1501 (0.0140)	101

*The values in parentheses indicate standard errors of the CV error.

<Table 6> Results for IONO dataset

	Depth=1		Depth=3	
Method	CV Error	Size	CV Error	Size
AdaBoost	0.0854 (0.0149)*	200	0.0711 (0.0137)	200
SCAD	0.0883 (0.0151)	75	0.0711 (0.0137)	49
LASSO	0.0854 (0.0149)	55	0.0711 (0.0137)	54
Cluster Pruning Method	0.0912 (0.0154)	172	0.0655 (0.0132)	171

*The values in parentheses indicate standard errors of the CV error.

<Table 7> Results for SONAR dataset

	Depth=1		Depth=3	
Method	CV Error	Size	CV Error	Size
AdaBoost	0.1587 (0.0253)*	200	0.1153 (0.0221)	200
SCAD	0.1730 (0.0262)	75	0.1057 (0.0213)	116
LASSO	0.1394 (0.0240)	97	0.1009 (0.0209)	66
Cluster Pruning Method	0.1490 (0.0247)	188	0.0962 (0.0204)	149

*The values in parentheses indicate standard errors of the CV error.

3.3.4 Classification Noise

Dietterich (2000) found that the performance of AdaBoost could be destroyed by some noise cases. In Dietterich (2000), he concluded that classification noise destroys the effectiveness of AdaBoost compared to other ensemble methods such as Bagging (Breiman, 1996) and Randomization (Dietterich, 2000), and AdaBoost is not a good choice in this situation. We expect that pruning methods can improve this weakness to some degree. That is, pruning methods can not only reduce the misclassification rate, but also reduce the size of the boosting ensemble. To explore the effects of classification noise, we added random classification noise. That is, we chose 20% of the data (randomly, without replacement) and changed their class labels to be incorrect. We expect that AdaBoost will try to fit the noise data and continue to produce ineffective trees. In this case, as the number of boosting iterations increases, the final prediction rule may overfit the training data. We thus choose 30 or 50 boosting iterations to construct an ensemble of classification trees. The results of the pruning ensemble with 100 and 200 iterations are almost the same as those for 50 iterations, so we omit the results in this paper. We again performed the pruning methods based on penalized regression. Because the performance of the clustering pruning method is worse than those of the penalized methods as we can see the results of previous sections, the clustering pruning method was not performed.

3.3.5 Results of Analysis with Classification Noise

The results of analysis with classification noise are shown in <Table 8> - <Table 11>. As in the without-noise cases (in Subsection 3.3.3), the pruning methods did not significantly increase the CV errors and reduced the sizes of boosting ensembles significantly. Because of the weakness of boosting in the classification noise cases, the performance of 50 iterations is similar to or worse than that of 30. The large boosting ensemble is not always necessary but a small boosting ensemble could be better than a large one. However, after pruning the boosting ensemble, the performance and the size of the ensemble are not significantly different. So we can expect that the behaviors of the ensembles after pruning are almost the same in terms of the CV errors and sizes as the number of iterations exceeds 30.

The results of the experiment show that in many cases the sizes and CV errors are reduced simultaneously after pruning. This fact is especially remarkable for the CRX dataset with depth=3. The CV errors after pruning are reduced by 10-20% compared to the AdaBoost CV errors and the sizes are reduced by more than 50%. The pruning method based on the LASSO penalty prunes more trees than other methods. As in the linear regression setting, the LASSO method is slightly better than the SCAD method when the dataset is contaminated with some noise. As expected, the pruning methods based on penalized regression can improve the weakness of AdaBoost to some degree.

<Table 8> Results for BREAST dataset with classification noise

	size 30		size 50	
Method	CV Error	Size	CV Error	Size
		Depth=1		
AdaBoost	0.2504 (0.0166)*	30	0.2445 (0.0164)	50
SCAD	0.2372 (0.0163)	12	0.2343 (0.0162)	16
LASSO	0.2299 (0.0161)	11	0.2269 (0.0161)	18
		Depth=3		
AdaBoost	0.2489 (0.0165)	30	0.2474 (0.0165)	50
SCAD	0.2372 (0.0163)	16	0.2328 (0.0162)	21
LASSO	0.2269 (0.1603)	2	0.2240 (0.0160)	5

*The values in parentheses indicate standard errors of the CV error.

<Table 9> Results for CRX dataset with classification noise

	size 30		size 50	
Method	CV Error	Size	CV Error	Size
		Depth=1		
AdaBoost	0.2986 (0.0179)*	30	0.2879 (0.0177)	50
SCAD	0.2848 (0.0177)	4	0.2848 (0.0177)	5
LASSO	0.2741 (0.0175)	5	0.2741 (0.0175)	5
		Depth=3		
AdaBoost	0.3706 (0.0189)	30	0.3721 (0.0189)	50
SCAD	0.3078 (0.0181)	11	0.3063 (0.0180)	11
LASSO	0.3002 (0.0179)	7	0.3032 (0.0180)	11

*The values in parentheses indicate standard errors of the CV error.

<Table 10> Results for IONO dataset with classification noise

	size 30		size 50	
Method	CV Error	Size	CV Error	Size
		Depth=1		
AdaBoost	0.3077 (0.0246)*	30	0.3105 (0.0247)	50
SCAD	0.2906 (0.0242)	8	0.2906 (0.0242)	10
LASSO	0.2906 (0.0242)	2	0.2906 (0.0242)	5
		Depth=3		
AdaBoost	0.3020 (0.0245)	30	0.3048 (0.0246)	50
SCAD	0.2991 (0.0244)	15	0.2877 (0.0242)	11
LASSO	0.2991 (0.0244)	2	0.2934 (0.0243)	4

*The values in parentheses indicate standard errors of the CV error.

<Table 11> Results for SONAR dataset with classification noise

	size 30		size 50	
Method	CV Error	Size	CV Error	Size
		Depth=1		
AdaBoost	0.4039 (0.0340)*	30	0.4087 (0.0341)	50
SCAD	0.3894 (0.0338)	10	0.3894 (0.0338)	9
LASSO	0.3942 (0.0339)	6	0.3942 (0.0339)	2
		Depth=3		
AdaBoost	0.3606 (0.0333)	30	0.3606 (0.0333)	50
SCAD	0.3798 (0.0337)	22	0.3750 (0.0336)	30
LASSO	0.3654 (0.0334)	10	0.3798 (0.0337)	12

*The values in parentheses indicate standard errors of the CV error.

4. Summary and Concluding Remarks

In this paper, pruning boosting ensemble methods via penalized regression are considered. Sometimes the ensemble constructed by AdaBoost becomes too large to be usefully applied in real problems. Thus it is necessary to prune the ensemble of decision trees without sacrificing too much accuracy. To prune the ensemble of decision trees effectively, we propose pruning methods that apply the variable selection methods via penalized regression technique using LASSO and SCAD penalty functions. The pruning methods via penalized regression reduce the size of the boosting ensemble without significant increase of test or CV errors. Occasionally the size of the ensemble can be reduced to about 1/10 without hurting the performance of the ensemble. Sometimes the pruned ensemble achieves an improvement in the test error. The performance of AdaBoost could be destroyed by some noise cases. We showed that the proposed pruning methods can improve this weakness of AdaBoost to some degree. The experiments were performed on real datasets by adding random errors to the class variable y. In most cases of high noise level, the number of trees pruned by the pruning methods via penalized regression is very large. Taking into account these results, it is very useful to apply the proposed methods after performing the AdaBoost procedure.

A unified algorithm proposed by Fan and Li (2001) is used to estimate regression coefficients for the SCAD penalty function. This algorithm contains a matrix inversion. Thus the stability of the algorithm can not be guaranteed. In addition, the solution may be a local one since the object function is locally approximated by a quadratic function. To improve these problems of the unified algorithm, stagewise approaches such as gradient boosting (Friedman, 2001) and margin boost (Mason et al., 2000) can be considered. It could be worth while studying a boosting ensemble via penalized regression using a SCAD penalty function.

The proposed pruning ensemble methods can be used for the estimation of the weights of various ensemble methods. For example, the balancing method (Heskes, 1997) is an intermediate form between bagging (Breiman, 1996) and bumping (Tibshirani and Knights, 1999). The balancing method uses quadratic programming to estimate the weights of classifiers and thus the computation is expensive and may be impossible for large datasets. Thus if the proposed pruning methods are applied to the balancing method, we expect that our method can give better results than Heskes' (1997) approach.

Referneces

[1] Breiman, L. (1996). Bagging predictors. *Machine Learning*, Vol. 24, 123-140.

[2] Breiman, L. (1998). Arcing classifiers (with discussion). *Annals of Statistics*, Vol. 26, 801-849.

[3] Breiman, L., Friedman, J., Olshen, R. and Stone, C. (1984). *Classification and Regression Trees*, Chapman and Hall, New York.

[4] Dietterich, T.G. (2000). An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and

randomization. *Machine Learning*, Vol. 40, 139-157.

[5] Fan, J. and Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, Vol. 96, 1348-1360.

[6] Freund, Y. and Schapire, R. E. (1997). A decision-theoretic generalization of online learning and application to boosting. *Journal of Computer and System Science*, Vol. 55, 119-139.

[7] Friedman, J. (2001). Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, Vol. 29, 1189-1232.

[8] Hastie, T., Tibshirani, R. and Friedman, J.H. (2001). *Elements of Statistical Learning*. Springer-Verlag, New York.

[9] Heskes, T. (1997). Balancing between bagging and bumping, In Mozer, M., Jordan, M., and Petsche, T. editors. *Advances in Neural Information Processing*, Morgan Kaufmann.

[10] Lazarevic, A. and Obradovic, Z. (2001). The effective pruning of neural network ensembles. *Proceedings of 2001 IEEE/INNS International Joint Conference on Neural Networks*, 796-801.

[11] Margineantu, D.D. and Dietterich, T.G. (1997). Pruning adaptive boosting. *Proceedings of the 14th International Conference in Machine Learning*, 211-218.

[12] Mason, L., Baxter, J., Bartlett, P.L. and Frean, M. (2000). Functional gradient techniques for combining hypotheses, In A. J. Smola, P. L. Bartlett, B. Scholkopf and D. Schuurmans, editors. *Advances in Large Margin Classifiers*, Cambridge: MIT press.

[13] Merz, C.J. and Murphy, P.M. (1998). UCI Repository of Machine Learning database. Available at http://www.ics.uci.edu/~mlearn/MLRepository.html

[14] Quinlan, J.R. (1993). *C4.5 : Programs for Machine Learning*, Morgan Kaufmann, San Maeto, CA.

[15] Quinlan, J.R. (1996). Bagging, boosting, and C4.5. *Proceeding of 13th National Conference on Artificial Intelligence*, 725-730.

[16] Rosset, S., Zhu, J. and Hastie, T. (2004). Boosting as a regularized path to a maximum margin classifier. *Journal of Machine Learning Research*, Vol. 5, 941-973.

[17] Tamon, C. and Xiang, J. (2000). On the boosting pruning problem. *Proceedings of 11th European Conference on Machine Learning, Lecture Notes in Computer Science*, Vol. 1810, 404-412.

[18] Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of Royal Statistical Society B*, Vol. 58, 267-288.

[19] Tibshirani, R. and Knight, K. (1999). Model selection and inference by bootstrap

"bumping". *Journal of Computational and Graphical Statistics*, Vol. 8, 671-686.

[Received April 2006, Accepted July 2006]