Journal of the Korean Data & Information Science Society
2016, **27**(1), 255–264

http://dx.doi.org/10.7465/jkdi.2016.27.1.255
한국데이터정보과학회지

Tree size determination for classification ensemble

Sung Hoon Choi[1] · Hyunjoong Kim[2]

[12]Department of Applied Statistics, Yonsei University

Received 19 November 2015, revised 4 January 2016, accepted 13 January 2016

Abstract

Classification is a predictive modeling for a categorical target variable. Various classification ensemble methods, which predict with better accuracy by combining multiple classifiers, became a powerful machine learning and data mining paradigm. Well-known methodologies of classification ensemble are boosting, bagging and random forest. In this article, we assume that decision trees are used as classifiers in the ensemble. Further, we hypothesized that tree size affects classification accuracy. To study how the tree size influences accuracy, we performed experiments using twenty-eight data sets. Then we compare the performances of ensemble algorithms; bagging, double-bagging, boosting and random forest, with different tree sizes in the experiment.

Keywords: Bagging, boosting, classification, decision tree, double-bagging, ensemble, random forest.

1. Introduction

Classification is defined as a model of prediction for a categorical target variable. There are several methods of classifier algorithms which are in popular. Logistic regression, linear discriminant analysis (LDA), and quadratic discriminant analysis (QDA) are typical statistical methods. Complicated methods are also used in common, such as neural network, support vector machine (SVM) and decision trees.

In classification ensemble method, many trained classifiers are combined to make a final prediction (Dietterich, 2000). Ensemble methods perform better than a single classifier in general (Hansen and Salamon, 1990). Many scholars have contributed to improving the accuracy of classification ensemble in machine learning and statistics community (Breiman, 1996a; Freund and Schapire, 1996; Bauer and Kohavi, 1999; Shim and Hwang, 2014). Three usual ensemble methods, Boosting (Schapire, 1990; Freund and Schapire, 1996), Bagging (Breiman, 1996a), and Random Forest (Breiman, 2001), have received remarkable attention and are widely used. These methods implemented with resampled or reweighted training data sets from the original data and a classifier was applied on each of them repeatedly.

[1] Graduate student, Department of Applied Statistics, Yonsei University, Seoul 03722, South Korea.
[2] Corresponding author: Professor, Department of Applied Statistics, Yonsei University, Seoul 03722, South Korea. E-mail: hkim@yonsei.ac.kr

2. Ensemble methods and previous research

Boosting has been developed to improve the performance of any weak learning algorithm that needs only to be mildly better than a random guess. From the performance of previous classifiers, boosting changes the distribution of the training data set flexibly. Then, boosting takes a weighted majority vote of their predictions. The most well-known method is AdaBoost (adaptive boosting) by Freund and Schapire (1996), and later Schapire and Singer (1999) generalized it. Furthermore, the multiclass AdaBoost algorithm, which extended from binary to multiclass with the forward stagewise estimation method, is suggested by Zhu *et al*. (2009). In this paper, we use multiclass AdaBoost algorithm described in Zhu *et al*. (2009). We show the algorithm for multiclass AdaBoost in Table 2.1.

Table 2.1 Multiclass AdaBoost algorithm

Given:

- $(x_1, c_1), ..., (x_n, c_n)$: a set of training data, where the input $x_i \in R^p$ and the output $c_i \in \{1,...,K\}$
- K : number of classes
- B : number of classifiers in Boosting
- $C(x)$: classification rule, where $C(x) : x \rightarrow 1,...,K$

Procedure:

1. Initialize the observation weights.
   $w_i = 1/n$, $i = 1, ..., n$.

2. For $b = 1$ to B
   (a) Fit a classifier $T^{(b)}(x)$ to the training data using weights w_i.
   (b) Compute
   $$err^{(b)} = \sum_{i=1}^{n} w_i \cdot I(c_i \neq T^{(b)}(x_i)) / \sum_{i=1}^{n} w_i$$
   (c) Compute
   $$\alpha^{(b)} = log((1 - err^{(b)})/err^{(b)}) + log(K - 1).$$
   (d) Set
   $$w_i \leftarrow w_i \cdot exp(\alpha^{(b)} I(c_i \neq T^{(b)}(x_i))), \quad i = 1, ..., n.$$
   (e) Renormalize w_i, $\quad i = 1, ..., n$.

3. Aggregate the B train classifiers using weighted vote.
   $$C(x) = argmax_k \sum_{i=1}^{B} \alpha^{(b)} \cdot I(T^{(b)}(x) = k).$$

The bagging algorithm (Breiman, 1996a) generates the classifiers in ensemble by using bootstrap samples. Each of them is established by random sampling, with replacement, the same number of instances as the original data. Then the final classification, using the classifiers in ensemble, is obtained by simple majority voting. Table 2.2 shows the algorithm for bagging. Usually, bagging constructs more accurate classifiers than a single classifier. When bootstrap samples are made from the original training set, some of the original examples may be selected several times while others may not be selected at all. Regarding each bootstrap sample, an average of approximately 63% of unique training individuals was selected while the size remained unchanged. The rest 37% of training sets can be used to produce more accurate classification method. It is referred as the out-of-bag (OOB) sample (Breiman, 1996b).

Table 2.2 Bagging algorithm

Given:
- L : training data set composed of n instances
- K : number of classes
- B : number of classifiers in Bagging
- $C(x)$: classification rule, where $C(x) : x \rightarrow 1,...,K$

Procedure:
1. Generate B bootstrap samples $L_1, ..., L_B$ from the original training data set L.

2. Construct train classifiers $C_b(x), b = 1, ..., B$ from each of L_b samples.

3. Aggregate the B train classifiers using simple majority vote.

$$C_B(\mathbf{x}) = argmax_j \sum_{b=1}^{B} I(C_b = j), \text{ for } j \in 1, ..., K.$$

More recently, random forest (Breiman, 2001) was established as an ensemble method to combine tree classifiers such that each level of trees depends on the values of their features, which are chosen among the features sampled independently. In particular, during the construction of the tree, the chosen split is no longer the best split among all features when splitting a node. Instead, the split is the best one among a random subset of the features. Because of this randomness, the bias usually slightly increases, but its variance decreases due to averaging effect. Table 2.3 describes the random forest algorithm.

Table 2.3 Random forest algorithm

Given:
- L : training data set composed of n instances
- K : number of classes
- B : number of classifiers in Random Forest
- $C(x)$: classification rule, where $C(x) : x \rightarrow 1,...,K$

Procedure:
For $b = 1$ to B
1. Generate B bootstrap samples $L_1, ..., L_B$ from the original training data set L.

2. Grow a random forest tree using a random feature selection from bootstrapped data
   : randomly select $\sqrt{n}$ or $n/3$ predictors at each node and split the data using the best predictors where n is the number of variables in x.

3. Construct train classifiers $C_b(x)$, $b = 1, ..., B$ from each ensemble of trees.

4. Aggregate the B train classifiers using simple majority vote.

$$C_B(x) = argmax_j \sum_{b=1}^{B} I(C_b(x) = j), \text{ for } j \in 1, ..., K$$

Furthermore, some researchers have designed hybrid ensemble methods to improve the accuracy substantially. Hothorn and Lausen (2003) suggested double-bagging, which is the combination method of LDA and a classification tree as the base model of bagging. They used the estimated coefficients of the linear discriminant function from the out-of-bag (OOB) sample, and then the corresponding discriminant scores from in-bag sample contribute to classification tree modeling as additional predictors. Table 2.4 shows the double-bagging algorithm.

Table 2.4 Double-Bagging algorithm

Given:
- L : training data set composed of n instances
- K : number of classes
- B : number of classifiers in Double-Bagging
- $C(x)$: classification rule, where $C(x) : x \rightarrow 1,...,K$

Procedure:

1. Draw B random samples $L_1, ..., L_B$ with replacement from the original training data set L and let $X^{(b)}$ denote the matrix of predictors from L_b, for $b = 1, ..., B$.

2. Compute an LDA using the out-of-bag sample $L \backslash L_b$, that gives a $p \times (K-1)$ matrix $Z^{(b)}$, where the columns are the coefficients of the linear discriminant functions.

3. Construct the classifier $C_b(x)$ using the original predictor variables as well as the discriminant variables of the bootstrap sample $(L_b, X^{(b)} Z^{(b)})$ from each of L_b samples.

4. Iterate steps (2) and (3) for $b = 1, ..., B$ bootstrap samples.

5. Aggregate the B train classifiers using simple majority vote.

$$C_B(x) = argmax_j \sum_{b=1}^{B} I(C_b(x) = j), \text{ for } j \in 1, ..., K$$

3. Effect of tree size on ensemble methods

In this research, we study the optimal size of decision trees on the performance of the classification ensemble methods where decision trees are used as classifiers. We define the tree size as the depth of the decision tree, which is the length of the longest path from a root node to a terminal node. For example, the Figure 3.1 has the tree size of 3.

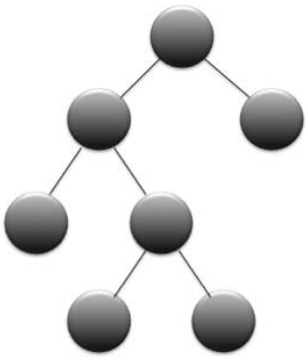

Figure 3.1 A decision tree with size of 3

Probably, the most common tree size in boosting is the number of categories of target variables. For example, if four categories of target variable exist, then the boosting uses the depth of four. Kim *et al.* (2012) studied this issue for boosting algorithms and provided

a guideline. The previous research suggested the model for estimating the right tree size suitable for the boosting algorithm, using variables that can explain the nature of a given dataset. The suggested model reveals that the optimal tree size for a given dataset can be estimated by the error rate of stump tree, the number of classes, the depth of a single tree, and the gini impurity (Kim *et al.*, 2012). Here, stump tree denotes a tree with two leaf nodes, or which has only one tree size. Table 3.1 describes the modified multiclass AdaBoost algorithm with tree size selection step.

For bagging and random forest, however, there are no typical rules for choosing the tree size. Depending on the parameters that the developer specified before fitting, each algorithm uses different tree sizes that result in varied accuracy. This motivated us to study the effect of tree size on various ensemble methods, which seem to have diverse trends.

To explore all possibilities, we consider every tree sizes. In other words, as setting parameters of classification ensemble model, we set the tree size with many candidate values. Then we construct the ensemble of decision trees. Since each data set has its own optimal tree size, we want to figure out a rule that governs the optimal tree size determination.

Table 3.1 Modified multiclass AdaBoost algorithm (tree size selection step included)

Given:

- $(x_1, c_1), ..., (x_n, c_n)$: a set of training data, where the input $x_i \in R^p$ and the output $c_i \in \{1,...,K\}$
- K : number of classes
- $C(x)$: classification rule, where $C(x) : x \rightarrow 1,...,K$

Procedure:

1. Initialize the observation weights.
   $w_i = 1/n$, $i = 1, ..., n$.

   (a) Fit a classifier $T^{(b)}(x)$ to the training data using weights w_i.
   (1) Find the following values for the given dataset.
   $StumpErr$: stump tree error rate / root node error rate
   $Depth$: tree size of unpruned tree
   $gini$: impurity of the given dataset $1 - \sum_{i=1}^{K} p_i^2$, where p_i is the proportion of i^th class
   (2) Find the appropriate tree size $D = int(\widehat{D})$.
   $\widehat{D} = -1.4820 + 1.4807 \cdot Depth - 1.4051 \cdot Depth \cdot I + 9.7583 \cdot gini \cdot I$
   where, $I = \begin{cases} 1 & \text{if } StumpErr \leq 0.7882 \\ 0 & \text{if } StumpErr > 0.7882 \end{cases}$
   (3) Fit a classifier $T^{(b)}(x; D)$, which has trees of size D, to the training data using weights w_i
   (b) Compute
   $err^{(b)} = \sum_{i=1}^{n} w_i \cdot I(c_i \neq T^{(b)}(x_i)) / \sum_{i=1}^{n} w_i$.
   (c) Compute
   $\alpha^{(b)} = log((1 - err^{(b)})/err^{(b)}) + log(K - 1)$.
   (d) Set
   $w_i \leftarrow w_i \cdot exp(\alpha^{(b)} I(c_i \neq T^{(b)}(x_i))), \quad i = 1, ..., n$.
   (e) Renormalize $w_i, \quad i = 1, ..., n$.

3. Output.
   $C(x) = argmax_k \sum_{i=1}^{B} \alpha^{(b)} \cdot I(T^{(b)}(x) = k)$.

4. Experimental design

Extending previous experiment of the tree size effect on boosting (Kim *et al.*, 2012), we applied the idea to other ensemble methods; bagging, double-bagging, and random forest, in an empirical study. The experiment is conducted based on 28 actual datasets, which mostly come from the UCI Data Repository (Asuncion and Newman, 2007). Table 4.1 refers to the descriptions of the datasets. To handle the missing data in some of datasets, we replace with column medians for numeric variables and the most frequent levels for categorical variables.

For each dataset, we control the tree sizes from 1 to 10 for each of ensemble methods, and compared the accuracies. To compare the performances fairly, all the ensemble methods are run on the same 10-fold cross-validation training data. We also repeat the cross-validation 50 times to remove the effect of random seeds. Different seed numbers are assigned on each iteration for stable performance. An average of 50 cross-validation accuracy outcomes for each tree sizes on each ensemble method is calculated. In the experiment, we use an ensemble size of 100, which means 100 decision trees are combined.

Table 4.1 Dataset description

Data	Description	♯ instances	♯ classes	♯ variable	source
bcw	Brest Cancer Wisconsin	699	2	10	UCI
bld	Liver Disorders	345	2	6	Kwak and Kim (2014)
bod	Body Dimension	507	2	24	Heinz *et al.* (2003)
bos	Boston Housing	506	3	14	UCI
cmc	Contraceptive Method Choice	1473	3	9	UCI
col	Horse Colic	368	3	27	UCI
cre	Credic Approval	690	2	15	UCI
cyl	Cylinder Bands	540	2	35	UCI
der	Dermatology	366	6	33	UCI
dia	Diabetes	532	2	7	Loh (2009)
fis	Fish	159	7	7	Kim and Loh (2003)
ger	Germa Credit	1000	2	20	UCI
gla	Glass	214	6	9	UCI
hea	Statlog (Heart)	270	2	13	UCI
int	Chessboard	1000	2	10	Kim *et al.* (2012)
ion	Ionosphere	351	2	34	UCI
iri	Iris	150	3	4	UCI
lak	Lakes	259	6	16	Loh (2009)
led	LED Display	6000	10	7	UCI
pid	Pima Indians Diabetes	768	2	8	UCI
pov	Poverty	97	6	6	Kim and Loh (2001)
sea	Vocalisation of Harp Seals	3000	3	7	Terhune (1994)
spe	SPECTF Heart	267	2	44	UCI
usn	Usnews	1320	3	27	Statlib (2010)
veh	Statlog (Vehicle Silhouettes)	946	4	18	UCI
vol	Volcano	1521	6	6	Loh (2009)
vot	Congressional Voting Records	435	2	16	UCI
vow	Vowel Recognition	990	11	10	UCI

For accuracy comparison, CART algorithm (Breiman *et al.*, 1984), the most famous decision tree algorithm, is employed as a base classifier, except that LDA is used in conjunction with CART in double-bagging methods. Comparisons are run under R environment. RPART (Therneau and Atkinson, 1997) is a CART implementation under R. We programmed bagging and double-bagging using RPART as a classifier. We use 'randomForest' package (Liew and Wiener, 2002) under the R environment to practice the random forest (Breiman, 2001).

To control the tree size setting on ensemble methods, we use 'rpart.control', which handles the aspects of the RPART fit. For random forest, specifically, we assign the maximum node through setting 'maxnodes' argument in that package due to different package developers. We implement two ways of comparison. First, we use relative mean accuracy. Next, we implement the paired t-test using 50 cross-validated accuracies for each of the 28 datasets. To compare the performance fairly, a single tree with a pruning option is used as a baseline and compared with the outcome of the ensemble methods.

5. Accuracy comparison

We demonstrate the relative mean accuracy by increasing the tree size. In Figures 5.1 and 5.2, a horizontal axis refers to tree size for each ensemble method. A vertical axis refers to the relative mean accuracy, which is the accuracy of an ensemble method for tree size over the accuracy of stump tree.

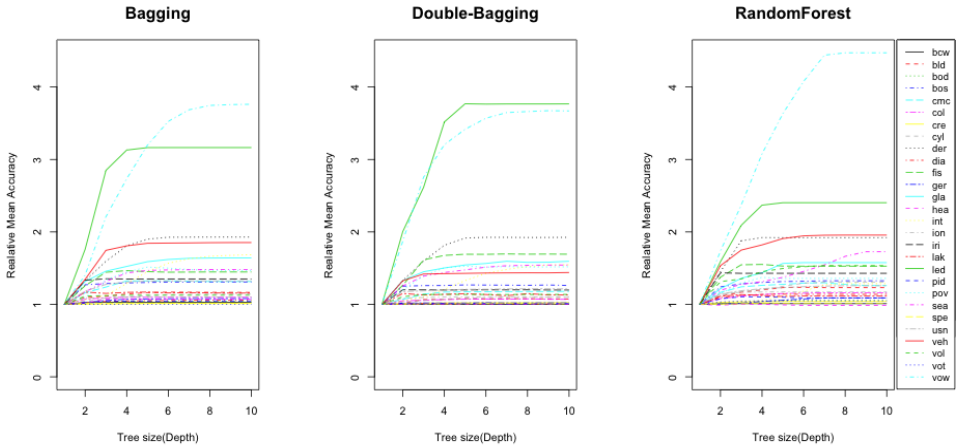

Figure 5.1 Relative mean accuracy of different tree sizes for Bagging, Double-Bagging, and Random Forest

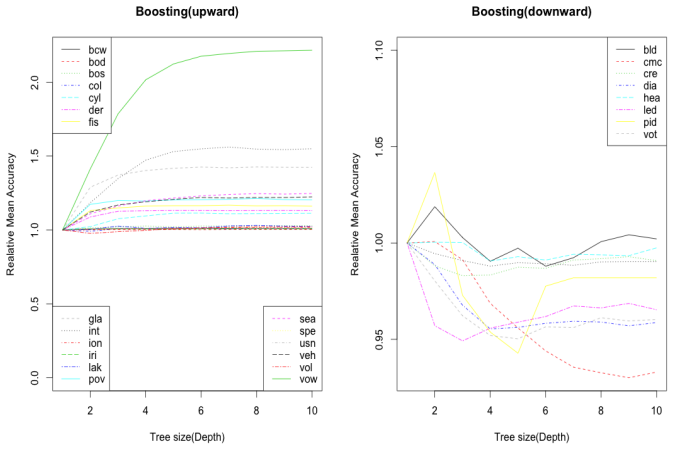

Figure 5.2 Relative mean accuracy of different tree sizes for boosting

First, the tree size may have a significant impact on improving the performance of ensemble trees. From the Figures 5.1 and 5.2, even though the boosting experiment shows a different behavior on accuracy between tree sizes among some datasets, other ensemble methods do not show remarkable differences. Second, as increasing the tree size, performance patterns are diverse for each dataset. However, unlike the boosting result, the larger tree size generally gives more accurate results under bagging, double-bagging, and random forest ensemble scheme. Finally, on a fixed data set, similar patterns are found for bagging, double-bagging, and random forest. For instance, the accuracy of ensemble trees increase very suddenly as tree size increases in some dataset: 'der', 'led', 'veh', and 'vow'.

6. Significance

Since we use the same 10-fold cross-validated data in the experiment, the cross-validation accuracy of each classification method can be compared pairwise. In addition, due to repeating the cross-validation 50 times, there are 50 paired accuracies to be compared. We implement the paired t-tests to examine the pairwise differences between ensemble methods and a single pruned tree. Figure 6.1 represents the 28 test-statistics of the paired t-test using all data sets. In the boxplot, a larger positive t-statistic indicates better accuracy for one method over the other. Tree sizes $(1, 2, \ldots, 10)$ is used to determine if a relationship between the t-statistics and tree sizes exist for the ensemble methods over a single pruned tree. First of all, bagging, double-bagging, and random forest methods are outperformed by the single pruned tree in smaller tree size. However, the boosting method consistently outperforms the single pruned tree at every tree size.

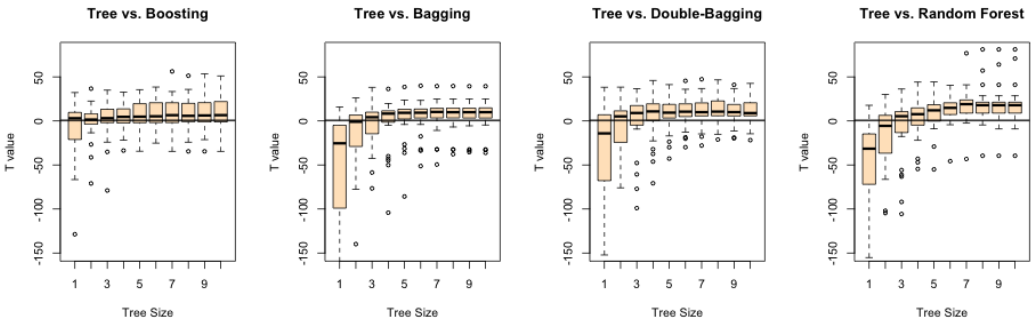

Figure 6.1 Pared t-statistics for comparing a single pruned tree vs. other methods. Large positive values mean significantly better accuracy for other methods over a single pruned tree.

As tree size increases, bagging, double-bagging and random forest methods gradually outperformed the single pruned tree. In addition, the double-bagging method produces more accurate results than bagging method for each tree size option. Conclusively, the random forest method produces slightly more accurate results than the double-bagging and bagging method.

7. Concluding remarks

This study assessed significant influences on accuracy of several ensemble methods by tree size. We defined the tree size as the depth of a decision tree and assumed that it affects the accuracy of classification ensemble. From the experiments comparing popular ensemble methods (boosting, bagging, double-bagging and random forest), we found that the tree sizes effects are different on two types of ensemble methods. First, in boosting type ensemble method, tree size effect is not uniform. Depending on the dataset, the optimal tree size can be small. On the other hand, bootstrap based ensemble methods (bagging, double-bagging and random forest) show more uniform results than boosting. In general, large number of tree size shows better classification accuracies in most data sets. Therefore, we conclude to use large tree size in the construction of classification ensembles that uses bootstrap technique such as bagging, double-bagging and random forest.

To be more specific about the tree size determination, we suggest to use Table 3.1 for boosting type ensemble methods since the optimal tree size depends on the features of dataset. For bootstrap based ensemble methods, it would be unclear about how large is considered to be large. We suggest to use an unpruned tree because it generally gives large tree sizes for the data being analyzed. Therefore, it is not necessary to specify the tree size before the ensemble construction in this case.

As limitations of the study, when we experimented the double-bagging setting, only one-dimensional canonical LDA was considered to get an additional variable. However, it is possible to apply higher dimensional canonical LDA to produce additional predictors. Additionally, other classification method can be applied with the out-of-bag sample. For example, as a future study, we can utilize the random forest method combined with the OOB samples.

References

Asuncion, A. and Newman, D. J. (2007). UCI machine learning repository. University of California, Irvine, School of Information and Computer Science, `http://archive.ics.uci.edu/ml`.

Bauer, E. and Kohavi, R. (1999). An empirical comparison of voting classification algorithms: Bagging, boosting and variants. *Machine Learning*, **36**, 105-139.

Breiman, L. (1996a). Bagging predictors. *Machine Learning*, **26**, 123-140.

Breiman, L. (1996b). *Out-of-bag estimation*, Technical Report, Statistics Department, University of California Berkeley, Berkeley, California 94708, `https://www.stat.berkeley.edu/~breiman/OOBestimation.pdf`.

Breiman, L. (2001). Random forests. *Machine Learning*, **45**, 5-32.

Breiman, L., Friedman, J. H., Olshen, R. A. and Stone, C. J. (1984). *Classification and regression trees*, Chapman and Hall, New York.

Dietterich, T. (2000). *Ensemble methods in machine learning*, Springer, Berlin.

Freund, Y. and Schapire, R. (1996). Game theory, on-line prediction and boosting. *Proceedings of the Ninth Annual Conference on Computational Learning Theory*, 325-332.

Hansen, L. K., Salamon, P. (1990). Neural network ensembles. *IEEE Transactions on Pattern Analysis and machine Intelligence*, **12**, 993-1001.

Heinz, G., Peterson, L. J., Johnson, R. W. and Kerk, C. J. (2003). Exploring relationships in body dimensions. *Journal of Statistics Education*, **11**, `http://www.amstat.org/publications/jse/v11n2/datasets.heinz.html`.

Hothorn, T. and Lausen, B. (2003). Double-bagging: Combining classifiers by bootstrap aggregation. *Pattern Recognition*, **36**, 1303-1309.

Kim, A., Kim, J. and Kim, H. (2012). The guideline for choosing the right-size of tree for boosting algorithm. *Journal of the Korean Data and Information Science Society*, **23**, 949-959.

Kim, H. and Loh, W. Y. (2001). Classification trees with unbiased multiway splits. *Journal of the American Statistical Association*, **96**, 589-604.

Kim, H. and Loh, W. Y. (2003). Classification trees with bivariate linear discriminant node models. *Journal of Computational and Graphical Statistics*, **12**, 512-530.

Kwak, S. and Kim, H. (2014). Comparison of ensemble pruning methods using Lasso-bagging and WAVE-bagging. *Journal of the Korean Data and Information Science Society*, **25**, 1371-1383.

Liew, A. and Wiener, M. (2002). Classification and regression by random forests. *R News*, **2**, 18-22.

Loh, W. Y. (2009). Improving the precision of classification trees. *The Annals of Applied Statistics*, **3**, 1710-1737.

Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning*, **5**, 197-227.

Schapire, R. E. and Singer, Y. (1999). Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, **37**, 297-336.

Shim, J. and Hwang, C. H. (2014). Support vector quantile regression ensemble with bagging. *Journal of the Korean Data and Information Science Society*, **25**, 677-684.

Statlib. (2010). Datasets archive. Carnegie Mellon University, Department of Statistics, `http://lib.stat.cmu.edu`.

Terhune, J. M. (1994). Geographical variation of harp seal underwater vocalizations. *Canadian Journal of Zoology*, **72**, 892-897.

Therneau, T. and Atkinson, E. (1997). *An introduction to recursive partitioning using the RPART routines*, Mayo Foundation, Rochester, New York. `http://eric.univ-lyon2.fr/~ricco/cours/didacticiels/r/longdocrpart.pdf`.

Zhu, J., Zou, H., Rosset, S. and Hastie, T. (2009). Multi-class AdaBoost. *Statistics and its Interface*, **2**, 349-360.