

# Report

Aldo Giovanni e Giacomo

2023-05-25

## INTRODUCTION

Bagging predictors is a method for generating multiple versions of a predictor and using these to get an aggregated predictor. The aggregation averages over the versions when predicting a numerical outcome and does a plurality vote when predicting a class. The multiple versions are formed by making bootstrap replicates of the learning set and using these as new learning sets. The vital element is the instability of the prediction method. If perturbing the learning set can cause significant changes in the predictor constructed, then bagging can improve accuracy.  $\mathcal{L}$  A learning set  $\mathcal{L}$  consists of data  $\{(y_n, x_n), n = 1 \dots, N\}$

$y$  : class labels/numeric responses

$x$  : input

$\varphi(x, \mathcal{L})$  : predictor

$\mathcal{L}_{\parallel}$  : sequence of learning sets consisting of  $N$  independent observations from the same underlying distribution as  $\mathcal{L}$ .

Our scope of this project is to show that the sequence of predictors  $\varphi(x, \mathcal{L}_{\parallel})$  is a better predictor than  $\varphi(x, \mathcal{L})$ .

- $y$  is numeric

Replace  $\varphi(x, \mathcal{L})$  by the average of  $\varphi(x, \mathcal{L}_{\parallel})$ . In other words find the expectation of the predictor over  $\mathcal{L}$

.

- $y$  is class labels

- 

## CLASSIFICATION TREE

A classification tree is a structural mapping of binary decisions that lead to a decision about the class (interpretation) of an object (such as a pixel). Although sometimes referred to as a decision tree, it is more properly a type of decision tree that leads to categorical decisions. A regression tree, another form of decision tree, leads to quantitative decisions.

A classification tree is composed of branches that represent attributes, while the leaves represent decisions. In use, the decision process starts at the trunk and follows the branches until a leaf is reached. For a classification tree, we predict that each observation belongs to the most commonly occurring class of training observations in the region to which it belongs. In interpreting the results of a classification tree, we are often interested not only in the class prediction corresponding to a particular terminal node region, but also in the class proportions among the training observations that fall into that region. The task of growing a classification tree is quite similar to the task of growing a regression tree. Just as in the regression setting, we use recursive binary splitting to grow a classification tree. However, in the classification setting, RSS (Residual Sum Square) cannot be used as a criterion for making the binary splits. The classification tree suffer from high variance. This means that if we split the training data into two parts at random, and fit a classification tree to both halves, the results that we get could be quite different. In contrast, a procedure with low

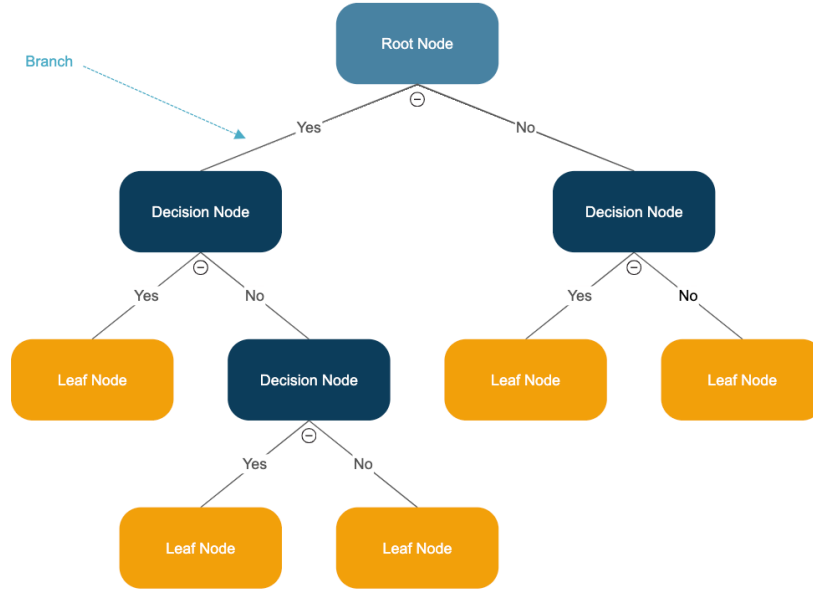


Figure 1: Figure 1.Classification Tree

variance will yield similar results if applied repeatedly to distinct data sets; linear regression tends to have low variance, if the ratio of  $n$  to  $p$  is moderately large

### Bagging

The bootstrap is used in many situations in which it is hard or even impossible to directly compute the standard deviation of a quantity of interest. Here the bootstrap can be used in order to improve statistical learning methods such as decision trees. The decision trees suffer from high variance. Bootstrap aggregation, or bagging, is a general-purpose procedure for reducing the variance of a statistical learning method; we introduce it here because it is particularly useful and frequently used in the context of decision trees.

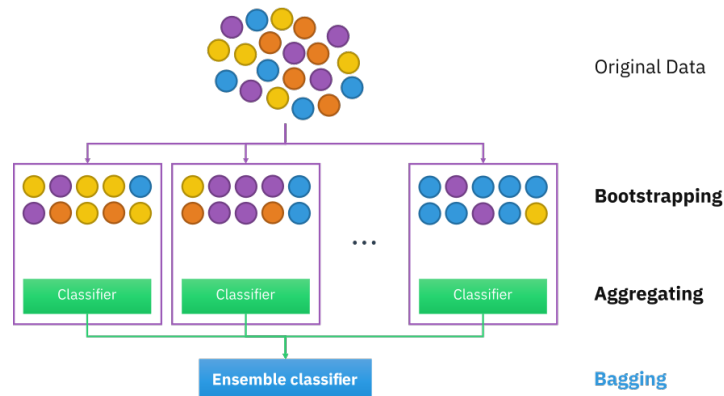


Figure 2: Figure 1.Bagging

Given a set of  $n$  independent observations  $Z_1, \dots, Z_n$ , each with variance  $\sigma^2$ , the variance of the mean  $\bar{Z}$  of the observations is given by  $\sigma^2/n$ . In other words, averaging a set of observations reduces variance. Hence a natural way to reduce the variance and increase the test set accuracy of a statistical learning method is to take many training sets from the population, build a separate prediction model using each training set, and average the resulting predictions.

## Random forest

Random forests are an improvement over bagged trees that aim to make the trees less similar to each other. In bagging, we create multiple decision trees using different training samples. However, in random forests, when building these trees, we make a small change. Instead of considering all predictors at each split, we randomly select a subset of predictors and choose only one from that subset for the split. We refresh this subset for each split, usually picking the square root of the total number of predictors.

This tweak is done because in bagged trees, if there is a strong predictor in the dataset, most or all of the trees will use that predictor in the top split. As a result, the predictions from these trees will be highly correlated. Unfortunately, averaging highly correlated predictions doesn't reduce the variability as much as averaging uncorrelated predictions. Therefore, bagging doesn't significantly reduce the variance compared to a single tree in such cases.

Random forests solve this problem by restricting each split to consider only a subset of predictors. This means that, on average, a portion of the splits won't even consider the strong predictor. By doing this, other predictors have a better chance to contribute to the splits. This process "decorrelates" the trees, making their average less variable and more reliable.

The main difference between bagging and random forests lies in the choice of the predictor subset size. If we use all predictors, it's equivalent to bagging. However, when using the square root of the total predictors, random forests show a reduction in the test error.

Using a small subset of predictors is particularly helpful when we have many correlated predictors.

Just like bagging, random forests don't overfit if we increase the number of trees (denoted as  $B$ ). Therefore, in practice, a sufficiently large value of  $B$  is used until the error rate stabilizes.

In the process of constructing a random forest, multiple decision trees are created. Each tree is generated independently, and for each tree, a random vector  $\Theta_k$  is generated. The generation of  $\Theta_k$  is unrelated to the previously generated random vectors  $\Theta_1, \dots, \Theta_{k-1}$ , but they are all drawn from the same distribution.

To grow a tree, the training set and the corresponding  $\Theta_k$  are used. This combination results in a classifier  $h(\mathbf{x}, \Theta_k)$ , where  $\mathbf{x}$  represents an input vector. In bagging, for example,  $\Theta$  is generated by counting the number of darts that randomly fall into  $N$  boxes, with  $N$  being the number of examples in the training set. In random split selection, the content of  $\Theta$  depends on its usage in constructing the tree.

Once a large number of trees are generated, they collectively contribute to the final classification decision. Each tree casts a vote, and the class with the highest number of votes is considered the most popular. This aggregation of trees is what we refer to as a random forest.

A random forest is a classifier consisting of a collection of tree-structured classifiers  $\{h(\mathbf{x}, \Theta_k), k=1, \dots\}$  where the  $\{\Theta_k\}$  are independent identically distributed random vectors and each tree casts a unit vote for the most popular class at input  $\mathbf{x}$ .

We are going to use diabetes dataset gathered among the Pima Indians by the National Institute of Diabetes and Digestive and Kidney Diseases and compare the result of decision tree, bagged and random forest.

```
misclassification_rate_vec <- rep(0,100)
misclassification_rate_vec_rf <- rep(0,100)
misclassification_rate_vec_bagged<- rep(0,100)

E_sim <- read.table('diabetes.csv',sep=',',header = TRUE)
E_sim$Outcome<- as.factor(E_sim$Outcome)

head(E_sim)
```

```
##   Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI
```

## 1	6	148	72	35	0	33.6
## 2	1	85	66	29	0	26.6
## 3	8	183	64	0	0	23.3
## 4	1	89	66	23	94	28.1
## 5	0	137	40	35	168	43.1
## 6	5	116	74	0	0	25.6

##	DiabetesPedigreeFunction	Age	Outcome
## 1	0.627	50	1
## 2	0.351	31	0
## 3	0.672	32	1
## 4	0.167	21	0
## 5	2.288	33	1
## 6	0.201	30	0

The dataset consists of 768 cases, 8 variables and two classes. The variables are medical measurements on the patient plus age and pregnancy information. The classes are: tested positive for diabetes (268) or negative (500).

In the simulated data, Breiman uses 300 samples as learning set from the 1800 samples generated. However we are going to use the raw dataset and split the dataset into 2 - a learning set, L, (10% of the data) with 76 rows and a test set, T, (90% of the data) with 692 rows. Next we will construct the classification tree model using `train()` function with 10-fold cross-validation. A bagged decision tree is created in parallel using the function `randomForest()` and setting the number of trees as 25 and `mtry` (the number of variables used) as 8. Last we create a Random Forest model using the same function but here we `mtry` as 3 ( $\sqrt{8}$ ). So the best 3 variables among the 8 will be chosen.

The three models are made to predict the result using the test dataset T and the miscalculation rate is calculated. The whole process is iterated 100 times for different combination of learning and test data and the mean error is calculated.

```
for (i in 1:100){
  index <- createDataPartition(E_sim$Outcome, p = 0.7, list = FALSE)

  # Divide the data into test set T and learning set L
  T <- E_sim[-index, ]
  L <- E_sim[index, ]

  # Train the classification tree model with 10-fold cross-validation
  modeldt <- train(
    Outcome ~ .,
    data = L,
    method = "rpart",
    trControl = trainControl(method = "cv", number = 10)
  )

  rf_model <- randomForest(Outcome ~ ., data = L, mtry = 3, importance=TRUE)
  bagged_tree <- randomForest(Outcome ~ ., data = L, mtry = 8, ntrees = 25)

  # Predict the class labels for the test set using the trained model
  predictions <- predict(modeldt, newdata = T)
  # Calculate misclassification rate
  misclassification_rate <- mean(predictions != T$Outcome)
  misclassification_rate_vec[i] <- misclassification_rate
}
```

```

predictions_rf <- predict(rf_model, newdata = T)
# Calculate misclassification rate
misclassification_rate_rf <- mean(predictions_rf != T$Outcome)
misclassification_rate_vec_rf[i] <- misclassification_rate_rf

predictions_bagged <- predict(bagged_tree, newdata = T)
# Calculate misclassification rate
misclassification_rate_bagged <- mean(predictions_bagged != T$Outcome)

misclassification_rate_vec_bagged[i] <- misclassification_rate_bagged
}

mis<- mean(misclassification_rate_vec)
mis_rf <- mean(misclassification_rate_vec_rf)
mis_bagged <- mean(misclassification_rate_vec_bagged)
err_data <- data.frame(Model = c("Decision Tree", "Bagging Predictor", "Random Forest"),
                        Error = c(mis,mis_bagged,mis_rf))
err_data

##           Model      Error
## 1    Decision Tree 0.2628261
## 2 Bagging Predictor 0.2516087
## 3    Random Forest 0.2423478

```

From the table above we can see that the Error is maximum for Decision Tree and least for Random Forest. Bagged Predictor model performs better than Decision Tree ie it can reduce the variance of an unstable dataset.

### Larger Data Sets

For large dataset that is dataset with more than 2000 samples, 10% of the data is set aside as training set and the remaining 90% as learning set. The set aside 10% was then used to select the best pruned subtree. In bagging, 50 bootstrap replicates of the training set were generated and a large tree grown on each one. The error rate given in the paper shown below.

Data Set	$e_S$	$e_B$	Decrease
letters	12.6	6.4	49%
satellite	14.8	10.3	30%
shuttle	.062	.014	77%
DNA	6.2	5.0	19%

Figure 3: Figure 1. Bagging

The percentage of decrease in misclassification rate is similar to the rate of difference in small dataset. But if we have a closer look at the errors, the error for classification tree is very much small and the improvement is not extremely significant.

**Bagging Regression Trees** Regression trees are decision trees in which the target variables can take continuous values instead of class labels in leaves. Regression trees use modified split selection criteria and stopping criteria. By using a regression tree, you can explain the decisions, identify possible events that might occur, and see potential outcomes. The analysis helps you determine what the best decision would be. To divide the data into subsets, regression tree models use nodes, branches, and leaves.

