Trees and Random Forest Example

Jimmy G. Moore

April 9, 2018

Use the training data set to build a regression tree with ozone (03) as the outcome

To start we will upload the training dataset to our environment. The training data is found at https://bit.ly/2JaHsJn (https://bit.ly/2JaHsJn).

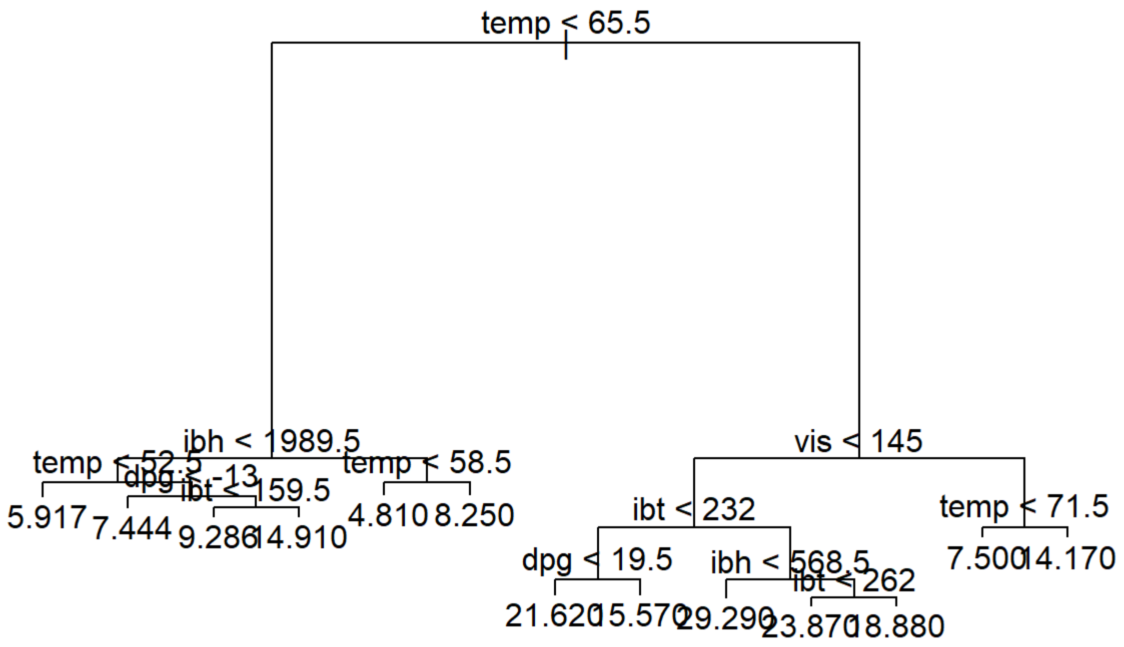
```
#Import Training Set
trainingSet = read.csv("C:/Users/James Moore/Documents/Random_Forrest/dataFile/smogData_train.cs
v")
trainingSet$X = NULL
```

Because we want to avoid overfitting our model we will break the data set into a training and test set that will be used for cross validation. In our case, we will use 5-fold cross validation.

```
#format data into training and test set
train = trainingSet[-seq(from = 5,to=255,by=5),]
test = trainingSet[seq(from = 5, to= 255,by = 5),]
```

Now that we have our data formatted properly we will begin building our tree model to predict ozone for our data. Let’s start by just building a tree with all variables as possible predictors. This will give us a good starting point to prune our model.

```
library("tree")
full_tree = tree(O3~.,data = train)
plot(full_tree)
text(full_tree)
```



```
summary(full_tree)
```

```
##
## Regression tree:
## tree(formula = O3 ~ ., data = train)
## Variables actually used in tree construction:
## [1] "temp" "ibh"  "dpg"  "ibt"  "vis"
## Number of terminal nodes:  13
## Residual mean deviance:  11.26 = 2151 / 191
## Distribution of residuals:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -11.290  -1.810  -0.250   0.000   2.133  11.120
```

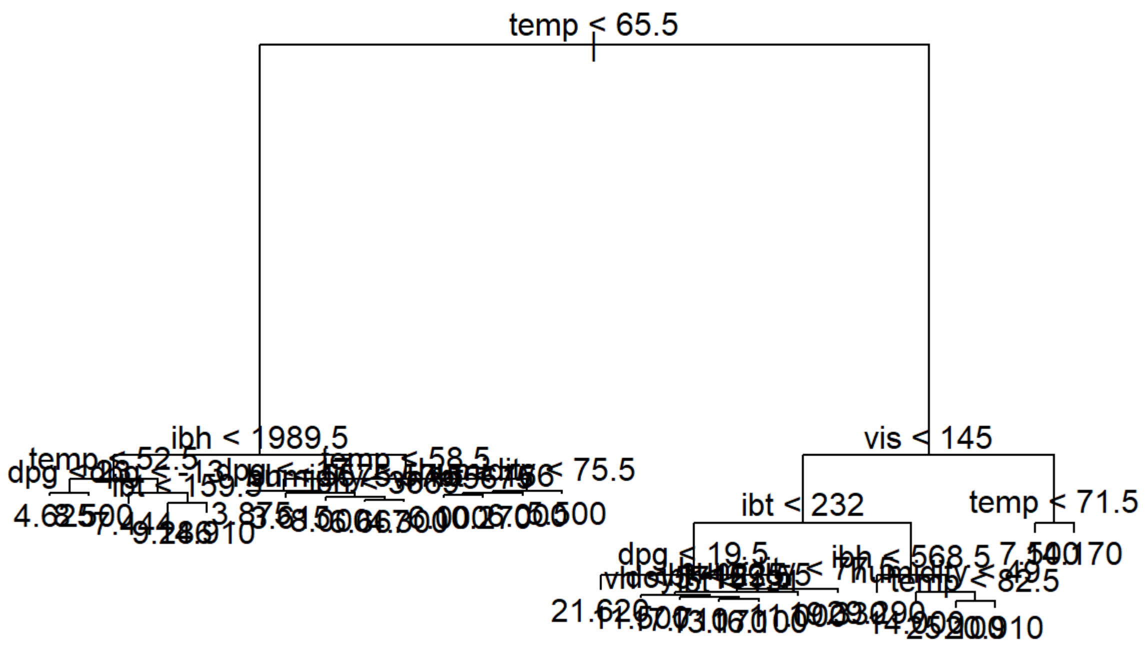
Now we will see how the predictions work on our test set by performing cross validation and using the MSE to evaluate performance.

```
#Make function to calculate MSE
MSE = function(estPred,test){
  squaredError = rep(NA,length(estPred))
  for (i in 1:length(estPred)) {
    squaredError[i] = (estPred[i]-test[i,1])^2
  }
  return(mean(squaredError))
}

#Evaluate Performance
performanceResult = predict(full_tree,test)
MSE(performanceResult,test)
```

```
## [1] 20.7104
```

Our initial MSE is 20.71 which is OK but we would like to prune and control our tree to see if we can reduce this number and/or make the tree a little more simple for interpretation and generalization purposes. To start, we will also modify `nobs` , `mincut` , `minsize` , and `mindev` from `tree.control` to control our tree.

```
tree_control = tree(O3~.,data = train, control = tree.control(nobs = 204,mincut = 2,minsize = 12
,mindev = 0.001))

plot(tree_control)
text(tree_control)
```



```
summary(tree_control)
```

```
##
## Regression tree:
## tree(formula = O3 ~ ., data = train, control = tree.control(nobs = 204,
##     mincut = 2, minsize = 12, mindev = 0.001))
## Variables actually used in tree construction:
## [1] "temp"     "ibh"      "dpg"      "ibt"      "vh"       "humidity"
## [7] "vis"      "doy"
## Number of terminal nodes:  27
## Residual mean deviance:  8.769 = 1552 / 177
## Distribution of residuals:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -11.290  -1.300  -0.050   0.000   1.308   9.091
```

```
#Evaluate Performance
performanceResult = predict(tree_control,test)
MSE(performanceResult,test)
```

```
## [1] 18.54274
```

Here we can see an reduction in overall MSE from 20.71 to 18.54, however, this is at the expense of adding more variables. Further analysis can be done to see whether or not the reduction in MSE is worth the addition of 4 more variables to our model. However, this is beyond the scope of our current analysis.

Abandon CART, focus on Random Forest

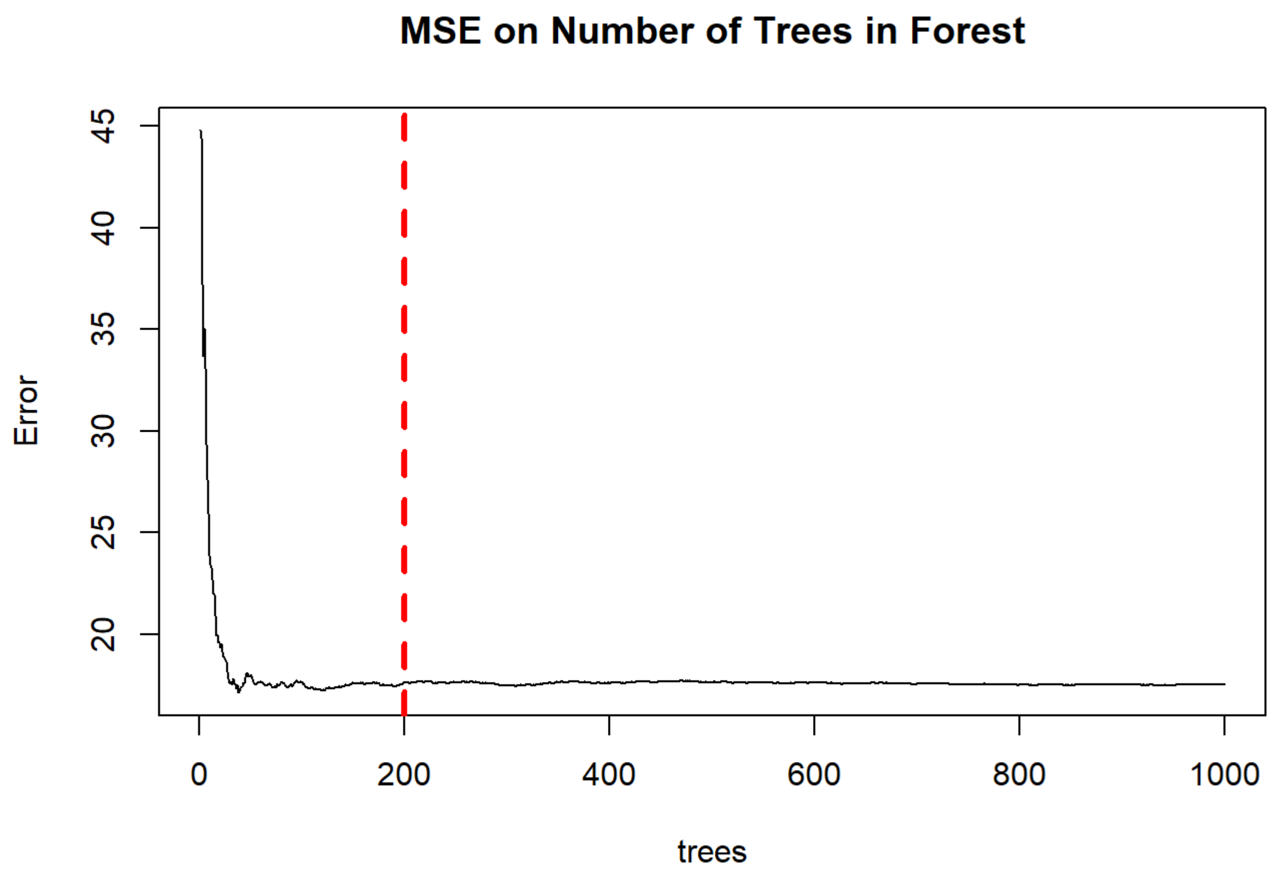
We will now shift focus from a single CART Model to a random forest. Ensemble methods are generally better at producing a smaller prediciton variance and for that reason we would like to build one to predict `O3` for or data.

```
set.seed(1234)
library(randomForest)

#Default Random Forest
forest = randomForest(O3~.,data = train, ntree = 1000, importance = TRUE)

#Evaluate Performance
performanceResult = predict(forest,test)
MSE(performanceResult,test)
```

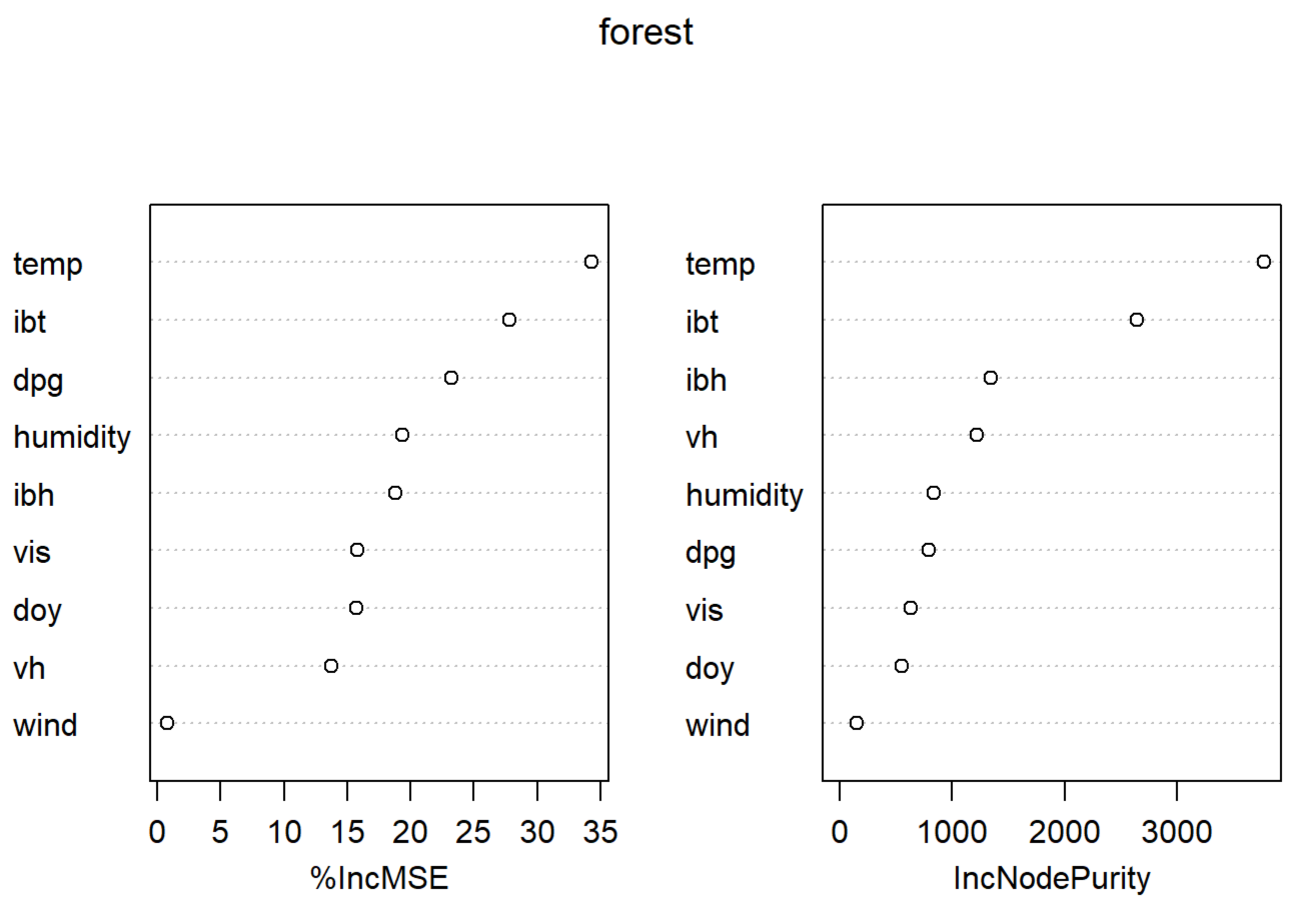
```
## [1] 8.708602
```

This is a major improvement from our single tree model. Having said that, the previous random forest was a very basic model using many default values. Now we will make some adjustments to see if we can improve our predictions.

```
plot(forest,main = "MSE on Number of Trees in Forest")
abline(v = 200, col="red", lwd=3, lty=2)
```



```
varImpPlot(forest)
```



We can see that the error stabilizes around 200 trees in the model. For that reason we will include that in our final model.

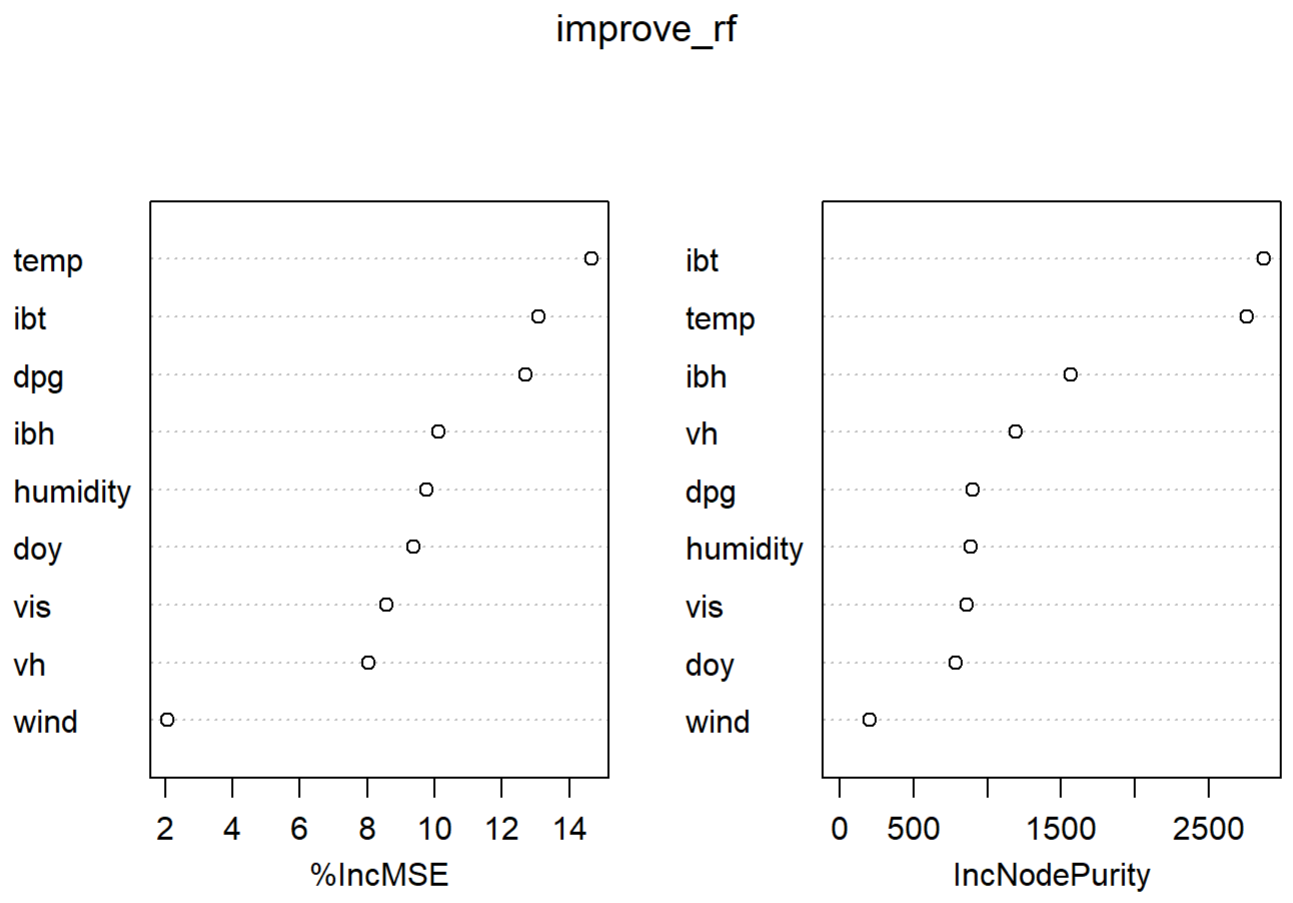
```
set.seed(1234)

#improved random forest
improve_rf = randomForest(O3~.,data = train, ntree = 200, importance=TRUE,
                          mtry=2,
                          nodesize=3,
                          maxnodes = 100,
                          nPerm = 3,
                          corr.bias = TRUE)

#Evaluate Performance
performanceResult = predict(improve_rf,test)
MSE(performanceResult,test)
```

```
## [1] 8.03463
```

Here we can see that our imporved forest has a large drop in MSE (about 1). This is due primarily to changing the values of `mtry` , `nodesize` , `maxnodes` , `nPerm` , and controlling or correllation biases between predictor varialbes with `corr.bias = TRUE` . The performance is pretty good with this model. Unofrtunately, this random forest may be hard to interpret, as expected by the black box nature of these models, however, there is a significant difference in

MSE between our single tree model and our random forest model. The individual trees in the random forest are much more complex than that of our single tree but we are more interested in prediction than we are interpretation so that is what we will run with.

```
varImpPlot(improve_rf)
```



We can also see from the above variable importance plot that `temp` , `ibt` , and `dpg` are the most important variables in our model. Notice these are different then observed in our full, untuned model

Now we will apply our random forest to the prediction data and see how we did.

Start by importing the prediction data

```
#Import prediciton data
#Import Training Set
testSet = read.csv("C:/Users/James Moore/Documents/Random_Forrest/dataFile/smogDataPredict.csv")
testSet$X = NULL
```

Now we will run prediction with our final Model on the unknown data to see what we can expect

```
finalPredictions = predict(improve_rf,testSet)
#write predictions to file
write.table(finalPredictions,
            "C:/Users/James Moore/Documents/Random_Forrest/predicted_O3_Values.txt",row.names =
F)
```

Use these final prediction values to evaluate the accuracy of our model. For ease of use the values have been writen to a txt file and is attached with the homework submission.