Problem Set 9

Noah Estrada-Rand,

11/12/2019

a) Setting the Seed

```
set.seed(2019)
```

b) Creating Test and Train Data

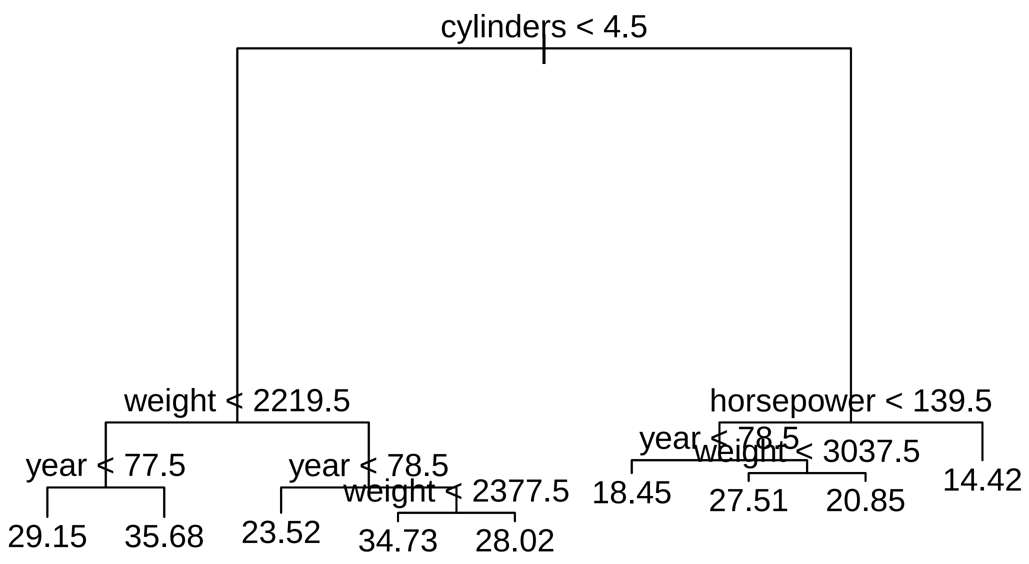
```
train_index <- sample(1:nrow(Auto),floor(.75*nrow(Auto)),replace = FALSE)
auto_train <- Auto[train_index,]
auto_test <- Auto[-train_index,]
```

c) Fitting the Tree

```
library(tree)
auto_reg_tree <- tree(mpg~cylinders+ displacement + horsepower + weight +
                        acceleration + year + origin,data=  auto_train)
```

d) Plot of the Regression Tree for MPG

```
plot(auto_reg_tree)
text(auto_reg_tree,pretty = 0)
```



e)

The tree shows that the most important variable in the tree model is the variable of cylinders. Next weight and horsepower were the next most important. Later year was used to further build the tree. In total, there are 9 leaves and 3 splits on each side.

f) Calculating Train and Test MSE

```
train_preds <- predict(auto_reg_tree)
test_preds <- predict(auto_reg_tree,newdata = auto_test)
head(train_preds)
```

```
##      283      126       70      239      275      120
## 28.02083 18.45088 14.41719 29.14865 18.45088 23.51731
```

```
head(test_preds)
```

```
##        2        4        5       15       18       25
## 14.41719 14.41719 14.41719 23.51731 18.45088 18.45088
```

```
train_mse <- mean((train_preds- auto_train$mpg)^2)
test_mse <- mean((test_preds- auto_test$mpg)^2)
print(train_mse)
```

```
## [1] 8.076883
```

```
print(test_mse)
```

```
## [1] 12.21892
```

g) Tree Cross Validation

```
tree_mod_cv <- cv.tree(auto_reg_tree)
best_tree_index <- which.min(tree_mod_cv$dev)
```

From the above cross validation it would appear that the optimal tree size for minimizing the error is a tree of 9 splits.

```
tree_mod_cv$size[best_tree_index]
```

```
## [1] 9
```

h)

```
pruned_tree <- prune.tree(auto_reg_tree,best = 9)
pruned_train_preds <- predict(pruned_tree)
pruned_test_preds <- predict(pruned_tree,newdata = auto_test)
```

Finding the MSE of the model on the Train and Test Datasets

```
mse_train <- mean((pruned_train_preds - auto_train$mpg)^2)
mse_test <- mean((pruned_test_preds - auto_test$mpg)^2)
print(mse_train)
```

```
## [1] 8.076883
```

```
print(mse_test)
```

```
## [1] 12.21892
```

i) Bagging Models and MSE

```
dim(auto_train)
```

```
## [1] 294   9
```

```
bag_mod <- randomForest(mpg~cylinders+ displacement + horsepower + weight +
                          acceleration + year + origin,
                        data = auto_train,
                        mtry = 7,
                        ntrees = 500,
                        importance = TRUE)
bag_train_preds <- predict(bag_mod)
bag_test_preds <- predict(bag_mod,newdata = auto_test)

mse_train <- mean((bag_train_preds - auto_train$mpg)^2)
mse_test <- mean((bag_test_preds - auto_test$mpg)^2)

print(mse_train)
```

```
## [1] 7.999761
```

```
print(mse_test)
```

```
## [1] 7.268231
```

j) Estimating Random Forest with Mtry 3

```
random_forest_mod <- randomForest(mpg~cylinders+ displacement + horsepower + weight +
                                    acceleration + year + origin,
                                  data = auto_train,
                                  mtry = 3,
                                  ntrees = 500,
                                  importance = TRUE)
```

k) How the Mtry Parameter Does

The mtry parameter denotes how many variables to sample to consider at each split to optimize that split. It helps with variations for random forest models by forcing the splits to be different instead of using the same set of variables for each iteration of the cross validation.

l) Getting Predictions from Random Forest on Train and Test Sets

```
rando_preds_train <- predict(random_forest_mod)
rando_preds_test <- predict(random_forest_mod,newdata =  auto_test)
train_mse <- mean((rando_preds_train - auto_train$mpg)^2)
test_mse <- mean((rando_preds_test - auto_test$mpg)^2)
print(train_mse)
```

```
## [1] 8.118358
```

```
print(test_mse)
```

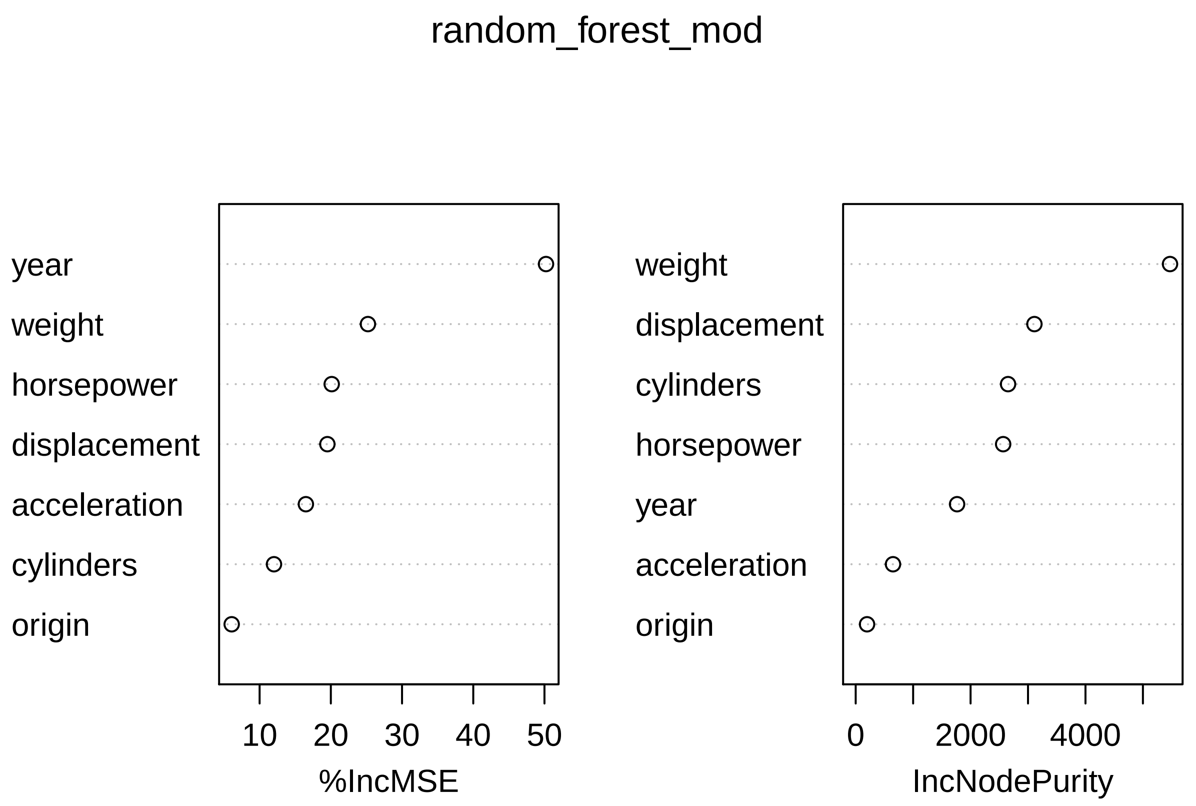
```
## [1] 7.177143
```

m) Importance of Variables

```
importance(random_forest_mod)
```

```
##                 %IncMSE IncNodePurity
## cylinders    12.017690     2652.4215
## displacement 19.516720     3111.0198
## horsepower   20.127104     2567.1287
## weight       25.212097     5471.9002
## acceleration 16.497911      649.7267
## year         50.218004     1765.2524
## origin        6.078645      198.0910
```

```
varImpPlot(random_forest_mod)
```



From the above visualization and information, it becomes clear that year, weight, and horsepower are the most important variables in building the trees within random forest. The Year variable, if left out of the tree would lead to an increase in MSE of 50%, an extremely large increase. Additionally, weight would lead to a 25.21% increase in MSE and horsepower would lead to an increase of 20.12%.

n) Which Models Performed Best and Why

Of all the models studied in this assignment, the bagging and random forest models performed the best in terms of MSE. Both had substantially lower mse for both training and test data than the original tree that was then pruned. This is the case for the simple fact that bagging and random forest introduce greater variabilitiy in the trees created whereas pruning only considers one tree. Thus, bagging and random forest were able to consider possible trees that may not have been observed in the first iteration of creating the tree. As a result, the models not only better captured the training and test data but also reduced the amount of overfitting that occurred.