Math 154 Tree-based Methods Lab

Roja, Franklin, Alejandra

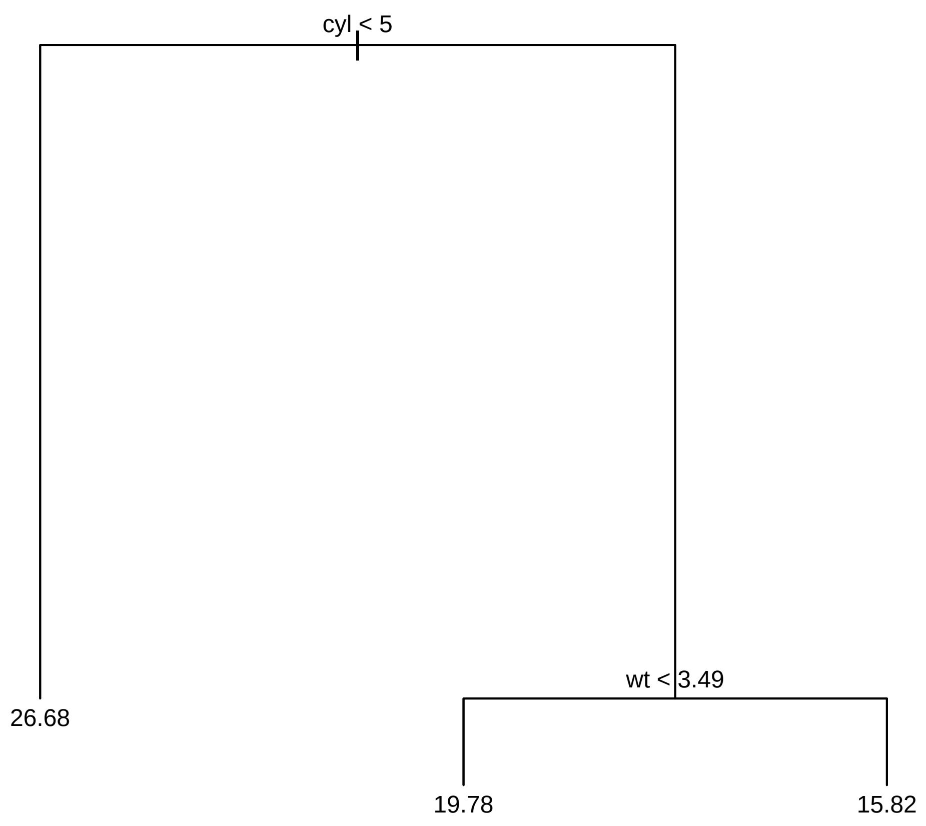
November 15, 2016

In addition to using the kaggle data, load the data set mtcars into R via the command data(mtcars). Try to predict the fuel efficiency (mpg) via a regression tree, and try to predict the transmission type (automatic or manual) via a classification tree. You may want to compare the regression tree to a linear model. plot(mtcars) will give you all the pairwise scatter plots. Notice that most of the relationships with MPG are non-linear. Additionally, R has a package called randomForest. The most useful function, which implements the algorithm discussed in class, is of the same name. Compare the classification rates for a random forest to that of a simple tree.

```r
# install.packages("tree")
# install.packages("randomForest")
# install.packages('reprtree')
library(tree)
library(randomForest)

## randomForest 4.6-12
## Type rfNews() to see new features/changes/bug fixes.

library(datasets)
# library(reprtree)
data(mtcars)
```

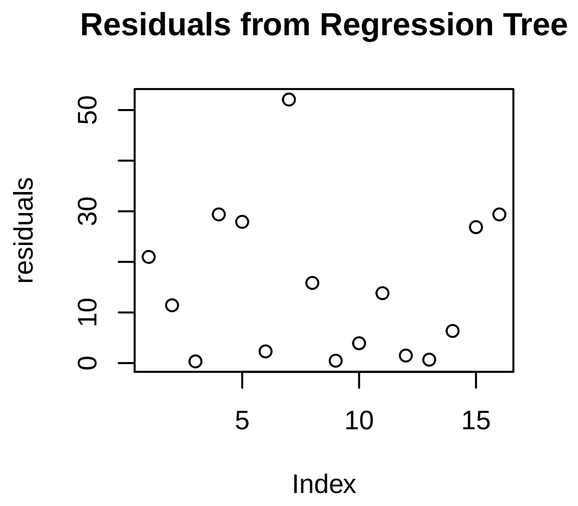
In order to predict the fuel efficiency we implemented a regression tree and a linear regression model to later compare results. We divided the data into a training and test set. Since the sample size is only thirty-two, the two sets only had sixteen observations. One method we used to compare the performance of both models was residual sum of squares. We consistently found that predictions made with a linear regression model had a lower residual sum of squares than those made with a regression tree. It may be due to there only being three nodes and therefore, three averages which new observations can take on. The variables that the regression tree selected were the number of cylinders and the weight of the car to create partitions.

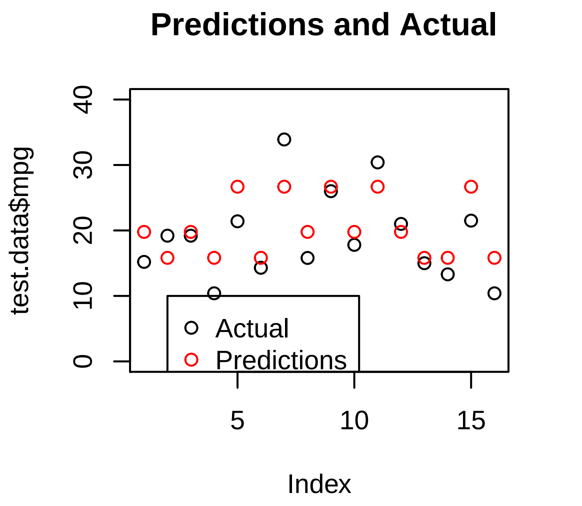
```r
# Regression Tree
test.index = sample(c(1:nrow(mtcars)),nrow(mtcars)/2 )
training.index= c(1:nrow(mtcars))[-test.index]
test.data = mtcars[test.index,]
training.data = mtcars[training.index,]
cars.regression <- tree(mpg ~ cyl+disp+hp+drat+wt+qsec+vs+am+gear+carb, data=training.data)
plot(cars.regression )
text(cars.regression , cex=.75)
```

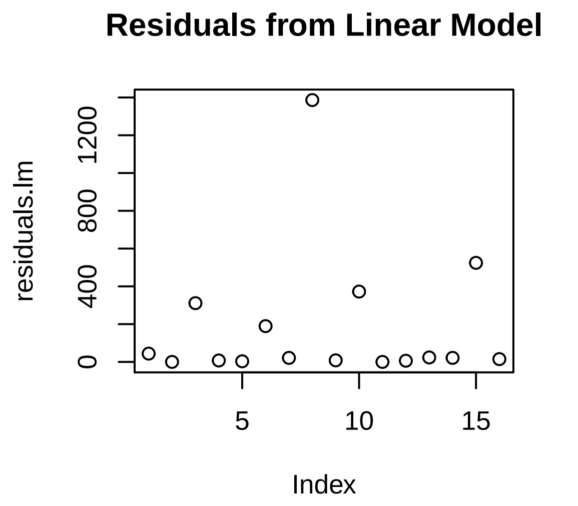


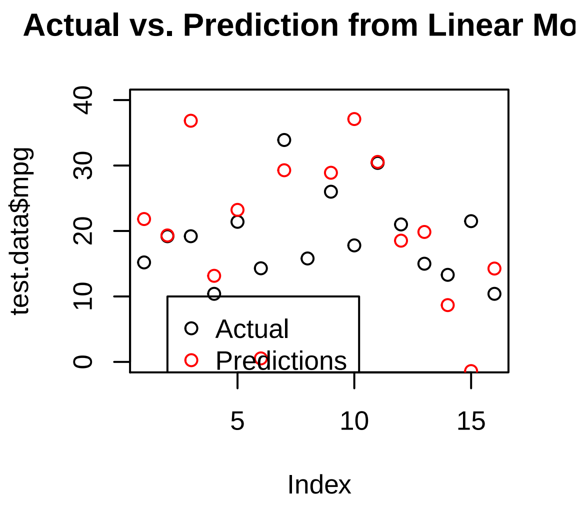
```
my.prediction <- predict(cars.regression, test.data)
# find RSS
residuals = (test.data$mpg - my.prediction)^2
sum(residuals^2)

## [1] 7015.598

# plot residuals
par(mfrow=c(2,2))
plot(residuals, main= 'Residuals from Regression Tree')
# plot of actual and predictions
plot(test.data$mpg, ylim = c(0,40), main = ' Predictions and Actual')
points(my.prediction, col = 'red')
legend(2, 10, legend=c("Actual", "Predictions"),
       col=c("black", "red"),pch = 1)
#####
# Comparing to a Linear model
cars.lm <- lm ( mpg ~ cyl+disp+hp+drat+wt+qsec+vs+am+gear+carb, data= training.data)
prediction.lm <- predict(cars.lm, test.data)
# find RSS
```

```r
residuals.lm = (test.data$mpg - prediction.lm)^2
sum(residuals.lm^2)

## [1] 2472197

# plot residuals
plot(residuals.lm, main = "Residuals from Linear Model")
# plot of actual and predictions
plot(test.data$mpg, main= " Actual vs. Prediction from Linear Model", ylim = c(0,40))
points(prediction.lm, col = 'red')
legend(2, 10, legend=c("Actual", "Predictions"),
       col=c("black", "red"),pch = 1)
```



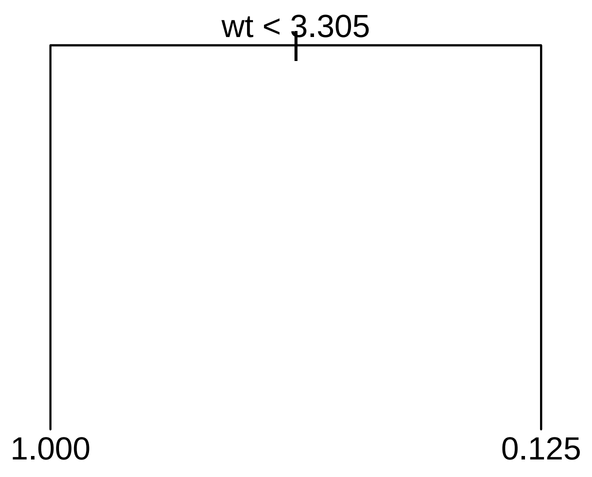




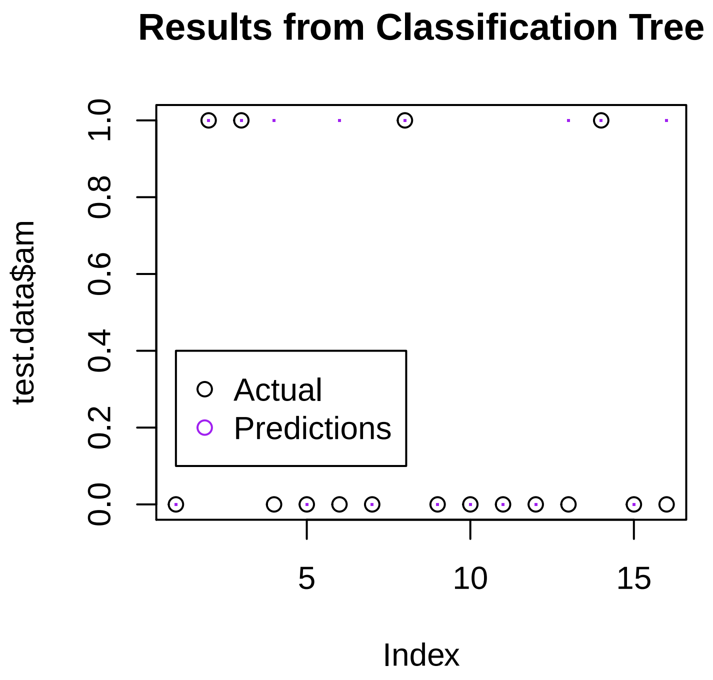


We used Classification and Random Forests to predict the transmission type of a car. The classification tree had two terminal nodes and used *weight* as a point of partition. Since the response variable was binary we calculated the proportion of times each method predicted accurately. The accuracy of the predictions from both methods were about the same. To compare the success rates of both we ran multiple simulations and recorded the proportion of successes. After running the simulation below several times we observed that random forests provided consistently had less incorrect predictions.

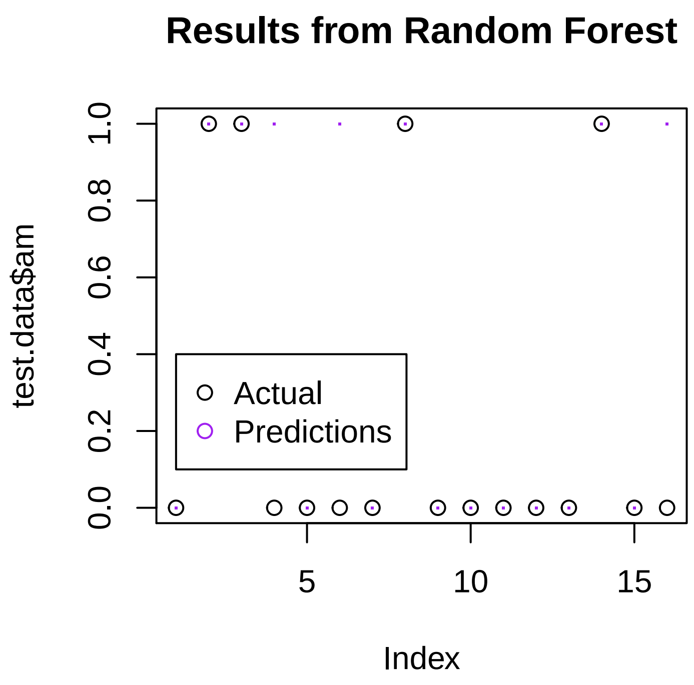
```r
# Classification Tree
# Use a classification tree to predict transmission type of car
# Transmission (0 = automatic, 1 = manual)

incorrect.predict.Forest = c()
incorrect.predict.class = c()
for (i in c(1:40)){
test.index = sample(c(1:nrow(mtcars)),nrow(mtcars)/2 )
training.index= c(1:nrow(mtcars))[-test.index]

test.data = mtcars[test.index,]
training.data = mtcars[training.index,]

cars.class<-tree(am ~ mpg+cyl+disp+hp+drat+wt+qsec+vs+gear+carb, data=training.data)
summary(cars.class)

my.prediction.class <- predict(cars.class, test.data)
my.prediction.class[(my.prediction.class < 0.5)] = 0
my.prediction.class[(my.prediction.class > 0.5)] = 1
# how many incorrect predictions
incorrect.predict.class[i] = sum(abs(my.prediction.class - test.data$am) > .5)

####
# Using a Random Forest
cars.Forest <- randomForest(as.factor(am) ~ mpg+cyl+disp+hp+drat+wt+qsec+vs+gear+carb, data=training.data ,
                            ntree=128)
prediction.Forest<- predict(cars.Forest, test.data)
# how many incorrect predictions
incorrect.predict.Forest[i] = sum(abs((as.numeric(prediction.Forest) - 1) - test.data$am) > .5)
}

plot(cars.class, , main= 'Classification Results')
text(cars.class)
```



```r
plot(test.data$am, ylim = c(0,1), main= 'Results from Classification Tree')
points(my.prediction.class,col = 'purple', pch = ".")
legend(1, 0.4, legend=c("Actual", "Predictions"),
       col=c("black", "purple"),pch = 1)
```



```r
plot(test.data$am, main= 'Results from Random Forest')
points(as.numeric(prediction.Forest)-1,col = 'purple', pch = ".")
legend(1, 0.4, legend=c("Actual", "Predictions"),
       col=c("black", "purple"),pch = 1)
```



```r
incorrect.predict.Forest

##  [1] 0 5 5 4 2 2 3 3 4 4 2 3 5 2 2 1 5 5 1 1 4 4 4 1 3 1 4 2 6 1 4 5 5 3 4
## [36] 2 3 3 1 3

incorrect.predict.class

##  [1] 1 6 4 2 2 3 2 4 4 4 4 5 5 3 3 3 6 3 2 2 5 0 5 1 4 2 2 2 4 2 6 2 2 3 4
## [36] 3 3 3 2 4

mean(incorrect.predict.Forest)

## [1] 3.05

mean(incorrect.predict.class)

## [1] 3.175
```