Homework 5

Alex Soupir

October 13, 2019

Packages: vcd, lattice, randomForest, party, partykit, mboost, TH.data, ipred, rpart, randomForest, mlbench, tidyverse, boot, adabag

Collaborators:

Answer all questions specified on the problem and include a discussion on how your results answered/addressed the question.

Submit your **.rmd** file with the knitted **PDF** (or knitted Word Document saved as a PDF). If you are having trouble with .rmd, let us know and we will help you, but both the .rmd and the PDF are required.

This file can be used as a skeleton document for your code/write up. Please follow the instructions found under Content for Formatting and Guidelines. No code should be in your PDF write-up unless stated otherwise.

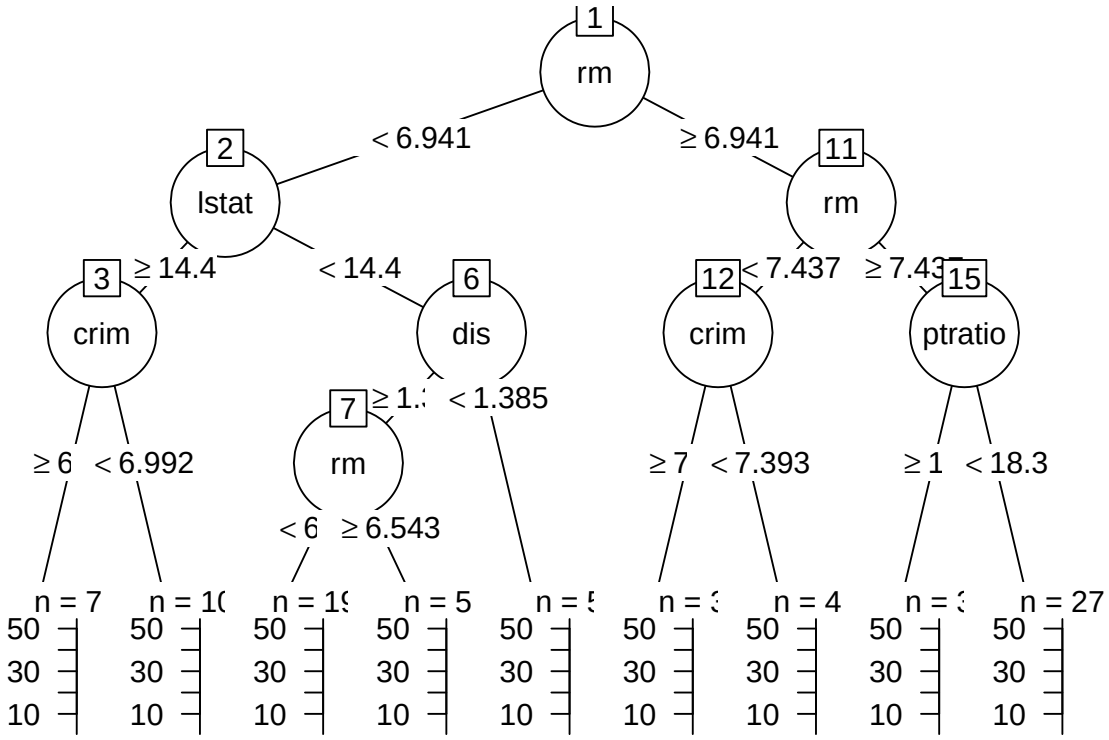
For any question asking for plots/graphs, please do as the question asks as well as do the same but using the respective commands in the GGPLOT2 library. (So if the question asks for one plot, your results should have two plots. One produced using the given R-function and one produced from the GGPLOT2 equivalent). This doesn't apply to questions that don't specifically ask for a plot, however I still would encourage you to produce both.

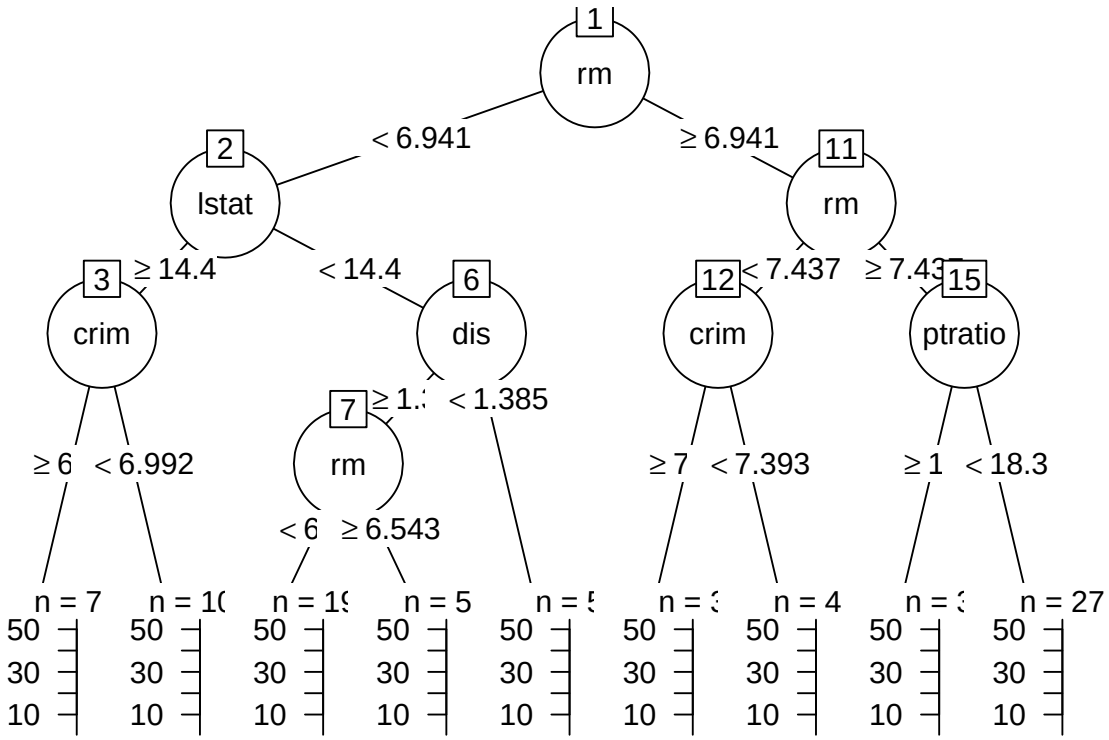
You do not need to include the above statements.

Please do the following problems from the text book R Handbook and stated.

1. The **BostonHousing** dataset reported by Harrison and Rubinfeld (1978) is available as data.frame package **mlbench** (Leisch and Dimitriadou, 2009). The goal here is to predict the median value of owner-occupied homes (medv variable, in 1000s USD) based on other predictors in the dataset. Use this dataset to do the following

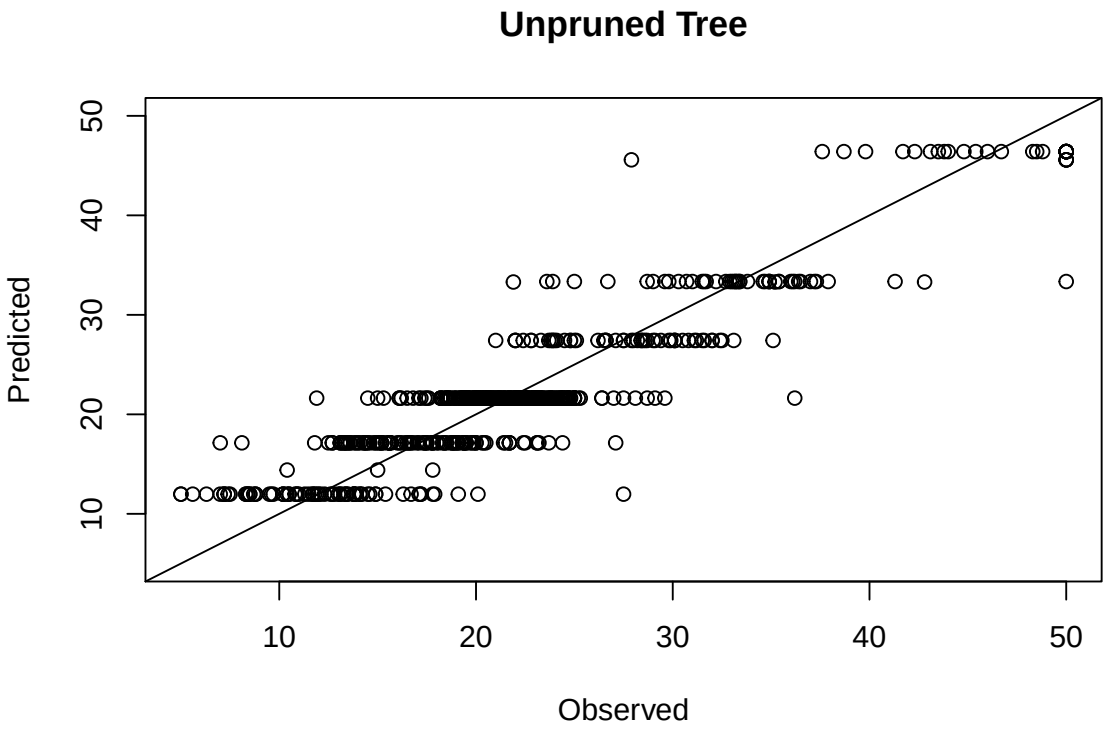
   a.) Construct a regression tree using rpart(). The following need to be included in your discussion. How many nodes did your tree have? Did you prune the tree? Did it decrease the number of nodes? What is the prediction error (calculate MSE)? Provide a plot of the predicted vs. observed values. Plot the final tree.

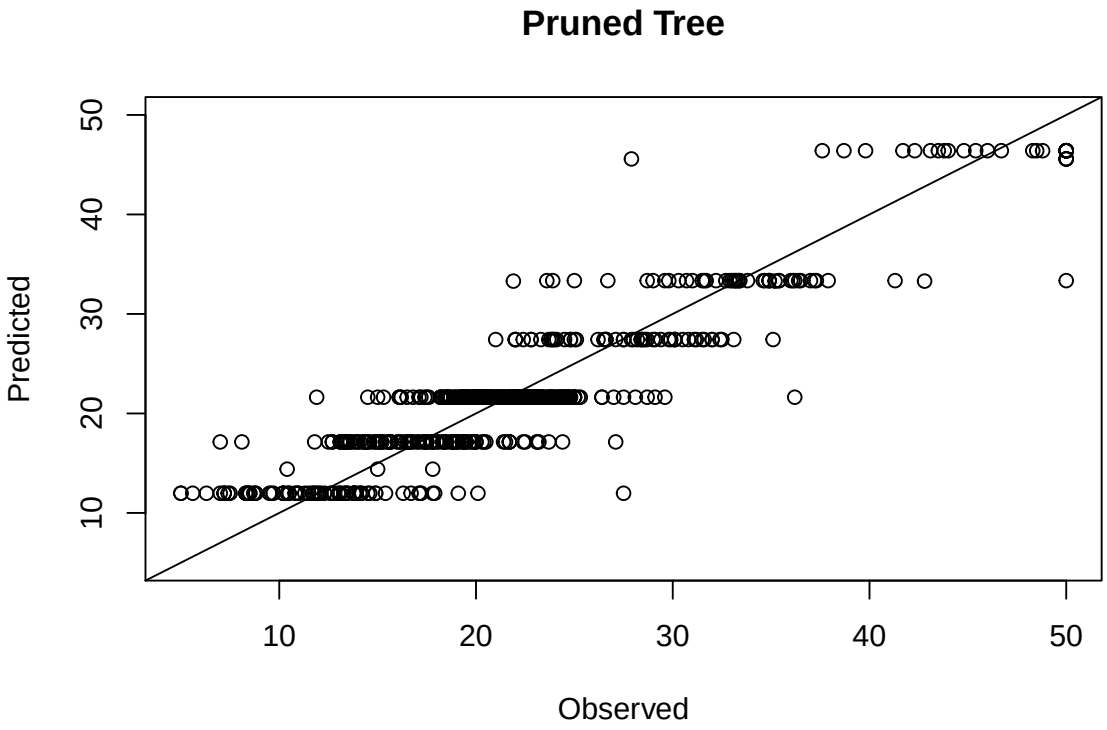




Tree without Pruning:

```
## [1] 12.71556
##
## Tree with Pruning:
## [1] 12.71556
```





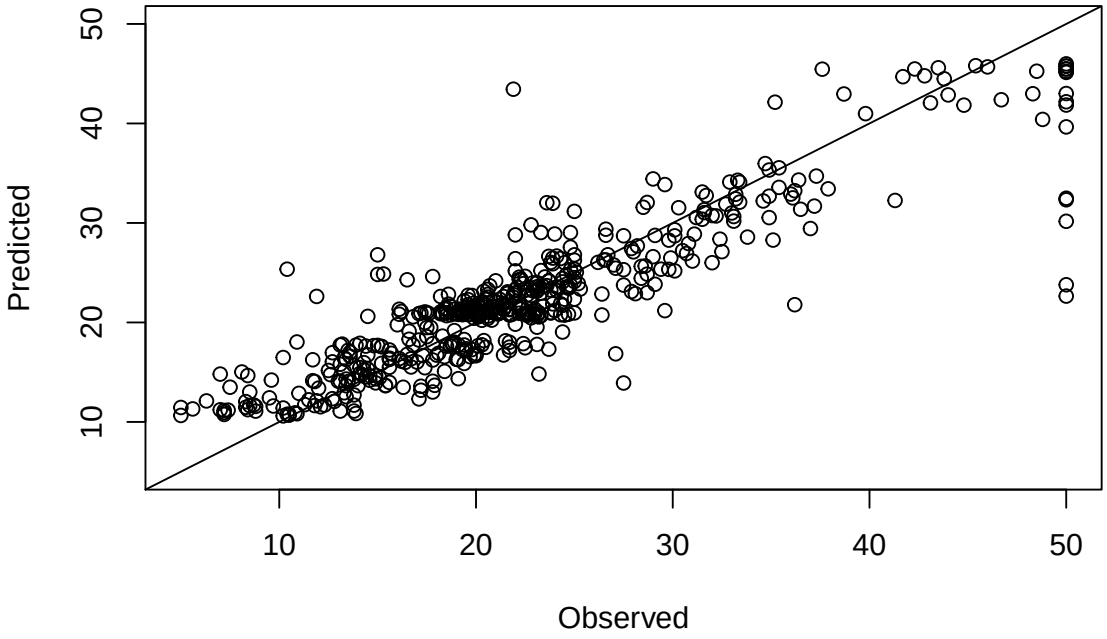
I created a base tree as well as a pruned tree after setting seed to *333*. Both trees are using all parameters from the data frame to build the tree. Both trees had 8 internal nodes and 8 terminal nodes and both had the same MSE of 12.71556. Looking at the predicted vs observed values, the variance increases as the medv increases.

b) Perform bagging with 50 trees. Report the prediction error (MSE). Provide the predicted vs observed plot.

```
##
## lstat    rm
##    19    31
```

```
##
## 50 Tree Bagging MSE:
```

```
## [1] 17.02426
```



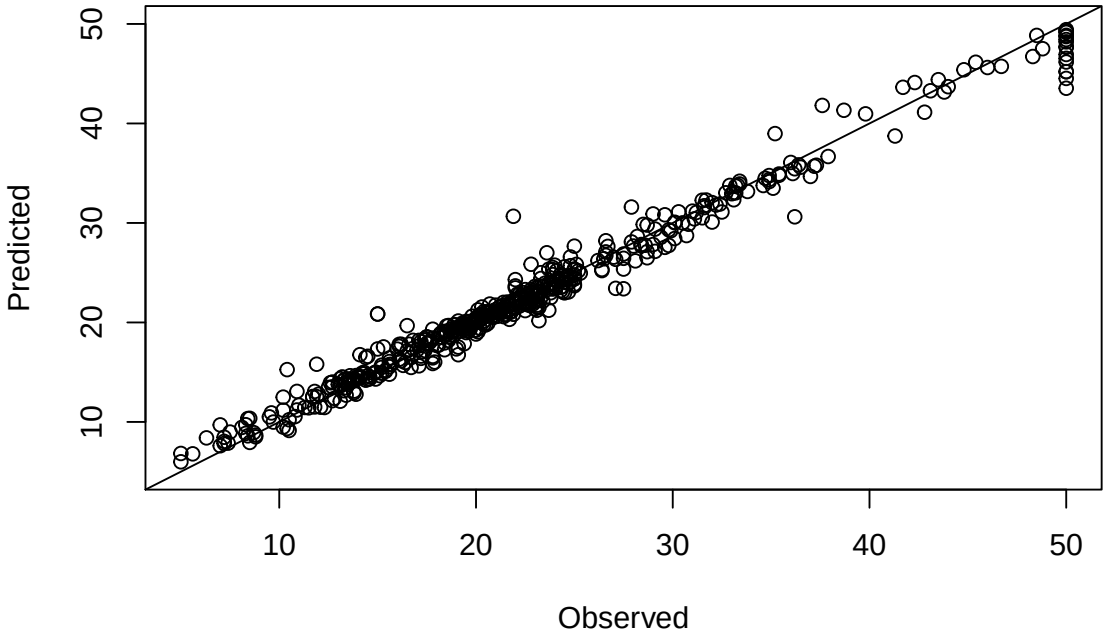
The MSE from bagging 50 trees is 17.02426 with seed *333*.

c) Use randomForest() function in R to perform bagging. Report the prediction error (MSE). Was it the same as (b)? If they are different what do you think caused it? Provide a plot of the predicted vs. observed values.

```
##
## Call:
##  randomForest(formula = medv ~ ., data = bhous, mtry = 13)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 13
##
##           Mean of squared residuals: 10.40821
##                     % Var explained: 87.67

##
## MSE from RandomForest() bagging:

## [1] 1.810938
```



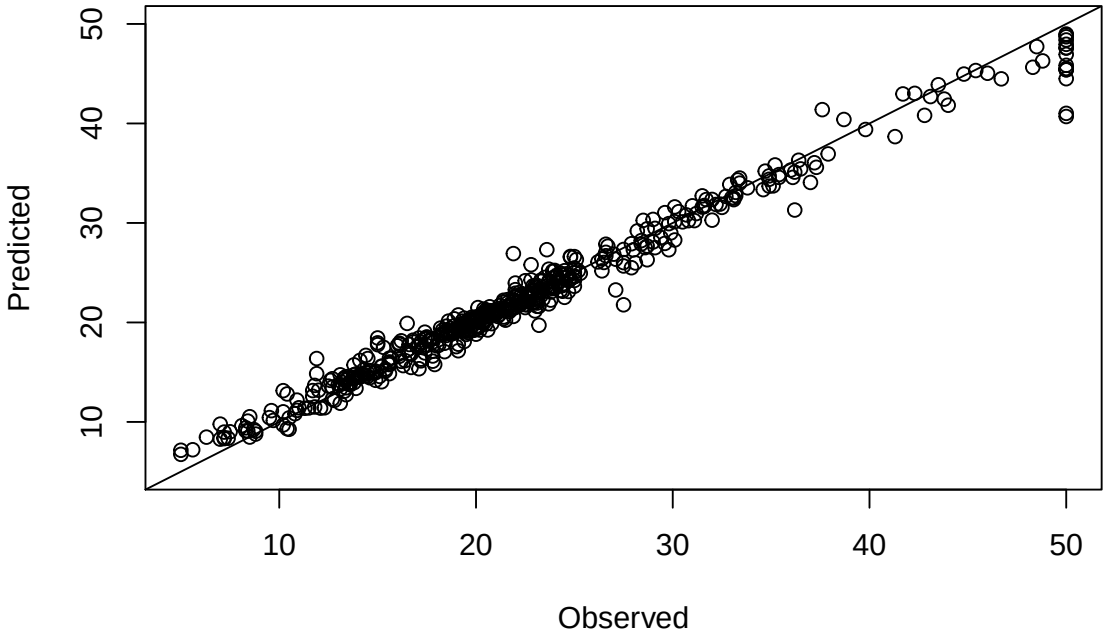
The random forest with *333* as the seed using all of the features (bagging by using *mtry = 13*) method provided a much lower MSE value than the 50 tree ensemble or the single tree (1.811 v 17.024 v 12.716). This is most likely due to the number of trees being created and used for predictions since the random forest function is creating 500 trees instead of 50 or just 1.

d) Use randomForest() function in R to perform random forest. Report the prediction error (MSE). Provide a plot of the predicted vs. observed values.

```
##
## Call:
##  randomForest(formula = medv ~ ., data = bhous)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 4
##
##           Mean of squared residuals: 9.79886
##                     % Var explained: 88.39

##
## MSE from RandomForest() bagging:

## [1] 1.958057
```



By creating a random forest with *333* as the seed and not telling it how many of the features to used to create the trees, the MSE went up slightly from bagging (1.958 v 1.811). Looking at the predicted vs observed plot, there were some values in the middle that moved closer to the x=y line, but values on the high end of medv spread out slightly more than with bagging, which increased the MSE.

e) Provide a table containing each method and associated MSE. Which method is more accurate?

```
##                    Method Mean_Square_Error
## 1 Single Tree (un-pruned)         12.715559
## 2    Single Tree (pruned)         12.715559
## 3        50 Tree Ensemble         17.024265
## 4                 Bagging          1.810938
## 5           Random Forest          1.958057
```

2. Consider the glacoma data (data = "**GlaucomaM**", package = "**TH.data**").

a) Build a logistic regression model. Note that most of the predictor variables are highly correlated. Hence, a logistic regression model using the whole set of variables will not work here as it is sensitive to correlation.

```
glac_glm <- glm(Class ~., data = GlaucomaM, family = "binomial")
#warning messages  -- variable selection needed
```

The solution is to select variables that seem to be important for predicting the response and using those in the modeling process using GLM. One way to do this is by looking at the relationship between the response variable and predictor variables using graphical or numerical summaries - this tends to be a tedious process. Secondly, we can use a formal variable selection approach. The *step*() function will do this in R. Using the *step* function, choose any direction for variable selection and fit logistic regression model. Discuss the model and error rate.

```
#use of step() function in R
?step
glm.step <- step(glac_glm)
```

Do not print out the summaries of every single model built using variable selection. That will end up being dozens of pages long and not worth reading through. Your discussion needs to include the direction you chose. You may only report on the final model, the summary of that model, and the error rate associated with that model.

```
##
## Call:
## glm(formula = Class ~ phct + phci + hvc + mr + rnf + emd, family = "binomial",
##     data = glau1)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.13784  -0.63261  -0.00377   0.60628   2.50871
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)    6.968      2.289   3.044 0.002337 **
## phct           4.306      2.989   1.440 0.149759
## phci           9.242      2.643   3.497 0.000470 ***
## hvc           -5.405      1.981  -2.729 0.006350 **
## mr            -6.423      2.479  -2.591 0.009569 **
## rnf          -12.973      3.546  -3.659 0.000253 ***
## emd            9.688      2.525   3.836 0.000125 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 271.71  on 195  degrees of freedom
## Residual deviance: 154.13  on 189  degrees of freedom
## AIC: 168.13
##
## Number of Fisher Scoring iterations: 6

##
## MSE from step glm:

## [1] 0.4846939
```

The direction that I chose to go given that there are many metrics that correlate with each other is to remove those with a correlation greater than 0.5. The remaining features were than used in the step() function to determine the best combination. The final model created used *phct*, *phci*, *hvc*, *mr*, *rnf*, and *emd*. All features were significant at P<=0.01 besides *phct*. The AIC of the model is 168.13 which is better than the other model created within the step() function with AIC=169.92. This indicates the final model fits the response variable (glaucoma or normal) better than the first model. The mean squared error of this final model is 0.485, which seems relatively low but without other models to compare to is hard to tell whether this is good or poor.

b) Build a logistic regression model with K-fold cross validation (k = 10). Report the error rate.

```
## K-fold cross validation (K=10) error rate:
```

```
## [1] 0.2704082
```

The error from cross validation is 0.27 which is almost half of the using step for glm. This indicates that k-fold cross validation better predicts whether the patient has glaucoma or not."

c) Find a function (package in R) that can conduct the "adaboost" ensemble modeling. Use it to predict glaucoma and report error rate. Be sure to mention the package you used.

```
## MSE from whole data boosting:

## [1] 0

##
## MSE from train boosting:

## [1] 0.195122
```

Created a 80:20 train:test split because the model was perfectly fitting the data. With predicting if the patient has glaucoma the boosting model has an MSE of 0.195. This is better than both step glm and k-fold cross validation. I suspect this is due to the 100 trees that it uses to build the trees and it calculating the weight of each tree based on how well it performs. This is with using a control of *minsplit=10* as per the documentation for the *boosting* funciton.

d) Report the error rates based on single tree, bagging and random forest. (A table would be great for this).

```
##                           Method Mean_Square_Error
## 1                      Step GLM         0.4846939
## 2                     K-Fold CV         0.2704082
## 3 Adaptive Boosting: Whole Data         0.0000000
## 4             Adaptive Boosting         0.1951220
```

e) Write a conclusion comparing the above results (use a table to report models and corresponding error rates). Which one is the best model?

Of the above models, the adaptive boosting performs the best both in whole data training and train/test split training. This is likely do to each of the trees being given a weight based on its performance and then all trees being used in an ensemble to predict the patients glaucoma status. Next best after boosting is the K-fold cross validation model which performs about 1.5x worse than the AdaBoost model. Finally, the worse model is the glm which is almost twice as bad as the K-fold CV model. The glm model is the simplest of the models trained, and this explains why its performace isn't the 'best'.

f) From the above analysis, which variables seem to be important in predicting Glaucoma?

```
##      emd      hvc       mr       mv     phci     phct      rnf
## 15.79327 14.17464 12.01873 11.10487 16.15698 14.11649 16.63500
```

From the above analysis, the features that seem to be important are those included in creating the models; *phct*, *phci*, *hvc*, *mr*, *rnf*, and *emd*. The glm model gave the significance of the coefficients for each and *phct* wasn't significantly different from zero. The boosting function provides the importance of the features used in making the trees and *rnf* was the most important with *mv* being the least important.

Resources Used:

- StackOverflow
- datascienceplus.com
- rdocumentation.org