Statistical Data Mining I Homework 4

Question 1

In this problem we had to use the prostate data to carry out a best subset linear regression analysis. We had to compute the AIC, BIC, five- and tenfold cross-validation, and bootstrap .632 estimates of prediction error.

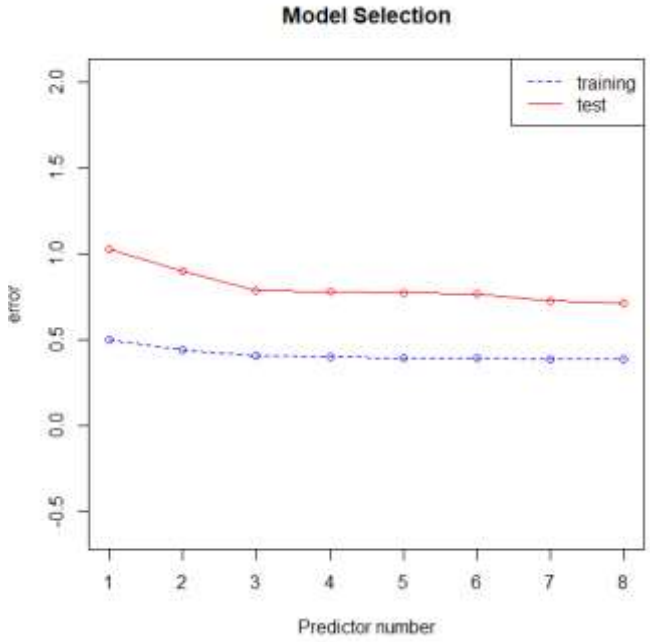
From the dataset we can see that there are 8 predictors with the lspa (biomarker) being the response. After applying best subset selection, we can see that the best 2 variable model are lcavol and lweight predictors.

```
Selection Algorithm: exhaustive
         lcavol lweight age lbph svi lcp gleason pgg45
1  ( 1 ) "*"    " "     " " " "  " " " " " "    " "
2  ( 1 ) "*"    "*"     " " " "  " " " " " "    " "
3  ( 1 ) "*"    "*"     " " " "  "*" " " " "    " "
4  ( 1 ) "*"    "*"     "*" " "  "*" " " " "    " "
5  ( 1 ) "*"    "*"     "*" " "  "*" " " "*"    " "
6  ( 1 ) "*"    "*"     "*" " "  "*" "*" "*"    " "
7  ( 1 ) "*"    "*"     "*" "*"  "*" "*" "*"    " "
8  ( 1 ) "*"    "*"     "*" "*"  "*" "*" "*"    "*"
```

We determined the cp and the bic (bayseian info criteria) and we obtained the minimum values for both cp and bic as these are the best ones. Both agree the optimum model is with 3 variable lcavol, lweight and svi.

Cp= 2.484056
BIC= -72.28763

We performed the selection based on the hold out method and used outmat to tell us the best number variable model. After we determined the training and testing error, as shown in figure 1 and the AIC and BIC values as shown in figure 2. We also determined the bootstrap .632 as shown in figure 3. The five- and tenfold cross-validation prediction error we determined as shown in figure 4.



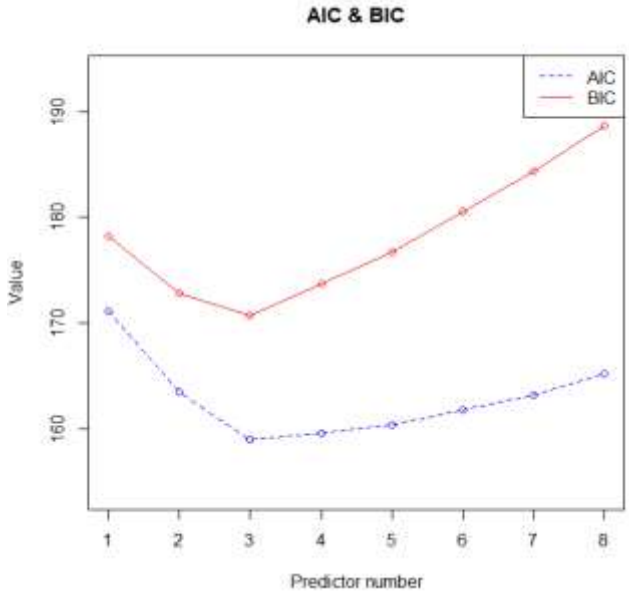


Figure 1: Training and testing prediction errors for the best variable model

Figure 2: AIC and BIC for the best variable model

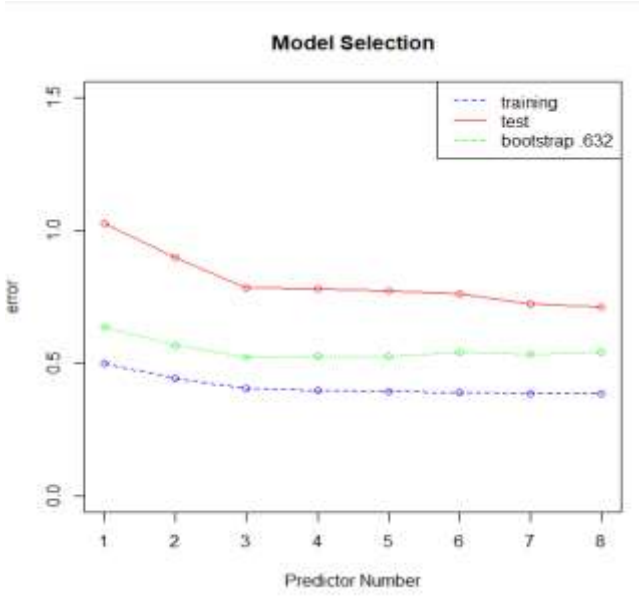


Figure 3: Training, testing error and booststrap.632 of the best variable model

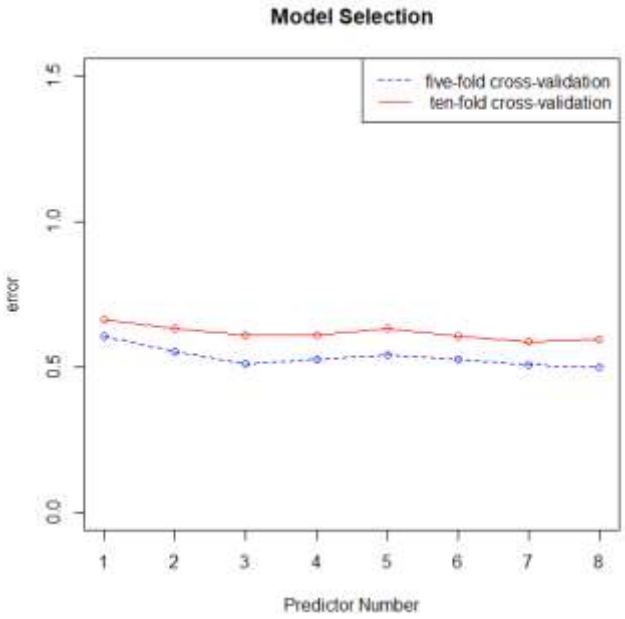


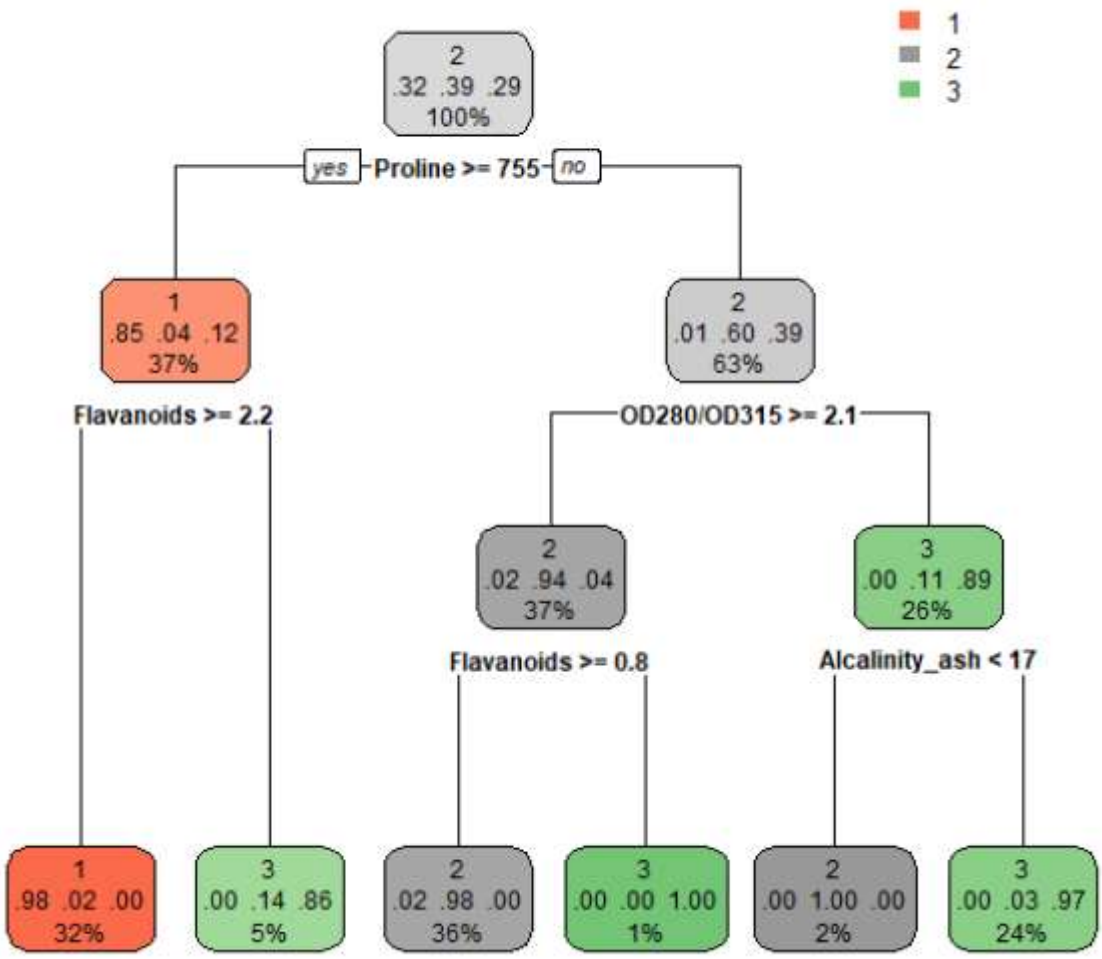
Figure 4: 5 fold cv and 10 fold cv for the best variable model

From the figures above:

The minimum BIC= 159.0590
The minimum AIC= 170.7781
The minimum bootstrap.632= 0.5213986
The minimum 5 fold cv = 0.5487116
The minimum 10 fold cv= 0.5867488

Question 2

In this problem we had to use the wine data from the UCI machine learning repository. These data are the results of a chemical analysis of 178 wines grown over the decade 1970-1979 in the same region of Italy, but derived from three different cultivars (Barolo, Grignolino, Barbera). There are 50 Barolo wines, 71 Grignolino wines, and 48 Barbera wines. We had to construct an appropriate-size classification tree for this dataset.



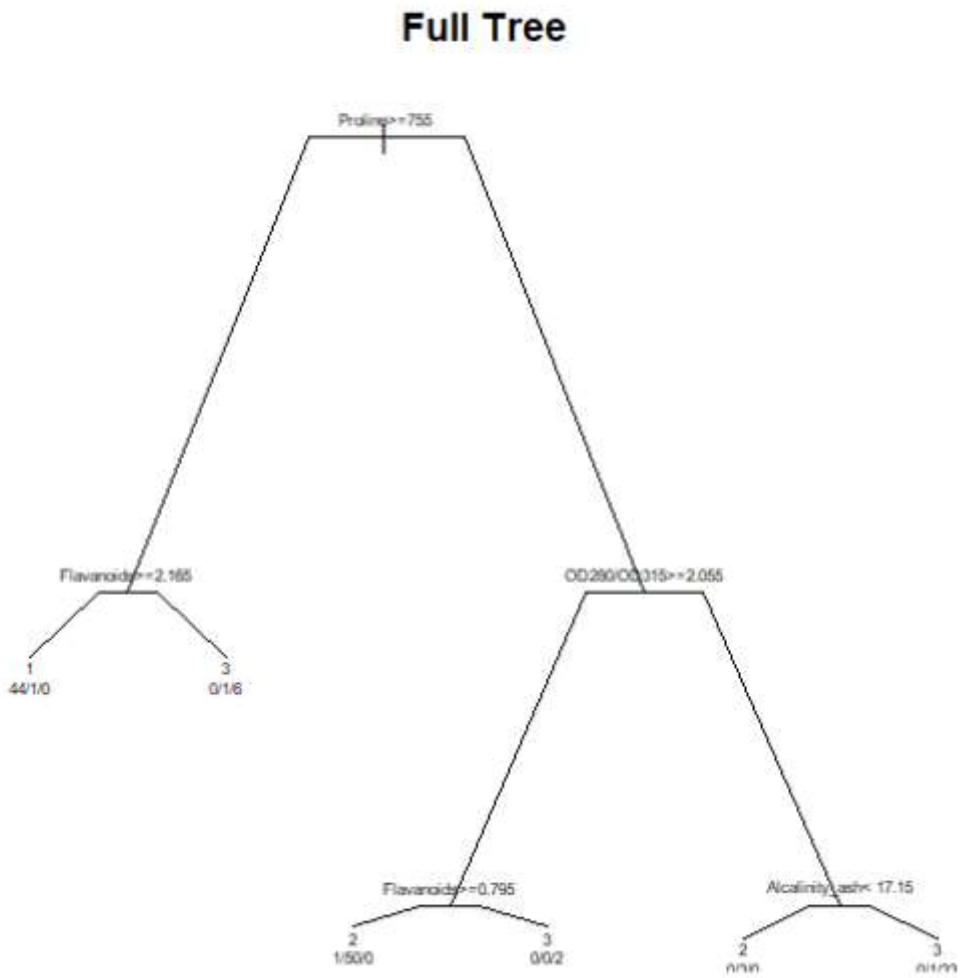


Figure 5: Full size classification tree for wine dataset.

From figure 5 we can see that the decision is made using five predictors' proline, flavonoids >=2.2, flavonoids <=0.8 , alkalinity ash and OD280/OD315, despite having 13 predictors.

We determined the testing and training to see where the training points fall. This was six for both the testing and training. This number of unique predictions correspond directly to the number of terminal nodes, as shown in the full sized tree in figure 5. To see how accurate the predictions were we determined the confusion matrix, as shown below.

For terminal node 1= label "44 / 1 / 0" tells us that there are 44 cases of category 1 (Barabera), 1 case of category 2 (Barolo), and 0 of category 3 (Grignolino).

For terminal node 2= label "0 / 1 / 6" tells us that there are 0 cases of category 1 (Barabera), only 1 case of category 2 (Barolo), and 6 of category 3 (Grignolino).

For terminal node 3= label "1 / 50 / 0" tells us that there are 1 cases of category 1 (Barabera), 50 cases of category 2 (Barolo), and 0 of category 3 (Grignolino).

For terminal node 4= label "0 / 0 / 2" tells us that there are 0 cases of category 1 (Barabera), 0 cases of category 2 (Barolo), and 2 of category 3 (Grignolino).

For terminal node 5= label "0 / 3 / 0" tells us that there are 0 cases of category 1 (Barabera), 3 cases of category 2 (Barolo), and 2 of category 3 (Grignolino).

For terminal node 6= label "0/1 / 33" tells us that there are 0 cases of category 1 (Barabera), 1 cases of category 2 (Barolo), and 33 of category 3 (Grignolino).

From the confusion matrix we can see that the accuracy of the predictions for the tree are about 93%, indicating a good fit.

```
Confusion Matrix and Statistics

          Reference
Prediction  1  2  3
         1  0  0  0
         2  1 12  0
         3  0  0  2

Overall Statistics

               Accuracy : 0.9333
                 95% CI : (0.6805, 0.9983)
    No Information Rate : 0.8
    P-Value [Acc > NIR] : 0.1671

                  Kappa : 0.7692

 Mcnemar's Test P-Value : NA
```

We determine the class counts at that node in the full sized tree, as shown below.

	#ofclass1countsatnode	#ofclass2countsatnode	#ofclass3countsatnode
3	44	1	0
5	1	50	0
69	0	3	0
71	0	1	6
132	0	1	33
143	0	0	2

We later grew a classification tree and pruned the tree which snips the tree based of the cp . To prune the tree we used the min_cp value. We can see from figure 6 that index 4 is the index of model complexity of the pruned tree as shown in figure 7.

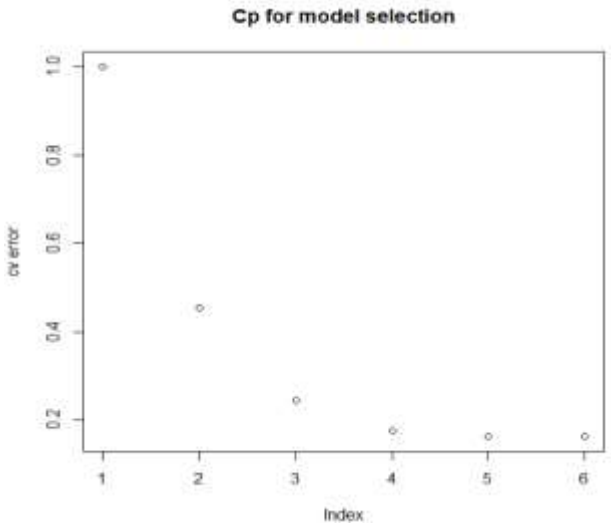
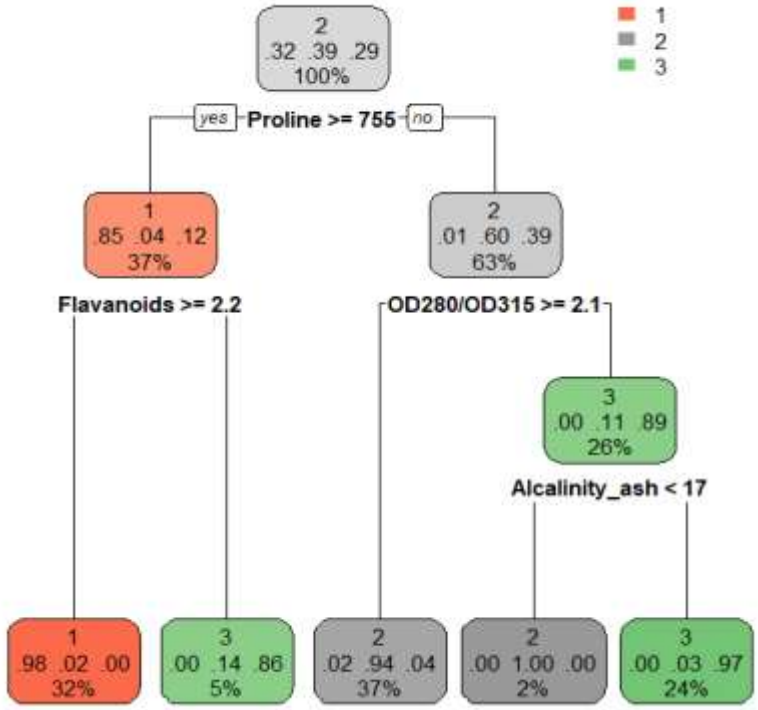


Figure 6: Cp for model selection to determine the mini cp value to prune tree



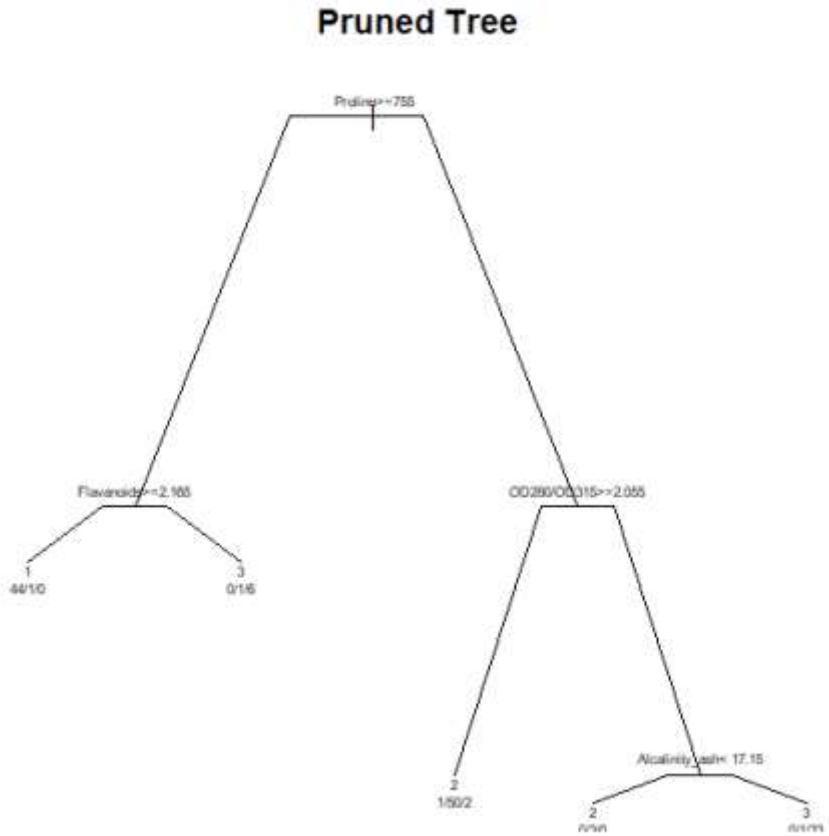


Figure 7: Pruned classification tree for wine dataset.

We determined the training and testing prediction and saw where the training and testing point s fall . The number of unique predictions was five which corresponds directly to the number of terminal nodes in the pruned tree as shown in figure 7. The number of testing samples which fa ll into each node is 4 6 14 5 15 and the training points was 6 15 4 5 14.

For terminal node 1= label "44 / 1 / 0" tells us that there are 44 cases of category 1 (Barabera), 1 case of category 2 (Barolo), and 0 of category 3 (Grignolino).

For terminal node 2= label "0 / 1 / 6" tells us that there are 0 cases of category 1 (Barabera), only 1 case of category 2 (Barolo), and 6 of category 3 (Grignolino).

For terminal node 3= label "1 / 50 / 2" tells us that there are 1 cases of category 1 (Barabera), 50 cases of category 2 (Barolo), and 2 of category 3 (Grignolino).

For terminal node 4= label "0 / 3 / 0" tells us that there are 0 cases of category 1 (Barabera), 3 cases of category 2 (Barolo), and 2 of category 3 (Grignolino).

For terminal node 5= label "0/1 / 33" tells us that there are 0 cases of category 1 (Barabera), 1 cases of category 2 (Barolo), and 33 of category 3 (Grignolino).

Using the tree package, we split the tree using gini, which does misclassification as 0/1, the number gini index is a measure of purity of the class, it minimizes the criterion and we have the terminal regions be pure.

The summary for the full sized tree is as follows:

```
Classification tree:
tree(formula = factor(Wine) ~ ., data = X.train, split = "gini")
Variables actually used in tree construction:
[1] "Magnesium"           "Nonflavanoid_phenols" "Malic_acid"
[4] "Flavanoids"          "Color_intensity"      "Total_phenols"
[7] "Alcohol"
Number of terminal nodes:  12
Residual mean deviance:  0.5132 = 66.71 / 130
Misclassification error rate: 0.09859 = 14 / 142
```

The summary for the pruned sized tree is as follows:

```
Classification tree:
snip.tree(tree = fit, nodes = c(509L, 126L, 255L))
Variables actually used in tree construction:
[1] "Magnesium"           "Nonflavanoid_phenols" "Flavanoids"
[4] "Color_intensity"
Number of terminal nodes:  9
Residual mean deviance:  0.8091 = 107.6 / 133
Misclassification error rate: 0.1479 = 21 / 142
```

From the full classification tree we can see there are 12 terminal nodes and the misclassification is minimal with 0.098 error rate. From the pruned classification tree we can

see there are 9 terminal nodes and the misclassification is minimal with 0.148 error rate. The full sized tree is grown as shown in figure 8 and the pruned tree, as shown in figure 9.

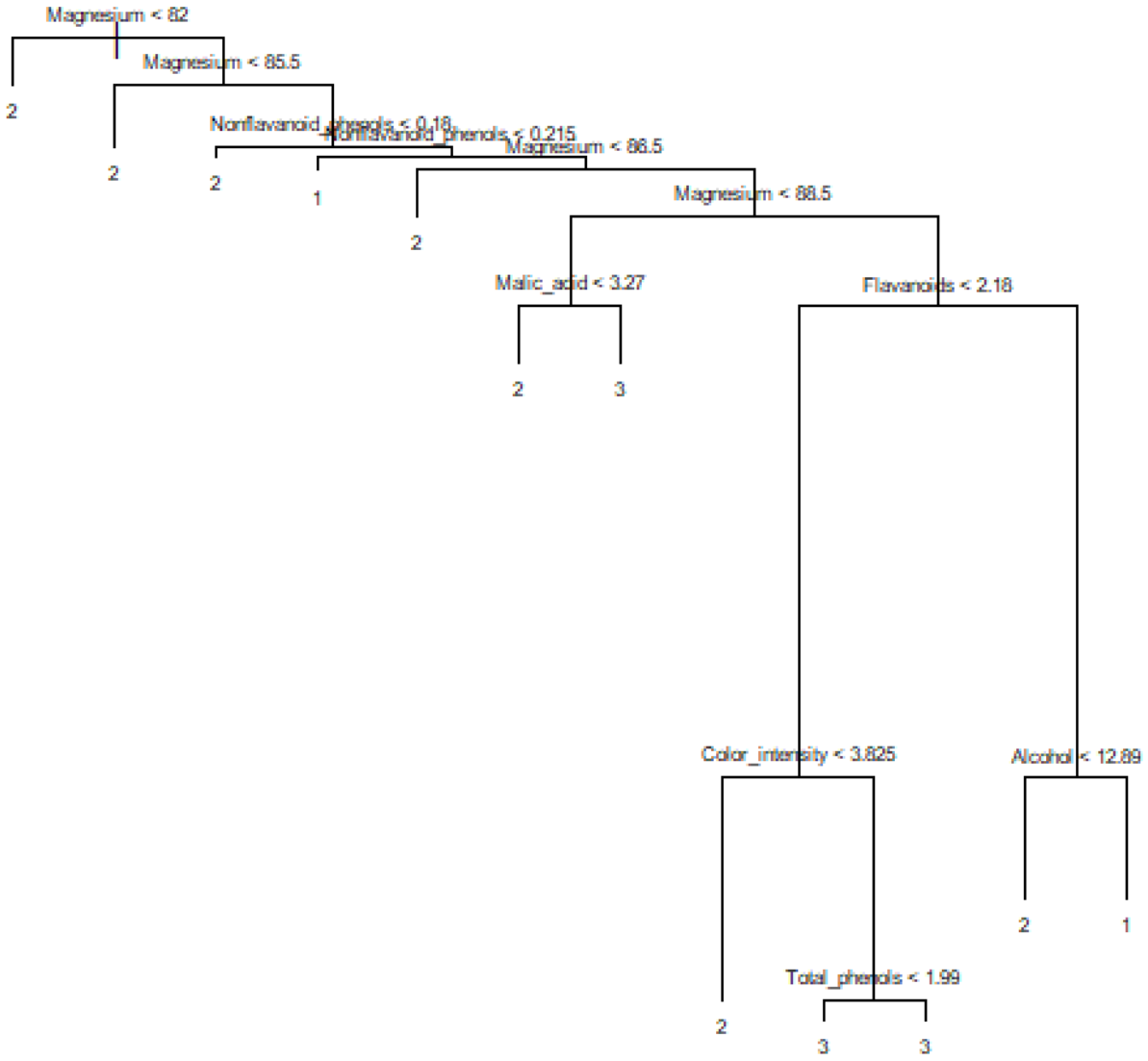


Figure 8: Full sized tree using the tree package

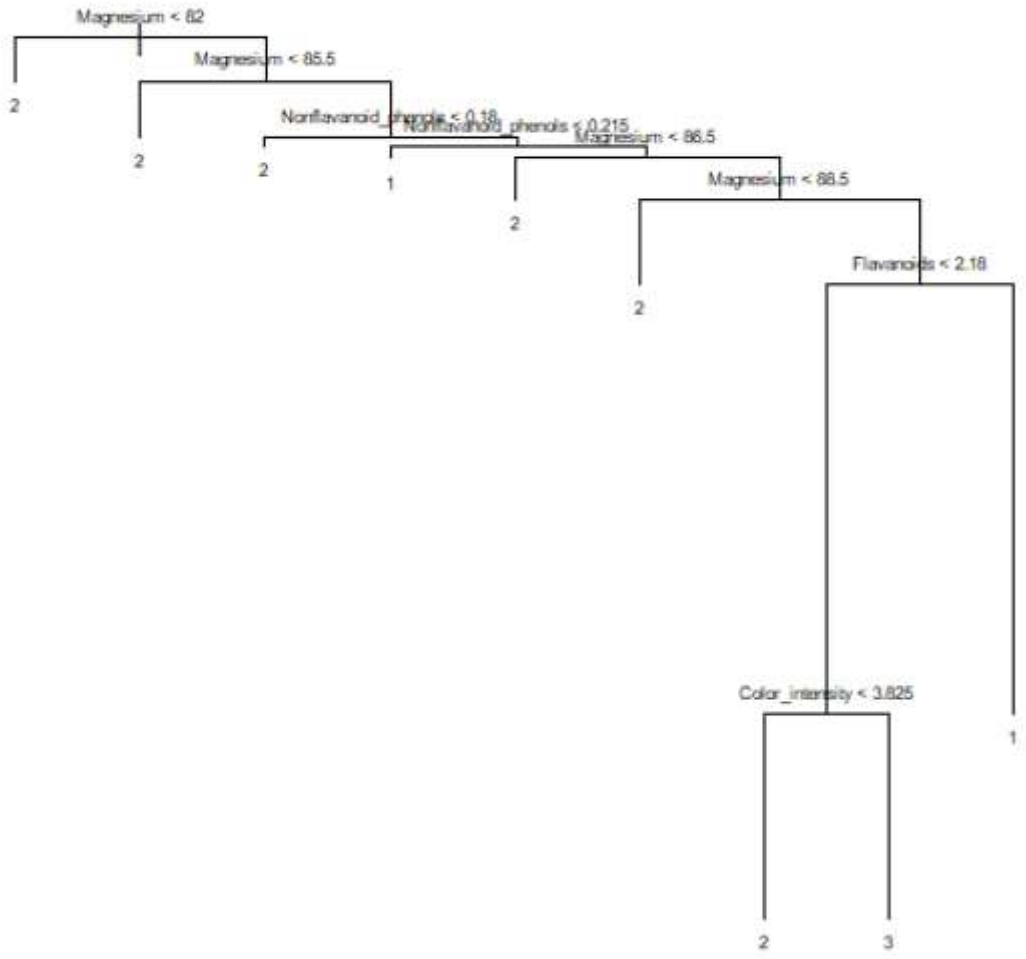


Figure 9: Pruned classification tree using tree package

From computing the confusion matrix we get an accuracy of 97% overall, as shown below.

```
Confusion Matrix and Statistics

          Reference
Prediction  1  2  3
         1 14  1  0
         2  0 14  0
         3  0  0  7

Overall Statistics

               Accuracy : 0.9722
                 95% CI : (0.8547, 0.9993)
    No Information Rate : 0.4167
    P-Value [Acc > NIR] : 1.055e-12

                  Kappa : 0.9565

 Mcnemar's Test P-Value : NA
```

Question 3

In this problem we had to apply bagging, boosting, and random forests to a data set of our own. I chose a heart disease data set (https://www.kaggle.com/ronitf/heart-disease-uci) and performed a classification on it. The data set had 303 rows and 14 columns. It was split into training set (2/3) and testing set (1/3). It is a binary classification dataset. It has 13 predictors with the response variable being the target (i.e. presence of heart disease=1 and no heart disease=0. We had to fit the models on a training set, and evaluate them on a test set.

Random forest

After applying random forest in order to assess how important each variable is on the tree we determined the node purity. The variable that is less important is on the bottom and the biggest indicator is on the top, as shown in figure 10. From figure 10 we can see that thal is the important predictor.

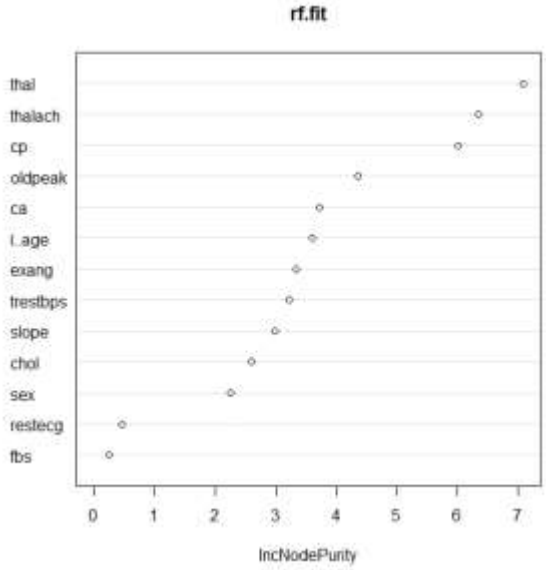


Figure 10: rf.fit using varImPlot showing the importance of each variable

The error/misclassification rate for the random forest was 0.2754411.

Bagging

After applying bagging in order to assess how important each variable is on the tree we determined the node purity. The variable that is less important is on the bottom and the biggest indicator is on the top, as shown in figure 11. From figure 11 we can see that thal is the important predictor.

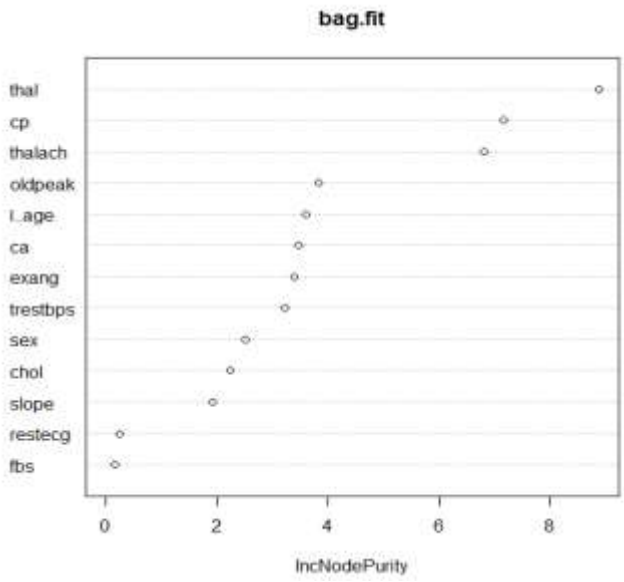


Figure 11: bag.fit using varImPlot showing the importance of each variable

The error/misclassification rate for bagging was 0.2821142.

Boosting

After fitting boosting, we outpiut the summary of the boost fit and we hows the relative importance of the variables. From figure 12 we can see that the thalach is more important followed by cp, which were one of the predictors in random forest and bagging.

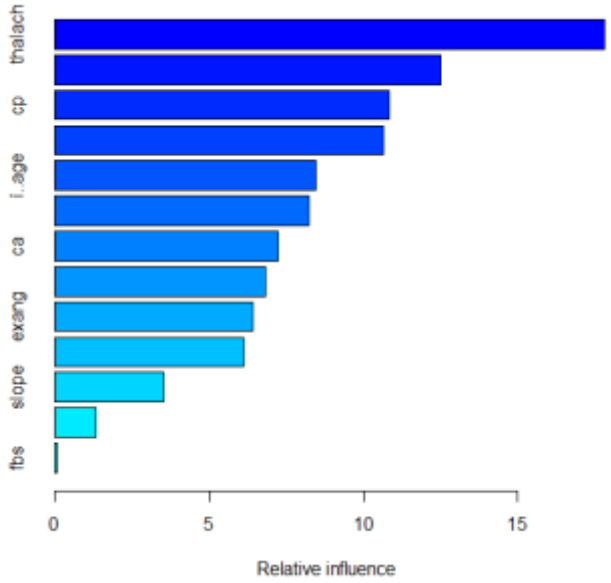


Figure 12: A graph of the boost.fit showing the variables of most importance

The error/misclassification rate for boosting with shrinkage of 0.1 was 0.2531233
The error/misclassification rate for boosting boosting with shrinkage of 0.6 was 0.209939

We can see that the greater the shrinkage the lower the error for boosting.

KNN

The KNN was performed for 20 different k value with the corresponding test errors as shown in figure 12.

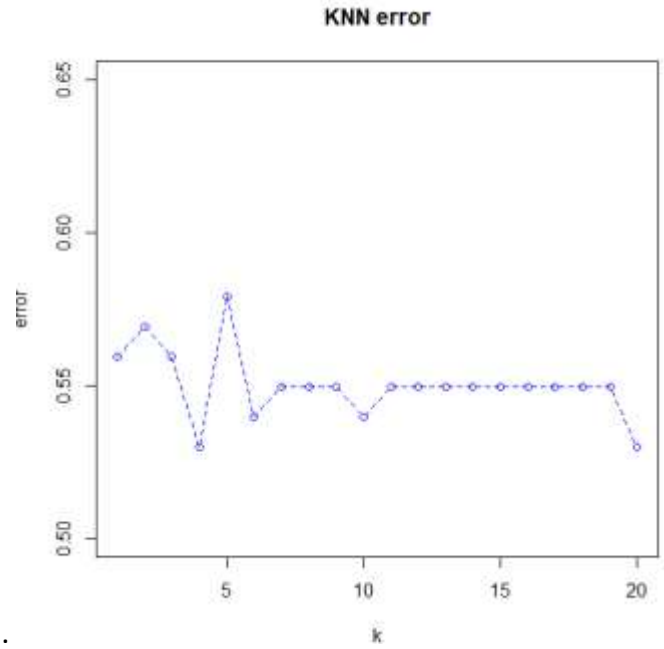


Figure 13: KNN test error for 20 KNN values.

Logisitc regression

Logistic regression was performed with an error of 0.3079208.

From the confusion matrix we can see that the accuracy of the logistic regression is not very go od with

79% accuracy.

```
Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0 32 10
         1 11 48

               Accuracy : 0.7921
                 95% CI : (0.6999, 0.8664)
    No Information Rate : 0.5743
    P-Value [Acc > NIR] : 3.405e-06

                  Kappa : 0.5735

 Mcnemar's Test P-Value : 1

            Sensitivity : 0.7442
            Specificity : 0.8276
         Pos Pred Value : 0.7619
         Neg Pred Value : 0.8136
             Prevalence : 0.4257
         Detection Rate : 0.3168
   Detection Prevalence : 0.4158
      Balanced Accuracy : 0.7859

       'Positive' Class : 0
```

From all the above test errors, we can conclude that the KNN and logistic regressions methods i.e. more simplistic (non-ensemble) methods are not as accurate, as compared to ensemble met hods such as the random forest, boosting, and bagging (ensemble methods).

This is because ensemble methods combine several models in order to produce one optimal pre dictive model. They don't just rely on one decision tree and make the right decision at each split. They make use of a sample of decision trees and calculate which features to use at each split. Th us the final predictor is based on the aggregated results of the sampled decision trees, which lea ds to better accuracy as compared to simple methods mentioned above.

What are some advantages (and disadvantages) do committee machines have related to the data set that you selected?

An advantage of the committee methods i.e.(bagging, boosting and random forests) is that they give improvements to weak classifiers, thus increasing the accuracy of the model. This is shown using the MSE from the binary classification dataset above. With respect to trees, they improve them by growing ensembles of trees, and letting them classify based on the majority vote. This is seen in the dataset used as they have better accuracy/ lower test error than the simple method. They are also good at reducing variance and providing higher stability.

However, some disadvantages for ensemble methods include higher computational time since some handle many large trees and large trees can be hard to interpret

Using the ensemble method, random forest provides accuracy which is often as good as AdaBoost and sometimes better, it is relatively more robust to outliers and is faster than bagging or boosting, as it performs tree growing through random selection of input variables.

Whereas bagging enhances accuracy when random features are used and can provide ongoing updates regarding the estimated generalization error of the ensemble of trees. However since it fits many large trees to bootstrap-resampled versions of the training data which leads to higher bias, but it can solve the over fitting problem

Boosting fits many large or small trees to reweighted versions of the training data and tries to reduce bias, however it can increase the over-fitting of the data, thus bagging is more effective often than boosting.

Question 4

In this problem we had to fit a series of random-forest classifiers to the SPAM data, to explore the sensitivity to m (the number of randomly selected inputs/predictors for each tree). We also had to plot both the test error and the OOB error against a suitably chosen range of values for m.

The random forest was fit using 100 decision trees. Random Forest uses the sqrt (p) variables when building a random forest of classification trees. Here we use a mtry=sqrt(57)=7.5= 8 . Thus I included the mtry equal to the square root of the total number of predictors when determining the test and OOB error and included the half and double of it is. 4 and 16.

The test error was plotted for values of mtry in the range of 1 to 50 to determine the appropriate range of m, as shown in figure 13. We can see that the test error does not change drastically after 20 mtry values, thus the test error for mtry values up to 20 was plotted as shown in figure 14. The OOB error was plotted against a different values for mtry (2,4,6,8).

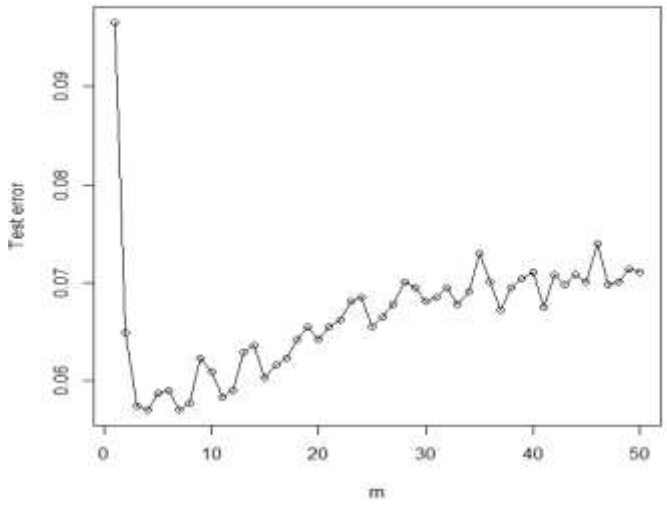


Figure 14: Test error for random forest for 50 mtry values (i.e. the number of randomly selected inputs for each tree)

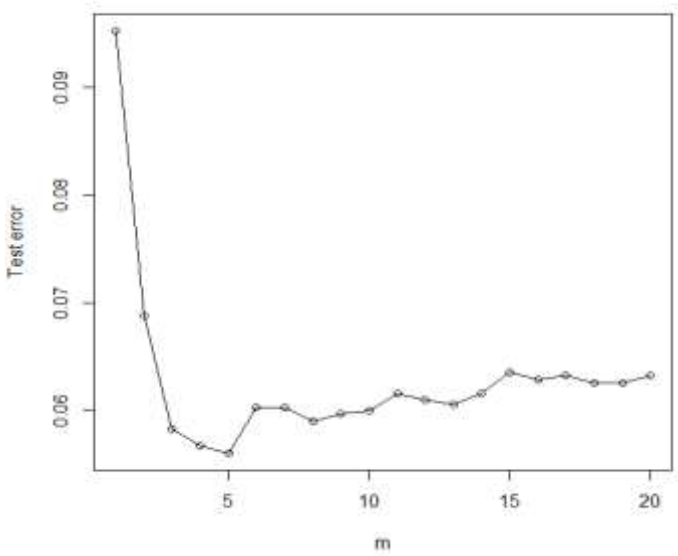


Figure 15: Test error for random forest for 20 mtry values (i.e. the number of randomly selected inputs for each tree)

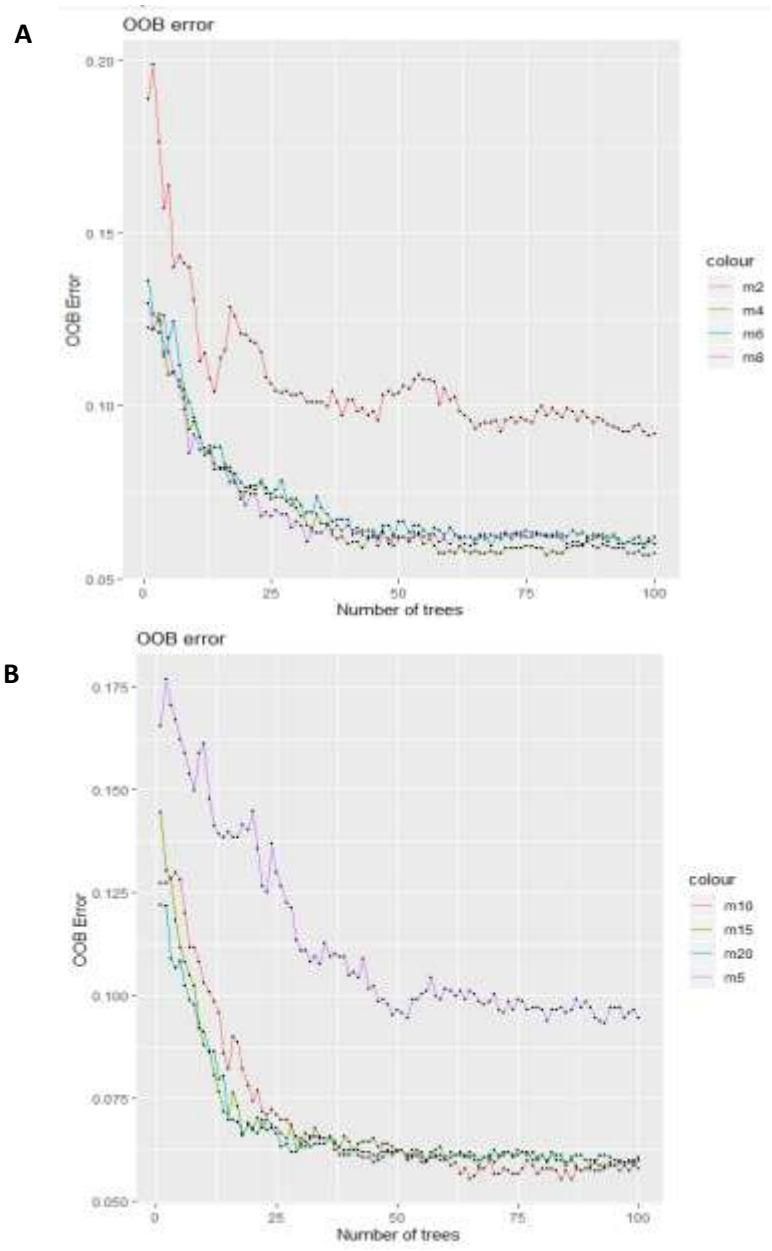


Figure 16: A and B show the OOB error for random forest for different range values of m

Out-of-bag (OOB) error is used to measure the prediction error of random forest. It is used to get a running unbiased estimate of the classification error as trees are added to the random forest, as shown in figure 16.

From figure 16 above we see that random forest does not over fit as more trees are added to the forest and the OOB error rate in random forest is not sensitive to the value of mtry over a very wide range such as from value m=10 to m=20 and from m=4 to m=8.

Also, from figure 16 we can see that the number of trees increase the OOB error decreases.

Question 5

What is "random" about a random forest?

A random forest is a classifier which consists of a collection of tree structured classifiers i.e. many decision trees, $\{h(x,\Theta_k), k=1,\ldots,M\}$ where M is the number of trees and $\{\Theta_k\}$ are independent identically distributed random vectors for the kth tree. The random vector $\{\Theta_k\}$ creates randomness for the random forest. In other words, the randomly selected inputs or combination of inputs to grow the tree at each node leads to the randomness in random forest. Each tree casts a unit vote for the most popular class at input x.

Therefore RF results from the combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. Due to the random feature selection, the trees are more independent of each other which often results in better predictive performance.

Furthermore, as the number of trees increases, for almost all sequences the test error converges to a minimum. Therefore, we are do not over fit when we add more and more trees. This randomness and the aggregation of the predictions made by multiple decision trees of varying depth improves the accuracy and makes RF more robust to outliers.