Lab 4: Wine Data set (Sam Farren)

1) Preliminary data analysis (15% of grade)

It was pretty difficult to make sense of the wine dataset at first glance with there being 12 attributes and having 1500 records. The first thing I did was load the dataset into R and compute the summary statistic which can be seen below:

```
      ID          fx_acidity     vol_acidity      citric_acid     resid_sugar
Min.   :   1.0   Min.   : 4.60   Min.   :0.1200   Min.   :0.000   Min.   : 0.900
1st Qu.: 400.5   1st Qu.: 7.10   1st Qu.:0.3900   1st Qu.:0.090   1st Qu.: 1.900
Median : 800.0   Median : 7.90   Median :0.5200   Median :0.260   Median : 2.200
Mean   : 800.0   Mean   : 8.32   Mean   :0.5278   Mean   :0.271   Mean   : 2.539
3rd Qu.:1199.5   3rd Qu.: 9.20   3rd Qu.:0.6400   3rd Qu.:0.420   3rd Qu.: 2.600
Max.   :1599.0   Max.   :15.90   Max.   :1.5800   Max.   :1.000   Max.   :15.500
  chlorides        free_sulf_d     tot_sulf_d       density             pH
Min.   :0.01200   Min.   : 1.00   Min.   :  6.00   Min.   :0.9901   Min.   :2.740
1st Qu.:0.07000   1st Qu.: 7.00   1st Qu.: 22.00   1st Qu.:0.9956   1st Qu.:3.210
Median :0.07900   Median :14.00   Median : 38.00   Median :0.9968   Median :3.310
Mean   :0.08747   Mean   :15.87   Mean   : 46.47   Mean   :0.9967   Mean   :3.311
3rd Qu.:0.09000   3rd Qu.:21.00   3rd Qu.: 62.00   3rd Qu.:0.9978   3rd Qu.:3.400
Max.   :0.61100   Max.   :72.00   Max.   :289.00   Max.   :1.0037   Max.   :4.010
    sulph           alcohol         quality       class
Min.   :0.3300   Min.   : 8.40   Min.   :3.000   High:855
1st Qu.:0.5500   1st Qu.: 9.50   1st Qu.:5.000   Low :744
Median :0.6200   Median :10.20   Median :6.000
Mean   :0.6581   Mean   :10.42   Mean   :5.636
3rd Qu.:0.7300   3rd Qu.:11.10   3rd Qu.:6.000
Max.   :2.0000   Max.   :14.90   Max.   :8.000
```

Out of all of the attributes mentioned above only two of them looked problematic. These were the free_sulf_d and tot_sulf_d attributes. Free_sulf_d had a minimum of 1 and a maximum of 72 with a median of 14. This meant that 72 was an outlier by a lot and possibly other values too were outliers in this column. The same went for tot_sulf_d as the minimum was 6 and the maximum was 289 with a median of 38. Between these two there were a decent amount of outliers which could skew the data. To decide what to do I obtained a table of the most significant correlations between attributes which can be seen in the table below.

The table on the left shows the highest attributes that correlate with each other. The table on the right shows how each attribute correlated with alcohol and quality specifically since quality was the basis of choosing whether the class of a data entry was high or low. From that, alcohol had the highest correlation with quality at .476 so it was determined this held the most importance with the classification. .476 is by no means significant in terms of correlation however it was the highest among all of the attributes.

	X1	X2	variable	value
4	pH	fx_acidity	value	-0.6829782
14	fx_acidity	pH	value	-0.6829782
2	citric_acid	fx_acidity	value	0.6717034
6	fx_acidity	citric_acid	value	0.6717034
3	density	fx_acidity	value	0.6680473
13	fx_acidity	density	value	0.6680473
10	tot_sulf_d	free_sulf_d	value	0.6676665
11	free_sulf_d	tot_sulf_d	value	0.6676665
5	citric_acid	vol_acidity	value	-0.5524957
7	vol_acidity	citric_acid	value	-0.5524957
9	pH	citric_acid	value	-0.5419041
15	citric_acid	pH	value	-0.5419041

	alcohol	quality
fx_acidity	-0.06166827	0.12405165
vol_acidity	-0.20228803	-0.39055778
citric_acid	0.10990325	0.22637251
resid_sugar	0.04207544	0.01373164
chlorides	-0.22114054	-0.12890656
free_sulf_d	-0.06940835	-0.05065606
tot_sulf_d	-0.20565394	-0.18510029
density	-0.49617977	-0.17491923
pH	0.20563251	-0.05773139
sulph	0.09359475	0.25139708
[illegible]	[illegible]	[illegible]
quality	0.47616632	1.00000000

2) Data Transformations (5% of grade)

The following decisions were based primarily on the preliminary data analysis. For outliers and missing data, there were no missing values as stated by the data set on the website so this didn't need to be handled, but outliers were handled by a feature subset selection from within the data. The following analysis and decisions were made for each attribute.

"ID" -> Doesn't need to be in models as it doesn't hold any predictive power
"fx_acidity" & "vol_acidity"-> these were taken out since pH accounts for much of the information contained in these two attributes
"citric_acid" -> Important to the calculation of quality so kept
"resid_sugar" -> Important to the calculation of quality so kept
"chlorides" -> Important to the calculation of quality so kept
"free_sulf_d" -> This was removed and tot_sulf_d was kept to hold information for the both of these attributes
"tot_sulf_d" -> Kept to represent free_sulf_d attribute information
"density" -> Important to the calculation of quality so kept
"pH" -> Important to the calculation of quality(correlated with fx_acidity and vol_acidity so kept)
"sulph" -> Important to the calculation of quality
"alcohol" -> Very important attribute
"quality" -> Class is taken directly from the calculation of this attribute and therefore shouldn't be used as you would be able to tell the class solely from this value
"class" -> This is the predicted(output) variable in the models

The final feature subset used on the models were:

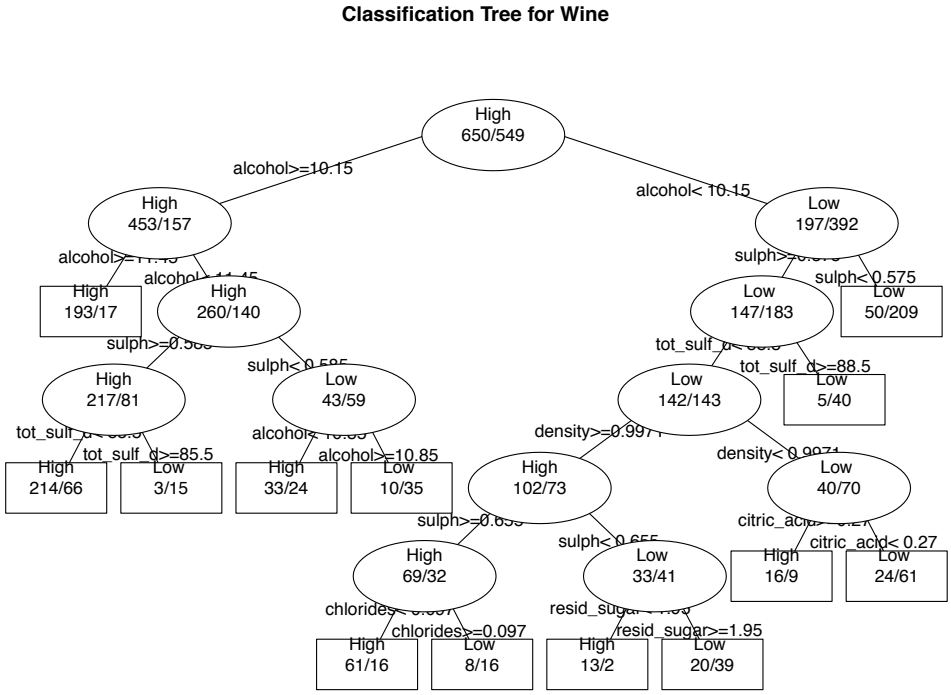
(Citric Acid, Residual Sugar, Chlorides, Total Sulf_d, Density, pH, Sulph, and Alcohol)

3) Model development (50% of grade)

For all cases of these models, the training set consisted of a sample of 75% of the wine data that was used to train and build each model. Then the remaining 25% of the data was ran through each model to test and classify each data entry.

1) **A. Decision Tree**

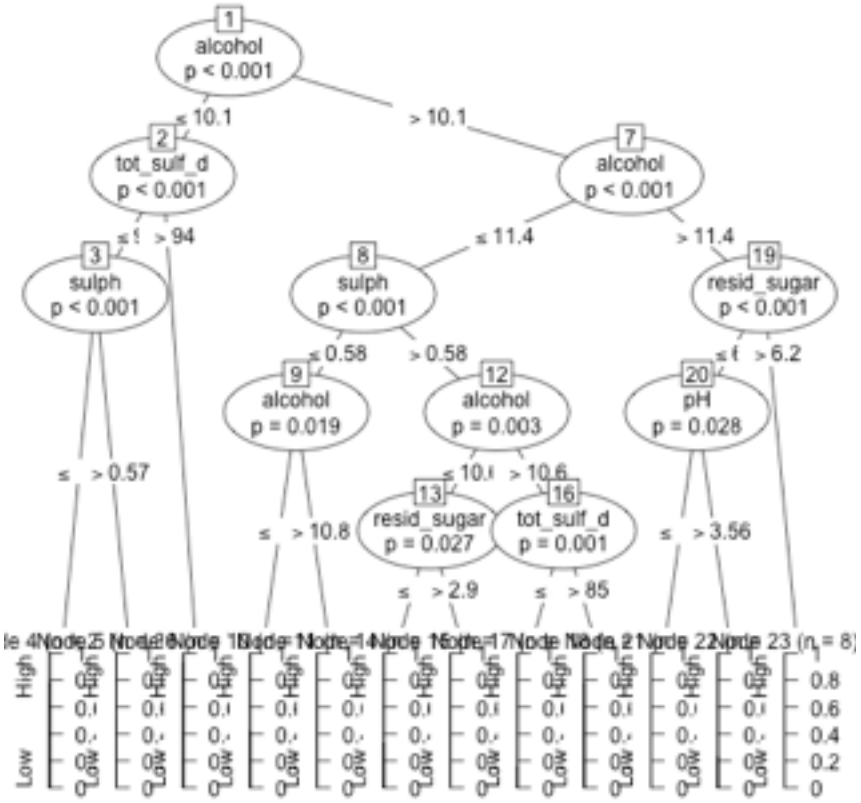
Initially I did the decision tree with all of the data, except ID, quality, and class, however the tree was huge and the splits didn't seem to perform well, so I went ahead and used the feature subset as discussed above. The following tree was the model created from the training data set, and when the test set was ran through, the below confusion matrix was produced. The tree was built from a package in R called "Party" which uses the "rpart" function to determine the best splits. This differs from the Conditional Inference Tree because the splits are based off of Gini coefficients and not significance tests(explained more in Conditional Inference Tree Section). The confusion matrix was computed for the test data and can be seen below.



pred	High	Low
High	153	55
Low	52	140

2) **B. A rules-based classifier (Conditional Inference Tree)**

The rule based classifier or conditional inference tree was implemented with the “party” library in R. It is still a tree based solution however it differs slightly from the decision tree because it uses a function called ctree which doesn’t perform splits based off of the Gini values. It uses a significance test based off of permutated rearrangements of the splits generated at the beginning of the algorithm to compute the best splits instead. Error was decently high with this coming in at around 30%.



	Predicted class	
Actual class	High	Low
High	128	77
Low	40	155

3) C. Naive Bayes

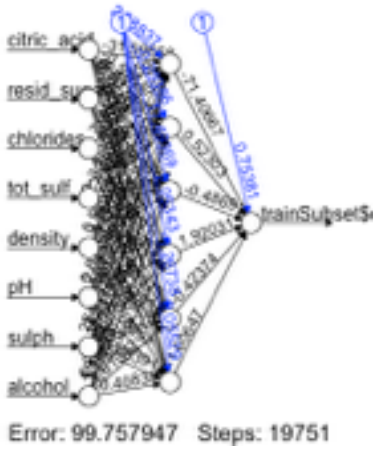
The Naive Bayes classifier was implemented in R with a package called “e1071.” This was surprisingly the most simple to implement with the predefined functions. I just called it on my feature subset and tested on the test data and no errors were produced. The accuracy ended up being around 75% which is about what the other models were. This one definitely scored more points for the ease of use section. This model had very little arguments to play around

with. The only one to change the way in which it was implemented was changing the order attribute, however it was kept at 1 so that the order was based solely on the mean of the data. The following confusion matrix was obtained from the naive bayes model.

	Predicted class	
Actual class	High	Low
High	149	43
Low	56	152

4) D. Artificial Neural Network

The following graph was obtained for the neural network implementation of the wine dataset. The perceptrons can be seen on the left side being: citric acid, residual sugar, chlorides, total sulfur, density, pH, sulph, and alcohol (the feature subset section from above). These inputs were decided to be the best and most optimal for classifying the wine as high or low. I played around with the threshold parameter to see how the neural net would respond. It was difficult because a very low threshold wouldn't allow the ANN to converge, so it would time out. However, I wanted a relatively low error rate for an optimal model, so I didn't want to make the threshold too high. after trying .01, .05, .1, and .2, .1 seemed to be the most reasonable choice as it ran in an efficient amount of time and produced a decent model. After the model was created with the training data however, I could not for the life of me run my test data through and extract the statistics from the model.



5) E. Support Vector Machine

The support vector machine was also implemented in R using the "e1071" library. This had a little better results than the rest of the models giving an accuracy of around 78% as compared to 75%. The parameters used were a C-classification with a Gaussian radial kernel, cost of 1, and

a gamma of .125. The package uses a gamma value that is equal to 1/dimension of data so since the feature subset was used, the gamma was 1/8.

```
Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1
      gamma:  0.125
```

```
                Predicted class
Actual class High Low
        High  147  29
        Low    58 166
```

6) F. Ensemble learner: Bagging

For the Ensemble learner, I decided to go with a bagging implementation. This was done using the “adabag” library in R. This seemed to be a really good library, so I chose this instead of trying to implement a Random Forest model. The main parameter I played around with was “mfinal” which determined how many iterations boosting would occur. I found that 100 iterations produced the smallest error rate at 22% however, it was inefficient for run time. 1 iteration produced an error rate of 28%, and 10 iterations of boosting produced the below confusion matrix with an error of 25.75%. I felt the 10 iterations was efficient and gave a decent error free model so I chose this as optimal.

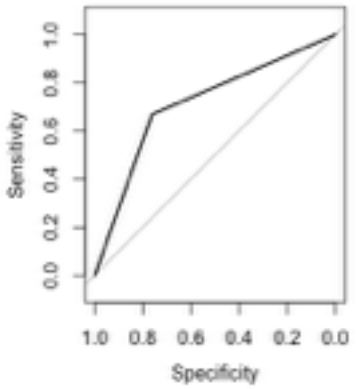
```
                Observed Class
Predicted Class High Low
           High  158  56
           Low    47 139
> test.bagging.pred$error
[1] 0.2575
```

4) **Model evaluation (30% of grade)** -

To test the performance and implementation of each of the models described above, different statistics were taken into account such as Accuracy, F-measure, and the Roc curve for each prediction model. Other things taken into account to decide which model was the best was also how easy and customizable each of the models were. The following statistics were obtained for each of the six different models:

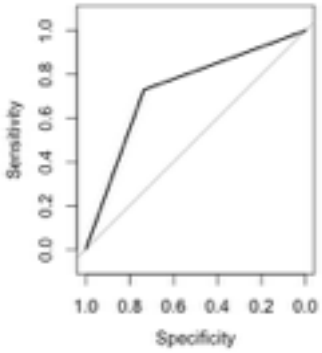
Decision Tree:

Accuracy: 73.25%
F-measure: .7409
Roc-Curve:
Area Under Curve: .715



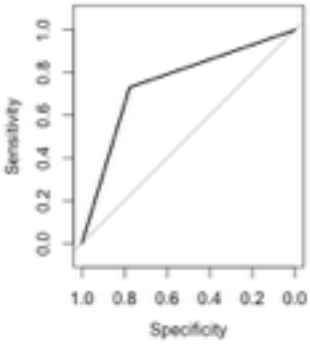
Conditional Inference Tree:

Accuracy: 70.75%
F-measure: .6863
Roc-Curve:
Area Under Curve: .7324



Naive Bayes:

Accuracy: 75.25%
F-measure:.7506
Roc-Curve:
Area Under Curve: .7534



Artificial Neural Network:

Accuracy: TBD
F-measure:
Roc-Curve:
Area Under Curve:

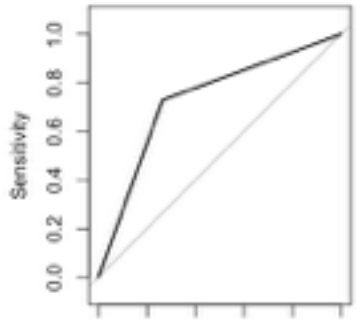
Support Vector Machine:

Accuracy: 78.25%

F-measure: .7716

Roc-Curve:

Area Under Curve: .7324



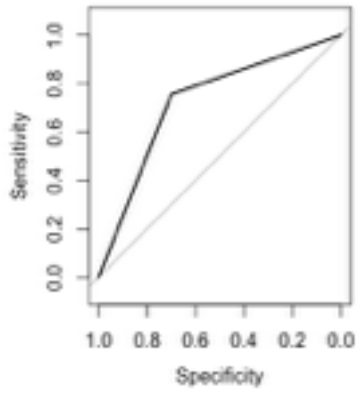
Ensemble Learner(Bagging and Boosting):

Accuracy: 74.25%

F-measure: .7542

Roc-Curve:

Area Under Curve: .7276



Overall most of the models ended up having pretty similar numbers in terms of Accuracy and f-measure. The support vector machine ended up having the highest f-measure and accuracy, along with the second highest AUC for it's ROC curve. In a close second to the SVM, my ensemble learner and Naive Bayes had a close accuracy, f-measure and AUC. I ran into quite a few problems while trying to implement the ANN due to the nature of its complexity. I found it to be highly customizable, which can be a disadvantage or an advantage depending on how you look at it. The ANN run time was decently slow, especially if you placed a low threshold on it. There were about 15 other parameters that you could also use, and if given more time, I think that you could come up with a very intricate and precise model to predict the class of the wine. However, in terms of usability, the learning curve was pretty high and caused me a lot of issues. I could not figure out for the life of me how to correctly run my test data through the neural net package model and extract the statistics from it. On the complete opposite end was the Naive Bayes implementation. This was very easy to implement and only had three parameters . The results were good since they were near the top of the other models, however I don't see much improvement in future runs due to its' low model customizability. The only thing that really could have changed the performance for the Naive Bayes was if a better feature subset was selected. The conditional Inference and Decision tree were very similar to each other. Both chose splits on Alcohol at the root since this was deemed the most important attribute that held the most value, but I don't think many of the other attributes held enough value to create an above average model. For this dataset however, I don't think a tree based solution was best.

Overall, I found the SVM to be the best implementation and not just because it had the highest accuracy, but it ran efficiently(provided a reasonable number of iterations), created a better

model than all of the others, and was very easy to work with. It also had a decent amount of customization to its parameters that allowed for the creation of a better model. This included pruning over multiple iterations or boosting over multiple iterations as well. Some of the cons however are that it can be hard to select the optimal kernel to use. I just had to play around with it to see which ones worked the best. For this instance it happened to work best with a radial kernel. Another con is that with larger datasets, I can see the SVM performing slow especially if you have to increase the iterations for pruning or boosting. However, for the wine dataset, it was my favorite and it had high predictive power.