Group 9

Elijah M Raffo

Jiazhou Li

Xunyi Jiang

Wine Quality Predictive

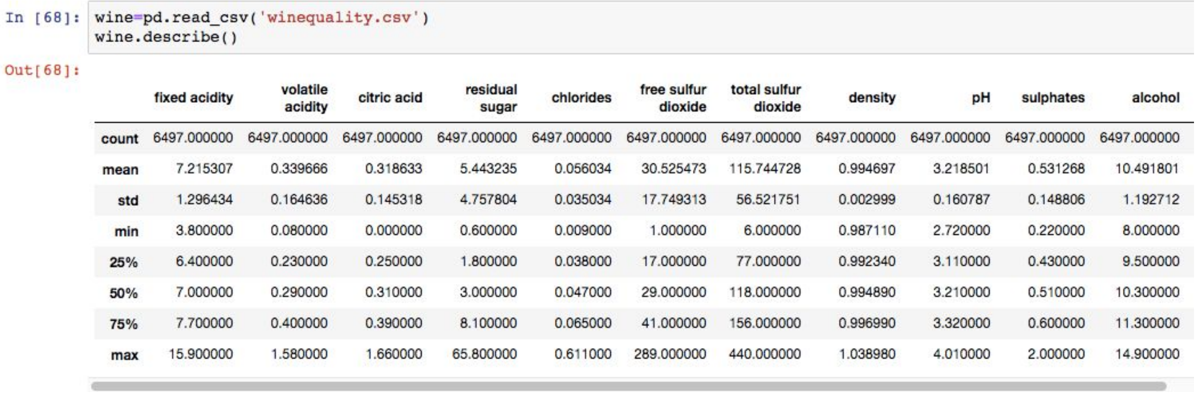
Introduction

The primary data that we used comes from Analytic Vidhya. It provides two datasets which include Red and White wine. The dataset has over 5000 individual data points which we combined into one spreadsheet. We hoped to find more data sets, and combine them into our primary data set; however, as the project continued finding data with similar columns proved to difficult. We changed the quality score into a class score to create a classification model. Our goal is to create a robust model for classifying wines whose variables are linked to real world methods of testing batches of wine. By combining data, we hope to create a large enough dataset to create accurate test and training datasets.

Descriptive Analysics

For the descriptive analytics, we have 11 input variables for both white wine and red wine based on the chemical signature tests. They are fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates and alcohol. Our target variable is quality which score is between 0 to 10, and the higher the score the better the wine taste.

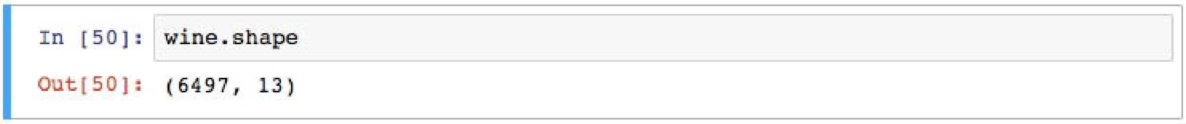
In order to clean the data, we first tested to see if there are any missing data rows in our datasets. We used the describe formula "wine.describe()" to find out the total counts for both red and white wine which when ran displayed 6497 total rows.



```
In [68]: wine=pd.read_csv('winequality.csv')
         wine.describe()

Out[68]:
```

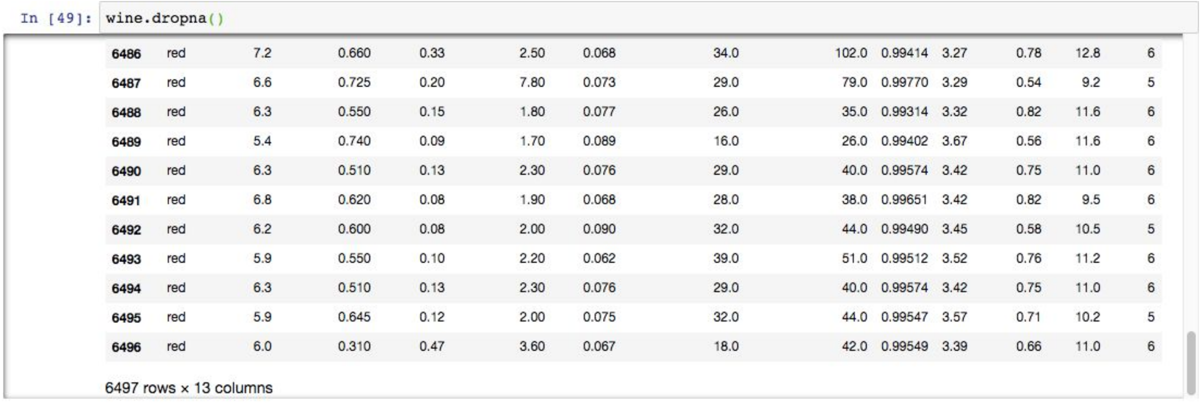
	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol
count	6497.000000	6497.000000	6497.000000	6497.000000	6497.000000	6497.000000	6497.000000	6497.000000	6497.000000	6497.000000	6497.000000
mean	7.215307	0.339666	0.318633	5.443235	0.056034	30.525473	115.744728	0.994697	3.218501	0.531268	10.491801
std	1.296434	0.164636	0.145318	4.757804	0.035034	17.749313	56.521751	0.002999	0.160787	0.148806	1.192712
min	3.800000	0.080000	0.000000	0.600000	0.009000	1.000000	6.000000	0.987110	2.720000	0.220000	8.000000
25%	6.400000	0.230000	0.250000	1.800000	0.038000	17.000000	77.000000	0.992340	3.110000	0.430000	9.500000
50%	7.000000	0.290000	0.310000	3.000000	0.047000	29.000000	118.000000	0.994890	3.210000	0.510000	10.300000
75%	7.700000	0.400000	0.390000	8.100000	0.065000	41.000000	156.000000	0.996990	3.320000	0.600000	11.300000
max	15.900000	1.580000	1.660000	65.800000	0.611000	289.000000	440.000000	1.038980	4.010000	2.000000	14.900000

Moreover, we also checked the shape of the datasets which displayed the raw data's shape to be 13 columns and 6497 rows. The purpose of testing the shape is to find out the total numbers of data and variables in the datasets before cleaning data.



```
In [50]: wine.shape

Out[50]: (6497, 13)
```

Then, we cleaned the data by using the formula "wine.dropna()". The value of rows and columns in the dataset remained the same after we finished cleaning the data. Therefore, we believe that there was and is no missing data in our datasets.

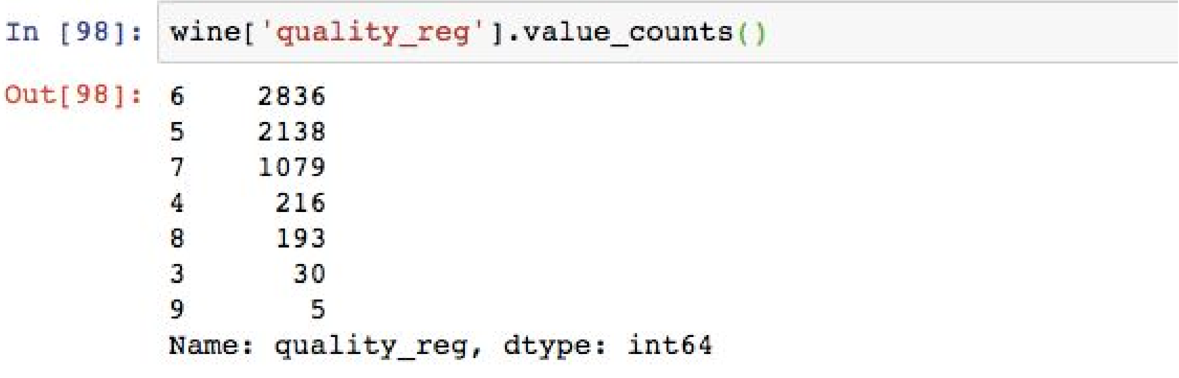


```
In [49]: wine.dropna()
```

6486	red	7.2	0.660	0.33	2.50	0.068	34.0	102.0	0.99414	3.27	0.78	12.8	6
6487	red	6.6	0.725	0.20	7.80	0.073	29.0	79.0	0.99770	3.29	0.54	9.2	5
6488	red	6.3	0.550	0.15	1.80	0.077	26.0	35.0	0.99314	3.32	0.82	11.6	6
6489	red	5.4	0.740	0.09	1.70	0.089	16.0	26.0	0.99402	3.67	0.56	11.6	6
6490	red	6.3	0.510	0.13	2.30	0.076	29.0	40.0	0.99574	3.42	0.75	11.0	6
6491	red	6.8	0.620	0.08	1.90	0.068	28.0	38.0	0.99651	3.42	0.82	9.5	6
6492	red	6.2	0.600	0.08	2.00	0.090	32.0	44.0	0.99490	3.45	0.58	10.5	5
6493	red	5.9	0.550	0.10	2.20	0.062	39.0	51.0	0.99512	3.52	0.76	11.2	6
6494	red	6.3	0.510	0.13	2.30	0.076	29.0	40.0	0.99574	3.42	0.75	11.0	6
6495	red	5.9	0.645	0.12	2.00	0.075	32.0	44.0	0.99547	3.57	0.71	10.2	5
6496	red	6.0	0.310	0.47	3.60	0.067	18.0	42.0	0.99549	3.39	0.66	11.0	6

6497 rows × 13 columns

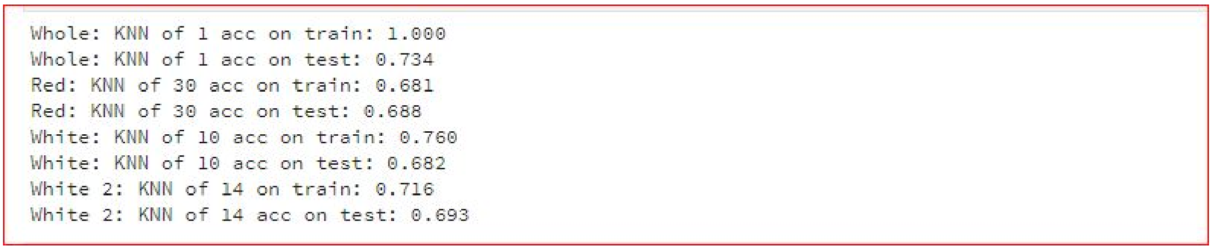
To create an accurate classification data set we wanted to divide the data into two groups: good and bad. To understand the quality scores of the wine, a score between 0-10, we used the formula "wine['quality_reg'].value_counts()". In addition, we notice that most of the wine had a score of 6 or 5. This was good because our data could be divided into "good", and "bad" groups by separating them by on 6. Therefore, 6 and above were good, and everything under 6 was bad.



```
In [98]: wine['quality_reg'].value_counts()

Out[98]: 6    2836
         5    2138
         7    1079
         4     216
         8     193
         3      30
         9       5
         Name: quality_reg, dtype: int64
```

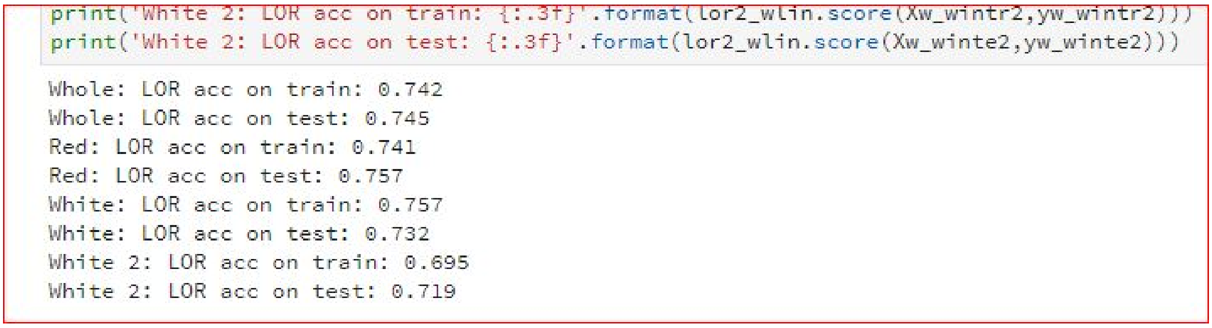
Predictive Analytics

We know that our dataset is a classification model because we are classifying wines by taste quality. It would be a regression dataset if we were predicting a price for the wine; however, we are not. We decided to not create a classification model for each quality score due to the high amount of quality 5, and 6 in our data set. The first phase of our project was built around building models for our combined dataset of red and white wines. The second phase of our model was to run these same models with seperated red, and white datasets. We did this because our data was comprised of an unbalanced mix of red and white data. In other words, there was much more white wine data than redwine data. While that did not affect accuracy it did affect how authentic our finding would be. The final phase of our project was to take all good quality white wine, and create another classification model. The hopes of this model was to see if we could create a model that turns good wines [wines with a quality score of 6 and above] into a great wine [wines with a quality score of 7 and above].



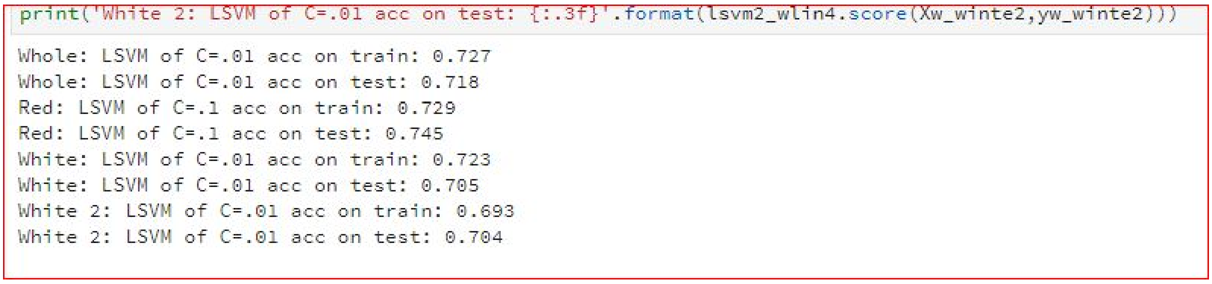
```
Whole: KNN of 1 acc on train: 1.000
Whole: KNN of 1 acc on test: 0.734
Red: KNN of 30 acc on train: 0.681
Red: KNN of 30 acc on test: 0.688
White: KNN of 10 acc on train: 0.760
White: KNN of 10 acc on test: 0.682
White 2: KNN of 14 on train: 0.716
White 2: KNN of 14 acc on test: 0.693
```

Our first model we ran was the K-Nearest Neighbors model. The whole data set preformed the best at 1 nearest neighbor. While the red, and white sets performed better with more neighbors. The whole dataset could be oversaturated with scores of 5, and 6; therefore, an overfit model would work will for our test dataset. While for the individual red, and white datasets a higher neighbor count resulted in an increased accuracy. Red KNN performed best at 30 neighbors, and White KNN performed best at 10 neighbors. Finally, when the white dataset was further divided into “great” {Phase 3} wine it had a high accuracy using the KNN model. This could be because wine with similar chemical results “feel” the same to consumers; however, other models yielded better accuracy.



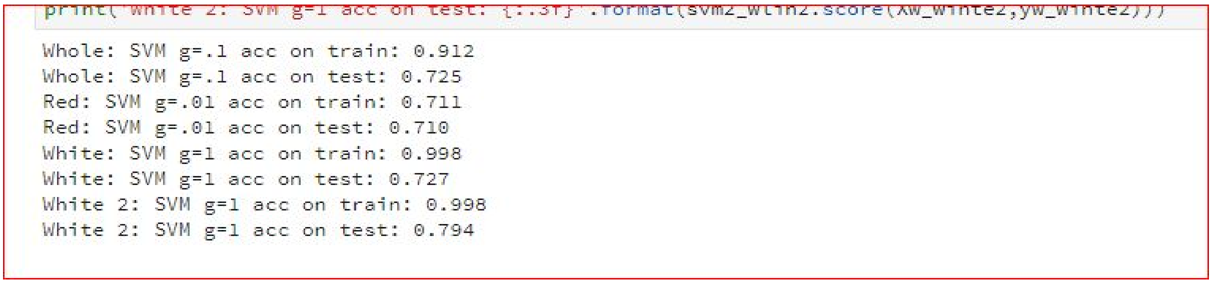
```
print('White 2: LOR acc on train: {:.3f}'.format(lor2_wlin.score(Xw_wintr2,yw_wintr2)))
print('White 2: LOR acc on test: {:.3f}'.format(lor2_wlin.score(Xw_winte2,yw_winte2)))

Whole: LOR acc on train: 0.742
Whole: LOR acc on test: 0.745
Red: LOR acc on train: 0.741
Red: LOR acc on test: 0.757
White: LOR acc on train: 0.757
White: LOR acc on test: 0.732
White 2: LOR acc on train: 0.695
White 2: LOR acc on test: 0.719
```

Our second model which on all tests displayed one of our higher accuracies was Logistic Regression. The highest accuracy dataset was the red wine dataset. And all datasets on training and test data had an accuracy of 70% or higher. This could be because the chemicals of a wine require a formulaic balance. Instead of people identifying similar tasting wines, and assigning a quality. People require a balance of flavors, and chemicals in a wine. It would be overconfident to say a formula could be what makes wine delicious; however, we can say that a formula predicts wine quality more accurately than KNN.



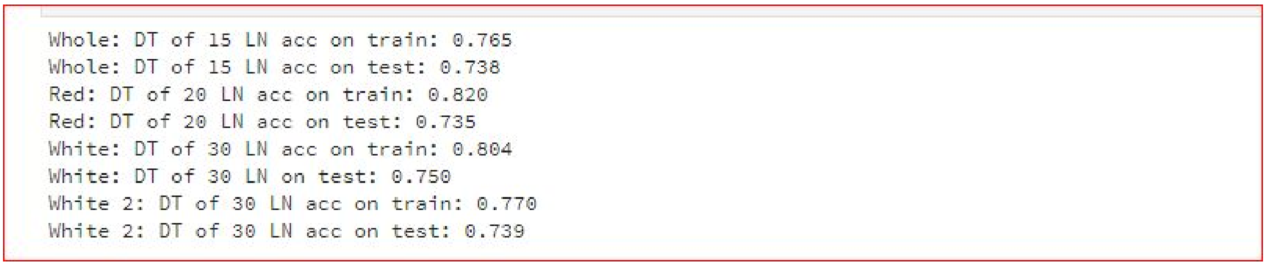
```
print('White 2: LSVM of C=.01 acc on test: {:.3f}'.format(lsvm2_wlin4.score(Xw_winte2,yw_winte2)))

Whole: LSVM of C=.01 acc on train: 0.727
Whole: LSVM of C=.01 acc on test: 0.718
Red: LSVM of C=.1 acc on train: 0.729
Red: LSVM of C=.1 acc on test: 0.745
White: LSVM of C=.01 acc on train: 0.723
White: LSVM of C=.01 acc on test: 0.705
White 2: LSVM of C=.01 acc on train: 0.693
White 2: LSVM of C=.01 acc on test: 0.704
```

When we analyzed our four datasets [whole, red, white, white 2] with the linear support vector machine we achieved high accuracy. Interestingly all models except red preformed best with a C=.01. However, red in an LSVM model had the highest accuracy. This could be an error because red train performed worse than red test. In other words, the random state assigned when splitting the red wine dataset accidently favors an LSVM model with C=.1. Or red wine can be linearly split into good and bad wine unlike white wine. A smaller C means that some misclassification is accepted in the LSVM model which most datasets did well with.



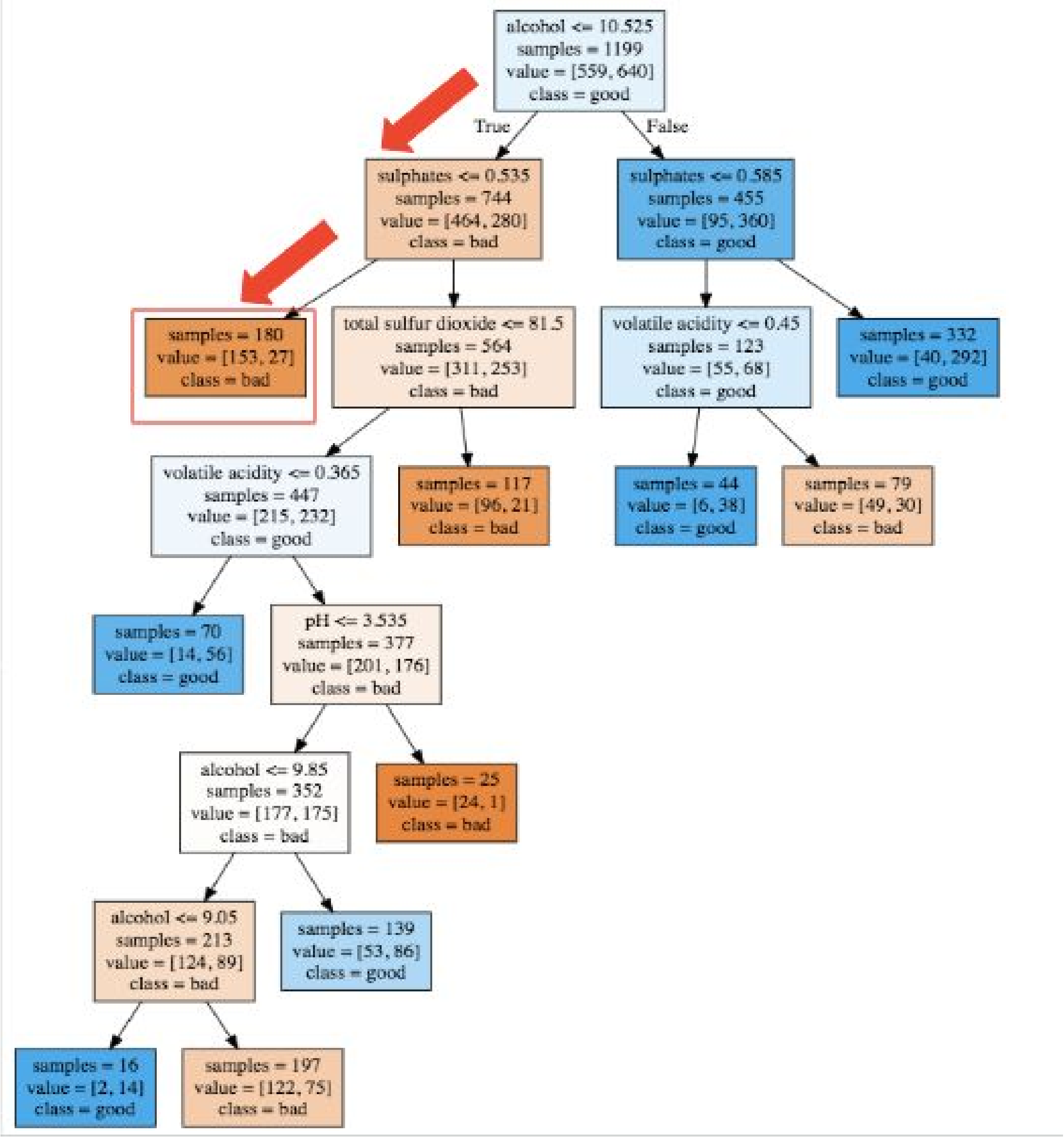
```
print('White 2: SVM g=1 acc on test: {:.3f}'.format(svm2_wlin2.score(Xw_winte2,yw_winte2)))

Whole: SVM g=.1 acc on train: 0.912
Whole: SVM g=.1 acc on test: 0.725
Red: SVM g=.01 acc on train: 0.711
Red: SVM g=.01 acc on test: 0.710
White: SVM g=1 acc on train: 0.998
White: SVM g=1 acc on test: 0.727
White 2: SVM g=1 acc on train: 0.998
White 2: SVM g=1 acc on test: 0.794
```

When analyzing our four datasets with a support vector machine we, similarly too LSVM, reached a high accuracy. You can note that both white, and white 2 held the highest accuracies at Gamma equal to 1. This would show that white wine, unlike red, can be split non-linearly. In other words a “goldilocks zone” or “wiggly line” for white wine vs a linear split for red wine. This is also supported by the fact that red wine preformed best under a low Gamma score. A higher Gamma means that data points closer effect the model more.

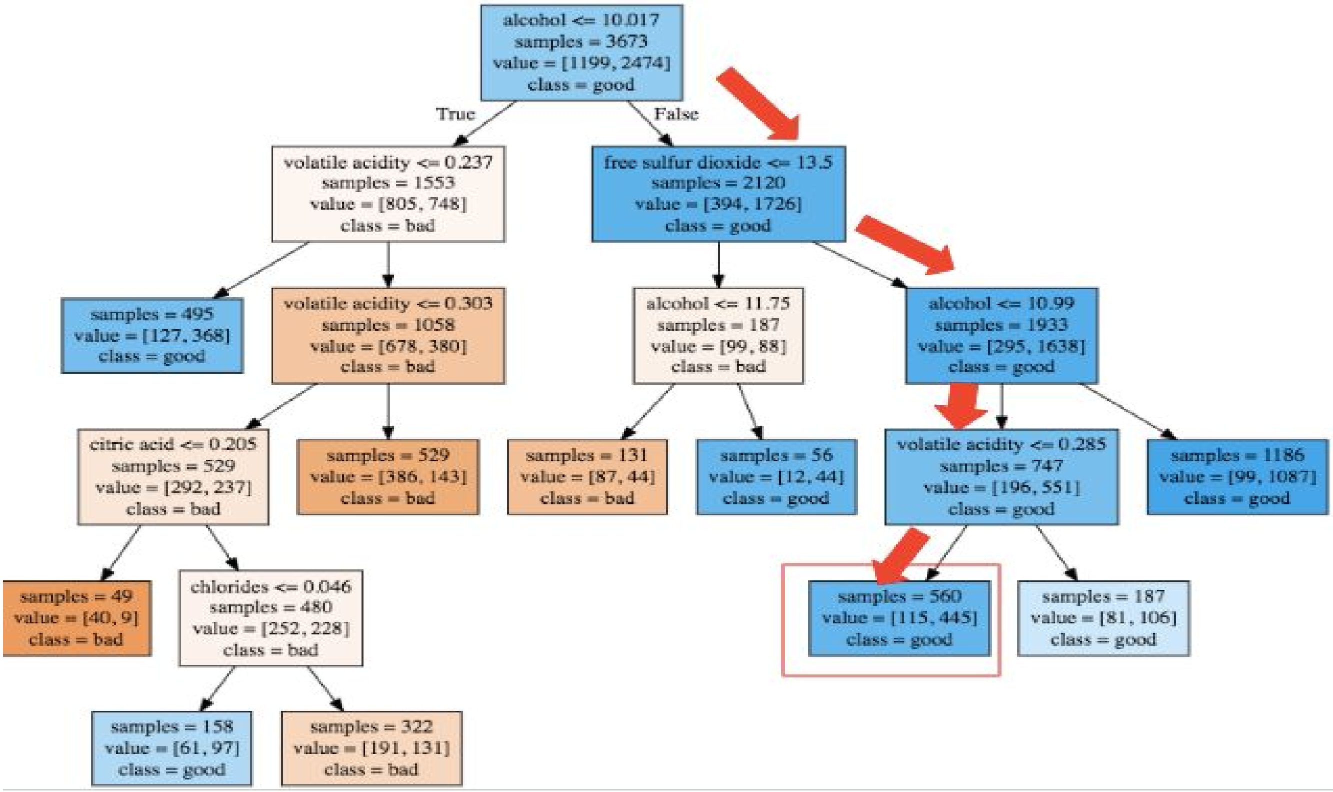


```
Whole: DT of 15 LN acc on train: 0.765
Whole: DT of 15 LN acc on test: 0.738
Red: DT of 20 LN acc on train: 0.820
Red: DT of 20 LN acc on test: 0.735
White: DT of 30 LN acc on train: 0.804
White: DT of 30 LN on test: 0.750
White 2: DT of 30 LN acc on train: 0.770
White 2: DT of 30 LN acc on test: 0.739
```

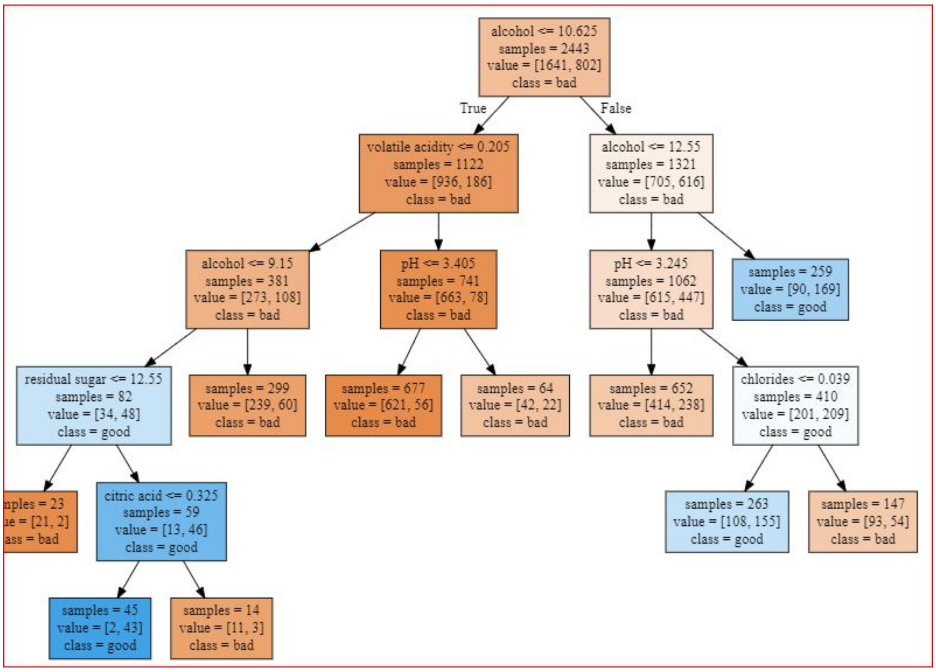
Our decision tree models all had an accuracy of over 73% for all models ran, and we successfully avoided overfitting in all of our datasets. Decision trees allow us to easily visualize, and act upon our findings, which we will visualize and demonstrate. Our best performing model is our white wine dataset. Wine makers can use these decision trees to tweak and modify their wine. By understanding which characteristics are most important, and then by following the leaf nodes of the tree down to slowly tweak and adjust a wine till it classifies as "good".



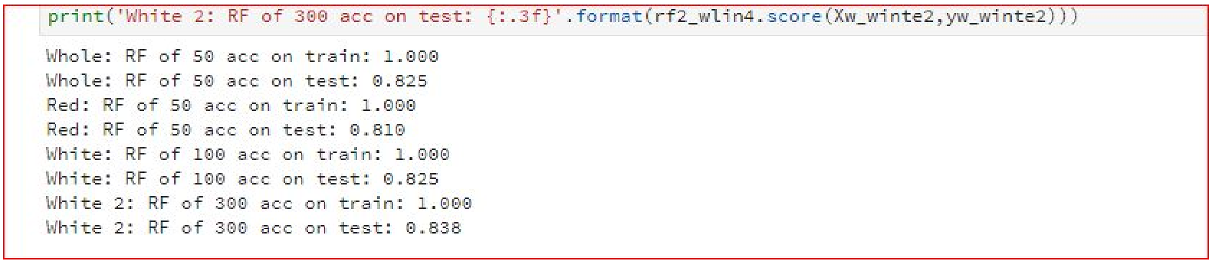
Our group made a 10 leaf decision tree for red wine, and the decision tree is show as following. Here we are predicting an example about how we can find out the results by following the steps. The red wine with the alcohol is 8.8 and sulphates is 0.45, we can see on the graph that alcohol with 8.8 is lower than 10.525 which means we should go to the True direction. To compare sulphates of red wine, we got the sulphates with 0.45, which is much smaller compare to 0.535. Therefore, we finally get the result that the class is bad.


alcohol <= 10.017 samples = 3673 value = [1199, 2474] class = good
True
False
volatile acidity <= 0.237 samples = 1553 value = [805, 748] class = bad
free sulfur dioxide <= 13.5 samples = 2120 value = [394, 1726] class = good
samples = 495 value = [127, 368] class = good
volatile acidity <= 0.303 samples = 1058 value = [678, 380] class = bad
alcohol <= 11.75 samples = 187 value = [99, 88] class = bad
alcohol <= 10.99 samples = 1933 value = [295, 1638] class = good
citric acid <= 0.205 samples = 529 value = [292, 237] class = bad
samples = 529 value = [386, 143] class = bad
samples = 131 value = [87, 44] class = bad
samples = 56 value = [12, 44] class = good
volatile acidity <= 0.285 samples = 747 value = [196, 551] class = good
samples = 1186 value = [99, 1087] class = good
samples = 49 value = [40, 9] class = bad
chlorides <= 0.046 samples = 480 value = [252, 228] class = bad
samples = 560 value = [115, 445] class = good
samples = 187 value = [81, 106] class = good
samples = 158 value = [61, 97] class = good
samples = 322 value = [191, 131] class = bad

We also have 10 leaf decision trees for the white wine. Here is an prediction that when the white wine's alcohol is equal to 10.1, free sulfur dioxide is 30, and the volatile acidity is 0.28. It is obvious that the alcohol 10.1 is lesser than 10.017, so we should follow the false direction to find the number of free sulfur dioxide is whether larger than 13.5 or not. Then we got the value of 30 which is bigger than 13.5; therefore we go to the false direction. After that, the volatile acidity is equal to 0.28 which is smaller than 0.285. So we finally get the result that the class is good.

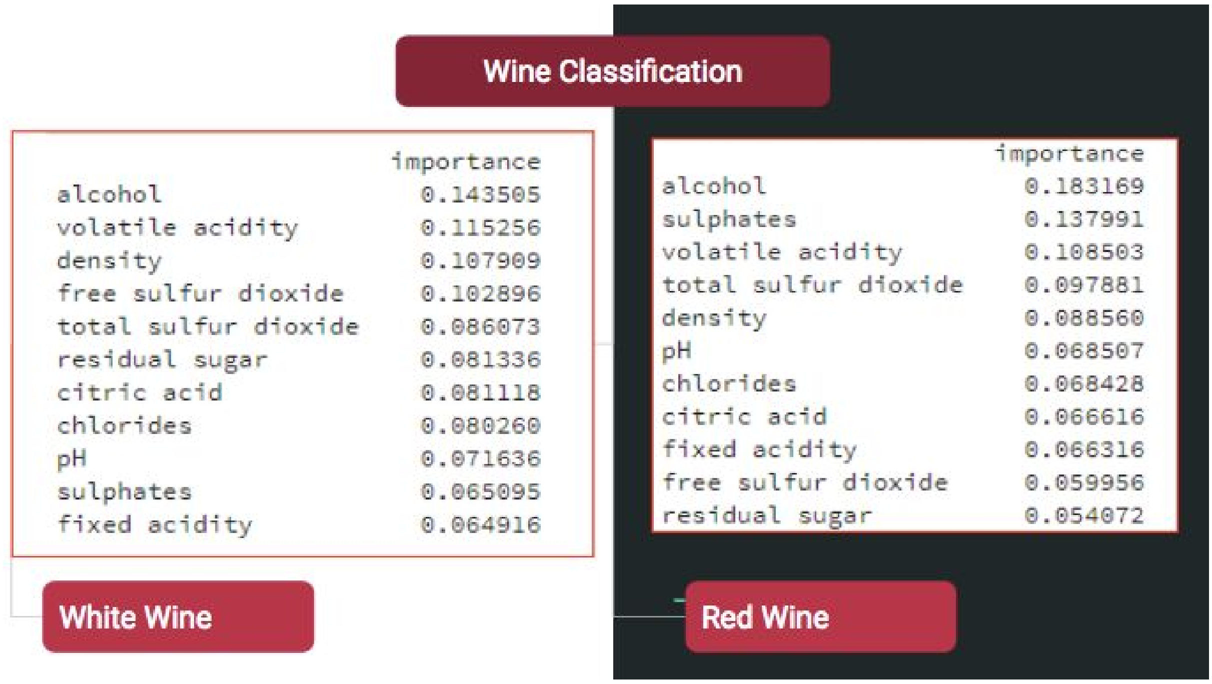

alcohol <= 10.625 samples = 2443 value = [1641, 802] class = bad
True
False
volatile acidity <= 0.205 samples = 1122 value = [936, 186] class = bad
alcohol <= 12.55 samples = 1321 value = [705, 616] class = bad
alcohol <= 9.15 samples = 381 value = [273, 108] class = bad
pH <= 3.405 samples = 741 value = [663, 78] class = bad
pH <= 3.245 samples = 1062 value = [615, 447] class = bad
samples = 259 value = [90, 169] class = good
residual sugar <= 12.55 samples = 82 value = [34, 48] class = good
samples = 299 value = [239, 60] class = bad
samples = 677 value = [621, 56] class = bad
samples = 64 value = [42, 22] class = bad
samples = 652 value = [414, 238] class = bad
chlorides <= 0.039 samples = 410 value = [201, 209] class = good
nples = 23 ue = [21, 2] ass = bad
citric acid <= 0.325 samples = 59 value = [13, 46] class = good
samples = 263 value = [108, 155] class = good
samples = 147 value = [93, 54] class = bad
samples = 45 value = [2, 43] class = good
samples = 14 value = [11, 3] class = bad

The final decision tree above is the decision tree generated with 10 leaf nodes for our white 2 dataset. You can not the small amount of good leaf nodes. This means that for this dataset it is hard to make a wine that has a quality of 7 and above. However, wine makers could use this chart to guide their decision with blending, and adding chemicals to the wine.



```
print('White 2: RF of 300 acc on test: {:.3f}'.format(rf2_wlin4.score(Xw_winte2,yw_winte2)))

Whole: RF of 50 acc on train: 1.000
Whole: RF of 50 acc on test: 0.825
Red: RF of 50 acc on train: 1.000
Red: RF of 50 acc on test: 0.810
White: RF of 100 acc on train: 1.000
White: RF of 100 acc on test: 0.825
White 2: RF of 300 acc on train: 1.000
White 2: RF of 300 acc on test: 0.838
```

The final model we used was the random forest model. This model had the highest accuracy out of any models, and was the model we chose to assess the significance of our features. Interestingly, all training data could be fit completely; however, test data could not break 85% accuracy. I believe the random forest model proves that wine quality can be modeled accurately. Therefore, our data is applicable to the real world, and the importance reports ran from the random forest model are accurate. We initially ran a importance report of the whole wine dataset; however, we disregarded the findings because the results said that a wine being red vs white did not matter. We then proceeded find the importance of the feature variables in the red and white dataset individually.



Wine Classification
White Wine
Red Wine

```
                      importance
alcohol                 0.143505
volatile acidity        0.115256
density                 0.107909
free sulfur dioxide     0.102896
total sulfur dioxide    0.086073
residual sugar          0.081336
citric acid             0.081118
chlorides               0.080260
pH                      0.071636
sulphates               0.065095
fixed acidity           0.064916
```

```
                      importance
alcohol                 0.183169
sulphates               0.137991
volatile acidity        0.108503
total sulfur dioxide    0.097881
density                 0.088560
pH                      0.068507
chlorides               0.068428
citric acid             0.066616
fixed acidity           0.066316
free sulfur dioxide     0.059956
residual sugar          0.054072
```

We can see the difference between white and red wine here. When comparing red and white wine we can see that the importance is different depending on the wine type. Alcohol is the percent alcohol content of the wine. Volatile acidity is the amount of acetic acid in wine, high levels can lead to an unpleasant, vinegar taste. The density of wine is close to that of water depending on the percent alcohol and sugar content. Free sulfur dioxide is the free form of SO2 that exists at equilibrium, it prevents microbial growth and the oxidation of wine. Total sulfur dioxide is the amount of free and bound forms of SO2. In concentrations with free SO2 over 50 ppm, SO2 becomes evident in the nose and taste of wine. Sulphates are a wine additive which can contribute to sulfur dioxide gas (SO2) levels, which acts as an antimicrobial and antioxidant.

Findings and Conclusion

Throughout this project we dissected a dataset that presented two similar, but different wine. While the whole dataset, which had both red and white wines, could create accurate models we found that high quantities of certain scores combined with more white than red data created models that weren't realistic. By breaking apart our data into red and white datasets our models grew in accuracy, and in the applicability of our findings. We found that a linear SVM worked better for red wine, and a SVM worked better for the white wine dataset. We also created decision trees that can be used, and referenced in the wine making process to create good wines. For example, wine makers making a red blend could reference our decision tree when combining many gallons of wine to create a good wine. Furthermore, we also developed highly accurate random forest models that showed the importance of our features. Alcohol was found to be the most important feature of both red and white wine; however, after that white wine and red wine show different characteristics. All these models and findings can be used to develop and optimize wine.

Our dataset was easy to clean, and all the features meant something. We also applied ourselves to learn about wine during our project to understand our findings. Something that would have lended to more accurate and applicable data is having more categorical features. It would have been interesting to see what specific wine was used thereby adding depth to the numeric features we analysed. Furthermore, more research on the process of making wine would have lended to our understanding of applying these results. While the blending process of the wine industry can use these results we do not understand the nuance of wine making. And, that means we don't know all the possibilities of these findings. From the beginning of this project we wanted to predict the quality of wine. We now know that we can accurately predict the quality of wine using multiple models. Furthermore, we also know what characteristics are important in a good wine (red or white), and have a visualization to help foster making good wines.