Wine Data Analysis

Yashwin Aneja
190143446
Group Size: 3

Table of individual contribution by each member of my group:

Student	Effort
Ankit Kottewar	33%
Harnoor Singh Oberai	34%
Yashwin Aneja	33%

Table of Contents

1.Introduction and Background

One of the most common alcoholic drink served at a get-together or a special occasion is Wine. An ideal wine is usually made from the fermented grapes. Yeast acts as a catalyst to break the sugars in the grapes and convert it to ethanol, heat and carbon dioxide. Various percentages of these substances cause complex reactions between the grapes which outputs the formation of wine. There are other ways of producing wine-like fermentation of different crops, rice & fruits like cherry, plum, pomegranate, etc.

The million-dollar question popped to us whenever invited at guest's party is - "Is this fine?"

Sadly, it is difficult to know by just gazing at the bottle and going through its content, we try to find whether that wine is worth trying. Just a screw of cork, a fine glass, and your taste receptors can do that.

Also, this is not sufficient to tell whether a wine is fine or not. We could prefer a wine because of personal preference, but finding out if a wine's quality is good depends upon various factors to search its final quality.

The alcohol industry has seen a significant spurt as drinking socially is increasing as per new trends. The cost price of wine usually depends on the metaphysical concept of favouring wine by the tasters whose value of that wine may change variably. Sometimes the cost price of wine depends on the volatility factor. Other factors in the certification of wine and assessment of quality are lab-based chemical tests which uses factors like sugar, acidity, pH level and other substance properties. Considering the market of wine, it is a fascinating task of finding out of human tasting quality can be influenced by chemical properties of wine to ensure assessment of quality is effective.

2.Project Aims

As such, this project aims to use the data provided by the UC Irvine Machine Learning Repository on two datasets related to red and white variants of the Portuguese "Vinho Verde" wine to gain insights into the characteristics of both red and white wine. Can we understand what makes the attribute such as pH level, acidity, alcohol, etc more or less likely to contribute towards the quality of wine? Observation of red and white wine is made using their chemical attributes and ranking them using the knowledge of wine connoisseur. We have a red wine dataset having 1599 different varieties and white has 4898 varieties. Both the wines are being analyzed. All wines in the data set are from Portugal.

Using the chemical properties of different wines such as alcohol content, pH, acidity, sugar, we use the data collected. We observe that all chemical properties are continuous. We use quality as a label to analyze our model.

3.Literature Review

Previous studies have been conducted to show different approaches for the classification of wine grade. One such example is used in gaussian process regression and multi-task learning which Yeo et al [7] found in his study that advanced machine learning algorithms have a potential for prediction of wine price. Ribeiro et al. [8] in his paper predicted wine vinification using data mining tools that predicted on the best accuracies.

In 1991, a "Wine" data-set was donated to UCI repository which contained 178 instances with 13 unique chemical constituents such as magnesium, alcohol. This data set was used to classify three cultivars from Italy[4]. This was one of the benchmarks in classifying data because it was easily discriminated. PCA(Principal Component Analysis) and was reported in [5]. 33 Greek wines are used to classify wine using physicochemical information which was to be used in wine aroma chromatogram and was measured with Fast GC Analyser [6]. Cortez et al. [1] has proposed in his paper about taste predicting technique. A support vector machine, multiple regression and neural network were applied to carry chemical analysis on wines for taste prediction.

Cortez et al. [9] presented models like neural network, multiple regression, and support vector machine to measure the quality and the taste of wines. Shanmuganathan's [10] in his study predicted the relationship between the various temperatures on wine crops and quality.
Chen et al. [11] focused more on improving the taste and quality of a wine by using the end-users feedback and applied hierarchical clustering of wine which can provide an improved framework for wine classification.

4.Data Retrieval

The data utilized for this analysis was provided by the UC Irvine Machine Learning Repository. Data retrieval was easy as the dataset were readily available for user download. It contains CSV files for white wine and red wine. The dataset obtained were:
winequality-red.csv: white variants of the Portuguese "Vinho Verde" wine.
winequality-red.csv: red variants of the Portuguese "Vinho Verde" wine.

Prior to any data manipulation, it is essential to extract data and transform it into a format that can be easily used in the processing stage. The two datasets used were represented in Panda DataFrame. Below is the same as white and red wine dataset. The rest of the dataset is shown in the appendix.

4.1 White wine:

```
df_w = pd.read_csv('winequality-white.csv', sep=';')
```

```
df_w.head(5)
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.0	0.27	0.36	20.7	0.045	45.0	170.0	1.0010	3.00	0.45	8.8	6
1	6.3	0.30	0.34	1.6	0.049	14.0	132.0	0.9940	3.30	0.49	9.5	6
2	8.1	0.28	0.40	6.9	0.050	30.0	97.0	0.9951	3.26	0.44	10.1	6
3	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	9.9	6
4	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	9.9	6

4.2 Red wine:

```
df_r = pd.read_csv('winequality-red.csv', sep=';')
```

```
df_r.head(5)
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5

5.Data Representation

The main language we have chosen to conduct our analysis is Python for its vast data ecosystem and its powerful libraries and packages that are required for our analytical needs.

The following libraries were used:

- **Numpy** : It is a library for Python language, which adds support for huge nd arrays and matrices, together with a big collection of major mathematical operations or functions to apply on these arrays.
- **Pandas**: It is a library for Python Language, used for data manipulation and analysis. it offers data frames and calculations for interpreting numeric tables and other time series.

- **Matplotlib**: It is a thorough library for python, primarily used for creating stationary, interactive and active visualizations.It significantly supplements pandas by providing graphics to better to study and investigate the data.
- **Seaborn**: Seaborn is a Python data visualization library based on matplotlib. It provides an advanced interface for plotting striking informative statistical graphics.
- **Scikit-learn:** It encapsulates different, regression,classification and cluster algorithms including support vector machines, gradient boosting, random forests,k-means and dBSCAN and is made to interact with the scientific libraries of Python -NumPy & SciPy.
- The other language that we have opted for our data exploration is R for its marvelous visualization and the capability to make understandable graphical plots. We have also used R to explore data and gather main insights about the the variables and data.

The below libraries were used:

- **ggplot2**: ggplot2 is a package for data visualization for the statistical programming language R. ggplot2 serves as alternative for the base graphics of R and contains several packages for web and printing display of common scales.
- **Dplyr:** dplyr is yet another substantial package of R to manipulate and summarise table data with columns and rows. It also gives a set of tools for effectively transforming datasets in R.
- **DataExplorer**: This package of R aims to mechanize most of the data handling and visualisation to ensure users can focus on learning the data and pulling insights.

5.1Data Cleaning and Pre-processing

We transform our data into a plausable format by pre-porcessing our data. Pre-processing data is a crucial factor in data mining which helps to improve data efficiency. Efficent pre-processing would directly improve the outcomes of any analytic algorithm.

Data pre-processing involves the transformation of the raw dataset into an understandable format. Pre-processing data is a fundamental stage in data mining to improve data efficiency since the pre-processing methods directly affect the outcomes of any analytic algorithm.

The Data cleaning procedures that we have used are :

- Removing Duplicates:
  Merging the two datasets to obtain one combined dataset led to a lot of duplicated entries which, if left unchecked will reduce performance and accuracy. Drop_duplicates() function was used to drop the duplicates.

```
]: sum(df_r.duplicated())
]: 240

]: sum(df_w.duplicated())
]: 937
```

- Removing Missing Values:
  There were various observations were missing values were present in the dataset. Missing data can have a significant negative impact in the model performance and accuracy that therefore we decided to remove those observations. Dropna() function was used for the same.

```
def missing_values_table(df):
        mis_val = df.isnull().sum()
        mis_val_percent = 100 * df.isnull().sum() / len(df)
        mis_val_table = pd.concat([mis_val, mis_val_percent], axis=1)
        mis_val_table_ren_columns = mis_val_table.rename(
        columns = {0 : 'Missing Values', 1 : '% of Total Values'})
        mis_val_table_ren_columns = mis_val_table_ren_columns[
            mis_val_table_ren_columns.iloc[:,1] != 0].sort_values(
        '% of Total Values', ascending=False).round(1)
        print ("Your selected dataframe has " + str(df.shape[1]) + " columns.\n"
            "There are " + str(mis_val_table_ren_columns.shape[0]) +
              " columns that have missing values.")
        return mis_val_table_ren_columns
```

- Merging Datasets into one dataframe:
  Since we obtained datasets from two different sources, they had to be combined into one to be processed and worked upon. Pandas merge function was used to merge the two datasets. Pd.merge()

6.Data Exploration and Visualisation

Data Exploration is one of the most important steps in any data analytics project as it helps in identifying key insights and trends that would not be detected without exploration. These findings, in turn, play a major role in optimisations and increasing the accuracy and performance of the model. These techniques also enable us to understand our dataset better to perform and illustrate better visualisations of the data.
R has been used since it has powerful packages to conduct analysis.
The **Red Wine** dataset has :

Observations : 1,599
Variables: 12

The **White Wine** dataset has :

Observations : 4,898
Variables: 12

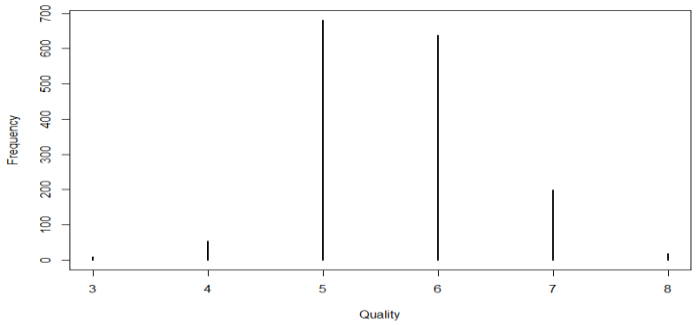
6.1Red Wine Dataset Statistics:

```
> summary(winequality.red)
 fixed.acidity    volatile.acidity  citric.acid     residual.sugar     chlorides       free.sulfur.dioxide
 Min.   : 4.60   Min.   :0.1200   Min.   :0.000   Min.   : 0.900   Min.   :0.01200   Min.   : 1.00
 1st Qu.: 7.10   1st Qu.:0.3900   1st Qu.:0.090   1st Qu.: 1.900   1st Qu.:0.07000   1st Qu.: 7.00
 Median : 7.90   Median :0.5200   Median :0.260   Median : 2.200   Median :0.07900   Median :14.00
 Mean   : 8.32   Mean   :0.5278   Mean   :0.271   Mean   : 2.539   Mean   :0.08747   Mean   :15.87
 3rd Qu.: 9.20   3rd Qu.:0.6400   3rd Qu.:0.420   3rd Qu.: 2.600   3rd Qu.:0.09000   3rd Qu.:21.00
 Max.   :15.90   Max.   :1.5800   Max.   :1.000   Max.   :15.500   Max.   :0.61100   Max.   :72.00
 total.sulfur.dioxide    density             pH          sulphates         alcohol         quality
 Min.   :  6.00       Min.   :0.9901   Min.   :2.740   Min.   :0.3300   Min.   : 8.40   Min.   :3.000
 1st Qu.: 22.00       1st Qu.:0.9956   1st Qu.:3.210   1st Qu.:0.5500   1st Qu.: 9.50   1st Qu.:5.000
 Median : 38.00       Median :0.9968   Median :3.310   Median :0.6200   Median :10.20   Median :6.000
 Mean   : 46.47       Mean   :0.9967   Mean   :3.311   Mean   :0.6581   Mean   :10.42   Mean   :5.636
 3rd Qu.: 62.00       3rd Qu.:0.9978   3rd Qu.:3.400   3rd Qu.:0.7300   3rd Qu.:11.10   3rd Qu.:6.000
 Max.   :289.00       Max.   :1.0037   Max.   :4.010   Max.   :2.0000   Max.   :14.90   Max.   :8.000
```

```
> glimpse(winequality.red)
Observations: 1,599
Variables: 12
$ fixed.acidity        <dbl> 7.4, 7.8, 7.8, 11.2, 7.4, 7.4, 7.9, 7.3, 7...
$ volatile.acidity     <dbl> 0.700, 0.880, 0.760, 0.280, 0.700, 0.660, ...
$ citric.acid          <dbl> 0.00, 0.00, 0.04, 0.56, 0.00, 0.00, 0.06, ...
$ residual.sugar       <dbl> 1.9, 2.6, 2.3, 1.9, 1.9, 1.8, 1.6, 1.2, 2....
$ chlorides            <dbl> 0.076, 0.098, 0.092, 0.075, 0.076, 0.075, ...
$ free.sulfur.dioxide  <dbl> 11, 25, 15, 17, 11, 13, 15, 15, 9, 17, 15,...
$ total.sulfur.dioxide <dbl> 34, 67, 54, 60, 34, 40, 59, 21, 18, 102, 6...
$ density              <dbl> 0.9978, 0.9968, 0.9970, 0.9980, 0.9978, 0....
$ pH                   <dbl> 3.51, 3.20, 3.26, 3.16, 3.51, 3.51, 3.30, ...
$ sulphates            <dbl> 0.56, 0.68, 0.65, 0.58, 0.56, 0.56, 0.46, ...
$ alcohol              <dbl> 9.4, 9.8, 9.8, 9.8, 9.4, 9.4, 9.4, 10.0, 9...
$ quality              <int> 5, 5, 5, 6, 5, 5, 5, 7, 7, 5, 5, 5, 5, 5, ...
```

Quality is the feature that we are predicting, Frequency distribution of quality in the dataset:

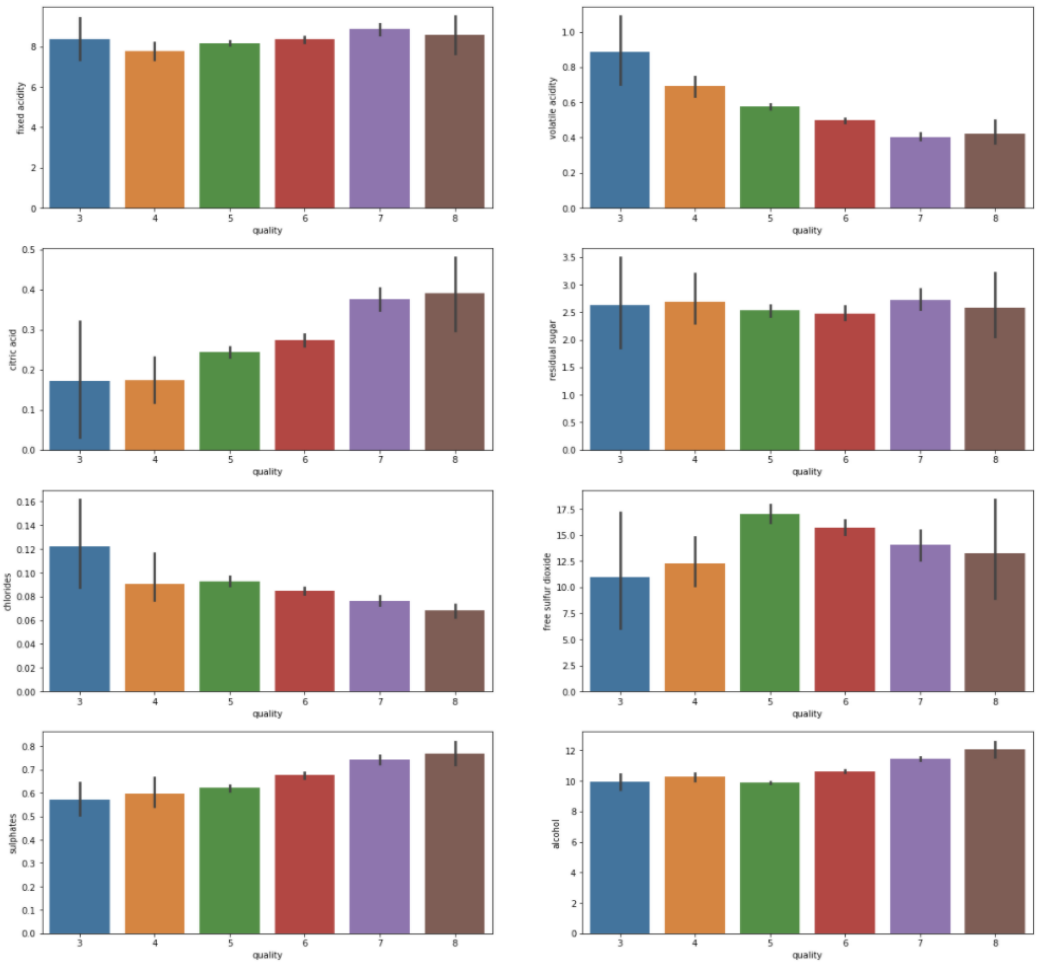
```
> table(winequality.red[["quality"]])

  3   4   5   6   7   8
 10  53 681 638 199  18
```



Red wine stats against quality:

In [4]:
```
#Here we see that fixed acidity does not give any specification to classify the quality.
plt.figure(figsize = (20,20))
plt.subplot(4, 2, 1)
sns.barplot(x = 'quality', y = 'fixed acidity', data = wine)
plt.subplot(4, 2, 2)
sns.barplot(x = 'quality', y = 'volatile acidity', data = wine)
plt.subplot(4, 2, 3)
sns.barplot(x = 'quality', y = 'citric acid', data = wine)
plt.subplot(4, 2, 4)
sns.barplot(x = 'quality', y = 'residual sugar', data = wine)
plt.subplot(4, 2, 5)
sns.barplot(x = 'quality', y = 'chlorides', data = wine)
plt.subplot(4, 2, 6)
sns.barplot(x = 'quality', y = 'free sulfur dioxide', data = wine)
plt.subplot(4, 2, 7)
sns.barplot(x = 'quality', y = 'sulphates', data = wine)
plt.subplot(4, 2, 8)
sns.barplot(x = 'quality', y = 'alcohol', data = wine)
```

Out[4]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x119e2e7b8>
```

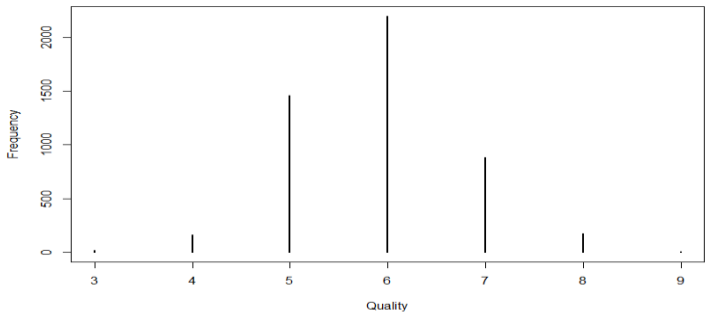


6.2White Wine Statistics:

```
> glimpse(winequality.white)
Observations: 4,898
Variables: 12
$ fixed.acidity        <dbl> 7.0, 6.3, 8.1, 7.2, 7.2, 8.1, 6.2, 7.0, 6.3, 8.1, 8.1, 8.6, 7.9, 6.6, 8.3, 6.6, 6.3, 6.2, 7...
$ volatile.acidity     <dbl> 0.27, 0.30, 0.28, 0.23, 0.23, 0.28, 0.32, 0.27, 0.30, 0.22, 0.27, 0.23, 0.18, 0.16, 0.42, 0...
$ citric.acid          <dbl> 0.36, 0.34, 0.40, 0.32, 0.32, 0.40, 0.16, 0.36, 0.34, 0.43, 0.41, 0.40, 0.37, 0.40, 0.62, 0...
$ residual.sugar       <dbl> 20.70, 1.60, 6.90, 8.50, 8.50, 6.90, 7.00, 20.70, 1.60, 1.50, 1.45, 4.20, 1.20, 1.50, 19.25...
$ chlorides            <dbl> 0.045, 0.049, 0.050, 0.058, 0.058, 0.050, 0.045, 0.045, 0.049, 0.044, 0.033, 0.035, 0.040, ...
$ free.sulfur.dioxide  <dbl> 45, 14, 30, 47, 47, 30, 30, 45, 14, 28, 11, 17, 16, 48, 41, 28, 30, 29, 17, 34, 29, 19, 41,...
$ total.sulfur.dioxide <dbl> 170, 132, 97, 186, 186, 97, 136, 170, 132, 129, 63, 109, 75, 143, 172, 112, 99, 75, 171, 13...
$ density              <dbl> 1.0010, 0.9940, 0.9951, 0.9956, 0.9956, 0.9951, 0.9949, 1.0010, 0.9940, 0.9938, 0.9908, 0.9...
$ pH                   <dbl> 3.00, 3.30, 3.26, 3.19, 3.19, 3.26, 3.18, 3.00, 3.30, 3.22, 2.99, 3.14, 3.18, 3.54, 2.98, 3...
$ sulphates            <dbl> 0.45, 0.49, 0.44, 0.40, 0.40, 0.44, 0.47, 0.45, 0.49, 0.45, 0.56, 0.53, 0.63, 0.52, 0.67, 0...
$ alcohol              <dbl> 8.8, 9.5, 10.1, 9.9, 9.9, 10.1, 9.6, 8.8, 9.5, 11.0, 12.0, 9.7, 10.8, 12.4, 9.7, 11.4, 9.6,...
$ quality              <int> 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 5, 5, 5, 7, 5, 7, 6, 8, 6, 5, 8, 7, 8, 5, 6, 6, 6, 6, 6, 7, 6...
> summary(winequality.white)
 fixed.acidity    volatile.acidity  citric.acid     residual.sugar     chlorides       free.sulfur.dioxide
 Min.   : 3.800   Min.   :0.0800   Min.   :0.0000   Min.   : 0.600   Min.   :0.00900   Min.   :  2.00
 1st Qu.: 6.300   1st Qu.:0.2100   1st Qu.:0.2700   1st Qu.: 1.700   1st Qu.:0.03600   1st Qu.: 23.00
 Median : 6.800   Median :0.2600   Median :0.3200   Median : 5.200   Median :0.04300   Median : 34.00
 Mean   : 6.855   Mean   :0.2782   Mean   :0.3342   Mean   : 6.391   Mean   :0.04577   Mean   : 35.31
 3rd Qu.: 7.300   3rd Qu.:0.3200   3rd Qu.:0.3900   3rd Qu.: 9.900   3rd Qu.:0.05000   3rd Qu.: 46.00
 Max.   :14.200   Max.   :1.1000   Max.   :1.6600   Max.   :65.800   Max.   :0.34600   Max.   :289.00
 total.sulfur.dioxide    density             pH           sulphates         alcohol          quality
 Min.   :  9.0        Min.   :0.9871   Min.   :2.720   Min.   :0.2200   Min.   : 8.00   Min.   :3.000
 1st Qu.:108.0        1st Qu.:0.9917   1st Qu.:3.090   1st Qu.:0.4100   1st Qu.: 9.50   1st Qu.:5.000
 Median :134.0        Median :0.9937   Median :3.180   Median :0.4700   Median :10.40   Median :6.000
 Mean   :138.4        Mean   :0.9940   Mean   :3.188   Mean   :0.4898   Mean   :10.51   Mean   :5.878
 3rd Qu.:167.0        3rd Qu.:0.9961   3rd Qu.:3.280   3rd Qu.:0.5500   3rd Qu.:11.40   3rd Qu.:6.000
 Max.   :440.0        Max.   :1.0390   Max.   :3.820   Max.   :1.0800   Max.   :14.20   Max.   :9.000
```

Frequency distribution of quality in the dataset:

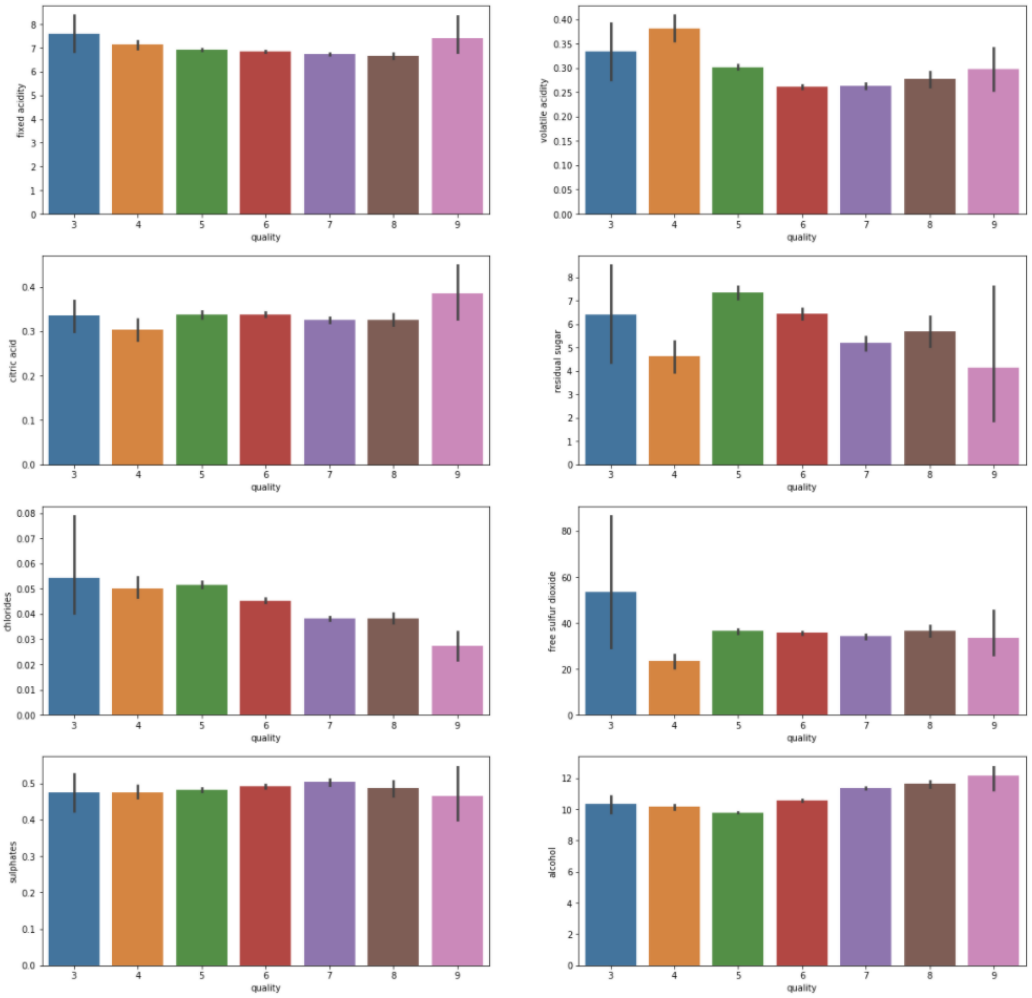
```
table(winequality.white[["quality"]])

  3   4   5   6   7   8   9
 20 163 1457 2198 880 175   5
```



White wine stats Vs quality:

In [5]:
```
#Here we see that fixed acidity does not give any specification to classify the quality.
plt.figure(figsize = (20,20))
plt.subplot(4, 2, 1)
sns.barplot(x = 'quality', y = 'fixed acidity', data = wine)
plt.subplot(4, 2, 2)
sns.barplot(x = 'quality', y = 'volatile acidity', data = wine)
plt.subplot(4, 2, 3)
sns.barplot(x = 'quality', y = 'citric acid', data = wine)
plt.subplot(4, 2, 4)
sns.barplot(x = 'quality', y = 'residual sugar', data = wine)
plt.subplot(4, 2, 5)
sns.barplot(x = 'quality', y = 'chlorides', data = wine)
plt.subplot(4, 2, 6)
sns.barplot(x = 'quality', y = 'free sulfur dioxide', data = wine)
plt.subplot(4, 2, 7)
sns.barplot(x = 'quality', y = 'sulphates', data = wine)
plt.subplot(4, 2, 8)
sns.barplot(x = 'quality', y = 'alcohol', data = wine)
```

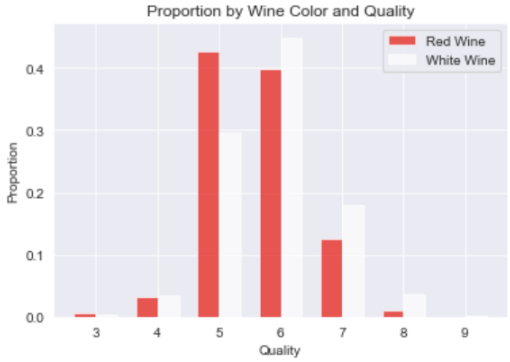
Out[5]: <matplotlib.axes._subplots.AxesSubplot at 0x12351cef0>



6.3 Proportion by wine color quality:

```
# plot bars
red_bars = plt.bar(ind, red_proportions, width, color='r', alpha=.7, label='Red Wine')
white_bars = plt.bar(ind + width, white_proportions, width, color='w', alpha=.7, label='White W

# title and labels
plt.ylabel('Proportion')
plt.xlabel('Quality')
plt.title('Proportion by Wine Color and Quality')
locations = ind + width / 2  # xtick locations
labels = ['3', '4', '5', '6', '7', '8', '9']  # xtick labels
plt.xticks(locations, labels)

# legend
plt.legend()
```

```
<matplotlib.legend.Legend at 0x1250333c8>
```



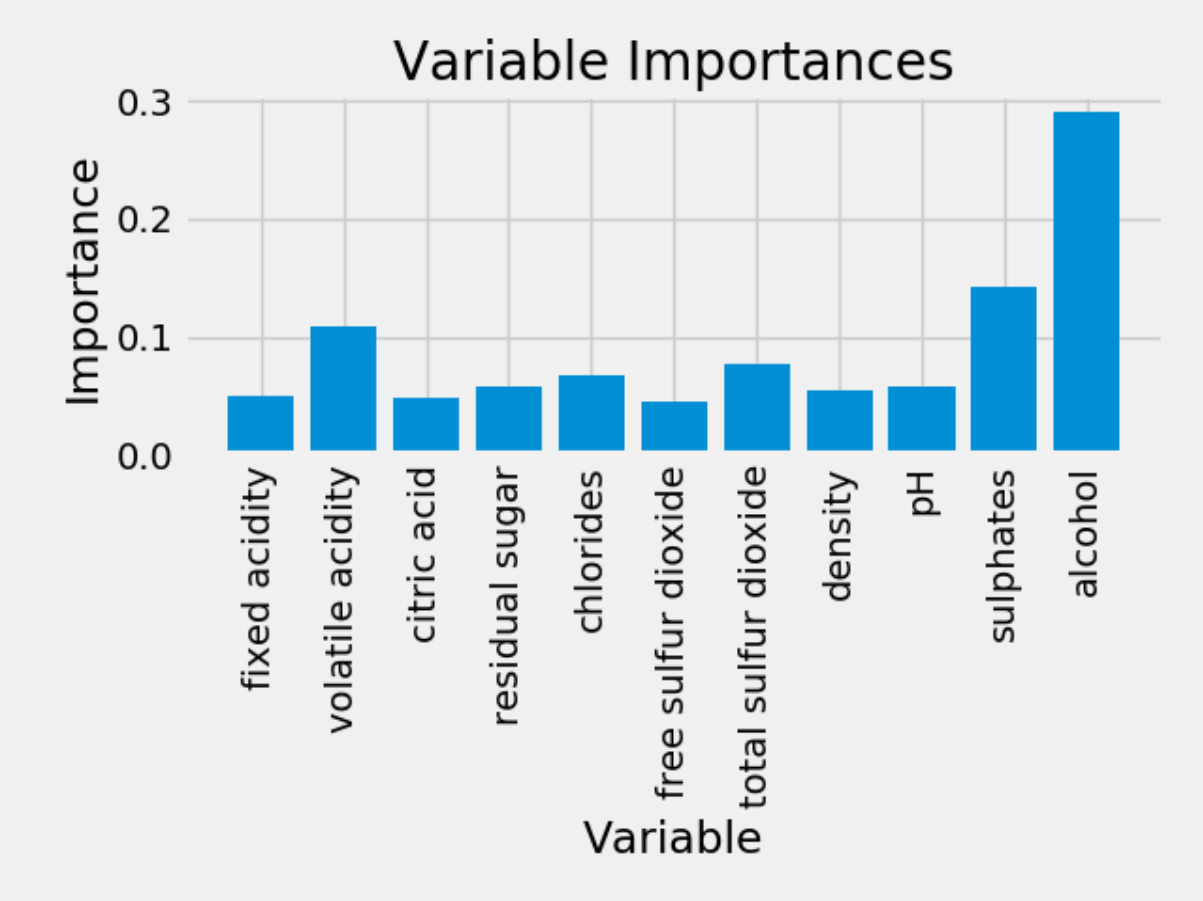
9.Feature Selection

We used feature_importances_ function from scikit-learn to evaluate the importance of features on the label. Our observation was that 3 features have the most impact, the rest did not. However it fascinating to find that the impact of the features on our label Quality is different for both Red and White Wines.

In the Red Wine dataset using Random Forest model :
Variable Importance were as follows

Variable: alcohol Importance: 0.29
Variable: sulphates Importance: 0.14
Variable: volatile acidity Importance: 0.11
Variable: total sulfur dioxide Importance: 0.08
Variable: chlorides Importance: 0.07
Variable: residual sugar Importance: 0.06
Variable: pH Importance: 0.06
Variable: fixed acidity Importance: 0.05
Variable: citric acid Importance: 0.05
Variable: density Importance: 0.05
Variable: free sulfur dioxide Importance: 0.04

Mean Absolute Error before feature selection: 0.42 degrees.
Accuracy before feature selection: 92.27 %

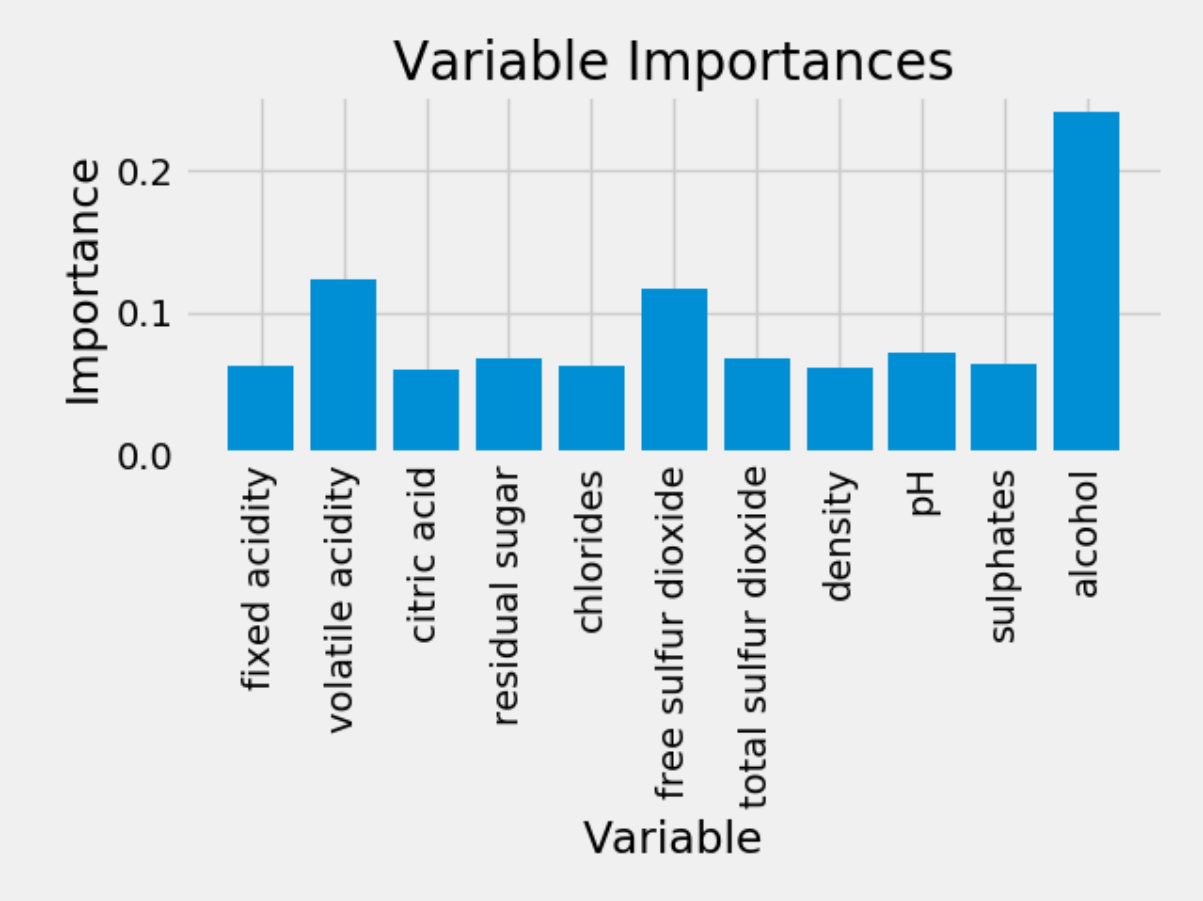


We used the 2 most important variables which are alcohol(29%) and sulphates(14%) and then found out the accuracy, the results obtained were –
Mean Absolute Error: 0.54 degrees.
Accuracy using Important Variables: 90.2 %.

We observed that by using the important features and ignoring the rest our accuracy dropped approx. ~2.25% and mean absolute error increased by ~12 degrees, therefore we decide to use all the features.

White Wine dataset using Random Forest model :
Similar observations were found in the white wine dataset, if we use only the most important features, our accuracy drops and mean absolute error increases, therefore we decided to use all features.

Accuracy Before feature selection: 92.38 %.
Mean Absolute Error before feature selection: 0.42 degrees.
Variable Importance were as follows:
Variable: alcohol Importance: 0.24
Variable: volatile acidity Importance: 0.12
Variable: free sulfur dioxide Importance: 0.12
Variable: residual sugar Importance: 0.07
Variable: total sulfur dioxide Importance: 0.07
Variable: pH Importance: 0.07
Variable: fixed acidity Importance: 0.06
Variable: citric acid Importance: 0.06
Variable: chlorides Importance: 0.06
Variable: density Importance: 0.06
Variable: sulphates Importance: 0.06
Mean Absolute Error: 0.59 degrees.
Accuracy using Important Variables: 89.51 %.



8.Models

8.1 SVM classifier:

Support Vector Machine is a supervised learning model that analyse wine data which will be used to classify whether the wine is worth drinking(good or bad). Feeding the model with our dataset and features, SVM builds a model with a set of rules which will help us to assign into either good or bad class.

Model Implementation:

- Creating 2 bins from our label quality and assigning 0(bad) or 1(good). A wine is good if the quality is between 2.5 to 6.5 and good if the wine is between 6.5 to highest value(8 for red, 9 for white).
- Split the data into train and testing
- x_train = features obtain from our data frame.
- y = label(quality) obtain from our dataset.
- Build and run the model.
- Use grid-search-cv to find best hyperparameters.
- Train the model using the hyperparameters obtained
- Check the accuracy of the model.

To further increase the accuracy of the model, we used Grid Search cv. Grid-search is used to find the best optimal hyperparameters of a model that will result in the best accurate predictions. Below figure gives us the snippet to use Grid-Search. best_params_ method gives us the best value of C, kernel and gamma associated with the model.

Grid Search CV

```
In [16]: #Finding best parameters for our SVC model
         param = {
             'C': [0.1,0.8,0.9,1,1.1,1.2,1.3,1.4],
             'kernel':['linear', 'rbf'],
             'gamma' :[0.1,0.8,0.9,1,1.1,1.2,1.3,1.4]
         }
         grid_svc = GridSearchCV(svc, param_grid=param, scoring='accuracy', cv=10)

In [17]: grid_svc.fit(X_train, y_train)

Out[17]: GridSearchCV(cv=10, error_score=nan,
                      estimator=SVC(C=1.0, break_ties=False, cache_size=200,
                                    class_weight=None, coef0=0.0,
                                    decision_function_shape='ovr', degree=3,
                                    gamma='scale', kernel='rbf', max_iter=-1,
                                    probability=False, random_state=None, shrinking=True,
                                    tol=0.001, verbose=False),
                      iid='deprecated', n_jobs=None,
                      param_grid={'C': [0.1, 0.8, 0.9, 1, 1.1, 1.2, 1.3, 1.4],
                                  'gamma': [0.1, 0.8, 0.9, 1, 1.1, 1.2, 1.3, 1.4],
                                  'kernel': ['linear', 'rbf']},
                      pre_dispatch='2*n_jobs', refit=True, return_train_score=False,
                      scoring='accuracy', verbose=0)

In [18]: #Best parameters for our svc model
         grid_svc.best_params_

Out[18]: {'C': 1.2, 'gamma': 0.9, 'kernel': 'rbf'}
```

8.1.1 White wine:

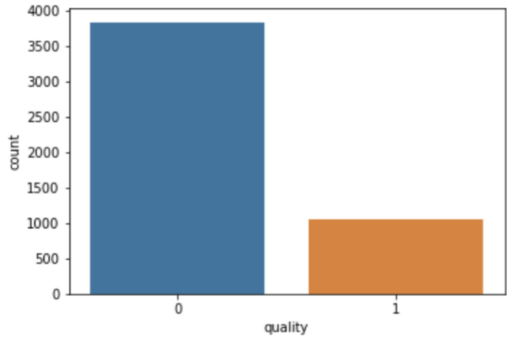
Our classification reports show us that our model was 87% precise predicting our test value with 85% accuracy.

```
In [20]: #Let's run our SVC again with the best parameters.
         svc2 = SVC(C = 1.2, gamma =  0.9, kernel= 'rbf')
         svc2.fit(X_train, y_train)
         pred_svc2 = svc2.predict(X_test)
         print(classification_report(y_test, pred_svc2))
```

```
              precision    recall  f1-score   support

           0       0.87      0.96      0.91       753
           1       0.81      0.51      0.63       227

    accuracy                           0.86       980
   macro avg       0.84      0.74      0.77       980
weighted avg       0.85      0.86      0.85       980
```

SVC improves from 81% to 85% using Grid Search CV

```
In [9]: sns.countplot(wine['quality'])

Out[9]: <matplotlib.axes._subplots.AxesSubplot at 0x1236c2fd0>
```



Support Vector Classifier

```
svc = SVC()
svc.fit(X_train, y_train)
pred_svc = svc.predict(X_test)
```

```
print(classification_report(y_test, pred_svc))
```

```
              precision    recall  f1-score   support

           0       0.83      0.97      0.89       753
           1       0.75      0.34      0.47       227

    accuracy                           0.82       980
   macro avg       0.79      0.65      0.68       980
weighted avg       0.81      0.82      0.79       980
```

81% accuracy using stochastic gradient descent classifier

```
print(confusion_matrix(y_test, pred_svc))
```

```
[[727  26]
 [150  77]]
```

8.1.2 Red Wine:

Our classification reports show us that our model was 90% precise predicting our test value with 90% accuracy.

```
[19]: #Let's run our SVC again with the best parameters.
svc2 = SVC(C = 1.2, gamma =  0.9, kernel= 'rbf')
svc2.fit(X_train, y_train)
pred_svc2 = svc2.predict(X_test)
print(classification_report(y_test, pred_svc2))
```

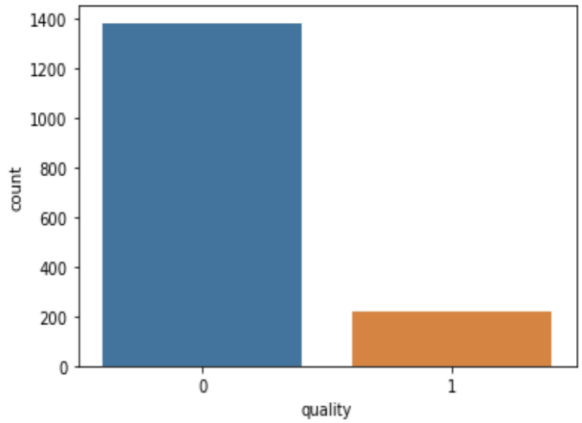
```
              precision    recall  f1-score   support

           0       0.90      0.99      0.94       273
           1       0.89      0.34      0.49        47

    accuracy                           0.90       320
   macro avg       0.89      0.67      0.72       320
weighted avg       0.90      0.90      0.88       320
```

SVC improves from 86% to 90% using Grid Search CV

```
[ ]:
```

In [8]:
```
sns.countplot(wine['quality'])
```

Out[8]: <matplotlib.axes._subplots.AxesSubplot at 0x11f7f2278>



Support Vector Classifier

n [13]:
```
svc = SVC()
svc.fit(X_train, y_train)
pred_svc = svc.predict(X_test)
```

n [14]:
```
print(classification_report(y_test, pred_svc))
```

```
              precision    recall  f1-score   support

           0       0.88      0.98      0.93       273
           1       0.71      0.26      0.37        47

    accuracy                           0.88       320
   macro avg       0.80      0.62      0.65       320
weighted avg       0.86      0.88      0.85       320
```

86% accuracy using stochastic gradient descent classifier

n [15]:
```
print(confusion_matrix(y_test, pred_svc))
```

```
[[268   5]
 [ 35  12]]
```

8.2 Random Forest :

Random forests or random decision forests are an ensemble learning method under supervised learning which will be used for mean prediction (regression) of individual random tress of wine datasets. Feeding the model with our dataset and features, Random Forest builds a tree model with a set of rules which will help us to predict wine quality.

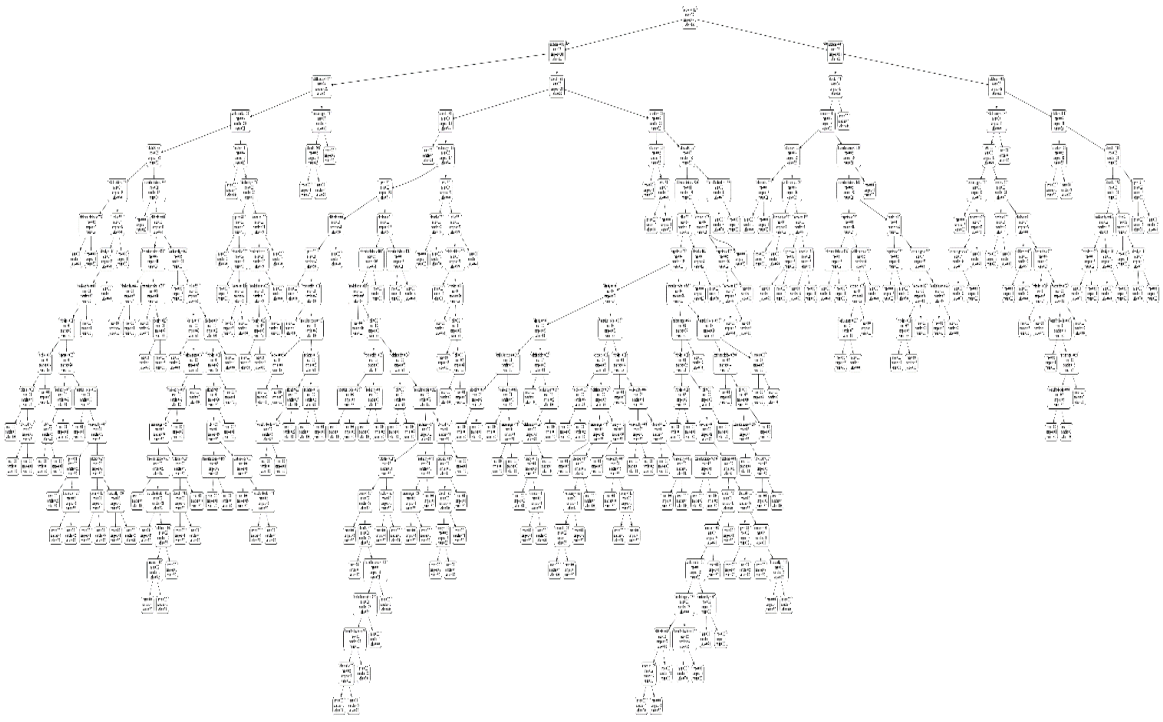
Model Implementation:

- Separate features and label from the dataset, assigning them into 2 respective variables
- Separate both features and label dataset into test and train. The train:test split will be 75:25%.
- Build and run the model
- Check the accuracy of the model and the mean absolute error
- Find the most important features using the random forest feature importance method.
- Create an actual vs predicted values graph based on the test results

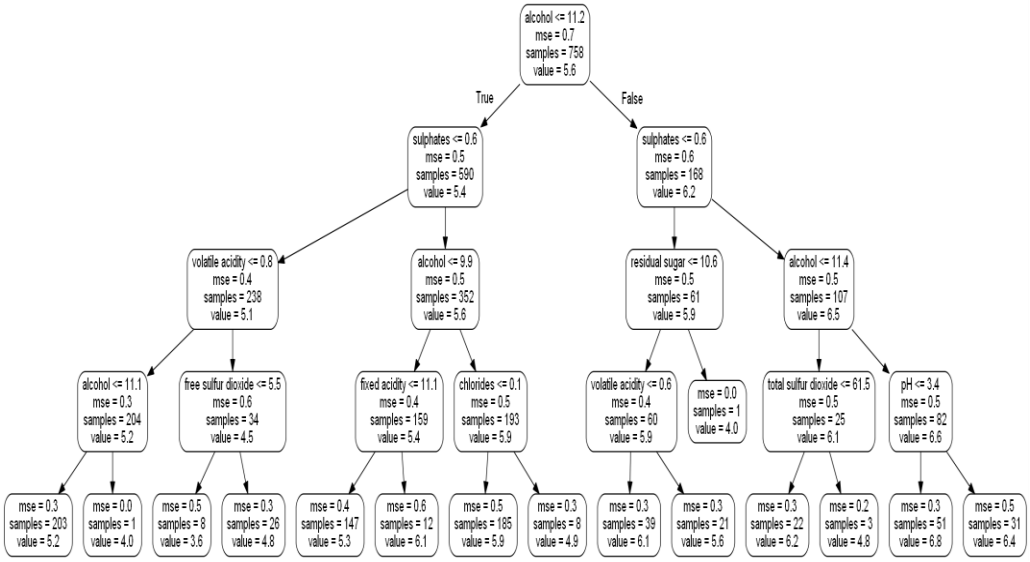
To further improve readability and ease, we have also created a small regression tree with max depth 4 and number of estimators to 10. Below 2 figures contain the two results.

8.2.1Red Wine:

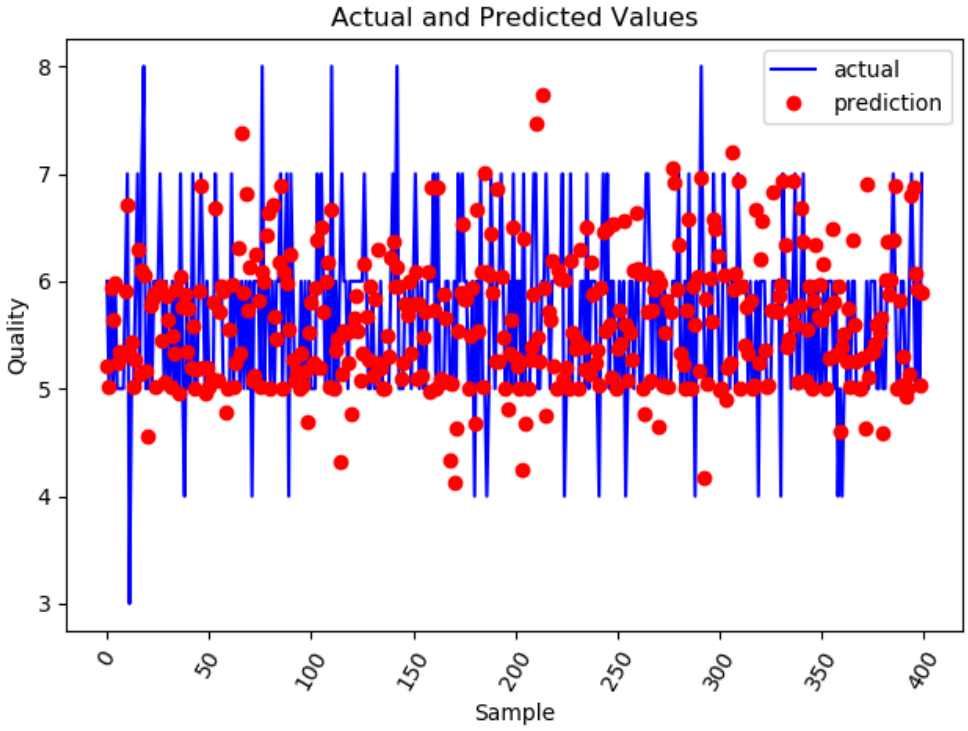
Available at: https://github.com/HarnoorOberai/WineDataAnalysis/blob/master/Random-Forest/tree.png



Small Tree –



Actual vs Predicted Values –

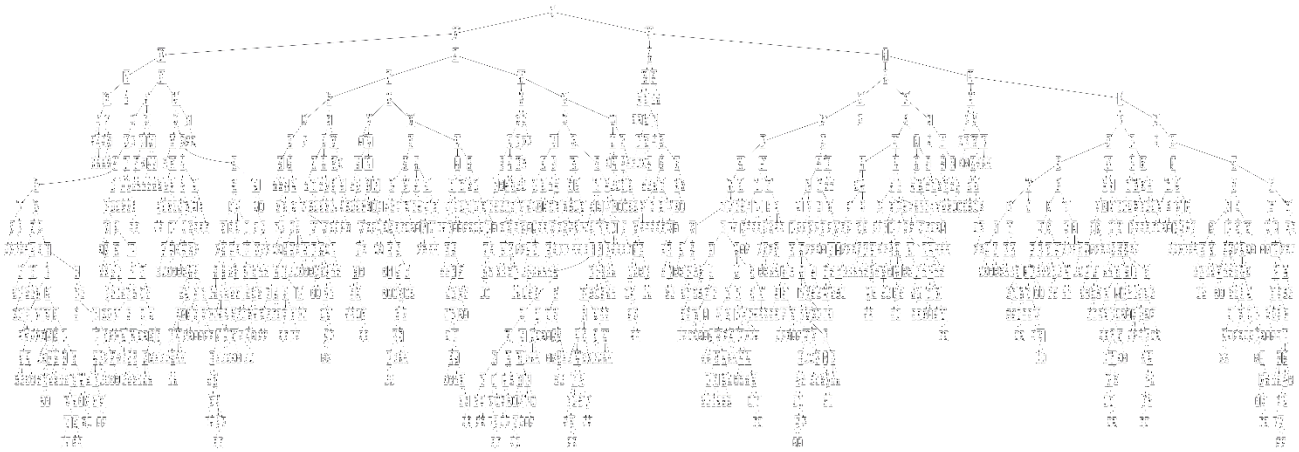


The reports show that the model has :

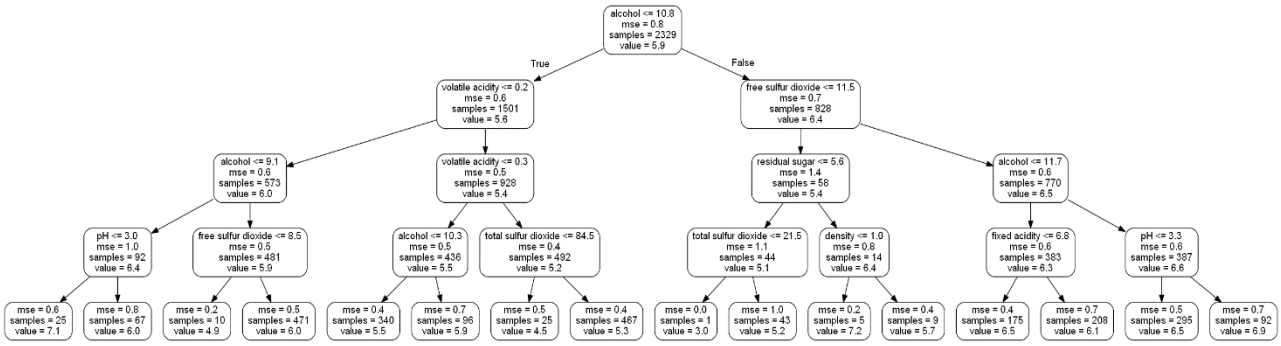
Mean Absolute Error: 0.42 degrees.
Prediction Accuracy: 92.27 %

8.2.2White Wine:

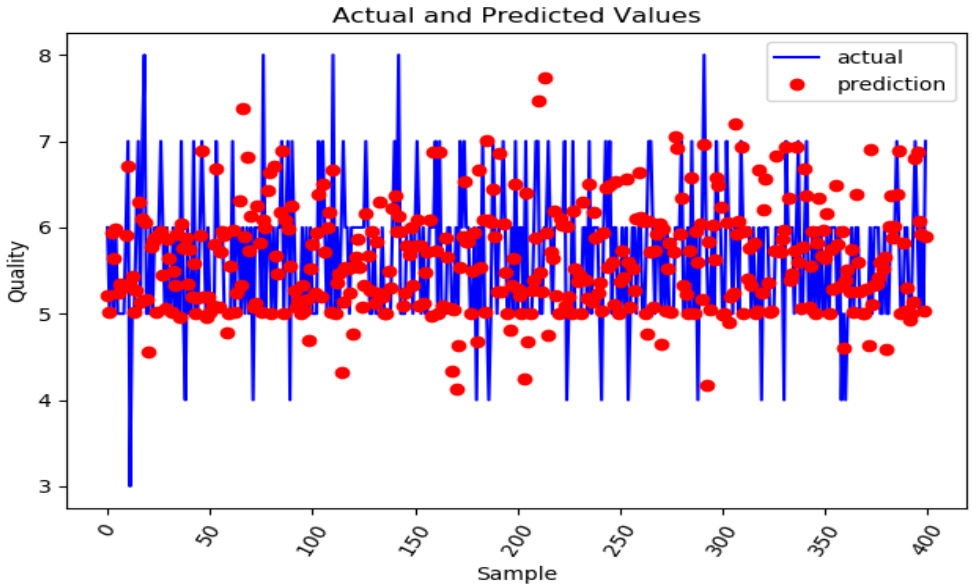
Available : https://github.com/HarnoorOberai/WineDataAnalysis/blob/master/Random-Forest/tree.png



Small Tree-



Actual VS Predicted Values-



The reports show that the model has :
Mean Absolute Error: 0.42 degrees
Prediction Accuracy: 92.38 %

8.3 k-Nearest Neighbors (kNN)

We are using this white wine quality dataset, and all of its attributes (e.g. sulfur dioxide content, pH) to find what constitutes a "good" quality wine. We used R language to conduct the analyses in this report. Understanding what makes a good wine is an interesting question because it allows us to check at "taste" differently without proper training in wine tasting, we can determine methodically how and why a wine is good using data analytics.
In this section, we have used k-nearest neighbours to classify each observation as either good or bad based on these features, while using a different number of variables, nodes, and the number of neighbours and comparing them.

8.3.1White Wine:

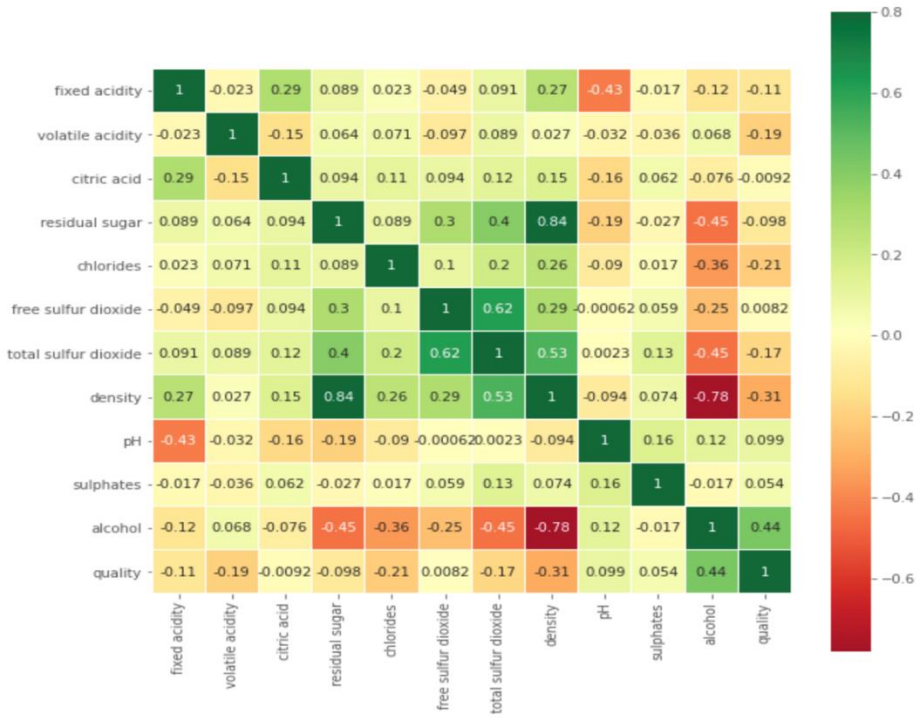
Check the missing values in the dataset.

```
[ ] df = pd.read_csv("winequality-white.csv", sep = ";")
    print(df.isnull().head()) # check for missing values (This is part of our CW req)
```

```
   fixed acidity  volatile acidity  citric acid  ...  sulphates  alcohol  quality
0          False             False        False  ...      False    False    False
1          False             False        False  ...      False    False    False
2          False             False        False  ...      False    False    False
3          False             False        False  ...      False    False    False
4          False             False        False  ...      False    False    False

[5 rows x 12 columns]
```

First of all, we can check the correlation matrix of the wine dataset, to see if any predictors are highly correlated with one another. We may have to strip them out to avoid multicollinearity, which can invalidate results.

```
[ ] corr = df.corr() #correlation dataframe
    # import seaborn as sns to show matrix
    plt.figure(figsize = (10,10))
    sns.heatmap(corr, vmax = .8, linewidths = 0.01, square = True, annot = True, cmap = 'RdYlGn', linecolor = 'white')
    plt.show()
```

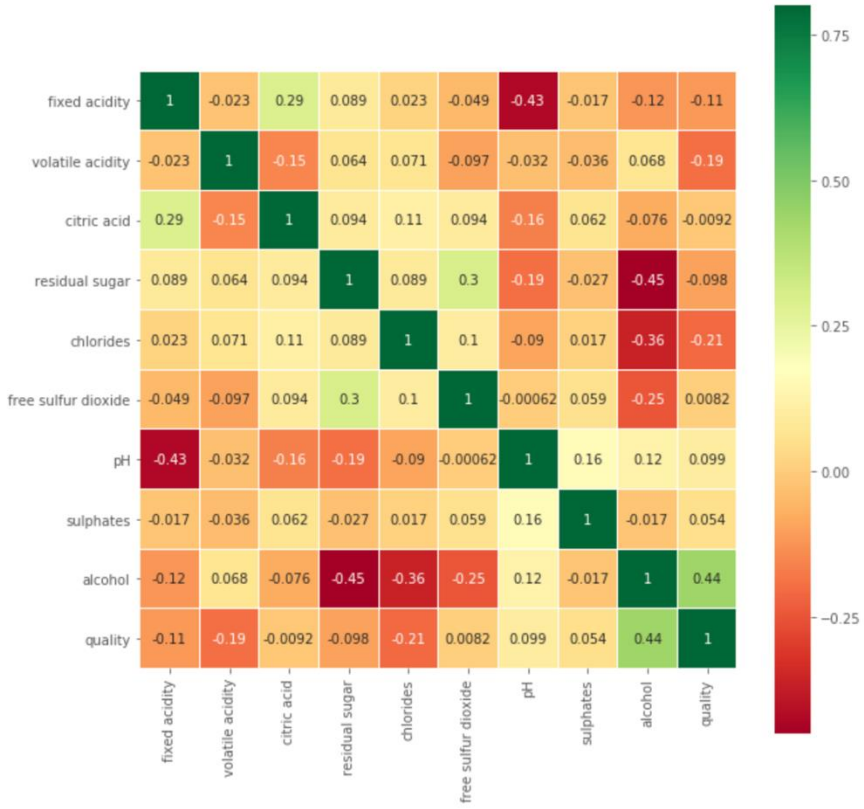


As per the above results of correlation coefficients, we can see that density is strongly connected to residual sugar(r=0.84) and alcohol (r=-0.78) and slightly correlated with total sulphur dioxide (r=0.53). Free sulphur dioxide and total sulphur dioxide are also slightly correlated with each other(r=0.62).

Apart from this, all correlations are less including quality which is the predictor.

Hence we removed the variables “density” and “total sulfur dioxide” because of density's high correlation with a few other features, and total sulfur dioxide's relationship with free sulfur dioxide.

```
# removing three predictors
df = pd.read_csv("winequality-white.csv", sep = ";")
# df.drop(["residual sugar", "density", "total sulfur dioxide"], axis = 1, inplace = True)
df.drop(["density", "total sulfur dioxide"], axis = 1, inplace = True)

corr = df.corr()
plt.figure(figsize = (10,10))
sns.heatmap(corr, vmax = .8, linewidths = 0.01, square = True, annot = True, cmap = 'RdYlGn', line
color = 'white')
plt.show()
```



As per the above-updated correlation matrix, it appears that none of the predictors have too much of correlation. Further, we are converting the “quality” into a binary variable so that we can use classification. Here 5> is good and 5< bad. This good/bad factorisation is made under a new column titled label.
We'll remove the variable “quality” afterwards, as it is as an attribute in the predictor, it will increase the skewness due to direct correlation to the label.

```
# assigning the independent  variables to X
X = df.loc[:, 'fixed acidity':'alcohol']
feature_names = X.columns.values

# converting the dependent variable from numeric to categorical
def score_to_label(x):
    if x > 5:
        return 1
    else:
        return 0

# replacing the numeric 'quality' with categorical 'label'
df.quality = df.quality.apply(score_to_label)
df.quality, class_names = pd.factorize(df.quality)
y = df.quality
```

Now we have 8 continuous predictor variables, and 1 categorical variable “quality” where 1 depicts 'good' and 0 depicts 'bad'. Further, we need to split the dataset into a training set and a test set. The first set will be used to train the classification model and then apply the algorithm to the test data set, and compute the accuracy.
We are also scaling the predictor variables (fixed acidity, volatile acidity, citric.acid, chlorides, free sulfur dioxide, pH, sulphates, alcohol) before applying any analysis.

```
# split the data intro training and test sets, test size is 30% of the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state = 100)

# scaling the numeric attributes
# from sklearn.preprocessingimport StandardScaler
scaler = StandardScaler().fit(X_train)
# scaled X
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

Finally, we apply the k-nearest neighbours classification algorithm, which is a non-parametric method. It predicts this label by plotting the test samples in the same dimensional space as the train data, then classifies it based on the "k nearest neighbour(s)", i.e. if k = 6, then the label of the 6 nearest neighbours in the train data to the test data outputs will be applied to that outputs.

```
# k-nearest neighbors
knn_clf = neighbors.KNeighborsClassifier(10, weights = "distance")
knn_clf.fit(X_train, y_train)
knn_y_pred = knn_clf.predict(X_test)

# metrics and ROC curve
knn_acc = accuracy_score(y_test, knn_y_pred)
print(knn_acc)
```

```
0.8333333333333334
```

We finally get an accuracy of 84 percent.

KNN algorithm uses Euclidean distance. This is one of the limitations because when calculating distance it assumes that features have the same effect, which is not usually true. Hence the Euclidean distance does not take into account the relationships of the features with each other, which can result in misclassification. we made sure that two features that were highly correlated with each other were dropped, and also applied to scale the remaining numeric features to minimize the shortcoming.

8.3.2 Red Wine:

Similarly using Red Wine dataset, we get the accuracy of around 78 %.

The complete code is given here:
https://github.com/HarnoorOberai/WineDataAnalysis/blob/master/kNN/kNN_Red.ipynb

```
# k-nearest neighbors
knn_clf = neighbors.KNeighborsClassifier(10, weights = "distance")
knn_clf.fit(X_train, y_train)
knn_y_pred = knn_clf.predict(X_test)

# metrics and ROC curve
knn_acc = accuracy_score(y_test, knn_y_pred)
print(knn_acc)
```

```
0.7729166666666667
```

9.Observations and Conclusion

	White	Red
Samples	1559	4898
Columns	12	12
Missing Values	None	None
Duplicates	937	937
Needs to be dropped	No	No
Unique Quality	7	6

- **Which wine is associated with higher quality?**
  White
- **Do wines with higher alcoholic content receive better ratings?**
  YES
- **Do sweeter wines (high residual sugar)receive better ratings?**
  YES

10.Bayes Questions

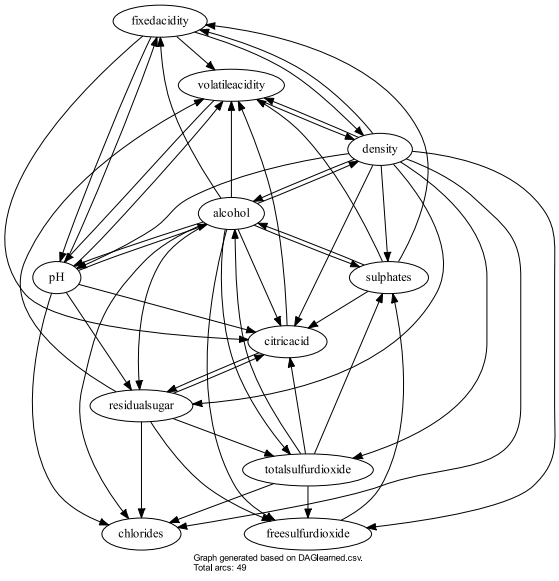
Answer 1:

The steps used to generate a knowledge-based causal graph for white and red wine are understanding the chemical properties of wine which included following:

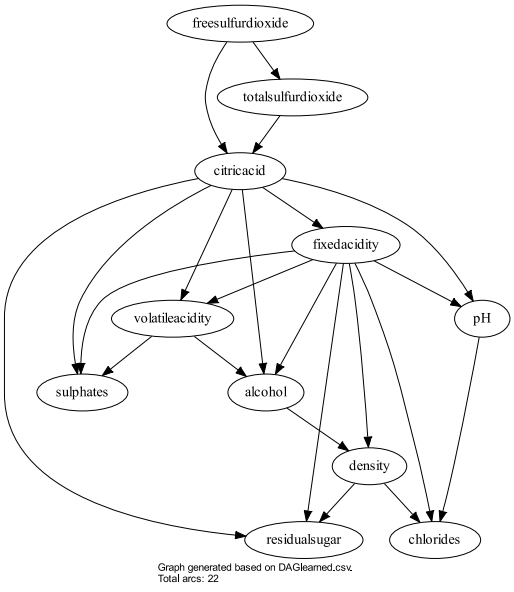
- fixed acidity
- volatile acidity citric acid
- residual sugar
- chlorides
- free sulfur dioxide
- total sulfur dioxide
- density
- pH
- sulphates
- alcohol.

Most of the chemical properties where continuous so, it was decided that we will convert these properties into discrete values of 0,1 and 2. Since wine has many properties we took the help of some wine connoisseur and google scholar pages to understand which factor might be dependent on each other to include into DAGtrue.cv file. We resolved our disagreement by continuous trial and error and checking the precision of our graph.

White Wine:



Red Wine:



Answer 2:

Extended Maximum Spanning Graph is actually an undirected graph visualized in phase1 using MeanMax-Marginal-Discrepancy(MMD) association scores of the established connections in DAGtrue.csv. The marginal probability discrepancy is shown by MMD score between posterior and prior distributions varying from 0 to 1. Stronger the dependency, higher is the score.

Once the required edges are done modified and less noteworthy features have been pruned, the generated graph is envisaged by the Extended Maximum Spanning Graph. SaiyanH utilizes Mean Max- Marginal-Discrepancy count to carry out restraint based learning within conditional independence checks across all the given nodes of the second phase graph and it also classifies them into independence, conditional dependence or unimportant.

Answer 3:

The Phase 3 graph is the output of the score-based learning process which is built on a second phase graph until the path which increases the Bayesian Information Criterion count function. SaiyanH uses the concept of the hill-climb to find the neighbor graphs where an edge is taken out, included/reversed, and go along the path of improving BIC score. sub-heuristic algorithm search is used as a chance to avoid any local maxima after the highest BIC count has retrieved and is recursively done until the algorithm cannot find a graph that increases Bayesian Information Criterion total. The graph is often the final output enabling full propagation of proof by principle.
The last phase includes a find procedure that locates adjacent graphs and a scoring condition to estimate every graph. The product of the second phase acts as the initial graph for searching in the third phase.

Answer 4:

White wine				
CSV	conditionalDep	conditionalInsignificance	marginalDep	conditionalIndep
Score	15	478	55	2

Red wine				
CSV	conditionalDep	conditionalInsignificance	marginalDep	conditionalIndep
Score	71	423	55	1

For both red and wine data set, we found that there is are an additional number of scores for conditionalInsignificance.csv than marginalDep.csv. This output is retrieved due to the conditional independence checks on triple sets (in the second phase) and not- on the group of nodes in the first phase. Therefore, there are bigger combinations of feasible features. The additional number of features are classified as conditionally irrelevant as we follow the correct classification criteria of dependence/ independence.
Additional conditional dependent features than conditionally independent features are seen for both the datasets as we choose only features that we suspect to have a causal connection in our knowledge-based graph.

Answer 5:

	White Wine	**Red Wine**	**Average case study**
F1	0.577	0.39	0.55
SHD	28.500	46	69
BSF	0.418	0.66	0.46

Both wines performed better in SHD-Score. This is an indication that it requires fewer edge insertions, arc reversal and deletion to convert learned graph into true graph. We did not expect BSF to be higher since we had 11 chemical properties and to include different arcs was challenging. Higher BSF score suggests we made a better accurate graph.

Answer 6:
We have observed structural learning of 1 second for both red and white wine which is consistent with the result shown in table 2. This is because we have a similar number of sample size, nodes and parameter.

TABLE 2
TIME COMPLEXITY OF SAIYANH BASED ON A SINGLE-CORE (TURBO BOOST) SPEED OF 4.7GHZ.

Nodes	True edges	Max in-degree	# free param	Sample size	Runtime (sec)
37	46	4	509	0.1k	1
8	8	2	18	0.1k	1
8	8	2	18	1k	1
8	8	2	18	10k	1
8	8	2	18	100k	1
27	31	3	3,056	1k	1
9	15	2	1,049	0.1k	1
9	15	2	1,049	1k	1
9	15	2	1,049	10k	1

Answer 7:

White Wine:

Step 3	Step 4
BIC/MDL score -109539.972	BIC/MDL score -67907.733

Red Wine:

Step 3	Step 4
BIC/MDL score -152523.756	BIC/MDL score -20630.979

BIC criteria are used to select best from a finite set of models. BIC/MDL score higher for both the wines at step 4 in comparison with step 3. Step 3 represents the casual model (knowledge-based graph) made by us with intuition and little knowledge before any constraint-based learning. Step 4 is the model made after Hill Climbing has completed and edges have been modified to maximize the BIC score. Hence Step 4 had a better score and agreed with expectations.

Answer 8:

White Wine:

Step 3	Step 4
# of free parameters 10314	# of free parameters 302

Red Wine:

Step 3	Step 4
# of free parameters 27270	# of free parameters 302

Variables which are not predicted correctly/precisely or constrained by the model and has to be estimated are represented by free parameters. A lower number is expected in Step 3 because our model is unpruned and has not been optimized using BIC criteria, whereas Step 4 the model has been pruned and optimized, thus requires less number of parameters.

11.Business Applications

One of the major future application of this project can be in business segment of alcohol and beverages. The model can used to predict a universal alcohol quality metric which can be used everywhere as a standard. This will help the customer to understand the rating associated with any brand of alcohol displayed on the container.

12.Concluding Remarks

After analysing the data, we came up with the below inferences:

1. When the alcohol percentage increases, the quality usually increases.
2. Some different combinations improve wine quality such as more percentage of alcohol and low concentration of the volatile acidity.
3. Free sulphur dioxide and total sulphur dioxide are correlated to a positive extent and rich quality wines are having less concentration of free & total sulphur dioxide.
4. In our dataset, most of the attributes are not strongly correlated with quality; the major 4 attributes, which has the highest correlation coefficient with quality, describes around 34% of the variance in quality, which is certainly not good enough to predict high-quality wines with fine precision.
5. The more occurring quality levels of red wines are mid-range i.e. 5 and 6.
6. There is more amount of wines with less alcohol percentage in our data set.
7. Wine with the greatest alcohol percentage has a quality level of around 7, where wine with the lowest alcohol percentage has a quality level of 5.

13.References

[1] P. Cortez, A. Cerderia, F. Almeida, T. Matos, and J. Reis, "Modelling wine preferences by data mining from
physicochemical properties," In Decision Support Systems,Elsevier, 47 (4): 547-553. ISSN: 0167-9236.

[4]A. Asuncion, and D. Newman (2007), UCI MachineLearning Repository, University of California, Irvine,[Online].Available: http://www.ics.uci.edu/~mlearn/MLRepository.html

[5]S. Kallithraka, IS. Arvanitoyannis, P. Kefalas, A. El-Zajouli, E. Soufleros, and E. Psarra, "Instrumental and sensory analysis of Greek wines; implementation ofprincipal component analysis (PCA) for classification according to geographical origin," Food Chemistry, 73(4):501-514, 2001.

[6]N. H. Beltran, M. A. Duarte - MErmound, V. A. S. Vicencio, S. A. Salah, and M. A. Bustos, "Chilean wine classification using volatile organic compounds dataobtained with a fast GC analyzer,"
Instrum. Measurement, IEEE Trans., 57: 2421-2436, 2008.

[7] M. Yeo, T. Fletcher, and J. Shawe-Taylor, "Machine Learning in Fine Wine Price Prediction," Journal of Wine Economics, Vol. 10, No. 2, pp.151-172, 2015.

[8] J. Ribeiro, J. Neves, J. Sanchez, M. Delgado, J. Machado, and P. Novais, "Wine Vinification Prediction using Data Mining Tools.," in Proc. Computing and Computational Intelligence, Tbilisi, the Republic of Georgia, pp. 78-85. June 2009.

[9] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, J. Reis, "Modeling Wine Preferences by Data Mining from Physicochemical Properties," Department of Information Systems/R&D Centre Algoritmi, University of Minho, 4800-058 Guimarães, Portugal, Viticulture Commission of the Vinho Verde Region (CVRVV), 4050- 501 Porto, Portugal.

[10] S. Shanmuganathan, P. Sallis, and A. Narayanan, "Data Mining Techniques for Modeling Seasonal Climate Effects on Grapevine Yield and Wine Quality," in Proc. IEEE International Conference on Computational Intelligence Communication Systems and Networks, pp. 82-89, July 2010.

[11] B. Chen, C. Rhodes, A. Crawford, and L. Hambuchen, "Wineinformatics: Applying Data Mining on Sensory Wine Reviews Processed by the Computational Wine Wheel," in Proc. IEEE International Conference on Data Mining Workshop, pp. 142-149, Dec. 2014.

14.Appendices:

Our well-documented code has been uploaded to Git where you can access our notebook, graphs, python files and Bayesian graphs and stats which can be found at:

- Root dir:
  - https://github.com/HarnoorOberai/WineDataAnalysis
- Models:
  - https://github.com/HarnoorOberai/WineDataAnalysis/tree/master/Random-Forest
  - https://github.com/HarnoorOberai/WineDataAnalysis/tree/master/SVC
  - https://github.com/HarnoorOberai/WineDataAnalysis/tree/master/kNN
- Bayesian Analysis
  - https://github.com/HarnoorOberai/WineDataAnalysis/tree/master/BayesianAnalysis