**Red wine quality prediction**

**Introduction:**

The wine industry has seen an exponential growth as drinking wine has become a commonplace in social gatherings. Nowadays, certifications are being used by market players as a benchmark for finest quality wine which is a time intensive process and always requires a human touch as tasting can only be done by us. This makes the process expensive, and straining. This can also become a major factor in driving up the cost of the wine. But the price can be based up on variety of factors, which can be rather abstract as the human side of tasting can sometimes be unscientific and based on emotions the wine provokes. But the quality assessment is usually based on physicochemical tests, which are laboratory-based and consider factors like acidity, pH level, sugar, and other chemical properties. This process can become more meticulous if human tasting can become analogous to be actually based on wine’s chemical properties.

**Problem Statement:**

My analysis will use Red Wine Quality Data Set, available on the UCI machine learning repository (<https://archive.ics.uci.edu/ml/datasets/wine+quality>) and also available in the GitHub repo(<https://github.com/dsrscientist/DSData/blob/master/winequality-red.csv>).

The dataset contains a total of 12 parameters, which were recorded for 1,599 observations. The data is related to red and white variants of the Portuguese "Vinho Verde" wine. Due to privacy and logistic issues, only physicochemical (inputs) and sensory (the output) variables are available (e.g., there is no data about grape types, wine brand, wine selling price, etc.). This project’s objective is to classify the wines in the good and the poor category using classification models and figure out physiochemical features are great indicators of a good wine.The classes are ordered and not balanced (e.g., there are many more normal wines than excellent or poor ones). Also, we are not sure if all input variables are relevant. We have to predict whether each wine belong to the “good” category (quality > or = 7).

With this dataset, we will create different classification models to determine how dissimilar independent features predict our target variable, quality. Understanding how each feature will influence the red wine quality will help businesses in the wine industry to enhance their evaluation metrics to improve their production, distribution, marketing and pricing tactic which can help them gain an edge over their competitors.

**Feature Information:**

Input variables (based on physicochemical tests):

1. *Alcohol*: the amount of alcohol in wine

2. *Volatile acidity*: are high acetic acid in wine which leads to an unpleasant vinegar taste

3. *Sulphates*: a wine additive that contributes to SO2 levels and acts as an antimicrobial and antioxidant

4. *Citric Acid*: acts as a preservative to increase acidity (small quantities add freshness and flavor to wines)

5. *Total Sulfur Dioxide*: is the amount of free + bound forms of SO2

6. *Density*: sweeter wines have a higher density

7. *Chlorides*: the amount of salt in the wine

8. *Fixed acidity*: are non-volatile acids that do not evaporate readily

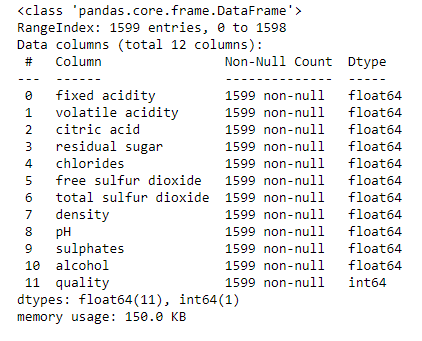
9. *pH*: the level of acidity *Free Sulfur Dioxide*: it prevents microbial

10. *Free Sulfur Dioxide*: it prevents microbial growth and the oxidation of wine

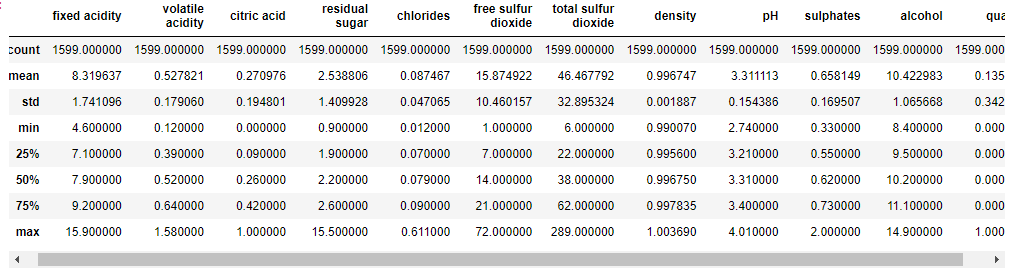
11. *Residual sugar*: is the amount of sugar remaining after fermentation stops. The key is to have a perfect balance between — sweetness and sourness (wines > 45g/ltrs are sweet)

Output variable (based on sensory data):

12. *quality*: score between 0 and 10

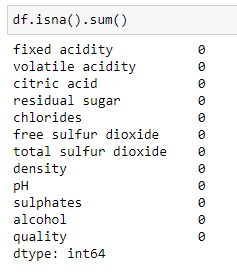
**Data Analysis:**

First step in ant machine learning project is to prepare and clean the data for the EDA process. I loaded the dataset and saved it as a pandas dataframe. Then, I proceeded to the next step where I checked different characteristics of the dataset like data types of the features and the statistical measures of the numerical features using *df.describe()*. Here, we can also use this command to include categorical features using *df.describe(include=”all”)*.



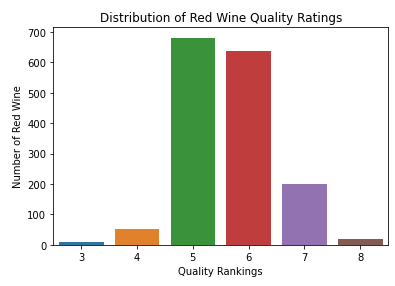
Here, we saw that all the features had the datatype as “float64” except the target columns which belonged to “int64”, indicating the fact that “quality” is a whole integer instead of decimal number.

Next step, would be to check if there are any null values present in dataset. If null values then we can go ahead and impute them with mean for numerical variables and with mode for categorical variables, only if the present quantity is small. If the null values are present in each feature and are a lot, then we can leave them and impute them at a later stage, possibly after Exploratory Data Analysis.

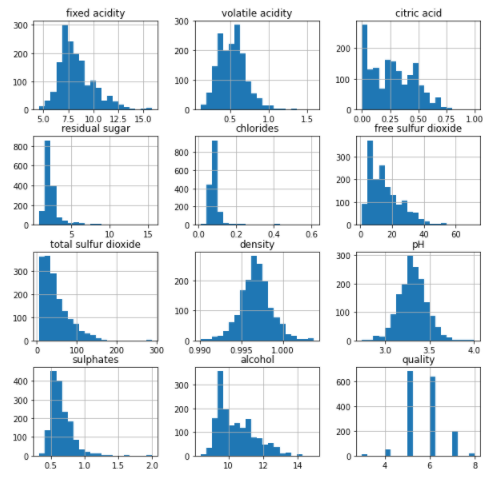


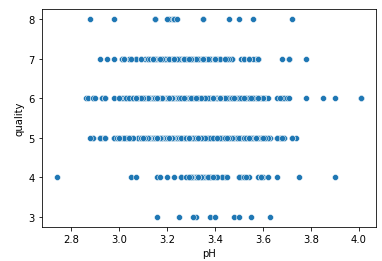
We found that there were no Null values present in the dataset.

**EDA concluding remark:**

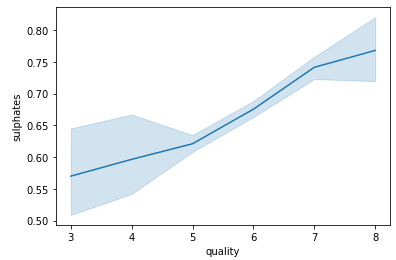
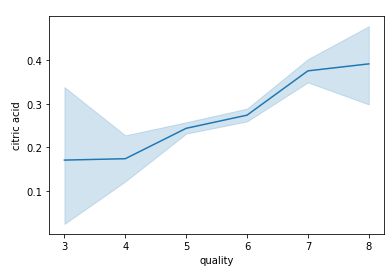
It’s always advisable to proceed with just checking the distribution of all the features including the target column to understand how the data is distributed and the unique values present in each of them. Here, we can infer from the graph below, red wines with 5 and 6 quality bands are present in higher numbers compared to other bands. 7+ quality band wines are second in number which indicates that there is a class imbalance as 7+ wines have to be considered as good according to the problem statement.

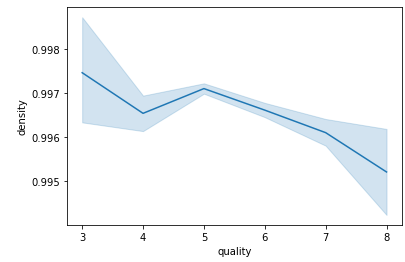
We can then simply call the hist method to plot histograms of each of the numerical feature, which indicates that some features are skewed and some follow gaussian distribution.



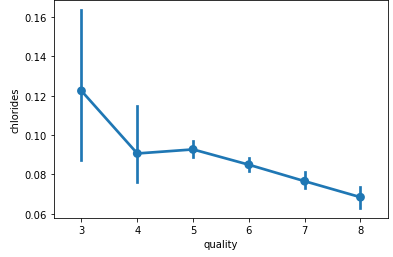
Then I used scatter plot to check the relationship between the “pH” and “quality” of the wine and found that good quality wines usually have a pH between 2.9 to 3.6 implying that 7+ quality wines are highly acidic in nature, also faintly informing us that they are high in alcohol content.

After this, I used lineplot to plot a graph between “quality” and “citric acid” and found out that the most of the good wines have a citric acid range in 0.26 to 0.49 with the average being just a little below 0.4. and the second graph between “sulphates” and “quality” indicates that higher quality wines have an average 0.76 meaning they are highly antimicrobial and have high antioxidant properties.

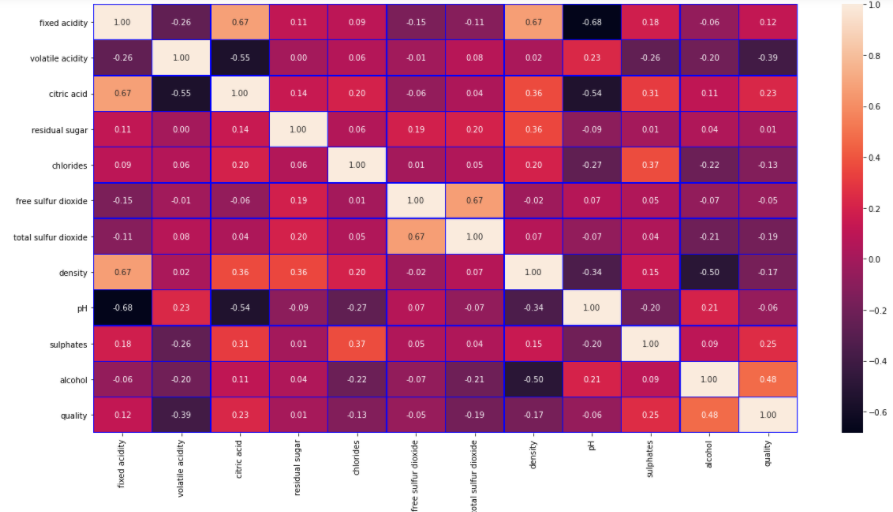


In the next lineplot, I found that higher quality wines have comparatively lower density than mediocre wines indicating higher sourness as higher density wines tend to be sweeter as mentioned in the details before.

I plotted a pointplot between “quality” & “chlorides” and saw a downward trend signifying that higher quality wines have lower salt than mediocre wines.



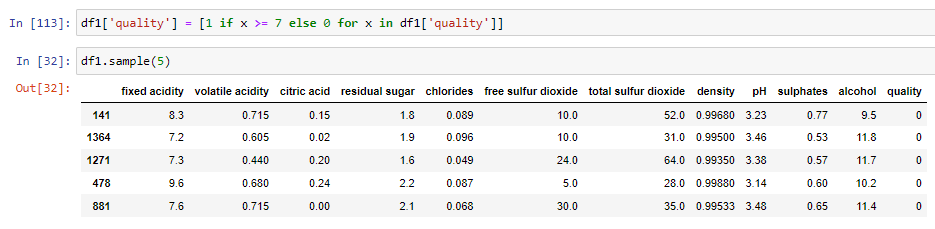
After that, I used a Pairplot which basically is a bivariate analysis plot. It plots each feature with each other and the target column. This makes it easier to see the bigger picture and not having to code for each relation.



Then I plotted the correlation heatmap and also used heatmap to plot the statistics of the dataset. The conclusion is as follows:

* Quality has a (+) positive relationship between alcohol
* Quality has a (-) negative weak relationship between volitile\_acidicity
* Quality has almost no relationship between residual\_sugar, free\_sulfur\_dioxide, and pH (correlation ~ 0)
* Alcohol has a (+) positive relationship between quality and weakly pH
* Alcohol has a (-) negative relationship between density
* Alcohol has almost no relationship between fixed\_acidicity, residual\_sugar, free\_sulfur\_dioxide, sulphates.
* Volitile\_acidicity has a weak (+) positive relationship between pH.
* Volitile\_acidicity has a strong (-) negative relationship between citric\_acid
* Volitile\_acidicity has weak (-) negative relationship between fixed\_acidicity and sulphates
* Volitile\_acidicity has almost no relationship between residual\_sugar, chlorides, free\_sulfur\_dioxide, total\_sulfur\_dioxide, density
* Density has (+) positive relationship between fixed\_acidicity
* Density has (-) relationship between density
* Density has almost no relationship between volitile\_acidicity, free\_sulfur\_dioxide, total\_sulfur\_dioxide.
* Citric\_acid has (+) positive relationship between fixed\_acidicity
* Citric\_acid has (-) negative relationship between volitile\_acidicity, pH
* Citric\_acid has almost no relationship between residual\_sugar, free\_sulfur\_dioxide, total\_sulfur\_dioxide

After this, we can rewrite the target column and convert it from multiclass classification to a binary classification problem by categorizing wines with quality 7 or above as good wine (1) and other quality wines as poor (0).

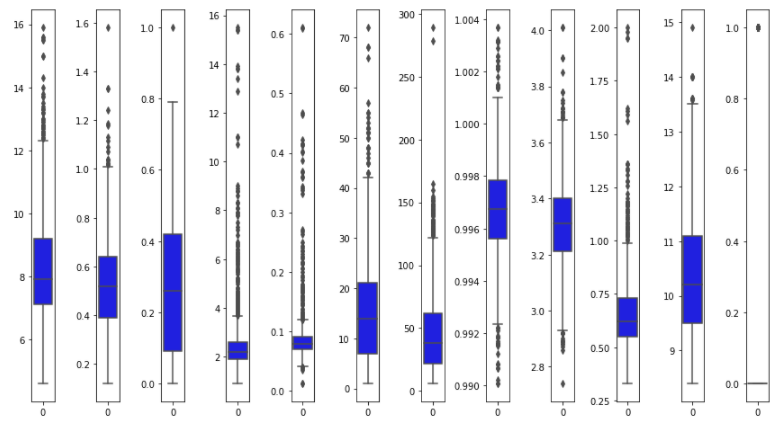


Also, heatmap shows which of the features are most important. Importance is in order as follows (by magnitude):

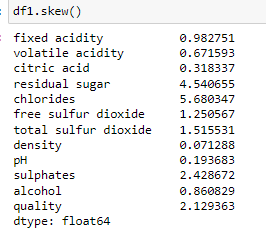
1. Alcohol
2. Volatile acidity
3. Sulphates
4. Citric acid
5. Total sulfur dioxide
6. Density
7. chlorides
8. Fixed acidity
9. pH
10. free sulfur dioxide
11. Residual sugar

**Pre-processing Pipeline:**

First step in preprocessing is to check for outliers in the data. This can be best done using boxplots. For this, we will simply use *sns.boxplot* for each column and check which of them have outliers.



From the plot above, we can see most of the columns have outliers except quality, citric acid and alcohol which contain relatively less outliers.

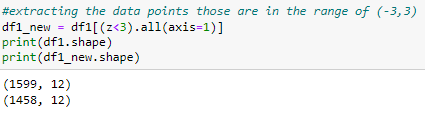
Secondly, we check the skewness of each numerical column, in this case our hole dataset is numerical, and if the skewness is found to be outside the range of +/- 0.5, we’ll use some kind of transformation like log1p or sqrt.

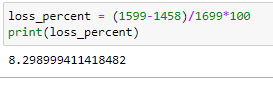
We can infer from the output above, so columns which needs to be transformed are:

1. fixed acidity
2. volatile acidity
3. residual sugar
4. chlorides
5. free sulphur dioxide
6. total sulphur dioxide
7. sulphates
8. alcohol

Now we can proceed to remove outliers and skewness from the data, but a very important thing to keep in mind, it is strongly advised to not perform any transformation to remove skewness and scaling (done to make features more or less in the same range) in the target column as it can change the data which is not the right approach, as model will learn the wrong target values.

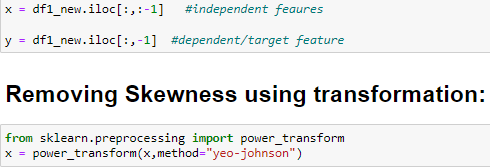
Now we can use zscore method to remove outliers from the data and then check the data loss percentage.





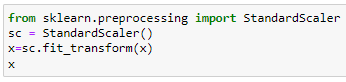
We can see that we have lost around 8.3% of the data.

Now we can split the data into Input (x) and Target (y) features. After that we can simply remove the skewness using power transform in x and then proceed to scale the input features using Standard Scaler.



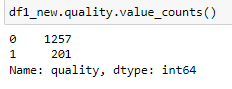
We can see that the skewness has been greatly reduce for every column except alcohol and residual sugar.

After Scaling:



*Handling the class imbalance:*

As we know the target variable is hugely imbalanced and to create an accurate model, we need to balance the classes using one of the resampling techniques.

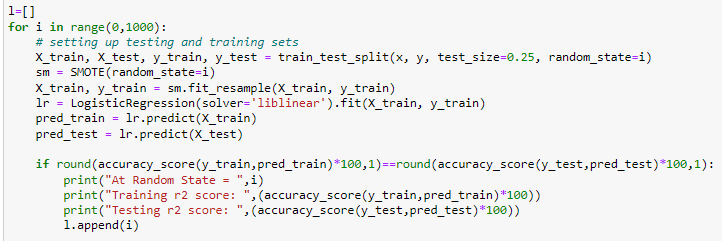


Here, SMOTE (Synthetic Minority Oversampling Technique) will be used to up sample the minority class so the dataset can be evenly divided into training and testing datasets.

It is also advisable to split the dataset into testing and training set before using oversampling techniques.

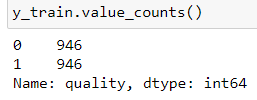
Oversampling before splitting the data can allow the exact same observations to be present in both the test and train sets. This can allow our model to simply memorize specific data points and cause overfitting and poor generalization to the test data.

But first we need to figure out the best random state for our models later. For this we just use a simple code and find the random state where training accuracy is almost equal to testing accuracy on a resampled X\_train and y\_train using SMOTE.



I found at the random\_state in the list above, we have found the best testing and training accuracy for logistic regression. At random state = 653. accuracy touches 83.5%.

Now we can use this random state to train\_test\_split the x and y and to resample X\_train and y\_train to be used for models in the next step. We can see the minority class have been upsampled and both classes have equal samples in the training target feature.



**Building Machine Learning Models:**

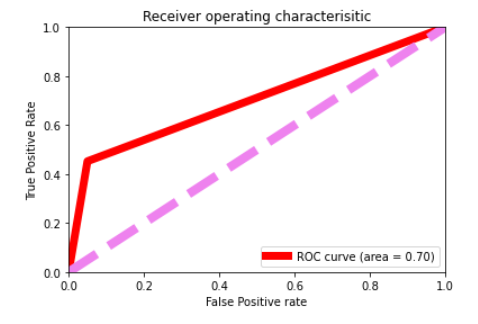
We’ll use gridsearchCV for hyperparameter tuning and get the best possible accuracy for our models.

1. **Logistic Regression:**

With GridSearchCV, I was able to get best hyperparameters: {'C': 1, 'penalty': 'l2'}.

Then, I trained the model and found the accuracy to be close to 84% on the testing data. To reaffirm this, I used K-fold cross validation and found that at k=2, the cross\_val score was the closest to testing accuracy of 83.56%. Then I also, auc\_roc score and curve area which I found to be 0.70.

In general, an AUC of 0.5 suggests no discrimination , 0.7 to 0.8 is considered acceptable, 0.8 to 0.9 is considered excellent, and more than 0.9 is considered outstanding.

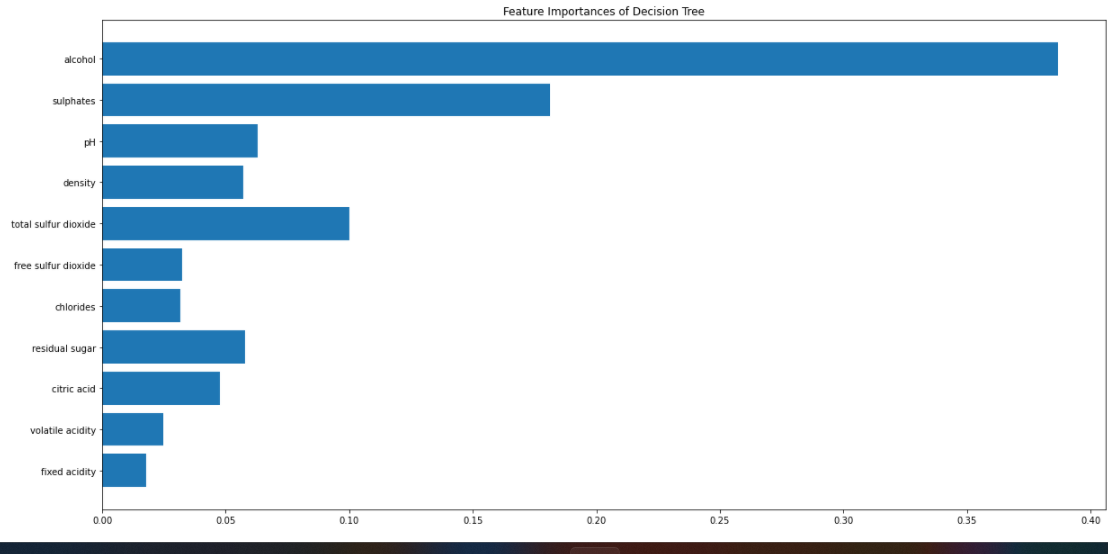


1. **Decision Tree Classifier:**

Another point to remember: Best\_score (an attribute of gridsearchCV) is the mean of the cross\_val\_score which is possible with these quantities on a limited dataset. We'll still have to perform cross validation separately to check how cross val impacts our accuracy\_score.

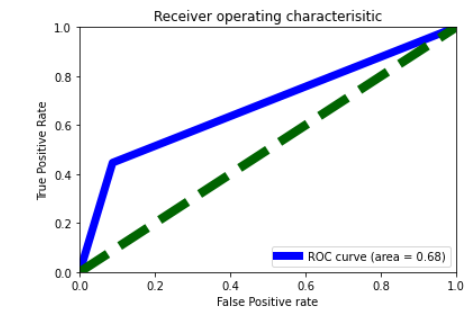
With GridSearchCV, I found :

Best Parameters for Decision Tree: {'criterion':'entropy', 'max\_depth': 14, 'min\_samples\_leaf': 7} Best Score for Decision Tree: 0.8834926248719353



“Alcohol” and “sulphates” were the most important features which had the maximum impact on the model.

At k= 6, Cross Val score: 83.54 and accuracy score is: 84.11



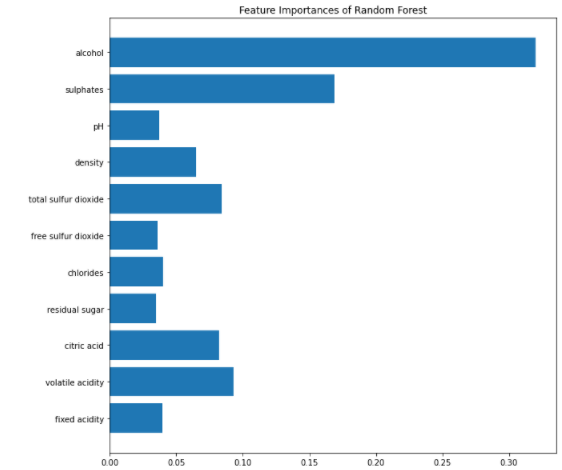
The ROC curve area was found to be 0.68, indicating that even after getting a higher accuracy than Log Reg., ROC curve area lags behind and this might not be a better model than the above model.

1. **Random Forest Classifier:**

With GridSearchCV:

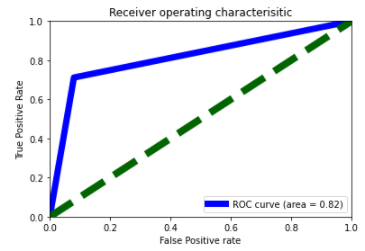
Best Parameters for Random Forest: {'bootstrap': False, 'max\_depth': 9, 'min\_samples\_leaf': 5}

Best Score for Random Forest: 0.9269139545001615



“Alcohol” and “sulphates” were the most important features which had the maximum impact on the model, similar to Decision Tree which makes sense as Random Forest is an ensemble technique where it creates a huge number of decision trees and then averages them to reach a final score.

At k = 9, Cross val score : 88.68 & accuracy score is : 89.04. This makes sense as ensemble techniques usually have better accuracy than their single counterparts.



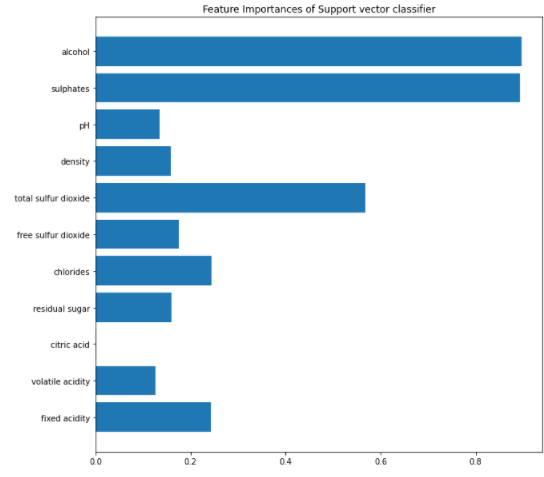
The ROC curve area is found to be 0.82 which makes this model an excellent one.

1. **Support Vector Classifier:**

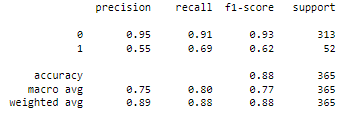
With GridSearchCV:

Best Parameters for SVM: {‘C’: 1000.0, ‘kernel’: ‘linear’}

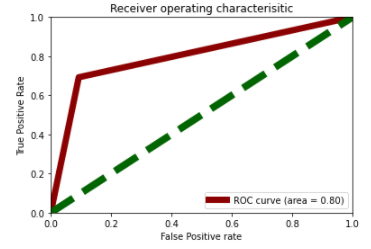
Best Score for SVM: 0.9555162589645348



Alcohol” and “sulphates” were the most important features which had the maximum and equal impact on the model.



At k= 7, Cross Val score: 86.21 and Accuracy score is: 81.37



The ROC curve area is 0.80 for SVC which is indicates it is a good model and will be useful at any test condition.

**Conclusion:**

We can conclude from these models that Random Forest Classifier performs the best on the testing data with an accuracy of 89%. Now, we can save this model so it can be deployed on a server using the pickle library and dump the model in a local file.

