**WineQuality Prediction**

The below data used for predicting the quality of wine based on the parameters or ingredients portion in it. The prediction model can be made by the machine learning techniques in my future article.



**Content:**

1. Problem Definition
2. Data Pre-processing
3. EDA (Exploratory Data analysis)
4. Cleaning the data
5. Model Deployment
6. Hyper Parameter Tuning
7. Creating Pipeline & Saving Model
8. Concluding Remarks

# Problem Definition:

The data-set is related to red and white variants of the Portuguese **“Vinho Verde”** wine. Here we will predict the quality of wine on the basis of giving features. We use the wine quality dataset from Kaggle. This dataset has the fundamental features which are responsible for affecting the quality of the wine. By the use of several Machine learning models, we will predict the quality of the wine. Here we will only deal with the white type wine quality, we use classification techniques to check further the quality of the wine i.e., is it good or bed.

**Understanding columns names:**

1. **fixed acidity**  
   most acids involved with wine or fixed or non-volatile (do not evaporate readily)
2. **volatile acidity**  
   the amount of acetic acid in wine, which at too high of levels can lead to an unpleasant, vinegar taste
3. **citric acid**  
   found in small quantities, citric acid can add ‘freshness’ and flavour to wines
4. **residual sugar**  
   the amount of sugar remaining after fermentation stops, it’s rare to find wines with less than 1 gram/liter and wines with greater than 45 grams/liter are considered sweet
5. **chlorides** the amount of salt in the wine
6. **free sulfur dioxide**  
   the free form of SO2 exists in equilibrium between molecular SO2 (as a dissolved gas) and bisulfited ion; it prevents microbial growth and the oxidation of wine
7. **total sulfur dioxide**  
   amount of free and bound forms of S02; in low concentrations, SO2 is mostly undetectable in wine, but at free SO2 concentrations over 50 ppm, SO2 becomes evident in the nose and taste of wine
8. **density**  
   the density of water is close to that of water depending on the percent alcohol and sugar con-tent
9. **pH**  
   describes how acidic or basic a wine is on a scale from 0 (very acidic) to 14 (very basic); most wines are between 3–4 on the pH scale
10. **sulphates**  
    a wine additive which can contribute to sulfur dioxide gas (S02) levels, which acts as an antimicrobial and antioxidant
11. **alcohol**  
    the percent alcohol content of the wine
12. **quality**  
    output variable (based on sensory data, score between 0 and 10) that we further transfer like change the quality that in range of 0 to 6 as poor (0) and quality that is or above 7 as good (1).

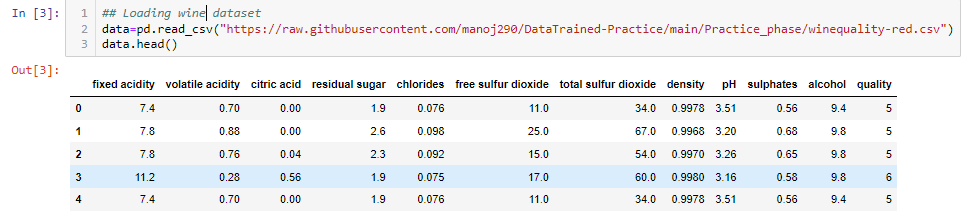
# ****Data Pre-processing****

**STEP1:** The first thing first, we need to import all the libraries that will support us to do the EDA on our data modelling and Evaluation.

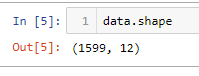
Here, I have imported:



**STEP 2**: Loading the data with python pandas library pd.read\_csv.

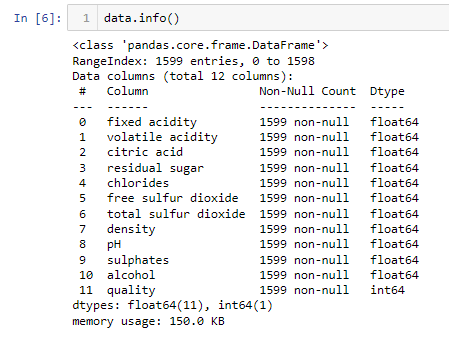
**

**STEP 3**: Checking Shape of the dataset.

**

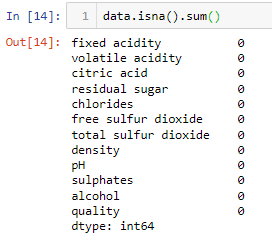
Data has 1599 no. of records and 12 no. of variables.

**STEP 4**: Checking information of the dataset.



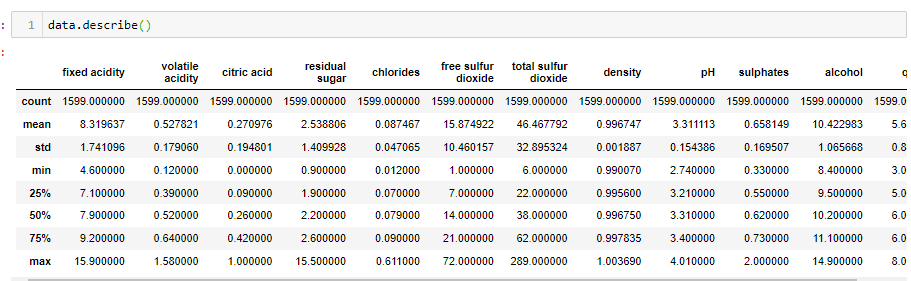
Only Quality columns contains the integer type value that is our label & and all other column are features that are of Decimal type.

**STEP 5**: Checking null values for variables of the dataset.



Here we can see that there are no null values in the data.

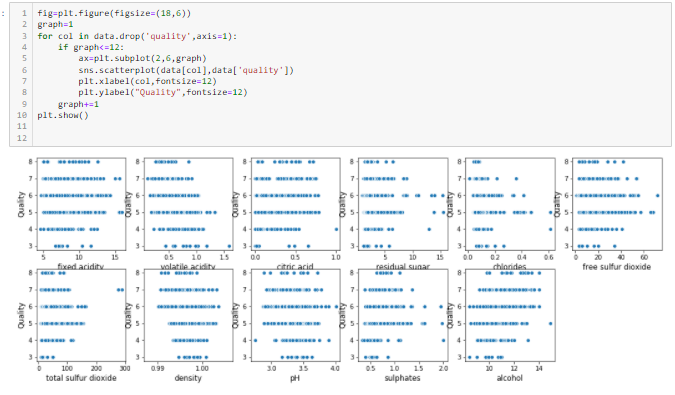
**STEP 6**: Checking Statistical description for variables of the dataset.



The max Values for variables chlorides, free sulfur dioxide and total sulfur dioxide are seeming to be high enough for wine quality.

Perform relation analysis by graphical approach

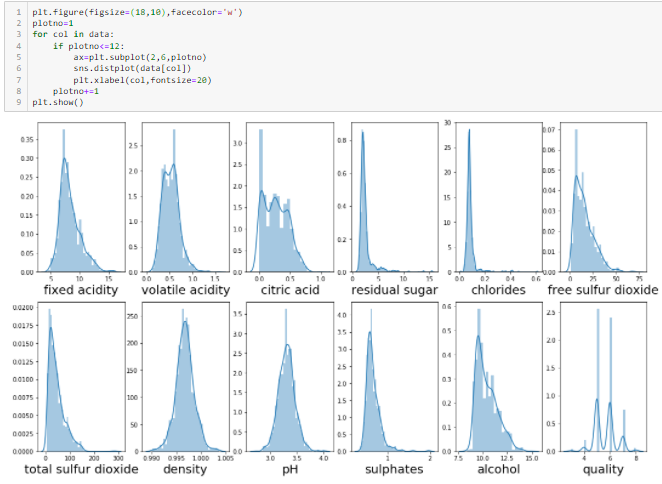
**STEP 7**: Doing EDA to know relation of variables with the labels and more to know the dataset.



Observations:

* Here we can see that fixed acidity may have some outliers and for range fixed acidity > 14 is continuous data.
* Here we can see that the data is continuous within range 0.2 to 1.1. And, it seem to have some outliers too in it.
* Here we can see that the citric acid column is continuous within range 0.0 to 0.8. And it seems to have some outliers too in it.
* Here we can see that the residual sugar column is continuous within range 2 to 9. And it seems to have some outliers too in it.
* Here we can see that the chlorides column is continuous within range 0.0 to 0.28. And it seems to have some outliers too in it.
* Here we can see that the free SO2 column is continuous within range 0 to 50. And it seems to have some outliers too in it.
* Here we can see that the total SO2 column is continuous within range 0 to 170. And it seems to have some outliers too in it for quality class 1.
* Here we can see that the density column is continuous within range 0.990 to 1.005 for both classes of quality column.
* Here we can see that the pH column is continuous within range 2.8 to 3.8. And it seems to have some outliers too in it.
* Here the data is continuous and have outliers too.
* Here the data is continuous for range 8 to 14.

### **Checking Data Distribution**



Observations:

Graph1:

In this the 'fixed acidity' column that is continuous is seems to be skewed.

Graph2:

In this the 'volatile acidity' column i.e., continuous column is not normally distributed.

Graph3:

In this the 'citric acid' column it seems to be that data is normally distributed.

Graph4:

In this the 'residual sugar' column the data is highly skewed.

Graph5:

In this the 'chlorides' column the data is highly skewed.

Graph6:

In this column i.e. 'free sulfur dioxide' is not Normally Distributed or skewed.

Graph7:

In this column i.e. 'total sulfur dioxide' is not Normally Distributed or skewed.

Graph8:

In this column i.e. density is normally distributed

Graph9:

In this column i.e. 'pH' is normally distributed

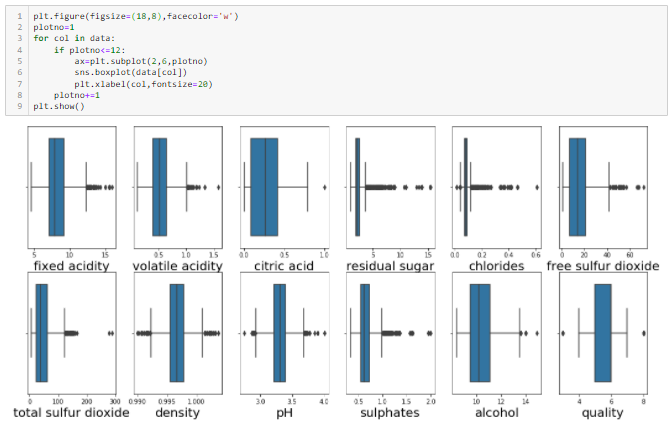
Graph 10:

In this column i.e. 'sulphates' is not normally distributed

Graph 11:

In this column i.e., 'alcohol' is not normally distributed

### **Checking for Outliers**



Observations:

##### *Graph1:*

In this the 'fixed acidity' column it seems there are some outliers seems like that are greater than 13.

##### *Graph2:*

In this the 'volatile acidity' column it seems there are some outliers seems like that are greater 1.0.

##### *Graph3:*

In this the 'citric acid' column it seems there are some outliers seems like for value 1.0.

##### *Graph4:*

In this the 'residual sugar' column it seems there are some outliers seems like that are greater 4.

##### *Graph5:*

In this 'chlorides' column it seems there are some outliers seems like that are greater 0.12 & smaller then 0.05.

##### *Graph6:*

In this column i.e., 'free sulfur dioxide' it seems there are some outliers seems like that are greater 42.

##### *Graph7:*

In this column i.e., 'total sulfur dioxide' it seems there are some outliers seems like that are greater 120.

##### *Graph8:*

In this column i.e., density it seems there are some outliers seems like that are greater 1.001 & smaller then 0.992.

##### *Graph9:*

In this column i.e., 'pH' it seems there are some outliers seems like that are greater 3.7 & smaller then 2.9.

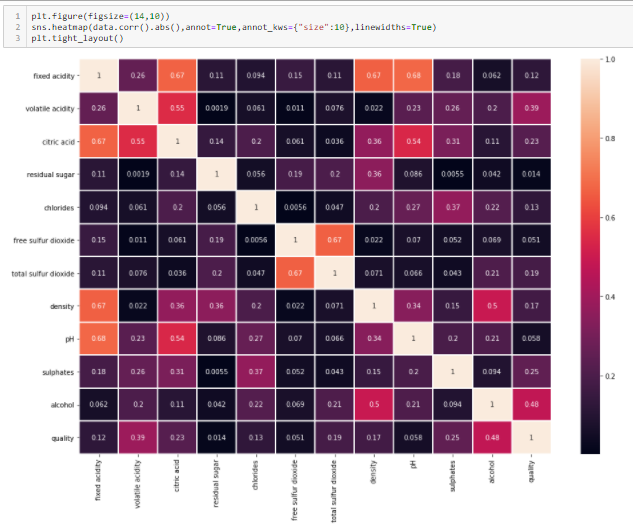
##### *Graph 10:*

In this column i.e., 'sulphates' it seems there are some outliers seems like that are greater 1.1.

##### *Graph 11:*

In this column i.e., 'alcohol' it seems there are some outliers seems like that are greater 14.

### **Checking for Multi-Collinearity:**

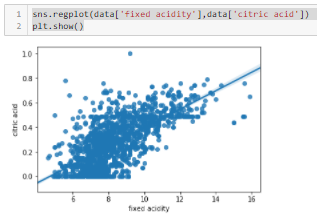
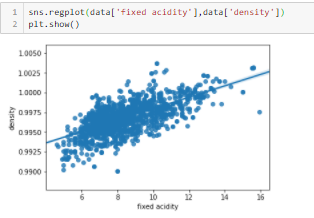


It’s seeming like there is problem existing with multicollinearity.:

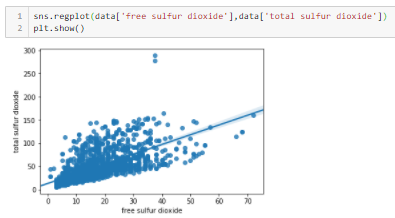
1. 'fixed acidity' is 67%,67% & 68% colinear with 'citric acid', 'density' & 'pH'.

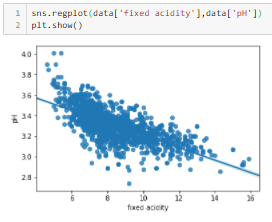
2.'free sulfur dioxide' is colinear with 'total sulfur dioxide' i.e., 67%.

**Let’s plot graph for these collinearity:**



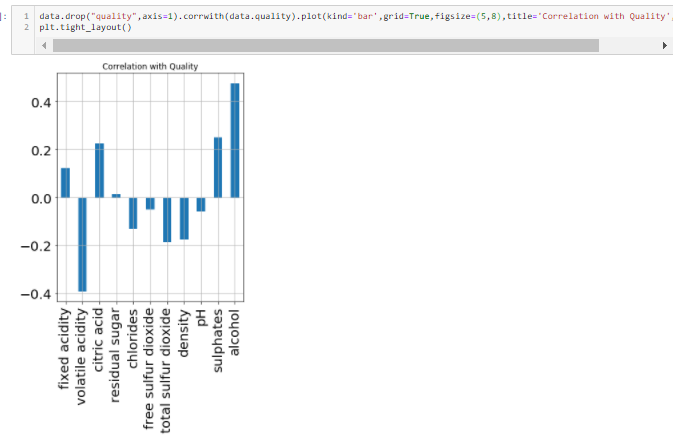
Here we can see the data is deviated and linear dependent too between ‘fixed acidity’ ~ ‘citric acid’. And ‘fixed acidity’ ~ ‘density`.





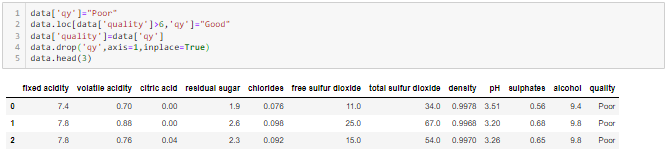
Here we can see there is linear dependent between ‘fixed acidity’ ~ ‘pH’. And ‘total sulfur dioxide’ ~ ‘free sulfur dioxide`.

### **Checking for Relation between features and Label:**



Here we can see the features relations with the target variable. In this graph some features positively corelate with the target variable & some are negatively correlated.

**Transforming Target Variable:**



Here we transform quality column in classes poor & good. where poor is the column in which the quality range from 3-6 exists & Good is the quality for 7 & 8.

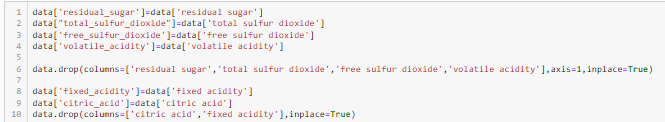
**Cleaning the data**

**STEP 8**: Dropping Duplicates



There were some duplicates values in our data that we dropped.

**STEP 9**: Transforming Column names.

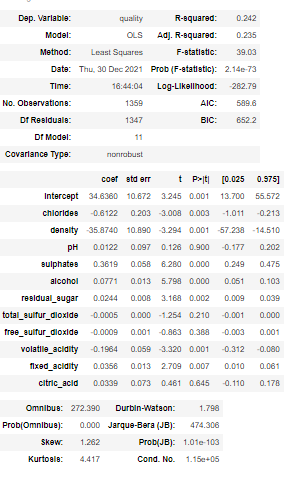


**STEP 10**: Transforming Target.



**STEP 11**: Features Selection.





Here we seeing the pValues of all the columns whose pvalues is greater than 0.05 that columns we dropped because such type columns are fails to reject the **null hypothesis**.

**citric\_acid, free\_sulfur\_dioxide, total\_sulfur\_dioxide, pH** has pValues greater than 0.05. So, we can remove these features for our model.

**STEP 12**: Dropping Columns:



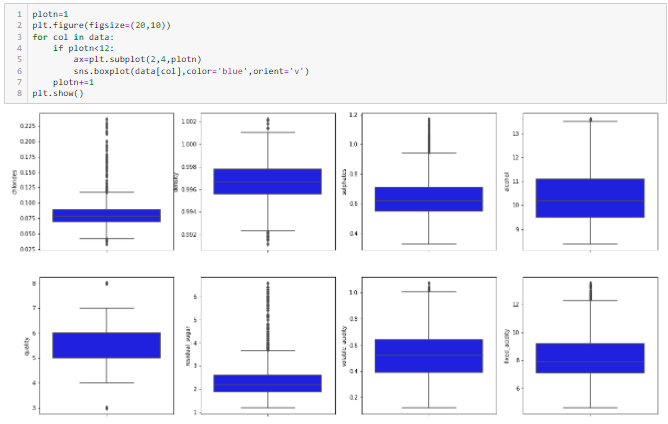
**STEP 12**: Removing Outliers using Z-score technique:



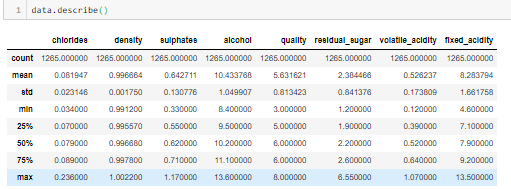
Here we take 3 for removing outliers that means we have taken 99.7% data and lost 0.3% data of each variable that dealt with Z-score and still getting outliers in some columns.

**STEP 13**: Checking for Outliers after using Z-score technique:

Z-score indicates **how much a given value differs from the standard deviation**. The Z-score, or standard score, is the number of standard deviations a given data point lies above or below mean. Standard deviation is essentially a reflection of the amount of variability within a given data set.

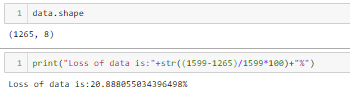


There are still some outliers like in columns 'residual sugar', 'chlorides', 'total sulfur dioxide', 'sulphates' but they are continuous too. Let's see the Statistical description of the data.



Now data seems to be good as earlier we were concerned about max values now, they also resolved.

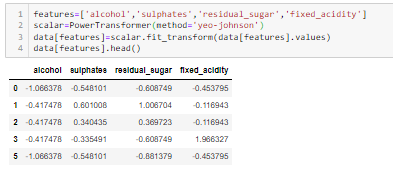
**STEP 14**: Data Lost:



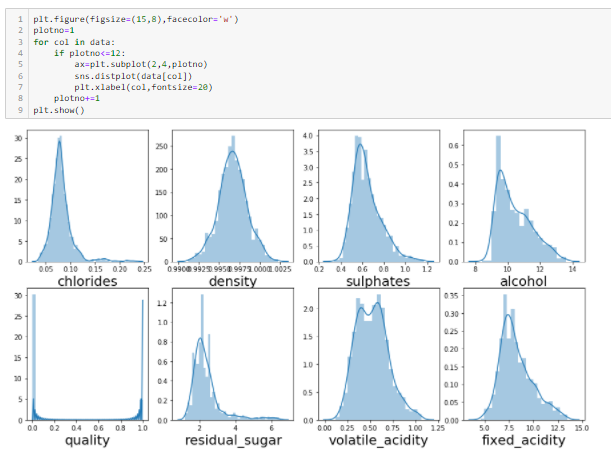
We lost 20.8% data that had duplicates values & outliers in it. That is huge loss and this because there are, 15%+ of data records has duplicates values that we removed.

**STEP 15**: Using Power Transformer technique to distribute the data columns more normally that we analyze in distplot with ‘yeo-johnson’ method because there is negative as well as 0 in the columns:

Power transforms are a technique for transforming numerical input or output variables to have a Gaussian or more-Gaussian-like probability distribution. How to use the Power Transformer in scikit-learn to use the ***Box-Cox*** and ***Yeo-Johnson*** transforms when preparing data for predictive modelling.

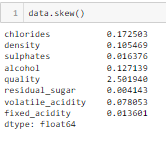


**STEP 16**: Checking distribution of the columns after Power transformer technique:



Here we can see all the columns are more normally distributed now.

**STEP 17**: Checking skewness of the columns after Power transformer technique:



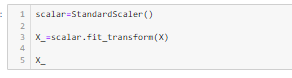
Now data has less skewed values.

**Model Deployment**

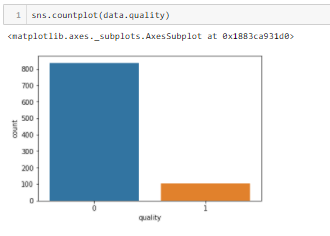
**STEP 18**: Splitting data into Features and Labels.



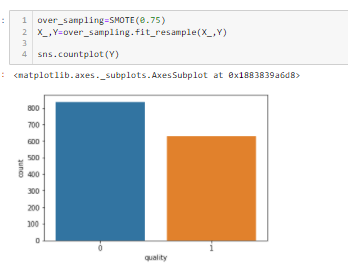
**STEP 19**: Standardized the features.



**STEP 20**: Checking Imbalancing of dataset.



**STEP 21**: Balancing dataset.



We used over Sampling technique. So, that the either any class that has low no. of count increased. Here we increased 75% of records. Due to which data is balanced now.

**STEP 22**: Increased no. of records.



Now there are 1912 no. of records and 7 standardized features.

**STEP 23**: Code for model’s accuracy and evaluations.



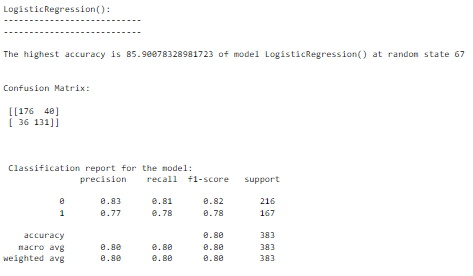
Above code shows the highest **accuracy** for the object that we created for that model that we used are:

* LogisticRegression()
* DecisionTreeClassifier()
* KNeighborsClassifier()
* RandomeForestClassifier()
* AdaBoostClassifier()
* BaggingClassifier()

The highest accuracy for each model we get for Splitting the Training and test features and labels at the **random state** between 1 to 150.

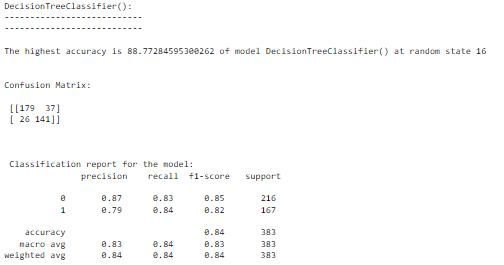
**STEP 23**: The Output of this code shows the highest accuracy, Classification Report and Confusion Matrix.

* **LogisticRegression()**



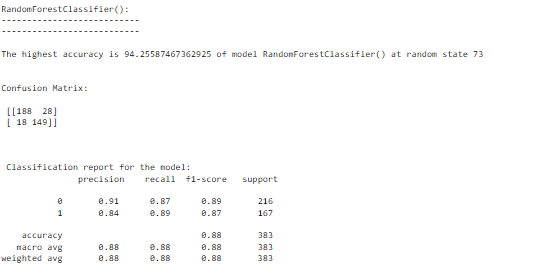
The above code shows that the model has highest accuracy **85.90%** at random state **67**. With **Confusion matrix** and **classification report**.

* **DecisionTreeClassifier()**



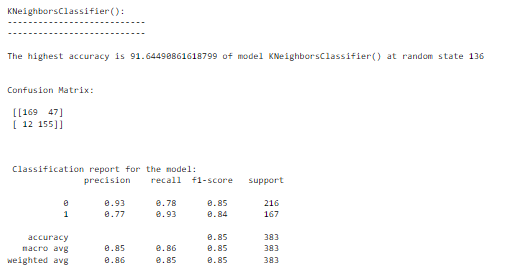
The above code shows that the model has highest accuracy **88.77%** at random state **16**. With **Confusion matrix** and **classification report**.

* **RandomForestClassifier()**

****

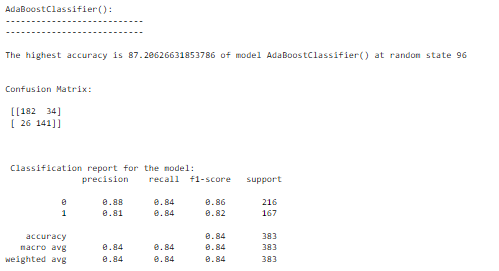
The above code shows that the model has highest accuracy **94.25%** at random state **73**. With **Confusion matrix** and **classification report**.

* **KNeighboursClassifier()**

****

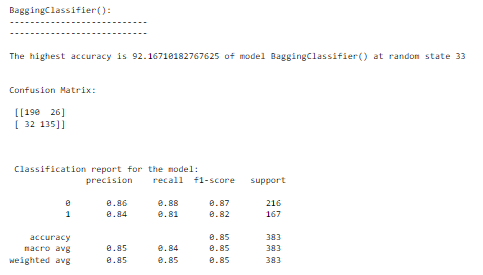
The above code shows that the model has highest accuracy **91.65%** at random state **136**. With **Confusion matrix** and **classification report**.

* **AdaBoostClassifier()**

****

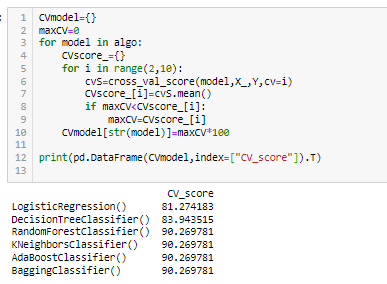
The above code shows that the model has highest accuracy **87.2%** at random state **96**. With **Confusion matrix** and **classification report**.

* **BaggingClassifier()**

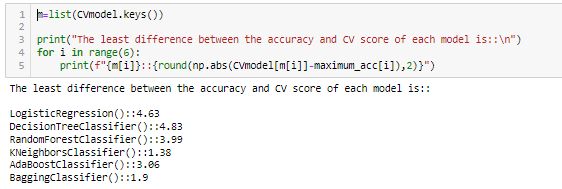
****

The above code shows that the model has highest accuracy **9.16%** at random state **33**. With **Confusion matrix** and **classification report**.

**STEP 24**: Code for model’s CV score with the best Cross-Validation.



**STEP 25**: Code for least difference between CV Score and accuracy of each Model.



**Model Selection**

Here for model **KNeighborsClassifier** we get the least value i.e., the difference between the accuracy and cv score of this model is **1.38**.

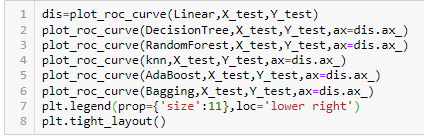
**STEP 26**: Splitting Data into train test dataset.



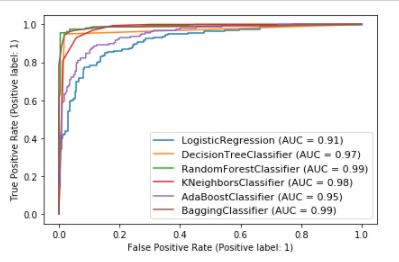
Splitting Data at random state i.e., best we get accuracy for KneighboursClassifier().

**STEP 27**: Plotting ROC Curve.

Code:



Output:



The Output shows the ROC curves are frequently used to **show in a graphical way the connection/trade-off between clinical sensitivity and specificity for every possible cut-off for a test** or a combination of tests.

**Hyper Parameter Tuning**

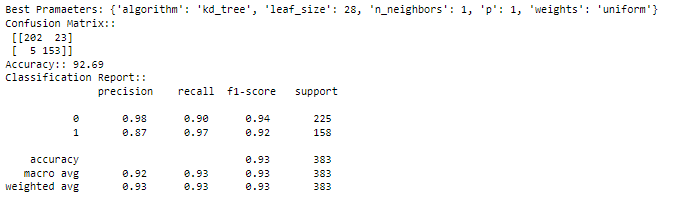
**STEP 28**: Hyper Tuning the model.

Code:



The above code is used to first get the best parameter by Hyper Tuned the model using Grid Search CV technique after the parameter we get from tuning we re-instantiated these parameters to the model after that we get the best accuracy along with that the Confusion Matrix and classification report.

Output:



The Output shows the best parameter that we used in model after that we get the best accuracy or increase in the accuracy of the model along with the Confusion Matrix and Classification Report.

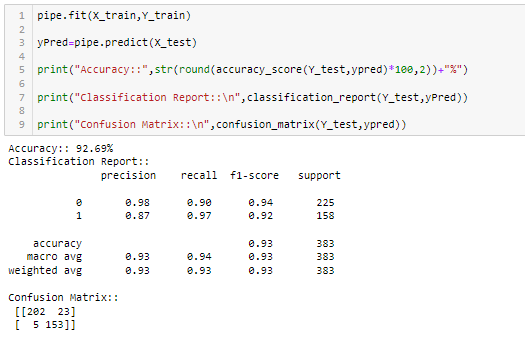
# Creating Pipeline:

**STEP 29**: Creating Pipeline for the model.

Code:



Here we created pipeline for the model. Pipeline is **a way to codify and automate the workflow it takes to produce a machine** learning model.



Through pipeline here we predicted the features and showing the accuracy, Confusion Matrix and Classification report.

**STEP 30**: Saving model.



Here we saving the model after creating PipeLine.

# Concluding Remarks:

A quick recap recap on all the steps that we went through starting from

1.Understanding the **Problem definition**: here we get to know about the features and Labels.

2. **Data Pre-processing**: In this section we get to know about the data types of the columns, Null-values, shape of the data and statistical description of the data.

3. **EDA**: In this section we do analysis get know about the multi-collinearity problem existence, Distribution problem, Outlier’s detection, Relation of features to the label by plotting different types of graphs.

4. **Data Cleaning:** in this section we remove the duplicate values, Outliers, Normalize the dataset after that seeing the again the statistical description of data to know about data is there anything that seems to be in-proper. And see the how much percentage of data we lost.

5. **Model Deployment**: In this section we split our data into features and labels. Then standardized the features. Seeing there is imbalance of data if it s then e use oversampling to balance the dataset. Then we used **6 model to know the best accuracy with its random state.** And also selected the best model by seeing the minimum difference between **cv score** and accuracy. And then we split the training and test features and labels with the best model random state. And plot the **ROC curve** of each model.

6. **Hyper Parameter Tuning**: in this we used Grid Search CV to know the best parameter of the best model through which we can increase the accuracy of the model.

7.**Pipeline**: In this we create pipeline for the model by using the standard scaler techniques and the best model with best its best parameter and saved that model.

What I do is code my entire project on my own and then take a peek at the internet to look through other’s coding style for inspiration and understand if I can incorporate anything to improvise further on accuracy or beautify the visuals. However, I have seen many people doing the complete opposite whereupon they don’t practise or create their own unique coding style first and rather copy paste lines from the web and perform some sort of messy patch work and when asked to explain might not be capable of conveying functioning or usage of those code blocks.

Before wrapping up my only advise to everyone is “No pain No gain” you will have to get your hands dirty with building your own code and trying out all the permutations and combinations. Create a self-made unique data story telling commandment list and follow it along with the standard project life cycle. Hope this at length article helps you in gaining the initial knowledge on building your first project from **JupyterNotebook**.

**DISCLAIMER:** I am a newbie myself in this field with some accumulated knowledge over a year’s time studying Data science and felt like since sharing is caring someone who’s stepping in now can benefit from my experience. I am also open to get some feedback from anyone that will help me in improving too! The content that I have written is solely my view of the project but it’s definitely inspired by others over the internet who have worked on similar projects before me.