SUMMER INTERNSHIP PROGRAM

Wine Quality Prediction

Team Name:-

Data Pyrates

Team Members:-

Abhinav Bajaj

Apoorav Nigam

Krishna Nathwani

Nasir Hussain

Submitted to:-

Miss. Ramya Guntamukkala

Wine Quality Prediction

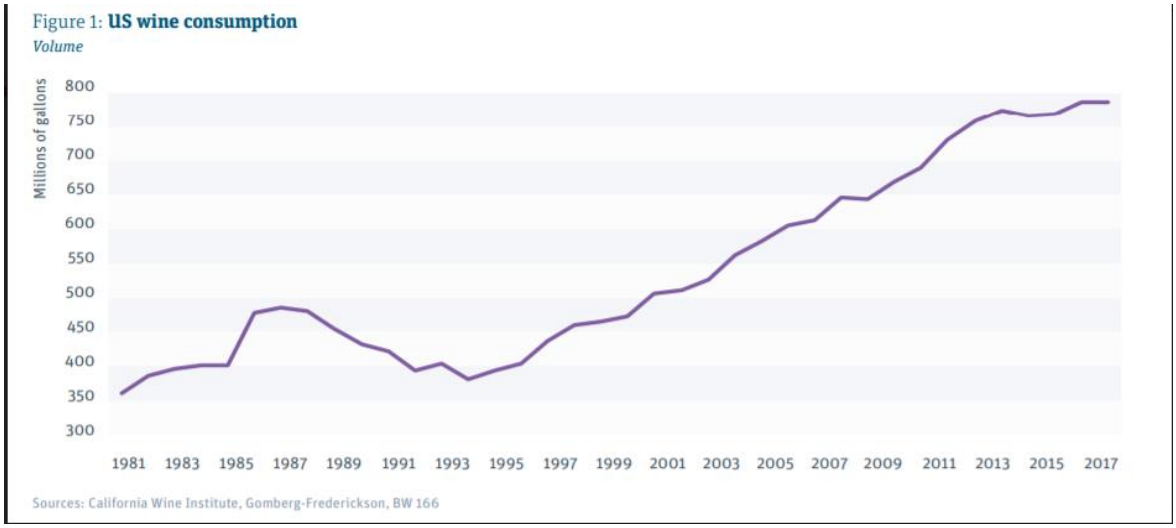
Introduction

Today, all types of industries are improving by adopting new technologies and applying them in all areas. These technologies help to enhance the production and making the whole production process smooth. But, still there are different areas, which demands human expertise such as product quality assurance. Nowadays, it becomes an expensive process as the demand of a product grows over time. Therefore, this project aims to use a machine learning algorithm for product quality assurance. These techniques performs quality assurance process with the help of available characteristics of product and automate the process by minimizing human interfere. The work also identifies the important features to predict the values of dependent variables.
In the project, the random forest algorithm implemented to determine dependency of wine quality on different 13 physicochemical characteristics. Moreover, the predictions are also made for wine quality on the basis of important variables/characteristics, selected according to their dependencies.

- **Objectives**
  1) To visualize and understand the given database by using graphs, diagrams and images.
  2) To test the dataset using different machine learning algorithms and find the best one.
  3) To fit the dataset into the best model
  4) To predict which classification the input given by the user belongs to.
  5) Finally to make the User Interface for easy use of the application

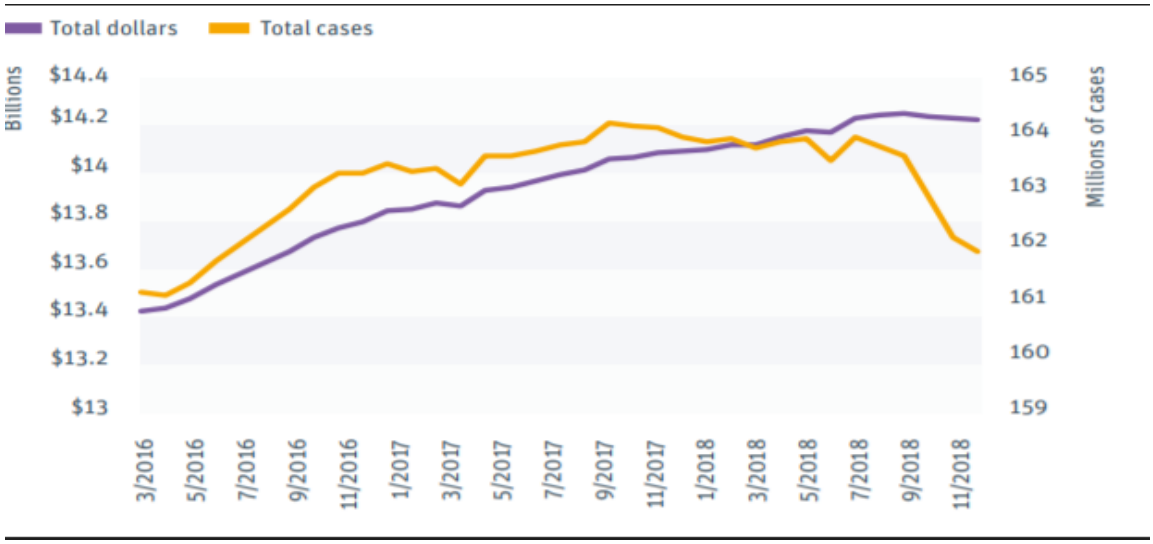
- **Problem Statement:**
  Wine is categorised based on quality and price is set accordingly allowing the customer to purchase wine of his choice. Quality in the sense, the ingredients used while manufacturing of wine. However, each type of wine has its own uniqueness. This classification is also used to set the prices accordingly so that the firm enjoys the profit. By prediction of the quality of wine every individual can purchase it as per their financial status. Since there is no, such a system to predict the quality of wine. This project would be worth making it

- **Industry Profile:**
  2018 was a good year for wine. Total wine sales for the year set a record, restaurant sales of wine were higher and premium wine sales were up as well. Strong consumer confidence and a healthy US economy contributed to the improved performance, but changes to long-term trends are telling us that we are at a transition point as an industry.



The US is the largest wine consuming country in the world, giving US producers an amazing home-court advantage. We have been researching the wine business since 1991 and making predictions for far more than a decade. Some years, we properly characterize a market change. In other years, our findings might be off in terms of timing or even wrong, 8 but we always review the forecasts made the prior year just to keep score. We predict sales growth of 4 to 8 percent for the premium wine segment, and between 0.5 and 2.5 percent growth for off-premise retail sales

We saw a routine trend of growing brand strength in the wine trade, evidenced by snowballing volumes and increasing pricing opportunities for retailers. In the late 1990s, consumer demand was so strong, Silicon Valley Bank often had winery clients selling out of wine by midyear and distributors scouring the corners of the wine business, begging small wineries for their product 11 that's clearly not the case today.



Review of Literature

- **Existing System:** There is no such system available to predict the Wine Quality considering all the factors. The wine sales for the year set a record, restaurant sales of wine were higher and premium wine sales were up as well. The wine quality was predicted by seeing only the content of Alcohol overall.

- **Scope of Extension:** This contradiction paved a path for Machine Learning to help in prediction of Wine Quality nevertheless there were many factors to be considered. In other years, our findings might be off in terms of timing or even wrong, 8 but we always review the forecasts made the prior year just to keep score. We predict sales growth of 4 to 8 percent for the premium wine segment, and between 0.5 and 2.5 percent growth for off-premise retail sales.

- **Technologies:**
    1. Operating System: Machine Learning is platform independent, it can be accessed on Android, Windows, Linux or any other OS the user wishes to open.

    2. Programming Languages: Only Python is used in the project.

    3. Tools: The tools used in the Machine Learning are:
        - IBM Cloud
        - IBM Watson studio
        - Node – Red
        - Jupyter Notebook

Data Collection

In the following project, here we are using data-set named 'wine.csv' which is a CSV (Comma Separated Values) file containing all the different contents of wine which differs and categorises quality of wine. Using the mentioned programming language "Python" we import the data-set in the Jupyter Notebook using one of the Python libraries named "Pandas".

Suggested website for Wine Dataset:

- https://github.com/shri1407/Wine-Quality-Dataset/blob/master/winequality-red.csv
- https://github.com/shri1407/Wine-Quality-Dataset/blob/master/winequality-white.csv
- https://www.kaggle.com/uciml/red-wine-quality-cortez-et-al-2009#winequality-red.csv

Data set description:

- **Alcohol:** A colourless volatile flammable liquid which is produced by the natural fermentation of sugars.
- **Malic Acid:** A crystalline acid present in unripe apples and other fruits.
- **Ash:** The powdery residue left after the burning of a substance.
- **Magnesium:** Magnesium is important for many processes in the body, including regulating muscle and nerve function.
- **Phenols:** They are probably the most important group of flavour chemicals in red wines.
- **Flavanoids:** Red wine is high in flavonoids, which are antioxidants.
- **Proanthocyanins:** Proanthocyanidins are the principal polyphenols in red wine that are under research to assess risk of coronary heart disease.
- **Color** Intensity: A simple measure of how dark the wine is using a summation of absorbance measurements in the violet, green and red.
- **Hue:** There are some fascinating insights you can gain just by looking at the color, hue, and intensity of red wine.
- **Proline:** Proline is typically the most abundant amino acid present in grape juice and wine.

Methodology

The following application was made to find the quality of wine. The given datasets had 14 properties where the final property is the category in which that particular wine is present. The category of the wine can be figured out by using the other 13 features present in the dataset. Each of these features contribute evenly to the quality of the wine.

The following application was made in Python for its back end and Node red for the front end. All the coding as well as the execution of the program was done in Watson Studio.

First the dataset was uploaded and inserted in the Jupyter Notebook using the following code.

```
import types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share your notebook.
client_97b32920e033492f91b7c1fa18bdebdd = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='LUN8sW6eQC33bBhT0j8XobIbrwSIfc7wxTEyZsic6oFL',
    ibm_auth_endpoint="https://iam.bluemix.net/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.eu-geo.objectstorage.service.networklayer.com')

body = client_97b32920e033492f91b7c1fa18bdebdd.get_object(Bucket='wine-donotdelete-pr-h1fkrosgo9wvy1',Key='Wine.csv')['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, body )

df_data_1 = pd.read_csv(body)
df_data_1.head()
```

The X and y variables were then initialized where y was the category and X all other features

```
X = df_data_1.iloc[:, 0:-1].values
y = df_data_1.iloc[:, -1].values
```

The analysis of data will start here

We first use seaborn’s pair-plot to find every graph possible

In [21]:
```
import seaborn as sns
#%matplotlib inline


sns.set_palette("GnBu_d")
sns.set_style('whitegrid')
#sns.jointplot(x='Administration',y='Profit',data=dataset)
# Visualising the Test set results

sns.pairplot(df_data_1)
```

- **Figures and tables:**
  - Dataset Sample
    - Head

```
df_data_1 = pd.read_csv(body)
df_data_1.head()
```

Out[1]:

	Alcohol	Malic_Acid	Ash	Ash_Alcanity	Magnesium	Total_Phenols	Flavanoids	Nonflavanoid_Phenols	Proanthocyanins	Color_Intensity	Hue	OD280	Proline	Customer_Segment
0	14.23	1.71	2.43	15.6	127	2.80	3.06	0.28	2.29	5.64	1.04	3.92	1065	1
1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	1.28	4.38	1.05	3.40	1050	1
2	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30	2.81	5.68	1.03	3.17	1185	1
3	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24	2.18	7.80	0.86	3.45	1480	1
4	13.24	2.59	2.87	21.0	118	2.80	2.69	0.39	1.82	4.32	1.04	2.93	735	1

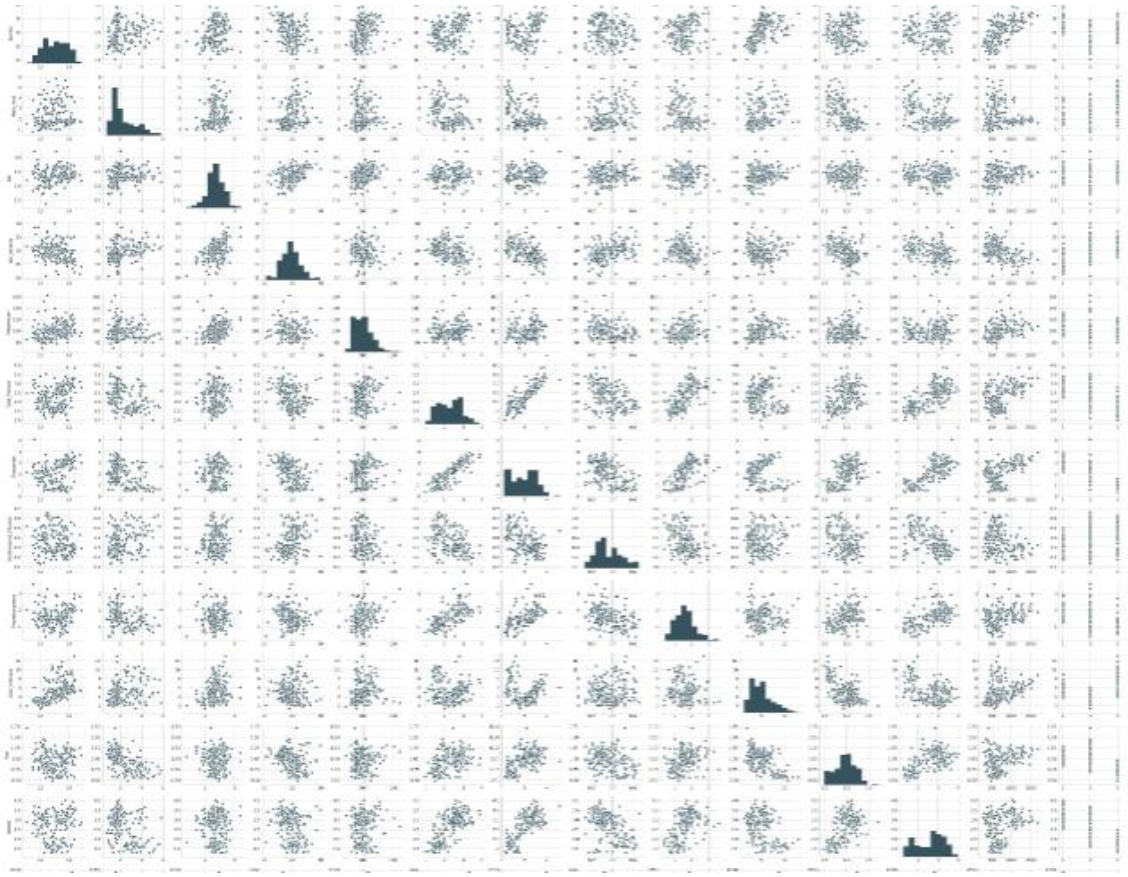
    - Describe

In [3]:
```
df_data_1.describe()
```

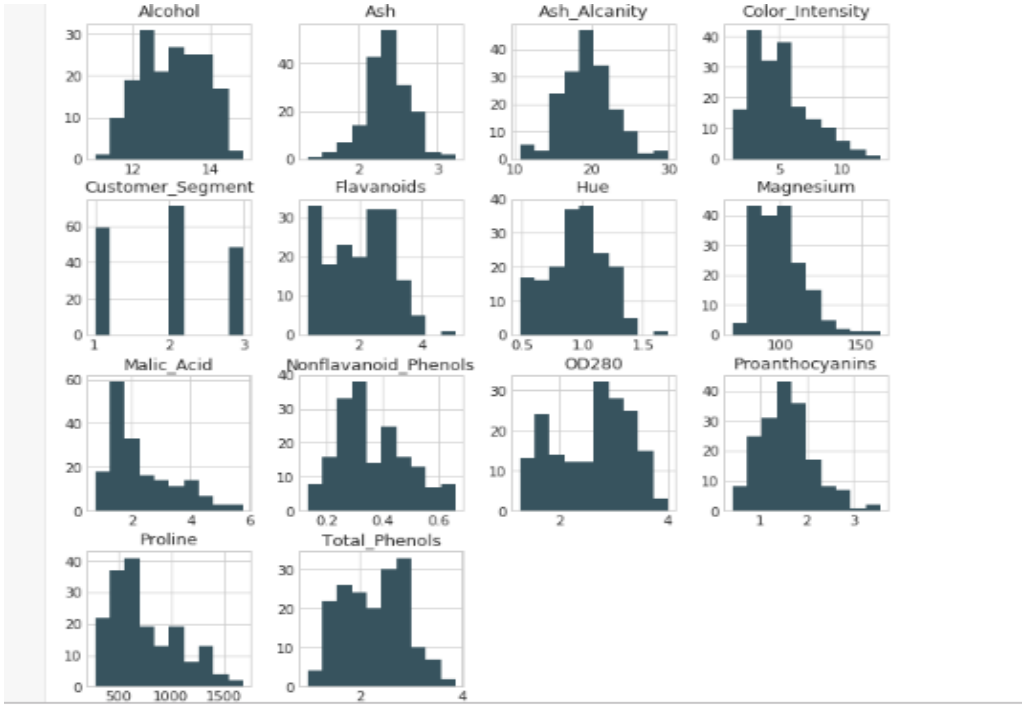
Out[3]:

	Alcohol	Malic_Acid	Ash	Ash_Alcanity	Magnesium	Total_Phenols	Flavanoids	Nonflavanoid_Phenols	Proanthocyanins	Color_Intensity	Hue	OD280	Proline	Customer_Segment
count	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000
mean	13.000618	2.336348	2.366517	19.494944	99.741573	2.295112	2.029270	0.361854	1.590899	5.058090	0.957449	2.611685	746.893258	1.938202
std	0.811827	1.117146	0.274344	3.339564	14.282484	0.625851	0.998859	0.124453	0.572359	2.318286	0.228572	0.709990	314.907474	0.775035
min	11.030000	0.740000	1.360000	10.600000	70.000000	0.980000	0.340000	0.130000	0.410000	1.280000	0.480000	1.270000	278.000000	1.000000
25%	12.362500	1.602500	2.210000	17.200000	88.000000	1.742500	1.205000	0.270000	1.250000	3.220000	0.782500	1.937500	500.500000	1.000000
50%	13.050000	1.865000	2.360000	19.500000	98.000000	2.355000	2.135000	0.340000	1.555000	4.690000	0.965000	2.780000	673.500000	2.000000
75%	13.677500	3.082500	2.557500	21.500000	107.000000	2.800000	2.875000	0.437500	1.950000	6.200000	1.120000	3.170000	985.000000	3.000000
max	14.830000	5.800000	3.230000	30.000000	162.000000	3.880000	5.080000	0.660000	3.580000	13.000000	1.710000	4.000000	1680.000000	3.000000

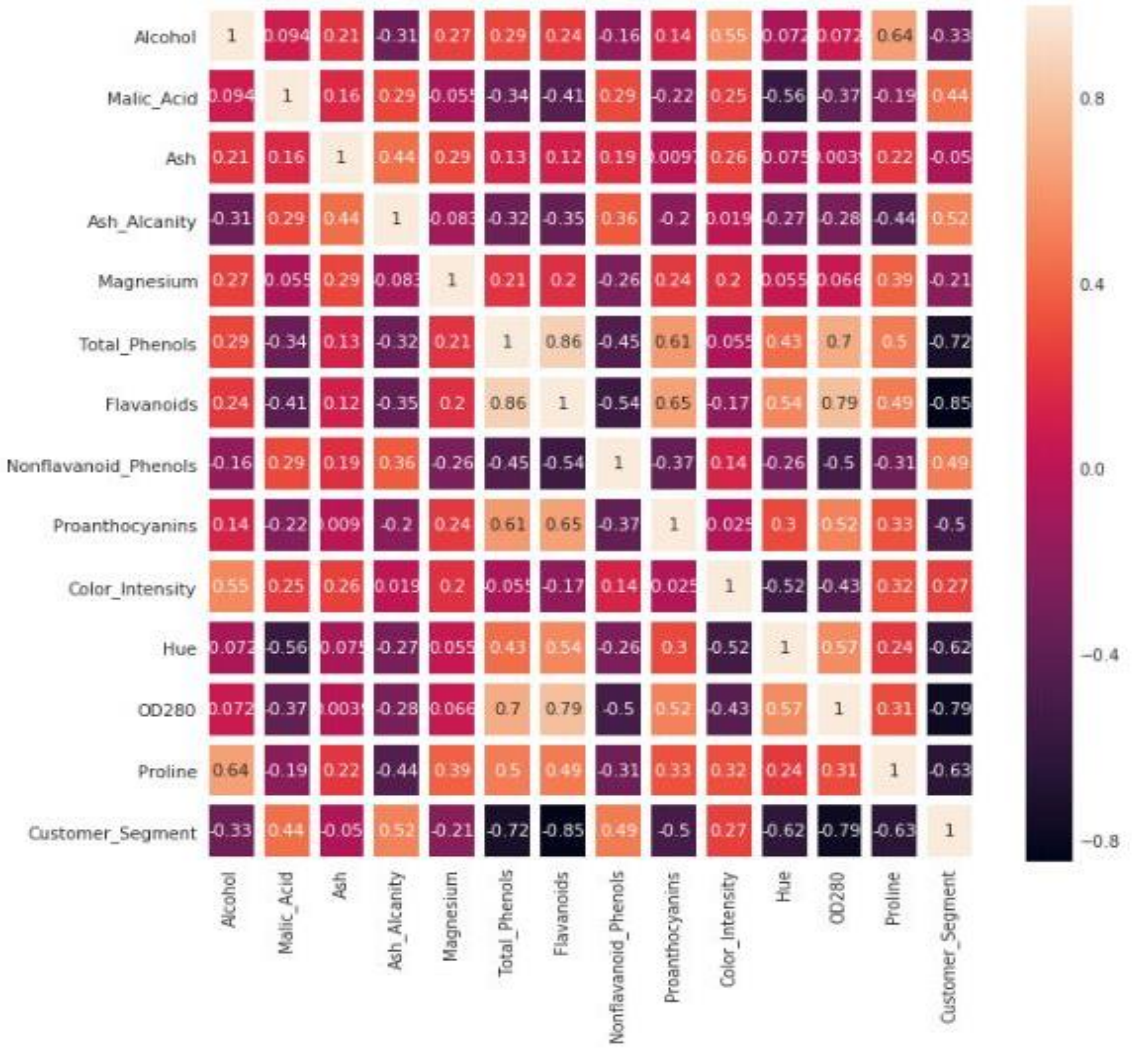
- Graphs
  - Pairplot



  - Histogram



- Heatmap



- **Statistical Techniques and Data Visualization:**

Then the data is spilt into test and train data. This is done so there is certain unseen data that the predicted values can be tested on. The train data is the data that the model is trained on and the test data the data on which the predicted value will be tested

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state = 0)
```

The different values of X_train have different ranges of values. To have the same range of value a scaler is applied. Here we have used Standard Scaler to scale all the values.

```
In [109]: # Feature Scaling
          from sklearn.preprocessing import StandardScaler
          sc = StandardScaler()
          X_train = sc.fit_transform(X_train)
          X_test = sc.transform(X_test)
```

- **Data Modelling using Supervised ML techniques:**

Finally the model is fit. We have used the Random Forest algorithm to create the classifier needed to fit the model. Then the model is fitted with X_train and y_train variables using the .fit function.

```
In [114]: # Fitting Random Forest Classification to the Training set
          from sklearn.ensemble import RandomForestClassifier
          classifier2 = RandomForestClassifier(n_estimators = 10, criterion = 'entropy', random_state = 1)
          classifier2.fit(X_train, y_train)

Out[114]: RandomForestClassifier(bootstrap=True, class_weight=None, criterion='entropy',
                      max_depth=None, max_features='auto', max_leaf_nodes=None,
                      min_impurity_decrease=0.0, min_impurity_split=None,
                      min_samples_leaf=1, min_samples_split=2,
                      min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=1,
                      oob_score=False, random_state=1, verbose=0, warm_start=False)
```

The .predict function is then called to predict the values of X_test

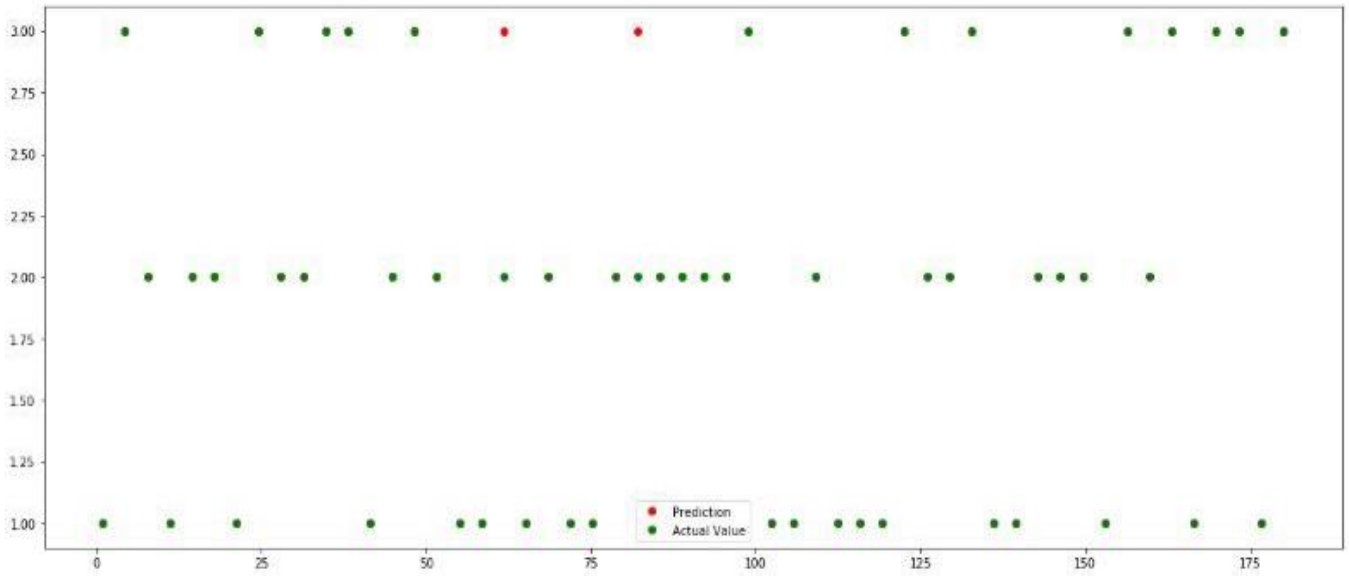
```
In [115]: # Predicting the Test set results
          y_pred1= classifier2.predict(X_test)
          y_pred1

Out[115]: array([1, 3, 2, 1, 2, 2, 1, 3, 2, 2, 3, 3, 1, 2, 3, 2, 1, 1, 3, 1, 2, 1, 1,
                 2, 3, 2, 2, 2, 2, 3, 1, 1, 2, 1, 1, 1, 3, 2, 2, 3, 1, 1, 2, 2, 2, 1,
                 3, 2, 3, 1, 3, 3, 1, 3])
```

The values for X_test is collected now we compare the predicted values with the real values i.e. y_test .We then score the predicted value according to its accuracy.

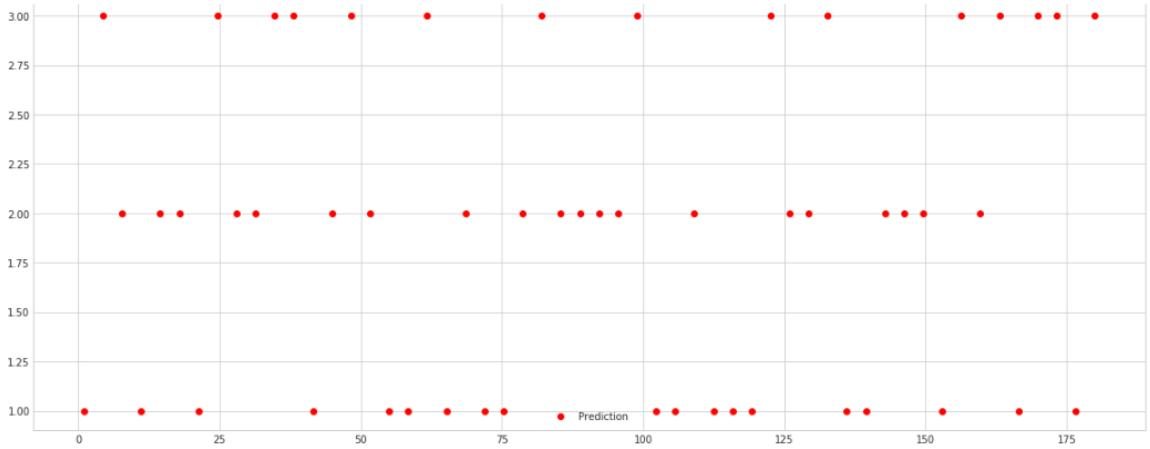
```
In [116]: from sklearn.metrics import r2_score
          r2_score(y_test, y_pred1)

Out[116]: 0.93617021276595747
```

As we can see the accuracy is 93.6% so the model was successfully trained and tested

- Graph to show the original y_test values and the errors made by the classifier



- Graph to show all predicted values



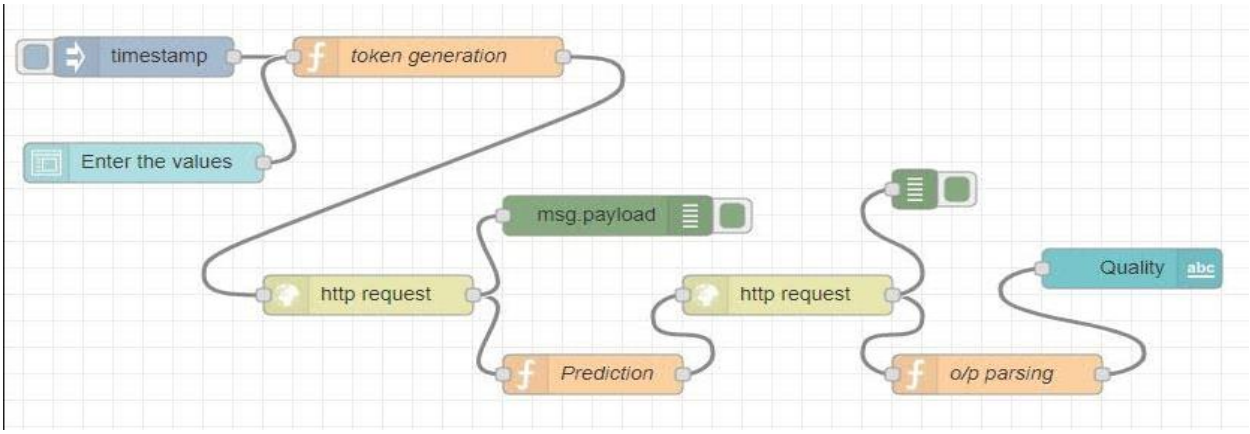
The User Interface of the application was made using Node Red.

Firstly, from Node Red starter kit we create a Cloud Foundry app. Then we proceed to create the application flow with following steps:

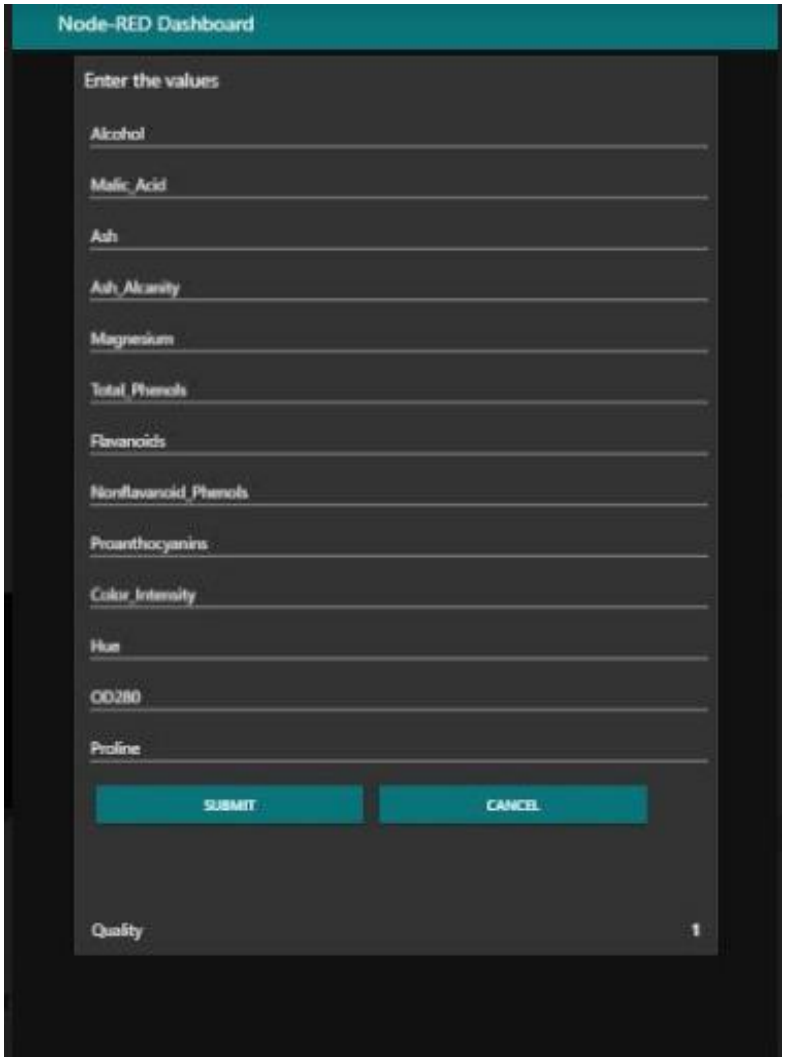
- For input we eject a timestamp
- Then we create a function to generate the token by encrypting username and password, and importing columns of the datasets
- Then we import a form node by using dashboard pallet to create the form elements like input box, buttons and labels
- Then we establish a connection that can interact back end and front end, we use GET method to get the data from server by the following URL given by service credentials of machine learning modules
- Then we create a function as 'prediction' to get the input value from the form that we created
- After that we post the input values through end point URL to predict the quality

- Then we import a text node to display the result
- After creating the flow we deploy it and run the URL on a new tab

- **The flow of the application**



- **The final UI**



Conclusion

The algorithm we used is efficient to predict the quality of Wine, here with this model we are close enough to successfully predict the quality of Wine. Since, we got the accuracy of 93.76% there is no such algorithm derived that can predict 100% accuracy. Machine Learning overall is a very powerful tool to revolutionize the way things work. It is very user-friendly and hence helps the user to predict the data easily and simultaneously keep improving the accuracy and efficiency by itself.

References

- Algorithm Suggestion - https://www.kaggle.com/vishalyo990/prediction-of-quality-of-wine

- Report Content - https://www.kaggle.com/vishalyo990/prediction-of-quality-of-wine