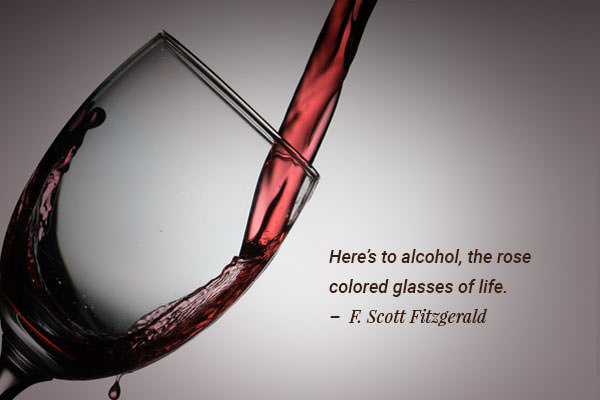
# Red Wine Quality Prediction Using Machine Learning

## **Steps to build and save a model**

* **Basics understanding of Wine.**
* **Data description**
* **Importing modules**
* **Study dataset**
* **Visualization**
* **Handle null values**
* **Split dataset**
* **Normalization**
* **Applying model**
* **Save model**
* **Endnote**

## **Introduction**



## **“Wine can be a better teacher than ink, and banter is often better than books”**

― **Stephen Fry,**[**The Fry Chronicles**](https://www.goodreads.com/work/quotes/13520837)

We definitely came across the fruit **graphs**, which is soo sweet on the test but graphs are not just to eat, they are used to make different types of things. Wine is one of them **Wine is an alcoholic drink that is made up of fermented grapes**. If you have come across wine then you will notice that wine has also their type they are red and white wine this was because of different varieties of graphs.

You are shocked to hear that the worldwide distribution of wine is 31 million tons which were huge in number.

### **What if you think about the quality of wine, how can you differentiate the wine according to their quality?**

According to experts, the wine is differentiated according to its **smell**, **flavor**, and **color,**but we are not a wine expert to say that wine is good or bad. What will we do then? Here’s the use of **Machine Learning** comes, yes you are thinking to write we are using machine learning to check wine quality. ML have some techniques that will discuss below:

To the ML model, we first need to have data for that you don’t need to go anywhere. <https://github.com/Priyapawar231/Datasets> can find the data set in the given link .

Now, we start our journey towards the prediction of wine quality, as you can see in the data that there is red and white wine, and some other features. Let’s start :

## **Description of Dataset**

If you download the dataset, you can see that several features will be used to classify the quality of wine, many of them are chemical, so we need to have a basic understanding of such chemicals.

* **volatile acidity:**   Volatile acidity*is the*gaseous acids present in wine.
* **fixed acidity:**Primary **fixed acids** found in wine are **tartaric**, **succinic**, **citric**, and **malic**
* **residual sugar:**Amount of sugar left after fermentation.
* **citric acid:** It is weak organic acid, found in citrus fruits naturally.
* **chlorides:**Amount of salt present in wine.
* **free sulfur dioxide:**   So2 is used for prevention of wine by oxidation and microbial spoilage.
* **total sulfur dioxide**
* **pH:**In wine pH is used for checking acidity
* **density**
* **sulphates:**    Added sulfites preserve freshness and protect **wine** from oxidation, and bacteria.
* **alcohol:**   Percent of alcohol present in wine.

Rather than chemical features, you can see that there is one feature named **Type**it contains the types of wine we here discuss on **red** and **white** wine, the percent of red wine is greater than white.

For the next step we have to import some important library :

### **Importing modules**

Let’s import,

Graphical user interface, text

Description automatically generated

Let’s we take brief about these libraries; **pandas** are used for data analysis **NumPy** is for n-dimensional array **seaborn** and **matplotlib**both have similar functionalities which are used for visualization.

The next step is to read the wine quality dataset and see their information:

### **Study dataset**

For the next step, we have to check what technical information contained in the data,

Table

Description automatically generated

As we see in the above image, there is vital information on features and with this information, we will process our next work.

### **Visualization**

We know that the “image speaks everything” here the visualization came into the work, we use visualization for explaining the data. In other words, we can say that it is a graphic representation of data that is used to find useful information.

Chart

Description automatically generated

The above image reveals that how that data is easily distributed on features.

Now, we plot the bar graph in which we check what value of alcohol can able to make changes in quality.

plt.figure(figsize=[10,6])

# plot bar graph

plt.bar(df['quality'],df['alcohol'],color='red')

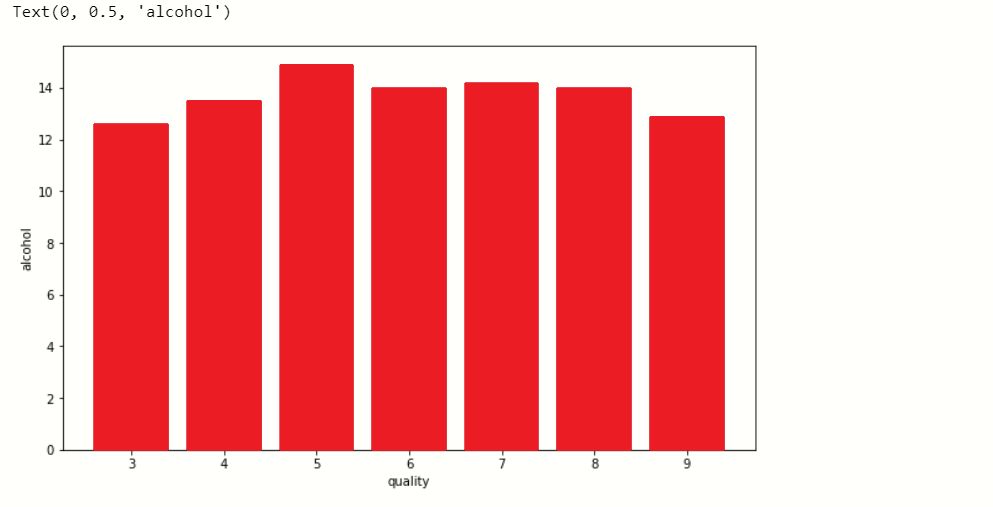
# label x-axis

plt.xlabel('quality')

#label y-axis

plt.ylabel('alcohol')

**output:-**



When we performing any machine learning operations then we have to study the data features deep, there are many ways by which we can differentiate each of the features easily. Now, we will perform a correlation on the data to see how many features are there they correlated to each other.

### Correlation:-

For checking correlation, we use a statistical method that finds the bonding and relationship between two features.

Chart

Description automatically generated

Now, we have to find those features that are fully correlated to each other by this we reduce the number of features from the data.

If you think that why we have to discard those correlated, because relationship among them is equal they equally impact on model accuracy so, we delete one of them.

for a in range(len(df.corr().columns)):

for b in range(a):

if abs(df.corr().iloc[a,b]) >0.7:

name = df.corr().columns[a]

print(name)

Here we write a python program with that we find those features whose correlation number is high, as you see in the program we set the correlation number greater than 0.7 it means if any feature has a correlation value above 0.7 then it was considered as a fully correlated feature, at last, we find the feature total sulfur dioxide which satisfy the condition.

So, we drop that feature

new\_df=df.drop('total sulfur dioxide',axis=1)

### **Handle null values**

In the dataset, there is so much notice data present, which will affect the accuracy of our ML model. In machine learning, there are many ways to handle null or missing values. Now, we will use them to handle our unorganized data.

new\_df.isnull().sum()

We see that there are not many null values are present in our data so we simply fill them with the help of the **fillna()** function.

new\_df.update(new\_df.fillna(new\_df.mean()))

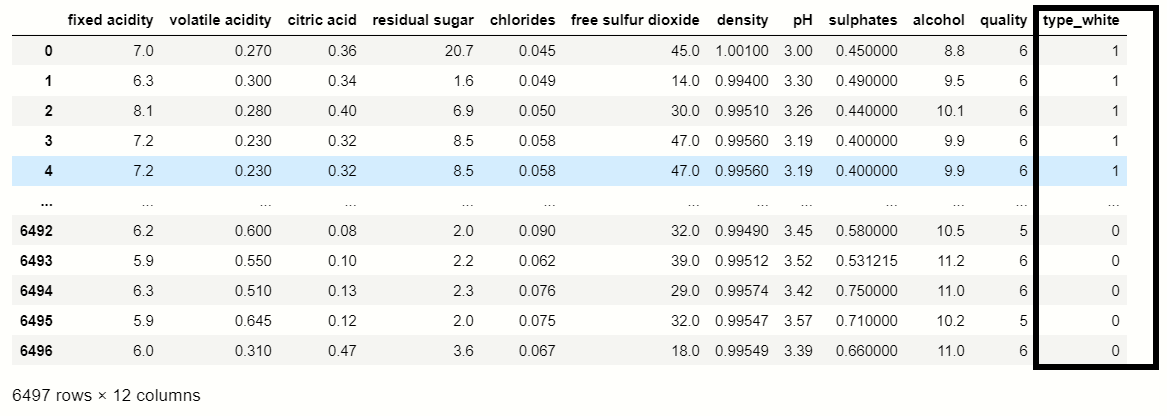
with this, we handle only numerical variables value because, we fill **mean()**and mean value is not for categorical variables, so for categorical variables:-

# catogerical vars

next\_df = pd.get\_dummies(new\_df,drop\_first=True)

# display new dataframe

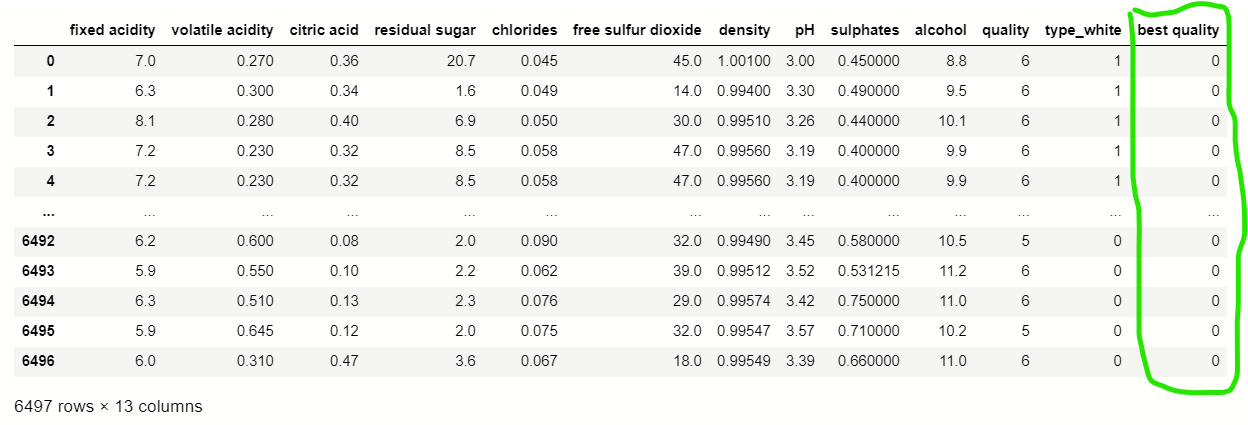
next\_df



You were able to see that the **get\_dummies()** function which is used for handling categorical columns, in this dataset **‘Type’**feature contains two types **Red**and **White,**where Red consider as **0** and white considers as **1.**

df\_dummies[''best quality''] = [ 1 if x>=7 else 0 for x in df.quality]

print(df\_dummies)



### **Splitting dataset**

Now we perform a split operation on our dataset:

from sklearn.model\_selection import train\_test\_split

x\_train,x\_test,y\_train,y\_test = train\_test\_split(x,y,test\_size=0.2,random\_state=40)

### **Normalization**

We do normalization on numerical data because our data is unbalanced it means the difference between the variable values is high so we convert them into 1 and 0.

#importing module

from sklearn.preprocessing import MinMaxScaler

# creating normalization object

norm = MinMaxScaler()

# fit data

norm\_fit = norm.fit(x\_train)

new\_xtrain = norm\_fit.transform(x\_train)

new\_xtest = norm\_fit.transform(x\_test)

# display values

print(new\_xtrain)

## **Applying Model**

This is the last step where we apply any suitable model which will give more accuracy, here we will use ***RandomForestClassifier*** because it was the only ML model that gives the 88% accuracy which was considered as the best accuracy.

#### Random Forest Classifier:-

# Importing modules

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import classification\_report

#creating RandomForestClassifier constructor

rnd = RandomForestClassifier()

# fit data

fit\_rnd = rnd.fit(new\_xtrain,y\_train)

# predicting score

rnd\_score = rnd.score(new\_xtest,y\_test)

print('score of model is : ',rnd\_score)

# display error rate

print('calculating the error')

# calculating mean squared error

rnd\_MSE = mean\_squared\_error(y\_test,y\_predict)

# calculating root mean squared error

rnd\_RMSE = np.sqrt(MSE)

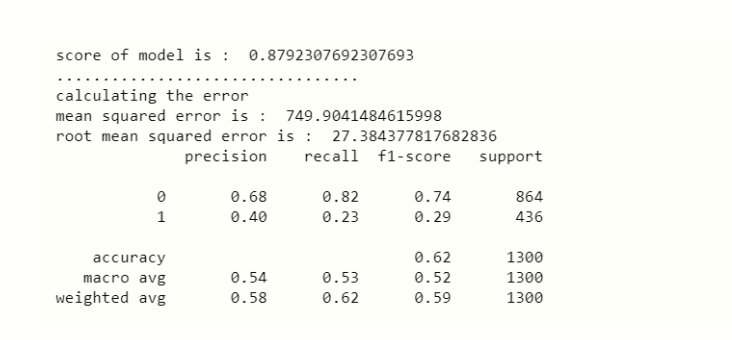
# display MSE

print('mean squared error is : ',rnd\_MSE)

# display RMSE

print('root mean squared error is : ',rnd\_RMSE)

print(classification\_report(x\_predict,y\_test))



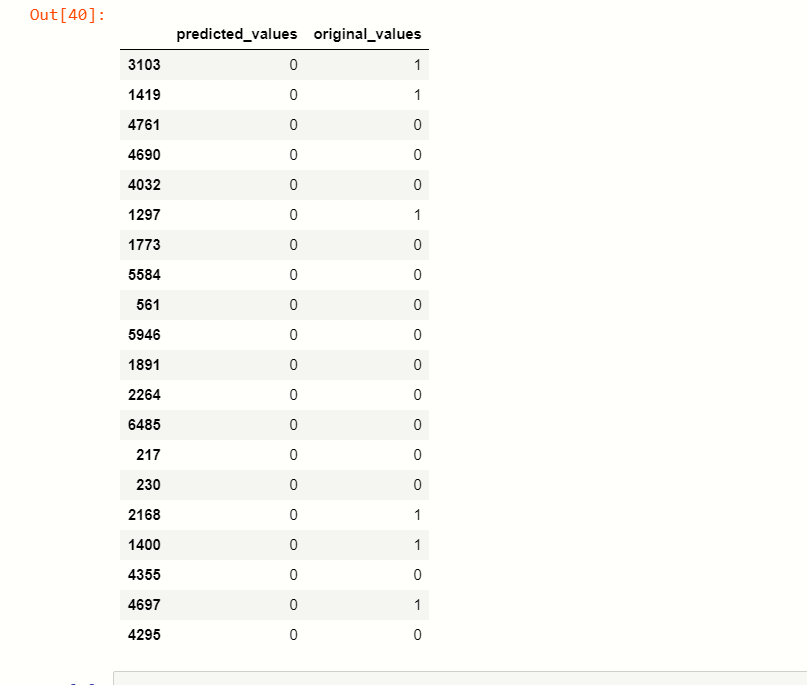
Now, we are at the end of our article, we can differentiate the predicted values and actual value.

x\_predict = list(rnd.predict(x\_test))

predicted\_df = {'predicted\_values': x\_predict, 'original\_values': y\_test}

#creating new dataframe

pd.DataFrame(predicted\_df).head(20)



### Saving Model

At last, we save our machine learning model:

import pickle

file = 'wine\_quality'

#save file

save = pickle.dump(rnd,open(file,'wb'))

So, at this step, our machine learning prediction is over.

## **End Notes**

This is my very first article that I have written because it was on current top technology machine learning, but I was used basic language to explain this problem so, you can’t get difficulty on understanding.

Thank you.

*Priyanka*