PREDICTIVE MODELLING PROJECT

WINE QUALITY PREDICTION

SUBMITTED BY - MEGHA

RED WINE

Red wine has been part of social, religious, and cultural events for hundreds of years. Medieval monasteries believed that their monks lived longer partly because of their regular, moderate drinking of wine.

Red wine is made by crushing and fermenting dark-colored, whole grapes.

There are many types of red wine, which vary in taste and color. Common varieties include Shiraz, Merlot, Cabernet sauvignon, Pinot noir and Zinfandel.

The alcohol content usually ranges from 12–15%.

According to a 2018 study, although notably there are no official recommendations around these benefits, drinking red wine in moderation has positive links with:

- **cardiovascular disease**
- **atherosclerosis**
- **hypertension**
- **certain types of cancer**
- **type 2 diabetes**
- **neurological disorders**
- **metabolic syndrome**

RESEARCH OBJECTIVES

1. **To experiment with different classification methods to see which yields the highest accuracy**

2. To determine which features are the most indicative of a good quality wine

RESEARCH METHODOLOGY

In this project Machine Learning techniques are used to determine dependency of wine quality on other variables and in wine quality predictions. Wine quality is predicted by comparing Machine Learning models : Decision Tree, Random Forest and Extra Tree Classifier.

For this project, I have used secondary data - I have utilized Kaggle's Red Wine Quality Dataset to anticipate whether quality of wine is good or bad.

ABOUT DATASET

The Wine Quality dataset contains information about various physicochemical properties of wines. The entire dataset is grouped into two categories: red wine and white wine. Each wine has a quality label associated with it. The label is in the range of 0 to 10.

FEATURES DESCRIPTION

1. Fixed acidity: It indicates the amount of tartaric acid in wine and is measured in g/dm3
2. Volatile acidity: It indicates the amount of acetic acid in the wine. It is measured in g/dm3.
3. Citric acid: It indicates the amount of citric acid in the wine. It is also measured in g/dm3
4. Residual sugar: It indicates the amount of sugar left in the wine after the fermentation process is done. It is also measured in g/dm3
5. Free sulfur dioxide: It measures the amount of sulfur dioxide (SO_2) in free form. It is also measured in g/dm3
6. Total sulfur dioxide: It measures the total amount of SO_2 in the wine. This chemical works as an antioxidant and antimicrobial agent.
7. Density: It indicates the density of the wine and is measured in g/dm3.
8. pH: It indicates the pH value of the wine. The range of value is between 0 to 14.0, which indicates very high acidity, and 14 indicates basic acidity.
9. Sulphates: It indicates the amount of potassium sulphate in the wine. It is also measured in g/dm3. 10.Alcohol: It indicates the alcohol content in the wine. 11.Quality: It indicates the quality of the wine, which is ranged from 1 to 10. Here, the higher the value is, the better the wine.

Import Required Libraries and Read the Dataset

In [46]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [24]:
```
df = pd.read_csv("winequality-red.csv")
```

In [81]:
```
#Understanding the data
#see the first five rows of the dataset
df.head()
```

Out[81]:

	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

DESCRIPTIVE STATISTICS

In [26]:
```
#Statistical Analysis:
df.describe()
```

Out[26]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alc
count	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.00
mean	8.319637	0.527821	0.270976	2.538806	0.087467	15.874922	46.467792	0.996747	3.311113	0.658149	10.42
std	1.741096	0.179060	0.194801	1.409928	0.047065	10.460157	32.895324	0.001887	0.154386	0.169507	1.06
min	4.600000	0.120000	0.000000	0.900000	0.012000	1.000000	6.000000	0.990070	2.740000	0.330000	8.40

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	al
25%	7.100000	0.390000	0.090000	1.900000	0.070000	7.000000	22.000000	0.995600	3.210000	0.550000	9.50
50%	7.900000	0.520000	0.260000	2.200000	0.079000	14.000000	38.000000	0.996750	3.310000	0.620000	10.20
75%	9.200000	0.640000	0.420000	2.600000	0.090000	21.000000	62.000000	0.997835	3.400000	0.730000	11.10
max	15.900000	1.580000	1.000000	15.500000	0.611000	72.000000	289.000000	1.003690	4.010000	2.000000	14.90

In [27]:

```
#datatype information:
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1599 entries, 0 to 1598
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   fixed acidity         1599 non-null   float64
 1   volatile acidity      1599 non-null   float64
 2   citric acid           1599 non-null   float64
 3   residual sugar        1599 non-null   float64
 4   chlorides             1599 non-null   float64
 5   free sulfur dioxide   1599 non-null   float64
 6   total sulfur dioxide  1599 non-null   float64
 7   density               1599 non-null   float64
 8   pH                    1599 non-null   float64
 9   sulphates             1599 non-null   float64
 10  alcohol               1599 non-null   float64
 11  quality               1599 non-null   int64  
dtypes: float64(11), int64(1)
memory usage: 150.0 KB
```

In [28]:

```
#Get the datatypes of each feature:
df.dtypes
```

Out[28]:

```
fixed acidity           float64
volatile acidity        float64
citric acid             float64
residual sugar          float64
chlorides               float64
free sulfur dioxide     float64
```

```
total sulfur dioxide    float64
density                 float64
pH                      float64
sulphates               float64
alcohol                 float64
quality                   int64
dtype: object
```

DATA PREPOCESSING

In [29]:
```
#check for missing values:
df.isnull().sum()
```

Out[29]:
```
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality                 0
dtype: int64
```

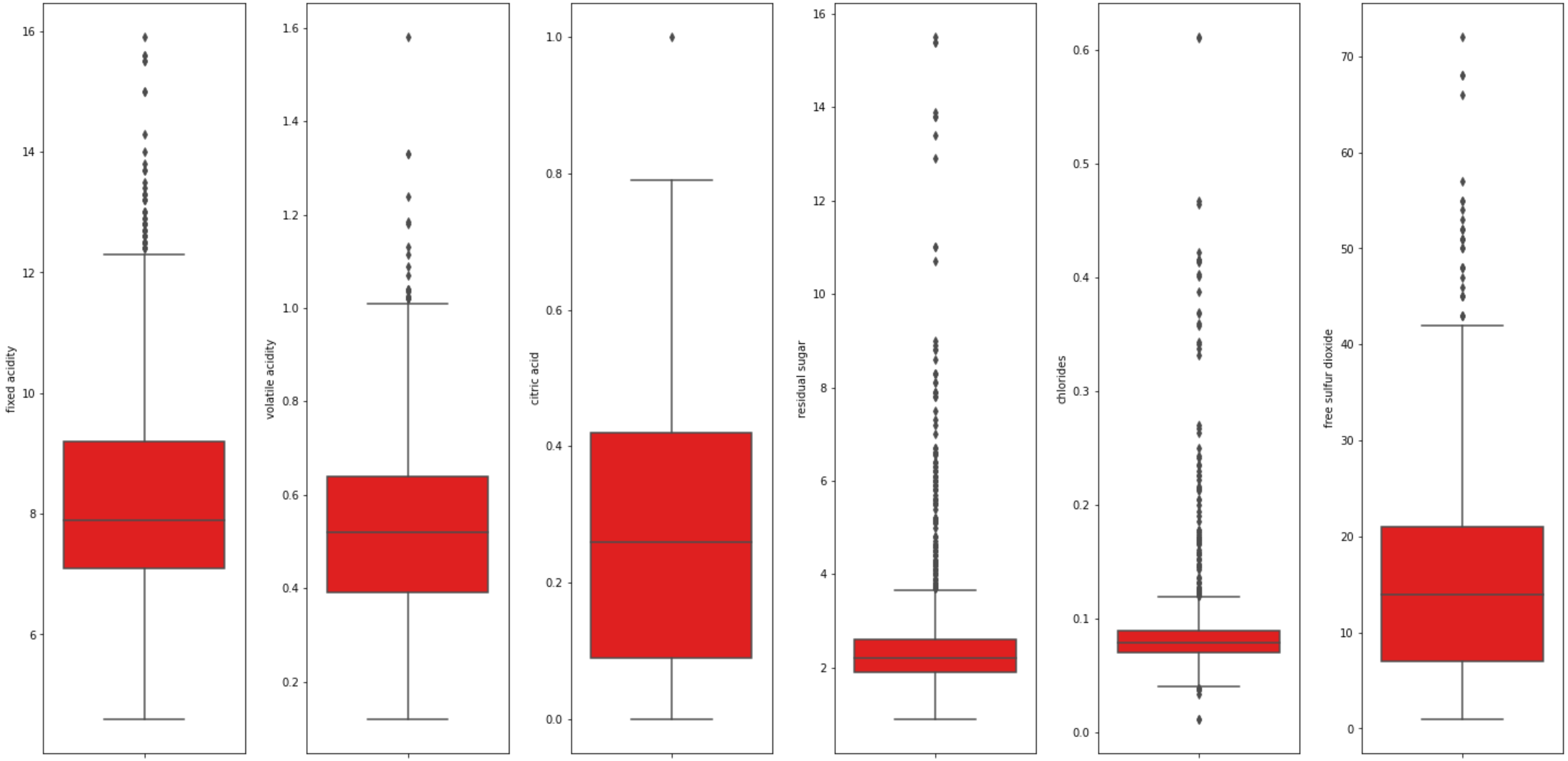
This is a very beginner-friendly dataset. we did not have to deal with any missing values, and there isn't much flexibility to conduct some feature engineering given these variables

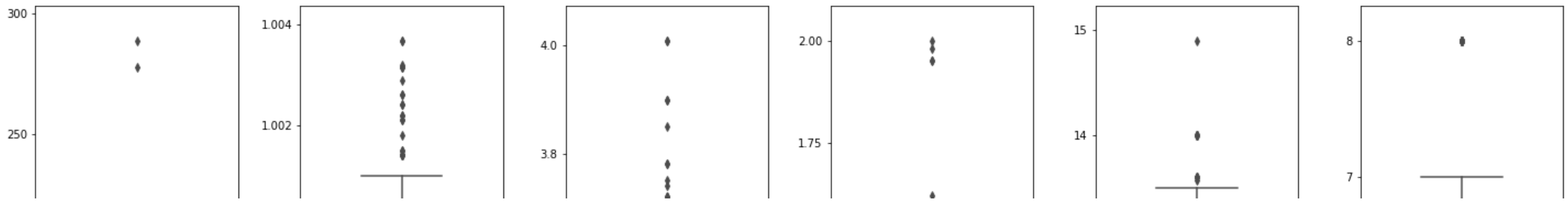
EXPLORATORY DATA ANALYSIS

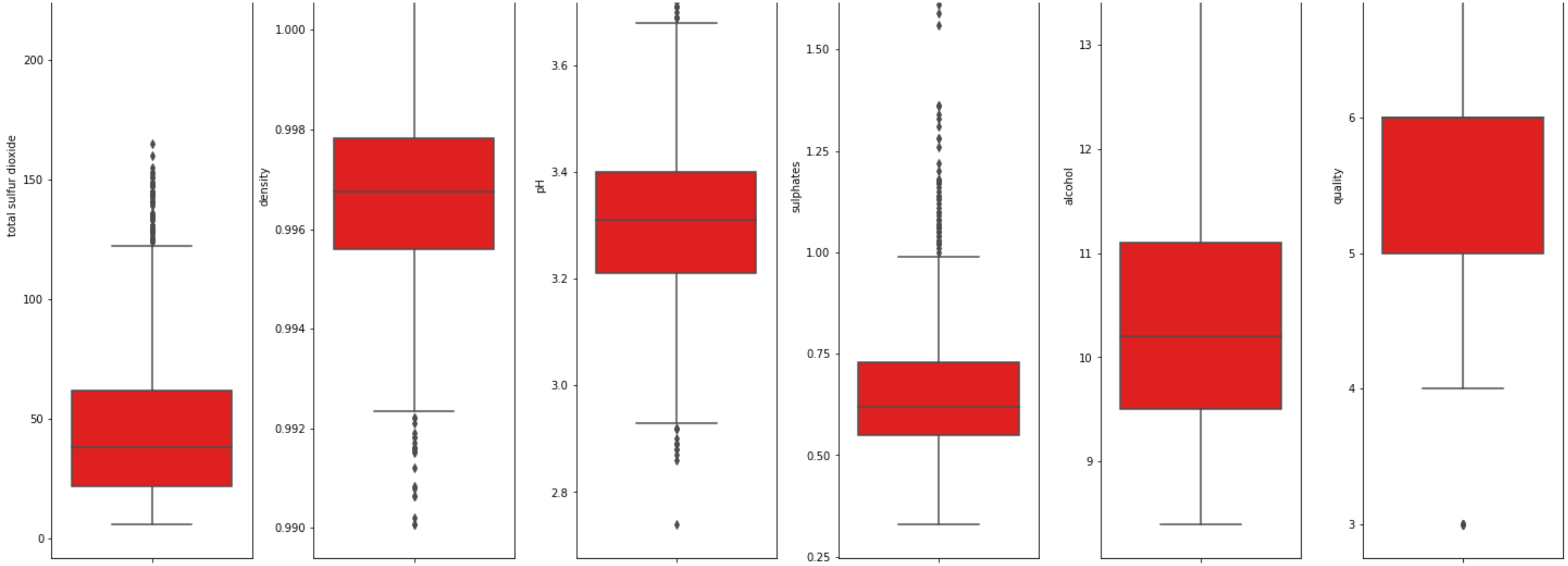
In [36]:
```
#to check whether data has outliers or not:


# create box plots
fig, ax = plt.subplots(ncols=6, nrows=2, figsize=(20,20))
index = 0
ax = ax.flatten()
```

```python
for color, value in df.items():
    sns.boxplot(y=color, data=df, color='r', ax=ax[index])
    index += 1
plt.tight_layout(pad=0.5, w_pad=0.7, h_pad=5.0)
```





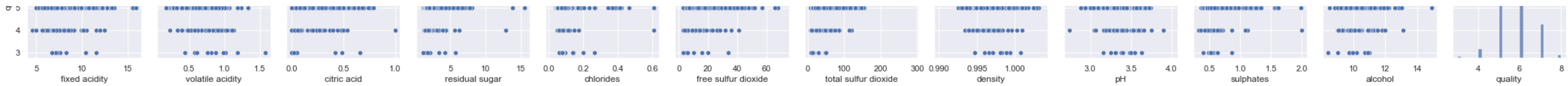


FIND CORRELATED COLUMNS

In [48]:
```
#Method 1
sns.pairplot(df)
```

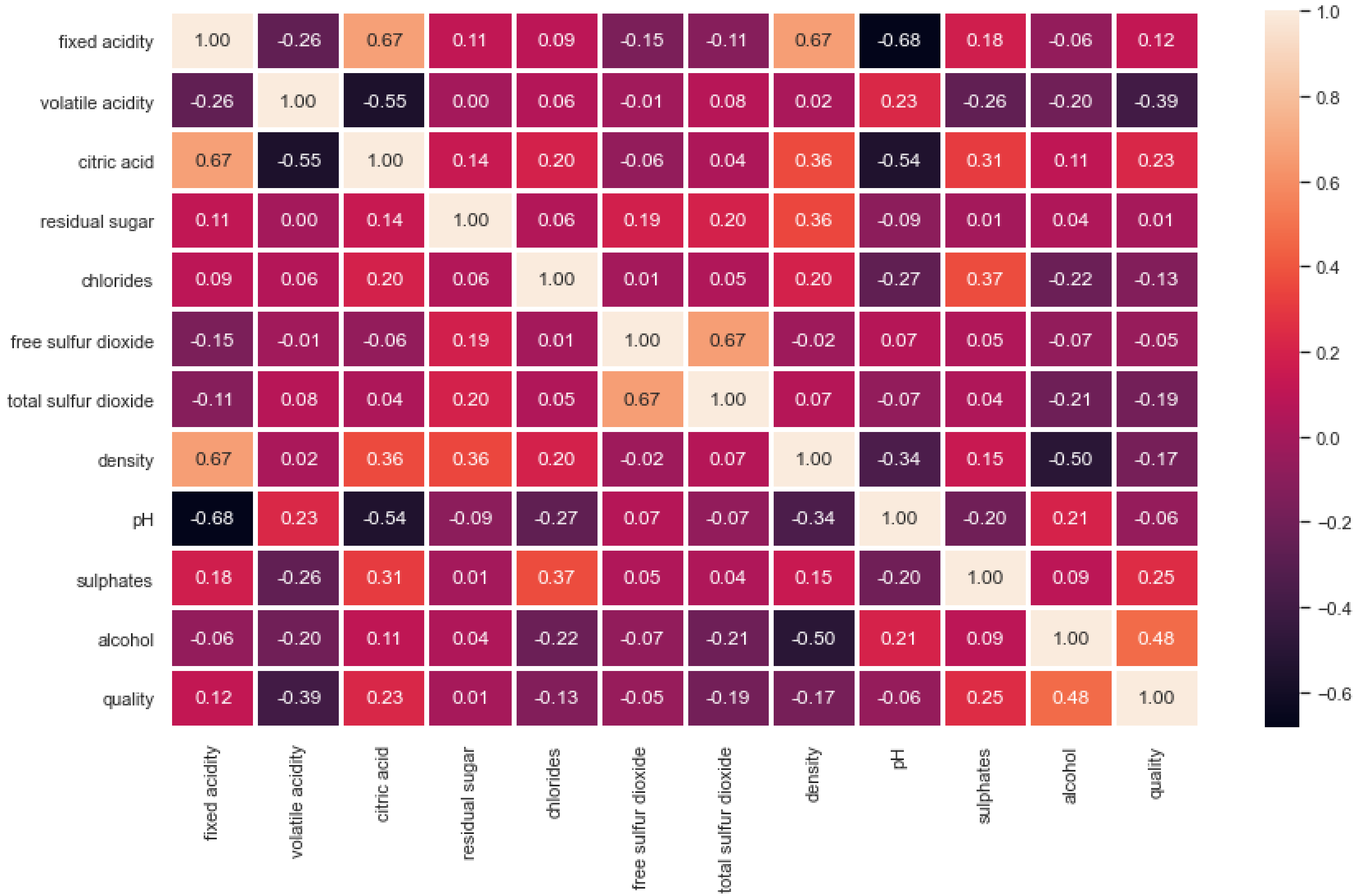
Out[48]: <seaborn.axisgrid.PairGrid at 0x1b332412c10>


citric acid
residual sugar
chlorides
free sulfur dioxide
total sulfur dioxide
density
pH
sulphates
alcohol
quality



In [49]:
```
#Method 2
sns.heatmap(df.corr(), annot=True, fmt='.2f', linewidths=2)
```

Out[49]: <AxesSubplot:>



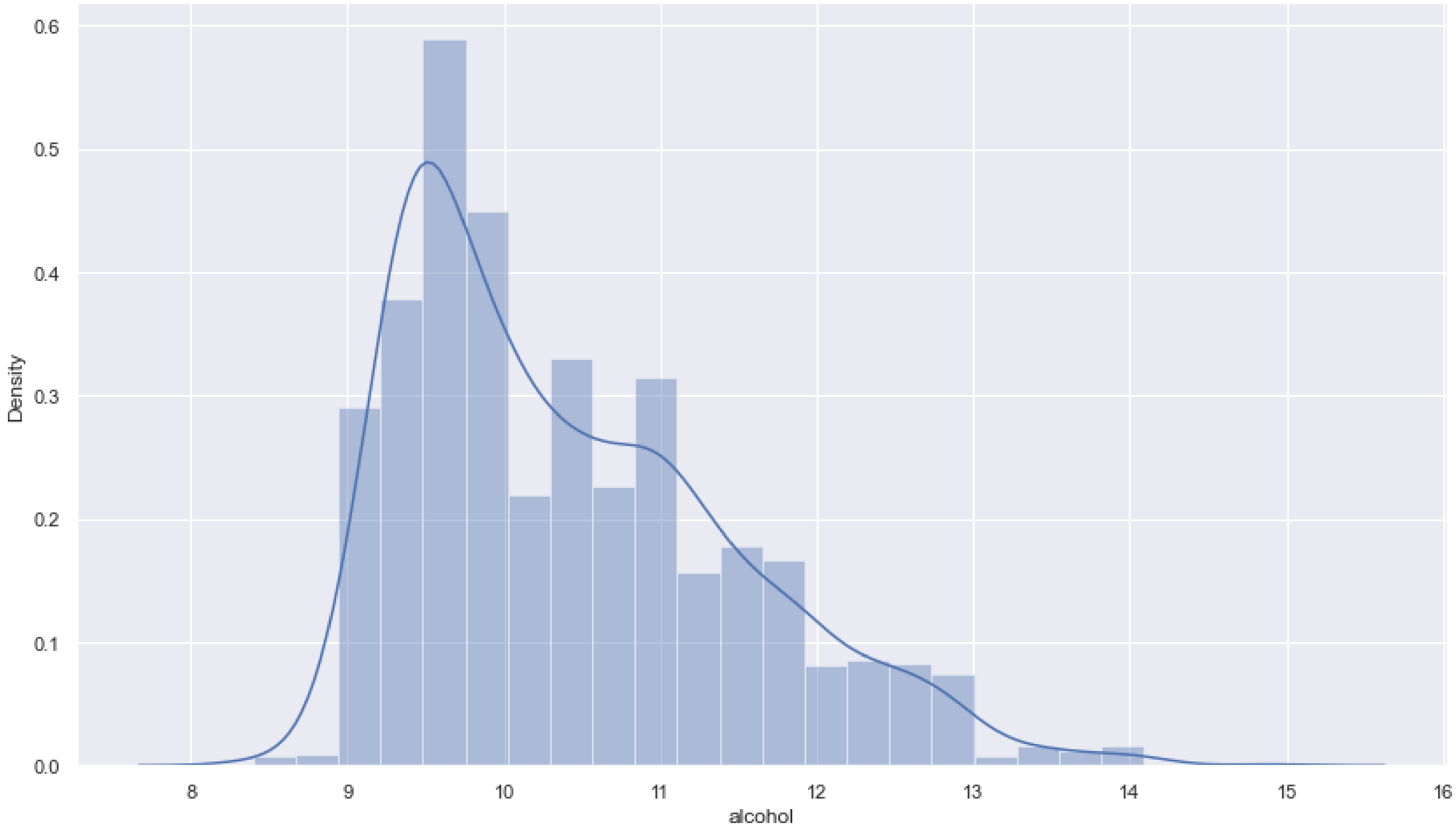
Insights From Above Figure:

1. **Alcohol is positively correlated with the quality of the red wine.**
2. **Alcohol has a weak positive correlation with the pH value.**

3. Citric acid and density have a strong positive correlation with fixed acidity.
4. pH has a negative correlation with density, fixed acidity, citric acid, and sulfates.

In [50]:
```
# to check how alcohol concentration is distributed with respect to the quality of the red wine.
sns.distplot(df['alcohol'])
```

Out[50]: <AxesSubplot:xlabel='alcohol', ylabel='Density'>

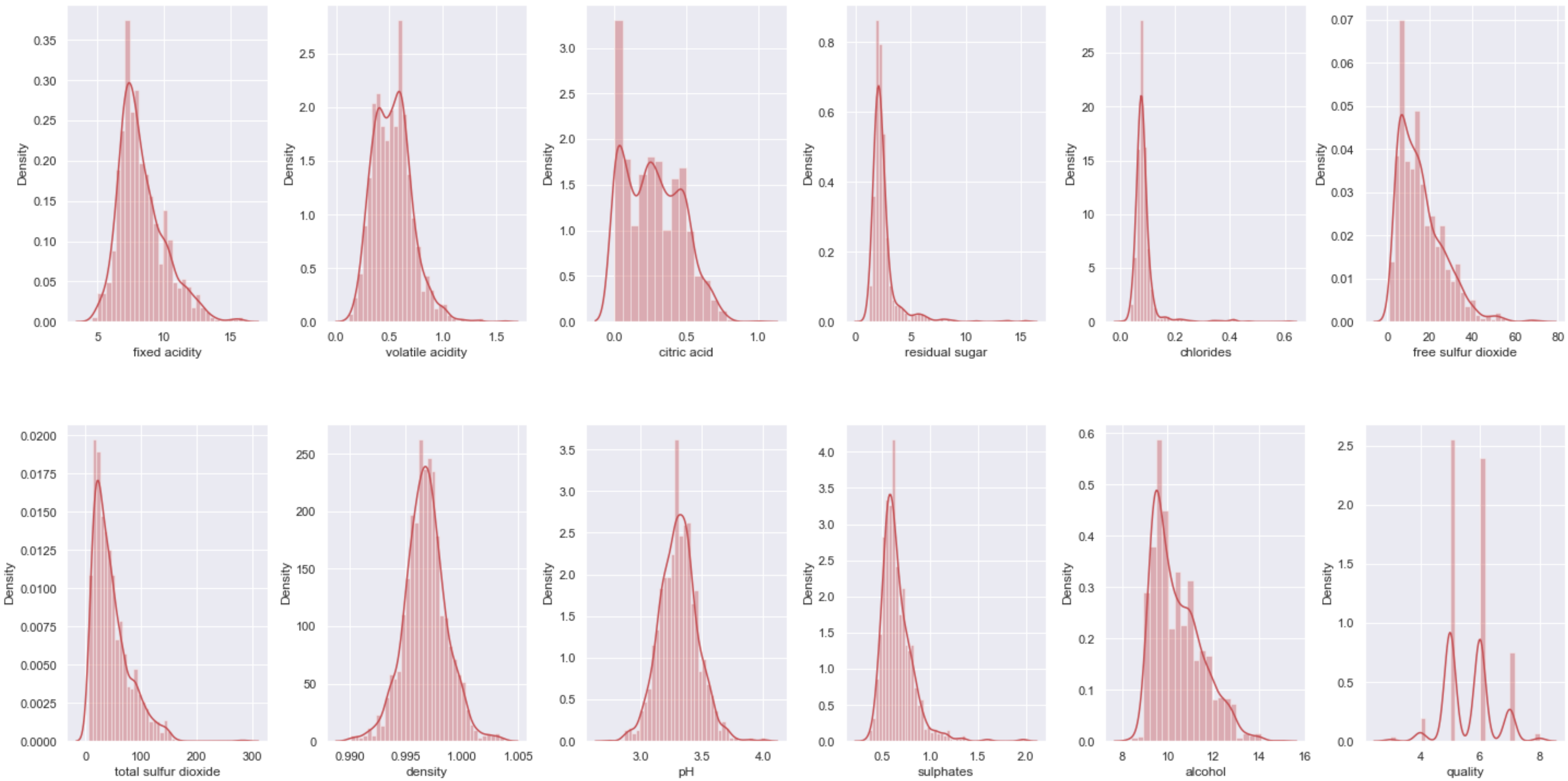


The alcohol distribution is positively skewed with the quality of the red wine.

In [51]:

```
#Dist plot of all features:
# create dist plot
fig, ax = plt.subplots(ncols=6, nrows=2, figsize=(20,10))
index = 0
ax = ax.flatten()

for col, value in df.items():
    sns.distplot(value, color='r', ax=ax[index])
    index += 1
plt.tight_layout(pad=0.5, w_pad=0.7, h_pad=5.0)
```



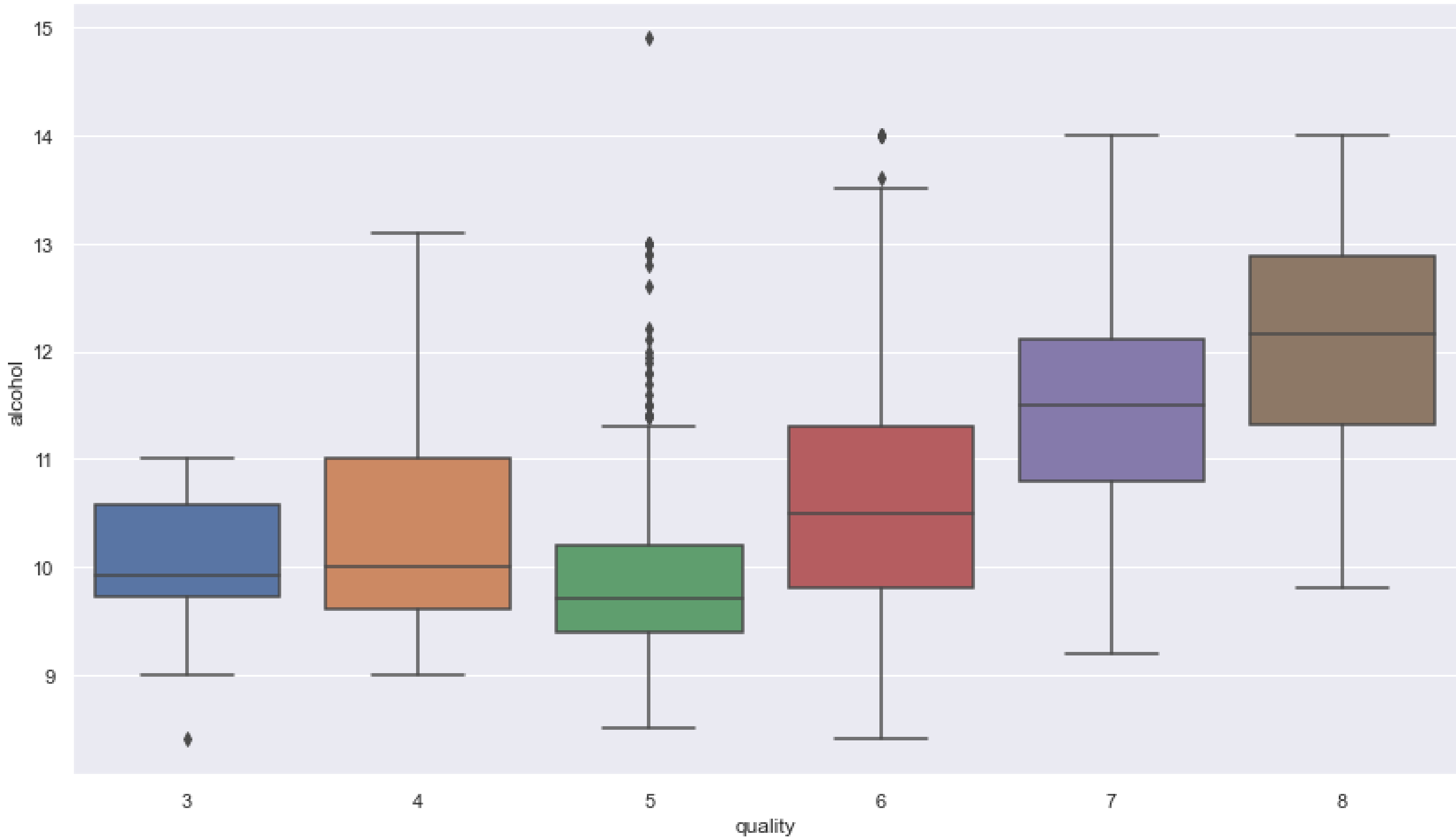
The above figures show the distribution of the features. Few of them are normally distributed where other are rightly skewed. The range

of each feature is also not huge.

ALCOHOL VS QUALITY

In [52]:
```
sns.boxplot(x='quality', y='alcohol', data = df)
```

Out[52]: <AxesSubplot:xlabel='quality', ylabel='alcohol'>

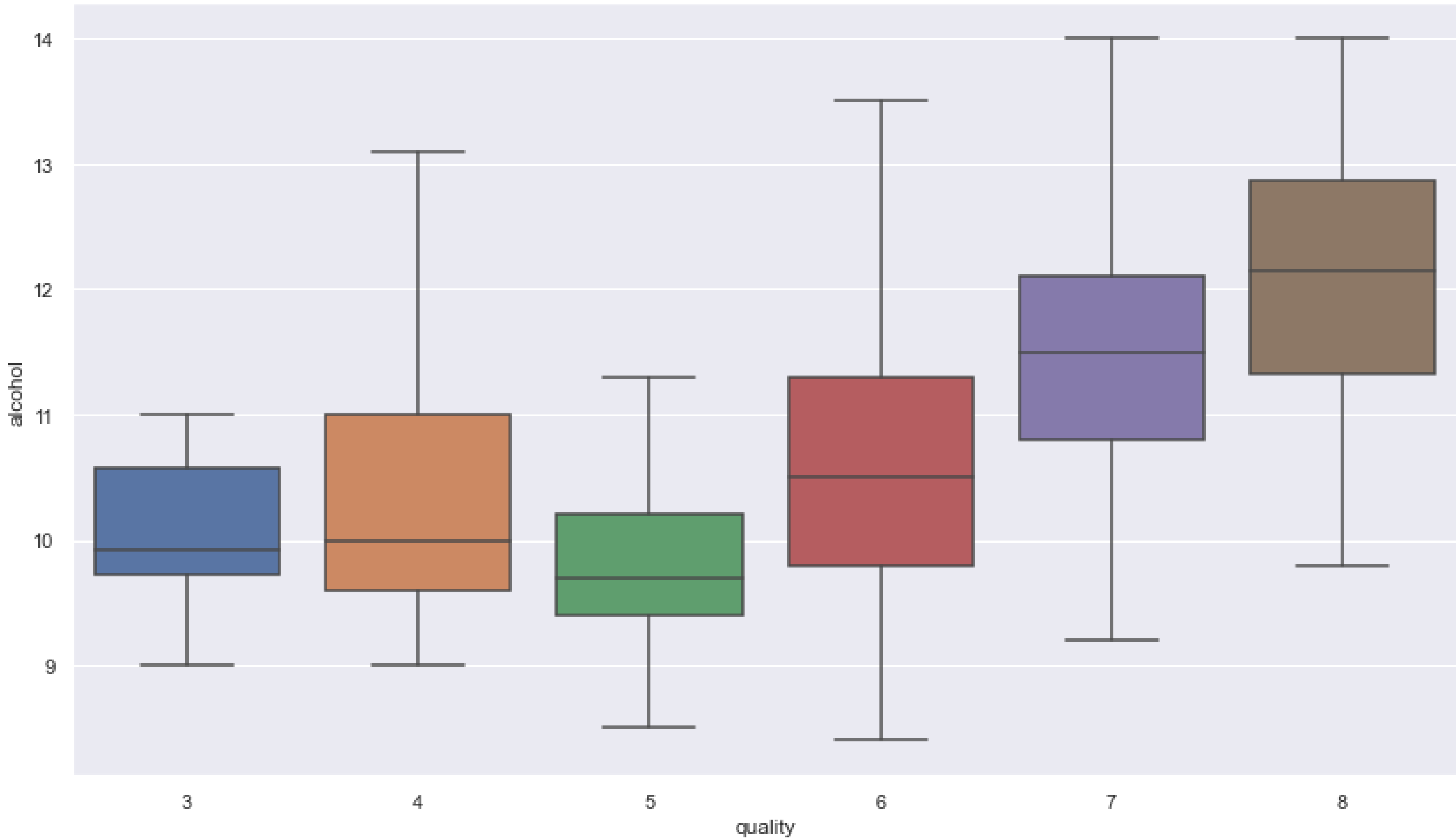


In above Figure - showing some dots outside of the graph. Those are outliers. Most of the outliers are around wine with quality 5 and 6.

We can remove the outliers by passing an argument, showoutliers=False

In [53]:
```
sns.boxplot(x='quality', y='alcohol', data = df, showfliers=False)
```

Out[53]: <AxesSubplot:xlabel='quality', ylabel='alcohol'>

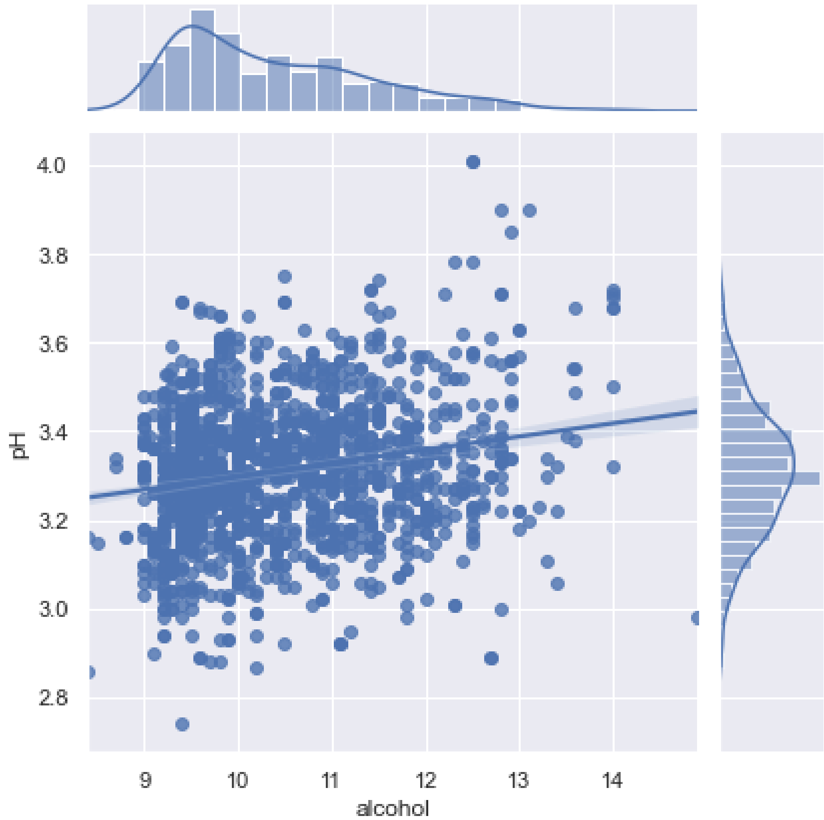


The higher the alcohol concentration is, the higher the quality of the wine.

ALCOHOL VS pH

In [54]:
```
sns.jointplot(x='alcohol',y='pH',data=df, kind='reg')
```

Out[54]: <seaborn.axisgrid.JointGrid at 0x1b333c62e50>



This Figure shows that alcohol is weakly positively related to the pH values. Moreover, the regression line is depicted in the figure, illustrating the correlation between them.

We can quantify the correlation using Pearson regression from scipy.stats

In [55]:
```
from scipy.stats import pearsonr
def get_correlation(column1, column2, df):
    pearson_corr, p_value = pearsonr(df[column1], df[column2])
```

```
    print("Correlation between {} and {} is {}".format(column1,column2, pearson_corr))
    print("P-value of this correlation is {}".format(p_value))
```

In [56]:
```
get_correlation('alcohol','pH', df)
```

```
Correlation between alcohol and pH is 0.2056325085054989
P-value of this correlation is 9.964497741462162e-17
```

CONVERT TO A CLASSIFICATION MODEL

In [83]:
```
# Create Classification version of target variable
df['goodquality'] = [1 if x >= 7 else 0 for x in df['quality']]
# Separate feature variables and target variable
X = df.drop(['quality','goodquality'], axis = 1)
y = df['goodquality']
```

Going back to our objective, we wanted to compare the effectiveness of different classification techniques, so we needed to change the output variable to a binary output. For this problem, I defined a bottle of wine as 'good quality' if it had a quality score of 7 or higher, and if it had a score of less than 7, it was deemed 'bad quality'. Once we converted the output variable to a binary output, we separated our feature variables (X) and the target variable (y) into separate dataframes

PROPORTION OF GOOD VS BAD WINES

In [84]:
```
# See proportion of good vs bad wines
df['goodquality'].value_counts()
```

Out[84]:
```
0    1382
1     217
Name: goodquality, dtype: int64
```

We wanted to make sure that there was a reasonable number of good quality wines. Based on the results below, it seemed like a fair enough number.

MODEL DEVELOPMENT AND EVALUATION

In [57]:
```
#Import Model libraries:

from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split,cross_validate
from sklearn.metrics import accuracy_score,precision_score,recall_score,f1_score
```

HANDLING IMBALANCED DATA

In [59]:
```
X = df.drop('quality', axis=1)
y = df['quality']
```

In [72]:
```
# Splitting the data
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=.25, random_state=0)
```

MODELLING

For this project, we wanted to compare different machine learning models: decision trees, random forests and Extra trees classifier. For the purpose of this project, We wanted to compare these models by their accuracy.

MODEL-1 : DECISION TREE

Decision trees are a popular model, used in operations research, strategic planning, and machine learning. Decision trees are intuitive and easy to build but fall short when it comes to accuracy.

In [76]:
```
model = DecisionTreeClassifier()
classify(model, X, y)
```

```
Accuracy: 58.5
CV Score: 48.27899686520376
```

MODEL-2 : RANDOM FOREST

Random forests are an ensemble learning technique that builds off of decision trees. Random forests involve creating multiple decision trees using bootstrapped datasets of the original data and randomly selecting a subset of variables at each step of the decision tree. The model then selects the mode of all of the predictions of each decision tree.

In [89]:
```
model = RandomForestClassifier()
classify(model, X, y)
```

```
Accuracy: 90.25
CV Score: 86.74333855799374
```

MODEL-3 : EXTRA TREES CLASSIFIER

Extremely Randomized Trees Classifier(Extra Trees Classifier) is a type of ensemble learning technique which aggregates the results of multiple de-correlated decision trees collected in a “forest” to output it's classification result

In [79]:
```
from sklearn.ensemble import ExtraTreesClassifier
model = ExtraTreesClassifier()
classify(model, X, y)
```

```
Accuracy: 68.25
CV Score: 57.412225705329156
```

CONCLUSION

I have used the Wine Quality dataset to perform EDA. I discussed how I can perform EDA techniques such as data loading, data wrangling, data transformation, correlation between variables, regression analysis, and building classical ML models based on the datasets.

In [ ]:
```

```