** PROJECT DAY - 24**

[Wine-Quality Prediction Using ML]

Importing the Dependencies

```
In [144]:
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
```

Data Collection

```
In [144]:
# loading the dataset to a Pandas DataFrame
wine_dataset = pd.read_csv('winequality.csv')
```

```
In [146]:
wine_dataset.head()
```

Out[146]:

	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

```
In [148]:
# number of rows & columns in the dataset
wine_dataset.shape
```

Out[148]: (1599, 12)

```
In [150]:
# checking for missing values
wine_dataset.isnull().sum()
```

Out[150]:
```
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
```

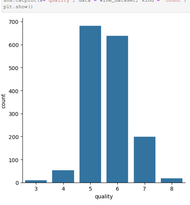
Data Analysis and Visulaization

```
In [152]:
# statistical measures of the dataset
wine_dataset.describe()
```

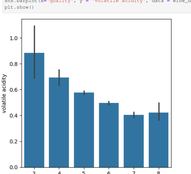
Out[153]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
count	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000
mean	8.319637	0.527821	0.270976	2.538806	0.087467	15.874922	46.467792	0.996747	3.311113	0.658149	10.422983	5.636023
std	1.741096	0.179060	0.194801	1.409928	0.047065	10.460157	32.895324	0.001887	0.154386	0.169507	1.065666	0.807569
min	4.600000	0.120000	0.000000	0.900000	0.012000	1.000000	6.000000	0.990070	2.740000	0.330000	8.400000	3.000000
25%	7.100000	0.390000	0.090000	1.900000	0.070000	7.000000	22.000000	0.995600	3.210000	0.550000	9.500000	5.000000
50%	7.900000	0.520000	0.260000	2.200000	0.079000	14.000000	38.000000	0.996750	3.310000	0.620000	10.200000	6.000000
75%	9.200000	0.640000	0.420000	2.600000	0.090000	21.000000	62.000000	0.997835	3.400000	0.730000	11.100000	6.000000
max	15.900000	1.580000	1.000000	15.500000	0.611000	72.000000	289.000000	1.003690	4.010000	2.000000	14.900000	8.000000

```
In [155]:
# number of values for each quality
sns.catplot(x="quality", data = wine_dataset, kind = 'count')
plot.show()
```

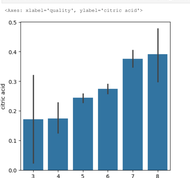



```
In [157]:
# volatile acidity vs Quality
plot = plt.figure(figsize=(5,5))
sns.barplot(x="quality", y = "volatile acidity", data = wine_dataset)
plot.show()
```




```
In [159]:
# citric acid vs Quality
plot = plt.figure(figsize=(5,5))
sns.barplot(x="quality", y = "citric acid", data = wine_dataset)
```

Out[159]: <Axes: xlabel="quality", ylabel="citric acid">

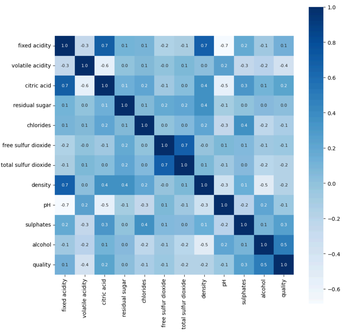


Correlation

1.Positive Correlation
2.Negative Correlation

```
In [161]:
correlation = wine_dataset.corr()
```

```
In [165]:
# constructing a heatmap to understand the correlation between the columns
plt.figure(figsize=(15,10))
sns.heatmap(correlation, cbar=True, square=True, fmt = '.1f', annot = True, annot_kws={'size':8}, cmap = 'Blues')
plot.show()
```



Data Preprocessing

```
In [168]:
# separate the data and Label
X = wine_dataset.drop('quality',axis=1)
```

```
In [170]:
X
```

Out[170]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol
0	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	0.56	9.4
1	7.8	0.880	0.00	2.6	0.098	25.0	67.0	0.99680	3.20	0.68	9.8
2	7.8	0.760	0.04	2.3	0.092	15.0	54.0	0.99700	3.26	0.65	9.8
3	11.2	0.280	0.56	1.9	0.075	17.0	60.0	0.99800	3.16	0.58	9.8
4	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	0.56	9.4
...	...	...	...	...	...	...	...	...	...	...	...
1594	6.2	0.600	0.08	2.0	0.090	32.0	44.0	0.99490	3.45	0.58	10.5
1595	5.9	0.550	0.10	2.2	0.062	39.0	51.0	0.99512	3.52	0.76	11.2
1596	6.3	0.510	0.13	2.3	0.076	29.0	40.0	0.99574	3.42	0.75	11.0
1597	5.9	0.645	0.12	2.0	0.075	32.0	44.0	0.99547	3.57	0.71	10.2
1598	6.0	0.310	0.47	3.6	0.067	18.0	42.0	0.99549	3.39	0.66	11.0

1599 rows × 11 columns

Label Binarizaton

```
In [173]:
Y = wine_dataset['quality'].apply(lambda y_value: 1 if y_value>=7 else 0)
```

```
In [175]:
Y
```

Out[175]:
```
0       0
1       0
2       0
3       0
4       0
       ..
1594    0
1595    0
1596    0
1597    0
1598    0
Name: quality, Length: 1599, dtype: int64
```

Train & Test Split

```
In [178]:
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=3)
```

```
In [180]:
Y.shape, Y_train.shape, Y_test.shape
```

Out[180]: ((1599,), (1279,), (320,))

Model Training:

Random Forest Classifier

```
In [184]:
model = RandomForestClassifier()
```

```
In [186]:
model.fit(X_train, Y_train)
```

Out[186]:
```
RandomForestClassifier()
```

Model Evaluation

Accuracy Score

```
In [190]:
# accuracy on test data
X_test_prediction = model.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
```

```
In [192]:
print('Accuracy : ', test_data_accuracy)
```

```
Accuracy :  0.934375
```

Building a Predictive System

```
In [195]:
# Feature names (must be SAME as training)
feature_names = [
    'fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
    'chlorides', 'free sulfur dioxide', 'total sulfur dioxide',
    'density', 'pH', 'sulphates', 'alcohol'
]

input_data = (7.5, 0.5, 0.36, 6.1, 0.071, 17.0, 102.0, 0.9978, 3.35, 0.8, 10.5)

# Convert to DataFrame
input_df = pd.DataFrame([input_data], columns=feature_names)

prediction = model.predict(input_df)

if prediction[0] == 1:
    print('Good Quality Wine ')
else:
    print('Bad Quality Wine ')
```

```
Bad Quality Wine
```

** END - PROJECT**