Wine Quality Prediction

> About This Project

1. **Importing Dependencies**: The necessary libraries like NumPy, Pandas, Matplotlib, Seaborn, and modules from scikit-learn are imported. These libraries are fundamental for data manipulation, visualization, and machine learning.
2. **Loading the Dataset**: The wine quality dataset is loaded into a Pandas DataFrame. This dataset contains information about various attributes of wines and their corresponding quality ratings.
3. **Data Exploration and Preprocessing**: The script then performs exploratory data analysis (EDA) tasks like checking the shape of the dataset, displaying the first few rows, and checking for missing values. Fortunately, there are no missing values in this dataset.
4. **Statistical Summary**: It calculates and displays the summary statistics of the dataset.
5. **Data Visualization**: Visualizations like bar plots and heatmaps are created to understand the relationship between different attributes and wine quality.
6. **Data Preparation**: The data is divided into features (X) and the target variable (Y). The target variable is binary, indicating whether a wine is of good quality (7 or higher) or not.
7. **Train-Test Split**: The data is split into training and testing sets for model evaluation.
8. **Model Building**: A Random Forest Classifier is chosen for this classification task. The model is trained on the training data.
9. **Model Evaluation**: The model's accuracy is calculated on the test data, which comes out to be approximately 93.13%.
10. **Prediction**: An input data point is provided, and the model predicts the quality. In this case, the model predicts it to be of lower quality.

ipeline for a classification problem. It starts from data loading, preprocessing, and visualization, followed by model building, evaluation, and prediction. The Random Forest Classifier, chosen for this task, yields a high accuracy of over 93%. This indicates that the features used in the dataset are fairly informative for predicting wine quality.

Importing the Dependencies

```
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
```

```
# loading the dataset to a Pandas DataFrame
wine_dataset = pd.read_csv('/content/winequality-red.csv')
```

```
# number of rows & columns in the dataset
wine_dataset.shape
```

```
(1599, 12)
```

```
# first 5 rows of the dataset
wine_dataset.head()
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8

```
# checking for missing values
wine_dataset.isnull().sum()
```

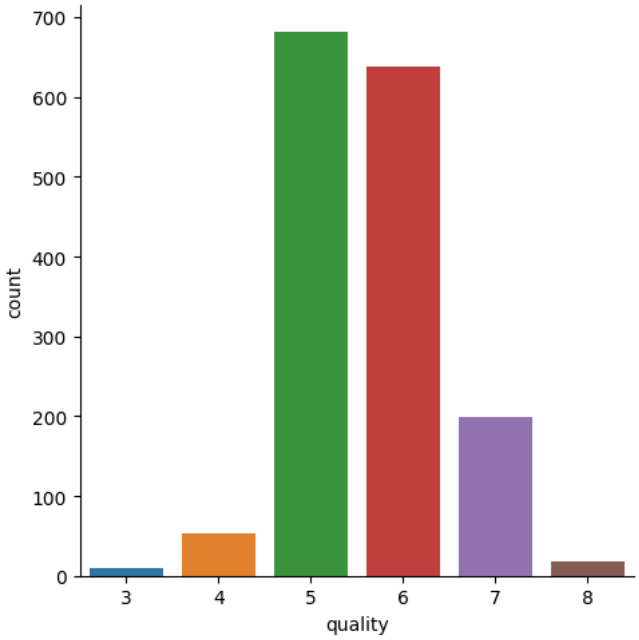
```
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality   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

```
# statistical measures of the dataset
wine_dataset.describe()
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	dens
count	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000
mean	8.319637	0.527821	0.270976	2.538806	0.087467	15.874922	46.467792	0.996
std	1.741096	0.179060	0.194801	1.409928	0.047065	10.460157	32.895324	0.001
min	4.600000	0.120000	0.000000	0.900000	0.012000	1.000000	6.000000	0.990
25%	7.100000	0.390000	0.090000	1.900000	0.070000	7.000000	22.000000	0.995
50%	7.900000	0.520000	0.260000	2.200000	0.079000	14.000000	38.000000	0.996
75%	9.200000	0.640000	0.420000	2.600000	0.090000	21.000000	62.000000	0.997
max	15.900000	1.580000	1.000000	15.500000	0.611000	72.000000	289.000000	1.003

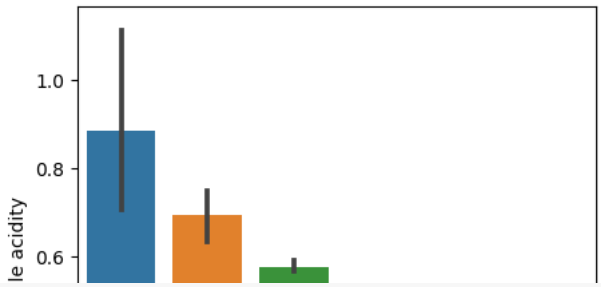

```
_dataset, kind = 'count')
```

```
<seaborn.axisgrid.FacetGrid at 0x7fa20f56afb0>
```



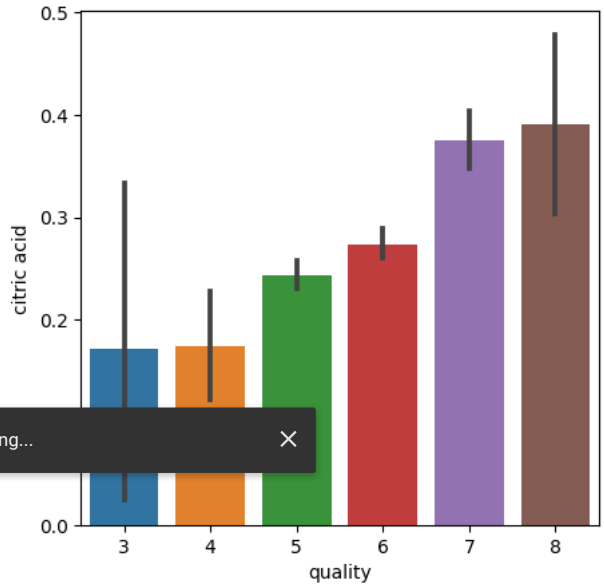

```
# volatile acidity vs Quality
plot = plt.figure(figsize=(5,5))
sns.barplot(x='quality', y = 'volatile acidity', data = wine_dataset)
```

```
<Axes: xlabel='quality', ylabel='volatile acidity'>
```




```
# citric acid vs Quality
plot = plt.figure(figsize=(5,5))
sns.barplot(x='quality', y = 'citric acid', data = wine_dataset)
```

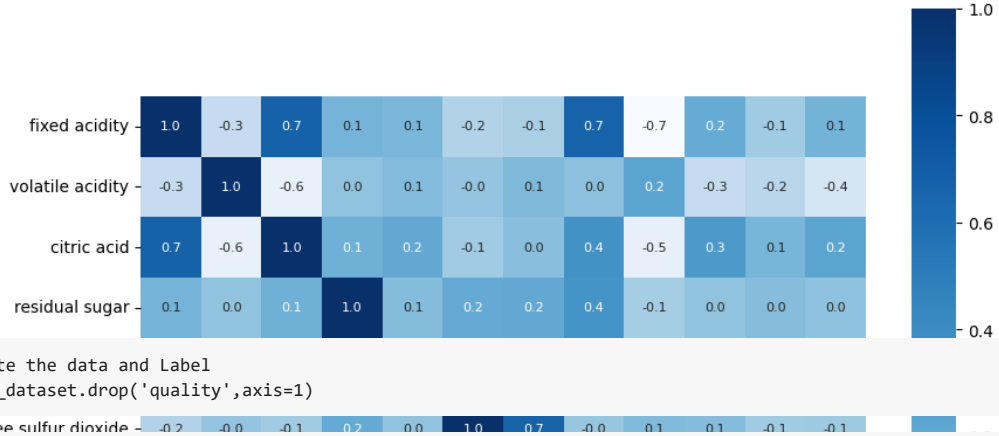
```
<Axes: xlabel='quality', ylabel='citric acid'>
```




```
correlation = wine_dataset.corr()
```

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

<Axes: >




```
# separate the data and Label
X = wine_dataset.drop('quality',axis=1)
```



```
print(X)
```

```
      fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0               7.4             0.700         0.00             1.9      0.076
1               7.8             0.880         0.00             2.6      0.098
2               7.8             0.760         0.04             2.3      0.092
3              11.2             0.280         0.56             1.9      0.075
4               7.4             0.700         0.00             1.9      0.076
...             ...               ...          ...             ...        ...
1594            6.2             0.600         0.08             2.0      0.090
1595            5.9             0.550         0.10             2.2      0.062
1596            6.3             0.510         0.13             2.3      0.076
1597            5.9             0.645         0.12             2.0      0.075
1598            6.0             0.310         0.47             3.6      0.067

      free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
0                    11.0                  34.0  0.99780  3.51       0.56
                     25.0                  67.0  0.99680  3.20       0.68
                     15.0                  54.0  0.99700  3.26       0.65
3                    17.0                  60.0  0.99800  3.16       0.58
4                    11.0                  34.0  0.99780  3.51       0.56
...                   ...                   ...      ...   ...        ...
1594                 32.0                  44.0  0.99490  3.45       0.58
1595                 39.0                  51.0  0.99512  3.52       0.76
1596                 29.0                  40.0  0.99574  3.42       0.75
1597                 32.0                  44.0  0.99547  3.57       0.71
1598                 18.0                  42.0  0.99549  3.39       0.66

      alcohol
0         9.4
1         9.8
2         9.8
3         9.8
4         9.4
...       ...
1594     10.5
1595     11.2
1596     11.0
1597     10.2
1598     11.0

[1599 rows x 11 columns]
```



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

```
print(Y)
```

```
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
```

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=3)
```

```
print(Y.shape, Y_train.shape, Y_test.shape)
```

```
(1599,) (1279,) (320,)
```

```
model = RandomForestClassifier()
```

```
model.fit(X_train, Y_train)
```

```
RandomForestClassifier()
```

```
# accuracy on test data
X_test_prediction = model.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
```

```
print('Accuracy : ', test_data_accuracy)
```

```
Accuracy :  0.93125
```

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

# changing the input data to a numpy array
input_data_as_numpy_array = np.asarray(input_data)

# reshape the data as we are predicting the label for only one instance
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

prediction = model.predict(input_data_reshaped)
print(prediction)

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

Saving...

```
Bad Quality Wine
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but RandomForestClas
  warnings.warn(
```

0s completed at 6:37 PM