wine-quality-prediction

July 6, 2023

1 Importing Dependencies

[2]:
```
#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
```

2 Data Collection

[3]:
```
# loading the dataset to a Pandas dataframe
df = pd.read_csv('winequality-red.csv')
```

[4]:
```
#no of rows and columns
df.shape
```

[4]:
```
(1599, 12)
```

[ ]:
```
# first 5 rows of the dataset
```

[5]:
```
df.head()
```

[5]:
```
   fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0            7.4              0.70         0.00             1.9      0.076
1            7.8              0.88         0.00             2.6      0.098
2            7.8              0.76         0.04             2.3      0.092
3           11.2              0.28         0.56             1.9      0.075
4            7.4              0.70         0.00             1.9      0.076

   free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
0                 11.0                  34.0   0.9978  3.51       0.56
1                 25.0                  67.0   0.9968  3.20       0.68
2                 15.0                  54.0   0.9970  3.26       0.65
```

```
3                 17.0                      60.0   0.9980  3.16        0.58
4                 11.0                      34.0   0.9978  3.51        0.56

   alcohol  quality
0      9.4        5
1      9.8        5
2      9.8        5
3      9.8        6
4      9.4        5
```

[6]:
```
#Checking for missing values
df.isnull().sum()
```

[6]:
```
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
```

3 Data Analysis and Visualization

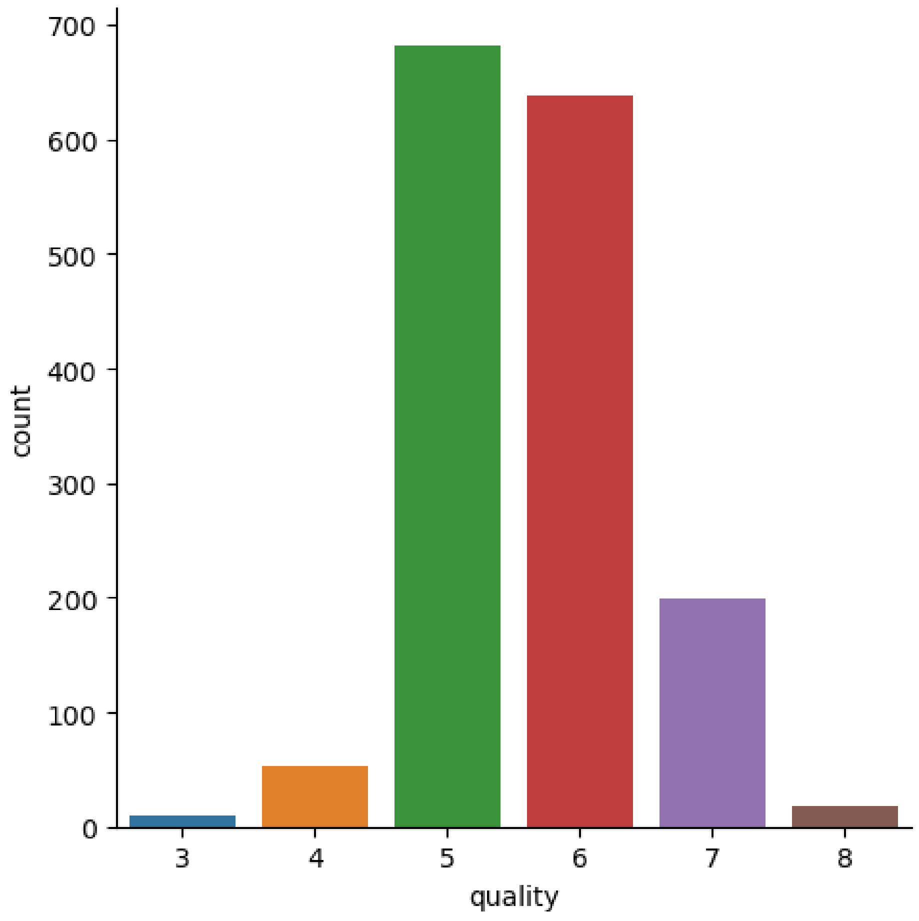
[7]:
```
#Statistical measures of the dataset
df.describe()
```

[7]:
```
       fixed acidity  volatile acidity  citric acid  residual sugar  \
count    1599.000000       1599.000000  1599.000000     1599.000000
mean        8.319637          0.527821     0.270976        2.538806
std         1.741096          0.179060     0.194801        1.409928
min         4.600000          0.120000     0.000000        0.900000
25%         7.100000          0.390000     0.090000        1.900000
50%         7.900000          0.520000     0.260000        2.200000
75%         9.200000          0.640000     0.420000        2.600000
max        15.900000          1.580000     1.000000       15.500000

         chlorides  free sulfur dioxide  total sulfur dioxide      density  \
count  1599.000000          1599.000000           1599.000000  1599.000000
mean      0.087467            15.874922             46.467792     0.996747
std       0.047065            10.460157             32.895324     0.001887
```

```
min       0.012000            1.000000            6.000000     0.990070
25%       0.070000            7.000000           22.000000     0.995600
50%       0.079000           14.000000           38.000000     0.996750
75%       0.090000           21.000000           62.000000     0.997835
max       0.611000           72.000000          289.000000     1.003690

                pH    sulphates      alcohol      quality
count  1599.000000  1599.000000  1599.000000  1599.000000
mean      3.311113     0.658149    10.422983     5.636023
std       0.154386     0.169507     1.065668     0.807569
min       2.740000     0.330000     8.400000     3.000000
25%       3.210000     0.550000     9.500000     5.000000
50%       3.310000     0.620000    10.200000     6.000000
75%       3.400000     0.730000    11.100000     6.000000
max       4.010000     2.000000    14.900000     8.000000
```

[8]:
```
#no. of values for each quality
sns.catplot(x= 'quality',data= df, kind= 'count')
```

[8]:
```
<seaborn.axisgrid.FacetGrid at 0x1e0166a1490>
```

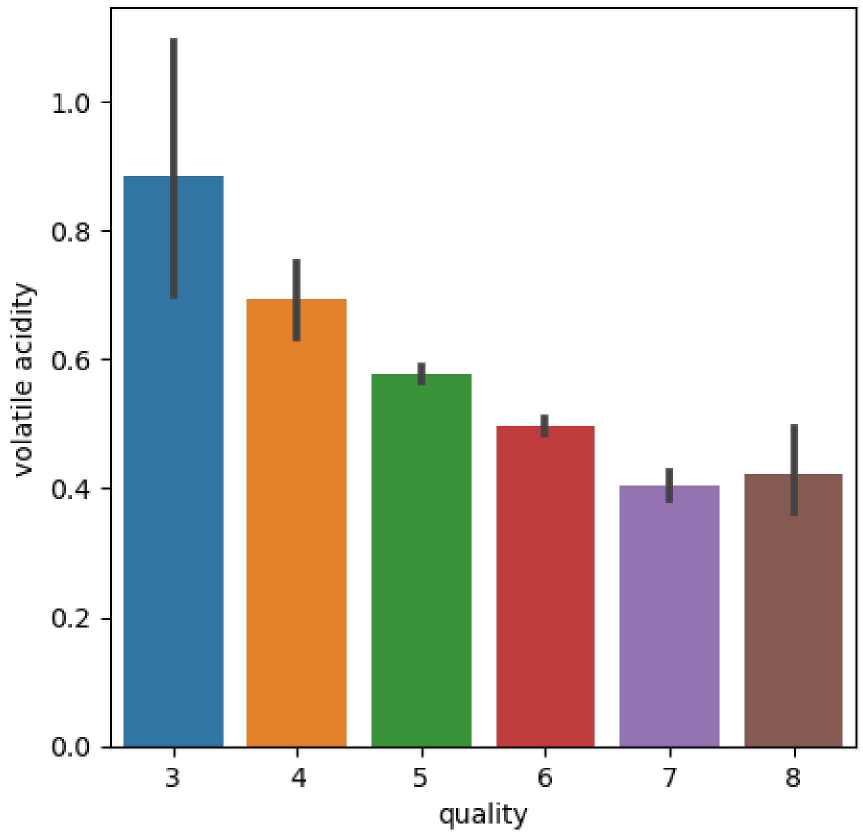


[10]:
```
#volatile acidity vs quality
plot = plt.figure(figsize = (5,5))
sns.barplot(x= 'quality',y='volatile acidity',data = df)
```

[10]:
```
<AxesSubplot:xlabel='quality', ylabel='volatile acidity'>
```

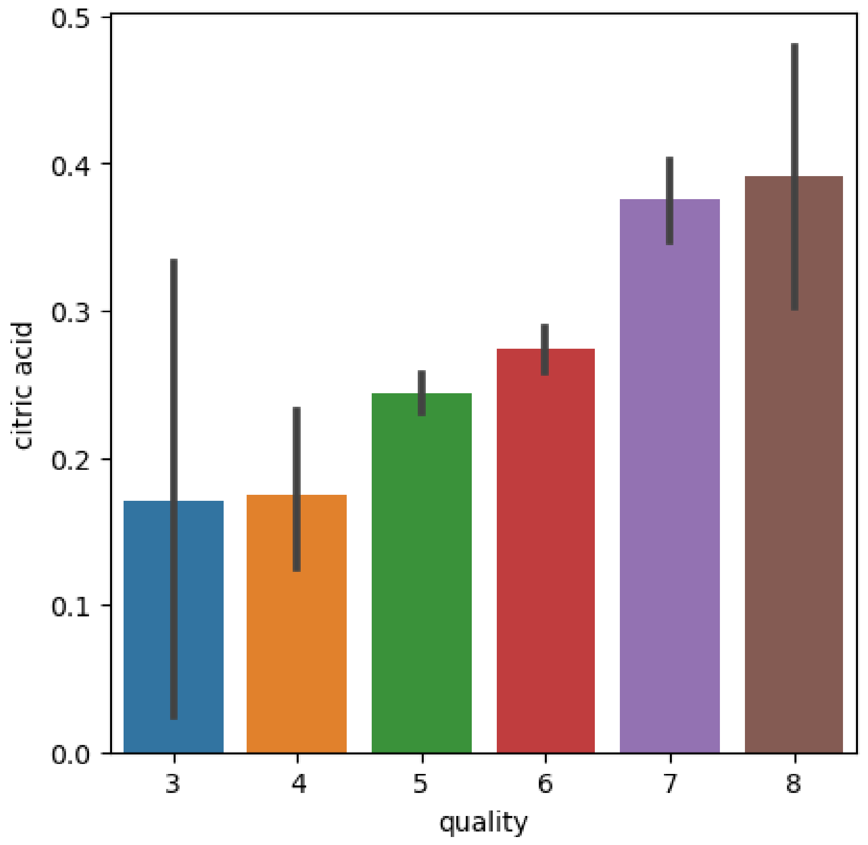
[11]:
```
plot
```

[11]:



[12]:
```
#citric acid vs quality
plot = plt.figure(figsize = (5,5))
sns.barplot(x= 'quality',y='citric acid',data = df)
plot
```

[12]:



4 Correlation

1. Positive correlation
2. Negative correlation

[17]:
```
correlation = df.corr()
```

[18]:
```
correlation
```

[18]:
```
                      fixed acidity  volatile acidity  citric acid  \
fixed acidity              1.000000         -0.256131     0.671703
volatile acidity          -0.256131          1.000000    -0.552496
citric acid                0.671703         -0.552496     1.000000
residual sugar             0.114777          0.001918     0.143577
chlorides                  0.093705          0.061298     0.203823
free sulfur dioxide       -0.153794         -0.010504    -0.060978
total sulfur dioxide      -0.113181          0.076470     0.035533
density                    0.668047          0.022026     0.364947
pH                        -0.682978          0.234937    -0.541904
```

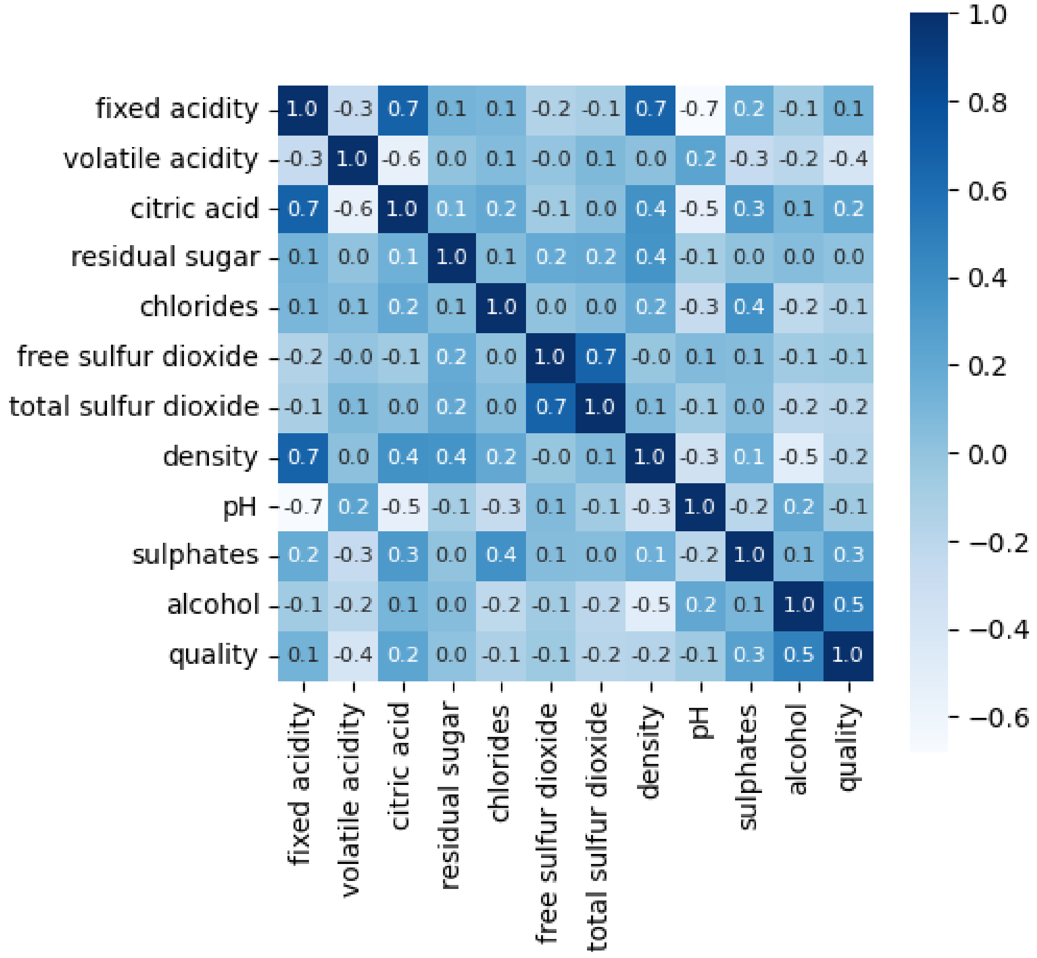
```
sulphates                0.183006         -0.260987     0.312770
alcohol                 -0.061668         -0.202288     0.109903
quality                  0.124052         -0.390558     0.226373

                      residual sugar  chlorides  free sulfur dioxide  \
fixed acidity               0.114777   0.093705            -0.153794
volatile acidity            0.001918   0.061298            -0.010504
citric acid                 0.143577   0.203823            -0.060978
residual sugar              1.000000   0.055610             0.187049
chlorides                   0.055610   1.000000             0.005562
free sulfur dioxide         0.187049   0.005562             1.000000
total sulfur dioxide        0.203028   0.047400             0.667666
density                     0.355283   0.200632            -0.021946
pH                         -0.085652  -0.265026             0.070377
sulphates                   0.005527   0.371260             0.051658
alcohol                     0.042075  -0.221141            -0.069408
quality                     0.013732  -0.128907            -0.050656

                      total sulfur dioxide   density        pH  sulphates  \
fixed acidity                    -0.113181  0.668047 -0.682978   0.183006
volatile acidity                  0.076470  0.022026  0.234937  -0.260987
citric acid                       0.035533  0.364947 -0.541904   0.312770
residual sugar                    0.203028  0.355283 -0.085652   0.005527
chlorides                         0.047400  0.200632 -0.265026   0.371260
free sulfur dioxide               0.667666 -0.021946  0.070377   0.051658
total sulfur dioxide              1.000000  0.071269 -0.066495   0.042947
density                           0.071269  1.000000 -0.341699   0.148506
pH                               -0.066495 -0.341699  1.000000  -0.196648
sulphates                         0.042947  0.148506 -0.196648   1.000000
alcohol                          -0.205654 -0.496180  0.205633   0.093595
quality                          -0.185100 -0.174919 -0.057731   0.251397

                       alcohol   quality
fixed acidity        -0.061668  0.124052
volatile acidity     -0.202288 -0.390558
citric acid           0.109903  0.226373
residual sugar        0.042075  0.013732
chlorides            -0.221141 -0.128907
free sulfur dioxide  -0.069408 -0.050656
total sulfur dioxide -0.205654 -0.185100
density              -0.496180 -0.174919
pH                    0.205633 -0.057731
sulphates             0.093595  0.251397
alcohol               1.000000  0.476166
quality               0.476166  1.000000
```

[21]:
```
# constructing a heatmap to understand the correlation between the columns
plot= plt.figure(figsize=(5,5))
sns.heatmap(correlation, cbar= True, square = True, fmt= '.1f', annot = True,␣
 ↪annot_kws= {'size': 8}, cmap = 'Blues')
```

[21]:
```
<AxesSubplot:>
```

[22]:
```
plot
```

[22]:



	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
fixed acidity	1.0	-0.3	0.7	0.1	0.1	-0.2	-0.1	0.7	-0.7	0.2	-0.1	0.1
volatile acidity	-0.3	1.0	-0.6	0.0	0.1	-0.0	0.1	0.0	0.2	-0.3	-0.2	-0.4
citric acid	0.7	-0.6	1.0	0.1	0.2	-0.1	0.0	0.4	-0.5	0.3	0.1	0.2
residual sugar	0.1	0.0	0.1	1.0	0.1	0.2	0.2	0.4	-0.1	0.0	0.0	0.0
chlorides	0.1	0.1	0.2	0.1	1.0	0.0	0.0	0.2	-0.3	0.4	-0.2	-0.1
free sulfur dioxide	-0.2	-0.0	-0.1	0.2	0.0	1.0	0.7	-0.0	0.1	0.1	-0.1	-0.1
total sulfur dioxide	-0.1	0.1	0.0	0.2	0.0	0.7	1.0	0.1	-0.1	0.0	-0.2	-0.2
density	0.7	0.0	0.4	0.4	0.2	-0.0	0.1	1.0	-0.3	0.1	-0.5	-0.2
pH	-0.7	0.2	-0.5	-0.1	-0.3	0.1	-0.1	-0.3	1.0	-0.2	0.2	-0.1
sulphates	0.2	-0.3	0.3	0.0	0.4	0.1	0.0	0.1	-0.2	1.0	0.1	0.3
alcohol	-0.1	-0.2	0.1	0.0	-0.2	-0.1	-0.2	-0.5	0.2	0.1	1.0	0.5
quality	0.1	-0.4	0.2	0.0	-0.1	-0.1	-0.2	-0.2	-0.1	0.3	0.5	1.0

5 Data Preprocessing

[34]:
```
# seperate the data and label
x = df.drop('quality', axis=1)
```

[35]:
```
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
1                    25.0                  67.0  0.99680  3.20       0.68
2                    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]
```

6 Label Binarization

[36]:
```
y = df['quality'].apply(lambda y_value:1 if y_value >= 7 else 0)
```

[37]:
```
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
```

7 Train and Test split

[39]:
```
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size= 0.2,␣
↪random_state =3)
```

[40]:
```
print(y.shape, y_train.shape,y_test.shape)
```

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

8 Model Training:

Random Forest Classifier Model

[41]:
```
model = RandomForestClassifier()
```

[42]:
```
model.fit(x_train, y_train)
```

[42]:
```
RandomForestClassifier()
```

9 Model Evaluation

Accuracy score

[43]:
```
# accuracy on test data
x_test_prediction = model.predict(x_test)
test_data_accuracy = accuracy_score(x_test_prediction, y_test)
```

[45]:
```
print('Accuracy:',test_data_accuracy)
```

```
Accuracy: 0.928125
```

10 Building a Predictive System

```
[47]: input_data = (7.3,0.65,0,1.2,0.065,15,21,0.9946,3.39,0.47,10)

      #changing the input data into 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:
          print('Bad quality Wine')
```

```
[1]
Good quality Wine
```

```
C:\Users\DELL\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning: X
does not have valid feature names, but RandomForestClassifier was fitted with
feature names
  warnings.warn(
```

```
[ ]:
```

```
[ ]:
```