6_Predicting_Wine_Quality_A_Machine_Learning_Approach_with_Rand

June 30, 2024

0.1 Predicting Wine Quality: A Machine Learning Approach with Random Forest - Vignesh Prabhu

Predicting wine quality is an important task in winemaking that combines scientific analysis and sensory evaluation. Using machine learning techniques provides a systematic approach to comprehending and forecasting the intricate interplay of many wine characteristics. This attempt not only improves quality control, but also gives essential insights for producers who want to continually supply excellent wines to discerning customers.

0.2 Import Dependencies

[ ]:
```
import pandas as pd
import numpy as np
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 And PreProcessing

[ ]:
```
wine_data = pd.read_csv('/content/Wine Data.csv')
```

[ ]:
```
# To show Top 5 datas in dataset
wine_data.head()
```

[ ]:

	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
```

[ ]:
```
#No of Rows and columns
wine_data.shape
```

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

[ ]:
```
#To check null values
wine_data.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
```

[ ]:
```
wine_data.describe()
```

[ ]:
```
       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
```

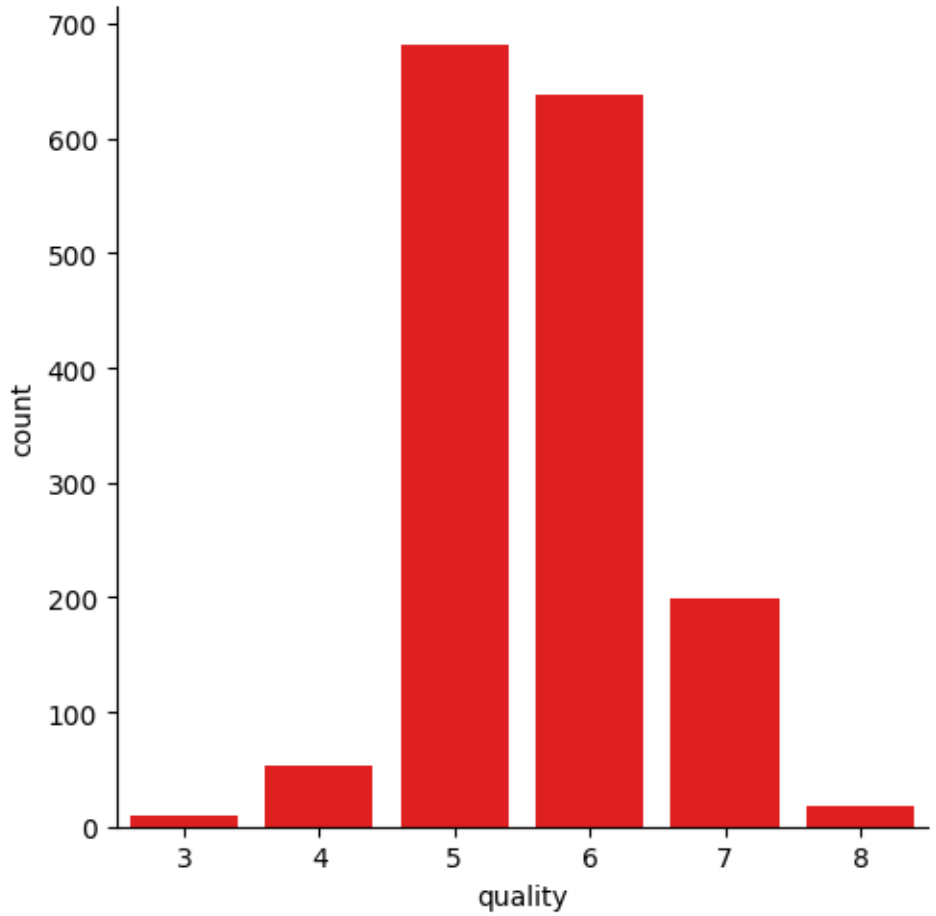
std	0.047065	10.460157	32.895324	0.001887
min	0.012000	1.000000	6.000000	0.990070
25%	0.070000	7.000000	22.000000	0.995600
50%	0.079000	14.000000	38.000000	0.996750
75%	0.090000	21.000000	62.000000	0.997835
max	0.611000	72.000000	289.000000	1.003690

	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

Data Analysis And Data Visualisation

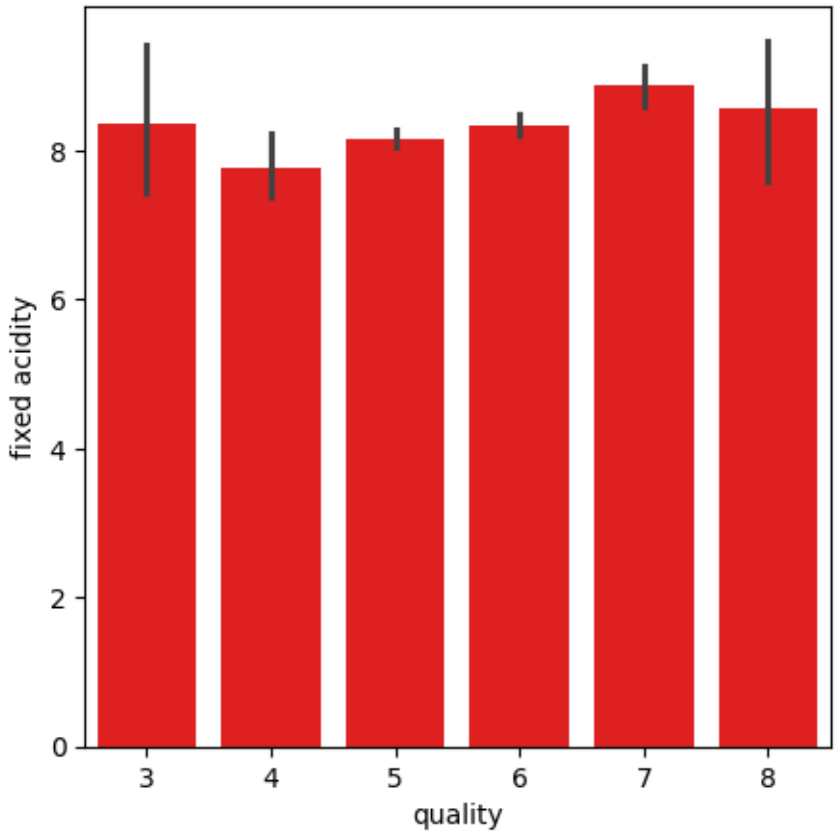
[ ]:
```
#Number of values for each quality
sns.catplot(x='quality', data=wine_data, kind='count', color='red')
```

[ ]:
```
<seaborn.axisgrid.FacetGrid at 0x7ad1d034dea0>
```



[ ]:
```
#Comparing Columns
plot = plt.figure(figsize=(5,5))
sns.barplot(x='quality', y='fixed acidity', data=wine_data,color='red')
```

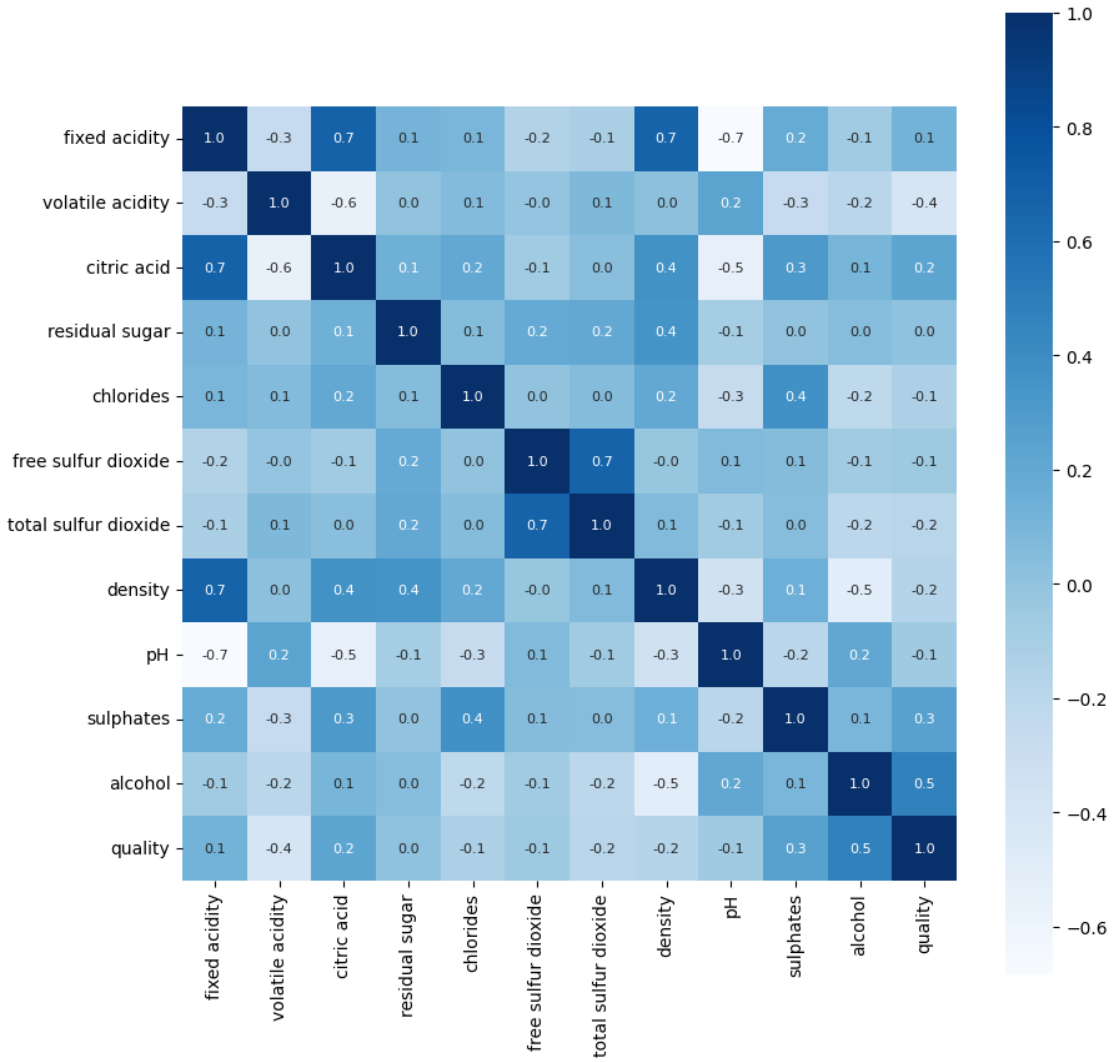
[ ]:
```
<Axes: xlabel='quality', ylabel='fixed acidity'>
```



[ ]:
```
#correlation
correlation = wine_data.corr()
```

[ ]:
```
plt.figure(figsize=(10,10))
sns.heatmap(correlation, cbar=True, square=True, fmt='.1f', annot=True,␣
↪annot_kws={'size':8}, cmap='Blues')
```

[ ]: <Axes: >



Data PreProcessing

[ ]:
```
#separate the Data and Label
X = wine_data.drop('quality',axis=1)
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]
```

Binarization

Binarization is a special form of discretization where data is converted into binary formats – typically '0' and '1'

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

Train and Test Data

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

[ ]:
```
print(X.shape,X_train.shape,X_test.shape)
```

```
(1599, 11) (1279, 11) (320, 11)
```

Model training

[52]:
```
model=  RandomForestClassifier()
```

[66]:
```
model.fit(X_train.values,Y_train)
```

[66]:
```
RandomForestClassifier()
```

Model Evalution

[63]:
```
#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 :  0.94375
```

[64]:
```
#accuracy on train data
X_train_prediction = model.predict(X_train)
train_data_accuracy = accuracy_score(X_train_prediction,Y_train)
print('Accuracy : ', train_data_accuracy)
```

```
Accuracy :  1.0
```

0.3 Model prediction

[67]:
```
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 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')
```

```
[0]
Bad Quality Wine
```

[69]:
```
input_data= (7.3,0.65,0.0,1.2,0.065,15.0,21.0,0.9946,3.39,0.47,10.0)
#changing the input data to 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
```

In conclusion, employing machine learning for wine quality prediction represents a significant advancement in the field of viticulture and oenology. By harnessing data-driven models, winemakers can optimize production processes, refine blending decisions, and ultimately elevate the overall quality of their products. As technology continues to evolve, the integration of predictive analytics promises to revolutionize the industry, offering new avenues for innovation and ensuring continued excellence in winemaking worldwide.

Thank You