Exploratory Data Analysis

Data : Red Wine

MUJAHID RAZA

linkedin.com/in/mujahidraza

github.com/mujahid777

mujahid.raza_07

@mujahidR_07

EDA With Red Wine Data

Information

Additional Information

The two datasets are related to red and white variants of the Portuguese "Vinho Verde" wine. Due to privacy and logistic issues, only physicochemical (inputs) and sensory (the output) variables are available (e.g. there is no data about grape types, wine brand, wine selling price, etc.).

These datasets can be viewed as classification or regression tasks. The classes are ordered and not balanced (e.g. there are many more normal wines than excellent or poor ones). Outlier detection algorithms could be used to detect the few excellent or poor wines. Also, we are not sure if all input variables are relevant. So it could be interesting to test feature selection methods.

Attribute Information

Input variables (based on physicochemical tests):

1 - fixed acidity

2 - volatile acidity

3 - citric acid

4 - residual sugar

5 - chlorides

6 - free sulfur dioxide

7 - total sulfur dioxide

8 - density

9 - pH

10 - sulphates

11 - alcohol

Output variable (based on sensory data):

12 - quality (score between 0 and 10)

In [1]:

```
import pandas as pd
df= pd.read_csv('winequality-red.csv')
```

In [2]:

```
df.head()
```

Out[2]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulpha
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	(
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	(
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	(
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	(
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	(

In [3]:

```
# summery of the dataset

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 [4]:

```
# Descriptive summary of the dataset

df.describe()
```

Out[4]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfu dioxid
count	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.00000
mean	8.319637	0.527821	0.270976	2.538806	0.087467	15.87492
std	1.741096	0.179060	0.194801	1.409928	0.047065	10.46015
min	4.600000	0.120000	0.000000	0.900000	0.012000	1.00000
25%	7.100000	0.390000	0.090000	1.900000	0.070000	7.00000
50%	7.900000	0.520000	0.260000	2.200000	0.079000	14.00000
75%	9.200000	0.640000	0.420000	2.600000	0.090000	21.00000
max	15.900000	1.580000	1.000000	15.500000	0.611000	72.00000

In [5]:

```
# Shape of the dataset

df.shape
```

Out[5]:

```
(1599, 12)
```

In [6]:

```
# List down the columns

df.columns
```

Out[6]:

```
Index(['fixed acidity', 'volatile acidity', 'citric acid', 'residua
l sugar',
       'chlorides', 'free sulfur dioxide', 'total sulfur dioxide',
'density',
       'pH', 'sulphates', 'alcohol', 'quality'],
      dtype='object')
```

In [7]:

```
df['quality'].unique()
```

Out[7]:

```
array([5, 6, 7, 4, 8, 3], dtype=int64)
```

In [8]:

```
# Conclusion-- Imbalanced Dataset

df['quality'].value_counts()
```

Out[8]:

```
5    681
6    638
7    199
4     53
8     18
3     10
Name: quality, dtype: int64
```

In [9]:

```
# Missing Values

df.isnull().sum()
```

Out[9]:

```
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
```

In [10]:

```
# To check duplicate records

df.duplicated()
```

Out[10]:

```
0       False
1       False
2       False
3       False
4        True
        ...  
1594    False
1595    False
1596     True
1597    False
1598    False
Length: 1599, dtype: bool
```

In [11]:

```
df[df.duplicated()]
```

Out[11]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	su
4	7.4	0.700	0.00	1.90	0.076	11.0	34.0	0.99780	3.51	
11	7.5	0.500	0.36	6.10	0.071	17.0	102.0	0.99780	3.35	
27	7.9	0.430	0.21	1.60	0.106	10.0	37.0	0.99660	3.17	
40	7.3	0.450	0.36	5.90	0.074	12.0	87.0	0.99780	3.33	
65	7.2	0.725	0.05	4.65	0.086	4.0	11.0	0.99620	3.41	
...	...	...	...	...	...	...	...	...	...	
1563	7.2	0.695	0.13	2.00	0.076	12.0	20.0	0.99546	3.29	
1564	7.2	0.695	0.13	2.00	0.076	12.0	20.0	0.99546	3.29	
1567	7.2	0.695	0.13	2.00	0.076	12.0	20.0	0.99546	3.29	
1581	6.2	0.560	0.09	1.70	0.053	24.0	32.0	0.99402	3.54	
1596	6.3	0.510	0.13	2.30	0.076	29.0	40.0	0.99574	3.42	

240 rows × 12 columns

In [12]:

```
# Remove the duplicate records

df.drop_duplicates(inplace=True)
```

In [13]:

```
df.shape
```

Out[13]:

```
(1359, 12)
```

In [14]:

```
df.corr()
```

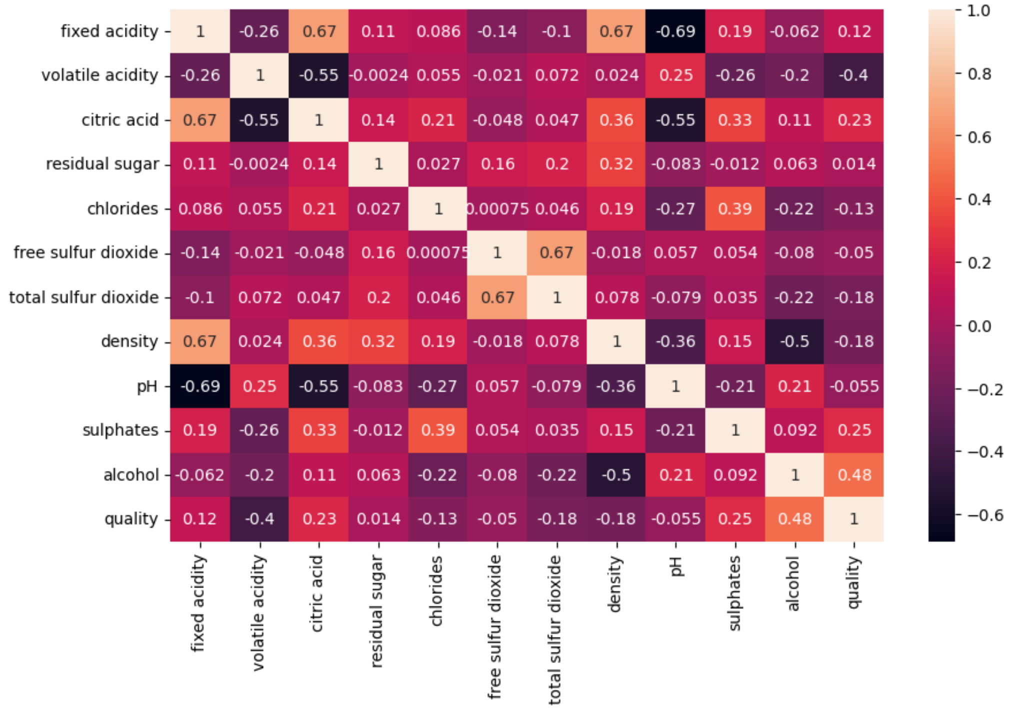
Out[14]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide
fixed acidity	1.000000	-0.255124	0.667437	0.111025	0.085886	-0.140580	-0.103777
volatile acidity	-0.255124	1.000000	-0.551248	-0.002449	0.055154	-0.020945	0.071701
citric acid	0.667437	-0.551248	1.000000	0.143892	0.210195	-0.048004	0.047358
residual sugar	0.111025	-0.002449	0.143892	1.000000	0.026656	0.160527	0.201038
chlorides	0.085886	0.055154	0.210195	0.026656	1.000000	0.000749	0.045773
free sulfur dioxide	-0.140580	-0.020945	-0.048004	0.160527	0.000749	1.000000	0.667246
total sulfur dioxide	-0.103777	0.071701	0.047358	0.201038	0.045773	0.667246	1.000000
density	0.670195	0.023943	0.357962	0.324522	0.193592	-0.018071	0.078141
pH	-0.686685	0.247111	-0.550310	-0.083143	-0.270893	0.056631	-0.079257
sulphates	0.190269	-0.256948	0.326062	-0.011837	0.394557	0.054126	0.035291
alcohol	-0.061596	-0.197812	0.105108	0.063281	-0.223824	-0.080125	-0.217829
quality	0.119024	-0.395214	0.228057	0.013640	-0.130988	-0.050463	-0.177855

```
In [15]:
```

```
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(10,6))
sns.heatmap(df.corr(),annot=True)
```

```
Out[15]:

<Axes: >
```



```
In [16]:
```

```
df.quality.value_counts()
```

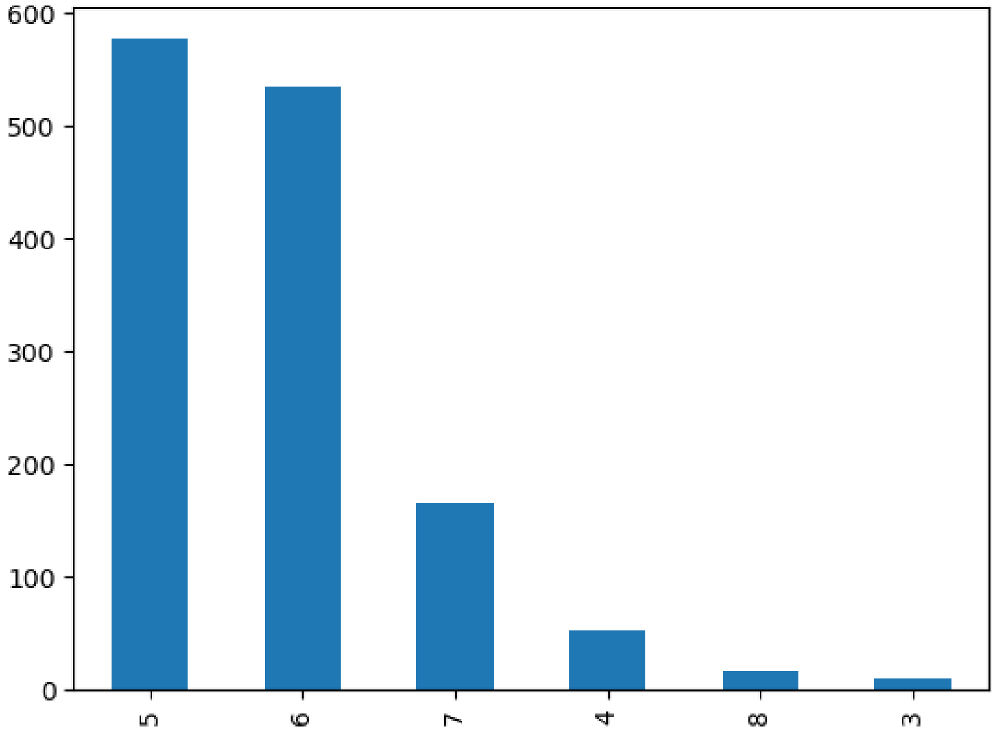
```
Out[16]:

5    577
6    535
7    167
4     53
8     17
3     10
Name: quality, dtype: int64
```

In [17]:

```
df.quality.value_counts().plot(kind='bar')
```

Out[17]:

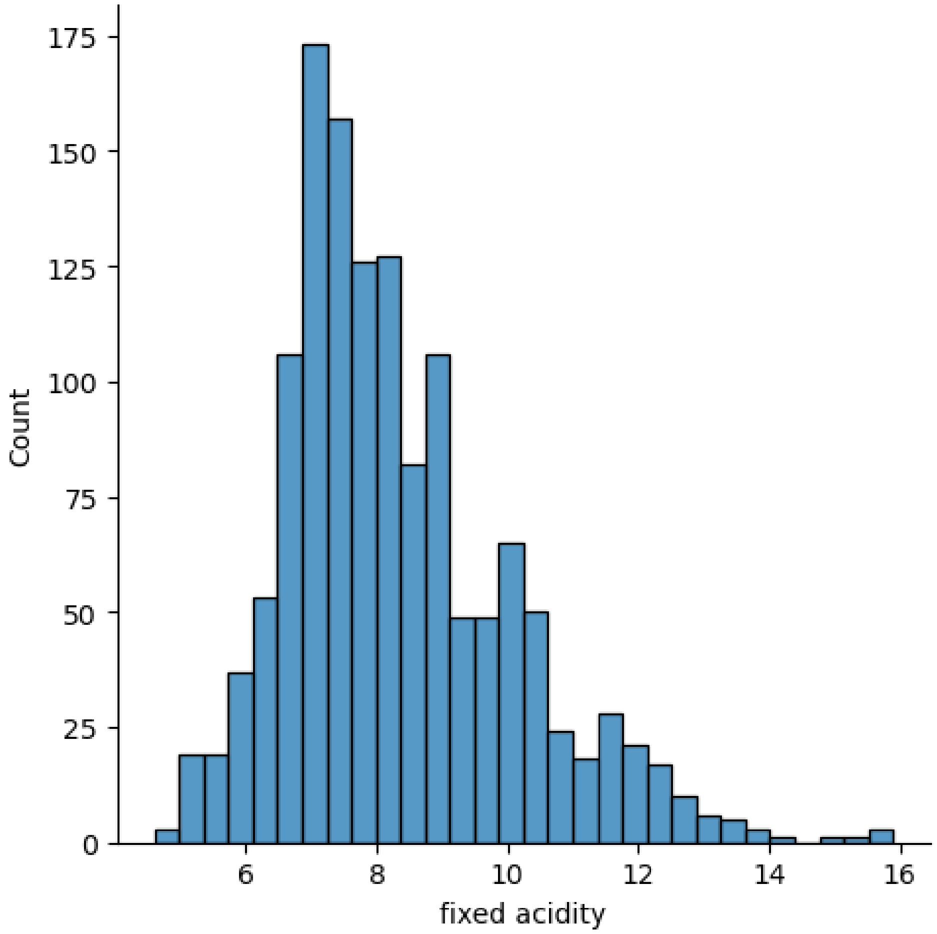
```
<Axes: >
```



```
In [18]:
```

```
sns.displot(df['fixed acidity'])
```

```
Out[18]:

<seaborn.axisgrid.FacetGrid at 0x178a037f970>
```

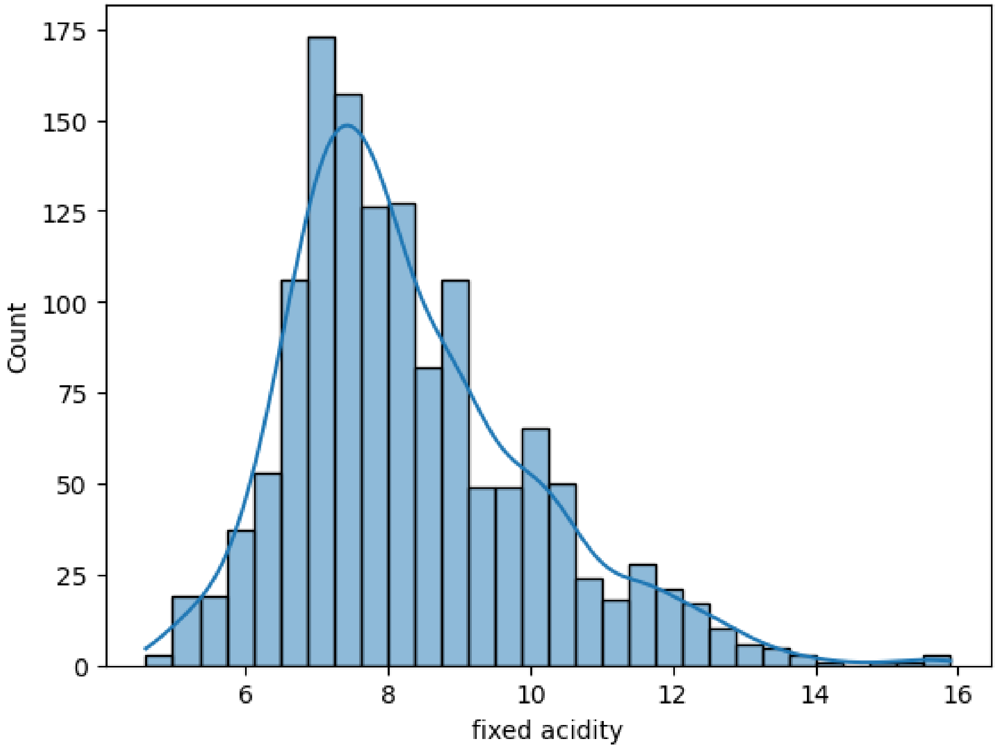


In [19]:

```
sns.histplot(df['fixed acidity'],kde=True)
```

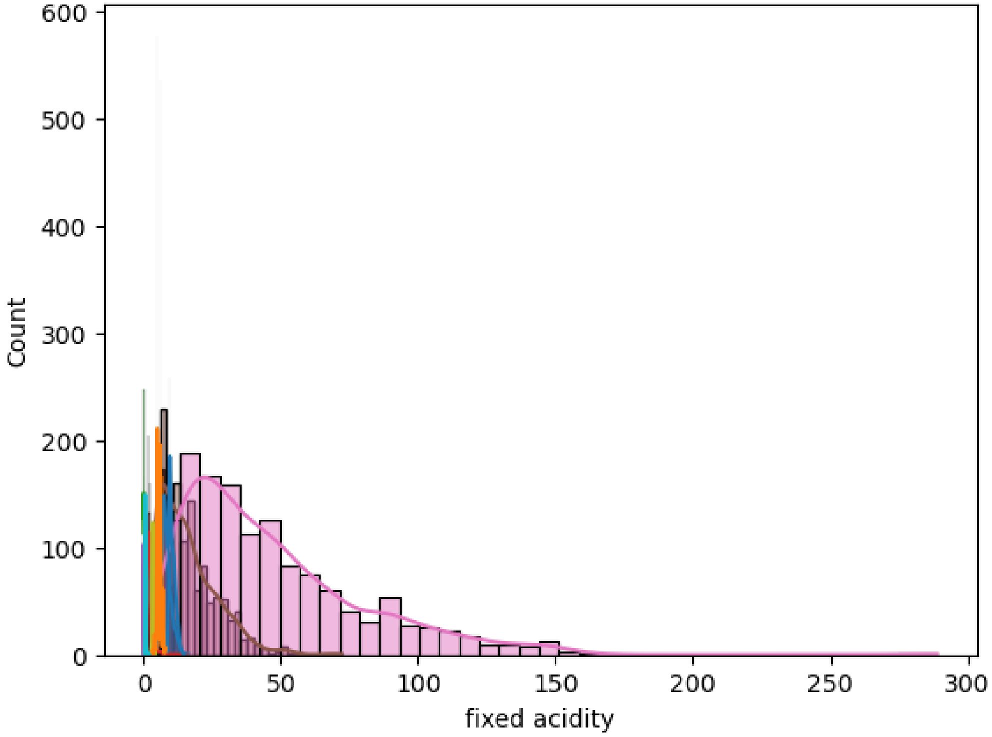
Out[19]:

```
<Axes: xlabel='fixed acidity', ylabel='Count'>
```



In [20]:

```
for i in df.columns:
    sns.histplot(df[i],kde=True)
```

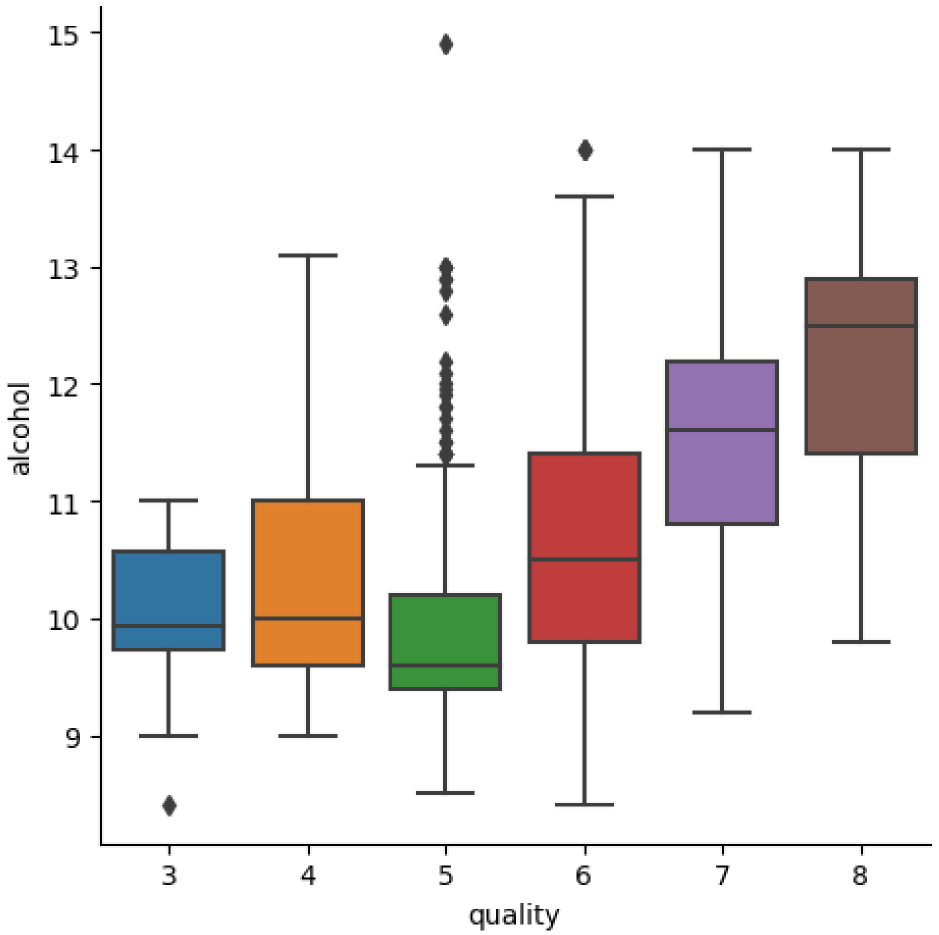


In [21]:

```
# Categorical plot

sns.catplot(x= 'quality',y= 'alcohol',data=df,kind='box')
```

Out[21]:

```
<seaborn.axisgrid.FacetGrid at 0x178a015ee30>
```

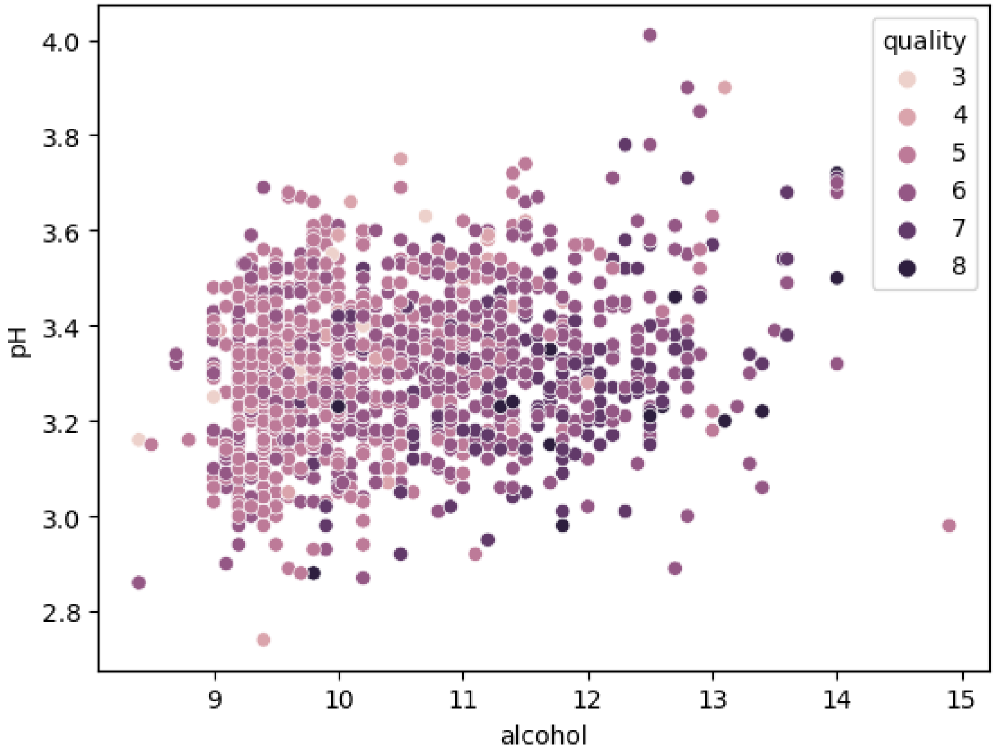


```
In [22]:
```

```
sns.scatterplot(x='alcohol', y= 'pH',hue= 'quality', data=df)
```

```
Out[22]:
```

```
<Axes: xlabel='alcohol', ylabel='pH'>
```

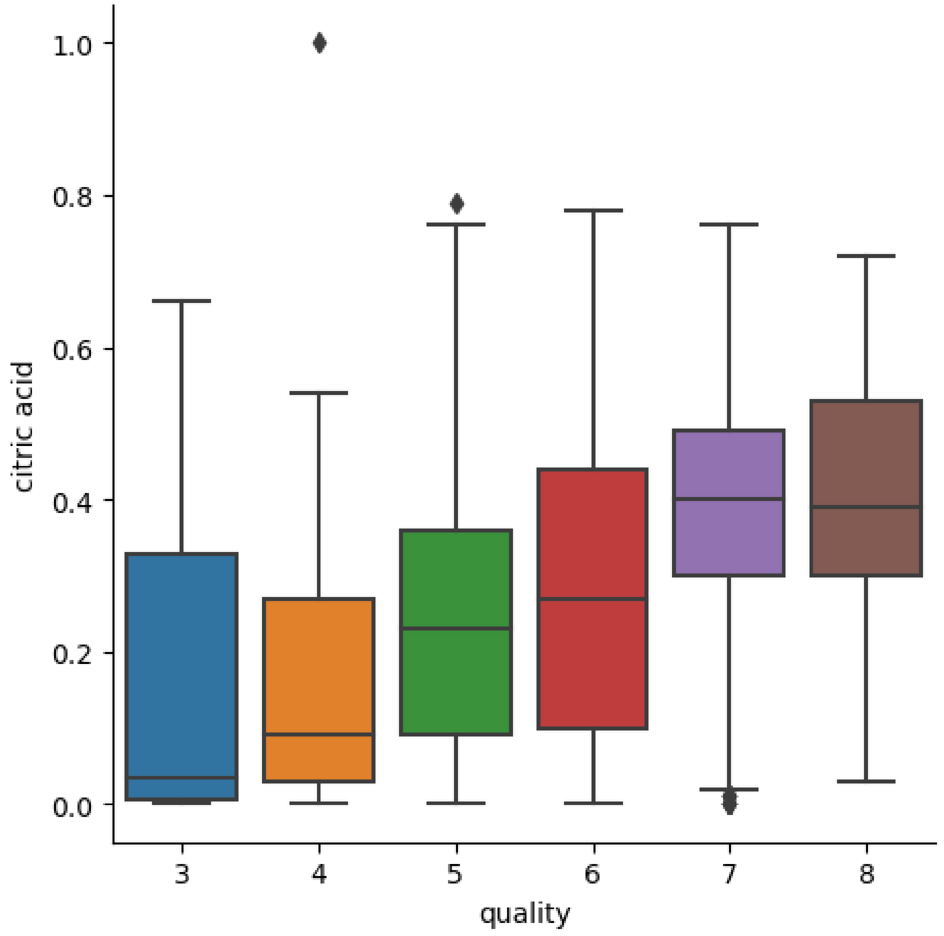


In [23]:

```
sns.catplot(x= 'quality',y= 'citric acid',data=df,kind='box')
```

Out[23]:

```
<seaborn.axisgrid.FacetGrid at 0x1789cdf3550>
```

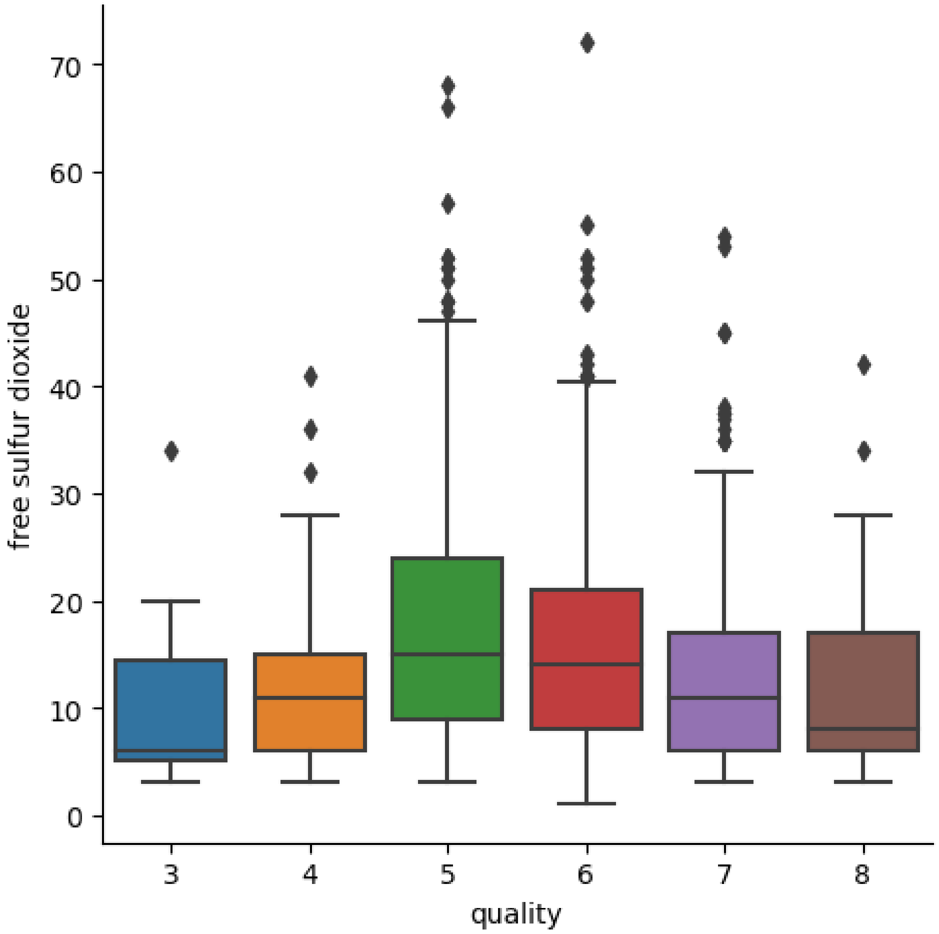


In [24]:

```
sns.catplot(x= 'quality',y= 'free sulfur dioxide',data=df,kind='box')
```

Out[24]:

```
<seaborn.axisgrid.FacetGrid at 0x178a06b10c0>
```

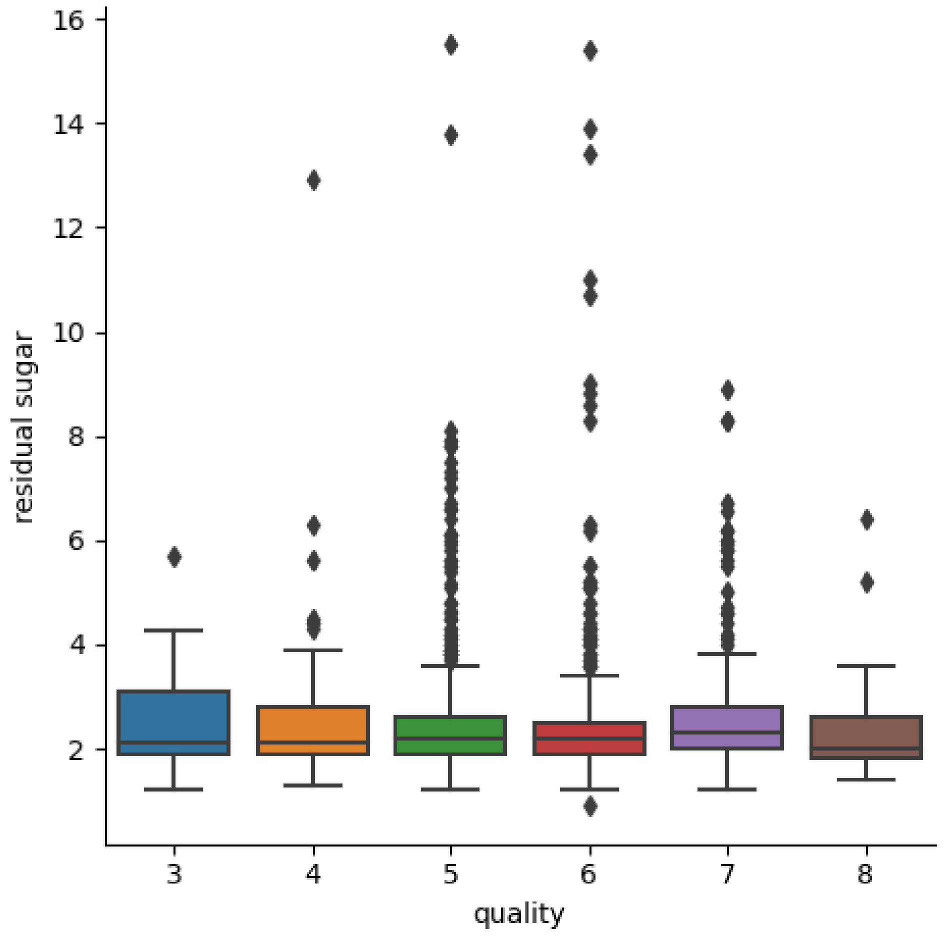


In [26]:

```
sns.catplot(x= 'quality',y= 'residual sugar',data=df,kind='box')
```

Out[26]:

```
<seaborn.axisgrid.FacetGrid at 0x178a26d52d0>
```



In [ ]:

```
```