EDA

Exploratory Data Analysis (EDA) is an approach to analyze the data using visual techniques. It is used to discover trends, patterns, or to check assumptions with the help of statistical summary and graphical representations.

Data Set Information:

The two datasets are related to red and white variants of the Portuguese "Vinho Verde" wine. For more details, consult: [Web Link] or the reference [Cortez et al., 2009]. 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)

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

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	f
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	

```
##summary of 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
```

```
##discriptive summary of dataset
df.describe()  ##it gives count mean median std min-max etc
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfu dioxid
count	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.0000
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	1.0000
25%	7.100000	0.390000	0.090000	1.900000	0.070000	7.0000
50%	7.900000	0.520000	0.260000	2.200000	0.079000	14.0000
75%	9.200000	0.640000	0.420000	2.600000	0.090000	21.0000
max	15.900000	1.580000	1.000000	15.500000	0.611000	72.0000

```
##shape of dataset (size like count of row and column)
df.shape
```

```
(1599, 12)
```

```
#list down all column
df.columns
```

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

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

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

```
##conclusion :Imbalanced dataset
df['quality'].value_counts()
```

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

```
##Missing values
df.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
```

```
##to check duplicate record
df[df.duplicated()]
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides
4	7.4	0.700	0.00	1.90	0.076
11	7.5	0.500	0.36	6.10	0.071
27	7.9	0.430	0.21	1.60	0.106
40	7.3	0.450	0.36	5.90	0.074
65	7.2	0.725	0.05	4.65	0.086
...	...	...	...	...	...
1563	7.2	0.695	0.13	2.00	0.076
1564	7.2	0.695	0.13	2.00	0.076
1567	7.2	0.695	0.13	2.00	0.076
1581	6.2	0.560	0.09	1.70	0.053
1596	6.3	0.510	0.13	2.30	0.076

```
##Removing the duplicated record
df.drop_duplicates(inplace=True)
df.shape ## now check by shape all duplicates is removed
```

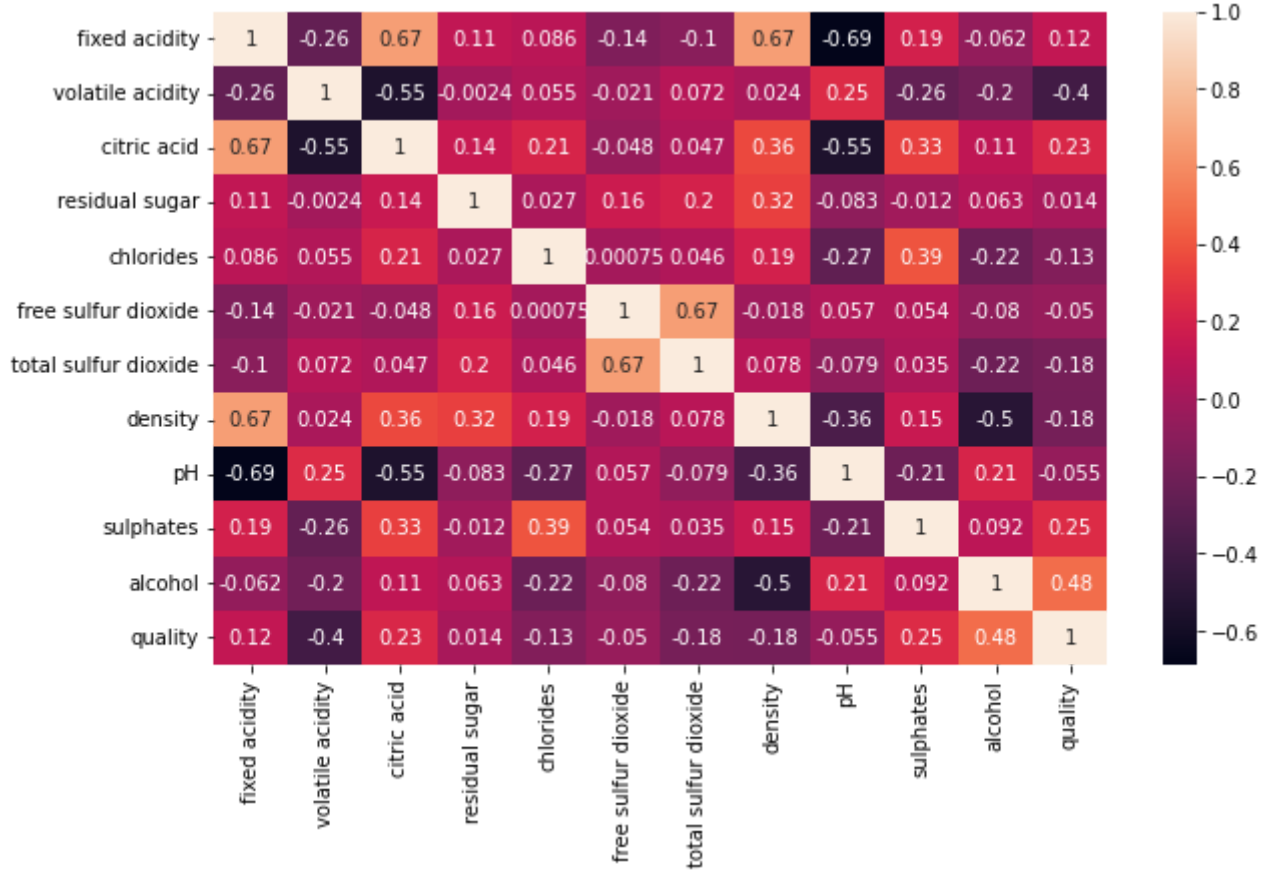
```
(1359, 12)
```

```
##Analyzing correlation
df.corr()
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	fr

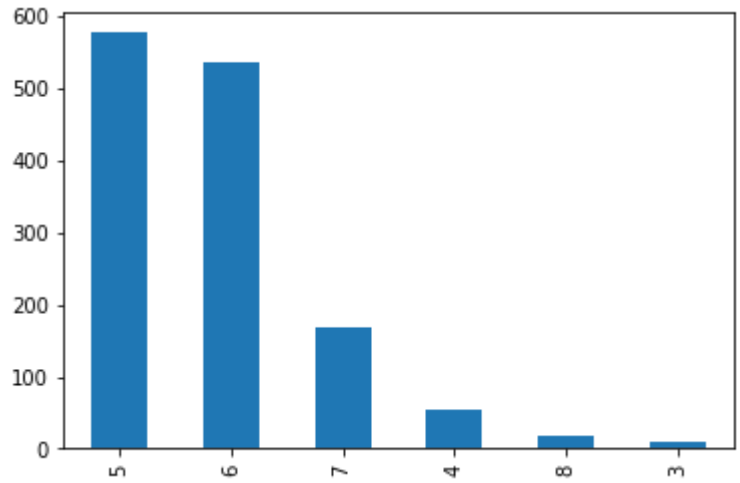
```
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(10,6))
sns.heatmap(df.corr(),annot=True)
```

```
<Axes: >
```



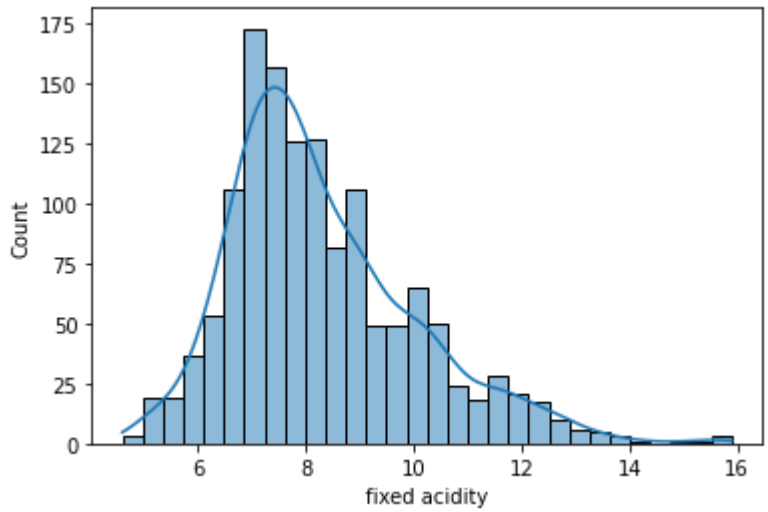

```
df.quality.value_counts().plot(kind='bar')
```

```
<Axes: >
```

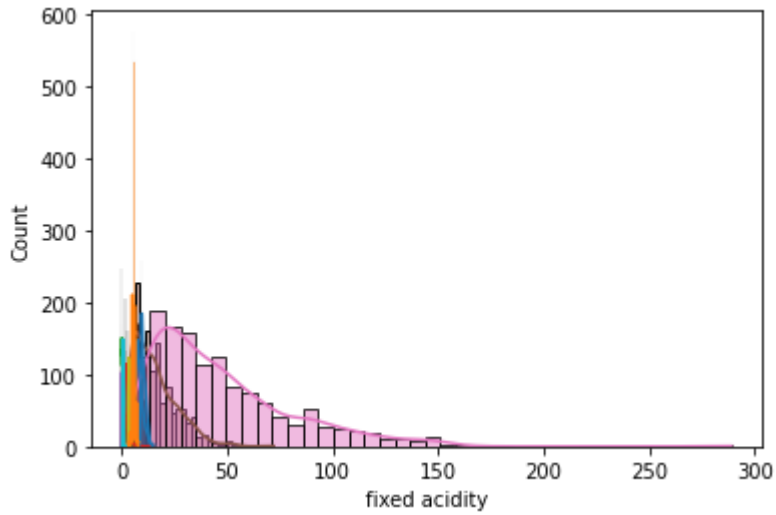



```
sns.histplot(df['fixed acidity'],kde=True)
```

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




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




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

```
<seaborn.axisgrid.FacetGrid at 0x7f77c93820a0>
```


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

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