```
#Adithya Vardhan
#21BAI1535
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

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

```
df.head()
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5

```
df.sample(5)
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
1596	6.3	0.51	0.13	2.30	0.076	29.0	40.0	0.99574	3.42	0.75	11.0	6
1455	6.5	0.90	0.00	1.60	0.052	9.0	17.0	0.99467	3.50	0.63	10.9	6
1111	5.4	0.42	0.27	2.00	0.092	23.0	55.0	0.99471	3.78	0.64	12.3	7
840	11.1	0.42	0.47	2.65	0.085	9.0	34.0	0.99736	3.24	0.77	12.1	7
431	7.8	0.55	0.35	2.20	0.074	21.0	66.0	0.99740	3.25	0.56	9.2	5

```
df.shape
```

```
(1599, 12)
```

```
df.describe()
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	su
count	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	159
mean	8.319637	0.527821	0.270976	2.538806	0.087467	15.874922	46.467792	0.996747	3.311113	
std	1.741096	0.179060	0.194801	1.409928	0.047065	10.460157	32.895324	0.001887	0.154386	
min	4.600000	0.120000	0.000000	0.900000	0.012000	1.000000	6.000000	0.990070	2.740000	
25%	7.100000	0.390000	0.090000	1.900000	0.070000	7.000000	22.000000	0.995600	3.210000	
50%	7.900000	0.520000	0.260000	2.200000	0.079000	14.000000	38.000000	0.996750	3.310000	
75%	9.200000	0.640000	0.420000	2.600000	0.090000	21.000000	62.000000	0.997835	3.400000	
max	15.900000	1.580000	1.000000	15.500000	0.611000	72.000000	289.000000	1.003690	4.010000	

```
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
```

CHECKING NULL VALUES

```
df.isnull().sum()
#THERE IS NO NULL VALUES
```

```
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
```

DULICATE VALUE CHECKING

```
df.duplicated().sum()
```

```
240
```

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

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
4	7.4	0.700	0.00	1.90	0.076	11.0	34.0	0.99780	3.51	0.56	9.4	5
11	7.5	0.500	0.36	6.10	0.071	17.0	102.0	0.99780	3.35	0.80	10.5	5
27	7.9	0.430	0.21	1.60	0.106	10.0	37.0	0.99660	3.17	0.91	9.5	5
40	7.3	0.450	0.36	5.90	0.074	12.0	87.0	0.99780	3.33	0.83	10.5	5
65	7.2	0.725	0.05	4.65	0.086	4.0	11.0	0.99620	3.41	0.39	10.9	5
...	...	...	...	...	...	...	...	...	...	...	...	...
1563	7.2	0.695	0.13	2.00	0.076	12.0	20.0	0.99546	3.29	0.54	10.1	5
1564	7.2	0.695	0.13	2.00	0.076	12.0	20.0	0.99546	3.29	0.54	10.1	5
1567	7.2	0.695	0.13	2.00	0.076	12.0	20.0	0.99546	3.29	0.54	10.1	5
1581	6.2	0.560	0.09	1.70	0.053	24.0	32.0	0.99402	3.54	0.60	11.3	5
1596	6.3	0.510	0.13	2.30	0.076	29.0	40.0	0.99574	3.42	0.75	11.0	6

240 rows × 12 columns

```
df.drop_duplicates(keep='first')
df.shape

     (1599, 12)
```

LABEL ENCODING FOR QUALITY COLUMN TO MAKE IT OUT OF 5

2 7.8 0.760 0.04 2.3 0.092 15.0 54.0 0.99700 3.26 0.65 9.8 5

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

     array([2, 3, 4, 1, 5, 0])
```

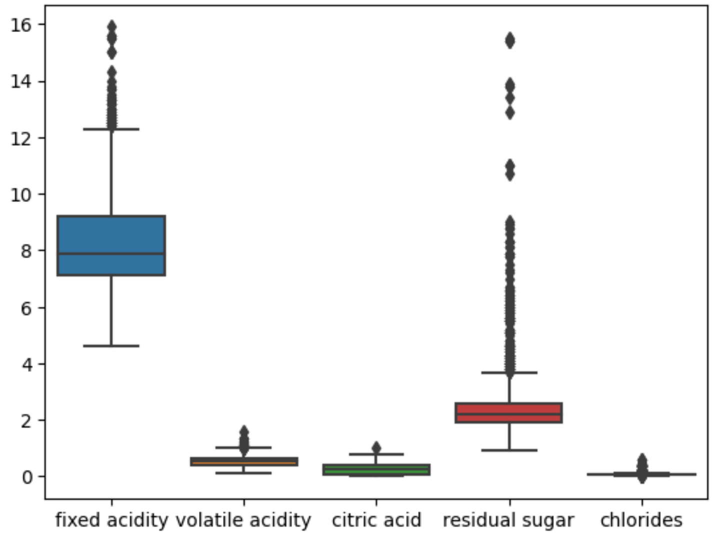
```
from sklearn import preprocessing
label_encoder = preprocessing.LabelEncoder()
df['quality']= label_encoder.fit_transform(df['quality'])

df['quality'].unique()

     array([2, 3, 4, 1, 5, 0])
```

1598 6.0 0.310 0.47 3.6 0.067 18.0 42.0 0.99549 3.39 0.66 11.0 6

BOXPLOT FOR OUTLIERS

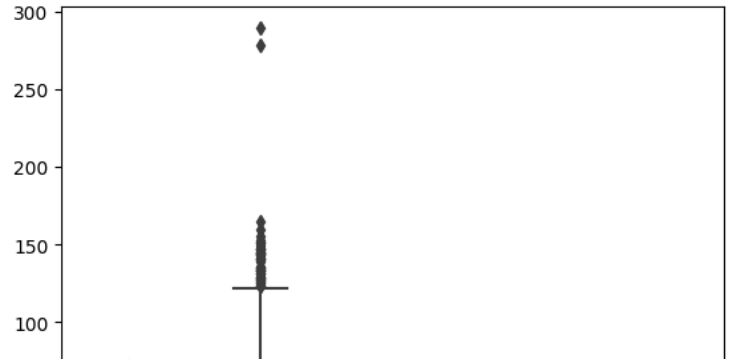
```
df.info()

     <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
```

```
sns.boxplot(df.iloc[:,0:5])
plt.show()
```



```
sns.boxplot(df.iloc[:,5:10])
```

```
<Axes: >
```



data visualisation

```
df.info()
```

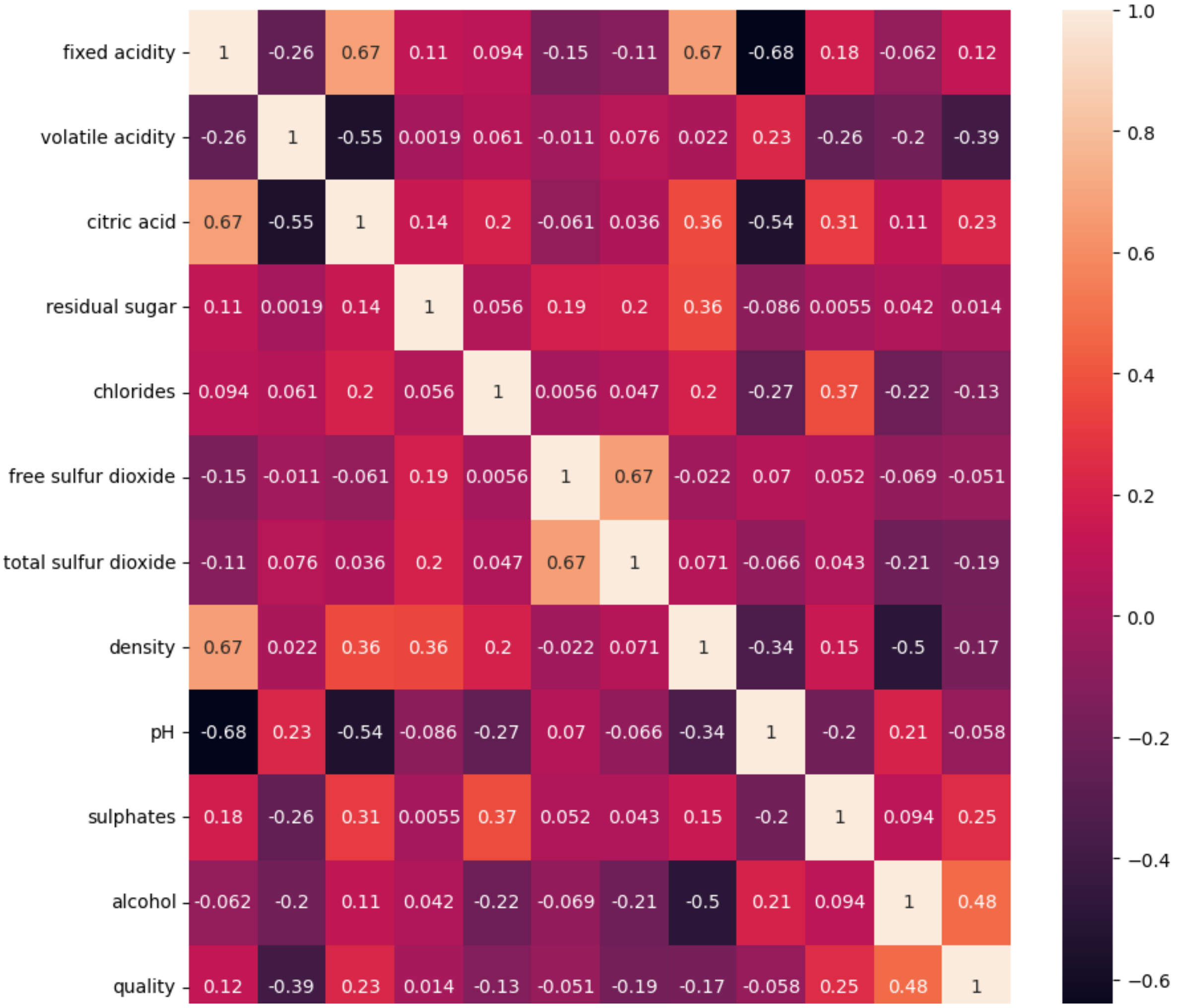
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1599 entries, 0 to 1598
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   fixed acidity         1599 non-null   float64
 1   volatile acidity      1599 non-null   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
```

```
df.corr()
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol
fixed acidity	1.000000	-0.256131	0.671703	0.114777	0.093705	-0.153794	-0.113181	0.668047	-0.682978	0.183006	-0.061668
volatile acidity	-0.256131	1.000000	-0.552496	0.001918	0.061298	-0.010504	0.076470	0.022026	0.234937	-0.260987	-0.202288
citric acid	0.671703	-0.552496	1.000000	0.143577	0.203823	-0.060978	0.035533	0.364947	-0.541904	0.312770	0.109903
residual sugar	0.114777	0.001918	0.143577	1.000000	0.055610	0.187049	0.203028	0.355283	-0.085652	0.005527	0.042075
chlorides	0.093705	0.061298	0.203823	0.055610	1.000000	0.005562	0.047400	0.200632	-0.265026	0.371260	-0.221141
free sulfur dioxide	-0.153794	-0.010504	-0.060978	0.187049	0.005562	1.000000	0.667666	-0.021946	0.070377	0.051658	-0.069408
total sulfur dioxide	-0.113181	0.076470	0.035533	0.203028	0.047400	0.667666	1.000000	0.071269	-0.066495	0.042947	-0.205654
density	0.668047	0.022026	0.364947	0.355283	0.200632	-0.021946	0.071269	1.000000	-0.341699	0.148506	-0.496180
pH	-0.682978	0.234937	-0.541904	-0.085652	-0.265026	0.070377	-0.066495	-0.341699	1.000000	-0.196648	0.205633
sulphates	0.183006	-0.260987	0.312770	0.005527	0.371260	0.051658	0.042947	0.148506	-0.196648	1.000000	0.093595

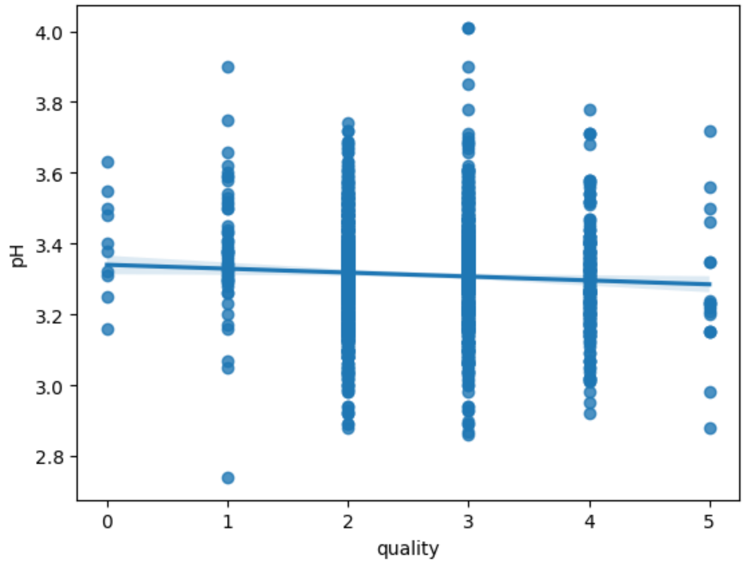
```
plt.figure(figsize=(10,10))
sns.heatmap(df.corr(),annot=True)
```

```
<Axes: >
```



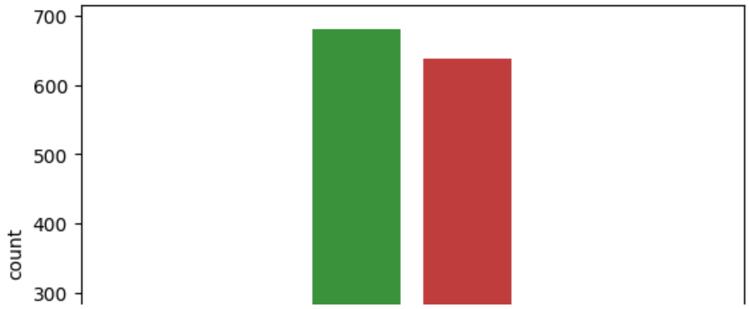
```
sns.regplot(x='quality',y='pH',data=df)
```

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



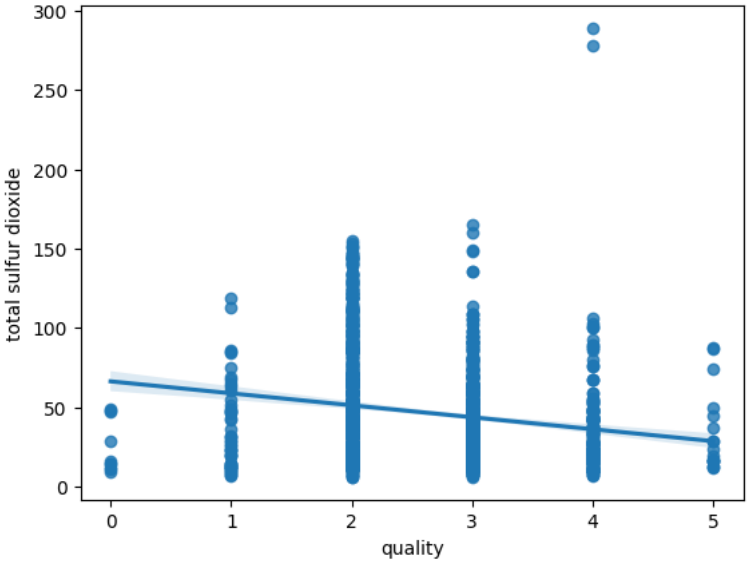
```
sns.countplot(x='quality',data=df)
```

```
<Axes: xlabel='quality', ylabel='count'>
```



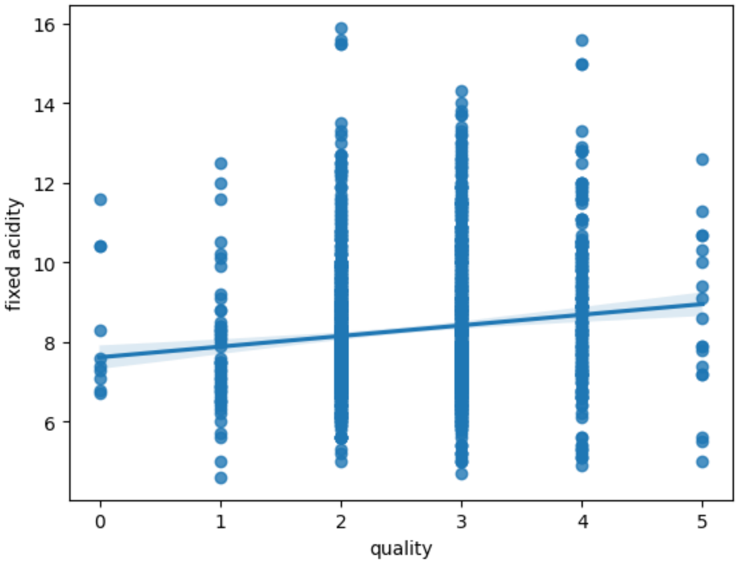
```
sns.regplot(x='quality',y='total sulfur dioxide',data=df)
```

```
<Axes: xlabel='quality', ylabel='total sulfur dioxide'>
```



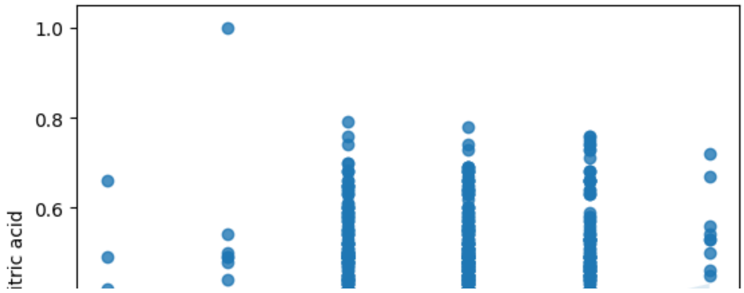
```
sns.regplot(x='quality',y='fixed acidity',data=df)
```

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



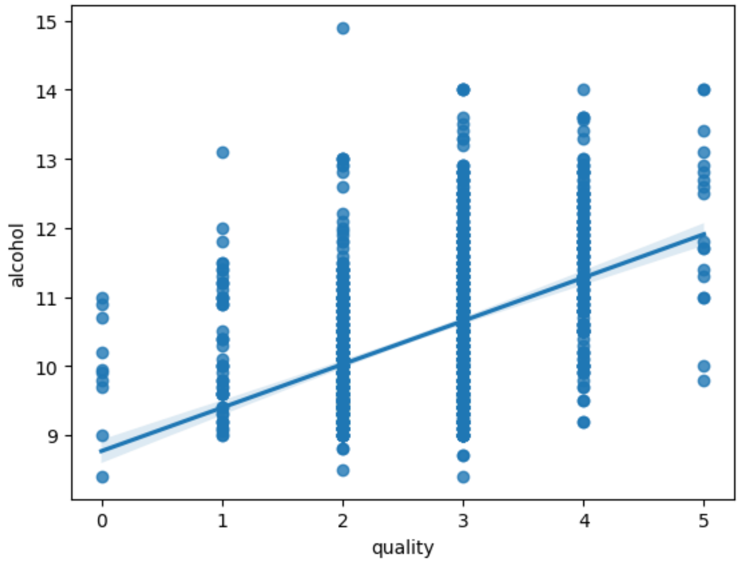
```
sns.regplot(x='quality',y='citric acid',data=df)
```

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



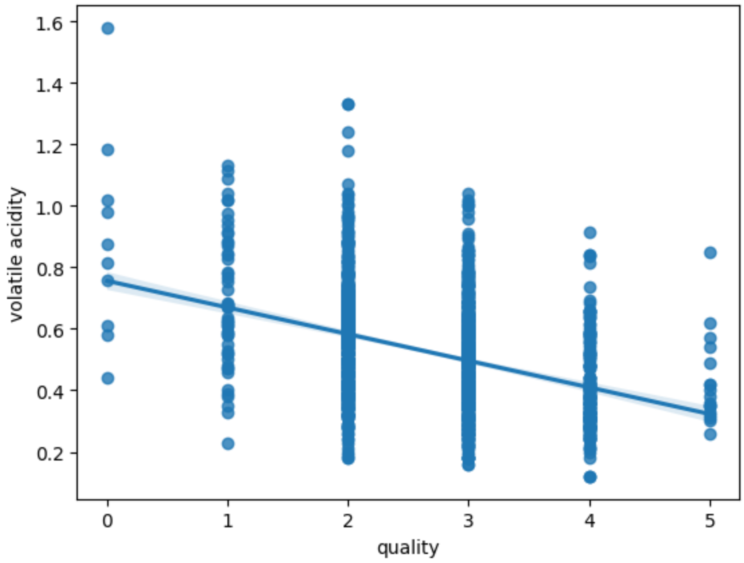
```
sns.regplot(x='quality',y='alcohol',data=df)
```

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



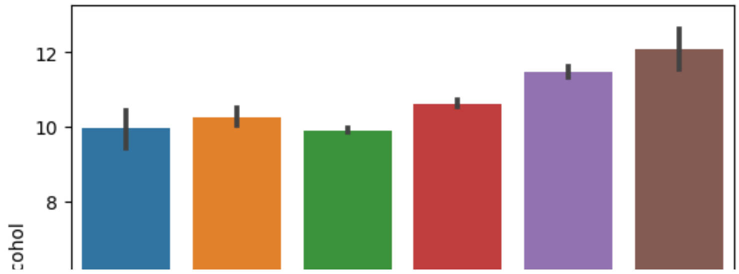
```
sns.regplot(x='quality',y='volatile acidity',data=df)
```

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



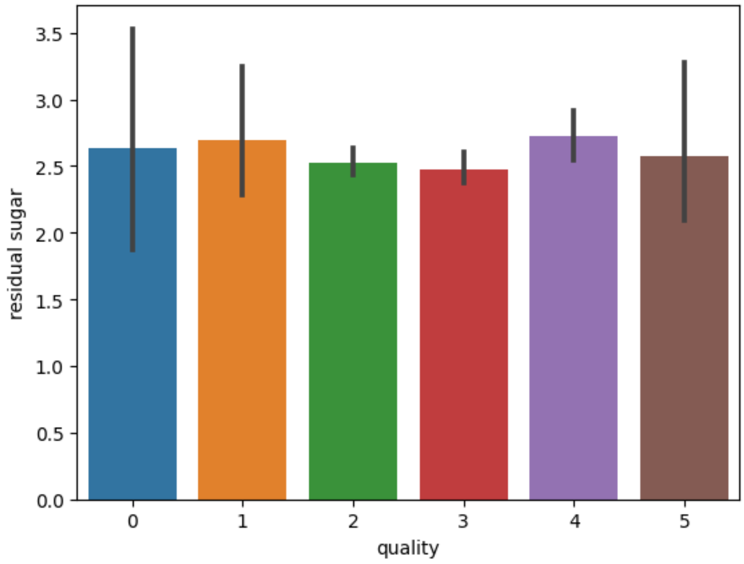
```
sns.barplot(x='quality',y='alcohol',data=df)
```

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



```
sns.barplot(x='quality',y='residual sugar',data=df)
```

```
<Axes: xlabel='quality', ylabel='residual sugar'>
```



TRAIN TEST SPLIT

```
X=df.drop(['quality'],axis=1)
```

```
y=df['quality']
```

```
y=df['quality'].apply(lambda y_value: 1 if (y_value>4 ) else 0)
```

```
print(y.tail(4))
```

```
1595    0
1596    0
1597    0
1598    0
Name: quality, dtype: int64
```

```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
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,)
```

```
print(X_train.shape,X_test.shape)
```

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

Model building

Random Forest

```
model = RandomForestClassifier()
```

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

```
▾ RandomForestClassifier
RandomForestClassifier()
```

```
X_test_prediction = model.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, y_test)
```

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

```
Accuracy :  0.996875
```

```
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:
  print('Bad Quality Wine')
```

```
[0]
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(
```