DATA COLLECTION

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:

```
a=pd.read_csv(r"C:\Users\user\Downloads\11_winequality-red.csv")
a
```

Out[2]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	a
0	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	0.56	
1	7.8	0.880	0.00	2.6	0.098	25.0	67.0	0.99680	3.20	0.68	
2	7.8	0.760	0.04	2.3	0.092	15.0	54.0	0.99700	3.26	0.65	
3	11.2	0.280	0.56	1.9	0.075	17.0	60.0	0.99800	3.16	0.58	
4	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	0.56	
...	...	...	...	...	...	...	...	...	...	...	
1594	6.2	0.600	0.08	2.0	0.090	32.0	44.0	0.99490	3.45	0.58	
1595	5.9	0.550	0.10	2.2	0.062	39.0	51.0	0.99512	3.52	0.76	
1596	6.3	0.510	0.13	2.3	0.076	29.0	40.0	0.99574	3.42	0.75	
1597	5.9	0.645	0.12	2.0	0.075	32.0	44.0	0.99547	3.57	0.71	
1598	6.0	0.310	0.47	3.6	0.067	18.0	42.0	0.99549	3.39	0.66	

1599 rows × 12 columns

In [3]:

```
a.head(5)
```

Out[3]:

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

DATA CLEANING AND PRE-PROCESSING

In [4]:

```
a.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 [5]:

```
a.describe()
```

Out[5]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total di
count	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.0
mean	8.319637	0.527821	0.270976	2.538806	0.087467	15.874922	46.4
std	1.741096	0.179060	0.194801	1.409928	0.047065	10.460157	32.8
min	4.600000	0.120000	0.000000	0.900000	0.012000	1.000000	6.0
25%	7.100000	0.390000	0.090000	1.900000	0.070000	7.000000	22.0
50%	7.900000	0.520000	0.260000	2.200000	0.079000	14.000000	38.0
75%	9.200000	0.640000	0.420000	2.600000	0.090000	21.000000	62.0
max	15.900000	1.580000	1.000000	15.500000	0.611000	72.000000	289.0

In [6]:

```
a.columns
```

Out[6]:

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

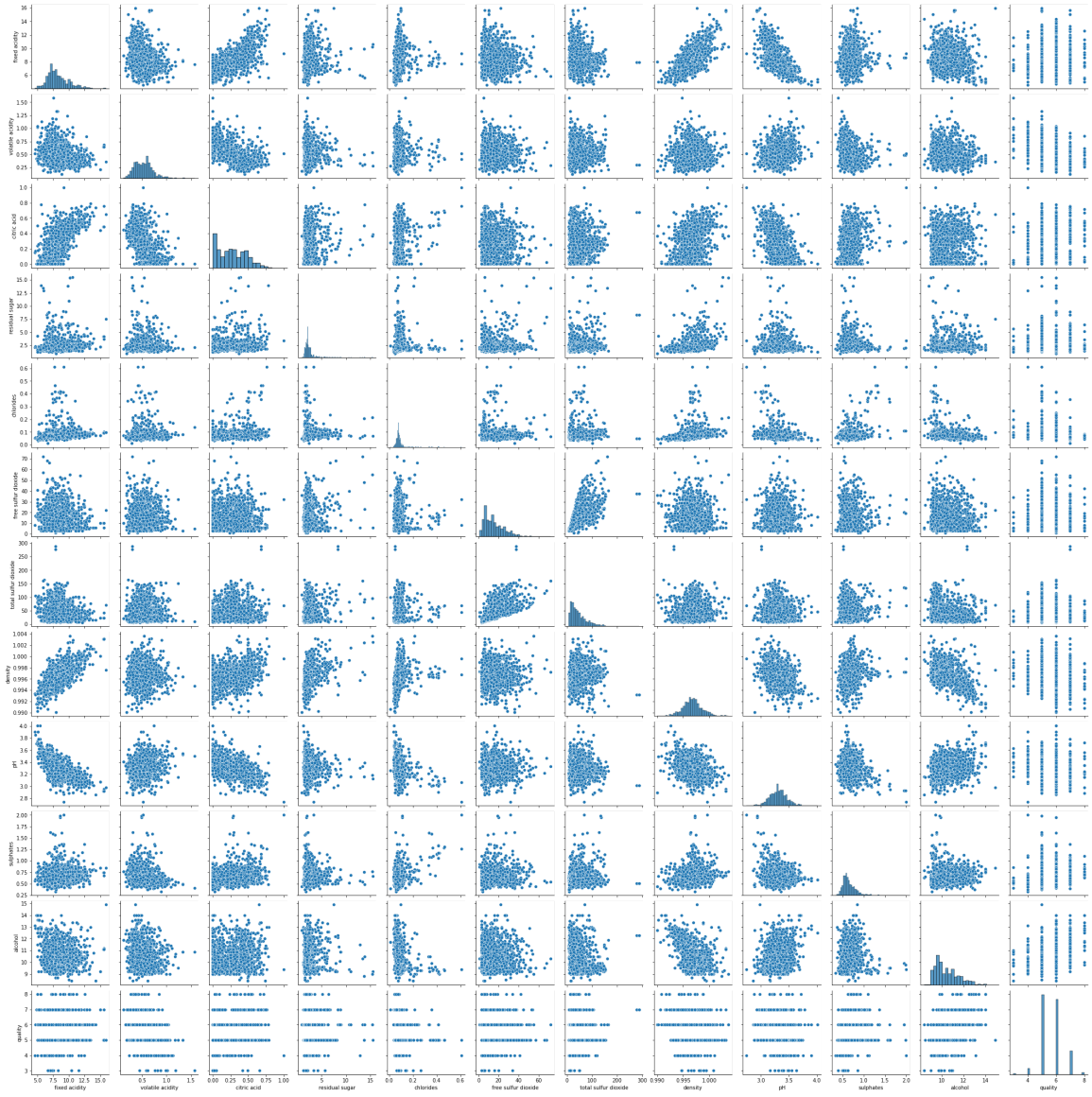
EDA and VISUALIZATION

In [7]:

```
sns.pairplot(a)
```

Out[7]:

```
<seaborn.axisgrid.PairGrid at 0x18d4f636d90>
```



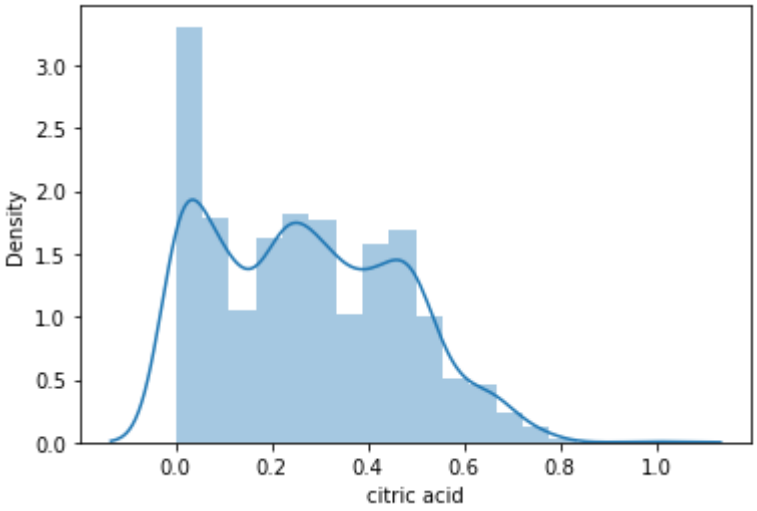
In [8]:

```
sns.distplot(a['citric acid'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557:
FutureWarning: `distplot` is a deprecated function and will be removed in
a future version. Please adapt your code to use either `displot` (a figure
-level function with similar flexibility) or `histplot` (an axes-level fun
ction for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[8]:

```
<AxesSubplot:xlabel='citric acid', ylabel='Density'>
```



In [9]:

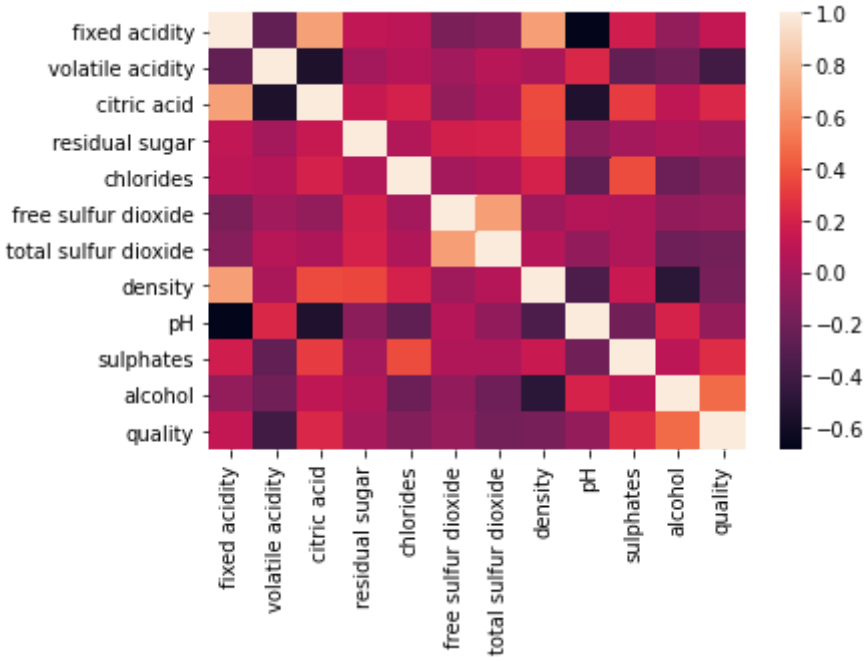
```
f=a[['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
       'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
       'pH', 'sulphates', 'alcohol', 'quality']]
```

In [10]:

```
sns.heatmap(f.corr())
```

Out[10]:

```
<AxesSubplot:>
```



To train the model-model building

In [11]:

```
x=f[['fixed acidity', 'volatile acidity','residual sugar',
       'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
       'pH', 'sulphates', 'alcohol', 'quality']]
y=f['citric acid']
```

In [12]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

In [13]:

```
from sklearn.linear_model import LinearRegression

lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[13]:

```
LinearRegression()
```

In [14]:

```
print(lr.intercept_)
```

```
2.5843290594097157
```

In [15]:

```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[15]:

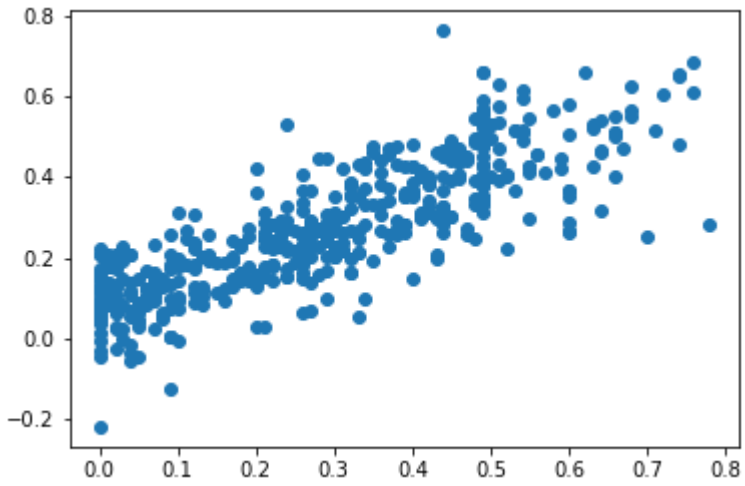
	Co-efficient
fixed acidity	0.061519
volatile acidity	-0.446338
residual sugar	0.005519
chlorides	0.825668
free sulfur dioxide	-0.002472
total sulfur dioxide	0.001272
density	-2.911242
pH	-0.013626
sulphates	0.027735
alcohol	0.022281
quality	-0.000503

In [16]:

```
prediction=lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[16]:

```
<matplotlib.collections.PathCollection at 0x18d58ebb160>
```



In [17]:

```
print(lr.score(x_test,y_test))
```

```
0.6876992302836324
```

In [18]:

```
lr.score(x_train,y_train)
```

Out[18]:

```
0.6743276120005179
```

RIDGE REGRESSION

In [19]:

```
from sklearn.linear_model import Ridge,Lasso
```

In [20]:

```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[20]:

```
Ridge(alpha=10)
```

In [21]:

```
rr.score(x_test,y_test)
```

Out[21]:

```
0.6775140538038511
```

LASSO REGRESSION

In [22]:

```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[22]:

```
Lasso(alpha=10)
```

In [23]:

```
la.score(x_test,y_test)
```

Out[23]:

```
-0.007117544613139071
```

In [24]:

```
from sklearn.linear_model import ElasticNet
p=ElasticNet()
p.fit(x_train,y_train)
```

Out[24]:

```
ElasticNet()
```

In [25]:

```
print(p.coef_)
```

```
[ 0. -0.  0.  0. -0.  0.  0. -0.  0.  0.  0.]
```

In [26]:

```
print(p.intercept_)
```

```
0.2659338695263629
```

In [27]:

```
print(p.predict(x_test))
```

```
[0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
```

```
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387
 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387 0.26593387]
```

In [28]:

```
prediction=p.predict(x_test)
print(p.score(x_test,y_test))
```

```
-0.007117544613139071
```

In [29]:

```
from sklearn import metrics
```

In [30]:

```
print("Mean Absolytre Error:",metrics.mean_absolute_error(y_test,prediction))
```

```
Mean Absolytre Error: 0.1686069221030682
```

In [31]:

```
print("Mean Squared Error:",metrics.mean_squared_error(y_test,prediction))
```

```
Mean Squared Error: 0.03991400865533665
```

In [32]:

```
print("Root Mean Squared Error:",np.sqrt(metrics.mean_squared_error(y_test,prediction)))
```

```
Root Mean Squared Error: 0.19978490597474236
```

In [ ]: