In [1]:

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

In [2]:

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

Out[2]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates
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

In [3]:
```
df=data.head(100)
df
```

Out[3]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcoho
0	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.
1	7.8	0.880	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.
2	7.8	0.760	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.
3	11.2	0.280	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.
4	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.
...	...	...	...	...	...	...	...	...	...	...	.
95	4.7	0.600	0.17	2.3	0.058	17.0	106.0	0.9932	3.85	0.60	12.
96	6.8	0.775	0.00	3.0	0.102	8.0	23.0	0.9965	3.45	0.56	10.
97	7.0	0.500	0.25	2.0	0.070	3.0	22.0	0.9963	3.25	0.63	9.
98	7.6	0.900	0.06	2.5	0.079	5.0	10.0	0.9967	3.39	0.56	9.
99	8.1	0.545	0.18	1.9	0.080	13.0	35.0	0.9972	3.30	0.59	9.

100 rows × 12 columns

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   fixed acidity         100 non-null    float64
 1   volatile acidity      100 non-null    float64
 2   citric acid           100 non-null    float64
 3   residual sugar        100 non-null    float64
 4   chlorides             100 non-null    float64
 5   free sulfur dioxide   100 non-null    float64
 6   total sulfur dioxide  100 non-null    float64
 7   density               100 non-null    float64
 8   pH                    100 non-null    float64
 9   sulphates             100 non-null    float64
 10  alcohol               100 non-null    float64
 11  quality               100 non-null    int64  
dtypes: float64(11), int64(1)
memory usage: 9.5 KB
```

In [5]:
```
df.describe()
```

Out[5]:

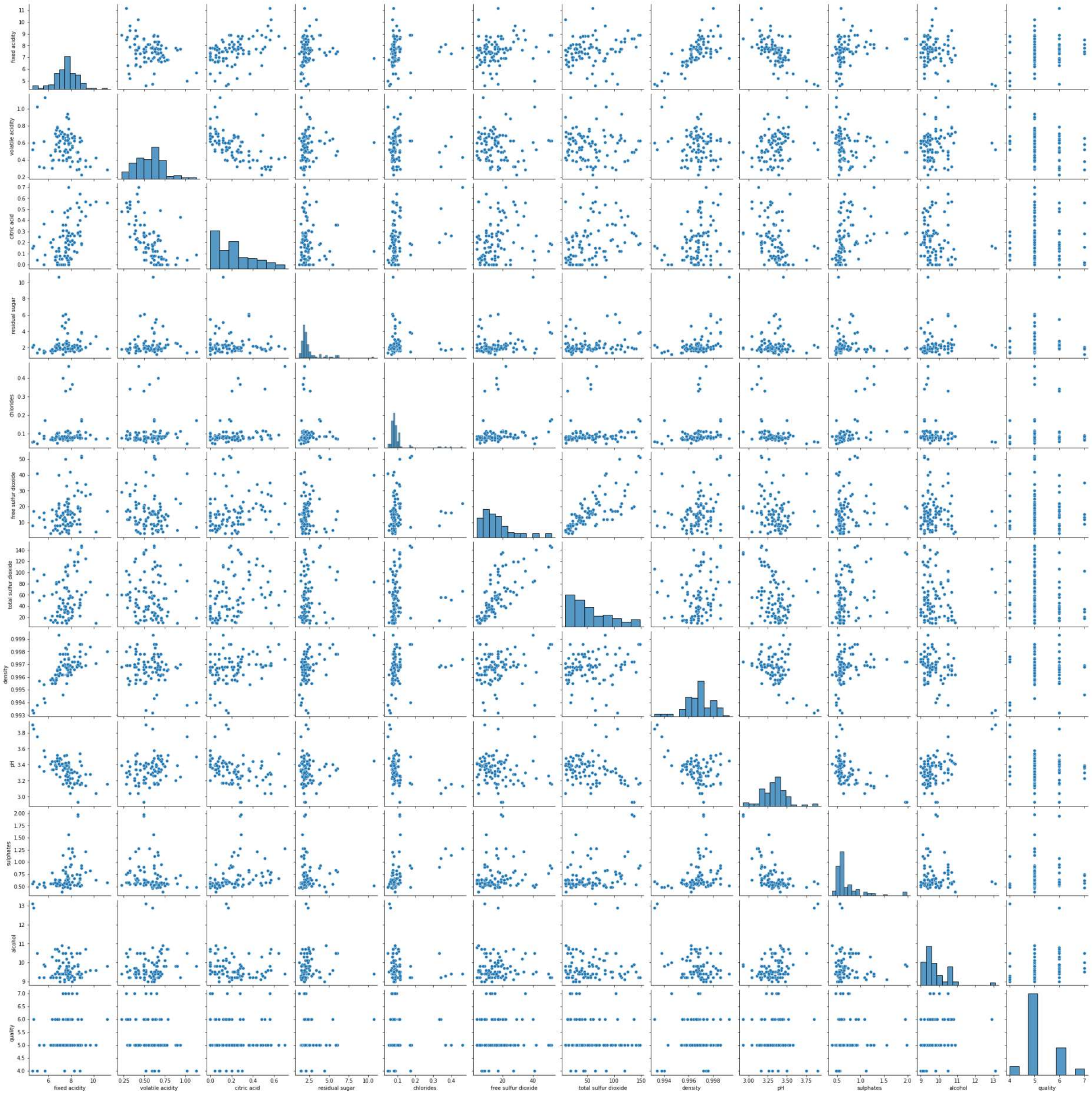
	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	d
count	100.000000	100.000000	100.000000	100.000000	100.000000	100.000000	100.00000	100.(
mean	7.580000	0.560500	0.217700	2.458000	0.100560	15.950000	56.39000	0.9
std	1.065909	0.164431	0.178336	1.361886	0.068928	10.575725	37.36684	0.(
min	4.600000	0.220000	0.000000	1.200000	0.045000	3.000000	10.00000	0.9
25%	7.000000	0.430000	0.067500	1.800000	0.074000	9.000000	26.25000	0.9
50%	7.650000	0.570000	0.195000	2.000000	0.082000	13.500000	46.00000	0.9
75%	8.100000	0.662500	0.302500	2.400000	0.097250	19.250000	82.25000	0.9
max	11.200000	1.130000	0.700000	10.700000	0.464000	52.000000	148.00000	0.9

In [6]:
```
df.columns
```

Out[6]:
```
Index(['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
       'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
       'pH', 'sulphates', 'alcohol', 'quality'],
      dtype='object')
```

In [7]:
```
sns.pairplot(df)
```

Out[7]:
```
<seaborn.axisgrid.PairGrid at 0x2aff1e56dc0>
```



In [8]:
```
da=df[['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
       'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
       'pH', 'sulphates', 'alcohol', 'quality']]
da
```

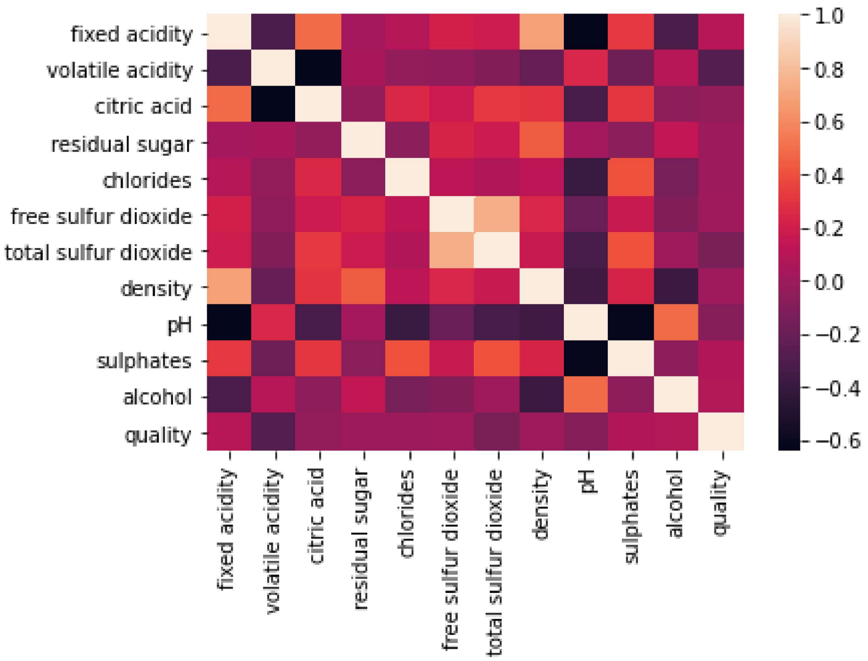
Out[8]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcoho
0	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.
1	7.8	0.880	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.
2	7.8	0.760	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.
3	11.2	0.280	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.
4	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.
...	...	...	...	...	...	...	...	...	...	...	.
95	4.7	0.600	0.17	2.3	0.058	17.0	106.0	0.9932	3.85	0.60	12.
96	6.8	0.775	0.00	3.0	0.102	8.0	23.0	0.9965	3.45	0.56	10.
97	7.0	0.500	0.25	2.0	0.070	3.0	22.0	0.9963	3.25	0.63	9.
98	7.6	0.900	0.06	2.5	0.079	5.0	10.0	0.9967	3.39	0.56	9.
99	8.1	0.545	0.18	1.9	0.080	13.0	35.0	0.9972	3.30	0.59	9.

100 rows × 12 columns

In [9]:
```
sns.heatmap(da.corr())
```

Out[9]:
```
<AxesSubplot:>
```



In [10]:
```
x=da[['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
       'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
       'pH', 'sulphates', 'alcohol']]
y=da['quality']
```

In [11]:
```
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 [12]:
```
from sklearn.linear_model import LinearRegression

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

Out[12]:
```
LinearRegression()
```

In [13]:
```
print(lr.intercept_)
```

```
135.30611172950282
```

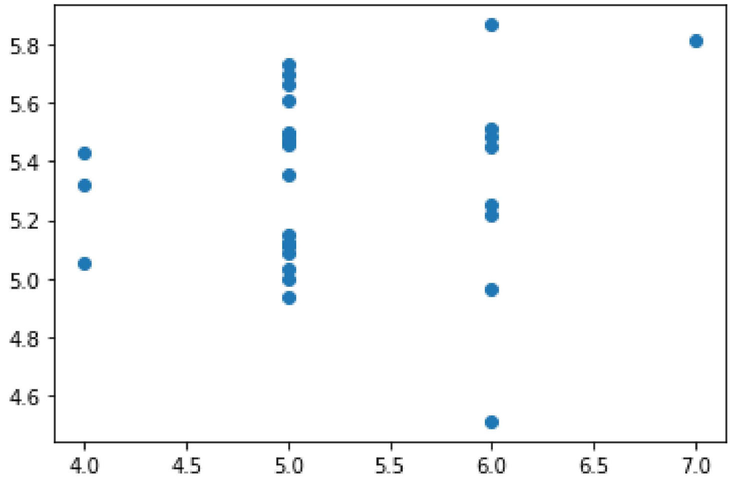
In [14]:
```
coeff = pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[14]:

	Co-efficient
fixed acidity	0.165743
volatile acidity	-1.776040
citric acid	-0.921176
residual sugar	0.073947
chlorides	-0.641759
free sulfur dioxide	0.013480
total sulfur dioxide	-0.007232
density	-129.110503
pH	-0.256082
sulphates	0.336342
alcohol	-0.071954

In [15]:
```
prediction=lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[15]:
```
<matplotlib.collections.PathCollection at 0x2affb0a8c10>
```



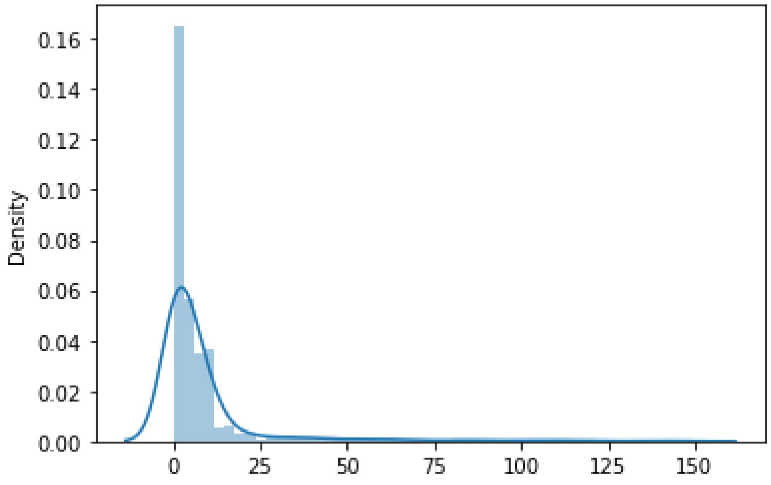
In [16]:
```
print(lr.score(x_test,y_test))
```

```
-0.10268722678281694
```

In [17]:
```
sns.distplot(da)
```

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

Out[17]:
```
<AxesSubplot:ylabel='Density'>
```



In [18]:
```
print(lr.score(x_train,y_train))
```

```
0.29650522054722706
```

```
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]: -0.14209998664908996
```

```
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]: -0.001272329380629822
```

```
In [ ]:
```