In [3]:
```
# import libaries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [51]:
```
x=pd.read_csv(r"C:\Users\user\Downloads\11_winequality-red - 11_winequality-re
```

Out[51]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcc
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 [52]:
```
x=x.head(10)
```

Out[52]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4
5	7.4	0.66	0.00	1.8	0.075	13.0	40.0	0.9978	3.51	0.56	9.4
6	7.9	0.60	0.06	1.6	0.069	15.0	59.0	0.9964	3.30	0.46	9.4
7	7.3	0.65	0.00	1.2	0.065	15.0	21.0	0.9946	3.39	0.47	10.0
8	7.8	0.58	0.02	2.0	0.073	9.0	18.0	0.9968	3.36	0.57	9.5
9	7.5	0.50	0.36	6.1	0.071	17.0	102.0	0.9978	3.35	0.80	10.5

In [53]:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   fixed acidity         10 non-null     float64
 1   volatile acidity      10 non-null     float64
 2   citric acid           10 non-null     float64
 3   residual sugar        10 non-null     float64
 4   chlorides             10 non-null     float64
 5   free sulfur dioxide   10 non-null     float64
 6   total sulfur dioxide  10 non-null     float64
 7   density               10 non-null     float64
 8   pH                    10 non-null     float64
 9   sulphates             10 non-null     float64
 10  alcohol               10 non-null     float64
 11  quality               10 non-null     int64  
dtypes: float64(11), int64(1)
memory usage: 1.1 KB
```

In [75]:

Out[75]:

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

In [77]:

```
d=x[['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
       'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
       'pH']]
```

Out[77]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH
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
5	7.4	0.66	0.00	1.8	0.075	13.0	40.0	0.9978	3.51
6	7.9	0.60	0.06	1.6	0.069	15.0	59.0	0.9964	3.30
7	7.3	0.65	0.00	1.2	0.065	15.0	21.0	0.9946	3.39
8	7.8	0.58	0.02	2.0	0.073	9.0	18.0	0.9968	3.36
9	7.5	0.50	0.36	6.1	0.071	17.0	102.0	0.9978	3.35

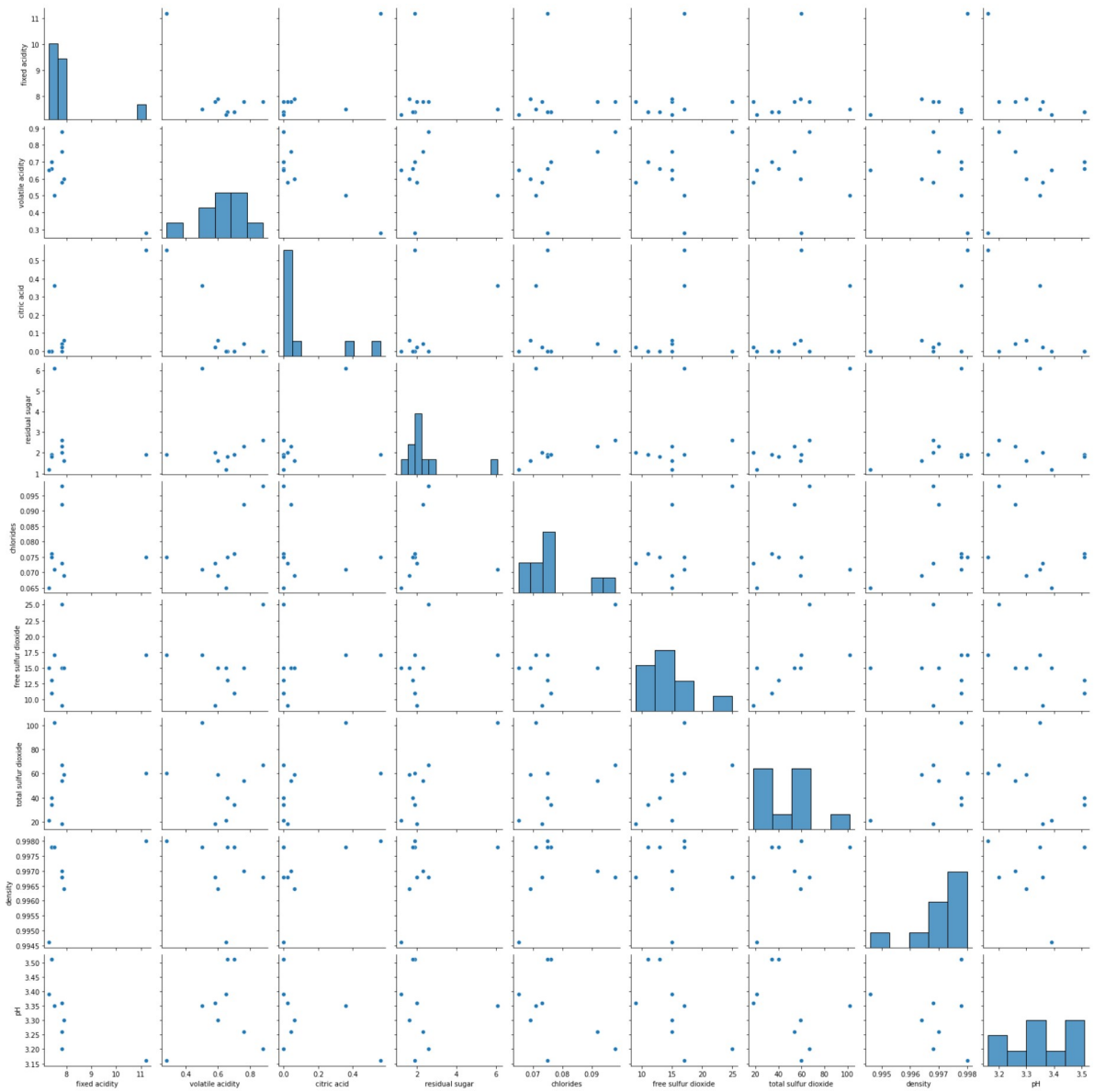
In [78]:

Out[78]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	
count	10.000000	10.000	10.000000	10.000000	10.000000	10.000000	10.000000	10.000000	1(
mean	7.950000	0.631	0.104000	2.330000	0.077000	14.800000	48.900000	0.997080	3
std	1.162612	0.161	0.194548	1.376025	0.010198	4.467164	25.066356	0.001038	(
min	7.300000	0.280	0.000000	1.200000	0.065000	9.000000	18.000000	0.994600	3
25%	7.400000	0.585	0.000000	1.825000	0.071500	11.500000	34.000000	0.996800	3
50%	7.650000	0.655	0.010000	1.900000	0.075000	15.000000	47.000000	0.997400	3
75%	7.800000	0.700	0.055000	2.225000	0.076000	16.500000	59.750000	0.997800	3
max	11.200000	0.880	0.560000	6.100000	0.098000	25.000000	102.000000	0.998000	3

In [79]:

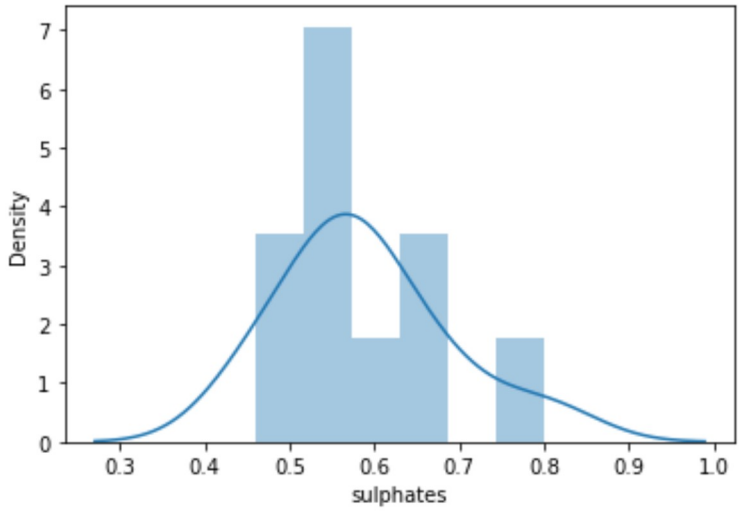
Out[79]: <seaborn.axisgrid.PairGrid at 0x1e647eebc40>



In [81]:

```
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[81]: <AxesSubplot:xlabel='sulphates', ylabel='Density'>

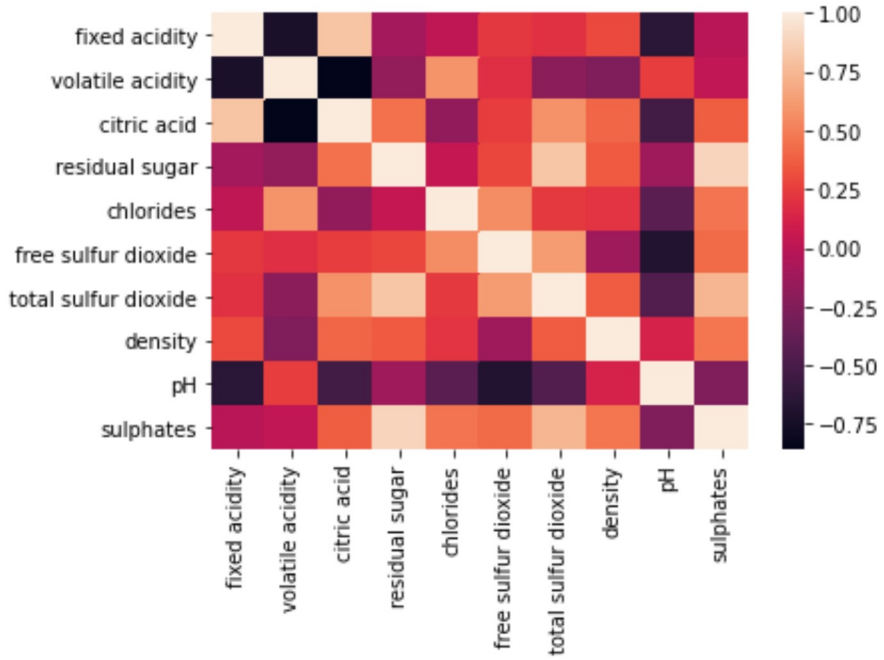


In [82]:

```
x1=x[['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
       'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
```

In [83]:

Out[83]: <AxesSubplot:>



In [85]:
```
x=x1[[ 'fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
       'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
       'pH', 'sulphates',  ]]
```

In [86]:
```
# to split my dataset into traning and test date

from sklearn.model_selection import train_test_split
```

In [87]:
```
from sklearn.linear_model import LinearRegression

lr=LinearRegression()
```

Out[87]:
```
LinearRegression()
```

In [88]:

```
-0.9703780205212952
```

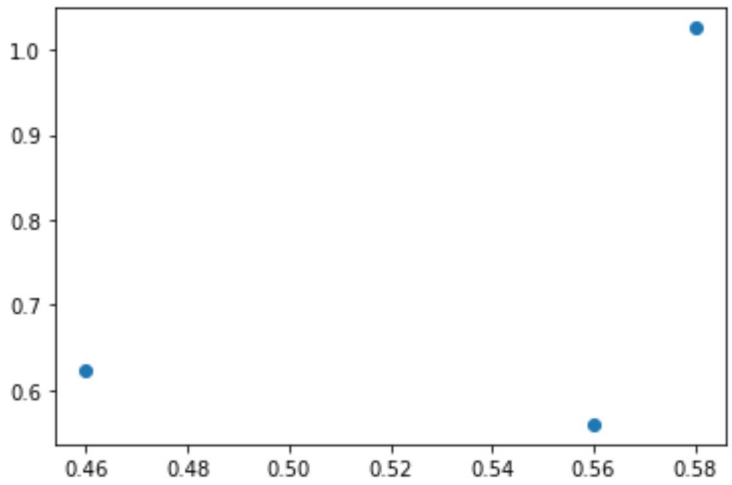
In [89]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
```

Out[89]:

	Co-efficient
fixed acidity	0.141824
volatile acidity	0.098129
citric acid	-0.078670
residual sugar	0.037998
chlorides	0.010764
free sulfur dioxide	-0.001724
total sulfur dioxide	0.001851
density	0.002353
pH	0.076197
sulphates	0.045368

In [90]:
```
prediction=lr.predict(x_test)
```

Out[90]: <matplotlib.collections.PathCollection at 0x1e65c254940>



In [91]:

Out[91]: -26.270529520100197

In [92]:

Out[92]: 1.0

In [93]:

In [94]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[94]: -4.562504032997997

In [95]:
```
la=Lasso(alpha=10)
```

Out[95]: Lasso(alpha=10)

In [96]:

Out[96]: -2.2950131665569473

In [ ]: