mk 27-07-2023

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [67]:
```
a=pd.read_csv(r"C:\Users\user\Downloads\11_winequality-red.csv")
a
```

	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	

In [68]:
```
a=a.head(200)
a
```

Out[68]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcoh
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
...	...	...	...	...	...	...	...	...	...	...	
195	7.8	0.590	0.33	2.0	0.074	24.0	120.0	0.9968	3.25	0.54	9
196	7.3	0.580	0.30	2.4	0.074	15.0	55.0	0.9968	3.46	0.59	10
197	11.5	0.300	0.60	2.0	0.067	12.0	27.0	0.9981	3.11	0.97	10
198	5.4	0.835	0.08	1.2	0.046	13.0	93.0	0.9924	3.57	0.85	13
199	6.9	1.090	0.06	2.1	0.061	12.0	31.0	0.9948	3.51	0.43	11

200 rows × 12 columns

In [69]:
```
a.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   fixed acidity         200 non-null    float64
 1   volatile acidity      200 non-null    float64
 2   citric acid           200 non-null    float64
 3   residual sugar        200 non-null    float64
 4   chlorides             200 non-null    float64
 5   free sulfur dioxide   200 non-null    float64
 6   total sulfur dioxide  200 non-null    float64
 7   density               200 non-null    float64
 8   pH                    200 non-null    float64
 9   sulphates             200 non-null    float64
 10  alcohol               200 non-null    float64
 11  quality               200 non-null    int64  
dtypes: float64(11), int64(1)
memory usage: 18.9 KB
```

In [70]:
```
a.columns
```

Out[70]:
```
Index(['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
       'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
       'pH', 'sulphates', 'alcohol', 'quality'],
      dtype='object')
```

In [71]:
```
a.head()
```

Out[71]:

	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

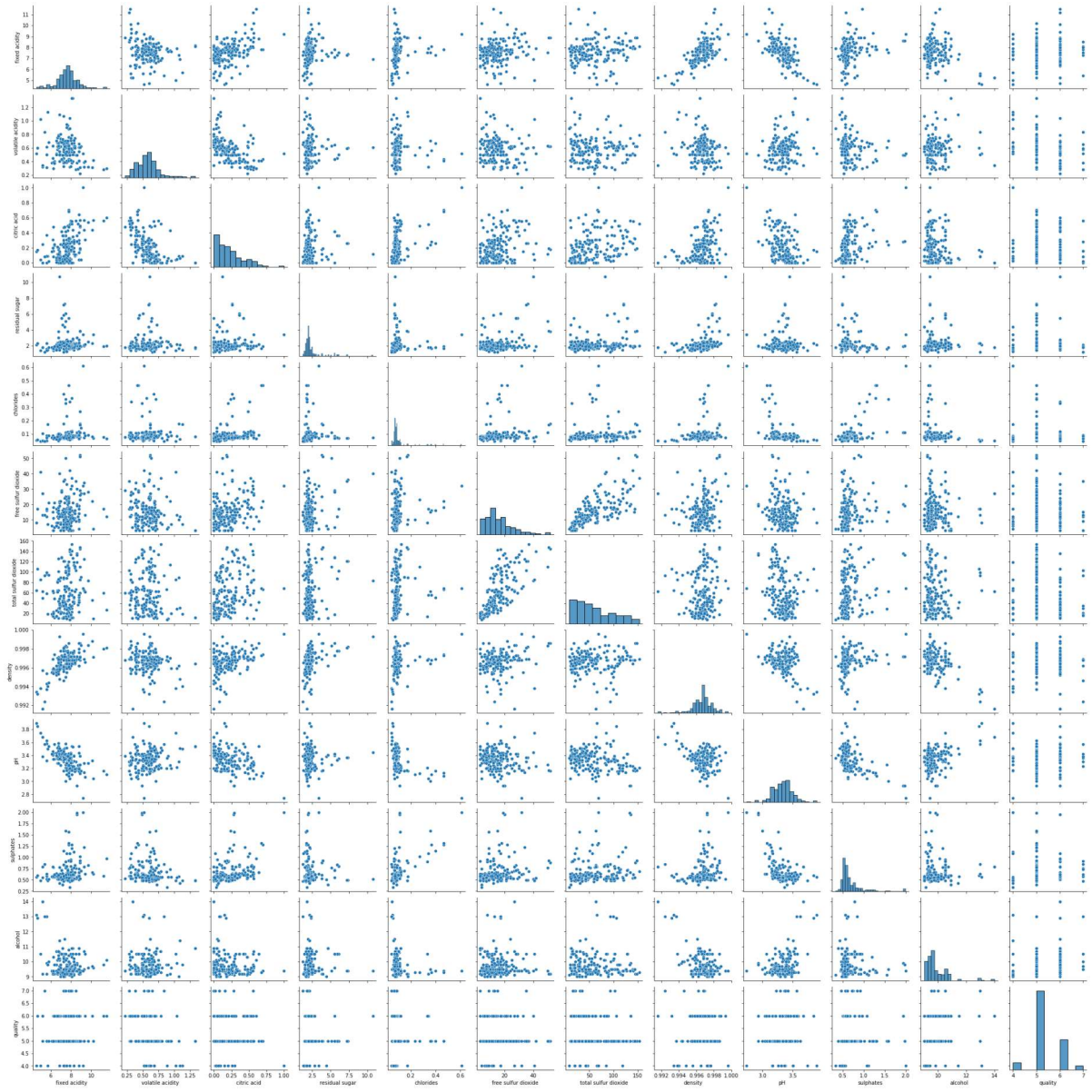
In [72]:
```
a.describe()
```

Out[72]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	
count	200.000000	200.000000	200.000000	200.000000	200.000000	200.000000	200.000000	200.
mean	7.594000	0.574350	0.215000	2.368000	0.098895	15.610000	60.530000	0.
std	1.013707	0.177606	0.182107	1.217441	0.072496	9.394102	38.097061	0.
min	4.600000	0.220000	0.000000	1.200000	0.045000	3.000000	8.000000	0.
25%	7.100000	0.450000	0.067500	1.800000	0.073000	9.000000	29.750000	0.
50%	7.650000	0.567500	0.180000	2.000000	0.081000	14.000000	52.000000	0.
75%	8.100000	0.662500	0.310000	2.300000	0.092000	19.250000	89.000000	0.
max	11.500000	1.330000	1.000000	10.700000	0.610000	52.000000	153.000000	0.

In [73]:
```
sns.pairplot(a)
```

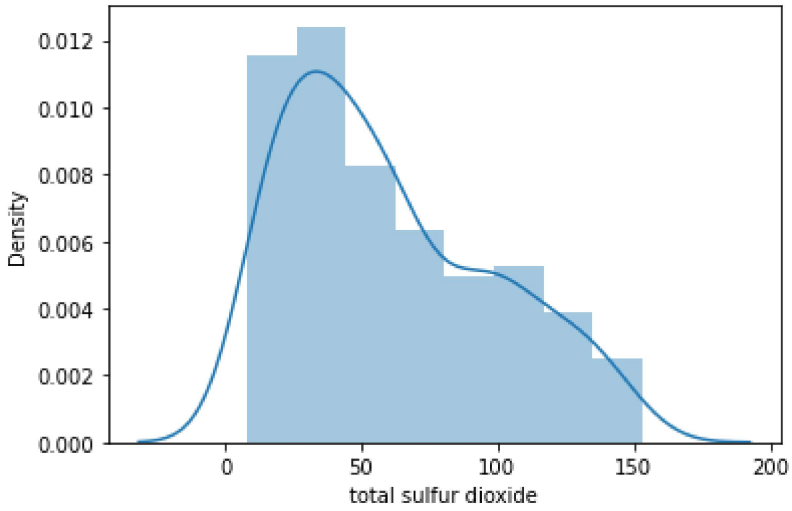
Out[73]:
```
<seaborn.axisgrid.PairGrid at 0x25f082245b0>
```



In [78]:
```
sns.distplot(a['total sulfur dioxide'])
```

```
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)
```

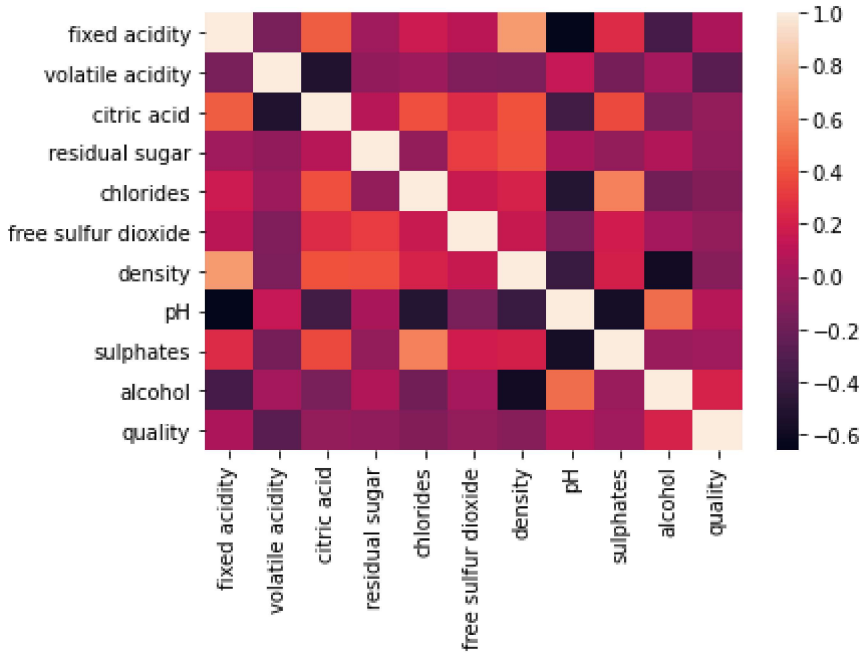
Out[78]: <AxesSubplot:xlabel='total sulfur dioxide', ylabel='Density'>



In [79]:
```
x1=a[['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
       'chlorides', 'free sulfur dioxide', 'density',
       'pH', 'sulphates', 'alcohol', 'quality']]
```

In [80]:
```
sns.heatmap(x1.corr())
```

Out[80]: <AxesSubplot:>



In [83]:
```
x=a[['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
       'chlorides', 'free sulfur dioxide',  'density',
       'pH', 'sulphates', 'alcohol', 'quality']]
y=a['total sulfur dioxide']
```

In [84]:
```
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 [85]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[85]:
```
LinearRegression()
```

In [86]:
```
print(lr.intercept_)
```

```
-1722.359468332406
```

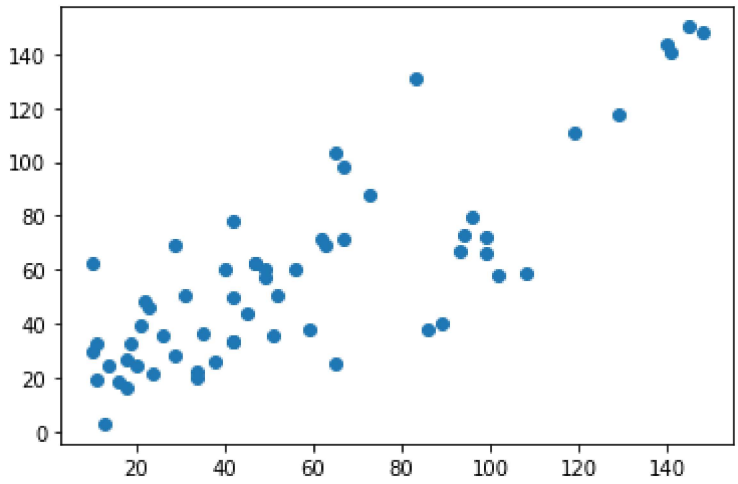
In [87]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[87]:

	Co-efficient
fixed acidity	-13.753146
volatile acidity	33.001623
citric acid	69.518310
residual sugar	1.508975
chlorides	-189.985090
free sulfur dioxide	2.638808
density	2249.745157
pH	-141.080896
sulphates	-1.743660
alcohol	7.457486
quality	-3.065147

In [88]:
```
prediction=lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[88]: <matplotlib.collections.PathCollection at 0x25f0f683f40>



In [89]:
```
print(lr.score(x_test,y_test))
```

```
0.6430412535339495
```