kaviyadevi 20106064

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

In [2]:
```
#to import dataset
data1=pd.read_csv(r"C:\Users\user\Downloads\11_winequality-red - 11_winequality-r
data1
```

Out[2]:

	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	0.99780	3.51	0.56	9
1	7.8	0.880	0.00	2.6	0.098	25.0	67.0	0.99680	3.20	0.68	9
2	7.8	0.760	0.04	2.3	0.092	15.0	54.0	0.99700	3.26	0.65	9
3	11.2	0.280	0.56	1.9	0.075	17.0	60.0	0.99800	3.16	0.58	9
4	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	0.56	9
...	...	...	...	...	...	...	...	...	...	...	
1594	6.2	0.600	0.08	2.0	0.090	32.0	44.0	0.99490	3.45	0.58	10
1595	5.9	0.550	0.10	2.2	0.062	39.0	51.0	0.99512	3.52	0.76	11
1596	6.3	0.510	0.13	2.3	0.076	29.0	40.0	0.99574	3.42	0.75	11
1597	5.9	0.645	0.12	2.0	0.075	32.0	44.0	0.99547	3.57	0.71	10
1598	6.0	0.310	0.47	3.6	0.067	18.0	42.0	0.99549	3.39	0.66	11

1599 rows × 12 columns

In [3]:
```
#to display top 5 rows
data=data1.head()
data
```

Out[3]:

	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

DATA CLEANING AND PREPROCESSING

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5 entries, 0 to 4
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   fixed acidity         5 non-null      float64
 1   volatile acidity      5 non-null      float64
 2   citric acid           5 non-null      float64
 3   residual sugar        5 non-null      float64
 4   chlorides             5 non-null      float64
 5   free sulfur dioxide   5 non-null      float64
 6   total sulfur dioxide  5 non-null      float64
 7   density               5 non-null      float64
 8   pH                    5 non-null      float64
 9   sulphates             5 non-null      float64
 10  alcohol               5 non-null      float64
 11  quality               5 non-null      int64  
dtypes: float64(11), int64(1)
memory usage: 608.0 bytes
```

In [5]:
```
#to display summary of statistics
data.describe()
```

Out[5]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH
count	5.000000	5.000000	5.000000	5.000000	5.000000	5.000000	5.000000	5.00000	5.000000
mean	8.320000	0.664000	0.120000	2.120000	0.083400	15.800000	49.800000	0.99748	3.328000
std	1.622344	0.226892	0.246577	0.319374	0.010807	5.761944	15.139353	0.00054	0.169912
min	7.400000	0.280000	0.000000	1.900000	0.075000	11.000000	34.000000	0.99680	3.160000
25%	7.400000	0.700000	0.000000	1.900000	0.076000	11.000000	34.000000	0.99700	3.200000
50%	7.800000	0.700000	0.000000	1.900000	0.076000	15.000000	54.000000	0.99780	3.260000
75%	7.800000	0.760000	0.040000	2.300000	0.092000	17.000000	60.000000	0.99780	3.510000
max	11.200000	0.880000	0.560000	2.600000	0.098000	25.000000	67.000000	0.99800	3.510000

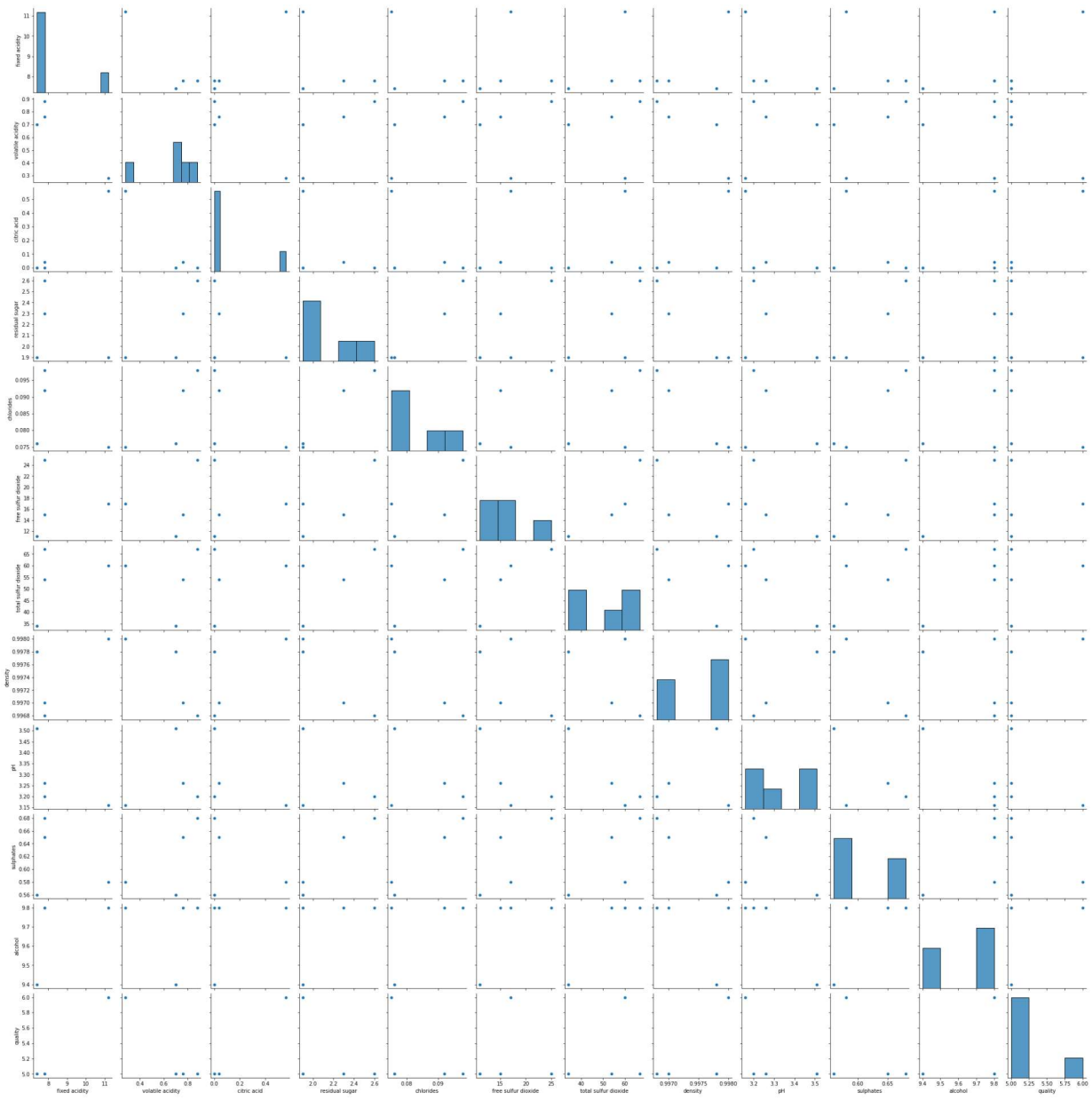
In [6]:
```
#to display the column heading
data.columns
```

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')
```

EDA and DATA VISUALIZATION

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

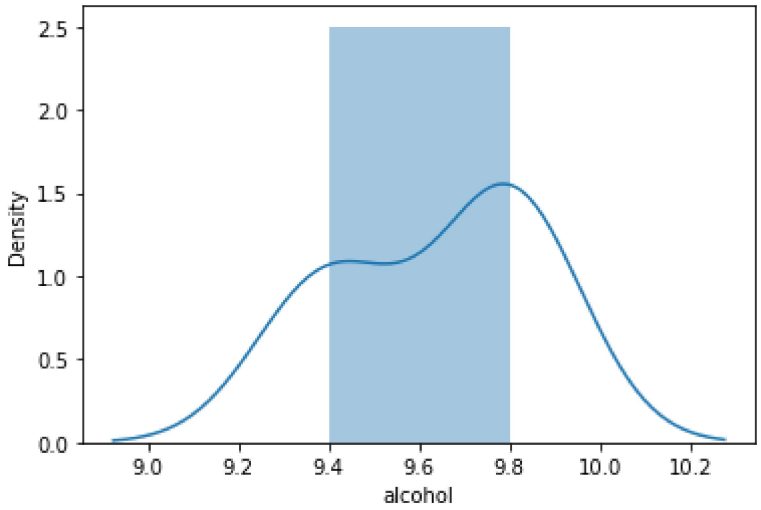
Out[7]: <seaborn.axisgrid.PairGrid at 0x1e8eebb1b50>



In [8]:
```
sns.distplot(data['alcohol'])
```

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

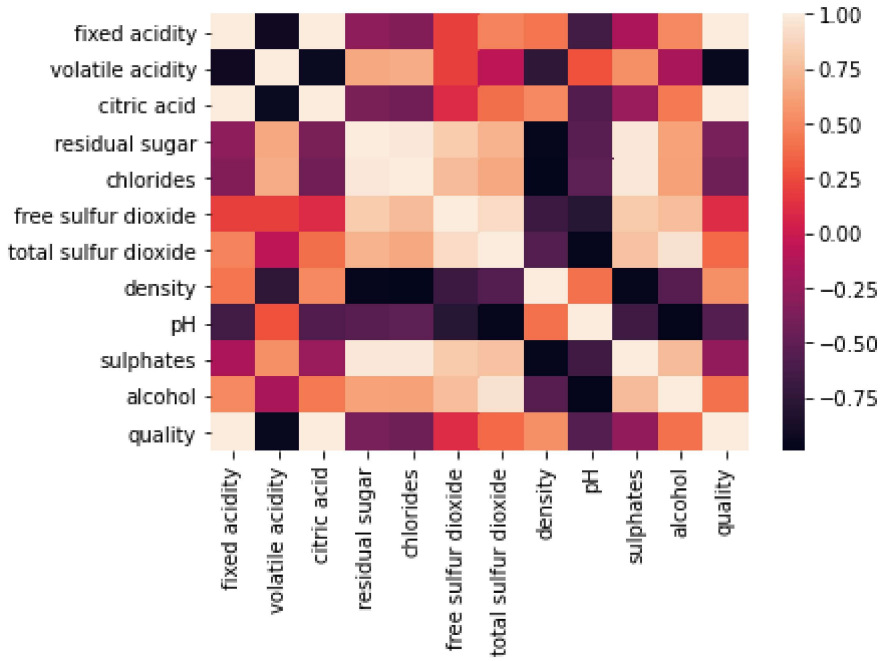
Out[8]: <AxesSubplot:xlabel='alcohol', ylabel='Density'>



In [9]:
```
df=data[['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
       'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
       'pH', 'sulphates', 'alcohol', 'quality']]
```

In [10]:
```
sns.heatmap(df.corr())
```

Out[10]: <AxesSubplot:>



TRAINING MODEL

In [11]:
```
x=df[['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
       'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
       'pH', 'sulphates','alcohol']]
y=df[['quality']]
```

In [12]:
```
#to split my dataset into trainning and test
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)
```

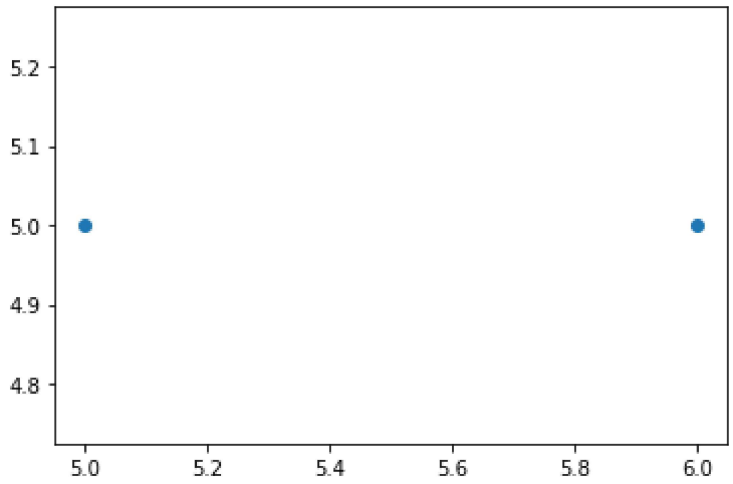
Out[13]:
```
LinearRegression()
```

In [14]:
```
#to find intercept
print(lr.intercept_)
```

```
[5.]
```

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

Out[15]:
```
<matplotlib.collections.PathCollection at 0x1e8f6ce71f0>
```



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

```
-1.0
```

RIDGE AND LASSO REGRESSION

In [17]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [18]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[18]:
```
Ridge(alpha=10)
```

In [19]:
```
rr.score(x_test,y_test)
```

Out[19]:
```
-1.0
```

In [20]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_coordinate_des
cent.py:530: ConvergenceWarning: Objective did not converge. You might want to
increase the number of iterations. Duality gap: 0.0, tolerance: 0.0
  model = cd_fast.enet_coordinate_descent(
```

Out[20]:
```
Lasso(alpha=10)
```

In [21]:
```
la.score(x_test,y_test)
```

Out[21]:
```
-1.0
```

In [22]:
```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_coordinate_des
cent.py:530: ConvergenceWarning: Objective did not converge. You might want to
increase the number of iterations. Duality gap: 0.0, tolerance: 0.0
  model = cd_fast.enet_coordinate_descent(
```

Out[22]:
```
ElasticNet()
```

In [23]:
```
print(en.coef_)
```

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

In [24]:
```
print(en.predict(x_test))
```

```
[5. 5.]
```

In [25]:
```
print(en.score(x_test,y_test))
```

```
-1.0
```

In [26]:
```
from sklearn import metrics
```

In [27]:
```
print("Mean Absolute error",metrics.mean_absolute_error(y_test,prediction))
```

```
Mean Absolute error 0.5
```

In [28]:
```
print("Mean Squared error",metrics.mean_squared_error(y_test,prediction))
```

```
Mean Squared error 0.5
```

In [29]:
```
print("Root Mean Absolute error",np.sqrt(metrics.mean_squared_error(y_test,predic
```

```
Root Mean Absolute error 0.7071067811865476
```