DAY 10:

winequality Dataset

In [1]:

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

In [2]:

```
df=pd.read_csv(r"C:\Users\user\Downloads\11_winequality-red.csv")[0:500]
df
```

Out[2]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	al
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	
...	...	...	...	...	...	...	...	...	...	...	
495	10.7	0.35	0.53	2.6	0.070	5.0	16.0	0.9972	3.15	0.65	
496	7.8	0.52	0.25	1.9	0.081	14.0	38.0	0.9984	3.43	0.65	
497	7.2	0.34	0.32	2.5	0.090	43.0	113.0	0.9966	3.32	0.79	
498	10.7	0.35	0.53	2.6	0.070	5.0	16.0	0.9972	3.15	0.65	
499	8.7	0.69	0.31	3.0	0.086	23.0	81.0	1.0002	3.48	0.74	

500 rows × 12 columns

In [3]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   fixed acidity         500 non-null    float64
 1   volatile acidity      500 non-null    float64
 2   citric acid           500 non-null    float64
 3   residual sugar        500 non-null    float64
 4   chlorides             500 non-null    float64
 5   free sulfur dioxide   500 non-null    float64
 6   total sulfur dioxide  500 non-null    float64
 7   density               500 non-null    float64
 8   pH                    500 non-null    float64
 9   sulphates             500 non-null    float64
 10  alcohol               500 non-null    float64
 11  quality               500 non-null    int64  
dtypes: float64(11), int64(1)
memory usage: 47.0 KB
```

In [4]:

```
df.columns
```

Out[4]:

```
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')
```

Linear Regression

In [6]:

```
x=df[['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
       'chlorides', 'free sulfur dioxide']]
y=df[ 'quality']
```

In [7]:

```
# to split my dataset into test and train data
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 [8]:

```
from sklearn.linear_model import LinearRegression

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

Out[8]:

```
LinearRegression()
```

In [9]:

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

```
0.16091846851024572
```

In [10]:

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

Out[10]:

```
0.1504275401993429
```

Ridge Regression

In [11]:

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

In [12]:

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

Out[12]:

```
0.14179907392676827
```

Lasso Regression

In [13]:

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

Out[13]:

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

In [14]:

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

Out[14]:

```
-0.008272415229287011
```

Elastic regression

In [15]:

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

Out[15]:

```
ElasticNet()
```

In [16]:

```
print(en.intercept_)
```

```
5.660540623598547
```

In [17]:

```
predict=(en.predict(x_test))
```

In [18]:

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

```
0.013717829632268197
```

Evalution matrics

In [19]:

```
from sklearn import metrics
print("Mean Absolute Error:",metrics.mean_absolute_error(y_test,predict))
```

```
Mean Absolute Error: 0.6536684801802912
```

In [20]:

```
print("Mean Square Error:",metrics.mean_squared_error(y_test,predict))
```

```
Mean Square Error: 0.5606027856370186
```

In [21]:

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

```
Root Mean Square Error: 0.7487341221268192
```