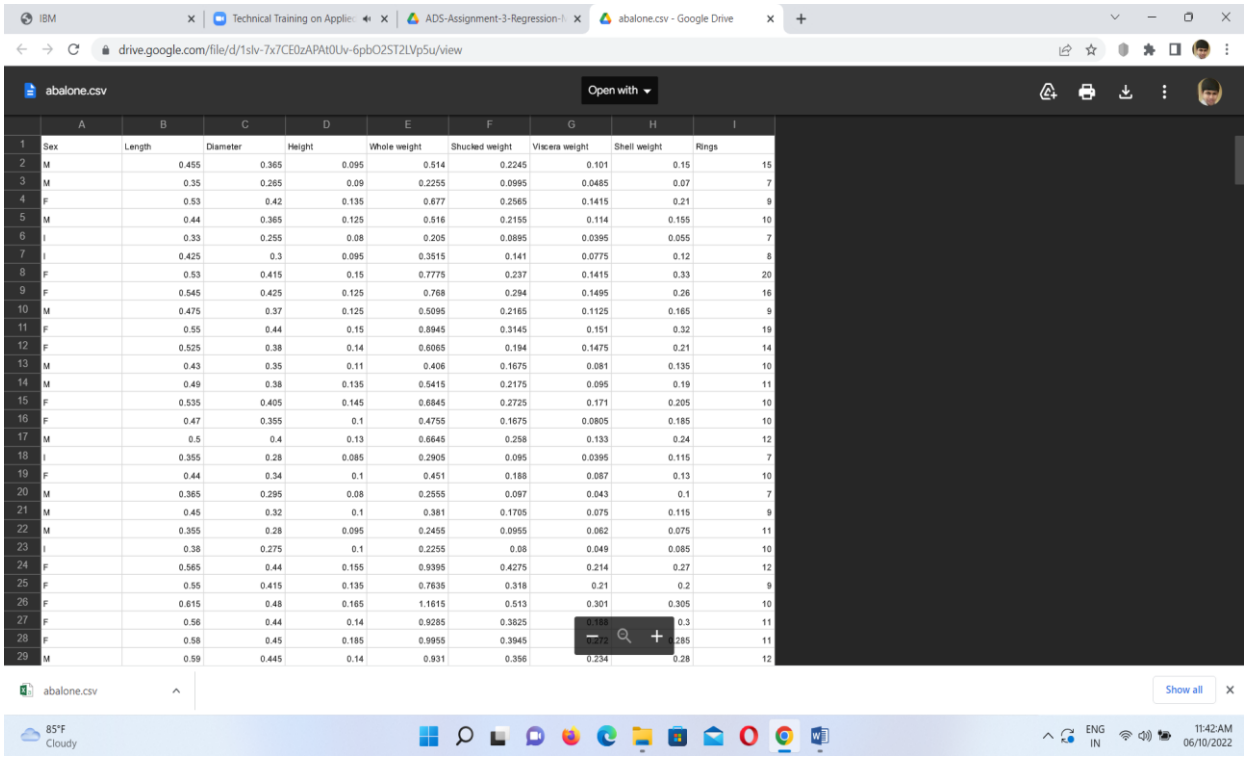
Assignment -3
Regression Model

Assignment Date	29 September 2022
Student Name	Naveen Anend S
Student Roll Number	727719EUCS098
Maximum Marks	2 Marks

Question-1:

Download the dataset: Dataset



	A	B	C	D	E	F	G	H	I
1	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
2	M	0.455	0.365	0.095	0.514	0.2245	0.101	0.15	15
3	M	0.35	0.265	0.09	0.2255	0.0995	0.0485	0.07	7
4	F	0.53	0.42	0.135	0.677	0.2565	0.1415	0.21	9
5	M	0.44	0.365	0.125	0.516	0.2155	0.114	0.155	10
6	I	0.33	0.255	0.08	0.205	0.0895	0.0395	0.055	7
7	I	0.425	0.3	0.095	0.3515	0.141	0.0775	0.12	8
8	F	0.53	0.415	0.15	0.7775	0.237	0.1415	0.33	20
9	F	0.545	0.425	0.125	0.768	0.294	0.1495	0.26	16
10	M	0.475	0.37	0.125	0.5095	0.2165	0.1125	0.165	9
11	F	0.55	0.44	0.15	0.8945	0.3145	0.151	0.32	19
12	F	0.525	0.38	0.14	0.6065	0.194	0.1475	0.21	14
13	M	0.43	0.35	0.11	0.406	0.1675	0.081	0.135	10
14	M	0.49	0.38	0.135	0.5415	0.2175	0.095	0.19	11
15	F	0.535	0.405	0.145	0.6845	0.2725	0.171	0.205	10
16	F	0.47	0.355	0.1	0.4755	0.1675	0.0805	0.185	10
17	M	0.5	0.4	0.13	0.6645	0.258	0.133	0.24	12
18	I	0.355	0.28	0.085	0.2905	0.095	0.0395	0.115	7
19	F	0.44	0.34	0.1	0.451	0.188	0.087	0.13	10
20	M	0.365	0.295	0.08	0.2555	0.097	0.043	0.1	7
21	M	0.45	0.32	0.1	0.381	0.1705	0.075	0.115	9
22	M	0.355	0.28	0.095	0.2455	0.0955	0.062	0.075	11
23	I	0.38	0.275	0.1	0.2255	0.08	0.049	0.085	10
24	F	0.565	0.44	0.155	0.9395	0.4275	0.214	0.27	12
25	F	0.55	0.415	0.135	0.7635	0.318	0.21	0.2	9
26	F	0.615	0.48	0.165	1.1615	0.513	0.301	0.305	10
27	F	0.58	0.44	0.14	0.9285	0.3825	0.188	0.3	11
28	F	0.58	0.45	0.185	0.9955	0.3945	0.272	0.285	11
29	M	0.59	0.445	0.14	0.931	0.356	0.234	0.28	12

Question-2:

Load the dataset.

Solution:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
data = pd.read_csv("E://abalone (1).csv")
data.head()
```

```
In [1]: import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import seaborn as sns
        data = pd.read_csv("E://abalone (1).csv")
        data.head()
```

Out[1]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	M	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150	15
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	7
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	9
3	M	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	10
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	7

Question-3:

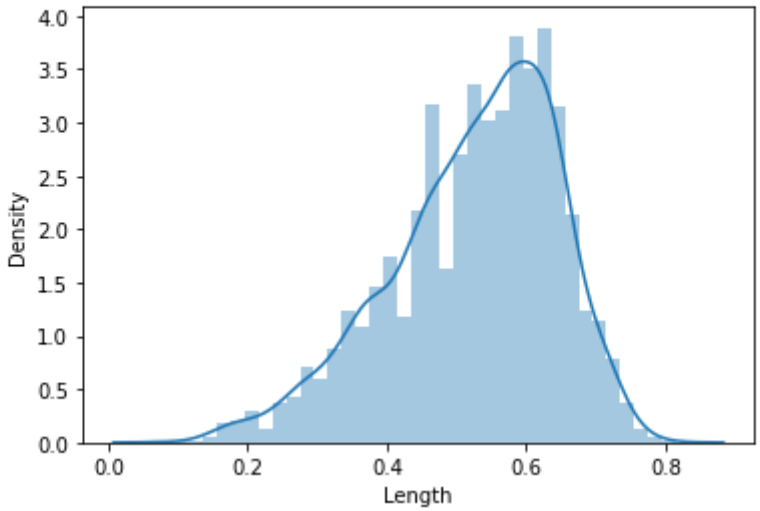
Perform Below Visualizations.

· Univariate Analysis

Solution:

```
In [4]: sns.distplot(data.Length)
```

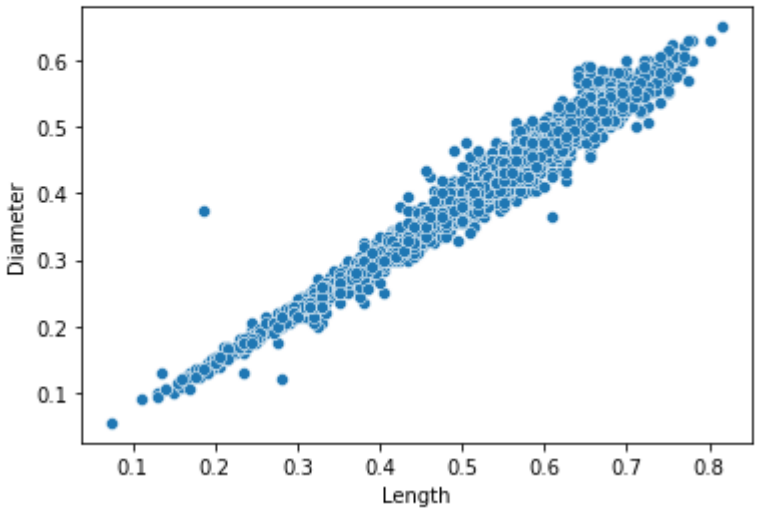
```
Out[4]: <AxesSubplot:xlabel='Length', ylabel='Density'>
```



· Bi-Variate Analysis

Solution:

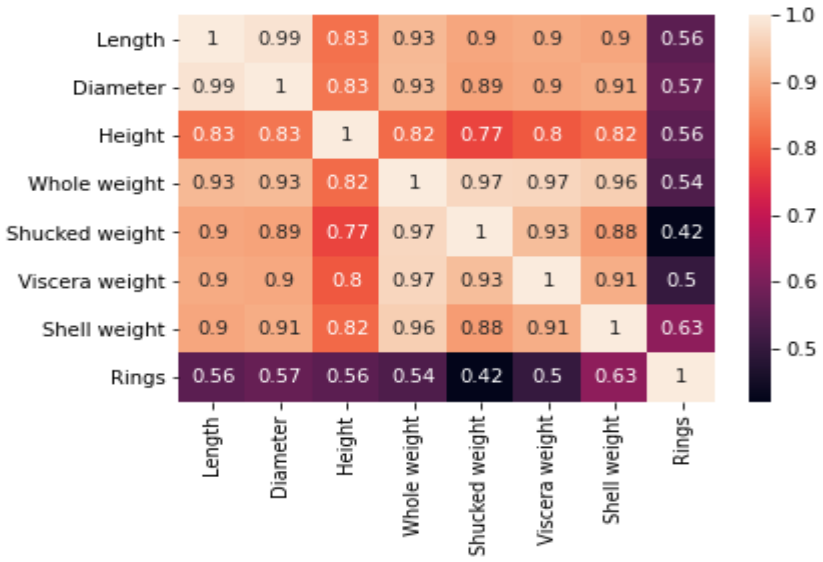
```
In [10]: sns.scatterplot(data.Length,data.Diameter)
Out[10]: <AxesSubplot:xlabel='Length', ylabel='Diameter'>
```



· Multi-Variate Analysis

Solution:

```
In [8]: sns.heatmap(data.corr(),annot=True)
Out[8]: <AxesSubplot:>
```



Question-4:

Perform descriptive statistics on the dataset.

Solution:

```
In [12]: data['Length'].mean()
Out[12]: 0.5239920995930099

In [13]: data['Length'].median()
Out[13]: 0.545

In [14]: data['Height'].mode()
Out[14]: 0    0.15
         dtype: float64

In [15]: data.skew()
Out[15]: Length           -0.639873
         Diameter         -0.609198
         Height            3.128817
         Whole weight      0.530959
         Shucked weight    0.719098
         Viscera weight    0.591852
         Shell weight      0.620927
         Rings             1.114102
         dtype: float64

In [16]: data.kurt()
Out[16]: Length            0.064621
         Diameter         -0.045476
         Height           76.025509
         Whole weight     -0.023644
         Shucked weight    0.595124
         Viscera weight    0.084012
         Shell weight      0.531926
         Rings             2.330687
         dtype: float64

In [17]: data.var()
Out[17]: Length            0.014422
         Diameter          0.009849
         Height            0.001750
         Whole weight      0.240481
         Shucked weight    0.049268
         Viscera weight    0.012015
         Shell weight      0.019377
         Rings            10.395266
         dtype: float64
```

```
In [18]: data.std()

Out[18]: Length            0.120093
         Diameter          0.099240
         Height            0.041827
         Whole weight      0.490389
         Shucked weight    0.221963
         Viscera weight    0.109614
         Shell weight      0.139203
         Rings             3.224169
         dtype: float64
```

Question-5:

Handle the Missing values.

Solution:

```
In [19]: data.isna().any()
```

```
Out[19]: Sex               False
         Length            False
         Diameter          False
         Height            False
         Whole weight      False
         Shucked weight    False
         Viscera weight    False
         Shell weight      False
         Rings             False
         dtype: bool
```

```
In [20]: data['Diameter'].fillna(data['Diameter'].mean(),inplace=True)
         data
```

Out[20]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	M	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.1500	15
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.0700	7
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.2100	9
3	M	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.1550	10
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.0550	7
...	...	...	...	...	...	...	...	...	...
4172	F	0.565	0.450	0.165	0.8870	0.3700	0.2390	0.2490	11

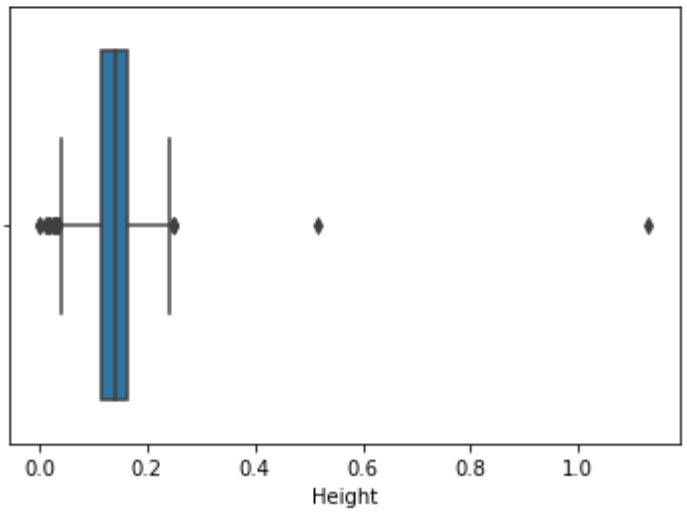
Question-6:

Find the outliers and replace the outliers

Solution:

```
In [22]: sns.boxplot(data['Height'])
```

```
Out[22]: <AxesSubplot:xlabel='Height'>
```



In [26]:
```
Q1=data.Height.quantile(0.25)
Q2=data.Height.quantile(0.75)
IQR=Q2-Q1
print(IQR)
```

```
0.05
```

In [27]:
```
data=data[~((data.Height<(Q1-1.5*IQR))|(data.Height>(Q2+1.5*IQR)))]
data
```

Out[27]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	M	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.1500	15
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.0700	7
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.2100	9
3	M	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.1550	10
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.0550	7
...	...	...	...	...	...	...	...	...	...
4172	F	0.565	0.450	0.165	0.8870	0.3700	0.2390	0.2490	11
4173	M	0.590	0.440	0.135	0.9660	0.4390	0.2145	0.2605	10
4174	M	0.600	0.475	0.205	1.1760	0.5255	0.2875	0.3080	9
4175	F	0.625	0.485	0.150	1.0945	0.5310	0.2610	0.2960	10
4176	M	0.710	0.555	0.195	1.9485	0.9455	0.3765	0.4950	12

Question-7:

Check for Categorical columns and perform encoding.

Solution:

In [28]:
```
data['Sex'].replace({'F':'Female','M':'Male'},inplace=True)
data.head()
```

Out[28]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	Male	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150	15
1	Male	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	7
2	Female	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	9
3	Male	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	10
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	7

Question-8:

Split the data into dependent and independent variables.

Solution:

```
In [29]: dt= pd.get_dummies(data,columns=['Length'])
         dt
```

Out[29]:

	Sex	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings	Length_0.135	Length_0.155	...	Length_0.74	Length_0.745	Length_0.75	Length_0.7
0	Male	0.365	0.095	0.5140	0.2245	0.1010	0.1500	15	0	0	...	0	0	0	
1	Male	0.265	0.090	0.2255	0.0995	0.0485	0.0700	7	0	0	...	0	0	0	
2	Female	0.420	0.135	0.6770	0.2565	0.1415	0.2100	9	0	0	...	0	0	0	
3	Male	0.365	0.125	0.5160	0.2155	0.1140	0.1550	10	0	0	...	0	0	0	
4	I	0.255	0.080	0.2050	0.0895	0.0395	0.0550	7	0	0	...	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
4172	Female	0.450	0.165	0.8870	0.3700	0.2390	0.2490	11	0	0	...	0	0	0	
4173	Male	0.440	0.135	0.9660	0.4390	0.2145	0.2605	10	0	0	...	0	0	0	
4174	Male	0.475	0.205	1.1760	0.5255	0.2875	0.3080	9	0	0	...	0	0	0	
4175	Female	0.485	0.150	1.0945	0.5310	0.2610	0.2960	10	0	0	...	0	0	0	
4176	Male	0.555	0.195	1.9485	0.9455	0.3765	0.4950	12	0	0	...	0	0	0	

4148 rows × 135 columns

```
In [48]: y = dt['Height']
         y
```

```
Out[48]: 0       0.095
         1       0.090
         2       0.135
         3       0.125
         4       0.080
                 ...
         4172    0.165
         4173    0.135
         4174    0.205
         4175    0.150
         4176    0.195
         Name: Height, Length: 4148, dtype: float64
```

```
In [50]: x = dt.drop(columns='Diameter',axis=1)
         x.head()
```

Out[50]:

	Sex	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings	Length_0.135	Length_0.155	Length_0.165	...	Length_0.74	Length_0.745	Length_0.75	Length_0.7
0	Male	0.095	0.5140	0.2245	0.1010	0.150	15	0	0	0	...	0	0	0	
1	Male	0.090	0.2255	0.0995	0.0485	0.070	7	0	0	0	...	0	0	0	
2	Female	0.135	0.6770	0.2565	0.1415	0.210	9	0	0	0	...	0	0	0	
3	Male	0.125	0.5160	0.2155	0.1140	0.155	10	0	0	0	...	0	0	0	
4	I	0.080	0.2050	0.0895	0.0395	0.055	7	0	0	0	...	0	0	0	

5 rows × 134 columns

Question-9:

Scale the independent variables

Solution:

```
In [51]: x=data.iloc[:,6:7].values
         from sklearn.preprocessing import StandardScaler
         std=StandardScaler()
         x=std.fit_transform(x)
         x

Out[51]: array([[-0.73839808],
                [-1.22152247],
                [-0.36570212],
                ...,
                [ 0.97784382],
                [ 0.73398103],
                [ 1.7968547 ]])
```

Question-10:

Split the data into training and testing

Solution:

```
In [52]: from sklearn.model_selection import train_test_split
         x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)

In [53]: x_train

Out[53]: array([[ 1.23090898],
                [-1.33195091],
                [-0.03441682],
                ...,
                [ 0.76158814],
                [-1.23072484],
                [-0.48533292]])

In [54]: x_test

Out[54]: array([[-7.47600447e-01],
                [ 2.22016369e+00],
                [-2.98156325e-02],
                [ 5.72939564e-01],
                [-1.14330233e+00],
                [ 2.37053081e-01],
                [ 1.22023463e-01],
                [-7.33796893e-01],
                [-1.40244066e-01],
                [-7.93612294e-01],
                [ 4.53308762e-01],
                [ 6.74165628e-01],
```

```
In [55]: y_train
```

```
Out[55]: 3780    0.185
         3161    0.070
         3919    0.135
         625     0.160
         2388    0.140
                 ...
         1042    0.195
         3289    0.160
         1667    0.180
         2630    0.095
         2756    0.130
         Name: Height, Length: 3318, dtype: float64
```

```
In [56]: y_test
```

```
Out[56]: 834     0.100
         4106    0.205
         980     0.155
         1513    0.140
         3201    0.100
                 ...
         1963    0.170
         693     0.120
         322     0.100
         1422    0.215
         803     0.100
         Name: Height, Length: 830, dtype: float64
```

Question-11:

Build the Model

Solution:

```
In [40]: from sklearn.linear_model import LinearRegression
         regressor=LinearRegression()
         regressor.fit(x_train,y_train)
```

```
Out[40]: LinearRegression()
```

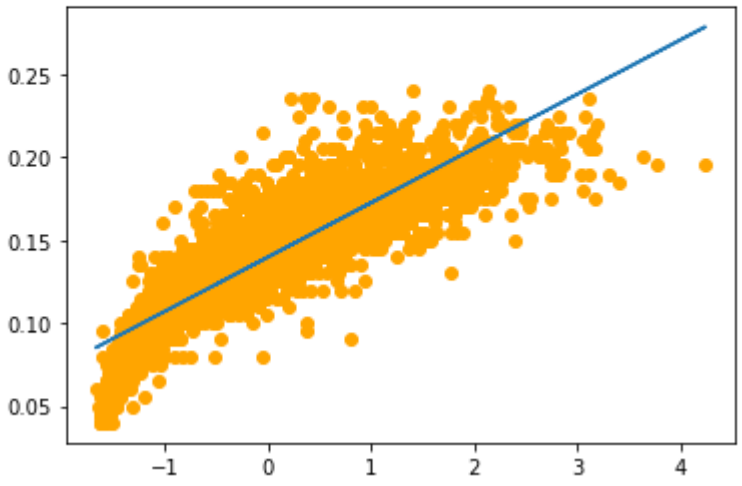
Question-12:

Train the Model

Solution:

```
In [58]: plt.scatter(x_train,y_train,color='orange')
         plt.plot(x_train,regressor.predict(x_train))

Out[58]: [<matplotlib.lines.Line2D at 0x2c176170160>]
```



Question-13:

Test the Model

Solution:

```
In [59]: y_pred=regressor.predict(x_test)
         y_pred
```

```
0.13372071, 0.11940644, 0.21101776, 0.11699562, 0.16732157,
0.12874839, 0.22261986, 0.13402206, 0.12211862, 0.14321333,
0.14592551, 0.15089783, 0.09122993, 0.09454482, 0.17952637,
0.10916044, 0.17756758, 0.13266598, 0.14426807, 0.1100645 ,
0.10283203, 0.09002452, 0.18314261, 0.13658357, 0.1472816 ,
0.13206327, 0.12663892, 0.16686955, 0.10584556, 0.14487077,
0.11126991, 0.15029513, 0.1203105 , 0.16656819, 0.11654359,
0.15421272, 0.14441874, 0.14954175, 0.12452944, 0.10720164,
0.1273923 , 0.12920042, 0.10720164, 0.10403744, 0.11353006,
0.10388676, 0.20137447, 0.15330866, 0.08595626, 0.12121456,
0.16054113, 0.15029513, 0.24808419, 0.12513215, 0.11985847,
0.19082711, 0.15767828, 0.10599623, 0.09303805, 0.16325331,
0.18329329, 0.20875762, 0.12151591, 0.14954175, 0.15767828,
0.17817028, 0.17289661, 0.11398209, 0.14803498, 0.09409279,
0.09891444, 0.14050115, 0.10192797, 0.10690029, 0.15406204,
0.11458479, 0.10222932, 0.1055442 , 0.13748762, 0.16114384,
0.10629759, 0.13537815, 0.10283203, 0.11729697, 0.16280128,
0.15165122, 0.13341936, 0.15571948, 0.11142059, 0.15828098,
0.22533203, 0.20212785, 0.16023978, 0.16023978, 0.13929574,
0.1299538 , 0.14381604, 0.10720164, 0.11609156, 0.10057188,
```

Question-14:

Measure the performance using Metrics.

Solution:

```
In [60]: from sklearn.metrics import r2_score
         a=r2_score(y_test,y_pred)
         a
```

```
Out[60]: 0.7433762312740924
```

```
In [61]: from sklearn import metrics
         np.sqrt(metrics.mean_squared_error(y_test,y_pred))
```

```
Out[61]: 0.018588576308433604
```