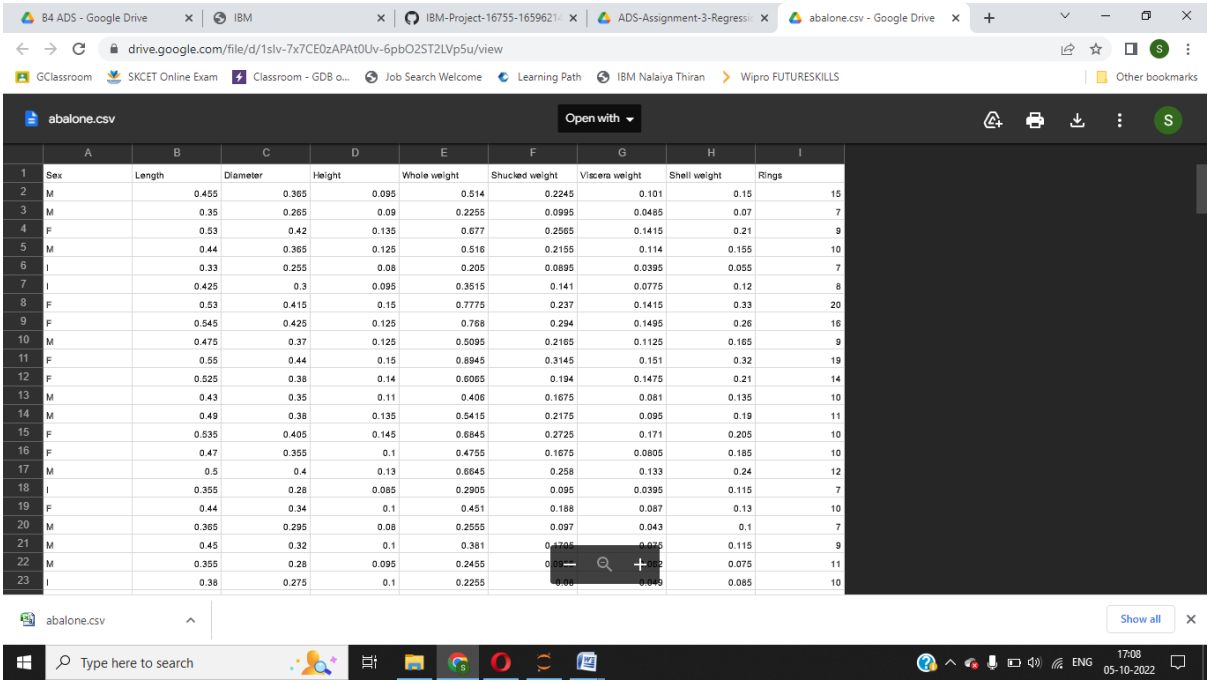
ASSIGNMENT -3
REGRESSION

Assignment Date	29 September 2022
Student Name	Manjunathan V
Student Roll Number	727719EUCS080
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	0.08	0.062	0.075	11
23	I	0.38	0.275	0.1	0.2255	0.08	0.049	0.085	10

Question-2:

Load the dataset.

Solution:

```
import pandas as pd
df=pd.read_csv("D://abalone.csv")
df.head()
```

```
import pandas as pd
df=pd.read_csv("D://abalone.csv")
df.head()
```

	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

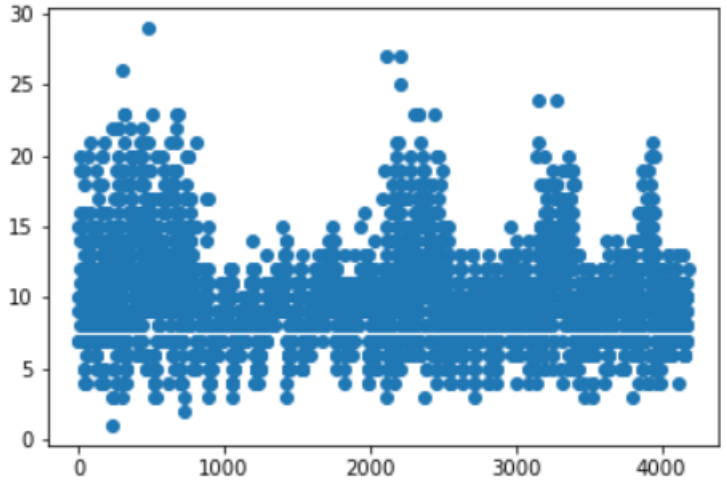
Question-3:

Perform Below Visualizations.

· Univariate Analysis

Solution:

```
import matplotlib.pyplot as plt
plt.scatter(df.index,df['Rings'])
plt.show()
```

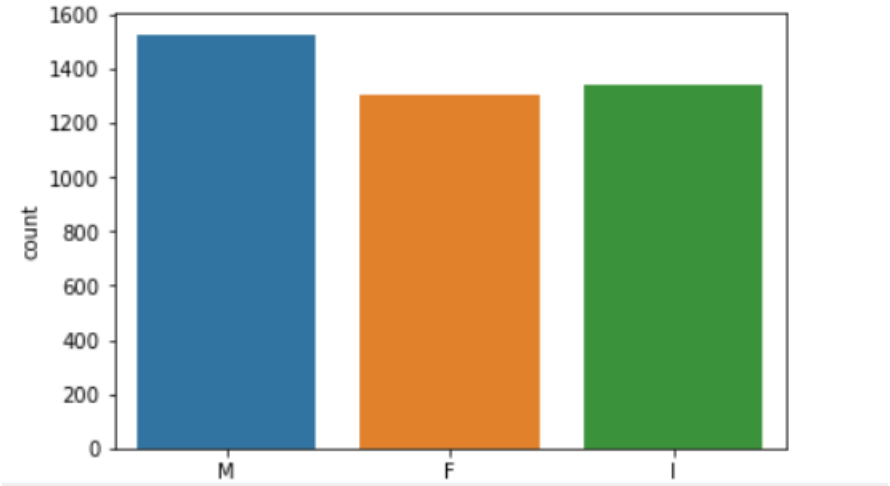


· Bi-Variate Analysis

Solution:

```
import seaborn as sns
sns.barplot(x='Sex',y='Height',data=df)
sns.countplot(x='Sex',data=df)
```

```
<AxesSubplot:xlabel='Sex', ylabel='count'>
```

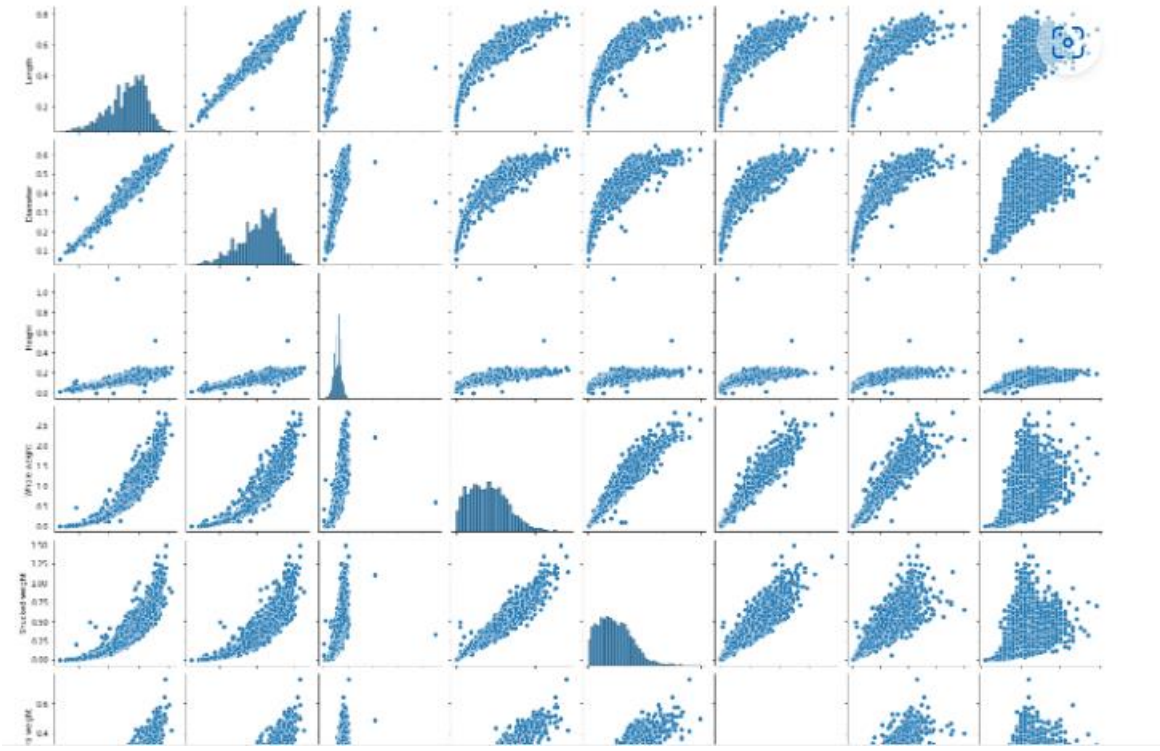


· Multi-Variate Analysis

Solution:

```
import seaborn as sns
sns.pairplot(df)
```

```
<seaborn.axisgrid.PairGrid at 0x23b7cd3b940>
```



Question-4:

Perform descriptive statistics on the dataset.

Solution:

```
df.mean()
```

```
C:\Users\Manjunathan V\AppData\Local\Temp\ipykerne
rame reductions (with 'numeric_only=None') is depr
umns before calling the reduction.
  df.mean()
```

```
Length            0.523992
Diameter          0.407881
Height            0.139516
Whole weight      0.828742
Shucked weight    0.359367
Viscera weight    0.180594
Shell weight      0.238831
Rings             9.933684
dtype: float64
```

```
df.median()
```

```
C:\Users\Manjunathan V\AppData\Local\Temp\ipykerne
ame reductions (with 'numeric_only=None') is depre
mns before calling the reduction.
  df.median()
```

```
Length            0.5450
Diameter          0.4250
Height            0.1400
Whole weight      0.7995
Shucked weight    0.3360
Viscera weight    0.1710
Shell weight      0.2340
Rings             9.0000
dtype: float64
```

```
df.mode()
```

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	M	0.550	0.45	0.15	0.2225	0.175	0.1715	0.275	9.0
1	NaN	0.625	NaN	NaN	NaN	NaN	NaN	NaN	NaN

```
df.skew()
```

```
C:\Users\Manjunathan V\AppData\Local\Temp\ipykernel_16952\1665899112.py:1: FutureWarning: D
rame reductions (with 'numeric_only=None') is deprecated; in a future version this will rai
umns before calling the reduction.
  df.skew()
```

```
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
```

```
df.kurt()
```

```
C:\Users\Manjunathan V\AppData\Local\Temp\ipykernel_1695
rame reductions (with 'numeric_only=None') is deprecated
umns before calling the reduction.
  df.kurt()
```

```
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
```

```
qu=df['Rings'].quantile(q=[0.75,0.25])
qu
```

```
0.75    11.0
0.25     8.0
Name: Rings, dtype: float64
```

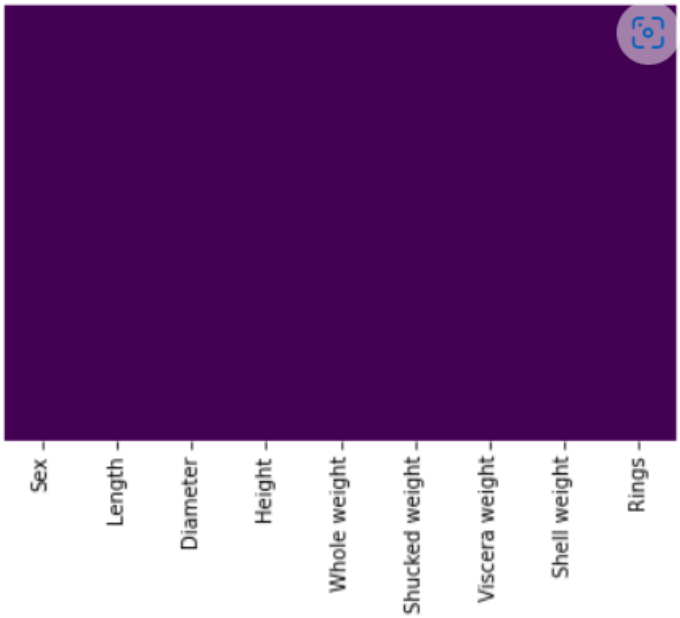
Question-5:

Handle the Missing values.

Solution:

```
sns.heatmap(df.isnull(),yticklabels=False,cbar=False,cmap="viridis")
#there is no missing values as per heatmap
```

```
<AxesSubplot:>
```



Question-6:

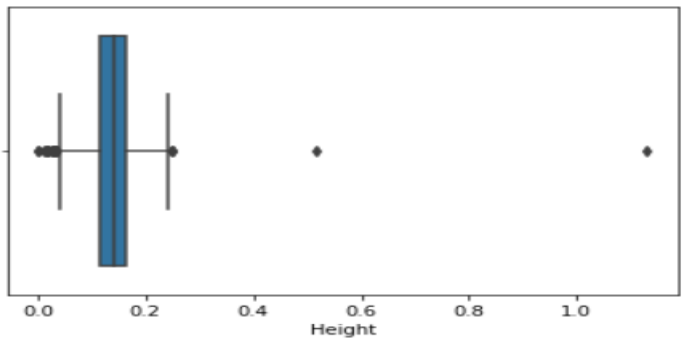
Find the outliers and replace the outliers

Solution:

```
sns.boxplot(df['Height'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_d
rg: x. From version 0.12, the only valid positional a
yword will result in an error or misinterpretation.
  warnings.warn(
```

```
<AxesSubplot:xlabel='Height'>
```




```
Q1=df.Height.quantile(0.25)
Q2=df.Height.quantile(0.75)
IQR=Q2-Q1
print(IQR)
```

```
0.05
```

```
df=df[~((df.Height<(Q1-1.5*IQR))|(df.Height>(Q2+1.5*IQR)))]
df
```

	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

4148 rows × 9 columns

Question-7:

Check for Categorical columns and perform encoding.

Solution:

```
df['Sex'].replace({'F':1,'M':0},inplace=True)
df.head()
```

```
C:\Users\Manjunathan V\AppData\Local\Temp\ipykernel_16952\199196216.py:1: SettingWithCop
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
rsus-a-copy
  df['Sex'].replace({'F':1,'M':0},inplace=True)
```

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	0	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150	15
1	0	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	7
2	1	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	9
3	0	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:

```
d=pd.get_dummies(df,columns=['Height'])
d
```

	Sex	Length	Diameter	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings	Height_0.04	Height_0.045	...	Height_0.195	Height_0.2	Height_0.205	Height_0.21	Heig
0	0	0.455	0.365	0.5140	0.2245	0.1010	0.1500	15	0	0	...	0	0	0	0	
1	0	0.350	0.265	0.2255	0.0995	0.0485	0.0700	7	0	0	...	0	0	0	0	
2	1	0.530	0.420	0.6770	0.2565	0.1415	0.2100	9	0	0	...	0	0	0	0	
3	0	0.440	0.365	0.5160	0.2155	0.1140	0.1550	10	0	0	...	0	0	0	0	
4	I	0.330	0.255	0.2050	0.0895	0.0395	0.0550	7	0	0	...	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
4172	1	0.565	0.450	0.8870	0.3700	0.2390	0.2490	11	0	0	...	0	0	0	0	
4173	0	0.590	0.440	0.9660	0.4390	0.2145	0.2605	10	0	0	...	0	0	0	0	
4174	0	0.600	0.475	1.1760	0.5255	0.2875	0.3080	9	0	0	...	0	0	1	0	
4175	1	0.625	0.485	1.0945	0.5310	0.2610	0.2960	10	0	0	...	0	0	0	0	
4176	0	0.710	0.555	1.9485	0.9455	0.3765	0.4950	12	0	0	...	1	0	0	0	

4148 rows × 49 columns

```
y1=d['Diameter']
y1
```

```
0       0.365
1       0.265
2       0.420
3       0.365
4       0.255
        ...
4172    0.450
4173    0.440
4174    0.475
4175    0.485
4176    0.555
Name: Diameter, Length: 4148, dtype: float64
```

```
x1=d.drop(columns='Diameter',axis=1)
x1.head()
```

	Sex	Length	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings	Height_0.04	Height_0.045	Height_0.05	...	Height_0.195	Height_0.2	Height_0.205	Height_0.21	Heig
0	0	0.455	0.5140	0.2245	0.1010	0.150	15	0	0	0	...	0	0	0	0	
1	0	0.350	0.2255	0.0995	0.0485	0.070	7	0	0	0	...	0	0	0	0	
2	1	0.530	0.6770	0.2565	0.1415	0.210	9	0	0	0	...	0	0	0	0	
3	0	0.440	0.5160	0.2155	0.1140	0.155	10	0	0	0	...	0	0	0	0	
4	I	0.330	0.2050	0.0895	0.0395	0.055	7	0	0	0	...	0	0	0	0	

5 rows × 48 columns

Question-9:

Scale the independent variables

Solution:

```
x1=df.iloc[:,6:7].values
from sklearn.preprocessing import StandardScaler
std=StandardScaler()
x1=std.fit_transform(x1)
x1
```

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

Question-10:

Split the data into training and testing

Solution:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x1,y1,test_size=0.2,random_state=0)
```

```
x_train
```

```
array([[ 1.23090898],
       [-1.33195091],
       [-0.03441682],
       ...,
       [ 0.76158814],
       [-1.23072484],
       [-0.48533292]])
```

```
x_test
```

```
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],
       [-1.67851174e-01],
       [ 2.14047157e-01],
       [-1.11109404e+00],
       [ 1.72636495e-01],
       [ 1.19870068e+00],
       [-1.50679592e+00],
       [ 3.65886252e-01],
```

```
y_train
```

```
3780    0.480
3161    0.220
3919    0.430
625     0.405
2388    0.375
        ...
1042    0.565
3289    0.420
1667    0.505
2630    0.285
2756    0.400
Name: Diameter, Length: 3318, dtype: float64
```

```
y_test
```

```
834     0.325
4106    0.590
980     0.445
1513    0.485
3201    0.300
        ...
1963    0.520
693     0.290
322     0.310
1422    0.575
803     0.275
Name: Diameter, Length: 830, dtype: float64
```

Question-11:

Build the Model

Solution:

```
from sklearn.linear_model import LinearRegression
r=LinearRegression()
r.fit(x_train,y_train)
```

```
LinearRegression()
```

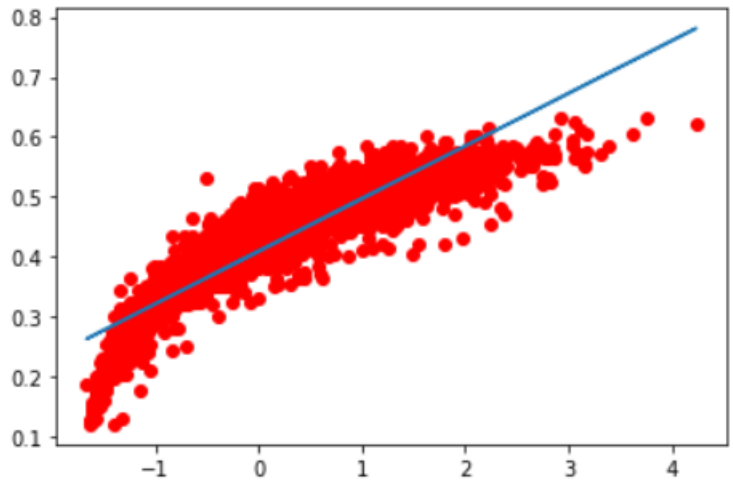
Question-12:

Train the Model

Solution:

```
import matplotlib.pyplot as plt
plt.scatter(x_train,y_train,color="red")
plt.plot(x_train,r.predict(x_train))
```

```
[<matplotlib.lines.Line2D at 0x23b0a2b4f70>]
```



Question-13:

Test the Model

Solution

```
y_p=r.predict(x_test)
y_p
```

```
array([0.34316784, 0.60401919, 0.40625747, 0.45923658, 0.30838766,
       0.42971387, 0.41960335, 0.3443811 , 0.39655137, 0.33912364,
       0.44872164, 0.46813384, 0.39412485, 0.42769177, 0.31121861,
       0.42405198, 0.5142378 , 0.27643843, 0.44103765, 0.31324071,
       0.46247195, 0.30838766, 0.33588827, 0.54578261, 0.48552393,
       0.61372529, 0.27724727, 0.41636799, 0.4264785 , 0.3403369 ,
       0.36096235, 0.33507943, 0.38886738, 0.41879451, 0.56034175,
       0.45842774, 0.3326529 , 0.51383337, 0.50008307, 0.46975152,
       0.3892718 , 0.29706388, 0.525966  , 0.29868157, 0.34518995,
       0.45357469, 0.50210517, 0.47703109, 0.45276585, 0.63839495,
       0.38199223, 0.62424022, 0.59350425, 0.47986204, 0.35489604,
       0.3775436 , 0.27441632, 0.3678375 , 0.47177362, 0.3071744 ,
       0.41030168, 0.34761647, 0.41555915, 0.30313019, 0.35934467,
       0.3168805 , 0.53688535, 0.43982439, 0.50978917, 0.3189026 ,
       0.31162303, 0.50170075, 0.42850061, 0.39857348, 0.46287637,
       0.34357226, 0.56074617, 0.3658154 , 0.28048263, 0.44103765,
       0.40504421, 0.32052028, 0.4713692 , 0.50736264, 0.42000777,
       0.27643843, 0.32173355, 0.47824435, 0.53445883, 0.34802089,
       0.43011829, 0.31243187, 0.40140442, 0.42081662, 0.3403369 ,
```

Question-14:

Measure the performance using Metrics.

```
from sklearn.metrics import r2_score
acc=r2_score(y_test,y_p)
acc
```

```
0.8060461426790042
```