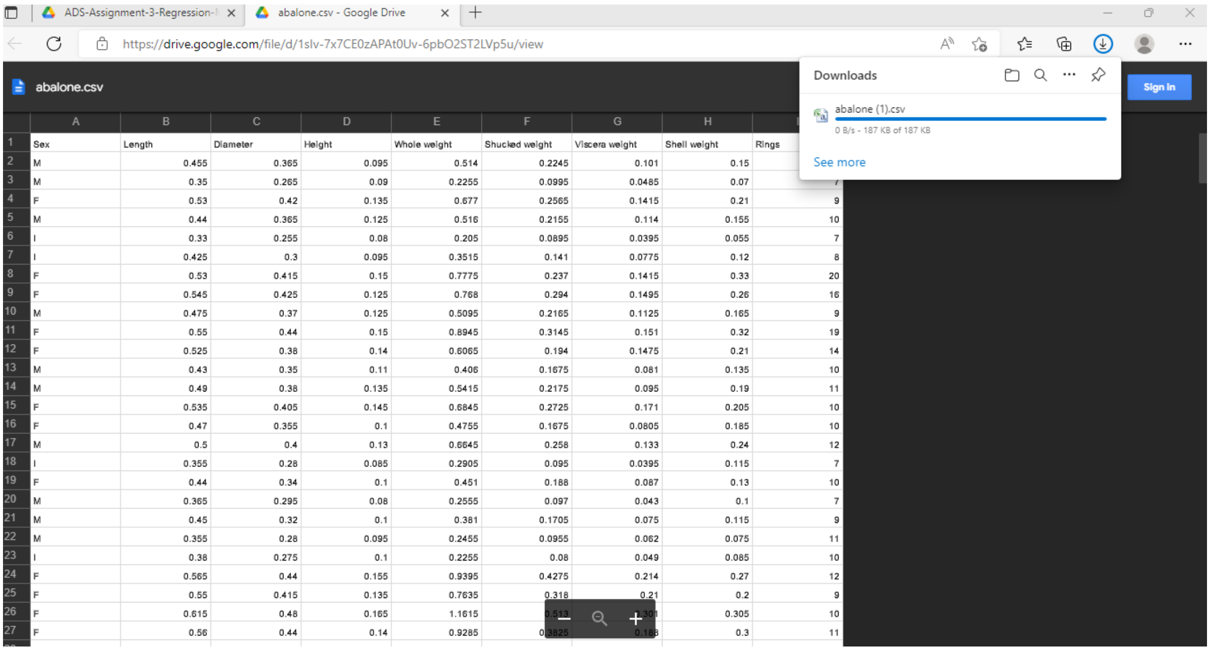
# Assignment -3
# Regression Model

| | |
|---|---|
| Assignment Date | 29 September 2022 |
| Student Name | Logeshkumar R |
| Student Roll Number | 727719EUCS074 |
| 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 | [illegible] |
| 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 | 0.56 | 0.44 | 0.14 | 0.9285 | 0.3825 | 0.188 | 0.3 | 11 |

**Question-2:**

Load the dataset.

**Solution:**

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

d = pd.read_csv("E://abalone (1).csv")

d.head()
```

```
In [2]: import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import seaborn as sns
        d = pd.read_csv("E://abalone (1).csv")
        d.head()
```

Out[2]:

| | 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:**

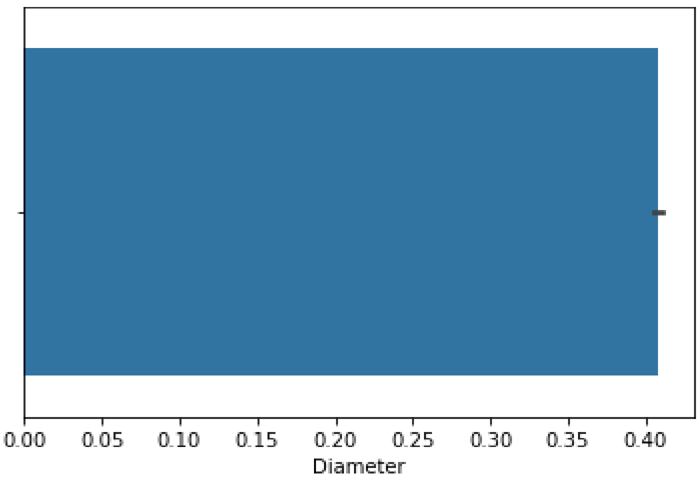
Perform Below Visualizations.

· Univariate Analysis

**Solution:**

```
In [5]: sns.barplot(d.Diameter)
```

Out[5]: <AxesSubplot:xlabel='Diameter'>

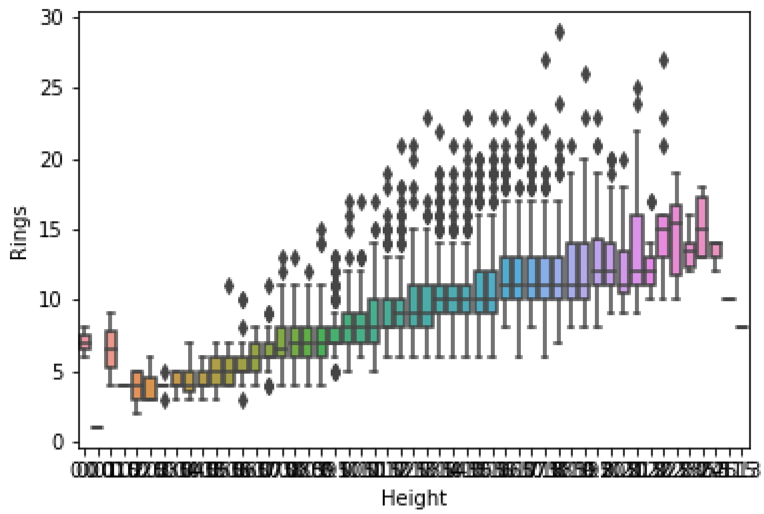


· Bi-Variate Analysis

**Solution:**

```
In [9]: sns.boxplot(y=d.Rings,x=d.Height)
```

```
Out[9]: <AxesSubplot:xlabel='Height', ylabel='Rings'>
```

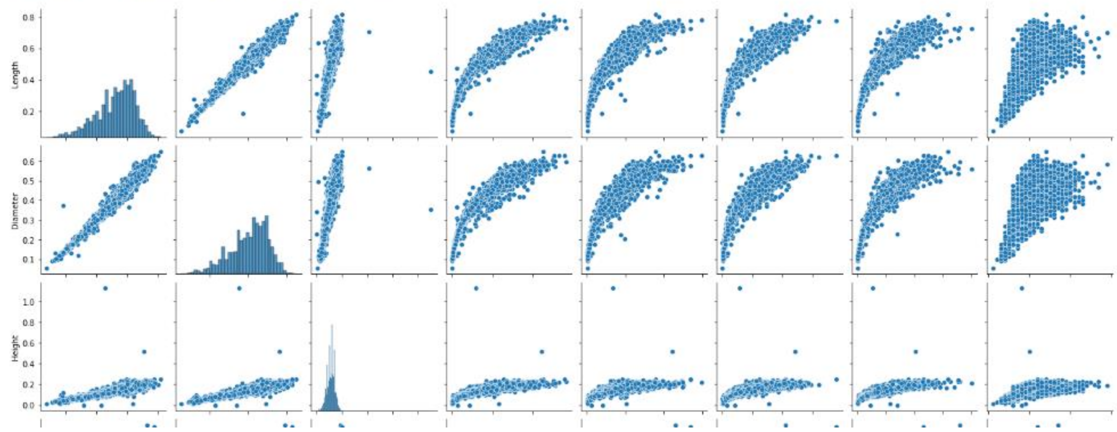



· Multi-Variate Analysis

**Solution:**

```
In [8]: sns.pairplot(d)
```

```
Out[8]: <seaborn.axisgrid.PairGrid at 0x2cb4763d400>
```



**Question-4:**

Perform descriptive statistics on the dataset.

**Solution:**

In [10]:
```
d['Diameter'].mean()
```

Out[10]:
```
0.407881254488869
```

In [11]:
```
d['Height'].median()
```

Out[11]:
```
0.14
```

In [13]:
```
d['Rings'].mode()
```

Out[13]:
```
0    9
dtype: int64
```

In [14]:
```
d.skew()
```

Out[14]:
```
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 [15]:
```
d.kurt()
```

Out[15]:
```
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
```

**Question-5:**

Check for Missing values and deal with them.

**Solution:**

```
In [16]: d.isna().any()

Out[16]: 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 [18]: d['Rings'].fillna(d['Rings'].mean(),inplace=True)
         d
```

Out[18]:

| | 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 |

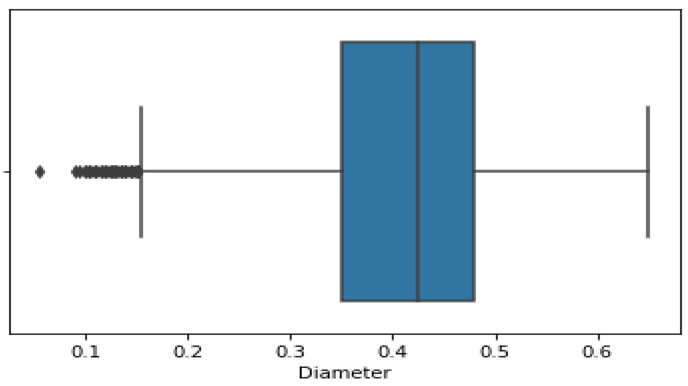
**Question-6:**

Find the outliers and replace the outliers

**Solution:**

```
In [19]: sns.boxplot(d['Diameter'])

Out[19]: <AxesSubplot:xlabel='Diameter'>
```




```
In [20]: Q1=d.Diameter.quantile(0.25)
         Q2=d.Diameter.quantile(0.75)
         IQR=Q2-Q1
         print(IQR)

0.13
```

```
In [21]: d=d[~((d.Diameter<(Q1-1.5*IQR))|(d.Diameter>(Q2+1.5*IQR)))]
         d
```

Out[21]:

| | 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 |

4118 rows × 9 columns

**Question-7:**

Check for Categorical columns and perform encoding.

**Solution:**

```
In [22]: d['Sex'].replace({'M':1,'F':0},inplace=True)
         d.head()
```

Out[22]:

| | Sex | Length | Diameter | Height | Whole weight | Shucked weight | Viscera weight | Shell weight | Rings |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 0.455 | 0.365 | 0.095 | 0.5140 | 0.2245 | 0.1010 | 0.150 | 15 |
| **1** | 1 | 0.350 | 0.265 | 0.090 | 0.2255 | 0.0995 | 0.0485 | 0.070 | 7 |
| **2** | 0 | 0.530 | 0.420 | 0.135 | 0.6770 | 0.2565 | 0.1415 | 0.210 | 9 |
| **3** | 1 | 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 [23]:
```python
dm= pd.get_dummies(d,columns=['Height'])
dm
```

Out[23]:

| | Sex | Length | Diameter | Whole weight | Shucked weight | Viscera weight | Shell weight | Rings | Height_0.0 | Height_0.015 | ... | Height_0.21 | Height_0.215 | Height_0.22 | Height_0.225 | Heig |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 0.455 | 0.365 | 0.5140 | 0.2245 | 0.1010 | 0.1500 | 15 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |
| **1** | 1 | 0.350 | 0.265 | 0.2255 | 0.0995 | 0.0485 | 0.0700 | 7 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |
| **2** | 0 | 0.530 | 0.420 | 0.6770 | 0.2565 | 0.1415 | 0.2100 | 9 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |
| **3** | 1 | 0.440 | 0.365 | 0.5160 | 0.2155 | 0.1140 | 0.1550 | 10 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |
| **4** | 1 | 0.330 | 0.255 | 0.2050 | 0.0895 | 0.0395 | 0.0550 | 7 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| **4172** | 0 | 0.565 | 0.450 | 0.8870 | 0.3700 | 0.2390 | 0.2490 | 11 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |
| **4173** | 1 | 0.590 | 0.440 | 0.9660 | 0.4390 | 0.2145 | 0.2605 | 10 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |
| **4174** | 1 | 0.600 | 0.475 | 1.1760 | 0.5255 | 0.2875 | 0.3080 | 9 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |
| **4175** | 0 | 0.625 | 0.485 | 1.0945 | 0.5310 | 0.2610 | 0.2960 | 10 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |
| **4176** | 1 | 0.710 | 0.555 | 1.9485 | 0.9455 | 0.3765 | 0.4950 | 12 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |

4118 rows × 54 columns

In [24]:
```python
y = dm['Diameter']
y
```

Out[24]:
```
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: 4118, dtype: float64
```

In [25]:
```python
x = dm.drop(columns='Diameter',axis=1)
x.head()
```

Out[25]:

| | Sex | Length | Whole weight | Shucked weight | Viscera weight | Shell weight | Rings | Height_0.0 | Height_0.015 | Height_0.04 | ... | Height_0.21 | Height_0.215 | Height_0.22 | Height_0.225 | Hei |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 0.455 | 0.5140 | 0.2245 | 0.1010 | 0.150 | 15 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |
| **1** | 1 | 0.350 | 0.2255 | 0.0995 | 0.0485 | 0.070 | 7 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |
| **2** | 0 | 0.530 | 0.6770 | 0.2565 | 0.1415 | 0.210 | 9 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |
| **3** | 1 | 0.440 | 0.5160 | 0.2155 | 0.1140 | 0.155 | 10 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |
| **4** | 1 | 0.330 | 0.2050 | 0.0895 | 0.0395 | 0.055 | 7 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |

5 rows × 53 columns

**Question-9:**

Scale the independent variables

**Solution:**

```
In [26]: x=d.iloc[:,6:7].values
         from sklearn.preprocessing import StandardScaler
         std=StandardScaler()
         x=std.fit_transform(x)
         x

Out[26]: array([[-0.75723081],
                [-1.24159927],
                [-0.38357514],
                ...,
                [ 0.96343047],
                [ 0.71893972],
                [ 1.78455033]])
```

**Question-10:**

Split the data into training and testing

**Solution:**

```
In [29]: 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 [30]: x_train

Out[30]: array([[ 0.49751414],
                [-0.68803532],
                [-1.2600514 ],
                ...,
                [ 0.55748357],
                [-0.68803532],
                [-0.4481576 ]])

In [31]: x_test
```

```
                [ 1.87681099e+00],
                [ 8.29652515e-01],
                [-1.37076419e+00],
                [-1.23237320e+00],
                [-2.40571123e-01],
                [-6.05000724e-01],
                [-1.03862582e+00],
                [-2.12892926e-01],
                [ 2.02280037e-01],
                [ 1.24482548e+00],
                [-3.69736045e-01],
                [-4.02027276e-01],
                [-4.25092440e-01],
```

```
In [32]: y_train
```

```
Out[32]: 780     0.410
         2411    0.395
         1553    0.290
         2627    0.205
         962     0.390
                 ...
         1060    0.195
         3312    0.410
         1681    0.540
         2650    0.395
         2777    0.420
         Name: Diameter, Length: 3294, dtype: float64
```

```
In [33]: y_test
```

```
Out[33]: 2004    0.275
         615     0.345
         2888    0.400
         2599    0.480
         464     0.195
                 ...
         2102    0.310
         410     0.500
         3138    0.400
         2520    0.425
         3358    0.215
         Name: Diameter, Length: 824, dtype: float64
```

**Question-11:**

Build the Model

**Solution:**

```
In [34]: from sklearn.linear_model import LinearRegression
         regressor=LinearRegression()
         regressor.fit(x_train,y_train)
```

```
Out[34]: LinearRegression()
```

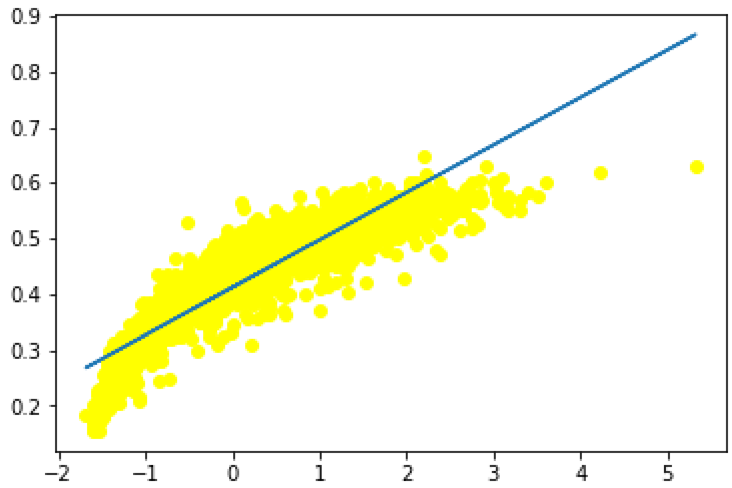
**Question-12:**

Train the Model

**Solution:**

```
In [35]: plt.scatter(x_train,y_train,color='yellow')
         plt.plot(x_train,regressor.predict(x_train))

Out[35]: [<matplotlib.lines.Line2D at 0x2cb4c5fadf0>]
```




**Question-13:**

Test the Model

**Solution:**

```
In [36]: y_pred=regressor.predict(x_test)
         y_pred

Out[36]: array([0.30739667, 0.3231634 , 0.37046358, 0.48398401, 0.28374658,
                0.5336492 , 0.44101968, 0.51867081, 0.42643546, 0.56557683,
                0.41027456, 0.43313632, 0.45087389, 0.4272238 , 0.3491785 ,
                0.41421625, 0.43274215, 0.46900562, 0.39884369, 0.38544197,
                0.42682963, 0.5671535 , 0.51985332, 0.35509102, 0.36415689,
                0.4457497 , 0.44377886, 0.45599807, 0.64835214, 0.40475621,
                0.37795277, 0.43116548, 0.43392466, 0.39332533, 0.481619  ,
                0.4619106 , 0.4445672 , 0.32355757, 0.29202411, 0.49738573,
                0.51157579, 0.35469685, 0.38150029, 0.33774762, 0.37125191,
                0.30936751, 0.34563098, 0.36967524, 0.34326597, 0.48634902,
                0.60578198, 0.45718058, 0.5584818 , 0.60420531, 0.44417303,
                0.28216991, 0.37204025, 0.48950237, 0.4556039 , 0.3144917 ,
                0.52300666, 0.29596579, 0.39214283, 0.28492908, 0.42958881,
                0.39293116, 0.47531231, 0.46072809, 0.48437818, 0.35982104,
                0.3677044 , 0.43037714, 0.51551747, 0.53483171, 0.39411367,
                0.28650576, 0.53877339, 0.54823343, 0.45639224, 0.4457497 ,
                0.41500458, 0.39569034, 0.40317954, 0.63495043, 0.5076341 ,
```

**Question-14:**

Measure the performance using Metrics.

**Solution:**

```
In [37]: from sklearn.metrics import r2_score
         accuracy=r2_score(y_test,y_pred)
         accuracy

Out[37]: 0.8137980390529289

In [38]: from sklearn import metrics
         np.sqrt(metrics.mean_squared_error(y_test,y_pred))

Out[38]: 0.040880060610600996
```