Deena 20104016

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as pp
```

Problem Statement

LINEAR REGRESSION

In [2]:
```
a = pd.read_csv("world.csv")
```

Out[2]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 6 columns

HEAD

In [3]:

Out[3]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

Data Cleaning and Preprocessing

In [4]:

Out[4]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

In [5]:

Out[5]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	75.500000	5.843333	3.054000	3.758667	1.198667
std	43.445368	0.828066	0.433594	1.764420	0.763161
min	1.000000	4.300000	2.000000	1.000000	0.100000
25%	38.250000	5.100000	2.800000	1.600000	0.300000
50%	75.500000	5.800000	3.000000	4.350000	1.300000
75%	112.750000	6.400000	3.300000	5.100000	1.800000
max	150.000000	7.900000	4.400000	6.900000	2.500000

To display heading

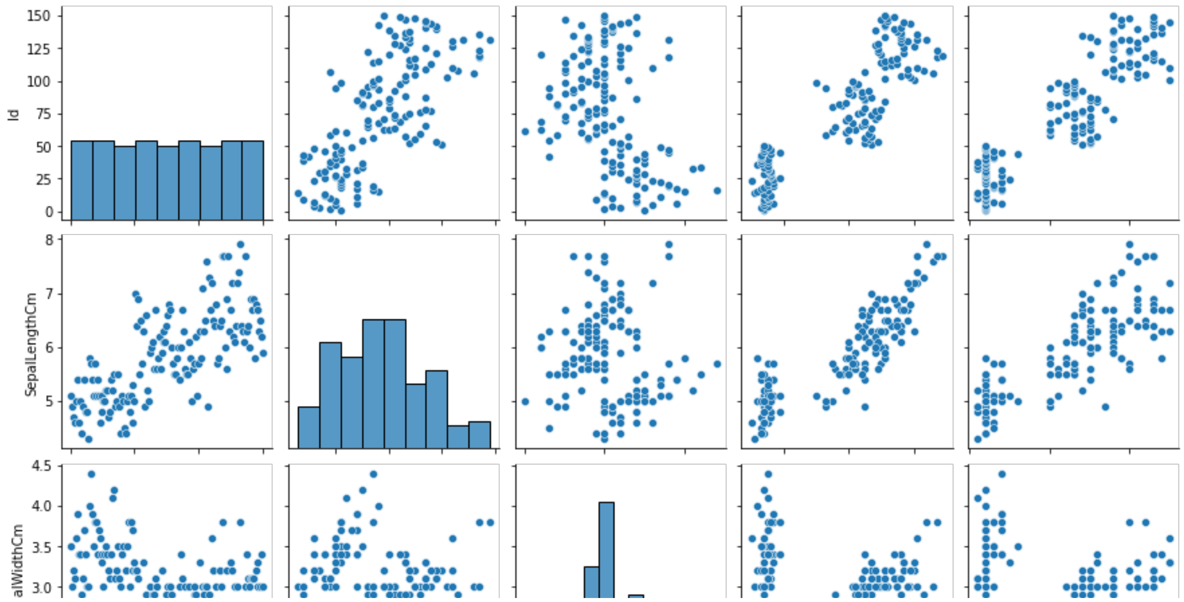
In [6]:

Out[6]:

```
Index(['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm
',
       'Species'],
      dtype='object')
```

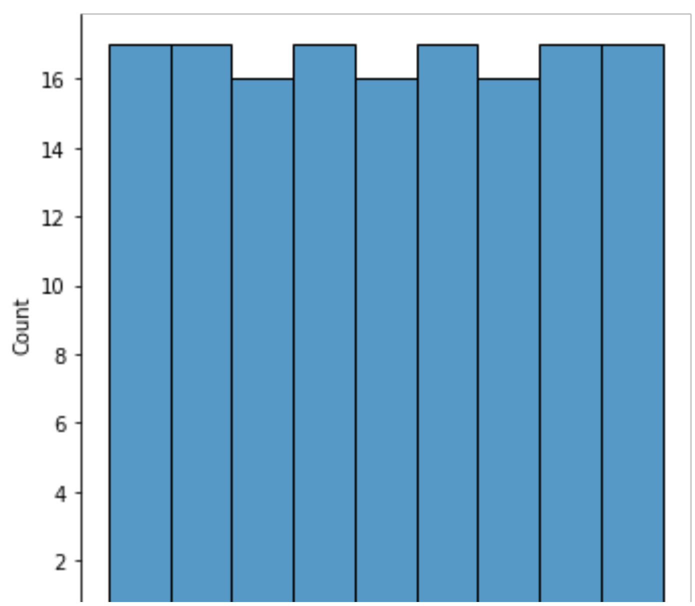
In [7]:

Out[7]: <seaborn.axisgrid.PairGrid at 0x1fe3c6d2dc0>



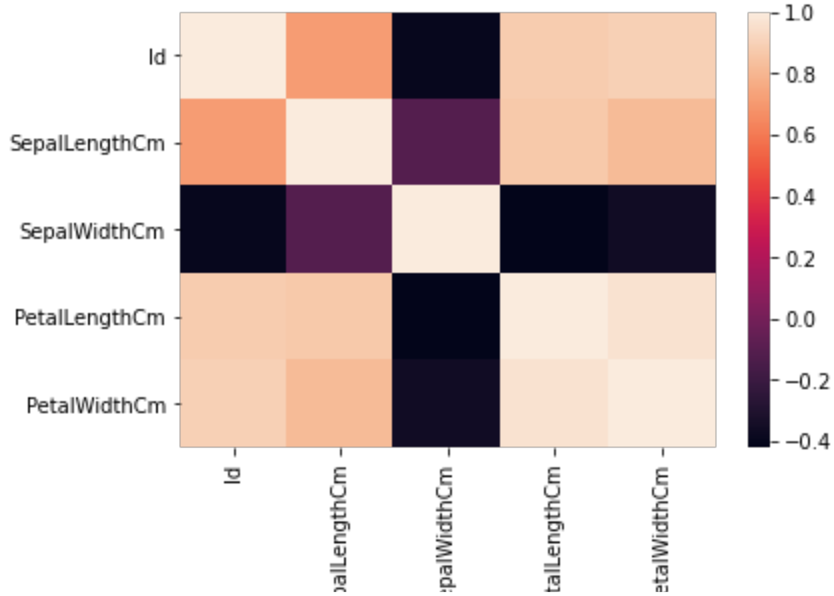
In [8]:

Out[8]: <seaborn.axisgrid.FacetGrid at 0x1fe3ea92e80>



In [9]:

Out[9]: <AxesSubplot:>



TO TRAIN THE MODEL - MODEL BUILDING

In [10]:
```
x = a[['Id']]
```

In [11]:
```
# to split my dataset into training and test data
from sklearn.model_selection import train_test_split
```

In [12]:
```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
```

Out[12]: LinearRegression()

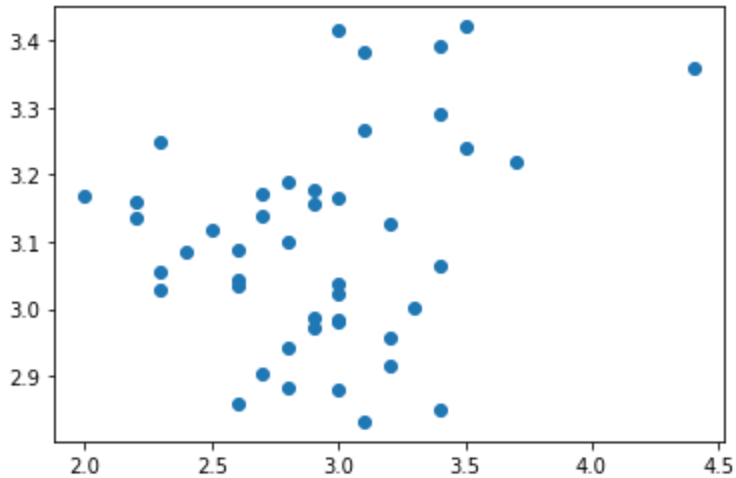
In [13]:
```
coeff = pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
```

Out[13]:

	Co-efficient
Id	-0.004195

In [14]:
```
prediction= lr.predict(x_test)
```

Out[14]: <matplotlib.collections.PathCollection at 0x1fe3f6c8dc0>



In [15]:

Out[15]: -0.11366093168042446

RIDGE & LASSO

In [16]:
```
from sklearn.linear_model import Ridge,Lasso
rr=Ridge(alpha=10)
```

Out[16]: Ridge(alpha=10)

In [17]:

Out[17]: -0.11365854539518905

In [18]:
```
la=Lasso(alpha=10)
```

Out[18]: Lasso(alpha=10)

In [19]:

Out[19]: -0.18298051577488494

In [20]:
```
from sklearn.linear_model import ElasticNet
a=ElasticNet()
```

Out[20]: ElasticNet()

In [21]:
```
print(a.coef_)
print(a.intercept_)
print(a.score(x_test,y_test))
```

```
[-0.00395455]
3.406400682297664
-0.11106227161974092
[3.03467255 3.13353641 2.995127   2.8646267  3.00699067 2.97930879
 2.84880848 3.15330919 3.13749097 3.12562731 3.04258166 3.34312781
 2.92789958 3.10189998 3.1177182  3.15726374 3.24030939 3.0386271
 3.04653621 3.18494562 2.99117245 2.87253581 3.02676344 3.36685514
 3.16517285 2.89230858 3.23240028 3.21262751 2.89626314 2.96744512
 3.05839988 3.39849157 3.16912741 3.25612761 3.06630899 3.1612183
 2.91603591 3.37476425 2.9516269  3.17308196 3.40244613 3.08608176
 3.09003631 3.27985494 2.98721789]
```

In [22]:
```
from sklearn import metrics
print(" Mean Absolute Error :",metrics.mean_absolute_error(y_test,prediction))
print(" Mean Squared Error :",metrics.mean_squared_error(y_test,prediction))
```

```
 Mean Absolute Error : 0.36502749673716817
 Mean Squared Error : 0.22302916258453298
 Root Mean Absolute Error : 0.6041750547127613
```