

```
In [1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
In [2]: df=pd.read_csv("14_Iris.csv")
df
```

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

```
In [3]: df.head()
```

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 DATA PREPROCESSING

```
In [4]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
```

#	Column	Non-Null Count	Dtype
0	Id	150 non-null	int64
1	SepalLengthCm	150 non-null	float64
2	SepalWidthCm	150 non-null	float64
3	PetalLengthCm	150 non-null	float64
4	PetalWidthCm	150 non-null	float64
5	Species	150 non-null	object

dtypes: float64(4), int64(1), object(1)  
memory usage: 7.2+ KB

In [5]:

```
df.describe()
```

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

In [6]:

```
df.columns
```

Out[6]:

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

In [7]:

```
df1=df.dropna(axis=1)  
df1
```

Out[7]:

	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

	<b>Id</b>	<b>SepalLengthCm</b>	<b>SepalWidthCm</b>	<b>PetalLengthCm</b>	<b>PetalWidthCm</b>	<b>Species</b>
<b>149</b>	150	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 6 columns

```
In [8]: df1.columns
```

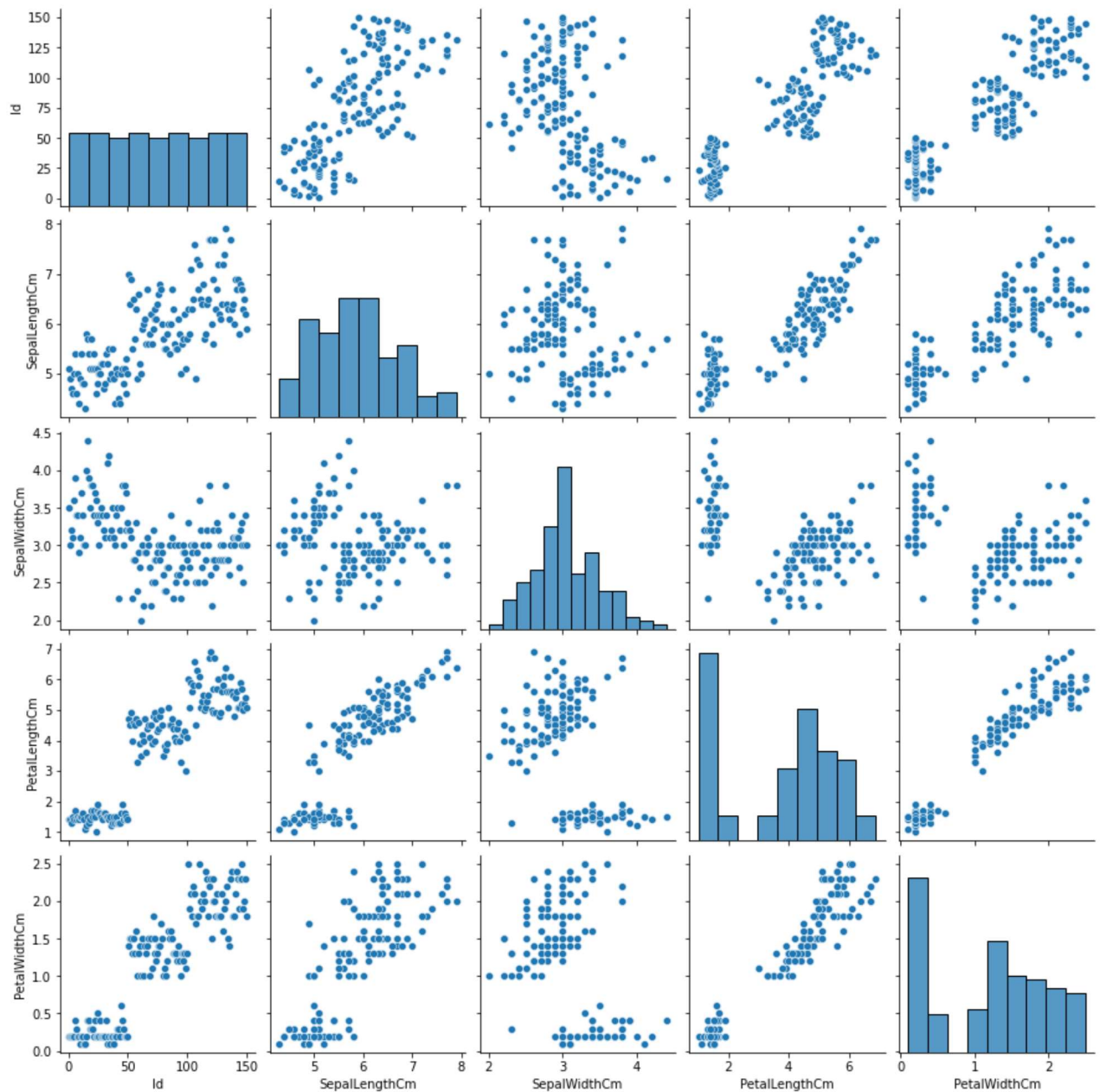
```
Out[8]: Index(['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm',  
              'Species'],  
             dtype='object')
```

```
In [9]: df1=df1[[ 'Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']]
```

## EDA AND VISUALIZATION

```
In [10]: sns.pairplot(df1)
```

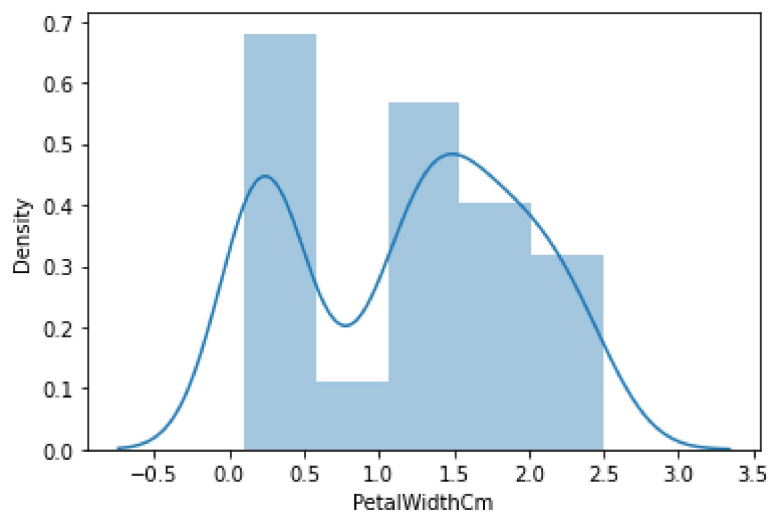
```
Out[10]: <seaborn.axisgrid.PairGrid at 0x282ad738970>
```



```
In [11]: sns.distplot(df1['PetalWidthCm'])
```

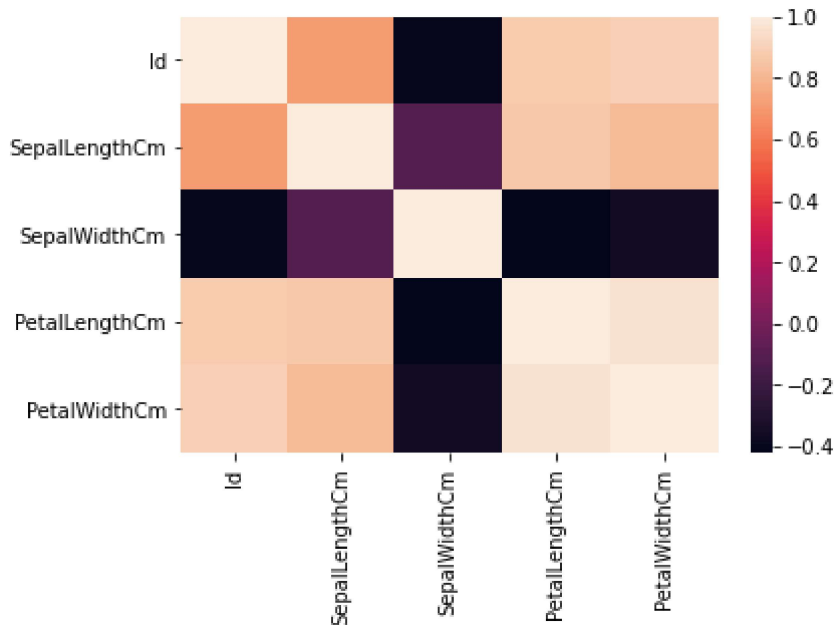
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).  
warnings.warn(msg, FutureWarning)

```
Out[11]: <AxesSubplot:xlabel='PetalWidthCm', ylabel='Density'>
```



```
In [12]: sns.heatmap(df1.corr())
```

```
Out[12]: <AxesSubplot:>
```



## TO TRAIN THE MODEL AND MODEL BULDING

```
In [13]: x=df[['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm']]
          y=df['PetalWidthCm']
```

```
In [14]: 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 [15]: from sklearn.linear_model import LinearRegression
          lr=LinearRegression()
          lr.fit(x_train,y_train)
```

Out[15]: LinearRegression()

In [16]: `lr.intercept_`

Out[16]: -0.3860374773412274

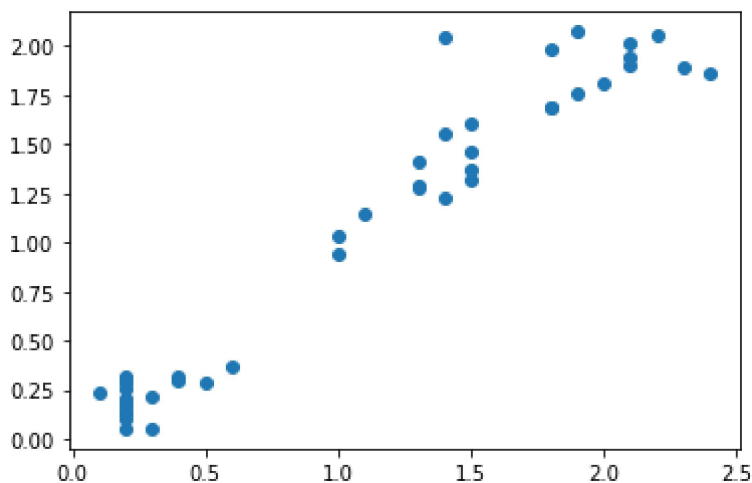
In [17]: `coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])`  
`coeff`

Out[17]:

	Co-efficient
<b>Id</b>	0.003161
<b>SepalLengthCm</b>	-0.173415
<b>SepalWidthCm</b>	0.221213
<b>PetalLengthCm</b>	0.444044

In [18]: `prediction =lr.predict(x_test)`  
`plt.scatter(y_test,prediction)`

Out[18]: <matplotlib.collections.PathCollection at 0x282b0725bb0>



## ACCURACY

In [19]: `lr.score(x_test,y_test)`

Out[19]: 0.94135730799608

In [20]: `lr.score(x_train,y_train)`

Out[20]: 0.9462165491754232

In [21]: `from sklearn.linear_model import Ridge,Lasso`

```
In [22]: rr=Ridge(alpha=10)
         rr.fit(x_train,y_train)
```

```
Out[22]: Ridge(alpha=10)
```

```
In [23]: rr.score(x_test,y_test)
```

```
Out[23]: 0.9378683549904536
```

```
In [24]: rr.score(x_train,y_train)
```

```
Out[24]: 0.9374124290116804
```

```
In [25]: la=Lasso(alpha=10)
         la.fit(x_train,y_train)
```

```
Out[25]: Lasso(alpha=10)
```

```
In [26]: la.score(x_test,y_test)
```

```
Out[26]: 0.7306922927169373
```

```
In [27]: la.score(x_train,y_train)
```

```
Out[27]: 0.6844683838890258
```