

Data collection

Importing libraries

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

Importing dataset

```
In [2]: data=pd.read_csv(r"C:\Users\user\Downloads\iris.csv")
data
```

```
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]: # to display first 8 dataset values
da=data.head(8)
da
```

```
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

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
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
5	6	5.4	3.9	1.7	0.4	Iris-setosa
6	7	4.6	3.4	1.4	0.3	Iris-setosa
7	8	5.0	3.4	1.5	0.2	Iris-setosa

info

In [4]:

```
# to identify missing values
data.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
```

describe

In [5]:

```
# to display summary of the dataset
data.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

columns

```
In [6]: # to display headings of the dataset
data.columns
```

```
Out[6]: Index(['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm',
              'Species'],
              dtype='object')
```

```
In [7]: a=data.dropna(axis=1)
a
```

```
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
149	150	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 6 columns

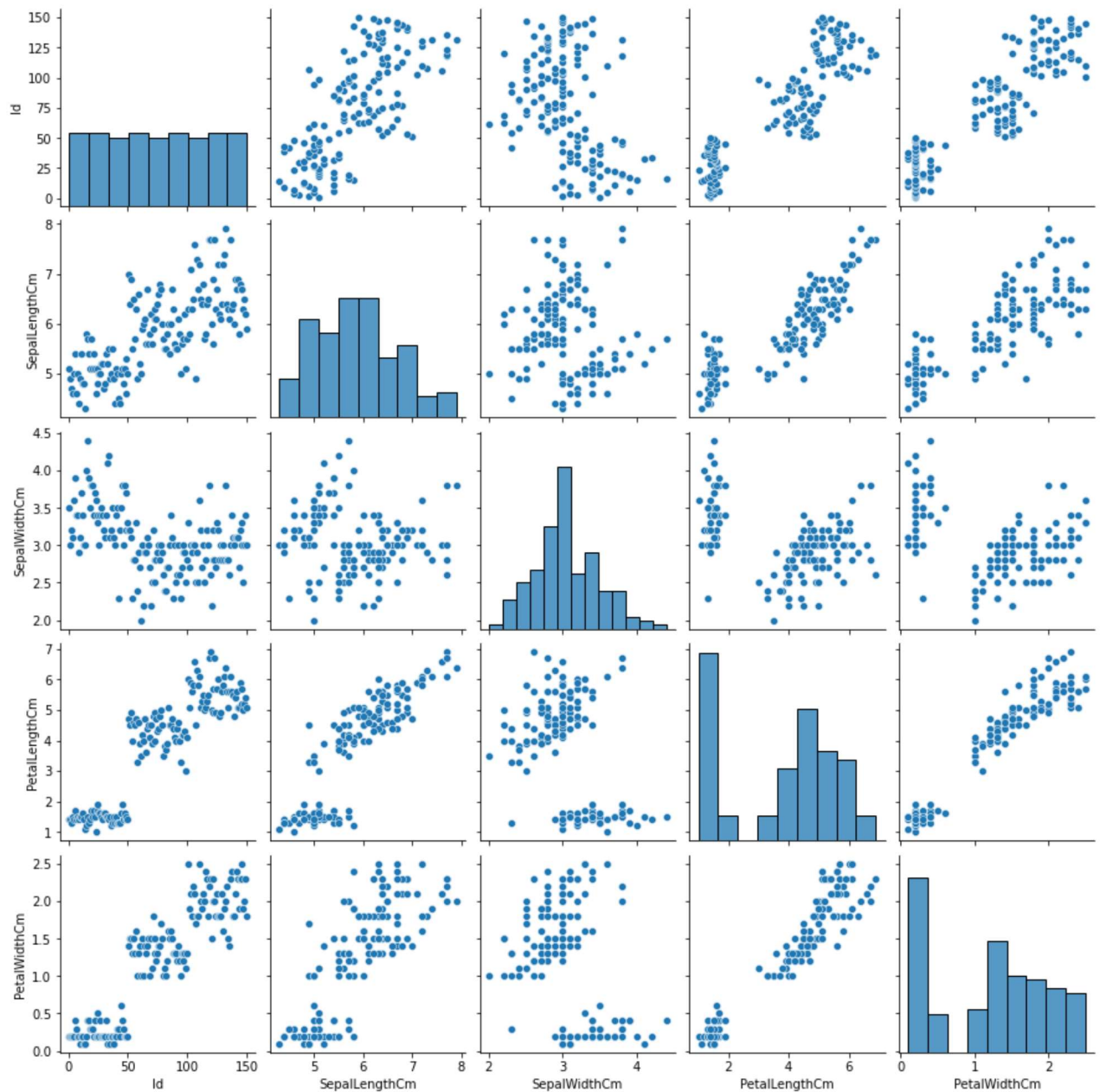
```
In [8]: a.columns
```

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

EDA and Visualization

```
In [9]: sns.pairplot(a)
```

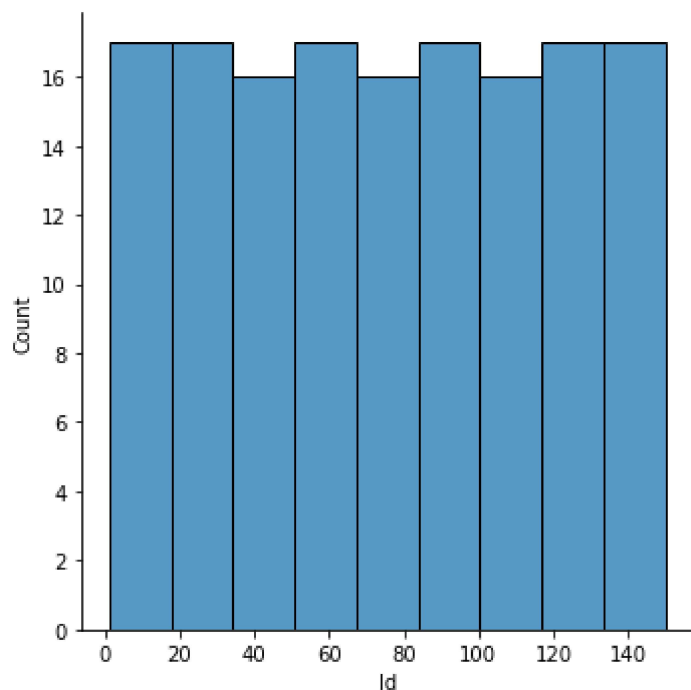
```
Out[9]: <seaborn.axisgrid.PairGrid at 0x284a1e20940>
```



distribution plot

```
In [11]: sns.displot(a["Id"])
```

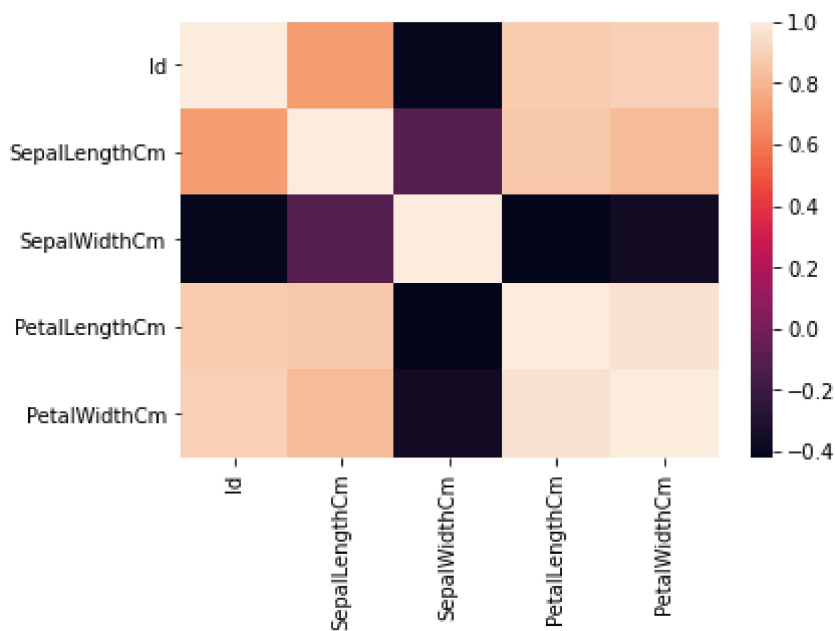
```
Out[11]: <seaborn.axisgrid.FacetGrid at 0x284a1e208b0>
```



correlation

```
In [12]: dat=data[['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm',
                  'Species']]
          sns.heatmap(dat.corr())
```

Out[12]: <AxesSubplot:>



To train the model-Model Building

```
In [13]: x=a[['Id']]
          y=a[['Id']]
```

```
In [14]: # to split my dataset into training and test data
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]: print(lr.intercept_)
```

1.4210854715202004e-14

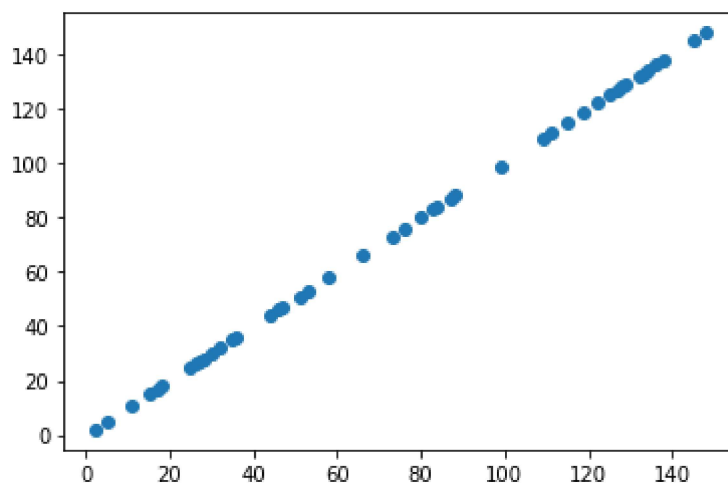
```
In [17]: coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[17]:

	Co-efficient
Id	1.0

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

Out[18]: <matplotlib.collections.PathCollection at 0x284a5050f40>



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

1.0

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

Out[20]: 1.0

Ridge regression

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

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

```
Out[22]: 0.9999999971732861
```

```
In [23]: rr.score(x_train,y_train)
```

```
Out[23]: 0.999999997173314
```

Lasso regression

```
In [24]: la=Lasso(alpha=10)
la.fit(x_train,y_train)
la.score(x_train,y_train)
```

```
Out[24]: 0.9999688324734263
```

```
In [25]: la.score(x_test,y_test)
```

```
Out[25]: 0.9999688321661742
```

```
In [ ]:
```