ostt4j6er

July 28, 2023

[11]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[12]:
```
df=pd.read_csv("/content/14_Iris.csv")
df
```

[12]:
```
      Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  \
0      1            5.1           3.5            1.4           0.2
1      2            4.9           3.0            1.4           0.2
2      3            4.7           3.2            1.3           0.2
3      4            4.6           3.1            1.5           0.2
4      5            5.0           3.6            1.4           0.2
..   …              …             …              …             …
145  146            6.7           3.0            5.2           2.3
146  147            6.3           2.5            5.0           1.9
147  148            6.5           3.0            5.2           2.0
148  149            6.2           3.4            5.4           2.3
149  150            5.9           3.0            5.1           1.8

            Species
0       Iris-setosa
1       Iris-setosa
2       Iris-setosa
3       Iris-setosa
4       Iris-setosa
..              …
145  Iris-virginica
146  Iris-virginica
147  Iris-virginica
148  Iris-virginica
149  Iris-virginica

[150 rows x 6 columns]
```

[13]:
```
df.head()
```

```
[13]:    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
```

1 DATA CLEANING AND DATA PREPROCESSING

```
[14]: 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
```

```
[15]: df.describe()
```

```
[15]:                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
```

```
[16]: df.columns
```

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

```
[17]: df1=df.dropna(axis=1)
      df1
```

```
[17]:        Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  \
      0      1            5.1           3.5            1.4           0.2
      1      2            4.9           3.0            1.4           0.2
      2      3            4.7           3.2            1.3           0.2
      3      4            4.6           3.1            1.5           0.2
      4      5            5.0           3.6            1.4           0.2
      ..   …              …             …              …             …
      145  146            6.7           3.0            5.2           2.3
      146  147            6.3           2.5            5.0           1.9
      147  148            6.5           3.0            5.2           2.0
      148  149            6.2           3.4            5.4           2.3
      149  150            5.9           3.0            5.1           1.8

                  Species
      0       Iris-setosa
      1       Iris-setosa
      2       Iris-setosa
      3       Iris-setosa
      4       Iris-setosa
      ..              …
      145  Iris-virginica
      146  Iris-virginica
      147  Iris-virginica
      148  Iris-virginica
      149  Iris-virginica

      [150 rows x 6 columns]
```

```
[18]: df1.columns
```

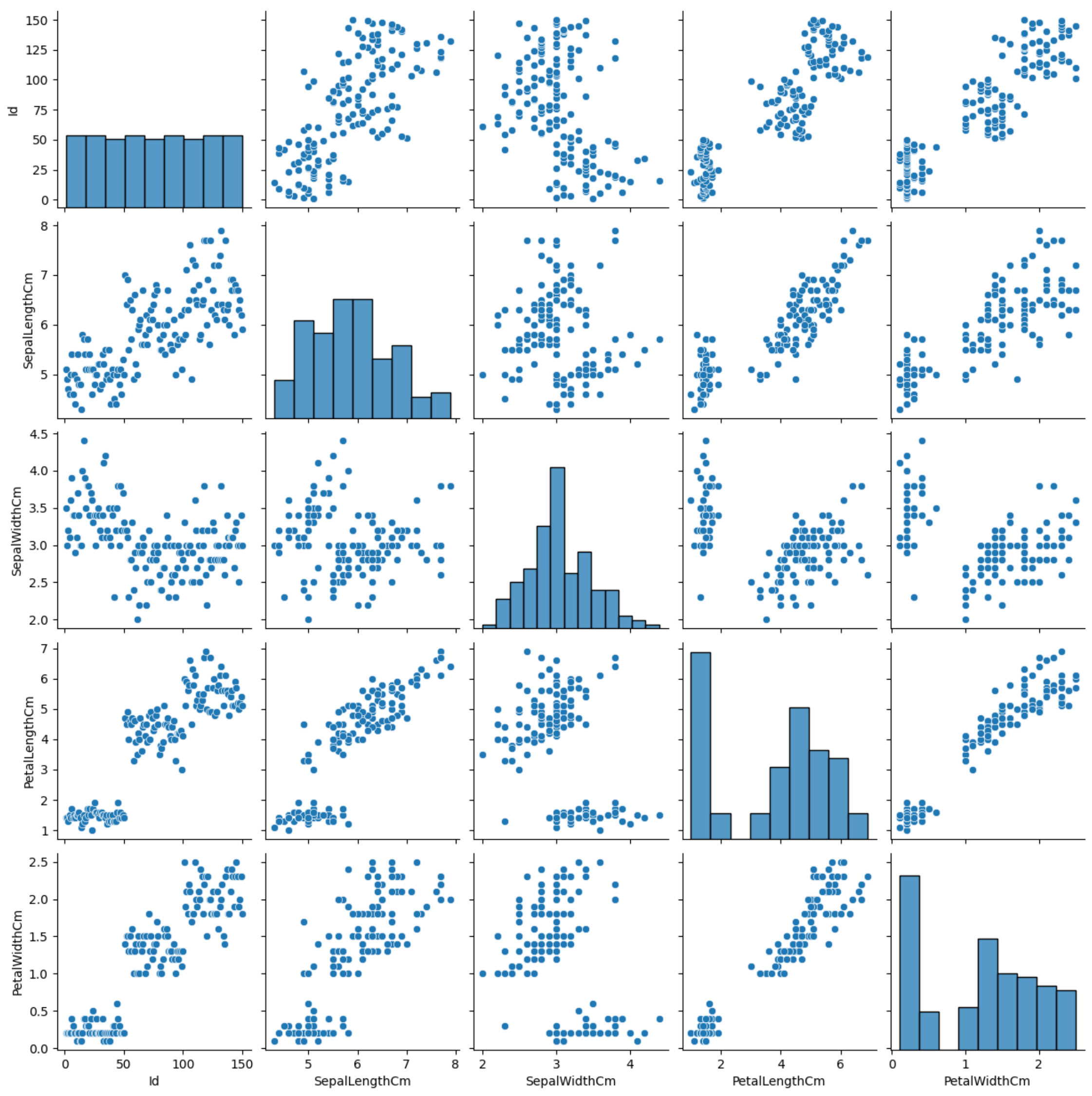
```
[18]: Index(['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm',
             'Species'],
            dtype='object')
```

```
[19]: df1=df1[['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm',
             'Species']]
```

2 EDA AND VISUALIZATION

```
[20]: sns.pairplot(df1)
```

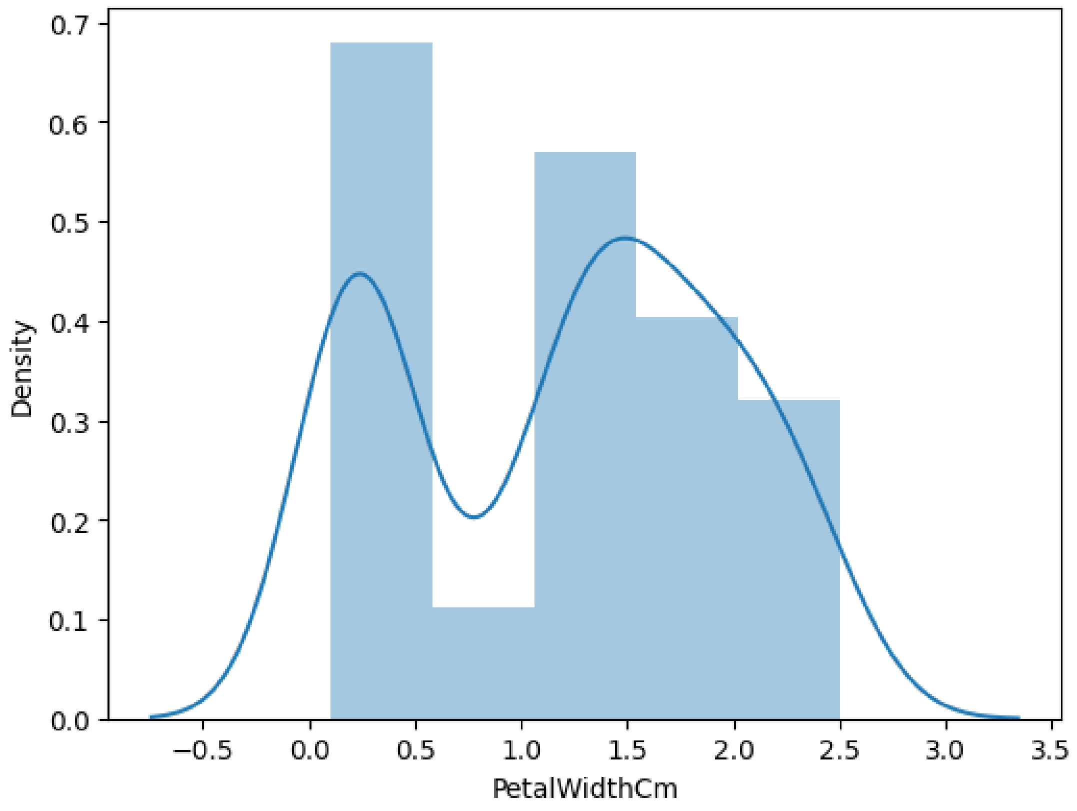
```
[20]: <seaborn.axisgrid.PairGrid at 0x78a53b491420>
```



[21]:
```
sns.distplot(df1['PetalWidthCm'])
```

```
<ipython-input-21-a51f8e882509>:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df1['PetalWidthCm'])
```

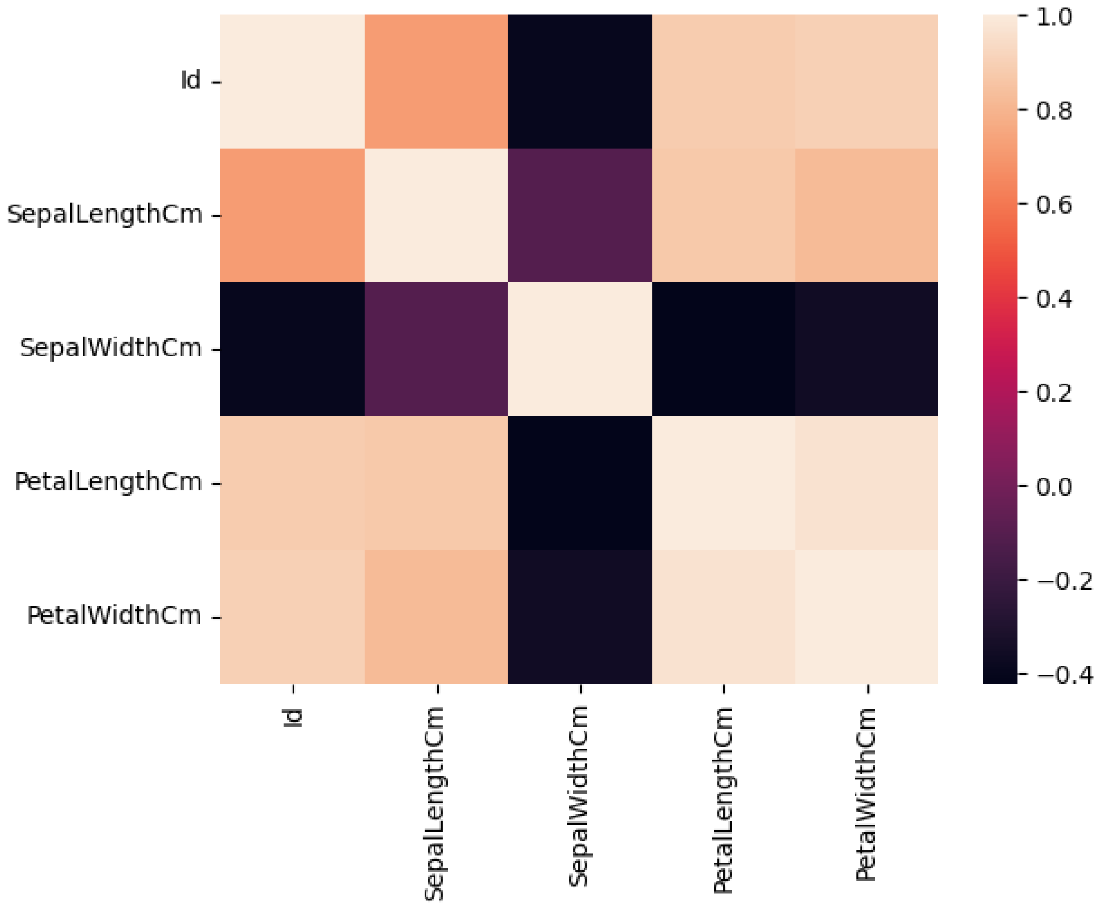
[21]: <Axes: xlabel='PetalWidthCm', ylabel='Density'>



[22]:
```
sns.heatmap(df1.corr())
```

```
<ipython-input-22-3ed1a1a51dc0>:1: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it will
default to False. Select only valid columns or specify the value of numeric_only
to silence this warning.
  sns.heatmap(df1.corr())
```

[22]: <Axes: >



3 TO TRAIN THE MODEL AND MODEL BULDING

[23]:
```
x=df[['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm']]
y=df['PetalWidthCm']
```

[24]:
```
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)
```

[25]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

[25]:
```
LinearRegression()
```

[26]:
```
lr.intercept_
```

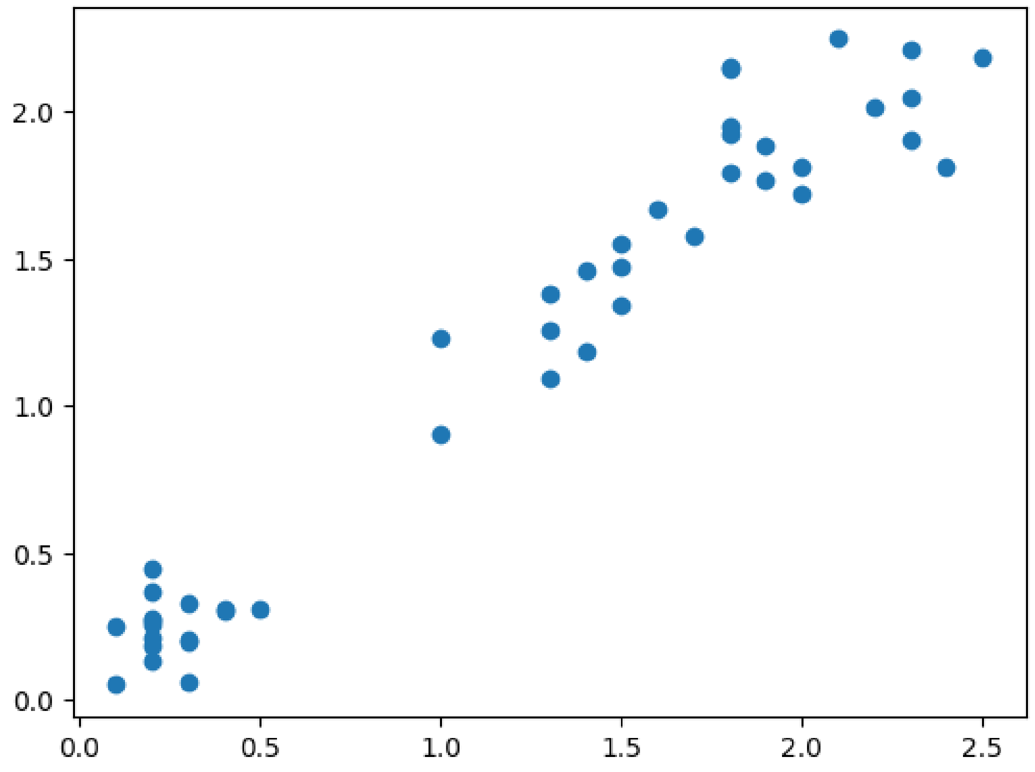
[26]:
```
-0.6111855584816039
```

[27]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

[27]:
```
               Co-efficient
Id                 0.003319
SepalLengthCm     -0.102933
SepalWidthCm       0.204350
PetalLengthCm      0.405633
```

[28]:
```
prediction =lr.predict(x_test)
plt.scatter(y_test,prediction)
```

[28]:
```
<matplotlib.collections.PathCollection at 0x78a5356c0640>
```



4 ACCURACY

[29]:
```
lr.score(x_test,y_test)
```

[29]:
```
0.9440440914460372
```

[30]:
```
lr.score(x_train,y_train)
```

[30]: 0.9456242631741283

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

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

[32]: Ridge(alpha=10)

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

[33]: 0.9317980072920794

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

[34]: 0.9390256034268054

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

[35]: Lasso(alpha=10)

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

[36]: 0.6844575227087786

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

[37]: 0.7102668907630487