Task 2

Prediction using Unsupervised Machine Learning

Author : Abin Johnson

Submitted to: The Sparks Foundation

Importing all the essential Libraries

In [23]:
```
from sklearn.cluster import KMeans
import pandas as pd
import numpy as np
import seaborn as sns
from matplotlib import pyplot as plt
%matplotlib inline
```

Loading the given Iris Dataset

In [3]:
```
pwd
```

Out[3]:
```
'C:\\Users\\Dell\\Spark Foundation'
```

In [44]:
```
ds= pd.read_excel("C:\\Users\\Dell\\OneDrive\\Documents\\Iris2.xlsx")
ds.head()
```

Out[44]:

	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 [45]:
```
ds.shape
```

Out[45]:
```
(150, 6)
```

In [46]:
```
ds.info
```

Out[46]:
```
<bound method DataFrame.info of       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]>
```

In [47]:
```
ds.isnull().sum()
```

Out[47]:
```
Id               0
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

In [48]:
```
ds.drop_duplicates(inplace=True)
```

Label Encoding

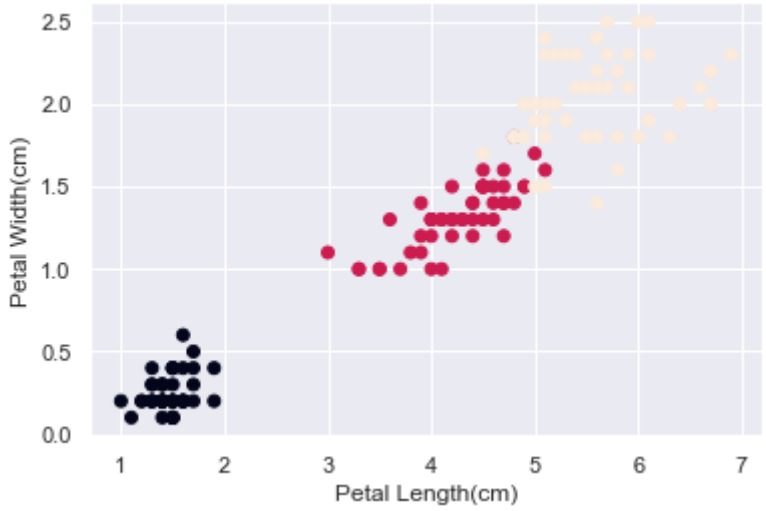
In [49]:
```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
ds['Species']=le.fit_transform(ds['Species'])
ds['Species'].value_counts()
```

Out[49]:
```
0    50
1    50
2    50
Name: Species, dtype: int64
```

PetalLengthCm vs PetalWidthCm

In [51]:
```
plt.scatter(ds['PetalLengthCm'],ds['PetalWidthCm'],c=ds.Species.values)
sns.set(style='darkgrid')
plt.xlabel('Petal Length(cm)')
plt.ylabel('Petal Width(cm)')
```

Out[51]:
```
Text(0, 0.5, 'Petal Width(cm)')
```



In [11]:
```
ds.corr()
```

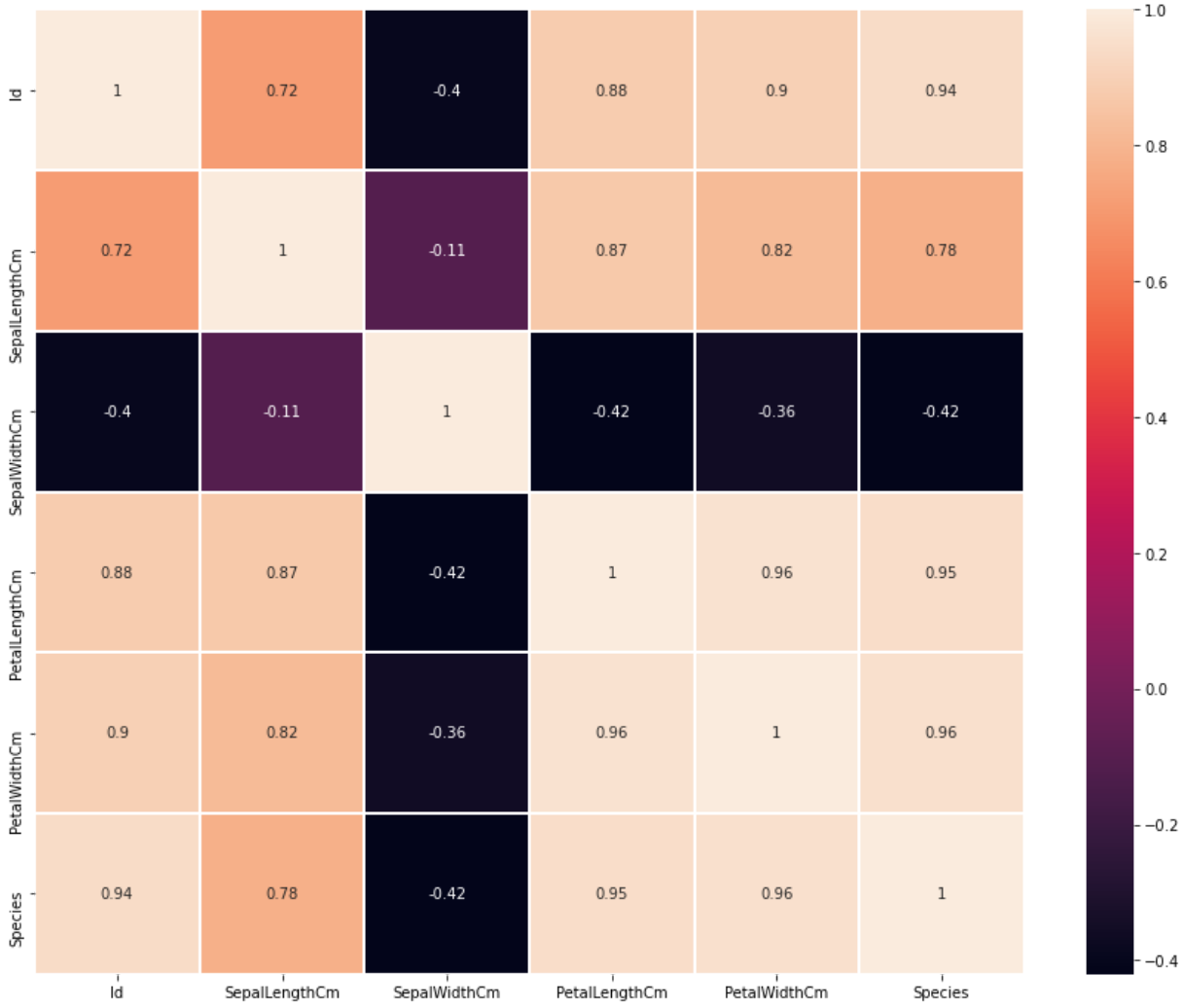
Out[11]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
Id	1.000000	0.716676	-0.397729	0.882747	0.899759	0.94283(
SepalLengthCm	0.716676	1.000000	-0.109369	0.871754	0.817954	0.782561
SepalWidthCm	-0.397729	-0.109369	1.000000	-0.420516	-0.356544	-0.41944(
PetalLengthCm	0.882747	0.871754	-0.420516	1.000000	0.962757	0.949043
PetalWidthCm	0.899759	0.817954	-0.356544	0.962757	1.000000	0.956464
Species	0.942830	0.782561	-0.419446	0.949043	0.956464	1.000000

Data Visualization

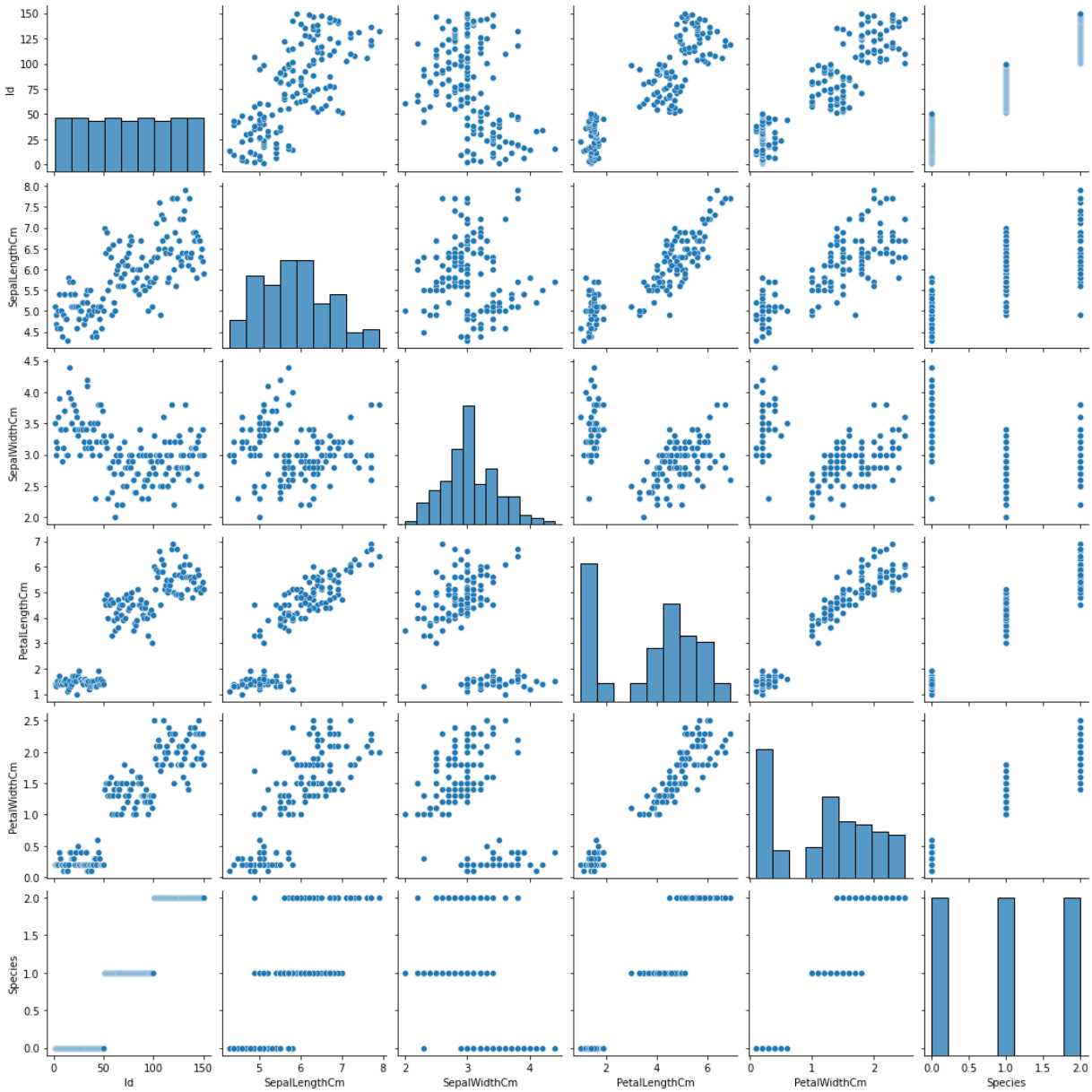
In [52]:
```
fig=plt.figure(figsize=(15,12))
sns.heatmap(ds.corr(),linewidths=1,annot=True)
```

Out[52]: <AxesSubplot:>



In [53]:
```
sns.pairplot(ds)
```

Out[53]: <seaborn.axisgrid.PairGrid at 0x213f4b8b6a0>



Data Preprocessing

In [28]:

```
from sklearn.cluster import KMeans
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters = i, init = 'k-means++', random_state = 42)
    kmeans.fit(ds)
    # inertia method returns wcss for that model
    wcss.append(kmeans.inertia_)

wcss
```

```
C:\Users\Dell\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:881: UserWarnin
g: KMeans is known to have a memory leak on Windows with MKL, when there are less ch
unks than available threads. You can avoid it by setting the environment variable OM
P_NUM_THREADS=1.
  warnings.warn(
```

Out[28]:

```
[281831.54466666654,
 70581.3808,
 31320.711199999994,
 17777.809912280707,
 11422.155508342603,
 7906.99401538462,
 5892.12121917937,
```

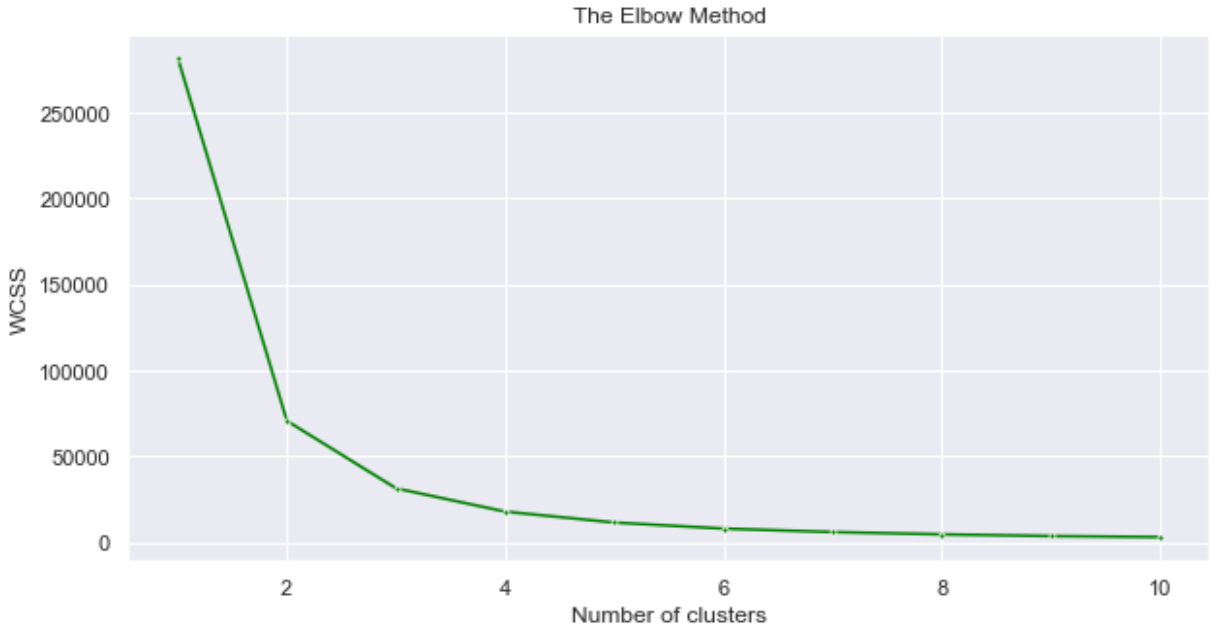
```
4559.9544367045055,
3568.0227491830083,
2962.787655058177]
```

Elbow Method

Plotiing the Graph

In [35]:
```
plt.figure(figsize=(10,5))
sns.set(style='darkgrid')
sns.lineplot(range(1, 11), wcss,marker='.',color='green')
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```

```
C:\Users\Dell\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:
Pass the following variables as keyword args: x, y. From version 0.12, the only vali
d positional argument will be `data`, and passing other arguments without an explici
t keyword will result in an error or misinterpretation.
  warnings.warn(
```



Predicting for ds

In [37]:
```
kmeans = KMeans(n_clusters = 3, init = 'k-means++', random_state = 5)
y_kmeans = kmeans.fit_predict(ds)
y_kmeans
```

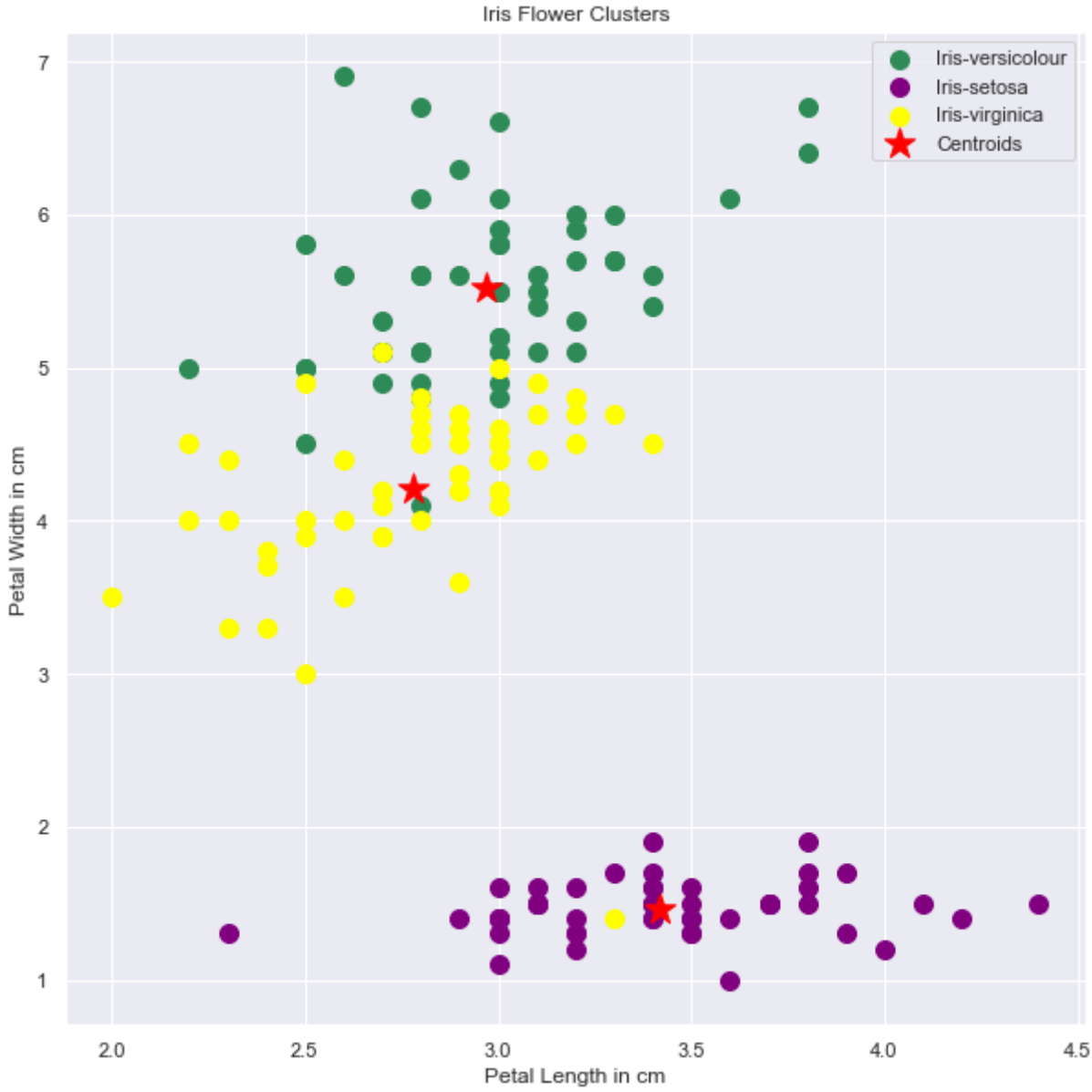
Out[37]:
```
array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
```

Plotting the centroids

In [39]:
```
fig = plt.figure(figsize=(10, 10))
plt.title('Clusters with Centroids',fontweight ='bold', fontsize=30)
plt.scatter(ds[y_kmeans == 0, 2], ds[y_kmeans == 0, 3], s = 100, c = 'seagreen', lab
```

```
plt.scatter(ds[y_kmeans == 1, 2], ds[y_kmeans == 1, 3], s = 100, c = 'purple', label
plt.scatter(ds[y_kmeans == 2, 2], ds[y_kmeans == 2, 3],s = 100, c = 'yellow', label
plt.scatter(kmeans.cluster_centers_[:, 2], kmeans.cluster_centers_[:,3], s = 300, c
            label = 'Centroids')
plt.title('Iris Flower Clusters')
plt.ylabel('Petal Width in cm')
plt.xlabel('Petal Length in cm')
plt.legend()
```

Out[39]: <matplotlib.legend.Legend at 0x2d6620157f0>



In [ ]: