

LetsGrowMore.

Task 1: Prediction using Unsupervised Machine learning

Author:Sujata Sambhaji Gaikwad

```
In [1]: from sklearn.datasets import load_iris
```

```
In [2]: iris=load_iris()
```

```
In [3]: iris.data
```

```
Out[3]: array([[5.1, 3.5, 1.4, 0.2],
               [4.9, 3. , 1.4, 0.2],
               [4.7, 3.2, 1.3, 0.2],
               [4.6, 3.1, 1.5, 0.2],
               [5. , 3.6, 1.4, 0.2],
               [5.4, 3.9, 1.7, 0.4],
               [4.6, 3.4, 1.4, 0.3],
               [5. , 3.4, 1.5, 0.2],
               [4.4, 2.9, 1.4, 0.2],
               [4.9, 3.1, 1.5, 0.1],
               [5.4, 3.7, 1.5, 0.2],
               [4.8, 3.4, 1.6, 0.2],
               [4.8, 3. , 1.4, 0.1],
               [4.3, 3. , 1.1, 0.1],
               [5.8, 4. , 1.2, 0.2],
               [5.7, 4.4, 1.5, 0.4],
               [5.4, 3.9, 1.3, 0.4],
               [5.1, 3.5, 1.4, 0.3],
               [5.7, 3.8, 1.7, 0.3],
               [5.1, 3.2, 1.5, 0.2],
               [5.2, 3.5, 1.4, 0.2],
               [5.3, 3.6, 1.5, 0.2],
               [5.4, 3.7, 1.6, 0.2],
               [5.5, 3.8, 1.7, 0.2],
               [5.6, 3.9, 1.8, 0.2],
               [5.7, 4.0, 1.9, 0.2],
               [5.8, 4.1, 2.0, 0.2],
               [5.9, 4.2, 2.1, 0.2],
               [6.0, 4.3, 2.2, 0.2],
               [6.1, 4.4, 2.3, 0.2],
               [6.2, 4.5, 2.4, 0.2],
               [6.3, 4.6, 2.5, 0.2],
               [6.4, 4.7, 2.6, 0.2],
               [6.5, 4.8, 2.7, 0.2],
               [6.6, 4.9, 2.8, 0.2],
               [6.7, 5.0, 2.9, 0.2],
               [6.8, 5.1, 3.0, 0.2],
               [6.9, 5.2, 3.1, 0.2],
               [7.0, 5.3, 3.2, 0.2],
               [7.1, 5.4, 3.3, 0.2],
               [7.2, 5.5, 3.4, 0.2],
               [7.3, 5.6, 3.5, 0.2],
               [7.4, 5.7, 3.6, 0.2],
               [7.5, 5.8, 3.7, 0.2],
               [7.6, 5.9, 3.8, 0.2],
               [7.7, 6.0, 3.9, 0.2],
               [7.8, 6.1, 4.0, 0.2],
               [7.9, 6.2, 4.1, 0.2],
               [8.0, 6.3, 4.2, 0.2],
               [8.1, 6.4, 4.3, 0.2],
               [8.2, 6.5, 4.4, 0.2],
               [8.3, 6.6, 4.5, 0.2],
               [8.4, 6.7, 4.6, 0.2],
               [8.5, 6.8, 4.7, 0.2],
               [8.6, 6.9, 4.8, 0.2],
               [8.7, 7.0, 4.9, 0.2],
               [8.8, 7.1, 5.0, 0.2],
               [8.9, 7.2, 5.1, 0.2],
               [9.0, 7.3, 5.2, 0.2],
               [9.1, 7.4, 5.3, 0.2],
               [9.2, 7.5, 5.4, 0.2],
               [9.3, 7.6, 5.5, 0.2],
               [9.4, 7.7, 5.6, 0.2],
               [9.5, 7.8, 5.7, 0.2],
               [9.6, 7.9, 5.8, 0.2],
               [9.7, 8.0, 5.9, 0.2],
               [9.8, 8.1, 6.0, 0.2],
               [9.9, 8.2, 6.1, 0.2],
               [10.0, 8.3, 6.2, 0.2],
               [10.1, 8.4, 6.3, 0.2],
               [10.2, 8.5, 6.4, 0.2],
               [10.3, 8.6, 6.5, 0.2],
               [10.4, 8.7, 6.6, 0.2],
               [10.5, 8.8, 6.7, 0.2],
               [10.6, 8.9, 6.8, 0.2],
               [10.7, 9.0, 6.9, 0.2],
               [10.8, 9.1, 7.0, 0.2],
               [10.9, 9.2, 7.1, 0.2],
               [11.0, 9.3, 7.2, 0.2],
               [11.1, 9.4, 7.3, 0.2],
               [11.2, 9.5, 7.4, 0.2],
               [11.3, 9.6, 7.5, 0.2],
               [11.4, 9.7, 7.6, 0.2],
               [11.5, 9.8, 7.7, 0.2],
               [11.6, 9.9, 7.8, 0.2],
               [11.7, 10.0, 7.9, 0.2],
               [11.8, 10.1, 8.0, 0.2],
               [11.9, 10.2, 8.1, 0.2],
               [12.0, 10.3, 8.2, 0.2],
               [12.1, 10.4, 8.3, 0.2],
               [12.2, 10.5, 8.4, 0.2],
               [12.3, 10.6, 8.5, 0.2],
               [12.4, 10.7, 8.6, 0.2],
               [12.5, 10.8, 8.7, 0.2],
               [12.6, 10.9, 8.8, 0.2],
               [12.7, 11.0, 8.9, 0.2],
               [12.8, 11.1, 9.0, 0.2],
               [12.9, 11.2, 9.1, 0.2],
               [13.0, 11.3, 9.2, 0.2],
               [13.1, 11.4, 9.3, 0.2],
               [13.2, 11.5, 9.4, 0.2],
               [13.3, 11.6, 9.5, 0.2],
               [13.4, 11.7, 9.6, 0.2],
               [13.5, 11.8, 9.7, 0.2],
               [13.6, 11.9, 9.8, 0.2],
               [13.7, 12.0, 9.9, 0.2],
               [13.8, 12.1, 10.0, 0.2],
               [13.9, 12.2, 10.1, 0.2],
               [14.0, 12.3, 10.2, 0.2],
               [14.1, 12.4, 10.3, 0.2],
               [14.2, 12.5, 10.4, 0.2],
               [14.3, 12.6, 10.5, 0.2],
               [14.4, 12.7, 10.6, 0.2],
               [14.5, 12.8, 10.7, 0.2],
               [14.6, 12.9, 10.8, 0.2],
               [14.7, 13.0, 10.9, 0.2],
               [14.8, 13.1, 11.0, 0.2],
               [14.9, 13.2, 11.1, 0.2],
               [15.0, 13.3, 11.2, 0.2],
               [15.1, 13.4, 11.3, 0.2],
               [15.2, 13.5, 11.4, 0.2],
               [15.3, 13.6, 11.5, 0.2],
               [15.4, 13.7, 11.6, 0.2],
               [15.5, 13.8, 11.7, 0.2],
               [15.6, 13.9, 11.8, 0.2],
               [15.7, 14.0, 11.9, 0.2],
               [15.8, 14.1, 12.0, 0.2],
               [15.9, 14.2, 12.1, 0.2],
               [16.0, 14.3, 12.2, 0.2],
               [16.1, 14.4, 12.3, 0.2],
               [16.2, 14.5, 12.4, 0.2],
               [16.3, 14.6, 12.5, 0.2],
               [16.4, 14.7, 12.6, 0.2],
               [16.5, 14.8, 12.7, 0.2],
               [16.6, 14.9, 12.8, 0.2],
               [16.7, 15.0, 12.9, 0.2],
               [16.8, 15.1, 13.0, 0.2],
               [16.9, 15.2, 13.1, 0.2],
               [17.0, 15.3, 13.2, 0.2],
               [17.1, 15.4, 13.3, 0.2],
               [17.2, 15.5, 13.4, 0.2],
               [17.3, 15.6, 13.5, 0.2],
               [17.4, 15.7, 13.6, 0.2],
               [17.5, 15.8, 13.7, 0.2],
               [17.6, 15.9, 13.8, 0.2],
               [17.7, 16.0, 13.9, 0.2],
               [17.8, 16.1, 14.0, 0.2],
               [17.9, 16.2, 14.1, 0.2],
               [18.0, 16.3, 14.2, 0.2],
               [18.1, 16.4, 14.3, 0.2],
               [18.2, 16.5, 14.4, 0.2],
               [18.3, 16.6, 14.5, 0.2],
               [18.4, 16.7, 14.6, 0.2],
               [18.5, 16.8, 14.7, 0.2],
               [18.6, 16.9, 14.8, 0.2],
               [18.7, 17.0, 14.9, 0.2],
               [18.8, 17.1, 15.0, 0.2],
               [18.9, 17.2, 15.1, 0.2],
               [19.0, 17.3, 15.2, 0.2],
               [19.1, 17.4, 15.3, 0.2],
               [19.2, 17.5, 15.4, 0.2],
               [19.3, 17.6, 15.5, 0.2],
               [19.4, 17.7, 15.6, 0.2],
               [19.5, 17.8, 15.7, 0.2],
               [19.6, 17.9, 15.8, 0.2],
               [19.7, 18.0, 15.9, 0.2],
               [19.8, 18.1, 16.0, 0.2],
               [19.9, 18.2, 16.1, 0.2],
               [20.0, 18.3, 16.2, 0.2],
               [20.1, 18.4, 16.3, 0.2],
               [20.2, 18.5, 16.4, 0.2],
               [20.3, 18.6, 16.5, 0.2],
               [20.4, 18.7, 16.6, 0.2],
               [20.5, 18.8, 16.7, 0.2],
               [20.
```

```
In [4]: iris.target
```

```
Out[4]: array([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, 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, 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])
```

```
In [5]: from sklearn.cluster import KMeans
kmeans=KMeans(n_clusters=3)
print(kmeans)
```

```
KMeans(n_clusters=3)
```

```
In [6]: kmodel=kmeans.fit(iris.data)
kmodel.labels_
```

```
Out[6]: 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, 1, 1, 0, 0, 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, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 2, 2, 2, 2, 0, 2, 2, 2,
                2, 2, 2, 0, 0, 2, 2, 2, 2, 0, 2, 0, 2, 0, 2, 2, 0, 0, 2, 2, 2, 2,
                2, 0, 2, 2, 2, 2, 0, 2, 2, 2, 0, 2, 2, 2, 0, 2, 2, 0])
```

```
In [7]: kmodel.cluster_centers_
```

```
Out[7]: array([[5.9016129 , 2.7483871 , 4.39354839, 1.43387097],
                [5.006      , 3.428      , 1.462      , 0.246      ],
                [6.85      , 3.07368421, 5.74210526, 2.07105263]])
```

```
In [8]: import pandas as pd
pd.crosstab(iris.target,kmodel.labels_)
```

```
Out[8]:
```

col_0	0	1	2
row_0			
0	0	50	0
1	48	0	2
2	14	0	36

this Prediction is taking dataset on the website

```
In [9]: import pandas as pd
import numpy as np
```

```
In [10]: data=pd.read_csv(r"C:\Users\HP\Downloads\Iris.csv")
data
```

```
Out[10]:
```

	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 [11]: data.isnull().sum()
```

```
Out[11]: Id          0
SepalLengthCm      0
SepalWidthCm       0
PetalLengthCm      0
PetalWidthCm       0
Species            0
dtype: int64
```

```
In [12]: data.describe()
```

```
Out[12]:
```

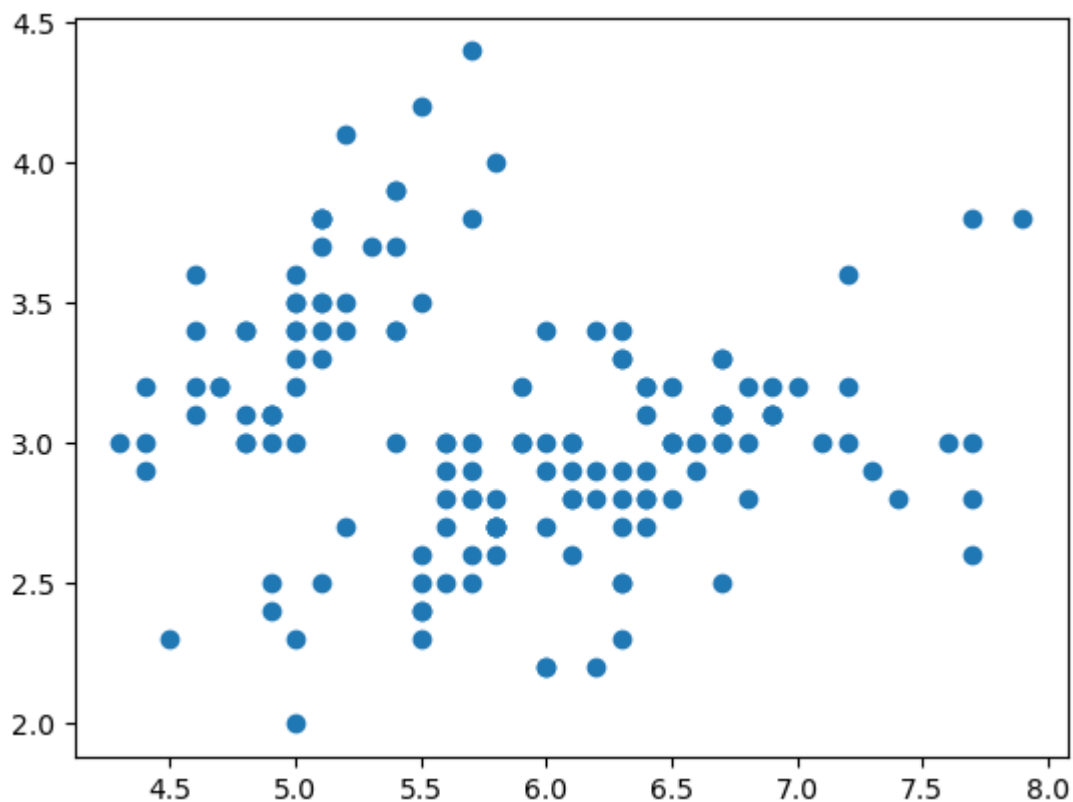
	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	75.500000	5.843333	3.054000	3.758667	1.198667
std	43.445368	0.828066	0.433594	1.764420	0.763161
min	1.000000	4.300000	2.000000	1.000000	0.100000
25%	38.250000	5.100000	2.800000	1.600000	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 [13]: 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
```

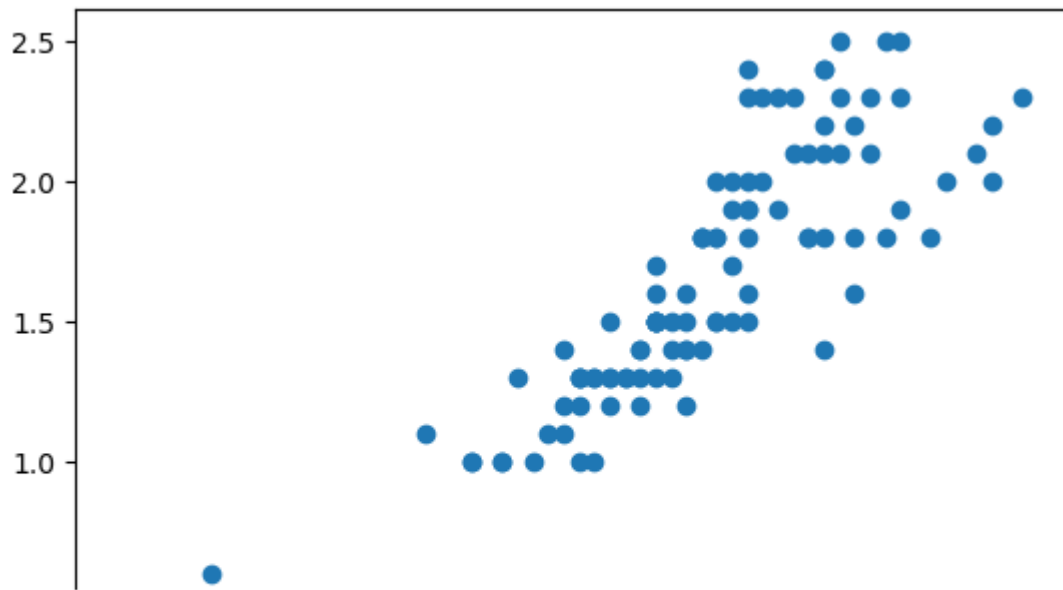
```
In [14]: import matplotlib.pyplot as plt  
plt.scatter(data["SepalLengthCm"], data["SepalWidthCm"])
```

```
Out[14]: <matplotlib.collections.PathCollection at 0x1b12e6ab8e0>
```



```
In [15]: plt.scatter(data["PetalLengthCm"],data["PetalWidthCm"])
```

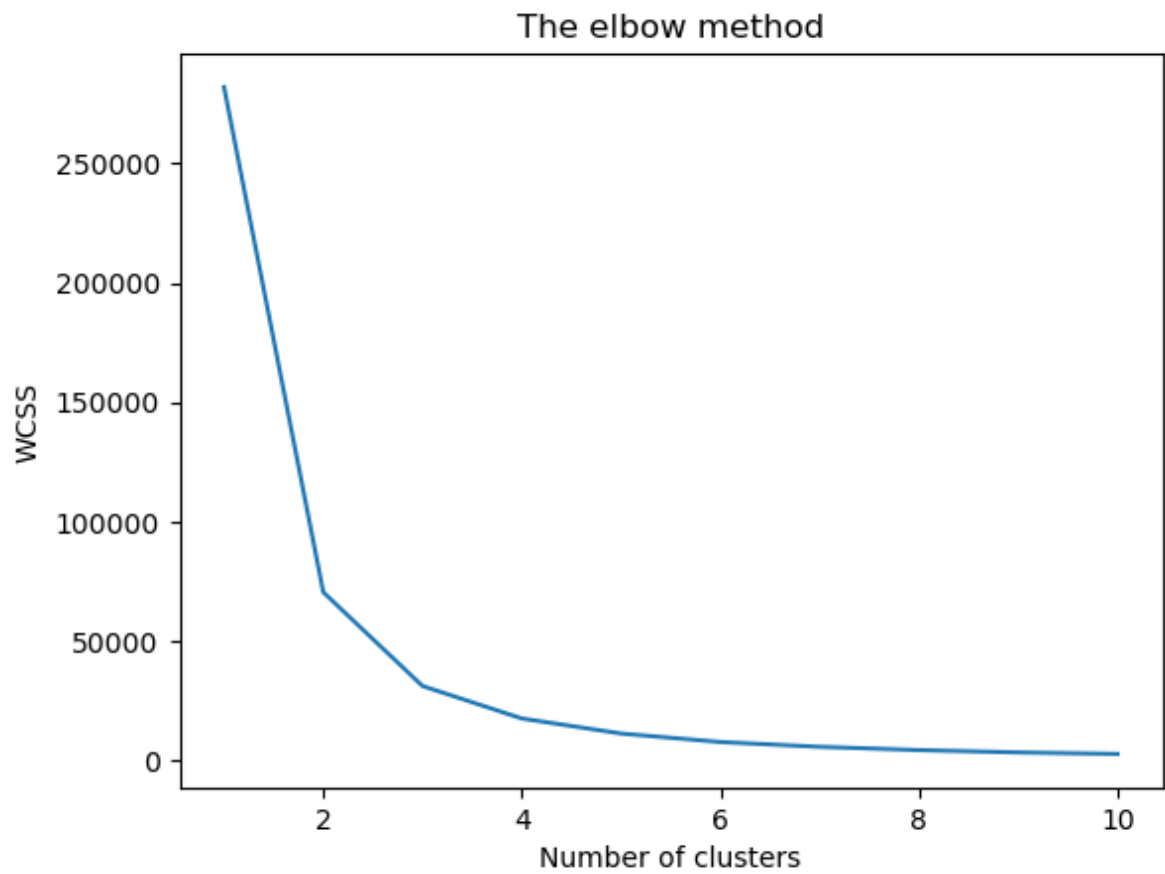
```
Out[15]: <matplotlib.collections.PathCollection at 0x1b12e703910>
```



```
In [16]: x=data.iloc[:,[0,1,2,3]].values
from sklearn.cluster import KMeans
wcss=[]
for i in range(1,11):
    kmeans=KMeans(n_clusters=i,init='k-means++',
                  max_iter=300,n_init=10,random_state=0)
    kmeans.fit(x)
    wcss.append(kmeans.inertia_)
```

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\cluster_kmeans.py:1036: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
warnings.warn(

```
In [17]: plt.plot(range(1, 11), wcss)
plt.title('The elbow method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```



```
In [18]: y_kmeans=kmeans.fit_predict(x)
```

```

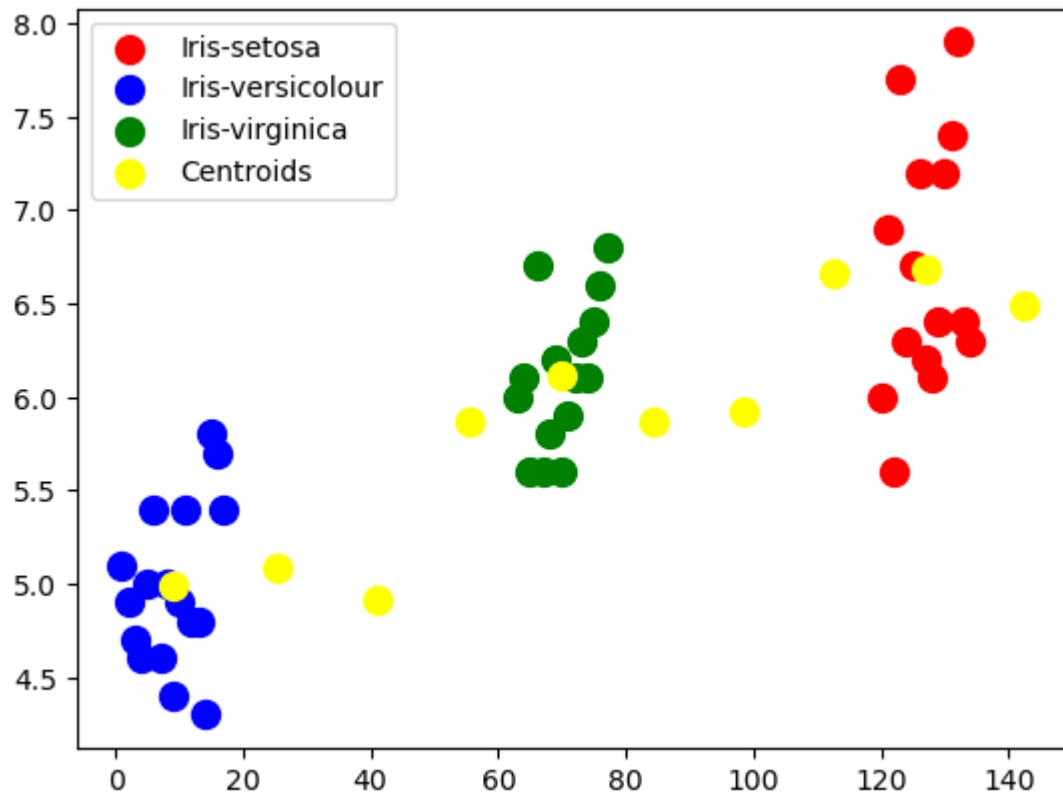
In [19]: plt.scatter(x[y_kmeans == 0, 0], x[y_kmeans == 0, 1], s = 100, c = 'red', label='Iris-setosa')
plt.scatter(x[y_kmeans == 1, 0], x[y_kmeans == 1, 1], s = 100, c = 'blue', label='Iris-versicolour')
plt.scatter(x[y_kmeans == 2, 0], x[y_kmeans == 2, 1], s = 100, c = 'green', label='Iris-virginica')

plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s = 100, c = 'yellow', label='Centroids')

plt.legend()

```

Out[19]: <matplotlib.legend.Legend at 0x1b12e83c8e0>



In []: