NAME: Srivally Garimella

## Task 2: Prediction using unsupervised ML

## Step 1: Import all the required libraries using the following commands

In [3]:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn import datasets
```

## Step 2: Observe the first 5 rows of the following Iris data set

In [5]:

```
iris = datasets.load_iris()
iris_df = pd.DataFrame(iris.data, columns = iris.feature_names)
iris_df.head()
```

Out[5]:

| | sepal length (cm) | sepal width (cm) | petal length (cm) | petal width (cm) |
|---|---|---|---|---|
| **0** | 5.1 | 3.5 | 1.4 | 0.2 |
| **1** | 4.9 | 3.0 | 1.4 | 0.2 |
| **2** | 4.7 | 3.2 | 1.3 | 0.2 |
| **3** | 4.6 | 3.1 | 1.5 | 0.2 |
| **4** | 5.0 | 3.6 | 1.4 | 0.2 |

## Step 3: Find the optimum number of clusters for K Means classification and determine the value of K

In [6]:

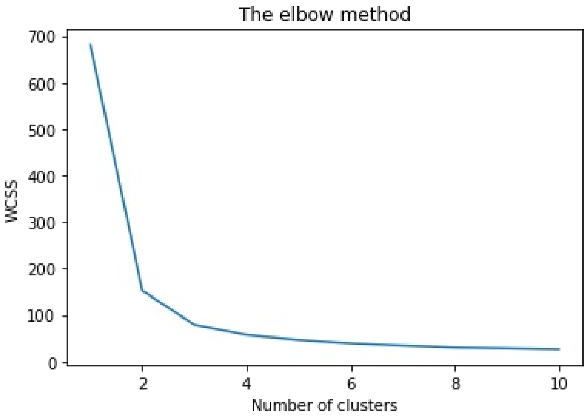
```
x = iris_df.iloc[:, [0, 1, 2, 3]].values

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_)

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

```
C:\Users\Srivally\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(
```



The graph clearly depicts 'The elbow method'. The optimum clusters is where the elbow occurs.

the within cluster sum of squares (WCSS) doesn't decrease significantly with every iteration.

.

## Step 4: Apply K Means to the dataset

In [7]:

```
kmeans = KMeans(n_clusters = 3, init = 'k-means++',
                max_iter = 300, n_init = 10, random_state = 0)
y_kmeans = kmeans.fit_predict(x)
```

## Step 5: Visualize the clusters on the first 2 columns and plot the centroids of the clusters

In [8]:

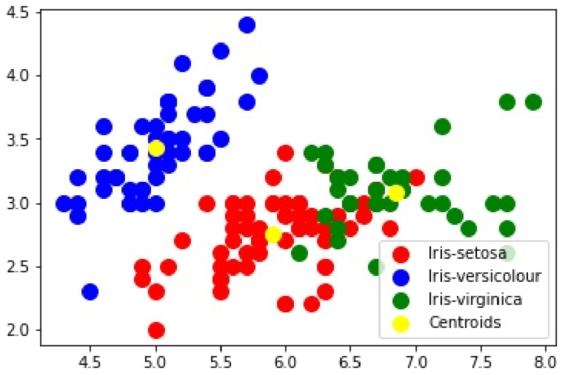
```
plt.scatter(x[y_kmeans == 0, 0], x[y_kmeans == 0, 1],
            s = 100, c = 'red', label = 'Iris-setosa')
plt.scatter(x[y_kmeans == 1, 0], x[y_kmeans == 1, 1],
            s = 100, c = 'blue', label = 'Iris-versicolour')
plt.scatter(x[y_kmeans == 2, 0], x[y_kmeans == 2, 1],
            s = 100,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


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

plt.legend()
```

Out[8]:

```
<matplotlib.legend.Legend at 0x26b70a27f10>
```



Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js