PREDICTION USING UNSUPERVISED ML
(LEVEL-BEGINNER)

AIM : From the given 'Iris' dataset,predict the optimum number of clusters and represent it visually DATASET:https://bit.ly/3kXTdox

To predict the optimum number of clusters in the Iris dataset, we can use the elbow method. The elbow method is a heuristic used to determine the optimal number of clusters in a dataset. It works by plotting the within-cluster sum of squares (WCSS) against the number of clusters used in the clustering process. The WCSS is defined as the sum of the squared distance between each data point and its centroid in the cluster. The idea is to choose the number of clusters where the decrease in WCSS begins to level off (forming an elbow-like shape).

Let's start by loading the necessary libraries and the Iris dataset.

In [6]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.datasets import load_iris

iris = load_iris()
X = iris.data
```

Next, we can apply the KMeans clustering algorithm to the dataset for different values of k and calculate the corresponding WCSS.

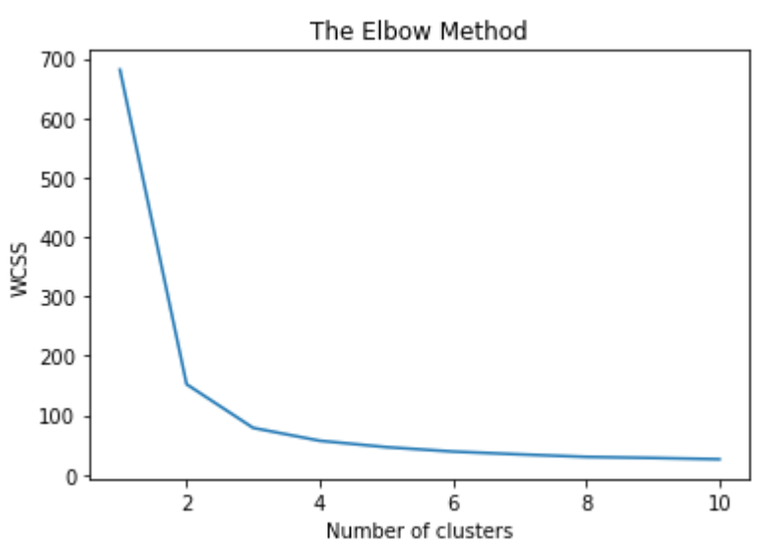
In [9]:
```
wcss = []

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, whe
n there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
```

Here, we have used the KMeans algorithm with a range of k values from 1 to 10. The "init" parameter specifies the method for initializing the centroids, "max_iter" specifies the maximum number of iterations, and "n_init" specifies the number of times the algorithm will be run with different centroid seeds. The "random_state" parameter is used to ensure reproducibility of the results.

Now, we can plot the WCSS against the number of clusters (k) and look for the elbow point.

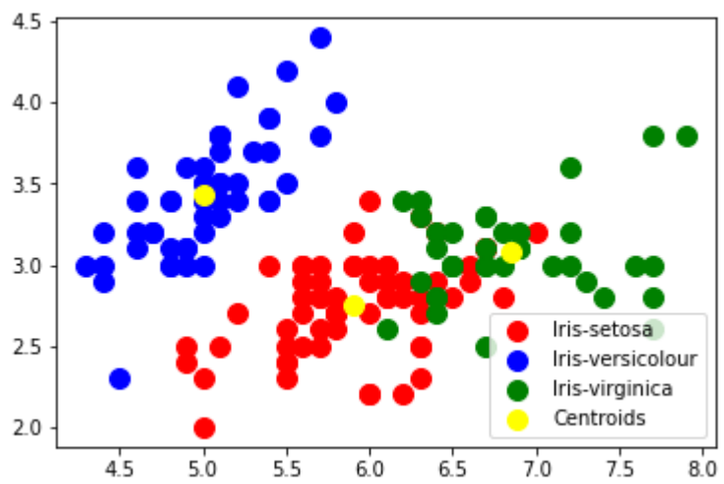
In [4]:
```
plt.plot(range(1, 11), wcss)
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```



The resulting plot will show the elbow point where the decrease in WCSS begins to level off. In this case, the elbow point is at k=3.

This indicates that the optimal number of clusters for this dataset is 3. We can now apply KMeans clustering with k=3 and visualize the clusters.

In [5]:
```
kmeans = KMeans(n_clusters = 3, init = 'k-means++', max_iter = 300, n_init = 10, random_state = 0)
y_kmeans = kmeans.fit_predict(X)

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()
plt.show()
```



The resulting plot will show the clusters identified by the KMeans algorithm, with the centroids represented by yellow circles.