ML2

May 23, 2022

```python
[1]: # Apply K-Means Clustering technique of machine learning to analyze the Iris␣
     ↪dataset.Use Elbow method to find best value of K.
```

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

```python
[3]: iris = load_iris()
```

```python
[4]: x = iris.data
     y = iris.target
```

```python
[9]: # Using Elbow Method

     wcss = []
     for i in range(1,11):
         kmeans = KMeans(n_clusters=i,init='k-means++',n_init=10,random_state=0)
         kmeans.fit(x)
         wcss.append(kmeans.inertia_)

     plt.figure()
     plt.title('Elbow Method')
     plt.xlabel('No  of Clusters')
     plt.ylabel('WCSS')
     plt.plot(range(1,11),wcss)
     plt.show()
```

```
c:\Anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:881: 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(
```
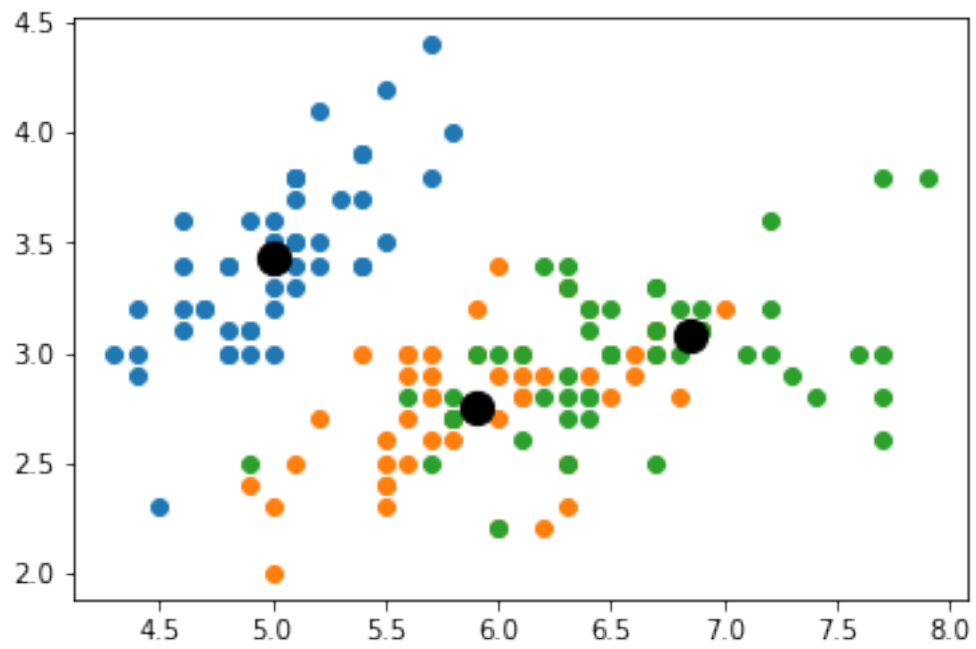
Elbow Method
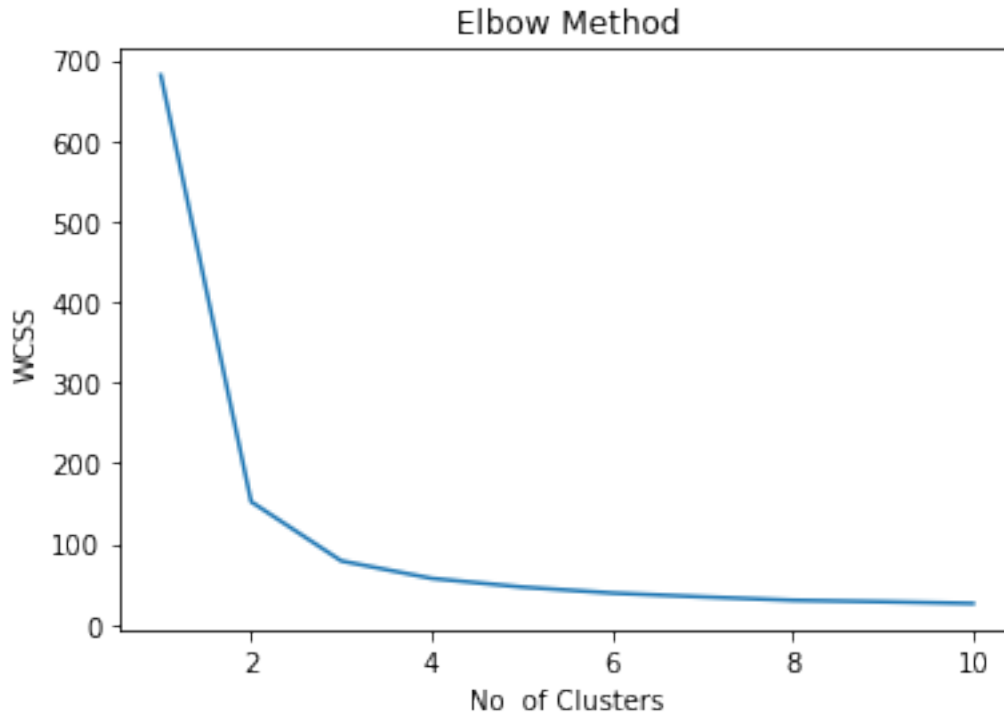
```
[12]: kmeans = KMeans(n_clusters=3,init='k-means++',n_init=10,random_state=0)
      y_kmeans = kmeans.fit_predict(x)
      y_kmeans
```

```
[12]: 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, 2, 0, 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])
```

```
[15]: n_classes = iris.target_names
      for i in range(len(n_classes)):
          index = np.where(y == i)
          plt.scatter(x[index,0],x[index,1],label = iris.target_names[i])

      plt.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:,1],label =␣
       ↪'centroids',c='black',s=150)
      plt.figure()
      plt.show()
```

2

<Figure size 432x288 with 0 Axes>