|  |
| --- |
| Experiment No. 6 |
| Implement clustering algorithm(K-means) |
| Date of Performance: 12/08/24 |
| Date of Submission:02/09/24 |

**AIM:** To Study and Implement K‐Means algorithm

**Objective:** Develop a program to implement K-Means Algorithm

**THEORY:**

In statistics and machine learning, k‐means clustering is a method of cluster analysis which aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean.

Input

K:-number of clusters

D:- data set containing n objects

Output

A set of k clusters

Given *k* , the *k-means* algorithm is implemented in 5 steps:

**Step 1:** Arbitrarily choose k objects from D as the initial cluster centers.

**Step 2:** Find the distance from each and every object in the dataset with respect to cluster centres **Step 3:** Assign each object to the cluster with the nearest seed point based on the mean value of the objects in the cluster.

**Step 4:**Update the cluster means i,e calculate the mean value of the objects for each cluster.  **Step 5:**Repeat the procedure **,** until there is no change in mean.

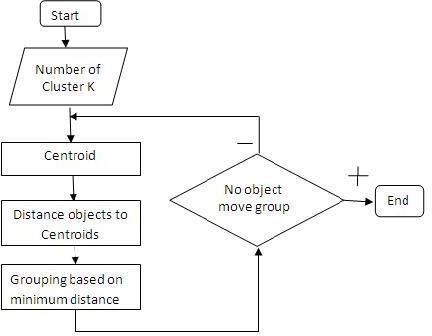


Figure. 1 Flow Chart

**Example:**  d= {2,4,10,12,3,20,30,11,25} k =2

1. Randomly assign mean m1=3 and m2 = 4

Therefore, k1 = {2,3} Therefore, k1 = {4,10,12,20,30,11,25}

1. Randomly assign mean m1=2.5 and m2 =

16 Therefore, k1 = {2,3,4} Therefore, k1 = {4,10,12,20,30,11,25}

1. Randomly assign mean m1=3 and m2 = 18

Therefore, k1 = {2,3,4,10} Therefore, k1 = {12,20,30,11,25}

1. Randomly assign mean m1=7 and m2

= 25 Therefore, k1 = {2,3,4,10,11,12} Therefore, k1 = {20,30,25}

1. Randomly assign mean m1=7 and m2 = 25 Therefore, we stop as we are getting same mean values.
2. Therefore, Final clusters are : k1 = {2,3,4,10,11,12} Therefore, k1 = {20,30,25}

**Code and output**: import numpy as np import matplotlib.pyplot as plt import pandas as pd dataset = pd.read\_csv('Mall\_Customers.csv') X = dataset.iloc[:, [3, 4]].values

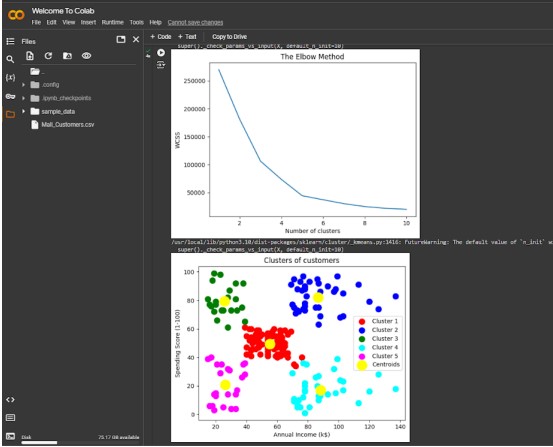
from sklearn.cluster import KMeans wcss = [] for i in range(1, 11):

kmeans = KMeans(n\_clusters = i, init = 'k-means++', random\_state = 42) kmeans.fit(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() kmeans = KMeans(n\_clusters = 5, init = 'k-means++', random\_state = 42) y\_kmeans = kmeans.fit\_predict(X) plt.scatter(X[y\_kmeans == 0, 0], X[y\_kmeans == 0, 1], s = 100, c = 'red', label = 'Cluster 1') plt.scatter(X[y\_kmeans == 1, 0], X[y\_kmeans == 1, 1], s = 100, c = 'blue', label = 'Cluster 2') plt.scatter(X[y\_kmeans == 2, 0], X[y\_kmeans == 2, 1], s = 100, c = 'green', label = 'Cluster 3') plt.scatter(X[y\_kmeans == 3, 0], X[y\_kmeans == 3, 1], s = 100, c = 'cyan', label = 'Cluster 4') plt.scatter(X[y\_kmeans == 4, 0], X[y\_kmeans == 4, 1], s = 100, c = 'magenta', label = 'Cluster 5')

plt.scatter(kmeans.cluster\_centers\_[:, 0], kmeans.cluster\_centers\_[:, 1], s = 300, c = 'yellow', label = 'Centroid

s')

plt.title('Clusters of customers') plt.xlabel('Annual Income (k$)') plt.ylabel('Spending Score (1-100)') plt.legend() plt.show()



**Conclusion :**

We successfully perform Implement clustering algorithm(K-means) .