Marwadi University	**Marwadi University** **Faculty of Technology** **Department of Information and Communication Technology**	
Subject: Machine Learning (01CT0519)	**Aim: To obtain the distint clusters for unsupervised data using KMeans Clustering**	
Experiment No: 10	**Date:28-09-2022**	**Enrolment No:92000133018**

Aim: To obtain the distint clusters for unsupervised data using KMeans Clustering

IDE: Google Colab

Theory:

K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters. Here K defines the number of pre-defined clusters that need to be created in the process, as if K=2, there will be two clusters, and for K=3, there will be three clusters, and so on. It is an iterative algorithm that divides the unlabeled dataset into k different clusters in such a way that each dataset belongs only one group that has similar properties.

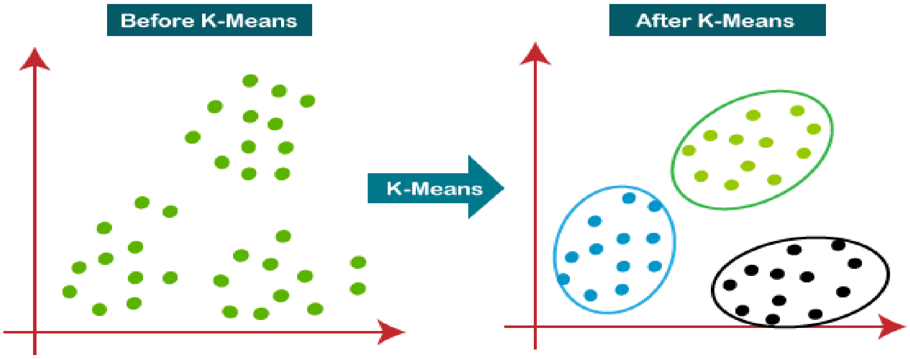
It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabeled dataset on its own without the need for any training.

It is a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.
The algorithm takes the unlabeled dataset as input, divides the dataset into k-number of clusters, and repeats the process until it does not find the best clusters. The value of k should be predetermined in this algorithm.

The k-means clustering algorithm mainly performs two tasks:

- Determines the best value for K center points or centroids by an iterative process.
- Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.

Hence each cluster has datapoints with some commonalities, and it is away from other clusters.
The below diagram explains the working of the K-means Clustering Algorithm:



How does the K-Means Algorithm Work?

The working of the K-Means algorithm is explained in the below steps:
Step-1: Select the number K to decide the number of clusters.
Step-2: Select random K points or centroids. (It can be other from the input dataset).
Step-3: Assign each data point to their closest centroid, which will form the predefined K clusters.
Step-4: Calculate the variance and place a new centroid of each cluster.
Step-5: Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.
Step-6: If any reassignment occurs, then go to step-4 else go to FINISH.
Step-7: The model is ready.

Program (Code):

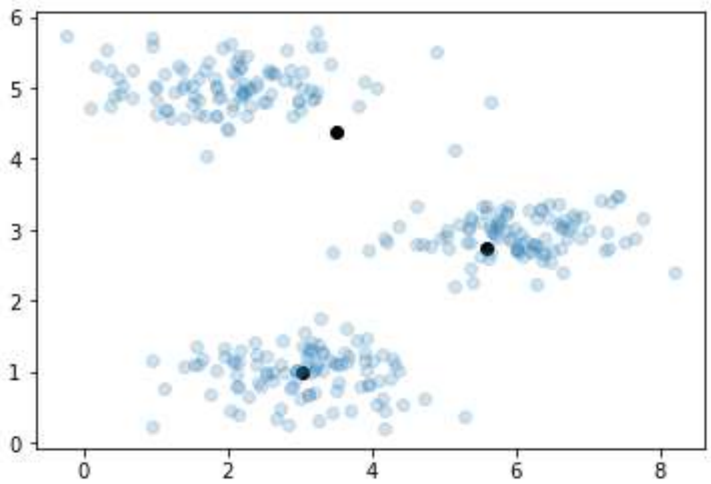
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from google.colab import drive

drive.mount('/content/drive')

cd /content/drive/"MyDrive/"

df1 = pd.read_excel('/content/drive/MyDrive/kmean.xlsx',header=None)
df1.head()

X = np.array(df1)

import random
initial_centroid=random.sample(range(0,len(df1)),3)

initial_centroid

centroid=[]
for i in initial_centroid:
  centroid.append(df1.iloc[i])
centroid

centroids=np.array(centroid)

centroids
```

```
def euclidean_distance(x1,x2):
  return(sum((x1-x2)**2))**0.5

def find_closest_centroid(centroid,x):
  assigned_cluster=[]
  for i in x:

    distance=[]
    for j in centroid:
      distance.append(euclidean_distance(i,j))
    assigned_cluster.append(np.argmin(distance))
  return assigned_cluster

get_centroid=find_closest_centroid(centroids,X)

centroids

get_centroid

def centroid_update(clusters,X):
  new_centroid=[]
  new_df=pd.concat([pd.DataFrame(X),pd.DataFrame(clusters,columns=['cluster'])],axis=1)
  for c in set(new_df['cluster']):
    c_cluster=new_df[new_df['cluster']==c][new_df.columns[:-1]]
    new_mean=c_cluster.mean(axis=0)
    new_centroid.append(new_mean)
  return new_centroid

#training process
for i in range(10):
  get_centroid=find_closest_centroid(centroids,X)
  new_centroids=centroid_update(get_centroid,X)
  #plot the figure
  plt.figure()
  plt.scatter(np.array(new_centroids)[:,0],np.array(new_centroids)[:,1],color="black") #centroid
  plt.scatter(X[:,0],X[:,1],alpha=0.2)
  plt.show()
```

Results:

To be attached with

a. Cluster distribution with the centroid after each iteration



Observation and Result Analysis:

a. Initialization of the centroid

b. Behavior of the centroid and cluster distribution after each iteration

c. When the cluster gets settled

Post Lab Exercise:

a. Is Feature Scaling required for the K means Algorithm?

b. Which metrics can you use to find the accuracy of the K means Algorithm?

c. What are the advantages and disadvantages of the K means Algorithm?

d. What are the ways to avoid the problem of initialization sensitivity in the K means Algorithm?

e. Differentiate Clustering and classification

Post Lab Activity:

Consider any dataset from **https://archive.ics.uci.edu/ml/datasets.php** and perform the clustering and obtain the best divided clusters. Make sure that the dataset is not matching with your classmates. You can also select the dataset from other ML repositories with prior permission from your concerned subject faculty.