Name: Sagar Patil
Roll No: 8061

Assignment 4

Title : We have given a collection of 8 points. P1=[0.1,0.6], P2=[0.15,0.71], P3=[0.08,0.9],P4=[0.16,0.85], P5=[0.2,0.3], P6=[0.25,0.5], P7=[0.24,0.1], P8=[0.3,0.2]. Perform the k-mean clustering with initial centroids as m1=P1=Cluster#1=C1 and m2=P8=cluster#2=C2.

Objectives : Learn How to Apply K Mean Clustering for given data points.

Theory : Clustering is dividing data points into homogeneous classes or clusters:Points in the same group are as similar as possible. Points in different groups are as dissimilar as possible When a collection of objects is given, we put objects into groups based on similarity.

A Clustering Algorithm tries to analyse natural groups of data on the basis of some similarity. It locates the centroid of the group of data points. To carry out effective clustering, the algorithm evaluates the distance between each point from the centroid of the cluster. K-means is one of the simplest unsupervised learning algorithms that solve the well-known clustering problem. K-means clustering is a method of vector quantization, originally from signal processing, that is popular for cluster analysis in data mining.

Algorithm :

1. Import the Required Packages
2. Create dataset using DataFrame
3. Find centroid points
4. plot the given points
5. for i in centroids():
6. plot given elements with centroid elements

7. import KMeans class and create an object of it.

8. using labels to find the population around centroid. 9. Find new centroids.

Code and Output :

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

#create dataset using DataFrame
    df = pd.DataFrame({'X':[0.1,0.15,0.08,0.16,0.2,0.25,0.24,0.3],
                       'y':[0.6,0.71,0.9,0.85,0.3,0.5,0.1,0.2]})
f1 = df['X'].values
f2 = df['y'].values
X = np.array(list(zip(f1, f2)))
print(X)

#centroid points
C_x = np.array([0.1,0.3])
C_y = np.array([0.6,0.2])
centroids=C_x,C_y
#plot the given points
```

```
colmap = {1: 'r', 2: 'b'}
plt.scatter(f1, f2, color='k')
plt.show()

#for i in centroids():
plt.scatter(C_x[0],C_y[0],     color=colmap[1])
plt.scatter(C_x[1],C_y[1],     color=colmap[2])
plt.show()

C = np.array(list((C_x, C_y)), dtype=np.float32)
print (C)

#plot given elements with centroid elements
plt.scatter(f1, f2, c='#050505')
plt.scatter(C_x[0], C_y[0], marker='*', s=200, c='r')
plt.scatter(C_x[1], C_y[1], marker='*', s=200, c='b')
plt.show()
#import KMeans class and create object of it
from sklearn.cluster import KMeans
model=KMeans(n_clusters=2,random_state=0)
model.fit(X)
labels=model.labels_
```

```
print(labels)


#using labels find population around centroid
count=0
for i in range(len(labels)):
    if (labels[i]==1):
        count=count+1


print('No of population around cluster 2:',count-1)


#Find new centroids
new_centroids = model.cluster_centers_


print('Previous value of m1 and m2 is:')
print('M1==',centroids[0])
print('M1==',centroids[1])
print('updated value of m1 and m2 is:')
print('M1==',new_centroids[0])
print('M1==',new_centroids[1])
```

'''

OUTPUT

[[0.1 0.6]

 [0.15 0.71]

 [0.08 0.9]

 [0.16 0.85]

 [0.2 0.3]

 [0.25 0.5]

 [0.24 0.1]

 [0.3 0.2]]

[[0.1 0.3]

 [0.6 0.2]]

[1 1 1 1 0 0 0 0]

No of population around cluster 2: 3
Previous value of m1 and m2 is:

M1== [0.1 0.3]

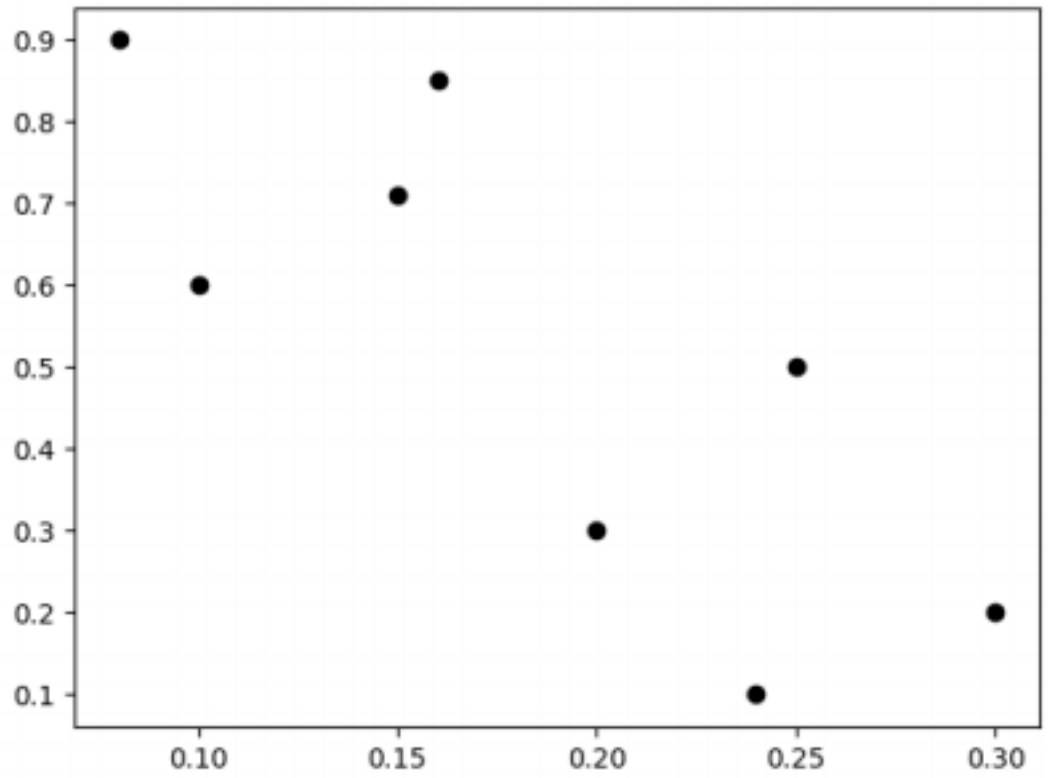
M1== [0.6 0.2]

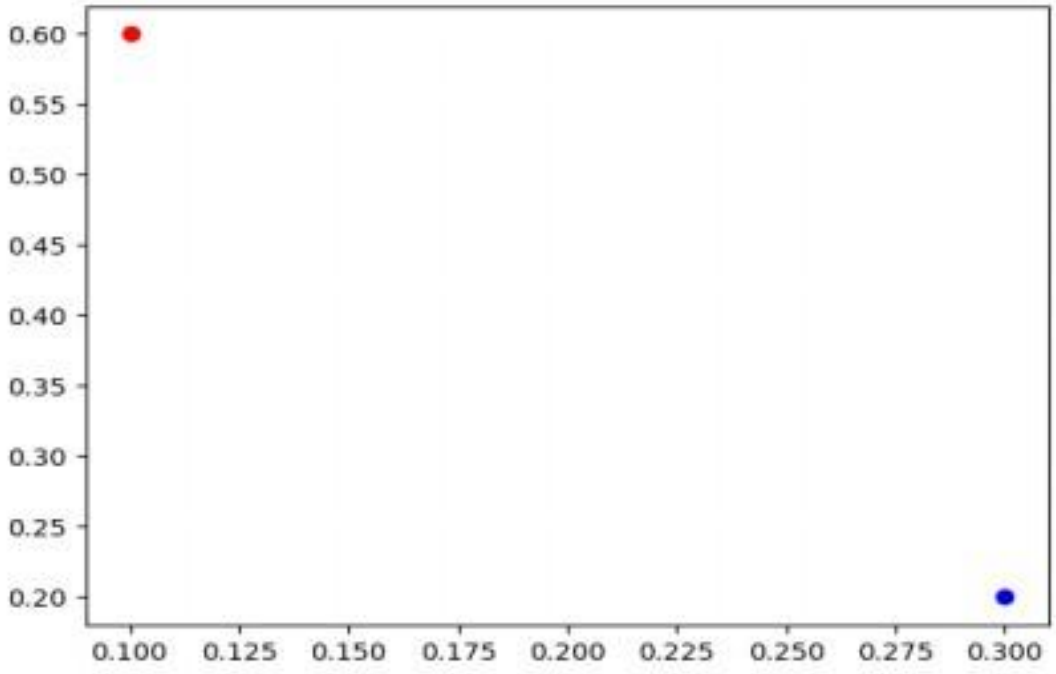
updated value of m1 and m2 is:

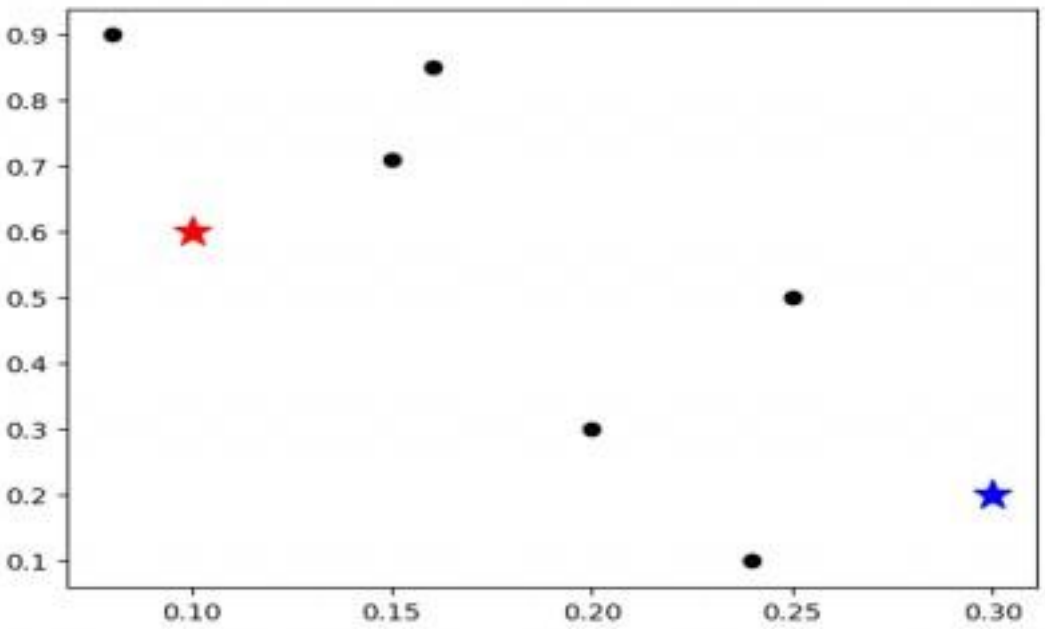
M1== [0.2475 0.275]

M1== [0.1225 0.765]

'''


0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.10
0.15
0.20
0.25
0.30


0.60
0.55
0.50
0.45
0.40
0.35
0.30
0.25
0.20
0.100
0.125
0.150
0.175
0.200
0.225
0.250
0.275
0.300



Conclusion : In this way we learned the K Means Clustering Algorithm.