**ASSIGNMENT: - 04**

**Problem Statement: -**

Write a program to do following:

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. Answer the following:

a) Which cluster does P6 belong to?

b) What is the population of a cluster around m2?

c) What is the updated value of m1 and m2?

**Software, library and package**

1. Software: Python, Google Colab
2. Library: scikit-learn (sklearn) - for k-means clustering algorithm
3. Package: NumPy (implicitly used by scikit-learn for numerical computations)

**Theory:**

**Methodology**:

* K-means Clustering is a popular unsupervised machine learning algorithm used for partitioning data into distinct clusters. It groups the unlabeled dataset into different clusters. Here K defines the number of predefined 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.
* The algorithm aims to minimize the variance within each cluster while maximizing the variance between clusters. The process involves iteratively assigning data points to the nearest cluster centroid and updating the centroids based on the mean of the points assigned to each cluster.
* The k-means clustering algorithm mainly performs two tasks:

1. 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.

**Advantages**:

1. **Simplicity**: K-means is easy to implement and understand, making it suitable for exploratory data analysis.
2. **Scalability**: It can handle large datasets efficiently.
3. **Versatility**: Works well with numerical data and is robust to outliers.

**Applications**:

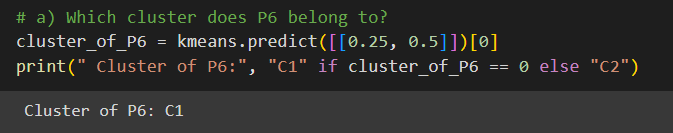
1. **Customer Segmentation**: Identifying customer segments based on purchasing behavior.
2. **Image Compression**: Clustering similar pixels to compress images.
3. **Anomaly Detection**: Identifying outliers or anomalies in data.
4. **Recommendation Systems**: Grouping similar items or users in recommendation engines.

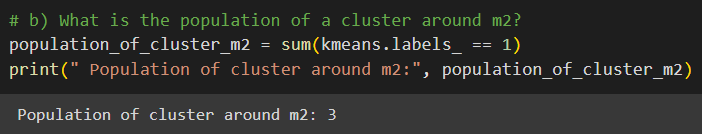
**Limitations**:

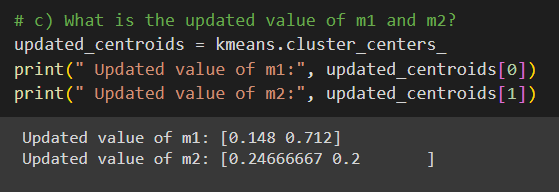
1. **Number of Clusters (K)**: The user needs to specify the number of clusters, which may not always be known a priori.
2. **Sensitive to Initial Centroids**: Results can vary based on the initial centroid placement.
3. **Assumes Spherical Clusters**: K-means works best with spherical clusters and struggles with non-linear or irregularly shaped clusters.

**Working/ Algorithm:**

1. Import necessary libraries (numpy, sklearn.cluster.KMeans).
2. Define data points as a numpy array.
3. Initialize initial centroids (m1 = P1, m2 = P8) as a list.
4. Create a KMeans object with n\_clusters=2, init=initial\_centroids, n\_init=1.
5. Fit K-means model to data points.
6. Predict the cluster of P6 using kmeans.predict([[0.25, 0.5]]).
7. Print the cluster of P6.
8. Calculate the population of the cluster around m2 using sum(kmeans.labels\_ == 1).
9. Print the population of the cluster around m2.
10. Get the updated centroid values using kmeans.cluster\_centers\_.
11. Print the updated values of m1 and m2.







**Conclusion:**

In conclusion, k-means clustering is a versatile and easy-to-implement algorithm for unsupervised clustering tasks. Its advantages include simplicity, scalability, and versatility in handling large datasets and numerical data. However, it requires specifying the number of clusters (K), is sensitive to initial centroid placement, and assumes spherical clusters. Despite its limitations, k-means finds applications in various domains like customer segmentation, image compression, anomaly detection, and recommendation systems.