Assignment: 4

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 Package:

Python with numpy , scikit-learn , matplotlib and seaborn.

1. Theory:

K-means clustering is an unsupervised machine learning algorithm that divides a dataset into a specified number of clusters. It assigns data points to clusters based on their similarity to cluster centroids, updating centroids iteratively until convergence. Widely used for tasks like image and customer segmentation, it's efficient and scalable. However, it's sensitive to initial centroid selection and may require multiple runs for optimal results.

1.1 Methodology

The program utilizes the K-means clustering algorithm, a popular method for unsupervised clustering. It leverages the `KMeans` class from the `sklearn.cluster` library to perform the clustering task.

1.2 Advantages and Applications

- Advantages:

  - Efficient implementation provided by `sklearn.cluster`.

  - Easy integration with other machine learning workflows.

  - Utilizes optimized algorithms for improved performance.

- Applications:

  - Commonly used in data analysis, pattern recognition, and image segmentation.

  - Suitable for tasks like customer segmentation, anomaly detection, and document clustering.

1.3 Limitations

- Sensitivity to initial centroid selection, although mitigated in this program by specifying initial centroids.

- May converge to local optima, depending on the initialization and the dataset's characteristics.

- Assumes clusters are spherical and of similar size, which might not hold true in all scenarios.

2. Working/Algorithm

The program follows these steps:

1. Initialization: Initializes centroids based on the specified initial centroids.

2. Clustering: Utilizes the `fit` method of the `KMeans` class to assign points to clusters.

3. Updating Centroids: After clustering, the updated centroids are obtained using the `cluster_centers_` attribute of the fitted `KMeans` object.

4. Analysis: Various computations are performed to answer specific questions about the clustering results.

3. Conclusion

This Python program effectively applies the K-means clustering algorithm to the given dataset. It leverages the `KMeans` class from the `sklearn.cluster` library, providing a robust and efficient implementation. By utilizing established libraries and functions, the program demonstrates good programming practices by leveraging existing tools for machine learning tasks. Additionally, it provides insights into the clustering results through computed values and visualization using `matplotlib.pyplot`.