5/5/24

Artificial Intellegance

21F-9217
21F-9323
6C

Assignment # 6

Task # 1:

K-Means Clustesing: with k=2.

1- Initialization:
We'll start by randomly selecting two initial centroids from the dataset.

2. Assignment:
Assign each point to the nearest centroid.

3 - Update centroids. (Based on the mean)

4. Repeat 2 and 3 until convergence.

So,

1- Let's choose A1 and A2 as initial centroids.

2 Assign each point to nearest centroid, A1 or A2

K-Means clustering:

Initial centroids: A1 (2,3,1) , A2(3,4,7)

Iteration 1:

cluster 1: A1, A6

cluster 2: A2, A3, A4, A5, A7.

Update centroids:

Cluster 1: Centroid = Mean of (2,3,1) and (7,2,6)

Centroid = (4.5, 2.5, 3.5)

... medoid, since it has

Cluster2: centroid = Mean of (5,4,7), (5,6.5), (6,7,9), (8,9,10), (1,3,11)

centroid = (4.6, 5.8, 8.4)

Iteration 2:

Assignment:

Cluster1: A1, A6

Cluster2: A2, A3, A4, A5, A7

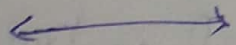
Update:

No change in centroids.

The algorithm converges so final centroids are:

Cluster1 Centroid: (4.5, 2.5, 3.5)

Cluster2 Centroid: (4.6, 5.8, 8.4)



K-Medoids Clustering: K=2:

Initial medoids: A1(2,3,11), A2(3,4,7)

Iteration 1:

Assign:

cluster1: A1, A6

cluster2: A2, A3, A4, A5, A7

Update medoids:

Cluster 1: Choose A1 as the medoid since it has minimum dissimilarity within the cluster.

Cluster 2: Choose A3 as medoid, since it has the minimum dissimilarity within the cluster.

→ The algorithm converges and final centroids are:

Cluster 1: centroid: A1 (2,3,1)
Cluster 2: Centroid: A2 (5,6,5)

Task # 4:

Sol:

Comparing k-means and k-medoids clustering algorithms reveals insights into their characteristics, strengths and weaknesses.

- Algorithm Description:

↳ k-means: It partitions data into k clusters by iteratively assigning each data point to the nearest centroid and updating the centroids to minimize the within-cluster sum of squares.

↳ Where k-medoids are: It partitions data into k clusters by selecting k representative objects called medoids and assigning each data point to the nearest medioid, updating medoids to minimize dissimilarities within clusters.

- Computational Complexity:

↳ k-means: It has a time complexity of $O(n*k*d*iter)$, where n is the number of data points, k is the number of clusters, d is the dimensionality of the data, and iter is the number of iterations until convergence.

↳ k-medoids: It has a time complexity of $O(n^2*k*iter)$, where n is the number of data points, k is the number of clusters and iter is the number of iterations until convergence.

So, k-means tends to be more computationally efficient for large datasets because it calculates centroids based on the mean of data points, while k-medoids calculates medoids based on the actual data points.

- Initialization Sensitivity:

  ↳ k-means: It is sensitive to the initial selection of centroids. Different initializations can lead to different final clustering outcomes. It may converge to local minima.

  ↳ k-medoids: It is less sensitive to the initial selection of medoids compared to k-means. However the choice of initial medoids can still affect the clustering outcome.

- Robustness to outliers:

  ↳ k-means: It is less robust to outliers because it minimizes the squared Euclidean distance, which is sensitive to extreme values. Outliers can significantly affect the position of centroids and, consequently, the clustering result.

  ↳ k-medoids: It is more robust to outliers compared to k-means because it uses the actual data points (medoids) to represent clusters. Outliers have less impact on the choice of medoids since they are less influenced by extreme values.

Overall, while k-means is computationally efficient and widely used, k-medoids tends to be more robust to outliers and less sensitive to initialization. However, the choice between the two algorithms depends on the specific characteristics of the dataset and the desired clustering outcome.