SAN JOSÉ STATE UNIVERSITY

CMPE 239 - HomeWork 6

K-Means Algorithm

1) Name: Jayam Malviya
   SJSU Id: 01435567

2) Rank and NMI score at the time of writing report :
   Rank : 9
   F1 score : 0.5029

3) Approach:

Preprocessing

- First we have read the train data line by line and have stored the row number, column number and the value in three separate arrays.
- These arrays are used to create the COO_Format of the data. We have first created a coo_matrix representation from our data, as this representation was easy to create using the data we were given, each document becomes the row number and each feature on that document becomes the column number for that document and as our is just a binary data we record "1" for that corresponding row & column in the matrix. After capturing the row, col & data in the array we convert them to numpy arrays for creating the coo_matrix representation. Finally we convert the coo_matrix representation to CSR_Matrix representation for the data.

- We have used scipy.sparse -> coo_matrix to hold our preprocessed data.
- Then we have normalized the CSR data matrix data using 'l2' norm.

K-Means:

- We have implemented "K-Means" algorithm using two metrics for closeness. To explain about the algorithm.
- We first choose any random K points from the data set. These random points become the centroids of the cluster they are representing.
- Then we calculate the closeness of each data point from our data to each of the centroids of the cluster. The centroid to which this data is the closest one, it is assigned to that cluster. In this manner all data points are assigned to the cluster in this iteration.
- After this iteration when all the data points are assigned to the clusters, we recompute the mean of each cluster. This mean may not necessarily be an actual data point in the data and can be a hypothetical point but we treat this as our new centroid for the cluster.
- After assigning new centroids to the new cluster we reiterate the same process and compute the closeness of the data points to each of the centroids and assign it to the corresponding cluster.
- While doing so we also keep track of the old cluster and the iteration we are executing. If the newly formed cluster is similar to the older one if we can terminate the iteration as there are no more changes going to happen and our clusters are stable.
- We also keep track of the iteration number that we are doing because if the older cluster and the newly formed do not match then we cannot go on forever for finding a perfect match. Thus we terminate iterating when the max_Iteration is reached.

Pairwise Distance as Closeness Metric:

- We have used pairwise distance as one of our closeness determining metric.
- Thus when we calculate the similarity of the data points with any of the centroid we select the one with the minimum distance meaning we assign that data point to the closest cluster.

Cosine Similarity as Closeness Metric:

- We have used cosine similarity as one of our closeness determining metric.
- Thus when we calculate the similarity of the data points with any of the centroid we select the one with the maximum similarity meaning we assign that data point to the closest similar cluster.