**Mohana Krishna Narsupalli**

**HW3: K-means Clustering**

username:rook\_in\_defence

**Approach and Methodology:**

**For Iris Dataset: Score-0.72**

Initially, we are given 150 records of Iris Dataset with 4 attributes namely sepal length,sepal width, petal length and petal width. Initially I initialised centroids with random sample subset. Then for each row present in the records, in the first phase it computes the distance between each record of the input file to the centroid and if it is minimum to any of the centroid, then the record is assigned to that particular centroid in the initial phase. Similar procedure is followed for all the records in the input dataset.

The resultant list cluster\_assign contains the nearest centroid label and for all 10000 records, it holds labels for the nearest centroids.Then for each of the centroid, calculate the number of instances belonging to that cluster and get average of that cluster. Then update the centroids and whole process continues for 1000 iterations until the centroids converges better. Then in final iteration cluster\_assign contains the cluster label for that corresponding vector. It is the final list of cluster centers and that is saved to file.

Then I plotted the SSE errors for clusters ranging from 2 to 20 in range of 2.The following table shows the error rate(SSE) for various k values when run. Table

Description automatically generated Chart, scatter chart

Description automatically generated

k-value-SSE plot

It shows that as k value is increasing, the corresponding error rate is decreasing. It shows that as centroids are converging, the corresponding distance of instances to their respective centroids are decreasing. As this dataset is small and has less attributes, there is no need for dimensionality reduction. But for Digits Clustering, we need to do it.

**For Digits Clustering: Score- 0.72**

A picture containing graphical user interface

Description automatically generated

PCA n\_components

The above diagram represents the PCA for 28\*28 pixels for image dataset. I tried to explain variance for 95% using PCA andtried to get the number of components required for running PCA for dataset. In above diagram, I plotted for range in step of 20 and variance in range of 0.2 on y-axis. It showed the number of components required to run in PCA. It is ranging about 160 to 180. Finally, I used PCA with n\_components = 0.99.

After that I did tsne - which is also used for dimensionality reduction. In this, number of components can be in multiple of 2 and max of 4.So initially used number of components as 2.So my tsne function is with parameters as following: (n\_components=2, perplexity=30.0, early\_exaggeration=12.0, learning\_rate=200.0,n\_iter=1000, n\_iter\_without\_progress=300,min\_grad\_norm=1e07, metric='euclidean', init='random', verbose=0,random\_state=None, method='barnes\_hut', angle=0.5, n\_jobs=None).

It greatly reduces the dimensionality of the data and keeps only required features.Initially I have performed only PCA and continued to cluster. But it did not yield better results. Then I used this tsne dimensionality reduction and it improved vmeasure score a lot.There’s much more we can play with the parameters of tsne.

In the actual function, initially I initialised centroids with random sample subset. Then for each row present in the records, in the first phase it computes the distance between each record of the input file to the centroid and if it is minimum to any of the centroid, then the record is assigned to that particular centroid in the initial phase. Similar procedure is followed for all the records in the input dataset.

The resultant list cluster\_assign contains the nearest centroid label and for all 10000 records, it holds labels for the nearest centroids.Then for each of the centroid, calculate the number of instances belonging to that cluster and get average of that cluster. Then update the centroids and whole process continues for 2500 iterations until the centroids converges better. Then in final iteration cluster\_assign contains the cluster label for that corresponding vector. It is the final list of cluster centers and that is saved to file.

Then the same logic is implemented for clusters ranging from 2 to 20 in a step of 2.Then I calculated the sum of squared errors and plotted.

Plot of SSE Curve

Table

Description automatically generated Chart, scatter chart

Description automatically generated

For above plot I have run manually and checked the sum of minimum distances of each instance belonging to corresponding cluster and calculated the SSE values. Y-axis shows SSE values and x-axis shows k-value correspondingly.

References:

For preprocessing-Sklearn Documentation, For plotting -matplotlib,Numpy,pandas