Name: Uday Kumar Kamalapuram
GMU Id: G01340201
Miner User Id: uday
Public Score Part I: 0.95
Rank Part I: 6
Public Score Part II: 0.76
Rank Part II: 96

K-Mean Clustering:

Clustering is a technique for categorizing raw data and searching for hidden patterns in datasets. It is the process of grouping data objects into fragmented groupings so that data inside the same cluster is identical but data between clusters differs. k-Means Clustering is a clustering algorithm that divides a training set into different clusters of examples that are near each other. It is a prototype based, partition clustering technique that attempt to find the user specified number of clusters K in terms of centroid.

Part I:

The IRIS data set consists of 50 samples from each of three species of Iris. It consists of four features the length and the width of the sepals and petals. Based on the combination of these four features we need to cluster the data.

Approach:

Selecting the Notebook:

I have chosen Google colab to start the Assignment as I am familiar with Google colab and I can work on it from the cloud.

Data Uploading and Framing:

I have imported the given text files "Assignment4-irisData.txt" through upload from google.colab. I am reading the files through Pandas library read_csv() and there are 150 rows, with 4 features.

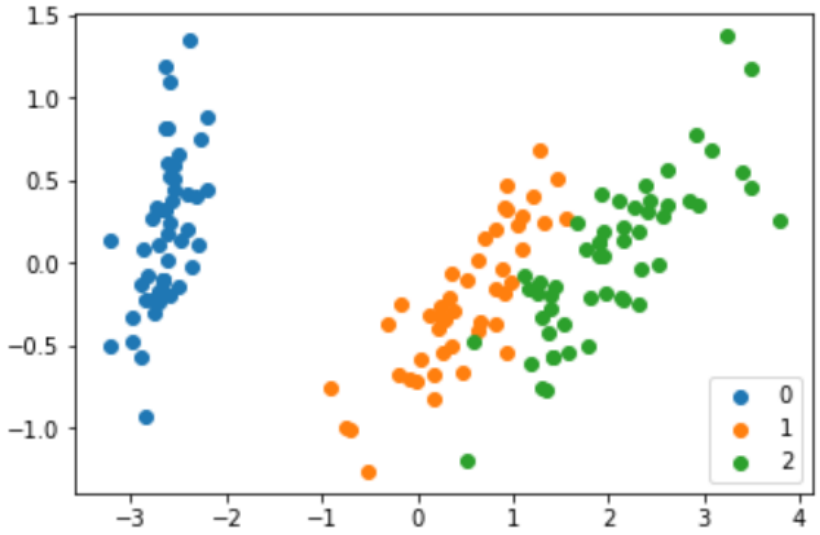
Building Model:

- I have chosen, K-initial random centroids, in this scenario k=3, for this I have used **np.random.choice** inbuilt method.
- After choosing random k-centriods, I am finding out the distance of the particular point from the centroid and I am using **cdist** method of **scipy.spatial.distance** library.
- While finding the distance of the point from the centroid using cdist, I have tried with **euclidean** and **cosine.**
- After this I am assigning each point to the closest centroid according to its distance.
- Next, I am updating the centroids by taking the mean of points in each cluster group.
- I am repeating above steps till centroid wouldn't change.

Pseudo Code of K-mean:

1. Select K random points as initial centroids.
2. Finding distance of points from centroid.
3. Assigning each point to closest centroid from K-clusters.
4. **Repeat**
5. Recompile the centroid of each cluster.
6. Assigning each point to closest centroid from K-clusters.
7. **Until** number of iterations. (Centroid don't change)

I have tried different iteration and I got optimal miner score at 75th iteration.
Below is the figure of the clustered IRIS data.



Part II:
The input data consists of 10,000 handwritten numerical images (0-9). Each digit of pixel is represented with an integer in the range of 0-255, with 0 representing white pixel and 255 representing completely black pixel. We will get total 28*28 pixel matrix for each digit. The 28X28 pixel data is flattened to 1x784 vector data.

Approach:

Data Uploading and Framing:
I have imported the given Assignment5-digitData txtfilr through upload from google.colab. I am reading the files through Pandas library read_csv() and there are 1000 rows, with 784 features.

Building Model:
I have used same **Pseudo Code of K-mean** used in Part I to implement the k-mean algorithm.

Data Pre-processing:
I have applied K-mean clustering on the image data without any pre-processing. I got less accuracy on the Miner. Later I reduced the dimensionality of the data to get the optimal score.

I have tried PCA to reduce the dimensionality of the image data. PCA (Principal Component Analysis) will reduce the dimensionality of large data sets. It transforms larger set of variables to smaller ones that contains most of the information in the original set.

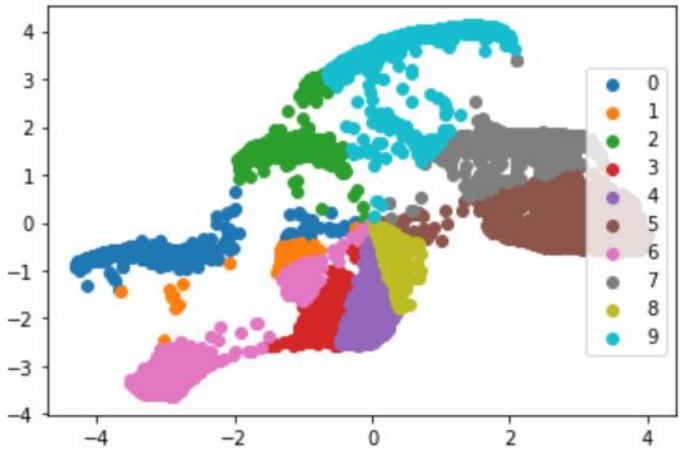
Assignment 5 – K-mean Clustering

I have used PCA inbuilt function of sklearn.decomposition library. I tried different n_components. I reduced the dimensionality/features to 75 from 784 by giving n_components=75 for which I got optimal solution.

I have also tried t-SNE technique to dimensional reduction. t-SNE is a t-Distributed Stochastic Neighbour Embedding method it is well suited to visualize high dimensional data. t-SNE minimizes the divergence between two distributions: a distribution that measures pairwise similarities of the input objects and a distribution that measures pairwise similarities of the corresponding low-dimensional points in the embedding.

I have used TSNE inbuilt method of sklearn.manifold library.

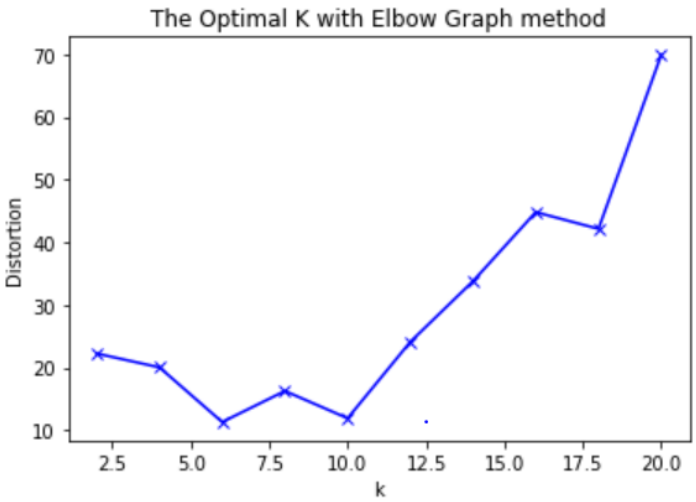
I got optimal miner score when I applied t-SNE dimensional reduction on PCA reduction data and also improved the time efficiency.



The following is the visualization form of clusters formed when we applied k-mean on pre-processed data with PCA and t-SNE.

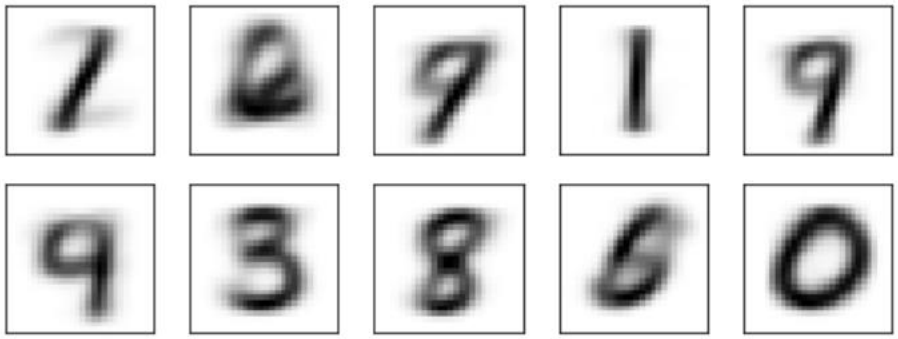
Observation and Analysis:

I calculated the Sum of Squared errors; I calculated the difference between predicted and observed value and squared and summed those and taken average and calculated this distortion for different clusters of k ranging from 2 to 20 and plotted the graph. The graph came in the elbow shape and distortion got flattened near to K=10 position.



The following is the graph with varying number of clusters K with Distortion.

The following is the 10 clustered digit images formed from the k-mean clstering with centroids.



Iterations:

The observation from the nuumber of iterations used is the miner score accuracy is good for 150 iterations, I got miner score of 0.76. Where as for same pre-processed data for 50 iterations I got 0.67 score.

Iterations	Miner Score
50	0.67
150	0.76

Conclusion:

In this Assignment I have implemented the K-mean clustering and using an Cosine cdist method I got an optimal soltion for IRIS data. I have cross verified the optimal number of clusters k=10 with Elbow Graph method. In part II, I got to know that the pre-processing of the given data is important to generate otimal clustering. Using feature reduction techniques like PCA and t-SNE in this project has sped up the process and enhanced the outcomes.

References:

https://www.askpython.com/python/examples/k-means-clustering-from-scratch

https://medium.com/analytics-vidhya/pca-vs-t-sne-17bcd882bf3d

https://towardsdatascience.com/understanding-k-means-clustering-in-machine-learning-6a6e67336aa1