CS584 HOMEWORK 5

NAME : Preeti Bhattacharya (G01302375)
Part I -**USERNAME IN MINER** : Preeti,
ACCURACY SCORE: 0.95,
RANK: 8 (At time of Upload)
Part II -**USERNAME IN MINER** : Preeti,
ACCURACY SCORE: 0.71,
RANK: 102 (At time of Upload)

Introduction:

Clustering is a technique for logically classifying raw data and searching for hidden patterns in datasets. It is the process of arranging data objects into fragmented clusters so that data in one cluster is identical to that in another, but data in other clusters differs. Clustering algorithms are frequently used in various application fields because to the necessity for organizing rapidly growing data sets and learning meaningful information from them.

Part I – As the famous Iris dataset serves as an easy benchmark for evaluation. We have tested our K-means Algorithm on the basis of sepal length in cm, sepal width in cm, petal length in cm, petal width in cm, and 150 instances.

Implementation of K-means Algorithm on IRIS data set:

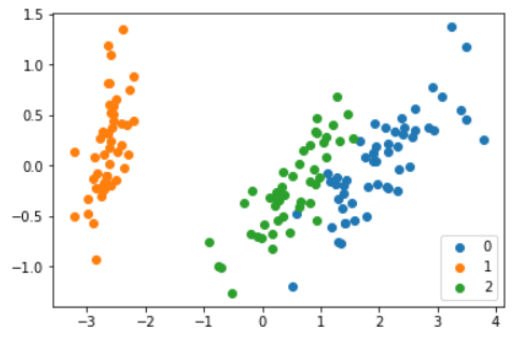
Step 1: I have selected k data points at random using np.random.choice to serve as our first Centroids.

Step 2: Then I have calculated the distance by using scipy.spatial.distance library between each data point in our training set using cosine function and the k centroids.

Step 3: Using the distance which I have calculated, I have assigned each data point to the closest centroid.

Step 4: Then I have taken the average of the points in each cluster group to update the centroid location.

Step 5: After I kept repeating steps 2–4 until our centroids remain unchanged.



This figure is of clustered iris data, where I got an optimal result on miner for my 75th iteration.

Pseudo code of K-means algorithm:

```
1) Initial centroid = k random points
2) Find distance of point from initial centroid.
3) From k-cluster assign each point to closest initial centroid.
4) Repeat
5)     Recompile centroid for each cluster
6)     Assign each point to closest centroid from k-cluster
7) Until number of iteration or centroid doesn’t change.
```

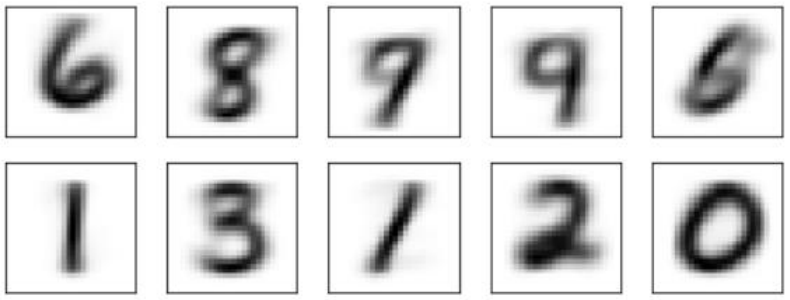
Part II- Now I have taken 10,000 handwritten images were digits (0-9) are written. Where for every digit, each pixel can be represented as an integer in the range [0, 255] where 0 corresponds to the pixel being completely white, and 255 corresponds to the pixel being completely black. This gives us a 28x28 matrix of integers for each digit. We can then flatten each matrix into a 1x784 vector.

I have used PCA (principal component analysis) before a clustering algorithm as it is believed that it improves the clustering results in practice (noise reduction).

Implementation of K-means Algorithm on handwritten data set:

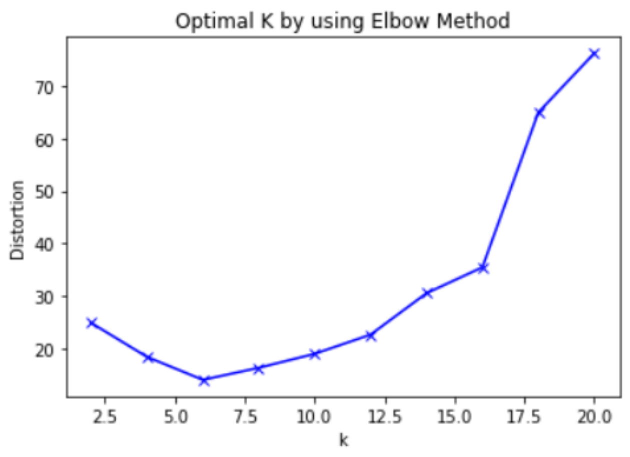
Step 1 – I have performed all the above 5 steps in IRIS data set.

Step 2 – After getting the points and centroids I have plotted the digit output by using k=10 where the result is 10 cluster in 784 Dimension.



Step 3 – I have selected the value of k by finding the difference between observed and predicted values and then by taking square and then summing all differences and by dividing summation by total value to obtain average.

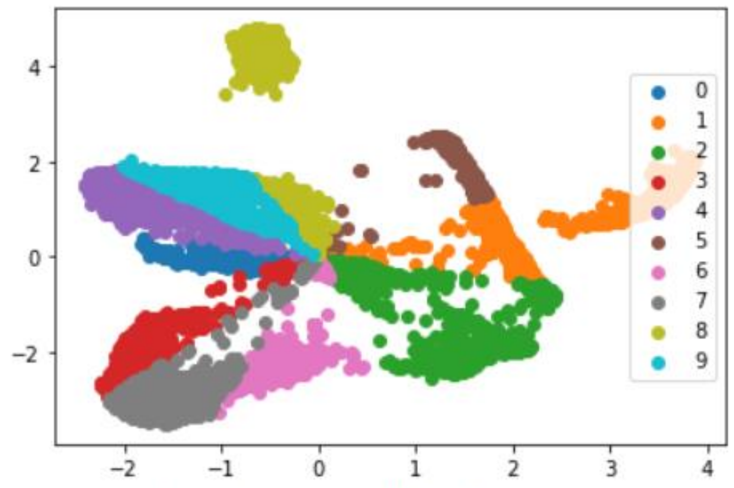
Step 4- I have computed sum of squared errors for different values of K from and plotted the elbow curve to select the value of K.



The graph shows the varying number of clusters k with respect to Distortion

Step 5 – Then I did Dimensionality reduction by using PCA and t-SNE.

- To yield the best clustering results I have utilized PCA to reduce the dimensionality from 784 to 72.
- Following PCA, I have used another dimensionality technique t-SNE, which minimizes the divergence between two distributions: one that measures pairwise similarities of the input objects and another that measures pairwise similarities of the corresponding low-dimensional points in the embedding.
- After the dimensionality reductions, K means was called with the updated data for a K value of 10, and the below output was shown to visualize cluster formations.
- By applying t-SNE dimensionality reduction I have obtained an optimized result on miner with improved time efficiency.



```
<Figure size 1440x1440 with 0 Axes>
```

CONCLUSION, INTERPREATION OF RESULT AND FUTURE WORK:

For the different number of iterations, we are getting different accuracy for example here for 50 iterations I got an accuracy of 0.71 for image data whereas when dealing with IRIS data I have got an accuracy of 0.95 on minor. From which we can say that IRIS data gives an optimal solution.

A lot of inferences can be drawn from the study of data collected after moments extraction. Images of the same iris with different orientations produced different eigenvalues and eccentricities. A change in image orientation, on the other hand, has only a minor impact on the values of rotation invariant moments, whereas raw and scale invariant moments are affected. The Euclidean distance of the moment vectors from the centroid is affected by changes in picture orientation. Despite this, due to coherence in scale invariant moments, the image has a high probability of being correctly categorized. In contrast to other strategies that need phase and size coherence, the model is resilient to changes in scale and rotation. The model may be enhanced much more by using a strategy that will process each image to provide uniform luminosity.

The algorithm does not guarantee that the global optimum will be reached. The outcome may be influenced by the original clusters. Because the algorithm is usually quick, it's normal to run it several times with different beginning conditions each time. However, in the worst-case scenario, performance can be slow: certain point sets, even in two dimensions, converge in exponential time. The fact that the smoothed running time of k-means is polynomial confirms that these point sets do not appear in practice.

References:-

https://en.wikipedia.org/wiki/K-means_clustering

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