PROJECT 2 : CLUSTERING ALGORITHMS

KAUSHIK RAMASUBRAMANIAN

TARIQ SIDDIQUI

JUNAID SHAIKH

Clustering Algorithms -

Clustering is a technique for finding groups of data points having similarity in their features, called clusters. There are two types of Clustering techniques:

1. **Unsupervised learning**
2. **Supervised learning**

Unsupervised learning is a type of clustering technique in which there are no prescribed labels in the data and no class values denoting the grouping of data instances that are given. Few types of Unsupervised learning algorithms are K-means, hierarchical clustering, DBSCAN.

Supervised learning is where you have input variables (x) and an output variable (Y) and you use an algorithm to learn the mapping function from the input to the output. Examples of Supervised learning are Support vector machines and Linear Regression.

In this project, we worked on three Unsupervised learning algorithms. They are as described below.

K-Means Algorithm –

OVERVIEW

It is a type of unsupervised learning, which is used when we have unlabeled data. This algorithm finds groups in the data, with the number of groups represented as a variable K. The algorithm works iteratively to assign each data point to one of K groups based on the features that are provided. Clustering takes place on the basis of a feature similarity (in our case being the Euclidean distance).

This algorithm results in the below two things:

1. The centroids of the K clusters, which can be used to label new data.
2. Labels of the training data.

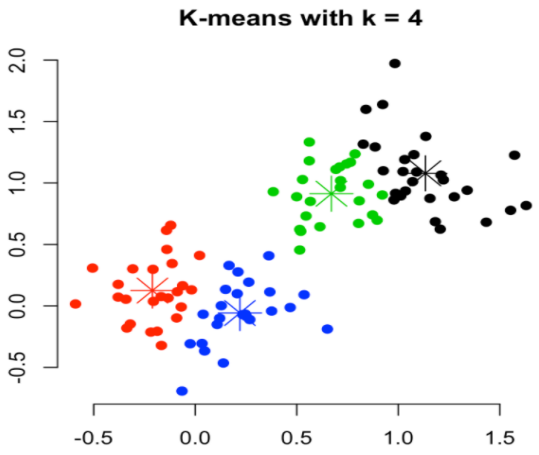


Figure 1: K-means pictorial representation

IMPLEMENTATION DETAILS

1. We select initial centroids based on requirements, and have used variety of methods like selecting random rows from the data as the initial centroids, or selecting the initial K rows, or selecting one row from each of the ground truth clusters provided in the file.
2. For each of the points in the file, calculate Euclidean distance from the selected centroids and assign the point to the cluster with the minimum distance from centroid.
3. For each of the clusters, new centroids are calculated by taking the mean of all points belonging to that cluster.
4. Once the centroids are updated, we check if the new centroids obtained are the same as the centroids in the previous step. This indicates that the cluster assignment is not changing any further and the centroids will not update further. The algorithm is stopped at this point.

ADVANTAGES OF K-Means

- K-Means has very intuitive approach and hence easy to implement
- If the clusters are big, then K-Means is, in most cases, faster for small K values.
- K-Means produces tighter clusters than Hierarchical clustering.
- Easy to interpret the clustering results.
- Clusters obtained tend to have similar density.

DISADVANTAGES OF K-Means

- Difficult to predict K value. Hence, different approaches such as squared errors of prediction(SSE) for regression are needed.
- It often produces uniform clusters with relatively uniform size even if the input data has different cluster size.
- Sensitive to outliers.

RESULTS FROM cho.txt and iyer.txt DATASETS

Given below are PCA plots for the datasets cho.txt and Iyer.txt after having executed K-means Algorithm.

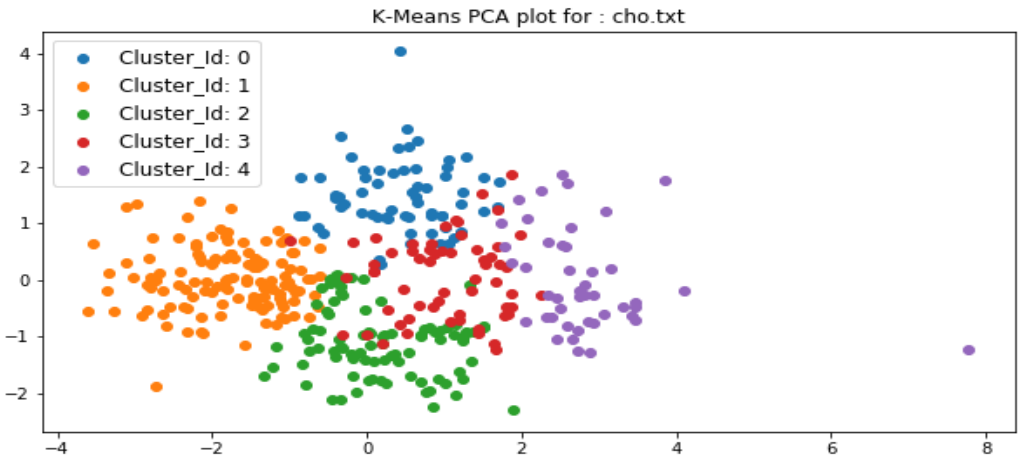


Figure 2: K-Means PCA plot for Cho.txt

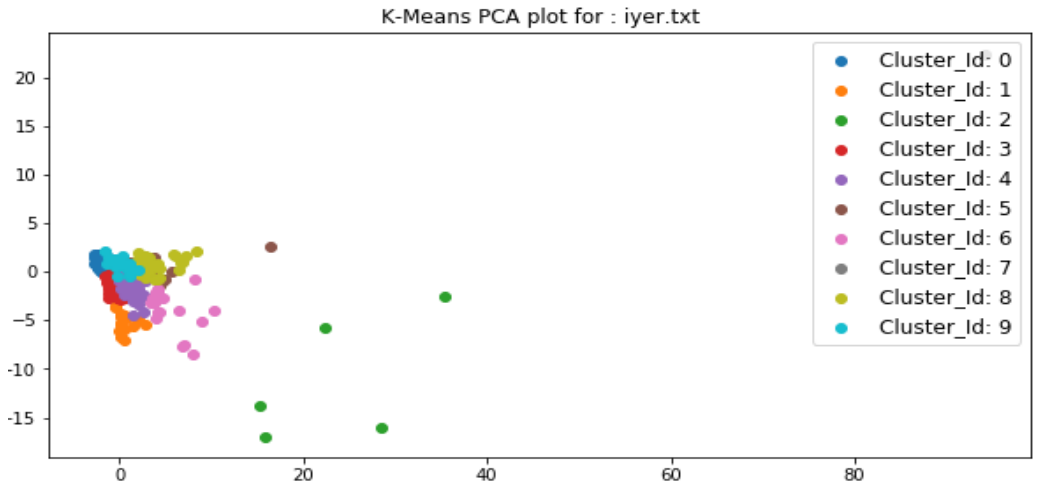


Figure 3: K-Means PCA plot for iyer.txt

Given below are the Jaccard Coefficients Rand Index for K-means algorithm for the two datasets cho.txt and Iyer.txt

Cho.txt – *Jaccard Coefficient* -> 0.37609587388401833, *Rand Index* -> 0.7917527987328519

Iyer.txt – *Jaccard Coefficient* -> 0.33329534554640683, *Rand Index* -> 0.7373554467262027

Hadoop MapReduce K-means :

We used open-source Apache Hadoop software framework for our given data processing using the MapReduce programming model to achieve K-Means clustering.

Under this approach Given input data is splitted across number of Mappers, that is, job of finding distance of a Input row from all the clusters is splitted across number of Mappers. These Mappers then operate individually and then find the closest cluster using Euclidean distance.

Mapper had default block size of 128MB, that is, this is the granularity level of file size at which Mapper operates.

Optimization:

1> As part of optimization, we modified default block in hdfs-site.xml to 1MB(minimum block size allowed by hadoop). This allowed to split given data set into blocks of 1MB so as to increase number of Mappers. However, there was no gain as size of given input data is in very Kilo Bytes.

2> Also, we modified **"dfs.replication"** factor to 1. This helped to conserve memory. We limited replication as in implementation like ours we are constantly monitoring Map and also is very little possibility of failure which can be instantly cross-checked through monitoring.

3> We used wrapper classes LongWritable and IntWritable for data as these are easy to parse as compared to Text.

4> Parallel K-Means gives better performance than non-parallel K-Means as multiple reducers(example 5 reducers for cluster of size of 5) simultaneously calculate means of data points in different clusters.

5> As expected, external indexes(Jaccard/Rank) and number of iterations to converge values with MapReduce K-Means model are same as non-parallel K-Means.

Results:

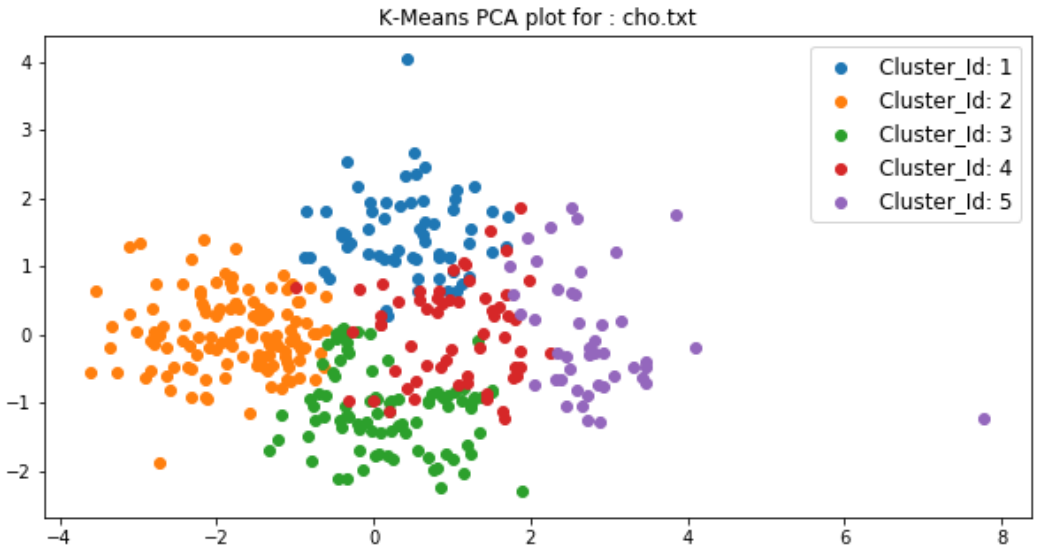


Figure 4: K-Means PCA plot for cho.txt using MapReduce

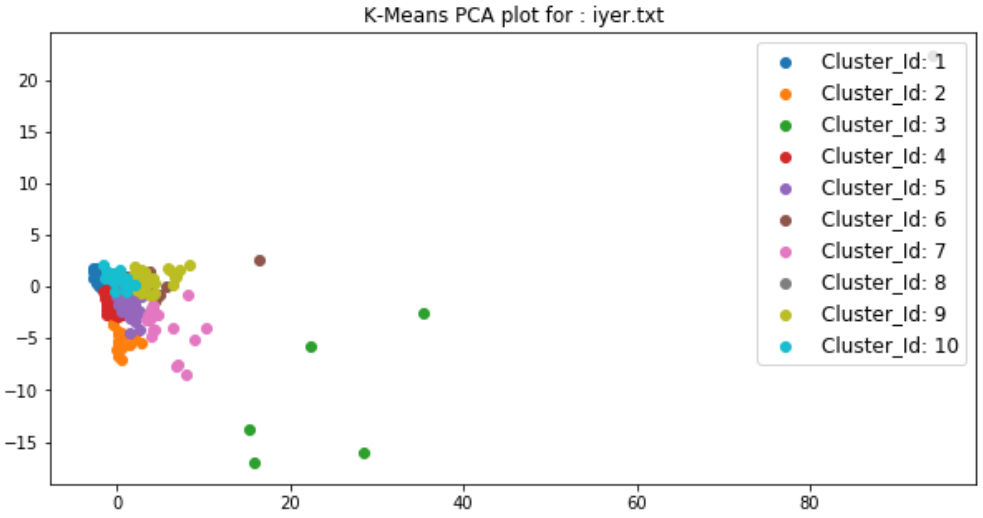


Figure 5K-Means PCA plot for iyer.txt using MapReduce

Given below are the Jaccard Coefficients Rand Index for K-means algorithm for the two datasets cho.txt and Iyer.txt using MapReduce

Cho.txt – Jaccard Coefficient: 0.37609587388401833, Rand Index: 0.7917527987328519, Number of Iterations: 23

Iyer.txt – Jaccard: 0.33329534554640683, Rand: 0.7373554467262027, Number of Iterations: 29

Hierarchical Clustering Algorithm –

OVERVIEW

Hierarchical clustering involves creating clusters predefined ordering from top to bottom. For example, all the files and folders on the hard disk are organized in a hierarchy. There are two types of hierarchical clustering, *Divisive* and *Agglomerative.*

Agglomerative (i.e bottom-up):

- Start with all points in their own group.
- Until there is only one cluster. Do this repeatedly: merge the two groups that have the smallest dissimilarity.

Divisive (i.e top-down):

- Start with all points in one cluster.
- Do this repeatedly, until all points are in their own cluster. Split the group into two resulting in the biggest dissimilarity.

We have implemented the Agglomerative approach in this project.

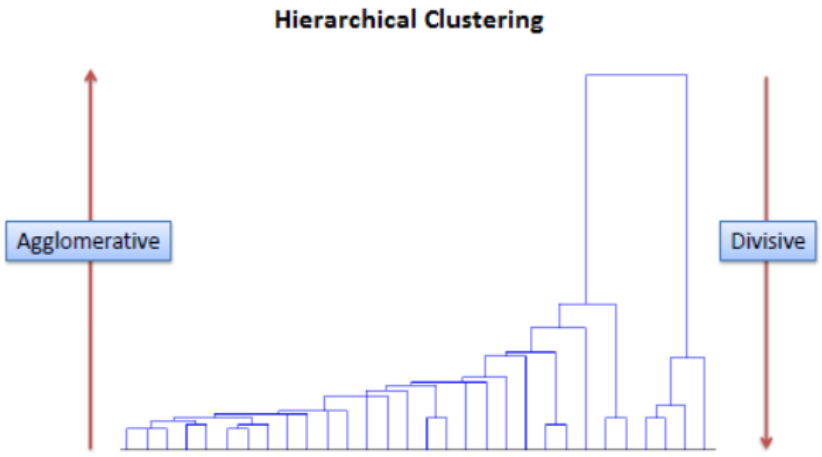
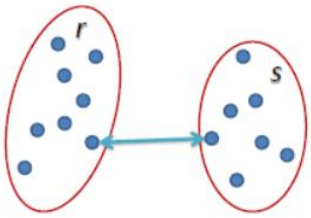


Figure 6: Hierarchical Clustering pictorial representation

Prior to applying clustering, it is required to determine the proximity matrix containing the distance between each point using a distance function. Then, the matrix is updated to display between the distance between each cluster. The following three methods differ in how the distance between each cluster is measured.

Single Linkage –

In single linkage hierarchical clustering, the dissimilarity between r,s is the smallest dissimilarity between two points in opposite groups.

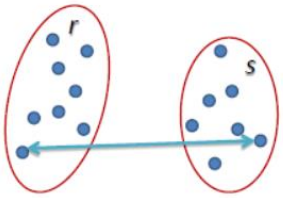


$$L(r,s) = \min(D(x_{ri}, x_{sj}))$$

Figure 7: Single Linkage Hierarchical clustering

Complete Linkage –

In complete linkage hierarchical clustering, the dissimilarity between r,s is the largest dissimilarity between two points in opposite groups.

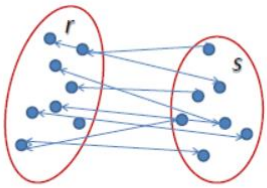


$$L(r,s) = \max(D(x_{ri}, x_{sj}))$$

Figure 8: Complete Linkage hierarchical clustering

Average Linkage –

In average linkage hierarchical clustering, the dissimilarity between r,s is the average dissimilarity over all points in opposite groups.



$$L(r,s) = \frac{1}{n_r n_s} \sum_{i=1}^{n_r} \sum_{j=1}^{n_s} D(x_{ri}, x_{sj})$$

Figure 9: Average Linkage Hierarchical clustering

Drawbacks of single and complete linkage –

Single and complete linkage can have some practical problems:

- Single linkage suffers from chaining. In order to merge two groups, only need one pair of points to be close, irrespective of all others. Therefore, the clusters can be too spread and not compact enough.
- Complete linkage avoids chaining, but suffers from crowding. Because, its score is based on the worst-case dissimilarity between pairs, a point can be closer to points in other clusters than to points in its own cluster. Clusters are compact, but not far enough apart.

Average linkage tries to strike a balance. It uses average pairwise dissimilarity, so clusters tend to be relatively compact and relatively far apart.

IMPLEMENTATION–

1. Initially, all the points in the dataset are considered as clusters
2. We create a distance matrix containing distances between all the clusters created in the Step 1
3. Search for the two clusters with the minimum distance in the matrix
4. These two clusters will now be merged. All the elements from the clusters are put into one single cluster
5. The distance matrix is updated to reflect the distances between the newly formed cluster and all the other clusters in the matrix
6. The entries for both the individual clusters are deleted from the distance matrix now that they have been merged and a single row + column entry in the distance matrix is formed for the combined cluster
7. The algorithm is repeated till the distance matrix and the number of clusters converges to the number of clusters desired. Continuing the algorithm further results in the convergence of the clusters to a single cluster at the end and the dendogram can be cut at the desired level to find out the cluster assignment. However, we stop when the algorithm has converged to the desired number of clusters.

ADVANTAGES OF HIERARCHICAL AGGLOMERATIVE CLUSTERING –

- No apriori information about clusters required.
- Easy to implement and gives best result in some cases.

DISADVANTAGES OF HIERARCHICAL AGGLOMERATIVE CLUSTERING –

- Sometimes it is difficult to identify the correct number of clusters by the dendogram.
- No objective function is directly minimized.
- Time complexity of at least $O(n^2 \log n)$ is required, where 'n' is the number of data points.
- Sensitivity to noise and clusters.
- Breaking large clusters
- Difficulty handling different sized clusters and convex shapes.

RESULTS FROM cho.txt and iyer.txt DATASETS –

Given below are PCA plots for the datasets cho.txt and Iyer.txt after having executed hierarchical agglomerative clustering Algorithm.

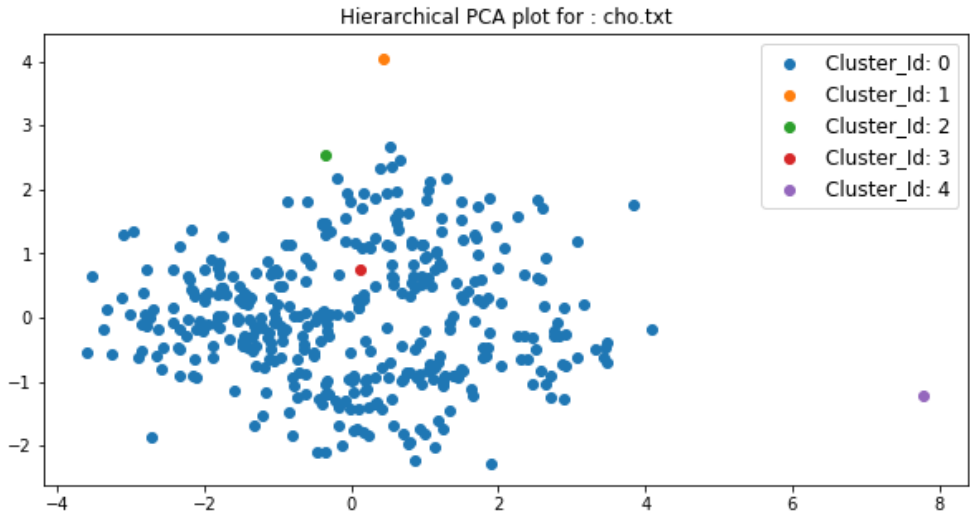


Figure 10: Hierarchical Clustering PCA plot for cho.txt for 5 clusters

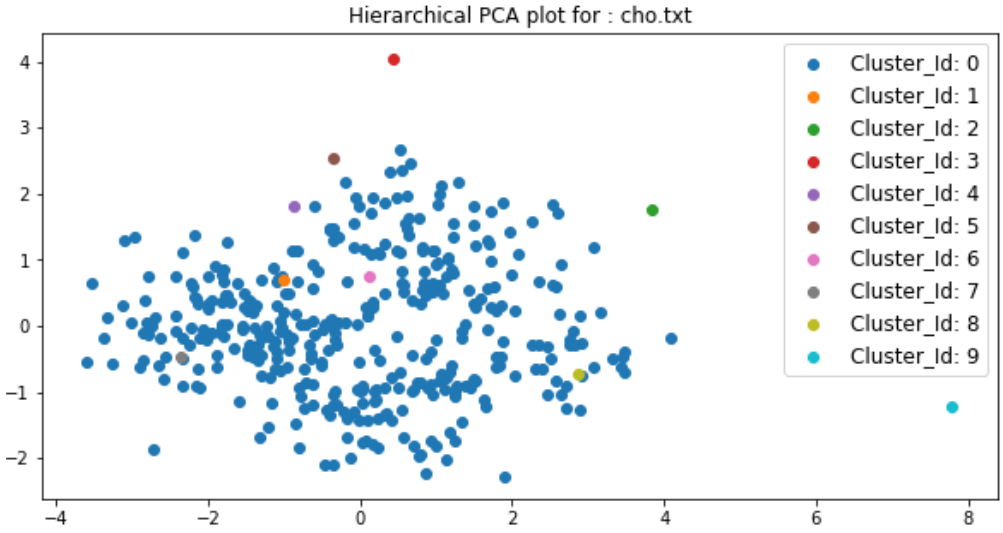


Figure 11: Hierarchical Clustering PCA plot for cho.txt for 10 clusters

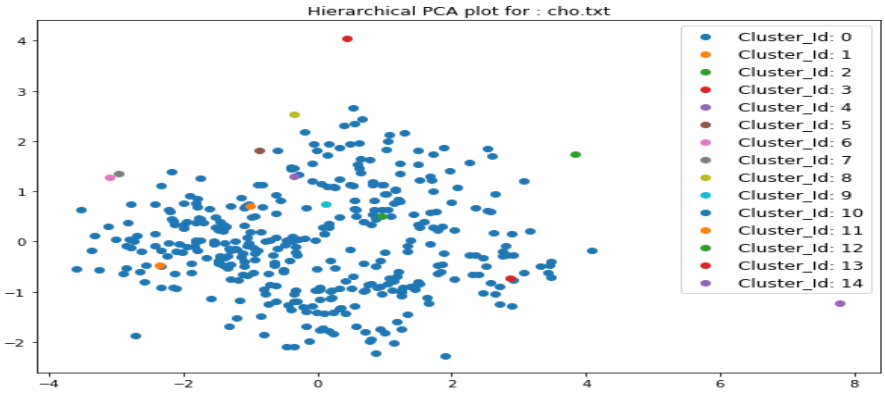


Figure 12: Hierarchical Clustering PCA plot for cho.txt for 15 clusters

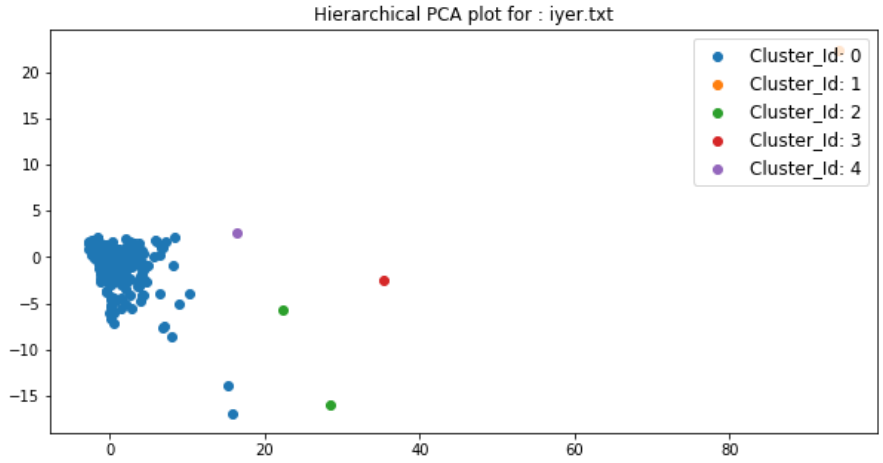


Figure 13: Hierarchical Clustering PCA plot for iyer.txt for 5 clusters

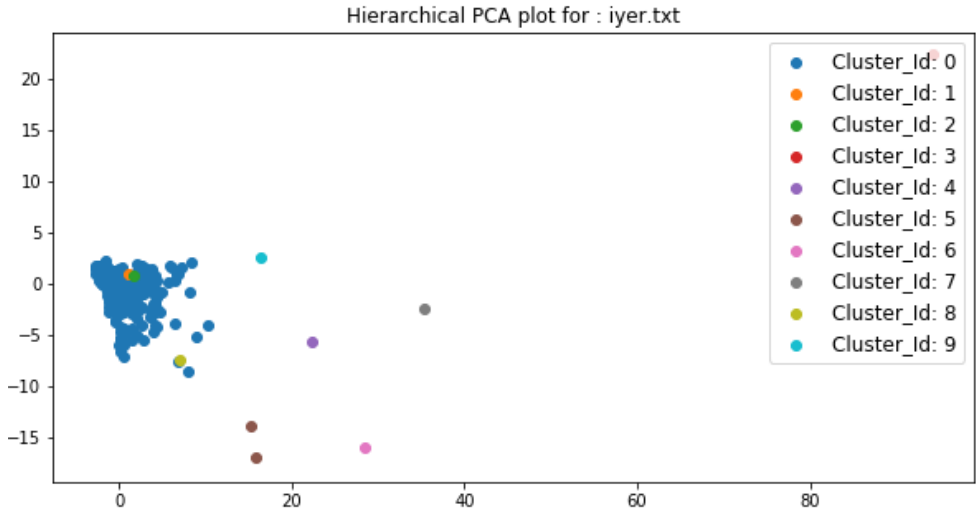


Figure 14: Hierarchical Clustering PCA plot for iyer.txt for 10 clusters

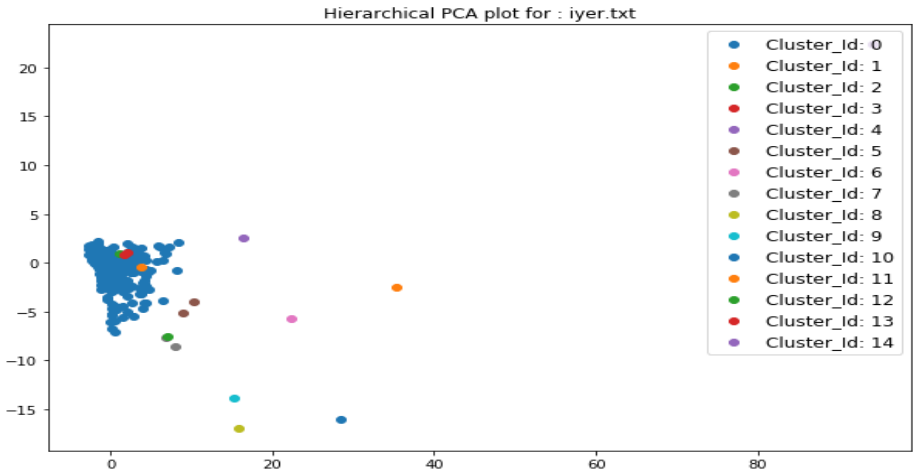


Figure 15: Hierarchical Clustering PCA plot for iyer.txt for 15 clusters

Given below are the Jaccard Coefficients, Rand Index for hierarchical clustering algorithm for the two datasets cho.txt and Iyer.txt

Cho.txt –

K = 5, *Jaccard Coefficient* -> 0.22839497757358454, *Rand Index* -> 0. 24027490670890495

K = 10, *Jaccard Coefficient* -> 0. 22778520126941706, *Rand Index* -> 0. 25529544417299793

K = 15, *Jaccard Coefficient* -> 0. 22503156968742463, *Rand Index* -> 0. 266839378238342

Iyer.txt –

K = 5, *Jaccard Coefficient* -> 0. 15647222380925174, *Rand Index* -> 0. 17144364339722173

K = 10, *Jaccard Coefficient* -> 0. 15824309696642858, *Rand Index* -> 0. 1882868355974245

K = 15, *Jaccard Coefficient* -> 0. 15945154219073931, *Rand Index* -> 0. 20667891308658418

DBSCAN Algorithm (Density-based spatial clustering of applications with noise)–

OVERVIEW

Unlike the previous two Unsupervised learning algorithms, this algorithm, for a given set of points in some space, groups together points that are closely packed together (points with many nearby neighbors), marking as outliers points that lie alone in low-density regions (whose nearest neighbors are too far away).

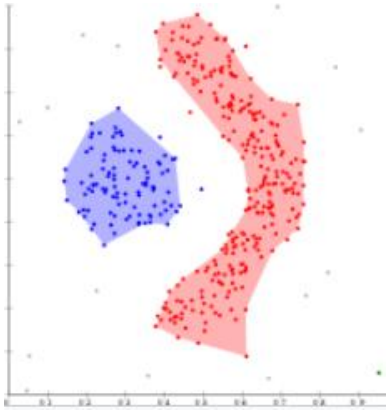


Figure 16: DBSCAN pictorial representation

ADVANTAGES OF HIERARCHICAL AGGLOMERATIVE CLUSTERING –

- This algorithm doesn't require one to specify the number of clusters in the data a priori, as opposed to k-means.
- It is robust to outliers, as it has a notion to noise.
- It is designed for use with databases that can accelerate region queries.
- It can find arbitrarily shaped clusters.
- The parameters MinPts and ε can be set by a domain expert, if the data is well understood.

DISADVANTAGES OF HIERARCHICAL AGGLOMERATIVE CLUSTERING –

- DBSCAN is not entirely deterministic: border points that are reachable from more than one cluster can be part of either cluster, depending on the order the data is processed.
- If the data and scale are not well understood, choosing a meaningful distance threshold ε can be difficult.
- DBSCAN cannot cluster data sets well with large differences in densities, since the minPts-ε combination cannot then be chosen appropriately for all clusters.

RESULTS FROM cho.txt and iyer.txt DATASETS –

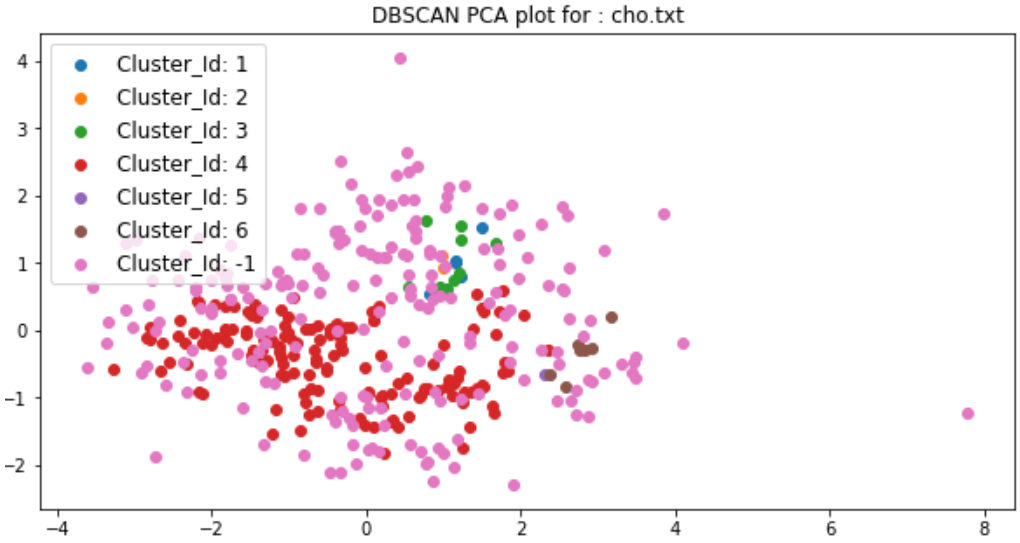


Figure 17: DBSCAN PCA plot for cho.txt -- minpts=4 and ε=1.03

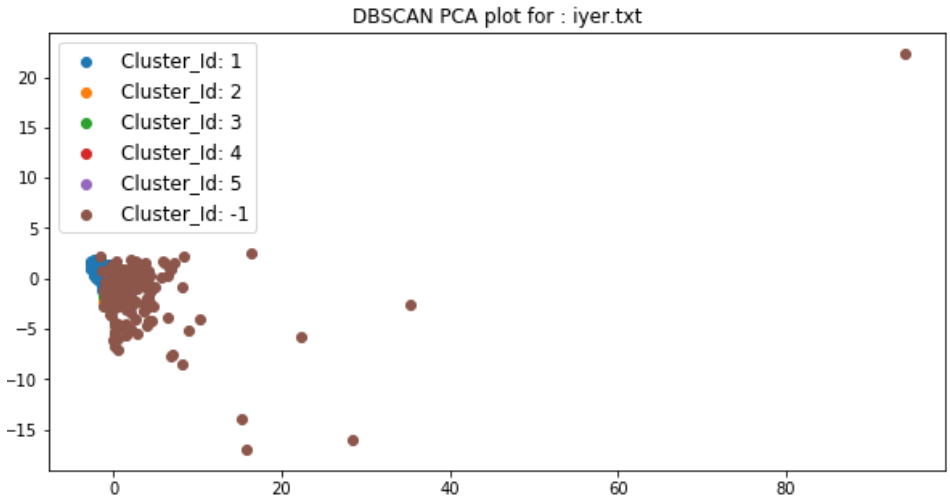


Figure 18: Figure 15: DBSCAN PCA plot for iyer.txt -- minpts=4 and ε=1.03

For Cho.txt = > external paramters : Jaccard : 0.2030359633981983 , rand : 0.5475583237133883

For iyer.txt => external paramters : Jaccard : 0.2840689352326217 , rand : 0.6523238891237574

COMPARISON OF PERFORMANCE OF THE ABOVE THREE ALGORITHMS:

TIME COMPLEXITY –

K-Means clustering is linear in the number of data objects i.e *O(n)* , 'n' being the number of data objects which is better when compared to that of Hierarchical clustering which is quadratic i.e $O(n^2)$. DBSCAN is considerably faster when compared to the other two algorithms as it doesn't have many iterations unlike the other two algorithms.

EXTERNAL INDEX –

Jaccard coefficient compares the members of two sets to see which members are shared and which are distinct. Therefore, the higher the percentage the more similar are the two datasets.

In the case of DBSCAN the lower value of minpts and higher ε gives higher Jaccard coefficant.

For hierarchical clustering, for k=5,10 and 15 the Jaccard coefficient values don't vary much.

For K-Means, the higher the K value, lower is the Jaccard Coefficiant.

Rand Index is also a measure of similarity between two data clusterings.

In the case of DBSCAN, the lower value of minpts and higher ε gives lower Rand index.

For hierarchical clustering, for k=5,10 and 15 the Rand index values don't vary much.

For K-Means, the higher the K value, lower is the Rand Index.

SHAPE OF CLUSTERS –

K-Means works well when shape of clusters are hyper-spherical (or circular in 2 dimensions). If If the natural clusters occurring in the dataset are non-spherical then probably K-means is not a good choice.

REPEATABILITY –

K-means starts with a random choice of cluster centers, therefore it may yield different clustering results on different runs of the algorithm. Thus, the results may not be repeatable and lack consistency. However, with hierarchical clustering, you will most definitely get the same clustering results.