CSE 601: Data Mining and Bioinformatics

Fall 2018

Submitted by: Tanushri Nayak, Chandani Jaiswal, Sai Kalyan Katta
UB Person Numbers: 50286751, 50288694, 50292522

PART1: Clustering Algorithms

K-Means:

- The implementation of K-Means starts with importing the dataset and generating the random centroids or assigning the user input as the centroids which can be seen in the function **populateData**.
- Then we calculate the distance between each data points using function **getEuclideanDistance** which calculates the distance between two data points as given below.

$$ED = \sqrt{\sum_{i=1}^{n}(qi - pi)^2}$$

- We create initial clusters with the centroids using the function **createCluster** and find the points which are at minimum distance from these existing centroids and add them to the cluster using **updateClusters**.
- Now compute the new centroid of the cluster using the function **updateCentoids** which is basically the mean of all the points in that cluster.
- At last we come to the termination where we check whether the centroids are changing or not. Once it is clear that there is no change in the centroids, we terminate the computation which is done by the function **checkTermination**.

Advantages:

- K-Means clustering is easy to implement Easy to implement.
- With a large number of variables, K-Means may be computationally faster than hierarchical clustering (if K is small).
- K-Means may produce higher clusters than hierarchical clustering
- An instance can change cluster (move to another cluster) when the centroids are recomputed.

Disadvantages:

- Difficult to predict the number of clusters (K-Value)
- The order of the data has strong impact on the final results.
- Sensitive to scale and rescaling dataset will completely change results.

Hierarchical Agglomerative Clustering:

- Hierarchical Agglomerative clustering with SingleLink(Min) starts with the computation of the distance matrix using all the data points from the gene dataset.
- In the next step, we find the least value from the distance matrix and add the corresponding data points to a cluster using the function **distanceMatrix**.

- Now, we compute the updated distance matrix by merging the data points from the previous cluster into one.
- The distance from the cluster to a point is defined as the minimum value of all the corresponding distances between the points in cluster to that point. The same holds good for the distance between two clusters.
- The same steps are repeated again until all the points are assigned to the clusters.
- Finally, a complete cluster is formed which is a combination of all the individual clusters that were formed earlier.

Advantages:

- Hierarchical clustering outputs a hierarchy in a structural form which is more informative than the unstructured set of flat clusters returned by k-mean. Therefore, it is easier to decide on the number of cluster by looking at the dendogram.
- Hierarchical clustering is easy to implement.

Disadvantages:

- It is not possible to undo the previous step i.e. once the instance has been assign to a cluster, they can no longer be moved around.
- Higher time complexity hence not suitable for large datasets.
- The order of the data has impact on the final results.
- Hierarchical clustering is very sensitive to outliers.

Density Based Clustering:

- The density based clustering basically involves two hyper-parameters namely the **ε** value and the **minpts** value.
- In the algorithm, we traverse through the unvisited points in the dataset and find their neighborhood points.
- The neighborhood points of point A are defined as the points that are at a distance <= ε from the data point A.
- If there are less neighborhood points than the **minpts** value, the point is marked as a noise.
- If the point clears the criterion of the **minpts** it is added to the cluster and the points in its neighborhood are taken and the same procedure is done to check whether they are noise points or not. If the neighborhood points clear the criterion of the **minpts**, they are added to their parent data point's cluster.
- These steps are repeated until all the data points are visited and added to cluster or marked as the noise.

Advantages:

- Density-based clustering algorithm can handle clusters of different shapes and sizes and play a vital role in finding non-linear shapes structure based on the density.
- DBSCAN is good for data sets with large amounts of noise as it is resistance to noise.

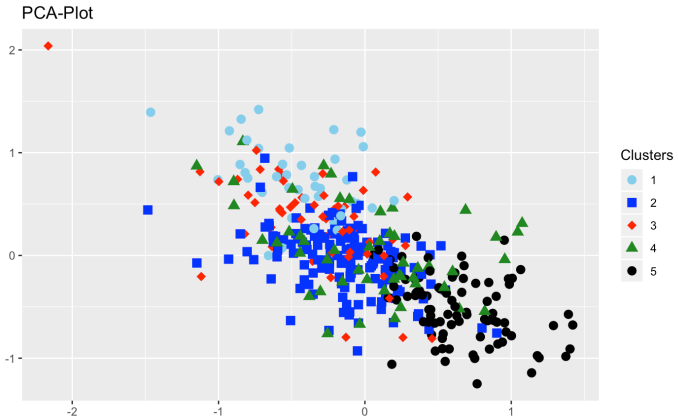
Disadvantages:

- Input parameters may be difficult to determine.

- In some situations, very sensitive to input parameter setting as they are hard to determine the correct set of parameters.
- It cannot handle varying densities of data.

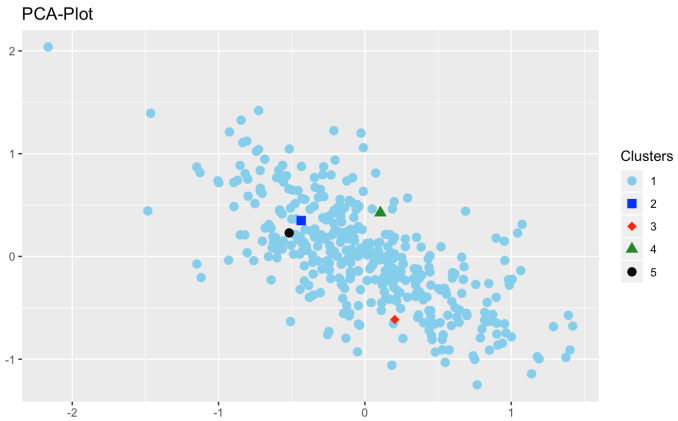
Results:
For the dataset cho.txt

K-Means:
For number of clusters - 5
Rand Index= 0.78465194
Jaccard co-efficient= 0.33566606



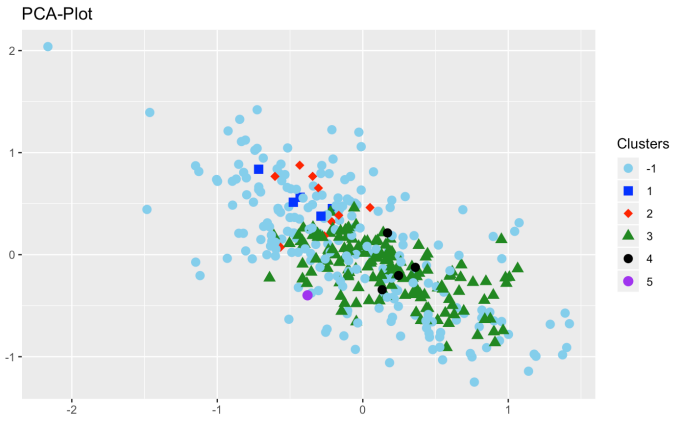
K-Means

Hierarchical Agglomerative Clustering:
For number of clusters - 5
Rand Index=0.2402749
Jaccard co-efficient= 0.22839497



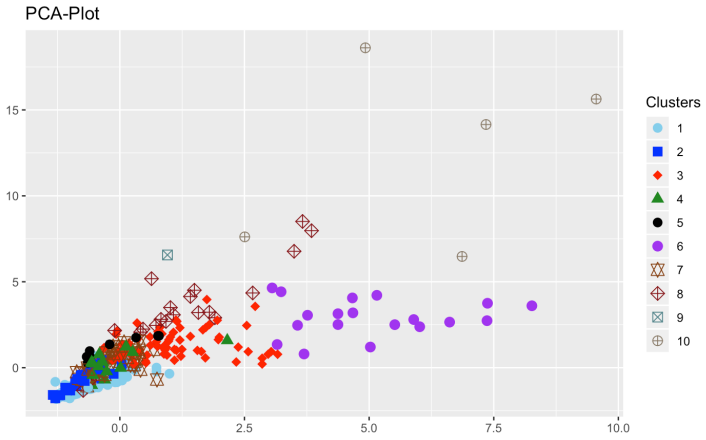
Hierarchical Agglomerative Clustering

Density Based Clustering:
Epsilon - 1.03, minpts - 4
Rand Index = 0.5385379
Jaccard co-efficient = 0.2033647



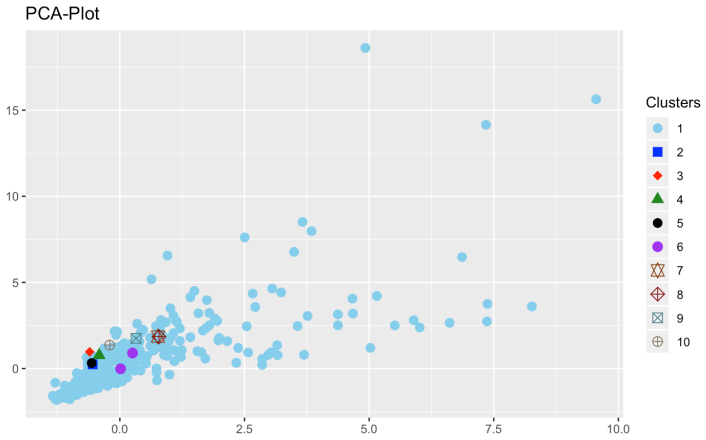
Density Based Clustering

K-Means:
For number of clusters = 10
Rand Index=0.75050974
Jaccard co-efficient= 0.34385484



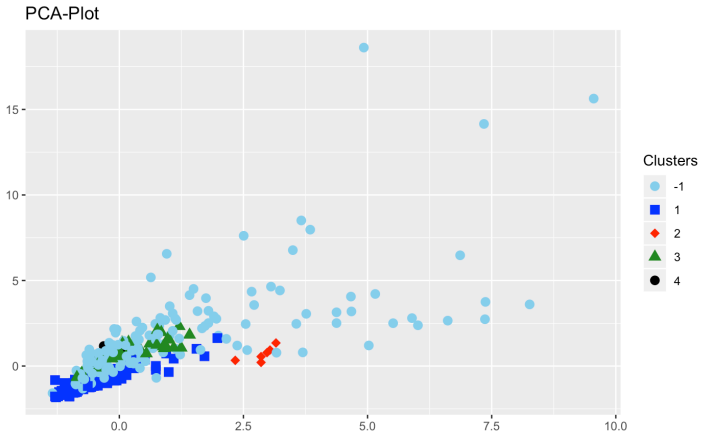
K-Means

Hierarchical Agglomerative Clustering:
For number of clusters = 10
Rand Index= 0.18828684
Jaccard co-efficient= 0.15824309



Hierarchical Agglomerative Clustering

Density Based Clustering:
Epsilon - 1.03, minpts - 4
Rand Index = 0.64998186
Jaccard co-efficient = 0.28350756



Density Based Clustering

Comparison:

Jaccard coefficient and the Rand Index compare the Clustering labels against the GroundTruth to see which members are similar and which are distinct. Similarity is higher when the values of Jaccard coefficient and the Rand Index are near to 1.

For K-Means,

- For higher values of K, both the rand Index and the Jaccard Coefficiant are low.

For Density Based Clustering,

- For lower values of minpts and higher values of ε, Rand Index and Jaccard coefficant are high and low respectively.

For hierarchical clustering,

- For different number of clusters, the Rand Index and Jaccard coefficient values don't vary much.

PART2: Single-node Hadoop cluster/MapReduce K- means

Hadoop MapReduce K-means:
The given input data set is split across number of Mappers which calculates distance of a Input row from all the clusters; the job is split across number of Mappers which operate individually to detect the closest cluster using Euclidean distance.
Reducers sum all the samples and compute the total number of samples assigned to the same cluster so that we can get the new centres used for next iteration.

Mapper:
Input: <key,value> pairs, where each <key,value> pair represents a record.
key is the offset in bytes of the record.
The value is a string of the contents of the record.
Output: <key1,value1>.
key1 is the centroid to which the data point belongs to.
Value1 is the data point.

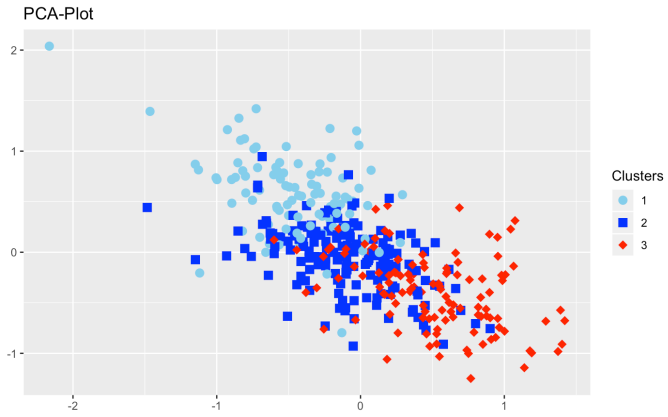
Reducer:
Input: <key1,V>.
key is any centroid.
V is the list of data points having the centroid 'key1'
Output: <key2, value2>
key2 is the new centroid.
value 2 is the string comprising of all the data points with this new centroid

To improve performance :

- We modified default block in hdfs-site.xml to 1MB(minimum block size allowed by hadoop). Note : Mapper has default block size of 128MB.This allowed to split given data set into blocks of 1MB in order to increase the number of Mappers. However, there was no significant change.
- We used wrapper classes IntWritable for data as these are easy to parse as compared to Text.
- 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.

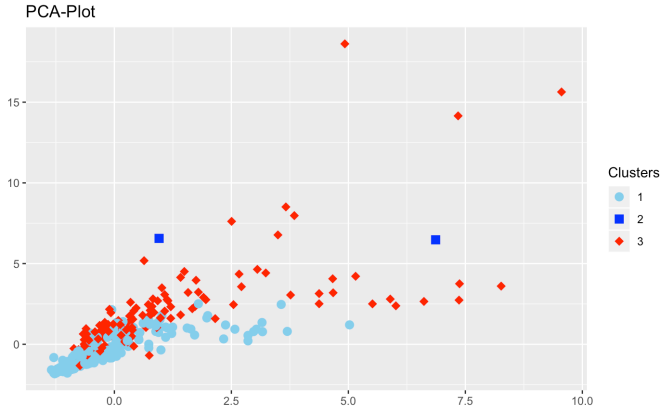
Results:
For the dataset cho.txt

Rand Index=0.7574028832988805
Jaccard co-efficient= 0.4088574886337618



For the dataset iyer.txt

Rand Index=0.4995304707638549
Jaccard co-efficient= 0.2199408701534227



We observed that the results of the non-parallel K-Means and parallel K-Means are very similar. However, for the given datasets, MapReduce implementation of K -Means took longer time than the non-parallel K-means which leads to our conclusion that MapReduce is efficient only for large datasets