CSE-601
DATA MINING AND BIOINFORMATICS

PROJECT 2: CLUSTERING ALGORITHMS

By:

Naina Nigam : 50208030

Surabhi Singh : 50208675

Vanshika Nigam : 50208031

PART 1: Clustering Algorithms

1. K-means

k-means is one of the simplest unsupervised learning algorithms that solve the well known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed apriori.

Following k-means algorithm has been implemented:

1. Partition the gene ids $\{x_1, x_2, \ldots, x_n\}$ into k clusters, where k is predefined.
2. Specify the initial cluster centers i.e. centroids.
3. For each gene id x_i
   - Calculate the distances between x_i and the k centroids.
   - Re-assign x_i to the cluster whose centroid is the closest to x_i.
4. Update the cluster centroids based on the current assignment.
5. Repeat steps 3 and 4 until cluster centroids do not change.

Advantages:

- Ease of implementation and high-speed performance.
- Gives best result when data set are distinct and well separated from each other.

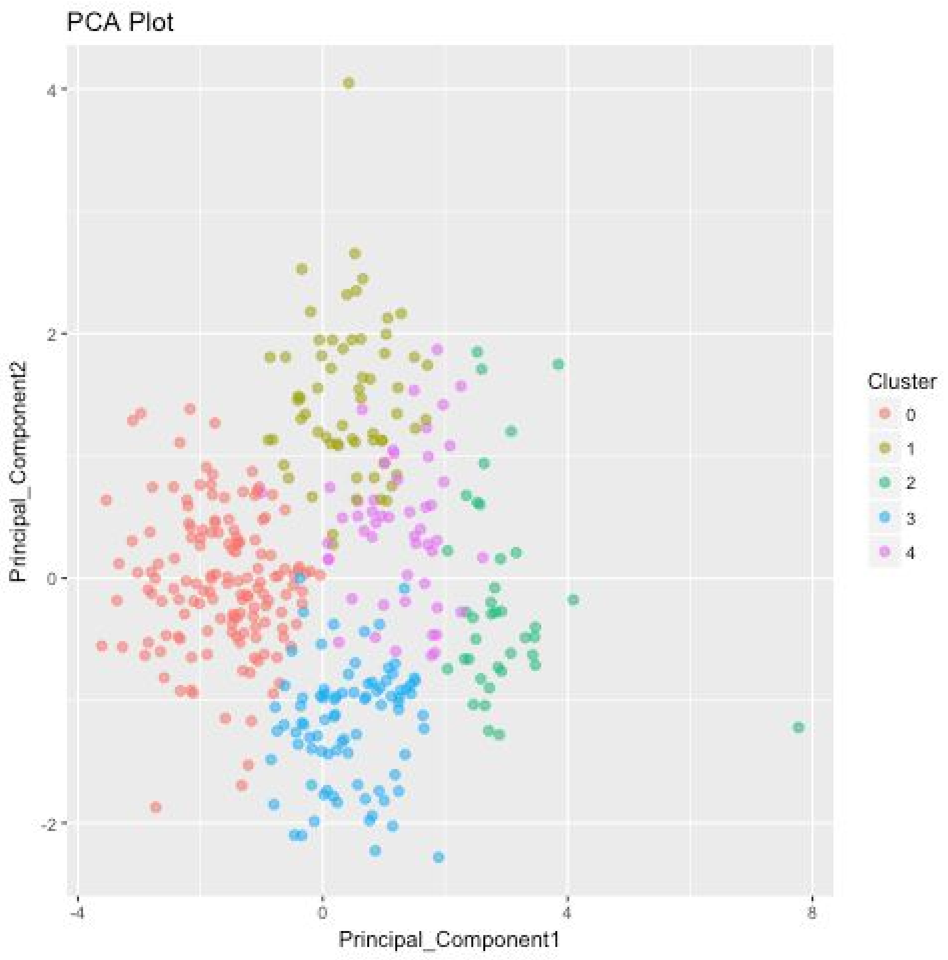
Disadvantages:

- Selection of optimal number of clusters is difficult.
- Selection of initial centroids is random.
- Unable to handle noisy data and outliers.
- Applicable only when mean is defined i.e. fails for categorical data.

Code Results :

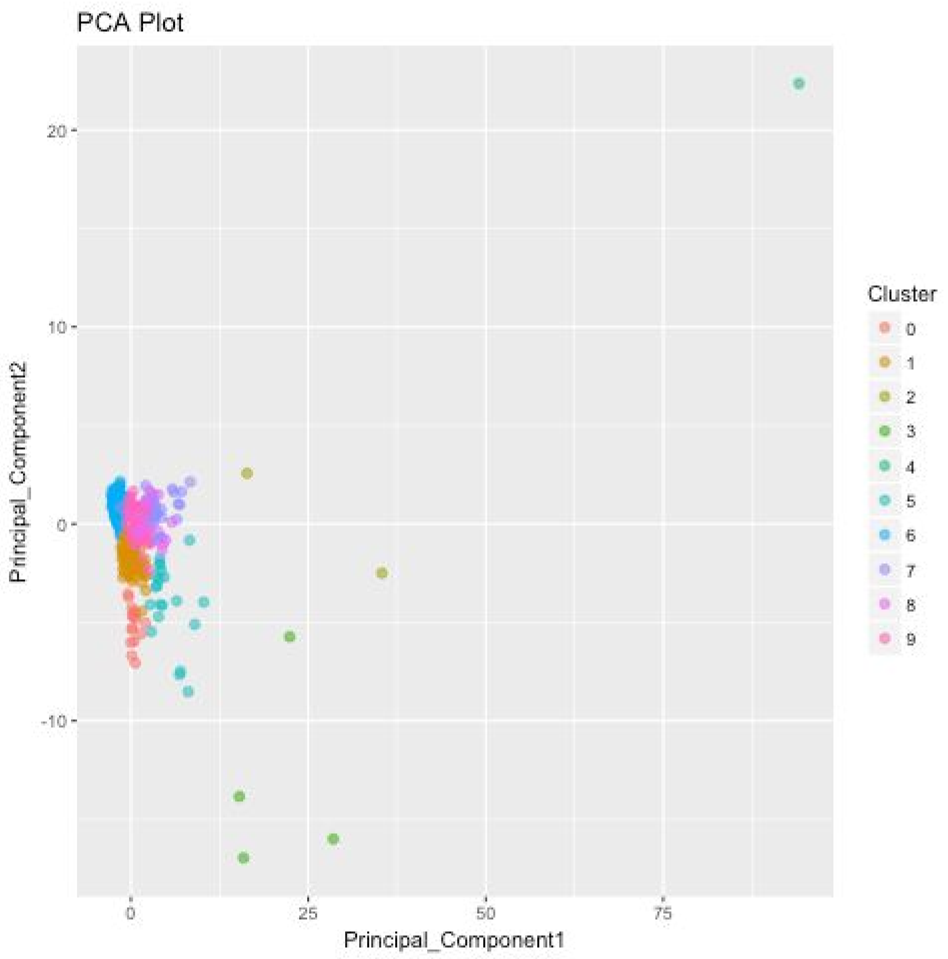
- File Name: cho.txt
- Number of Clusters(K) :5
- Initial Cluster Centroid Ids: [108, 9, 6, 376, 379]
- Rand Coefficient is: 0.802182608929
- Jaccard Coefficient is: 0.41563900234

<u>Plot for cho.txt for 5 Clusters</u>



- File Name: iyer.txt
- Number of Clusters : 10
- Initial Cluster Centroid Ids: [309 97 501 278 429 266 142 286 273 272]
- Rand Coefficient is: 0.777708772153
- Jaccard Coefficient is: 0.294757207801

Plot for iyer.txt for 10 Clusters



2. <u>Hierarchical Agglomerative Clustering</u>

Hierarchical clustering algorithms are either top-down or bottom-up. Bottom-up algorithms treat each document as a singleton cluster at the outset and then successively merge or agglomerate pairs of clusters until all clusters have been merged into a single cluster that contains all documents. This is called Hierarchical Agglomerative Clustering (HAC).

Algorithm:

```
1. Compute the distance matrix (Euclidean distance is used here).
2. Let each data point be a cluster.
3. Repeat.
4.         Merge the two closest clusters.
5.         Update the distance matrix.
6. Until only a single cluster remains.
```

Key operation here is the computation of the distance between two clusters.

Advantages:

- No a-priori information about the number of clusters is required.
- It can handle non-global shapes.

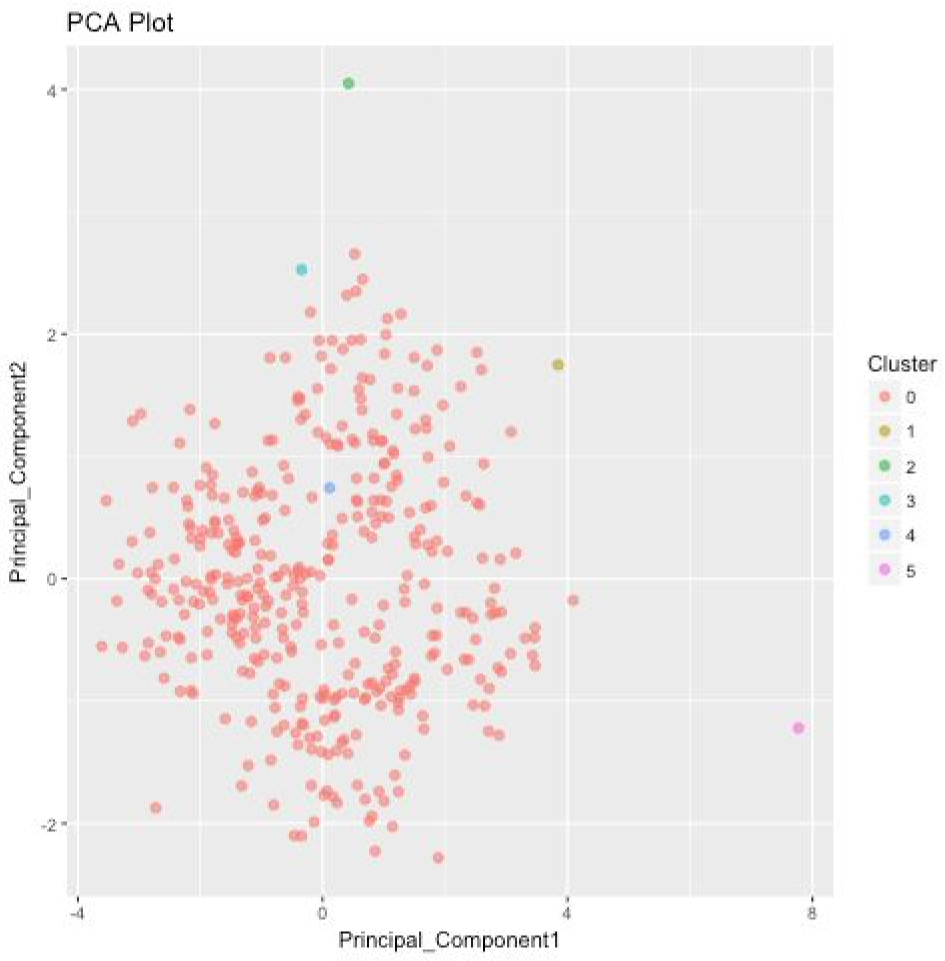
Disadvantages:

- Sensitive to noise and outliers.
- Time complexity of at least $O(n^2 \log n)$ is required, when n is the number of data points.
- Difficulty handling different size clusters and convex shapes.

Code Results :

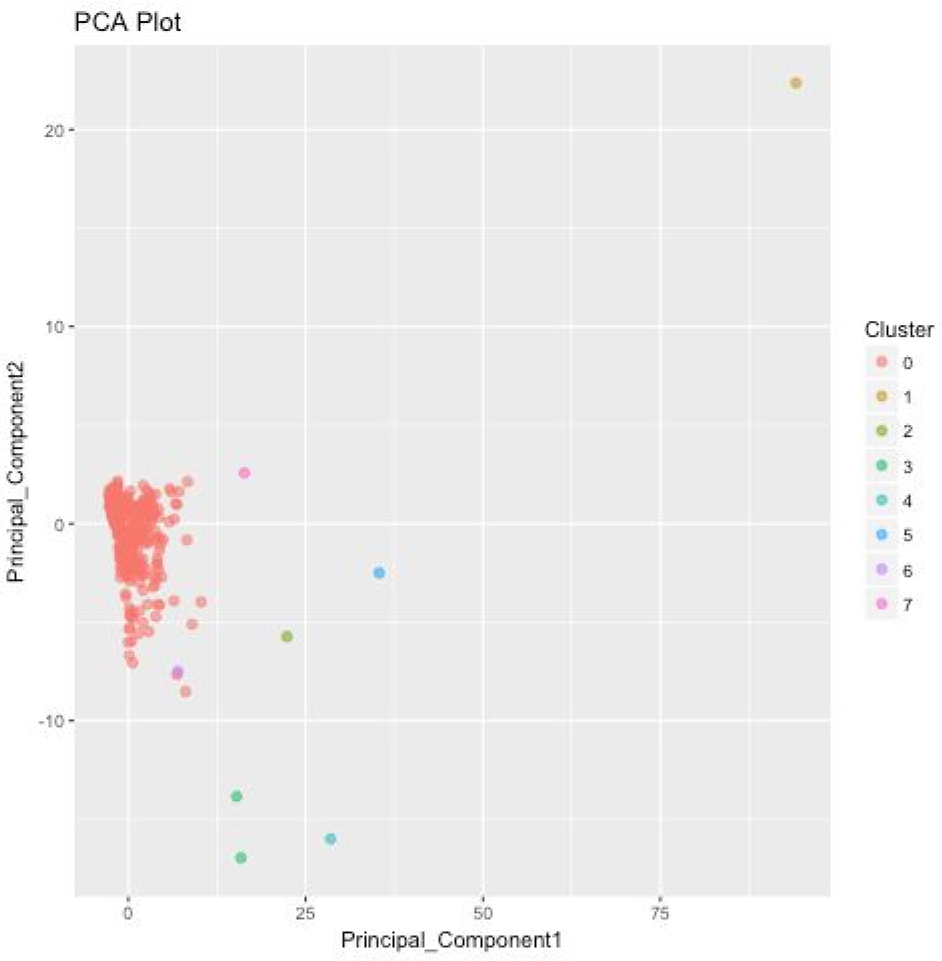
- File Name: cho.txt
- Number of Clusters : 6
- Rand Coefficient is: 0.2436441246744879
- Jaccard Coefficient is: 0.22849318819743958

<u>Plot for cho.txt for 6 Clusters</u>



- File Name: iyer.txt
- Number of Clusters : 8
- Rand Coefficient is: 0.1808566757330081
- Jaccard Coefficient is: 0.15710452461338867

<u>Plot for iyer.txt for 8 Clusters</u>



3. <u>Density based Clustering</u>

Density based clustering algorithm plays a vital role in finding non linear shapes structure based on the density. Density based Spatial Clustering of Applications with Noise (DBSCAN) is most widely used density based algorithm. It uses the concept of density reachability and density connectivity.

Algorithm :

DBSCAN requires two parameters: ε (eps) and the minimum number of points required to form a cluster (minpts)

1. Start with an arbitrary starting point that has not been visited.
2. Extract the neighbourhood of this point using ε (All points which are within the ε distance are neighbourhood).
3. If there are sufficient neighbourhood around this point i.e. neighbourhood is greater than or equal to the minpts, then clustering process starts and point is marked as visited else this point is labeled as noise (Later this point can become the part of the cluster).
4. If a point is found to be a part of the cluster then its ε neighbourhood is also the part of the cluster and the above procedure from step 2 is repeated for all ε neighbourhood points. This process is repeated until all points in the cluster is determined.
5. A new unvisited point is retrieved and processed, leading to the discovery of a further cluster or noise.
6. This process continues all the points are marked as visited.

Advantages:

- Resistant to noise.
- Can handle clusters of different shapes and sizes.
- Does not require a-priori specification of number of clusters.

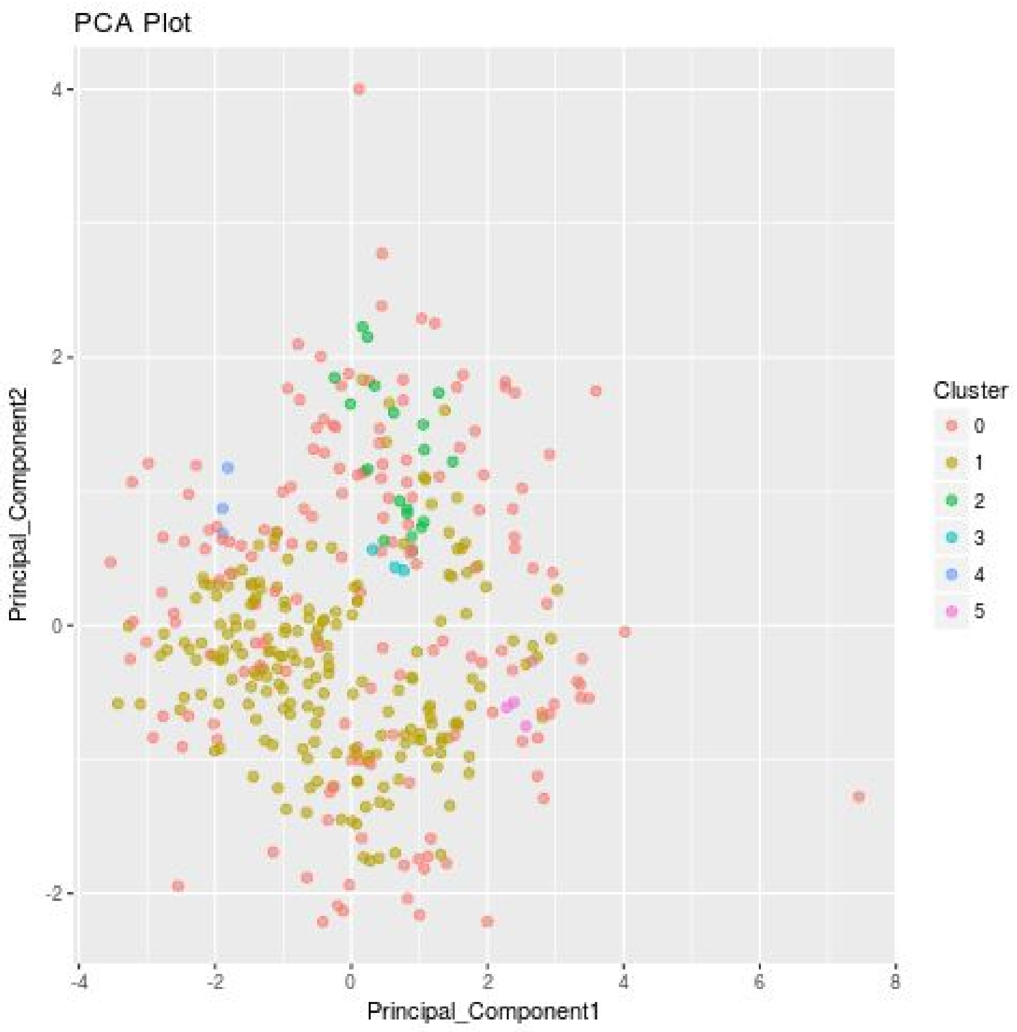
Disadvantages:

- It fails in case of varying density clusters.
- Does not work well in case of high dimensional data.
- Sensitive to parameters - hard to determine the correct set of parameters.

Code results:

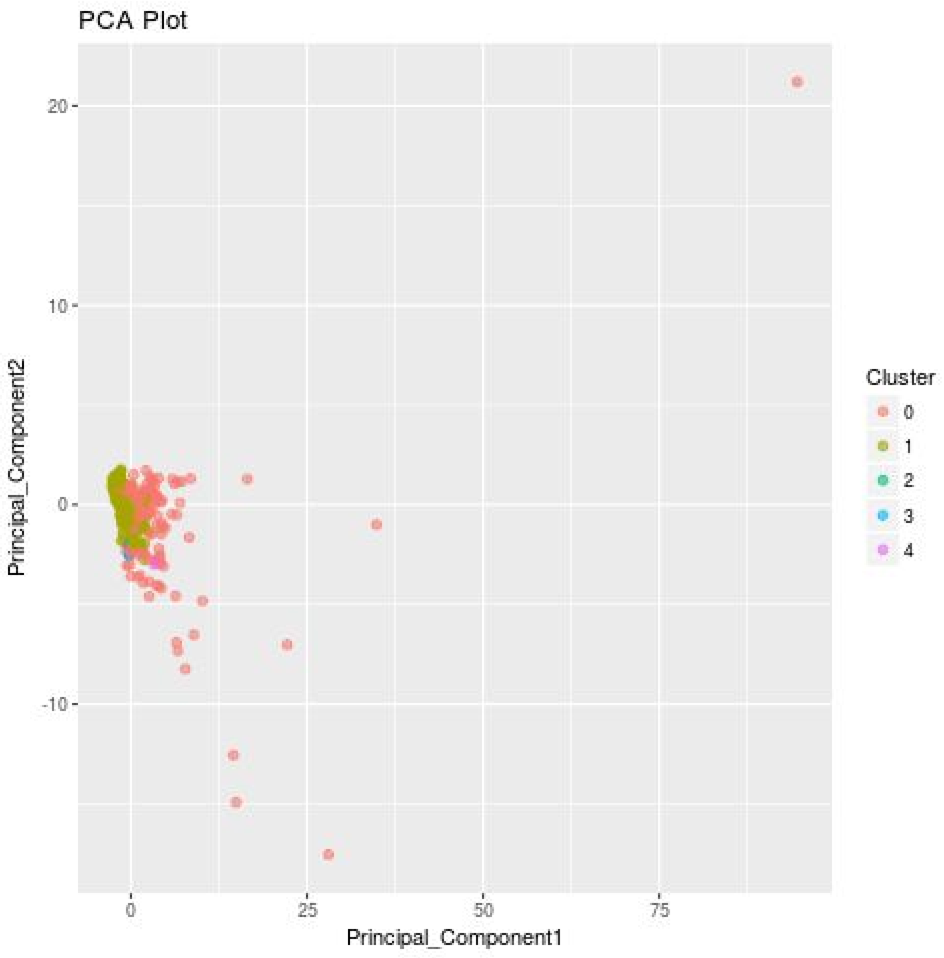
- File Name: cho.txt (ε=1.11 and minpts=3)
- Number of clusters formed = 5
- Rand index = 0.5581894815968214
- Jaccard coefficient = 0.20165906665373048
- In the plot below the clusters denoted by 0 are noise and outliers.

<u>Plot for cho.txt for ε=1.11 and minpts=3</u>



- File Name: iyer.txt (ε=1.21 and minpts=3)
- Number of clusters formed = 4
- Rand index = 0.5045362884368605
- Jaccard coefficient = 0.21840898494443428
- In the plot below the clusters denoted by 0 are noise and outliers.

<u>Plot for iyer.txt for ε=1.21 and minpts=3</u>



COMPARISON:

K-Means	**HAC**	**DBSCAN**
Prior knowledge of number of clusters required	No prior knowledge of number of clusters required	No prior knowledge of number of clusters required
Sensitive to outliers	Sensitive to outliers	Segregates the outliers
K-means is efficient with high speed performance	HAC takes more time to converge	Fast and efficient
Good for large datasets	Not very good for large dataset	It is scalable
Can only detect spherical clusters	Can handle non-global shapes	Can handle clusters of various shapes and sizes

PART 2: MapReduce K-means

There were three python files made:

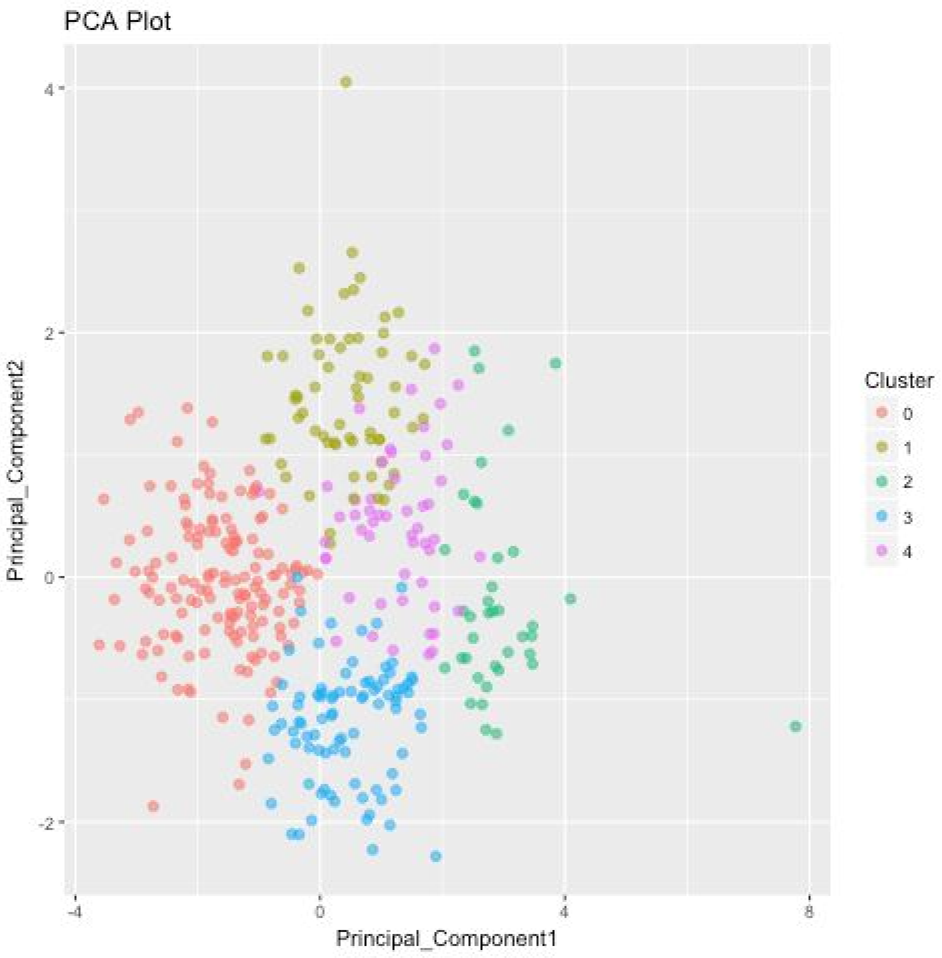
1. **Mapper.py** : The mapper functions takes in the input file line by line.For each line, the distance of it is calculated with each of the k centroids and the cluster with minimum distance value is emitted the line from mapper.
2. **Reduce.py-** In the reducer, each cluster centroid along with the genes that belong to that cluster are read line by line.For each cluster centroid, the average of all gene values is taken and this forms the new centroid of this cluster.This is emitted from the reducer as output
3. **Runner.py** : Started with setting the initial centroids,number of desired clusters and maximum iterations and then calling the mapper and reducer functions using the subprocess module . After the convergence the external indices(Jaccard and Rand) are calculated and PCA plots are made.

On running the K-Means and K-means Map Reduce codes on the same centroids for datasets ‘cho.txt’ and ‘iyer.txt’ we got the same values for Jaccard coefficient and Rand index.

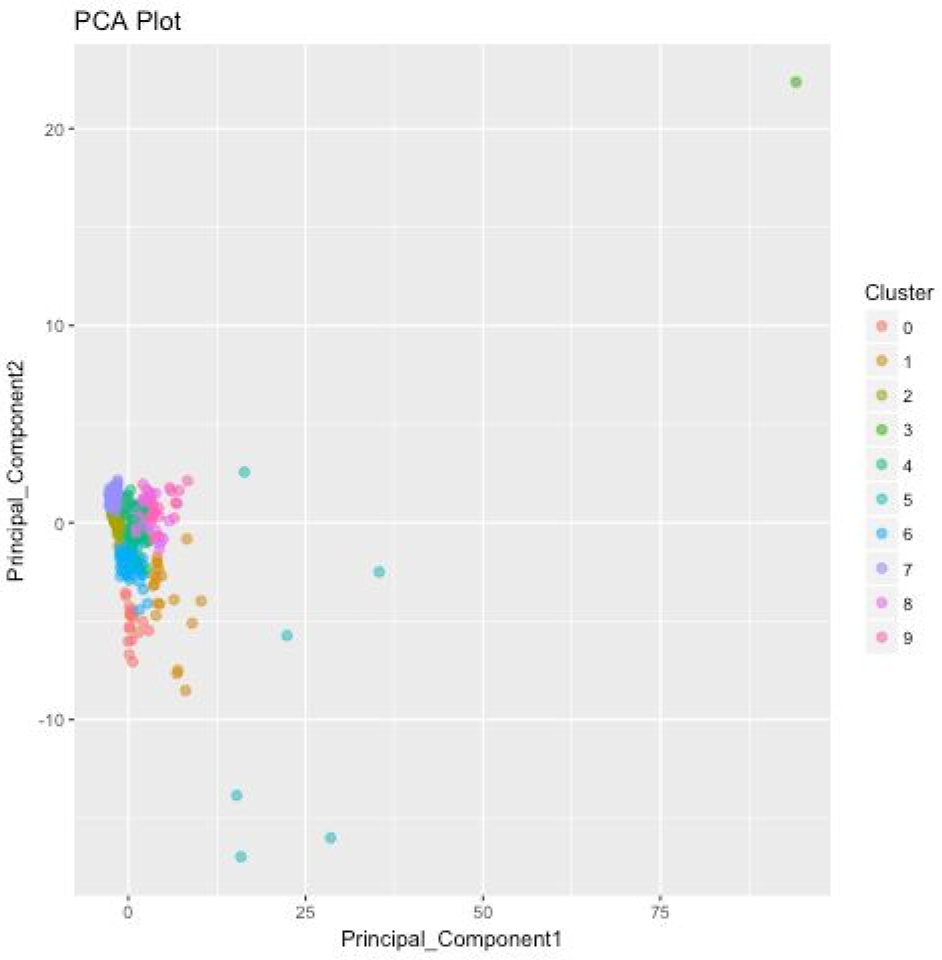
Parallel v/s Non-parallel Implementation of k-means

Non-parallel k-means clustering algorithm computes distance matrix serially whereas the parallel implementation does this computation in parallel thus reducing the overall runtime. Hence, parallel k-means clustering algorithm is extremely useful when clustering large datasets.

Plot for cho.txt for 5 Clusters



Plot for iyer.txt for 10 Clusters


PCA Plot
Principal_Component2
Principal_Component1
Cluster