**Project 2**

**Clustering Algorithms**

**Team Members**

**Abhijeet Jadhav**

**(UBIT Name – ajadhav4)**

**(UB Person No - 50291088)**

**Anirudh Dalmia**

**(UBIT Name – adalmia)**

**(UB Person No - 50291396)**

**Siddharth Pandey**

**(UB IT Name – sppandey)**

**(UB Person No - 50288608)**

**Hierarchical Clustering - Agglomerative**

**Algorithm Description:**

Hierarchical clustering is an algorithm that groups objects into clusters based on similarity. The end result is a set of distinct clusters, wherein each cluster element is broadly similar to other objects in the same cluster.

**Calculating the Similarity:**

In order to decide the clusters to be combined or split a measure of dissimilarity is required. This measure is based on two important factors i.e. a metric to measure distances between the pairs of observation and a linkage criterion that specifies the dissimilarity amongst sets as a function of distance taken pairwise of the observations in the sets.

**Let’s discuss the Metric and Linkage Criterion in detail:**

**Metric:**

There are numerous metrics to measure distance between the pairs of observation, few of them are Euclidean Distance, Squared Euclidean Distance, Manhattan distance, etc. It is necessary to choose the appropriate metric as that would influence the shape of the clusters, since different metrics have different ways of evaluating the distances between the pairs. In our testing we have used Euclidean Distance to find the distance between the pairs of observation.

**Linkage Criterion:**

Once a distance metric is used to get the distances between the observation pairs it is important to determine that if the distance is computed between the two most similar or two least similar parts of the cluster. This explanation of metric usage between two closer or two farther parts of cluster sets the base for linkage criterion.

Some Commonly used linkage criteria are as follows:

* Minimum or Single Link
* Maximum or Complete Link
* Group Average or Average Link
* Centroid Distance
* Ward’s Method

**Approach for Hierarchical Clustering:**

Hierarchical Clustering can be performed using the two following strategies:

**Agglomerative Clustering:**

* Often referred as “bottom-up” approach wherein each observation starts in its own cluster and pairs of clusters are merged as we move up the hierarchy.

**Divisive Clustering:**

* Often referred as “top-down” approach wherein each observation starts in one cluster and splits are performed on cluster as we move down the hierarchy.

For Our Implementation we have used Hierarchical Agglomerative Clustering using the Single Link lnkage criterion.

**Overview of the implementation:**

Step 1: Take the cluster number as input.

Step 2: Computation of the distance matrix - for computing the distance matrix we have used the ‘pdist’ function of the SciPy package by setting the distance metric to Euclidean.

Step 3: Assign each individual gene\_id a cluster id

Step 4: Merge the closest clusters iteratively based on similarity and update the distance matrix.

Step 5: Halt the Clustering process once we form k clusters and print the results.

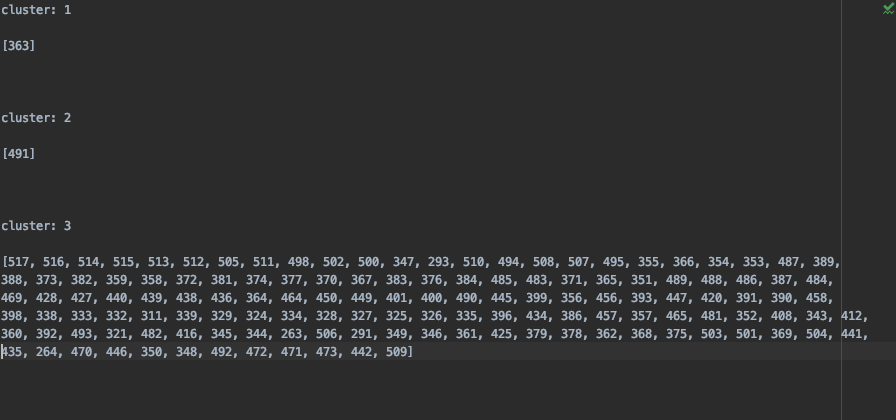
Step 6: Label each gene\_id with its assigned cluster

Step 7: Pass the gene\_id and assigned cluster labels to the PCA method to perform dimensionality reduction and plotting the scatter plot.

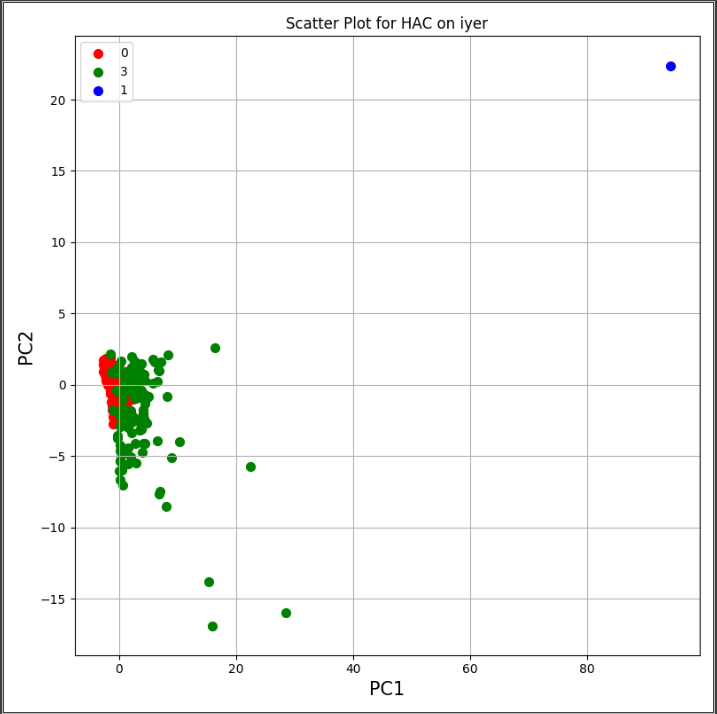
**Result Visualization:**

**For the purpose of our testing we have used two datasets, “iyer.txt” and “cho.txt”.**

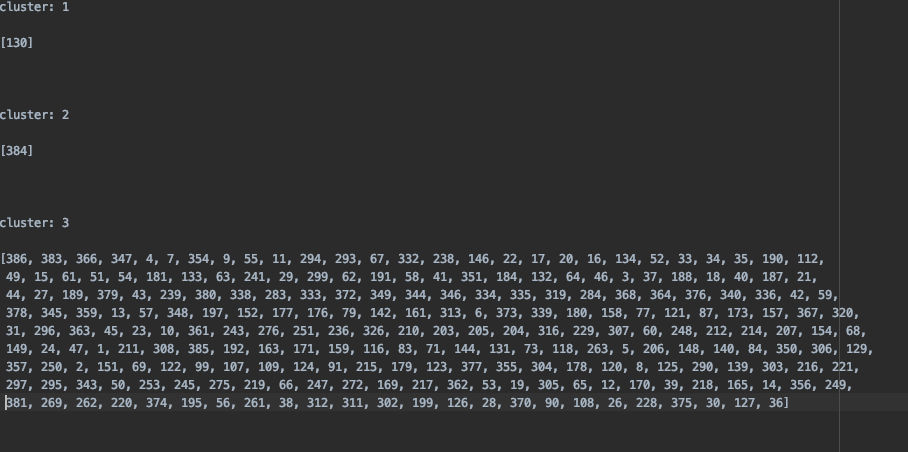
1. We did our intial testing on Iyer.txt and following results were obtained
2. For Cluster Size 3, below cluster output is obtained.



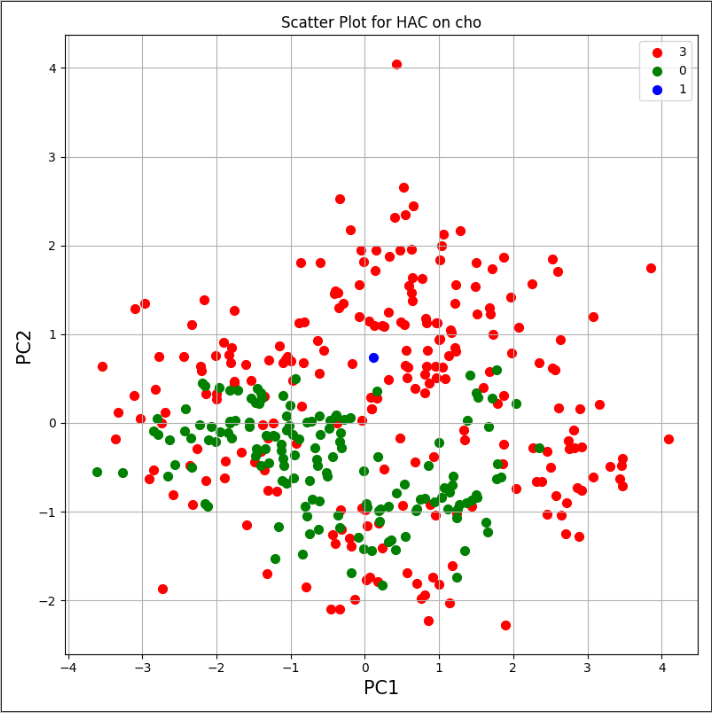
1. Scatter Plot for Clustered Data in Iyer.txt



1. **On cho.txt following results were obtained.**
2. For Cluster Size 3, below cluster output is obtained.



1. Scatter Plot for Clustered Data in Iyer.txt

****

**Pros of Hierarchical Clustering:**

1. This method has no necessity of assuming particular number of cluster.
2. A certain set of clusters can be obtained by halting the clustering process once desired number of clusters are formed
3. Easy To implement.

**Cons of Hierarchical Clustering:**

1. Once a decision is made to combine two clusters, it cannot be undone
2. Time complexity: not suitable for large datasets
3. The order of the data has an impact on the final results
4. Highly Sensitive to Noise and outliers
5. Few other cons are. – difficulty in handling different sized clusters and irregular shapes.

**Spectral Clustering**

**Algorithm Description:**

Spectral Clustering is a clustering method based on finding K- clusters using the eigen vectors of a matrix. The matrix used to calculate the eigen vectors is derived from a set of pairwise similarities between the points to be clustered.

**Calculating the Similarity:**

In order to move ahead with the clustering, the first step is to obtain the similarity matrix. There are different ways of obtaining the same, the simplest being, calculating the distance matrix, setting a threshold value and eliminating values above the threshold in the matrix. However, in our implementation each gene\_id has multiple features which can be treated as a vector and then the similarity is typically the Gaussian Kernel.

Below we discuss the step by step procedure of building all the matrices needed to find the eigen values and eigen vector.

**Step 1: Gaussian Kernel**

****

The parameter σ plays a similar role as the parameter ε in the ε-neighborhoods. Points in local neighborhoods are connected with a relatively high weights, while edges between far away points have a positive, but negligible weights.

In our code we have used the

**Step 2: Adjacency Matrix (A)**

An adjacency matrix in the simplest defines the edge between two nodes if has a value greater than 0. In our code we obtain the adjacency matrix after performing the gaussian kernel on our distance matrix.

**Step 3: Degree Matrix (D)**

Once we have the adjacency matrix ready we build the degree matrix. A degree matrix is basically a diagonal matrix where we sum all values of each row in adjacency matrix and store it in the respective diagonal location for each row and column. The other values in the matrix other than the diagonal values are zeroes.

**Step 4: Laplacian matrix (L)**

A Laplacian matrix is obtained by subtracting the adjacency matrix from degree matrix.

L = D-A

**Step 5: Eigen Values and Eigen Vectors**

The Laplacian matrix obtained can now be used to generate the eigen values and eigen vectors.

**Step 6: Find Eigen Gap and Perform K-means on the obtained eigen vectors**

Eigen Gap : Eigen Gap is the difference between two successive eigenvalues where all the eigen values are sorted.

Thus we sort the eigen values and find the maximum eigen gap. The index where we find the maximum eigen gap is the number of columns of eigen vector we will consider for performing k-means.

The number of cluster and initial centroids are accepted at the beginning of the code.

Once we perform K-means we return the cluster\_id\_labels for each gene and move towards plotting the PCA scatterplot.

**Overview of the implementation:**

Step 1: Take the number of cluster and initial centroids as input.

Step 2: Compute the Similarty matrix, Adjacency matrix, Degree Matrix and Laplacian Matrix.

Step 3: Find the eigen values and eigen vectors for the Laplacian matrix.

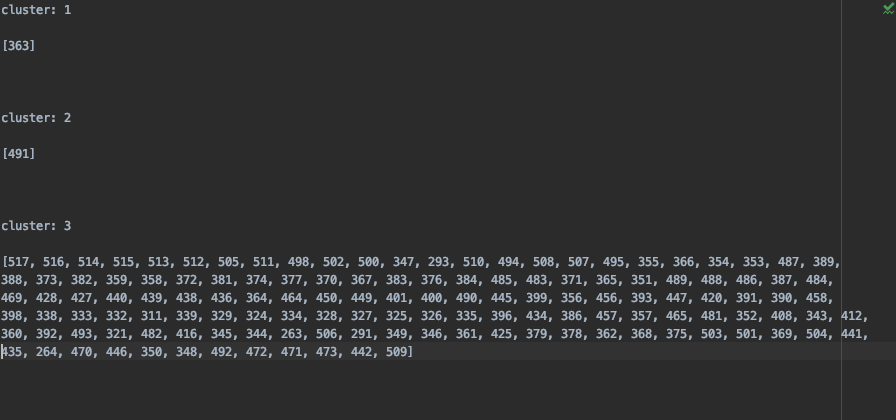
Step 4: sort the eigen values to find the nth maximum eigen gap and perform k-means on eigen vectors from 0 to n columns.

Step 5: Return the assigned cluster labels to each gene id and plot the PCA graph.

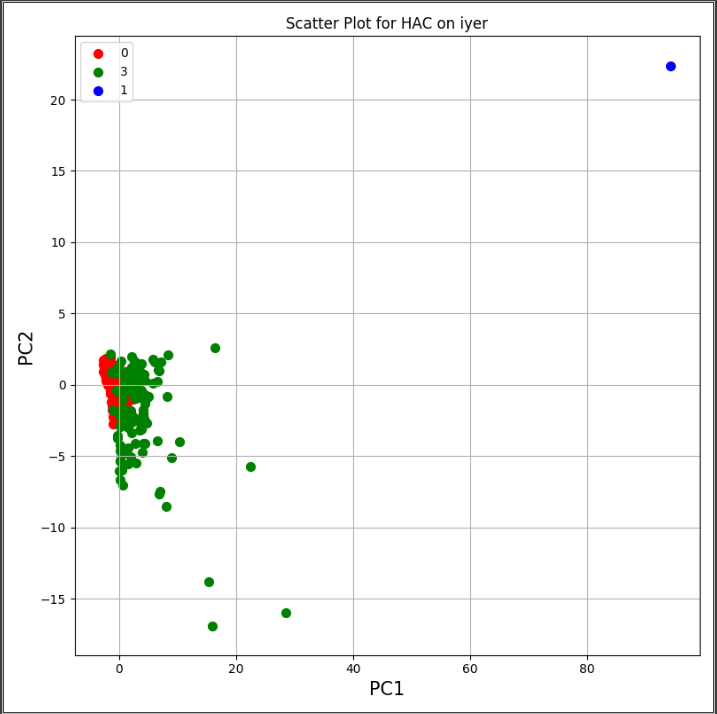
**Result Visualization:**

**For the purpose of our testing we have used two datasets, “iyer.txt” and “cho.txt”.**

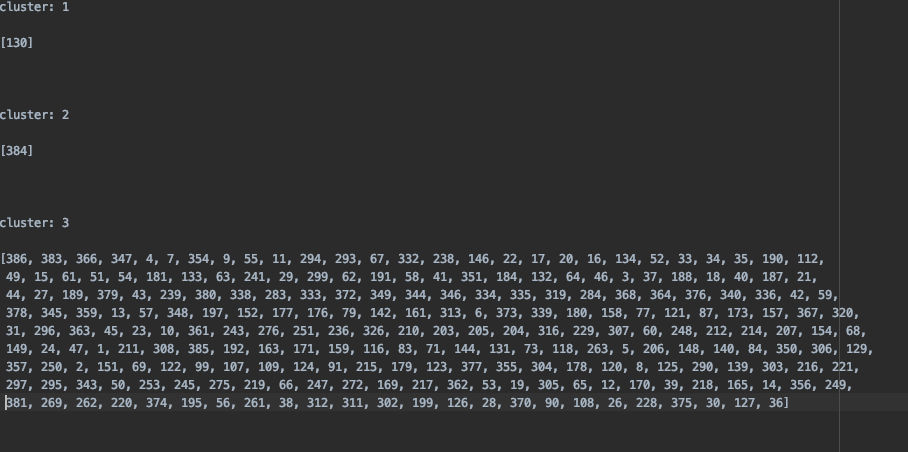
1. We did our intial testing on Iyer.txt and following results were obtained:
2. For Cluster Size 3, below cluster output is obtained.



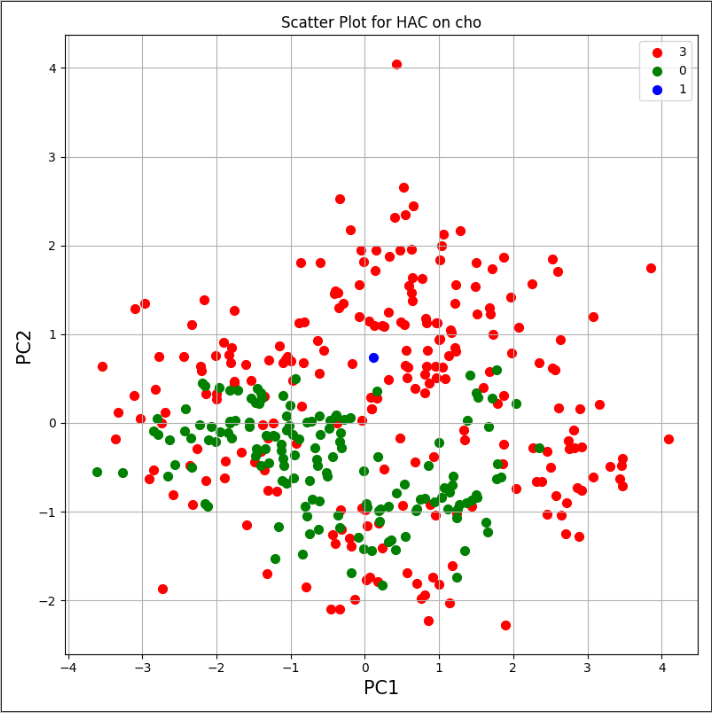
1. Scatter Plot for Clustered Data in Iyer.txt



1. **On cho.txt following results were obtained.**
2. For Cluster Size 3, below cluster output is obtained.



1. Scatter Plot for Clustered Data in Iyer.txt

****

**Pros of Spectral Clustering:**

1. Works well when relations are approximately transitive
2. Effectively handles clusters whose overlap or connectedness varies across clusters.
3. Well Founded Mathematically

**Cons of Spectral Clustering:**

1. Very Noisy Datasets cause problems – performance can drop suddenly from good to terrible.
2. Computing eigen vectors is a bottleneck for very large datasets.