Data Parsing

We have a dataset named “geneExpFinal.csv” which has 11590 entries and 197 features. We need to extract the following genes from the dataset and make a new dataset. The genes are: Acads, Dusp28, Emx1, Vip, Urah.

The workflows are given below:

- Read “geneExpFinal.csv” using pandas.read_csv()
- Extract the genes from the dataframe object
- Transpose the dataframe
- Thus, we have our new dataframe having 197 entries and 5 features.

Dimensionality reduction

Dimensionality reduction is defined as a method of reducing variables in a dataset. The process keeps a check on the dimensionality of data by projecting high dimensional data to a lower dimensional space that encapsulates the ‘core essence’ of the data.
PCA is a popular dimensionality reduction technique. It transforms the original features into a new set of uncorrelated variables called principal components.

Steps for PCA are stated below:

1. Standardize the data (subtract mean and divide by standard deviation)
2. Compute the covariance matrix of the standardized data
3. Calculate the eigenvectors and eigenvalues of the covariance matrix
4. Sort the eigenvectors by their corresponding eigenvalues in decreasing order
5. Choose the top k eigenvectors to form a transformation matrix
6. Multiply the original data by the transformation matrix to obtain the reduced-dimensional representation

How to do PCA:

In Python scikit-learn library, there are a class named PCA to perform principal component analysis. To import PCA we can write the following:

```
from sklearn.decomposition import PCA

# Perform PCA

pca_object = PCA(n_components=2, random_state=42)

pca_result = pca_object.fit_transform(data)
```

Clustering

Clustering is a machine learning task that involves grouping simillar objects into clusters. The goal of clustering is to separate groups with similar characteristics and assign them to clusters. One common clustering algorithm is K-means.

K-means Approach:

1. Initialization: Randomly select K initial cluster centroids
2. Assignment: Assign each data point to the nearest centroid, forming K clusters
3. Update Centroids: Recalculate the centroids as the mean of all points in each cluster
4. Repeat: Repeat steps 2 and 3 until convergence

How to do clustering:

There is a class named Kmeans in scikit-learn in Python. To do clustering we can write following code:

```
from sklearn.cluster import KMeans

n_clusters = 4

model = KMeans(n_clusters=n_clusters, random_state=42)

labels = model.fit_predict(data)

centroids = model.cluster_centers_
```

Results and Discussions

Clustering for all Genes:

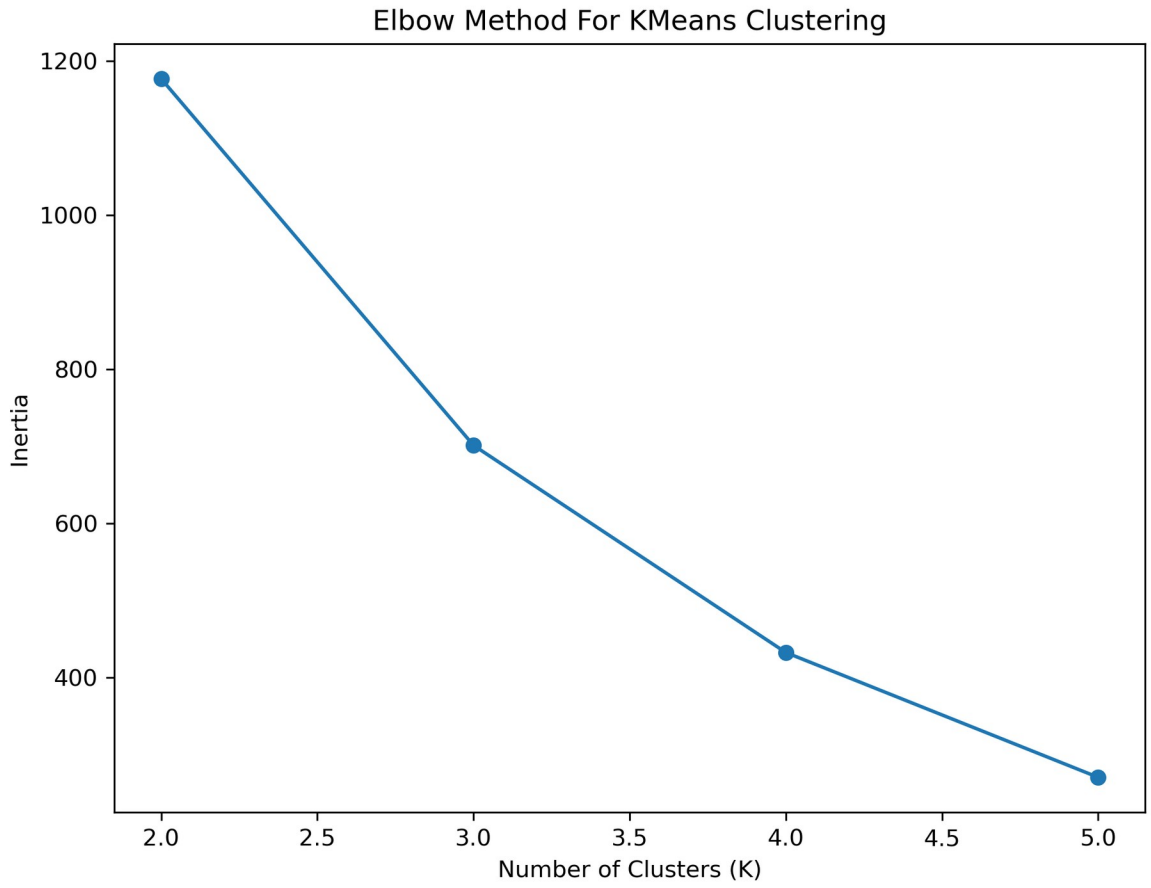

Figure 1: Elbow Method for choosing K for all Genes

From Figure-1 we can choose K clusters for Kmeans clustering method.

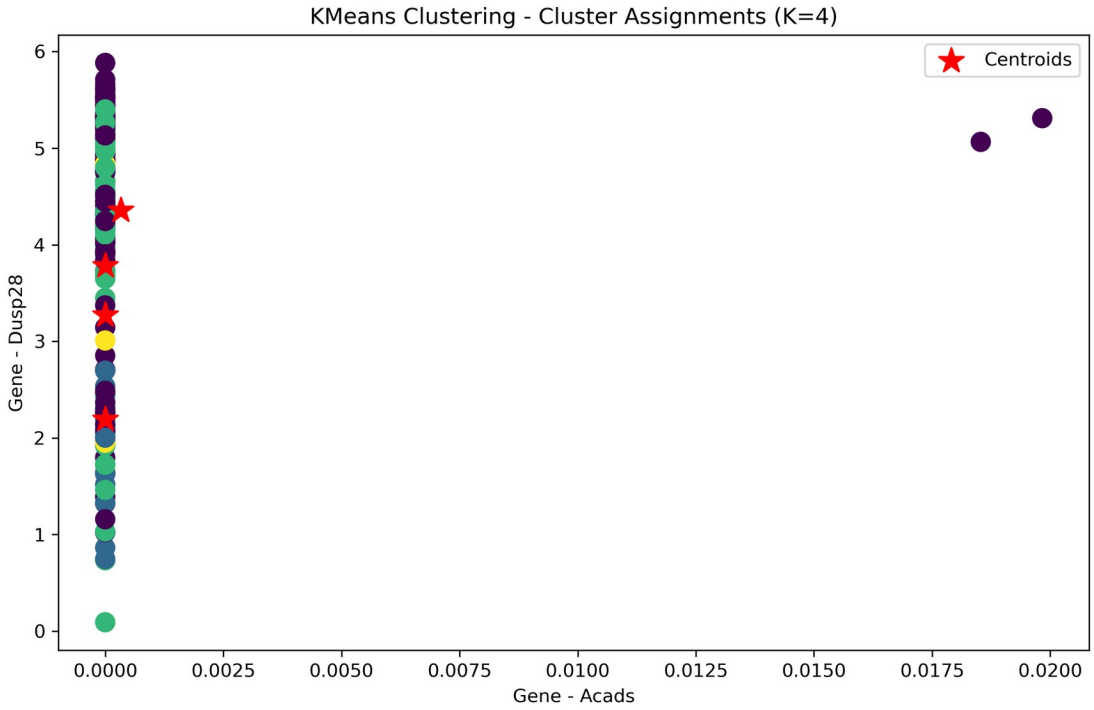

Figure 2: Clustered Data for all Genes

In Figure-2, there are 4 clusters for Gene Acads in x-axis and Gene Dusp28 for y-axis. Beyond three dimensions, it becomes difficult to visualize and interpret clusters in a meaningful way. While we can still perform clustering with higher-dimensional data, visualizing and understanding the clusters become increasingly challenging. In high-dimensional spaces, the data points tend to spread out, and the concept of distance becomes less meaningful. This makes it harder for the algorithm to identify meaningful clusters.

For this reason, we showed clustering between only two genes. In Figure-2, the scatter plot between two genes forms a liner representation. We can think the upper right two points as outliers in this case. The above figure doesn't show a meaningful cluster representation.

Dimensionality Reduction on Whole Data:

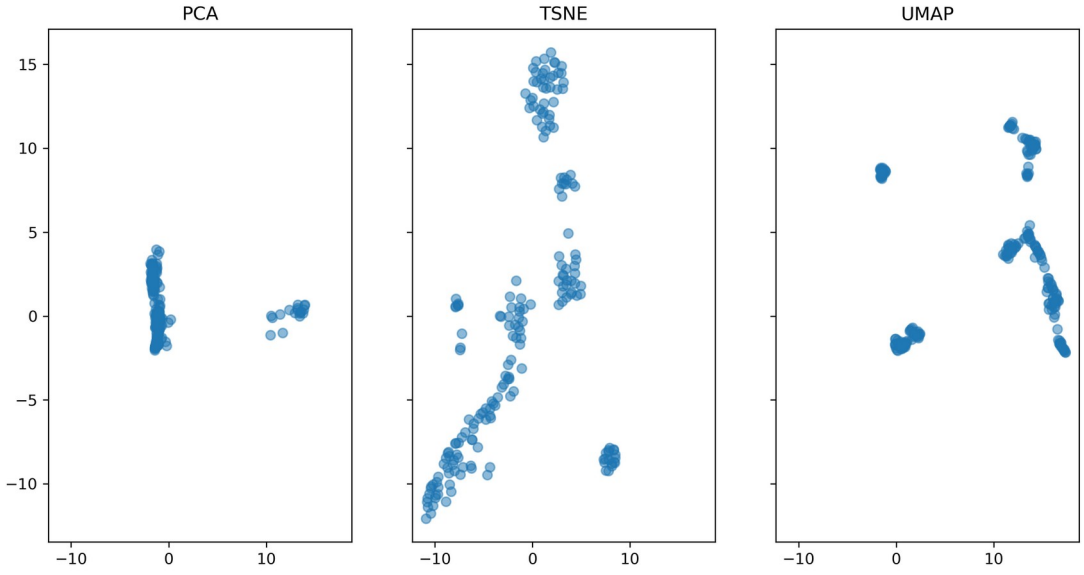

Figure 3: Dimensionality Reduction using PCA, TSNE and UMAP

PCA (Principal Component Analysis):

PCA aims to capture the maximum variance in the data by transforming it into a new set of uncorrelated variables (principal components). In a PCA plot, each point represents a data point projected onto the principal components. Points closer together in the plot have similar patterns, and the distance reflects their dissimilarity.

t-SNE (t-Distributed Stochastic Neighbor Embedding):

t-SNE focuses on preserving local relationships between data points. Points that are close in the original high-dimensional space are also close in the t-SNE plot. Clusters in the t-SNE plot indicate groups of points that share similar features.

UMAP (Uniform Manifold Approximation and Projection):

UMAP is similar to t-SNE but is generally faster and potentially preserves more global structure. UMAP emphasizes preserving both local and global data relationships. Clusters in a UMAP plot reveal patterns in the data, with distances representing similarities.

From Figure-3, we can see the UMAP plot is more nicely organized than others.

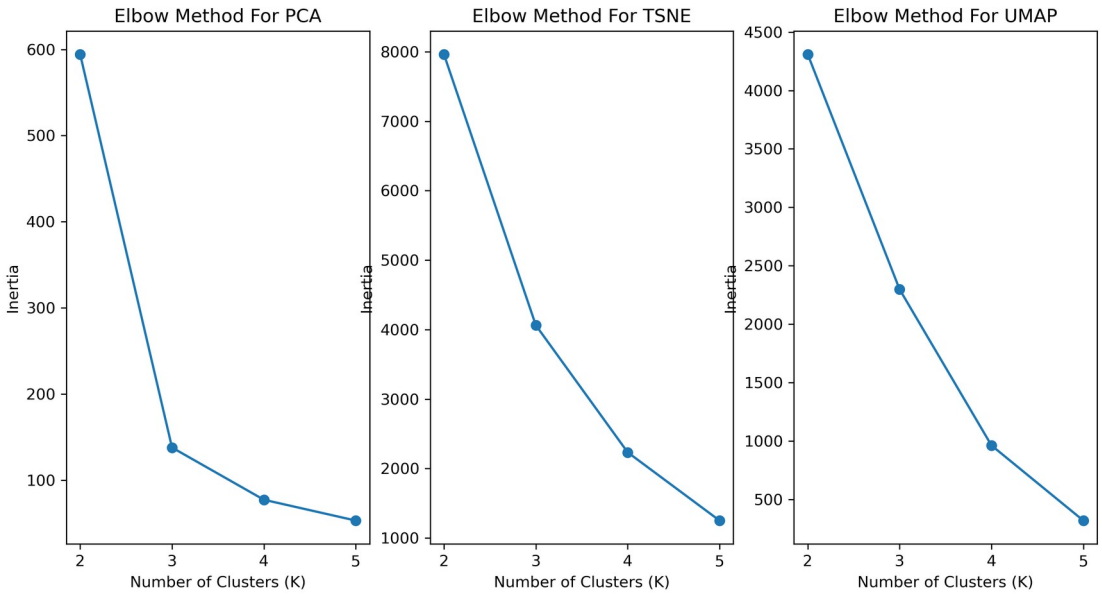

Figure 4: Elbow Method for Reduced Genes from 5 Genes

From the Figure-4, the optimal cluster numbers (K) is 4.

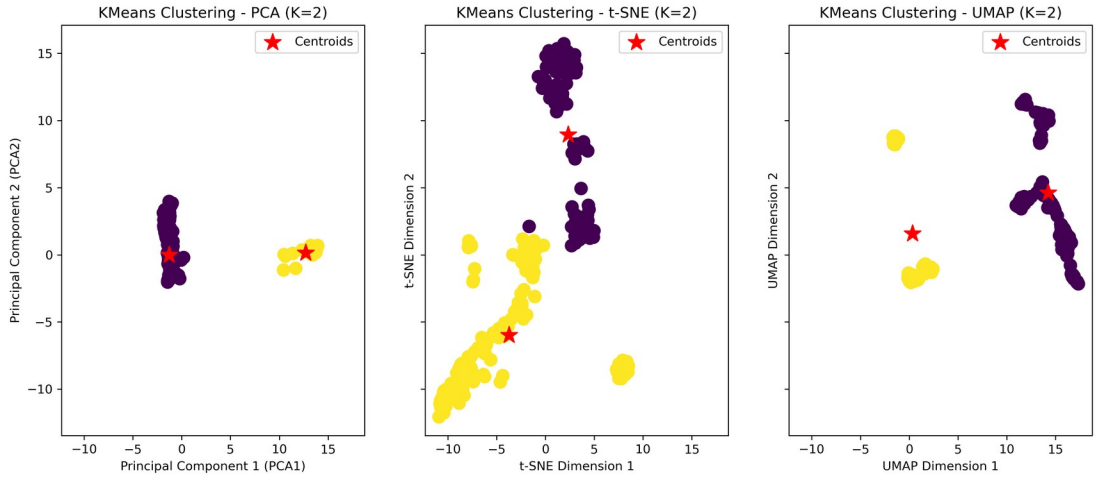

Figure 5: Clustering Reduced Data for K=2

We want to check if we choose a wrong K value, what will happen to cluster representation. In Figure-5, the clusters aren't properly assigned for K=2.

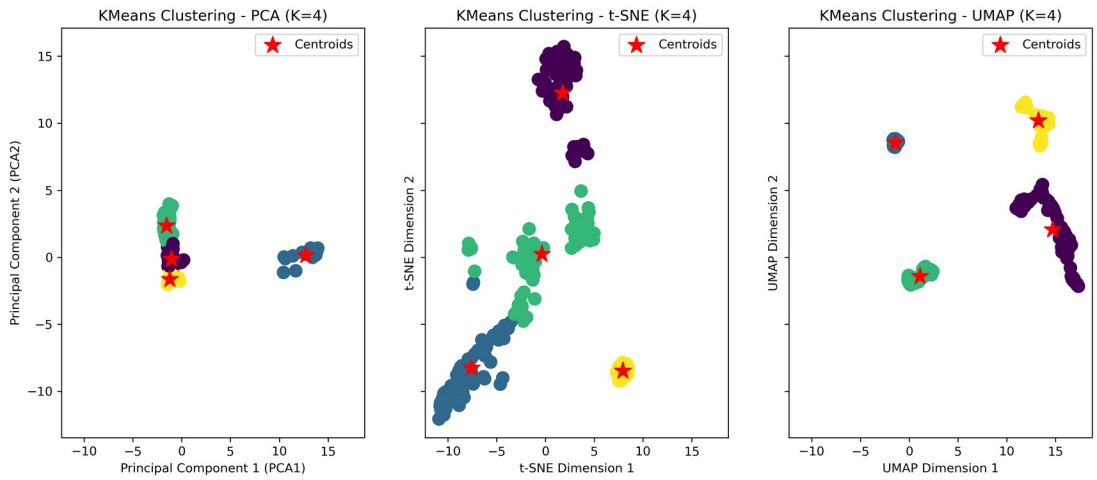

Figure 6: Clustering Reduced Data for K=4

While using K=4, we can see in Figure-6, the clusters are well assigned.

Clustering for 20 Genes:

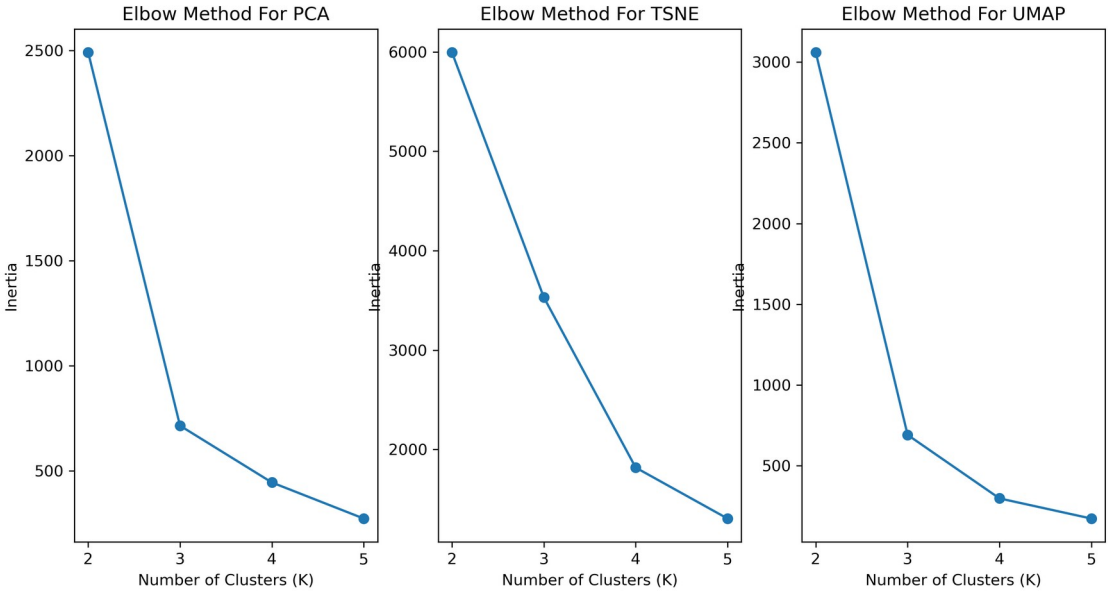

Figure 7: Elbow Method for Reduced Data from 20 Genes

We extracted top 3 genes for each assigned genes to us. Thus we got 3*5=15 and assigned 5, total 15 + 5 = 20 genes. We combined data for 20 genes to make a new dataframe. After dimensionality reduction on that data we got reduced data. We again performed elbow method for that data and got optimal K=4 in Figure-7.

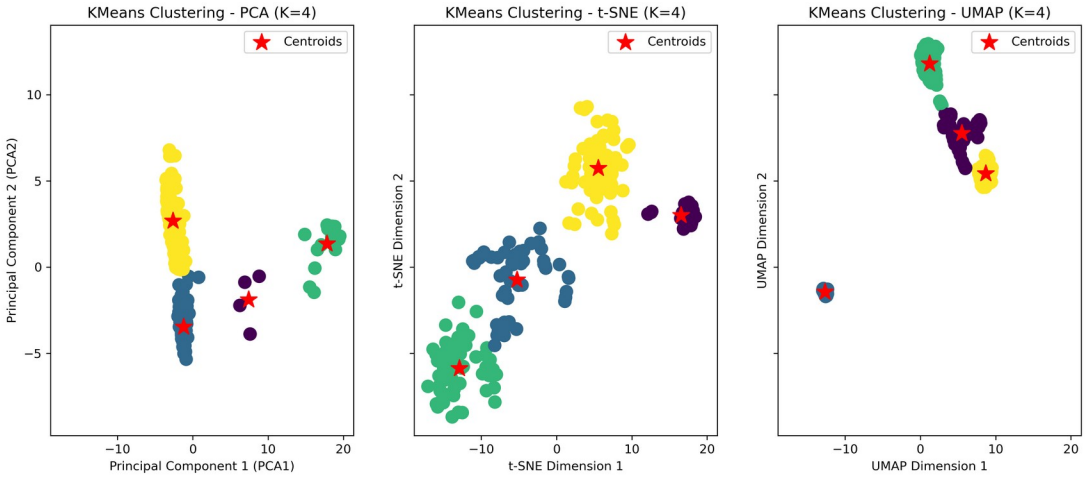

Figure 8: Clustering Reduced Data from 20 Genes

In Figure-8, we can see clusters for reduced data from 20 Genes data.

If we compare between Figure-6 and Figure-8, we can say Figure-6 consist of more better represention of clusters than Figure-8. In Figure-8, there could be outliers points those are represented as indivual clusters in the figure.