CSE 601: Data Mining and Bioinformatics

Project 1 - Part 1

Dimensionality Reduction

Presenters:

Hemant Koti (50338178)

Sai Hari Charan Barla (50336868)

Shravya Pentaparthi (50337027)

PROBLEM:

Implement PCA algorithm, project the high-dimensional data to 2 dimensions, and plot the 2-D plots.

DATASET:

We have implemented dimensionality reduction on three data files (pca_a.txt , pca_b.txt , pca_c.txt)

PCA Implementation:

The following steps are computed in the method '*PCA*'.

1. Find the mean (*mean*) of all the attributes
2. Adjust the original data by the mean(*X_bar*)
3. Compute the covariance matrix of the adjusted data (*covariance_matrix*)
4. Sort the eigen vectors by corresponding eigen values (maximum values) take the first 2 eigen vectors in our example as we are trying to reduce the dimensionality to 2-D
5. Compute the new data points corresponding to these eigen vectors (result 'PCA1' 'PCA2')
6. Color the points of these newly projected points (based on the labels 'Y')

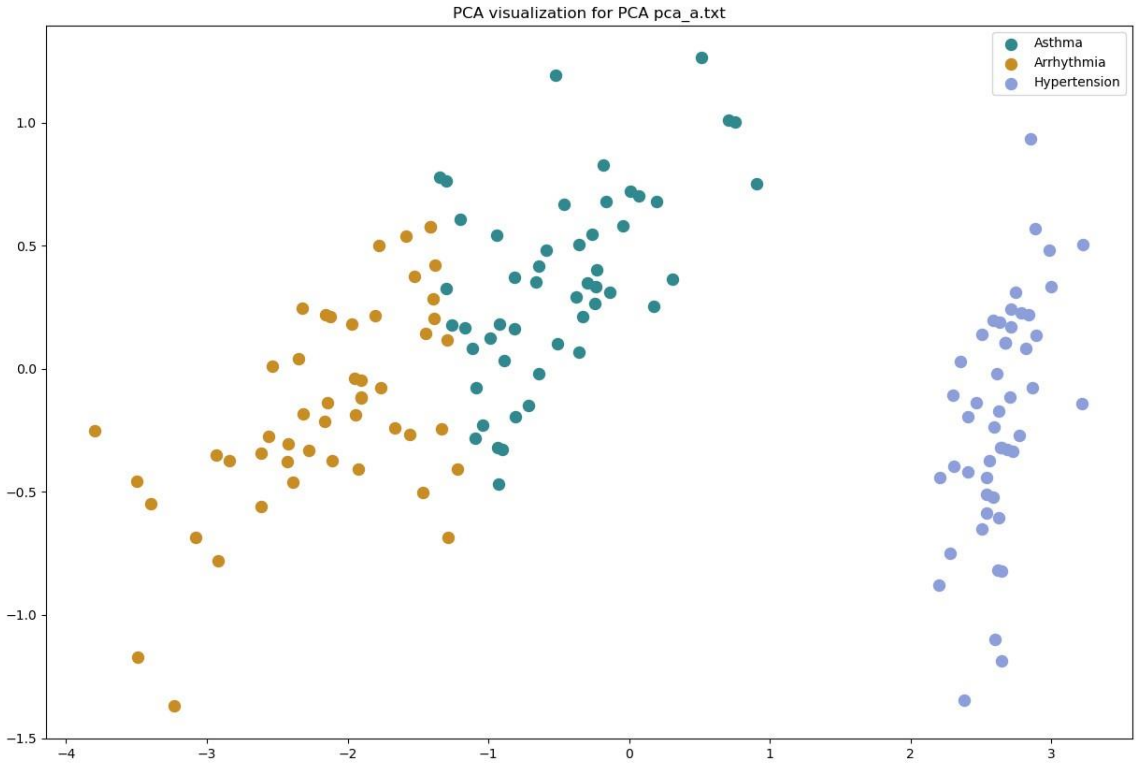
The input parameter to the method is the dataset

The method '*visualizepca'* is used to plot the graph. The input parameters to this file are the title to be printed, transformed points and their corresponding label.

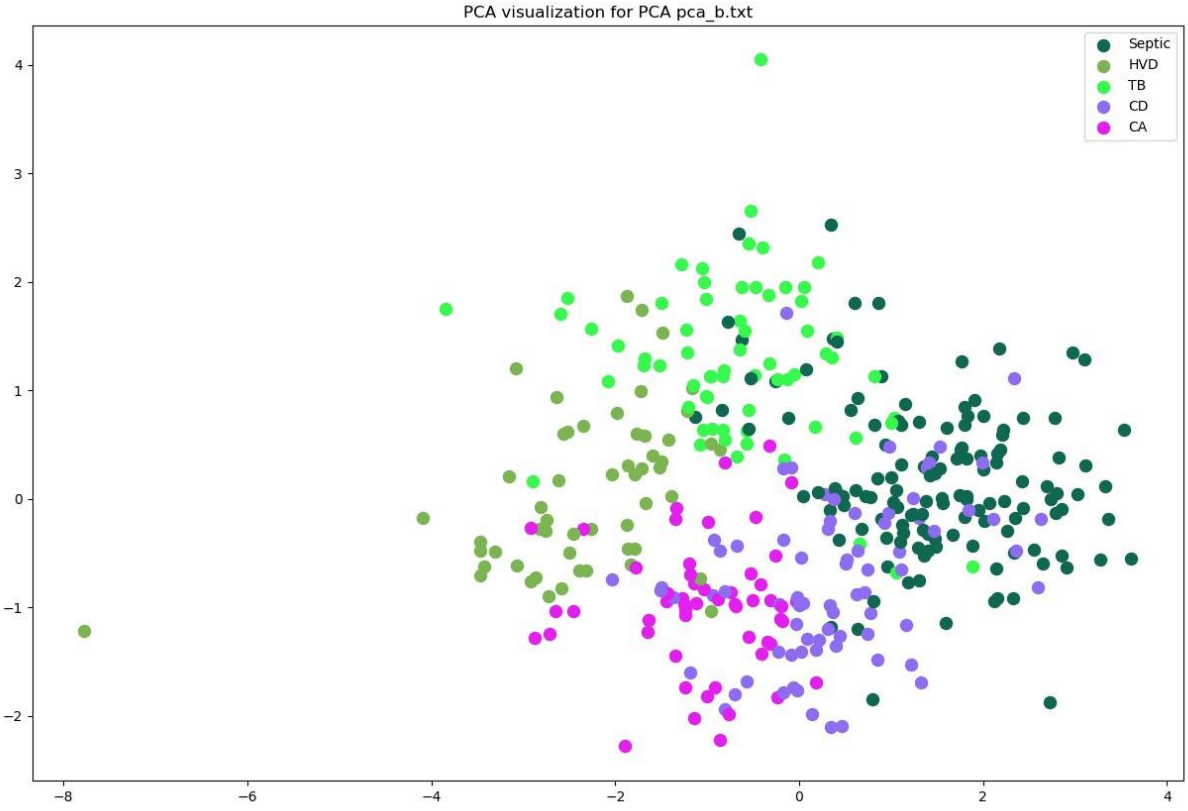
In this method we have generated random colors by merging random HEX values using '*random.choice*' .We have merged these colors to the distinct labels (*label_color*).

Obtained Results:

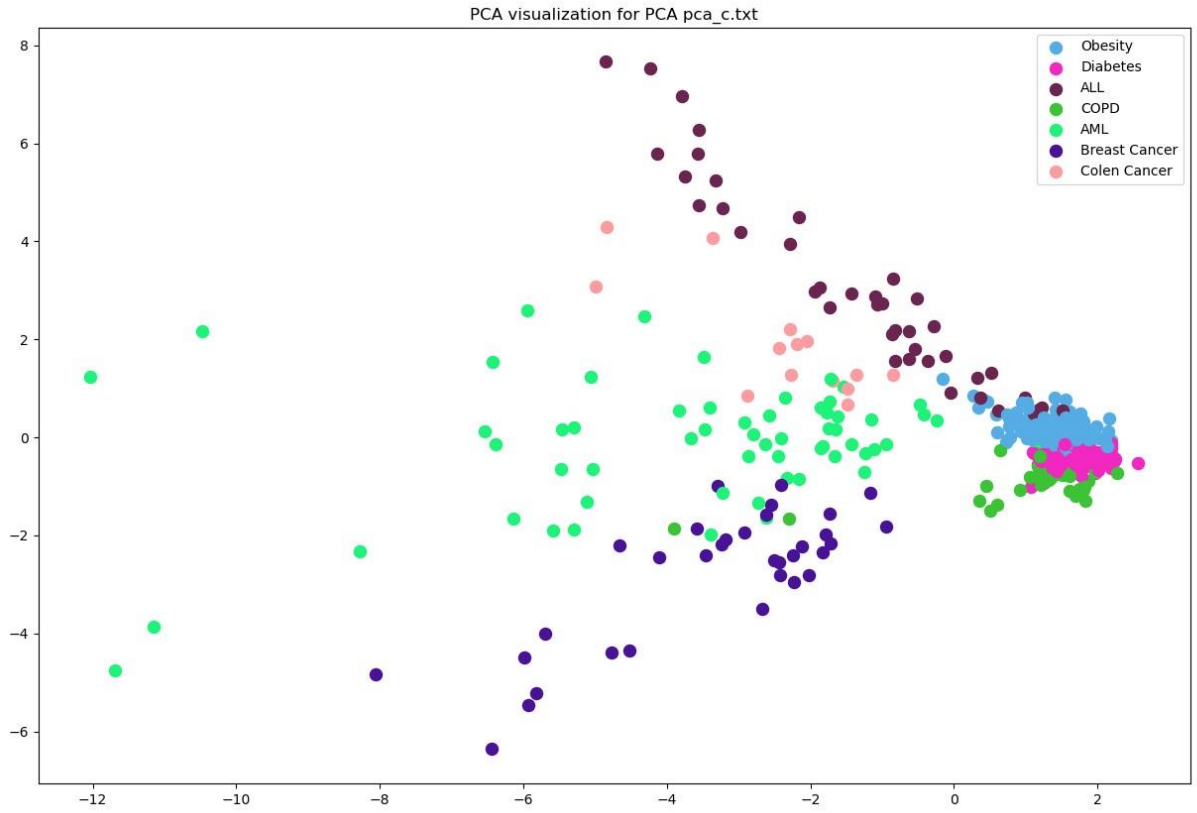
1. Implementing PCA on pca_a.txt



2. Implementing PCA on pca_b.txt



3. Implementing PCA on pca_c.txt



SVD:

Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables (entities each of which takes on various numerical values) into a set of values of linearly uncorrelated variables called principal components.

SVD is computed using the below formula:

$$X = U_{n \times n} \Sigma_{n \times p} V^T_{p \times p}$$

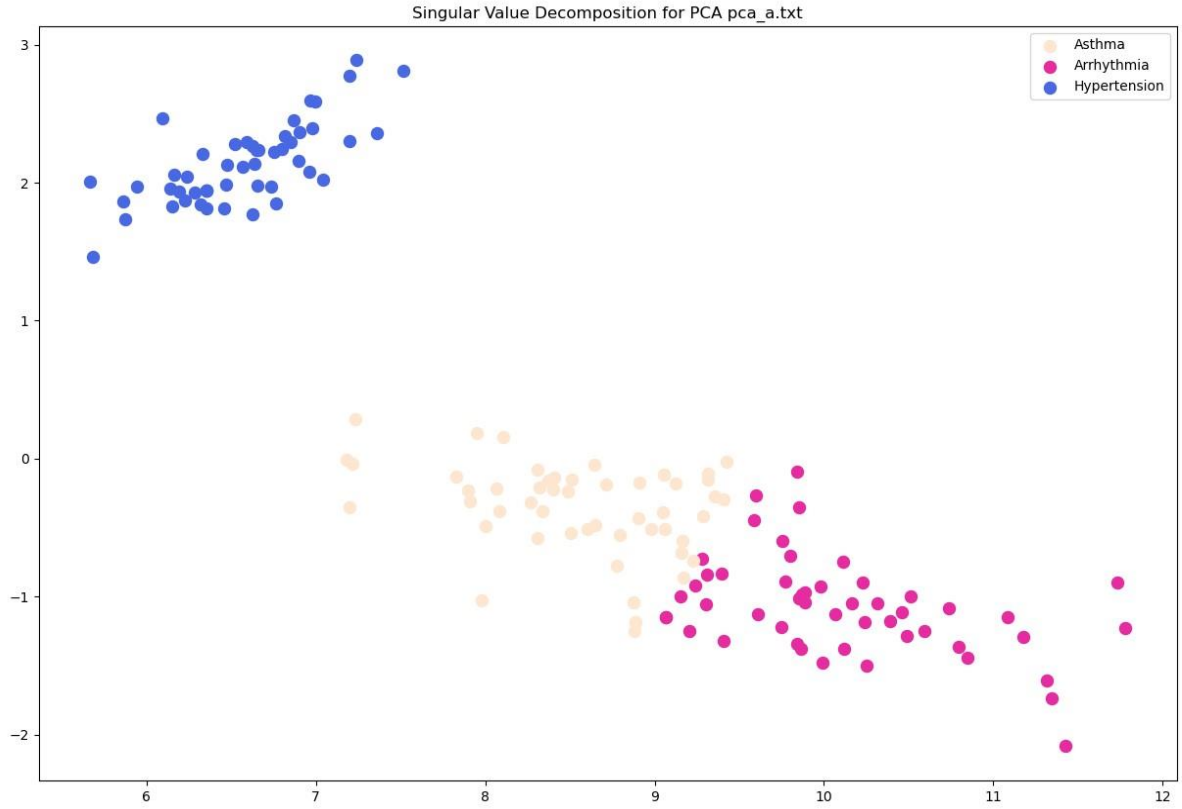
Implementation:

We have used "*TruncatedSVD*" library to implement the SVD. The following lines of code implement the SVD.

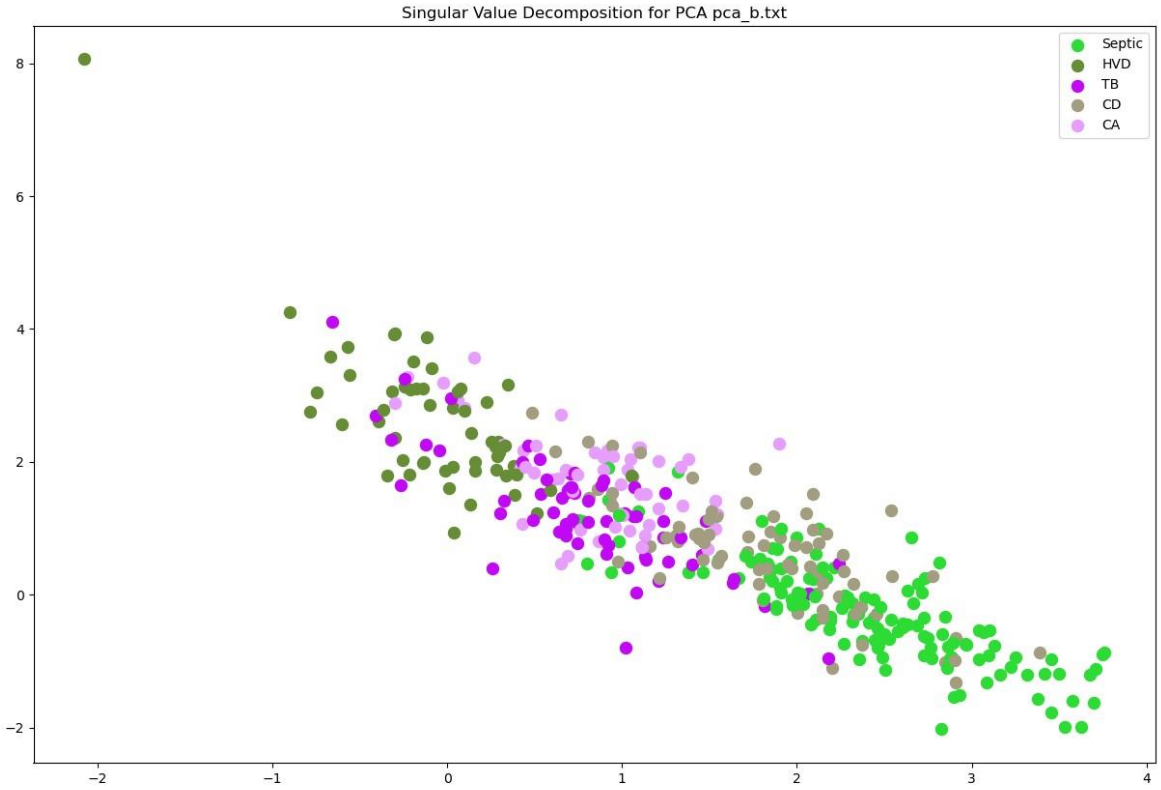
```
svd = TruncatedSVD(n_components=2)
svd.fit(attributes)
result = svd.transform(attributes)
```

The data in the result consists of reduced dimensional data points which is projected in the 2-D data space.

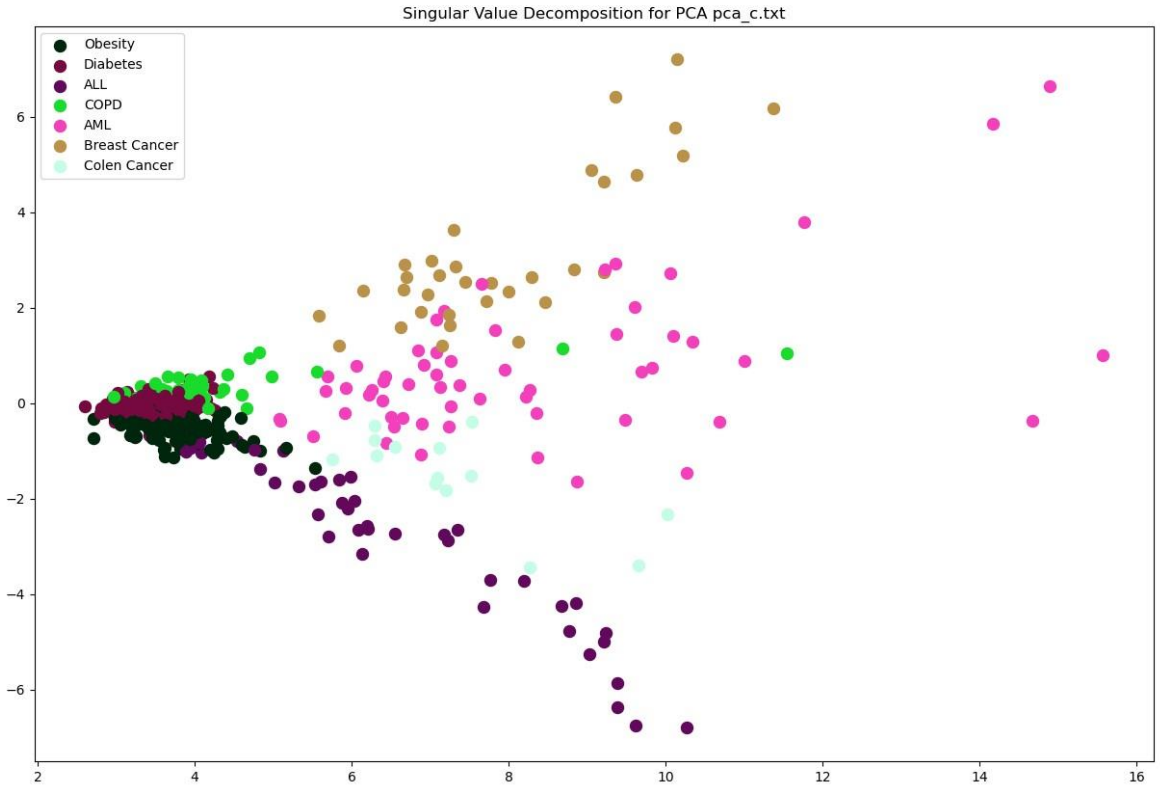
4. Implementing SVD on pca_a.txt dataset



5. Implementing SVD on pca_b.txt dataset



6. Implementing SVD on pca_c.txt dataset



tSNE:

t-Distributed Stochastic Neighbor Embedding (t-SNE) is an unsupervised, non-linear technique primarily used for data exploration and visualizing high-dimensional data.t-SNE gives us an intuition of how the data is arranged in a high dimensional space.

We have implemented tSNE using TSNE library from *sklearn.manifold* The following lines of code implement tSNE :

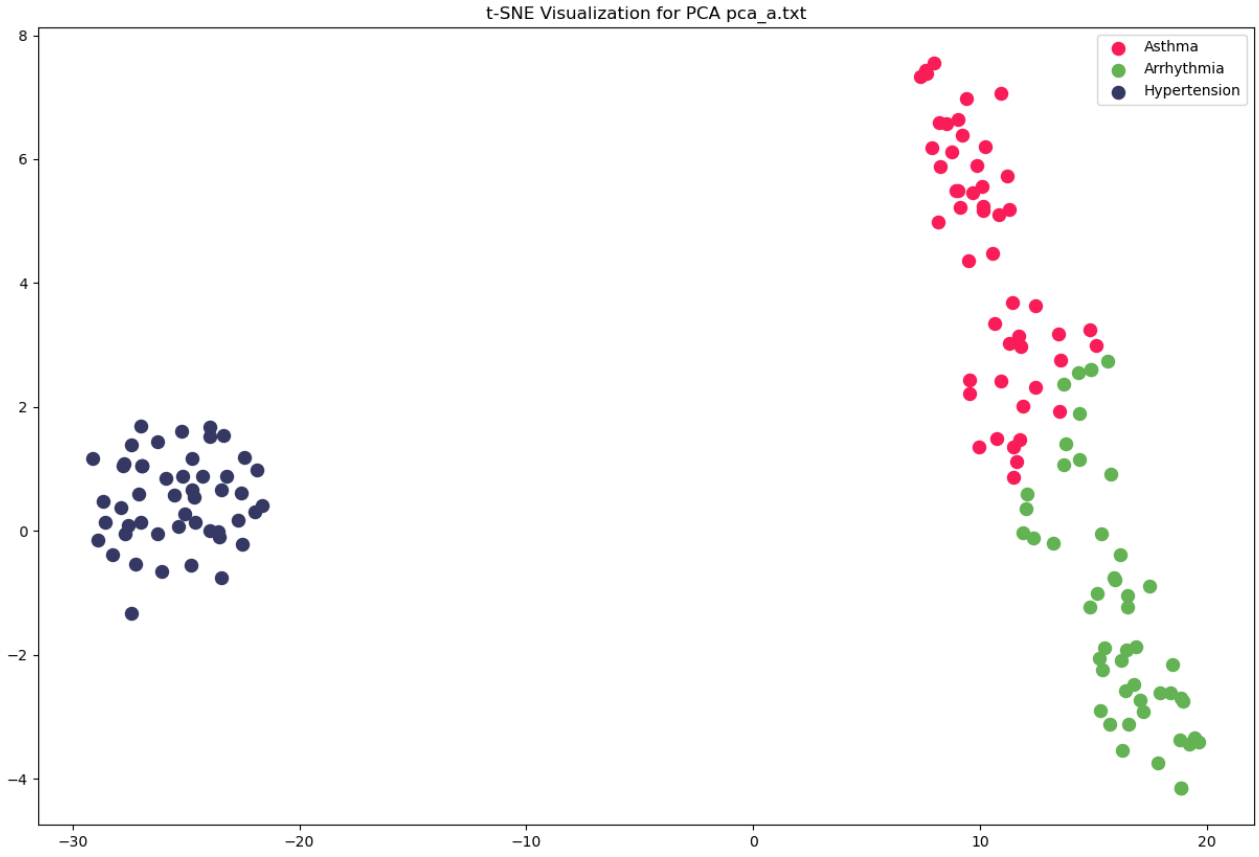
```
tsne = TSNE (n_components=2, init="pca")

result = tsne.fit_transform(attributes)
```

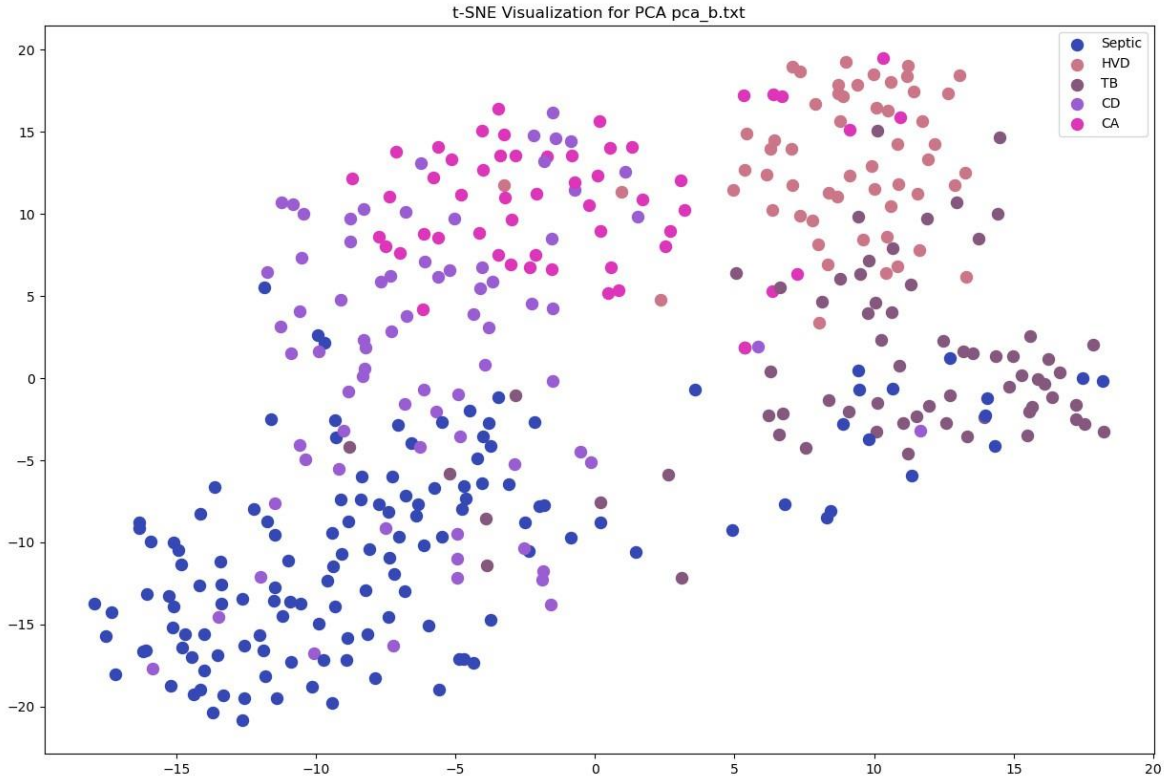
n_components: Determine the reduced dimensional space.

init: Initialization of embedding.

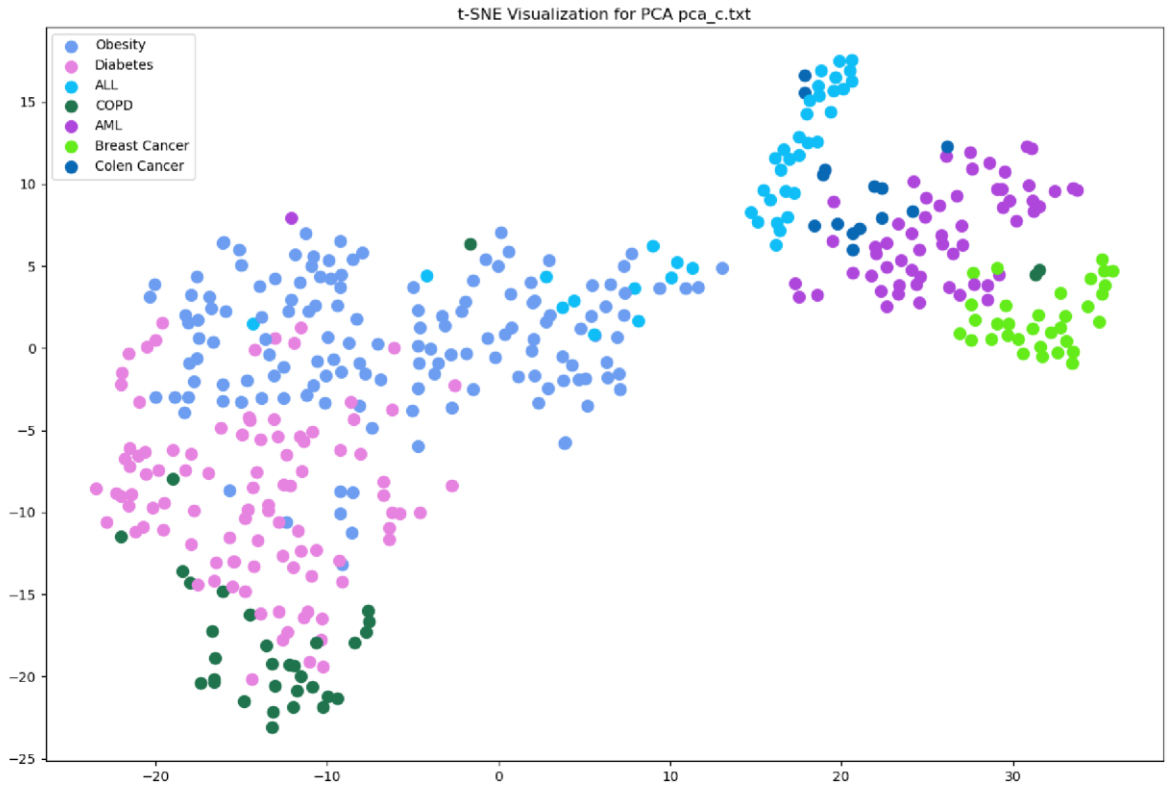
7. Implementing tSNE on pca_a.txt dataset



8. Implementing tSNE on pca_b.txt dataset



9. Implementing tSNE on pca_c.txt dataset



Result Analysis:

- PCA and SVD use linear methods of dimensionality reduction whereas t-SNE uses non-linear methods. Hence, the scatter plots of PCA and SVD look different from that of t-SNE.

- t-SNE is useful for visualization of high dimensional datasets. t-SNE preserves local similarities or small pairwise distances whereas PCA is concerned with preserving variance.

- Outliers are well handled in t-SNE compared to PCA and SVD.

- PCA is obtained by eigen value decomposition of covariance matrix and SVD uses matrix factorization technique. Hence, the plots of PCA and SVD look similar.

References:

- https://medium.com/@jonathan_hui/machine-learning-singular-value-decompositionsvd-principal-component-analysis-pca-1d45e885e491

- https://towardsdatascience.com/an-introduction-to-t-sne-with-python-example-5a3a293108d1

- https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html