Data Mining and Bioinformatics

CSE601

PROJECT 1

Devavrat D Raikar
Devavrat
50291115

Divya JeevanKumar Sanghvi
Divyajee
50288057

Dimensionality Reduction

The aim of the project was to implement dimension reduction of the given dataset by using three methods. They are Principal component Analysis, Singular value Decomposition and t-Distributed Stochastic Neighbor Embedding. PCA is used to project datasets with correlated attributes as a single attribute by reducing the dimension of the dataset. A dataset with lot of noises and ambiguities are bad way for visualization and analysis for predictions.

The transformation maps a data vector from an original space p to new space p variables which are uncorrelated over the dataset. Such dimensionality reduction can be a very useful step for visualizing and processing high-dimensional datasets, while still retaining as much of the variance in the dataset as possible.

Principal Component Analysis (PCA) combines different attributes linearly and tries to capture the original variance of the given data. The newly generated attributes are called Principal components, such that the first principal component holds the largest variance, the second principal component holds the second largest variance and so on.

Flow of PCA algorithm:

Language: R programming
Libraries: Rtsne, ggfortify

Steps:

- Read the .txt file from the command line with passing confidence and support parameters.
- Read the 1 to n-1 columns from the data frame
- And the last column is used as the Labels to the datasets. It is obtained by reversing the data frame and accessing the first column of reversed data frame.
- The data is scaled using the scale function in R that is calculating mean of the each attribute and subtracting from individual records of the attribute column. **X=x-mean**
- Calculated the covariance of the scaled data. We used cov() function in R to calculate the covariance. The formulae for the calculating covariance is:

  S = [1 / (Total number of rows)] * X * X T
- Eigen value of the covariance matrix is calculated using Eigen() function

- The Eigen values which was computed in the last step was used to extract two Eigen vectors from its matrix to form principal components.
- Eigen vectors are arranged increasing order of the variance
- The Eigen value vector with the input data is multiplied to obtain principal components of the data frame.
- The principal components along with the Labels are combined in the single data frame.
- ggplot() was used to plot the Principal components and Labels
- Again, each point is colored based on the disease it represents in the provided data (we have different colors for different diseases)

PCA vs SVD:

PCA and SVD have the same application of dimensionality reduction and in visualization. SVD and PCA uses the same technique of Eigen vector and Eigen values producing the information with highest variance with low dimension. The covariance of both techniques is almost same.

PCA:

$$XX^T = WDW^T$$

SVD:

$$XX^T = U\Sigma^2U^T$$

D and Σ are some constant values. W and U represents a data frame

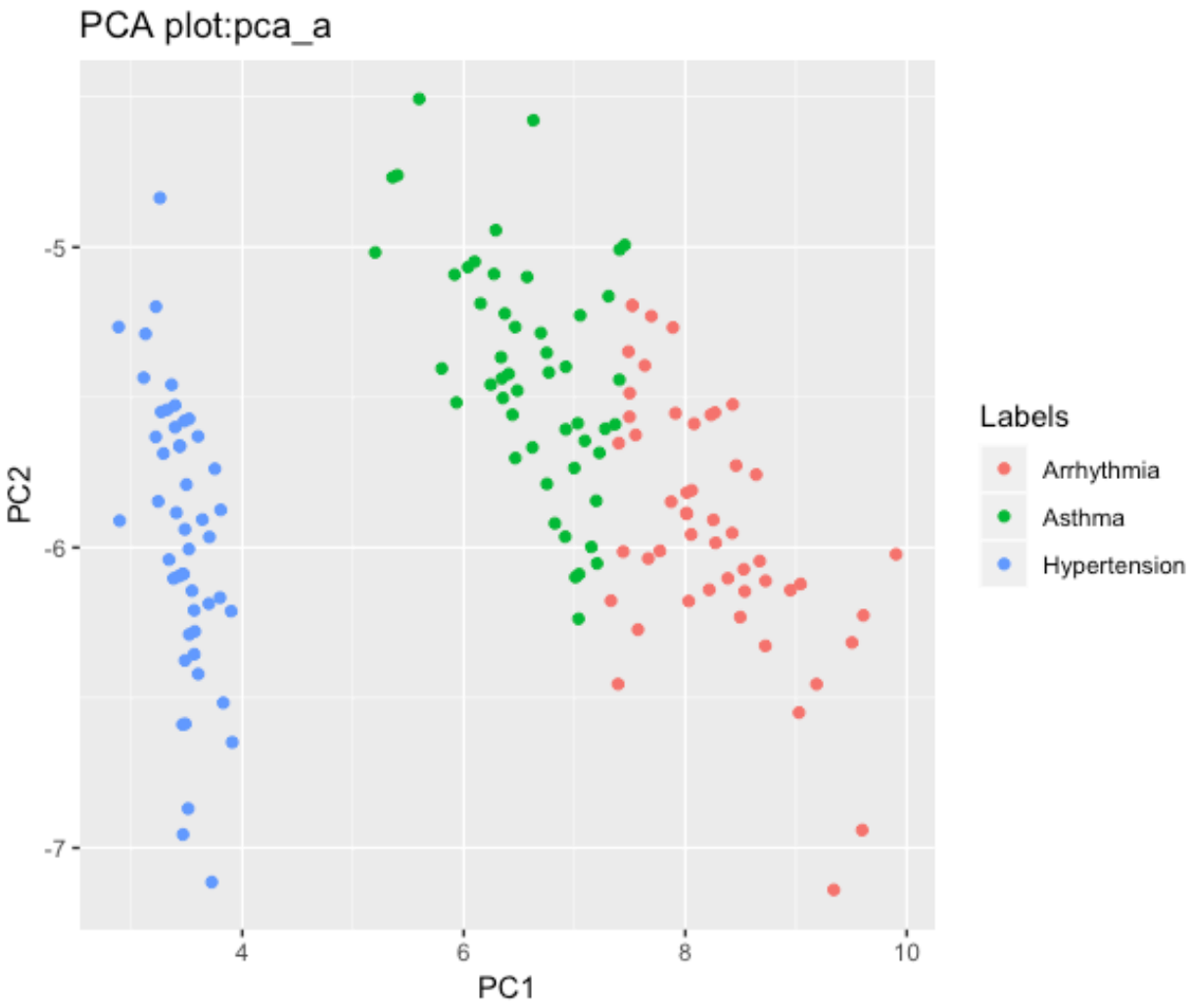
PCA vs t-SNE:

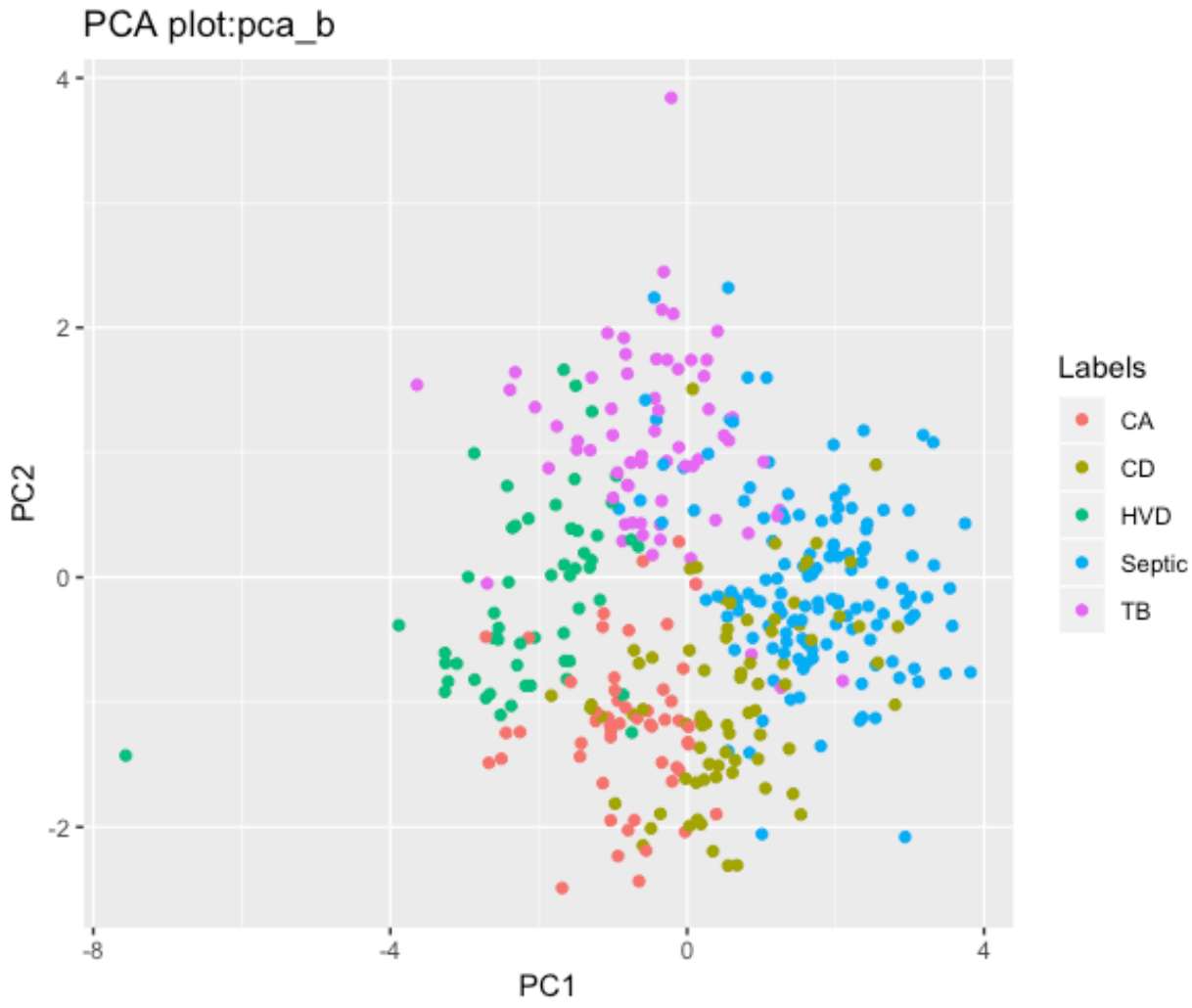
PCA used formula to calculate variance and Eigen vectors to find principal components of the datasets. It uses correlation between some of the dimensions and provides a minimum number of variables that keeps the maximum amount of variation or information about how the original data is distributed.

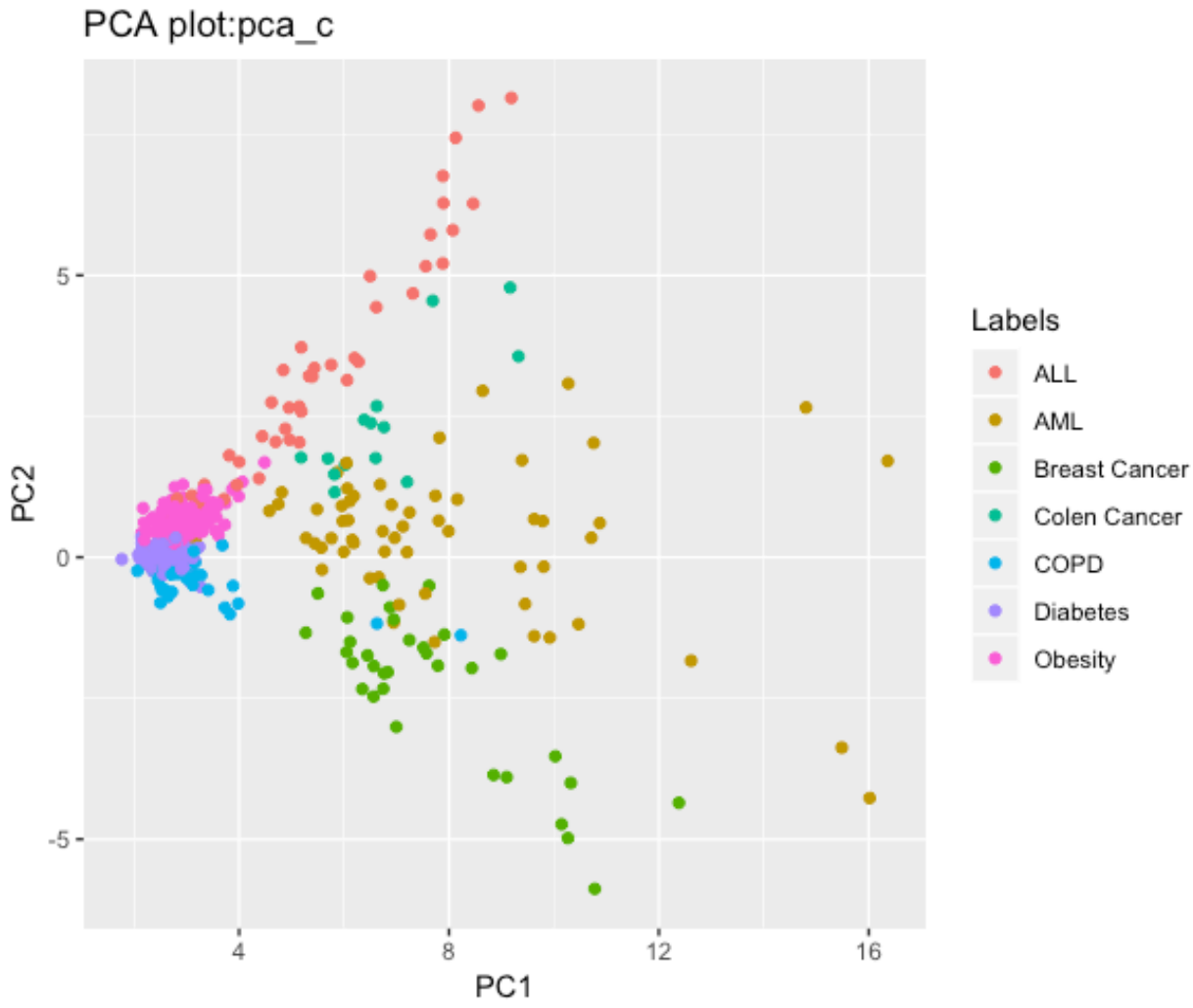
t-SNE:

- We used the t-SNE package from the library(tSNE)
- The parameters in t-SNE are tuned to produce the best clusters
- We set number of iterations to 1000 so that it optimizes the cluster
- Perplexity was set between 30, 40, 50 and 60 and it produced the best result at perplexity = 30
- t-SNE minimizes the divergence between two distributions: a distribution that measures pairwise similarities of the input objects and a distribution that measures pairwise similarities of the corresponding low-dimensional points

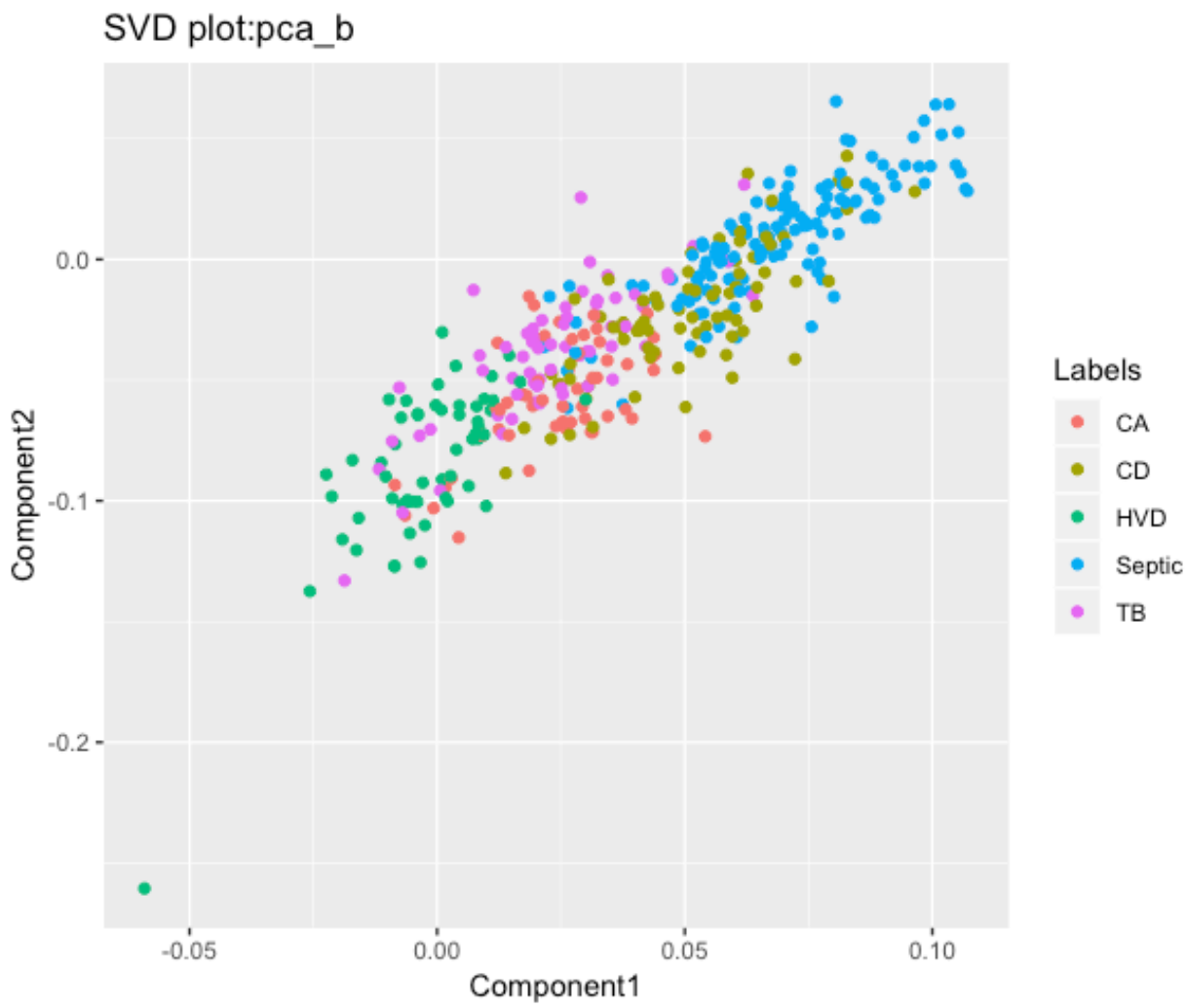
Results:
There are 9 plots totally with 3 for each method.

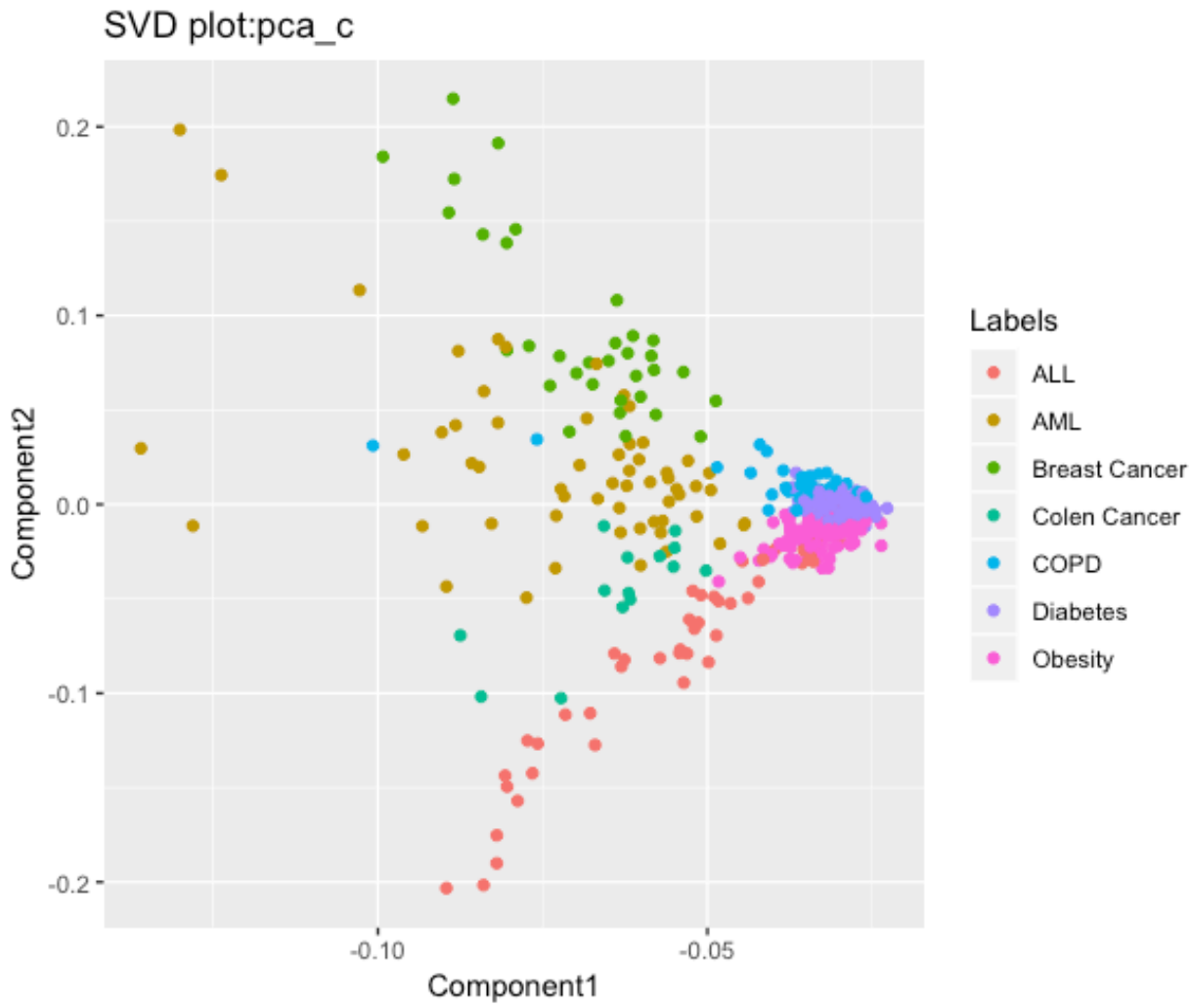


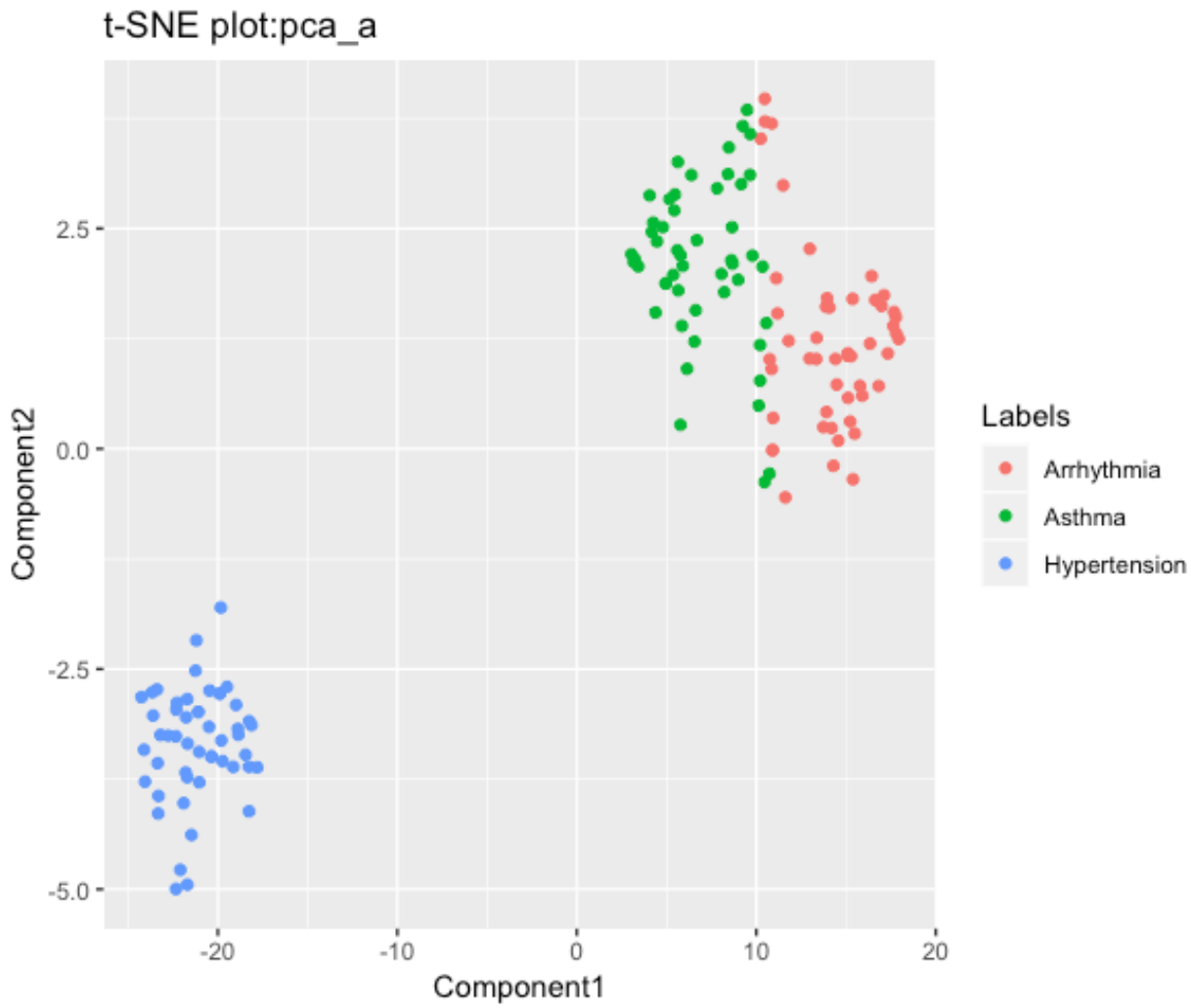

PCA plot:pca_b
4
2
0
-2
PC2
-8
-4
0
4
PC1
Labels
CA
CD
HVD
Septic
TB

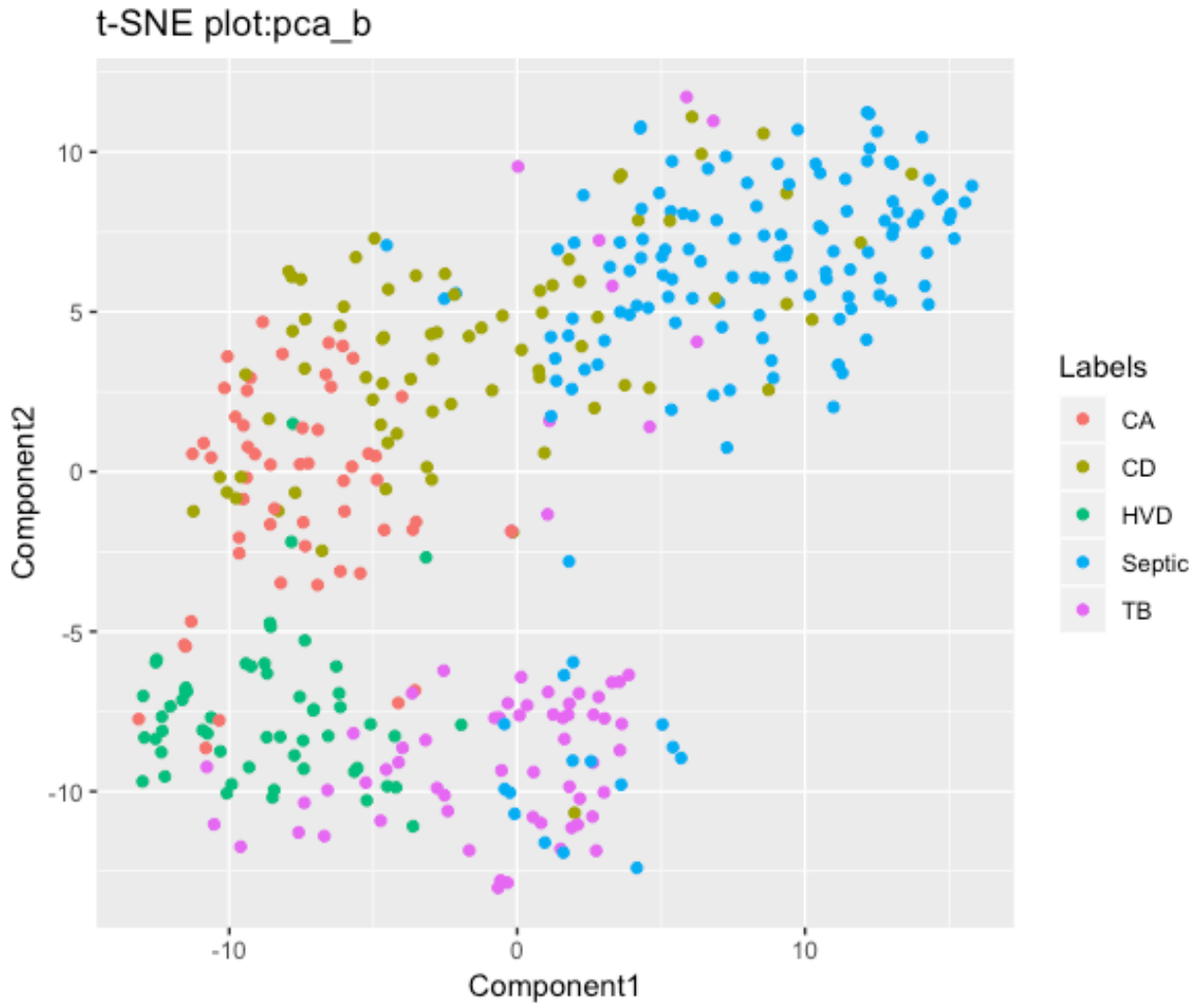

PCA plot:pca_c
PC2
PC1
5
0
-5
4
8
12
16
Labels
ALL
AML
Breast Cancer
Colen Cancer
COPD
Diabetes
Obesity

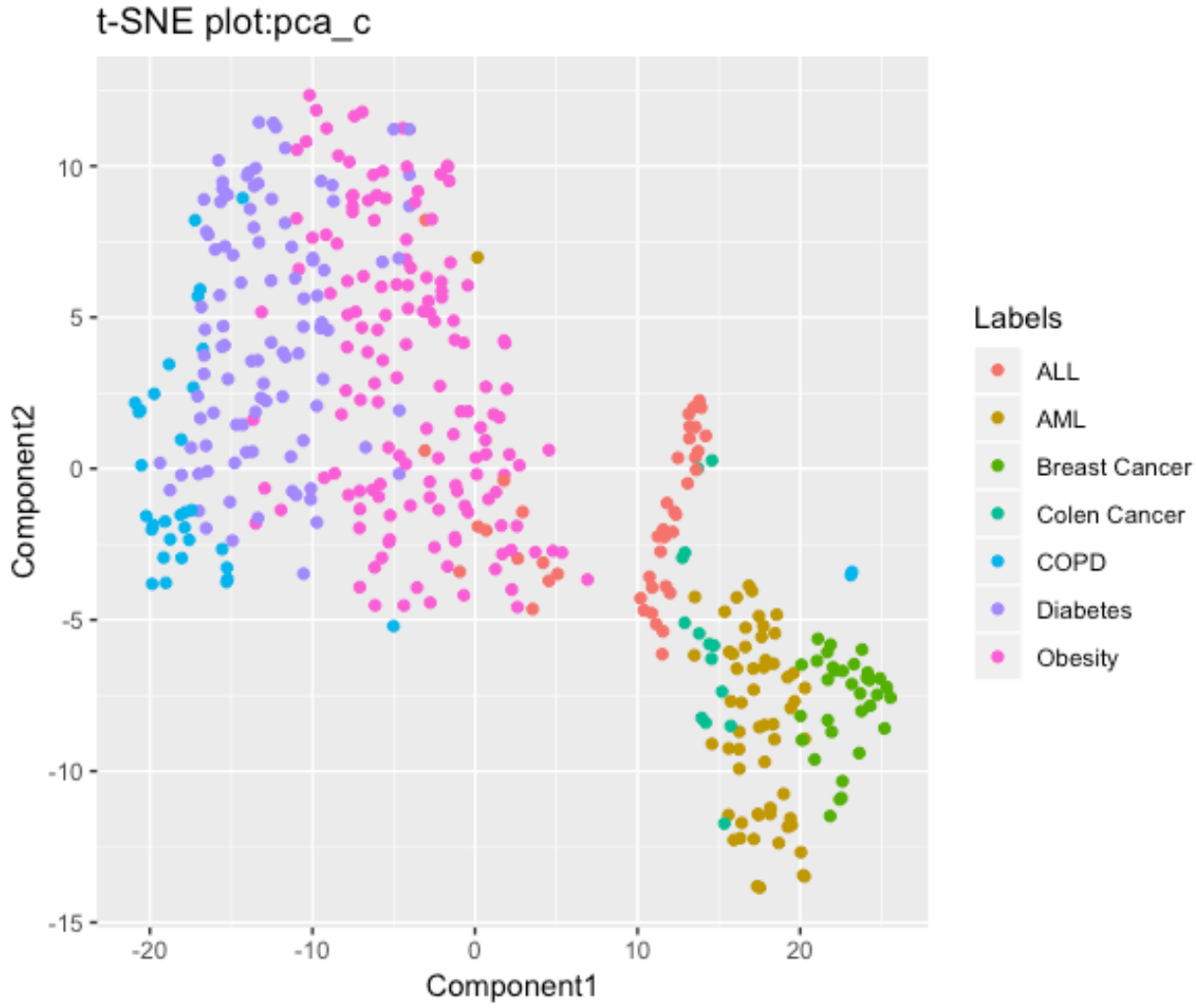

SVD plot:pca_a
0.1
0.0
-0.1
Component2
-0.11
-0.10
-0.09
-0.08
-0.07
-0.06
Component1
Labels
Arrhythmia
Asthma
Hypertension


SVD plot:pca_b
0.0
-0.1
-0.2
Component2
-0.05
0.00
0.05
0.10
Component1
Labels
CA
CD
HVD
Septic
TB


SVD plot:pca_c
0.2
0.1
0.0
-0.1
-0.2
Component2
-0.10
-0.05
Component1
Labels
ALL
AML
Breast Cancer
Colen Cancer
COPD
Diabetes
Obesity


t-SNE plot:pca_a
2.5
0.0
-2.5
-5.0
Component2
-20
-10
0
10
20
Component1
Labels
Arrhythmia
Asthma
Hypertension


t-SNE plot:pca_b
Component2
Component1
10
5
0
-5
-10
-10
0
10
Labels
CA
CD
HVD
Septic
TB


t-SNE plot:pca_c
Component2
Component1
10
5
0
-5
-10
-15
-20
-10
0
10
20
Labels
ALL
AML
Breast Cancer
Colen Cancer
COPD
Diabetes
Obesity

Code Snippet:

```
            ################## pca_a ################

# Choose the data file
inputData <- read.delim("pca_a.txt", header = FALSE)

# Extract the numeric data into a separate data frame
inputNumericData <- inputData[,1:ncol(inputData)-1]

# Extract the names of the Diseases in a seperate list
inputLabels <- as.data.frame((rev(inputData)[1]))
colnames(inputLabels) <- c("Labels")


# Normalise the data
scaledData <-  as.data.frame(scale(inputNumericData, center = TRUE, scale = FALSE))

# Find the covariance matrix
covMatrix <- cov(scaledData)

# Find the eigen vectors and eigen values
eigenDecomp <- eigen(covMatrix)

# Extract the first 2 loadings of Eigen Vectors
vectorLoadings <- eigenDecomp$vectors[,1:2]
vectorLoadings <- -vectorLoadings

# Calculate the Pricipal Component Scores
PC1 <- as.matrix(inputNumericData) %*% vectorLoadings[,1]
PC2 <- as.matrix(inputNumericData) %*% vectorLoadings[,2]

PC <- data.frame(Diseases = inputLabels, PC1, PC2)

# Plot Principal Components for each State
quartz()
p<-ggplot(PC,aes(x=PC1,y=PC2,color=Labels ))
p<-p+geom_point()
p
```