Data Mining and Bioinformatics

PCA

Komlan Aziagba 50158080

Saeb Ragani Lamooki 50205221

Implementation of PCA Algorithm

We imported the data files and extracted each line as a list. This list was split by tabs to extract each feature as a string and the strings were then converted into real numbers to create the data matrix x. However, the last column which included the list of disease was stored in a separate vector. We then computed the mean vector by taking the mean of each feature. Next using ‘tile’ command we created a mean matrix of the same dimension as the original data matrix with each feature mean repeated in the columns. We subtracted the mean matrix from the original data matrix to generate the adjusted matrix (x′).

To calculate the covariance matrix we multiplied the transpose of the x′ matrix by the x′ matrix and divided the result by n-1 where n is the number of features. Then using ‘linalg.eig’ function in ‘numpy’ library, we calculated the eigen vectors of the covariance matrix. We then multiplied the original data matrix by the two eigen vectors associated with the highest eigenvalues to reduce the dimension of our data to two.

The list of diseases were appended to an empty list each time a new value was read for that. When visualizing the results, in order to create a different color for each disease, a new vector named labels was created by assigning the index value of each disease name. the labels vector was then used for color in the scatter plot.

Implementation of SVD

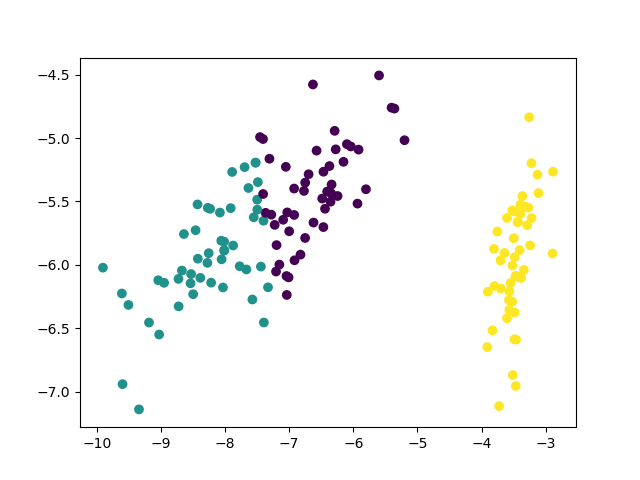
We have created the data matrix with all the features except for the las column similar to previous part. We also created the labels vector similarly. Using ‘np.linalg.svd’ function from ‘numpy’ library, we reduced the dimension of the original data. The singular value decomposition (SVD) method, when applied on the original data matrix generates the singular value vector [U, S, V] where S is a diagonal vector such that U.S.V = X.

Implementation of t-SNE

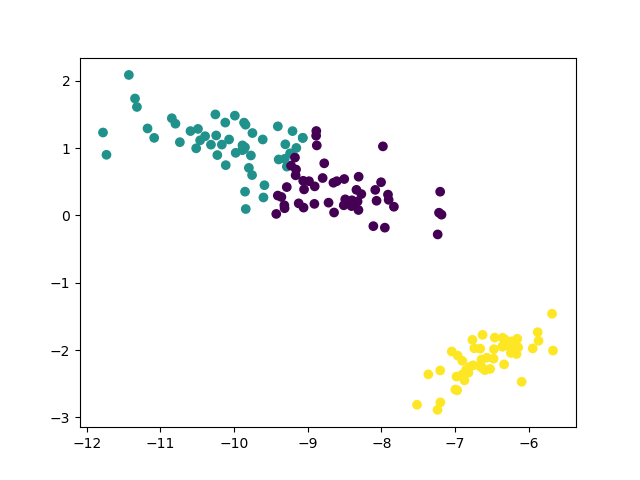
The data matrix and the labels vector were created similar to previous parts. The tSNE algorithm is based on probability distribution where objects are more probable to be picked from similar pairs as compared to dissimilar pairs by calculating their Euclidean distance. This method is generally used for the purpose of visualization of data with high dimensions by decreasing the dimension to 2 or 3. We used the TSNE function as ‘TSNE(n\_components=2).fit\_transform’ from’sklearn’ package.

Compare the Scatter plots from the Three methods

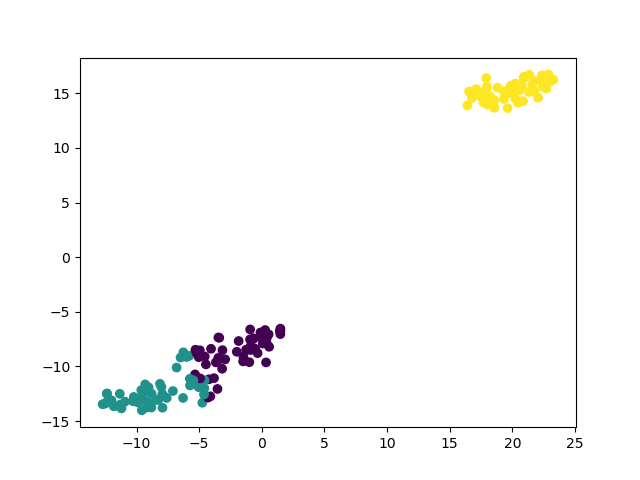
1. pca\_a.txt



PCA\_a

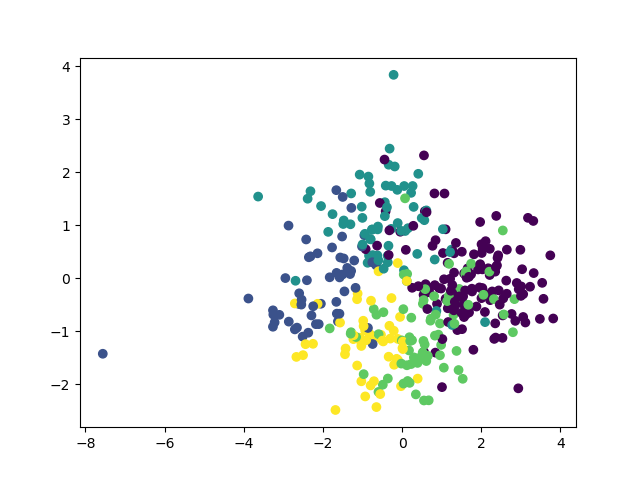
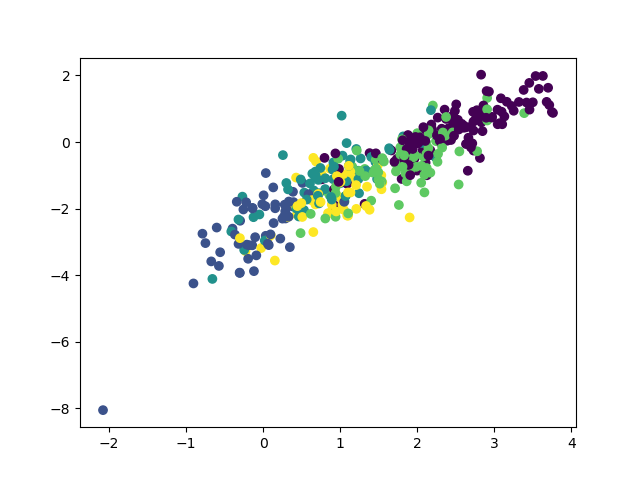
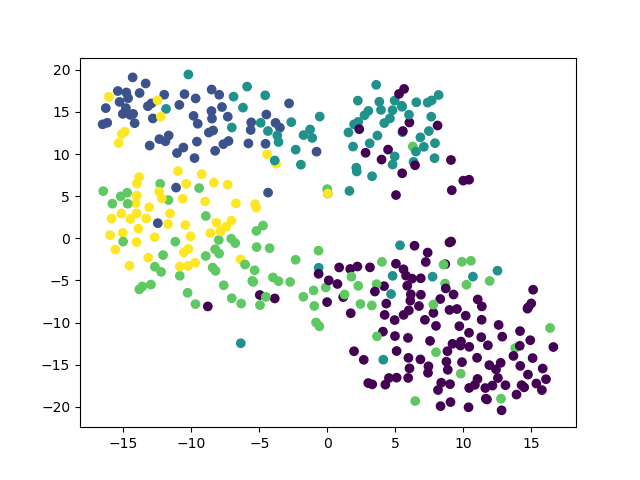


SVD\_a



tSNE\_a

1. pca\_b.txt

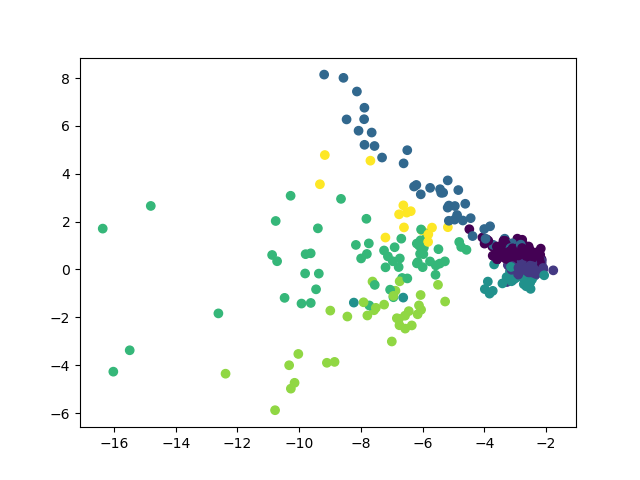
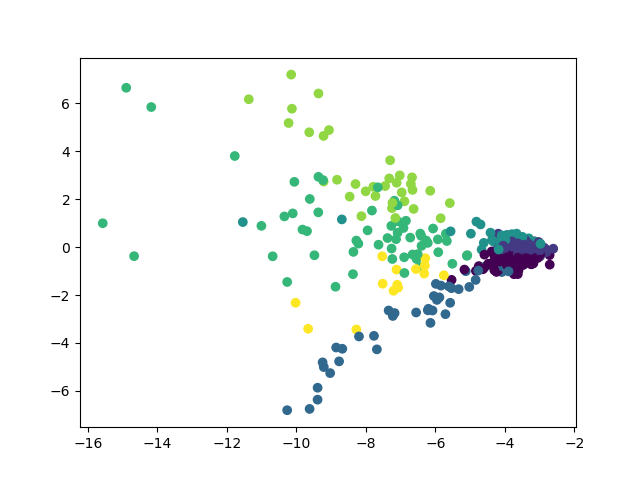
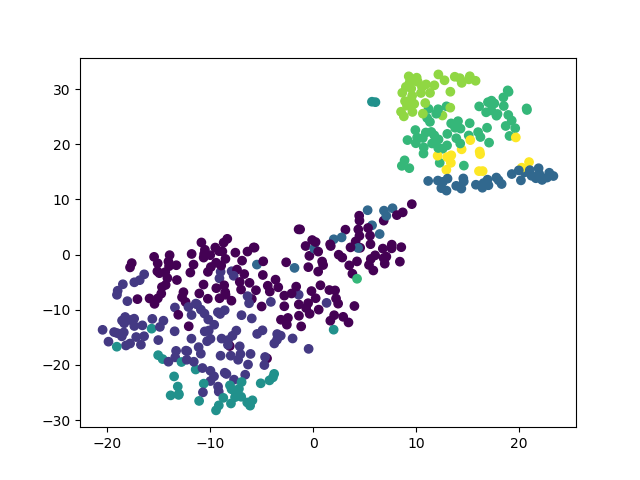


tSNE\_b

SVD\_b

PCA\_b

1. pca\_b.txt



tSNE\_c

SVD\_c

PCA\_c