[List of Figures 2](#_Toc177483622)

[DATASET 3](#_Toc177483623)

[Principal Component Analysis (PCA) 3](#_Toc177483624)

[Definition 3](#_Toc177483625)

[How PCA works? [2] 3](#_Toc177483626)

[Applications 3](#_Toc177483627)

[Visualizing the given data using PCA 4](#_Toc177483628)

[Calculation of first two principal components 4](#_Toc177483629)

[Plotting the components in a 2D space 4](#_Toc177483630)

[Visualizing distribution and statistics using violin plot 5](#_Toc177483631)

[t-distributed Stochastic Neighbor Embedding (t-SNE) 5](#_Toc177483632)

[Definition [4] 5](#_Toc177483633)

[How t-SNE works? [5] 6](#_Toc177483634)

[Applications 6](#_Toc177483635)

[Visualizing the given data using t-SNE 6](#_Toc177483636)

[Calculating the t-SNE values 6](#_Toc177483637)

[Plotting t-SNE values in 2D space 7](#_Toc177483638)

[Visualizing distribution and statistics using violin plot 7](#_Toc177483639)

[Conclusion 8](#_Toc177483640)

[REFERENCES 9](#_Toc177483641)

# List of Figures

[Figure 1: Clusters formed by PCA 4](#_Toc177483199)

[Figure 2: Violin plot for PC1 5](#_Toc177483200)

[Figure 3: Violin plot for PC2 5](#_Toc177483201)

[Figure 4: Clusters formed by t-SNE 7](#_Toc177483202)

[Figure 5: Violin plot for t-SNE1 8](#_Toc177483203)

[Figure 6: Violin plot for t-SNE2 8](#_Toc177483204)

# Dataset

The provided CSV data file consists of 2529 rows and 12311 columns. The first column represents the patient, and the last column labeled “Class” consists of the symbols which represent cancer types: e.g. “KRIC” for Kidney Renal Clear Cell Carcinoma. The other column names are ensemble ids which can be mapped to the corresponding gene names using “mygene” library in Python. Thus, the data altogether consists of the expression levels for 12,309 long non-coding RNA (lncRNA) for the patients with five different types of cancer.

# Principal Component Analysis (PCA)

## Definition

Principal Component Analysis (PCA) is a linear dimensionality reduction technique with applications in exploratory data analysis, visualization, and data processing. The data is linearly transformed onto a new coordinate system such that the directions (principal components) capturing the largest variation in the data can be easily identified. [1]

## How PCA works? [2]

PCA summarizes the information content of large datasets into a smaller set of uncorrelated variables known as principal components. These principal components are linear combinations of the original variables that have the maximum variance compared to other linear combinations. These components capture as much information from the original dataset as possible.

This statistical technique involves both linear algebra and matrix operations, and it transforms the original dataset into a new coordinate system that is structured by the principal components. The eigenvectors and eigenvalues from the covariance matrix that underpin the principal components allow for the analysis of these linear transformations.

Imagine you have mapped out a dataset with multiple features, resulting in a multi-dimensional scatterplot. *Eigenvectors* provide the direction of variance in the scatterplot. *Eigenvalues* are the coefficients of the eigenvectors; these denote the importance of this directional data. Therefore, a high eigenvalue means that the corresponding eigenvector is more critical. Since principal components represent the directions of maximum variance in the data, they are also the eigenvectors of the covariance matrix.

## Applications

Principal Component Analysis has wide variety of applications including but not limited to:

1. Image Compression: PCA reduces image dimensionality while retaining essential information. It helps create compact representations of images, making them easier to store and transmit.
2. Data Visualization: PCA helps to visualize high-dimensional data by projecting it into a lower-dimensional space, such as a 2D or 3D plot. This simplifies data interpretation and exploration.
3. Noise Filtering: PCA can remove noise or redundant information from data by focusing on the principal components that capture the underlying patterns.

## Visualizing the given data using PCA

### Calculation of first two principal components

PCA class from scikit-learn library in python can be used to reduce the dimensions to required number of components i.e. n\_components, and fit\_transform method can be used to calculate the principal components. The PCA class from scikit-learn uses the LAPACK implementation of the full SVD or a randomized truncated SVD by the method of Halko et al. 2009, depending on the shape of the input data and the number of components to extract. [3]

### Plotting the components in a 2D space

Fig. 1 shows the scatter plot formed by the first and second principal values of the given data shows five discernible clusters for each of the cancer types; however, LUAD and LUAD classes have overlaps, probably because of being two sub type of the lung cancer. Nevertheless, the same cancer type samples are grouped together closely in the same cluster.

A graph of different colored dots

Description automatically generated

Figure 1: Clusters formed by PCA

### Visualizing distribution and statistics using violin plot

The violin plots in fig. 2 and fig. 3 represent the first and second principal components. Each of

the violin demonstrates the following statistics about the data:

* Minimum, and maximum represented by the top and the bottom of the violin
* Median represented by the white point inside the violin
* First and third quartiles represented by the vertical rectangular box inside the violin
* Distribution of the values represented by the body of the violin, the wider the body is the higher number of values around it.

|  |  |
| --- | --- |
| A graph of different colored rhombuses  Description automatically generated  Figure 2: Violin plot for PC1 | A graph of different colored shapes  Description automatically generated  Figure 3: Violin plot for PC2 |

# t-distributed Stochastic Neighbor Embedding (t-SNE)

## Definition [4]

t-distributed stochastic neighbor embedding (t-SNE) is a statistical method for visualizing high-dimensional data by giving each datapoint a location in a two or three-dimensional map. It is based on Stochastic Neighbor Embedding originally developed by Geoffrey Hinton and Sam Roweis, where Laurens van der Maaten and Hinton proposed the t-distributed variant. It is a nonlinear dimensionality reduction technique for embedding high-dimensional data for visualization in a low-dimensional space of two or three dimensions. Specifically, it models each high-dimensional object by a two- or three-dimensional point in such a way that similar objects are modeled by nearby points and dissimilar objects are modeled by distant points with high probability.

The t-SNE algorithm comprises two main stages. First, t-SNE constructs a probability distribution over pairs of high-dimensional objects in such a way that similar objects are assigned a higher probability while dissimilar points are assigned a lower probability. Second, t-SNE defines a similar probability distribution over the points in the low-dimensional map, and it minimizes the Kullback–Leibler divergence (KL divergence) between the two distributions with respect to the locations of the points in the map. While the original algorithm uses the Euclidean distance between objects as the base of its similarity metric, this can be changed as appropriate. A Riemannian variant is UMAP.

## How t-SNE works? [5]

The t-SNE algorithm finds the similarity measure between pairs of instances in higher and lower dimensional space. After that, it tries to optimize two similarity measures. It does all of that in three steps.

1. t-SNE models a point being selected as a neighbor of another point in both higher and lower dimensions. It starts by calculating a pairwise similarity between all data points in the high-dimensional space using a Gaussian kernel. The points that are far apart have a lower probability of being picked than the points that are close together.
2. Then, the algorithm tries to map higher dimensional data points onto lower dimensional space while preserving the pairwise similarities.
3. It is achieved by minimizing the divergence between the probability distribution of the original high-dimensional and lower-dimensional. The algorithm uses gradient descent to minimize the divergence. The lower-dimensional embedding is optimized to a stable state.

The optimization process allows the creation of clusters and sub-clusters of similar data points in the lower-dimensional space that are visualized to understand the structure and relationship in the higher-dimensional data.

## Applications

t-SNE has been used for visualization in a wide range of applications, including genomics, computer security research, natural language processing, music analysis, cancer research, bioinformatics, geological domain interpretation, and biomedical signal processing.

## Visualizing the given data using t-SNE

### Calculating the t-SNE values

TSNE class from scikit-learn python library can be used to reduce the dimension of data using t-SNE algorithm. The following hypermeters can be configured to optimize the result:

* n\_components: int
* perplexity: float, default=30
* early\_exaggeration: float, default=12
* learning\_rate: float or “auto”, default=”auto”
* max\_iter: int, default=1000
* n\_iter\_without\_progress: int, default=300
* min\_grad\_norm: float, default=1e-7

### Plotting t-SNE values in 2D space

Looking at the scatter plot of the t-SNE values for the given cancer type in fig. 4, the clusters formed are more discernible than the clusters from the PCA. The data for the same cancer type forms a tightly dense cluster, whereas the distance from one cluster to another is much higher compared to the PCA. However, there are still some overlaps among the LUAD and LUSC samples for the same reason i.e. because of being two sub-types of the lung cancer.

A graph with different colored spots

Description automatically generated

Figure 4: Clusters formed by t-SNE

### Visualizing distribution and statistics using violin plot

Fig. 5 and fig. 6 demonstrate the violin plots for t-SNE1 and t-SNE2 with the same statistics as shown for the violin plots for PCA. However, the values for one cluster are highly dense around the same area whereas values for other clusters are farther compared to that of PCA. Additionally, long wigs in the violins articulate that the outliers are pushed far from the cluster, which can help to remove the outliers to improve the data quality.

|  |  |
| --- | --- |
| A graph of different colored shapes  Description automatically generated  Figure 5: Violin plot for t-SNE1 | A graph of different colored shapes  Description automatically generated  Figure 6: Violin plot for t-SNE2 |

# Conclusion

The assignment gave us hands-on experience with RNA data and two dimensionality reduction techniques: PCA and t-SNE. Based on the clusters formed from the PCA and t-SNE values, it can be concluded that t-SNE performs better than PCA for the given data. In fact, t-SNE values provide visually more discernible clusters compared to PCA primarily because t-SNE is a non-linear algorithm such that the related values are kept close together and unrelated values are pushed farther.

Additionally, the exercise allowed us to have hands-on experience with plotting library in python to plot different type of charts, specifically scatter plots and violin plots. The violin plot can be thought of as a more advanced version of a box plot –also known as five-points plot—which additionally shows the density or the distribution.

Comparing the violin plots from both the PCA and t-SNE, it can again be concluded that t-SNE performs better as the outliers are easily distinguishable, and it can further aid to remove the outliers from a data set as part of data processing/cleaning.

# References

|  |  |
| --- | --- |
| [1] | "Principal Component Analysis," 03 July 2024. [Online]. Available: https://en.wikipedia.org/wiki/Principal\_component\_analysis. |
| [2] | "What is principal component analysis (PCA)?," IBM, 08 December 2023. [Online]. Available: https://www.ibm.com/topics/principal-component-analysis. [Accessed 16 September 2024]. |
| [3] | "PCA -- scikit-learn," scikit-learn developers, [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html. [Accessed 17 September 2024]. |
| [4] | "t-distributed stochastic neighbor embedding," Wikipedia, 30 August 2024. [Online]. Available: https://en.wikipedia.org/wiki/T-distributed\_stochastic\_neighbor\_embedding#:~:text=A%20Riemannian%20variant%20is%20UMAP,interpretation%2C%20and%20biomedical%20signal%20processing.. [Accessed 16 September 2024]. |
| [5] | J. Hoare, "How t-SNE works and Dimensionality Reduction," [Online]. Available: https://www.displayr.com/using-t-sne-to-visualize-data-before-prediction/. [Accessed 17 September 2024]. |