Written Report of Module 2 - Genomics & High Dimensional Data

Tien Minh Dam {edX: @damminhtien}

25 June 2024

Problem 2: Larger unlabeled subset

Now we will work with the larger, unlabeled subset in 'p2_unsupervised'. This dataset is has not been processed, so you should process using the same log transform as in Problem 1.

Part 1: Visualization

A scientist tells you that cells in the brain are either excitatory neurons, inhibitory neurons, or non-neuronal cells. Cells from each of these three groups server different functions within the brain. Within each of these three types, there are numerous distinct sub-types that cell can be, and sub-types of the same larger class can serve similar functions. Your goal is to produce visualizations which show how the scientist's knowledge reflects in the data.

As in Problem 1, we recommend using PCA before running T-SNE or clustering algorithms, for quality and computational reasons.

1. *(3 points) Provide at least one visualization which clearly shows the existence of 3 main brain cell types as described by the scientist, and explain how it shows this. Your visualization should support the idea that cells from different groups can differ greatly.*

   Answer. In this question, the same steps in Problem 1 was reproduced. Firstly, I transformed the data by the log_transforms function $y = log_2(x + 1)$. After that, the data was projected into lower dimensional space by the PCA (n_components = 2), MDS (n_components = 2) and T-SNE (n_components = 2) on X_PCA (X reduced dimension by PCA) algorithms.

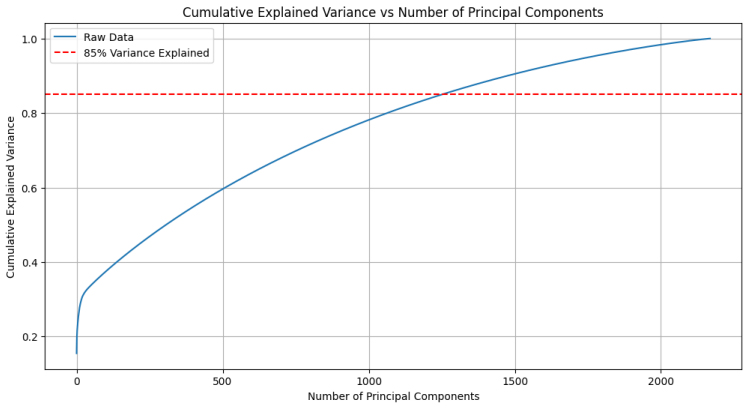

Figure 1: Number of PCs needed to explain 85% variance is 1253; Variance explained by the first PC is 0.15

The top two PCs was found, and I made scatterplots of the projections of the data onto these two PC's.

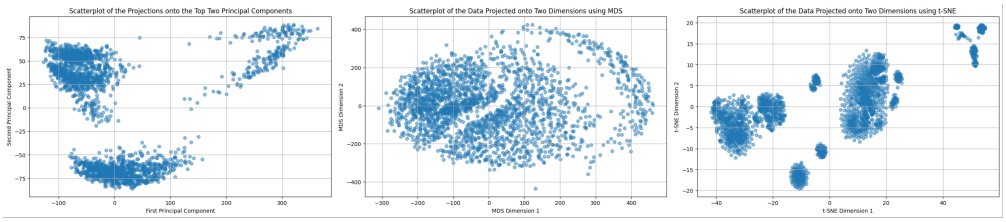

Figure 2: Scatterplot of the first and second coordinates of the data

Finally, the K-Means Clustering Algorithm was applied on X_PCA (X reduced dimension by PCA) to find 3 clusters of neurons. *We can see different 3 kinds of neurons in each cluster on the plot.*

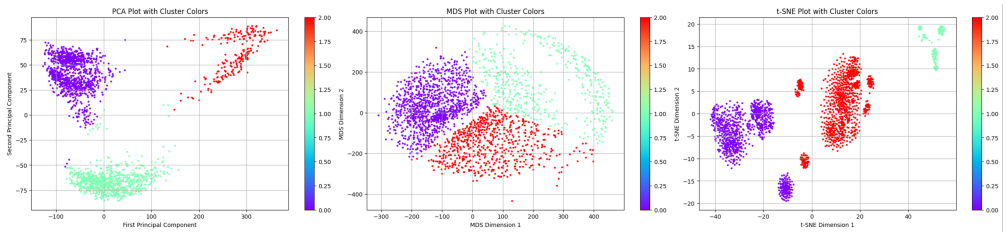

Figure 3: Three distinct groups of neurons are in red, purple and cyan colors

The plots shown clusters of points, where each cluster corresponded to one of the three main brain cell types: excitatory neurons, inhibitory neurons, and non-neuronal cells. Moreover, the clusters will be well-separated if there are significant differences between the groups. Cells from different groups will be positioned far apart, supporting the idea that they serve different functions within the brain.

□

2. *(4 points) Provide at least one visualization which supports the claim that within each of the three types, there are numerous possible sub-types for a cell. In your visualization, highlight which of the three main types these sub-types belong to. Again, explain how your visualization supports the claim.*

Answer. In this question, I made a deep investigation on how to find the sub-types of three given types efficiently. There are three different ways to archieve the goal of this question:

- METHOD 1 - Increase the number of cluster when applying the K-means clustering algorithms (the elbow can be used here to determine the best number of clusters): the simplest and straight forward approach but the new clusters can be not the same as the original
- METHOD 2 - Apply one more clustering algorithm on data in each given cluster: we need to use appropriate clustering algorithm to see the differences
- METHOD 3 - Use hierarchical clustering algorithm: this approach is very flexible to determine the number of cluster and the number of sub-cluster

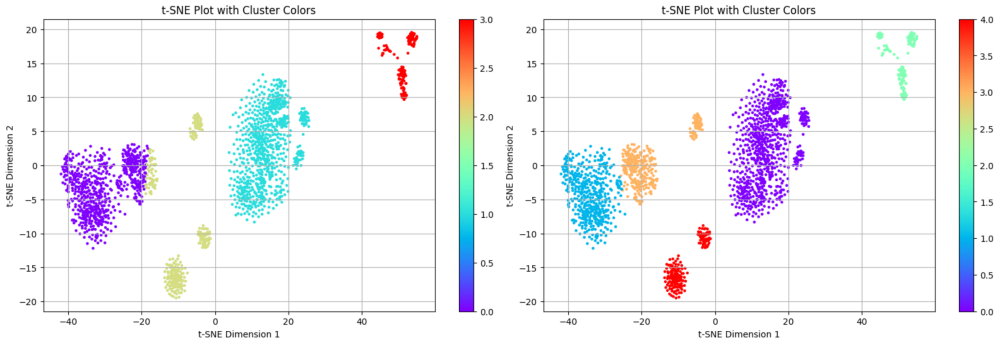

Figure 4: METHOD 1 - From left to right: different number of cluster: $k = 4, 5$, we can see from 1 to 2 sub-types of each type

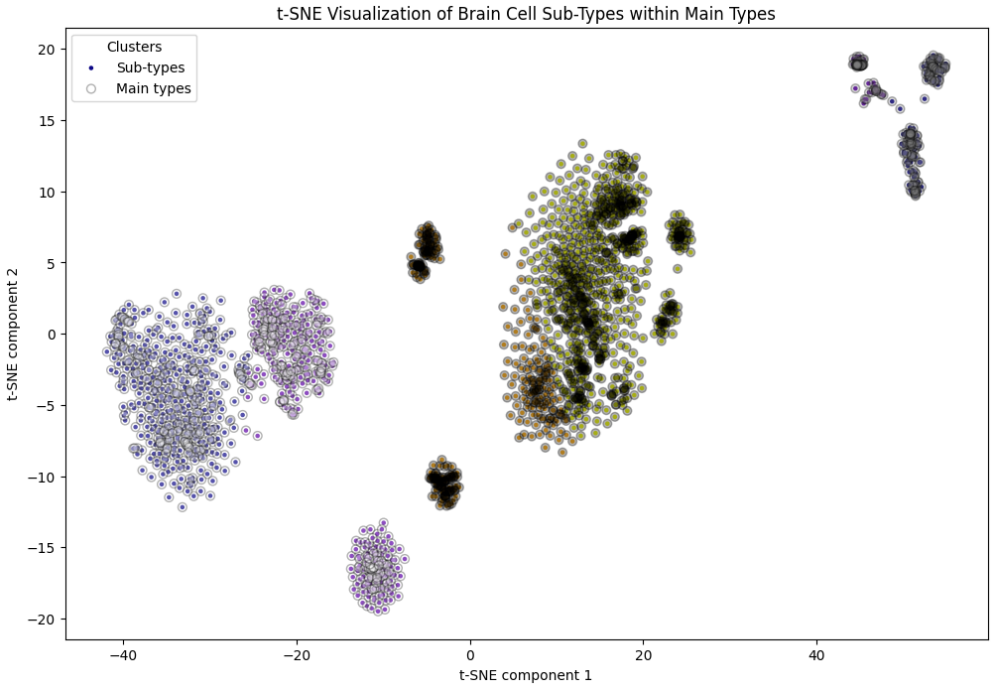

Figure 5: METHOD 2 - Clustering in each cluster: the colors of sub-types are on the small circles, the colors of main-types are on the wrapped circles

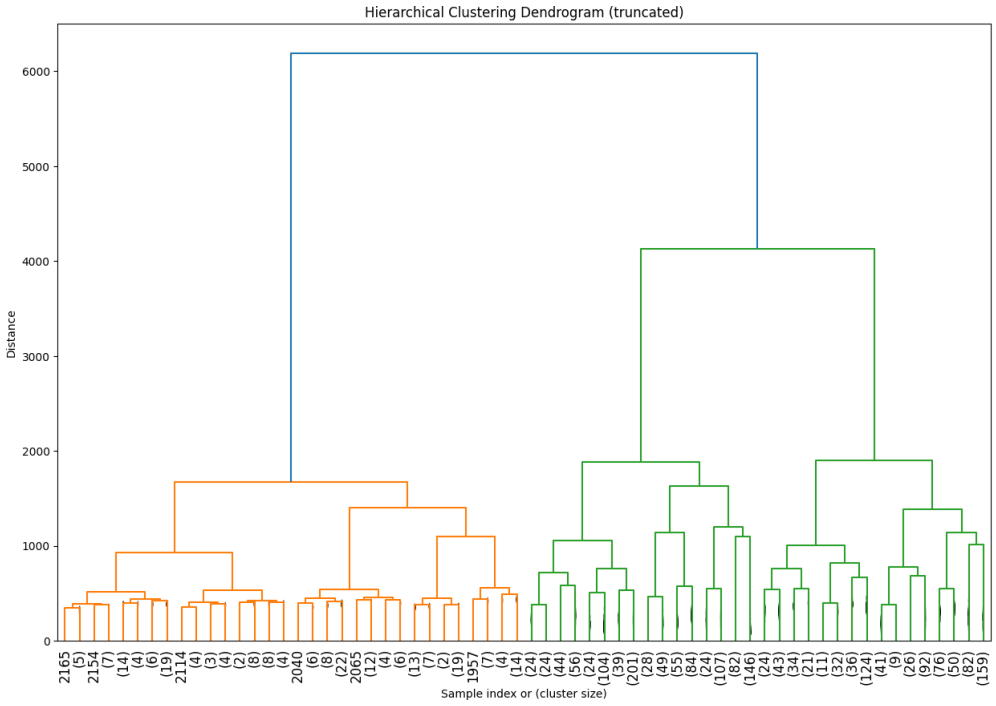

Figure 6: METHOD 3 - Hierarchical clustering algorithm offers flexible ways to select the number of cluster and the number of sub-cluster in each cluster; by examining the dendrogram, we can identify clusters at various levels of granularity, supporting the claim that within each main brain cell type, there are numerous possible sub-types

This visualization supports the claim that within each of the three main brain cell types, there are numerous possible sub-types. The distinct clusters of small points within the larger main clusters show that each main type contains several sub-types.

□

Part 2: Unsupervised Feature Selection

Now we attempt to find informative genes which can help us differentiate between cells, using only unlabeled data. A genomics researcher would use specialized, domain-specific tools to select these genes. We will instead take a general approach using logistic regression in conjunction with clustering. Briefly speaking, we will use the 'p2_unsupervised' dataset to cluster the data. Treating those cluster labels as ground truth, we will fit a logistic regression model and use its coefficients to select features. Finally, to evaluate the quality of these features, we will fit another logistic regression model on the training set in 'p2_evaluation', and run it on the test set in the same folder.

1. *(4 points) Using your clustering method(s) of choice, find a suitable clustering for the cells. Briefly explain how you chose the number of clusters by appropriate visualizations and/or numerical findings. (to cluster cells into the subcategories instead of categories)*

   Answer. In this problem, I used the Elbow method to find optimal number of clusters.

The Elbow method plot shows the sum of squared distances (inertia) for different numbers of clusters. The optimal number of clusters is typically at the "elbow" point where the inertia starts to decrease more slowly.

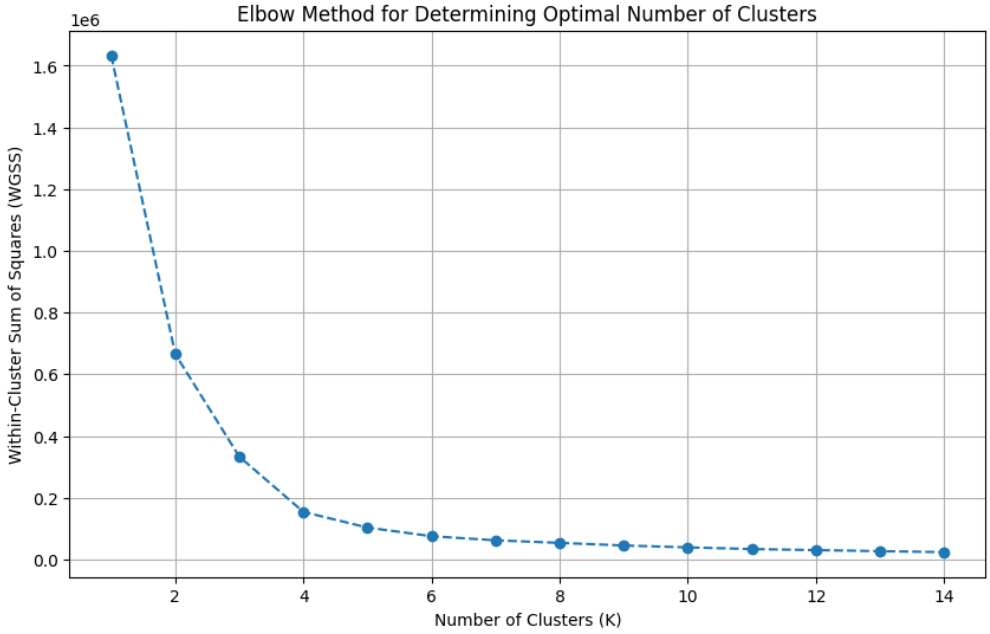

Figure 7: In this figure, the optimal of cluster when apply the K-Means clustering on X_TSNE (X reduced dimension by PCA then TSNE) would be 5 or 6

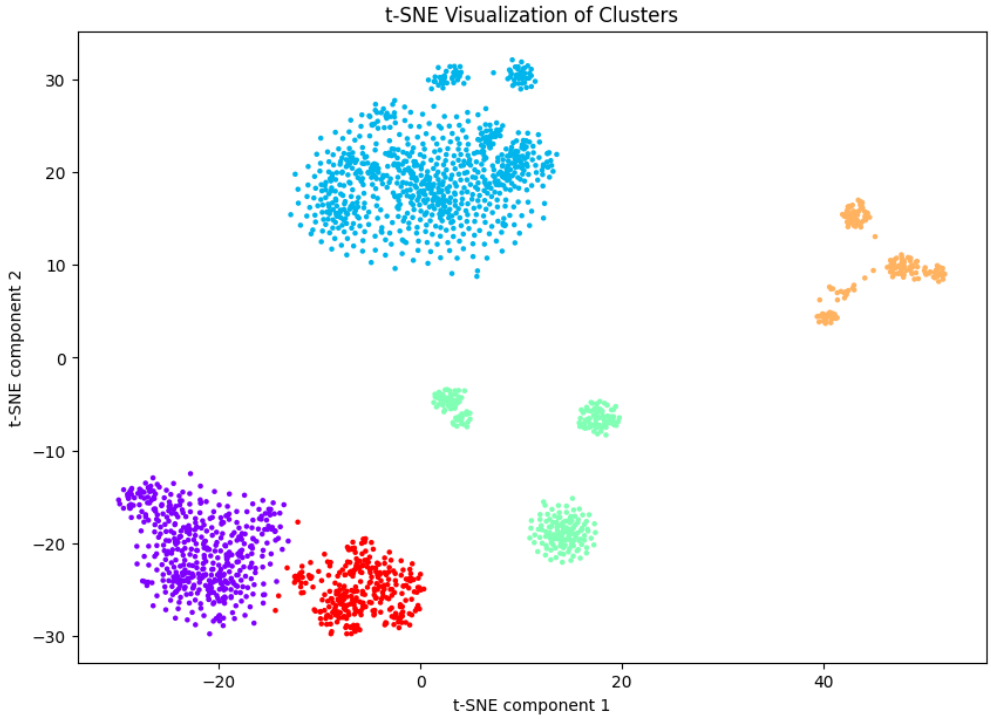

Figure 8: Five distinct groups of neurons are in red, purple, blue, green and yellow colors

□

2. *(6 points) We will now treat your cluster assignments as labels for supervised learning. Fit a logistic regression model to the original data (not principal components), with your clustering as the target labels. Since the data is high-dimensional, make sure to regularize your model using your choice of ℓ_1, ℓ_2, or elastic net, and separate the data into training and validation or use cross-validation to select your model. Report your choice of regularization parameter and validation performance.*

   Multi-class logistic regression: When the underlying data has more than two classes involved, we can adapt Logistic Regression which is usually used for binary classification by one-versus-rest approach. In particular, if we have K classes, we train K separate binary classification models using logistic regression. Each classifier $f_k, \forall k \in \{1, \ldots, K\}$ is trained to determine the probability of a data point belonging to class k. To predict the class for a new point x, we run all K classifiers on x and choose the class with the highest probability, i.e., $\widehat{y} = argmax_{k \in \{1,\ldots,K\}} f_k(x)$.

   Python tip: You may use liblinear solver in LogisticRegression or LogisticRegressionCV for one-versus-rest logistic regression.

   Note: Recall that the 'p2_unsupervised_reduced' and 'p2_evaluation_reduced' folders contain datasets with a reduced number of genes, in case you are unable to run some of the procedures on the larger versions. In particular, a full logistic regression could take 1 or 2 GB of memory to run.

   Note: In question 2, we train a logistic regression model and select 100 'best' features from 'p2_unsupervised' data, extracting such features from this data. In question 3 (the next one),

we take those previously selected features and use on 'p2_evaluation' data. We apply on different data sets, and we are comparing two different data sets.

Answer. In this question, I used the 6 cluster labels of previous experiments to fit into the LogisticRegressionCV. The dataset was split into training and validation sets by ratio 8:2. The parameters of LogisticRegressionCV are cv=5, penalty='l2', solver='liblinear', multi_class='ovr', and max_iter=1000.

After 15 mins of computation on my PC, I got the Validation Accuracy: 0.9816. The confusion matrix looks like:

Table 1: Confusion matrix after ran LogisticRegressionCV on neurons dataset

	precision	**recall**	**f1-score**	**support**
0	0.96	0.95	0.95	75
1	0.99	1.00	0.99	98
2	1.00	0.95	0.97	19
3	1.00	0.99	1.00	129
4	1.00	1.00	1.00	21
5	0.96	0.98	0.97	92
accuracy	0.98	434		
macro avg	0.98	0.98	0.98	434
weighted avg	0.98	0.98	0.98	434

The best regularization parameter (C) is [5.99e-03, 1.00e-04, 1.00e-04, 1.00e-04, 1.00e-04, 2.78e+00] □

3. *(9 points) Select the features with the top 100 corresponding coefficient values (since this is a multi-class model, you can rank the coefficients using the maximum absolute value over classes, or the sum of absolute values). Take the evaluation training data in 'p2_evaluation' and use a subset of the genes consisting of the features you selected. Train a logistic regression classifier on this training data, and evaluate its performance on the evaluation test data. Report your score. (Don't forget to take the log transform $\log_2(x+1)$ before training and testing.)*

   Compare the obtained score with two baselines: random features (take a random selection of 100 genes), and high-variance features (take the 100 genes with highest variance). Finally, compare the variances of the features you selected with the highest variance features by plotting a histogram of the variances of features selected by both methods.

   Note: The histogram should show the distribution of the variances of features selected by both methods. You could show the comparison by overlaying both histograms in the same plot.

   Hint: Refer to the recitation for some guidance if necessary.

   Answer. In this question, I used three kinds of feature: (1) top features (selects the top 100 features based on the logistic regression coefficients from a previously trained model), random features (randomly selects 100 features), and high-variance features (selects the 100 features with the highest variance). I trained three logistic regression models using each set of features, then, evaluated the models on the test set. The results can be shown here:

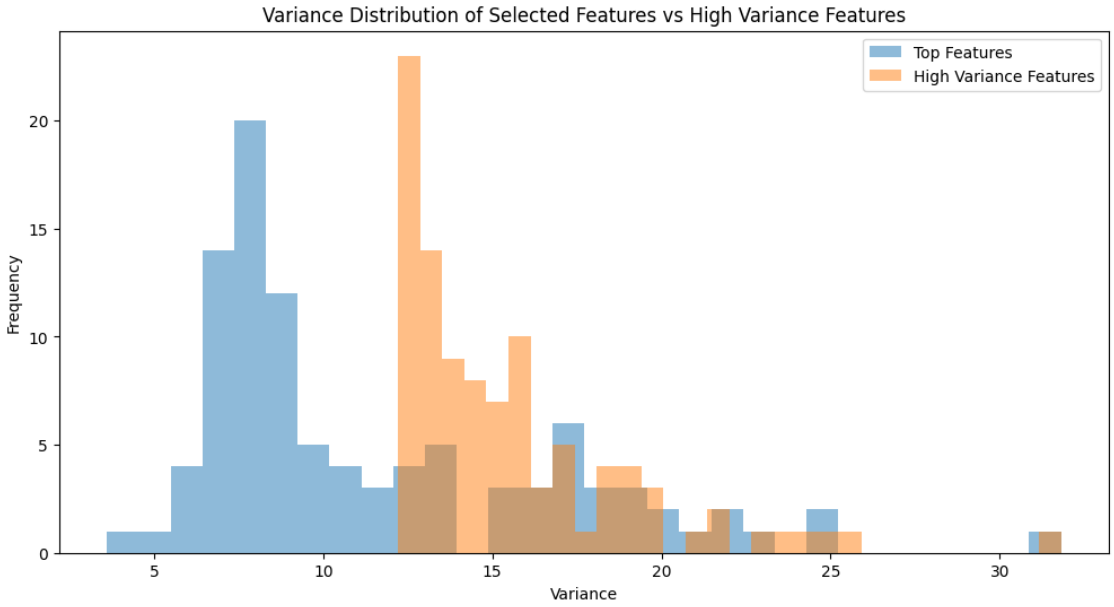

Figure 9: Variance distribution of features selected by two different methods: Top Features (Blue) and High-Variance Features (Orange)

- Top features: the logistic regression model achieved an accuracy of 0.8962; the variance distribution shows a mix of low and moderate variances
- Random features: the model had a significantly lower accuracy of 0.4070, indicating that randomly selected features are not informative
- High-variance features: this model achieved the highest accuracy of 0.9323; the histogram shows that these features have higher variances compared to the top features selected by the logistic regression coefficients

Hence, it can be witnessed from figure 9:

- The histogram for the top features shows a wider spread of variances; most of the selected features have variances between 5 and 15
- The distribution of high-variance features is more concentrated around higher variance values compared to the top features; most of these features have variances between 12 and 20

To conclude, top features based on logistic regression coefficients include a mix of both low and high variance features, optimizing for predictive power rather than variance alone. High-variance features tend to have higher variances overall, but the inclusion of only high-variance features does not necessarily lead to the best model, although in this case, it performed very well.

□

Problem 3: Influence of Hyper-parameters

The hyper-parameter choices used in data analysis techniques can have a large impact on the inferences made. As you may have encountered, finding the best choice of parameter such as perplexity in T-SNE or the number of clusters can be an ambiguous problem. We will now investigate the sensitivity of your results to changes in these hyper-parameters, with the goal of understanding how your conclusions may vary depending on these choices.

1. *(3 points) When we created the T-SNE plot in Problem 1, we ran T-SNE on the top 50 PC's of the data. But we could have easily chosen a different number of PC's to represent the data. Run T-SNE using 10, 50, 100, 250, and 500 PC's, and plot the resulting visualization for each. What do you observe as you increase the number of PC's used?*

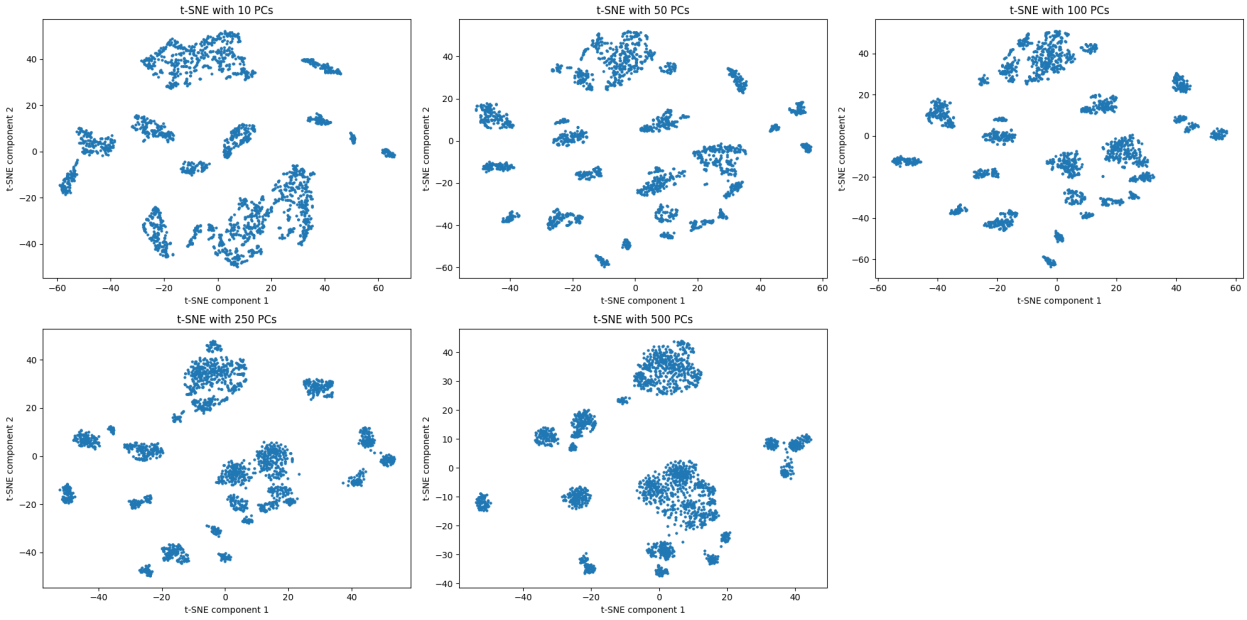

Figure 10: Visualizations of t-SNE on the dataset uses different numbers of principal components (PCs) for dimensionality reduction. The number of PCs used are 10, 50, 100, 250, and 500. Each subplot represents the t-SNE plot for a specific number of PCs, and the goal is to observe how increasing the number of PCs affects the visualization.

Answer. In this question, I ran the T-SNE on different number of PCs. Note that in previous question, the figure 1 illustrated cumulative explained variance vs number of PCs. As we can see from this figure, the more number of PCs was used, the more variance of data can be kept. Analysis the figure 10, it can witnessed:

- T-SNE with 10 PCs: the plot shows clusters, but they are less distinct and more scattered compared to other plots with more PCs; some groups of points appear to be closer together, indicating that some structure is captured, but it may not be very detailed
- T-SNE with 50 PCs: the clusters are more defined compared to the 10 PCs plot; there is more separation between clusters, and individual clusters are more compact.

- T-SNE with 100 PCs: the visualization is similar to the one with 50 PCs but with slightly more detail; clusters are still distinct and well-separated, showing that additional PCs help capture more variance and structure in the data.
- T-SNE with 250 PCs: the clusters are well-defined, and there are more distinct groups compared to the previous plots; the plot captures more fine-grained structures and sub-clusters within larger groups.
- T-SNE with 500 PCs: the clusters are still distinct, but the plot becomes more crowded with more small clusters and scattered points; this suggests that using 500 PCs retains a lot of information, including noise and less relevant details.

Overall, the figure demonstrates that while increasing the number of PCs can capture more variance, there is a trade-off with the introduction of noise, and an optimal number of PCs should be chosen based on the balance between these factors. □

2. *(13 points) Pick three hyper-parameters below (the 3 is the total number that a report needs to analyze. It can take a) 2 from A, 1 from B, or b) 1 from A, 2 from B.) and analyze how changing the hyper-parameters affect the conclusions that can be drawn from the data. Please choose at least one hyper-parameter from each of the two categories (visualization and clustering/feature selection). At minimum, evaluate the hyper-parameters individually, but you may also evaluate how joint changes in the hyper-parameters affect the results. You may use any of the datasets we have given you in this project. For visualization hyper-parameters, you may find it productive to augment your analysis with experiments on synthetic data, though we request that you use real data in at least one demonstration.*

   For visualization hyper-parameters, provide substantial visualizations and explanation on how the parameter affects the image.

   For clustering/feature selection, provide visualizations and/or numerical results which demonstrate how different choices affect the downstream visualizations and feature selection quality.

   Provide adequate explanations in words for each of these visualizations and numerical results.

   Answer. In this question, I choice the following hyper-parameters:

   1. T-SNE perplexity (Category A: visualization)
   2. T-SNE learning rate (Category A: visualization)
   3. Number of clusters chosen for use in unsupervised feature selection (Category B: clustering/feature selection)

   I investigated how changes in these hyper-parameters affect the conclusions that can be drawn from the data.

   T-SNE Perplexity: the perplexity parameter in t-SNE controls the balance between local and global aspects of the data. Low perplexity values consider only nearby points, while high perplexity values consider a larger neighborhood. I used perplexity values of 5, 30, 50, 100, 200 and 500 to observe their effects on the t-SNE visualization.

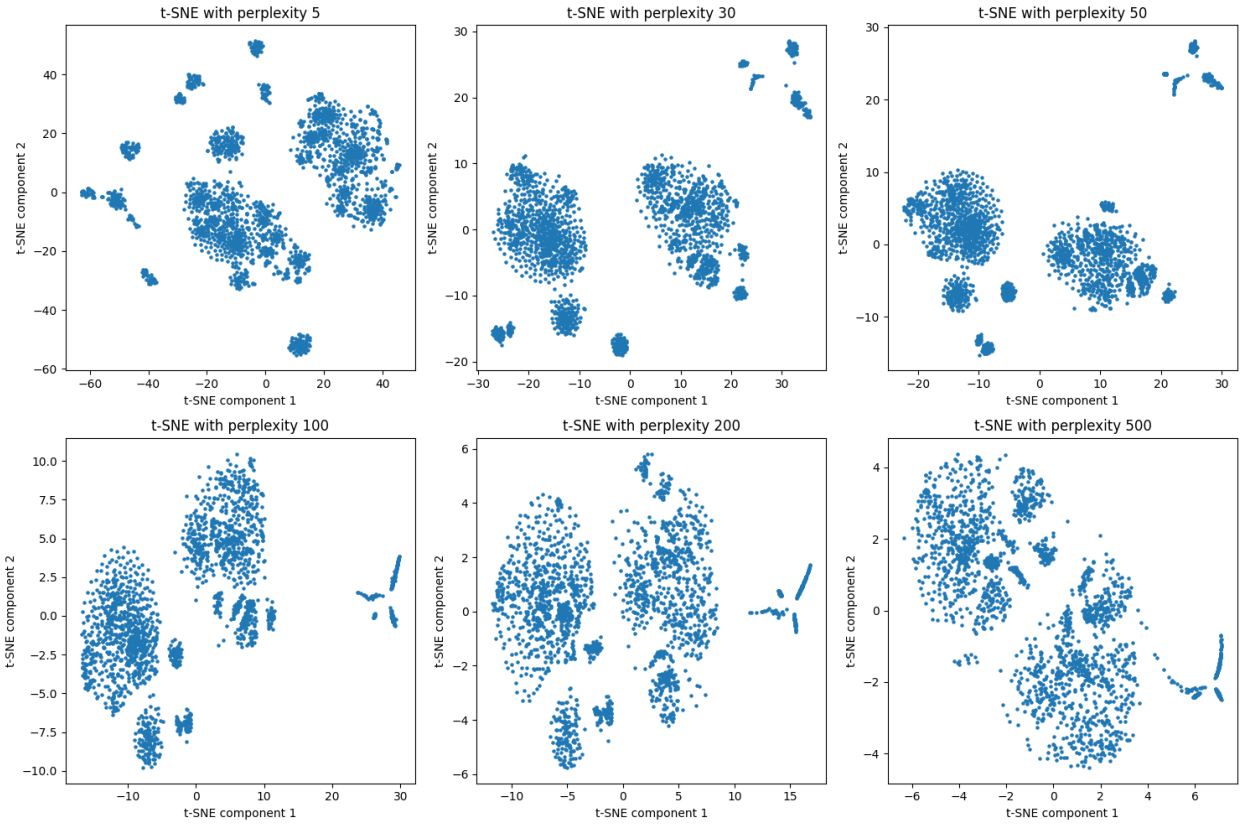

Figure 11: Visualizations of t-SNE on the dataset uses different value of perplexity

As can observed from figure 11:

- Perplexity 5: the plot shows many small, tight clusters, focuses more on local structures and might miss larger, global patterns
- Perplexity 30, 50: the plot provides a balance between local and global structures; clusters are well-separated and clear, capturing both local and larger patterns
- Perplexity 100, 200, 500: the plot shows more spread-out clusters; emphasizes global structures more, possibly at the expense of local details

T-SNE Learning rate: the learning rate in t-SNE affects the speed and stability of the convergence. A low learning rate can lead to slow convergence, while a high learning rate can cause instability. I used learning rate values of 50, 300, 500, 1000, 2000, and 5000 to observe their effects on the t-SNE visualization.

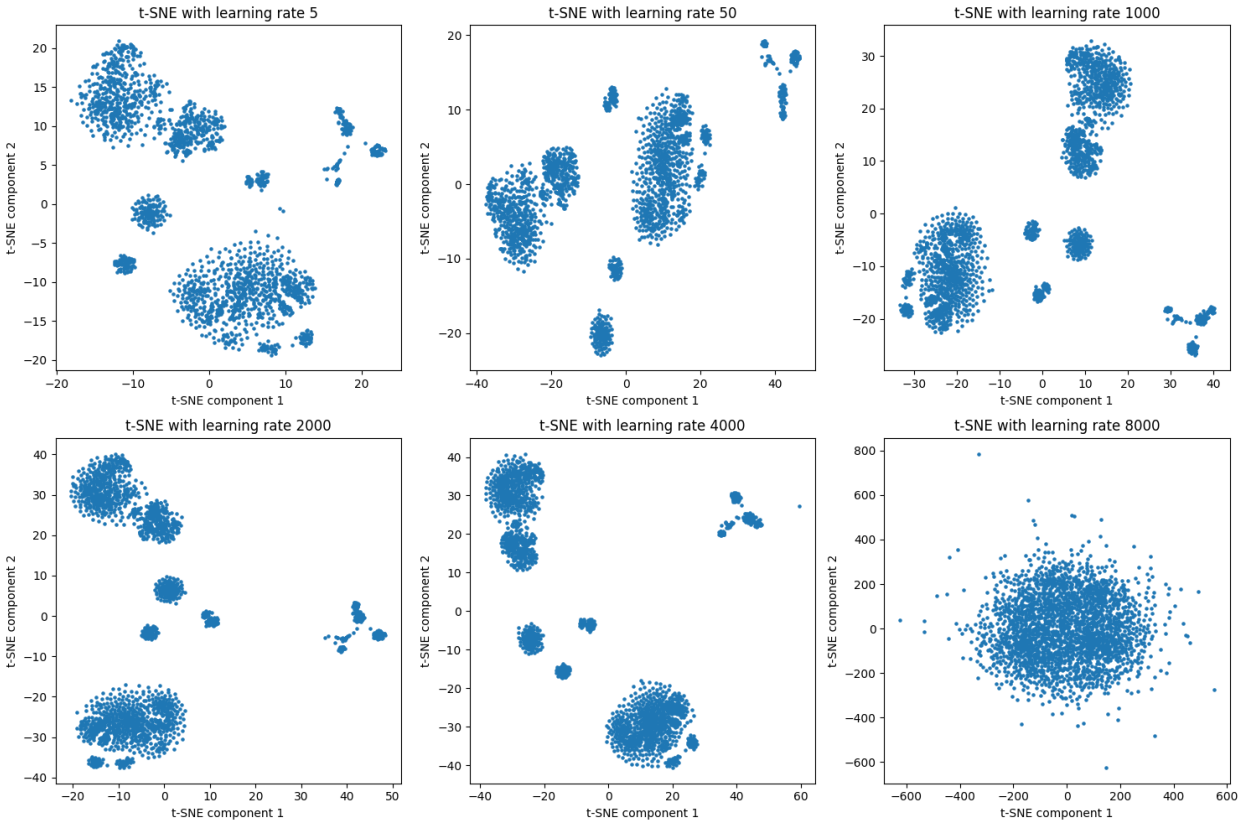

Figure 12: Visualizations of t-SNE on the dataset uses different value of learning rate

As can observed from figure 12:

- Learning Rate 5, 50: the plot is less clear and the convergence is slow; clusters are not well-defined
- Learning Rate 1000, 2000, 4000: the plot shows a good balance, with clear and well-separated clusters; convergence is stable and effective
- Learning Rate 8000: the plot might show instability and noise; clusters can be distorted due to too rapid convergence

Number of Clusters for Unsupervised Feature Selection: the number of clusters chosen for unsupervised feature selection can affect the quality of the selected features. In this experiment, I performed unsupervised feature selection by using KMeans clustering with different numbers of clusters (3, 4, 6, 8, 10, 20, 30), and selects the top 100 features based on logistic regression coefficients trained on these clusters. It then evaluates the classification accuracy of a logistic regression model on the evaluation data using these selected features. Finally, I plotted the evaluation accuracies for each number of clusters in table 2.

Table 2: Evaluation accuracies for different numbers of clusters used in unsupervised feature selection

No. of cluster	**Accuracy**
3	0.9043
4	0.8863
6	0.9179
8	0.9242
10	0.9197
20	0.9323
30	0.9296

As can be observed from the table 2:

- 3 Clusters: accuracy is 0.9043, indicating a reasonable balance but possibly missing finer details
- 4 Clusters: accuracy drops to 0.8863, suggesting that this number of clusters may not effectively capture the underlying structure
- 6 Clusters: accuracy improves to 0.9179, showing better feature representation and capturing more diversity
- 8 Clusters: further improvement to 0.9242, indicating a more optimal number of clusters for feature selection
- 10 Clusters: accuracy is 0.9197, slightly lower than with 8 clusters, possibly due to oversegmentation
- 20 Clusters: accuracy peaks at 0.9323, suggesting that a higher number of clusters can still capture useful features without too much noise
- 30 Clusters: accuracy is slightly lower at 0.9296, indicating that too many clusters may start to introduce noise and reduce generalizability

The results show that increasing the number of clusters generally improves accuracy up to a point, with 20 clusters yielding the highest accuracy of 0.9323. However, too few clusters (3 or 4) miss important details, while too many clusters (30) may introduce noise, slightly reducing accuracy. The optimal range for this dataset appears to be between 8 and 20 clusters.

Summary: T-SNE perplexity affects the balance between local and global structure in the visualization. Lower values focus on local clusters, while higher values capture larger patterns but can become noisy. T-SNE Learning Rate affects the convergence and stability of the t-SNE algorithm. Intermediate values (e.g., 1000) provide the best balance for clear visualizations. Number of Clusters for Feature Selection affects the quality of selected features. An intermediate number of clusters (e.g., 6-20) provides the best balance, capturing diverse yet specific features for high classification accuracy. Too few clusters may miss important features, while too many can introduce noise.

□