National and Kapodistrian Univercity of Athens
Machine Learning on Computational Biology
Assignment #3

Assignment 3: Unsupervised Learning

Konstantinos Mexis

Department of Process Engineering, NTUA, Iroon Politechniou 6 Zografou, Athens, Greece

1. Problem description

For this unsupervised learning assignment, we are given five single-cell synthetic datasets, each comprising the gene expression profiles of 200 cells with 200 genes. The objective is to create a data analysis pipeline that takes a dataset file as input and performs dimensionality reduction, clustering, and visualization of the results.

2. Pipeline implementation

2.1. Data pre-processing

As a preprocessing step, data normalization was performed on all 5 datasets. Data normalization ensures that each feature is on the same scale, reducing the impact of outliers and extreme values. Normalization helps improve the performance and reliability of a machine learning model. In this study, we used the *Standardization Scaling*, subtracting the mean of each observation and dividing it by the standard deviation.

2.2. Dimensionality reduction

- **Principal Component Analysis (PCA)**

PCA is a widely used technique for dimensionality reduction. It identifies the principal components that capture the maximum variance in the data. We performed **PCA** and selected the optimal number of dimensions based on the explained variance ratio or other relevant criteria. We identified the number of components that capture 95% of the total variance. The selected number of components for every dataset in presented in next table:

Dataset	Number of components
Dataset 1	109
Dataset 2	113
Dataset 3	111
Dataset 4	112
Dataset 5	110

TSNE is a non-linear dimensionality reduction technique that aims to preserve local similarities in the data. In our analysis, we used two components to construct the t-SNE reduced data.

UMAP is a non-linear dimensionality reduction technique that preserves both global and local structures in the data. In this study, we selected three components for the dimensional reduction process using UMAP. By reducing the dimensionality to three, we aimed to capture the essential characteristics of the data while minimizing information loss.

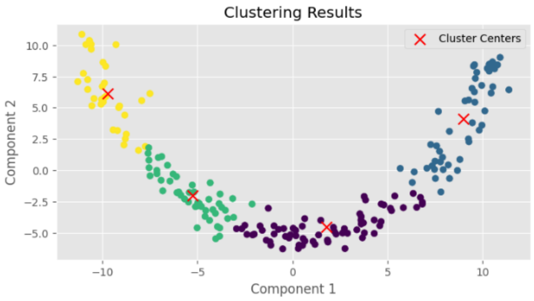
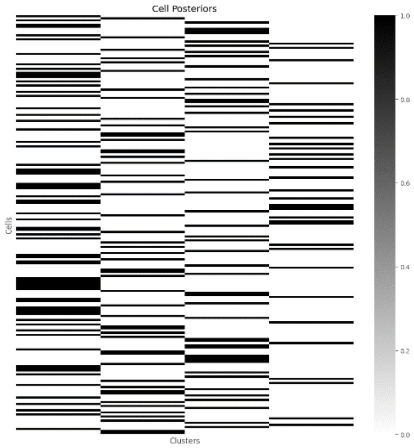
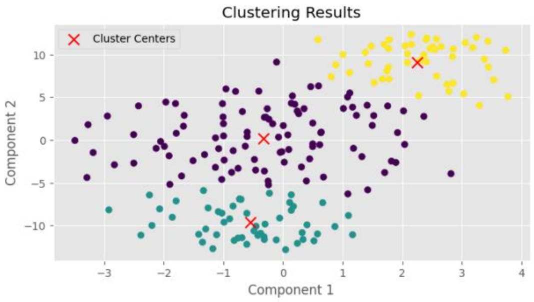
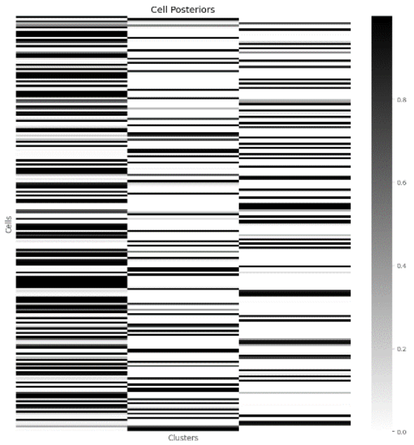
2.3. Clustering

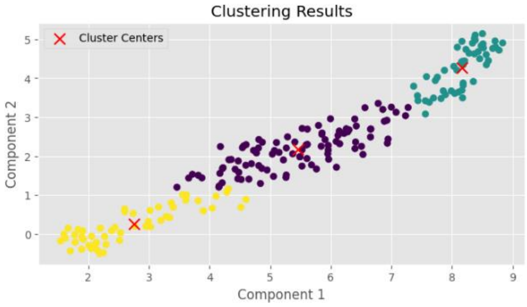
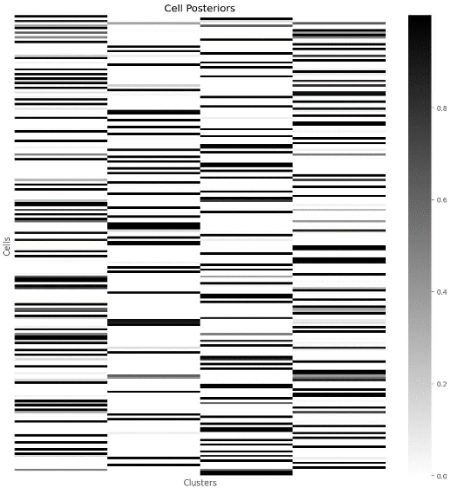
In order to cluster the dimensionality reduced data into the optimal number of cell "states" (clusters), we employed Gaussian Mixture Modeling (GMM). GMM is a probabilistic model that assigns each cell to a mixture component, representing a potential cell state. To determine the optimal GMM model, we utilized the BIC criterion. The BIC criterion takes into account both the goodness of fit and the complexity of the model, allowing us to identify the optimal number of components and the appropriate covariance matrix structure. Each cell was assigned a posterior distribution to each state, indicating the probability of belonging to a particular cluster. The number of extracted states was determined to be optimal based on the BIC criterion, ensuring an effective clustering solution. The optimal GMM model for every dataset and for every dimensionality reduction method is presented in next table:

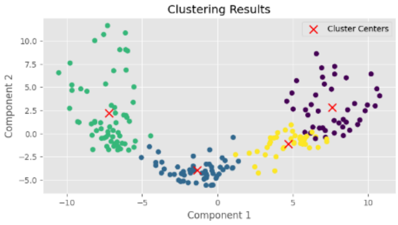
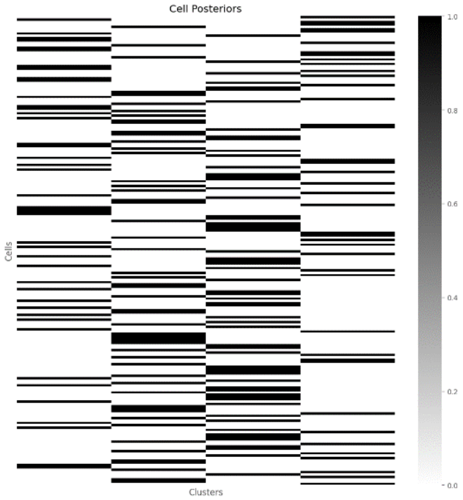
Dataset	**Optimal GMM model**	
Dataset 1	PCA	Optimal number of components: 4 Optimal covariance matrix type: full
	t-SNE	Optimal number of components: 3 Optimal covariance matrix type: diag
	UMAP	Optimal number of components: 6 Optimal covariance matrix type: tied
Dataset 2	PCA	Optimal number of components: 4 Optimal covariance matrix type: full
	t-SNE	Optimal number of components: 3 Optimal covariance matrix type: tied
	UMAP	Optimal number of components: 7 Optimal covariance matrix type: tied
Dataset 3	PCA	Optimal number of components: 3 Optimal covariance matrix type: full
	t-SNE	Optimal number of components: 5 Optimal covariance matrix type: diag
	UMAP	Optimal number of components: 9 Optimal covariance matrix type: spherical
Dataset 4	PCA	Optimal number of components: 5 Optimal covariance matrix type: full
	t-SNE	Optimal number of components: 4 Optimal covariance matrix type: spherical
	UMAP	Optimal number of components: 9 Optimal covariance matrix type: spherical
Dataset 5	PCA	Optimal number of components: 4 Optimal covariance matrix type: full
	t-SNE	Optimal number of components: 5 Optimal covariance matrix type: spherical
	UMAP	Optimal number of components: 6 Optimal covariance matrix type: full

2.4. Visualization

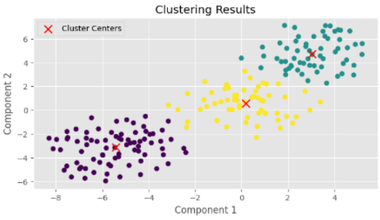
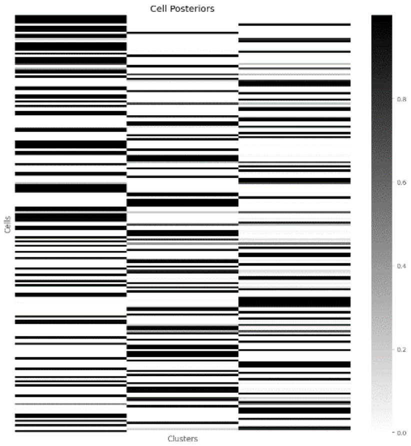
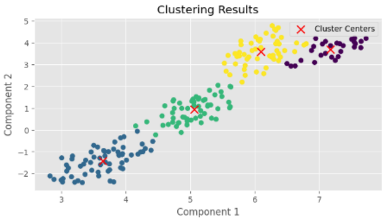
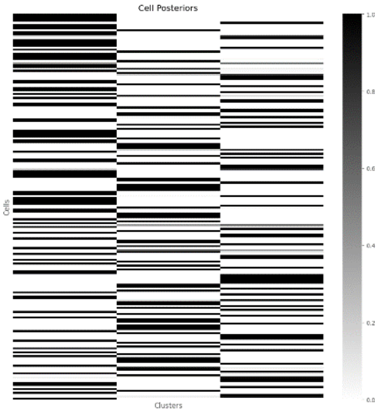
In the next figures, the clusters inferred for every dataset and for every dimensionality reduction method are presented. Also, the cell posteriors for every dataset and for every dimensionality reduction method are presented in the form of a heatmap. In every case, a 2D plot was designed using the first two and most important dimensions of the dataset.

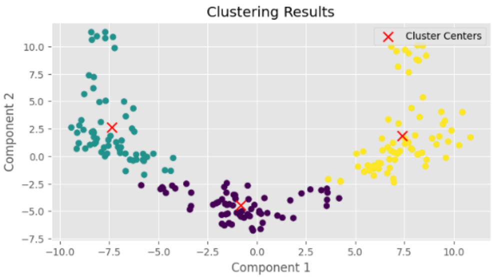
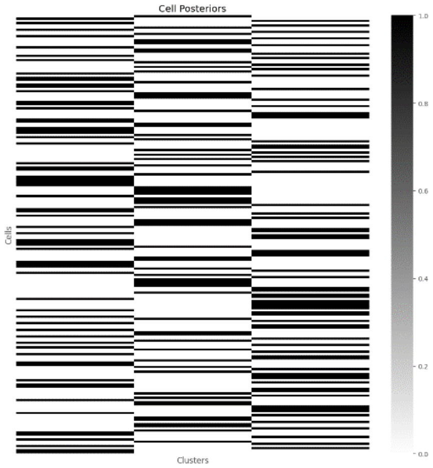
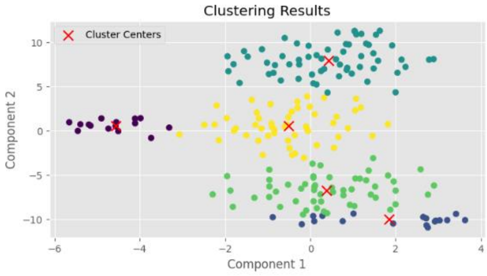
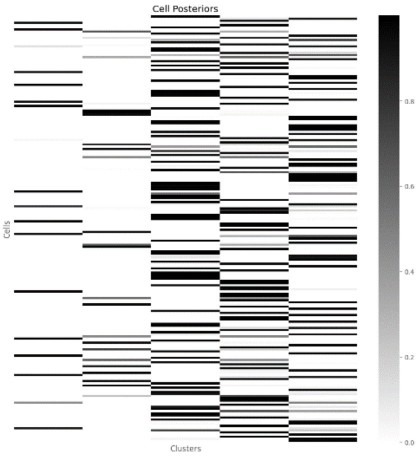
Dataset 1		
	Clusters inferred	**Cell posteriors**
PCA		
TSNE		

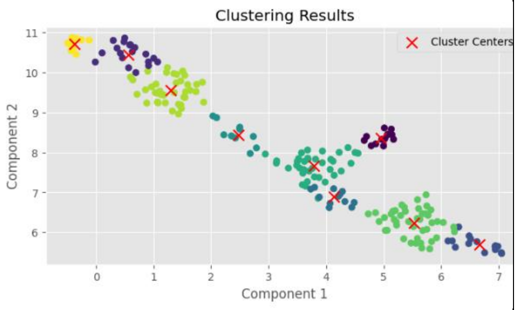
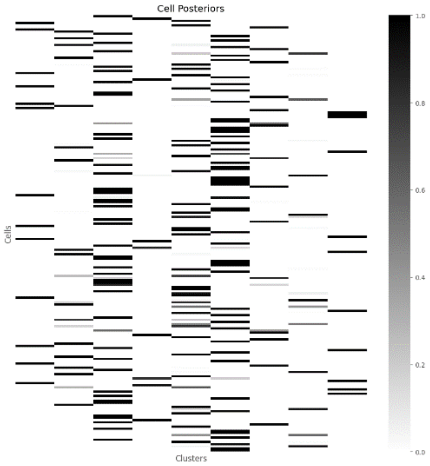
UMAP		

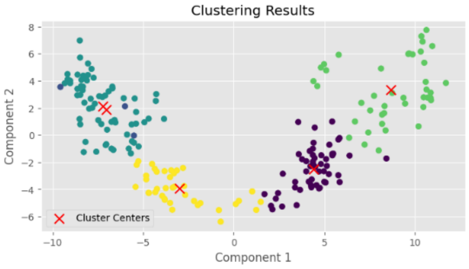
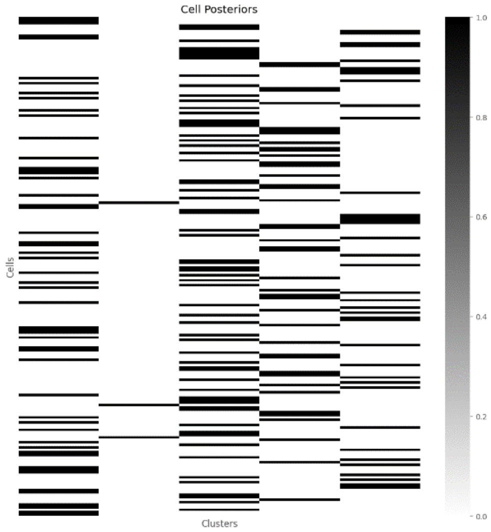
Dataset 2		
	Clusters inferred	**Cell posteriors**
PCA		

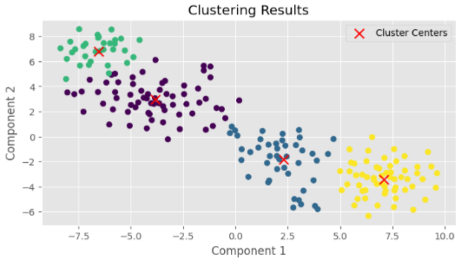
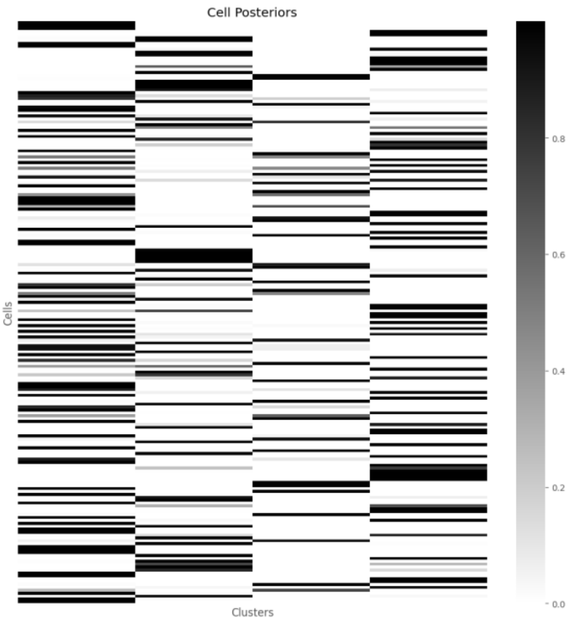
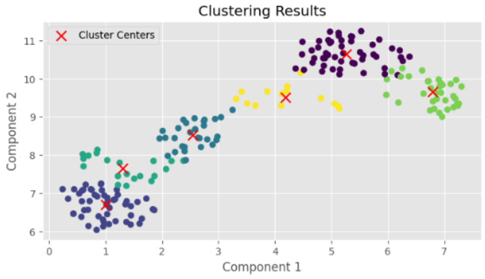
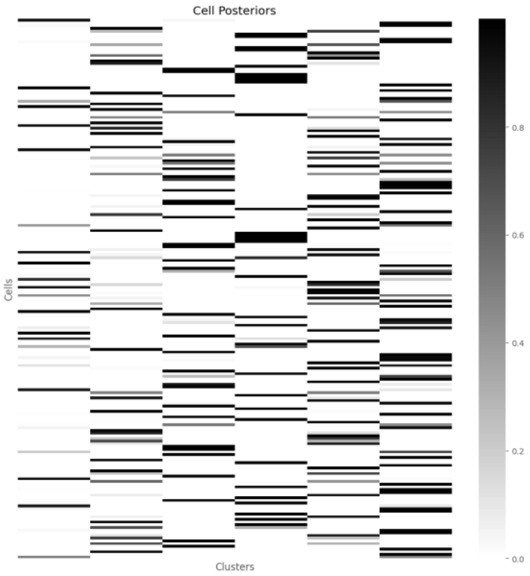
Assignment #3

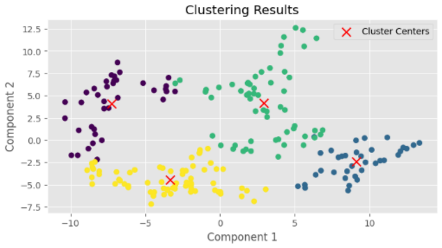
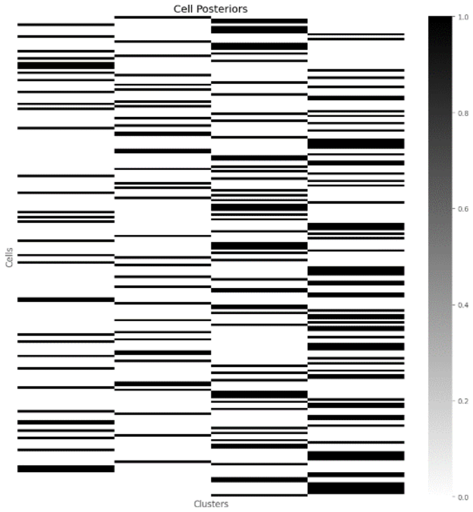
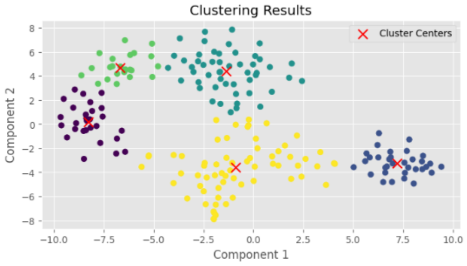
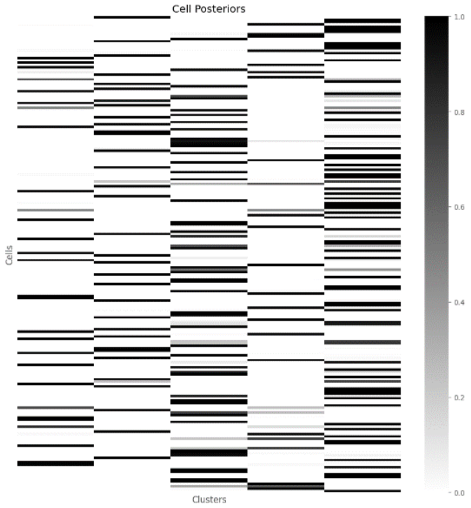
TSNE		
UMAP		

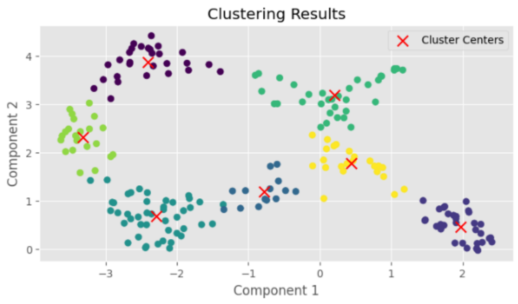
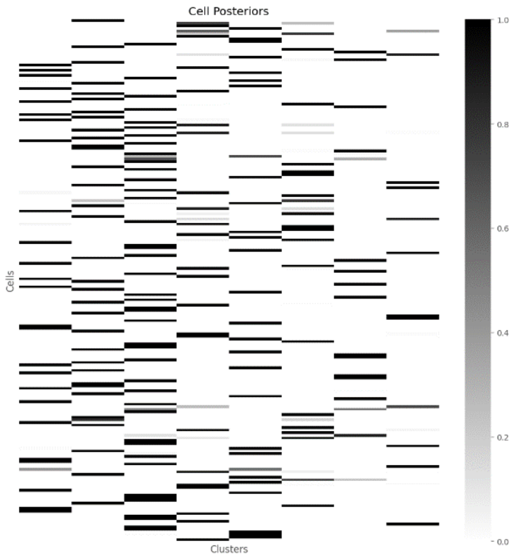
Dataset 3		
	Clusters inferred	**Cell posteriors**
PCA		
TSNE		

UMAP		

Dataset 4		
	Clusters inferred	**Cell posteriors**
PCA		

TSNE	Clustering Results Cluster Centers Component 1 Component 2 −7.5 −5.0 −2.5 0.0 2.5 5.0 7.5 10.0 8 6 4 2 0 −2 −4 −6	Cell Posteriors Cells Clusters 0.8 0.6 0.4 0.2 0.0
UMAP	Clustering Results Cluster Centers Component 1 Component 2 0 1 2 3 4 5 6 7 11 10 9 8 7 6	Cell Posteriors Cells Clusters 0.8 0.6 0.4 0.2 0.0

Dataset 5		
	Clusters inferred	**Cell posteriors**
PCA		
TSNE		

UMAP		

Appendix

The analysis presented in this technical report was performed using Python programming language. The code was executed in Jupyter notebooks, which allows for easy documentation of the code and results.

In order to successfully run the notebooks, the following Python libraries are required: pandas, numpy, matplotlib, seaborn, scikit-learn, and tqdm. These can be installed via pip or conda.

To install the required libraries via pip, simply run the following command in the command line: `pip install pandas numpy matplotlib seaborn scikit-learn tqdm umap-learn `

To install the required libraries via conda, simply run the following command in the command line: `conda install pandas numpy matplotlib seaborn scikit-learn tqdm umap-learn`

It is recommended to use Python version 3.7 or higher for running the notebooks.