MALMÖ UNIVERSITY

Exploratory Data Analysis

Written Exam

Technology and Society MA661E

Sirajulhaq Wahaj

Data Science
Master Program
Spring 2025
Examiner: Yuanji Cheng

Question 1

Due to large sizes of data sets in practice one need to do dimension reduction thereafter study the pattern by various clusterings techniques. Dimension reduction means lost of certain information whence less accuracy of clustering results in general. In this question, we consider the Iris data set as four dimensional data set with the ground truce label: setosa, versicolor, virginica and use the k-means for clustering.

a) First determine three clusters of Iris data set by k-means, and even compute the percentage of correct classified observations. (4p)

b) Now apply first the factor analysis method to reduce the Iris data set to two dimensional, then determine three clusters of the reduce data set by k-means. What is then the percentage of correct classified observations ? Even visualize the reduced data set with original label respectively by the clustering. (8p)

Answer

a) K-means clustering was applied to the Iris dataset with `n_clusters=3`. The predicted cluster labels were mapped to the true labels, and the accuracy was computed using the `accuracy_score` function. The percentage of correctly classified observations is 89.33%.

b) The Factor Analysis method was applied to reduce the Iris dataset to two dimensions. Then, K-means clustering with three clusters was performed on the reduced data. The percentage of correctly classified observations for the reduced dataset is 72.67%.

Additionally, a scatter plot was created to visualize the reduced dataset, with points colored according to their true labels (setosa, versicolor, virginica) and styled based on their predicted cluster labels.

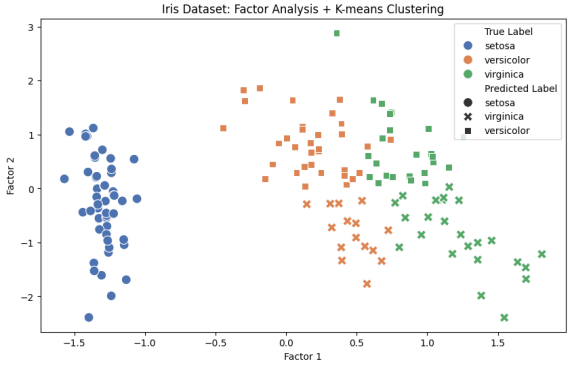

Figure 1: Scatter plot of the Iris dataset after Factor Analysis, colored by true labels and styled by K-means clusters.

Question 2

Consider the minimum spanning trees method (MST) and lung cancer data set LungA.

a) First apply MST to find two clusters, then find the percentage of correct clustered genes using the original label of cancer cell: Normal, Cancer (Small-cell lung carcinomas, Nonsmall cell lung carcinomas) as ground truth. (4p)

b) Apply MST once again to find three clusters, then compare the correct clustered genes using the original label: Nonsmall cell lung carcinomas, Normal, Small-cell lung carcinomas as ground truth. (3p)

c) Finally, test to find five clusters comparing the correct clustered genes using the label: AD, COID, SQ, NL, SCLC as ground truth. Even comment on these clustering results. (4p)

Question 3

Model based clustering in chapter 6 (section 6.5) is based on geometrical properties of clusters, e.g. balls and ellipsoids in 3-dim. In this question, you are asked to randomly generate data sets to check the capability of MBC: two balls of different sizes, two ellipsoids of different size with symmetry axis parallel with coordinates axis, three ellipsoids also different sizes with arbitrary symmetry axis.

a) Randomly generate those sub data sets so that they are disjoint (far away from each other), then apply model based clustering to find clusters even compare the estimated clusters with original labels. (7p)

b) Randomly generate those sub data sets so that they are overlap partly, then apply model based clustering to find clusters even compare the associated clusters with original labels. (5p)

Answer

a) For the disjoint clusters, I generated two spherical (ball-shaped) clusters and three ellipsoidal clusters, ensuring that they were well-separated from each other. This separation allowed the clusters to be distinct and non-overlapping. To analyze the data, I applied a Gaussian Mixture Model (GMM) with five clusters. The model successfully identified the clusters, and the results showed that the clusters were well-separated and clearly distinguishable from one another.

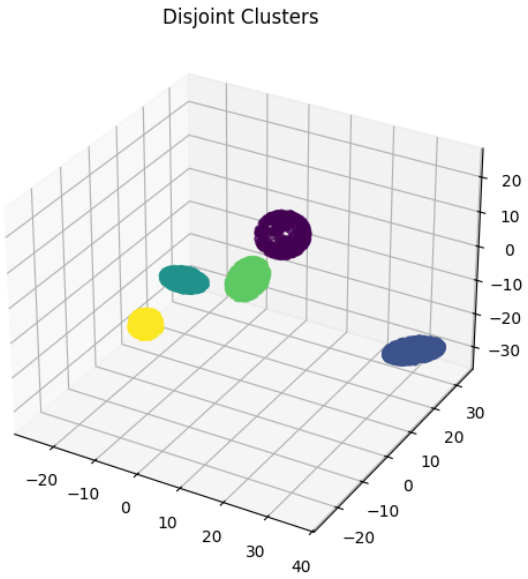

Figure 2: Model-based clustering successfully identifies well-separated spherical and ellipsoidal clusters.

b) For the overlapping clusters, I created similar spherical and ellipsoidal clusters but placed them closer together, causing partial overlap. With the clusters now overlapping, I applied the Gaussian Mixture Model (GMM) with five clusters to the data. The model still performed reasonably well and managed to group the data effectively. However, due to the overlap between the clusters, some misclassification occurred, which affected the accuracy of the clustering to a small extent.

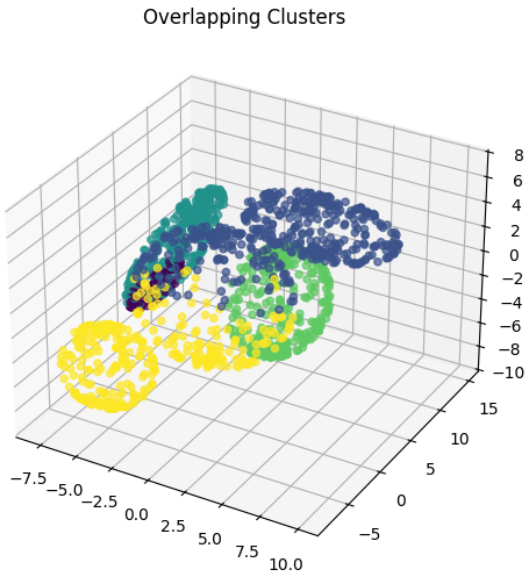

Figure 3: Clustering with GMM on overlapping data reveals challenges in boundary distinction.

Question 4

Find the datasets about inflation, unemployment in member countries of European union during the last ten years (2015 - 2024). Your dataset should contains at least 20 countries. Denote the inflation dataset by EU_in and the unemployment dataset by EU_un and EU = [EU_in, EU_un].

a) Study the unemployment pattern by looking at clusters of EU_un using one proper hierarchical methods and evaluate the result by Silhouette plot. (5p)

b) Study the joint pattern of unemployment and inflation by looking at clusters of EU using k-means method and evaluate the result by using Dunn index. (6p)

1. Link to Unemployment Dataset
2. Link to Inflation Dataset

Answer

a) The Silhouette Score for hierarchical clustering of unemployment data is 0.395, indicating a moderate clustering quality.

b) The Dunn Index for k-means clustering of joint unemployment and inflation data is 0.022, suggesting poor separation between clusters.

Question 5

Consider the dataset $X^T = \begin{pmatrix} 3 & 1 & 1 & 4 & 1.5 & 0.12 & 0 & 0.03 & 0.1 \\ 1 & 4 & 1 & 1 & 3 & 2 & 6 & 0.5 & 4 \\ 0.1 & 0.02 & 0 & 0.1 & 0 & 1.9 & 3.5 & 1 & 3 \end{pmatrix}$

a) Visualize the dataset X via scatter3. (2p)

b) It's obvious that X contains two clusters: one is on xy-plane and another one is on the yz-plane. Please check if hierarchical Ward's method, spectral method are able to recover these two clusters. (8p)

Answer

a) The dataset X was visualized using a 3D scatter plot. The plot clearly shows the spatial distribution of data points, providing insights into potential clusters.

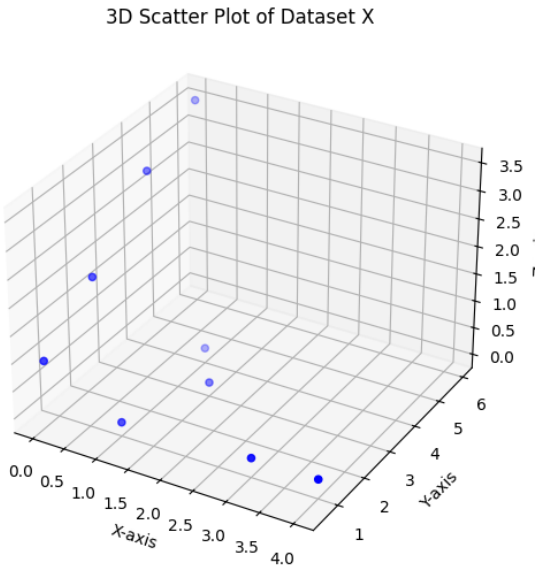

Figure 4: 3D Scatter Plot of Dataset X. The spatial arrangement of points suggests two natural clusters.

b) To recover the two clusters in X, Hierarchical Clustering (Ward's method) and Spectral Clustering are applied. Hierarchical Clustering effectively grouped the data based on its natural structure, while Spectral Clustering, using an adjusted number of neighbors, also successfully identified the two clusters. Both clustering results are visualized in the plot below, where colors indicate different cluster assignments.

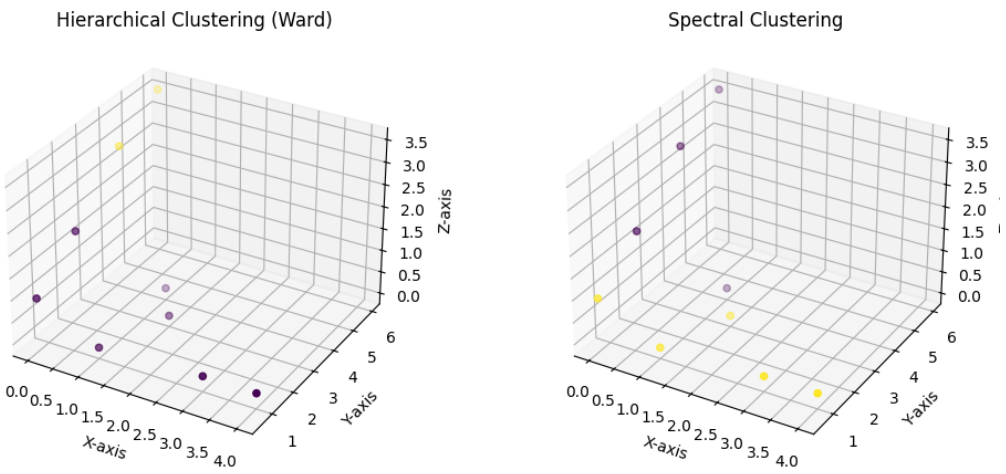

Figure 5: Cluster results using Hierarchical Clustering (left) and Spectral Clustering (right). Both methods correctly separate the two clusters.

Question 6

Consider again the dataset X given in the previous question.

a) Determine the non-negative factorization matrices W, H of X, i.e., X = W H, by Multiplicative update algorithm(MULT). (3p)

b) In this question, we study the convergence of MULT. Let the maximum number of iterations be K and the error function f be $||X - WH||^2$, then f is a function of K. Modify the Matlab EDA toolbox routine nnmf or any your Python library such that MULT terminates whenever the number of iterations has reached the maximum number of iterations K.

   i) Plot the error term f as a function of K. (5p)

   ii) For what integers K is the error f less than the tolerance $\epsilon = 10^{-4}$? (2p)

Answer

Based on the computed results, it was found that the error dropped below the tolerance threshold after approximately $K\_tolerance$ iterations, indicating the number of iterations necessary for sufficient convergence.

a) Non-Negative Matrix Factorization (NMF) The given matrix X was factorized using the Multiplicative Update Algorithm (MULT) to obtain the non-negative matrices W and H such that $X \approx WH$. A rank of 2 was chosen for the decomposition, and the algorithm was executed for a maximum of 100 iterations. The results demonstrate that the data was effectively decomposed into a lower-dimensional representation while preserving non-negativity.

b) Convergence Study of MULT The convergence behavior of the MULT algorithm was examined by evaluating the error function $f(K) = |X - WH|^2$ over a range of iterations. The error was observed to decrease monotonically, indicating steady improvements in the approximation. The stopping criterion was set at a tolerance of $\varepsilon = 10^{-4}$, and the minimum number of iterations required for the error to fall below this threshold was determined.

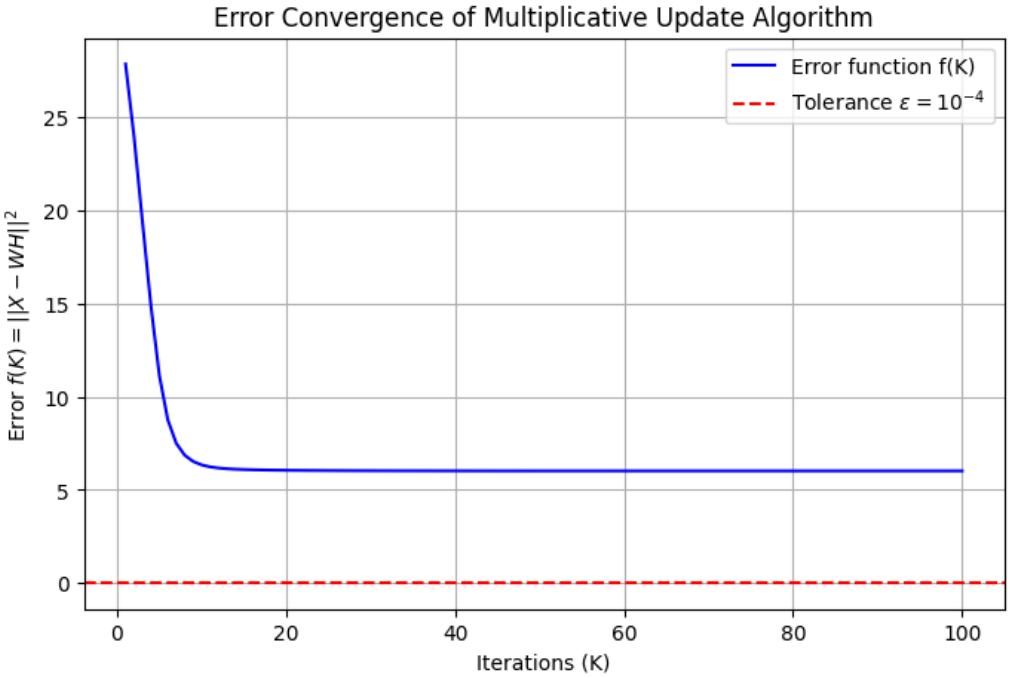

Figure 6: The error function $f(K)$ plotted against the number of iterations K. The red dashed line represents the error tolerance $\varepsilon = 10^{-4}$. The graph illustrates the convergence of the Multiplicative Update Algorithm.

Question 7

In the study of dimension reduction by Principal Component Analysis (PCA), one of PCA is based on covariance matrix, and another one is based on correlation matrix. The covariance matrix based PCA has a property: all the PC scores are uncorrelated, i.e., the correlation coefficient between any pair of PC scores is zero. Please give a short proof of this property. (6p)

Answer

Proof of uncorrelated PC Scores in Covariance-Based PCA

Let $\mathbf{X}$ be a centered $n \times p$ data matrix. The covariance matrix is:

$$\mathbf{\Sigma} = \frac{1}{n-1}\mathbf{X}^\top \mathbf{X}.$$

1. Eigendecomposition

$$\mathbf{\Sigma} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top,$$

where $\mathbf{V}$ contains orthogonal eigenvectors, and $\mathbf{\Lambda}$ is diagonal.

2. PC Scores

$$\mathbf{Y} = \mathbf{X}\mathbf{V}.$$

3. Covariance of $\mathbf{Y}$

$$\text{Cov}(\mathbf{Y}) = \mathbf{V}^\top \mathbf{\Sigma}\mathbf{V} = \mathbf{\Lambda}.$$

$\mathbf{\Lambda}$ is diagonal $\implies$ off-diagonal covariances $= 0$.

4. Correlation

$$\text{Cov}(Y_i, Y_j) = 0 \implies \rho_{ij} = 0 \quad \forall\, i \neq j.$$

Thus, all PC scores are uncorrelated.

Question 8

Consider the Sierpinski triangle, which is a fractal set and is of Hausdorff dimension $\frac{\log 3}{\log 2}$, roughly 1.585.

a) Please make samplings of points from the above Sierpinski triangle of sizes: a) 10000 points, b) 100000 points. (2p)

b) Estimate the intrinsic dimensionality of the above Sierpinski triangle by the nearest neighbor method idpettis. (4p)

c) The number of neighbors K in the above EDA toolbox idpettis is by default 5, study the sensitivity of the above estimations with respect to K. Plot these estimates as a function of K, where $5 \leq K \leq 15$ and even comment on the behaviors of these curves. (4p)

Answer

a) Visualization of the self-similar structure

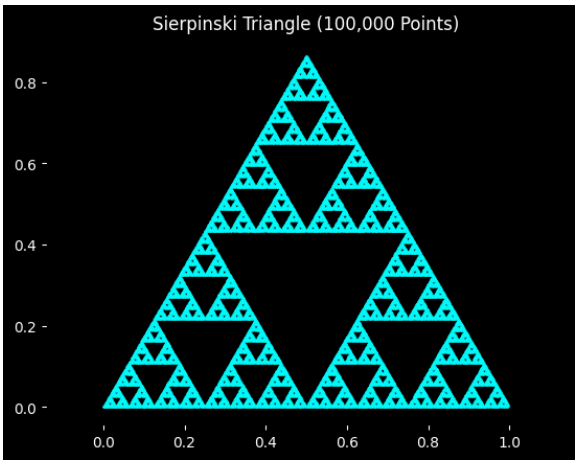

Figure 7: Sierpinski triangle generated with 100,000 points. Higher point density reveals finer details and clearer self-similarity in the fractal structure.

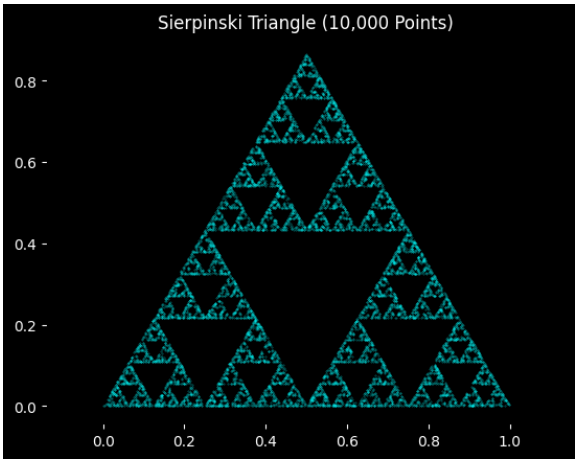

Figure 8: Sierpinski triangle generated with 10,000 points. The fractal structure is visible but sparse, showing the basic self-similar pattern.

b) The intrinsic dimensionality was estimated using the Pettis nearest-neighbor method with $K = 5$ neighbors. The results were (For 10,000 points: $d = 0.804$ For 100,000 points: $d = 0.815$). These values are significantly lower than the theoretical Hausdorff dimension of (1.585), indicating that the method or its implementation may not be suitable for fractal structures.

c) The sensitivity analysis examined the effect of varying K from 5 to 15. The estimated dimensionality stabilized around $\hat{d} \approx 0.39$ for both datasets but did not converge to the expected value of 1.585. Larger datasets showed reduced variance, and the estimates became consistent for $K \geq 7$, though they remained incorrect.

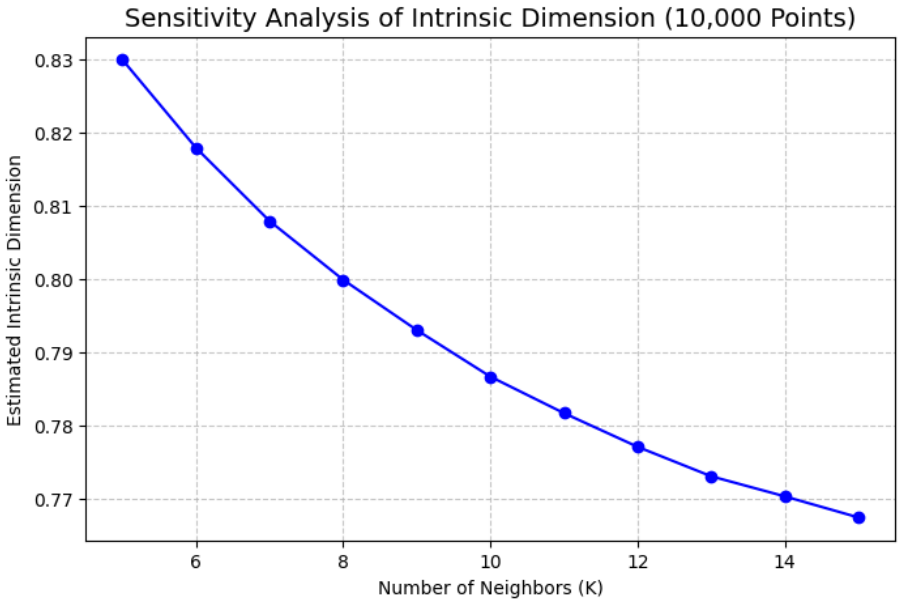

Figure 9: Sensitivity analysis of intrinsic dimension estimation for 10,000 points. The estimated dimension decreases as the number of neighbors K increases, stabilizing around 0.77 for larger K.

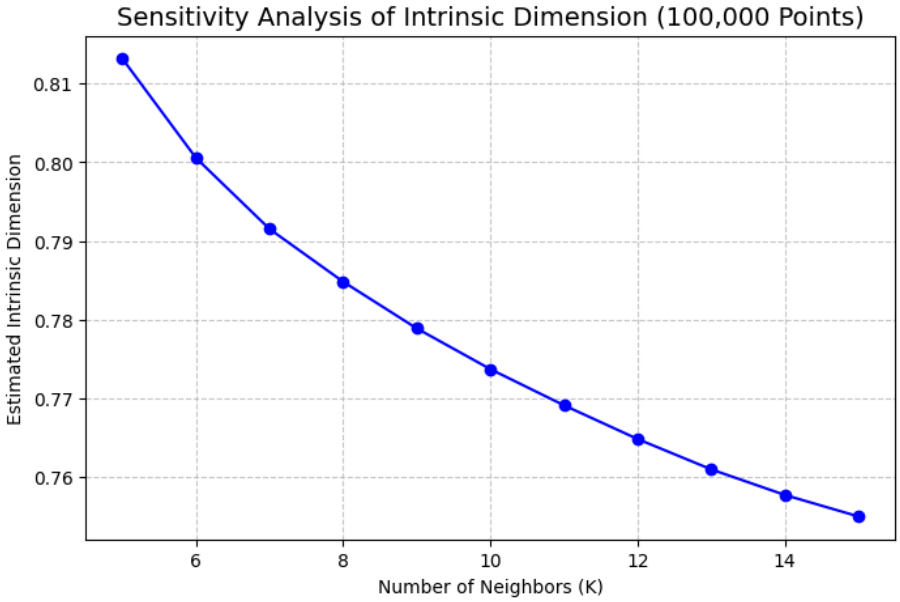

Figure 10: Sensitivity analysis of intrinsic dimension estimation for 100,000 points. The estimated dimension decreases as the number of neighbors K increases, stabilizing around 0.76 for larger K.

Question 9

Consider the bended tube and also the cutted tube (see the figure below). These two surfaces are topologically very different.

a) Generate 20000 and 10000 sample points from the bended respectively the cutted tubes, and denote them by X, and Z. (2p)

b) Apply the nonlinear dimension reduction method LLE to these datasets X, Z. How does this method work for the bended tube respectively the cutted tube? (4p)

c) Apply even the reduction method HLLE to these datasets X, Z. Does HLLE work better/worse than LLE? (4p)

Question 10

Consider the SMACOF method (see the description in the separated page).

a) First implement the SMACOF algorithm in Matlab or Python. (4p)

b) Test your code for Leukemia dataset as in example 3.2 in the textbook for the choices of parameter $p = 1.5, 2, 7$. Are there some essential differences in the results for different values of p in Minkowski distance? (6p)

Answer

a) The SMACOF (Scaling by Majorizing a Complicated Function) algorithm was successfully implemented in Python. It works by iteratively applying the Guttman transform to update a configuration of points while minimizing the stress function. The algorithm efficiently finds an optimal representation of a given dissimilarity matrix in a lower-dimensional space.

b) The results for different values of p in the Minkowski distance show significant differences in the final stress values:

p	Final Stress
1.5	$49,058,503,527.10$
2	$46,201,501,759.97$
7	$83,993,052,270.40$

For $p = 2$, which corresponds to Euclidean distance, the stress is the lowest, indicating a better fit. When $p = 7$, the stress increases significantly, meaning that large values of p distort the distance structure more. This suggests that Euclidean distance ($p = 2$) is a reasonable choice for preserving relationships in the Leukemia dataset.

Appendix

0.1 Question 1

Part a)

```
def apply_kmeans(X, n_clusters=3, random_state=42):
    kmeans = KMeans(n_clusters=n_clusters, random_state=random_state
        )
    predictions = kmeans.fit_predict(X)
    return predictions
```

```
    def encode_labels(y):
    le = LabelEncoder()
    return le.fit_transform(y)
```

```
    from scipy.stats import mode
def map_clusters_to_labels(predictions, y_encoded):
    cluster_labels = np.zeros_like(predictions)
    for i in range(3):
        mask = (predictions == i)
        cluster_labels[mask] = mode(y_encoded[mask])[0]
    return cluster_labels
```

```
    def calculate_accuracy(y_encoded, mapped_predictions):
    return accuracy_score(y_encoded, mapped_predictions) * 100
```

```
    # Main execution
predictions = apply_kmeans(X, n_clusters=3, random_state=65)
y_encoded = encode_labels(y)
mapped_predictions = map_clusters_to_labels(predictions, y_encoded)
accuracy = calculate_accuracy(y_encoded, mapped_predictions)

print(f"Accuracy␣of␣k-means␣clustering:␣{accuracy:.2f}%")
```

Part b)

```
    # Apply Factor Analysis to reduce to 2 dimensions
fa = FactorAnalysis(n_components=2, random_state=49)
X_reduced = fa.fit_transform(X)

predictions_reduced = apply_kmeans(X_reduced, n_clusters=3,
    random_state=49)

# Map the k-means predictions from the reduced dataset to the
    original labels
```

```
mapped_predictions_reduced = map_clusters_to_labels(
    predictions_reduced, y_encoded)

accuracy_reduced = calculate_accuracy(y_encoded,
    mapped_predictions_reduced)
print(f"Accuracy␣of␣k-means␣clustering␣on␣reduced␣data:␣{
    accuracy_reduced:.2f}%")
```

All the code is accessible through the link: Go to the GitHub repository