Amirkabir University of Technology

(Tehran Polytechnic)

Project(4) Report:Topics in Mathematics and Applications

Zohreh Sarmeli Saeedi

May 2025

1 First Section: Computing Positions of Iranian Cities in 2D Space Using ISOMAP

1.1 Description

In this section, we aim to compute the positions of Iranian cities in a two-dimensional space using the ISOMAP algorithm. ISOMAP is a nonlinear dimensionality reduction technique that preserves geodesic distances on a manifold (Tenenbaum et al., 2000). Recent applications of ISOMAP in geospatial data analysis have shown its effectiveness in mapping high-dimensional distance data to lower dimensions while maintaining topological structures (Joshi et al., 2025).

1.2 Steps

1. Load the file `distances.csv`, which contains information about distances between Iranian cities.
2. Convert the data in the file to a matrix.
3. Implement the ISOMAP algorithm using the KNN approach:
   - Construct a KNN graph from the distance matrix: Using the `NearestNeighbors` function with the distance metric set to `precomputed`, compute the k+1 nearest neighbors for each point (including the point itself).
   - Use the function `kneighbors_graph(..., mode='distance')` to create a graph where edge weights are equal to distances.
   - In the KNN graph, only edges between some points exist; the rest are 0, meaning no connection. Change zeros to `inf` (except the diagonal, which remains zero, representing the distance of each point to itself).
   - Using Dijkstra's algorithm (`method='D'`), compute the shortest paths between all points, assuming the graph is undirected (`directed=False`).
   - The `geodesic_dist` matrix represents the graph distances between all points (not just direct distances, but shortest paths).
   - Use MDS to attempt to map to a lower-dimensional space (e.g., 2D) that preserves these distances (as "dissimilarities"). The input to MDS is the geodesic distance matrix.

- The output embedding is an n $\times$ n_components array (e.g., n $\times$ 2), providing coordinates for each point in the new space.

4. The output image obtained after applying some transformations such as reflection and rotation will look as follows:

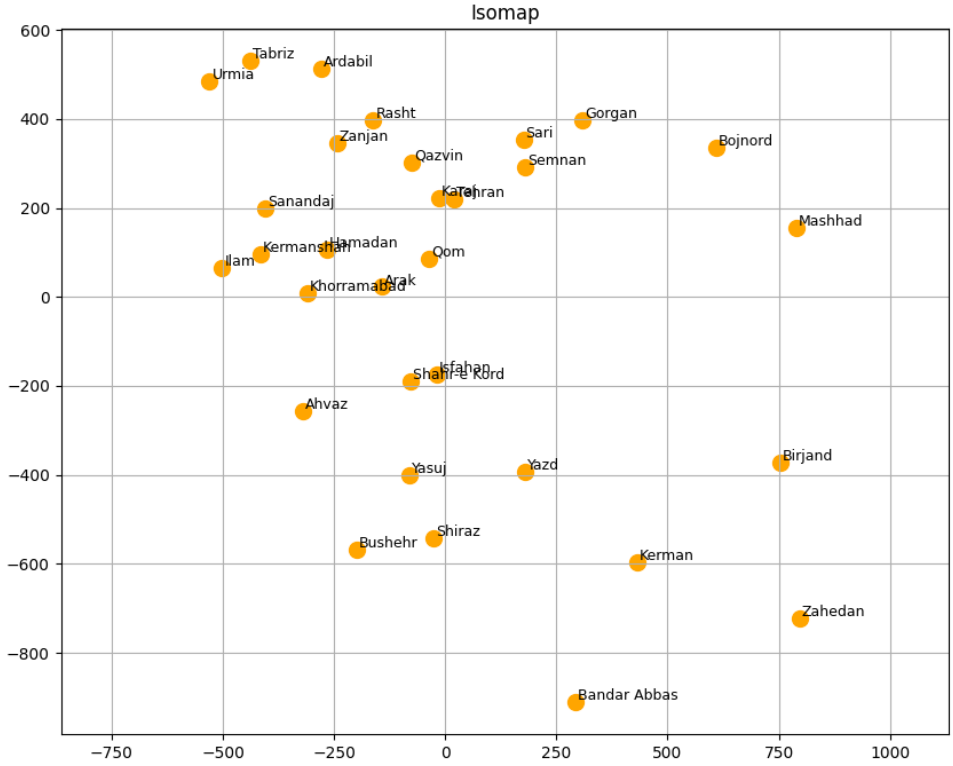

Figure 1: ISOMAP output from manual implementation.

It can be observed that this algorithm has relatively well captured the positions of the cities.

5. Implementing the ISOMAP algorithm using the scikit-learn library:

   - `sklearn.manifold` is used for nonlinear dimensionality reduction.
   - Create an instance of the `Isomap` class:
     - `n_neighbors=5`: Each point is considered only with its 5 nearest neighbors.
     - `n_components=2`: We want to map the data to a two-dimensional space (for plotting).
     - `metric='precomputed'`: Distances are precomputed (in the CSV file), so Isomap uses them.
   - Then run `fit_transform`: This fits the Isomap model based on the distance data (`fit`) and transforms the data to the new two-dimensional space (`transform`).

   The output image obtained after applying some transformations such as reflection and rotation will look as follows:

Figure 2: ISOMAP output from scikit-learn implementation .

It can be observed that this algorithm has also relatively well captured the positions of the cities.

1.3 Comparison of the Two Algorithm Outputs

1.3.1 Similarities

1. Overall Layout Structure: Cities that are geographically adjacent are clustered close together in both images. For example, northwestern cities like Tabriz, Urmia, Ardabil are in the top-left in both outputs. Southeastern cities like Zahedan, Bandar Abbas, Kerman are in the bottom-right or bottom-middle in both outputs.

2. Relative Orientation: The overall direction of clusters (e.g., west-east or north-south) is approximately similar in both images.

1.3.2 Differences

1. Scaling and Dispersion: In the library version (second image), points are displayed more symmetrically and less compressed. The manual version (first image) has more dispersion, and some areas are less stretched or more compressed.

2. Precise Positions of Some Cities: Mashhad in the library version is shifted more to the top-right, with greater distance from other cities. Isfahan and Shahre Kord in the manual version are placed lower, while in the library version they are closer to the center and more balanced. Some cities like Gorgan, Bojnord are closer to the center in the manual version but in the top-right in the library version.

3. Visual Quality and Cluster Uniformity: The library version, due to optimized functions, provides a smoother and more symmetric representation of cities. This indicates better stability in computing the geodesic distance matrix and MDS transformation.

1.4 Conclusion

The manual implementation performs acceptably and reconstructs the overall data structure well. However, the library version provides a smoother, more balanced, and in some points more accurate output. These differences may arise from:

- The method of computing shortest paths.
- Different settings like the number of neighbors.

- The way of centering and dimensionality reduction with MDS.

2 Second Section: Dimensionality Reduction Analysis on the MNIST Dataset

2.1 Introduction

This project aims to reduce the dimensions of the MNIST dataset (handwritten digit images) to a two-dimensional space and analyze the results using PCA and t-SNE algorithms. The steps include converting data to one-dimensional vectors, applying dimensionality reduction, evaluating quality with the Trustworthiness metric, and examining the impact of the Perplexity parameter on t-SNE. PCA is a linear technique that maximizes variance preservation (Jolliffe and Cadima, 2016), while t-SNE focuses on preserving local structures through probabilistic modeling (van der Maaten and Hinton, 2008). Recent studies have explored federated variants of t-SNE for distributed data visualization (Qiao et al., 2025).

2.2 Methodology

1. Data Preparation:
    - The MNIST dataset consists of 70,000 28×28 pixel images. Each image is flattened to a 784-element vector (28×28).
    - Due to computational limitations, the first 10,000 samples were used.
    - Data were normalized (divided by 255) to have values between 0 and 1.
2. Dimensionality Reduction:
    - PCA: A linear method for dimensionality reduction by preserving maximum variance.
    - t-SNE: A nonlinear method focusing on preserving local structures (default Perplexity 30).
    - Both methods reduced the data to two dimensions.
3. Quality Evaluation:
    - The Trustworthiness metric was used to compare PCA and t-SNE. This metric measures the preservation of neighborhood structures (value 1 = perfect preservation) (Venna and Kaski, 2001).
    - The effect of Perplexity on t-SNE was examined with values [5, 10, 20, 30, 40, 50], and the KL Divergence plot was drawn.

2.3 Responses to Project Parts

1. Converting MNIST Images to One-Dimensional Vectors:
    - The MNIST dataset consists of 70,000 28×28 pixel images of handwritten digits. Each image is converted to a 28×28 matrix, which can be flattened to a one-dimensional vector of 784 elements (28×28=784).
    - In this project, the flattening method was used, as the loaded MNIST data (from `fetch_openml`) is already in 784-dimensional vector form.
    - An alternative method is using image encoders (like autoencoders or convolutional networks), but this is more complex and unnecessary for the project's goal (dimensionality reduction with PCA and t-SNE).
    - For quality assurance, data were normalized (divided by 255) so pixel values are between 0 and 1.
2. Applying PCA and t-SNE with Scikit-learn:
    - PCA: Used the `PCA` class in `sklearn.decomposition`. The number of components was set to 2 to reduce data to two dimensions. This is a linear method focusing on overall variance.
    - t-SNE: Used the `TSNE` class in `sklearn.manifold`. The number of components was set to 2, and the default Perplexity to 30. t-SNE is a nonlinear method focusing on local structures.
    - To reduce execution time, only 10,000 MNIST samples were used.

3. Explanation of the Trustworthiness Metric:

   - Trustworthiness is a metric for evaluating dimensionality reduction quality, checking if the neighborhood structure of points in the original high-dimensional space is preserved in the reduced low-dimensional space.
   - In summary: If points that were neighbors in the original space remain neighbors in the reduced space, Trustworthiness will be high.
   - This metric ranges from 0 to 1; a value of 1 indicates perfect preservation of neighborhood structure.
   - Method: For each point, neighbors in the original and reduced spaces are compared. If points that were neighbors in the original space are not in the reduced space, a penalty is applied.
   - The formula compares rankings of neighbors and penalizes differences.
   - Approximate formula (where n is the number of samples, k is the number of neighbors, $U_k(i)$ are points in the reduced space but not in the original, $r(i,j)$ is the rank of point j relative to i in the original space):

$$T(k) = 1 - \frac{2}{nk(2n - 3k - 1)} \sum_{i=1}^{n} \sum_{j \in U_k(i)} (r(i,j) - k)$$

4. Evaluating PCA and t-SNE with Trustworthiness:

   - Trustworthiness was calculated for the reduced data with PCA and t-SNE.
   - Comparison: t-SNE performs better in preserving neighborhood structure (0.6744 vs. 0.5170). This is logical, as t-SNE focuses on local structures, while PCA preserves overall variance and may lose local structures.

5. Examining the Effect of Perplexity and Plotting KL Divergence:

   - t-SNE was run with Perplexity values = [5, 10, 20, 30, 40, 50].
   - KL Divergence was calculated for each Perplexity, and its plot was drawn.
   - Results:
     - The KL Divergence plot showed that as Perplexity increases, KL Divergence decreases:
       - Perplexity = 5: KL Divergence ≈ 1.90
       - Perplexity = 10: KL Divergence ≈ 1.88
       - Perplexity = 20: KL Divergence ≈ 1.83
       - Perplexity = 30: KL Divergence ≈ 1.79
       - Perplexity = 40: KL Divergence ≈ 1.75
       - Perplexity = 50: KL Divergence ≈ 1.70
     - Optimal value: Perplexity = 50 with the lowest KL Divergence (1.70).
     - Low Perplexity (e.g., 5) overemphasizes local structures and may introduce noise.
     - With increasing Perplexity, a better balance between local and global structures is achieved. At Perplexity = 50, t-SNE provided the best match between original and reduced distributions.

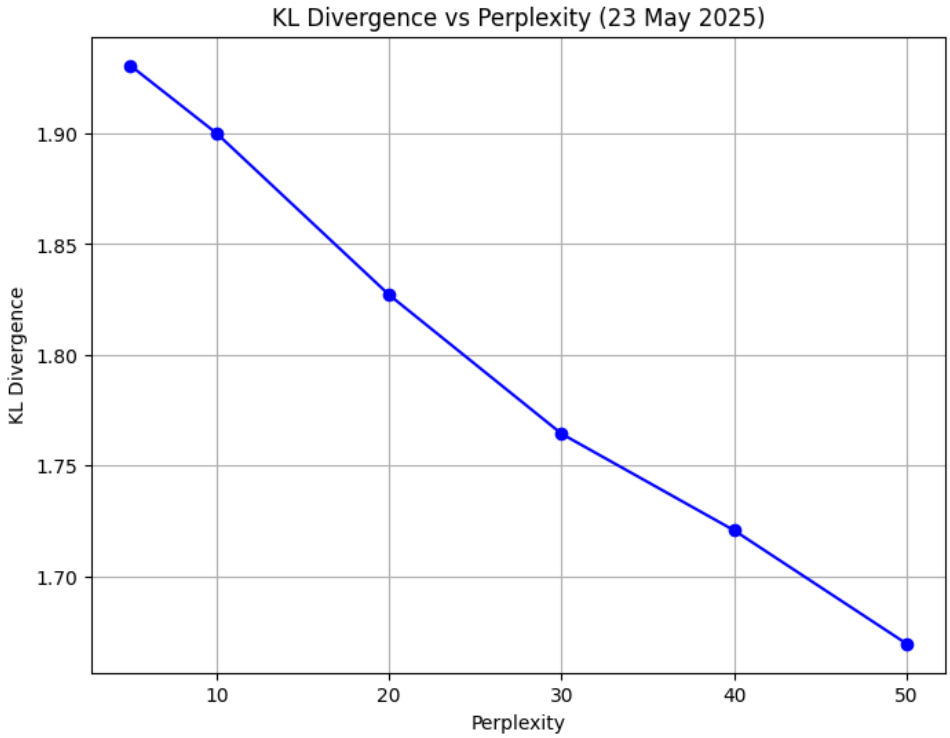

Figure 3: KL Divergence vs. Perplexity plot.

2.4 Results for the Entire Second Section

1. Data Preparation:
   - MNIST images were converted to 784-dimensional vectors (pre-done in loaded data).
   - Normalization was performed in seconds.
2. Dimensionality Reduction:
   - Output: Data reduced to two-dimensional space (10000, 2).
3. Trustworthiness:
   - PCA: 0.5170
   - t-SNE: 0.6744
   - t-SNE performed better in preserving local structures.
4. Effect of Perplexity:
   - KL Divergence plot showed decreasing KL Divergence with increasing Perplexity:

Perplexity	KL Divergence
5	≈ 1.90
10	≈ 1.88
20	≈ 1.83
30	≈ 1.79
40	≈ 1.75
50	≈ 1.70

Table 1: KL Divergence values for different Perplexity levels.

- Optimal Perplexity = 50, with the lowest KL Divergence.

2.5 Analysis

- Comparison of PCA and t-SNE: The higher Trustworthiness of t-SNE (0.6744) compared to PCA (0.5170) shows that t-SNE is more suitable for complex data like MNIST, as it better preserves local structures.
- Effect of Perplexity: The decrease in KL Divergence with increasing Perplexity indicates improved balance between local and global structures. Perplexity = 50 provided the best result.

References

J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

A. Joshi, P. Singh, and B. Raman. IsoMapGen: Framework for early prediction of peak ground acceleration using tripartite feature extraction and gated attention model. *Computers & Geosciences*, 196:105849, 2025. doi: 10.1016/j.cageo.2024.105849.

I. T. Jolliffe and J. Cadima. Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202, 2016. doi: 10.1098/rsta.2015.0202.

L. J. P. van der Maaten and G. E. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

D. Qiao, X. Ma, and J. Fan. Federated t-SNE and UMAP for Distributed Data Visualization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(19):20014–20023, 2025. doi: 10.1609/aaai.v39i19.34204.

J. Venna and S. Kaski. Neighborhood preservation in nonlinear projection methods: An experimental study. In *Proceedings of the International Conference on Artificial Neural Networks*, pages 485–491. Springer, Berlin, 2001.

Appendix

The complete project implementation, along with related documentation, is available in the GitHub repository below:

`https://github.com/Zohreh004/Implementation-Project-4-`