DES431 Lecture 4

Dimensionality Reduction and Data Visualization

Contents

In the current age of technology, the amount of data produced and collected are increased rapidly. However, too much data may not necessary be a good thing. In machine learning and big data analytic fields, many dimensions of features may be the cause of a model's accuracy since some features or dimensions may be noises and may not contribute to the generalization behavior of the data as a whole. Moreover, it is hard to visualize high-dimension data for further analysis. This problem is known as the **curse of dimensionality**.

Dimensionality Reduction One way out of the curse of dimensionality is a group of techniques called **dimensionality reduction**. The dimensionality reduction techniques are used as tools to reduce the complexity of a model and avoid overfitting as well as reduce the effects of noises in the data. The dimensionality reduction techniques can be classified into two categories: feature selection and feature extraction. In feature selection techniques, we select a subset of the original features. In feature extraction techniques, we construct new features from the original set of features.

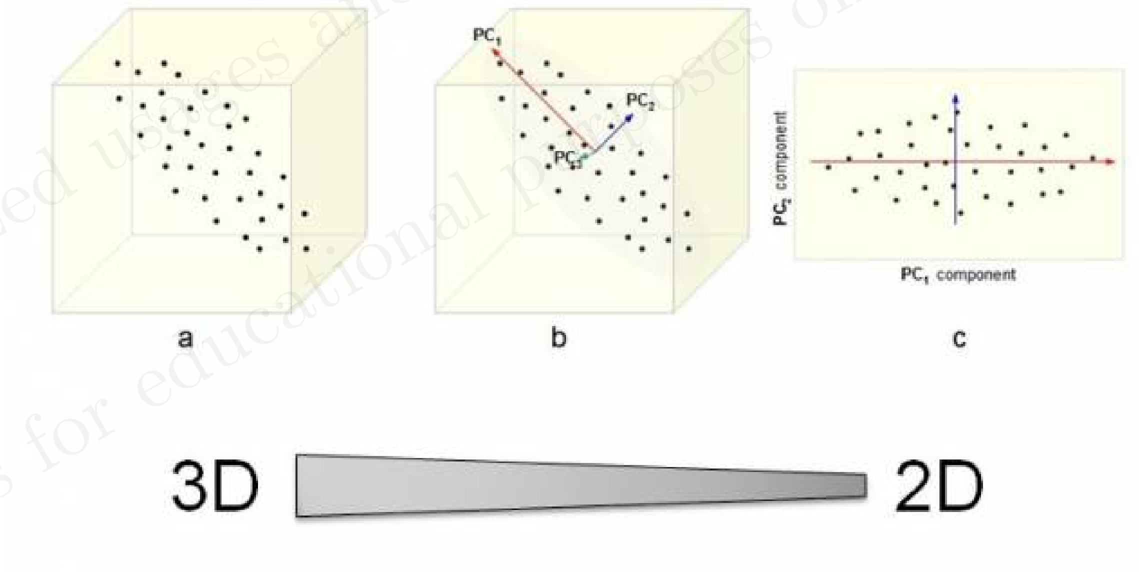
Data Visualization Data Visualization is the graphical representation of data. These graphical images display relationships of data and show data-driven insights in a way that is easy to understand. Low-dimensional data can be illustrated using simple visualization techniques such as graphs, charts, maps etc.

It is impossible for human being to visualize data in high-dimensional spaces. So when it comes to the world of big data, dimensionality reduction techniques are needed in order to reduce a high-dimensional space into 2D or 3D space. Different dimensionality reduction techniques produce different images.

In this lecture, we will discuss 3 popular dimensionality reduction and/or data visualization techniques which are the Principle Component Analysis (PCA), the t-distributed Stochastic Neighbor Embedding (t-SNE) and the Uniform Manifold Approximation and Projection for dimension reduction (UMAP).

4.1 The Principle Component Analysis (PCA)

The Principle Component Analysis (PCA) is an unsupervised linear transformation dimensionality reduction technique. It is widely used in the fields from bioinformatics to finances such as stock market prices. Intuitively, PCA identifies the directions of maximum variance (how far each number in the set is from the mean) in high-dimensional data and project onto a new space with equal or fewer dimensions.



Thus, PCA is a method that brings together: A measure of how each variable is associated with one another. (Covariance matrix.) The directions in which our data are dispersed. (Eigenvectors.) The relative importance of these different directions. (Eigenvalues.)

Summary of PCA algorithm

1. Standardize the dataset with n dimensions.
2. Construct the covariance matrix
3. Find eigenvalues and eigenvectors of the covariance matrix
4. Construct a transformation by using m eigenvectors associated with top m highest eigenvalues
5. Transform n-dimensional data using W to obtain m-dimensional data. This new m-dimensional space is what we call feature space.

4.1.1 Step 1: Standardize the dataset

Standardization is a process to shift each input variable (dimension) into a distributions having a mean of zero and a standard deviations of one. It is useful when input dataset have large differences between ranges or measured in different units.

Let μ_j be the mean of and σ_j be the standard deviation of a feature (dimension) j. The standardized data is defined as

$$\bar{x}_j^i = \frac{x_j^i - \mu_j}{\sigma_j},$$

where x_j^i is a jth dimension value of a data point x^i.

Note: the new mean $\bar{\mu}_j$ of standardized data is zero.

Step 1: Example

Suppose we have a dataset

$$\{[1,2,3,4],[5,5,6,7],[1,4,2,3],[5,3,2,1],[8,1,2,2]\}$$

This dataset has 5 data points in a dataset with 4 features (dimensions). We create a 5×4 matrix that represents this dataset

$$X = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 5 & 6 & 7 \\ 1 & 4 & 2 & 3 \\ 5 & 3 & 2 & 1 \\ 8 & 1 & 2 & 2 \end{bmatrix}$$

Calculating a mean and a standard deviation of each dimension, we have

$$\begin{array}{llll} \mu_1 = 4 & \mu_2 = 3 & \mu_3 = 3 & \mu_4 = 3.4 \\ \sigma_1 = 3 & \sigma_2 = 1.58114 & \sigma_3 = 1.73205 & \sigma_4 = 2.30217 \end{array}$$

The standardized dataset is

$$\overline{X} = \begin{bmatrix} \frac{1-4}{3} & \frac{2-3}{1.58114} & \frac{3-3}{1.73205} & \frac{4-3.4}{2.30217} \\ \frac{5-4}{3} & \frac{5-3}{1.58114} & \frac{6-3}{1.73205} & \frac{7-3.4}{2.30217} \\ \frac{1-4}{3} & \frac{4-3}{1.58114} & \frac{2-3}{1.73205} & \frac{3-3.4}{2.30217} \\ \frac{5-4}{3} & \frac{3-3}{1.58114} & \frac{2-3}{1.73205} & \frac{1-3.4}{2.30217} \\ \frac{8-4}{3} & \frac{1-3}{1.58114} & \frac{2-3}{1.73205} & \frac{2-3.4}{2.30217} \end{bmatrix} = \begin{bmatrix} -1 & -0.63 & 0 & 0.26 \\ 0.33 & 1.26 & 1.73 & 1.56 \\ -1 & 0.63 & -0.58 & -0.17 \\ 0.33 & 0 & -0.58 & -1.04 \\ 1.33 & -1.26 & -0.58 & -0.61 \end{bmatrix}$$

4.1.2 Step 2: Construct the covariance matrix

If the dataset have n dimensions (features), then the covariance matrix is an $n \times n$ matrix, defined as

$$C = \begin{bmatrix} cov(d_1, d_1) & cov(d_1, d_2) & \cdots & cov(d_1, d_n) \\ cov(d_2, d_1) & cov(d_2, d_2) & \cdots & cov(d_2, d_n) \\ \cdots & \cdots & \ddots & \cdots \\ cov(d_n, d_1) & cov(d_n, d_1) & \cdots & cov(d_n, d_n) \end{bmatrix},$$

where d_j is dimension j, $j = 1 \cdots n$

$$cov(d_j, d_k) = \frac{1}{n} \sum_{i=1}^{n} (\bar{x}_j^i - \bar{\mu}_j)(\bar{x}_k^i - \bar{\mu}_k)$$

Step 2: Example From step 1, the standardized dataset matrix is

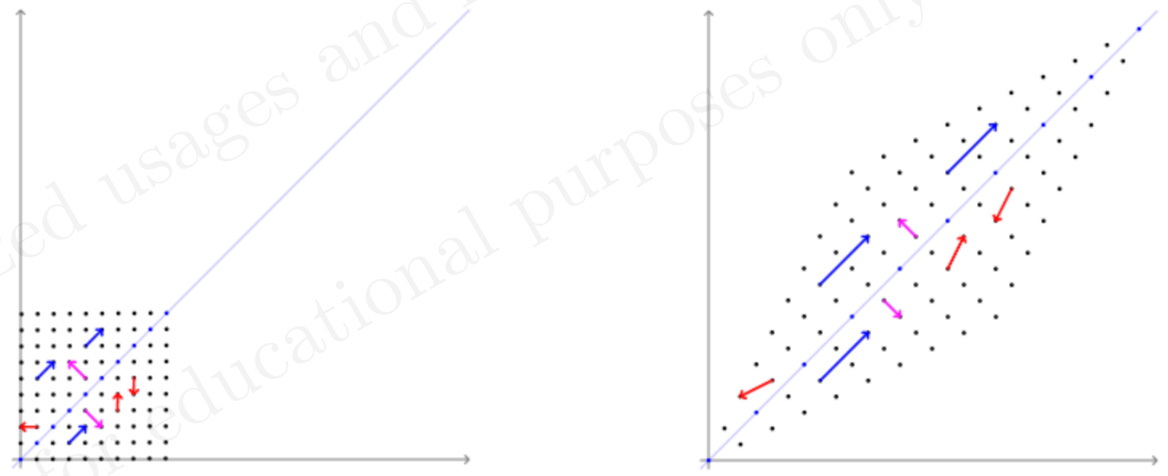
$$\overline{X} = \begin{bmatrix} -1 & -0.63246 & 0 & 0.26062 \\ 0.33333 & 1.26491 & 1.73205 & 1.56374 \\ -1 & 0.63246 & -0.57735 & -0.17375 \\ 0.33333 & 0 & -0.57735 & -1.04249 \\ 1.33333 & -1.26491 & -0.57735 & -0.60812 \end{bmatrix}$$

Then the covariance matrix of the dataset X is

$$C = \begin{bmatrix} 0.8 & -0.25298 & 0.03846 & -0.14479 \\ -0.25298 & 0.8 & 0.51121 & 0.4945 \\ 0.03849 & 0.52221 & 0.8 & 0.75236 \\ -0.14479 & 0.4945 & 0.75236 & 0.8 \end{bmatrix}$$

4.1.3 Step 3: Find eigenvalues and eigenvectors of the covariance matrix

An eigenvalues λ and an eigenvector $\mathbf{v}$ of a matrix A are a scalar and a non-zero vector, respectively, that satisfie $A\mathbf{v} = \lambda\mathbf{v}$. Geometrically speaking, an eigenvector $\mathbf{v}$ is stretched by a factor of λ under the transformation A in with no change in direction, i.e. $\mathbf{v}$ is not rotated under A.



Note: Eigenvectors of the covariance matrix is linearly independent and orthogonal.

Step 3: Example Eigenvalues and eigenvectors of the covariance matrix are

$$\left\{\lambda_1 = 2.52, \mathbf{v}_1 = \begin{bmatrix} 0.16 \\ -0.52 \\ 0.59 \\ -0.97 \end{bmatrix}\right\}, \left\{\lambda_2 = 1.07, \mathbf{v}_2 = \begin{bmatrix} -0.92 \\ 0.21 \\ -0.32 \\ -0.12 \end{bmatrix}\right\},$$

$$\left\{\lambda_3 = 0.39, \mathbf{v}_3 = \begin{bmatrix} -0.31 \\ -0.82 \\ 0.19 \\ 0.45 \end{bmatrix}\right\}, \left\{\lambda_4 = 0.03, \mathbf{v}_4 = \begin{bmatrix} 0.20 \\ 0.12 \\ -0.72 \\ 0.65 \end{bmatrix}\right\}$$

4.1.4 Step 4: Construct a transformation

Once we find all eigenvalues and eigenvectors of the covariance matrix, we need to sort eigenvalues in the descending order. If we want to reduce the dimension of the dataset X from n dimensions, to m dimensions, then we need to pick m eigenvectors corresponding to the top m highest eigenvalues to construct a transformation matrix W where each column of W is a selected eigenvector.

Intuitively, we are picking m directions (eigenvectors) or new dimensions such that the standardized data have the highest variances (eigenvalues) from the covariance matrix.

Step 4: Example If we want to transform the dataset X from 4 dimensions into 2 dimensions, then we select

$$\left\{\lambda_1 = 2.52, \mathbf{v}_1 = \begin{bmatrix} 0.16 \\ -0.52 \\ 0.59 \\ -0.97 \end{bmatrix}\right\}, \left\{\lambda_2 = 1.07, \mathbf{v}_2 = \begin{bmatrix} -0.92 \\ 0.21 \\ -0.32 \\ -0.12 \end{bmatrix}\right\}$$

to construct

$$W = \begin{bmatrix} 0.16 & -0.92 \\ -0.52 & 0.21 \\ 0.59 & -0.32 \\ -0.97 & -0.12 \end{bmatrix}$$

4.1.5 Step 5: Transform n-dimensional data using W to obtain m-dimensional data

Compute $\overline{X}W$ to obtain the feature space

Step 5: Example

4.2 The t-distributed stochastic neighbor embedding (t-SNE)

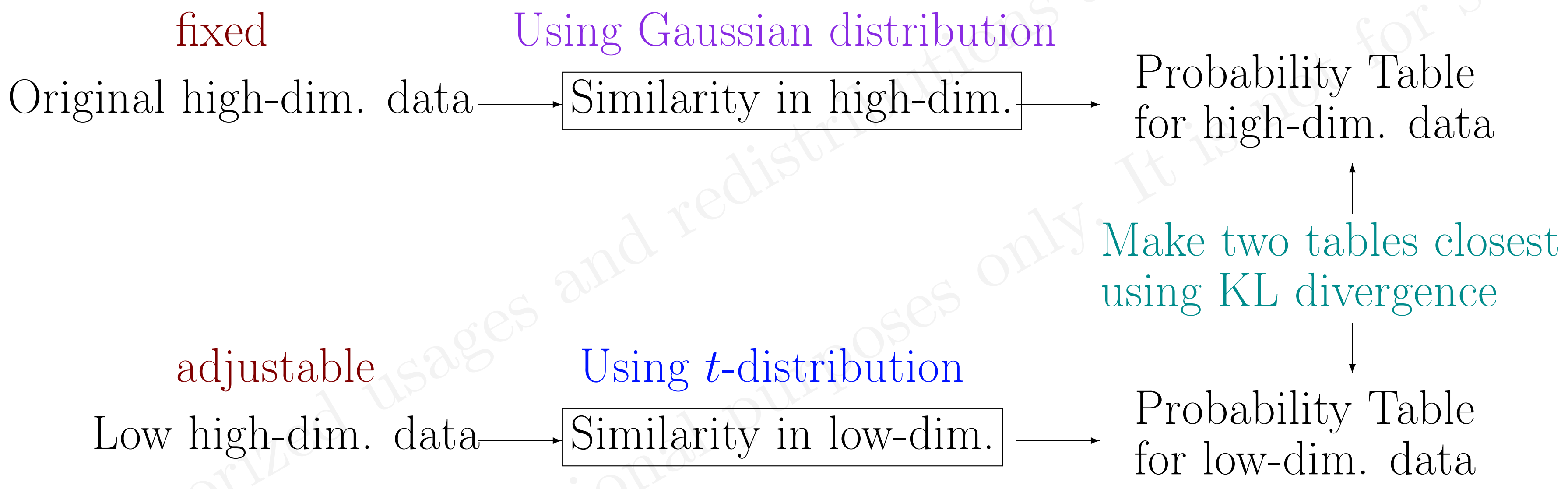
The t-distributed stochastic neighbor embedding or t-SNE is an unsupervised, randomized, non-linear dimensionality reduction. The t-SNE aims to keep similar data points close together in lower-dimensional space.

Summary of the t-SNE algorithm

1. Find similarity p_{ij} between any two data points x^i and x^j in the original high-dimensional dataset using Gaussian distribution

2. Map each data points in high-dimensional space into lower-dimensional space. Then find similarity q_{ij} between any two data points y^i and y^j in the low-dimensional space, where y^i is mapped from x^i and y^j is mapped from x^j

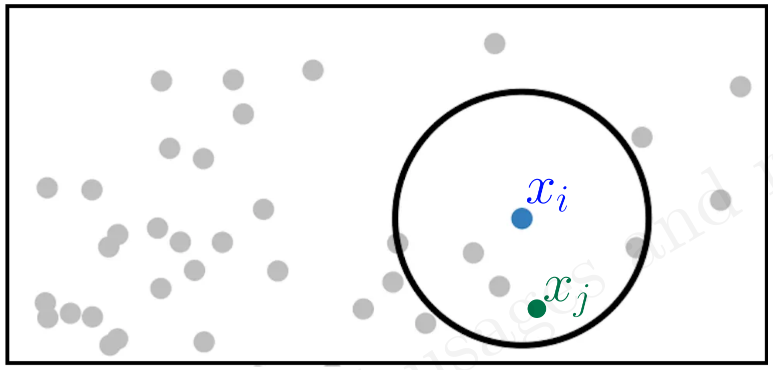
3. Apply KL divergence and gradient descent to minimize the difference between p_{ij} and q_{ij}


fixed
Using Gaussian distribution
Original high-dim. data
Similarity in high-dim.
Probability Table for high-dim. data
Make two tables closest using KL divergence
adjustable
Using t-distribution
Low high-dim. data
Similarity in low-dim.
Probability Table for low-dim. data

Overview of t-SNE

4.2.1 Step 1: Find similarity in high-dimensional space

A given center of Gaussian distribution x^i, a probability that x^j is in the neighborhood of size (perplexity) σ_i is given by



$$p(j|i) = \frac{\exp\left(-||x^i - x^j||^2/(2\sigma_i^2)\right)}{\sum_{k\neq i}\exp\left(-||x^i - x^k||^2/(2\sigma_i^2)\right)}$$

A similarity between x^i and x^j is defined as

$$p_{ij} = \frac{p(j|i) + p(i|j)}{2N},$$

where N is the number of data points.

4.2.2 Step 2: Map high-dimensional space into low-dimensional space and Find similarity in low-dimensional space

Mapping of high-dimensional space into low-dimensional space can be done using PCA or just simply randomly assign a point y^i in a low-dimensional space to a point x^i in the original high-dimensional space.

A similarity between y^i and y^j in a low-dimensional space is given by

$$q_{ij} = \frac{\exp\left(-||y^i - y^j||^j\right)}{\sum_k \sum_{l \neq k} -||y^l - y^k||^2}$$

This similarity is based on the Student's t-distribution.

4.2.3 Step 3: Apply KL divergence

Measure distance between distribution (probability matrix) $P = [p_{ij}]$ and distribution (probability matrix) $Q = [q_{ij}]$ using KL divergence

$$KL(Q||P) = \sum_{ij} q_{ij} \log\left(\frac{q_{ij}}{p_{ij}}\right)$$

Then minimize $KL(Q||P)$ using the gradient descent to update the mapping into low-dimensional space.

4.3 The Uniform Manifold Approximation and Projection for Dimension Reduction (UMAP)

UMAP is an algorithm for dimensionality reduction and visualization based on manifold learning techniques and topological data analysis (TDA). UMAP is designed to ensure that local structures are preserved in balance with the global structure.

Summary of UMAP algorithm

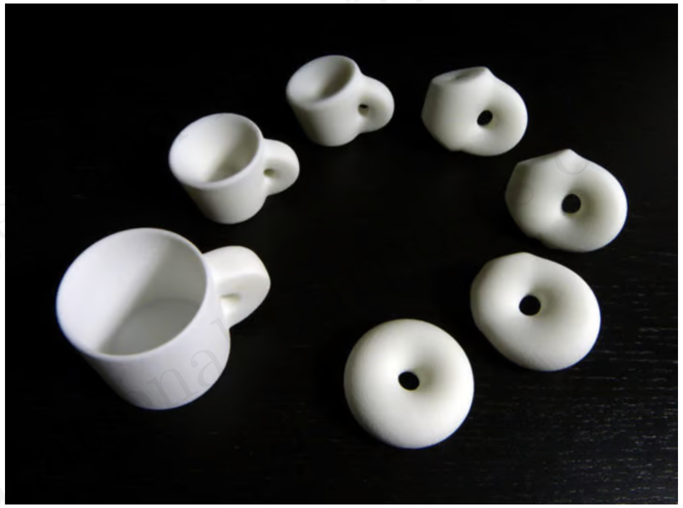
1. Find a fuzzy topological structure of high-dim. data.
2. Find a low-dim. space that has similar fuzzy topological structure as the original one.

$$\begin{tikzcd}
\text{Original high-dim. data} \arrow[d] & \\
\boxed{\text{Construct Fuzzy Simplicial set}} \arrow[d] & \begin{array}{l}\text{- Using Manifold learning and TDA} \\ \text{- n\_neighbors parameter}\end{array} \\
\text{Fuzzy Topological Structure} \arrow[d] & \\
\boxed{\text{Find Optimal embedding}} \arrow[d] & \begin{array}{l}\text{- Using Cross-Entropy} \\ \text{- min\_dist parameter}\end{array} \\
\text{Low-dim. data} &
\end{tikzcd}$$

Overview of UMAP

4.3.1 What is Topology

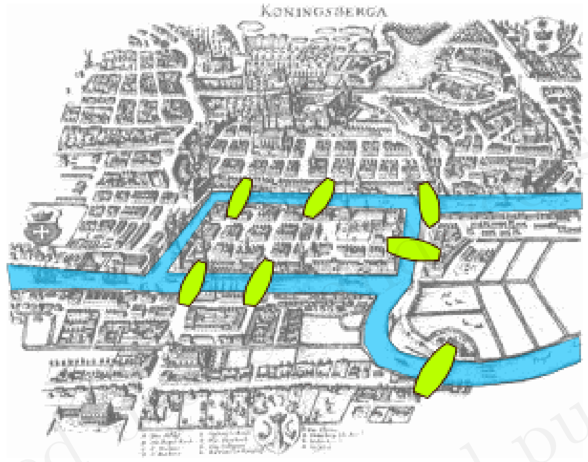
Topology is a field of study in Mathematics. It concerns with the properties of geometric object that are preserved under continuous deformations such as stretching, twisting, bending without closing holes, making new holes, tearing, gluing or passing through itself

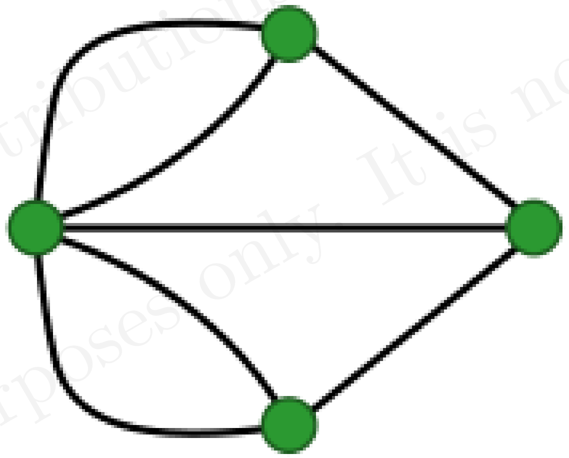


Topological joke: coffee mug = donut

The Seven Bridges of Konigsberg Problem

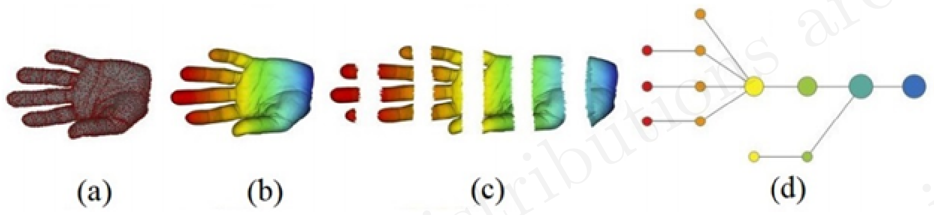
Can you find a walk through the city that would cross each of the seven bridges once and only once?





Euler, using topology, said "NO!".

Topological Data Analysis (TDA) is a data analysis technique that uses topological tools to transform information in high dimension form it to simplicial complexes in low dimension while preserve the notion of closeness.

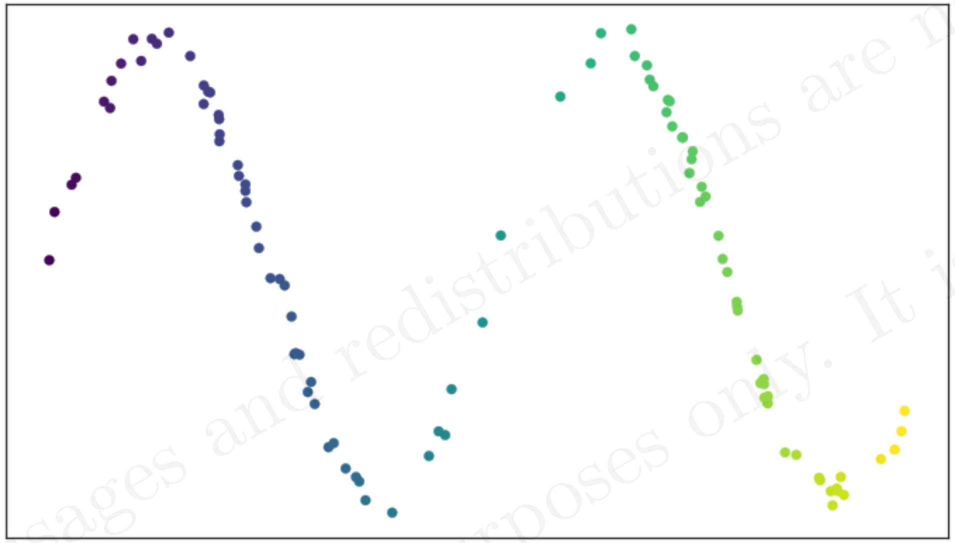


Manifold is a geometric object that locally similar to Euclidean space but globally not necessarily Euclidean.

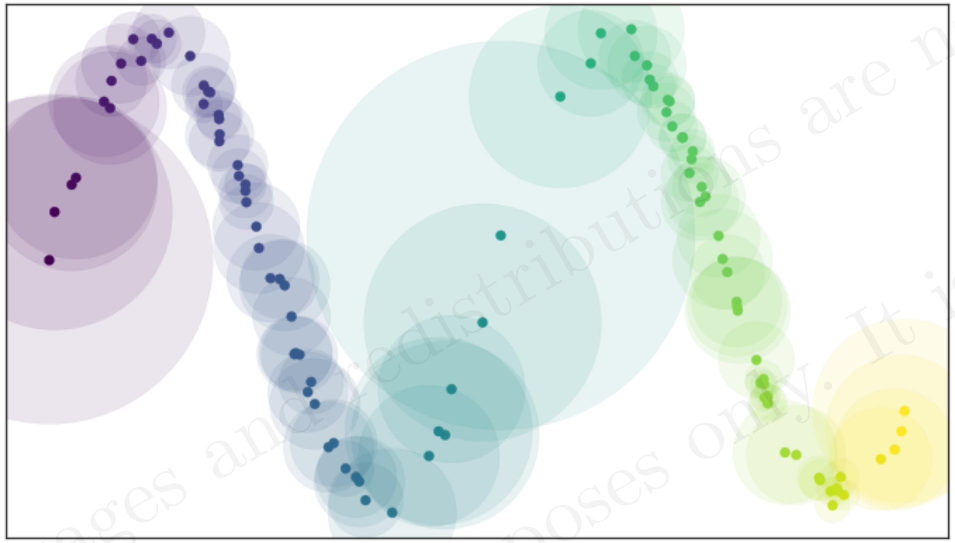
Manifold Learning is a class of unsupervised estimators that try to describe dataset as low-dimensional manifolds embedded in high-dim. space.

4.3.2 UMAP: Step 1 Constructing fuzzy topological structure

Consider a dataset like



Create a “ball” for each data point so that it contains k neighbors (n_neigbors = k).

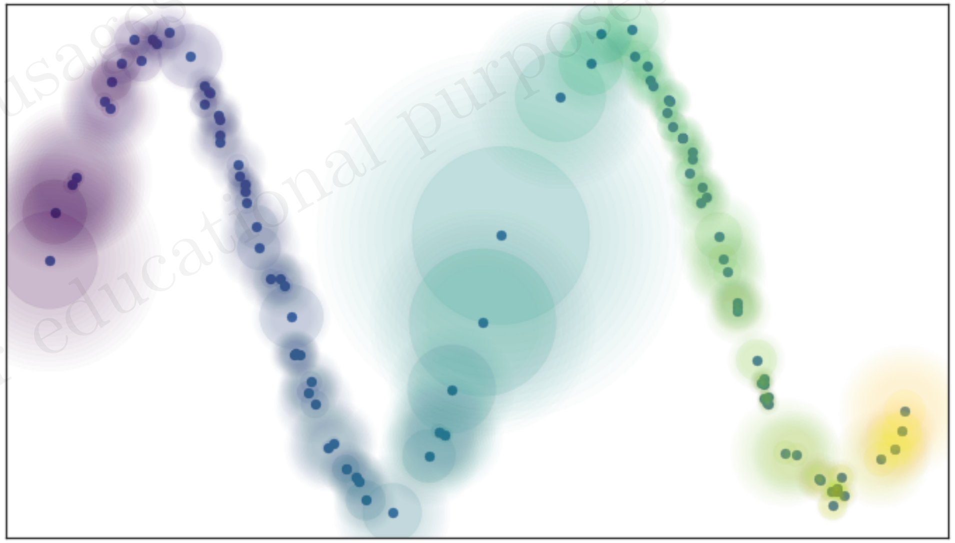


n_neigbors =3

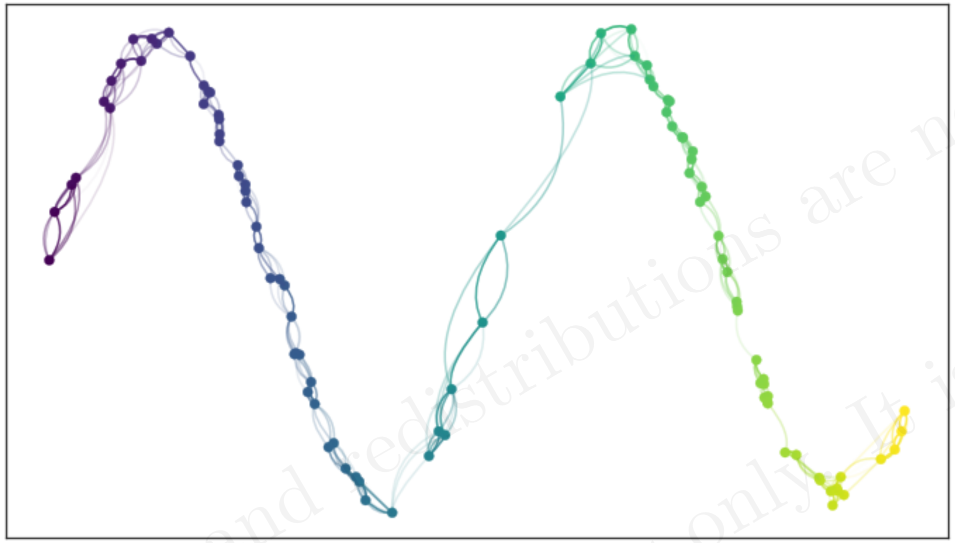
Make each ball "fuzzy" by using exponential probability distribution. This means that the probability that a point x^j is in the ball centered at x^i is given by

$$p(j|i) = \exp\left(-\frac{\max(0, d(x^i, x^j) - \rho_i)}{\sigma_i}\right),$$

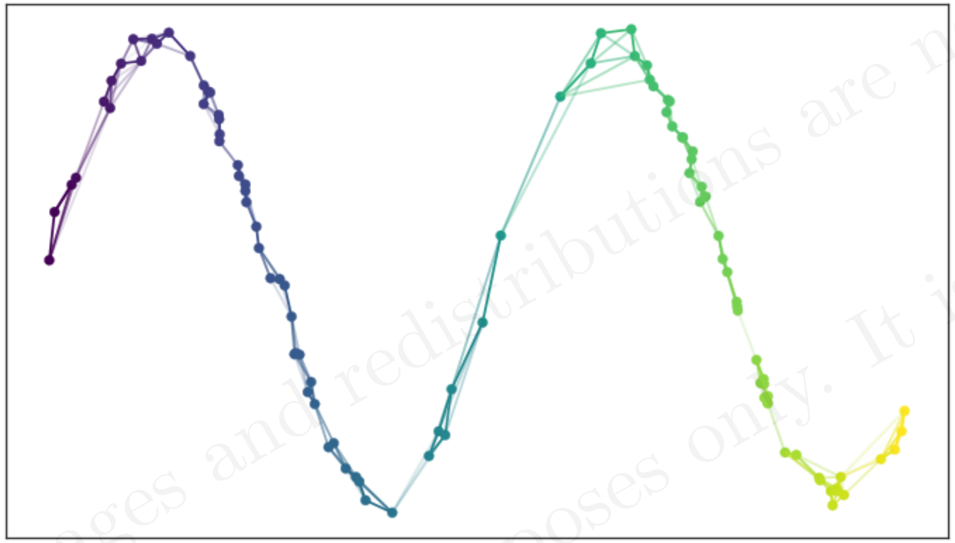
where ρ_i is a distance from x^i to the nearest neighbor and σ_i is a normalization factor (how the shadow is faded out) local to the point x^i This is to ensure the weighted connectivity of local structures



Draw weighted line between two points if their ball are intersected.



Merge edges between two points to get a single weighted line (If two edges have weights of **a** and **b**, then the weight of the combined edge is $\mathbf{a}+\mathbf{b}-\mathbf{a}\cdot\mathbf{b}$



4.3.3 UMAP: Step 2 Finding a Low Dimensional Representation

1. Fuzzy topological structure from the previous step can be viewed as weighted graph. UMAP uses a spectral layout to initialize the embedding via constructing the Laplacian matrix and solving the eigenvalue decomposition problem

$$L = D^{1/2}(D - A)D^{1/2}$$

   This provides both faster convergence and greater stability within the algorithm. And the distance probability in low-dim space is given by

$$q_{ij} = (1+a(y^i-y^j)^{2b})^{-1} \approx \begin{cases} 1 & \text{if } y^i - y^j \leq \text{min\_dist}, \\ \exp\left(-(y^i - y^j) - \text{min\_dist}\right) & \text{if } y^i - y^j > \text{min\_dist} \end{cases}$$

2. The Cross-Entropy loss function is used to measure the difference between the fuzzy topological structure of the original high-dim. data and that of

the low-dim. embedded space.

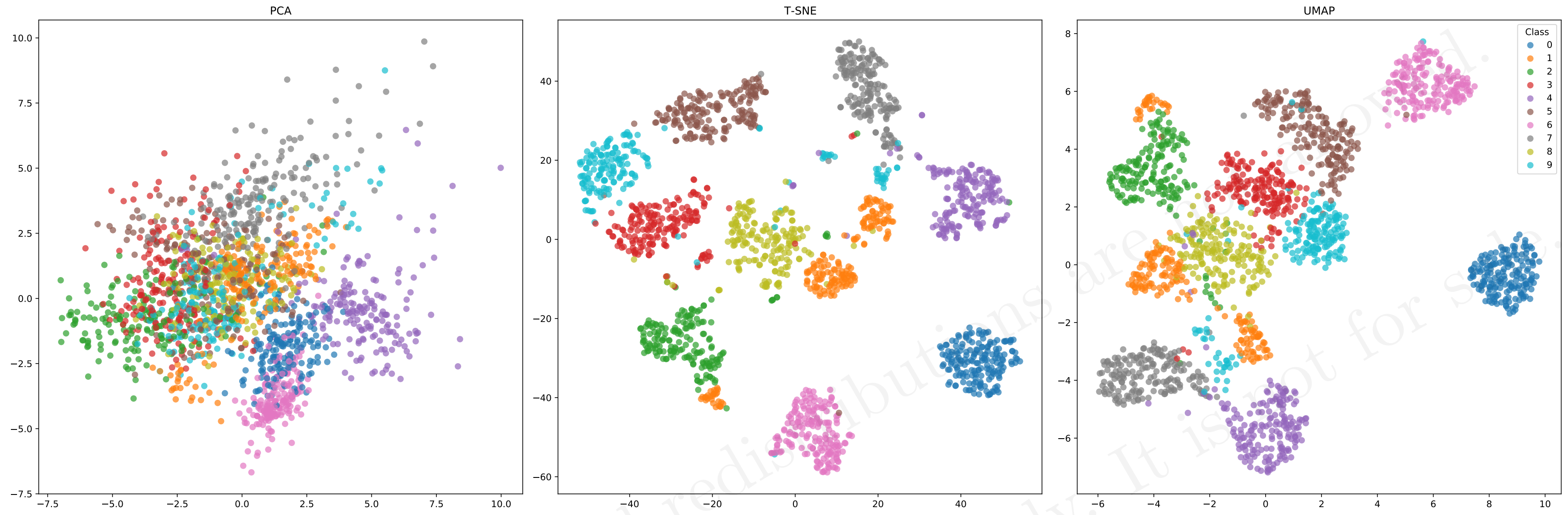
$$CE(X,Y) = \sum_i \sum_j \left[p_{ij}(X) \log \left(\frac{p_{ij}(X)}{q_{ij}(Y)} \right) + (1 - p_{ij}(X)) \log \left(\frac{1 - p_{ij}(X)}{1 - q_{ij}(Y)} \right) \right]$$

Stochastic Gradient Descent (instead of gradient descent) is used to minimize the loss function.

4.4 PCA vs. *t*-SNE vs. UMAP

	PCA	***t*-SNE**	**UMAP**
Aim	Project a dataset onto low-dim. by identifying directions of maximum variance	Embedding high-dim data on low-dim. space while keeping similar instances close and dissimilar instances apart (no necessary preserve global structure	Embedding high-dim data on low-dim. space while preserving local and global connectivities
Type	Linear Projection	Non-linear Embedding	Non-Linear Embedding

	PCA	***t*-SNE**	**UMAP**
How it works	Use the covariance matrix to find the Principle components (Eigenvalues and Eigenvectors). The project data onto a space where top k Principle Components form a basis	Use the Gaussian distribution in high-dim. and t-distribution in low-dim. to create probability table. Then use KL divergence and the gradient descent to minimize the difference	Use Manifold Learning and TDA to find a fuzzy topological structure of high-dim. data. Then use cross-entropy and stochastic gradient descent to find low-dim. representation.
Time	Fast	Relatively Slow	Relatively Fast


PCA
T-SNE
UMAP
Class
0
1
2
3
4
5
6
7
8
9