DATA MINING

Dimensionality Reduction

Manifold Learning

Ashish Pujari

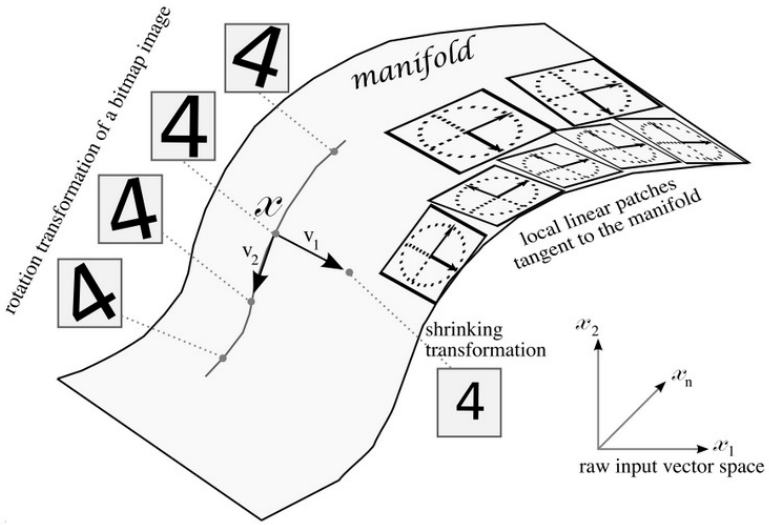
Lecture Outline

- Manifold Learning
- Algorithms
  - Kernel PCA
  - MDS
  - t-SNE
  - U-MAP

MANIFOLD LEARNING

Manifold Learning

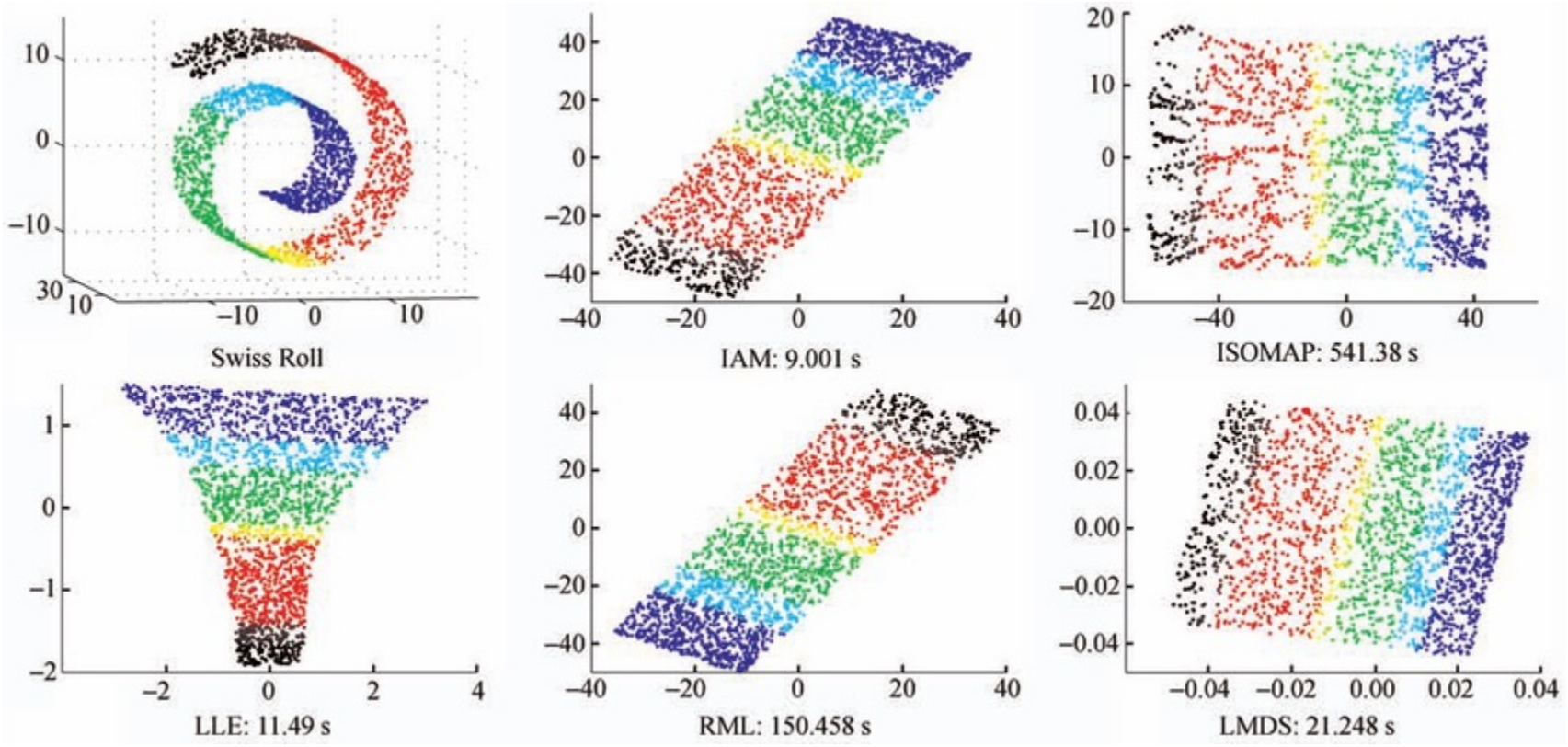
- Approach to non-linear dimensionality reduction
- Manifold Hypothesis
  - Dimensionality of many data sets is only artificially high
  - Observed data lie on a low-dimensional manifold embedded in a higher-dimensional space i.e.
  - Mostly unsupervised, though supervised variants exist
- Examples
  - PCA – linear manifold
  - Autoencoders - non-linear manifold



Space of natural images has the differential-geometric structure of a low-dimensional manifold embedded in the high-dimensional pixel space

Manifold Learning: Global vs Local Structure

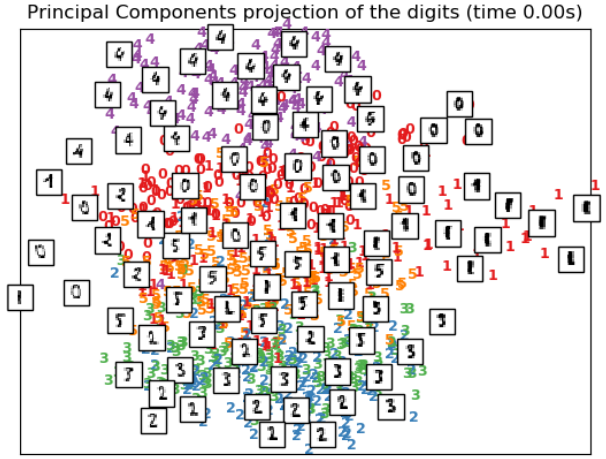
- Preserving Topology vs Intrinsic Geometry

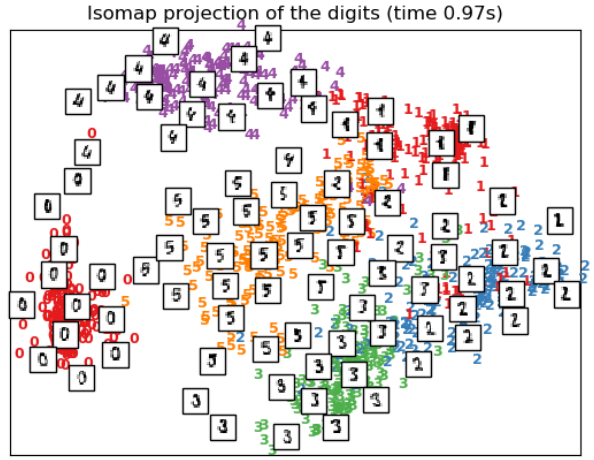


https://www.researchgate.net/publication/220585010_Incremental_Alignment_Manifold_Learning

Manifold Learning: Steps

- Problem
  - Given input data $X = \{x_1, x_2, \ldots, x_n\} \in \mathbb{R}^D$, we assume that the data points lie on a d-dimensional manifold embedded into $\mathbb{R}^D$, where $d < D$
  - Task is to learn the d-dimensional manifold structure M.
- Solution Approach
  1. Data Preprocessing
  2. Structure Representation (Kernel, Graph, etc.)
  3. Embedding (projection to low dimensions)
  4. Optimization Objective
  5. Hyperparameter Tuning
  6. Visualization/Interpretation
  7. Evaluation (if applicable)
  8. Refinement (if needed)



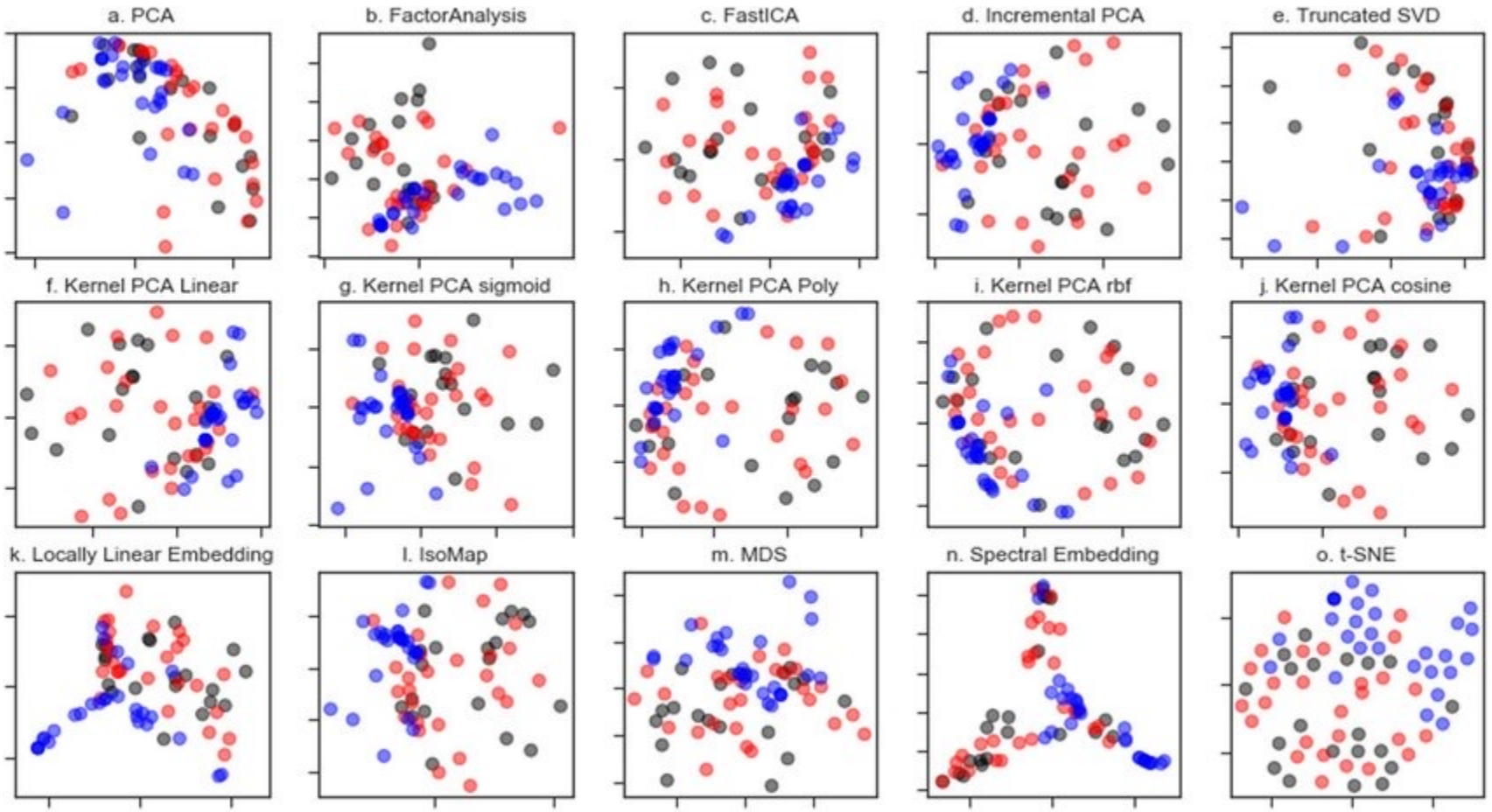


https://en.wikipedia.org/wiki/Nonlinear_dimensionality_reduction

Structure Representation

- Kernel-Based Methods: Maximize variance in the mapped space.
  - E.g., Kernel PCA, Kernel t-SNE.
- Matrix Factorization: Preserve pairwise dissimilarities.
  - E.g., Multidimensional Scaling (MDS).
- Graph-Based Methods: Minimize stress, preserve pairwise distances.
  - E.g., t-SNE, Isomap, Laplacian Eigenmaps.
- Autoencoder-Based Methods: Minimize reconstruction loss.
  - E.g., Variational Autoencoders (VAEs).
- Spectral Methods: Analyze eigenvalues and eigenvectors.
  - E.g., Spectral Embedding.

Manifold Learning : Algorithms



Dimensionality reduction methods applied to the FTIR dataset.

https://www.researchgate.net/figure/Dimensionality-reduction-methods-applied-to-the-FTIR-dataset-Matrix-decomposition_fig1_328200153

Distance/Similarity Measure

- Conditions that a distance measure must satisfy:
  - Non-negativity: $d(x, y) >= 0$
  - Identity: $d(x, y) = 0 \; if \; and \; only \; if \; x == y$
  - Symmetry: $d(x, y) = d(y, x)$
  - Triangle Inequality: $d(x, y) + d(y, z) >= d(x, z)$

- Examples:
  - Euclidean, Manhattan, Chebyshev, Minkowski, Geodesic, Hamming, etc.

Euclidean Distance

- Straight-line distances in a multidimensional space

$$D_{pq} = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

  - p and q are two points is Euclidean space
  - p_i q_i are Euclidean vectors starting from the origin of the n-dimensional space

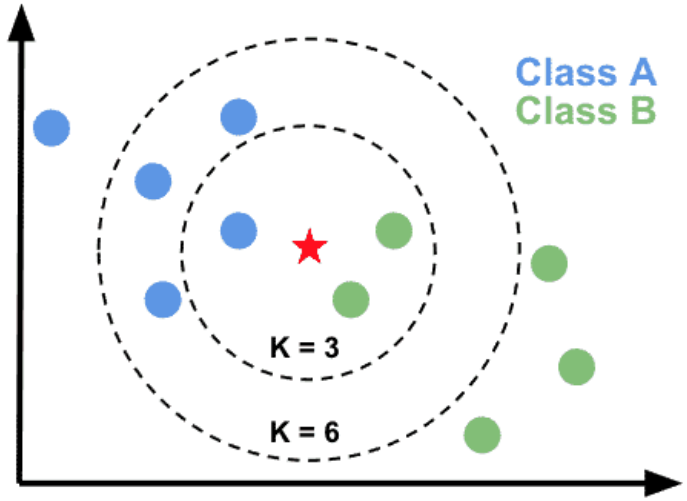
- Applications
  - Clustering based on spatial proximity, K-nearest neighbors algorithms.
  - Data with continuous features.

KNN (k-Nearest Neighbor)

- Learning algorithm to predict the label of data points by looking what is the majority in its closest neighbors

$$\mathrm{N}_k(x_i) = \{x \in X | \; x \text{ is a k-nearest neighbour of } x\_i\}$$

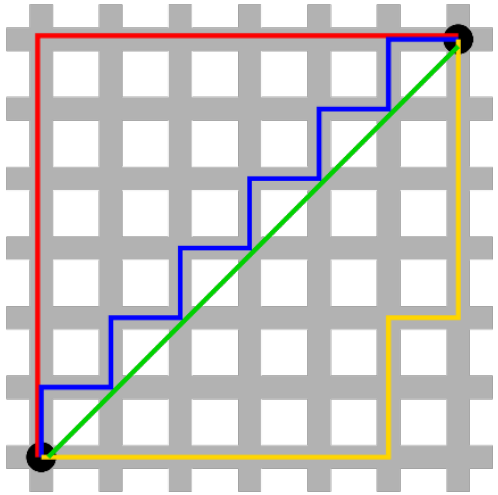
- Pros
  - Simple to implement and robust to noisy training data.
- Cons
  - Need to define which k value to use
  - High cost of computation compared to other algorithms.
  - Memory based data storage, so less efficient.

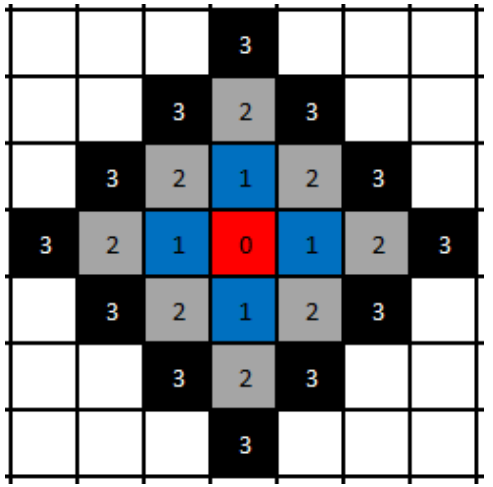


Manhattan/City Block Distance

- Calculates the distance between two data points in a grid-like path
- Example
  - In a plane with $p_1\ at\ (x_1, y_1)$ and $p_2\ at\ (x_2, y_2)$, it is

$$|x_1 - x_2| + |y_1 - y_2|$$





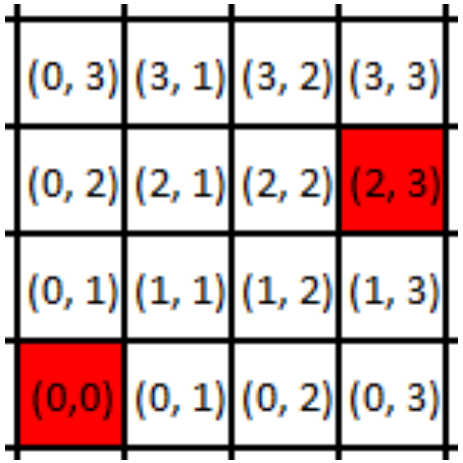
Chebyshev/Maximum metric Distance

- Measures maximum absolute difference along any dimension

$$D(p,q) = max(|p_1 - q_1|, |p_2 - q_2|, \ldots, |p_n - q_n|)$$

For two vectors $p = (p_1, p_2, \ldots, p_n)$ and $q = (q_1, q_2, \ldots, q_n)$

- Applications
  - Board games (e.g., chess), spatial planning, robustness to outliers.
- Example

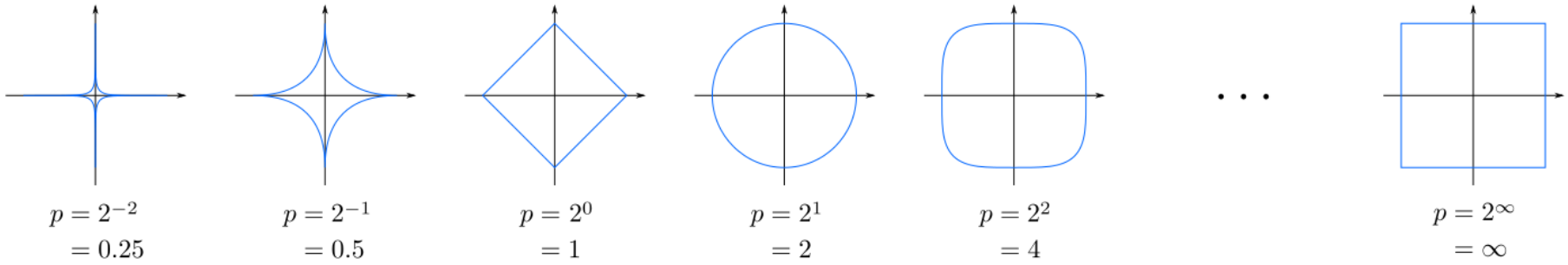


Minkowski Distance

- Distance metric intended for real-valued vector spaces
- Minkowski distance of order p (integer) between two points: $X = \{x_{1,}x_{2,}, \dots, x_{n,} \in \mathbb{R}^n\}$ and $Y = \{y_{1,}y_{2,}, \dots, y_{n,} \in \mathbb{R}^n\}$

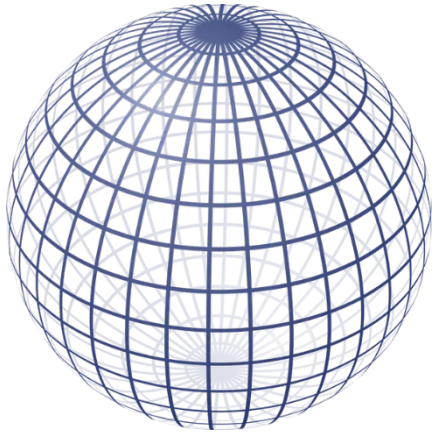
$$D(X,Y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{\frac{1}{p}}$$

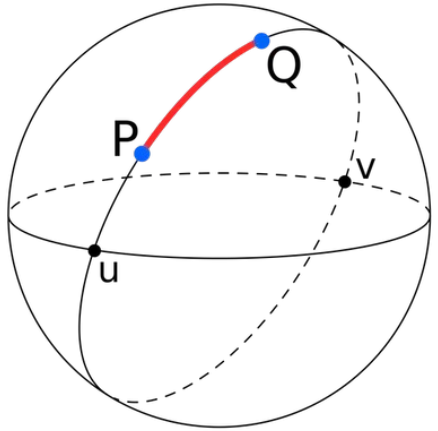
- Generalization of the Manhattan distance $(p = 1)$, Euclidean distance $(p = 2)$, and Chebyshev distance $(p \to \infty)$



Geodesic Distance

- A geodesic is the shortest path between two points in space in a curved manifold





Geodesic distance is calculated in a 3D spherical space as the distance across the curved surface of the world.

Source: Wikipedia

Entropy

- Entropy is used to measure the uncertainty of a system.
- Quantifies the number of bits required to encode and transmit an event from a probability distribution.

$$H(P) = H(p_1, p_2, .., p_n) = -\sum_x P(x) \log P(x)$$

- When there are n symbols, all equally probable, the probability of any of them is 1/n.

$$h = -\sum \frac{1}{n} (\log 1/n) = \log (n)$$

- Examples:
  - Fair Coin : $(½ * \log(1/2) + ½ * \log(1/2)) = \log(2) = 1$
  - Fair Dice : $(6 * (1/6 * \log(1/6))) = \log(6) = 2.58$

KL Divergence

- Measures how one probability distribution differs from a second distribution.
- Discrete Probability Distributions

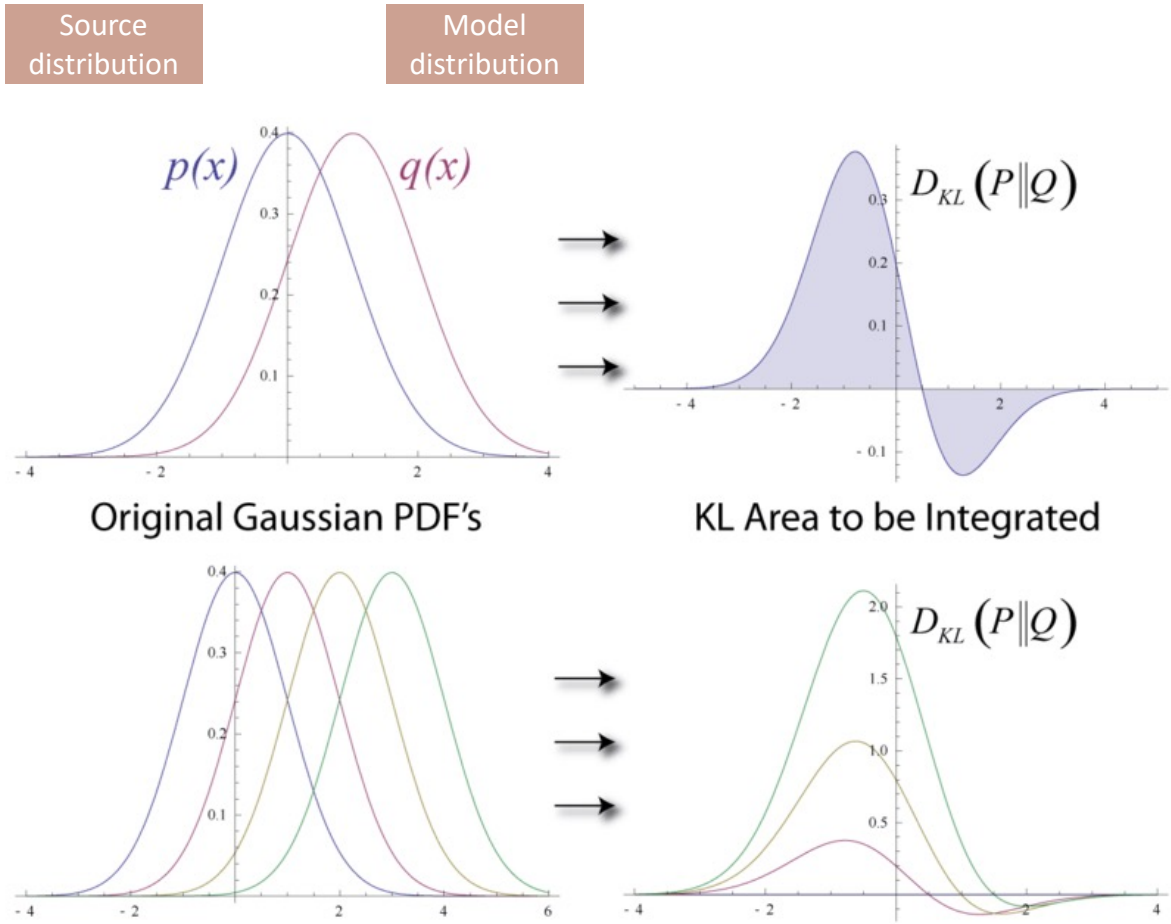
$$KL(P \parallel Q) = \sum_{x} P(x) \log \frac{P(x)}{Q(x)}$$

- Continuous Probability Distributions

$$KL(P \parallel Q) = \int_{-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)} d(x)$$

where p and q denote densities of P and Q

- KL divergence is non symmetric and non-negative



Cross Entropy

- Average number of bits used to transfer the information and always greater than or equal to the entropy

$$H(q,p) = H(p) + D_{KL}(p,q) = -\sum_i p_i log(q_i)$$

$$Cross\ Entropy\ = Entropy\ + KL\ Divergence$$

- Minimizing cross entropy is the same as minimizing the KL divergence
- Used as a loss function when optimizing logistic regression and neural networks

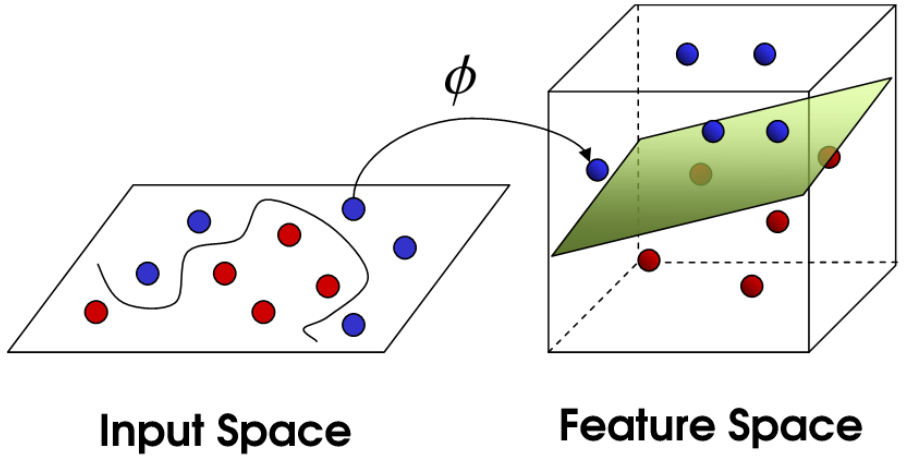
MANIFOLD LEARNING ALGORITHMS

Kernel PCA, MDS, t-SNE, UMAP

Kernel Function

- A kernel function takes as its inputs vectors in the original space and returns the dot product of the vectors in the feature space.
- If we have data $x, y \in X$ and a map $\boldsymbol{\varphi}$: X → $\mathbb{R}^{N}$ then below is a kernel function

$$\boldsymbol{k}(\boldsymbol{x}, \mathbf{y}) = \langle \boldsymbol{\varphi}(\boldsymbol{x}), \boldsymbol{\varphi}(\mathbf{y}) \rangle$$



Polynomial Kernel

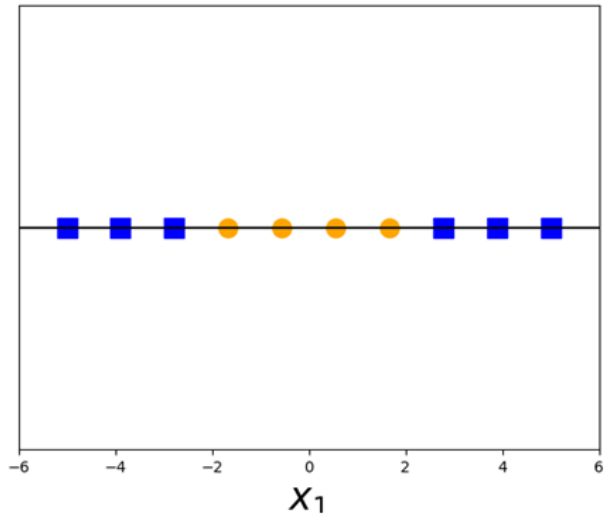
- Computes relationships between pairs of observations

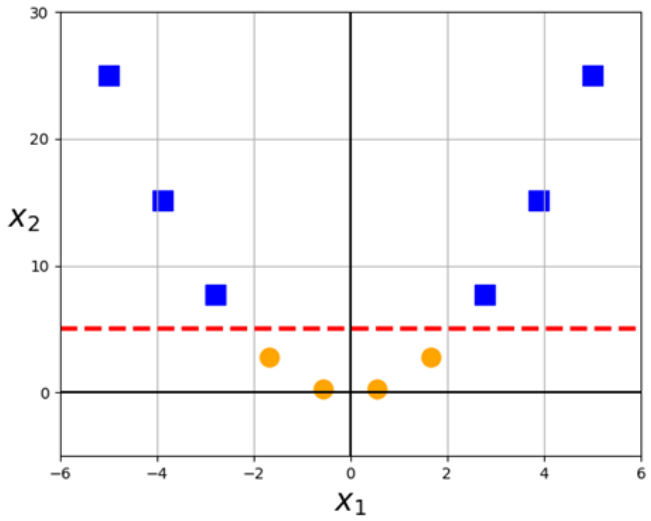
$$k(x, y) = (x^\top y + c_0)^d$$

  - x, y are the input vectors
  - c_0 is the coefficient
  - d is the kernel degree

- Linear Kernel
  - Special case of polynomial kernel where $d = 1$ and $c_0 = 0$

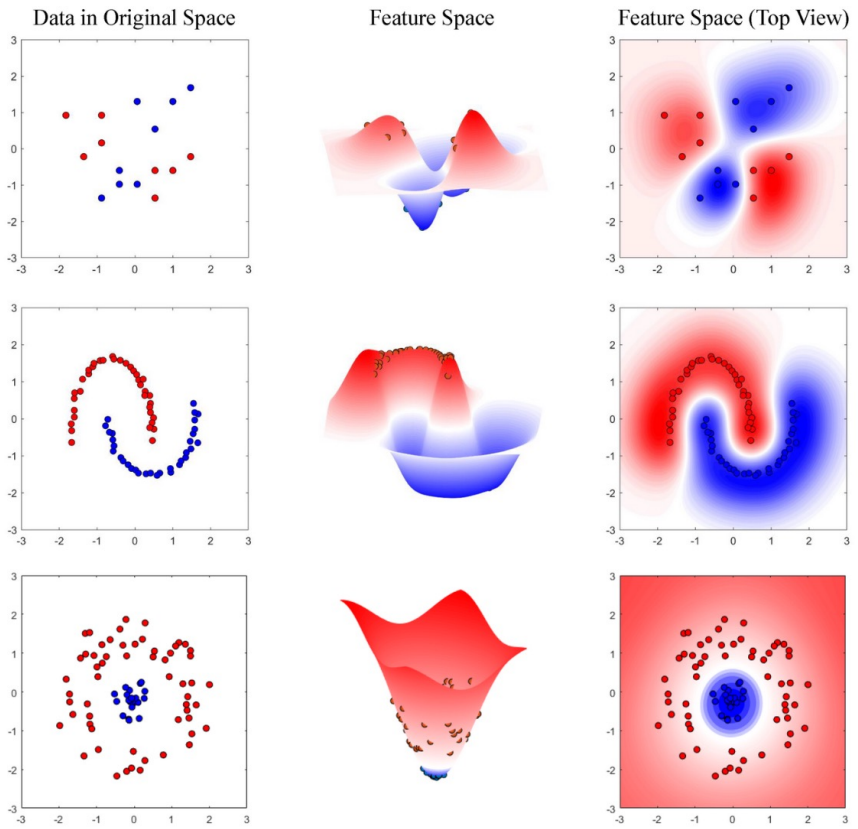
$$k(x, y) = x^\top y$$





Kernel PCA (KPCA)

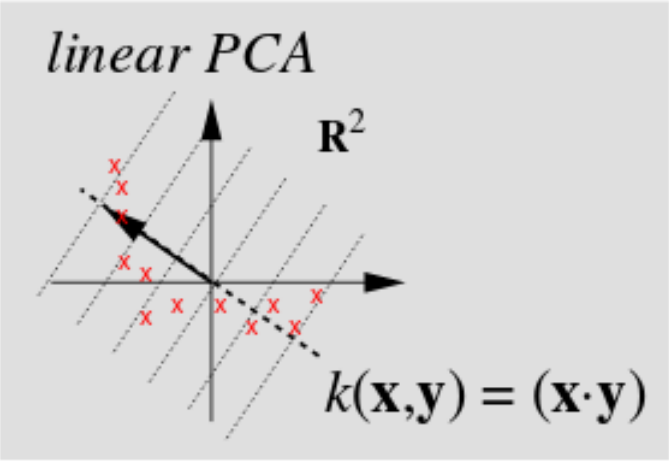
- Non-linear dimensionality reduction through the use of kernels
- Approach
  - Extends data to a high dimensional feature space using the kernel trick and extracts principal components in that space using PCA.
  - The result will be non-linear in the original data space.
- Challenges
  - Non-trivial to find a good kernel for a given problem; may not yield good results with standard kernels.
  - The kernel matrix is $n \; x \; n$, so kernel PCA can have difficulties for large data sizes.



PCA vs KPCA

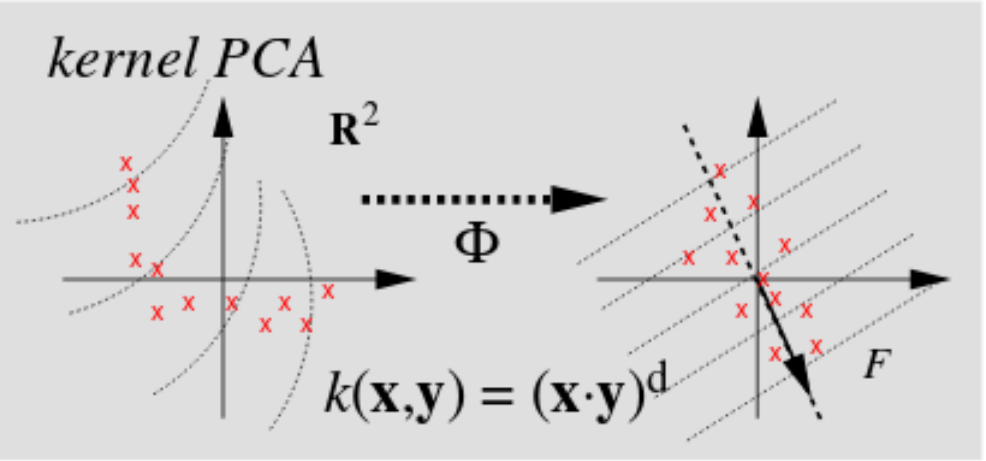
- Standard PCA is equivalent to Kernel PCA with linear kernel.
- PCAs covariance matrix scales with the number of input dimensions N

$$\text{Cov} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x_i}\,\mathbf{x_i}^T$$



- Unlike standard PCA, the eigenvectors of KPCA are not the principal component axes but the samples projected onto those axes.
- KPCAs kernel matrix scales with the number of data points

$$\text{Cov} = \frac{1}{N}\sum_{i=1}^{N} \phi(\mathbf{x_i})\phi(\mathbf{x_i})^T$$



Scholkopf B, Smola A, and Muller KR, Kernel principal component analysis, 1999

KPCA

- The kernel is related to the transform $\phi(\mathbf{x})$, and the resulting $n\ x\ n$ kernel matrix looks like this:

$$\mathbf{K} = \mathbf{k}(\mathbf{x}, \mathbf{y}) = (\phi(\mathbf{x}), \phi(\mathbf{y})) = \phi(\mathbf{x})^T \phi(\mathbf{y})$$

- Mean of the data in feature space

$$\mu = \frac{1}{n} \sum_{i=1}^{n} \phi(\boldsymbol{x_i}) = 0$$

- Covariance

$$C = \frac{1}{n} \sum_{i=1}^{n} \phi(\mathbf{x_i}) \phi(\mathbf{x_i})^T$$

- Eigenvectors

$$Cv = \lambda v$$

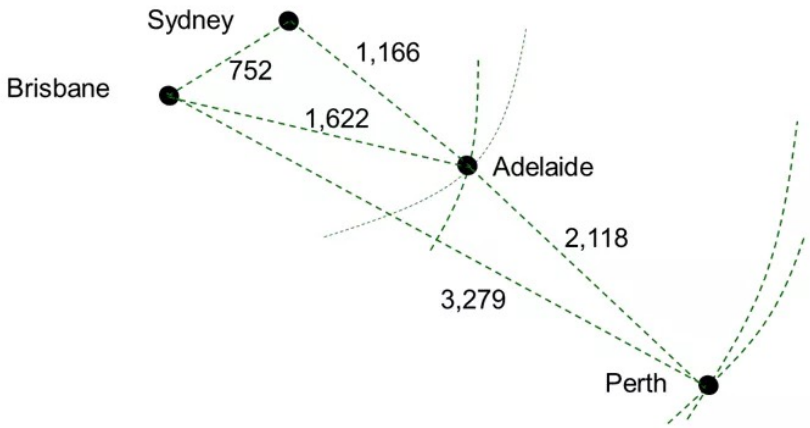
- Eigenvectors can be expressed as a linear combination of features

$$v = \sum_{i=1}^{n} \alpha_i \phi(\boldsymbol{x_i}) = 0$$

- Finding the eigenvectors is equivalent to finding the coefficients α_i
- KPCA algorithm collects the top k eigenvectors of the centered kernel matrix based on their corresponding eigenvalues, which are ranked by decreasing magnitude.

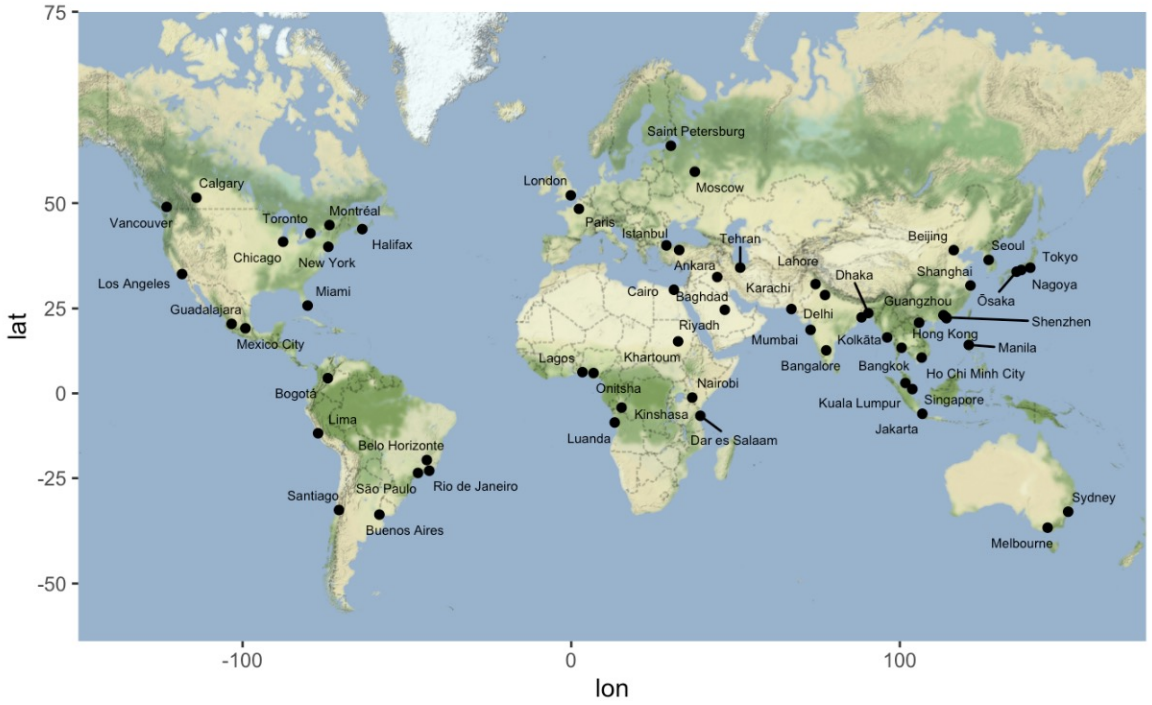
Multidimensional Scaling (MDS)

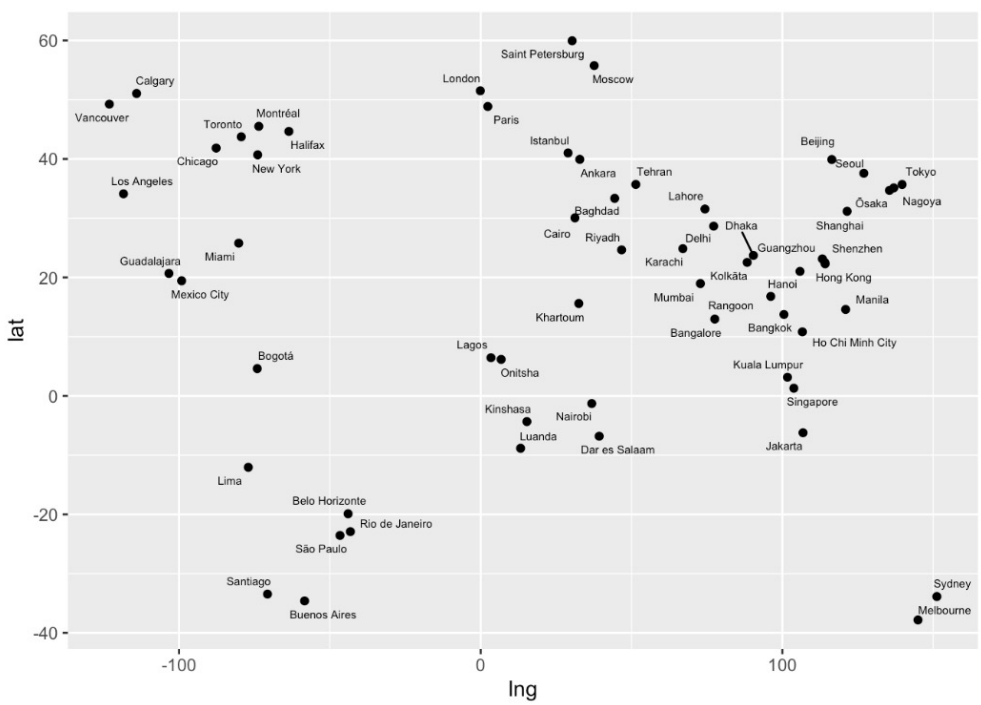
- MDS
  - Means of visualizing the level of similarity of individual cases of a dataset
- Approach
  - Takes the pairwise distances between the entities and finds best-fit representations of the points in all lower dimensional spaces.
  - Input is a similarity/distance matrix, providing measurements between all pairs of data.
  - The output is typically a two-dimensional scatterplot, where each of the objects is represented as a point.
- Applications
  - Perceptual Mapping
  - Geo-statistics



https://www.displayr.com/what-is-multidimensional-scaling-mds/

MDS: Application





Perceptual Mapping

MDS: Algorithm

- Given a distance matrix D $\in \mathbb{R}^{n \times n}$ whose entries are the pairwise distances between the data points, the MDS attempts to find a d−dimensional representation of the data $(d < D)$ such that the pairwise distances are preserved.

$$\min \sum_i \sum_j (D_{ij} - \widehat{D}_{ij})^2$$

- If the distances are the Euclidean distances:

$$D_{ij} = \|x_i - xj\|^2$$
$$\widehat{D}_{ij} = \|\hat{x}_i - \hat{x}_j\|^2$$

- MDS on Euclidean distances is equivalent to PCA

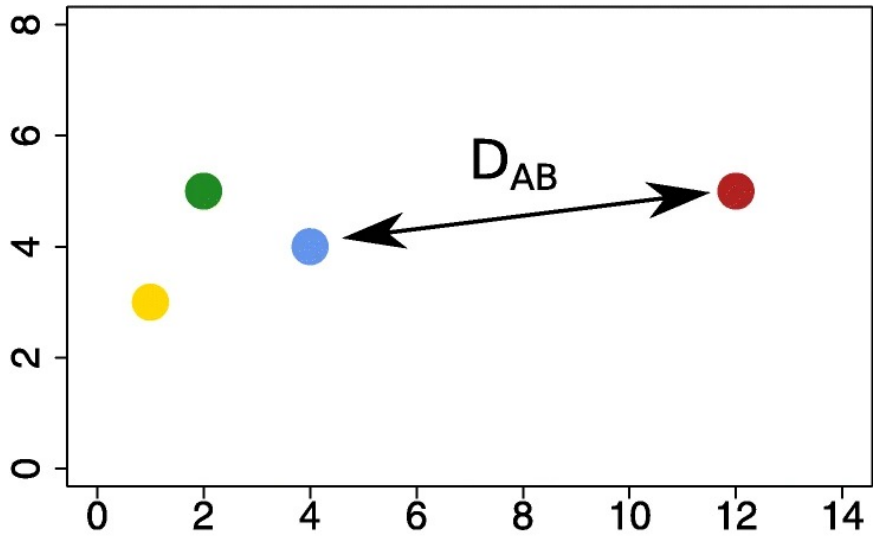
MDS: Algorithm

- The distances between data points in the lower dimensions are not preserved exactly. The distortion in distances between the lower dimension and higher order space is called stress, which is minimized by the MDS implementation.

A

	A	B	C	D
A	0	d_{AB}	d_{AC}	d_{AD}
B	d_{AB}	0	d_{BC}	d_{BD}
C	d_{AC}	d_{BC}	0	d_{CD}
D	d_{AD}	d_{BD}	d_{CD}	0

B



C

$$Stress = S = \sum (D_{ij} - d_{ij})^2$$

https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-019-2780-y/figures/1

t-SNE (t-Stochastic Neighbor Embedding)

- t-SNE
  - Reduces dimensionality while preserving local structure of the data
- Approach
  - Converts high-dimensional distances between datapoints into conditional probabilities that represent similarities.
  - Student-t distribution is used to compute the similarity between two points in the low-dimensional space, to address crowding problem
  - Trained using gradient descent and tries to minimize entropy between distributions.
- Applications
  - Primarily for visualization (detangling) high-dimensional data with multiple manifolds
- Demo
  - t-SNE

t-SNE: Step 1

Find pairwise similarity between nearby points in a high dimensional space.

- Given high-dimensional data $X = \{x_1, x_2, \dots, x_n\}$ and its low dimensional mapping $Y = \{y_1, y_2, \dots, y_n\}$

- The similarity of datapoint x_j to datapoint x_i is the conditional probability, $p_{j|i}$

$$p_{j|i} = \frac{\exp\left(-\|x_i - x_j\|^2 / 2\sigma_i^2\right)}{\sum_{k \neq i} \exp\left(-\|x_i - x_k\|^2 / 2\sigma_i^2\right)}$$

- Symmetrize the conditional probabilities in high dimension space to get the final similarities in high dimensional space.

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

t-SNE: Step 2

Map each point in high dimensional space to a low dimensional map

- For low-dimensional counterparts y_i and y_j of the high-dimensional datapoints x_j and x_j , it is possible to compute a similar conditional probability, which we denote by $\mathrm{q}_{j|i}$
- We compute the conditional probability $\mathrm{q}_{j|i}$ centered under a t distribution centered at point y_i and then symmetrize the probability.

$$\mathrm{q}_{j|i} = \frac{\left(1 + \left\|y_i - \mathrm{y}_j\right\|^2\right)^{-1}}{\sum_{k \neq i} \quad (1 + \ \|\mathrm{y}_i - \mathrm{y}_k\|^2)^{-1}}$$

t-SNE: Step 3

Find a low-dimensional data representation that minimizes KL Divergence

- A natural measure of the faithfulness with which $q_{j|i}$ models $p_{j|i}$ is the KL divergence
- SNE minimizes the sum of KL divergences over all datapoints using a gradient descent method

$$C = \sum_i KL(P_i \parallel Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

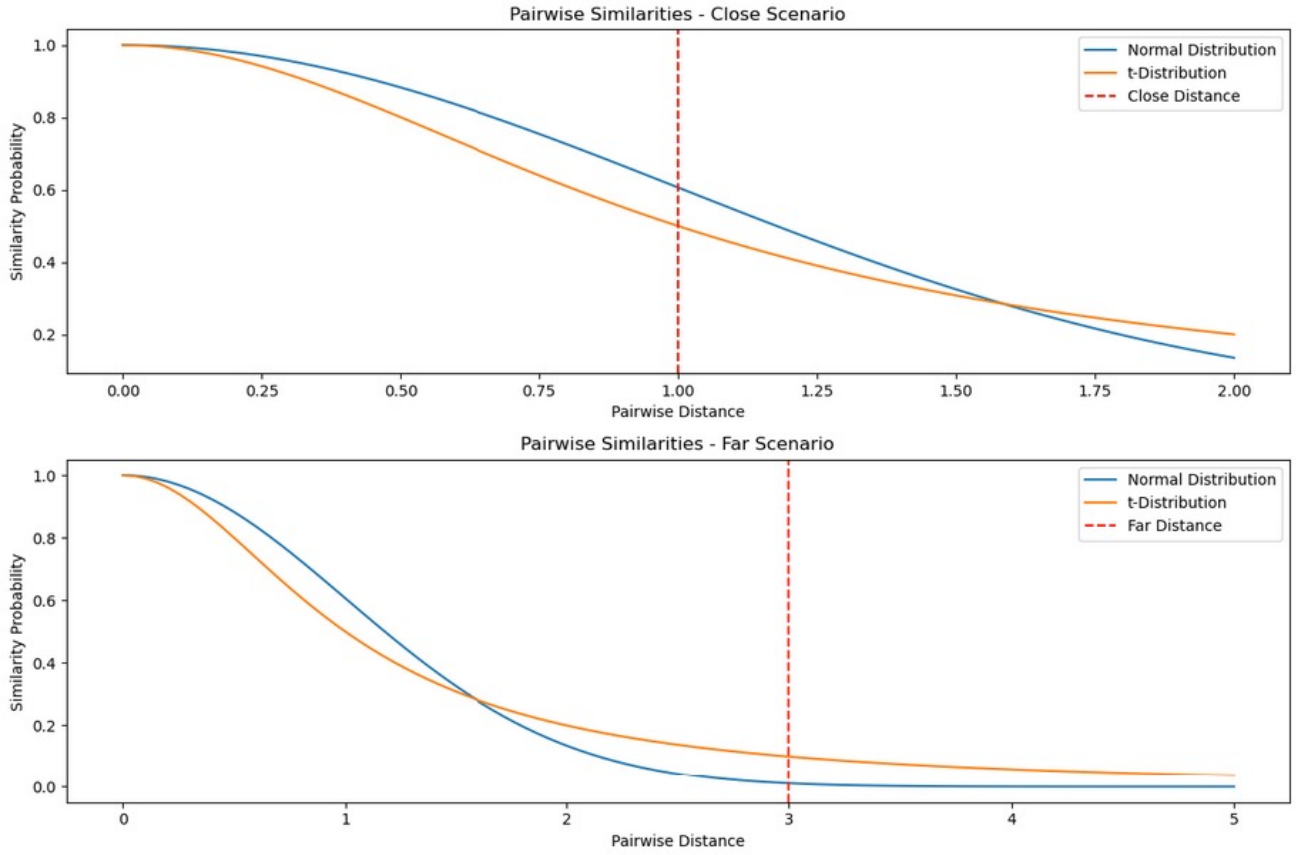
$$\frac{dKL}{dy_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j)(1 + \|y_i - y_j\|^2)^{-1}$$

- SNE cost function focuses on retaining the local structure of the data in the map

t-SNE: Step 4

Use Student-t distribution to compute the similarity in the low-dimensional space

- The heavy tails of the Student t-distribution with one degree of freedom are here to overcome the Crowding Problem when embedding into low dimensions.



t-SNE: Perplexity

- The perplexity can be interpreted as a measure of the effective number of neighbors

$$Perp(P_i) = 2^{\mathrm{H}(P_i)}$$

- where $\mathrm{H}(P_i)$ is the Shannon entropy of P_i measured in bits

$$H(P_i) = -\sum_j P_{j|i} \log_2 P_{j|i}$$

- Typical values are between 5 and 50. The default value is 30.
- Larger datasets usually require a larger perplexity. Perplexity can have a value between 5 and 50.

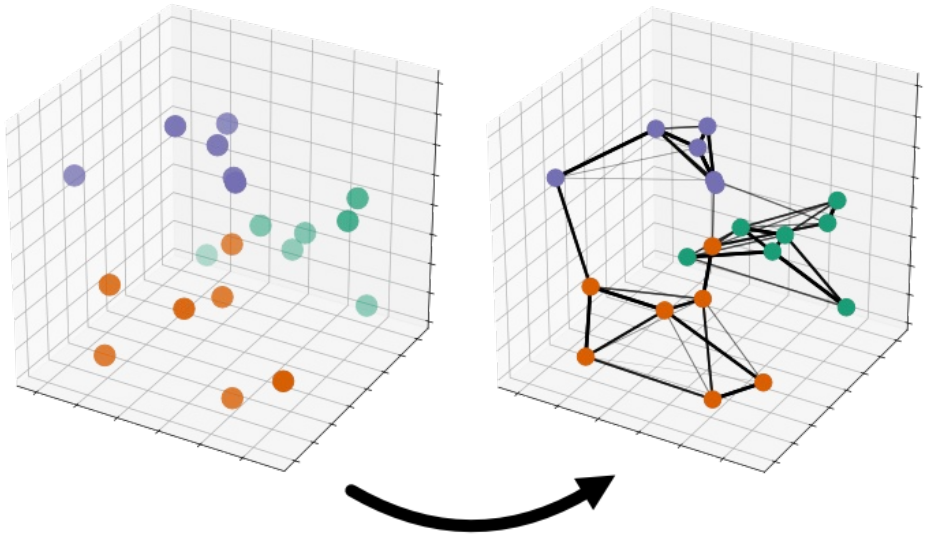
t-SNE: Limitations

- Does not scale well to large datasets
  - Computational and memory complexity is quadratic in the number of datapoints.
  - Large memory consumption (especially when using large perplexity hyperparameter)
  - Can practically only embed into 2 or 3 dimensions
- Cannot work with high-dimensional data directly
  - Autoencoder/PCA are often used for a pre-dimensionality reduction before tSNE
- Does not preserve global data structure
  - Within cluster distances are meaningful while between cluster similarities are not guaranteed

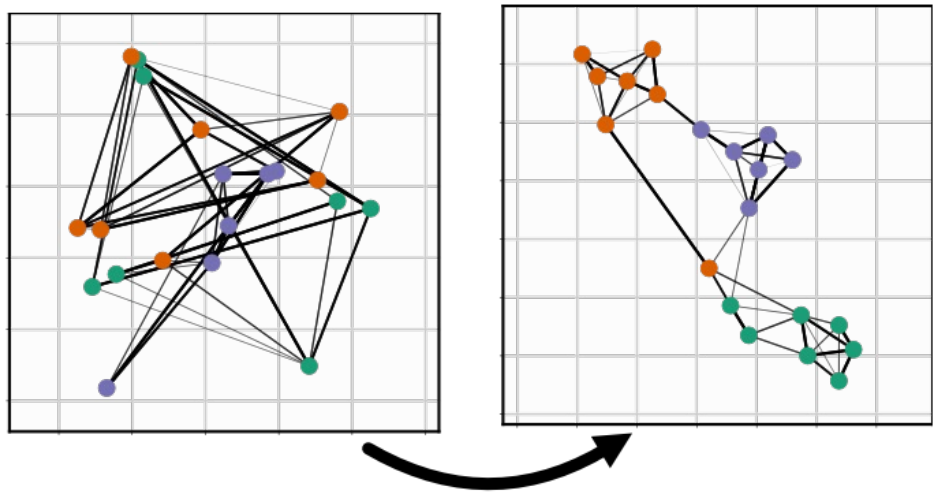
U-Map (Uniform Manifold Approximation and Projection)

- UMAP
  - Manifold learning techniques and ideas from topological data analysis
  - Seeks to learn the manifold structure of data and find a low dimensional embedding that preserves the essential topological structure of that manifold
- Approach
  - Builds a “fuzzy simplicial complex” and calculates similarity scores in clustered points in high dimensional space and preserve the clusters in the low dimensional graph
  - Optimization problem of finding the low dimensional representation with the closest fuzzy topological structure
  - Uses binary cross-entropy (CE) as a cost function instead of the KL-divergence like t-SNE does.
- Demo
  - UMAP – Zoo
  - Understanding U-Map

U-Map: Algorithm



Compute a graphical representation of the dataset



Learn an embedding that preserves the structure of the graph

https://umap-learn.readthedocs.io/en/latest/parametric_umap.html

U-Map: Algorithm

$$\sum_{e \in E} w_h(e) \log\left(\frac{w_h(e)}{w_l(e)}\right) + (1 - w_h(e)) \log\left(\frac{1 - w_h(e)}{1 - w_l(e)}\right)$$

Provides an attractive force between the points spans whenever there is a large weight associated to the high dimensional case

Provides a repulsive force between the ends of e whenever $w_h(e)$ is small

- On balance this process of pull and push, mediated by the weights on edges of the topological representation of the high dimensional data, will let the low dimensional representation settle into a state that relatively accurately represents the overall topology of the source data.

https://umap-learn.readthedocs.io/en/latest/how_umap_works.html

Summary

	PCA	t-SNE	UMAP
Linearity	Linear method.	Non-linear method.	Non-linear method.
Structure	Tries to preserve the global structure of the data.	Tries to preserve the local structure(cluster) of data.	Tries to preserve the local structure(cluster) of data while also preserving global structure as best as possible
Hyperparameters	It does not involve Hyperparameters.	It involves Hyperparameters such as perplexity, learning rate and number of steps.	It involves Hyperparameters such as number of neighbors, components, metric
Outliers	It gets highly affected by outliers.	It can handle outliers.	It can handle outliers and can also be used for outlier detection
Algorithm	Deterministic	Non-deterministic/stochastic	Non-deterministic/stochastic
Approach	Works by rotating the vectors for preserving variance.	Works by minimizing the distance between the point in a gaussian.	Works by using graph layout algorithms to arrange data in low-dimensional space
Preserves	global structures and variance by selecting appropriate number of eigen values.	preserve local structures - distance from high dimensional representation using hyperparameters.	preserve the clusters or topological structure of the high dimensional representation
Scale and Speed	Variations can scale well to large datasets and is relatively fast	Does not scale well to large datasets and can be slow	Handles large datasets and high dimensional data, and is faster than TSNE