Lecture 16. Manifold learning. Similarity measures

COMP90051 Statistical Machine Learning

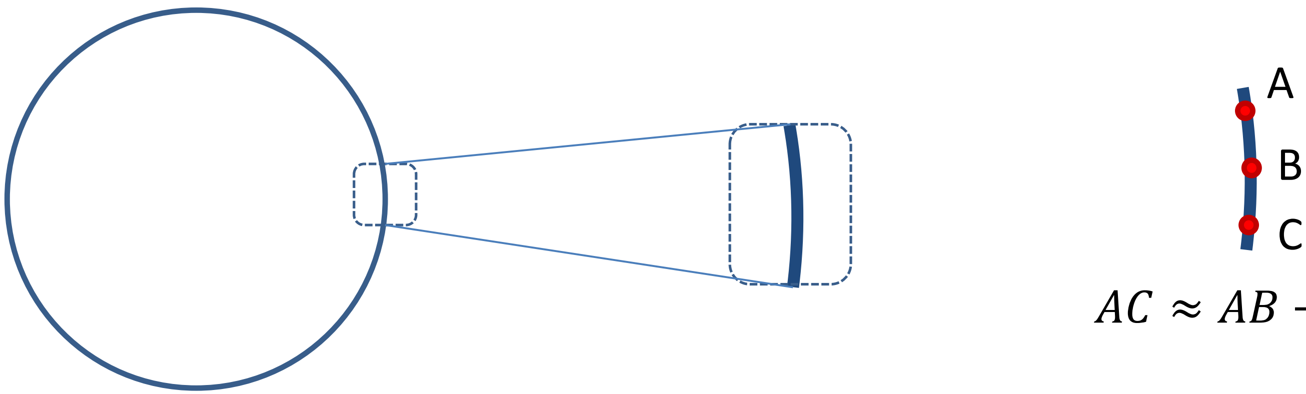
Semester 2, 2015
Lecturer: Andrey Kan

Swiss roll image due to Evan-Amos, Wikimedia Commons, CC0

Manifold Learning

An approach to learn from data using non-linear dimensionality reduction

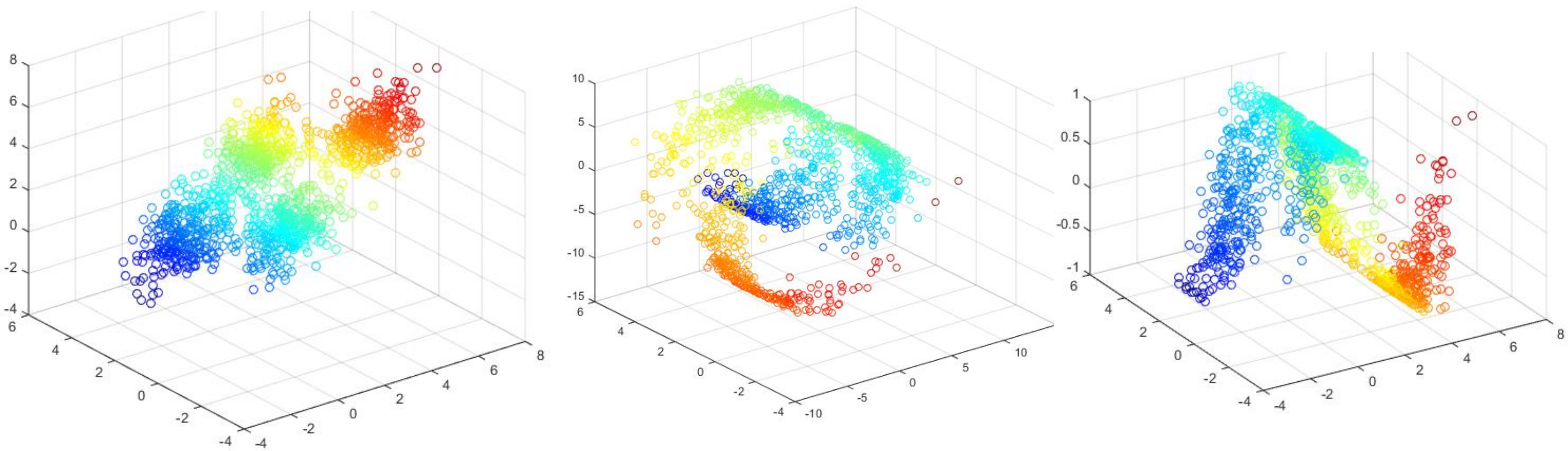
Manifold learning



$$AC \approx AB + BC$$

- **Manifold** is a space that resembles Euclidean space near each point
- Example: circumference
  * consider a tiny bit of a circumference → can treat as line

Manifold examples



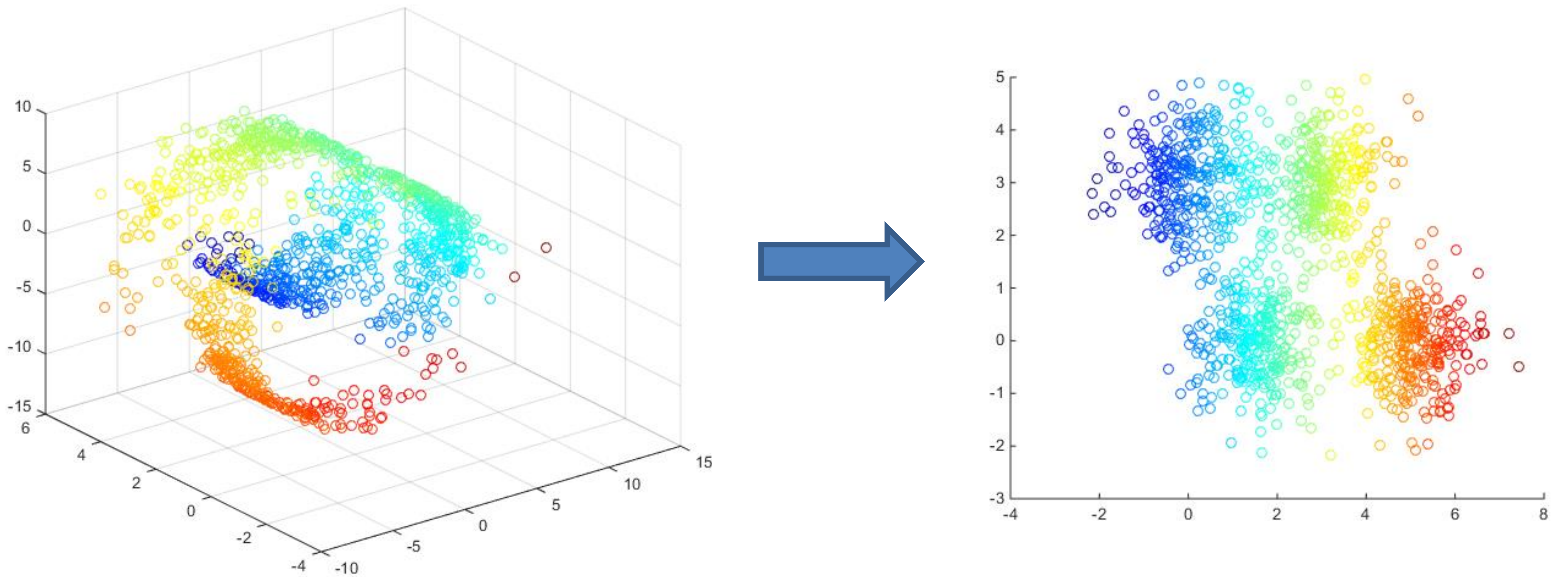
- **Manifold** is a space that resembles Euclidean space near each point

More manifold examples

- **Manifold** is a space that resembles Euclidean space near each point

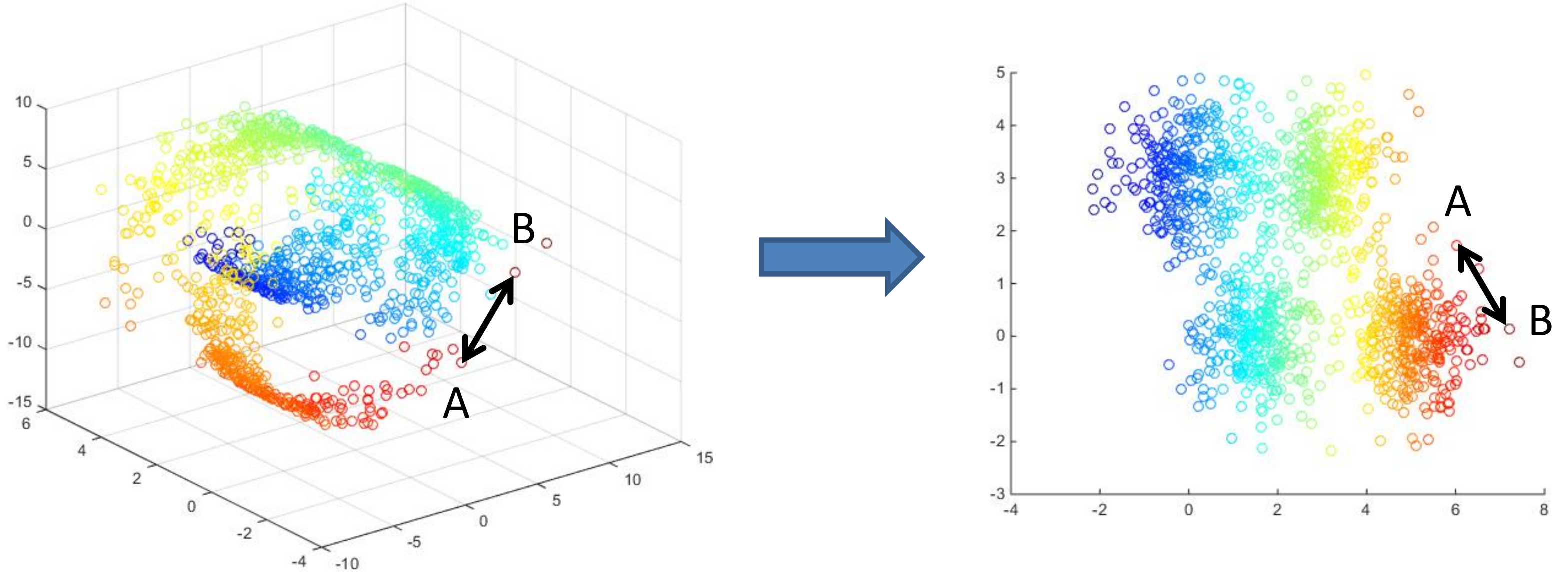
Picnic blanket image due to Theo Wright, Flickr, CC2
Swiss roll image due to Evan-Amos, Wikimedia Commons, CC0

Key assumption: it's simpler than it looks!



- Assumption: high dimensional data actually lies on a lower-dimensional manifold
- Data does not have to perfectly lie on the manifold, some noise can be expected

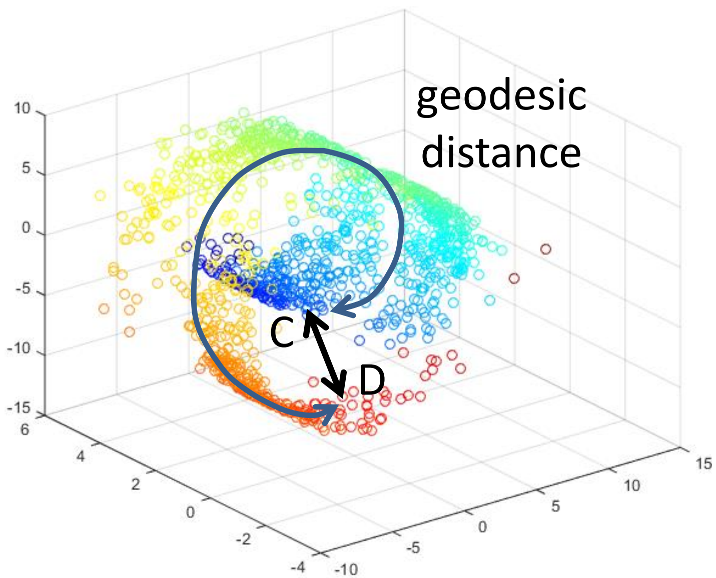
General idea: dimensionality reduction

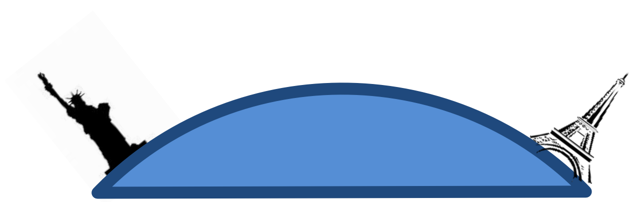


- Find a lower dimensional representation of data that preserves distances between points (MDS)
- Do visualization, clustering, etc. on lower dimensional representation

Problems?

"Global distances" VS geodesic distances



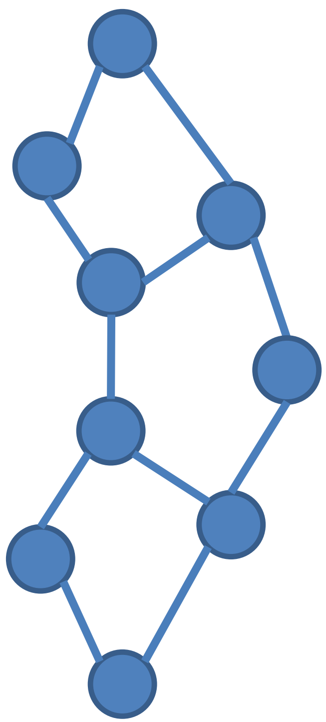


- "Global distances" cause a problem: we may not want to preserve them
- We are interested in preserving distances along the manifold (*geodesic distances*)

Images: ClkerFreeVectorImages and Kaz @pixabay.com (CC0)

Isomap Algorithm

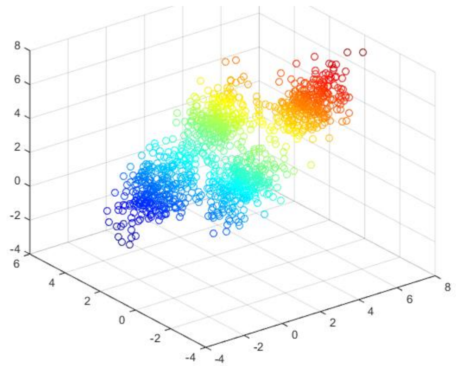
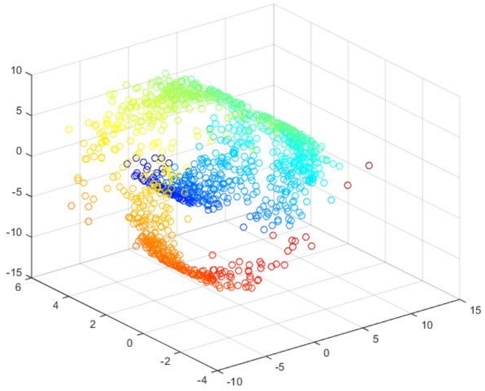
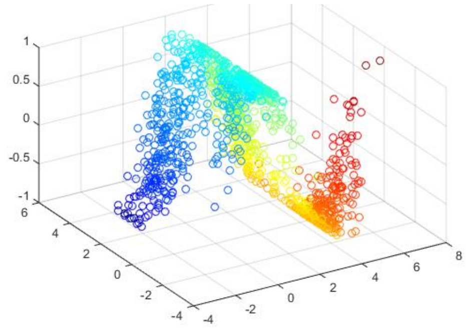
ε specifies how "local" or how smooth the manifold is



- Input: $\boldsymbol{x}_1, \dots, \boldsymbol{x}_N \in \mathbb{R}^p$
  * Global (dis-)similarity measure $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$
- Construct the graph
  * Nodes – original data points
  * Edge ij – if two nodes are close to each other: $d(\boldsymbol{x}_i, \boldsymbol{x}_j) < \varepsilon$
  * (alternative: edges connect k nearest neighbors)
  * Edge length = $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$
- Compute shortest paths
  * E.g., using Floyd-Warshall algorithm, $O(n^3)$
  * New similarity measure $d_G(i, j) =$ length of shortest path between nodes i and j
- Apply MDS using $d_G(i, j)$

Methods summary

- PCA: finds a new basis via a linear transformation, like rotation
- Kernel PCA: first (implicitly) maps data into a new feature space, then applies PCA
- Classical MDS: finds lower dimensional embedding that preserves pairwise distances
- Isomap: finds lower dimensional embedding that preserves geodesic distances
  - This is achieved using the adjacency graph and applying MDS on distances defined on the graph

Keywords/references for further reading

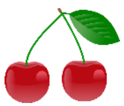
- Isomaps
  - Tenenbaum, Joshua B., Vin De Silva, and John C. Langford. "A global geometric framework for nonlinear dimensionality reduction." Science 290.5500 (2000): 2319-2323.
- Locally linear embedding
  - Roweis, Sam T., and Lawrence K. Saul. "Nonlinear dimensionality reduction by locally linear embedding." *Science* 290.5500 (2000): 2323-2326.
- Laplacian eigenmaps
  - Belkin, Mikhail, and Partha Niyogi. "Laplacian eigenmaps for dimensionality reduction and data representation." Neural computation 15.6 (2003): 1373-1396.

Checkpoint

- Which of the following statements is true?

 MDS requires dissimilarity/similarity matrix as input

 MDS preserves all the information from the original dataset

 All principal components (after performing PCA) are needed in order to visualize high-dimensional data

art: OpenClipartVectors at pixabay.com (CC0)

Similarity Measures

A view of data that generalizes to arbitrary objects

Data points as generic objects

- Commonly used view of data $\boldsymbol{x}_1, \dots, \boldsymbol{x}_N \in \mathbb{R}^p$
- Sometimes we don't have point "values"
  - Wine tasting experiment collects preferences
  - "Data points" are sequences (e.g., GAACAAAACTCATC ...)
  - "Data points" are distributions
  - "Data points" are images
  - "Data points" are graphs

Pairwise similarities/dissimilarities

- Many learning algorithms do require original values for data points, <u>only pairwise dissimilarities</u>
  - Kernels can be viewed as pairwise similarities
  - Dimensionality reduction algorithms
- Proximity matrix for n data points is a square $n \times n$ matrix of pairwise similarities
- Sometimes it is more convenient to talk about dissimilarities, and the corresponding dissimilarity matrix

Dissimilarity functions

- Dissimilarity function is usually defined so that
  - $d(\boldsymbol{u}, \boldsymbol{v}) \geq 0$
  - $d(\boldsymbol{u}, \boldsymbol{u}) = 0$
  - $d(\boldsymbol{u}, \boldsymbol{v}) = d(\boldsymbol{v}, \boldsymbol{u})$
- Some functions also satisfy
  - $d(\boldsymbol{x}, \boldsymbol{v}) \leq d(\boldsymbol{x}, \boldsymbol{u}) + d(\boldsymbol{u}, \boldsymbol{v})$
  - These are called *metrics* or *distances*

Popular choices

- If each data point can be represented $\boldsymbol{x} = [x_1, \dots, x_n]$
- Euclidean distance is commonly used as a distance measure
  - $d(\boldsymbol{u}, \boldsymbol{v}) = \|\boldsymbol{u} - \boldsymbol{v}\|$
- Centered inner product is often used as a similarity measure
  - $s(\boldsymbol{u}, \boldsymbol{v}) = \langle \boldsymbol{u} - \overline{\boldsymbol{x}}, \boldsymbol{v} - \overline{\boldsymbol{x}} \rangle$

Same data, different perspectives

- Two objects can be compared from different perspectives
  - E.g., two employees can be considered similar if they are from same/neighboring countries and have similar age OR they can be compared based on the number of years they worked and work performance
- Different measures → different clustering → study of different aspects of data

Designing similarity/dissimilarity measures

- Choose which features to use
- Decide how to combine them
- Examples:
  - $d(\boldsymbol{u},\boldsymbol{v}) = \sum_{i=1}^{d} d_i(u_i, v_i)$
  - $d_{abs}(\boldsymbol{u},\boldsymbol{v}) = \sum_{i=1}^{d} |u_i - v_i|$
  - $d_{sq}(\boldsymbol{u},\boldsymbol{v}) = \sum_{i=1}^{d} (u_i - v_i)^2 = \|\boldsymbol{u}-\boldsymbol{v}\|^2$
  - $d_w(\boldsymbol{u},\boldsymbol{v}) = \sum_{i=1}^{d} w_i (u_i - v_i)^2$

Be careful with scaling

- In supervised learning setup weights for features are learned from training data
- In unsupervised learning, weights need to be set manually

- Consider grouping loan applicants
- Each applicant is characterized with a salary and age.
  - For example, $[90K, 35]$, $[92K, 35]$, $[90K, 45]$
- With equal weights, the first feature will dominate, and the second will have almost no impact

Ordinal and categorical variables

- Ordinal and categorical features requires special treatment
- Example for an ordinal variable A ∈ {*strongly agree, agree, neutral, disagree, strongly disagree*}
  - Convert to 0 to 4
  - Normalise: $0, \frac{1}{4}, \frac{2}{4}, \frac{3}{4}, 1$
  - Compare two categorical variables as numbers
- Example for categorical variable B ∈ *{female, male}*
  - $d(B_1, B_2) = 1$ if $B_1 = B_2$, and $d(B_1, B_2) = 0$ otherwise
- Ultimately, the design is application specific

Levenshtein distance

- Dissimilarity between two sequences: minimum number of single-character edits (insertions, deletions or substitutions) required to turn one string into another
- Example: $d(\text{"}carrot\text{"}, \text{"}parrot\text{"}) = 1$
  - “carrot” → “parrot”
- Example: $d(\text{"}hello\text{"}, \text{"}help\text{"}) = 2$
  - “hello” → “helpo” → “help”

Graph edit distance

- Dissimilarity between two graphs: minimum number of edits (edge/vertex insertions or deletions) required to turn one graph into another
- Example: $d(G1, G2) = 2$

Cross-correlation

- Can be used for images or strings
- Example: grayscale images $f(x, y)$ and $t(x, y)$

$$r = \frac{1}{n} \sum_{x,y} \frac{(f(x,y) - \bar{f})(t(x,y) - \bar{t})}{\sigma_f \sigma_t},$$

  - where $\bar{f}$ and $\bar{t}$ are mean values;
  - n is the number of pixels; and
  - σ_f, and σ_t are sample standard deviations.

Bhattacharyya distance

- For discrete probability distributions

$$D_B(p, q) = -\ln \sum_{x \in X} \sqrt{p(x)q(x)}$$

- For continuous probability distributions

$$D_B(p, q) = -\ln \int \sqrt{p(x)q(x)}\, dx$$

Normalised compression distance

- Applicable to any type of objects

$$NCD(x, y) = \frac{zip(xy) - \min\{zip(x), zip(y)\}}{\max\{zip(x), zip(y)\}}$$

Summary

- The key assumption of manifold learning is that the data is “simpler than it looks”
  - Assumption: high dimensional data lies on a lower dimensional manifold
- There are different ways to “flatten the manifold”
  - MDS and Isomap methods attempt to find a lower dimensional embedding that preserves pairwise distances
  - Isomap is a more adequate method because it aims to preserve geodesic distances
- Kernel methods and manifold learning methods can work using dissimilarity matrix as input, original data values are not required
- There has been many ways to quantify dissimilarity. Different measures can be applied to objects of different types.