CSE8803/CX4803 Machine Learning in Computational Biology

Lecture 8: Learning from high-dimensional data:
MDS, tSNE, UMAP

Xiuwei Zhang

School of Computational Science and Engineering

Logistics

- Submit your bid for presentation dates by Feb. 7, 11:59pm https://forms.gle/YJYDxDRavjPNoQRc7

- Comments on Canvas assignment submissions
  - Are typically not noticeable
  - For post-submission comments please email instructors/TAs
  - Ed is the primary place to ask questions

Dimensionality reduction methods

Principal Component Analysis (PCA)

Autoencoder

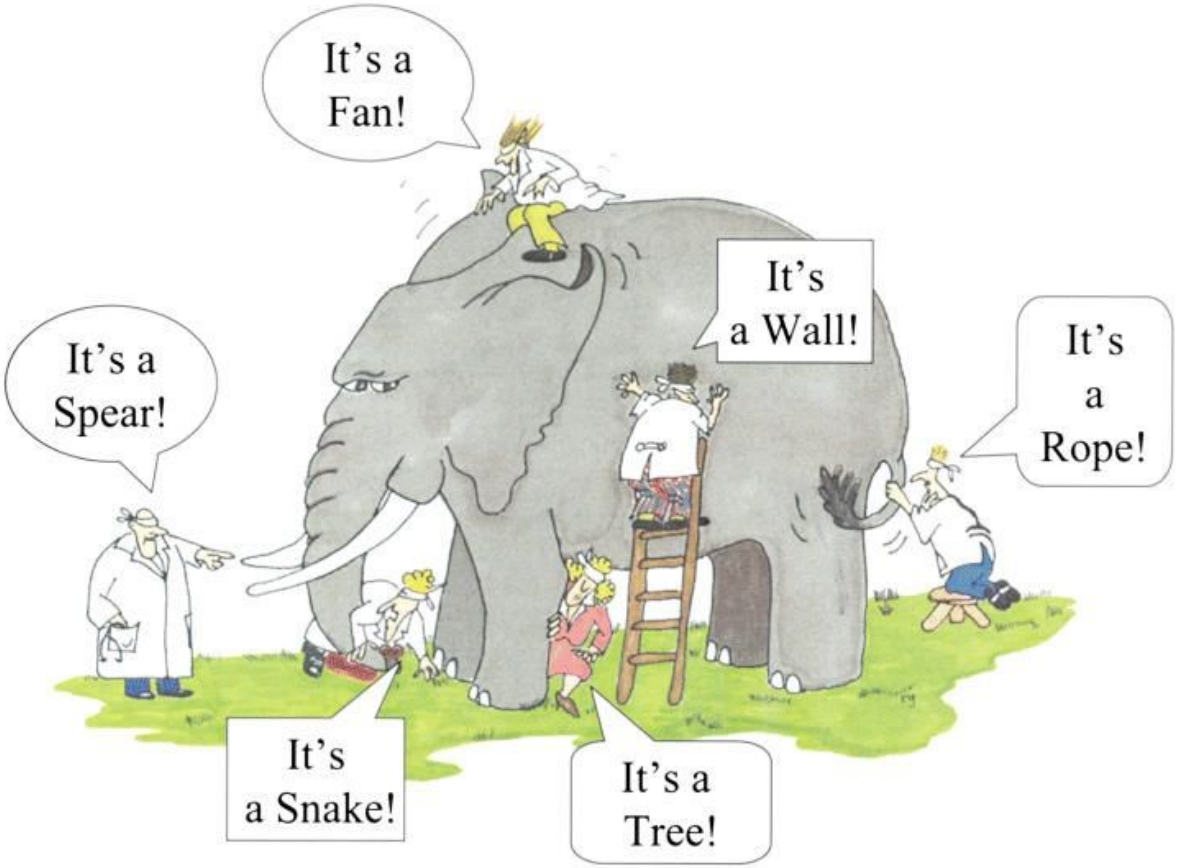
Variational autoencoder (VAE)

Multidimensional scaling (MDS)

t-SNE (t-Distributed Stochastic Neighbor Embedding)

UMAP (Uniform Manifold Approximation and Projection)

Different
Objectives give us
different results



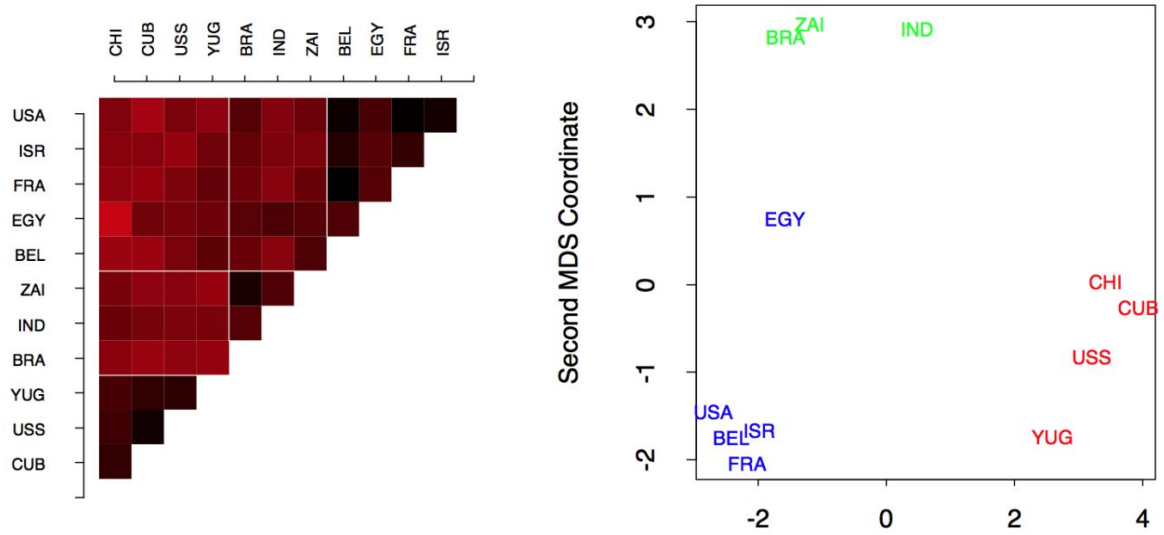
MDS

Represent high-dimensional point cloud in few (usually 2) dimensions keeping distances between points similar

- Classical multidimensional scaling
- Metric multidimensional scaling (mMDS)
- Non-metric multidimensional scaling (nMDS)
- Generalized multidimensional scaling (GMDS)

MDS

Given pairwise dissimilarities, reconstruct a map (in *p*-dimensional space) that preserves distances.



MDS

High-dimensional input: distance matrix between *n* samples $D \in \mathbb{R}^{n \times n} \quad D = [d_{ij}]$

Low-dimensional embedding: $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^p$ such that $d_{ij} \approx \|\mathbf{x}_i - \mathbf{x}_j\|_2$.

- If *D* is Euclidean distance, there exists a *p*, where we can find exact match for $d_{ij} \approx \|\mathbf{x}_i - \mathbf{x}_j\|_2$
- If D is not Euclidean distance (eg. Radian distance function on a circle), we often can not find exact match. Nevertheless, MDS seeks to find an optimal configuration $\boldsymbol{x}_i$ that gives $d_{ij} \approx \|\mathbf{x}_i - \mathbf{x}_j\|_2$ as close as possible.

Classical MDS

Suppose D is Euclidean distance.

Objective: find $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^p$ such that $\left\| \mathbf{x}_i - \mathbf{x}_j \right\|_2 = d_{ij}$.

Question: is this solution unique?

Answer: no.

Center the points:

$$\sum_{i=1}^{n} x_{ik} = 0, \quad \text{for all } k$$

Classical MDS

Steps:

1. Calculate $A = [-\frac{1}{2}d_{ij}^2]$.
2. Calculate $B = HAH$, where $H = I - \frac{1}{n}\mathrm{e}\mathrm{e}^T$ $(\mathrm{e} = (1, ..., 1)^T)$
3. Eigenvalue decomposition on B

$$B = V\Lambda V^T = \begin{pmatrix}\mathbf{v}_1 \cdots \mathbf{v}_p\end{pmatrix} \cdots \mathbf{v}_n) \begin{pmatrix} \lambda_1 & & & \\ & \ddots & & 0 \\ & & \lambda_p & \\ 0 & & & \ddots \\ & & & & \lambda_n \end{pmatrix} \begin{pmatrix} \mathbf{v}_1^T \\ \vdots \\ \mathbf{v}_p^T \\ \vdots \\ \mathbf{v}_n^T \end{pmatrix}$$

4. $X = \Lambda^{\frac{1}{2}} V^T$, $X_p = \Lambda_p^{\frac{1}{2}} V_p^T$

Classical MDS

Proof.

Our goal is to find the low-dimensional representation of the *n* samples, $X_{(qxn)}$, q>p

Objective: $\left\| \mathbf{x}_i - \mathbf{x}_j \right\|_2 = d_{ij}$

$$\left\| \mathbf{x}_i - \mathbf{x}_j \right\|_2^2 = d_{ij}^2$$

$$\mathbf{x}_i^T \mathbf{x}_i + \mathbf{x}_j^T \mathbf{x}_j - 2\mathbf{x}_i^T \mathbf{x}_j = b_{ii} + b_{jj} - 2b_{ij}$$

Introduce matrix $B=X^TX$:

b_{ii}: inner product of x_i and x_i

b_{ij}: inner product of x_i and x_j

Classical MDS

Proof.

Our goal is to find the low-dimensional representation of the *n* samples, $X_{(qxn)}$

Objective: $\left\|\mathbf{x}_i - \mathbf{x}_j\right\|_2 = d_{ij}$

$$\left\|\mathbf{x}_i - \mathbf{x}_j\right\|_2^2 = d_{ij}^2$$

$$\sum_{i=1}^{n} d_{ij}^2 = \sum_{i=1}^{n} b_{ii} + n \cdot b_{jj}$$

$$\sum_{j=1}^{n} d_{ij}^2 = \sum_{j=1}^{n} b_{jj} + n \cdot b_{ii}$$

Find B, such that:

$$b_{ii} + b_{jj} - 2b_{ij} = d_{ij}^2$$

$$\sum_{i=1}^{n} x_{ik} = 0, \quad \text{for all } k \quad \Rightarrow \quad \sum_{i=1}^{n} b_{ij} = \sum_{i=1}^{n}\sum_{k=1}^{q} x_{ik}x_{jk} = \sum_{k=1}^{q} x_{jk} \sum_{i=1}^{n} x_{ik} = 0$$

Classical MDS

Proof.

Our goal is to find the low-dimensional representation of the *n* samples, $X_{(qxn)}$

Objective: $\left\|\mathbf{x}_i - \mathbf{x}_j\right\|_2 = d_{ij}$

$$\left\|\mathbf{x}_i - \mathbf{x}_j\right\|_2^2 = d_{ij}^2$$

Find B, such that:

$$b_{ii} + b_{jj} - 2b_{ij} = d_{ij}^2$$

$$\sum_{i=1}^{n} d_{ij}^2 = \sum_{i=1}^{n} b_{ii} + n \cdot b_{jj}$$

$$\sum_{j=1}^{n} d_{ij}^2 = \sum_{j=1}^{n} b_{jj} + n \cdot b_{ii}$$

$T = \text{trace}(B) = \sum_{i=1}^{n} b_{ii} \Rightarrow \sum_{i=1}^{n} d_{ij}^2 = T + nb_{jj}, \sum_{j=1}^{n} d_{ij}^2 = T + nb_{ii} \Rightarrow \sum_{j=1}^{n}\sum_{i=1}^{n} d_{ij}^2 = 2nT$

Classical MDS

Proof.

Our goal is to find the low-dimensional representation of the *n* samples, $X_{(qxn)}$

Objective: $\left\|\mathbf{x}_i - \mathbf{x}_j\right\|_2 = d_{ij}$

$$\left\|\mathbf{x}_i - \mathbf{x}_j\right\|_2^2 = d_{ij}^2$$

n 1 +n 2 - 3

Find B, such that:

$$b_{ii} + b_{jj} - 2b_{ij} = d_{ij}^2$$

$$b_{ii} + b_{jj} = \frac{1}{n}\sum_{i=1}^{n} d_{ij}^2 + \frac{1}{n}\sum_{j=1}^{n} d_{ij}^2 - \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{2} d_{ij}^2$$

$$\sum_{i=1}^{n} d_{ij}^2 = T + nb_{jj}, \quad \sum_{j=1}^{n} d_{ij}^2 = T + nb_{ii} \qquad \sum_{j=1}^{n}\sum_{i=1}^{n} d_{ij}^2 = 2nT$$

1 2 3

Classical MDS

Proof.

Our goal is to find the low-dimensional representation of the *n* samples, $X_{(qxn)}$

Objective: $\left\|\mathbf{x}_i - \mathbf{x}_j\right\|_2 = d_{ij}$

$$\left\|\mathbf{x}_i - \mathbf{x}_j\right\|_2^2 = d_{ij}^2$$

Find B, such that:

$$b_{ii} + b_{jj} - 2b_{ij} = d_{ij}^2$$

$$b_{ii} + b_{jj} = \frac{1}{n}\sum_{i=1}^{n} d_{ij}^2 + \frac{1}{n}\sum_{j=1}^{n} d_{ij}^2 - \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{2} d_{ij}^2$$

$$b_{ij} = -\frac{1}{2}\left(d_{ij}^2 - \frac{1}{n}\sum_{i=1}^{n} d_{ij}^2 - \frac{1}{n}\sum_{j=1}^{n} d_{ij}^2 + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{2} d_{ij}^2\right) \Rightarrow \begin{cases} A = [-\frac{1}{2}d_{ij}^2] \\ H = I - \frac{1}{n}\mathrm{e}\mathrm{e}^T \ (\mathrm{e} = (1, ..., 1)^T) \\ B = HAH \end{cases}$$

Classical MDS

Steps:

1. Calculate $A = [-\frac{1}{2}d_{ij}^2]$.
2. Calculate $B = HAH$, where $H = I - \frac{1}{n}\mathrm{e}\mathrm{e}^T$ $(\mathrm{e} = (1, ..., 1)^T)$ $B=X^TX$
3. Eigenvalue decomposition on B

$$B = V\Lambda V^T = (\mathbf{v}_1 \cdots \mathbf{v}_p \cdots \mathbf{v}_n) \begin{pmatrix} \lambda_1 & & & & \\ & \ddots & & \mathbf{0} & \\ & & \lambda_p & & \\ & \mathbf{0} & & \ddots & \\ & & & & \lambda_n \end{pmatrix} \begin{pmatrix} \mathbf{v}_1^T \\ \vdots \\ \mathbf{v}_p^T \\ \vdots \\ \mathbf{v}_n^T \end{pmatrix}$$

4. $X = \Lambda^{\frac{1}{2}} V^T$, $X_p = \Lambda_p^{\frac{1}{2}} V_p^T$

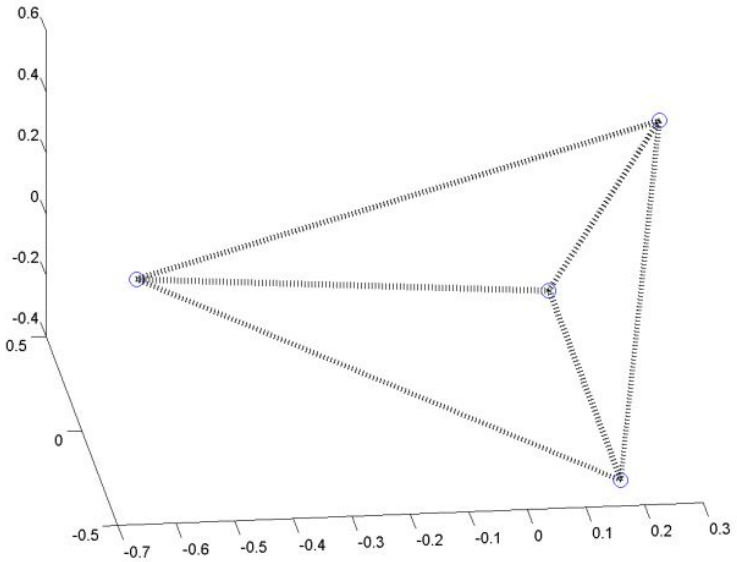
Classical MDS

Example: tetrahedron

$$D = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

eigenvalues (.5, .5, .5, 0)

Set p=3:



Dimensionality reduction methods

Principal Component Analysis (PCA)

Autoencoder

Variational autoencoder (VAE)

Multidimensional scaling (MDS)

t-SNE (t-Distributed Stochastic Neighbor Embedding)

UMAP (Uniform Manifold Approximation and Projection)

t-SNE (t-Distributed Stochastic Neighbor Embedding)

- PCA, MDS try to find a global structure
- t-SNE tries to preserve *local structure*
  - Low dimensional neighborhood should be the same as original neighborhood.
  - Used mainly for visualization

t-SNE (t-Distributed Stochastic Neighbor Embedding)

- Represents the distance relationships between samples in high dimensional space by probability distribution (neighborhood)
- Find low dimensional points such that their neighborhood distribution is similar.
- Compare the two distributions: KL divergence

KL (Kullback–Leibler) divergence: $$D_{\mathrm{KL}}(P \parallel Q) = \int_{-\infty}^{\infty} p(x) \log\left(\frac{p(x)}{q(x)}\right) dx$$

t-SNE (t-Distributed Stochastic Neighbor Embedding)

- Consider the neighborhood around an input data point $\mathbf{x}_i \in \mathbb{R}^d$
- Imagine that we have a Gaussian distribution centered around $\mathbf{x}_i$
- Then the probability that $\mathbf{x}_i$ chooses some other datapoint $\mathbf{x}_j$ as its neighbor is in proportion with the density under this Gaussian
- A point closer to $\mathbf{x}_i$ will be more likely than one further away

t-SNE (t-Distributed Stochastic Neighbor Embedding)

- *High dimensional space*:
  The probability that point x_i chooses x_j as its neighbor:

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

RBF (radial basis function) kernel
Gaussian kernel

- *Low dimensional space*:
  Find embedding $\mathbf{y}^{(1)}, .., \mathbf{y}^{(N)} \in \mathbb{R}^d$
  And define the distribution in the *d*-dim space:

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

?

- Such that the KL divergence between P and Q is minimized

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

t-SNE (t-Distributed Stochastic Neighbor Embedding)

- *High dimensional space*:
  The probability that point x_i chooses x_j as its neighbor:

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

RBF (radial basis function) kernel
Gaussian kernel

- *Low dimensional space*:
  Find embedding $\mathbf{y}^{(1)}, .., \mathbf{y}^{(N)} \in \mathbb{R}^d$
  And define the distribution in the *d*-dim space:

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

Crowding problem.
Use t-distribution instead of Gaussian-like

- Such that the KL divergence between P and Q is minimized

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

t-SNE (t-Distributed Stochastic Neighbor Embedding)

- *High dimensional space*:
  The probability that point x_i chooses x_j as its neighbor:
- *Low dimensional space*:
  Find embedding $\mathbf{y}^{(1)}, .., \mathbf{y}^{(N)} \in \mathbb{R}^d$
  And define the distribution in the *d*-dim space:
- Such that the KL divergence between P and Q is minimized

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

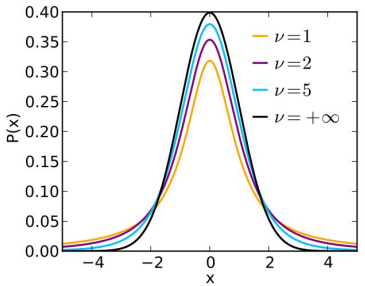
RBF (radial basis function) kernel
Gaussian kernel

$$Q_{ij} = \frac{(1 + ||\mathbf{y}_i - \mathbf{y}_j||^2)^{-1}}{\sum_k \sum_{l \neq k} (1 + ||\mathbf{y}_k - \mathbf{y}_l||^2)^{-1}}$$

Crowding problem.
Use t-distribution instead of Gaussian-like

$$KL(Q||P) = \sum_{ij} Q_{ij} \log \left($$

$$p(x) \propto \left(1 + \frac{x^2}{v}\right)^{-(v+1)/2}$$



t-SNE (t-Distributed Stochastic Neighbor Embedding)

- Parameters
  - Perplexity

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

$$perp(P_{j|i}) = 2^{H(P_{j|i})}$$

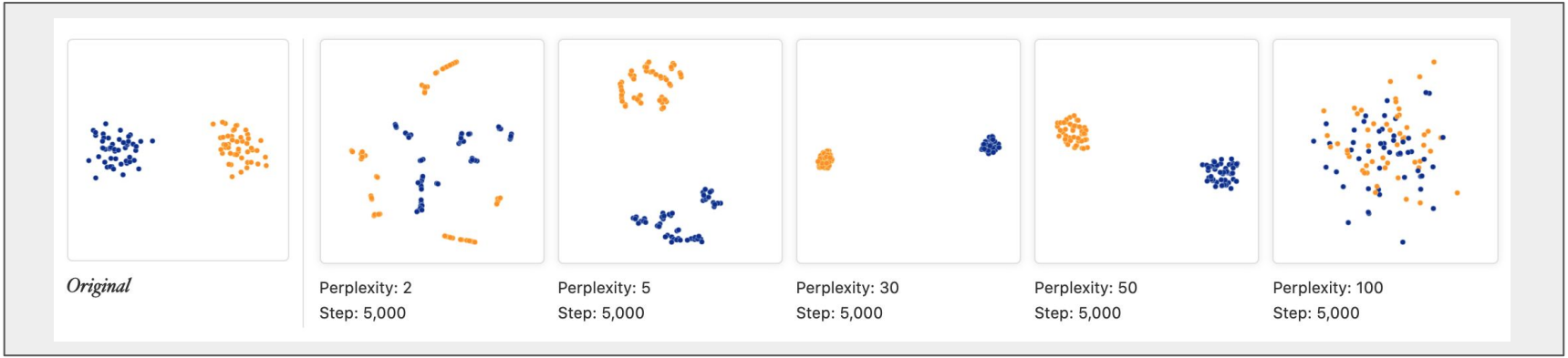
$$H(P) = -\sum_i P_i \log(P_i)$$

If P is uniform over k elements - perplexity is k

In practice, we give perplexity, and the algorithm adjusts σ
Low perplexity $\rightarrow$ smaller σ $\rightarrow$ considers smaller neighborhood
High perplexity $\rightarrow$ larger σ $\rightarrow$ considers larger neighborhood

  - Balances between local and global structure. Value between 5-50.

t-SNE (t-Distributed Stochastic Neighbor Embedding)



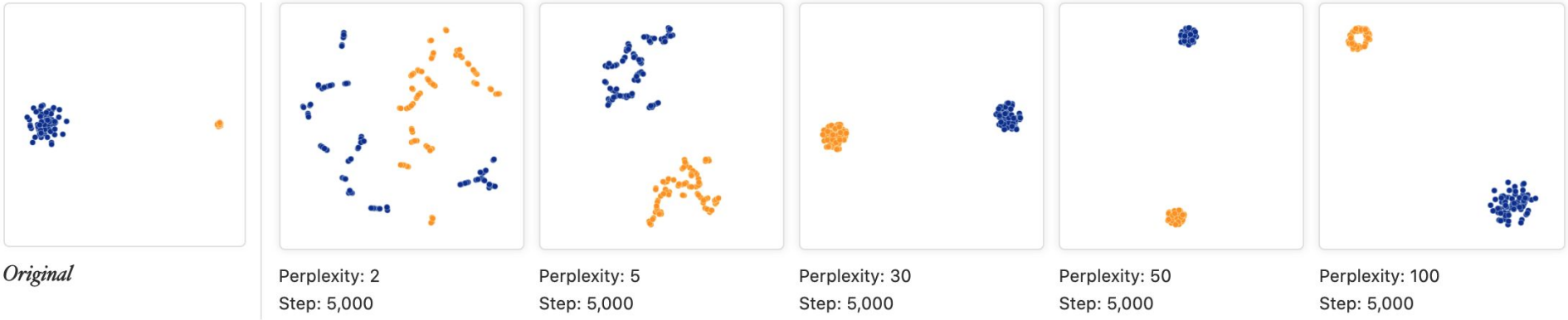
In practice, we give perplexity, and the algorithm adjusts σ
Low perplexity $\rightarrow$ smaller σ $\rightarrow$ considers smaller neighborhood
Low perplexity $\rightarrow$ larger σ $\rightarrow$ considers larger neighborhood

- Balances between local and global structure. Value between 5-50.

Figure credit: https://distill.pub/2016/misread-tsne/

t-SNE (t-Distributed Stochastic Neighbor Embedding)

- Other practical aspects
  - Does cluster size mean heterogeneity in the cluster?

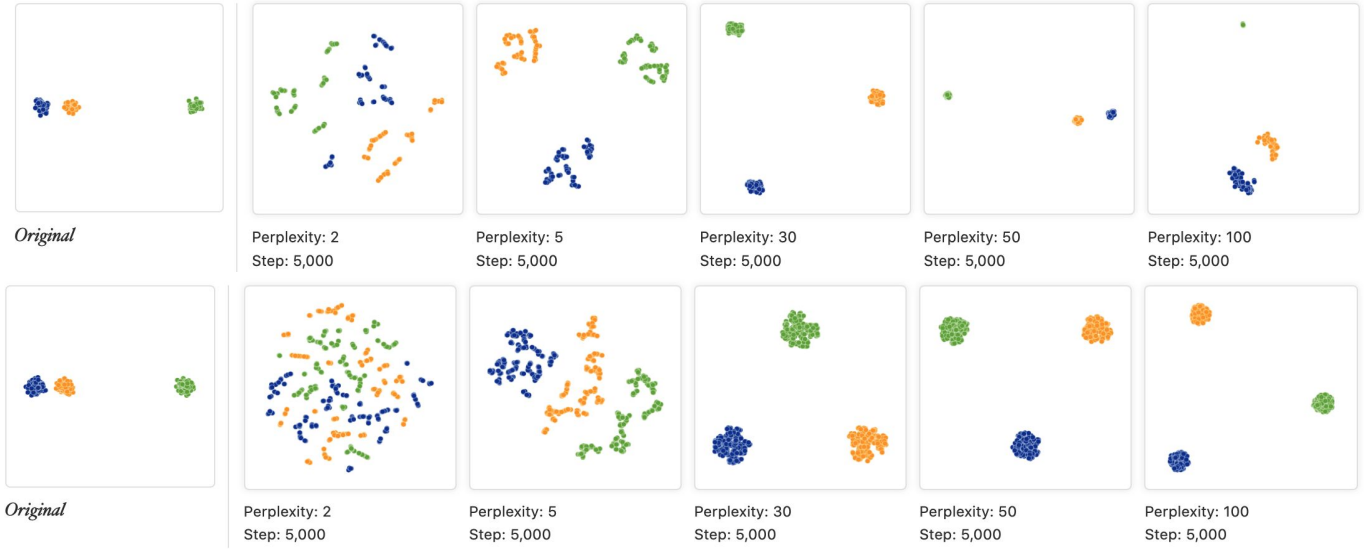


No.

Figure credit: https://distill.pub/2016/misread-tsne/

t-SNE (t-Distributed Stochastic Neighbor Embedding)

- Other practical aspects
  - Does the distance between clusters mean anything?



Not always

Figure credit: https://distill.pub/2016/misread-tsne/

MDS vs t-SNE

MDS

Given a dissimilarity matrix, find low-dimensional embeddings of data points where the distance between points in the low-dimensional space is as close as possible to D.

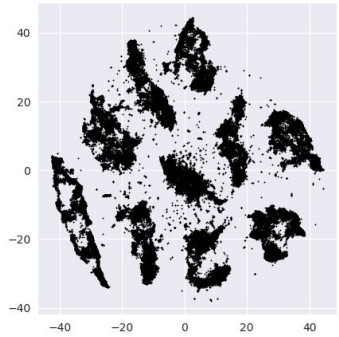
tSNE

Given data in high-dim space, calculate probability distribution P which represents the chance of a point being a neighbor of another point. low-dimensional embeddings of data points where such a probability distribution Q is as close as possible to P.

t-SNE

Why its output (the low-dimensional representations of data) is more often used for visualization than for further downstream analysis (clustering, etc)?

- It doesn’t learn a function from high-dim data to low-dim, so if new data points come we can’t directly convert it to low-dim.
- It doesn’t directly preserve distance but rather preserves the neighborhood of every datapoint. So the distance between points in low-dim space can’t be interpreted as close representation of the original distance. Distance-based clustering methods should be used with caution to the output of tSNE. In particular, larger distances are not preserved.
- It’s output is often good for human eyes (also considering the effect of varying parameters), but not good for automatic clustering methods like k-means.
- Still controversial.



UMAP Uniform Manifold Approximation and Projection

MDS: preserving distance

tSNE: preserving neighborhood

UMAP: preserving graph topology

Compared to t-SNE, UMAP seems to be

- faster
- deterministic

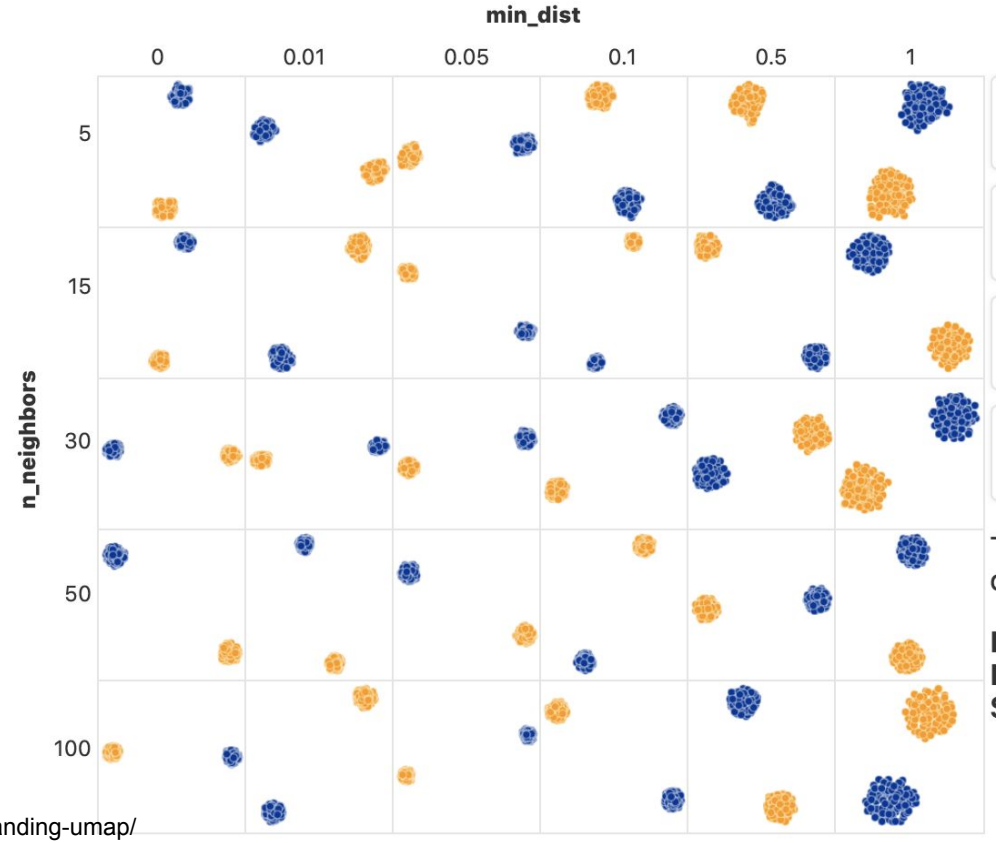
Very similar intuition but different mathematical framework

UMAP Uniform Manifold Approximation and Projection

Hyper-parameters

N_neighbors: the number of nearest neighbors to consider

Min_dist: minimum distance apart that points are allowed to be in the low dimensional representation



Two clusters with equal numbers of points, but different variances within the clusters.

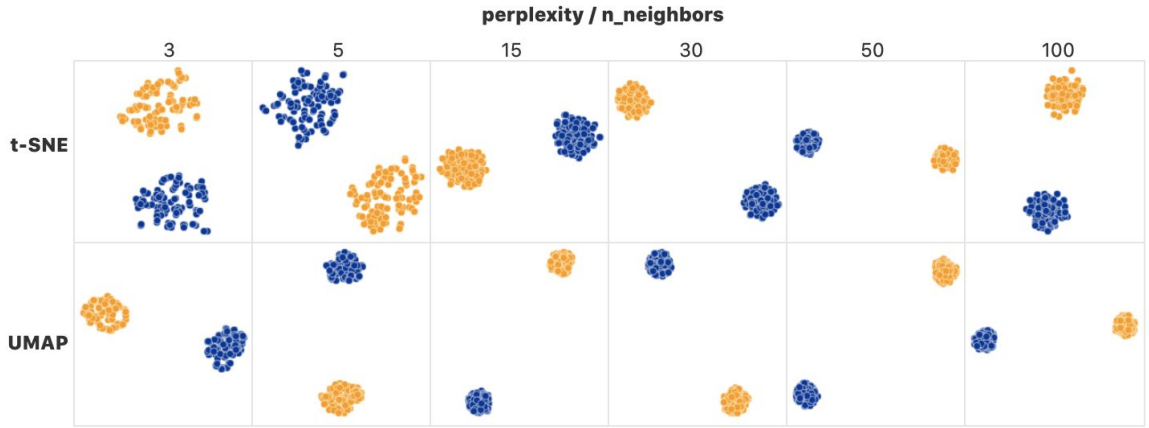
https://pair-code.github.io/understanding-umap/

UMAP Uniform Manifold Approximation and Projection

Hyper-parameters

N_neighbors: the number of nearest neighbors to consider

Min_dist: minimum distance apart that points are allowed to be in the low dimensional representation



Two clusters with equal numbers of points.

https://pair-code.github.io/understanding-umap/