Data Visualization of High Dimensional Data

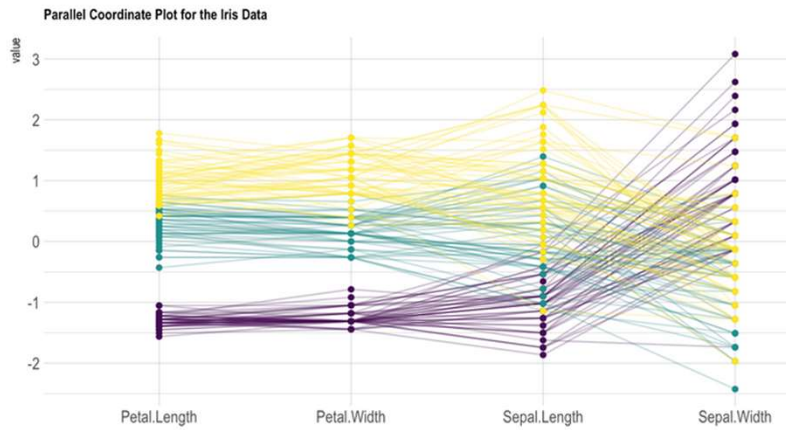
Techniques

High-Dimensional Data

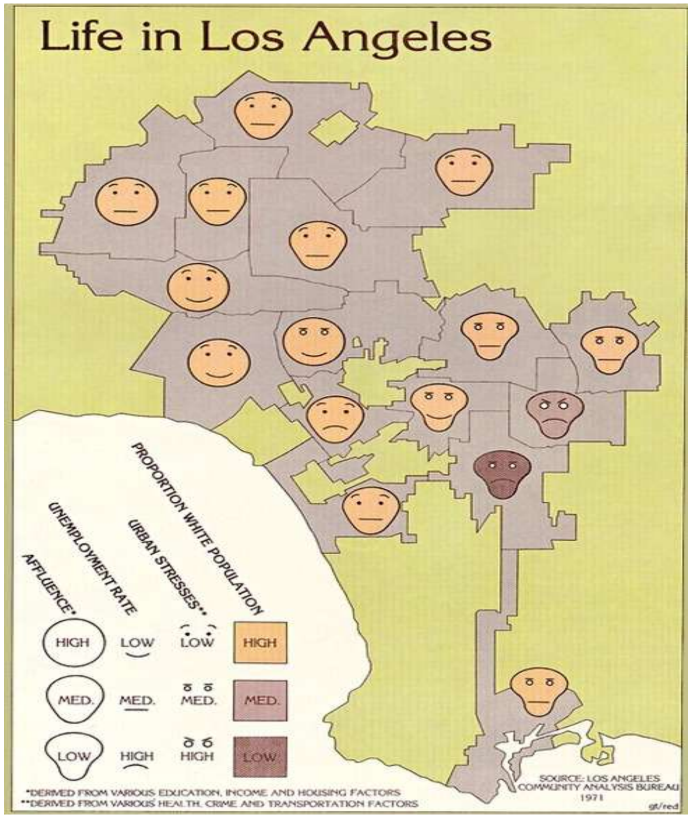
- What is meant by high-dimensional data?
  - Can be some prediction described by 30+ features
  - Images (with the pixels considered as dimensions)
- High dimensional data is hard to visualize and work with
- Different techniques have been proposed in the past

Earlier Techniques – Direct Visualization

- Parallel Co-ordinates: Allows for the comparison of multiple data records, by using parallel lines to connect points based on multiple numerical variables



- Chernoff Faces: Symbolizing Data using Faces

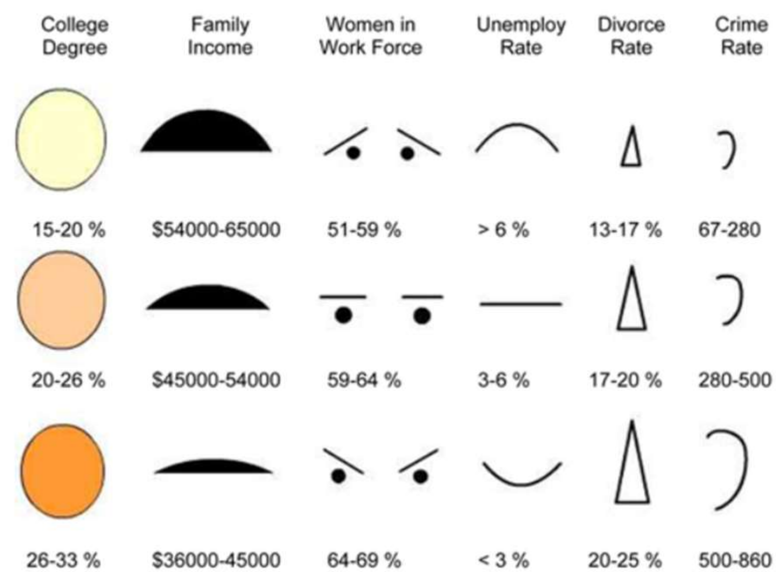


https://maphugger.com/post/44499755749/the-trouble-with-chernoff

Challenges with HD Visualization

- Direct Methods does not preserve ordinal nature of features
  - e.g., In Chernoff faces, emotions are not ordinal in eyebrow slant

- Modern Approaches therefore uses Dimensionality Reduction for visualization



High-Dimensional Data

Can be some prediction described by 30+ features

- What is meant by high-dimensional data?

Images (with the pixels considered as dimensions)

- High dimensional data is hard to visualize and work with

- Embedding to low dimensional spaces helps visualize the data

$$\boldsymbol{\mathcal{X}} = \{\boldsymbol{x_1}, \boldsymbol{x_2}, \dots, \boldsymbol{x_n} \in \mathbb{R}^{\boldsymbol{N}}\} \rightarrow \boldsymbol{\mathcal{Y}} = \{\boldsymbol{y_1}, \boldsymbol{y_2}, \dots, \boldsymbol{y_n} \in \mathbb{R}^{\boldsymbol{M}}\}$$

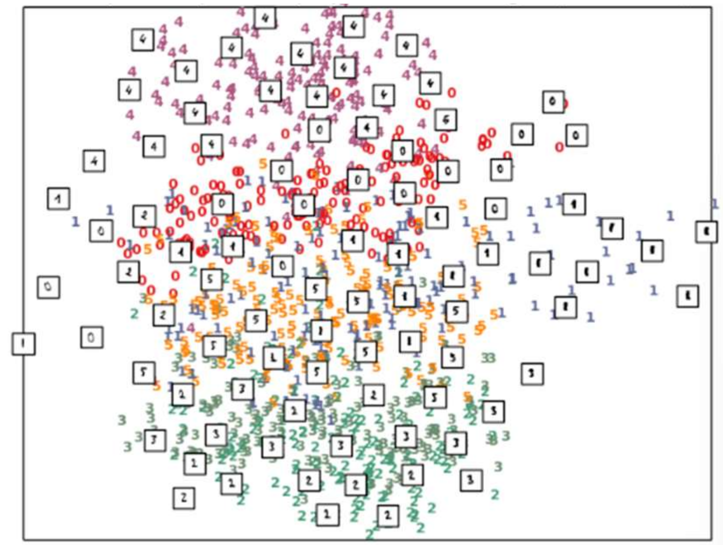
$$\min_{\boldsymbol{\mathcal{Y}}} \boldsymbol{C}(\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}})$$

Distance Preservation	MDS, Isomap
Topology Preservation	Isomap
Information Preservation	PCA

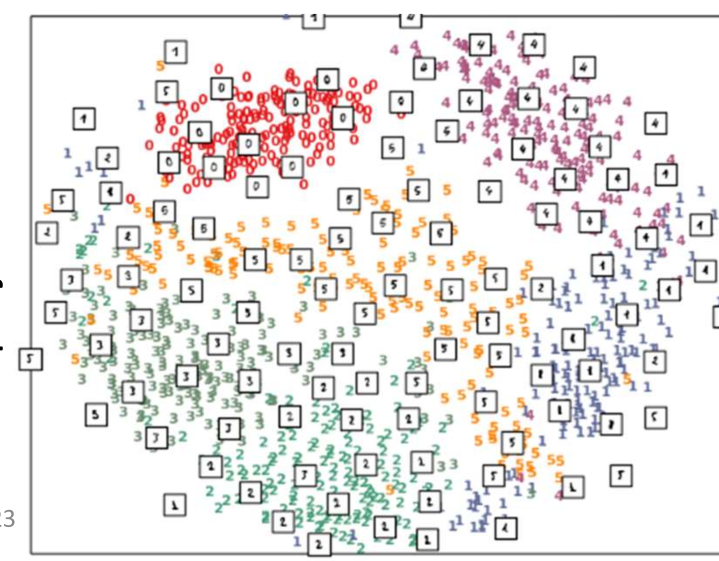
MNIST Dataset

High Dimensional Data - 10 INTRINSIC Dimensions in 8X8 pixel images

PCA projection of the digits

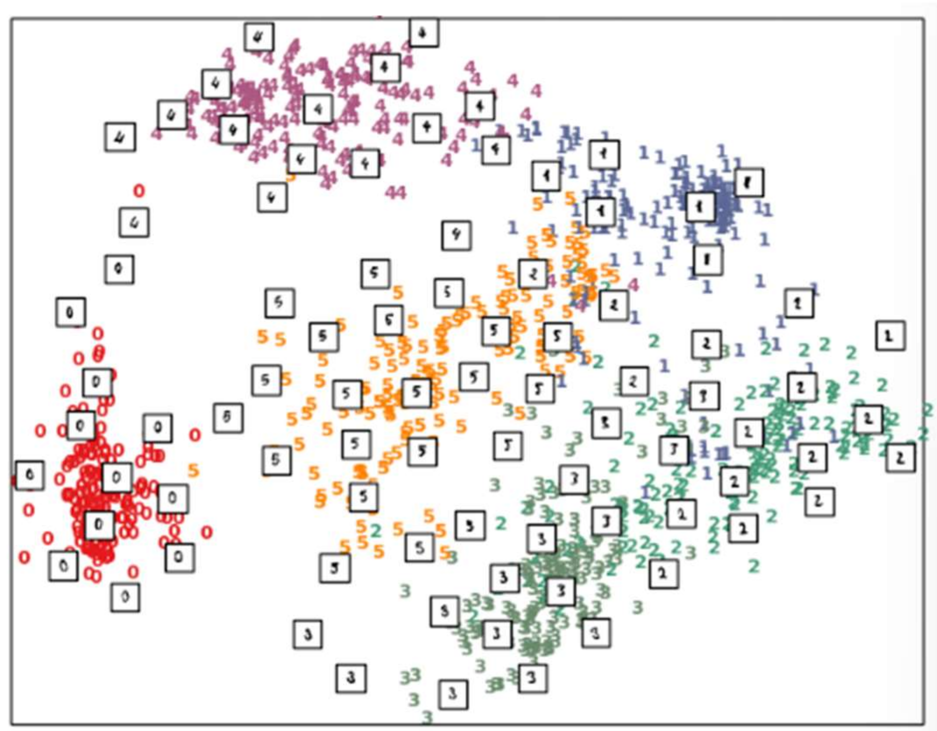


MDS projection



MNIST Dataset

ISOMAP projection of the digits



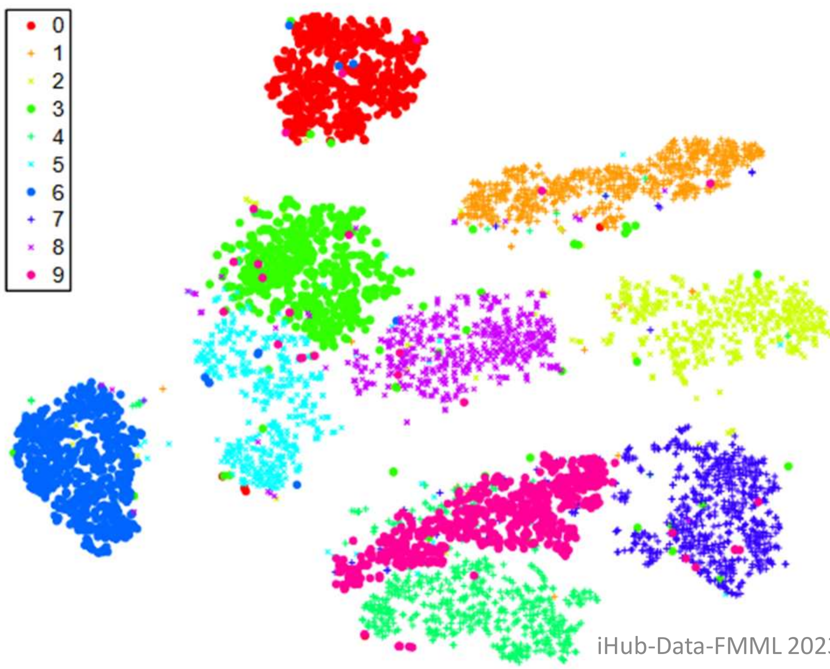
High Dimensional Data - 10 INTRINSIC Dimensions in 8X8 pixel images

Recap on ISOMAP

- Isomap focuses on preserving the global structure and geodesic distances, which can be useful for understanding the underlying manifold or shape of the data
- Isomap can be computationally expensive, particularly as the size of the dataset or the dimensionality increases
- Isomap may require careful parameter selection for optimal results, such as the number of neighbors to consider
- What to do when understanding local relationships is crucial, such as the neighbourhood relationships?
  - We today study t-SNE

Why t-SNE?

- t-SNE provides better visualizations than other methods
- It helps to uncover patterns, clusters, and relationships in the data, by preserving the local relationships and structures present in the data



Stochastic Neighbour Embedding (SNE)

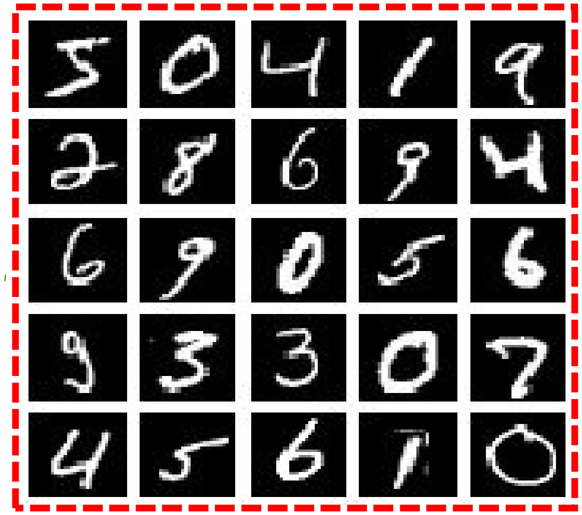
- An unsupervised technique which focusses on preserving neighbourhoods, instead of preserving distances
- Objects nearby (in a metric space) are considered neighbours
- Models the probabilities that **high-d** points $\boldsymbol{x_i}, \boldsymbol{x_j}$ are neighbours
- Models the probabilities the corresponding **low-d** points $\boldsymbol{y_i}, \boldsymbol{y_j}$ are neighbours
- SNE finds a **low-d** representation that is faithful to the **high-d** model

t-SNE Algorithm

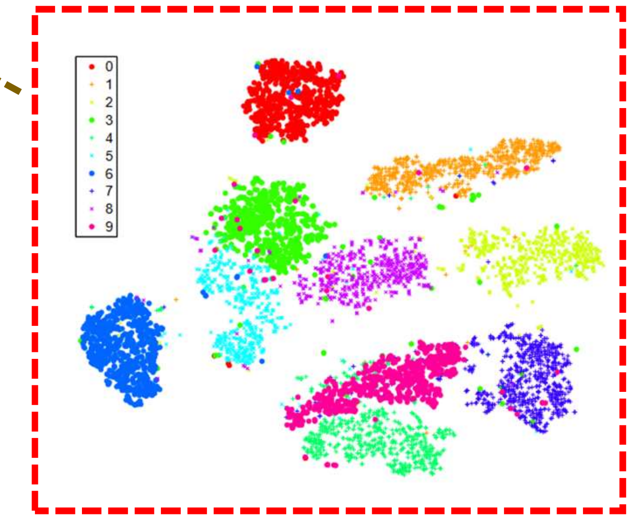
Algorithm 1: Simple version of t-Distributed Stochastic Neighbor Embedding.

Data: data set $X = \{x_1, x_2, ..., x_n\}$,
cost function parameters: perplexity *Perp*,
optimization parameters: number of iterations T, learning rate η, momentum $\alpha(t)$.
Result: low-dimensional data representation $\mathcal{Y}^{(T)} = \{y_1, y_2, ..., y_n\}$.
begin
  compute pairwise affinities $p_{j|i}$ with perplexity *Perp* (using Equation 1)
  set $p_{ij} = \frac{p_{j|i}+p_{i|j}}{2n}$
  sample initial solution $\mathcal{Y}^{(0)} = \{y_1, y_2, ..., y_n\}$ from $\mathcal{N}(0, 10^{-4}I)$
  for t=*1* **to** T **do**
    compute low-dimensional affinities q_{ij} (using Equation 4)
    compute gradient $\frac{\delta C}{\delta \mathcal{Y}}$ (using Equation 5)
    set $\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t) \left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}\right)$
  end
end

Input



Output

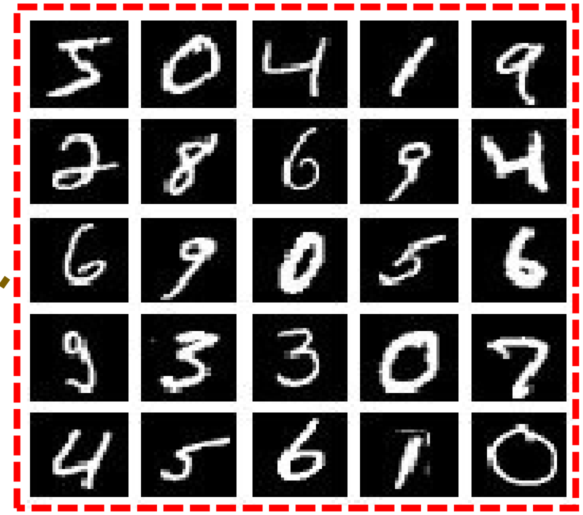


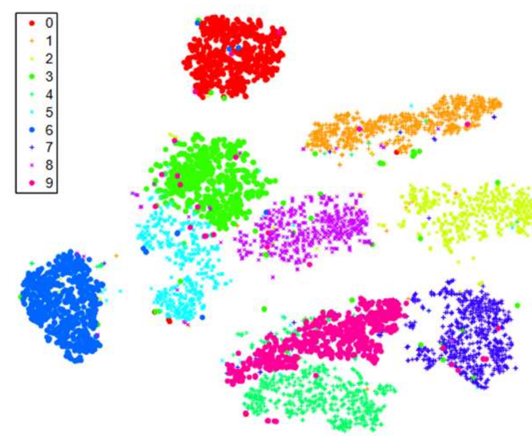
t-SNE Algorithm

Algorithm 1: Simple version of t-Distributed Stochastic Neighbor Embedding.

Data: data set $X = \{x_1, x_2, \ldots, x_n\}$,
cost function parameters: perplexity $Perp$,
optimization parameters: number of iterations T, learning rate η, momentum $\alpha(t)$.
Result: low-dimensional data representation $\mathcal{Y}^{(T)} = \{y_1, y_2, \ldots, y_n\}$.
begin
  compute pairwise affinities $p_{j|i}$ with perplexity $Perp$ (using Equation 1)
  set $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$
  sample initial solution $\mathcal{Y}^{(0)} = \{y_1, y_2, \ldots, y_n\}$ from $\mathcal{N}(0, 10^{-4} I)$
  for t=*1* **to** T **do**
    compute low-dimensional affinities q_{ij} (using Equation 4)
    compute gradient $\frac{\delta C}{\delta \mathcal{Y}}$ (using Equation 5)
    set $\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t)\left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}\right)$
  end
end

Compute probabilities P that x_i and x_j are neighbours, in high-d space



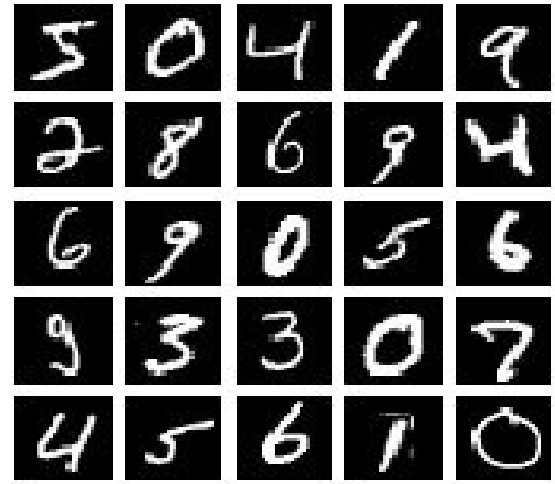


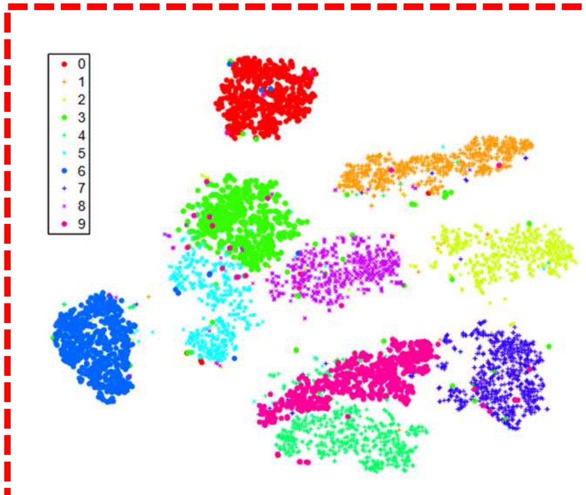
t-SNE Algorithm

Algorithm 1: Simple version of t-Distributed Stochastic Neighbor Embedding.

Data: data set $X = \{x_1, x_2, \ldots, x_n\}$,
cost function parameters: perplexity $Perp$,
optimization parameters: number of iterations T, learning rate η, momentum $\alpha(t)$.
Result: low-dimensional data representation $\mathcal{Y}^{(T)} = \{y_1, y_2, \ldots, y_n\}$.
begin
  compute pairwise affinities $p_{j|i}$ with perplexity $Perp$ (using Equation 1)
  set $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$
  sample initial solution $\mathcal{Y}^{(0)} = \{y_1, y_2, \ldots, y_n\}$ from $\mathcal{N}(0, 10^{-4} I)$
  for $t=1$ **to** T **do**
    compute low-dimensional affinities q_{ij} (using Equation 4)
    compute gradient $\frac{\delta C}{\delta \mathcal{Y}}$ (using Equation 5)
    set $\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t) \left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}\right)$
  end
end

Compute probabilities Q that y_i and y_j are neighbours in low-d space (corresponding to x_i and x_j)



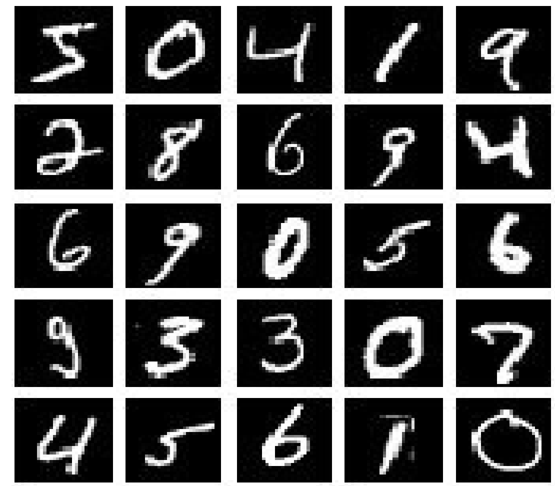


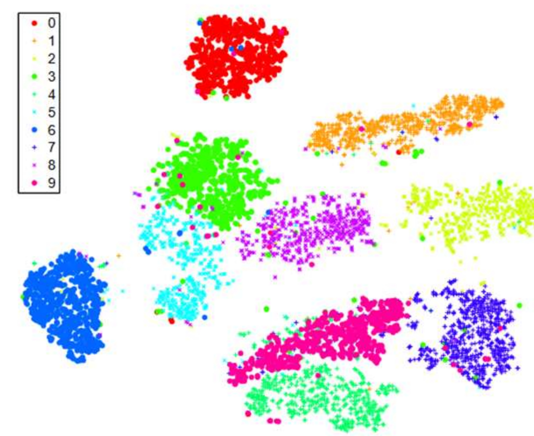
t-SNE Algorithm

Algorithm 1: Simple version of t-Distributed Stochastic Neighbor Embedding.

Data: data set $X = \{x_1, x_2, \ldots, x_n\}$,
cost function parameters: perplexity $Perp$,
optimization parameters: number of iterations T, learning rate η, momentum $\alpha(t)$.
Result: low-dimensional data representation $\mathcal{Y}^{(T)} = \{y_1, y_2, \ldots, y_n\}$.
begin
  compute pairwise affinities $p_{j|i}$ with perplexity $Perp$ (using Equation 1)
  set $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$
  sample initial solution $\mathcal{Y}^{(0)} = \{y_1, y_2, \ldots, y_n\}$ from $\mathcal{N}(0, 10^{-4} I)$
  for $t=1$ **to** T **do**
    compute low-dimensional affinities q_{ij} (using Equation 4)
    compute gradient $\frac{\delta C}{\delta \mathcal{Y}}$ (using Equation 5)
    set $\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t) \left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)} \right)$
  end
end

Key assumption is that the high-d P and the low-d Q probability distributions should be the same



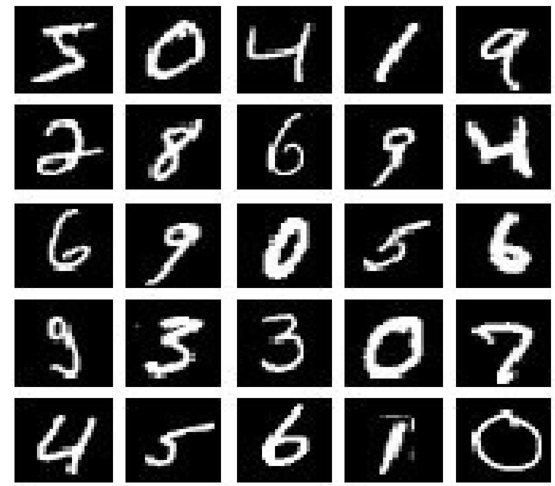


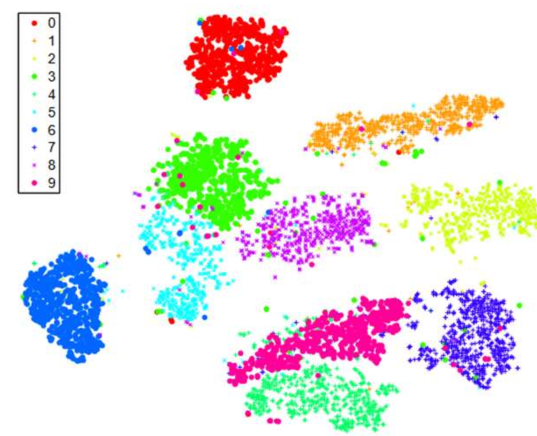
t-SNE Algorithm

Algorithm 1: Simple version of t-Distributed Stochastic Neighbor Embedding.

Data: data set $\mathcal{X} = \{x_1, x_2, ..., x_n\}$,
cost function parameters: perplexity $Perp$,
optimization parameters: number of iterations T, learning rate η, momentum $\alpha(t)$.
Result: low-dimensional data representation $\mathcal{Y}^{(T)} = \{y_1, y_2, ..., y_n\}$.
begin
  compute pairwise affinities $p_{j|i}$ with perplexity $Perp$ (using Equation 1)
  set $p_{ij} = \frac{p_{j|i}+p_{i|j}}{2n}$
  sample initial solution $\mathcal{Y}^{(0)} = \{y_1, y_2, ..., y_n\}$ from $\mathcal{N}(0, 10^{-4}I)$
  for t=1 **to** T **do**
    compute low-dimensional affinities q_{ij} (using Equation 4)
    compute gradient $\frac{\delta C}{\delta \mathcal{Y}}$ (using Equation 5)
    set $\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t)\left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}\right)$
  end
end

Goal: To find a **low-d** map that minimizes the difference between the P(**high-d**) and Q (**low-d**) distributions
(if x_i, x_j has high probability of being neighbours in **high-d,** then y_i, y_j should have high probability in **low-d**)



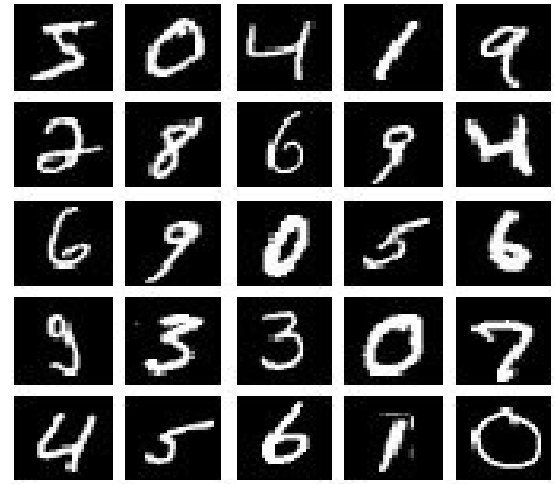


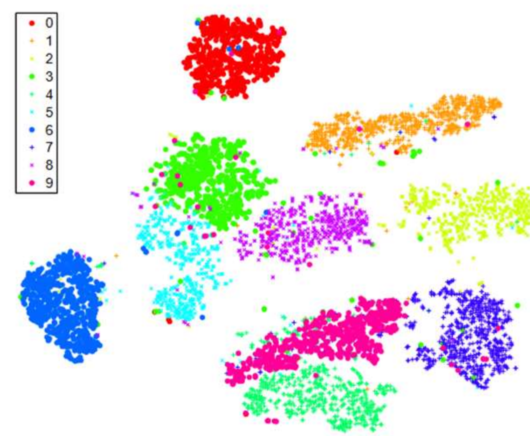
t-SNE Algorithm

Algorithm 1: Simple version of t-Distributed Stochastic Neighbor Embedding.

Data: data set $X = \{x_1, x_2, \ldots, x_n\}$,
cost function parameters: perplexity $Perp$,
optimization parameters: number of iterations T, learning rate η, momentum $\alpha(t)$.
Result: low-dimensional data representation $\mathcal{Y}^{(T)} = \{y_1, y_2, \ldots, y_n\}$.
begin
  compute pairwise affinities $p_{j|i}$ with perplexity $Perp$ (using Equation 1)
  set $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$
  sample initial solution $\mathcal{Y}^{(0)} = \{y_1, y_2, \ldots, y_n\}$ from $\mathcal{N}(0, 10^{-4} I)$
  for $t=1$ **to** T **do**
    compute low-dimensional affinities q_{ij} (using Equation 4)
    compute gradient $\frac{\delta C}{\delta \mathcal{Y}}$ (using Equation 5)
    set $\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t)\left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}\right)$
  end
end

The difference between the **high-d** and **low-d** maps are minimized using **gradient descent**





t-SNE Algorithm - Hyperparameters

Algorithm 1: Simple version of t-Distributed Stochastic Neighbor Embedding.

Data: data set $X = \{x_1, x_2, \ldots, x_n\}$,
cost function parameters: perplexity $Perp$,
optimization parameters: number of iterations T, learning rate η, momentum $\alpha(t)$.
Result: low-dimensional data representation $\mathcal{Y}^{(T)} = \{y_1, y_2, \ldots, y_n\}$.
begin
    compute pairwise affinities $p_{j|i}$ with perplexity $Perp$ (using Equation 1)
    set $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$
    sample initial solution $\mathcal{Y}^{(0)} = \{y_1, y_2, \ldots, y_n\}$ from $\mathcal{N}(0, 10^{-4} I)$
    for $t=1$ **to** T **do**
        compute low-dimensional affinities q_{ij} (using Equation 4)
        compute gradient $\frac{\delta C}{\delta \mathcal{Y}}$ (using Equation 5)
        set $\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t) \left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}\right)$
    end
end

Measures the effective number of neighbours, usually between 5 and 50

Momentum encourages a step that is in the same direction as previous steps

GD steps in the direction the error is the minimum. η defines how big will be the step

Parameters to speed up optimization and avoid poor local minima

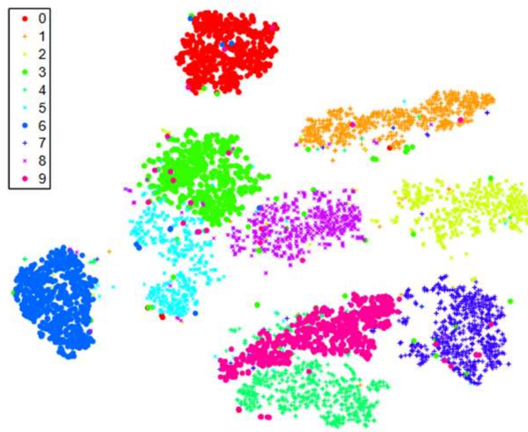
Number of steps needed to reach the optimization as GD is iterative

t-SNE Algorithm

Algorithm 1: Simple version of t-Distributed Stochastic Neighbor Embedding.

Data: data set $X = \{x_1, x_2, \ldots, x_n\}$,
cost function parameters: perplexity $Perp$,
optimization parameters: number of iterations T, learning rate η, momentum $\alpha(t)$.
Result: low-dimensional data representation $\mathcal{Y}^{(T)} = \{y_1, y_2, \ldots, y_n\}$.
begin
  compute pairwise affinities $p_{j|i}$ with perplexity $Perp$ (using Equation 1)
  set $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$
  sample initial solution $\mathcal{Y}^{(0)} = \{y_1, y_2, \ldots, y_n\}$ from $\mathcal{N}(0, 10^{-4}I)$
  for t=*1* **to** T **do**
    compute low-dimensional affinities q_{ij} (using Equation 4)
    compute gradient $\frac{\delta C}{\delta \mathcal{Y}}$ (using Equation 5)
    set $\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t) \left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)} \right)$
  end
end

This is the low-d solution corresponding to the high-d data that we want to optimize for



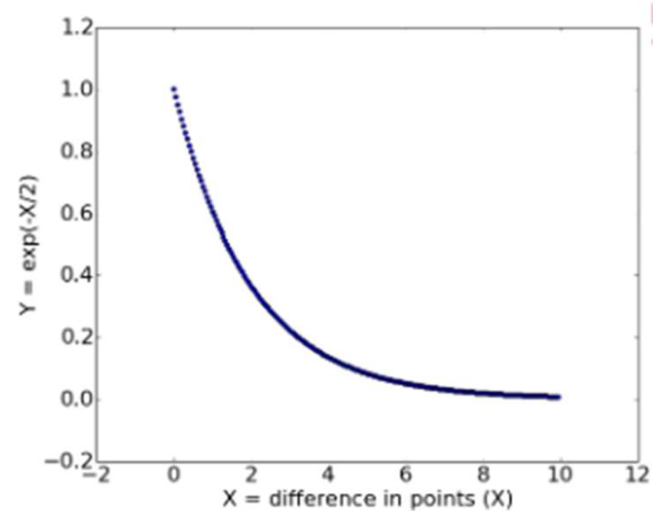
t-SNE Algorithm

Algorithm 1: Simple version of t-Distributed Stochastic Neighbor Embedding.

Data: data set $\mathcal{X} = \{x_1, x_2, ..., x_n\}$,
cost function parameters: perplexity $Perp$,
optimization parameters: number of iterations T, learning rate η, momentum $\alpha(t)$.
Result: low-dimensional data representation $\mathcal{Y}^{(T)} = \{y_1, y_2, ..., y_n\}$.
begin
  compute pairwise affinities $p_{j|i}$ with perplexity $Perp$ (using Equation 1)
  set $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$
  sample initial solution $\mathcal{Y}^{(0)} = \{y_1, y_2, ..., y_n\}$ from $\mathcal{N}(0, 10^{-4}I)$
  for t=1 **to** T **do**
    compute low-dimensional affinities q_{ij} (using Equation 4)
    compute gradient $\frac{\delta C}{\delta \mathcal{Y}}$ (using Equation 5)
    set $\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t) \left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}\right)$
  end
end

The probability that $\boldsymbol{x_i}$ would choose $\boldsymbol{x_j}$ as its neighbours, in the **high-d** space

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



- x_i that are "close" (low Euclidean distance) return a high value
- x_i that are "far" (high Euclidean distance) return a low value

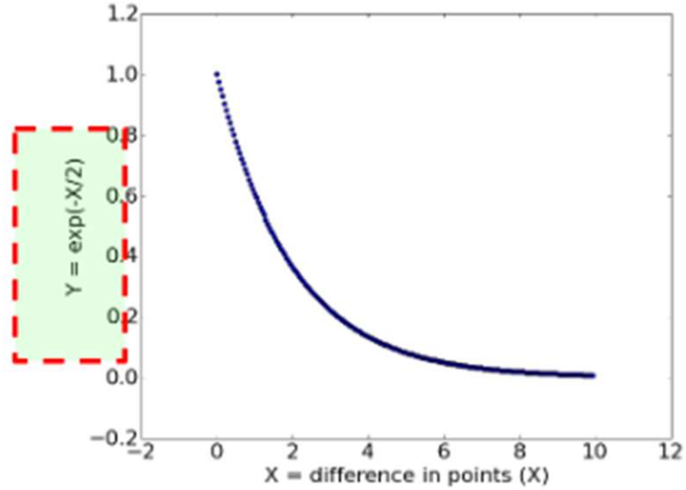
t-SNE Algorithm

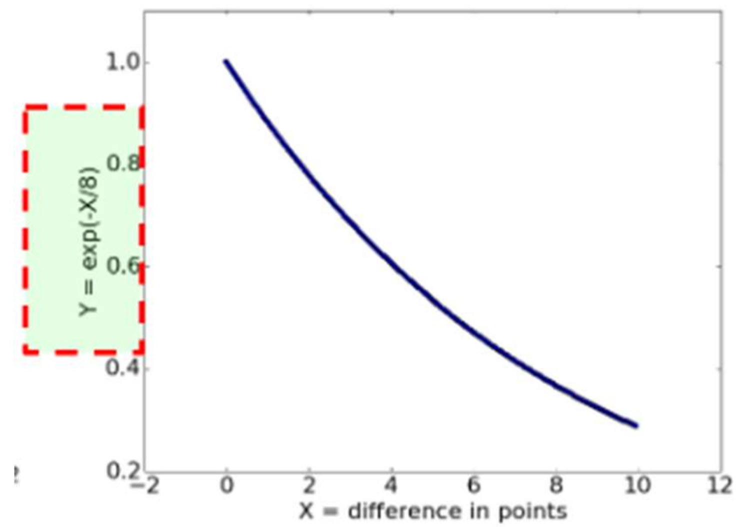
Algorithm 1: Simple version of t-Distributed Stochastic Neighbor Embedding.

Data: data set $X = \{x_1, x_2, \ldots, x_n\}$,
cost function parameters: perplexity $Perp$,
optimization parameters: number of iterations T, learning rate η, momentum $\alpha(t)$.
Result: low-dimensional data representation $\mathcal{Y}^{(T)} = \{y_1, y_2, \ldots, y_n\}$.
begin
    compute pairwise affinities $p_{j|i}$ with perplexity $Perp$ (using Equation 1)
    set $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$
    sample initial solution $\mathcal{Y}^{(0)} = \{y_1, y_2, \ldots, y_n\}$ from $\mathcal{N}(0, 10^{-4} I)$
    for t=*1* **to** T **do**
        compute low-dimensional affinities q_{ij} (using Equation 4)
        compute gradient $\frac{\delta C}{\delta \mathcal{Y}}$ (using Equation 5)
        set $\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t) \left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}\right)$
    end
end

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

σ is the variance, change of variance will change the values we assign to the distances





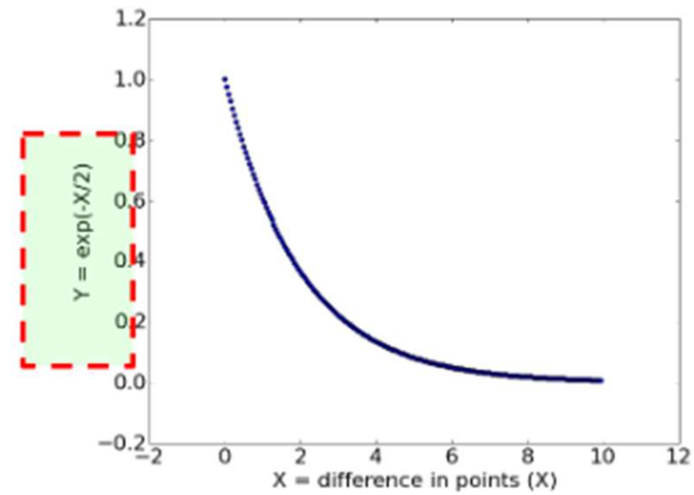
t-SNE Algorithm

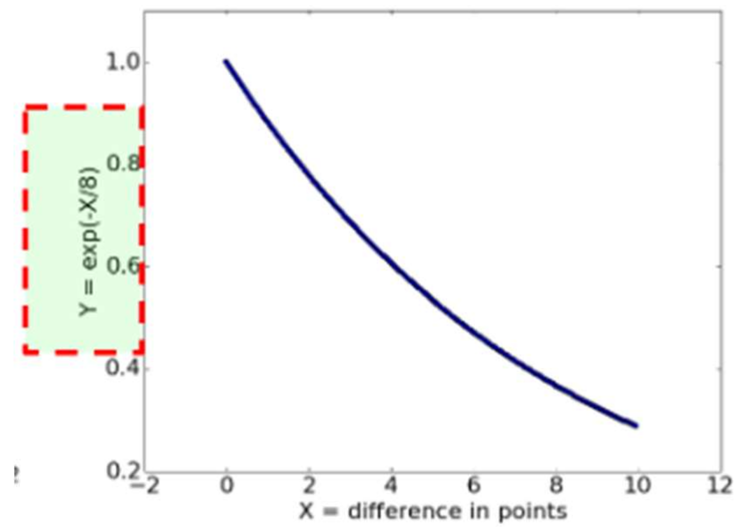
$$p_{j|i} = \frac{\exp(-|x_i - x_j|^2/2\sigma_i^2)}{\sum_{k \neq i} exp(-|x_i - x_k|^2/2\sigma_i^2)}$$

- A single variance is not likely optimal since the density of the data will vary

- In **dense** regions, we probably want a **smaller** variance
- In **sparse** regions, we probably want a **larger** variance

- Given a user specified Perplexity, the algorithm finds the variance that yield the required Perplexity





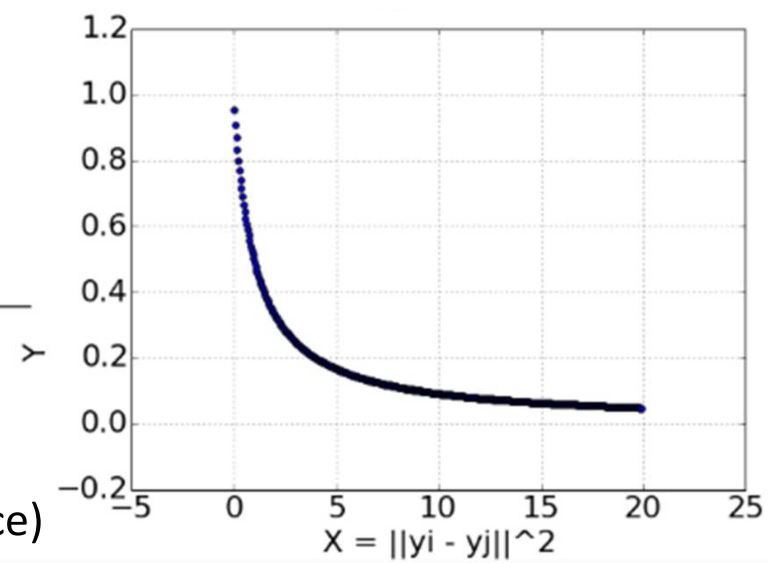
t-SNE Algorithm

Algorithm 1: Simple version of t-Distributed Stochastic Neighbor Embedding.

Data: data set $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$,
cost function parameters: perplexity *Perp*,
optimization parameters: number of iterations T, learning rate η, momentum $\alpha(t)$.
Result: low-dimensional data representation $\mathcal{Y}^{(T)} = \{y_1, y_2, \ldots, y_n\}$.
begin
  compute pairwise affinities $p_{j|i}$ with perplexity *Perp* (using Equation 1)
  set $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$
  sample initial solution $\mathcal{Y}^{(0)} = \{y_1, y_2, \ldots, y_n\}$ from $\mathcal{N}(0, 10^{-4} I)$
  for $t=1$ **to** T **do**
    compute low-dimensional affinities q_{ij} (using Equation 4)
    compute gradient $\frac{\delta C}{\delta \mathcal{Y}}$ (using Equation 5)
    set $\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t) \left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}\right)$
  end
end

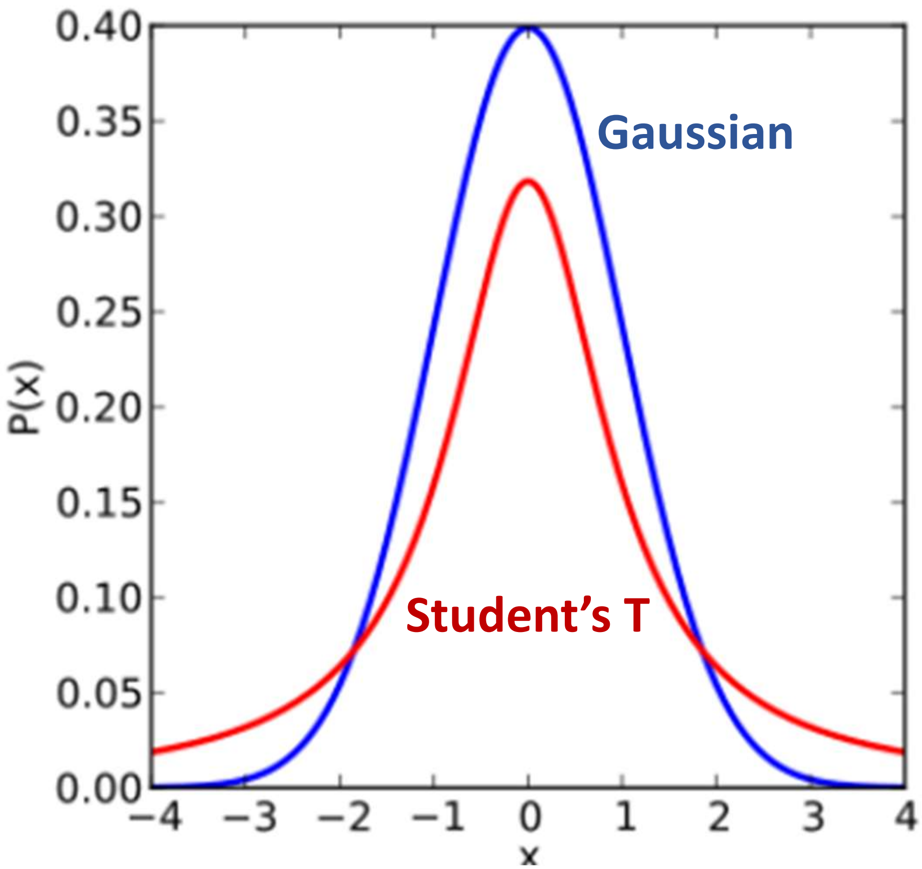
The probability that $\boldsymbol{y_i}$ would choose $\boldsymbol{y_j}$ as its neighbours, in the **low-d** space

$$q_{ij} = \frac{(1 + |y_i - y_j|^2)^{-1}}{\sum_{k \neq l} (1 + |y_k - y_l|^2)^{-1}}$$



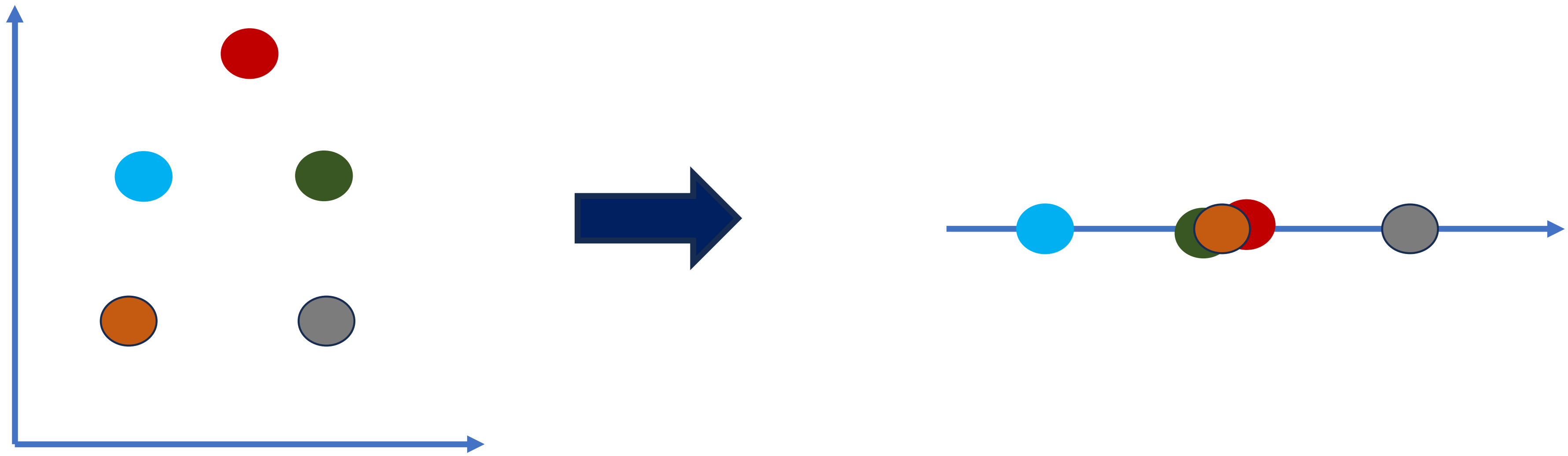
They use a Student t-distribution, with similar points (low distance) getting a high probability

Gaussian vs. Student t-distribution

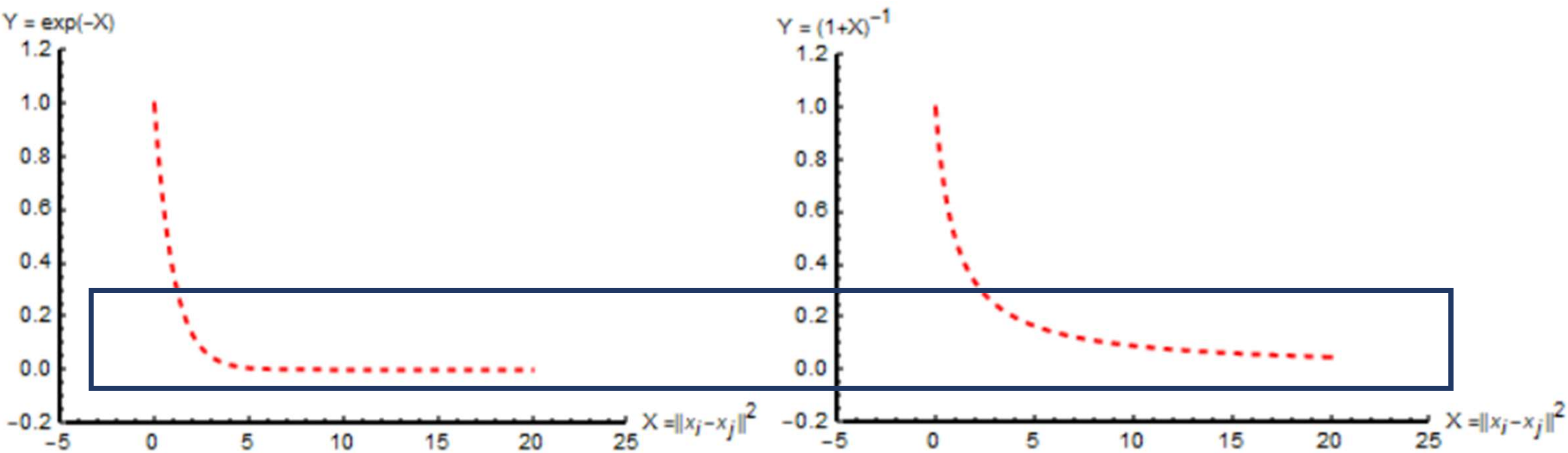


- Student's have longer tails compared to Gaussian
- Gives higher probabilities to points that are further away
- Desirable as, we have limited **low-d** space and want to focus on modelling the close **high-d** points.
- Want to have moderately far high-d points further apart in the low-d space (avoids crowding)

Crowding in SNE



Gaussian vs. Student t-distribution



t-SNE Algorithm

Algorithm 1: Simple version of t-Distributed Stochastic Neighbor Embedding.

Data: data set $\mathcal{X} = \{x_1, x_2, ..., x_n\}$,
cost function parameters: perplexity $Perp$,
optimization parameters: number of iterations T, learning rate η, momentum $\alpha(t)$.
Result: low-dimensional data representation $\mathcal{Y}^{(T)} = \{y_1, y_2, ..., y_n\}$.
begin
  $\boxed{\text{compute pairwise affinities } p_{j|i} \text{ with perplexity } Perp \text{ (using Equation 1)}}$
  $\boxed{\text{set } p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}}$
  sample initial solution $\mathcal{Y}^{(0)} = \{y_1, y_2, ..., y_n\}$ from $\mathcal{N}(0, 10^{-4}I)$
  for $t=1$ **to** T **do**
    $\boxed{\text{compute low-dimensional affinities } q_{ij} \text{ (using Equation 4)}}$
    $\boxed{\text{compute gradient } \frac{\delta C}{\delta \mathcal{Y}} \text{ (using Equation 5)}}$
    $\boxed{\text{set } \mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t)\left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}\right)}$
  end
end

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

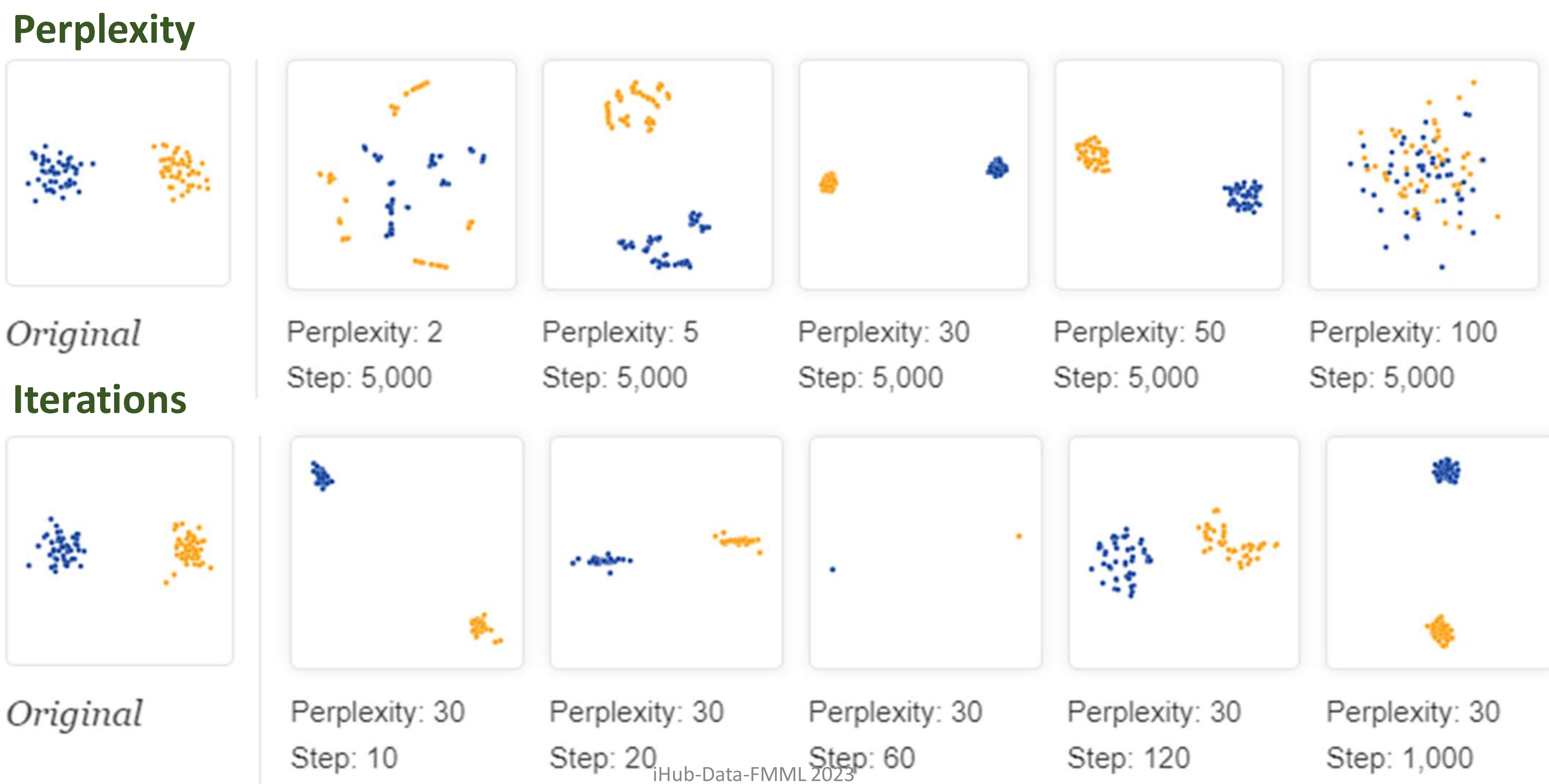
$$q_{ij} = \frac{(1 + |y_i - y_j|^2)^{-1}}{\sum_{k \neq l}(1 + |y_k - y_l|^2)^{-1}}$$

- The **low-d** points are moved around to minimize the difference between the two distributions

$$C = KL(P||Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

- We find a low−d representation that captures the high−d data after successive iterations

Dependence on Hyperparameters

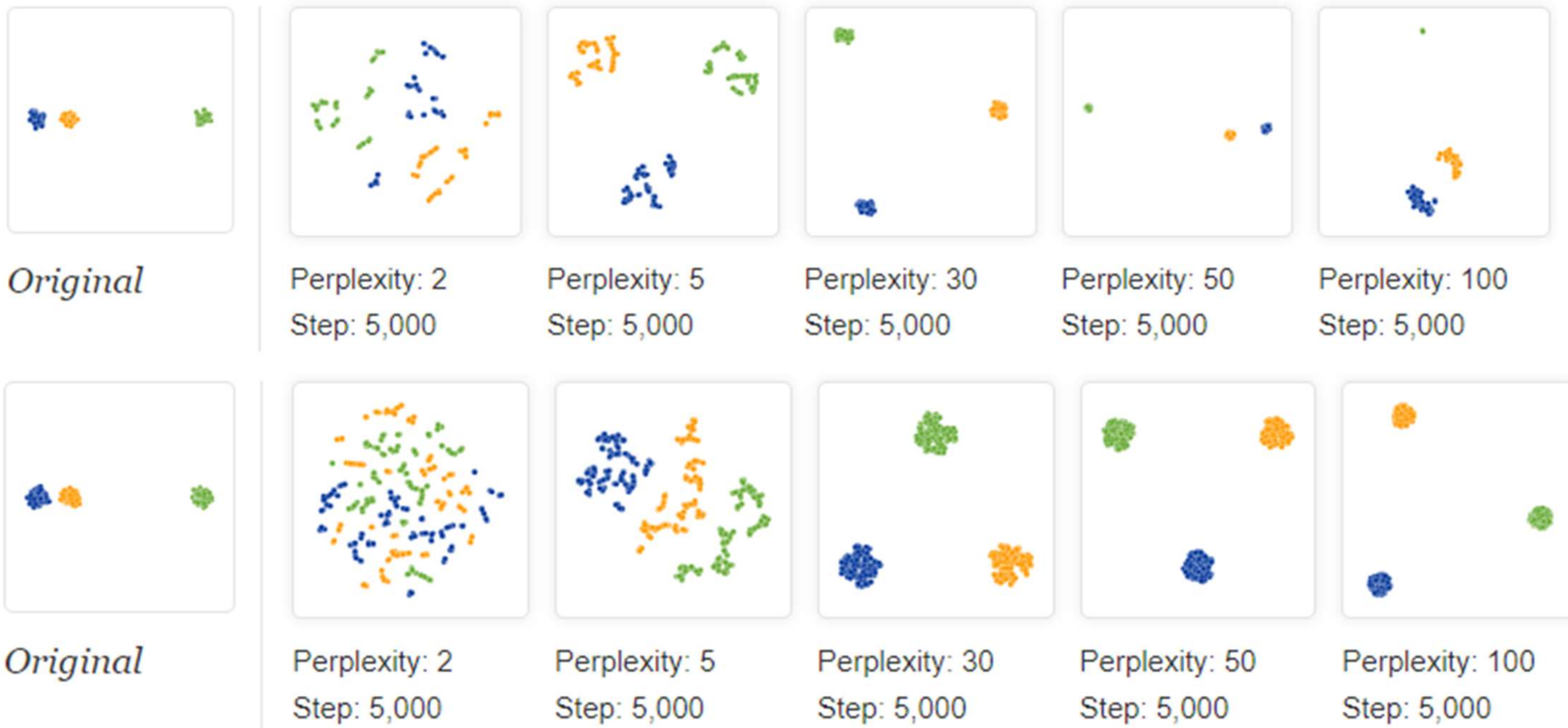


Interpretation of t-SNE clusters

Clusters with different standard deviations and sizes

- **You cannot see relative size of clusters in a t-SNE plot**

Distance between clusters



Distance between clusters

- Fine tuning perplexity is required for seeing global geometry
- One perplexity value is not sufficient to capture distances across all clusters
- There is no correct interpretation between well-separated clusters in a t-SNE plot

Understanding Topology with t-SNE

Original | Perplexity: 2, Step: 5,000 | Perplexity: 5, Step: 5,000 | Perplexity: 30, Step: 5,000 | Perplexity: 50, Step: 5,000 | Perplexity: 100, Step: 5,000

Original | Perplexity: 50, Step: 5,000 | Perplexity: 50, Step: 5,000 | Perplexity: 50, Step: 5,000 | Perplexity: 50, Step: 5,000 | Perplexity: 50, Step: 5,000

Take-Home Points

- Understanding t-SNE
  - Mainly used for visualization purposes, not DR
- Running t-SNE
  - Always run multiple trials
  - Use appropriate perplexity
  - Let the samples stabilize (iterations)
- Reading t-SNE
  - Do not give importance to distances between far-away points
  - Do not give importance to density of clusters
  - Do not infer anything from a single output

References

- https://kawahara.ca/visualizing-data-using-t-sne-slides/
- https://distill.pub/2016/misread-tsne/
- https://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf