# Advanced Machine Learning

Lecture 8: Dimension Reduction and Data Visualization

Nora Ouzir : nora.ouzir@centralesupelec.fr
Lucca Guardiola : lucca.guardiola@centralesupelec.fr

Oct. - Nov. 2020

CentraleSupélec

# Content

1. Reminders on ML
2. Robust regression
3. Hierarchical clustering
4. Classification and supervised learning
5. Non-negative matrix factorization
6. Mixture models fitting
7. Model order selection
8. Dimension reduction and data visualization

# Today's Lecture

1. Introduction

2. Reminders on PCA

3. Stochastic Neighbor Embedding

# Today's lecture

1. Introduction
2. Reminders on PCA
3. Stochastic Neighbor Embedding

# Why should we reduce the dimension?

Huge number of dimensions causes problems....

- Data becomes very sparse, some algorithms become meaningless (e.g., density based clustering)
- Complexity usually depends on the dimensionality
- Eliminate non-relevant and/or noisy features
- Interpretation
- Visualize structure
- Storage, processing, ...

Data can be described with fewer dimensions, without losing much of its meaning

# Today's lecture

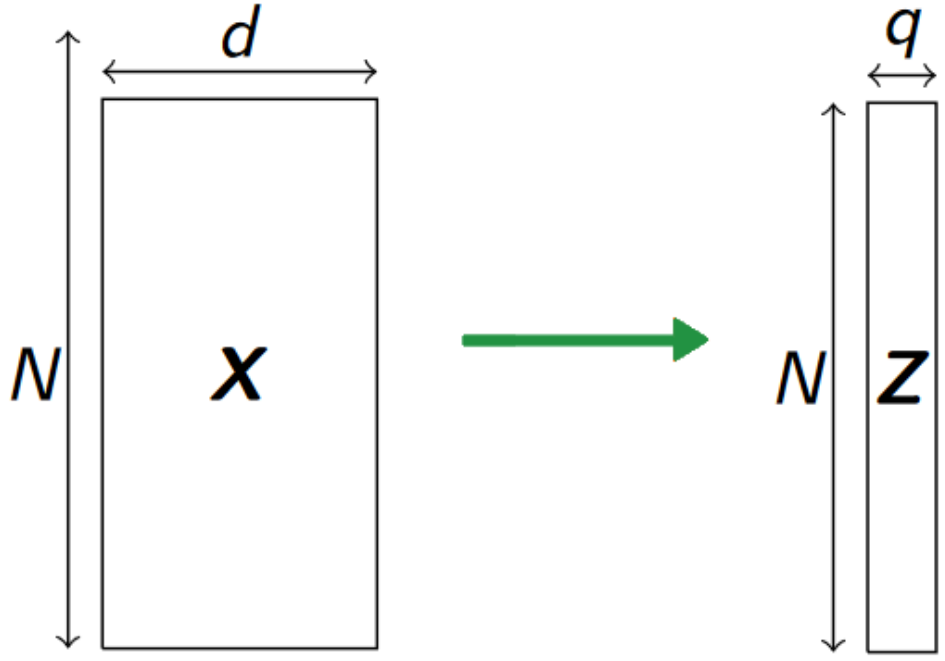
1. Introduction
2. Reminders on PCA
3. Stochastic Neighbor Embedding

# PCA: principle

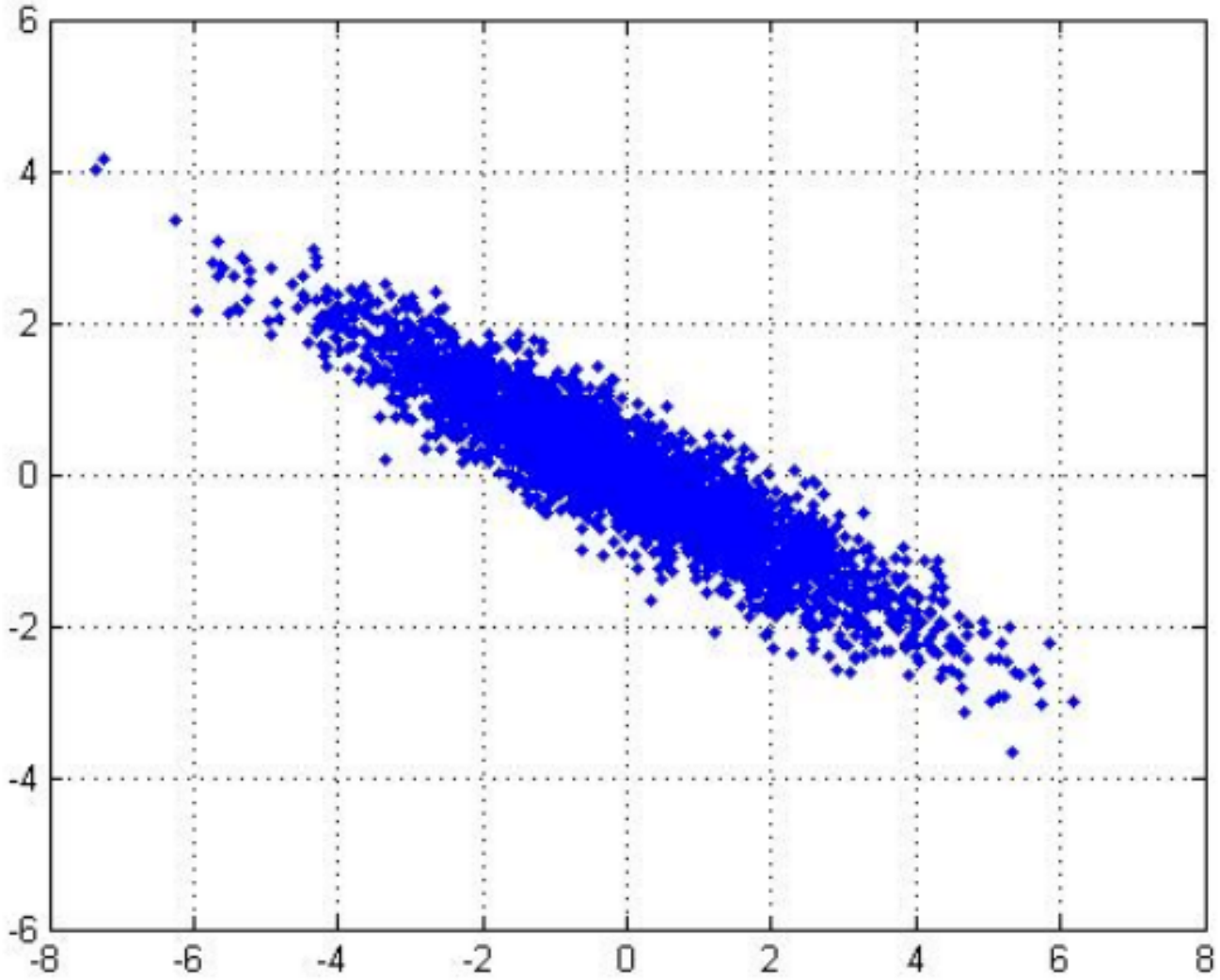
Linear orthogonal projection $\boldsymbol{P} \in \mathbb{R}^{d \times q}$ of the data $\boldsymbol{X} \in \mathbb{R}^{N,d}$ into a subspace of dimension $q < d$ s.t the variance of the projected data is maximized

$$\boldsymbol{X} = \boldsymbol{Z}\boldsymbol{P}^T + noise$$

where $\boldsymbol{Z} \in \mathbb{R}^{N \times q}$



# Example: 2D Gaussian



# PCA: Two equivalent views

1. Minimization of the reconstruction error between $\boldsymbol{x}$ and $\boldsymbol{P}^T\boldsymbol{P}\boldsymbol{x}$

$$\min_{\boldsymbol{P}\in\mathbb{R}^{N\times q}} \sum_{i=1}^{N} \|\mathbf{x}_i - \boldsymbol{P}\boldsymbol{P}^T\mathbf{x}_i\|^2$$

2. Maximization of variance: $q$ largest eigenvalues of $\Sigma$

$$\max_{\boldsymbol{P}\in\mathbb{R}^d} \|\mathbf{X}\boldsymbol{p}\|_2^2 \text{ with } \|\boldsymbol{p}\|_2^2 = 1 \text{ and } \boldsymbol{Z} = \mathbf{X}\boldsymbol{p}$$

## Classic matrix computation tools

- Eigenvalues of the covariance matrix $\Sigma \rightarrow$ capture variance direction and scale: $\Sigma = \boldsymbol{U}\Lambda\boldsymbol{U}^T$
- Singular Value Decomposition: $\mathbf{X} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T$

These are equivalent views $\rightarrow$

# Minimization of error $\rightsquigarrow$ Maximization of variance

$$
\begin{aligned}
J(\boldsymbol{P}) &= \sum_{i=1}^{N} \|\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i\|^2 = \sum_{i=1}^{N} (\boldsymbol{x}_i - \boldsymbol{P}\boldsymbol{P}^\top \boldsymbol{x}_i)^\top (\boldsymbol{x}_i - \boldsymbol{P}\boldsymbol{P}^\top \boldsymbol{x}_i) \\
&= \sum_{i=1}^{N} (\boldsymbol{x}_i^\top \boldsymbol{x}_i - 2\boldsymbol{x}_i^\top \boldsymbol{P}\boldsymbol{P}^\top \boldsymbol{x}_i + \boldsymbol{x}_i^\top \boldsymbol{P}\boldsymbol{P}^\top \boldsymbol{P}\boldsymbol{P}^\top \boldsymbol{x}_i) \\
&= \sum_{i=1}^{N} \boldsymbol{x}_i^\top \boldsymbol{x}_i - \sum_{i=1}^{N} \boldsymbol{x}_i^\top \boldsymbol{P}\boldsymbol{P}^\top \boldsymbol{x}_i = \boldsymbol{x}_i^\top \boldsymbol{x}_i - \sum_{i=1}^{N} \boldsymbol{z}_i^\top \boldsymbol{z}_i \\
&= trace\left(\sum_{i=1}^{N} \boldsymbol{x}_i \boldsymbol{x}_i^\top - \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{z}_i \boldsymbol{z}_i^\top\right) = trace\left(\sum_{i=1}^{N} \boldsymbol{x}_i \boldsymbol{x}_i^\top - \sum_{i=1}^{N} \boldsymbol{P}^\top \boldsymbol{x}_i \boldsymbol{x}_i^\top \boldsymbol{P}\right) \\
J(\boldsymbol{P}) &= trace\left(\boldsymbol{X}^\top \boldsymbol{X}\right) - trace\left(\boldsymbol{P}^\top \boldsymbol{X}^\top \boldsymbol{X} \boldsymbol{P}\right)
\end{aligned}
$$

Minimize $\boldsymbol{J(P)}$ $\iff$ maximize variance of the projection wrt $\boldsymbol{P}$

# Whiteboard

# PCA: Solution

The solution to the PCA problem is the matrix $\boldsymbol{P}$ whose columns are the $q$ eigenvectors of the covariance matrix $\Sigma$ corresponding to the $q$ largest eigenvalues

- $\rightarrow$ N.B: The solution can be obtained with SVD of the data matrix

$$\mathbf{X} = \boldsymbol{USV}^T$$

## Choice of $q$

Choose the $q$ largest components that explain *e.g.*, 90% of the variance in the data

$$\text{total variance} \rightarrow \textit{trace}(\Lambda) = \lambda_1 + ... + \lambda_d$$

The first $q$ components account for: $\frac{\lambda_1 + ... + \lambda_q}{\lambda_1 + ... + \lambda_d}$

# PCA: Algorithm

## Algorithm

1. Normalize the data $\mathbf{X} \in \mathbb{R}^{N \times d}$: $\mathbf{X}_{ij} = \frac{\mathbf{X}_{ij} - \mu_j}{\sigma_j}, j = 1, ..., d$ and $i = 1, .., N$
2. Compute the correlation matrix $\Sigma = \frac{1}{N-1}\mathbf{X}^T\mathbf{X}$
3. Find the eigenvalue decomposition of $\Sigma$: $\{\boldsymbol{p}_i \in \mathbb{R}^d, \lambda_j \in \mathbb{R}\}$
4. Order the eigenvalues $\lambda_j$ by decreasing order
5. The projection matrix is

$$\boldsymbol{P} = (\boldsymbol{p}_1, ..., \boldsymbol{p}_q) \in \mathbb{R}^{d \times q}$$

where $\{\boldsymbol{p}_1, ..., \boldsymbol{p}_q\}$ are the eigenvectors associated with the $q$ largest eigenvalues

# PCA: Summary

- Linear dimension reduction method
- PCA projects the data onto a subspace which maximizes the projected variance, or equivalently, minimizes the reconstruction error.
- The optimal subspace is given by the top eigenvectors of the empirical covariance matrix.
- PCA gives a set of decorrelated features.

## Extensions

- Kernel PCA: non-linear projection
- Probabilistic PCA
  - → Bypass covariance matrix computation, missing data, mixtures, ...
  - → Bishop 2006 Chapter 12.2

# Example: Non-linear subspace

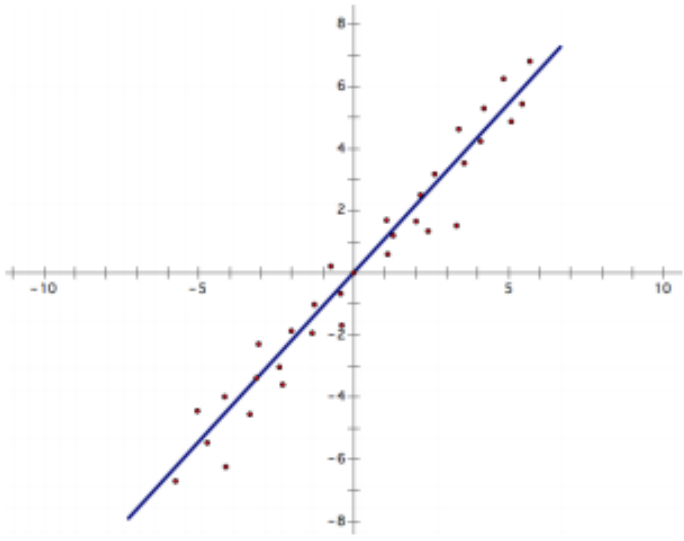



**Figure 1.** Data lying near a linear subspace

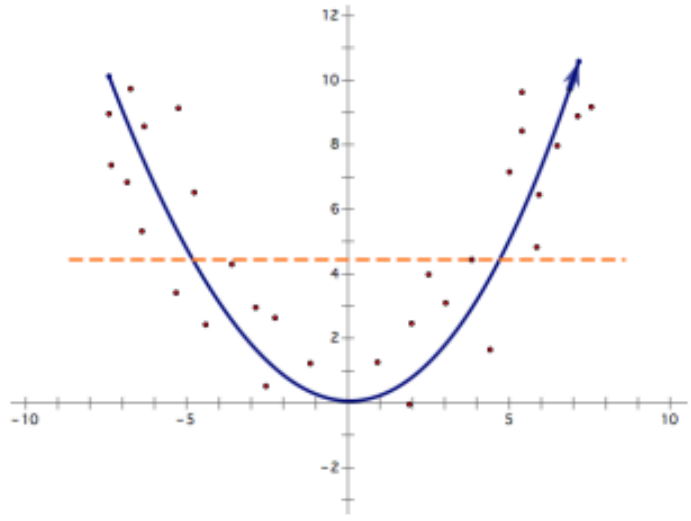


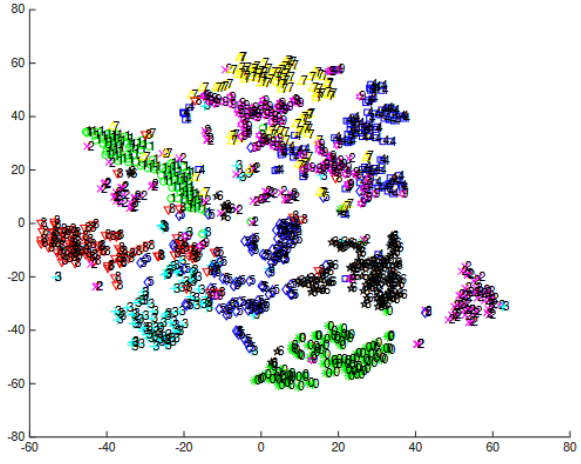

**Figure 2.** Data lying near a parabola

# Today's lecture

1. Introduction
2. Reminders on PCA
3. Stochastic Neighbor Embedding

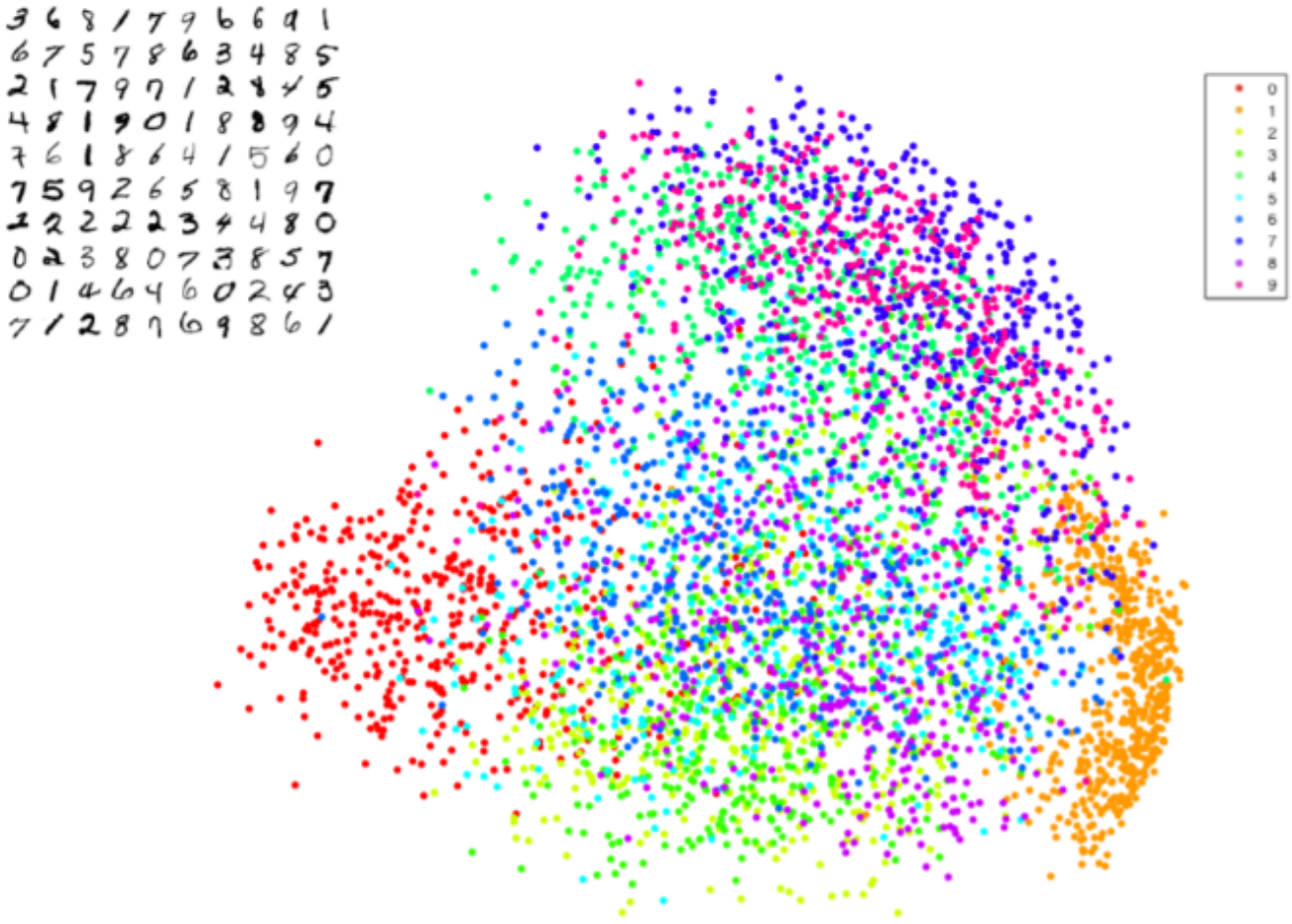
# MNIST dataset

$d = 784$




$q = 2$

# MNIST dataset: PCA results




What can you say about the structure of this data?

# SNE: Principles

Key Idea: Treat pairwise distances as probabilities $\rightarrow$ stochastic

Low distance $\Rightarrow$ High probability

## Principles

- Small distance does not mean proximity on manifold
- *Encode* high dimensional neighborhood information as a distribution
- Picking two similar points is much more probable than picking two points which are well far apart
- Preserve local structure

  $\rightarrow$ Low-dim and original neighborhoods should be the same

# Whiteboard

# SNE: Key steps

1. Transform pairwise distances $\boldsymbol{dist}(\mathbf{x}_i, \mathbf{x}_j)$ into probabilities $P_X(\mathbf{x}_i|\mathbf{x}_j)$
2. Do the same for the projection space $\boldsymbol{dist}(\mathbf{y}_i, \mathbf{y}_j) \rightarrow P_Y(\mathbf{y}_i|\mathbf{y}_j)$
3. Minimize the distance between the distributions $P_X$ and $P_Y \rightarrow$ Optimization problem

SNE

# SNE: Key steps

1. Transform pairwise distances $\boldsymbol{dist}(\mathbf{x}_i, \mathbf{x}_j)$ into probabilities $P_X(\mathbf{x}_i|\mathbf{x}_j)$
2. Do the same for the projection space $\boldsymbol{dist}(\mathbf{y}_i, \mathbf{y}_j) \rightarrow P_Y(\mathbf{y}_i|\mathbf{y}_j)$
3. Minimize the distance between the distributions $P_X$ and $P_Y \rightarrow$ Optimization problem

## SNE

- Compare distributions $\rightarrow$ Kullback-Leibler (KL) divergence

# SNE: Key steps

1. Transform pairwise distances $\boldsymbol{dist}(\mathbf{x}_i, \mathbf{x}_j)$ into probabilities $P_X(\mathbf{x}_i|\mathbf{x}_j)$
2. Do the same for the projection space $\boldsymbol{dist}(\mathbf{y}_i, \mathbf{y}_j) \rightarrow P_Y(\mathbf{y}_i|\mathbf{y}_j)$
3. Minimize the distance between the distributions $P_X$ and $P_Y \rightarrow$ Optimization problem

## SNE

- Compare distributions $\rightarrow$ Kullback-Leibler (KL) divergence
- Model $P_X$ and $P_Y \rightarrow$ Gaussian distributions

# SNE: Key steps

1. Transform pairwise distances $\boldsymbol{dist}(\mathbf{x}_i, \mathbf{x}_j)$ into probabilities $P_X(\mathbf{x}_i|\mathbf{x}_j)$
2. Do the same for the projection space $\boldsymbol{dist}(\mathbf{y}_i, \mathbf{y}_j) \rightarrow P_Y(\mathbf{y}_i|\mathbf{y}_j)$
3. Minimize the distance between the distributions $P_X$ and $P_Y \rightarrow$ Optimization problem

## SNE

- Compare distributions $\rightarrow$ Kullback-Leibler (KL) divergence
- Model $P_X$ and $P_Y \rightarrow$ Gaussian distributions
- Different distribution for each sample! The size of the local neighborhood changes

# Neighborhood Distributions: Gaussian case

# Neighborhood Distributions: Gaussian case

Goal: model the probability that a point $\mathbf{x}_i$ chooses $\mathbf{x}_j$ as its neighbor

- Input space: $P_X(\mathbf{x}_j|\mathbf{x}_i) = p_{j|i} = \frac{\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2\sigma_i^2)}{\sum_{k\neq i}\exp(-\|\mathbf{x}_i - \mathbf{x}_k\|^2/2\sigma_i^2)}$
- Output space: $P_Y(\mathbf{y}_j|\mathbf{y}_i) = q_{j|i} = \frac{\exp(-\|\mathbf{y}_i - \mathbf{y}_j\|^2)}{\sum_{k\neq i}\exp(-\|\mathbf{y}_i - \mathbf{y}_k\|^2)}$
- $p_{i|i} = q_{i|i} = 0$
- Symmetrization $p_{ji} = \frac{1}{2N}(p_{j|i} + p_{i|j})$

# Neighborhood Distributions: Gaussian case

Goal: model the probability that a point $\mathbf{x}_i$ chooses $\mathbf{x}_j$ as its neighbor

- Input space: $P_X(\mathbf{x}_j|\mathbf{x}_i) = p_{j|i} = \frac{\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2\sigma_i^2)}{\sum_{k\neq i}\exp(-\|\mathbf{x}_i - \mathbf{x}_k\|^2/2\sigma_i^2)}$
- Output space: $P_Y(\mathbf{y}_j|\mathbf{y}_i) = q_{j|i} = \frac{\exp(-\|\mathbf{y}_i - \mathbf{y}_j\|^2)}{\sum_{k\neq i}\exp(-\|\mathbf{y}_i - \mathbf{y}_k\|^2)}$
- $p_{i|i} = q_{i|i} = 0$
- Symmetrization $p_{ji} = \frac{1}{2N}(p_{j|i} + p_{i|j})$

How to choose $\sigma_i$ ?

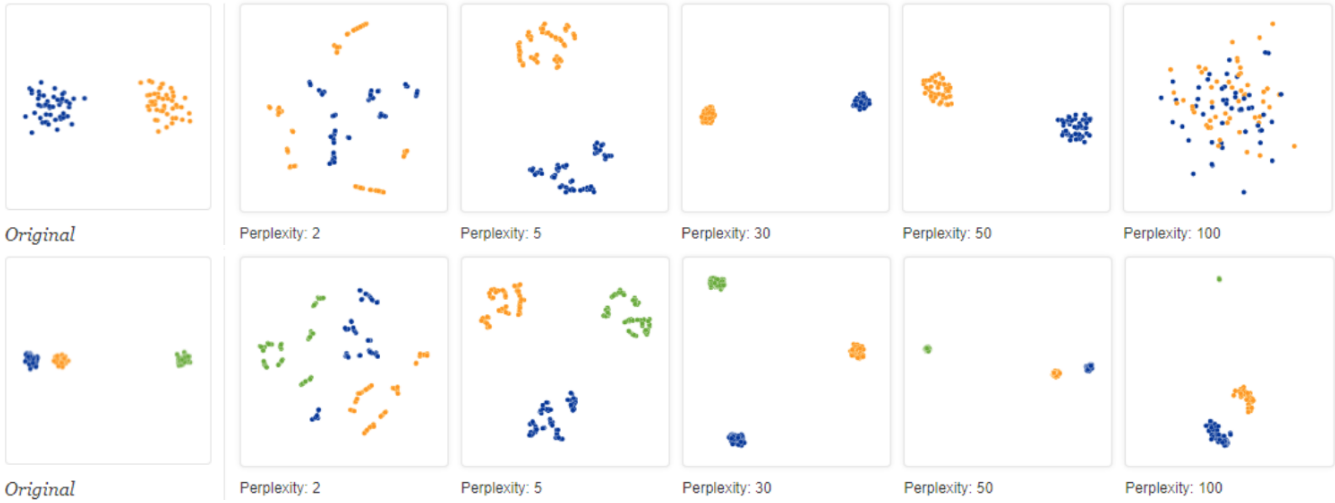
- Different value for each point $i$
- Very low $\sigma_i \rightarrow$ all the probability is in the nearest neighborhood
- Very high $\sigma_i \rightarrow$
- In practice $\rightarrow$ perplexity

# Perplexity

Perplexity is a smooth measure of the number of neighbors

$$Perp(P_i) = 2^{-\sum_j p_{ji} \log_2(p_{ji})}$$

- Uniform $P$ over $k$ elements $\rightarrow$ $Perp(P) = k$
- High/low perplexity $\rightarrow$ Large/small $\sigma_i$
- In practice: values between **5** and **50**




Perplexity should be smaller than N!

# SNE: Optimization Problem

Minimize the KL divergence between $p_{ji}$ and $q_{ji}$

$$\min_{y_1,\dots,y_N} C(X,Y) = \sum_{i,j} KL(P_{ji}|Q_{ji}) = \sum_{i,j} p_{ji} \log \frac{p_{ji}}{q_{ji}}$$

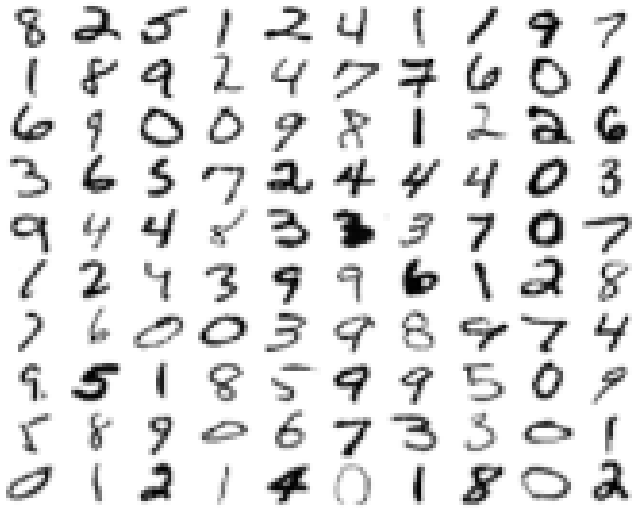
→ One can show that (symmetric case)

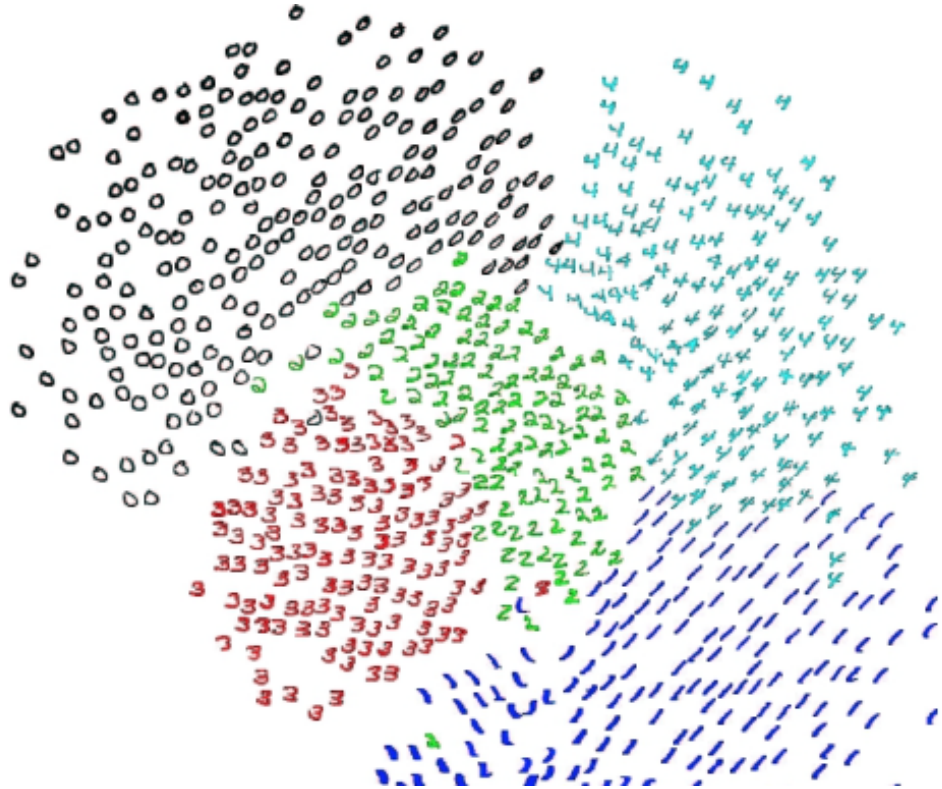
$$\frac{\partial C}{\partial \mathbf{y}_i} = \sum_j (p_{ji} - q_{ji})(\mathbf{y}_i - \mathbf{y}_j)$$

→ Non-convex: no convergence guarantees, multiple restarts

Solved with numerical methods

# SNE: MNIST results





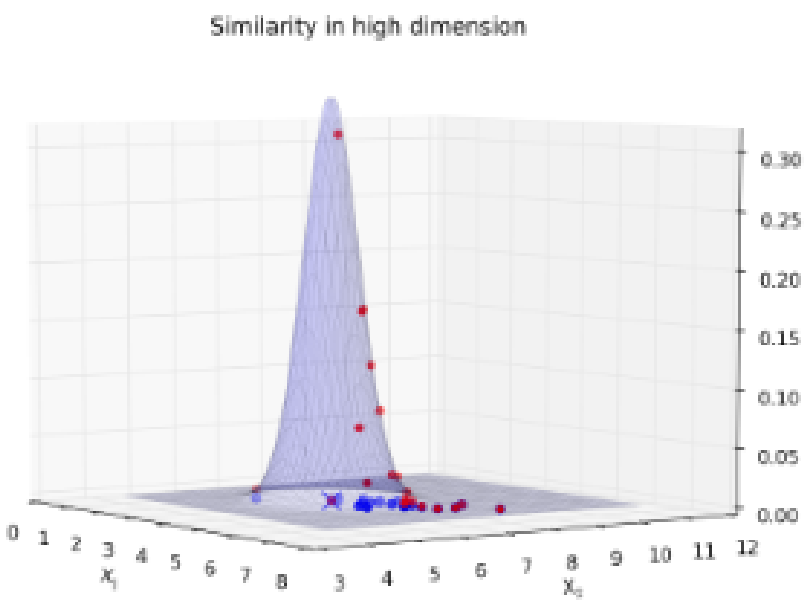
But...

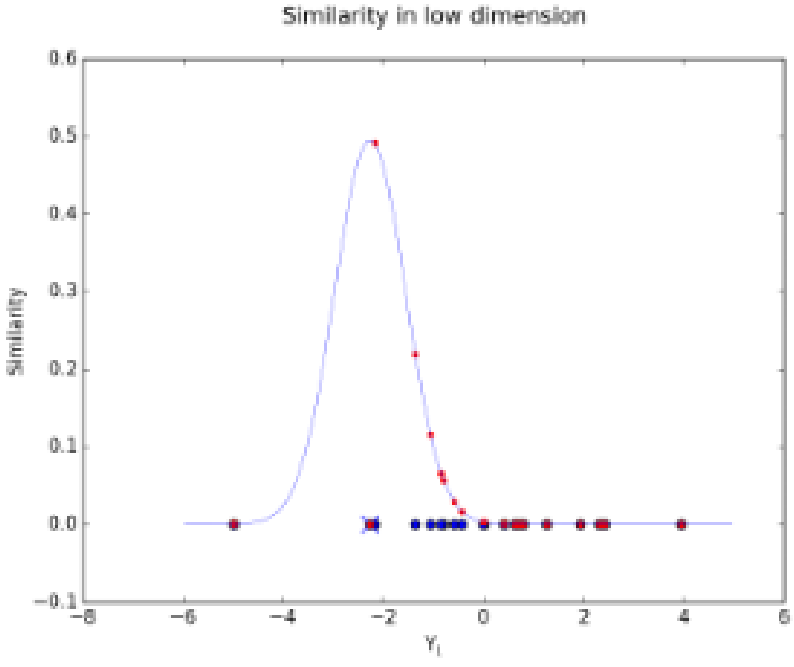
- Impossible to preserve the whole structure due to the distortions introduced by the dimensionality reduction
- Crowding problem!

# SNE: Limitation

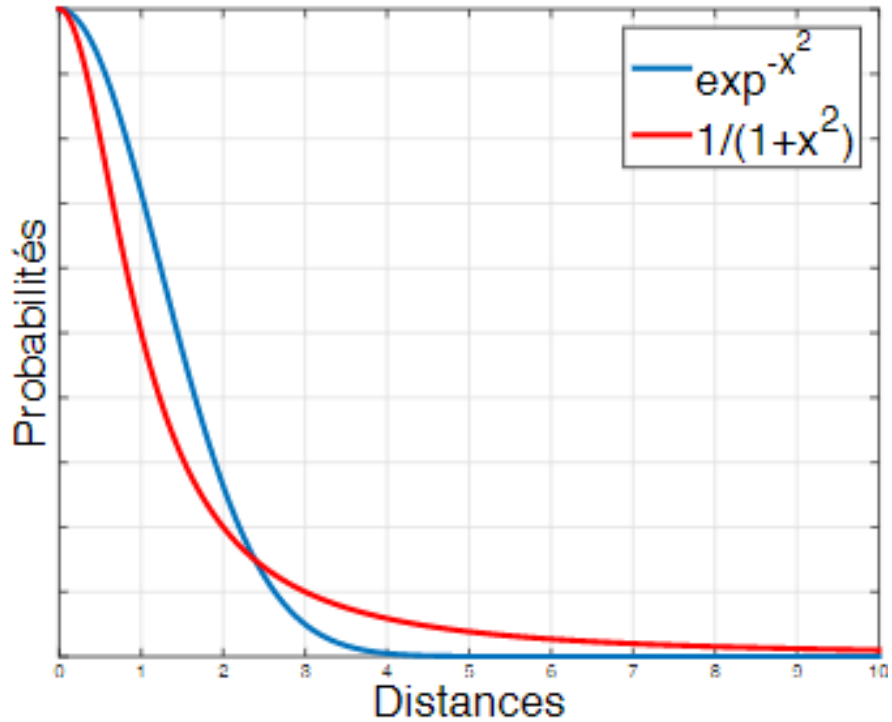
## Crowding problem

- We don't have enough room to accommodate all neighbors
- Example: embed a circular neighborhood from 2D to 1D



# t-SNE: dealing with the crowding problem




## Heavy-tailed q distribution

- Local neighbors are attracted to a slightly closer position
- Non-local neighbors might get a little repulsed
- Non-neighbors are potentially pushed away

# t-SNE principles

Key Idea: model the output space using a heavy-tailed distribution
$\rightarrow$ Student's t-distribution $\nu = 1$

- Input space: $P_X(\mathbf{x}_j|\mathbf{x}_i) = p_{j|i} = \frac{\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2\sigma_i^2)}{\sum_{k\neq i}\exp(-\|\mathbf{x}_i - \mathbf{x}_k\|^2/2\sigma_i^2)}$
- Output space: $P_Y(\mathbf{y}_j|\mathbf{y}_i) = q_{j|i} = \frac{(1+\|\mathbf{y}_i - \mathbf{y}_j\|^2)^{-1}}{\sum_{k\neq i}(1+\|\mathbf{y}_i - \mathbf{y}_k\|^2)^{-1}}$
- $p_{i|i} = q_{i|i} = 0$
- Symmetrization $p_{ji} = \frac{1}{2N}(p_{j|i} + p_{i|j})$

## Attraction/repulsion

- large probability $\rightarrow d_Y < d_X$
- low probability $\rightarrow d_Y > d_X$

# t-SNE: Optimization Problem

Minimize the KL divergence between $p_{ji}$ and $q_{ji}$

$$\min_{y_1,\dots,y_N} C(X,Y) = \sum_{i,j} KL(P_{ji}|Q_{ji}) = \sum_{i,j} p_{ji} \log \frac{p_{ji}}{q_{ji}}$$

$\rightarrow$ Gradient

$$\frac{\partial C}{\partial \mathbf{y}_i} = \sum_j (p_{ji} - q_{ji})(\mathbf{y}_i - \mathbf{y}_j)(1 + \|\mathbf{y}_i - \mathbf{y}_j\|^2)^{-1}$$

$\rightarrow$ Non-convex: no convergence guarantees, multiple restarts

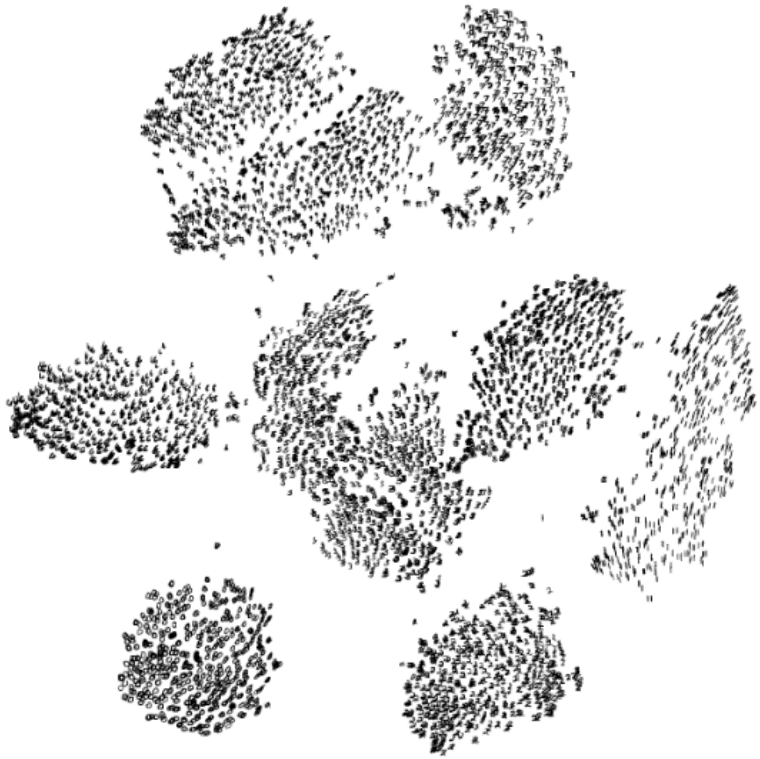
Solved with gradient descent

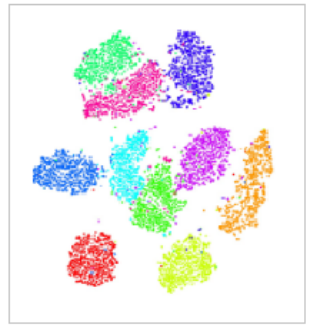
# t-SNE: Algorithm

**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 $\eta$, 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**

# t-SNE: MNIST results





But...

- Different perplexity $\rightarrow$ different results
- Different results with different runs

# PCA vs t-SNE

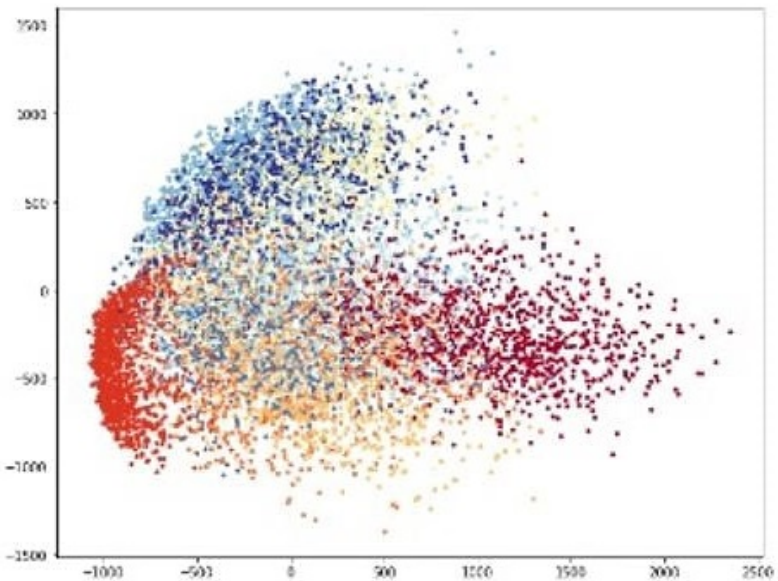
## PCA

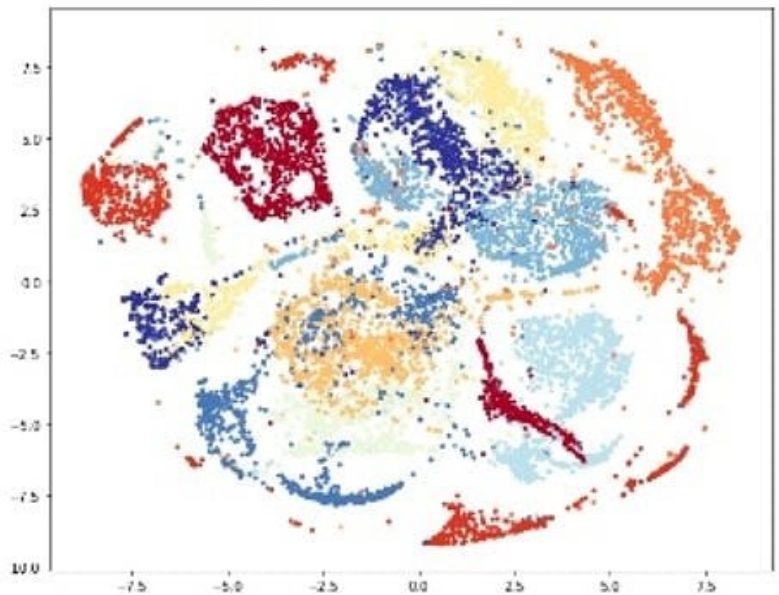
- Finds global structure
- Low-dim subspace
  - Inconsistencies: Far away points can become nearest neighbors
  - Doesn't always reveal hidden/complex structure (linear)
  + Relatively fast
  + Features can be traced back

## t-SNE

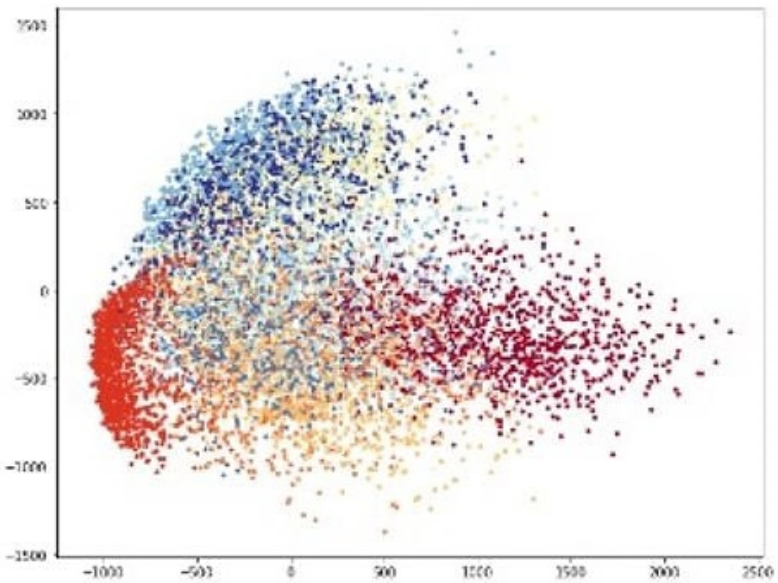
- Preserves local structure
- Low and high-dim neighborhoods should be the same
- Used for visualization
  + Reveals hidden structure
  - Relatively slow ($O(N^2)$)
  - Meaning of the features in lost
  - Cannot project a new point onto an already computed map (+ variations exist)

# Example

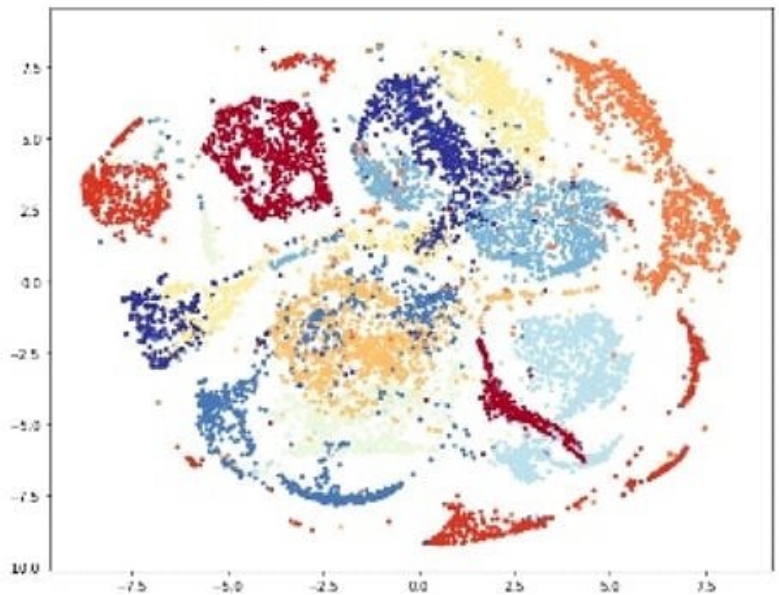

1500
1000
500
0
−500
−1000
−1500
−1000
−500
0
500
1000
1500
2000
2500


7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5

# Example



PCA



t-SNE