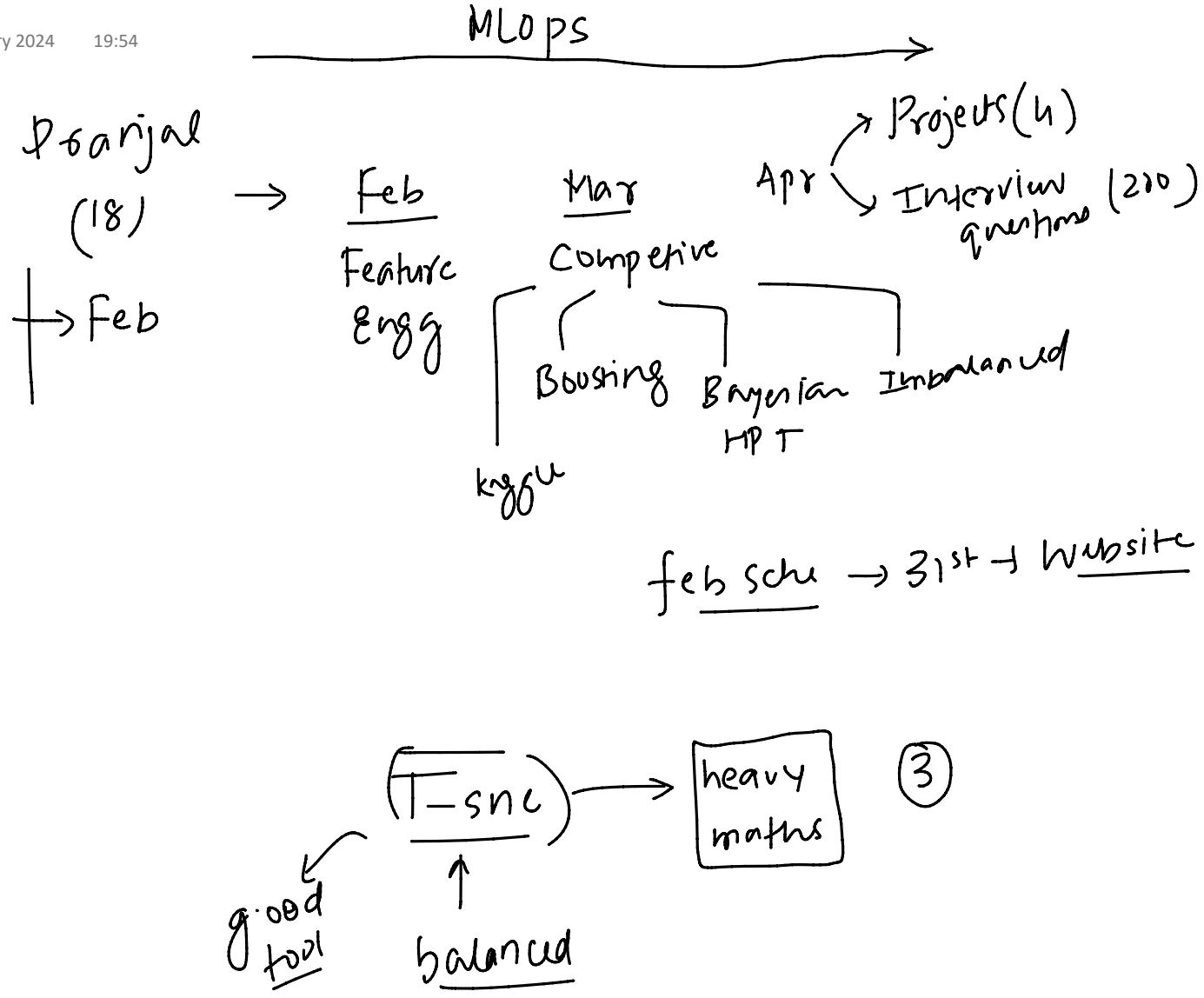
30 January 2024 19:54



feb sche → 31st → website

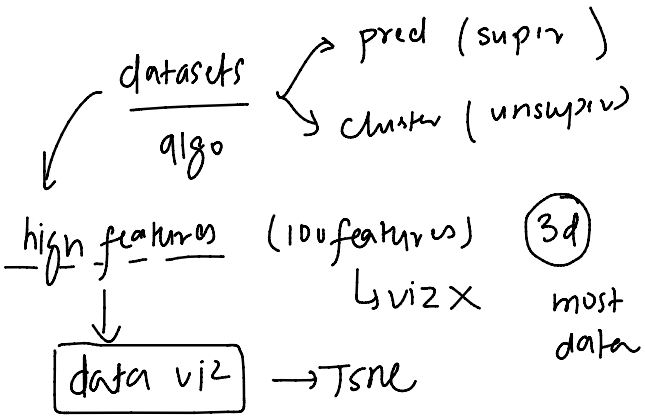
What is T-SNE

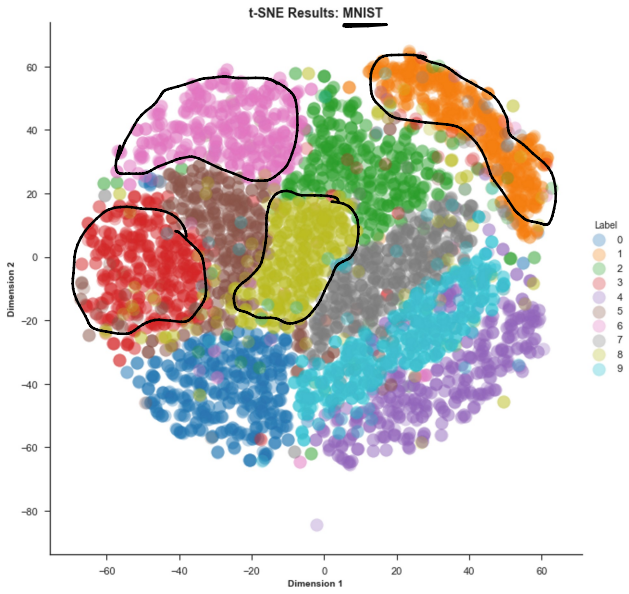
29 January 2024 18:03

100d → 3d
↘ 2d

784 dim

t-SNE, or t-Distributed Stochastic Neighbour Embedding is a statistical method for visualizing high-dimensional data by reducing it to lower-dimensional spaces, typically two or three dimensions. This makes it easier to visualize and interpret the data, especially when dealing with complex datasets like those in machine learning and data science.



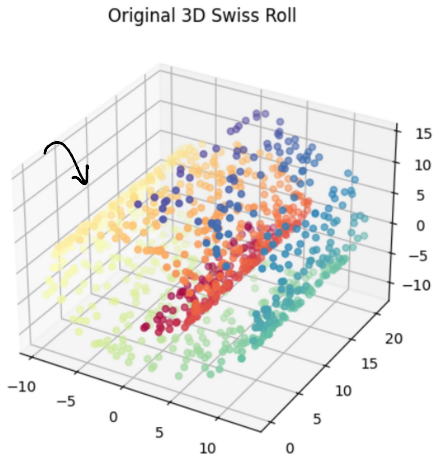


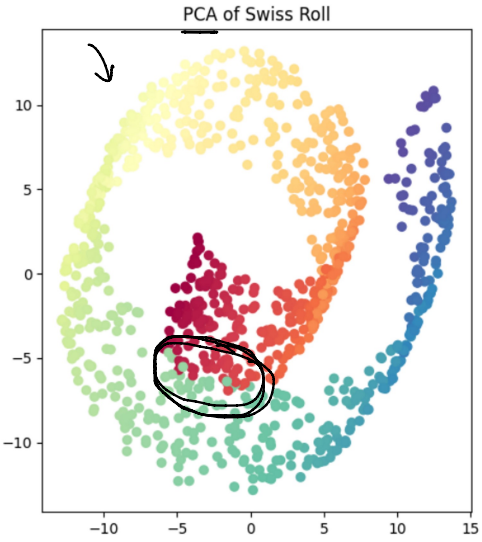
Why learn T-SNE

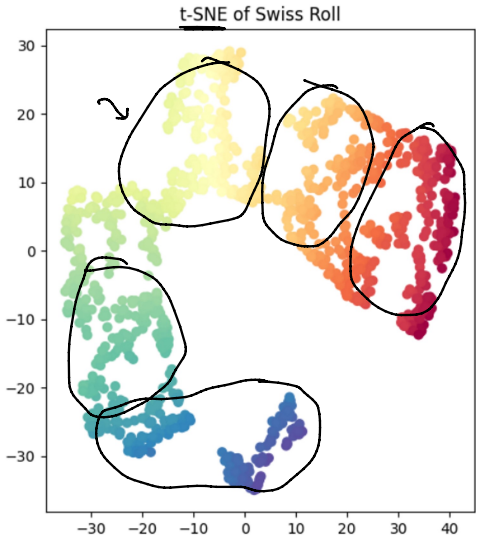
29 January 2024 18:05

PCA → feature extraction → 100f → 2f / 3f
↓ plot
tSNE

1) Linear datasets



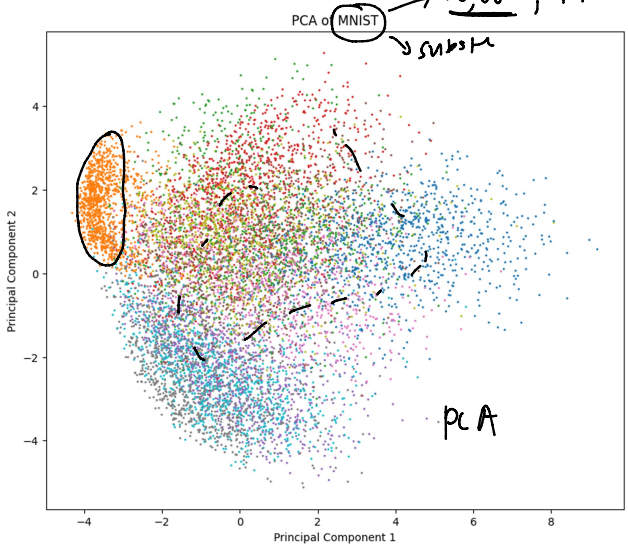




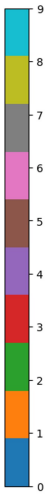
swiss roll

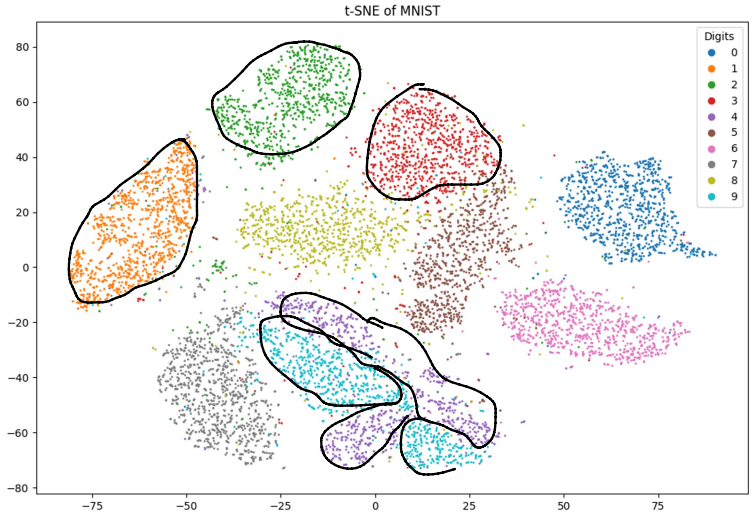
easy to interpret → tsne is better viz

PCA of MNIST → 10,000 points
↓ subset



PCA





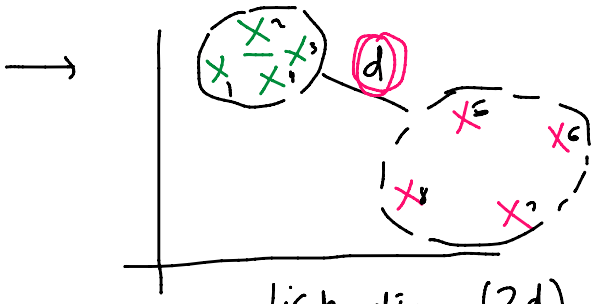
Geometric Intuition

29 January 2024 18:05

2008

→ Simplify things

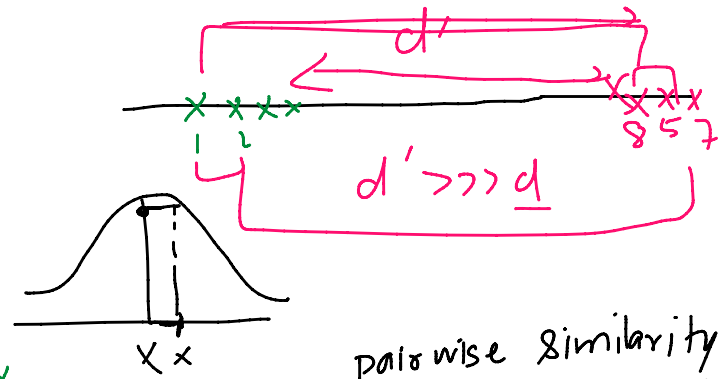
high → 2d low dim → 1d

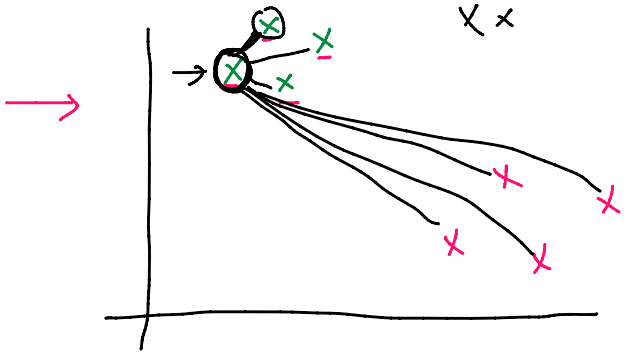


high dim (2d) → 1d plot

tsne preserve the local structure of data
global structure

1-2 - close
1-8 - far
8-5 → close
7-2 far





pairwise similarity of all the points

similarity → distance → similarity

high dimension → distance reliable measure

Curse of dimensionality

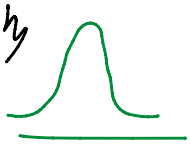
6 similarity

X_1	X_2	X_3

$X_{1\to2}$	$X_{2\to1}$
X_{2-3}	$X_{3\to2}$
$X_{1\to3}$	X_{3-1}

NO

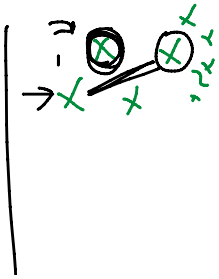
1-2 / 2→1

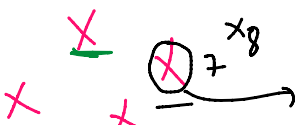


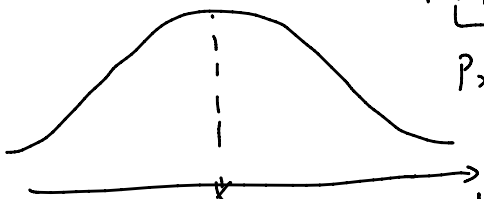
	X_1	X_2	X_3	...	X_7	X_8
X_1	0	0.3	0.1		0.03	0.04
X_2	0.4	0	- - -		- -	- -
X_3						
⋮						
X_8						

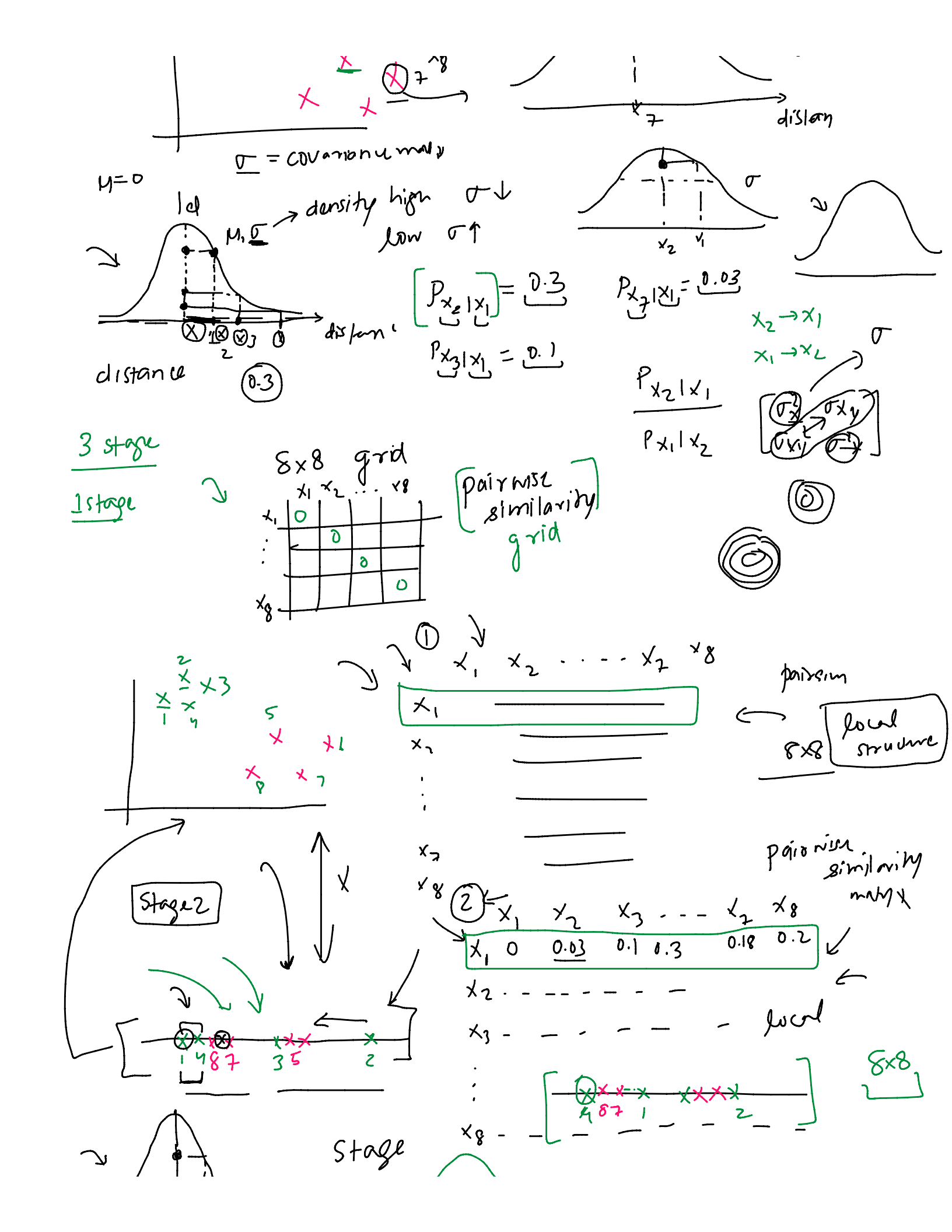
$P_{X_1|X_2} = 0.4$

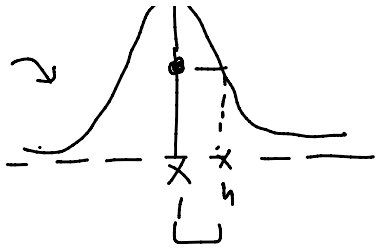
$P_{X_3|X_2} =$



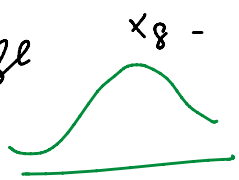





distance
density high σ↓
low σ↑
μ, σ
0.3
$P_{x_2|x_1} = 0.3$
$P_{x_7|x_1} = 0.03$
$P_{x_3|x_1} = 0.1$
$P_{x_2|x_1}$
$P_{x_1|x_2}$
$x_2 \to x_1$
$x_1 \to x_2$
3 stage
1 stage
8×8 grid
pairwise similarity grid
Stage2
Stage
pairwise similarity matrix
local structure
8×8



Stage



+ve -ve

Stage 3

8×8 table
high dim

	x_1	x_2	x_3	...	x_8
x_1	0	0.2	0.3	...	0.03

far

grid

x_2	-	-	0.2	=

0.198

8×8 table

dimension

	x_1	x_2	x_2	...	x_8
x_1	0	0.02	0.4	...	0.4

→ 1.3

→ net

nesh

far

x_2	-	-	-	-	-	-

0.03 - 0.4

= -0.397

Mathematical Formulation

29 January 2024 18:05

1. **High-Dimensional Similarities**:
   - Given a set of N points in a high-dimensional space, $\{x_1, x_2, ..., x_N\}$, the similarity of datapoint x_j to datapoint x_i is represented as a conditional probability $P_{j|i}$, which is the probability that x_i would pick x_j as its neighbor.
   - This probability is given by the Gaussian distribution centered on x_i:
   $$P_{j|i} = \frac{\exp(-||x_i - x_j||^2 / \sigma_i^2)}{\sum_{k \neq i} \exp(-||x_i - x_k||^2 / 2\sigma_i^2)}$$
   - Here, $||x_i - x_j||$ is the Euclidean distance between x_i and x_j, and σ_i is the variance of the Gaussian that is centered on datapoint x_i. The value of σ_i is chosen such that the perplexity of the conditional distribution equals a predefined perplexity.

The probabilities are symmetrized using:

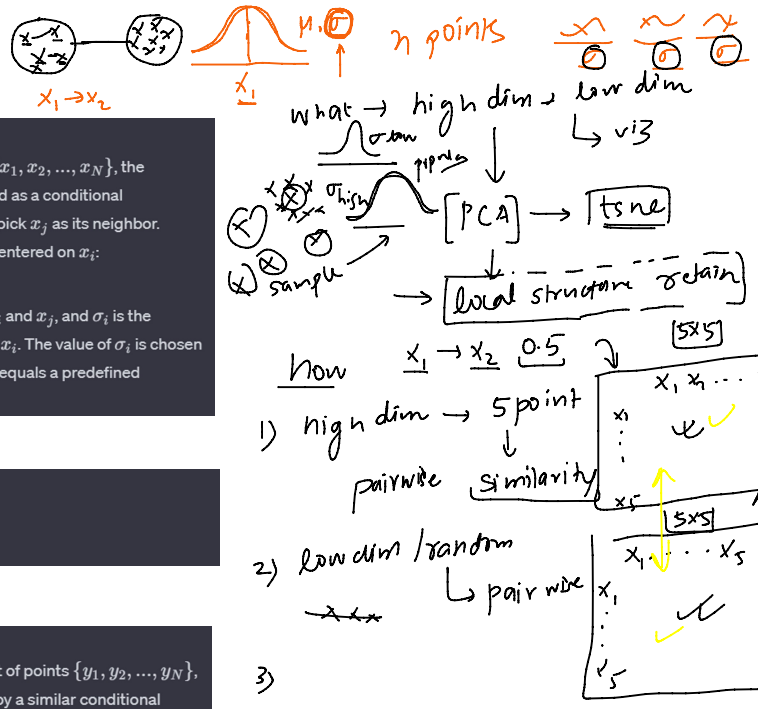
$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}$$

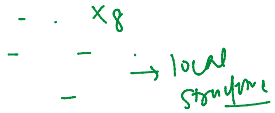
2. **Low-Dimensional Similarities**:
   - In the low-dimensional space, for a corresponding set of points $\{y_1, y_2, ..., y_N\}$, the similarity of datapoint y_j to datapoint y_i is given by a similar conditional probability, but using a Student's t-distribution (with one degree of freedom, equivalent to the Cauchy distribution) instead of the Gaussian distribution:
   $$Q_{j|i} = \frac{(1+||y_i - y_j||^2)^{-1}}{\sum_{k \neq i}(1+||y_i - y_k||^2)^{-1}}$$ — t-dist
   - This heavy-tailed distribution in the low-dimensional space is what helps t-SNE to alleviate the crowding problem.

Prob dist

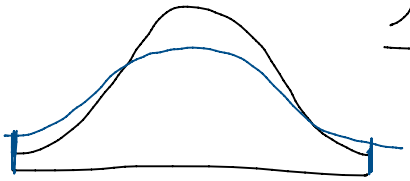
3. **Cost Function (Kullback-Leibler Divergence)**: → w.r.t y_i
   - The t-SNE algorithm aims to minimize the difference between these two probability distributions P and Q, which is quantified by the Kullback-Leibler (KL) divergence:
   $$C = KL(P||Q) = \sum_i \sum_j P_{j|i} \log \frac{P_{j|i}}{Q_{j|i}}$$ high dim grid 0, low dim grid 0
   - This is a measure of how one probability distribution diverges from a second, expected probability distribution.

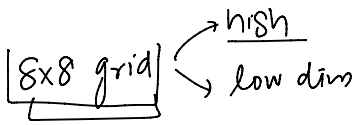
4. **Optimization**:
   - The positions of points y_i in the low-dimensional space are optimized (usually through gradient descent) to minimize this KL divergence.
   - The gradient of the KL divergence with respect to the point y_i can be computed, and this gradient is used to update the positions of the points in the low-dimensional map.

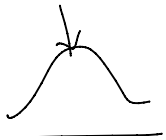


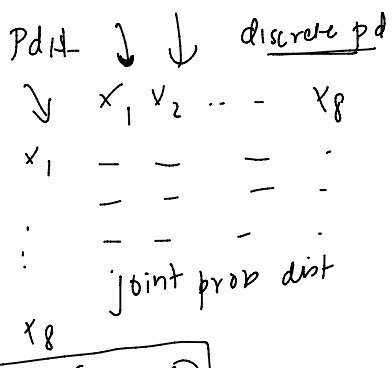


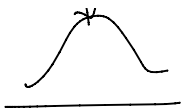
diff betn 2 prob distributn





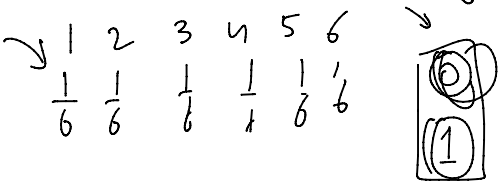


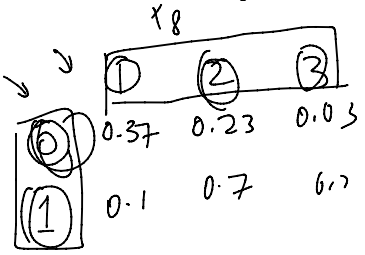




prob distribution

joint prob ...





KL divergence

JPD

X_1 ---- X_9

X_1

X_8

for every σ_1 σ_2 ... σ_n

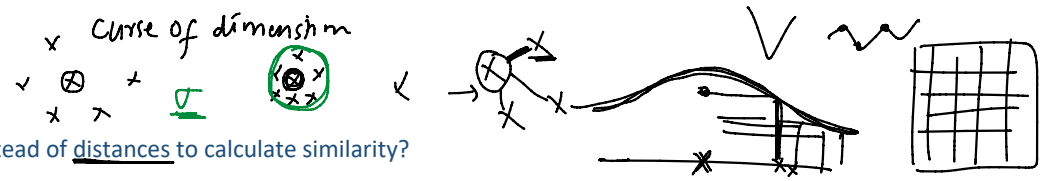
σ_i

→ [perplexity] → 4, 5, 50

↳ how many neighbors surround each point

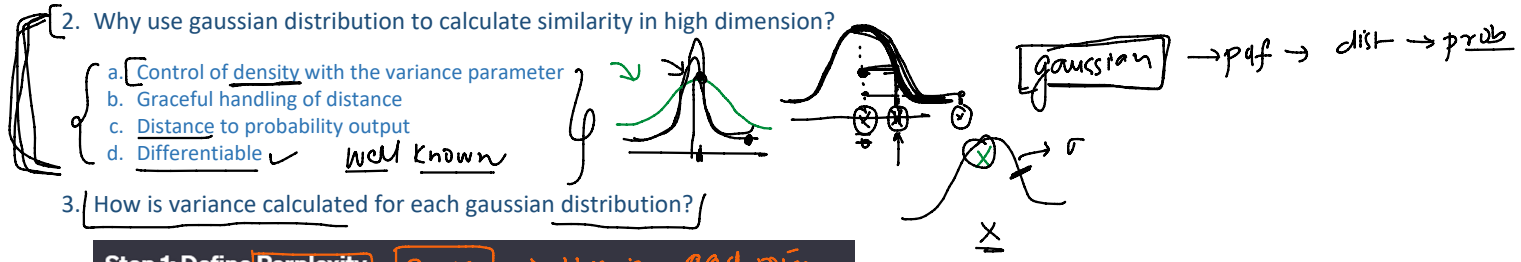
Some Questions!

29 January 2024 18:26



1. Why use probabilities instead of distances to calculate similarity?

2. Why use gaussian distribution to calculate similarity in high dimension?
   a. Control of density with the variance parameter
   b. Graceful handling of distance
   c. Distance to probability output
   d. Differentiable ✓ well known



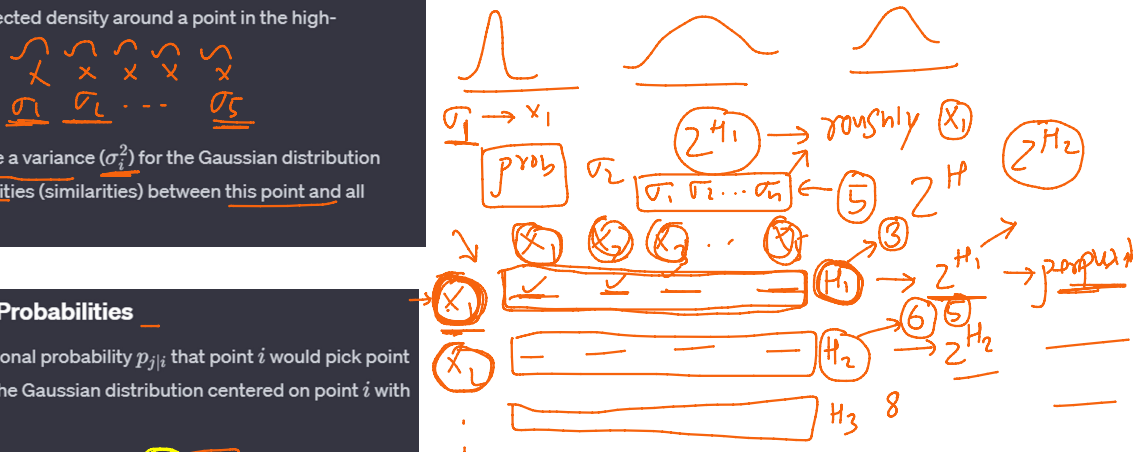
3. How is variance calculated for each gaussian distribution?

Step 1: Define Perplexity $p = 5$ → # neighbors

- **Perplexity** is a measure set by the user that indirectly controls the number of effective nearest neighbors. It reflects the expected density around a point in the high-dimensional space.

Step 2: Initialize Variances

- For each point in the dataset, initialize a variance (σ_i^2) for the Gaussian distribution that will be used to calculate probabilities (similarities) between this point and all other points.



Step 3: Calculate Conditional Probabilities

- For each point i, calculate the conditional probability $p_{j|i}$ that point i would pick point j as its neighbor. This is done using the Gaussian distribution centered on point i with variance σ_i^2:

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

- These probabilities are normalized so that they sum to 1 for each point i.

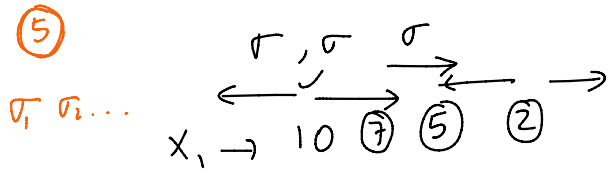
Step 4: Calculate Shannon Entropy and Perplexity

- Compute the Shannon entropy $H(P_i)$ of the conditional probability distribution for each point i:

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

gini entropy → → decision

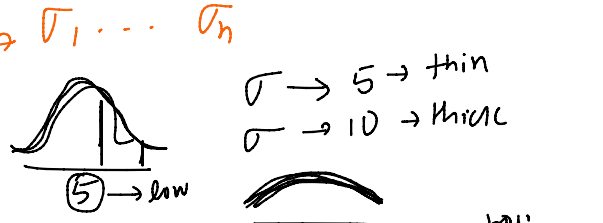
- The perplexity is then calculated as $2^{H(P_i)}$ which represents the effective number of neighbors around point i.



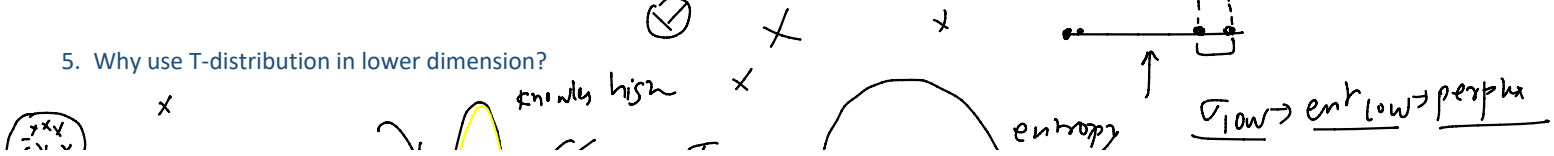
Step 5: Adjust Variance to Match User-Specified Perplexity

- For each point, adjust the variance σ_i^2 so that the calculated perplexity from the conditional probabilities matches the user-specified perplexity. This involves:
  - **Binary Search**: Iteratively adjust σ_i^2 through binary search. If the calculated perplexity is too high (indicating too broad a distribution), decrease σ_i^2; if too low (too narrow a distribution), increase σ_i^2.
  - **Convergence**: Continue adjusting until the calculated perplexity closely matches the user-specified perplexity. This ensures that the local structure around each point is consistent with the user's expectations.

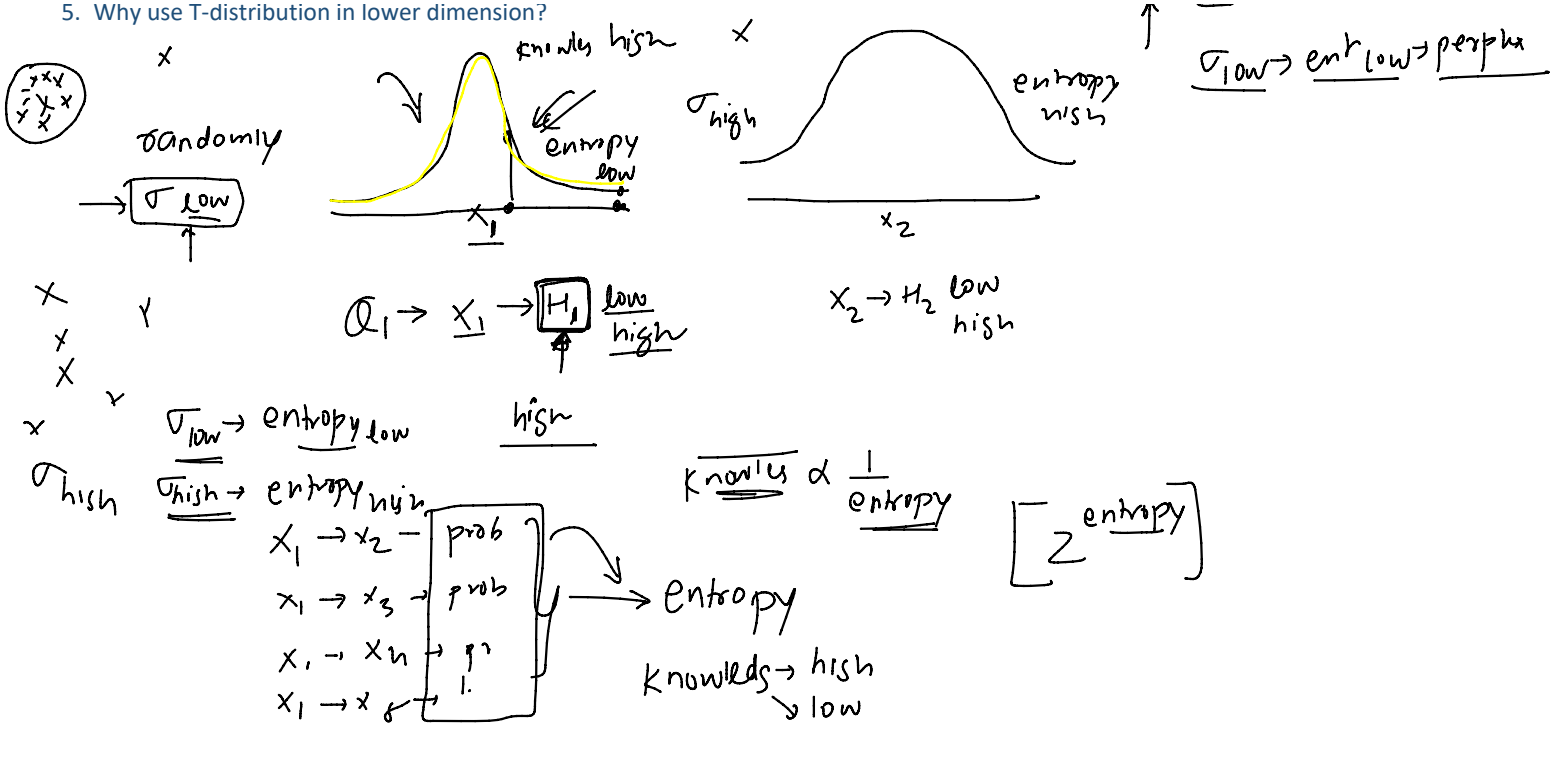
4.

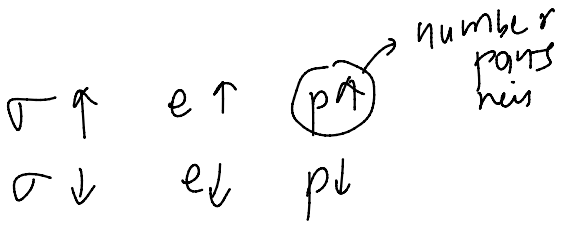


5. Why use T-distribution in lower dimension?



5. Why use T-distribution in lower dimension?





Crowding Problem

30 January 2024 10:43

Code Example

29 January 2024 18:05

Hyperparameters

29 January 2024 18:11

5 ↔ 250

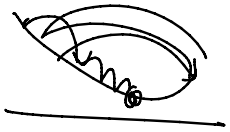
1. Perplexity:

- Perplexity is perhaps the most important hyperparameter in t-SNE. It can be thought of as a measure of the effective number of neighbours for each point.
- The value of perplexity affects the balance between local and global aspects of your data. A small perplexity emphasizes local structure, while a larger perplexity brings more of the global structure into play.
- Typical values for perplexity range between 5 and 50, but this can vary depending on the dataset. It's often recommended to experiment with different values to see how they affect the results.

2. Learning Rate:

gd

$param_{new} = param_{old} - deriv\ \eta$



low → slow → optimum
fast → jump

- The learning rate determines the step size at each iteration while moving toward a minimum of the cost function.
- A too high learning rate might cause the algorithm to oscillate and miss the global minimum, while a too low learning rate can result in a long training process that might get stuck in a local minimum.
- Common values for the learning rate are between 10 and 1000. Again, experimenting with different values is key to finding the best setting for a given dataset.

3. Number of Iterations:

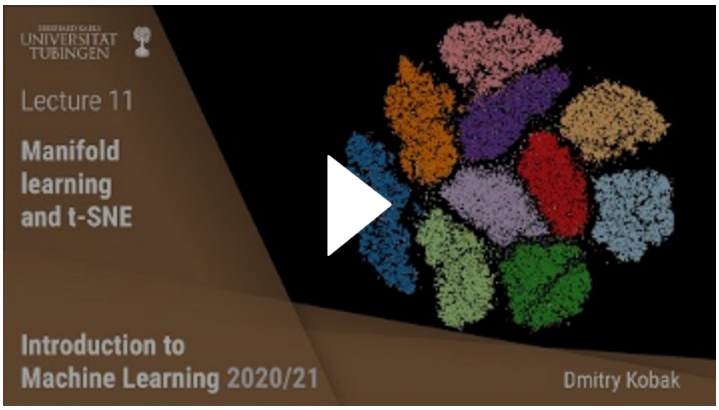
- This hyperparameter controls how many iterations the algorithm runs before it terminates.
- If the number is too low, the algorithm might not fully converge. If it's too high, you might waste computational resources without gaining much in terms of the quality of the embedding.
- The default number of iterations is often set to a value like 1000, but this might need to be increased for larger datasets.

50 / 100

Resources

29 January 2024 18:07

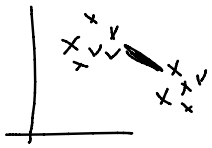
1. Research Paper - https://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf
2. Blog 1 - https://distill.pub/2016/misread-tsne/
3. Blog 2 - https://colah.github.io/posts/2014-10-Visualizing-MNIST/
4. A good lecture - Introduction to Machine Learning - 11 - Manifold learning and t-SNE

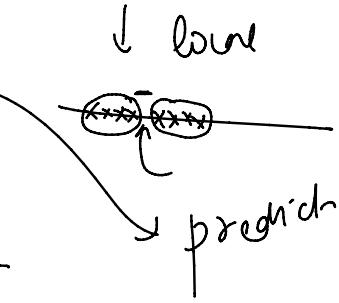


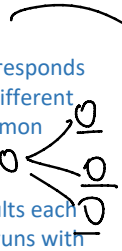
Points of Wisdom

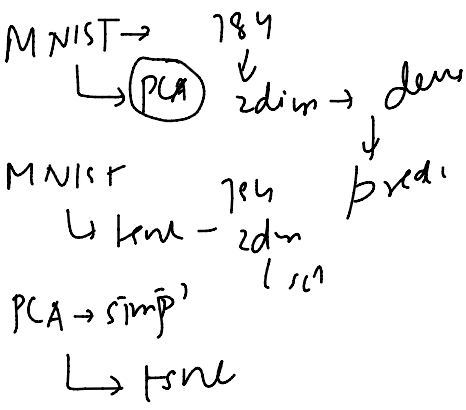
29 January 2024 18:06

1. Interpreting Clusters: t-SNE can reveal clusters and local structures very effectively. However, the distance between clusters or the relative position of clusters in the plot may not have a meaningful interpretation. Avoid over-interpreting global relationships.
2. Axes have no meaning: Axes in T-sne has no interpretable meaning.
3. Perplexity Matters: Perplexity is a crucial hyperparameter in t-SNE. It roughly corresponds to the number of effective nearest Neighbors. There's no one-size-fits-all value; different values can reveal different structures, so experiment with a range of values. Common values are between 5 and 50.
4. Reproducibility: t-SNE starts with a random initialization, leading to different results each time you run it. If reproducibility is important, set a random seed. Also, multiple runs with different initializations can give a fuller picture of your data's structure.
5. Scaling the Data: Pre-processing steps like scaling or normalizing your data, especially if features are on different scales, can have a significant impact on the results of t-SNE.
6. Curse of Dimensionality: t-SNE can mitigate but not completely overcome the curse of dimensionality. Very high-dimensional data might require other steps, like initial dimensionality reduction with PCA, before applying t-SNE.
7. Learning Rate and Number of Iterations: Beyond perplexity, other parameters like the learning rate and the number of iterations also impact the results. A learning rate that's too high or too low can lead to poor embeddings, and insufficient iterations might mean the algorithm doesn't fully converge. 1000
8. It's Not a Silver Bullet: While t-SNE is a powerful tool, it's not suitable for every kind of dataset or analysis. Sometimes other dimensionality reduction techniques like PCA, UMAP, or MDS might be more appropriate.









Advantages & Disadvantages

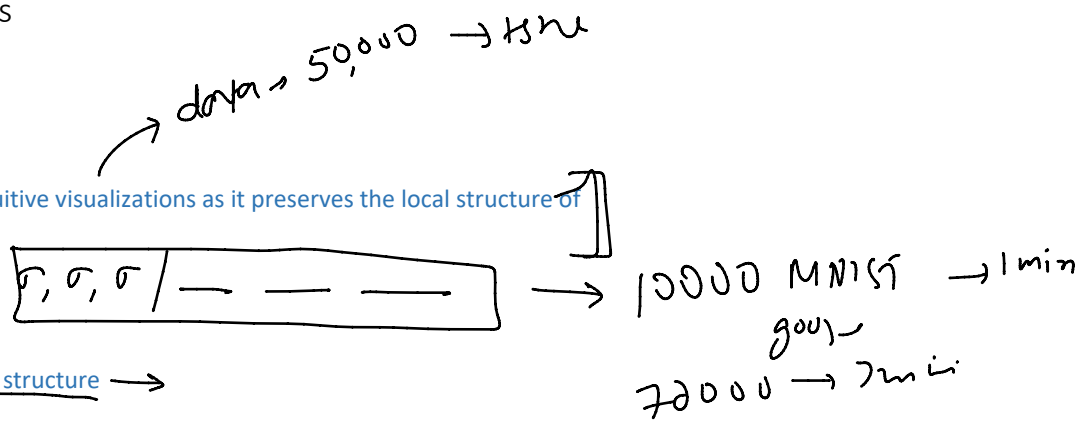
29 January 2024 18:05

Advantages

- If done correctly, can give very intuitive visualizations as it preserves the local structure of the data in the lower dimensions

Disadvantages

- Computationally Expensive
- Not very good at preserving global structure
- Sensitive to hyperparameters
- Can get stuck in local minima
- Interpretation is challenging



02 February 2024 19:22