t-distributed Stochastic Neighbour Embedding (t-SNE), What Is It?

t-SNE makes use of probabilities to calculate two properties of the original dataset, known as the similarity score and perplexity. The similarity score is a normalised value that represents how likely each point - which we will call point B for simplicity- is to be chosen as the nearest neighbour for the point being analysed - henceforth referred to as point A. This is calculated using the distance from point A to point B. Using the position of point A as the centre for the t-distribution curve (which is a lot like a normal distribution, or a gaussian bell curve), with the distance from point A determining the x value under the t-distribution. The corresponding y value on this distribution is then used to calculate the final similarity score of point B relative to point A.

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

Fig 1, equation for finding the similarity score

Taking the y-value for all points B under the distribution for point A, we then normalise all the values, by taking each likelihood to be nearest neighbour and dividing by the likelihoods for each point B in the distribution. This final similarity score will land on a range from 0-1. The perplexity, simply summarised, roughly represents the balance, or difference in strength of effect, between local and global features in the data. This loosely can be likened to a rough estimate of the denseness of the data clusters in our dataset. According to [3] "Perplexity [as defined] by Van der Maaten & Hinton can be interpreted as a smooth measure of the effective number of neighbours." It is important to note that a perplexity score beyond the range of 5-50, as proposed by the authors of the original paper behind t-SNE, can lead to unexpected behaviours in the algorithm.

$$Perp(P_i) = 2^{-\sum p_{j|i} log_2 p_{j|i}}$$

Fig 2. Equation of perplexity, where the exponent represents 'Shannon Entropy'

Once the perplexity score is chosen, and all the similarity scores are chosen, we can create a vector with the properties we need to begin reducing the dimensionality of our data.

The t-SNE algorithm makes use of the Gradient Descent technique in order to find a lower dimensional data set with similar properties as our original data. As a part of gradient descent we need to initialise our reduced dimensionality data with some positions. However, trying to guess where best to start our descent with little other information is inadvisable. Hence, we will initialise our data at random. Once we have our randomly set data, we can calculate all of the similarity scores and the perplexity of our randomly generated sequence. Then, by calculating the loss function, or how different the properties of our new data are from the original, we can create a gradient vector that can be used to follow along and better approximate the original's properties.

$$C = D_{\mathrm{KL}}(P \parallel Q) = \sum_{x \in \mathcal{X}} P(x) \log \left(\frac{P(x)}{Q(x)} \right)$$

Fig 3. Equation of Kullback-Leibler divergence loss function

The gradient vector is used in order to figure out how we can change our data to better approximate our desired properties. We shift the data according to the gradient we calculated, a distance based on steps of a set rate, commonly known as the learning rate, and repeat this process.

$$\frac{\delta C}{\delta y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j)(1 + \|y_i - y_j\|^2)^{-1}$$

Fig 4. The derivative of the loss function used to build the gradient

This process is gradient descent, and we continue to 'descend the gradient,' so to speak, until the difference between the steps is too small to make any significant changes in our perplexity and similarity score values of our dimensionally reduced representation of our original data.

Non-Linear:

- A lot of dimensionality reduction approaches are linear, which simply means that they learn linear projections of data from a high-dimensional space to a lower-dimensional one. A good illustration of this is Principal Component Analysis (PCA).

- t-SNE, in contrast, uses a non-linear approach. This means that it can describe more complicated, non-linear interactions in addition to only capturing linear relationships or correlations across dimensions.

- As a result, linear approaches could miss complex data structures that t-SNE might disclose. If you have, for instance, non-linearly structured data clusters in high-dimensional

space (such as concentric circles), t-SNE may be a better option than linear approaches for visualising this structure.

No guarantees:

- A t-SNE plot could lead someone to believe that the separation between clusters or the exact placement of clusters has a clear, comprehensible purpose. This is not the case, though.

- Points that are near together in the original high-dimensional space should also be close together in the lower-dimensional representation since the primary goal of t-SNE is to preserve local structures.

- While groups that are distant from one another on an t-SNE plot are probably distinct, you cannot draw conclusions about the "amount" of difference purely based on their geographical separation.

Computational Intensity:

Given the temporal complexity of the original t-SNE technique, very big datasets cannot be processed using it. Every time the method iterates, each point must be compared to every other point, which might be computationally costly.

Several improvements have been made in response to this.

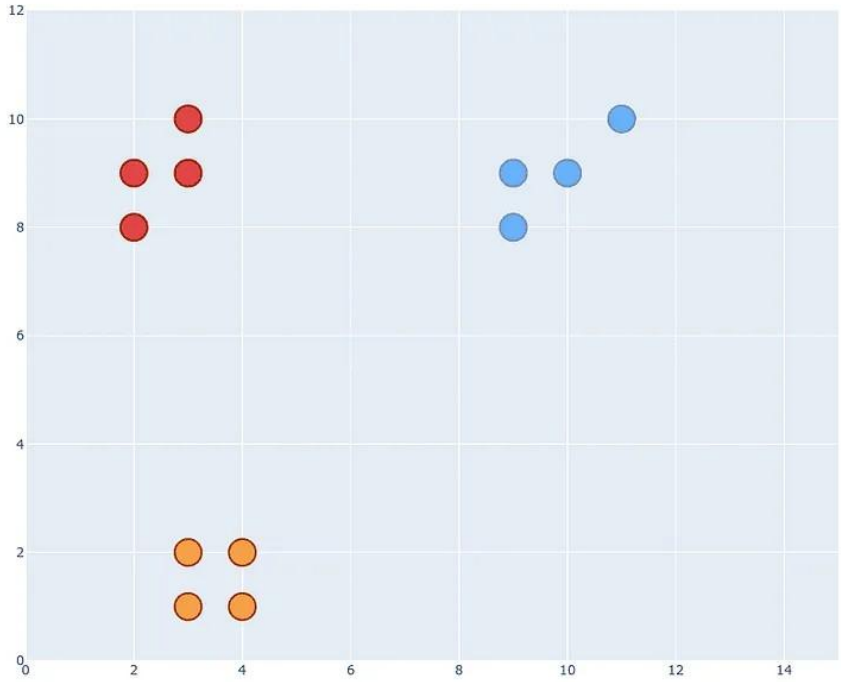
- Multiscale t-SNE: By combining neighboring points into groups and optimizing these groupings as single entities, this technique accelerates t-SNE.

- LargeVis: This method of viewing huge datasets is another one. While using a different algorithmic strategy and being more scalable, it has the same goal of maintaining local structure as t-SNE.

How t-SNE works?

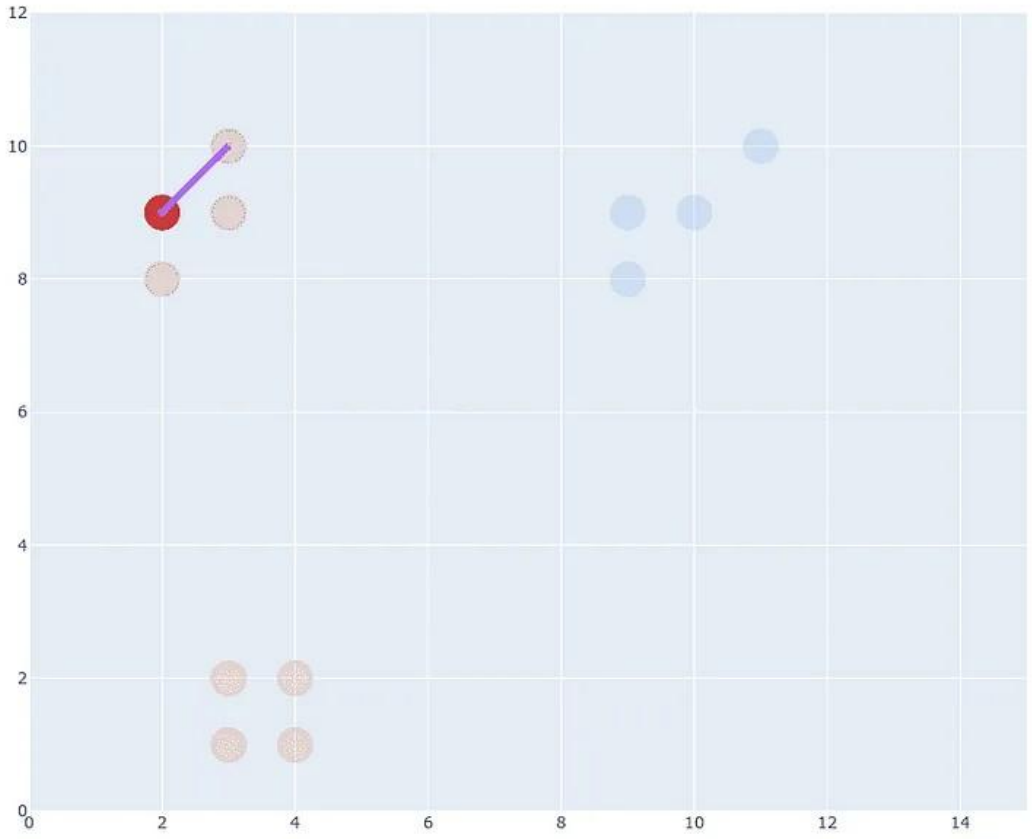
Probability Distribution

Let’s start with SNE part of t-SNE. I’m far better with explaining things visually so this is going to be our dataset:

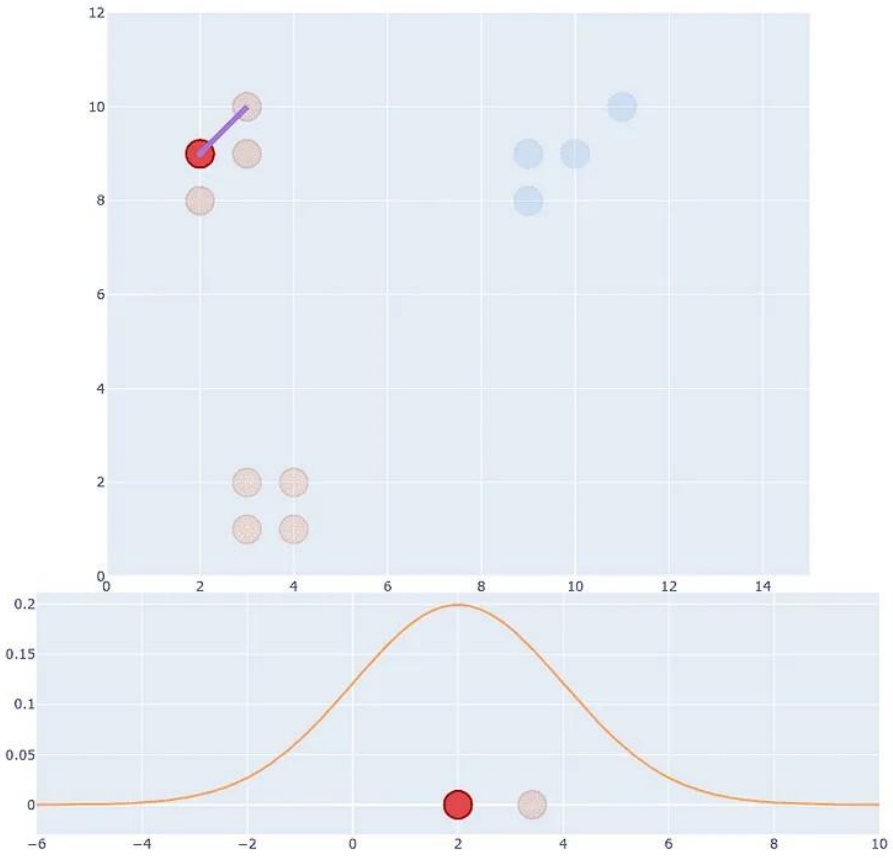


It has 3 different classes and you can easily distinguish them from each other. The first part of the algorithm is to create a probability distribution that represents similarities between neighbors. What is “similarity”? Original paper states “ similarity of datapoint x to datapoint x_i is the conditional probability p_{j|i}, that x_i would pick x as its neighbor “.

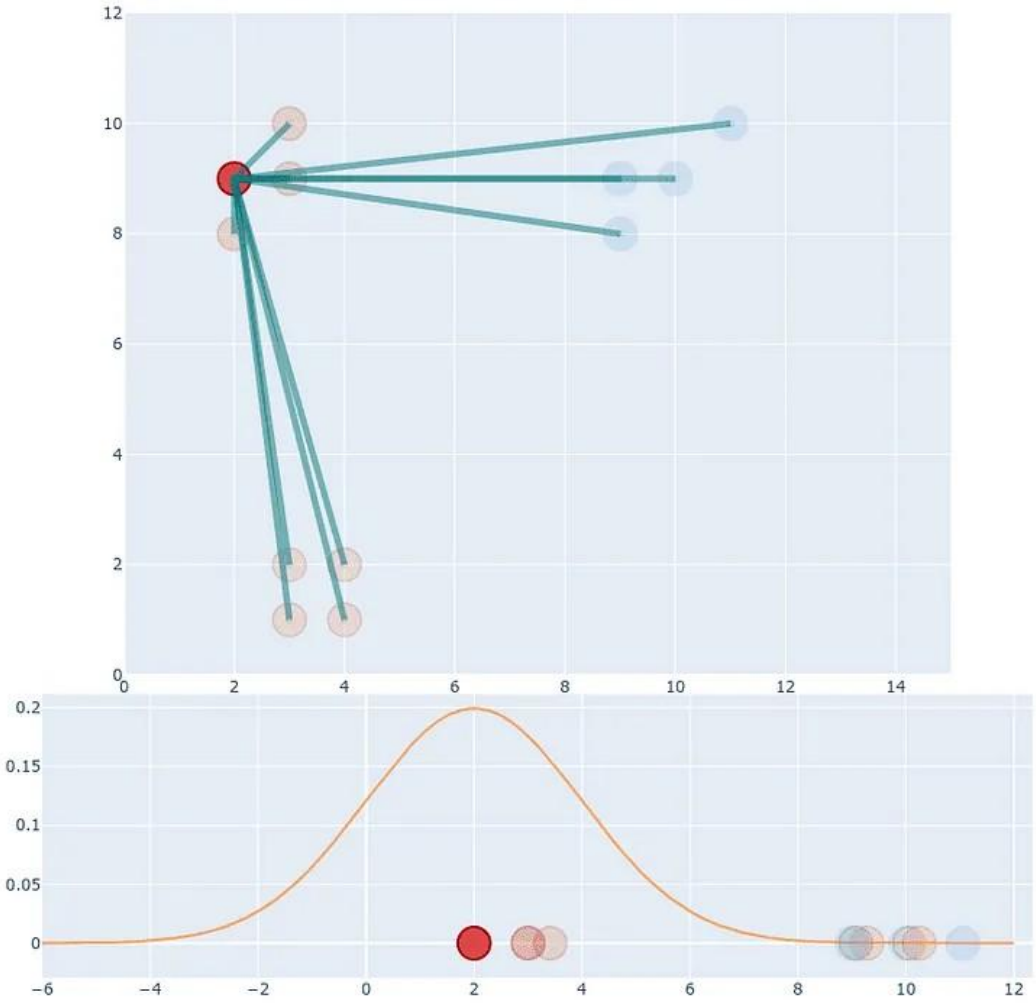
We've picked one of the points from the dataset. Now we have to pick another point and calculate Euclidean Distance between them $|x_i - x$□$|$



The next part of the original paper states that it has to be proportional to probability density under a Gaussian centered at x_i. So we have to generate Gaussian distribution with mean at x_i, and place our distance on the X-axis.



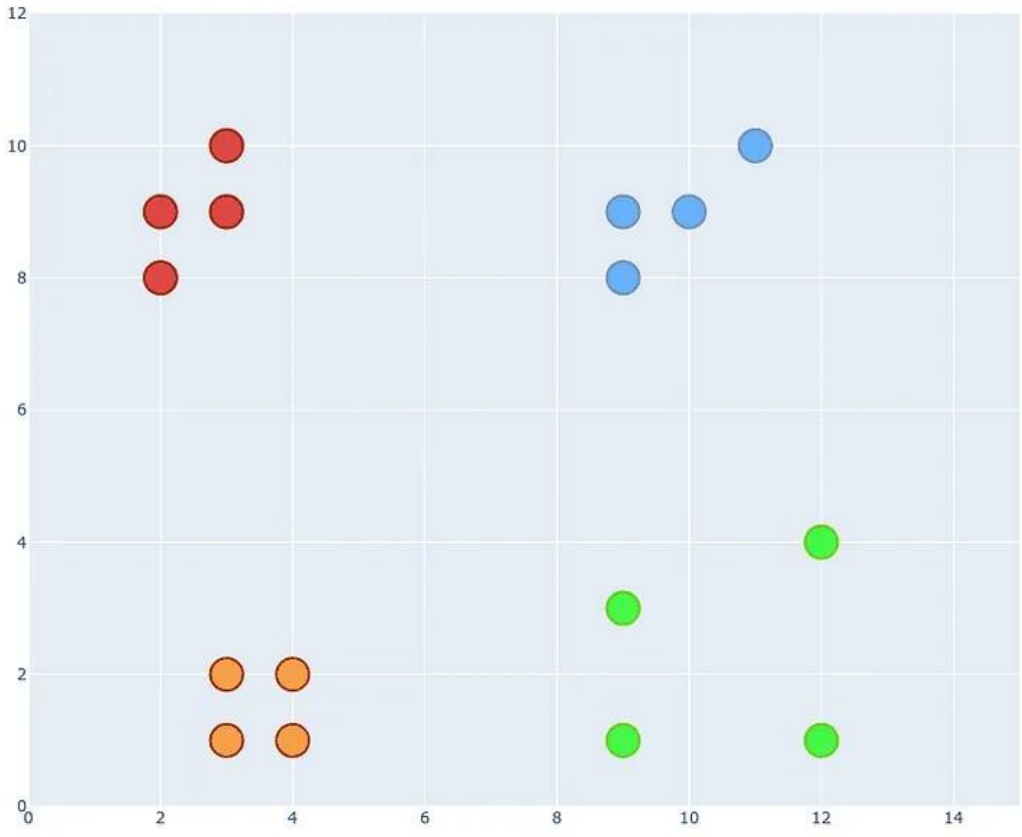
Right now you might wonder about σ^2 (variance) and that's a good thing. But let's just ignore it for now and assume I've already decided what it should be. After calculating the first point we have to do the same thing for every single point out there.



You might think, we're already done with this part. But that's just the beginning.

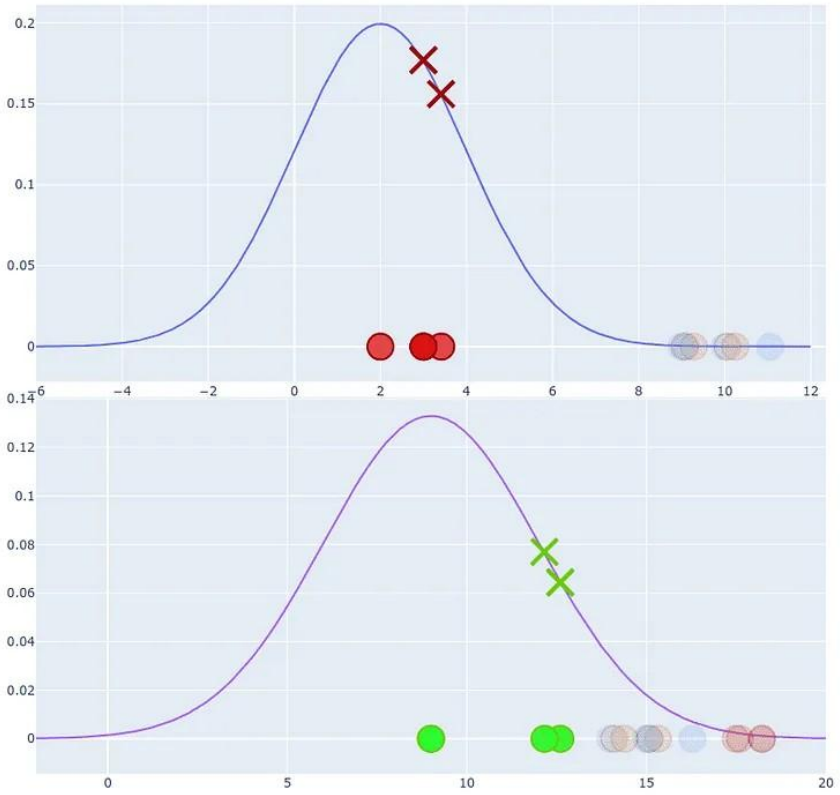
Scattered clusters and variance

Up to this point, our clusters were tightly bounded within its group. What if we have a new cluster like that:



We should be able to apply the same process as before, shouldn't we?

We're still not done. You can distinguish between similar and non-similar points but absolute values of probability are much smaller than in the first example (compare Y-axis values).



We can fix that by dividing the current projection value by the sum of the projections.

References

[1] (burnpiro), K. E. (2022, July 21). *T-SNE clearly explained*. Medium. https://towardsdatascience.com/t-sne-clearly-explained-d84c537f53a

[2] Agarwal, H. (n.d.). *T-SNE (t-distributed stochastic neighbor embedding) algorithm*. enjoyalgorithms. https://www.enjoyalgorithms.com/blog/tsne-algorithm-in-ml

[3] Suárez, D. (2022, May 5). *Dimensionality reduction - TSNE*. Apiumhub. https://apiumhub.com/tech-blog-barcelona/dimensionality-reduction-tsne/#:~:text=A%20second%20feature%20of%20t,effect%20on%20the%20resulting%20pictures.

[4] IBM. (n.d.). *What is gradient descent?*. IBM. https://www.ibm.com/topics/gradient-descent

[5]Awan, A. A. (2023). Introduction to T-SNE. https://www.datacamp.com/tutorial/introduction-t-sne

[6] *T-SNE clearly explained* (no date) *t-SNE clearly explained - Blog by Kemal Erdem*. Available at: https://erdem.pl/2020/04/t-sne-clearly-explained/ (Accessed: 15 September 2023).