<u>ML</u>

Dimensionality Reduction - PCA

Dimensionality Reduction

In both Statistics and Machine Learning, the number of attributes, features or input variables of a dataset is referred to as its **dimensionality**.

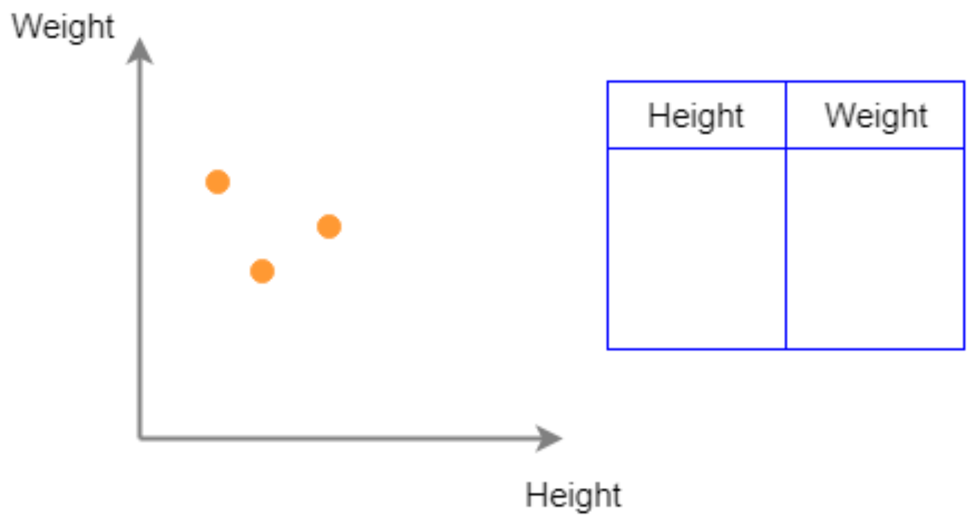


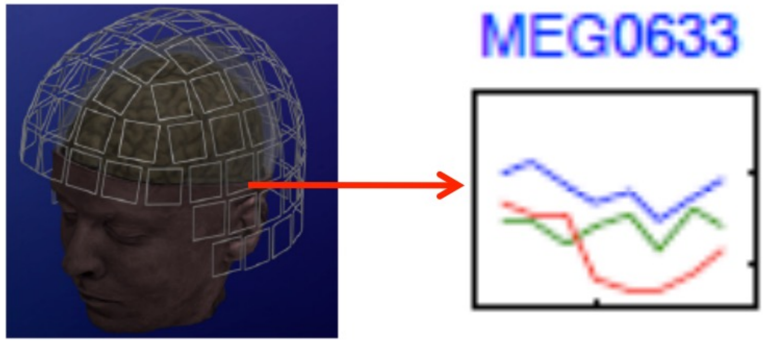
Image Copyright: Rukshan Pramoditha

Dimensionality Reduction

High dimensions = high volume of features.

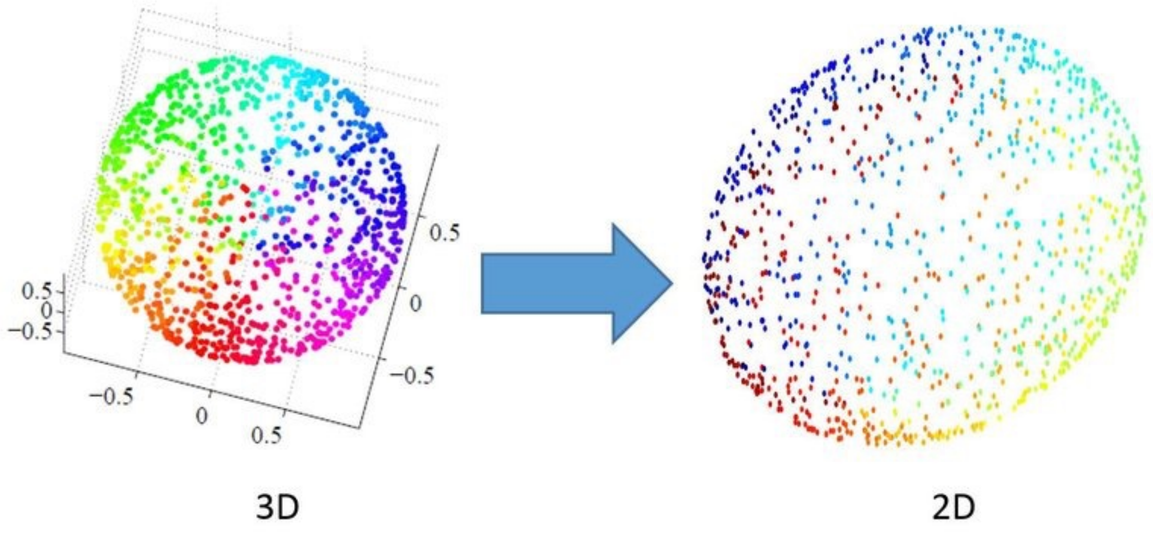
For example,

- **Document classification** Features per document is thousands of words/unigrams millions of contextual information
- MEG brain imaging
- Any high-volume data



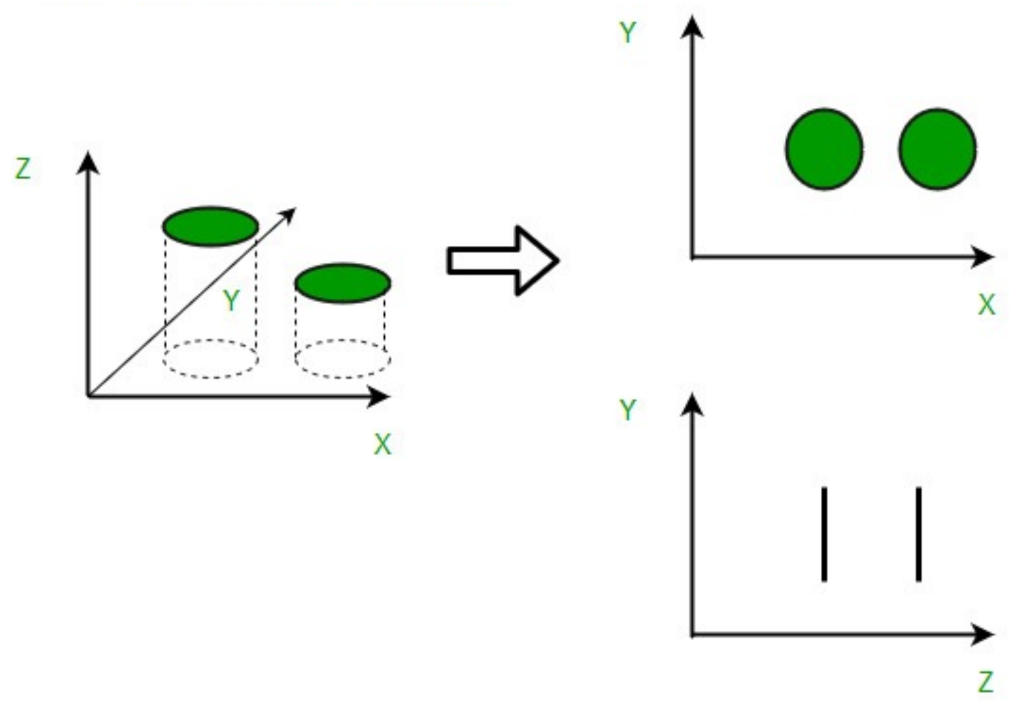
Dimensionality Reduction

Dimensionality reduction simply refers to the process of reducing the number of attributes in a dataset while keeping as much of the variation in the original dataset as possible. It is a data preprocessing step meaning that we perform dimensionality reduction before training the model.



The Importance of Dimensionality Reduction

When we reduce the dimensionality of a dataset, we lose some percentage (usually 1%-15% depending on the number of components or features that we keep) of the variability in the original data.



The Importance of Dimensionality Reduction

Advantages

- A lower number of dimensions in data means less training time and less computational resources and increases the overall performance of machine learning algorithms
- Dimensionality reduction avoids the problem of *overfitting*
- Dimensionality reduction is extremely useful for *data visualization*

The Importance of Dimensionality Reduction

Advantages

- Dimensionality reduction takes care of *multicollinearity*
- Dimensionality reduction removes noise in the data
- Dimensionality reduction can be used for image compression
- Dimensionality reduction can be used to transform non-linear data into a linearly-separable form

Dimensionality Reduction Methods

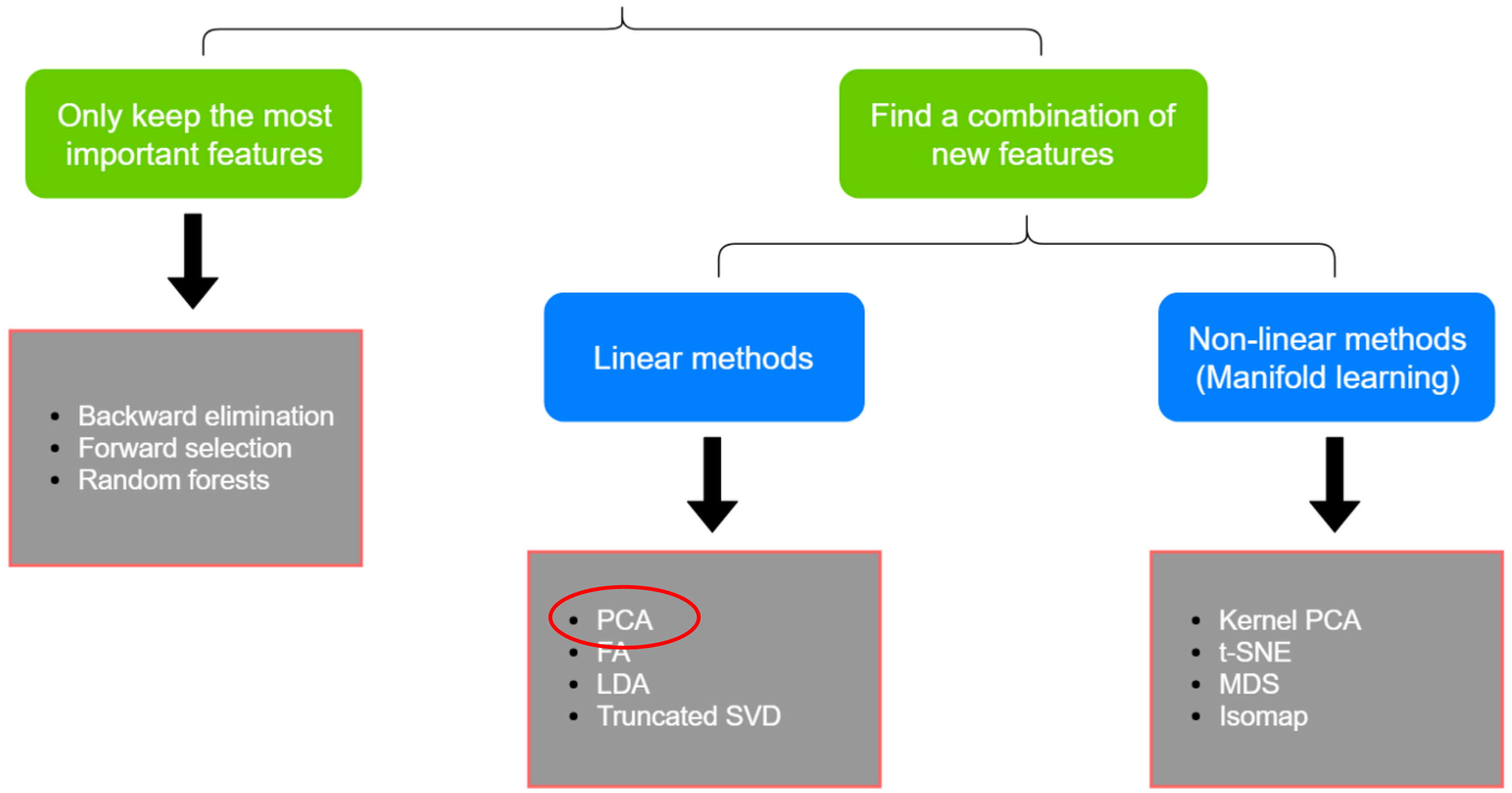


Image copyright: Rukshan Pramoditha

Principal Components Analysis (PCA)

Principal component analysis, or PCA, is a dimensionality-reduction method that is often used to reduce the dimensionality of large datasets, by transforming a large set of variables into a smaller one that still contains most of the information in the large set.

Useful for:

- Visualization
- Statistical: less dimensions >> better generalization
- Improving data quality
- Efficient use of resources (time / memory)

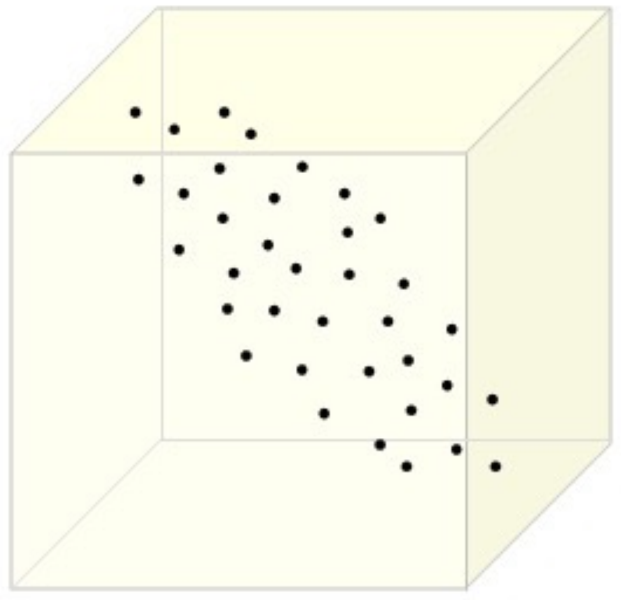
Principal Components Analysis (PCA)

PCA is an orthogonal projection or transformation of the data into a (possibly lower dimensional) subspace so that the variance of the projected data is maximized.

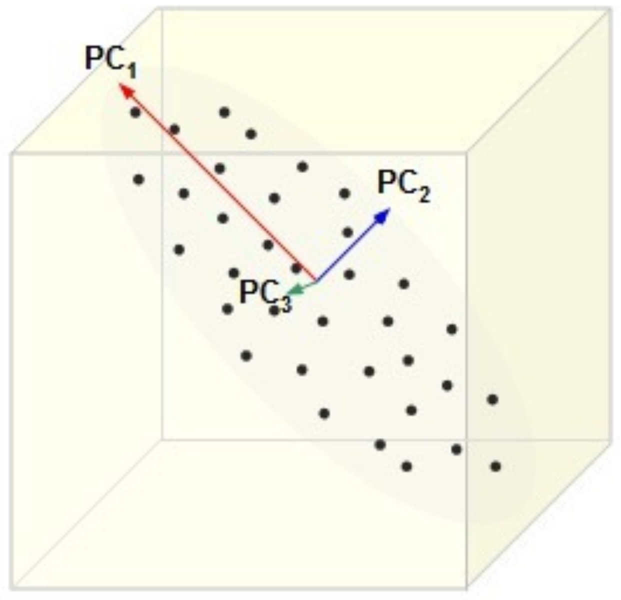
It is unsupervised technique for extracting variance structure from high dimensional datasets.

How can we create a new components of data that are related to the original dataset?

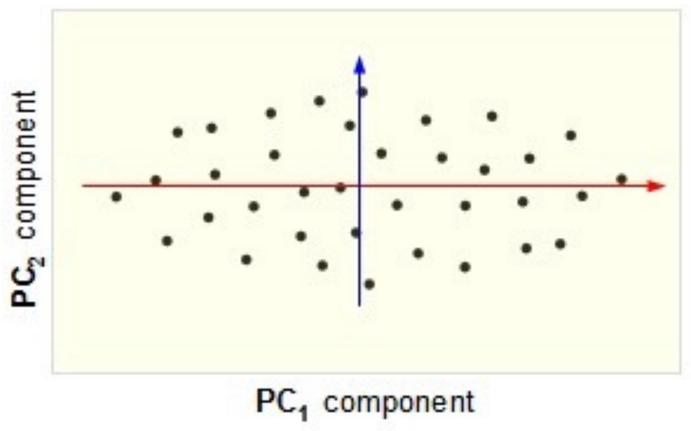
Principal Components Analysis (PCA)



a



b



c

PCA –Algorithm

Step-01: Get data.

Step-02: Compute the mean vector (µ).

Step-03: Subtract mean from the given data.

Step-04: Calculate the covariance matrix.

Step-05: Calculate the eigen vectors and eigen values of the covariance matrix.

Step-06: Choosing components and forming a feature vector.

Step-07: Deriving the new data set.

Principal Components Analysis - Example

Given the dataset –

f_1	f_2
2	1
3	5
4	3
5	6
6	7
7	8

PCA – COV Algorithm

Step-01: Get data.

Step-02: Compute the mean vector (μ).

Step-03: Subtract mean from the given data.

Step-04: Calculate the covariance matrix.

Step-05: Calculate the eigen vectors and eigen values of the covariance matrix.

Step-06: Choosing components and forming a feature vector.

Step-07: Deriving the new data set.

Principal Components Analysis - Example

f_1	f_2
2	1
3	5
4	3
5	6
6	7
7	8

$$\mu = \left(\frac{2+3+4+5+6+7}{6}, \frac{1+5+3+6+7+8}{6}\right) = (4.5, 5)$$

PCA – COV Algorithm

Step-01: Get data.

Step-02: Compute the mean vector (μ).

Step-03: Subtract mean from the given data.

Step-04: Calculate the covariance matrix.

Step-05: Calculate the eigen vectors and eigen values of the covariance matrix.

Step-06: Choosing components and forming a feature vector.

Step-07: Deriving the new data set.

Principal Components Analysis - Example

f_1	f_2
2	1
3	5
4	3
5	6
6	7
7	8

$\Longrightarrow$

f_1	f_2
-2.5	-4
-1.5	0
-0.5	-2
0.5	1
1.5	2
2.5	3

$\mu = (4.5, 5)$

PCA – COV Algorithm

Step-01: Get data.

Step-02: Compute the mean vector (μ).

Step-03: Subtract mean from the given data.

Step-04: Calculate the covariance matrix.

Step-05: Calculate the eigen vectors and eigen values of the covariance matrix.

Step-06: Choosing components and forming a feature vector.

Step-07: Deriving the new data set.

Principal Components Analysis - Example

f_1	f_2
-2.5	-4
-1.5	0
-0.5	-2
0.5	1
1.5	2
2.5	3

$$COV = \frac{1}{n} \cdot \sum_{i=1}^{n} (x_i - \mu)(x_i - \mu)^T$$

$$m_1 = \begin{bmatrix} -2.5 \\ -4 \end{bmatrix} (-2.5, -4) = \begin{pmatrix} 6.25 & 10 \\ 10 & 16 \end{pmatrix}$$

$$m_2 = \begin{bmatrix} -1.5 \\ 0 \end{bmatrix} (-1.5, 0) = \begin{pmatrix} 2.25 & 0 \\ 0 & 0 \end{pmatrix}$$

$$m_3 = \begin{bmatrix} -0.5 \\ -2 \end{bmatrix} (-0.5, -2) = \begin{pmatrix} 0.25 & 1 \\ 1 & 4 \end{pmatrix}$$

$$m_4 = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} (0.5, 1) = \begin{pmatrix} 0.25 & 0.5 \\ 0.5 & 1 \end{pmatrix}$$

$$m_5 = \begin{bmatrix} 1.5 \\ 2 \end{bmatrix} (1.5, 2) = \begin{pmatrix} 2.25 & 3 \\ 3 & 4 \end{pmatrix}$$

$$m_6 = \begin{bmatrix} 2.5 \\ 3 \end{bmatrix} (2.5, 3) = \begin{pmatrix} 6.25 & 7.5 \\ 7.5 & 9 \end{pmatrix}$$

Principal Components Analysis - Example

$$COV = \frac{1}{n} \cdot \sum_{i=1}^{n} (x_i - \mu)(x_i - \mu)^T$$

$$COV = \frac{1}{6} \cdot \left(m_1 + m_2 + \cdots + m_6\right)$$

$$COV = \frac{1}{6} \cdot \begin{pmatrix} 17.5 & 22 \\ 22 & 34 \end{pmatrix} = \begin{pmatrix} 2.92 & 3.67 \\ 3.67 & 5.67 \end{pmatrix}$$

PCA – COV Algorithm

Step-01: Get data.

Step-02: Compute the mean vector (μ).

Step-03: Subtract mean from the given data.

Step-04: Calculate the covariance matrix.

Step-05: Calculate the eigen vectors and eigen values of the covariance matrix.

Step-06: Choosing components and forming a feature vector.

Step-07: Deriving the new data set.

Principal Components Analysis - Example

To find the eigenvalues and vector we need to solve:

$$|COV - \lambda I| = 0$$

$$\left|\begin{pmatrix} 2.92 & 3.67 \\ 3.67 & 5.67 \end{pmatrix} - \begin{pmatrix} \lambda & 0 \\ 0 & \lambda \end{pmatrix}\right| = 0$$

$$\left|\begin{pmatrix} 2.92 - \lambda & 3.67 \\ 3.67 & 5.67 - \lambda \end{pmatrix}\right| = 0$$

We get:

$$(2.92 - \lambda)(5.67 - \lambda) - 3.67^2 = 0$$

$$\lambda^2 - 8.59\lambda + 3.09 = 0$$

$$\lambda_1 = 8.22, \lambda_2 = 0.38$$

PCA – COV Algorithm

Step-01: Get data.

Step-02: Compute the mean vector (μ).

Step-03: Subtract mean from the given data.

Step-04: Calculate the covariance matrix.

Step-05: Calculate the eigen vectors and eigen values of the covariance matrix.

Step-06: Choosing components and forming a feature vector.

Step-07: Deriving the new data set.

Principal Components Analysis - Example

Clearly, the second eigen value is very small compared to the first eigen value.

So, the second eigen vector can be left out. Eigen vector corresponding to the greatest eigen value is the principal component for the given data set.

Finding an eigenvector:

$$COV \cdot \vec{x} = \lambda \cdot \vec{x}$$

$$\begin{pmatrix} 2.92 & 3.67 \\ 3.67 & 5.67 \end{pmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 8.22 \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Principal Components Analysis - Example

$$\begin{pmatrix} 2.92 & 3.67 \\ 3.67 & 5.67 \end{pmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 8.22 \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$2.92x_1 + 3.67x_2 = 8.22x_1$$

$$3.67x_1 + 5.67x_2 = 8.22_{x_2}$$

$$\Downarrow$$

Linear system with infinty number of solutions

$$-5.3x_1 + 3.67x_2 = 0$$

$$-4.55x_1 + 5.67x_2 = 0$$

Principal Component

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_1 = 2.55 \\ x_2 = 3.67 \end{bmatrix}$$

PCA – COV Algorithm

Step-01: Get data.

Step-02: Compute the mean vector (μ).

Step-03: Subtract mean from the given data.

Step-04: Calculate the covariance matrix.

Step-05: Calculate the eigen vectors and eigen values of the covariance matrix.

Step-06: Choosing components and forming a feature vector.

Step-07: Deriving the new data set.

Principal Components Analysis - Example

Use PCA Algorithm to transform the pattern (2, 1) onto the eigenvector:

$$\begin{bmatrix} x_1 = 2.55 \\ x_2 = 3.67 \end{bmatrix}^T \cdot \big((2,1) - (4.5,5)\big) =$$

$$(2.55, 3.67) \cdot (-2.5, -4) =$$

$$-21.055$$

The value -21.055 is the compressed value of the vector (2,1)

PCA – Full Example

Principal Components Analysis - Example

f_1	f_2	f_3	Target
3	6	3	Yes
6	3	3	No
3	3	6	Yes

$$\mu = \left(\frac{3+6+3}{3}, \frac{6+3+3}{3}, \frac{3+3+6}{3}\right) = (4,4,4)$$

f_1	f_2	f_3	Target
-1	2	-1	**Yes**
2	-1	-1	**No**
-1	-1	2	**Yes**

Principal Components Analysis - Example

$$COV = \frac{1}{n} \cdot \sum_{i=1}^{n} (x_i - \mu)(x_i - \mu)^T$$

$$m_1 = \begin{bmatrix} -1 \\ 2 \\ -1 \end{bmatrix} (-1, 2, -1) = \begin{pmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{pmatrix}$$

$$m_2 = \begin{bmatrix} 2 \\ -1 \\ -1 \end{bmatrix} (2, -1, -1) = \begin{pmatrix} 4 & -2 & -2 \\ -2 & 1 & 1 \\ -2 & 1 & 1 \end{pmatrix}$$

$$m_3 = \begin{bmatrix} -1 \\ -1 \\ 2 \end{bmatrix} (-1, -1, 2) = \begin{pmatrix} 1 & 1 & -2 \\ 1 & 1 & -2 \\ -2 & -2 & 4 \end{pmatrix}$$

$$COV = \frac{1}{3} \cdot \begin{pmatrix} 6 & -3 & -3 \\ -3 & 6 & -3 \\ -3 & -3 & 6 \end{pmatrix} = \begin{pmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{pmatrix}$$

Principal Components Analysis - Example

$$|COV - \lambda I| = 0$$

$$\left|\begin{pmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{pmatrix} - \begin{pmatrix} \lambda & 0 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & \lambda \end{pmatrix}\right| = 0$$

$$\left|\begin{pmatrix} 2-\lambda & -1 & -1 \\ -1 & 2-\lambda & -1 \\ -1 & -1 & 2-\lambda \end{pmatrix}\right| = 0$$

$$\left|\begin{pmatrix} 2-\lambda & -1 & -1 \\ -1 & 2-\lambda & -1 \\ -1 & -1 & 2-\lambda \end{pmatrix}\right| = C_2 \to C_2 - C_3 = \left|\begin{pmatrix} 2-\lambda & 0 & -1 \\ -1 & 3-\lambda & -1 \\ -1 & -3+\lambda & 2-\lambda \end{pmatrix}\right| =$$

$$R_3 \to R_2 + R_3 = \left|\begin{pmatrix} 2-\lambda & 0 & -1 \\ -1 & 3-\lambda & -1 \\ -2 & 0 & 1-\lambda \end{pmatrix}\right|$$

Principal Components Analysis - Example

$$(3-\lambda)\cdot\left|\begin{pmatrix}2-\lambda & -1\\ -2 & 1-\lambda\end{pmatrix}\right| = 0$$

$$(3-\lambda)\cdot[(2-\lambda)(1-\lambda)-2] = 0$$

$$(3-\lambda)(\lambda^2-3\lambda) = 0$$

$$\lambda_1 = 0, \lambda_2 = 3$$

$$\begin{pmatrix}2 & -1 & -1\\ -1 & 2 & -1\\ -1 & -1 & 2\end{pmatrix}\begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix} = 3\begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix}$$

$\lambda = 3$

$$-x_1 - x_2 - x_3 = 0$$

$$-x_1 - x_2 - x_3 = 0$$

$$-x_1 - x_2 - x_3 = 0$$

$$\begin{pmatrix}-1 & -1 & -1\\ -1 & -1 & -1\\ -1 & -1 & -1\end{pmatrix} \rightarrow \begin{pmatrix}1 & 1 & 1\\ 0 & 0 & 0\\ 0 & 0 & 0\end{pmatrix}$$

$$Solution: \quad t\begin{bmatrix}-1\\0\\1\end{bmatrix} + s\begin{bmatrix}-1\\1\\0\end{bmatrix} = \begin{bmatrix}-2\\1\\1\end{bmatrix}$$

Principal Components Analysis - Example

$\lambda = 0$

$$\begin{pmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{pmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = 0$$

$$\begin{pmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{pmatrix} \rightarrow \begin{pmatrix} 1 & -2 & 1 \\ 2 & -1 & -1 \\ -1 & -1 & 2 \end{pmatrix} \rightarrow \begin{pmatrix} 1 & -2 & 1 \\ 0 & 3 & -3 \\ 0 & -3 & -3 \end{pmatrix} \rightarrow \begin{pmatrix} 1 & -2 & 1 \\ 0 & 1 & -1 \\ 0 & 0 & 0 \end{pmatrix}$$

$$Solution: \quad t \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

$PCA\ Compression:$

$$PC1 = \begin{bmatrix} -2 \\ 1 \\ 1 \end{bmatrix}^T (x_i - \mu)$$

$$PC2 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}^T (x_i - \mu)$$

T-distributed Stochastic Neighbor Embedding

t-SNE

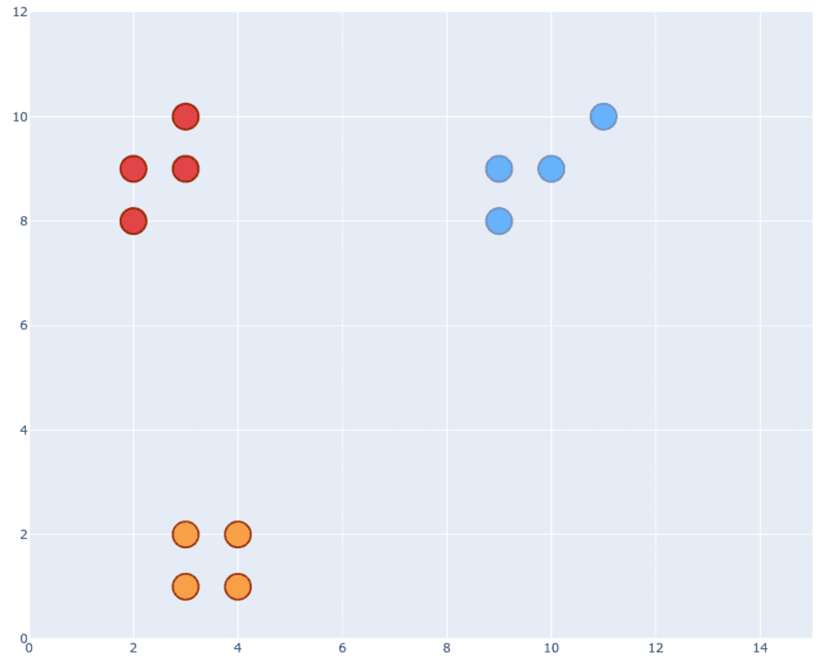
T-SNE

Close points should remain tight, distant points shall stay far.

- An unsupervised non-linear dimensionality reduction technique for data exploration and visualizing high-dimensional data.
- Based on probability distribution and the **Kullback-Leibler (KL) divergence.**

The SNE Phase

The first part of the algorithm is to create a **probability distribution** that represents similarities between neighbors.

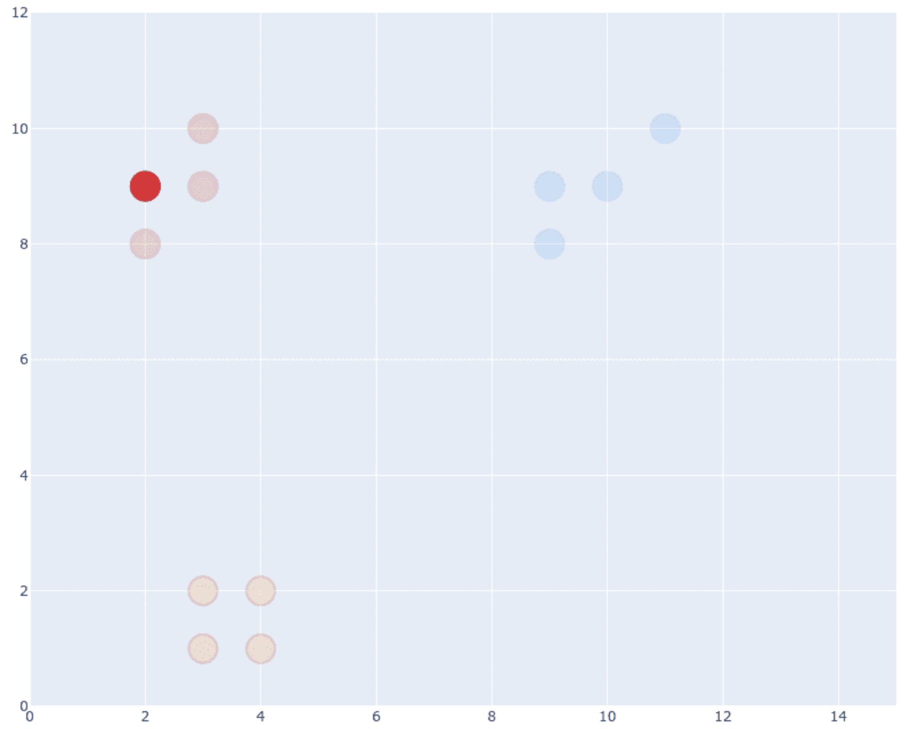


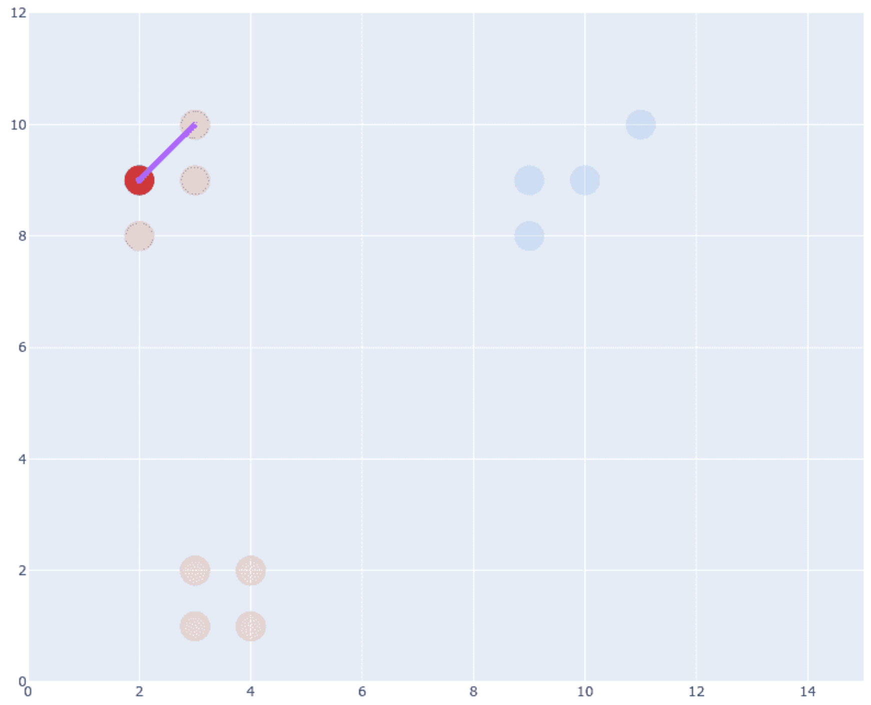
The similarity between x_j, x_i is the conditional probability $P\left(x_j \middle| x_i\right)$.

The SNE Phase

We've picked one of the points from the dataset.

Next, we have to pick another point and calculate Euclidean , denoted as $|x_i - x_j|$

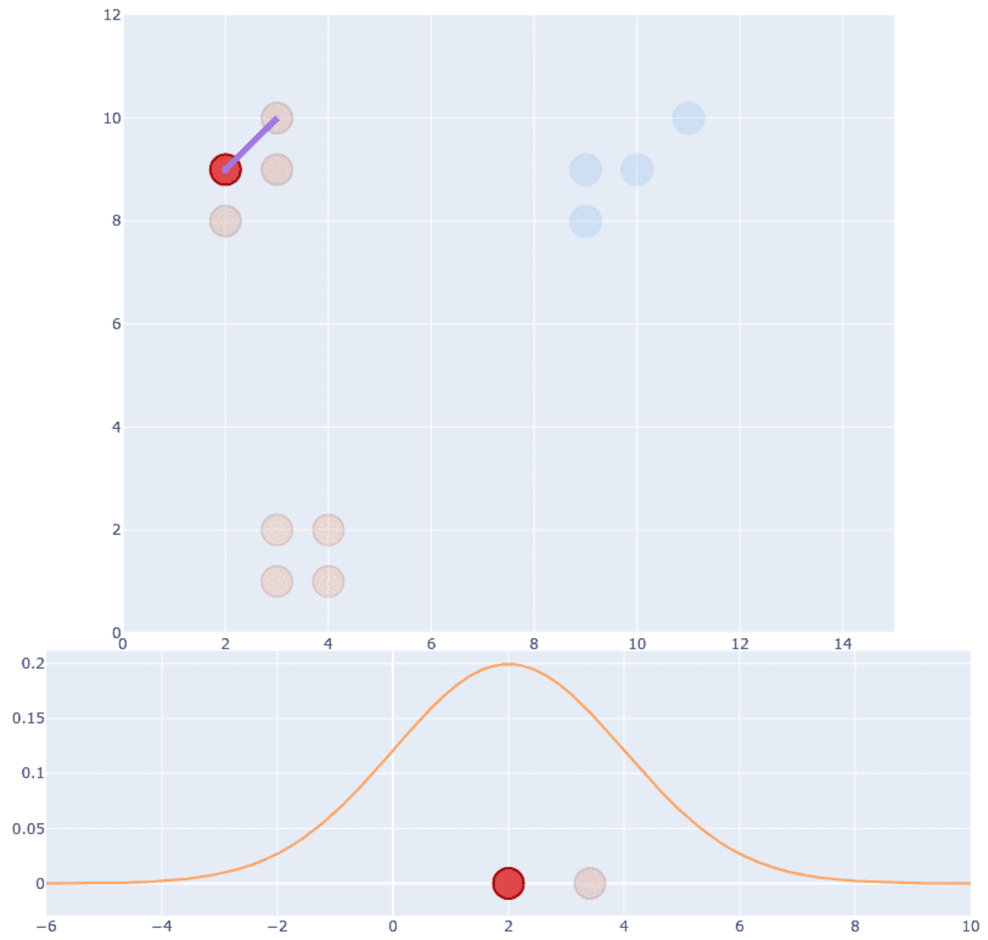




The SNE Phase

The next part of the original paper states that it has to be **proportional to probability density under a Gaussian centered at x_i.**

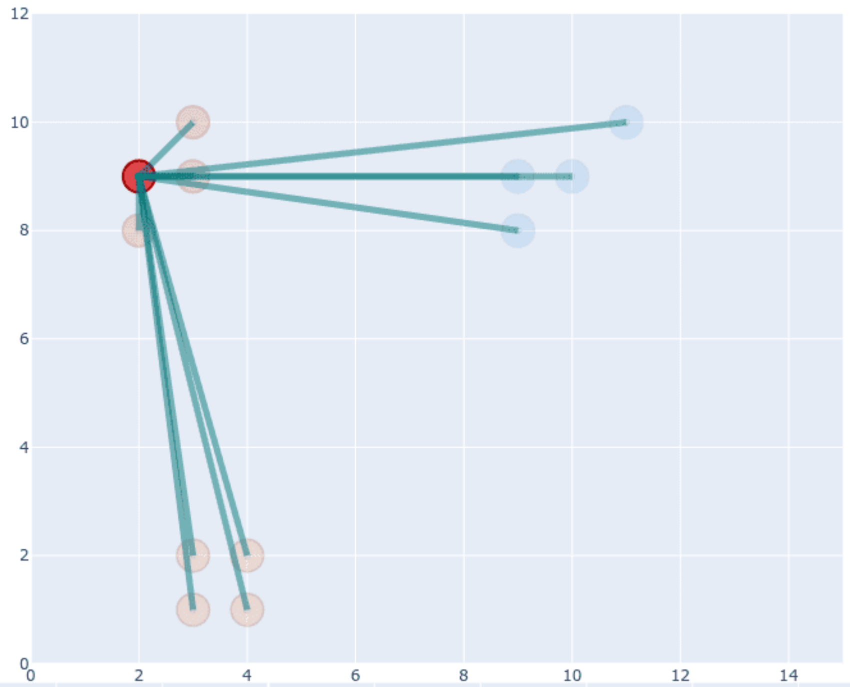
We need to generate Gaussian distribution with mean at x_i and place our distance on the X-axis.

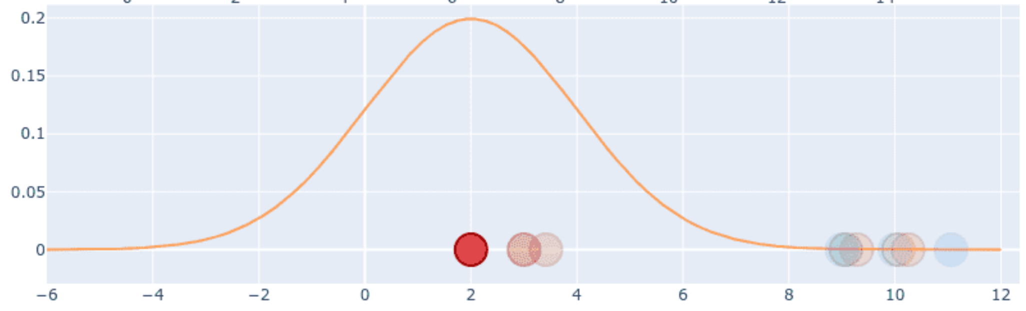


The SNE Phase

The next part of the original paper states that it has to be **proportional to probability density under a Gaussian centered at x_i**.

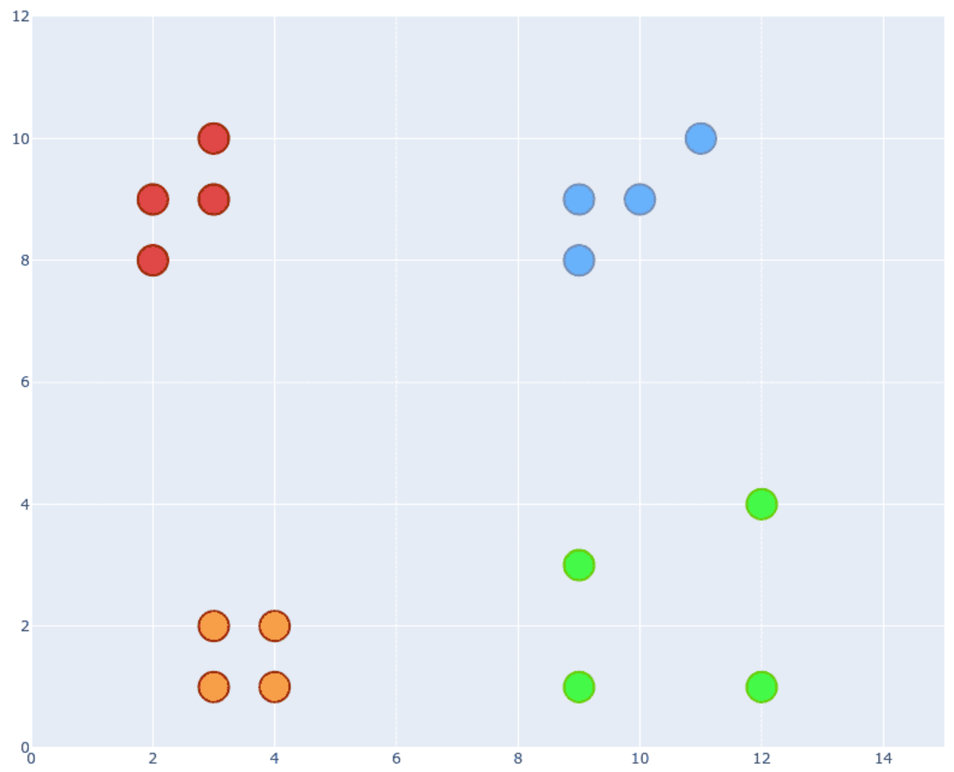
We need to generate Gaussian distribution with mean at x_i and place our distance on the X-axis.





The SNE Phase

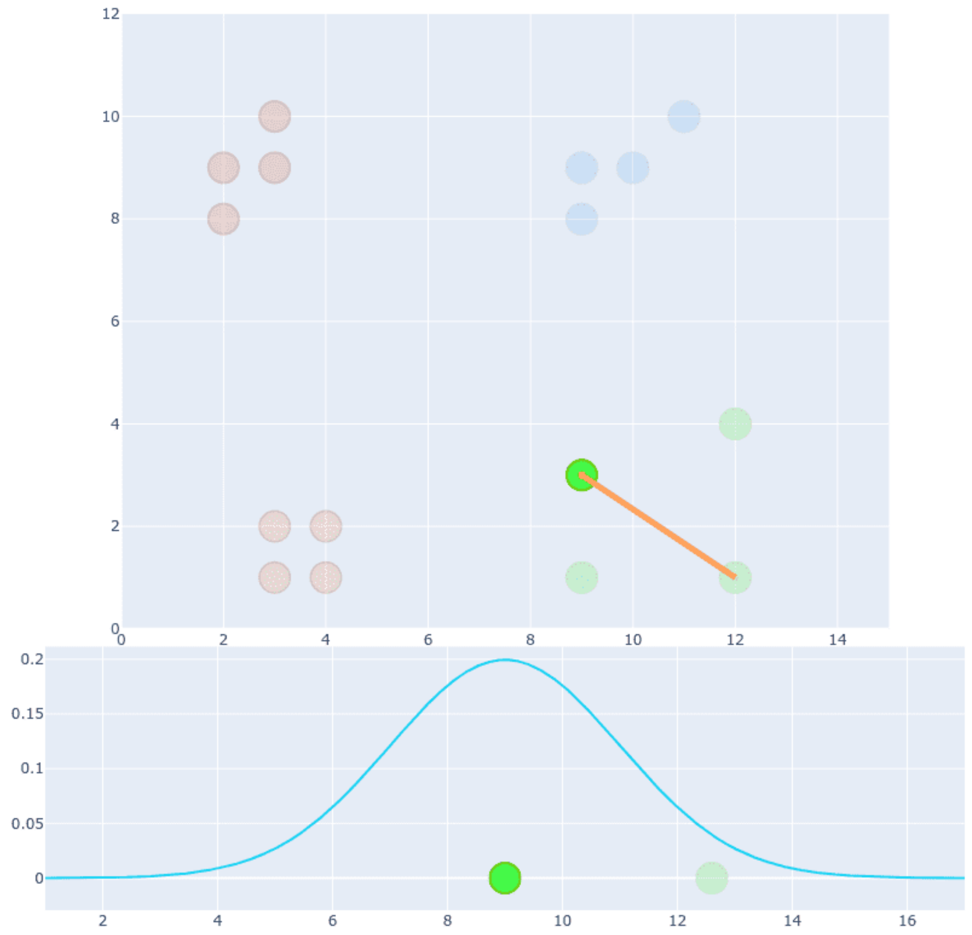
Consider updating the dataset by adding one more group:



The SNE Phase

You can distinguish between similar and non-similar points, but absolute values of probability are much smaller than in the first example.

How can standardize it?



The SNE Phase

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

Let g be the map function:

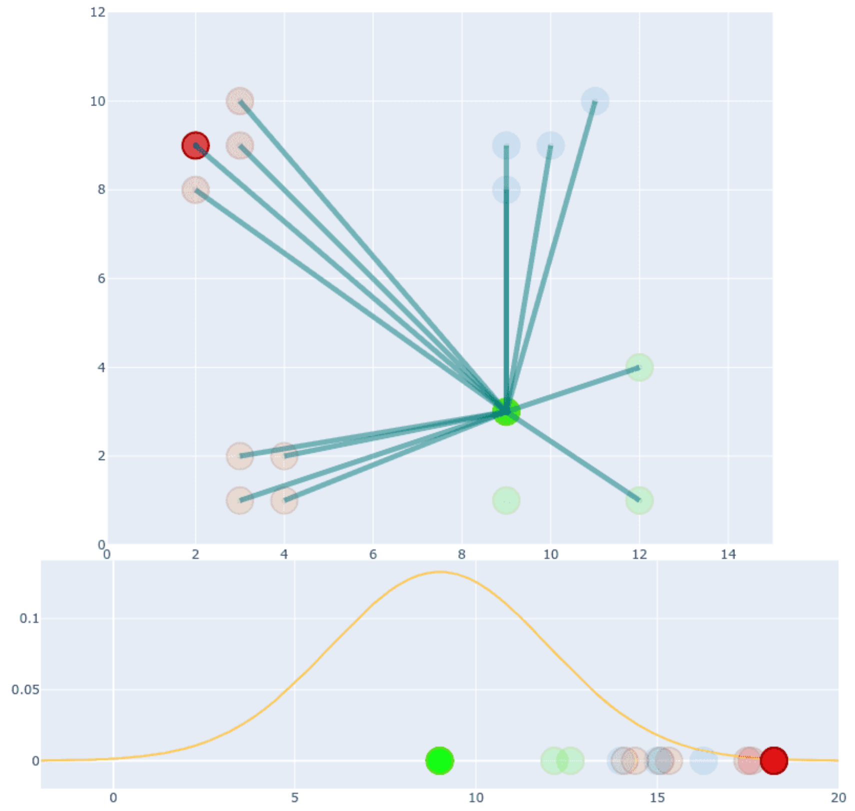
$$P\left(x_j \middle| x_i\right) = \frac{g(|x_i - x_j|)}{\sum_{k \neq i} g(|x_i - x_k|)}$$

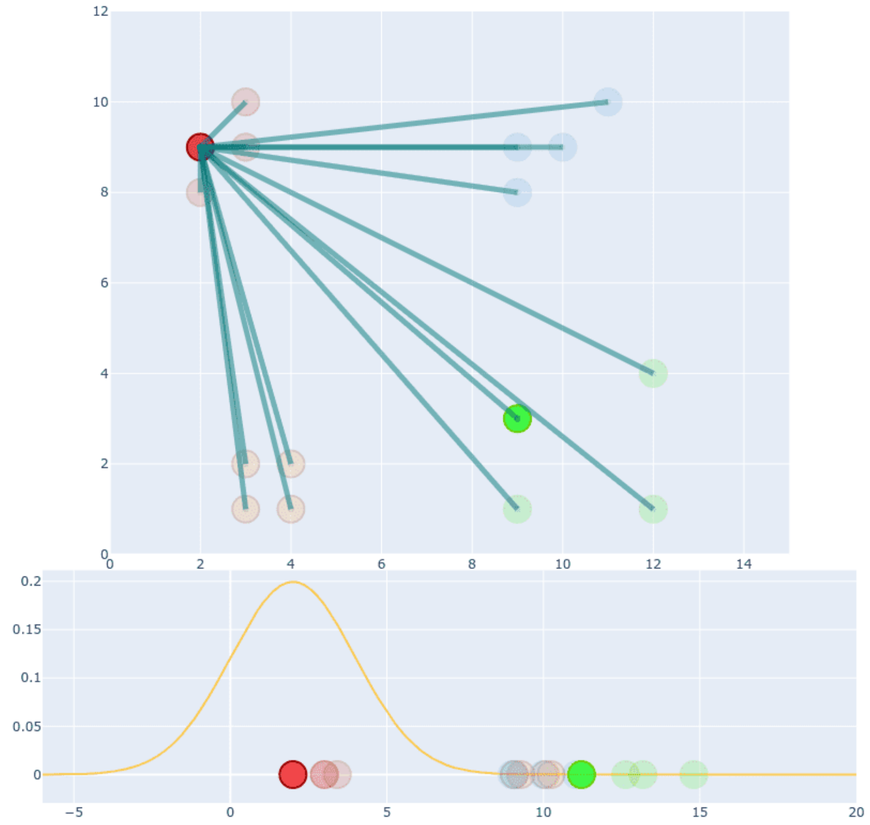
For Example,

$$[0.2, 0.05, 0.3]$$

$$\frac{0.2}{0.2 + 0.05 + 0.3} = 0.36 \qquad \frac{0.05}{0.2 + 0.05 + 0.3} = 0.1 \qquad \frac{0.3}{0.2 + 0.05 + 0.3} = 0.54$$

How to Choose between $j|i$ or $i|j$?





How to Choose between $j|i$ or $i|j$?

Let $X \sim N(\mu, \sigma^2)$ then the probability mass function is defined by:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \cdot e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Let g be the Gaussian Mapping defined as follows:

$$d_{j|i} = |x_j - x_i|$$

$$g(d_{j|i}) = e^{-\frac{\left\|x_i - x_j\right\|^2}{2\sigma_i^2}}$$

Where σ^2 is the variance of the original group distribution.

Perplexity

- A perplexity is a target number of neighbors for our central point.
- Generally, the higher the perplexity is the higher value variance has.

Definition from the original paper:

SNE performs a binary search for the value of sigma that produces probability distribution with a fixed perplexity that is specified by the user

Mostly used in range of 5 to 50

The Problem

- The next part of t-SNE is to create low-dimensional space with the same number of points as in the original space.
- Points should be spread randomly on a new space.
- The goal of this algorithm is to find similar probability distribution in low-dimensional space.

But how can we find such distribution to compress???

The Kullback-Leibler (KL) Divergence

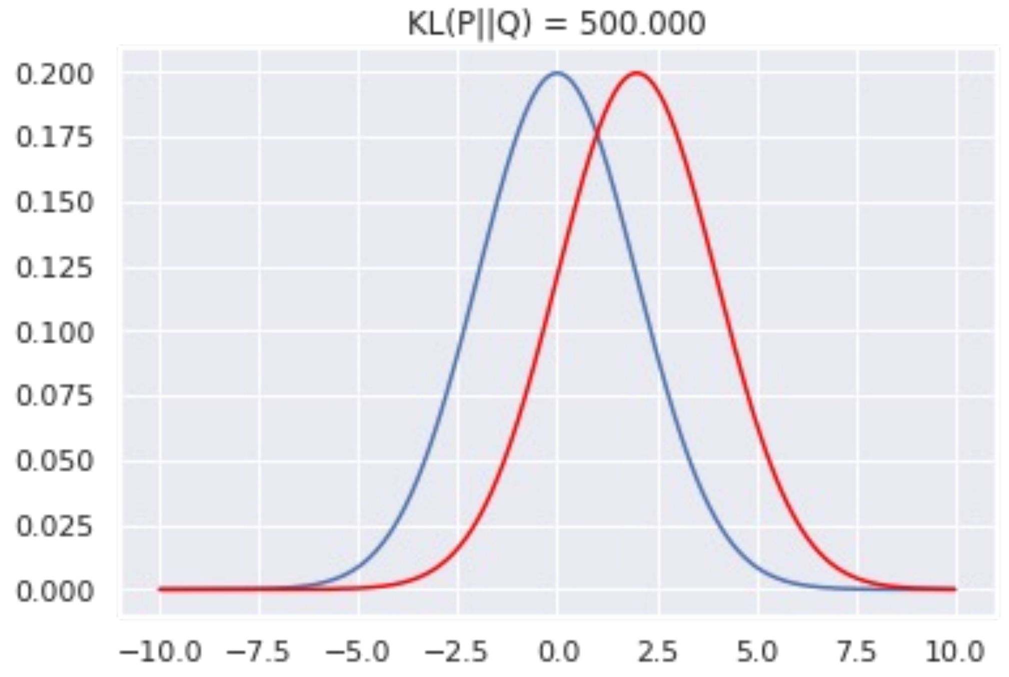
- Measure the difference between two probability distribution.
- Two exactly similar distribution raise KL-divergence of 0.

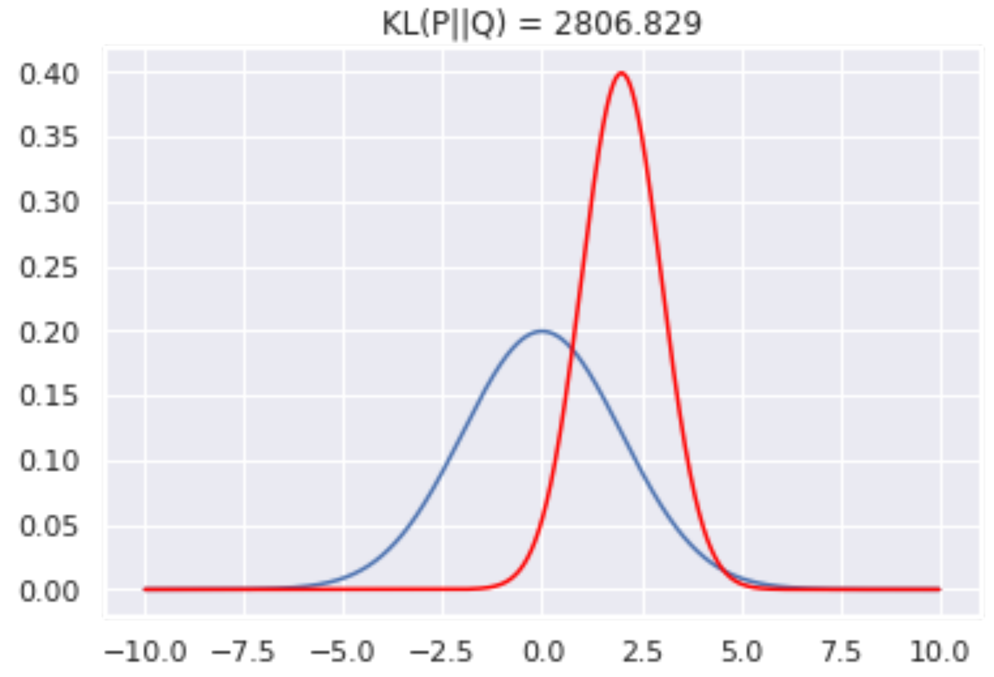
Let P, Q be two distributions from Ω probability space, then the

KL-divergence is calculated by:

$$D_{KL}(P\|Q) = \sum_{x\in\Omega} P(x)\left[\log\frac{P(x)}{Q(x)}\right]$$

Similarity, NOT Distance





The KL Divergence in t-SNE

- To optimize this distribution t-SNE is using KL divergence we can use Gradient Descent method.
- Let P be the data distribution, and Q an arbitrary distribution selected.

  The cost function $C = D_{KL}(P \| Q)$ can be minimize each step by:

$$C^* = C^* - \alpha \cdot \frac{\partial C}{\partial Q}$$

The t-SNE Algorithm

Input: Dataset $X = \{x_1, \dots, x_n\}$, Perplexity, T – number of iterations

Output: Low dimensional factors $Y = \{y_1, \dots, y_n\}$

```
- Compute $p_{ij}$ and $\sigma_i$ for each $x_i, x_j$
- Initialize $Y \sim N(0, 10^{-4})$
- For $t \leftarrow 0$ to $T$
    - Compute $\frac{\partial D_{KL}(P\|Q)}{\partial Y}$
    - Use SGD and update the distribution.
```

Multidimensional Scaling

MDS

Introduction

Metric MDS — is also known as **Principal Coordinate Analysis (PCoA).**

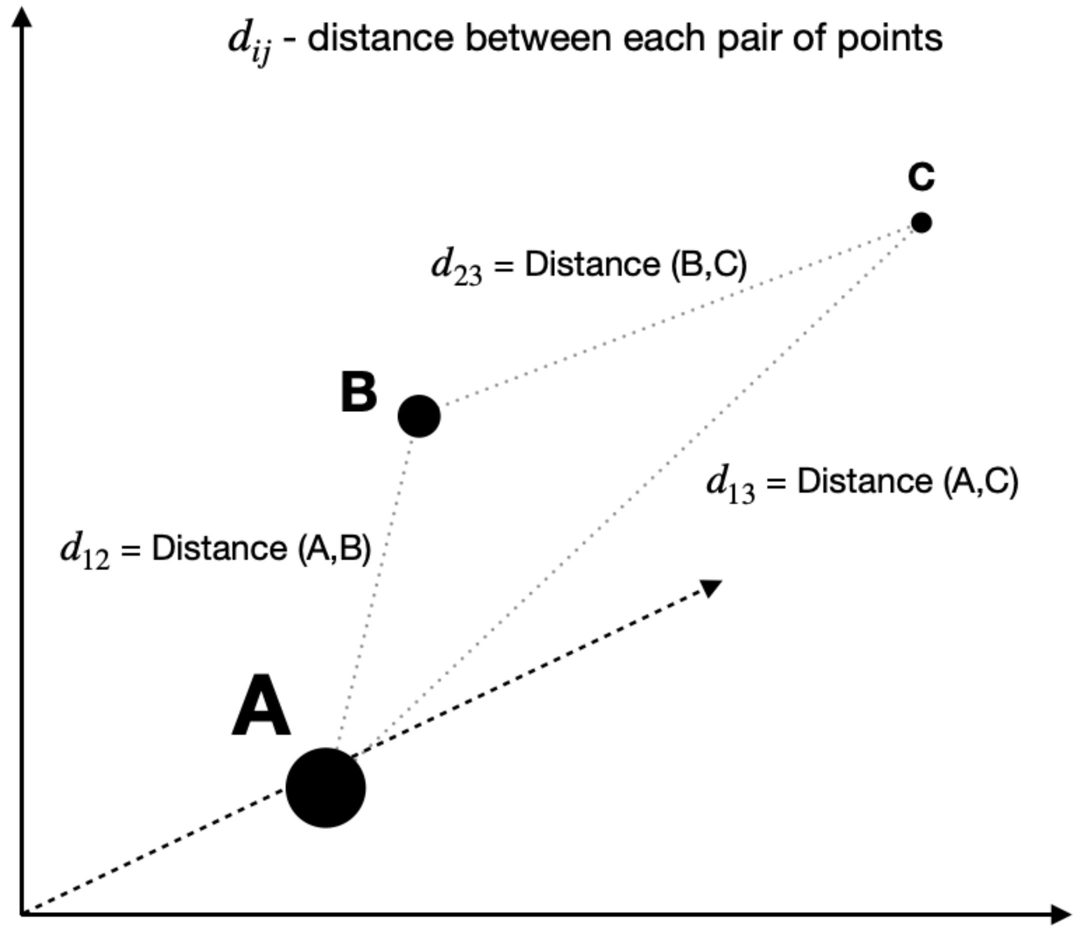
Designed to deal to continuous dataset.

Non-metric MDS — is designed to deal with ordinal / encoded categorical data.

We would focus on **Metric (classical) MDS**. The algorithm defines **<u>Stress</u>** - the degree of agreement between the distances in the original and the compressed dimensions.

How it Works?

Step 1 - The algorithm calculates distances between each pair of points, as illustrated below.


d_{ij} - distance between each pair of points
C
d_{23} = Distance (B,C)
B
d_{13} = Distance (A,C)
d_{12} = Distance (A,B)
A

How it Works?

Step 2 - Minimize the value of **Stress**.

$$Stress_D(x_1, x_2, \ldots, x_N) = \sqrt{\sum_{i \neq j = 1,\ldots,N} (d_{ij} - ||x_i - x_j||)^2}$$

The goal of the algorithm is to minimize the value of stress.

Where $x_1, \ldots, x_N$ are data points with their new set of coordinates in lower dimensional space.

d_{ij} is the actual distance we have calculated between the two corresponding data points in their original dimensional space.

$||x_i - x_j||$ is the distance between the two corresponding data points in their lower dimensional space.

The closer the value of $||x_i - x_j||$ is to d_{ij} the smaller will be the value of stress.

The MDS Algorithm

<u>Input</u>: Dataset $X = \{x_1, \dots, x_n\}$, k – dimension of the compressed data

<u>Output</u>: Low dimensional factors

- D – Similarity matrix between all $i \neq j$
- Minimize the Stress cost function and update the matrices.

Note. Behind the scenes, it uses matrix factorization (i.e., split matrix to multiplication of two matrices)