Xử lí đa chiều

♢ Nhóm:

1. Vũ Quang Minh 18110150
2. Lê Thành Công 18110066
3. Nguyễn Tiến Thông 18110229

♢ Mục lục

A Thuật toán

I Principal Component Analysis (PCA)

Mục tiêu của PCA là tìm ra được một hệ thống chiều khác có ý nghĩa nhất để mô tả tập dữ liệu đã cho.
Một biến ngẫu nhiên $\mathbf{x}$ có N phần tử, $\{\mathbf{x}_n\}$, $n = 1, 2, \ldots, N$, và mỗi samples có D chiều. Ta sẽ có $N \times D$ ma trận $\mathbf{X}$, với mỗi dòng là một sample $\mathbf{x}_i$. Ánh xạ tuyến tính $\mathbf{P}$ sẽ được chọn và dữ liệu sau khi biến đổi sẽ là:

$$\mathbf{Y} = \mathbf{XP}$$

với mỗi cột của $\mathbf{P}$ là $\mathbf{p}_i$
Phương trình trên đại diện cho việc chuyển đổi hệ thống chiều và có thể được diễn giải thế này
* $\mathbf{P}$ là ma trận chuyển đổi $\mathbf{X}$ thành $\mathbf{Y}$.
* $\mathbf{P}$ là một cách xoay và cách nới rộng.
* Các cột của $\mathbf{P}$, $\{\mathbf{p}_1, \ldots, \mathbf{p}_{D'}\}$ là tập các vector đơn vị mới cho việc chuyển đổi các sample trong $\mathbf{X}$.

Ta có các cách để tìm ra ma trận $\mathbf{P}$:
1. Thực hiện PCA sao cho phương sai lớn nhất.
2. Thực hiện PCA sao cho mean-square error (MSE) là nhỏ nhất.

Dữ liệu ban đầu có những thành phần này: noise, rotation và redundancy.
Ta thường làm một bước tiền xử lí dữ liệu ban đầu: mỗi điểm sẽ được trừ đi kì vọng của dữ liệu. Điều này sẽ làm cho dữ liệu có kì vọng là không.

Covariance của $\mathbf{X}$ là $\mathbf{C}_\mathbf{X} \equiv \frac{1}{n-1}\mathbf{X}^T\mathbf{X}$
Ta muốn (1) redundancy nhỏ nhất có thể, theo covariance, và (2) tín hiệu là lớn nhất, theo phương sai.
Covariance $\mathbf{C}_\mathbf{Y}$ sẽ được chọn sao cho ma trận này là ma trận đường chéo.

Vấn đề là làm sao để tìm ra ma trận $\mathbf{P}$ với $\mathbf{Y} = \mathbf{XP}$ sao cho $\mathbf{C_Y} \equiv \frac{1}{n-1}\mathbf{Y}^T\mathbf{Y}$ là ma trận đường chéo?
Trước khi đi vào chi tiết phương pháp, ta sẽ xét đến giới hạn của PCA.
* Tuyến tính.
* Trung bình và phương sai mô tả theo phân phối xác xuất.
* Các thành phần chính trực giao với nhau.

Ta sẽ thực hiện các bước sau:
+ Chọn một hướng trong không gian D chiều sao cho phương sai của $\mathbf{X}$ là lớn nhất.
+ Chọn một hướng khác trong không gian D chiều sao cho phương sai lớn nhất, biết rằng hướng này sẽ phải trực giao với các hướng đã chọn trước đó.
+ Lặp lại các bước này cho đến khi D vector được chọn.

Xét tổ hợp tuyến tính $y_m = \mathbf{w}_m^T\mathbf{x}$, ta sẽ chọn $\mathbf{w}_k$ sao cho y_m không liên quan gì đến tất cả các thành phần chính tìm được

$$E[y_m y_k] = 0, k < m \qquad (\text{vì } E[y_i] = 0 \qquad \forall i)$$

Vậy ta sẽ có

$$E[y_m y_k] = E[(\mathbf{w}_m^T\mathbf{x})(\mathbf{w}_m^T\mathbf{x})] = \mathbf{w}_m^T E[\mathbf{x}\mathbf{x}^T]\mathbf{w}_k = 0$$

Ma trận $E[\mathbf{x}\mathbf{x}^T]$ là ma trận covariance của $\mathbf{x}$.

Với thành phần chính đầu tiên, $\mathbf{w}_1$, ta có

$$E[y_1^2] = \mathbf{w}_1^T E[\mathbf{x}\mathbf{x}^T]\mathbf{w}_1$$

thoả $\|\mathbf{w}_1\| = 1$
Tham số $\mathbf{w}_1$ được chọn theo Lagrange multiplier. Vậy ta tìm được $\mathbf{w}_1 = \mathbf{e}_1$ là vector riêng làm chéo ma trận covariance $E[\mathbf{x}\mathbf{x}^T]$
Ở thành phần chính thứ hai, ta sẽ chọn $\mathbf{w}_2$ sao cho $\mathbf{w}_2^T E[\mathbf{x}\mathbf{x}^T]\mathbf{w}_1$ lớn nhất. Vậy ta có $\mathbf{w}_2 = \mathbf{e}_2$
Cứ thế cho thành phần chính thứ ba trở đi.

Suy ra ta có $\mathbf{P}$ có các phần tử $\mathbf{p}_i = \mathbf{w}_i$ là các vector riêng của ma trận covariance của $\mathbf{X}$.

Vậy để tìm một ma trận $\mathbf{P}$ với $\mathbf{Y} = \mathbf{XP}$ sao cho $\mathbf{C_Y} \equiv \frac{1}{n-1}\mathbf{Y}^T\mathbf{Y}$ là ma trận đường chéo, ta sẽ làm theo thuật toán dưới đây:
(1) Trừ kì vọng cho từng điểm dữ liệu.
(2) Tính ma trận covariance của tập dữ liệu $\mathbf{X}$.
(3) Tính trị riêng và vector riêng của ma trận covariance.
(4) Sắp xếp vector riêng ứng với trị riêng từ lớn đến bé.
(5) Chọn số lượng thành phần chính cần thiết.
(6) Tính ma trận chuyển đổi $\mathbf{Y}$.

II Kernel PCA

Đầu tiên xét một ánh xạ từ tất cả các điểm $\mathbf{x}$ đến $f(\mathbf{x})$ trong một không gian đa chiều F, mà ma trận covariance có thể được ước tính

$$\Sigma_f = \frac{1}{N}\sum_{n=1}^{N} f(\mathbf{x}_n)f(\mathbf{x}_n)^\mathsf{T}$$

Thế $\Sigma_f\phi_i = \lambda_i\phi_i$ vào phương trình chéo hoá của ma trận covariance, ta được

$$\left[\frac{1}{N}\sum_{n=1}^{N} f(\mathbf{x})f(\mathbf{x}_n)^\mathsf{T})\right]\phi_i = \frac{1}{N}\sum_{n=1}^{N}\langle f(\mathbf{x}_n), \phi_i\rangle f(\mathbf{x}_n) = \lambda_i\phi_i$$

Ta thấy rằng vector riêng ϕ_i là một tổ hợp tuyến tính của N điểm dữ liệu

$$\phi_i = \frac{1}{\lambda_i N} \sum_{n=1}^{N} \langle f(\mathbf{x}_n), \phi_i \rangle \, f(\mathbf{x}_n) = \sum_{n=1}^{N} a_n^{(i)} f(\mathbf{x}_n)$$

với

$$a_n^{(i)} = \frac{1}{\lambda_i N} \langle f(\mathbf{x}_n), \phi_i \rangle$$

Nhân bên trái cho $f(\mathbf{x}_m)^\mathsf{T}$ vào hai vế của phương trình trên, ta được:

$$\langle f(\mathbf{x}_n), \phi_i \rangle = \lambda_i N a_m^{(i)} = \sum_{n=1}^{N} a_n^{(i)} \langle f(\mathbf{x}_m), f(\mathbf{x}_n) \rangle = \sum_{n=1}^{N} a_n^{(i)} k(\mathbf{x}_m, \mathbf{x}_n)$$

với

$$k(\mathbf{x}_m, \mathbf{x}_n) = \langle f(\mathbf{x}_m), f(\mathbf{x}_n) \rangle, \qquad m, n = 1, 2, \ldots, N$$

là nhân đại diện cho một tích vô hướng của hai vectơ trong không gian F. Nếu ta xét $m = 1, 2, \ldots, N$, phương trình trên trở thành m thành phần của phương trình vectơ

$$\lambda_i N \mathbf{a}_i = \mathbf{K} \mathbf{a}_i$$

với

$$\mathbf{K} = \begin{bmatrix} \ldots & \ldots & \ldots \\ \ldots & k(\mathbf{x}_i, \mathbf{x}_j) & \ldots \\ \ldots & \ldots & \ldots \end{bmatrix}_{N \times N}$$

là một ma trận $N \times N$ phần tử nhân $k(\mathbf{x}_i, \mathbf{x}_j)$, $i, j = 1, 2, \ldots, N$, và $\mathbf{a}_i = [a_1^{(i)}, a_2^{(i}), \ldots, a_N^{(i)}]^\mathsf{T}$, $i = 1, 2, \ldots, N$, là N vectơ chéo của $\mathbf{K}$, có thể được tính bằng cách giải vấn đề chéo hoá của ma trận $\mathbf{K}$. Vì các trị riêng của $\mathbf{K}$ tỉ lệ với các trị riêng λ_i của ma trận covariance Σ_f trong không gian đặc trưng, việc chọn các đặc trưng trong PCA có thể được tính toán bằng cách giữ chỉ một số ít số lượng thành phần tương ứng với các giá trị trị riêng mà không bị mất nhiều thông tin.

Các điểm dữ liệu mới $\mathbf{x}$ có thể được ánh xạ đến $f(\mathbf{x})$ trong không gian đa chiều F, với thành phần PCA thứ i có thể được tính như là một hình chiếu của nó trong vectơ riêng thứ i, $\phi_i = \sum_{n=1}^{N} a_n^{(i)} f(\mathbf{x}_n)$

$$\langle f(\mathbf{x}), \phi_i \rangle = \left\langle f(\mathbf{x}), \sum_{n=1}^{N} a_n^{(i)} f(\mathbf{x}_n) \right\rangle = \sum_{n=1}^{N} a_n^{(i)} \langle f(\mathbf{x}), f(\mathbf{x}_n) \rangle$$

Đây là các bước để cài đặt thuật toán kernel PCA:

S1: Chọn một nhân ánh xạ $k(\mathbf{x}_m, \mathbf{x}_n)$.

S2: Tính $\mathbf{K}$ dựa vào dữ liệu train $\{\mathbf{x}_n\}$, $n = 1, 2, \ldots, N$

S3: Giải quyết vấn đề chéo hóa của $\mathbf{K}$ để lấy được trị riêng λ_i và $\mathbf{a}_i$

S4: Với mỗi điểm $\mathbf{x}$, tìm thành phần chính trong không gian đặc trưng:

$\langle f(\mathbf{x}), \phi_i \rangle = \sum_{n=1}^{N} a_n^{(i)} k(\mathbf{x}, \mathbf{x}_n)$.

S5: Làm các xử lí (việc lựa chọn đặc điểm, việc chia các thành phần thành các lớp) trong không gian đặc trưng.

Một số nhân hay dùng:

* Gaussian:

$$k(\mathbf{x}_m, \mathbf{x}_n) = \exp\left[-\frac{\|\mathbf{x}_m - \mathbf{x}_n\|}{2\sigma^2} \right]$$

* Sigmoid:

$$k(\mathbf{x}_m, \mathbf{x}_n) = \tanh\left[\langle \mathbf{x}_m, \mathbf{x}_n \rangle + \theta \right]$$

* Đa thức:

$$k(\mathbf{x}_m, \mathbf{x}_n) = \langle \mathbf{x}_m, \mathbf{x}_n \rangle^k$$

Các thảo luận bên trên dựa vào giả định là các điểm dữ liệu có trung bình 0, nên covariance của hai điểm dữ liệu có cùng phổ. Trong khi các điểm dữ liệu có thể được xử lí trong không gian ban đầu bằng việc trừ vectơ trung bình từ các điểm dữ liệu, nó được thực hiện khác trong không gian đặc trưng. Việc chuẩn hoá trung bình cho các điểm dữ liệu trong không gian đặc trưng nghĩa là

$$\tilde{f}(\mathbf{x}_m) = f(\mathbf{x}_m) - \frac{1}{N}\sum_{k=1}^{N} f(\mathbf{x}_k)$$

Nhưng điều này không thể được suy ra vì ánh xạ này không được rõ ràng và $f(\mathbf{x}_k)$ không bao giờ được cho trước. Tuy nhiên, ta vẫn có thể tìm được ma trận nhân $\mathbf{K}$ cho các điểm với trung bình không $\tilde{f}(\mathbf{x})$ về mặt ma trận $\mathbf{K}$ cho $f(\mathbf{x})$

$$k_{mn} = \tilde{f}((x)_m)^T \tilde{f}(\mathbf{x}_n) = \left[f((\mathbf{x})_m) - \frac{1}{N}\sum_{k=1}^{N} f(\mathbf{x}_k)\right]^T \left[f((x)_n) - \frac{1}{N}\sum_{k=1}^{N} f(\mathbf{x}_k)\right]$$

$$= f(\mathbf{x}_m)^T f(\mathbf{x}_n) - \frac{1}{N}\sum_{k=1}^{N} f(\mathbf{x}_k)^T f(\mathbf{x}_m) - \frac{1}{N}\sum_{k=1}^{N} f(\mathbf{x}_k)^T f(\mathbf{x}_n) + \frac{1}{N^2}\sum_{k=1}^{N}\sum_{l=1}^{N} f(\mathbf{x}_k)^T f(\mathbf{x}_l)$$

$$= k_{mn} - \frac{1}{N}\sum_{k=1}^{N} k_{km} - \frac{1}{N}\sum_{k=1}^{N} k_{kn} + \frac{1}{N^2}\sum_{k=1}^{N}\sum_{l=1}^{N} k_{kl}$$

Tổng kết lại, hàm $f(\mathbf{x})$ trong thuật toán kernel PCA không bao giờ được xác định rõ ràng, và cả số chiều của không gian đặc trưng F. Giống như vậy, ma trận Σ_f và vectơ riêng ϕ_i chỉ được nhắc tới trong hướng đi trên, nhưng chúng không cần thiết được tính trong lúc cài đặt. Số đa chiều tiềm năng trong F không bị mất thêm một bước tính, vì chỉ có nhân $k(\mathbf{x}, \mathbf{x}_n)$ cần cho cài đặt.

III Independent Component Analysis (ICA)

Đây là phương pháp chọn những thành phần độc lập với nhau, và ta có n mixtures $x_1, x_2, \ldots, x_n$ của n thành phần độc lập

$$x_j = a_{j1}s_1 + a_{j2}s_2 + \cdots + a_{jn}s_n \quad \text{với mọi } j$$

Ta có thể giả định rằng biến mixture và các thành phần chính có kì vọng bằng không.

Ta có thể viết lại

$$\mathbf{x} = \mathbf{A}\mathbf{s}$$

với

$$\mathbf{x} = (x_1, x_2, \ldots, x_n)^T, \mathbf{s} = (s_1, x_2, \ldots, s_n)^T$$

$$\mathbf{A} = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix}$$

Khuôn mẫu này là khuôn mẫu ICA.

* Các giới hạn của ICA:
+ $\mathbf{s}_i$ độc lập theo thống kê.
+ Các thành phần độc lập phải có phân phối không khác với phân phối gaussian, và ta cũng giả định rằng phân phối này ta chưa biết.
+ Ma trận mixing là ma trận vuông, nhưng ta cũng có thể giả định rằng nó không phải là ma trận vuông.

Sau khi đã ước lượng được $\mathbf{A}$, ta có thể tính được các thành phần độc lập theo phương trình sau

$$\mathbf{s} = \mathbf{A}^{-1}\mathbf{x} = \mathbf{W}\mathbf{x}$$

* Có hai vấn đề chưa rõ ràng:
+ Các phương sai (năng lượng) của các thành phần độc lập $\mathbf{s}_i$ không thể được xác định.
+ Các vị trí của các thành phần độc lập không thể được xác định.
Vậy làm sao tìm ra được $\mathbf{A}$?

Có hai phương pháp phổ biến:

1. Mutual Information là nhỏ nhất.
2. Ước lượng likelihood lớn nhất.

Ta sẽ chỉ xét đến phương pháp thứ nhất: mutual Information là nhỏ nhất.
Differential Entropy được định nghĩa là:

$$H(\mathbf{y}) = -\int p(\mathbf{y}) \log p(\mathbf{y}) d\mathbf{y}$$

Negentropy J sẽ là

$$J(\mathbf{y}) = H(\mathbf{y}_{\text{gauss}}) - H(\mathbf{y})$$

với $\mathbf{y}_{\text{gauss}}$ là biến ngẫu nhiên gaussian của cùng ma trận covariance của $\mathbf{y}$. Negentropy luôn không âm, và bằng không chỉ cho biến ngẫu nhiên gaussian. Mutual information I giữa m biến ngẫu nhiên, $y_i, i = 1, 2, \ldots, m$, được định nghĩa là

$$I[y_1, y_2, \ldots, y_m] = \sum_{i=1}^{m} H(y_i) - H(\mathbf{y})$$

Mutual information cho ta biết sự liên hệ giữa các biến ngẫu nhiên. Vậy ta có thể dùng mutual information để tìm ra đại diện cho ICA. Ta có

$$\mathbf{s} = \mathbf{W}\mathbf{x}$$

mà $\mathbf{W}$ được xác định sao cho mutual information là nhỏ nhất.
Ta có

$$I[y_1, y_2, \ldots, y_m] = \sum_{i} H(y_i) - H(\mathbf{x}) - \log|\det(\mathbf{W})|$$

Ta sẽ xem xét trường hợp y_i không liên quan đến nhau và có phương sai là một. Điều này nghĩa là $E[\mathbf{y}\mathbf{y}^T] = \mathbf{W}E[\mathbf{x}\mathbf{x}^T]\mathbf{W}^T = \mathbf{I}$, suy ra

$$\det(\mathbf{I}) = 1 = \det(\mathbf{W}E[\mathbf{x}\mathbf{x}^T]\mathbf{W}^T) = \det(\mathbf{W})\det(E[\mathbf{x}\mathbf{x}^T])\det(\mathbf{W}^T)$$

Nghĩa là $\det(\mathbf{W})$ phải là hằng số. Thêm nữa, với y_i có phương sai đơn vị, entropy và negentropy khác nhau mỗi hằng số và dấu. Vì thế ta được

$$I[y_1, y_2, \ldots, y_m] = \text{const} - \sum_{i} J(y_i)$$

với const không phụ thuộc vào $\mathbf{W}$. Điều này cho ta thấy sự liên hệ giữa negentropy và mutual information.
Phương trình trên cho ta thấy để tìm $\mathbf{W}$ sao cho mutual information là nhỏ nhất tương đương với tìm một hướng để negentropy lớn nhất. Vậy phương trình này cho biết ước lượng ICA bằng cách làm cho mutual information nhỏ nhất tương đương với việc làm cho tổng của những ước lượng không gaussian của thành phần độc lập lớn nhất.

IV t-Distribution Stochastic Neighbor Embedding (t-SNE)

t-SNE là một công cụ giảm chiều không tuyến tính, thích hợp cho việc trực quan hoá dữ liệu đa chiều. Nó được dùng trong xử lí ảnh, NLP, giọng nói.

Đầu tiên xét ma trận $N \times N$ $\mathbf{P}$ trong không gian nhiều chiều có các phần tử

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

với xác xuất để x_j là neighbor của x_i

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

Ma trận $\mathbf{Q}$ $N \times N$ trong không gian thấp chiều hơn ban đầu với các phần tử có phân phối t-Student

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

Hàm đánh giá sẽ là

$$C = \sum_i \mathrm{KL}(\mathbf{P}_i|\mathbf{Q}_i) = \sum_{i=1}^{N}\sum_{i=1}^{N} p_{ij}\log\left(\frac{p_{ij}}{q_{ij}}\right)$$

Gradient của hàm đánh giá này là:

$$\begin{aligned}\frac{\delta C}{\delta y_i} &= 4\sum_{j=1,j}^{N}(p_{ij}-q_{ij})(1+\|y_i-y_j\|^2)^{-1}(y_i-y_j)\\ &= 4\sum_{j=1,j}^{N}(p_{ij}-q_{ij})q_{ij}Z(y_i-y_j)\\ &= 4\left[\sum_{j\neq i}^{N}p_{ij}q_{ij}Z(y_i-y_j)-\sum_{j\neq i}^{N}q_{ij}^2Z(y_i-y_j)\right]\\ &= 4(F_{\text{attraction}}+F_{\text{repulsion}})\end{aligned}$$

với $Z = \sum_{l,s=1,l\neq s}^{N}(1+\|y_l-y_s\|^2)^{-1}$

Thuật toán t-SNE:

Input: Tập dữ liệu $X = x_1,\ldots,x_n \in \mathbb{R}^d$, perplexity k, exaggeration parameter α, kích cỡ bước nhảy $h > 0$, số lượng $T \in \mathbb{N}$

Tính $\{p_{ij}|i,j \in [n], i \neq j\}$

Khởi tạo $y_1^0, y_2^0, \ldots, y_n^0$ i.i.d từ phân phối uniform trên $[-0.01, 0.01]^2$

for $t = 0$ to $T-1$ do

$$Z^{(t)} \leftarrow \sum_{i,j\in[n],i\neq j}\left(1+\left\|y_i^{(t)}-y_j^{(t)}\right\|^2\right)^{-1}$$

$$q_{ij}^{(t)} \leftarrow \frac{\left(1+\left\|y_i^{(t)}-y_j^{(t)}\right\|^2\right)^{-1}}{Z^{(t)}}, \forall i,j \in [n], i \neq j$$

$$y_i^{(t)} \leftarrow y_i^{(t)} + h\sum_{j\in[n]/\{i\}}\left(\alpha p_{ij}-q_{ij}^t\right)q_{ij}^t Z^t\left(y_i^{(t)}-y_j^{(t)}\right), \forall i \in [n]$$

Output: dữ liệu có số chiều thấp hơn ban đầu $Y^{(T)} = \left\{y_1^{(T)}, y_2^{(T)}, \ldots, y_n^{(T)}\right\} \in \mathbb{R}^2$

Các giả định trong thuật toán t-SNE:

* t-SNE sẽ không hoạt động tốt trong vấn đề đa chiều tổng quát, khi mà lớn hơn 2D hay 3D nhưng trung bình khoảng cách giữa các điểm cần được giữ nguyên giống như cấu trúc tổng quát
* Curse of Dimensionality
* $O(n^2)$ computational complexity
* Perplexit number, số lượng lặp, giá trị của tham số exaggeration phải được chọn thủ công.

Sự so sánh giữa PCA và t-SNE được cho trong bảng 1.

V Autoencoder - June 2021

1 Introduction

Một Autoencoder bao gồm encoder và decoder, được định nghĩa là 2 quá trình chuyển đổi ϕ và ψ:

$$\phi : X \to F$$
$$\psi : F \leftarrow X$$
$$\phi, \psi = \arg\min_{\phi,\psi}\|X-(\psi\circ\phi)X\|^2$$

Trong trường hợp đơn giản nhất, với một lớp ẩn, encoder sẽ lấy đầu vào $x \in \mathbb{R} = X$ và ánh xạ nó tới $\mathbf{h} \in \mathbb{R}^1 = F$:

$$\mathbf{h} = \sigma(\mathbf{W}*\mathbf{x}+\mathbf{b})$$

Ở đây $\mathbf{W}$ là ma trận trọng số, $\mathbf{b}$ là bias vector. $\mathbf{W}$ và $\mathbf{b}$ thường được khởi tạo ngẫu nhiên và được cập nhật đi lặp lại trong quá trình training bằng thuật toán **backpropagation**. Cuối cùng decoder $\mathbf{h}$ để tái thiết lập $\mathbf{x'}$ có cùng hình dạng với $\mathbf{x}$:

$$\mathbf{x'} = \sigma'(\mathbf{W'}*\mathbf{h}+\mathbf{b})$$

Autoencoder làm giảm thiểu hàm $\mathbf{L}$ của 2-norm:

$$\mathbf{L}(\mathbf{x},\mathbf{x'}) = \|\mathbf{x} - \mathbf{x}'\|^2 = \left\|\mathbf{x} - \sigma'(\mathbf{W}'(\sigma(\mathbf{W} * \mathbf{x} + \mathbf{b})) + \mathbf{b}')\right\|^2$$

BẢNG 1: So sánh giữa hai thuật toán PCA và t-SNE

S. NO	PCA	t-SNE
1	Là công cụ giảm số chiều tuyến tính	Là công cụ giảm số chiều không tuyến tính
2	Cố gắng bảo toàn cấu trúc tổng thể của dữ liệu	Cố gắng bảo toàn cấu trúc cục bộ (cluster) của dữ liệu
3	Không hoạt động tốt như t-SNE	Là một trong những công cụ giảm số chiều tốt nhất
4	Không liên quan đến hyperparameter	Liên quan đến các siêu tham số như perplexity, tốc độ học và số bước
5	Bị ảnh hưởng lớn bởi các outlier	Có thể xử lí outlier
6	Là thuật toán xác định	Là thuật toán không xác định hay thuật toán ngẫu nhiên
7	Hoạt động bằng cách xoay các vectơ để giữ nguyên phương sai	Hoạt động bằng cách làm cho khoảng cách giữa các điểm là nhỏ nhất
8	Ta có thể quyết định bảo toàn bao nhiêu phương sai bằng cách dùng các trị riêng	Ta không thể bảo toàn phương sai, thay vào đó ta bảo toàn khoảng cách bằng cách dùng các hyperparameter

B Mã nguồn

* Mã nguồn đã sử dụng những hàm thư viện để tính toán.
* Mục tiêu của các việc tiền xử lí này là để lọc những thông tin quan trọng, và sau đó ta có thể xử lí và lấy thông tin từ dataset.
* Đây chỉ là một mô tả ngắn gọn về mã nguồn và thông tin liên quan. Để biết thêm thông tin, xem trong các file mã nguồn.

I Principal Component Analysis (PCA)

* Ví dụ nhận diện khuôn mặt rất thú vị và đầy đủ để làm ví dụ áp dụng cho thuật toán PCA.

+ File mã nguồn: *PCA Mini-Project.ipynb*
+ Example: Faces regconition example - http://vis-www.cs.umass.edu/lfw/ - Download: http://vis-www.cs.umass.edu/lfw/lfw-funneled.tgz

II Kernel PCA

+ File mã nguồn: *Kernel PCA (another copy).ipynb*
+ Dataset: *Social_Network_Ads.csv*

III Independent Component Analysis (ICA)

* Tín hiệu âm thanh thường gồm nhiều tín hiệu gộp lại, điều này cho thấy đây là một ví dụ thích hợp để minh hoạ thuật toán ICA.

+ File mã nguồn: *Independent Component Analysis Lab.ipynb*

+ Dataset là các file âm thanh (.wav)
- Âm nhạc:

[1] Piano - The Carnival of the Animals - XIII. The Swan (Solo piano version). Performer: Markus Staab

[2] Cello - Cello Suite no. 3 in C, BWV 1009 - I. Prelude. Performer: European

IV t-Distribution Stochastic Neighbor Embedding (t-SNE)

+ File mã nguồn: *t_SNE.ipynb*
+ Dataset: Breast_cancer_wisconsin_diagnostic.csv

V Autoencoder

* Ta sử dụng dữ liệu MNIST write hand digits vì
- Dữ liệu có dạng non-linear
- Autoencoder giữ cho hàm L của 2-norm là nhỏ nhất, làm giảm chiều dữ liệu nhưng vẫn giữ lại cấu trúc toàn cục. Điều này giúp hình ảnh các chữ số dễ nhận dạng hơn so với các thuật toán làm giảm chiều dữ liệu nhưng giữ lại cấu trục cục bộ của hình ảnh.

+ File mã nguồn: *autoencoder.ipynb*
+ Dataset: MNIST write hand digits