Minería de Datos

Stochastic Neighbor Embedding (SNE)

Objetivo: Proyectar los datos a 2D o 3D para visualización.

Stochastic Neighbor Embedding (SNE)

Objetivo: Proyectar los datos a 2D o 3D para visualización.

Idea: Convertir distancias (Euclideanas) a probabilidades condicionales.

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

Parámetro del método

Stochastic Neighbor Embedding (SNE)

Objetivo: Proyectar los datos a 2D o 3D para visualización.

Idea: Convertir distancias (Euclideanas) a probabilidades condicionales.

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

Parámetro del método

Definimos la proyección tal que: $$q_{j|i} = \frac{\exp\left(-\|y_i - y_j\|^2\right)}{\sum_{k \neq i} \exp\left(-\|y_i - y_k\|^2\right)}$$

Stochastic Neighbor Embedding (SNE)

Objetivo: Proyectar los datos a 2D o 3D para visualización.

Idea: Convertir distancias (Euclideanas) a probabilidades condicionales.

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

Parámetro del método

Definimos la proyección tal que: $$q_{j|i} = \frac{\exp\left(-\|y_i - y_j\|^2\right)}{\sum_{k \neq i} \exp\left(-\|y_i - y_k\|^2\right)}$$

Notar que: $p_{i|i} = q_{i|i} = 0.$

Stochastic Neighbor Embedding (SNE)

Objetivo: Proyectar los datos a 2D o 3D para visualización.

Idea: Convertir distancias (Euclideanas) a probabilidades condicionales.

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

Parámetro del método

Definimos la proyección tal que: $q_{j|i} = \frac{\exp\left(-\|y_i - y_j\|^2\right)}{\sum_{k \neq i} \exp\left(-\|y_i - y_k\|^2\right)}$

Notar que: $p_{i|i}$ = $q_{i|i} = 0.$

Función objetivo: $C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$

Stochastic Neighbor Embedding (SNE)

Objetivo: Proyectar los datos a 2D o 3D para visualización.

Idea: Convertir distancias (Euclideanas) a probabilidades condicionales.

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

Parámetro del método

Definimos la proyección tal que: $q_{j|i} = \frac{\exp\left(-\|y_i - y_j\|^2\right)}{\sum_{k \neq i} \exp\left(-\|y_i - y_k\|^2\right)}$

Notar que: $p_{i|i}$ = $q_{i|i} = 0.$

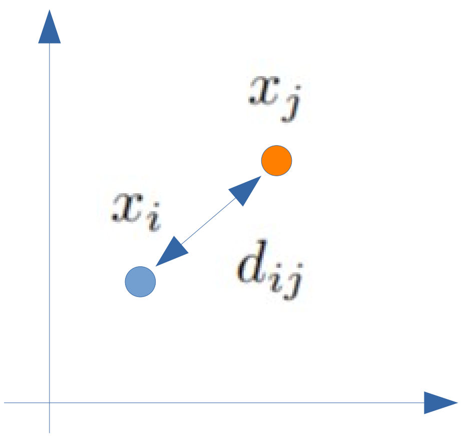
Función objetivo: $C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$

El usuario define: $Perp(P_i) = 2^{H(P_i)}$

Stochastic Neighbor Embedding (SNE)

Profundizemos para entender:

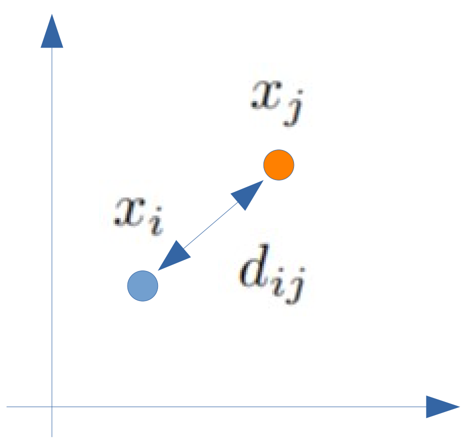
$$p_{j|i} = \frac{\exp\left(-\overbrace{\|x_i - x_j\|}^{d_{ij}}{}^2/2\sigma_i^2\right)}{\sum_{k \neq i} \exp\left(-\|x_i - x_k\|^2/2\sigma_i^2\right)}$$

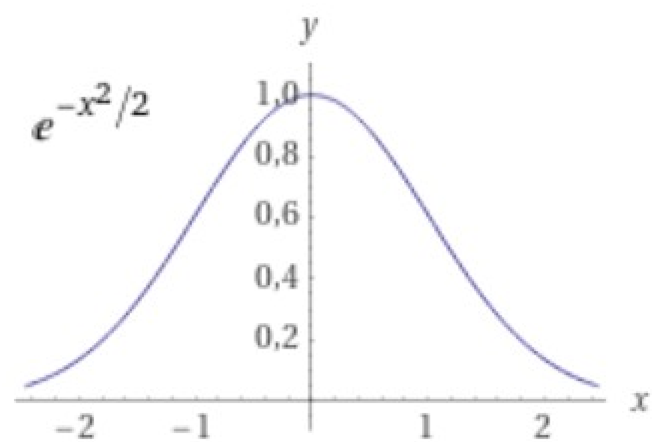


Stochastic Neighbor Embedding (SNE)

Profundizemos para entender:

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



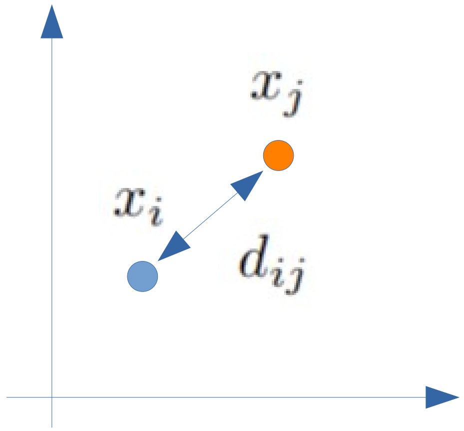


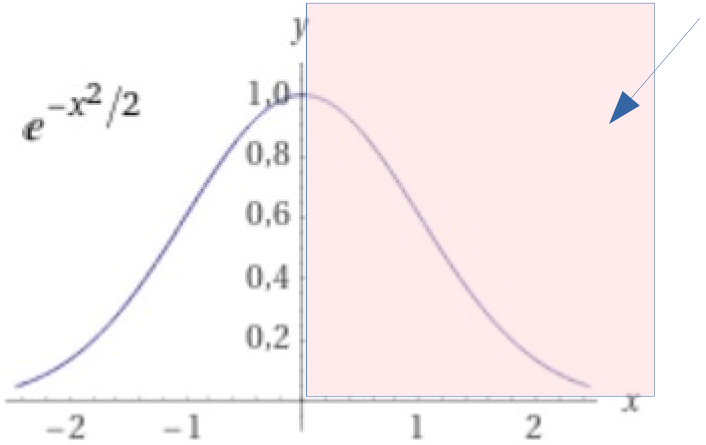
Stochastic Neighbor Embedding (SNE)

Profundizemos para entender:

d_{ij} >=0

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

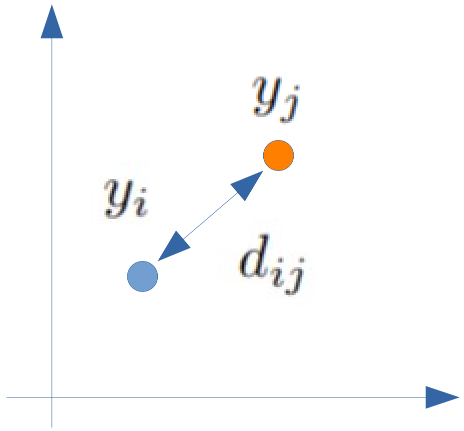


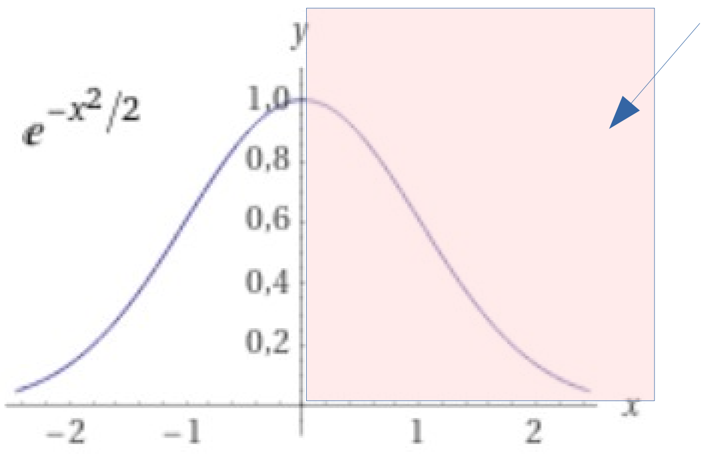


Stochastic Neighbor Embedding (SNE)

Hacemos lo mismo en un espacio de menor dimensionalidad (proyección):

$$q_{j|i} = \frac{\exp\left(-\|y_i - y_j\|^2\right)}{\sum_{k \neq i} \exp\left(-\|y_i - y_k\|^2\right)}$$





Stochastic Neighbor Embedding (SNE)

¿Cómo mido cuanto se parece el espacio original al proyectado?

Voy a comparar distribuciones de probabilidad.

Divergencia de Kullback-Leibler:

$$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

La divergencia es menor en la medida que ambas distribuciones son más parecidas.

Stochastic Neighbor Embedding (SNE)

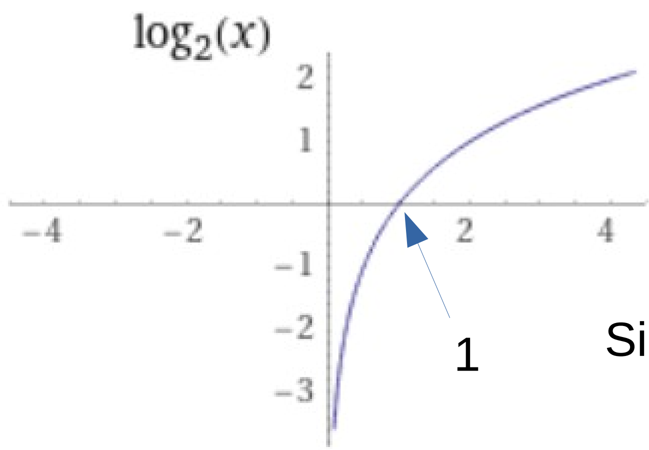
¿Cómo mido cuanto se parece el espacio original al proyectado?

Voy a comparar distribuciones de probabilidad.

<u>Divergencia de Kullback-Leibler</u>:

$$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

La divergencia es menor en la medida que ambas distribuciones son más parecidas.



Si p~q → log p/q ~ 0

Stochastic Neighbor Embedding (SNE)

Objetivo: Proyectar los datos a 2D o 3D para visualización.

Idea: Convertir distancias (Euclideanas) a probabilidades condicionales.

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

Parámetro del método

Definimos la proyección tal que: $q_{j|i} = \frac{\exp\left(-\|y_i - y_j\|^2\right)}{\sum_{k \neq i} \exp\left(-\|y_i - y_k\|^2\right)}$

Notar que: $p_{i|i} = q_{i|i} = 0.$

Función objetivo: $C = \sum_i KL(P_i || Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$

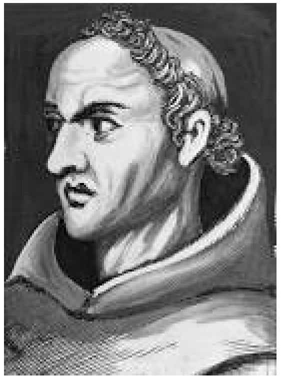
El usuario define: $Perp(P_i) = 2^{H(P_i)} \longrightarrow H(P_i) = -\sum_j p_{j|i} \log_2 p_{j|i}.$

lo cual permite determinar σ_i .

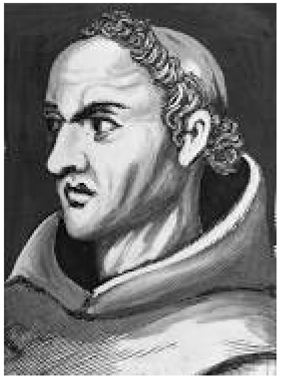
Model complexity

Principio (navaja de Ockham o principio de parsimonia)

“El modelo más simple es también el modelo más plausible”



Model complexity



Principio (navaja de Ockham o principio de parsimonia)

"El modelo más simple es también el modelo más plausible"

Una medida de complejidad: Entropía (basada en familias de objetos)

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

Explicación: entropía como medida de información.

Lanzamos una moneda 4 veces. Posibles estados del ejercicio: $2 \cdot 2 \cdot 2 \cdot 2$

estados

$\log_2(16) = 4$ ← Bits para codificar los estados

ej. CSSC

p=1/estados → $-\log_2(1/16) = 4$

Model complexity

Si los eventos no son equiprobables, debemos promediar:

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

Información codificada en el espacio original

Volvamos a SNE:

El usuario define: $Perp(P_i) = 2^{H(P_i)} \longrightarrow H(P_i) = -\sum_j p_{j|i} \log_2 p_{j|i}.$

lo cual permite determinar σ_i .

Model complexity

Si los eventos no son equiprobables, debemos promediar:

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

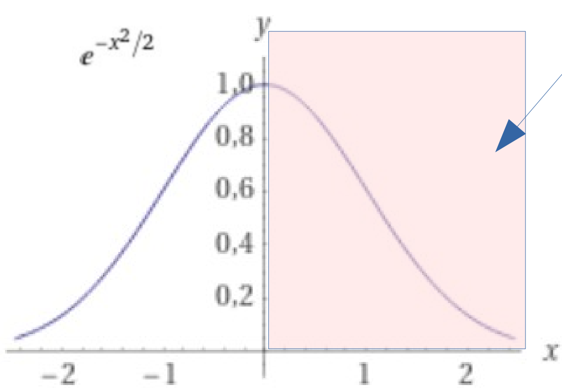
Información codificada en el espacio original

Volvamos a SNE:

El usuario define: $Perp(P_i) = 2^{H(P_i)} \longrightarrow H(P_i) = -\sum_j p_{j|i} \log_2 p_{j|i}.$

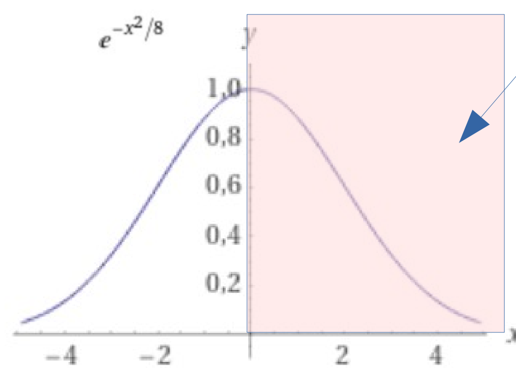
lo cual permite determinar σ_i (internamente).

Es decir, el usuario define la complejidad de la proyección, la cual es modelada en sigma!!!



Menos pares

$\sigma = 1$



Más pares

$\sigma = 2$

Model complexity

Si los eventos no son equiprobables, debemos promediar:

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

Información codificada en el espacio original

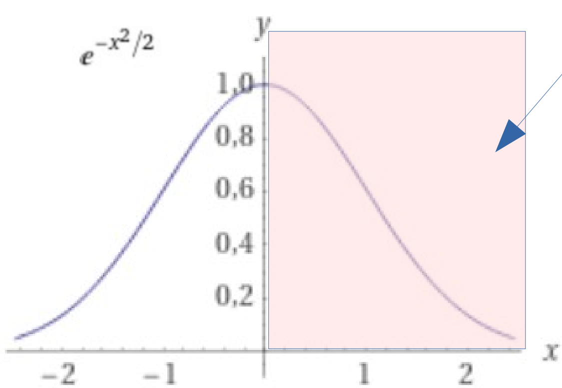
Volvamos a SNE:

Me da el # de estados promedio (vecinos de cada punto)

El usuario define: $Perp(P_i) = \boxed{2^{H(P_i)}} \longrightarrow H(P_i) = -\sum_j p_{j|i} \log_2 p_{j|i}.$

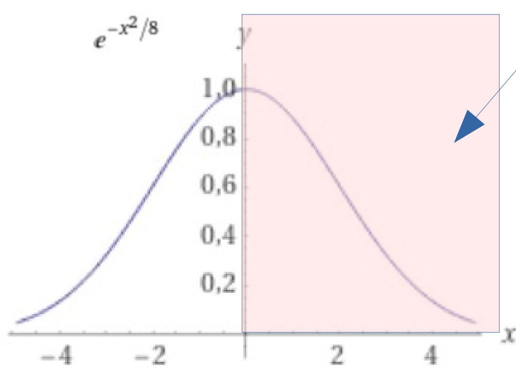
lo cual permite determinar σ_i (internamente).

Es decir, el usuario define la complejidad de la proyección, la cual es modelada en sigma!!!



Menos pares

$\sigma = 1$



Más pares

$\sigma = 2$

Stochastic Neighbor Embedding (SNE)

Minimizar: $$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}.$$

gradiente $$\frac{\delta C}{\delta y_i} = 2 \sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j).$$

Stochastic Neighbor Embedding (SNE)

Minimizar: $$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}.$$

gradiente

$$\frac{\delta C}{\delta y_i} = 2\sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j).$$

Iterativo (gradiente descendente)

$$\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t)\left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}\right)$$

Stochastic Neighbor Embedding (SNE)

Minimizar: $$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}.$$

gradiente

$$\frac{\delta C}{\delta y_i} = 2 \sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j).$$

Iterativo (gradiente descendente)

$$\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t) \left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)} \right)$$

Limitación SNE: no escala a alta dimensionalidad, no hay simetría, ...

Stochastic Neighbor Embedding (SNE)

Minimizar: $C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$.

gradiente

$$\frac{\delta C}{\delta y_i} = 2 \sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j).$$

Iterativo (gradiente descendente)

$$\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t) \left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)} \right)$$

Limitación SNE: no escala a alta dimensionalidad, no hay simetría, ...

Versión simetrizada (t-SNE): $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$

Stochastic Neighbor Embedding (SNE)

Minimizar: $$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}.$$

gradiente

$$\frac{\delta C}{\delta y_i} = 2\sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j).$$

Iterativo (gradiente descendente)

$$\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t)\left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}\right)$$

Limitación SNE: no escala a alta dimensionalidad, no hay simetría, ...

Versión simetrizada (t-SNE): $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$

$$C = KL(P||Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}.$$

Stochastic Neighbor Embedding (SNE)

Minimizar: $$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

gradiente

$$\frac{\delta C}{\delta y_i} = 2 \sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j).$$

Iterativo (gradiente descendente)

$$\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t) \left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)} \right)$$

Limitación SNE: no escala a alta dimensionalidad, no hay simetría, ...

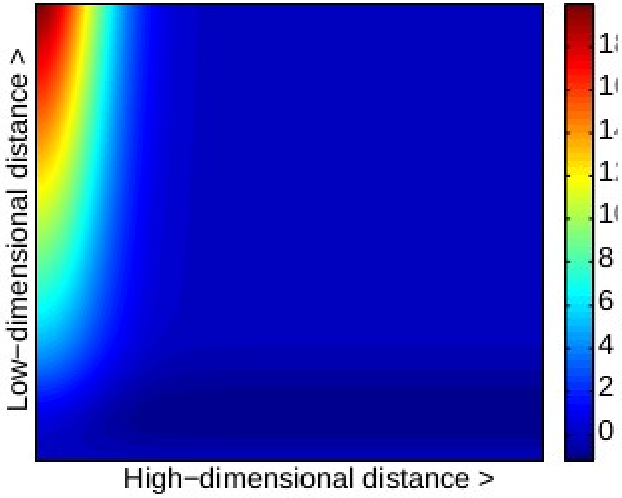
Versión simetrizada (t-SNE): $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$

$$C = KL(P||Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}.$$

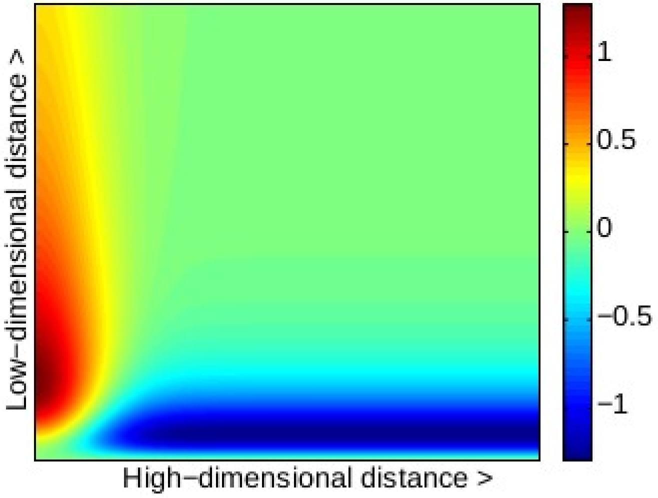
gradiente

$$\frac{\delta C}{\delta y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j)$$

Stochastic Neighbor Embedding (t-SNE)



Gradient of SNE.



Gradient of t-SNE.

Stochastic Neighbor Embedding (t-SNE)

Algorithm 1: Simple version of t-Distributed Stochastic Neighbor Embedding.

Data: data set $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$,
cost function parameters: perplexity $Perp$,
optimization parameters: number of iterations T, learning rate η, momentum $\alpha(t)$.
Result: low-dimensional data representation $\mathcal{Y}^{(T)} = \{y_1, y_2, \ldots, y_n\}$.
begin
compute pairwise affinities $p_{j|i}$ with perplexity $Perp$ (using $p_{j|i} = \frac{\exp(-\|x_i - x_j\|^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(-\|x_i - x_k\|^2 / 2\sigma_i^2)}$
set $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$
sample initial solution $\mathcal{Y}^{(0)} = \{y_1, y_2, \ldots, y_n\}$ from $\mathcal{N}(0, 10^{-4} I)$
Puede inicializarse con PCA
for $t=1$ **to** T **do**
compute low-dimensional affinities q_{ij} (using $q_{j|i} = \frac{\exp(-\|y_i - y_j\|^2)}{\sum_{k \neq i} \exp(-\|y_i - y_k\|^2)}$
compute gradient $\frac{\delta C}{\delta \mathcal{Y}}$ (using $\frac{\delta C}{\delta y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j)$
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

Stochastic Neighbor Embedding (t-SNE)

- Implementaciones:

    - Python: sklearn

https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html