



Conceptos generales de aprendizaje no supervisado

Curso de aprendizaje automático para
IGMAT el INE

Víctor Gallego y Roi Naveiro

2019-03-31

Introducción

Aprendizaje Supervisado

- Espacio de las muestras de entrada: \mathcal{X}
- Espacio de las salidas: \mathcal{Y}

Datos:

- Conjunto de **entrenamiento**: $S = \{x_i, y_i\}_{i=1}^N$, con $x_i, y_i \in \mathcal{X} \times \mathcal{Y}$
- Visión probabilística: $x_i, y_i \sim P(X, Y)$

Objetivo:

- Aprender una regla de predicción (hipótesis), $h : \mathcal{X} \rightarrow \mathcal{Y}$
- Visión probabilística: estimar $P(Y|X)$

Aprendizaje Supervisado

Estrategia básica:

- MLE de algún modelo paramétrico

$$\arg \max_w \prod_{i=1}^N P(y_i | x_i, w)$$

Facilidades:

- \mathcal{Y} es tiene dimensión baja
- Es sencillo cuantificar el error: natural definir función de coste. Error = valor esperado de coste bajo $P(X, Y)$.

Aprendizaje No Supervisado

Datos:

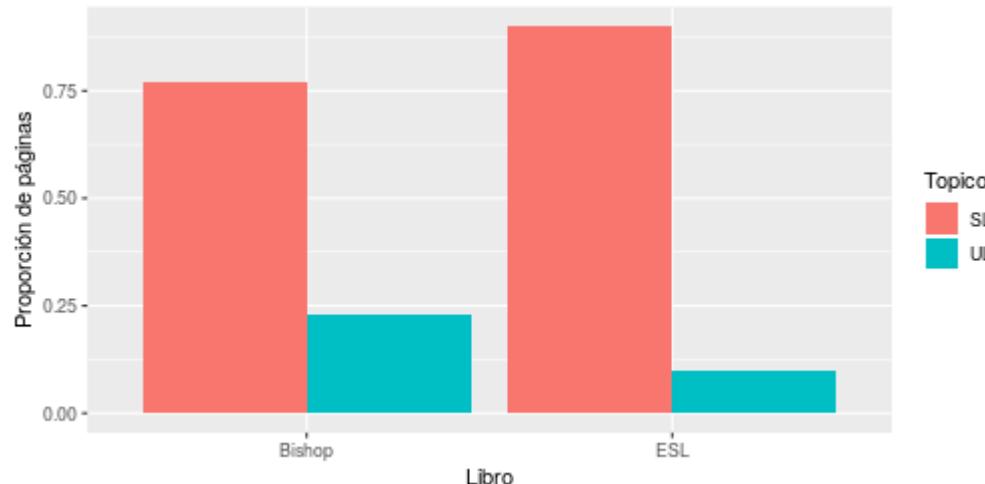
- No hay salidas: $S = \{x_i\}_{i=1}^N$, con $x_i \in \mathcal{X}$
- Visión probabilística: $x_i \sim P(X)$

Objetivo:

- Estimar $P(X)$
- Inferir alguna propiedad de $P(X)$
- Muestrear de $P(X)$

Retos del Aprendizaje No Supervisado

- X generalmente es de alta dimensión (piensa en imágenes: $128 \times 128 \times 3 = 49152$)
- Propiedades de interés que queremos inferir son más complejas que simples parámetros
- No hay una medida directa de cuantificar el error
- Métodos heurísticos no solo para motivar los algoritmos sino también para medir la calidad de los resultados



Buen proxy de la dificultad de cada área !!

Una taxonomía de algoritmos de aprendizaje no supervisado según su objetivo

- Métodos de estimación de densidades
- Manifold learning: PCA, PCA no lineal, self-organizing maps, modelos de variables latentes, ...
- Encontrar regiones convexas del espacio que contengan modas de $P(X)$: análisis de cluster, modelos de mixturas, ...
- Muestrear de $P(X)$: GAN, autoencoders, autoencoders variacionales, ...

Repaso Álgebra Lineal

Aplicaciones lineales

- Dado $\mathbf{x} \in \mathbb{R}^N$, una **aplicación (función) lineal** $f : \mathbb{R}^N \rightarrow \mathbb{R}^M$ se expresa como

$$f(\mathbf{x}) = W\mathbf{x}$$

donde W es una matriz de tamaño $M \times N$.

- Para el caso $M = N$, los **autovalores** $\lambda \in \mathbb{R}$ y los **autovectores** $\mathbf{v} \in \mathbb{R}^M$ son los elementos que cumplen

$$W\mathbf{v} = \lambda\mathbf{v}$$

- Si los vectores columna de $W = [w_1, \dots, w_M]$ son ortonormales (esto es, $w_i^\top w_j = 0$, $w_i^\top w_i = 1$), se dice que W es una proyección ortonormal. En este caso, los vectores $[w_1, \dots, w_M]$ forman una base ortonormal.

Derivadas matriciales

- Será necesario considerar derivadas de vectores respecto a escalares. En este caso,

$$\left(\frac{\partial \mathbf{a}}{\partial x} \right)_i = \frac{\partial \mathbf{a}_i}{\partial x}$$

- También podemos derivar respecto a vectores o matrices:

$$\left(\frac{\partial x}{\partial \mathbf{a}} \right)_i = \frac{\partial x}{\partial \mathbf{a}_i}, \quad \left(\frac{\partial \mathbf{a}}{\partial \mathbf{b}} \right)_{i,j} = \frac{\partial \mathbf{a}_i}{\partial \mathbf{b}_j}$$

- *Ejercicio.* Probar que

$$\frac{\partial \mathbf{x}^\top \mathbf{a}}{\partial \mathbf{x}} = \frac{\partial \mathbf{a}^\top \mathbf{x}}{\partial \mathbf{x}} = \mathbf{a}$$

y que

$$\frac{\partial \mathbf{A}\mathbf{B}}{\partial x} = \frac{\partial \mathbf{A}}{\partial x} \mathbf{B} + \frac{\partial \mathbf{B}}{\partial x} \mathbf{A}.$$

Optimización

- Queremos optimizar una función diferenciable $f(\boldsymbol{x})$ tal que $f : \mathbb{R}^N \rightarrow \mathbb{R}$. Los **óptimos locales** verifican

$$\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}} = 0$$

- En el caso de querer optimizar $f(\boldsymbol{x})$ sujeto a además a una restricción $g(\boldsymbol{x}) = 0$, podemos utilizar el **Teorema de los multiplicadores de Lagrange** y optimizar la siguiente función objetivo (ya sin restricciones):

$$f(\boldsymbol{x}) + \lambda g(\boldsymbol{x})$$

Métodos Lineales reducción de dimensionalidad

Análisis de Componentes Principales

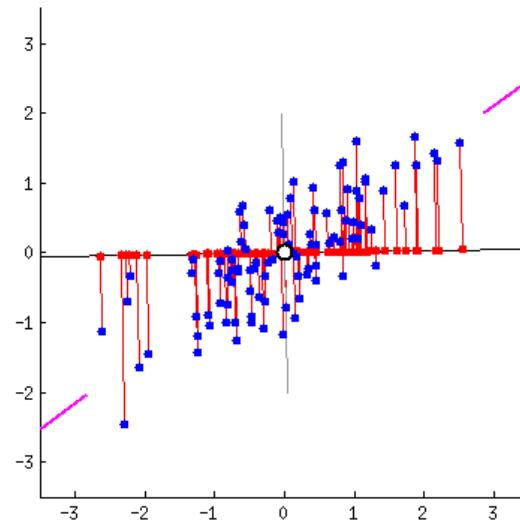
Dos definiciones alternativas

- Proyección ortogonal de datos a subespacio de dimensión inferior tal que varianza de proyecciones es máxima
- Proyección lineal que minimiza el *coste medio de proyección* = distancia media cuadrática entre datos y sus proyecciones
- Ambos dan lugar al mismo algoritmo!
- Diferentes aplicaciones: reducción de dimensionalidad, compresión, visualización de datos, extracción de variables predictoras...

Formulación por Máxima Varianza

PCA: Formulación por Máxima Varianza (1)

- Dados: $x_n \in \mathbb{R}^D$, $n = 1, \dots, N$
- Objetivo: encontrar proyección lineal $\pi : \mathbb{R}^D \rightarrow \mathbb{R}^M$ tal que $M < D$ y se maximice la varianza de los datos proyectados.
- Ejemplo $\mathbb{R}^2 \rightarrow \mathbb{R}^1$:



PCA: Formulación por Máxima Varianza (2)

- Empezamos considerando proyección a \mathbb{R} ($M = 1$).
- Una proyección viene representada por su dirección, esto es, un vector $\mathbf{u}_1 \in \mathbb{R}^D$. Como sólo nos interesa la dirección, imponemos $\mathbf{u}_1^\top \mathbf{u}_1 = 1$.
- $\mathbf{u}_1^\top \mathbf{x}_n$ es la proyección del n-ésimo punto.
- También nos interesa calcular:
 - La media de los datos proyectados

$$\frac{1}{N} \sum_{n=1}^N \mathbf{u}_1^\top \mathbf{x}_n = \mathbf{u}_1^\top \bar{\mathbf{x}}$$

- La varianza de los datos proyectados

$$\frac{1}{N} \sum_{n=1}^N (\mathbf{u}_1^\top \mathbf{x}_n - \mathbf{u}_1^\top \bar{\mathbf{x}})^2 = \mathbf{u}_1^\top \mathbf{S} \mathbf{u}_1$$

PCA: Formulación por Máxima Varianza (3)

- Ahora ya podemos plantear un problema de optimización, con objetivo:

$$\max_{\mathbf{u}_1} \mathbf{u}_1^\top \mathbf{S} \mathbf{u}_1$$

- con la restricción:

$$\mathbf{u}_1^\top \mathbf{u}_1 = 1$$

- Para resolverlo, utilizamos la formulación Lagrangiana, con lo que lo convertimos al siguiente problema de optimización sin restricciones:

$$\max_{\mathbf{u}_1} \mathbf{u}_1^\top \mathbf{S} \mathbf{u}_1 + \lambda_1 (\mathbf{u}_1^\top \mathbf{u}_1 - 1)$$

- Derivamos...

PCA: Formulación por Máxima Varianza (4)

- Queda que

$$\mathbf{S}\mathbf{u}_1 = \lambda_1 \mathbf{u}_1$$

es decir, \mathbf{u}_1 es *autovector de la matriz de covarianzas \mathbf{S} .*

- Más aún,

$$\mathbf{u}_1^\top \mathbf{S} \mathbf{u}_1 = \lambda_1$$

la *varianza es precisamente el mayor autovalor*

- El autovector \mathbf{u}_1 asociado al mayor autovalor, λ_1 es conocido como *primera componente principal.*

Minimización de Error de Proyección

PCA: Minimización de Error de Proyección (1)

- Considérese el conjunto de observaciones $\{x_n\}_{n=1}^N$, donde $x_n \in \mathbb{R}^D$
- $\{u_i\}_{i=1}^D$: base ortonormal completa de dimension D

$$x_n = \sum_{i=1}^D \alpha_{ni} u_i$$

- Sin pérdida de generalidad

$$x_n = \sum_{i=1}^D (x_n^\top u_i) u_i$$

- Interés: aproximar dato usando representación que requiera $M < D$ parámetros.

PCA: Minimización de Error de Proyección (2)

- Representamos el subespacio de dimensión M con los primeros M vectores de la base

$$\tilde{x}_n = \sum_{i=1}^M (z_{ni} u_i) + \sum_{i=M+1}^D b_i u_i$$

- Escogemos $\{z_{in}\}$, $\{b_i\}$ y $\{u_i\}$ para distorsión introducida por reducción de dimensión

$$J = \frac{1}{N} \sum_{n=1}^N \|x_n - \tilde{x}_n\|^2$$

- Minimizando respecto $\{z_{in}\}$

$$z_{nj} = x_n^\top u_j$$

- Minimizando respecto $\{b_i\}$

$$b_j = \left(\frac{1}{N} \sum_{n=1}^N x_n^\top \right)^\top u_j = \bar{x}^\top u_j$$

PCA: Minimización de Error de Proyección (3)

- Substituyendo en la expresión de \tilde{x}_n

$$x_n - \tilde{x}_n = \sum_{i=M+1}^D \{(x_n - \bar{x})^\top u_i\} u_i$$

- Vector desplazamiento ortogonal al *subespacio principal*. Substituendo en J

$$J = \frac{1}{N} \sum_{n=1}^N \sum_{i=M+1}^D (x_n^\top u_i - \bar{x}^\top u_i)^2 = \sum_{i=M+1}^D u_i^\top S u_i$$

Donde $S = \frac{1}{N} \sum_{i=1}^N (x_n - \bar{x})(x_n - \bar{x})^\top$.

- Falta minimizar respecto de $\{u_i\}$, sujeto a $u_i^\top u_i = 1$

PCA: Minimización de Error de Proyección (4)

- Intuición: $D = 2$ y $M = 1$: encuentra u_2 que minimice $J = u_2^\top S u_2$, sujeto a $u_2^\top u_2 = 1$.

$$\tilde{J} = u_2^\top S u_2 + \lambda_2(1 - u_2^\top u_2)$$

- Derivando e igualando a 0: $S u_2 = \lambda_2 u_2 \Rightarrow$ todo autovector define un punto estacionario.
- En el mínimo $J = \lambda_2$: escogemos u_2 con autovalor mínimo. Luego **subespacio principal** definido por autovectores de autovalor máximo.

PCA: Minimización de Error de Proyección (5)

- Solución general: escoger como $\{u_i\}$ los autovectores de la matriz de covarianza

$$Su_i = \lambda_i u_i$$

- El valor de distorsión es entonces $J = \sum_{i=M+1}^D \lambda_i$.
- J será mínimo si escogemos los $D - M$ autovectores de menor autovalor.
- Los autovectores definiendo el subespacio principal, serán los de mayor autovalor.

Aplicaciones de PCA

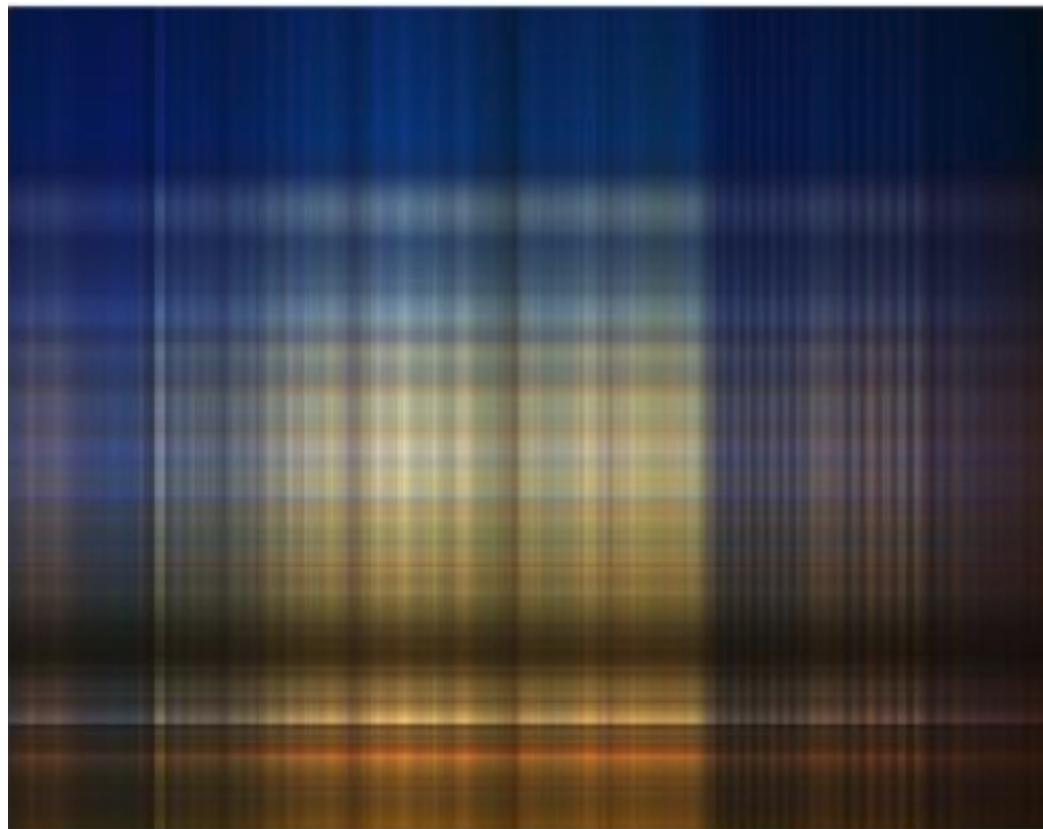
Aplicación: compresión de datos

- Cada punto de dimensión D se representa como vector de dimensión M

$$\tilde{x}_n = \bar{x} + \sum_{i=1}^M (x_n^\top - \bar{x}^\top u_i) u_i$$

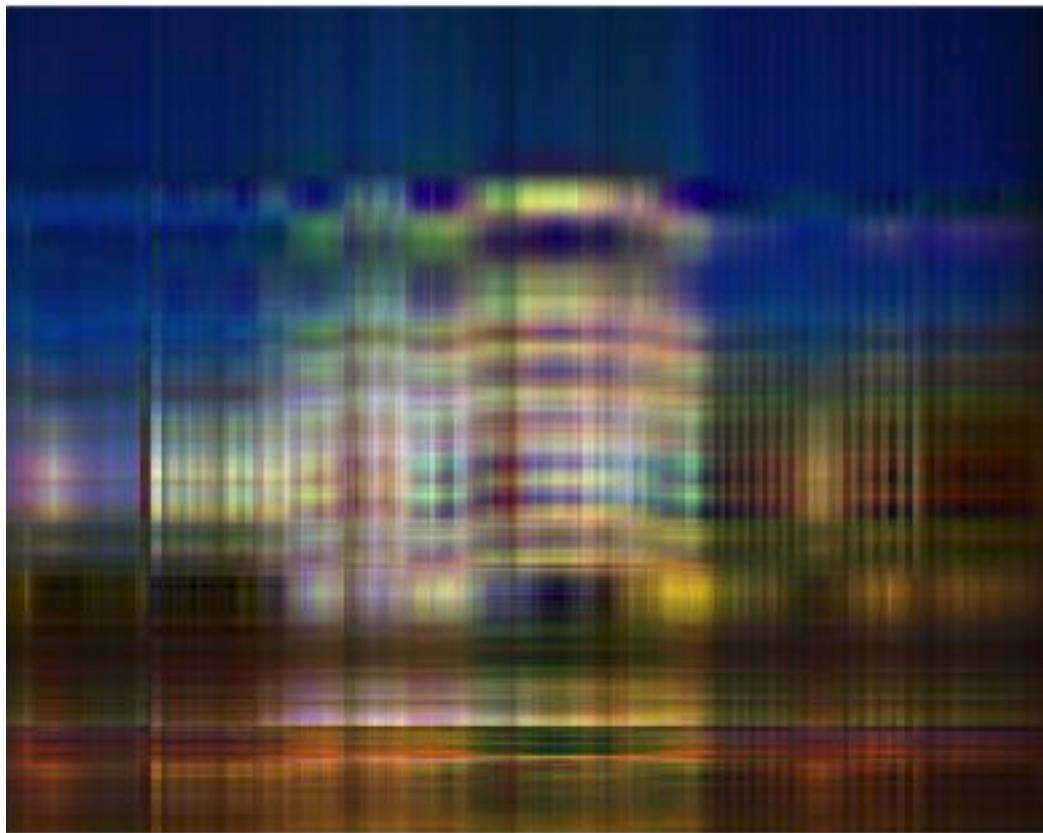
Aplicación: compresión de datos

- $M = 1$



Aplicación: compresión de datos

- $M = 3$



Aplicación: compresión de datos

- $M = 10$



Aplicación: compresión de datos

- $M = 20$



Aplicación: compresión de datos

- $M = 50$



Aplicación: compresión de datos

- $M = 200$



Aplicación: visualización de datos

- Es conveniente realizarla antes de elegir el modelo predictivo, para tener una idea de cómo es la estructura de los datos.
- Representar los datos directamente es fácil cuando están en 2D ó 3D.
- ¿Cómo hacerlo cuando $D \gg 3$? Situación habitual:
 - MNIST: $D = 28 \times 28$.
 - CIFAR10: $D = 32 \times 32 \times 3$.
- Con PCA: $Z = XW^\top$ donde
 1. W es $M \times D$.
 2. X es la matriz de datos $N \times D$.

Ejemplo práctico

- Código en *exercises/03-unsupervised/src/tSNE_coil20R.R*.
- Base de datos COIL20: imágenes de 20 objetos desde 72 ángulos diferentes.



Ejemplo práctico

- Código en *exercises/03-unsupervised/src/tSNE_coil20R.R*.
- Base de datos COIL20: imágenes de 20 objetos desde 72 ángulos diferentes.



Ejemplo práctico

- Código en *exercises/03-unsupervised/src/tSNE_coil20R.R*.
- Base de datos COIL20: imágenes de 20 objetos desde 72 ángulos diferentes.



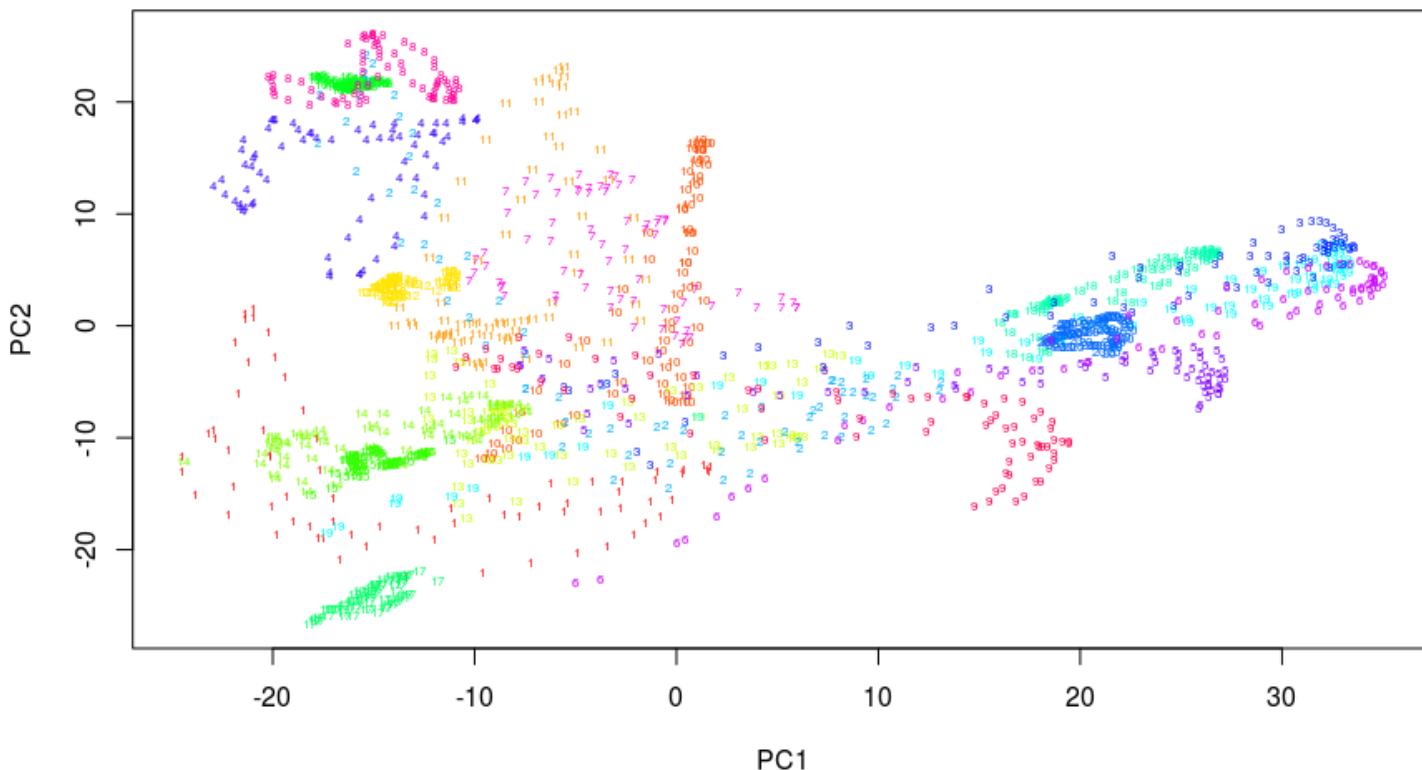
Ejemplo práctico

- Código en *exercises/03-unsupervised/src/tSNE_coil20R.R*.
- Base de datos COIL20: imágenes de 20 objetos desde 72 ángulos diferentes.



Ejemplo práctico

- Proyección a 2D mediante PCA:



Cuestiones de implementación

Datos de alta dimensionalidad

- En muchos casos $D > N$, por ejemplo imágenes: $|D| = \text{ancho} \times \text{alto} \times 3$.
- La complejidad de calcular los autovectores de una matriz $D \times D$ escala según $\mathcal{O}(D^3)$.
- N puntos en un espacio de dimensión $D > N$ forman un subespacio de dimensión $N - 1$ (¡o menos!).
- No tiene sentido aplicar PCA con $M > N - 1$: saldrán autovalores 0.

Datos de alta dimensionalidad

- Consideramos $\mathbf{X} \in \mathbb{R}^{N \times D}$ cuya fila n-ésima es $x_n - \bar{x}$.
- La matriz de covarianzas es $\mathbf{S} = N^{-1} \mathbf{X}^\top \mathbf{X}$, luego obtenemos autovectores mediante

$$\frac{1}{N} \mathbf{X}^\top \mathbf{X} \mathbf{u}_i = \lambda_i \mathbf{u}_i$$

- Multiplicando ambos miembros por la izquierda por \mathbf{X} llegamos a

$$\frac{1}{N} \mathbf{X} \mathbf{X}^\top (\mathbf{X} \mathbf{u}_i) = \lambda_i (\mathbf{X} \mathbf{u}_i)$$

con lo que $\mathbf{v}_i = \mathbf{X} \mathbf{u}_i$ es un autovector de la matriz $N^{-1} \mathbf{X} \mathbf{X}^\top$, de tamaño $N \times N$.

- ¡La complejidad ahora es $\mathcal{O}(N^3)$!

Datos de alta dimensionalidad

- Pero tenemos que obtener los autovectores en el espacio original...
- Multiplicando ahora por \mathbf{X}^\top :

$$\left(\frac{1}{N}\mathbf{X}^\top\mathbf{X}\right)(\mathbf{X}^\top\mathbf{v}_i) = \lambda_i(\mathbf{X}^\top\mathbf{v}_i)$$

- Con lo que $\mathbf{X}^\top\mathbf{v}_i$ es autovector de \mathbf{S} con mismo autovalor λ_i .

En resumen:

1. Calculamos autovectores \mathbf{v}_i .
2. $\mathbf{u}_i = \mathbf{X}^\top\mathbf{v}_i$ y normalizamos \mathbf{u}_i .
3. En concreto,

$$\mathbf{u}_i = \frac{1}{(N\lambda_i)^2}\mathbf{X}^\top\mathbf{v}_i$$

Análisis de Componentes Principales Probabilístico

PCA Probabilístico

- PCA = solución de máxima verosimilitud de modelo probabilístico de variables latentes.
- Permite tratamiento natural de datos ausentes.
- Permite la formulación Bayesiana en la que la dimensión del subespacio principal puede ser aprendida de los datos.
- Permite modelizar densidades condicionadas a clases y por tanto clasificar.
- Puede generar muestras de la distribución de interés.

PPCA - Modelo Generativo

- Idea: explicar cómo los datos observados se han generado a partir de variables latentes.
- Cada dato observado \mathbf{x} se ha generado de esta manera:
 1. Se muestrea la variable latente $\mathbf{z} \sim \mathcal{N}(\mathbf{z}|0, \mathbf{I})$.
 2. $\mathbf{x} = \mathbf{W}\mathbf{z} + \boldsymbol{\mu} + \boldsymbol{\epsilon}$. Donde $\boldsymbol{\epsilon}$ sigue una distribución normal de media 0 y covarianza $\sigma^2\mathbf{I}$.
- Ahora, supongamos que queremos determinar \mathbf{W} , $\boldsymbol{\mu}$ y σ^2 usando máxima verosimilitud. Necesitamos escribir la distribución marginal $p(\mathbf{x})$.

$$p(\mathbf{x}) = \int p(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z}$$

- Como estamos ante un modelo lineal-Gaussiano, la marginal seguirá una distribución normal con

$$\begin{aligned}\mathbb{E}[\mathbf{x}] &= \mathbb{E}[\mathbf{W}\mathbf{z} + \boldsymbol{\mu} + \boldsymbol{\epsilon}] = \boldsymbol{\mu} \\ \text{cov}[\mathbf{x}] &= \mathbb{E}[(\mathbf{W}\mathbf{z} + \boldsymbol{\epsilon})(\mathbf{W}\mathbf{z} + \boldsymbol{\epsilon})^\top] = \mathbb{E}[\mathbf{W}\mathbf{z}\mathbf{z}^\top\mathbf{W}^\top] + \mathbb{E}[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top] = \mathbf{W}\mathbf{W}^\top + \sigma^2\mathbf{I} = \mathbf{C}\end{aligned}$$

PPCA - Solución de máxima verosimilitud

- Dado un conjunto de datos observados $(X) = \{\mathbf{x}_n\}$, la log-verosimilitud viene dada por

$$\begin{aligned}\log p(\mathbf{X}|\mathbf{W}, \boldsymbol{\mu}, \sigma^2) &= \sum_{n=1}^N \log p(\mathbf{x}_n|\mathbf{W}, \boldsymbol{\mu}, \sigma^2) \\ &= -\frac{ND}{2} \log(2\pi) - \frac{N}{2} \log(|\mathbf{C}|) - \frac{1}{2} \sum_{n=1}^N (\mathbf{x}_n - \boldsymbol{\mu})^\top \mathbf{C}^{-1} (\mathbf{x}_n - \boldsymbol{\mu})\end{aligned}$$

- Tipping and Bishop, **Probabilistic principal component analysis** resuelven el problema de optimización.

$$\begin{aligned}\boldsymbol{\mu} &= \bar{\mathbf{x}} \\ \mathbf{W}_{ML} &= \mathbf{U}_M (\mathbf{L}_M - \sigma^2 \mathbf{I})^{1/2} \mathbf{R} \\ \sigma_{ML}^2 &= \frac{1}{D-M} \sum_{i=M+1}^D \lambda_i\end{aligned}$$

PPCA - Recuperando PCA

- PCA: proyección de puntos de un espacio D -dimensional a uno M -dimensional.
- PPCA: al revés. Para aplicaciones, invertimos esta proyección usando el teorema de Bayes.
- Cualquier punto \mathbf{x} , puede ser resumido usando media y covarianza a posteriori.

$$\begin{aligned}\mathbb{E}[\mathbf{z}|\mathbf{x}] &= \mathbf{M}^{-1} \mathbf{W}_{ML}^\top (\mathbf{x} - \bar{\mathbf{x}}) \\ \text{cov}[\mathbf{z}|\mathbf{x}] &= \sigma^2 \mathbf{M}^{-1}\end{aligned}$$

con $\mathbf{M} = \mathbf{W}^\top \mathbf{W} + \sigma^2 \mathbf{I}$.

- En el límite $\sigma^2 \rightarrow 0$, la media a posteriori representa una proyección ortogonal del punto al espacio latente y la covarianza es cero, por tanto la densidad es singular, recuperando PCA.
- **IMPORTANTE:** PPCA permite definir una distribución Gaussiana multivariante en la que el número de grados de libertad, puede ser controlado y al mismo tiempo capturar correlaciones en los datos.

Métodos Lineales reducción de dimensionalidad

Factorización de matrices no negativas

NMF - Algoritmo

- Sea \mathbf{X} la matriz $N \times p$ de observaciones. Buscamos aproximarla por

$$\mathbf{X} \simeq \mathbf{WH}$$

- \mathbf{W} matrix $N \times r$ y \mathbf{H} matriz $r \times p$. \mathbf{X} , \mathbf{W} y \mathbf{H} tiene todos sus elementos no negativos.
- \mathbf{W} y \mathbf{H} son tales que minimizan alguna función de coste.
- Tantos algoritmos diferentes como funciones de coste. Dos comunes:

1. Norma de Frobenius

$$\|\mathbf{X} - \mathbf{WH}\|^2 = \sum_{i=1}^N \sum_{j=1}^p (\mathbf{X}_{ij} - [\mathbf{WH}]_{ij})^2$$

2. Divergencia Kullback-Leibler

$$D(\mathbf{X} \parallel \mathbf{WH}) = \sum_{i=1}^N \sum_{j=1}^p \left(\mathbf{X}_{ij} \log \frac{\mathbf{X}_{ij}}{[\mathbf{WH}]_{ij}} - \mathbf{X}_{ij} + [\mathbf{WH}]_{ij} \right)$$

Aquí se explica cómo resolver los problemas de optimización correspondientes.

Ejercicio

Demuéstrese que encontrar \mathbf{WH} que minimizan la *Divergencia Kullback-Leibler*, equivale a maximizar la log-verosimilitud de un modelo que asume $\mathbf{X}_{ij} \sim \text{Po}([\mathbf{WH}]_{ij})$. Es decir, \mathbf{X}_{ij} sigue una distribución de Poisson de media $[\mathbf{WH}]_{ij}$.

NMF - Sistemas de Recomendación

- Muchos usos: sistemas de recomendación, minería de textos, reducción de dimensionalidad
- Ejemplo: **Sistemas de recomendación.**

$$\begin{matrix} & \text{C1} & \text{C2} & \text{C3} & \text{C4} & \text{C5} \\ \text{P1} & | & | & | & | & | \\ \text{P2} & | & | & | & | & | \\ \text{P3} & | & | & | & | & | \\ \text{P4} & | & | & | & | & | \end{matrix} = \begin{matrix} & \text{S1} & \text{S2} \\ \text{P1} & | & | \\ \text{P2} & | & | \\ \text{P3} & | & | \\ \text{P4} & | & | \end{matrix} \times \begin{matrix} & \text{S1} & \text{S2} \\ \text{C1} & | & | & | & | & | \\ \text{C2} & | & | & | & | & | \\ \text{C3} & | & | & | & | & | \\ \text{C4} & | & | & | & | & | \\ \text{C5} & | & | & | & | & | \end{matrix}$$

- Cada elemento de **X** es número de compras que el cliente ha realizado del producto.
- Cada columna de **W** define un segmento. Cuanto mayor es el *peso* de un producto en el segmento, más determinado está este segmento por el producto.
- Las columnas de **H** asignan a cada cliente pesos de pertenencia a cada segmento.
- Cada cliente está descrito por una combinación lineal de segmentos, con coeficientes dados por las columnas de **H**.

NMF - Sistemas de Recomendación

- Cada cliente se genera como combinación de variables ocultas (segmentos). NMF genera estas variables.
- El analista debe interpretar los segmentos
- **¿Cómo recomendar?**
 1. Reconstruir la matrix \mathbf{X} .
 2. Para un cliente dado, recomendar productos con mayor peso.
 3. Para un producto dado, recomendar a los clientes que mayor peso dan al producto.

¿Cómo usarías la Factorización No Negativa de Matrices en problemas de minería de textos?

Métodos no lineales reducción de dimensionalidad

Métodos no lineales

- PCA realiza una transformación lineal a los datos: $z_n = \mathbf{W}x_n$.
- ¿Cómo podemos extenderlo de forma no lineal?
- ¡Apilando múltiples capas!

$$z_n^{(1)} = \sigma(\mathbf{W}^{(i)}x_n)$$

...

$$z_n^{(i+1)} = \sigma(\mathbf{W}^{(i)}z_n^{(i)})$$

donde σ es una función no lineal (por ejemplo $\sigma(z) = \max\{0, z\}$).

- Es la base de los *autoencoders* (autocodificadores), uno de los bloques principales del *deep learning* (aprendizaje profundo).
- Ejemplo de autoencoders en Keras

t-distributed Stochastic Neighbor Embedding (tSNE)

- Usaremos técnicas de reducción de dimensionalidad, concretamente aquellas que:
 1. Preserven distancias.
 2. Preserven topologías.
- Problema original: encontrar una transformación (no lineal)

$$\mathcal{X} := \{x_1, \dots, x_N \in \mathbb{R}^D\} \rightarrow \mathcal{Y} := \{y_1, \dots, y_N \in \mathbb{R}^M\}$$

de forma que ambas distribuciones *se parezcan* lo más posible

$$\min_{\mathcal{Y}} C(\mathcal{X}, \mathcal{Y})$$

- donde C será una **divergencia** (mide similaridad entre distribuciones).

Stochastic Neighbor Embedding (SNE)

- SNE convierte **distancias euclídeas** en **similaridades**, que pueden ser interpretadas como probabilidades:

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

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

- Las distribuciones de los vecinos del punto i son $P_i = \{p_{1|i}, \dots, p_{N|i}\}$ y $Q_i = \{q_{1|i}, \dots, q_{N|i}\}$
- Optimizamos la divergencia de Kullback-Leiber

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

De SNE a tSNE

- SNE simétrico: optimizamos más eficientemente $C = KL(P||Q)$ definiendo:

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

$$q_{ij} = \frac{\exp\{-||y_i - y_j||^2\}}{\sum_{k \neq j} \exp\{-||y_i - y_k||^2\}}$$

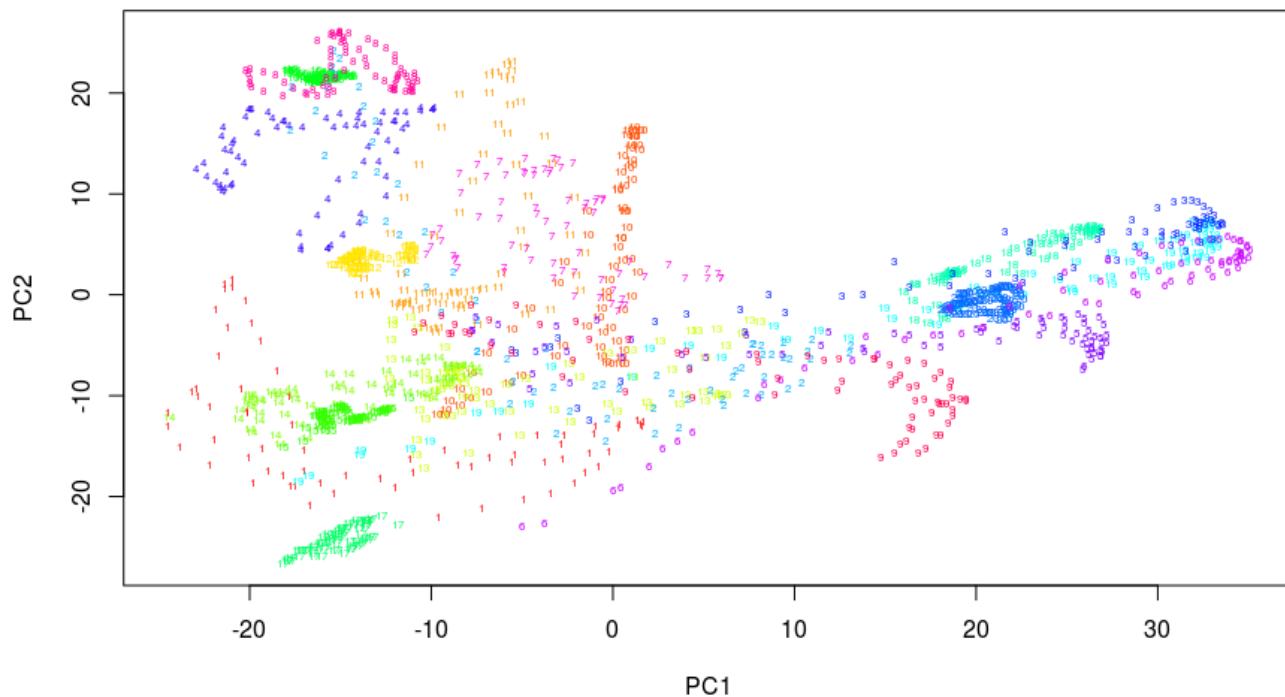
- tSNE: en lugar de kernel Gaussiano usamos t-Student en el espacio \mathcal{Y} :

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

(la t-Student tiene colas más pesadas que la Normal, evitando que los puntos se colapsen mucho en el espacio \mathcal{Y})

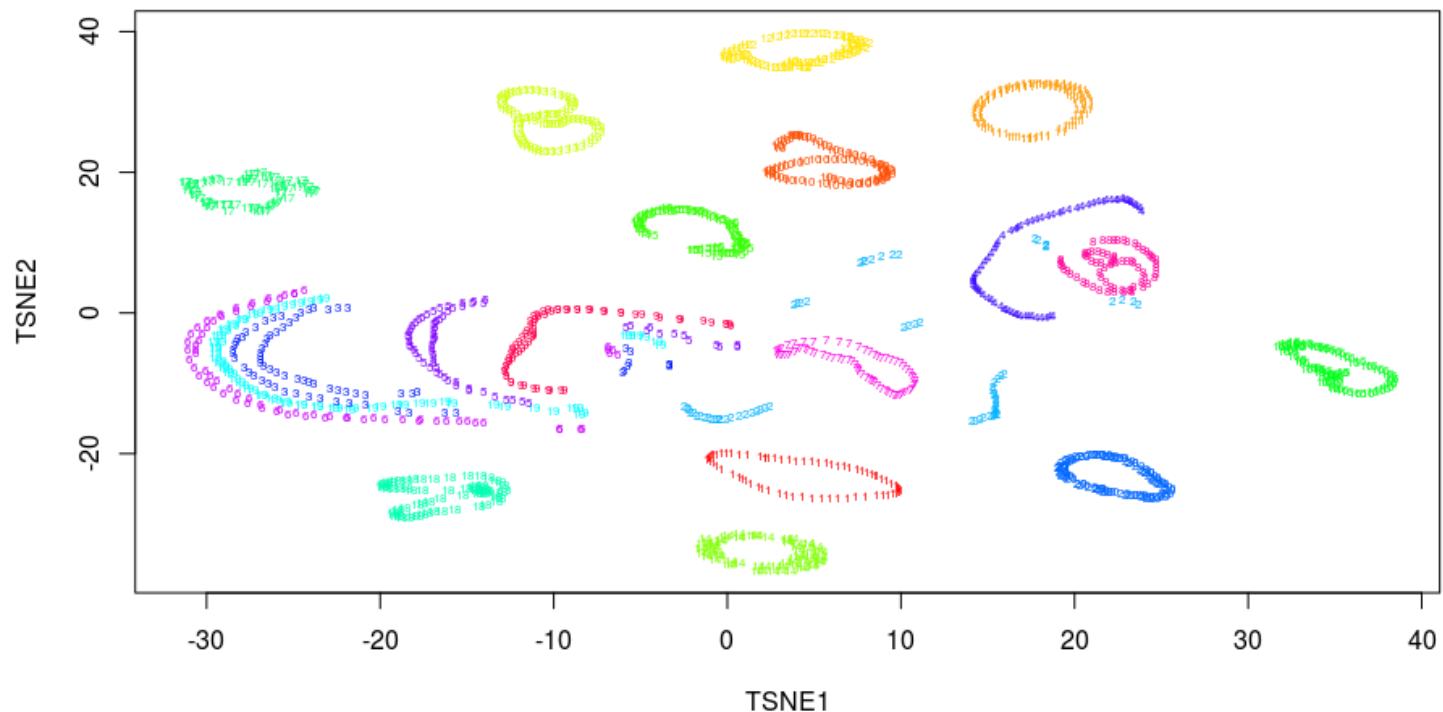
Ejemplo práctico

- Base de datos COIL20: imágenes de 20 objetos desde 72 ángulos diferentes.
- Proyección a 2D mediante PCA:



Ejemplo práctico

- Proyección a 2D mediante tSNE:



Referencias

1. Randal J. Barnes [Matrix Differentiation \(and some othe stuff\)](#)
2. Lee and Seung [Algorithms for Non-negative Matrix Factorization](#)
3. Tipping and Bishop, [Probabilistic principal component analysis](#)