Introducción a la Inteligencia Artificial

Clase 6

Índice

1. Motivación
    a. Aprendizaje No supervisado
    b. Aplicaciones
2. kMeans
3. Teoría - Principal Component Analysis
    a. Concepto
    b. Demostración Matemática

Algoritmos no supervisados

Aprendizaje no supervisado

Machine Learning Supervisado	Machine Learning no Supervisado
Proceso aleatorio $\bar{X}, y$	Proceso aleatorio $\bar{X}$
¿ $f_{y/\bar{x}}(y\|\bar{x})$? ⟶ Bayes y M.V.	¿ $f_{\bar{x}}(\bar{x})$? ⟶ Bayes y M.V.
Inferencias, predicciones	Clusterización, Reducción Dimensionalidad

como conozco y yo puedo definir un error.

$(y, \hat{y})$

Error? no tenemos un pto de comp. mis algoritmos se van a valer de un concepto de relajación (estabilidad)

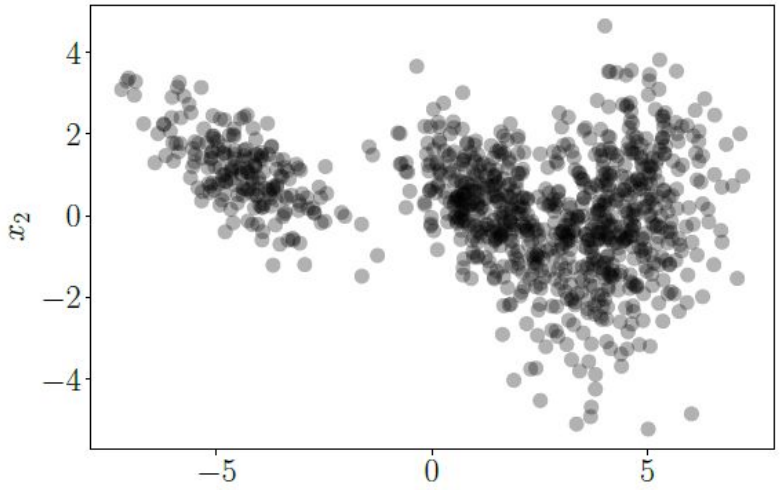
Aprendizaje no supervisado

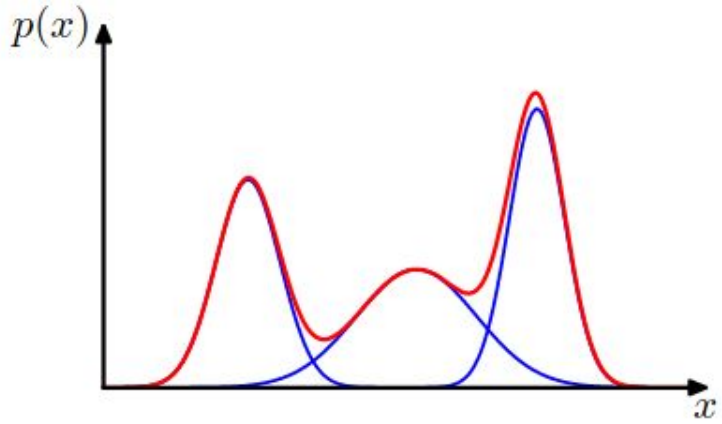
Aplicaciones Generales

- Data Mining
- Pattern Recognition
- Statistical Analysis

Aplicaciones Específicas

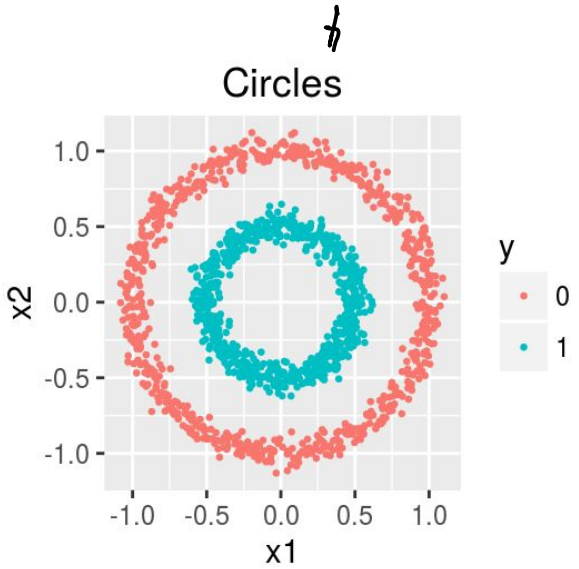
- Density Estimation
- Clustering
- Anomaly Detection
- Object Tracking
- Speech Feature Extraction

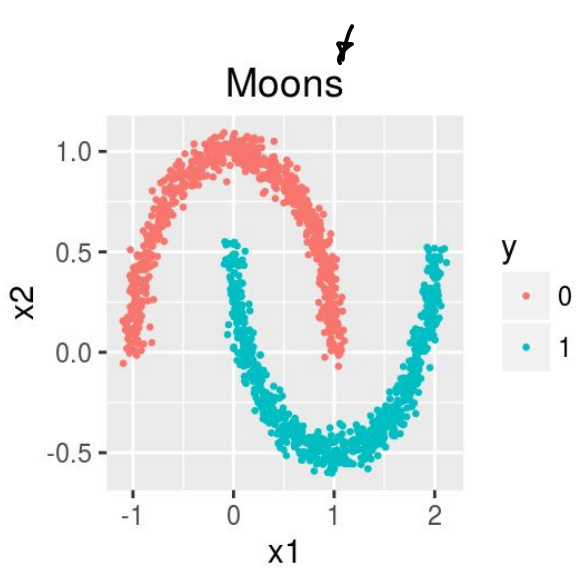


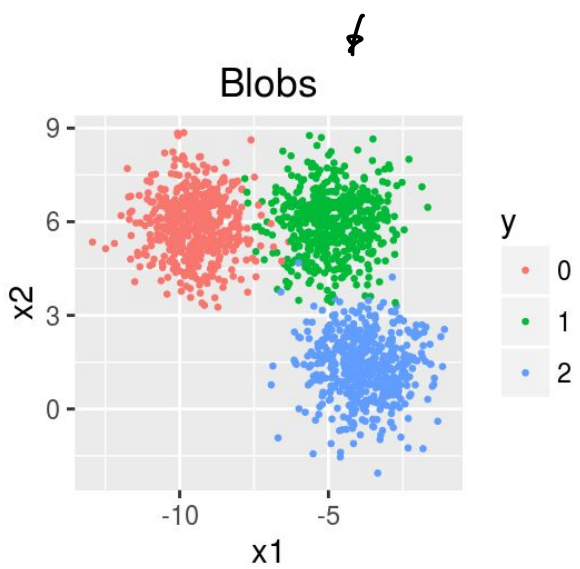


Clustering

La clusterización o clustering, es el proceso de agrupar objetos en grupos de manera que sean más similares entre sí que con los objetos de otros clusters.
Para generar estos grupos existen diferentes técnicas y diferentes medidas de similaridad.







kMeans

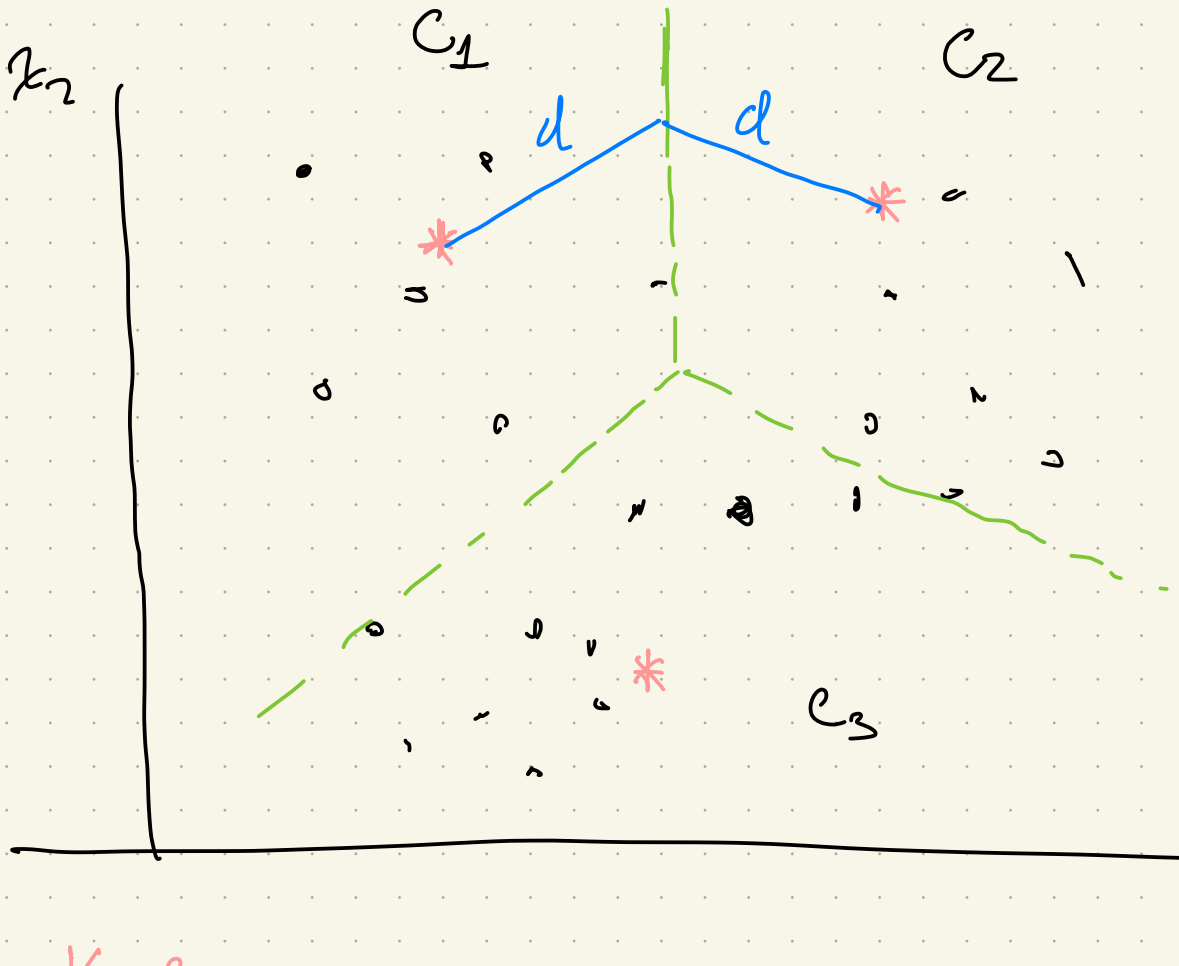
K-means es uno de los algoritmos más básicos en Machine Learning no supervisado. Es un algoritmo de **clusterización**, que agrupa los datos que comparten características similares. Recordemos que entendemos datos como n realizaciones del vector aleatorio X.

El algoritmo K-means funciona de la siguiente manera:

1. El usuario selecciona la cantidad de clusters a crear (n).
2. Se seleccionan n elementos aleatorios de X como posiciones iniciales del los centroides C.
3. Se calcula la distancia entre todos los puntos en X y todos los puntos en C.
4. Para cada punto en X se selecciona el centroide más cercano de C.
5. Se recalculan los centroides C a partir de usar las filas de X que pertenecen a cada centroide.
6. Se itera entre 3 y 5 una cantidad fija de veces o hasta que la posición de los centroides no cambie.

Implementar la función *def k_means(X, n)* de manera tal que al finalizar devuelva la posición de los centroides y a qué cluster pertenece cada fila de X.

Hint: para (2) utilizar funciones de np.random, para (3) y (4) usar los ejercicios anteriores, para (5) es válido utilizar un for. Iterar 10 veces entre (3) y (5).



Kmeans genera celdas de Voronoi

$K = 3$

$$\sigma_{c_1} = \frac{1}{N} \sum (x_{i_{c_1}} - \mu_{c_1})^2$$

$\sigma_{c_1} \sim \sigma_{c_2} \sim \sigma_{c_3}$

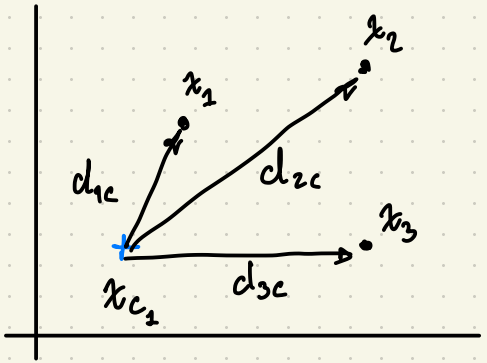
σ_{c_2}

σ_{c_3}

σ_{ig} ~> Varianza intra grupo

σ_{eg} → vi entre grupos

$\left.\begin{matrix} \sigma_{ig} \\ \sigma_{eg} \end{matrix}\right\} \frac{\sigma_{eg}}{\sigma_{ij}}$



$d(x_i, x_c)$ ⇝ Euclideana ✓ $\|\cdot\|^2$

⇝ Manhattan $\|\cdot\|$

⇝ city block ⇝ alternativa escala nula

└⇝ Mahalanobis

K means es valido para $\underset{=}{X} \in \mathbb{R}^{n \times p}$ ① $p \geq 1$ ②

└ preferible mayor estricto

$$d(x_i, x_c) = \sum_{j=1}^{p} (x_{ij} - x_{cj})^2 = \|x_i - x_c\|^2$$

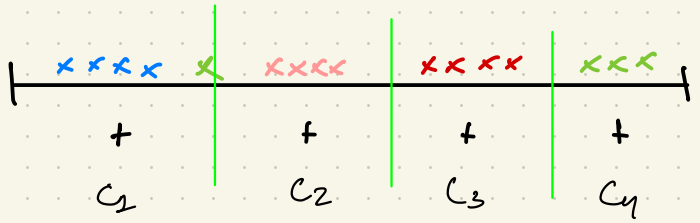
$$W(C) = \frac{1}{2} \sum_{k=1}^{K} \sum_{C(i)=k} \sum_{C(j)=K} \|x_i - x_j\|^2 \quad \rightsquigarrow \quad \min W ?$$

pag 529 Elements of statistical Learning

① Si mi x tiene categorias puedo usar K-proto.

② porque $p \geq 1$:

si $p = 1$



tenemos un problema de conv. con los centroides

posibles soluciones:

+ mejorar c-init
+ cambiar d_{ij} para ponderar clust. pequeños
+ introducir variaciones
+ modifico el solver

Alternativas: discretización → por anclo fijo
↳ u freq. fija.

buscamos: $\arg\min_S \sum_{i=1}^{k} \sum_{x \in S_i} \|x - \mu_i\|^2 = \arg\min_S \sum_{i=1}^{k} |S_i| \operatorname{Var} S_i$

media del cluster o el centroide

donde $\|\cdot\|^2$ es la norma, $|\cdot|$ es el conteo de muestras.

$$\mu_i = \frac{1}{|S_i|} \sum_{x \in S_i} x$$

$$\arg\min_S \sum_{i=1}^{k} 1/|S_i| \sum_{x,y \in S} \|x-y\|^2$$

BCSS

between cluster sum of squares

Algoritmo estandar:

- etapa de asignación: Aca labeleamos basado en su centroide más cercano. (en un paso inicial lo hacemos basado en c_init)

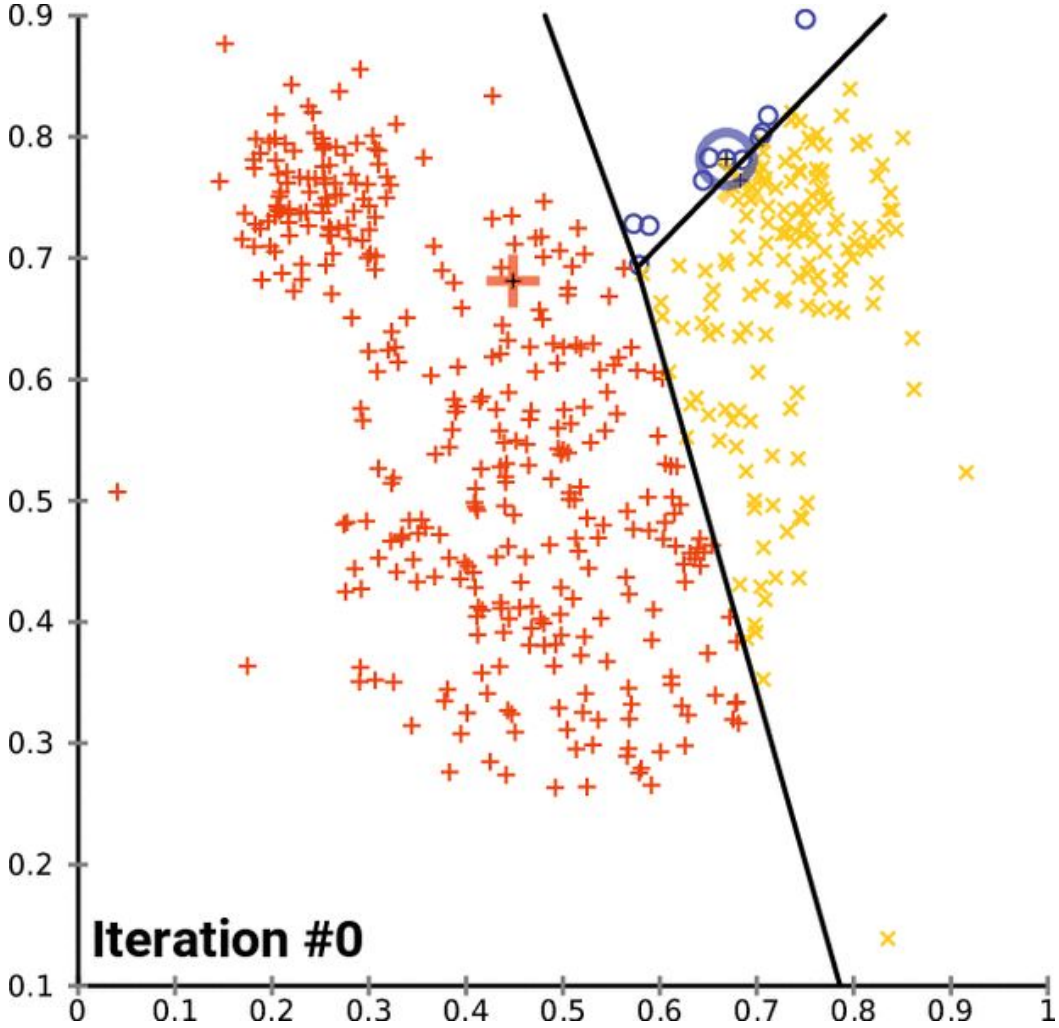
$$S_i^{(t)} = \left\{ x_p : \|x_p - m_i^{(t)}\|^2 \leq \|x_p - m_j^{(t)}\|^2 \quad \forall j,\ 1 \leq j \leq k \right\}$$

- proceso de update: recalculamos la media de cada cluster:

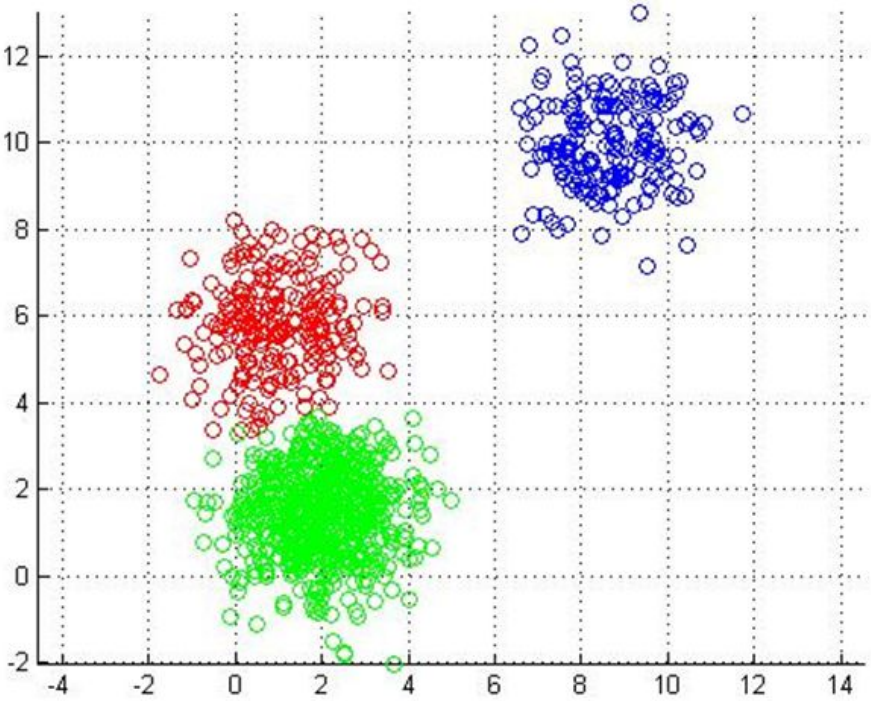
$$m_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \cdot \sum_{x_j \in S_i^{(t)}} x_j$$

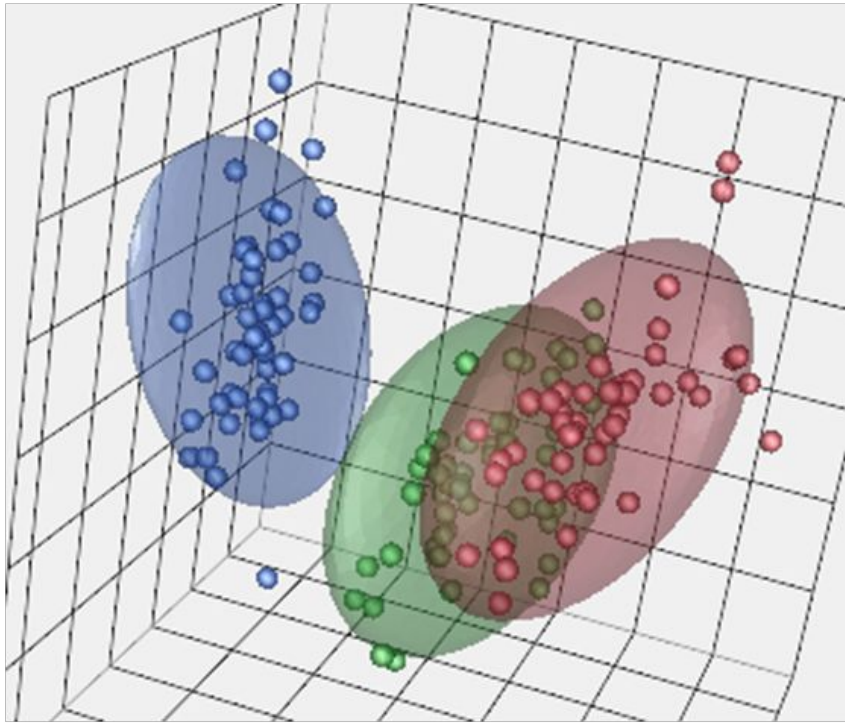
$$\Delta m = m_i^{(t+1)} - m_i^{(t)} \sim 0$$

. Si necesitamos extender la capacidad de clusterización con Var. no numéricas → K-proto, K-medians, K-medoids


0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Iteration #0

kMeans - Image segmentation





kMeans en R3

K no se calcula → K se fija

¿Como validamos K-means? → Endogenos (Validación interna)

↳ exogenos (Validación externa)

Validación externa → conocimiento experto

↳ Validación empírica

↳ " "crurada"

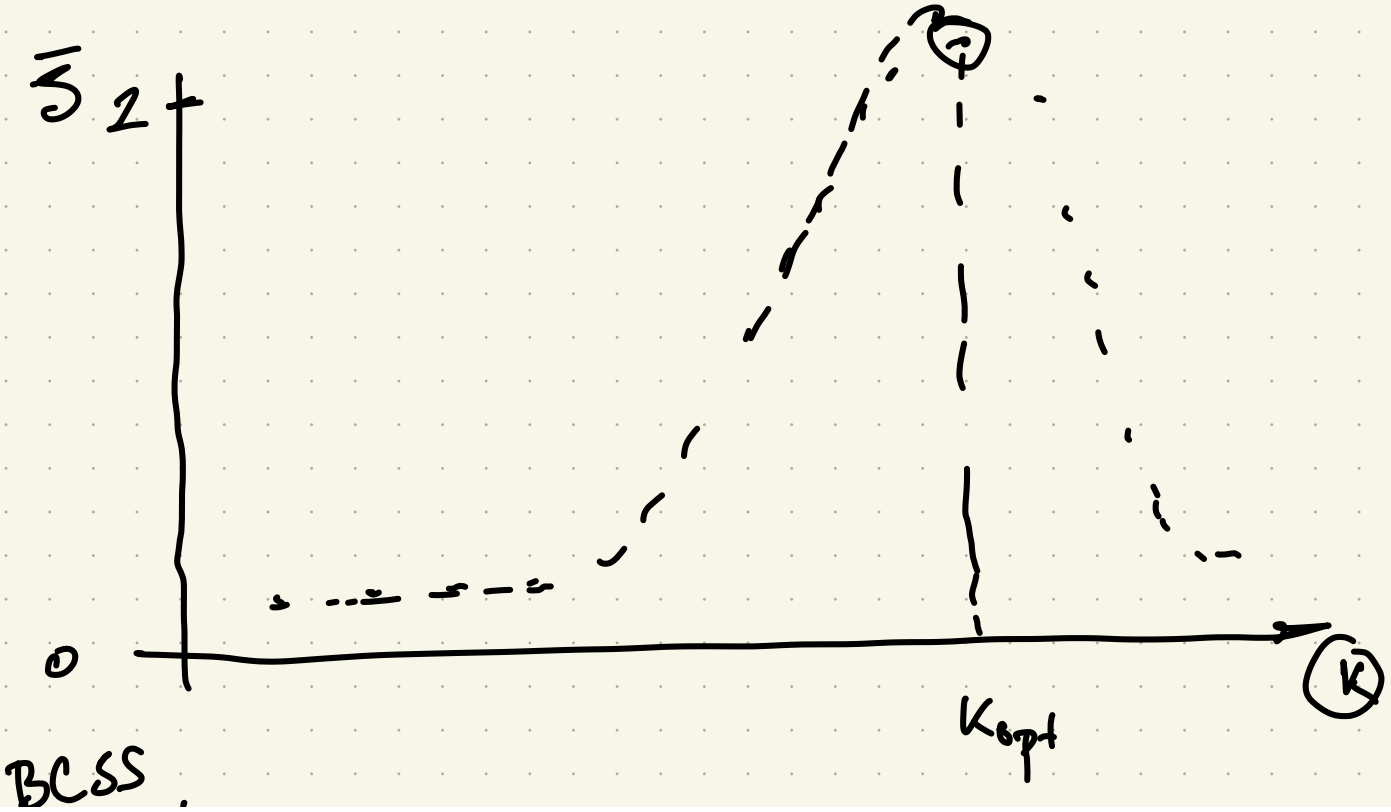
Validación interna → método elbow (el método del codo)

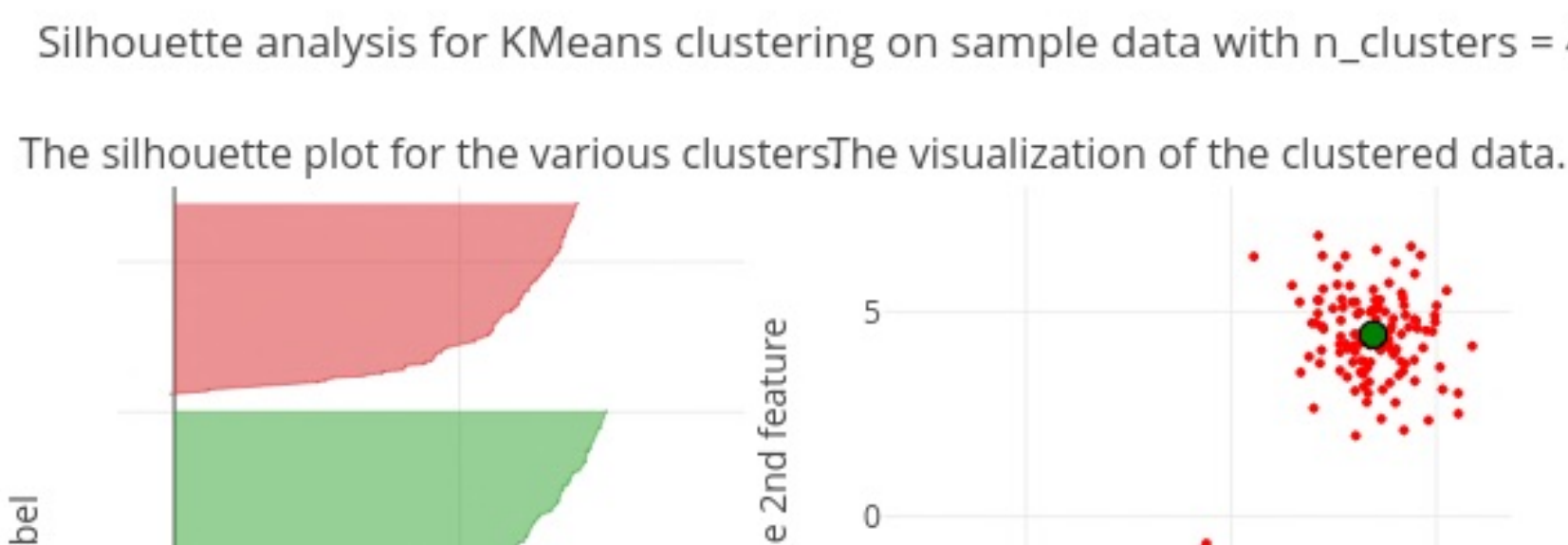
↳ métricas de relajación → factor chico

↳ métricas de clusterización

↳ coef. silhouette

como volido K?





Reducción de dimensionalidad

$\mathbb{R}^{n \times p} \longmapsto \mathbb{R}^{m \times p} \qquad m << n$

El objetivo de los modelos de reducción de dimensionalidad es encontrar una “mejor” representación de los datos.
Con “mejor” nos referimos a una representación que preserve la mayor cantidad de información posible de los datos, bajo una determinada penalidad o restricción, que haga que la representación sea más accesible o simple.

Ejemplos de representaciones más simples:

- Representación de menor dimensionalidad
- Representación sparsa
- Representación independiente

Ingeniería de Features - PCA

En ocasiones los datos de entrada tienen muchas features y se torna costoso en tiempo y recursos entrenar modelos de ML con todo el dataset. En la práctica se pueden utilizar técnicas de reducción de la dimensión no supervisadas como PCA (Principal Component Analysis).

Casos de Uso

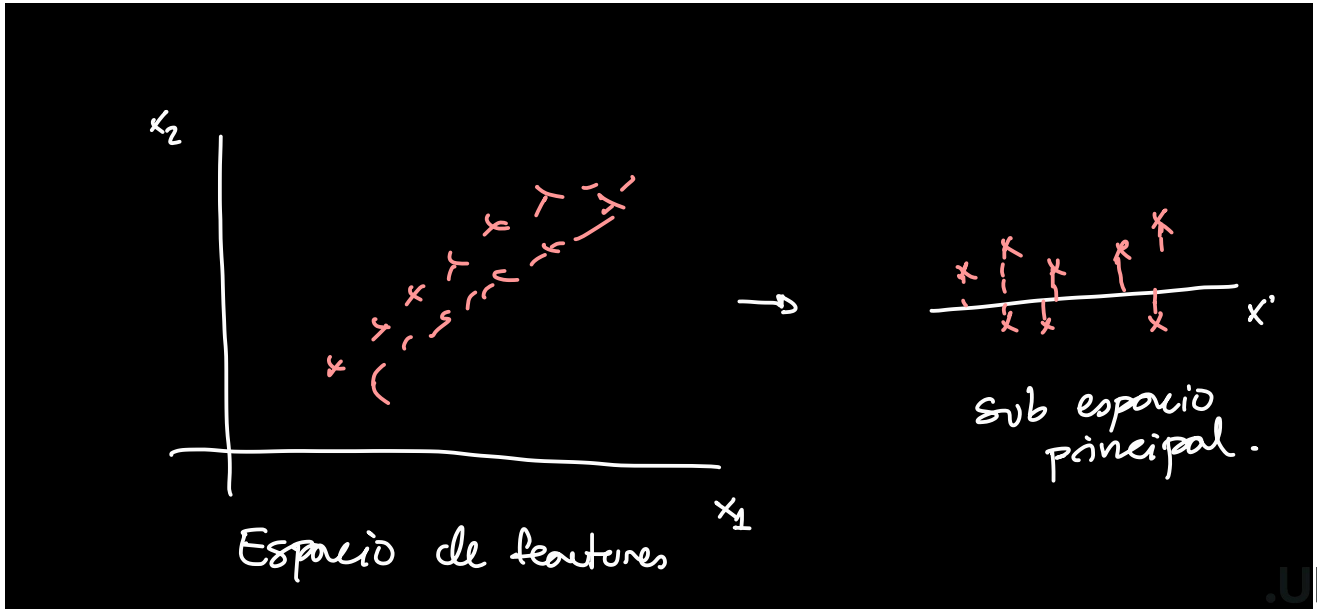
- Compresión de datos
- Identificación de patrones
- Factores latentes
- Visualización

Conocimientos Previos

- Bases y cambio de bases
- Proyecciones
- Valores y vectores propios
- Distribución gaussiana
- Optimización con restricciones

PCA

Queremos encontrar proyecciones … de observaciones de datos ..., que sean lo más similares posibles a los originales, pero con significativamente menos dimensiones.



PCA

me dice que solo soporta numéricas var.

Dado un dataset i.i.d:

$$\chi = \{x_1, \cdots, x_N\}, x_N \in \mathbb{R}^D$$

x_{11}	$\cdots$	$\cdots$	x_{1n}
$\cdots$	$\cdots$	$\cdots$	$\cdots$
x_{d1}	$\cdots$	$\cdots$	x_{dn}

χ

con **media cero**, la matriz de covarianza es: (1)

$$S = \frac{1}{N} \sum_{n=1}^{N} x_n x_n^T$$

Definimos transformaciones lineales:

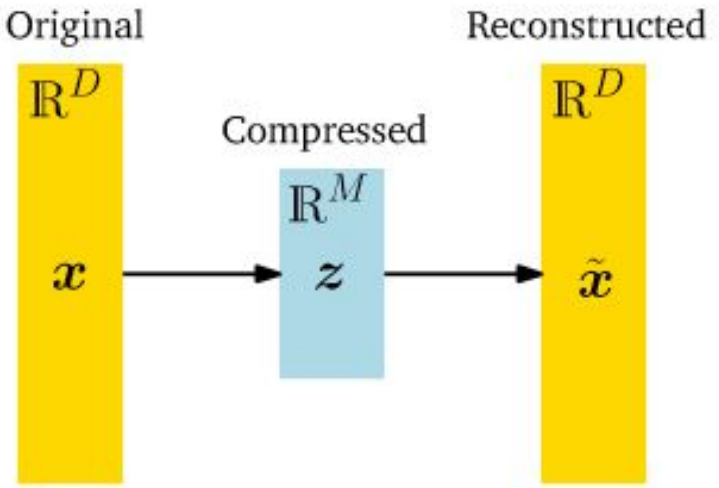
$$z_n = B^T x_n \in \mathbb{R}^M \qquad M \leq D$$

$$B = [b_1, \cdots, b_m] \in \mathbb{R}^{DxM}, b_i^T b_j = 0 \; \forall \; i \neq j \quad (\|b_i\| = 1)$$

(1) me dice que X tiene que estar escaleado.

PCA

Autoencoder



Buscamos un subespacio

$$U \subseteq \mathbb{R}^D / \ dim(U) = M < D$$

donde proyectar los datos. Es decir encontrar para:

$$\tilde{x}_n \in \mathbb{R}^D \begin{cases} z_n \\ [b_1, \cdots, b_m] \end{cases}$$

$$z_n = B^T x_n$$
$$\tilde{x}_n = B z_n$$

i. Enfoque de máxima varianza
ii. Enfoque de error de reconstrucción mínimo
iii. Enfoque de variables latentes

Jamboard - Desarrollo Matemático PCA

- Introducción
- Enfoque de maximización de varianza
- Enfoque de minimización de error de reconstrucción
- Enfoque por variables latentes

Desarrollo matemático de PCA:

buscamos una proyección $\tilde{x}_n$ de mis datos originales x_n / $\dim(\tilde{x}_n) \leq \dim(x_n)$

- x es el dataset iid $\wedge$ $x \in \mathbb{R}^D$ $\wedge$ $\vec{\mu}_x = \emptyset$
- matriz de cov es: $S = \frac{1}{N} \sum x_n x_n^t$

Metodo de maxima varianza: buscamos maximizar la varianza en una dimensión inferior

Partimos con una columna de B $(\mathbb{R}^{M \times D})$, $b_1 \in \mathbb{R}^D$

Lo maximizamos la varianza de z_1 de $z \in \mathbb{R}^M$:

$$Var[z] = Var[B^t(X - \mu)] = Var[B^t x - B^t \mu] = Var[B^t x]$$

($B^t \mu = 0$)

$$Var_1 = Var[z_{1n}] = \frac{1}{N}\sum_{n=1}^{N} z_{1n}^2 \quad ; \qquad \boxed{z_{1n} = b_1^t x_{1n}}$$

↳ proyección ortogonal de x_1 en el subesp. unidimensional formado por b_1

$$Var_1 = \frac{1}{N}\sum_{i=1}^{N}\left(b_1^t \cdot x_n\right)^2 = \frac{1}{N}\sum_{i=1}^{N}(b_1^t x_n)^t (b_1^t x_n)$$

$$= \frac{1}{N}\sum_{n=1}^{N} b_1^t x_n x_n^t b_1 = b_1^t \underbrace{\frac{1}{N}\sum_{n=1}^{N} x_n x_n^t}_{S} b_1$$

$$\boxed{Var_1 = b_1^t S \cdot b_1}$$

si aumento b_1 ⇒ incremento Var_1

objetivo: $\max_b \; b_1^t S b_1 \;,\; \|b_1\| = 1$ maximización condicionada

$$L(b_1, \lambda_1) = b_1^t S b_1 + \lambda_1 (1 - b_1^t b_1)$$

$$\begin{cases} \partial_{\lambda_1} L = 0 \rightarrow 1 - b_1^t b_1 = 0 \rightarrow b_1^t b_1 = 1 \\ \partial_{b_1} L = 0 \rightarrow (2) \end{cases}$$

② → $2\, b_1^t S - 2\lambda_1 b_1^t = 0$

(*) puedo hacer esto porque S es simétrica

$(b_1^t S)^t = (\lambda_1 b_1^t)^t$

$S^t \cdot b_1 = b_1 \lambda_1^t$

(*) ↓

$S \cdot b_1 = \lambda_1 b_1$ ⇝ $S \cdot b_1 = \lambda \cdot b_1$

λ ← Valor propio

b_1 ← Vector propio de S

$S \cdot b_1 = \lambda_1 b_1$ — def. autovalor

⇝ seleccionar los autovectores asociados a los m autovalores más grandes de la matriz de covarianza.

con esto:

- Varianza explicada: $\sum_{i=1}^{M} \lambda_i$ → % de la varianza total.
- " perdida: $\sum_{i=M+1}^{D} \lambda_i$

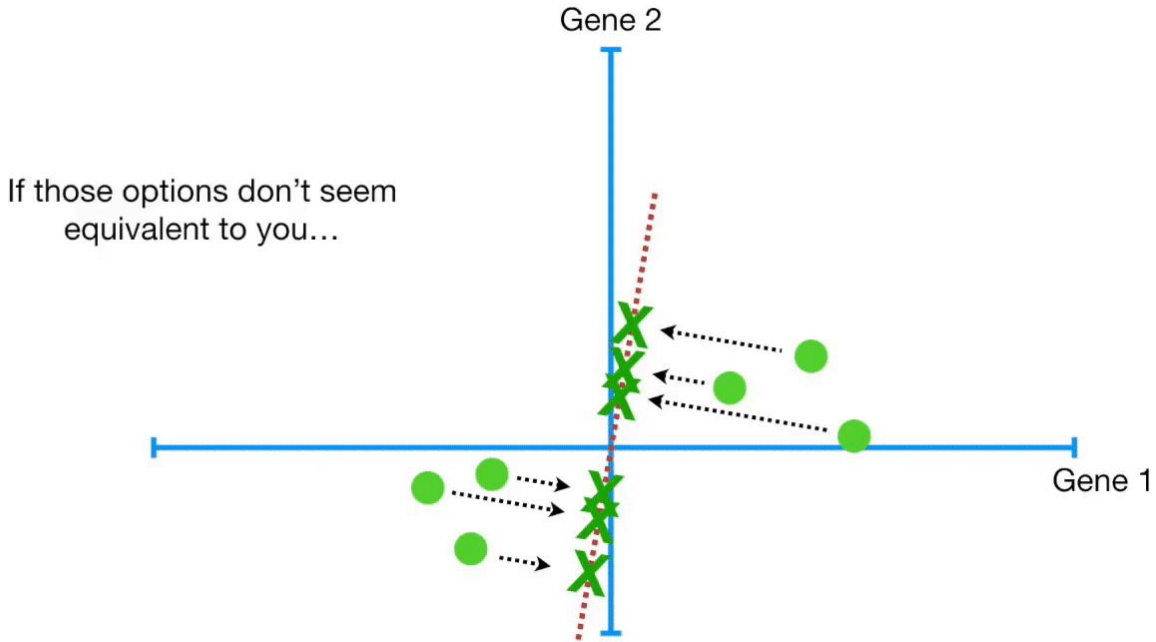
$\mathbb{R}^D \to \mathbb{R}^M$

M es un parametro de nuestro modelo.

$M \leq D$

PCA

Comparación métodos 1 y 2.



Principal Component Analysis - Teoría

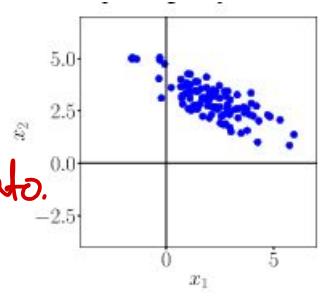
PCA

Pasos principales: 0. Validar tipo de dato.

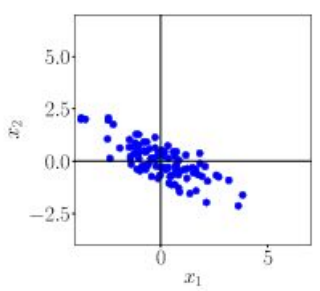
1. Centramos los datos
2. Estandarización
3. Autovalores de la matriz de covarianza
4. Proyección

$$z_n = B^T x_n$$

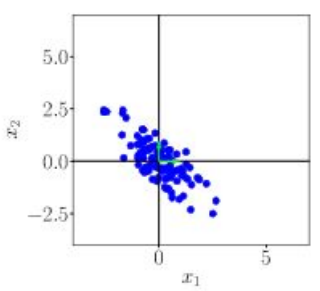
3'. Selecciono los m autovectores



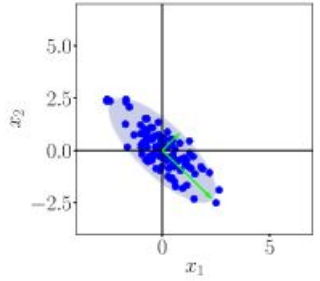
(a) Original dataset.



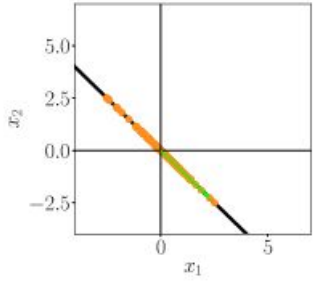
(b) Step 1: Centering by subtracting the mean from each data point.



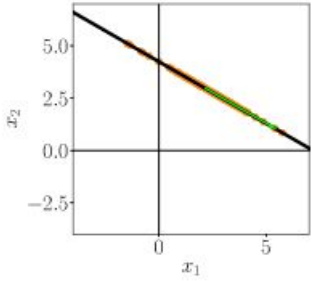
(c) Step 2: Dividing by the standard deviation to make the data unit free. Data has variance 1 along each axis.



(d) Step 3: Compute eigenvalues and eigenvectors (arrows) of the data covariance matrix (ellipse).



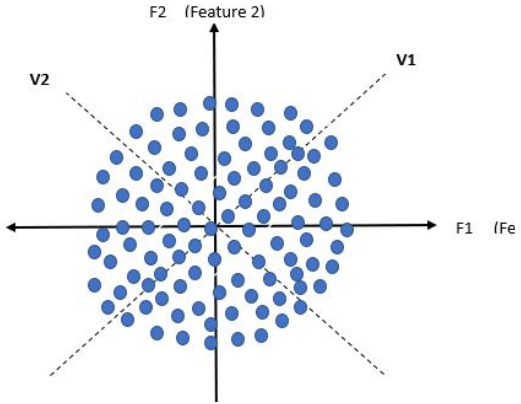
(e) Step 4: Project data onto the principal subspace.

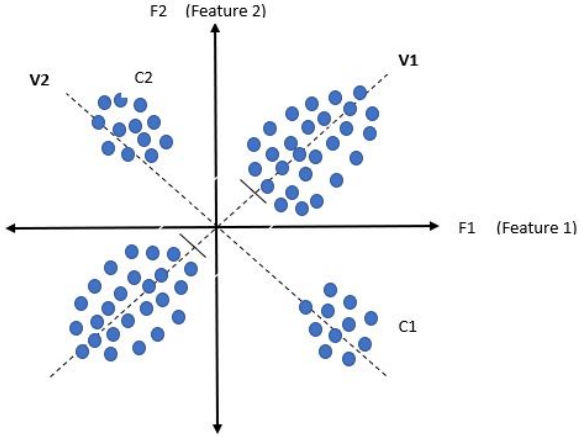


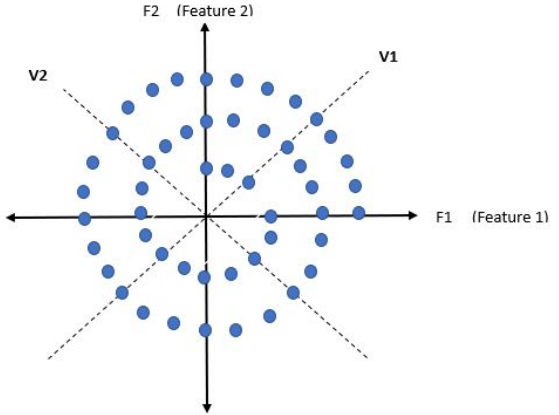
(f) Undo the standardization and move projected data back into the original data space from (a).

PCA

Limitaciones







PCA

Derivaciones

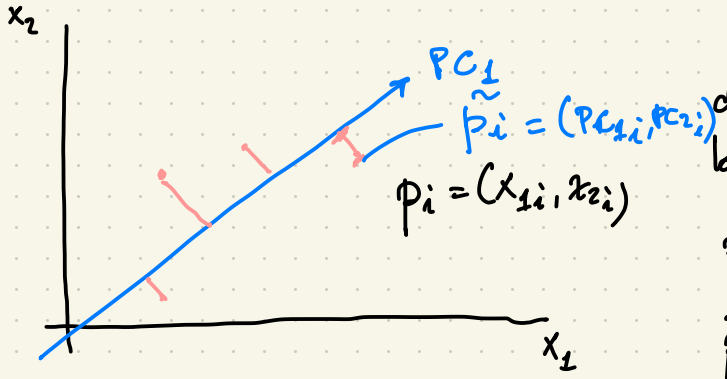
$z_n = \phi(B^t x_n)$ ← Kernel-PCA
↳ fn. Kernel

- Si en PCA cambiamos el mapeo lineal por uno no-lineal, obtenemos un auto-encoder. Si el mapeo no-lineal es una red neuronal, tenemos un deep auto-encoder.
- Cuando la varianza del ruido gaussiano es cero, PPCA → PCA.
- Si para cada dimensión, el ruido tiene una varianza distinta → Factor Analysis.
- Si cambiamos la distribución a priori de z por una no gaussiana → ICA

$x_n = \cancel{x} + \varepsilon \rightarrow \quad \varepsilon \sim N(\cancel{\bar{\phi}}, \bar{\Sigma})$

Segunda Forma de derivación de PCA:

Minimización del error de reducción:



Si proyectamos un pto. sobre una dirección de la reconstrucción vamos a poder definir la posición del pto $\tilde{p}$

p_i : pto. original
$\tilde{p}_i$: pto. proyectado

} Vamos a minimizar $\| p_i - \tilde{p}_i \|$

Vamos a suponer $\exists$ B base ortonormal de $\mathbb{R}^D$ con esto decimos que $\forall x \in \mathbb{R}^D$ podemos asegurar:

$$\vec{x} = \sum_{d=1}^{D} \alpha_d \cdot \bar{b}_d = \underbrace{\sum_{i=1}^{M} \alpha_i \bar{b}_i}_{\tilde{x}} + \sum_{d=M+1}^{D} \alpha_d \bar{b}_d$$

queremos encontrar $\tilde{x} = \sum_{i=1}^{m} \alpha_i \cdot b_i \in U \subseteq \mathbb{R}^D$ tal que podamos maximizar la similaridad entre $\tilde{x}$ y x:

$$\tilde{x} = \sum_{i=1}^{m} z_i b_i = \bar{B} \cdot \bar{z}_i$$

buscamos minimizar el MSE $\| x - \tilde{x} \|^2$: → $x \in \mathbb{R}^D$

$\tilde{x} \in \mathbb{R}^M \rightarrow [\tilde{x} \mid \emptyset] \in \mathbb{R}^D$

① . optimizar z_n para una dada base.

② . encontrar una base óptima.

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

$$\partial_{z_{in}} J_M = \frac{\partial J_M}{\partial \tilde{x}_n} \cdot \frac{\partial \tilde{x}_n}{\partial z_{in}}$$

$$= \left(-\frac{2}{N} \cdot (x_n - \tilde{x}_n)^t \right) \cdot \left(\partial_{z_{in}} \left(\sum_{i=1}^{M} z_{in} \cdot b \right) \right) = -\frac{2}{N} (x_n - \tilde{x}_n)^t \cdot b_i$$

$$\partial_{z_{in}} J_M = -\frac{2}{N}\left(x_n - \sum_{i=1}^{M} z_{in}\cdot b_i\right)^t \cdot b_i = -\frac{2}{N}\left(x_n^t b_i - z_{in}\cdot \underbrace{b_i^t \cdot b_i}_{=1}\right)$$

$$= -\frac{2}{N}\left(x_n^t b_i - z_{in}\right)$$

$$\partial_{z_{in}} J_M = 0 \Rightarrow -\frac{2}{N}\left(x_n^t b_i - z_{in}\right) = 0 \rightarrow \boxed{z_{in} = b_i\, x_n^t}$$

Los z_{in} son las coord. que vamos a encontrar para cada vector $(b_i\, x_n^t)$

- buscamos ahora la base óptima:

$$\tilde{x}_n = \sum_{n=1}^{m} z_{mn}\cdot b_n = \sum_{n=1}^{M}\left(x_n^t b_n\right) b_n = \sum_{n=1}^{M}\left(b_n b_n^t\right)\cdot x_n$$

$$x = \left(\sum_{n=1}^{M} b_n b_n^t\right)\cdot x_n + \left(\sum_{j=M+1}^{D} b_j b_j^t\right)\cdot x_j$$

$$x = (x_1, x_2, \ldots, x_m, \ldots, x_n)$$

$$\tilde{x} = (x_1', x_2', \ldots, x_m', 0, \ldots, 0)$$

$$x - \tilde{x} = (0, 0, \ldots, \underbrace{x_{m+1}, \ldots, x_n})$$

distancia (error): $x_n - \tilde{x}_n = \sum_{j=M+1}^{D} b_j b_j^t \cdot x_j = \sum_{j=M+1}^{D} (x_n^t \cdot b_j) \cdot b_j$

$$J_M = \frac{1}{N} \sum_n \|x_n - \tilde{x}_n\|^2 = \frac{1}{N} \cdot \sum_{n=1}^{N} \left\| \sum_{j=M+1}^{D} (x_n^t \cdot b_j) b_j \right\|^2$$

$$= \sum_{j=M+1}^{D} b_j^t \underbrace{\left(\frac{1}{N} \sum_{n=1}^{N} x_n \cdot x_n^t \right)}_{S} \cdot b_j$$

$$= \sum_{j=M+1}^{D} b_j^t S b_j = \sum_{j=M+1}^{D} tr(b_j^t S b_j) = \sum_{j=M+1}^{D} tr(S b_j^t \cdot b_j)$$

$$= tr\left(\underbrace{\sum_{j=M+1}^{D} b_j b_j^t}_{\text{Matriz de proyección}} \cdot S \right)$$

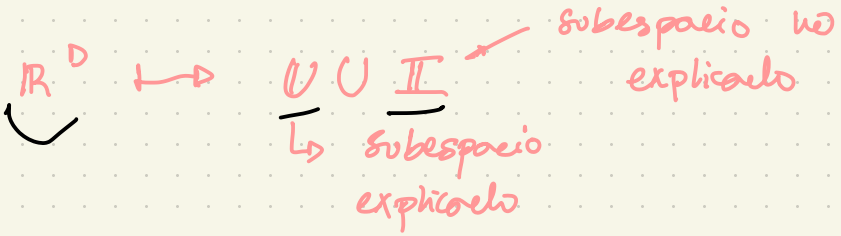
→ el error de reconstrucción se puede pensar cómo la matriz S proyectada sobre el complemento ortogonal de U.

↳ subesp. principal explicado

$\mathbb{D} = U \cup \bar{U}$

↳ subesp. perdido

Al reducir dimensiones:

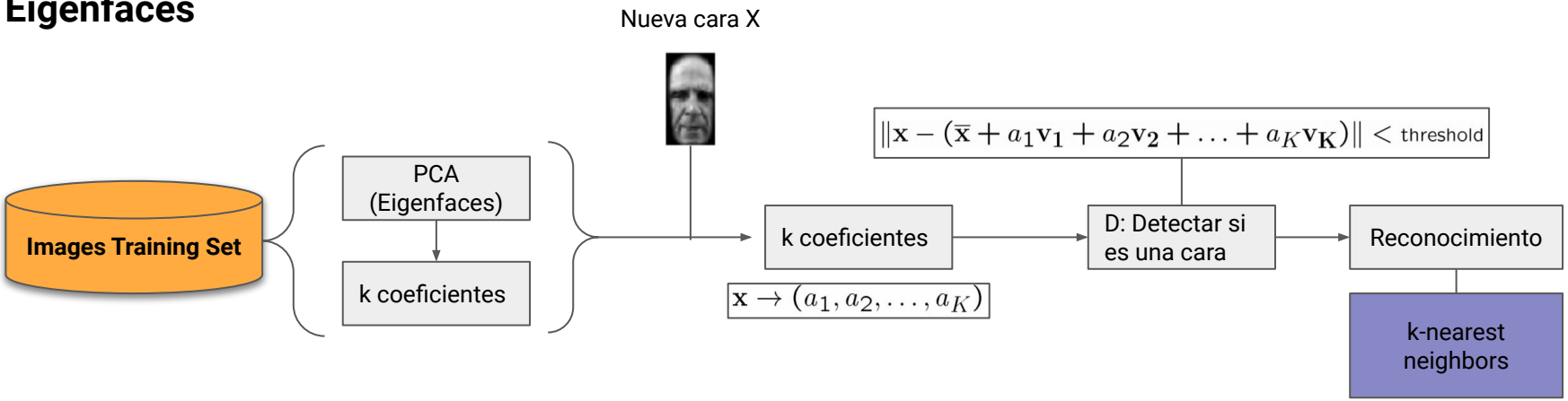


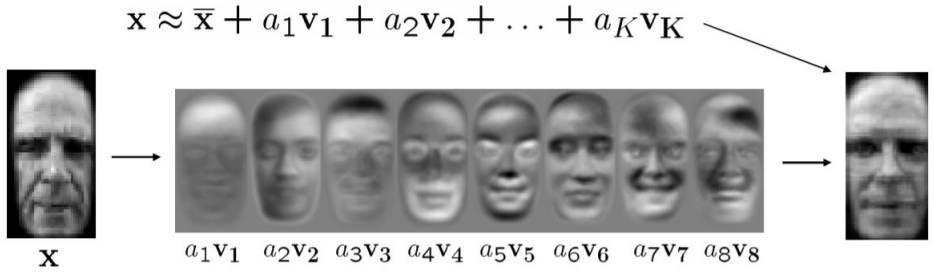
para calcular PCA, nosotros podemos elegir maximizar $\mathbb{U}$ o minimizar $\mathbb{II}$.

PCA - Ejemplo

PCA

Eigenfaces





Bibliografía

- The Elements of Statistical Learning | Trevor Hastie | Springer
- An Introduction to Statistical Learning | Gareth James | Springer
- Deep Learning | Ian Goodfellow | https://www.deeplearningbook.org/
- Stanford | CS229T/STATS231: Statistical Learning Theory | http://web.stanford.edu/class/cs229t/
- Mathematics for Machine Learning | Deisenroth, Faisal, Ong
- Artificial Intelligence, A Modern Approach | Stuart J. Russell, Peter Norvig