

<http://www.computervisionmodels.com/>

Descargar el contenido del tema 13



# Tema 3: Reducción de dimensionalidad

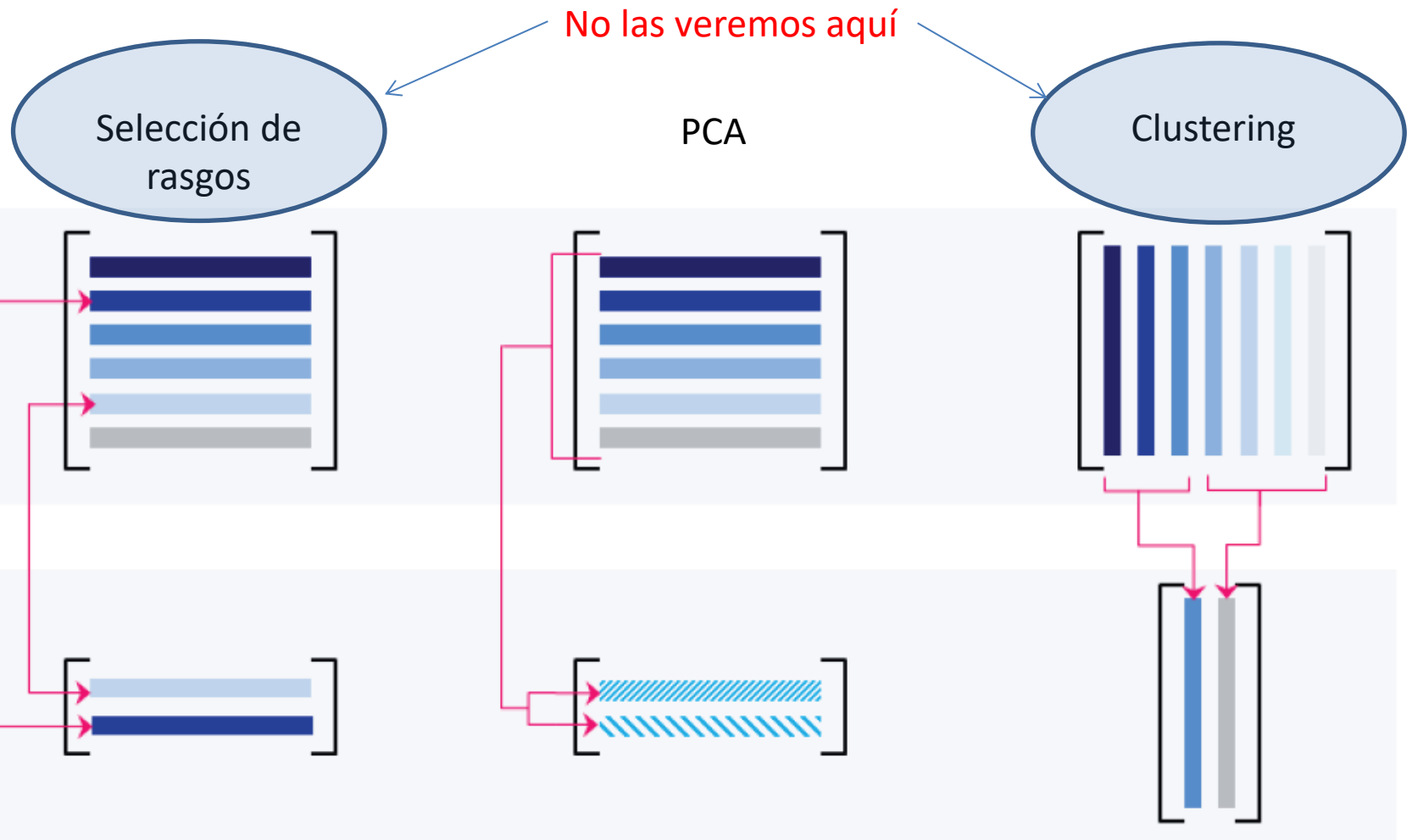
Rafael Molina

- En este tema vamos a analizar técnicas **deterministas** para reducir la dimensionalidad de una base de datos en general y de imágenes en particular.
- Veremos que la utilidad de estas técnicas no se limita a la reducción de dimensionalidad. Hablaremos, en particular, de variables latentes.
- En temas posteriores estudiaremos la modelización probabilística de estas técnicas. Analizaremos modelos lineales simples y modelos basados en deep learning.

- Comentaremos en la diapositiva siguiente técnicas que:
  - seleccionan un subconjunto de rasgos de cada muestra (selección de rasgos)
  - reducen el número de muestras tipo en una base de datos pero mantienen la dimensión del vector de rasgos
- Nos concentraremos después en una técnica que
  - transforma los vectores de rasgos a otro espacio y allí selecciona un subconjunto de nuevos rasgos.

Matriz de datos  
de entrada

Matriz de datos  
de salida



Las columnas representan (en esta diapositiva) los vectores de rasgos. Cada fila representa un rasgo concreto. Observa y entiende la matriz de datos de salida

# Estructura del Tema

- PCA
- Ejemplos
- PCA para alta dimensión
- Núcleos (Kernel)
- KPCA
- Apéndice: Autovalores y autovectores

# Principal Component Analysis (PCA)

## Análisis de Componentes Principales

Una recomendación inicial. Dados  $I$  vectores de rasgos (o instancias)  $\{x_1, x_2, \dots, x_I\}$ , cada uno con  $D$  componentes, es conveniente que :

1. cada uno de los rasgos en el vector esté centrado en cero y
2. normalizado por la varianza o el rango de cada uno de ellos.

# Principal Component Analysis (PCA)

## Análisis de Componentes Principales

**Muy importante.**

$$X^T = [x_1, x_2, \dots, x_I]_{D \times I}$$

$I$  es el número de vectores de rasgos

$D$  es el número de componentes que tiene cada vector de rasgos

Observa: las filas de  $X$  contienen las instancias o muestras de entrenamiento. Estas filas son  $x_i^T$ .

# Principal Component Analysis (PCA)

## Análisis de Componentes Principales

El objetivo de la reducción de la dimensionalidad (de los rasgos) es encontrar una representación de menor dimensión (o escondida)  $\mathbf{h}$  que pueda explicar aproximadamente el dato (vector de rasgos)  $\mathbf{x}$  de modo que

$$\mathbf{x} \approx f(\mathbf{h}, \boldsymbol{\theta})$$

donde  $f[\bullet, \bullet]$  es una función de variables ocultas y un conjunto de parámetros  $\boldsymbol{\theta}$ .

Normalmente elegimos una familia de funciones  $f[\bullet, \bullet]$  y entonces aprendemos  $\mathbf{h}$  y  $\boldsymbol{\theta}$  de un conjunto de entrenamiento.



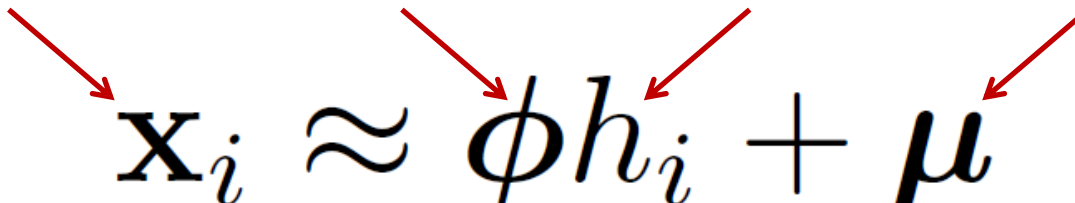
# Criterio de Mínimos cuadrados

$$\hat{\boldsymbol{\theta}}, \hat{\mathbf{h}}_{1 \dots I} = \underset{\boldsymbol{\theta}, \mathbf{h}_{1 \dots I}}{\operatorname{argmin}} \left[ \sum_{i=1}^I (\mathbf{x}_i - f[\mathbf{h}_i, \boldsymbol{\theta}])^T (\mathbf{x}_i - f[\mathbf{h}_i, \boldsymbol{\theta}]) \right]$$

Elegir los parámetros  $\boldsymbol{\theta}$  y las variables ocultas  $\mathbf{h}$  (fíjate en la terminología) de forma que minimicen el error cuadrático medio de aproximación (una medida de cómo de bien la aproximación puede reconstruir los datos  $\mathbf{x}$ ).

# Ejemplo simple

Vector columna      Vector columna      Escalar      Vector columna


$$\mathbf{x}_i \approx \boldsymbol{\phi} h_i + \boldsymbol{\mu}$$

Aproximamos cada vector de rasgos  $\mathbf{x}$  mediante un escalar  $h$ .

Los datos son reconstruidos multiplicando  $h$  por un vector de parámetros  $\boldsymbol{\phi}$  y sumándole un vector de medias  $\boldsymbol{\mu}$ .

Observa que estamos aproximando un vector por un escalar que al multiplicarlo por un vector debe parecerse al vector inicial.

resta la media  $\boldsymbol{\mu}$  a cada vector de datos y tendremos vectores de media cero.

# Ejemplo Simple

$$\begin{pmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{iD} \end{pmatrix} = \mathbf{x}_i \approx \boldsymbol{\phi} h_i = h_i \begin{pmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_D \end{pmatrix}$$

Aproximar cada vector de rasgos  $\mathbf{x}$  por un valor escalar  $h$ , usando un vector común  $\boldsymbol{\phi}$ .

Los vectores de datos se reconstruyen (aproximan, estiman) multiplicando  $h$  por un factor (vector)  $\boldsymbol{\phi}$ .

**Criterio:**

$$\hat{\boldsymbol{\phi}}, \hat{h}_{1\dots I} = \underset{\boldsymbol{\phi}, h_{1\dots I}}{\operatorname{argmin}} [E] = \underset{\boldsymbol{\phi}, h_{1\dots I}}{\operatorname{argmin}} \left[ \sum_{i=1}^I (\mathbf{x}_i - \boldsymbol{\phi} h_i)^T (\mathbf{x}_i - \boldsymbol{\phi} h_i) \right]$$

# Criterio

$$\hat{\phi}, \hat{h}_{1...I} = \underset{\phi, h_{1...I}}{\operatorname{argmin}} [E] = \underset{\phi, h_{1...I}}{\operatorname{argmin}} \left[ \sum_{i=1}^I (\mathbf{x}_i - \phi h_i)^T (\mathbf{x}_i - \phi h_i) \right]$$

Problema: la solución no es única. Si multiplicamos  $\phi$  por cualquier constante  $\alpha$  y dividimos cada una de las variables ocultas  $h_{1...I}$  por la misma constante obtenemos el mismo coste (es decir  $(\phi\alpha) (h_i/\alpha) = \phi h_i$ )

Solución: Hacemos la solución única restringiendo a que la norma de  $\phi$  sea 1 usando un multiplicador de Lagrange.

# Criterio

Tenemos ahora la nueva función de coste

$$\begin{aligned} E &= \sum_{i=1}^I (\mathbf{x}_i - \phi h_i)^T (\mathbf{x}_i - \phi h_i) + \lambda(\phi^T \phi - 1) \\ &= \sum_{i=1}^I (\mathbf{x}_i^T \mathbf{x}_i - 2h_i \phi^T \mathbf{x}_i + h_i^2) + \lambda(\phi^T \phi - 1). \end{aligned}$$

Para minimizarla calculamos derivadas con respecto a  $\phi$  y  $h_i$ , igualamos a cero y reordenamos.

# Solución

Derivando con respecto a  $h_i$  e igualando a cero obtenemos

$$\hat{h}_i = \hat{\phi}^T \mathbf{x}_i$$

Observa que para calcular la variable oculta lo único que tenemos que hacer es calcular el producto escalar del vector inicial de rasgos con el vector  $\phi$ . Vamos ahora a calcular el vector  $\phi$ .

# Solución

Derivando con respecto a  $\phi_i$  e igualando a cero obtenemos  
(comprobar)

$$\sum_{i=1}^I \mathbf{x}_i \mathbf{x}_i^T \hat{\phi} = \lambda \hat{\phi}$$

donde

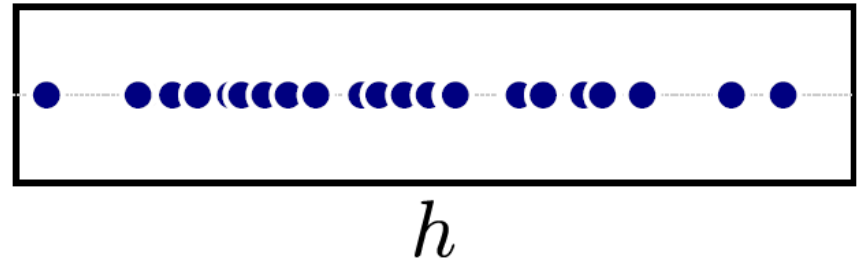
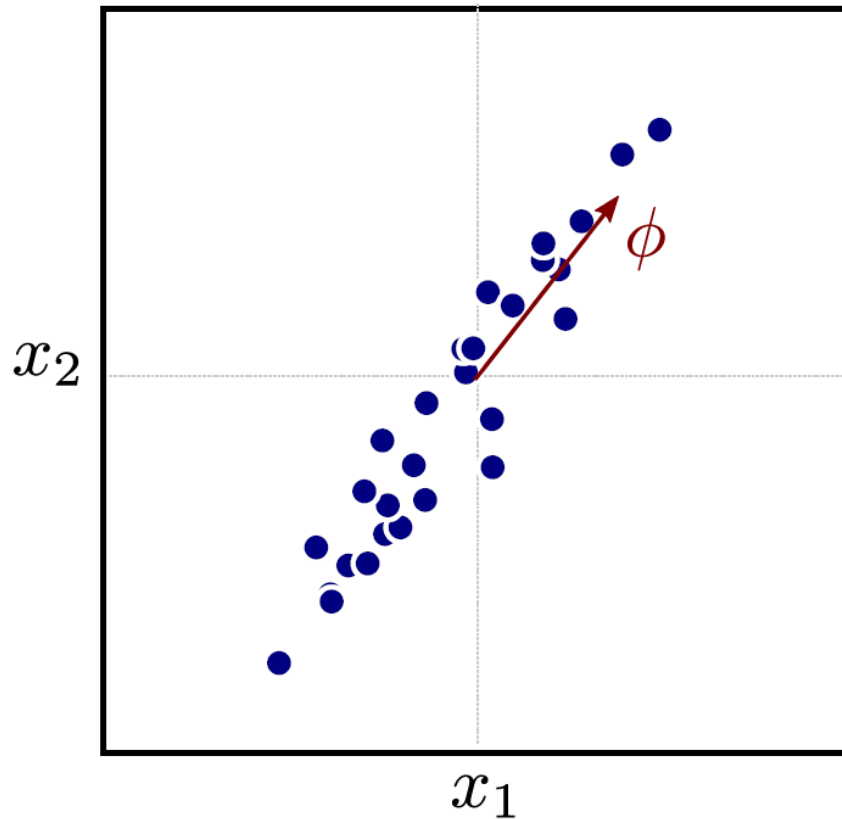
o

$$\mathbf{X}^T \mathbf{X} \hat{\phi} = \lambda \hat{\phi}$$

$$\mathbf{X}^T = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_I]$$

Para calcular el vector  $\phi$ , calculamos el primer autovector (el que tiene un  $\lambda$  mayor) de la matriz de dispersión  $\mathbf{X}^T \mathbf{X}$  ya que el valor de la función objetivo es  $-\lambda$  (comprobar)

# Calculando $h$

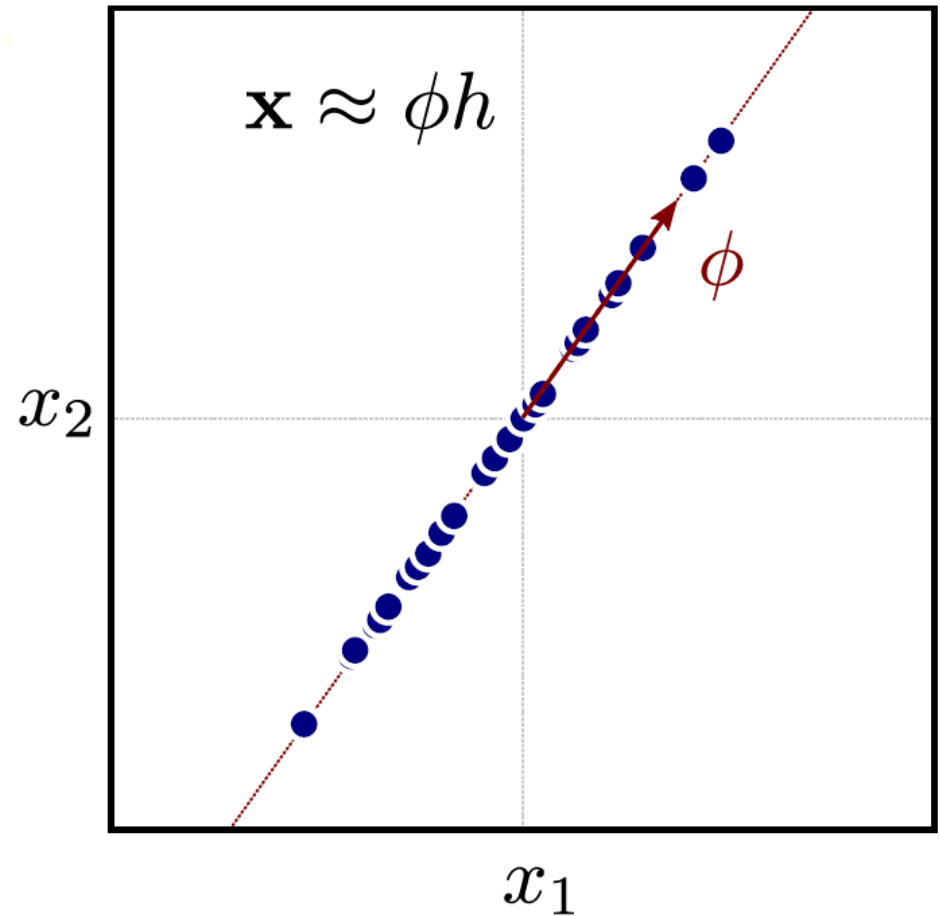
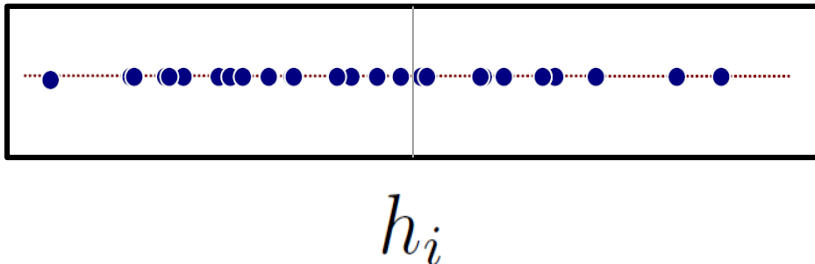


Para calcular el valor oculto, realizar el producto escalar con el vector  $\phi$



# Reconstrucción

$$\mathbf{x}_i \approx \phi h_i$$



Para reconstruir, multiplica la variable oculta  $h$  por el vector  $\phi$ .

# PCA

La misma idea, pero ahora la variable oculta  $\mathbf{h}$  es multidimensional. Cada componente pesa una columna de la matriz  $\Phi$  de forma que los datos se aproximan

$$\mathbf{x}_i \approx \Phi \mathbf{h}_i$$

Conduce a la función de coste:

$$\Phi, \hat{\mathbf{h}}_{1 \dots I} = \underset{\Phi, \mathbf{h}_{1 \dots I}}{\operatorname{argmin}} [E] = \underset{\Phi, \mathbf{h}_{1 \dots I}}{\operatorname{argmin}} \left[ \sum_{i=1}^I (\mathbf{x}_i - \Phi \mathbf{h}_i)^T (\mathbf{x}_i - \Phi \mathbf{h}_i) \right]$$

Esta función no tiene un óptimo único de forma que forzamos la restricción  $\Phi^T \Phi = \mathbf{I}$ . Es decir,  $\Phi$  es una matriz de rotación (truncada)

# Solución PCA

$$\mathbf{h}_i = \Phi^T \mathbf{x}_i$$

Para calcular los vectores ocultos, realiza el producto escalar cada vector de rasgos con cada columna de  $\Phi$ .

---

Para calcular la matriz  $\Phi$ , calcula los primeros  $D_h$  autovectores de la matriz  $\mathbf{X}^T \mathbf{X}$ . Es decir, los asociados a los mayores autovalores

Las funciones base en las columnas de  $\Phi$  son las **componentes principales** y las entradas de  $\mathbf{h}$  reciben el nombre de **carga**

# PCA

## Teorema (Resume los resultados anteriores)

Supongamos que queremos encontrar un conjunto ortonormal de  $L$  vectores base lineal  $\Phi = [\phi_1, \dots, \phi_L]$  y las correspondientes cargas  $h_i \in \mathbb{R}^L$  de los vectores  $\mathbf{x}_i, i \in \{1, \dots, I\}$ ,  $\mathbf{H} = [h_1, \dots, h_I]$  que minimice el error de reconstrucción

$$J(\Phi, \mathbf{H}) = \sum_{i=1}^I \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|^2$$

donde  $\hat{\mathbf{x}}_i = \Phi \mathbf{h}_i$ .

# PCA

Observa que podemos escribir

$$J(\Phi, \mathbf{H}) = \| \mathbf{X}^T - \Phi \mathbf{H} \|_F^2$$

donde  $\mathbf{H}$  es una matriz  $L \times l$  con  $\mathbf{h}_i$  en sus columnas y  $\| \mathbf{A} \|_F$  es la norma de Frobenius de la matriz  $\mathbf{A}$  definida por

$$\| \mathbf{A} \|_F = \sqrt{\sum_{i=1}^m \sum_{j=1}^n a_{ij}^2} = \sqrt{\text{tr}(\mathbf{A}^T \mathbf{A})} = \| \mathbf{A}(:,) \|_2$$

La solución óptima se obtiene con

$$\hat{\Phi} = \mathbf{V}_L$$

donde

$$\mathbf{V}_L$$

contiene los  $L$  autovectores con mayores autovalores de la matriz de covarianza empírica

$$\hat{\Sigma} = \frac{1}{I} \sum_{i=1}^I \mathbf{x}_i \mathbf{x}_i^T = \frac{1}{I} \mathbf{X}^T \mathbf{X}$$

Además la codificación de baja dimensión óptima de los vectores viene dada por

$$\hat{\mathbf{h}}_i = \Phi^T \mathbf{x}_i$$

que es la proyección ortogonal de los vectores de rasgos en las columnas generadas por los autovectores.

La reconstrucción viene dada por

$$\hat{\mathbf{x}}_i = \Phi \hat{\mathbf{h}}_i$$

## Vamos a resumir:

1.  $\mathbf{X}^T$  contiene la matriz de datos por columnas (I columnnas)
2. Calculamos los autovectores de la matriz de covarianza muestral (I es el número de datos)

$$\hat{\Sigma} = \frac{1}{I} \sum_{i=1}^I \mathbf{x}_i \mathbf{x}_i^T = \frac{1}{I} \mathbf{X}^T \mathbf{X}$$

3. Si vamos a utilizar L autovectores, seleccionamos los asociados a los L autovalores más grandes de la matriz de covarianza muestral

$$\hat{\Phi} = \mathbf{V}_L$$

4. ¿Cuáles son las variables latentes?

$$\hat{\mathbf{h}}_i = \Phi^T \mathbf{x}_i$$

5. Además, la codificación de baja dimensión óptima de los vectores viene dada por

$$\hat{\mathbf{x}}_i = \Phi \hat{\mathbf{h}}_i$$



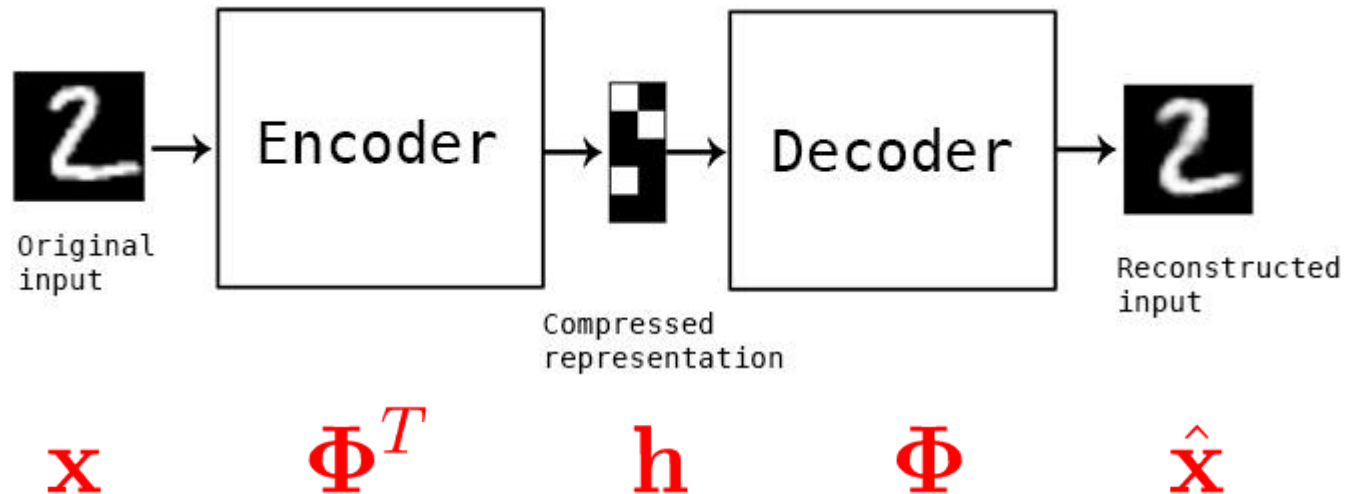
Fíjate, tenemos

$$\hat{\mathbf{h}}_i = \Phi^T \mathbf{x}_i$$

y

$$\hat{\mathbf{x}}_i = \Phi \hat{\mathbf{h}}_i$$

es la estructura de un autoencoder



## Una nota de precaución

Estamos utilizando los autovectores de

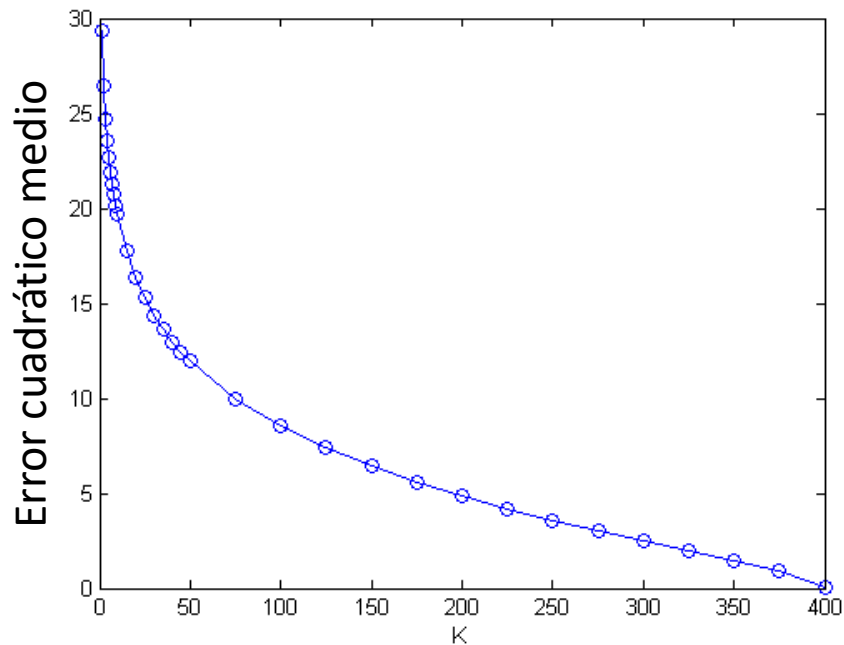
$$\mathbf{X}^T \mathbf{X}$$

que son los mismos que los de

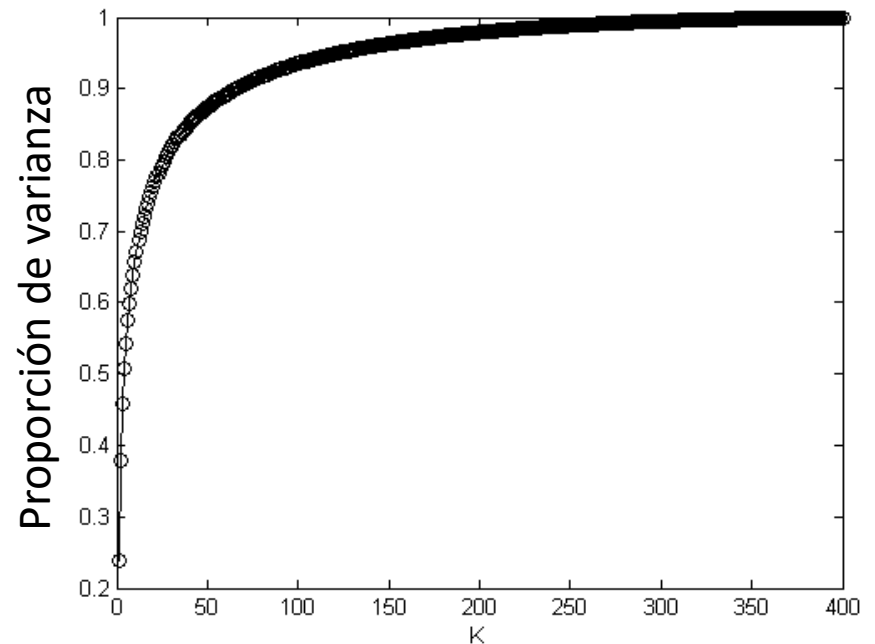
$$\frac{1}{I} \mathbf{X}^T \mathbf{X}$$

no obstante, los autovalores no son los mismos.

# Ejemplo PCA



Errores de reconstrucción en  
función del número de  
autocarar utilizado



Proporción de varianza total (suma de  
los autovalores) en función del número  
de autovalores (autocarar) utilizado

# Ejemplo PCA

Reconstrucción de una cara con 2 bases



Reconstrucción de una cara con 10 bases



Reconstrucción de una cara con 100 bases



Reconstrucción de una cara con 400 bases



# PCA para alta dimensión

**Problema:** como lo hemos descrito, PCA tiene un problema importante. Necesitamos calcular los autovectores de la matriz

$$\mathbf{X}^T \mathbf{X} \quad \text{recuerda} \quad \mathbf{X}^T = [x_1, x_2, \dots, x_I]_{D \times I}$$

que tiene tamaño  $D \times D$ . Los datos visuales son altamente dimensionales y, por tanto, la matriz de dispersión de tamaño  $D \times D$  puede ser muy grande, mucho más grande que la matriz  $\mathbf{X}\mathbf{X}^T$  de tamaño  $I \times I$ .

**¿Podríamos trabajar con  $\mathbf{X}\mathbf{X}^T$  para calcular los autovectores de  $\mathbf{X}^T\mathbf{X}$  ?**

# PCA para alta dimensión

De la descomposición SVD (ver apéndice)

$$\mathbf{X}^T = \mathbf{V}\mathbf{D}\mathbf{U}^T$$

**U** es una matriz con columnas ortonormales

**D** es una matriz diagonal con elementos no negativos

**V** es una matriz con columnas ortonormales

# PCA para alta dimensión

Tenemos, por tanto,

$$\mathbf{X}^T \mathbf{X} = \mathbf{V} \mathbf{D} \mathbf{U}^T \mathbf{U} \mathbf{D} \mathbf{V}^T = \mathbf{V} \mathbf{D} \mathbf{D} \mathbf{V}^T$$

de donde las columnas de  $\mathbf{V}$  son los autovectores de  $\mathbf{X}^T \mathbf{X}$ . Es decir, es nuestra matriz  $\Phi$  (pruébalo).

Además, de la descomposición SVD también obtenemos (pruébalo)

$$\mathbf{X}^T \mathbf{U} = \mathbf{V} \mathbf{D}^T$$

de donde concluimos finalmente que podemos escribir las componentes principales como sumas ponderadas de los datos. Es decir,

$$\Phi = \mathbf{X}^T \Psi$$

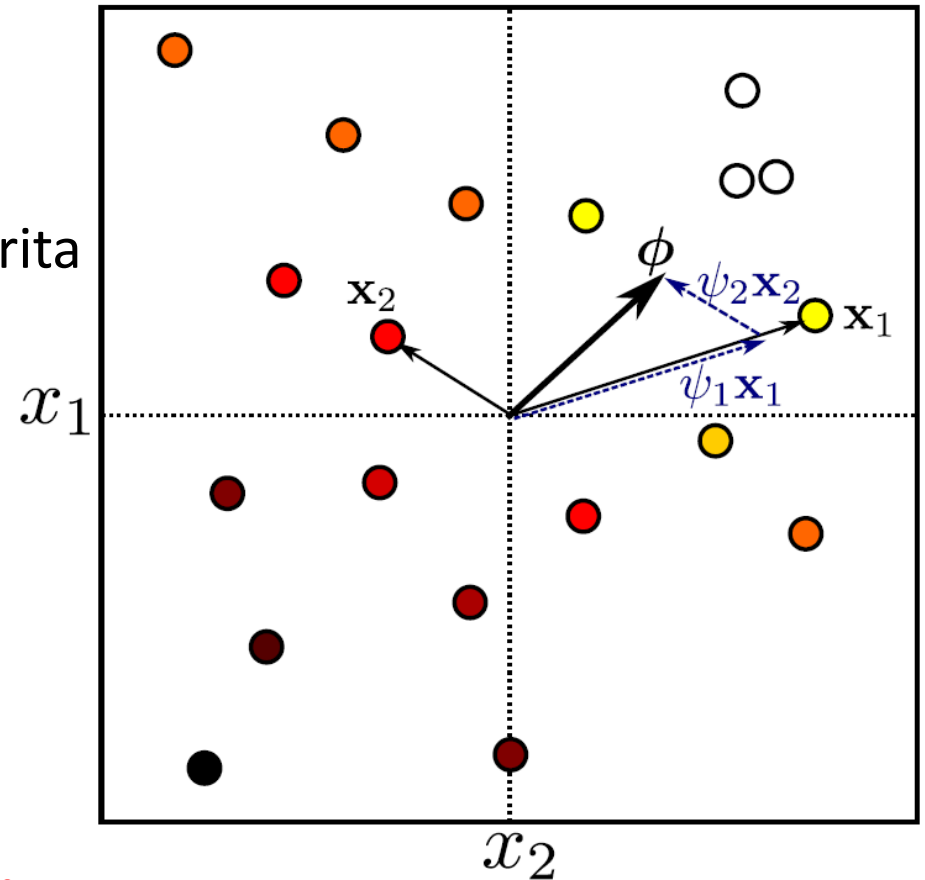
# Interpretación geométrica

$$\Phi = X^T \Psi$$

**Si te pierdes en la diapositiva anterior, éste es su resumen:**

Cada columna de  $\Phi$  puede ser descrita como una suma ponderada de las muestras originales.

Los pesos están en las correspondientes columnas de la nueva matriz (variable)  $\Psi$ .



**Por tanto, con encontrar  $\Psi$  es suficiente.**



# Motivación

Por tanto, podemos reparametrizar las componentes principales como sumas ponderadas de los datos

$$\Phi = X^T \Psi$$

**Observa:** Si el número de muestras  $I$  es menor que la dimensión de las observaciones  $D$  entonces las columnas de  $\Psi$  tienen menos componentes que las de  $\Phi$  y es menos costoso calcular  $\Psi$  (y luego  $\Phi$  usando la fórmula de arriba) que calcular  $\Phi$  directamente.

¿ Cómo calculamos  $\Psi$ ?

# Solución

¿Cómo calculamos  $\psi_i$ ? Sea  $\psi_i$  un autovector de un conjunto de vectores ortonormales de  $\mathbf{X}\mathbf{X}^T/I$ , entonces

$$\frac{1}{I}\mathbf{X}\mathbf{X}^T\psi_i = \lambda_i\psi_i$$

Lo que implica que

$$\frac{1}{I}\mathbf{X}^T\mathbf{X}\mathbf{X}^T\psi_i = \mathbf{X}^T\lambda_i\psi_i$$

Además el número de autovectores no nulos de  $\mathbf{X}\mathbf{X}^T$  y  $\mathbf{X}^T\mathbf{X}$  conciden. Por tanto,

$$\phi_i = \mathbf{X}^T\psi_i$$

es un autovector de  $\mathbf{X}^T\mathbf{X}$  que podemos normalizar observando que

$$\phi_i^T\phi_i = \psi_i^T\mathbf{X}\mathbf{X}^T\psi_i = \lambda_i I$$

# Solución

Basta, por tanto, con

1. Encontrar las soluciones de

$$\frac{1}{I} \mathbf{X} \mathbf{X}^T \boldsymbol{\psi}_i = \lambda_i \boldsymbol{\psi}_i$$

2. y definir

$$\boldsymbol{\phi}_i = \frac{1}{\sqrt{\lambda_i I}} \mathbf{X}^T \boldsymbol{\psi}_i$$

para obtener el conjunto de vectores ortonormales que definen  $\Phi$ .

Observa que

$$\boldsymbol{\phi}_i^T \boldsymbol{\phi}_j = \frac{1}{\sqrt{\lambda_i \lambda_j}} \boldsymbol{\psi}_i^T \mathbf{X} \mathbf{X}^T \boldsymbol{\psi}_j = \delta_{ij}$$

Hemos pues obtenido los autovectores de  $\mathbf{X}^T \mathbf{X} / I$  a través de los de  $\mathbf{X} \mathbf{X}^T / I$  que se calculan mucho más rápidamente.

# IV. Núcleos (Kernels)

Antes de analizar los Kernels PCA vamos a introducir el concepto de **núcleo de Mercer** . Una función  $k(\mathbf{x}, \mathbf{x}')$  que cumple que

$$\mathbf{K} = \begin{pmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & \dots & k(\mathbf{x}_1, \mathbf{x}_n) \\ \vdots & \vdots & \vdots \\ k(\mathbf{x}_n, \mathbf{x}_1) & \dots & k(\mathbf{x}_n, \mathbf{x}_n) \end{pmatrix}$$

es definida positiva para cualquier  $\{\mathbf{x}_i\}_{i=1}^N$  recibe el nombre de núcleo de Mercer.

Ejemplos de núcleo de Mercer. Existen muchísimos más, algunos relacionados con hileras de caracteres,

$$k(\mathbf{x}, \mathbf{x}') = \exp \left( -\frac{1}{\sigma^2} \|\mathbf{x} - \mathbf{x}'\|^2 \right)$$

$$k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}'$$

# V. Kernel PCA

Introduzcamos el concepto de KPCA.

Supongamos que queremos realizar una transformación de los rasgos y después aplicarles PCA

$$\mathbf{x}_i = \begin{pmatrix} x_{1i} \\ x_{2i} \\ \vdots \\ x_{Di} \end{pmatrix} \xrightarrow{\mathbf{t}} \mathbf{z}_i = \mathbf{t}(\mathbf{x}_i) = \begin{pmatrix} t_1(\mathbf{x}_i) \\ t_2(\mathbf{x}_i) \\ \vdots \\ t_{D'}(\mathbf{x}_i) \end{pmatrix}$$

Observa que las dimensiones de un vector de rasgos y su transformado pueden no ser las mismas. **Vamos a ver que la dimensión  $D'$  puede ser infinito!!!**

# V. Kernel (Núcleo) PCA

Podríamos hacer, por ejemplo, la siguiente transformación

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \xrightarrow{\mathbf{t}} \mathbf{z} = \mathbf{t}(\mathbf{x}) = \begin{pmatrix} x_1 \\ x_2 \\ x_1 x_2 \\ x_1^2 \\ x_2^2 \end{pmatrix}$$

Observa que las dimensiones de un vector de rasgos y su transformado pueden no ser las mismas.

# Solución

Sea  $\mathbf{Z}^T = [\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_I]$  y sea  $\xi_i$  un autovector de  $\mathbf{Z}\mathbf{Z}^T/I$ , observa que estamos siguiendo la misma aproximación que cuando teníamos más filas que columnas en  $\mathbf{X}$ , entonces

$$\frac{1}{I} \mathbf{Z}\mathbf{Z}^T \xi_i = \lambda_i \xi_i$$

de donde multiplicando por  $\mathbf{Z}^T$

$$\frac{1}{I} \mathbf{Z}^T \mathbf{Z}\mathbf{Z}^T \xi_i = \lambda_i \mathbf{Z}^T \xi_i$$

y tras la transformación  $\mathbf{t}(\mathbf{x})$  tenemos para los datos transformados

$$\Phi = \left[ \frac{1}{\sqrt{\lambda_1 I}} \mathbf{Z}^T \xi_1, \dots, \frac{1}{\sqrt{\lambda_I I}} \mathbf{Z}^T \xi_I \right]$$

# Solución

¿Para resolver en  $\xi_i$   $\frac{1}{I} \mathbf{Z} \mathbf{Z}^T \xi_i = \lambda_i \xi_i$  necesitamos conocer  $\mathbf{Z}$  explícitamente?

NO!!.

Observa: sea

$$\mathbf{K} = \mathbf{Z} \mathbf{Z}^T$$

¿Quiénes son los elementos de  $\mathbf{K}$ ?

$$K(i, j) = \mathbf{t}^T(\mathbf{x}_i) \mathbf{t}(\mathbf{x}_j) = k(\mathbf{x}_i, \mathbf{x}_j)$$

Sólo necesitamos que la matriz  $\mathbf{K}$  sea definida positiva. NO necesitamos saber quien es  $\mathbf{t}(\mathbf{x})$  exactamente

$$\mathbf{K} = \begin{pmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & \dots & k(\mathbf{x}_1, \mathbf{x}_I) \\ \vdots & \vdots & \vdots \\ k(\mathbf{x}_I, \mathbf{x}_1) & \dots & k(\mathbf{x}_I, \mathbf{x}_I) \end{pmatrix}$$



# Solución

¿Cuál es la proyección de un vector  $\mathbf{x}$ ?

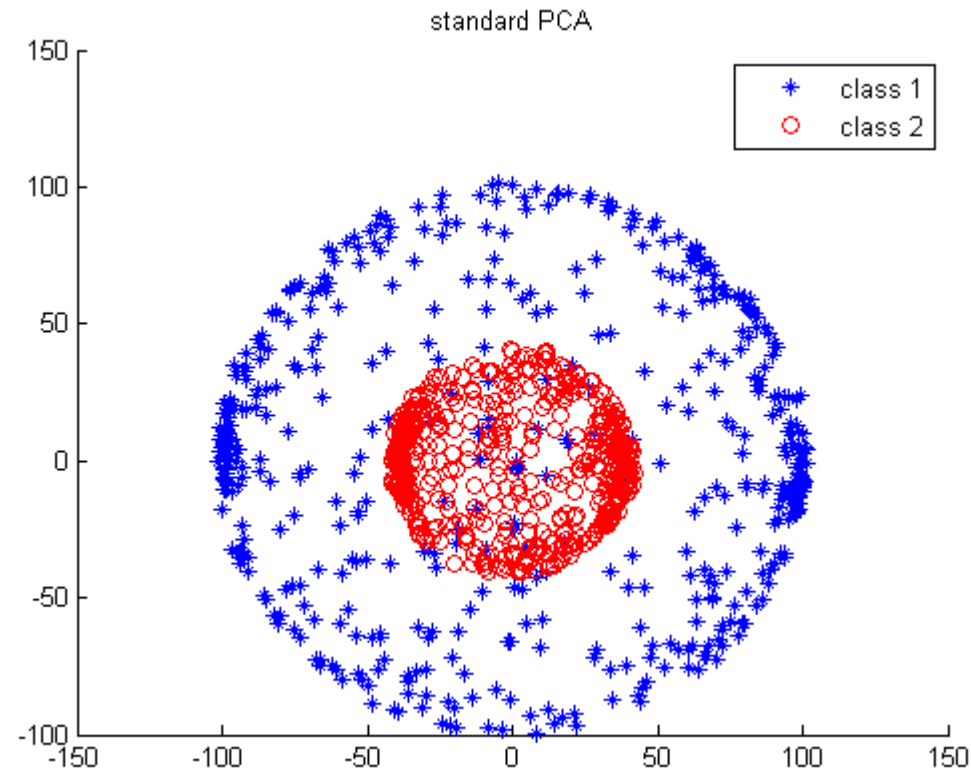
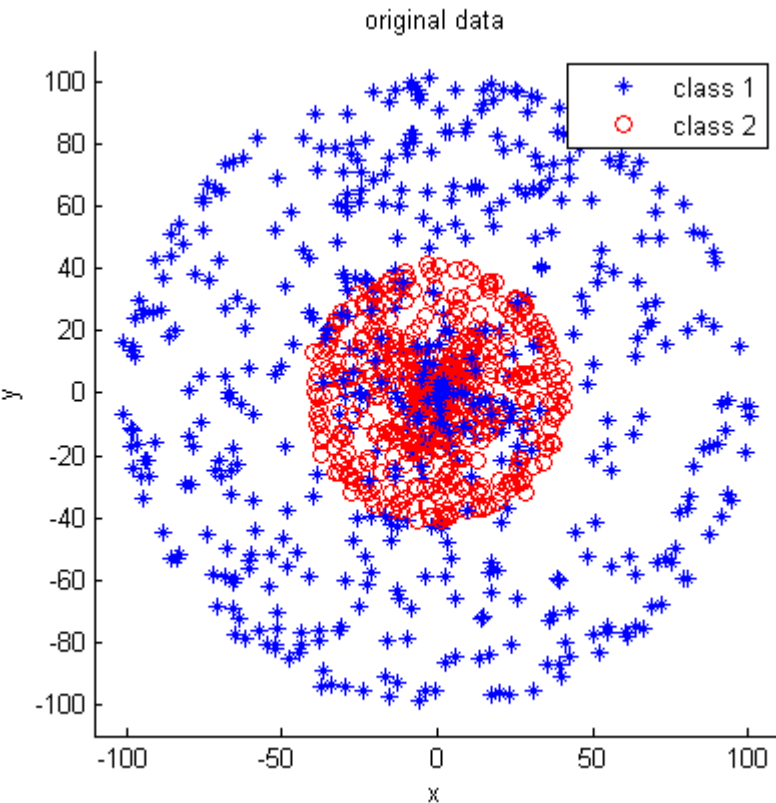
$$\begin{array}{ccc}
 \text{Transformado} & & \text{Proyectado} \\
 \text{a} & & \text{a}
 \end{array}$$

$$\mathbf{x} \xrightarrow{\quad} \mathbf{t}(\mathbf{x}) \xrightarrow{\quad} \begin{bmatrix} \frac{1}{\sqrt{\lambda_1 I}} \boldsymbol{\xi}_1^T \\ \frac{1}{\sqrt{\lambda_2 I}} \boldsymbol{\xi}_2^T \\ \vdots \\ \frac{1}{\sqrt{\lambda_I I}} \boldsymbol{\xi}_I^T \end{bmatrix} \mathbf{Z} \mathbf{t}(\mathbf{x}) = \begin{bmatrix} \frac{1}{\sqrt{\lambda_1 I}} \boldsymbol{\xi}_1^T \\ \frac{1}{\sqrt{\lambda_2 I}} \boldsymbol{\xi}_2^T \\ \vdots \\ \frac{1}{\sqrt{\lambda_I I}} \boldsymbol{\xi}_I^T \end{bmatrix} \begin{bmatrix} k(\mathbf{x}, \mathbf{x}_1) \\ k(\mathbf{x}, \mathbf{x}_2) \\ \vdots \\ k(\mathbf{x}, \mathbf{x}_I) \end{bmatrix}$$

¿Conocemos  $\mathbf{t}(\mathbf{x})$ ,  $\mathbf{Z}^T \boldsymbol{\xi}_i$  y la proyección de  $\mathbf{t}(\mathbf{x})$  en el subespacio generado por  $\mathbf{Z}^T \boldsymbol{\xi}_i$ ,  $i = 1, \dots, I$  ?.

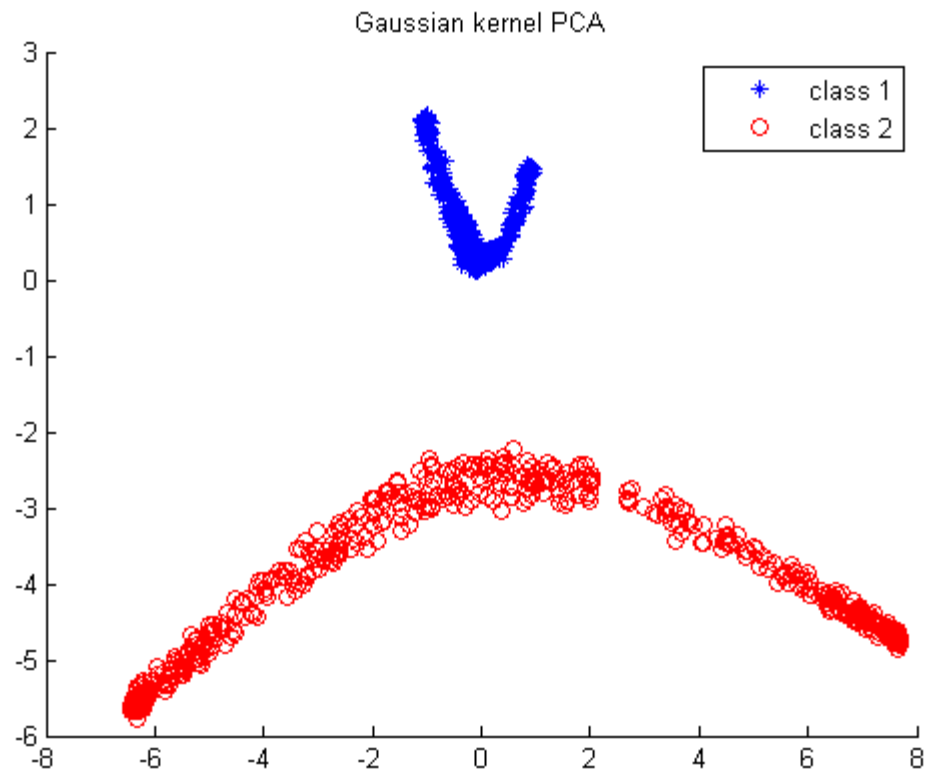
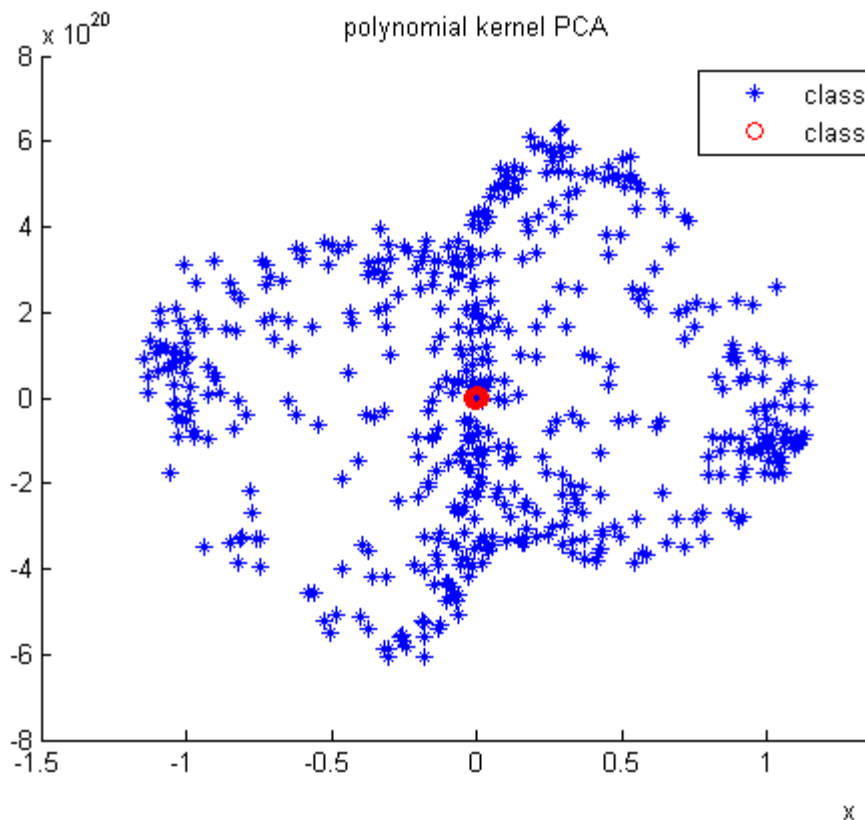
Desgraciadamente no. Sólo conocemos las coordenadas en el dominio proyectado.

# Kernel PCA



Datos originales y su proyección standard usando PCA.  
Software utilizado kPCA\_v3.1

# Kernel PCA



PCA usando un núcleo polinómico y un núcleo Gaussiano.  
Software utilizado kPCA\_v3.1

# Kernel PCA

En el desarrollo anterior para calcular  $\mathbf{K} = \mathbf{Z}\mathbf{Z}^T$  hemos supuesto que

$$\frac{1}{I} \sum_{i=1}^I \mathbf{t}(x_i) = \underline{0}$$

Pero ¿qué ocurre si la media no es cero o si incluso no conocemos explícitamente la función de proyección  $\mathbf{t}$ ?

Tendríamos que centrar los nuevos rasgos y utilizar una nueva matriz  $\tilde{\mathbf{K}}$ . Como puedes comprobar en las siguientes diapositivas esto no es un problema.

# Kernel PCA

$$\begin{aligned}\tilde{\mathbf{K}} &= \left( \begin{bmatrix} \mathbf{t}^T(\mathbf{x}_1) \\ \mathbf{t}^T(\mathbf{x}_2) \\ \vdots \\ \mathbf{t}^T(\mathbf{x}_I) \end{bmatrix} - \frac{1}{I} \mathbf{1}_I \mathbf{1}_I^T \begin{bmatrix} \mathbf{t}^T(\mathbf{x}_1) \\ \mathbf{t}^T(\mathbf{x}_2) \\ \vdots \\ \mathbf{t}^T(\mathbf{x}_I) \end{bmatrix} \right) \\ &\quad \times \left( \begin{bmatrix} \mathbf{t}^T(\mathbf{x}_1) \\ \mathbf{t}^T(\mathbf{x}_2) \\ \vdots \\ \mathbf{t}^T(\mathbf{x}_I) \end{bmatrix} - \frac{1}{I} \mathbf{1}_I \mathbf{1}_I^T \begin{bmatrix} \mathbf{t}^T(\mathbf{x}_1) \\ \mathbf{t}^T(\mathbf{x}_2) \\ \vdots \\ \mathbf{t}^T(\mathbf{x}_I) \end{bmatrix} \right)^T \\ &= \mathbf{K} - \frac{1}{I} \mathbf{1}_I \mathbf{1}_I^T \mathbf{K} - \mathbf{K} \frac{1}{I} \mathbf{1}_I \mathbf{1}_I^T + \frac{1}{I} \mathbf{1}_I \mathbf{1}_I^T \mathbf{K} \frac{1}{I} \mathbf{1}_I \mathbf{1}_I^T\end{aligned}$$

con  $\mathbf{1}_I = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}_{I \times 1}$

# Kernel PCA

¿Cuál es la proyección de un vector  $\mathbf{x}$ ?

Tendríamos que quitarle vector media  $\mu$  en el dominio transformado (que no conocemos)

Transformado  $\mathbf{x} \xrightarrow{a} \mathbf{t}(\mathbf{x})$     Proyectado  $\mathbf{t}(\mathbf{x}) \xrightarrow{a} \begin{bmatrix} \frac{1}{\sqrt{\lambda_1 I}} \xi_1^T \\ \frac{1}{\sqrt{\lambda_2 I}} \xi_2^T \\ \vdots \\ \frac{1}{\sqrt{\lambda_I I}} \xi_i^T \end{bmatrix} \mathbf{Z}(\mathbf{t}(\mathbf{x}) - \mu)$

Autovalores obtenidos a partir de  $\tilde{\mathbf{K}}$

$$= \begin{bmatrix} \frac{1}{\sqrt{\lambda_1 I}} \xi_1^T \\ \frac{1}{\sqrt{\lambda_2 I}} \xi_2^T \\ \vdots \\ \frac{1}{\sqrt{\lambda_I I}} \xi_i^T \end{bmatrix} \left( \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}) \\ k(\mathbf{x}_2, \mathbf{x}) \\ \vdots \\ k(\mathbf{x}_I, \mathbf{x}) \end{bmatrix} - \frac{1}{I} \sum_{i=1}^I \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_i) \\ k(\mathbf{x}_2, \mathbf{x}_i) \\ \vdots \\ k(\mathbf{x}_I, \mathbf{x}_i) \end{bmatrix} \right)$$

# Apéndice

## Matrices Ortogonales

Una matriz ortogonal es aquella cuya inversa es igual a su traspuesta

$$A^{-1} = A^T, \text{ o lo que es lo mismo, } A^T A = A A^T = I$$

## Autovalores y Autovectores

Dada una matriz A cuadrada se definen sus autovalores como la solución de

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

Los autovectores v son los vectores soluciones de

$$Av = \lambda v$$



# Matrices Definidas Positivas y Formas Cuadráticas

Se dice que una matriz real,  $A_{N \times N}$  es definida positiva (o semidefinida positiva) si la forma cuadrática

$$x^T A x, \quad \forall x \neq 0$$

es positiva (o no negativa), respectivamente

Observa que para cualquier matriz  $Z$ ,  $x^T Z^* Z^T x \geq 0$ . Pruébalo

## Matrices simétricas. Autovalores, autovectores y formas cuadráticas. Descomposición de Jordan.

Como sabemos, para una matriz  $A$  de tamaño  $M \times M$ , la ecuación de los autovectores está definida mediante

$$A u_i = \lambda_i u_i \quad i = 1, \dots, M$$

$u_i$  recibe el nombre de autovector y  $\lambda_i$  autovalor.

Un caso de especial interés corresponde a las matrices  $A$  reales que son simétricas (su inversa, si existe también es simétrica).

Resultado: si  $A$  es simétrica todos sus autovalores son reales  
Observa que para cualquier matriz  $A$   $AA^T$  es simétrica

---

## Teorema de Descomposición Espectral o Teorema de Descomposición de Jordan:

Si  $A$  es simétrica entonces podemos escribir

$$AU = U\Lambda$$

Donde  $\Lambda$  es una matriz diagonal con los autovalores de  $A$  y  $U$  es una matriz ortogonal cuyos vectores columna son autovectores de  $A$ .

---

Observa que de los resultados anteriores podemos escribir

$$A = U\Lambda U^T$$

En Matlab usamos eig para obtener esta descomposición de una matriz

Tendremos que

$$A^{-1} = U\Lambda^{-1}U^T$$

---

Consideremos la transformación  $U\bar{x} = x$

La multiplicación por  $U$  puede entenderse como una transformación a una representación basada en vectores ortogonales que conserva la norma.

Se cumple

$$\bar{x}^T \bar{x} = x^T x$$

$$x^t Ax = x^t U U^t A U U^t x = \bar{x}^T \Lambda \bar{x} = \sum_i \lambda_i \bar{x}_i^2$$

Todo lo anterior nos lleva a

$$A = \sum_{i=1}^M \lambda_i u_i u_i^T \quad y \quad A^{-1} = \sum_{i=1}^M \frac{1}{\lambda_i} u_i u_i^T$$

Además puede probarse fácilmente que

$$|A| = \prod_{i=1}^M \lambda_i \quad y \quad tr(A) = \sum_{i=1}^M \lambda_i$$

## Descomposición por Valores Singulares

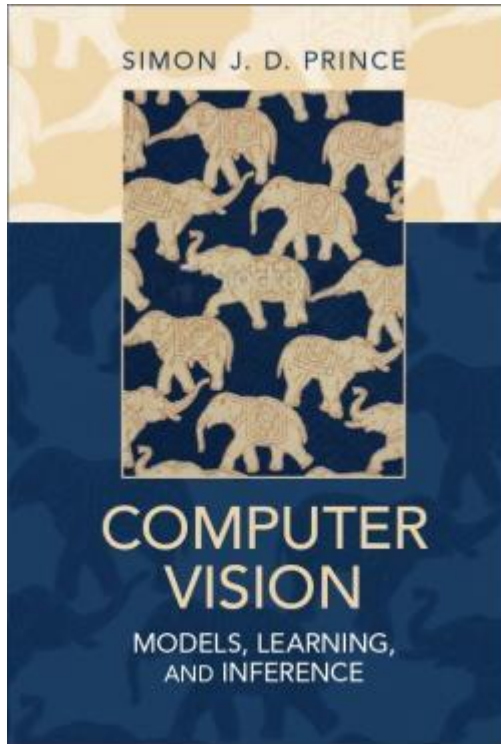
### Teorema de descomposición en valores singulares

Si  $A$  es una matriz de dimensiones  $n \times p$  y rango  $r$  entonces

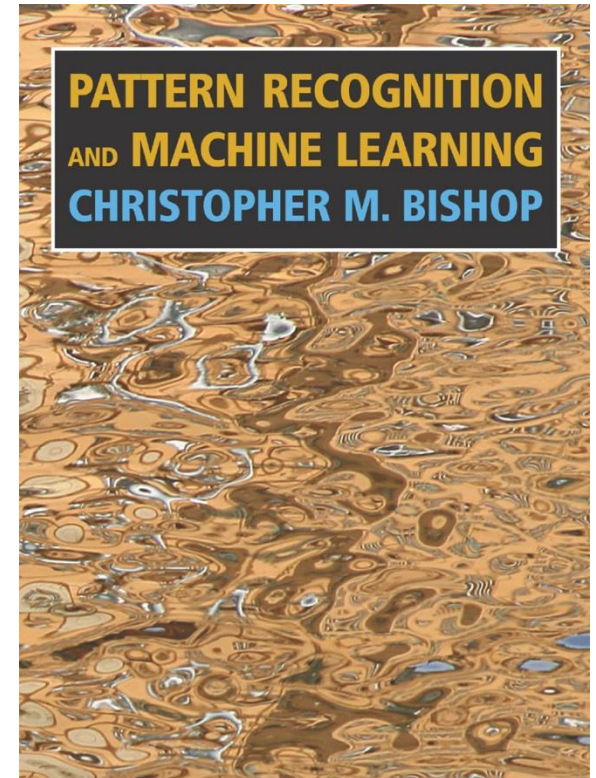
$$A = U L V^T$$

donde  $U_{n \times r}$  y  $V_{p \times r}$  son matrices con columnas ortonormales ( $U^T U = V^T V = I_r$ ) y  $L$  es una matriz diagonal con elementos positivos.

# Bibliografía



<http://www.computervisionmodels.com/>

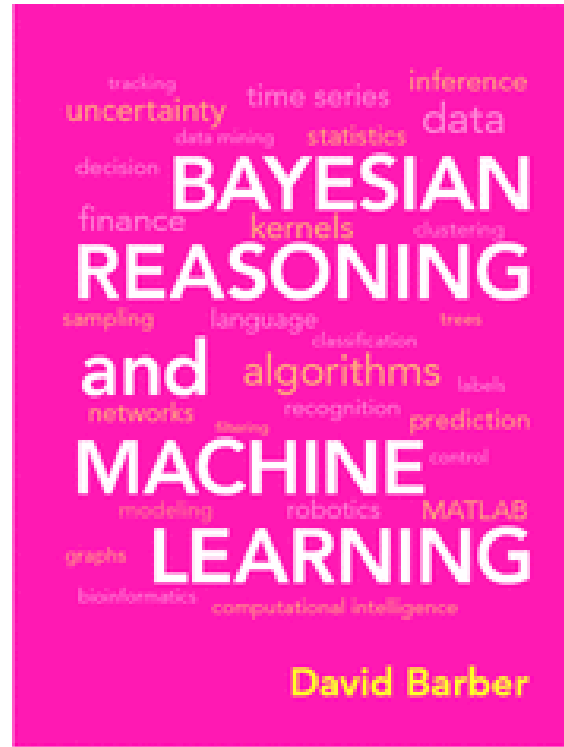
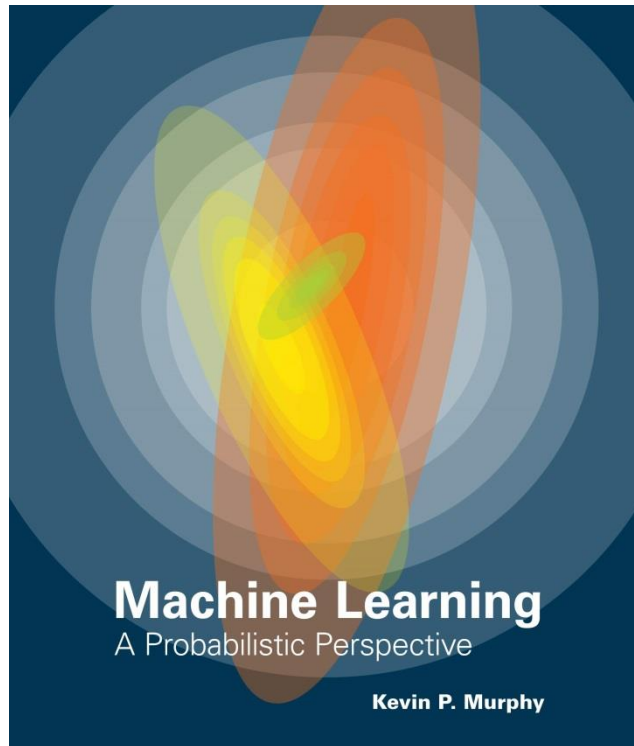


<http://research.microsoft.com/en-us/um/people/cmbishop/prml/>

# Bibliografía

<http://www.cs.ubc.ca/~murphyk/MLbook/>

<http://www.inference.phy.cam.ac.uk/mackay/itila/>



<http://www.cs.ucl.ac.uk/staff/d.barber/brml/>