

APRENDIZAJE AUTOMÁTICO

Universidad de Zaragoza

2017 – 2018

Índice

- Regresión monovariable y multivariable
- Descenso de gradiente
- Ecuación normal
- Regresión polinómica y otras

Tema 1. Regresión

30231 - Aprendizaje Automático
Grado en Ingeniería Informática
Escuela de Ingeniería y Arquitectura

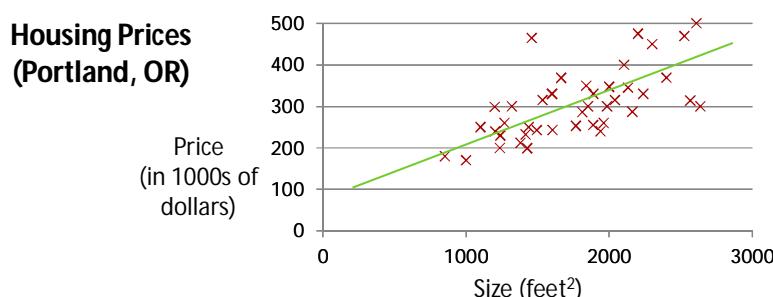


Créditos de transparencias y figuras:

- Andrew Ng, *Machine Learning*, Stanford AI Lab.
<https://www.coursera.org/course/ml>
- Kevin P. Murphy, *Machine Learning: A Probabilistic Perspective*, MIT Press, 2012



Regresión Monovariable



Aprendizaje Supervisado
Se conoce la “respuesta correcta” para cada ejemplo de entrenamiento

Regresión
Predecir una variable continua
 $\hat{y} = h_{\theta}(x) = \theta_0 + \theta_1 x$

Regresión Multivariable

x_1 Size (feet ²)	x_2 Number of bedrooms	x_3 Number of floors	x_4 Age of home (years)	y Price (\$1000)
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178
...

$$\hat{y} = h_{\theta}(\mathbf{x}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_4 x_4$$

- En general:

$$\theta = (\theta_0, \theta_1, \dots, \theta_D)^T$$

$$\hat{y} = h_{\theta}(\mathbf{x}) = \theta^T \mathbf{x}$$

$$\mathbf{x} = (x_0, x_1, \dots, x_D)^T$$

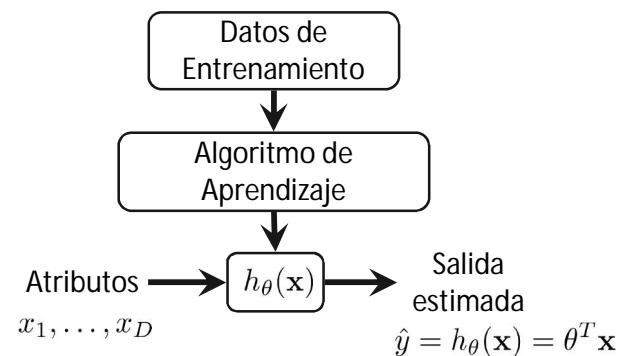
$$x_0 = 1$$



Nomenclatura

- Muestras de entrenamiento $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^N$
- Variables de entrada o atributos (*input / features*) $x_1, \dots, x_D \quad x_0 = 1$
 $\mathbf{x} = (x_0, x_1, \dots, x_D)^T$
- Variable de salida u objetivo (*output / target*) y
- Parámetros o pesos (*weights*) o a veces:
 $\theta = (\theta_0, \theta_1, \dots, \theta_D)^T$
 $\mathbf{w} = (w_0, w_1, \dots, w_D)^T$
- Hipótesis $h_{\theta}(\mathbf{x}) = \theta^T \mathbf{x} = \sum_{i=0}^D \theta_i x_i$

Regresión



- ¿Cómo elegimos el valor de los parámetros θ ?

Función de Coste

- Suma de errores cuadáticos

$$J(\theta) = \frac{1}{2} \sum_{i=1}^N (h_{\theta}(\mathbf{x}^{(i)}) - y^{(i)})^2$$

- Estimación por mínimos cuadrados (LMS)

$$\hat{\theta} = \arg \min_{\theta} J(\theta)$$

- Notas:

$$SSE(\theta) = \sum_{i=1}^N (h_{\theta}(\mathbf{x}^{(i)}) - y^{(i)})^2$$

Sum of Squared Errors
Suma de errores cuadráticos

$$MSE(\theta) = \frac{SSE(\theta)}{N} = \frac{1}{N} \sum_{i=1}^N (h_{\theta}(\mathbf{x}^{(i)}) - y^{(i)})^2$$

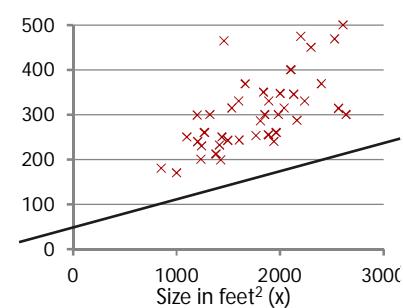
Mean Square Error
Error cuadrático medio

$$RMSE(\theta) = \sqrt{\frac{SSE(\theta)}{N}}$$

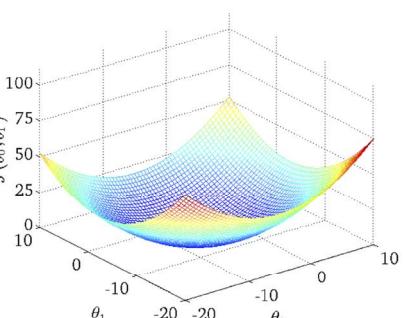
Root-Mean-Square Error
Error RMS

Función de Coste

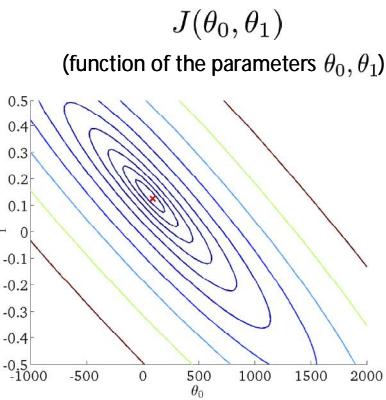
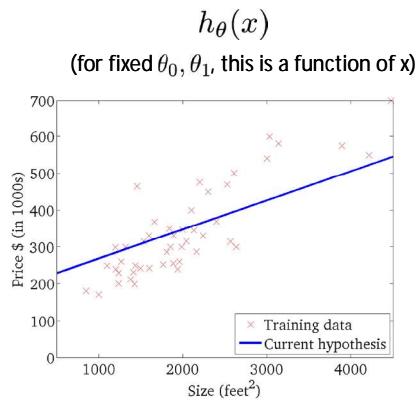
$h_{\theta}(x)$
(dados θ_0, θ_1 , es una función de x)



$J(\theta_0, \theta_1)$
(función de los parámetros θ_0, θ_1)



Función de Coste



Algoritmo de Descenso de Gradiente

Coste

$$J(\theta) = \frac{1}{2} \sum_{i=1}^N (h_{\theta}(\mathbf{x}^{(i)}) - y^{(i)})^2$$

$$h_{\theta}(\mathbf{x}^{(i)}) = \theta^T \mathbf{x}^{(i)}$$

Gradiente

$$\mathbf{g}(\theta) = \frac{\partial}{\partial \theta} J(\theta) = \begin{pmatrix} \frac{\partial J(\theta)}{\partial \theta_0} \\ \vdots \\ \frac{\partial J(\theta)}{\partial \theta_D} \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^N (h_{\theta}(\mathbf{x}^{(i)}) - y^{(i)}) x_0^{(i)} \\ \vdots \\ \sum_{i=1}^N (h_{\theta}(\mathbf{x}^{(i)}) - y^{(i)}) x_D^{(i)} \end{pmatrix}$$

Descenso de gradiente (Batch):

Repetir

$$\theta_{k+1} := \theta_k - \alpha \mathbf{g}(\theta_k)$$

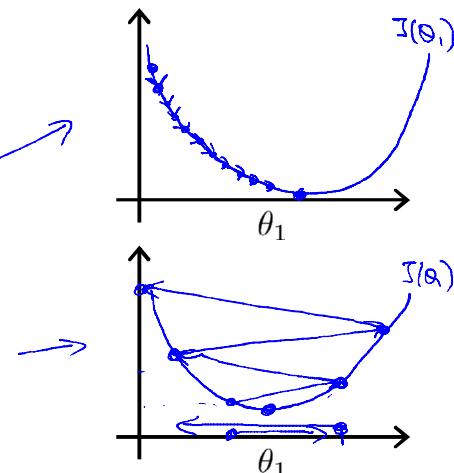
α : Factor de aprendizaje

Hasta que converja

¿Factor de Aprendizaje?

$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

If α is too small, gradient descent can be slow.



If α is too large, gradient descent can overshoot the minimum. It may fail to converge, or even diverge.

Cálculo Vectorizado (Matlab)

Matriz de diseño:

$$X = \begin{pmatrix} 1 & x_1^{(1)} & \dots & x_D^{(1)} \\ 1 & x_1^{(2)} & \dots & x_D^{(2)} \\ \vdots & \ddots & \ddots & \vdots \\ 1 & x_1^{(N)} & \dots & x_D^{(N)} \end{pmatrix} \quad \theta = \begin{pmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_D \end{pmatrix} \quad \mathbf{y} = \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{pmatrix}$$

Salidas predichas:

$$\hat{\mathbf{y}} = \begin{pmatrix} h_{\theta}(\mathbf{x}^{(1)}) \\ \vdots \\ h_{\theta}(\mathbf{x}^{(N)}) \end{pmatrix} = X\theta$$

Residuos:

$$\mathbf{r} = \begin{pmatrix} r^{(1)} \\ \vdots \\ r^{(N)} \end{pmatrix} = (X\theta - \mathbf{y})$$

Coste:

$$\begin{aligned} J(\theta) &= \frac{1}{2} \sum_{i=1}^N (h_{\theta}(\mathbf{x}^{(i)}) - y^{(i)})^2 \\ &= \frac{1}{2} (\mathbf{X}\theta - \mathbf{y})^T (\mathbf{X}\theta - \mathbf{y}) = \frac{1}{2} \mathbf{r}^T \mathbf{r} \end{aligned}$$

Gradiente:

$$\begin{aligned} \mathbf{g}(\theta) &= X^T (\mathbf{X}\theta - \mathbf{y}) \\ &= X^T \mathbf{X}\theta - X^T \mathbf{y} \end{aligned}$$

Ecuación Normal

- Con coste cuadrático, la solución que minimiza el coste puede calcularse analíticamente

$$\begin{aligned} J(\theta) &= \frac{1}{2} \sum_{i=1}^N \left(h_\theta(\mathbf{x}^{(i)}) - y^{(i)} \right)^2 \\ &= \frac{1}{2} (\mathbf{X}\theta - \mathbf{y})^T (\mathbf{X}\theta - \mathbf{y}) \\ &= \frac{1}{2} \|\mathbf{X}\theta - \mathbf{y}\|_2^2 \end{aligned}$$

$$\hat{\theta} = \arg \min_{\theta} J(\theta) = \arg \min_{\theta} \frac{1}{2} \|\mathbf{X}\theta - \mathbf{y}\|_2^2$$

- En el mínimo se cumplirá:

$$\mathbf{g}(\hat{\theta}) = \mathbf{X}^T \mathbf{X} \hat{\theta} - \mathbf{X}^T \mathbf{y} = 0$$

Ecuación Normal

- Ecuación Normal: $\mathbf{X}^T \mathbf{X} \hat{\theta} = \mathbf{X}^T \mathbf{y}$
- Solución: $\hat{\theta} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} = \mathbf{X}^+ \mathbf{y}$ Pseudo-inversa de Moore-Penrose

- En Matlab:

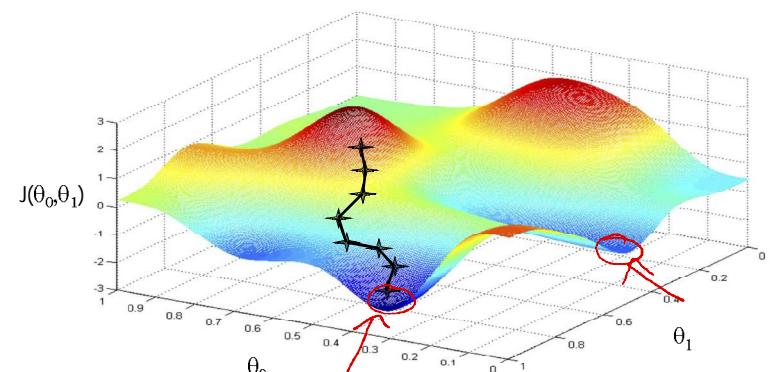
```
theta = inv(X'*X)*X'*y % Nunca se programa así
theta = pinv(X)*y       % Algo mejor
theta = X \ y             % Mucho mejor así
```

Comparación de los dos Algoritmos

- | | |
|------------------------------|------------------------|
| <u>Descenso de Gradiente</u> | <u>Ecuación Normal</u> |
|------------------------------|------------------------|
- Solución iterativa:
$$\theta_{k+1} := \theta_k - \alpha \mathbf{g}(\theta_k)$$
- Se necesita elegir α
 - Funciona bien incluso si D es muy grande
 - Algoritmo más general, válido con otras funciones de coste no cuadráticas

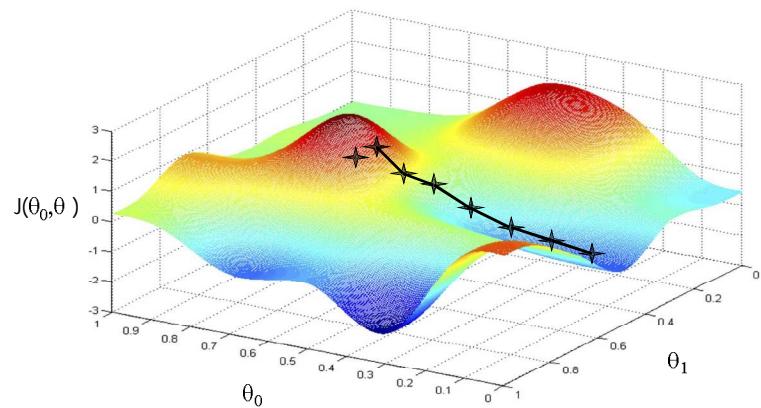
- | | |
|------------------------------|------------------------|
| <u>Descenso de Gradiente</u> | <u>Ecuación Normal</u> |
|------------------------------|------------------------|
- Solución analítica directa:
$$\begin{aligned} \hat{\theta} &= (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} \\ &= \mathbf{X} \setminus \mathbf{y} \end{aligned}$$
- No es necesario elegir α
 - Hay que invertir $\mathbf{X}^T \mathbf{X}$, que tiene dimensión $(D+1) \times (D+1)$
 - Coste computacional $O(D^3)$
 - Si $D \gg 1000$, mejor descenso de gradiente

Descenso de Gradiente: ¿Convergencia?



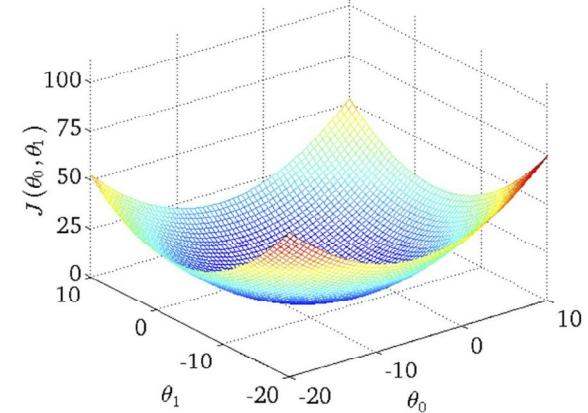
- En el caso general, el descenso de gradiente puede converger a un mínimo local

Descenso de Gradiente: ¿Convergencia?



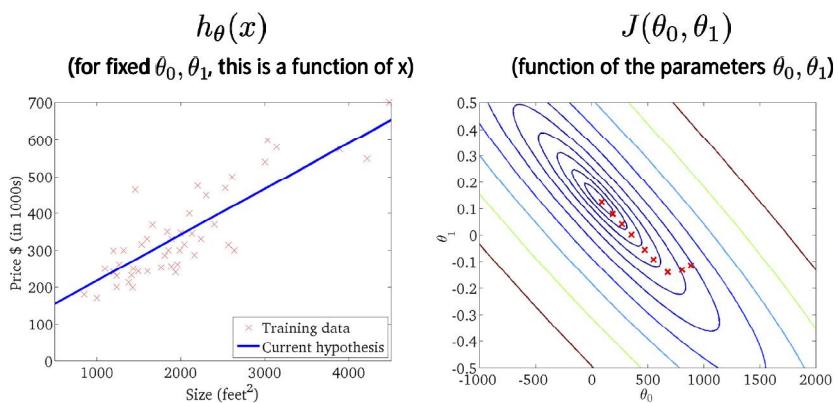
- En el caso general, el descenso de gradiente puede converger a un mínimo local

Convergencia Global (!!)



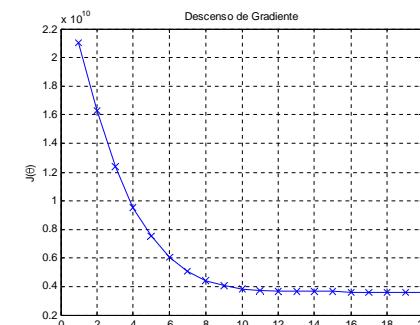
- En mínimos cuadrados la función de coste es convexa
→ Convergencia global

Descenso de Gradiente



¿Como Depurar el Descenso de Gradiente?

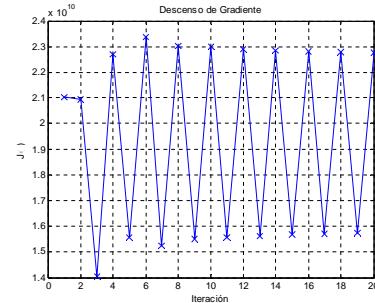
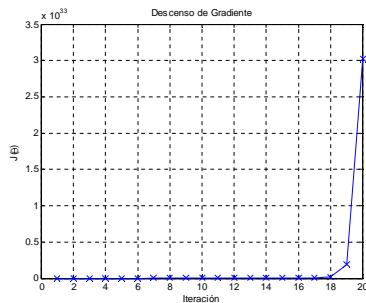
- Dibujar el coste → debería disminuir en cada iteración



♦ Ejemplo de test de terminación de las iteraciones: $|\Delta J| < 10^{-3}$

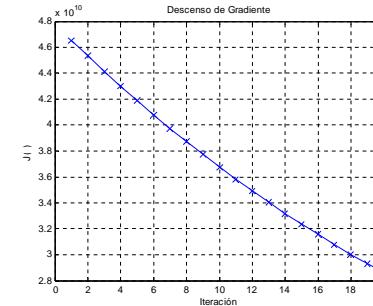
¿Como Depurar el Descenso de Gradiente?

- Si α es demasiado grande: diverge u oscila



¿Como Depurar el Descenso de Gradiente?

- Si α es demasiado pequeño: convergencia lenta

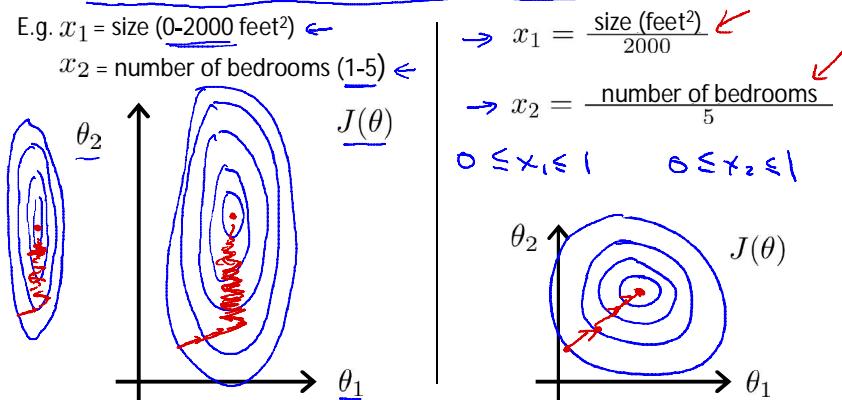


◆ Probar valores α : 0.001 → 0.01 → 0.1 → 1.....
0.003 0.03 0.3 3.....

Escalado de los Atributos

Feature Scaling

Idea: Make sure features are on a similar scale.



Escalar facilita la convergencia del descenso de gradiente

Normalizado de los Atributos

- Sustituir cada atributo por:

$$x_i' = \frac{x_i - \mu_i}{s_i}$$

Atributo escalado
o normalizado

- Donde:

$$\mu_i = \frac{1}{N} \sum_{j=1}^N (x_i^{(j)})$$

Media

$$s_i = \max(x_i^{(j)}) - \min(x_i^{(j)})$$

Rango

ó

$$s_i = std(x_i^{(j)}) = \sqrt{\frac{\sum_{j=1}^N (x_i^{(j)} - \mu_i)^2}{N-1}}$$

Desv. Típica

Des-Normalizado de los Pesos

■ Entrenamos con: $x'_i = \frac{x_i - \mu_i}{s_i}$

Cuidado: no hay que normalizar x_0

■ Y tendremos que:

$$\hat{y} = \theta'^T \mathbf{x}' = \theta'_0 + \sum_{i=1}^D \theta'_i x'_i = \theta'_0 + \sum_{i=1}^D \theta'_i \frac{(x_i - \mu_i)}{s_i} = \theta'_0 - \sum_{i=1}^D \frac{\theta'_i \mu_i}{s_i} + \sum_{i=1}^D \frac{\theta'_i x_i}{s_i}$$

■ Des-normalización:

$$\hat{y} = h_\theta(\mathbf{x}) = \theta^T \mathbf{x}$$

$$\begin{aligned}\theta_0 &= \theta'_0 - \sum_{i=1}^D \frac{\theta'_i \mu_i}{s_i} \\ \theta_i &= \frac{\theta'_i}{s_i}\end{aligned}$$

Regresión con Normalización en Matlab

```
function th = regresion(X, y)
% Ejemplo función de regresión
```

$$X = \begin{pmatrix} 1 & x_1^{(1)} & \dots & x_D^{(1)} \\ 1 & x_1^{(2)} & \dots & x_D^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(N)} & \dots & x_D^{(N)} \end{pmatrix} \quad y = \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{pmatrix} \quad \theta = \begin{pmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_D \end{pmatrix}$$

%Normalizar los atributos

```
N = size(X,1);
mu = mean(X(:,2:end));
sig = std(X(:,2:end));
X(:,2:end) = (X(:,2:end) - repmat(mu,N,1)) ./ repmat(sig,N,1);
```

$$x'_i = \frac{x_i - \mu_i}{s_i}$$

%Resolver la Regresión

%por ejemplo, iterando $\text{th} = \text{th} - \text{alfa} * \text{g}$

%Des-Normalizar los pesos

```
th(2:end) = th(2:end) ./ sig';
th(1) = th(1) - (mu * th(2:end));
```

$$\begin{aligned}\theta_0 &= \theta'_0 - \sum_{i=1}^D \frac{\theta'_i \mu_i}{s_i} \\ \theta_i &= \frac{\theta'_i}{s_i}\end{aligned}$$

Regresión Polinómica (u otras funciones)

■ Hasta ahora hemos hecho:

$$h_\theta(\mathbf{x}) = \theta^T \mathbf{x} = \theta_0 + \theta_1 x_1 + \dots + \theta_D x_D$$

■ Si queremos hacer un ajuste de un polinomio:

$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \dots + \theta_n x^n$$

■ Podemos resolverlo con regresión lineal tomando:

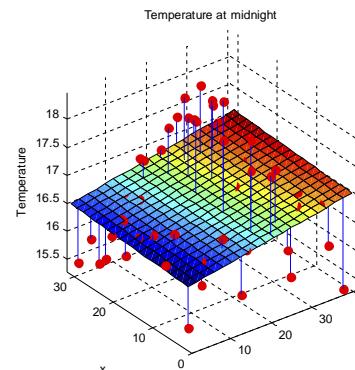
$$\phi(x) = (1, x, x^2, \dots, x^n)^T$$

Expansión de funciones base
(añadir atributos extra: x^2, x^3, \dots)

$$h_\theta(x) = \theta^T \phi(x)$$

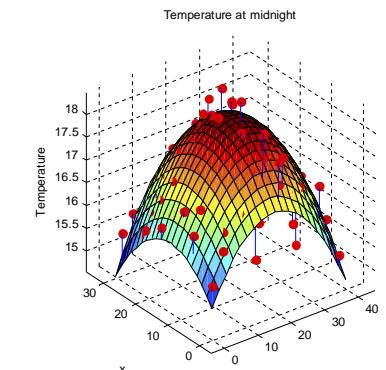
◆ También podrían añadirse $\sin(x)$, \sqrt{x}, \dots

Ejemplo Multivariable



$$h_\theta(\mathbf{x}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

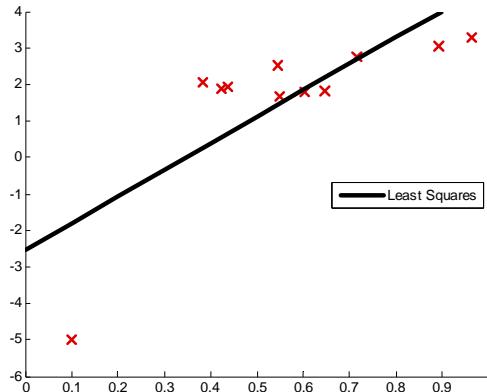
$$\begin{aligned}J(\theta) &= 34.5769 \\ \text{RMSE} &= 0.8077\end{aligned}$$



$$\begin{aligned}h_\theta(\mathbf{x}) &= \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \\ &\quad \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2 \\ J(\theta) &= 9.0763 \\ \text{RMSE} &= 0.4138\end{aligned}$$

Regresión Robusta

- Con coste cuadrático, un error el doble de grande influye cuatro veces más → Los datos esporádicos pueden influir demasiado en la solución (puntos palanca)



Regresión Robusta

- Coste cuadrático:

$$J_{L2}(\theta) = \frac{1}{2} \sum_{i=1}^N (h_\theta(\mathbf{x}^{(i)}) - y^{(i)})^2$$

- Coste valor absoluto:

$$J_{L1}(\theta) = \sum_{i=1}^N |h_\theta(\mathbf{x}^{(i)}) - y^{(i)}|$$

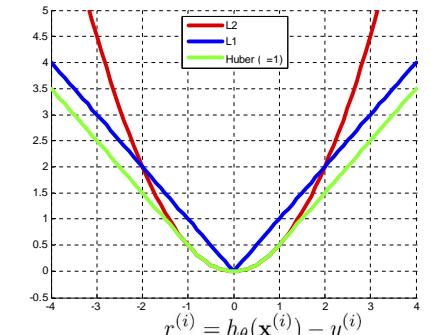
No derivable ☹

- Coste de Huber:

$$L_H(r, \delta) = \begin{cases} r^2/2 & \text{if } |r| \leq \delta \\ \delta|r| - \delta^2/2 & \text{if } |r| > \delta \end{cases}$$

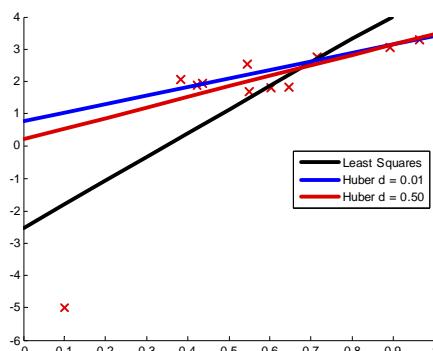
Derivable ☺

$$J_H(\theta) = \sum_{i=1}^N L_H(r^{(i)}, \delta) = \sum_{|r^{(i)}| \leq \delta} r^{(i)2}/2 + \sum_{|r^{(i)}| > \delta} \delta|r^{(i)}| - \delta^2/2$$



Regresión Robusta (Coste de Huber)

- Los datos con residuo > delta influyen poco
 - Escoger delta = residuo razonable de los puntos buenos



Métodos de Optimización de 2º Orden

- El descenso de gradiente sólo utiliza la primera derivada:

$$\theta_{k+1} := \theta_k - \alpha \mathbf{g}(\theta_k)$$

- Aproximación de 2º orden del coste:

$$J(\theta_k + \Delta\theta) \approx J(\theta_k) + \mathbf{g}_k^T \Delta\theta + \frac{1}{2} \Delta\theta^T \mathbf{H}_k \Delta\theta$$

$$\mathbf{g}_k = \mathbf{g}(\theta_k) = \left. \frac{\partial J(\theta)}{\partial \theta} \right|_{\theta=\theta_k} \quad \mathbf{H}_k = \mathbf{H}(\theta_k) = \left. \frac{\partial^2 J(\theta)}{\partial \theta^2} \right|_{\theta=\theta_k}$$

- Mínimo: $\frac{\partial J(\theta_k + \Delta\theta)}{\partial (\Delta\theta)} = \mathbf{g}_k + \mathbf{H}_k \Delta\theta = 0$

$$\theta_{k+1} := \theta_k - \mathbf{H}_k^{-1} \mathbf{g}_k \quad \text{Paso de Newton}$$

Ejemplo: `fminunc` de Matlab y Octave

Coste Cuadrático en Matlab

$$J(\theta) = \frac{1}{2}(X\theta - y)^T(X\theta - y) = \frac{1}{2}r^T r \quad g(\theta) = X^T(X\theta - y)$$

```
function [J,grad,Hess] = CosteL2(theta,X,y)
% Calcula el coste cuadrático, y si se piden,
% su gradiente y su Hessiano
r = X*theta-y;
J = (1/2)*sum(r.^2);
if nargout > 1
    grad = X'*r;
end
if nargout > 2
    Hess = X'*X;
end
```

Nota: si se dividen por N, al cambiar el numero de muestras, no es necesario re-ajustar

$$J(\theta) = \frac{1}{2N}r^T r$$

$$g(\theta) = \frac{1}{N}X^T r$$

$$H(\theta) = \frac{1}{N}X^T X$$

Coste de Huber en Matlab

$$J_H(\theta) = \sum_{i=1}^N L_H(r^{(i)}, \delta) = \sum_{|r^{(i)}| \leq \delta} r^{(i)2}/2 + \sum_{|r^{(i)}| > \delta} \delta |r^{(i)}| - \delta^2/2$$

```
function [J,grad,Hess] = CosteHuber(theta,X,y,d)
r = X*theta-y;
good = abs(r) <= d;
J = (1/2)*sum(r(good).^2) + ...
d*sum(abs(r(~good))) - (1/2)*sum(~good)*d^2;
if nargout > 1
    grad = X(good,:)'*r(good) + ...
    d*X(~good,:)'*sign(r(~good));
end
if nargout > 2
    Hess = X(good,:)'*X(good,:);
end
```

Nota: si se dividen por N, al cambiar el numero de muestras, no es necesario re-ajustar α

Tema 2. Regularización y Selección de Modelos

30231 - Aprendizaje Automático
Grado en Ingeniería Informática
Escuela de Ingeniería y Arquitectura

Índice

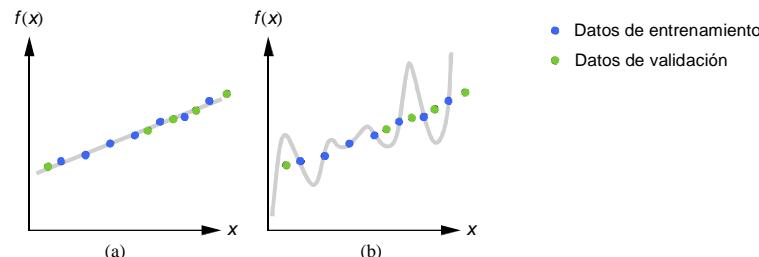
- El problema de sobreajuste
- Selección de modelos: validación cruzada
- Regularización

Créditos de transparencias y figuras:

- Andrew Ng, *Machine Learning*, Stanford AI Lab.
<https://www.coursera.org/course/ml>
- Kevin P. Murphy, *Machine Learning: A Probabilistic Perspective*, MIT Press, 2012

Generalización

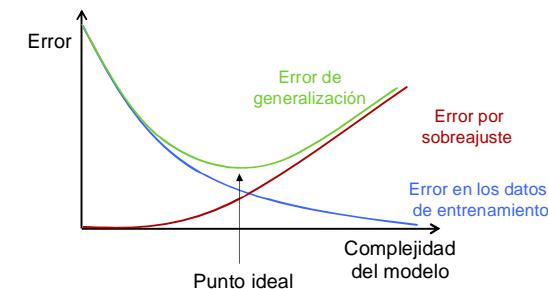
- Capacidad de predecir la salida para nuevos datos



- Es preferible la función menos compleja

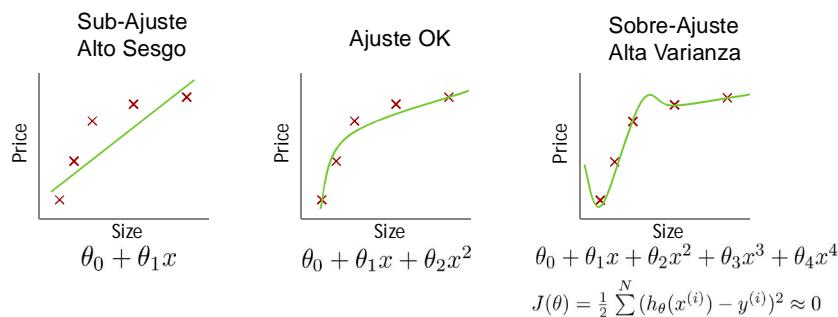
- Principio de la navaja de Occam: En igualdad de condiciones, elegir la hipótesis más simple

Sobreajuste (Overfitting)



Sobreajuste (Overfitting)

- Ejemplo en regresión

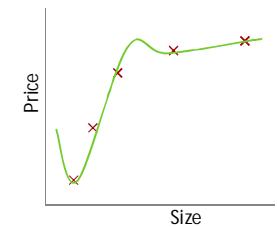


- **Sobreajuste:** Si hay demasiados atributos, la hipótesis puede ajustarse muy bien a los datos de entrenamiento, pero puede no generalizar bien a nuevos ejemplos.

Sobreajuste (Overfitting)

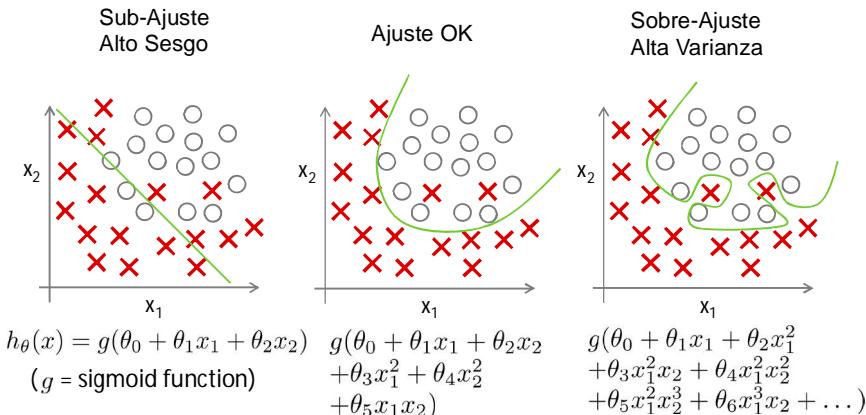
- Ejemplo con demasiados atributos

x_1 = size of house
 x_2 = no. of bedrooms
 x_3 = no. of floors
 x_4 = age of house
 x_5 = average income in neighborhood
 x_6 = kitchen size
 \vdots
 x_{100}



Sobreajuste (Overfitting)

- Ejemplo de clasificación (regresión logística)



Como Evitar el Sobre-Ajuste

Opciones

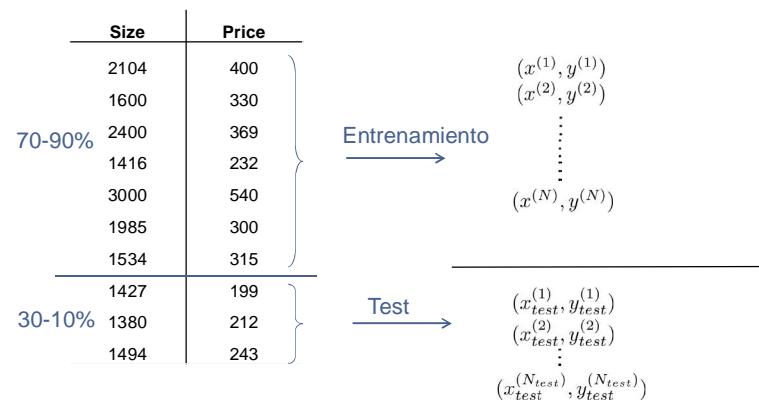
- Reducir el número de atributos
 - Seleccionar manualmente los atributos a mantener
 - Selección de modelos (validación cruzada)
- Regularización
 - Mantener todos los atributos, pero reducir la magnitud de los pesos
 - Funciona bien si hay muchos atributos, y cada uno contribuye un poco a la predicción

Diagnóstico del Aprendizaje

- Diagnóstico: un test que podemos ejecutar para saber qué está funcionando y qué no con el algoritmo de aprendizaje, y obtener pistas de cómo mejorarlo
- Un diagnóstico puede tomar tiempo, pero es tiempo bien invertido
- Siempre utilizar datos de validación / test distintos de los datos de entrenamiento
- Sub-ajuste .vs. Sobre-ajuste

Evaluación de una Hipótesis

- Con datos de test, distintos de los datos de entrenamiento



Varias Hipótesis: Selección de Modelos

- Queremos elegir entre estos modelos

1. $h_\theta(x) = \theta_0 + \theta_1 x$ (d = grado del polinomio)
2. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$
3. $h_\theta(x) = \theta_0 + \theta_1 x + \dots + \theta_3 x^3$
- \vdots
10. $h_\theta(x) = \theta_0 + \theta_1 x + \dots + \theta_{10} x^{10}$

Supongamos que el menor error se da para: $J_{test}(\theta^{(5)})$

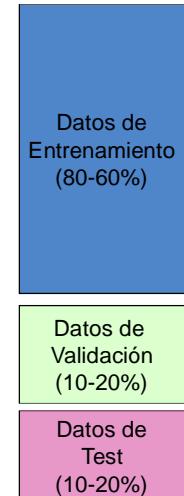
Por tanto, elegimos el polinomio de orden 5: $\theta_0 + \dots \theta_5 x^5$

Problema: $J_{test}(\theta^{(5)})$ es una estimación optimista del error de generalización, porque el parámetro extra d se ha ajustado con los datos de test.

Selección de Modelos: Validación Cruzada

- Datos: casos con salida conocida, por ej. emails marcados como spam/ham. Los dividimos en:

- Datos de entrenamiento
- Datos de validación
- Datos de Test (guardados bajo llave hasta el final)



Proceso de aprendizaje

- Aprender los parámetros con los datos de entrenamiento
- Ajustar los hyper-parámetros (p.e. el tamaño del modelo, el suavizado, la regularización...) con los datos de validación
- SOLO UNA VEZ, AL FINAL: calcular la precisión con los datos de test
- Nunca "espiar" los datos de test

K-fold Cross-Validation

```

Function kfold_cross_validation(Learner, k, examples) returns hypothesis
  best_size ← 0; best_errV ← inf;
  for size = 1 to n do {para los distintos valores de los hyper-parámetros}
    err_T ← 0; err_V ← 0
    for fold = 1 to k do {separar N/k ejemplos para validación}
      [training_set, validation_set] ← Partition(examples, fold, k)
      h ← Learner(size, training_set) {aprender con el resto}
      err_T ← err_T + Error(h, training_set)
      err_V ← err_V + Error(h, validation_set)
    end for
    err_T ← err_T/k; err_V ← err_V/k {calcular el error medio de las k veces}
    if has_converged(err_T) and err_V < best_errV then
      best_size ← size {guardar el mejor valor de los hyper-parámetros}
    end if
  end for
  return Learner(best_size, examples) {aprender de nuevo con todos}
  
```

K-fold Cross-Validation

- Valor típico: K = 10

Leave-one-out Cross Validation:

- Si hay muy pocas muestras de entrenamiento N, tomar K=N
- En cada iteración:
 - ◆ N-1 muestras para el entrenamiento
 - ◆ 1 muestra para la validación

Errores

$$J_{train}(\theta) = \frac{1}{2N} \sum_{i=1}^N (h_\theta(x^{(i)}) - y^{(i)})^2$$

$$J_{cv}(\theta) = \frac{1}{2N_{cv}} \sum_{i=1}^{N_{cv}} (h_\theta(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$$

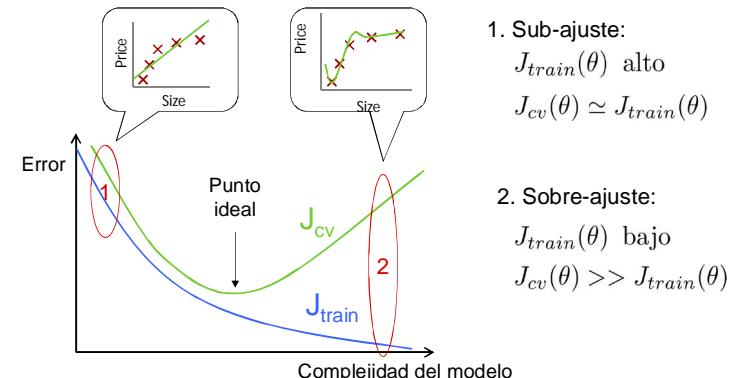
$$J_{test}(\theta) = \frac{1}{2N_{test}} \sum_{i=1}^{N_{test}} (h_\theta(x_{test}^{(i)}) - y_{test}^{(i)})^2$$

Notas:

- Dividido por el número de muestras, para que sean comparables
- Puede quitarse el $\frac{1}{2}$, para que sea directamente el MSE

Sub-Ajuste .vs. Sobre-Ajuste

- Supongamos que el resultado del aprendizaje no tan bueno como pensábamos (J_{cv} o J_{test} son elevados)
- ¿Es un problema de sub-ajuste o de sobre-ajuste?

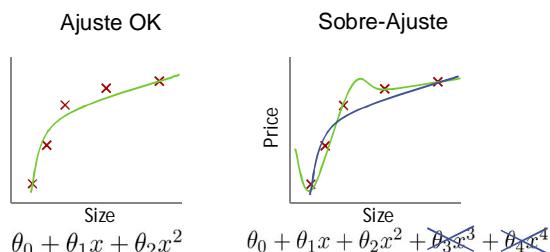


1. Sub-ajuste:
 $J_{train}(\theta)$ alto
 $J_{cv}(\theta) \simeq J_{train}(\theta)$

2. Sobre-ajuste:
 $J_{train}(\theta)$ bajo
 $J_{cv}(\theta) \gg J_{train}(\theta)$

Regularización

- Penalizar la complejidad del modelo



Idea: penalizamos θ_3 y θ_4 para que sean muy pequeños

Regularización

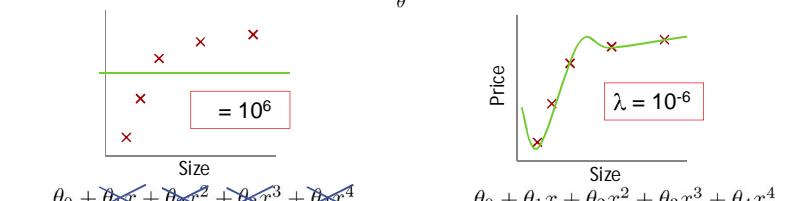
- Valores pequeños para los parámetros $\theta_1, \dots, \theta_D$
- Hipótesis "más simples"
- Menos propenso al sobre-ajuste

θ_0 no se penaliza

- Regresión lineal regularizada

$$J(\theta) = \frac{1}{2} \left[\sum_{i=1}^N (h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^D \theta_j^2 \right]$$

$$\hat{\theta} = \arg \min_{\theta} J(\theta)$$



Solución con Descenso de Gradiente

- Coste $J(\theta) = \frac{1}{2} \left[\sum_{i=1}^N (h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^D \theta_j^2 \right]$

- Gradiente $\mathbf{g}(\theta) = \frac{\partial}{\partial \theta} J(\theta) = \begin{pmatrix} \sum_{i=1}^N (h_\theta(\mathbf{x}^{(i)}) - y^{(i)}) x_0^{(i)} \\ \sum_{i=1}^N (h_\theta(\mathbf{x}^{(i)}) - y^{(i)}) x_1^{(i)} + \lambda \theta_1 \\ \vdots \\ \sum_{i=1}^N (h_\theta(\mathbf{x}^{(i)}) - y^{(i)}) x_D^{(i)} + \lambda \theta_D \end{pmatrix}$

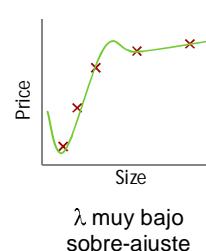
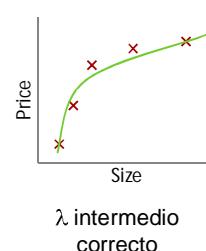
- Descenso de gradiente:

$$\theta_{k+1} := \theta_k - \alpha \mathbf{g}(\theta_k) \quad \alpha: \text{Factor de aprendizaje}$$

Elección del Parámetro de Regularización λ

- Modelo: $h(\theta) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

$$J(\theta) = \frac{1}{2} \left[\sum_{i=1}^N (h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^D \theta_j^2 \right]$$



Solución con la Ecuación Normal

- Gradiente: $\mathbf{g}(\theta) = X^T(X\theta - \mathbf{y}) + \lambda \begin{bmatrix} 0 & 1 & \dots & 1 \end{bmatrix} \theta$
 $= \left(X^T X + \lambda \begin{bmatrix} 0 & 1 & \dots & 1 \end{bmatrix} \right) \theta - X^T \mathbf{y}$

- Ecuación normal:

$$\hat{\theta} = \left(X^T X + \lambda \begin{bmatrix} 0 & 1 & \dots & 1 \end{bmatrix} \right)^{-1} X^T \mathbf{y}$$

- En Matlab:

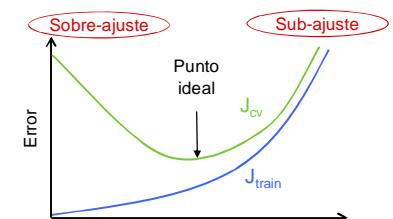
```
H = X'*X + lambda*diag([0 ones(1,D)]);
theta = H \ (X'*y)
```

Elección del Parámetro de Regularización λ

$$J(\theta) = \frac{1}{2} \left[\sum_{i=1}^N (h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^D \theta_j^2 \right]$$

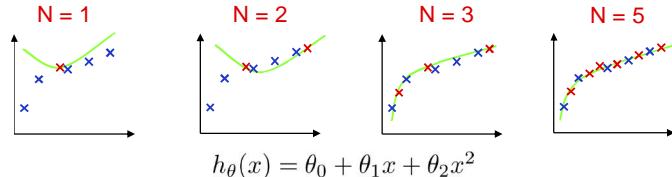
$$J_{train}(\theta) = \frac{1}{2N} \sum_{i=1}^N (h_\theta(x^{(i)}) - y^{(i)})^2$$

$$J_{cv}(\theta) = \frac{1}{2N_{cv}} \sum_{i=1}^{N_{cv}} (h_\theta(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$$

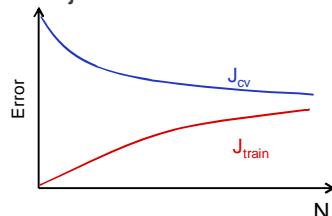


Curvas de Aprendizaje

- Al aumentar las muestras de entrenamiento:



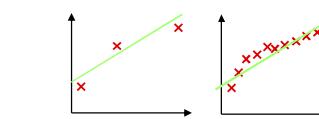
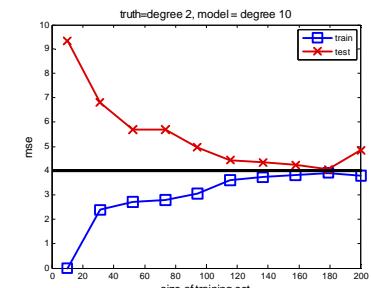
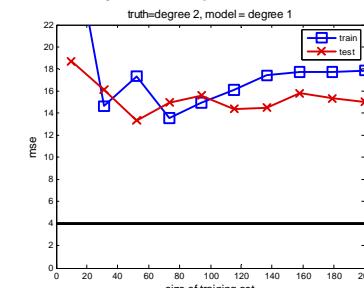
- Curva de aprendizaje:



Efecto Regularizador de Muchos Datos

- Usar muchos datos reduce el sobre-ajuste

- Si hay sub-ajuste, no mejora



Resumen: ¿Cómo Depurar el Aprendizaje?

- Supongamos que hemos resuelto la regresión lineal regularizada para los precios de los pisos

$$J(\theta) = \frac{1}{2} \left[\sum_{i=1}^N (h_{\theta}(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^D \theta_j^2 \right]$$

- Pero la probamos con datos nuevos y comete errores inaceptables. ¿Qué intentamos a continuación?:

- ♦ Más muestras de entrenamiento
- ♦ Usar menos atributos
- ♦ Conseguir atributos nuevos
- ♦ Atributos polinómicos u otros
- ♦ Aumentar λ
- ♦ Disminuir λ

Tema 3. Regresión Logística

Índice

- Problemas de Clasificación
- Regresión Logística
- Regresión Logística Regularizada
- Optimización Avanzada
- Medidas de Error
- Clasificación Multi-Clase

Créditos de transparencias y figuras:

- Andrew Ng, *Machine Learning*, Stanford AI Lab.
<https://www.coursera.org/course/ml>

Aprendizaje Supervisado: Clasificación

- Clasificación: salida discreta

- Binaria: $y \in \{0, 1\}$ $\begin{cases} 0: \text{"Clase Negativa"} \\ 1: \text{"Clase Positiva"} \end{cases}$

- ◆ Email: Spam / Ham
- ◆ Transacción Online Fraudulenta: Si / No
- ◆ Tumor: Maligno / Benigno

- Multi-Clase: $y \in \{1, \dots, C\}$

- ◆ Email: trabajo / amigos / familia / ...
- ◆ Reconocimiento de dígitos: 1, 2, 3, ...
- ◆ Reconocimiento de personas: Pedro, Juan, María, ...

Aprendizaje Supervisado: Clasificación

- ¿Qué pasa si hago una regresión lineal y umbralizo la salida?:

$$\hat{y} = \begin{cases} 0 & \text{if } h_{\theta}(x) < 0.5 \\ 1 & \text{if } h_{\theta}(x) \geq 0.5 \end{cases}$$



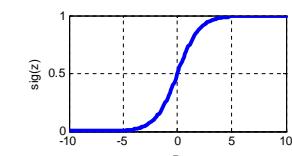
- Regresión Logística: $0 \leq h_{\theta}(\mathbf{x}) \leq 1$

- ◆ Es una técnica de Clasificación

Regresión Logística

- Función sigmoidal (función logística):

$$\text{sig}(z) = \frac{1}{1 + e^{-z}}$$



- ◆ Derivada: $\frac{\partial \text{sig}(z)}{\partial z} = \text{sig}(z)(1 - \text{sig}(z))$

- Tomamos:
$$h_{\theta}(\mathbf{x}) = \frac{1}{1 + e^{-\theta^T \mathbf{x}}} \quad 0 \leq h_{\theta}(\mathbf{x}) \leq 1$$

- Interpretación probabilista: $h_{\theta}(\mathbf{x}) = p(y = 1 | \mathbf{x}; \theta)$

Regresión Logística

- Interpretación: $h_\theta(\mathbf{x}) = p(y=1|\mathbf{x}; \theta)$

♦ Probabilidad de $y=1$ dado \mathbf{x} , parametrizado por θ



♦ Ejemplo:

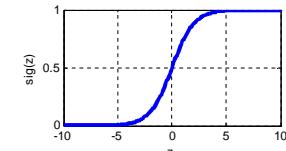
$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \text{Size} \end{bmatrix} \quad h_\theta(\mathbf{x}) = \frac{1}{1 + e^{-\theta^T \mathbf{x}}} = 0.7$$

→ La probabilidad de que el tumor sea maligno es 0.7

Frontera de Decisión

- Supongamos que predecimos:

$$\hat{y} = \begin{cases} 0 & \text{if } h_\theta(\mathbf{x}) < 0.5 \\ 1 & \text{if } h_\theta(\mathbf{x}) \geq 0.5 \end{cases} \quad h_\theta(\mathbf{x}) = \text{sig}(\theta^T \mathbf{x}) = \frac{1}{1 + e^{-\theta^T \mathbf{x}}}$$

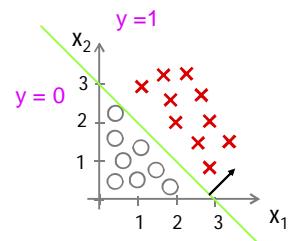


- Equivale a:

$$\hat{y} = \begin{cases} 0 & \text{if } \theta^T \mathbf{x} < 0 \\ 1 & \text{if } \theta^T \mathbf{x} \geq 0 \end{cases}$$

La frontera de decisión es lineal en el espacio de los atributos \mathbf{x}

Frontera de Decisión



$$h_\theta(\mathbf{x}) = \text{sig}(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

$$\hat{y} = \begin{cases} 0 & \text{if } h_\theta(\mathbf{x}) < 0.5 \\ 1 & \text{if } h_\theta(\mathbf{x}) \geq 0.5 \end{cases}$$

$$\theta = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$$

$$\hat{y} = 1 \quad \text{if } \theta^T \mathbf{x} = -3 + x_1 + x_2 \geq 0 \\ x_1 + x_2 \geq 3$$

Fronteras de Decisión No Lineales

- Atributos originales:

$$\mathbf{x} = (1, x_1, x_2, \dots)^T$$

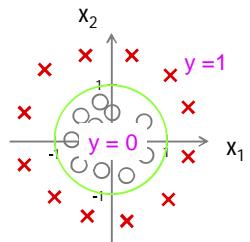
- Expansión de funciones base:

$$\phi(\mathbf{x}) = (1, x_1, x_2, x_1^2, x_1 x_2, x_2^2, x_1^3, x_1^2 x_2, x_1 x_2^2, x_2^3, \dots)^T$$

$$h_\theta(\mathbf{x}) = \text{sig}(\theta^T \phi(\mathbf{x})) = \frac{1}{1 + e^{-\theta^T \phi(\mathbf{x})}}$$

Fronteras de Decisión No Lineales

- Ejemplo: $h_\theta(\mathbf{x}) = \text{sig}(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$



$$\theta = (-1, 0, 0, 1, 1)^T$$

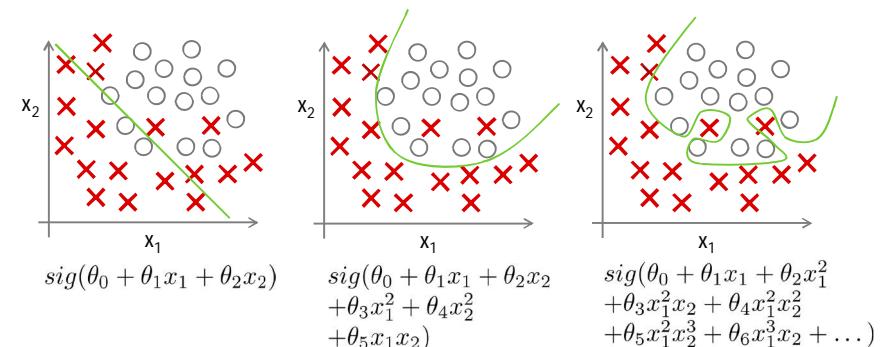
$$\hat{y} = 1 \quad \text{if} \quad \theta^T \mathbf{x} = -1 + x_1^2 + x_2^2 \geq 0$$

$$x_1^2 + x_2^2 \geq 1$$

Nota: La frontera de decisión es lineal en el espacio de los atributos expandidos $\phi(\mathbf{x})$
Pero no lo es en el espacio original \mathbf{x}

Fronteras de Decisión No Lineales

- Pueden conseguirse fronteras arbitrariamente complejas
 - ◆ Pero: ¡cuidado con el sobre-ajuste!



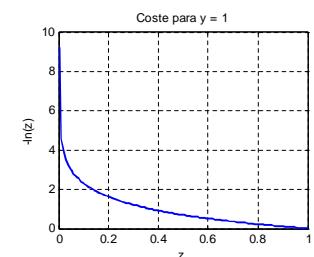
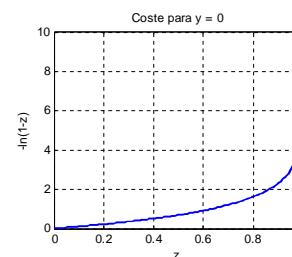
Aprendizaje en Regresión Logística

- Muestras de entrenamiento $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^N$
- Variables de entrada o atributos (*input / features*) $x_1, \dots, x_D \quad x_0 = 1$ $\mathbf{x} = (x_0, x_1, \dots, x_D)^T$
- Variable de salida u objetivo (*output / target*) $y \in \{0, 1\}$
- Parámetros o pesos (*weights*) $\theta = (\theta_0, \theta_1, \dots, \theta_D)^T$
- Hipótesis $h_\theta(\mathbf{x}) = \frac{1}{1 + e^{-\theta^T \mathbf{x}}}$

¿Cómo aprender los valores de los parámetros ?

Función de Coste

$$J^{(i)}(\theta) = \begin{cases} -\ln(h_\theta(\mathbf{x}^{(i)})) & \text{if } y^{(i)} = 1 \\ -\ln(1 - h_\theta(\mathbf{x}^{(i)})) & \text{if } y^{(i)} = 0 \end{cases}$$



Si $h_\theta(x) = 0 \quad J = 0$
Si $h_\theta(x) \rightarrow 1 \quad J \rightarrow \infty$

Si $h_\theta(x) = 1 \quad J = 0$
Si $h_\theta(x) \rightarrow 0 \quad J \rightarrow \infty$

Función de Coste

$$J^{(i)}(\theta) = \begin{cases} -\ln(h_{\theta}(\mathbf{x}^{(i)})) & \text{if } y^{(i)} = 1 \\ -\ln(1 - h_{\theta}(\mathbf{x}^{(i)})) & \text{if } y^{(i)} = 0 \end{cases}$$

$$J^{(i)}(\theta) = -y^{(i)} \ln(h_{\theta}(\mathbf{x}^{(i)})) - (1 - y^{(i)}) \ln(1 - h_{\theta}(\mathbf{x}^{(i)}))$$

$$J(\theta) = -\frac{1}{N} \sum_{i=1}^N y^{(i)} \ln(h_{\theta}(\mathbf{x}^{(i)})) + (1 - y^{(i)}) \ln(1 - h_{\theta}(\mathbf{x}^{(i)}))$$

- Para aprender los parámetros:

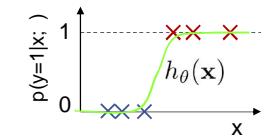
$$\hat{\theta} = \arg \min_{\theta} J(\theta)$$

Coste: Interpretación Probabilista

- Queremos aprender un modelo probabilista:

$$p(y=1|\mathbf{x}; \theta) = h_{\theta}(\mathbf{x})$$

$$p(y=0|\mathbf{x}; \theta) = 1 - h_{\theta}(\mathbf{x})$$



- Por lo tanto: $p(y|\mathbf{x}; \theta) = h_{\theta}(\mathbf{x})^y (1 - h_{\theta}(\mathbf{x}))^{(1-y)}$

- Si las muestras de entrenamiento son independientes, la verosimilitud de los parámetros θ viene dada por:

$$\begin{aligned} L(\theta) &= p(\mathbf{y}|\mathbf{X}; \theta) = \prod_{i=1}^N p(y^{(i)}|\mathbf{x}^{(i)}; \theta) \\ &= \prod_{i=1}^N h_{\theta}(\mathbf{x}^{(i)})^{y^{(i)}} (1 - h_{\theta}(\mathbf{x}^{(i)}))^{(1-y^{(i)})} \end{aligned}$$

Coste: Interpretación Probabilista

- Tenemos: $L(\theta) = \prod_{i=1}^N h_{\theta}(\mathbf{x}^{(i)})^{y^{(i)}} (1 - h_{\theta}(\mathbf{x}^{(i)}))^{(1-y^{(i)})}$
- $\ln(L(\theta)) = \sum_{i=1}^N y^{(i)} \ln(h_{\theta}(\mathbf{x}^{(i)})) + (1 - y^{(i)}) \ln(1 - h_{\theta}(\mathbf{x}^{(i)}))$

- La solución de máxima verosimilitud es:

$$\hat{\theta} = \arg \max_{\theta} L(\theta) = \arg \max_{\theta} \ln(L(\theta)) = \arg \min_{\theta} J(\theta)$$

- Con: $J(\theta) = -\frac{1}{N} \sum_{i=1}^N y^{(i)} \ln(h_{\theta}(\mathbf{x}^{(i)})) + (1 - y^{(i)}) \ln(1 - h_{\theta}(\mathbf{x}^{(i)}))$

- Minimizar el coste J equivale a maximizar la verosimilitud de los parámetros θ
- Máxima verosimilitud → valor de θ que mejor explica los datos de entrenamiento que tenemos

Algoritmo de Descenso de Gradiente

- Coste $J(\theta) = -\frac{1}{N} \sum_{i=1}^N y^{(i)} \ln(h_{\theta}(\mathbf{x}^{(i)})) + (1 - y^{(i)}) \ln(1 - h_{\theta}(\mathbf{x}^{(i)}))$
- $h_{\theta}(\mathbf{x}^{(i)}) = \text{sig}(\theta^T \mathbf{x}^{(i)}) = \frac{1}{1 + e^{-\theta^T \mathbf{x}^{(i)}}}$

- Gradiente

$$\mathbf{g}(\theta) = \frac{\partial}{\partial \theta} J(\theta) = \begin{pmatrix} \frac{\partial J(\theta)}{\partial \theta_0} \\ \vdots \\ \frac{\partial J(\theta)}{\partial \theta_D} \end{pmatrix} = \begin{pmatrix} \frac{1}{N} \sum_{i=1}^N (h_{\theta}(\mathbf{x}^{(i)}) - y^{(i)}) x_0^{(i)} \\ \vdots \\ \frac{1}{N} \sum_{i=1}^N (h_{\theta}(\mathbf{x}^{(i)}) - y^{(i)}) x_D^{(i)} \end{pmatrix}$$

- Descenso de gradiente (Batch):

Repetir

$$\boldsymbol{\theta}_{k+1} := \boldsymbol{\theta}_k - \alpha \mathbf{g}(\boldsymbol{\theta}_k)$$

Hasta que converja

α : Factor de aprendizaje

Cálculo Vectorizado (Matlab)

Matriz de diseño:

$$X = \begin{pmatrix} 1 & x_1^{(1)} & \dots & x_D^{(1)} \\ 1 & x_1^{(2)} & \dots & x_D^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(N)} & \dots & x_D^{(N)} \end{pmatrix} \quad \theta = \begin{pmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_D \end{pmatrix} \quad \mathbf{y} = \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{pmatrix}$$

Salidas predichas:

$$\mathbf{h} = \begin{pmatrix} h_\theta(\mathbf{x}^{(1)}) \\ \vdots \\ h_\theta(\mathbf{x}^{(N)}) \end{pmatrix} = \frac{1}{1 + e^{-X\theta}}$$

Coste:

$$J(\theta) = -\frac{1}{N} (\mathbf{y}^T \ln(\mathbf{h}) + (1 - \mathbf{y}^T) \ln(1 - \mathbf{h}))$$

Gradiente:

$$\mathbf{g}(\theta) = \frac{1}{N} X^T (\mathbf{h} - \mathbf{y})$$

Cálculo Vectorizado (Matlab)

Hessiano:

$$H(\theta) = \frac{1}{N} X^T R X$$

donde:

$$\mathbf{h} = \begin{pmatrix} h_1 \\ \vdots \\ h_N \end{pmatrix} = \begin{pmatrix} h_\theta(\mathbf{x}^{(1)}) \\ \vdots \\ h_\theta(\mathbf{x}^{(N)}) \end{pmatrix} = \frac{1}{1 + e^{-X\theta}}$$

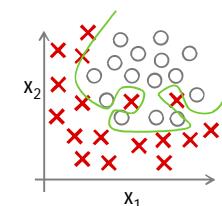
$$R = \begin{pmatrix} h_1(1 - h_1) & 0 & \dots & 0 \\ 0 & h_2(1 - h_2) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & h_N(1 - h_N) \end{pmatrix}$$

- Nota: El Hessiano H es definido positivo, luego la función de coste es convexa → La función de coste tiene un único mínimo

Coste Logístico en Matlab

```
function [J,grad,Hess] = CosteLogistico(theta,X,y)
% Calcula el coste logístico, y si se piden,
% su gradiente y su Hessiano
N = size(X,2);
h = 1./(1+exp(-(X*theta)));
J = (-y'*log(h) - (1-y')*log(1-h))/N;
if nargout > 1
    grad = X'*(h-y)/N;
end
if nargout > 2
    R = diag(h.*(1-h));
    Hess = X'*R*X/N;
end
```

Regresión Logística Regularizada



$$\text{sig}(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 + \theta_6 x_1^3 x_2 + \dots)$$

$$J(\theta) = -\frac{1}{N} \sum_{i=1}^N y^{(i)} \ln(h_\theta(\mathbf{x}^{(i)})) + (1 - y^{(i)}) \ln(1 - h_\theta(\mathbf{x}^{(i)})) + \frac{\lambda}{2} \sum_{j=1}^D \theta_j^2$$

$$\mathbf{g}(\theta) = \frac{1}{N} X^T (\mathbf{h} - \mathbf{y}) + \lambda \begin{bmatrix} 0 & 1 & & \\ & \ddots & & \\ & & \ddots & 1 \end{bmatrix} \theta$$

$$H(\theta) = \frac{1}{N} X^T R X + \lambda \begin{bmatrix} 0 & 1 & & \\ & \ddots & & \\ & & \ddots & 1 \end{bmatrix}$$

Coste Logístico Regularizado en Matlab

```
function [J,grad,Hess] = CosteLogReg(theta,X,y,lambda)
% Calcula el coste logistico regularizado,
% y si se piden, su gradiente y su Hessiano
[N, D] = size(X);
h = 1./(1+exp(-(X*theta)));
th = theta; th(1)= 0; %Para no penalizar theta_0
J =(-y'*log(h) - (1-y')*log(1-h))/N +(lambda/2)*(th'*th);
if nargout > 1
    grad = X'*(h-y)/N + lambda*th;
end
if nargout > 2
    R = diag(h.*(1-h));
    Hess = X'*R*X/N + diag([0 lambda*ones(1,D-1)]);
end
```

Optimización Avanzada

■ Con control del paso

- Calcular la dirección de avance \mathbf{d}_k resolviendo:

$$\mathbf{H}_k \mathbf{d}_k = -\mathbf{g}_k$$

- Buscar el paso η_k que consiga el mayor descenso posible de J:

$$\theta_{k+1} := \theta_k + \eta_k \mathbf{d}_k \quad J(\theta_{k+1}) << J(\theta_k)$$

■ Algoritmo de Levenger-Marquardt

- $(\mathbf{H}_k + \lambda_k \mathbf{I}) \mathbf{d}_k = -\mathbf{g}_k$

$\lambda_k \rightarrow 0$ Dirección de Newton

$\lambda_k \rightarrow \infty$ Descenso de Gradiente

- $\theta_{k+1} := \theta_k + \eta_k \mathbf{d}_k$

Métodos de Optimización de 2º Orden

- El descenso de gradiente sólo utiliza la primera derivada:

$$\theta_{k+1} := \theta_k - \alpha \mathbf{g}(\theta_k)$$

- Aproximación de 2º orden del coste:

$$J(\theta_k + \Delta\theta) \approx J(\theta_k) + \mathbf{g}_k^T \Delta\theta + \frac{1}{2} \Delta\theta^T \mathbf{H}_k \Delta\theta$$

$$\mathbf{g}_k = \mathbf{g}(\theta_k) = \left. \frac{\partial J(\theta)}{\partial \theta} \right|_{\theta=\theta_k} \quad \mathbf{H}_k = \mathbf{H}(\theta_k) = \left. \frac{\partial^2 J(\theta)}{\partial \theta^2} \right|_{\theta=\theta_k}$$

- Mínimo: $\frac{\partial J(\theta_k + \Delta\theta)}{\partial (\Delta\theta)} = \mathbf{g}_k + \mathbf{H}_k \Delta\theta = 0$

$$\theta_{k+1} := \theta_k - \mathbf{H}_k^{-1} \mathbf{g}_k \quad \text{Paso de Newton}$$

Optimización Avanzada

- Algoritmos disponibles en librerías de optimización

- | | |
|-------------------------|------------------------------------------|
| ■ Descenso de gradiente | Puede ser lento |
| ■ Gradiente conjugado | Para problemas muy grandes |
| ■ Newton | Necesita Hessiano analítico |
| ■ Quasi-Newton | Aproxima el Hessiano |
| ■ BFGS | Aproximación de bajo rango del Hessiano |
| ■ L-BFGS | Con memoria limitada (para D muy grande) |
| ■ | |

- Ventajas

- No necesitamos elegir manualmente α
- Más rápidos que descenso de gradiente

Optimización Avanzada en Matlab/Octave

- Optimization toolbox de Matlab / Octave

$$x = \text{fminunc}(\text{fun}, x_0, \text{options}) \quad \hat{x} = \arg \min_{\mathbf{x}} f(\mathbf{x})$$

- Ejemplo de uso:

$$\hat{\theta} = \arg \min_{\theta} J(\theta)$$

```
function [J,grad,Hess] = CosteLogReg(theta,X,y,lambda)
.....
options = optimset('GradObj','on' ,.... % usar gradiente
                  'Hessian','on' ,....); % usar Hessiano
theta = fminunc(@(t)(CosteLogReg(t,X,y,lambda)),...
                 initial_theta, options);
```

- Más información:

`help fminunc`
`doc fminunc`

Optimización Avanzada en Matlab/Octave

- Paquete minFunc Mark Schmidt (2005-2012)

$$x = \text{minFunc}(f,x_0,\text{options},\text{varargin}) \quad \hat{x} = \arg \min_{\mathbf{x}} f(\mathbf{x})$$

- Ejemplo de uso:

$$\hat{\theta} = \arg \min_{\theta} J(\theta)$$

```
function [J,grad,Hess] = CosteLogReg(theta,X,y,lambda)
.....
options.display = 'final'; % otros: 'iter' , 'none'
options.method = 'newton'; % por defecto: 'lbfgs'
theta = minFunc(@CosteLogReg, theta_ini, options, ...
                 Xtr, ytr, lambda);
```

- Más información:

<http://www.di.ens.fr/~mschmidt/Software/minFunc.html>

Nota: Versión compatible con Matlab 6.5 en: add.unizar.es

Medidas de Error

- Predicción de la salida:

$$h_{\theta}(\mathbf{x}) = \frac{1}{1 + e^{-\theta^T \mathbf{x}}} \quad \hat{y} = \begin{cases} 0 & \text{if } h_{\theta}(\mathbf{x}) < 0.5 \\ 1 & \text{if } h_{\theta}(\mathbf{x}) \geq 0.5 \end{cases}$$

- Tasa de acierto:

$$A_{cv}(\theta) = \frac{1}{N_{cv}} \sum_{i=1}^{N_{cv}} [\hat{y}_{cv}^{(i)} = y_{cv}^{(i)}]$$

- Tasa de error:

$$E_{cv}(\theta) = \frac{1}{N_{cv}} \sum_{i=1}^{N_{cv}} [\hat{y}_{cv}^{(i)} \neq y_{cv}^{(i)}]$$

$$[P] = \begin{cases} 1 & \text{if } P = \text{True} \\ 0 & \text{otherwise} \end{cases}$$

Problema: Clases Sesgadas

- Supongamos que hemos entrenado un clasificador para predecir:

$$\begin{aligned} y &= 1 \text{ cáncer} \\ y &= 0 \text{ no cáncer} \end{aligned}$$

- Lo evaluamos y obtenemos: $E_{cv} = 1\%$
- ¿Es un clasificador bueno?

- Si sólo el 0.5% de los pacientes tiene cáncer, este otro "clasificador" tiene mejor tasa de aciertos:

```
function y = predictCancer(x)
    y = 0
    return

```

$$E_{cv} = 0.5\%$$

Precision / Recall (Precisión / Recobrado)

- Convención: $y=1$ es la clase rara que queremos detectar

		Clase Real	
		1	0
Clase Predicha	1	True Positive TP	False Positive FP
	0	False Negative FN	True Negative TN

Matriz de confusión

$$\text{Precision} = \frac{TP}{TP+FP}$$

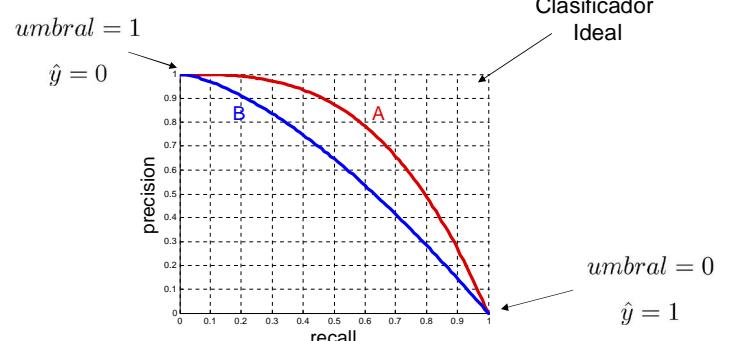
$$\text{Recall} = \frac{TP}{TP+FN}$$

- Precision: de los que damos como positivos, qué fracción lo son realmente
- Recall: de los casos positivos, qué fracción detectamos

Compromiso entre Precision y Recall

$$\hat{y} = \begin{cases} 0 & \text{if } h_{\theta}(\mathbf{x}) < \text{umbral} \\ 1 & \text{if } h_{\theta}(\mathbf{x}) \geq \text{umbral} \end{cases}$$

- Cambiando el umbral podemos mejorar la precisión a costa del recall, o viceversa



F₁ score

$$F_1 = 2 \frac{P \cdot R}{P+R}$$

- Permite comparar y elegir el mejor (validación cruzada)

	Precision	Recall	F1
Clasificador 1	0.5	0.4	0.444
Clasificador 2	0.7	0.2	0.311
Clasificador 3	0.02	1	0.039

Clasificación Multi-Clase

- Ejemplos Multi-Clase: $y \in \{1, \dots, C\}$

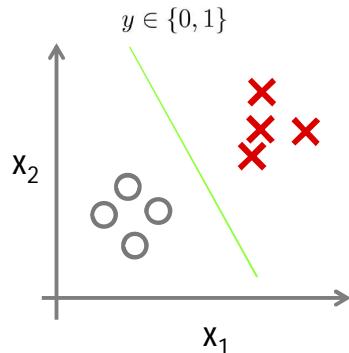
- ◆ Email: Spam, trabajo, amigos, familia, ...
- ◆ Diagnóstico médico: sano, catarro, gripe, ...
- ◆ Metereología: sol, nuboso, lluvia, nieve
- ◆ Reconocimiento de dígitos: 1, 2, 3, ...
- ◆ Reconocimiento de personas: Pedro, Juan, María, ...

- Vamos a aprender j modelos discriminativos:

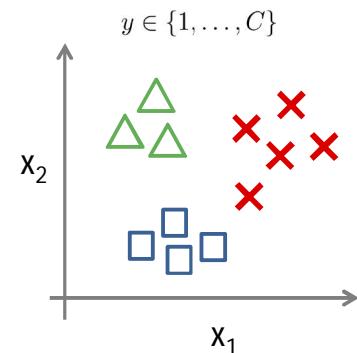
$$p(y = j | \mathbf{x}; \theta) = h_{\theta}^{(j)}(\mathbf{x}) \quad (j = 1, 2, \dots, C)$$

¿Como Aplicamos Regresión Logística?

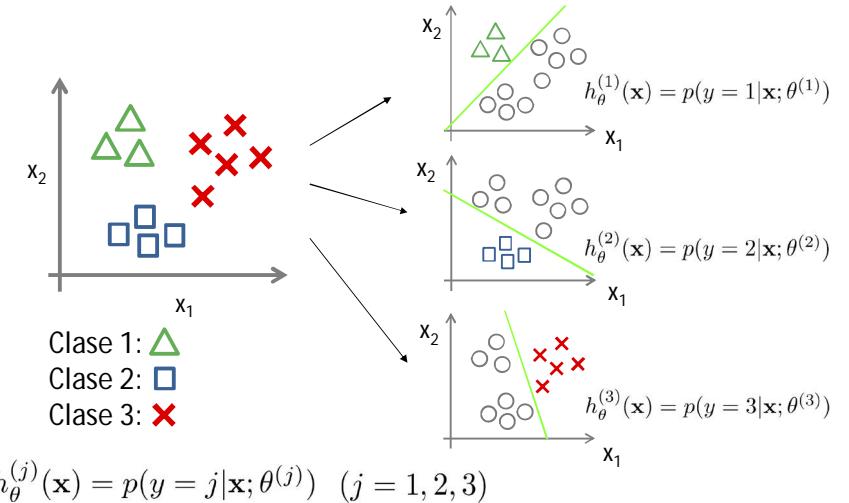
Clasificación Binaria:



Clasificación Multi clase:



One-vs-all (One-vs-rest)



One-vs-all (One-vs-rest)

- Entrenar un clasificador por regresión logística para cada clase j , que estime la probabilidad de $y = j$

$$h_{\theta}^{(j)}(\mathbf{x}) = p(y = j | \mathbf{x}; \theta^{(j)})$$

- Dada una nueva entrada x , dar como predicción la clase j más probable:

$$\hat{y} = \arg \max_j h_{\theta}^{(j)}(\mathbf{x})$$

Cálculo Vectorizado (Matlab)

Matriz de diseño:

$$X = \begin{pmatrix} 1 & x_1^{(1)} & \dots & x_D^{(1)} \\ 1 & x_1^{(2)} & \dots & x_D^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(N)} & \dots & x_D^{(N)} \end{pmatrix} \quad \mathbf{y} = \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{pmatrix} \quad y^{(i)} \in \{1, \dots, C\}$$

Para cada clase j entrenamos un clasificador one-vs-all con:

$$\mathbf{y}_j = \begin{pmatrix} [y^{(1)} = j] \\ [y^{(2)} = j] \\ \vdots \\ [y^{(N)} = j] \end{pmatrix} \quad \rightarrow \quad \theta^{(j)} = \begin{pmatrix} \theta_0^{(j)} \\ \theta_1^{(j)} \\ \vdots \\ \theta_D^{(j)} \end{pmatrix}$$

$$y_j^{(i)} \in \{0, 1\}$$

Cálculo Vectorizado (Matlab)

Matriz de diseño:

$$X = \begin{pmatrix} 1 & x_1^{(1)} & \dots & x_D^{(1)} \\ 1 & x_1^{(2)} & \dots & x_D^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(N)} & \dots & x_D^{(N)} \end{pmatrix}$$

C clasificadores one-vs-all:

$$\Theta = \begin{pmatrix} \theta_0^{(1)} & \dots & \theta_0^{(C)} \\ \theta_1^{(1)} & \dots & \theta_1^{(C)} \\ \vdots & & \vdots \\ \theta_D^{(1)} & \dots & \theta_D^{(C)} \end{pmatrix}$$

Salidas predichas:

$$\mathbf{h} = \begin{pmatrix} h_\theta^{(1)}(\mathbf{x}^{(1)}) & \dots & h_\theta^{(C)}(\mathbf{x}^{(1)}) \\ \vdots & & \vdots \\ h_\theta^{(1)}(\mathbf{x}^{(N)}) & \dots & h_\theta^{(C)}(\mathbf{x}^{(N)}) \end{pmatrix} = \frac{1}{1 + e^{-\mathbf{x}\Theta}}$$

$$\hat{y}^{(i)} = \arg \max_j h_\theta^{(j)}(\mathbf{x}^{(i)})$$

Máximos de cada fila

Matriz de Confusión

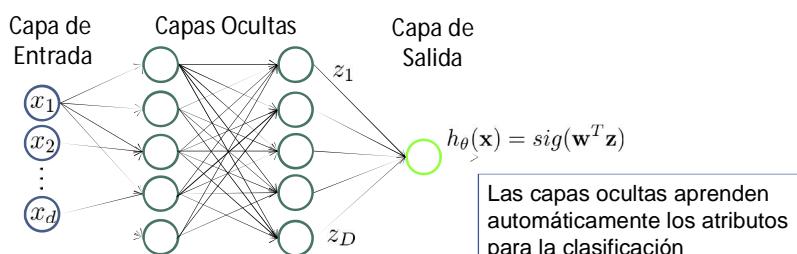
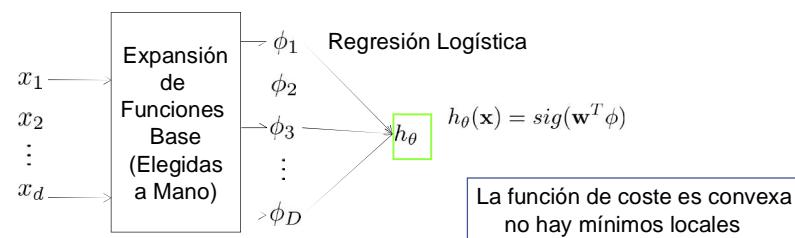
	Clase Predicha				
	Peatón	Coche	Moto	Camión	Suma
Peatón	120	0	3	0	123
Coche	0	301	15	20	336
Moto	1	15	50	5	71
Camión	0	12	3	70	85
Suma	121	328	71	95	615

■ Para peatones:

$$Precision = \frac{120}{121}$$

$$Recall = \frac{120}{123}$$

Regresión Logística .vs. Redes Neuronales



Tema 4. Modelos Generativos. Bayes Ingenuo

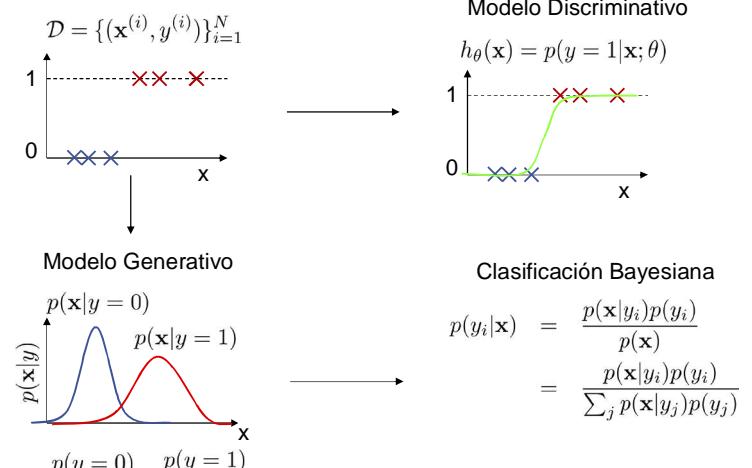
Índice

- Modelos Generativos .vs. Discriminativos
- Clasificación Bayesiana
- La Distribución Gaussiana
- Clasificación con modelos Gaussianos
- Bayes Ingenuo

Créditos de transparencias y figuras:

- Andrew Ng, *Machine Learning*, Stanford AI Lab. <https://www.coursera.org/course/ml>
- C.M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006
- R.O. Duda, P.E. Hart, D.G. Stork, *Pattern Classification*, Wiley, 2001
- K.P. Murphy, *Machine Learning: a Probabilistic Perspective*, MIT press 2012
- J. Neira, *Visión por Computador*, Universidad de Zaragoza

Modelos Generativos .vs. Discriminativos



$$\begin{aligned}\text{Clasificación Bayesiana} \\ p(y_i|\mathbf{x}) &= \frac{p(\mathbf{x}|y_i)p(y_i)}{p(\mathbf{x})} \\ &= \frac{p(\mathbf{x}|y_i)p(y_i)}{\sum_j p(\mathbf{x}|y_j)p(y_j)}\end{aligned}$$

Modelos Generativos .vs. Discriminativos

- Modelos Generativos
 - Aprenden la distribución de los atributos para cada clase $p(\mathbf{x}|y_i)$
 - Y las probabilidades a priori de cada clase $p(y_i)$
 - Generativo: permite generar muestras sintéticas $(\mathbf{x}^{(i)}, y^{(i)})$
 - Suelen hacer suposiciones adicionales, p.ej.:
 - ◆ Gaussianidad de los atributos
 - ◆ Independencia condicional de los atributos
 - Si las suposiciones son acertadas suelen aprender más deprisa que los modelos discriminativos (con menos muestras)
- Modelos Discriminativos (regresión logística, R.N.,...)
- Menos suposiciones → más robusto

Clasificación Bayesiana

- Notaciones equivalentes $y \in \{1, \dots, C\}$
 - Prob. a priori: $p(y_i) \equiv p(y = i) \equiv p(\omega_i)$
 - Prob. a posteriori: $p(y_i|\mathbf{x}) \equiv p(y = i|\mathbf{x}) \equiv p(\omega_i|\mathbf{x})$
 - Verosimilitud: $p(\mathbf{x}|y_i) \equiv p(\mathbf{x}|y = i) \equiv p(\mathbf{x}|\omega_i)$

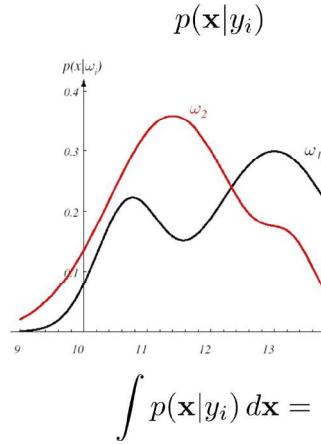
■ Clasificación Bayesiana:

$$p(y_i|\mathbf{x}) = \frac{p(\mathbf{x}|y_i)p(y_i)}{p(\mathbf{x})} = \frac{p(\mathbf{x}|y_i)p(y_i)}{\sum_{j=1}^C p(\mathbf{x}|y_j)p(y_j)}$$

- Regla de decisión:

$$\hat{y} = \arg \max_i p(y_i|\mathbf{x}) = \arg \max_i p(\mathbf{x}|y_i)p(y_i)$$

Clasificación Bayesiana



$$\int p(\mathbf{x}|y_i) d\mathbf{x} = 1$$

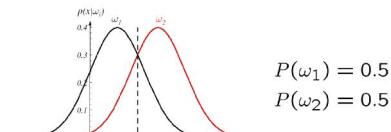
$$p(y_i|\mathbf{x}) = \frac{p(\mathbf{x}|y_i)p(y_i)}{\sum_{j=1}^C p(\mathbf{x}|y_j)p(y_j)}$$

Graph showing the posterior probability $P(\omega_i|x)$ versus x for two classes ω_1 (black) and ω_2 (red). The x-axis ranges from 9 to 15. The curves are sigmoidal, starting near 0 for ω_1 and approaching 1 for ω_2 . The regions \mathcal{R}_1 and \mathcal{R}_2 are indicated by vertical dashed lines at $x \approx 11.5$ and $x \approx 13.5$ respectively.

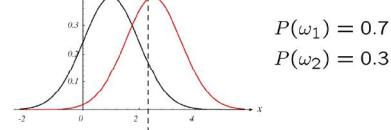
Modelos Generativos 6

Clasificación Bayesiana

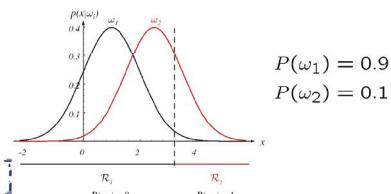
Influencia de $P(\omega_i)$:



$$P(\omega_1) = 0.5 \\ P(\omega_2) = 0.5$$

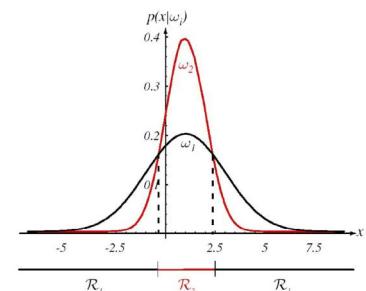


$$P(\omega_1) = 0.7 \\ P(\omega_2) = 0.3$$



$$P(\omega_1) = 0.9 \\ P(\omega_2) = 0.1$$

Influencia de $p(x|\omega_i)$:

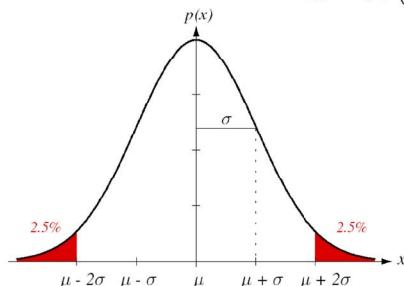


Modelos Generativos 7

La distribución Gaussiana

Monovariable:

$$x \sim \mathcal{N}(\mu, \sigma^2)$$



$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

$$Pr \{ |x - \mu| \leq 2\sigma \} \simeq 0.95$$

Multivariable:

$$\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

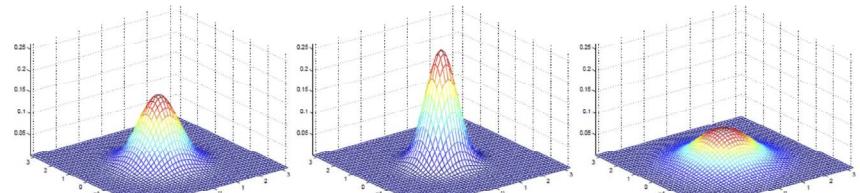
$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^D/2} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \exp \left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right\}$$

La distribución Gaussiana

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^D/2} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \exp \left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right\}$$

$$\mathbb{E}[\mathbf{x}] = \boldsymbol{\mu}$$

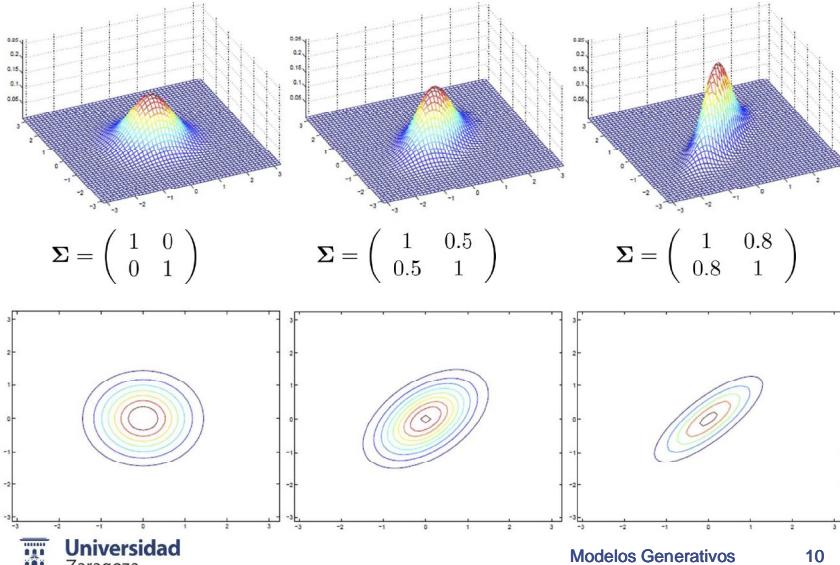
$$\text{cov}[\mathbf{x}] = \mathbb{E} [(\mathbf{x} - \mathbb{E}[\mathbf{x}])(\mathbf{x} - \mathbb{E}[\mathbf{x}])^\top] = \boldsymbol{\Sigma}$$



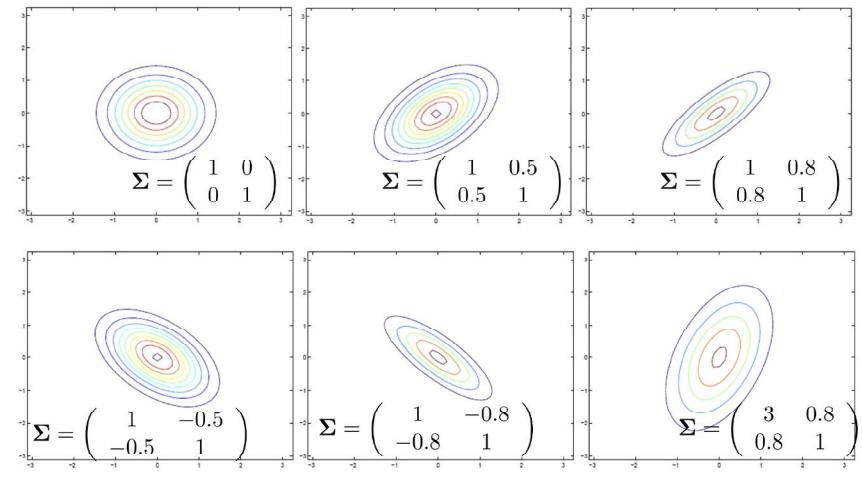
$$\boldsymbol{\Sigma} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \boldsymbol{\Sigma} = \begin{pmatrix} 0.6 & 0 \\ 0 & 0.6 \end{pmatrix} \quad \boldsymbol{\Sigma} = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$$

Diagonal → las variables x_i son independientes

La distribución Gaussiana



La distribución Gaussiana



La distribución Gaussiana

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \exp \left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right\}$$

- El lugar de los puntos de densidad de probabilidad constante son hyper-elipsoides, en los que:

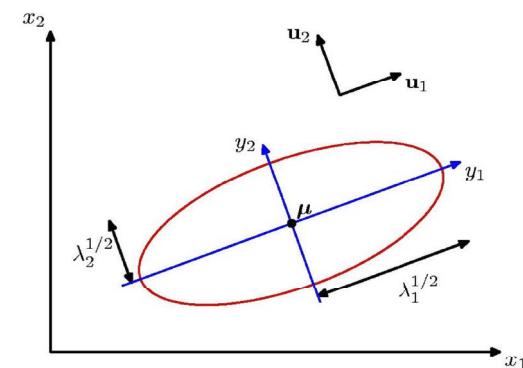
$$r^2 = (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) = cte$$

- r : Distancia de Mahalanobis de \mathbf{x} a $\boldsymbol{\mu}$

La distribución Gaussiana

■ Matriz de Covarianza

- Los ejes de los elipsoides son los vectores propios de $\boldsymbol{\Sigma}$
- La longitud de los ejes corresponde a los valores propios de $\boldsymbol{\Sigma}$



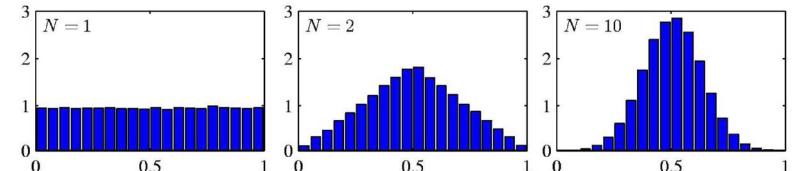
¿Por qué usar modelos Gaussianos?

- Muchas veces media y varianza es lo único que se puede estimar con precisión a partir de los datos de entrenamiento
- Si solo conocemos la media y la covarianza de una variable aleatoria, la distribución Gaussiana es la de máxima entropía
- Es decir, la que hace menos suposiciones adicionales

¿Por qué usar modelos Gaussianos?

- Teorema Central del Límite: La suma de N variables aleatorias idénticamente distribuidas tiende a una Gaussiana al aumentar N

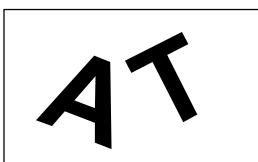
- Ejemplo: suma de N distribuciones uniformes [0, 1]



- La suma de perturbaciones aleatorias independientes conduce a distribuciones Gausianas
- Muchos atributos pueden verse como un valor ideal corrompido por numerosos procesos aleatorios
 - El modelo Gaussiano es a menudo un buen modelo

¿Por qué usar modelos Gaussianos?

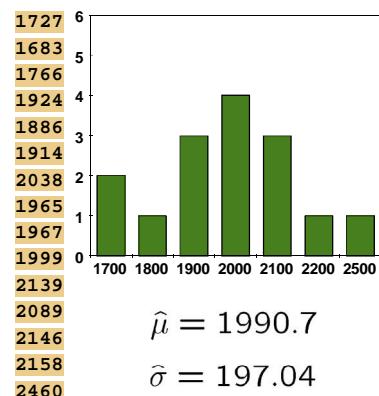
- Reconocimiento de objetos basado en descriptores: atributos globales invariantes



- Área
- Perímetro
- Elongación
- ...

- Ejemplo: el área

N=15



Aprendizaje del modelo Gaussiano

- Datos de entrenamiento:

$$\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^N \quad \mathbf{x} = (x_1, \dots, x_D)^T$$

- Modelo de cada clase j:

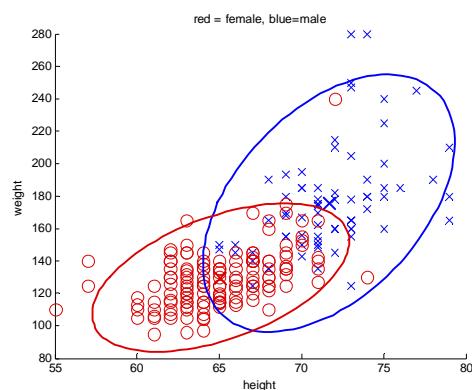
$$p(y_j) = \frac{N_j}{N} \quad N_j = \sum_i [y^{(i)} = j]$$

$$p(\mathbf{x}|y_j) \sim \mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \quad \boldsymbol{\mu}_j = \frac{1}{N_j} \sum_{y^{(i)}=j} \mathbf{x}^{(i)}$$

$$\boldsymbol{\Sigma}_j = \frac{1}{N_j - 1} \sum_{y^{(i)}=j} (\mathbf{x}^{(i)} - \boldsymbol{\mu}_j)(\mathbf{x}^{(i)} - \boldsymbol{\mu}_j)^T$$

Nº de parámetros de cada Gaussiana: D + D(D+1)/2

Ejemplo de modelo Gaussiano



Ajuste de una Gaussiana para cada clase.
El 95% de la probabilidad está contenida en las elipses

Análisis Gaussiano Discriminante (GDA)

■ Modelo Gaussiano

$$p(\mathbf{x}|y_j) = \frac{1}{(2\pi)^{D/2} |\Sigma_j|^{1/2}} \exp \left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_j)^T \Sigma_j^{-1} (\mathbf{x} - \boldsymbol{\mu}_j) \right\}$$

■ Clasificación Bayesiana:

$$p(y_i|\mathbf{x}) = \frac{p(\mathbf{x}|y_i)p(y_i)}{p(\mathbf{x})} = \frac{p(\mathbf{x}|y_i)p(y_i)}{\sum_{j=1}^C p(\mathbf{x}|y_j)p(y_j)}$$

$$\hat{y} = \arg \max_j p(y_j|\mathbf{x}) = \arg \max_j p(\mathbf{x}|y_j)p(y_j)$$

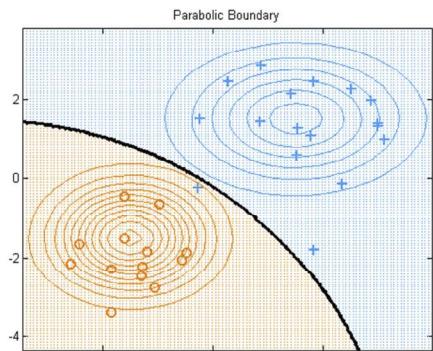
■ Luego:

$$\begin{aligned} \hat{y} &= \arg \max_j |\Sigma_j|^{-\frac{1}{2}} \exp \left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_j)^T \Sigma_j^{-1} (\mathbf{x} - \boldsymbol{\mu}_j) \right\} p(y_j) \\ &= \arg \max_j -\frac{1}{2} \ln |\Sigma_j| - \frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_j)^T \Sigma_j^{-1} (\mathbf{x} - \boldsymbol{\mu}_j) + \ln p(y_j) \end{aligned}$$

Análisis Gaussiano Discriminante (GDA)

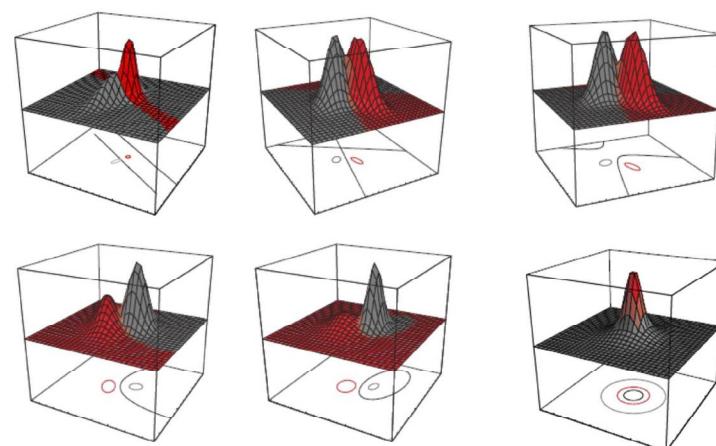
■ La superficie de separación es cuadrática

■ QDA: Quadratic Discriminant Analysis



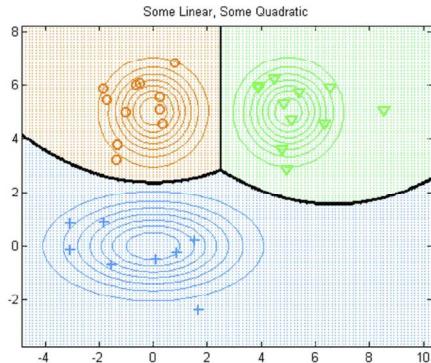
$$\hat{y} = \arg \max_j |\Sigma_j|^{-\frac{1}{2}} \exp \left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_j)^T \Sigma_j^{-1} (\mathbf{x} - \boldsymbol{\mu}_j) \right\} p(y_j)$$

Análisis Gaussiano Discriminante (GDA)



Análisis Gaussiano Discriminante (GDA)

- Ejemplo multi-clase



Estrategias para Evitar el Sobreajuste

- Con muchas dimensiones puede producirse sobreajuste

$$\text{size}(\boldsymbol{\mu}_j) = D \quad \text{size}(\boldsymbol{\Sigma}_j) = D \times D$$

- Nº de parámetros por clase: $D + D(D+1)/2$
- Si $N_j < D$, la covarianza no es inversible
- Aunque $N_j > D$, puede estar mal condicionada (mejor si $N_j > 10 D$)

Posibles soluciones

- Covarianza compartida para todas las clases $\boldsymbol{\Sigma}_j = \boldsymbol{\Sigma}$
- Bayes ingenuo: Atributos independientes \rightarrow covarianza diagonal $\boldsymbol{\Sigma}_j = \text{diag}(\sigma_{j1}^2, \dots, \sigma_{jD}^2)$

- Regularización en la estimación de las covarianzas

$$\boldsymbol{\Sigma}'_j = (1 - \lambda)\boldsymbol{\Sigma}_j + \lambda \text{diag}(\boldsymbol{\Sigma}_j)$$

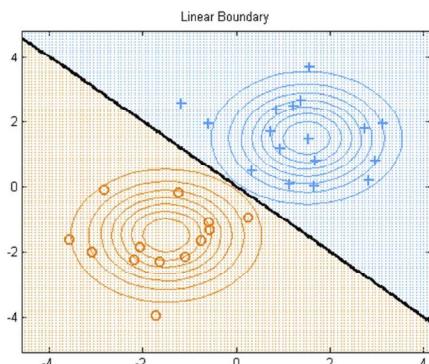
$$\boldsymbol{\Sigma}'_j = \boldsymbol{\Sigma}_j + \lambda \mathbf{I}_D$$

Covarianza Compartida entre Clases

- Análisis Discriminante Lineal (LDA)

- La superficie de separación es lineal (hyper-plano)

$$\boldsymbol{\Sigma}_j = \boldsymbol{\Sigma}$$



$$\hat{y} = \arg \max_j -\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_j)^T \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu}_j) + \ln p(y_j)$$

Aprendizaje del modelo LDA

- Datos de entrenamiento:

$$\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^N \quad \mathbf{x} = (x_1, \dots, x_D)^T$$

- Modelo de cada clase j:

$$p(y_j) = \frac{N_j}{N} \quad N_j = \sum_i [y^{(i)} = j]$$

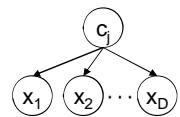
$$p(\mathbf{x}|y_j) \sim \mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}) \quad \boldsymbol{\mu}_j = \frac{1}{N_j} \sum_{y^{(i)}=j} \mathbf{x}^{(i)}$$

$$\boldsymbol{\Sigma} = \frac{1}{N-1} \sum_i (\mathbf{x}^{(i)} - \boldsymbol{\mu}_{y^{(i)}})(\mathbf{x}^{(i)} - \boldsymbol{\mu}_{y^{(i)}})^T$$

Nº de parámetros totales: $C D + D(D+1)/2$

Bayes Ingenuo con Atributos Gaussianos

- Supone que los atributos son condicionalmente independientes dada la clase:



$$p(\mathbf{x}|y_j) = \prod_{i=1}^D p(x_i|y_j)$$

- En el caso Gaussiano: $\Sigma_j = \text{diag}(\sigma_{j1}^2, \dots, \sigma_{jD}^2)$

$$\begin{aligned} p(\mathbf{x}|y_j) &= \frac{1}{(2\pi)^{D/2} |\Sigma_j|^{1/2}} \exp \left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_j)^T \Sigma_j^{-1} (\mathbf{x} - \boldsymbol{\mu}_j) \right\} \\ &= \prod_{i=1}^D \frac{1}{\sqrt{2\pi}\sigma_{ji}} \exp \left\{ -\frac{1}{2} \left(\frac{x_i - \mu_{ji}}{\sigma_{ji}} \right)^2 \right\} \end{aligned}$$

Nº de parámetros por clase: 2 D

Aprendizaje de Bayes Ingenuo Gaussiano

$$\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^N$$

$$y \in \{1, \dots, C\}$$

$$\mathbf{x} = (x_1, \dots, x_D)^T$$

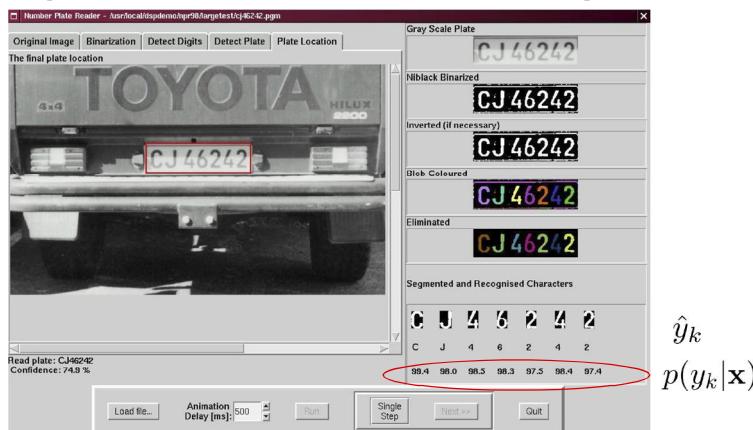
- Cada atributo se puede aprender de forma independiente
 - Atributo k de la clase j: $p(x_k|y_j) \sim \mathcal{N}(\mu_{jk}, \sigma_{jk}^2)$

$$N_j = \sum_i [y^{(i)} = j]$$

$$\mu_{jk} = \frac{1}{N_j} \sum_{y^{(i)}=j} x_k^{(i)}$$

$$\sigma_{jk}^2 = \frac{1}{N_j - 1} \sum_{y^{(i)}=j} (x_k^{(i)} - \mu_{jk})^2$$

Ejemplo: Reconocimiento con Descriptores



- London Toll Video System:

♦ 688 cameras in 203 places, 40.000 vehicles per hour

1. Umbralización de la Imagen



Imagen
Niveles de gris
•Pixels 0..255



Imagen
binaria
•Pixels 0 y 1

2. Operadores Morfológicos



Morfología

Imagen binaria



Imagen suavizada

3. Análisis de Conectividad



Conectividad

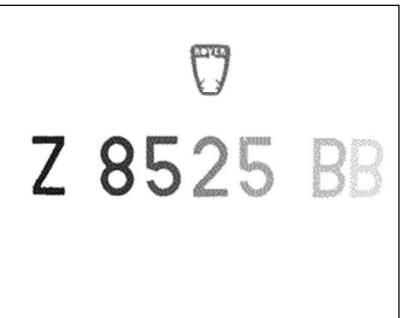
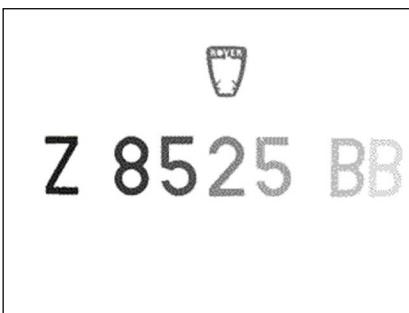


Imagen coloreada

- Partición de pixels 1
- Eliminación de regiones sin interés

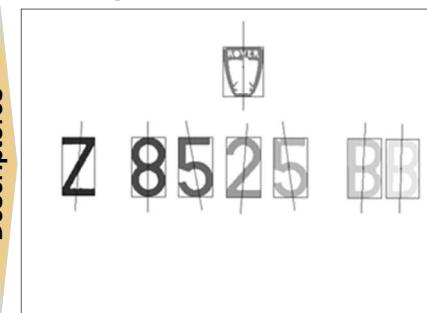
Modelos Generativos 31

4. Cálculo de Descriptores



Descriptores

Imagen
coloreada



- Imagen anotada
- Posición y orientación
 - Rect. Envolvente
 - Área, perímetro,...
 - Momentos de imagen

Modelos Generativos 32

5. Aprendizaje de los Descriptores

- Atributo k de la clase j:

- Media: valor representativo

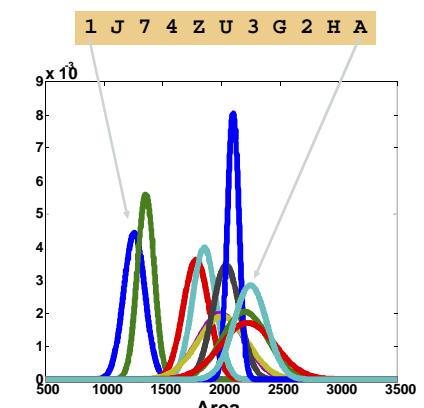
$$N_j = \sum_i [y^{(i)} = j]$$

$$\mu_{jk} = \frac{1}{N_j} \sum_{y^{(i)}=j} x_k^{(i)}$$

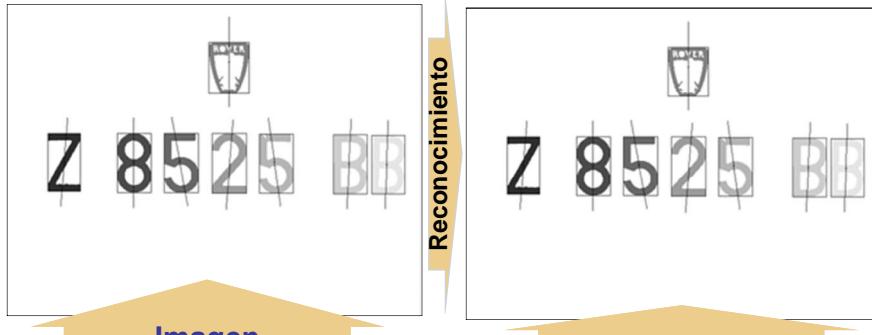
- Varianza: dispersión

$$\sigma_{jk}^2 = \frac{1}{N_j - 1} \sum_{y^{(i)}=j} (x_k^{(i)} - \mu_{jk})^2$$

$$p(x_k|y_j) \sim \mathcal{N}(\mu_{jk}, \sigma_{jk}^2)$$



6. Reconocimiento con Bayes Ingenuo



$$p(\mathbf{x}|y_j) = \prod_{i=1}^D \frac{1}{\sqrt{2\pi}\sigma_{ji}} \exp \left\{ -\left(\frac{x_i - \mu_{ji}}{2\sigma_{ji}} \right)^2 \right\}$$

$$\hat{y} = \arg \max_j p(\mathbf{x}|y_j)p(y_j)$$

Distancia de Mahalanobis

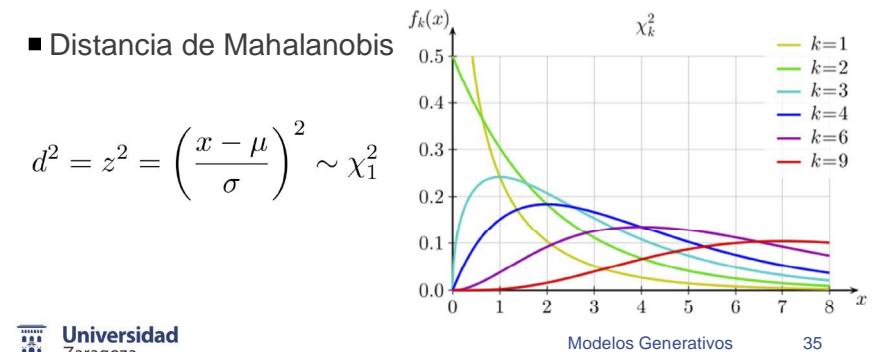
- Gaussiana monovariante

$$x \sim \mathcal{N}(\mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} \exp \left\{ -\frac{1}{2} \left(\frac{x - \mu}{\sigma} \right)^2 \right\}$$

$$z = \frac{x - \mu}{\sigma} \sim \mathcal{N}(0, 1)$$

- Distancia de Mahalanobis

$$d^2 = z^2 = \left(\frac{x - \mu}{\sigma} \right)^2 \sim \chi_1^2$$



Distancia de Mahalanobis

- Gaussiana Multivariante

$$\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{\frac{D}{2}} |\boldsymbol{\Sigma}|^{\frac{1}{2}}} \exp \left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right\}$$

$$\mathbf{z} = (\mathbf{R}^T)^{-1}(\mathbf{x} - \boldsymbol{\mu}) \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_D)$$

- Donde: $\boldsymbol{\Sigma} = \mathbf{R}^T \mathbf{R} = \mathbf{L} \mathbf{L}^T$

$$\mathbf{R} = \text{Chol}(\boldsymbol{\Sigma})$$

Descomposición de Cholesky
R: Matriz Triangular Superior
L=R^T: Matriz Triangular Inferior

- Distancia de Mahalanobis

$$d^2 = \mathbf{z}^T \mathbf{z} = (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) = \sum_{i=1}^D z_i^2 \sim \chi_D^2$$

Detección de Espurios: Test de chi-cuadrado

- Clasifico una muestra mediante:

$$\hat{y} = \arg \max_j |\boldsymbol{\Sigma}_j|^{-\frac{1}{2}} \exp \left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_j)^T \boldsymbol{\Sigma}_j^{-1} (\mathbf{x} - \boldsymbol{\mu}_j) \right\} p(y_j)$$

- ¿Pertenece realmente a la clase j?

- Distancia de la muestra a la clase:

$$d_j^2 = (\mathbf{x} - \boldsymbol{\mu}_j)^T \boldsymbol{\Sigma}_j^{-1} (\mathbf{x} - \boldsymbol{\mu}_j) \sim \chi_D^2$$

- Test de Chi-cuadrado:

$$d_j^2 \leq \chi_{D, 1-\alpha}^2$$

- α : nivel de significación del test

♦ α : 0.05, 0.01, 0.001

- Fracción de casos buenos rechazados

D	0.95	0.99
1	3.8415	6.6349
2	5.9915	9.2103
3	7.8147	11.3449
4	9.4877	13.2767
5	11.0705	15.0863
6	12.5916	16.8119
7	14.0671	18.4753
8	15.5073	20.0902
9	16.9190	21.6660
10	18.3070	23.2093
...		

Ejemplo con D=2

- Bayes ingenuo estima covarianzas diagonales

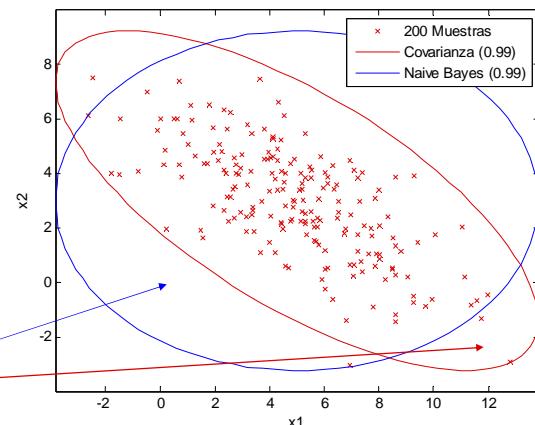
- Covarianza real:

$$\Sigma = \begin{pmatrix} 8 & -4 \\ -4 & 4 \end{pmatrix}$$

- Bayes Ingenuo

$$\Sigma = \begin{pmatrix} 8 & 0 \\ 0 & 4 \end{pmatrix}$$

- Falsos Positivos
- Falsos Negativos



Cálculos con Gaussianas en Matlab

$$d^2 = (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu})$$

$$\mathbf{z}^T = (\mathbf{x} - \boldsymbol{\mu})^T R^{-1}$$

```
function [d2, R] = mahalanobis(mu, Sigma, X)
% Distancias de Mahalanobis de las muestras X a
% la Gaussiana N(mu, Sigma) y R = chol(Sigma)
X_menos_mu = bsxfun(@minus, X, mu');
R = chol(Sigma);
z = X_menos_mu / R;
d2 = sum(z.^2, 2); %Distancia de Mahalanobis
```

Cálculos con Gaussianas en Matlab

$$p(\mathbf{x}|y) = \frac{1}{(2\pi)^{\frac{D}{2}} |\boldsymbol{\Sigma}|^{\frac{1}{2}}} \exp \left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right\}$$

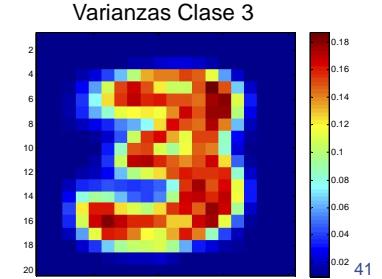
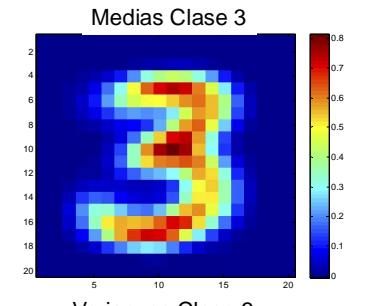
$$\ln(p(\mathbf{x}|y)) = -\frac{D}{2} \ln(2\pi) - \ln(|\boldsymbol{\Sigma}|^{\frac{1}{2}}) - \frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu})$$

```
function logp = gaussLog(mu, Sigma, X)
% Calcula el Logaritmo neperiano de la pdf
% Gaussiana para las muestras X
D = size(X, 2);
[d2,R] = mahalanobis(mu, Sigma, X);
logp = -D*log(2*pi)/2 - sum(log(diag(R))) - d2/2;
```

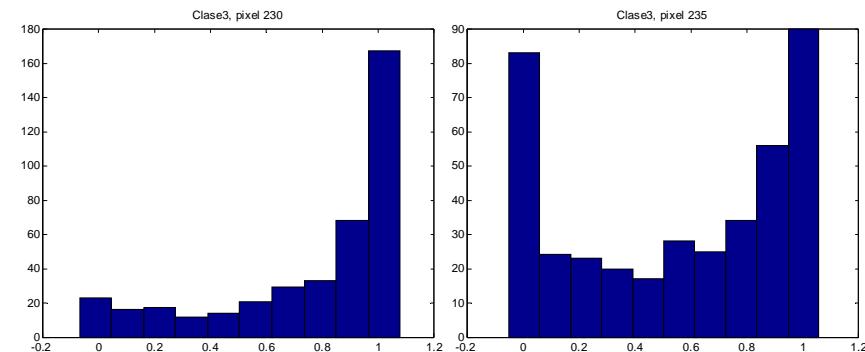
Práctica 5: Clasificación Bayesiana

- Atributos: 20x20 pixels

1	1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	3	3
4	4	4	4	4	4	4	4	4
5	5	5	5	5	5	5	5	5
6	6	6	6	6	6	6	6	6
7	7	7	7	7	7	7	7	7
8	8	8	8	8	8	8	8	8
9	9	9	9	9	9	9	9	9
0	0	0	0	0	0	0	0	0

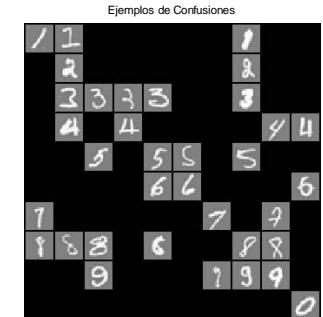
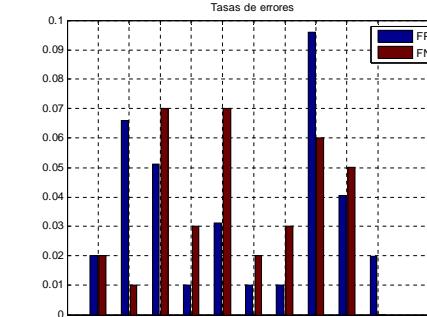


¿Son Gaussianos los Atributos?



Y sin embargo Funciona...

■ Clasificación Bayesiana con Correlación completa



- ♦ Va mejor que la regresión logística
- ♦ ¿Por qué?

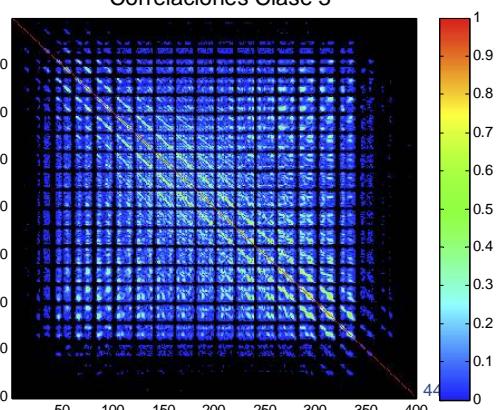
¿Son Independientes los Atributos?

$$\Sigma = \begin{pmatrix} \sigma_1^2 & \rho_{12}\sigma_1\sigma_2 & \dots & \rho_{1D}\sigma_1\sigma_D \\ \rho_{12}\sigma_1\sigma_2 & \sigma_2^2 & \dots & \rho_{2D}\sigma_2\sigma_D \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{1D}\sigma_1\sigma_D & \rho_{2D}\sigma_2\sigma_D & \dots & \sigma_D^2 \end{pmatrix}$$

Correlaciones Clase 3

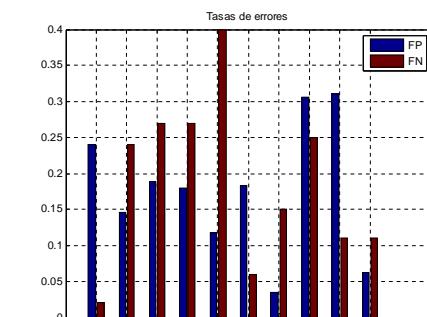
■ Matriz de correlaciones:

$$\rho = \begin{pmatrix} 1 & \rho_{12} & \dots & \rho_{1D} \\ \rho_{12} & 1 & \dots & \rho_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{1D} & \rho_{2D} & \dots & 1 \end{pmatrix}$$



Ignorando las Correlaciones

■ Clasificación con Bayes ingenuo



- ♦ Se pierde mucha precisión
- ♦ Va peor que la regresión logística

Índice

- Planteamiento
- Modelo Gaussiano
- Aprendizaje y Evaluación del Modelo

Tema 5. Detección de Anomalías

30231 - Aprendizaje Automático
Grado en Ingeniería Informática
Escuela de Ingeniería y Arquitectura

Créditos de transparencias y figuras:

- Andrew Ng, *Machine Learning*, Stanford AI Lab.
<https://www.coursera.org/course/ml>

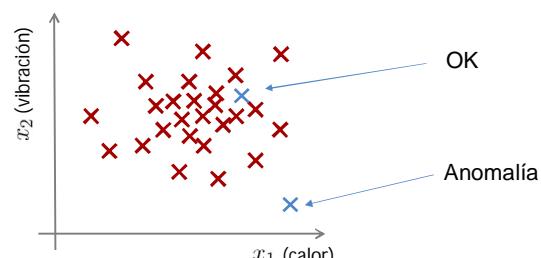
Ejemplo

Atributos Motor de Avión:

→ x_1 = calor generado

→ x_2 = vibración

...



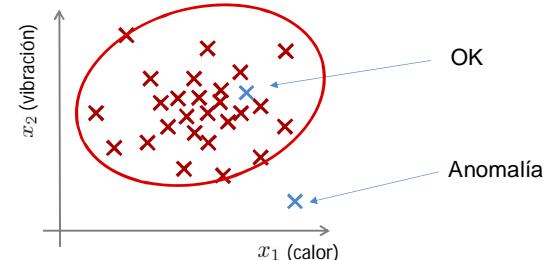
Datos: $\{x^{(1)}, x^{(2)}, \dots, x^{(N)}\}$

Nueva muestra: x_{test}

Detección de Anomalías

- Es un problema de clasificación supervisada binaria:

- $y=0$ - Clase Normal: muchas muestras de entrenamiento
- $y=1$ - Anomalías: muy pocas muestras disponibles



- Aproximación (modelo generativo)

- Aprender la distribución de la clase normal: $p(\mathbf{x}) \equiv p(\mathbf{x}|y = 0)$
- Detección de una anomalía cuando: $p(\mathbf{x}_{test}) < \epsilon$

Ejemplos de Aplicación

Detección de fraudes

- Descriptores de la actividad del usuario i:
- Modelar a partir de los datos de usuarios
- Identificar usuarios inusuales cuando

$$\mathbf{x}^{(i)} = (x_1^{(i)}, \dots, x_D^{(i)})^T$$

$$p(\mathbf{x})$$

$$p(\mathbf{x}_{test}) < \epsilon$$

Monitorización de computadores en un centro de datos

- Descriptores de la actividad de la máquina i: $\mathbf{x}^{(i)} = (x_1^{(i)}, \dots, x_D^{(i)})^T$
- x_1 = uso de memoria
- x_2 = accesos a disco por segundo
- x_3 = carga de CPU
- x_4 = carga de CPU / tráfico de red
-

Manufactura, ...



Detección de anomalías .vs. Clasificación Supervisada

- Muy pocos ejemplos de casos positivos (0-20 es habitual)

- Gran número de ejemplos negativos

- Muchos tipos diferentes de anomalías. Es muy difícil que algún algoritmo pueda aprender qué pinta tienen las anomalías

- Las anomalías futuras pueden no parecerse a las que hemos visto hasta ahora.

- Gran número de ejemplos positivos y negativos

- Suficientes ejemplos para que el algoritmo de aprendizaje pueda aprender cómo son los casos positivos.

- Los ejemplos positivos futuros serán probablemente similares a los que tenemos en los datos de entrenamiento

Detección de anomalías .vs. Clasificación Supervisada

Detección de fraudes

Detección de SPAM

Manufactura (motores de avión)

Predicción del tiempo (soleado, nublado, lluvia, nieve,...)

Monitorización de máquinas en un centro de datos

Clasificación de Cáncer

Modelo Gaussiano

- Datos de entrenamiento de casos normales:

$$\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)} = 0)\}_{i=1}^N \quad \mathbf{x} = (x_1, \dots, x_D)^T$$

- Modelo de los casos normales:

$$p(\mathbf{x}|y=0) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \quad \boldsymbol{\mu} = \frac{1}{N} \sum_{i=1}^N \mathbf{x}^{(i)}$$

$$\boldsymbol{\Sigma} = \frac{1}{N-1} \sum_{i=1}^N (\mathbf{x}^{(i)} - \boldsymbol{\mu})(\mathbf{x}^{(i)} - \boldsymbol{\mu})^T$$

Nº de parámetros del modelo: $D + D(D+1)/2$



Detección con Distancia de Mahalanobis

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{\frac{D}{2}} |\Sigma|^{\frac{1}{2}}} \exp \left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right\}$$

$$p(\mathbf{x}_{test}) < \epsilon \Leftrightarrow (\mathbf{x}_{test} - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x}_{test} - \boldsymbol{\mu}) > \epsilon'$$

- Distancia de Mahalanobis:

$$d^2 = (\mathbf{x} - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}) \sim \chi_D^2$$

- Test de Chi-cuadrado

- Señalar anomalía cuando: $d^2 > \chi_{D, 1-\alpha}^2$
- α : nivel de significación del test = 0.05, 0.01, 0.001
- Probabilidad de falsa alarma
- Puede ajustarse por validación cruzada

Modelo Gaussiano Independiente (ingenuo)

- Supone que los atributos son independientes
 - Matriz de covarianza diagonal

$$\Sigma = \text{diag}(\sigma_1^2, \dots, \sigma_D^2)$$

- Cada atributo se puede aprender de forma independiente

- Atributo k:

$$p(x_k | y = 0) \sim \mathcal{N}(\mu_k, \sigma_k^2)$$

$$\mu_k = \frac{1}{N} \sum_{i=1}^N x_k^{(i)}$$

$$\sigma_k^2 = \frac{1}{N-1} \sum_{i=1}^N (x_k^{(i)} - \mu_k)^2$$

Nº de parámetros del modelo: 2D

Ejemplo con D=2

- Si utilizamos covarianzas diagonales y no lo son

- Covarianza real:

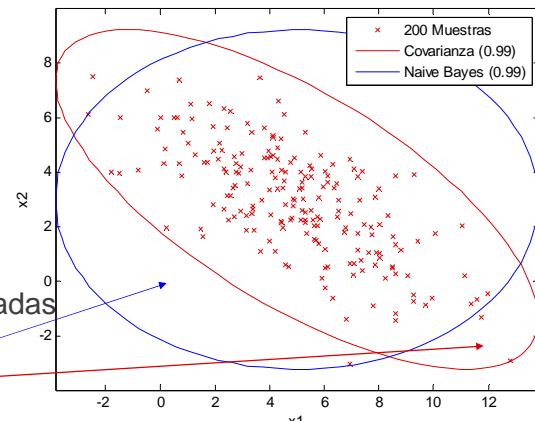
$$\Sigma = \begin{pmatrix} 8 & -4 \\ -4 & 4 \end{pmatrix}$$

- Modelo diagonal:

$$\Sigma = \begin{pmatrix} 8 & 0 \\ 0 & 4 \end{pmatrix}$$

- Anomalías detectadas

- Falsos Negativos
- Falsos Positivos



Modelo Independiente .vs. Gaussiano Multivariable

$$\Sigma = \text{diag}(\sigma_1^2, \dots, \sigma_D^2)$$

$$p(\mathbf{x}) = \frac{\exp \left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right\}}{(2\pi)^{\frac{D}{2}} |\Sigma|^{\frac{1}{2}}}$$

- Añadir manualmente atributos que capturen las dependencias

$$x_5 = \frac{\text{cargaCPU}}{\text{TraficoRed}}$$

- Menor carga computacional
 - Escala mejor si D es muy grande

- Funciona bien si N es pequeño

- La covarianza captura automáticamente las dependencias (lineales) entre atributos

- Requiere invertir la matriz de covarianza O(D³)

- No es inversible si N < D

- Más propenso al sobre-ajuste
 - Si hay pocos datos, usar regularización

Aprendizaje y Evaluación del Modelo

■ Separar los datos disponibles

■ Entrenamiento $\{(\mathbf{x}_{tr}^{(i)}, y_{tr}^{(i)})\}_{i=1}^{N_{tr}} ; y_{tr}^{(i)} = 0$

■ Validación $\{(\mathbf{x}_{cv}^{(i)}, y_{cv}^{(i)})\}_{i=1}^{N_{cv}} ; y_{cv}^{(i)} \in \{0, 1\}$

■ Test $\{(\mathbf{x}_{test}^{(i)}, y_{test}^{(i)})\}_{i=1}^{N_{test}} ; y_{test}^{(i)} \in \{0, 1\}$

■ Métricas de evaluación

- Matriz de confusión
- Precision / Recall
- F₁ score

Ejemplo

■ Motores de Avión, supongamos que tenemos:

■ 10000 motores buenos (normales)

■ 20 motores malos (anomalías)

■ Forma de separarlos (elegidos aleatoriamente):

■ Entrenamiento: 6000 buenos

■ Validación: 2000 buenos ($y=0$), 10 malos ($y=1$)

■ Test: 2000 buenos ($y=0$), 10 malos ($y=1$)

Selección de Atributos

- Elegir atributos que puedan tomar valores muy altos o muy bajos en presencia de anomalías
- Monitorización de computadores en un centro de datos
 - x_1 = uso de memoria
 - x_2 = accesos a disco por segundo
 - x_3 = carga de CPU
 - x_4 = tráfico de red

$$x_5 = \frac{\text{cargaCPU}}{\text{TraficoRed}}$$

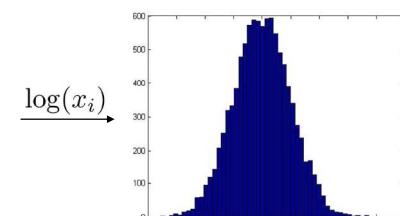
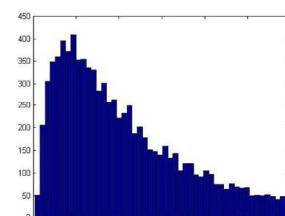
$$x_6 = \frac{(\text{cargaCPU})^2}{\text{TraficoRed}}$$

¿Y si los atributos no son Gaussianos?

■ Probar transformaciones sencillas:

$$x'_i = \log(x_i) \quad x'_i = \sqrt{x_i}$$

$$x'_i = \log(x_i + c) \quad x'_i = \sqrt[n]{x_i}$$



$\log(x_i) \rightarrow$

En Matlab: hist(x,100)

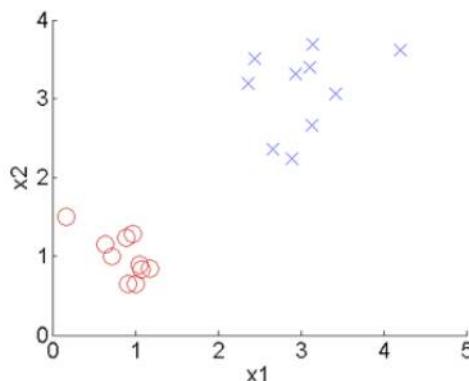
Tema 6. Aprendizaje no supervisado. Análisis de Componentes Principales

30231 - Aprendizaje Automático
Grado en Ingeniería Informática
Escuela de Ingeniería y Arquitectura



Aprendizaje supervisado

- Entrada: conjunto de datos $D = \{\mathbf{x}^{(i)}, \mathbf{y}^{(i)}\}_{i=1}^m$
- Ejemplos:
- Regresión: $\mathbf{x}^{(i)} \in \mathbb{R}^n, \mathbf{y}^{(i)} \in \mathbb{R}$
- Clasificación: $\mathbf{x}^{(i)} \in \mathbb{R}^n, \mathbf{y}^{(i)} \in C = \{1, 2, \dots, K\}$



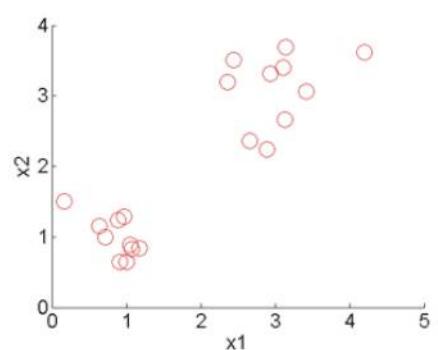
Índice

- Introducción al aprendizaje no supervisado
- Introducción al PCA
 - Reducción de dimensión de 2d a 1d y 3d a 2d
- Algoritmos para PCA
 - Estandarización de los datos
 - PCA basado en vectores propios y valores propios
 - Recuperación aproximada de la dimensionalidad
 - Descomposición en valores singulares (SVD)
- Bibliografía, lecturas y material recomendado:
 - Bishop; K.P. Murphy
 - Nando de Freitas (UBC)
 - Andrew Ng (Stanford)

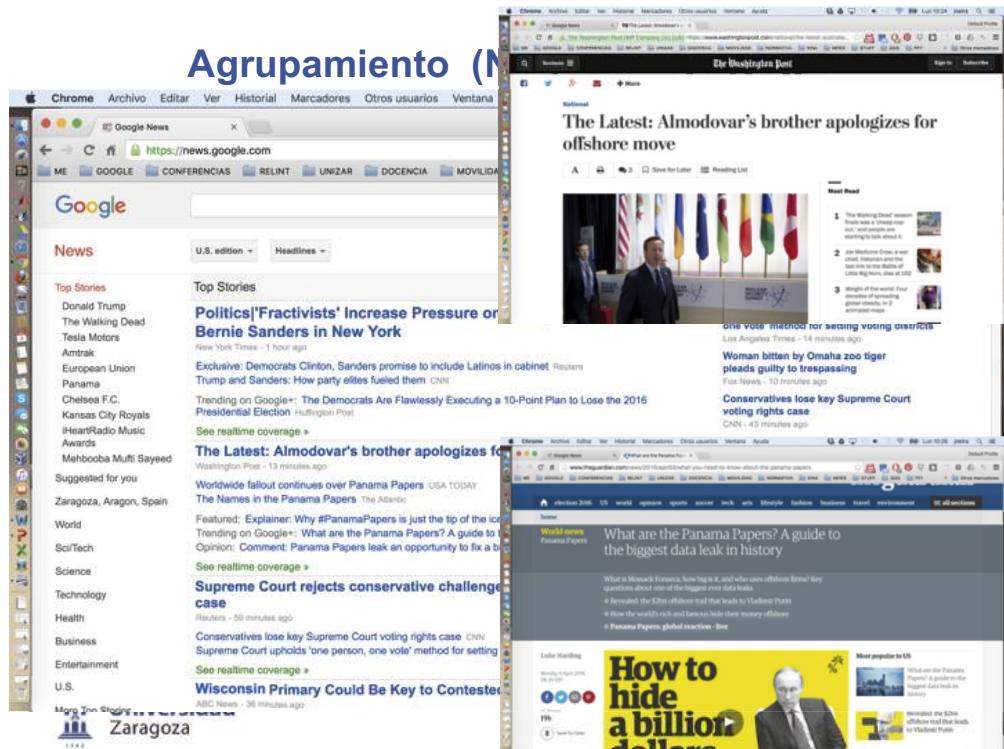
Aprendizaje no supervisado

- Entrada: conjunto de datos $D = \{\mathbf{x}^{(i)}\}_{i=1}^m, \mathbf{x}^{(i)} \in \mathbb{R}^n$
- Objetivo: encontrar información sobre la estructura de los datos

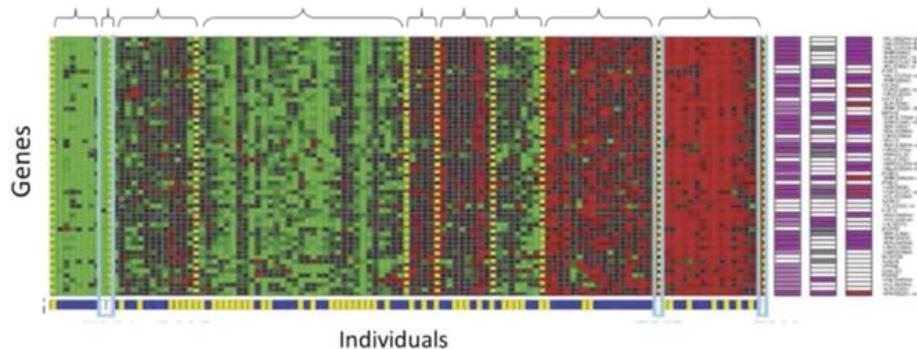
- Etiquetas no disponibles (o muy caras de obtener)
- Descubrir conocimiento (agrupamiento)
- Visualización datos en alta dimensión
- Compresión y reducción de dimensión



Agrupamiento (NMF)



Agrupamiento (Koller):

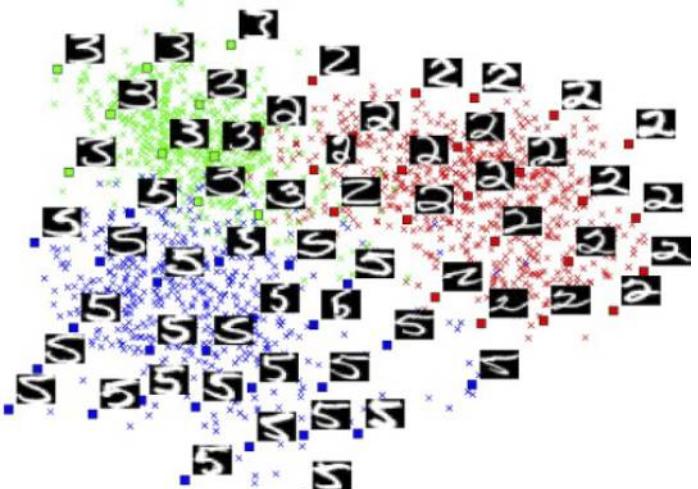


[Source: Su-In Lee, Dana Pe'er, Aimee Dudley, George Church, Daphne Koller]



6

Reducción de dimensión: compresión de datos



¿podemos usar datos de menor dimensión?

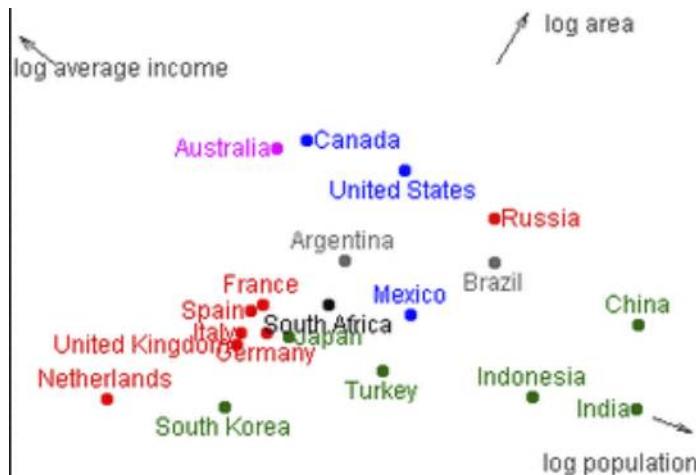
Reducción de dimensión: visualización

The figure is a screenshot of Microsoft Excel with a table titled 'ExcelCountry'. The table has columns for Country, Continent, Population, Total Area, Life Exp., and Pop. Growth. The data includes information for countries like United States, France, Argentina, China, Chad, Denmark, Egypt, Chile, Japan, Canada, and Algeria.

	A	B	C	D	E	F
1	Country	Continent	Population	Total Area	Life Exp.	Pop. Growth
2	United States	North America	298,444,215	9,631,420	77.85	0.91%
3	France	Europe	60,876,136	547,030	79.73	0.35%
4	Argentina	South America	39,921,833	2,766,890	76.12	0.96%
5	China	Asia	1,313,973,713	9,596,960	72.58	0.59%
6	Chad	Africa	9,944,201	1,284,000	47.52	2.93%
7	Denmark	Europe	5,450,661	43,094	77.79	0.33%
8	Egypt	Africa	78,887,007	1,001,450	71.29	1.75%
9	Chile	South America	16,134,219	756,950	76.77	0.94%
10	Japan	Asia	127,463,611	377,835	81.25	0.02%
11	Canada	North America	33,098,932	9,984,670	80.22	0.88%
12	Algeria	Africa	32,930,091	2,381,740	73.26	1.22%

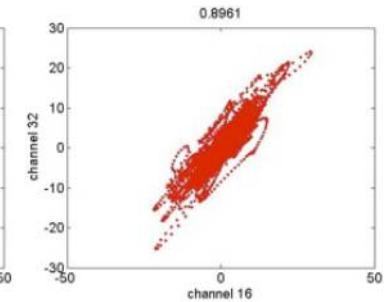
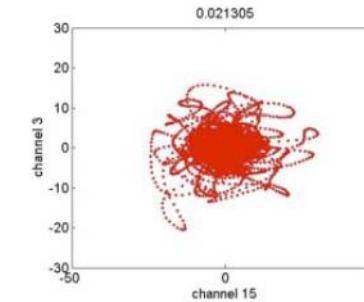
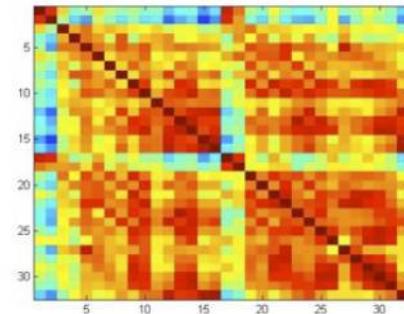
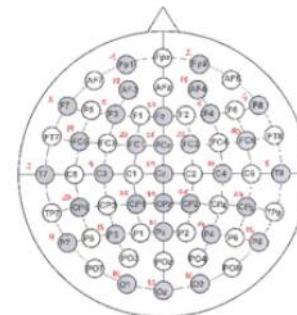
Difícil de visualizar e interpretar

Reducción de dimensión: visualización



Más fácilmente interpretable

Datos de electroencefalografía



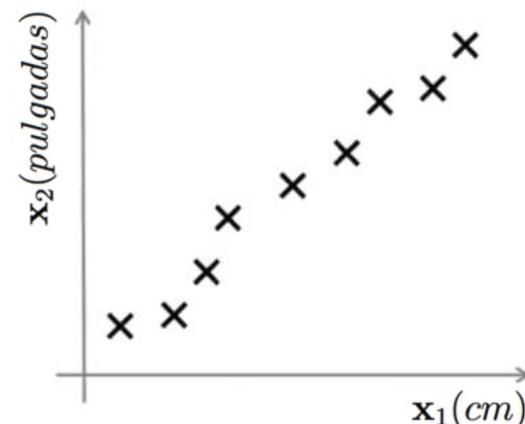
Ejercicio

De los siguientes problemas, cuáles se deben resolver con aprendizaje no supervisado?

- Dado un conjunto de emails etiquetados como spam / no spam, programar un filtro de spam.
- Dado un conjunto de artículos de periódicos, agruparlos en artículos sobre la misma noticia
- Dada una base de datos de consumo de clientes, descubrir segmentos de mercado y agrupar a los clientes según estos segmentos.
- Dado un conjunto de pacientes diagnosticados o no con diabetes, clasificar nuevos pacientes como diabéticos o no.

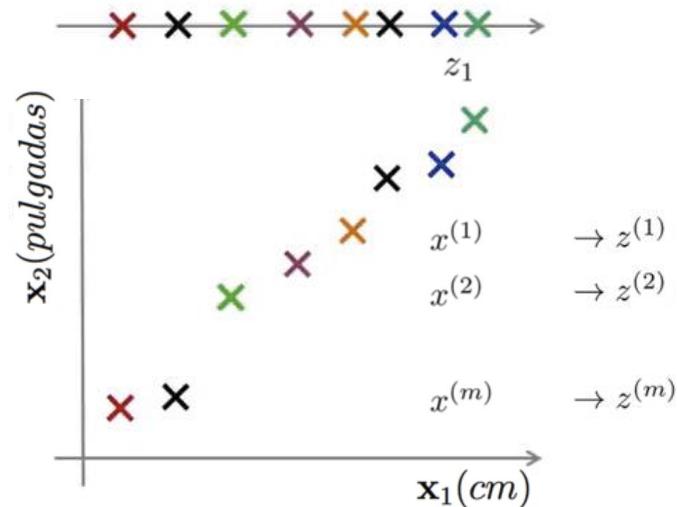
Reducción de dimensión: 2D a 1D

- Datos correlados: longitud en cm y pulgadas, velocidad en km/h y en m/sec (o longitud de piernas v. velocidad al correr)



¿Porqué no forman una línea recta?

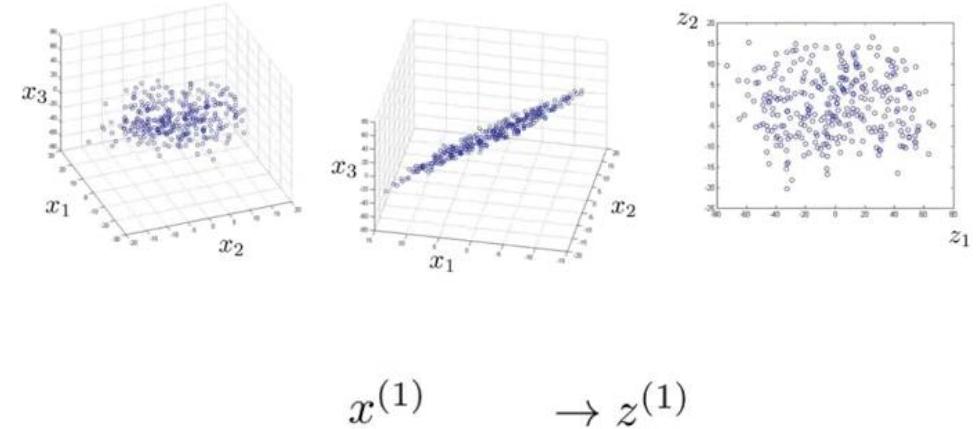
Reducción de dimensión: 2D a 1D



¿Se pierde información?

13

Reducción de dimensión 3D a 2D



14

Ejercicio

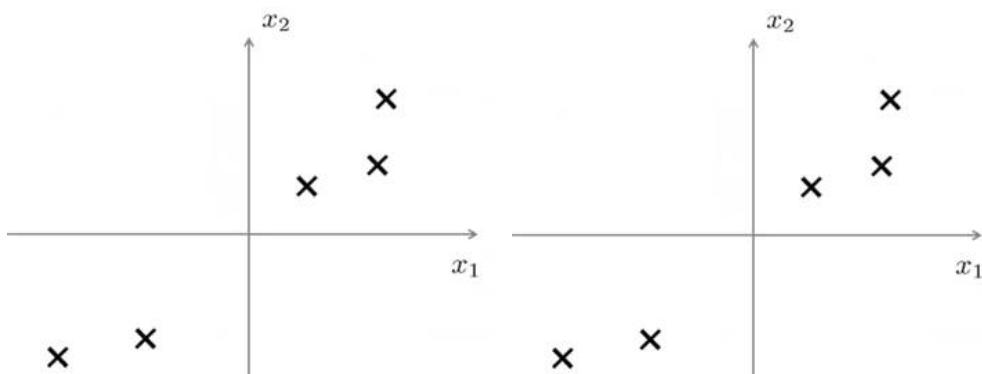
Si aplicamos reducción de la dimensionalidad a m muestras $\{x^{(1)}, x^{(2)}, \dots, x^{(m)}\}$, con $x^{(i)} \in \mathbb{R}^n$, obtenemos:

- k muestras $\{z^{(1)}, z^{(2)}, \dots, z^{(k)}\}$, con $k \leq n$
- k muestras $\{z^{(1)}, z^{(2)}, \dots, z^{(k)}\}$, con $k > n$
- m muestras $\{z^{(1)}, z^{(2)}, \dots, z^{(m)}\}$, con $z^{(i)} \in \mathbb{R}^k$, $k > n$
- m muestras $\{z^{(1)}, z^{(2)}, \dots, z^{(m)}\}$, con $z^{(i)} \in \mathbb{R}^k$, $k \leq n$

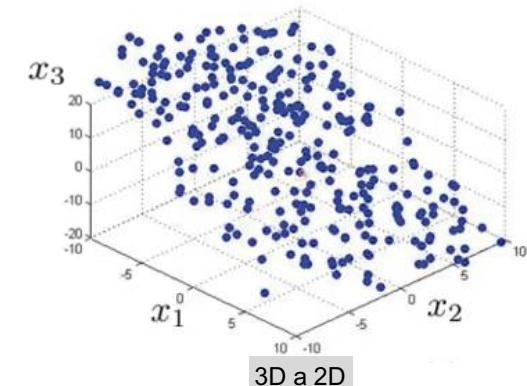
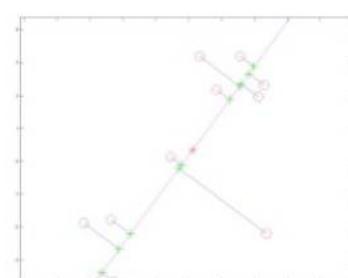
15

16

Análisis de componentes principales (PCA)



Minimizar el error de reprojeción:

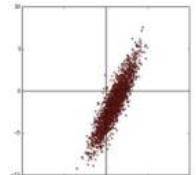


Objetivo: minimizar el error de reprojeción

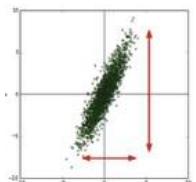
Estandarización de los datos

- Dados los datos: $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(m)}$
- Normalización de la media:

$$-\mu_j = \frac{1}{m} \sum_{i=1}^m x_j^{(i)}$$

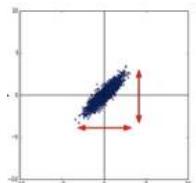


$$- \text{Reemplazar } x_j^{(i)} \text{ con } x_j^{(i)} - \mu_j$$



- Normalización de la varianza:

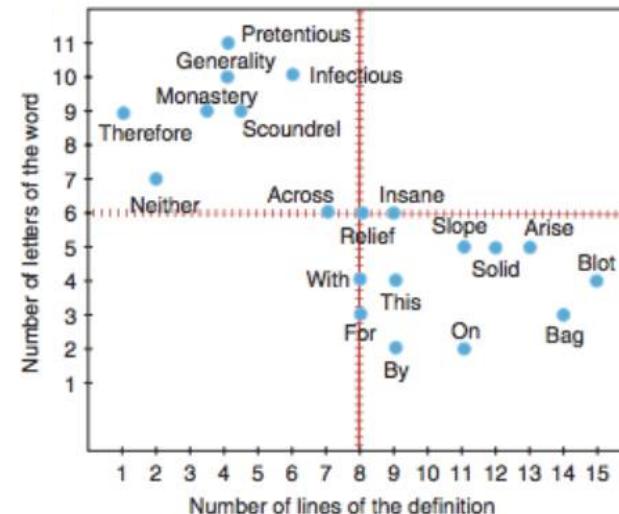
$$-\sigma_j^2 = \frac{1}{m-1} \sum_{i=1}^m (x_j^{(i)} - \mu_j)^2$$



$$- \text{Reemplazar } x_j^{(i)} \text{ con } \frac{x_j^{(i)} - \mu_j}{\sigma_j}$$

¿da igual $u^{(i)}$ que $-u^{(i)}$?

Ejercicio



$$u^{(1)} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

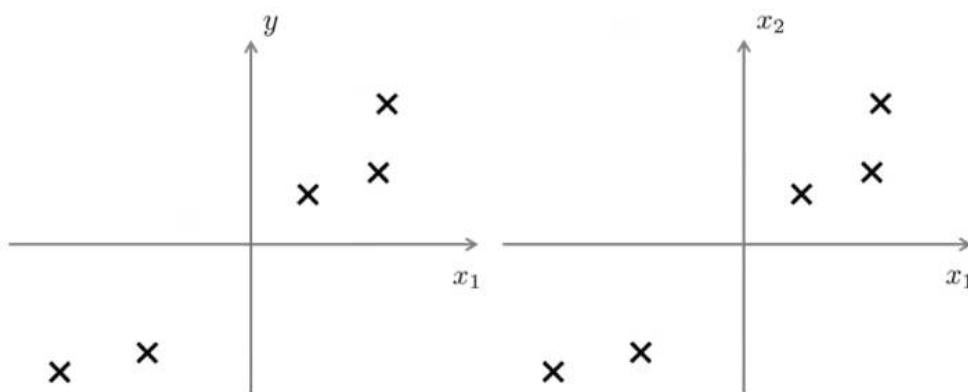
$$u^{(1)} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$u^{(1)} = \begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix}$$

$$u^{(1)} = \begin{bmatrix} -1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix}$$

¿cuál vector minimiza el error de reprojeción?

Problemas similares: ¿regresión?



$$\hat{y} = h_{\theta}(\mathbf{x}) = \theta_0 + \theta_1 x_1$$

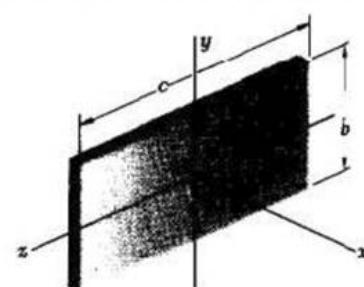
Problemas similares: eje de mínima inercia

Thin rectangular plate

$$I_x = \frac{1}{12}m(b^2 + c^2)$$

$$I_y = \frac{1}{12}mc^2$$

$$I_z = \frac{1}{12}mb^2$$

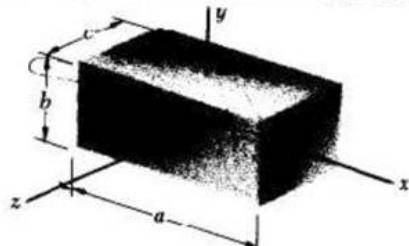


Rectangular prism

$$I_x = \frac{1}{12}m(b^2 + c^2)$$

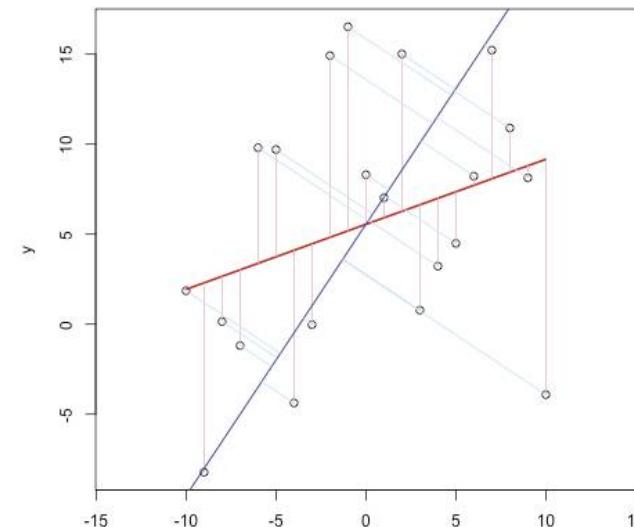
$$I_y = \frac{1}{12}m(c^2 + a^2)$$

$$I_z = \frac{1}{12}m(a^2 + b^2)$$

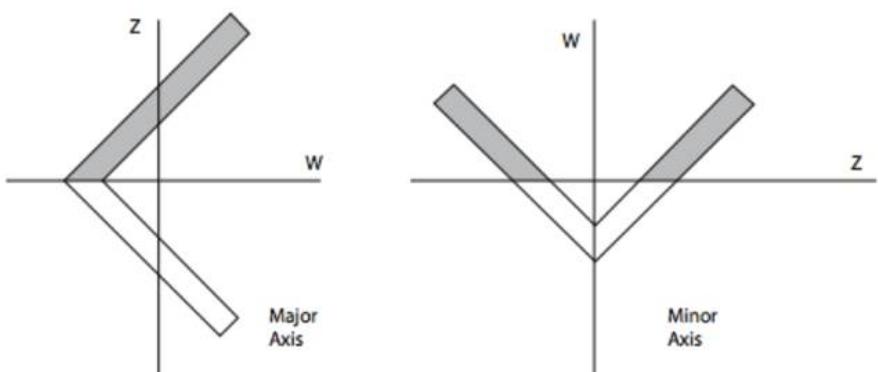


Ej eje de mínima inercia es el eje mínimo de orden 2.

Problemas similares: ¿regresión?



La regresión minimiza los residuos,
PCA las distancias normales



Ej eje de mínima inercia es el eje mínimo de orden 2.

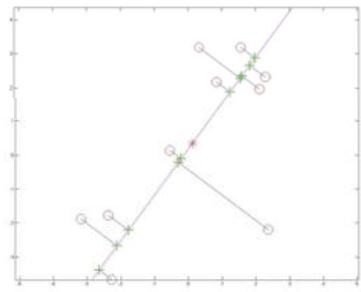
Fundamentos del algoritmo PCA

- Dado un conjunto de datos estandarizados

$$\mathbf{X} = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(m)}\} \in \mathbb{R}^n$$

- Aplicar una transformación lineal para que cada dato $\mathbf{x}^{(i)}$ se transforme en $\hat{\mathbf{x}}^{(i)}$, de tal forma que se minimice:

$$J(\mathbf{U}, \mathbf{Z}) = \frac{1}{N} \sum_{i=1}^N \|\mathbf{x}^{(i)} - \hat{\mathbf{x}}^{(i)}\|^2$$



Ejemplo 2D a 1D

25

Reducción de la dimensión

- Reducción de la dimensión: utilizar los primeros k vectores de \mathbf{U} :

$$\mathbf{U}_k = \left(\begin{array}{c|c|c|c} u^{(1)} & u^{(2)} & \cdots & u^{(k)} \\ \hline | & | & & | \end{array} \right)_{n \times k}$$

- $\mathbf{z}^{(i)} = \mathbf{U}_k^T \mathbf{x}^{(i)}$ es la transformación de cada dato a la nueva base
- $\hat{\mathbf{x}}^{(i)} = \mathbf{U}_k \mathbf{z}^{(i)}$ es la reconstrucción de cada dato
- Si $k = n$, $\mathbf{x}^{(i)} = \hat{\mathbf{x}}^{(i)}$,
- Si $k < n$, $\mathbf{x}^{(i)} \neq \hat{\mathbf{x}}^{(i)}$,

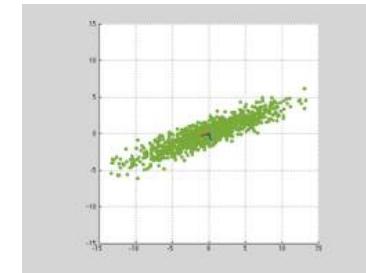
27

Fundamentos del algoritmo PCA

- Podemos utilizar cualquier base ortonormal \mathbf{U} con vectores $\mathbf{u}^{(j)}$:

$$\mathbf{U} = \left(\begin{array}{c|c|c|c} & & & \\ u^{(1)} & u^{(2)} & \cdots & u^{(n)} \\ \hline | & | & & | \end{array} \right)_{n \times n}$$

- $\mathbf{z}^{(i)} = \mathbf{U}^T \mathbf{x}^{(i)}$ es la transformación de cada dato a la nueva base
- $\hat{\mathbf{x}}^{(i)} = \mathbf{U} \mathbf{z}^{(i)}$ es la reconstrucción de cada dato

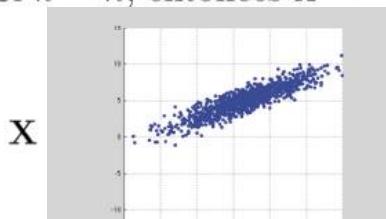


26

Ejemplo

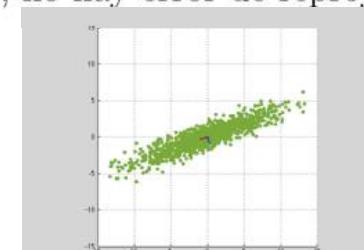
$$\mathbf{U}_k = \begin{pmatrix} -0.9382 & 0.3460 \\ -0.3460 & -0.9382 \end{pmatrix}$$

- Si $k = n$, entonces $\mathbf{x}^{(i)} = \hat{\mathbf{x}}^{(i)}$, no hay error de reprojeción



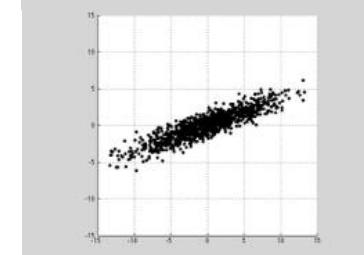
X

$\mathbf{z}^{(i)}$



Est

28



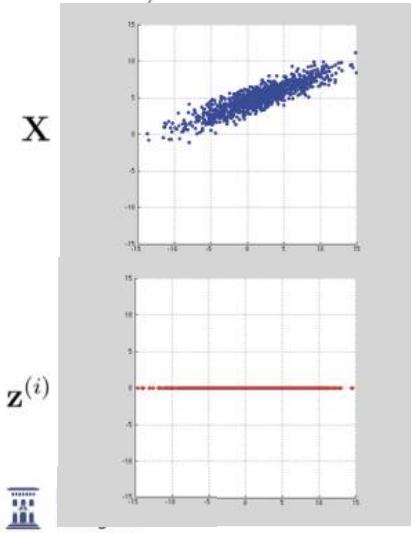
$\hat{\mathbf{x}}^{(i)}$

Ejemplo

$U_k =$

$$\begin{bmatrix} -0.9384 \\ -0.3455 \end{bmatrix}$$

- Si $k < n$, entonces $\mathbf{x}^{(i)} \neq \hat{\mathbf{x}}^{(i)}$, hay error de reproyección



Est

$\hat{\mathbf{x}}^{(i)}$

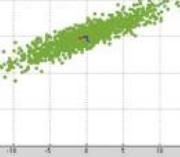
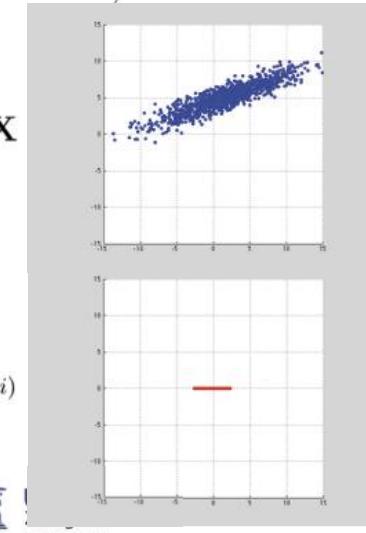
29

Ejemplo

$U_k =$

$$\begin{bmatrix} 0.3476 \\ -0.9376 \end{bmatrix}$$

- Si $k < n$, entonces $\mathbf{x}^{(i)} \neq \hat{\mathbf{x}}^{(i)}$, hay error de reproyección



Est

$\hat{\mathbf{x}}^{(i)}$

30

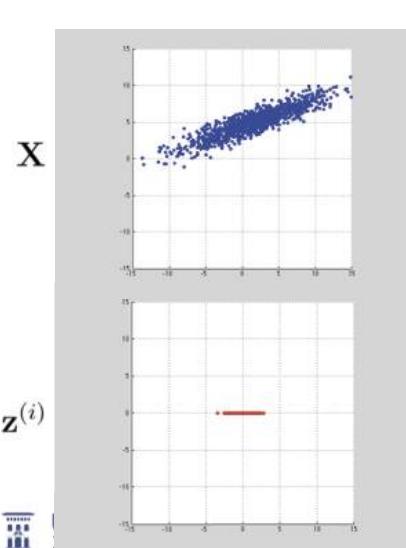
Ejemplo

$U_k =$

$$\begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$>> U_k * U_k'$

$$\text{ans} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$$

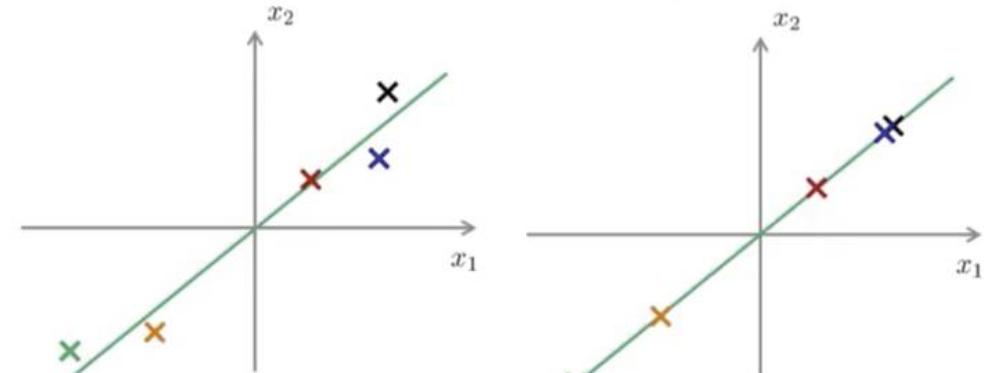


Est

$\hat{\mathbf{x}}^{(i)}$

31

Recuperación de la dimensión



Solución Óptima

- Sea \mathbf{X} la matriz de muestras estandarizadas:

$$\mathbf{X} = \begin{bmatrix} | & | & | & | \\ x^{(1)} & x^{(2)} & \dots & x^{(m)} \\ | & | & | & | \end{bmatrix}$$

- dimensiones de \mathbf{X} ?
- Sea $\Sigma = \frac{1}{m-1}\mathbf{X}\mathbf{X}^t$ la covarianza muestral de \mathbf{X}
- dimensiones de Σ ?
- Σ es simétrica positiva definida

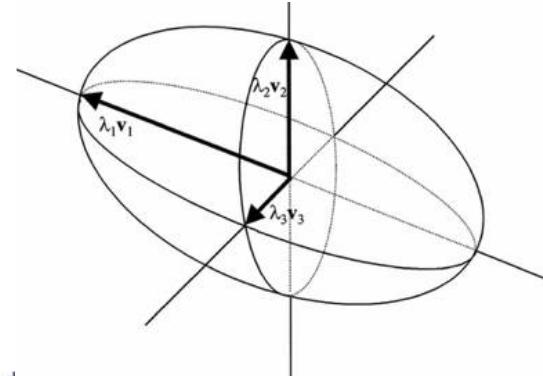
Solución Óptima

- Sea \mathbf{U} la matriz formada por los vectores propios de \mathbf{X}
 - Sea Λ la matriz formada por los valores propios de \mathbf{X}
- $$[\mathbf{U}, \Lambda] = \text{eig}(\Sigma);$$
- Ordenar los vectores propios de \mathbf{U} según los valores propios en Λ
 - Escoger el valor de k
 - $\mathbf{Z} = \mathbf{U}_k^T \mathbf{X}$

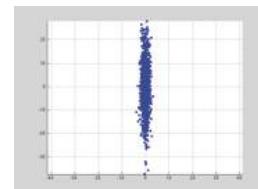
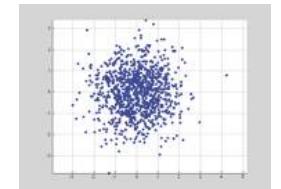
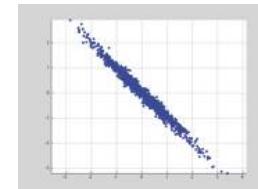
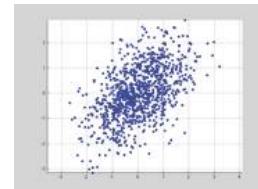
Descomposición en vectores y valores propios

$$\Sigma = \mathbf{U}\Lambda\mathbf{U}^t$$

$$= [\mathbf{u}^{(1)} \ \mathbf{u}^{(2)} \ \dots \ \mathbf{u}^{(n)}] \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_n \end{bmatrix} \begin{bmatrix} \mathbf{u}^{(1)t} \\ \mathbf{u}^{(2)t} \\ \vdots \\ \mathbf{u}^{(n)t} \end{bmatrix}$$



Ejercicio



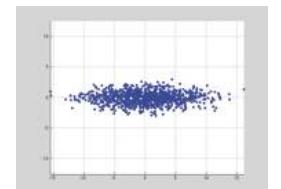
$$\mathbf{u}^{(1)} = (1, 0)^t$$

$$\mathbf{u}^{(1)} = (0, 1)^t$$

$$\mathbf{u}^{(1)} = (0.7071, 0.7071)^t$$

$$\mathbf{u}^{(1)} = (-0.7071, 0.7071)^t$$

$$\mathbf{u}^{(1)} = (1, 0)^t$$



¿Cómo escoger el valor de k ?

- Error medio de reproyección:

$$\frac{1}{m} \sum_{i=1}^m \|\mathbf{x}^{(i)} - \hat{\mathbf{x}}^{(i)}\|^2$$

- Variación media de los datos:

$$\frac{1}{m} \sum_{i=1}^m \|\mathbf{x}^{(i)}\|^2$$

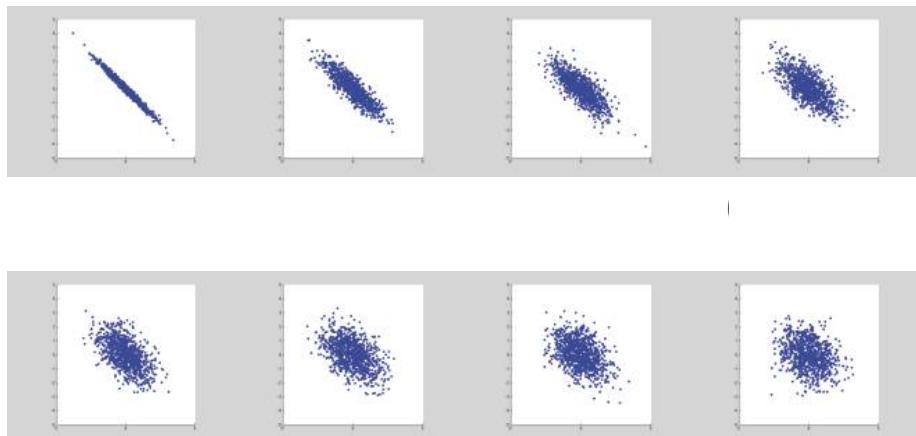
- Escoger k para que sea el menor valor tq:

$$\frac{\frac{1}{m} \sum_{i=1}^m \|\mathbf{x}^{(i)} - \hat{\mathbf{x}}^{(i)}\|^2}{\frac{1}{m} \sum_{i=1}^m \|\mathbf{x}^{(i)}\|^2} < 0.01 (1\%)$$

- Se mantiene el 99% de la varianza

 **Costoso, es necesario iterar el PCA variando k**

$n = 2$; ¿es suficiente con $k = 1$?



Solución más eficiente:

- Sea Λ la matriz de valores propios de Σ :

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_n \end{bmatrix}$$

- Escoger k para que sea el menor valor tq:

$$1 - \frac{\sum_{i=1}^k \lambda_i}{\sum_{i=1}^n \lambda_i} < 0.01$$

- Alternativamente:

 **Uni Zaragoza** $\frac{\sum_{i=1}^k \lambda_i}{\sum_{i=1}^n \lambda_i} > 0.99$

38

Descomposición en Valores Singulares (Singular Value Decomposition SVD)

- Sea M una matriz de tamaño $m \times n$. M es factorizable de la siguiente manera:

$$M_{m \times n} = \mathbf{U}_{m \times m} \boldsymbol{\Sigma}_{m \times n} \mathbf{V}_{n \times n}^T$$

- $\mathbf{U}_{m \times m} = \left(\begin{array}{c|c|c|c} u^{(1)} & u^{(2)} & \cdots & u^{(m)} \\ \hline | & | & & | \end{array} \right)$ ortogonal, genera el espacio de las filas de M .

- $\boldsymbol{\Sigma} = \begin{pmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_n \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & 0 & 0 \end{pmatrix}$ diagonal no negativa, contiene los valores singulares λ_i .

- $\mathbf{V}^T = \left(\begin{array}{c|c|c|c} v^{(1)T} & - & - & - \\ - & v^{(2)T} & - & - \\ - & - & \ddots & - \\ - & - & v^{(n)T} & - \end{array} \right)$ ortogonal, genera el espacio de las columnas de M .

¿Qué hace SVD?

- Descompone una matriz en la suma de productos de vectores

$$M = \lambda_1 \begin{pmatrix} u^{(1)} \\ | \\ u^{(n)} \end{pmatrix} (-v^{(1)^T} -) + \dots + \lambda_n \begin{pmatrix} u^{(1)} \\ | \\ u^{(n)} \end{pmatrix} (-v^{(n)^T} -)$$

■ Compresión:

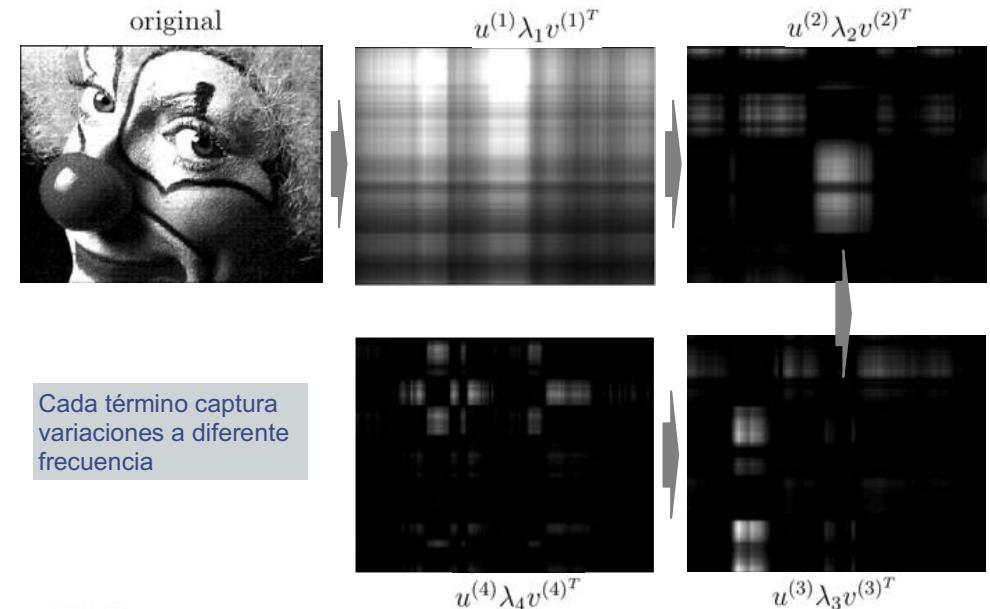
- guardamos los vectores en lugar de las matrices:

$\mathbf{U}_{m \times m} \Sigma_{m \times n} \mathbf{V}_{n \times n}^T$ tiene componentes

- Si consideramos los primeros k elementos:

$\mathbf{U}_{m \times k} \Sigma_{k \times k} \mathbf{V}_{k \times n}^T$ tiene componentes

Ejemplo: compresión de imágenes
[U, S, V] = svd(image);

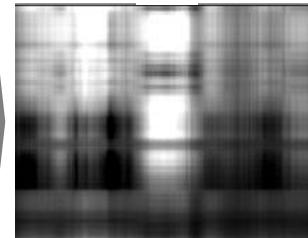


Sumas Parciales

original



$1 + 2$



$1 + 2 + 3 + 4 + 5$



$1 + \dots + 200$



$1 + \dots + 20$



$1 + \dots + 10$

Diferencia con imagen original

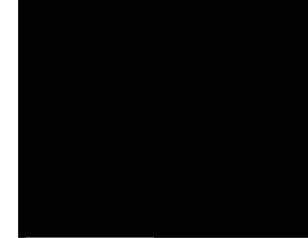
original



original - $1 + 2$



original - $1 + 2 + 3 + 4 + 5$



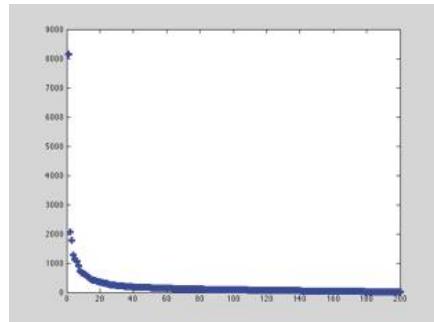
original - $1 + \dots + 200$



original - $1 + \dots + 10$

Ratio de Compresión

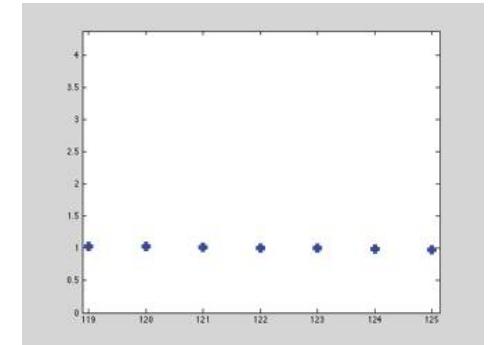
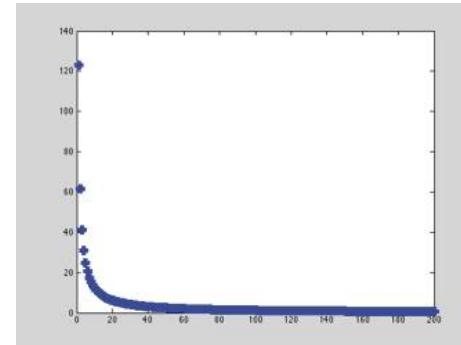
- Valores singulares: `plot(diag(S));`



Ratio de Compresión

- Ratio de compresión:

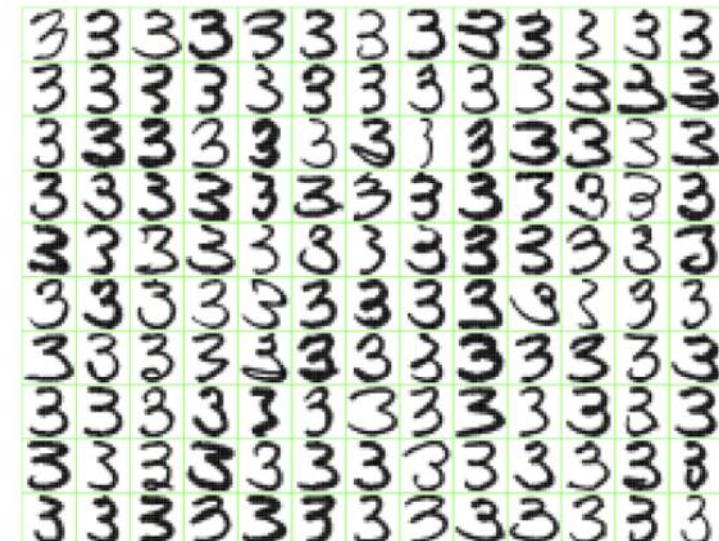
```
■ plot( (m * n) ./ ((1:1:n)*(m+n+1)));
```



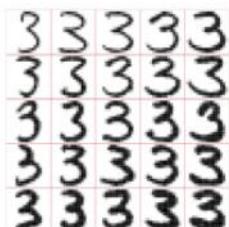
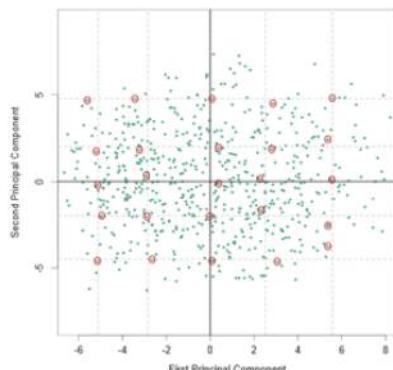
PCA para aprendizaje supervisado

- Dados los datos de entrenamiento:
 $(\mathbf{x}^{(1)}, \mathbf{y}^{(1)}), (\mathbf{x}^{(2)}, \mathbf{y}^{(2)}), \dots, (\mathbf{x}^{(m)}, \mathbf{y}^{(m)})$
- Extraemos los datos no etiquetados y aplicamos PCA:
$$\begin{array}{c} \mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(m)} \\ \text{PCA} \\ \downarrow \\ \mathbf{z}^{(1)}, \mathbf{z}^{(2)}, \dots, \mathbf{z}^{(m)} \end{array}$$
- Entrenamos con:
 $(\mathbf{z}^{(1)}, \mathbf{y}^{(1)}), (\mathbf{z}^{(2)}, \mathbf{y}^{(2)}), \dots, (\mathbf{z}^{(m)}, \mathbf{y}^{(m)})$

PCA para aprendizaje supervisado Ejemplo: MNIST



$$\begin{aligned}\hat{f}(\lambda) &= \bar{x} + \lambda_1 v_1 + \lambda_2 v_2 \\ &= \text{3} + \lambda_1 \cdot \text{3} + \lambda_2 \cdot \text{3}.\end{aligned}$$



- Compresión:

- Reducir el espacio requerido para almacenar datos, pasando de n atributos o $m * n$ datos, a $k < n$ atributos o $m * k$ datos (potencialmente $k \ll n$).

- Visualización:

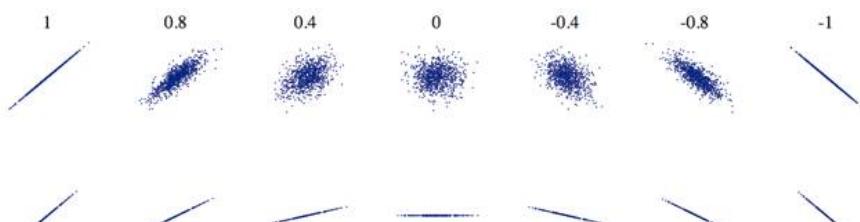
- $k = 2$, $k = 3$, ¿ $k = 4$?

- Aumento de eficiencia en problemas de aprendizaje supervisado:

- Utilizar $\mathbf{z}^{(i)}$ en vez de $\mathbf{x}^{(i)}$

PCA y no linealidades

$$\rho_{ij} = \frac{\sum_{k=1}^m (x_i^{(k)} - \bar{x}_i)(x_j^{(k)} - \bar{x}_j)}{\sqrt{\sum_{k=1}^m (x_i^{(k)} - \bar{x}_i)^2} \sqrt{\sum_{k=1}^m (x_j^{(k)} - \bar{x}_j)^2}}$$



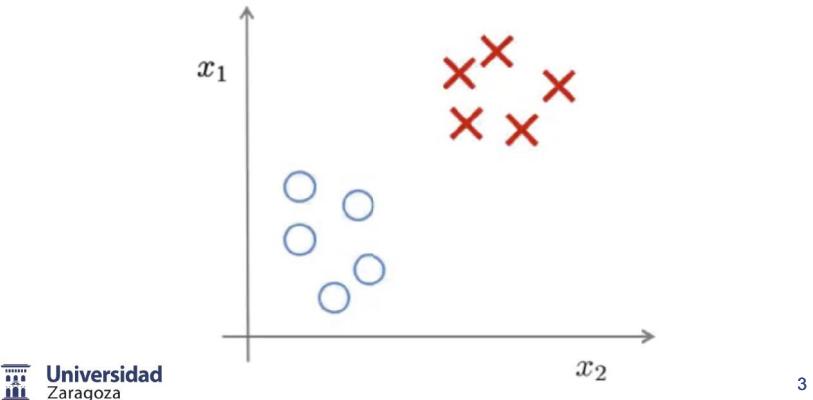
Tema 7. Aprendizaje no supervisado Agrupamiento (Clustering)

Índice

- Introducción
- Agrupamiento particional
 - Algoritmo K-Medias
 - Algoritmo E-M
- Agrupamiento jerárquico
 - Aglomerativo y de división
- Bibliografía y Lecturas recomendadas
 - Bishop (Capítulo 9); K.P. Murphy (Capítulo 25);
Hastie, Tibshirani, Friedman (Capítulo 14)
- Video lectures: Andrew Ng, Sebastian Thrun

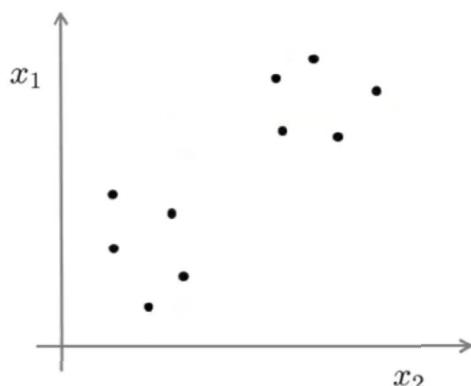
Introducción: Aprendizaje supervisado

- Entrada: conjunto de datos $D = \{\mathbf{x}^{(i)}, \mathbf{y}^{(i)}\}_{i=1}^m$
- Clasificación: $\mathbf{x}^{(i)} \in \mathbb{R}^n, \mathbf{y}^{(i)} \in C = \{1, 2, \dots, K\}$



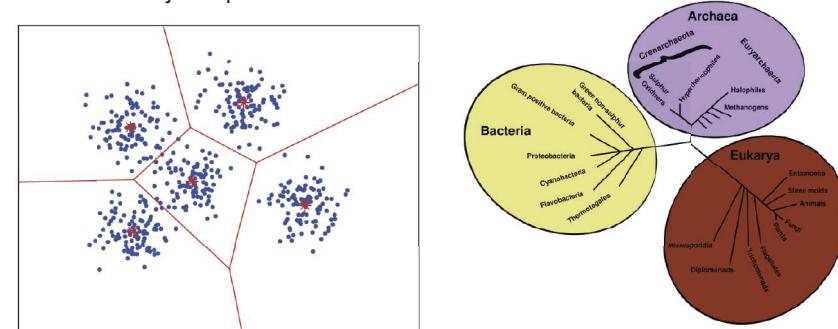
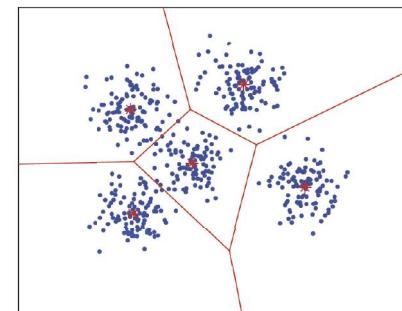
Aprendizaje no supervisado

- Entrada: conjunto de datos $D = \{\mathbf{x}^{(i)}\}_{i=1}^m, \mathbf{x}^{(i)} \in \mathbb{R}^n$
- Objetivo: buscar patrones o estructura en los datos



Tipos de agrupamiento

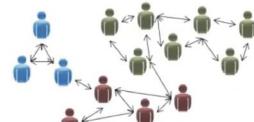
- Entradas:
 - Semejanza
 - Características (métrica de semejanza)
- Resultado:
 - Partición del espacio en grupos o *clusters*
 - Árbol o jerarquía de *clusters*



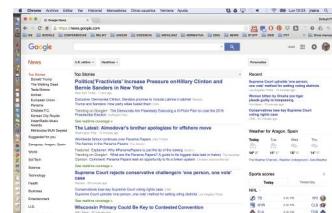
Ejemplos



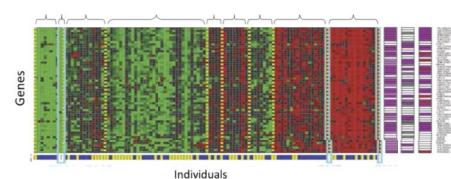
Segmentos de mercado



Análisis de redes sociales



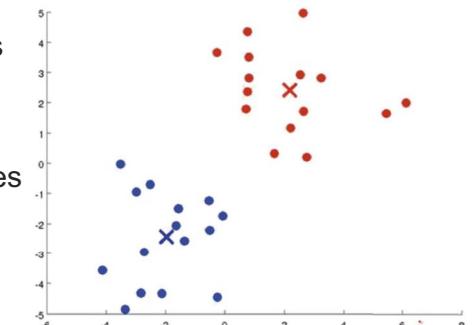
Google News



Análisis Genético

K – medias (K – means)

1. Inicializar k semillas aleatorias (centroides)
2. Etiquetar muestras más cercanas a cada centroide
3. Recalcular los centroides
4. ¿Han cambiado las etiquetas? Paso 2.
5. Fin



K – medias (K – means)

- Entradas:
 - K (número de grupos (clusters))
 - Datos: $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(m)}\}, \mathbf{x}^{(i)} \in \mathbb{R}^n$
- Inicializar K centroides $\{\mu_1, \mu_2, \dots, \mu_K\}, \mu_i \in \mathbb{R}^n$
- Repetir:

Para $i = 1$ hasta m

$$c^{(i)} := k : \min_k \|\mathbf{x}^{(i)} - \mu_k\|^2$$

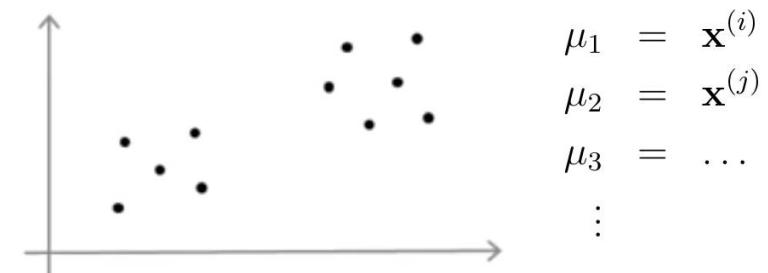
Para $k = 1$ hasta K

$$\mu_k := \text{promedio de } \{\mathbf{x}^{(i)} : c^{(i)} = k\}$$

- Hasta que $c^{(i)}$ no cambie para ningún $\mathbf{x}^{(i)}$

Inicialización de los centroides

$$\mu_1, \dots, \mu_K$$



Hay muchas estrategias para intentar una buena dispersión

Ejercicio

- Si los valores iniciales de μ_1, \dots, μ_K se asignan aleatoriamente, ¿es posible que alguno quede al final sin muestras asociadas?
- Si los valores iniciales de μ_1, \dots, μ_K se asignan según este criterio:

$$\mu_1 = \mathbf{x}^{(i)}$$

$$\mu_2 = \mathbf{x}^{(j)}$$

$$\mu_3 = \dots$$

:

¿es posible que alguno quede al final sin muestras asociadas?

Convergencia: función de distorsión

$$J(c^{(1)}, \dots, c^{(m)}, \mu_1, \dots, \mu_K) = \frac{1}{m} \sum_{i=1}^m \|x^{(i)} - \mu_{c^{(i)}}\|^2$$

$$\min_{\substack{c^{(1)}, \dots, c^{(m)}, \\ \mu_1, \dots, \mu_K}} J(c^{(1)}, \dots, c^{(m)}, \mu_1, \dots, \mu_K)$$

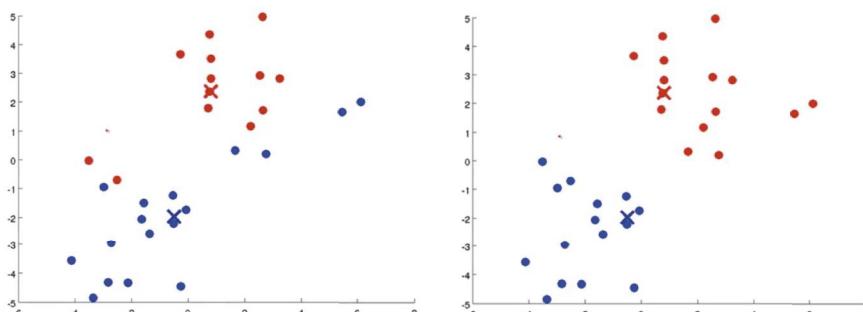
- K-medias los resuelve en dos pasos:

- Paso 1: Dados μ_1, \dots, μ_K , minimizar con respecto a $c^{(1)}, \dots, c^{(m)}$
- Paso 2: Dados $c^{(1)}, \dots, c^{(m)}$, minimizar con respecto a μ_1, \dots, μ_K

K-medias hace un descenso coordinado en J

Al recalcular las asignaciones

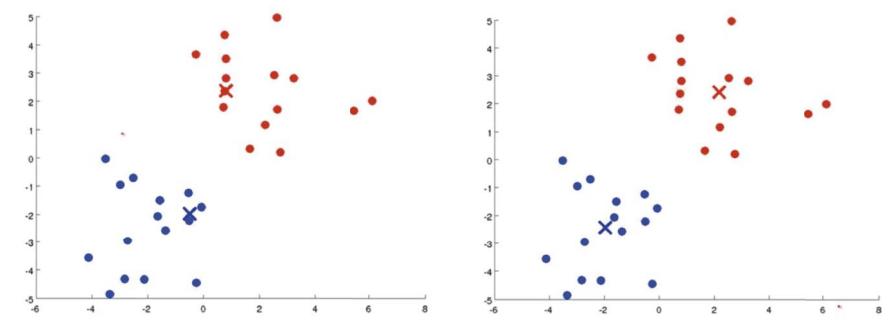
$$c^{(1)}, \dots, c^{(m)}$$



J se reduce en cada asignación

Al recalcular los centroides

$$\mu_1, \dots, \mu_K$$



J se reduce en cada recálculo del centroide

Ejercicio

- Demuestra para el caso monodimensional que la media o centroide es el lugar geométrico cuya distancia cuadrática a un conjunto de puntos es mínima.

$$\min_{\hat{x}} \sum_m^{i=1} (x^{(i)} - \hat{x})^2$$

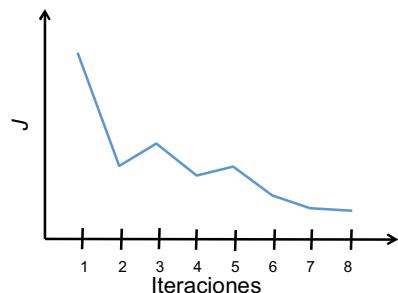
Ejercicio

- ¿Es posible utilizar el siguiente criterio de minimización para la reasignación de muestras a clusters?

$$c^{(i)} := k : \min_k \| \mathbf{x}^{(i)} - \mu_k \|$$

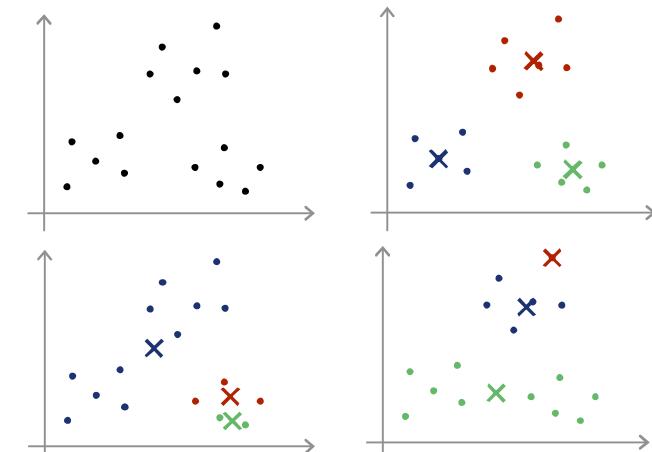
Ejercicio

- Supón que has implementado k-medias y para comprobar que su ejecución es correcta graficas $J(c^{(1)}, c^{(2)}, \dots, c^{(m)}, \mu_1, \mu_2, \dots, \mu_K)$ en función del número de iteraciones



- El algoritmo funciona correctamente
- Funciona, pero K es demasiado grande
- Debe haber un error en el código

Convergencia: ¿puede haber mínimos locales?



No hay convergencia global garantizada

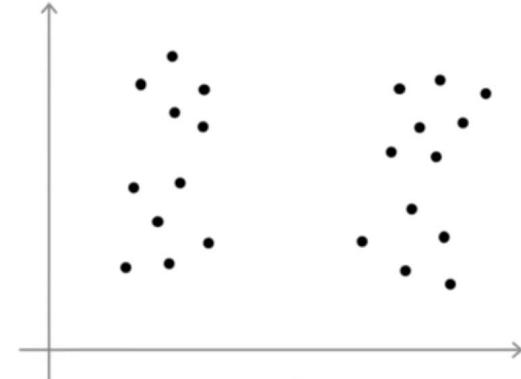
Mejora hacia la convergencia global

Repetir N veces:

- Inicializar K centroides $\{\mu_1, \mu_2, \dots, \mu_K\}$
- Ejecutar K - medias, obtener $\{c^{(1)}, c^{(2)}, \dots, c^{(m)}, \mu_1, \mu_2, \dots, \mu_K\}$
- Calcular $J(c^{(1)}, c^{(2)}, \dots, c^{(m)}, \mu_1, \mu_2, \dots, \mu_K)$

Escoger agrupamiento con $\min J(c^{(1)}, c^{(2)}, \dots, c^{(m)}, \mu_1, \mu_2, \dots, \mu_K)$

Valor de K: (número de grupos)

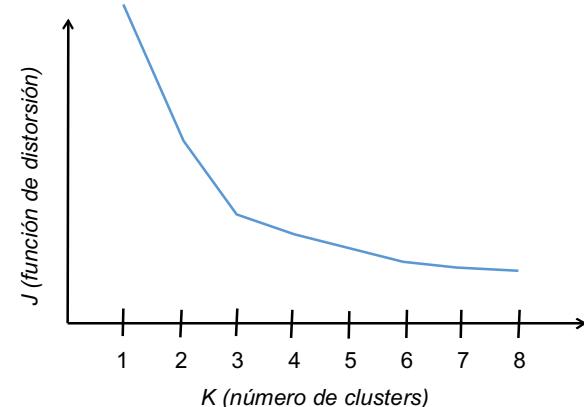


¿Cuántos clusters hay?

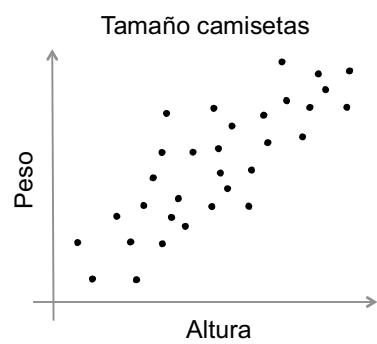
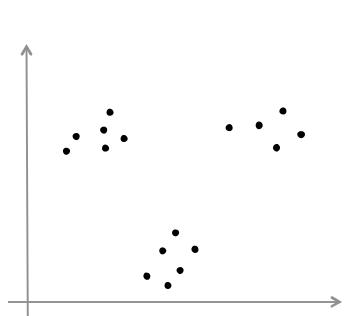
Ejercicio

- Un buen criterio para escoger el valor de K es:
 - $K < n$
 - $K = 1$
 - $K = m$
 - $K < m$

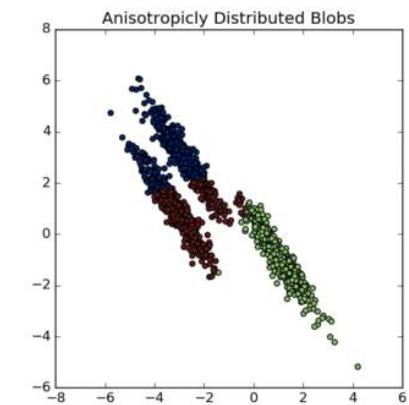
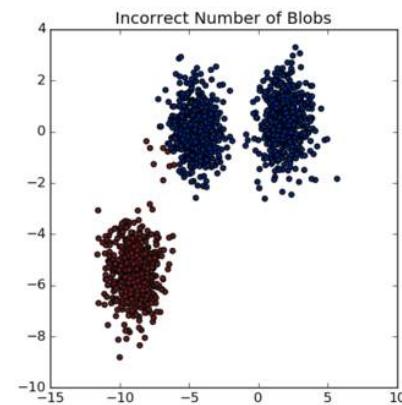
Valor de K: método del “codo”



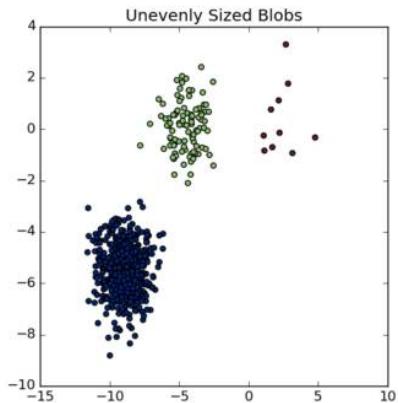
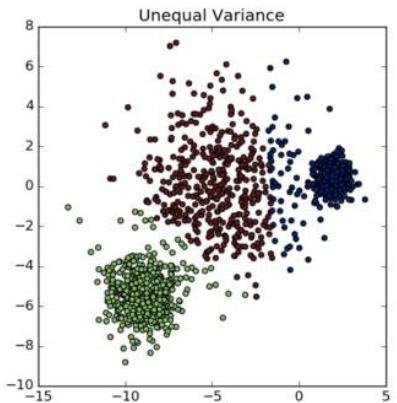
K-medias y separabilidad de clases



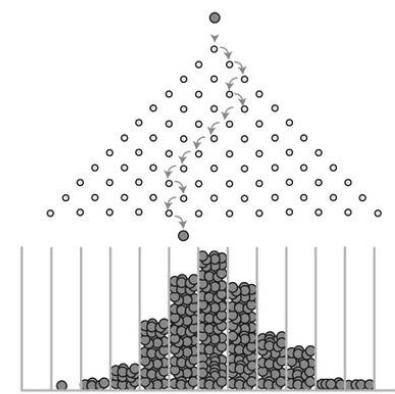
Limitaciones, casos difíciles



Limitaciones, casos difíciles

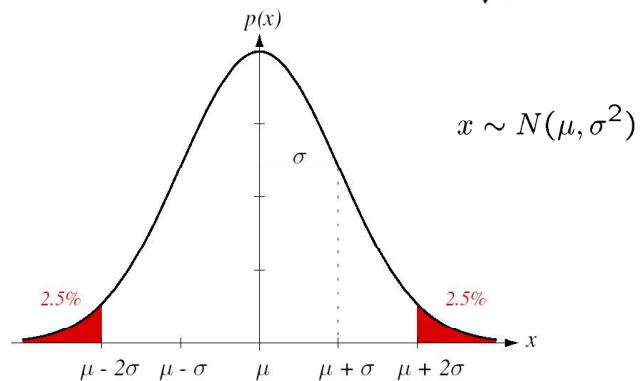


Galton Board



Procesos Gaussianos

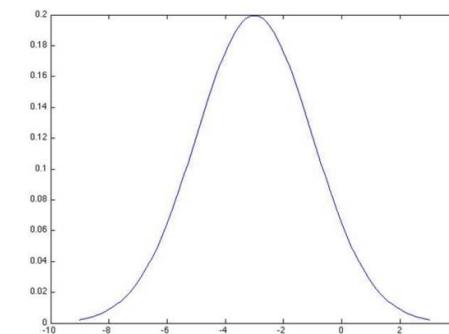
$$p(x) = f(x|\mu, \sigma^2) = f(x|\theta) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$



$$Pr \{ |x - \mu| \leq 2\sigma \} \simeq 0.95$$



Ejercicio



$$p(x) = \frac{1}{\sqrt{2\pi}\times 2} \exp\left(-\frac{(x-3)^2}{2\times 4}\right)$$

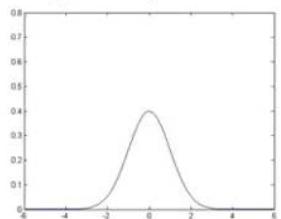
$$p(x) = \frac{1}{\sqrt{2\pi}\times 4} \exp\left(-\frac{(x-3)^2}{2\times 2}\right)$$

$$p(x) = \frac{1}{\sqrt{2\pi}\times 2} \exp\left(-\frac{(x+3)^2}{2\times 4}\right)$$

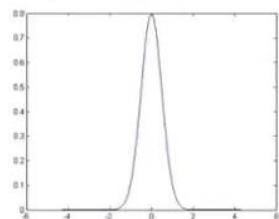
$$p(x) = \frac{1}{\sqrt{2\pi}\times 4} \exp\left(-\frac{(x+3)^2}{2\times 2}\right)$$

Ejercicio

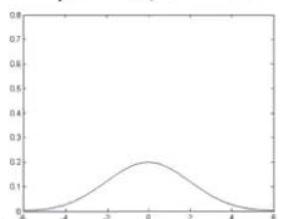
$$\mu = , \sigma =$$



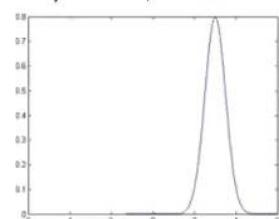
$$\mu = , \sigma =$$



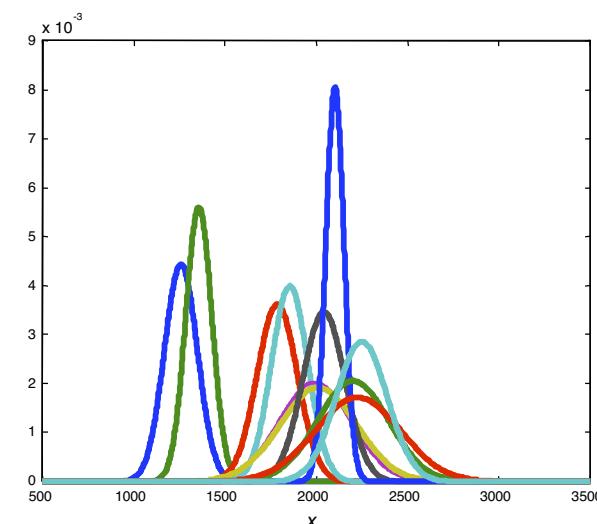
$$\mu = , \sigma =$$



$$\mu = , \sigma =$$



Ejercicio



Estimación de Máxima Verosimilitud (EMV) de los parámetros de una Gaussiana

- Dado un conjunto de datos $\{x^{(1)}, \dots, x^{(m)}\}$
- Supongamos que provienen de una distribución $\mathcal{N}(\mu, \sigma^2)$
- Su función conjunta de verosimilitud será:

$$\begin{aligned}\mathcal{L}(\theta; x^{(1)}, \dots, x^{(m)}) &= f(x^{(1)}, \dots, x^{(m)} | \theta) \\ &= f(x^{(1)} | \theta) \times \dots \times f(x^{(m)} | \theta) \\ &= \prod_{i=1}^m f(x^{(i)} | \theta) \\ &= \prod_{i=1}^m \mathcal{N}(x^{(i)}, \mu, \sigma^2)\end{aligned}$$

- El EMV $\hat{\theta}$ será el valor que maximize \mathcal{L} , o equivalentemente que maximize $\log \mathcal{L}$:

$$\begin{aligned}\frac{\partial \log \mathcal{L}}{\partial \mu} &= 0 \\ \frac{\partial \log \mathcal{L}}{\partial \sigma} &= 0\end{aligned}$$



La media muestral es un estimador insesgado

$$E[\hat{\mu}]$$

Estimación de Máxima Verosimilitud (EMV) de los parámetros de una Gaussiana

$$\frac{\partial \log \mathcal{L}}{\partial \mu} = 0 \rightarrow$$

$$\frac{\partial \log \mathcal{L}}{\partial \sigma} = 0 \rightarrow$$



La varianza muestra es un estimador sesgado

$$\begin{aligned}E[\hat{\sigma}^2] &= E\left[\frac{1}{m} \sum_{i=1}^m (x^{(i)} - \hat{\mu})^2\right] \\ &= \frac{1}{m} E\left[\sum_{i=1}^m (x^{(i)} - \hat{\mu})^2\right] \\ &\quad \vdots \\ &= \sigma^2 - E[(\hat{\mu} - \mu)^2] \\ &< \sigma^2\end{aligned}$$



Estimación insesgada de la varianza

- Estimador insesgado de la varianza:

$$\hat{\sigma}^2 = \frac{1}{m-1} \sum_{i=1}^m (x^{(i)} - \hat{\mu})^2$$

- No está definido para $m=1$

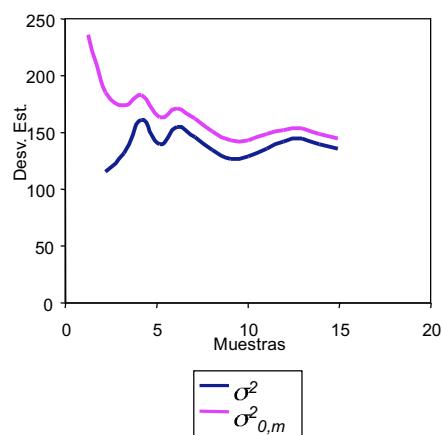
- Si m es pequeño, tiende a ser optimista (subestima la varianza)

$$\hat{\sigma}_m^2 = \frac{\sigma_{0,m}^2}{m} + \frac{m-1}{m} \hat{\sigma}^2$$

- $\sigma_{0,m}^2$: estimación a priori de la varianza (p. e. el cuadrado del 1% del valor del atributo)

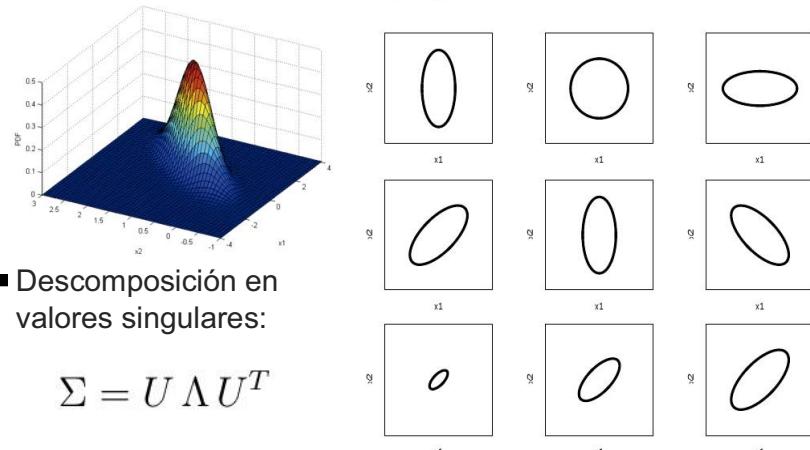


- Para m grande, tiende a la varianza insesgada



Distribución Gaussiana Multivariable

$$f(x|\mu, \Sigma) = \frac{1}{|\Sigma|^{1/2}(2\pi)^{n/2}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)}$$



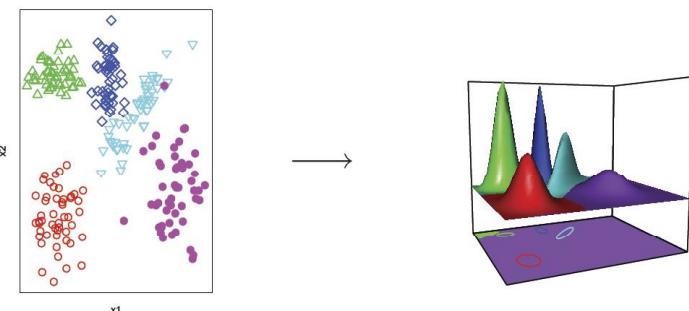
- Descomposición en valores singulares:

$$\Sigma = U \Lambda U^T$$



35

Suma o mezcla de Gaussianas (Gaussian Mixture Models)

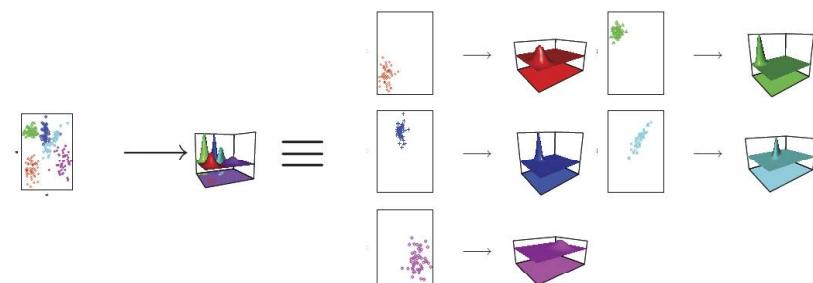


$$p(x) = \sum_{i=1}^K p(C=i) p(x|C=i)$$

$$0 \leq \pi_i \leq 1, \sum_{i=1}^K \pi_i = 1$$

$$= \sum_{i=1}^K \pi_i f(x|\mu_i, \Sigma_i)$$

Ejercicio



- ¿Conocidas las etiquetas, cómo se estima una mezcla de Gaussianas?

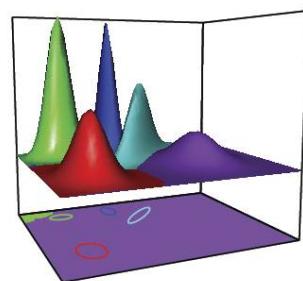
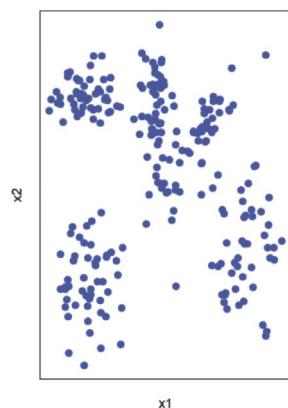


36



37

Agrupamiento con Mezcla de Gaussianas



Dados $\{x^{(1)}, \dots, x^{(m)}\}$, K , estimar π_i, μ_i, Σ_i

Expectation – Maximization (EM)

- Paso E: Supuestos conocidos los parámetros π_i, μ_i, Σ_i , calculamos la probabilidad de que la muestra j pertenezca a la clase i :

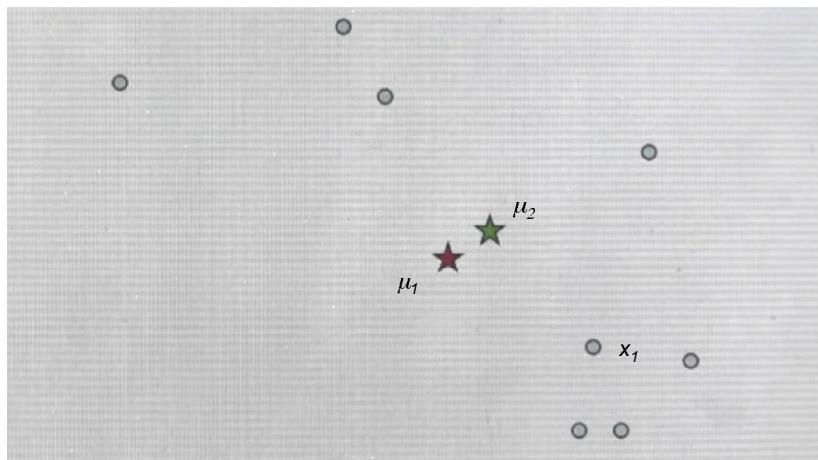
$$\begin{aligned} e_{ij} &= \pi_i \mathcal{N}(x^{(j)}; \mu_i, \Sigma_i) \\ &= \pi_i \frac{1}{|\Sigma_i|^{1/2} (2\pi)^{n/2}} e^{-\frac{1}{2}(x^{(j)} - \mu_i)^T \Sigma_i^{-1} (x^{(j)} - \mu_i)} \end{aligned}$$

- Paso M: Recalculamos los parámetros π_i, μ_i, Σ_i :

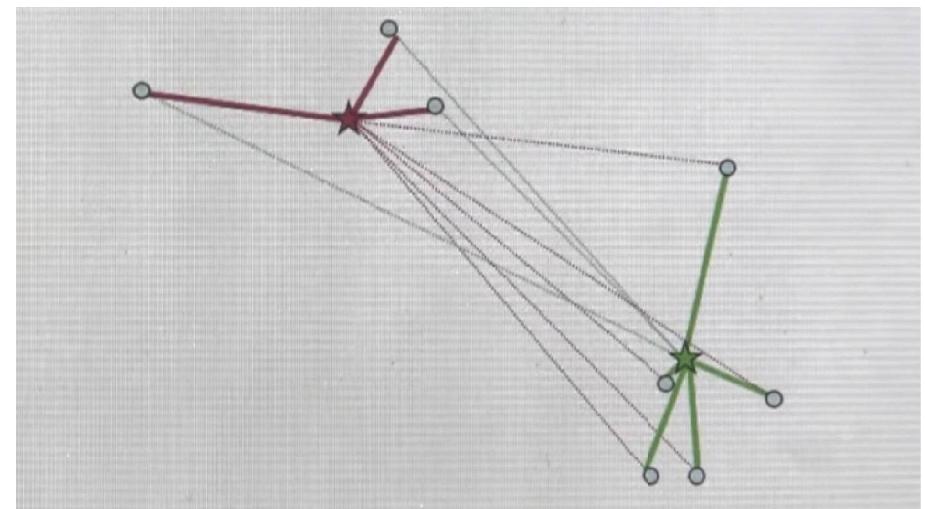
$$\begin{aligned} \pi_i &= \frac{1}{m} \sum_{j=1}^m e_{ij} \\ \mu_i &= \sum_{j=1}^m e_{ij} x^{(j)} / \sum_{j=1}^m e_{ij} \\ \Sigma_i &= \sum_{j=1}^m e_{ij} (x^{(j)} - \mu_i)^T (x^{(j)} - \mu_i) / \sum_{j=1}^m e_{ij} \quad 39 \end{aligned}$$

Ejercicio

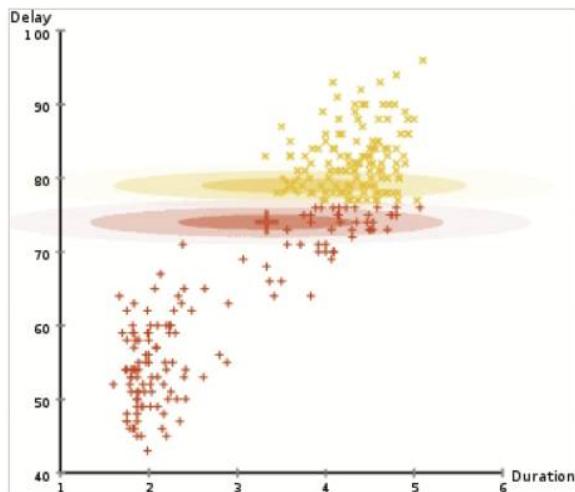
¿A qué clúster pertenece x_1 ?



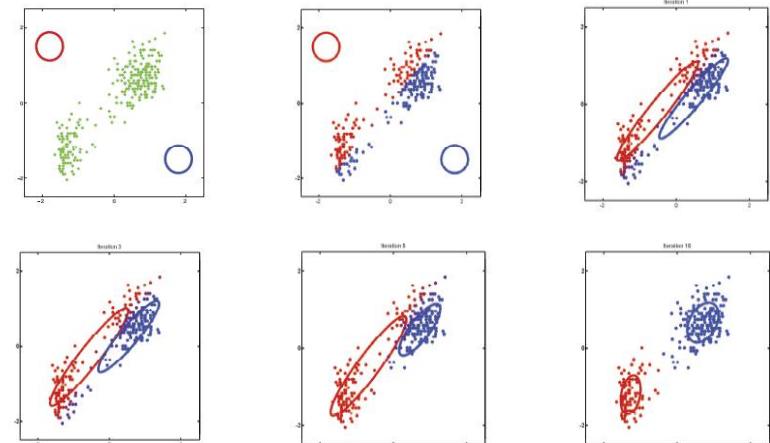
Ejemplo



Expectation - Maximization



Otro ejemplo de EM con GMM y 2 clústers



Practicalidades

- Inicialización
 - Aleatoria
 - Para evitar mínimos locales, varias iteraciones de EM

- Número de clústers: minimizar

$$-\sum_i \sum_j \log f(x^{(j)}, \mu_i, \Sigma_i) + \text{costo} \cdot K$$

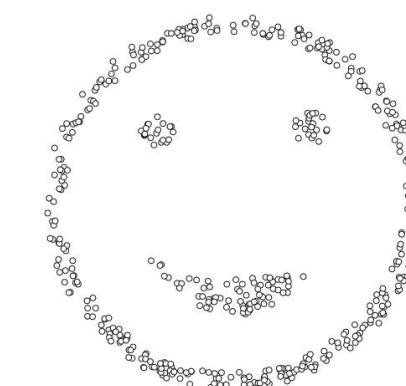
■ MAP

- ¿Se puede poner una prior $p(\theta)$ sobre los parámetros θ ?
- ¿Se puede poner una prior $p(\pi)$ sobre los parámetros π ?

- EM se puede aplicar a cualquier problema donde faltan datos o en problemas donde variables auxiliares hagan más simple la verosimilitud (likelihood)

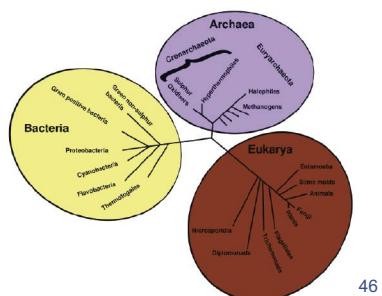
Ejercicio

- ¿Podrá ir bien K-Medias, o EM en este caso?



Agrupamiento jerárquico

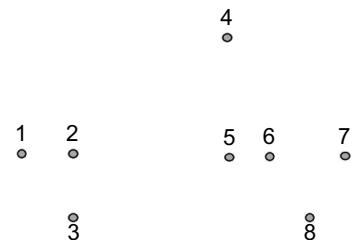
- Los datos tienen en ocasiones una estructura jerárquica
 - Interesa el agrupamiento y la jerarquía
- Dos tipos
 - Aglomerativo (bottom-up)
 - Divisorio (top-down)
- Son algoritmos heurísticos (no tienen una función de optimización bien definida)



46

Single Linkage Clustering

- Inicialmente, cada muestra es un clúster
- Distancia entre dos clústers: distancia entre muestras más cercanas
- Fusionar los dos clústers más cercanos
- Repetir m-K veces hasta obtener K clústers



47

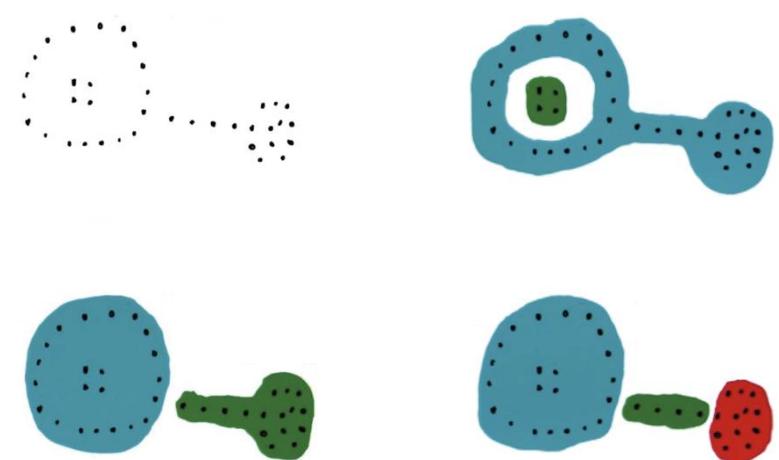
Propiedades

- Determinista, no depende de inicializaciones aleatorias
- En su forma básica, equivalente a MST, MSF (Kruskal)
- Complejidad computacional para m muestras, K clústers:

$$\begin{aligned}\mathcal{O}(m^K) \\ \mathcal{O}(2^{n-K}) \\ \mathcal{O}(m^3) \\ \mathcal{O}(m + K)\end{aligned}$$

48

Ejercicio

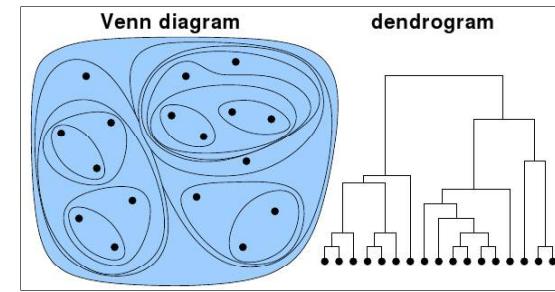


49

Agrupamiento Jerárquico Aglomerativo

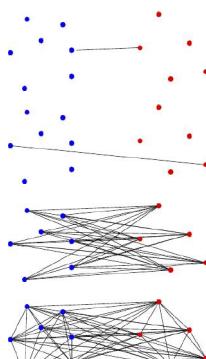
1. Inicializar un cluster por punto: $\forall i, C_i \leftarrow i$
 2. Inicializar cluster para fusión: $S \leftarrow 1..N$
 3. Inicializar matriz de distancias
 - repetir
 4. Coger los dos clusters más parecidos: $(j, k) \leftarrow \arg \min_{j, k \in S} d_{j, k}$
 5. Fusionar C_j y C_k : $C_l \leftarrow C_j \cup C_k$
 6. Eliminar j y k de S : $S \leftarrow S \setminus \{j, k\}$
 7. Añadir C_l a S
 8. Actualizar matriz de distancias $d(i, l)$
- hasta que todo este fusionado

Clustering jerárquico aglomerativo



Distancia entre clusters

- Dadas las distancias entre puntos $d(\mathbf{x}_i, \mathbf{x}_j)$
- Dados los clusters C_k, C_l
- ¿Cómo definimos la distancia $d(C_1, C_2)$?
 - Single linkage
 - $$d(c_1, c_2) = \min_{\mathbf{x}_m \in c_1, \mathbf{x}_n \in c_2} d(\mathbf{x}_m, \mathbf{x}_n)$$
 - Complete linkage
 - $$d(c_1, c_2) = \max_{\mathbf{x}_m \in c_1, \mathbf{x}_n \in c_2} d(\mathbf{x}_m, \mathbf{x}_n)$$
 - Average linkage
 - $$d(c_1, c_2) = \frac{1}{|c_1||c_2|} \sum_{\mathbf{x}_m \in c_1, \mathbf{x}_n \in c_2} d(\mathbf{x}_m, \mathbf{x}_n)$$
 - Group linkage
 - $$d(c_1, c_2) = \frac{1}{(|c_1|+|c_2|)^2} \sum_{\mathbf{x}_m, \mathbf{x}_n \in (c_1 \cup c_2)} d(\mathbf{x}_m, \mathbf{x}_n)$$



Medidas de semejanza/distancia

- Matriz de distancias
 - Cada elemento i, j es una medida de la distancia del elemento i al j
 - p características: $D(x_i, x_j) = \sum_{k=1}^p d_k(x_{ik}, x_{jk})$
- Medidas
 - Distancia Euclídea (norma L2): $d_k(x_{ik}, x_{jk}) = (x_{ik} - x_{jk})^2$
 - Distancia de Manhattan (norma L1): $d_k(x_{ik}, x_{jk}) = |x_{ik} - x_{jk}|$
 - Coeficiente de correlación: $\rho_t = \frac{\sum_t (x_{it} - \bar{x}_i)(x_{jt} - \bar{x}_j)}{\sqrt{\sum_t (x_{it} - \bar{x}_i)^2 \sum_t (x_{jt} - \bar{x}_j)^2}}$
 - ◆ Usado para vectores (e.g. series temporales)
 - Variables ordinales: se usa el orden
 - ◆ Calificaciones: A,B,C,D,E-> 1/5, 2/5, 3/5, 4/5, 5/5
 - Variables categóricas: se debe especificar,
 - ◆ e.g. 0 si son iguales, 1 si son distintas

Agrupamiento Jerárquico Divisivo

Li Fei-Fei (Stanford)

Image

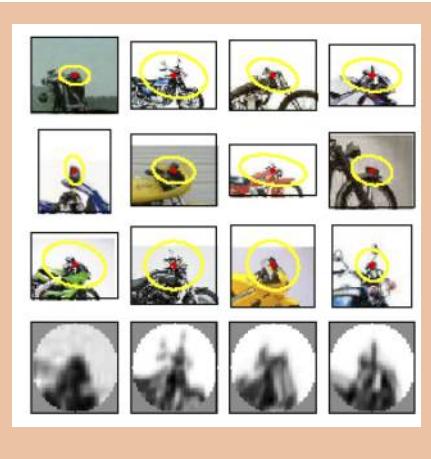
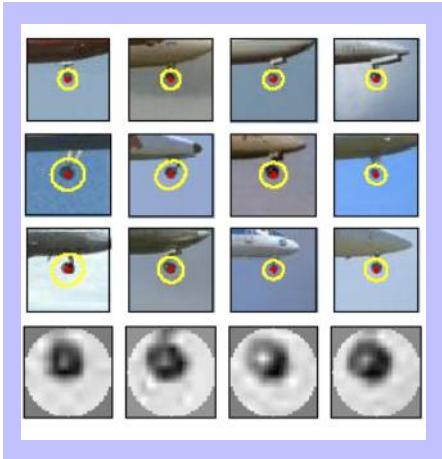


→ Bag of 'words'



Universidad
Zaragoza

Extracción de Carácterísticas (Features)



Universidad
Zaragoza

Sivic et al. 2005

Analogía con documentos

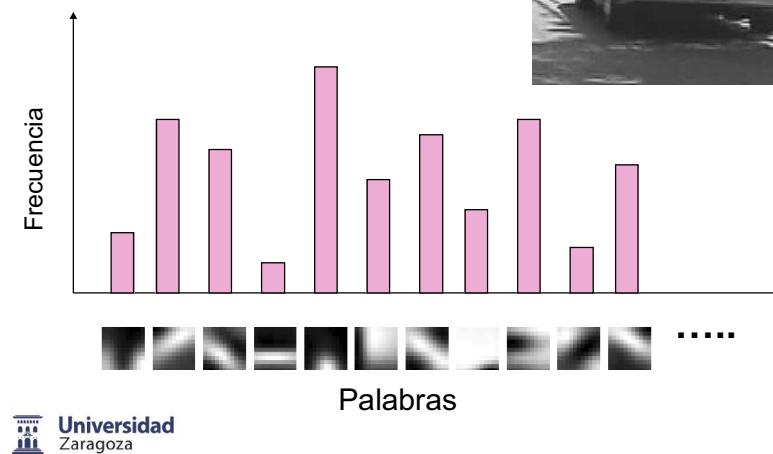
Of all the sensory impressions proceeding to the brain, the visual experiences are the dominant ones. Our perception of the world around us is based essentially on the messages that reach us through our eyes. For a long time it was believed that the retinal image was processed directly in the retinal centers of the brain. In other words, a movie screen was thought to be the image discovered by the eye. We now know that the perception of the visual world is a much more complex process. After traveling a path to the various centers of the visual cortex, Hubel and Wiesel have demonstrated that the message about the image falling on the retina undergoes a two-stage analysis in a system of nerve cells stored in columns. In this system each column has its specific function and is responsible for a specific detail in the pattern of the retinal image.

Universidad
Zaragoza

China is forecasting a trade surplus of \$90bn (£51bn) to \$100bn this year, a threefold increase on 2004's \$32bn. The Commerce Ministry said the surplus would be created by a predicted 30% growth in exports to \$750bn, compared with \$660bn. The Chinese government is annoyed that China's trading partners believe that China deliberately agrees to let the yuan rise against the dollar. The Chinese government has permitted it to trade within a narrow band, but the US wants the yuan to be allowed to fluctuate freely. However, Beijing has made it clear that it will take its time and tread carefully before allowing the yuan to rise further in value.

China, trade, surplus, commerce, exports, imports, US, yuan, bank, domestic, foreign, increase, trade, value

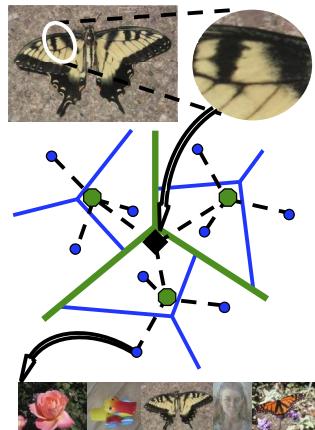
Representación con Bolsas de Palabras



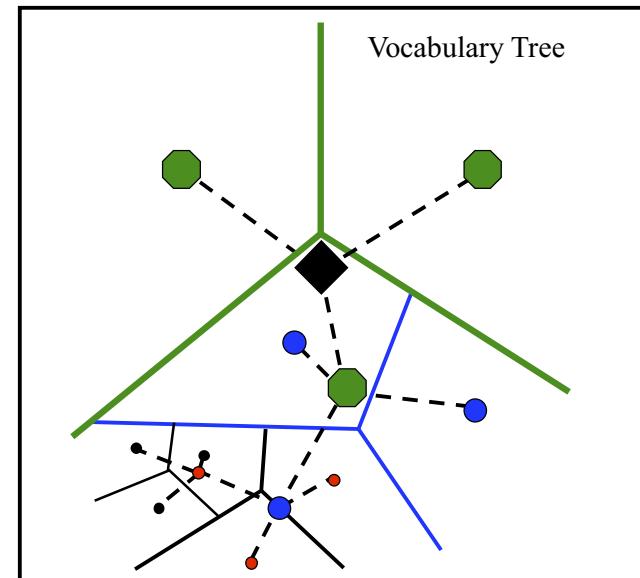
Universidad
Zaragoza

Scalable Recognition with a Vocabulary Tree

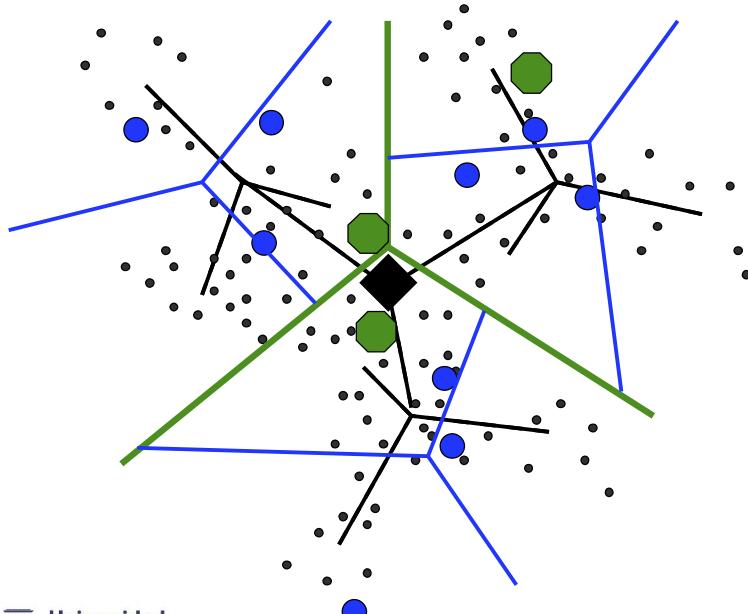
David Nistér, Henrik Stewénius



Universidad
Zaragoza



Universidad
Zaragoza



Universidad
Zaragoza

Ejemplo: búsqueda en base de datos de 5000 imágenes de flickr

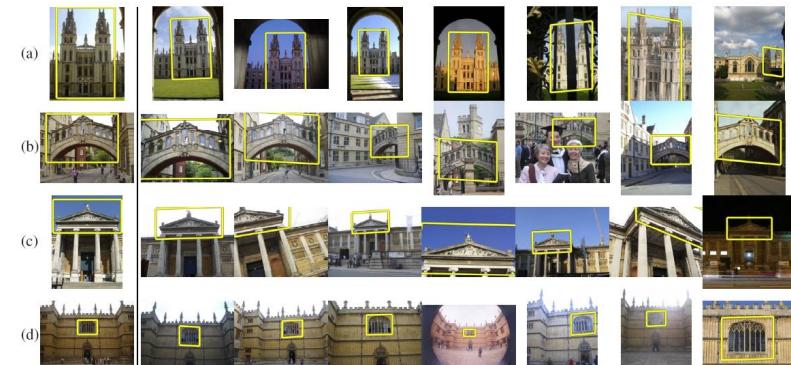


Figure 4. Examples of searching the 5K dataset for: (a) All Soul's College. (b) Bridge of sighs, Hertford College. (c) Ashmolean Museum. (d) Bodleian window. The query is shown on the left, with selected top ranked retrieved images shown to the right. All results displayed are returned before the first false positive for each query.

Philbin, J. and Chum, O. and Isard, M. and Sivic, J. and Zisserman, A.,
“Object retrieval with large vocabularies and fast spatial matching”, Proc.
CVPR pag 1575—1589, 2007

Universidad
Zaragoza

Ejemplo final: compresión de imágenes

Quantized image (64 colors, K-Means)



Tema 8. Sistemas de Recomendación

30231 - Aprendizaje Automático
Grado en Ingeniería Informática
Escuela de Ingeniería y Arquitectura

Índice

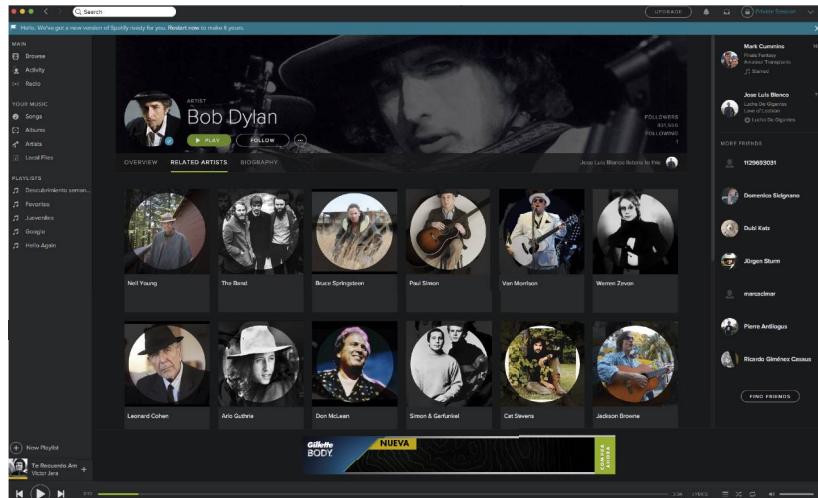
- Introducción
- (1) Filtrado basado en contenidos
- (2) Filtrado colaborativo
- (3) Vecino más próximo
- (4) Algunas consideraciones importantes

Amazon

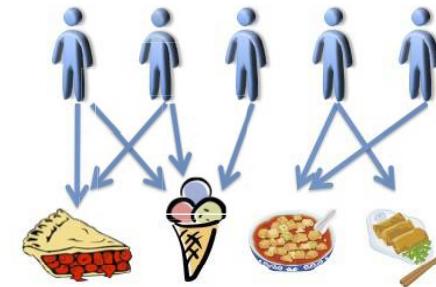


Home new store in Made in Italy

Spotify



Idea principal



- Intentan predecir los gustos/intereses de una persona dada la relación de sus gustos/intereses con el resto
- Se consideran atributos de los productos
- Se consideran parámetros de las personas
- Distinto de dar una media general

(1) Recomendadores basados en contenidos

Película	Luis	Mingo	Ana	Pedro	
Love Actually	5	5	0	0	★★★★★
Mamma mia	5	?	?	0	★★★★★
Pretty Woman	?	4	0	?	★★★★★
X-men	0	0	5	4	★★★★★
Alien vs predator	0	0	5	?	★★★★★

- n_u : número de usuarios
- n_m : número de películas
- $r(i, j) = 1$ si el usuario j ha calificado el ítem i
- $y^{(i,j)}$: calificación de j para el ítem i (definido ssi $r(i, j) = 1$)

Ejercicio

.	User 1	User 2	User 3
Movie 1	0	1	?
Movie 2	?	5	5

$$\begin{aligned} r(2, 1) &= \\ y^{(2,1)} &= \\ r(1, 2) &= \\ y^{(1,2)} &= \end{aligned}$$

(1) Recomendadores basados en contenidos

Película	Luis	Mingo	Ana	Pedro	Romance	Acción
Love Actually	5	5	0	0	0.9	0
Mamma mia	5	?	?	0	1	0.01
Pretty Woman	?	4	0	?	0.99	0
X-Men	0	0	5	4	0.1	1
Alien vs predator	0	0	5	?	0	0.9

Ejercicio

Película	Luis	Mingo	Ana	Pedro	Romance	Acción
Love Actually	5	5	0	0	0.9	0
Mamma mia	5	?	?	0	1	0.01
Pretty Woman	?	4	0	?	0.99	0
X-Men	0	0	5	4	0.1	1
Alien vs predator	0	0	5	?	0	0.9

- n : número de atributos
- $x^{(i)}$: atributos de item i
- $\theta^{(j)}$: parámetros del usuario j
- Para cada usuario j , aprender el vector $\theta^{(j)} \in \mathbb{R}^{n+1}$
- El usuario j calificará el item i como $(\theta^{(j)})^T x^{(i)}$

8



9

$$\theta^{(3)} = \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix} ? \quad \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} ? \quad \begin{bmatrix} 1 \\ 0 \\ 4 \end{bmatrix} ? \quad \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix} ?$$

Recomendadores basados en contenidos

- $r(i, j) = 1$ si el usuario j ha calificado el item i
- $y^{(i,j)}$: calificación de j para el item i (definido ssi $r(i, j) = 1$)
- $x^{(i)}$: atributos de item i
- $\theta^{(j)}$: parámetros del usuario j
- $m^{(j)}$: número de items calificados por el usuario j

- Para el usuario j :

$$\min_{\theta^{(j)}}$$

Recomendadores basados en contenidos

- Para un usuario: $\theta^{(j)}$

$$\min_{\theta^{(j)}} \frac{1}{2} \sum_{i:r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)} \right)^2 + \frac{\lambda}{2} \sum_{k=1}^n (\theta_k^{(j)})^2$$

- Para n usuarios: $\theta^{(1)}, \theta^{(2)}, \dots, \theta^{(n_u)}$

$$\min_{\theta^{(1)}, \dots, \theta^{(n_u)}} \frac{1}{2} \sum_{i:r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)} \right)^2 + \frac{\lambda}{2} \sum_{k=1}^n (\theta_k^{(j)})^2$$

Recomendadores basados en contenidos

- Función a optimizar:

$$\min_{\theta^{(1)}, \dots, \theta^{(n_u)}} \frac{1}{2} \sum_{j=1}^{n_u} \sum_{i:r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)})^2 + \frac{\lambda}{2} \sum_{j=1}^{n_u} \sum_{k=1}^n (\theta_k^{(j)})^2$$

- Actualización para descenso de gradiente:

$$\begin{aligned}\theta_k^{(j)} &:= \theta_k^{(j)} - \alpha \sum_{i:r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)}) x_k^{(i)} \quad (\text{for } k = 0) \\ \theta_k^{(j)} &:= \theta_k^{(j)} - \alpha \left(\sum_{i:r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)}) x_k^{(i)} + \lambda \theta_k^{(j)} \right) \quad (\text{for } k \neq 0)\end{aligned}$$

Conocidos $x^{(i)}$, obtenemos $\theta^{(j)}$.

(2) Filtrado colaborativo

Película	Luis	Mingo	Ana	Pedro	Romance	Acción
Love Actually	5	5	0	0	?	?
Mamma mia	5	?	?	0	?	?
Pretty Woman	?	4	0	?	?	?
X-Men	0	0	5	4	?	?
Alien vs predator	0	0	5	?	?	?

$$\theta^{(1)} = \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix}, \theta^{(2)} = \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix}, \theta^{(3)} = \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix}, \theta^{(4)} = \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix}$$

Conocidos $\theta^{(j)}$, obtenemos $x^{(i)}$.

Ejercicio

.	User 1	User 2	User 3	(romance)
Movie 1	0	1.5	2.5	?

$$\theta^{(1)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \theta^{(2)} = \begin{bmatrix} 0 \\ 3 \end{bmatrix}, \theta^{(3)} = \begin{bmatrix} 0 \\ 5 \end{bmatrix}$$

$$x_1^{(1)} ?$$

Aprendizaje de atributos

- Dado $\theta^{(1)}, \dots, \theta^{(n_u)}$ estimamos $x^{(i)}$ para un ítem

$$\min_{x^{(i)}} \frac{1}{2} \sum_{j:r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)})^2 + \frac{\lambda}{2} \sum_{k=1}^n (x_k^{(i)})^2$$

- Dado $\theta^{(1)}, \dots, \theta^{(n_u)}$, estimamos $x^{(1)}, \dots, x^{(n_m)}$

$$\min_{x^{(1)}, \dots, x^{(n_m)}} \frac{1}{2} \sum_{i=1}^{n_m} \sum_{j:r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)})^2 + \frac{\lambda}{2} \sum_{i=1}^{n_m} \sum_{k=1}^n (x_k^{(i)})^2$$

Ejercicio

- Para ejecutar un descenso de gradiente para minimizar:

$$\min_{x^{(1)}, \dots, x^{(n_m)}} \frac{1}{2} \sum_{i=1}^{n_m} \sum_{j:r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)})^2 + \frac{\lambda}{2} \sum_{i=1}^{n_m} \sum_{k=1}^n (x_k^{(i)})^2$$

- ¿Cuál es la regla para $k > 0$?

$$x_k^{(i)} := x_k^{(i)} + \alpha \left(\sum_{j:r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)}) \theta_k^{(j)} \right)$$

$$x_k^{(i)} := x_k^{(i)} - \alpha \left(\sum_{j:r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)}) \theta_k^{(j)} \right)$$

$$x_k^{(i)} := x_k^{(i)} + \alpha \left(\sum_{j:r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)}) \theta_k^{(j)} + \lambda x_k^{(i)} \right)$$

$$x_k^{(i)} := x_k^{(i)} - \alpha \left(\sum_{j:r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)}) \theta_k^{(j)} + \lambda x_k^{(i)} \right)$$

(2) Filtrado Colaborativo

- Dados $x^{(1)}, \dots, x^{(n_m)}$, estimar $\theta^{(1)}, \dots, \theta^{(n_u)}$:

$$\min_{\theta^{(1)}, \dots, \theta^{(n_u)}} \frac{1}{2} \sum_{j=1}^{n_u} \sum_{i:r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)})^2 + \frac{\lambda}{2} \sum_{j=1}^{n_u} \sum_{k=1}^n (\theta_k^{(j)})^2$$

- Dados $\theta^{(1)}, \dots, \theta^{(n_u)}$, estimar $x^{(1)}, \dots, x^{(n_m)}$:

$$\min_{x^{(1)}, \dots, x^{(n_m)}} \frac{1}{2} \sum_{i=1}^{n_m} \sum_{j:r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)})^2 + \frac{\lambda}{2} \sum_{i=1}^{n_m} \sum_{k=1}^n (x_k^{(i)})^2$$

- Minimizar $x^{(1)}, \dots, x^{(n_m)}$ y $\theta^{(1)}, \dots, \theta^{(n_u)}$ simultáneamente:

$$J(x^{(1)}, \dots, x^{(n_m)}, \theta^{(1)}, \dots, \theta^{(n_u)}) = \frac{1}{2} \sum_{(i,j):r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)})^2 + \frac{\lambda}{2} \sum_{i=1}^{n_m} \sum_{k=1}^n (x_k^{(i)})^2 + \frac{\lambda}{2} \sum_{j=1}^{n_u} \sum_{k=1}^n (\theta_k^{(j)})^2$$

$$\min_{\theta^{(1)}, \dots, \theta^{(n_u)}} J(x^{(1)}, \dots, x^{(n_m)}, \theta^{(1)}, \dots, \theta^{(n_u)})$$

(2) Filtrado Colaborativo

1. Inicializar $x^{(1)}, \dots, x^{(n_m)}, \theta^{(1)}, \dots, \theta^{(n_u)}$
2. Utilizar un optimizador (e.g. descenso de gradiente):

$$x_k^{(i)} := x_k^{(i)} - \alpha \left(\sum_{j:r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)}) \theta_k^{(j)} + \lambda x_k^{(i)} \right)$$

$$\theta_k^{(j)} := \theta_k^{(j)} - \alpha \left(\sum_{i:r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)}) x_k^{(i)} + \lambda \theta_k^{(j)} \right)$$

3. La predicción para el usuario j del ítem i será:

$$\hat{y}^{(i,j)} = \theta^{(j)^T} x^{(i)}$$

Ejercicio

¿Qué valor inicial deben tener $x^{(1)}, \dots, x^{(n_m)}, \theta^{(1)}, \dots, \theta^{(n_u)}$?

- Podrían ser todos 0.
- Valores iniciales aleatorios pequeños.

Recomendación de productos

- Para cada ítem i , aprendemos un vector de atributos $x^{(i)} \in \mathbb{R}^n$
- Encontrar ítems j parecidos a i :

$$\min_j \|x^{(i)} - x^{(j)}\|$$

- Los 5 ítems a recomendar son los cinco más cercanos a i
- Están x_k y x_m en las mismas unidades?

Cold start: nuevo usuario

- ¿Qué ocurre cuando aparece un nuevo usuario?

Película	Luis	Mingo	Ana	Pedro	Juan	Romance	Acción
Love Actually	5	5	0	0	?	?	?
Mamma mia	5	?	?	0	?	?	?
Pretty Woman	?	4	0	?	?	?	?
X-Men	0	0	5	4	?	?	?
Alien vs predator	0	0	5	?	?	?	?

$$\min_{\substack{x^{(1)}, \dots, x^{(n_m)} \\ \theta^{(1)}, \dots, \theta^{(n_u)}}} \frac{1}{2} \sum_{(i,j): r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)})^2 + \frac{\lambda}{2} \sum_{i=1}^{n_m} \sum_{k=1}^n (x_k^{(i)})^2 + \frac{\lambda}{2} \sum_{j=1}^{n_u} \sum_{k=1}^n (\theta_k^{(j)})^2$$

$$\theta^{(5)} = \quad \theta^{(5)} x^{(i)} =$$

Solución: normalización de la media

$$Y = \begin{bmatrix} 5 & 5 & 0 & 0 & ? \\ 5 & ? & ? & 0 & ? \\ ? & 4 & 0 & ? & ? \\ 0 & 0 & 5 & 4 & ? \\ 0 & 0 & 5 & 0 & ? \end{bmatrix} \quad \mu = \quad Y = \begin{bmatrix} 2.5 & 2.5 & -2.5 & -2.5 & ? \\ 2.5 & ? & ? & -2.5 & ? \\ ? & 2 & -2 & ? & ? \\ -2.25 & -2.25 & 2.75 & 1.75 & ? \\ -1.25 & -1.25 & 3.75 & -1.25 & ? \end{bmatrix}$$

- Predicción de la calificación del usuario j para el ítem i :

$$\square \text{ Predicciones:} \\ Y = \begin{bmatrix} 5 & 5 & 0 & 0 \\ 5 & ? & ? & 0 \\ ? & 4 & 0 & ? \\ 0 & 0 & 5 & 4 \\ 0 & 0 & 5 & 0 \end{bmatrix} \begin{bmatrix} (\theta^{(1)})^T(x^{(1)}) & (\theta^{(2)})^T(x^{(1)}) & \dots & (\theta^{(n_u)})^T(x^{(1)}) \\ (\theta^{(1)})^T(x^{(2)}) & (\theta^{(2)})^T(x^{(2)}) & \dots & (\theta^{(n_u)})^T(x^{(2)}) \\ \vdots & \vdots & \ddots & \vdots \\ (\theta^{(1)})^T(x^{(n_m)}) & (\theta^{(2)})^T(x^{(n_m)}) & \dots & (\theta^{(n_u)})^T(x^{(n_m)}) \end{bmatrix}$$

$$X = \begin{bmatrix} x^{(1)}, \dots, x^{(n_m)} \end{bmatrix} \\ \theta = \begin{bmatrix} \theta^{(1)}, \dots, \theta^{(n_u)} \end{bmatrix} \\ Y = ?$$

(3) vecino más próximo (usuario)

- a, b : usuarios, p: artículos
- $r_{a,p}$: calificación del usuario al artículo p,
- P: artículos calificados por a y b

$$pred(a, p) = \bar{r}_a + \frac{\sum_{b \in N} sim(a, b) * (r_{b,p} - \bar{r}_b)}{\sum_{b \in N} sim(a, b)}$$

■ Correlación

$$sim(a, b) = \frac{\sum_{p \in P} (r_{a,p} - \bar{r}_a)(r_{b,p} - \bar{r}_b)}{\sqrt{\sum_{p \in P} (r_{a,p} - \bar{r}_a)^2} \sqrt{\sum_{p \in P} (r_{b,p} - \bar{r}_b)^2}}$$

	p1	p2	p3	p4	media	p5	sim_a	p-prom	
a	5	3	4	4	4	?	0,768	3,483	4,54
b	3	1	2	3	2,25	3	0,85	0,75	0,64
c	4	3	4	3	3,5	5	0,71	1,5	1,06
d	3	3	1	5	3	4	0	1	0
e	1	5	5	2	3,25	1	-0,79	-2,25	1,78

(3) vecino más próximo (usuario)

- a, b : usuarios, p: artículos
- $r_{a,p}$: calificación del usuario al artículo p,
- P: artículos calificados por a y b

$$pred(a, p) = \bar{r}_a + \frac{\sum_{b \in N} sim(a, b) * (r_{b,p} - \bar{r}_b)}{\sum_{b \in N} sim(a, b)}$$

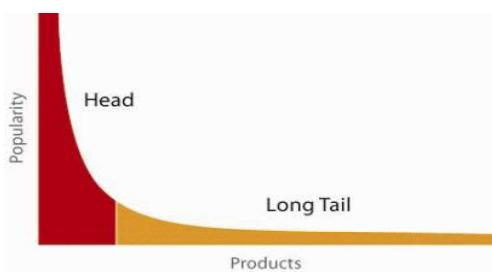
■ Coseno

$$sim(\vec{a}, \vec{b}) = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}| * |\vec{b}|}$$

	p1	p2	p3	p4	media	p5	sim_a	p-prom	
a	5	3	4	4	4	?	3,654975	1,318	4,36
b	3	1	2	3	2,25	3	0,97532	0,75	0,73
c	4	3	4	3	3,5	5	0,99224	1,5	1,49
d	3	3	1	5	3	4	0,89072	1	0,89
e	1	5	5	2	3,25	1	0,79669	-2,25	-1,79

(4) Ley potencial

- El 20% de los artículos acumulan el 80% de evaluaciones positivas



- El sistema debe recomendar artículos no conocidos que los usuarios descubran que les gustan

(4) Evaluación



- Training set (99,072,112 ratings):
<user, movie, date of grade, grade>
- Test set (1,408,789 ratings):
<user, movie, date of grade>
- Evaluación: RMSE

Laboratorio

- Sistema de recomendación de películas
- Implementar el descenso de gradiente de filtrado colaborativo por usuario
- Probar con una base de datos real

Tema 9. Métodos no paramétricos. Procesos Gaussianos

30231 - Aprendizaje Automático
Grado en Ingeniería Informática
Escuela de Ingeniería y Arquitectura

Índice

- Introducción
- Gaussianas, 2
- Regresión con Procesos Gaussianos
 - Sin ruido de observación
 - Con ruido de observación
- Conclusiones
 - Kernels
 - Aprendizaje Activo
- Lecturas
 - Murphy, Bishop, Nando de Freitas

Introducción: Métodos paramétricos

- Métodos basados en funciones definidas por **parámetros**
- **Número** de parámetros del modelo fijado de antemano
$$h_{\theta}(\mathbf{x}) = \theta^T \mathbf{x} = \theta_0 + \theta_1 x_1 + \dots + \theta_D x_D$$
$$h_{\theta}(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \dots + \theta_n x^n$$
- **Aprendizaje:** en estimar los valores de estos **parámetros** utilizando los datos de entrenamiento
- **Predicción:** en base a los parámetros aprendidos
- Puede ser necesario elegir el orden del modelo mediante validación cruzada

Métodos no paramétricos

- Parámetros **NO** fijados de antemano
- Su número puede crecer con los datos
- Las predicciones utilizan los datos de entrenamiento
- Ejemplos:
 - Clasificación por vecino más próximo
 - Procesos Gaussianos
 - SVM (Support Vector Machines)

Clasificación

Notación:

- n objetos conocidos (clases):

$$\omega = \{\omega_1, \dots, \omega_i, \dots, \omega_n\}$$

- m descriptores utilizados:

$$\mathbf{x} = (x_1, \dots, x_j, \dots, x_m)^T$$

- N muestras del descriptor j de la clase i :

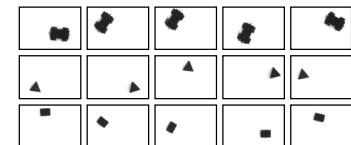
$$\mathbf{x}_{ij} = (x_{ij1}, \dots, x_{ijk}, \dots, x_{jN})^T$$

Dada una muestra \mathbf{x} ,
¿a qué clase ω_i corresponde?

¿se trata de un
objeto desconocido?

Aprendizaje:

- Tomar varias imágenes de cada objeto, y calcular sus descriptores



Explotación:

- Calcular de los descriptores de cada objeto en la imagen
- Comparar con los valores estimados de los descriptores de los objetos conocidos

Diferentes métodos usan diferentes técnicas para efectuar esta comparación

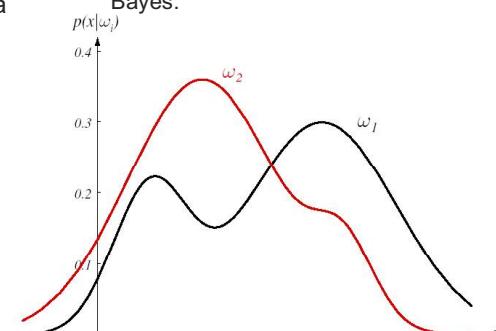
Clasificación Bayesiana

- Los descriptores se consideran variables aleatorias.
- Modelo probabilístico de cada clase:

$$p(\mathbf{x}/\omega_i)$$

- Probabilidad a priori de cada clase:

$$P(\omega_i)$$



$$p(\mathbf{x}/\omega_i) = p(\mathbf{x}/\omega_j)? \\ P(\omega_i) = P(\omega_j)?$$

Clasificación Bayesiana

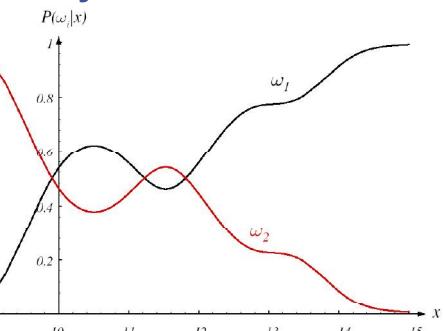
Regla de Bayes:

$$P(\omega_i/\mathbf{x}) = \frac{p(\mathbf{x}/\omega_i)P(\omega_i)}{p(\mathbf{x})}$$

$$p(\mathbf{x}) = \sum_{j=1}^n p(\mathbf{x}/\omega_j)P(\omega_j)$$

- Distribuye la probabilidad entre todas las clases:

$$\sum_{i=1}^n P(\omega_i/\mathbf{x}) = 1$$



- Decidimos ω_i si:

$$P(\omega_i/\mathbf{x}) = \max_{j=1}^n P(\omega_j/\mathbf{x})$$

Métodos no paramétricos

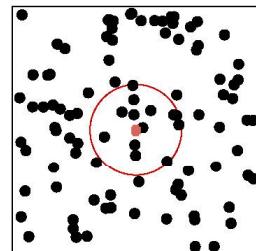
- Estimación de la función de probabilidad:

$$p(\mathbf{x}/\omega_i) ?$$

- Estimación de la probabilidad a posteriori:

$$P(\omega_i/\mathbf{x}) ?$$

$$P(\omega_i/\mathbf{x}) \simeq \frac{k_i}{k}$$



8

- **Probabilidad a posteriori:** Dada una ventana V alrededor de \mathbf{x} , que incluye k muestras, k_i de las cuales pertenecen a ω_i :

El vecino más próximo

- Dadas N muestras de m descriptores de n clases:

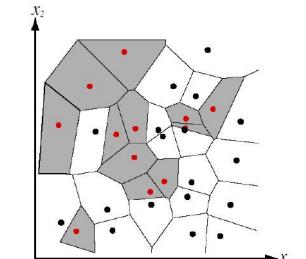
$$\mathbf{x}_{i,k} = (x_{i,1}, \dots, x_{i,N})^T$$

- Dados m descriptores del objeto a identificar:

$$\mathbf{x} = (x_1, \dots, x_m)^T$$

- Escoger la clase ω_i tal que:

$$\mathbf{x}_{i,k} = \min_k D^2(\mathbf{x}, \mathbf{x}_{i,k})$$



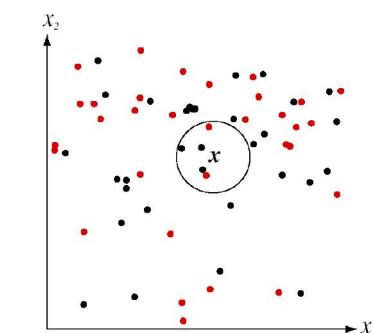
- **Método subóptimo:** tasa de errores mayor que la Clasificación Bayesiana.

¡La tasa de errores
no es más del doble!

9

Los k vecinos más próximos

- Escoger la clase ω_i más frecuente en los k vecinos más próximos a \mathbf{x} .



- Los empates se pueden evitar aumentando k ($n=2$, k impar).

- Para n, m y N grande, es computacionalmente costoso.

$$O(n m N)$$

- Distancias parciales:

$$D_r^2(\mathbf{a}, \mathbf{b}) = \sum_{i=1}^r (a_i - b_i)^2$$

$$r < m$$

- Se abandona el cálculo de la distancia a una muestra cuando la distancia parcial es mayor que la distancia total al k -ésimo vecino.

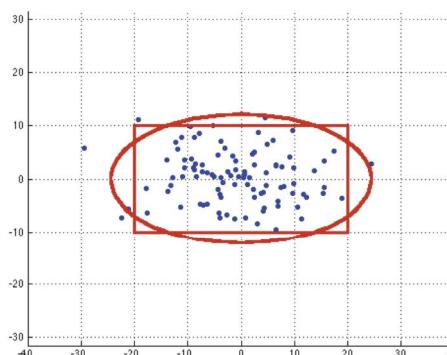
10

Gaussianas, 2

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \sim \mathcal{N}(\mu, \Sigma)$$

$$\mu =$$

$$\Sigma =$$



$$\mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \Sigma = \begin{bmatrix} \sigma_1^2 & \rho_{12}\sigma_1\sigma_2 \\ \rho_{12}\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix}$$

11

Distribución de Chi Cuadrado

$$x_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$$

$$\frac{x_i - \mu_i}{\sigma_i} \sim \mathcal{N}(0, 1)$$

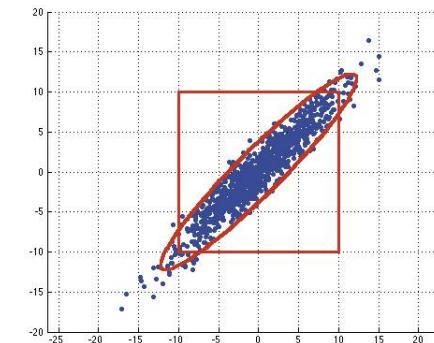
$$\left(\frac{x_i - \mu_i}{\sigma_i} \right)^2 \sim \chi_1^2$$

$$(\mathbf{x} - \boldsymbol{\mu})^t \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \sim \chi_n^2$$

Percentage Points of the Chi-Square Distribution

Degrees of Freedom	Probability of a larger value of χ^2								
	0.99	0.95	0.90	0.75	0.50	0.25	0.10	0.05	
1	0.000	0.004	0.016	0.102	0.455	1.32	2.71	3.84	6.63
2	0.020	0.103	0.211	0.575	1.386	2.77	4.61	5.99	9.21
3	0.115	0.352	0.584	1.212	2.366	4.11	6.25	7.81	11.34
4	0.297	0.711	1.064	1.923	3.357	5.39	7.78	9.49	13.28
5	0.554	1.145	1.610	2.675	4.351	6.63	9.24	11.07	15.09
6	0.872	1.635	2.204	3.455	5.348	7.84	10.64	12.59	16.81
7	1.239	2.167	2.833	4.255	6.346	9.04	12.02	14.07	18.48
8	1.647	2.733	3.490	5.071	7.344	10.22	13.36	15.51	20.09

Ejercicio

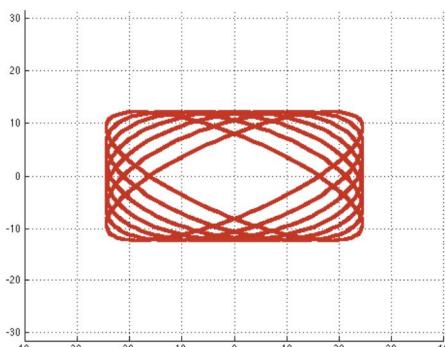


$$\boldsymbol{\mu} = \quad \quad \quad \boldsymbol{\Sigma} =$$



13

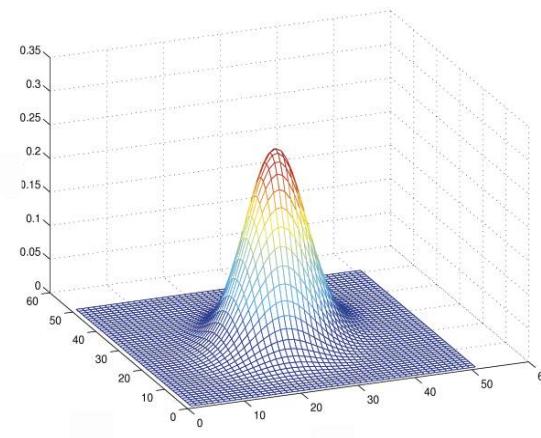
Gaussianas, 2



$$\sigma_1 = 10, \sigma_2 = 5$$

$$\rho_{12} = [-0.75 \ -0.5 \ -0.25 \ 0.0 \ 0.25 \ 0.5 \ 0.75]$$

Gaussianas, 2



Probabilidad Marginal y Condicional

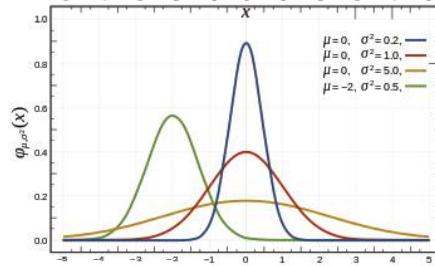
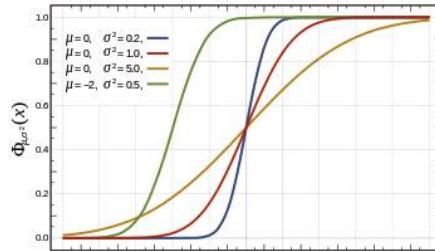
$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \sim \mathcal{N} \left(\mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix} \right)$$

$$p(x_1) =$$

$$p(x_2) =$$

$$p(x_1|x_2 = \hat{x}_2) =$$

Muestreo por Transformación Inversa



$$u \sim \mathcal{U}(0, 1)$$

$$x = P^{-1}(u) \sim \mathcal{N}(0, 1)$$

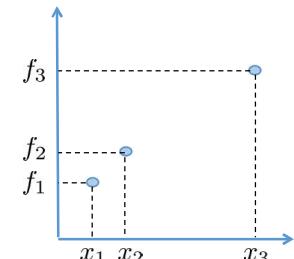
$$y = x\sigma + \mu \sim \mathcal{N}(\mu, \sigma^2)$$

$$\mathbf{y} = \mathbf{x}\mathbf{L} + \mu \sim \mathcal{N}(\mu, \Sigma)$$

\mathbf{L} ?

Regresión con procesos Gaussianos

Dados datos $D = \{(x_1, f_1), (x_2, f_2), (x_3, f_3)\}$



Modelar f_1, f_2, f_3 como un vector de una Gausiana multi-varia

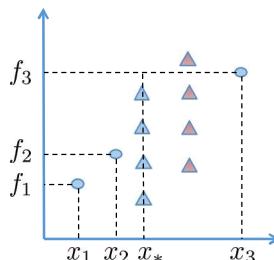
$$\begin{pmatrix} f_1 \\ f_2 \\ f_3 \end{pmatrix} =$$

K_{ij} es una medida de semejanza, e.j.

$$k_{ij} =$$

Regresión con procesos Gaussianos

Dados datos $D = \{(x_1, f_1), (x_2, f_2), (x_3, f_3)\}$
Dado x_* , predecir $f_* = ?$

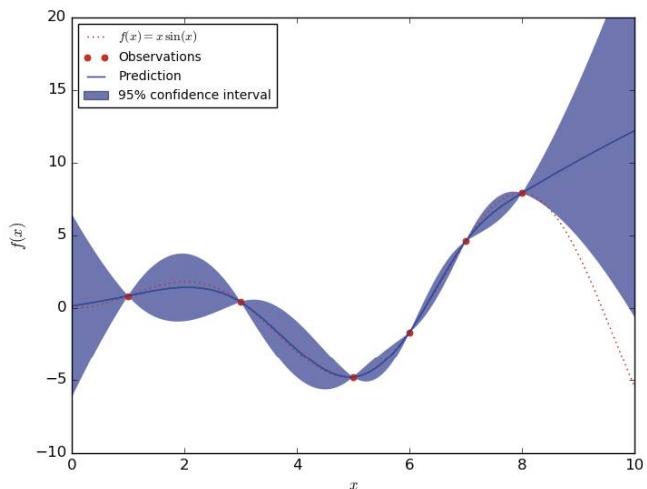


$$f = \begin{pmatrix} f_1 \\ f_2 \\ f_3 \end{pmatrix} \sim N(0, K)$$

¿Cómo predecimos f_* ?

Regresión con Procesos Gaussianos

■



GP: distribuciones sobre funciones

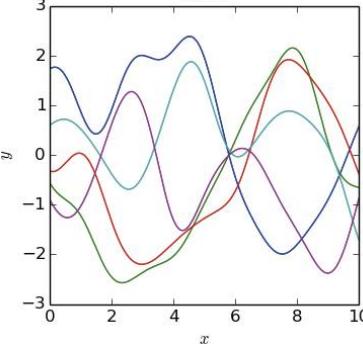
- Un proceso Gaussiano (GP) es una distribución sobre funciones:

$$f(x) = GP(m(x), k(x, x'))$$

$$m(x) = E[f(x)]$$

$$\begin{aligned} k(x, x') &= E[(f(x) - m(x))(f(x') - m(x'))] \\ &= e^{-\frac{1}{2}(x-x')^2} \end{aligned}$$

Prior



- Muestrear un GP:

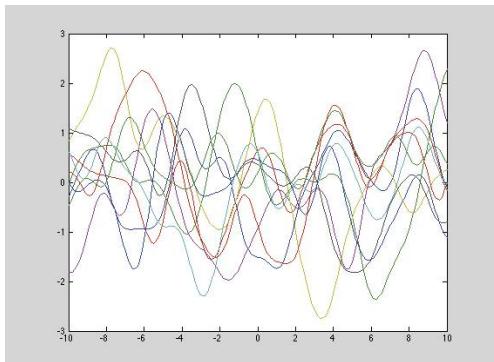
- Crear $x_{1..N}$
- Crear $K_{N \times N}$
- Cholesky: $K = LL^t$
- $f^i = \mathcal{N}(0, \mathcal{I})L$

GP: distribuciones sobre funciones

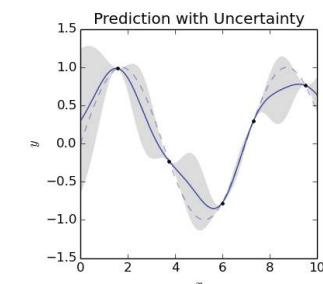
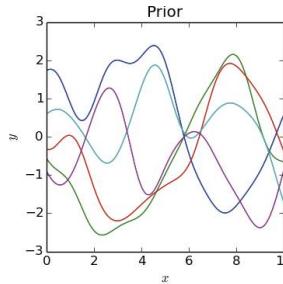
- Matlab:

```
x = -10:1:10;
n = length(x);

K = exp((-1/2)*(repmat(x, n, 1) - repmat(x', 1, n)).^2);
L = chol(K);
F = randn(n)*L;
plot(x,F);
```

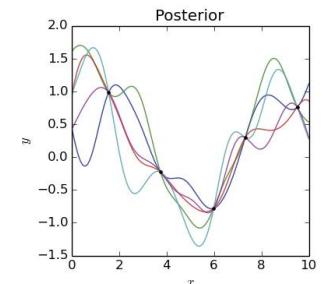


GP a posteriori



$$\mathcal{D} = \{(x_i, f_i), i = 1 : N\}$$

$$p(f|\mathcal{D}) = \frac{p(\mathcal{D}|f)p(f)}{p(\mathcal{D})}$$



Regresión con GPs sin ruido de observación

- Dados un conjunto de datos de entrenamiento $\mathcal{D} = \{(x_i, f_i), i = 1 : N\}$, donde $f_i = f(x_i)$
- Dado un conjunto de N_* datos de prueba x_* , queremos predecir los valores f_*

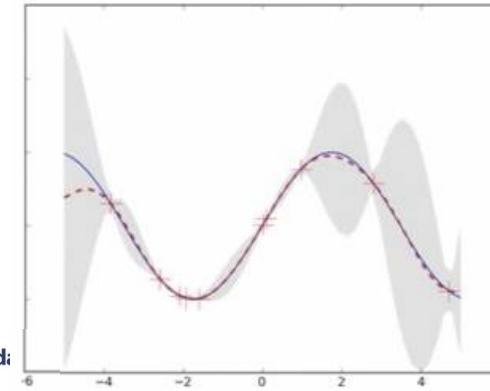
$$\begin{pmatrix} f \\ f_* \end{pmatrix} = \mathcal{N} \left(\begin{pmatrix} \mu \\ \mu_* \end{pmatrix}, \begin{pmatrix} K & K_* \\ K_*^T & K_{**} \end{pmatrix} \right)$$

- donde $K = k(x, x)$ es $N \times N$, $K_* = k(x, x_*)$ es $N \times N_*$, $K_{**} = k(x, x)$ es $N_* \times N_*$
- Supongamos que:

$$k(x, x') = \sigma_f^2 \exp \left(-\frac{1}{2l^2}(x - x')^2 \right)$$

Regresión con GPs sin ruido de observación

$$\begin{aligned} \begin{pmatrix} f \\ f_* \end{pmatrix} &= \mathcal{N} \left(\begin{pmatrix} \mu \\ \mu_* \end{pmatrix}, \begin{pmatrix} K & K_* \\ K_*^T & K_{**} \end{pmatrix} \right) \\ p(f_*|x_*, x, f) &= \mathcal{N}(\mu_{f_*}, K_{f_*}) \\ \mu_{f_*} &= \mu_* + K_*^T K^{-1} (f - \mu) \\ K_{f_*} &= K_{**} - K_*^T K^{-1} K_* \end{aligned}$$

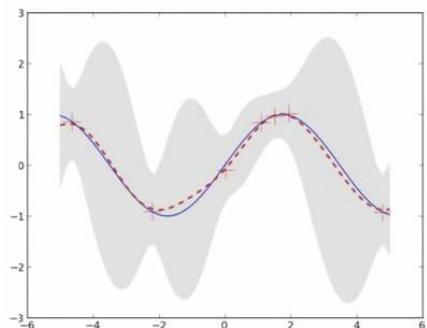


Regresión con GPs y ruido

$$y = f(x) + \epsilon, \text{ donde } \epsilon \sim \mathcal{N}(0, \sigma_y^2)$$

$$\begin{aligned} p(\mathbf{f}|\mathbf{x}) &\sim \\ p(\mathbf{y}|\mathbf{f}) &\sim \\ \text{cov}(\mathbf{y}|\mathbf{x}) &= \end{aligned}$$

$$\begin{pmatrix} \mathbf{y} \\ \mathbf{f}_* \end{pmatrix} =$$



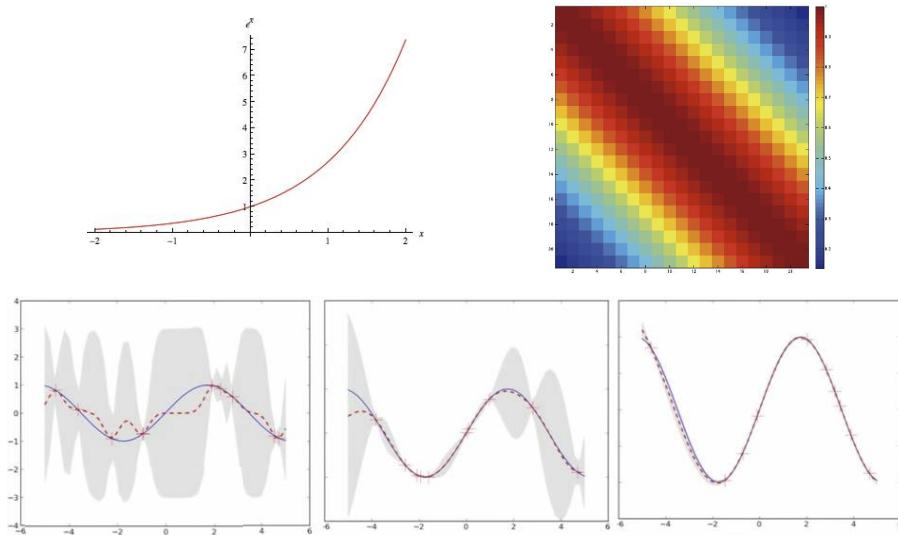
Algoritmo GP

$$\begin{pmatrix} \mathbf{y} \\ \mathbf{f}_* \end{pmatrix} = \mathcal{N} \left(\mathbf{0}, \begin{pmatrix} K_y & K_* \\ K_*^T & K_{**} \end{pmatrix} \right) \quad \begin{aligned} p(\mathbf{f}_*|\mathbf{x}_*, \mathbf{x}, \mathbf{y}) &= \mathcal{N}(\mu_{f_*}, K_{f_*}) \\ \mu_{f_*} &= K_*^T K_y^{-1} \mathbf{y} \\ K_{f_*} &= K_{**} - K_*^T K_y^{-1} K_* \end{aligned}$$

$$\begin{aligned} \mathbf{K}_y &= \mathbf{K} + \sigma_y^2 * \text{eye}(n); \\ \mathbf{L} &= \text{chol}(\mathbf{K}_y); \\ \alpha &= \mathbf{L}^T \backslash (\mathbf{L} \backslash \mathbf{y}); \\ \mathbf{V} &= \mathbf{L} \backslash \mathbf{K}_*; \\ \mu_{f_*} &= \mathbf{K}_*^T * \alpha; \\ \mathbf{K}_{f_*} &= \mathbf{K}_{**} - \mathbf{V}^T * \mathbf{V}; \end{aligned}$$

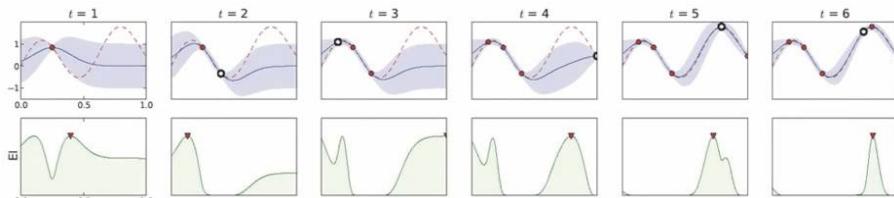
Kernels

$$k(x, x') = \sigma_f^2 \exp\left(-\frac{1}{2l^2}(x - x')^2\right)$$



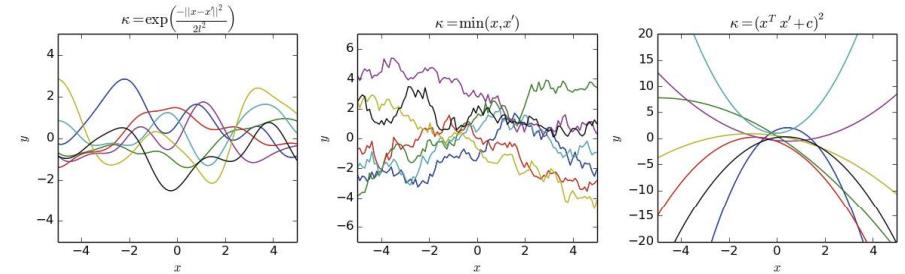
Aprendizaje activo con GPs

- Seleccionar puntos a medir para maximizar la información obtenida, minimizar el error de predicción, minimizar la entropía...



- Origen: minería (Kriging)
- Otras aplicaciones: ciencias medioambientales, epidemiología, robótica...

Kernels



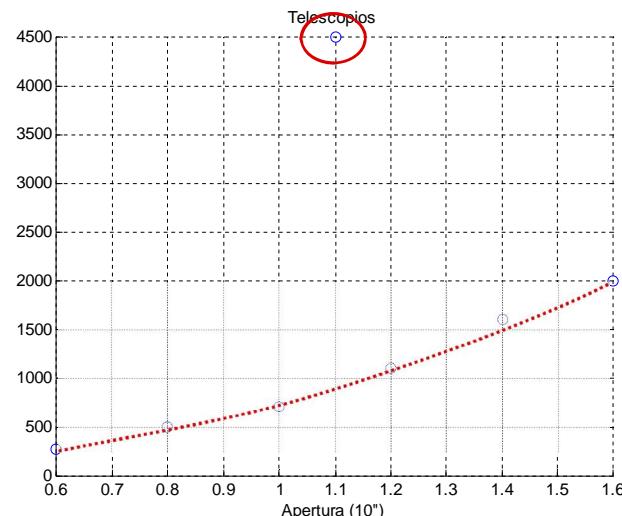
Ejercicio de Regresión

Queremos predecir el precio de los telescopios Newtonianos, en función de su apertura, y hemos encontrado en Internet los siguientes precios:

Apertura (pulgadas)	Precio (Euros)
6	270
8	500
10	700
11	4500
12	1100
14	1600
16	2000

- Explica qué técnica de aprendizaje es más adecuada
- Detalla el algoritmo de entrenamiento, y la forma de obtener la predicción indicando el valor de todas las variables necesarias.
- Si utilizas un algoritmo con solución analítica calcula el modelo. Si utilizas uno iterativo, realiza una iteración, partiendo de un modelo en el que el precio sea igual a 100 euros por pulgada. Usa el modelo obtenido para predecir el precio de un telescopio de 12,5"

Echamos un vistazo a los datos



a) Algoritmo de Aprendizaje

- Regresión monovariante
 - La relación es claramente no-lineal → modelo polinómico
 - Un polinomio de 2º grado podría ser suficiente
 - ◆ Es un modelo simple, no hará falta regularizar
- Hay un dato claramente espurio (el precio del de 11'')
 - A. Eliminamos el dato espurio y hacemos regresión convencional
 1. Ecuación normal $\hat{\theta} = (X^T X)^{-1} X^T \mathbf{y}$
 2. Descenso gradiente $\mathbf{g}(\theta) = X^T \mathbf{r} = X^T (X\theta - \mathbf{y})$
 $\theta_{k+1} := \theta_k - \alpha \mathbf{g}(\theta_k)$
 - B. Utilizamos regresión robusta (con el coste de Huber)
 - ◆ Descenso de gradiente $\mathbf{g}(\theta) = \sum_{|r_i| \leq \delta} X_i^T r_i + \sum_{|r_i| > \delta} \delta X_i^T \text{signo}(r_i)$
 $= X_{good}^T \mathbf{r}_{good} + \delta X_{bad}^T \text{signo}(\mathbf{r}_{bad})$

b) Detalle del Algoritmo

- Regresión polinómica de grado 2
 - Normalizamos los atributos: $x = \text{apertura} / 10$

X			y
1's	x	x^2	
1	0,6	0,36	270
1	0,8	0,64	500
1	1	1	700
1	1,1	1,21	4500
1	1,2	1,44	1100
1	1,4	1,96	1600
1	1,6	2,56	2000

Notas sobre normalización:

- Imprescindible si usas descenso de gradiente, opcional si usas la ecuación normal
- Puede hacerse con la media y la varianza, pero basta con una normalización sencilla como ésta.

- Predicción, con X normalizada:

$$\theta = \begin{pmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{pmatrix}$$

$$\hat{\mathbf{y}}_{test} = X_{test} \theta$$

c) Cálculo del Modelo con coste L2

- A1) Ecuación normal $\hat{\theta} = (X^T X)^{-1} X^T \mathbf{y}$

X			y
1's	x	x^2	
1	0,6	0,36	270
1	0,8	0,64	500
1	1	1	700
1	1,1	1,21	4500
1	1,2	1,44	1100
1	1,4	1,96	1600
1	1,6	2,56	2000

$$X^T X = \begin{pmatrix} N & \sum x_i & \sum x_i^2 \\ \sum x_i & \sum x_i^2 & \sum x_i^3 \\ \sum x_i^2 & \sum x_i^3 & \sum x_i^4 \end{pmatrix} \quad X^T \mathbf{y} = \begin{pmatrix} \sum y_i \\ \sum x_i y_i \\ \sum x_i^2 y_i \end{pmatrix}$$

c) Cálculo del Modelo con coste L2

■ A1) Ecuación normal (cont.)

1's	x	x^2	x^3	x^4	y	xy	x^2y
1	0,6	0,36	0,216	0,1296	270	162	97,2
1	0,8	0,64	0,512	0,4096	500	400	320
1	1	1	1	1	700	700	700
1	1,1	1,21	1,331	1,4041	4500	4950	5445
1	1,2	1,44	1,728	2,0736	1100	1322	1584
1	1,4	1,96	2,744	3,8416	1600	2240	3136
1	1,6	2,56	4,096	6,5536	2000	3200	5120
6	6,6	7,96	10,296	14,008	6170	8022	10957,2

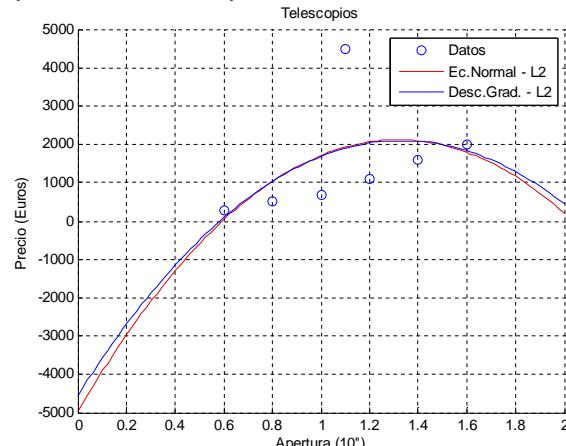
$$X^T X = \begin{pmatrix} 6 & 6.6 & 7.96 \\ 6.6 & 7.96 & 10.296 \\ 7.96 & 10.296 & 14.008 \end{pmatrix} \quad X^T y = \begin{pmatrix} 6170 \\ 8022 \\ 10957.2 \end{pmatrix}$$

$$\hat{\theta} = (X^T X)^{-1} X^T y = \begin{pmatrix} 88.2 \\ -249.1 \\ 915.2 \end{pmatrix}$$

6

Solución con coste L2

■ Si no se quita el dato espurio



♦ Despues de 2000 iteraciones de descenso de gradiente (hay que seguir iterando)

c) Cálculo del Modelo con coste L2

■ A2) Descenso de gradiente

$$\mathbf{g}(\theta) = X^T \mathbf{r} = X^T (X\theta - \mathbf{y})$$

$$\theta_{k+1} := \theta_k - \alpha \mathbf{g}(\theta_k)$$

x			y
1's	x	x^2	
1	0,6	0,36	270
1	0,8	0,64	500
1	1	1	700
1	1,1	1,21	4500
1	1,2	1,44	1100
1	1,4	1,96	1600
1	1,6	2,56	2000

100 Euro/pulgada =
1000Euro/10 pulgadas

$$\theta_0 = \begin{pmatrix} 0 \\ 1000 \\ 0 \end{pmatrix}$$

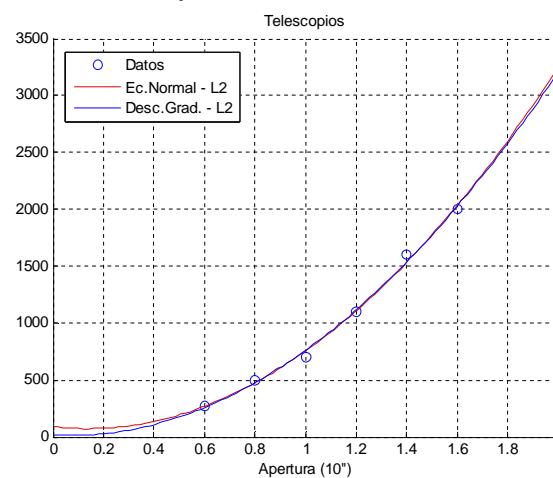
$$\mathbf{r} = X\theta - \mathbf{y} = \begin{pmatrix} 330 \\ 300 \\ 300 \\ 100 \\ -200 \\ -400 \end{pmatrix}$$

$$\mathbf{g}(\theta) = X^T \mathbf{r} = \begin{pmatrix} 430 \\ -62 \\ -661.2 \end{pmatrix}$$

$$\theta_1 := \theta_0 - 0.1 \mathbf{g}(\theta_k) = \begin{pmatrix} -43 \\ 1006.2 \\ 66.1 \end{pmatrix}$$

Solución con coste L2

■ Quitando el dato espurio



♦ Despues de 2000 iteraciones de descenso de gradiente (hay que seguir iterando)

c) Cálculo del Modelo con coste de Huber

- B) Regresión robusta (Huber) $\theta_{k+1} := \theta_k - \alpha \mathbf{g}(\theta_k)$

$$\begin{aligned}\mathbf{g}(\theta) &= \sum_{|r_i| \leq \delta} X_i^T r_i + \sum_{|r_i| > \delta} \delta X_i^T signo(r_i) \\ &= X_{good}^T \mathbf{r}_{good} + \delta X_{bad}^T signo(\mathbf{r}_{bad})\end{aligned}$$

X			y
1's	x	x^2	
1	0,6	0,36	270
1	0,8	0,64	500
1	1	1	700
1	1,1	1,21	4500
1	1,2	1,44	1100
1	1,4	1,96	1600
1	1,6	2,56	2000

100 Euro/pulgada =
1KEuro/10 pulgadas

$$\theta_0 = \begin{pmatrix} 0 \\ 1000 \\ 0 \end{pmatrix}$$

$$\mathbf{r} = X\theta - \mathbf{y} = \begin{pmatrix} 340 \\ 300 \\ 300 \\ -3400 \\ 100 \\ -200 \\ -400 \end{pmatrix} \quad \delta = 500$$

bad

10

c) Cálculo del Modelo con coste de Huber

- B) Regresión robusta (Huber)

$$\begin{aligned}\mathbf{g}(\theta) &= \sum_{|r_i| \leq \delta} X_i^T r_i + \sum_{|r_i| > \delta} \delta X_i^T signo(r_i) \\ &= X_{good}^T \mathbf{r}_{good} + \delta X_{bad}^T signo(\mathbf{r}_{bad}) \\ &= \begin{pmatrix} 430 \\ -62 \\ -661 \end{pmatrix} - 500 \begin{pmatrix} 1 \\ 1.1 \\ 1.21 \end{pmatrix} = \begin{pmatrix} -70 \\ -612 \\ -1266 \end{pmatrix}\end{aligned}$$

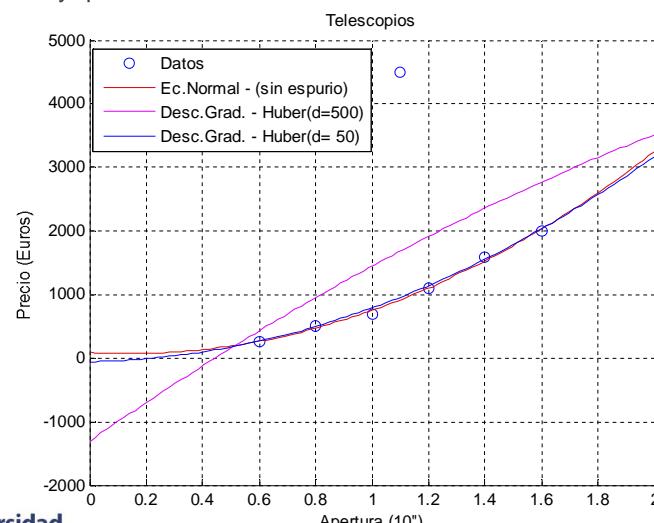
$$\theta_0 = \begin{pmatrix} 0 \\ 1000 \\ 0 \end{pmatrix}$$

$$\theta_1 := \theta_0 - 0.1 \mathbf{g}(\theta_k) = \begin{pmatrix} 7 \\ 1061.2 \\ 126.6 \end{pmatrix}$$

Solución con coste de Huber

- Con norma de Huber no es necesario quitar el espurio

♦ Pero hay que afinar con el valor de δ



12

Ejercicio de Clasificación...

Queremos diseñar un sistema de reconocimiento de objetos con visión 2D, basado en parámetros. Supondremos que los parámetros son Gaussianos y condicionalmente independientes dada la clase. Durante la fase de aprendizaje se han tomado 5 imágenes de los tres objetos a reconocer, y se ha calculado en cada una de ellas el perímetro y el radio máximo del objeto:

Círculo	
Perim	Rmax
307	46
316	51
296	53
333	50
319	50

Cuadrado	
Perim	Rmax
396	73
388	78
397	71
421	71
398	68

Triángulo	
Perim	Rmax
346	79
354	92
330	89
363	78
317	87

Ejercicio de Clasificación (continuación)

Durante la fase de reconocimiento, tomamos una imagen, y en ella se detectan 3 objetos que queremos clasificar, cuyos parámetros son:

	Perim	Rmax
Objeto1	335	88
Objeto2	398	50
Objeto3	327	52

- a) Explica qué técnica de clasificación vas a utilizar, y entrena el modelo.
- b) Clasifica el objeto 1 y calcula la probabilidad de que corresponda a la clase asignada.
- c) Sospechamos que uno de los objetos a reconocer no corresponde a ninguna de las tres clases aprendidas. Compruébalo.

a) Clasificador Bayes Ingénuo

$$\hat{y} = \arg \max_i p(\mathbf{x}|y_i)p(y_i)$$

■ Modelo:

$$p(x_k|y_j) \sim \mathcal{N}(\mu_{jk}, \sigma_{jk}^2)$$

$$p(\mathbf{x}|y_j) = \prod_{k=1}^D p(x_k|y_j) = \prod_{k=1}^D \frac{1}{\sqrt{2\pi}\sigma_{jk}} \exp \left\{ -\frac{1}{2} \left(\frac{x_k - \mu_{jk}}{\sigma_{jk}} \right)^2 \right\}$$

■ Aprendizaje:

$$N_j = \sum_i [y^{(i)} = j]$$

$$\mu_{jk} = \frac{1}{N_j} \sum_{y^{(i)}=j} x_k^{(i)}$$

$$\sigma_{jk}^2 = \frac{1}{N_j - 1} \sum_{y^{(i)}=j} (x_k^{(i)} - \mu_{jk})^2$$

a) Aprendemos los modelos

$$\mu_{jk} = \frac{1}{N_j} \sum_{y^{(i)}=j} x_k^{(i)} \quad \sigma_{jk}^2 = \frac{1}{N_j - 1} \sum_{y^{(i)}=j} (x_k^{(i)} - \mu_{jk})^2$$

Círculo	j = 1		j = 2		j = 3	
	Perim	Rmax	Perim	Rmax	Perim	Rmax
307	46	396	73	346	79	
316	51	388	78	354	92	
296	53	397	71	330	89	
333	50	421	71	363	78	
319	50	398	68	317	87	
μ_{jk}	314,2	50	400	72,2	342	85
σ_{jk}	13,81	2,55	12,39	3,70	18,51	6,20
	k=1		k=2			

b) Clasificar Objeto 1

$$p(\mathbf{x}|y_j) = \prod_{k=1}^D p(x_k|y_j) = \prod_{k=1}^D \frac{1}{\sqrt{2\pi}\sigma_{jk}} \exp \left\{ -\frac{1}{2} \left(\frac{x_k - \mu_{jk}}{\sigma_{jk}} \right)^2 \right\}$$

	Círculo		Cuadra		Triángul	
	Perim	Rmax	Perim	Rmax	Perim	Rmax
μ_{jk}	314,2	50	400	72,2	342	85
σ_{jk}	13,81	2,55	12,39	3,70	18,51	6,20
Objeto1	335	88	335	88	335	88
$\left(\frac{x_k - \mu_{jk}}{\sigma_{jk}} \right)^2$	2,27	222,15	27,52	18,22	0,14	0,23
$p(x_k y_j)$	0,0092	9E-50	3,4E-08	1,19E-05	0,0200	0,0572
$p(\mathbf{x} y_j)$	8,36E-52		4,04E-13		0,001148	

$$p(y_i|\mathbf{x}) = \frac{p(\mathbf{x}|y_i)p(y_i)}{\sum_j p(\mathbf{x}|y_j)p(y_j)} = 0,99999...$$

c) ¿Objeto desconocido?

Círculo	
Perim	Rmax
307	46
316	51
296	53
333	50
319	50

Cuadrado	
Perim	Rmax
396	73
388	78
354	92
397	71
421	71
398	68

Triángulo	
Perim	Rmax
346	79
354	92
330	89
363	78
317	87

	Perim	Rmax
Objeto1	335	88
Objeto2	398	50
Objeto3	327	52

c) Hacemos test de chi-cuadrado

$$d_j^2 = \sum_{k=1}^D z_{jk}^2 = \sum_{k=1}^D \left(\frac{x_k - \mu_{jk}}{\sigma_{jk}} \right)^2 \sim \chi_D^2$$

Círculo		Cuadra		Triángul		
Perim	Rmax	Perim	Rmax	Perim	Rmax	
μ_{jk}	314,2	50	400	72,2	342	85
σ_{jk}	13,81	2,55	12,39	3,70	18,51	6,20

$$\begin{aligned} \text{Objeto2} & 398 & 50 & 398 & 50 & 398 & 50 \\ \left(\frac{x_k - \mu_{jk}}{\sigma_{jk}} \right)^2 & 36,82 & 0,00 & 0,03 & 35,97 & 9,16 & 31,82 \\ d_j^2 & 36,82 & & 36,00 & & 40,98 & \end{aligned}$$

$$\chi_{2, 0,95}^2 = 5,9915 \quad d_j^2 \not\in \chi_{D, 1-\alpha}^2$$

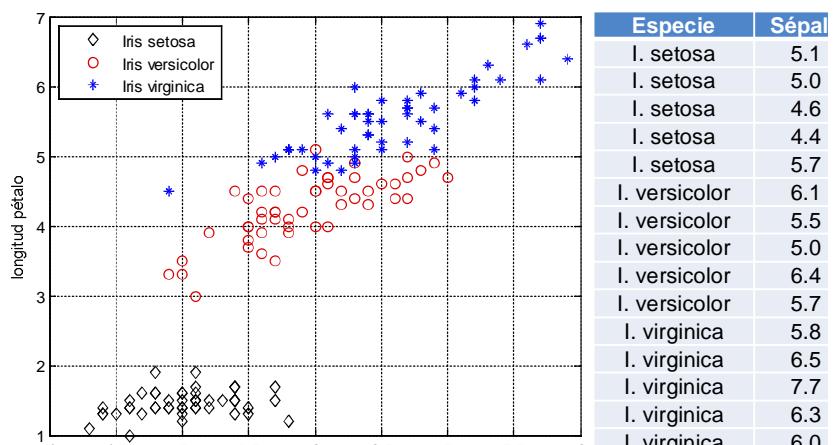
Luego no corresponde a ninguno de los 3 modelos

Notas

- Observa que si aplicamos ciegamente la regla de clasificación al objeto 2, nos sale que corresponde a la clase Cuadrado.
- La fórmula de Bayes nos daría que la probabilidad de que sea un cuadrado es: $p(\mathbf{x}|y_j) = 0,527$
 - Aunque la probabilidad salga muy alta, puede ser un espurio
 - ◆ Piensa por qué
 - ◆ Conviene comprobar siempre el test de chi-cuadrado
- Observa que el objeto 1 si que cumple el test de chi-cuadrado para la clase triángulo: $d_j^2 = 0,37$
- ¿Y el objeto 3?

Clasificación de Lirios (examen 13-6-20)

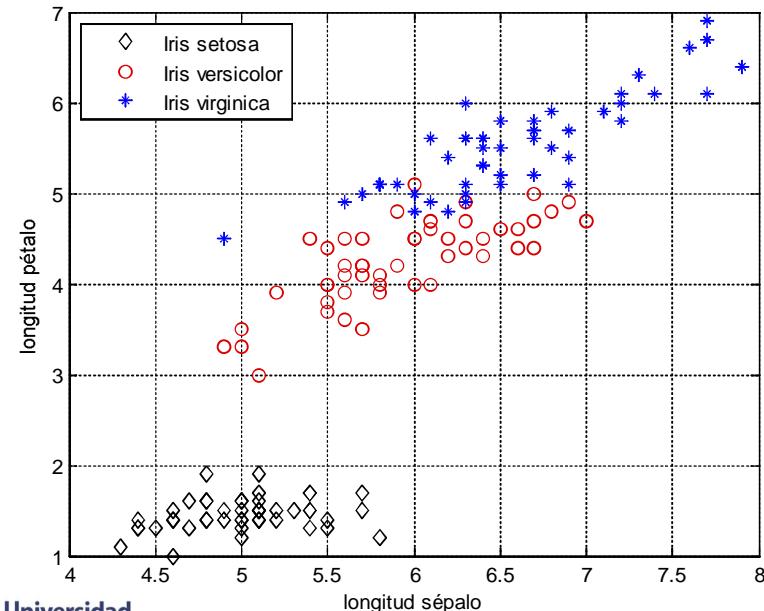
Queremos diseñar un clasificador para tres especies de lirio, a partir de las longitudes del sépalo y del pétalo de la flor. La figura muestra todos los datos de entrenamiento disponibles, y la tabla recoge 5 muestras de cada especie.



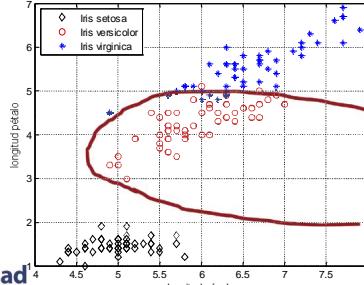
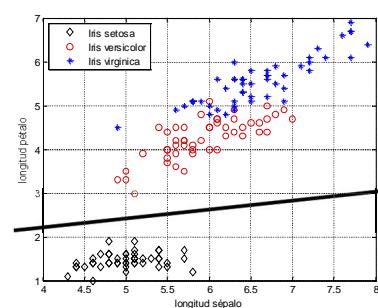
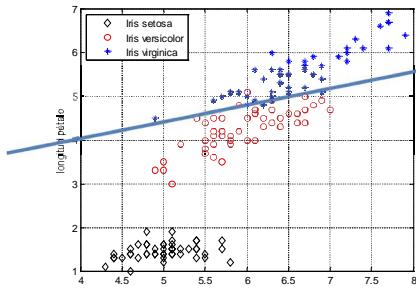
Se pide:

1. Explica si es adecuado diseñar el clasificador utilizando cada una de las siguientes técnicas, y cómo lo plantearías:
 - a) Regresión logística
 - b) Bayes ingenuo
 - c) Bayes completo
2. Escoge el clasificador que te parezca más adecuado y entrénalo con los datos de la tabla.
Nota: para abreviar, haz las cuentas solamente para I. virginica, y si utilizas un algoritmo iterativo, haz solamente una iteración.
3. Cuál sería el resultado del clasificador que has entrenado para un ejemplar que tuviera longitudes de sépalo = 5,5 y de pétalo = 4,7.
Nota: haz las cuentas solamente para I. virginica, y deja indicado cómo se obtendría el resultado final del clasificador.

¿Regresión Logística?: 1 .vs. all

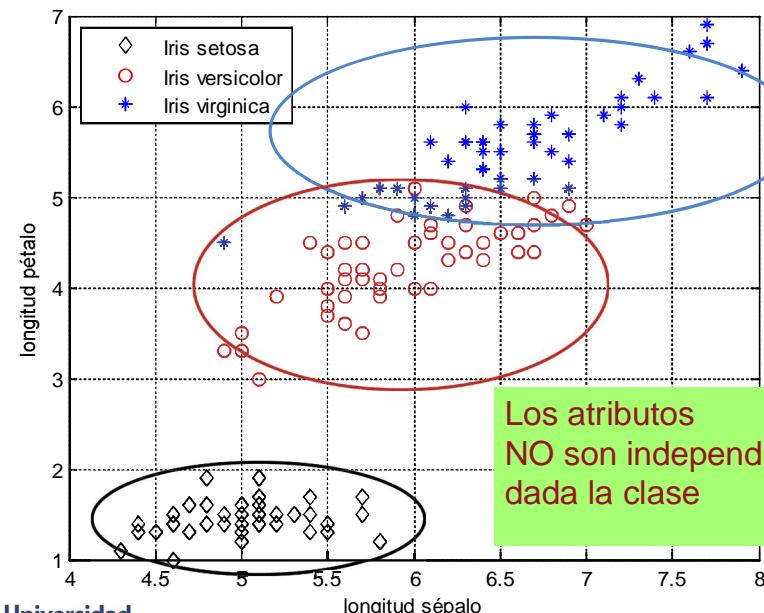


¿Regresión Logística?: 1 .vs. all

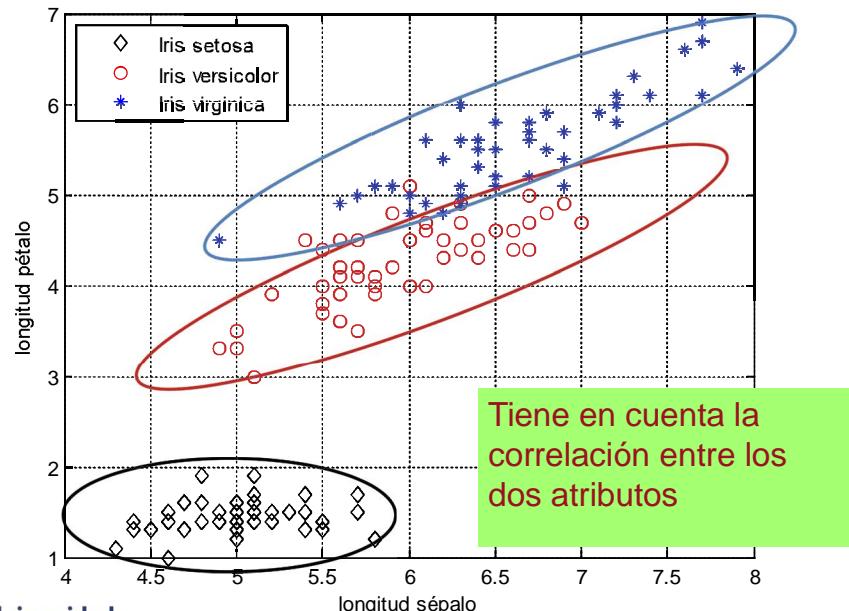


Se necesita expansión
de atributos:
 $LP, LS, LP^2, LS^2, LP^*LS$

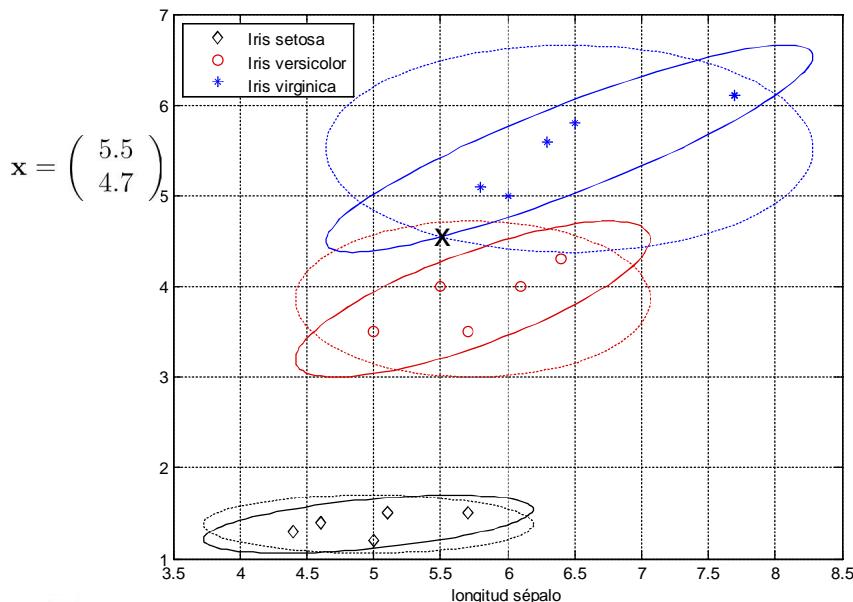
¿Bayes Ingenuo?



¡Bayes con Covarianza Completa!



Modelo Ingenuo .vs. Modelo Completo



Aprendizaje del Modelo Gaussiano

Especie	Sépalo	Pétalo	x1-m1	x2-m2	(x1-m1) ²	(x2-m2) ²	(x1-m1)(x2-m2)
I. setosa	5,1	1,5	0,14	0,12	0,0196	0,0144	0,0168
I. setosa	5	1,2	0,04	-0,18	0,0016	0,0324	-0,0072
I. setosa	4,6	1,4	-0,36	0,02	0,1296	0,0004	-0,0072
I. setosa	4,4	1,3	-0,56	-0,08	0,3136	0,0064	0,0448
I. setosa	5,7	1,5	0,74	0,12	0,5476	0,0144	0,0888
Media	4,96	1,38			Covarianza:	0,253	0,017
I. versicolor	6,1	4	0,36	0,14	0,1296	0,0196	0,0504
I. versicolor	5,5	4	-0,24	0,14	0,0576	0,0196	-0,0336
I. versicolor	5	3,5	-0,74	-0,36	0,5476	0,1296	0,2664
I. versicolor	6,4	4,3	0,66	0,44	0,4356	0,1936	0,2904
I. versicolor	5,7	3,5	-0,04	-0,36	0,0016	0,1296	0,0144
Media	5,74	3,86			Covarianza:	0,293	0,123
I. virginica	5,8	5,1	-0,66	-0,42	0,4356	0,1764	0,2772
I. virginica	6,5	5,8	0,04	0,28	0,0016	0,0784	0,0112
I. virginica	7,7	6,1	1,24	0,58	1,5376	0,3364	0,7192
I. virginica	6,3	5,6	-0,16	0,08	0,0256	0,0064	-0,0128
I. virginica	6	5	-0,46	-0,52	0,2116	0,2704	0,2392
Media	6,46	5,52			Covarianza:	0,553	0,217
							0,3085

■ Iris Virginica:

$$\mu = \begin{pmatrix} 6.46 \\ 5.52 \end{pmatrix}$$

$$\Sigma = \begin{pmatrix} 0.553 & 0.308 \\ 0.308 & 0.217 \end{pmatrix}$$

Clasificación Bayesiana

$$\hat{y} = \arg \max_j p(\mathbf{x}|y_j) p(y_j)$$

Suponiendo equiprobables

$$\ln(p(\mathbf{x}|y_j)) = -\frac{D}{2} \ln(2\pi) - \ln(|\Sigma_j|^{\frac{1}{2}}) - \frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_j)^T \boldsymbol{\Sigma}_j^{-1} (\mathbf{x} - \boldsymbol{\mu}_j)$$

$$\mathbf{x} = \begin{pmatrix} 5.5 \\ 4.7 \end{pmatrix}$$

$$\ln(p(\mathbf{x}|y_1)) = -423.7127$$

$$\boldsymbol{\mu}_3 = \begin{pmatrix} 6.46 \\ 5.52 \end{pmatrix} \quad \boldsymbol{\Sigma}_3 = \begin{pmatrix} 0.553 & 0.308 \\ 0.308 & 0.217 \end{pmatrix}$$

$$\ln(p(\mathbf{x}|y_2)) = -9.1814$$

$$\ln(p(\mathbf{x}|y_3)) = -1.7243$$

Es Iris Virginica