



REGRESIÓN LINEAL (CONCEPTOS BÁSICOS)

Dr. Jorge Hermosillo

Laboratorio de Semántica Computacional



FUNCIONES BASE DE CARACTERÍSTICAS (FEATURE BASIS FUNCTIONS)

Nociones básicas

MODELOS LINEALES DE REGRESIÓN

- **Regresión lineal:** combinación lineal de variables de entrada:

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1x_1 + \cdots + w_dx_d \text{ donde } \mathbf{x} = (x_1, \dots, x_d)^T$$

- **Regresión lineal con funciones base:** combinaciones lineales de funciones no-lineales de las variables de entrada:

$$y(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x})$$

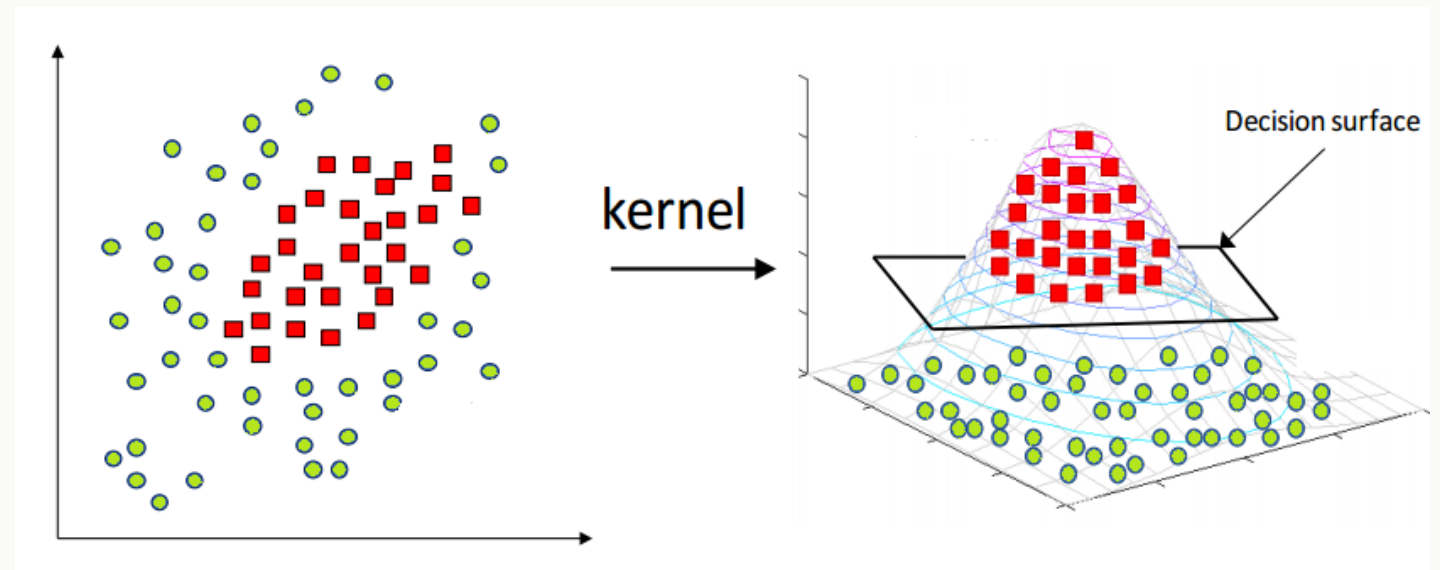
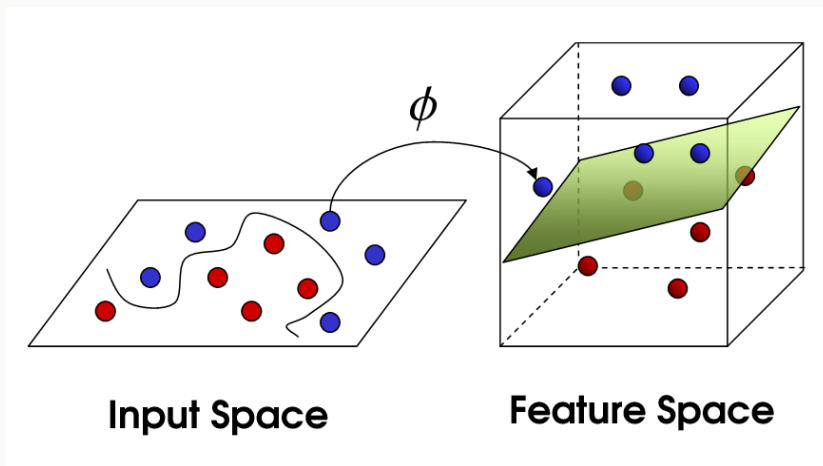
- Sea $\phi_0(\mathbf{x}) = 1$, de tal forma que:

$$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x})$$

donde $\mathbf{w} = (w_0, \dots, w_{M-1})^T$ y $\boldsymbol{\phi} = (\phi_0, \dots, \phi_{M-1})^T$

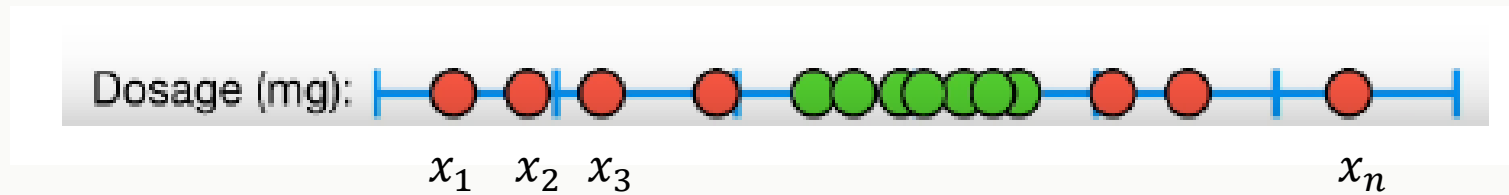
Efecto de las funciones base sobre los datos

- ▶ Permiten dar a los datos “textura”, incrementando el número de dimensiones.
- ▶ En tareas de clasificación, permiten encontrar un hiperplano que separe los datos

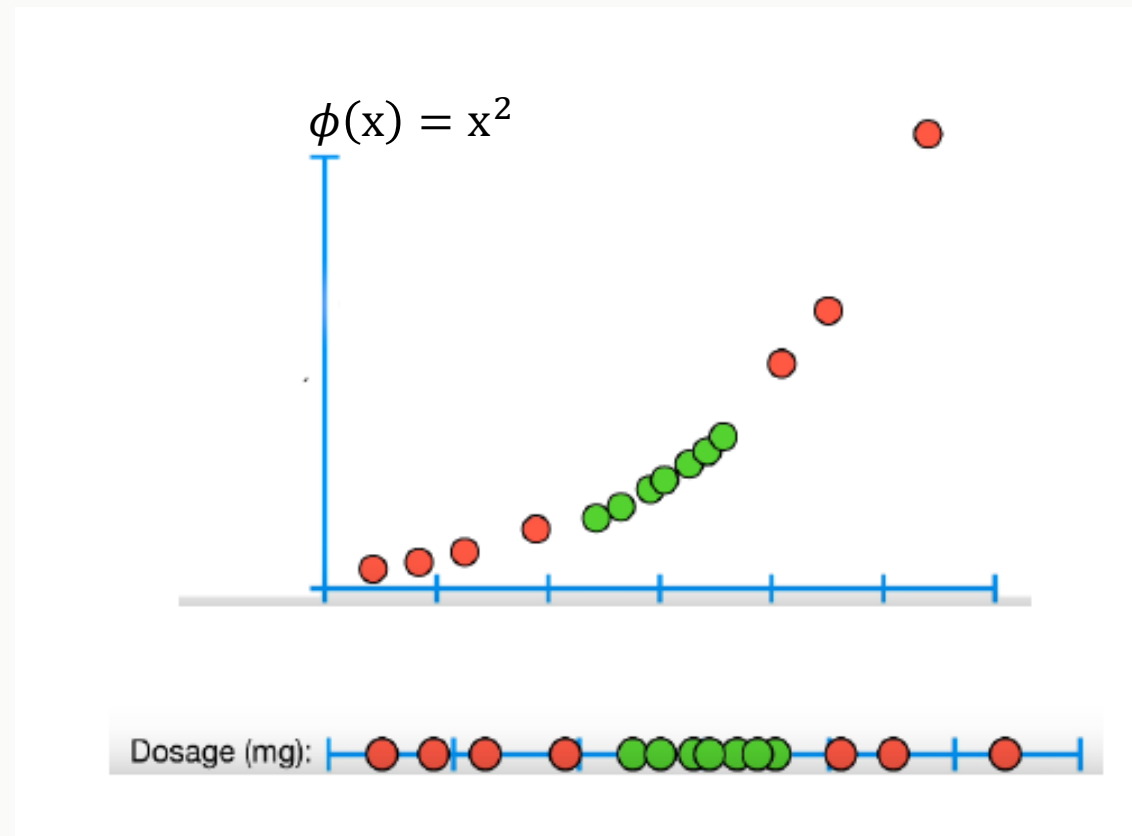


Ejemplo. Funciones base sobre datos unidimensionales

$$\mathbf{x} = \{x_1, x_2, \dots, x_n\}$$



Ejemplo. Funciones de características sobre datos unidimensionales



FEATURES (CARACTERÍSTICAS) COMUNES

- ▶ Si los datos originales están dados por el vector \mathbf{x} , entonces sus características están dadas por $\{\phi_j(\mathbf{x})\}$

- ▶ Funciones base Polinomiales (“*polynomial features*”):

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

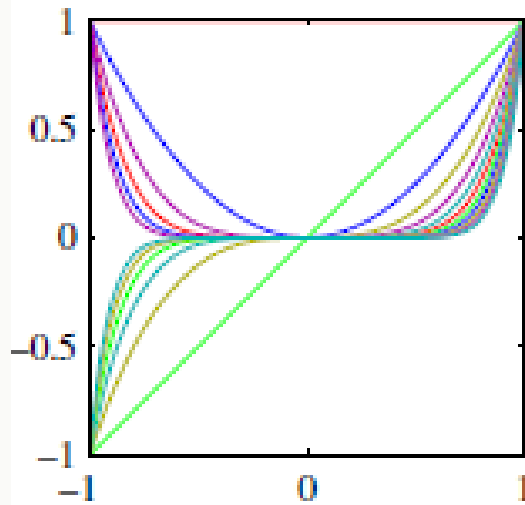
- ▶ Funciones base Gaussianas (“*Gaussian features*”)

$$\phi_j(\mathbf{x}) = \exp \left\{ -\frac{(\mathbf{x} - \mu_j)^2}{2s^2} \right\}$$

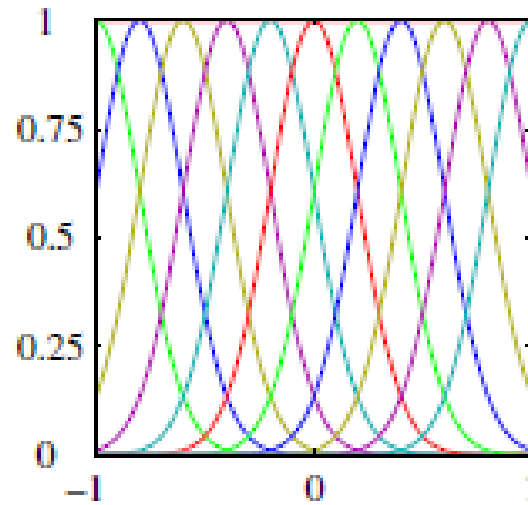
- ▶ Funciones base Sigmoidales (“*sigmoidal features*”)

$$\phi_j(\mathbf{x}) = \sigma \left(\frac{\mathbf{x} - \mu_j}{s} \right) \quad \text{donde} \quad \sigma(a) = \frac{1}{1 + \exp(-a)}$$

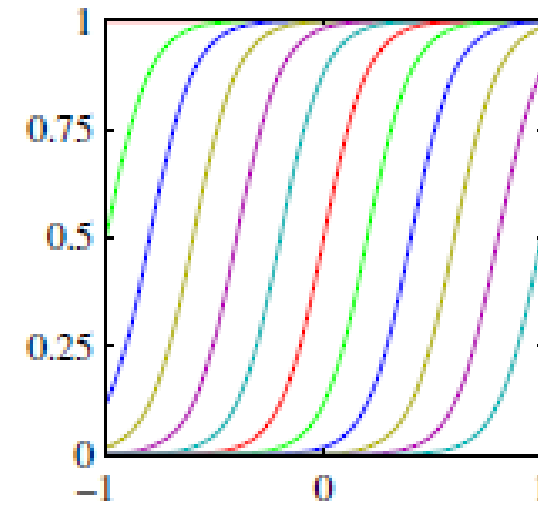
Distintas funciones de características



Polinomiales

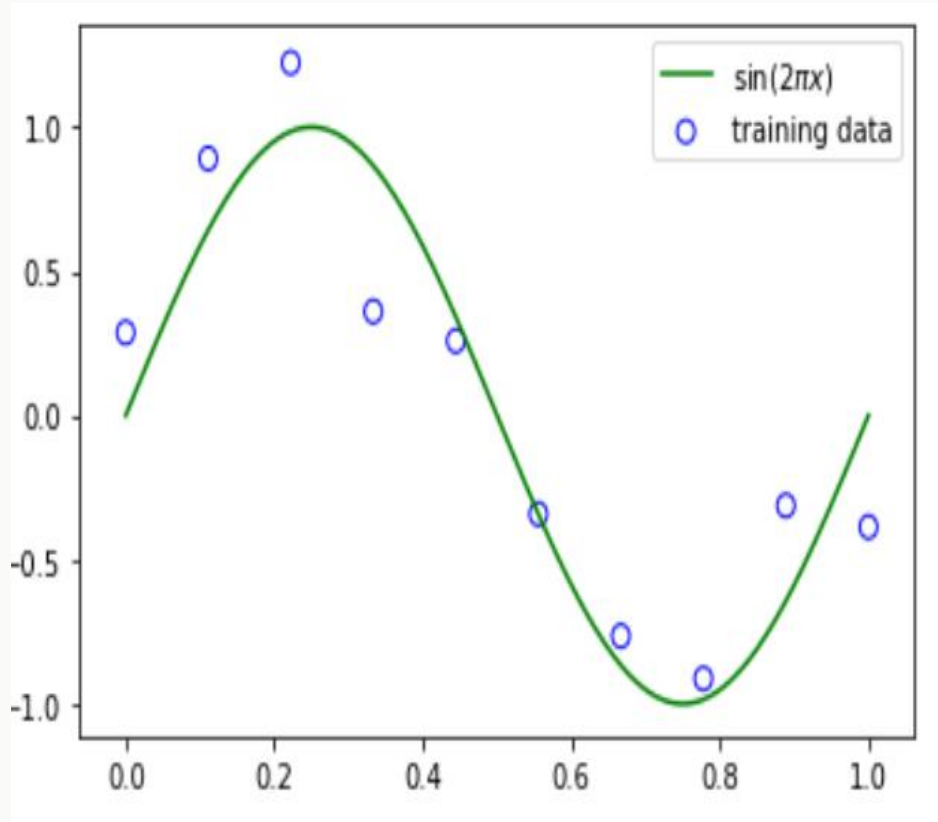


Gaussianas



Sigmoidales

EJEMPLO REGRESIÓN POLINOMIAL



DATOS DE ENTRADA:

Un conjunto de N datos de entrenamiento y sus respectivos valores objetivo:

$$\mathbf{x} \equiv (x_1, x_2, \dots, x_N)^T$$

$$\mathbf{t} \equiv (t_1, t_2, \dots, t_N)^T$$

donde:

$$t_n = \sin(2\pi x_n) + \epsilon$$

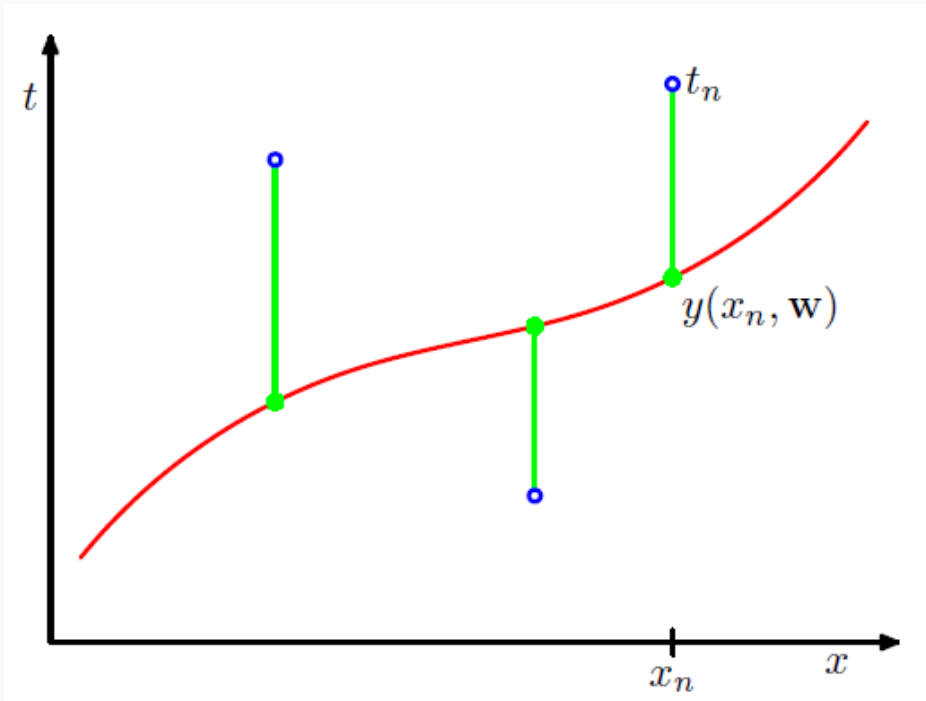
con $\epsilon \sim \mathcal{N}(0, (0.3)^2)$

SALIDA:

Un modelo lineal de ajuste polinomial:

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 \mathbf{x} + \dots + w_M \mathbf{x}^M = \sum_{j=0}^M w_j \mathbf{x}^j$$

FUNCIÓN DE ERROR/PÉRDIDA (LOSS FUNCTION)



$$y(x, \mathbf{w}) = w_0 + w_1x + \cdots + w_Mx^M = \sum_{j=0}^M w_jx^j$$

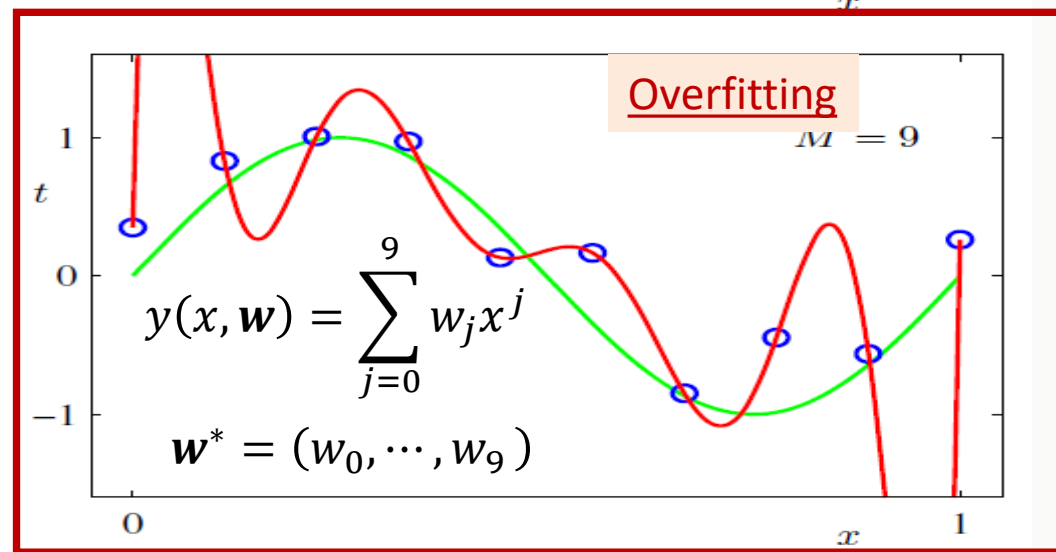
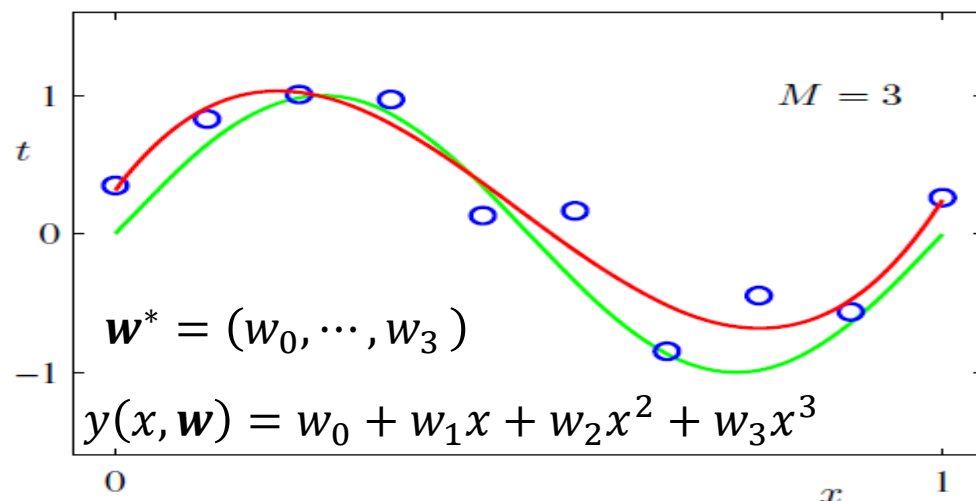
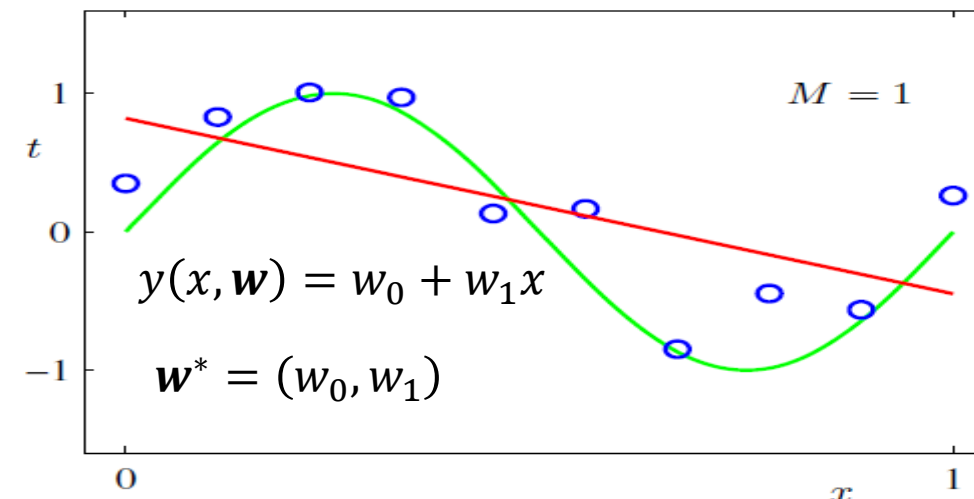
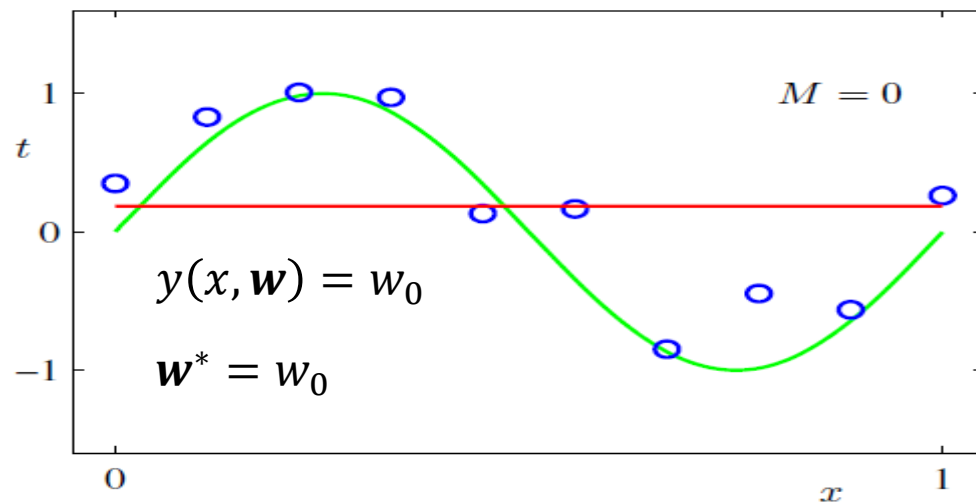
el ajuste se hace mediante la minimización de la función de error:

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N \{y(x_n, \mathbf{w}) - t_n\}^2$$

Donde t_n es el objetivo (valor de la muestra – puntos azules en la figura).

Tomada de: Bishop, Christopher M. (2006).
Pattern recognition and machine learning. New
York. Springer.

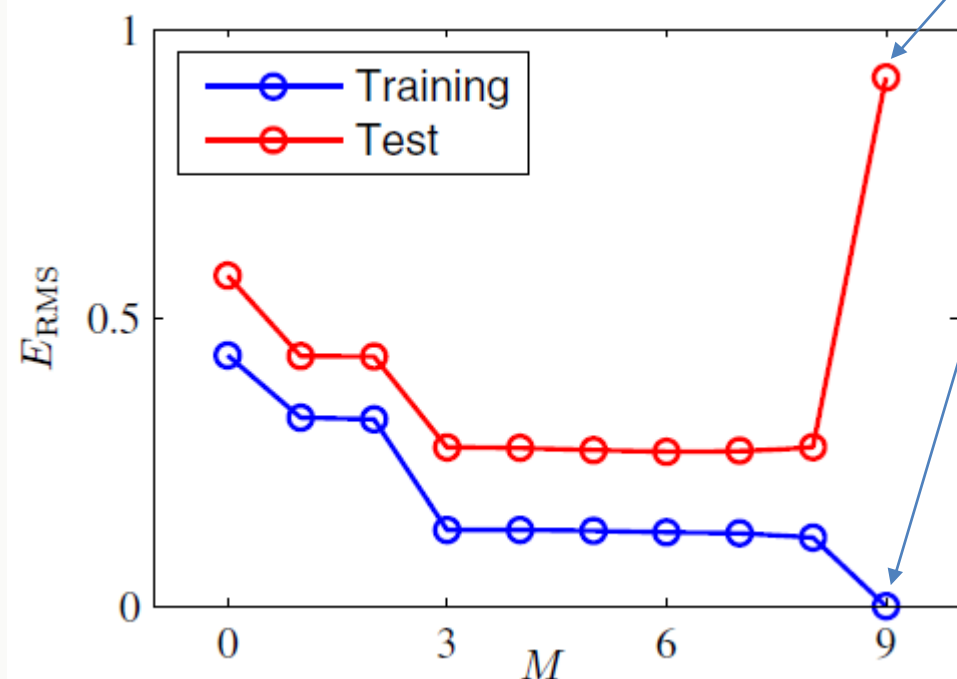
EFFECTO DE LA COMPLEJIDAD DEL MODELO EN EL AJUSTE



MÉTRICA DE DESEMPEÑO: E_{RMS}

Error cuadrático medio:

$$E_{\text{RMS}} = \sqrt{2E(\mathbf{w}^*)/N}$$

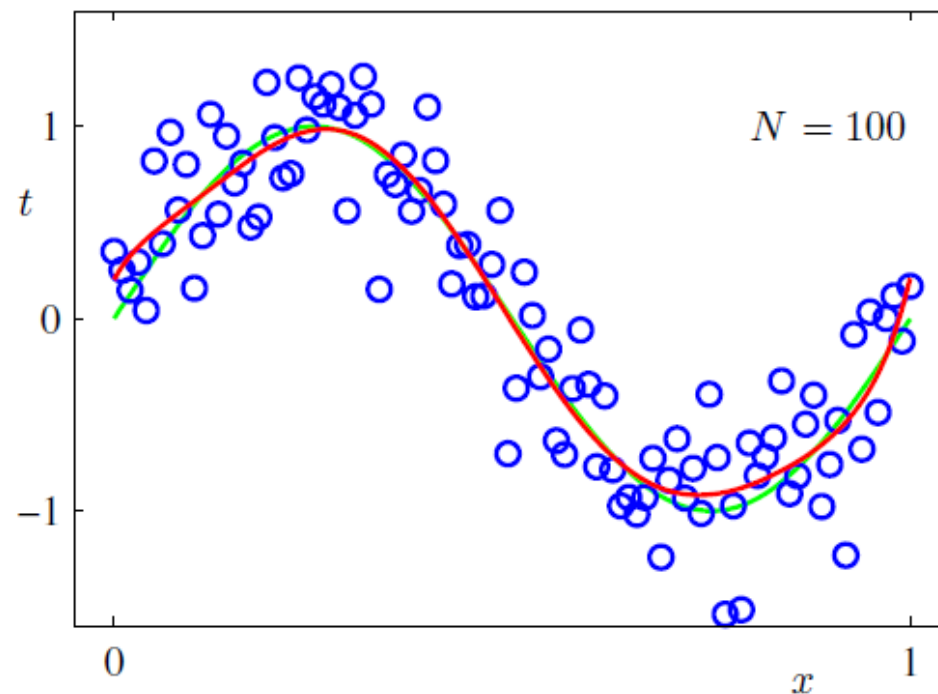
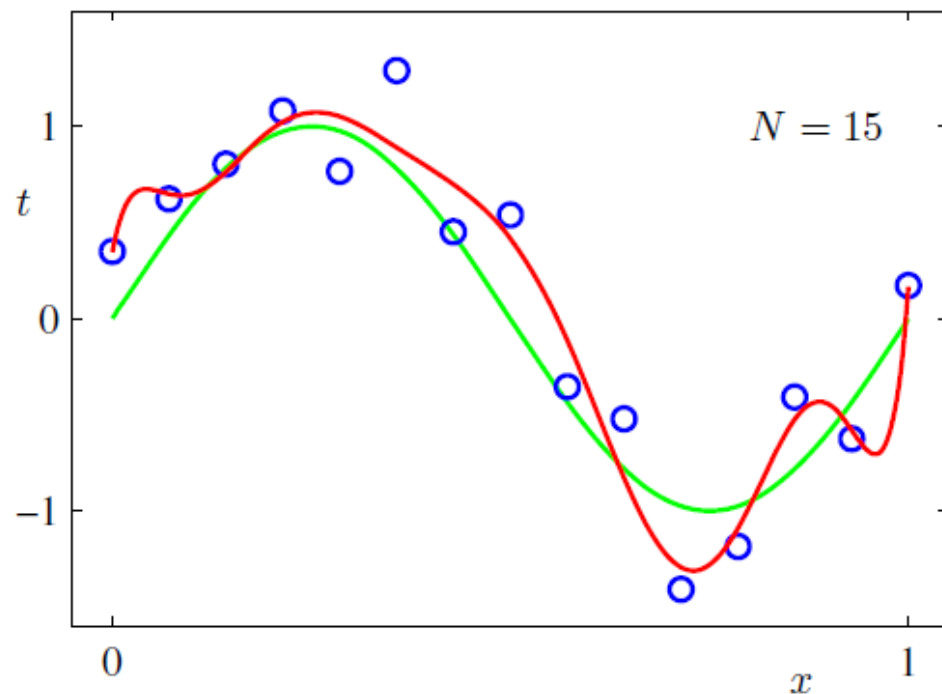


Overfitting

	$M = 0$	$M = 1$	$M = 6$	$M = 9$
w_0^*	0.19	0.82	0.31	0.35
w_1^*		-1.27	7.99	232.37
w_2^*			-25.43	-5321.83
w_3^*			17.37	48568.31
w_4^*				-231639.30
w_5^*				640042.26
w_6^*				-1061800.52
w_7^*				1042400.18
w_8^*				-557682.99
w_9^*				125201.43

Tomadas de: Bishop, Christopher M. (2006). Pattern recognition and machine learning. New York. Springer.

EFFECTO DE LA CANTIDAD DE DATOS N EN EL AJUSTE



$$y(x, \mathbf{w}) = \sum_{j=0}^9 w_j x^j$$

CONCLUSIONES

- ▶ El modelo es lineal en los parámetros, no en los datos. Esto es importante porque la optimización se hace con respecto a los parámetros.
- ▶ A mayor número de pesos, mayor complejidad -> mayor flexibilidad. Pero, modelos muy complejos son propensos al sobreajuste con pocos datos
- ▶ Con pocos datos, se requiere un modelo poco complejo, pero que resulta poco flexible ante el incremento en el número de datos.
- ▶ Deseamos un compromiso entre la complejidad del modelo (número de parámetros) y su flexibilidad (capacidad de ajuste).
- ▶ El sobreajuste se da cuando los parámetros escalan a valores muy altos.
- ▶ Idea: tratar de penalizar valores altos de los parámetros en función de la complejidad => **Regularización**

REGULARIZACIÓN

- La forma más simple es la penalización sobre la norma del vector de pesos (parámetros).

$$\tilde{E}(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N \{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{\lambda}{2} \|\mathbf{w}\|^2$$

Donde:

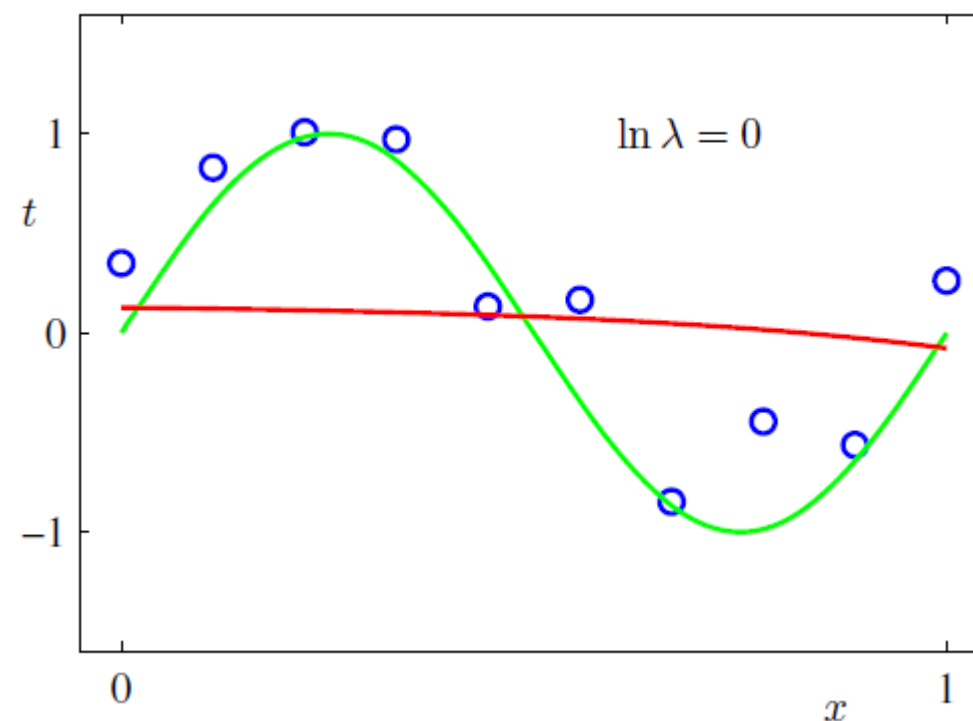
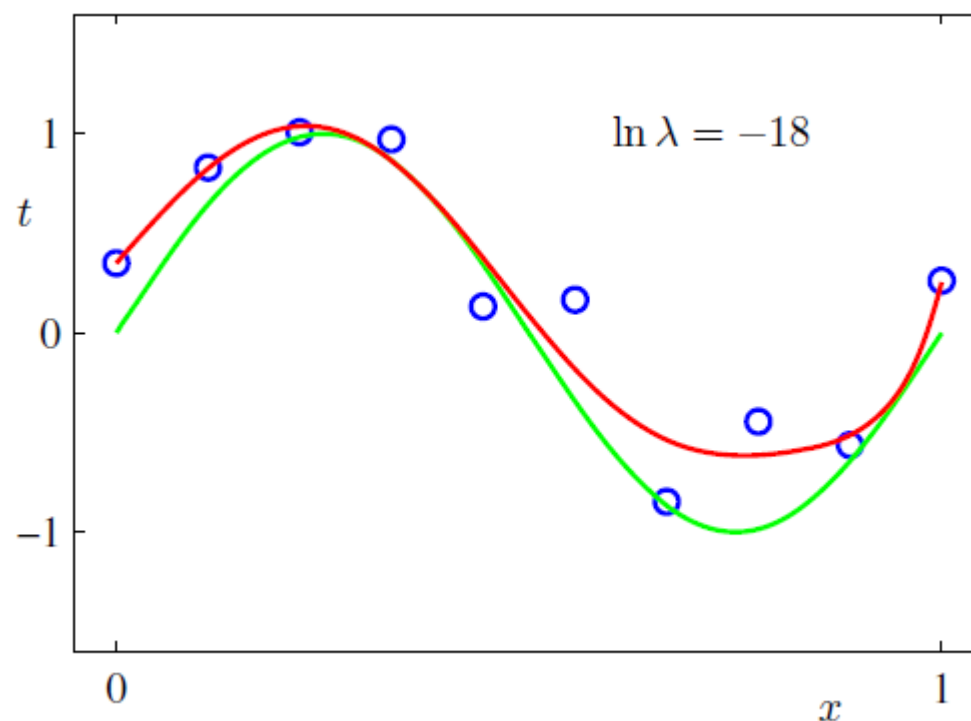
$$\|\mathbf{w}\|^2 \equiv \mathbf{w}^T \mathbf{w} = w_0^2 + w_1^2 + \dots + w_M^2$$

EFFECTO DE LA REGULARIZACIÓN EN LOS PARÁMETROS

Tabla de pesos w para $M = 9$ polinomios con varios valores para el parámetro de regularización λ . Nota que $\ln \lambda = -\infty$ corresponde a un modelo sin regularización. Vemos que, a medida que el valor de λ aumenta, la magnitud típica de los pesos disminuye.

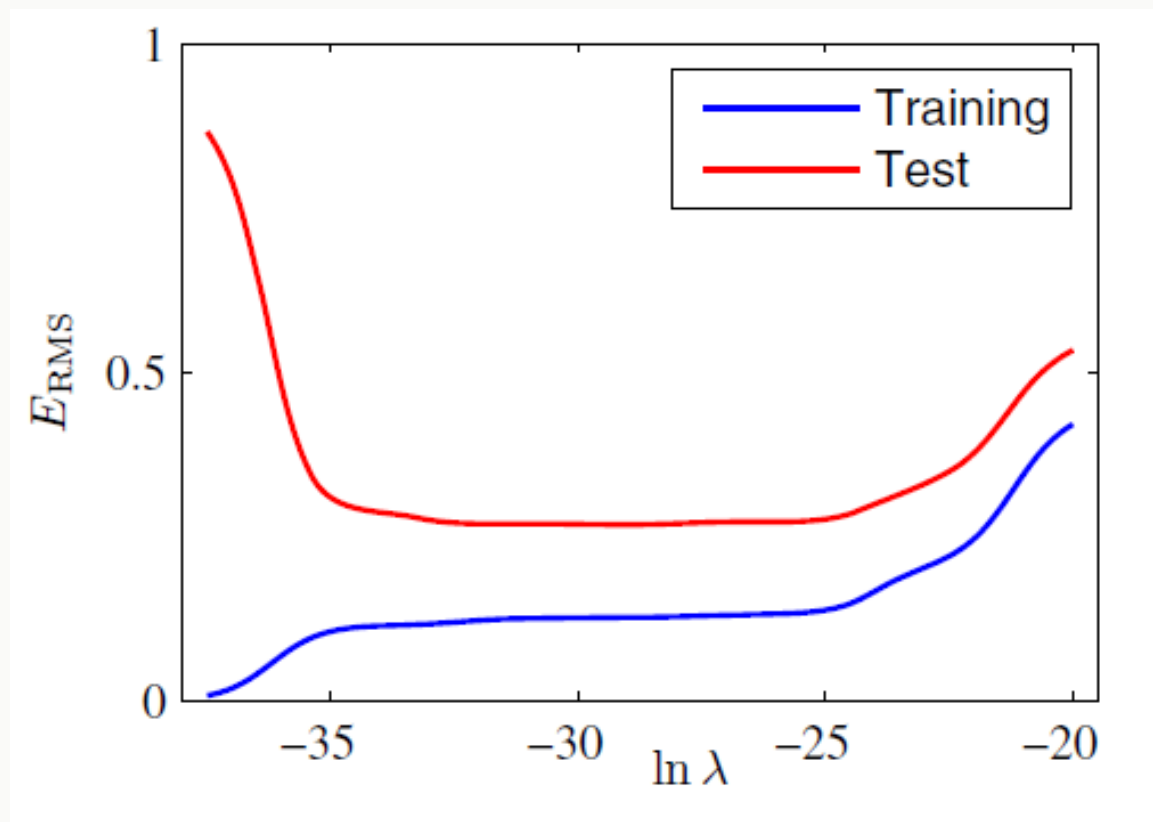
	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
w_0^*	0.35	0.35	0.13
w_1^*	232.37	4.74	-0.05
w_2^*	-5321.83	-0.77	-0.06
w_3^*	48568.31	-31.97	-0.05
w_4^*	-231639.30	-3.89	-0.03
w_5^*	640042.26	55.28	-0.02
w_6^*	-1061800.52	41.32	-0.01
w_7^*	1042400.18	-45.95	-0.00
w_8^*	-557682.99	-91.53	0.00
w_9^*	125201.43	72.68	0.01

EFECTO DE LA REGULARIZACIÓN EN EL AJUSTE



$M = 9$ con $\ln \lambda = -18$ y $\ln \lambda = 0$

EFFECTO DE LA REGULARIZACIÓN EN EL DESEMPEÑO



FUNCIONES DE PÉRDIDA (COSTO)

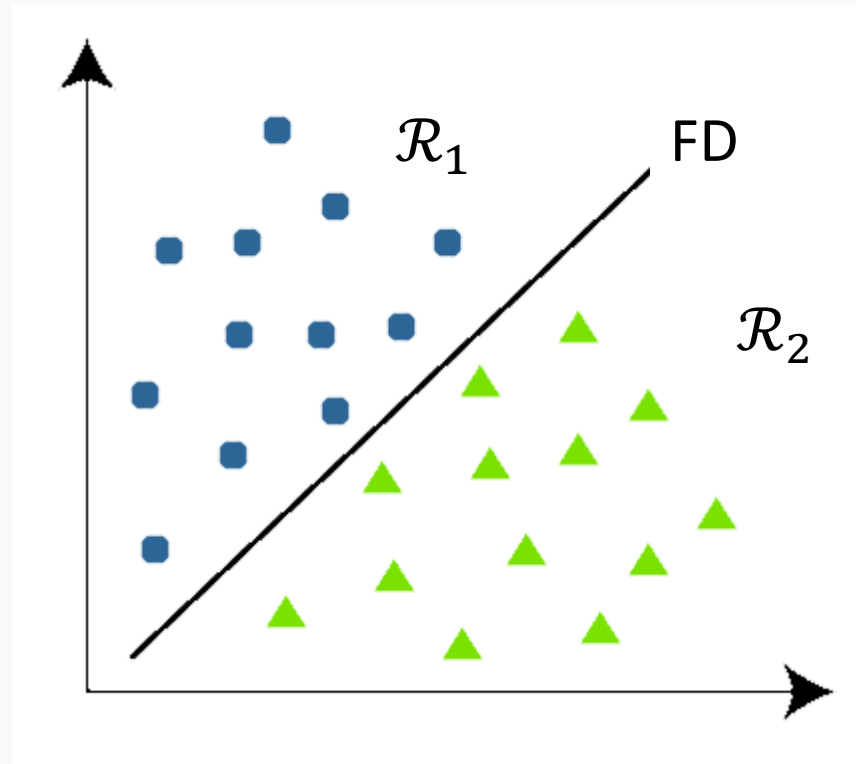
Nociones básicas

¿POR QUÉ FUNCIÓN DE “PÉRDIDA”?

- ▶ En ML, el propósito final se basa en minimizar o maximizar una función llamada "función objetivo".
- ▶ El grupo de funciones que se minimizan se denominan "funciones de pérdida".
- ▶ La función de pérdida se usa como medida de cuán bueno es un modelo de regresión/clasificación en términos de poder predecir el resultado esperado.

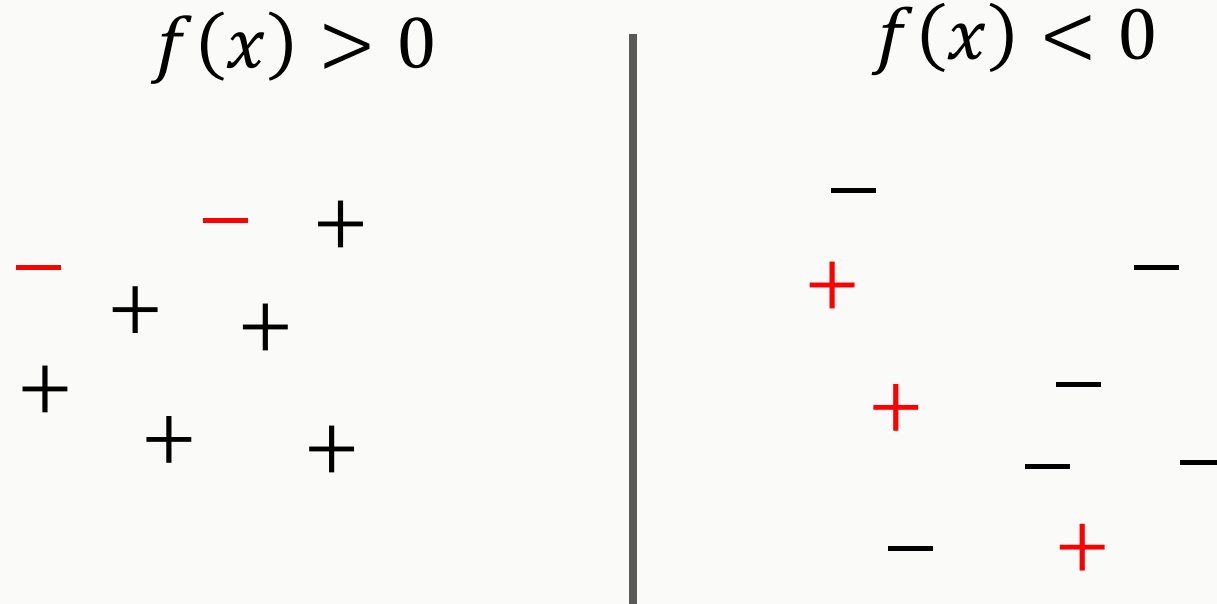
FUNCIÓN DE PÉRDIDA PARA CLASIFICACIÓN

- Un clasificador separa los datos de entrada en **regiones de decisión** cuyos límites llamamos **Fronteras (o superficies) de Decisión (FD)**.



FUNCIÓN DE PÉRDIDA PARA CLASIFICACIÓN

- Supongamos por ahora, que un punto está **bien clasificado** si $\text{sign}(f(x_i))$ es y_i , donde $f(\cdot)$ es la función de clasificación y $y_i := \{-1, +1\}$.



Fracción de veces que $\text{sign}(f(x_i))$ no es y_i :

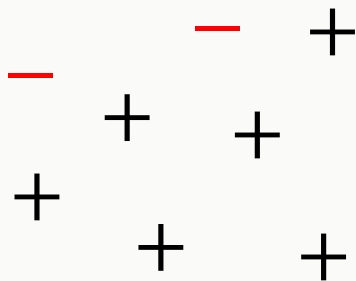
$$\frac{1}{n} \sum_{i=1}^n y_i \neq \text{sign}(f(x_i))$$

Minimizar esta función es computacionalmente muy difícil.

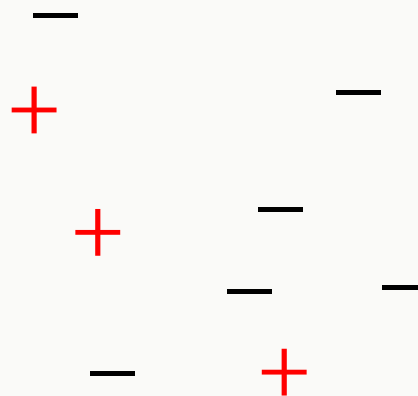
FUNCIÓN DE PÉRDIDA PARA CLASIFICACIÓN

- Supongamos por ahora, que un punto está **bien clasificado** si $\text{sign}(f(x_i))$ es y_i , donde $f(\cdot)$ es la función de clasificación y $y_i := \{-1, +1\}$.

$$yf(x) \approx 0$$



$$yf(x) \approx 0$$

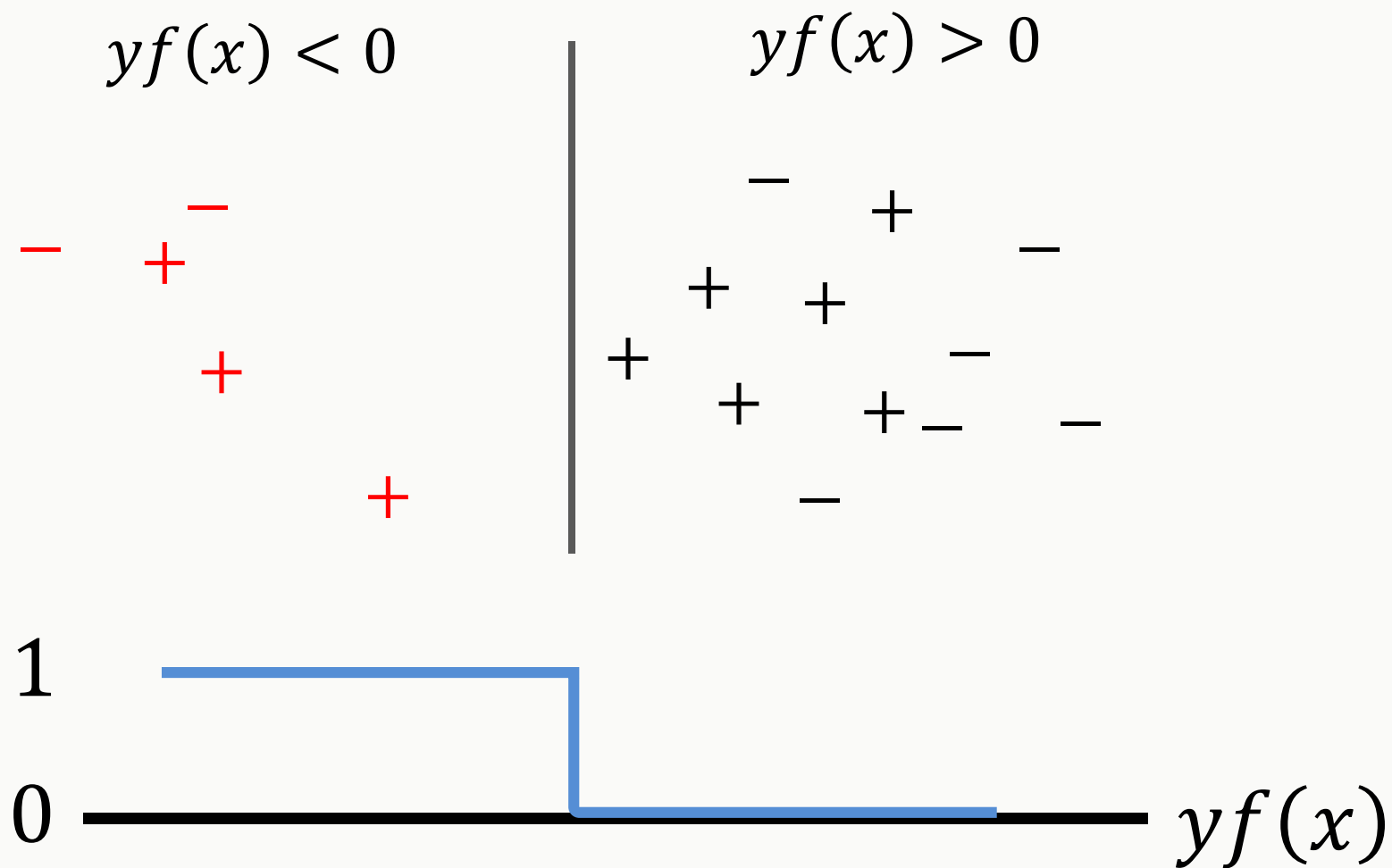


Fracción de veces que $\text{sign}(f(x_i))$ no es y_i :

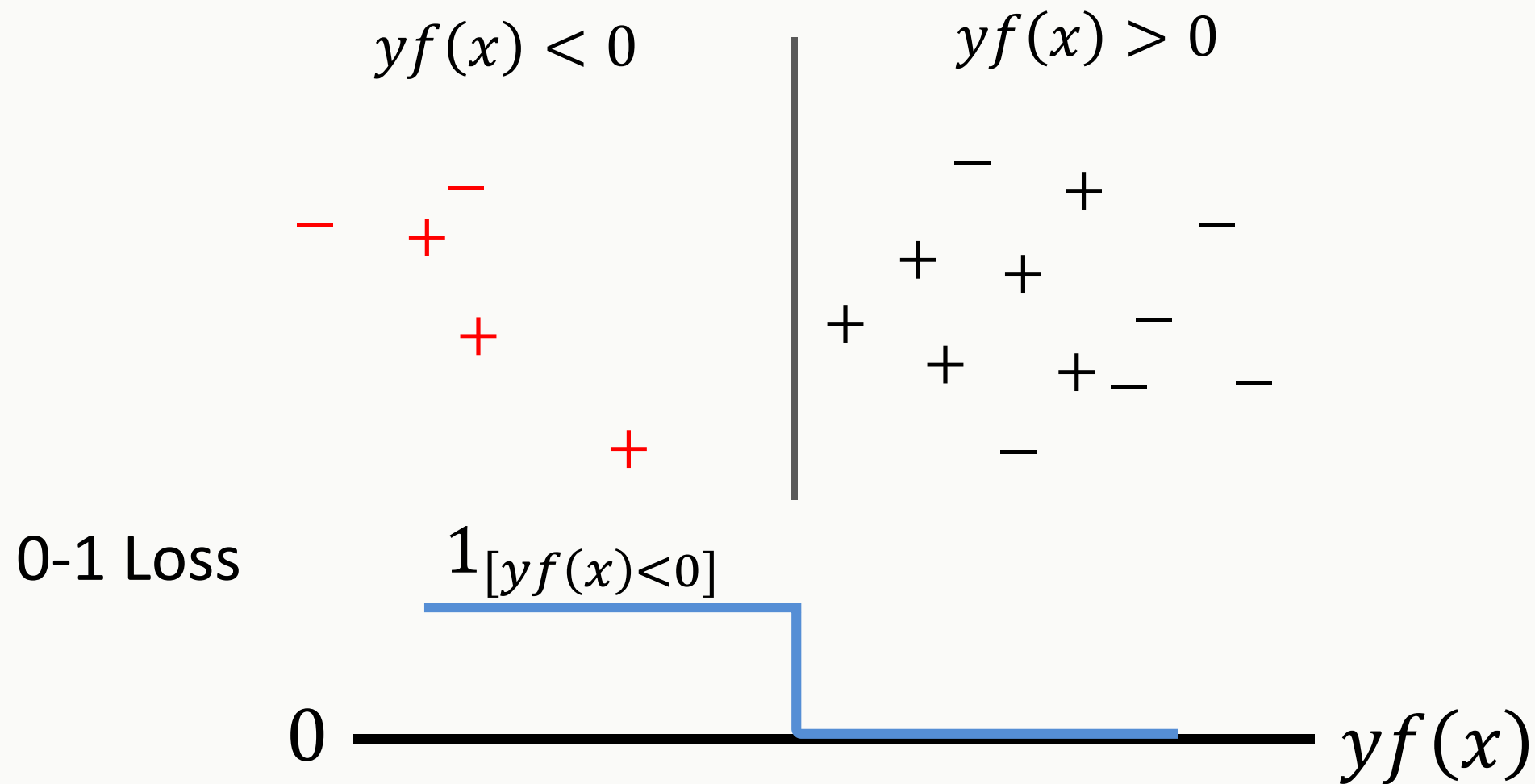
$$\frac{1}{n} \sum_{i=1}^n y_i \neq \text{sign}(f(x_i))$$

Minimizar esta función es computacionalmente muy difícil.

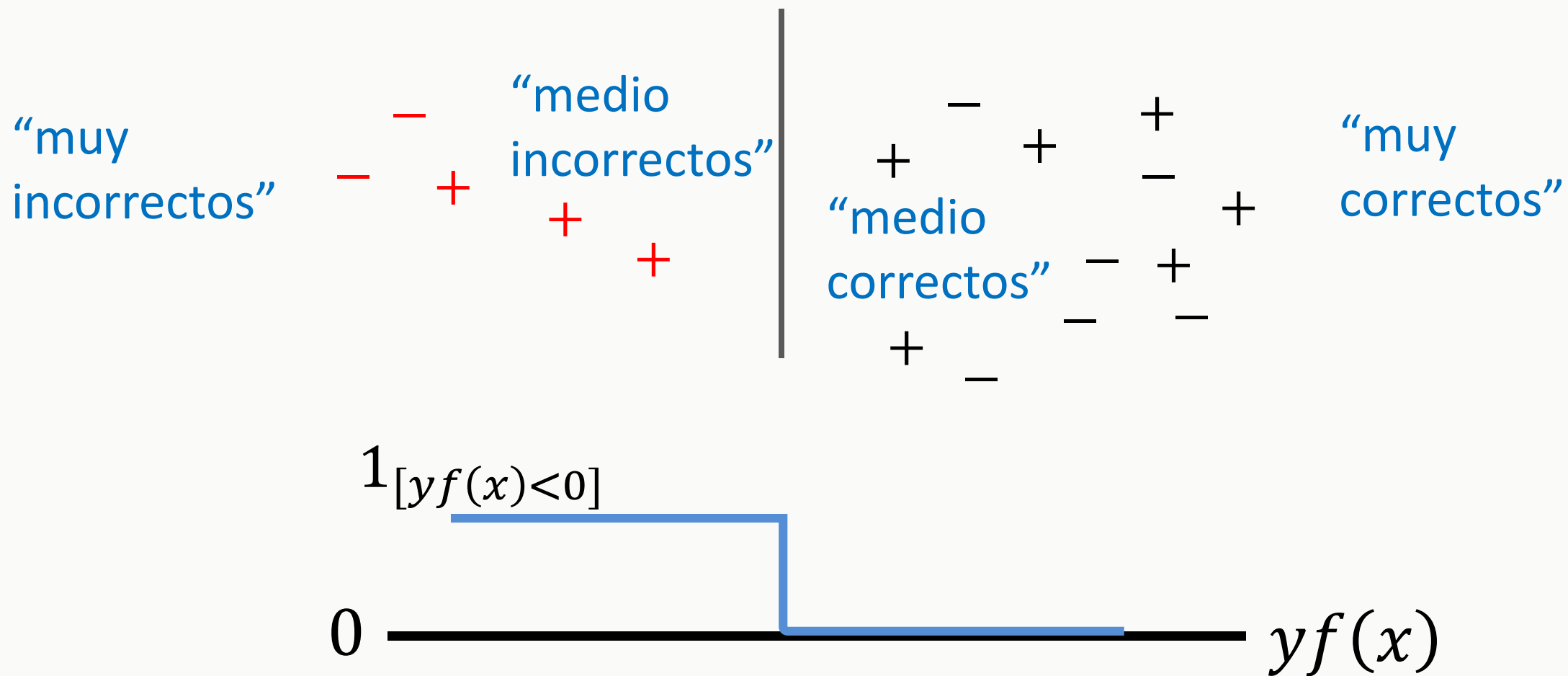
FUNCIÓN DE PÉRDIDA PARA CLASIFICACIÓN



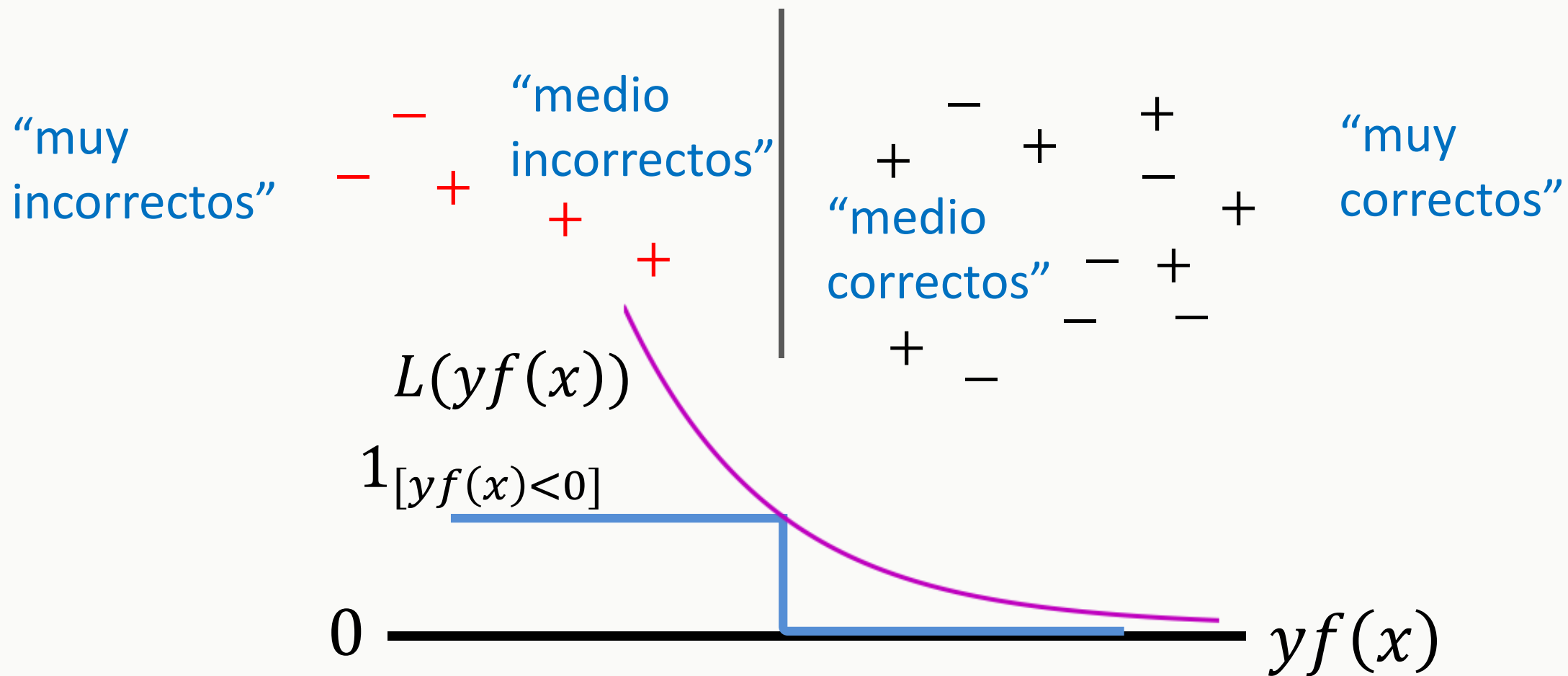
FUNCIÓN DE PÉRDIDA PARA CLASIFICACIÓN



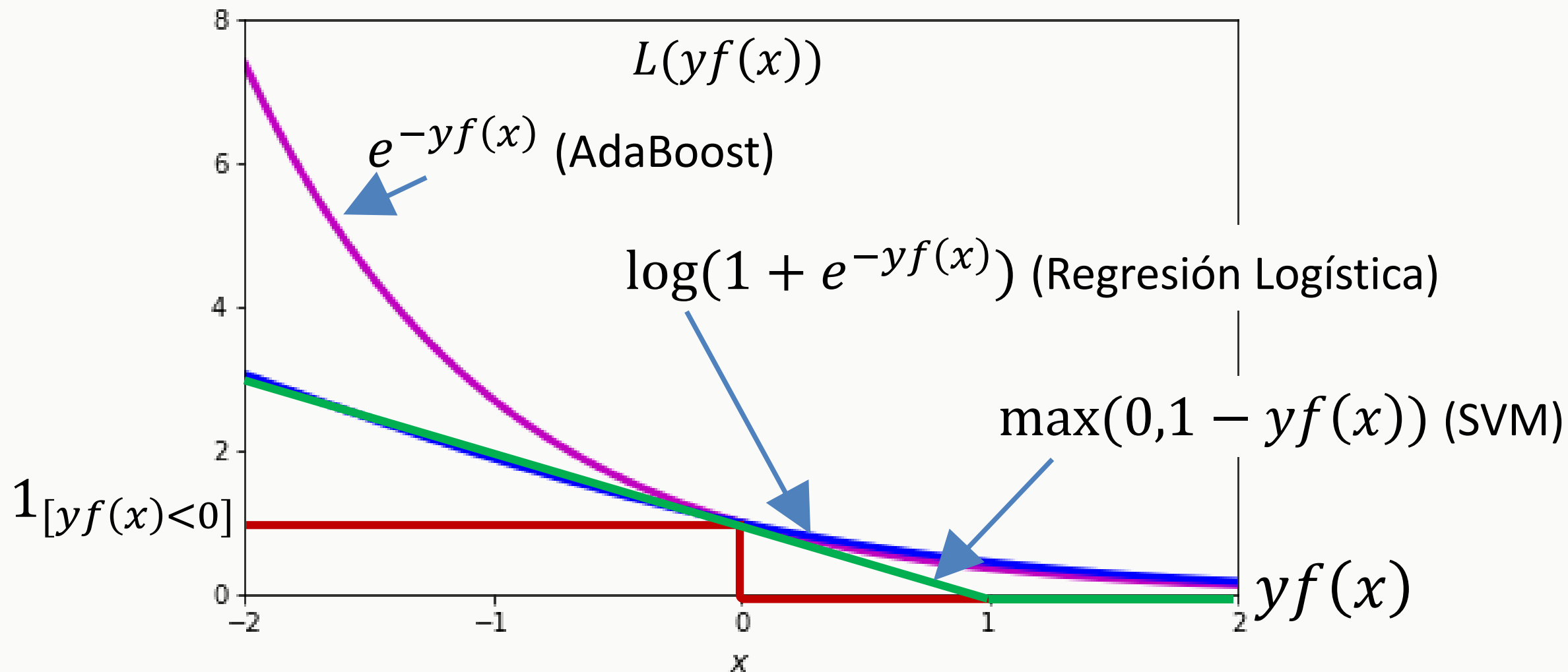
FUNCIÓN DE PÉRDIDA PARA CLASIFICACIÓN



FUNCIÓN DE PÉRDIDA PARA CLASIFICACIÓN



FUNCIÓN DE PÉRDIDA PARA CLASIFICACIÓN



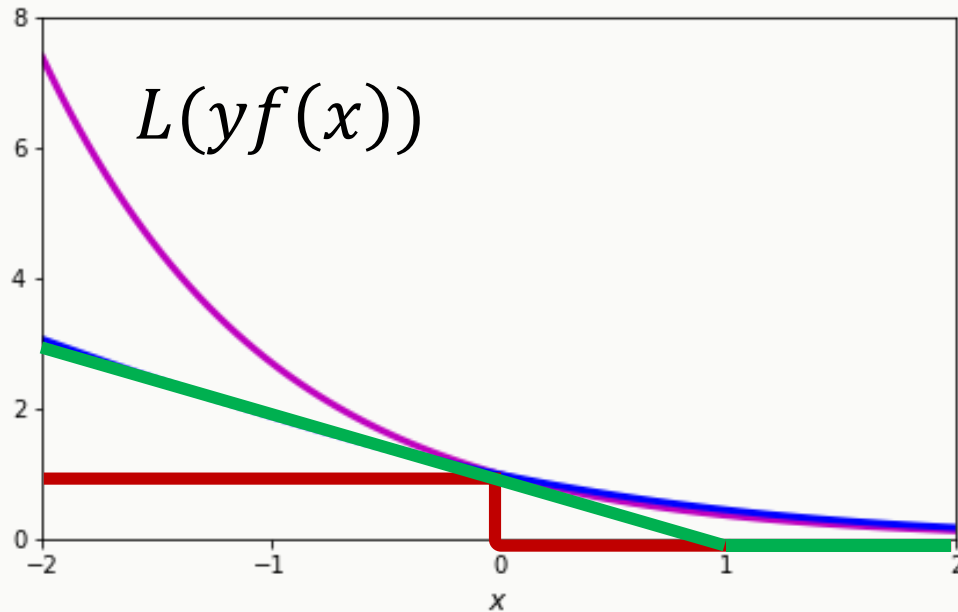
FUNCIÓN DE PÉRDIDA PARA CLASIFICACIÓN

Fracción de veces que $\text{sign}(f(x_i))$ no es y_i

$$= \frac{1}{n} \sum_{i=1}^n 1_{[y_i \neq \text{sign}(f(x_i))]}$$

$$= \frac{1}{n} \sum_{i=1}^n 1_{[y_i f(x_i) < 0]}$$

$$\leq \frac{1}{n} \sum_{i=1}^n L(y_i f(x_i))$$



FUNCIÓN DE PÉRDIDA PARA CLASIFICACIÓN

Fracción de veces que $\text{sign}(f(x_i))$ no es y_i $= \frac{1}{n} \sum_{i=1}^n 1_{[y_i \neq \text{sign}(f(x_i))]}$

Objetivo

$$\min_{\text{modelos } f} \frac{1}{n} \sum_{i=1}^n L(y_i f(x_i))$$

$$= \frac{1}{n} \sum_{i=1}^n 1_{[y_i f(x_i) < 0]}$$

$$\leq \frac{1}{n} \sum_{i=1}^n L(y_i f(x_i))$$

DESCENSO DE GRADIENTE

Nociones básicas

Ejemplo de problema

**Conjunto de
entrenamiento** de precios
de casas (Portland, Oregon,
USA)

Tamaño en pies ² (x)	Precio (\$) en miles (y)
2104	460
1416	232
1534	315
852	178
...	...

Modelación del problema (I)

**Conjunto de
entrenamiento** de precios
de casas (Portland, Oregon,
USA)

N=47

Tamaño en pies (x)	Precio (\$) en miles (y)
2104	460
1416	232
1534	315
852	178
...	...

Notación:

N : número de instancias (ejemplos) de entrenamiento

x : variable de “entrada” / **atributos** (*features*)

y : variable de “salida” / variable “objetivo”

Modelación del problema (II)

**Conjunto de
entrenamiento** de precios
de casas (Portland, Oregon,
USA)

	Tamaño en pies ² (x)	Precio (\$) en miles (y)
N=47	2104 $x^{(1)}$	460 $y^{(1)}$
	1416 $x^{(2)}$	232
	1534	315
	852	178

Notación:

N : número de instancias (ejemplos) de entrenamiento

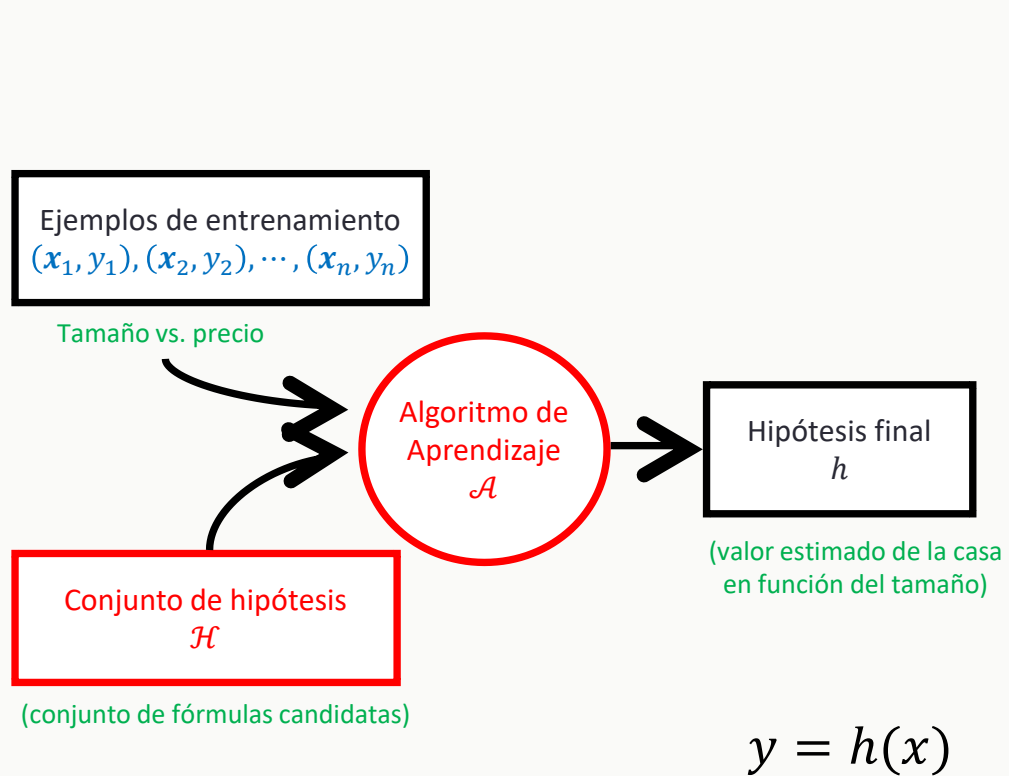
x : variable de “entrada” / **atributos** (*features*)

y : variable de “salida” / variable “objetivo”

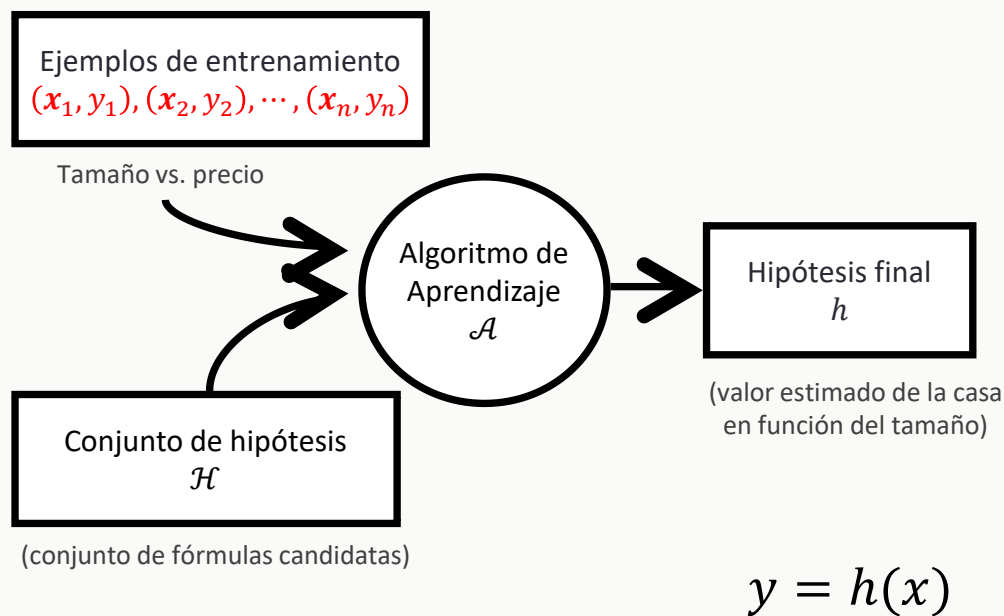
(x, y) – un ejemplo de
entrenamiento

$(x^{(i)}, y^{(i)})$ – i-ésimo ejemplo de
entrenamiento

Problema de Aprendizaje

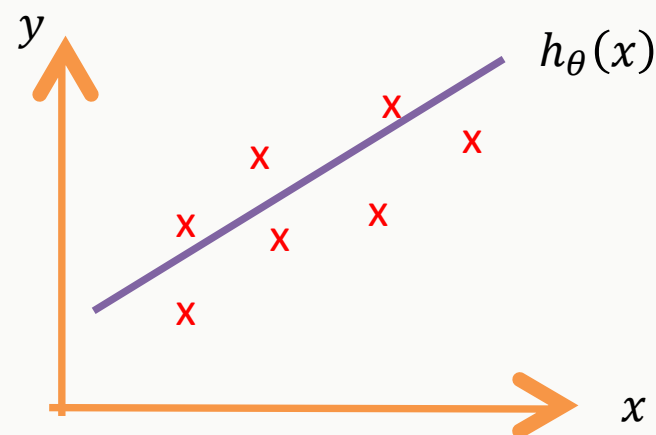


Problema de Aprendizaje



¿Cómo representamos h ?

$$h_{\theta}(x) = \theta_0 + \theta_1 x$$



Regresión lineal de una variable (univariada).

FUNCIÓN DE COSTO

Problema de aprendizaje

Conjunto de entrenamiento

	Tamaño en pies ² (x)	Precio (\$) en miles (y)
N=47	2104	460
	1416	232
	1534	315
	852	178

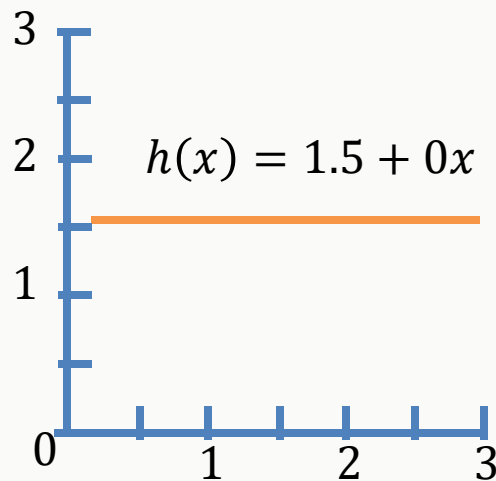
Hipótesis: $h_{\theta}(x) = \theta_0 + \theta_1 x$

θ_i 's : Parámetros

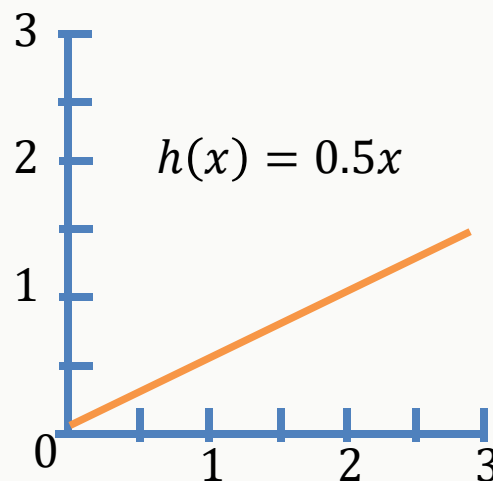
¿Cómo elegir los θ_i 's?

Hipótesis en función de los parámetros

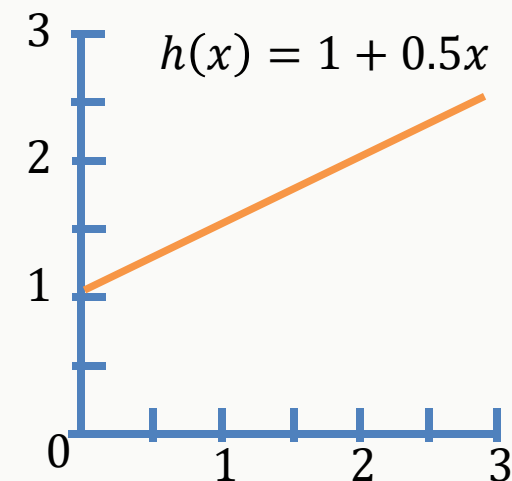
$$h_{\theta}(x) = \theta_0 + \theta_1 x$$



$$\begin{aligned}\theta_0 &= 1.5 \\ \theta_1 &= 0\end{aligned}$$

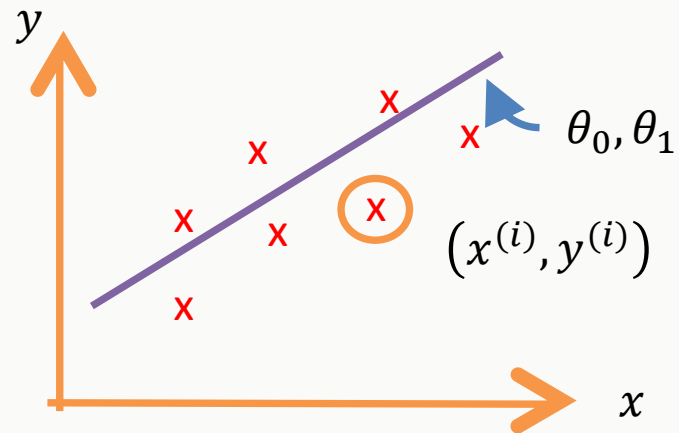


$$\begin{aligned}\theta_0 &= 0 \\ \theta_1 &= 0.5\end{aligned}$$



$$\begin{aligned}\theta_0 &= 1 \\ \theta_1 &= 0.5\end{aligned}$$

Función de costo

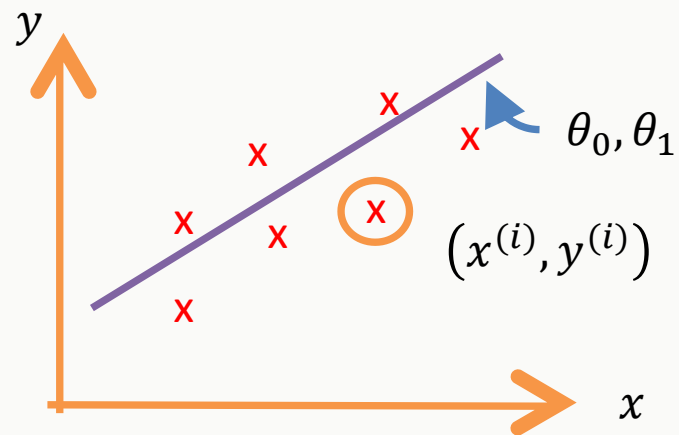


Idea: Elige θ_0, θ_1 de tal forma que $h_\theta(x)$ esté cerca de y para nuestros ejemplos de entrenamiento (x, y)

Función de error cuadrático:

$$\Rightarrow J(\theta_0, \theta_1) = \frac{1}{2N} \sum_{i=1}^N \underbrace{(h_\theta(x^{(i)}) - y^{(i)})^2}_{h_\theta(x^{(i)}) = \theta_0 + \theta_1 x^{(i)}}$$

Minimización de costo



Idea: Elige θ_0, θ_1 de tal forma que $h_\theta(x)$ esté cerca de y para nuestros ejemplos de entrenamiento (x, y)

$$J(\theta_0, \theta_1) = \frac{1}{2N} \sum_{i=1}^N (h_\theta(x^{(i)}) - y^{(i)})^2$$

Función de costo

$$\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$$

$$\min_{\theta_0, \theta_1} \frac{1}{2N} \sum_{i=1}^N (h_\theta(x^{(i)}) - y^{(i)})^2$$

$$h_\theta(x^{(i)}) = \theta_0 + \theta_1 x^{(i)}$$

FUNCIÓN DE COSTO

Intuición I

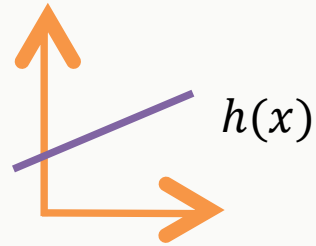
Problema de minimización

Hipótesis:

$$h_{\theta}(x) = \theta_0 + \theta_1 x$$

Parámetros:

$$\theta_0, \theta_1$$



Función de costo:

$$J(\theta_0, \theta_1) = \frac{1}{2N} \sum_{i=1}^N (h_{\theta}(x^{(i)}) - y^{(i)})^2$$

Objetivo: $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$

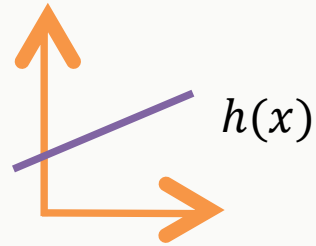
Minimización simplificada

Hipótesis:

$$h_{\theta}(x) = \theta_0 + \theta_1 x$$

Parámetros:

$$\theta_0, \theta_1$$



Función de costo:

$$J(\theta_0, \theta_1) = \frac{1}{2N} \sum_{i=1}^N (h_{\theta}(x^{(i)}) - y^{(i)})^2$$

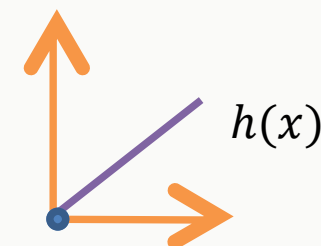
Objetivo: $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$

Simplificación:

$$h_{\theta}(x) = \theta_1 x$$

Parámetros:

$$\theta_1$$

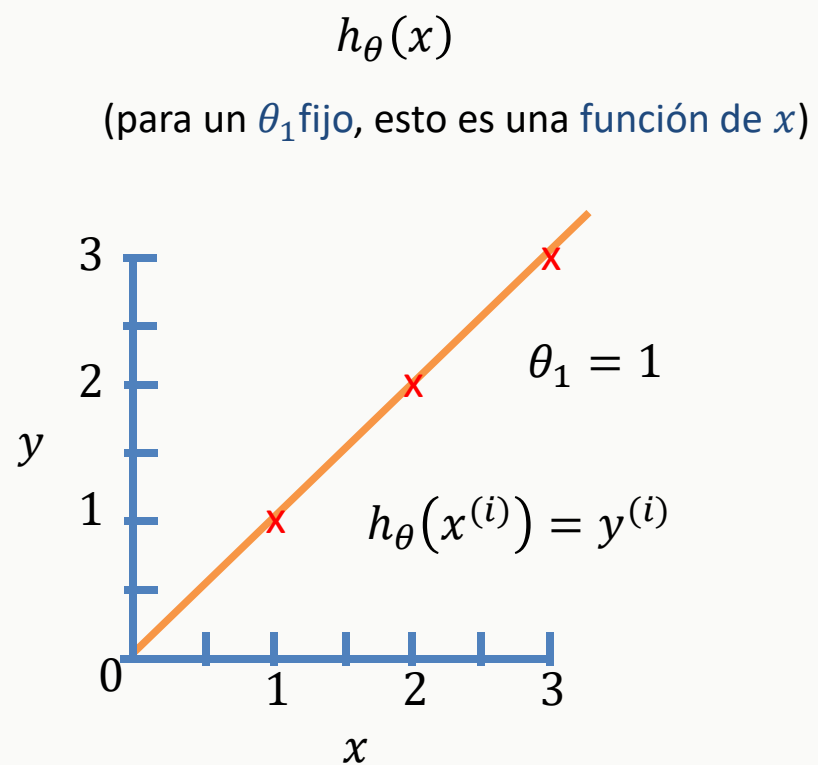


Función de costo:

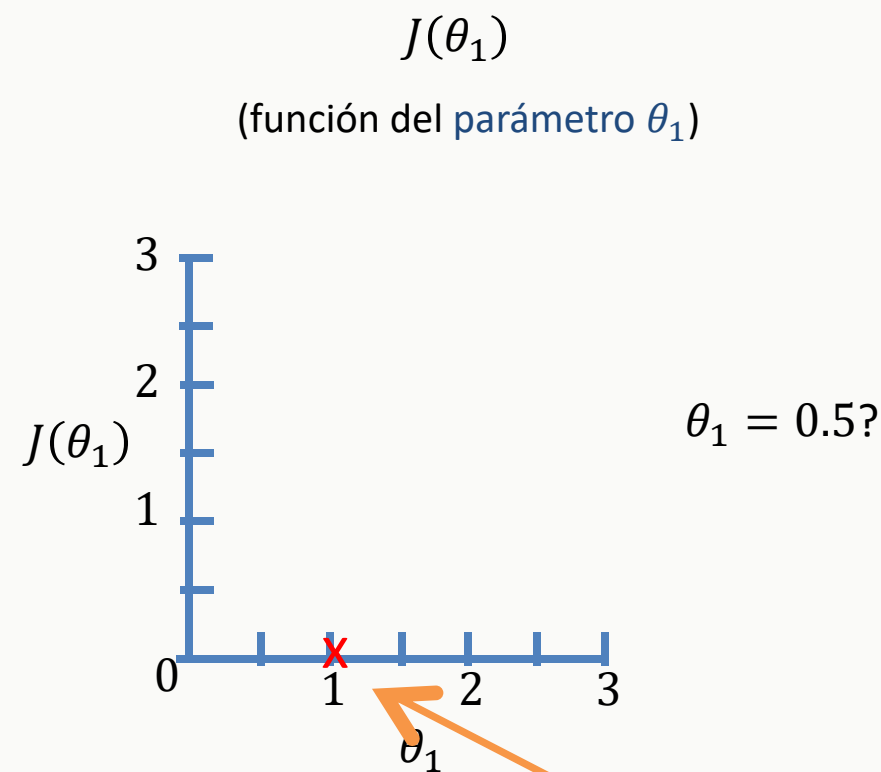
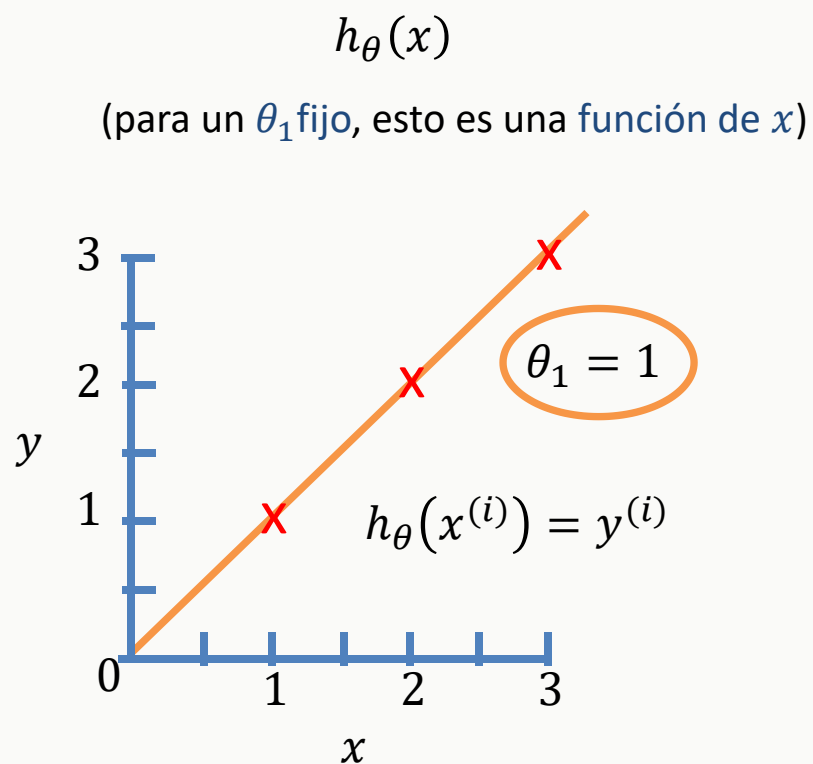
$$J(\theta_1) = \frac{1}{2N} \sum_{i=1}^N \underbrace{(h_{\theta}(x^{(i)}) - y^{(i)})^2}_{\theta_1 x^{(i)}}$$

Objetivo: $\min_{\theta_1} J(\theta_1)$

Hipótesis vs. Costo



Hipótesis vs. Costo

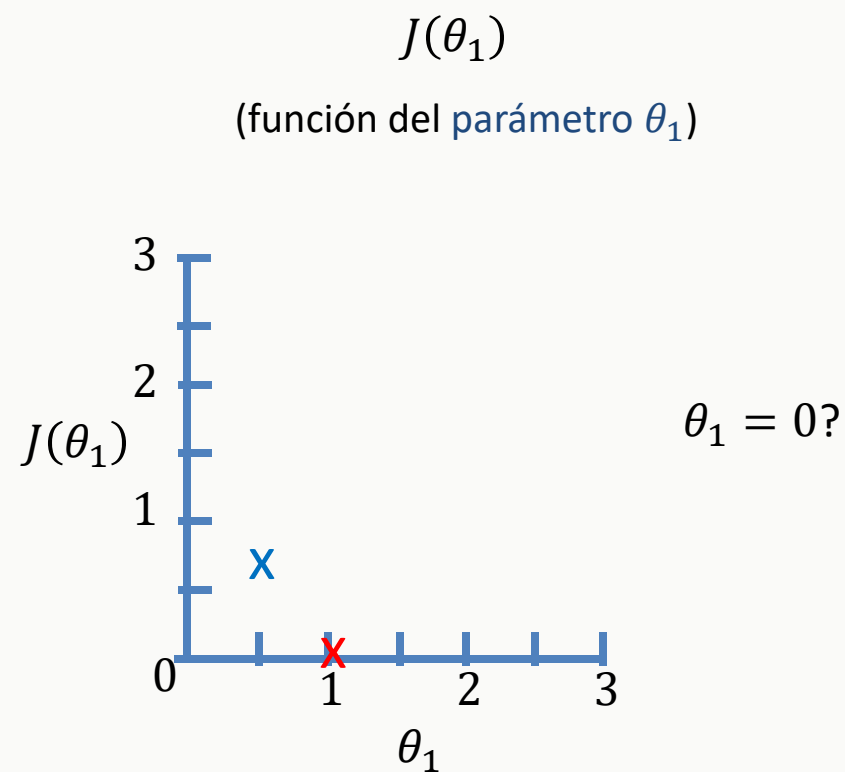
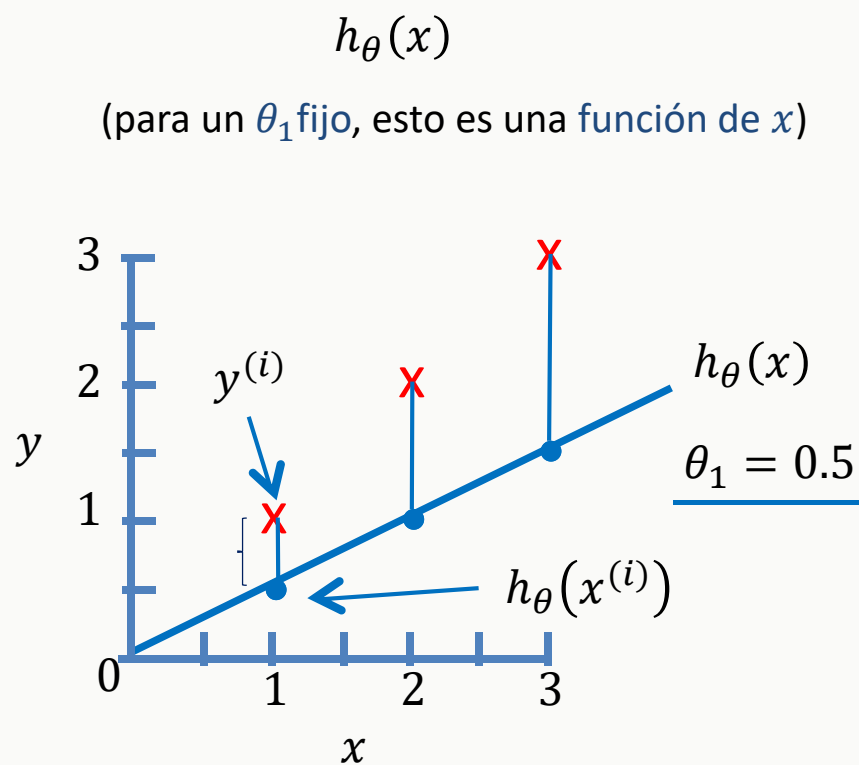


$$J(\theta_1) = \frac{1}{2N} \sum_{i=1}^N (h_{\theta}(x^{(i)}) - y^{(i)})^2 = \frac{1}{2N} \sum_{i=1}^N (\theta_1 x^{(i)} - y^{(i)})^2$$

$J(1) = 0$

$$J(1) = \frac{1}{2 \cdot 3} (0^2 + 0^2 + 0^2) = 0$$

Hipótesis vs. Costo

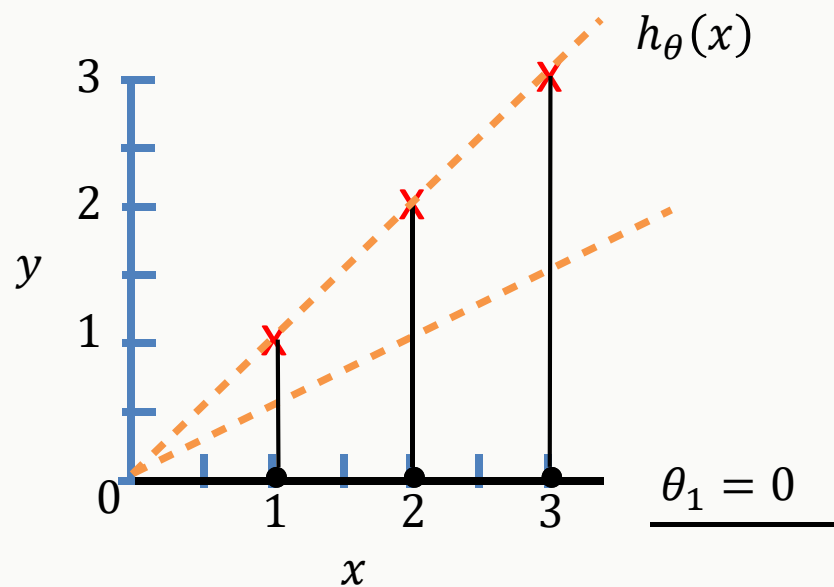


$$J(0.5) = \frac{1}{2 \cdot 3} [(0.5 - 1)^2 + (1 - 2)^2 + (1.5 - 3)^2] \cong 0.58$$

Hipótesis vs. Costo

$h_{\theta}(x)$

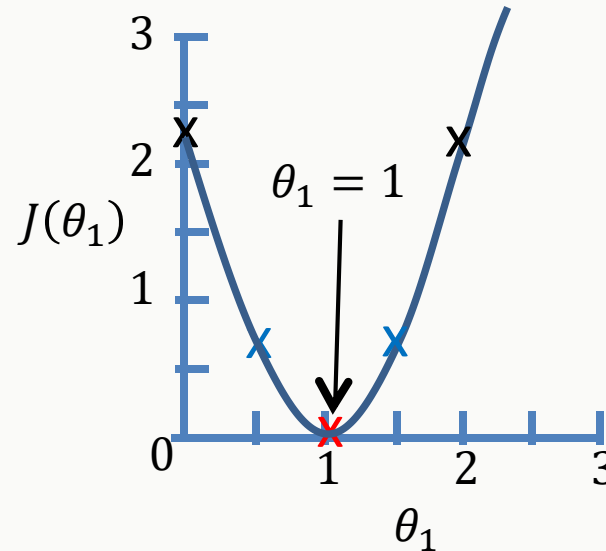
(para un θ_1 fijo, esto es una función de x)



$$J(0) = \frac{1}{2 \cdot 3} [(1)^2 + (2)^2 + (3)^2] \cong 2.3$$

$J(\theta_1)$

(función del parámetro θ_1)



FUNCIÓN DE COSTO

Intuición II

Problema de minimización de costo

Hipótesis: $h_{\theta}(x) = \theta_0 + \theta_1 x$

Parámetros: θ_0, θ_1

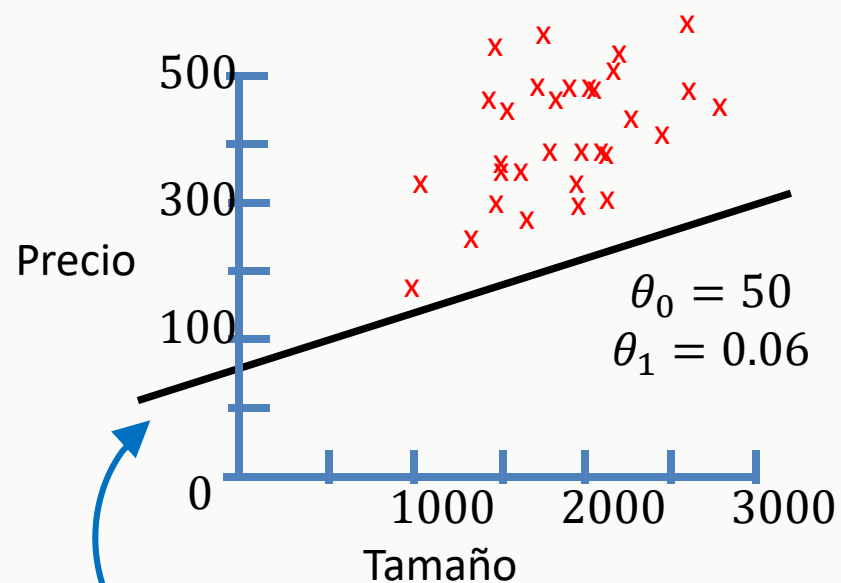
Función de costo: $J(\theta_0, \theta_1) = \frac{1}{2N} \sum_{i=1}^N (h_{\theta}(x^{(i)}) - y^{(i)})^2$

Objetivo: $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$

Hipótesis vs. Costo

$$h_{\theta}(x)$$

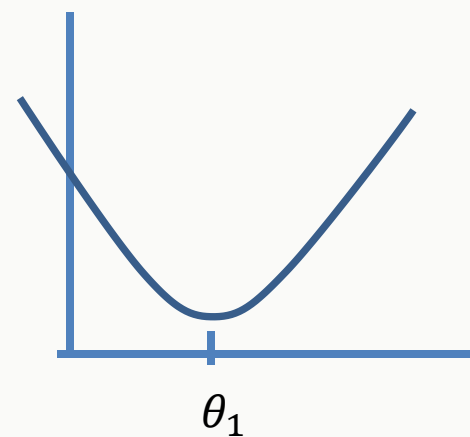
(para un θ_1, θ_2 fijos, esto es una función de x)



$$h_{\theta}(x) = 50 + 0.06x$$

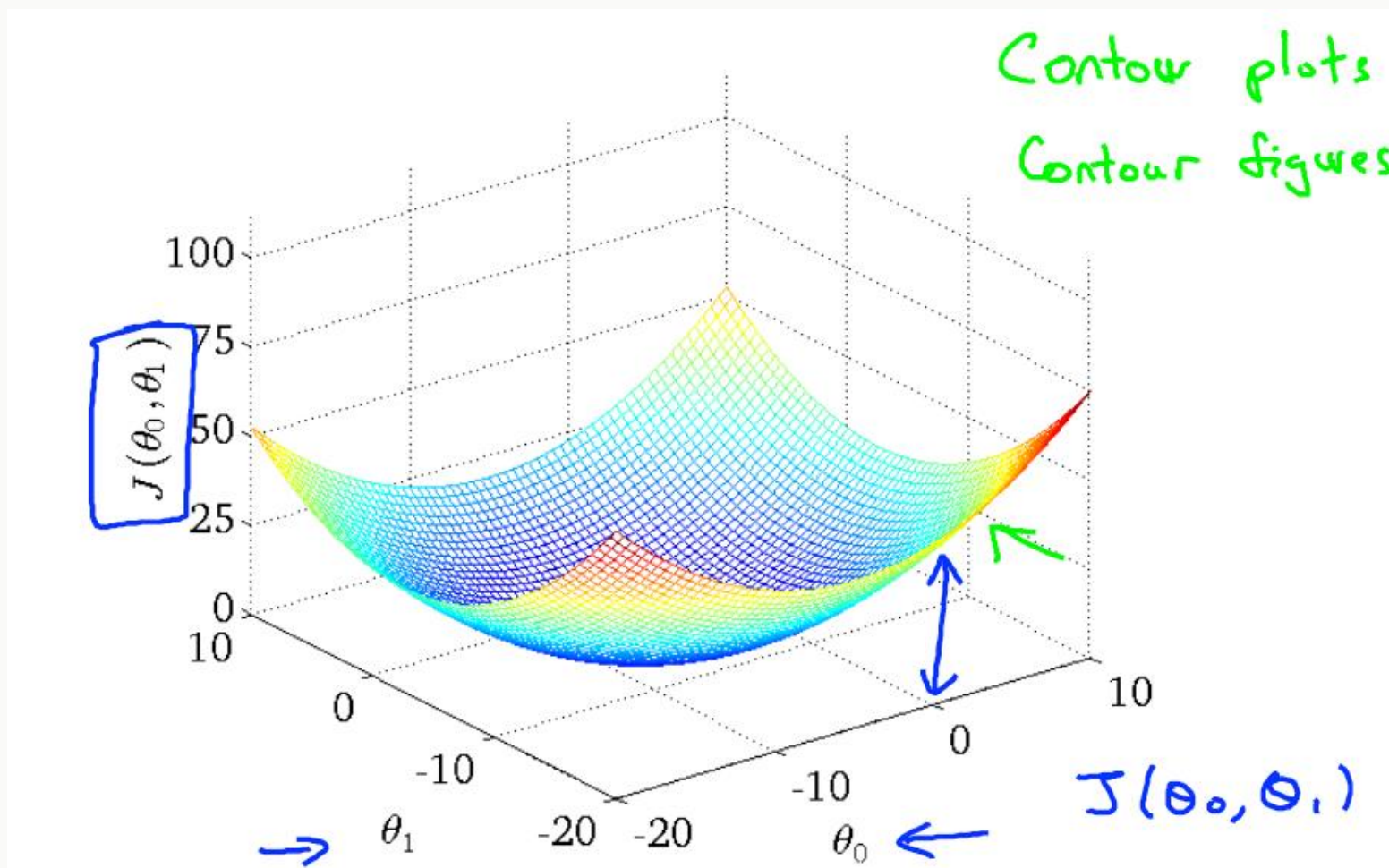
$$J(\theta_0, \theta_1)$$

(función de los parámetros θ_0, θ_1)



$$\theta_0, \theta_1$$

Gráficos de contorno



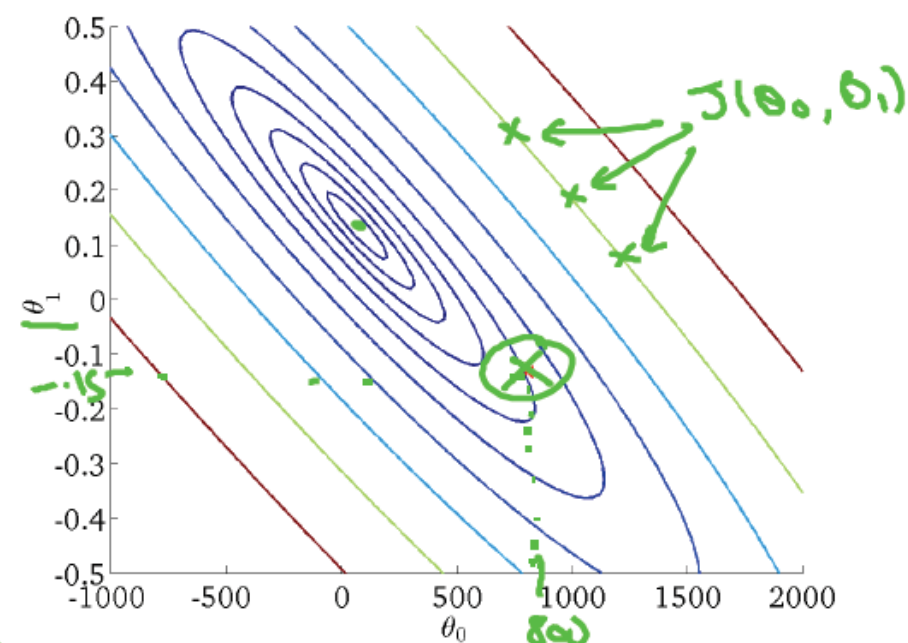
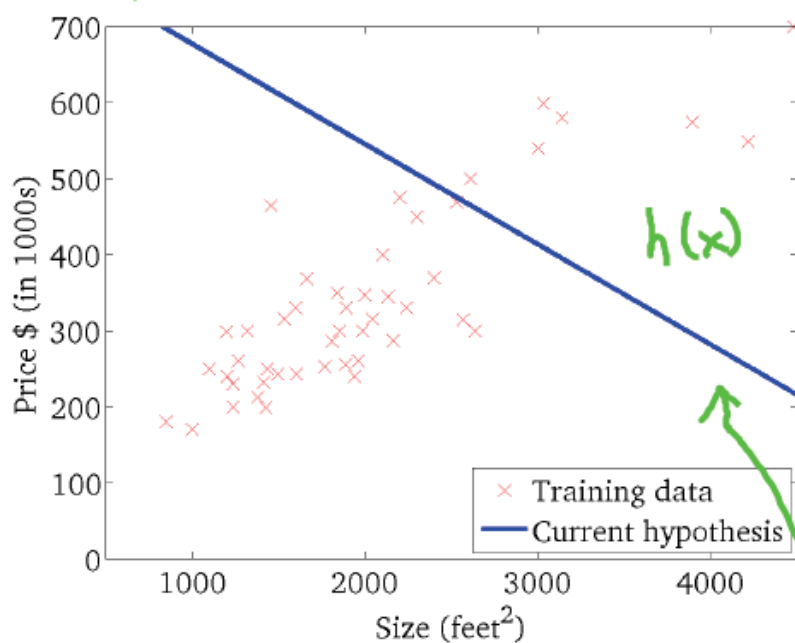
Hipótesis vs. Costo

$$h_{\theta}(x)$$

(para un θ_1, θ_2 fijos, esto es una función de x)

$$J(\theta_0, \theta_1)$$

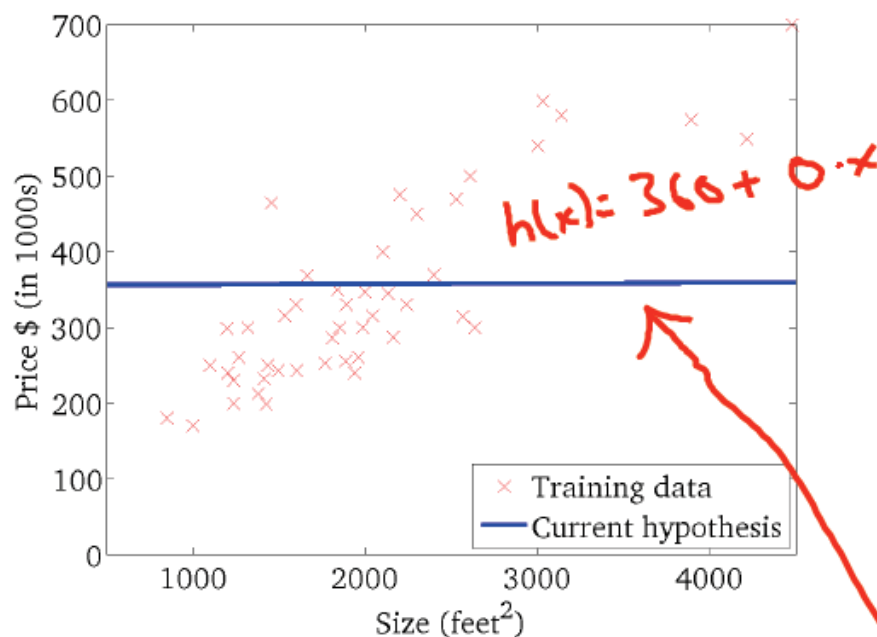
(función de los parámetros θ_0, θ_1)



Hipótesis vs. Costo

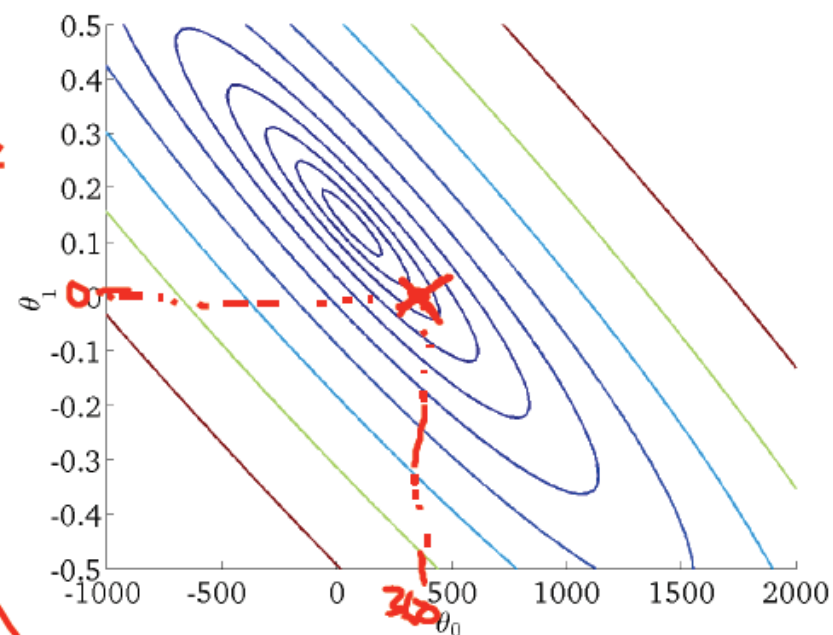
$$h_{\theta}(x)$$

(para un θ_1, θ_2 fijos, esto es una función de x)



$$J(\theta_0, \theta_1)$$

(función de los parámetros θ_0, θ_1)

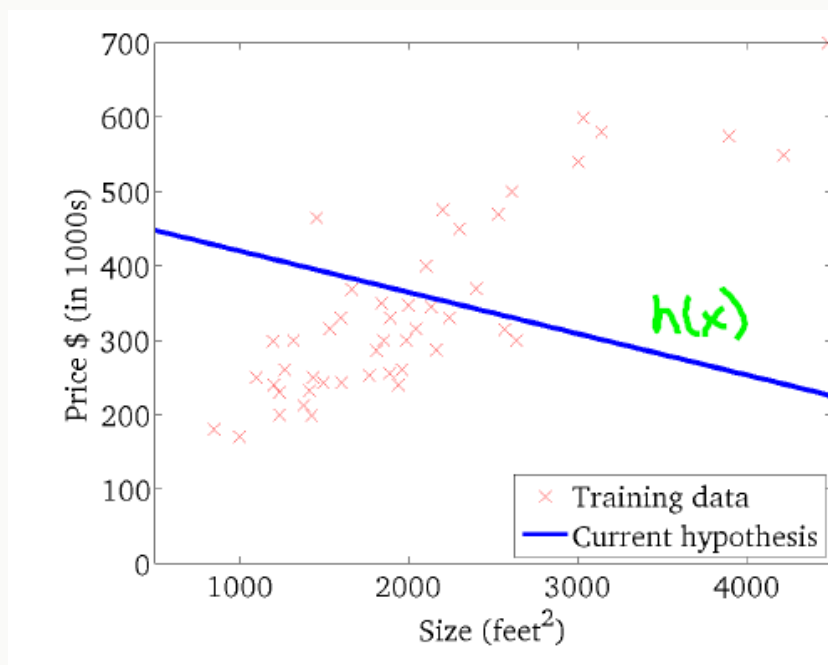


$$\begin{cases} \theta_0 = 360 \\ \theta_1 = 0 \end{cases}$$

Hipótesis vs. Costo

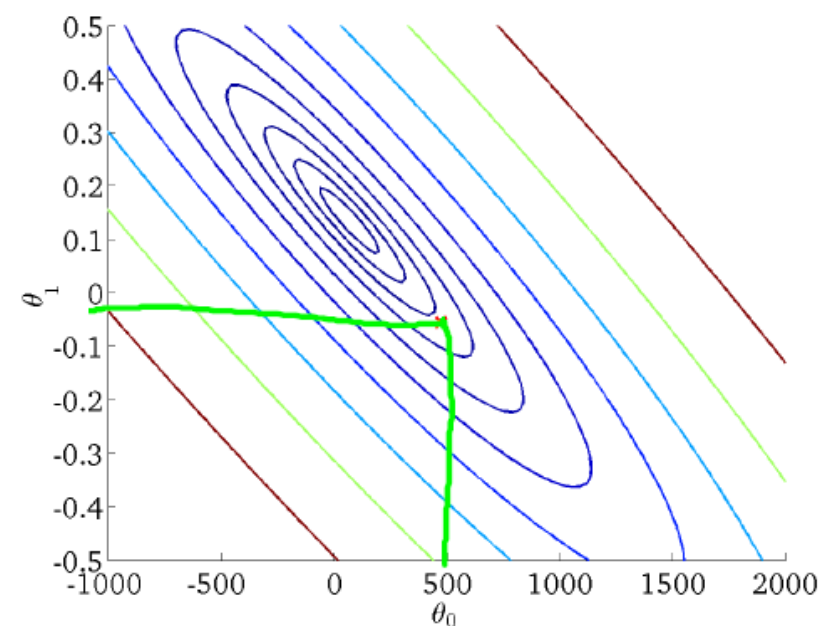
$$h_{\theta}(x)$$

(para un θ_1, θ_2 fijos, esto es una función de x)



$$J(\theta_0, \theta_1)$$

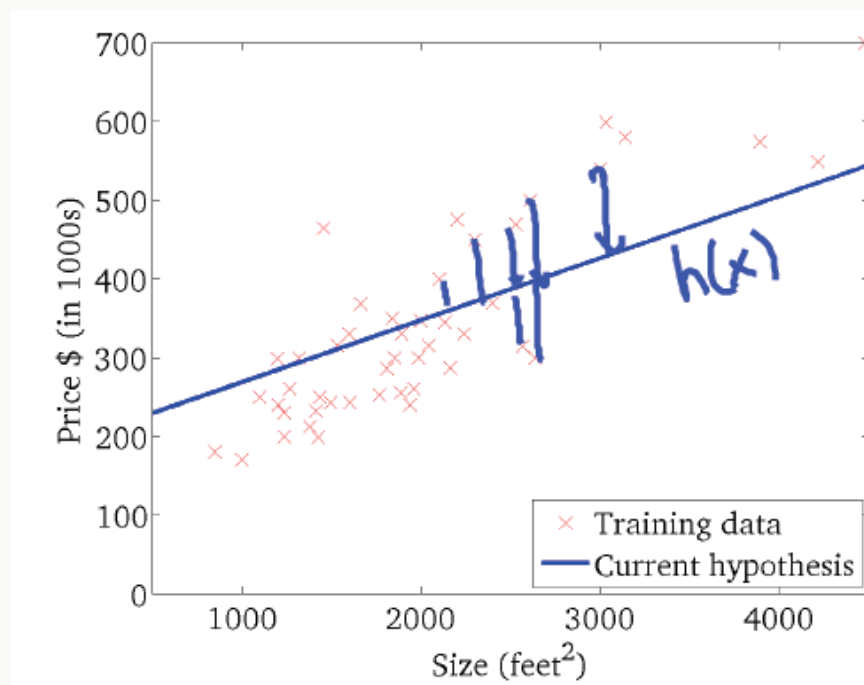
(función de los parámetros θ_0, θ_1)



Hipótesis vs. Costo

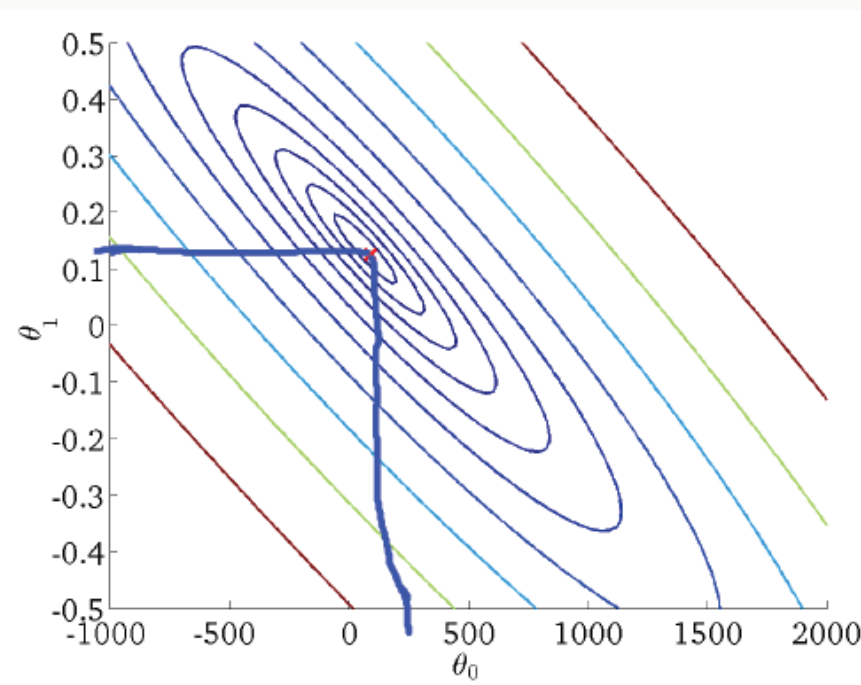
$$h_{\theta}(x)$$

(para un θ_1, θ_2 fijos, esto es una función de x)



$$J(\theta_0, \theta_1)$$

(función de los parámetros θ_0, θ_1)



DESCENSO DE GRADIENTE

Problema de minimización

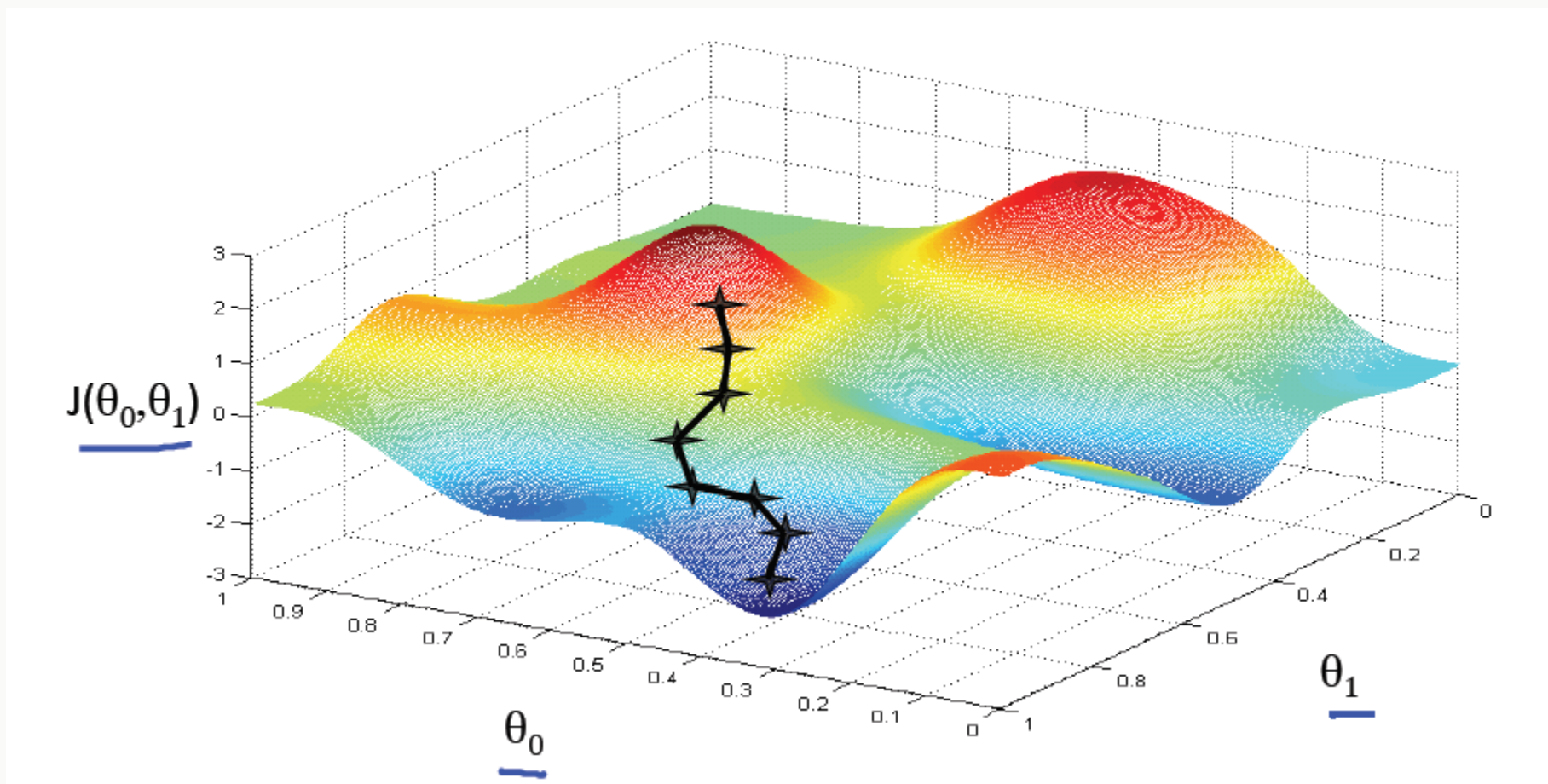
Función de costo: $J(\theta_0, \theta_1)$

Objetivo: $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$

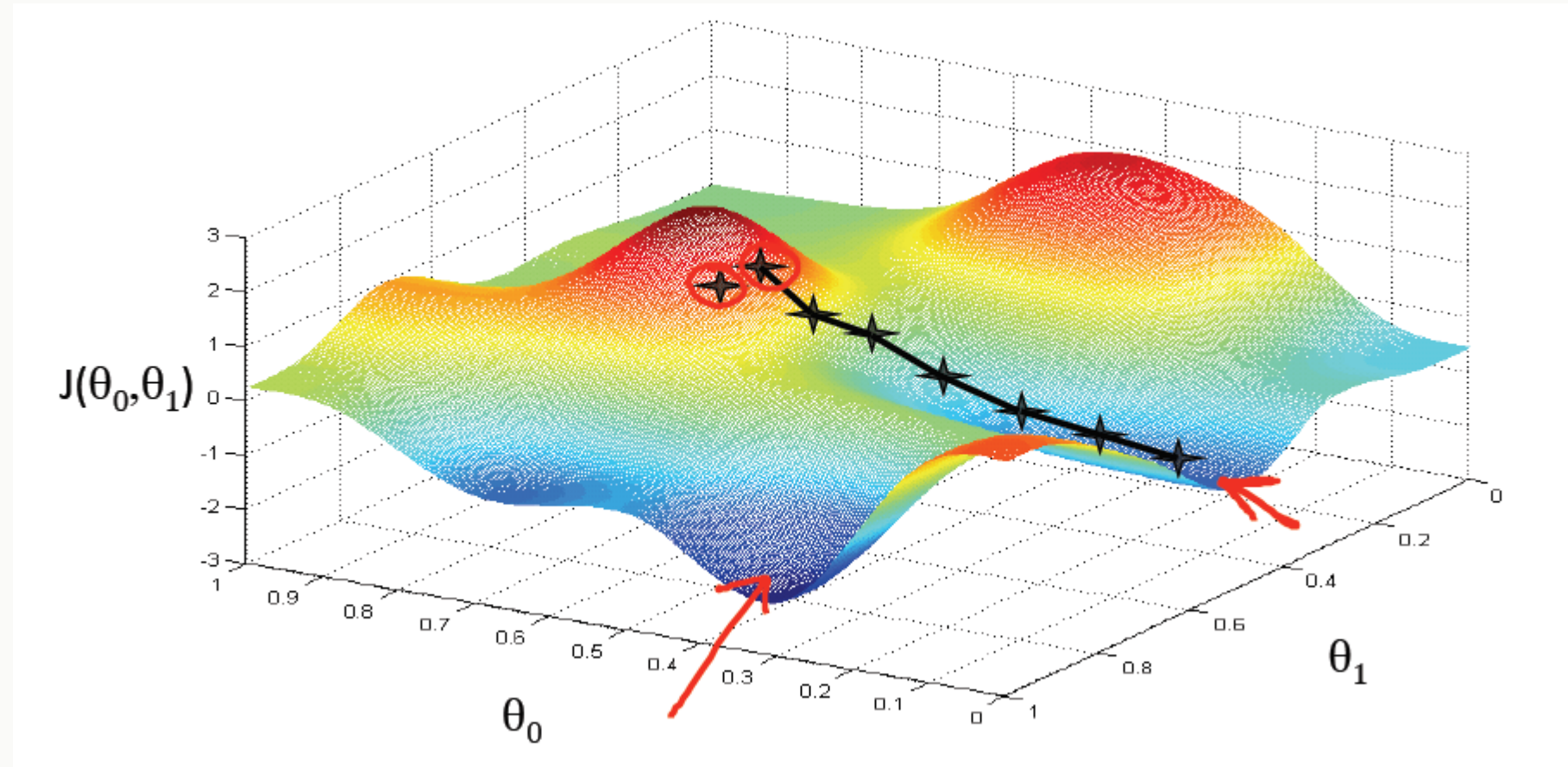
Descripción resumida:

- Comienza con algún θ_0, θ_1
- Cambia θ_0, θ_1 para reducir $J(\theta_0, \theta_1)$ hasta que (ojalá) lleguemos a un mínimo.

Topología de la función de costo



Topología de la función de costo



Algoritmo de descenso de gradiente

Repite hasta la convergencia {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \quad (\text{para } j = 0 \text{ y } j = 1)$$

}

Asignación simultánea!!!

Correcto: actualización simultánea

$$\text{temp0} := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$

$$\text{temp1} := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$

$$\theta_0 := \text{temp0}$$

$$\theta_1 := \text{temp1}$$

Incorrecto:

$$\text{temp0} := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$

$$\theta_0 := \text{temp0}$$

$$\text{temp1} := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$

$$\theta_1 := \text{temp1}$$

INTUICIÓN DEL DESCENSO DE GRADIENTE

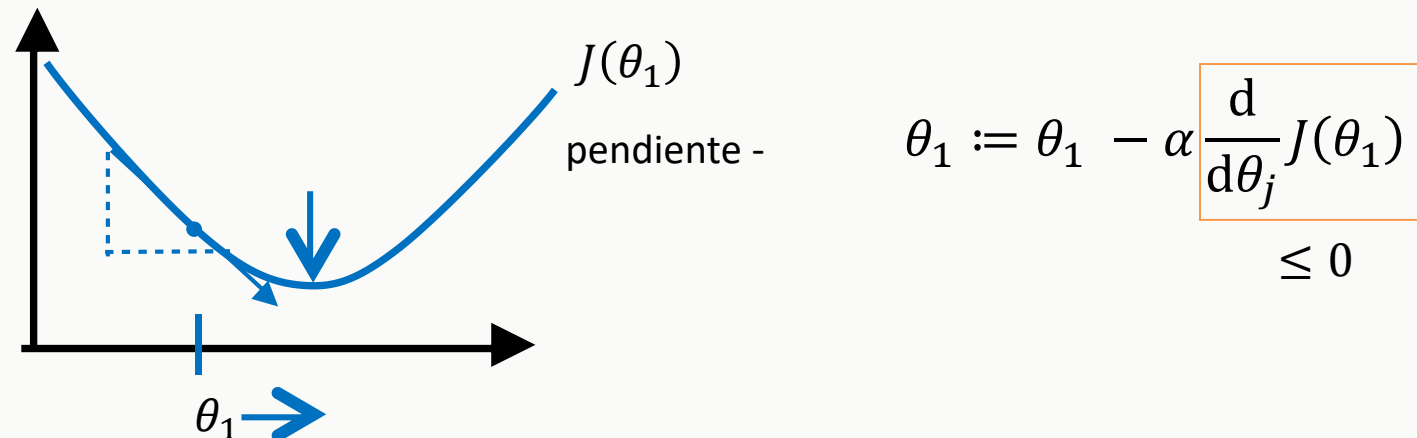
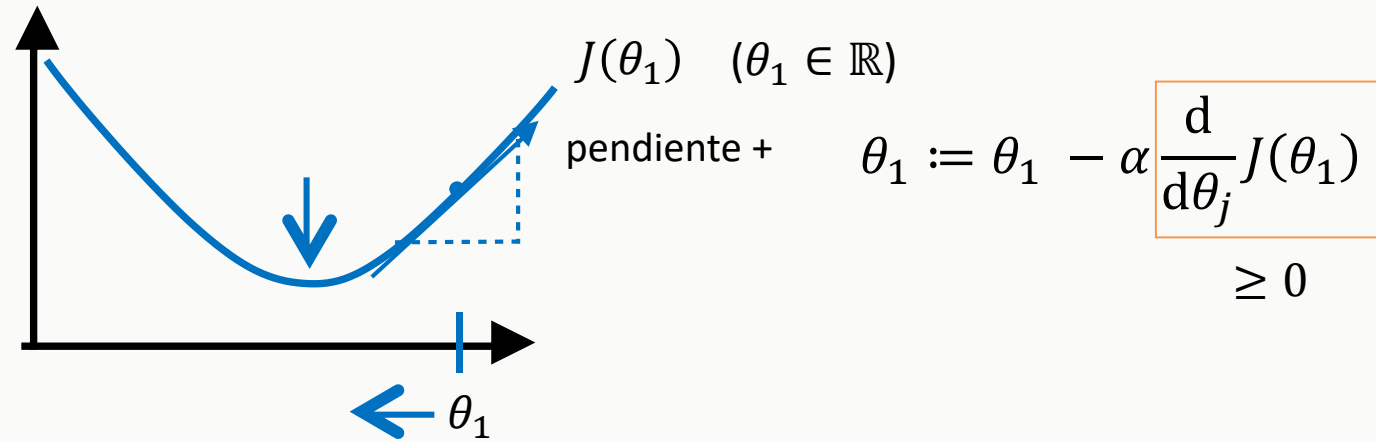
Algoritmo de descenso de gradiente

Repite hasta la convergencia {

$$\underbrace{\theta_j}_{-} := \underbrace{\theta_j}_{-} - \underbrace{\alpha}_{\text{tasa de aprendizaje}} \frac{\partial}{\partial \theta_j} \underbrace{J(\theta_0, \theta_1)}_{\text{derivada}} \quad \text{(actualiza simultáneamente } j = 0 \text{ y } j = 1)$$

$$\min_{\theta_1} J(\underbrace{\theta_1}_{-}) \quad \theta_1 \in \mathbb{R}$$

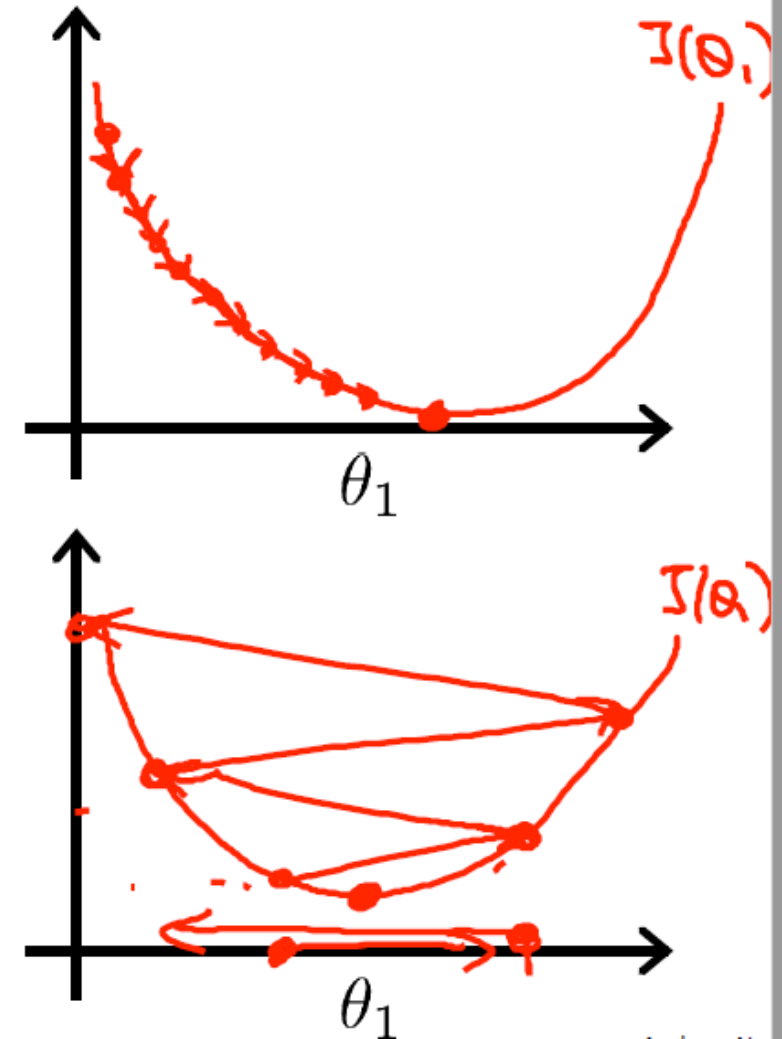
Actualización en función de la pendiente



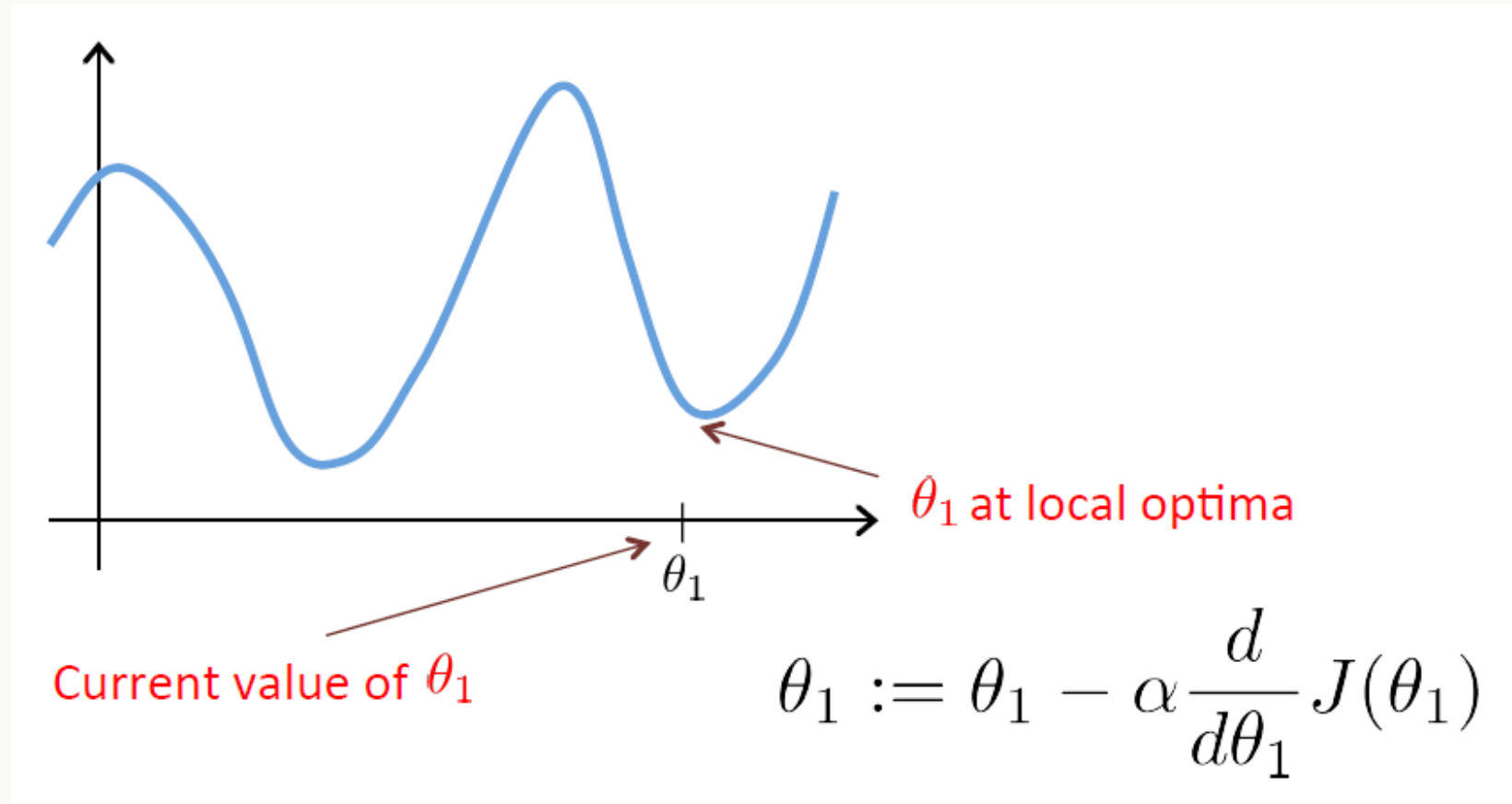
$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

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.



Mínimos locales



Convergencia

- El descenso de gradiente puede converger a un mínimo local, aún con una tasa de aprendizaje α fija.

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

- Conforme nos acercamos a un mínimo local, el descenso de gradiente toma automáticamente pasos más pequeños.
- De esta forma, no es necesario decrementar α en el tiempo.

REGRESIÓN LINEAL

Regresión por Máximo de Verosimilitud (Maximum Likelihood)

$$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x})$$

- Suponiendo que los datos a predecir (objetivo) t están centrados en $y(\mathbf{x}, \mathbf{w})$ con ruido aditivo gaussiano, es decir

$$p(t|\mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(t|y(\mathbf{x}, \mathbf{w}), \beta^{-1})$$

- Consideremos un conjunto de entradas $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ con objetivos correspondientes t_1, \dots, t_N . Juntamos estas variables objetivo en un vector \mathbf{t} (conjunto de variables unidimensionales).

Regresión por Máximo de Verosimilitud (Maximum Likelihood)

- ▶ La función de verosimilitud está dada por

$$p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^N \mathcal{N}(t_n | \mathbf{w}^T \phi(\mathbf{x}_n), \beta^{-1})$$

- ▶ Tomando el logaritmo tenemos

$$\ln p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta) = \sum_{n=1}^N \ln \mathcal{N}(t_n | \mathbf{w}^T \phi(\mathbf{x}_n), \beta^{-1})$$

- ▶ Considerando la forma estándar de la gaussiana obtenemos

$$\ln p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta) = \frac{N}{2} \ln \beta - \frac{N}{2} \ln(2\pi) - \beta E_D(\mathbf{w})$$

- ▶ Donde la función de error de suma de cuadrados está definida como

$$E_D = \frac{1}{2} \sum_{n=1}^N \{t_n - \mathbf{w}^T \phi(\mathbf{x}_n)\}^2$$

Regresión por Máximo de Verosimilitud (Maximum Likelihood)

- ▶ Usamos Máximo de Verosimilitud, donde maximizar
 $p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta)$

- ▶ Es equivalente a minimizar

$$E_D = \frac{1}{2} \sum_{n=1}^N \{t_n - \mathbf{w}^T \phi(\mathbf{x}_n)\}^2$$

- ▶ Calculando el gradiente con respecto a \mathbf{w} :

$$\nabla_{\mathbf{w}} p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta) = \sum_{n=1}^N \{t_n - \mathbf{w}^T \phi(\mathbf{x}_n)\} \phi(\mathbf{x}_n)^T$$

- ▶ Igualando a cero, obtenemos:

$$0 = \sum_{n=1}^N t_n \phi(\mathbf{x}_n)^T - \mathbf{w}^T \left(\sum_{n=1}^N \phi(\mathbf{x}_n) \phi(\mathbf{x}_n)^T \right)$$

Regresión por Máximo de Verosimilitud (Maximum Likelihood)

- Despejando para w

$$\mathbf{w}_{\text{ML}} = (\Phi^T \Phi)^{-1} \Phi^T \mathbf{t}$$

- Que se conocen como las *ecuaciones normales* del problema de mínimos cuadrados. Aquí Φ es una matriz $M \times M$ llamada la *matriz de diseño*. Cuyos elementos son $\Phi_{nj} = \phi_j(x_n)$

$$\Phi = \begin{pmatrix} \phi_0(x_1) & \phi_1(x_1) & \dots & \phi_{M-1}(x_1) \\ \phi_0(x_2) & \phi_1(x_2) & \dots & \phi_{M-1}(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(x_N) & \phi_1(x_N) & \dots & \phi_{M-1}(x_N) \end{pmatrix}$$

- La cantidad

$$\Phi^\dagger = (\Phi^T \Phi)^{-1} \Phi^T$$

se conoce como la *pseudo-inversa Moore-Penrose* de la matriz Φ .