NOCIONES BÁSICAS EN MACHINE LEARNING

Dr. Jorge Hermosillo
Laboratorio de Semántica Computacional

Centro de Investigación en Ciencias

Universidad Autónoma del Estado de Morelos

MODELOS LINEALES DE REGRESIÓN

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

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + \cdots + w_D x_D \text{ donde } \mathrm{x} = (x_1, \dots, x_D)^{\mathrm{T}}$$

- ***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:

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

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

FEATURES (CARACTERÍSTICAS) COMUNES

- Hemos visto la importancia de hacer pre-procesamiento de los datos, también llamado el ***proceso de extracción de características*** (“*feature extraction*”) de las variables de entrada.
- 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*”):

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

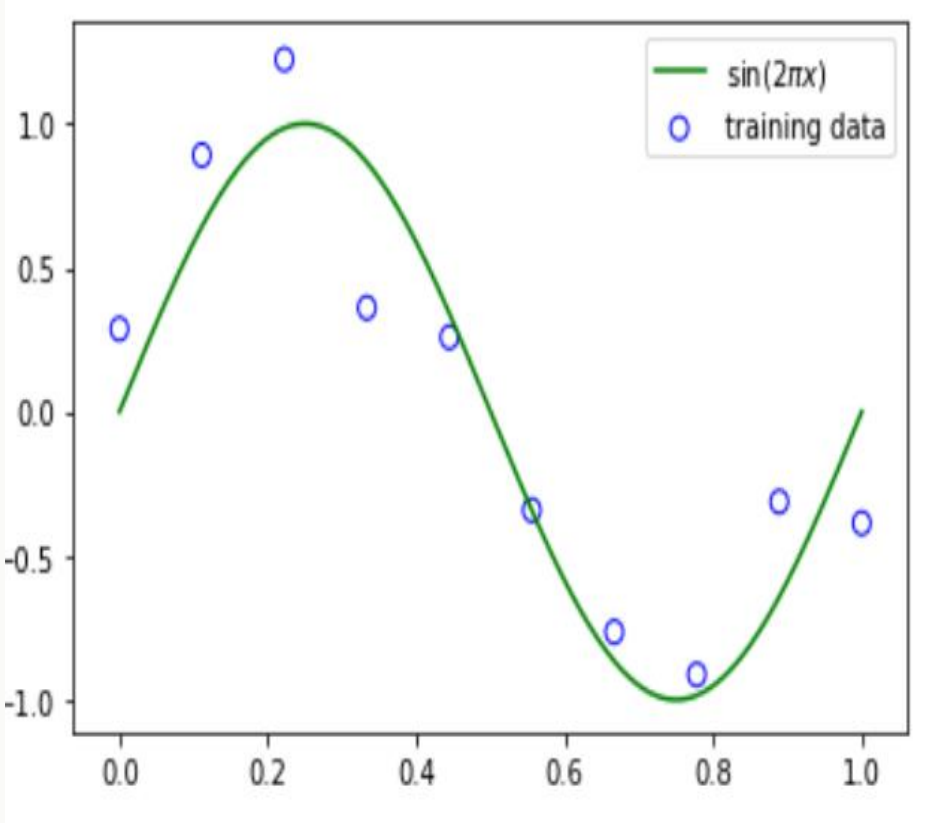
- Funciones base Gaussianas (“*Gaussian features*”)

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

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

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

REGRESIÓN POLINOMIAL



DATOS DE ENTRADA:

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

$$\mathbf{x} \equiv (x_1, x_2, \cdots, x_N)^T$$
$$\mathbf{t} \equiv (t_1, t_2, \cdots, t_N)^T$$

donde:

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

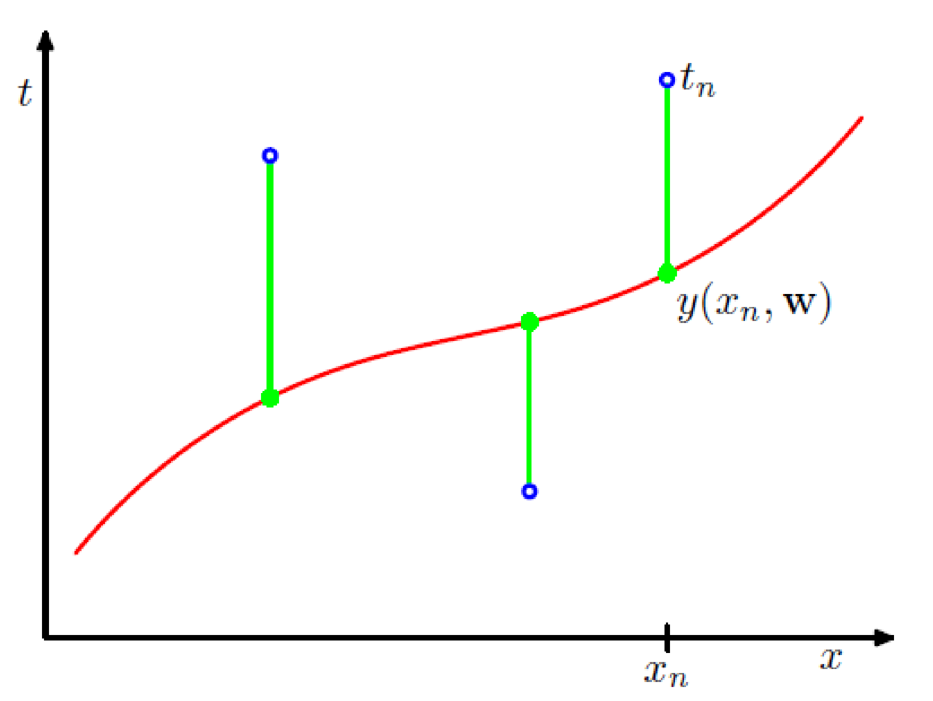
con $\epsilon \sim \mathcal{N}(0, (0.3)^2)$

SALIDA:

Un modelo lineal de ajuste polinomial:

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

FUNCIÓN DE ERROR/PÉRDIDA (LOSS FUNCTION)



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

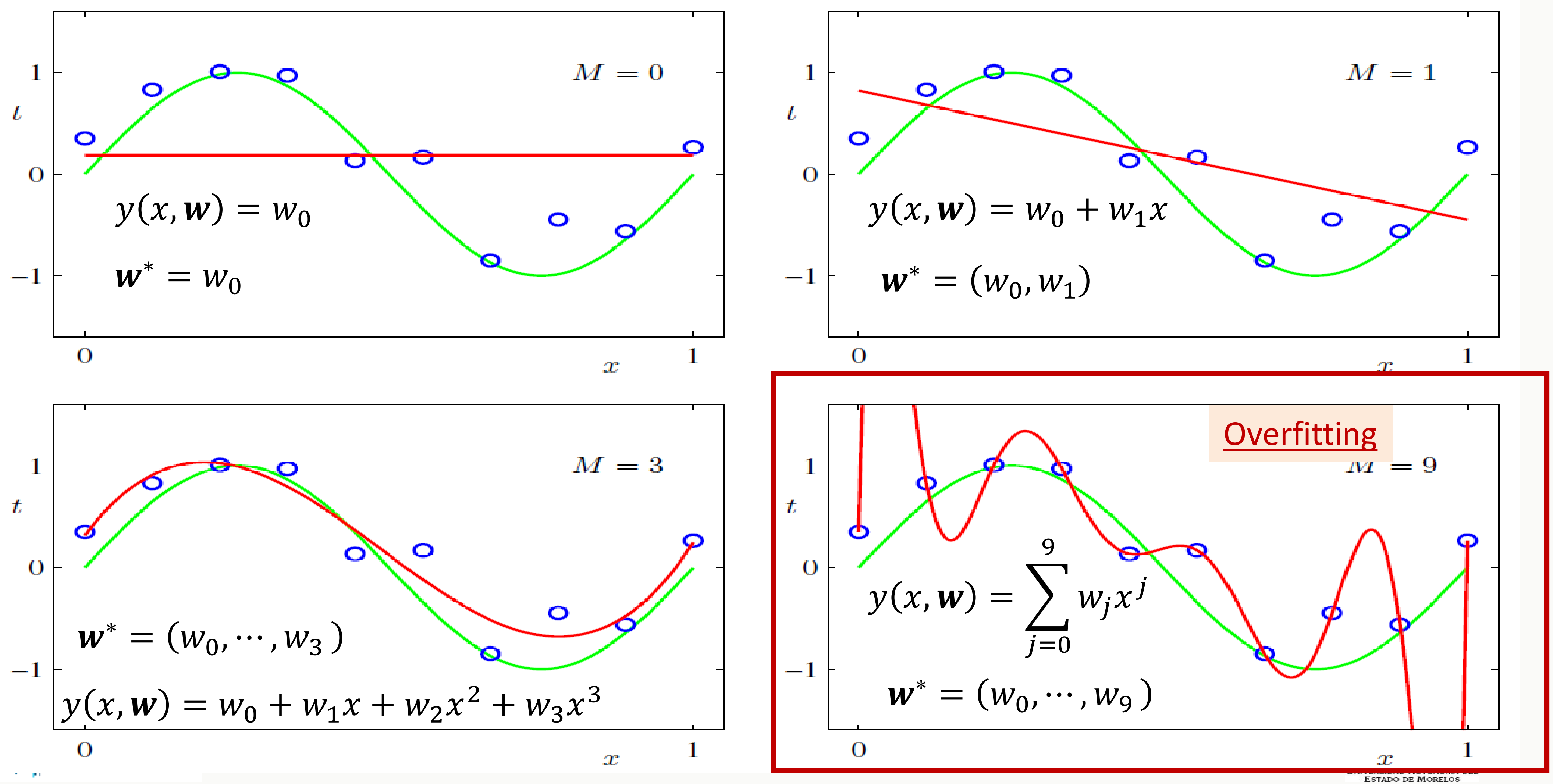
$$y(x, \boldsymbol{w}) = w_0 + w_1 x + \cdots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

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

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

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

EFECTO DE LA COMPLEJIDAD DEL MODELO EN EL AJUSTE

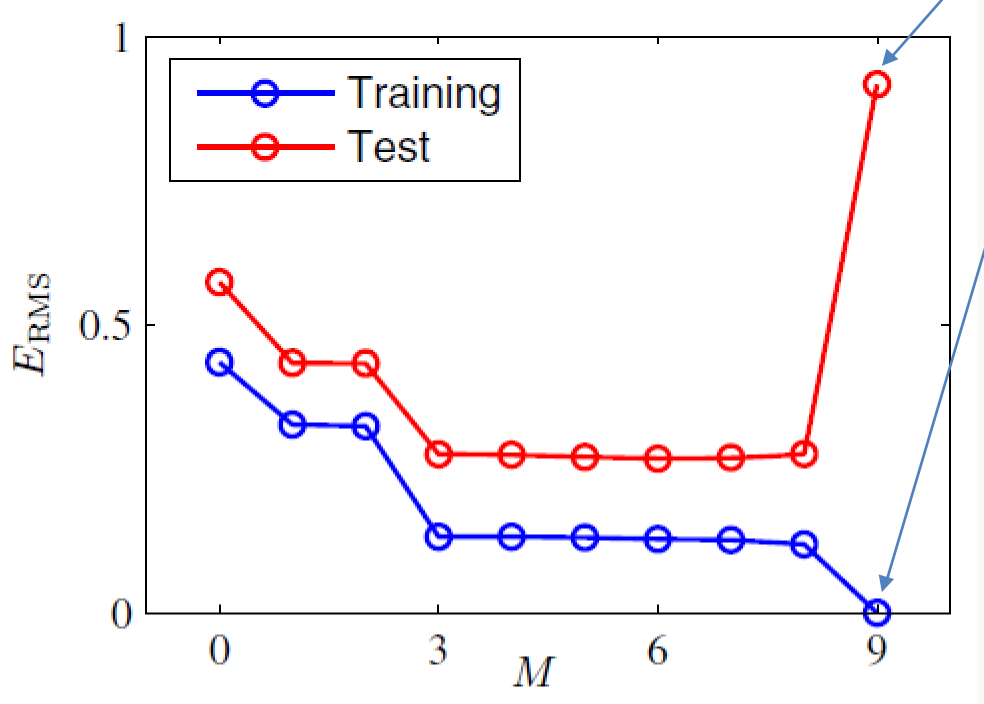


MÉTRICA DE DESEMPEÑO: ERMS

Overfitting

Error cuadrático medio:

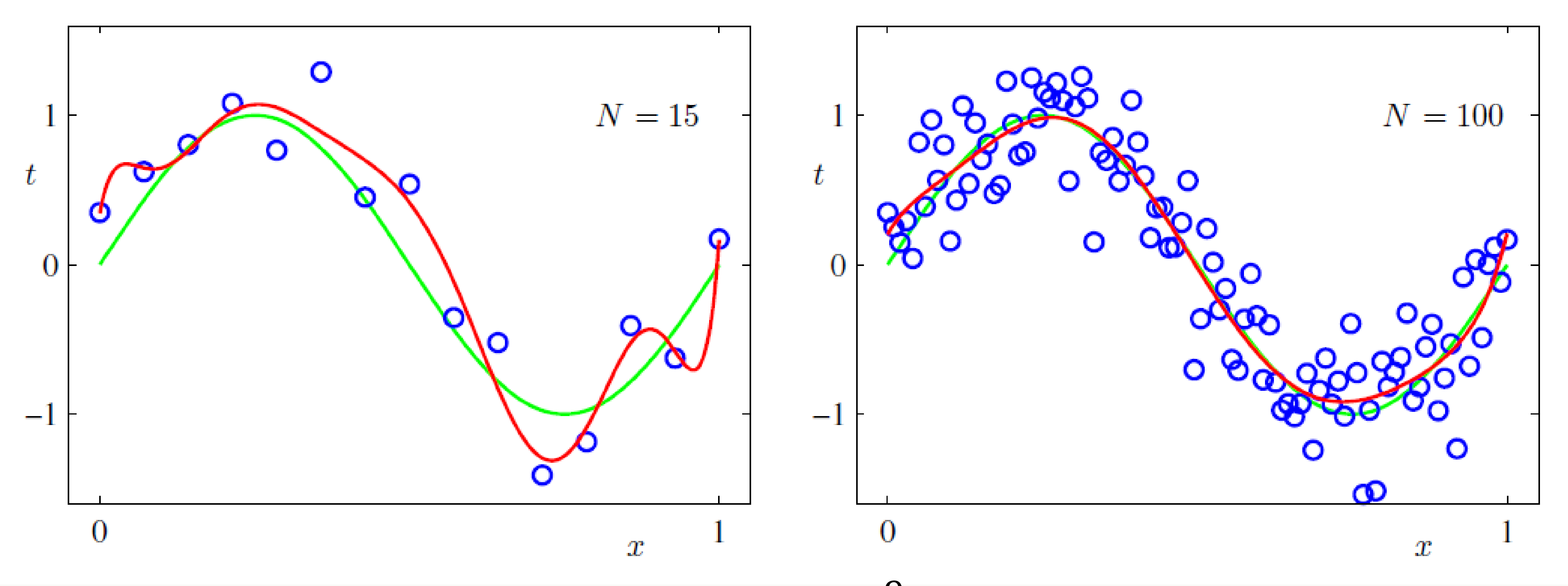
$$E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w}^{*})/N}$$



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

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

EFECTO DE LA CANTIDAD DE DATOS N EN EL AJUSTE



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

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).

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

Donde:

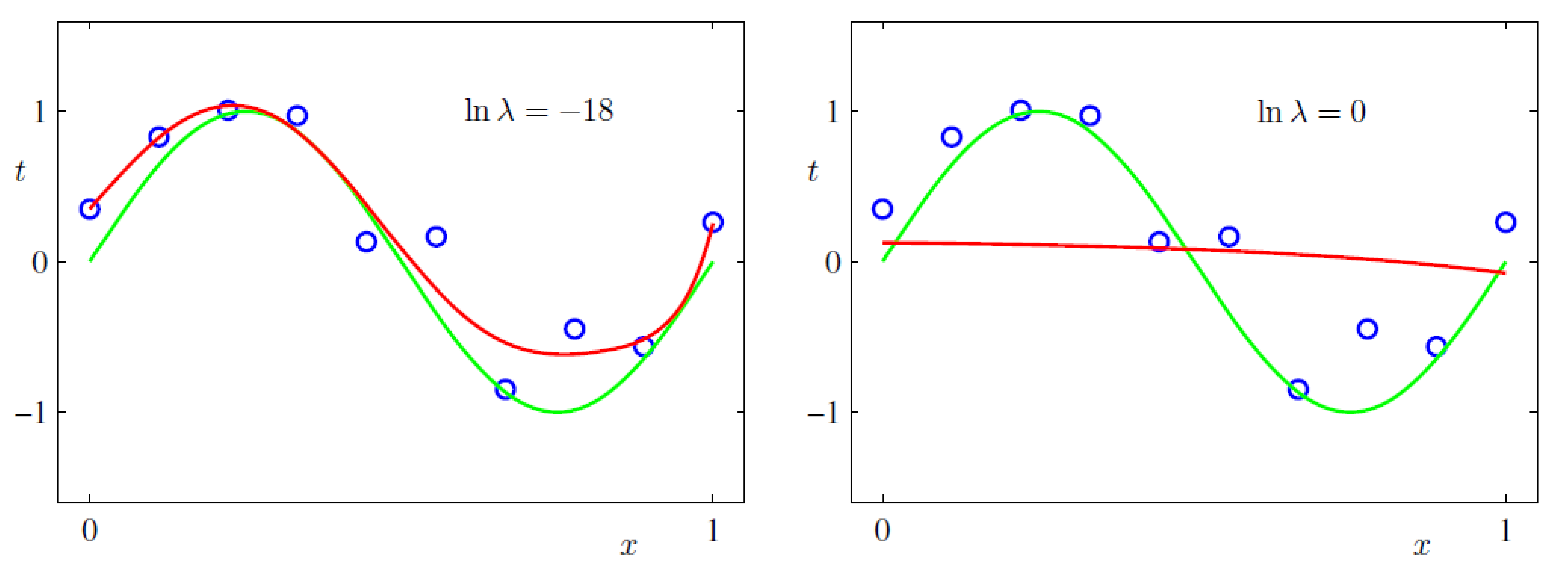
$$\|w\|^2 \equiv \boldsymbol{w}^{\mathrm{T}}\boldsymbol{w} = w_0^2 + w_1^2 + \cdots + w_M^2$$

EFECTO 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 λ = −∞ corresponde a un modelo sin regularización. Vemos que, a medida que el valor de λ aumenta, la magnitud típica de los pesos disminuye.

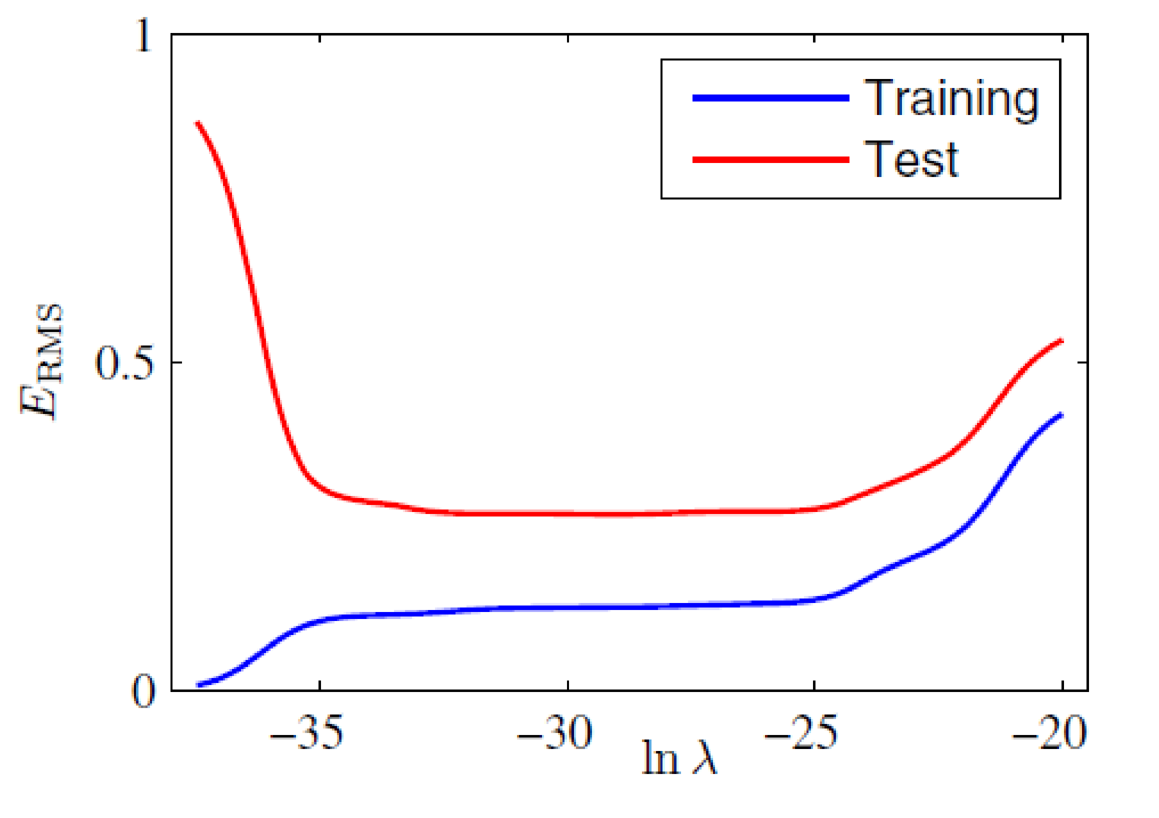
	$\ln\lambda = -\infty$	$\ln\lambda = -18$	$\ln\lambda = 0$
$w_0^\star$	0.35	0.35	0.13
$w_1^\star$	232.37	4.74	-0.05
$w_2^\star$	-5321.83	-0.77	-0.06
$w_3^\star$	48568.31	-31.97	-0.05
$w_4^\star$	-231639.30	-3.89	-0.03
$w_5^\star$	640042.26	55.28	-0.02
$w_6^\star$	-1061800.52	41.32	-0.01
$w_7^\star$	1042400.18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

EFECTO DE LA REGULARIZACIÓN EN EL AJUSTE



M = 9 con ln λ = −18 y ln λ = 0

EFECTO DE LA REGULARIZACIÓN EN EL DESEMPEÑO



FUNCIONES DE PÉRDIDA (COSTO)

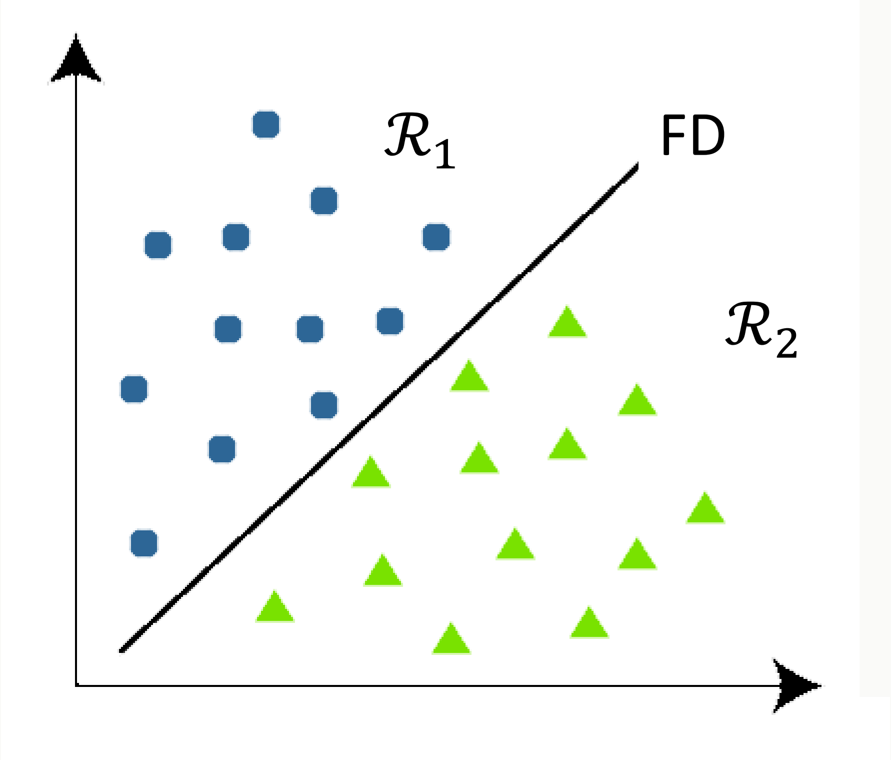
Nociones básicas

¿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.

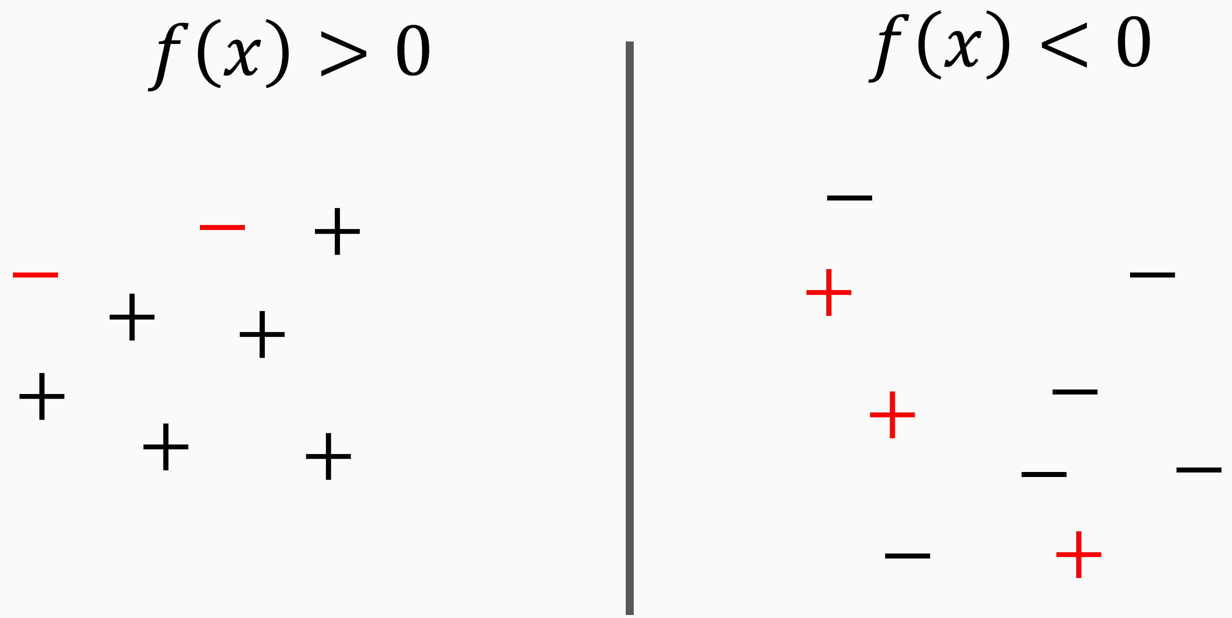
FUNCIÓN DE PÉRDIDA PARA CLASIFICACIÓN

- 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 $\text{sign}\big(f(x_i)\big)$ es y_i, donde $f(\cdot)$ es la función de clasificación y $y_i \coloneqq \{-1, +1\}$.



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

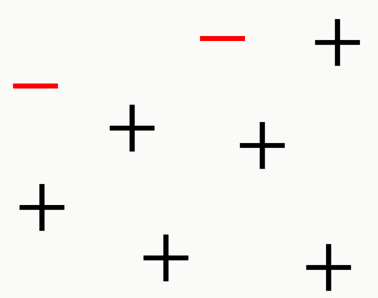
$$\frac{1}{n}\sum_{i=1}^{n} y_i \neq \text{sign}\big(f(x_i)\big)$$

Minimizar esta función es computacionalmente muy difícil.

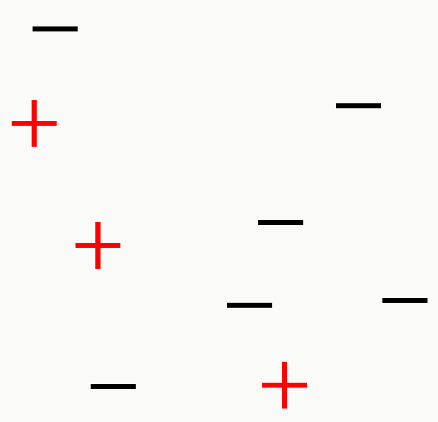
FUNCIÓN DE PÉRDIDA PARA CLASIFICACIÓN

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

$yf(x) \gtrless 0$



$yf(x) \lessgtr 0$

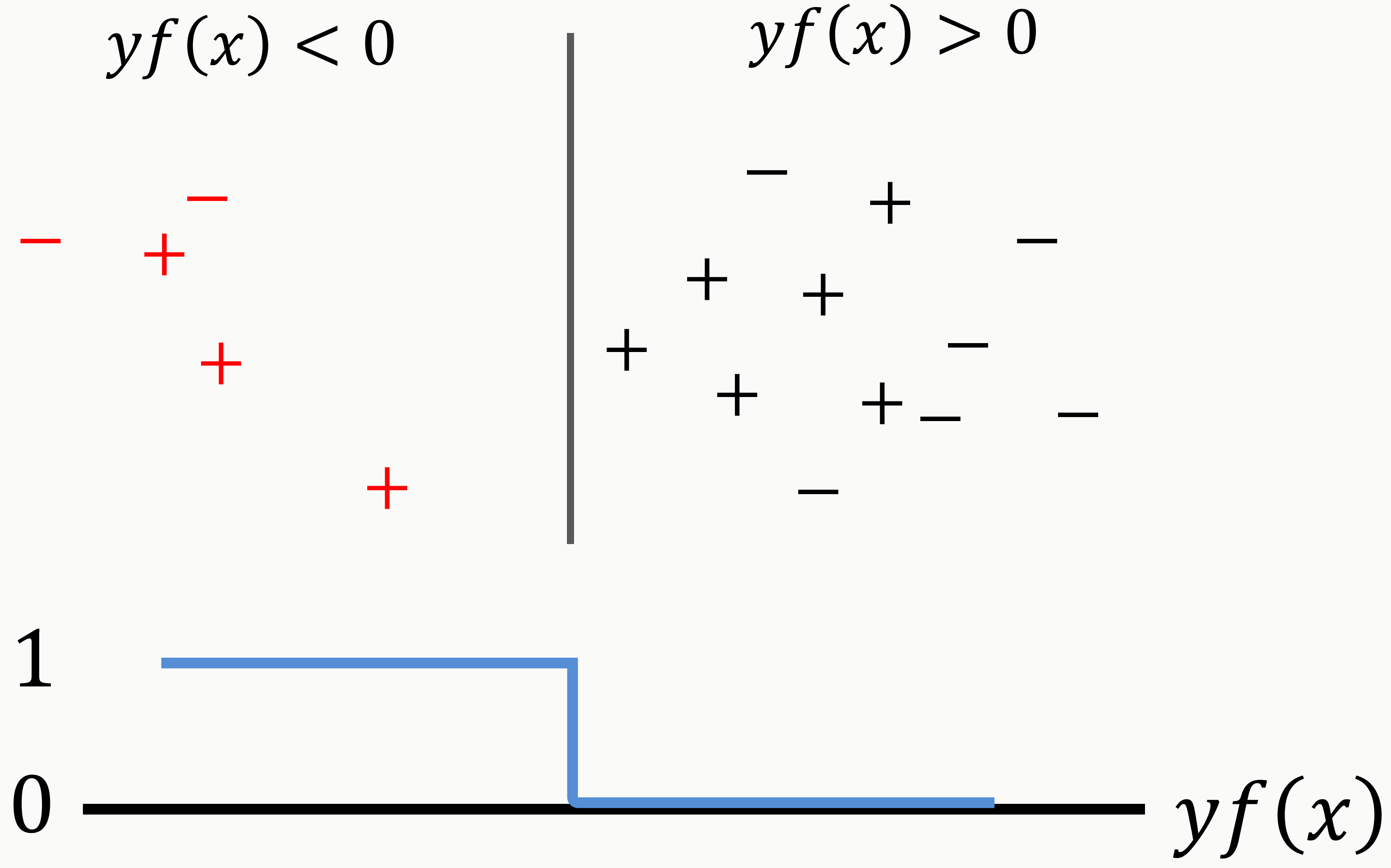


Fracción de veces que $\mathrm{sign}(f(x_i))$ no es y_i:

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

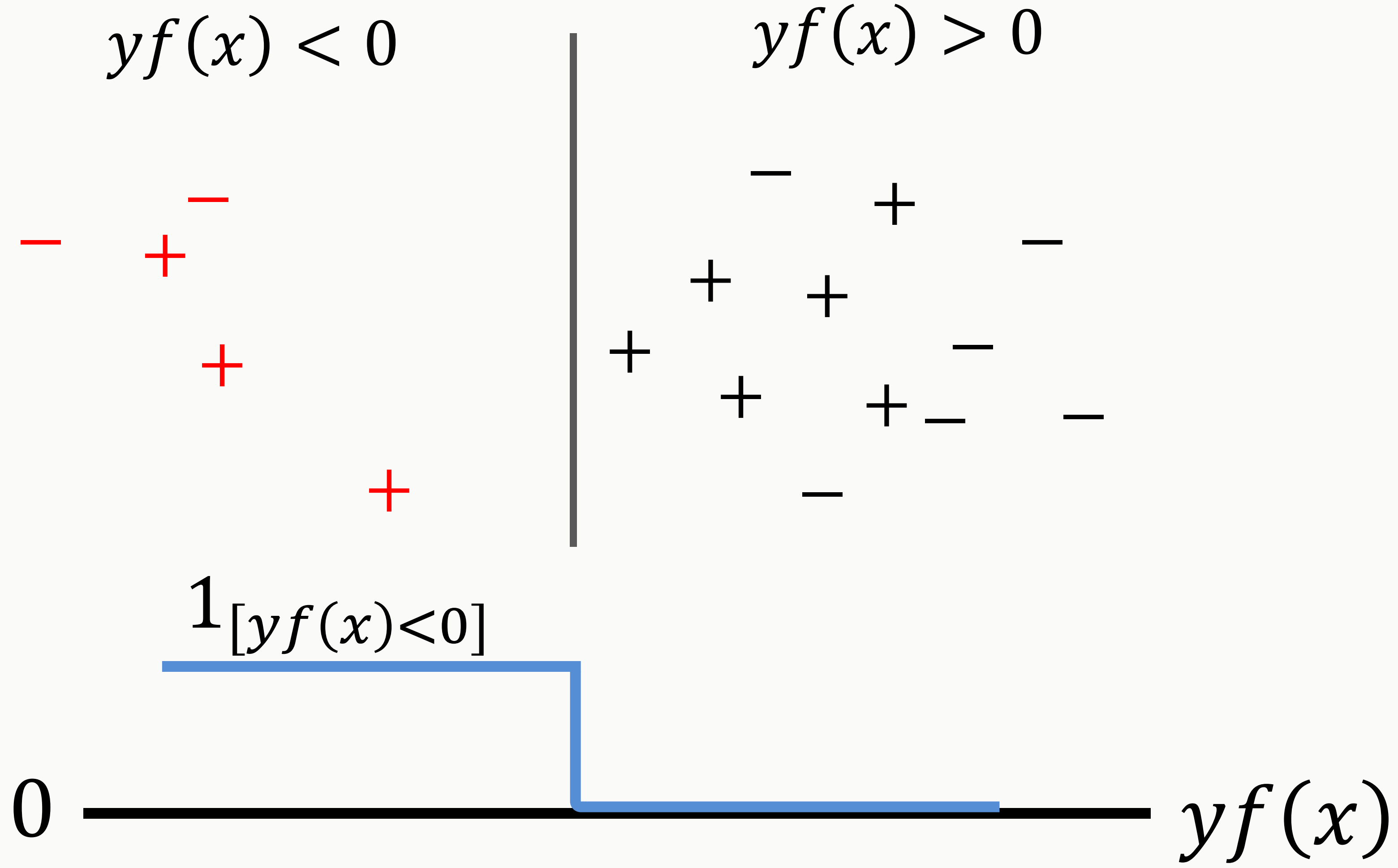
Minimizar esta función es computacionalmente muy difícil.

FUNCIÓN DE PÉRDIDA PARA CLASIFICACIÓN

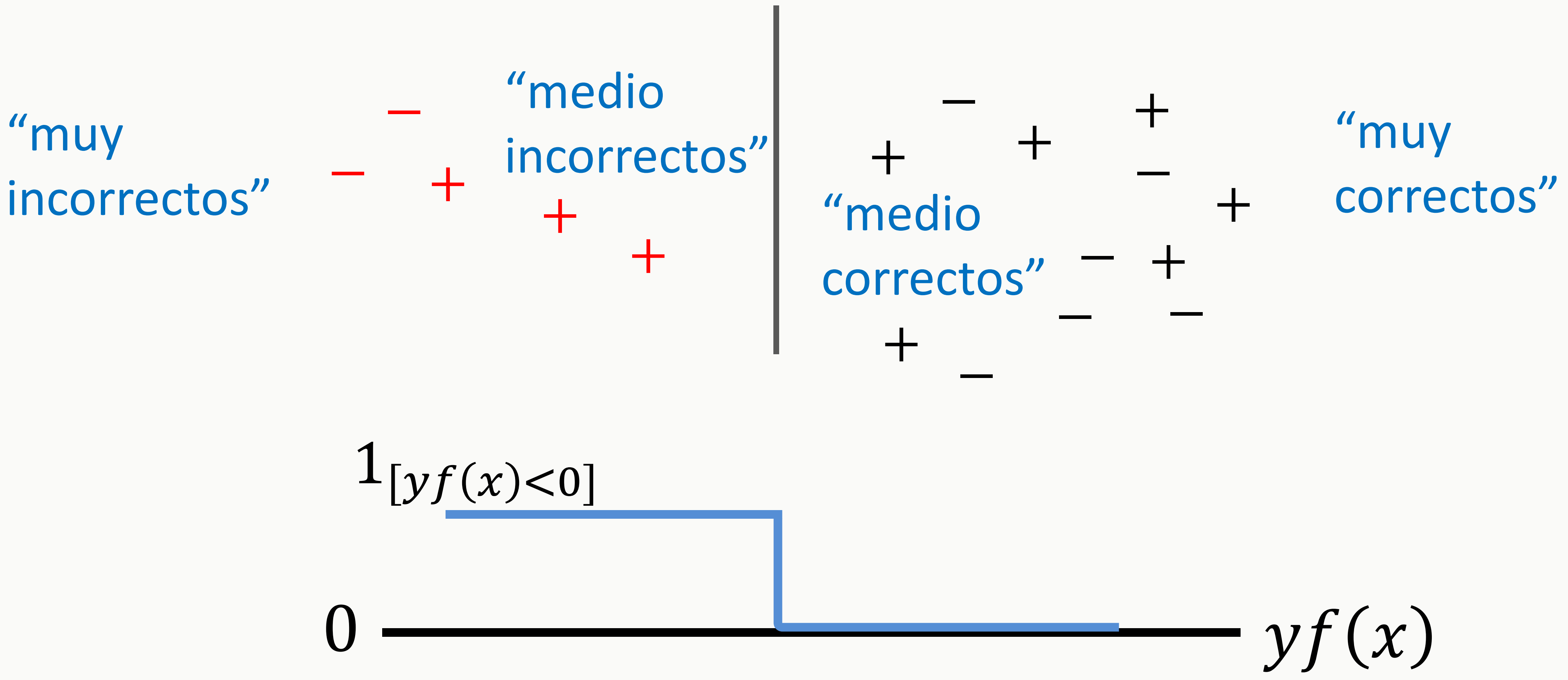


FUNCIÓN DE PÉRDIDA PARA CLASIFICACIÓN

0-1 Loss



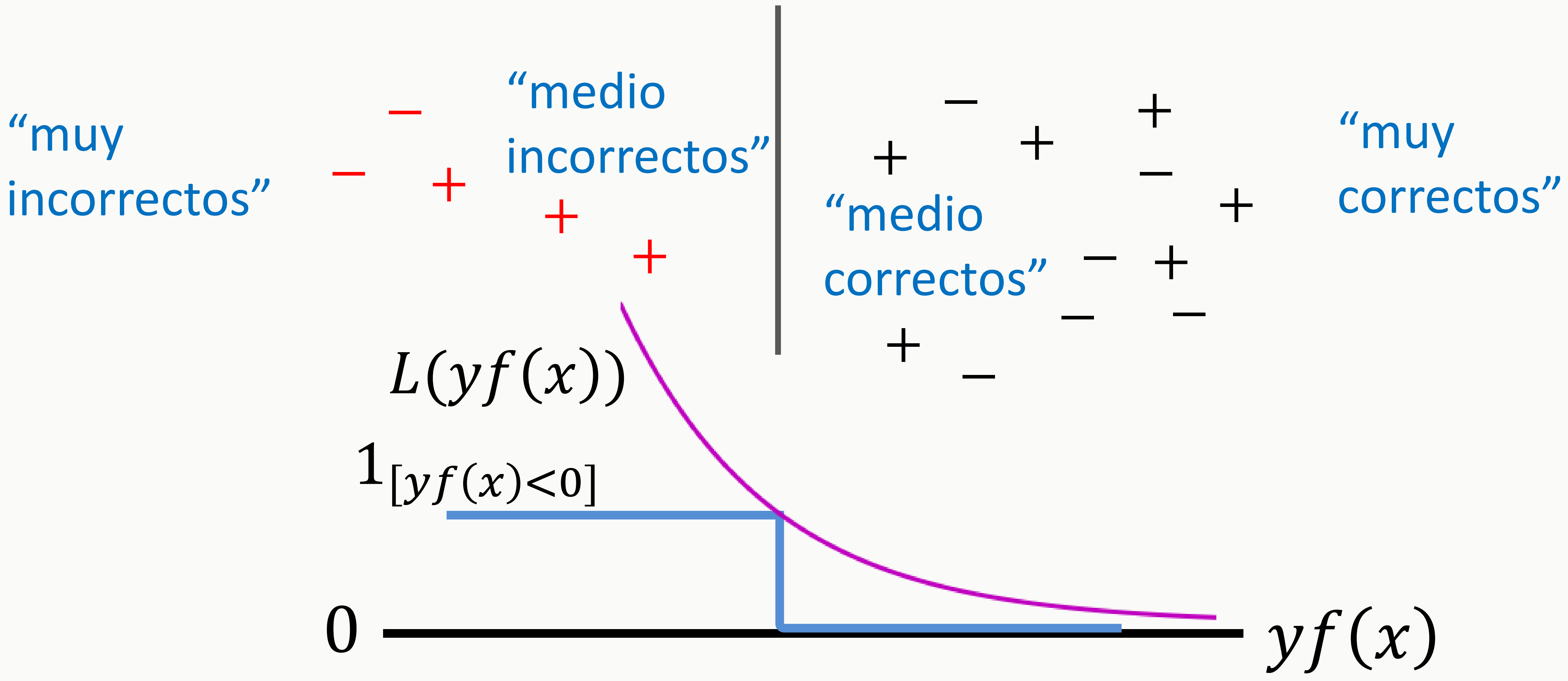
FUNCIÓN DE PÉRDIDA PARA CLASIFICACIÓN



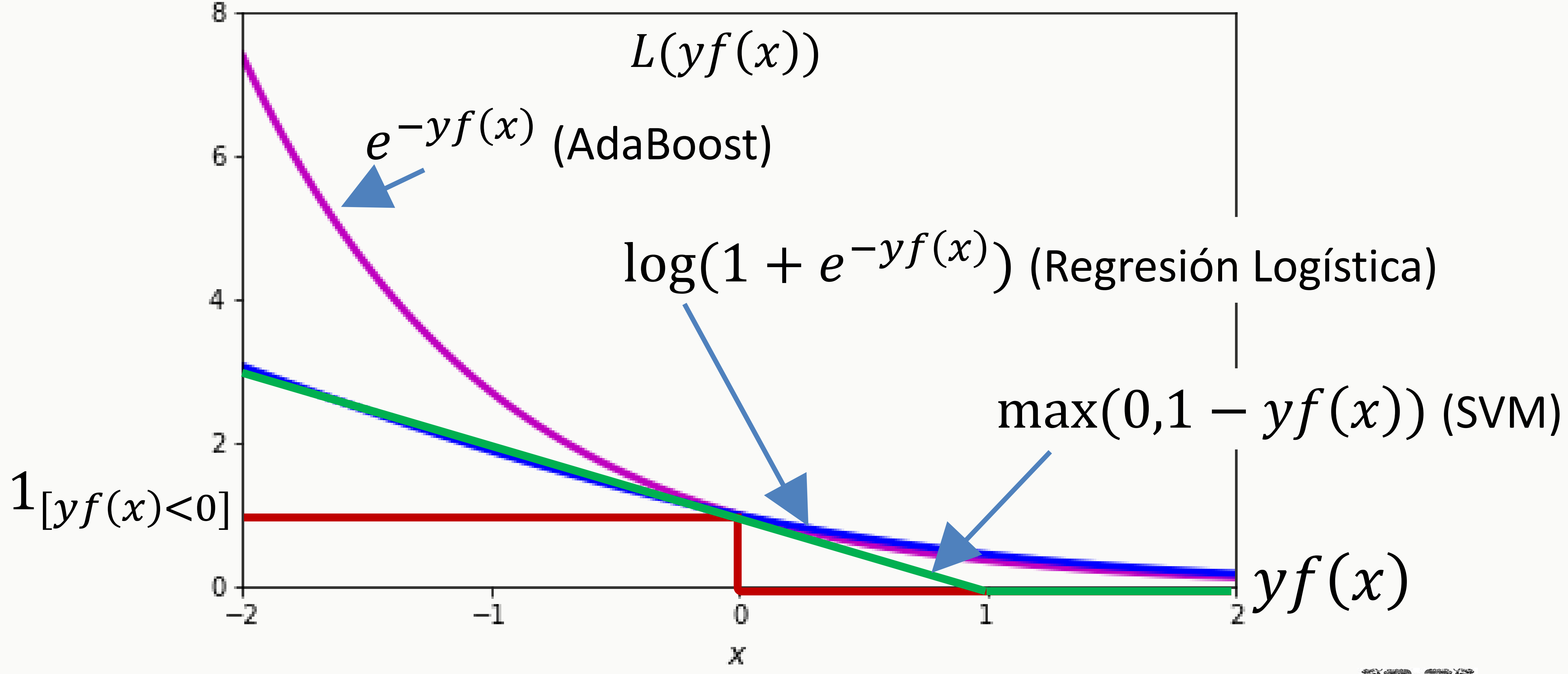
Centro de Investigación en Ciencias

Universidad Autónoma del Estado de Morelos

FUNCIÓN DE PÉRDIDA PARA CLASIFICACIÓN



FUNCIÓN DE PÉRDIDA PARA CLASIFICACIÓN



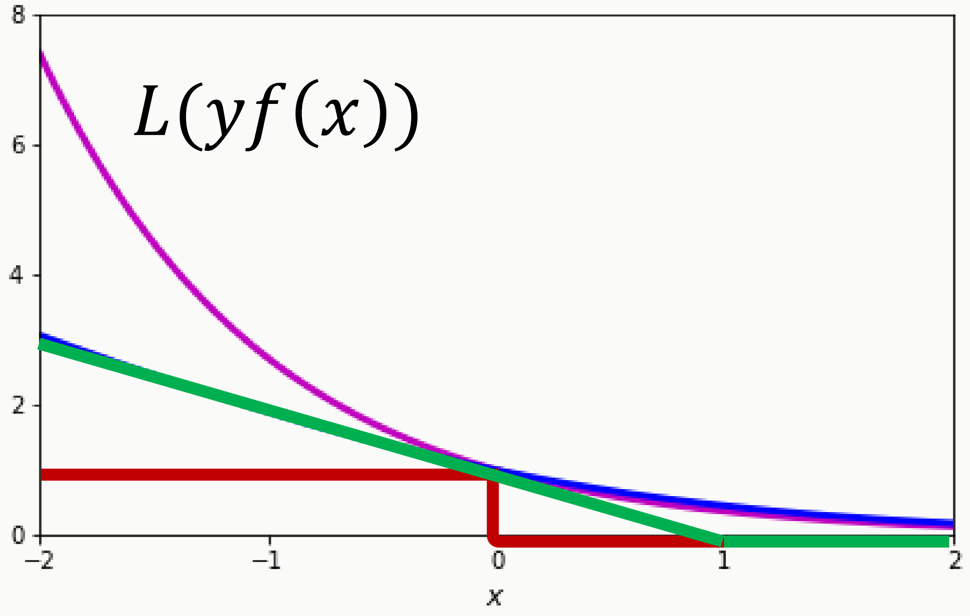
FUNCIÓN DE PÉRDIDA PARA CLASIFICACIÓN

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

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

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

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



FUNCIÓN DE PÉRDIDA PARA CLASIFICACIÓN

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

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

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

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

Objetivo

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

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^2 (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)

N=47

Tamaño en pies^2 (x)		Precio ($) en miles (y)	
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

Ejemplos de entrenamiento
$(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \cdots, (\boldsymbol{x}_n, y_n)$

Tamaño vs. precio

Algoritmo de Aprendizaje
$\mathcal{A}$

Hipótesis final
h

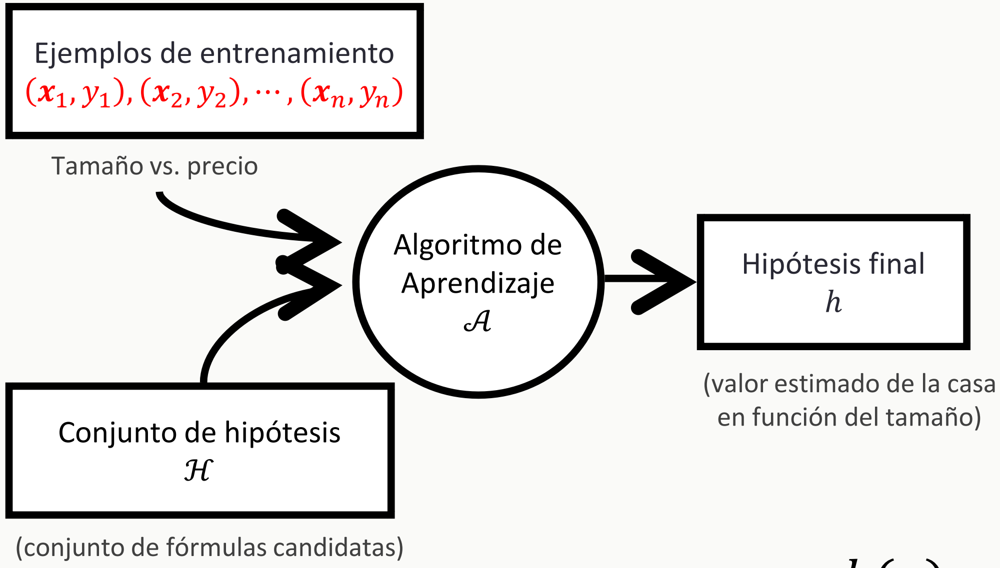
(valor estimado de la casa en función del tamaño)

Conjunto de hipótesis
$\mathcal{H}$

(conjunto de fórmulas candidatas)

$$y = h(x)$$

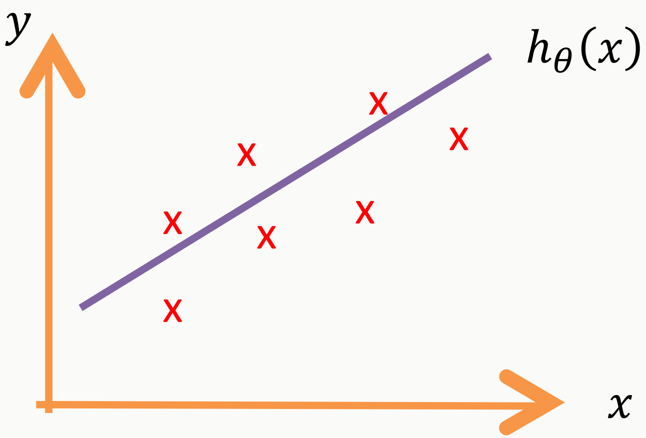
Problema de Aprendizaje



$$y = h(x)$$

¿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

N=47

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

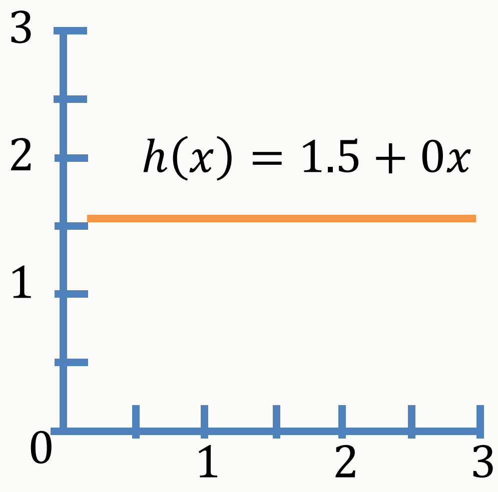
Hipótesis: $h_\theta(x) = \theta_0 + \theta_1 x$

θ_i’s : Parámetros

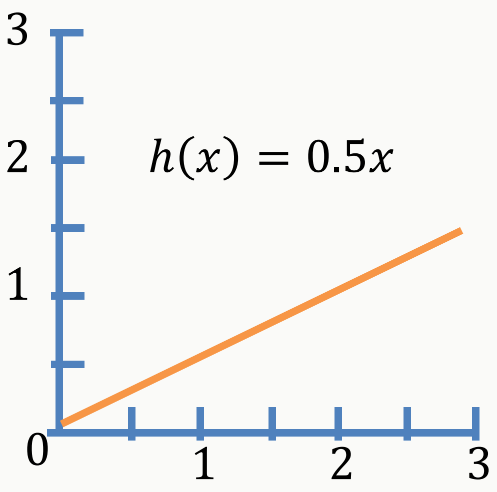
¿Cómo elegir los θ_i’s?

Hipótesis en función de los parámetros

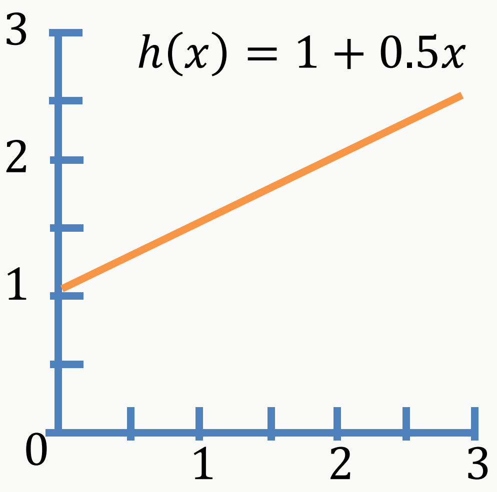
$$h_\theta(x) = \theta_0 + \theta_1 x$$



$\theta_0 = 1.5$
$\theta_1 = 0$

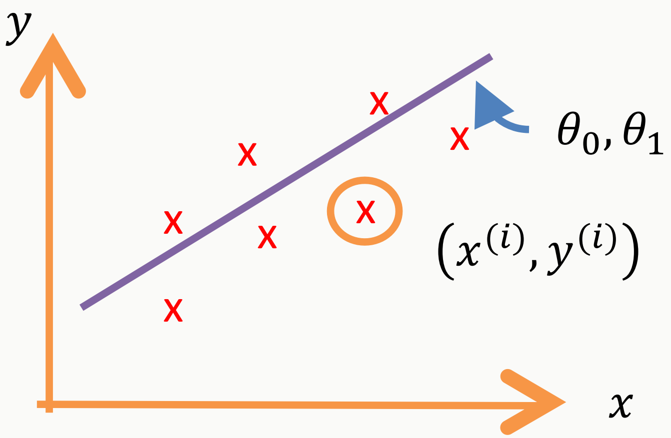


$\theta_0 = 0$
$\theta_1 = 0.5$



$\theta_0 = 1$
$\theta_1 = 0.5$

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)

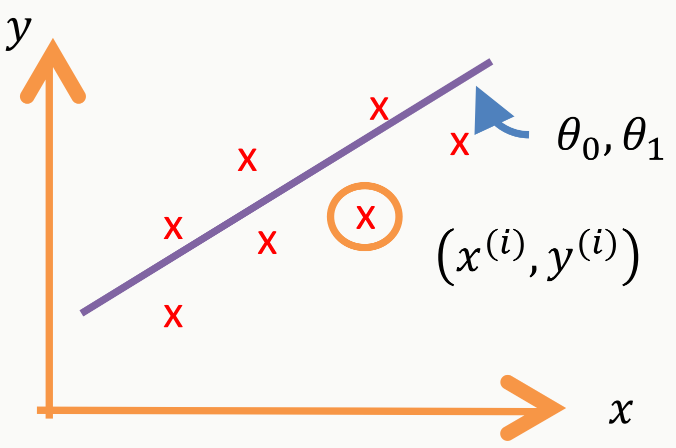
$\Rightarrow$

Función de error cuadrático:

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

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

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)

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

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

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

FUNCIÓN DE COSTO

Intuición I

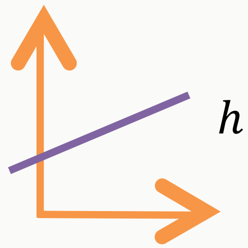
Problema de minimización

Hipótesis:

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

Parámetros:

$$\theta_0, \theta_1$$



$h(x)$

Función de costo:

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

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

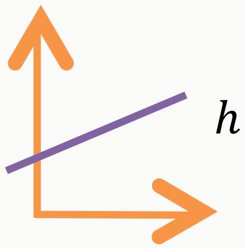
Minimización simplificada

Hipótesis:

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

Parámetros:

θ_0, θ_1



$h(x)$

Función de costo:

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

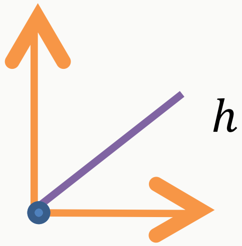
Objetivo: $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$

<u>Simplificación:</u>

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

Parámetros:

θ_1



$h(x)$

Función de costo:

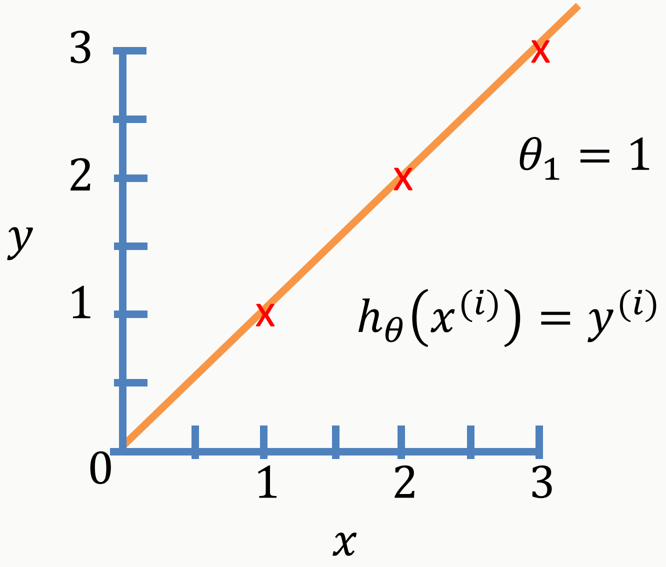
$$J(\theta_1) = \frac{1}{2N}\sum_{i=1}^{N}\left(\underbrace{h_\theta\left(x^{(i)}\right)}_{\theta_1 x^{(i)}} - y^{(i)}\right)^2$$

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

Hipótesis *vs.* Costo

$h_\theta(x)$

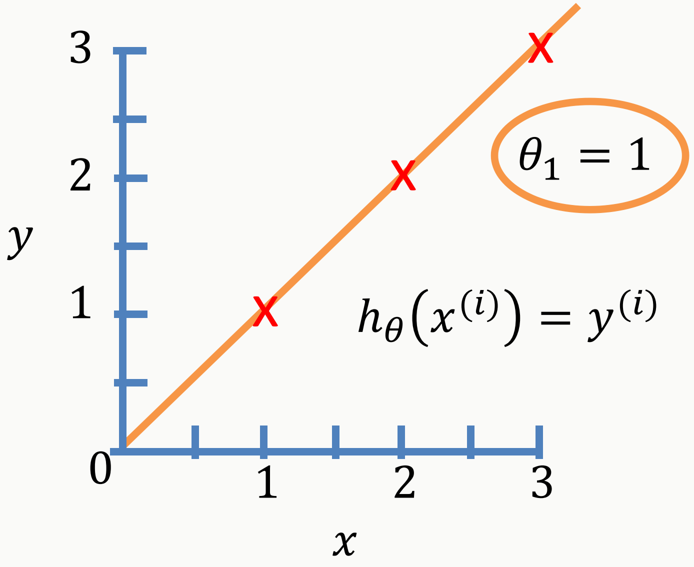
(para un θ_1 fijo, esto es una función de x)



Hipótesis *vs.* Costo

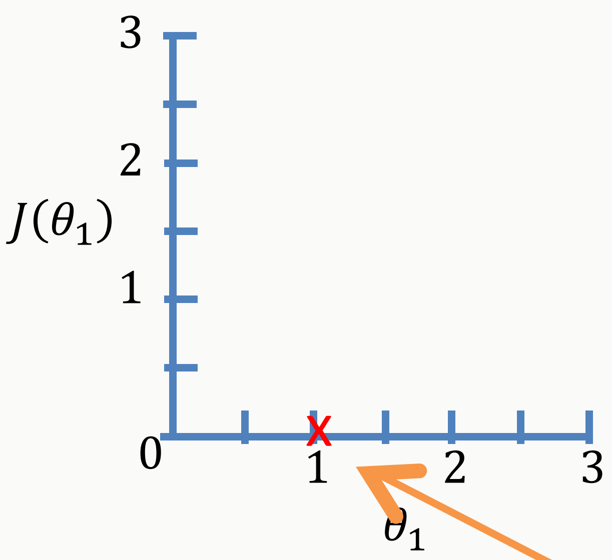
$h_\theta(x)$

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



$J(\theta_1)$

(función del parámetro θ_1)



$\theta_1 = 0.5?$

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

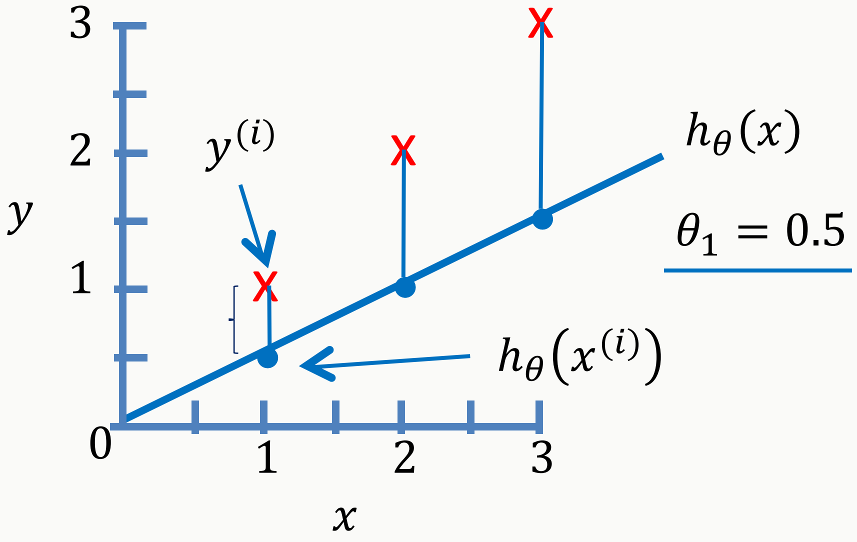
$J(1) = 0$

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

Hipótesis *vs.* Costo

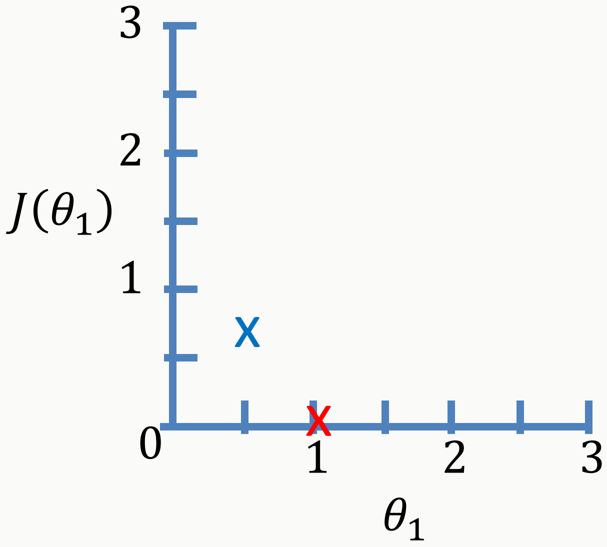
$h_\theta(x)$

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



$J(\theta_1)$

(función del parámetro θ_1)



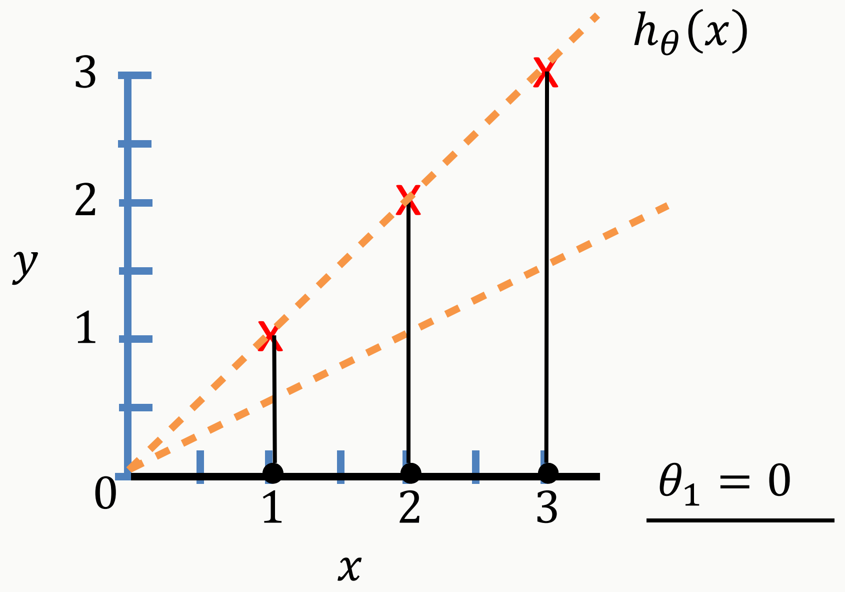
$\theta_1 = 0?$

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

Hipótesis *vs.* Costo

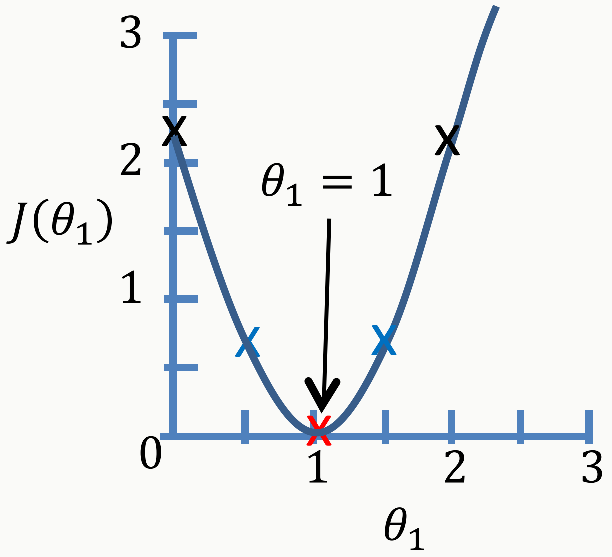
$h_\theta(x)$

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



$J(\theta_1)$

(función del parámetro θ_1)



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

FUNCIÓN DE COSTO

Intuición II

Problema de minimización de costo

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

Parámetros: θ_0, θ_1

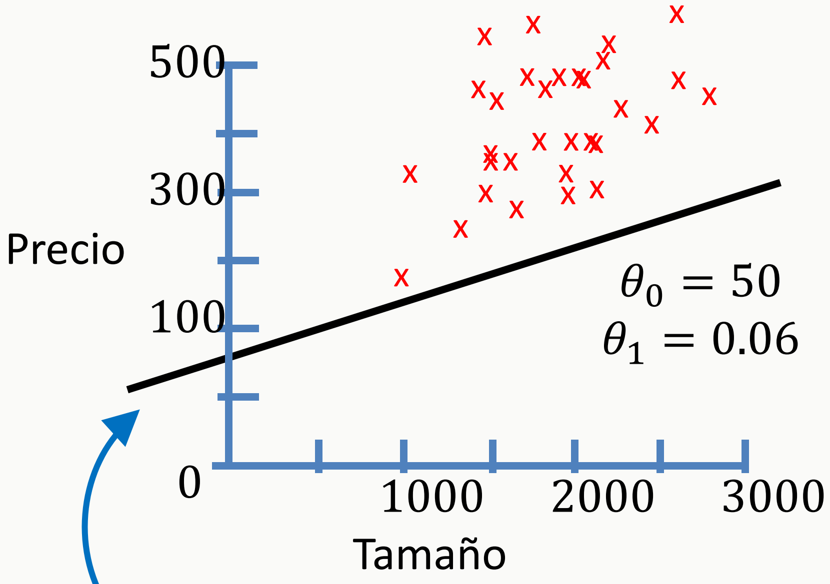
Función de costo: $J(\theta_0, \theta_1) = \frac{1}{2N}\sum_{i=1}^{N}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$

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

Hipótesis *vs.* Costo

$h_\theta(x)$

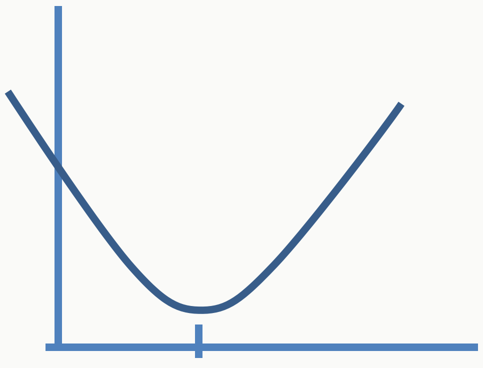
(para un θ_1, θ_2fijos, esto es una función de x)



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

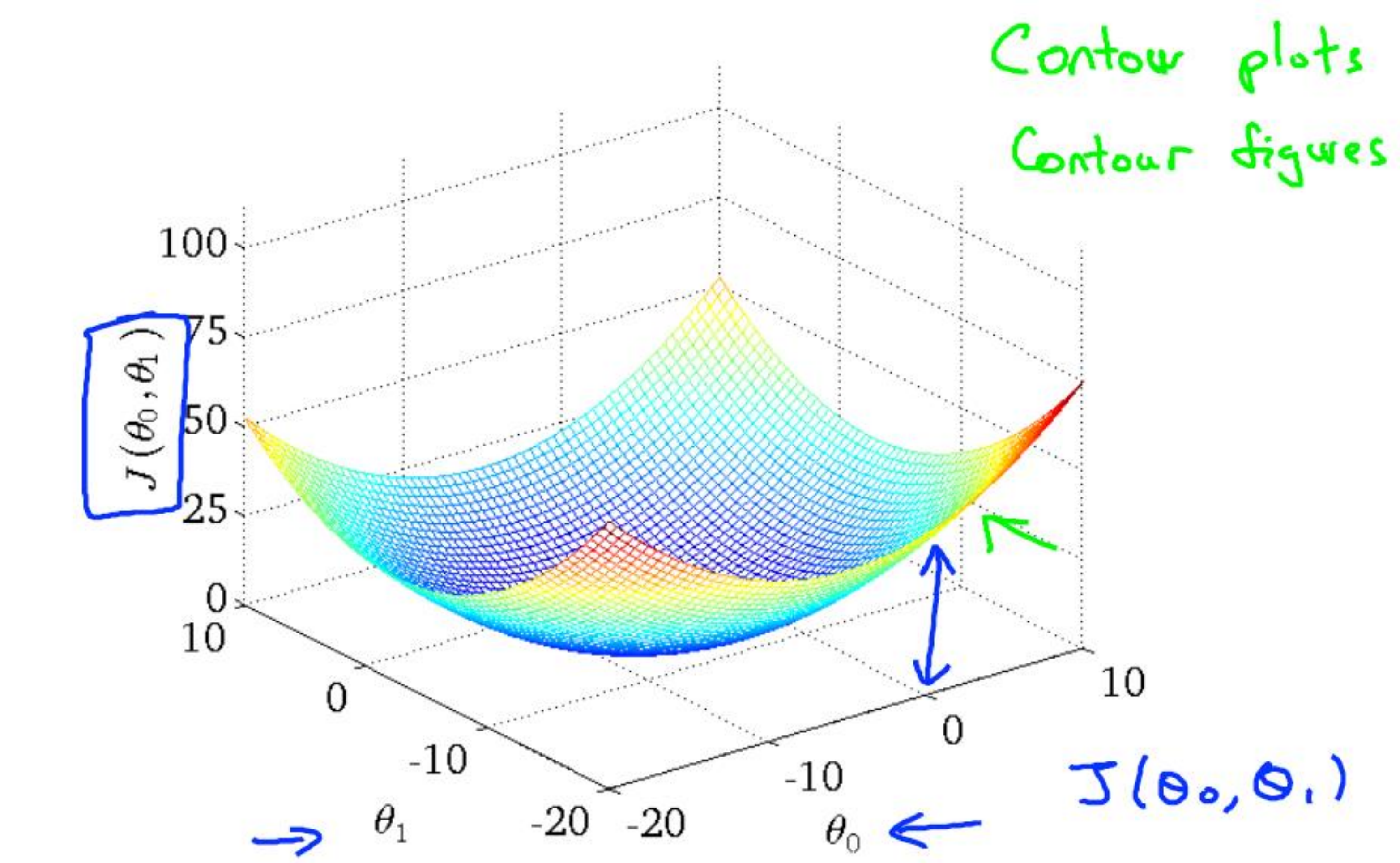
$J(\theta_0, \theta_1)$

(función de los parámetros θ_0, θ_1)



$\boxed{\theta_0, \theta_1}$

Gráficos de contorno



Centro de Investigación en Ciencias

Universidad Autónoma del Estado de Morelos

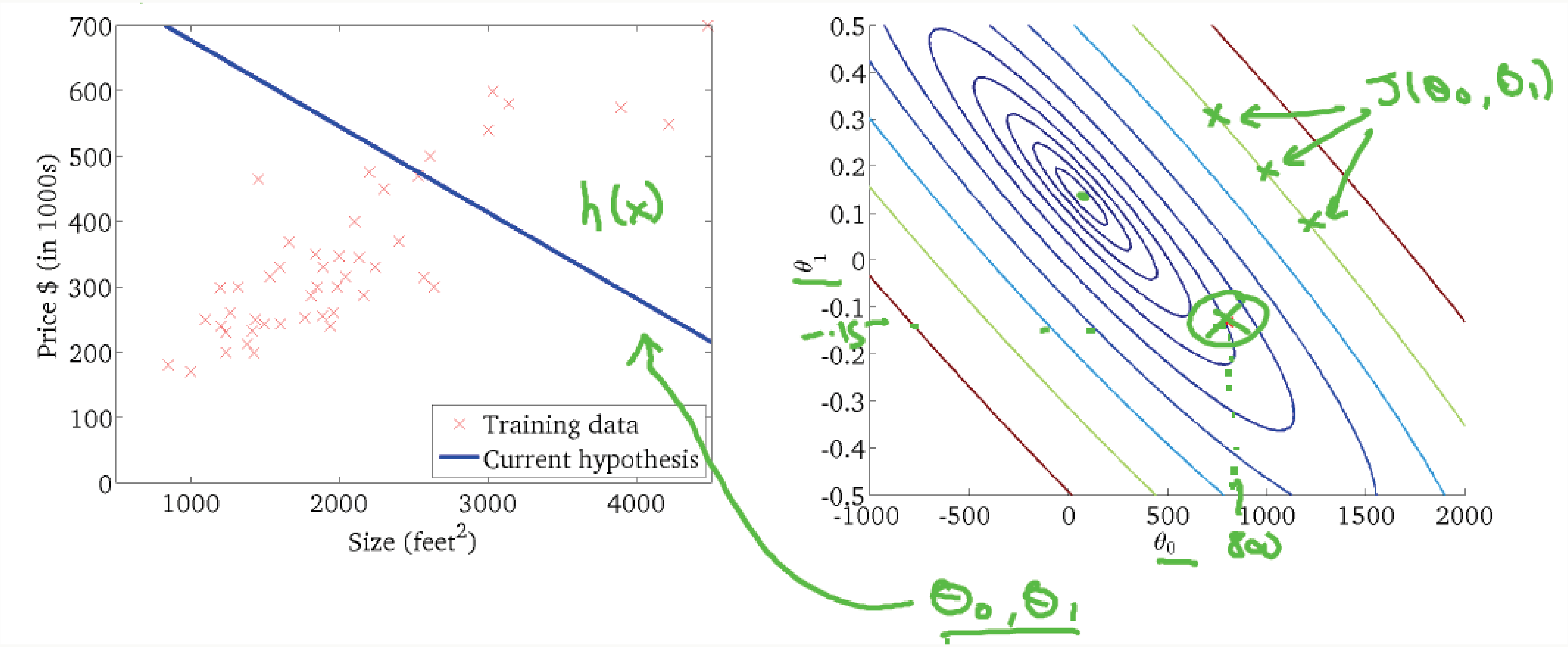
Hipótesis *vs.* Costo

$h_\theta(x)$

(para un θ_1, θ_2fijos, esto es una función de x)

$J(\theta_0, \theta_1)$

(función de los parámetros θ_0, θ_1)



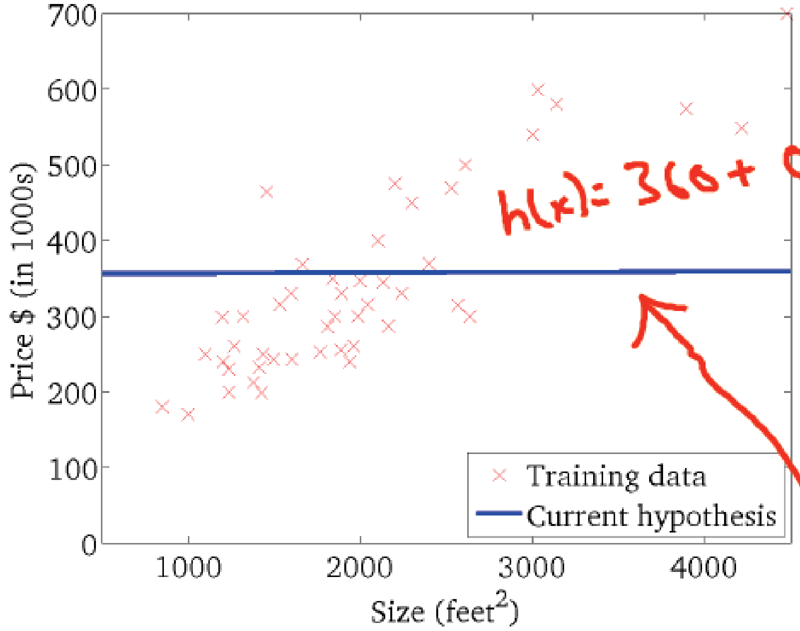
Hipótesis *vs.* Costo

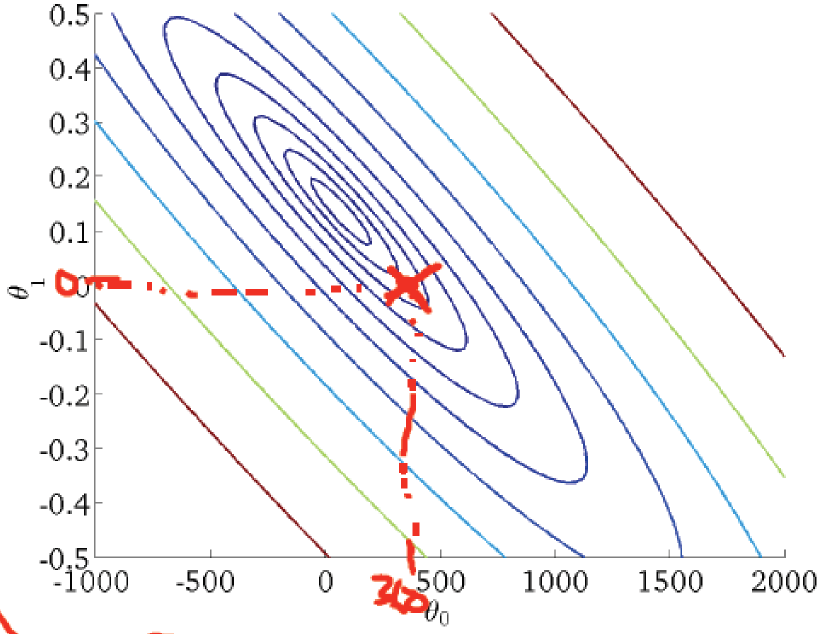
$h_\theta(x)$

(para un θ_1, θ_2fijos, esto es una función de x)

$J(\theta_0, \theta_1)$

(función de los parámetros θ_0, θ_1)





$\theta_0 = 360$

$\theta_1 = 0$

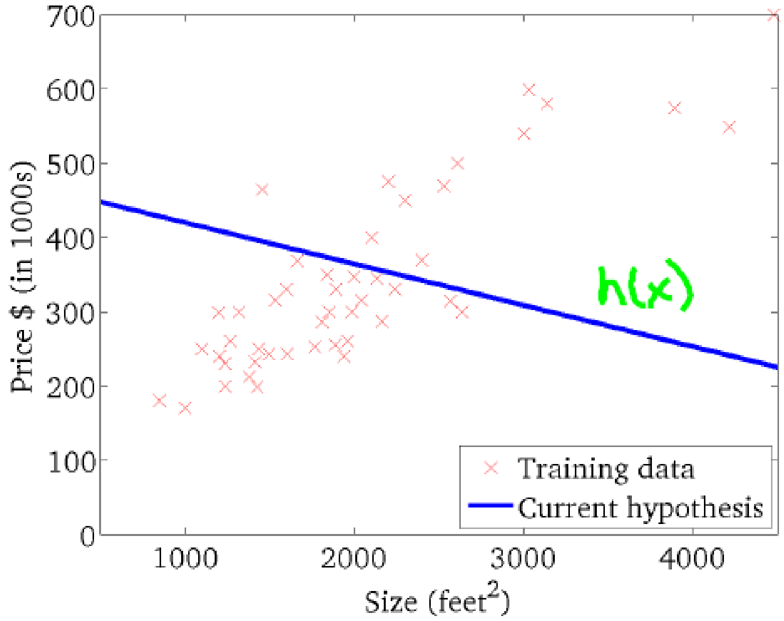
Hipótesis *vs.* Costo

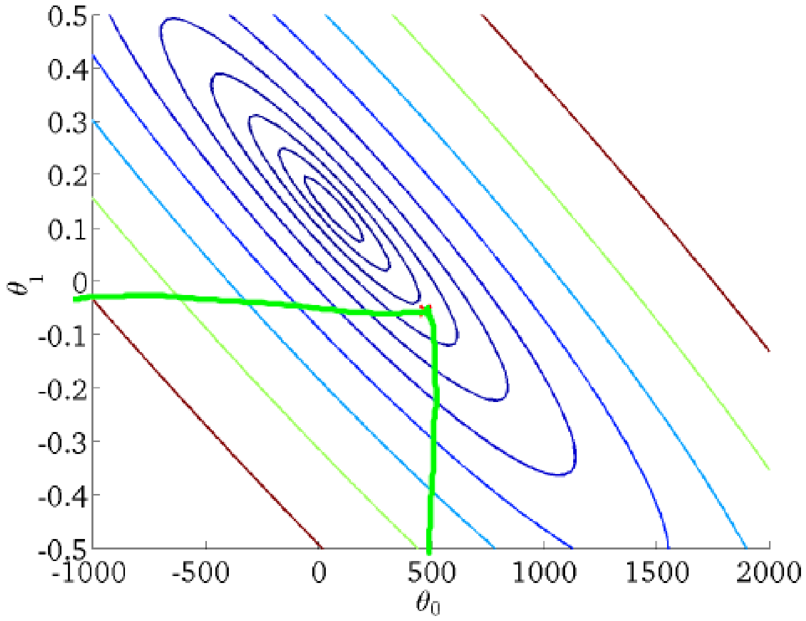
$h_\theta(x)$

(para un θ_1, θ_2fijos, esto es una función de x)

$J(\theta_0, \theta_1)$

(función de los parámetros θ_0, θ_1)

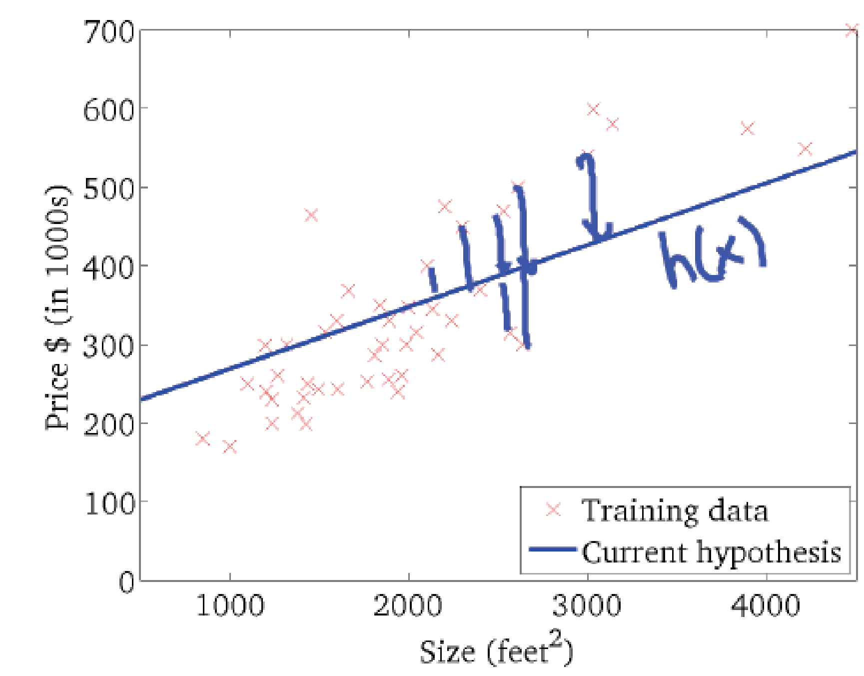




Hipótesis *vs.* Costo

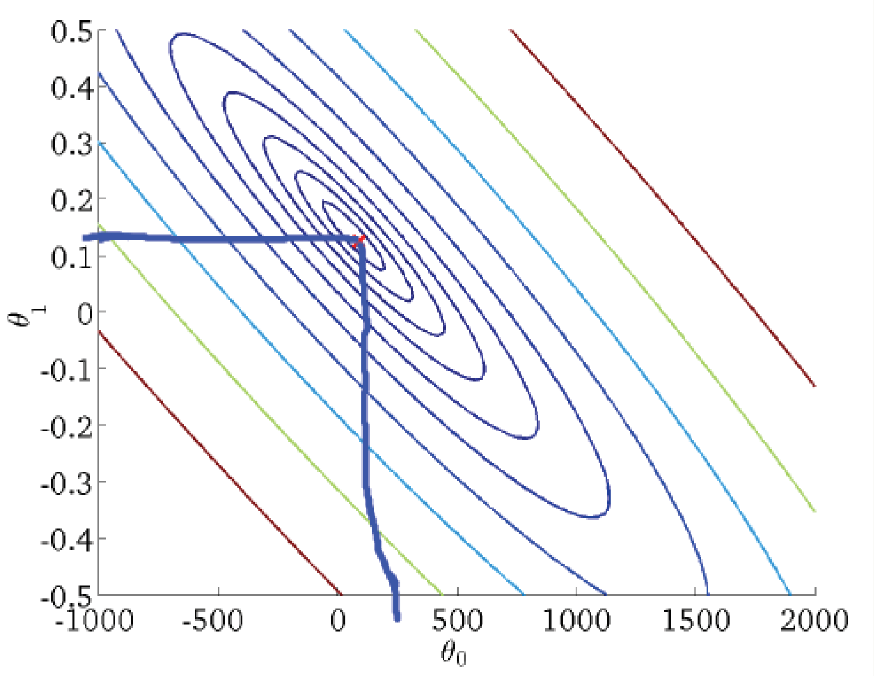
$h_\theta(x)$

(para un θ_1, θ_2fijos, 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

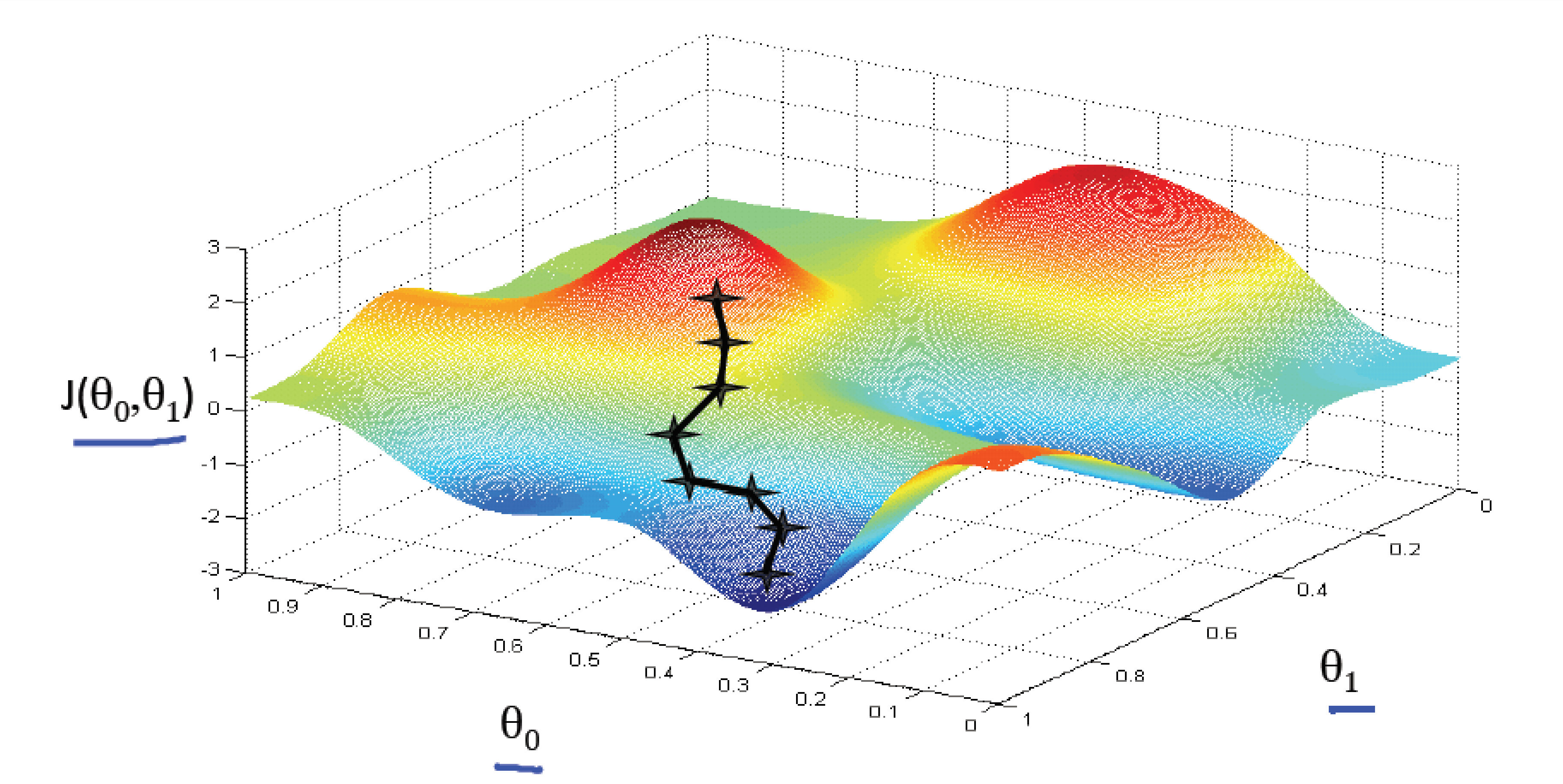
Función de costo: $J(\theta_0, \theta_1)$

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

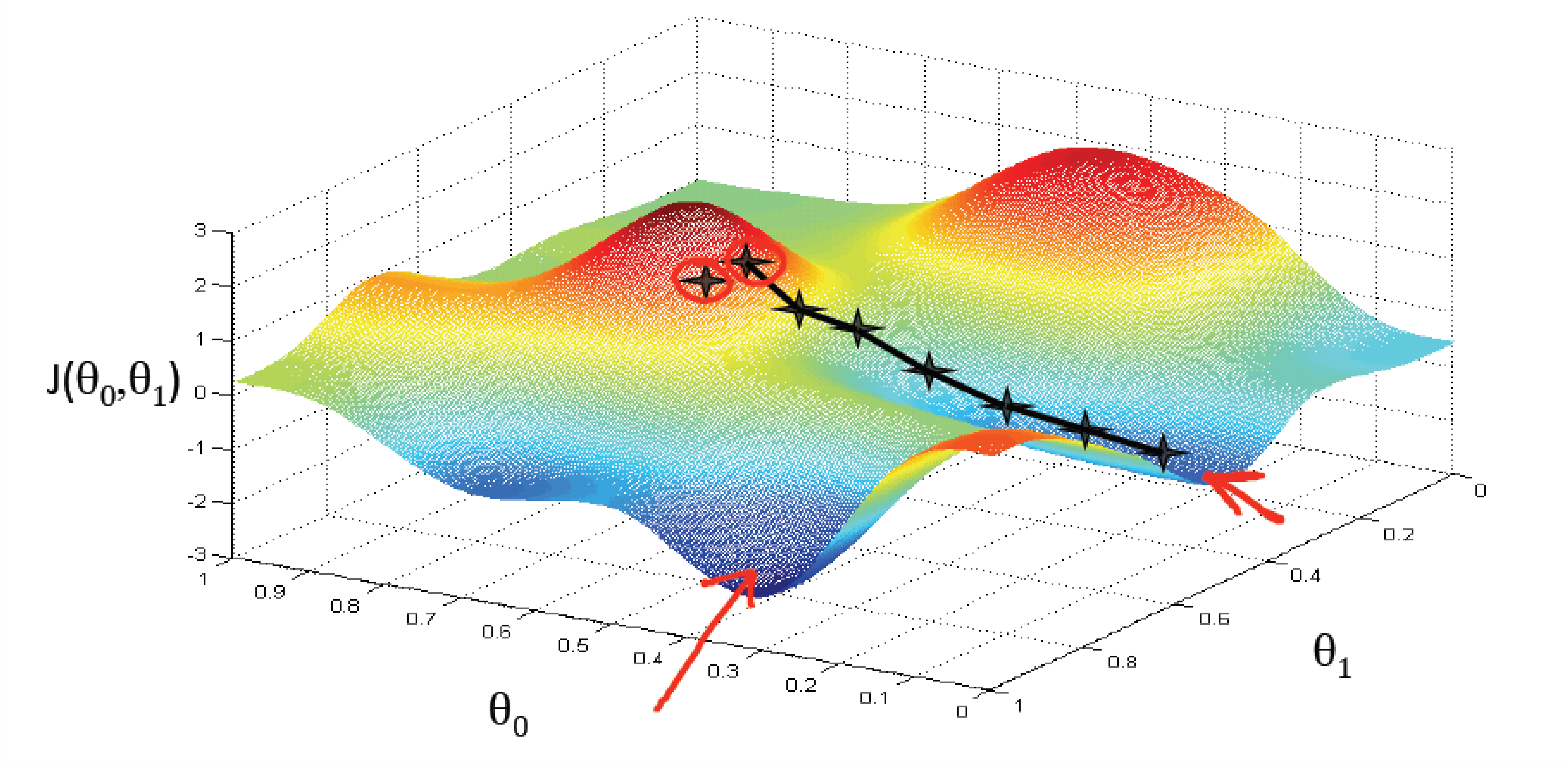
Descripción resumida:

- 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 {

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

}

Asignación simultánea!!!

Correcto: actualización simultánea

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

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

$$\theta_0 := temp0$$

$$\theta_1 := temp1$$

Incorrecto:

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

$$\theta_0 := temp0$$

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

$$\theta_1 := temp1$$

INTUICIÓN DEL DESCENSO DE GRADIENTE

Algoritmo de descenso de gradiente

Repite hasta la convergencia {

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

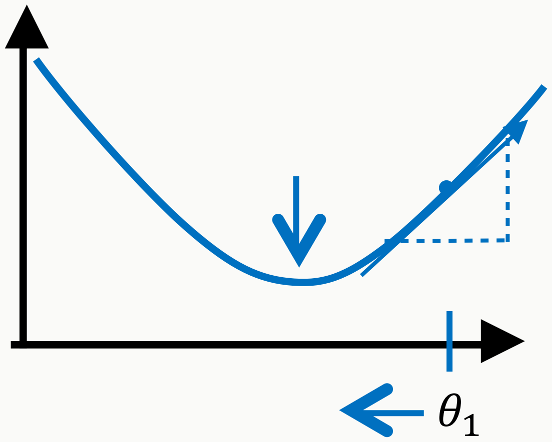
}

tasa de aprendizaje

derivada

$$\min_{\theta_1} J(\theta_1) \qquad \theta_1 \in \mathbb{R}$$

Actualización en función de la pendiente

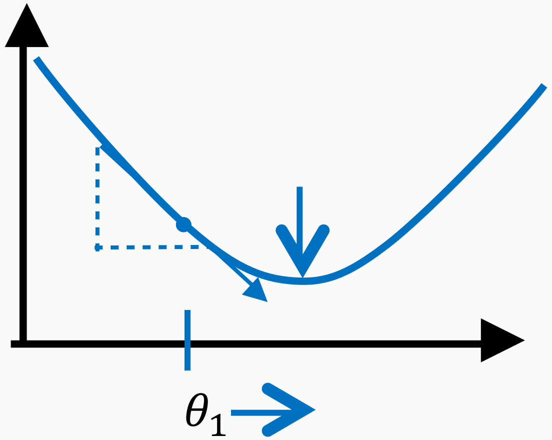


$J(\theta_1) \quad (\theta_1 \in \mathbb{R})$

pendiente +

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

≥ 0



$J(\theta_1)$

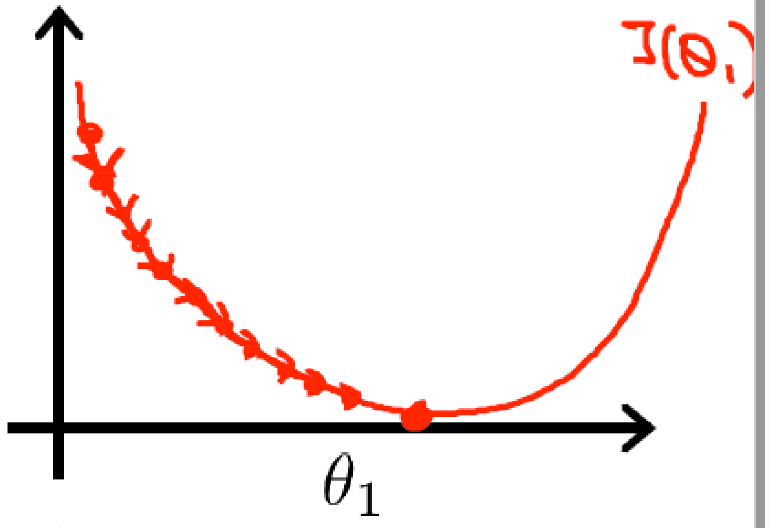
pendiente -

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

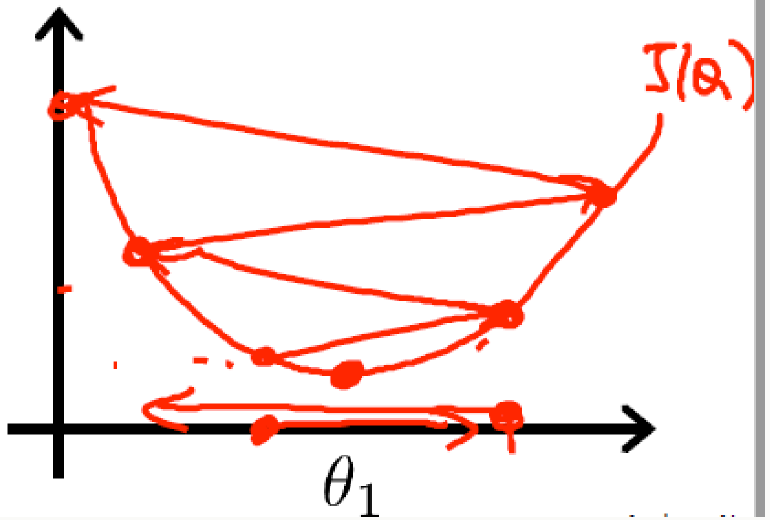
≤ 0

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

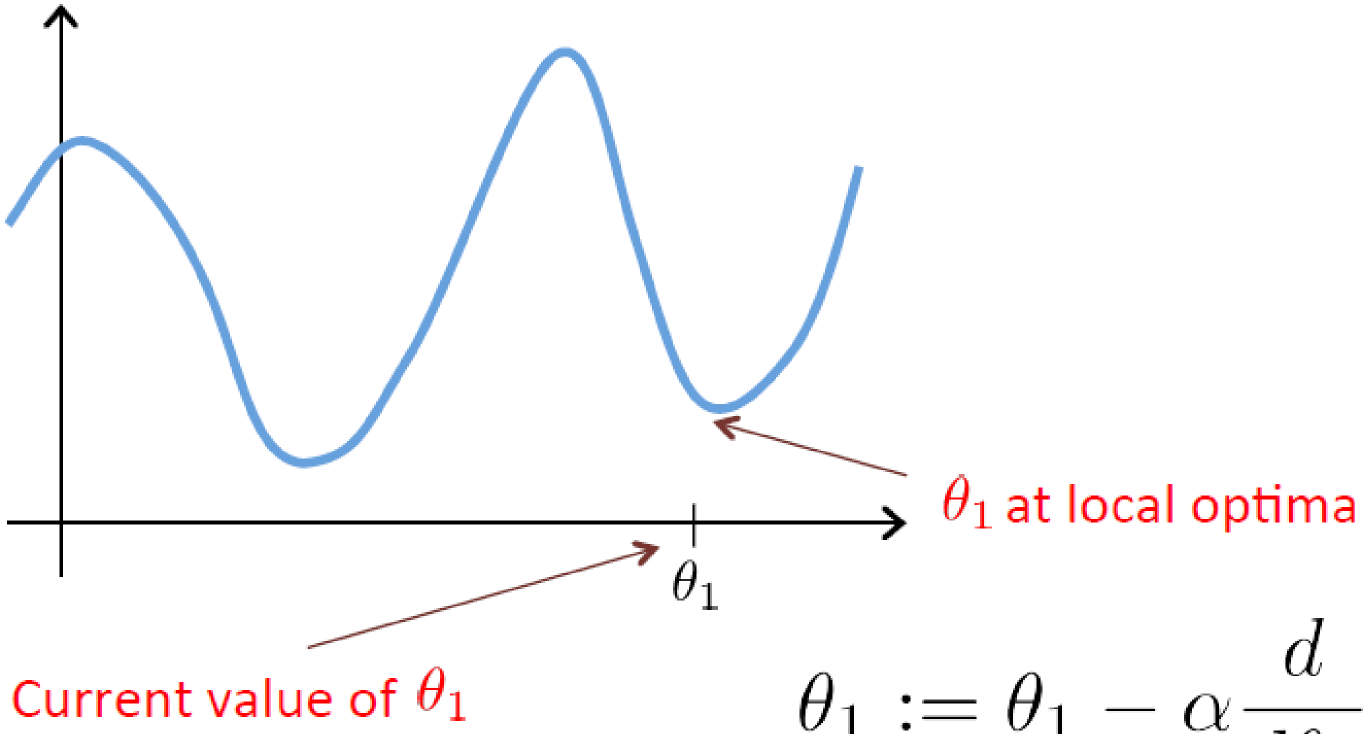
If α is too small, gradient descent can be slow.



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



Mínimos locales



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

Convergencia

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

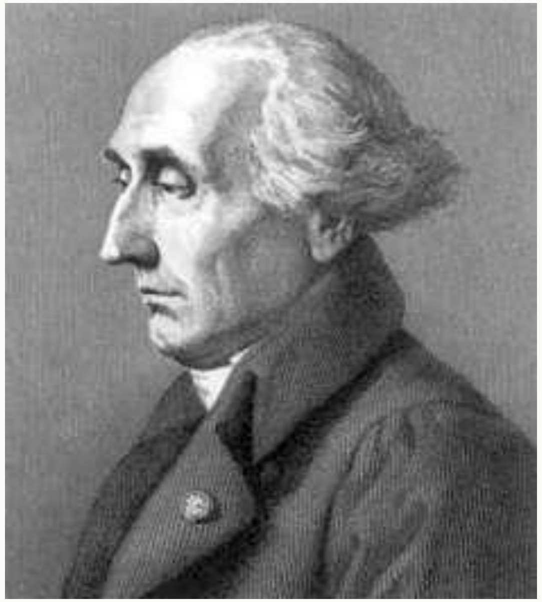
$$\theta_1 := \theta_1 - \alpha \boxed{\frac{\mathrm{d}}{\mathrm{d}\theta_j} J(\theta_1)}$$

- 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.

OPTIMIZACIÓN POR MULTIPLICADORES DE LAGRANGE

Nociones básicas

Joseph Louis de Lagrange



Bautizado como **Giuseppe Lodovico Lagrangia**, fue un Físico, Matemático y Astrónomo italiano naturalizado francés.

JOSEPH LOUIS LAGRANGE. Senador. Conde del Imperio. Gran Oficial de la Legión de Honor. Gran Cruz de la Imperial Orden de la Reunión. Miembro del Instituto y la Oficina de Longitudes. Nacido en Turín el 25 de enero de 1736. Muerto en París el 10 de abril de 1813.

Profesor de Joseph Fourier quien, aparentemente, no lo tenía en muy buen concepto.

Fuente: Wikipedia

MÉTODO DE LAGRANGE

- La técnica del multiplicador de Lagrange permite encontrar el máximo o mínimo de una función multivariable $f(x_1, x_2, \dots, x_N)$ cuando hay alguna restricción en los valores de entrada del tipo:

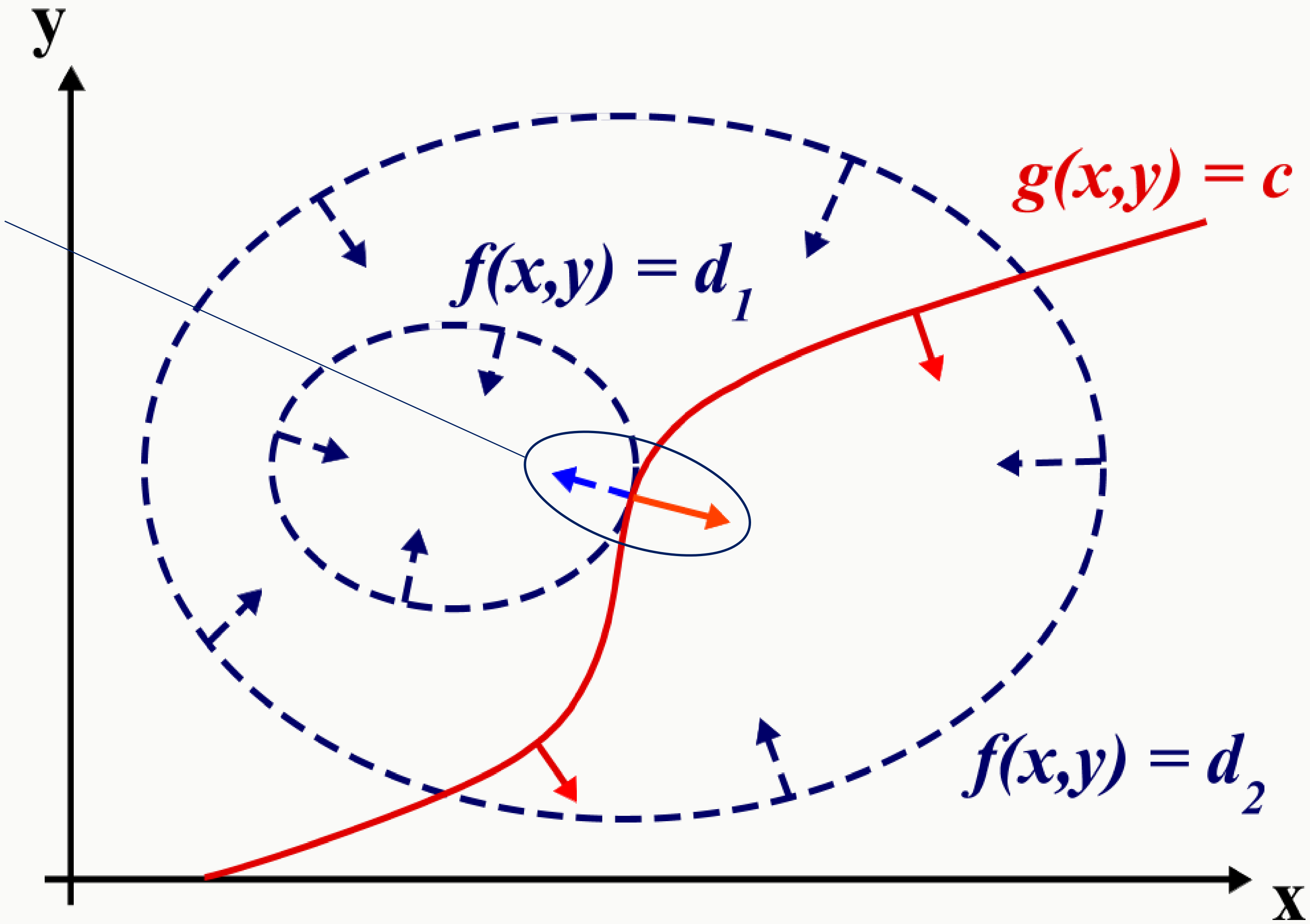
$$\mathrm{g}(x_1, x_2, \dots, x_N) = c \qquad c \in \mathbb{R}$$

- La idea central es buscar puntos en los que las líneas de contorno de $f(x_1, x_2, \dots, x_N)$ y $\mathrm{g}(x_1, x_2, \dots, x_N)$ son tangentes entre sí.

CONDICIÓN DE TANGENCIA

Puntos donde los vectores de gradiente de ambas curvas son paralelos.

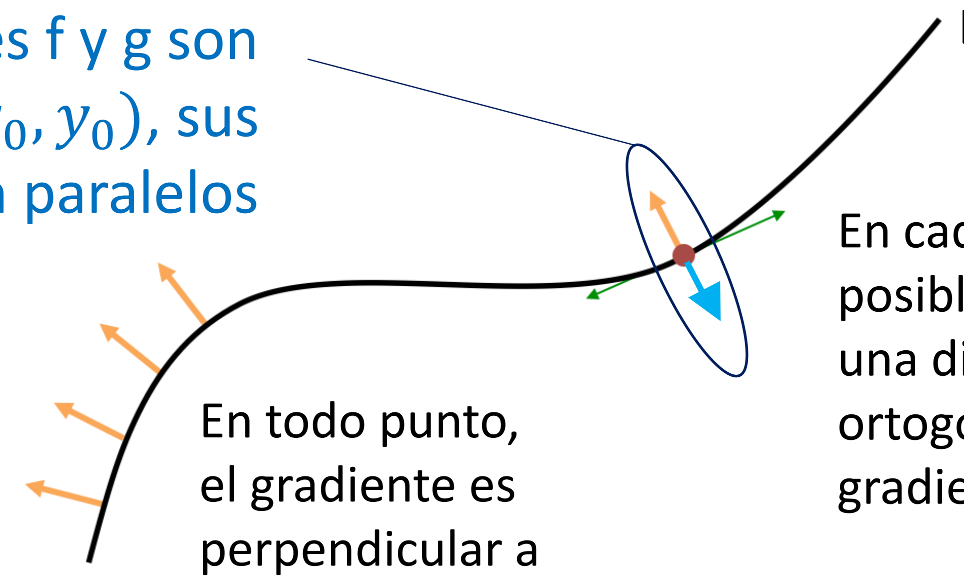
¡Veamos el gradiente!



CONDICIÓN DE TANGENCIA

- Como hemos mencionado el gradiente ∇f de f evaluado en un punto (x_0, y_0) siempre da un vector perpendicular a la línea de contorno que pasa por ese punto.

Esto significa que cuando las líneas de contorno de dos funciones f y g son tangentes en un punto (x_0, y_0), sus vectores de gradiente son paralelos en ese punto.



MULTIPLICADOR DE LAGRANGE

- La condición de paralelismo se traduce en :

$$\nabla f(x_1, x_2, \dots, x_N) = \lambda_0 \nabla \mathrm{g}(x_1, x_2, \dots, x_N) \qquad (1)$$

- Para una función $f(x_1, x_2, \dots, x_N)$:

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_N} \end{bmatrix}$$

MULTIPLICADOR DE LAGRANGE

- Notación práctica: $\nabla f = \left(\partial_{x_1} f, \partial_{x_2} f, \cdots, \partial_{x_N} f\right)$

- En esta forma (1) se puede desglosar como:

$$
\begin{gathered}
\partial_{x_1} f = \lambda_0 \partial_{x_1} g \\
\partial_{x_2} f = \lambda_0 \partial_{x_2} g \\
\vdots \\
\partial_{x_N} f = \lambda_0 \partial_{x_N} g
\end{gathered}
$$

LAGRANGIANO

- Lagrange propuso una nueva función especial (el Lagrangiano) que recoge las mismas variables de entrada que f y g, junto con λ que ahora se considera una variable en lugar de una constante:

$$\mathcal{L}(x_1, x_2, \ldots, x_N, \lambda) = f(x_1, x_2, \ldots, x_N) - \lambda(\mathrm{g}(x_1, x_2, \ldots, x_N) - c)$$

- Los puntos de tangencia se encuentran calculando:

$$\nabla\mathcal{L}(x_1, x_2, \ldots, x_N, \lambda) = 0$$

EJEMPLO

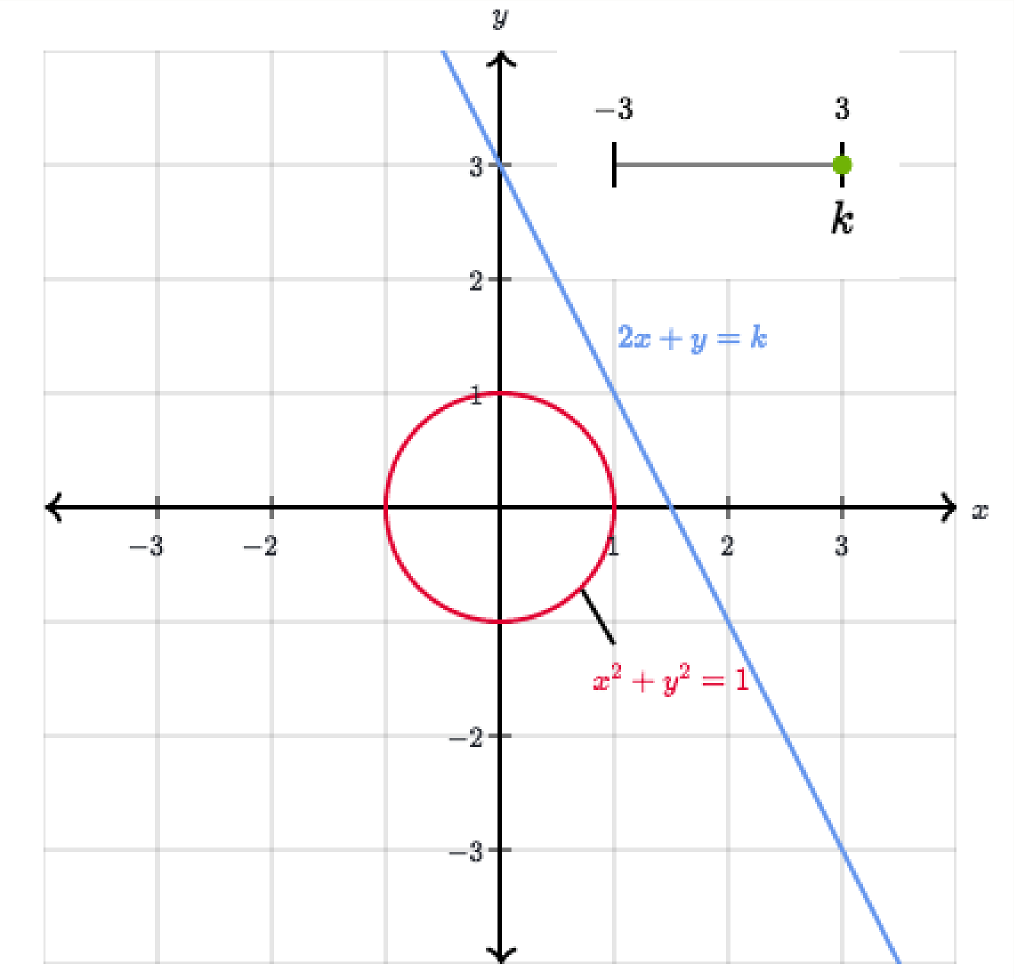
- Encuentre el valor máximo de las lineas de contorno f:

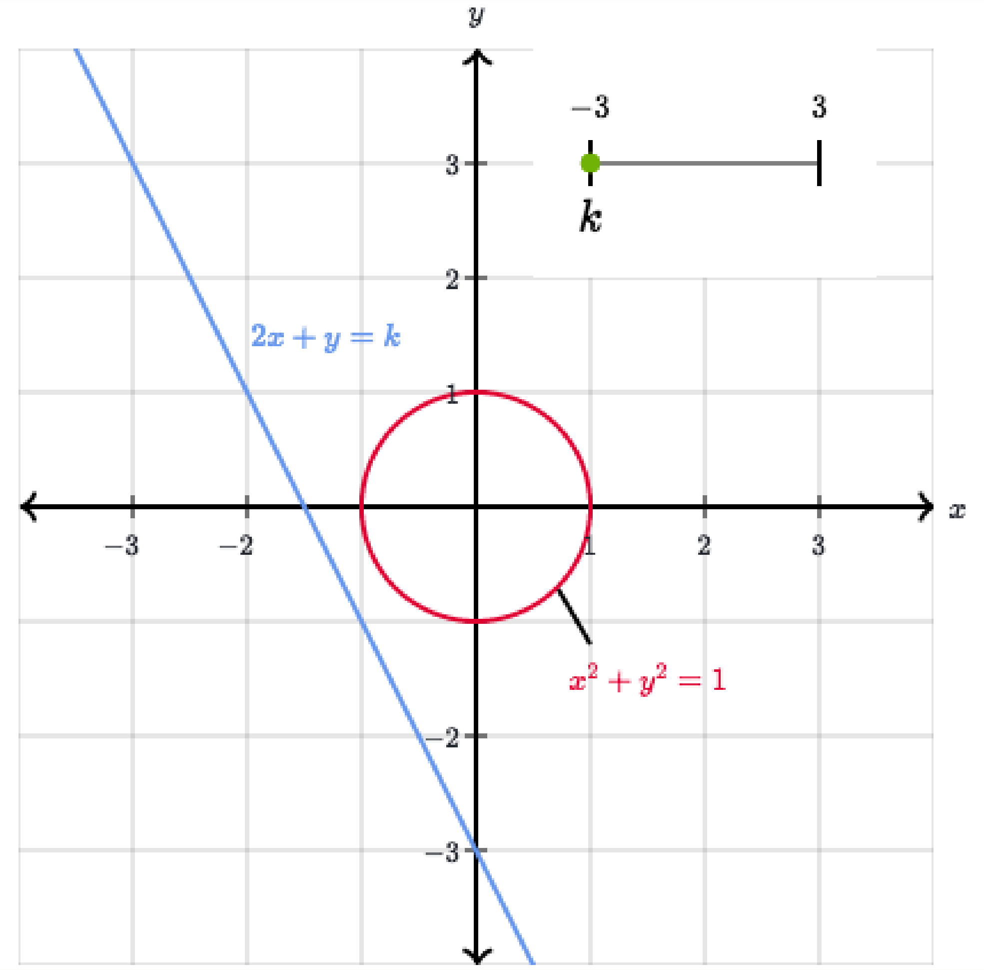
$$f(x, y) = 2x + y = k$$

con la restricción sobre la entrada (x, y) tal que éstas deben cumplir con:

$$g(x, y) = x^2 + y^2 = 1$$

EJEMPLO





Centro de Investigación en Ciencias

Universidad Autónoma del Estado de Morelos

EJEMPLO (LAGRANGIANO)

- *Para este ejemplo el Lagrangiano se escribe:*

$$\mathcal{L}(x, y, \lambda) = f(x, y) - \lambda(\mathrm{g}(x, y) - 1)$$
$$= 2x + y \; - \lambda(x^2 + y^2 - 1)$$

- *Podemos ver que nuestra restricción se escribe:*

$$\partial_\lambda \mathcal{L}(x, y, \lambda) = 0 \;\Rightarrow \mathrm{g}(x, y) = 1$$

EJEMPLO (LAGRANGIANO)

- *También vemos que* $\nabla\mathcal{L}$ *tiene dos componentes:*

$$\nabla f = \begin{bmatrix} 2 \\ 1 \end{bmatrix} y\; \nabla g = \begin{bmatrix} 2x \\ 2y \end{bmatrix}$$

- *Por lo que la condición de tangencia* $\nabla\mathcal{L} = 0$ *se escribe:*

$$\begin{bmatrix} 2 \\ 1 \end{bmatrix} = \lambda_0 \begin{bmatrix} 2x_0 \\ 2y_0 \end{bmatrix}$$

EJEMPLO (EL PROBLEMA)

- Resumiendo buscamos:
  - *Un punto* (x_0, y_0) *tal que:* $\mathrm{g}(x_0, y_0) = 1$ *que para el ejemplo eso es:*

$$x_0^2 + y_0^2 = 1$$

  - *Además este punto es tal que:*

$$\begin{cases} 2\lambda_0 x_0 = 2 \\ 2\lambda_0 y_0 = 1 \end{cases}$$

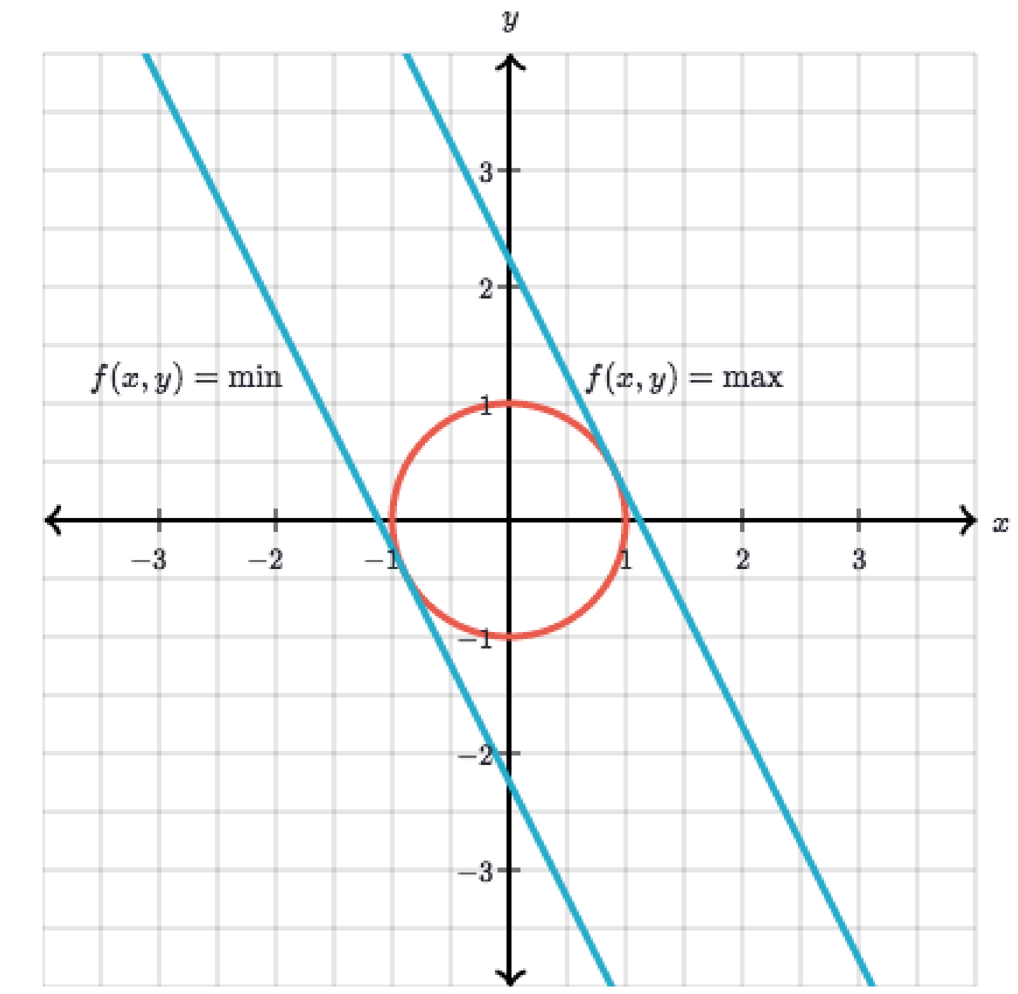
EJEMPLO (LA SOLUCIÓN)

$$\lambda_0 = \frac{\pm\sqrt{5}}{2} \qquad (x_0, y_0) = \left(\frac{1}{\lambda_0}, \frac{1}{2\lambda_0}\right)$$

$$(x_0, y_0) = \left(\frac{2}{\sqrt{5}}, \frac{1}{\sqrt{5}}\right) \Longrightarrow f(x_0, y_0) = \max$$

o bien

$$(x_0, y_0) = \left(\frac{-2}{\sqrt{5}}, \frac{-1}{\sqrt{5}}\right) \Longrightarrow f(x_0, y_0) = \min$$



OPTIMIZACIÓN CON RESTRICCIONES DE IGUALDAD

- Paso 1: Define el lagrangiano

$$\mathcal{L}(x, y, \cdots, \lambda) = f(x, y, \ldots) - \lambda(\mathrm{g}(x, y, \ldots) - c)$$

- Paso 2: Plantea las ecuaciones de soluciones óptimas

$$\nabla\mathcal{L}(x, y, \cdots, \lambda) = 0$$

- Paso 3: Resuelve las ecuaciones.

OPTIMIZACIÓN CON RESTRICCIONES DE DESIGUALDAD

- Se consideran ahora restricciones del tipo $g(x) \leq 0, g(x) \geq 0$.
- Básicamente la estrategia es la misma que la descrita anteriormente, minimizando el lagrangiano con condiciones de igualdad y luego aplicando una de las siguientes reglas:

$$g(x) \geq 0 \Rightarrow \lambda \geq 0$$

$$g(x) \leq 0 \Rightarrow \lambda \leq 0$$

$$g(x) = 0 \Rightarrow \lambda \text{ no está restringida}$$