IIC 2433 Minería de Datos

https://github.com/marcelomendoza/IIC2433

- CLASIFICACIÓN -

Clasificación

¿Cuáles de estos ejemplos son árboles?








Clasificación

$\textit{input } \mathbf{x} \in \mathbb{R}^d = \mathcal{X}.$

$\textit{output } y \in \{-1, +1\} = \mathcal{Y}.$ (Clasificación binaria)

$\textit{target function } f : \mathcal{X} \mapsto \mathcal{Y}.$ ← f desconocida

$\textit{data set } \mathcal{D} = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N).$ ← Escenario supervisado

Objetivo de aprendizaje:

Queremos **aprender** f usando $\mathcal{D}$.

Clasificación

El algoritmo de aprendizaje

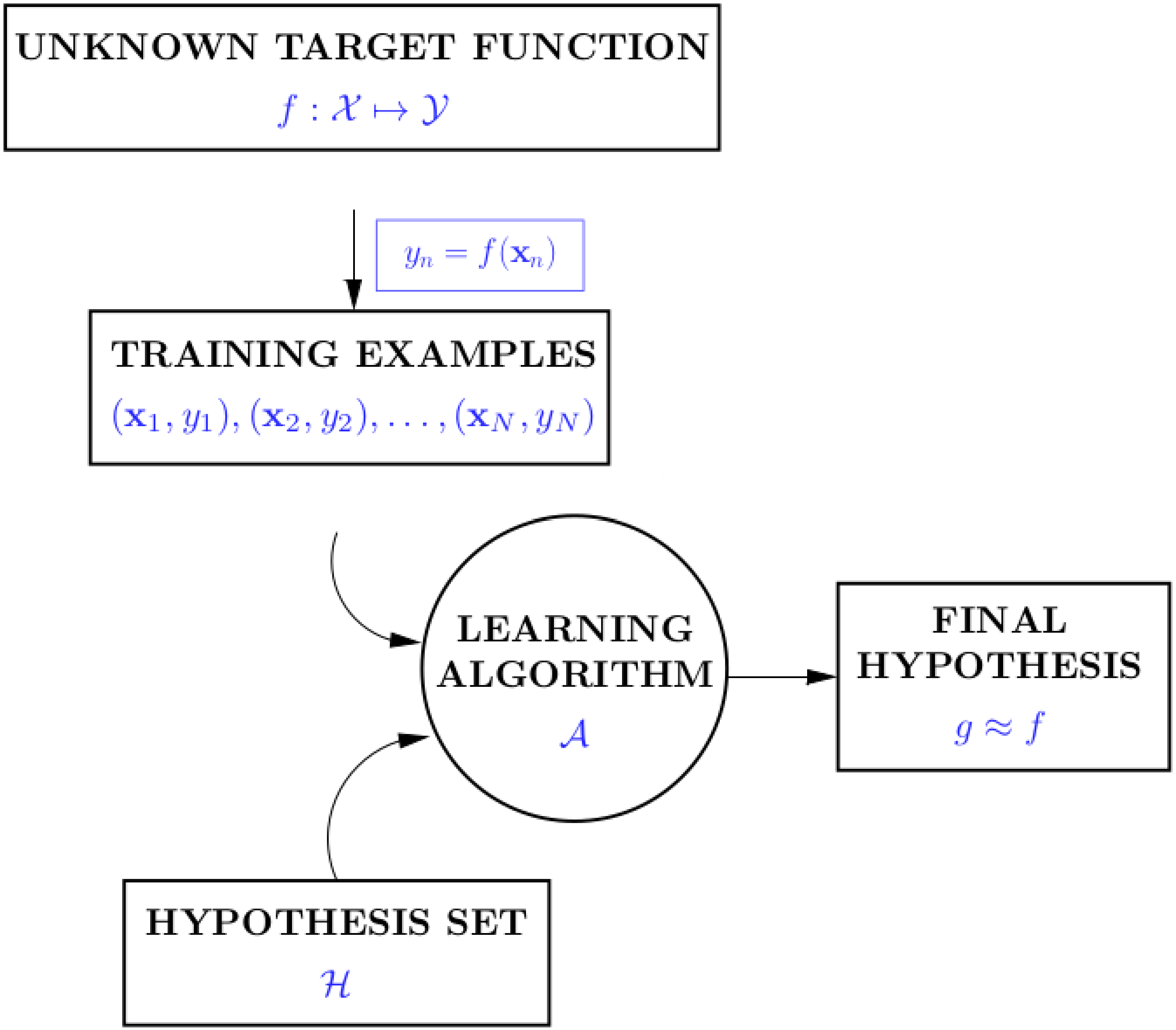
- Comenzamos con un conjunto de hipótesis candidatas $\mathcal{H}$ que son factibles de representar f.

$$\mathcal{H} = \{h_1, h_2, \ldots, \}$$ ← Conjunto de hipótesis

- Un algoritmo $\mathcal{A}$ selecciona una hipótesis g desde $\mathcal{H}$. El algoritmo opera sobre datos etiquetados disponibles para seleccionar g (datos de entrenamiento).

- Para nuevos datos no etiquetados (i.e., y desconocido), usamos g para inferir y .

Clasificación



- PERCEPTRÓN LINEAL -

Un modelo simple (lineal)

age	32 years
gender	male
salary	40,000
debt	26,000
years in job	1 year
years at home	3 years
...	...

¿Sujeto de crédito?

- Vector de características: $\mathbf{x} = [x_1, \ldots, x_d]^{\mathrm{T}}.$

- Aprenderemos distintos 'pesos' para las variables de entrada:

$$\text{“Credit Score”} = \sum_{i=1}^{d} w_i x_i.$$

- Usaremos un **umbral** para decidir si aprobamos el crédito:

Aprobamos si: $$\sum_{i=1}^{d} w_i x_i > \text{threshold},$$ (score de crédito suficiente)

Rechazamos si: $$\sum_{i=1}^{d} w_i x_i < \text{threshold}.$$ (score de crédito insuficiente)

Variable útil para mejorar el score: $w_i > 0$
Variable útil para disminuir el score: $w_i < 0$

Un modelo simple (lineal)

La hipótesis puede escribirse formalmente:

$$h(\mathbf{x}) = \text{sign}\left(\left(\sum_{i=1}^{d} w_i x_i\right) + w_0\right)$$

Usamos el signo para tomar la decisión

Bias (el umbral)

- El conjunto de hipótesis es:

$$\mathcal{H} = \{h(\mathbf{x}) = \text{sign}(\mathbf{w}^{\mathrm{T}}\mathbf{x})\}$$

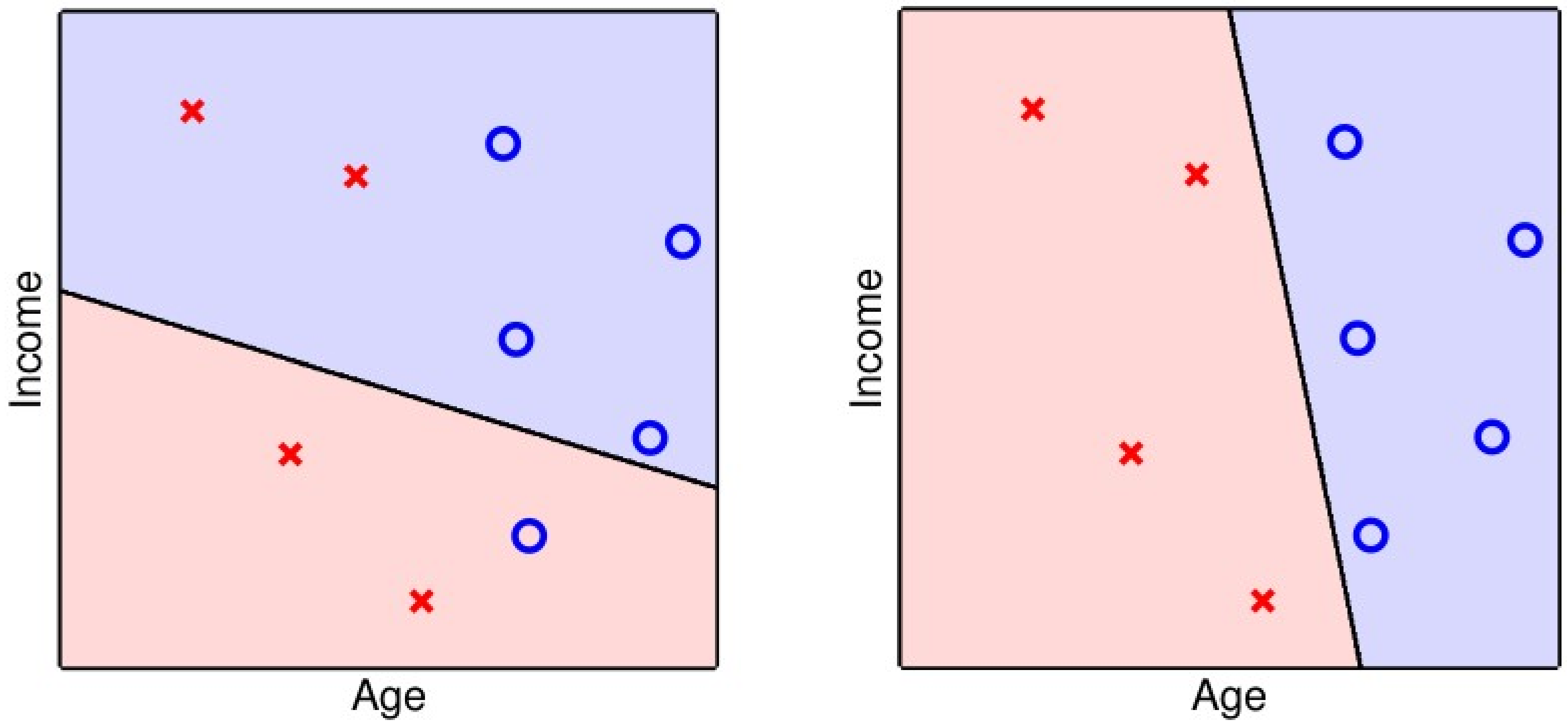
Infinito si los pesos están en $\mathbb{R}$

- Este modelo se llama **Perceptrón**:

$$\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_d \end{bmatrix} \in \mathbb{R}^{d+1}, \qquad \mathbf{x} = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{bmatrix} \in \{1\} \times \mathbb{R}^d.$$

El Perceptrón

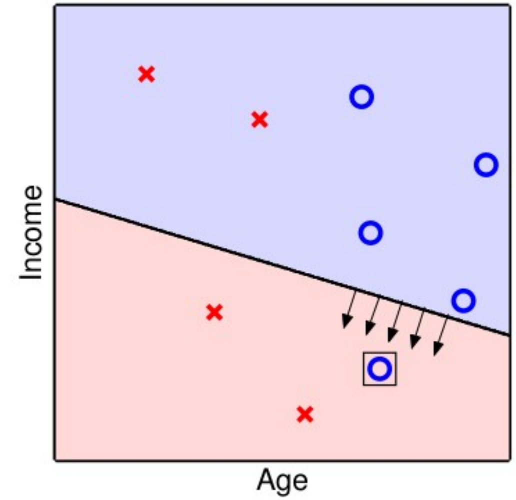
- El Perceptrón usa los datos para encontrar una separación lineal según el atributo de clase y



- Como los datos están en un espacio de representación *d*-dimensional, el separador es un hiperplano.

El Perceptrón

Clasificación binaria: $\mathcal{Y} = \{+1, -1\}$



Enfoque iterativo

$\mathbf{w}(1) = \mathbf{0}$

for iteration $t = 1, 2, 3, \ldots$

the weight vector is $\mathbf{w}(t)$. ← Random

From $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$ pick any misclassified example.

Call the misclassified example $(\mathbf{x}_*, y_*)$,

$$\text{sign}\left(\mathbf{w}(t) \bullet \mathbf{x}_*\right) \neq y_*.$$

Update the weight:

$$\mathbf{w}(t+1) = \mathbf{w}(t) + y_* \mathbf{x}_*.$$

$t \leftarrow t + 1$

Update rule

El Perceptrón

Clasificación binaria: $\mathcal{Y} = \{+1, -1\}$

- Update rule (en ejemplos mal clasificados):

$$\mathbf{w}(t+1) = \mathbf{w}(t) + y(t)\mathbf{x}(t).$$

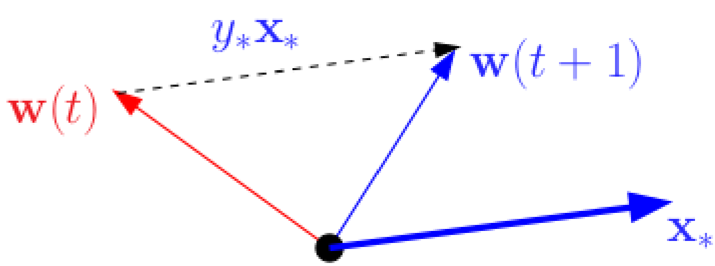
El Perceptrón

Clasificación binaria: $\mathcal{Y} = \{+1, -1\}$

- Update rule (en ejemplos mal clasificados):

$$\mathbf{w}(t+1) = \mathbf{w}(t) + y(t)\mathbf{x}(t).$$

Ground truth → $\boxed{y_* = +1}$



Orienta $\mathbf{w}$ hacia la dirección de $\mathbf{x}$

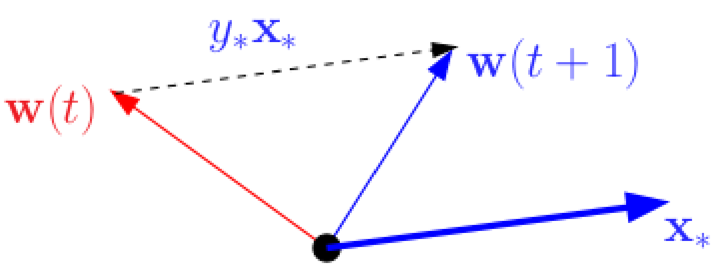
El Perceptrón

Clasificación binaria: $\mathcal{Y} = \{+1, -1\}$

- Update rule (en ejemplos mal clasificados):

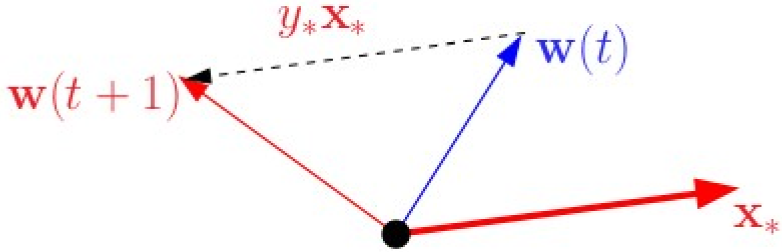
$$\mathbf{w}(t+1) = \mathbf{w}(t) + y(t)\mathbf{x}(t).$$

Ground truth $\boxed{y_* = +1}$



Orienta $\mathbf{w}$ hacia la dirección de $\mathbf{x}$

$\boxed{y_* = -1}$ Ground truth



Orienta $\mathbf{w}$ en la dirección contraria de $\mathbf{x}$

- REGRESIÓN LOGÍSTICA -

Modelos lineales

entrada

$$\text{señal} \longrightarrow S = \mathbf{w}^{\mathrm{T}}\mathbf{x}$$

parámetros

Modelos lineales

activación

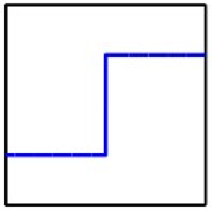
$$y = \theta(s)$$

entrada

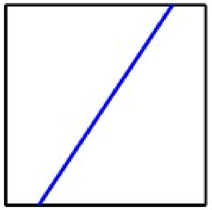
$$s = \mathbf{w}^{\mathrm{T}}\mathbf{x}$$

parámetros

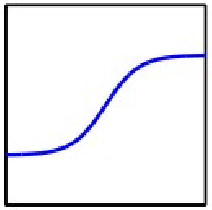
$s = \mathbf{w}^{\mathrm{T}}\mathbf{x} \longrightarrow$



$\rightarrow \mathrm{sign}(\mathbf{w}^{\mathrm{T}}\mathbf{x})$ $\{-1, +1\}$



$\rightarrow \mathbf{w}^{\mathrm{T}}\mathbf{x}$ $\mathbb{R}$



$\rightarrow \theta(\mathbf{w}^{\mathrm{T}}\mathbf{x})$ $[0, 1]$

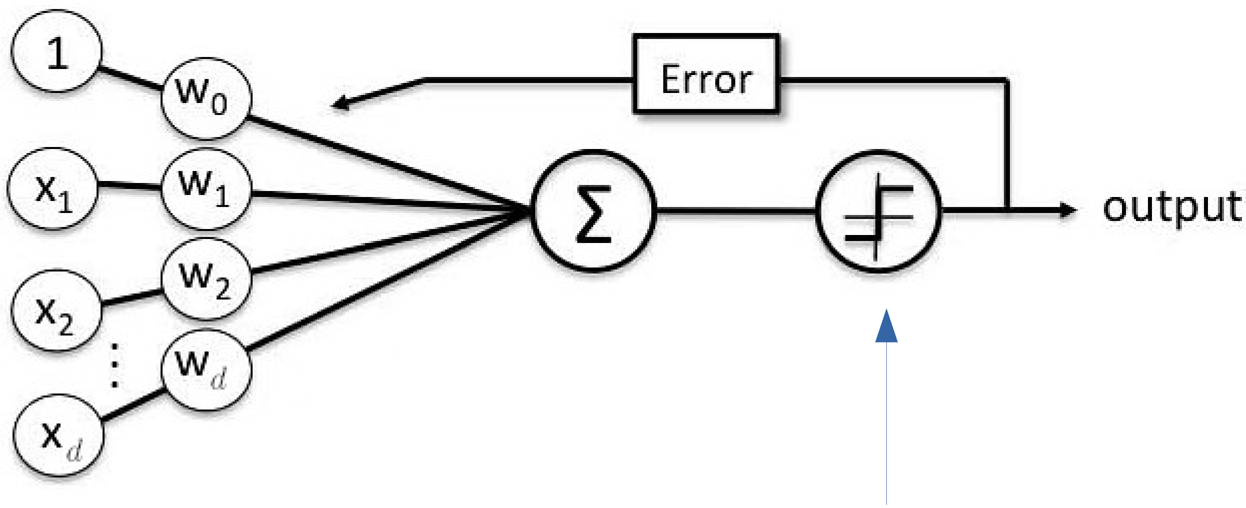
entrada

$$s = \mathbf{w}^{\mathrm{T}}\mathbf{x}$$

parámetros

Perceptrón lineal

$$\mathcal{H}_{\text{lin}} = \{h(\mathbf{x}) = \text{sign}(\mathbf{w}^{\mathrm{T}}\mathbf{x})\}$$



Función signo

Regresión logística

Clase objetivo

Objetivo:

$$f(\mathbf{x}) = \mathbb{P}[y = +1 \mid \mathbf{x}].$$

Regresión logística

Clase objetivo

Objetivo:

$$f(\mathbf{x}) = \mathbb{P}[y = +1 \mid \mathbf{x}].$$

Modelo:

$$h(\mathbf{x}) = \theta\left(\sum_{i=0}^{d} w_i x_i\right) = \theta(\mathbf{w}^{\mathrm{T}}\mathbf{x})$$

Regresión logística

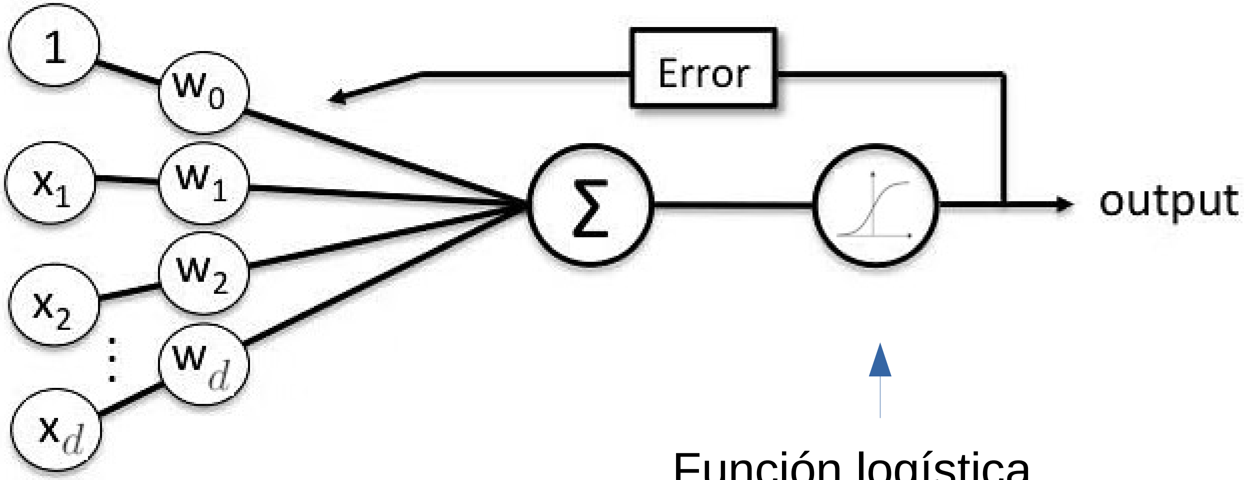
Clase objetivo

Objetivo:

$$f(\mathbf{x}) = \mathbb{P}[y = +1 \mid \mathbf{x}].$$

Modelo:

$$h(\mathbf{x}) = \theta\left(\sum_{i=0}^{d} w_i x_i\right) = \theta(\mathbf{w}^{\mathrm{T}}\mathbf{x})$$



Función logística

Regresión logística

Clase objetivo

Objetivo:

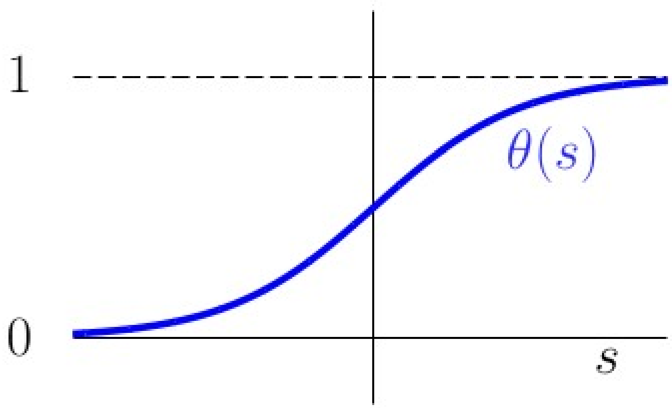
$$f(\mathbf{x}) = \mathbb{P}[y = +1 \mid \mathbf{x}].$$

Modelo:

$$h(\mathbf{x}) = \theta\left(\sum_{i=0}^{d} w_i x_i\right) = \theta(\mathbf{w}^{\mathrm{T}}\mathbf{x})$$

Función logística:

$$y \in [0, 1]$$



$$\theta(s) = \frac{e^s}{1+e^s} = \frac{1}{1+e^{-s}}.$$

$$\theta(-s) = \frac{e^{-s}}{1+e^{-s}} = \frac{1}{1+e^{s}} = 1 - \theta(s).$$

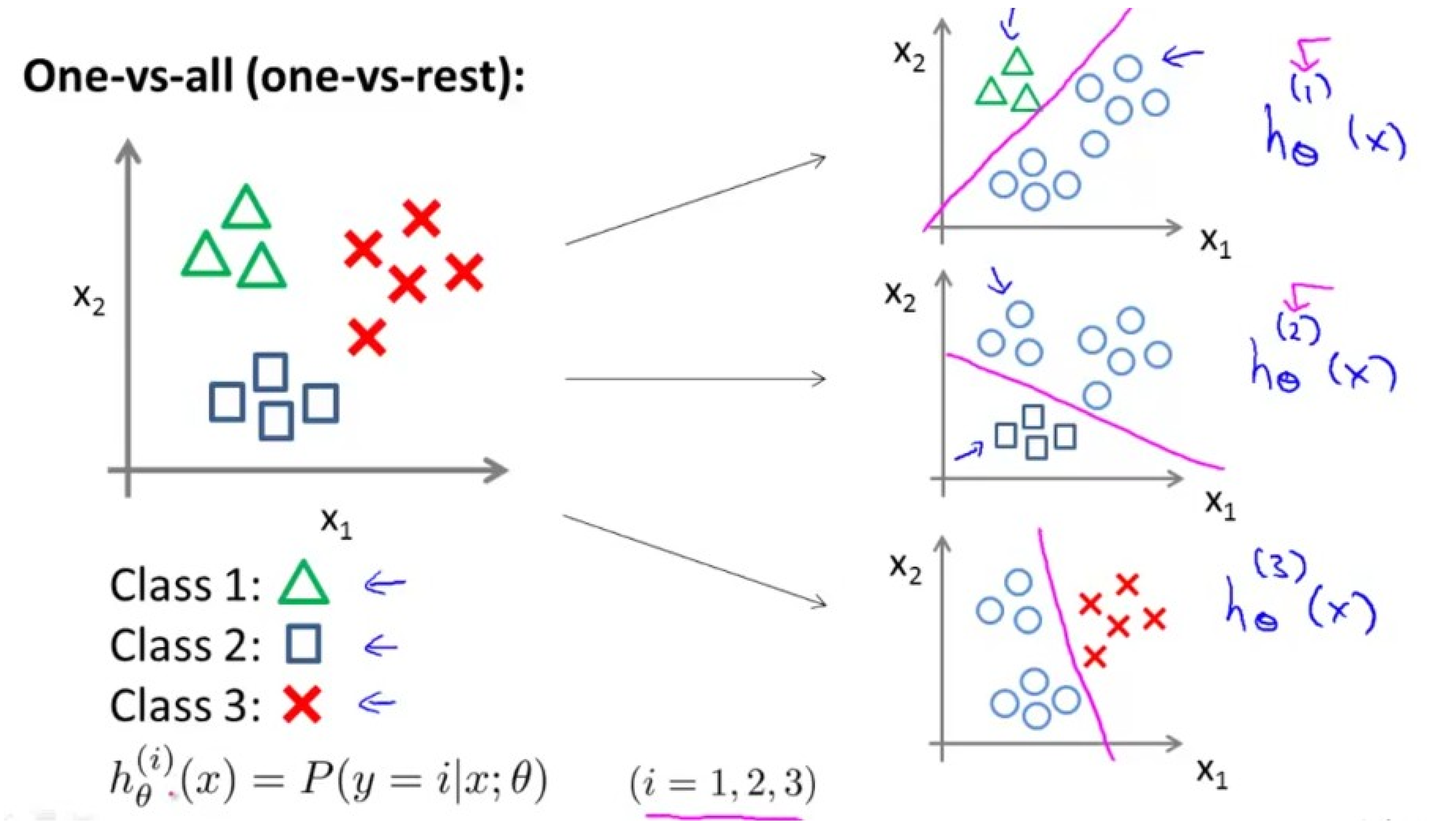
Regresión logística

Notar que: $\mathcal{D} = (\mathbf{x}_1, y_1 = \pm 1), \cdots, (\mathbf{x}_N, y_N = \pm 1)$

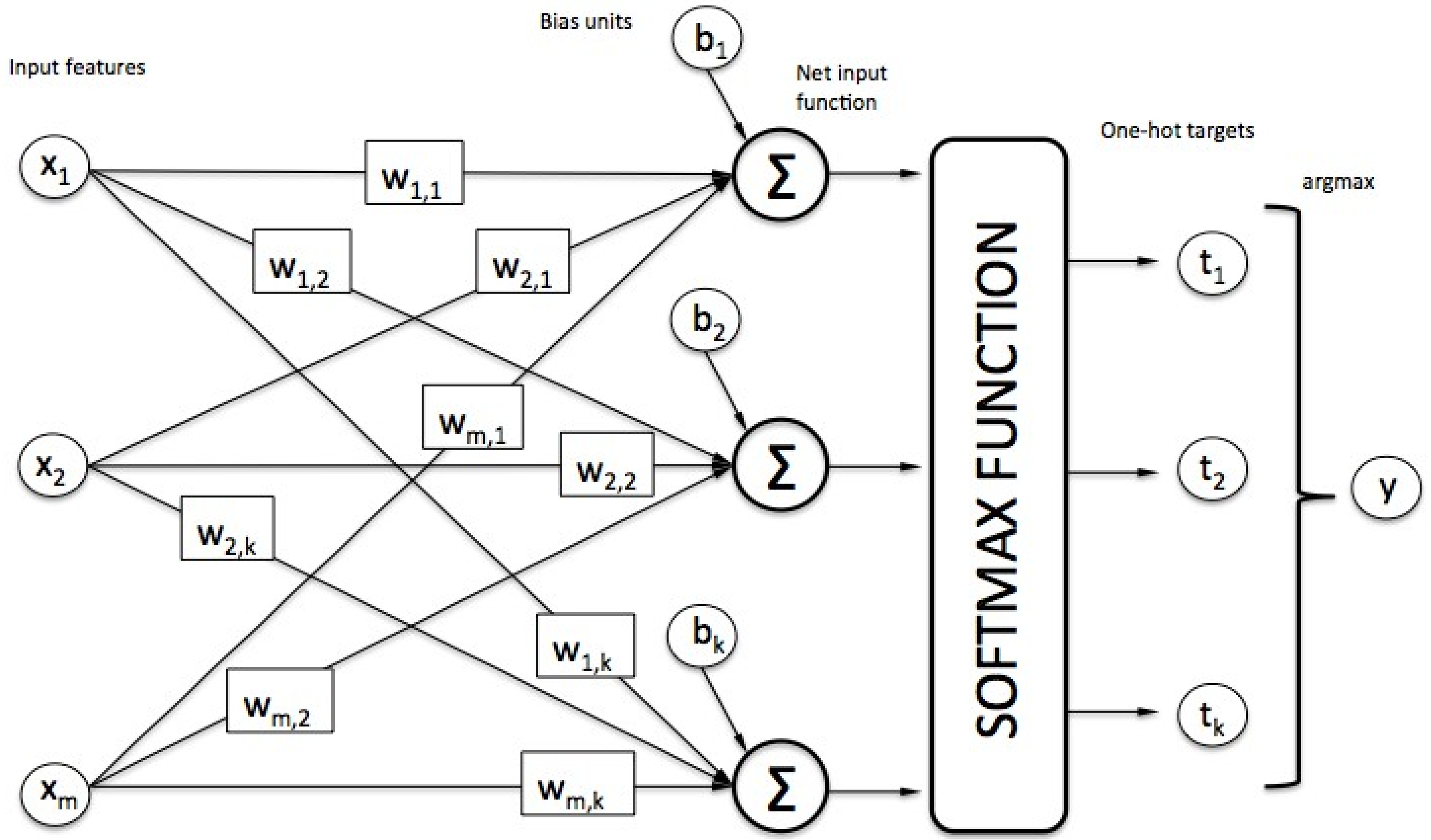
Un buen modelo logra lo siguiente:

$$\begin{cases} h(\mathbf{x}_n) \approx 1 & \text{si} \quad y_n = +1; \\ h(\mathbf{x}_n) \approx 0 & \text{si} \quad y_n = -1. \end{cases}$$

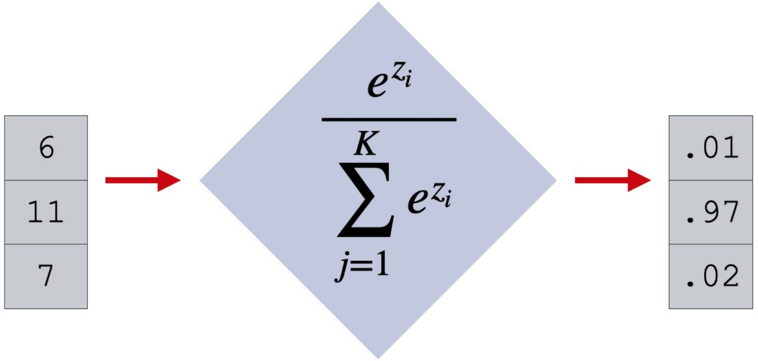
Clasificación multiclase



Clasificación multiclase



$$P(y = j \mid z^{(i)}) = \phi_{softmax}(z^{(i)})$$



- MÉTRICAS DE RENDIMIENTO -

Métricas

+1 -1

+1

-1

Métricas

verdaderos positivos (TP)	falsos positivos (FP)
falsos negativos (FN)	verdaderos negativos (TN)

$$\text{accuracy} = (TP + TN)/(TP + FP + FN + TN)$$

$$\begin{aligned} P &= TP/(TP + FP) \\ R &= TP/(TP + FN) \end{aligned}$$

$$F = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}$$

F-measure balanceada con $\beta = 1$

- GRADIENTE DESCENDENTE -

Regresión logística

Si el ajuste es adecuado:

$$P(y \mid \mathbf{x}) = \theta(y \cdot \mathbf{w}^{\mathrm{T}}\mathbf{x})$$

Luego, podemos expresar la función de verosimilitud:

$$P(y_1, \ldots, y_N \mid \mathbf{x}_1, \ldots, \mathbf{x}_n) = \prod_{n=1}^{N} P(y_n \mid \mathbf{x}_n).$$

Regresión logística

Maximizamos la función de verosimilitud:

$$
\begin{aligned}
& \max \quad \prod_{n=1}^{N} P(y_n \mid \mathbf{x}_n) \\
\Leftrightarrow \; & \max \quad \ln\left(\prod_{n=1}^{N} P(y_n \mid \mathbf{x}_n)\right) \\
\equiv \; & \max \quad \sum_{n=1}^{N} \ln P(y_n \mid \mathbf{x}_n)
\end{aligned}
$$

Regresión logística

Maximizamos la función de verosimilitud:

$$
\begin{aligned}
& \max && \prod_{n=1}^{N} P(y_n \mid \mathbf{x}_n) \\
\Leftrightarrow\ & \max && \ln\left(\prod_{n=1}^{N} P(y_n \mid \mathbf{x}_n)\right) \\
\equiv\ & \max && \sum_{n=1}^{N} \ln P(y_n \mid \mathbf{x}_n) \\
\Leftrightarrow\ & \min && -\frac{1}{N}\sum_{n=1}^{N} \ln P(y_n \mid \mathbf{x}_n) \\
\equiv\ & \min && \frac{1}{N}\sum_{n=1}^{N} \ln \frac{1}{P(y_n|\mathbf{x}_n)} \\
\equiv\ & \min && \frac{1}{N}\sum_{n=1}^{N} \ln \frac{1}{\theta(y_n \cdot \mathbf{w}^{\mathrm{T}}\mathbf{x}_n)}
\end{aligned}
$$

Regresión logística

Maximizamos la función de verosimilitud:

$$
\begin{aligned}
& \max && \prod_{n=1}^{N} P(y_n \mid \mathbf{x}_n) \\
\Leftrightarrow\ & \max && \ln\left(\prod_{n=1}^{N} P(y_n \mid \mathbf{x}_n)\right) \\
\equiv\ & \max && \sum_{n=1}^{N} \ln P(y_n \mid \mathbf{x}_n) \\
\Leftrightarrow\ & \min && -\frac{1}{N}\sum_{n=1}^{N} \ln P(y_n \mid \mathbf{x}_n) \\
\equiv\ & \min && \frac{1}{N}\sum_{n=1}^{N} \ln \frac{1}{P(y_n \mid \mathbf{x}_n)} \\
\equiv\ & \min && \frac{1}{N}\sum_{n=1}^{N} \ln \frac{1}{\theta(y_n \cdot \mathbf{w}^{\mathrm{T}}\mathbf{x}_n)} \\
\equiv\ & \min && \frac{1}{N}\sum_{n=1}^{N} \ln\left(1 + e^{-y_n \cdot \mathbf{w}^{\mathrm{T}}\mathbf{x}_n}\right)
\end{aligned}
$$

$$\theta(s) = \frac{1}{1+e^{-s}}.$$

Regresión logística

Tenemos una expresión para:

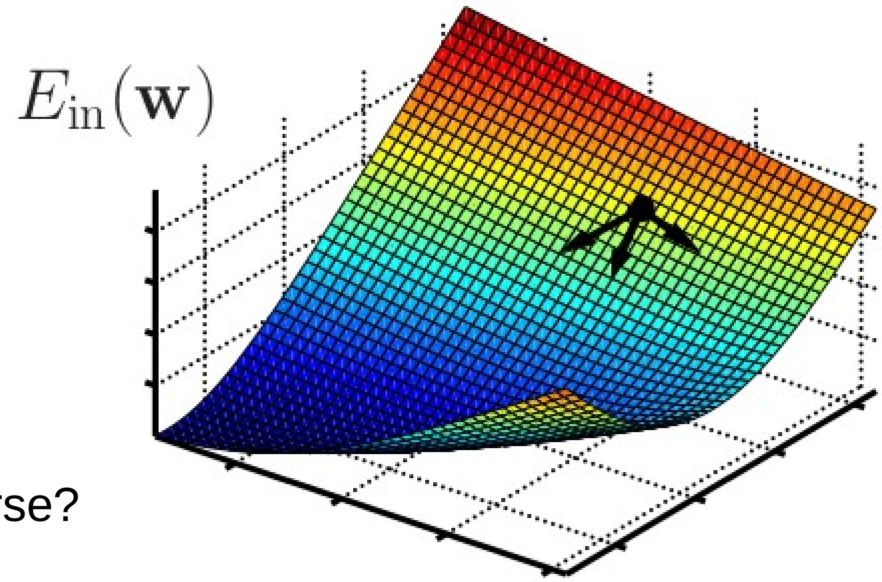
Parámetros del modelo

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} \ln(1 + e^{-y_n \cdot \mathbf{w}^{\mathrm{T}}\mathbf{x}_n})$$

Cross-entropy

La función es convexa, por lo que podemos optimizarla de forma iterativa:

<u>Idea del gradiente descendente</u>:



$$\mathbf{w}(t+1) = \mathbf{w}(t) + \eta\hat{\mathbf{v}}$$

¿En cuál dirección conviene moverse?

Regresión logística

Tenemos una expresión para:

Parámetros del modelo

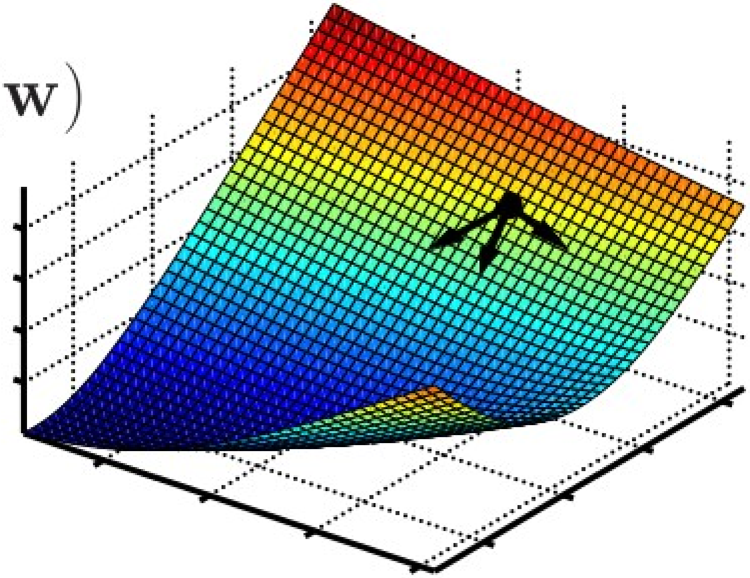
$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} \ln(1 + e^{-y_n \cdot \mathbf{w}^{\mathrm{T}}\mathbf{x}_n})$$

Cross-entropy

La función es convexa, por lo que podemos optimizarla de forma iterativa:

Idea del gradiente descendente:

$E_{\text{in}}(\mathbf{w})$



$$\mathbf{w}(t+1) = \mathbf{w}(t) + \eta\hat{\mathbf{v}}$$

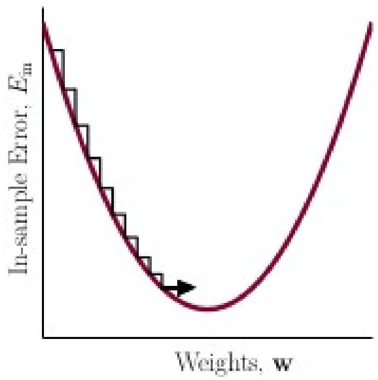
¿En cuál dirección conviene moverse?

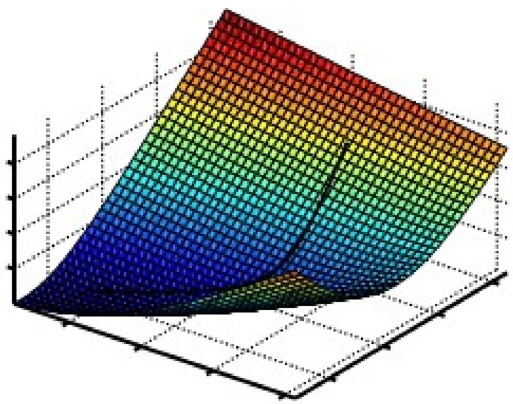
Move in the direction $\mathbf{v}_t = -\mathbf{g}_t$.

Gradiente descendente

El efecto del **learning rate**:

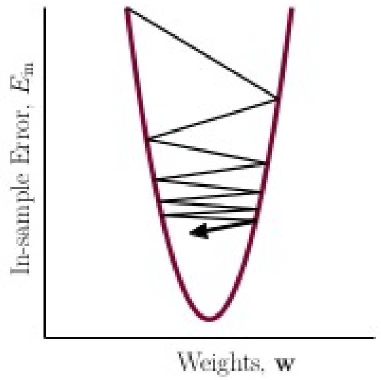
η muy pequeño

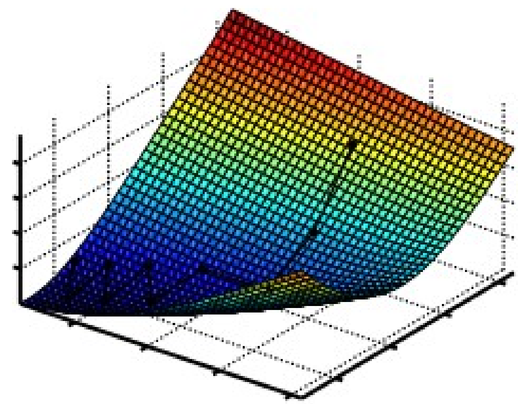




$\eta = 0.1$; 75 steps

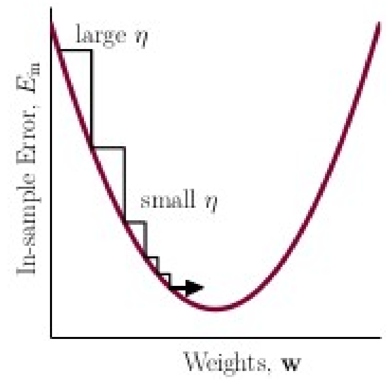
η muy grande



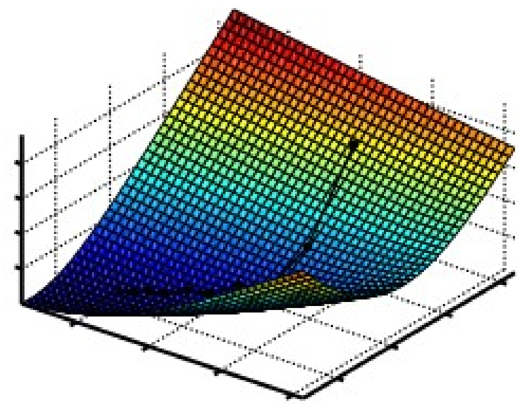


$\eta = 2$; 10 steps

η_t



Adam



variable η_t; 10 steps

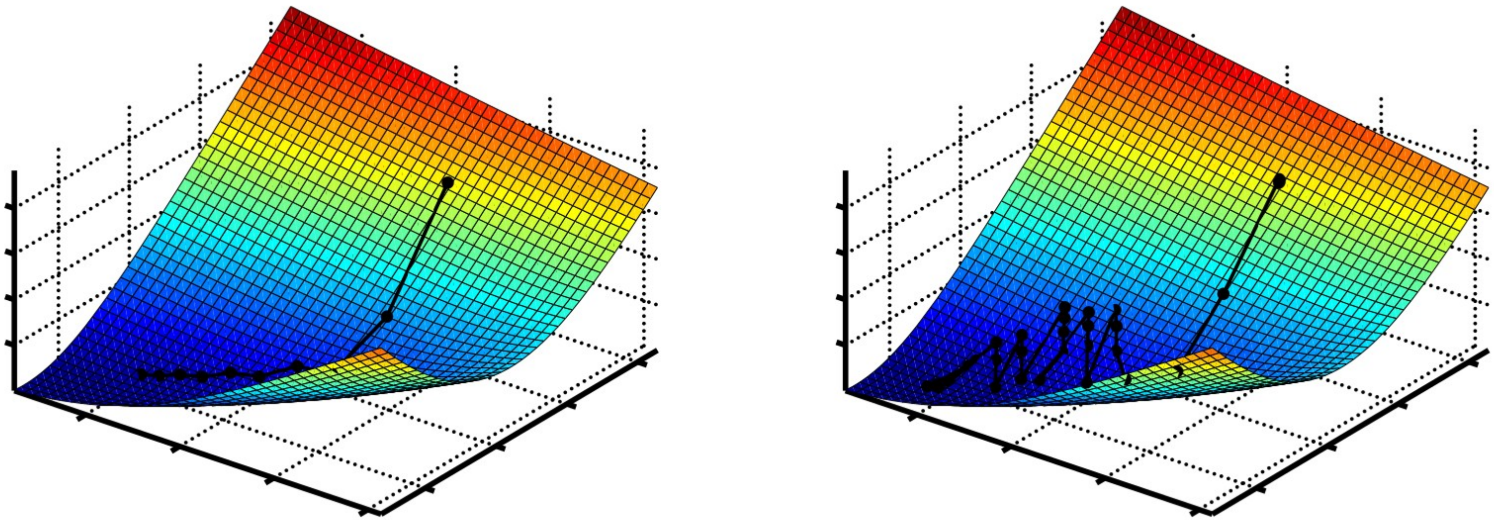
Gradiente descendente

Una variación de gradiente descendente considera la evaluación del gradiente en una muestra del training set.

$$\mathbf{w}(t+1) \leftarrow \mathbf{w}(t) - \eta \nabla_{\mathbf{w}} e(\mathbf{w}, \mathbf{x}_*, y_*)$$

Dado que la muestra se toma al azar, se le denomina gradiente descendente estocástico.

GD SGD



$\eta = 6$
10 steps
$N = 10$

$\eta = 2$
30 steps

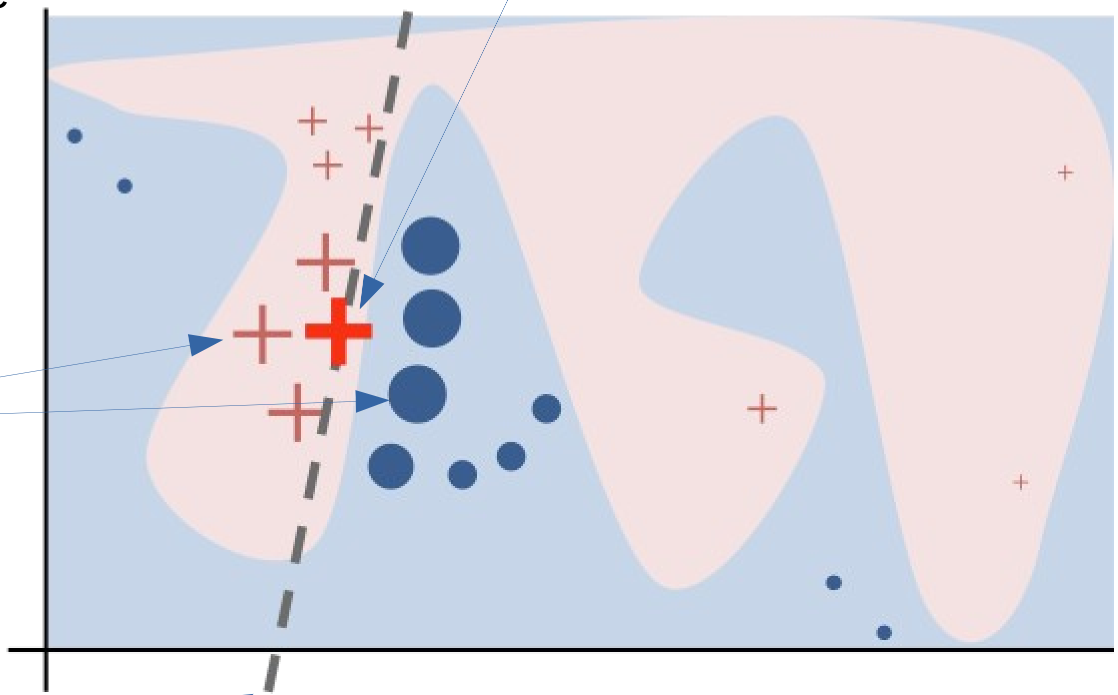
Puede ayudar a escapar de óptimos locales

Explainers

LIME (Local Interpretable Model-Agnostic Explanations)

Instancia a explicar

1. Se perturban (eliminan features) de la instancia

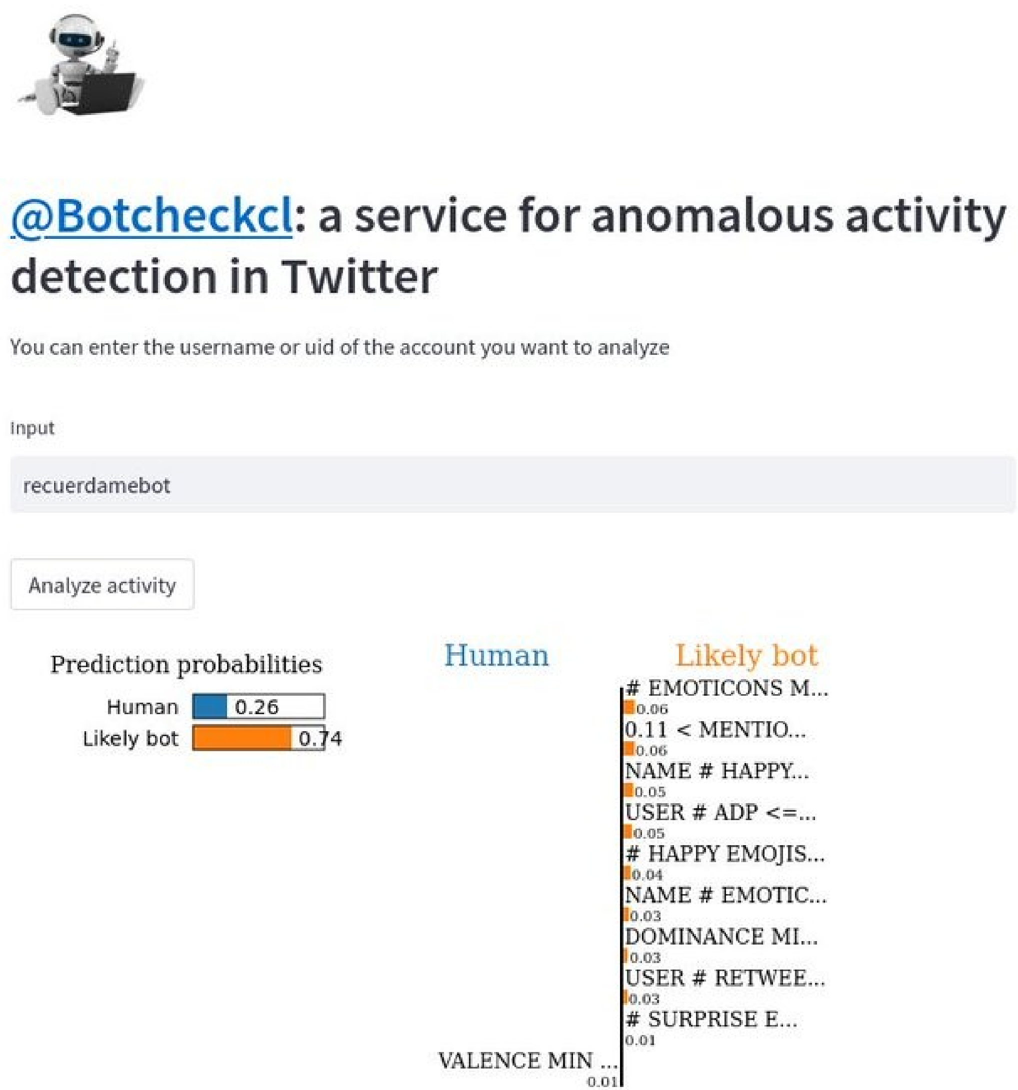


2. Se obtienen las predicciones sobre los ejemplos perturbados

3. Se construye un modelo lineal que aproxima al modelo original usando las predicciones (noisy label)

4. Se identifica cuales perturbaciones inclinan la decisión

LIME (Local Interpretable Model-Agnostic Explanations)



- ANEXO: GRADIENTE DESCENDENTE -

Gradiente descendente

$$\begin{aligned}\Delta E_{\text{in}} &= E_{\text{in}}(\mathbf{w}(t+1)) - E_{\text{in}}(\mathbf{w}(t)) \\ &= E_{\text{in}}(\mathbf{w}(t) + \eta\hat{\mathbf{v}}) - E_{\text{in}}(\mathbf{w}(t)) \\ &= \eta\, \nabla E_{\text{in}}(\mathbf{w}(t))^{\mathrm{T}}\hat{\mathbf{v}} + O(\eta^2)\end{aligned}$$

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{f''(x_0)}{2!}(x - x_0)^2 + \cdots + \frac{f^{(k+1)}(c)}{(k+1)!}(x - x_0)^{k+1}.$$

$$\Delta E_{\text{in}} = E_{\text{in}}(\mathbf{w}(t+1)) - E_{\text{in}}(\mathbf{w}(t))$$

$$\underbrace{f(x_0 + \eta \cdot x)}_{f(x_0) + \nabla f(x_0) \cdot (x_0 + \eta \cdot x - x_0) + \ldots} - f(x_0)$$

(expansión de Taylor de 1er orden)

Gradiente descendente

$$\begin{aligned}\Delta E_{\text{in}} &= E_{\text{in}}(\mathbf{w}(t+1)) - E_{\text{in}}(\mathbf{w}(t)) \\ &= E_{\text{in}}(\mathbf{w}(t) + \eta\hat{\mathbf{v}}) - E_{\text{in}}(\mathbf{w}(t)) \\ &= \eta\, \nabla E_{\text{in}}(\mathbf{w}(t))^{\text{T}}\hat{\mathbf{v}} + O(\eta^2)\end{aligned}$$

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{f''(x_0)}{2!}(x - x_0)^2 + \cdots + \frac{f^{(k+1)}(c)}{(k+1)!}(x - x_0)^{k+1}.$$

$$\Delta E_{\text{in}} = E_{\text{in}}(\mathbf{w}(t+1)) - E_{\text{in}}(\mathbf{w}(t))$$

$$\underbrace{f(x_0 + \eta \cdot x)} - f(\cancel{x}_0)$$

(expansión de Taylor de 1er orden)

$$f(\cancel{x}_0) + \nabla f(x_0) \cdot (\cancel{x}_0 + \eta \cdot x - \cancel{x}_0) + \ldots$$

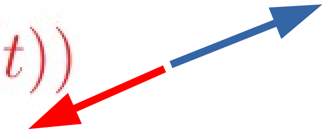
Gradiente descendente

$$= \ \eta \underbrace{\nabla E_{\text{in}}(\mathbf{w}(t))^{\mathrm{T}} \hat{\mathbf{v}}} + O(\eta^2)$$

Minimizado en $\hat{\mathbf{v}} = -\dfrac{\nabla E_{\text{in}}(\mathbf{w}(t))}{\| \nabla E_{\text{in}}(\mathbf{w}(t)) \|}$

$-\nabla E_{\text{in}}(\mathbf{w}(t))$

$\nabla E_{\text{in}}(\mathbf{w}(t))$



Gradiente descendente

$$-\nabla E_{\text{in}}(\mathbf{w}(t))$$

$$= \underbrace{\eta \nabla E_{\text{in}}(\mathbf{w}(t))^{\mathrm{T}} \hat{\mathbf{v}}}_{} + O(\eta^2) \qquad \nabla E_{\text{in}}(\mathbf{w}(t))$$

Minimizado en $\hat{\mathbf{v}} = -\dfrac{\nabla E_{\text{in}}(\mathbf{w}(t))}{\| \nabla E_{\text{in}}(\mathbf{w}(t)) \|}$

Gradiente descendente

Initialize at step $t = 0$ to $\mathbf{w}(0)$.

for $t = 0, 1, 2, \ldots$ **do**

Compute the gradient

$$\mathbf{g}_t = \nabla E_{\text{in}}(\mathbf{w}(t)).$$

Move in the direction $\mathbf{v}_t = -\mathbf{g}_t$.

Update the weights:

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \eta \mathbf{v}_t.$$

Iterate 'until it is time to stop'.

end for

Return the final weights.