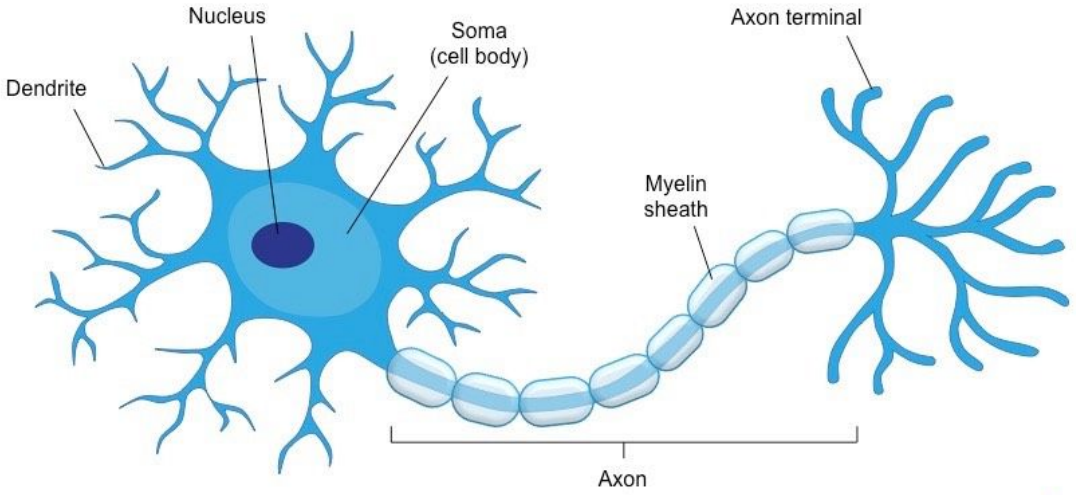
# Redes Neuronales para Lenguaje Natural

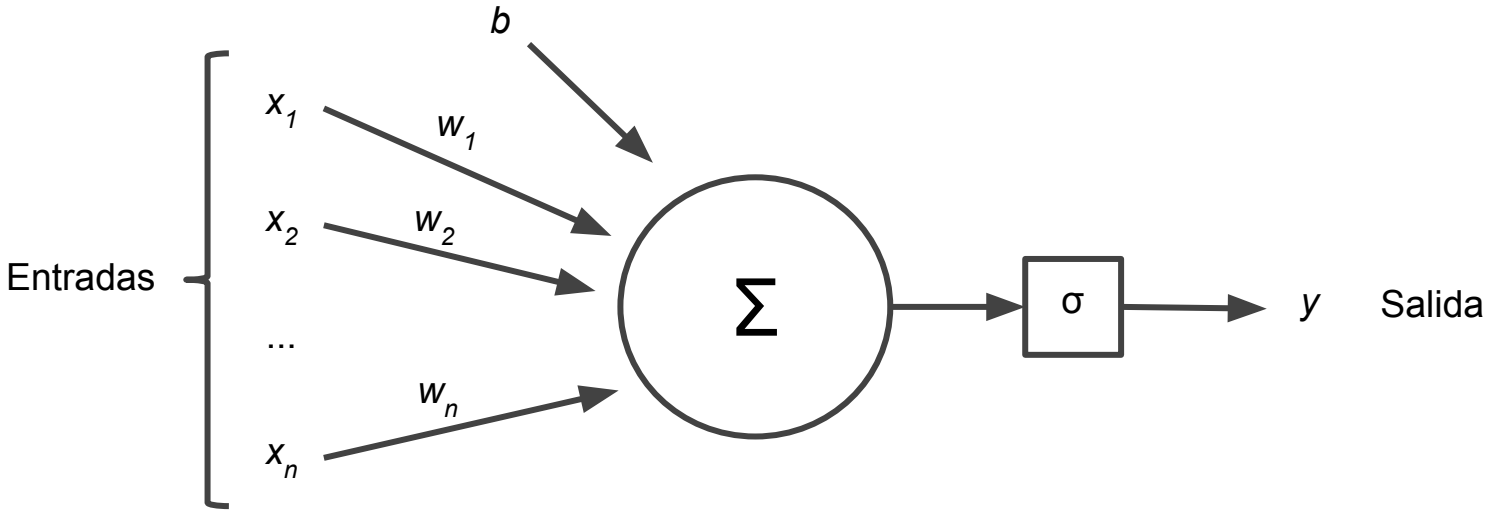
2024

Grupo de Procesamiento de Lenguaje Natural
Instituto de Computación

# Neurona


Nucleus
Soma
(cell body)
Axon terminal
Dendrite
Myelin
sheath
Axon

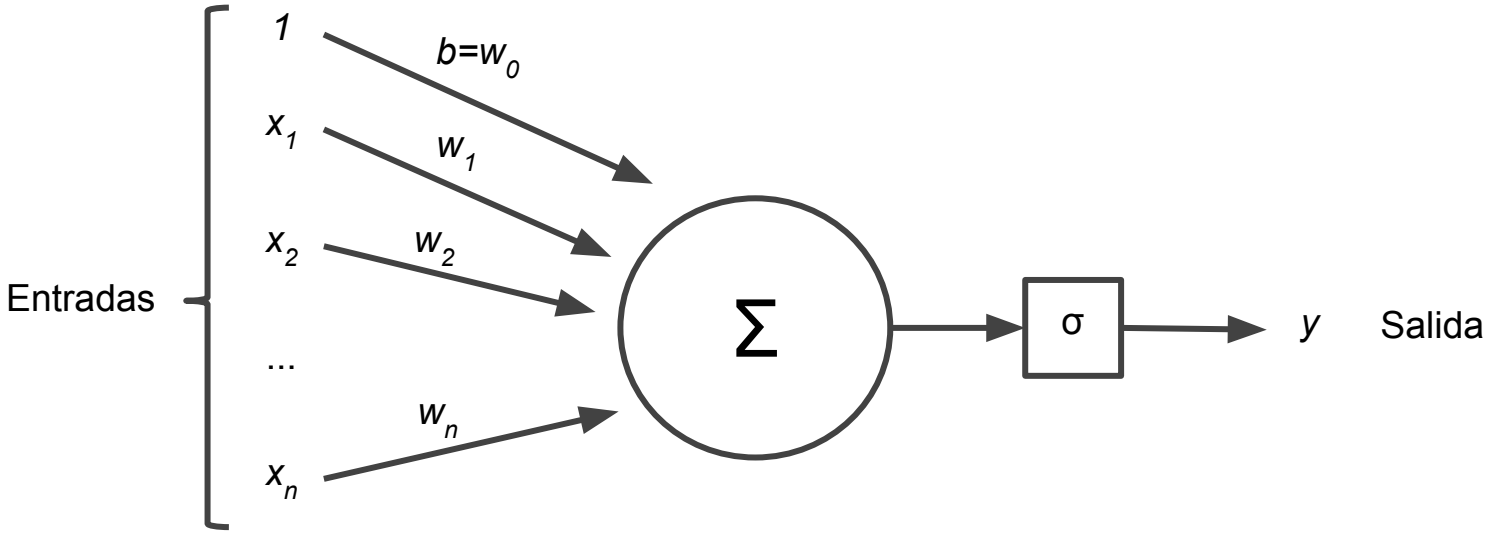
# Neurona Artificial




$$y = \sigma(\sum_i x_i w_i + b)$$

*Neurona de McCulloch-Pitts, 1943*

# Neurona Artificial

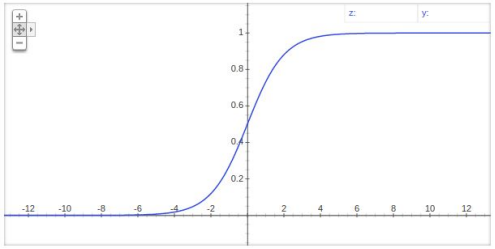



$$\hat{x} = [1, x_1, x_2, ..., x_n]$$
$$\hat{w} = [w_0, w_1, w_2, ..., w_n]$$

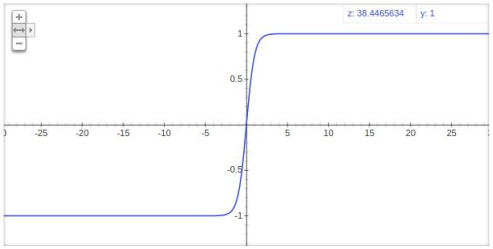
$$y = \sigma(\hat{x}.\hat{w})$$

# Función de Activación

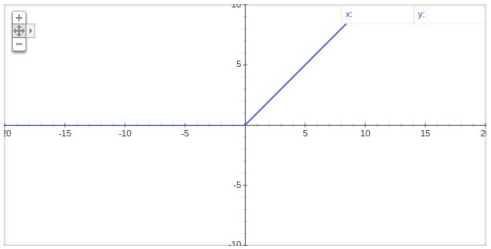
- Función sigmoide o logística: $\sigma(z) = \frac{1}{1+e^{-z}}$



- Tangente hiperbólica: $\tanh(z) = \frac{e^{z}-e^{-z}}{e^{z}+e^{-z}}$



- ReLU: $\mathrm{relu}(z) = max(0, z)$



- Otras...

# Ejemplo: Análisis de Sentimiento

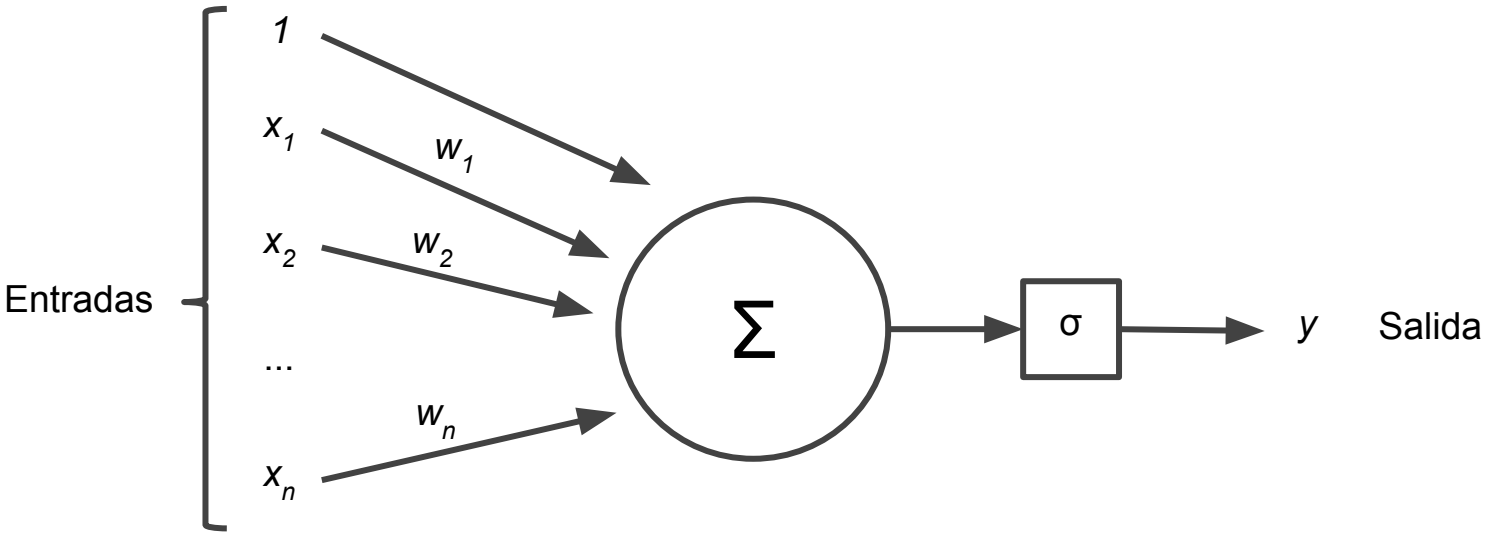
Sea un conjunto de comentarios (reviews) de películas, cada uno categorizado como positivo o negativo (0 o 1)

| **Review** | **Clasificación** |
|---|---|
| Buenísima. Entretenida y bien lograda. | 1 |
| Escenas traídas de los pelos. No la recomiendo. | 0 |
| No me gustó la película. | 0 |
| Horrible! Me aburrí como un hongo. | 0 |
| Muy buena la película. La super recomiendo. | 1 |
| Muy linda película. | 1 |
| Me gustó! La recomiendo totalmente. | 1 |
| No la recomiendo. Es un divague. | 0 |
| Una historia que es un mamarracho. | 0 |

# Ejemplo: Análisis de Sentimiento

Las entradas de mi red son números, y la salida también

¿Cómo hago para representar mi texto mediante números?




$$\hat{x} = [1, x_1, x_2, ..., x_n]$$
$$\hat{w} = [w_0, w_1, w_2, ..., w_n]$$

$$y = \sigma(\hat{x}.\hat{w})$$

# Ejemplo: Análisis de Sentimiento

Las entradas de mi red son números, y la salida también

¿Cómo hago para representar mi texto mediante números?

Alternativa 1: Extracción de features (manuales)

Lista de palabras positivas P={buenísima, buena, gustó, linda, recomiendo}

Lista de palabras negativas N={horrible, divague, mamarracho}

$x_1$ = cantidad de palabras de la lista P en el texto

$x_2$ = cantidad de palabras de la lista N en el texto

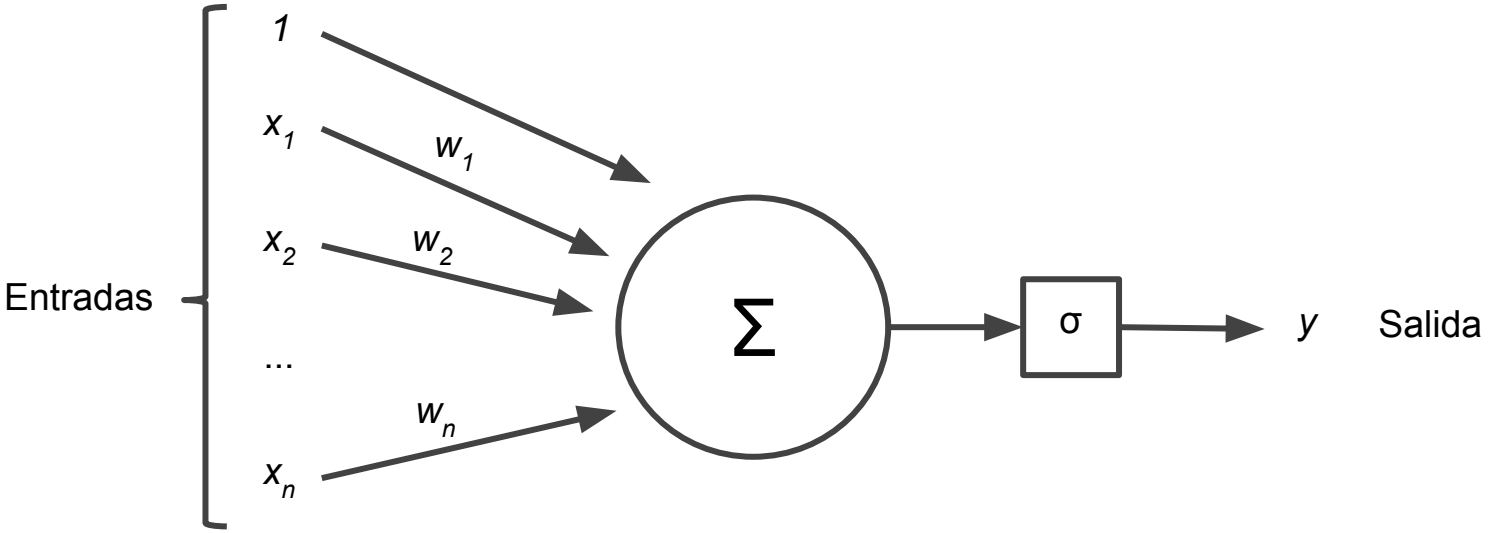
$x_3$ = la palabra “no” aparece en el texto

$x_4$ = el signo de exclamación “!” aparece en el texto

$x_5$ = cantidad de palabras en el texto

…

# Ejemplo: Análisis de Sentimiento




$$\hat{x} = [1, x_1, x_2, ..., x_n]$$
$$\hat{w} = [w_0, w_1, w_2, ..., w_n]$$

$$y = \sigma(\hat{x}.\hat{w})$$

Por ejemplo, eligiendo σ como la sigmoide (o función logística) $\frac{1}{1+e^{-z}}$

tenemos que este clasificador es exactamente una regresión logística

# Ejemplo: Análisis de Sentimiento

*Escenas traídas de los pelos. No la recomiendo.*

P={buenísima, buena, gustó, linda, recomiendo}

N={horrible, divague, mamarracho}

$x_1$ = cantidad de palabras de la lista P en el texto = 1

$x_2$ = cantidad de palabras de la lista N en el texto = 0

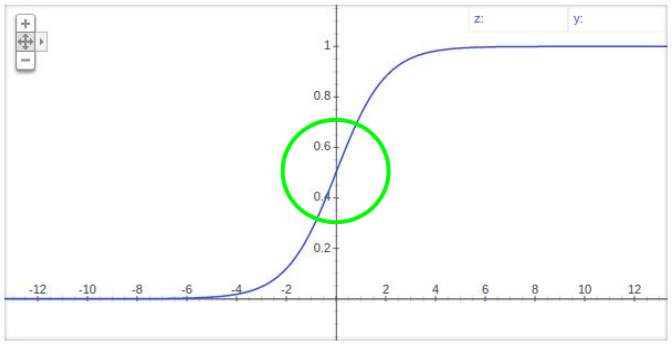
$x_3$ = la palabra “no” aparece en el texto = 1

$x_4$ = el signo de exclamación “!” aparece en el texto = 0

$x_5$ = cantidad de palabras en el texto = 8

*w = [0.5, 0.6, -0.8, 0.3, -0.1]*

$$\frac{1}{1+e^{-x \cdot w}} = 0.2497$$



# Entrenamiento

(por ahora con una sola unidad)

# Entrenamiento

Cómo aprendo los mejores pesos w?

Aprendizaje supervisado: partir el conjunto en train, dev, test

Resumen de la metodología:

- Entreno con train
- Si estoy comparando varios modelos o variaciones de un modelo, uso dev
- Cuando tengo el modelo “final”, evalúo sobre test

Alternativa: usar cross-validation

# Entrenamiento

Cómo aprendo los mejores pesos w?

Conjunto de entrenamiento, ejemplos:

$t^{(1)} \rightarrow y^{(1)}$

$t^{(2)} \rightarrow y^{(2)}$

…

$t^{(m)} \rightarrow y^{(m)}$

Siendo cada $t^{(i)}$ un texto, cada $y^{(i)}$ vale 0 o 1

Pero de cada ejemplo $t^{(i)}$ extraigo el vector de features $x_1^{(i)}$, $x_2^{(i)}$, … $x_k^{(i)} = x^{(i)}$

# Función de pérdida (Loss)

Necesito una forma de relacionar los valores obtenidos con mi red y los valores esperados

Para cierto ejemplo x:

ŷ = σ(x.w)

y es el gold standard

Para estos casos en que y puede tomar los valores 0 o 1, se suele usar la entropía cruzada

$$L_{CE} = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log(\hat{y}^{(i)}) + (1-y^{(i)})\log(1-\hat{y}^{(i)})\right]$$

$$= -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log(\sigma(x^{(i)}\cdot w)) + (1-y^{(i)})\log(1-\sigma(x^{(i)}\cdot w))\right]$$

# Función de pérdida (Loss)

Entropía cruzada / Cross-entropy: $H(p, q) = -\sum_{x \in \mathcal{X}} p(x) \log q(x)$

Mide la “discrepancia” entre dos distribuciones de probabilidad

Cuando  más bajo sea el valor, más similares serán las distribuciones

$\hat{y} = \sigma(x.w)$

$CE(\hat{y},y) = y \cdot \log(\hat{y}) + (1-y) \cdot \log(1 - \hat{y})$

$CE(\hat{y},y) = y \cdot \log(\sigma(x.w)) + (1-y) \cdot \log(1 - \sigma(x.w))$

Maximizar este valor equivale a minimizar el opuesto

$L_{CE}(\hat{y},y) = - [y \cdot \log(\sigma(x.w)) + (1-y) \cdot \log(1 - \sigma(x.w))]$

Ahora tengo una fórmula que puedo minimizar para encontrar el óptimo

# Formalización del problema

Se suele denominar θ al conjunto de parámetros de una familia de funciones de aprendizaje automático (en este caso una red neuronal)

En el ejemplo, nuestro θ es w

Entonces la función a aprender es $\hat{y} = f(x;\theta)$

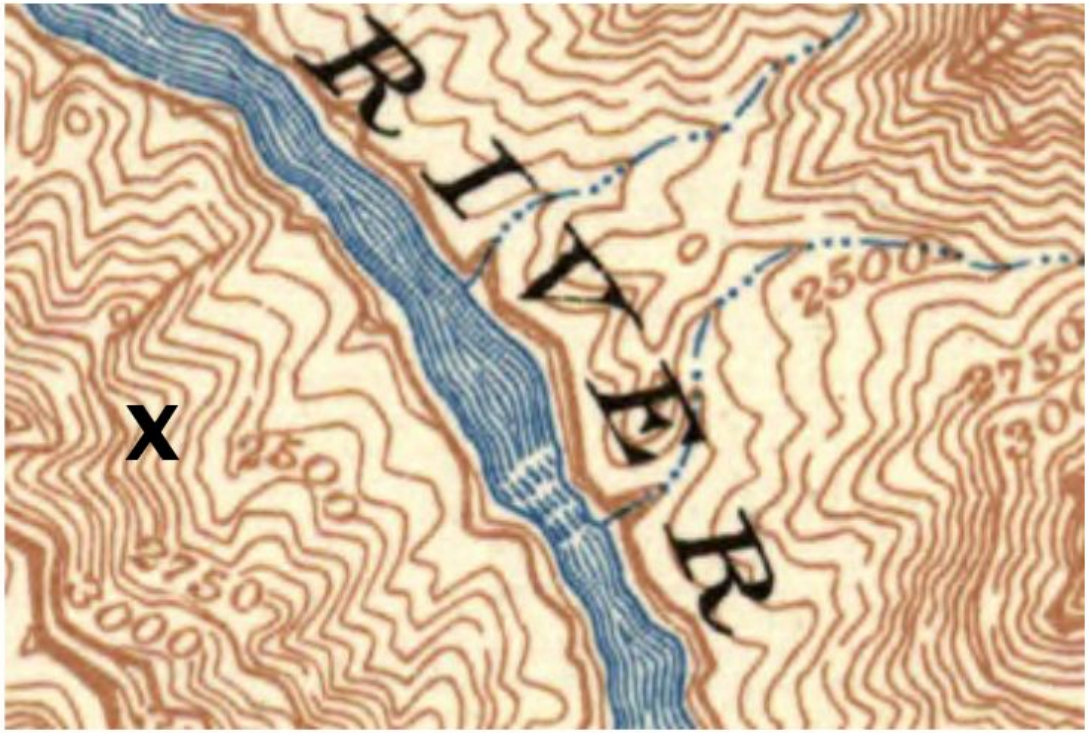
Queremos encontrar el conjunto de parámetros que haga mínimo lo siguiente

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \frac{1}{m} \sum_{i=1}^{m} L_{\mathrm{CE}}(f(x^{(i)};\theta), y^{(i)})$$

Notar que tenemos m ejemplos, lo que hacemos es sumar la función de loss en todos los ejemplos.

# Descenso por gradiente

Cómo llegar al fondo del cañón del río?




Mirar 360º alrededor de nuestra posición

Encontrar la dirección en que la pendiente es más pronunciada hacia abajo

Caminar en esa dirección

# Descenso por gradiente

Objetivo: minimizar la función de *loss*

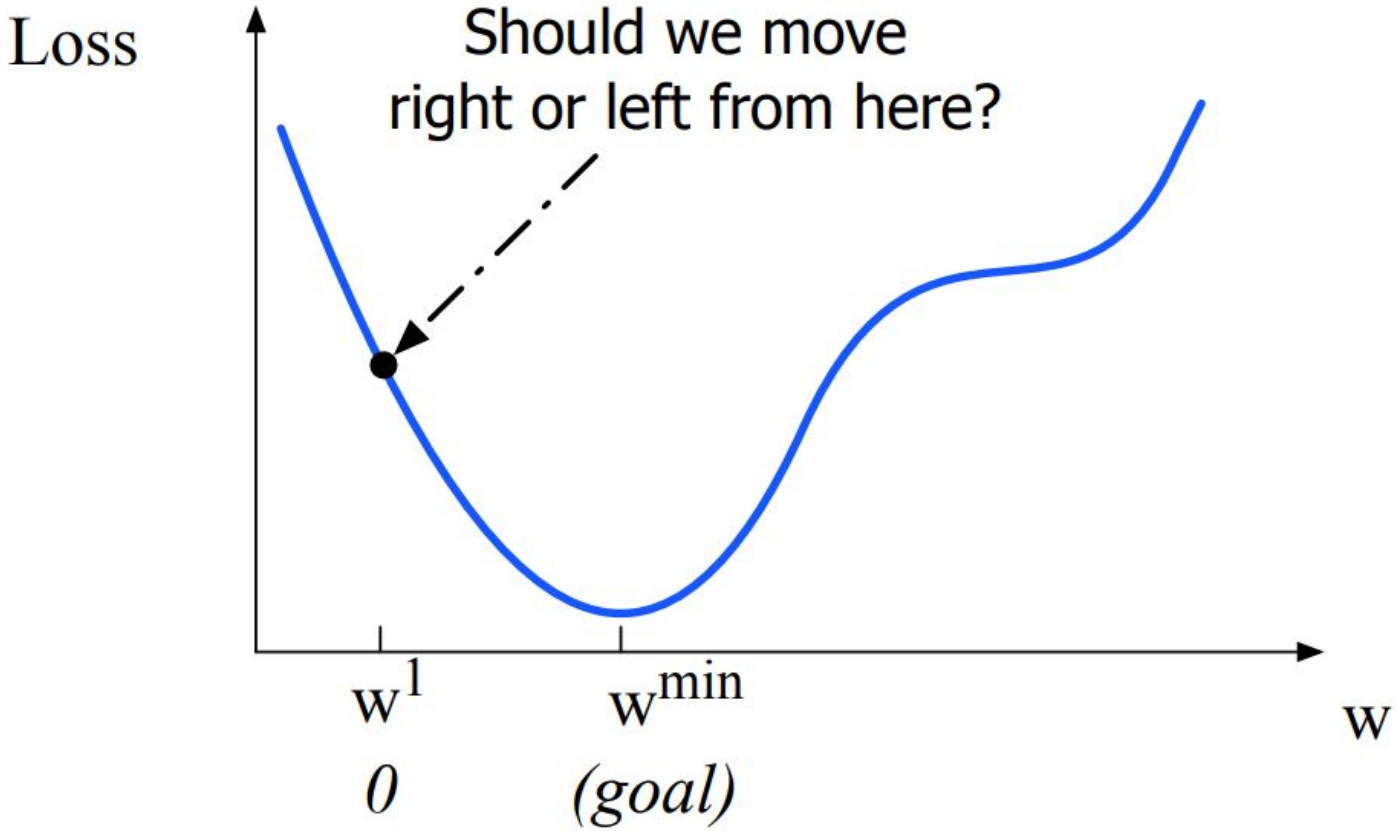
Notar que en el caso de que σ sea la función logística, esta función de *loss* es convexa

Tiene un solo mínimo

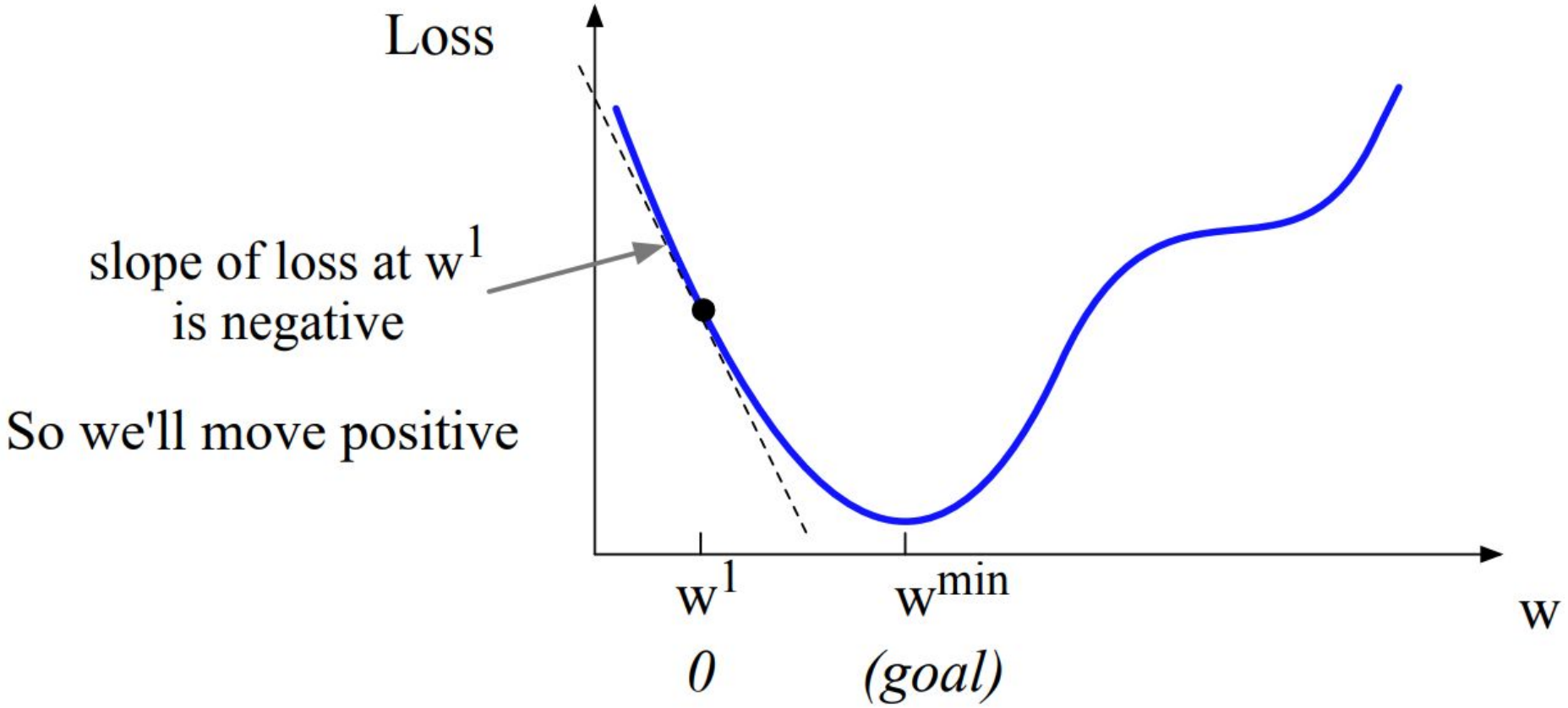
El descenso por gradiente empezando desde cualquier punto seguro llega al mínimo

Esto no es verdad en el caso general de las redes neuronales, pero sí en este ejemplo simplificado

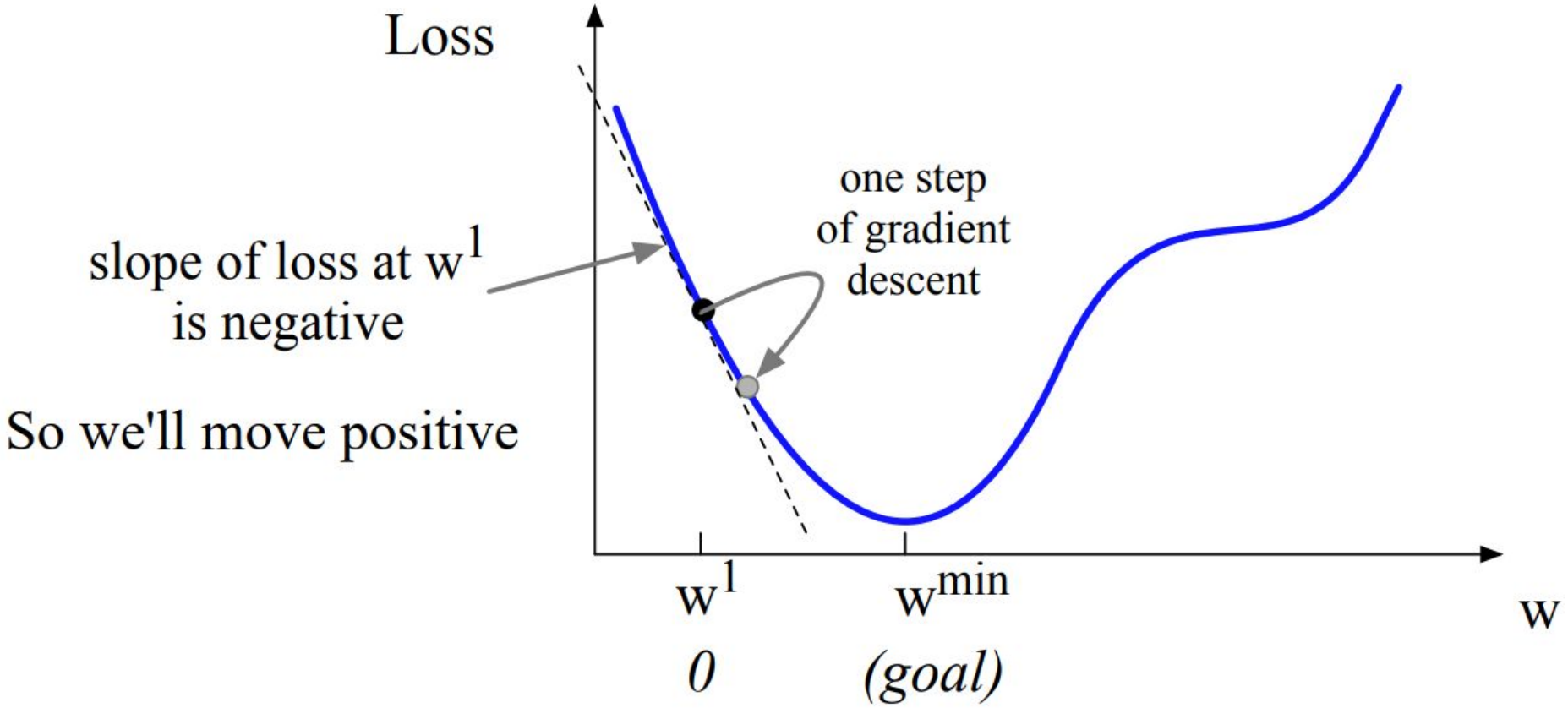
# Visualización en una sola dimensión



# Visualización en una sola dimensión



# Visualización en una sola dimensión



# En varias variables: Gradiente

El gradiente de una función de varias variables es un vector que apunta en la dirección de mayor crecimiento

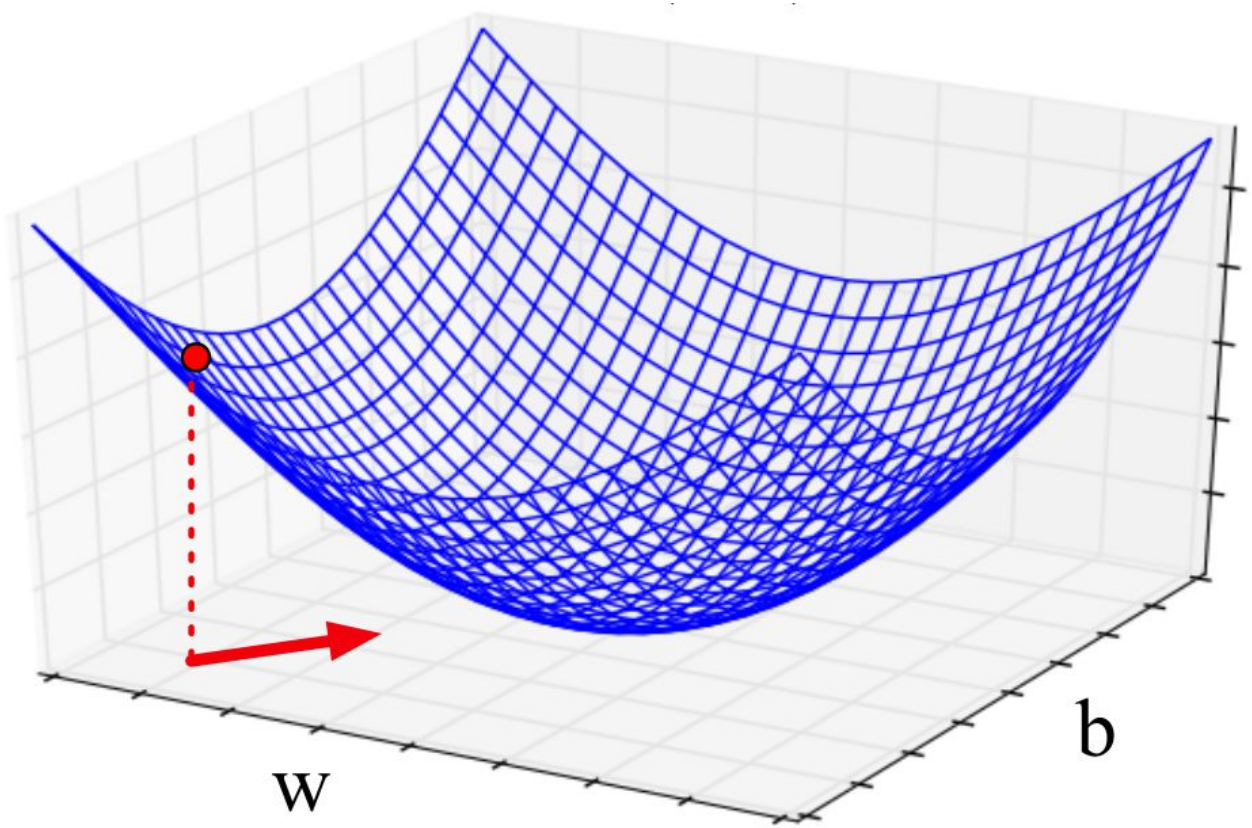
Descenso por gradiente: Significa encontrar el gradiente de la función de *loss* en el punto actual y moverse en la dirección opuesta

Cuánto nos movemos? Un “paso” que llamaremos η

$$w^{t+1} = w^t - \eta \frac{d}{dw} L(f(x;w), y)$$

Learning rate
(tasa de aprendizaje)

# En varias variables: Gradiente



# Gradiente

En una red vamos a tener muchísimos pesos, incluso en una tan simple como la del ejemplo

$$\nabla_\theta L(f(x;\theta),y)) = \begin{bmatrix} \frac{\partial}{\partial w_1} L(f(x;\theta),y) \\ \frac{\partial}{\partial w_2} L(f(x;\theta),y) \\ \vdots \\ \frac{\partial}{\partial w_n} L(f(x;\theta),y) \end{bmatrix}$$

Fórmula para actualizar θ según el gradiente

$$\theta_{t+1} = \theta_t - \eta \nabla L(f(x;\theta),y)$$

# Descenso por gradiente estocástico

**function** STOCHASTIC GRADIENT DESCENT($L()$, $f()$, $x$, $y$) **returns** $\theta$

# where: L is the loss function
# f is a function parameterized by $\theta$
# x is the set of training inputs $x^{(1)}$, $x^{(2)}$,..., $x^{(m)}$
# y is the set of training outputs (labels) $y^{(1)}$, $y^{(2)}$,..., $y^{(m)}$

$\theta \leftarrow 0$
**repeat** til done
  For each training tuple $(x^{(i)}, y^{(i)})$ (in random order)
    1. Optional (for reporting): # How are we doing on this tuple?
       Compute $\hat{y}^{(i)} = f(x^{(i)}; \theta)$ # What is our estimated output $\hat{y}$?
       Compute the loss $L(\hat{y}^{(i)}, y^{(i)})$ # How far off is $\hat{y}^{(i)}$) from the true output $y^{(i)}$?
    2. $g \leftarrow \nabla_\theta L(f(x^{(i)}; \theta), y^{(i)})$ # How should we move $\theta$ to maximize loss?
    3. $\theta \leftarrow \theta - \eta\, g$ # Go the other way instead
return $\theta$

# Hiperparámetros

El learning rate η es un hiperparámetro

- si es muy alto: el procedimiento tomará saltos muy grandes y puede perderse el mínimo global y saltar para el otro lado
- si es muy bajo: el procedimiento tendrá que hacer muchos saltos para llegar al mínimo (muy lento)

Hiperparámetros:

Vamos a distinguir entre los parámetros de una red (los pesos) y los hiperparámetros: valores que no se aprenden al entrenar, sino que se eligen antes de entrenar y afectan el proceso de entrenamiento

Puede haber distintas clases de hiperparámetros

# Descenso por gradiente estocástico

Tres formas de entrenar:

- SGD: Presentando un ejemplo a la vez
- Batch training: Poniendo todos los ejemplos en una matriz y calculando el gradiente para todos a la vez
- Un punto medio que funciona mejor: usar mini-batches

    Se elige una cantidad de ejemplos (por ejemplo 512 o 1024) y se hace la actualización del gradiente con esos

    Luego se elige otro mini-batch hasta agotar el conjunto


Generalicemos los
problemas

# ¿Y si tenemos más de dos categorías?

Ejemplo: competencia TASS de análisis de sentimiento en español

http://tass.sepln.org/

Hasta 2017 cuatro clases: P, N, NEU, NONE

Luego de 2018 tres clases: P, N, NEU

Subconjuntos de varios países! Incluido Uruguay

seria mejor que dejasen de emitir esa basura ya  hay que evolucionar para bien y eso

N

Aunque pensaba que no podría hacerlo hasta el mes de octubre, el lunes próximo volveré a ver a mi familia de #Mataró *.*  #felicidad

P

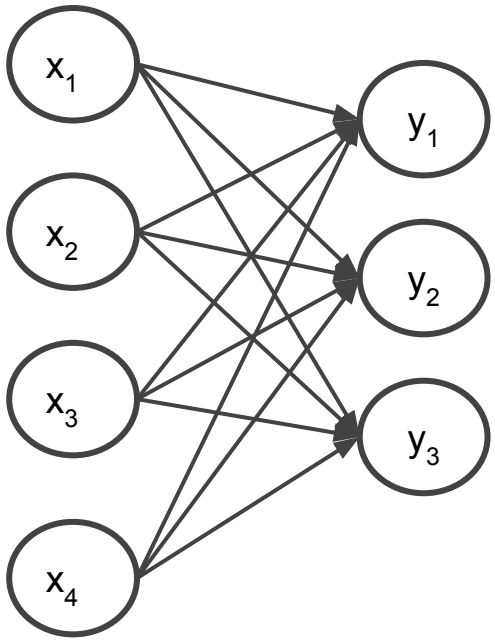
Es muy raro el sentimiento que tengo ahora, aunque en fin... Qué más dará

NEU

cuando estén los 20 contactados y aprobados se publicará el listado en nuestra web

NONE

# Clasificación Multiclase




Problemas de clasificación discretos, queremos que la salida sea una distribución de probabilidad

Función de activación *softmax*

$$P(j|x) = \frac{e^{y_j}}{\sum_k e^{y_k}}$$

Cada $y_i$ es combina los valores de las demás para normalizar a una probabilidad

# Clasificación Multiclase

¿Cómo queda la función de loss?

Sean N ejemplos $y_i$ cada uno con una categoría entre k

Pero para cada $y_i$ solo una de las clases es la correcta

O sea en cada caso: $y_{ij}$ = 1, los demás son 0

La entropía cruzada en este caso es:

$$L_{CE} = -\frac{1}{N} \sum_{i=1, y_{ij}=1}^{N} log(\hat{y}_{ij})$$

# Problemas de regresión

Ejemplo: task 2 de las competencias HAHA de análisis de humor en español

https://www.fing.edu.uy/inco/grupos/pln/haha/

Task 1: ¿Es un chiste o no?

Task 2: ¿Qué tan bueno es de 1 a 5?

Otros tasks…

Hay 10 tipos de personas, los que entienden binario y los que no.

4.5 ★ HUMOR

¿¡Cómo que el 15% del segundo aguinaldo debe destinarse a una sola empresa nacional!?
¿Es cierto eso?

NO HUMOR

Buenos días a todos, menos a los que se la pasan criticando a los demás, en lugar de arreglar su propia vida.

NO HUMOR

- Hola guapa, ¿me dices tu teléfono?
- Un iPhone.
- Pero el número.
- El 5.

1.6 ★ HUMOR

# Problemas de regresión

Una forma de tratar estos problemas es directamente dejar activación lineal

De esta manera se puede obtener cualquier valor como salida

Sean N ejemplos, cada uno con un valor real $y_i$ , la red devuelve un valor $\hat{y}_i$

En estos casos se suele usar el error cuadrático medio como función de loss:

$$MSE = \frac{1}{m}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2$$

# Funciones de Activación

- Función logística o sigmoide
- ReLU
- Tangente hiperbólica

Otras…

- Identidad (salida lineal)
- Arcotangente
- Leaky ReLU
- SiLU
- Swish

Ojo! No se usa! Luego veremos por qué

¿Qué tienen que cumplir?

Derivable

Monótona ?

No lineal

# Funciones de Loss

- Entropía cruzada
- Entropía cruzada categórica
- Error cuadrático medio

Otras…

¿Qué tienen que cumplir?

Derivable

Valores más bajos cuando los resultados de la red se parecen más a los esperados