UT2

# Deep Learning

Fundamentos del Aprendizaje Automático

Profesor: Ing. Juan Francisco Kurucz

juan.kuruczsosa@ucu.edu.uy

# Deep Learning

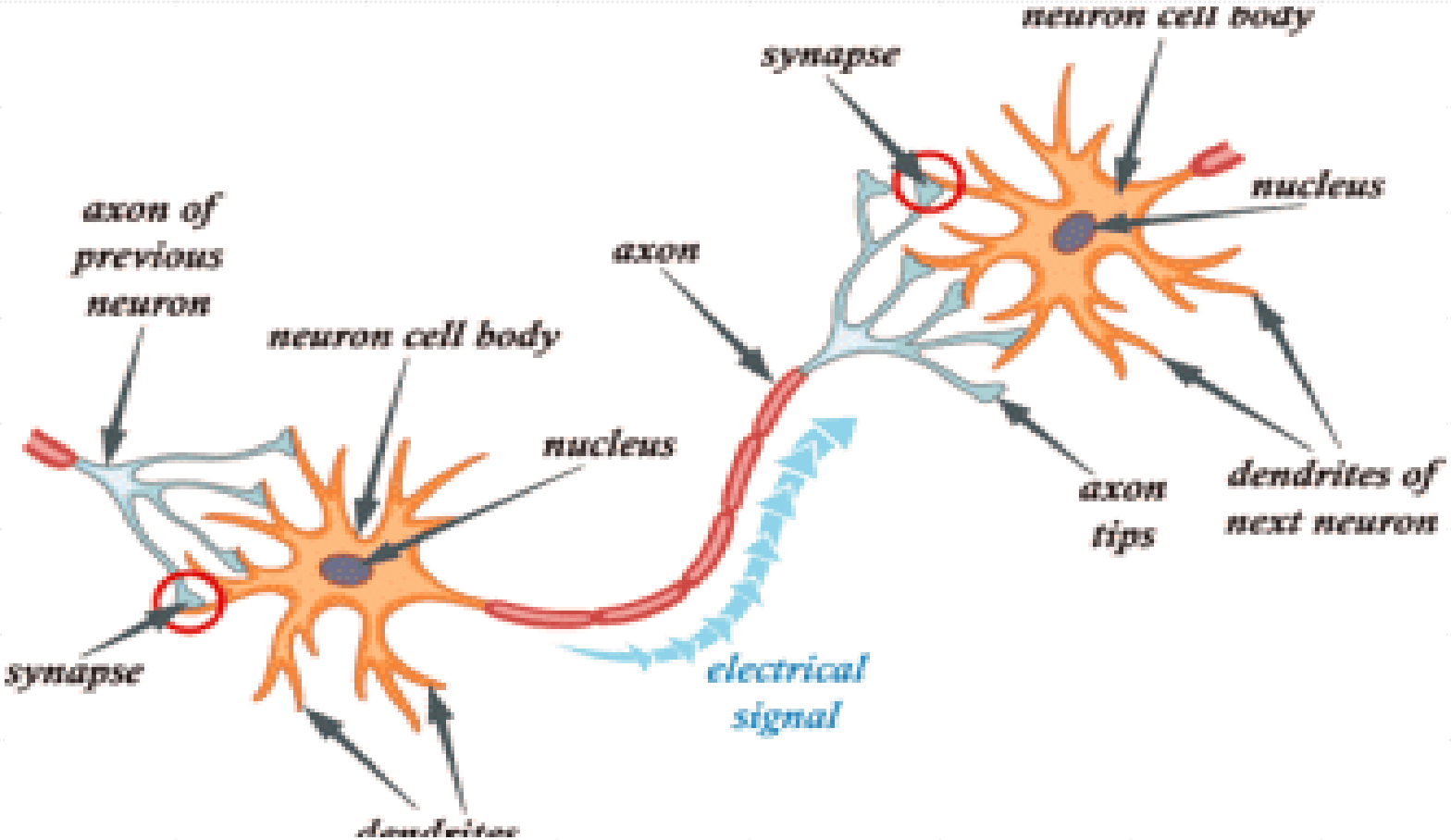
- Neuronas
  - Perceptron
  - Funciones de activacion
- Redes de neuronas
  - Tipos de neuronas

# ¿Qué es una red neuronal?

- Representan funciones:
  - valores discretos o enteros,
  - valores numéricos (reales),
  - vectores formados por los anteriores.
- Algoritmos de aprendizaje (entrenamiento).
  - Entrenamiento puede ser lento.
  - Evaluación rápida.
- No es interpretable.

# Metáfora biológica

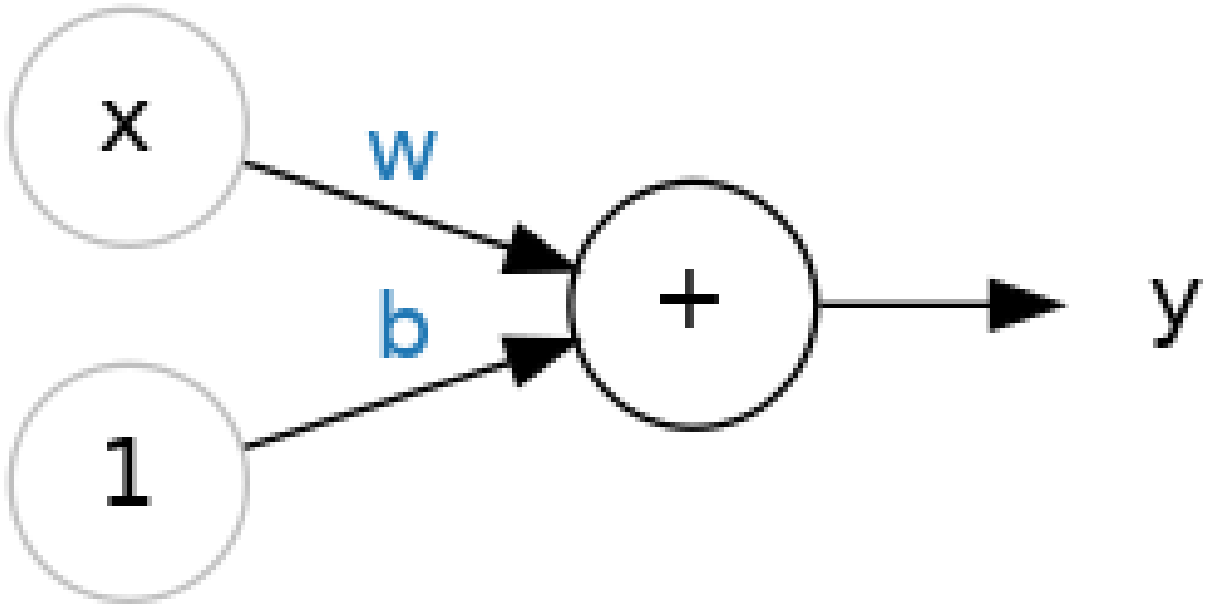
- Inspiradas en los sistemas nerviosos de los animales.



# Neuronas artificiales

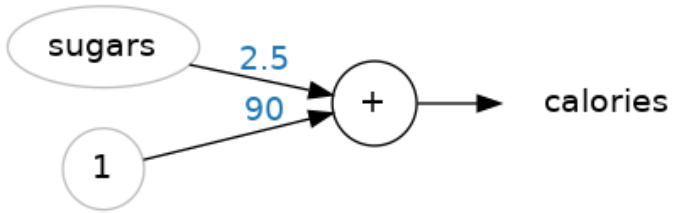
- Abstracción matemática.
  - Unidades que adoptan un valor.
  - Conexiones  o *sinapsis* con pesos.
  - Salida modificada por una función de activación.
- Binary Threshold Unit - McCulloch & Pitts (1943).
- Perceptrón - Frank Rosenblatt (1958)

# Perceptron

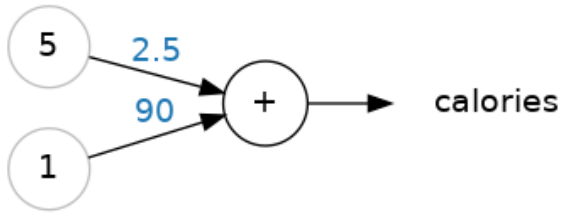



y=wx+b

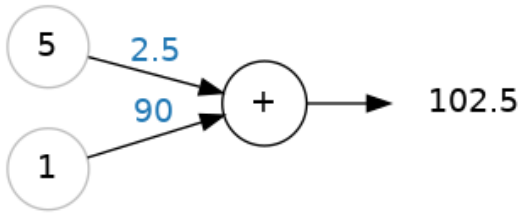
# Perceptron




w=2.5, b=90

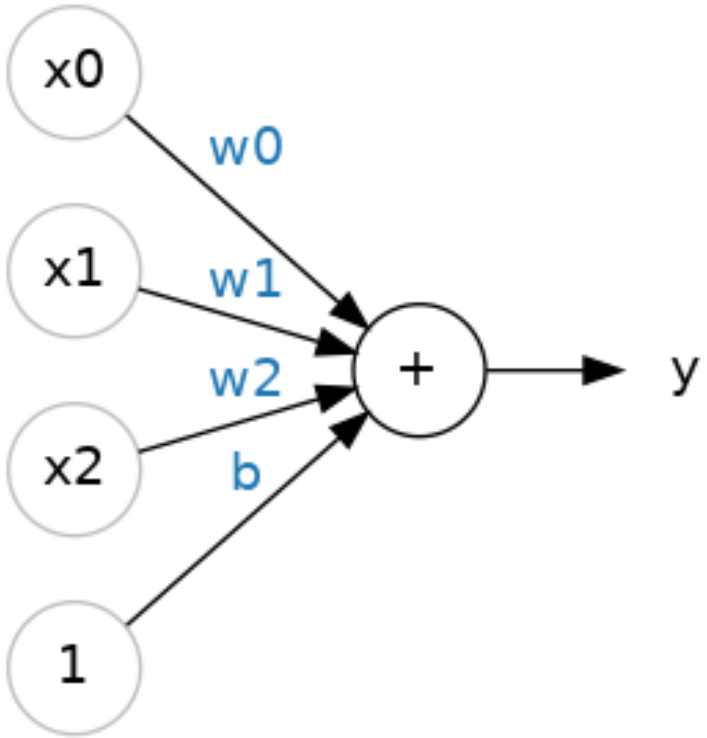



Input sugars=5

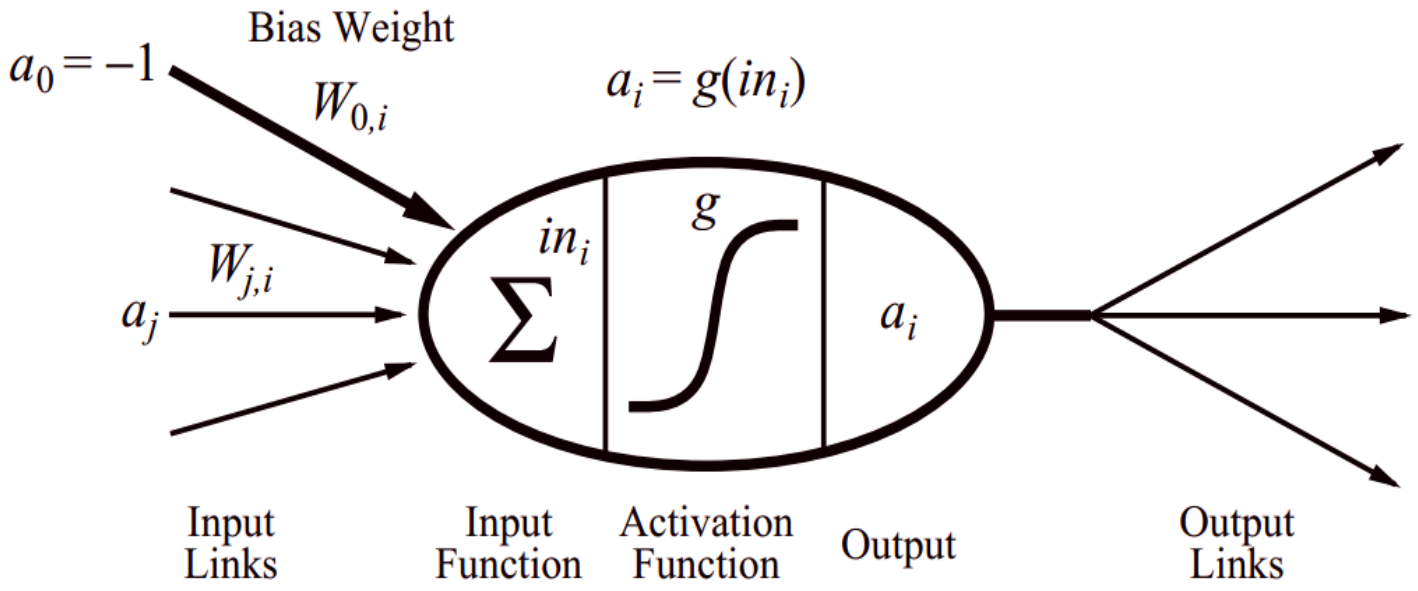



Multiply inputs by weights and sum.

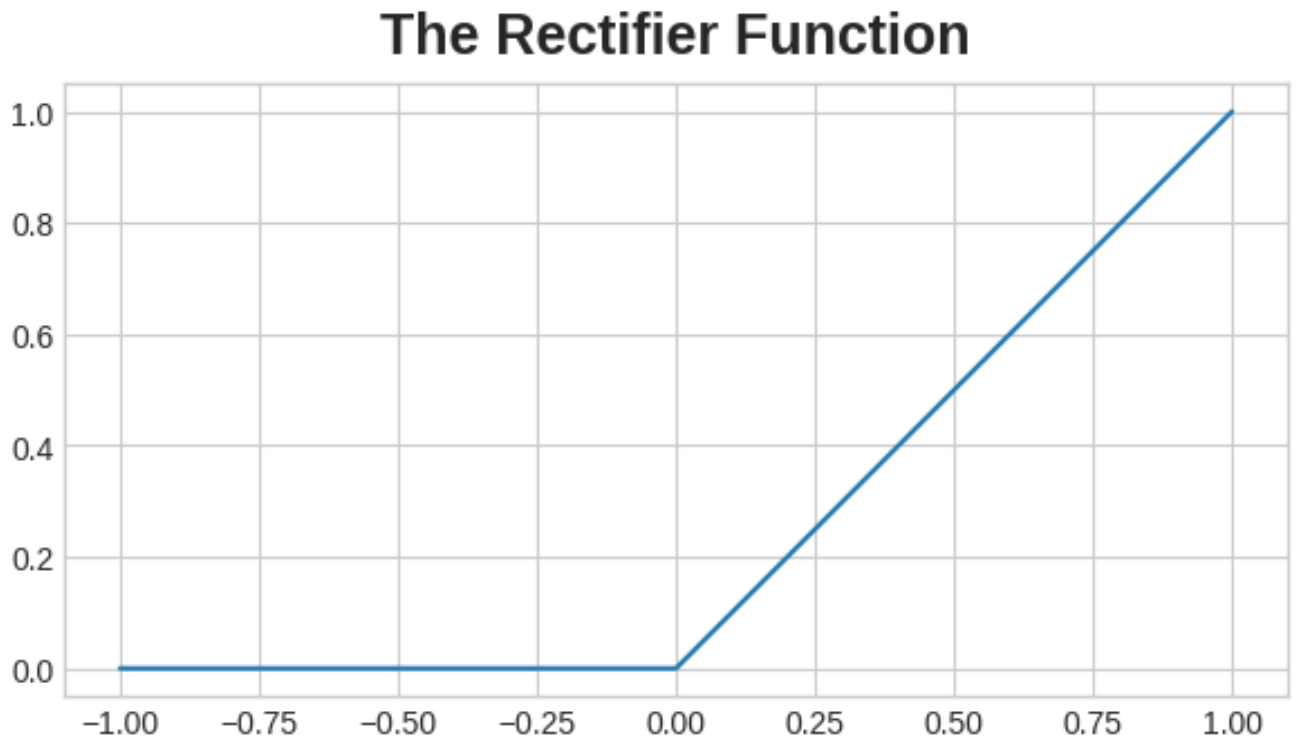
# Perceptron



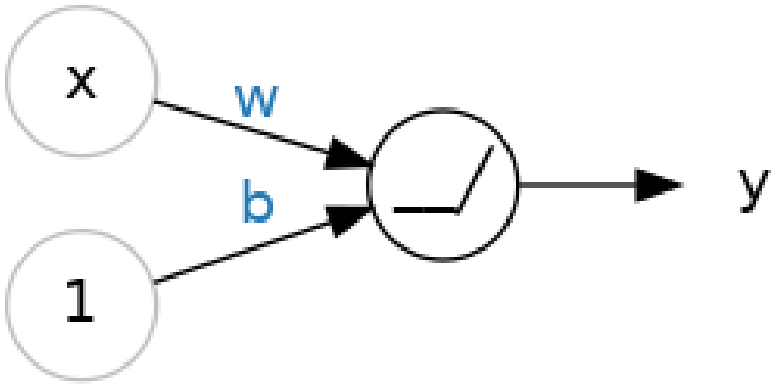
# Perceptron


Bias Weight
$a_0 = -1$
$W_{0,i}$
$a_i = g(in_i)$
$W_{j,i}$
$a_j$
$in_i$
$\Sigma$
$g$
$a_i$
Input Links
Input Function
Activation Function
Output
Output Links

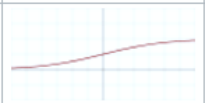
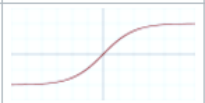
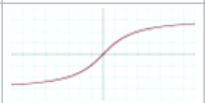
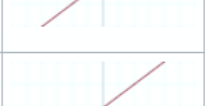
# Funciones de activación



# Funciones de activación

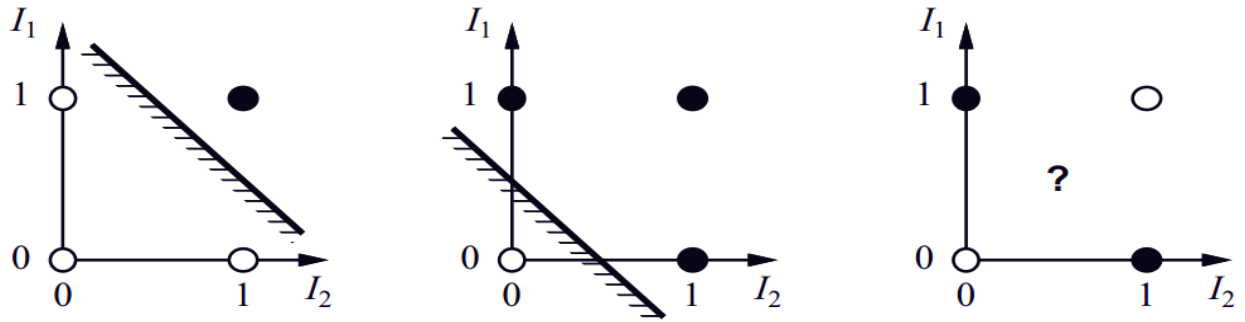


# Funciones de activación

| Name | Plot | Equation | Derivative |
|---|---|---|---|
| Identity |  | $f(x) = x$ | $f'(x) = 1$ |
| Binary step |  | $f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$ | $f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$ |
| Logistic (a.k.a Soft step) |  | $f(x) = \frac{1}{1+e^{-x}}$ | $f'(x) = f(x)(1-f(x))$ |
| TanH |  | $f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1$ | $f'(x) = 1 - f(x)^2$ |
| ArcTan |  | $f(x) = \tan^{-1}(x)$ | $f'(x) = \frac{1}{x^2+1}$ |
| Rectified Linear Unit (ReLU) |  | $f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$ | $f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$ |
| Parameteric Rectified Linear Unit (PReLU) [2] |  | $f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$ | $f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$ |
| Exponential Linear Unit (ELU) [3] |  | $f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$ | $f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$ |
| SoftPlus |  | $f(x) = \log_e(1+e^x)$ | $f'(x) = \frac{1}{1+e^{-x}}$ |

# Interpretación Geométrica

- w = [w0, w1, ..., wn]  x = [1, x1, x2, ..., xn]
- w . x =  0 define un hiperplano en el espacio de entrada
- Separa los casos según su clase
- El perceptrón umbral  es llamado “separador lineal”




(a) $I_1$ **and** $I_2$ (b) $I_1$ **or** $I_2$ (c) $I_1$ **xor** $I_2$

# Aprendizaje simple del perceptrón

Entrenamiento:

1. **Datos** → ejemplos con entradas (X) y salidas (Y).
2. **Modelo** → fórmula con pesos, inicializados aleatoriamente
3. **Predice** → calcula valores con pesos iniciales.
4. **Error** → compara predicción vs. valor real.
5. **Ajusta** → corrige pesos según el error.
6. **Repite** → predecir → medir → ajustar.
7. **Final** → pesos “aprendidos” que minimizan error.

# Aprendizaje simple del perceptrón

# Aprendizaje simple del perceptrón

Actualizar los pesos de acuerdo a:

$$w_j(k+1) = w_j(k) + \Delta w_j(k)$$

$$\Delta w_j(k) = \eta(d - y)x_j$$

Donde $k$ es la iteración, $w$ es el peso, $d$ es la salida deseada, $y$ es la salida obtenida, y $\eta$ (entre 0.0 y 1.0) la taza de aprendizaje.

# Aprendizaje simple del perceptrón

$$\Delta w_j(k) = \eta(d - y)x_j$$

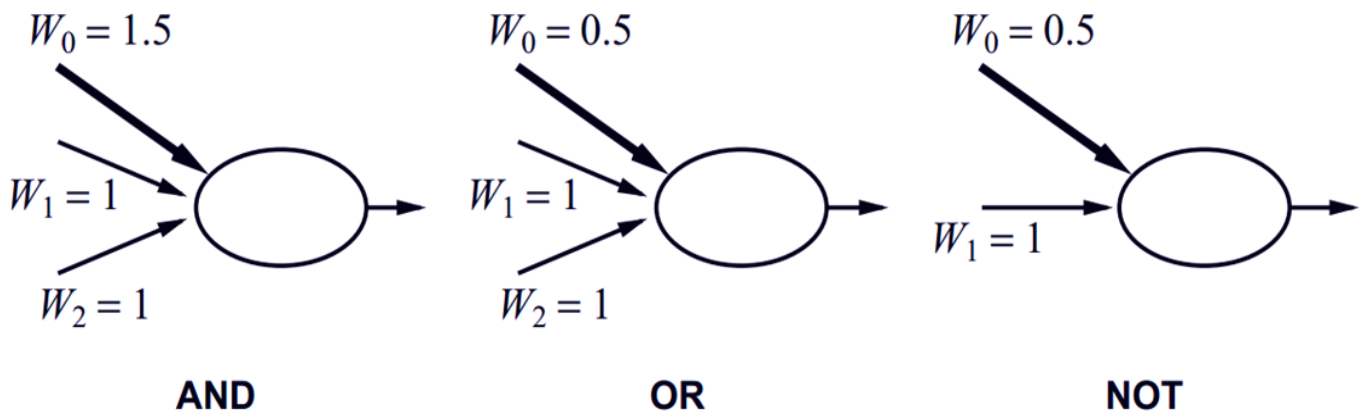
tasa de aprendizaje

error

$$w_j(k+1) = \begin{cases} w_j(k) & si\ y = d \\ w_j(k) + \eta x_j & si\ y=0\ d=1 \\ w_j(k) - \eta x_j & si\ y=1\ d=0 \end{cases}$$

# TA7 - EJ1 – Entendimiento de perceptron

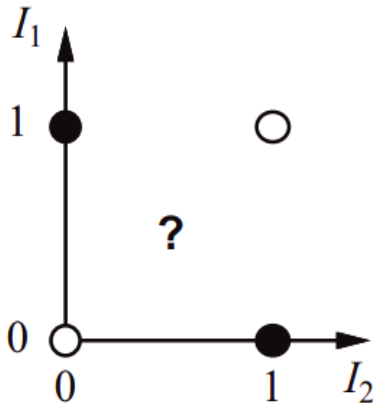
# Funciones Booleanas




- ¿podemos representar el XOR?
- ¿Por qué no?

# Limitaciones del perceptrón

- Minsky y Papert (1968)
- Existen problemas en los que no existe un hiperplano que separa los casos
- Es necesario múltiples capas de unidades

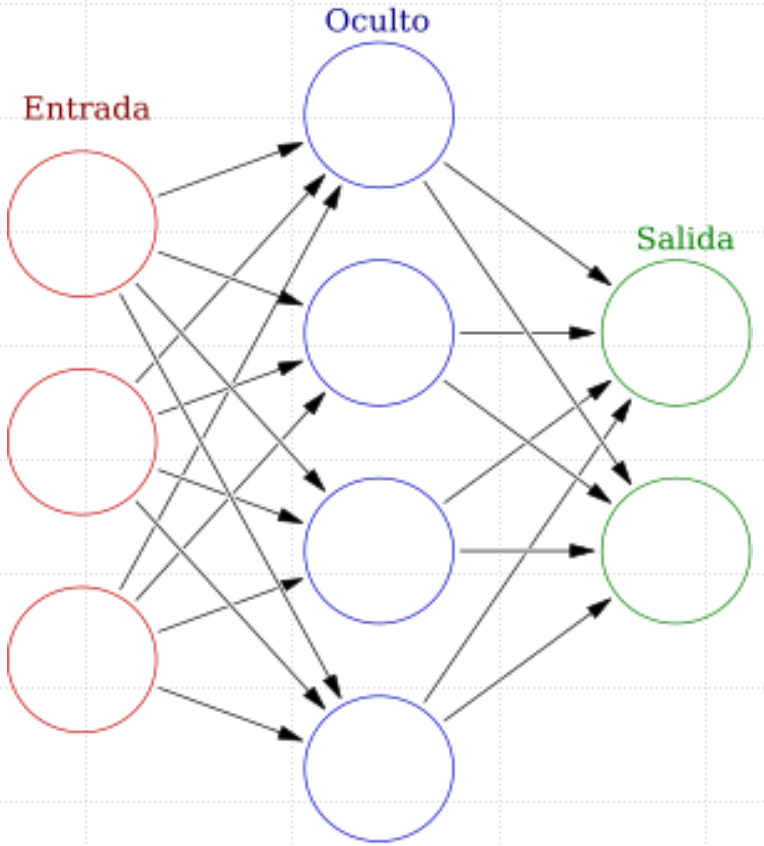



(c) $I_1$ **xor** $I_2$

# ¿Qué es una red neuronal?

Una red neuronal es **suma ponderada + giro no lineal**, repetida en **capas**.

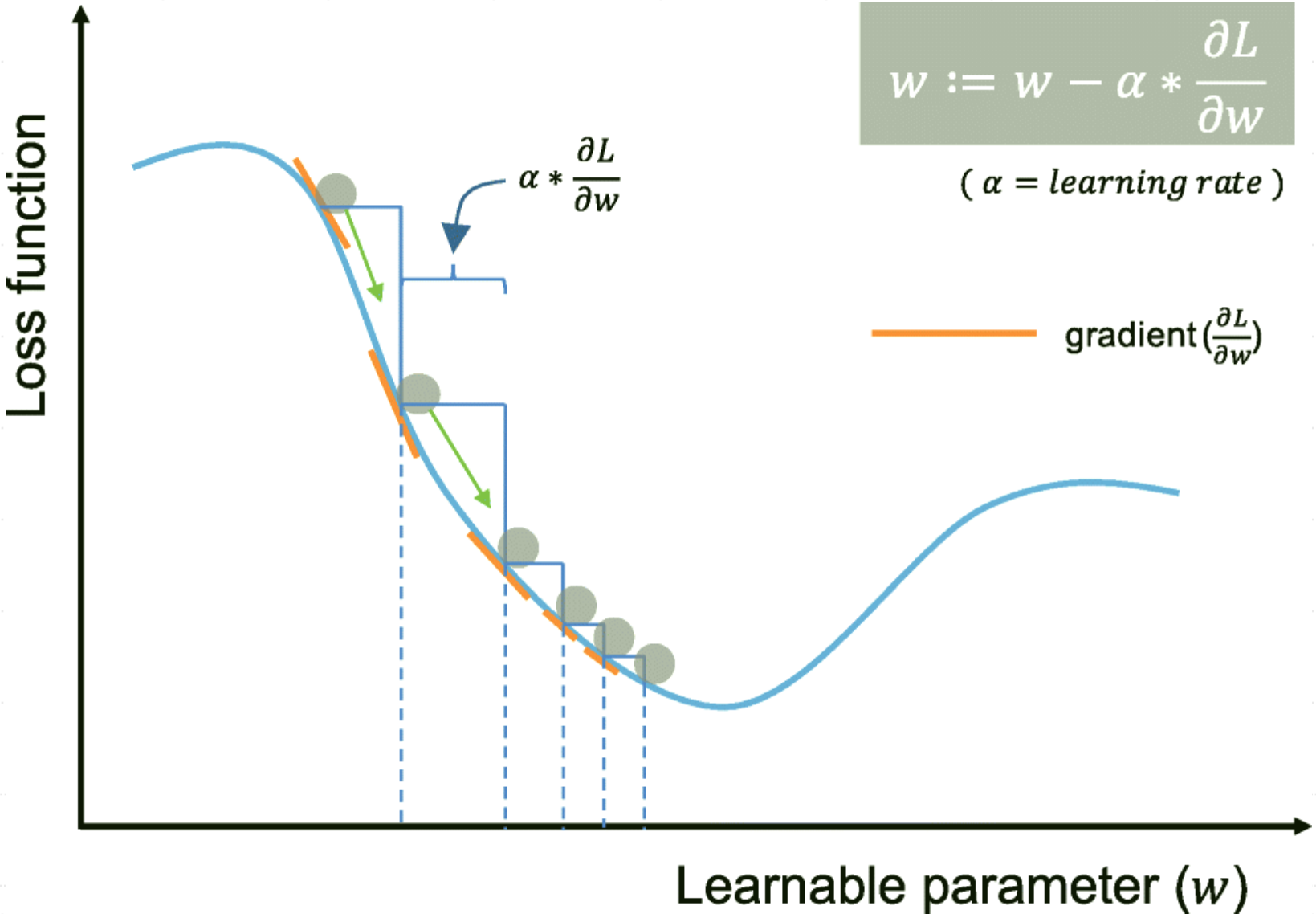
- **Perceptrón:** multiplica entradas por **pesos (w)**, suma **sesgo (b)** → saca un número.
- **Capa densa:** muchas neuronas con las **mismas entradas**.
- **MLP (feedforward):** capas **en cadena**: entrada → ocultas → salida.



# Gradient Descent

$$w := w - \alpha * \frac{\partial L}{\partial w}$$

( $\alpha$ = *learning rate* )



# TA7 – Continuación