# Fundamentos de las Redes Neuronales Profundas

## TLU's y Perceptrones

**Clase 4**

Dra. Wendy Aguilar

# Modelos Generativos Profundos

UN ENFOQUE DESDE LA CREATIVIDAD COMPUTACIONAL

¿Cuáles son los orígenes del aprendizaje profundo?

# Orígenes del Aprendizaje Profundo

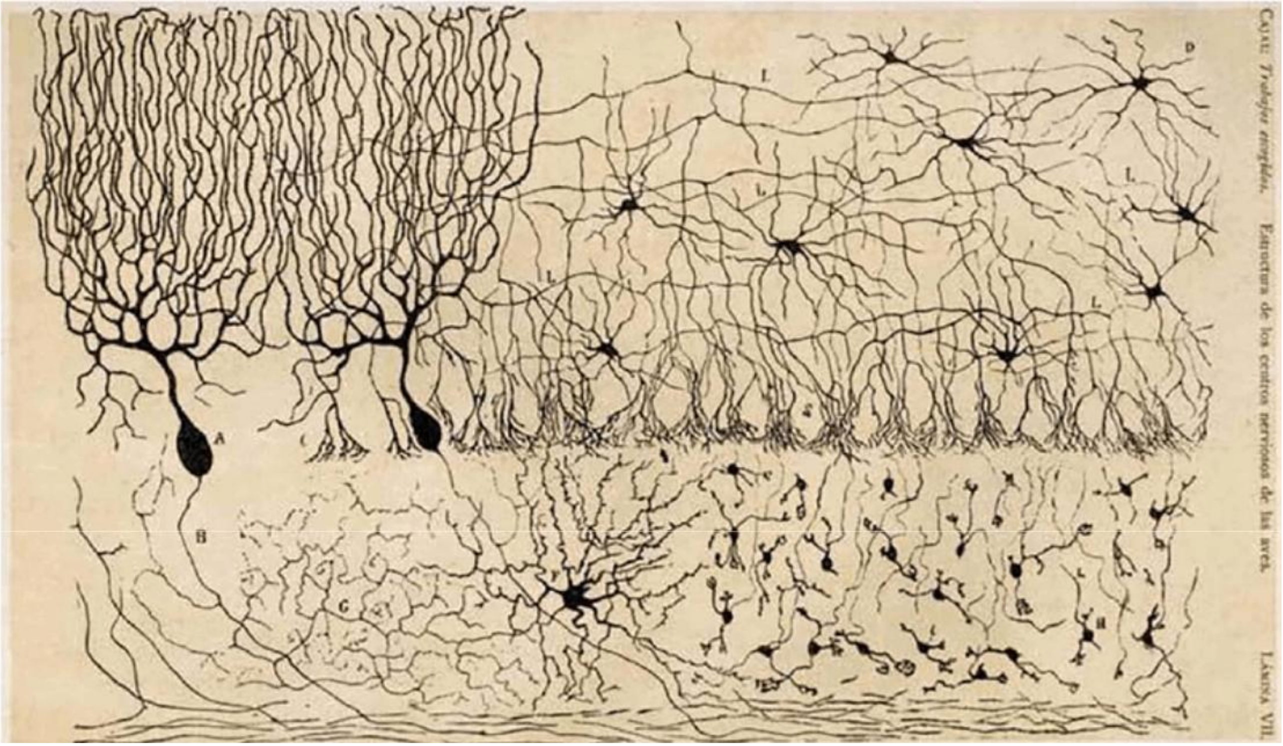
- Muchos de los modelos recientes se basan en descubrimientos realizados hace décadas, que han sido revitalizados gracias a:
    - Enormes recursos computacionales disponibles en la nube
    - Hardware especializado para cálculos paralelos con matrices, como:
        - GPUs (Unidades de Procesamiento Gráfico)
        - TPUs (Unidades de Procesamiento Tensorial)
- Si consideramos que la investigación sobre redes neuronales incluye tanto su inspiración como la teoría computacional, este campo tiene **más de 100 años de antigüedad**.

# Orígenes del Aprendizaje Profundo

- Muchos de los modelos recientes se basan en descubrimientos realizados hace décadas, que han sido revitalizados gracias a:

    - Enormes recursos computacionales disponibles en la nube
    - Hardware especializado para cálculos paralelos con matrices, como:
        - GPUs (Unidades de Procesamiento Gráfico)
        - TPUs (Unidades de Procesamiento Tensorial)

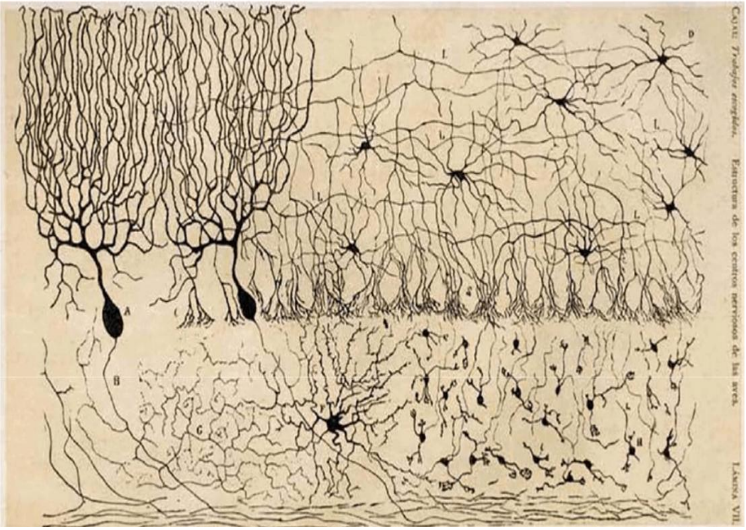
- Si consideramos que la investigación sobre redes neuronales incluye tanto su inspiración como la teoría computacional, este campo tiene **más de 100 años de antigüedad**.



Una de las **primeras redes neuronales** descritas aparece en las detalladas ilustraciones anatómicas del científico del siglo **XIX** Santiago Ramón y Cajal (basadas en observaciones experimentales de capas de células neuronales interconectadas).

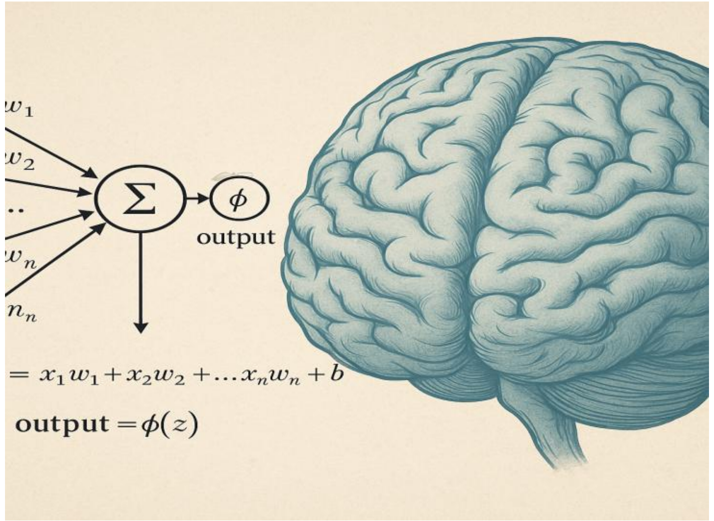
Las capas diferenciadas de la **retina** observadas por Cajal sirvieron como **inspiración** para arquitecturas específicas de ANNs como las **CNNs**.

# Orígenes del Aprendizaje Profundo



Esta observación de células neuronales simples interconectadas en grandes redes …

Llevo a la idea de …




Representar la actividad mental mediante operaciones lógicas simples que, combinadas, dieran lugar a fenómenos mentales complejos.

Intentar **representar el funcionamiento del cebro mediante fórmulas matemáticas.**

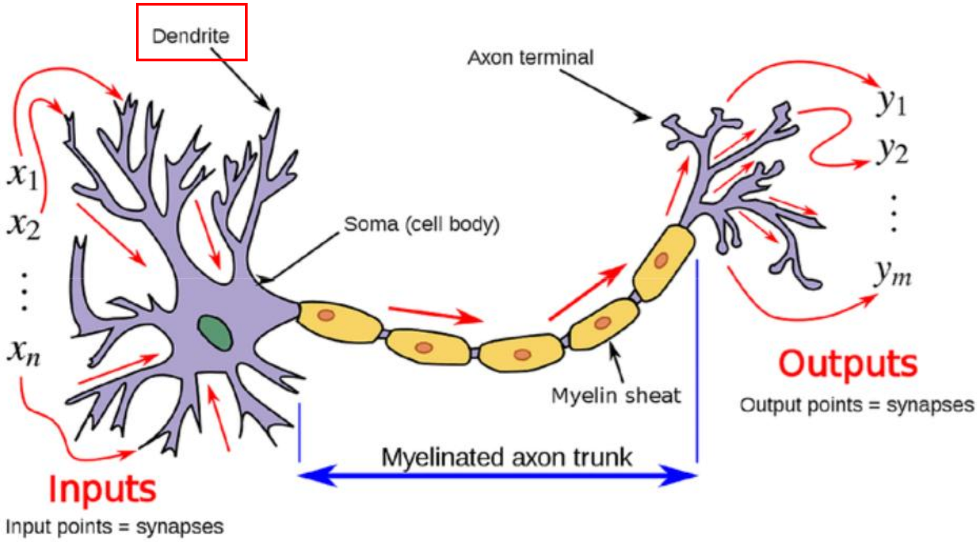
# Threshold Logic Unit (TLU)

1943, McCulloch & Pitts (MIT)

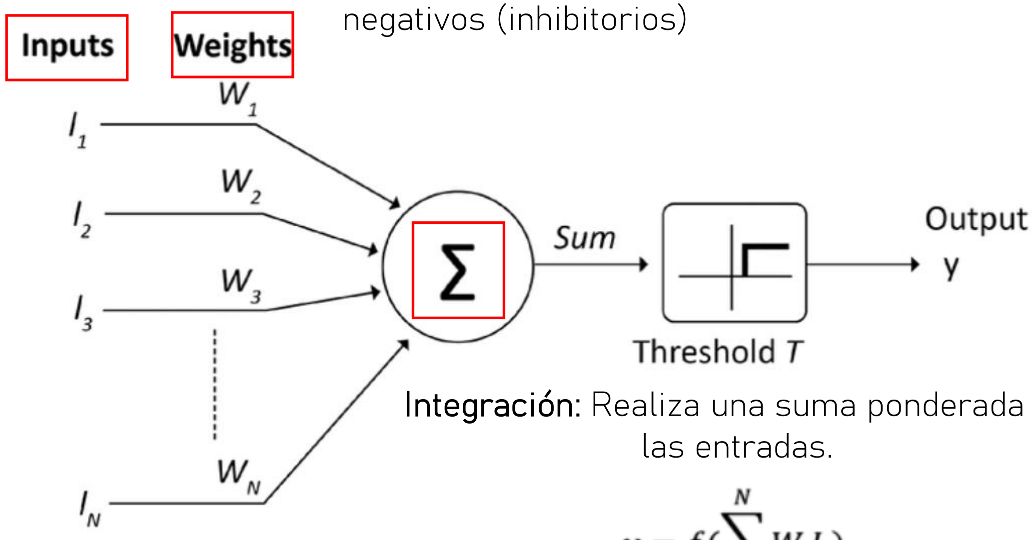
Entradas binarias se traducen en una salida binaria basada en un umbral

**Reciben señales de entrada** de otras neuronas a través de sinapsis.

**Integran y modulan** la fuerza de la señal de entrada




**Modulación:** Indican la importancia de cada entrada: pueden ser positivos (excitatorios) o negativos (inhibitorios)




**Integración:** Realiza una suma ponderada de las entradas.

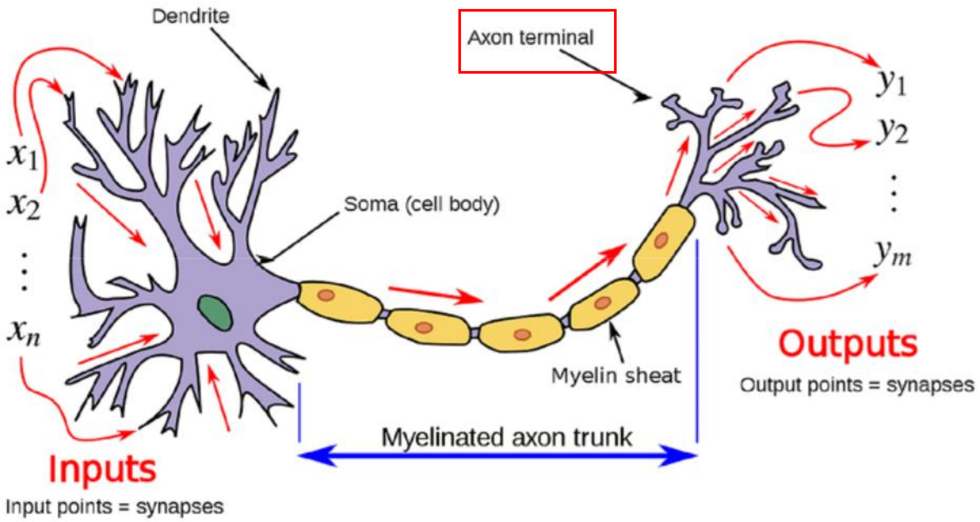
$$y = f(\sum_{i=1}^{N} W_i I_i)$$

# Threshold Logic Unit (TLU)

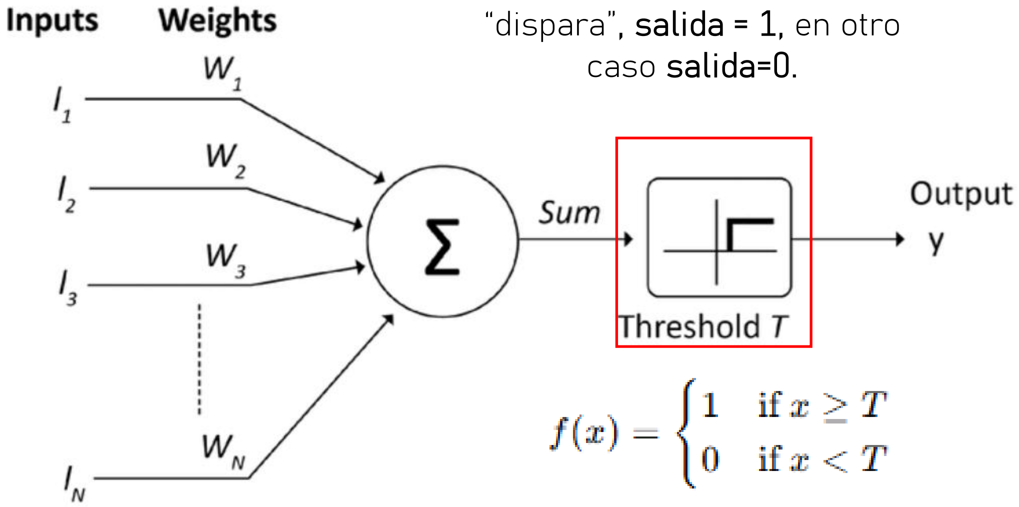
## 1943, McCulloch & Pitts (MIT)

Entradas binarias se traducen en una salida binaria basada en un umbral

Responsable de **disparar el potencial de acción**. **Si la despolarización alcanza cierto nivel** (~–55 mV), la neurona **dispara un pulso eléctrico**,




**Valor fijo T**: si la suma supera el umbral T, la neurona "dispara", **salida = 1**, en otro caso **salida=0**.



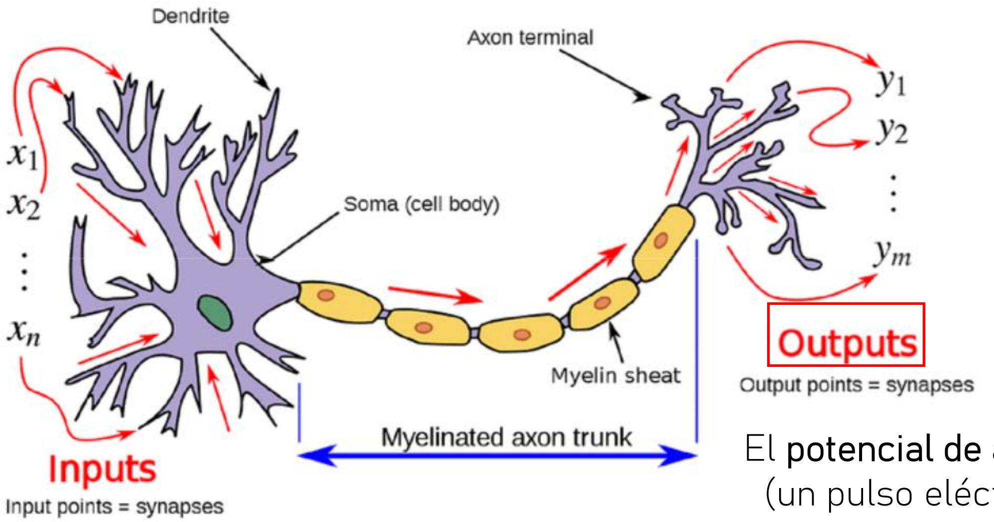

$$f(x) = \begin{cases} 1 & \text{if } x \geq T \\ 0 & \text{if } x < T \end{cases}$$

Función escalón

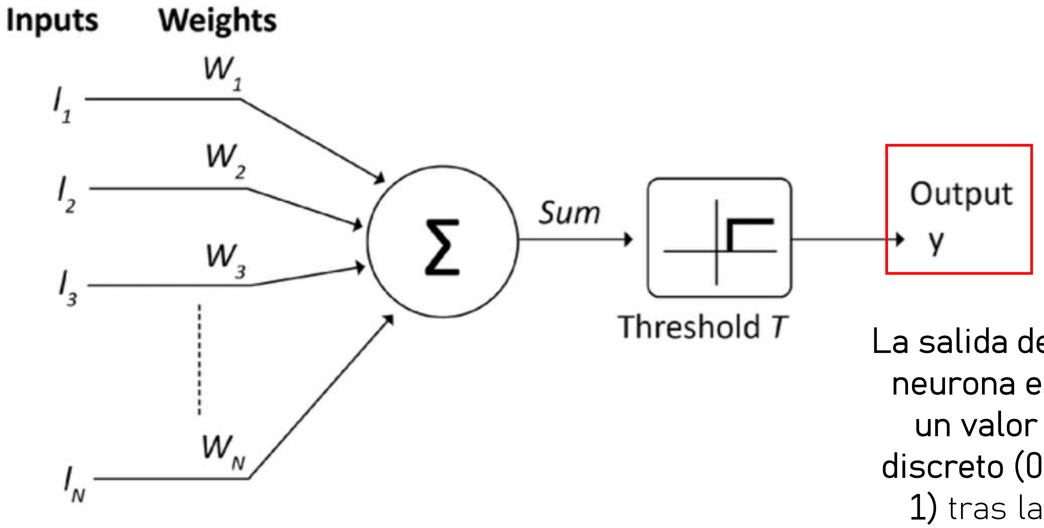
# Threshold Logic Unit (TLU)

1943, McCulloch & Pitts (MIT)

Entradas binarias se traducen en una salida binaria basada en un umbral




El **potencial de acción** (un pulso eléctrico) recorre el axón y se convierte en la salida de la neurona).



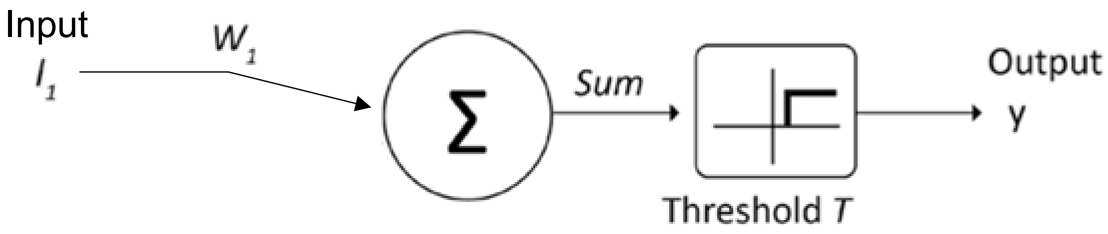

**La salida de a neurona es un valor discreto (0 o 1)** tras la función escalón.

Podemos imaginar que, así como las células neuronales **se agrupan en redes** para formar circuitos biológicos complejos, **estas unidades simples podrían conectarse** para simular procesos de decisión sofisticados.

# Threshold Logic Unit (TLU)

1943, McCulloch & Pitts (MIT)

Usando este modelo simple ya podemos empezar a representar varias operaciones lógicas.




| Identity | |
|---|---|
| **Input** | **Output** |
| 1 | 1 |
| 0 | 0 |

$w_1$ =1

*0<T<=1, p.ej. T=0.5*

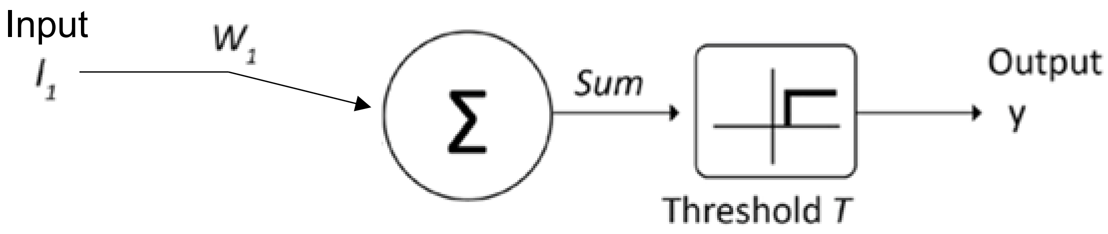
$$y = f\left(\sum_{i=1}^{N} W_i I_i\right)$$

$$f(x) = \begin{cases} 1 & \text{if } x \geq T \\ 0 & \text{if } x < T \end{cases}$$

# Threshold Logic Unit (TLU)

1943, McCulloch & Pitts (MIT)

Usando este modelo simple ya podemos empezar a representar varias operaciones lógicas.




| Identity | |
|---|---|
| Input | Output |
| 1 | 1 |
| 0 | 0 |

$w_1$ =1
*0<T<=1, p.ej. T=0.5*

$$y = f\left(\sum_{i=1}^{N} W_i I_i\right)$$

$$f(x) = \begin{cases} 1 & \text{if } x \geq T \\ 0 & \text{if } x < T \end{cases}$$

| Negation | |
|---|---|
| Input | Output |
| 1 | 0 |
| 0 | 1 |

¿Qué valor tendría que tener $w_1$ y $T$ para funcionar?

# Implementación del TLU

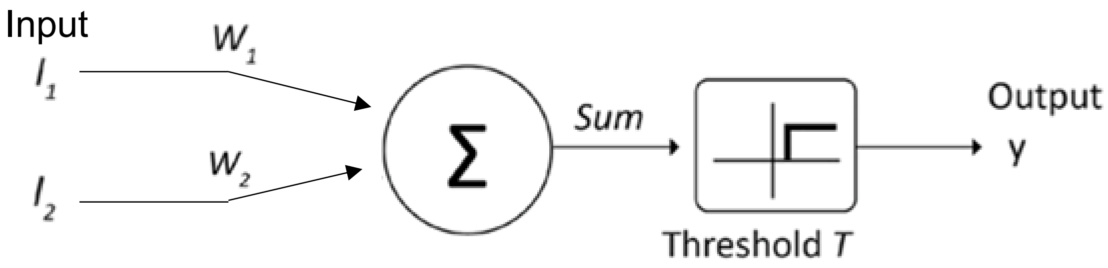
ejercicios_TLU_y_perceptron.ipynb

Ejercicios 1, 2 y 3

# Threshold Logic Unit (TLU)

1943, McCulloch & Pitts (MIT)

¿Cómo se podrían representar el AND y el OR con el TLU?




$$y = f\left(\sum_{i=1}^{N} W_i I_i\right)$$

$$f(x) = \begin{cases} 1 & \text{if } x \geq T \\ 0 & \text{if } x < T \end{cases}$$

**AND**

| Input 1 | Input 2 | Output |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |

¿Qué valor tendría que tener $w_1$ $w_2$ y T para funcionar?

**OR**

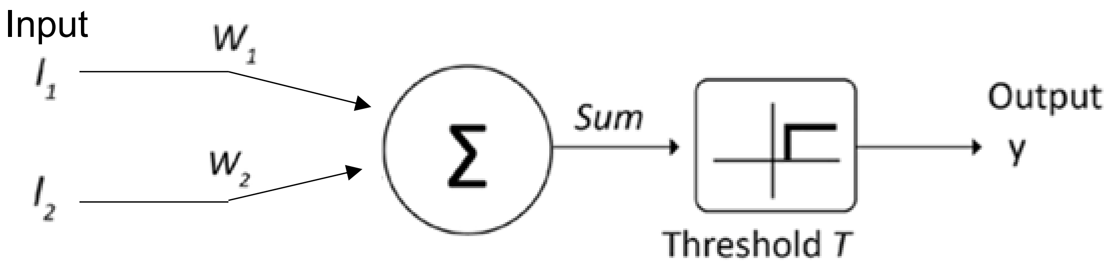
| Input 1 | Input 2 | Output |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

¿Qué valor tendría que tener $w_1$ $w_2$ y T para funcionar?

# Threshold Logic Unit (TLU)

1943, McCulloch & Pitts (MIT)

¿Cómo se podrían representar el AND y el OR con el TLU?

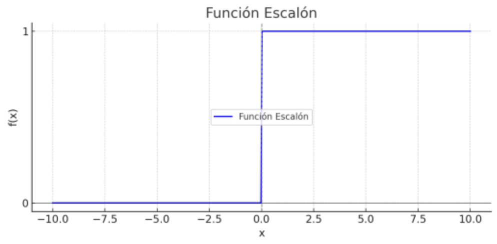


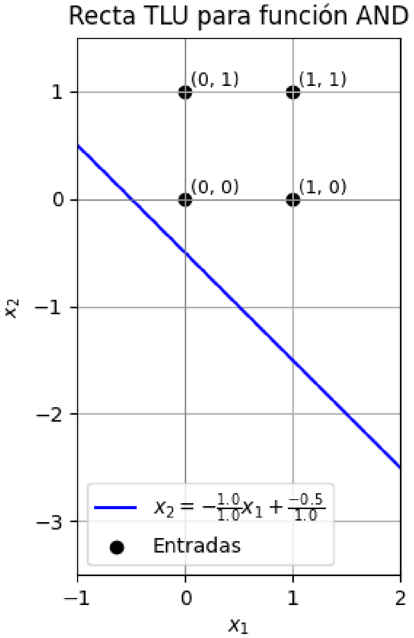

$$y = f\left(\sum_{i=1}^{N} W_i I_i\right)$$

$$f(x) = \begin{cases} 1 & \text{if } x \geq T \\ 0 & \text{if } x < T \end{cases}$$

| AND | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |

¿Qué valor tendría que tener $w_1$ $w_2$ y T para funcionar?




## Ecuación de la frontera de decisión

Es el conjunto de puntos donde la salida cambia: cuando el argumento del escalón es igual al umbral.

$$w_1 x_1 + w_2 x_2 = T$$

$$w_2 x_2 = T - w_1 x_1$$

$$x_2 = \frac{T - w_1 x_1}{w_2}$$

$$x_2 = -\frac{w_1}{w_2} x_1 + \frac{T}{w_2}$$

Es la ecuación de una recta de la forma:

$$y = mx + b$$

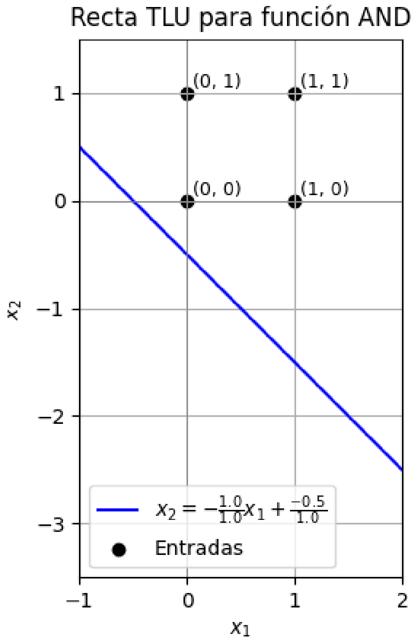
Con pendiente $-\frac{w_1}{w_2}$ y con intersección con el eje $x_2$ en $\frac{T}{w_2}$

¿Para esta recta, qué valores tiene $w_1$, $w_2$ y $T$?

# Implementación del TLU

| AND | | |
|---|---|---|
| **Input 1** | **Input 2** | **Output** |
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |

¿Qué valor tendría que tener $w_1$ $w_2$ y T para funcionar?



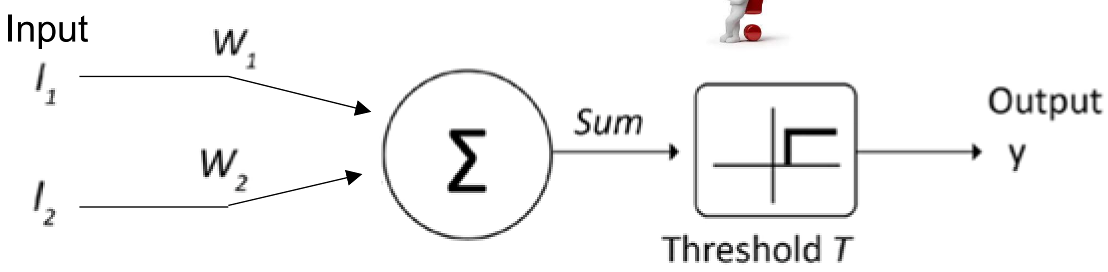

ejercicios_TLU_y_perceptron.ipynb

Ejercicios 4 y 5.

# Threshold Logic Unit (TLU)

1943, McCulloch & Pitts (MIT)

¿Se podrá representar la operación XOR con el TLU?




ejercicios_TLU_y_perceptron.ipynb

Ejercicio 6

$$y = f\left(\sum_{i=1}^{N} W_i I_i\right)$$

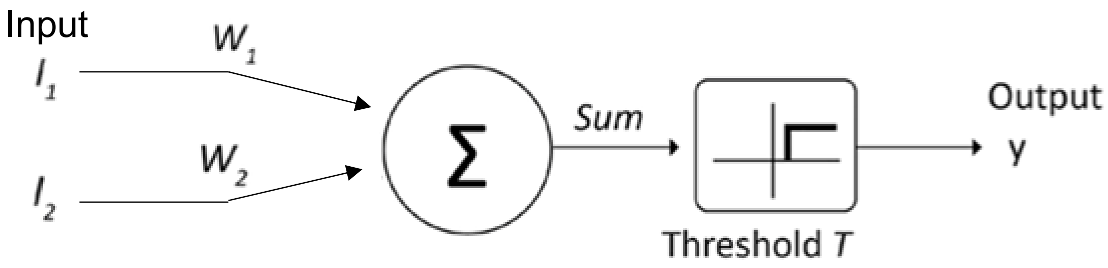
$$f(x) = \begin{cases} 1 & \text{if } x \geq T \\ 0 & \text{if } x < T \end{cases}$$

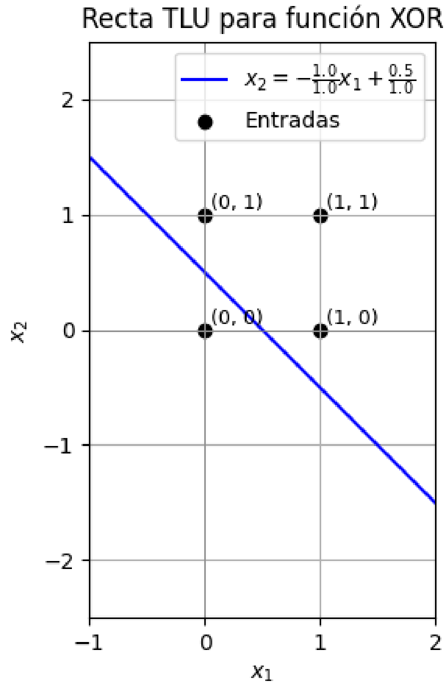
| XOR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 0 |

¿Qué valor tendría que tener $w_1$ $w_2$ y T para funcionar?

# Threshold Logic Unit (TLU)

1943, McCulloch & Pitts (MIT)




$$y = f\left(\sum_{i=1}^{N} W_i I_i\right)$$

$$f(x) = \begin{cases} 1 & \text{if } x \geq T \\ 0 & \text{if } x < T \end{cases}$$

| XOR | | |
|---|---|---|
| **Input 1** | **Input 2** | **Output** |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 0 |

¿Qué valor tendría que tener $w_1$ $w_2$ y T para funcionar?

# Limitaciones de los TLUs

Limitaciones de los TLUs

- No pueden representar las operaciones XOR y XNOR.
- Los pesos son fijos.
- La salida solo puede ser binaria (0 o 1).

- Para que un sistema como una neurona pueda "aprender", necesita responder al entorno y determinar la relevancia de diferentes entradas basándose en la retroalimentación de experiencias previas.

$$y = f(\sum_{i=1}^{N} W_i I_i)$$

# Limitaciones de los TLUs

### Limitaciones de los TLUs

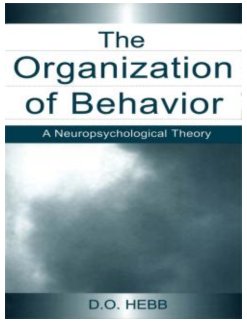
- No pueden representar las operaciones XOR y XNOR.
- Los pesos son fijos.
- La salida solo puede ser binaria (0 o 1).

- Para que un sistema como una neurona pueda "aprender", necesita responder al entorno y determinar la relevancia de diferentes entradas basándose en la retroalimentación de experiencias previas.

$$y = f(\sum_{i=1}^{N} W_i I_i)$$

### Organization of Behavior

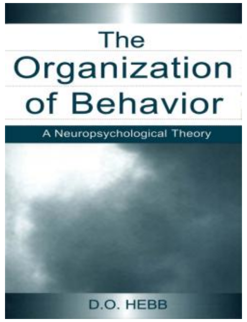
Donald Hebb, 1949 (psicólogo)




- La actividad de células neuronales cercanas tendería a sincronizarse con el tiempo.

- Ley de Heb:
  - Las neuronas que se activan juntas, se conectan juntas.

# Perceptrón




### Organization of Behavior

Donald Hebb, 1949 (psicólogo)

### Limitaciones de los TLUs

- No pueden representar las operaciones XOR y XNOR.
- Los **pesos son fijos**.
- La salida solo puede ser binaria (0 o 1).

- Para que un sistema como una neurona pueda "aprender", necesita responder al entorno y determinar la relevancia de diferentes entradas basándose en la retroalimentación de experiencias previas.

$$y = f\left(\sum_{i=1}^{N} W_i I_i\right)$$

- La actividad de células neuronales cercanas tendería a sincronizarse con el tiempo.

- Ley de Heb:
  - Las neuronas que se activan juntas, se conectan juntas.

### Perceptrón

Frank Rosenblatt, Laboratorio Aeronáutico de Cornell

50s

- Reemplazó los pesos fijos del modelo TLU con **pesos adaptativos** y añadió un término de **sesgo** (bias), dando lugar a una nueva función:

Entradas

$$\hat{y} = f\left(\sum_{i=1}^{n} w_i x_i + b\right)$$

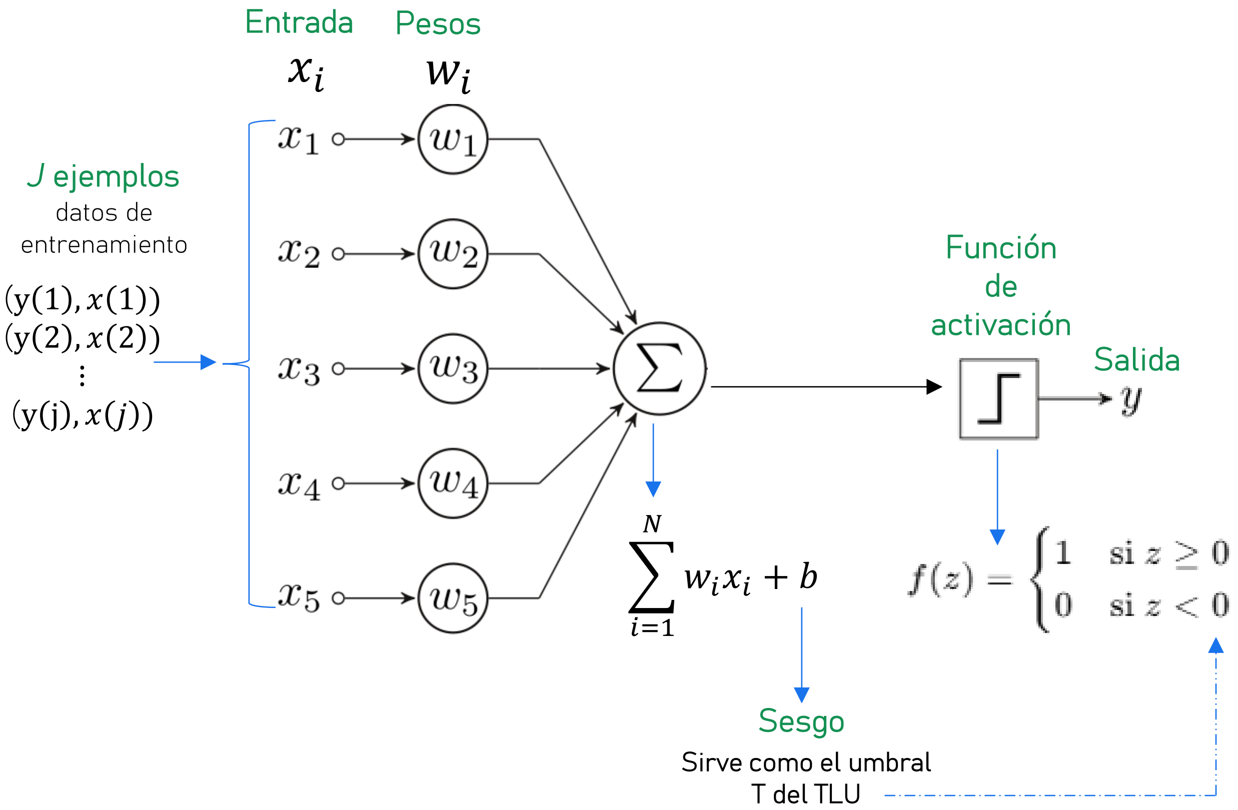
Sesgo

Actúa como un umbral que desplaza la frontera de decisión, sin necesidad de cambiar los pesos.

# Perceptrón

Combinando las observaciones de
Hebb con el modelo TLU

## Ajuste automático de los pesos y sesgo




*i* = índice del número de entrada
*i* = índice del número de entrada

1. Inicia con un **conjunto de *J* ejemplos** *x(1) … x(j)*.
Todos los ejemplos tienen una etiqueta que toma el valor de 0 o 1.
*(y,x)(1) … (y,x)(j)*

2. **Inicializa todos los pesos** y **sesgo** con un valor aleatorio pequeño o con el valor de 0,

3. **Calcula el valor estimado de $\hat{y}$**, para todos los ejemplos X usando la función del perceptrón.

4. **Actualiza los pesos y sesgo** usando una tasa de aprendizaje $r$ con la finalidad de que se aproxime mejor la salida para la entrada a la salida deseada para cada paso de entrenamiento $t$ :

$$w_{i(t+1)} = w_{i(t)} + r\overbrace{(y(j) - \hat{y}(j))}^{\text{Error}}x(j)_i$$

$$b = b + r(y(j) - \hat{y}(j))$$

para los J ejemplos y N características (entradas)

5. **Repite los pasos del 3 al 4** hasta que el error sea menor que un cierto umbral.

# Ejecutando el algoritmo a mano

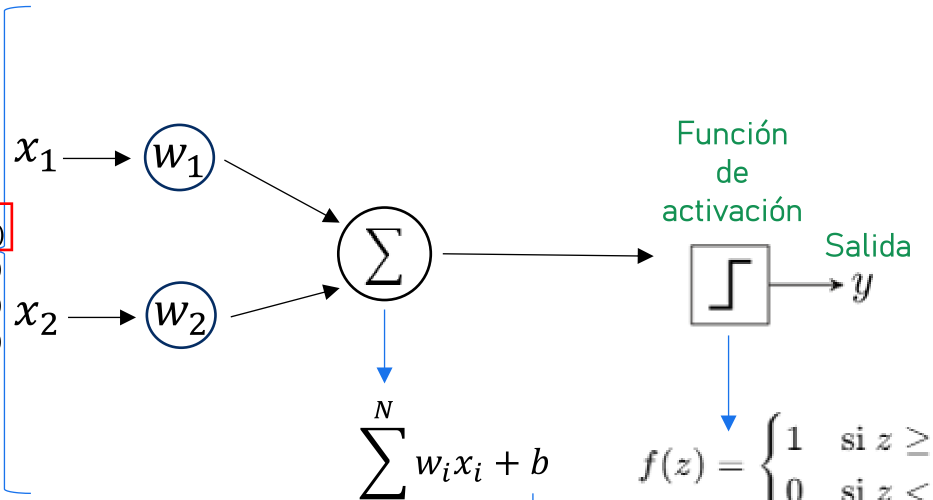
| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

## Ajuste automático de los pesos y sesgo

Entrada $x_i$   Pesos $w_i$

*J* = 4 ejemplos
datos de entrenamiento

$\boxed{(y(1), x(1)) = (0, (0,0))}$
$(y(2), x(2)) = (1, (1,0))$
$(y(3), x(3)) = (1, (0,1))$
$(y(4), x(4)) = (1, (1,1))$




$$\sum_{i=1}^{N} w_i x_i + b$$

$$f(z) = \begin{cases} 1 & \text{si } z \geq 0 \\ 0 & \text{si } z < 0 \end{cases}$$

Sesgo
Sirve como el umbral
T del TLU

*i* = índice del número de entrada
*i* = índice del número de entrada

1. Inicia con un **conjunto de** ***J*** **ejemplos**

*J* = 4

$$\begin{array}{ccccc} & j=1 & j=2 & j=3 & j=4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[0,0], & [1,0], & [0,1], & [1,1]] \\ & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \end{array}$$

$$\begin{array}{ccccc} & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & 1, & 1] \\ & y & y & y & y \end{array}$$

2. **Inicializa todos los pesos** y **sesgo** con un valor aleatorio pequeño o con el valor de 0,

3. **Calcula el valor estimado de** $\hat{y}$ para el valor de j actual

j=1

$t$ =1

4. **Actualiza los pesos** usando una tasa de aprendizaje $r$

$r$ =0.1

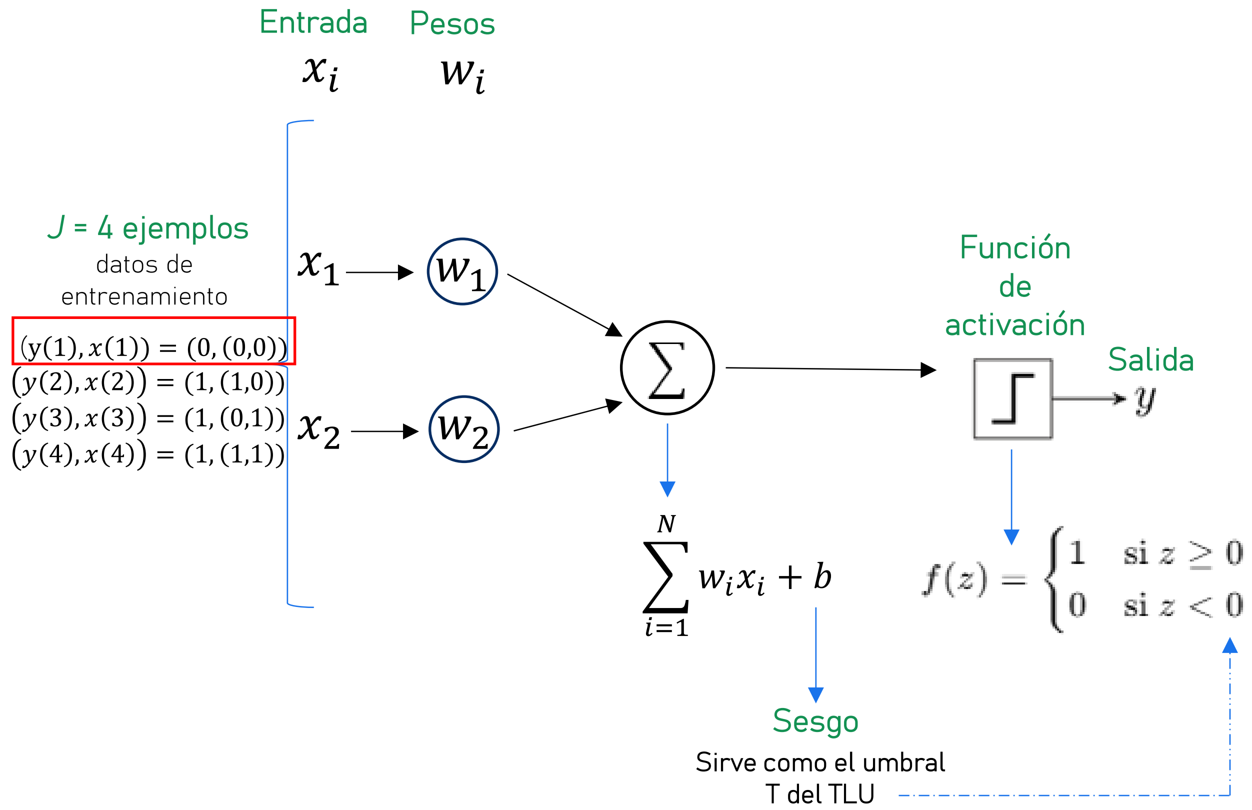
$$w_{i(t+1)} = w_{i(t)} + r\overbrace{(y(j) - \hat{y}(j))}^{\text{Error}}x(j)_i$$

$$b = b + r(y(j) - \hat{y}(j))$$

# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

## Ajuste automático de los pesos y sesgo




$i$ = índice del número de entrada
$i$ = índice del número de entrada

1. Inicia con un conjunto de $J$ ejemplos

$J = 4$

$$\begin{array}{ccccc} & j=1 & j=2 & j=3 & j=4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[0,0], & [1,0], & [0,1], & [1,1]] \\ & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \\ & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & 1, & 1] \\ & y & y & y & y \end{array}$$

2. Inicializa todos los pesos y sesgo con un valor aleatorio pequeño o con el valor de 0,

$w_1 = 0 \quad w_2 = 0 \quad b = 0$

j=1

$t$ =1

3. Calcula el valor estimado de $\hat{y}$ para el valor de j actual

$$\hat{y}(1) = f(w_1x_1 + w_2x_2 + b) = f(0*0 + 0*0 + 0) = f(0) = 1$$

4. Actualiza los pesos usando una tasa de aprendizaje $r$

$r$ =0.1

$$w_{1(t+1)} = w_{1(t)} + r\underbrace{(y(1) - \hat{y}(1))}_{\text{Error}}x(1)_1 = 0 + 0.1\underbrace{(0-1)}_{\text{Error}}0 = 0$$

$$w_{2(t+1)} = w_{2(t)} + r(y(1) - \hat{y}(1))x(1)_2 = 0 + 0.1(0-1)0 = 0$$

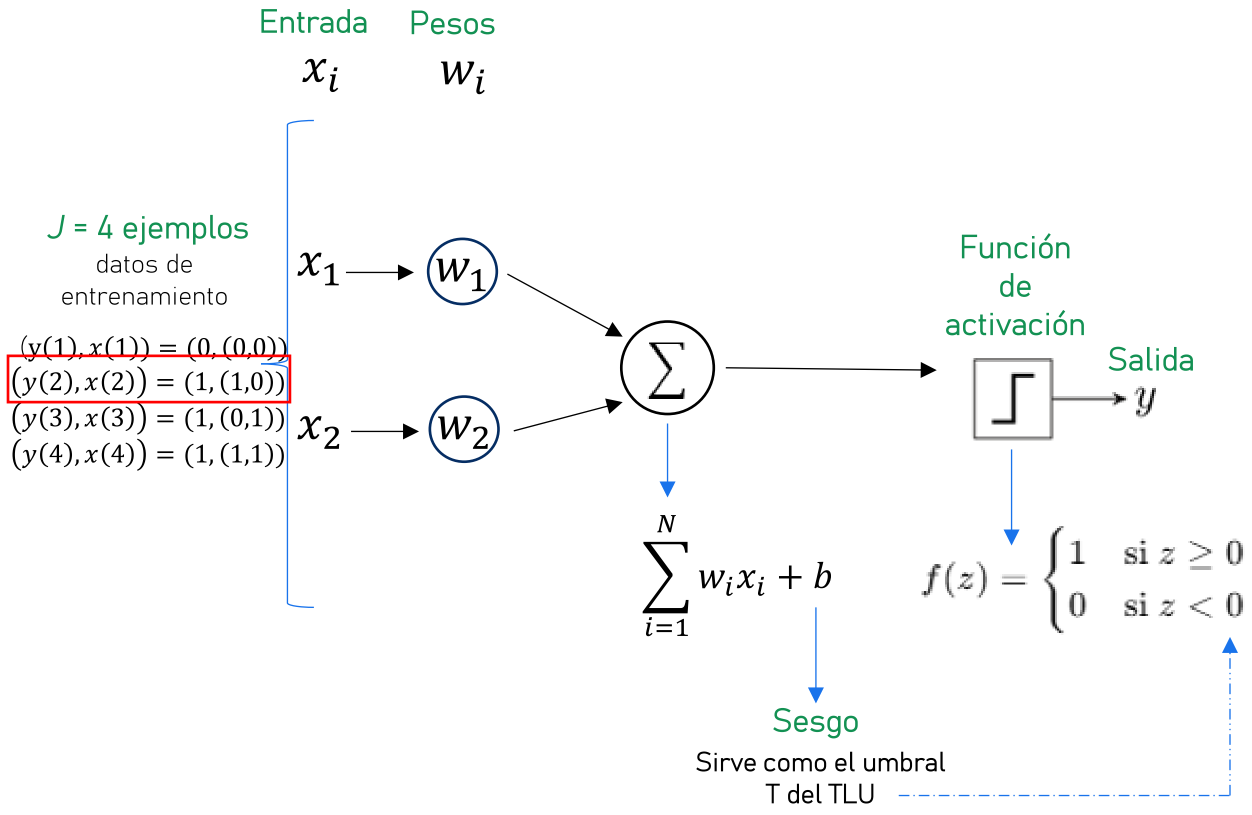
$$b = b + r(y(1) - \hat{y}(1)) = 0 + 0.1(0-1) = -0.1$$

$w_1 = 0 \quad w_2 = 0 \quad b = -0.1$

# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

Ajuste automático de los pesos y sesgo




$$\begin{array}{ccccc} & j = 1 & j = 2 & j = 3 & j = 4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[0,0], & [1,0], & [0,1], & [1,1]] \\ & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \end{array}$$

$$\begin{array}{ccccc} & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & 1, & 1] \\ & y & y & y & y \end{array}$$

$w_1 = 0$ $w_2 = 0$ $b = -0.1$

j=2

$t$ =1

3. Calcula el valor estimado de $\hat{y}$ para el valor de j actual

4. Actualiza los pesos usando una tasa de aprendizaje $r$

$r$ =0.1

$$w_{i(t+1)} = w_{i(t)} + r\underbrace{(y(j) - \hat{y}(j))}_{\text{Error}}x(j)_i$$

$$b = b + r(y(j) - \hat{y}(j))$$

$i$ = índice del número de entrada

$i$ = índice del número de entrada

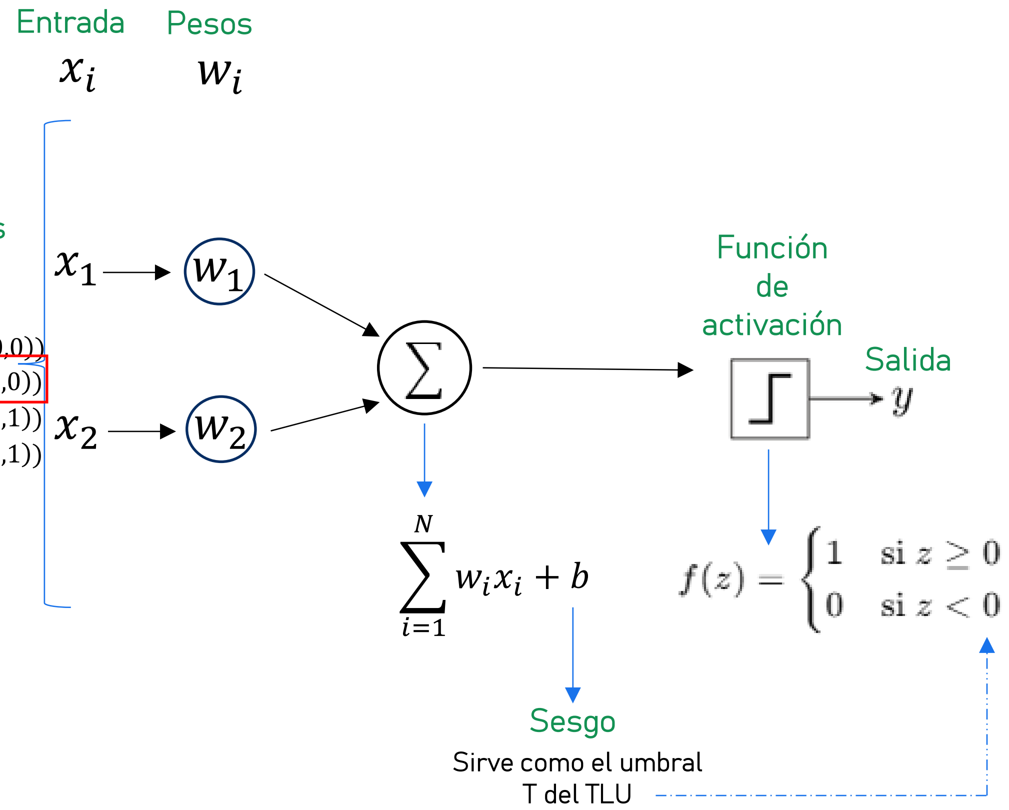
# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

## Ajuste automático de los pesos y sesgo

$J$ = 4 ejemplos
datos de entrenamiento

$(y(1), x(1)) = (0, (0,0))$
$\boxed{(y(2), x(2)) = (1, (1,0))}$
$(y(3), x(3)) = (1, (0,1))$
$(y(4), x(4)) = (1, (1,1))$




$$\begin{array}{ccccc} & j=1 & j=2 & j=3 & j=4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[0,0], & [1,0], & [0,1], & [1,1]] \\ & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \\ & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & 1, & 1] \\ & y & y & y & y \end{array}$$

$w_1 = 0 \quad w_2 = 0 \quad b = -0.1$

j=2

$t$ =1

3. Calcula el valor estimado de $\hat{y}$ para el valor de j actual

$$\hat{y}(2) = f(w_1x_1 + w_2x_2 + b) = f(0*1 + 0*0 - 0.1) = f(-0.1) = 0$$

4. Actualiza los pesos usando una tasa de aprendizaje $r$

$r$ =0.1

$$w_{1(t+1)} = w_{1(t)} + r\underbrace{(y(2) - \hat{y}(2))}_{\text{Error}}x(2)_1 = 0 + 0.1\underbrace{(1-0)}_{\text{Error}}1 = 0.1$$
$$w_{2(t+1)} = w_{2(t)} + r(y(2) - \hat{y}(2))x(2)_2 = 0 + 0.1(1-0)0 = 0$$
$$b = b + r(y(2) - \hat{y}(2)) = -0.1 + 0.1(1-0) = 0$$

$w_1 = 0.1 \quad w_2 = 0 \quad b = 0$

$i$ = índice del número de entrada
$i$ = índice del número de entrada

# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

Ajuste automático de los pesos y sesgo

$J$ = 4

$j$ = 1 $j$ = 2 $j$ = 3 $j$ = 4

$$X = [\underbrace{[0,0]}_{x(1)},\ \underbrace{[1,0]}_{x(2)},\ \underbrace{[0,1]}_{x(3)},\ \underbrace{[1,1]}_{x(4)}]$$

$x_1$ $x_2$ $x_1$ $x_2$ $x_1$ $x_2$ $x_1$ $x_2$

$$Y = [\underbrace{0}_{y(1)},\ \underbrace{1}_{y(2)},\ \underbrace{1}_{y(3)},\ \underbrace{1}_{y(4)}]$$

$y$ $y$ $y$ $y$

Entrada $x_i$ Pesos $w_i$

$J$ = 4 ejemplos
datos de
entrenamiento

$(y(1), x(1)) = (0, (0,0))$
$(y(2), x(2)) = (1, (1,0))$
$\boxed{(y(3), x(3)) = (1, (0,1))}$
$(y(4), x(4)) = (1, (1,1))$

$x_1 \rightarrow w_1$
$x_2 \rightarrow w_2$

$\sum$

$$\sum_{i=1}^{N} w_i x_i + b$$

Función de activación

Salida $y$

$$f(z) = \begin{cases} 1 & \text{si } z \geq 0 \\ 0 & \text{si } z < 0 \end{cases}$$

Sesgo
Sirve como el umbral
T del TLU

$$w_1 = 0.1 \quad w_2 = 0 \quad b = 0$$

j=3

$t$ =1

3. Calcula el valor estimado de $\hat{y}$ para el valor de j actual

4. Actualiza los pesos usando una tasa de aprendizaje
$r$ =0.1

$$w_{i(t+1)} = w_{i(t)} + r\underbrace{(y(j) - \hat{y}(j))}_{\text{Error}}x(j)_i$$

$$b = b + r(y(j) - \hat{y}(j))$$

$i$ = índice del número de entrada
$i$ = índice del número de entrada

# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

## Ajuste automático de los pesos y sesgo

$J$ = 4 ejemplos

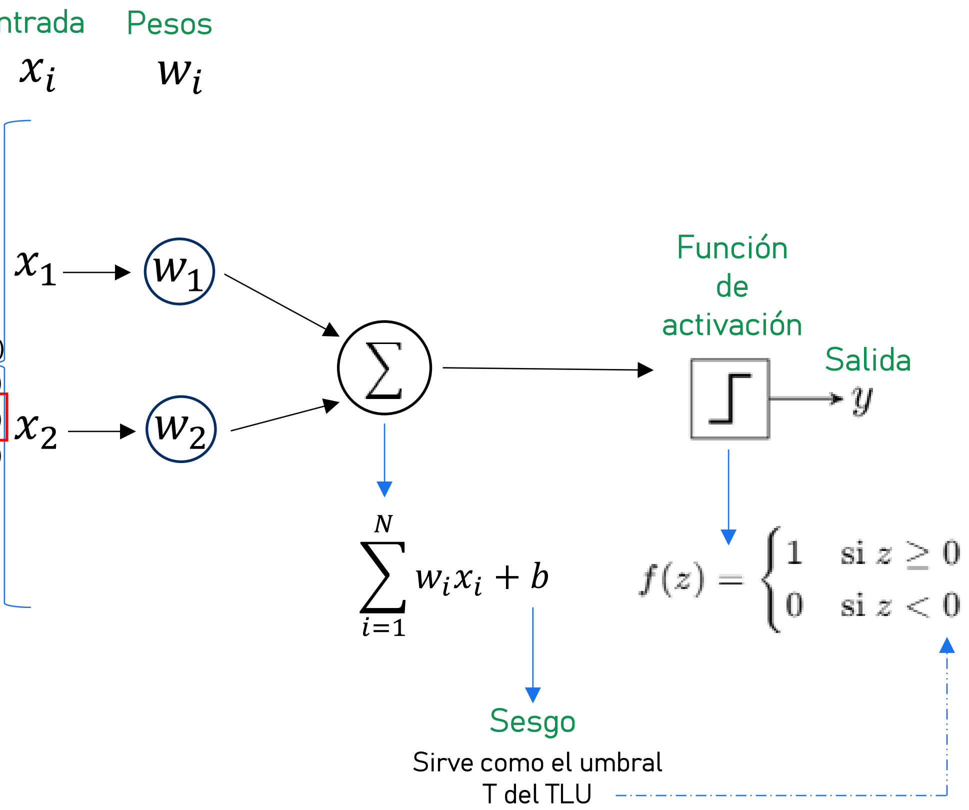
datos de entrenamiento

$(y(1), x(1)) = (0, (0{,}0))$

$(y(2), x(2)) = (1, (1{,}0))$

$\boxed{(y(3), x(3)) = (1, (0{,}1))}$

$(y(4), x(4)) = (1, (1{,}1))$




$J$ = 4

$$\begin{array}{ccccc} & j=1 & j=2 & j=3 & j=4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[\underset{x_1\ x_2}{0{,}0}], & [\underset{x_1\ x_2}{1{,}0}], & [\underset{x_1\ x_2}{0{,}1}], & [\underset{x_1\ x_2}{1{,}1}]] \\ & y(1) & y(2) & y(3) & y(4) \\ Y = & [\underset{y}{0}, & \underset{y}{1}, & \underset{y}{1}, & \underset{y}{1}] \end{array}$$

$w_1 = 0.1 \quad w_2 = 0 \quad b = 0$

3. Calcula el valor estimado de $\hat{y}$ para el valor de j actual

j=3

$t$ =1

$$\hat{y}(3) = f(w_1x_1 + w_2x_2 + b) = f(0.1 * 0 + 0 * 1 + 0) = f(0) = 1$$

4. Actualiza los pesos usando una tasa de aprendizaje $r$

$r$ =0.1

$$w_{1(t+1)} = w_{1(t)} + r\underbrace{(y(3) - \hat{y}(3))}_{\text{Error}}x(3)_1 = 0.1 + 0.1\underbrace{(1-1)}_{\text{Error}}0 = 0.1$$

$$w_{2(t+1)} = w_{2(t)} + r(y(3) - \hat{y}(3))x(3)_2 = 0 + 0.1(1-1)1 = 0$$

$$b = b + r(y(3) - \hat{y}(3)) = 0 + 0.1(1-1) = 0$$

$w_1 = 0.1 \quad w_2 = 0 \quad b = 0$

$i$ = índice del número de entrada

$i$ = índice del número de entrada

# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

Ajuste automático de los pesos y sesgo

$J$ = 4

$$
\begin{array}{ccccc}
 & j=1 & j=2 & j=3 & j=4 \\
 & x(1) & x(2) & x(3) & x(4) \\
X = & [[0,0], & [1,0], & [0,1], & [1,1]] \\
 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2
\end{array}
$$

$$
\begin{array}{ccccc}
 & y(1) & y(2) & y(3) & y(4) \\
Y = & [0, & 1, & 1, & 1] \\
 & y & y & y & y
\end{array}
$$

$J$ = 4 ejemplos

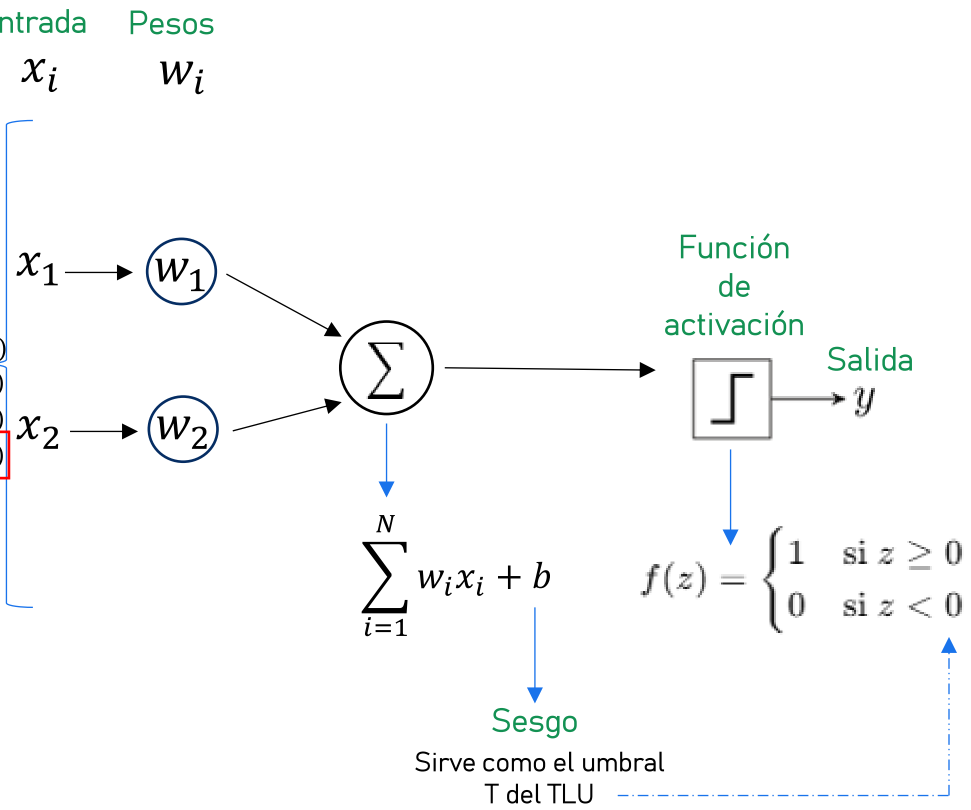
datos de entrenamiento

$(y(1), x(1)) = (0, (0,0))$

$(y(2), x(2)) = (1, (1,0))$

$(y(3), x(3)) = (1, (0,1))$

$\boxed{(y(4), x(4)) = (1, (1,1))}$




$w_1 = 0.1 \quad w_2 = 0 \quad b = 0$

j=4

$t$ =1

3. Calcula el valor estimado de $\hat{y}$ para el valor de j actual

4. Actualiza los pesos usando una tasa de aprendizaje $r$

$r$ =0.1

$$w_{i(t+1)} = w_{i(t)} + r\overbrace{(y(j) - \hat{y}(j))}^{\text{Error}}x(j)_i$$

$$b = b + r(y(j) - \hat{y}(j))$$

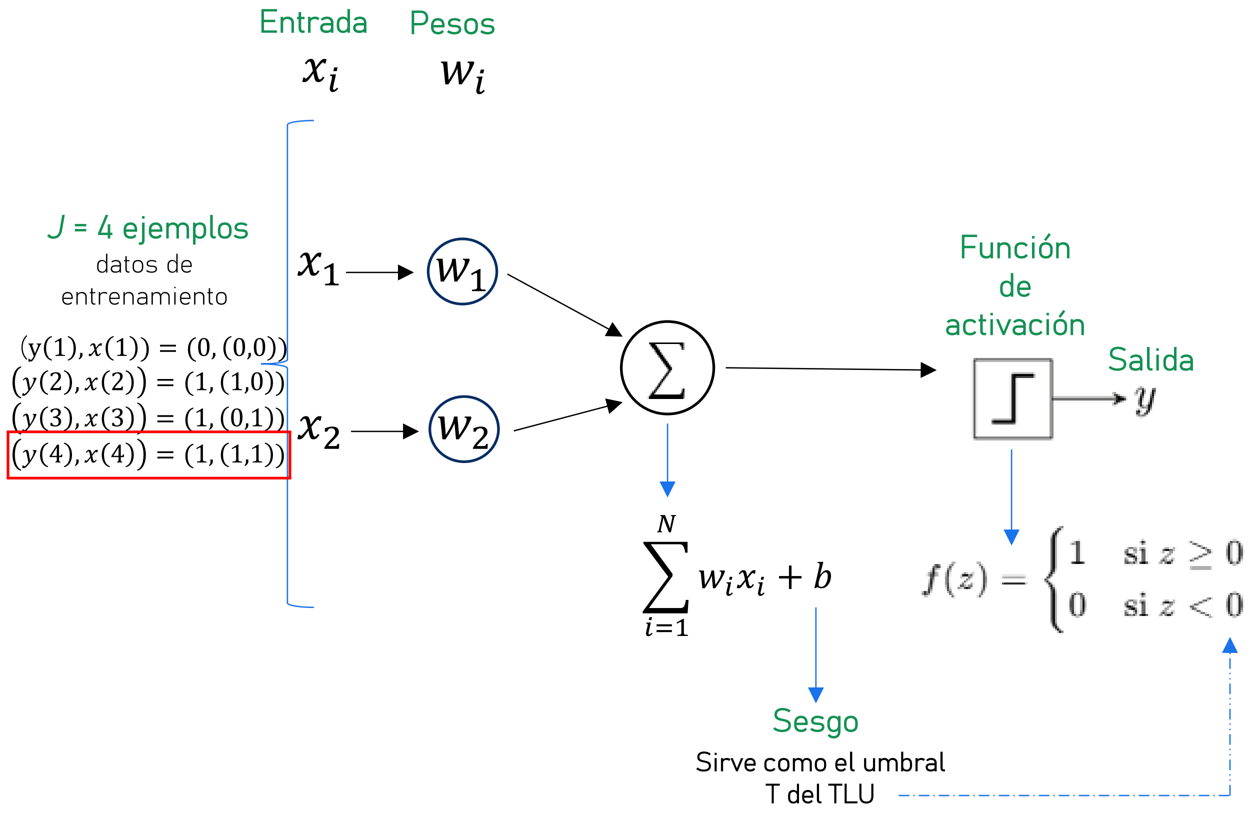
$i$ = índice del número de entrada

$i$ = índice del número de entrada

# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

## Ajuste automático de los pesos y sesgo




$i$ = índice del número de entrada
$i$ = índice del número de entrada

$J$ = 4

$$\begin{array}{ccccc} & j=1 & j=2 & j=3 & j=4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[0,0], & [1,0], & [0,1], & [1,1]] \\ & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \end{array}$$

$$\begin{array}{ccccc} & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & 1, & 1] \\ & y & y & y & y \end{array}$$

$w_1 = 0.1 \quad w_2 = 0 \quad b = 0$

j=4

$t$ =1

3. Calcula el valor estimado de $\hat{y}$ para el valor de j actual

$$\hat{y}(4) = f(w_1x_1 + w_2x_2 + b) = f(0.1 * 1 + 0 * 1 + 0) = f(0.1) = 1$$

4. Actualiza los pesos usando una tasa de aprendizaje $r$

$r$ =0.1

$$w_{1(t+1)} = w_{1(t)} + r\underbrace{(y(4) - \hat{y}(4))}_{\text{Error}}x(4)_1 = 0.1 + 0.1\underbrace{(1-1)}_{\text{Error}}0 = 0.1$$

$$w_{2(t+1)} = w_{2(t)} + r(y(4) - \hat{y}(4))x(4)_2 = 0 + 0.1(1-1)1 = 0$$

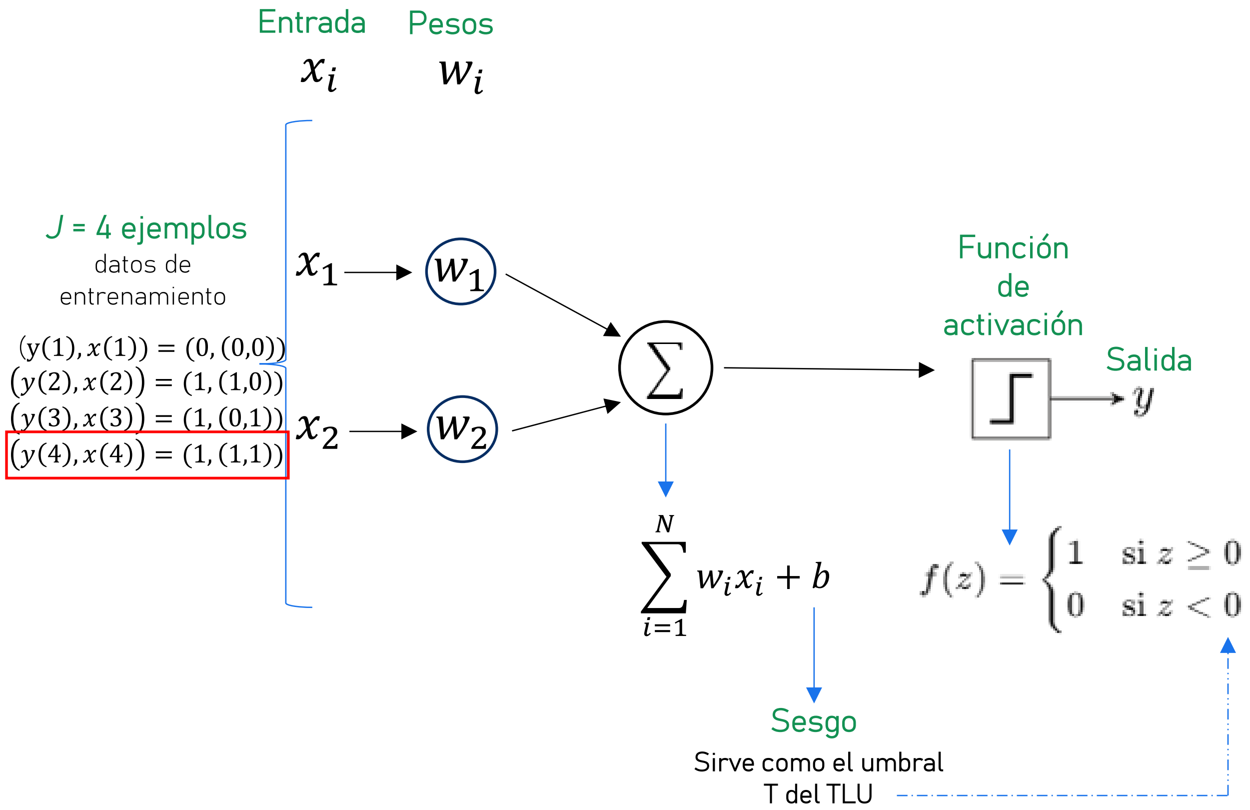
$$b = b + r(y(4) - \hat{y}(4)) = 0 + 0.1(1-1) = 0$$

$w_1 = 0.1 \quad w_2 = 0 \quad b = 0$

# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

Ajuste automático de los pesos y sesgo




$i$ = índice del número de entrada
$i$ = índice del número de entrada

$J$ = 4

$$\begin{array}{ccccc} & j=1 & j=2 & j=3 & j=4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[0,0], & [1,0], & [0,1], & [1,1]] \\ & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \end{array}$$

$$\begin{array}{ccccc} & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & 1, & 1] \\ & y & y & y & y \end{array}$$

$w_1 = 0.1 \quad w_2 = 0 \quad b = 0$

5. Repite los pasos del 3 al 4 hasta que el error sea menor que un cierto umbral.

¿Hubo algún error != 0?

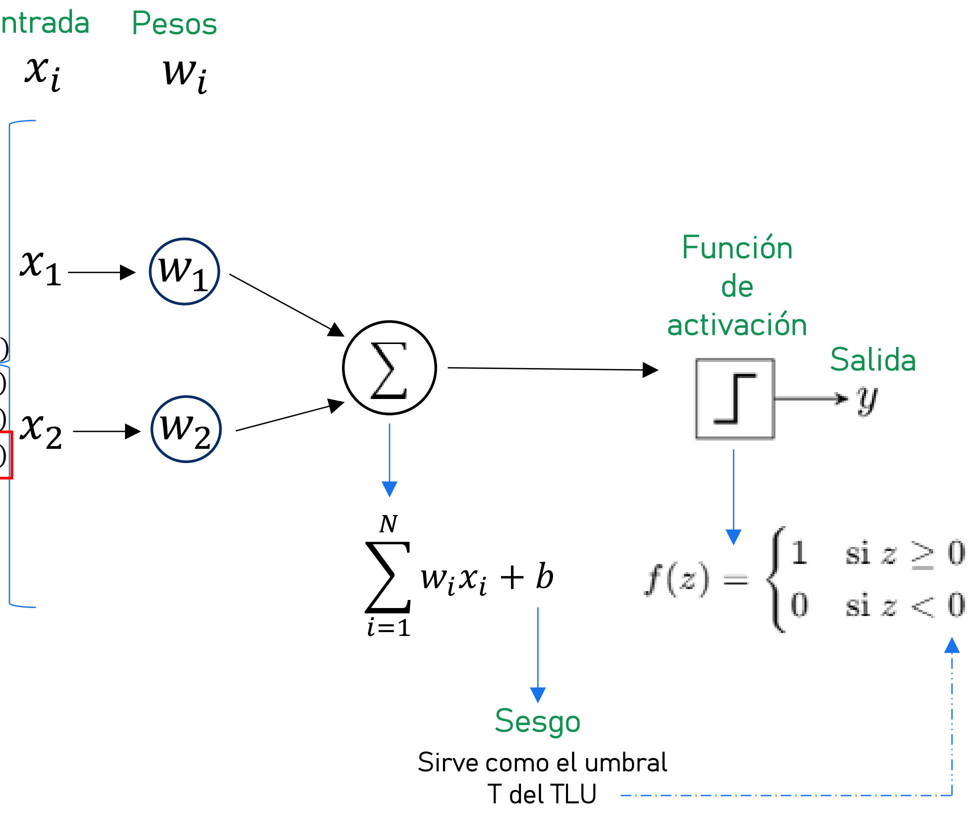
# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

Ajuste automático de los pesos y sesgo

$J$ = 4 ejemplos
datos de
entrenamiento

$(y(1), x(1)) = (0, (0,0))$
$(y(2), x(2)) = (1, (1,0))$
$(y(3), x(3)) = (1, (0,1))$
$\boxed{(y(4), x(4)) = (1, (1,1))}$




$i$ = índice del número de entrada

$i$ = índice del número de entrada

$J$ = 4

| | $j$ = 1 | $j$ = 2 | $j$ = 3 | $j$ = 4 |
|---|---|---|---|---|
| | $x(1)$ | $x(2)$ | $x(3)$ | $x(4)$ |
| $X =$ | $[[0,0],$ | $[1,0],$ | $[0,1],$ | $[1,1]]$ |
| | $x_1\ x_2$ | $x_1\ x_2$ | $x_1\ x_2$ | $x_1\ x_2$ |
| | $y(1)$ | $y(2)$ | $y(3)$ | $y(4)$ |
| $Y =$ | $[0,$ | $1,$ | $1,$ | $1]$ |
| | $y$ | $y$ | $y$ | $y$ |

$w_1 = 0.1 \quad w_2 = 0 \quad b = 0$

5. Repite los pasos del 3 al 4 hasta que el error sea menor que un cierto umbral.

Hubo error != 0 para j=1 y j=2

Volvemos a iterar

# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

Ajuste automático de los pesos y sesgo

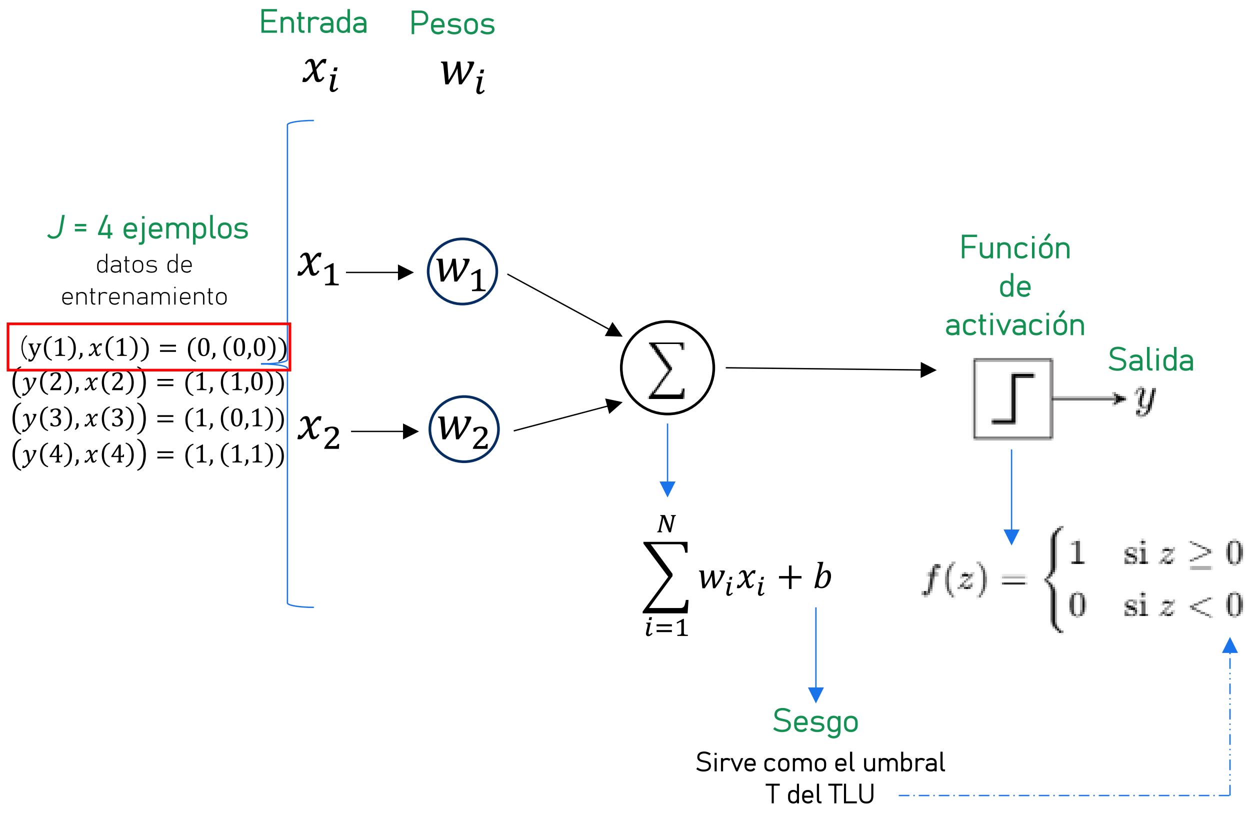
$J = 4$

$$\begin{array}{cccc} j=1 & j=2 & j=3 & j=4 \\ x(1) & x(2) & x(3) & x(4) \end{array}$$

$$X = [[\underset{x_1\ x_2}{0,0}], [\underset{x_1\ x_2}{1,0}], [\underset{x_1\ x_2}{0,1}], [\underset{x_1\ x_2}{1,1}]]$$

$$\begin{array}{cccc} y(1) & y(2) & y(3) & y(4) \end{array}$$

$$Y = [\underset{y}{0}, \underset{y}{1}, \underset{y}{1}, \underset{y}{1}]$$




$w_1 = 0.1$ $w_2 = 0$ $b = 0$

j=1

$t$ =2

3. Calcula el valor estimado de $\hat{y}$ para el valor de j actual

4. Actualiza los pesos usando una tasa de aprendizaje $r$

$r$ =0.1

$$w_{i(t+1)} = w_{i(t)} + r\overbrace{(y(j) - \hat{y}(j))}^{\text{Error}}x(j)_i$$

$$b = b + r(y(j) - \hat{y}(j))$$

$i$ = índice del número de entrada

$i$ = índice del número de entrada

# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

Ajuste automático de los pesos y sesgo

$J$ = 4

$$\begin{array}{ccccc} & j=1 & j=2 & j=3 & j=4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[0,0], & [1,0], & [0,1], & [1,1]] \\ & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \end{array}$$

$$\begin{array}{ccccc} & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & 1, & 1] \\ & y & y & y & y \end{array}$$

$J$ = 4 ejemplos

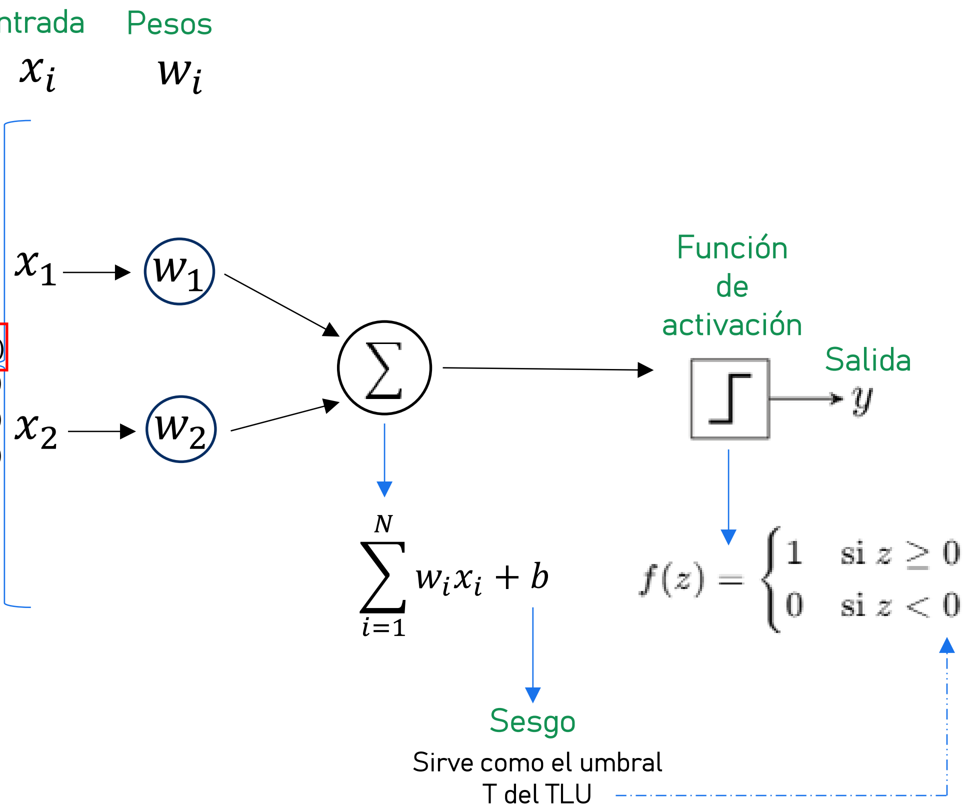
datos de entrenamiento

$\boxed{(y(1), x(1)) = (0, (0,0))}$

$(y(2), x(2)) = (1, (1,0))$

$(y(3), x(3)) = (1, (0,1))$

$(y(4), x(4)) = (1, (1,1))$




$i$ = índice del número de entrada

$i$ = índice del número de entrada

$w_1 = 0.1 \quad w_2 = 0 \quad b = 0$

j=1

$t$ =2

3. Calcula el valor estimado de $\hat{y}$ para el valor de j actual

$$\hat{y}(1) = f(w_1x_1 + w_2x_2 + b) = f(0.1*0 + 0*0 + 0) = f(0) = 1$$

4. Actualiza los pesos usando una tasa de aprendizaje $r$

$r$ =0.1

$$w_{1(t+1)} = w_{1(t)} + r\overbrace{(y(1) - \hat{y}(1))}^{\text{Error}}x(1)_1 = 0.1 + 0.1\overbrace{(0-1)}^{\text{Error}}0 = 0.1$$

$$w_{2(t+1)} = w_{2(t)} + r(y(1) - \hat{y}(1))x(1)_2 = 0 + 0.1(0-1)0 = 0$$

$$b = b + r(y(1) - \hat{y}(1)) = 0 + 0.1(0-1) = -0.1$$

$w_1 = 0.1 \quad w_2 = 0 \quad b = -0.1$

# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

Ajuste automático de los pesos y sesgo

$J$ = 4

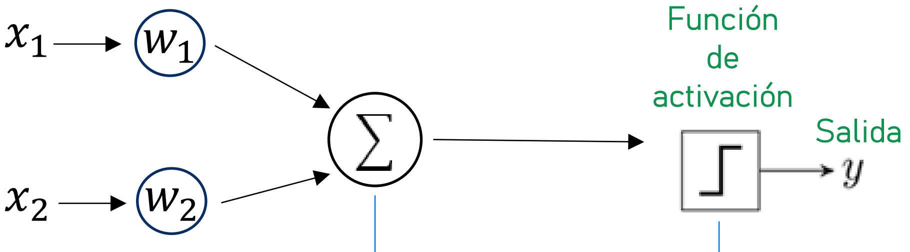
$$\begin{array}{ccccc} & j=1 & j=2 & j=3 & j=4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[0,0], & [1,0], & [0,1], & [1,1]] \\ & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \end{array}$$

$$\begin{array}{ccccc} & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & 1, & 1] \\ & y & y & y & y \end{array}$$

Entrada $x_i$ Pesos $w_i$

$J$ = 4 ejemplos
datos de
entrenamiento

$(y(1), x(1)) = (0, (0,0))$
$(y(2), x(2)) = (1, (1,0))$
$(y(3), x(3)) = (1, (0,1))$
$(y(4), x(4)) = (1, (1,1))$




$$\sum_{i=1}^{N} w_i x_i + b$$

$$f(z) = \begin{cases} 1 & \text{si } z \geq 0 \\ 0 & \text{si } z < 0 \end{cases}$$

Sesgo
Sirve como el umbral
T del TLU

$w_1 = 0.1$ $w_2 = 0$ $b = -0.1$

j=2

$t$ =2

3. Calcula el valor estimado de $\hat{y}$ para el valor de j actual

$\hat{y}(2) = f(w_1 x_1 + w_2 x_2 + b) =$

4. Actualiza los pesos usando una tasa de aprendizaje $r$

$r$ =0.1

$$w_{i(t+1)} = w_{i(t)} + r\overbrace{(y(j) - \hat{y}(j))}^{\text{Error}}x(j)_i$$

$$b = b + r(y(j) - \hat{y}(j))$$

$i$ = índice del número de entrada
$i$ = índice del número de entrada

# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

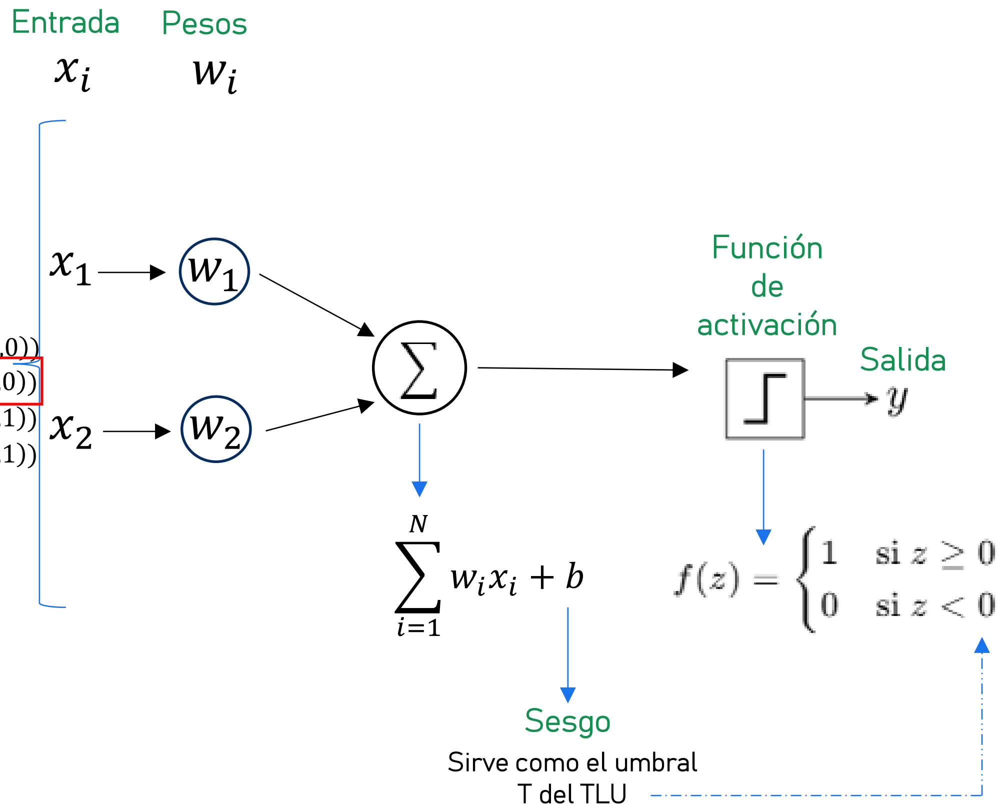
Ajuste automático de los pesos y sesgo

$J = 4$

$$\begin{array}{ccccc} & j=1 & j=2 & j=3 & j=4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[0,0], & [1,0], & [0,1], & [1,1]] \\ & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \\ & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & 1, & 1] \\ & y & y & y & y \end{array}$$

$J$ = 4 ejemplos
datos de
entrenamiento

$(y(1), x(1)) = (0, (0,0))$
$\boxed{(y(2), x(2)) = (1, (1,0))}$
$(y(3), x(3)) = (1, (0,1))$
$(y(4), x(4)) = (1, (1,1))$




$i$ = índice del número de entrada
$i$ = índice del número de entrada

$w_1 = 0.1 \quad w_2 = 0 \quad b = -0.1$

j=2

$t$ =2

3. Calcula el valor estimado de $\hat{y}$ para el valor de j actual

$$\hat{y}(2) = f(w_1x_1 + w_2x_2 + b) = f(0.1 * 1 + 0 * 0 - 0.1) = f(0) = 1$$

4. Actualiza los pesos usando una tasa de aprendizaje $r$

$r$ =0.1

$$w_{1(t+1)} = w_{1(t)} + r\underbrace{(y(2) - \hat{y}(2))}_{\text{Error}}x(2)_1 = 0.1 + 0.1\underbrace{(1-1)}_{\text{Error}}1 = 0.1$$

$$w_{2(t+1)} = w_{2(t)} + r(y(2) - \hat{y}(2))x(2)_2 = 0 + 0.1(1-1)0 = 0$$

$$b = b + r(y(2) - \hat{y}(2)) = -0.1 + 0.1(1-1) = -0.1$$

$w_1 = 0.1 \quad w_2 = 0 \quad b = -0.1$

# Ejecutando el algoritmo a mano

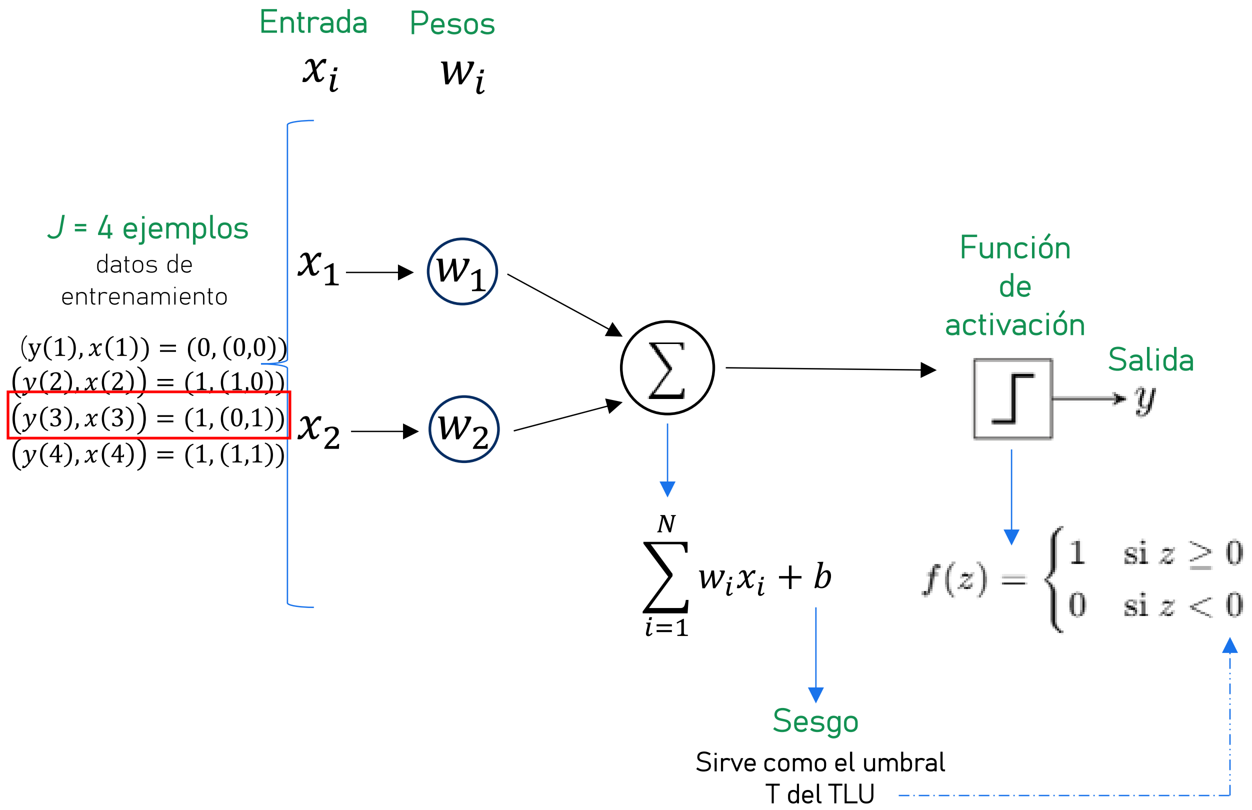
| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

Ajuste automático de los pesos y sesgo

$J = 4$

$$\begin{array}{ccccc} & j=1 & j=2 & j=3 & j=4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[0,0], & [1,0], & [0,1], & [1,1]] \\ & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \end{array}$$

$$\begin{array}{ccccc} & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & 1, & 1] \\ & y & y & y & y \end{array}$$




$w_1 = 0.1 \quad w_2 = 0 \quad b = -0.1$

j=3

$t$ =2

3. Calcula el valor estimado de $\hat{y}$ para el valor de j actual

$$\hat{y}(3) = f(w_1 x_1 + w_2 x_2 + b) =$$

4. Actualiza los pesos usando una tasa de aprendizaje $r$

$r$ =0.1

$$w_{i(t+1)} = w_{i(t)} + r\underbrace{(y(j) - \hat{y}(j))}_{\text{Error}}x(j)_i$$

$$b = b + r(y(j) - \hat{y}(j))$$

$i$ = índice del número de entrada

$i$ = índice del número de entrada

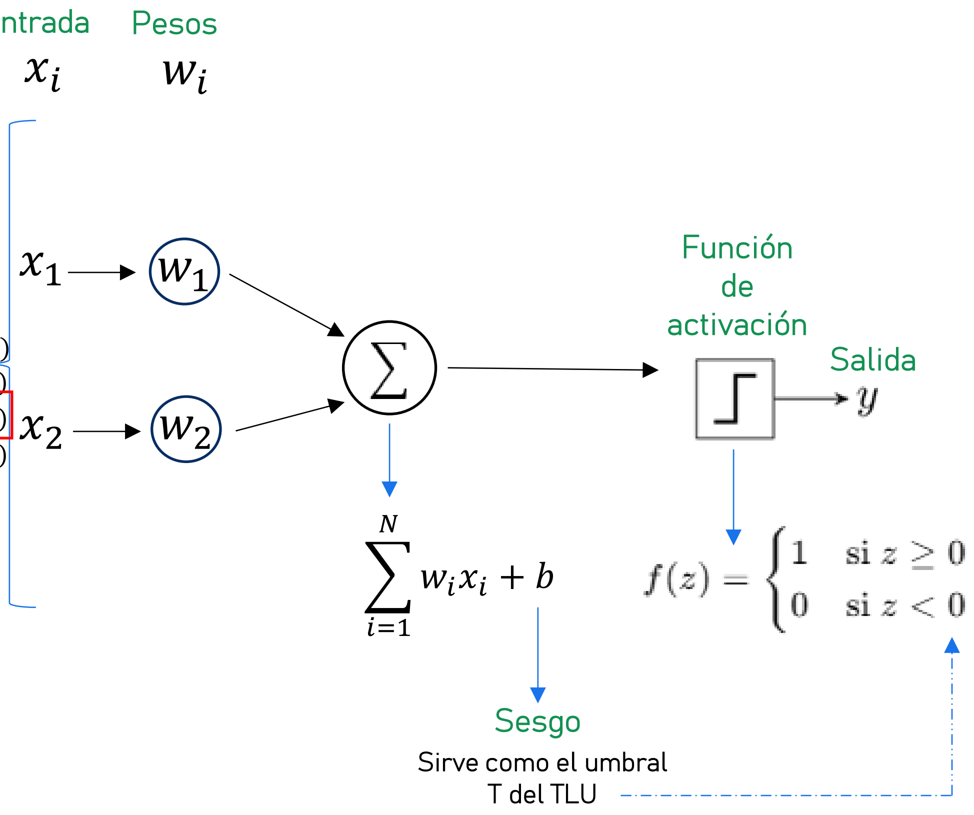
# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

## Ajuste automático de los pesos y sesgo

$J$ = 4 ejemplos
datos de entrenamiento

$(y(1), x(1)) = (0, (0{,}0))$
$(y(2), x(2)) = (1, (1{,}0))$
$\boxed{(y(3), x(3)) = (1, (0{,}1))}$
$(y(4), x(4)) = (1, (1{,}1))$




$i$ = índice del número de entrada
$i$ = índice del número de entrada

$J$ = 4

$$\begin{array}{ccccc} & j=1 & j=2 & j=3 & j=4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[0{,}0], & [1{,}0], & [0{,}1], & [1{,}1]] \\ & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \\ & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & 1, & 1] \\ & y & y & y & y \end{array}$$

$w_1 = 0.1 \quad w_2 = 0 \quad b = -0.1$

j=3

$t$ =2

3. Calcula el valor estimado de $\hat{y}$ para el valor de j actual

$$\hat{y}(3) = f(w_1x_1 + w_2x_2 + b) = f(0.1 * 0 + 0 * 1 - 0.1) = f(-0.1) = 0$$

4. Actualiza los pesos usando una tasa de aprendizaje $r$

$r$ =0.1

$$w_{1(t+1)} = w_{1(t)} + r\underbrace{(y(3) - \hat{y}(3))}_{\text{Error}}x(3)_1 = 0.1 + 0.1\underbrace{(1-0)}_{\text{Error}}0 = 0.1$$

$$w_{2(t+1)} = w_{2(t)} + r(y(3) - \hat{y}(3))x(3)_2 = 0 + 0.1(1-0)1 = 0.1$$

$$b = b + r(y(3) - \hat{y}(3)) = -0.1 + 0.1(1-0) = 0$$

$w_1 = 0.1 \quad w_2 = 0.1 \quad b = 0$

# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

Ajuste automático de los pesos y sesgo

$J$ = 4

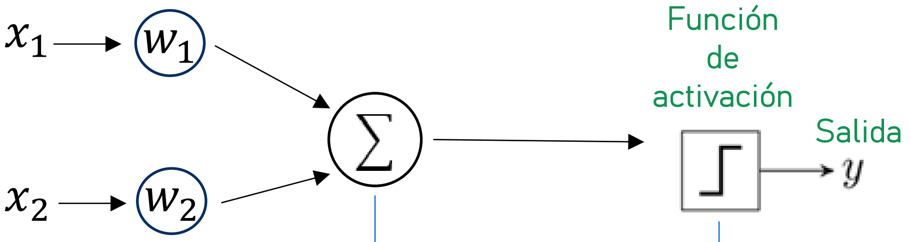
$$\begin{array}{ccccc} & j=1 & j=2 & j=3 & j=4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[0,0], & [1,0], & [0,1], & [1,1]] \\ & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \end{array}$$

$$\begin{array}{ccccc} & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & 1, & 1] \\ & y & y & y & y \end{array}$$

Entrada $x_i$ Pesos $w_i$

$J$ = 4 ejemplos
datos de entrenamiento

$(y(1), x(1)) = (0, (0{,}0))$
$(y(2), x(2)) = (1, (1{,}0))$
$(y(3), x(3)) = (1, (0{,}1))$
$\boxed{(y(4), x(4)) = (1, (1{,}1))}$




$$\sum_{i=1}^{N} w_i x_i + b$$

$$f(z) = \begin{cases} 1 & \text{si } z \geq 0 \\ 0 & \text{si } z < 0 \end{cases}$$

Sesgo
Sirve como el umbral T del TLU

$i$ = índice del número de entrada
$i$ = índice del número de entrada

$w_1 = 0.1 \quad w_2 = 0.1 \quad b = 0.$

j=4

$t$ =2

3. Calcula el valor estimado de $\hat{y}$ para el valor de j actual

$\hat{y}(4) = f(w_1 x_1 + w_2 x_2 + b) =$

4. Actualiza los pesos usando una tasa de aprendizaje $r$

$r$ =0.1

$$w_{i(t+1)} = w_{i(t)} + r\overbrace{(y(j) - \hat{y}(j))}^{\text{Error}}x(j)_i$$

$$b = b + r(y(j) - \hat{y}(j))$$

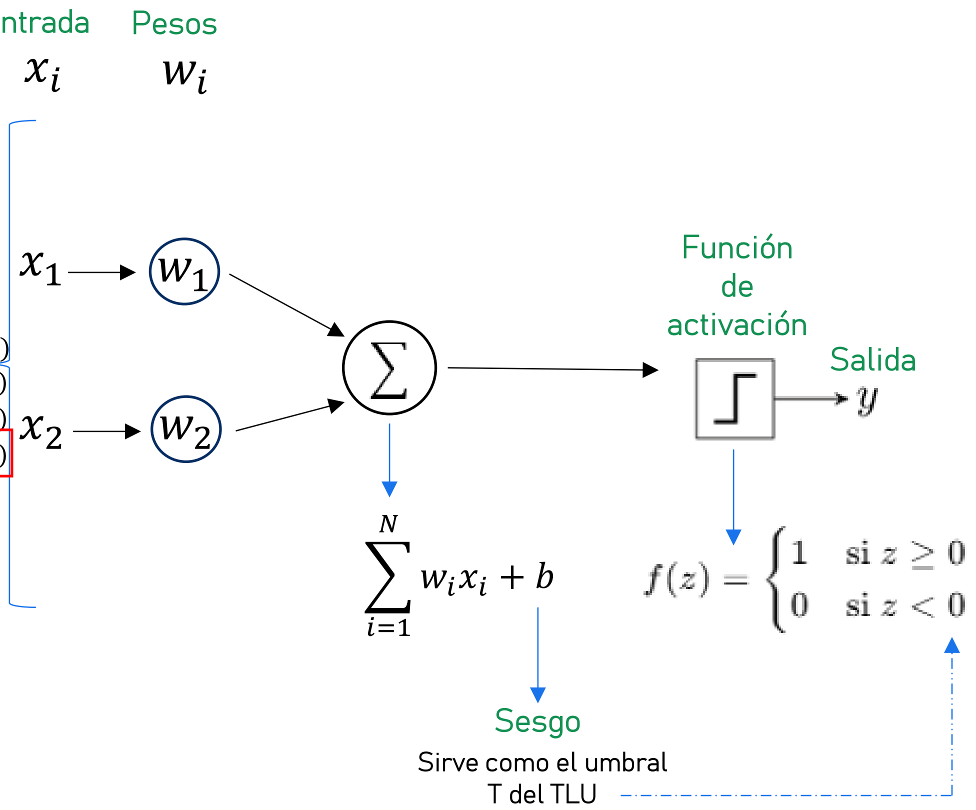
# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

Ajuste automático de los pesos y sesgo

$J$ = 4 ejemplos
datos de entrenamiento

$(y(1), x(1)) = (0, (0{,}0))$
$(y(2), x(2)) = (1, (1{,}0))$
$(y(3), x(3)) = (1, (0{,}1))$
$\boxed{(y(4), x(4)) = (1, (1{,}1))}$




$i$ = índice del número de entrada
$i$ = índice del número de entrada

$J$ = 4

$$\begin{array}{ccccc} & j=1 & j=2 & j=3 & j=4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[0{,}0], & [1{,}0], & [0{,}1], & [1{,}1]] \\ & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \end{array}$$

$$\begin{array}{ccccc} & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & 1, & 1] \\ & y & y & y & y \end{array}$$

$w_1 = 0.1 \quad w_2 = 0.1 \quad b = 0.$

j=4

$t$ =2

3. Calcula el valor estimado de $\hat{y}$ para el valor de j actual

$$\hat{y}(4) = f(w_1x_1 + w_2x_2 + b) = f(0.1 * 1 + 0.1 * 1 + 0) = f(0.2) = 1$$

4. Actualiza los pesos usando una tasa de aprendizaje $r$

$r$ =0.1

$$w_{1(t+1)} = w_{1(t)} + r\underbrace{(y(4) - \hat{y}(4))}_{\text{Error}}x(4)_1 = 0.1 + 0.1\underbrace{(1-1)}_{\text{Error}}1 = 0.1$$

$$w_{2(t+1)} = w_{2(t)} + r(y(4) - \hat{y}(4))x(4)_2 = 0.1 + 0.1(1-1)1 = 0.1$$

$$b = b + r(y(4) - \hat{y}(4)) = 0 + 0.1(1-1) = 0$$

$w_1 = 0.1 \quad w_2 = 0.1 \quad b = 0$

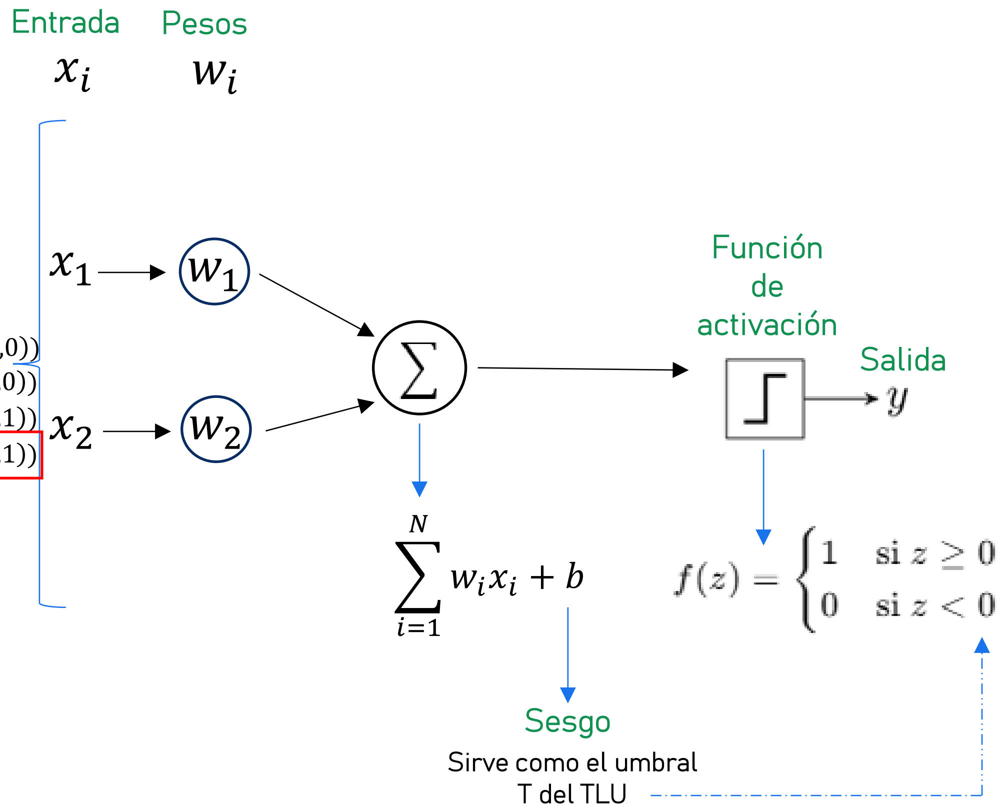
# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

## Ajuste automático de los pesos y sesgo

$J$ = 4 ejemplos
datos de entrenamiento

$(y(1), x(1)) = (0, (0{,}0))$
$(y(2), x(2)) = (1, (1{,}0))$
$(y(3), x(3)) = (1, (0{,}1))$
$\boxed{(y(4), x(4)) = (1, (1{,}1))}$




$i$ = índice del número de entrada
$i$ = índice del número de entrada

$J = 4$

$$\begin{array}{ccccc} & j=1 & j=2 & j=3 & j=4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[0{,}0], & [1{,}0], & [0{,}1], & [1{,}1]] \\ & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \\ & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & 1, & 1] \\ & y & y & y & y \end{array}$$

$w_1 = 0.1 \quad w_2 = 0.1 \quad b = 0$

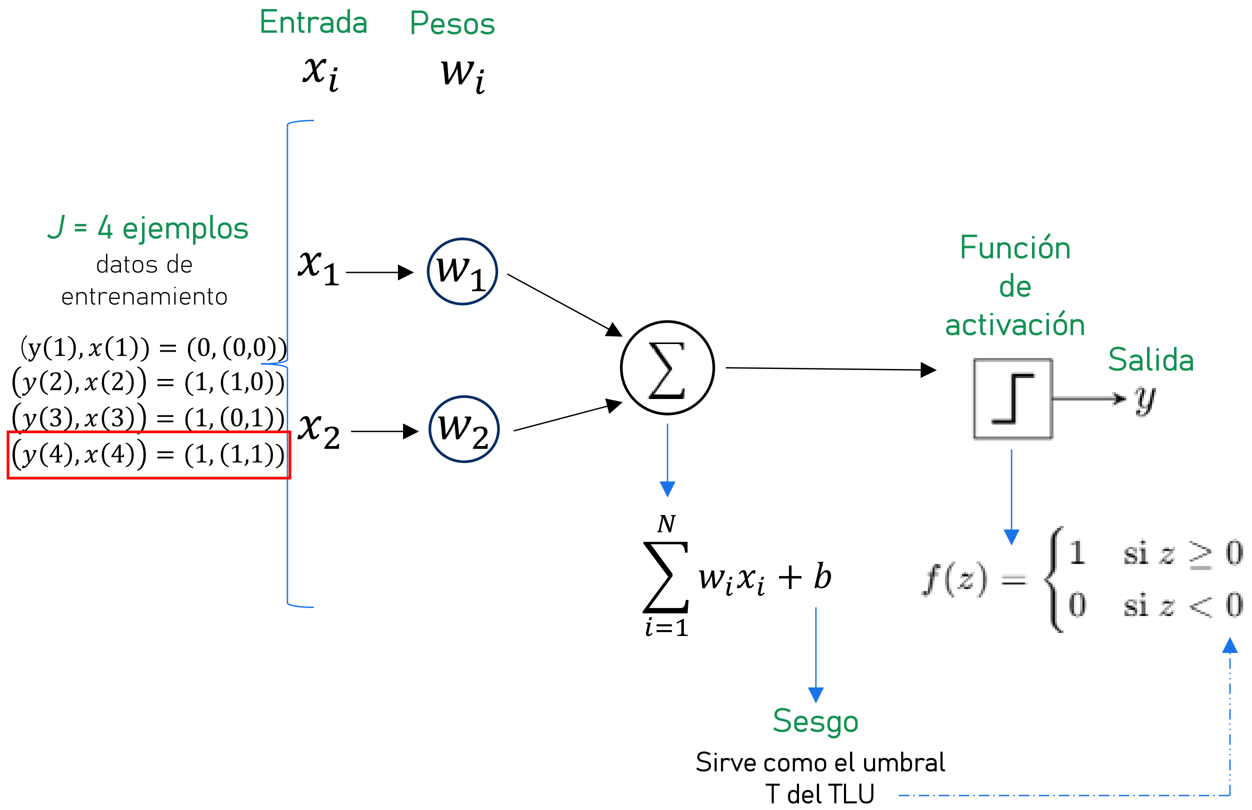
5. Repite los pasos del 3 al 4 hasta que el error sea menor que un cierto umbral.

¿Hubo error != 0?

# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

## Ajuste automático de los pesos y sesgo




$i$ = índice del número de entrada

$i$ = índice del número de entrada

$J$ = 4

$$\begin{array}{ccccc} & j=1 & j=2 & j=3 & j=4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[0,0], & [1,0], & [0,1], & [1,1]] \\ & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \end{array}$$

$$\begin{array}{ccccc} & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & 1, & 1] \\ & y & y & y & y \end{array}$$

$w_1 = 0.1 \quad w_2 = 0.1 \quad b = 0$

5. Repite los pasos del 3 al 4 hasta que el error sea menor que un cierto umbral.

Hubo error != 0 para j=1 y j=3

Volvemos a iterar

# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

Ajuste automático de los pesos y sesgo

$J$ = 4

$$\begin{array}{ccccc} & j=1 & j=2 & j=3 & j=4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[0,0], & [1,0], & [0,1], & [1,1]] \\ & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \end{array}$$

$$\begin{array}{ccccc} & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & 1, & 1] \\ & y & y & y & y \end{array}$$

Entrada $x_i$ Pesos $w_i$

$J$ = 4 ejemplos
datos de
entrenamiento

$(y(1), x(1)) = (0, (0,0))$
$(y(2), x(2)) = (1, (1,0))$
$(y(3), x(3)) = (1, (0,1))$
$(y(4), x(4)) = (1, (1,1))$

$x_1 \rightarrow w_1$, $x_2 \rightarrow w_2$ $\rightarrow \sum$

Función de activación

Salida $y$

$$\sum_{i=1}^{N} w_i x_i + b$$

$$f(z) = \begin{cases} 1 & \text{si } z \geq 0 \\ 0 & \text{si } z < 0 \end{cases}$$

Sesgo
Sirve como el umbral
T del TLU

$i$ = índice del número de entrada
$i$ = índice del número de entrada

$w_1 =$ $w_2 =$ $b =$

j=1

$t$ =3

3. Calcula el valor estimado de $\hat{y}$ para el valor de j actual

$\hat{y}(1) =$

4. Actualiza los pesos usando una tasa de aprendizaje $r$

$r$ =0.1

$w_{1(t+1)} =$

$b =$

$w_{2(t+1)} =$

# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

Ajuste automático de los pesos y sesgo

$J$ = 4

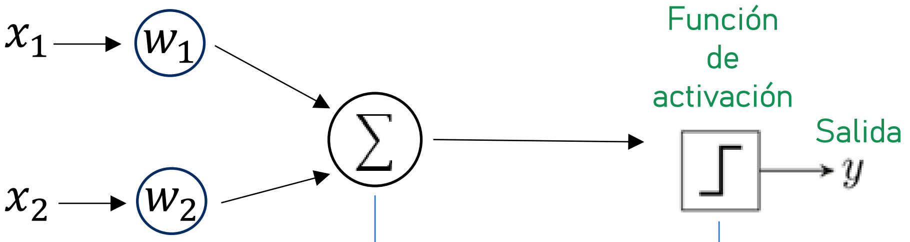
$$\begin{array}{ccccc} & j=1 & j=2 & j=3 & j=4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[0,0], & [1,0], & [0,1], & [1,1]] \\ & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \end{array}$$

$$\begin{array}{ccccc} & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & 1, & 1] \\ & y & y & y & y \end{array}$$

Entrada $x_i$ Pesos $w_i$

$J$ = 4 ejemplos
datos de entrenamiento

$(y(1), x(1)) = (0, (0,0))$
$(y(2), x(2)) = (1, (1,0))$
$(y(3), x(3)) = (1, (0,1))$
$(y(4), x(4)) = (1, (1,1))$




$$\sum_{i=1}^{N} w_i x_i + b$$

$$f(z) = \begin{cases} 1 & \text{si } z \geq 0 \\ 0 & \text{si } z < 0 \end{cases}$$

Sesgo
Sirve como el umbral T del TLU

$w_1 =$ $w_2 =$ $b =$

j=2

$t$ =3

3. Calcula el valor estimado de $\hat{y}$ para el valor de j actual

$\hat{y}(1) =$

4. Actualiza los pesos usando una tasa de aprendizaje $r$

$r$ =0.1

$w_{1(t+1)} =$
$w_{2(t+1)} =$
$b =$

$i$ = índice del número de entrada
$i$ = índice del número de entrada

# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

## Ajuste automático de los pesos y sesgo

$J$ = 4

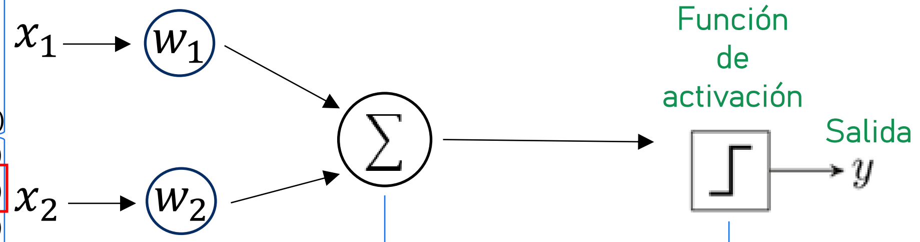
$$\begin{array}{ccccc} & j=1 & j=2 & j=3 & j=4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[0,0], & [1,0], & [0,1], & [1,1]] \\ & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \end{array}$$

$$\begin{array}{ccccc} & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & 1, & 1] \\ & y & y & y & y \end{array}$$

Entrada $x_i$ Pesos $w_i$

$J$ = 4 ejemplos
datos de
entrenamiento

$(y(1), x(1)) = (0, (0,0))$
$(y(2), x(2)) = (1, (1,0))$
$\boxed{(y(3), x(3)) = (1, (0,1))}$
$(y(4), x(4)) = (1, (1,1))$




$$\sum_{i=1}^{N} w_i x_i + b$$

$$f(z) = \begin{cases} 1 & \text{si } z \geq 0 \\ 0 & \text{si } z < 0 \end{cases}$$

Sesgo
Sirve como el umbral
T del TLU

$i$ = índice del número de entrada
$i$ = índice del número de entrada

$w_1 =$ $w_2 =$ $b =$

j=3

$t$ =3

3. Calcula el valor estimado de $\hat{y}$ para el valor de j actual

$\hat{y}(1) =$

4. Actualiza los pesos usando una tasa de aprendizaje $r$

$r$ =0.1

$w_{1(t+1)} =$
$w_{2(t+1)} =$
$b =$

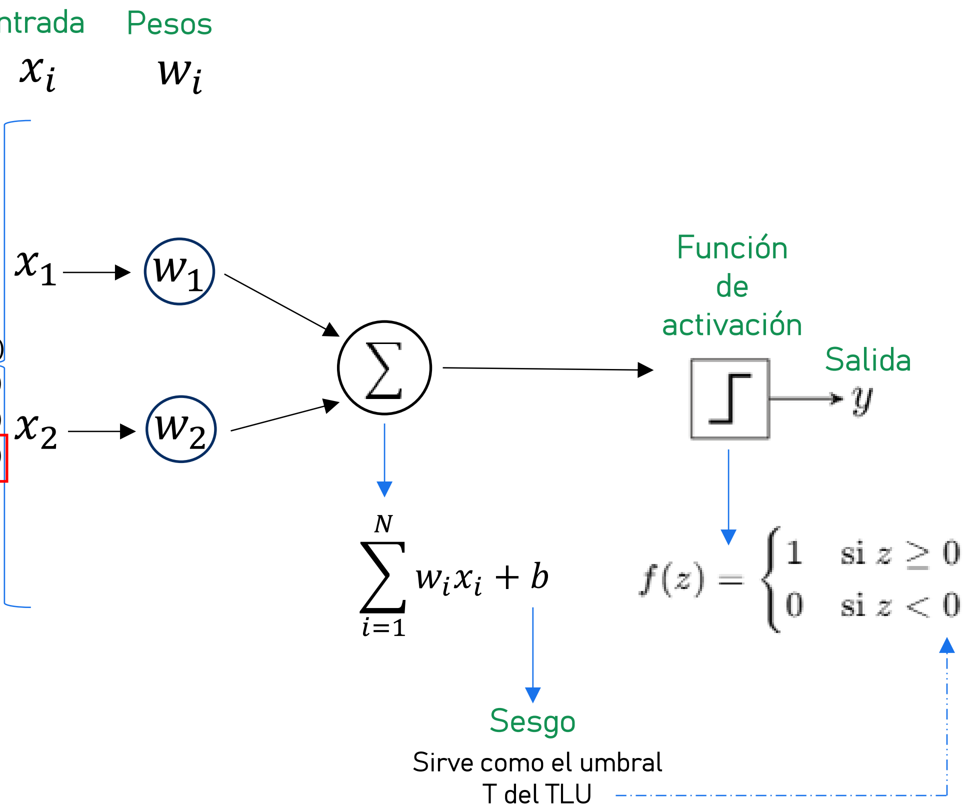
# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

## Ajuste automático de los pesos y sesgo

*J* = 4 ejemplos
datos de
entrenamiento

$(y(1), x(1)) = (0, (0{,}0))$
$(y(2), x(2)) = (1, (1{,}0))$
$(y(3), x(3)) = (1, (0{,}1))$
$\boxed{(y(4), x(4)) = (1, (1{,}1))}$




*J* = 4

$$\begin{array}{cccccc} & & j=1 & j=2 & j=3 & j=4 \\ & & x(1) & x(2) & x(3) & x(4) \\ X & = & [[0{,}0], & [1{,}0], & [0{,}1], & [1{,}1]] \\ & & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \end{array}$$

$$\begin{array}{cccccc} & & y(1) & y(2) & y(3) & y(4) \\ Y & = & [0, & 1, & 1, & 1] \\ & & y & y & y & y \end{array}$$

$w_1 =$ $w_2 =$ $b =$

3. Calcula el valor estimado de $\hat{y}$ para el valor de j actual

j=4

$t$ =3

$\hat{y}(1) =$

4. Actualiza los pesos usando una tasa de aprendizaje $r$

$r$ =0.1

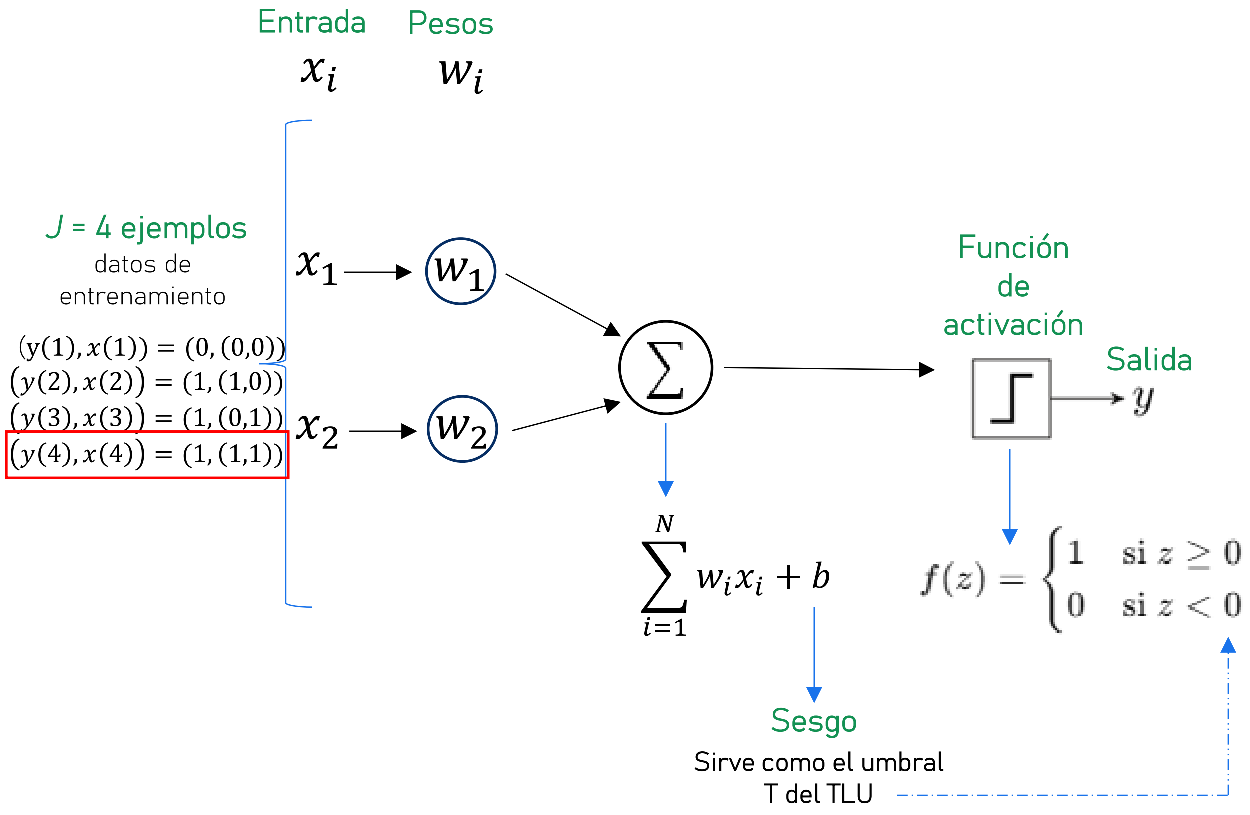
$w_{1(t+1)} =$
$w_{2(t+1)} =$
$b =$

$i$ = índice del número de entrada
$i$ = índice del número de entrada

# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

Ajuste automático de los pesos y sesgo




$i$ = índice del número de entrada

$i$ = índice del número de entrada

$J$ = 4

$$
\begin{array}{ccccc}
 & j=1 & j=2 & j=3 & j=4 \\
 & x(1) & x(2) & x(3) & x(4) \\
X = & [[0,0], & [1,0], & [0,1], & [1,1]] \\
 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \\
 & y(1) & y(2) & y(3) & y(4) \\
Y = & [0, & 1, & 1, & 1] \\
 & y & y & y & y
\end{array}
$$

$w_1 =$ $w_2 =$ $b =$

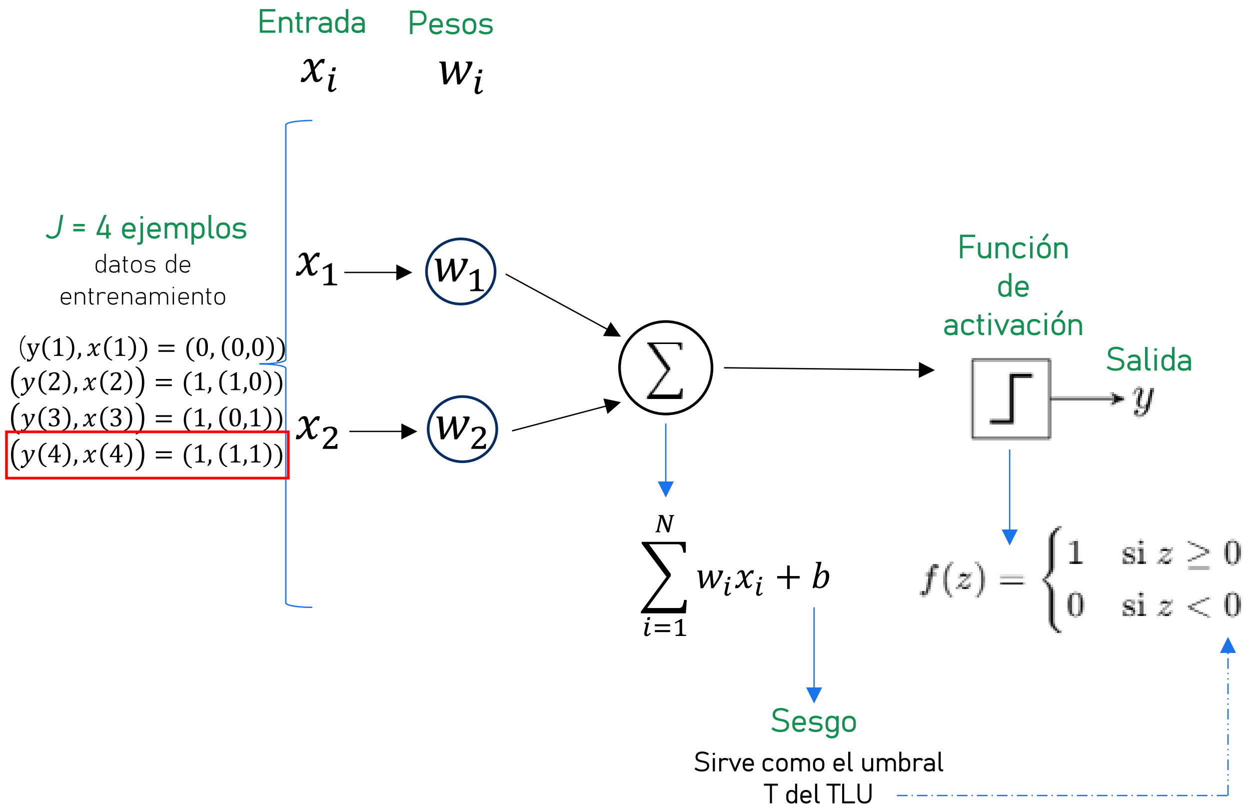
5. Repite los pasos del 3 al 4 hasta que el error sea menor que un cierto umbral.

Hubo error != 0 para

# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

## Ajuste automático de los pesos y sesgo




$i$ = índice del número de entrada

$i$ = índice del número de entrada

$J$ = 4

$$\begin{array}{ccccc} & j=1 & j=2 & j=3 & j=4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[0{,}0], & [1{,}0], & [0{,}1], & [1{,}1]] \\ & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \end{array}$$

$$\begin{array}{ccccc} & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & 1, & 1] \\ & y & y & y & y \end{array}$$

$w_1 =$ $w_2 =$ $b =$

5. Repite los pasos del 3 al 4 hasta que el error sea menor que un cierto umbral.

Hubo error != 0 para

Volvemos a iterar

# Ejecutando el algoritmo a mano

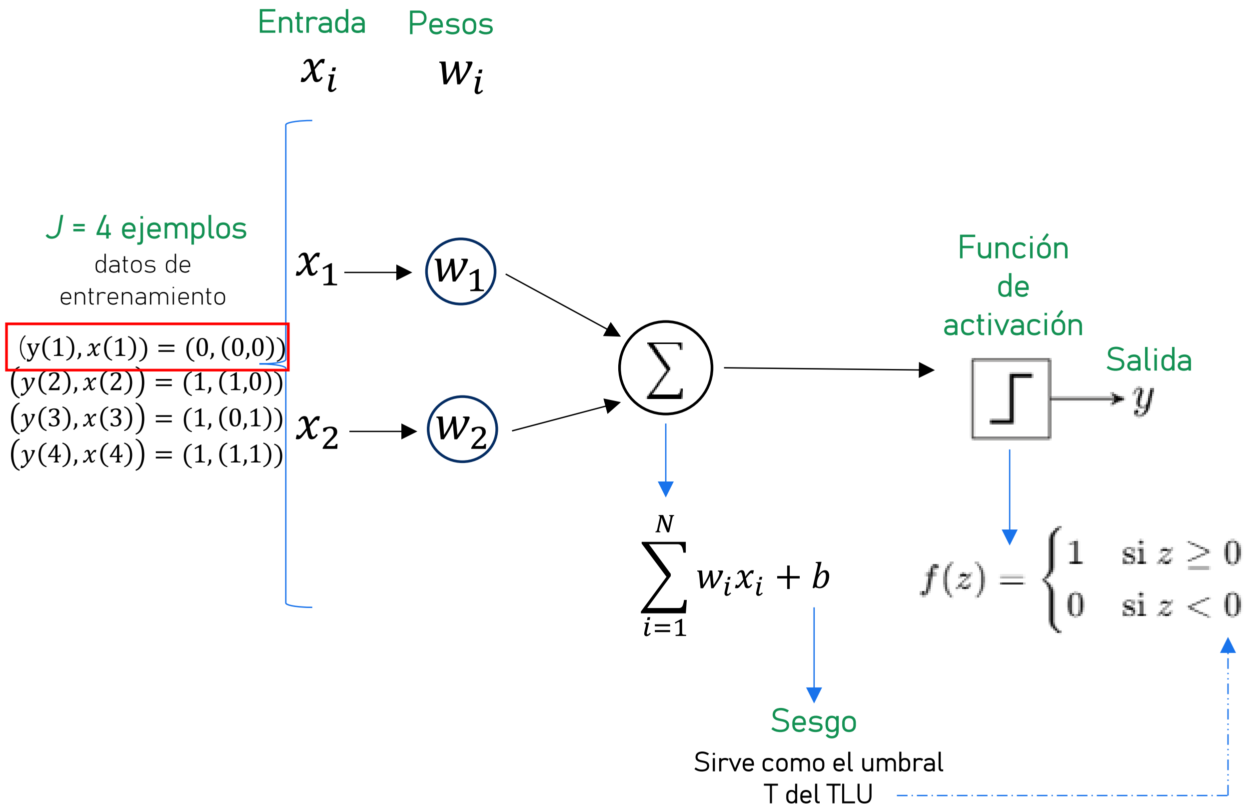
| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

Ajuste automático de los pesos y sesgo

$J = 4$

$$\begin{array}{ccccc} & j=1 & j=2 & j=3 & j=4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[0,0], & [1,0], & [0,1], & [1,1]] \\ & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \end{array}$$

$$\begin{array}{ccccc} & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & 1, & 1] \\ & y & y & y & y \end{array}$$




$w_1 =$ $w_2 =$ $b =$

j=1

$t$ =4

3. Calcula el valor estimado de $\hat{y}$ para el valor de j actual

$\hat{y}(1) =$

4. Actualiza los pesos usando una tasa de aprendizaje $r$

$r$ =0.1

$w_{1(t+1)} =$

$w_{2(t+1)} =$

$b =$

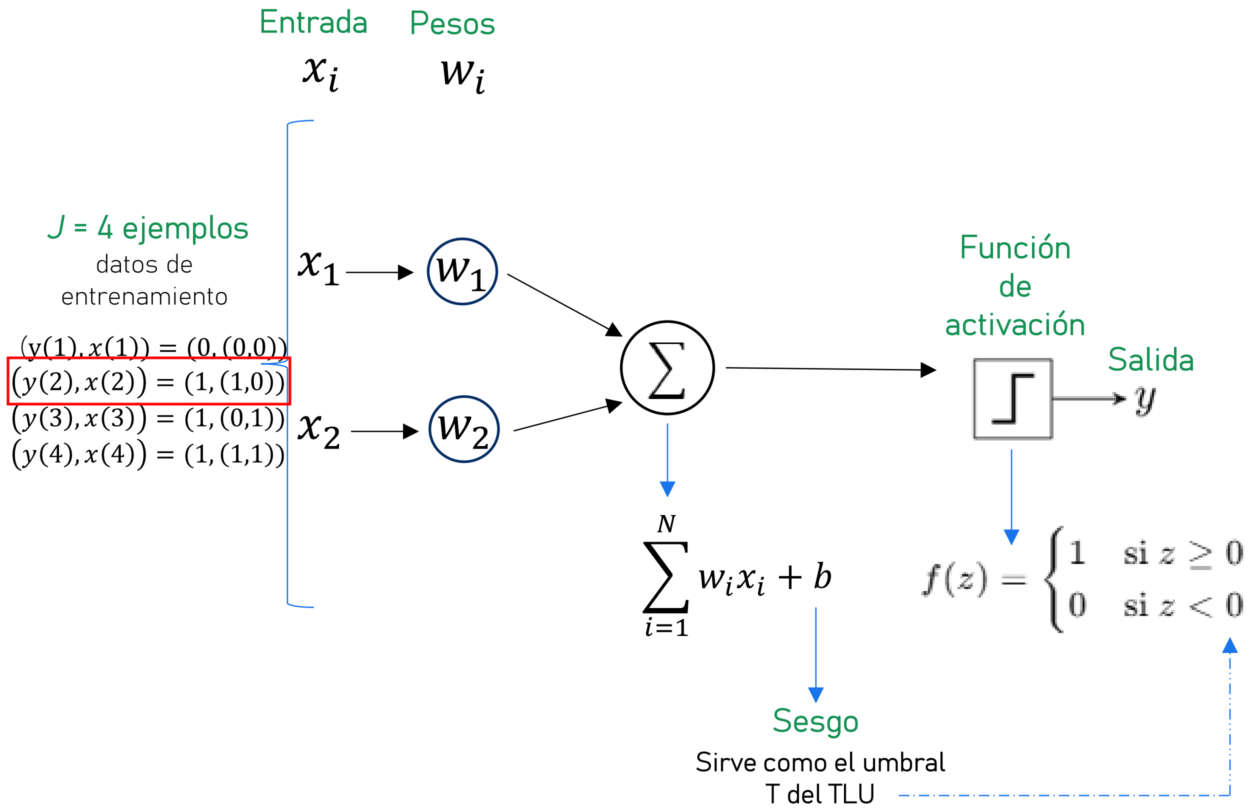
$i$ = índice del número de entrada

$i$ = índice del número de entrada

# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

## Ajuste automático de los pesos y sesgo




$i$ = índice del número de entrada
$i$ = índice del número de entrada

$J$ = 4

$$
\begin{array}{ccccc}
 & j = 1 & j = 2 & j = 3 & j = 4 \\
 & x(1) & x(2) & x(3) & x(4) \\
X = & [[0,0], & [1,0], & [0,1], & [1,1]] \\
 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \\
 & y(1) & y(2) & y(3) & y(4) \\
Y = & [0, & 1, & 1, & 1] \\
 & y & y & y & y
\end{array}
$$

$w_1 =$ $\quad w_2 =$ $\quad b =$

3. Calcula el valor estimado de $\hat{y}$ para el valor de j actual

j=2

$t$ =4

$\hat{y}(1) =$

4. Actualiza los pesos usando una tasa de aprendizaje $r$

$r$ =0.1

$w_{1(t+1)} =$
$w_{2(t+1)} =$
$b =$

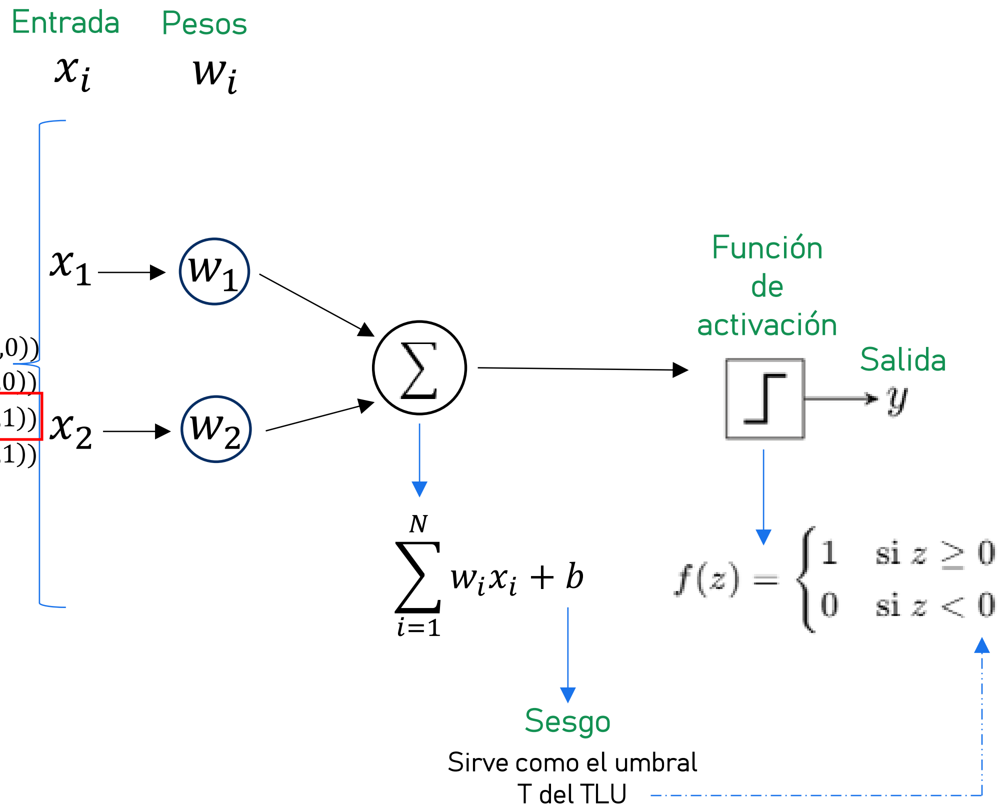
# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

Ajuste automático de los pesos y sesgo

$J$ = 4 ejemplos
datos de entrenamiento

$(y(1), x(1)) = (0, (0,0))$
$(y(2), x(2)) = (1, (1,0))$
$\boxed{(y(3), x(3)) = (1, (0,1))}$
$(y(4), x(4)) = (1, (1,1))$




$i$ = índice del número de entrada
$i$ = índice del número de entrada

$J$ = 4

$$\begin{array}{cccccccc} & j=1 & & j=2 & & j=3 & & j=4 \\ & x(1) & & x(2) & & x(3) & & x(4) \\ X = & [[0,0], & & [1,0], & & \boxed{[0,1],} & & [1,1]] \\ & x_1\ x_2 & & x_1\ x_2 & & x_1\ x_2 & & x_1\ x_2 \end{array}$$

$$\begin{array}{ccccc} & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & \boxed{1,} & 1] \\ & y & y & y & y \end{array}$$

$w_1 =$ $\quad w_2 =$ $\quad b =$

3. Calcula el valor estimado de $\hat{y}$ para el valor de j actual

j=3

$t$ =4

$\hat{y}(1) =$

4. Actualiza los pesos usando una tasa de aprendizaje $r$

$r$ =0.1

$w_{1(t+1)} =$
$w_{2(t+1)} =$
$b =$

# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

## Ajuste automático de los pesos y sesgo

$J$ = 4

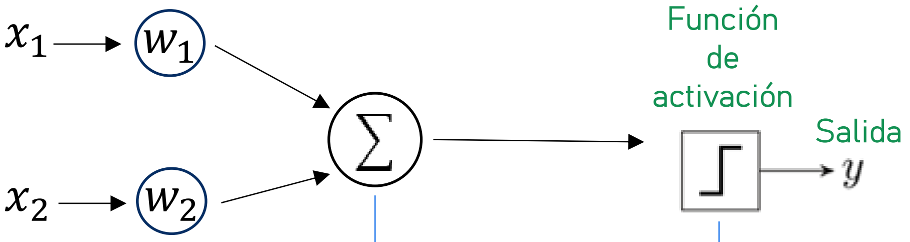
$$\begin{array}{ccccc} & j=1 & j=2 & j=3 & j=4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[0,0], & [1,0], & [0,1], & [1,1]] \\ & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \end{array}$$

$$\begin{array}{ccccc} & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & 1, & 1] \\ & y & y & y & y \end{array}$$

Entrada $x_i$ Pesos $w_i$

$J$ = 4 ejemplos
datos de entrenamiento

$(y(1), x(1)) = (0, (0,0))$
$(y(2), x(2)) = (1, (1,0))$
$(y(3), x(3)) = (1, (0,1))$
$\boxed{(y(4), x(4)) = (1, (1,1))}$




$$\sum_{i=1}^{N} w_i x_i + b$$

$$f(z) = \begin{cases} 1 & \text{si } z \geq 0 \\ 0 & \text{si } z < 0 \end{cases}$$

Sesgo
Sirve como el umbral T del TLU

$i$ = índice del número de entrada
$i$ = índice del número de entrada

$w_1 =$ $w_2 =$ $b =$

j=4

$t$ =4

3. Calcula el valor estimado de $\hat{y}$ para el valor de j actual

$\hat{y}(1) =$

4. Actualiza los pesos usando una tasa de aprendizaje $r$

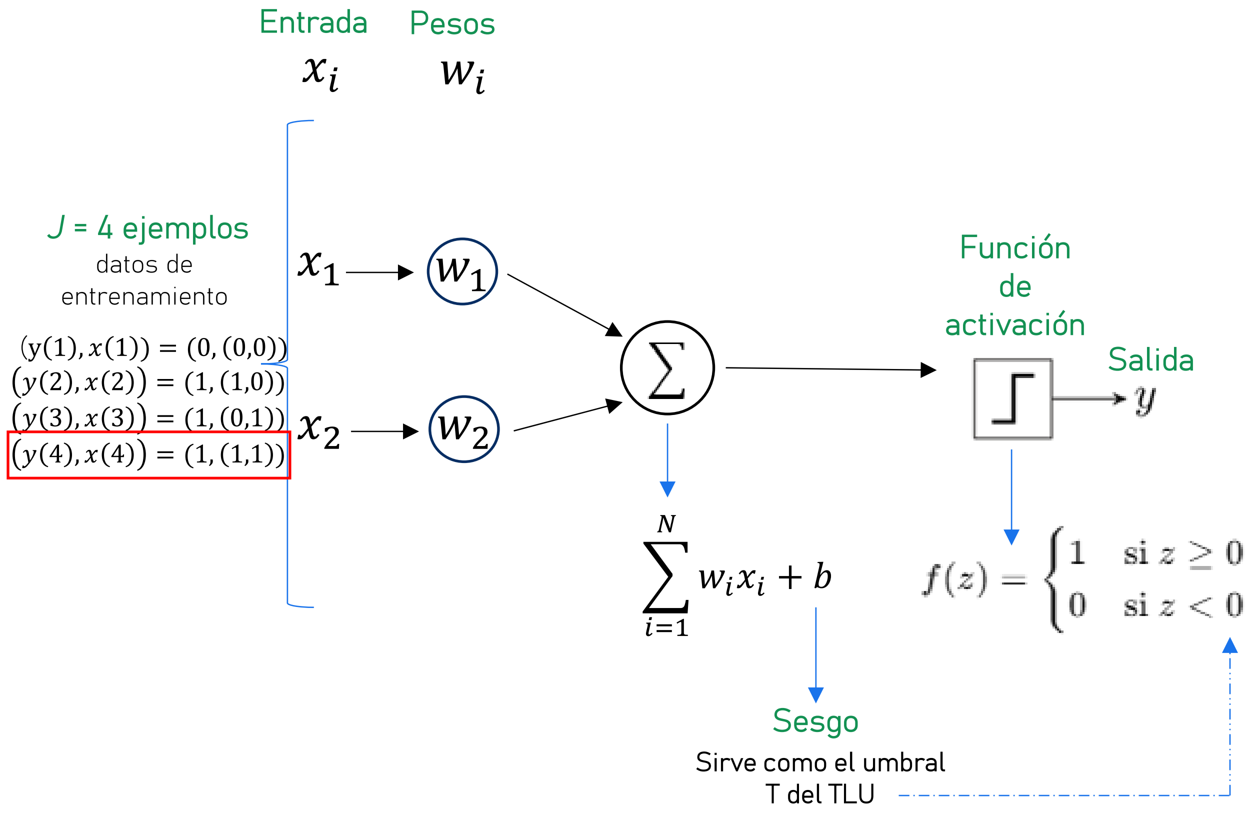
$r$ =0.1

$w_{1(t+1)} =$
$w_{2(t+1)} =$
$b =$

# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

Ajuste automático de los pesos y sesgo




$i$ = índice del número de entrada

$i$ = índice del número de entrada

$J = 4$

$$\begin{array}{ccccc} & j=1 & j=2 & j=3 & j=4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[0,0], & [1,0], & [0,1], & [1,1]] \\ & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \end{array}$$

$$\begin{array}{ccccc} & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & 1, & 1] \\ & y & y & y & y \end{array}$$

$w_1 =$ $w_2 =$ $b =$

5. Repite los pasos del 3 al 4 hasta que el error sea menor que un cierto umbral.

¿Hubo error != 0 para alguna j?

Terminamos

# Ejecutando el algoritmo a mano

| OR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

Ajuste automático de los pesos y sesgo

$J$ = 4 ejemplos

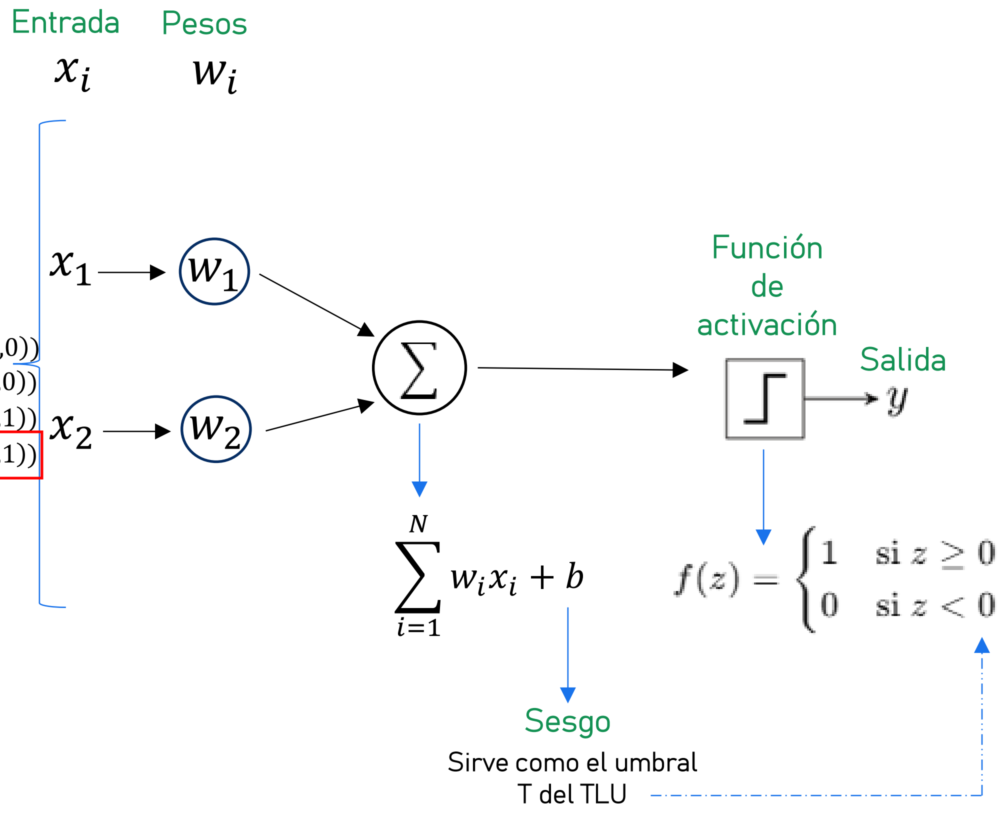
datos de entrenamiento

$(y(1), x(1)) = (0, (0,0))$

$(y(2), x(2)) = (1, (1,0))$

$(y(3), x(3)) = (1, (0,1))$

$\boxed{(y(4), x(4)) = (1, (1,1))}$




$i$ = índice del número de entrada

$i$ = índice del número de entrada

$J$ = 4

$$\begin{array}{cccccc} & j=1 & j=2 & j=3 & j=4 \\ & x(1) & x(2) & x(3) & x(4) \\ X = & [[0,0], & [1,0], & [0,1], & [1,1]] \\ & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 & x_1\ x_2 \\ & y(1) & y(2) & y(3) & y(4) \\ Y = & [0, & 1, & 1, & 1] \\ & y & y & y & y \end{array}$$

$w_1 =$ $w_2 =$ $b =$

5. Repite los pasos del 3 al 4 hasta que el error sea menor que un cierto umbral.

¿Hubo error != 0 para alguna j?

# Implementación del Perceptrón

ejercicios_TLU_y_perceptron.ipynb

Ejercicio 7, 8 y 9

# Perceptrón

Combinando las observaciones de
Hebb con el modelo TLU

## Ajuste automático de los pesos

Entradas

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right)$$

Sesgo

- Puede aprender muchos patrones.
- ¿Podrá aprender la función XOR?

| XOR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 0 |

ejercicios_TLU_y_perceptron.ipynb

Ejercicio 10

# Perceptrón

Combinando las observaciones de
Hebb con el modelo TLU

## Ajuste automático de los pesos

Entradas

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right)$$

Sesgo

- Puede aprender muchos patrones.
- ¿Podrá aprender la función XOR?

| XOR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 0 |

ejercicios_TLU_y_perceptron.ipynb

Ejercicio 10


¿por qué?

# Perceptrón


¿por qué?

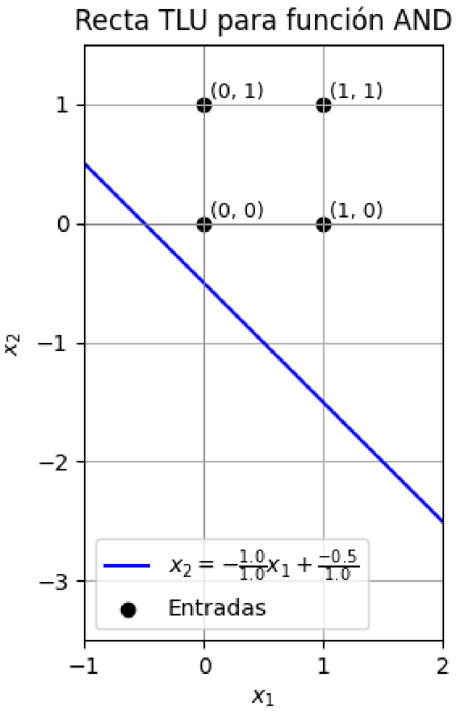
$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right)$$

Entradas

Sesgo

- Porque tanto el TLU como el Percentrón sólo pueden trazar fronteras lineales (hiperplanos).
- La función XOR no es linealmente separable.

| XOR | | |
|---|---|---|
| Input 1 | Input 2 | Output |
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 0 |




## Ecuación de la frontera de decisión

Es el conjunto de puntos donde la salida cambia: cuando el argumento del escalón es igual a 0.

$$w_1 x_1 + w_2 x_2 + b = 0$$

$$w_2 x_2 = -w_1 x_1 - b$$

$$x_2 = \frac{-w_1 x_1 - b}{w_2}$$

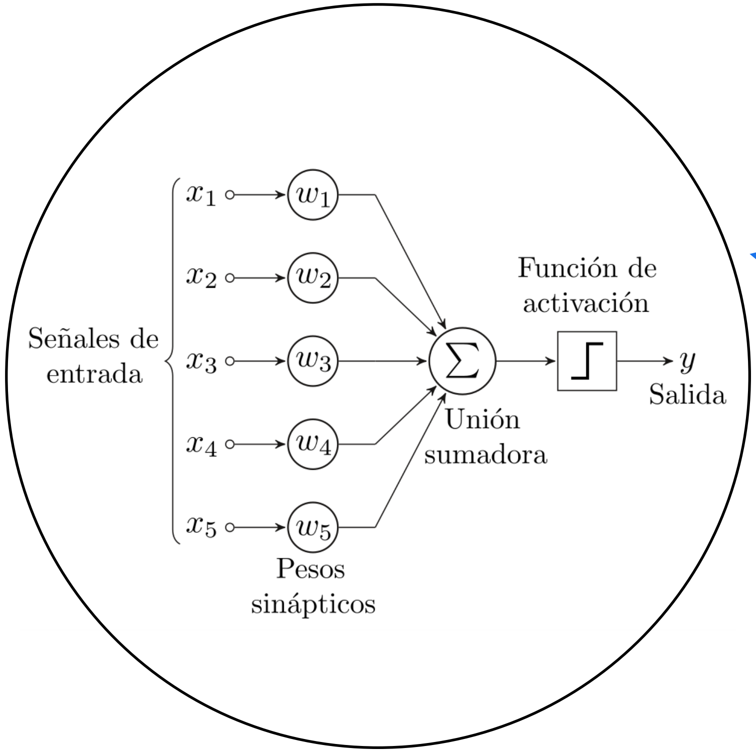
$$x_2 = -\frac{w_1}{w_2} x_1 - \frac{b}{w_2}$$

Es la ecuación de una recta de la forma:

$$y = mx + b$$

Con pendiente $-\frac{w_1}{w_2}$ y con intersección con el eje $x_2$ en $-\frac{b}{w_2}$

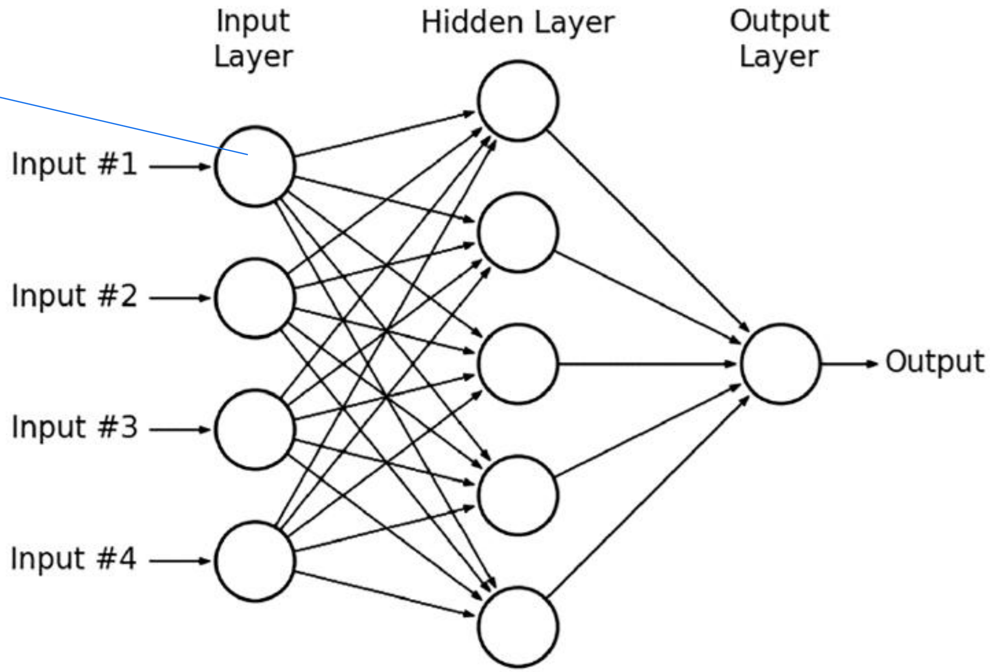
# Perceptrón Multicapa




Cada **perceptrón** lo podemos interpretar como una **neurona artificial individual**, con un algoritmo de aprendizaje que opera de forma independiente.

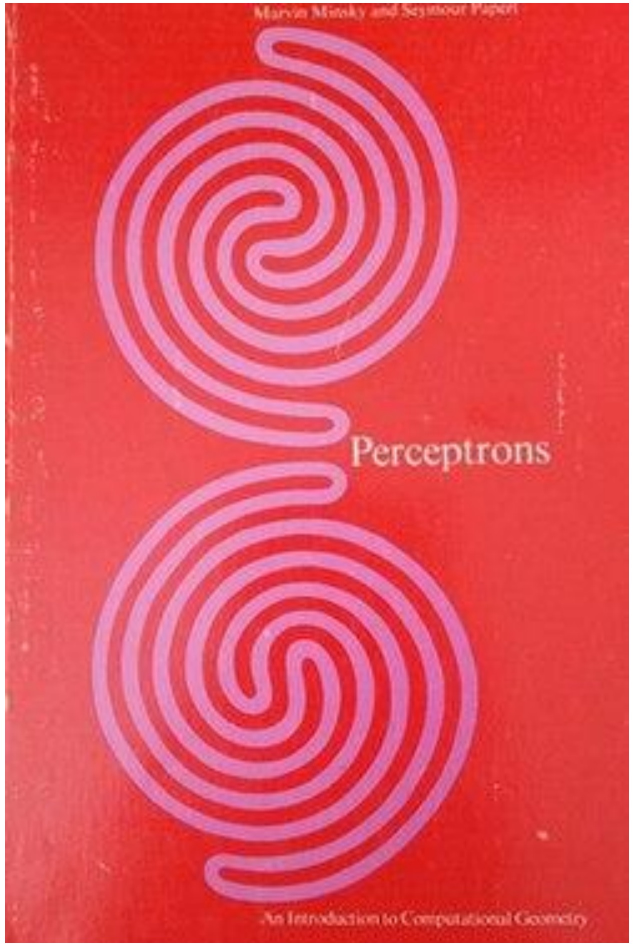
Limitación

- Podemos combinar varios perceptrones en múltiples capas:



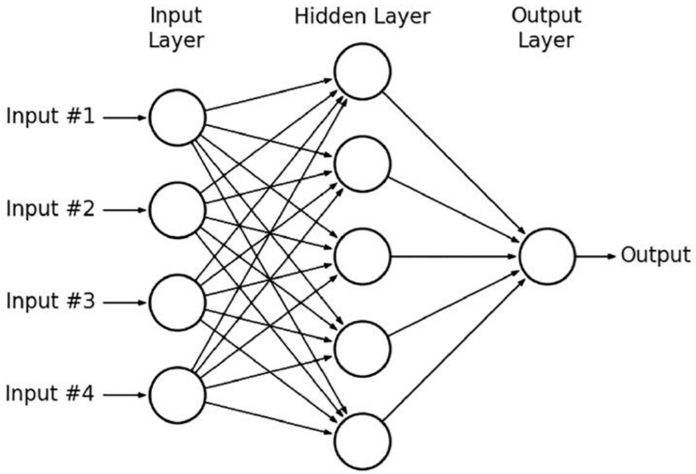

Perceptrón multicapa

# Perceptrón Multicapa




Minsky y Papert, 1969

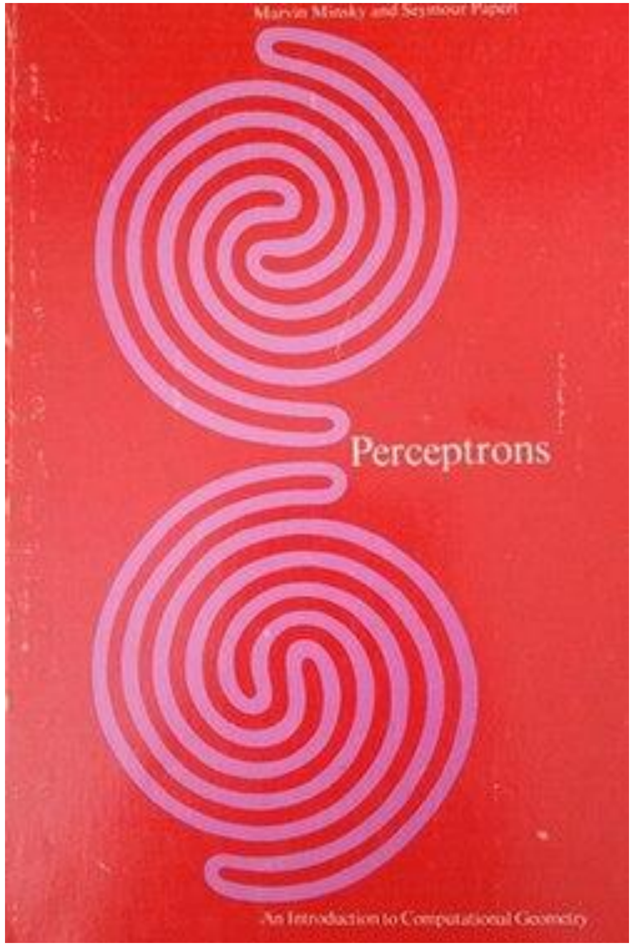
- Analizaron las **capacidades y limitaciones matemáticas** de los perceptrones.
- Analizaron qué se requería en una red con **una capa oculta** para representar **cualquier función booleana**.
- La condición era que al menos una de las neuronas de la capa oculta debía estar **conectada a todas las entradas con pesos no nulos** (es decir, que esa neurona tuviera "visibilidad total" de la entrada).
- Esto garantizaba la capacidad de esa red para expresar cualquier función booleana con suficiente número de neuronas en la capa oculta.




- En lugar de tener una estructura muy dispersa, como las neuronas biológicas —que solo están conectadas a unas pocas de sus vecinas—, estos modelos computacionales **requerían conexiones muy densas**.

¡mayor costo computacional y más parámetros que ajustar!

# Perceptrón Multicapa




Minsky y Papert, 1969

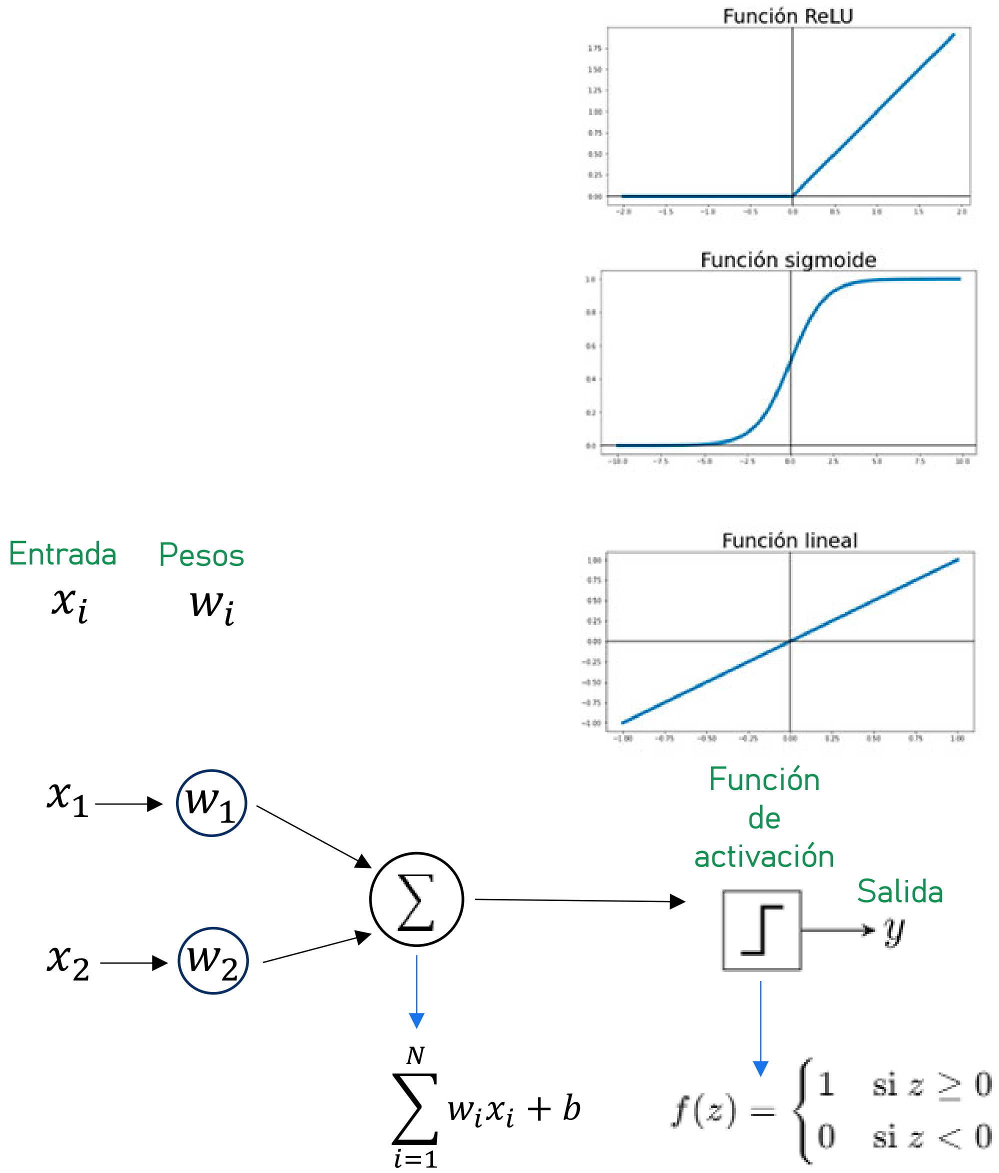
- Estos modelos resultaban computacionalmente poco manejables en el hardware de la época.

- La observación (errónea) de que los modelos dispersos no podían calcular todas las operaciones lógicas se interpretó de manera más amplia por la comunidad investigadora como que *los perceptrones no pueden calcular XOR*.

Invierno de la IA

- Caída drástica en interés, financiamiento y actividad de la investigación en IA.

# Perceptrones Multicapa

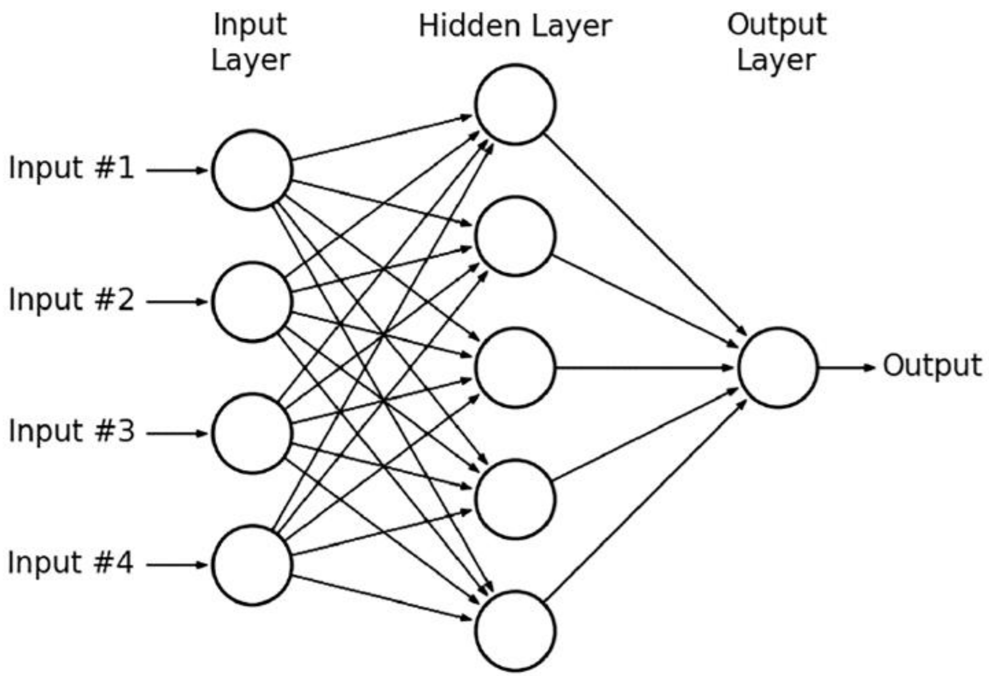



- Sin embargo, los investigadores seguían reconociendo que **estos modelos tenían valor**:
  - Especialmente cuando se ensamblaban en redes multicapa, cada una compuesta por varias unidades de perceptrón.

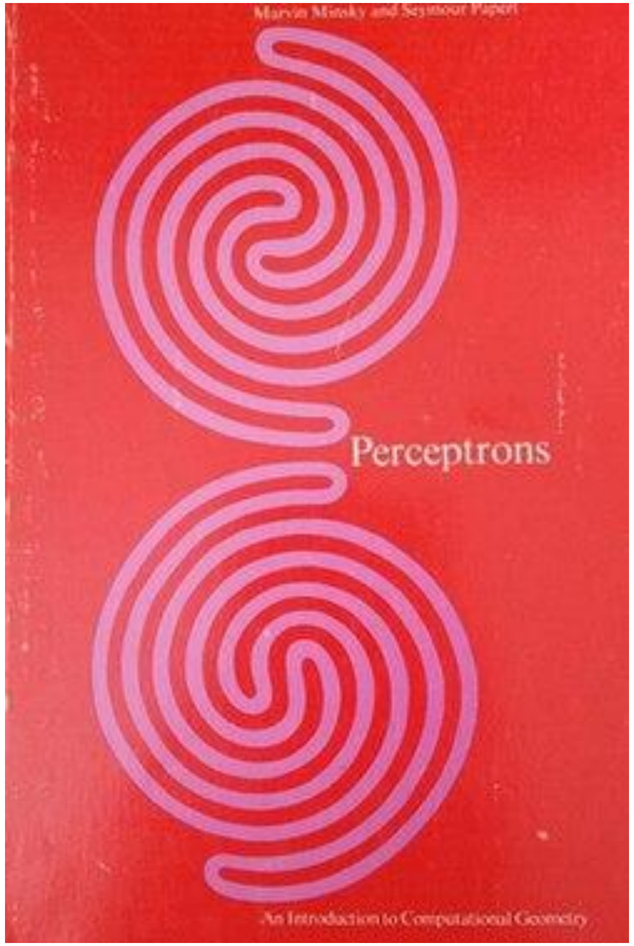
  - Cuando la forma matemática de la **función de salida del modelo se flexibilizó** para adoptar diversas formas (como una función lineal o una sigmoide),
    - **estas redes podían resolver tanto problemas de regresión como de clasificación**, con resultados teóricos que **mostraban que las redes de tres capas podían aproximar eficazmente cualquier salida**.

# Perceptrón Multicapa

- **Ninguno** de estos trabajos **resolvió** las **limitaciones prácticas** para **calcular soluciones en redes multicapa**.,
- El **algoritmo de aprendizaje del perceptrón** resultaba **insuficiente** para el entrenamiento aplicado de estos modelos.
- El principal desafío era **cómo estimar correctamente los pesos de las capas ocultas**.
- Dichos pesos determinan la **representación interna** de los datos en la red.



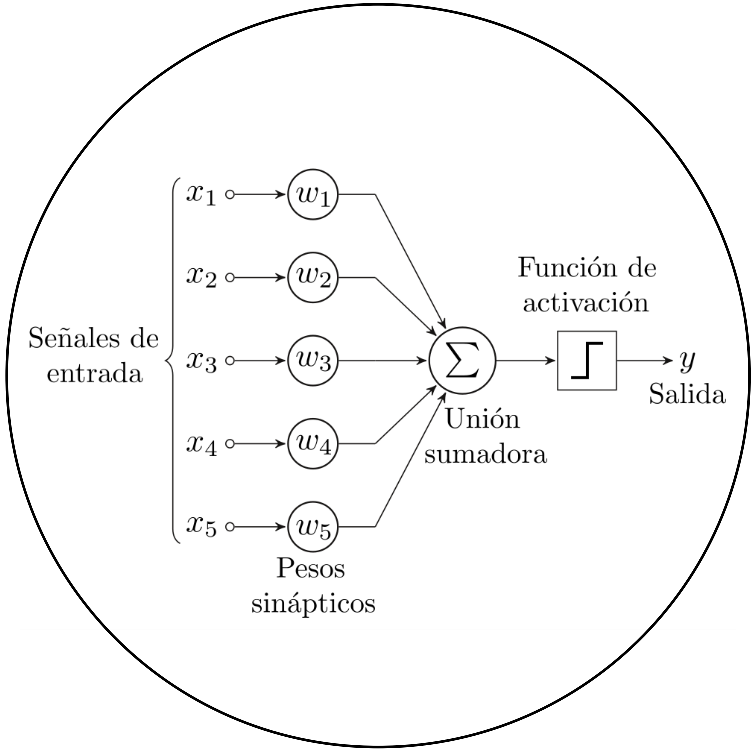
# Perceptrón Multicapa

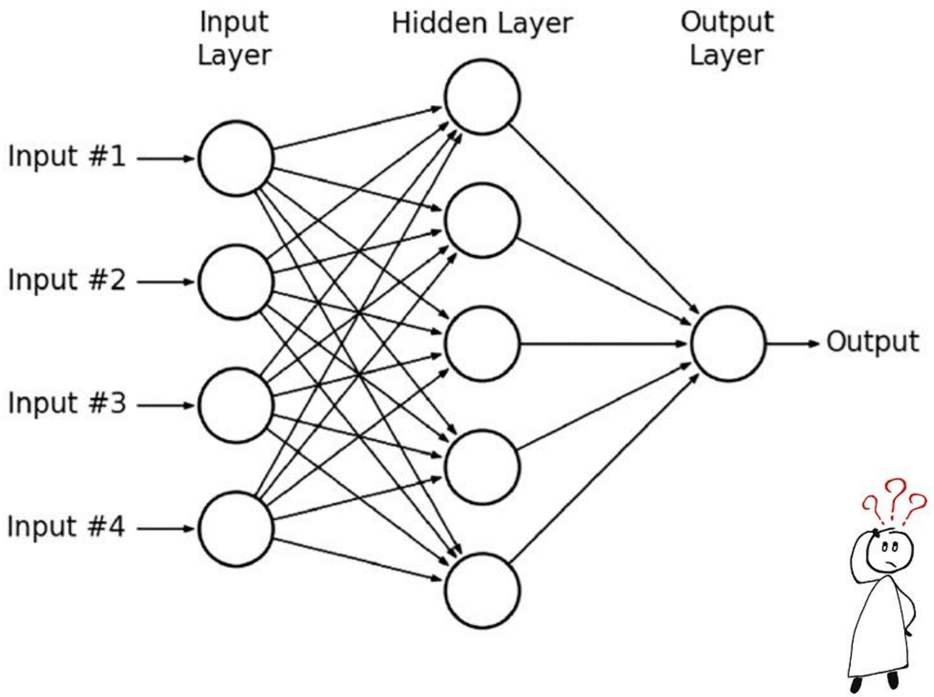



Minsky y Papert, 1969

- La siguiente revolución en la investigación de redes neuronales requeriría:
  - una **forma más eficiente de calcular los parámetros necesarios en modelos complejos.**
- En **arquitecturas posteriores:**
  - se usan conexiones dispersas, como en las CNN.
  - pero las conexiones densas siguen siendo una característica de muchos modelos modernos, particularmente en las capas totalmente conectadas que suelen formar las penúltimas capas ocultas de los modelos.

# Perceptrones Multicapa y Retropropagación




En el perceptrón, una regla de aprendizaje para actualizar pesos es relativamente fácil de derivar, mientras no existan capas ocultas.

La entrada se transforma una sola vez por el perceptrón para calcular un valor de salida → los pesos pueden ajustarse directamente para producir la salida.

Cuando existen capas ocultas entre la entrada y salida, el problema se vuelve más complejo.

¿cuándo debemos modificar los pesos internos para calcular las activaciones que alimentan la salida final?

¿Cómo debemos ajustarlos en relación con los pesos de entrada?