

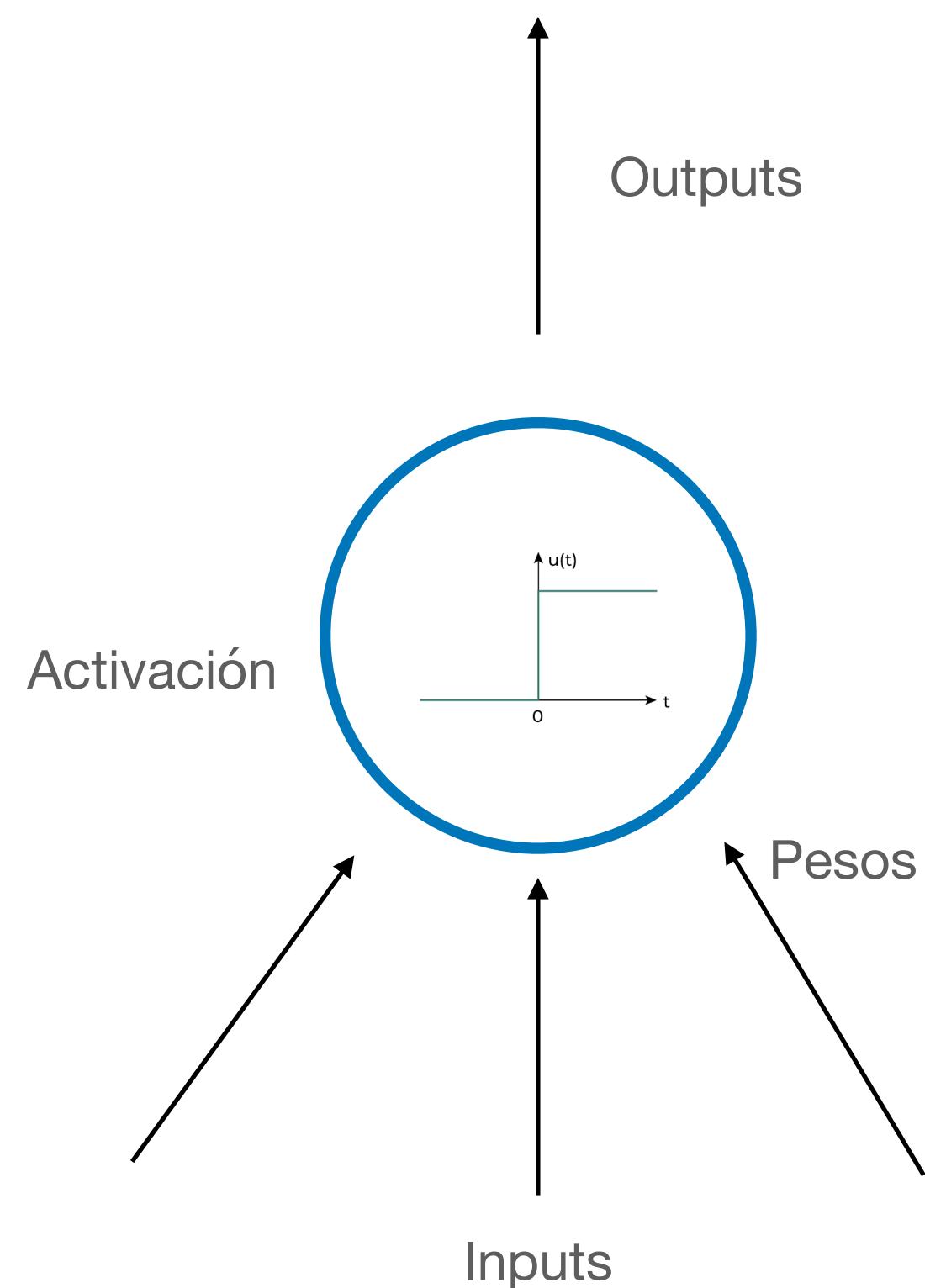
Redes neuronales

Maestría en Ciencia de Datos

M. en C. Iván Toledano, Octubre 2024

Neurona

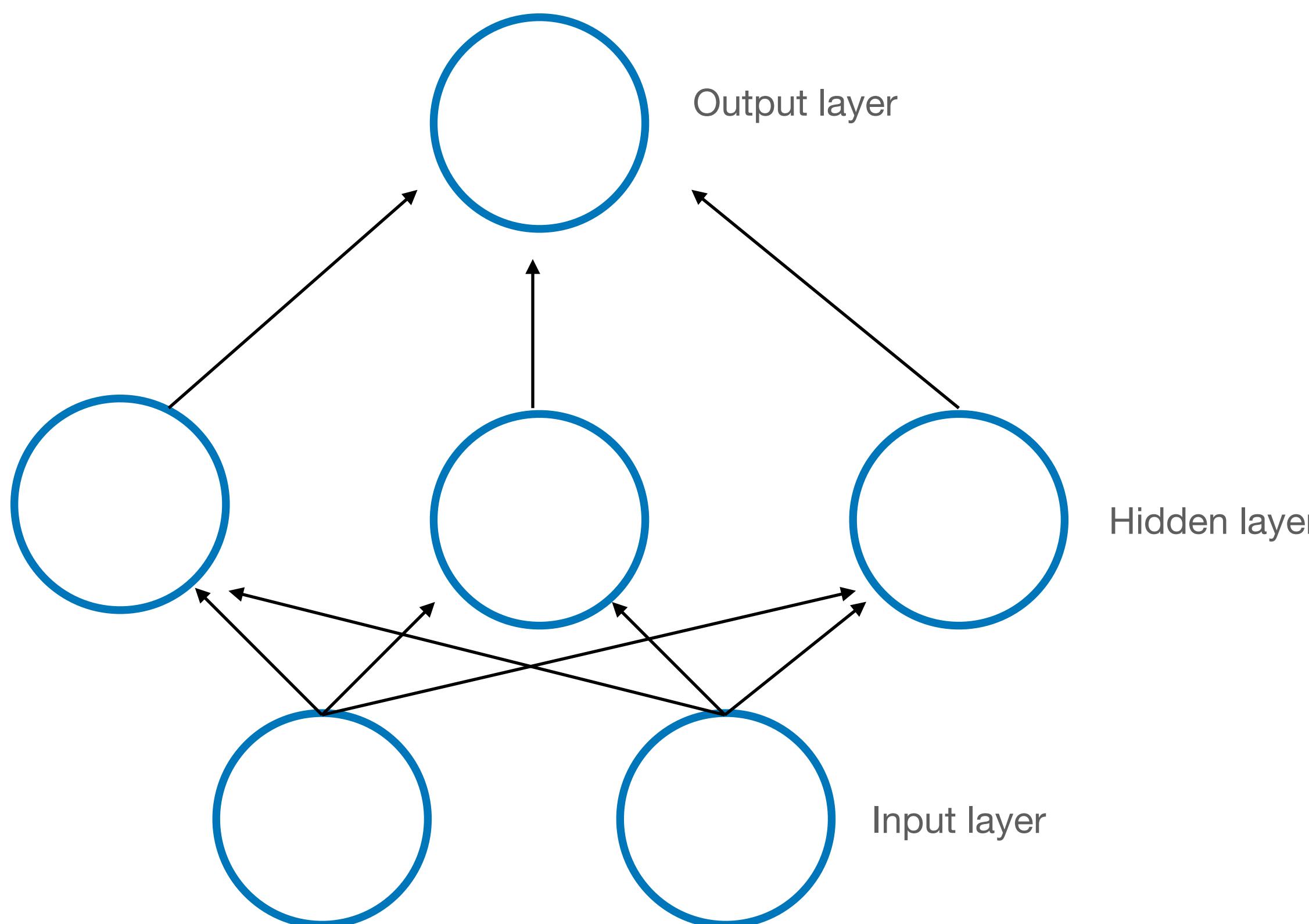
Unidad fundamental



- Se tienen una serie de entradas (features).
- Se aplica una función de activación para corregir el proceso de aprendizaje.

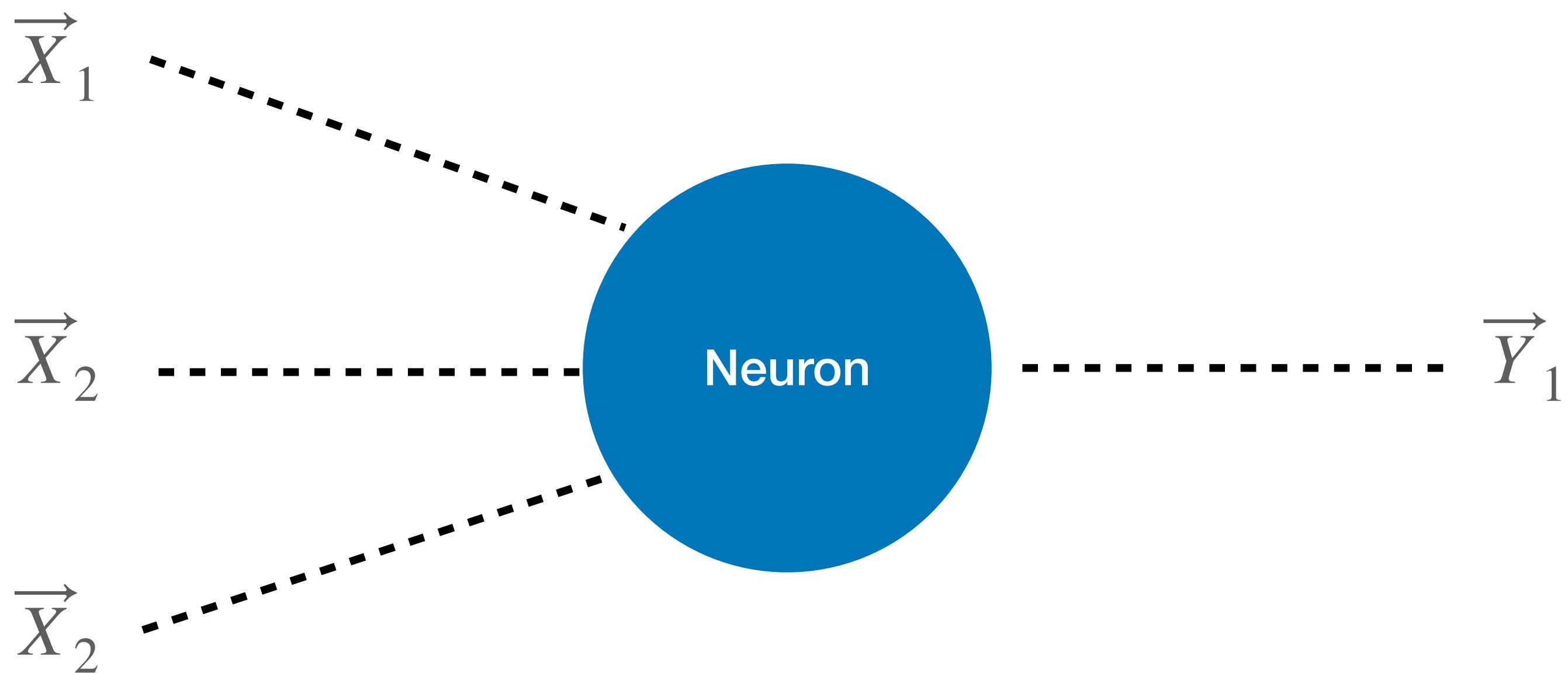
Neurona

Perceptron multicapa

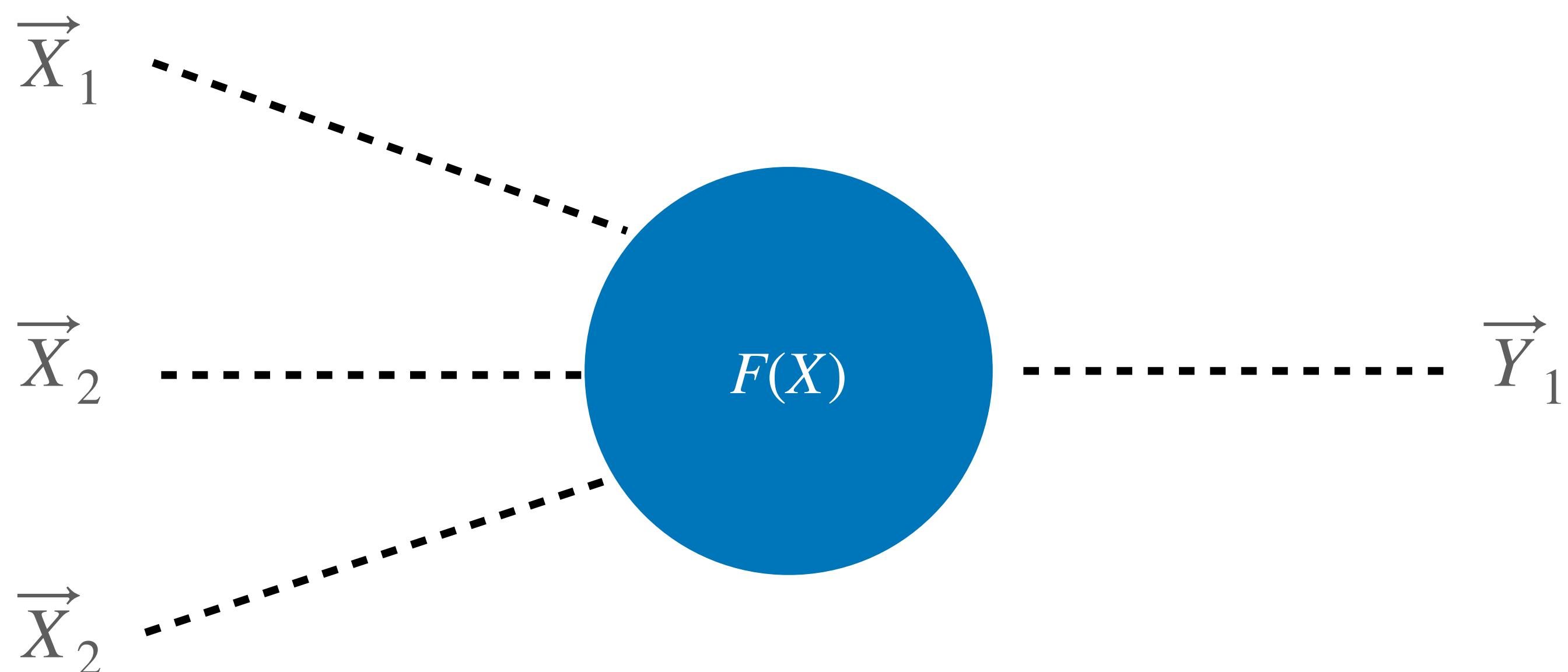


- Si juntamos varias unidades fundamentales o varias neuronas, tenemos lo que se llama un perceptron multicapa.
-

Neurona

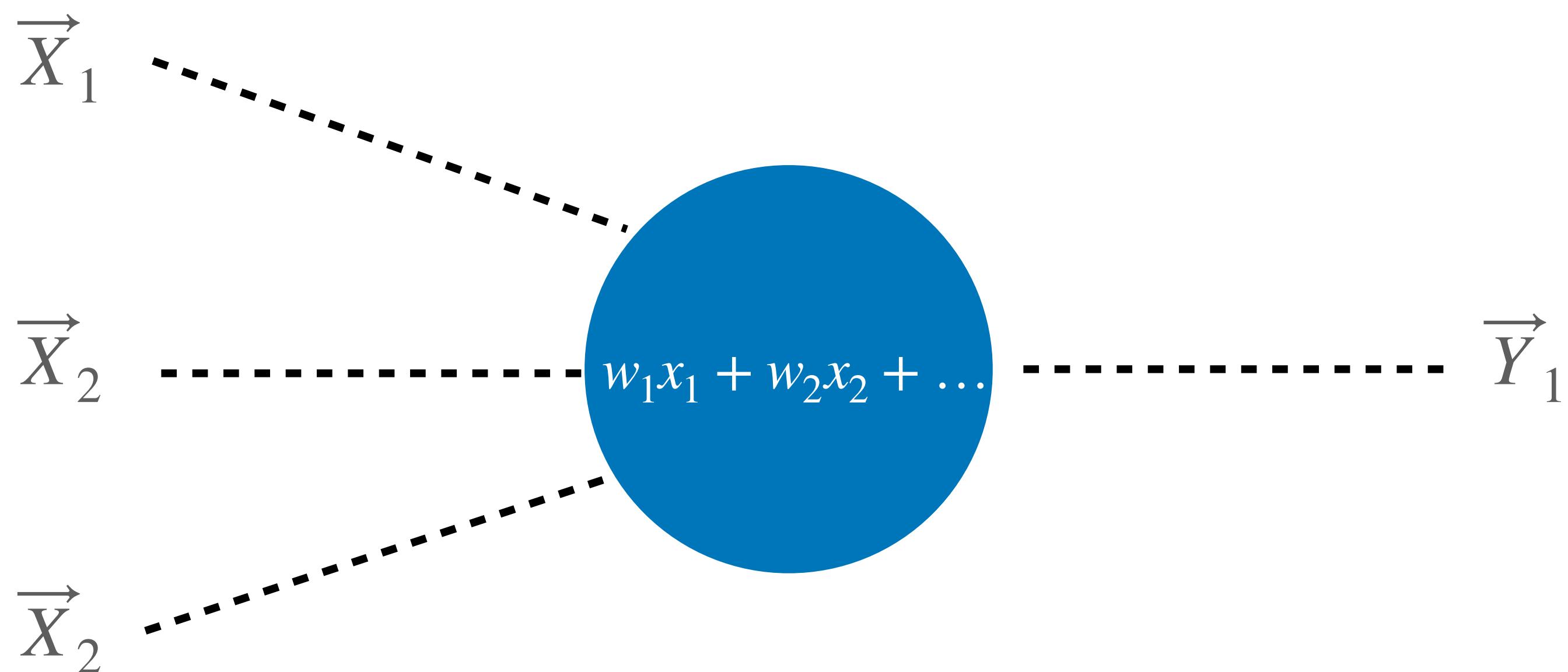


Neurona



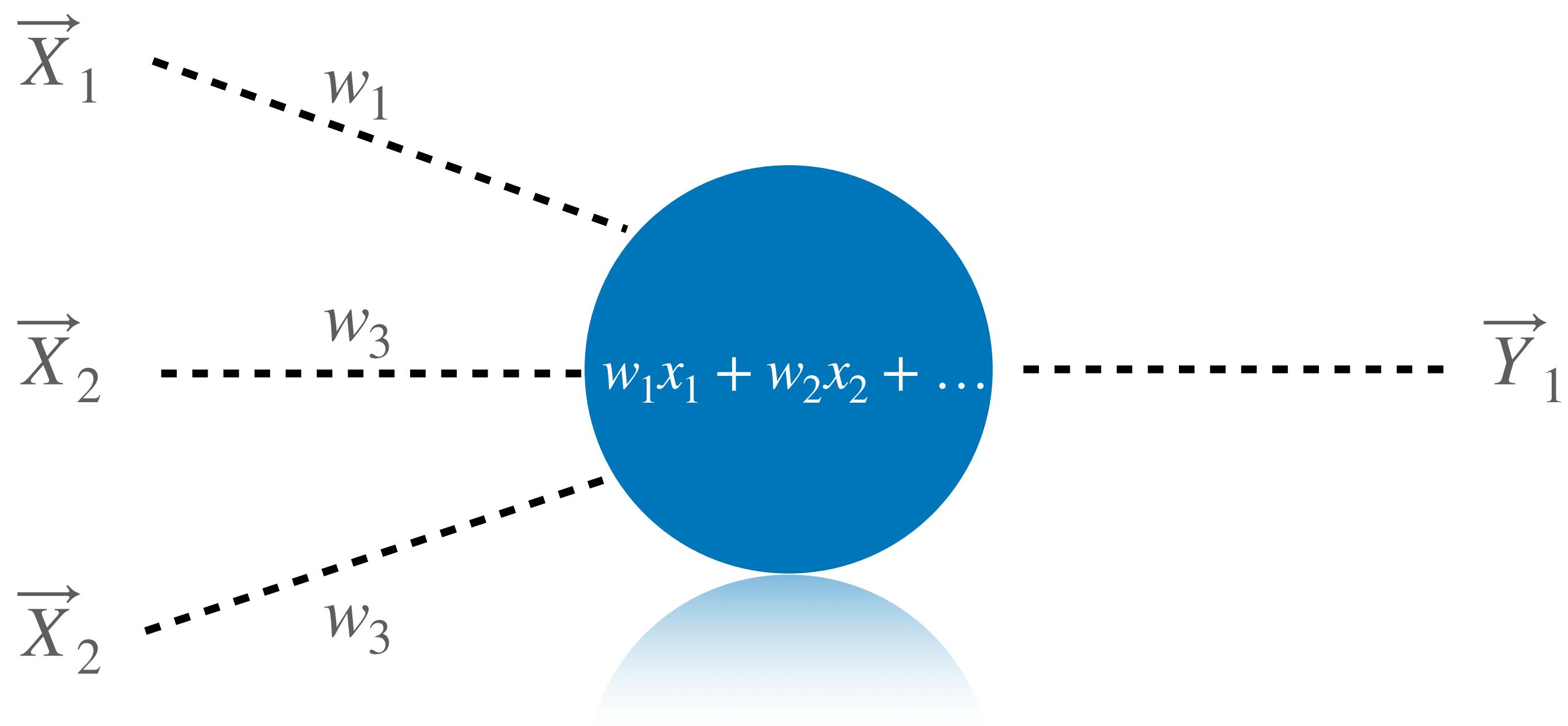
- Internamente, la neurona va a tener una función que va a evaluar estas entradas, junto con sus pesos.
- Esa función de x , podría ser en términos de los pesos.

Neurona



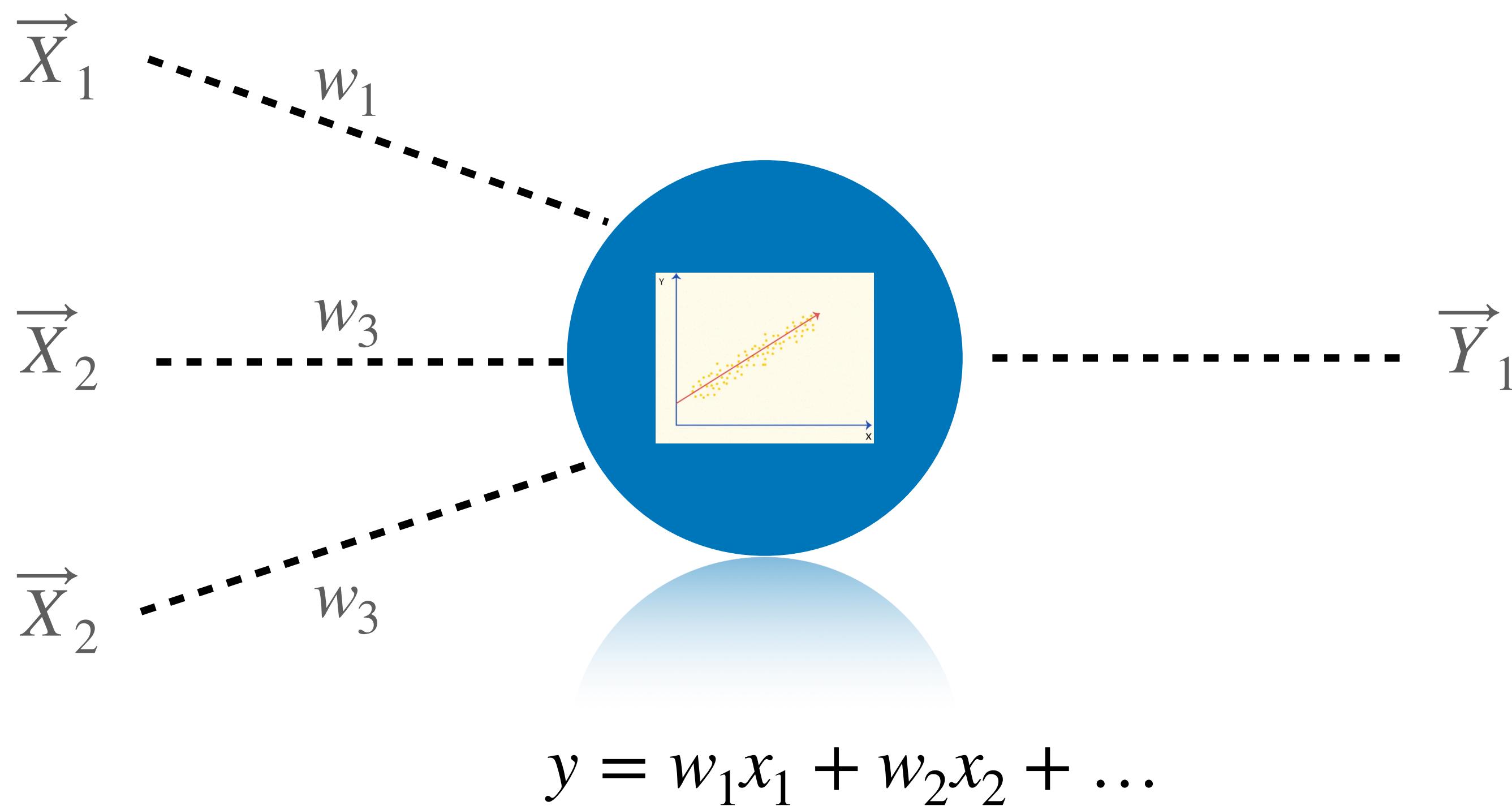
- Internamente, la neurona va a tener una función que va a evaluar estas entradas, junto con sus pesos.
- Esa función de x , podría ser en términos de los pesos.

Neurona



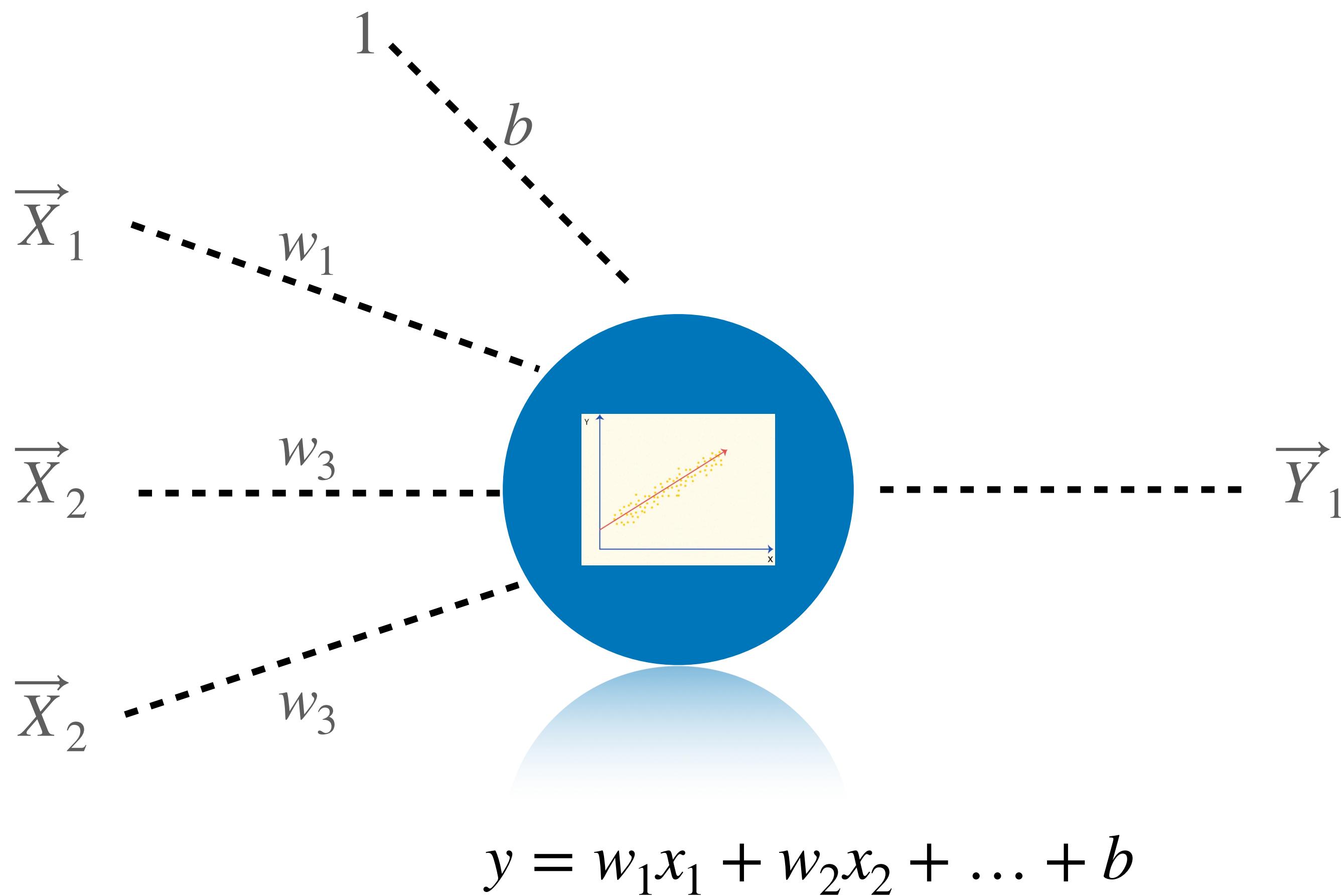
- $w_1x_1 + w_2x_2 + \dots$
- Es función de ejemplo
sería equivalente a una
regresión lineal, como
una suma ponderada.
- El peso w se asigna a
cada una de las
variables.

Neurona



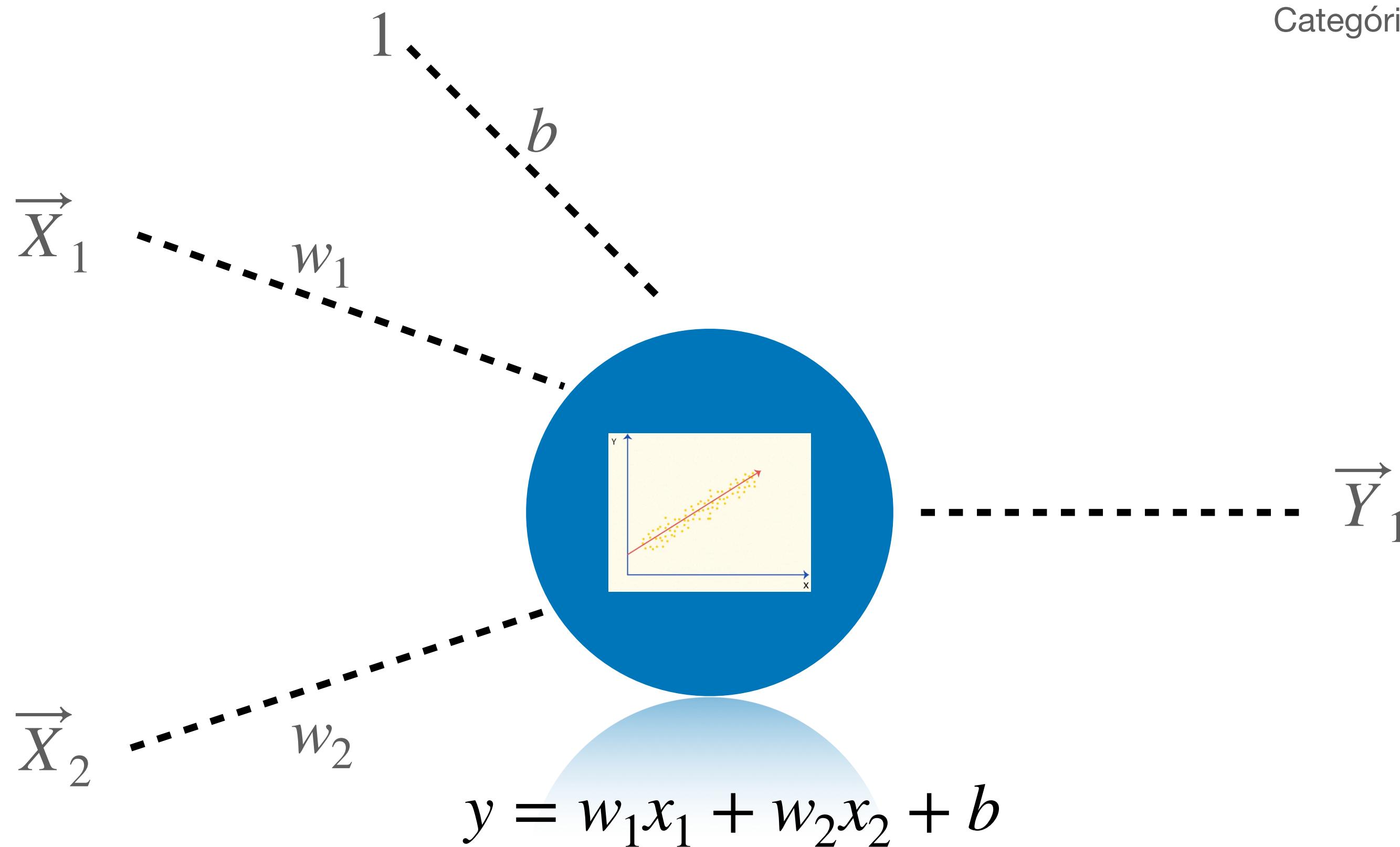
- $\hat{y} = \beta_0 + \beta_1x_1 + \dots$
- Una función de regresión lineal que se ajusta a los datos de entrada, salvo el término β_0 .
- En este caso, este término independiente es llamado **sesgo/bias**.

Neurona



- $\hat{y} = \beta_0 + \beta_1x_1 + \dots$
- Una función de regresión lineal que se ajusta a los datos de entrada, salvo el término β_0 .
- En este caso, este término independiente es llamado **sesgo/bias**.

Ejemplo



Problema:
Categórico. Aprobar/No aprobar un examen dados dos datos categóricos:
Estudiar y estar en casa.

| Estudiar | Estar en casa | |
|----------|---------------|-----------|
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| 0 | 0 | Resultado |

Ejemplo

$$\vec{W} \cdot \vec{X} \leq \text{threshold} \rightarrow Y = 0$$

$$\vec{W} \cdot \vec{X} > \text{threshold} \rightarrow Y = 1$$

Problema:
Categórico. Aprobar/No aprobar un examen dados dos datos categóricos:
Estudiar y estar en casa.

| Estudiar | Estar en casa | |
|----------|---------------|-----------|
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| 0 | 0 | Resultado |

- Podemos interpretar que un peso es asignado a esta característica, y que si su producto con ésta es mayor o menor a un umbral, se tiene una clasificación.
- Despejando tenemos que $\vec{W} \cdot \vec{X} - \text{threshold} \leq / > 0$, por lo que este umbral esta relacionado con el término independiente **bias** $\text{bias} = -\text{threshold}$.

Ejemplo

$$\vec{W} \cdot \vec{X} + \beta_0 \leq 0 \rightarrow Y = 0$$

$$\vec{W} \cdot \vec{X} + \beta_0 > 0 \rightarrow Y = 1$$

Problema:
Categórico. Aprobar/No aprobar un examen dados dos datos categóricos:
Estudiar y estar en casa.

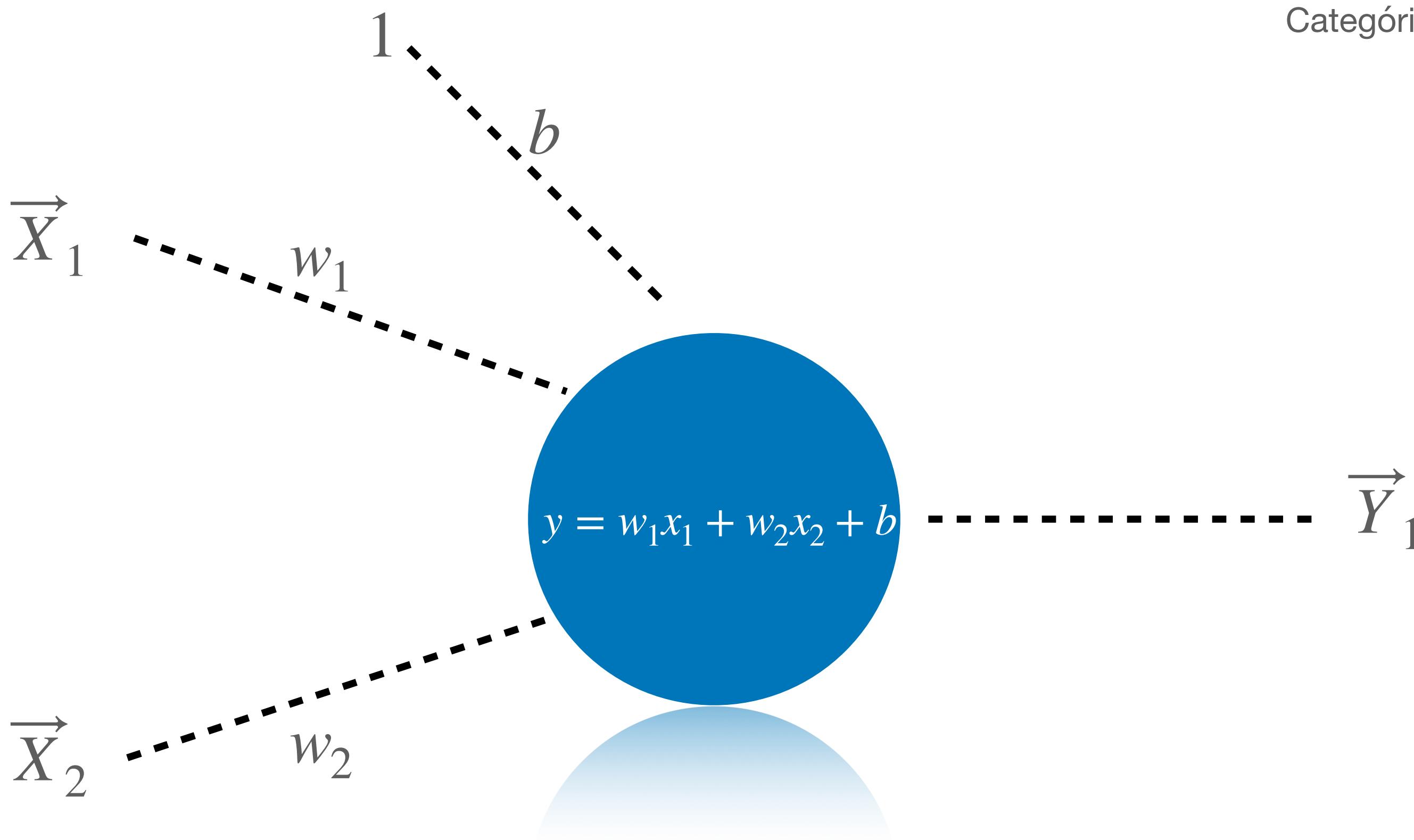
| Estudiar | Estar en casa | |
|----------|---------------|-----------|
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| 0 | 0 | Resultado |

- Podemos interpretar que un peso es asignado a esta característica, y que si su producto con ésta es mayor o menor a un umbral, se tiene una clasificación.
- Despejando tenemos que $\vec{W} \cdot \vec{X} - \text{threshold} \leq / > 0$, por lo que este umbral esta relacionado con el término independiente **bias** $\text{bias} = -\text{threshold}$.

- El resultado de la red neuronal estará condicionada por el valor de los **parámetros/features**.
- Por tanto, tenemos que encontrar cuánto valen estos pesos y el bias.

Ejemplo

Buscando parámetros

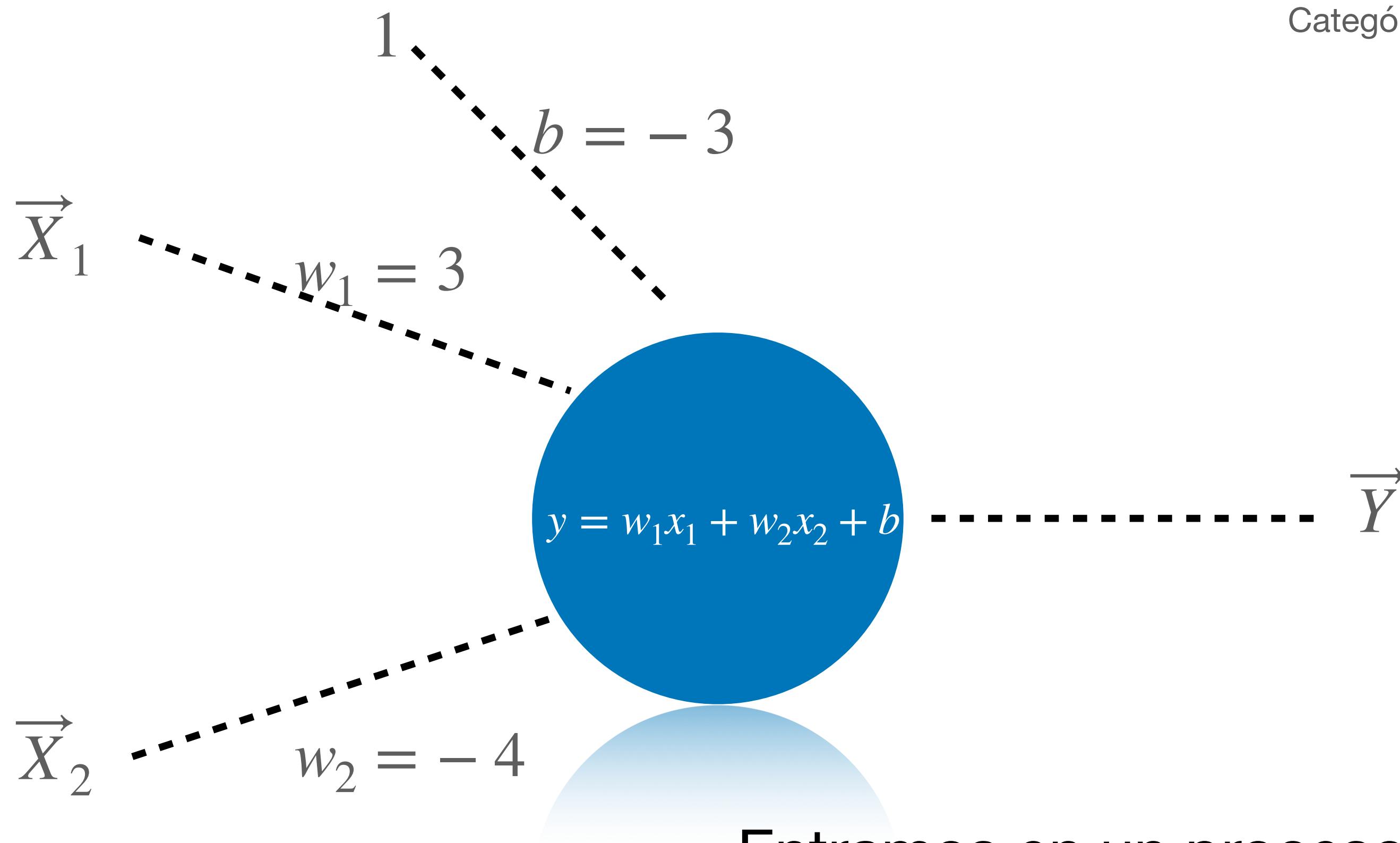


Problema:
Categórico. Aprobar/No aprobar un examen dados dos datos categóricos:
Estudiar y estar en casa.

| X1 | X2 | Target | Y |
|----|----|----------|---|
| 0 | 0 | 0 | |
| 1 | 0 | 0 | |
| 0 | 1 | 0 | |
| 1 | 1 | 1 | |

Ejemplo

Buscando parámetros



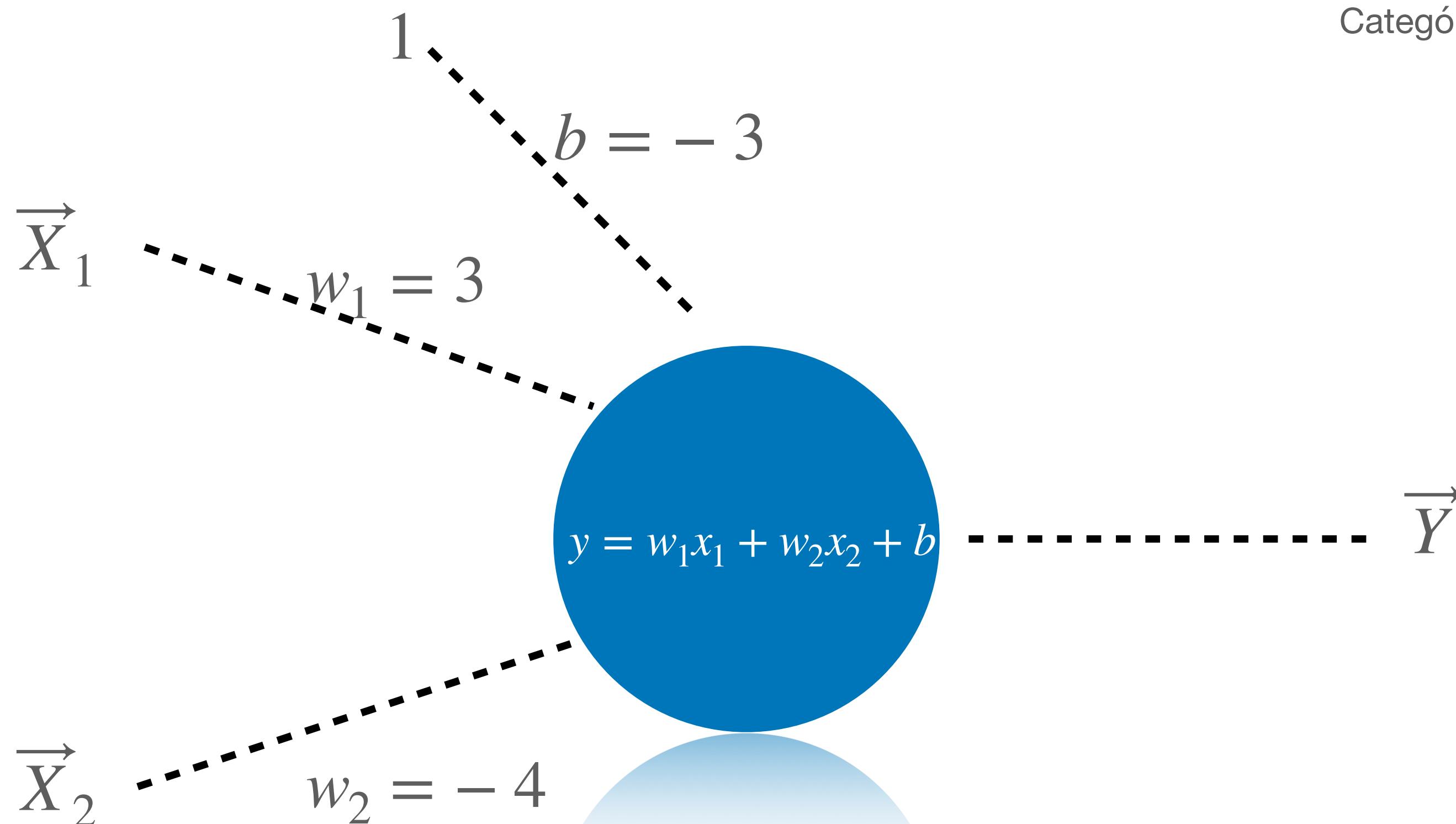
Problema:
Categórico. Aprobar/No aprobar un examen dados dos datos categóricos:
Estudiar y estar en casa.

| X1 | X2 | Target | Y |
|----|----|--------|----|
| 0 | 0 | 0 | -3 |
| 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | -7 |
| 1 | 1 | 1 | -4 |

- Entramos en un proceso de asignación de pesos.
- Los resultados de clasificación con estos parámetros se encuentran en la tabla.

Ejemplo

Buscando parámetros



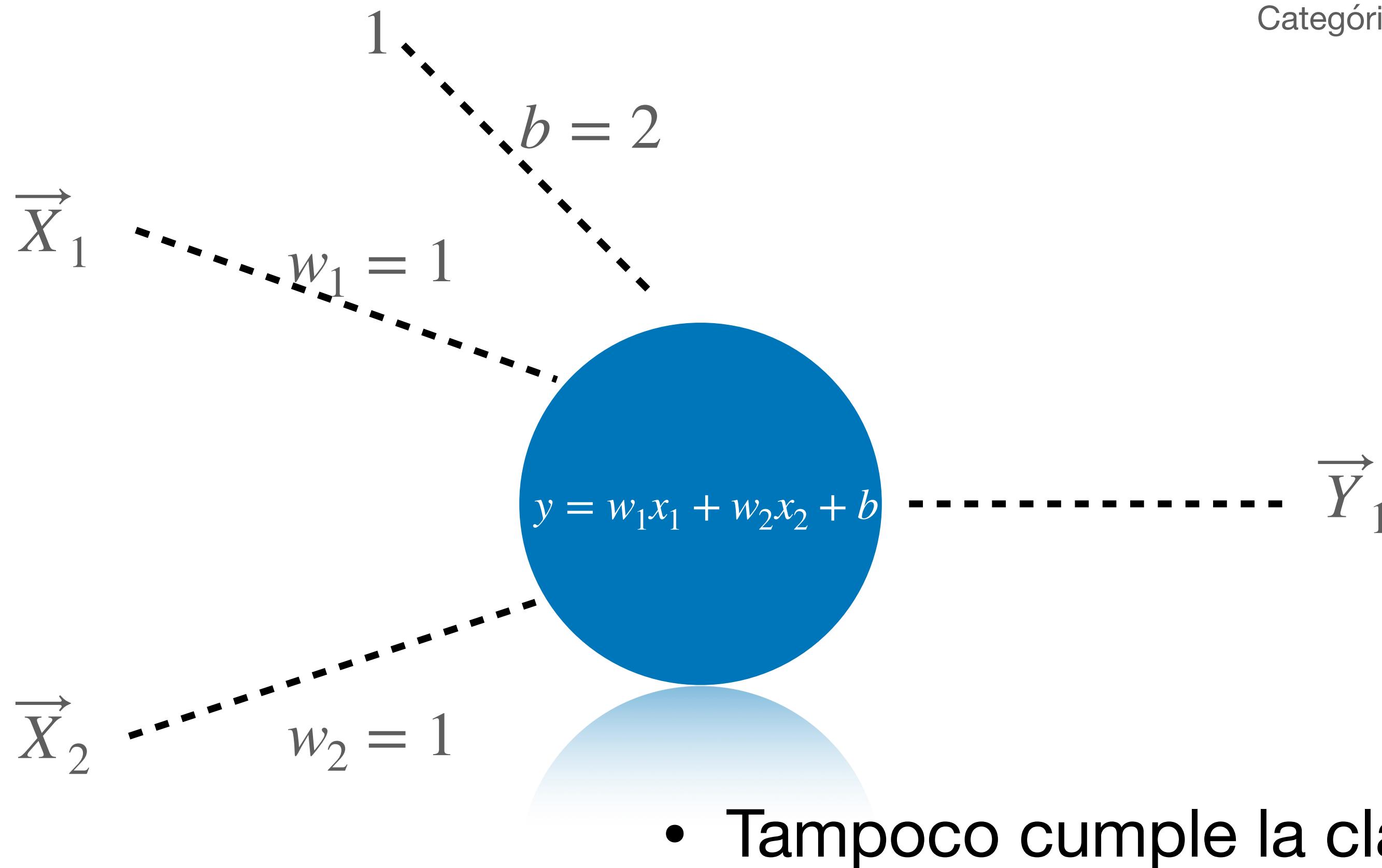
Problema:
Categórico. Aprobar/No aprobar un examen dados dos datos categóricos:
Estudiar y estar en casa.

| X1 | X2 | Target | Y |
|----|----|--------|----|
| 0 | 0 | 0 | -3 |
| 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | -7 |
| 1 | 1 | 1 | -4 |

- En este caso de asignación de pesos, tres de los casos salieron negativos y sólo un caso salió positivo (cero). Esto **no cumple** con la clasificación que teníamos como objetivo.
- Esta configuración de pesos de parámetros no cumple. En este caso se pasaría a otra iteración (**epoch**). Aquí se corrigen los pesos.

Ejemplo

Otra época

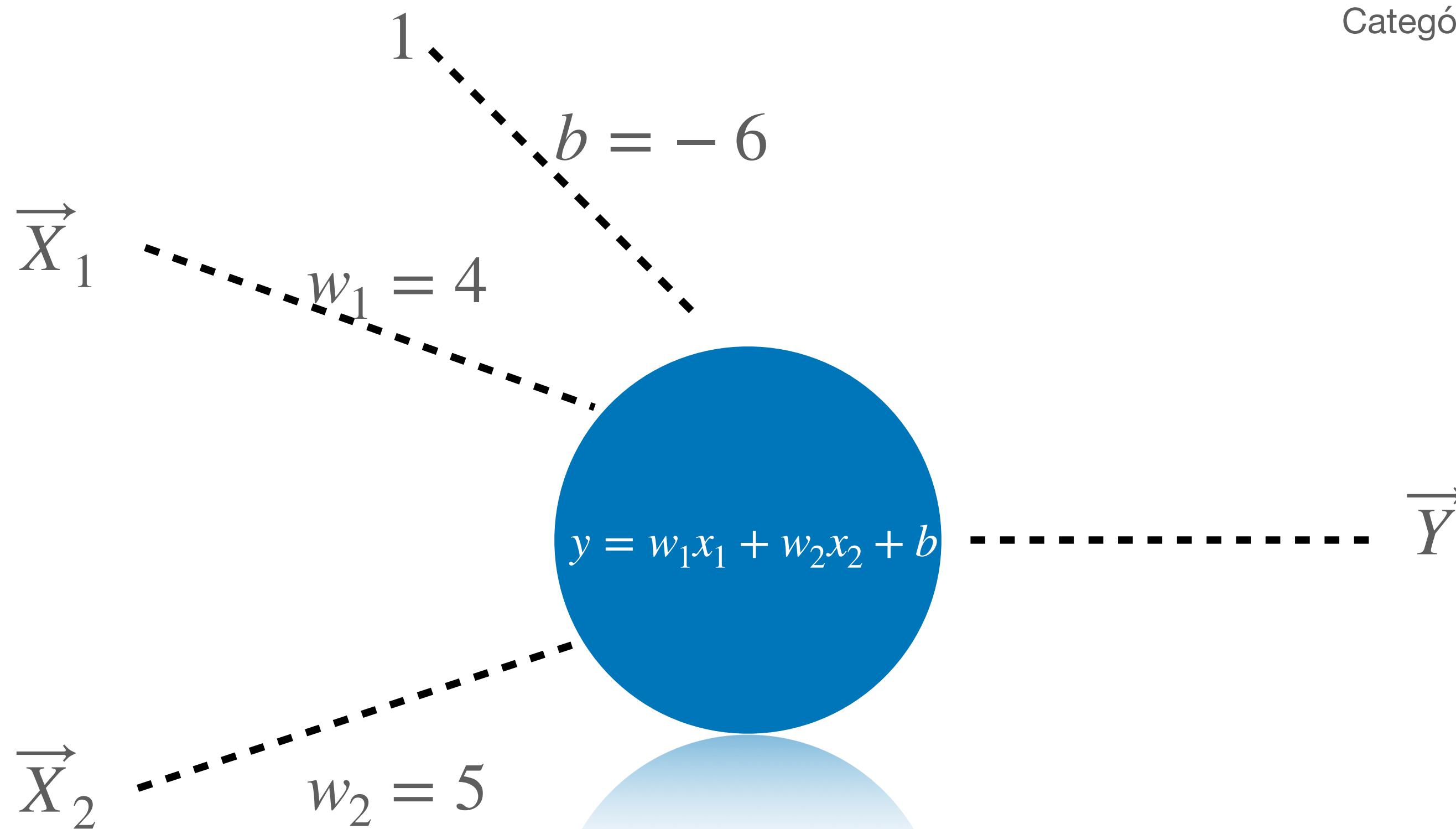


Problema:
Categórico. Aprobar/No aprobar un examen dados dos datos categóricos:
Estudiar y estar en casa.

| X1 | X2 | Target | Y |
|----|----|--------|---|
| 0 | 0 | 0 | 2 |
| 1 | 0 | 0 | 3 |
| 0 | 1 | 0 | 3 |
| 1 | 1 | 1 | 4 |

Ejemplo

Otra época



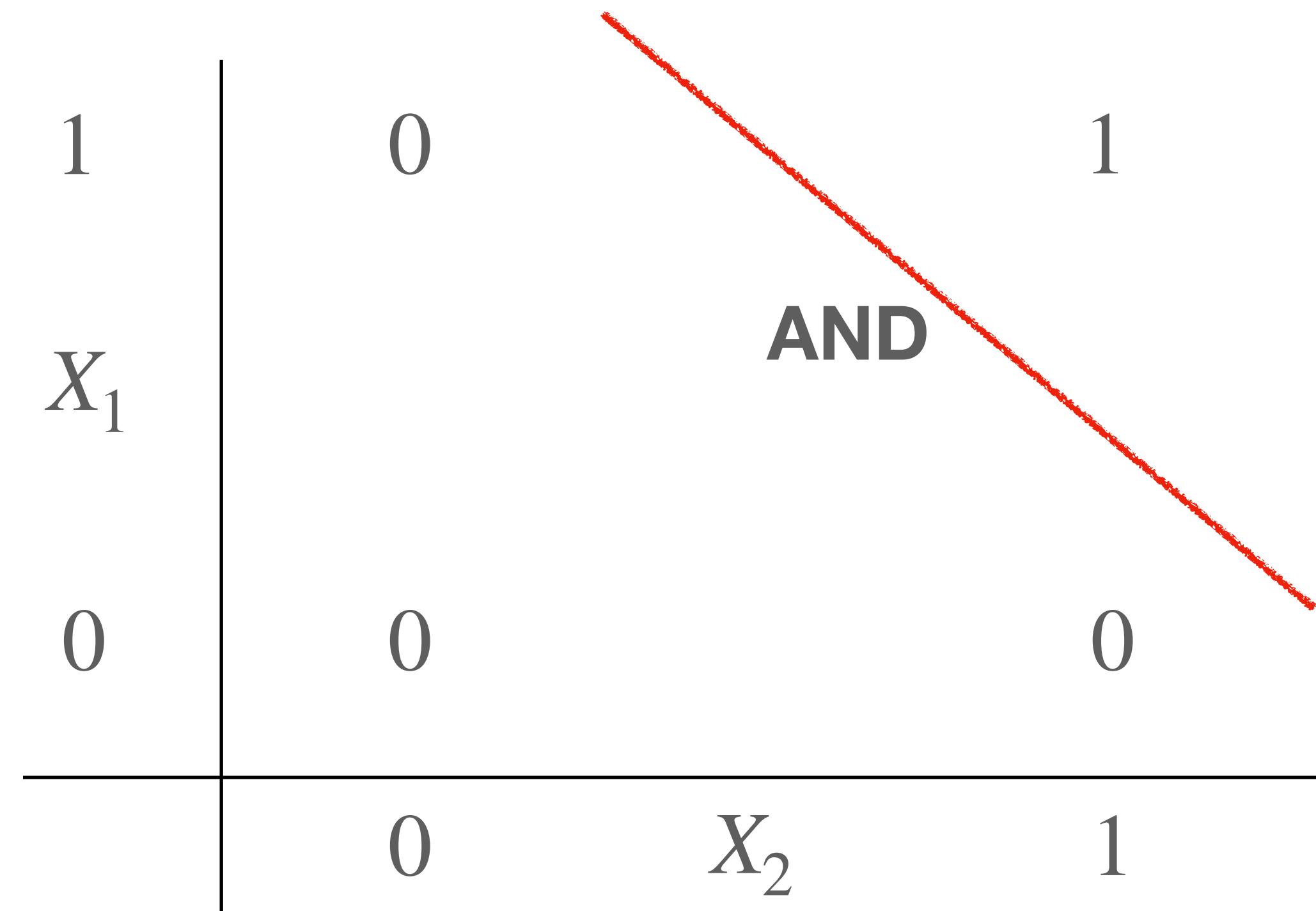
- En este caso sí se cumple la clasificación.
- Tengo asignados valores positivos Y para el caso 1, y tengo asignados valores negativos Y para el caso 0.
- El algoritmo **backpropagation** realiza la búsqueda de estos pesos y el bias, en un proceso multicapa.

Problema:
Categórico. Aprobar/No aprobar un examen dados dos datos categóricos:
Estudiar y estar en casa.

| X1 | X2 | Target | Y |
|----|----|--------|----|
| 0 | 0 | 0 | -6 |
| 1 | 0 | 0 | -2 |
| 0 | 1 | 0 | -1 |
| 1 | 1 | 1 | 3 |

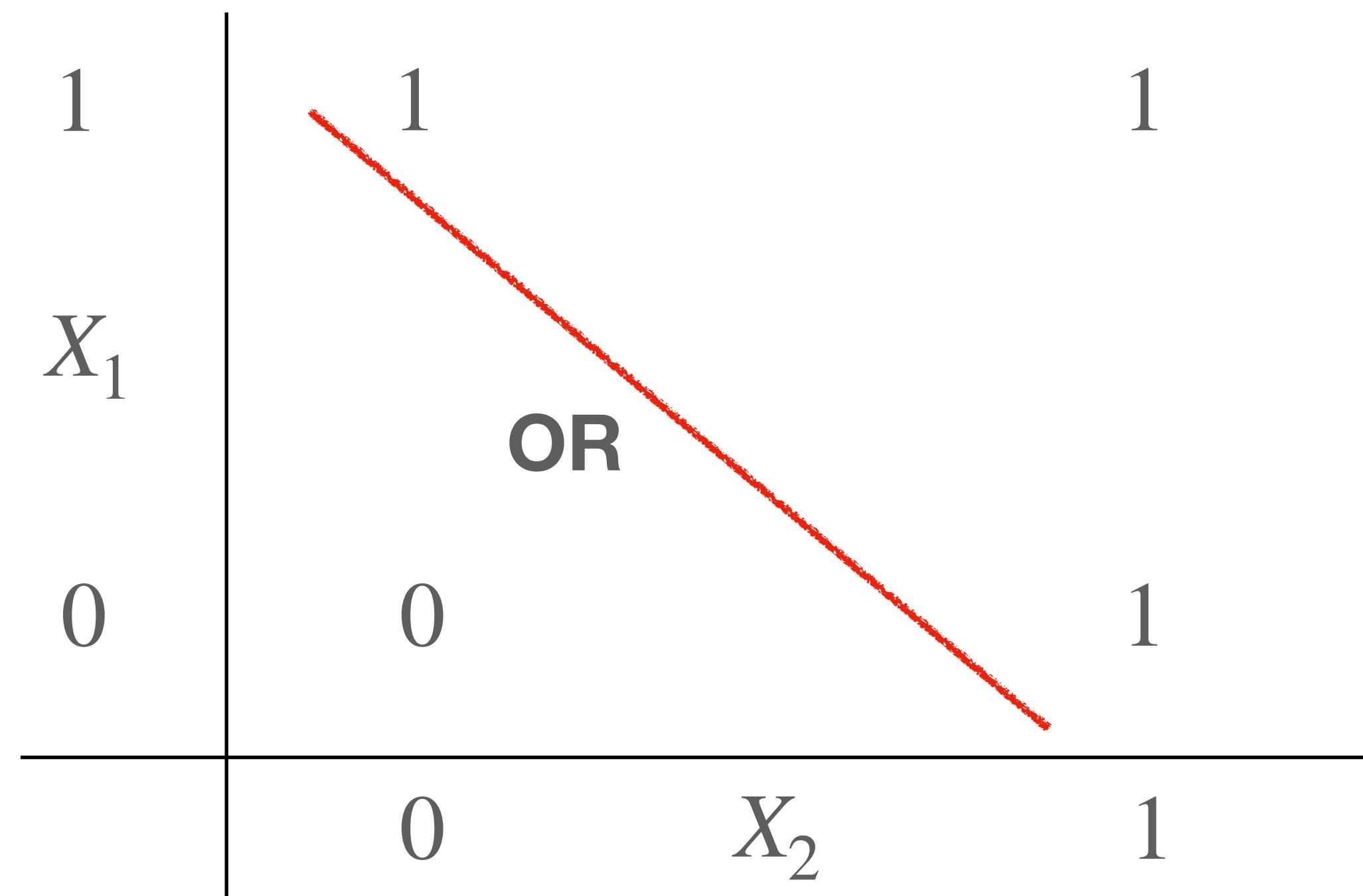
Ejemplo

Puerta lógica (AND)



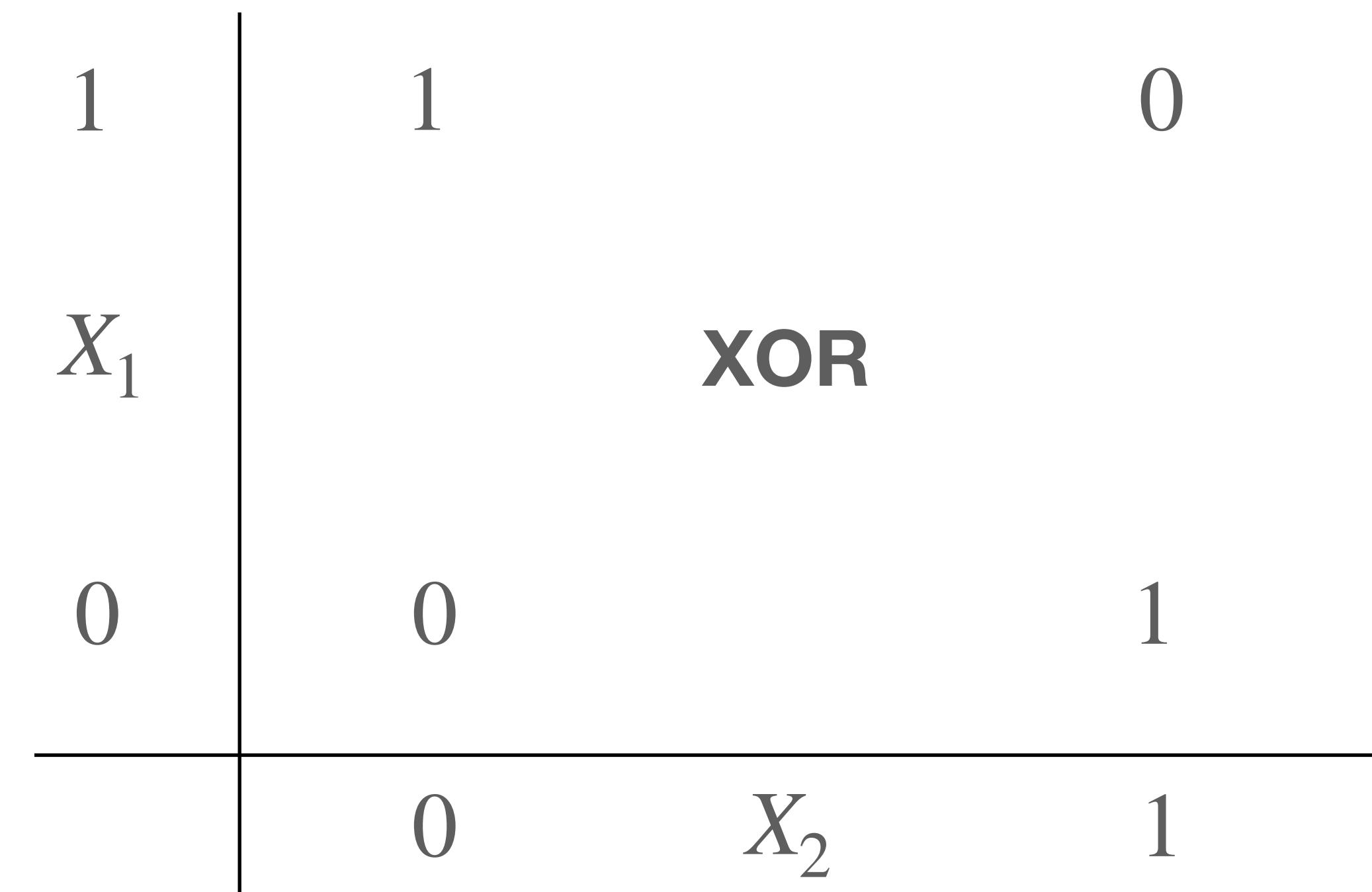
Ejemplo

Puerta lógica (AND)



Ejemplo

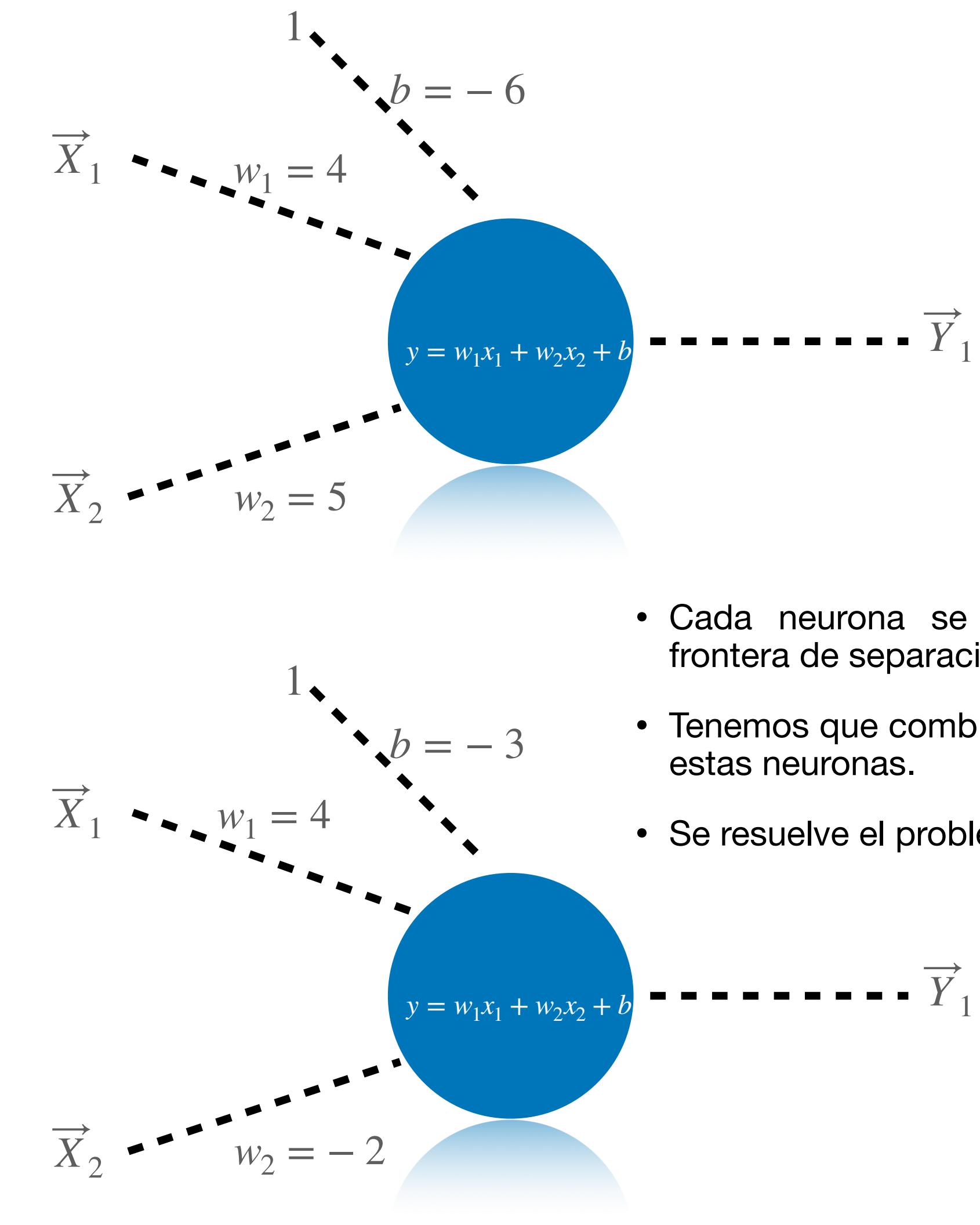
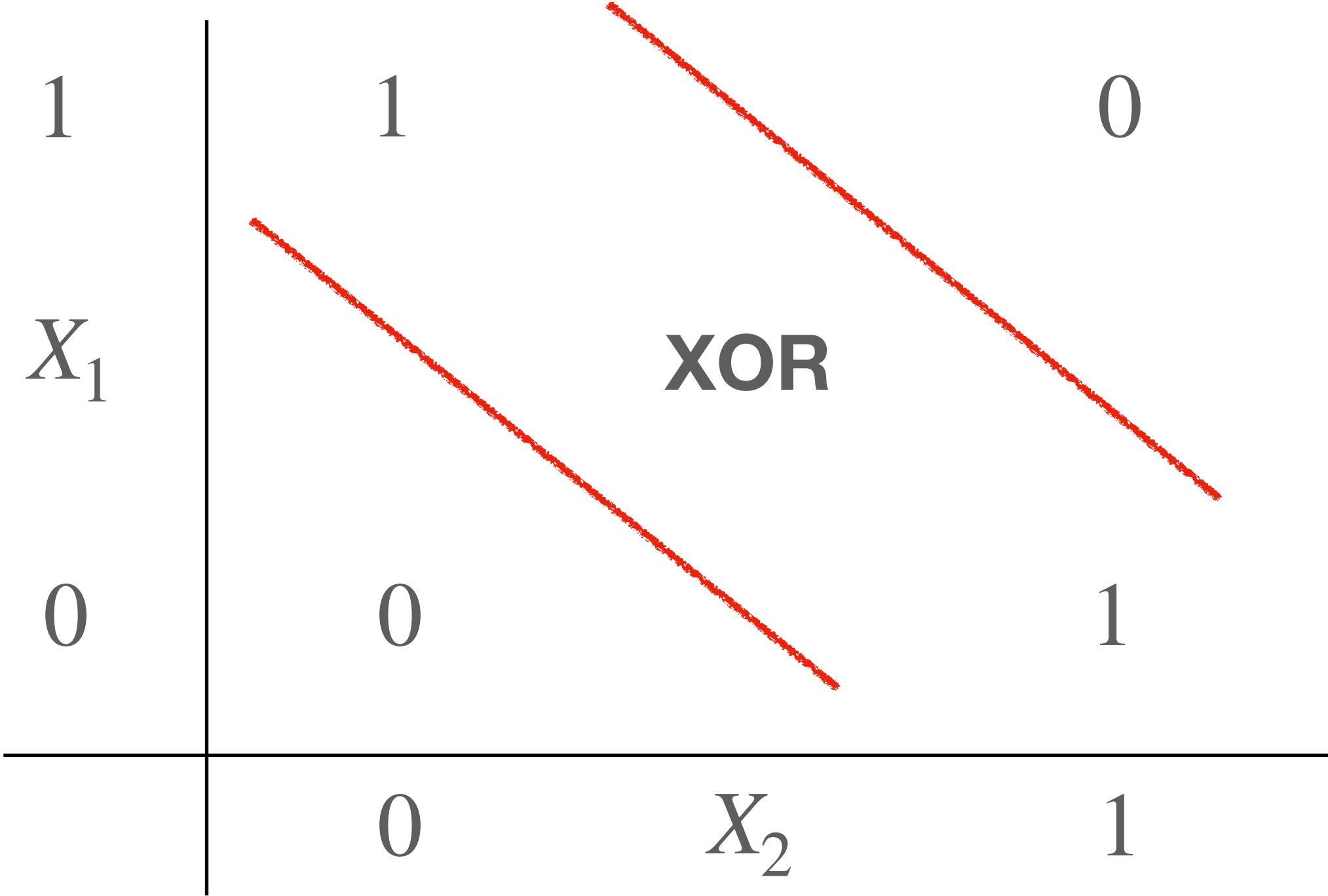
Puerta lógica (AND)



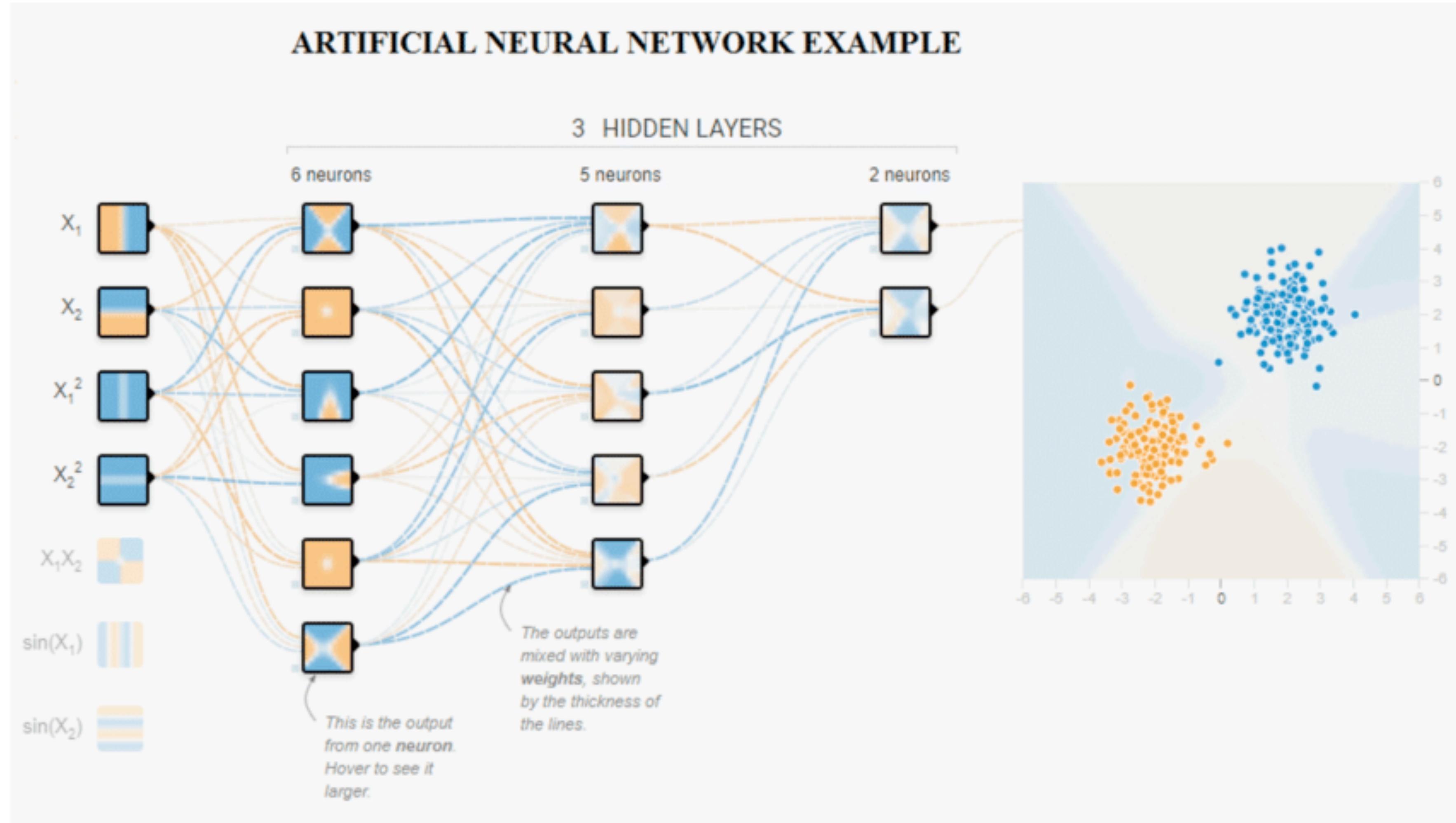
- En este caso no se puede trazar una línea de frontera para separar las clases.
- No es posible modelar el problema con una única neurona. No podemos separar linealmente estas clases.

Ejemplo

Puerta lógica (AND)

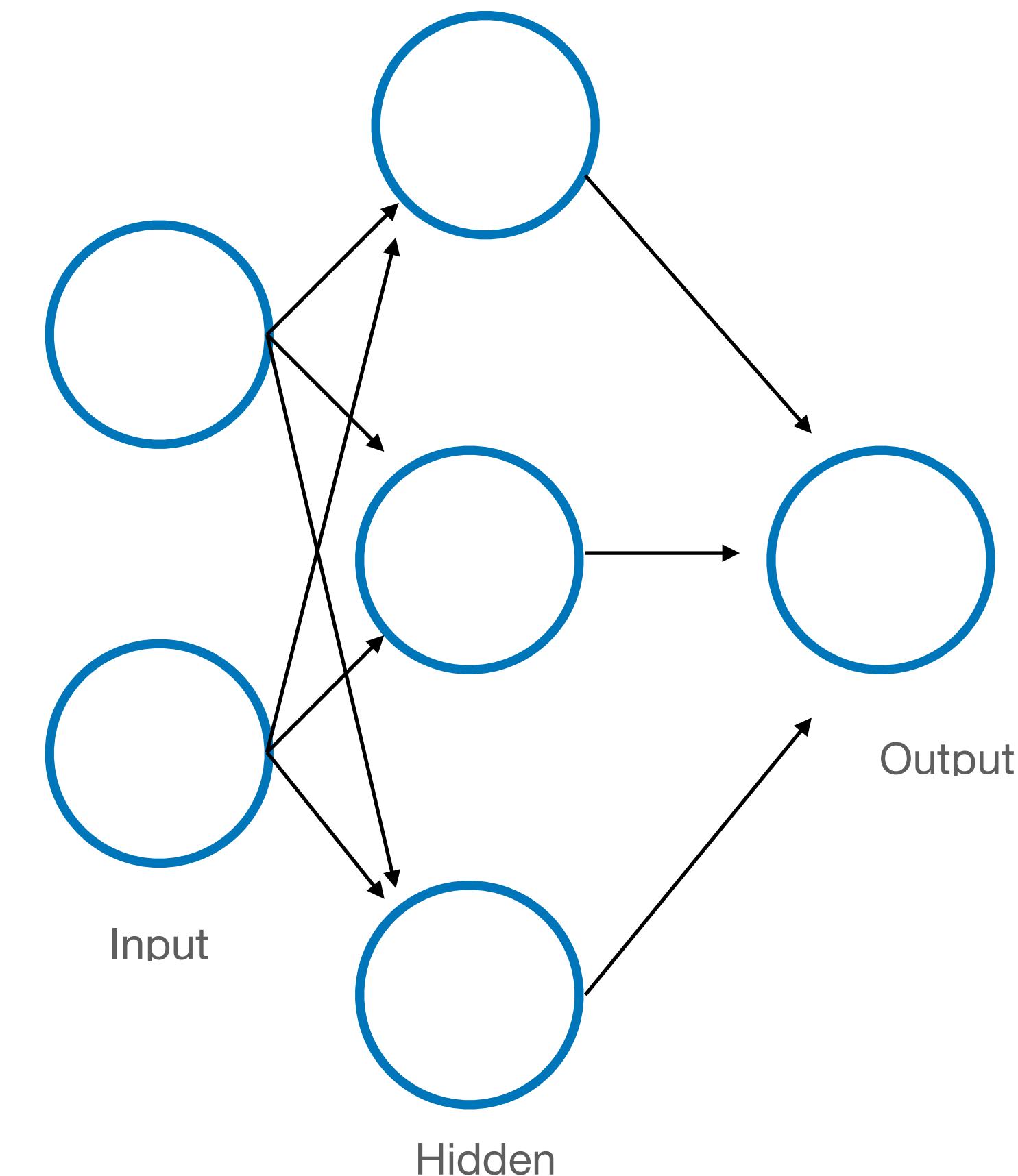


Arquitectura de redes neuronales



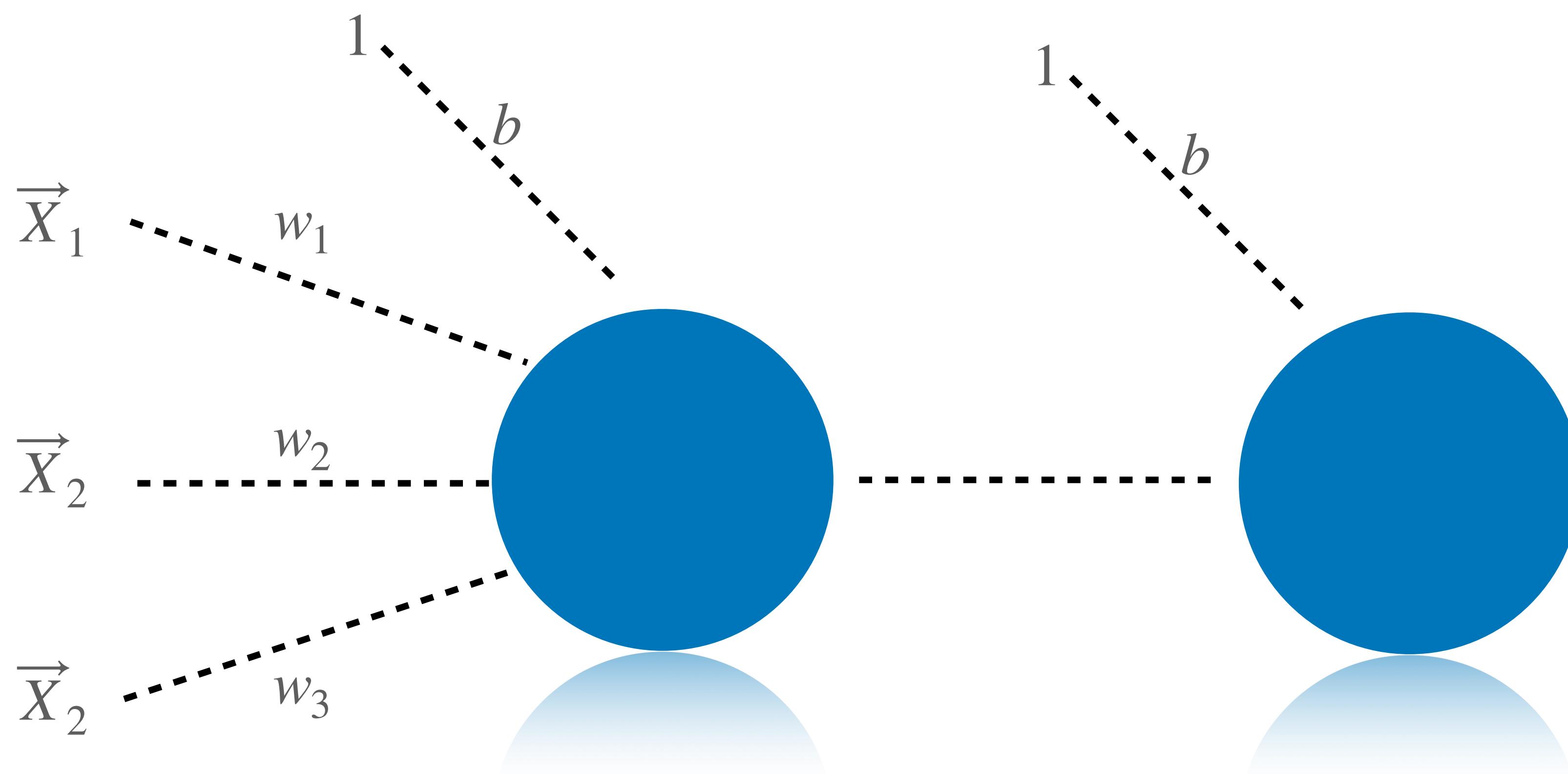
Multilayer Perceptron

- Una sola neurona no puede representar toda la información.
- Necesitamos una agrupación de éstas.
- Se pueden tener múltiples salidas.



Multilayer Perceptron

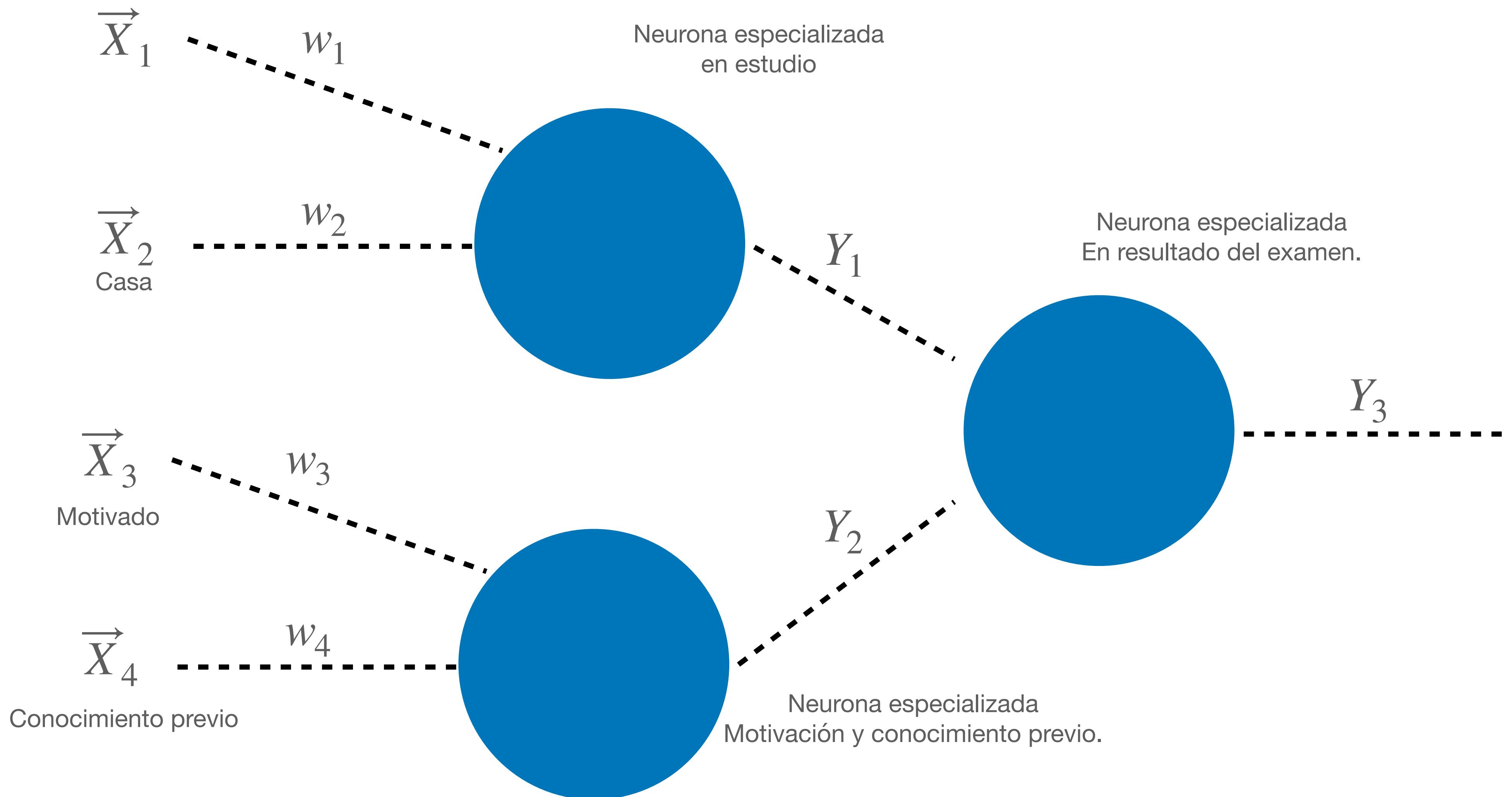
Neuronas en secuencia



- Si tenemos neuronas en forma secuencial, tenemos un aprendizaje ordenado (jerárquico).
- Cada capa "aprende" estructuras más complejas.

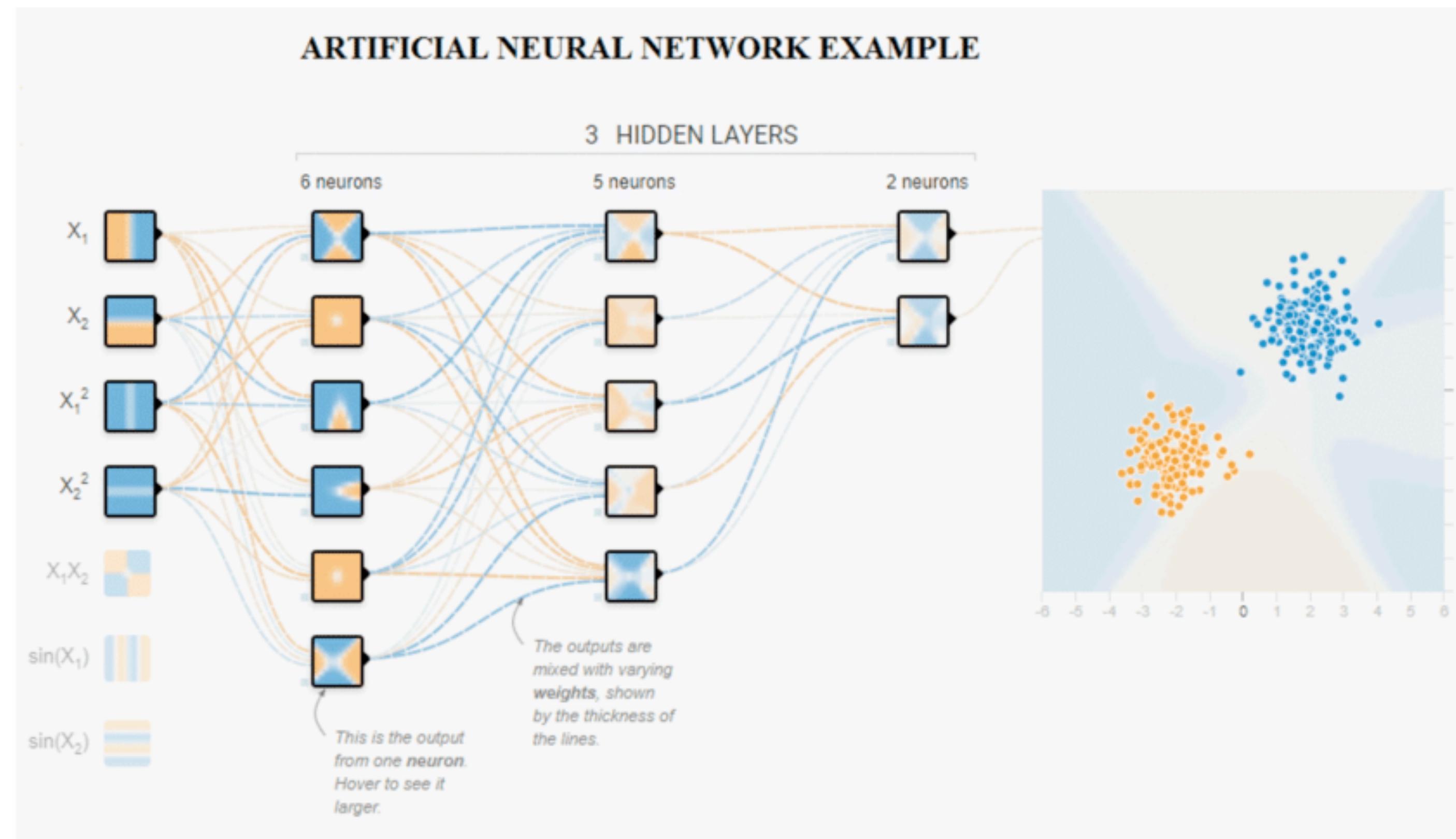
Multilayer Perceptron

Ejemplo

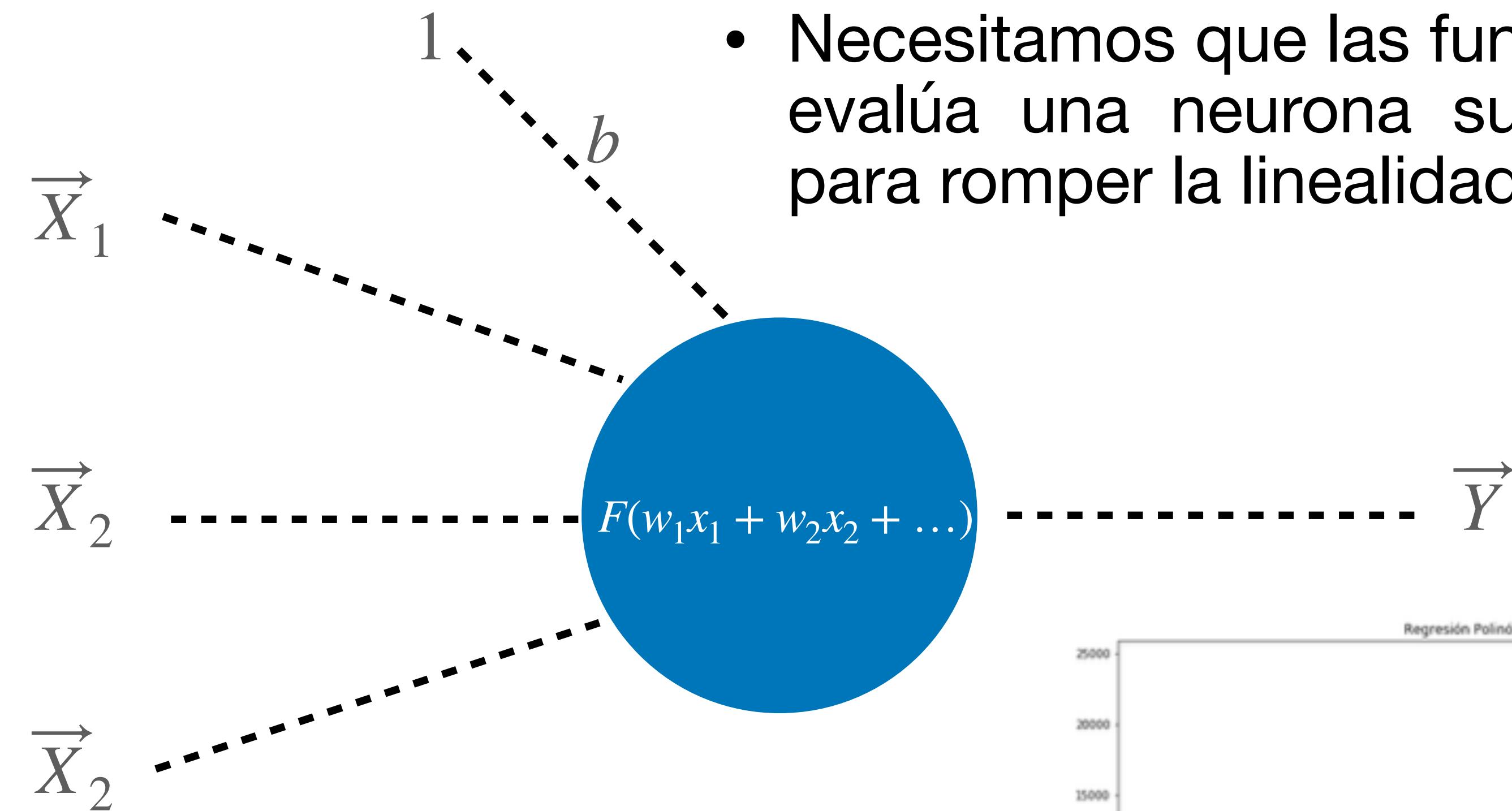


Funciones de activación

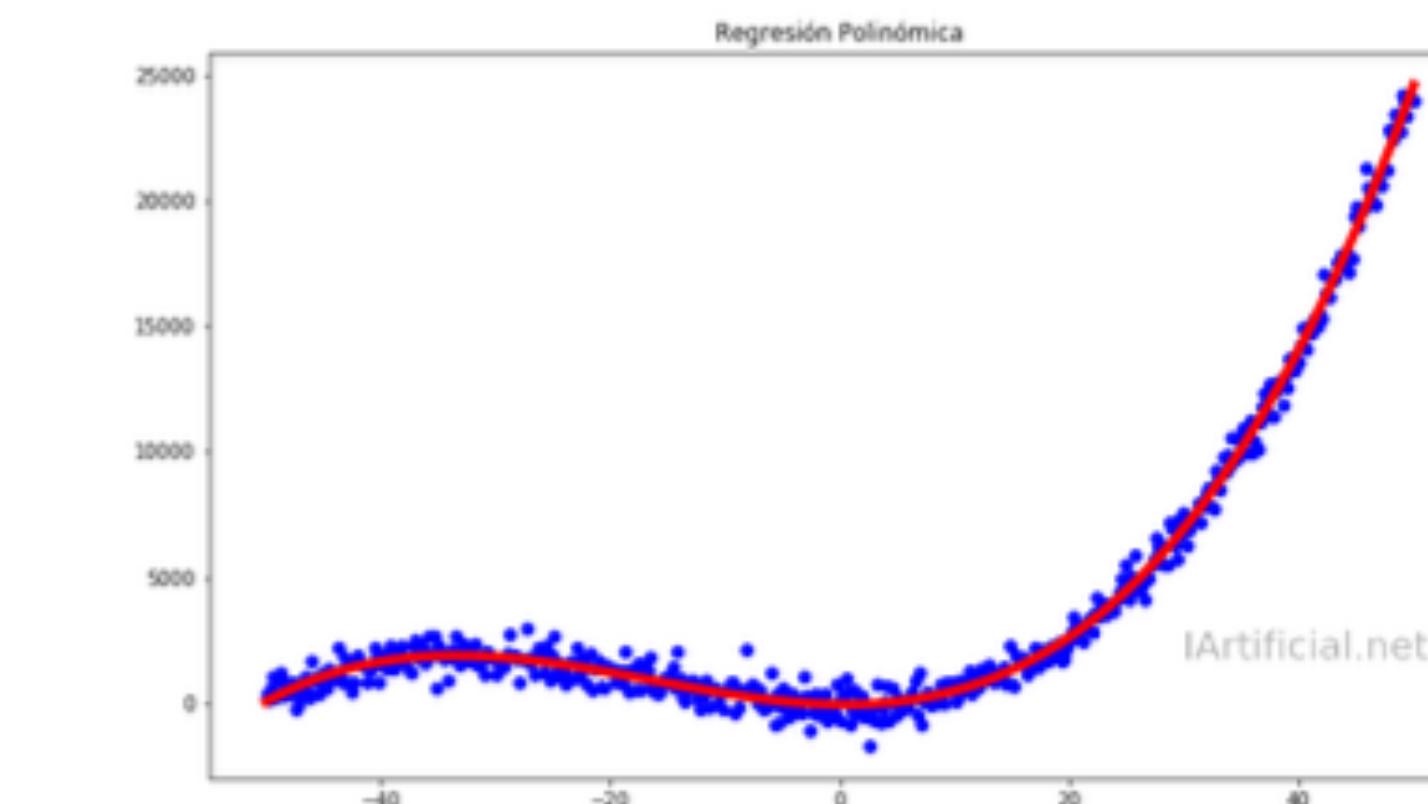
- Necesitamos que las funciones en con las cuales evalúa una neurona sufren de una distorsión, para romper la linealidad del problema.



Funciones de activación



- Necesitamos que las funciones en con las cuales evalúa una neurona sufran de una distorsión, para romper la linealidad del problema.

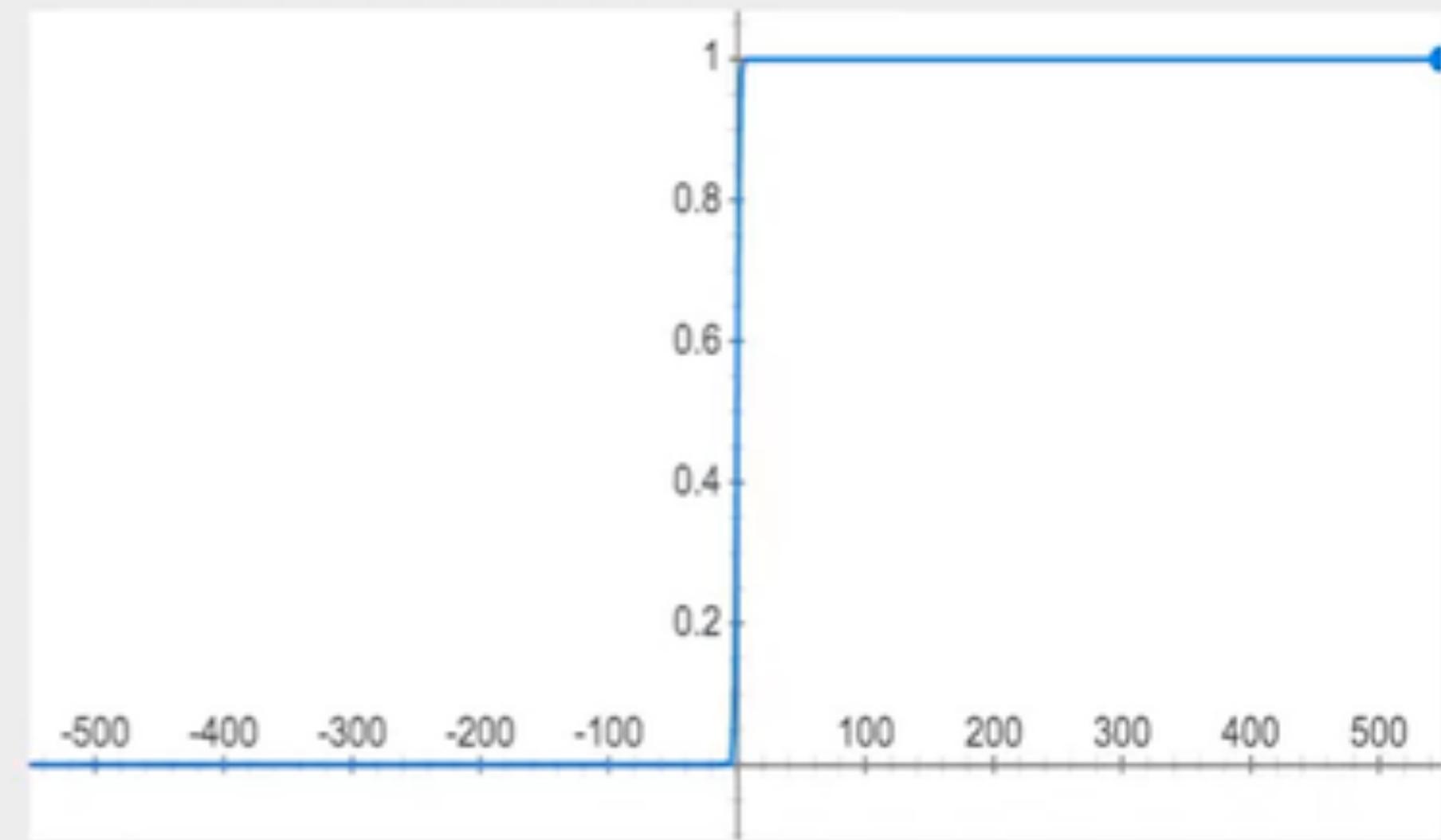


Funciones de activación

Binaria

Binary Step

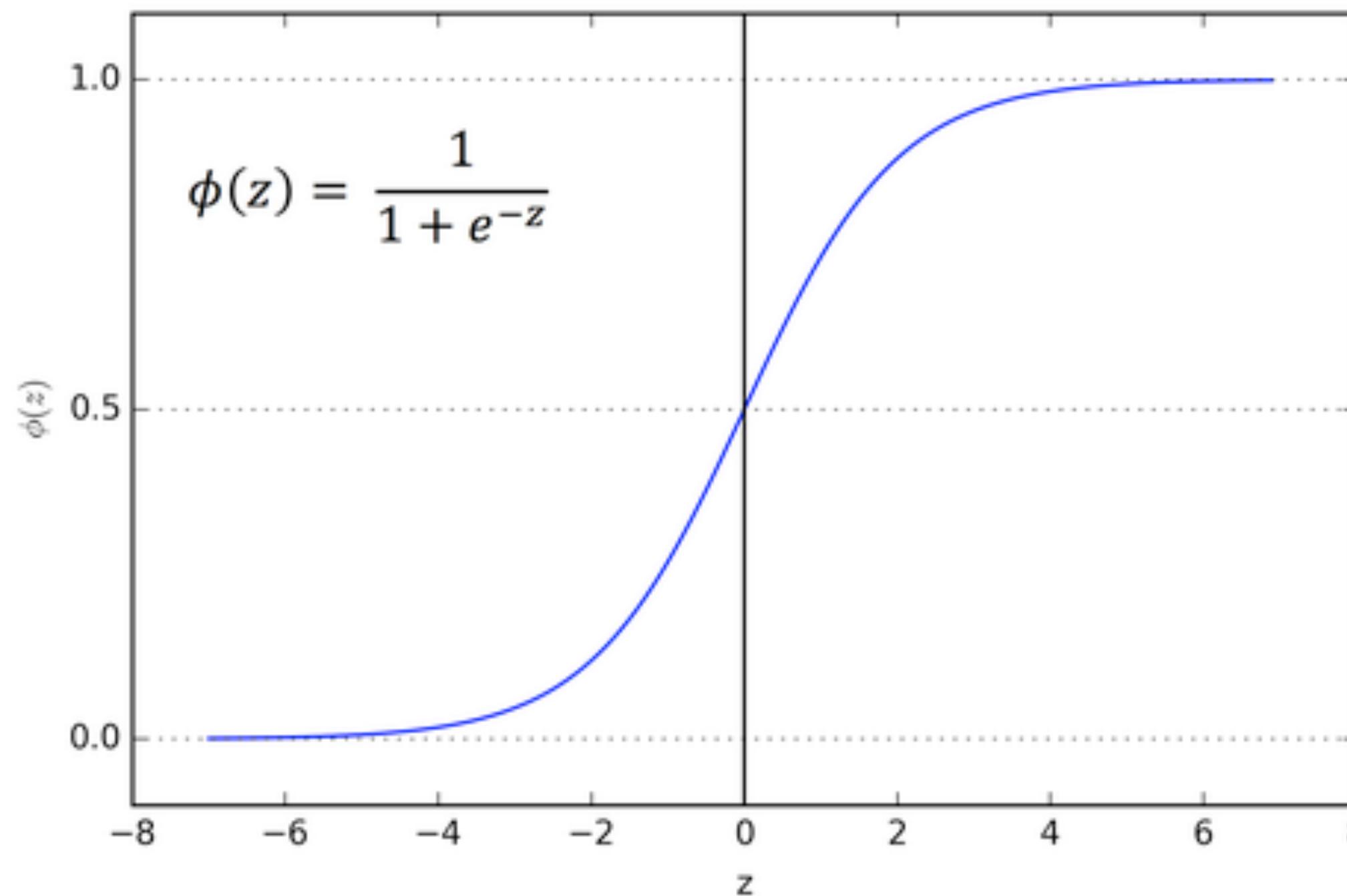
$$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{For } x \geq 0 \end{cases}$$



$$F(WX) \rightarrow Y = \{0,1\}$$

Funciones de activación

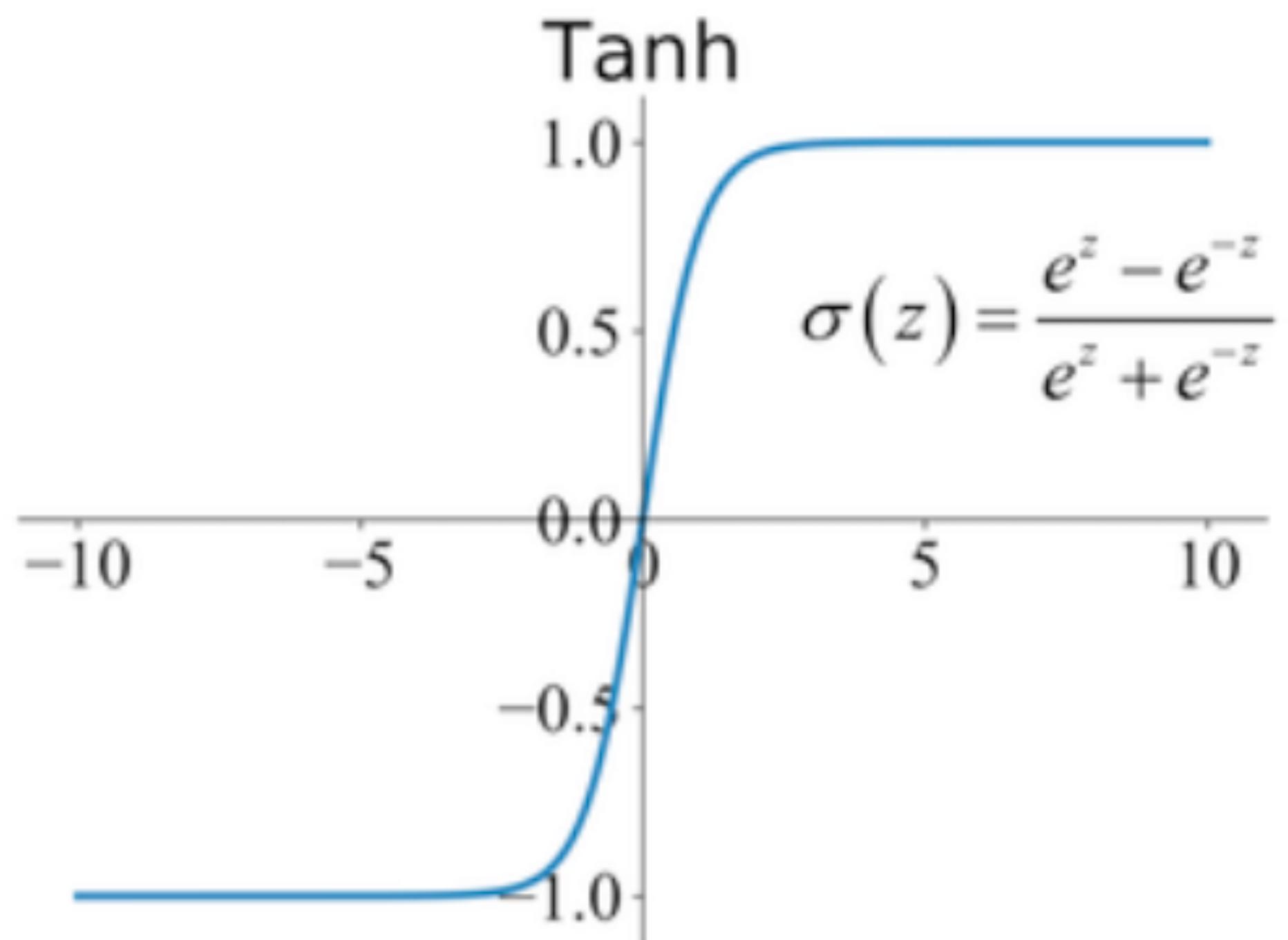
Sigmoid



- Esta función distorsiona la señal pero no tenemos sólo 1 ó 0, sino valores intermedios.
- Los valores muy grandes se saturan en 1.
- Los valores muy pequeños se saturan en 0.
- Nos permite representar probabilidades entre 0 y 1.

Funciones de activación

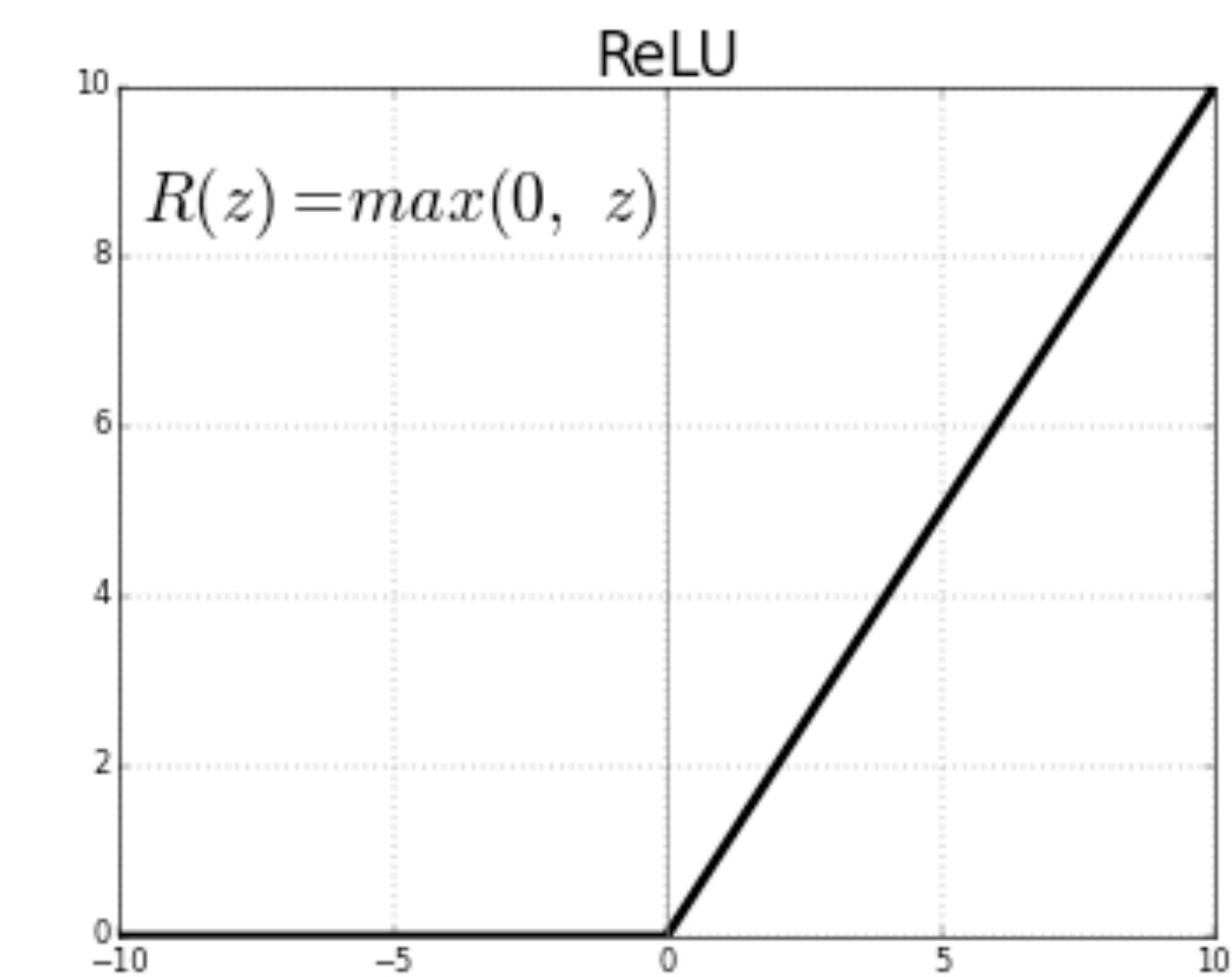
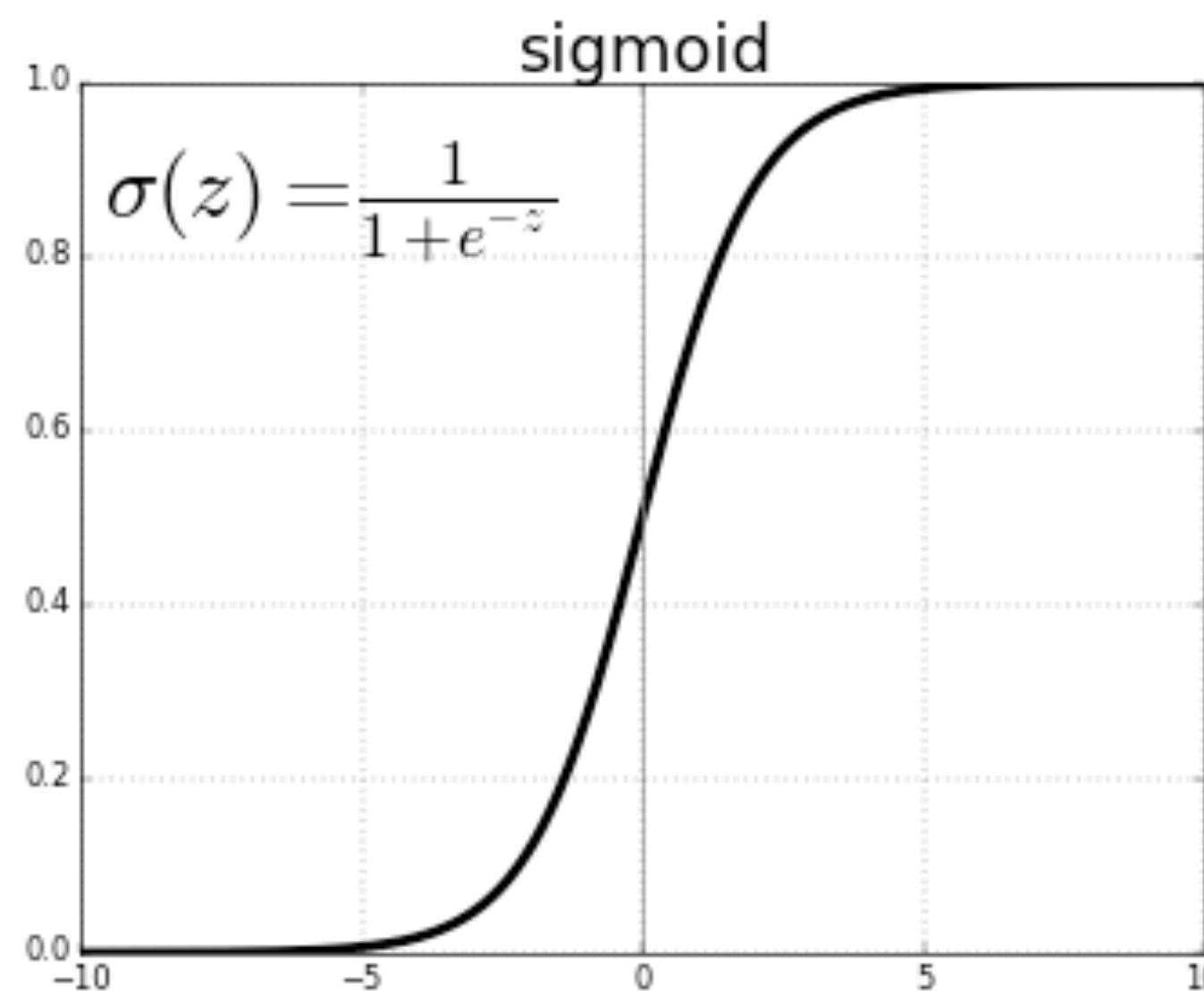
Tanh



- Muy parecida a la sigmoidal pero va de -1 a 1.

Funciones de activación

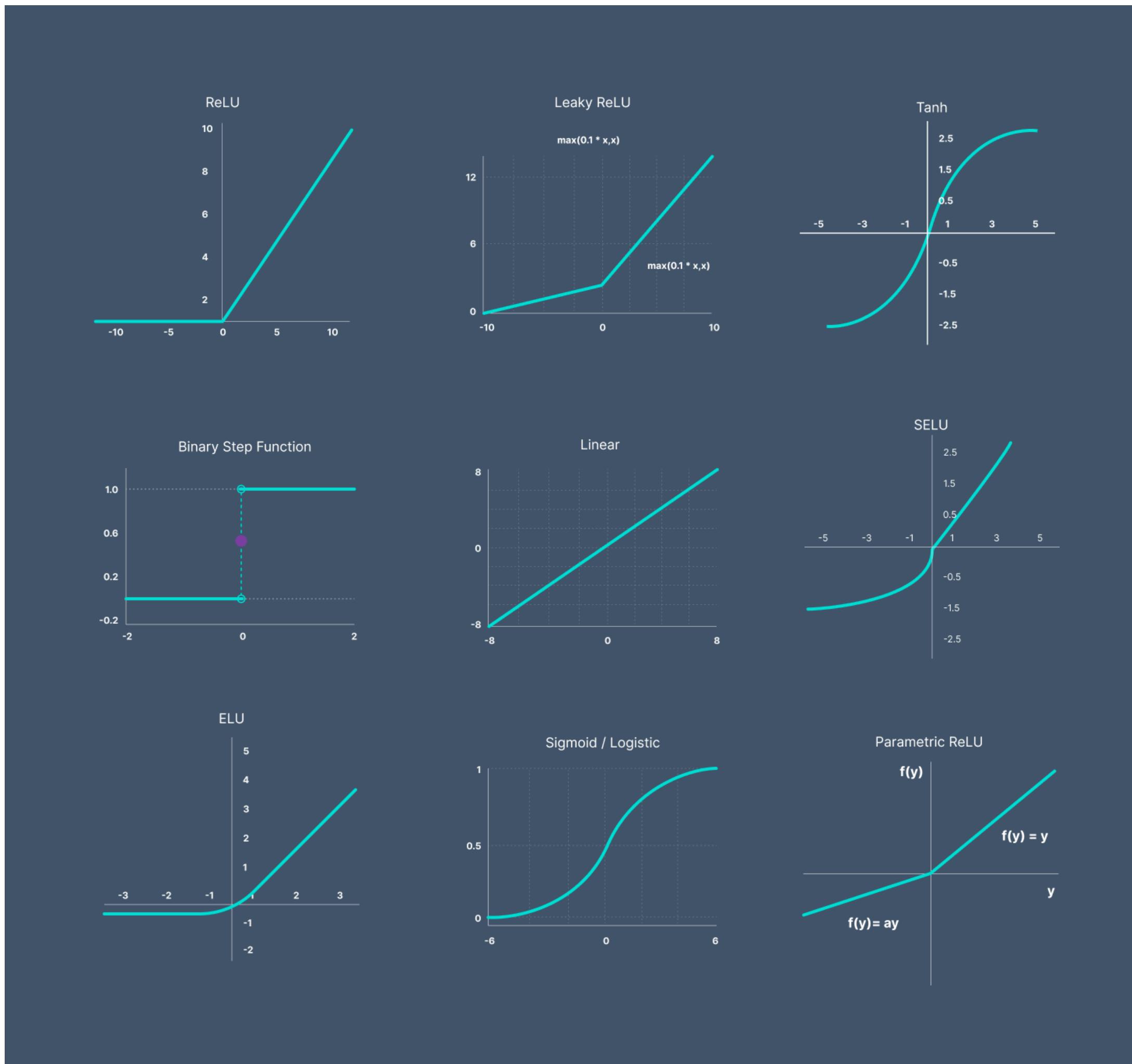
Rectified Linear Unit (ReLU)



- Muy utilizada por tener resultados buenos y un aprendizaje rápido.
- Es una función lineal cuando es positiva la entrada.
- Es igual a 0 cuando el valor de entrada es negativo.

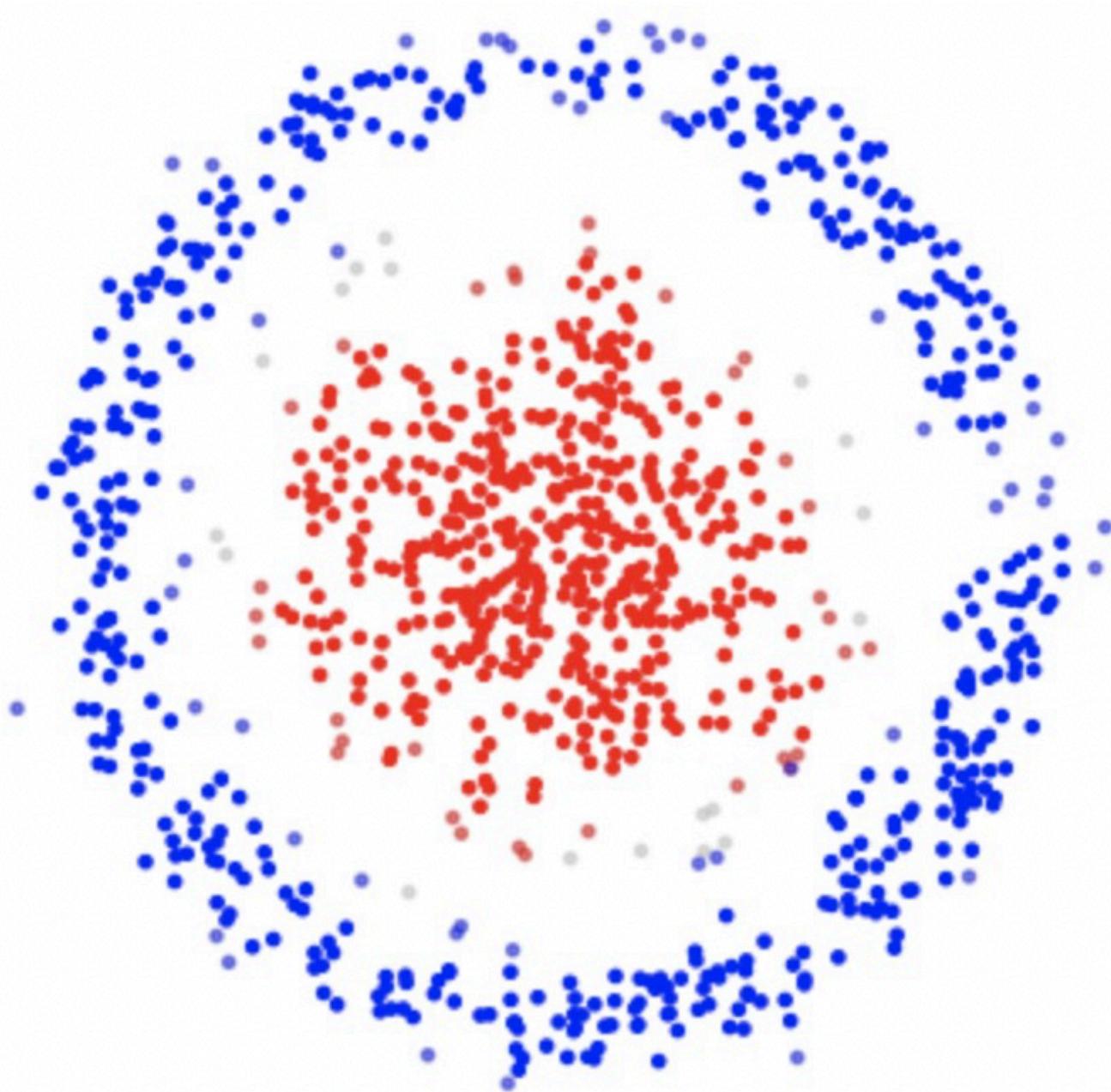
Funciones de activación

Otras



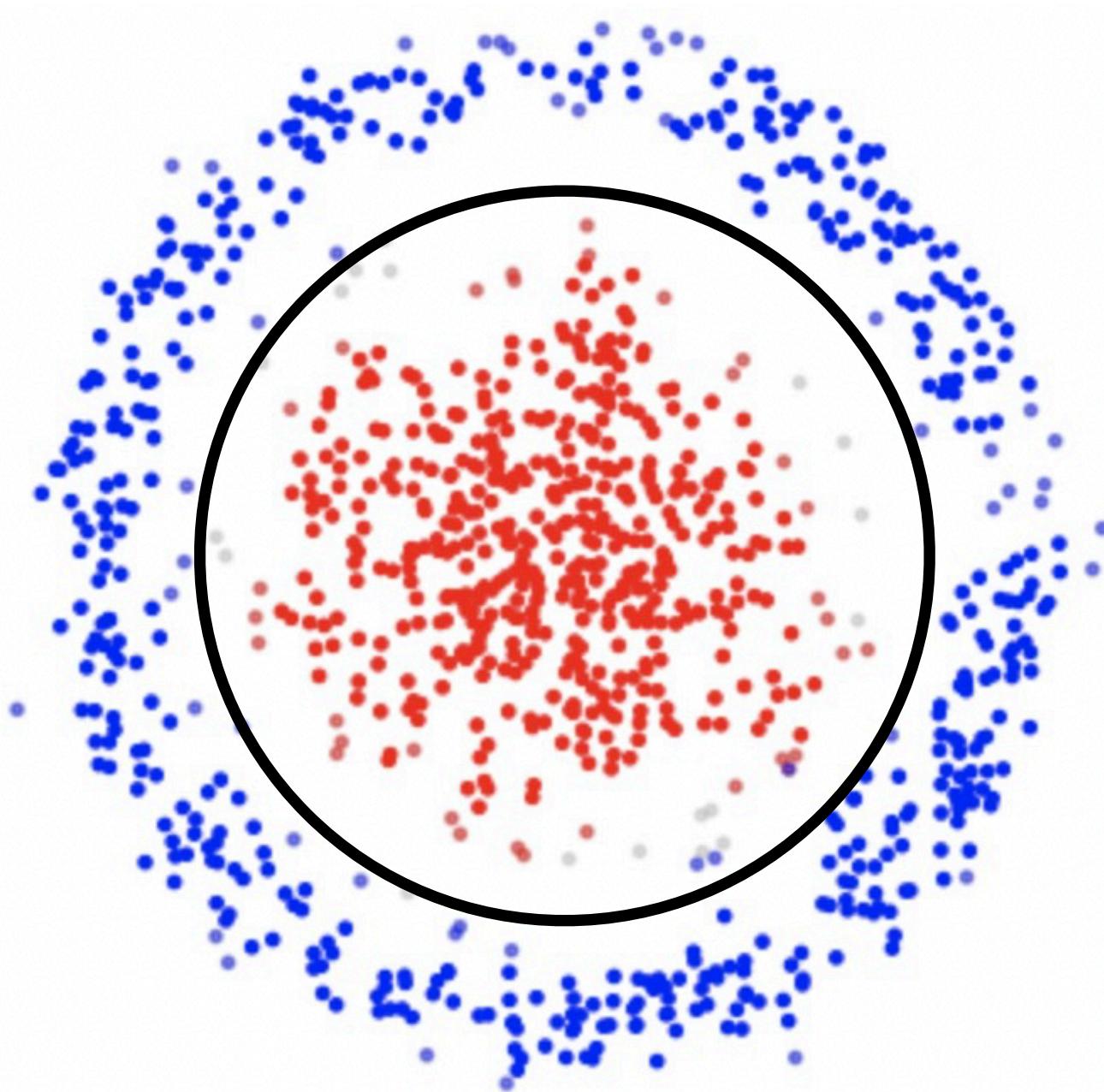
Multilayer Perceptron

Hiperplano



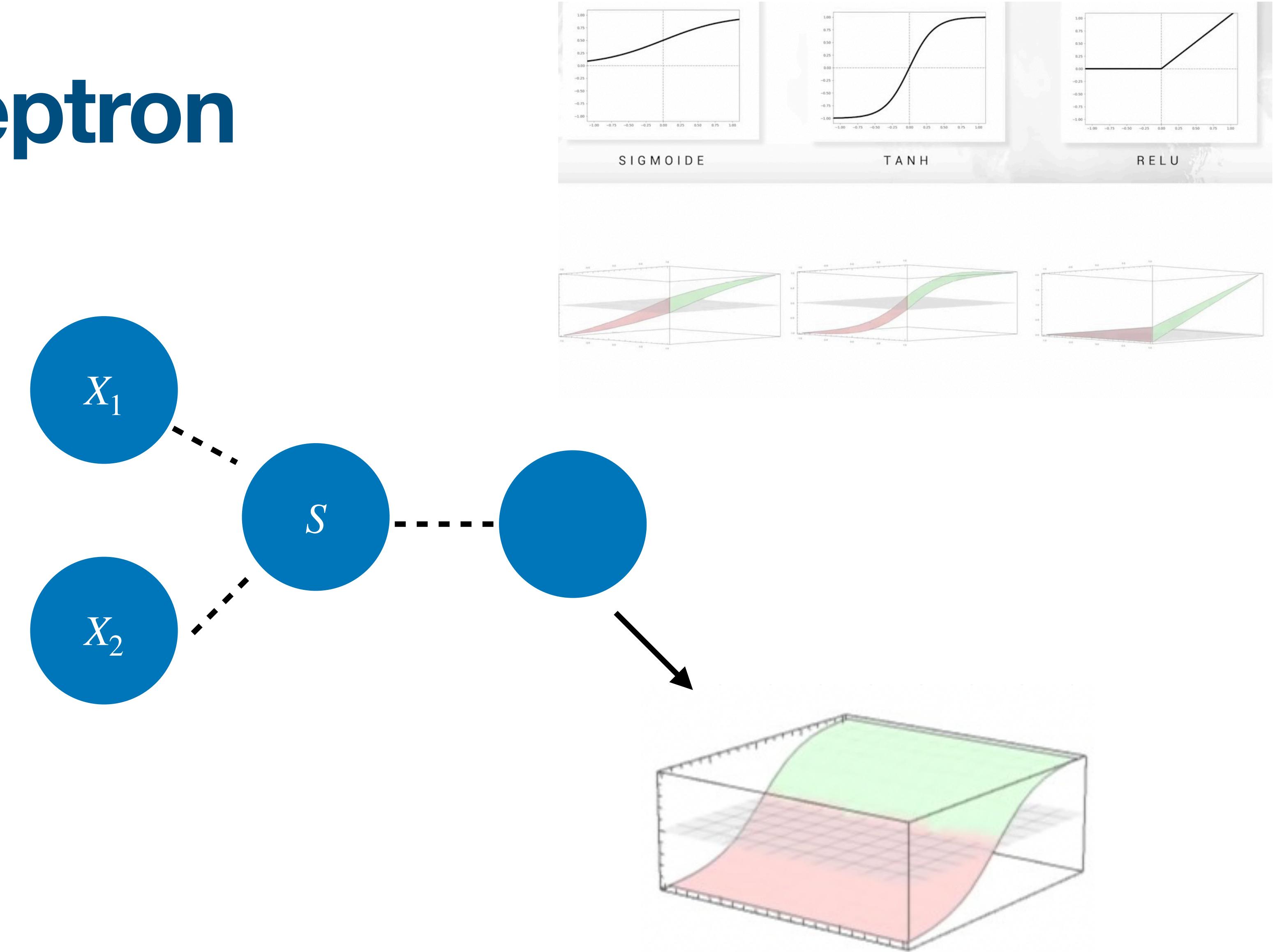
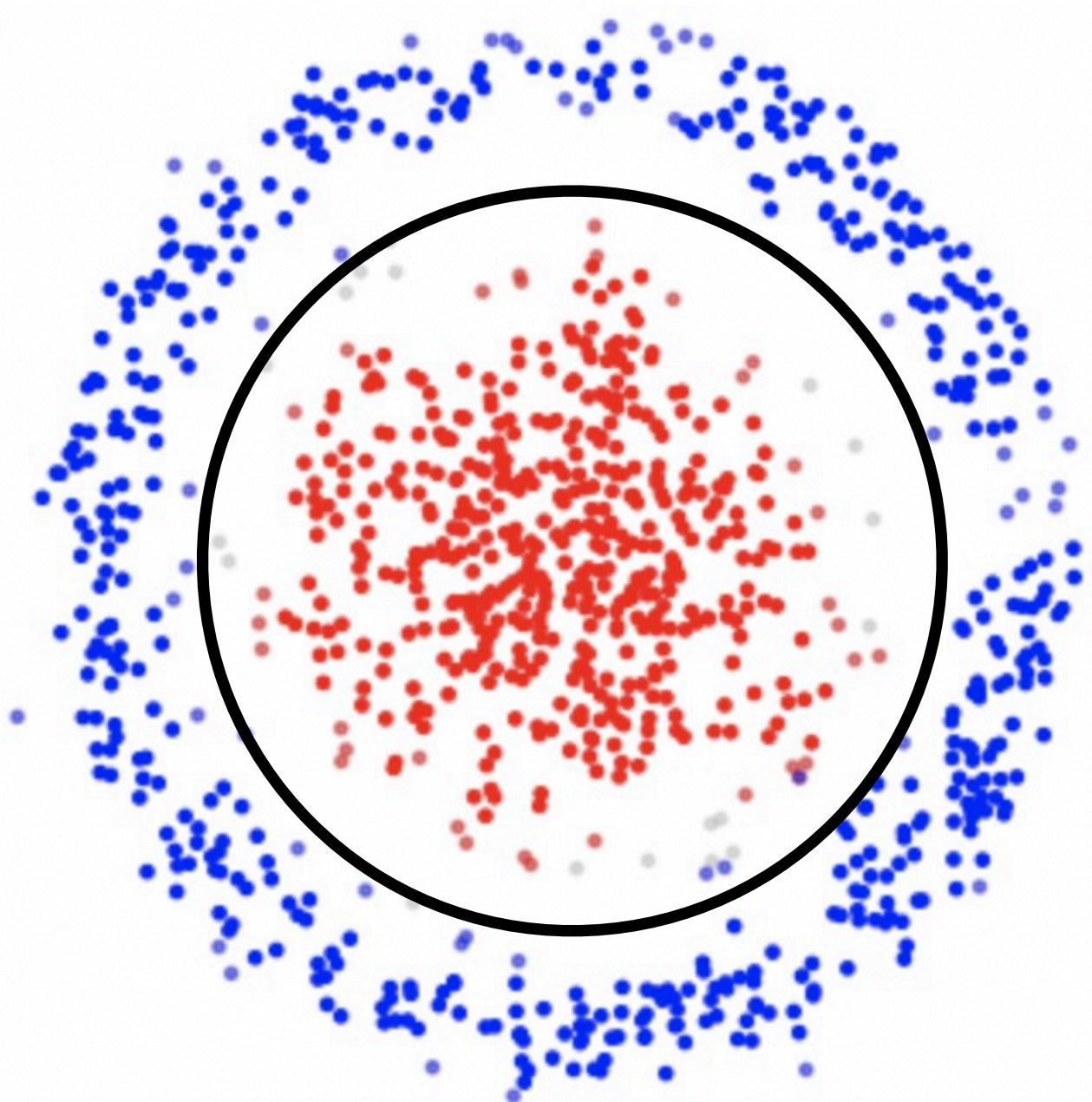
Multilayer Perceptron

Hiperplano



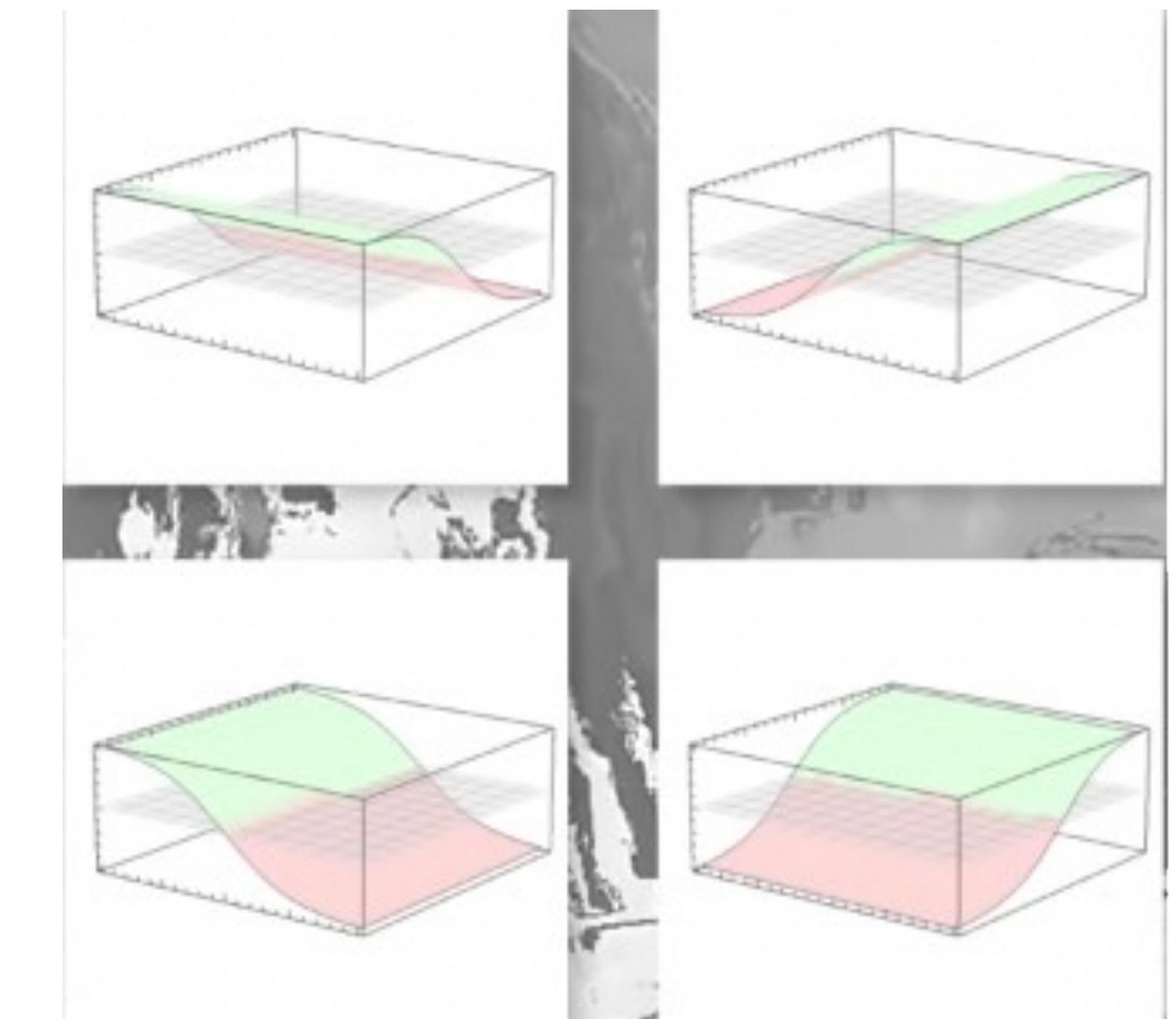
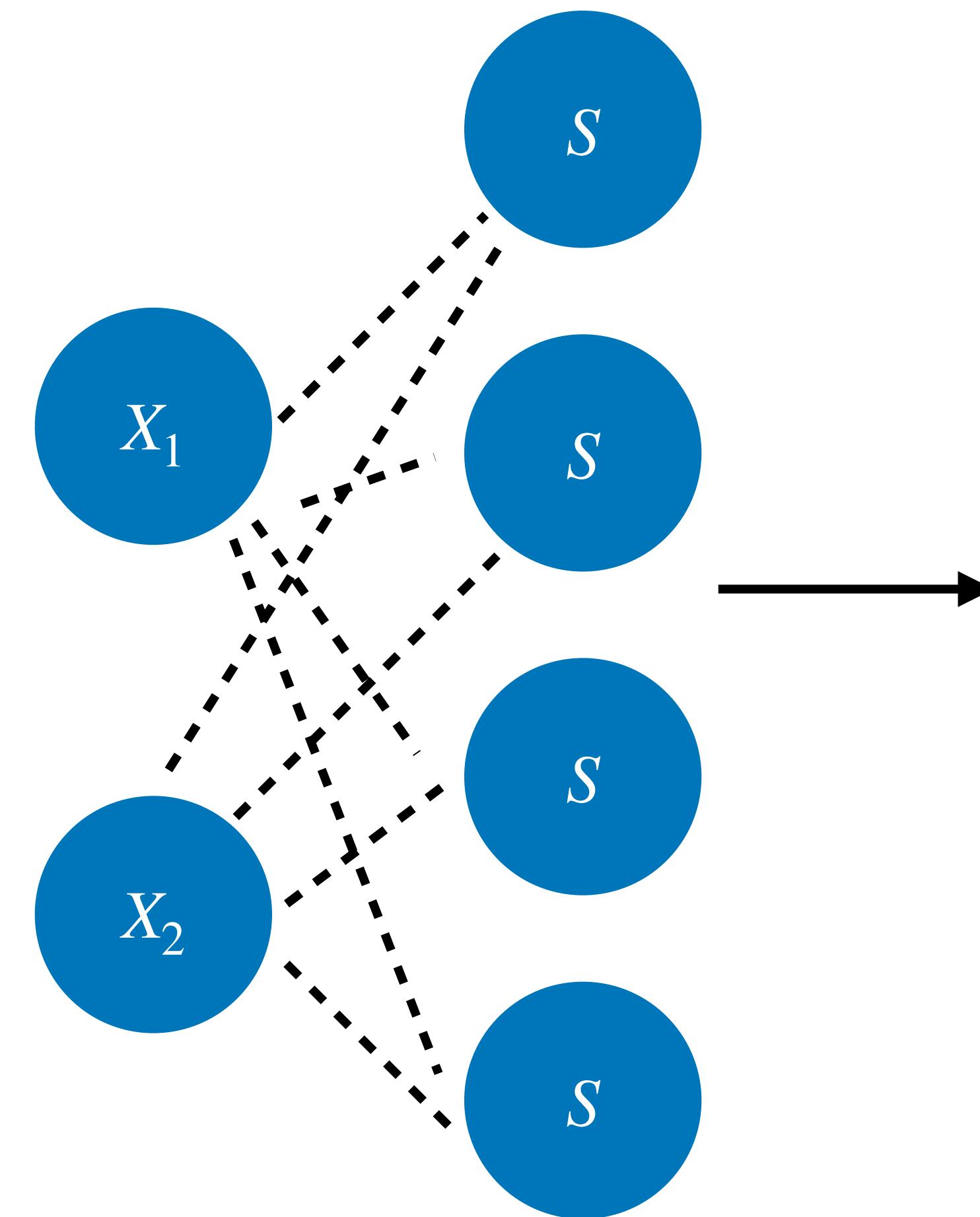
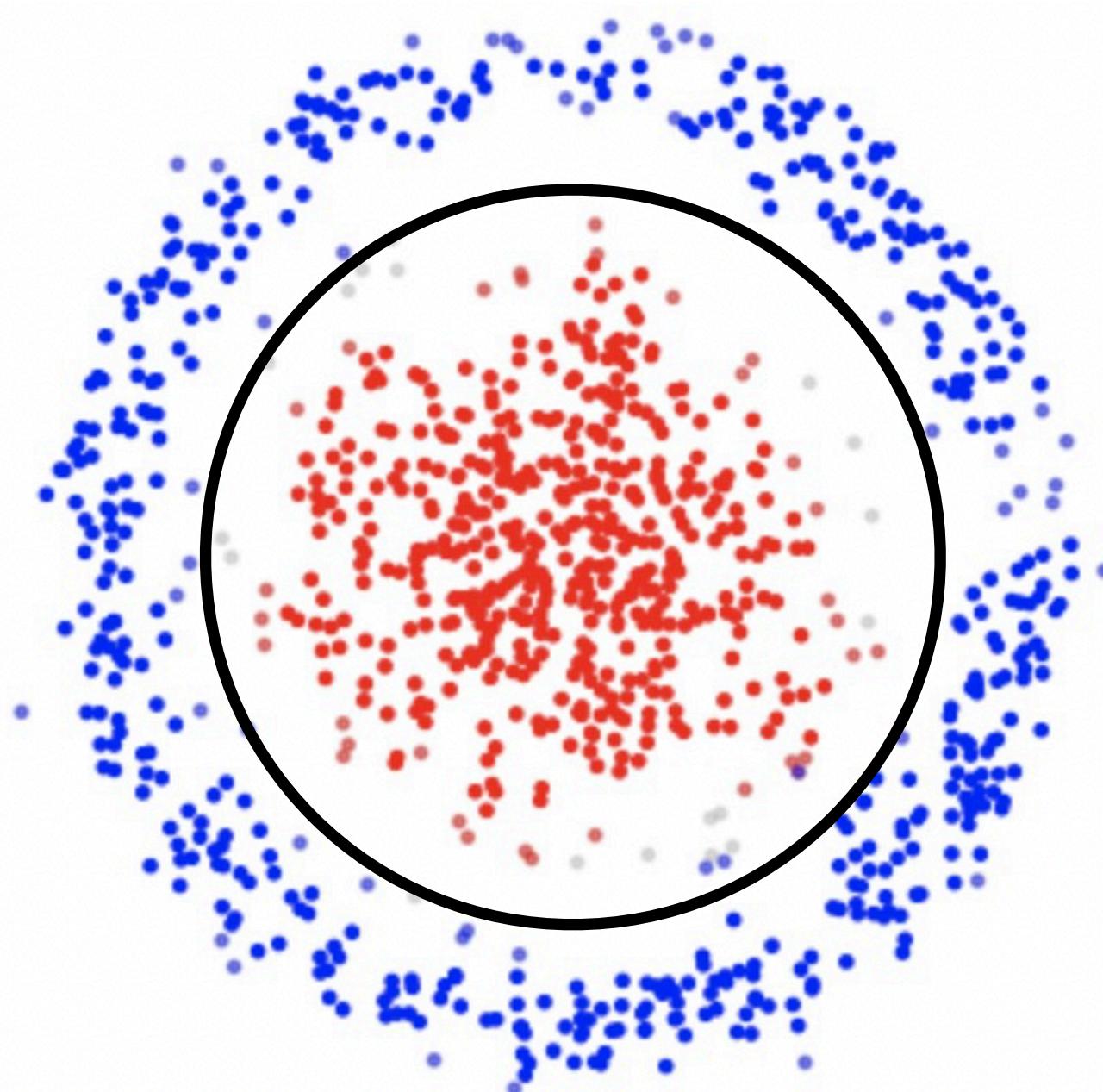
Multilayer Perceptron

Hiperplano



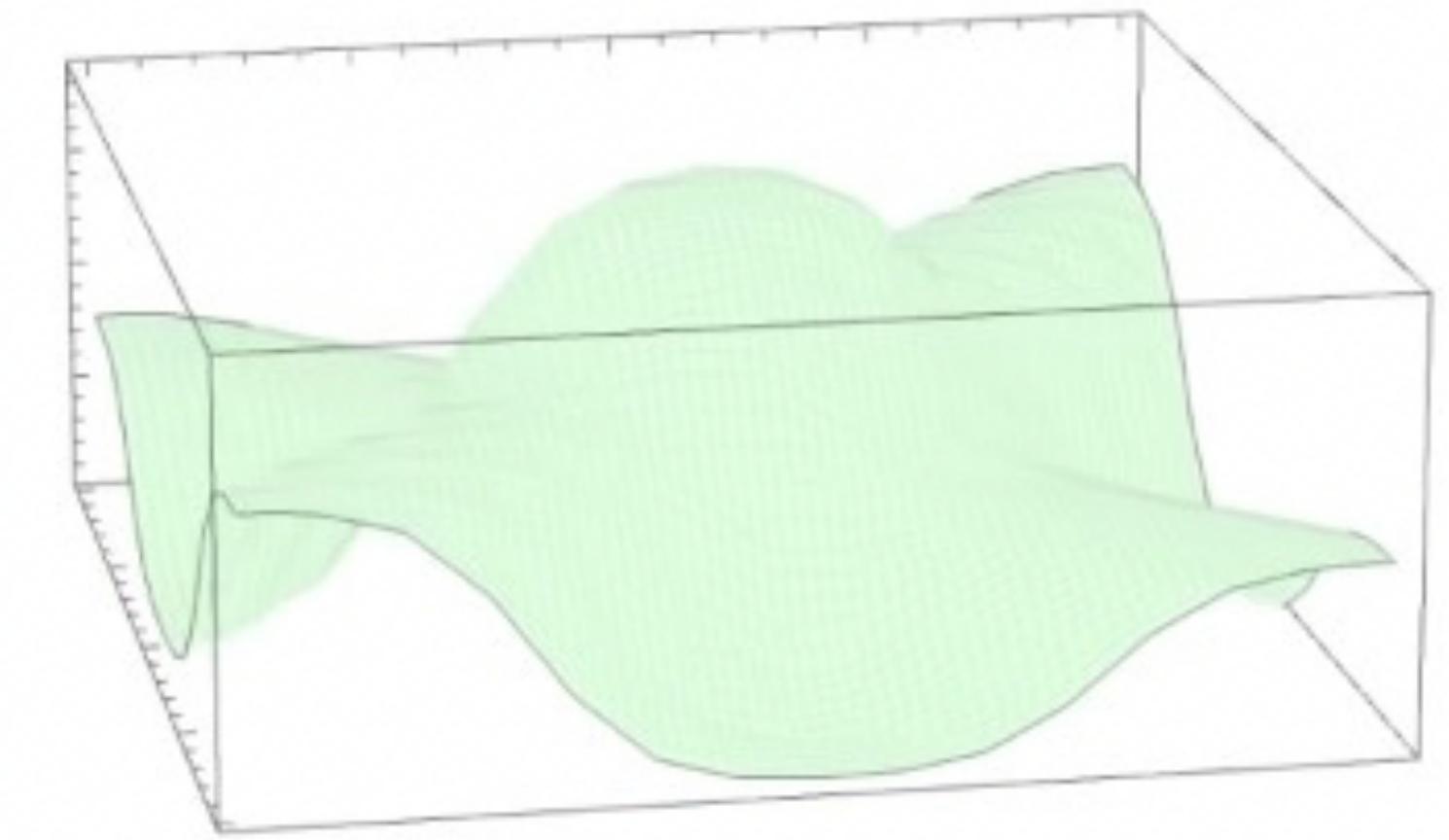
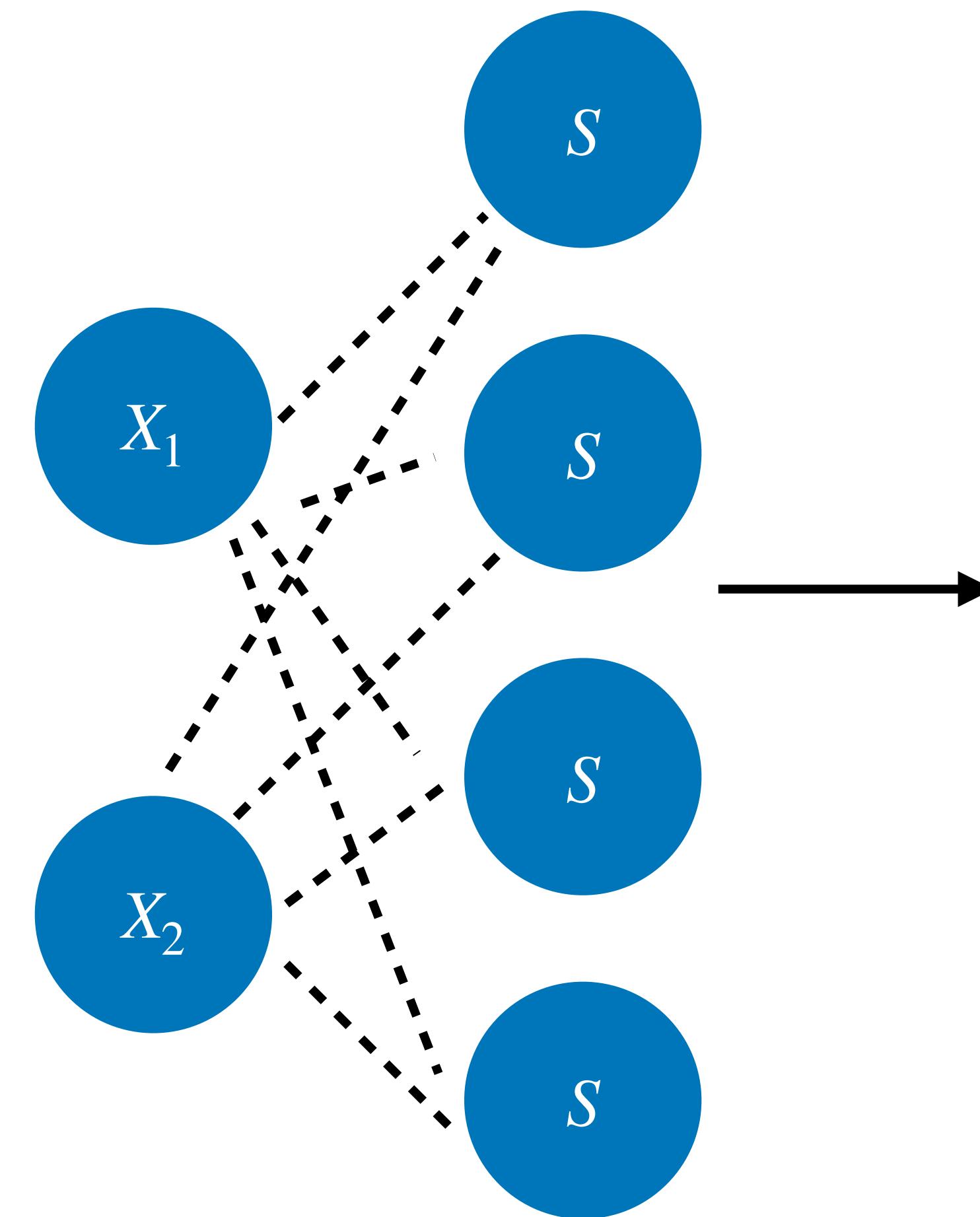
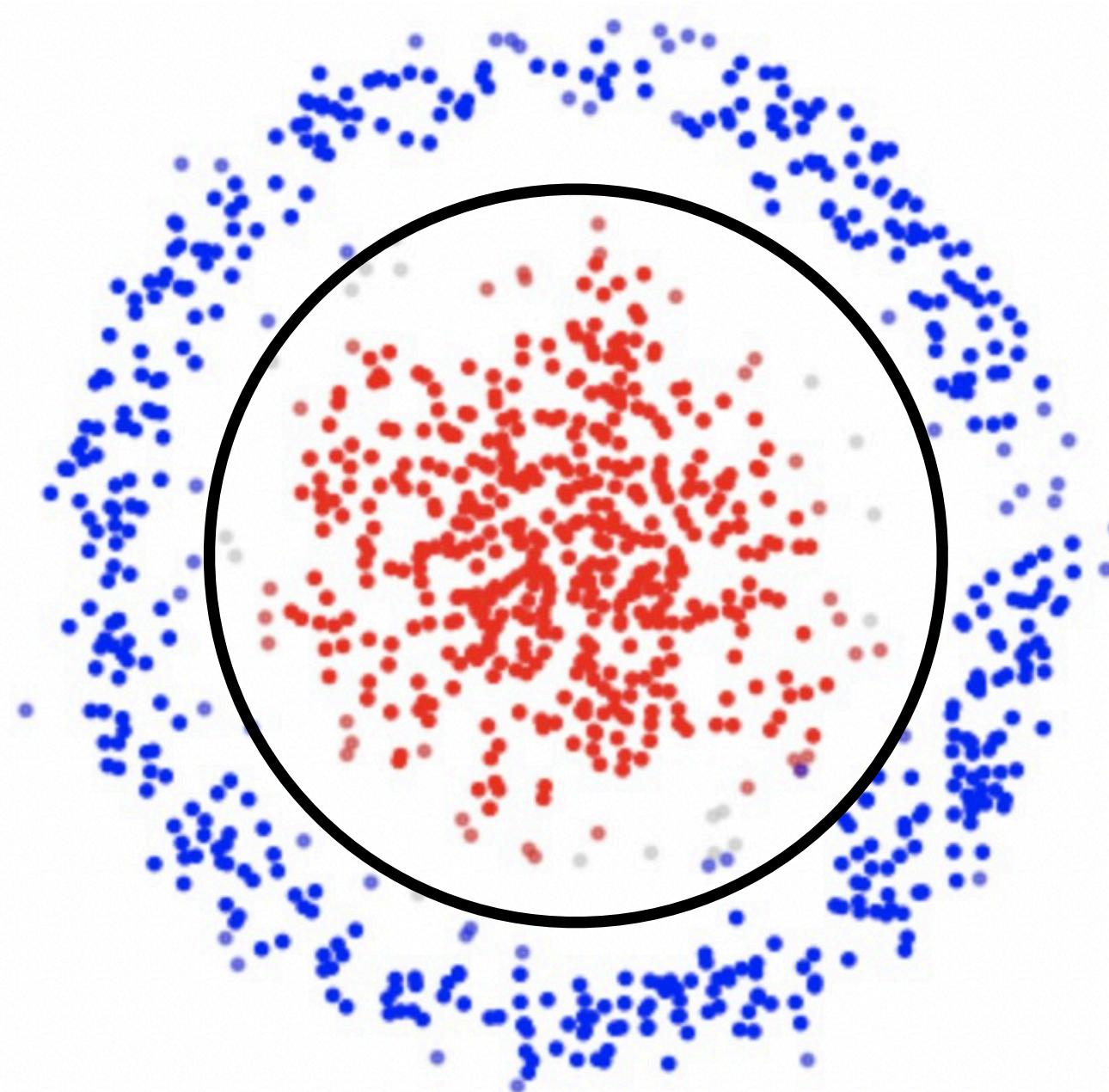
Multilayer Perceptron

Hiperplano



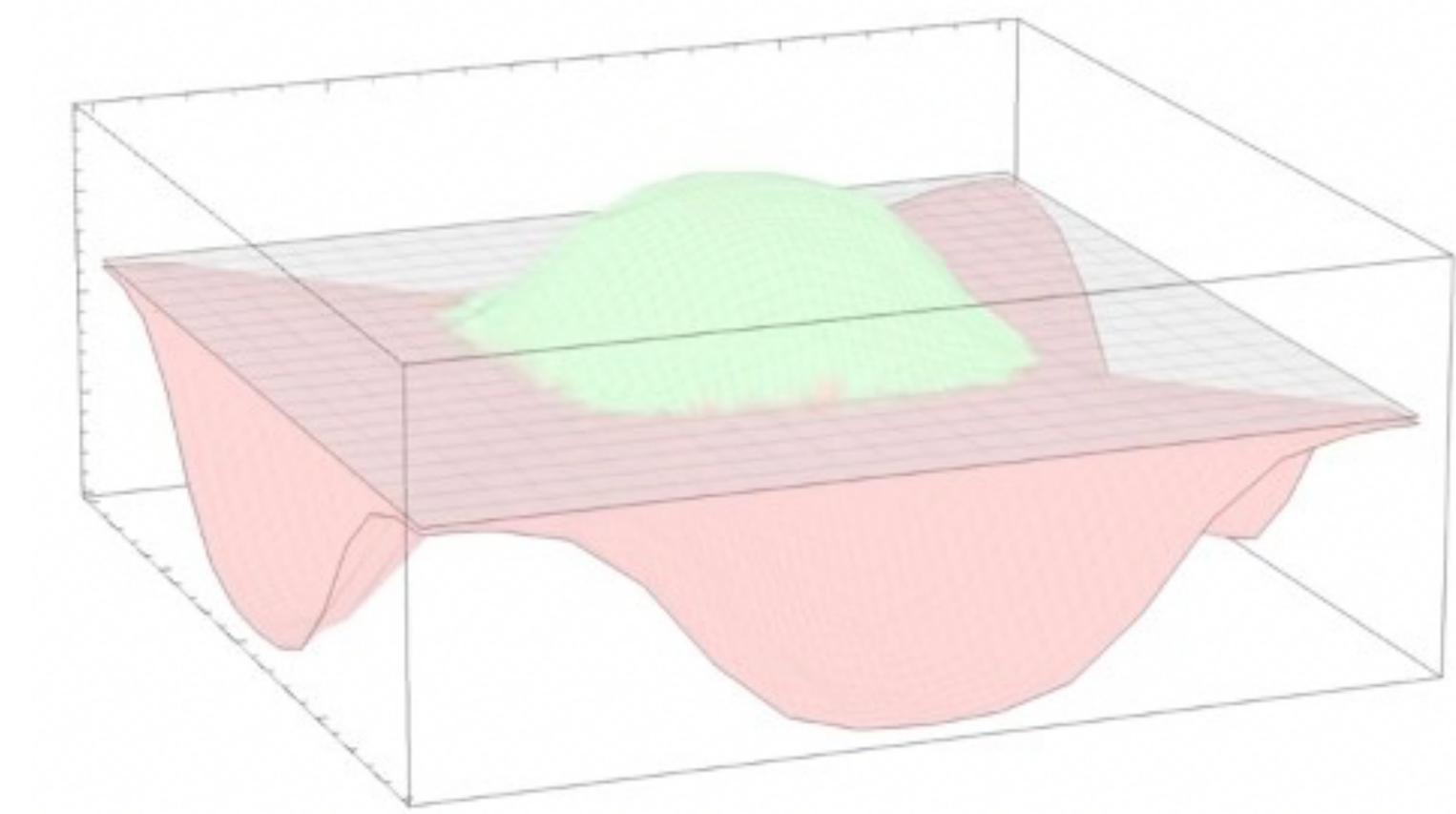
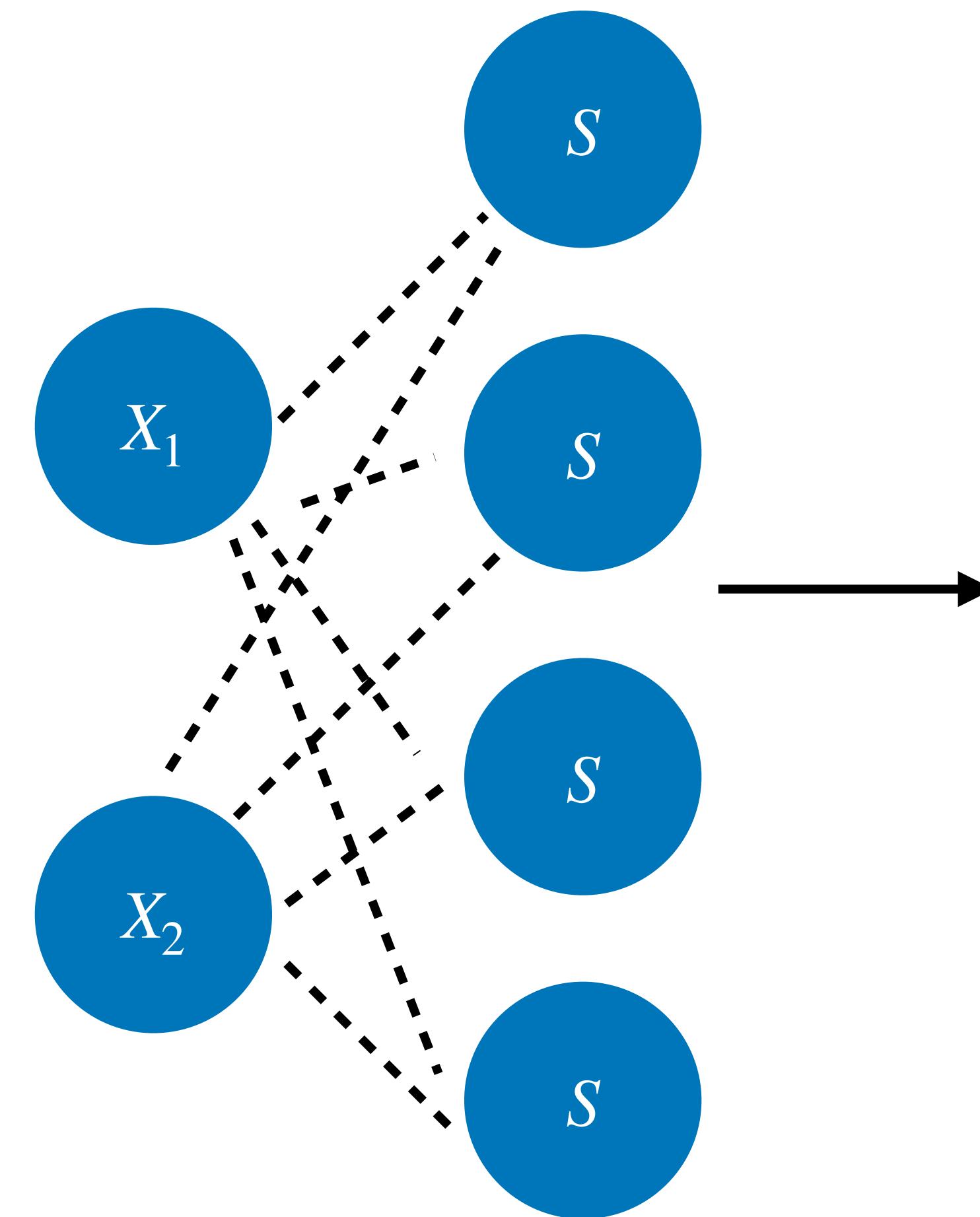
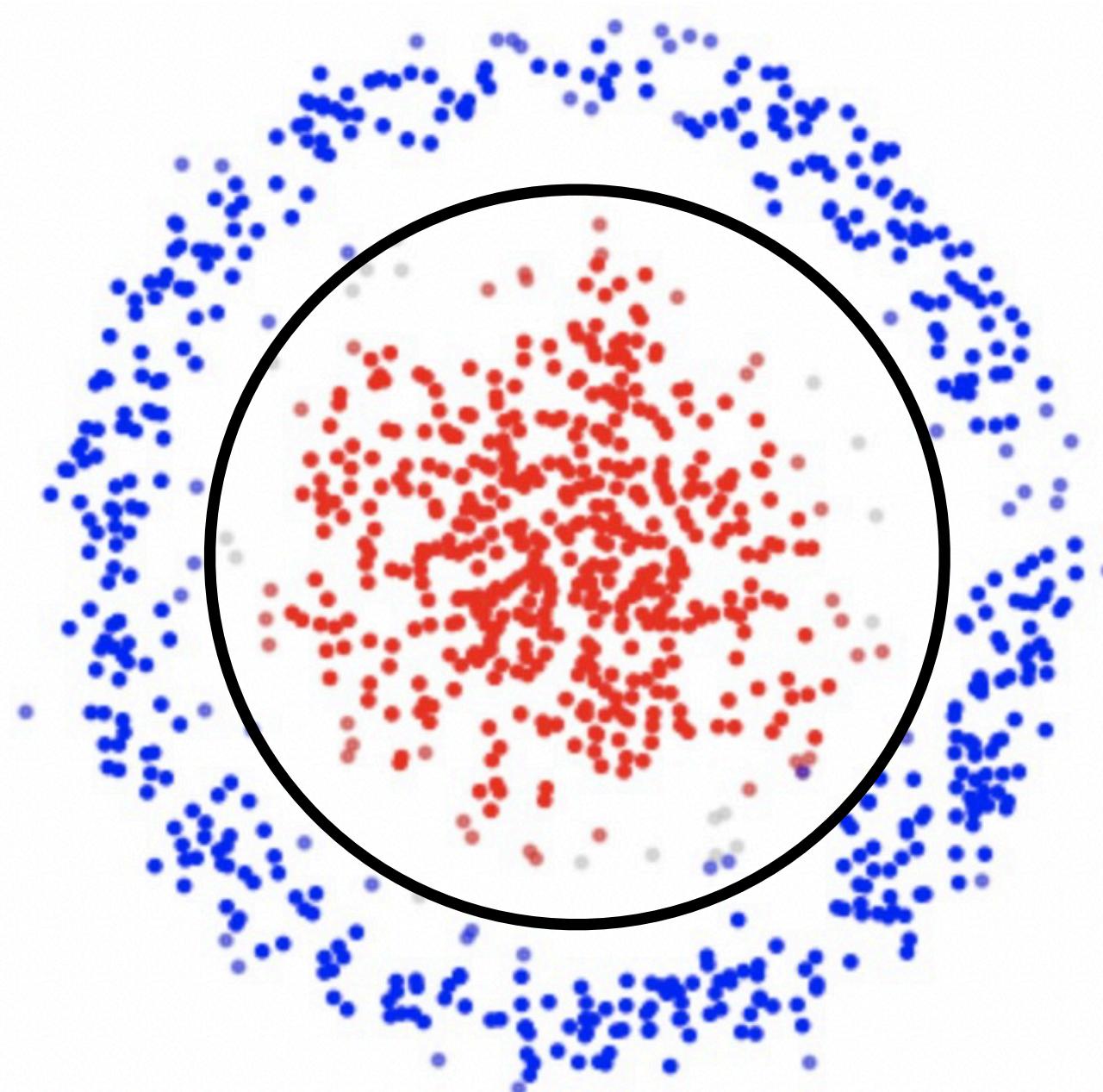
Multilayer Perceptron

Hiperplano

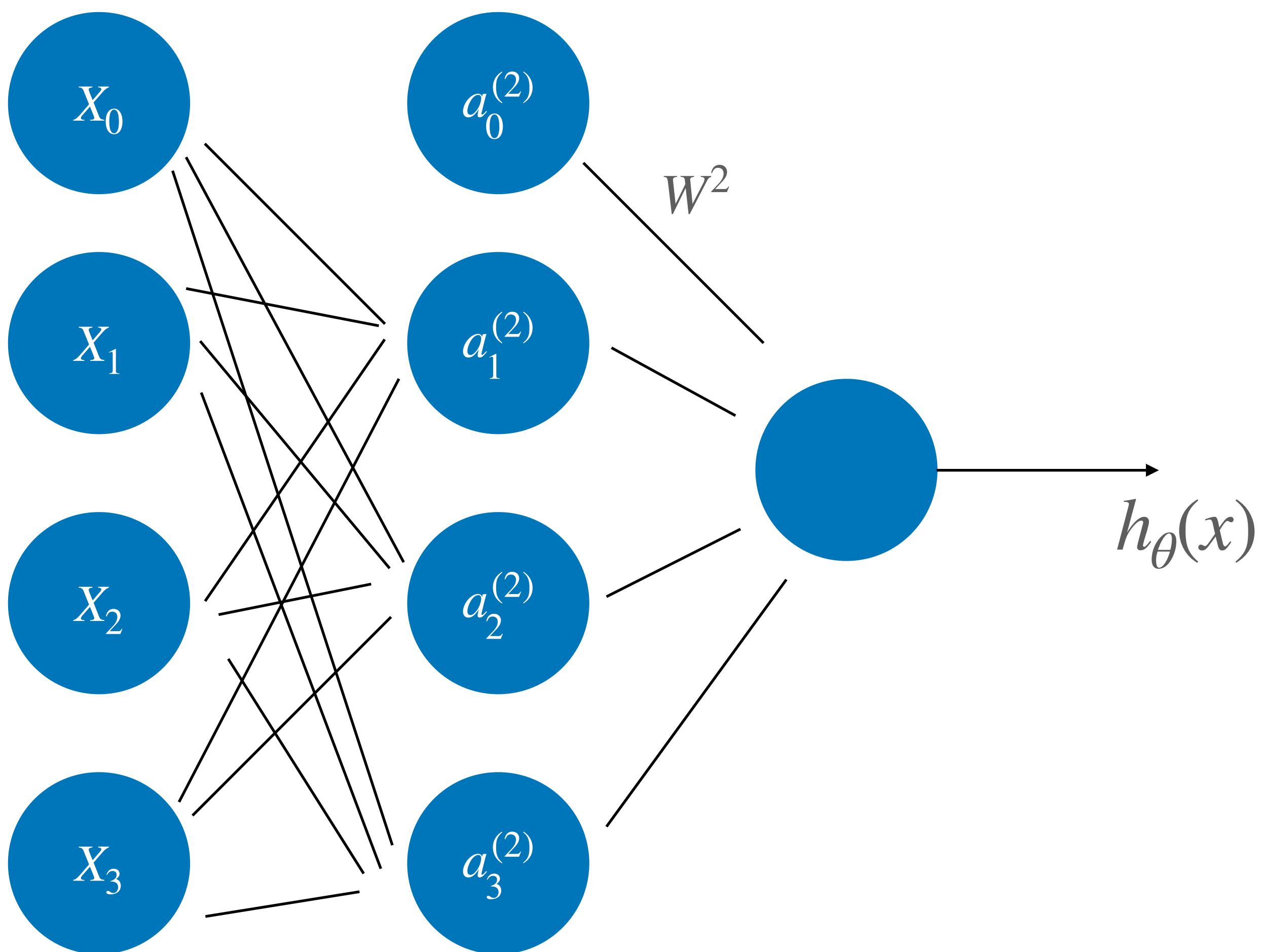


Multilayer Perceptron

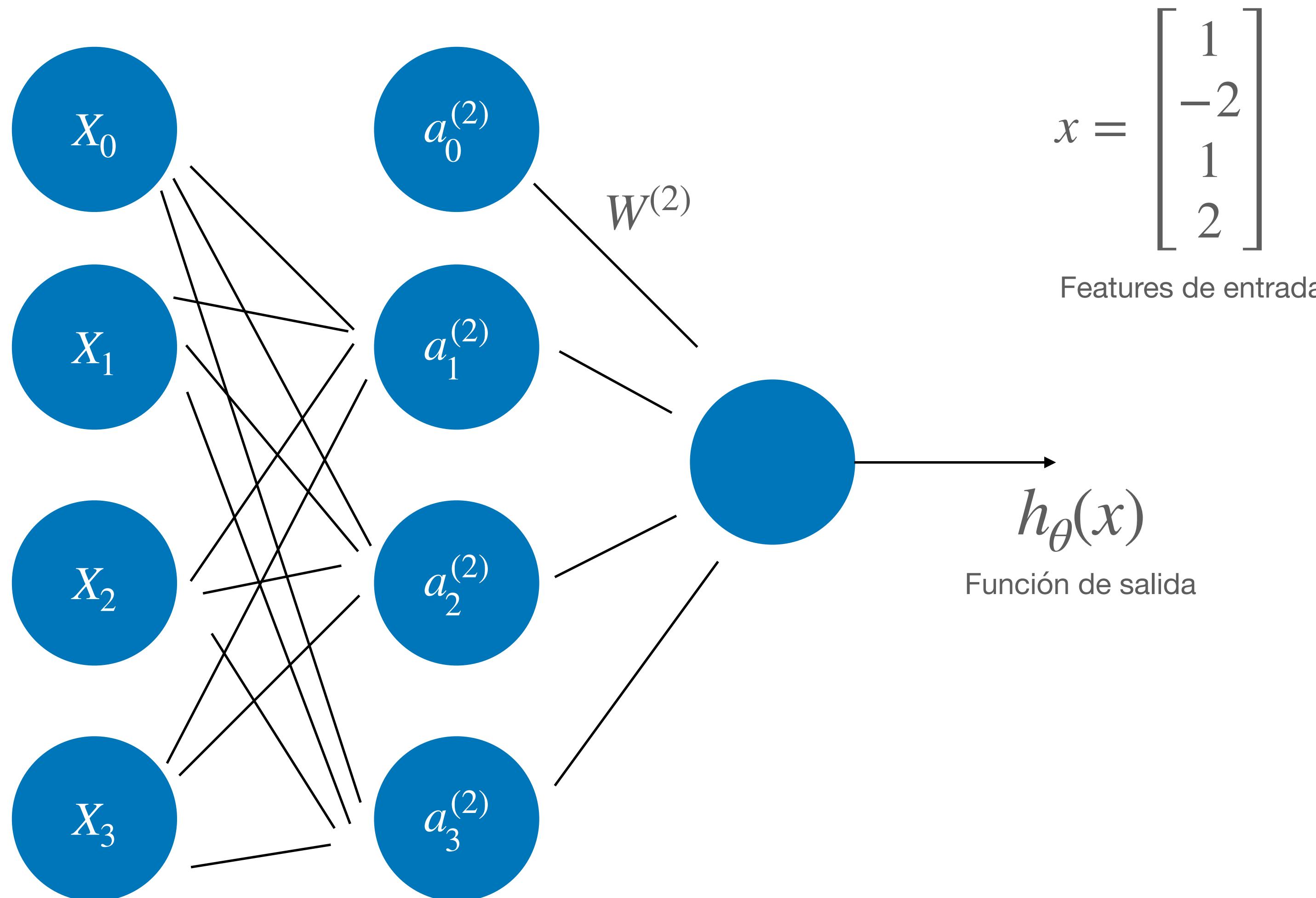
Hiperplano



Forward Propagation



Forward Propagation



$$x = \begin{bmatrix} 1 \\ -2 \\ 1 \\ 2 \end{bmatrix}$$

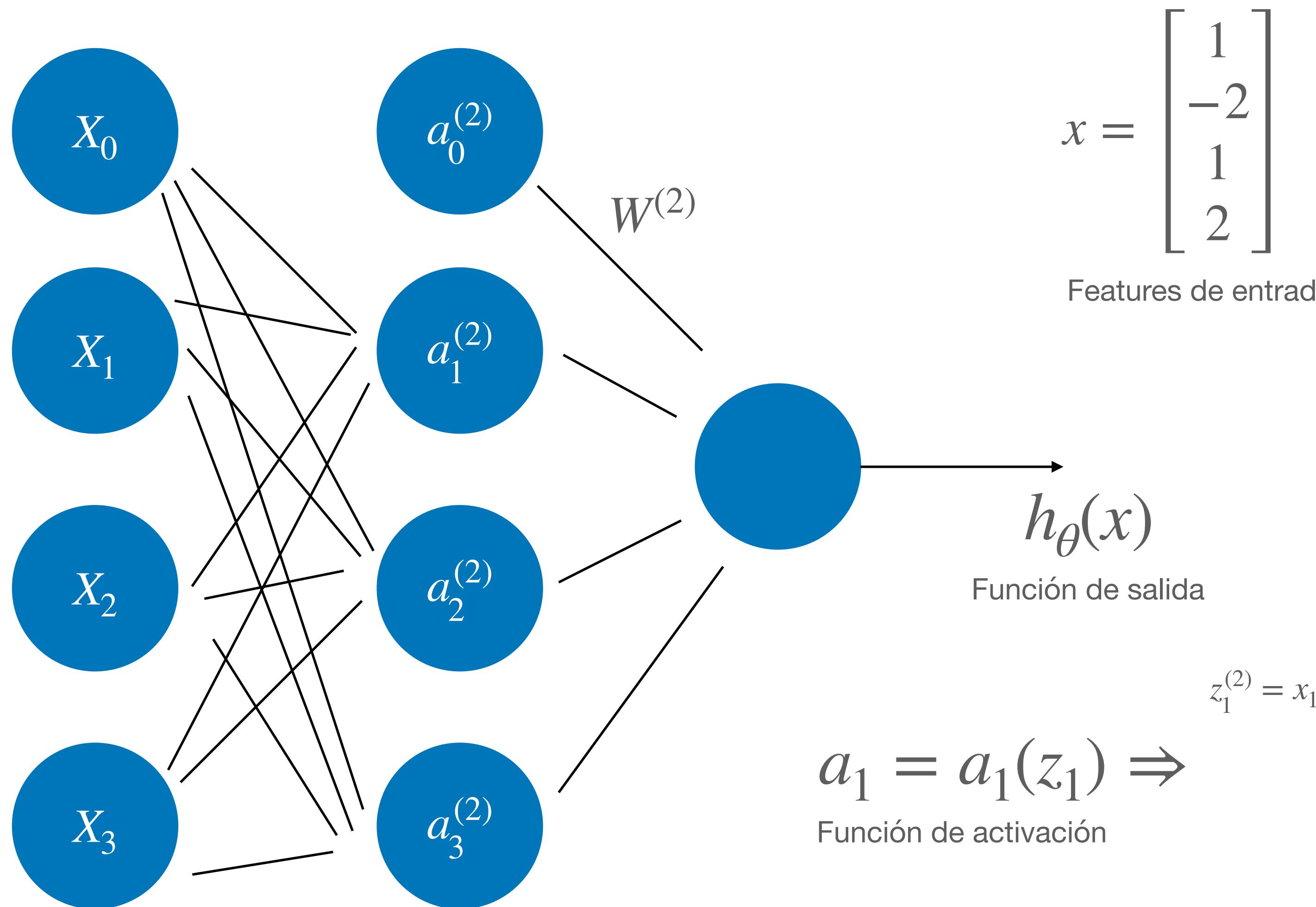
$$W^{(1)} = \begin{bmatrix} 0.6 & -0.1 & -1.0 & -3.0 \\ -2.2 & -3.4 & 0.2 & -2.6 \\ 4.3 & -3.2 & -4.6 & 2.8 \end{bmatrix}$$

Pesos de entrada

$$W^{(2)} = [3.8 \quad -0.1 \quad -5.0 \quad 2.2]$$

Pesos de salida

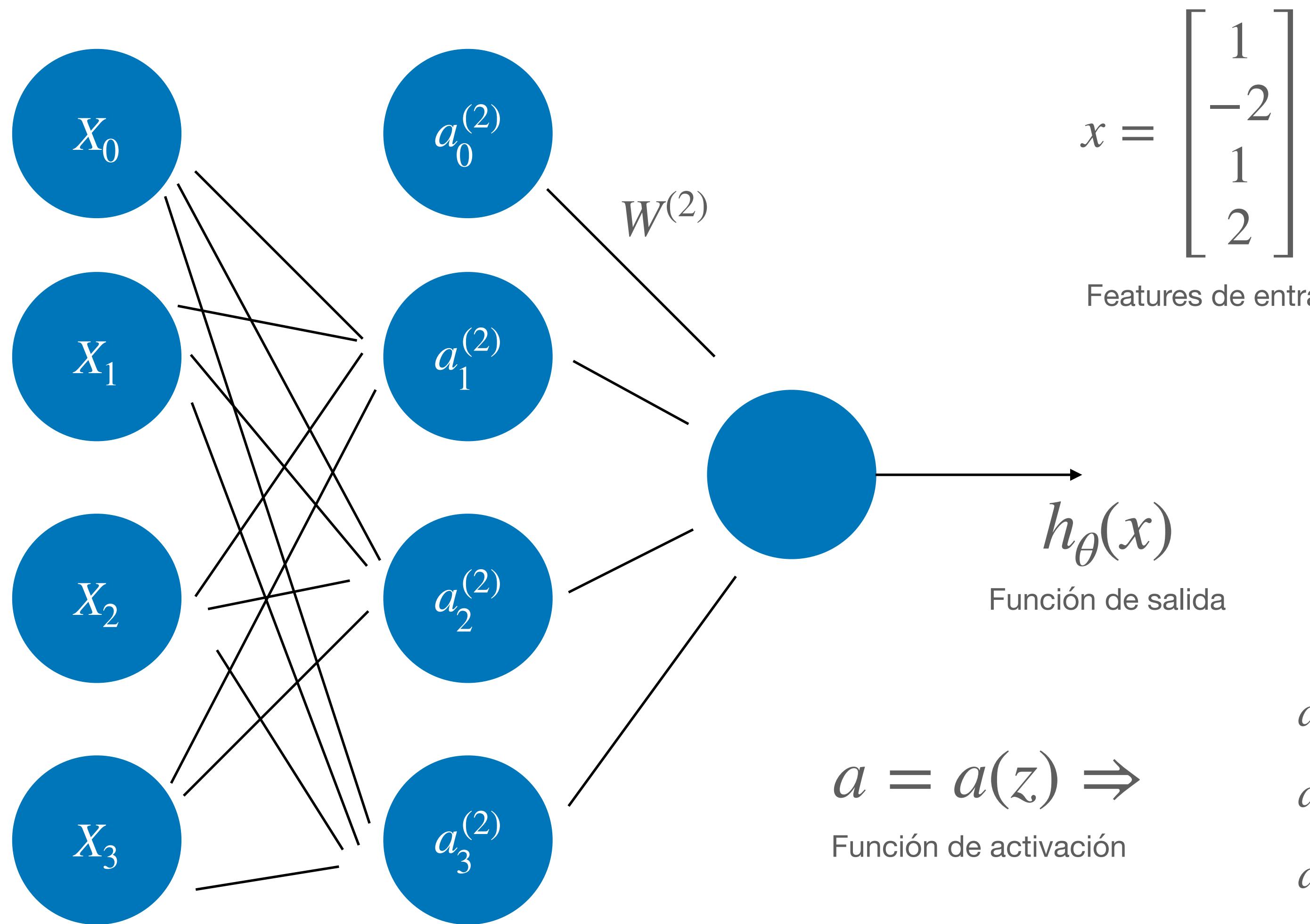
Forward Propagation



$$W^{(1)} = \begin{bmatrix} 0.6 & -0.1 & -1.0 & -3.0 \\ -2.2 & -3.4 & 0.2 & -2.6 \\ 4.3 & -3.2 & -4.6 & 2.8 \end{bmatrix}$$

$$W^{(2)} = [3.8 \quad -0.1 \quad -5.0 \quad 2.2]$$

Forward Propagation



$$x = \begin{bmatrix} 1 \\ -2 \\ 1 \\ 2 \end{bmatrix}$$

Features de entrada

$$W^{(1)} = \begin{bmatrix} 0.6 & -0.1 & -1.0 & -3.0 \\ -2.2 & -3.4 & 0.2 & -2.6 \\ 4.3 & -3.2 & -4.6 & 2.8 \end{bmatrix}$$

Pesos de entrada

$$W^{(2)} = [3.8 \quad -0.1 \quad -5.0 \quad 2.2]$$

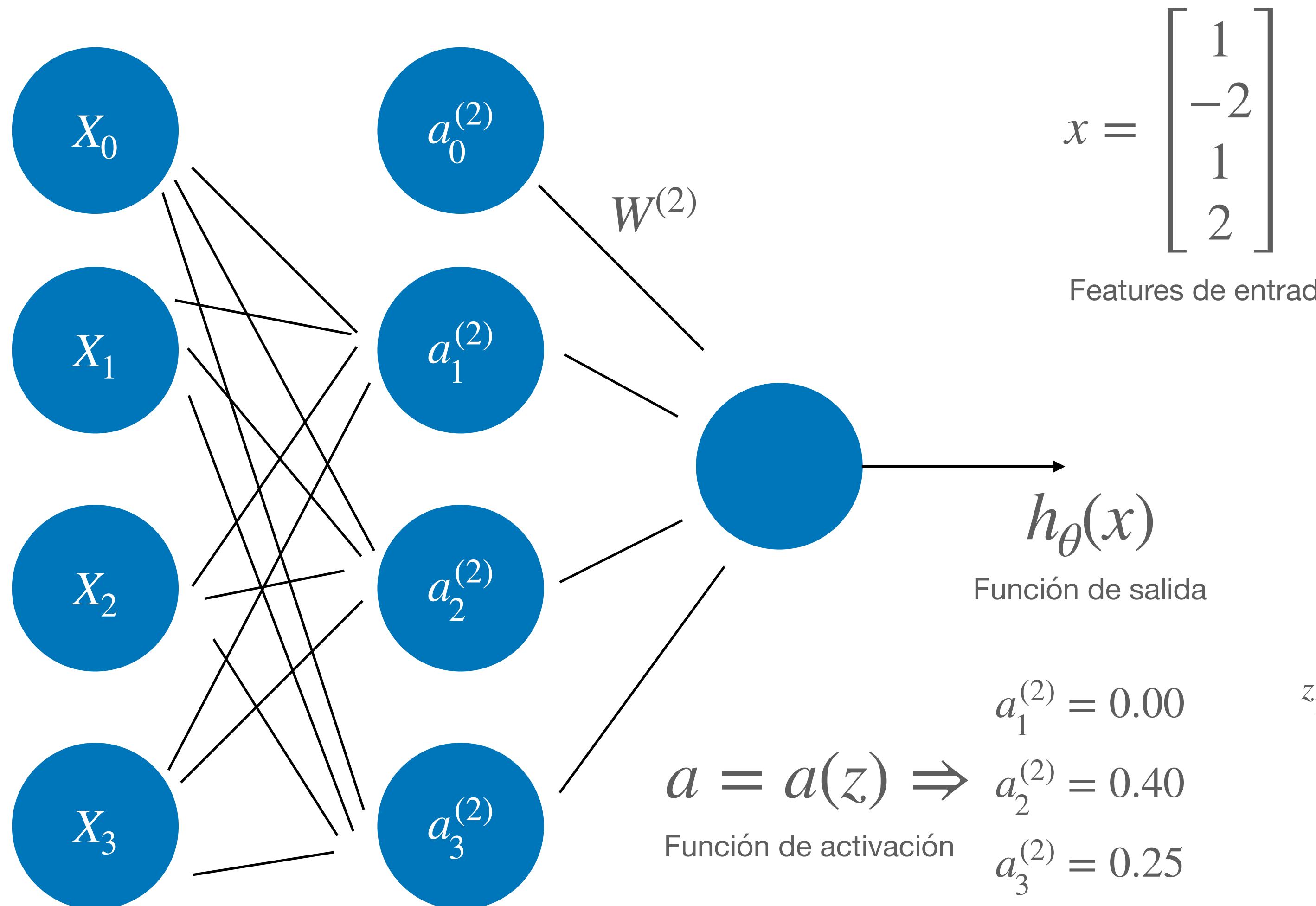
Pesos de salida

$$a_1^{(2)} = 0.00$$

$$a_2^{(2)} = 0.40$$

$$a_3^{(2)} = 0.25$$

Forward Propagation



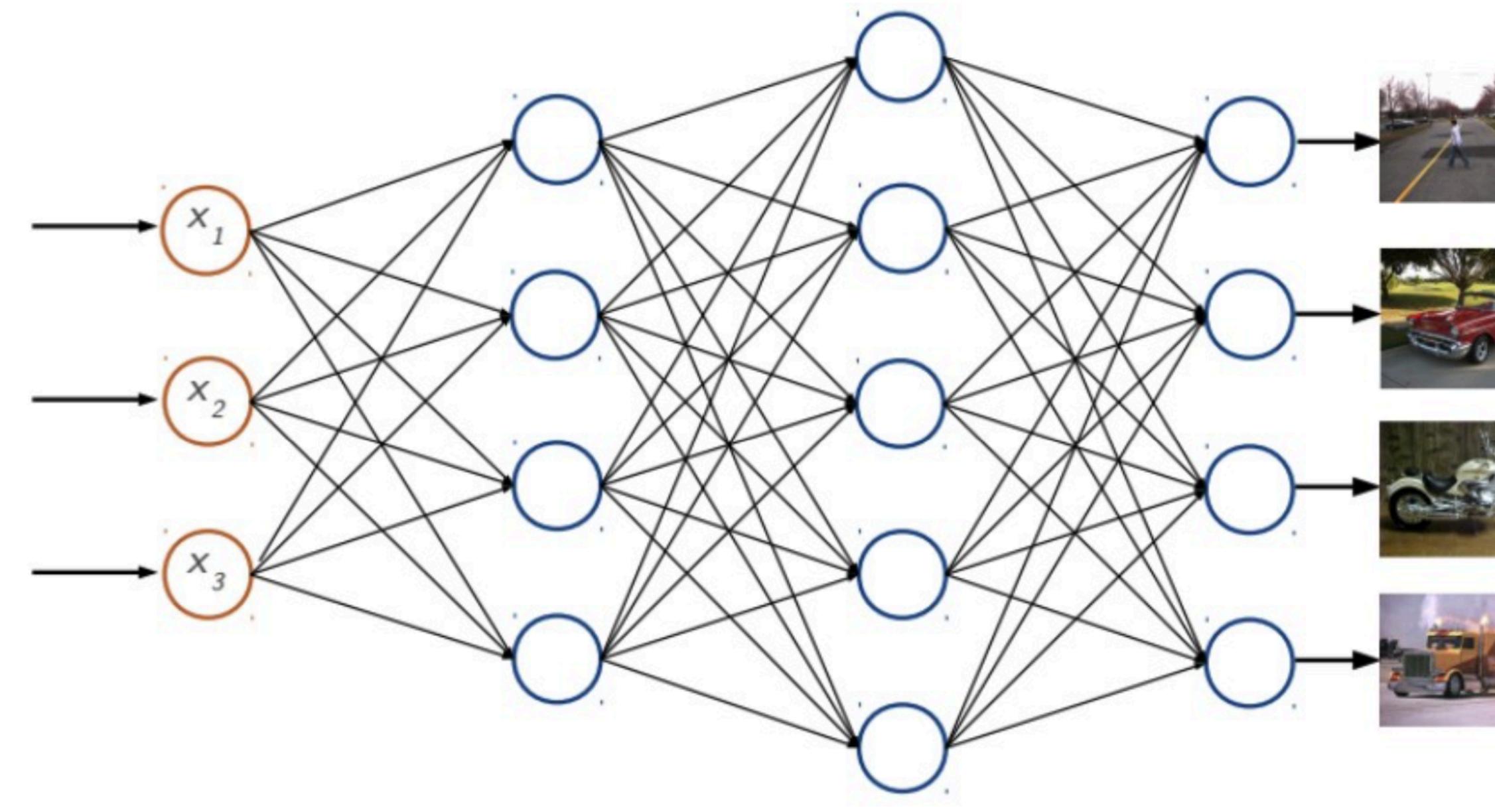
$$z_1^{(3)} = (3.8 \times 1) + (-0.1 \times 0) + (-5.0 \times 0.4) + (2.2 \times 0.25) = 2.35$$

$$a_1^{(3)} = \frac{1}{1 + e^{-z_1^{(3)}}} = 0.91$$

$$h_W(x) = 0.91$$

Clasificación en varias categorías

One vs Rest



$$h_{\Theta}(x) \approx \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Peatón

$$h_{\Theta}(x) \approx \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

Coche

$$h_{\Theta}(x) \approx \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Moto

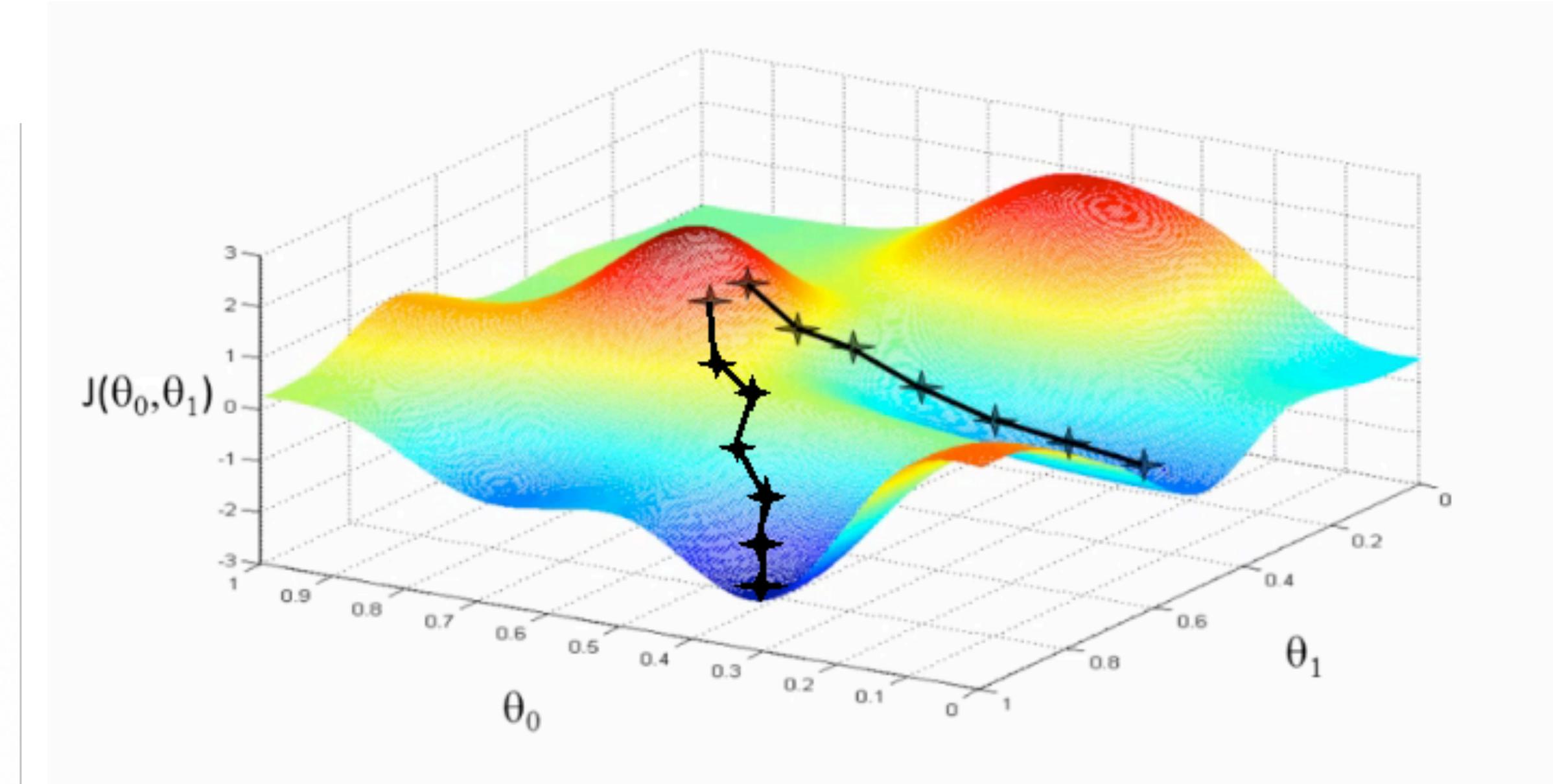
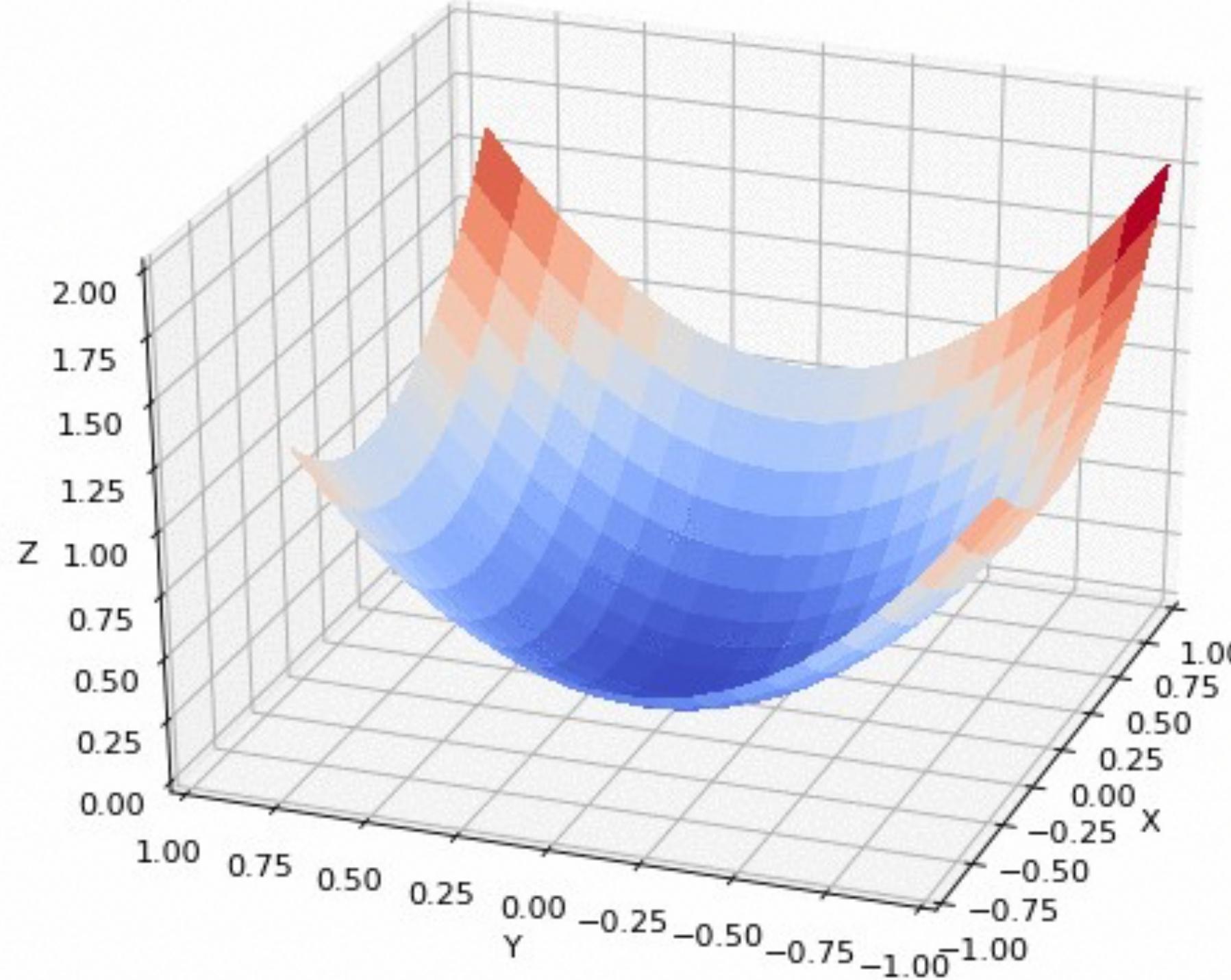
$$h_{\Theta}(x) \approx \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Camión

Gradient Descent

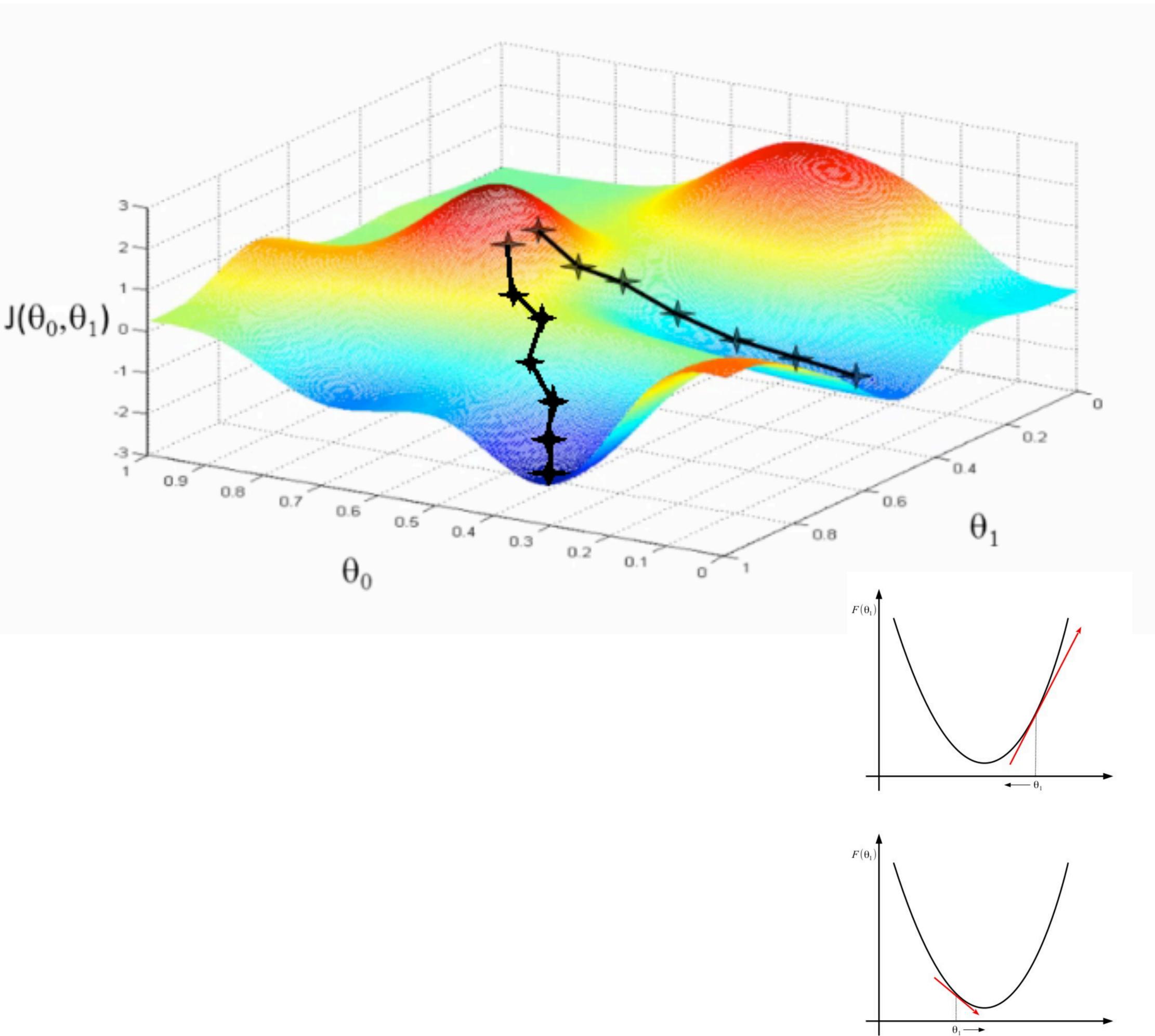
- La función de costo (**cost function/loss function**) representa lo alejando que tiene la predicción con el resultado deseado. Se pretende realizar una minimización de dicho valor.
- Se utiliza la técnica de **gradiente descendente**, donde la hipótesis es: el valor a reducir decrece rápidamente si en cada parámetro de la función se aplica un corrector negativo, obtenido a partir de la derivada parcial en el peso a establecer.
- Para no tener un problema de mínimos locales, se aplica un **ratio de aprendizaje**, de forma que en cada iteración, el valor de un peso es actualizado
- $w_i(t + 1) = w_i(t) \times (1 - \text{ratio}) - \text{ratio} \times \text{gradient}$

Gradient Descent



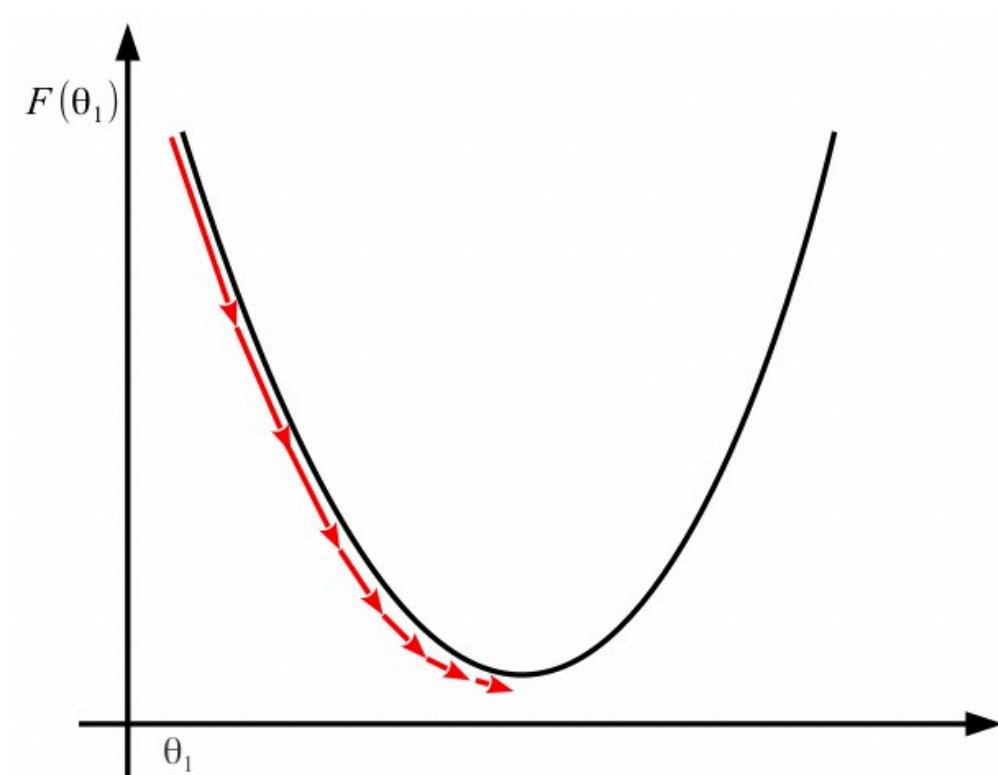
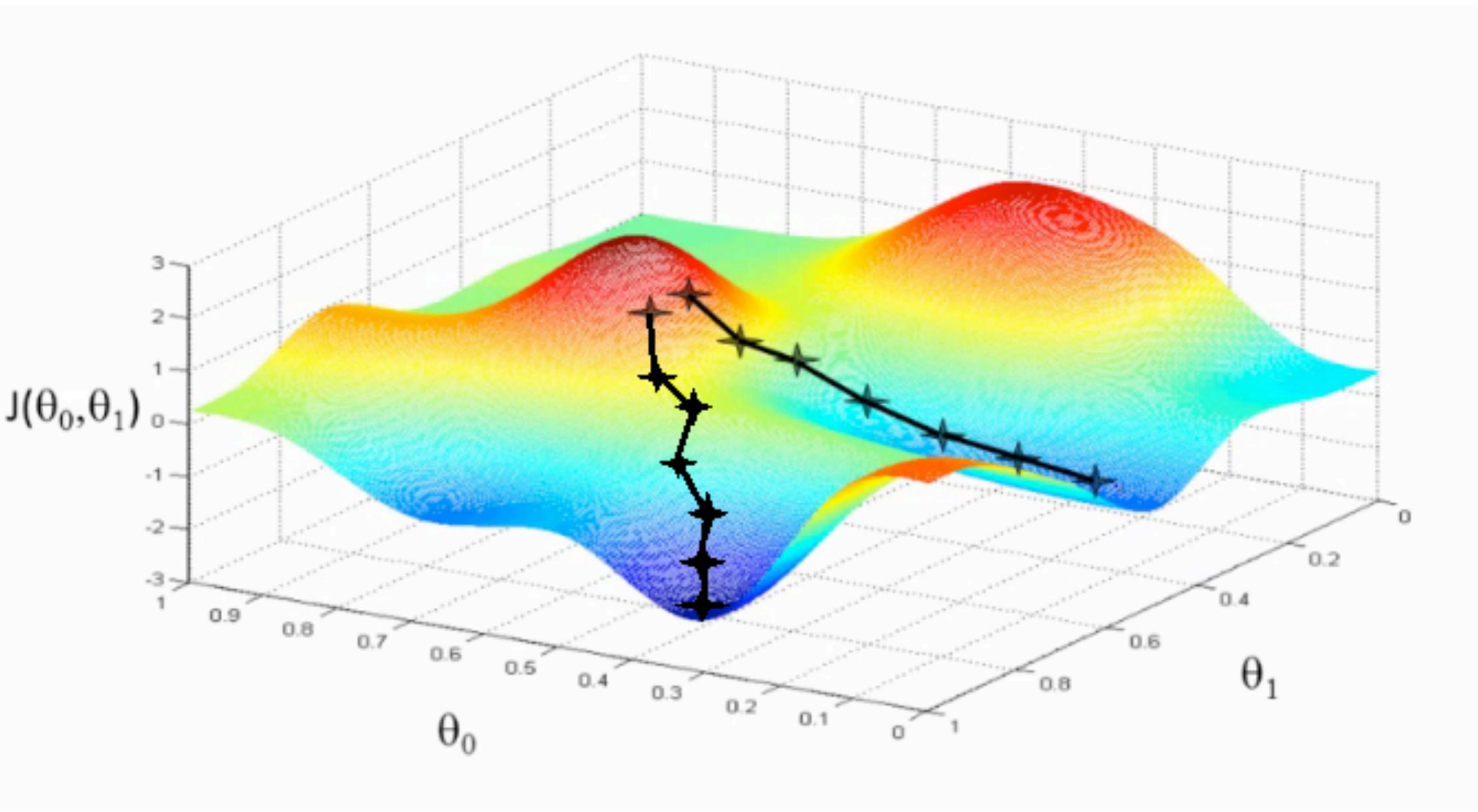
- Tomando una función $F(\theta_0, \theta_1)$ que se pretende minimizar.
- Se eligen valores θ_0, θ_1 elegidos de manera aleatoria.
- Estos valores son actualizados mientras esta función se reduzca.

Gradient Descent



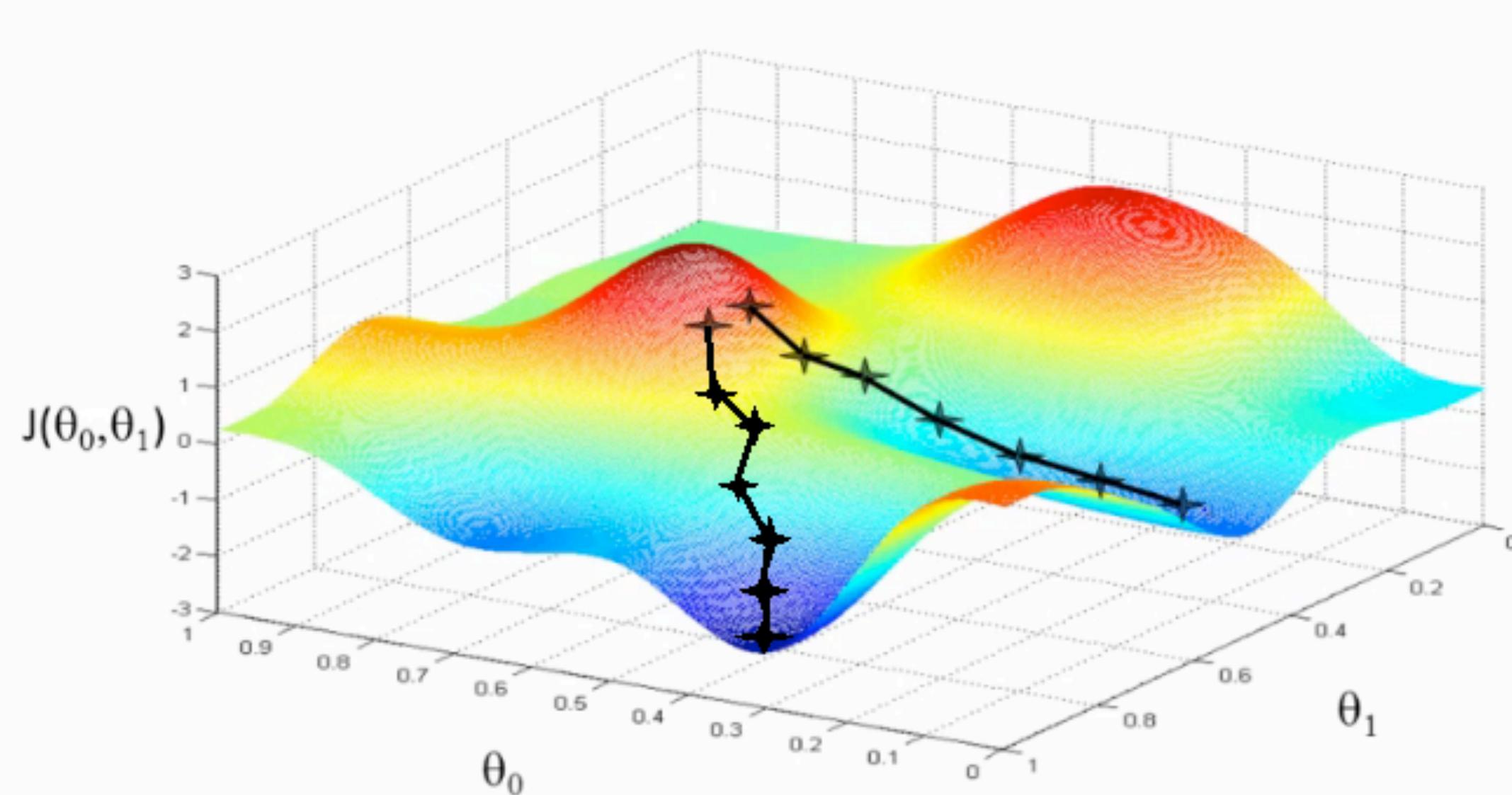
- La derivada parcial nos indicaría la pendiente (y signo) de la función en ese punto asociado a cierto parámetro. La siguiente actualización se repite hasta tener convergencia.
- $$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} F(\theta_0, \theta_1); \quad j = 0, j = 1, \alpha > 0$$
- Donde el factor α se denomina **razón de aprendizaje**.
- La actualización de los parámetros se realiza de manera simultánea, i. e., primero se calculan los nuevos parámetros θ_j (los que haya), y luego se reasignan al parámetro θ_j original.

Gradient Descent



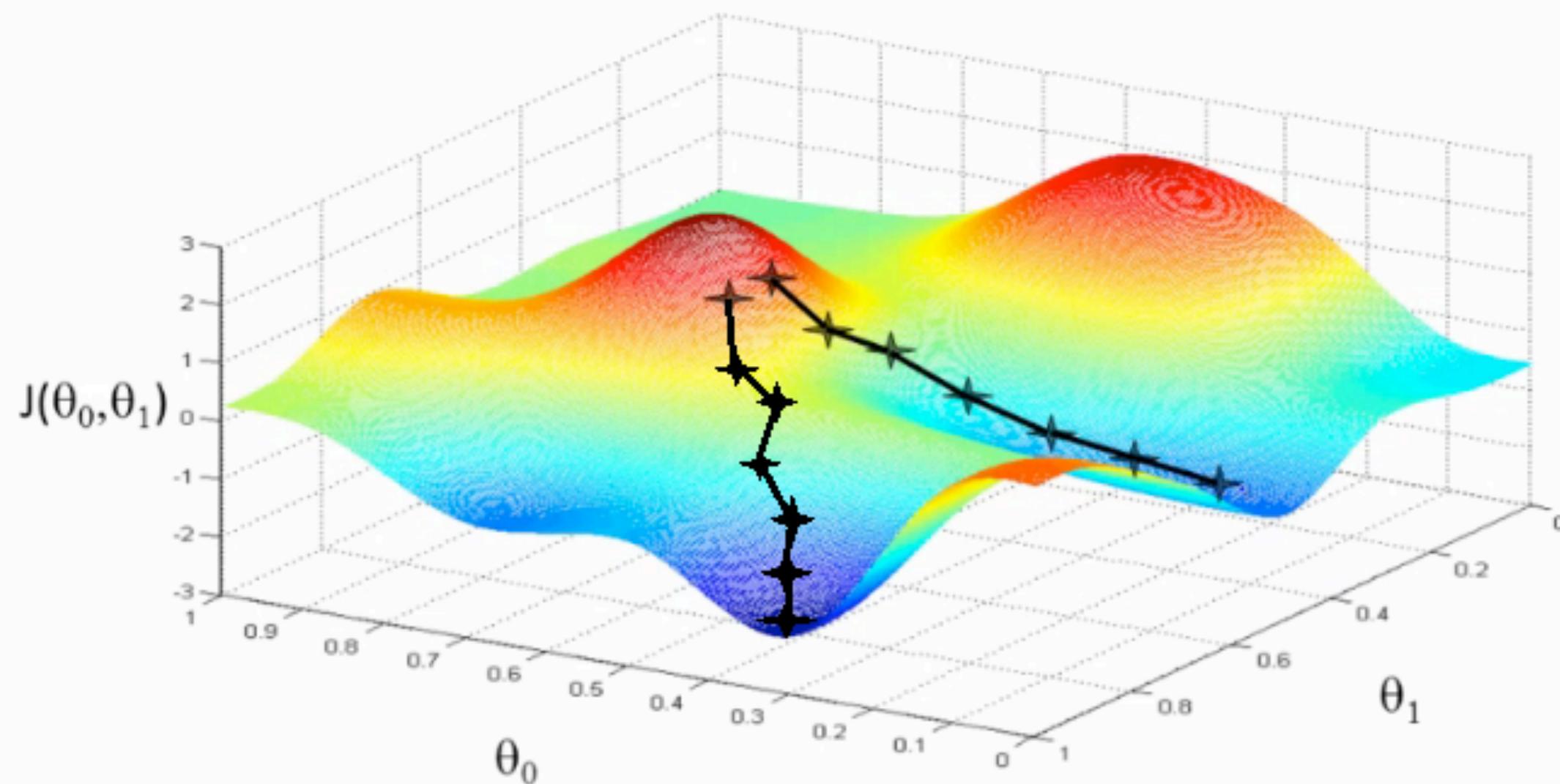
- Cuando el valor de los parámetros se aproxima a un mínimo (**local**), éste da pasos más pequeños..
- El factor α se tiene que fijar a un valor aceptable. Si éste es muy pequeño, la convergencia puede ser demasiado lenta, y si fuera demasiado grande, el algoritmo pudiera no tener convergencia.

Gradient Descent



- **Delta rule.** Consiste en dirigir la búsqueda hacia el descenso por máximo gradiente.
- Considerando el error: $E(w) = \frac{1}{2} \sum_{d \in D} (t_d - o_d)^2$
- Siendo t_d el valor asociado a la variable de respuesta para el caso d-ésimo, y o_d el valor de salida producido por la red en dicho caso.
- Para calcular la dirección de máximo descenso se calcula la derivada de E con respecto a todos sus pesos,
- $\nabla E(w) = \left[\frac{\partial E}{\partial w_0}, \frac{\partial E}{\partial w_1}, \dots, \frac{\partial E}{\partial w_n} \right]$

Gradient Descent



$$f(x) = \frac{1}{1 + e^{-x}} \Rightarrow f'(x) = f(x)(1 - f(x))$$
$$f(x) = \tanh(x) \Rightarrow f'(x) = (1 - f^2(x))$$

- $\nabla E(w) = \left[\frac{\partial E}{\partial w_0}, \frac{\partial E}{\partial w_1}, \dots, \frac{\partial E}{\partial w_n} \right]$
- Este vector representa al dirección que produce el mayor aumento de E en un punto determinado. Por tanto, nos interesa en su lugar $-\nabla E(w)$, la dirección que maximiza la disminución de dicho error.
- $w \leftarrow w + \Delta w; \quad \Delta w = -\eta \nabla E(w).$
- Donde este factor η es otra razón de aprendizaje.
- En el caso general, donde se tienen funciones de activación f , se tiene que

- $\Delta w_i = \eta \sum_{d \in D} (t_d - o_d) f'(o_d) x_{id}$

Aprendizaje