Fundamentos de Deep Learning

Dr. Manuel Castillo-Cara
www.manuelcastillo.eu
Departamento de Inteligencia Artificial
Escuela Técnica Superior de Ingeniería Informática
Universidad Nacional de Educación a Distancia (UNED)

Preliminar

- Conceptos avanzados de redes neuronales © 2022 by Manuel Castillo-Cara is licensed under Attribution-NonCommercial 4.0 International

Attribution-NonCommercial 4.0 International (CC BY-NC 4.0)

- Contrato de Manuel Castillo-Cara se encuentra financiado por la Unión Europea-NextGenerationEU

UNED

ETS de Ingeniería Informática

PROGRAMA IBEROAMERICANO CYTED CIENCIA Y TECNOLOGÍA PARA EL DESARROLLO

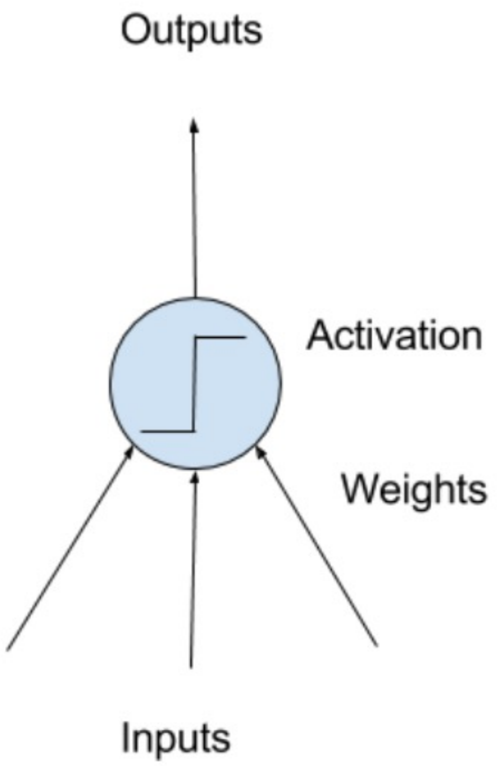
Índice

Background

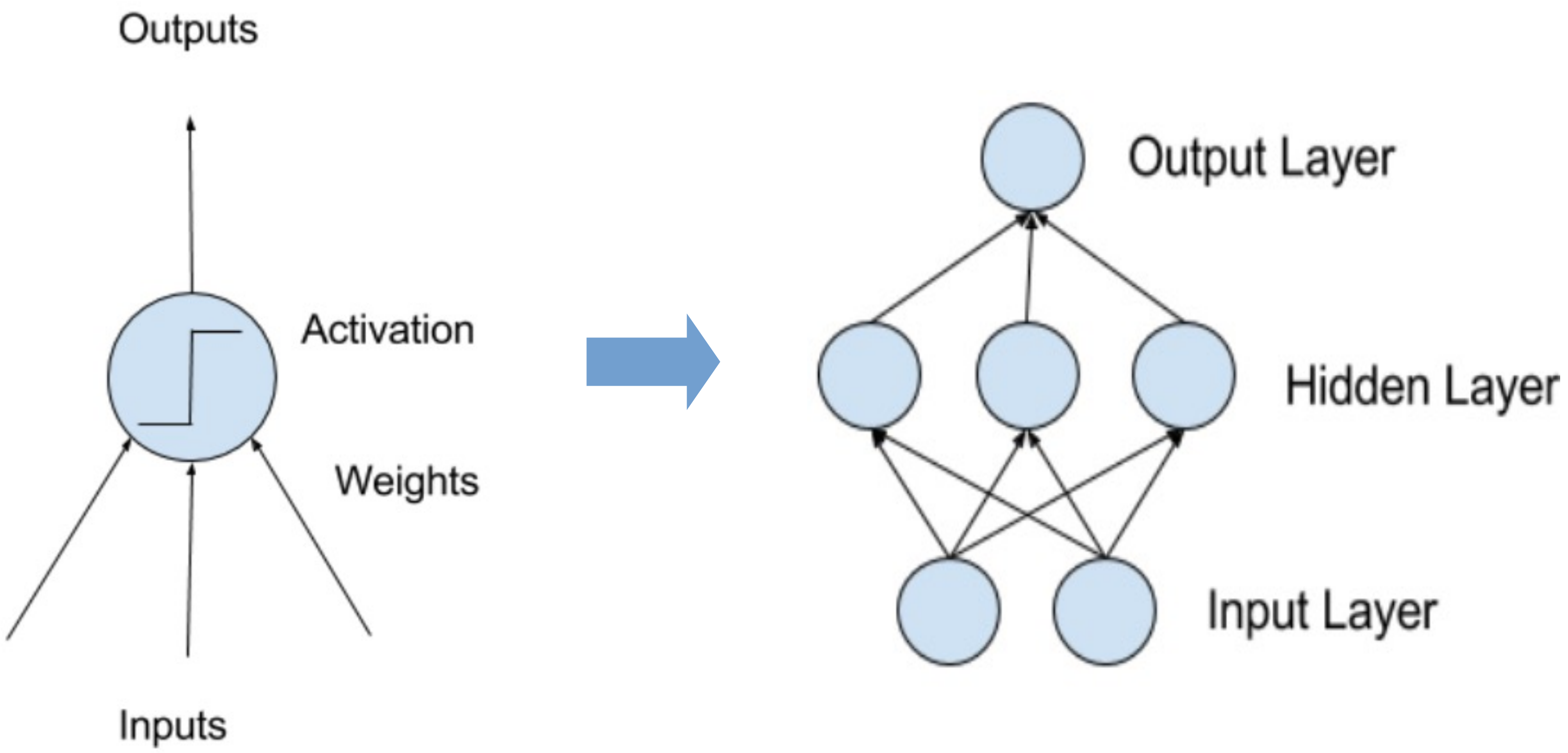
ETS de Ingeniería Informática

UNED

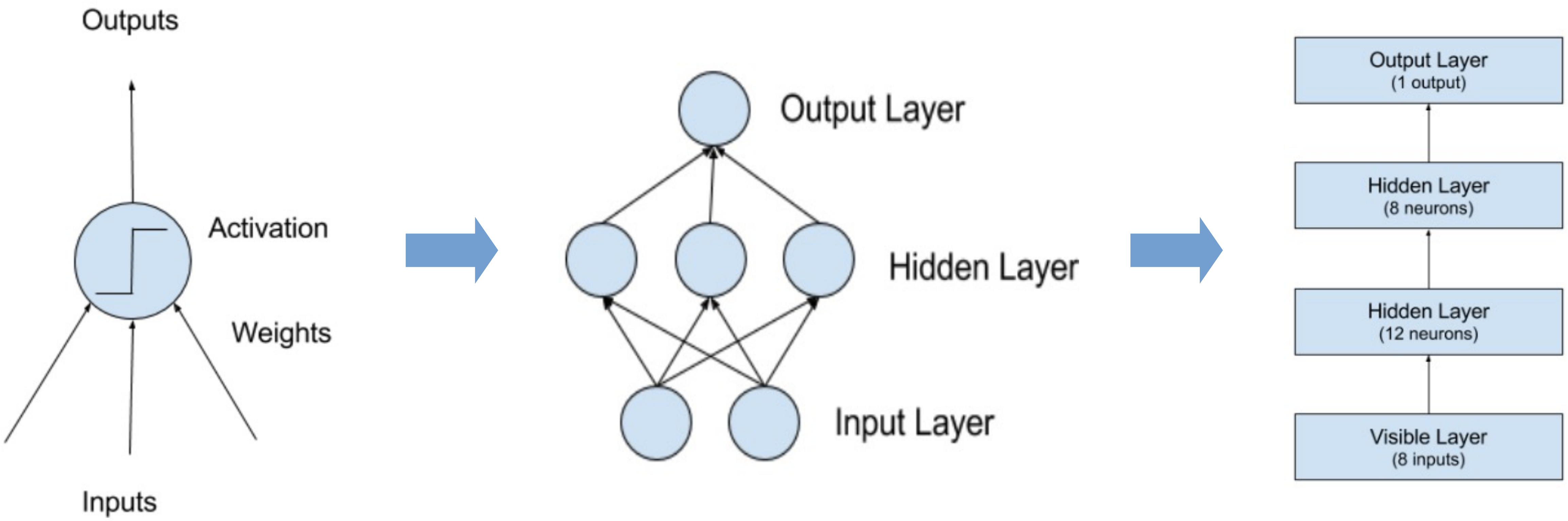
0. Background



0. Background



0. Background



La neurona

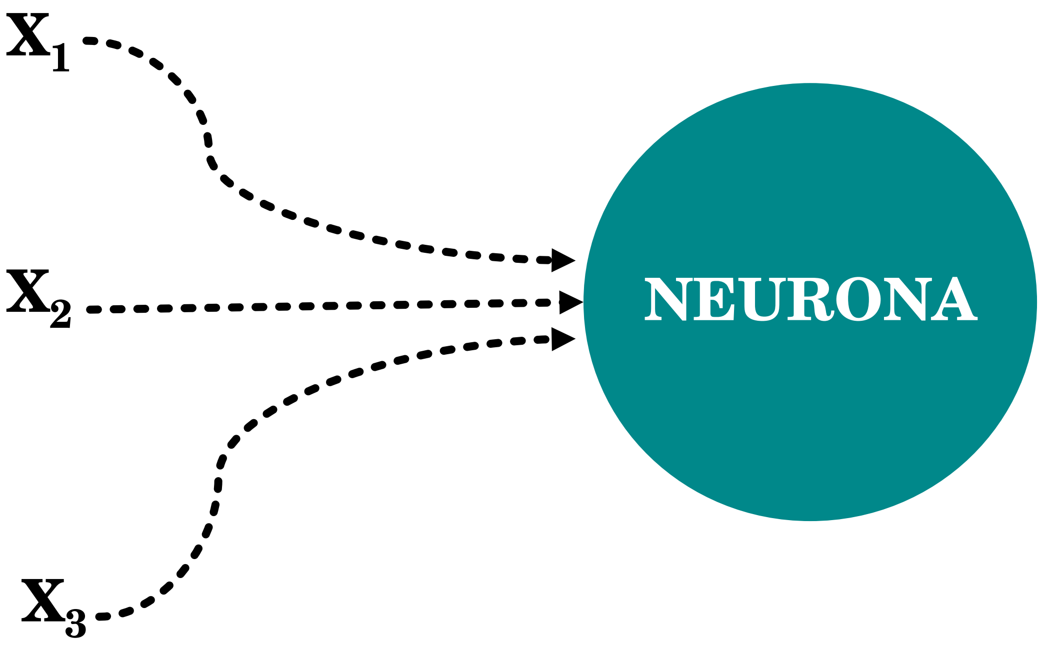
ETS de Ingeniería Informática

UNED

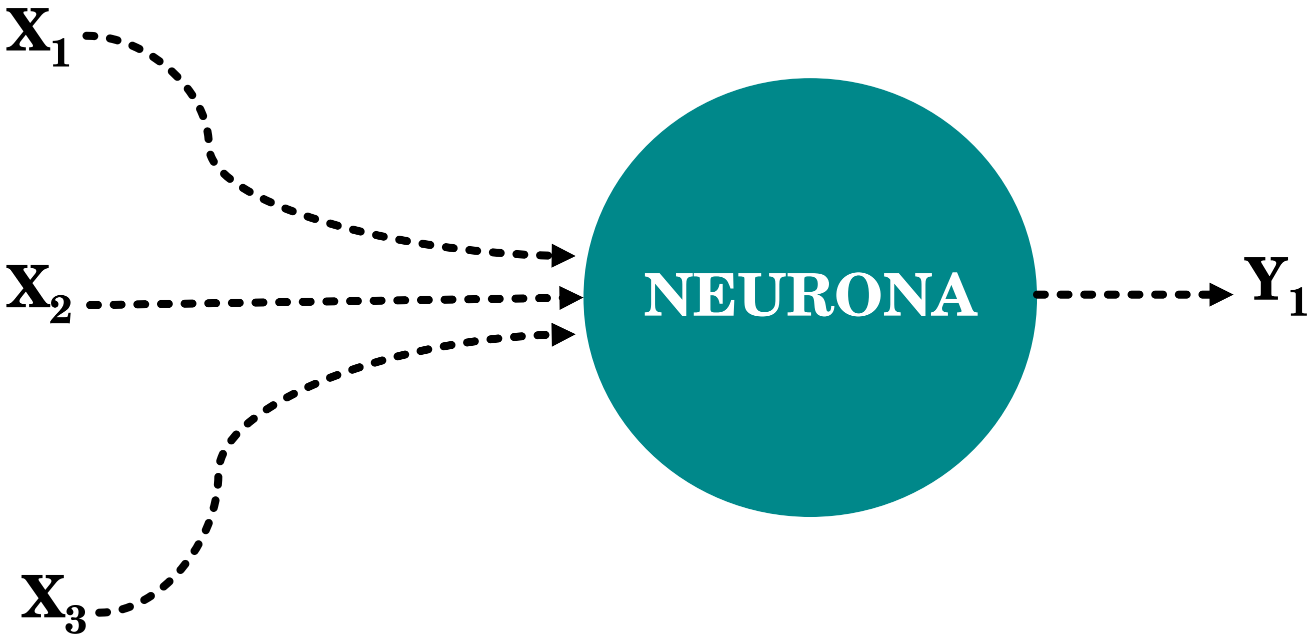
1. Neurona

NEURONA

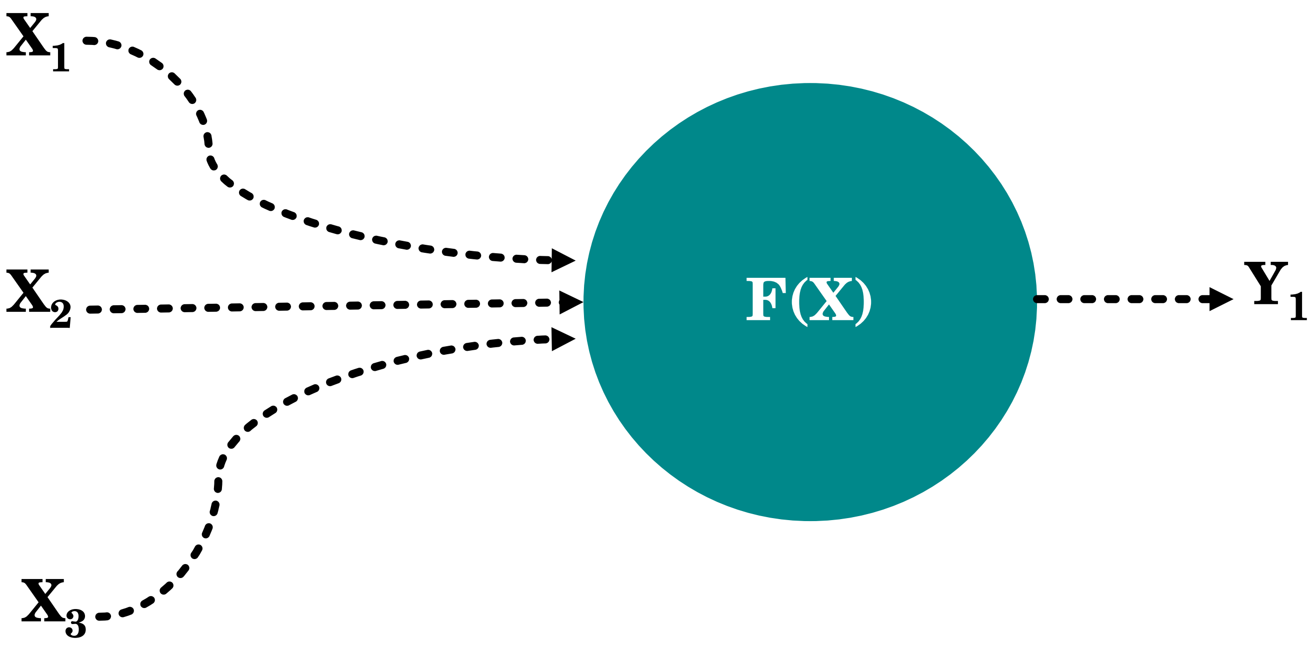
1. Neurona



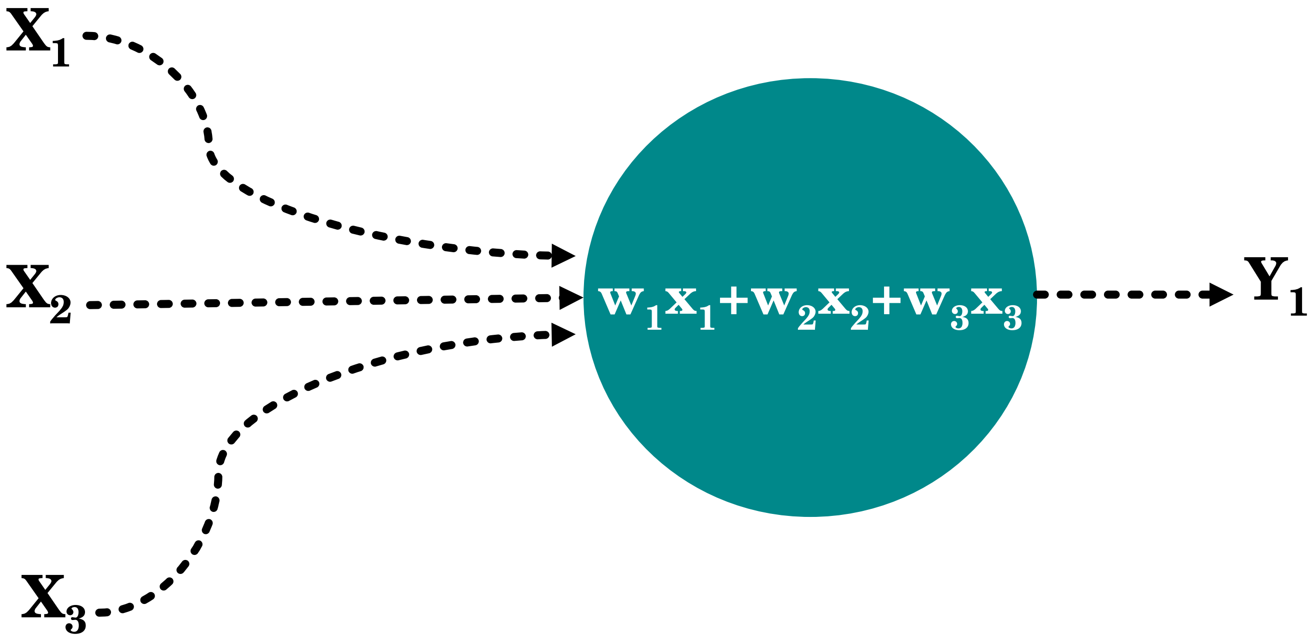
1. Neurona



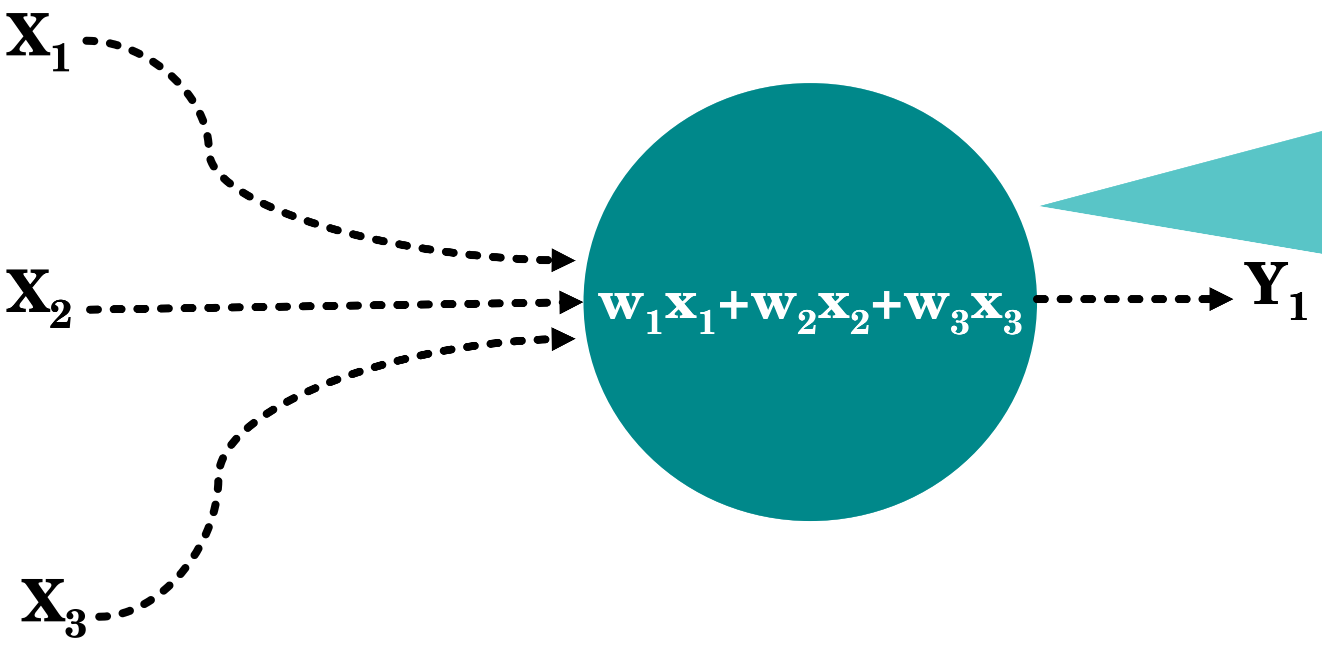
1. Neurona



1. Neurona

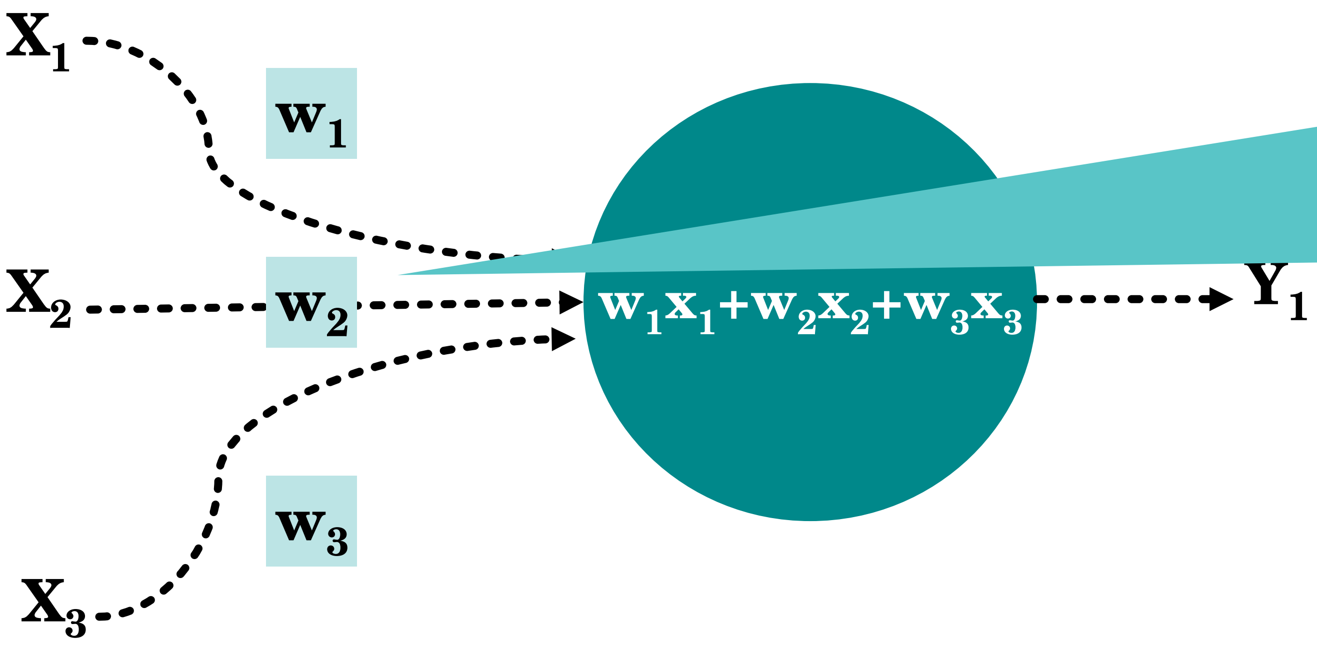


1. Neurona



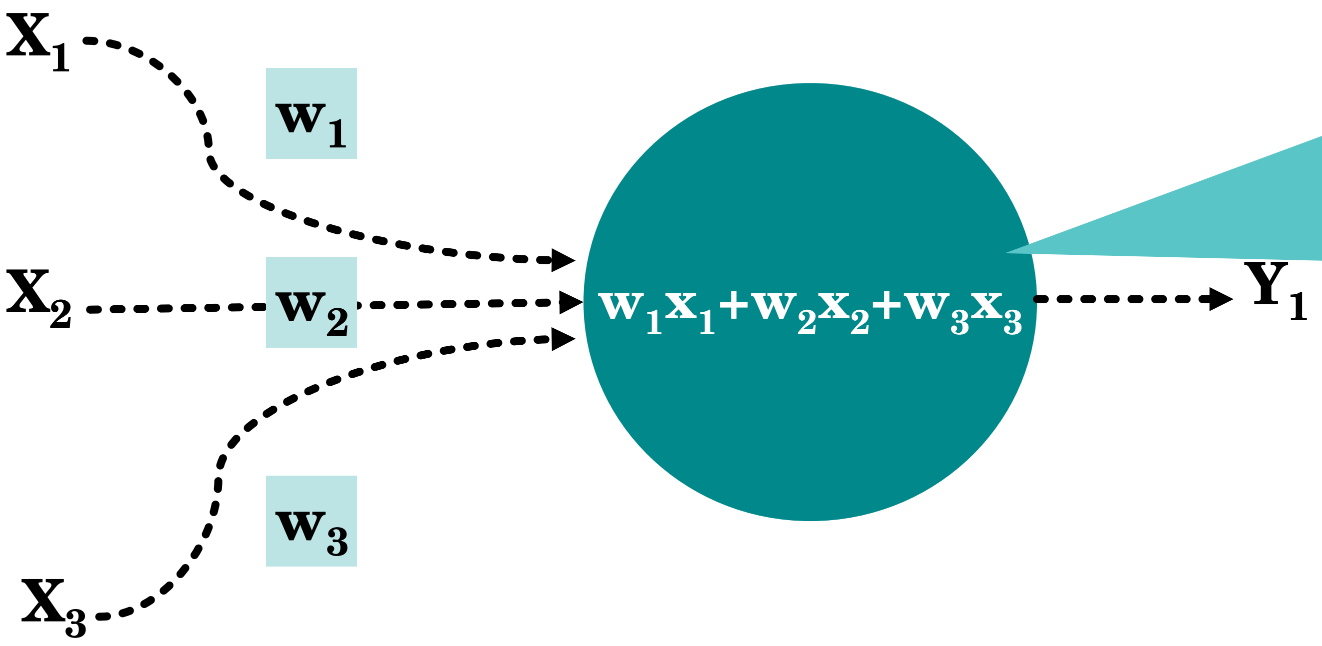
Utiliza todas la entradas para realizar una **suma ponderada** de ellos.

1. Neurona



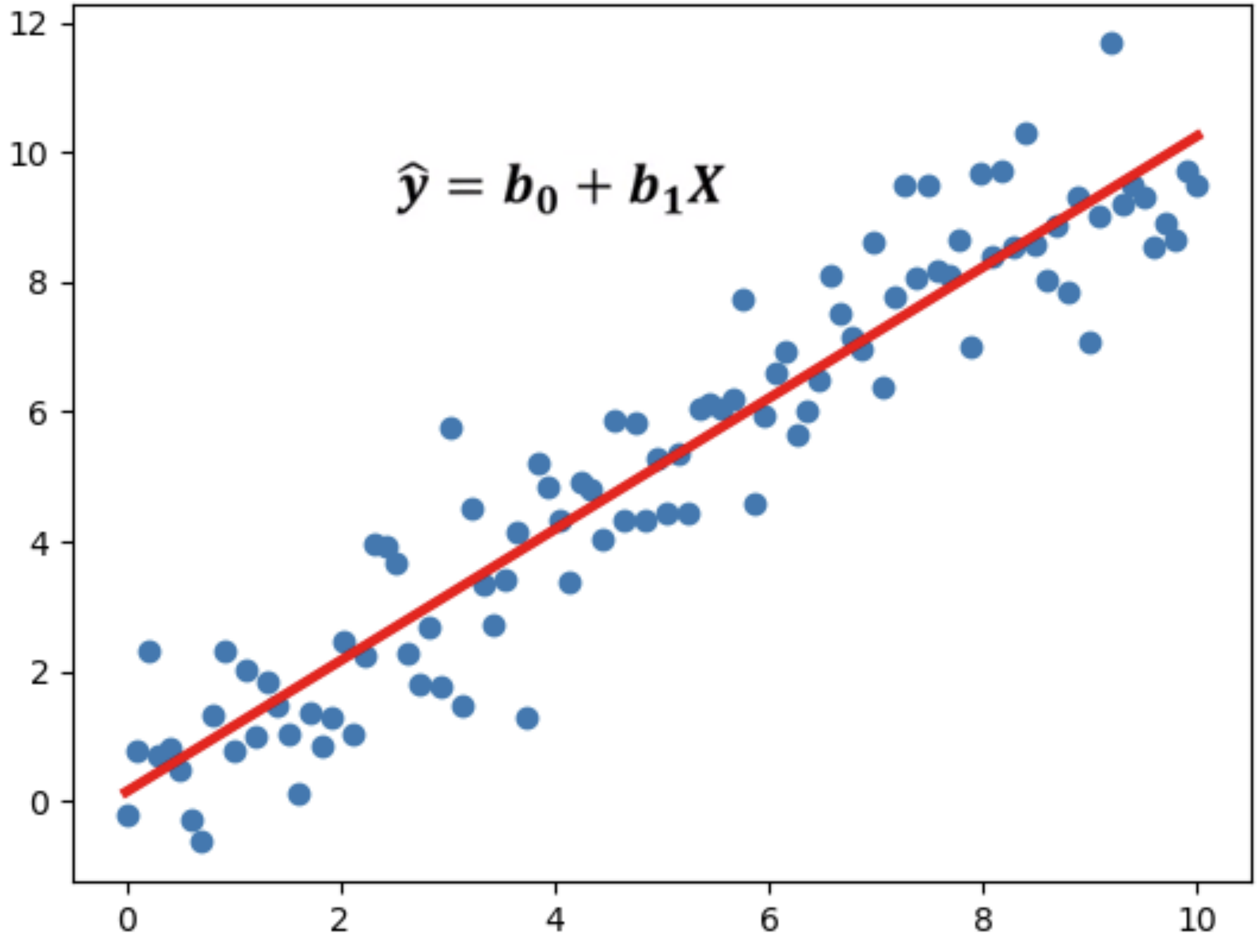
Hay que tener en cuenta que la suma viene dada por el **peso** *(w)* que se le asigna a cada una de las entradas

1. Neurona

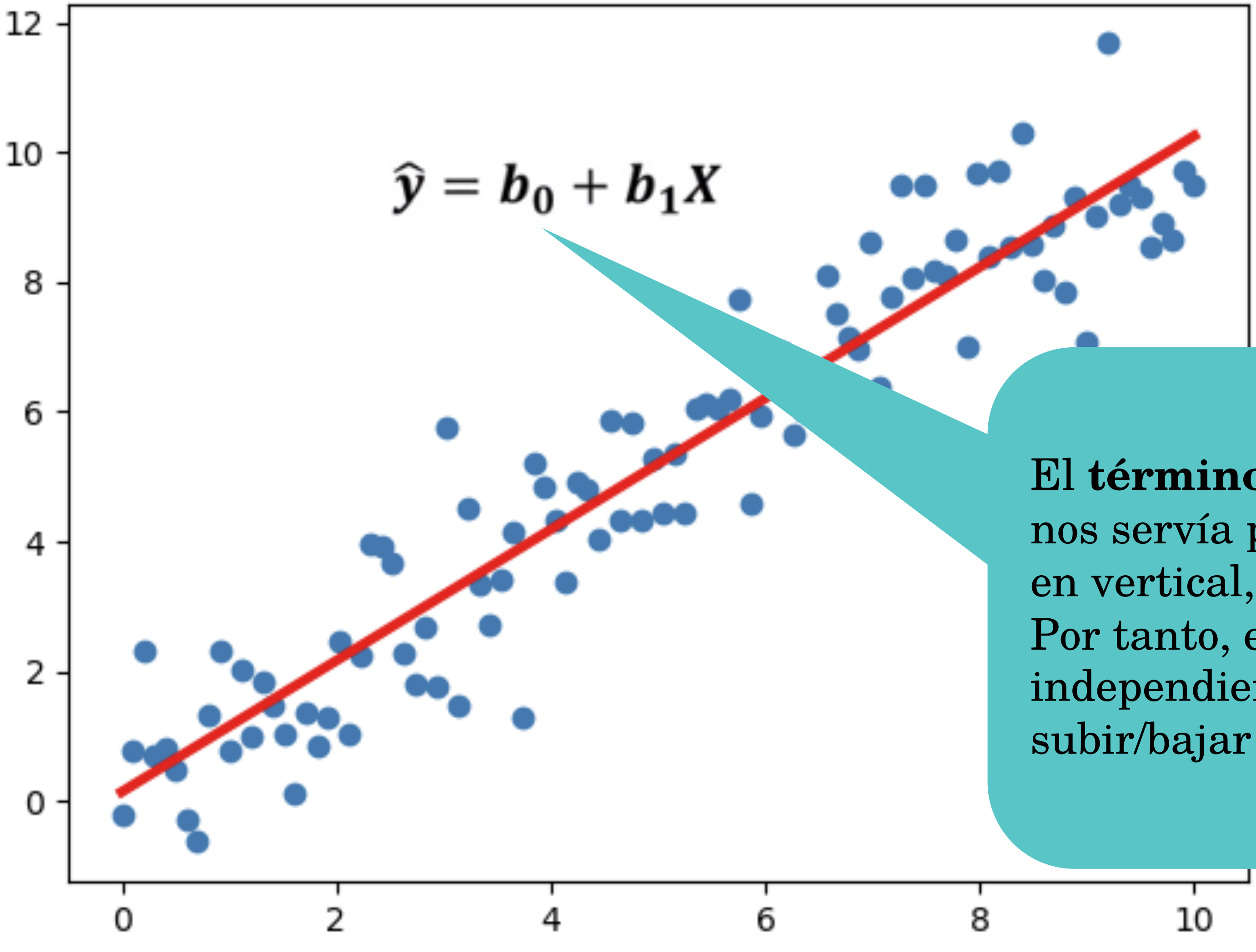


¿A qué nos suena?

2. Regresión lineal

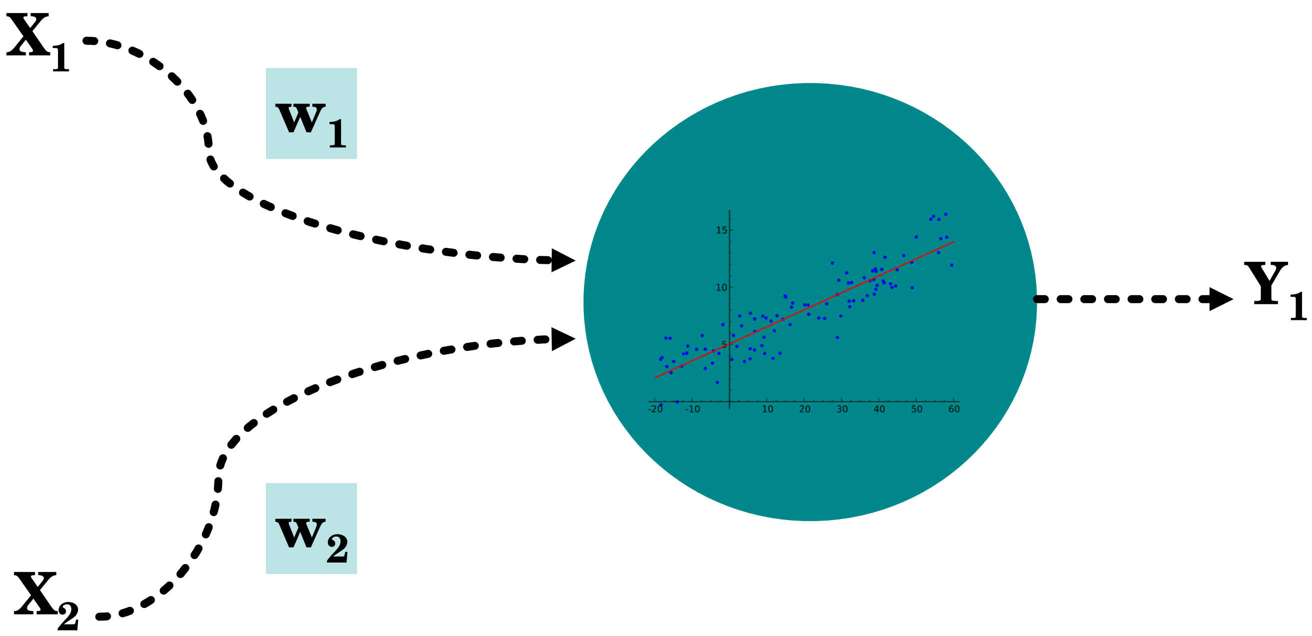


2. Regresión lineal



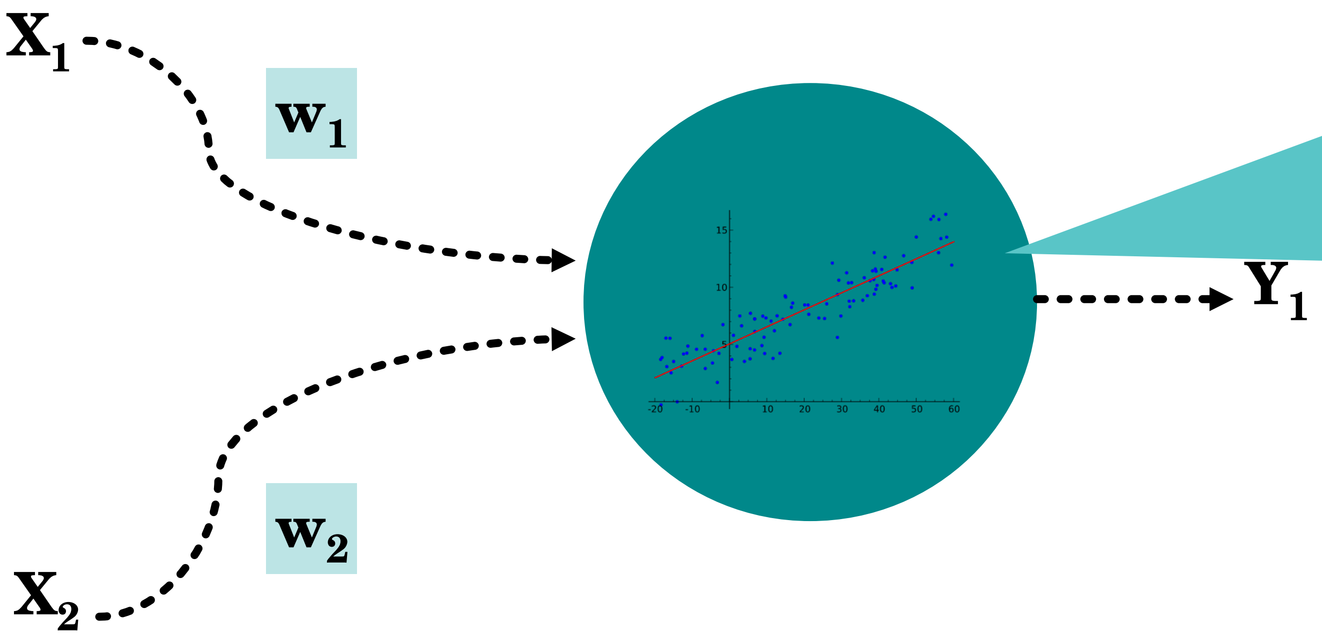
El **término independiente** nos servía para mover la recta en vertical, i.e., en el eje y. Por tanto, el término independiente nos servía para subir/bajar la recta

3. Función de la NN



$$y = w_1x_1 + w_2x_2$$

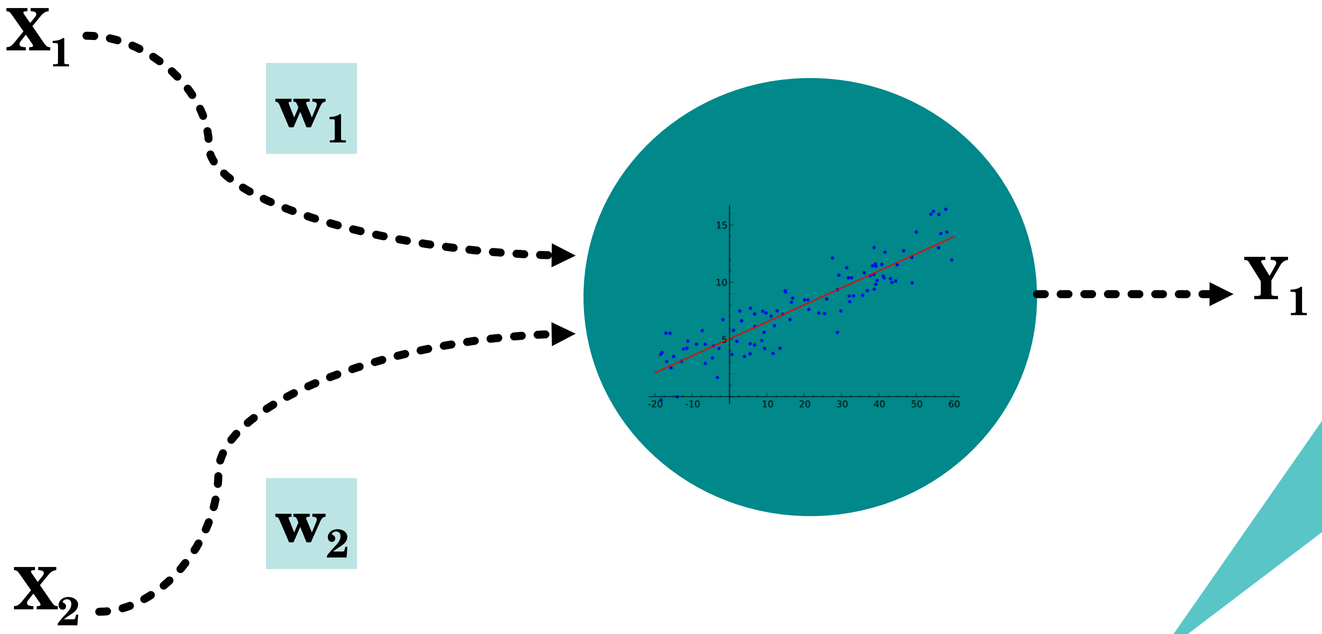
3. Función de la NN



Podemos verlo como una función de regresión lineal en el que una recta o hiperplano se ajusta a nuestros datos de entrada

$$y = w_1x_1 + w_2x_2$$

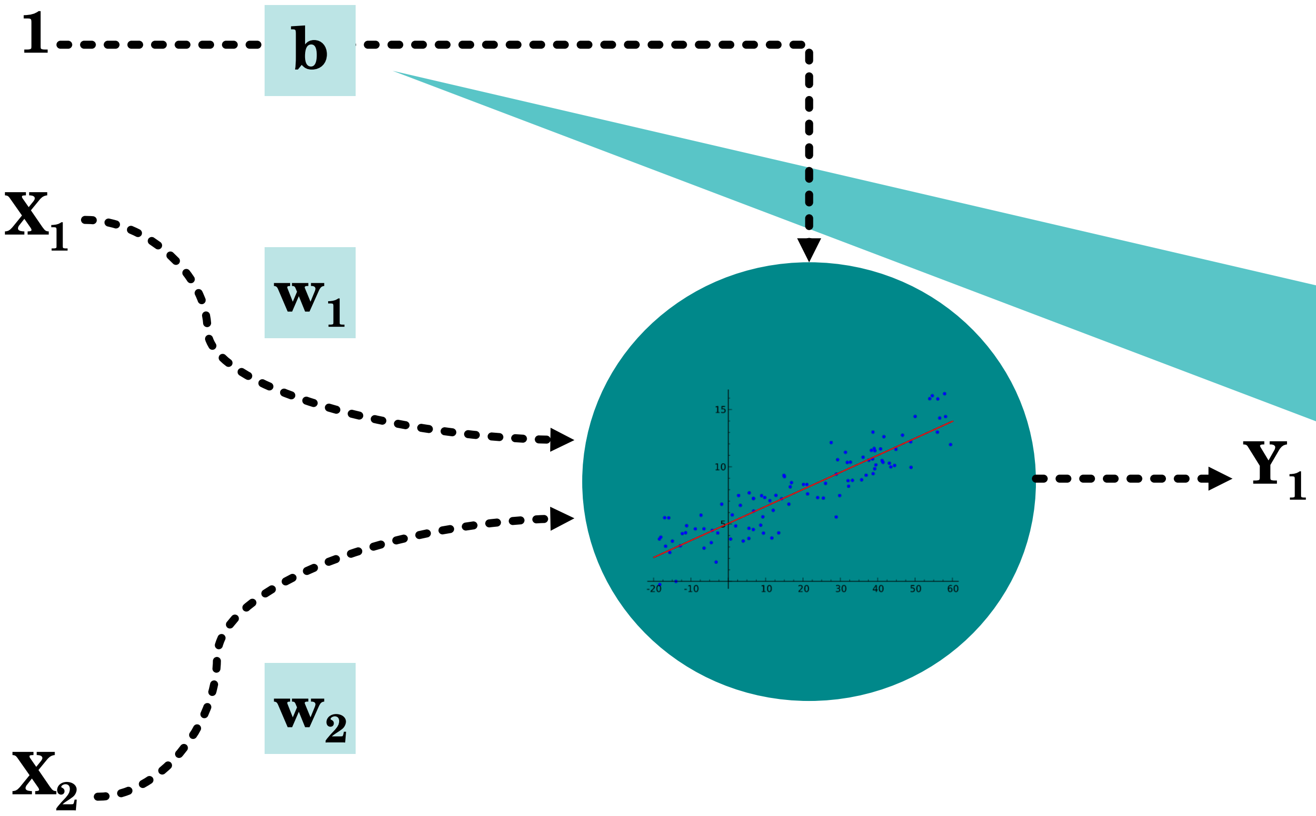
3. Función de la NN



El término independiente, llamado **bias** o **sesgo**, nos permite el desplazamiento de la recta.

$$y = w_1x_1 + w_2x_2 + b$$

3. Función de la NN



Así nuestra neruona funcionaría exactamente como un **modelo de regresión**

$$y = w_1x_1 + w_2x_2 + b$$

Simulación

ETS de Ingeniería Informática

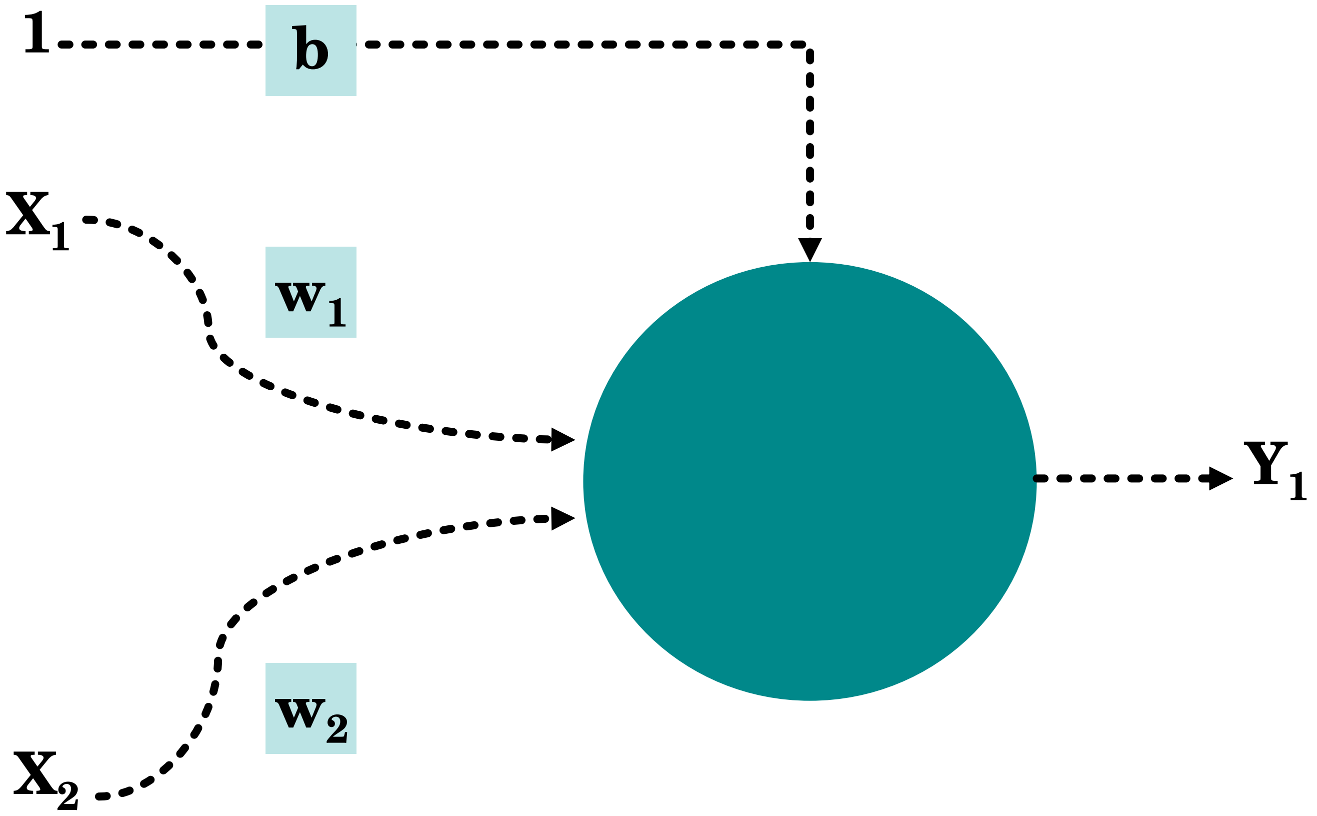
UNED

1. Ejemplo

Estudiar + Estar en casa = Aprobar el examen

1. Ejemplo

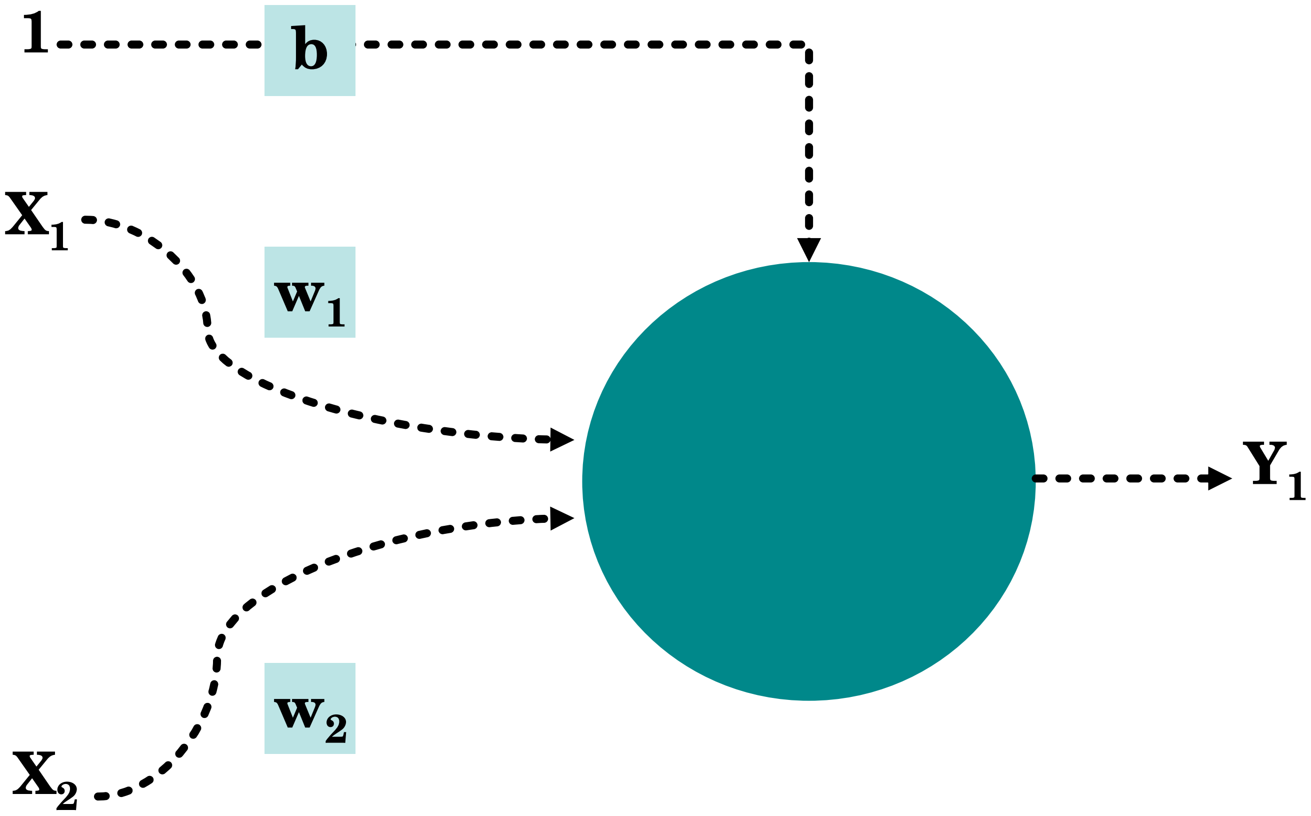
Estudiar + Estar en casa = Aprobar el examen



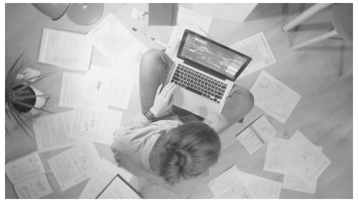
$$y = w_1x_1 + w_2x_2 + b$$

1. Ejemplo

Estudiar + Estar en casa = Aprobar el examen

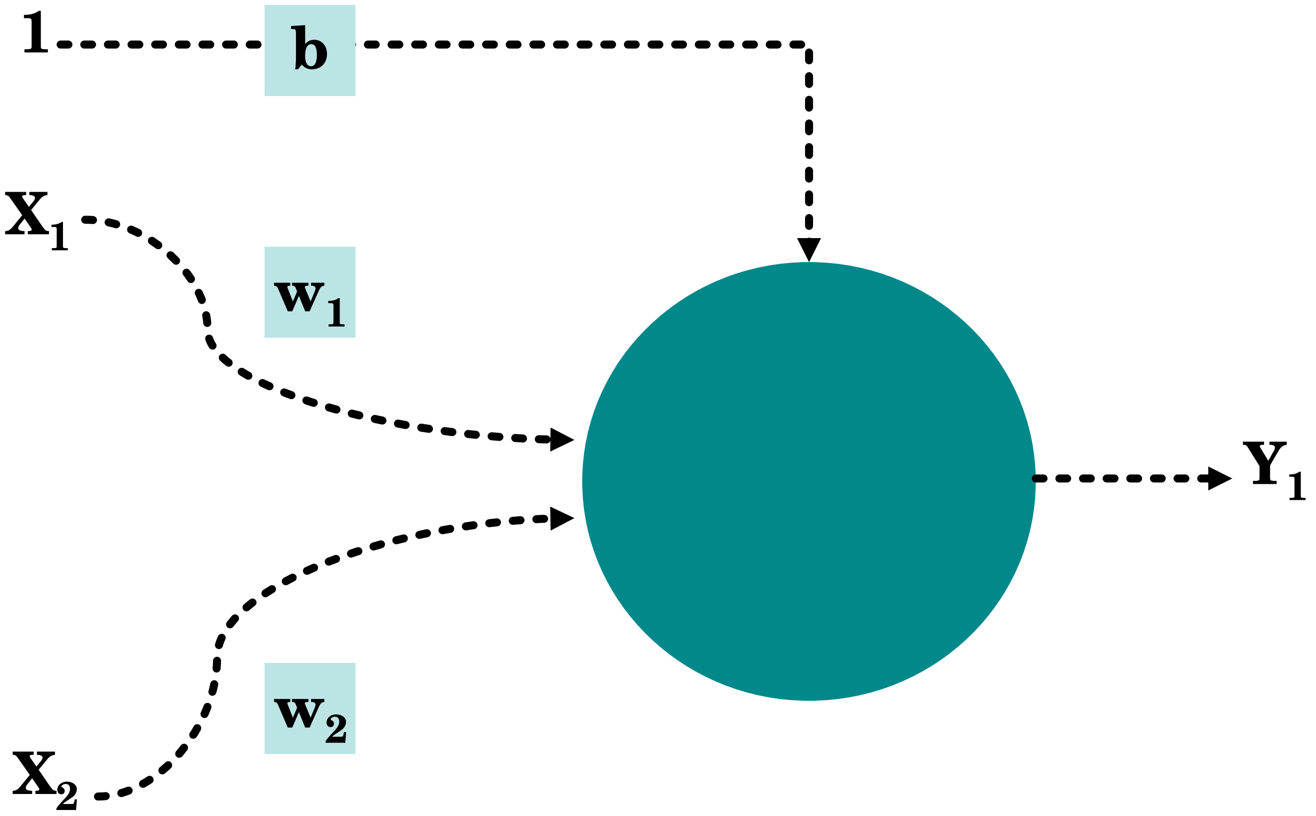


$$y = w_1 x_1 + w_2 x_2 + b$$

	1	0
X_1		
X_2		

1. Ejemplo

Estudiar + Estar en casa = Aprobar el examen

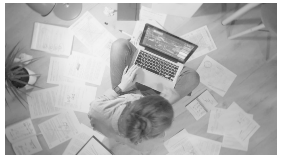
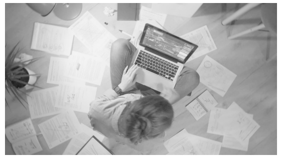


$$y = w_1x_1 + w_2x_2 + b$$

1. Ejemplo

Modelamos el problema

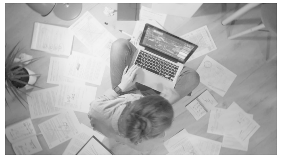
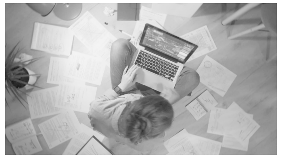
Estudiar + Estar en casa = Aprobar el examen

X_1	X_2	Target Y
		
		
		
		

1. Ejemplo

Modelamos el problema

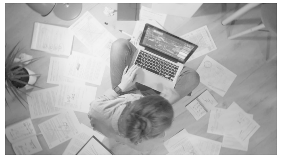
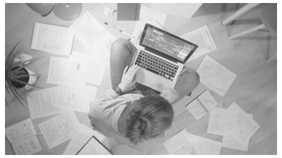
Estudiar + Estar en casa = Aprobar el examen

X_1	X_2	Target Y
		
		
		
		

1. Ejemplo

Modelamos el problema

Estudiar + Estar en casa = Aprobar el examen

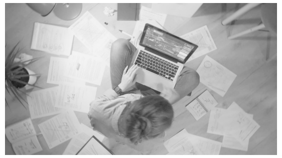
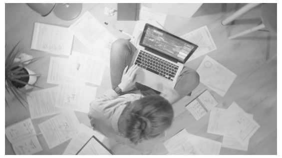
X_1	X_2	Target	Y
			?
			?
			?
			?

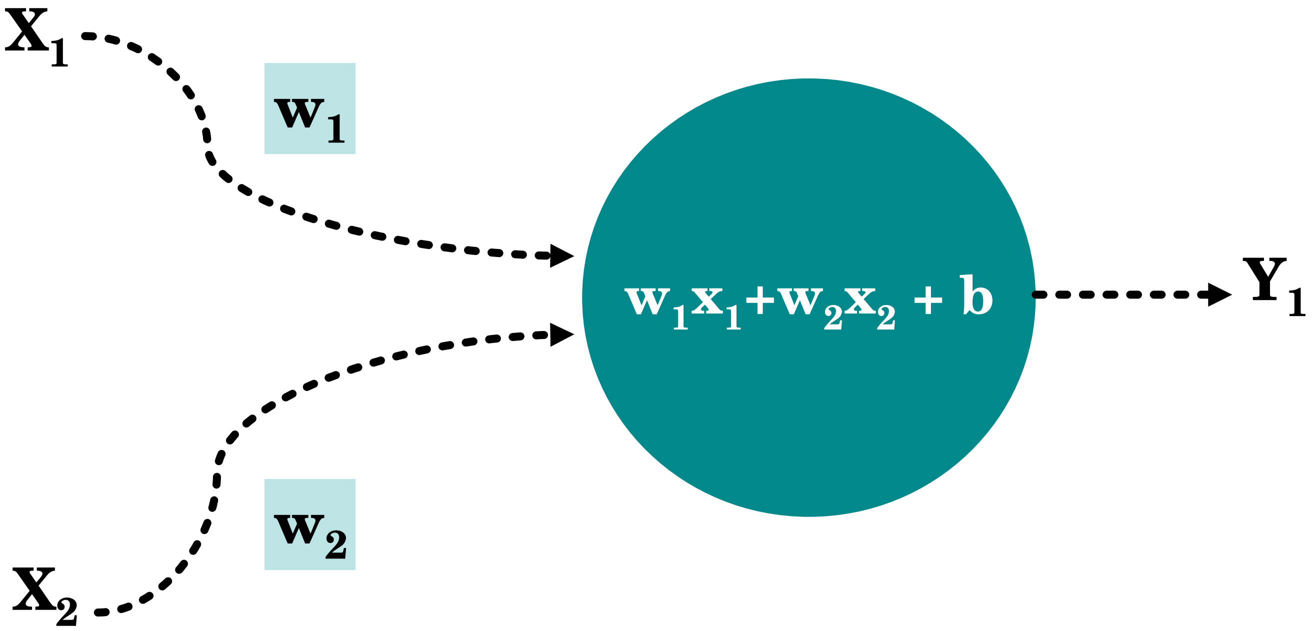
El resultado de la NN estará condicionado por el valor de los **parámetros**. Por tanto, tenemos que buscar esos pesos y bias.

1. Ejemplo

Buscamos los parámetros

Estudiar + Estar en casa = Aprobar el examen

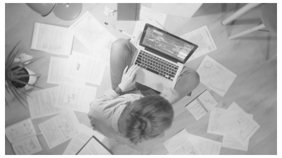
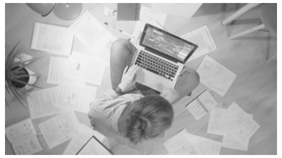
X_1	X_2	Target	Y
			?
			?
			?
			?

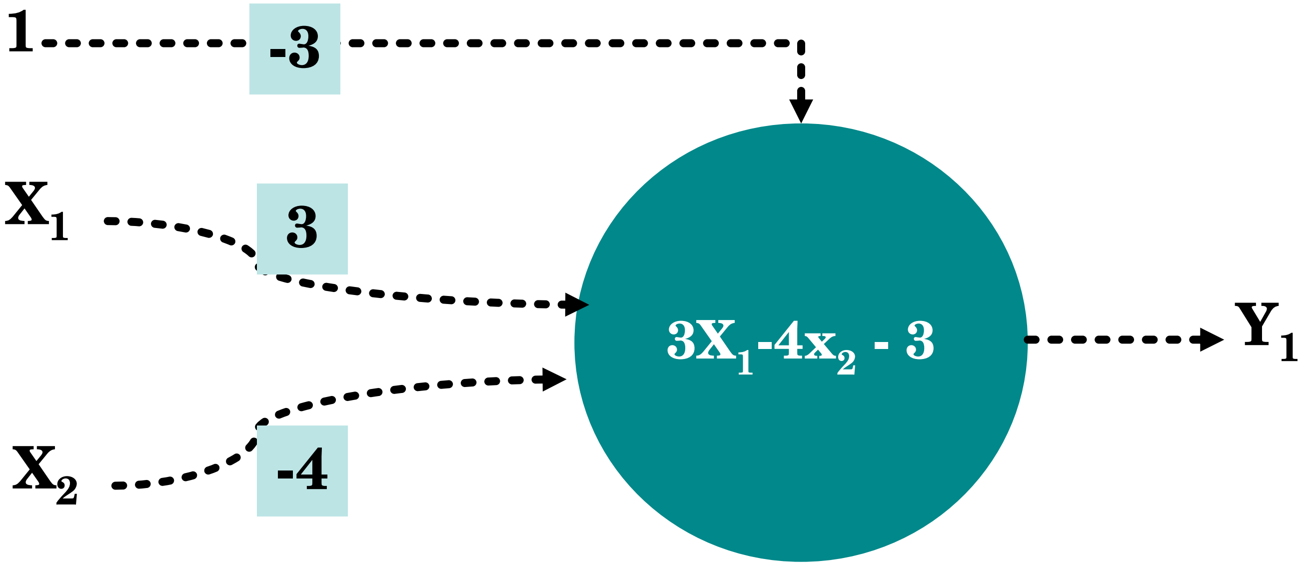


1. Ejemplo

Buscamos los parámetros

Estudiar + Estar en casa = Aprobar el examen

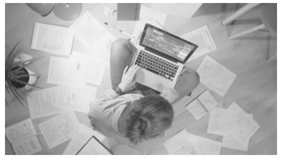
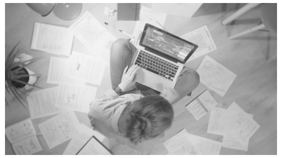
X_1	X_2	Target	Y
			-3
			0
			-7
			-4

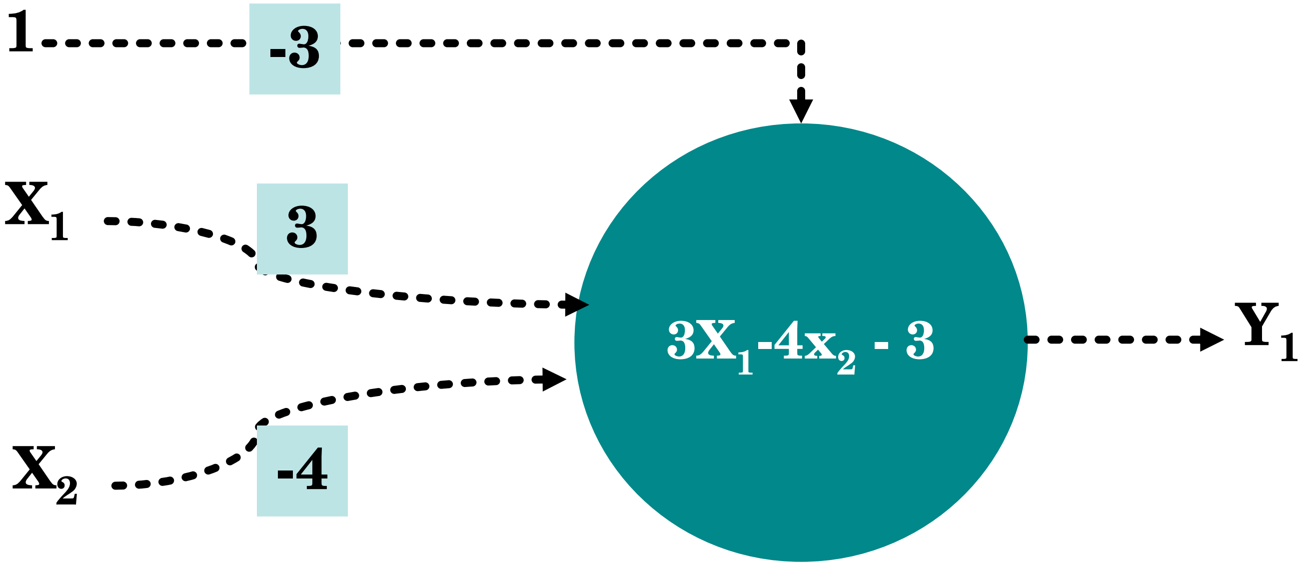


1. Ejemplo

Buscamos los parámetros

Estudiar + Estar en casa = Aprobar el examen

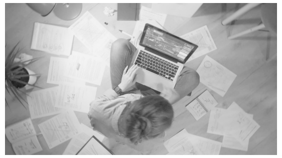
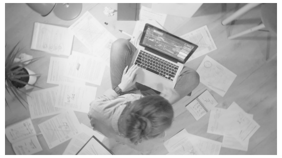
X_1	X_2	Target	Y
			-3
			0
			-7
			-4

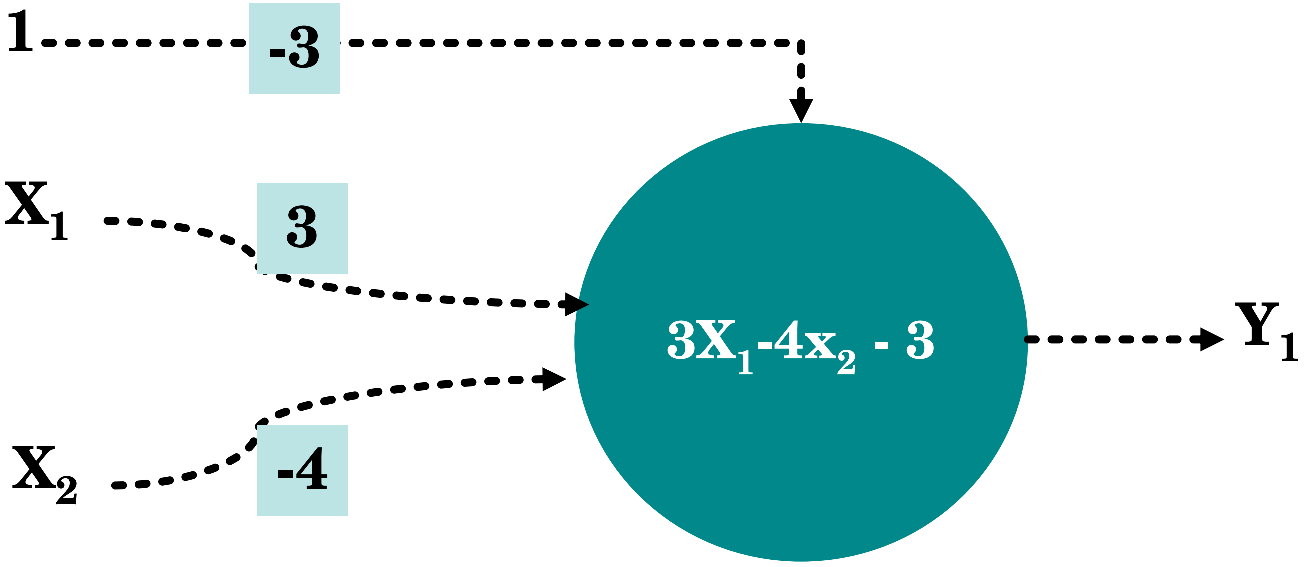


El objetivo es que salga un resultado positivo el último Y negativo los tres primeros

1. Ejemplo
Buscamos los parámetros

Estudiar + Estar en casa = Aprobar el examen

X_1	X_2	Target	Y
			-3
			0
			-7
			-4

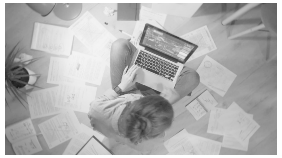
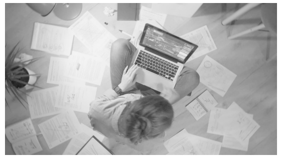


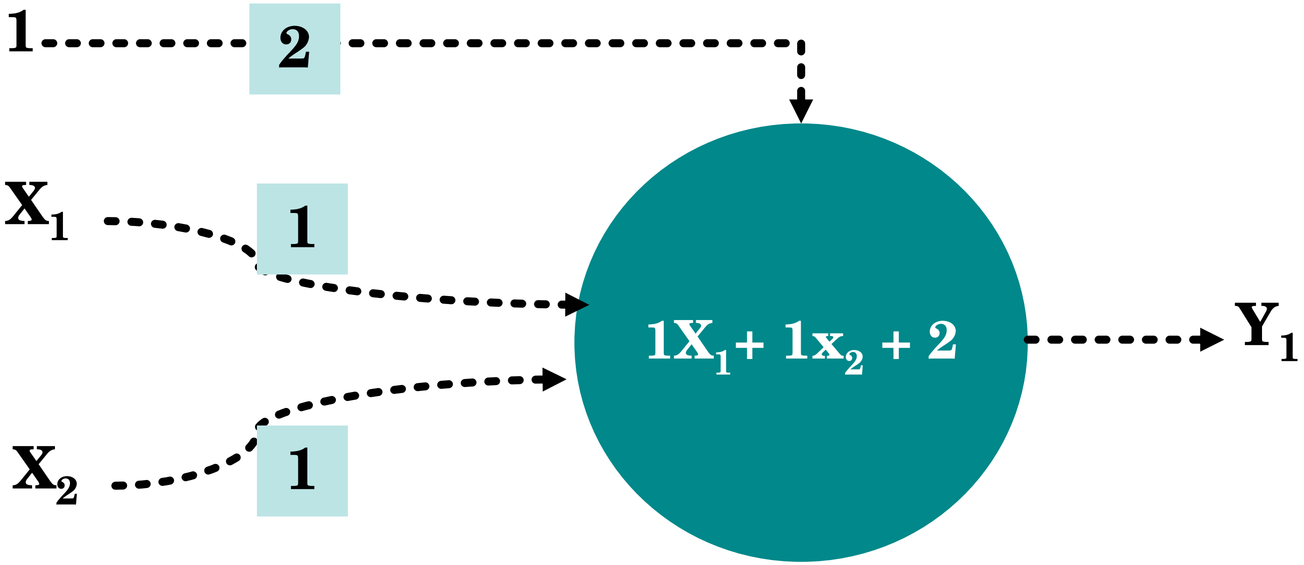
¡No cumple!

1. Ejemplo

Buscamos los parámetros

Estudiar + Estar en casa = Aprobar el examen

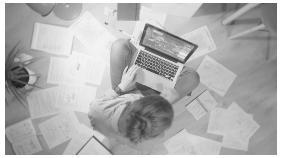
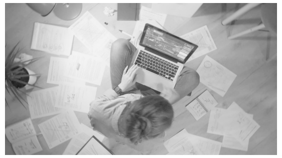
X_1	X_2	Target	Y
			2
			3
			3
			4

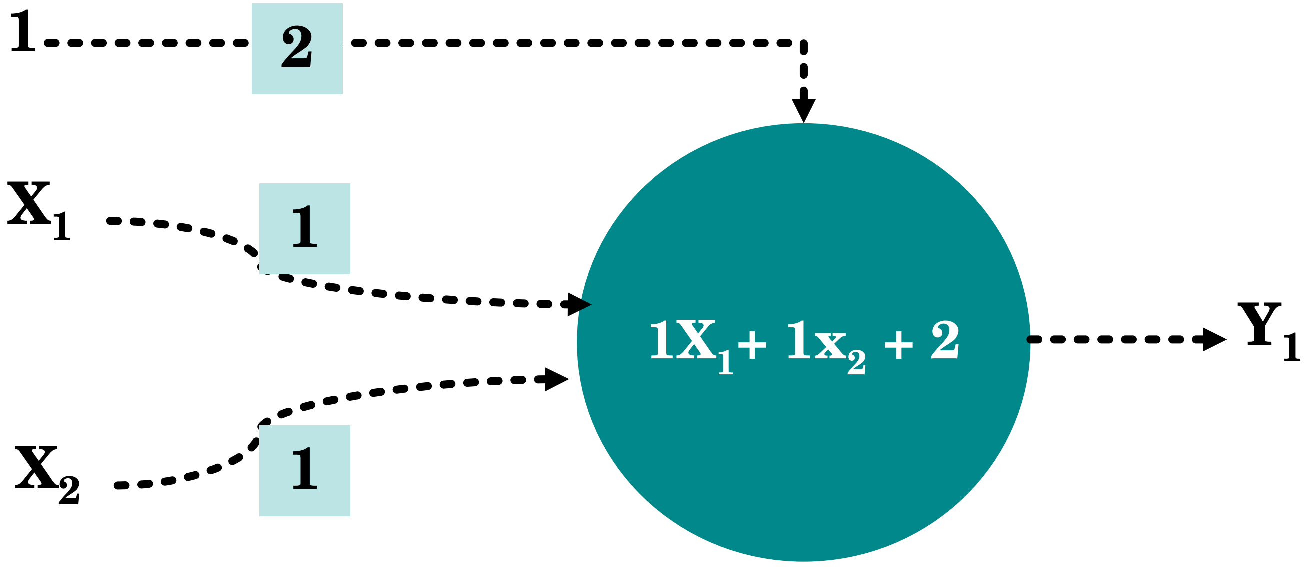


1. Ejemplo

Buscamos los parámetros

Estudiar + Estar en casa = Aprobar el examen

X_1	X_2	Target	Y
			2
			3
			3
			4

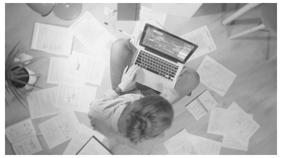
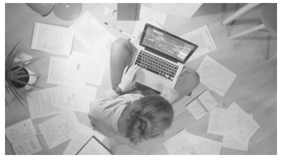


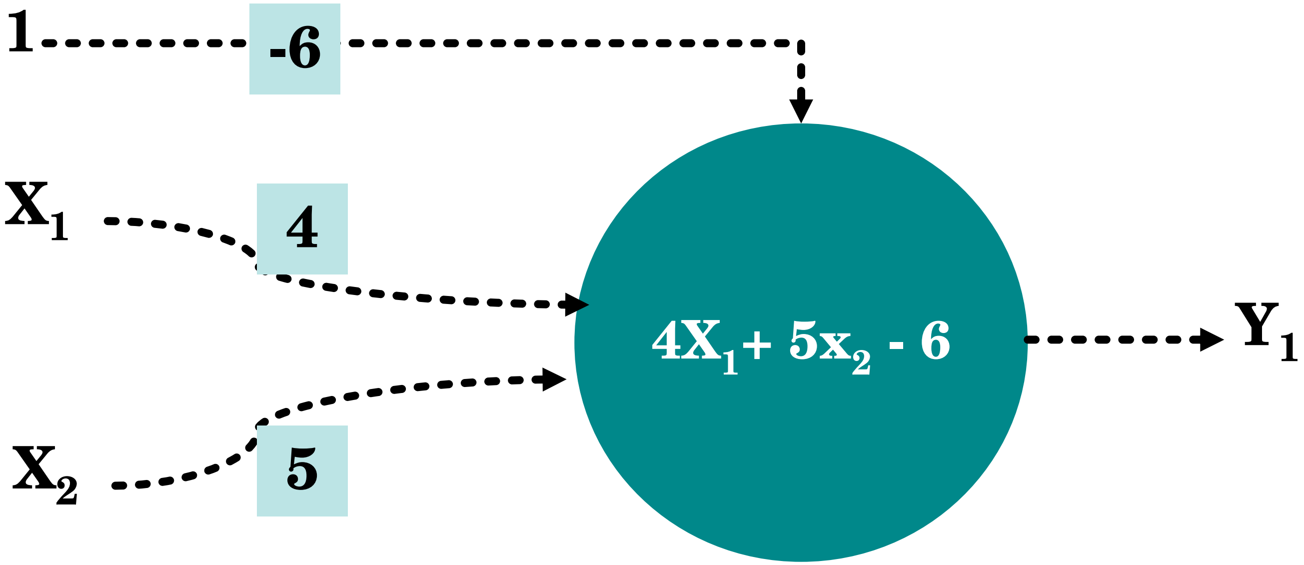
¡No cumple!

1. Ejemplo

Buscamos los parámetros

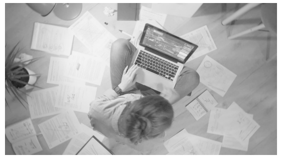
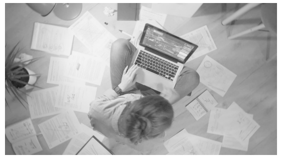
Estudiar + Estar en casa = Aprobar el examen

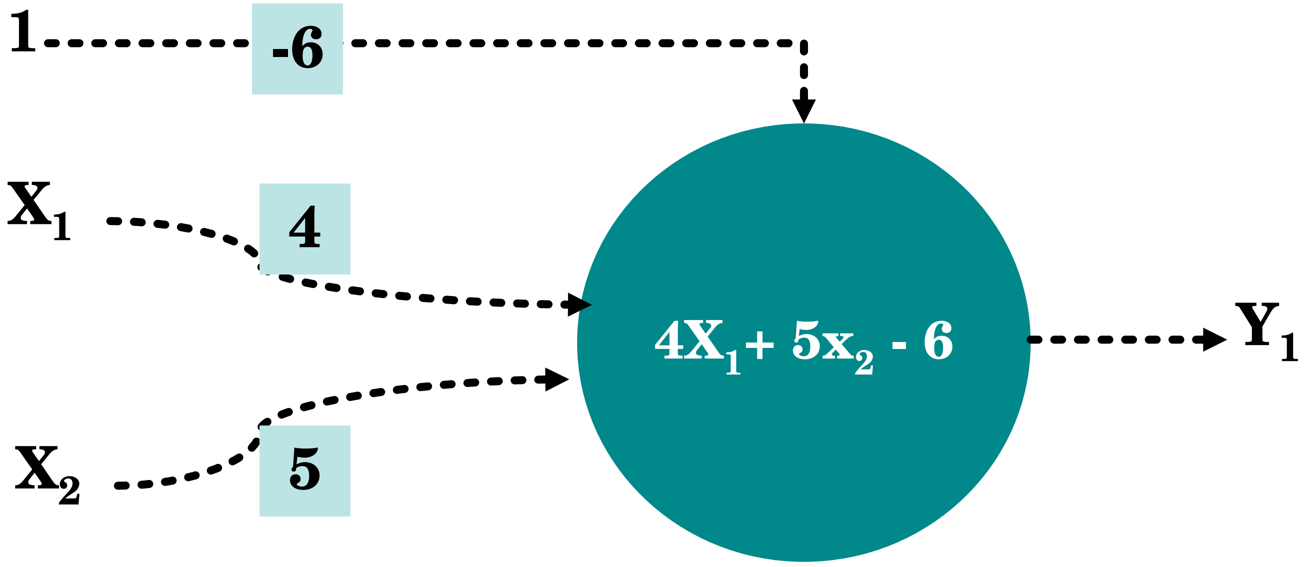
X_1	X_2	Target	Y
			-6
			-2
			-1
			3



1. Ejemplo
Buscamos los parámetros

Estudiar + Estar en casa = Aprobar el examen

X_1	X_2	Target	Y
			-6
			-2
			-1
			3

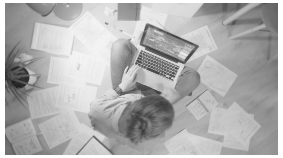


¡SI cumple!

1. Ejemplo

Entender el procedimiento

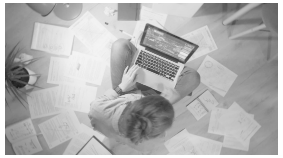
X_1



X_2

1. Ejemplo
Entender el procedimiento


$\mathbf{X}_1$

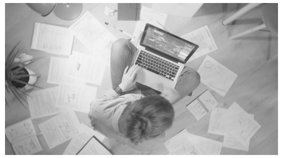


$\mathbf{X}_2$

1. Ejemplo
Entender el procedimiento



x_1



x_2

1. Ejemplo

Entender el procedimiento

	1	0	1
x_1			
	0	0	0
		0	1
		x_2	

1. Ejemplo

Entender el procedimiento

x_1 \ x_2	0	1
1	**0**	**1**
0	**0**	**0**

¿Qué observamos?

1. Ejemplo

Puerta lógica AND

1 **0** **1**

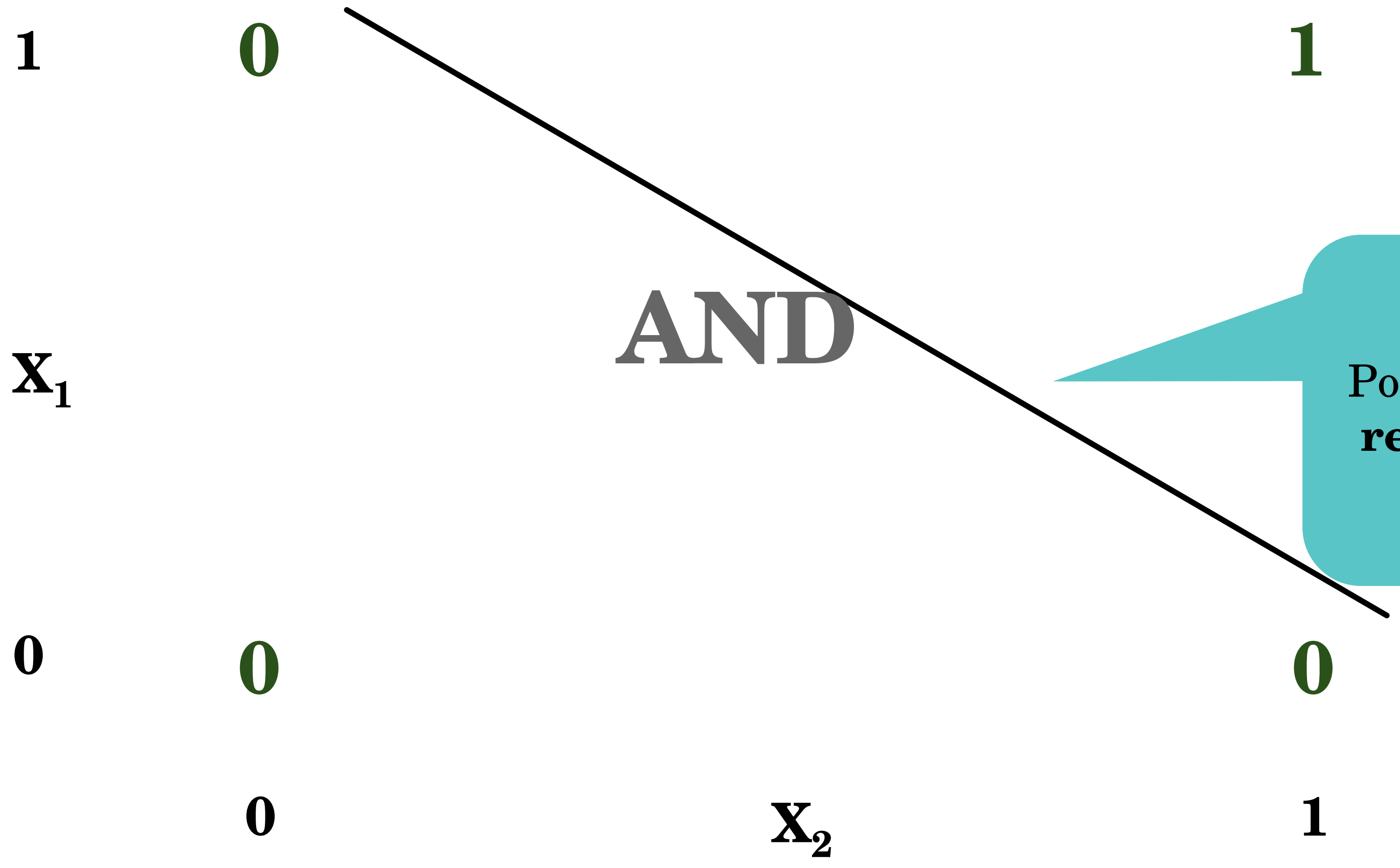
X_1

¿Qué observamos?
Que tenemos una **puerta lógica AND**

0 **0** **0**

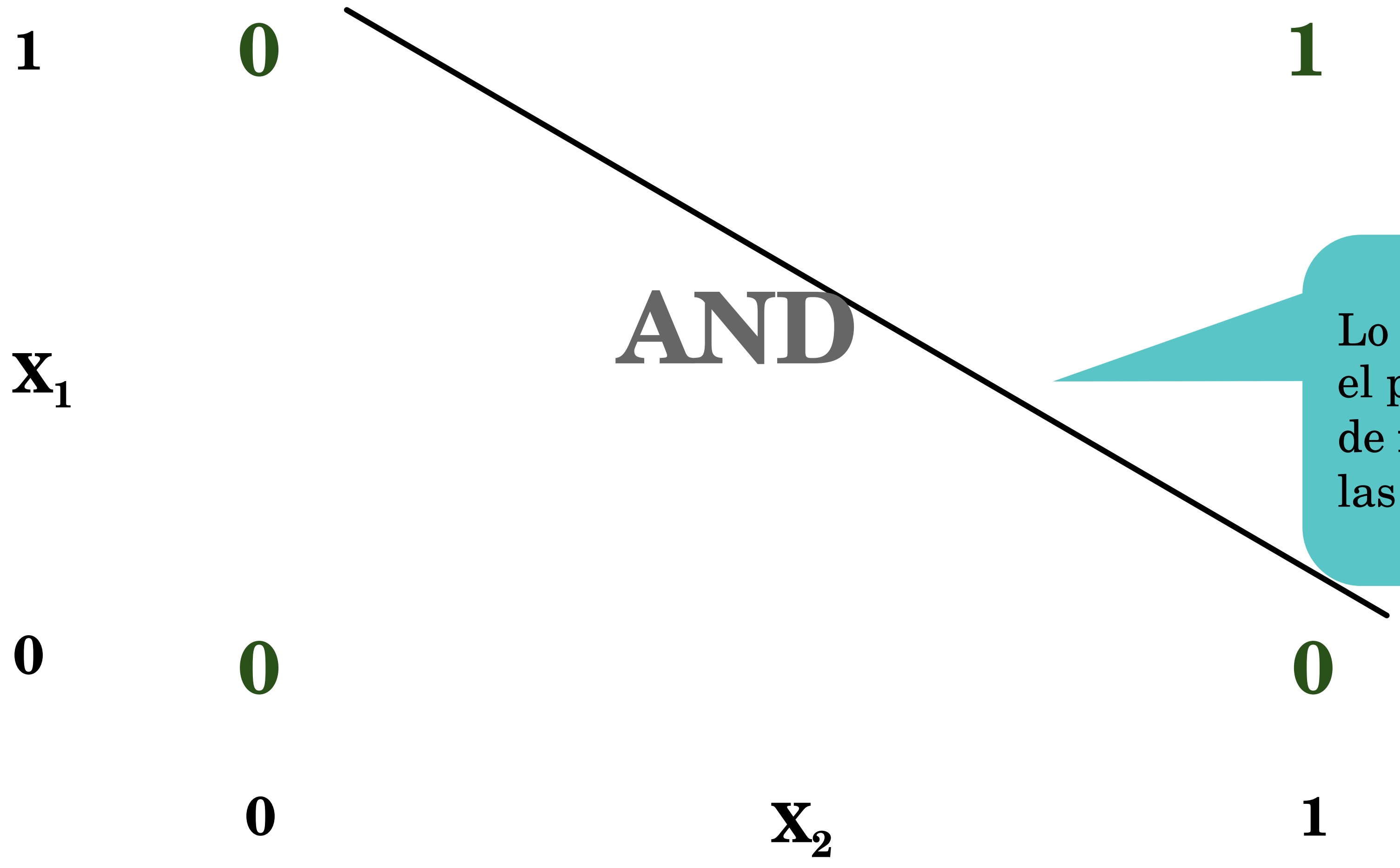
0 X_2 1

1. Ejemplo
Puerta lógica AND



Podemos establecer la **recta de regresión** que hemos hablado

1. Ejemplo
Puerta lógica AND



Lo podemos entender como el problema de trazar una línea de **frontera** en la que separar las clases

1. Ejemplo
Puerta lógica OR

1	1		1
x_1		OR	
0	0		1
	0	x_2	1

1. Ejemplo

Puerta lógica OR

1 1 1

x_1 **OR**

Tendríamos que ajustar los parámetros de la neurona hasta ajustar los resultados esperados

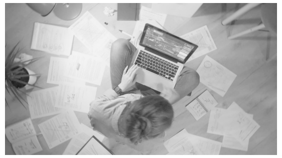
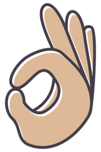
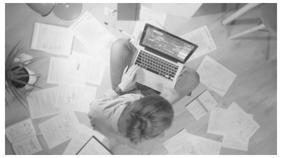
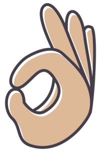
0 0 1

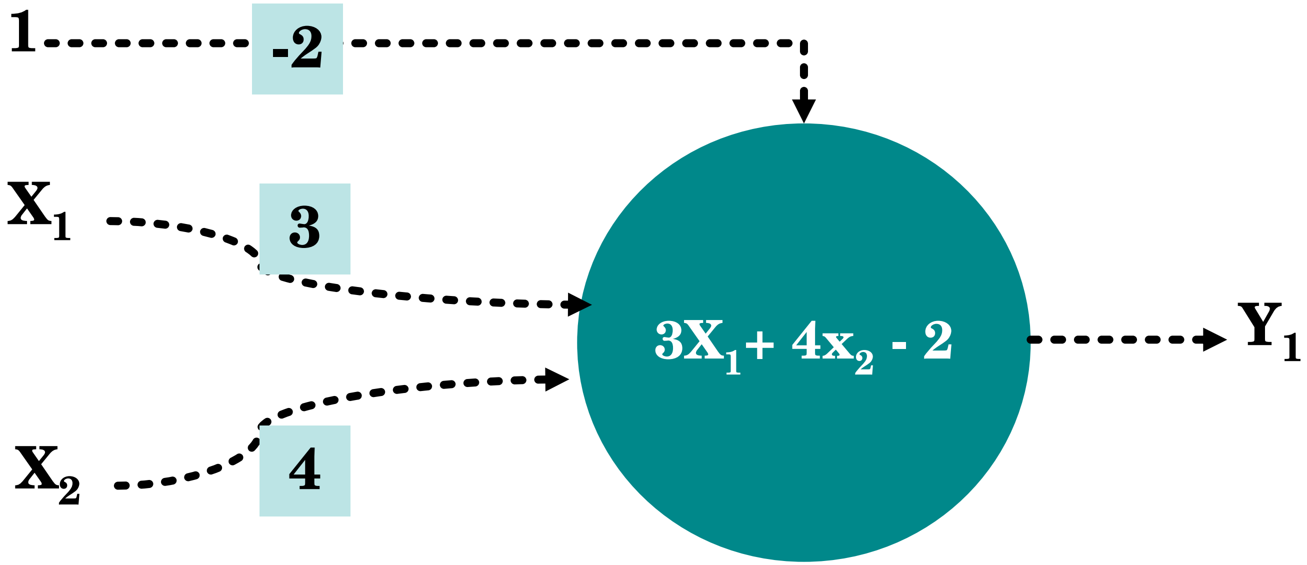
0 x_2 1

1. Ejemplo

Puerta lógica OR – Buscar parámetros

Estudiar + Estar en casa = Aprobar el examen

X_1	X_2	Target	Y
			-2
			1
			2
			5

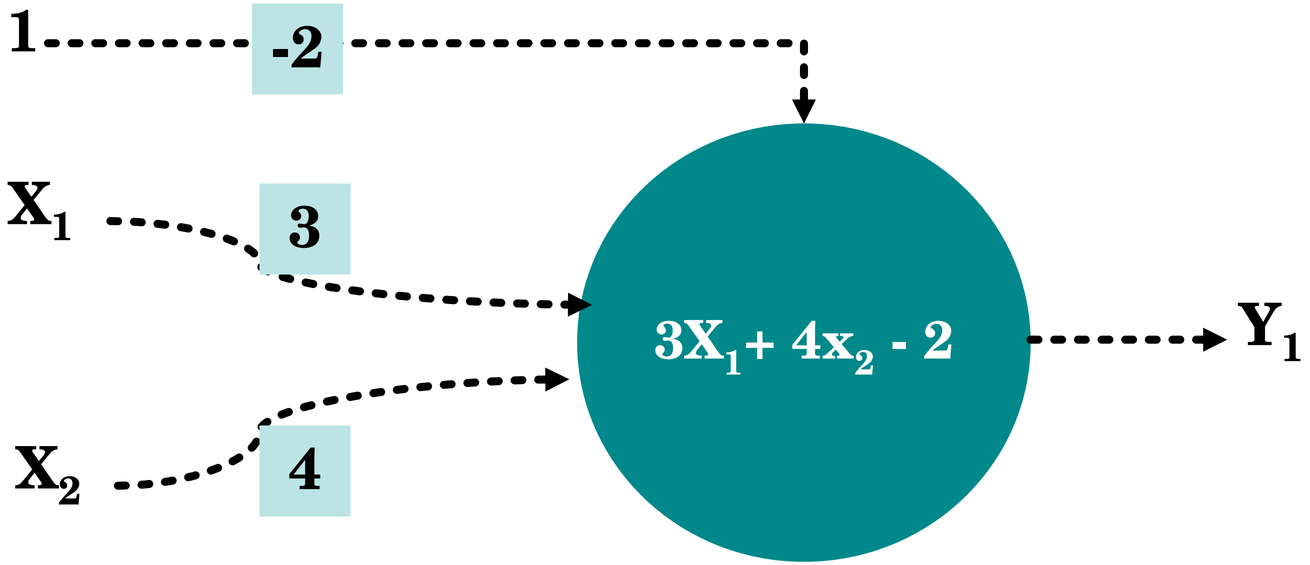


1. Ejemplo

Puerta lógica OR – Buscar parámetros

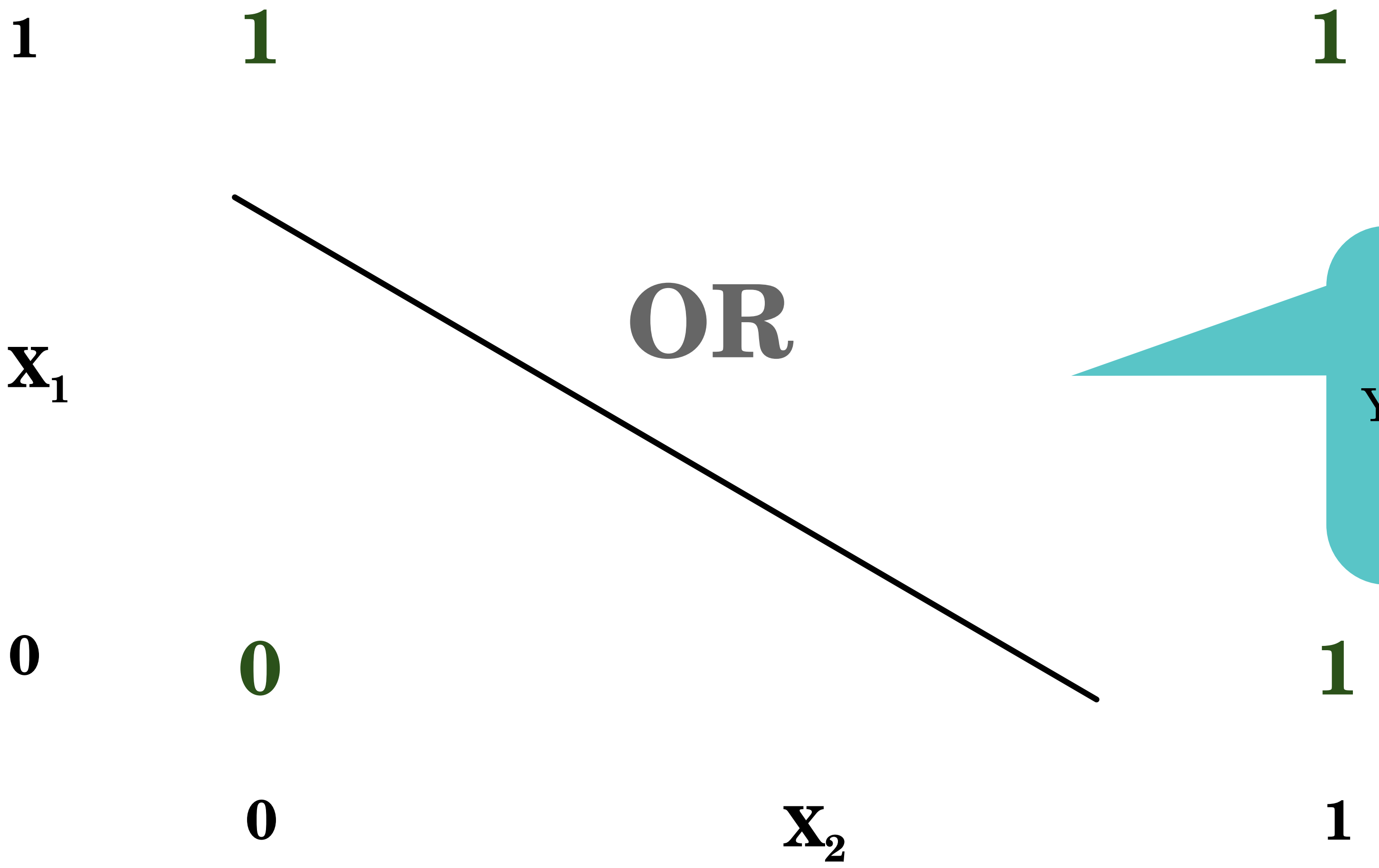
Estudiar + Estar en casa = Aprobar el examen

X_1	X_2	Target	Y
			-2
			1
			2
			5



¡SI cumple!

1. Ejemplo
Puerta lógica OR



Ya tenemos la línea de frontera

1. Ejemplo
Puerta lógica XOR

1 1 0

x_1 **XOR**

¿Como sería con XOR?
¿Recordáis la sesión anterior?

0 0 1

0 x_2 1

1. Ejemplo
Puerta lógica XOR

1 1 0

XOR

x_1

Es imposible modelar el problema con una única neurona, i.e., es imposible **separar linealmente** ambas clases

0 0 1

0 x_2 1

1. Ejemplo

Puerta lógica XOR

1 **1** **0**

x_1 **XOR**

¿Como lo solucionamos?

0 **0** **1**

0 x_2 1

1. Ejemplo
Puerta lógica XOR

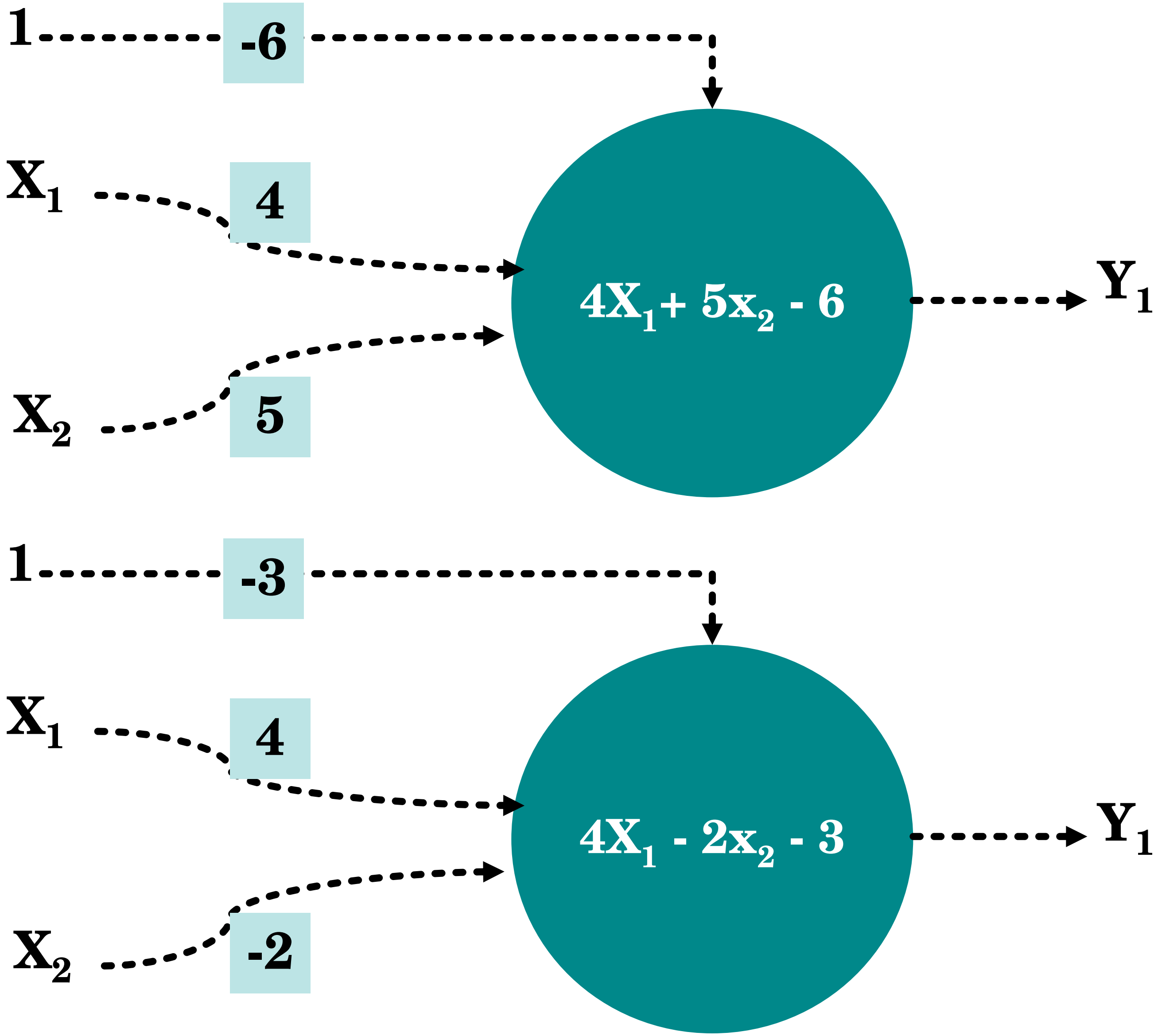
1 1 0

XOR

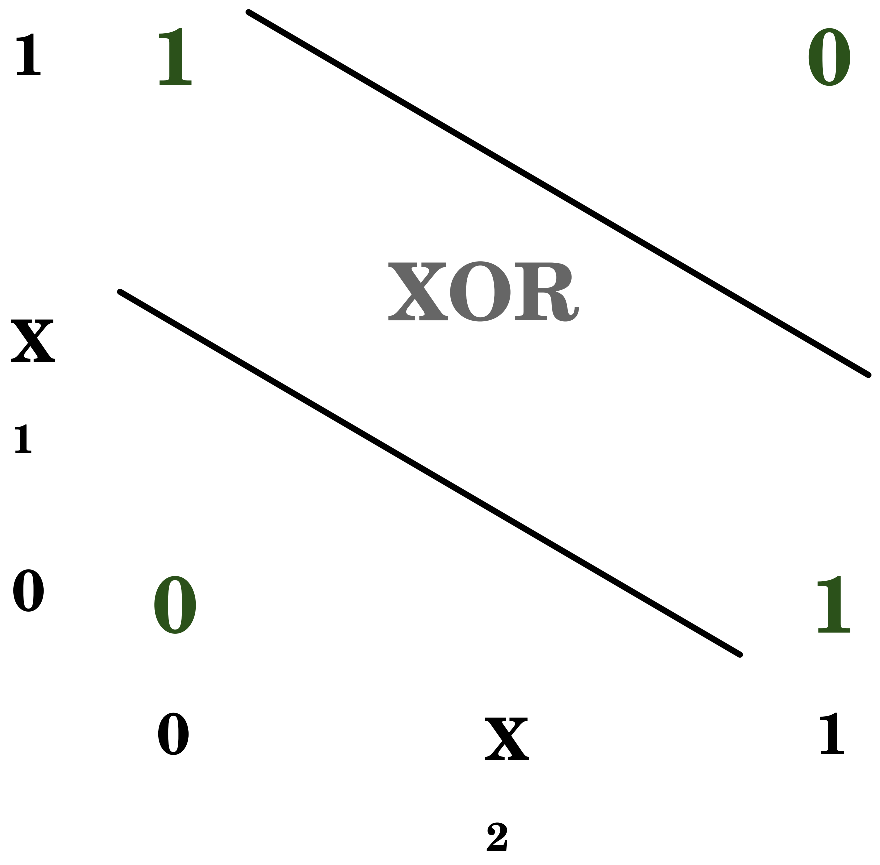
X_1

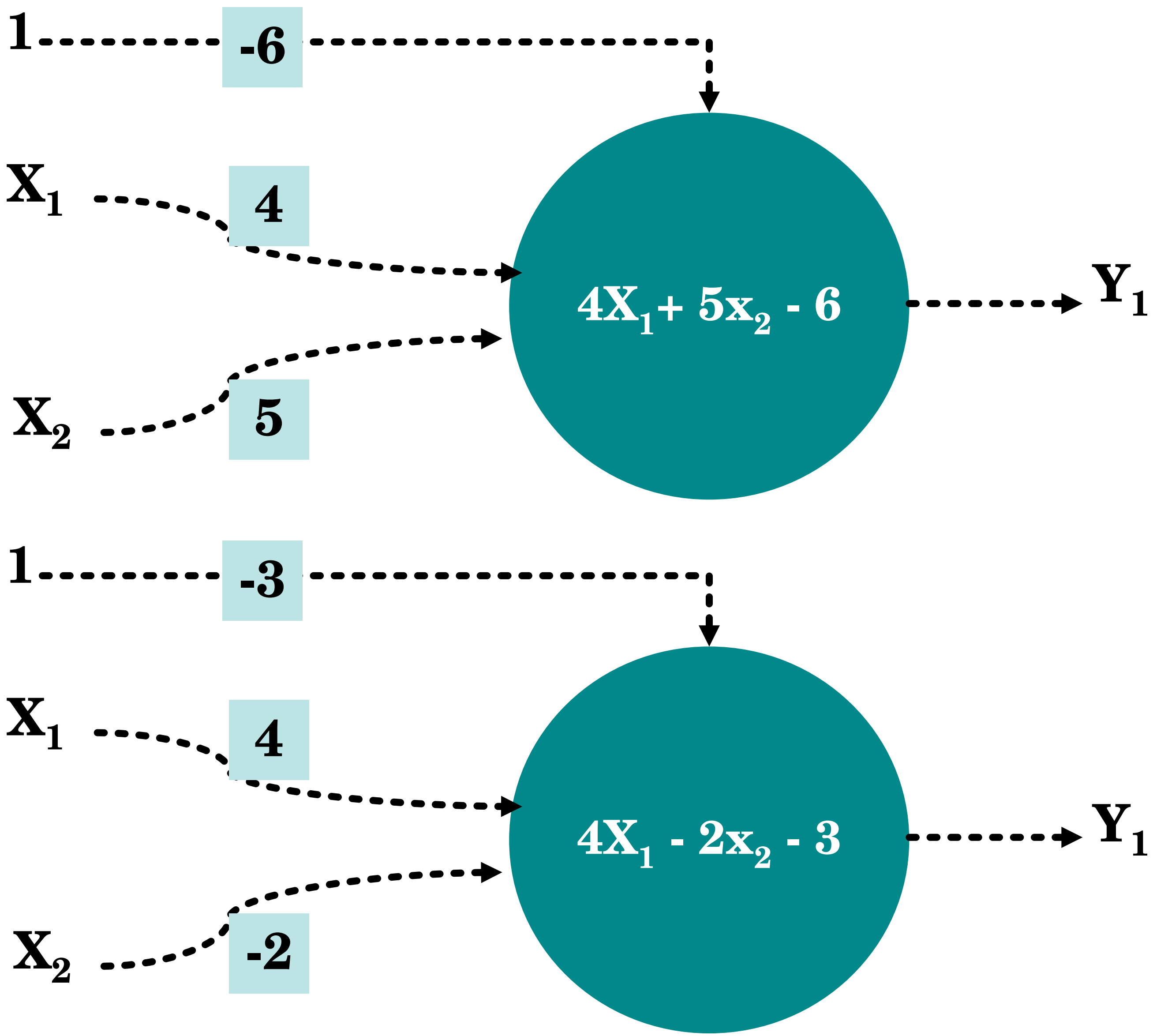
0 0 1

0 X_2 1



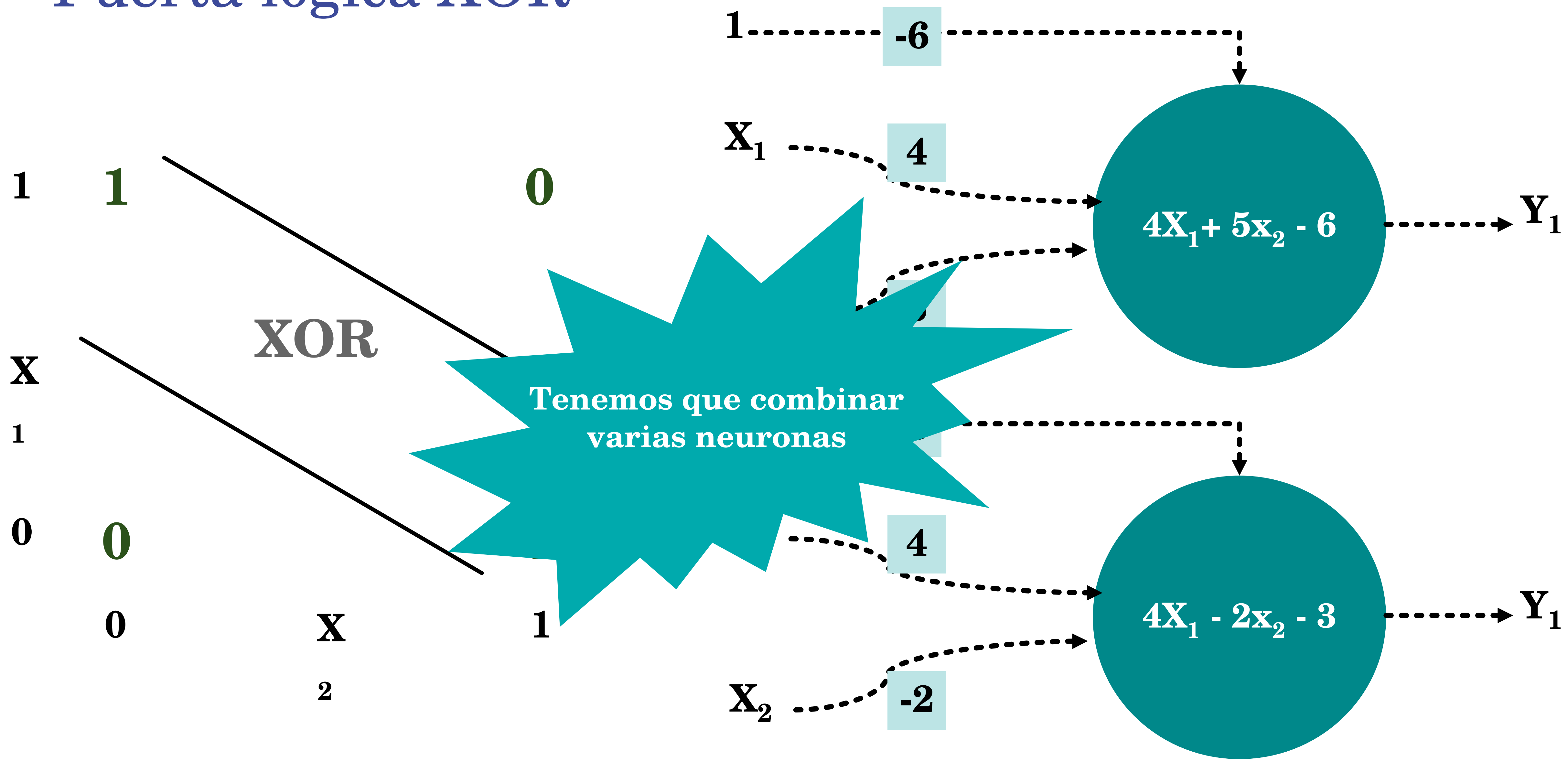
1. Ejemplo
Puerta lógica XOR





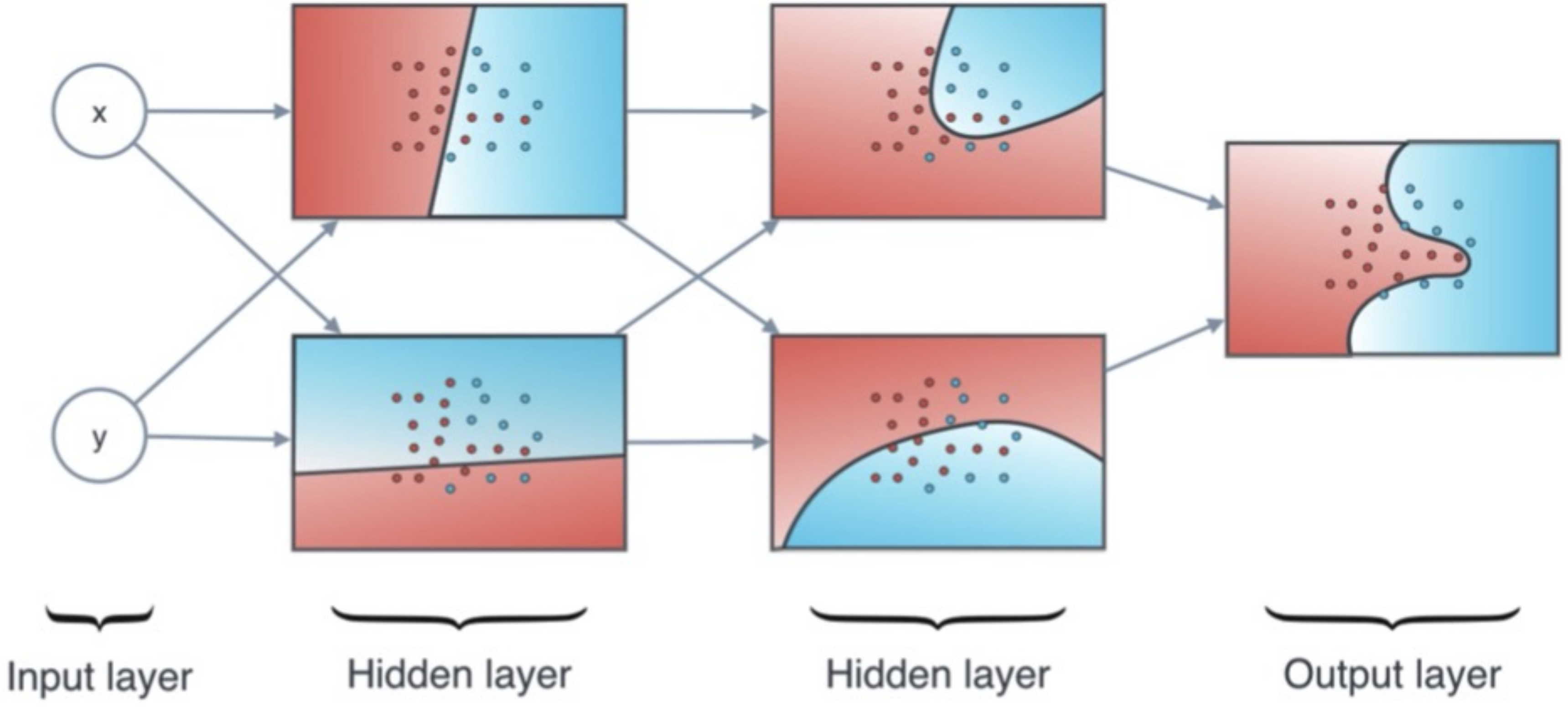
1. Ejemplo

Puerta lógica XOR



1. Ejemplo
Arquitectura de redes neuronales

- Son la base de las **redes profundas**, actualmente muy utilizadas con mucho éxito.
- Cada **capa extrae características** cada vez más complejas



Perceptrón multicapa

ETS de Ingeniería Informática

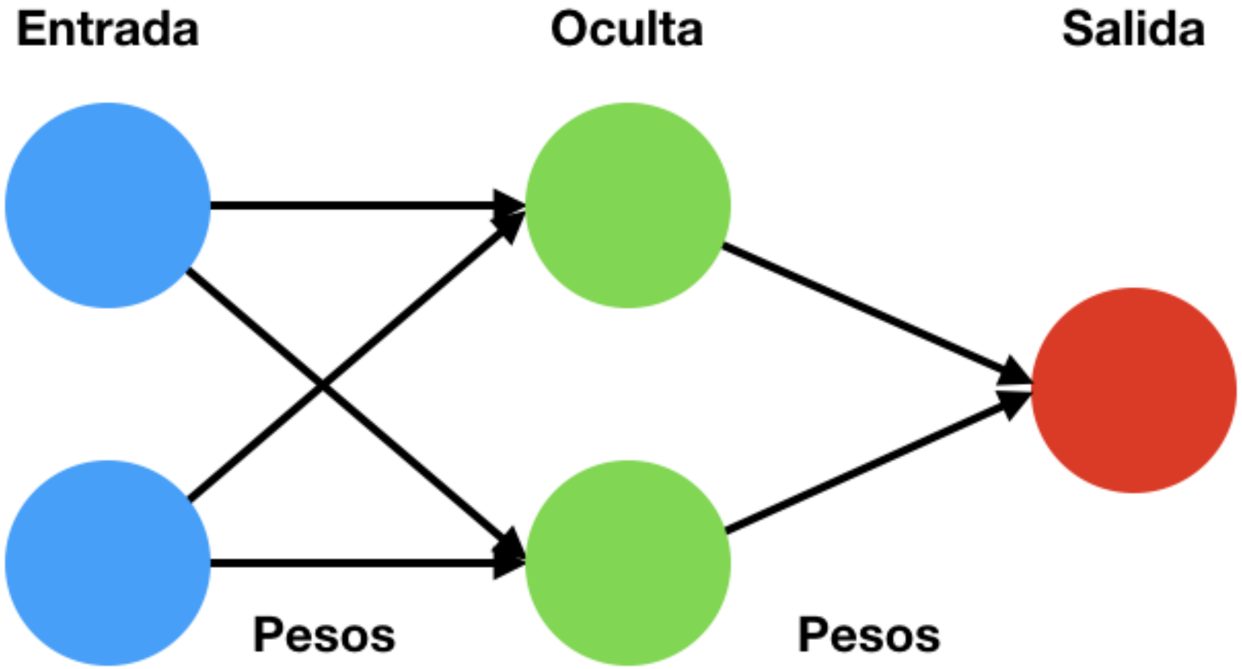
UNED

1. Definición

- Una neurona **no puede representar** toda la información
  - Problema de linealidad como XOR
- Necesitamos una **agrupación** de neuronas

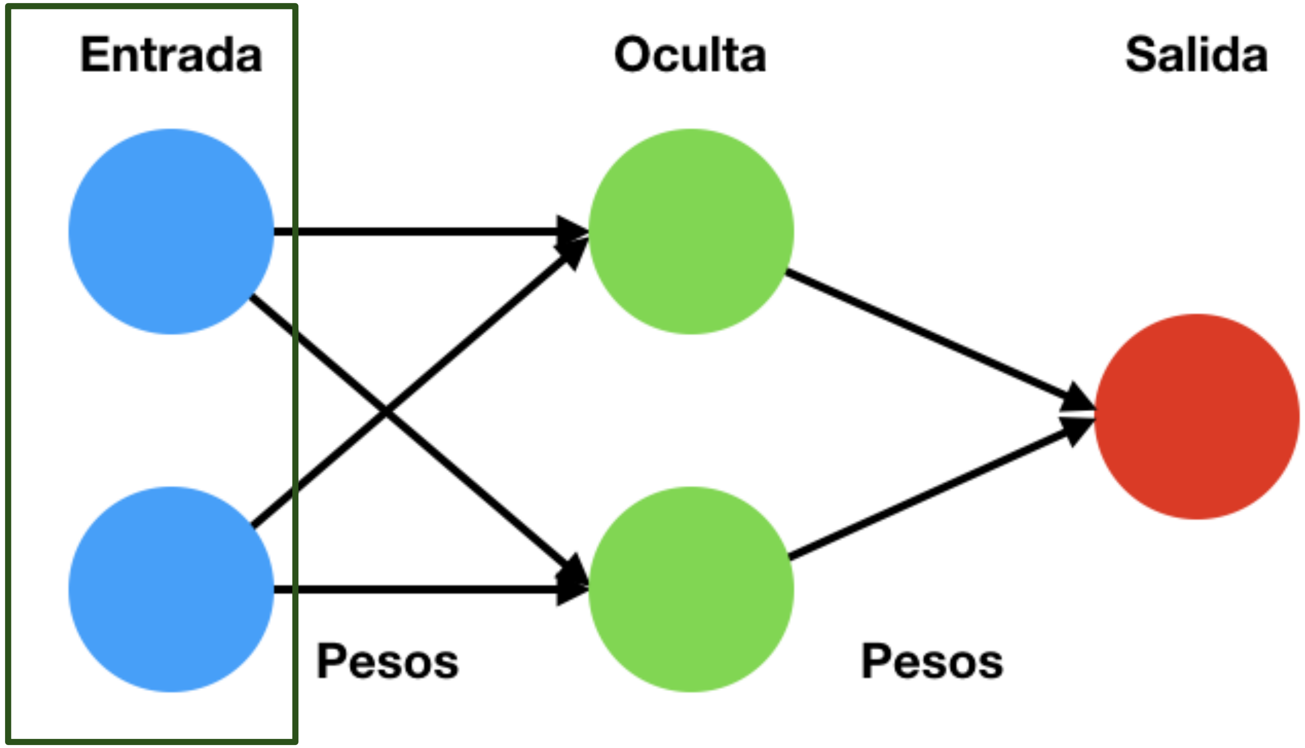
1. Definición

- Una neurona **no puede representar** toda la información
  - Problema de linealidad como XOR
- Necesitamos una **agrupación** de neuronas



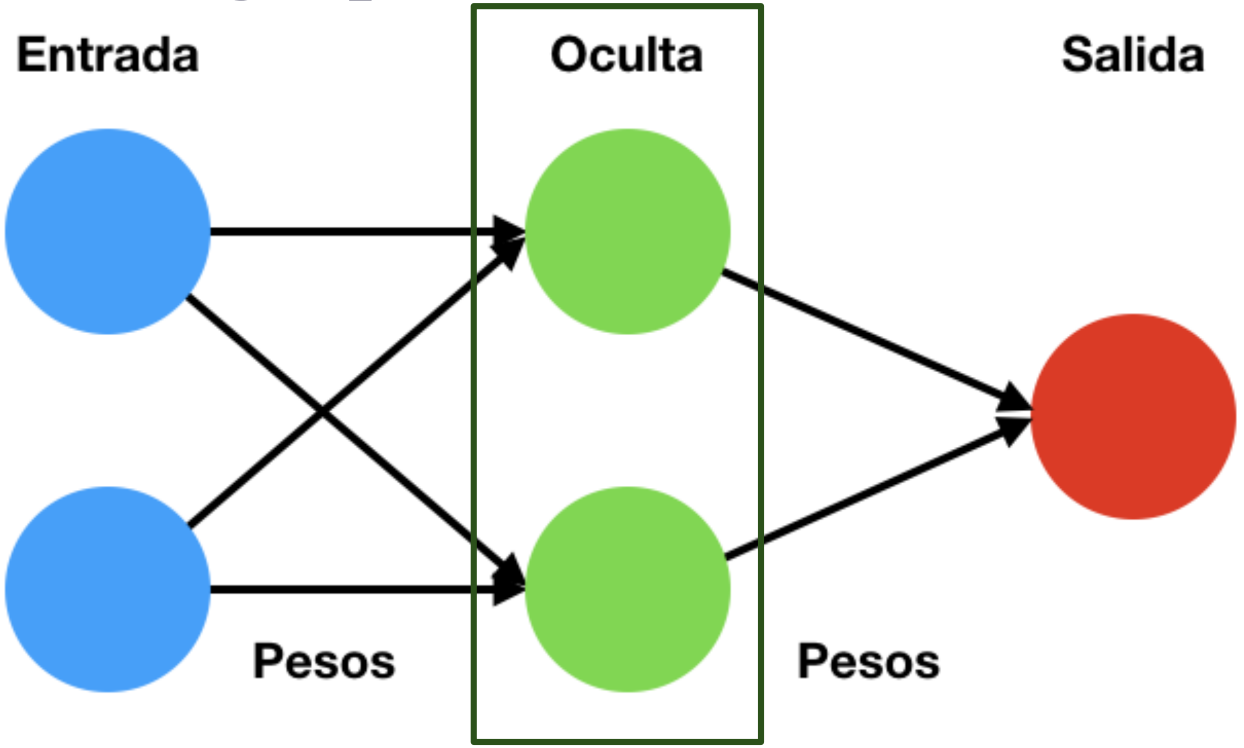
1. Definición

- Una neurona **no puede representar** toda la información
  - Problema de linealidad como XOR
- Necesitamos una **agrupación** de neuronas



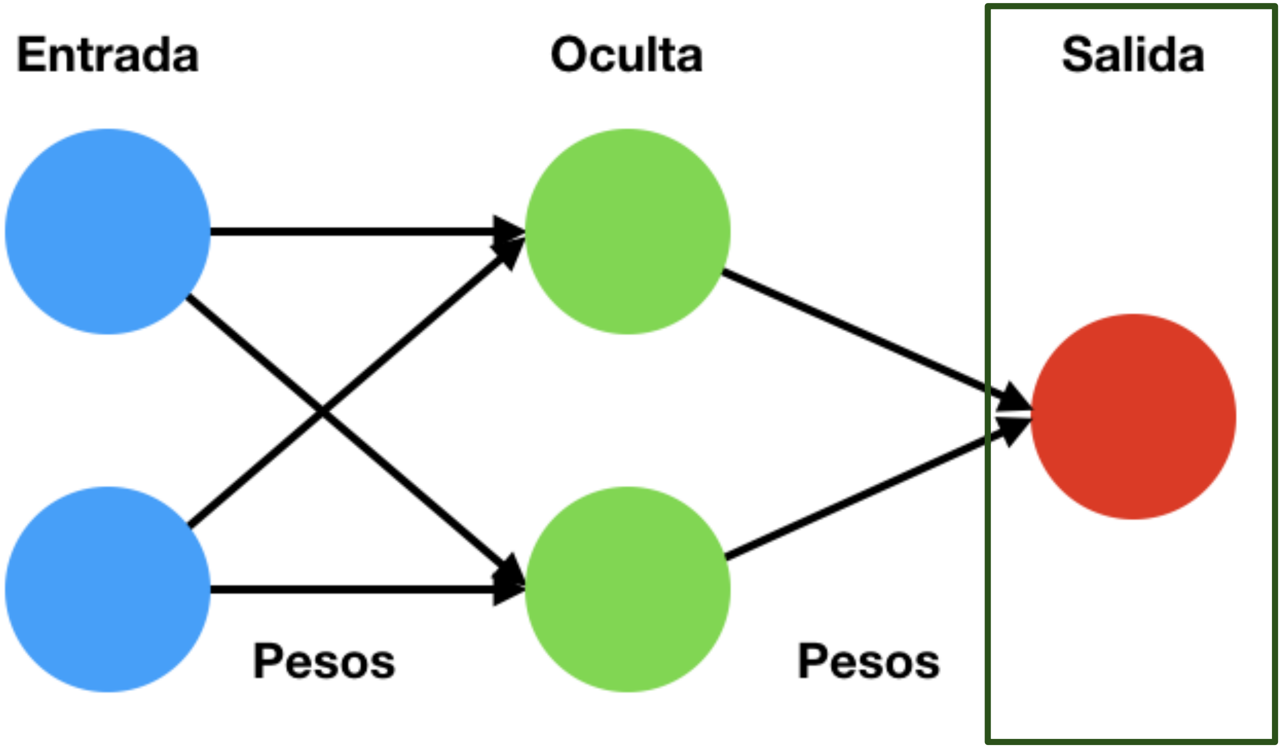
1. Definición

- Una neurona **no puede representar** toda la información
  - Problema de linealidad como XOR
- Necesitamos una **agrupación** de neuronas



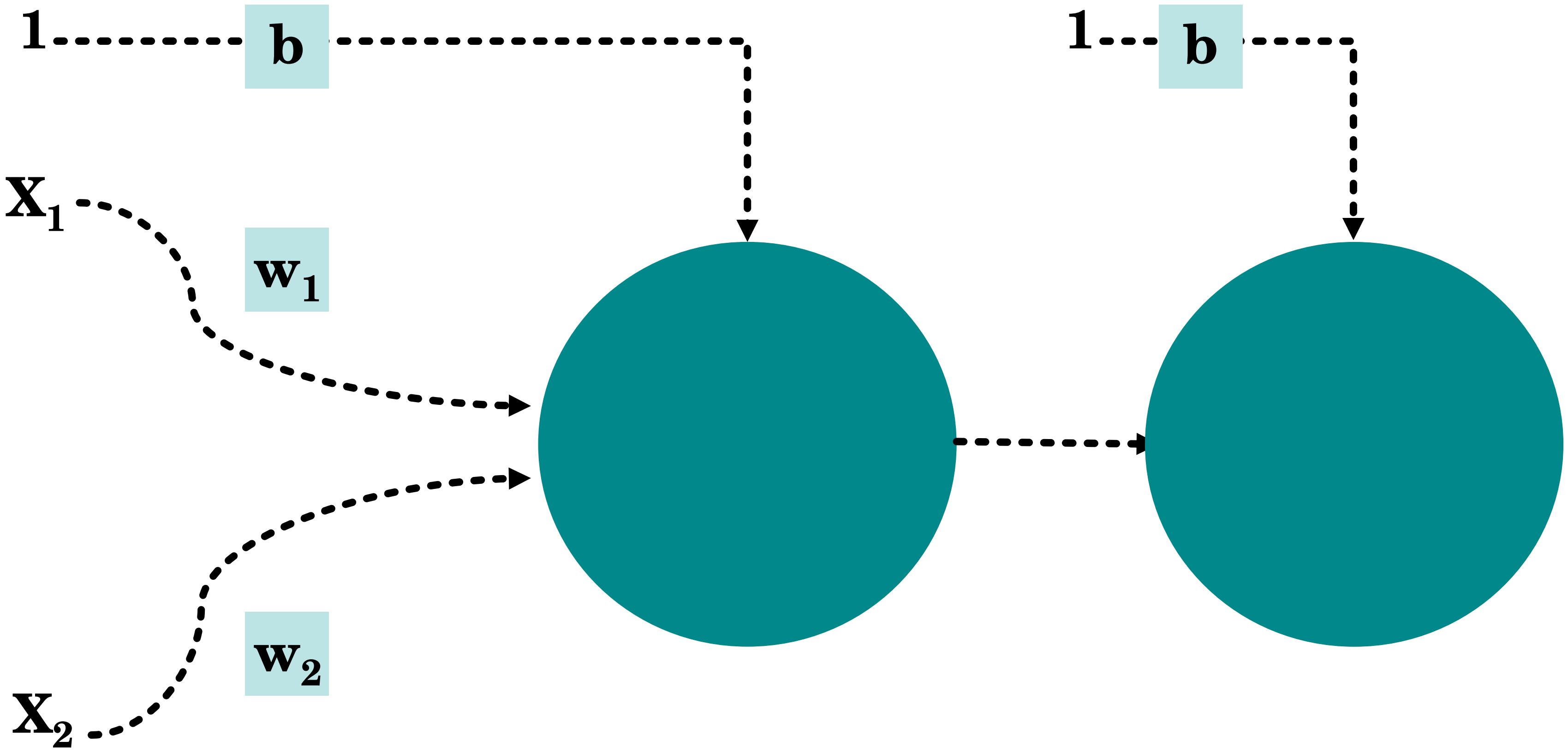
1. Definición

- Una neurona **no puede representar** toda la información
  - Problema de linealidad como XOR
- Necesitamos una **agrupación** de neuronas



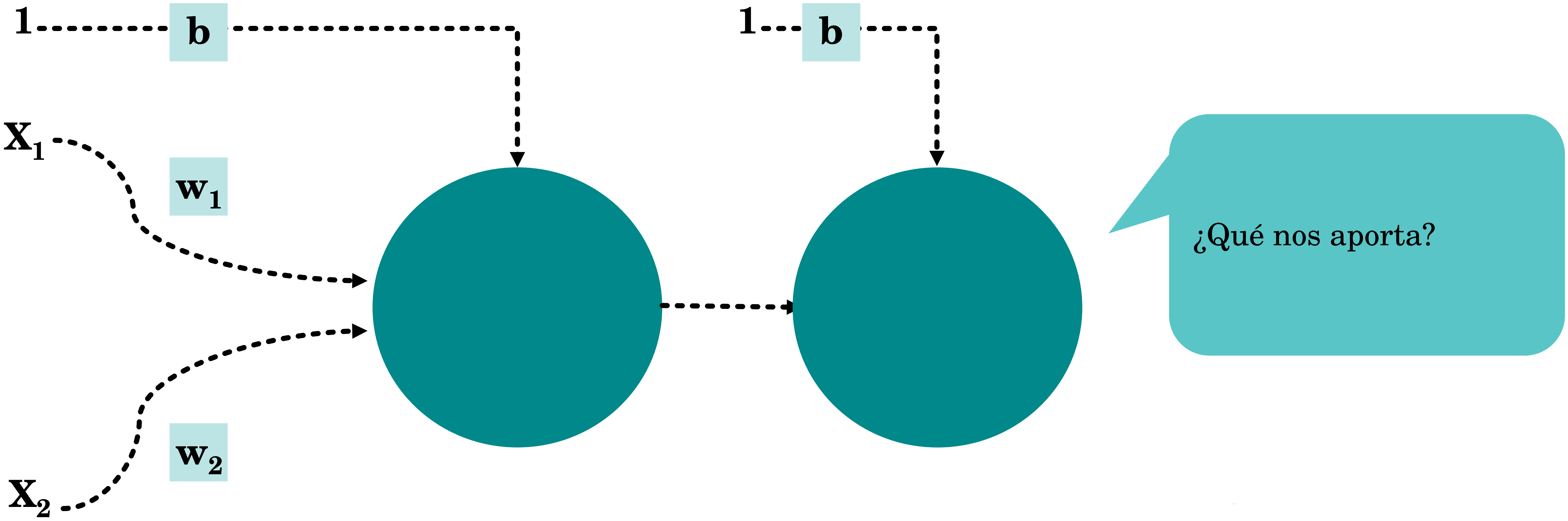
2. Secuencialidad de neuronas

- Si ponemos neuronas de forma secuencial



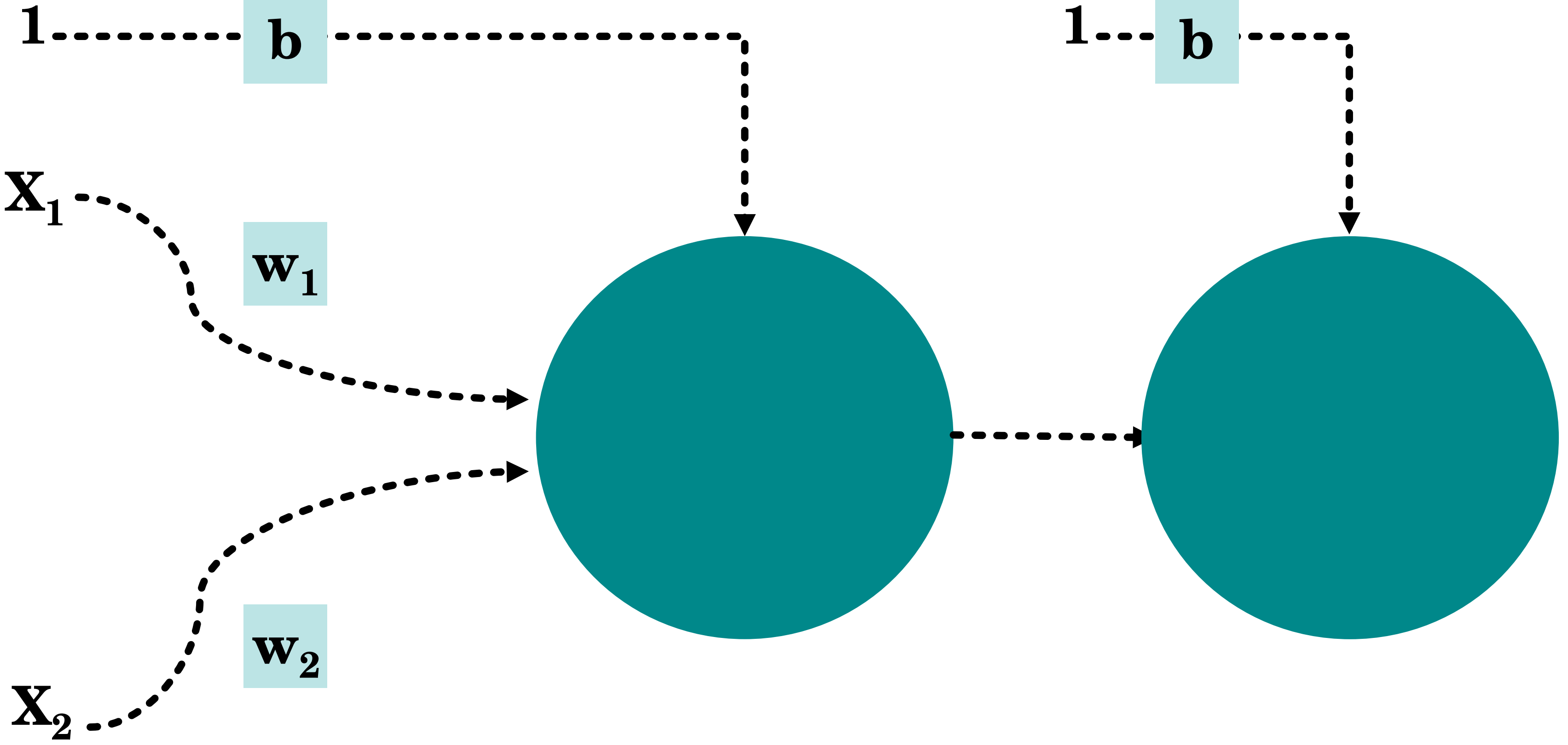
2. Secuencialidad de neuronas

- Si ponemos neuronas de forma secuencial



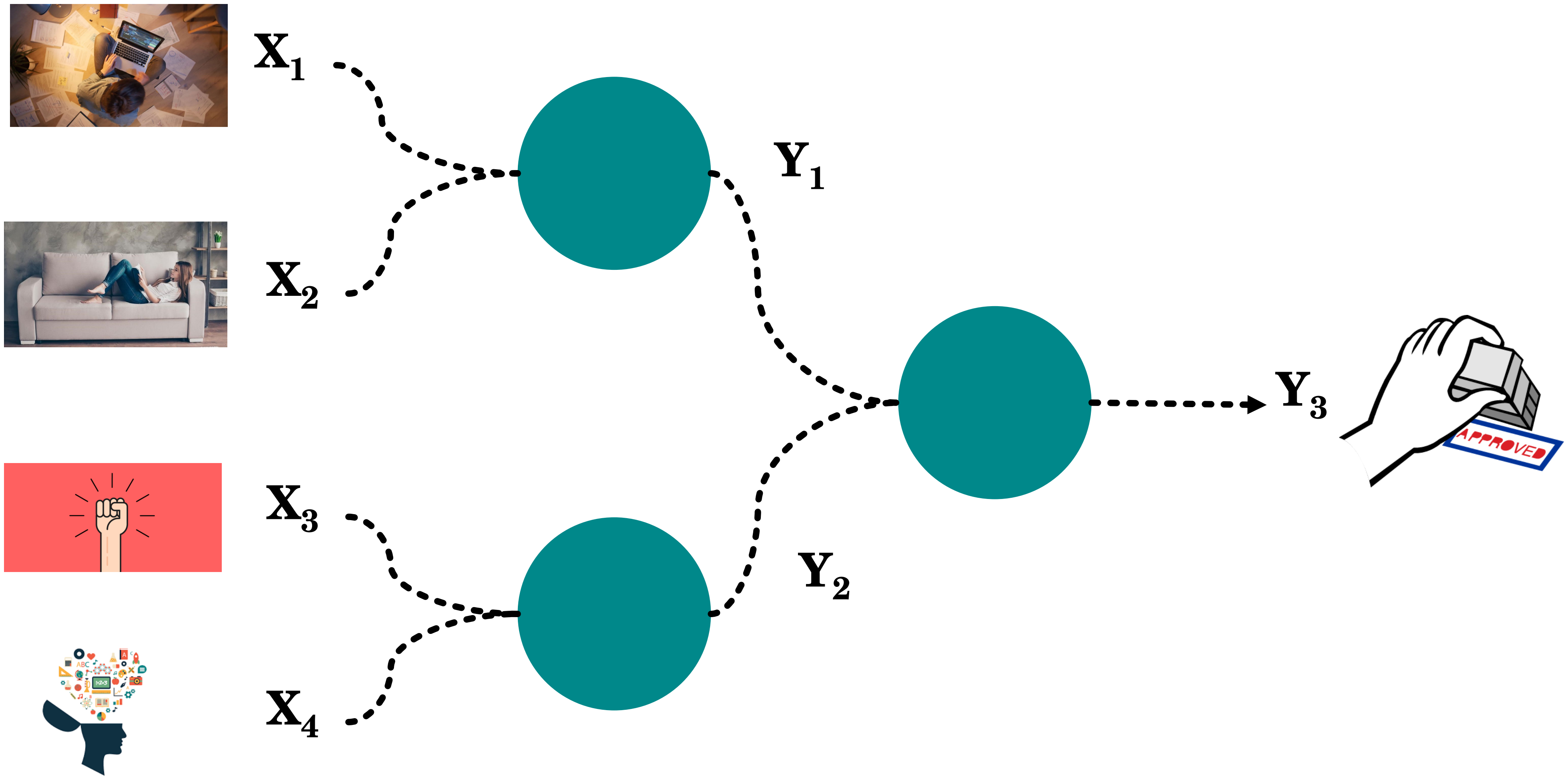
2. Secuencialidad de neuronas

- Si ponemos neuronas de forma secuencial

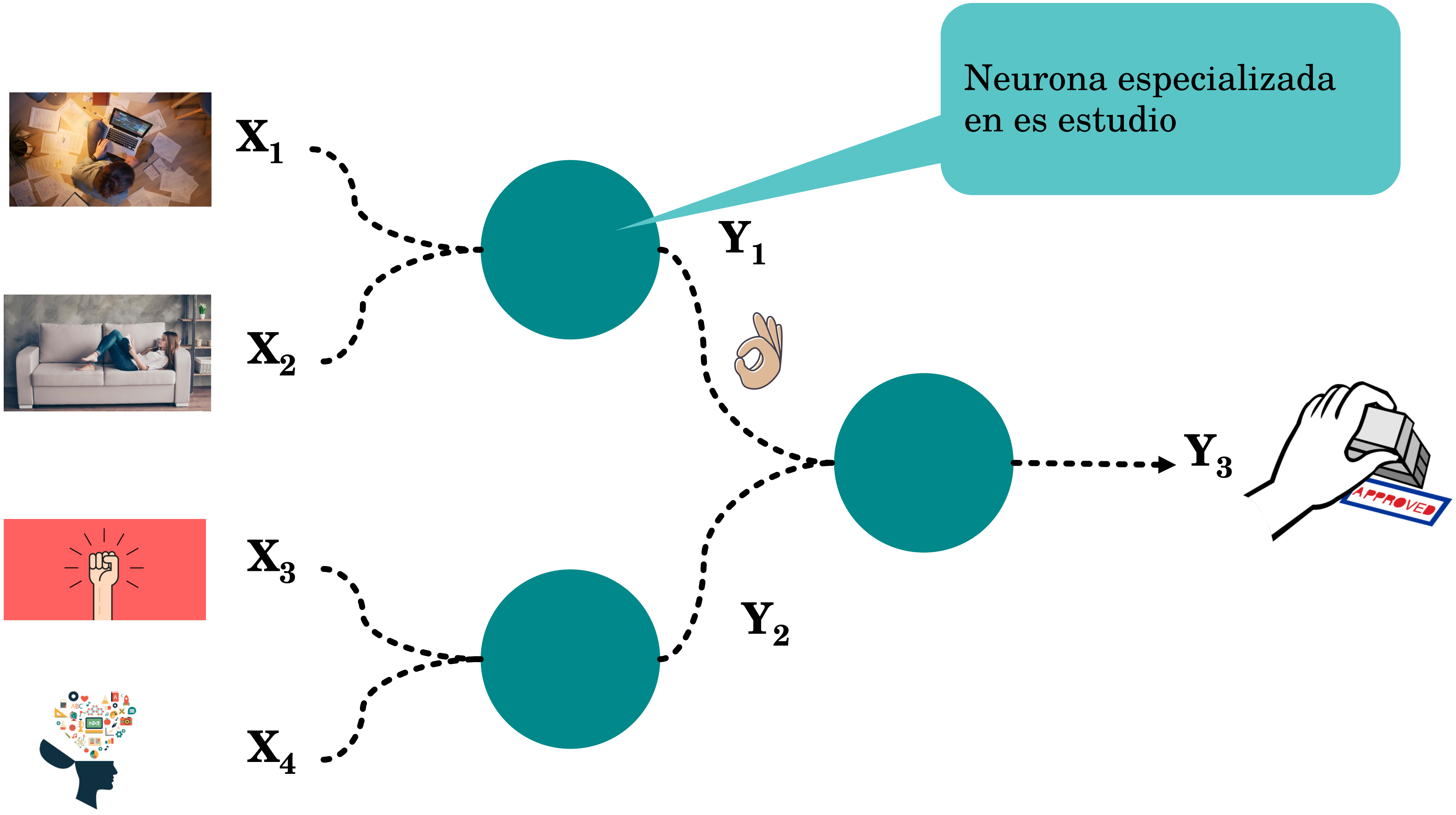


¿Qué nos aporta?
Conocimiento jerarquizado

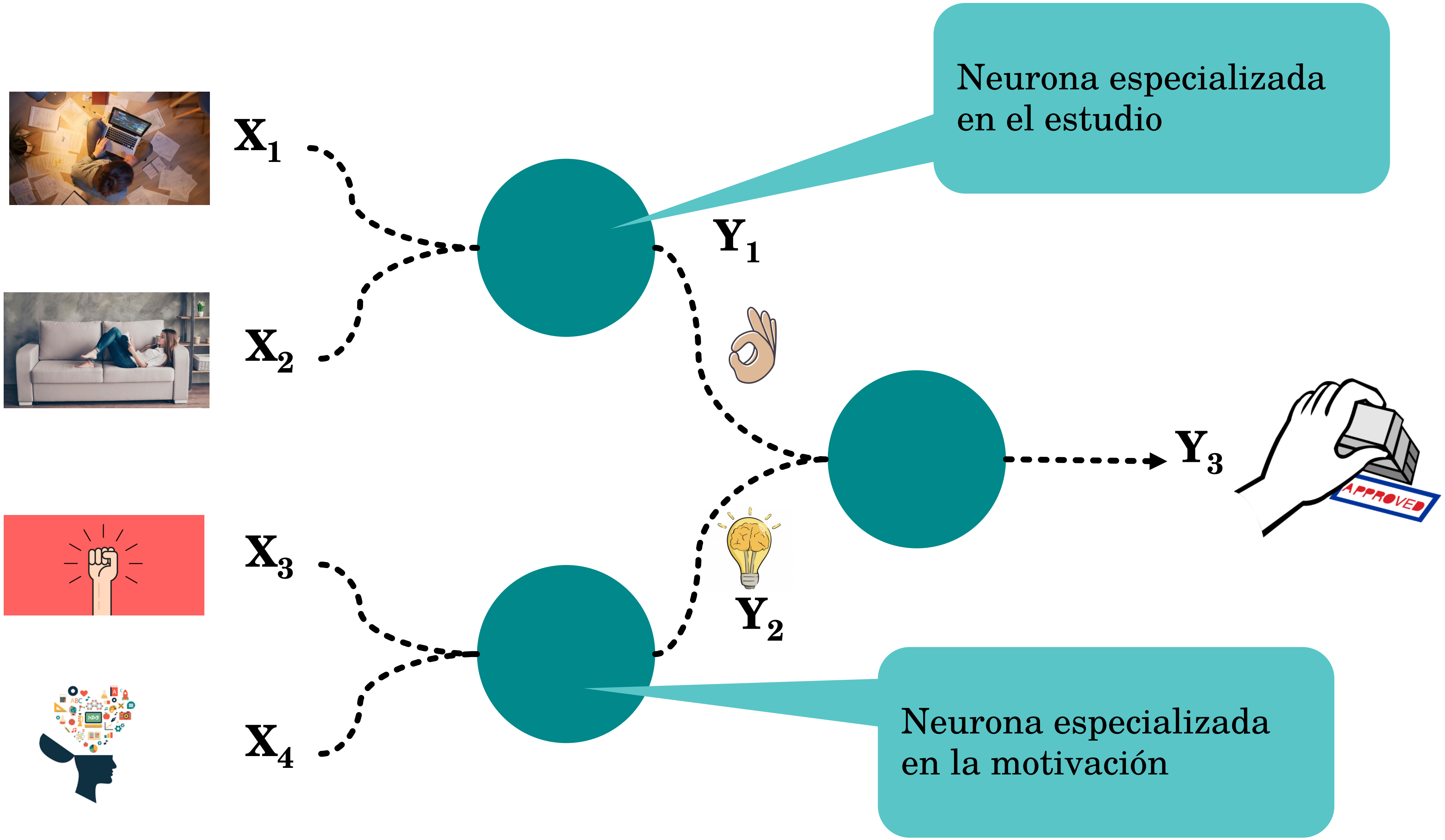
3. Conocimiento Jerarquizado



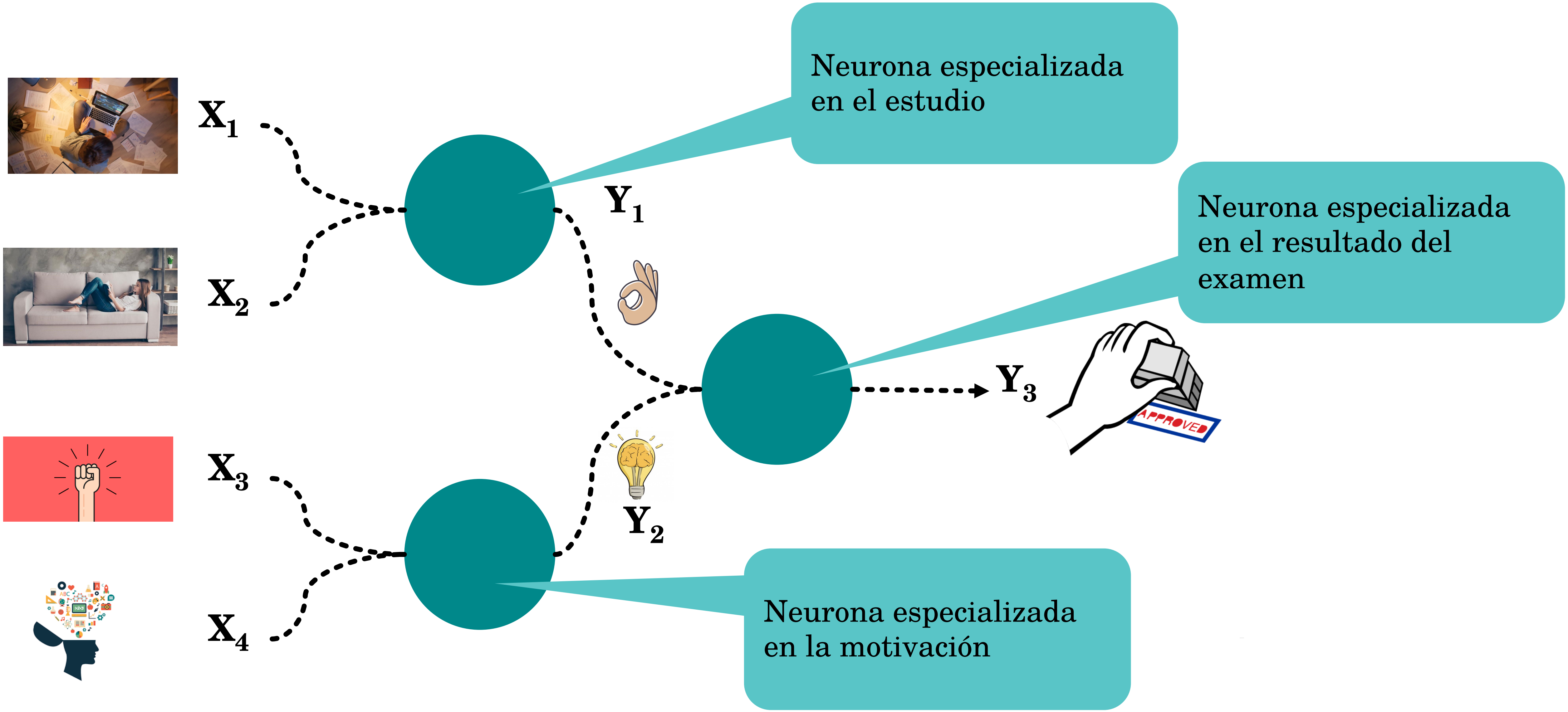
3. Conocimiento Jerarquizado



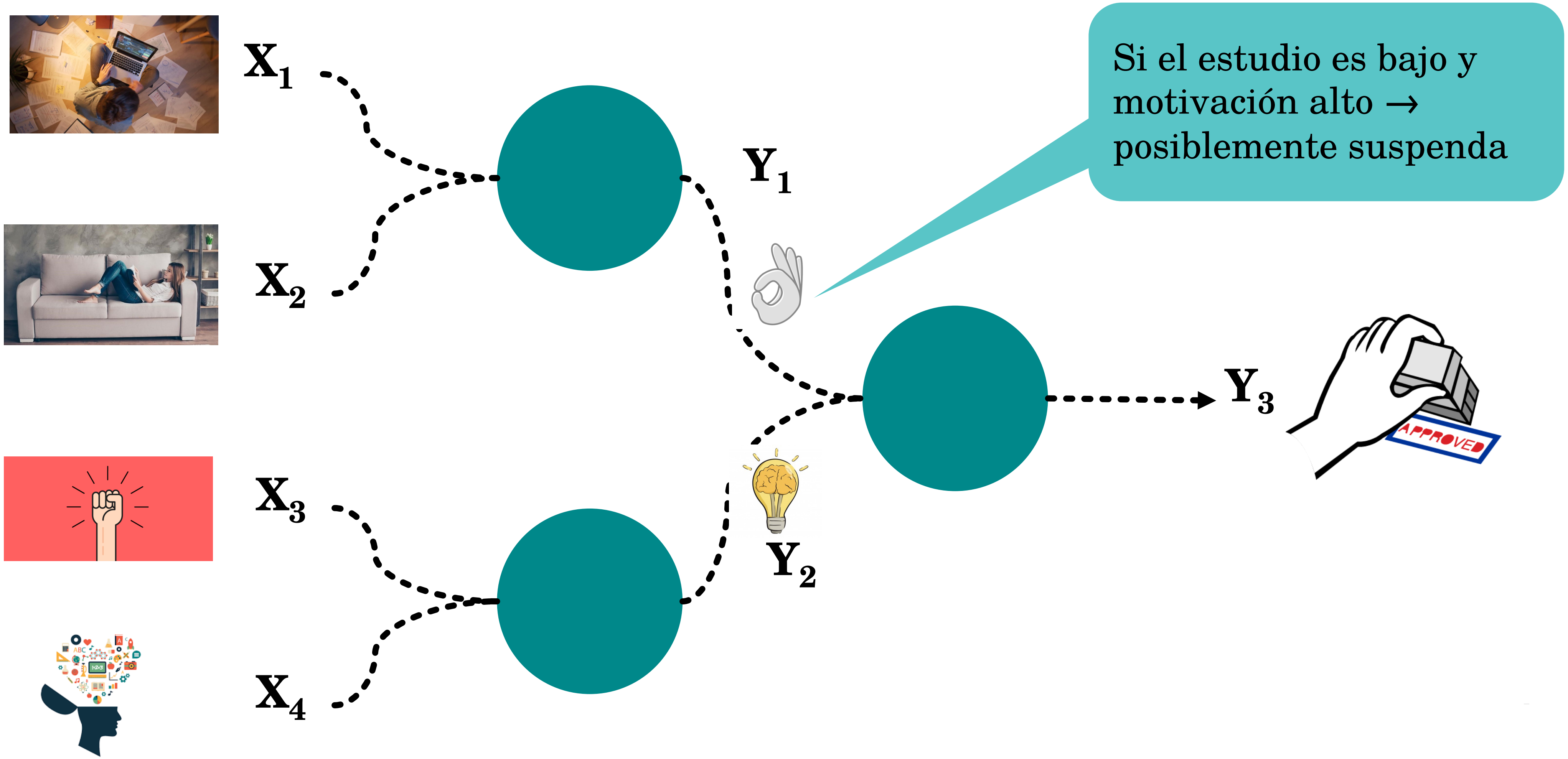
3. Conocimiento Jerarquizado



3. Conocimiento Jerarquizado

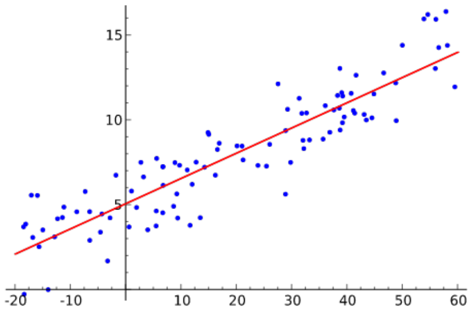


3. Conocimiento Jerarquizado
Ejemplo

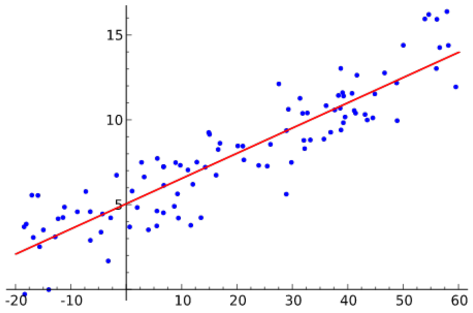


4. Problema

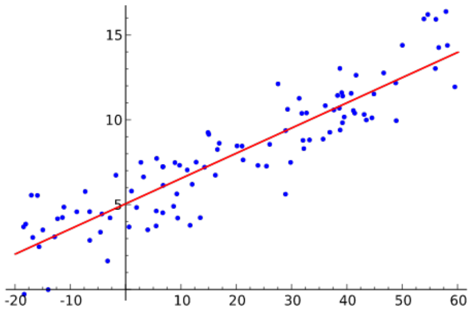
- El problema es que la concatenación de regresiones lineales es **igual** a una regresión lineal



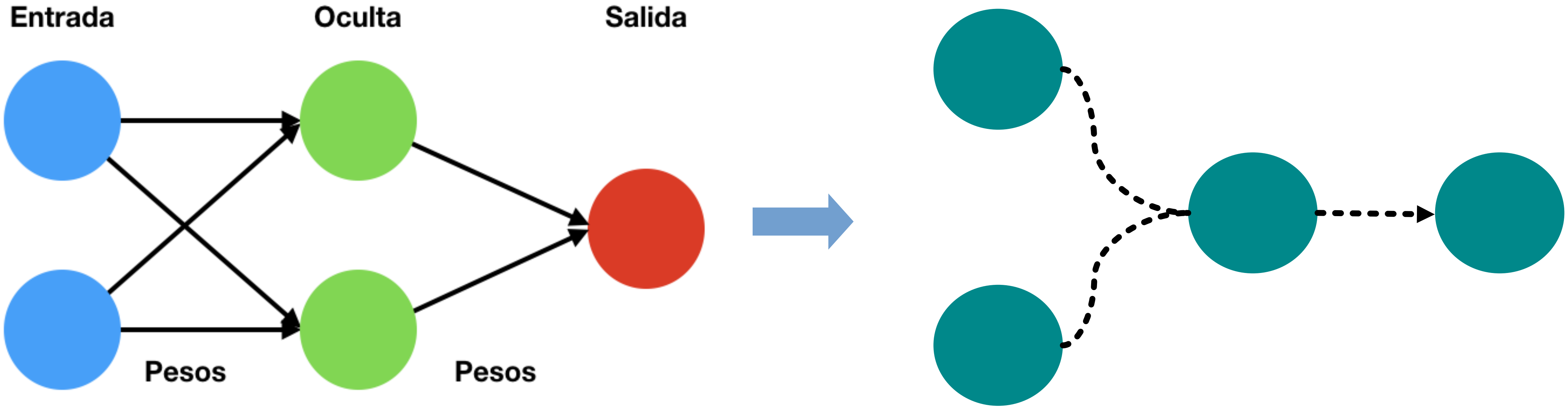
+ … +



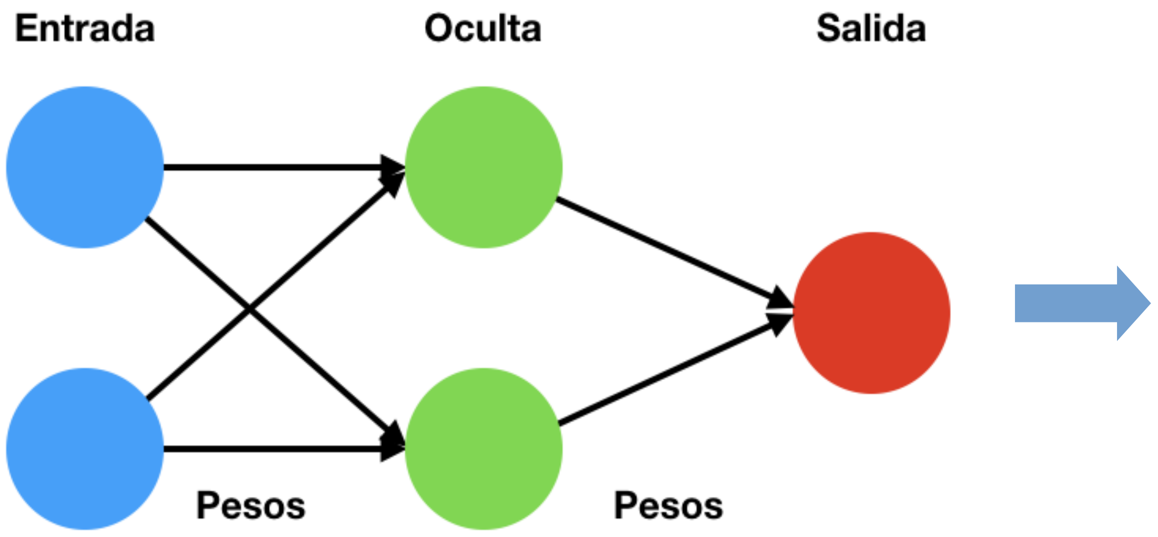
=

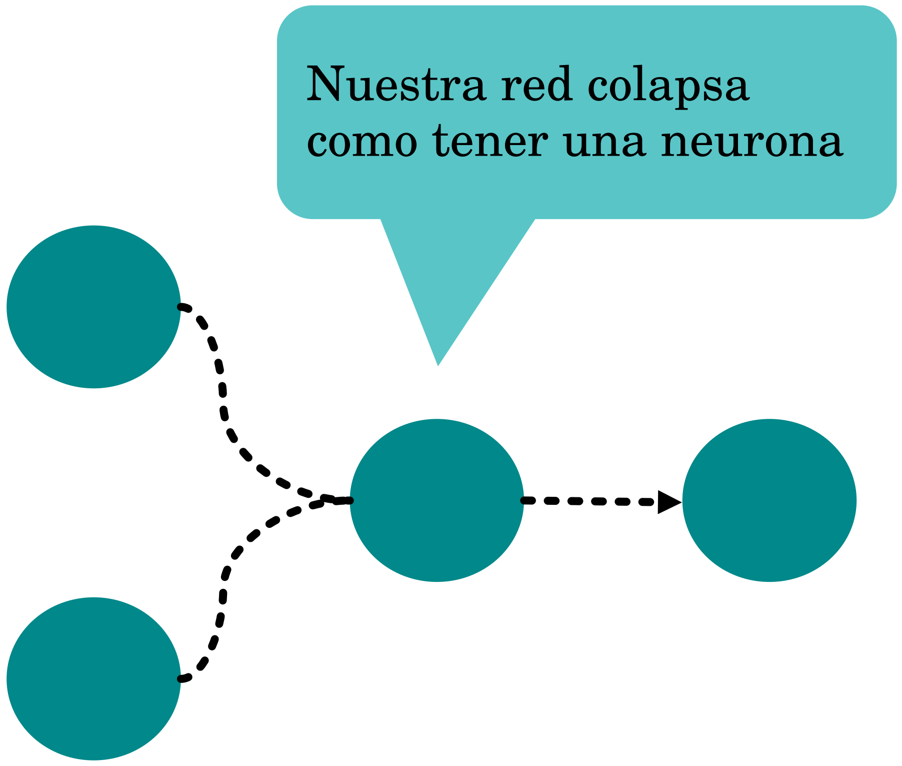


4. Problema

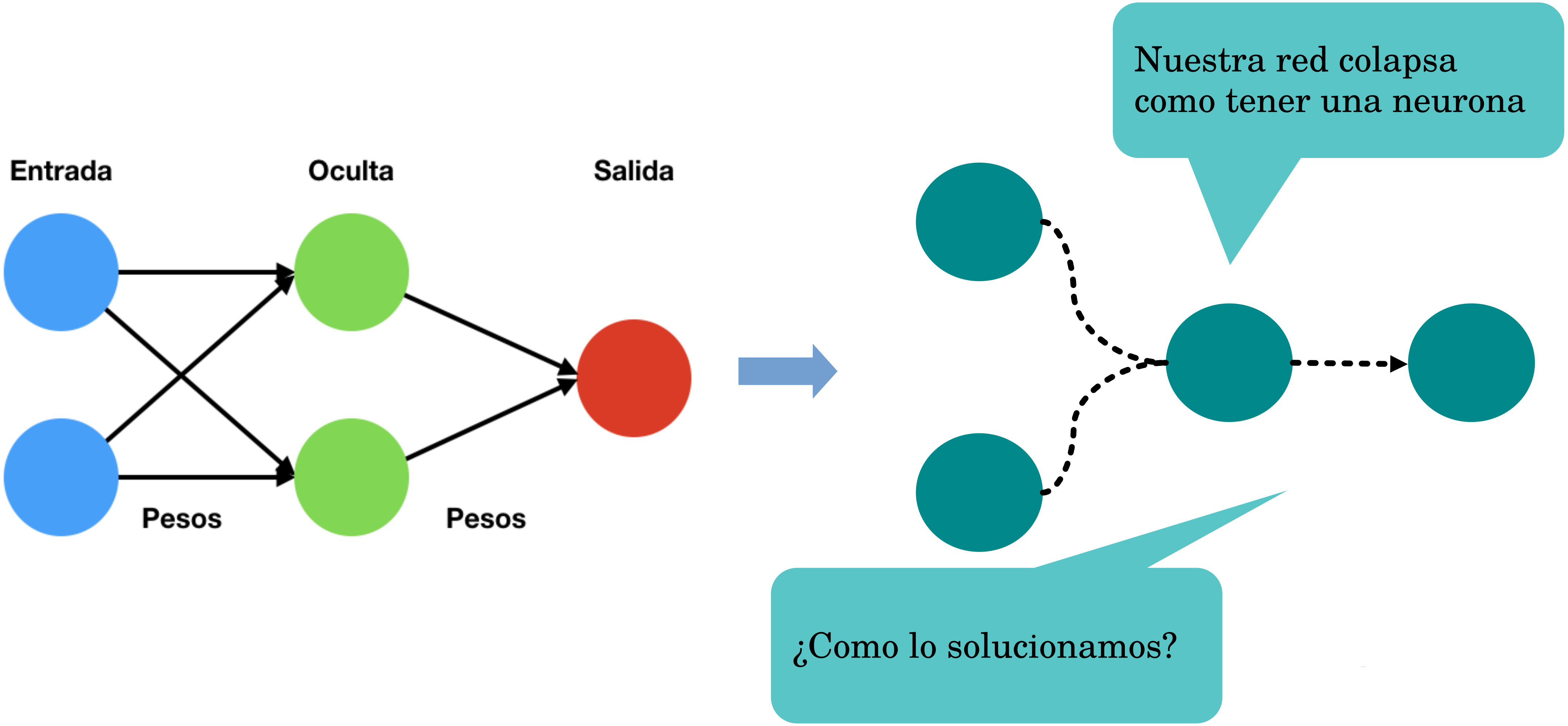


4. Problema


Entrada
Oculta
Salida
Pesos
Pesos

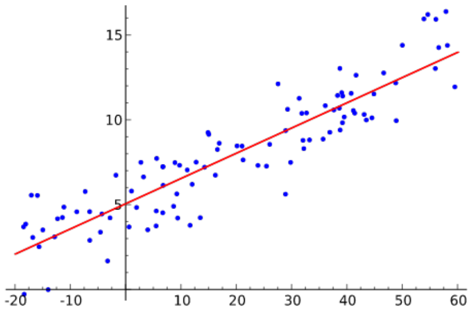

Nuestra red colapsa
como tener una neurona

4. Problema

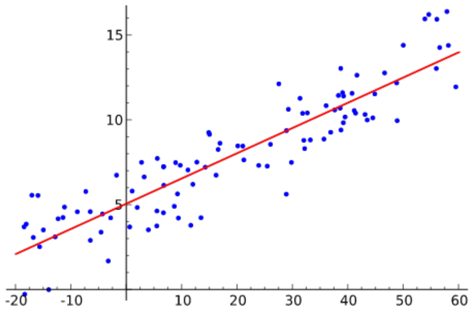

Entrada
Oculta
Salida
Pesos
Pesos
Nuestra red colapsa como tener una neurona
¿Como lo solucionamos?

4. Problema

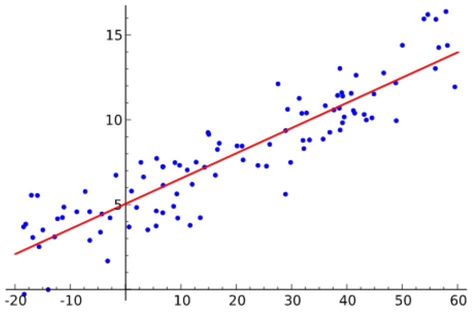
- Que nuestras funciones sufran una distorsión para romper la linealidad



+ … +



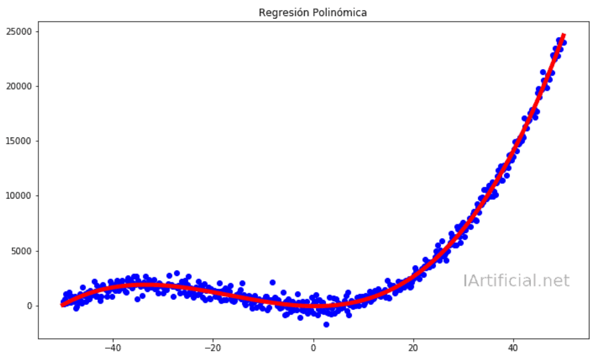
=



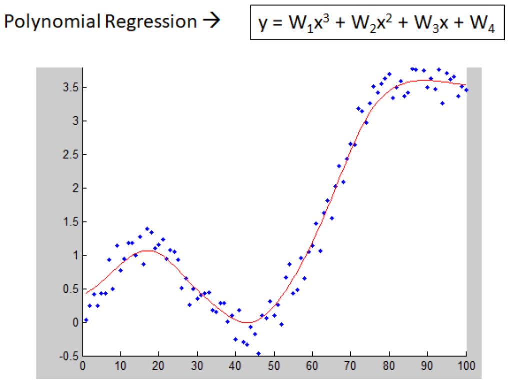
¿Entoces distosionamos las funciones?

4. Problema

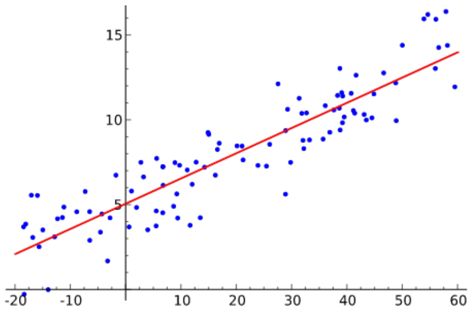
- Que nuestras funciones sufran una distorsión para romper la linealidad



+ … +



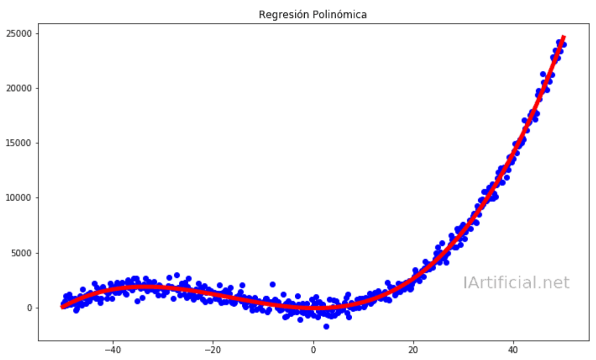
=/=



Así el resultado **no es Lineal**

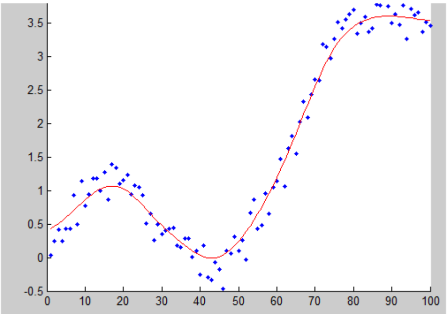
4. Problema

- Que nuestras funciones sufran una distorsión para romper la linealidad

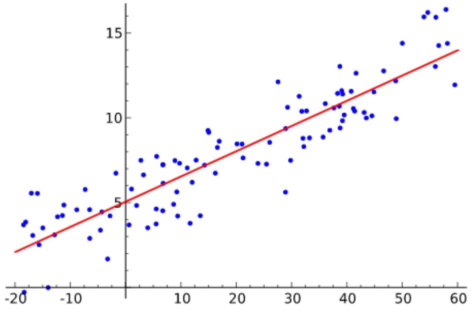


+ … +

Polynomial Regression → $y = W_1x^3 + W_2x^2 + W_3x + W_4$



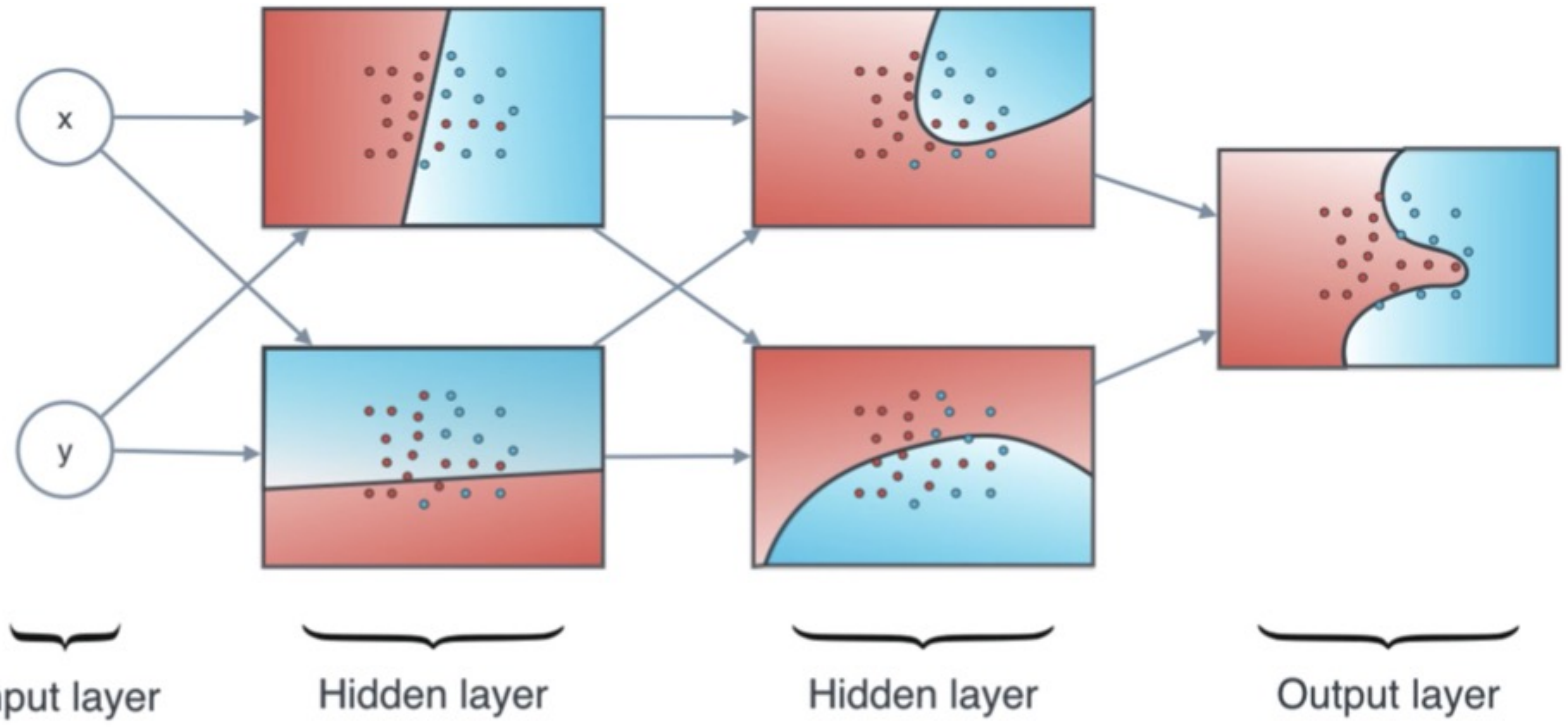
=/=



Añadimos **funciones de activación**

5. Arquitectura de redes neuronales

- Son la base de las **redes profundas**, actualmente muy utilizadas con mucho éxito.
- Cada **capa extrae características** cada vez más complejas

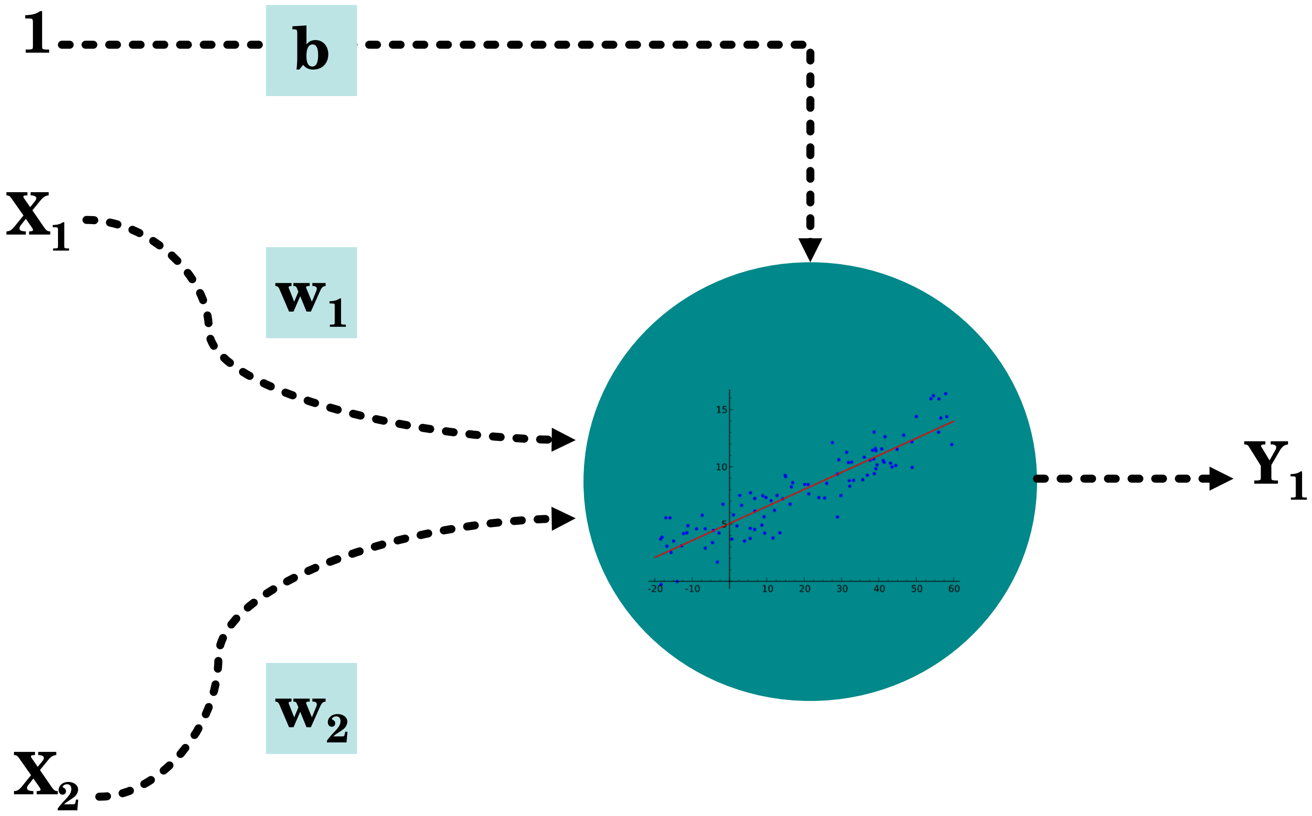


Función de activación

ETS de Ingeniería Informática

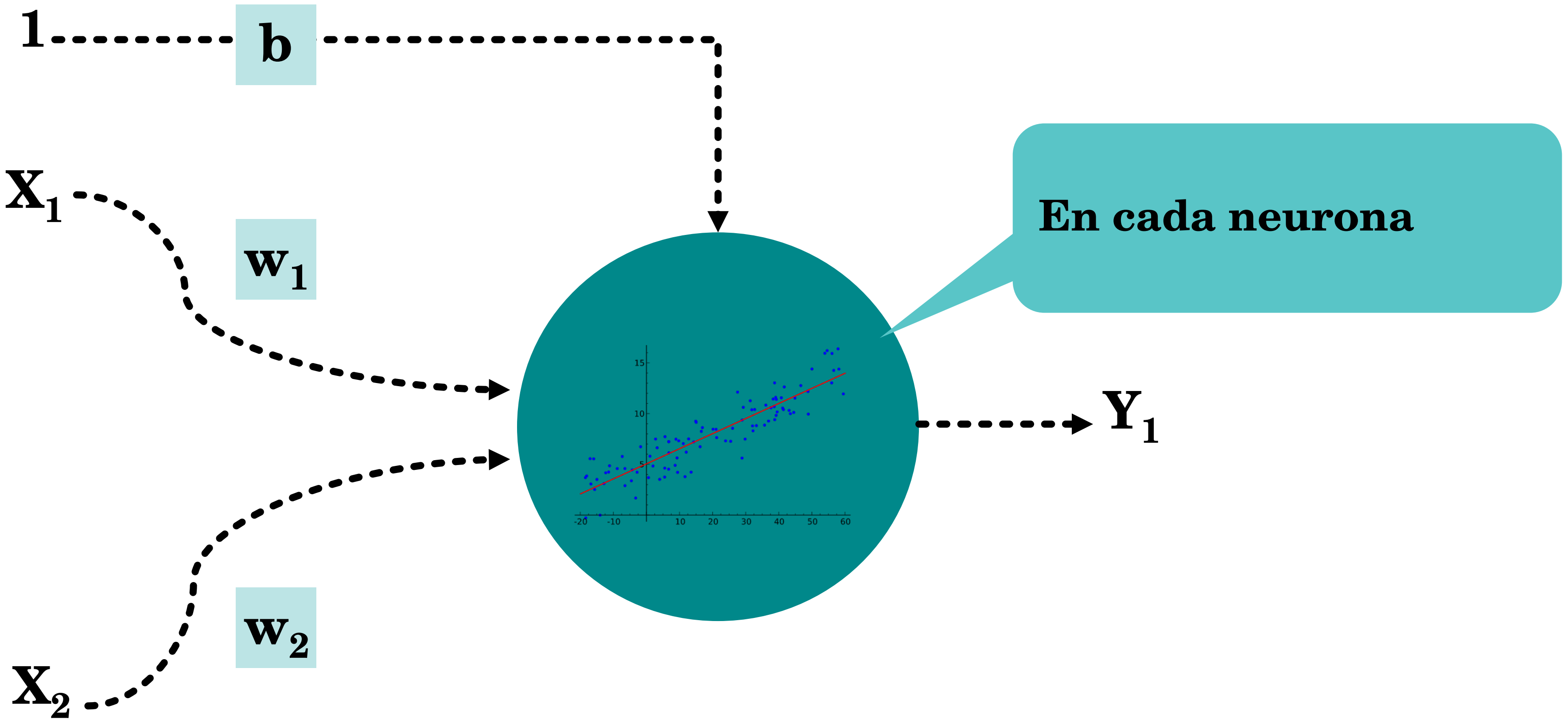
UNED

1. ¿Donde se encuentra?



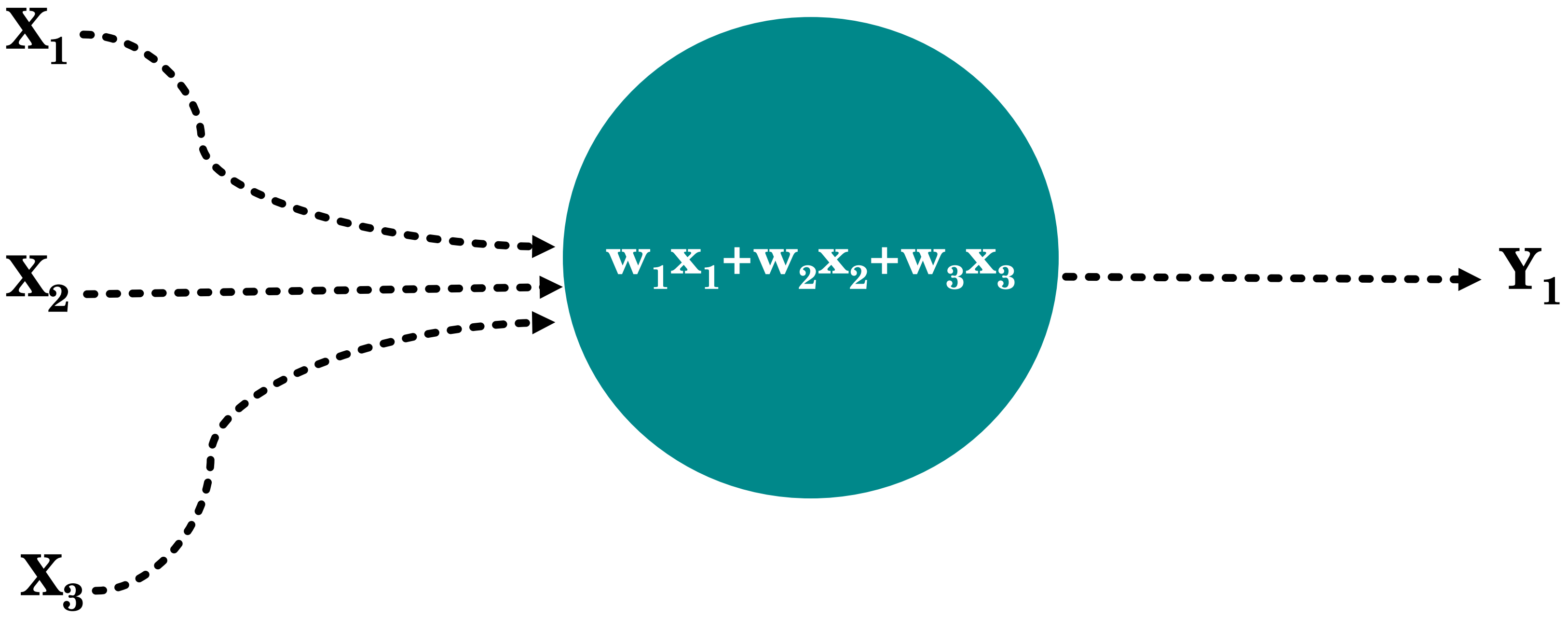
$$y = w_1x_1 + w_2x_2 + b$$

1. ¿Donde se encuentra?

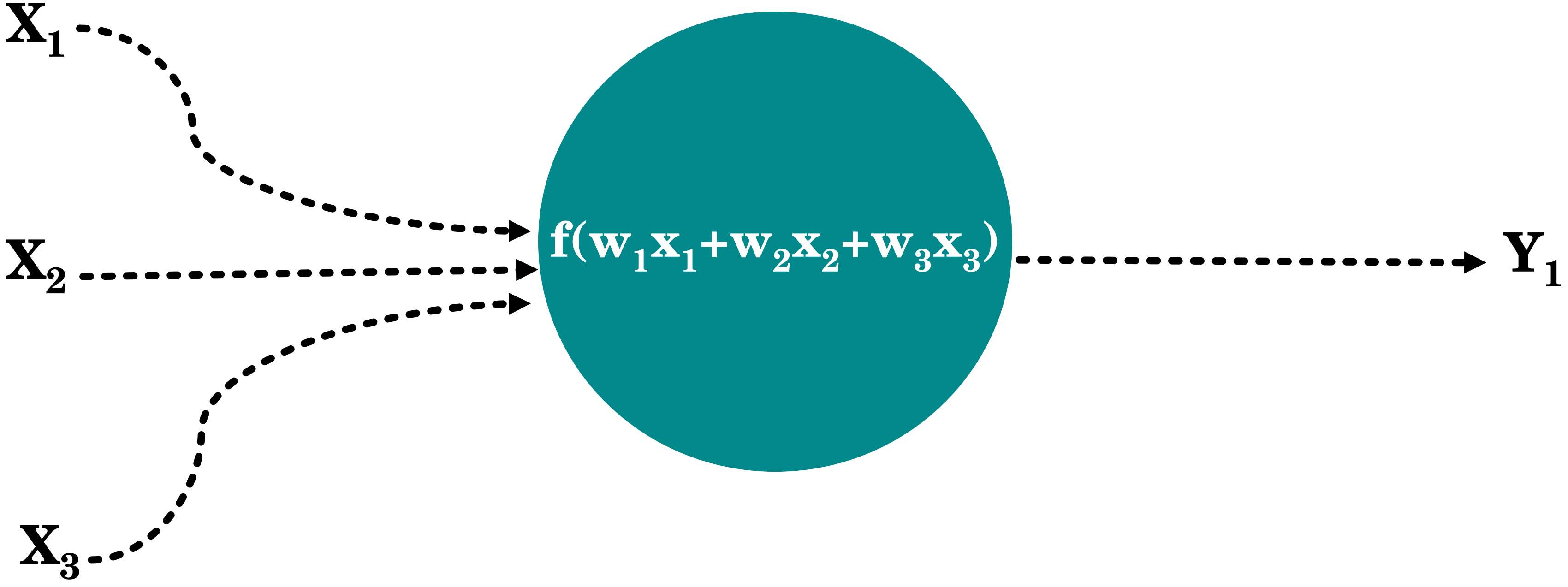


$$y = w_1x_1 + w_2x_2 + b$$

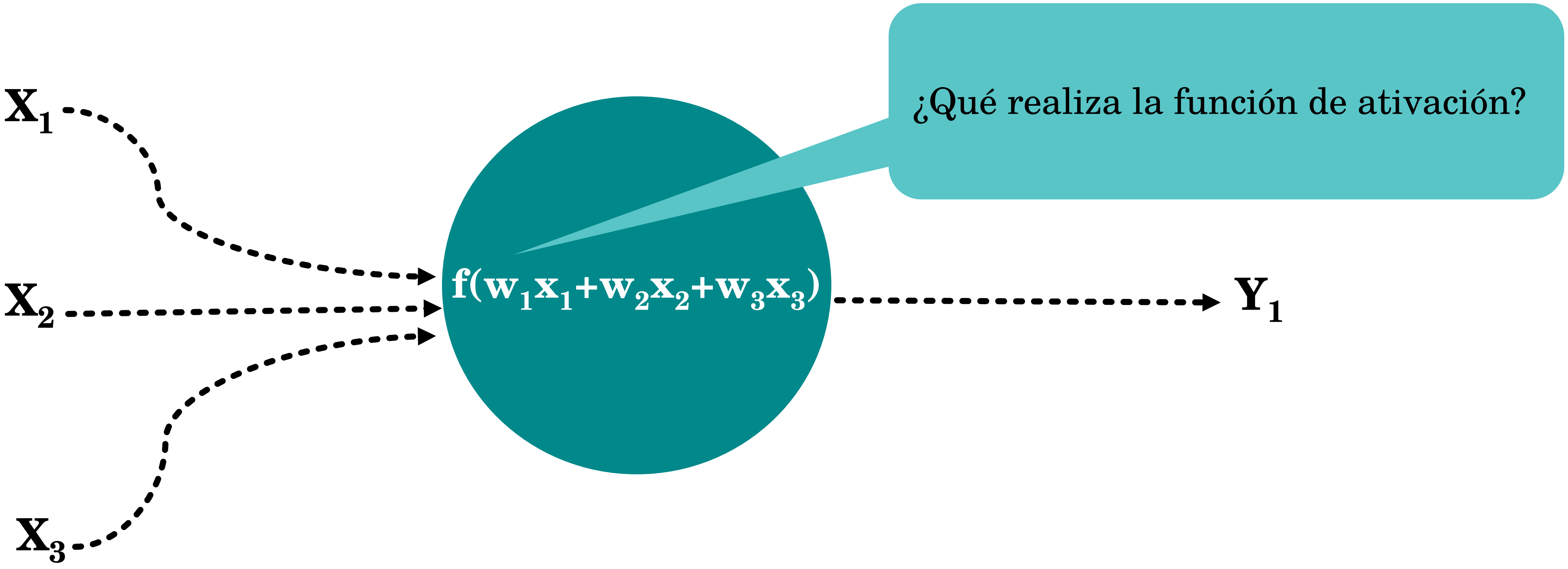
2. ¿Como la representamos?



2. ¿Como la representamos?



2. ¿Como la representamos?



2. ¿Como la representamos?

X_1

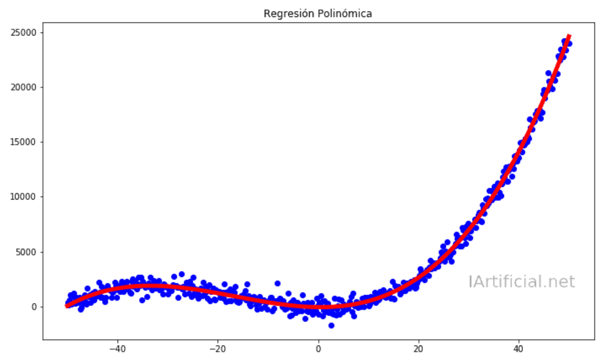
X_2

X_3

$$f(w_1x_1+w_2x_2+w_3x_3)$$

Y_1

¿Qué realiza la función de ativación?



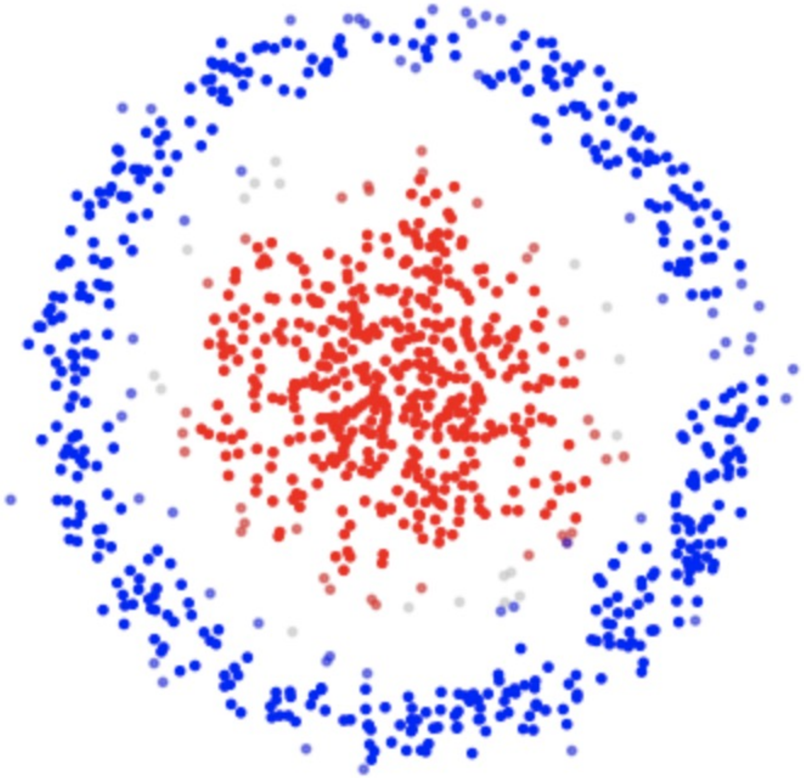
Distorsiona la salida dotándola de
No linealidad

Cómo opera el MLP

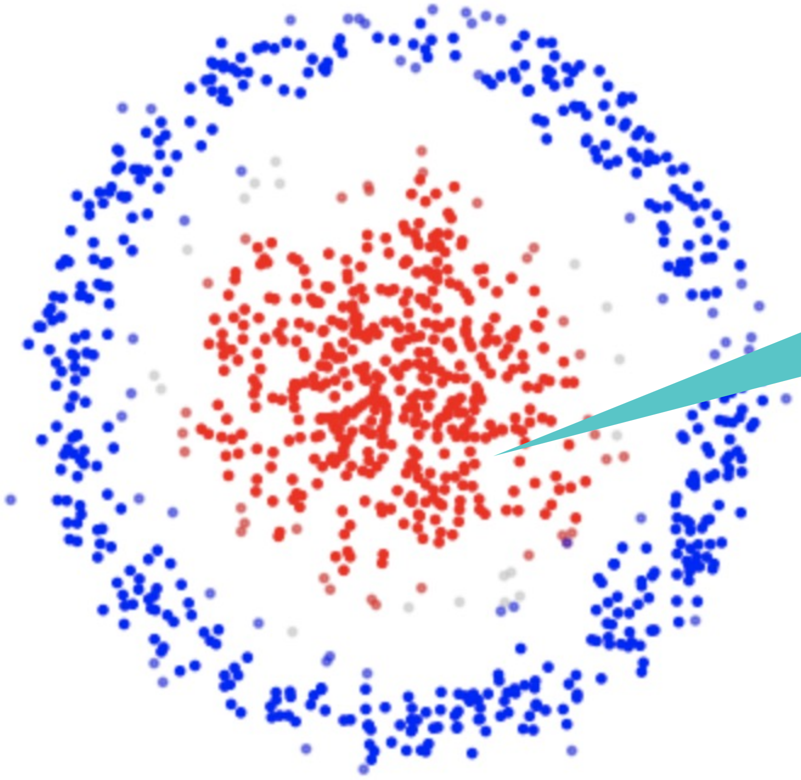
ETS de Ingeniería Informática

UNED

1. Problema

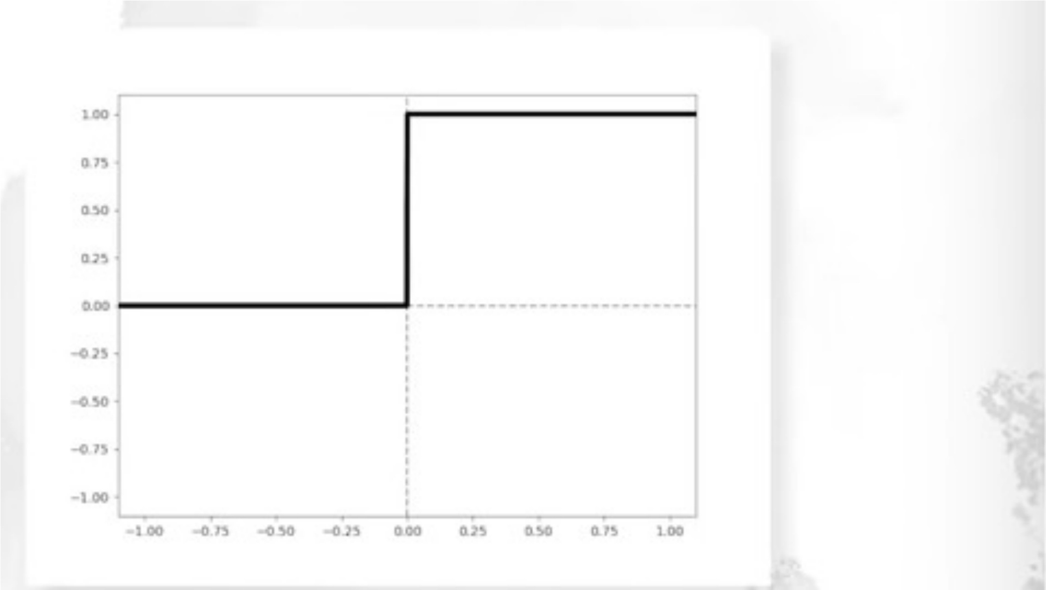


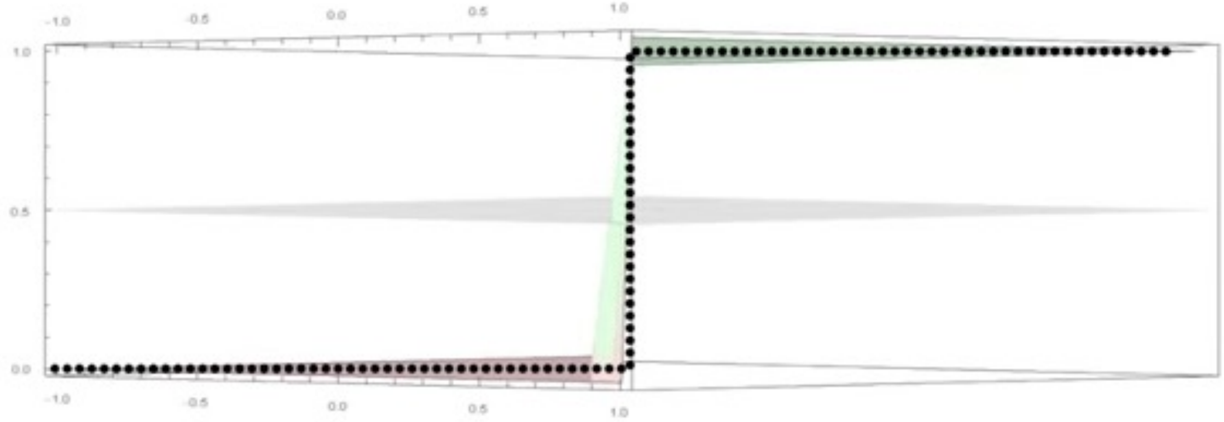
1. Problema



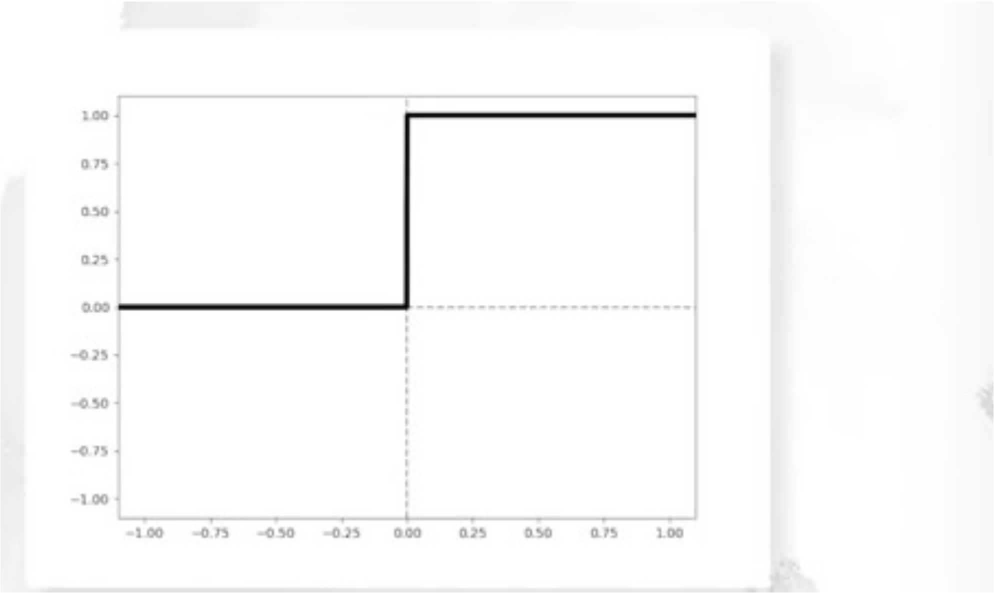
¿Cómo ver geométricamente el efecto de una función de activación?

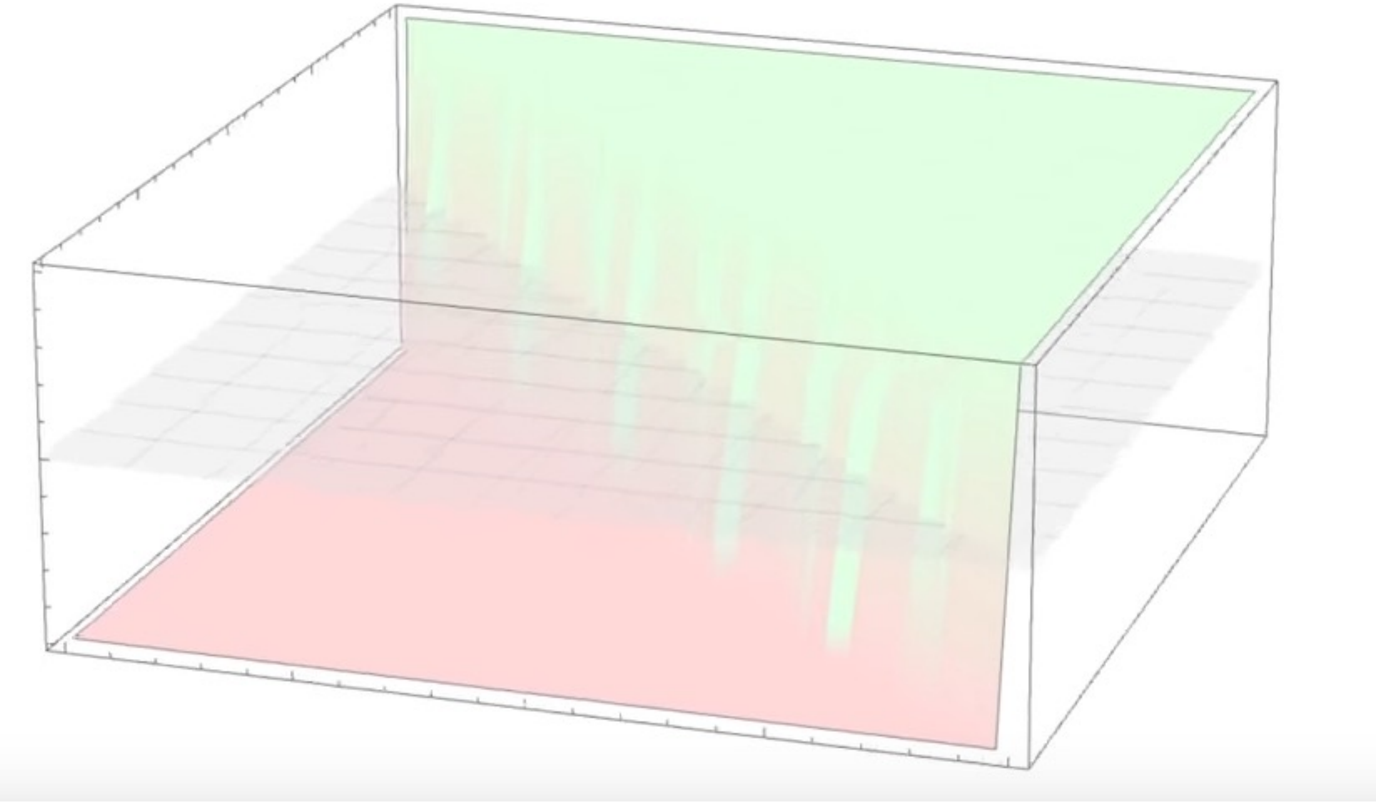
2. Función de activación binaria



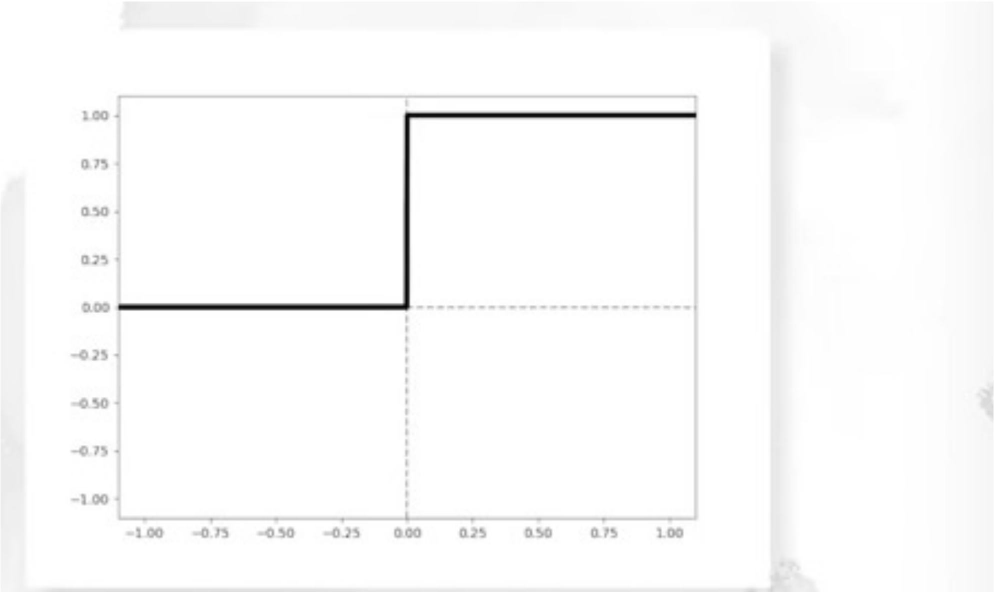


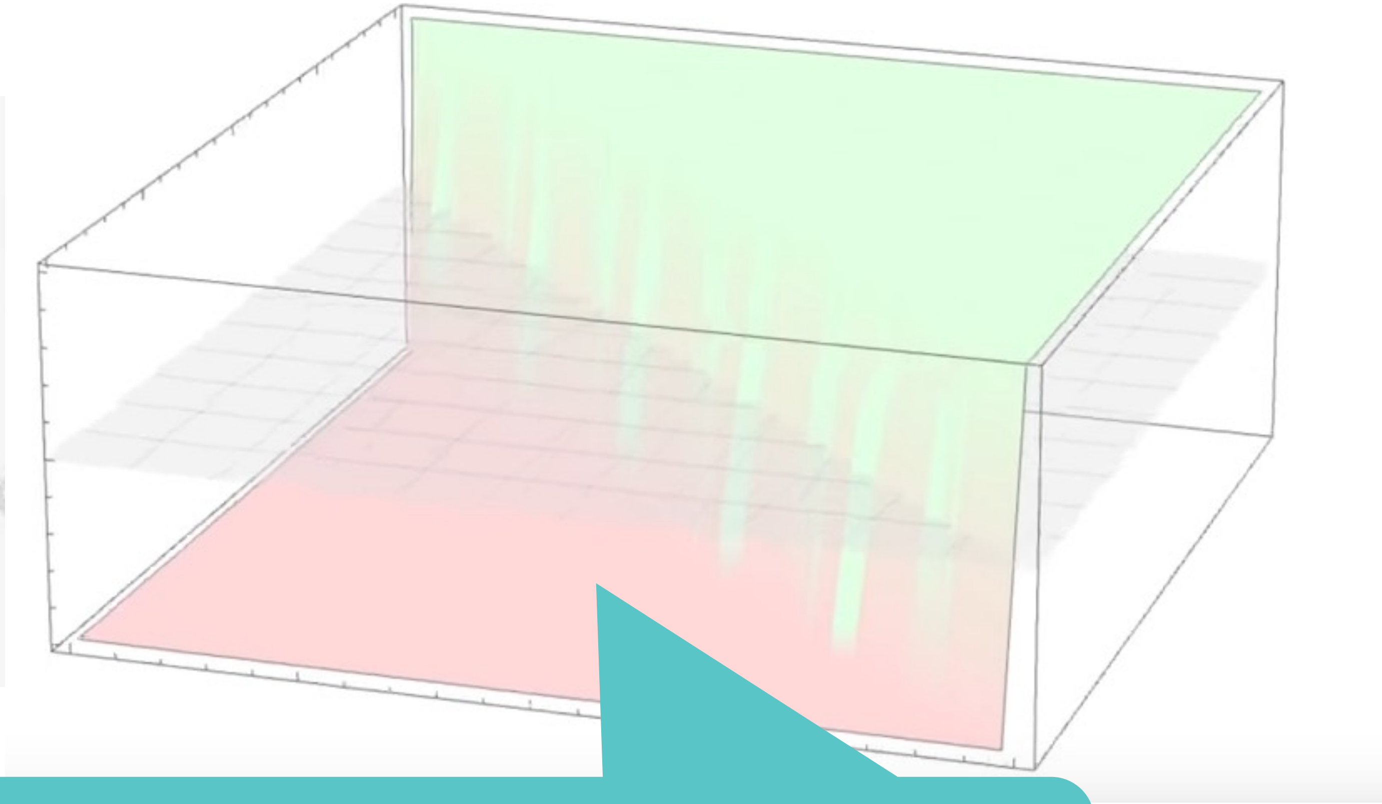
2. Función de activación binaria





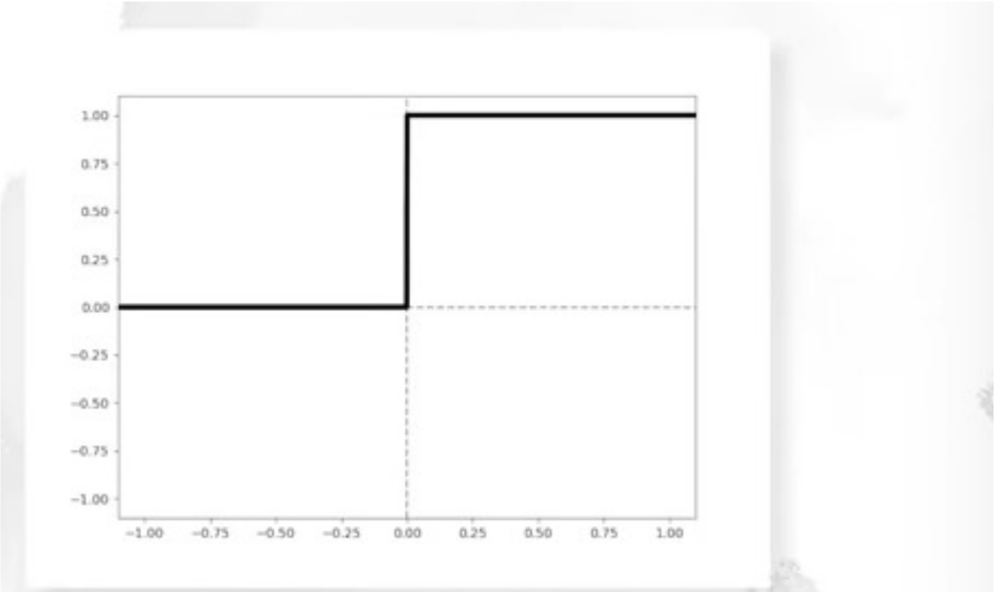
2. Función de activación binaria

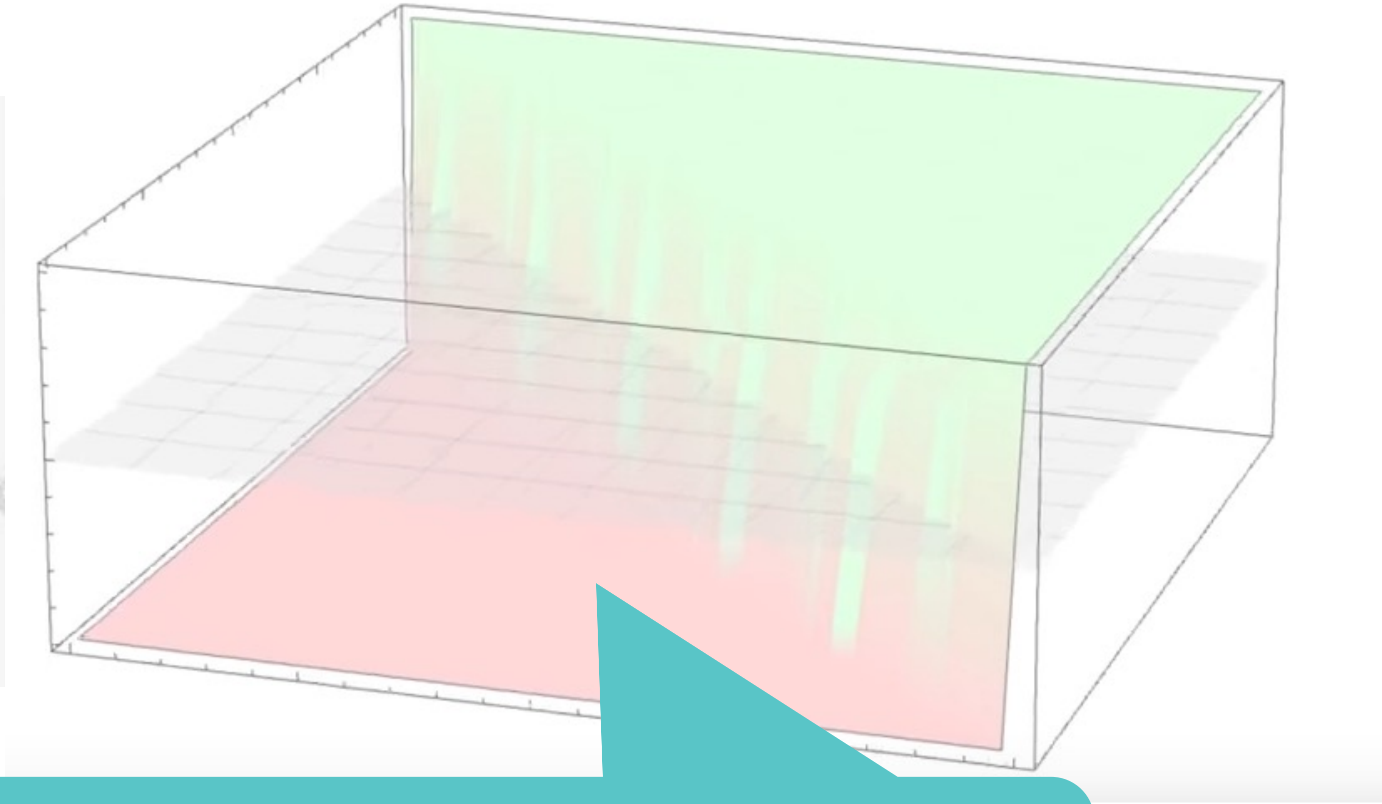




Todo lo que está encima del hiperplano
será una clase y lo que está debajo otra

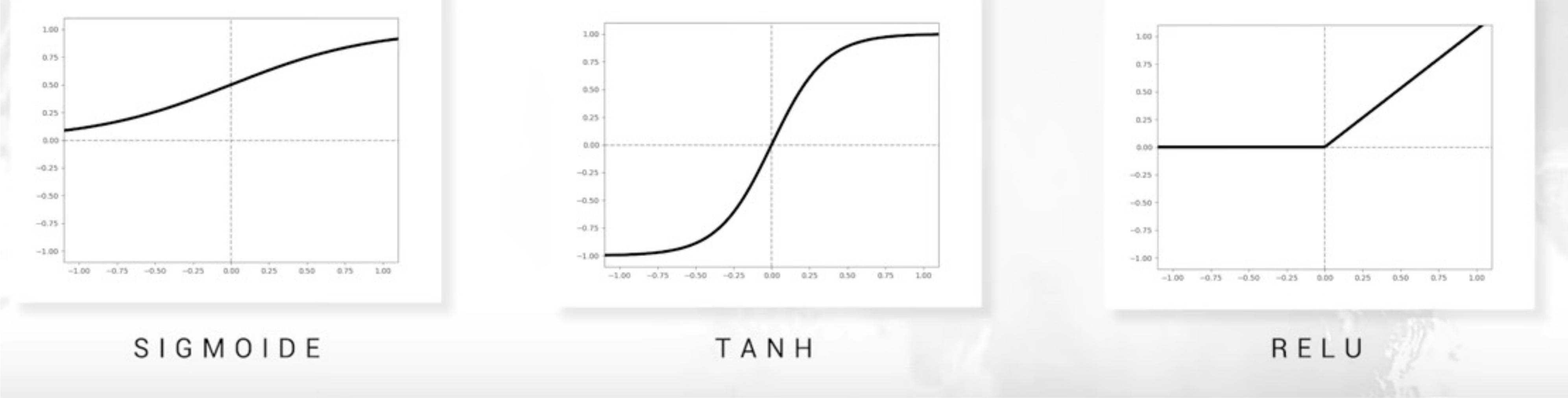
2. Función de activación binaria

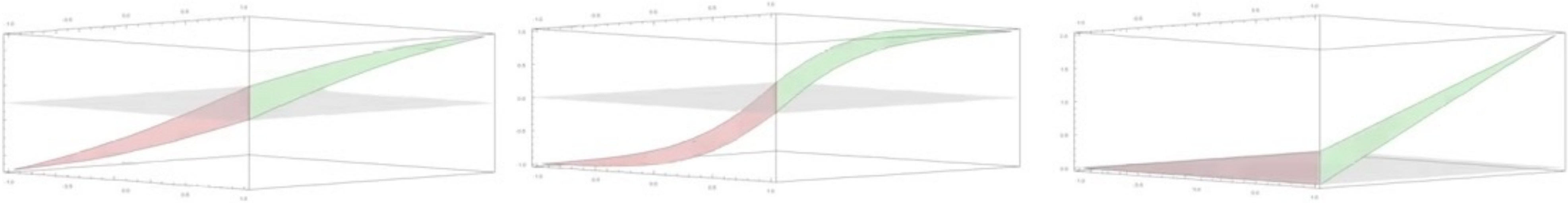




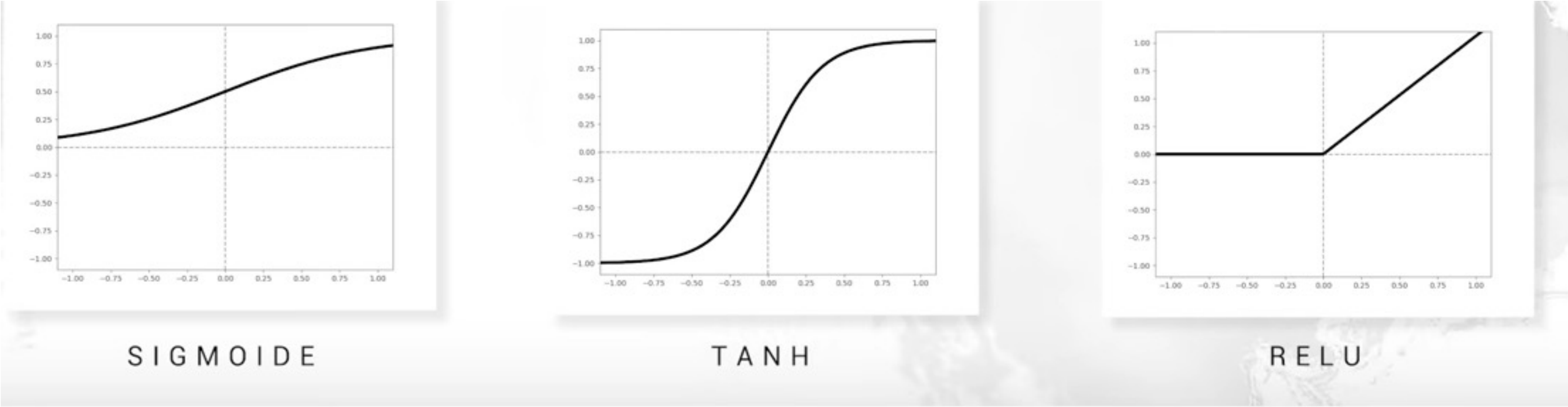
Necesitamos distorsionar la señal para acomodar el problema inicial

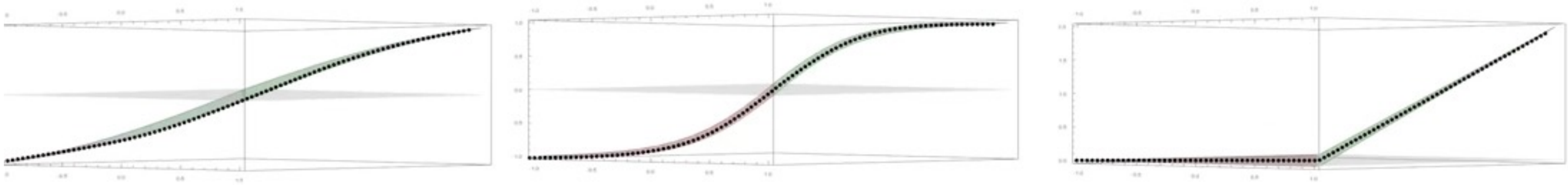
3. Otras Funciones de activación



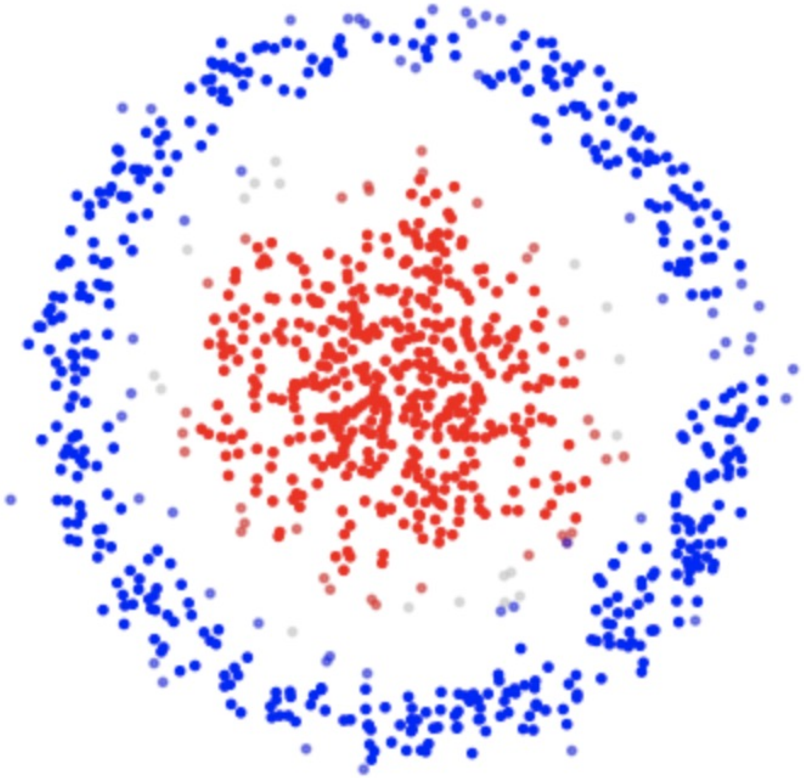


3. Otras Funciones de activación

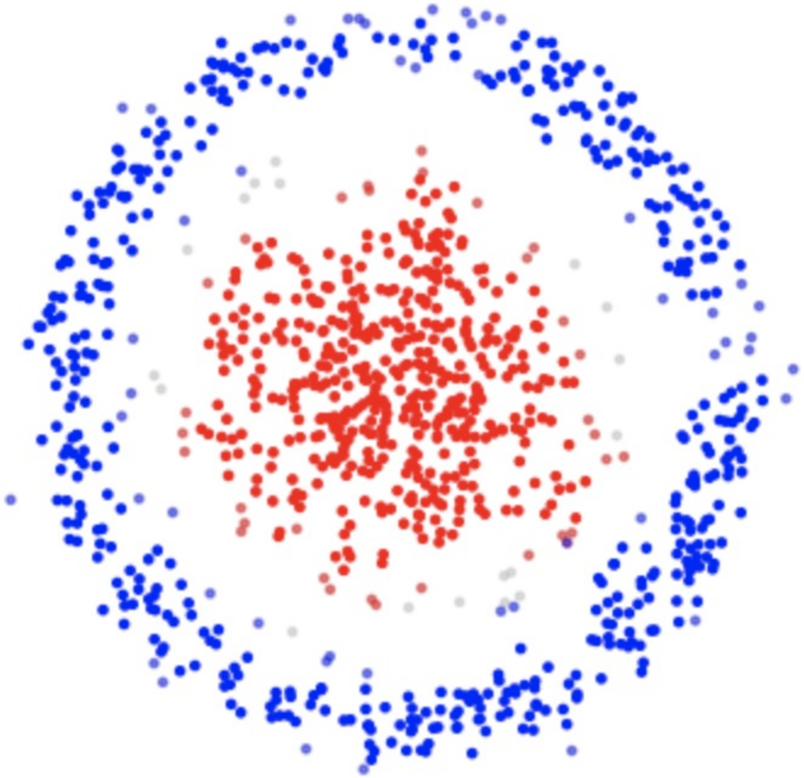




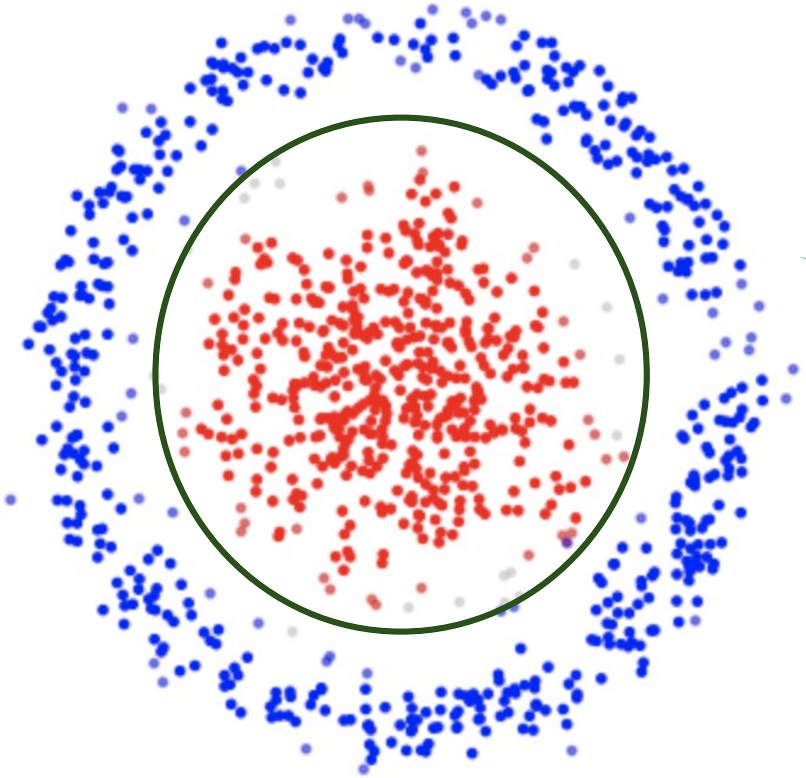
4. Encontrar el hiperplano



4. Encontrar el hiperplano

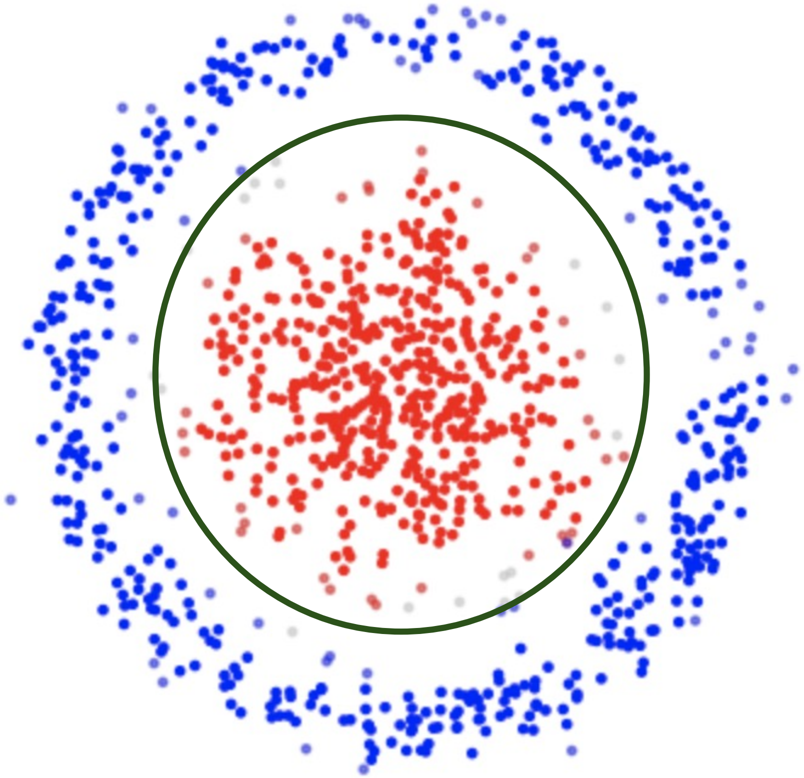


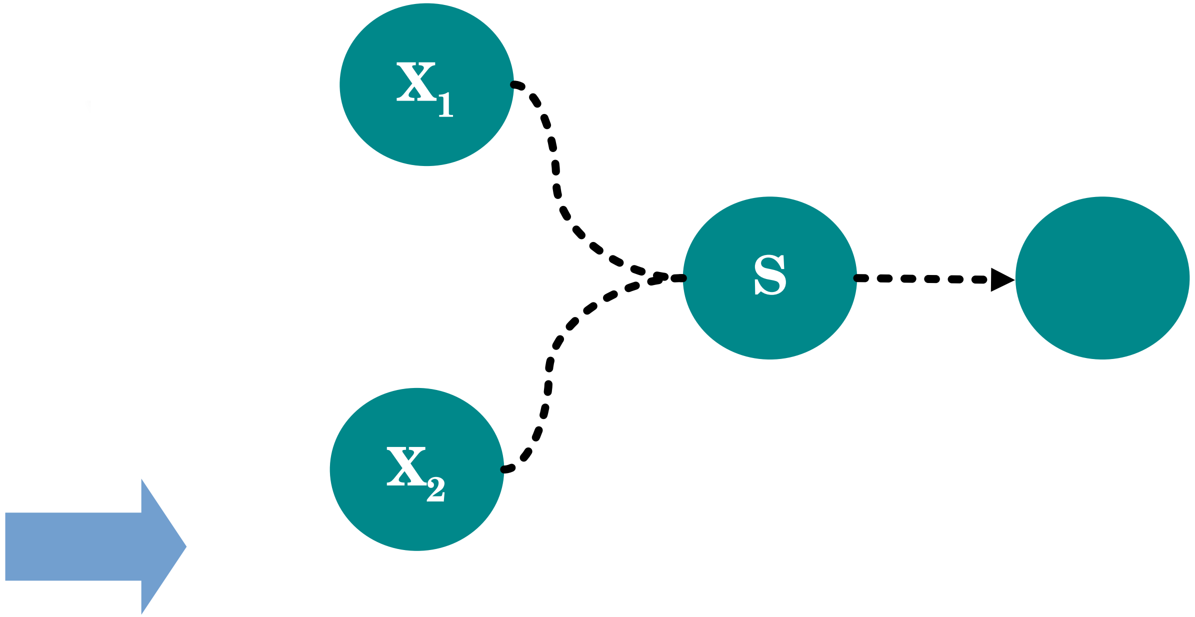
4. Encontrar el hiperplano



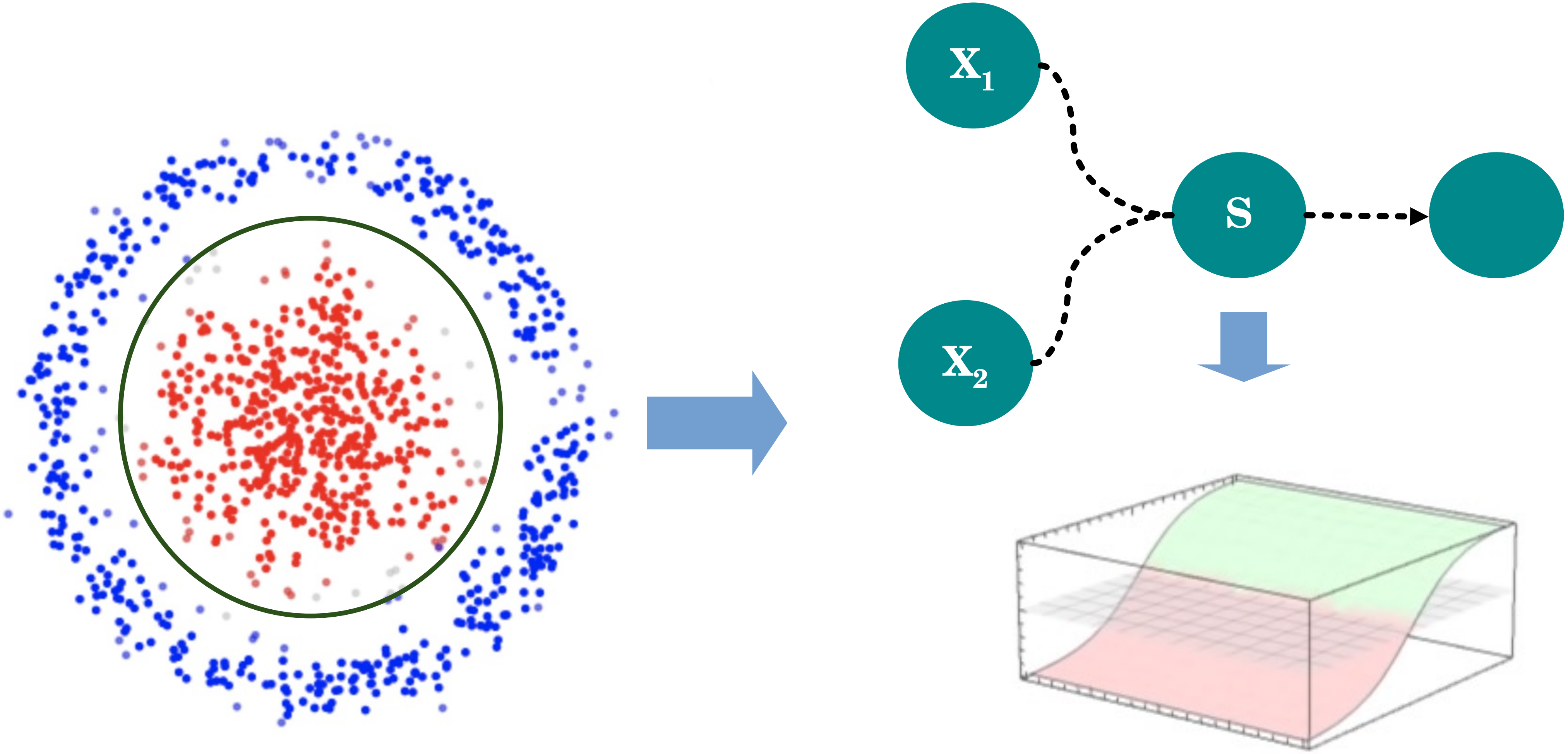
¿Cómo trazar esa frontera?

4. Encontrar el hiperplano

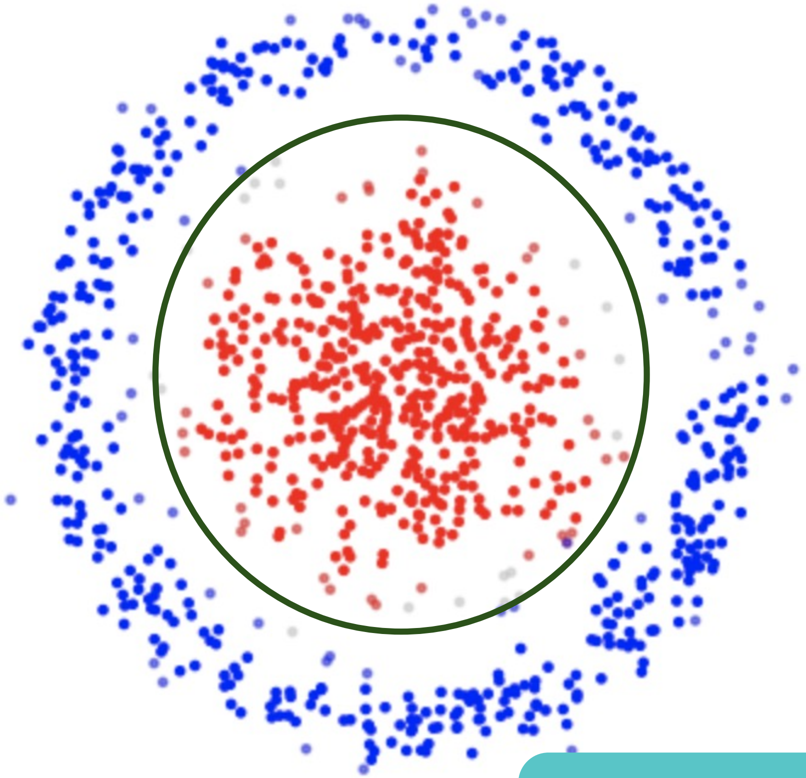


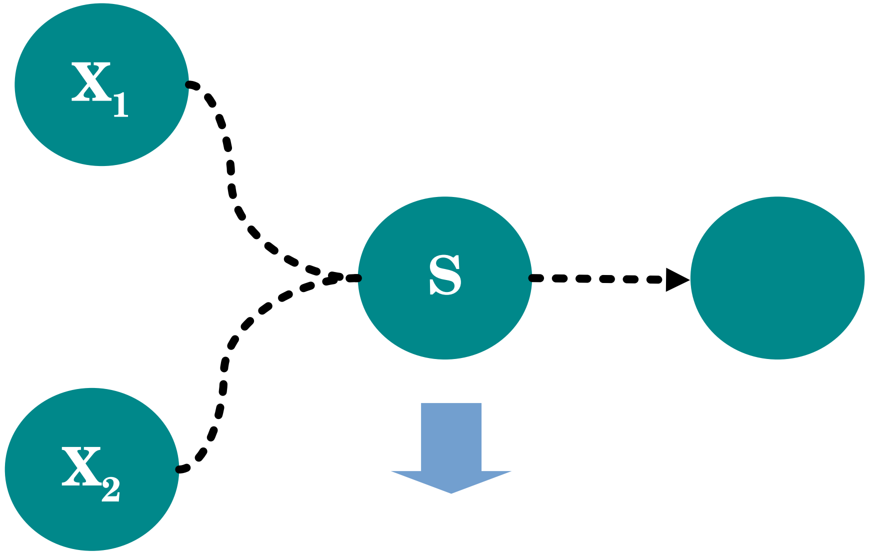


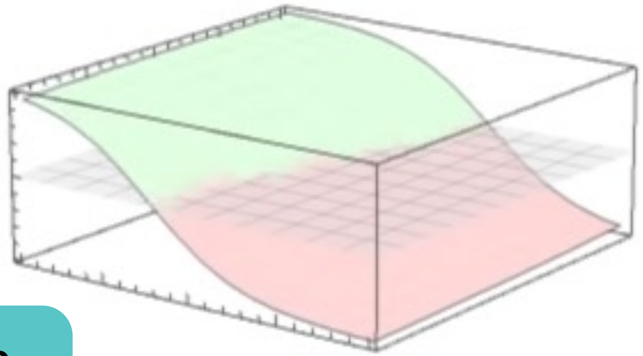
4. Encontrar el hiperplano



4. Encontrar el hiperplano



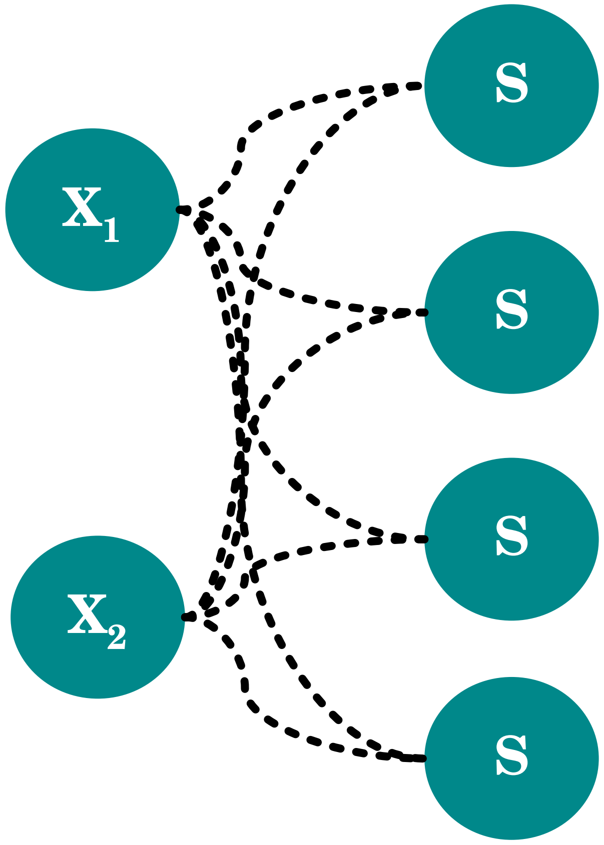




Puede cambiar la orientación cambiando parámetros

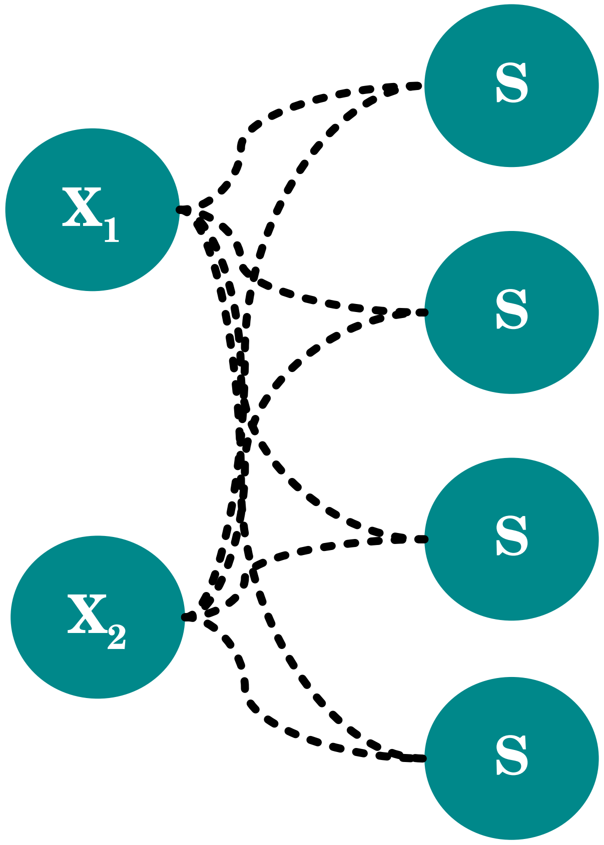
4. Encontrar el hiperplano

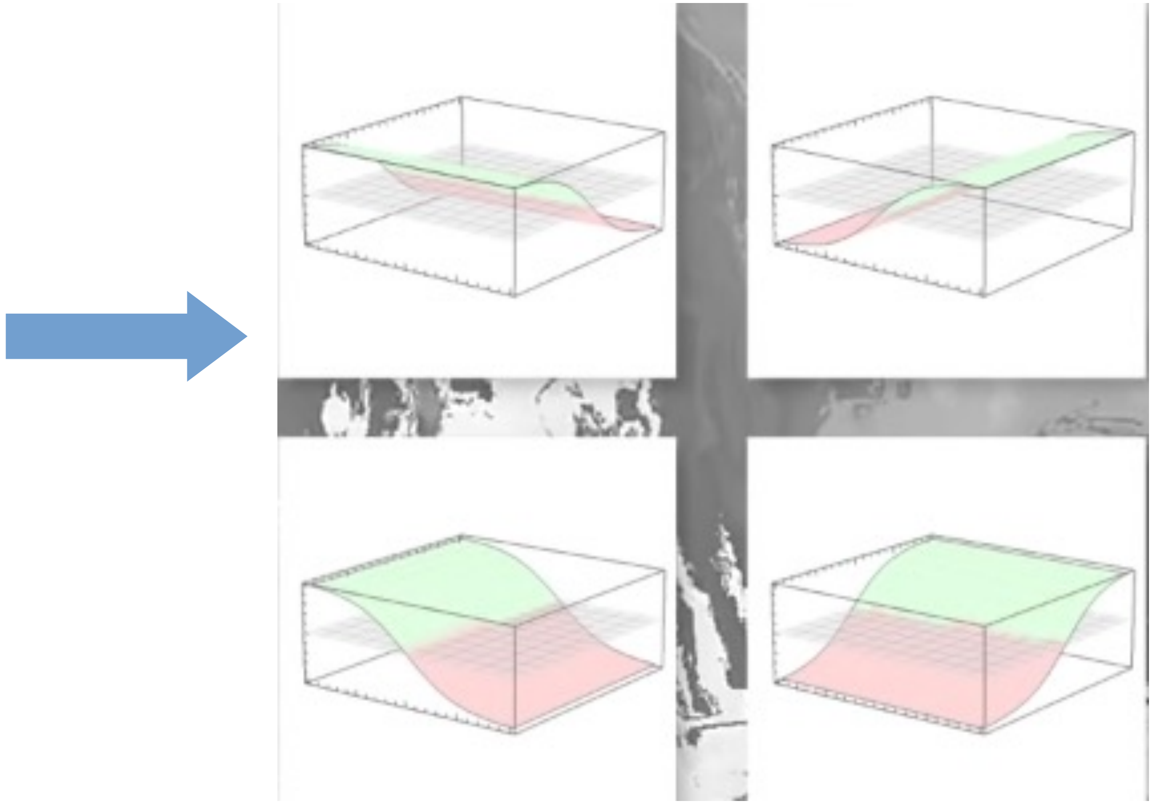
Aumentamos las neuronas



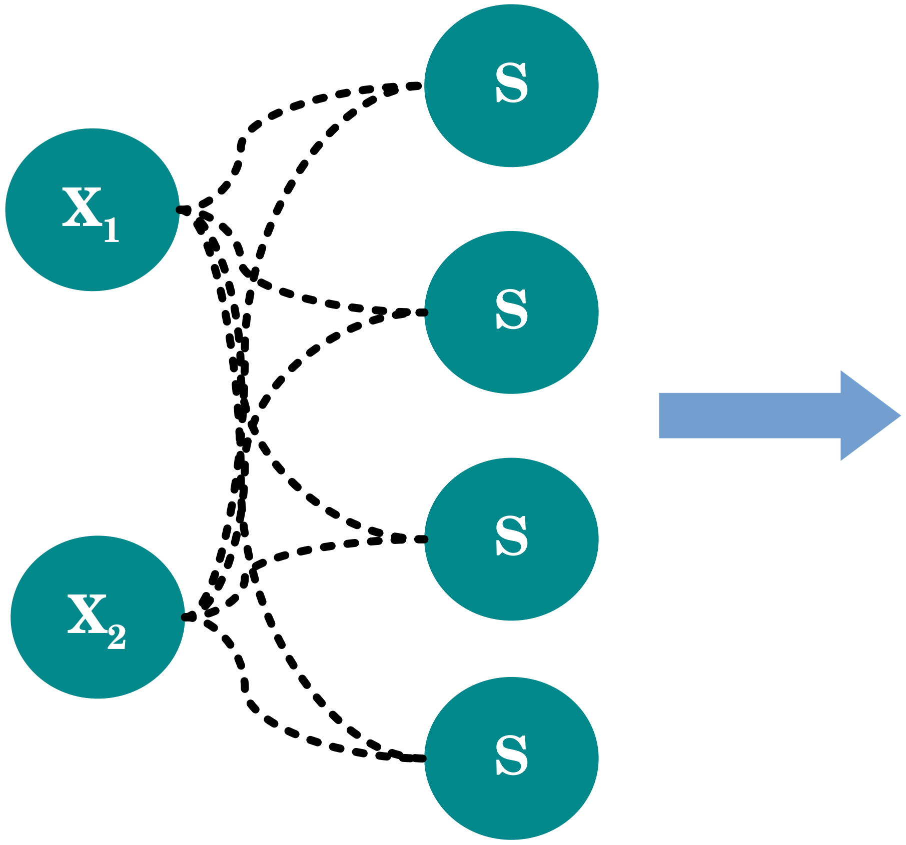
4. Encontrar el hiperplano

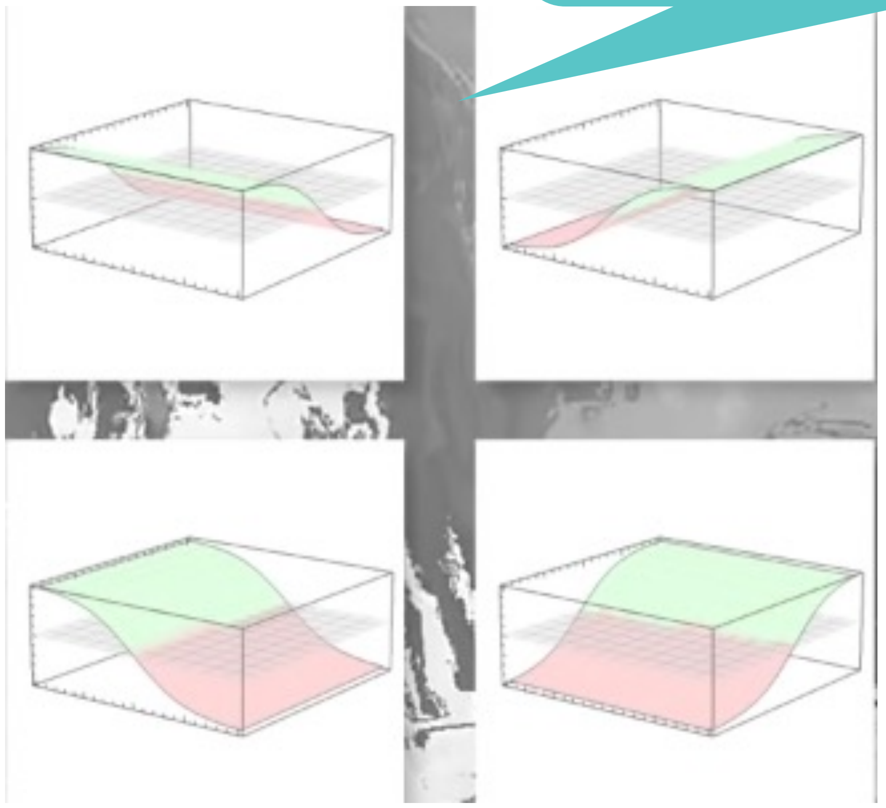
Aumentamos las neuronas





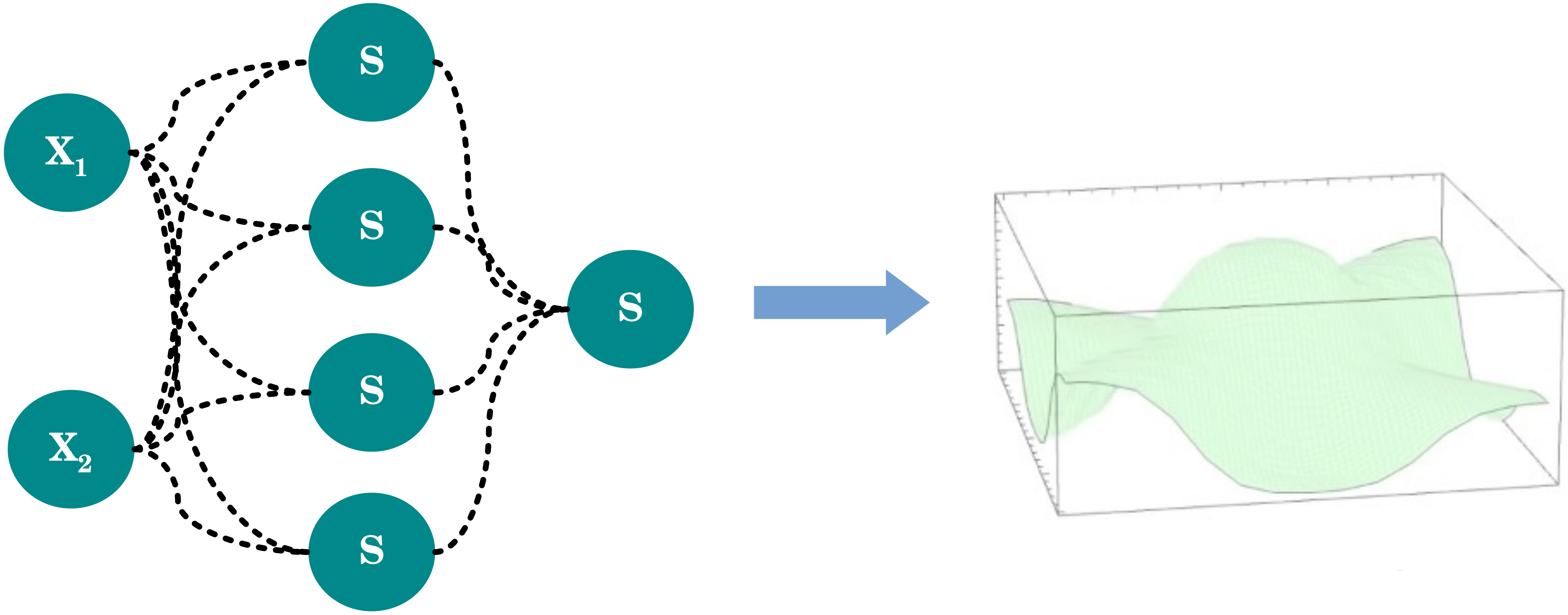
4. Encontrar el hiperplano
Aumentamos las neuronas





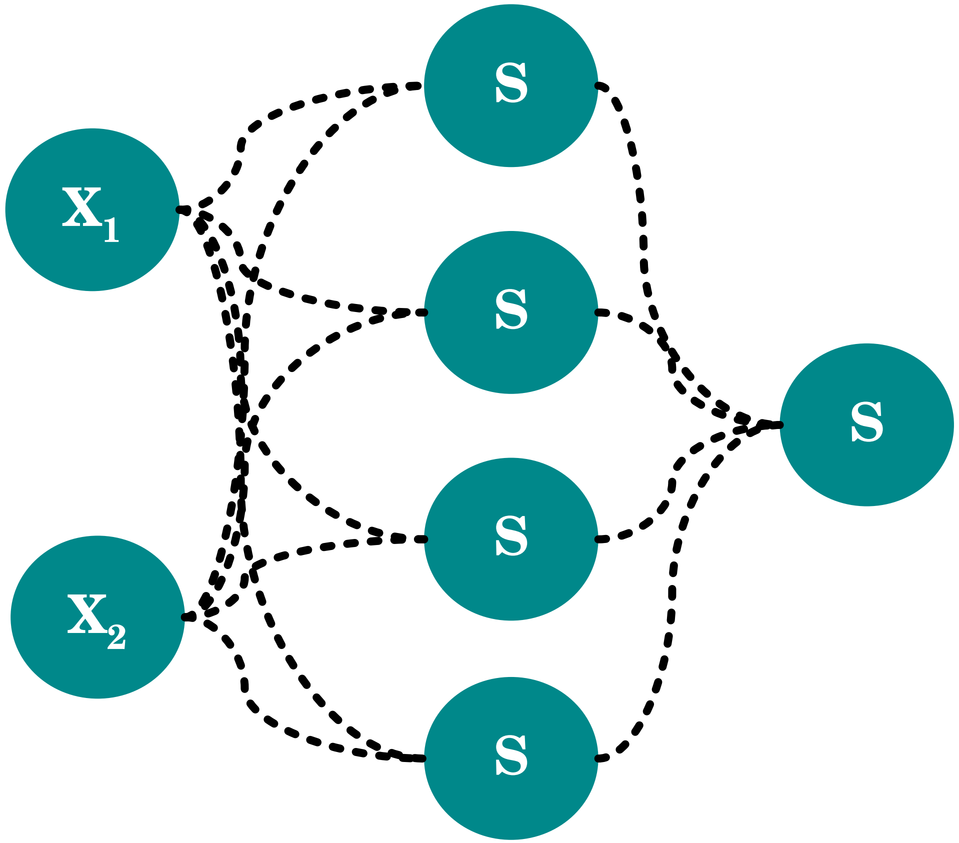
Pero cada una tiene una orientación diferente
Las juntamos…

4. Encontrar el hiperplano
Segunda capa oculta

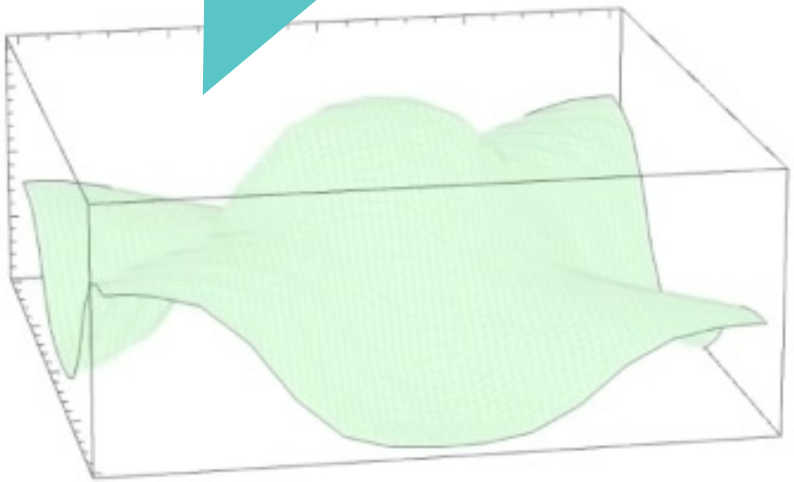


4. Encontrar el hiperplano

Segunda capa oculta

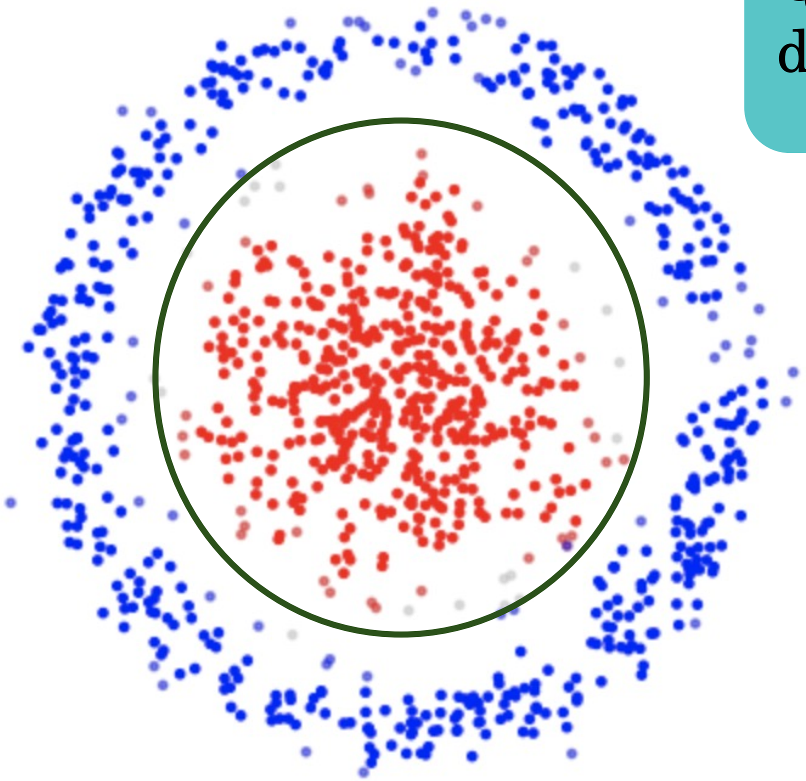


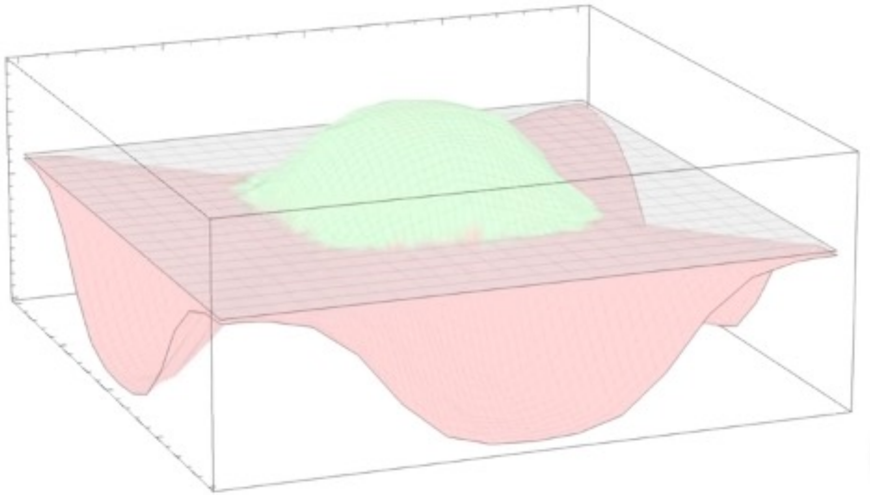
Obtenemos la superposción geométrica con un “bulto” al medio



4. Encontrar el hiperplano
Segunda capa oculta

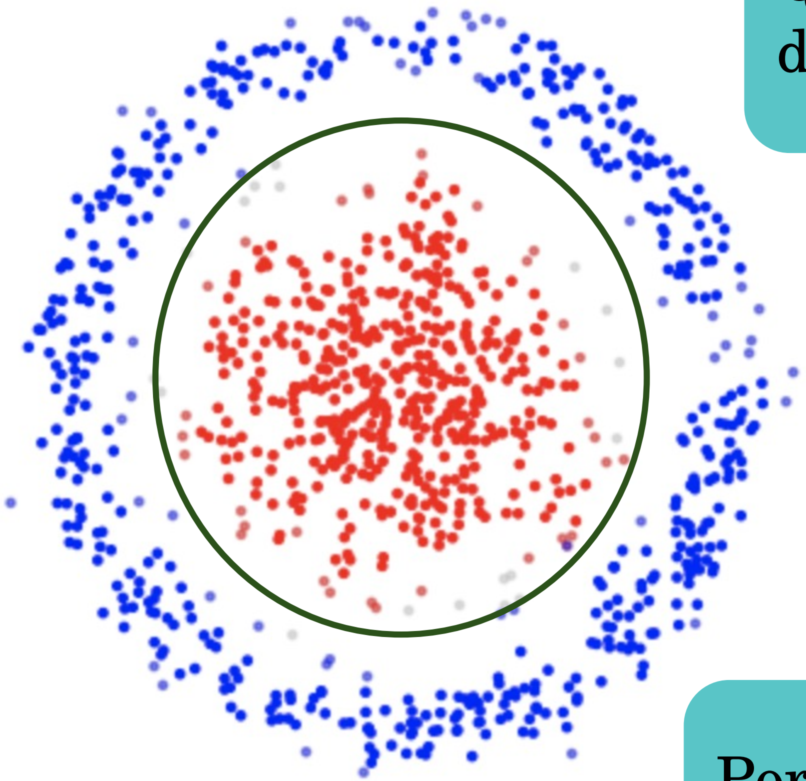
Qué podemos "cortar" con la intersección del hiperplano con la "montaña"



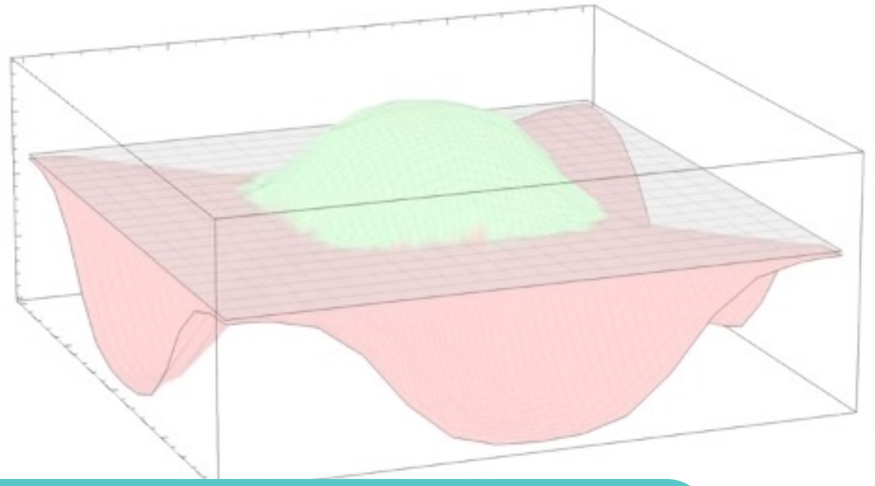


4. Encontrar el hiperplano

Segunda capa oculta



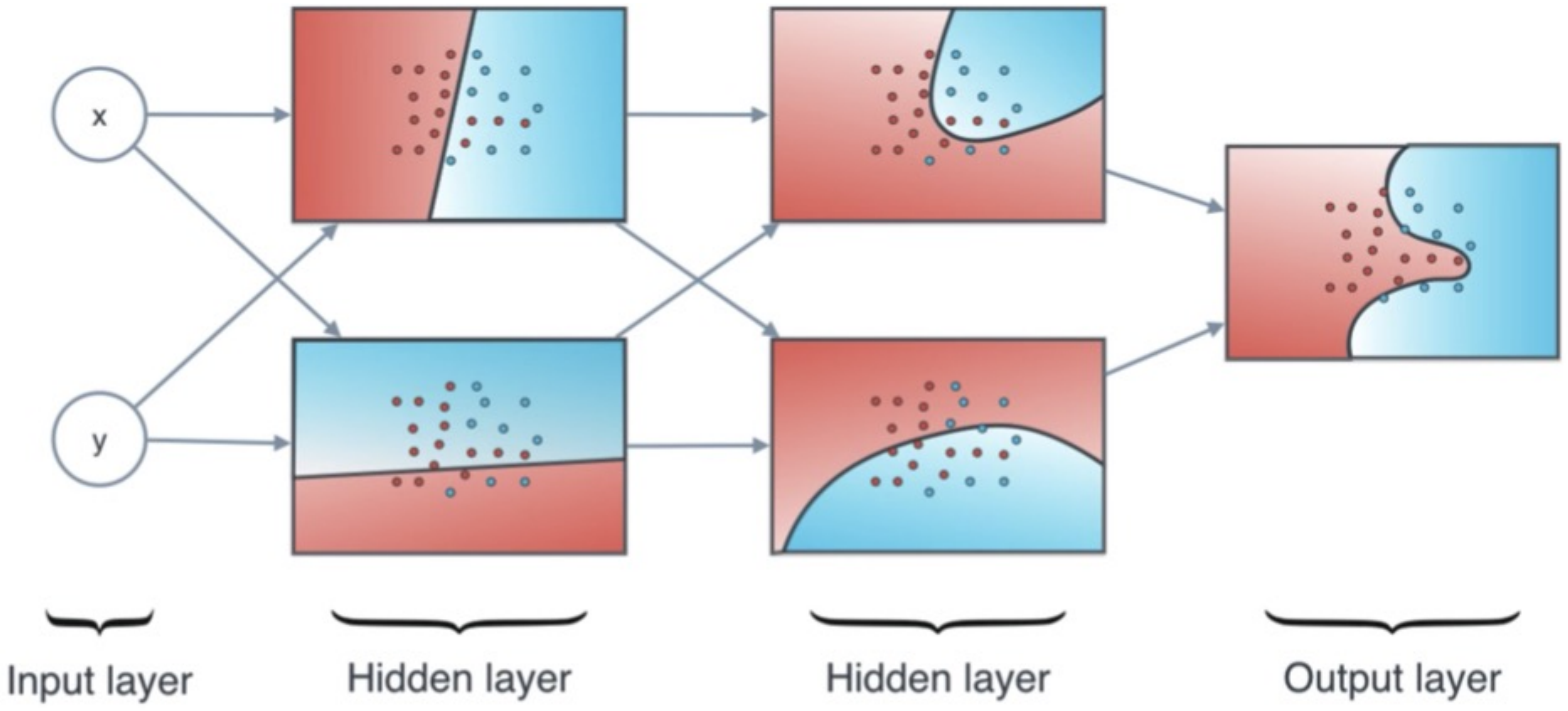
Qué podemos "cortar" con la intersección del hiperplano con la "montaña"



Por lo que se pueden resolver problemas complejos

5. Arquitectura de redes neuronales

- Son la base de las **redes profundas**, actualmente muy utilizadas con mucho éxito.
- Cada **capa extrae características** cada vez más complejas
  - Necesitamos un conjunto de neuronas para resolver problemas no lineales.

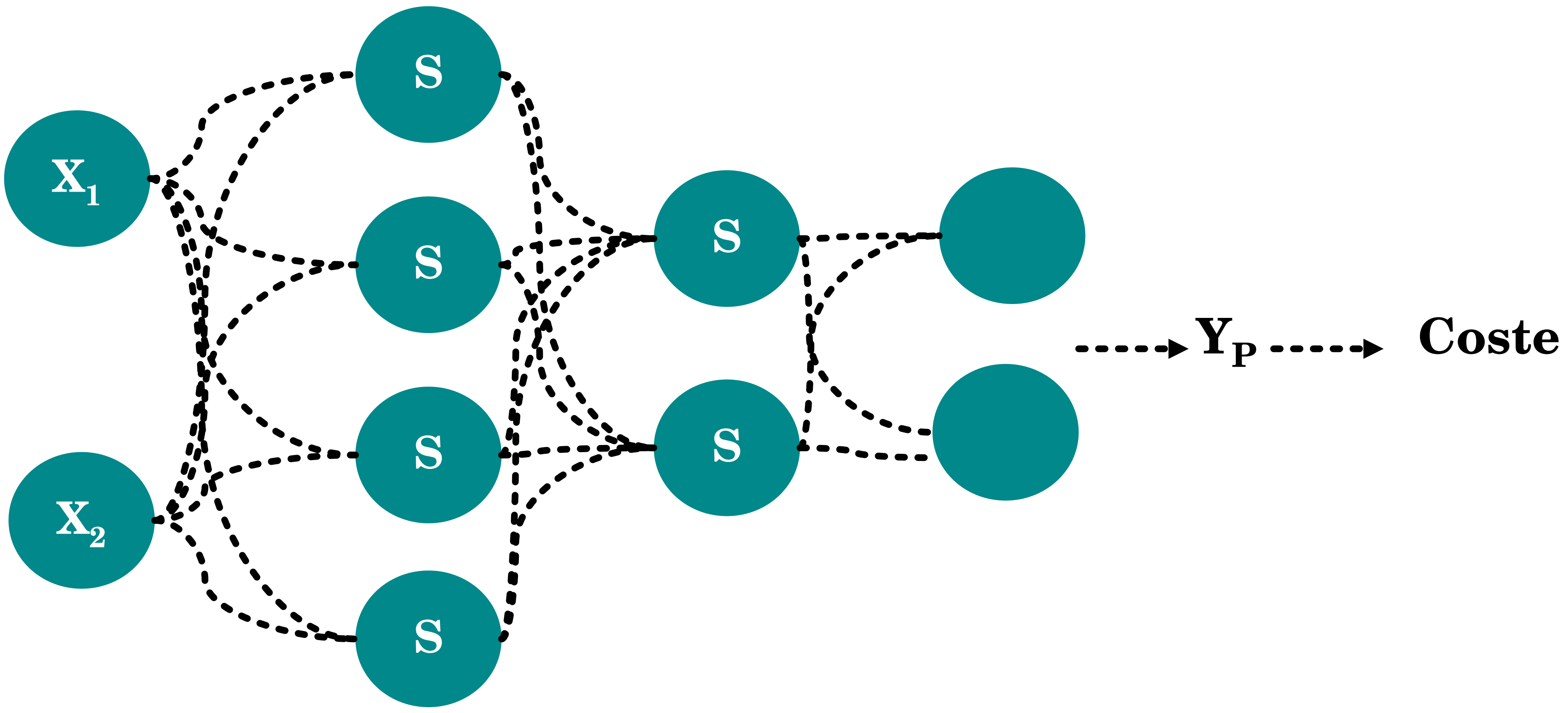


Backpropagation

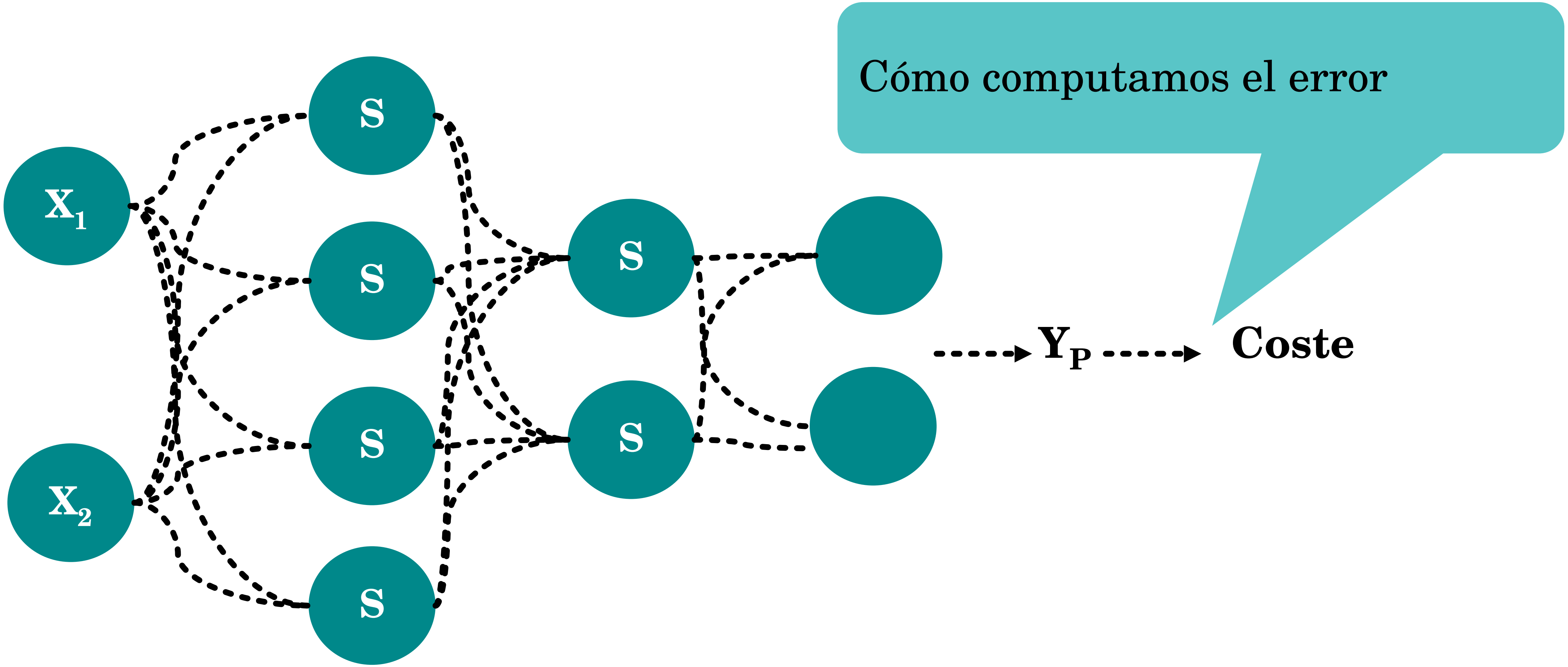
ETS de Ingeniería Informática

UNED

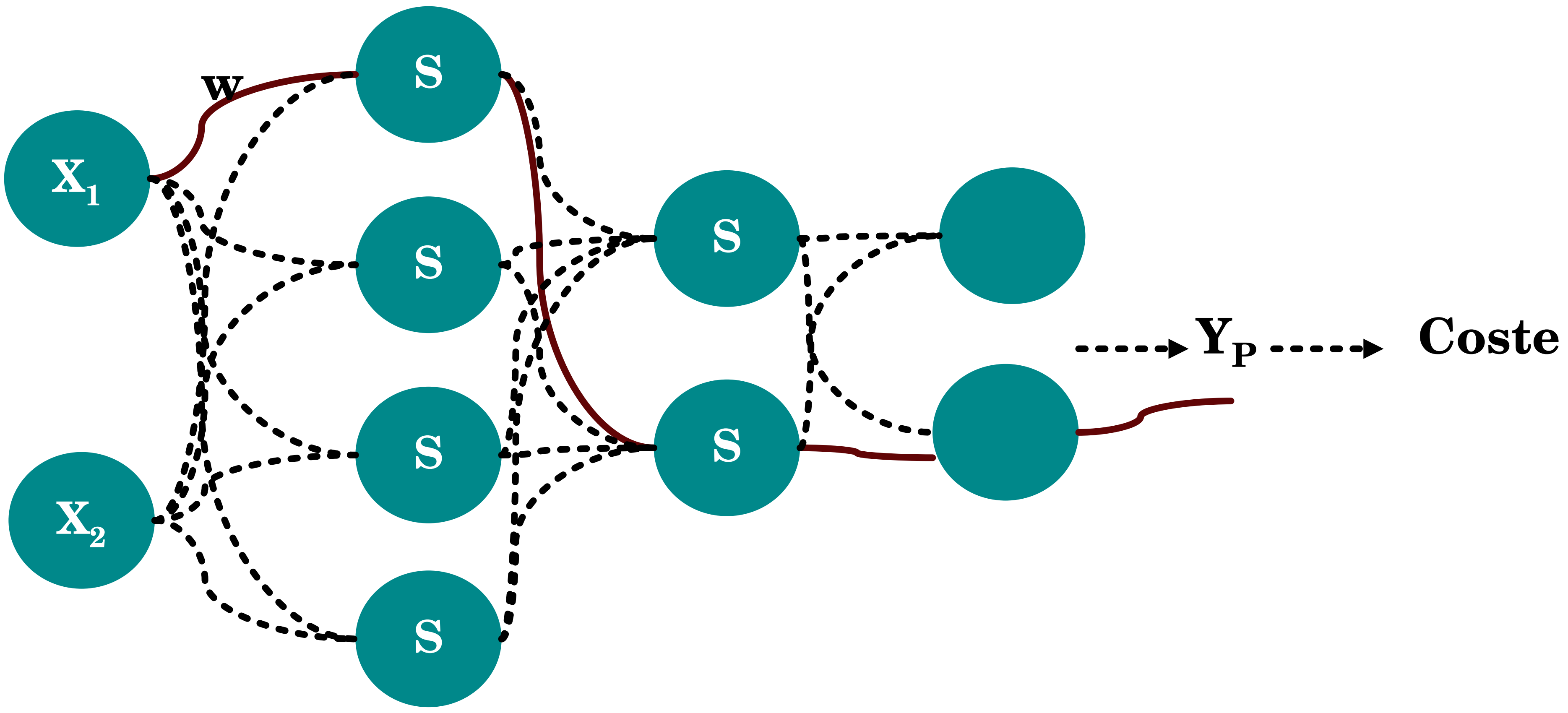
1. Contexto



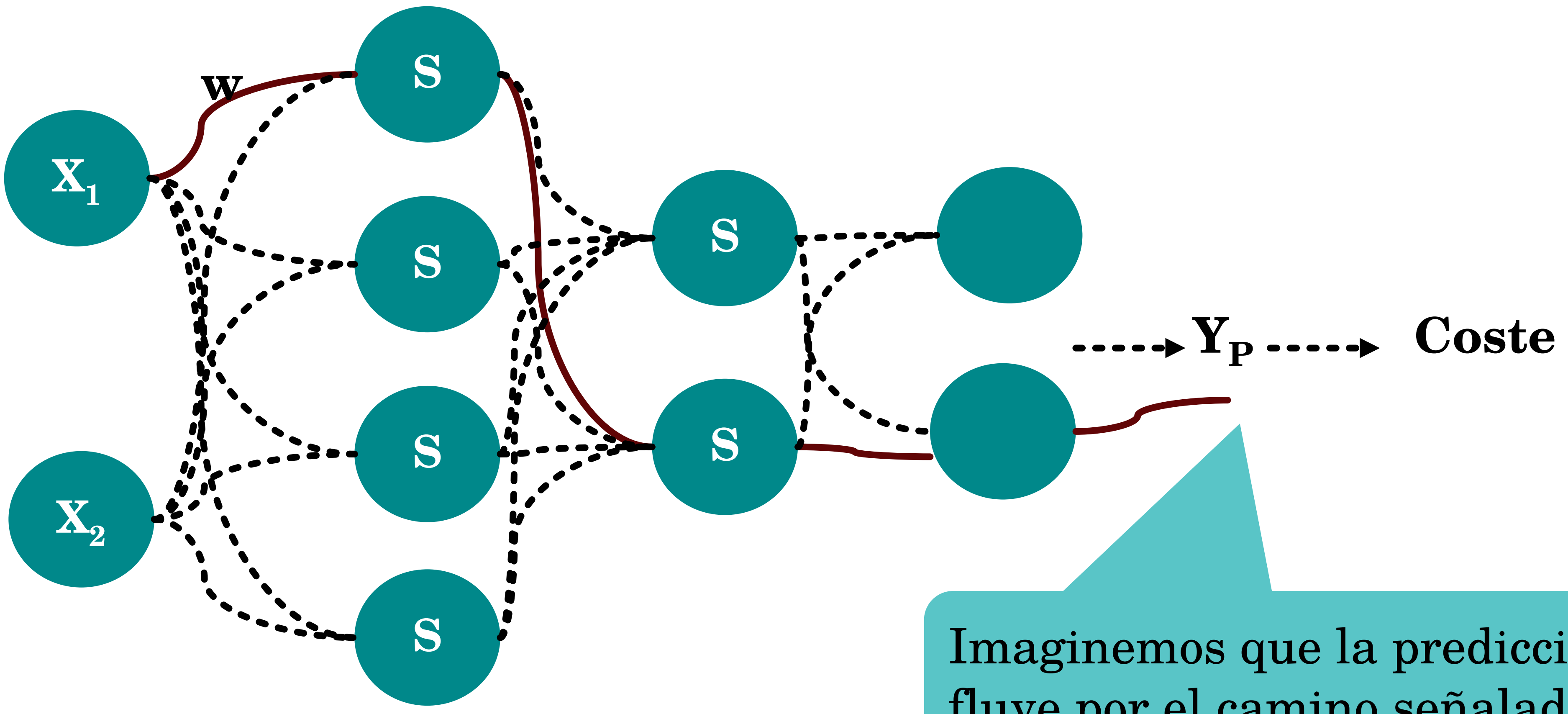
2. Conexiones e iteraciones



2. Conexiones e iteraciones



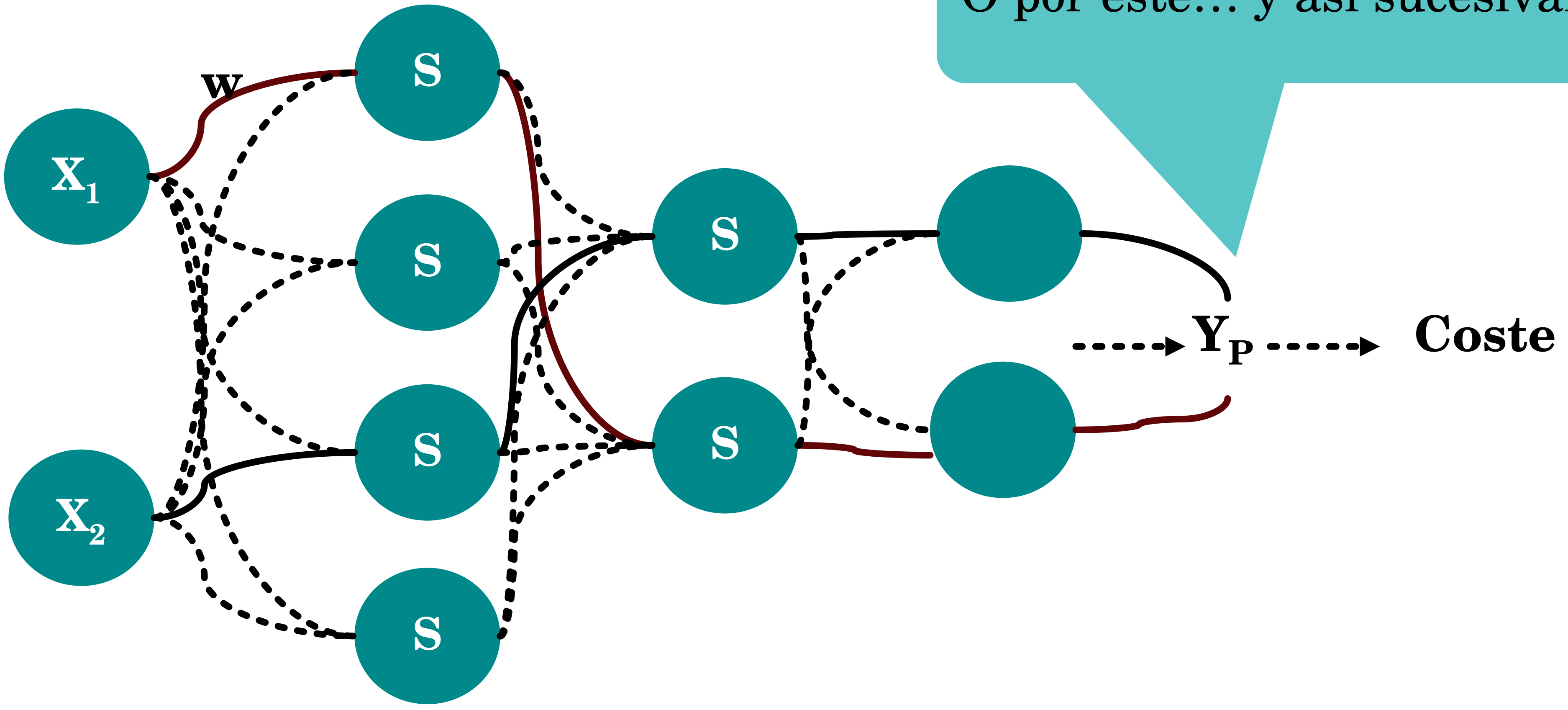
2. Conexiones e iteraciones



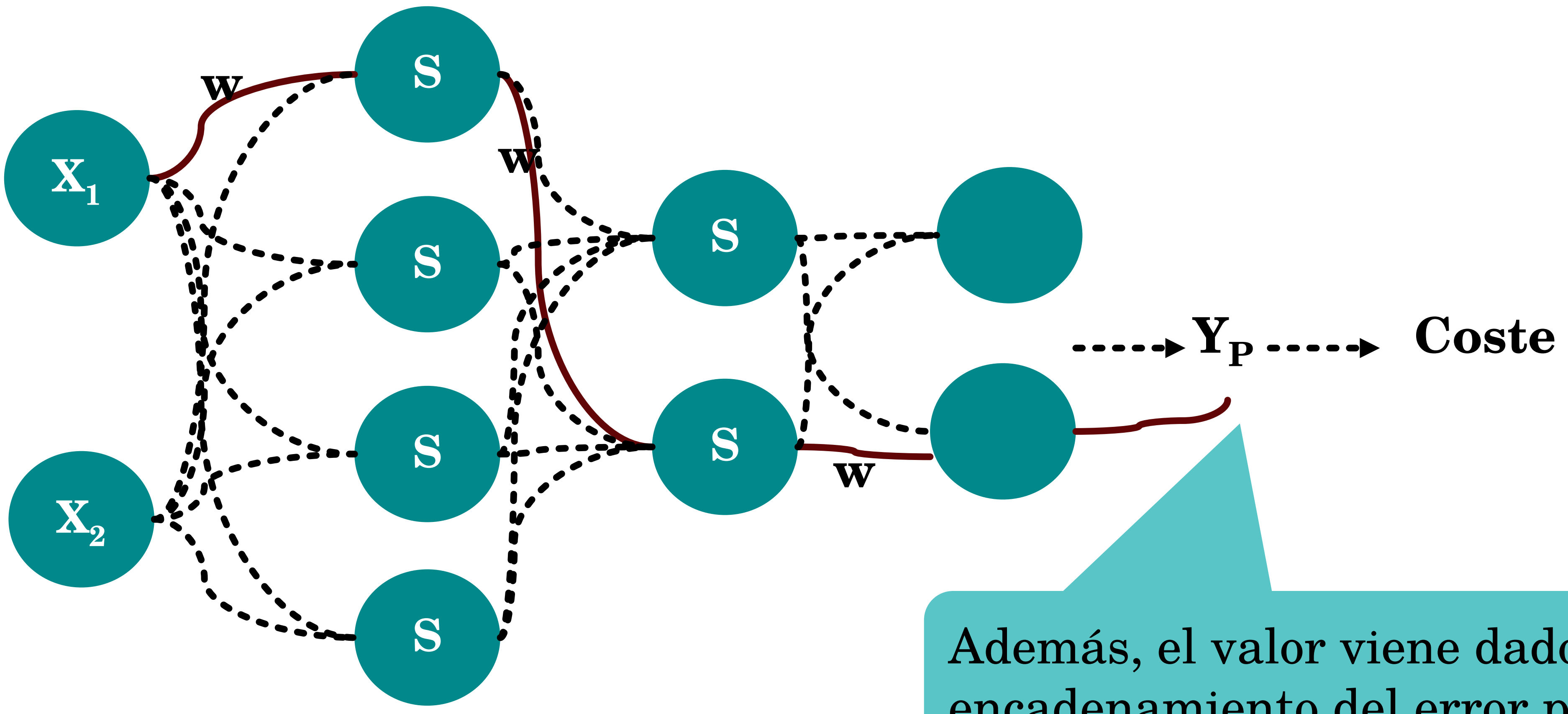
Imaginemos que la predicción fluye por el camino señalado

2. Conexiones e iteraciones

O por este… y así sucesivamente

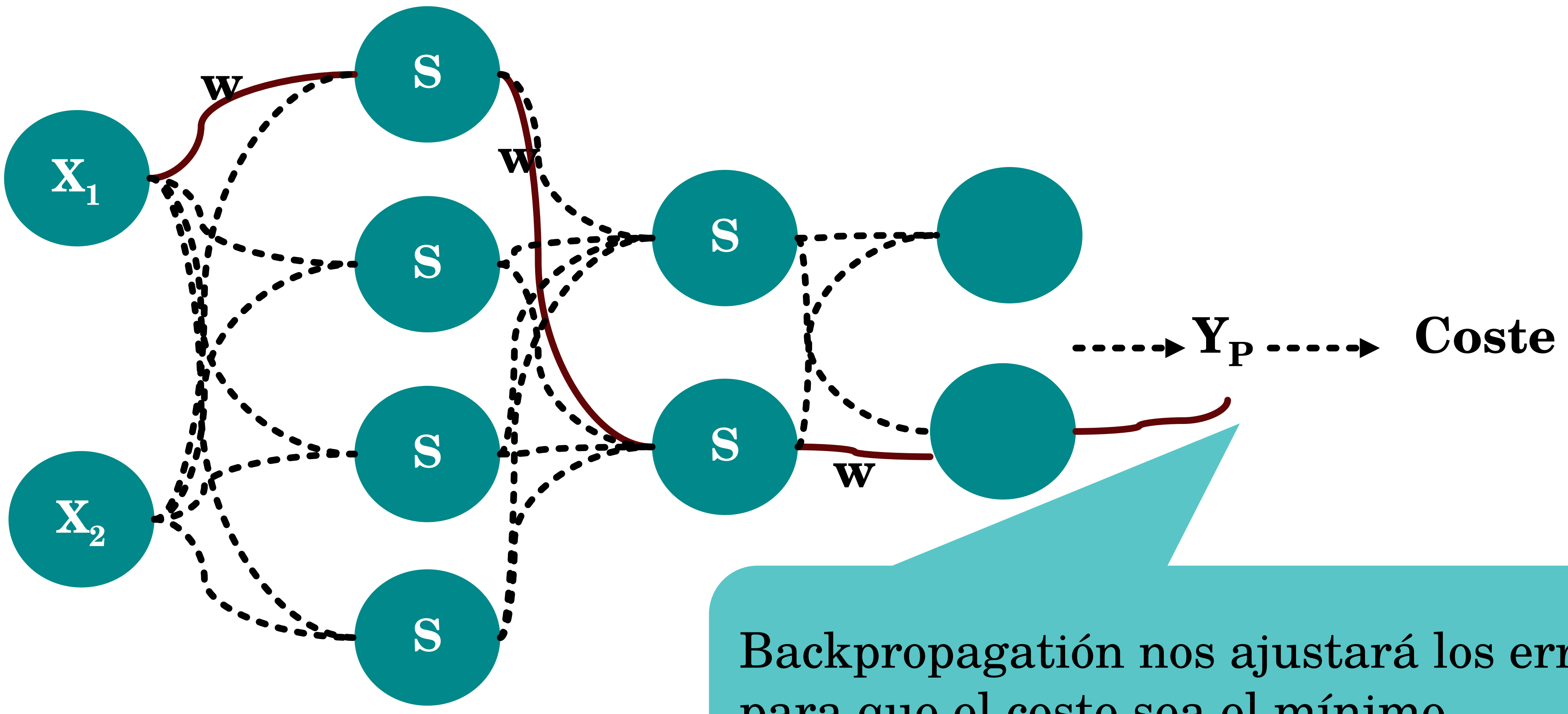


2. Conexiones e iteraciones



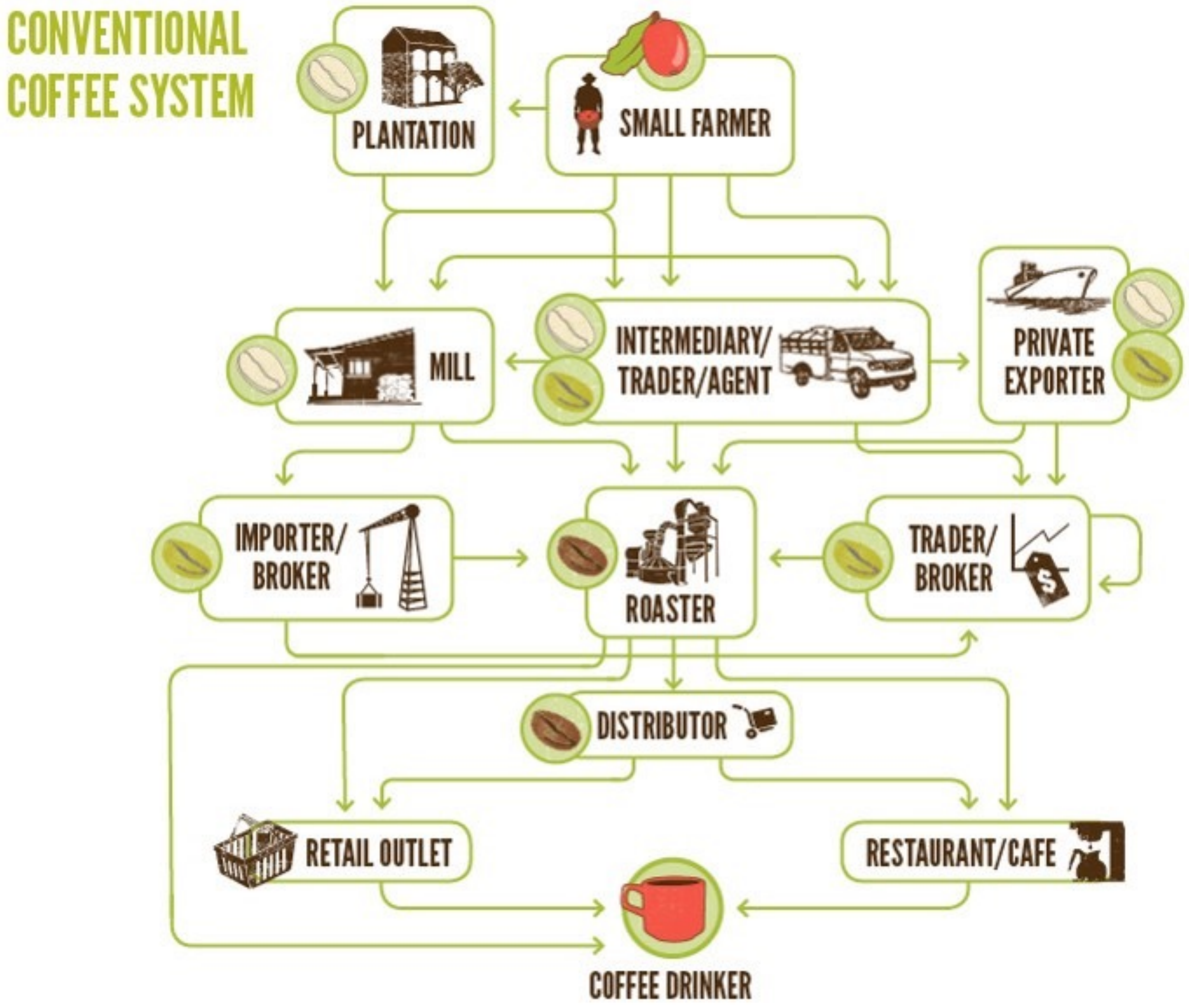
Además, el valor viene dado por encadenamiento del error previo

2. Conexiones e iteraciones

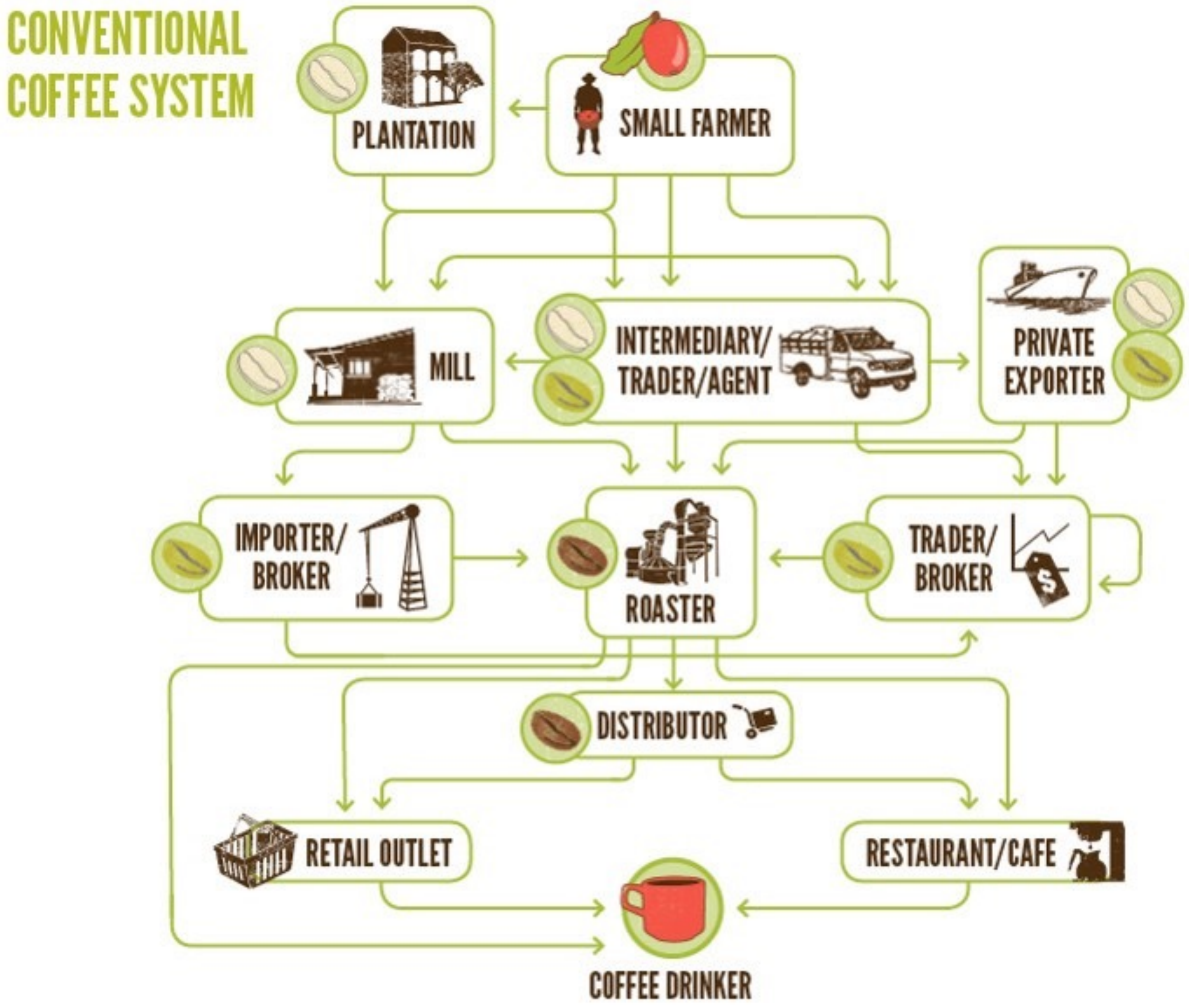


Backpropagatión nos ajustará los errores para que el coste sea el mínimo.
Para ello, utilizará el Gradiente Descendiente

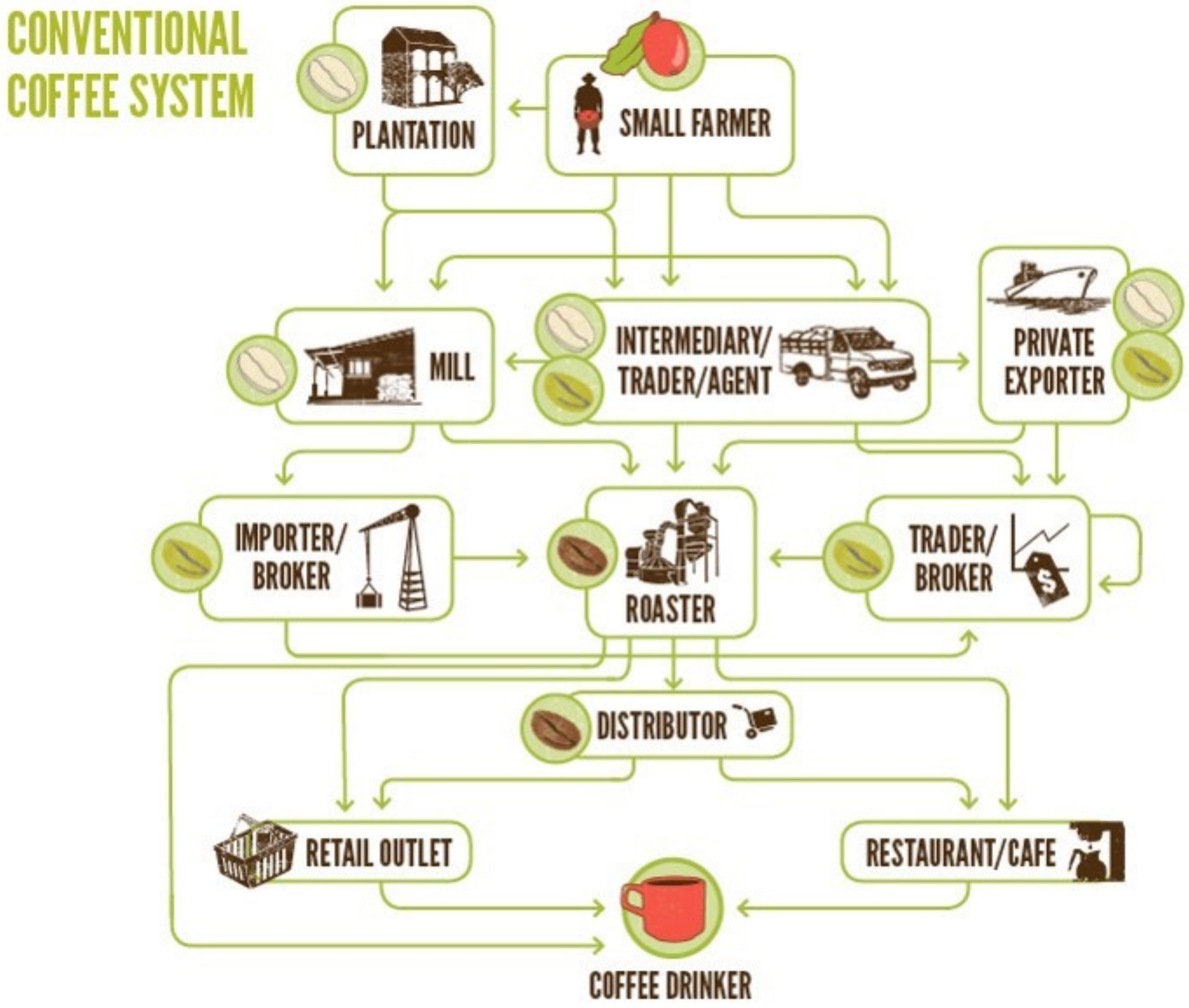
3. Pensemos en el día a día



3. Pensemos en el día a día

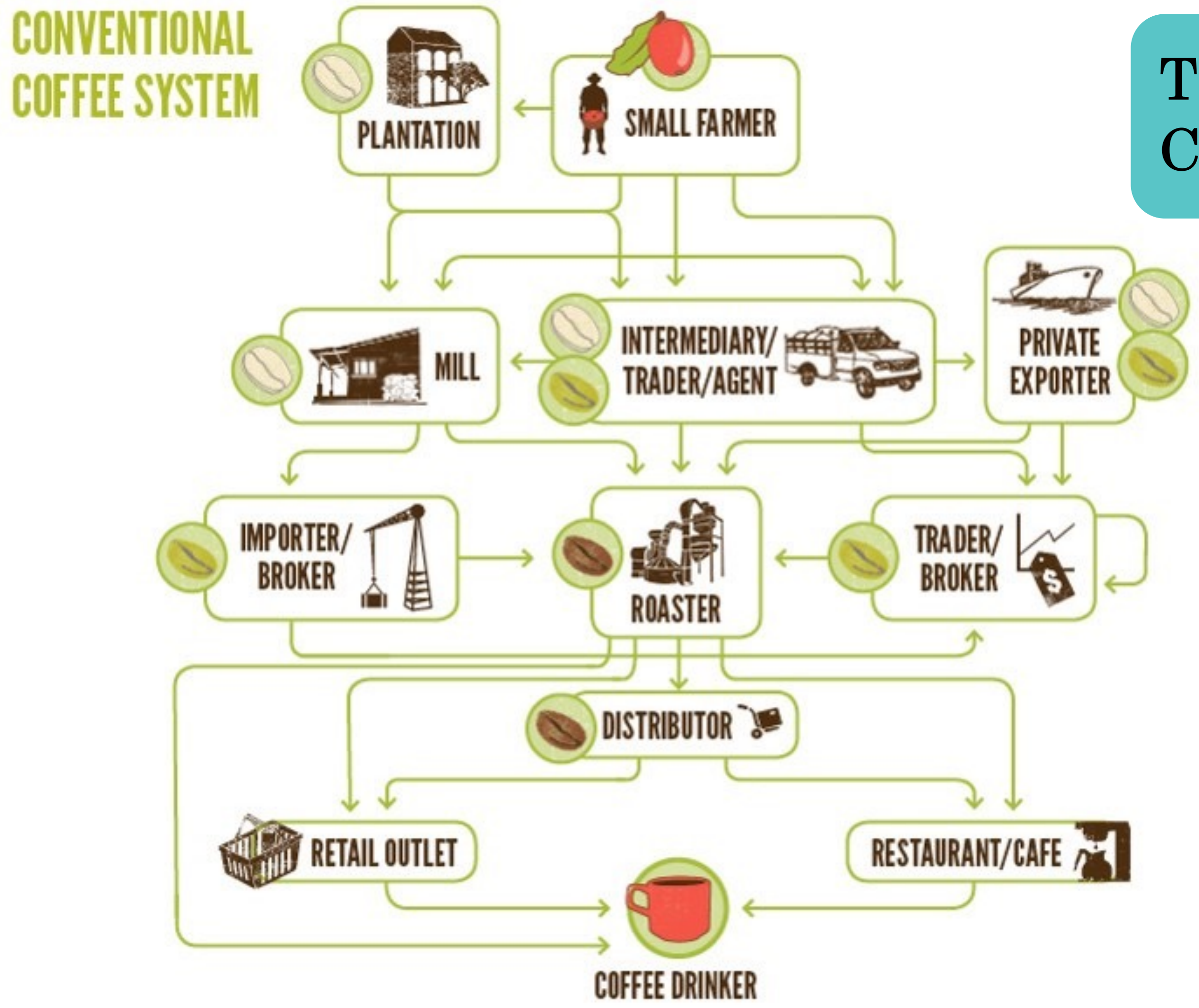


3. Pensemos en el día a día



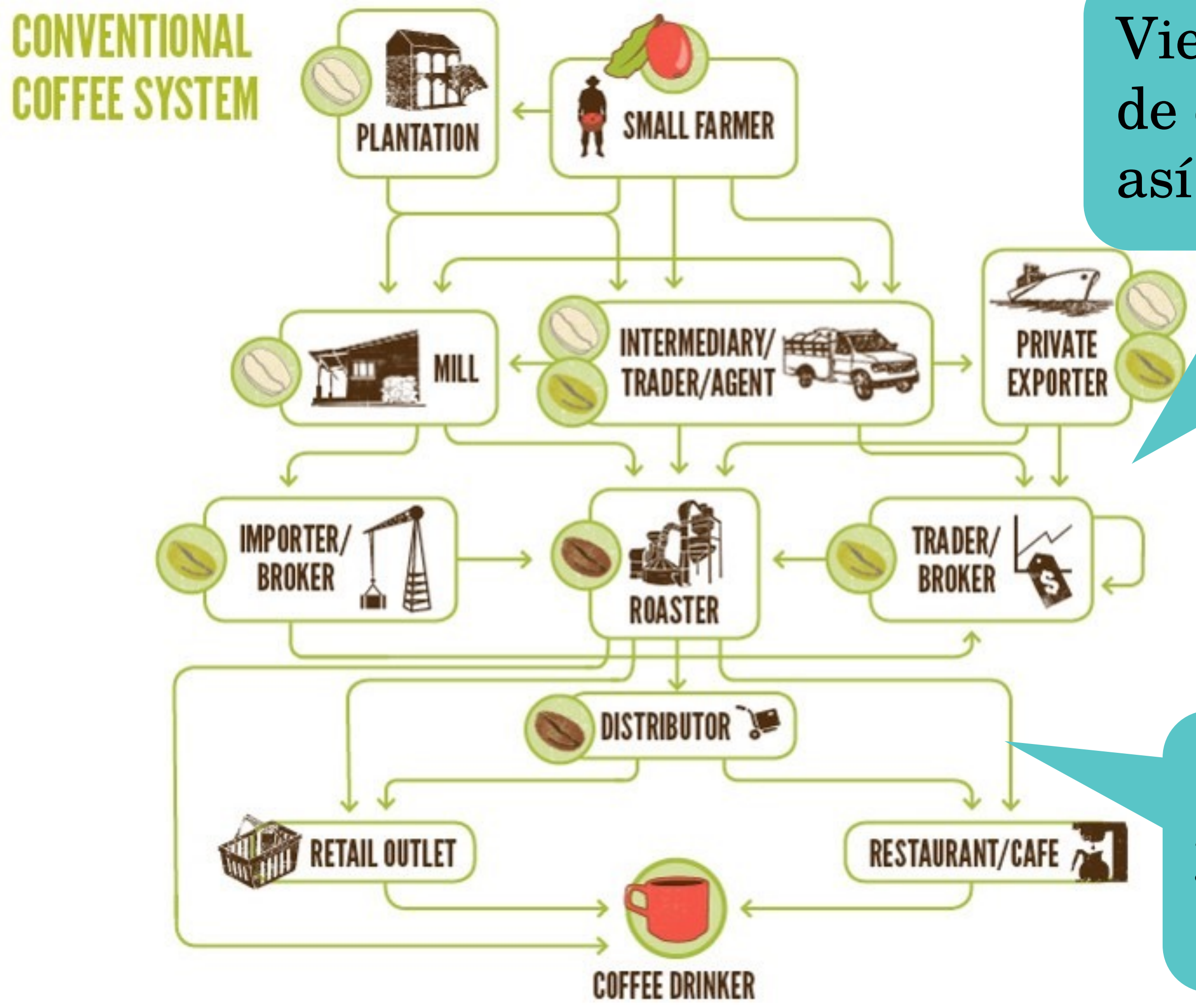
¿De quién es la culpa?

3. Pensemos en el día a día



Tenemos que analizar toda la Cadena para ver quién es el culpable

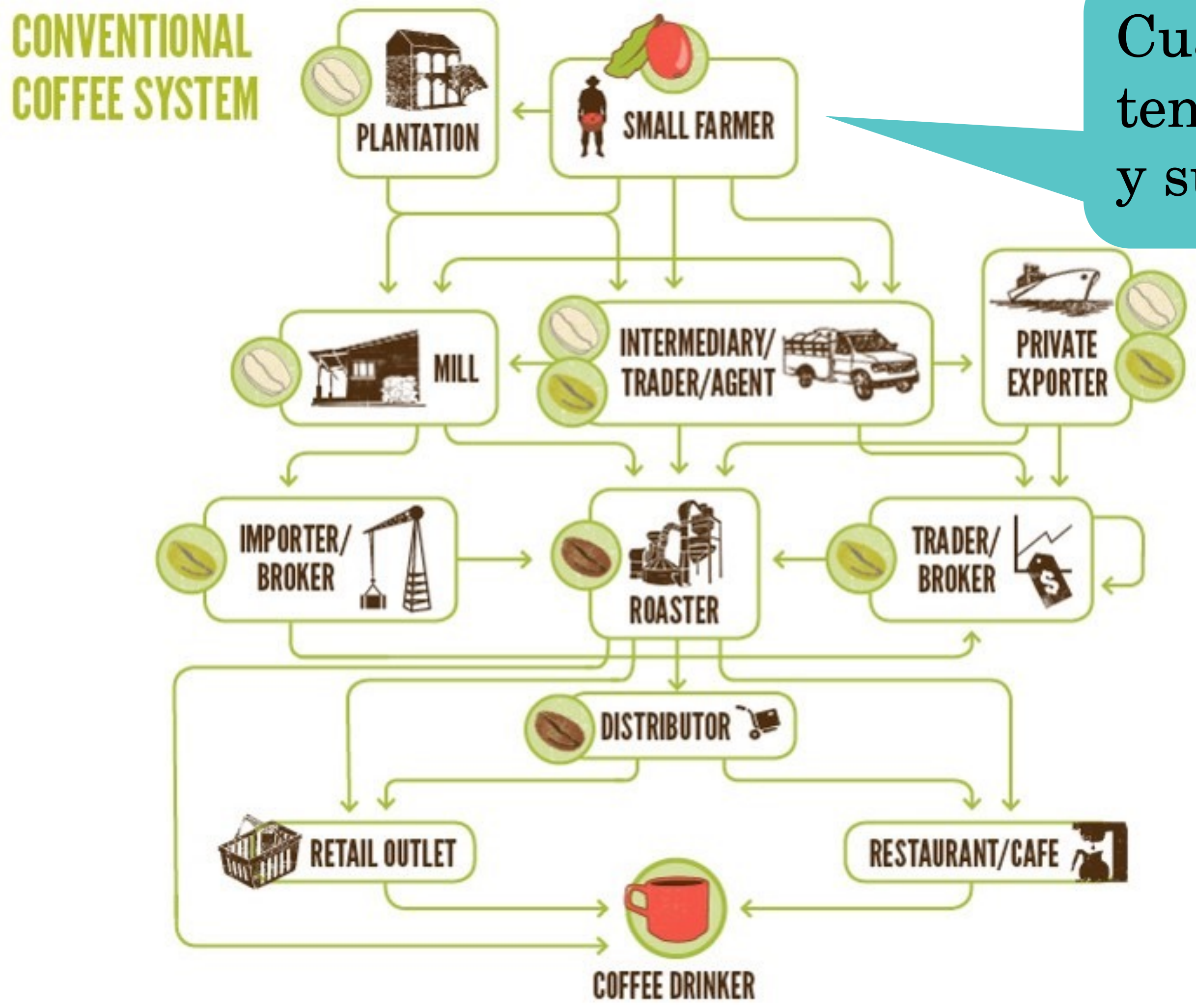
3. Pensemos en el día a día



Viendo el porcentaje de implicación de cada neurona en la predicción y, así, corregirlo

Para ello, analizamos el error y vamos analizando hacia atrás para corregirlo

3. Pensemos en el día a día



Cuando lleguemos a la primera capa tendremo el error de cada neurona y sus parámetros

¡Gracias!

Dr. Manuel Castillo-Cara
www.manuelcastillo.eu
Departamento de Inteligencia Artificial
Escuela Técnica Superior de Ingeniería Informática
Universidad Nacional de Educación a Distancia (UNED)