

# Introducción a las Redes Neuronales & Aprendizaje profundo

Programa en Analítica

Curso *Capstone - Inteligencia Artificial & Deep Learning*  
Analítica Prescriptiva

Educación continua | Universidad de los Andes  
octubre 6 – noviembre 24  
2021

# Hoy

De qué vamos a hablar

## 1. Introducción a las Redes Neuronales:

*¿Qué representa el modelo? ¿Cómo se ve?  
¿Qué significa (interpretable)? Ejemplos de aplicación.*

## 2. Entrenamiento de ANN:

*¿Cómo funcionan los métodos para que las redes aprendan? Hay paquetes en R ya programados.*

## 3. Nota sobre overfitting:

*Cuando sobre ajustamos a los datos conocidos...*



Photo by [nappy](#) from [Pexels](#)

# Pero antes

Notas generales

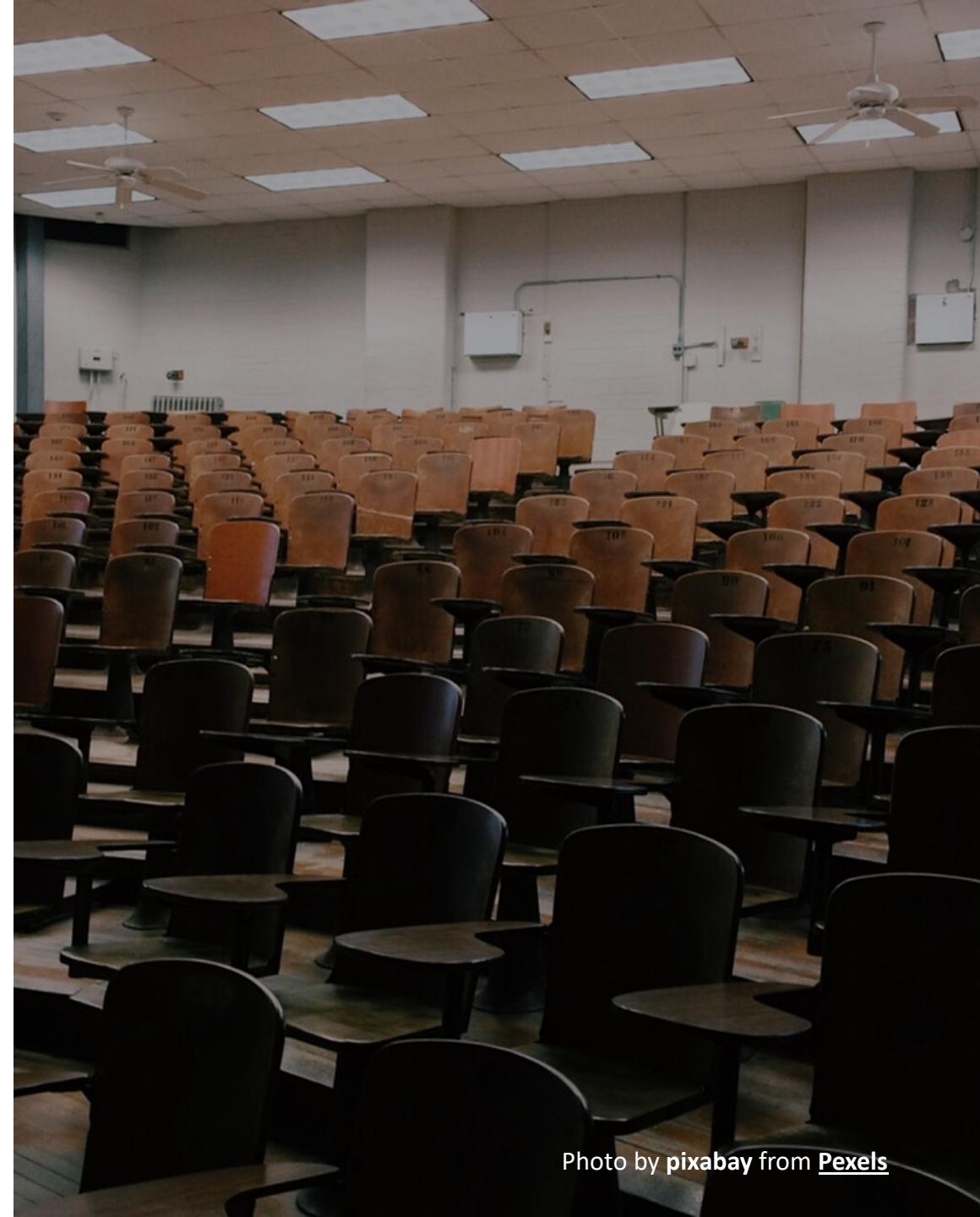


Photo by [pixabay](#) from [Pexels](#)

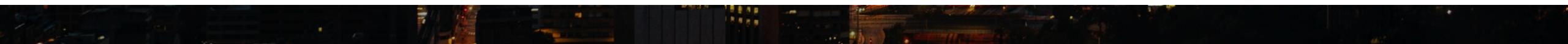
# Sobre el curso

- Esta semana no vamos a darle tiempo de clase al proyecto.
  - PERO: hay retroalimentación en el repositorio, terminen de formalizar reto. Lean su retroalimentación, agendemos vernos un ratico con los demás grupos, escríbanme ante dudas e inquietudes si quieren hablar.
  - Y descarguen los datos. La siguiente semana vamos a trabajar sobre eso.
- No todos/as están pensando un reto de analítica prescriptiva
  - Un reto de prescriptiva siempre tiene un conjunto de decisiones **concreto** en mente, entre las cuales tomar una decisión (puede ser continua o discreta)
  - Caso confuso: exploración predictiva para luego tomar decisiones diferentes para cada posible predicción (ej. clusterización)



# Sobre el curso

- No todos/as están pensando un reto de analítica prescriptiva
  - Ojo: tener una predicción no es lo mismo que decidir qué hacer con esa predicción. Distinción **criterio de decisión** vs implicaciones de la decisión.
  - Caso confuso: en algunos de sus retos: ¡las posibles decisiones eran las mismas posibles predicciones! (cuidado con automatizar a ciegas)
  - Caso confuso: el criterio de decisión es “muy obvio” entonces queda implícito (ej. La máquina predice qué tipo de transacciones son más características de una entidad, la decisión puede ser supervisar más ese tipo de transacciones)



# Un aprendizaje de la clase pasada

- Si el profesor pasa por cada grupo, el promedio es mínimo dedicar una hora entera (10 minutos por grupo) a retroalimentación personalizada (talleres futuros son de 2 horas, o bien colectivos).
- Dos momentos: 1) escuchar, 2) absorber, 3) reflexionar: si se me ocurren cosas nuevas, les voy a estar escribiendo.
- Talleres son literalmente tiempo para trabajar en sus proyectos: así el profesor no esté con ustedes, la experiencia constituye un entorno de aprendizaje valioso: aprovechenlo.



# Piensen en formalizar su reto de analítica así:

¿Qué decisiones pueden tomar?

Decisión 1

Decisión 2

Decisión 3

Decisión 4

Para que una máquina aprenda cómo mi decisión afecta mi variable de criterio, hay que tener datos históricos

¿Con qué criterio de decisión vamos a usar la predicción del modelo para escoger la decisión?

Info útil	Info útil



TAL COSA

min/max TAL COSA

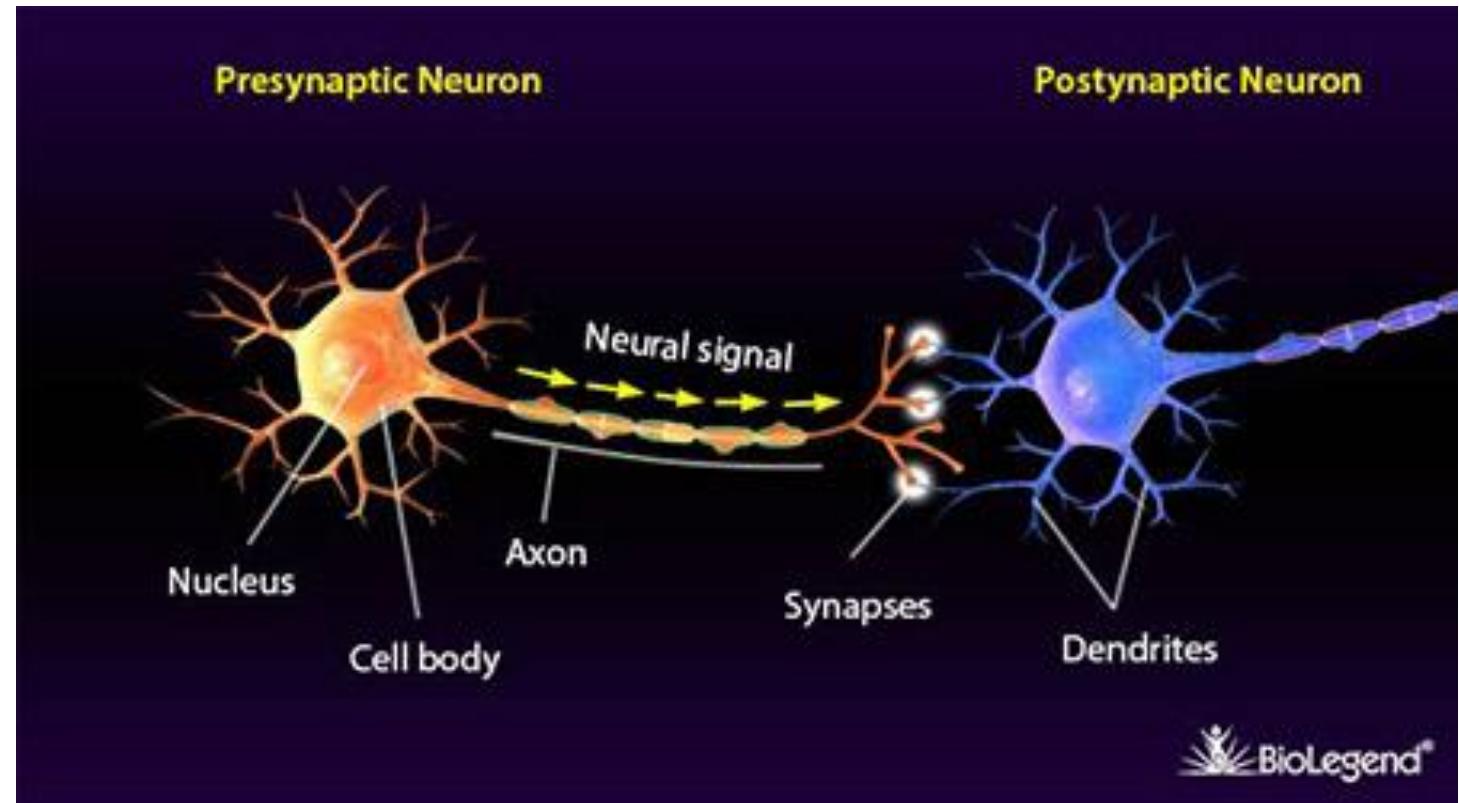
# Introducción a las Redes Neuronales

Artificial Neural Networks (ANN) – Un  
modelo inspirado en la naturaleza



# Redes neuronales: la sinapsis

- Tres de las partes:
  - Cuerpo
  - Axón
  - Dentritas
- Pueden emitir cargas electroquímicas
- Se pueden estimular mutuamente



# Las neuronas están activas/inactivas

- Imágenes reales de neuronas en la corteza prefrontal:

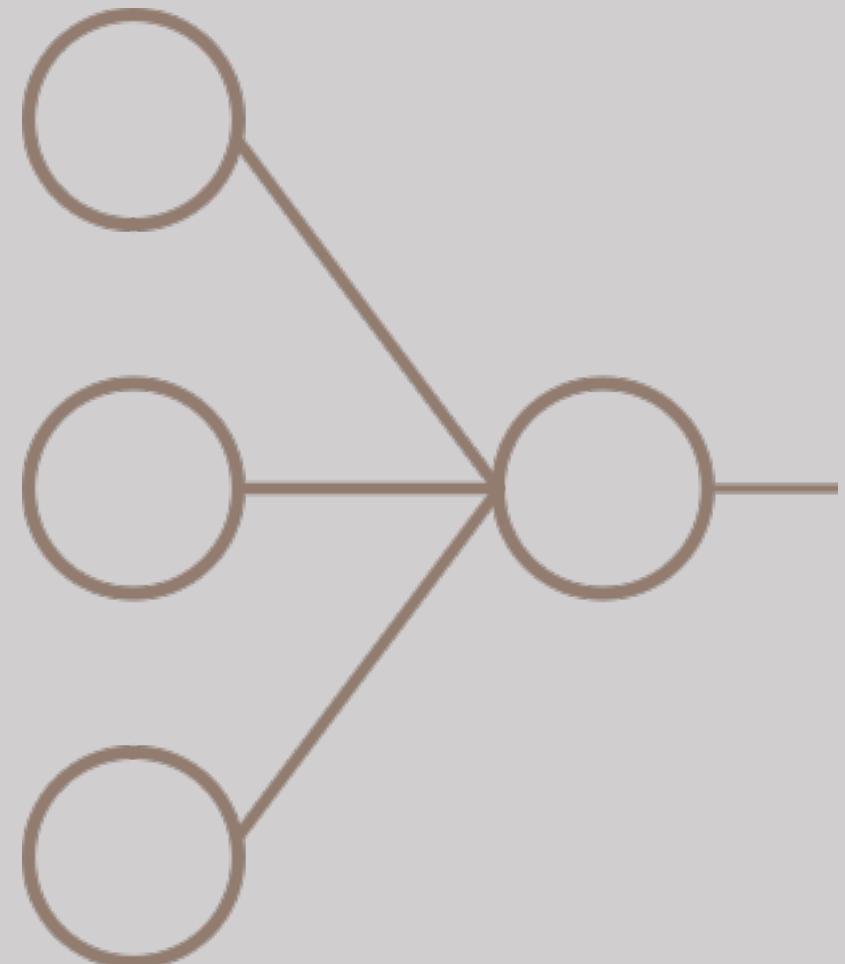
[https://youtu.be/m0rHZ\\_RDdyQ](https://youtu.be/m0rHZ_RDdyQ)

Una vez la carga electroquímica alcanza cierto umbral, se dispara una señal de activación para otras neuronas:

si bien la carga es continua, el estado de activación de las neuronas es binario

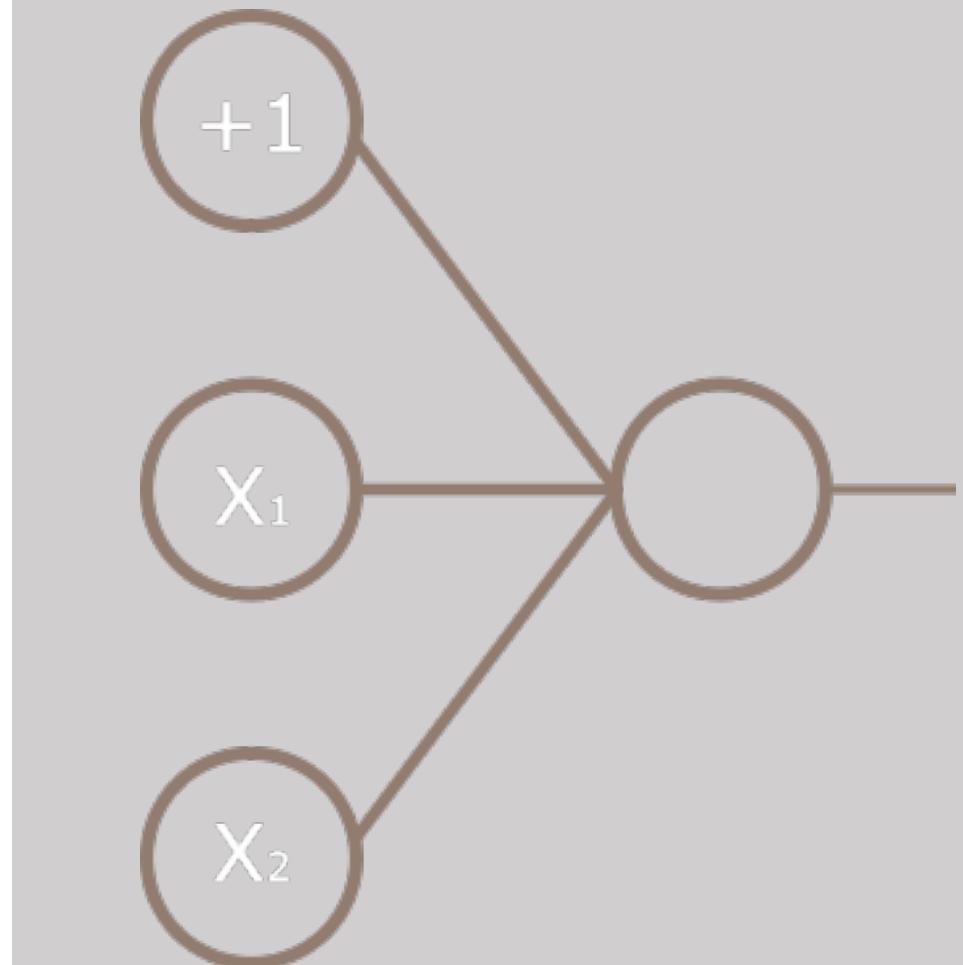
# La representación formal

- Una neurona se representa por un círculo
- Un axón se representa por una línea, indica qué neuronas están conectadas con qué otras
- El círculo a la derecha es otra neurona



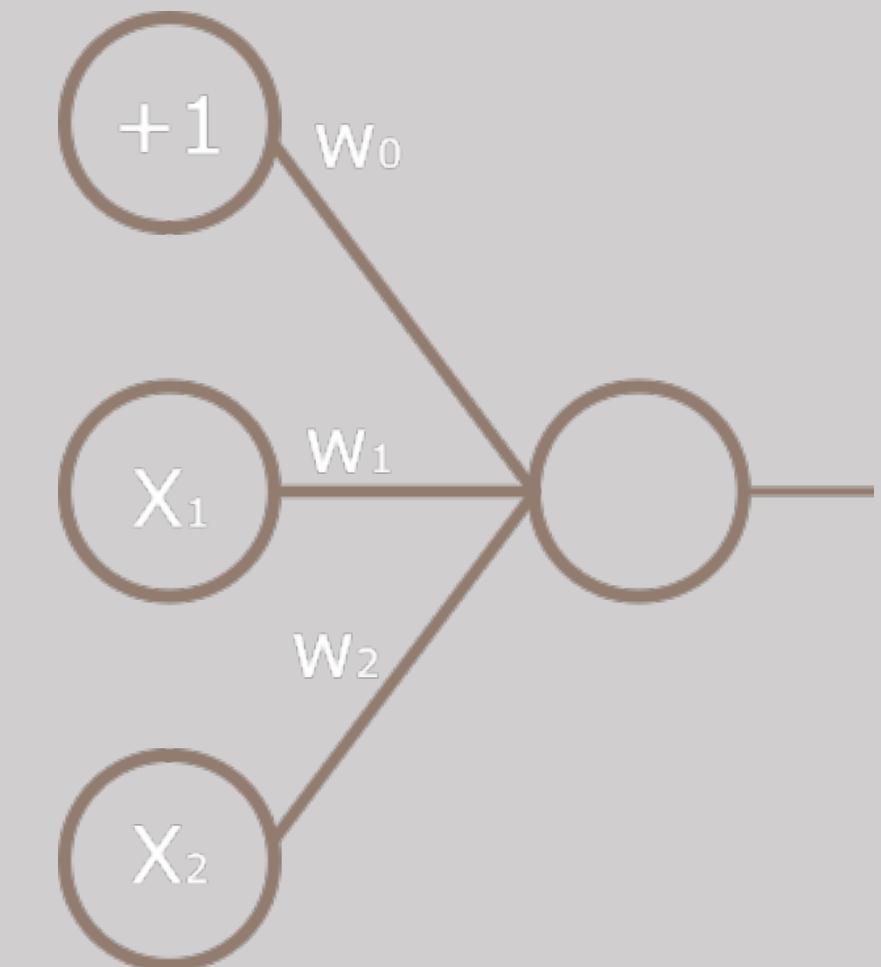
# La representación formal

- Típicamente la primera neurona toma el valor de 1 (no hay que preocuparse por esto ahora).
- La primera capa de una red neuronal corresponde a los datos ingresados (*inputs*).

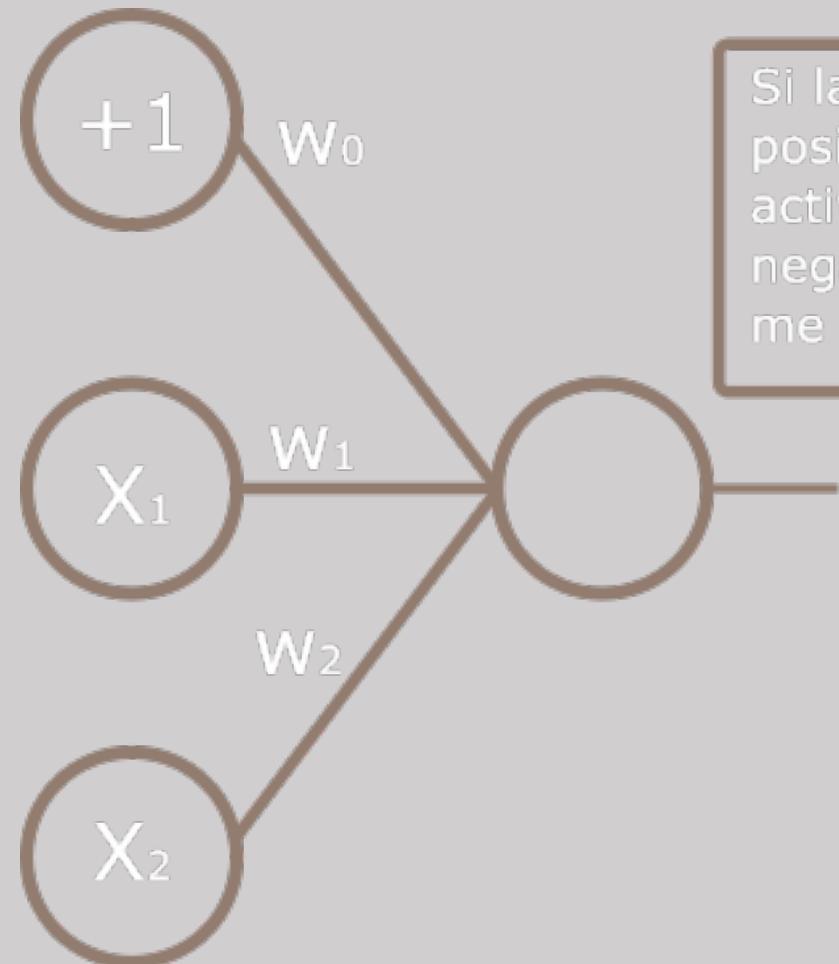


# La representación formal

- Cada conexión tiene asociado un número que representa la intensidad de esa conexión.
- Un valor más alto significa que esa neurona afecta en mayor magnitud el resultado de la neurona siguiente.



La última  
neurona se  
activa si:

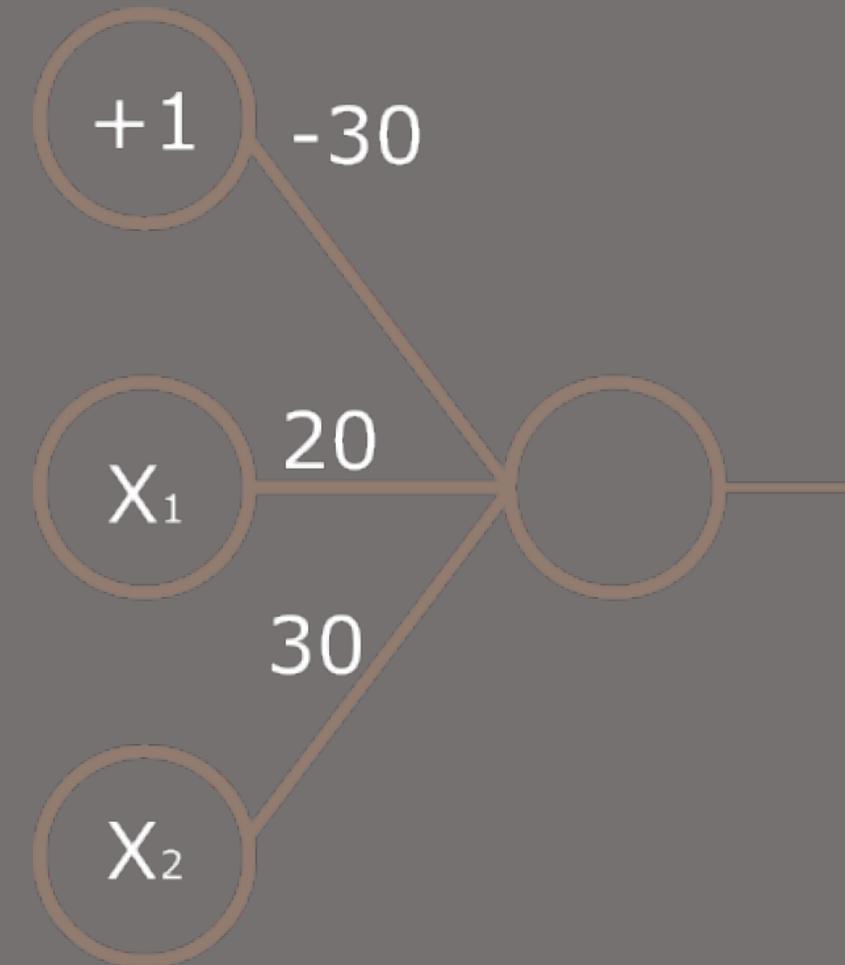


Si la suma ponderada es positiva, entonces, me activo (igual a 1), si es negativa, entonces, no me activo (igual a 0)

# Ejemplo:

¿Se activa o no se activa?

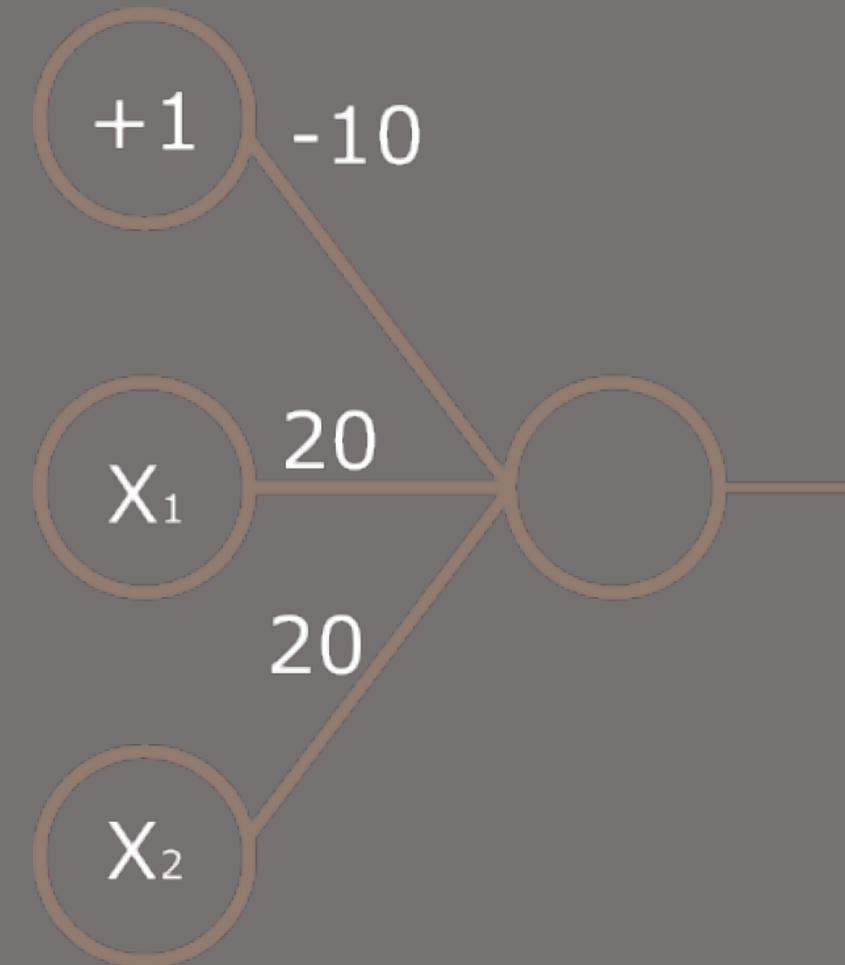
$x_1$	$x_2$
2	-1
...	...



# Ejercicio:

¿Se activa o no se activa?

$x_1$	$x_2$
1	0
...	...

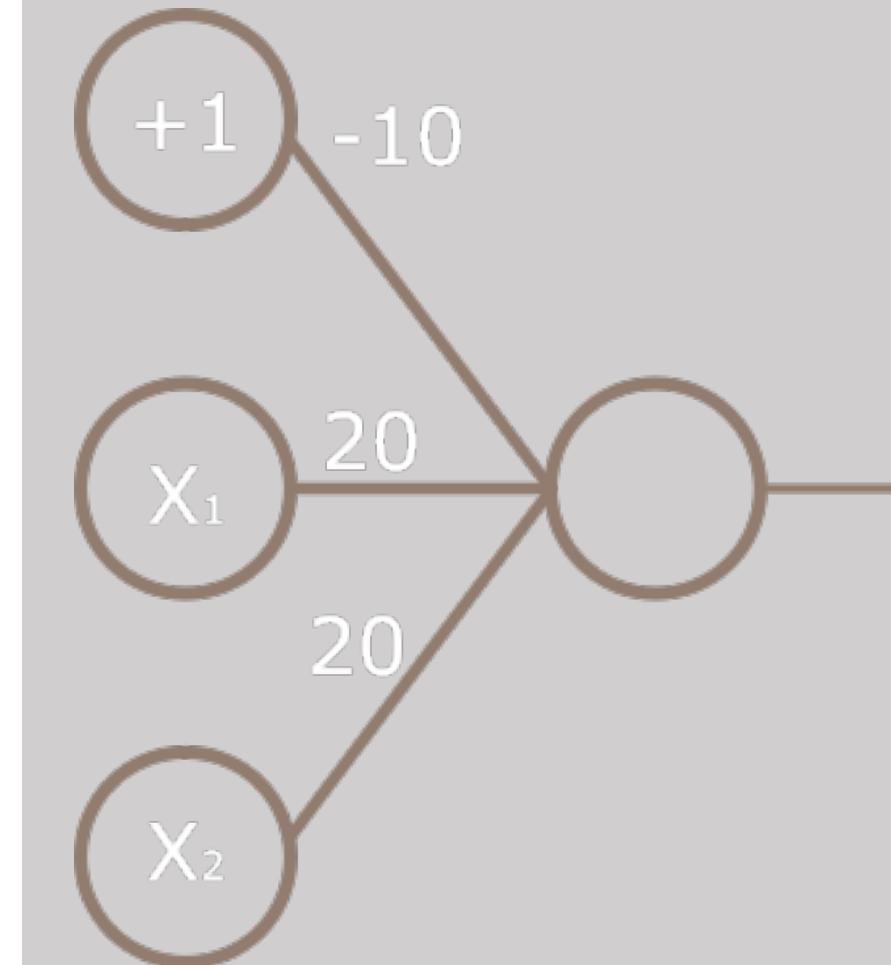


# Qué significa aprender

- Según la configuración de la red neuronal, se captura información sobre los fenómenos aprendidos.

Noten en el ejemplo anterior

X1	X2	Se Activa
1	0	Sí (10)
...	...	...

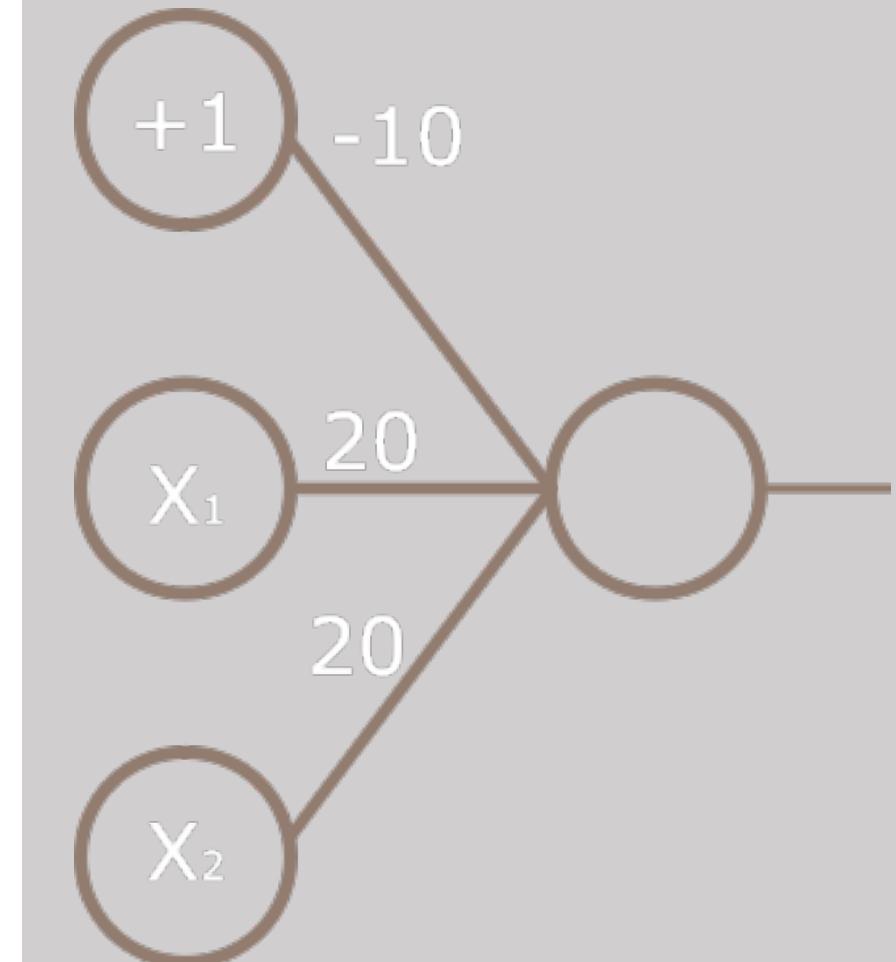


# Qué significa aprender

- Según la configuración de la red neuronal, se captura información sobre los fenómenos aprendidos.

Noten en el ejemplo anterior

X1	X2	Se Activa
1	0	Sí (10)
1	1	Sí (30)
0	1	Sí (10)
0	0	No (-10)

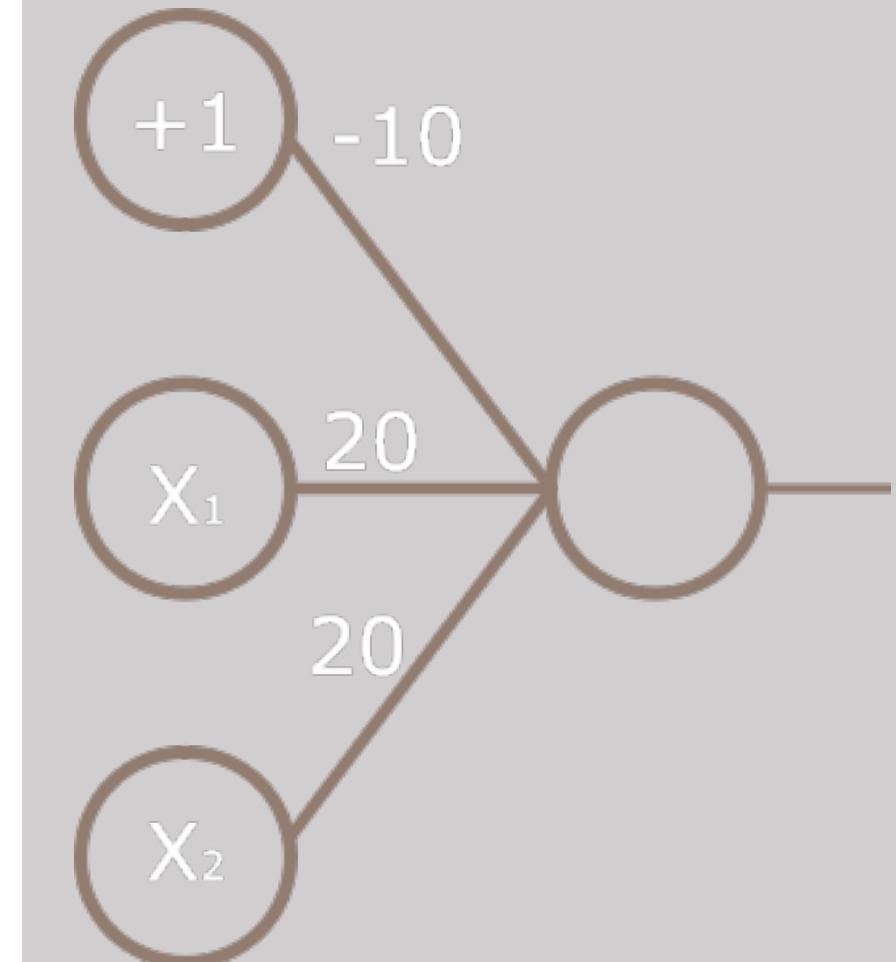


# Qué significa aprender

- Según la configuración de la red neuronal, se captura información sobre los fenómenos aprendidos.

Noten en el ejemplo anterior

X1	X2	Se Activa
1	0	Sí (10)
1	1	Sí (30)
0	1	Sí (10)
0	0	No (-10)



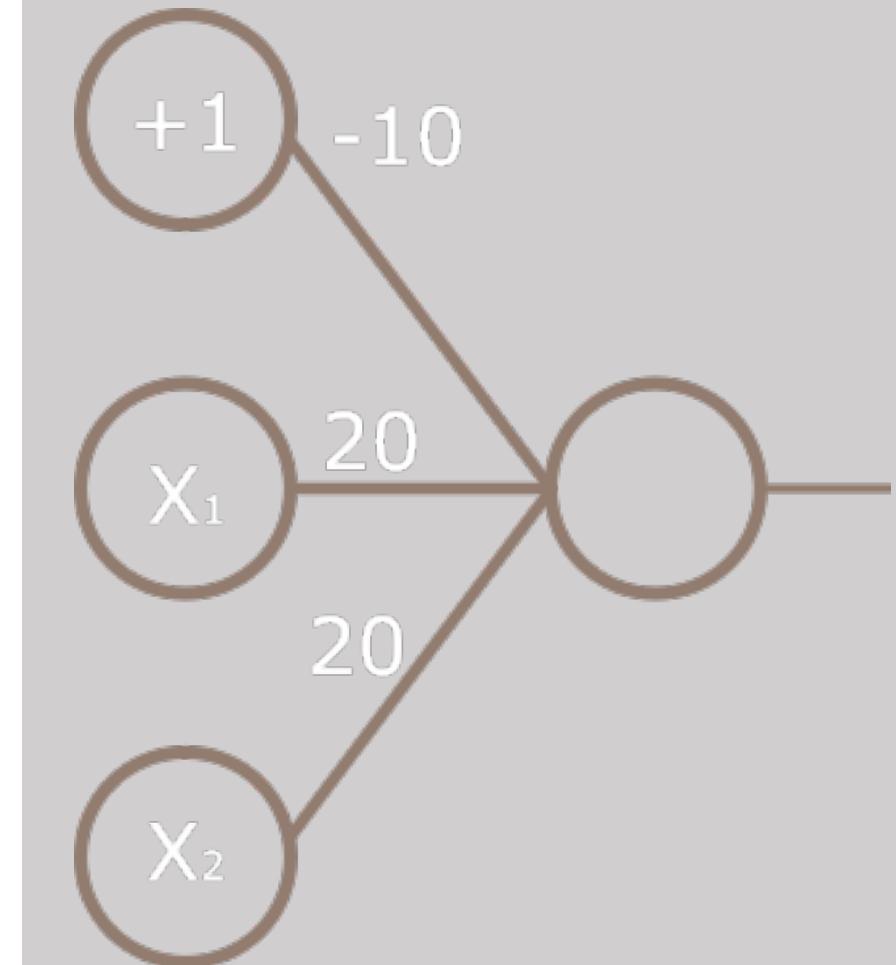
El único caso en que no se activa, es cuando los dos insumos son 0...

# Qué significa aprender

- Según la configuración de la red neuronal, se captura información sobre los fenómenos aprendidos.

Noten en el ejemplo anterior

X1	X2	Se Activa
1	0	Sí (10)
1	1	Sí (30)
0	1	Sí (10)
0	0	No (-10)

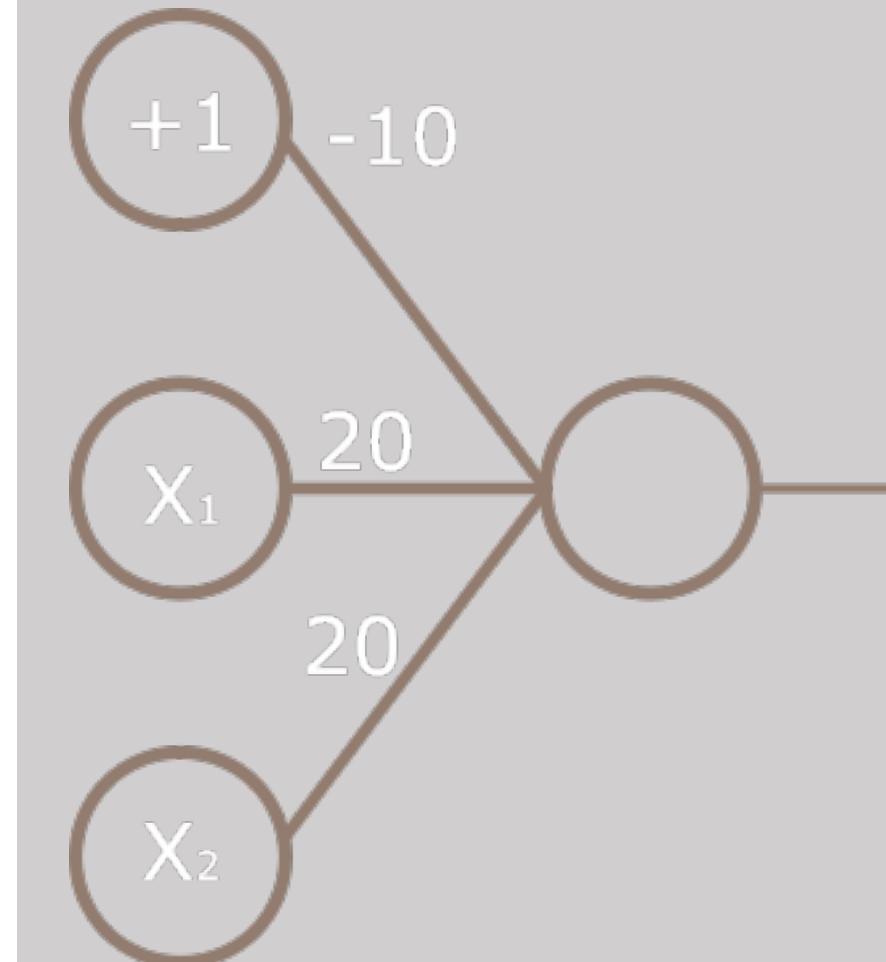


... como con una operación  
“O lógica”

# Qué significa aprender

- Esta red neuronal fue entrenada para aprender cómo simular una “O lógica”

X1	X2	Se Activa
1	0	Sí (10)
1	1	Sí (30)
0	1	Sí (10)
0	0	No (-10)

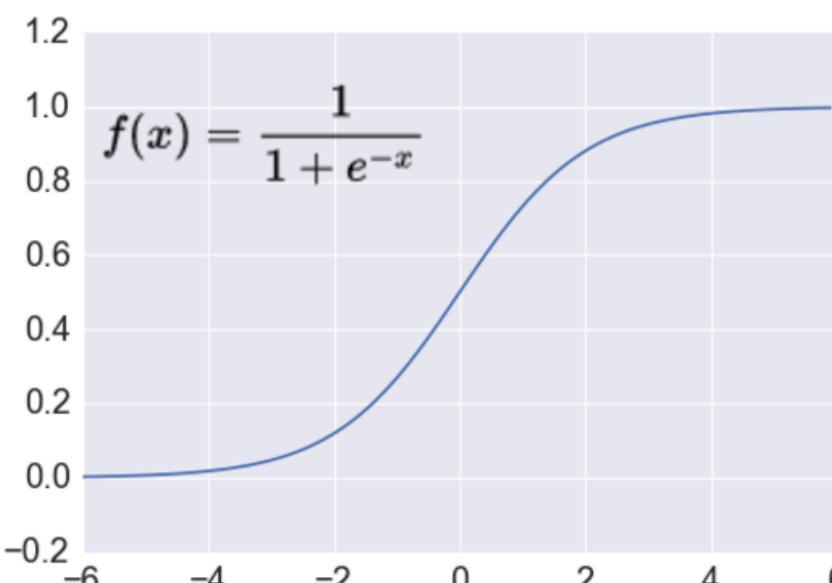


... como con una operación  
“O lógica”

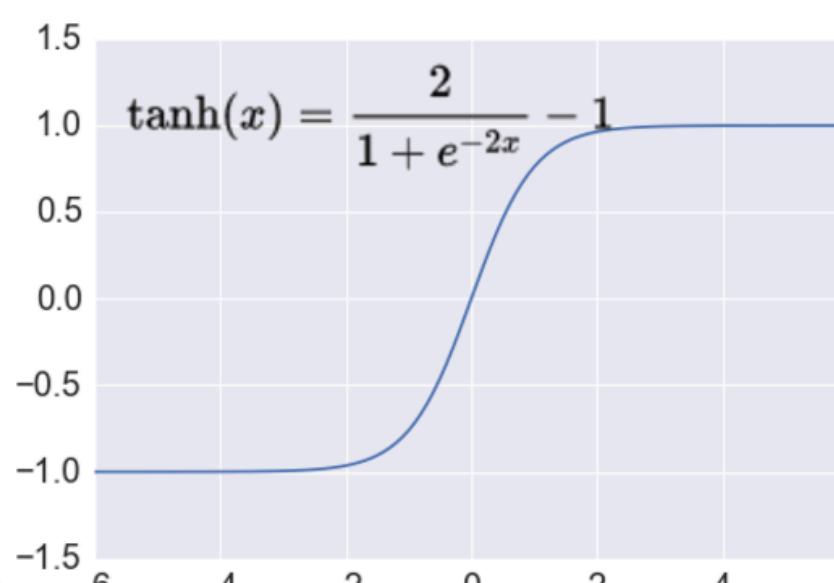
# Hay varias “funciones de activación”

- Algunas centradas en 0.. Otras no... (es útil que no tenga puntas)

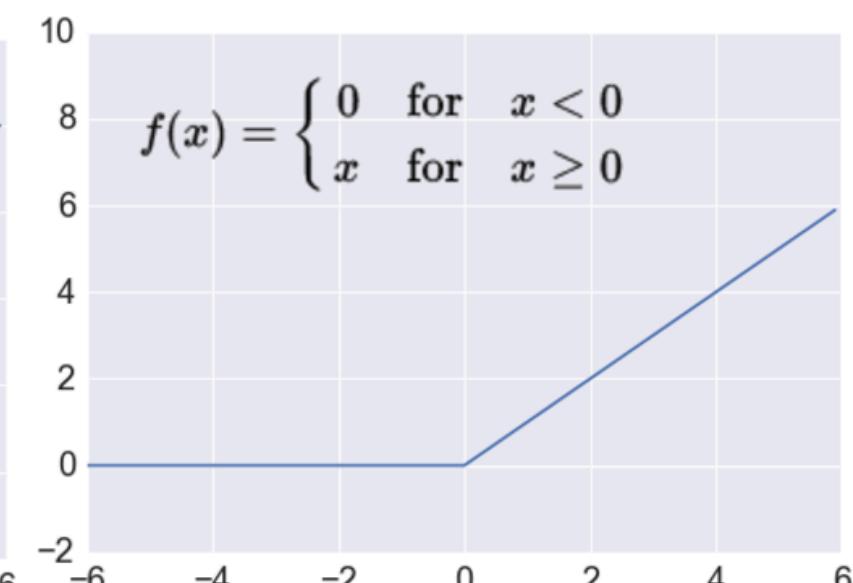
Sigmoid



TanH

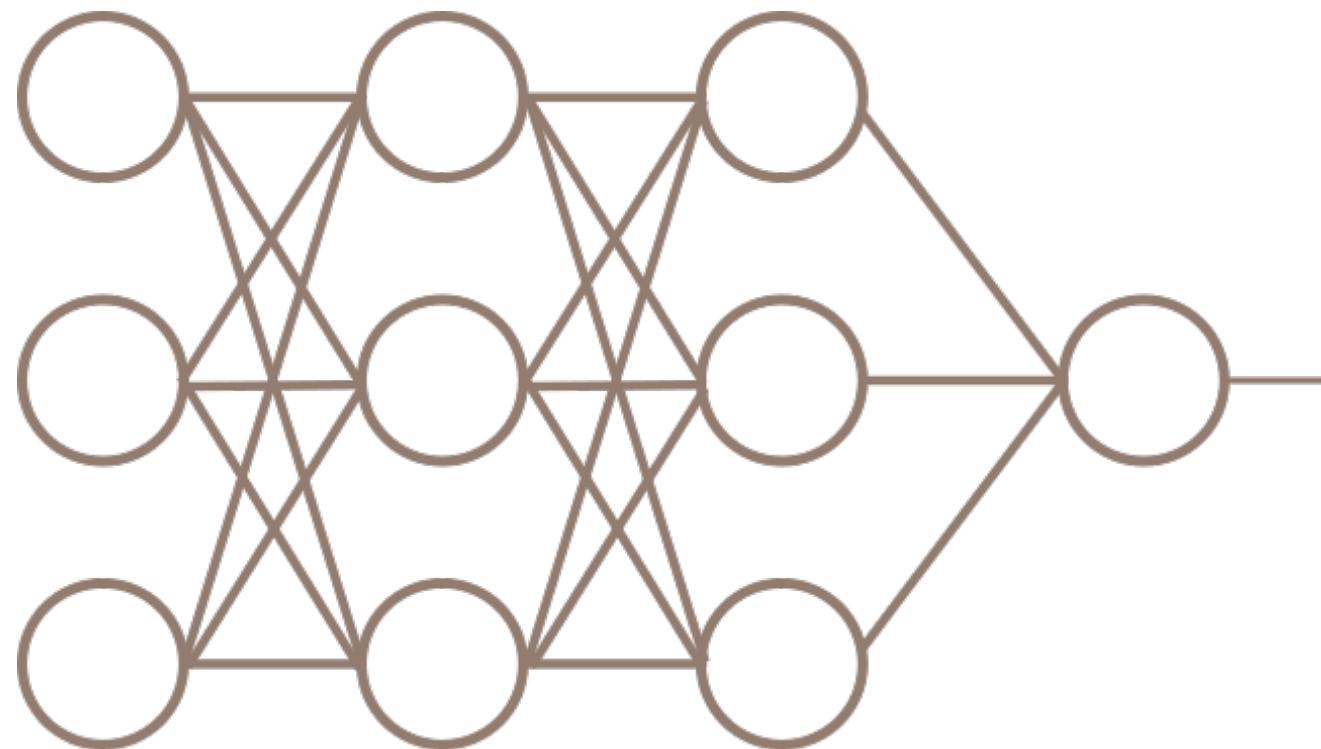


ReLU



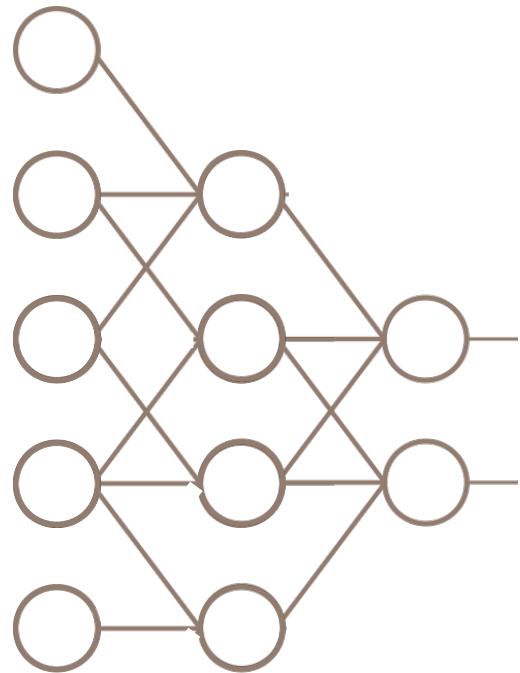
# ¡Note!

- ¿Qué significa la capa número 2? Ni idea...



# También note

- No toda red neuronal tiene que ser densa:



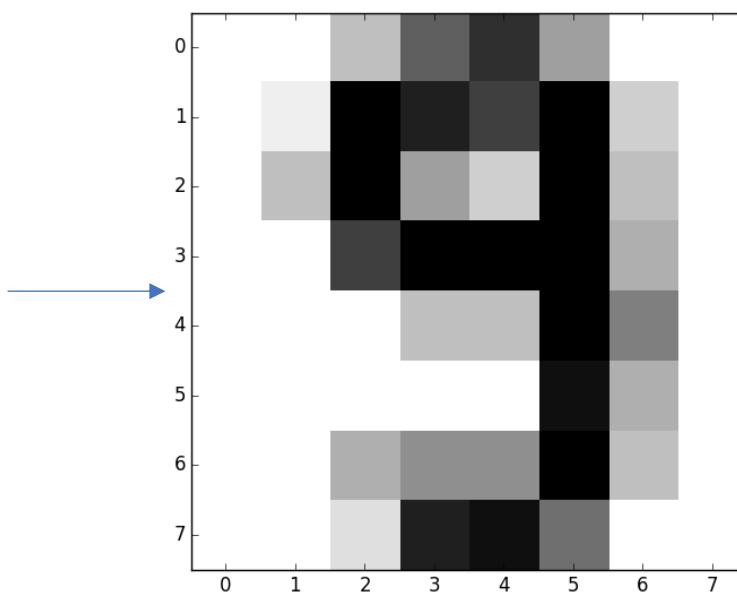
# Caso de aplicación: visión por computador



0 0 0 0 0 0 0 0 0 0  
1 1 1 1 1 1 1 1 1 1  
2 2 2 2 2 2 2 2 2 2  
3 3 3 3 3 3 3 3 3 3  
4 4 4 4 4 4 4 4 4 4  
5 5 5 5 5 5 5 5 5 5  
6 6 6 6 6 6 6 6 6 6  
7 7 7 7 7 7 7 7 7 7  
8 8 8 8 8 8 8 8 8 8  
9 9 9 9 9 9 9 9 9 9

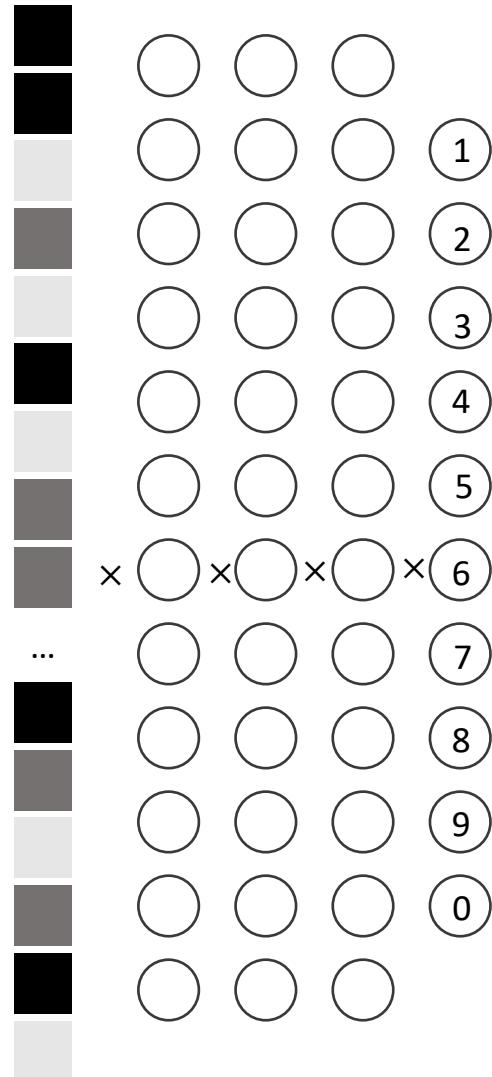
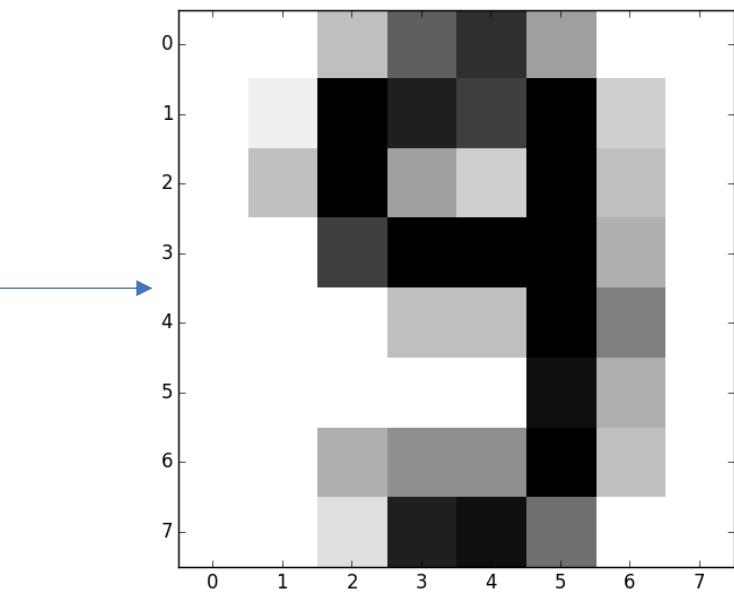
# Caso de aplicación: visión por computador

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4
5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6
7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7
8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9

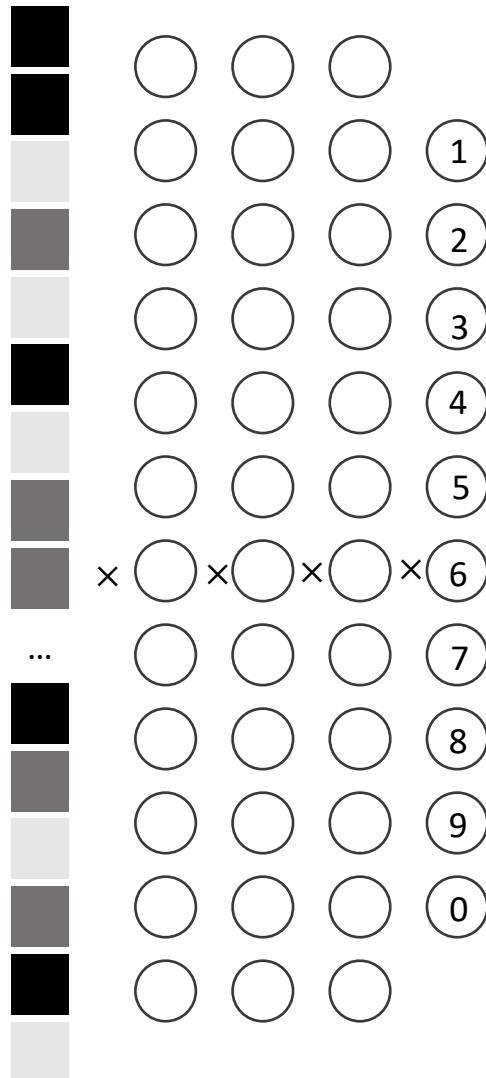
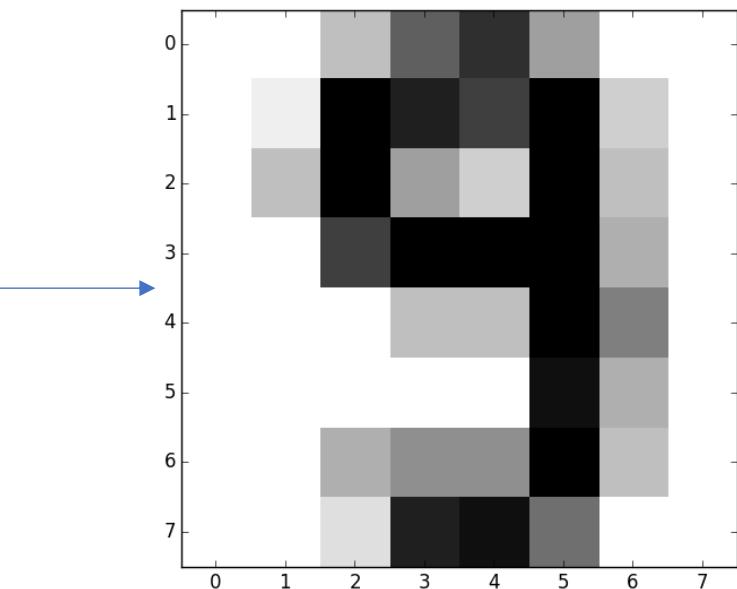


# Caso de aplicación: visión por computador

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4
5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6
7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7
8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9



# Caso de aplicación: visión por computador



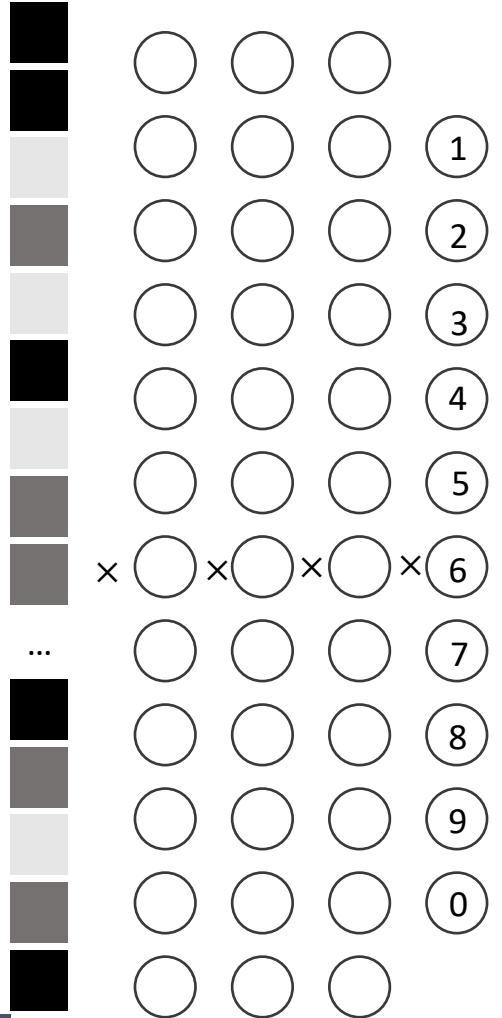
¿Cuántos parámetros hay que ajustar? ¡¡Fácilmente 64x60 sólamente en la primera capa!!

# Caso de aplicación: visión por computador

[Clasificación múltiples categorías]

Cada posible categoría se representa con un vector (algo que puede ocurrir es que nos den las probabilidades):

$$1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad 2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad \dots \quad 9 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \quad 0 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$



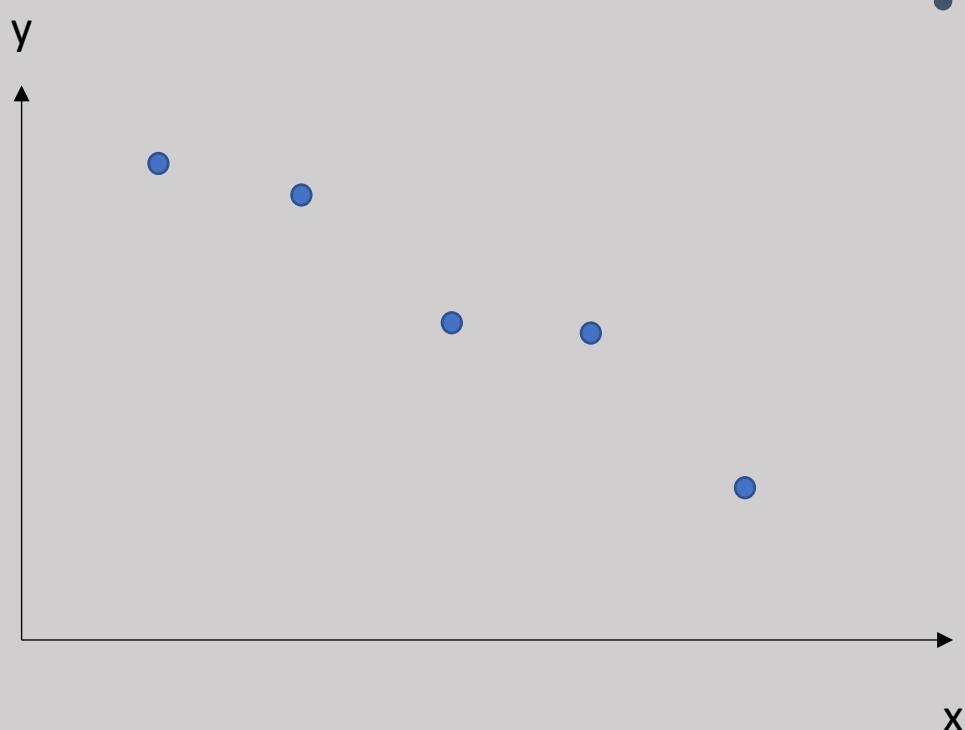
¿Cuántos parámetros hay que ajustar? ¡¡Fácilmente 64x60 sólamente en la primera capa!!

# Entrenamiento de Redes Neuronales

Algoritmos de entrenamiento

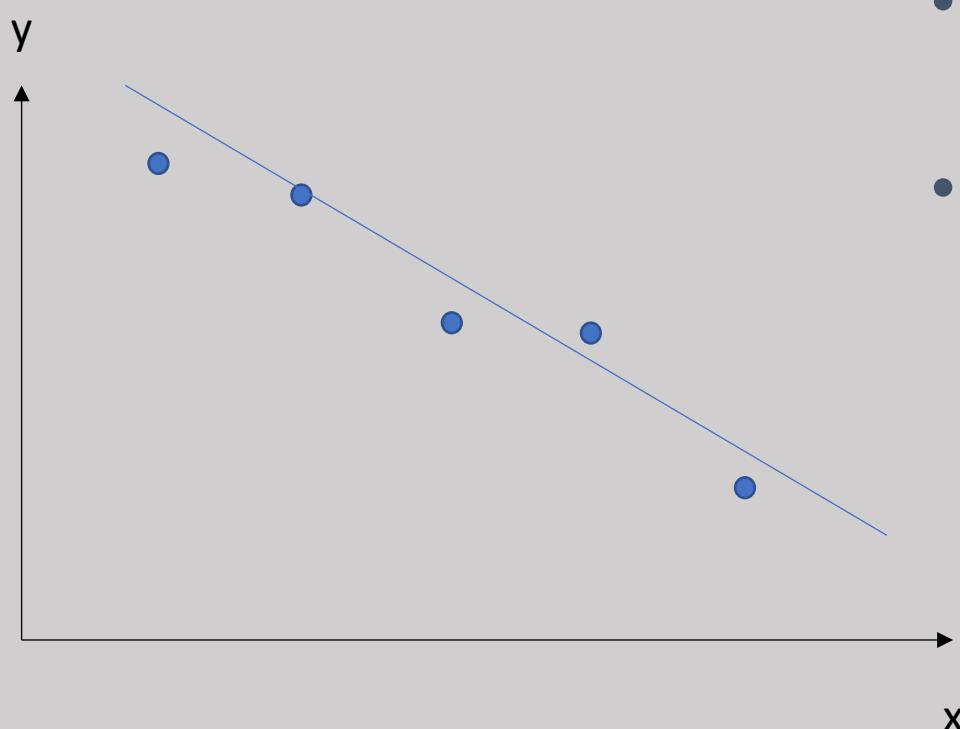


# Cómo escoger los parámetros



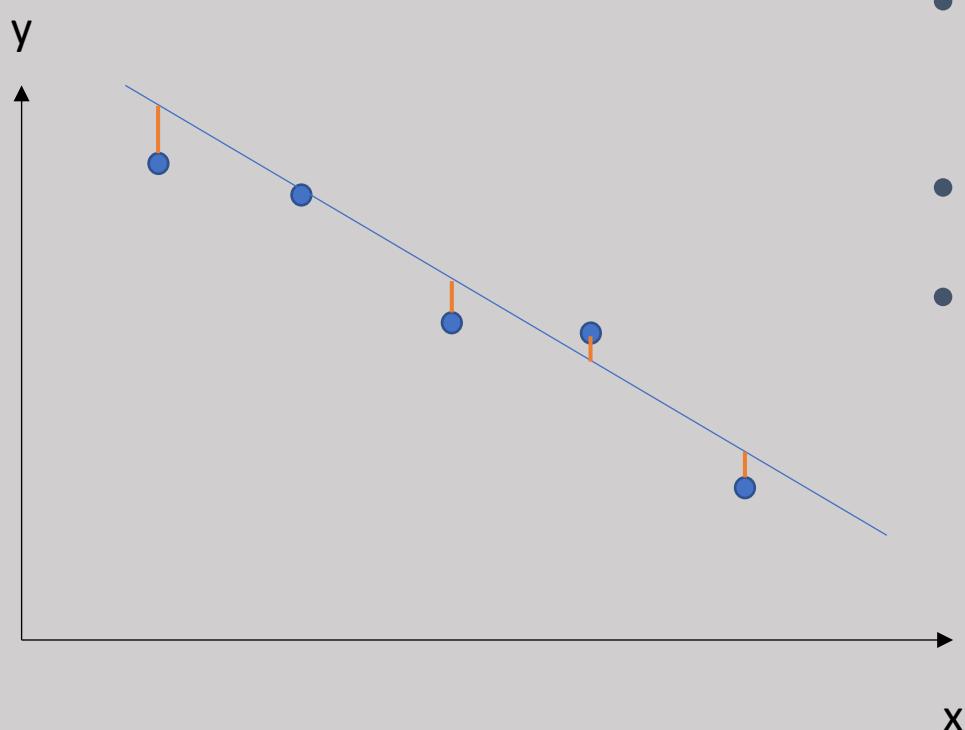
- ¿Recuerdan cuando ajustábamos líneas a modelos?

# Cómo escoger los parámetros



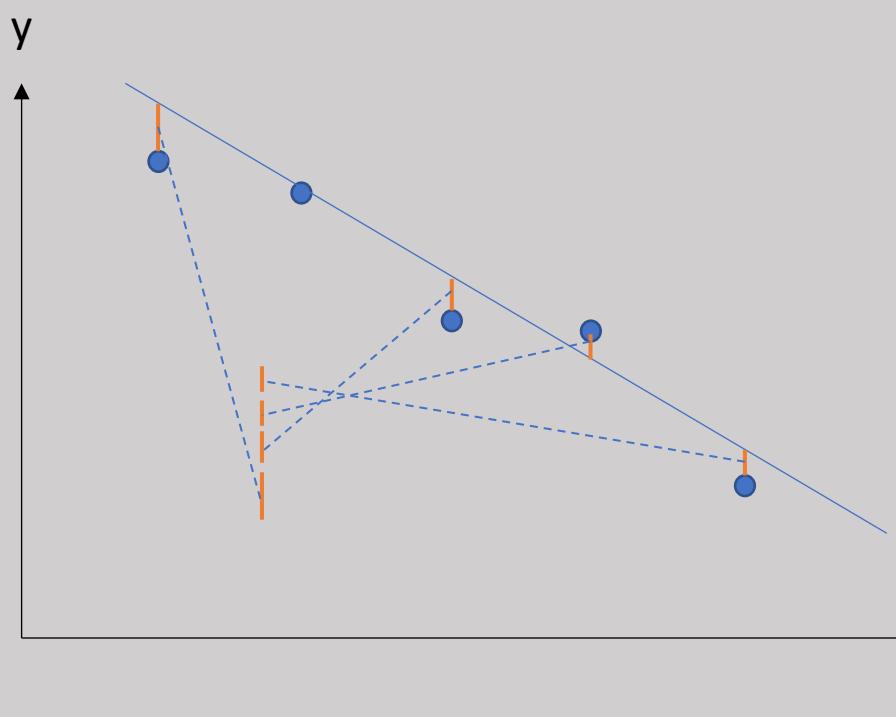
- ¿Recuerdan cuando ajustábamos líneas a modelos?
- ¿Cómo escogíamos esa línea?

# Cómo escoger los parámetros



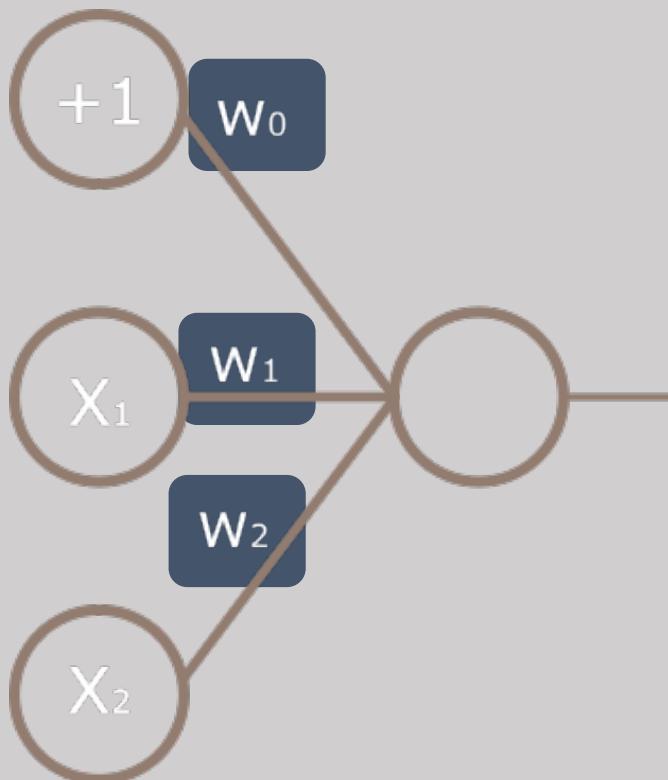
- ¿Recuerdan cuando ajustábamos líneas a modelos?
- ¿Cómo escogíamos esa línea?
- Observábamos los errores de predicción

# Cómo escoger los parámetros



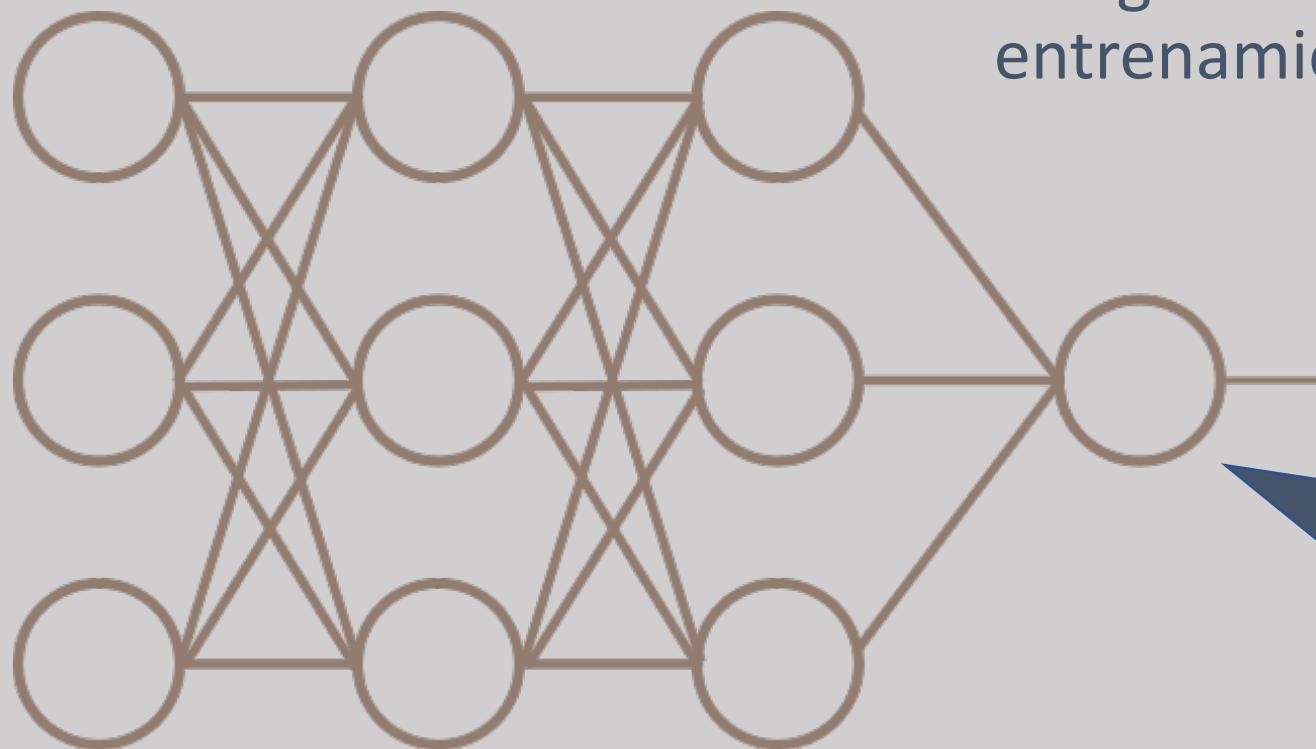
- ¿Recuerdan cuando ajustábamos líneas a modelos?
- ¿Cómo escogíamos esa línea?
- Observábamos los errores de predicción
- Y escogíamos la línea que hacía la suma de los errores (al cuadrado) la más pequeña posible

# Cómo escoger los parámetros



- Entrenar a una red neuronal significa escoger los parámetros que reducen ciertos errores al cuadrado de predicción agregados en los datos de entrenamiento.
- Es como ajustar muchas líneas en un sólo modelo.

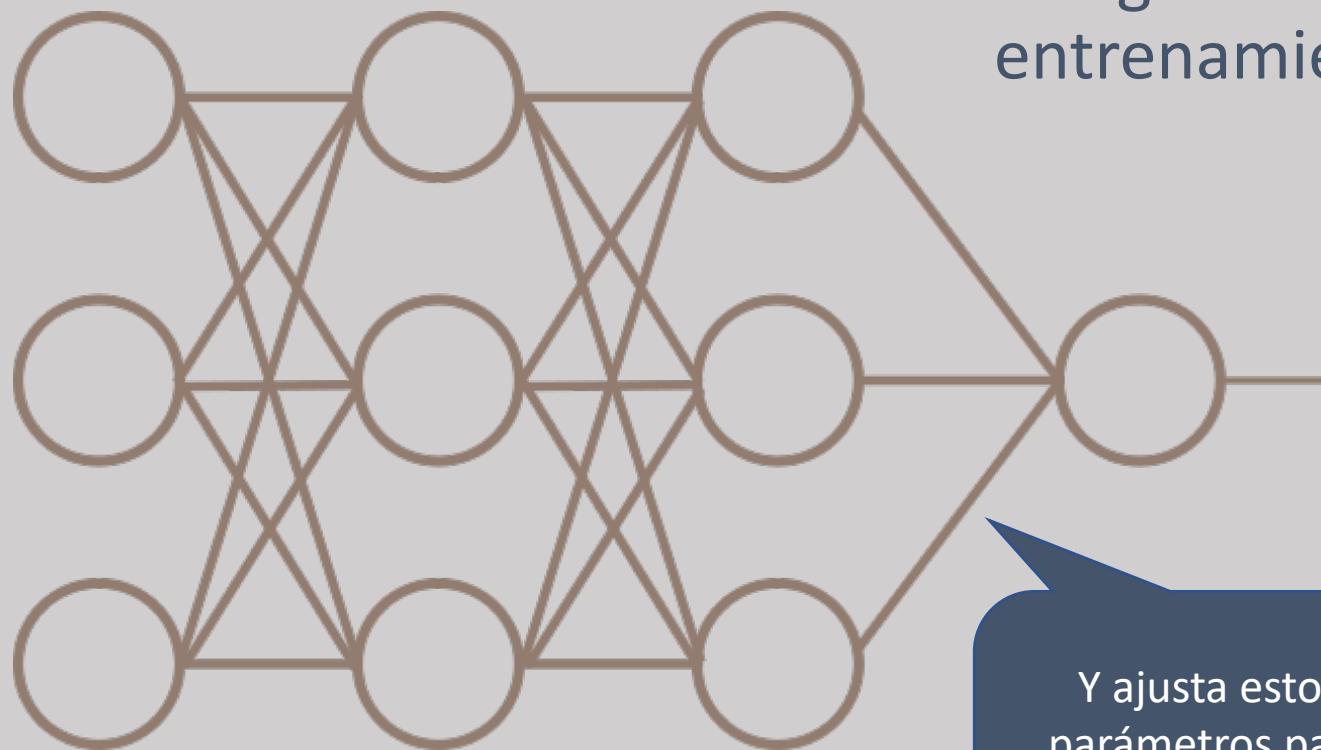
# Cómo escoger los parámetros



- El algoritmo más común de entrenamiento es back-propagation

Toma los datos finales (reales)...

# Cómo escoger los parámetros

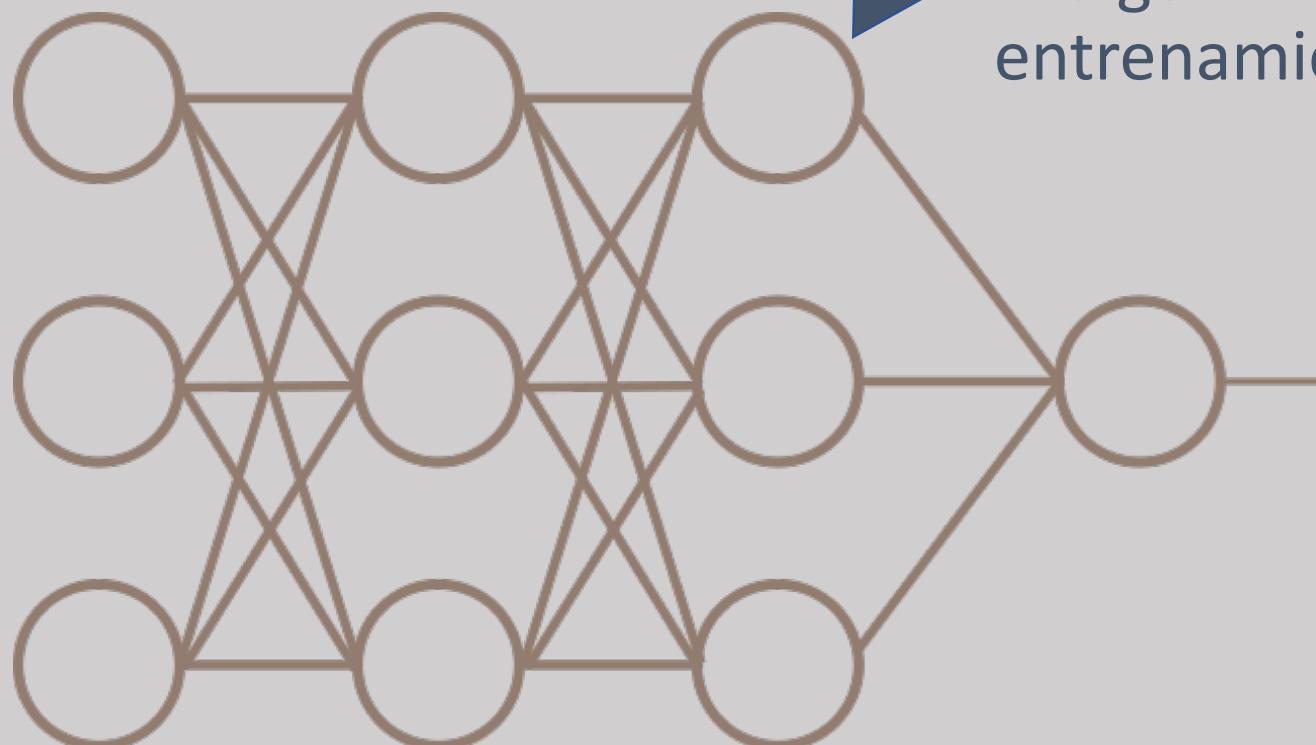


- El algoritmo más común de entrenamiento es back-propagation

Y ajusta estos parámetros para que el error sea el menor posible

# Cómo escoger los parámetros

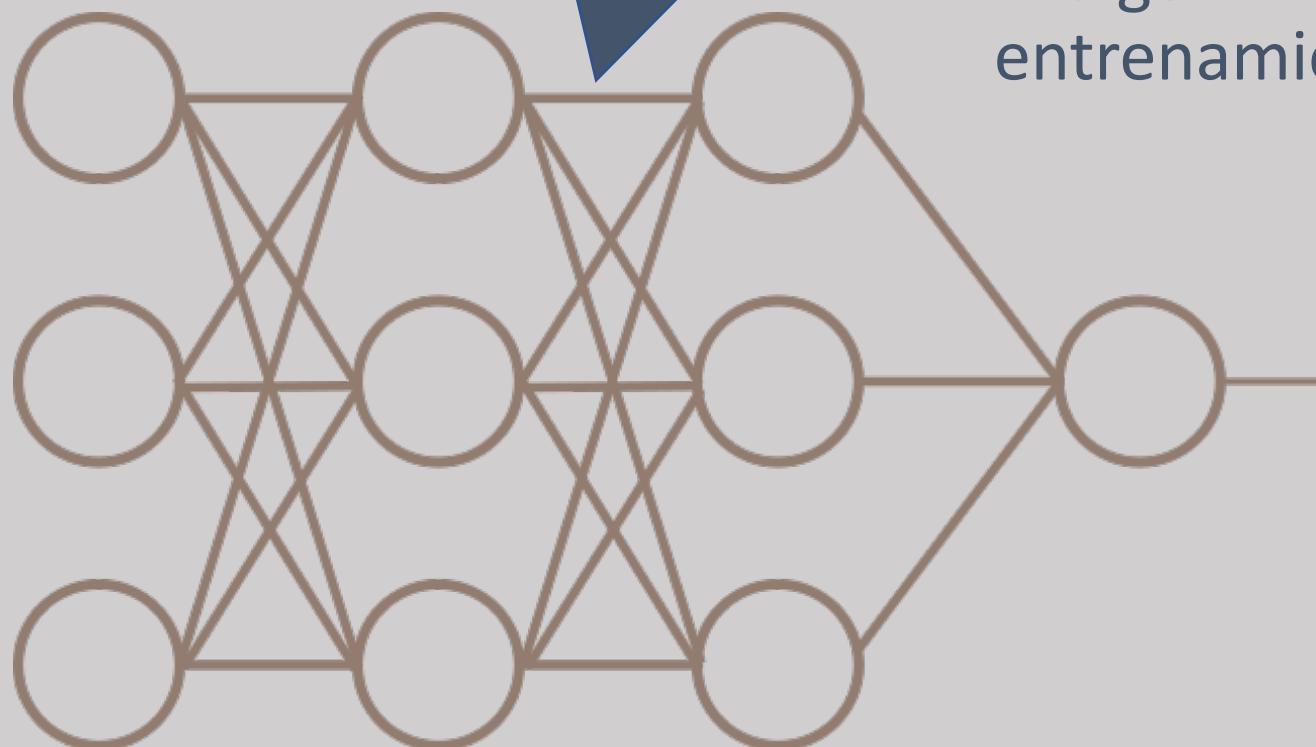
Calcula los valores  
de la capa  
anterior...



El algoritmo más común de entrenamiento es back-propagation

# Cómo escalar los parámetros

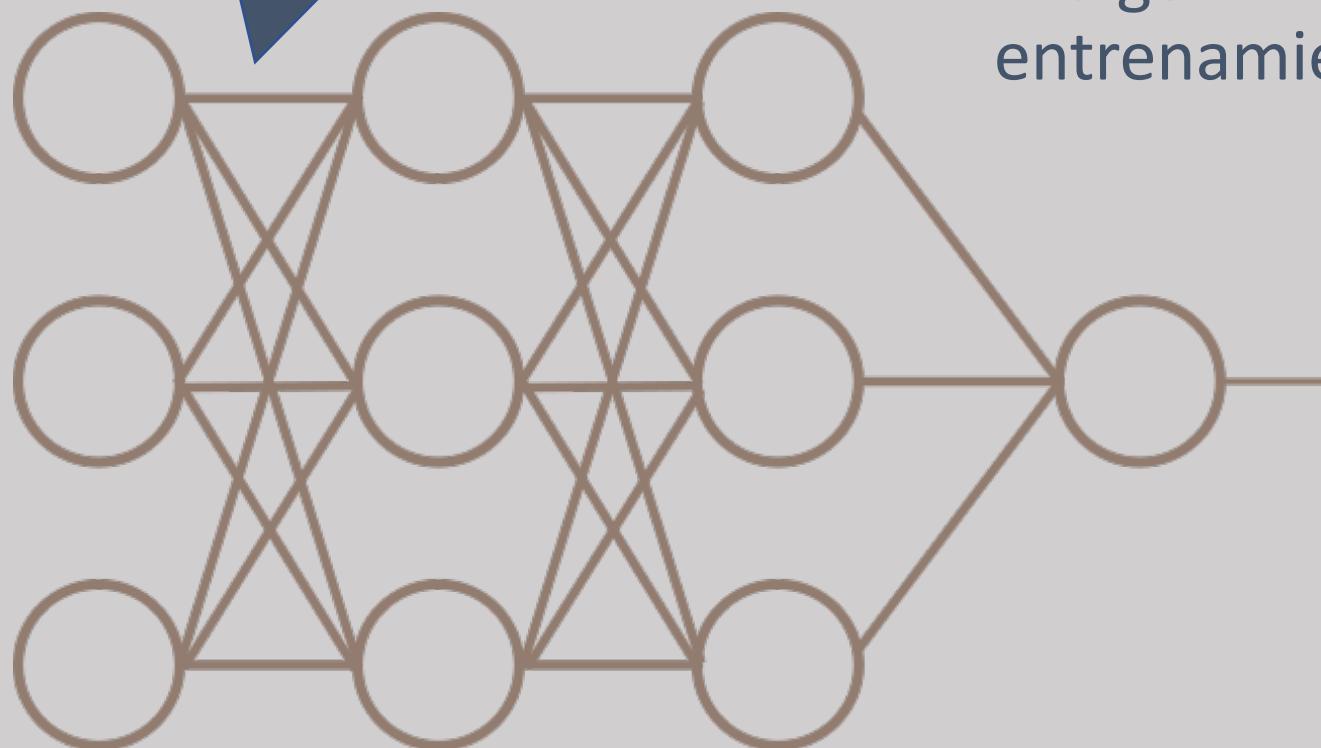
Y ajusta estos parámetros para minimizar el error



- El algoritmo más común de entrenamiento es back-propagation

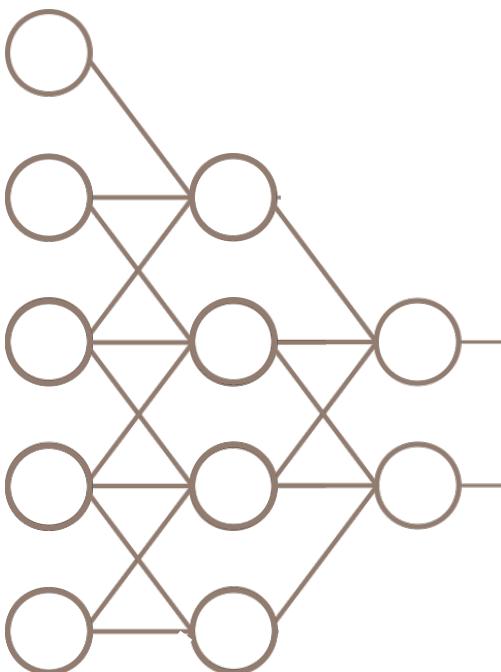
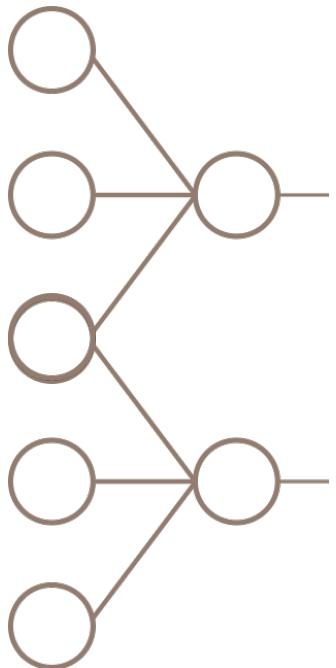
# Cómo ajustar los parámetros

Y así...



- El algoritmo más común de entrenamiento es back-propagation

# Pero ¿cómo saber qué red entrenar?



... ? ? ?

# Pero ¿cómo saber qué red entrenar?

- No existe regla general para determinar una arquitectura de red
- Lo común es comenzar con redes pequeñas y complejizar a partir de ahí
- Si tienen redes con el mismo desempeño pero una es más compleja que la otra, puede ser preferible la más simple
- Aunque existen técnicas como algoritmos evolutivos inspirados en biología de la evolución: [Marl/O](#)



# Redes Neuronales en R

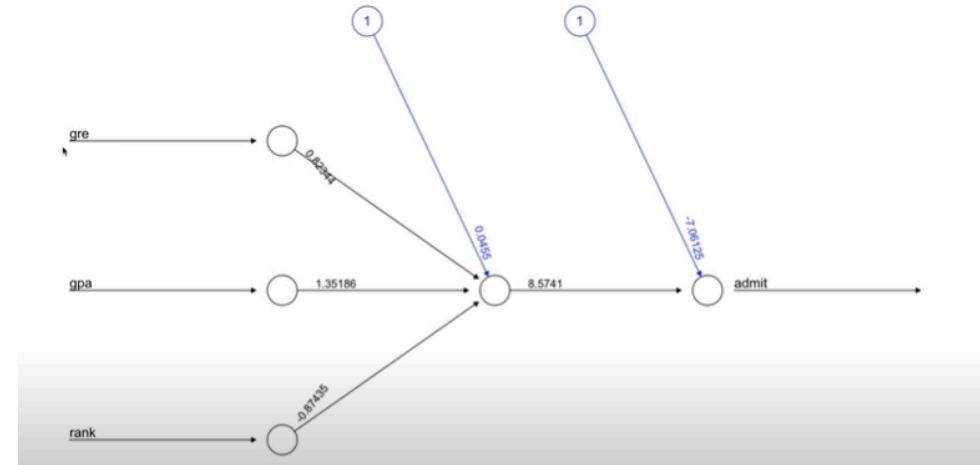
```
# Neural Networks  
library(neuralnet)
```

# Redes Neuronales en R

```
# Neural Networks  
library(neuralnet)  
n <- neuralnet(admit~gre+gpa+rank,  
                 data = training,  
                 hidden = 1,  
                 err.fct = "ce",  
                 linear.output = FALSE)
```

# Redes Neuronales en R

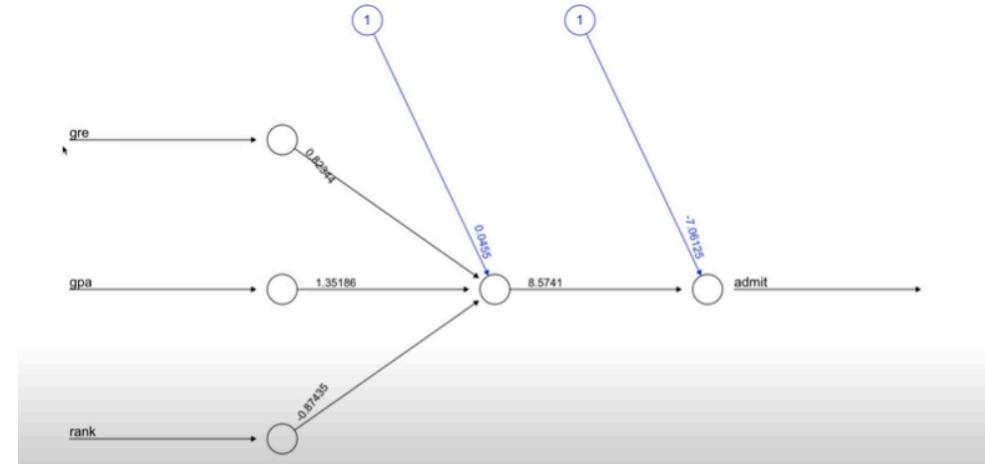
```
# Neural Networks  
library(neuralnet)  
n <- neuralnet(admit~gre+gpa+rank,  
                 data = training,  
                 hidden = 1,  
                 err.fct = "ce",  
                 linear.output = FALSE)  
  
plot(n)
```



# Redes Neuronales en R

```
# Neural Networks
library(neuralnet)
n <- neuralnet(admit~gre+gpa+rank,
                 data = training,
                 hidden = 1,
                 err.fct = "ce",
                 linear.output = FALSE)
plot(n)

# Predicción
output <- compute(n, training[,-1]) head(output$net.result) head(training[1,])
head(output$net.result) # Valores predichos (probabilidad de que se active)
head(training[1,]) # Datos originales
```



# Si queremos más capas y nodos...

```
# Neural Networks
library(neuralnet)
n <- neuralnet(admit~gre+gpa+rank,
                 data = training,
                 hidden = c(1,2),
                 err.fct = "ce",
                 linear.output = FALSE)
plot(n)
# Predicción
output <- compute(n, training[,-1]) head(output$net.result) head(training[1,])
head(output$net.result) # Valores predichos (probabilidad de que se active)
head(training[1,]) # Datos originales
```

# Una nota sobre sobre-ajuste

Si alcanzamos a ver este comentario  
sobre *overfitting*

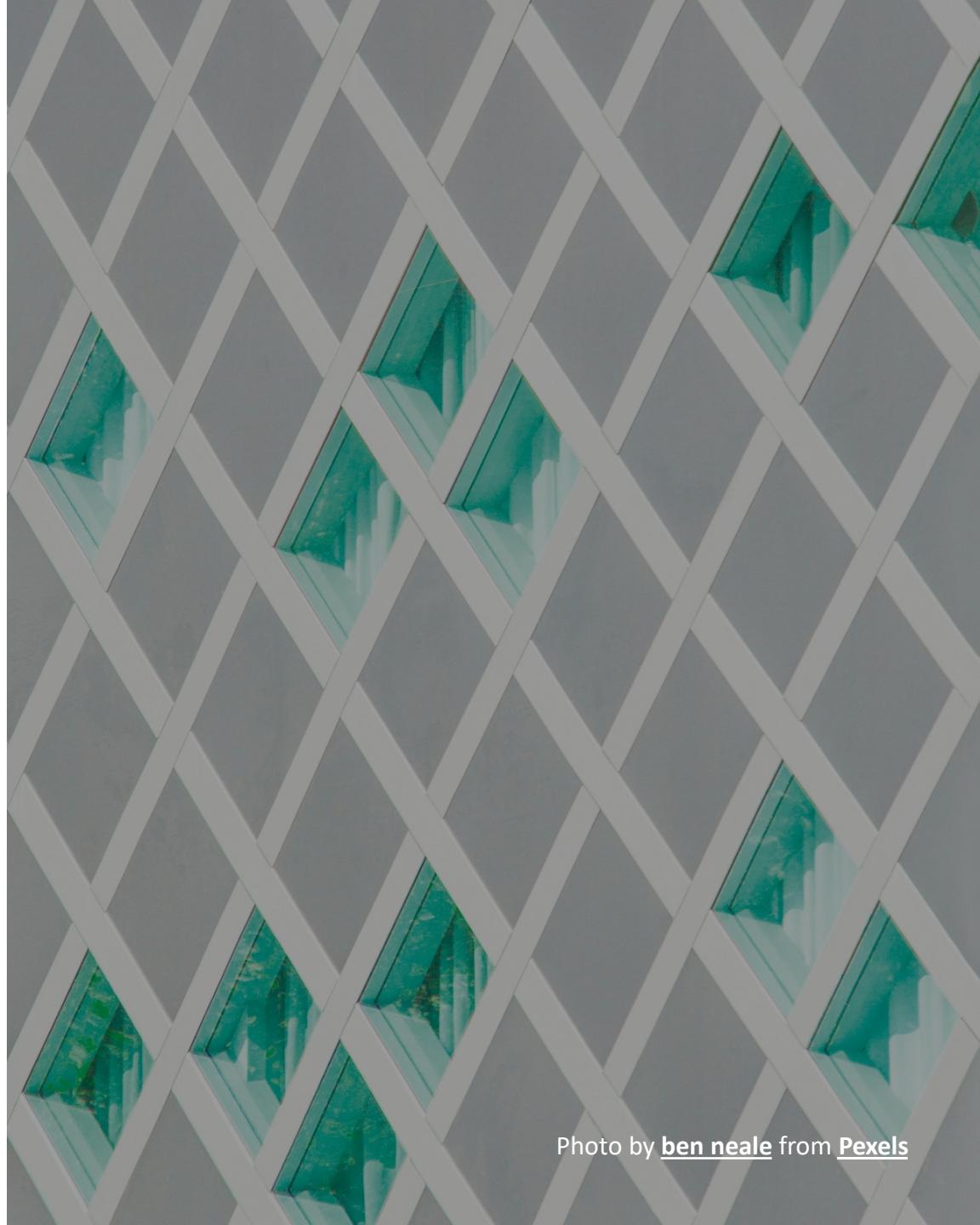
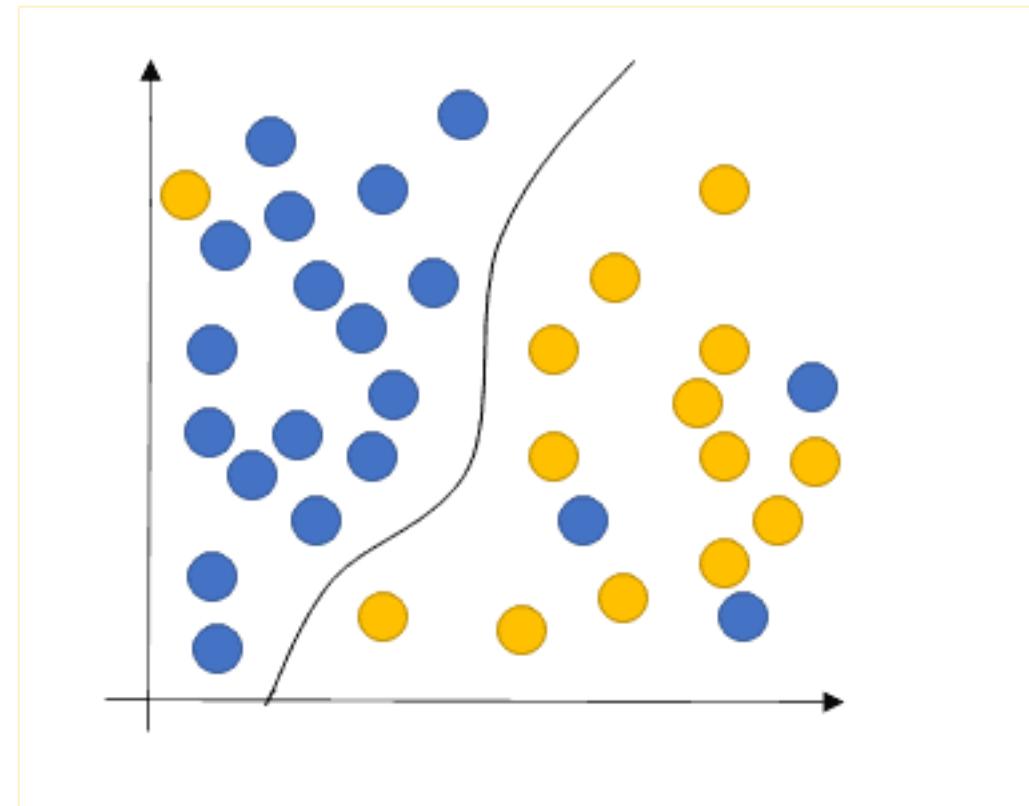


Photo by [ben neale](#) from [Pexels](#)

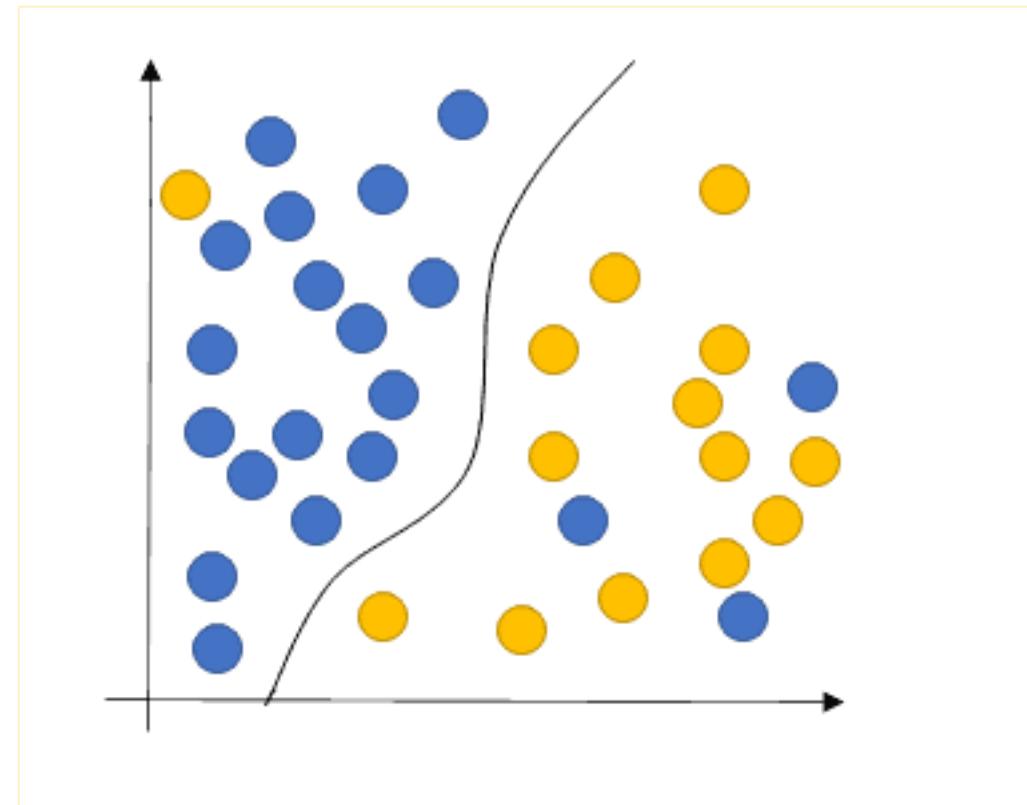
# Sobre ajuste es un problema general del ML

- Siempre hay que distribuir nuestros datos en grupos: (ej.) *train, test*
- Entrenamos nuestros modelos con los datos de entrenamiento (*train*)
- Pero igual, podríamos elaborar modelos muy muy complejos para predecir a la perfección



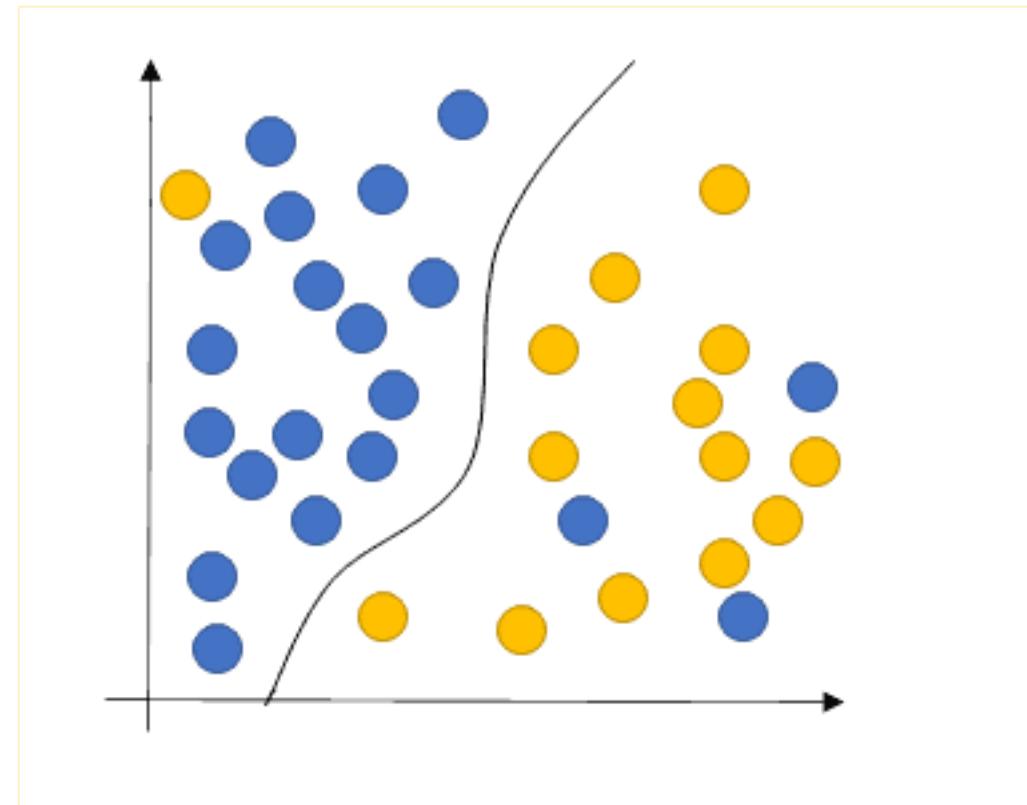
# Sobre ajuste es un problema general del ML

- Cuando ajustamos un modelo en exceso, su error de predicción en datos conocidos es muy bajo.
- Pero cuando se ponen datos nuevos, falla mucho.  
¡Y recuerden que entrenamos estas cosas para usarlas en datos nuevos!



# Sobre ajuste es un problema general del ML

- Regla general: para validar por sobreajuste se puede predecir sobre los datos de prueba (*test*).
- Un truco cool de redes neuronales: en ocasiones eliminar aleatoriamente conexiones (asignarles valor de 0) mejora la generalización ¿Por qué?



# Siguiente clase

- ¡Taller de código desde la primera hora! (todos/as en sus compus)
  1. Replicar código de **analítica prescriptiva**: el caso de fútbol
  2. Implementar **redes neuronales** y jugaremos con múltiples variaciones de arquitectura
- **Recuerden que:** retroalimentación está en el repositorio, cuadremos para vernos los otros dos grupos (10 minutos)

