



A fluorescence micrograph showing a complex, three-dimensional network of microtubules. The microtubules are visualized as thin, glowing lines in various colors including yellow, green, blue, and cyan, creating a dense, interconnected web against a dark background.

REDES NEURONALES ARTIFICIALES

2017 Nikon Photomicrography Competition

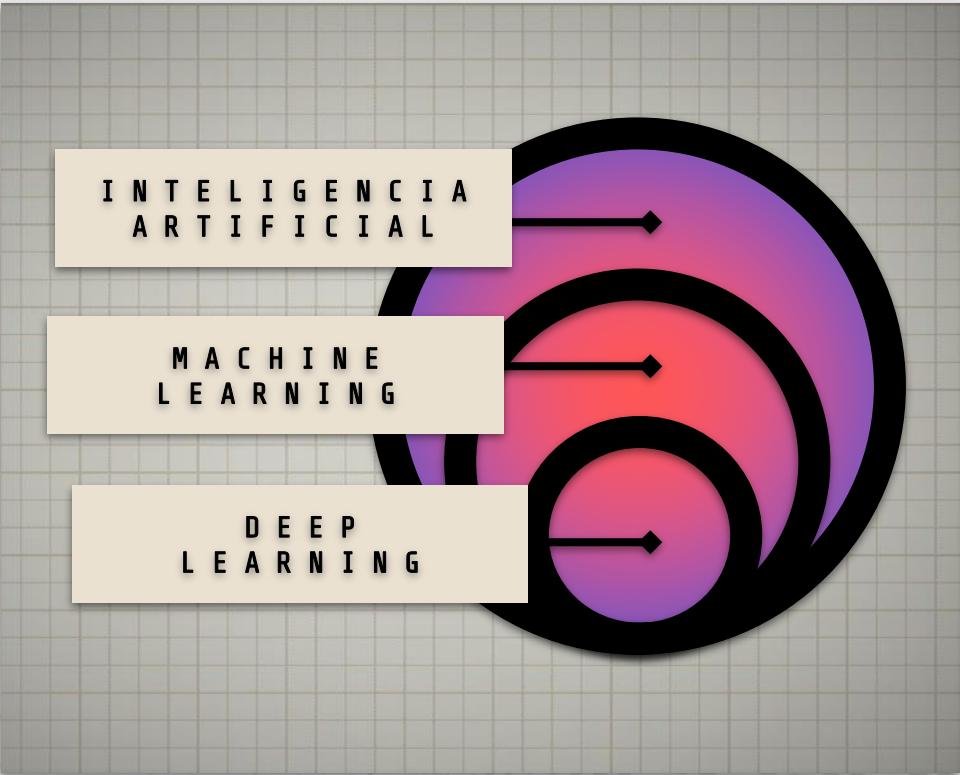
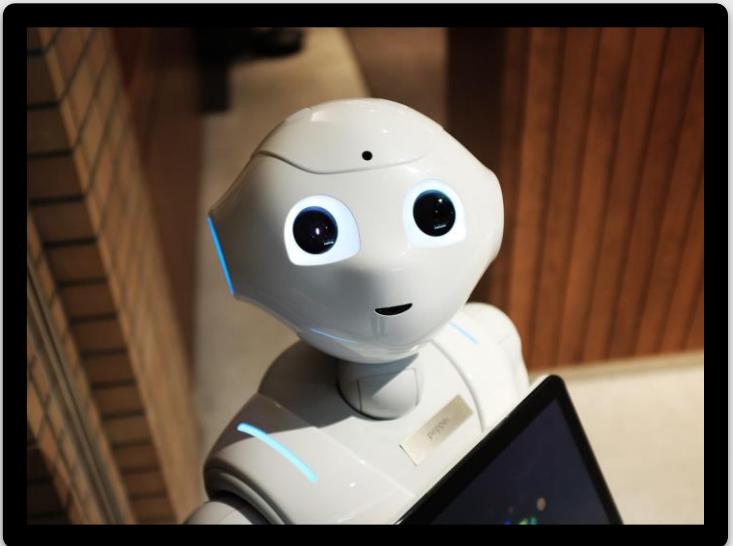
Zhen Zhang

National University of Singapore, Mechanobiology Institute

Arquitectura 3D a nanoscala de una red de microtúbulos.

Inteligencia Artificial

La **inteligencia artificial** busca la creación de máquinas que puedan imitar comportamientos inteligentes.



Machine Learning



Pepper
Laboratorio de IA
de SoftBank Robotics

video de SoftBank Robotics Europe

El **machine learning** (o aprendizaje automatizado) se dedica a desarrollar algoritmos que doten a las máquinas de la **capacidad de aprendizaje**.





video de SoftBank Robotics Europe



Aprendizaje supervisado

El modelo debe encontrar cuál es la relación entre unas variables de **entrada** y unas variables de **salida**.

RASGOS



ETIQUETAS

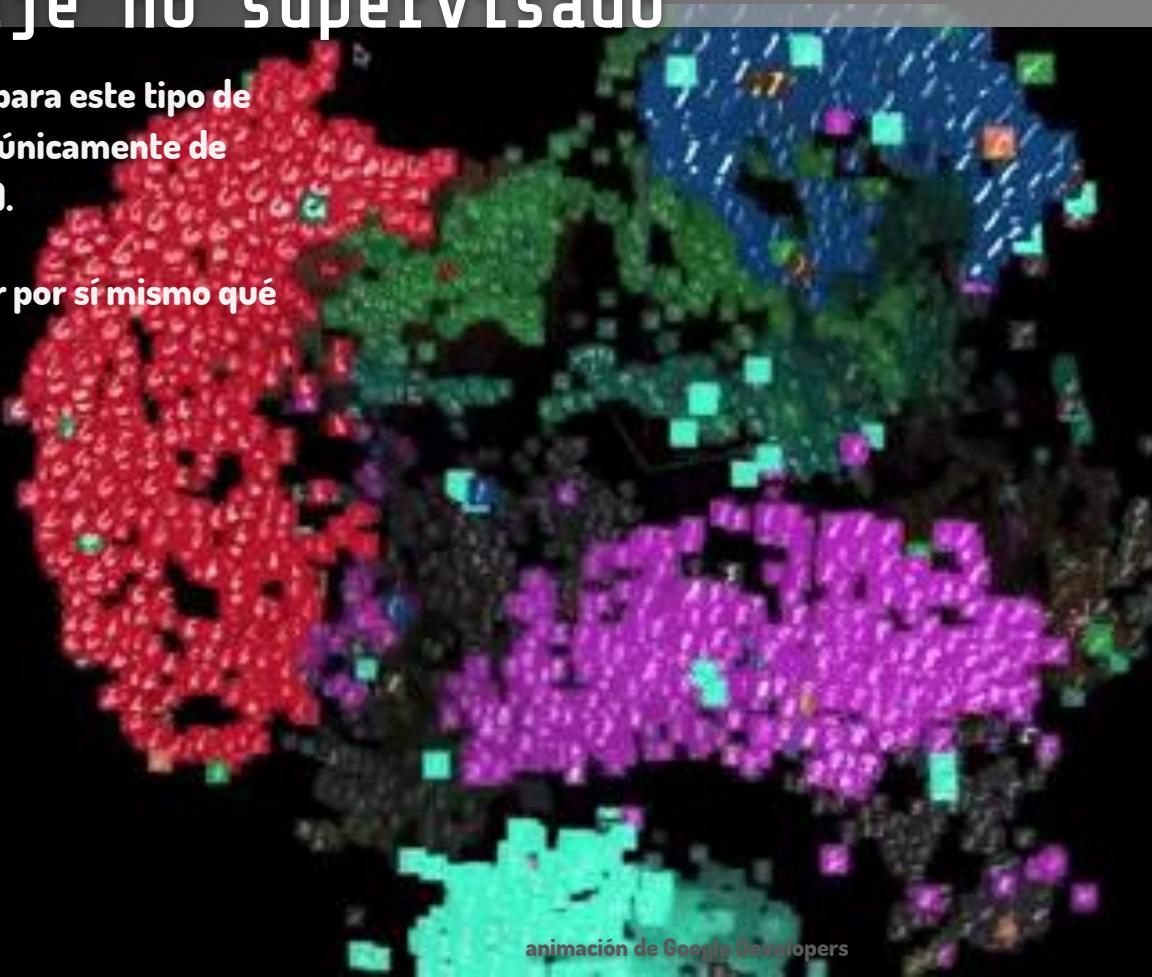


Proceso en el que se **estima** la etiqueta correspondiente a un conjunto de rasgos.

Aprendizaje no supervisado

El conjunto de datos para este tipo de aprendizaje consiste únicamente de rasgos (sin etiquetas).

El modelo debe hallar por sí mismo qué hacer con los datos.



Aprendizaje reforzado

Con este tipo de aprendizaje, el modelo interactúa con sus alrededores y aprende a partir de los estímulos que recibe de su entorno.



La supercomputadora Deep Blue vence al campeón mundial de ajedrez Garry Kasparov. (1997)



Google AlphaGo derrota al mejor jugador de Go del mundo Lee Sedol. (2016)



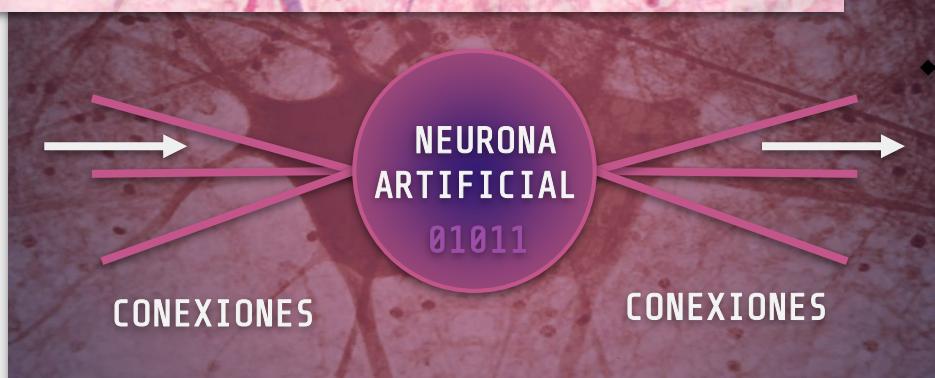
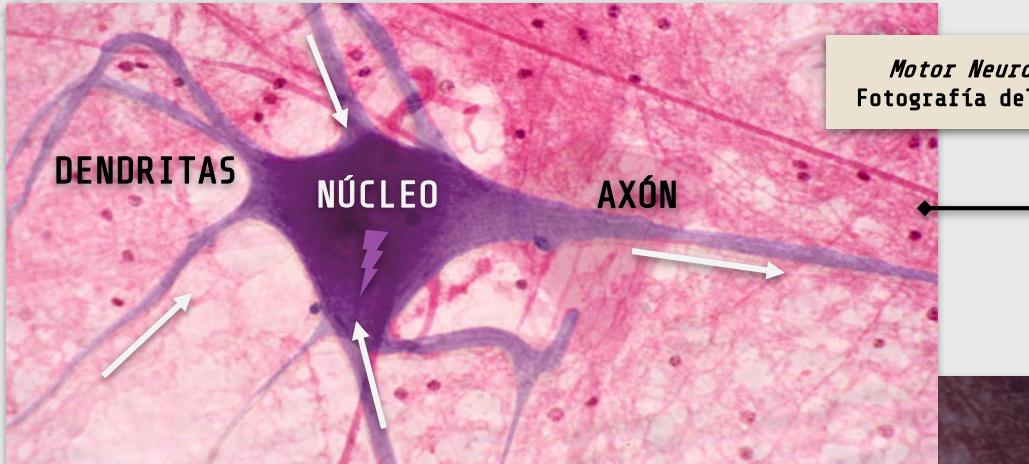
2017 Nikon Photomicrography Competition

Dr. Rodrigo Madeiro Costa, Stem Cell Laboratory

Neuronas sensoriales humanas derivadas de células iPS

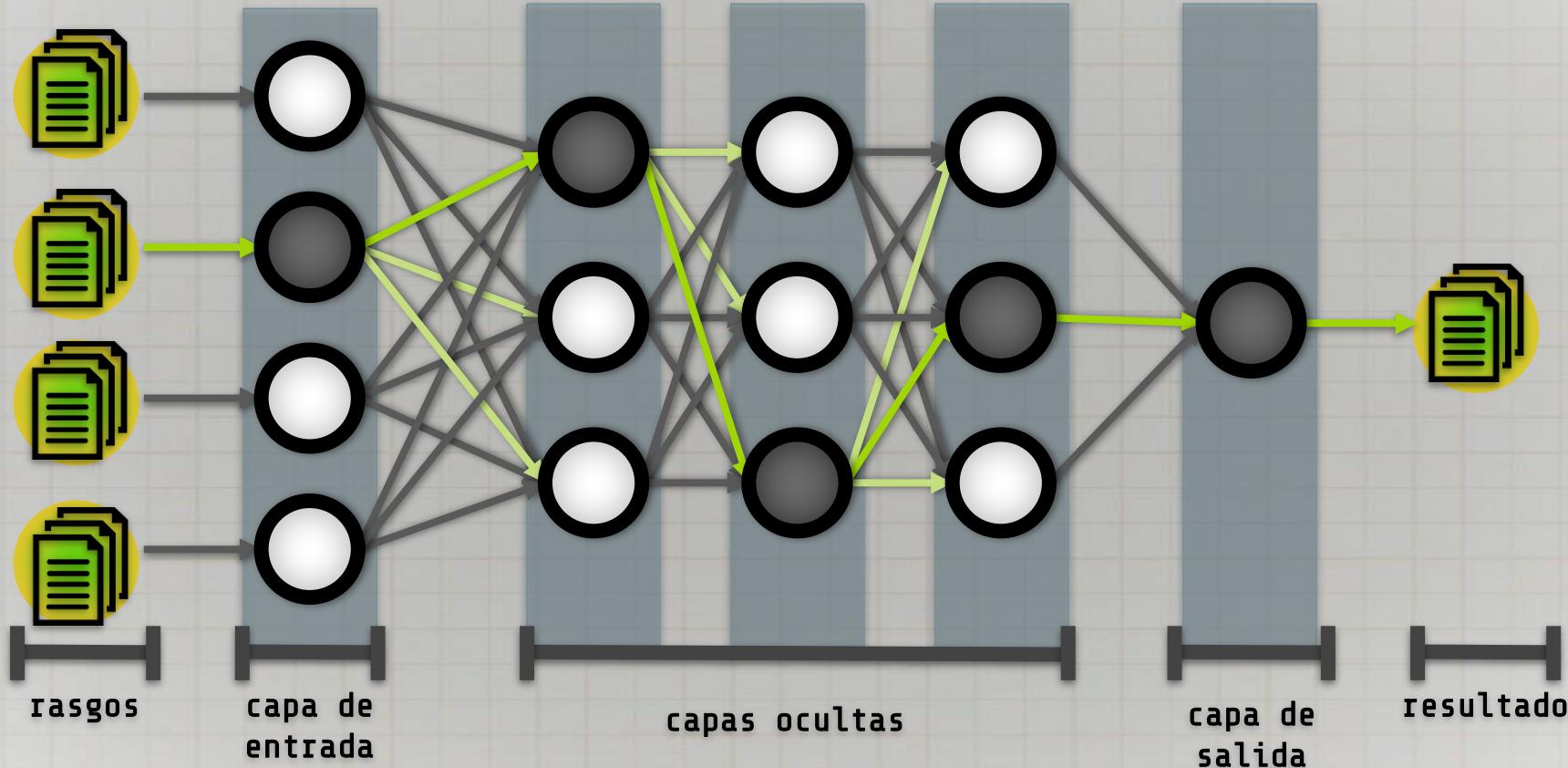
DEFINICIÓN

Neuronas biológicas vs artificiales



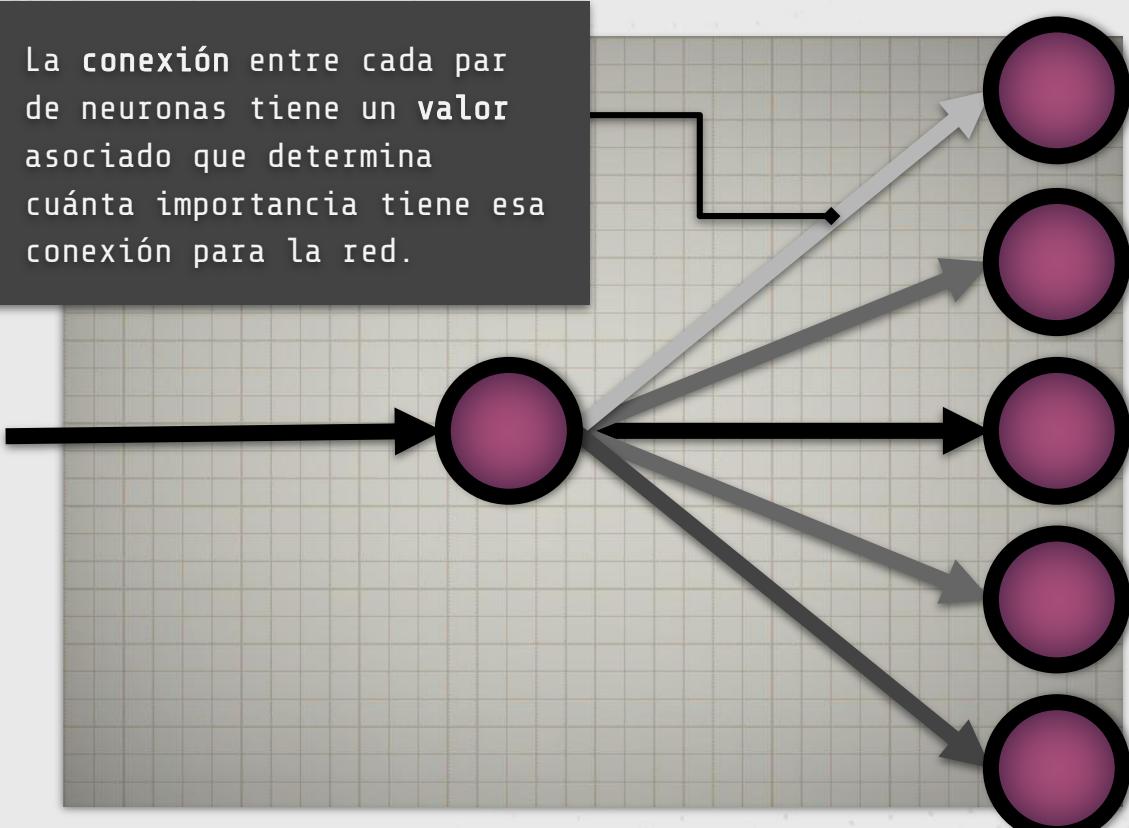
- Señales eléctricas se transmiten a través de las dendritas hacia el **núcleo** de la neurona donde serán procesadas, y la señal resultante se envía a través del **axón** hacia otras neuronas.
- Información (generalmente valores numéricos) se transmite a través de **conexiones**. Dentro de la **neurona artificial** procesa esa información y el resultado se transmite a otras neuronas artificiales por medio de **otras conexiones**.

Estructura



Conexiones neuronales

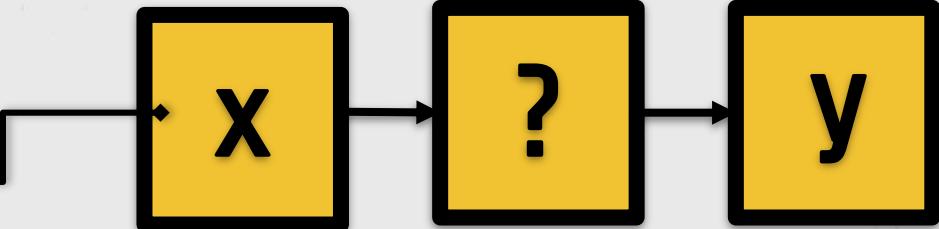
La conexión entre cada par de neuronas tiene un valor asociado que determina cuánta importancia tiene esa conexión para la red.



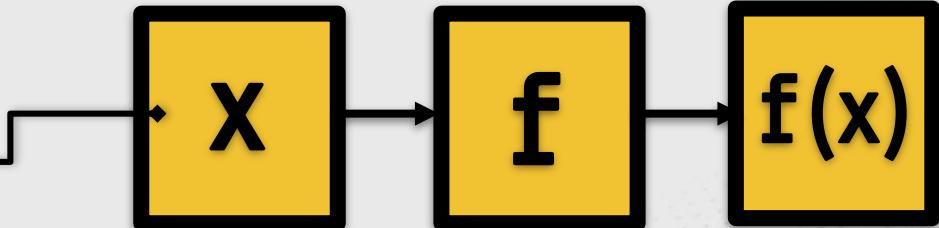
Una vez que una neurona ha recibido toda la información de la capa anterior, la neurona realiza algún tipo de transformación sobre esa información y transfiere el resultado a todas las neuronas de la siguiente capa.

Modelos matemáticos

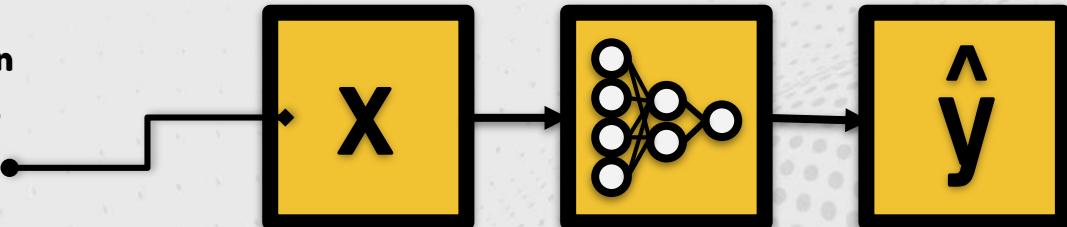
Los **modelos matemáticos** ayudan a encontrar la relación entre una variable conocida x y una variable desconocida y .



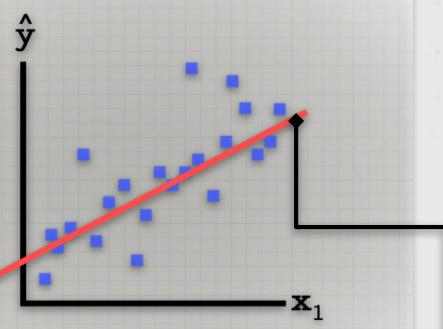
Eso **modelos usualmente se representan con una función f , por lo que $f(x)$ se refiere a la estimación del modelo sobre la variable x .**



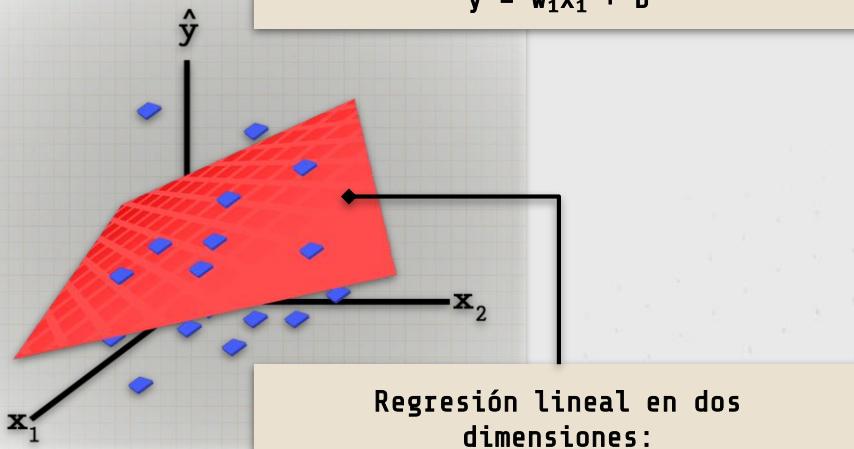
Las **redes neuronales artificiales** se pueden interpretar como los modelos matemáticos que encuentran esa relación entre variables.



Modelo de regresión lineal



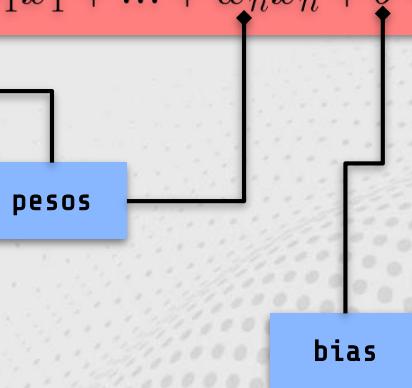
Regresión lineal en una dimensión:
 $y = w_1x_1 + b$



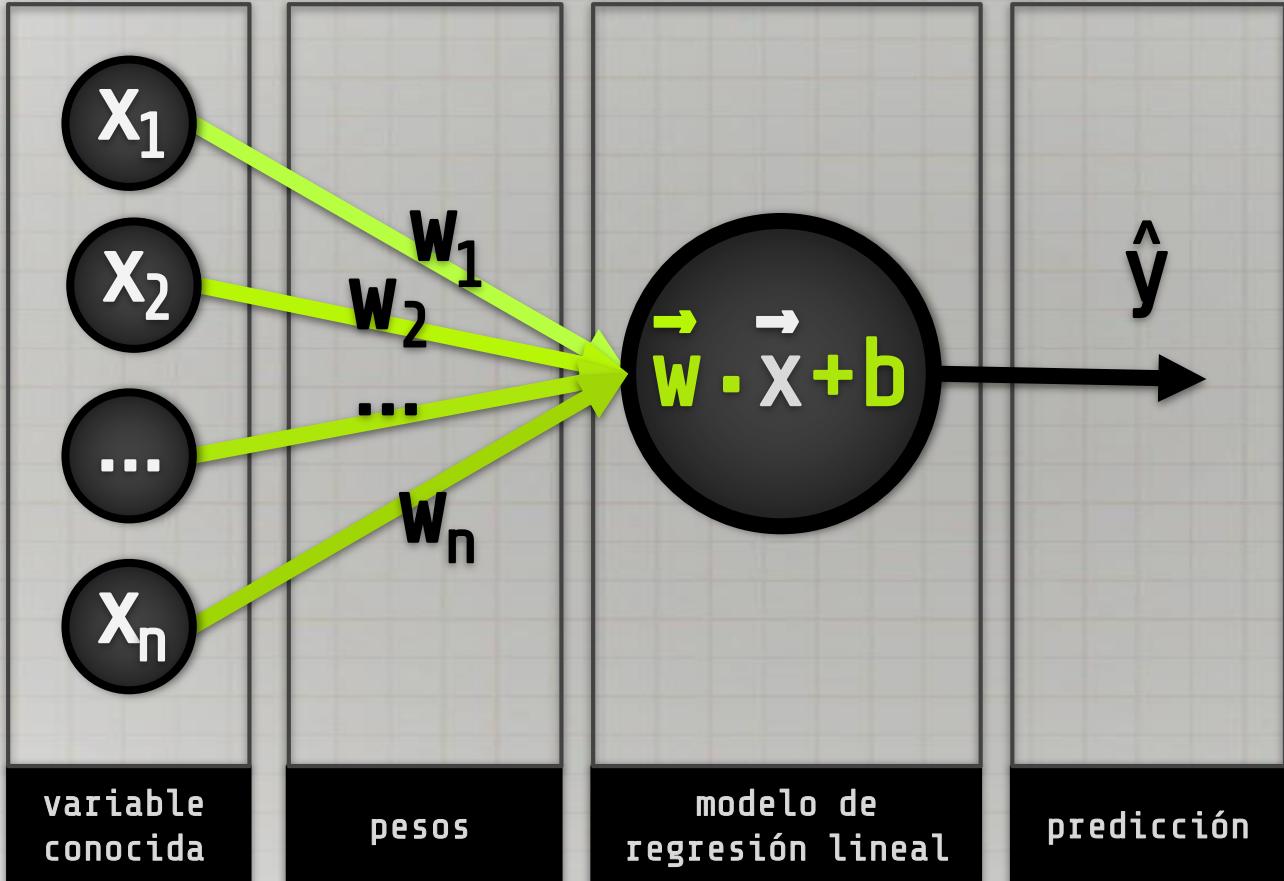
Regresión lineal en dos dimensiones:
 $y = w_1x_1 + w_2x_2 + b$

En un modelo de regresión lineal se supone que f es una función lineal, es decir, que \hat{y} es igual a la combinación lineal de todas las componentes del vector \vec{x} .

$$\hat{y} = f(\vec{x}) = w_1x_1 + \dots + w_nx_n + b = \vec{w} \cdot \vec{x} + b$$



Regresión lineal en una red

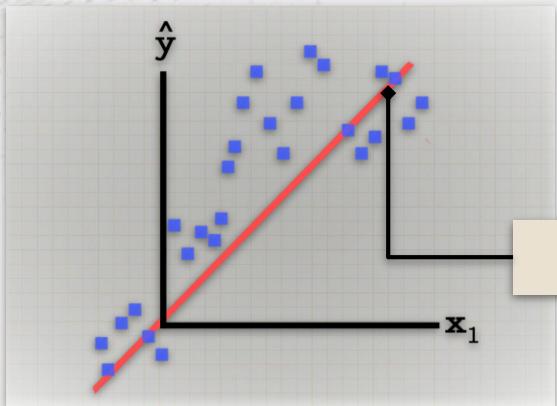


El modelo de regresión lineal se puede representar con una red neuronal artificial.

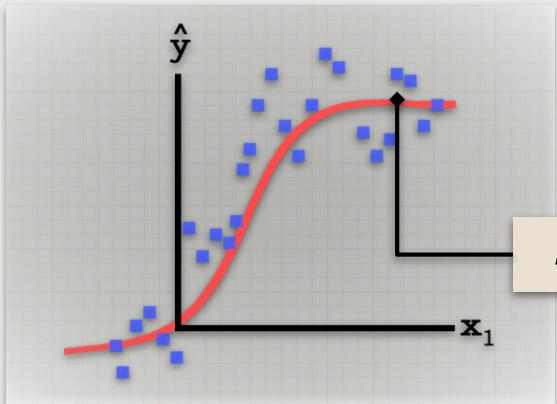
El resultado final de esta red neuronal está dado por la ecuación:

$$\hat{y} = \vec{w} \cdot \vec{x} + b$$

Funciones de activación



Ajuste lineal



Ajuste no lineal

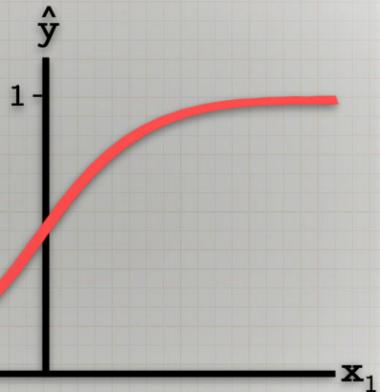
Las **funciones de activación** a son transformaciones no lineales que se aplican sobre el modelo de regresión lineal.

$$z = \vec{w} \cdot \vec{x} + b$$

$$\hat{y} = a(z)$$

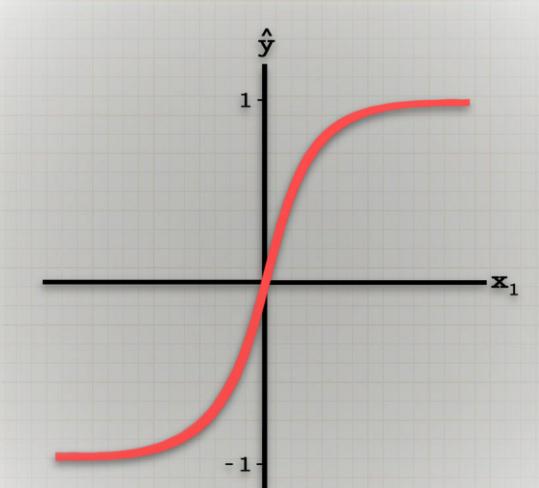
Función de activación

Funciones de activación



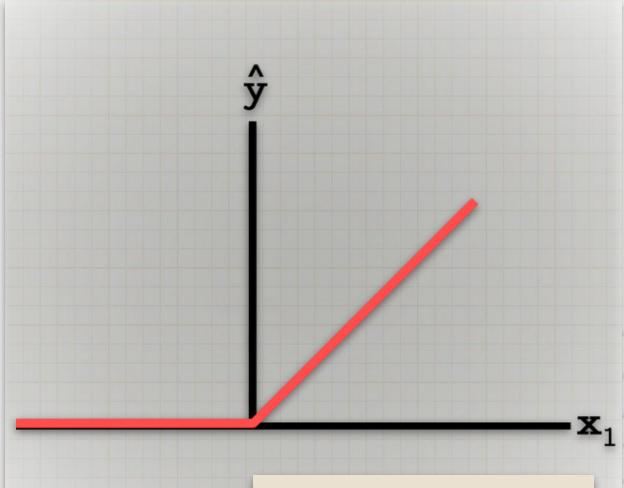
Función sigmoide

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$



Tangente
hiperbólica

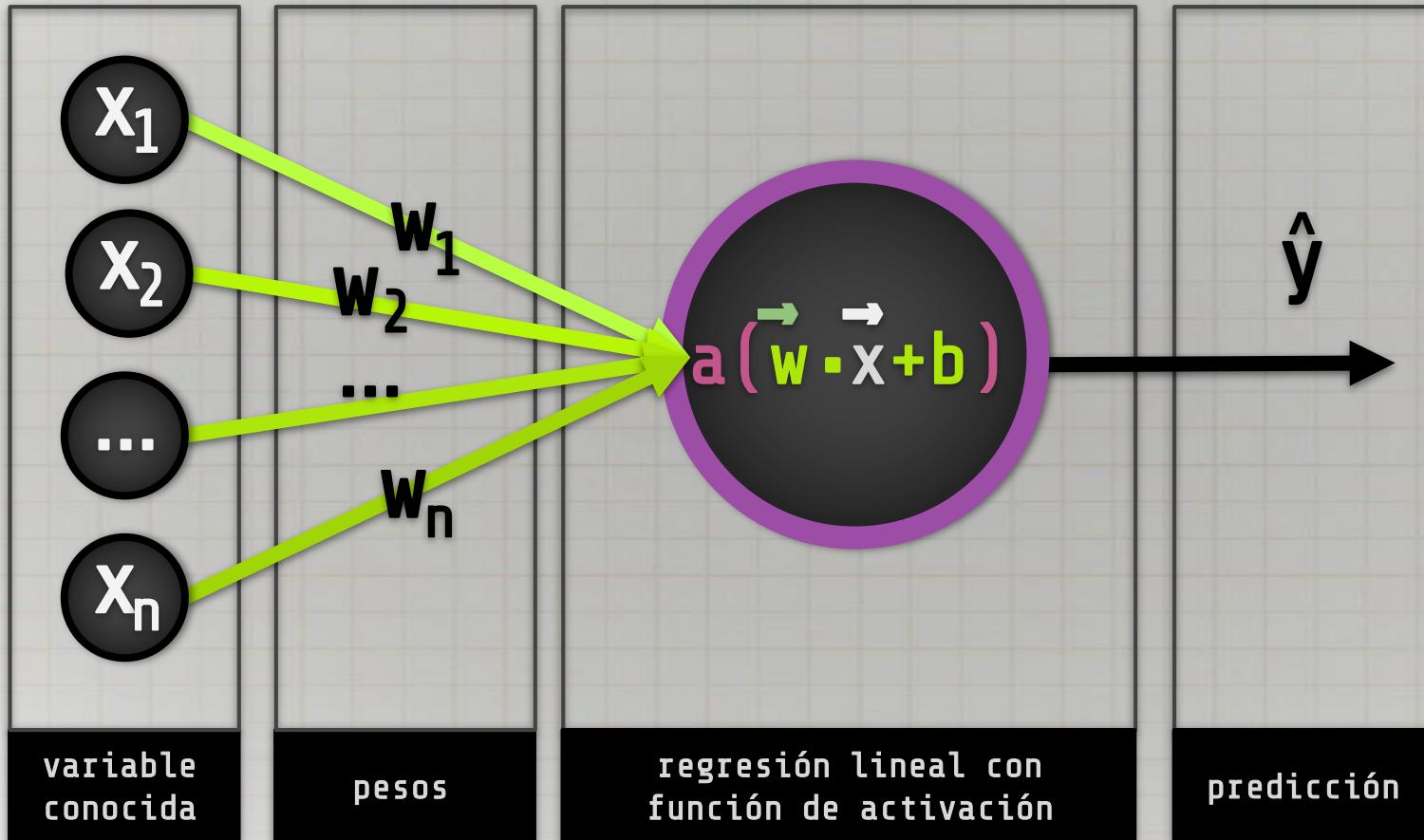
$$\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$



Función ReLU

$$ReLU(z) = \max(0, z)$$

Función de activación en una red



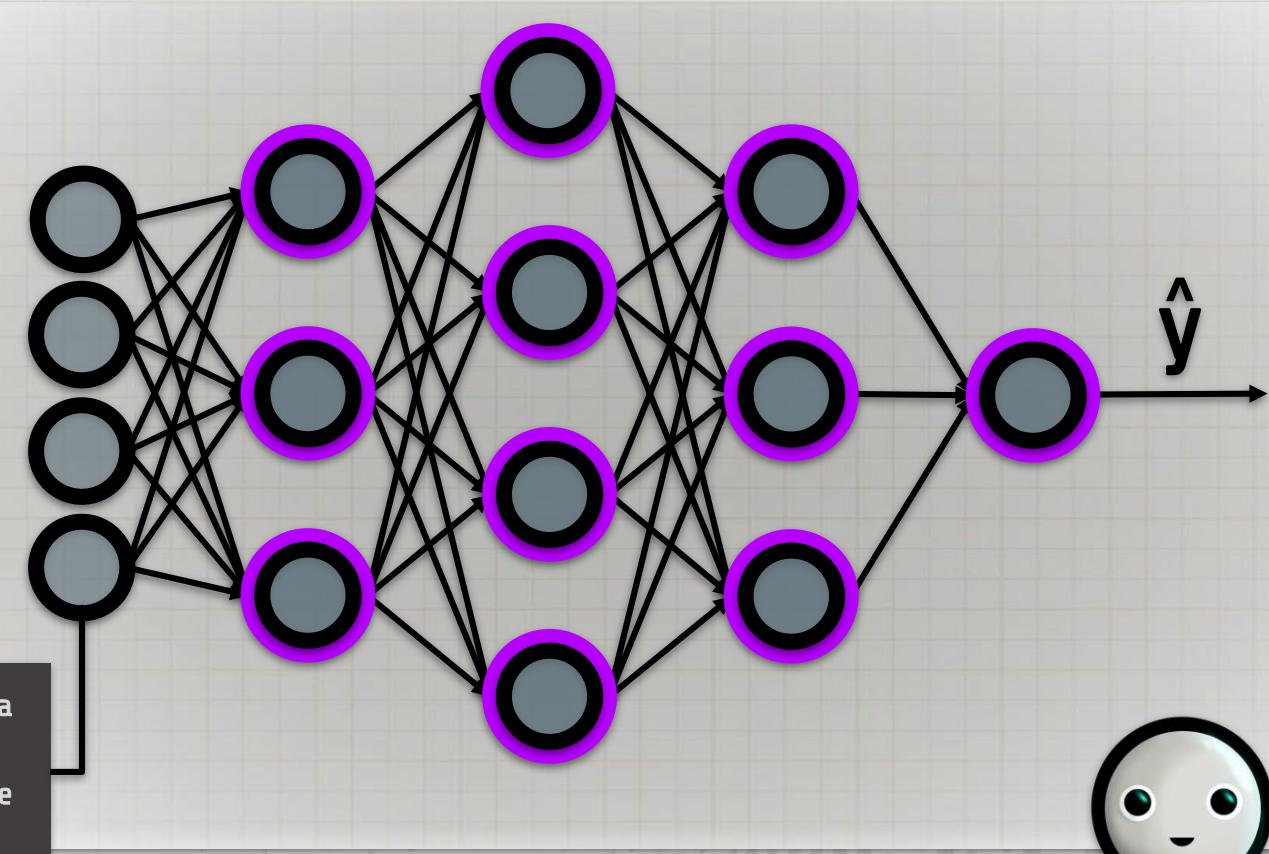
La función de activación se aplica sobre la neurona.

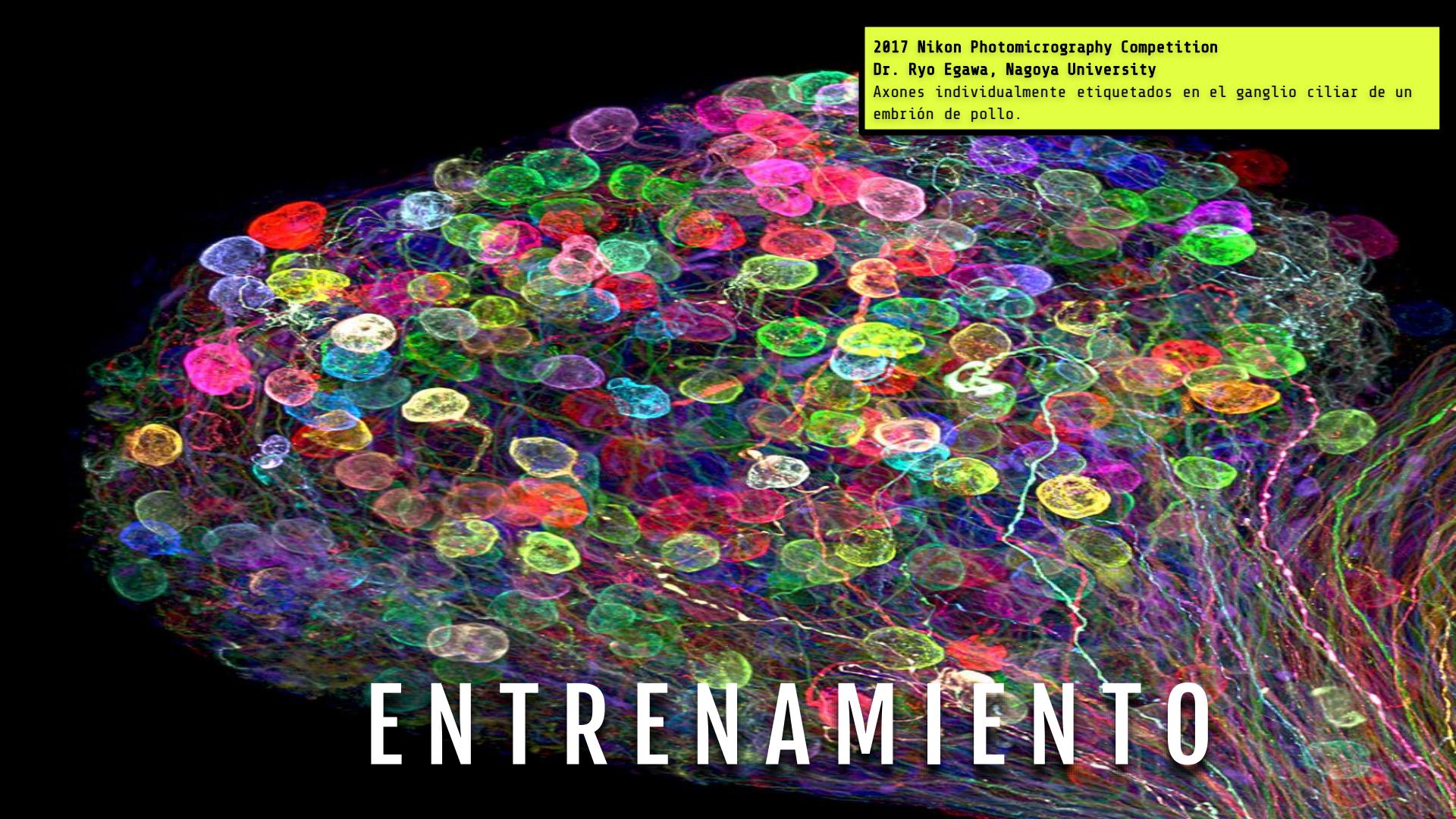
Ahora el resultado final de la red neuronal es:

Funciones de activación



Sobre las neuronas de la capa de entrada no se aplican las funciones de activación.



A fluorescence micrograph showing a dense cluster of cells, likely neurons, in a ganglion ciliare. Each cell body is outlined in a distinct color (green, red, blue, yellow, etc.), and numerous thin, colored fibers (axons) extend from these cells, forming a complex network that fills the frame.

2017 Nikon Photomicrography Competition

Dr. Ryo Egawa, Nagoya University

Axones individualmente etiquetados en el ganglio ciliar de un
embrión de pollo.

ENTRENAMIENTO

Dataset



conseguir conjunto
de datos



analizar los
datos



procesar los
datos

Rasgos y etiquetas



R: 175
G: 154
B: 169

RASGOS

→ $x = (\dots, (175, 154, 169), \dots)$

ETIQUETA

y = 1

Cada muestra del dataset está compuesta por un conjunto de rasgos y etiquetas. Las etiquetas son variables que dependen de los valores que tomen los rasgos.

Dependiendo del tipo de datos, se debe aplicar algún tipo de transformación para que queden en términos numéricos.

Entrenamiento

Los **parámetros** en una red neuronal artificial son los pesos y los biases.

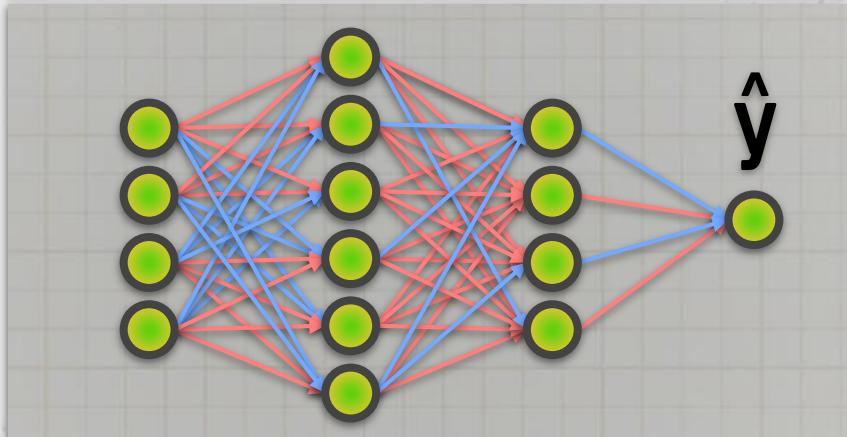
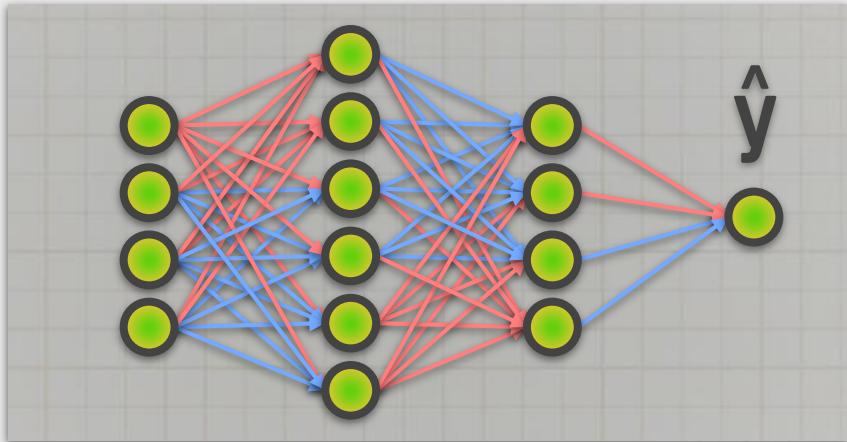
$$\hat{y} = a(z) = a(\vec{w} \cdot \vec{x} + b)$$

pesos

bias

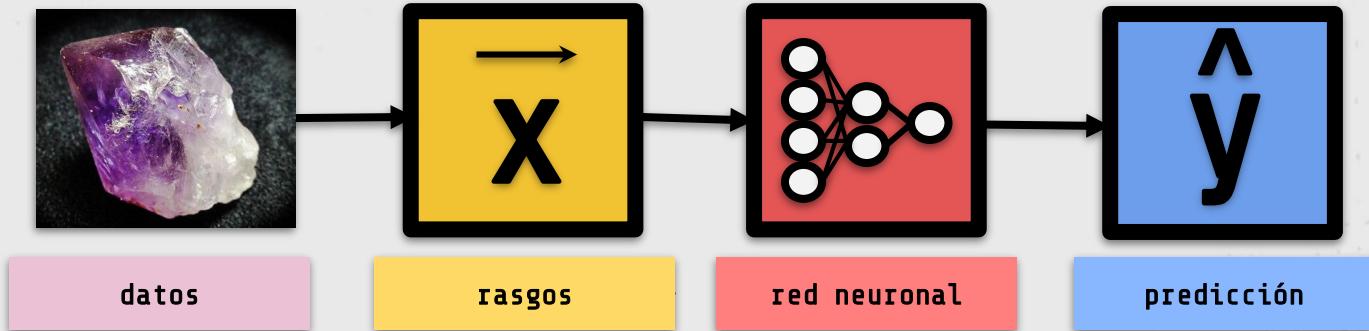
El objetivo del **entrenamiento** en una red neuronal artificial es determinar cuáles son los valores óptimos de los parámetros.

Con los **parámetros optimizados**, el resultado de la red neuronal \hat{y} será el más parecido a la etiqueta y .

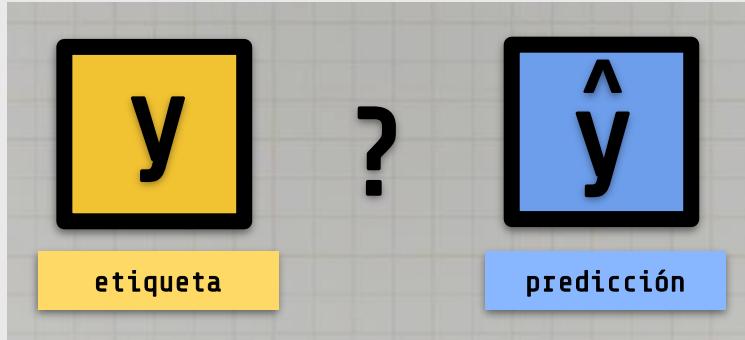


Propagación hacia delante

El primer paso consiste en tomar una muestra y determinar su vector de rasgos, después de lo cual se deberá introducir en la red neuronal.



A este proceso se le denomina como propagación hacia delante o **feedforward**.



Ya que se tiene el resultado de la red neuronal, lo siguiente es encontrar una manera de **comparar** el resultado con la etiqueta.

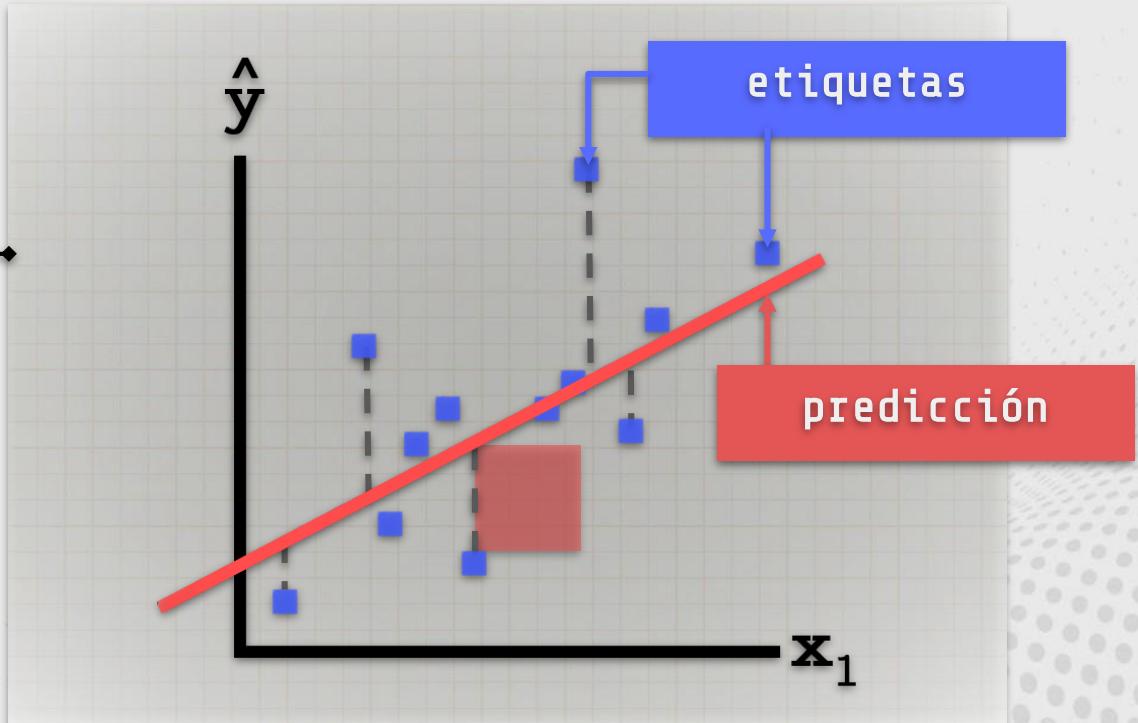
Función de pérdida

Una función de pérdida es una función que mide la distancia entre dos variables. En este caso, esas variables son la predicción y la etiqueta.

Una de las funciones de pérdida más utilizadas es la de mínimos cuadrados

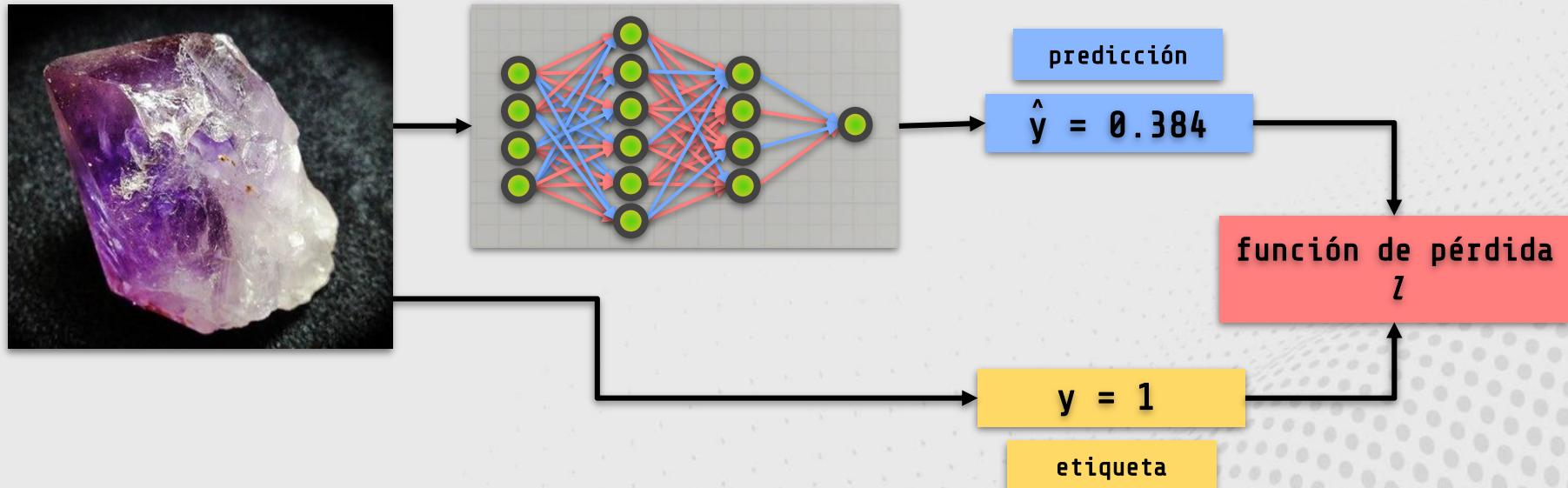
:

$$l = \frac{1}{2}(\hat{y} - y)^2$$



Función de pérdida

Para calcular la función de pérdida de una muestra, primero se tienen que propagar hacia delante los rasgos de la muestra y luego se hace el cálculo con la predicción y la etiqueta correspondiente.



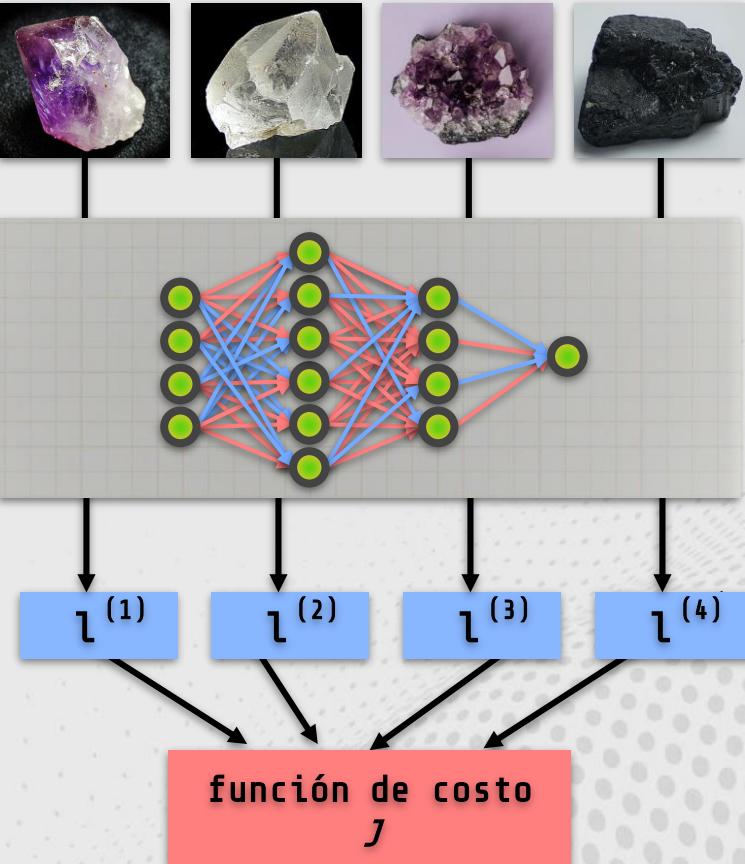
Función de costo

A todas las muestras en el dataset se les tiene que aplicar el proceso de propagación hacia delante y calcular su función de pérdida.

Al promedio de todas las funciones de pérdida se le conoce como **función de costo**:

$$J = \frac{1}{m} \sum_{i=1}^m l^{(i)} = \frac{1}{m} \sum_{i=1}^m \frac{1}{2} (\hat{y}^{(i)} - y^{(i)})^2$$

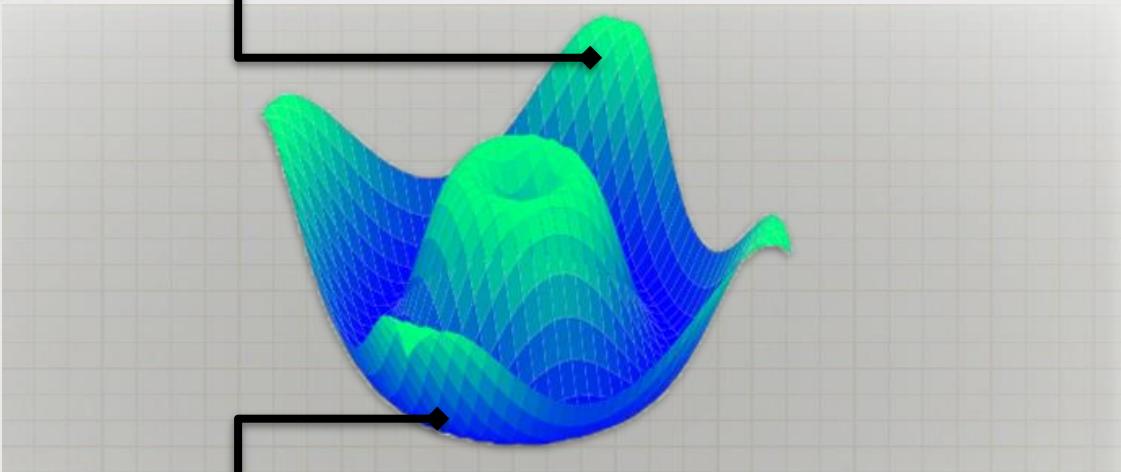
número de muestras
en el dataset



Función de costo

valores grandes
de J

y y \hat{y} tienen
valores alejados



valores pequeños
de J

y y \hat{y} tienen
valores cercanos

Geométricamente, la función de costo J se puede representar como una superficie donde cada punto corresponde a un conjunto de parámetros (pesos y biases).

Los valores óptimos de los parámetros son aquellos donde la función de costo alcanza su mínimo.

Para determinar el mínimo de la función de costo es necesario calcular el vector gradiente.

Vector gradiente

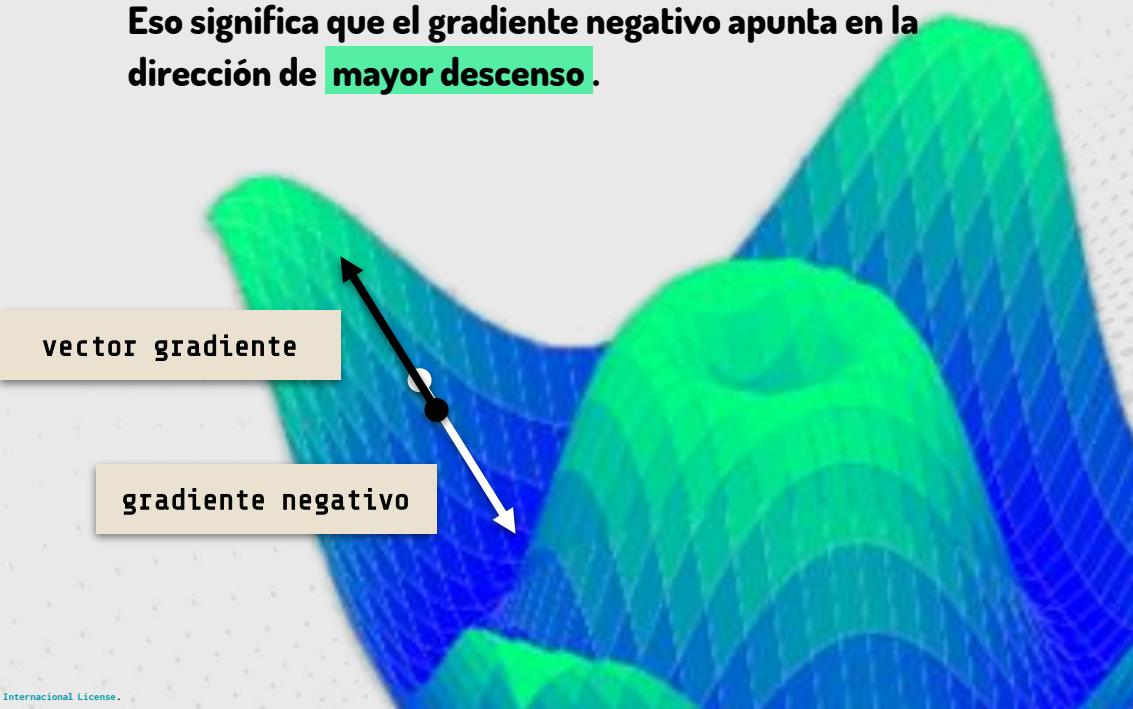
El **vector gradiente** de una función diferenciable g se define como el vector cuyas componentes corresponden a las derivadas parciales de la función:

$$\nabla g(\vec{x}) = \left(\frac{\partial g(\vec{x})}{\partial x_1}, \dots, \frac{\partial g(\vec{x})}{\partial x_n} \right)$$

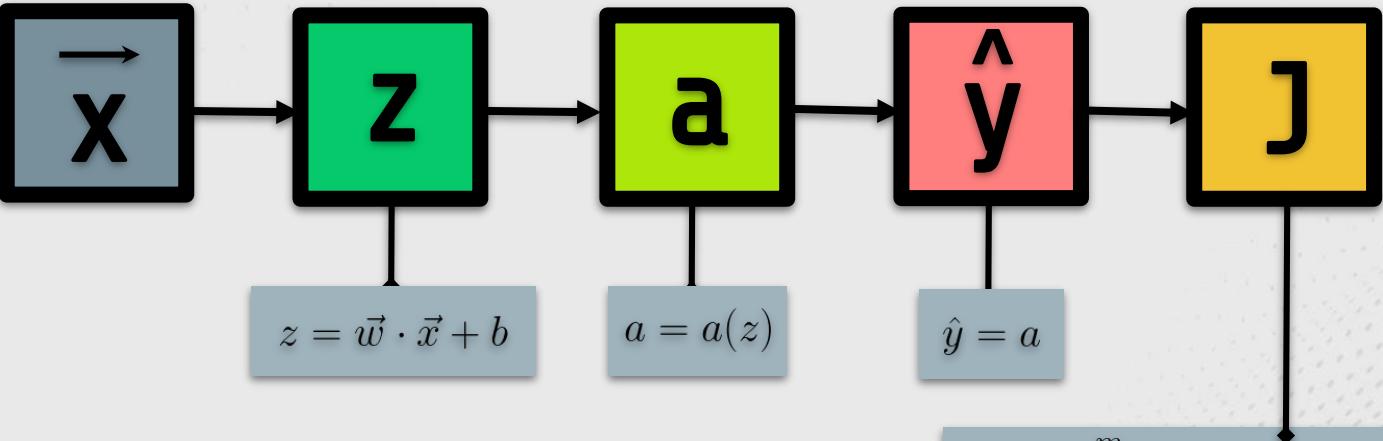
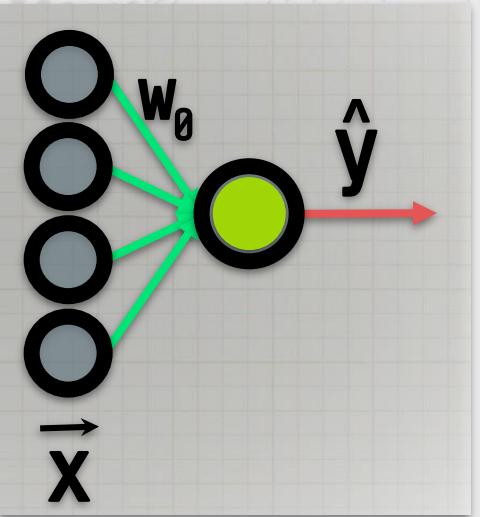
Para determinar el **gradiente de la función de costo** se tienen que calcular las derivadas respecto a los pesos y los biases.

Geométricamente, el vector gradiente se puede representar como el vector que apunta en la dirección de mayor ascenso.

Eso significa que el **gradiente negativo** apunta en la dirección de **mayor descenso**.



Gradiente de la función de costo

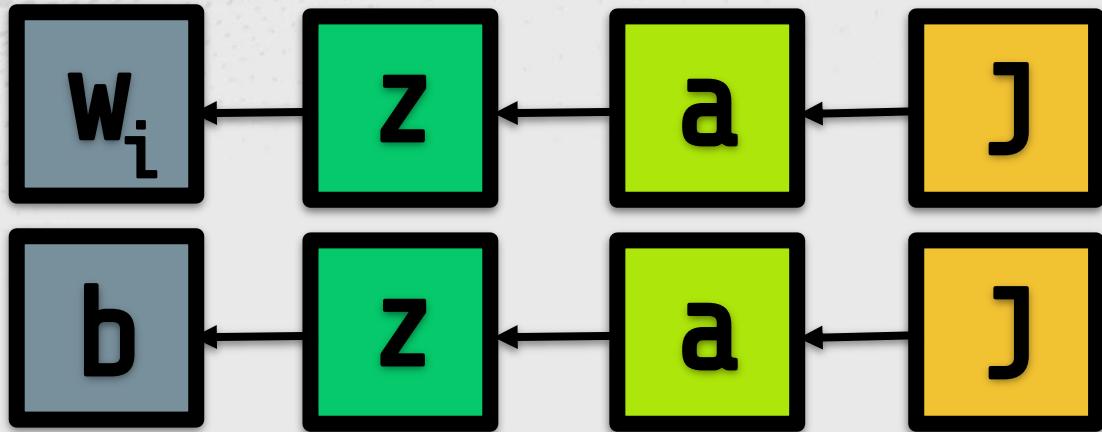


$$J = \frac{1}{m} \sum_{i=1}^m \frac{1}{2} (\hat{y}^{(i)} - y^{(i)})^2$$

Para calcular la derivada parcial de J respecto al peso w_i se tiene que utilizar la regla de la cadena:

$$\frac{\partial J}{\partial w_i} = \frac{\partial J}{\partial a} \frac{\partial a}{\partial z} \frac{\partial z}{\partial w_i}$$

Gradiente de la función de costo

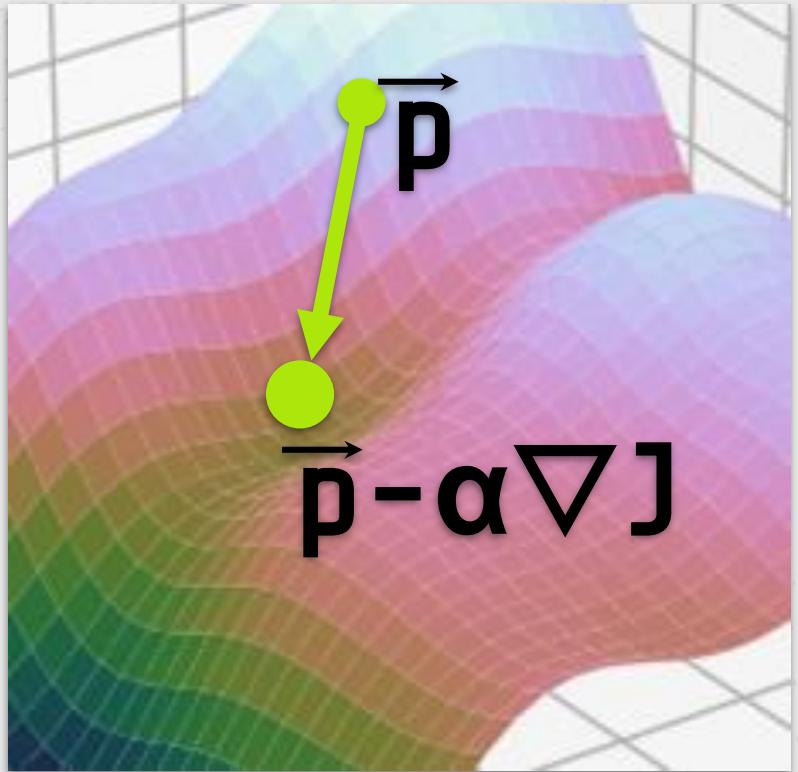


Dado que para calcular la derivada parcial de J respecto a algún peso o respecto al bias se tiene que recorrer a la red neuronal en la dirección opuesta, a este proceso se le conoce como propagación hacia atrás o **backpropagation**.

Después de haber calculado las derivadas parciales de la función de costo respecto a todos sus parámetros, se obtiene al **vector gradiente de la función de costo**:

$$\nabla J(\vec{w}, b) = \left(\frac{\partial J(\vec{w}, b)}{\partial w_1}, \dots, \frac{\partial J(\vec{w}, b)}{\partial w_n}, \frac{\partial J(\vec{w}, b)}{\partial b} \right)$$

Descenso del gradiente



Ya que se tiene al vector gradiente de la función de costo, lo siguiente es **actualizar los parámetros** de la red neuronal en un proceso conocido como **descenso del gradiente**.

La actualización de cada uno de los parámetros está dada por la siguiente transformación:

$$w_i = w_i - \alpha \frac{\partial J}{\partial w_i}$$

$$b = b - \alpha \frac{\partial J}{\partial b}$$

El coeficiente α se conoce como **razón de aprendizaje** y determina qué tanto cambia el valor de los parámetros.

Algoritmo de entrenamiento



El procedimiento que se siguió hasta ahora puede dividirse en tres pasos:

1. Propagación hacia delante y cálculo de la función de costo
2. Propagación hacia atrás y cálculo del vector gradiente
3. Descenso del gradiente

El **entrenamiento** de la red consiste en repetir este procedimiento iterativamente hasta que se alcance el mínimo de la función de costo, con lo cual se encontrarán los parámetros óptimos de la red neuronal.

Cuando se encuentren los parámetros óptimos, el entrenamiento de la red habrá terminado.

machine learning

inteligencia artificial

machine learning

deep learning

rasgos



dataset

etiquetas



núcleo

neurona

neurona
artificial
01011

$$J = \frac{1}{m} \sum_{i=1}^m l^{(i)} = \frac{1}{m} \sum_{i=1}^m \frac{1}{2} (\hat{y}^{(i)} - y^{(i)})^2$$

entrenamiento

$$\frac{\partial J}{\partial w_i} = \frac{\partial J}{\partial a} \frac{\partial a}{\partial z} \frac{\partial z}{\partial w_i}$$

gradiente

$$\nabla J(\vec{w}, b) = \left(\frac{\partial J(\vec{w}, b)}{\partial w_1}, \dots, \frac{\partial J(\vec{w}, b)}{\partial w_n}, \frac{\partial J(\vec{w}, b)}{\partial b} \right)$$

actualización de parámetros

$$w_i = w_i - \alpha \frac{\partial J}{\partial w_i}$$

inicio

costo

final

