Deep Learning

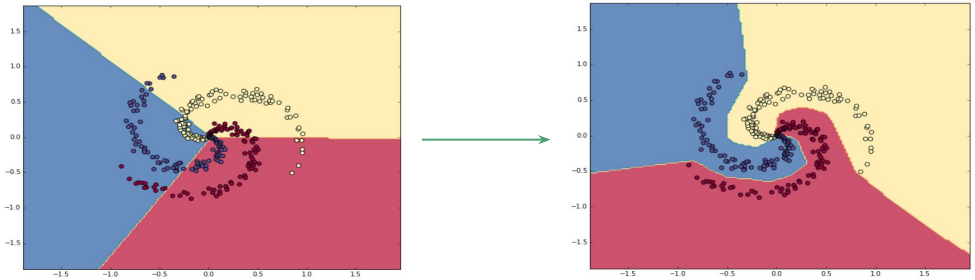
Sesión 2 - Resumen

De un vistazo

- Qué son las redes neuronales
- ¿Para qué sirven y para qué no?
- ¿Cómo aprenden las redes neuronales?
- Forward- y back- propagation
- Ejemplos y comparación TF1.x y TF2.x

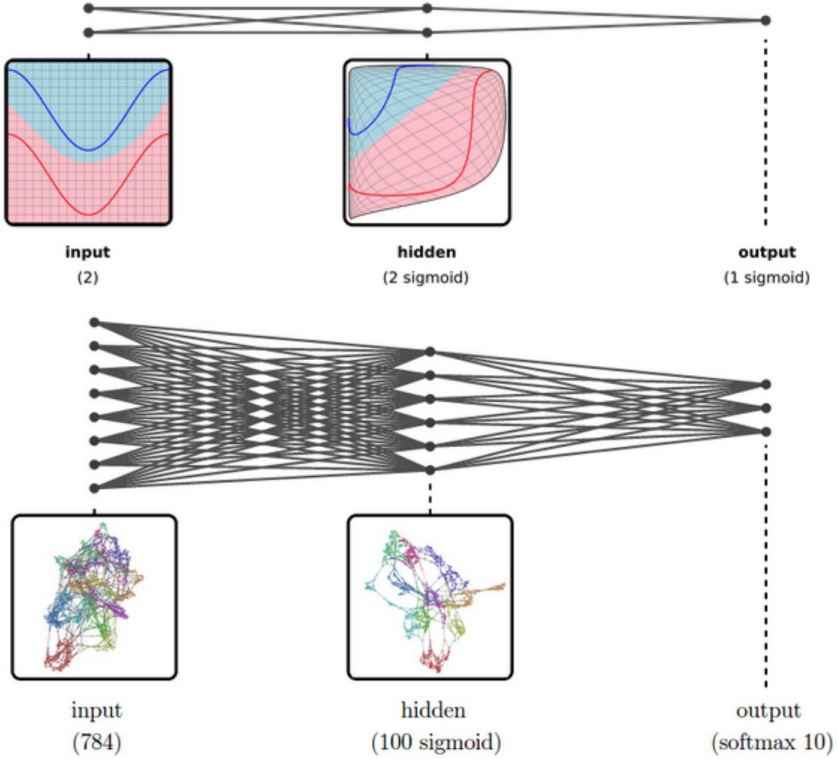
Qué son las redes neuronales

- Detectores de patrones muy complejos que permiten aproximar funciones
- Mapean los datos de entrada a un espacio diferente donde es más fácil resolver el problema en cuestión



- Son capaces de aproximar cualquier función, siempre que:
  - Sea continua
  - La aproximación se haga en un rango limitado de valores

Qué son las redes neuronales



¿Para qué sirven y para qué no?

Sirven para:

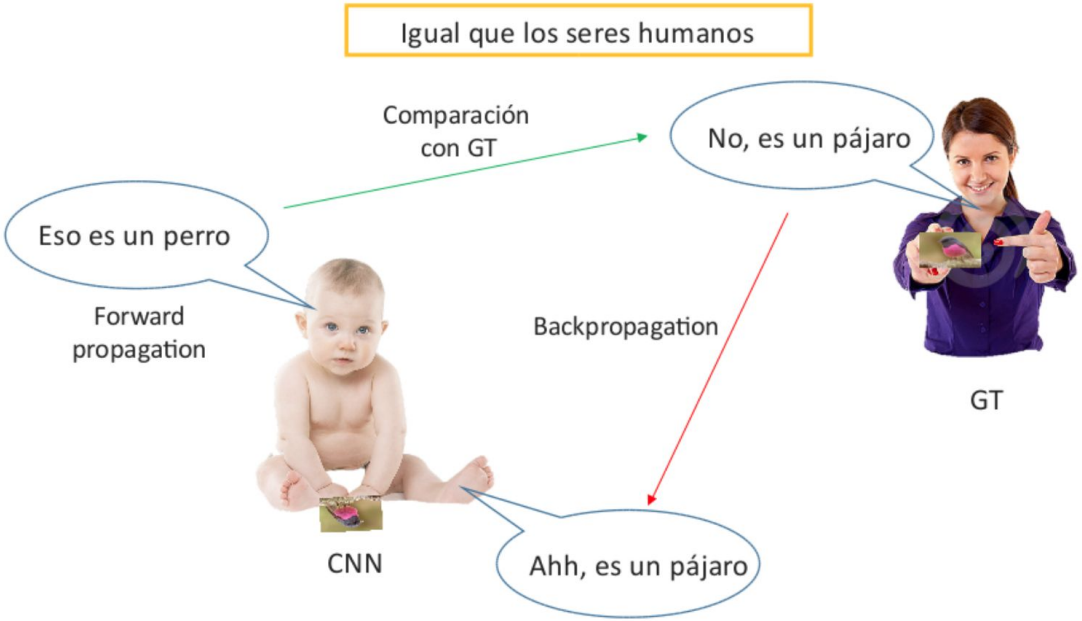
- Dar soluciones a problemas concretos y específicos que pueden ser modelados matemáticamente (incluso aquellos muy complejos)

No sirven para:

- “Pensar” o actuar de forma “inteligente” ➜ La inteligencia la ponemos nosotros

¿Cómo aprenden las redes neuronales?

¡Exactamente igual que nosotros!



¿Cómo aprenden las redes neuronales?

Los mecanismos de los que disponen las redes neuronales para aprender son:

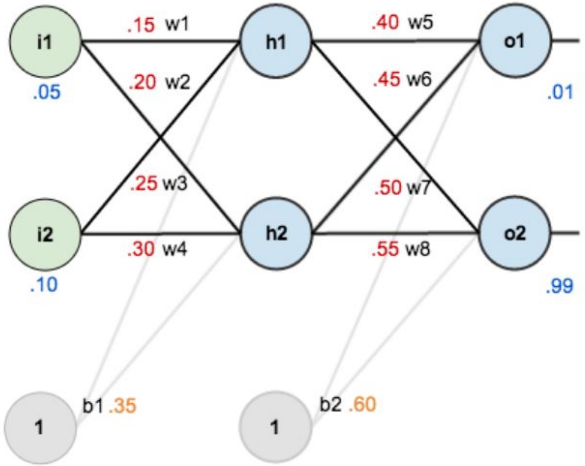
- **Forward propagation**: cuando el modelo hace una predicción basado en unos datos de entrada
- **Backward propagation**: cuando el modelo compara la predicción que ha realizado con la “ground truth”, calcula el error que ha cometido y actualiza sus parámetros (pesos) para tratar de dar una mejor predicción la próxima vez

La corrección del error, es decir, **el entrenamiento**, se realiza mediante el algoritmo de **descenso del gradiente** (gradient descent)

¿Cómo aprenden las redes neuronales?

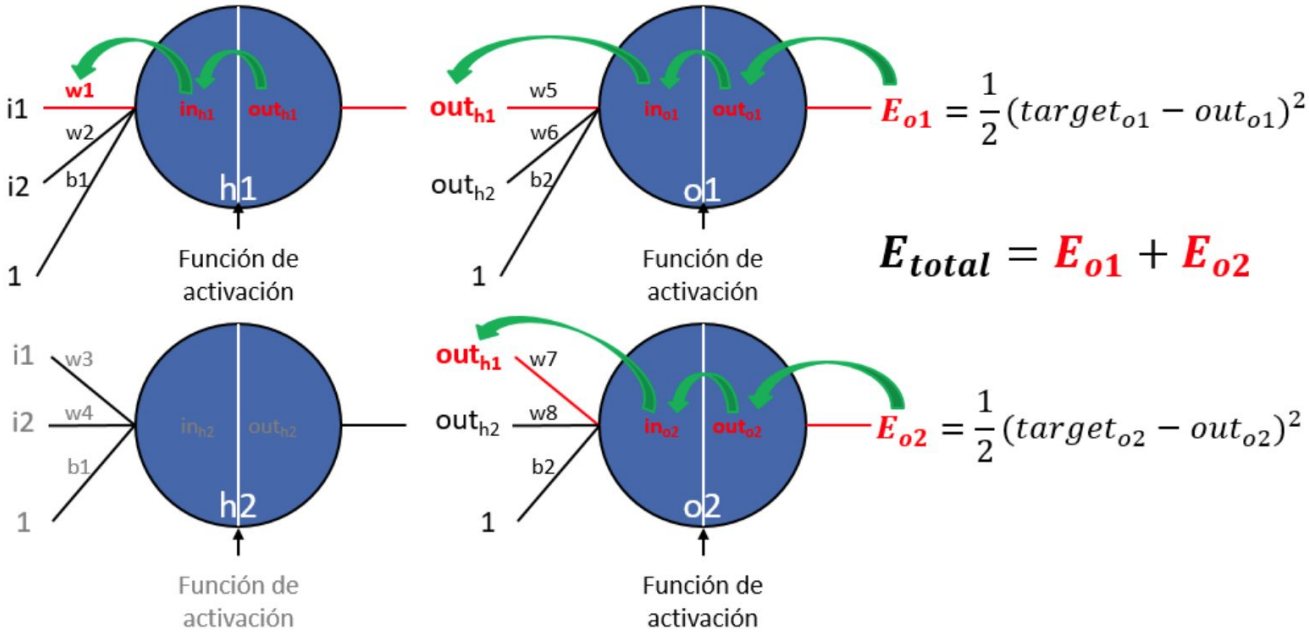
Para **entrenar la red**, calculamos la variación del error con respecto a cada parámetro

$$\frac{\partial E_{o1}}{\partial w_1} = \frac{\partial E_{o1}}{\partial out_{o1}} \cdot \frac{\partial out_{o1}}{\partial in_{o1}} \cdot \frac{\partial in_{o1}}{\partial out_{h1}} \cdot \frac{out_{h1}}{\partial in_{h1}} \cdot \frac{\partial in_{h1}}{\partial w_1}$$



¿Cómo aprenden las redes neuronales?

Veamoslo con un poco de zoom:



¿Cómo aprenden las redes neuronales?

Los pesos se actualizan en iteraciones a la velocidad que marca el **learning rate**

$$w_1^+ = w_1 - \eta \frac{\partial E_{total}}{\partial w_1}$$

$$w_2^+ = w_2 - \eta \frac{\partial E_{total}}{\partial w_2}$$

$$w_3^+ = w_3 - \eta \frac{\partial E_{total}}{\partial w_3}$$

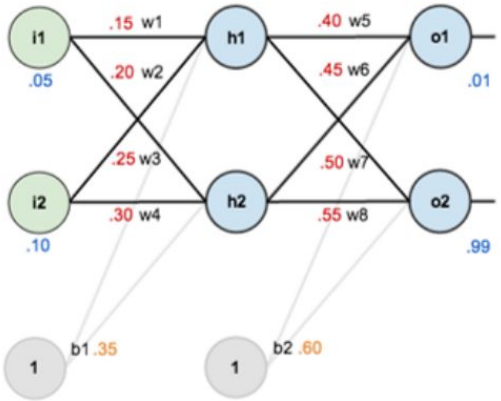
$$w_4^+ = w_4 - \eta \frac{\partial E_{total}}{\partial w_4}$$

$$w_5^+ = w_5 - \eta \frac{\partial E_{total}}{\partial w_5}$$

$$w_6^+ = w_6 - \eta \frac{\partial E_{total}}{\partial w_6}$$

$$w_7^+ = w_7 - \eta \frac{\partial E_{total}}{\partial w_7}$$

$$w_8^+ = w_8 - \eta \frac{\partial E_{total}}{\partial w_8}$$



Ejemplos y comparación TF1.x y TF2.x

- Por lo general TF1.x resulta más rápido que TF2.x, ya que al generar un grafo se realizan determinadas optimizaciones que permiten una ejecución más rápida
- TF2.x tiene la ventaja de que realiza una ejecución secuencial que permite su depuración paso a paso, a costa de ser, a menudo, más lenta
- Recurso interesante donde se comparan: https://stackoverflow.com/questions/58441514/why-is-tensorflow-2-much-slower-than-tensorflow-1