Redes Neuronales

Generalidades

- Aproximación a la inteligencia artificial que parte de tratar de modelar el funcionamiento físico del cerebro.
- Cerebro está compuesto por millones de millones de elementos computacionales simples (neuronas).
- Neuronas son lentas! Su velocidad no se ha incrementado evolutivamente.
- Capacidad computacional del cerebro proviene de paralelismo masivo.
- Es un computador no convencional, diseñado para realizar bien ciertas tareas:

  ? Percepción y representación del mundo. ? Inferencia probabilística.

  ? Manejar información conflictiva. ? Formar conceptos.

Inteligencia artificial - Breve reseña histórica

1888 Ramon y Cajal: Sistema Nervioso compuesto por células interconectadas.

1936 Alan Turing: Funcionamiento del cerebro humano como apx. para computación

1943 McCulloch y Pitts: Modelo de RNA simple con circuitos eléctricos. Propiedades Neuro – lógicas de conexiones entre neuronas, actividad de una neurona influenciado por otras.

1949 Hebb: Aprendizaje por refuerzo en individuos (redes neuronales biológicas), modificación de sinapsis. Dos neuronas se activan a la vez, se debe reforzar su conexión.

1958 Rosenblatt: Perceptron, APROXIMADOR UNIVERSAL

1960 Widrow: Red ADALINE (ADAptative LINear Elements)

1969 Minsky y Papert: Perceptron no es capaz de resolver problemas no lineales.   XOr

….?….

1982 Kohonen: Mapas auto-organizativos

Inteligencia artificial - Breve reseña histórica

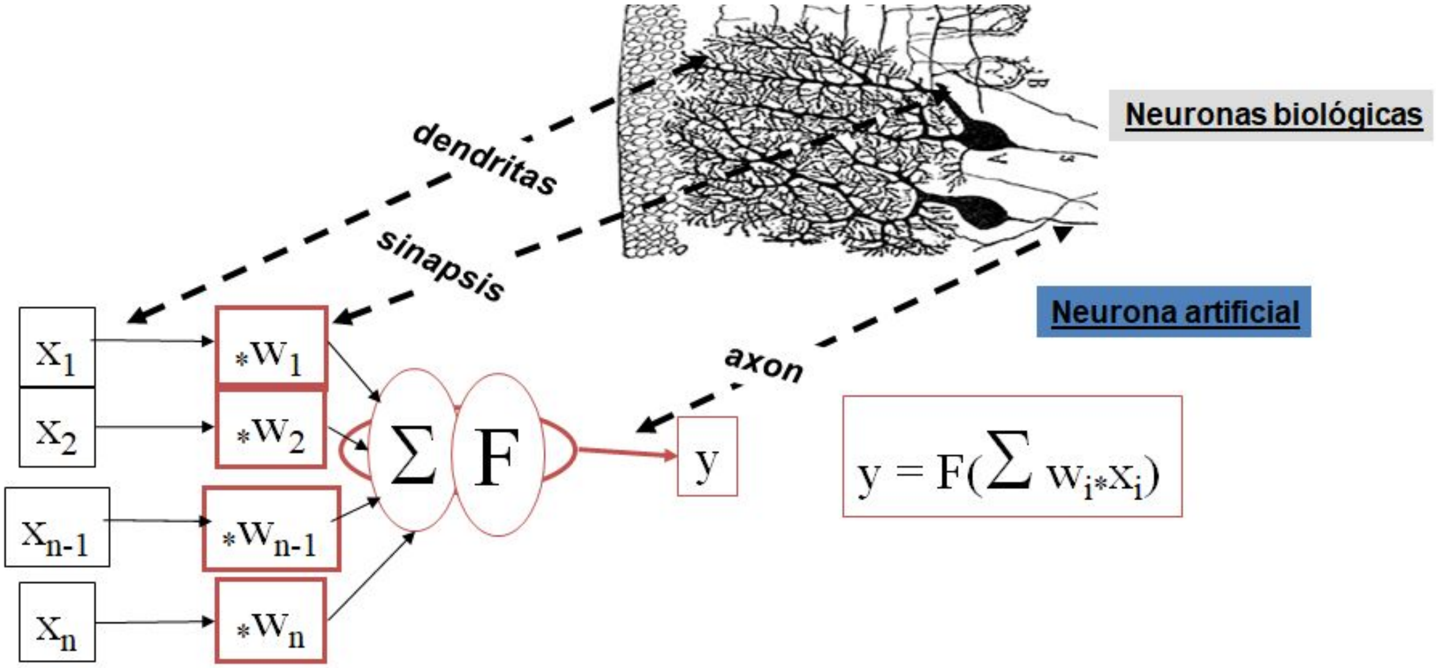
1985 Hopfield: Entrenamiento Perceptron Multicapa

1986 Rumelhart y Hinton: Redescubrimiento Algoritmo Backpropagation Rumelhart, Hinton y Williams (1986), Parker (1985), LeCun (1985) (Werbos,1974).

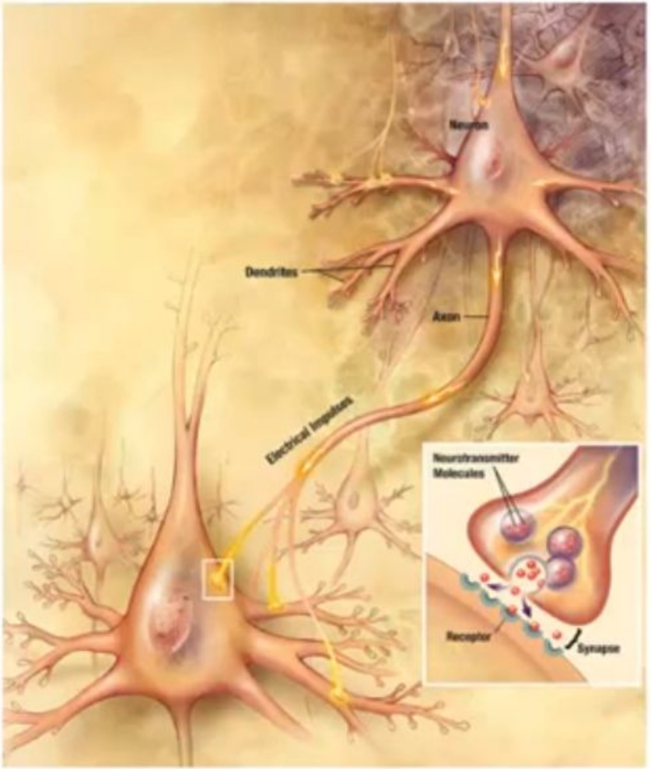
1988 Broomehead y Lowed: Funciones de Base Radial

1998 Teoría de generalización en redes neuronales

Biológico-Artificial: Neurona Artificial

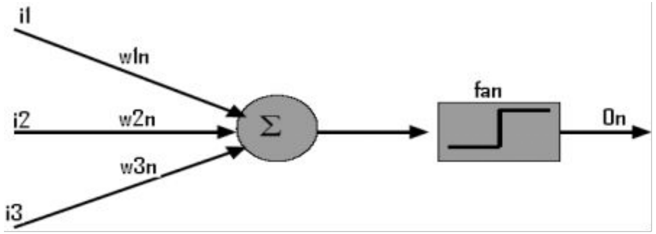


Biológico-Artificial: Neurona Artificial

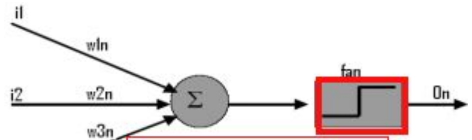


Redes neuronales - Neurona Artificial

- Nivel de Actividad: Señal de disparo.
- Conexiones de entrada: sinapsis.
- Peso de la conexión:
  - Positiva (excitatoria).
  - Cero
  - Negativa (inhibitoria).
- Conexión de salida: axón.
- Nivel de actividad de axones entrantes se multiplica por el peso de la sinapsis y determina el nivel de actividad en la salida

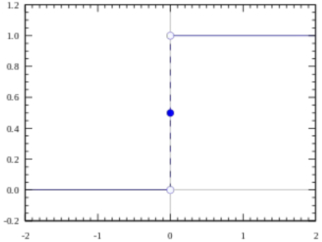


Redes neuronales - Modelo general

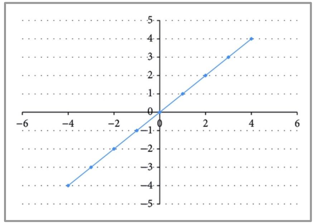


$$u_i(t) = f(y_i(t))$$

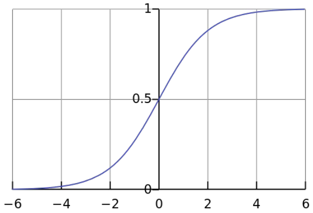
Funciones de activación: Dependiendo de su entrada su salida puede ser excitatoria o inhibitoria.



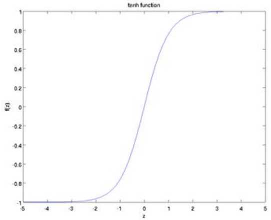
Step function



Linear function



Sigmoid Function



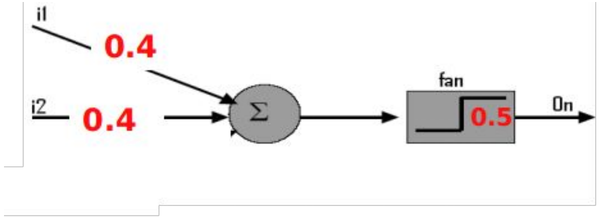
Tanh Function

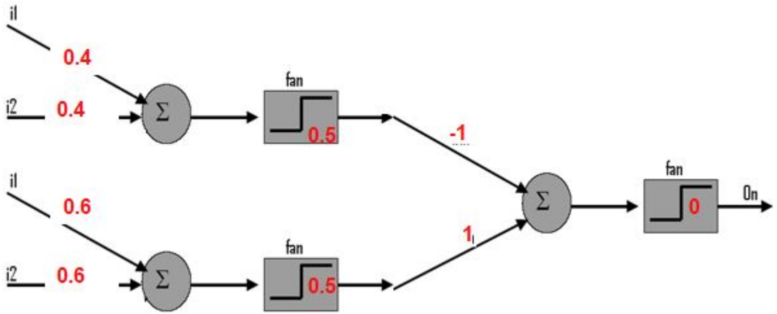
Artículo muy bueno para entenderlas:

https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0

Redes neuronales - Neurona Artificial

- Se comporta como una AND o una OR?





Redes neuronales - Fortaleza

- Computador masivamente paralelo.
- Neuronas efectúan operaciones muy básicas.
- Operación distribuida.
- Conocimiento adquirido a través de un proceso de aprendizaje.
- Capacidad de generalización.
- Conocimiento almacenado en fortaleza de conexiones sinápticas.
- Caja negra??? Para un sistema de decisión que parte tuvo la mayor influencia en la decisión?

Redes neuronales - Comparación cerebro

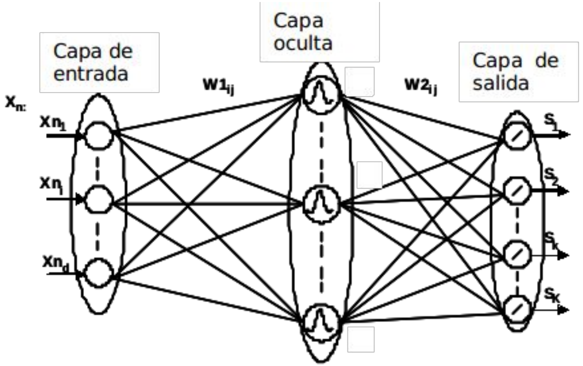
Redes neuronales

- Uno o dos tipos de neuronas.
- 1 – 100's de neuronas.
- 10 – 1000's de conexiones.
- 1 – 3 capas.

Cerebro

- 10^9 Neuronas.
- 10^11 Conexiones.
- Muchas Clases de Neuronas.

Redes neuronales - Partes

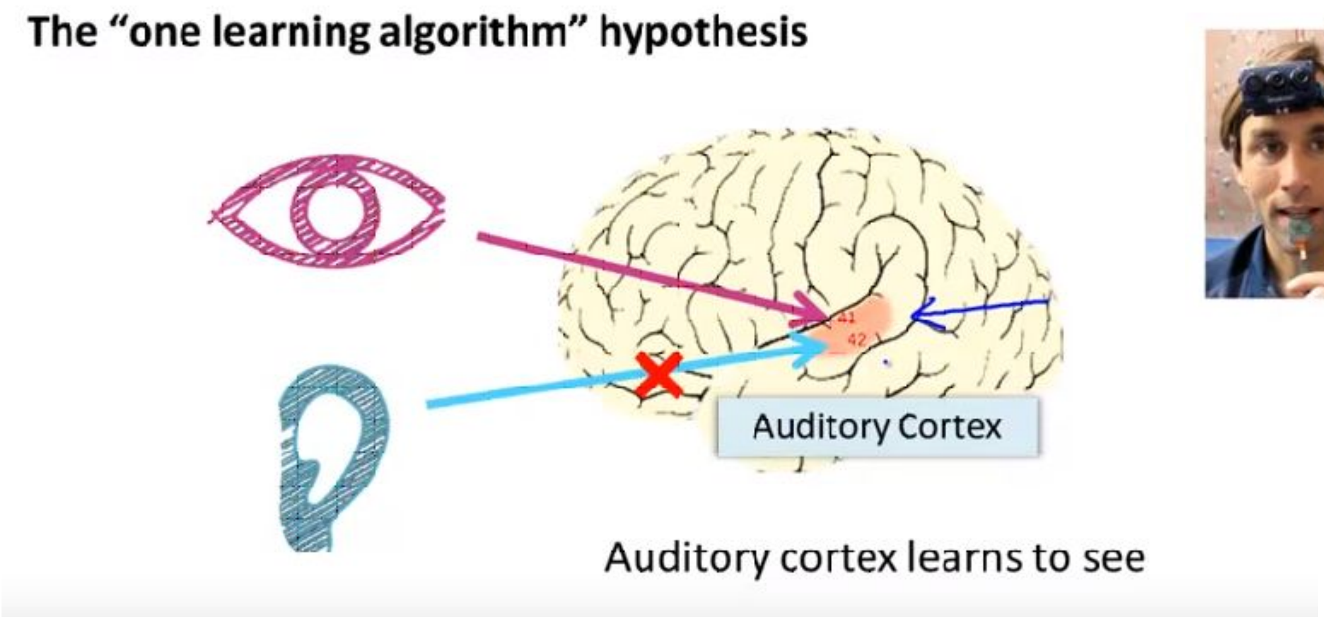


$$y_i(t) = \sum_{j=1}^{N} w_{ij} x_j(t)$$

$$u_i(t) = f(y_i(t))$$

- Normalmente el desempeño está relacionado con el número de neuronas de la capa oculta.

Redes neuronales - Experimento



Seeing with your tongue

Redes neuronales - Proceso

Fase de Aprendizaje: Conexiones sinápticas son modificadas

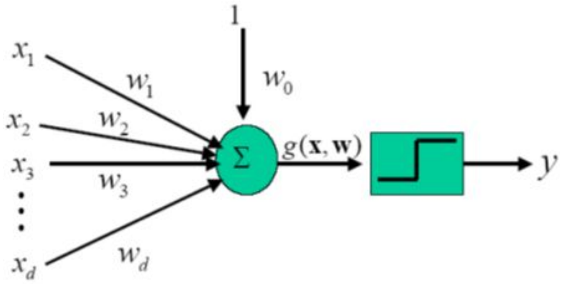
- A priori (condiciones iniciales).
- Algoritmo de aprendizaje.

Fase de Reconocimiento: Información inicial produce un patrón de salida.

- Computación distribuida.
- Comportamiento complejo.

Redes neuronales - Modelo de una neurona

$$
\begin{aligned}
\mathbf{x} &= \begin{bmatrix} x_1 & x_2 & \dots & x_d \end{bmatrix}^T \\
\mathbf{w} &= \begin{bmatrix} w_1 & w_2 & \dots & w_d \end{bmatrix}^T \\
g(\mathbf{x}, \mathbf{w}) &= \mathbf{w}^T \mathbf{x} + w_0 \\
y &= f_{LD}(g(\mathbf{x})) = f_{LD}(\mathbf{w}^T \mathbf{x} + w_0)
\end{aligned}
$$



Redes neuronales - Problema de aprendizaje

- Tenemos un conjunto de datos $\{\mathbf{x}_i, y_i\}, \mathbf{x}_i \in \mathbb{R}^d, y_i \in \{0, 1\}$.
- Queremos encontrar $\mathbf{w}$ que nos de una buena regla de clasificación *para datos futuros*.
- Si $z_i = f_{LD}(g(\mathbf{w}, \mathbf{x}_i))$, el objetivo es minimizar:

$$\mathbb{P}\{y_i \neq z_i\}$$

- En general, no podemos calcular $\mathbb{P}\{y_i \neq z_i\}$!
- Estrategia: minimizar función de error en los datos que sea calculable.

Redes neuronales - Problema de aprendizaje

- Algoritmo LMS
  - Widrow y Hoff (1960)
  - Adaptive linear networks (ADALINE).
- Algoritmo del Perceptrón
  - Rosemblatt (1962).
  - Prueba de convergencia.
- Perceptrón con bolsillo
  - Gallant (1986)
  - Para datos no linealmente separables.

Redes neuronales - LMS (Least Mean Squares)

- Función de error:

$$\begin{aligned} E(\mathbf{w}) &= \frac{1}{2}\sum_{i=1}^{n}(y_i - g(\mathbf{w}, \mathbf{x}_i))^2 \\ &= \frac{1}{2}\sum_{i=1}^{n}(y_i - \mathbf{w}^T\mathbf{x}_i))^2 \end{aligned}$$

- Problema de minimización:

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} E(\mathbf{w})$$

- Para el caso de una neurona, este problema admite una solución analítica.

Redes neuronales - LMS (Least Mean Squares)

- Defina:

$$\mathbf{X} = \begin{bmatrix}\mathbf{x}_1^T & \mathbf{x}_2^T & \cdots & \mathbf{x}_n^T\end{bmatrix}$$
$$\mathbf{y} = \begin{bmatrix}y_1 & y_2 & \cdots & \mathbf{y}_n\end{bmatrix}^T$$

- Entonces, para una solución con $E(\mathbf{w}) = 0$ se requiere:

$$\mathbf{X}\mathbf{w} = \mathbf{y}$$

- Es decir, $\mathbf{y}$ debe ser una combinación lineal de las columnas de $\mathbf{X}$.
- En general, no existe $\mathbf{w}$ que cumpla esta condición.

Redes neuronales - LMS (Least Mean Squares)

$$\begin{aligned}
E(\mathbf{w}) &= \frac{1}{2}\sum_{i=1}^{n}(\mathbf{w}^T\mathbf{x}_i - y_i)^2 \\
&= \frac{1}{2}\sum_{i=1}^{n}((\mathbf{w}^T\mathbf{x}_i)^2 - 2y_i\mathbf{w}^T\mathbf{x}_i + y_i^2) \\
&= \frac{1}{2}\sum_{i=1}^{n}(\mathbf{w}^T\mathbf{x}_i\mathbf{x}_i^T\mathbf{w} - 2y_i\mathbf{w}^T\mathbf{x}_i + y_i^2) \\
&= \frac{1}{2}\mathbf{w}^T\mathbf{H}\mathbf{w} - \mathbf{b}^T\mathbf{w} + c
\end{aligned}$$

$$\mathbf{H} = \textstyle\sum_{i=1}^{n} \mathbf{x}_i\mathbf{x}_i^T$$

$$\mathbf{b} = \textstyle\sum_{i=1}^{n} \mathbf{x}_i y_i$$

$$c = \tfrac{1}{2}\textstyle\sum_{i=1}^{n} y_i$$

Redes neuronales - LMS (Least Mean Squares)

- El mínimo de $E(\mathbf{w})$ ocurre donde $\nabla_{\mathbf{w}} E = 0$:

$$
\begin{aligned}
\nabla_{\mathbf{w}} E &= \mathbf{H}\mathbf{w} - \mathbf{b} = 0 \\
\hat{\mathbf{w}} &= \mathbf{H}^{-1}\mathbf{b}
\end{aligned}
$$

- Se requiere invertir $\mathbf{H}$, que puede no ser invertible o ser mal condicionada.
- LMS: solución iterativa.

Redes neuronales - LMS (Least Mean Squares)

- Procedimiento iterativo:

1. Comenzar en un punto (aleatorio):

$$\mathbf{w}_0 = \text{random}$$

2. Búsqueda de gradiente: Ir "hacia abajo de la colina".

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \mu \nabla_{\mathbf{w}} E|_{\mathbf{w}_k}$$

- $\nabla_{\mathbf{w}} E|_{\mathbf{w}_k}$ no se calcula exactamente.

Redes neuronales - LMS (Least Mean Squares)

$$
\begin{aligned}
E(\mathbf{w}) &= \frac{1}{2}\sum_{i=1}^{n}(y_i - \mathbf{w}^T\mathbf{x}_i))^2 \\
\nabla_{\mathbf{w}} E &= \sum_{i=1}^{n}(\mathbf{w}^T\mathbf{x}_i - y_i)\mathbf{x}_i
\end{aligned}
$$

- $\nabla_{\mathbf{w}} E|_{\mathbf{w}_k}$ se estima a partir de un subconjunto de los datos.
- Varias pasadas por los datos.
- Usualmente se usa un solo dato:

$$
\begin{aligned}
\nabla_{\mathbf{w}} E &\approx (\mathbf{w}^T\mathbf{x}_i - y_i)\mathbf{x}_i \\
&= e_i\mathbf{x}_i
\end{aligned}
$$

Redes neuronales - LMS (Least Mean Squares)

- Procedimiento iterativo:

1. Comenzar en un punto (aleatorio):

$$\mathbf{w}_0 = \text{random}$$

$\mathbf{w}_0$ a valores pequeños.

2. Búsqueda de gradiente: Ir "hacia abajo de la colina".

repeat
Escoja $(\mathbf{x}_i, y_i)$
$g = \mathbf{w}_k^T \mathbf{x}_i$
$e = g - y_i$
$\mathbf{w}_{k+1} = \mathbf{w}_k - \mu e \mathbf{x}_i$
until Condición de terminación.

$\boxed{E \leq E_{min}}$ $\boxed{\nabla_{\mathbf{w}} E \leq \nabla_{\mathbf{w}} E_{min}}$ Validación cruzada.

Redes neuronales - Perceptron

- Conjunto de datos:

$$\{\mathbf{x}_i, y_i\}, \mathbf{x}_i \in \mathbb{R}^d, y_i \in \{-1, 1\}$$

- $(\mathbf{x}_i, y_i)$ es clasificado correctamente si:

$$\begin{aligned} g(\mathbf{w}, \mathbf{x}_i)y_i &> 0 \\ (\mathbf{w}^T\mathbf{x}_i)y_i &> 0 \end{aligned}$$

- Criterio de error del perceptrón:

$$E(\mathbf{w}) = -\sum_{\mathbf{x}_i \in \mathcal{M}} (\mathbf{w}^T\mathbf{x}_i)y_i, \quad \mathcal{M} = \{\mathbf{x}_i : (\mathbf{w}^T\mathbf{x}_i)y_i < 0\}$$

Redes neuronales - Perceptron

- Procedimiento iterativo:

1. Comenzar en:

$$\mathbf{w}_0 = 0$$

2. Búsqueda de gradiente: Ir "hacia abajo de la colina".

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \nabla_{\mathbf{w}} E|_{\mathbf{w}_k}$$

- Nuevamente, $\nabla_{\mathbf{w}} E|_{\mathbf{w}_k}$ no se calcula exactamente:

$$\mathcal{M} = \{\mathbf{x}_i : (\mathbf{w}^T \mathbf{x}_i) y_i < 0\}$$

Los datos que están mal clasificados

$$\nabla_{\mathbf{w}} E = \sum_{\mathbf{x}_i \in \mathcal{M}} \mathbf{x}_i y_i$$
$$\approx -\mathbf{x}_i y_i$$

+Si se espera +1
- Si se espera -1

Incialize $\mathbf{w}_0 = 0$
repeat
  Escoja $(\mathbf{x}_i, y_i)$ al azar
  if $(\mathbf{w}^T \mathbf{x}_i) y_i < 0$ **then**
    $\mathbf{w}_{k+1} = \mathbf{w}_k + \mathbf{x}_i y_i$
  end if
until Convergencia.

- Cuando no hay separabilidad, el algoritmo oscila y no termina.

Redes neuronales - Perceptron con bolsillo

•PERCEPTRON: **Clasificación binaria**

$$E(\mathbf{w}) = -\sum_{\mathbf{x}_i \in \mathcal{M}} (\mathbf{w}^T \mathbf{x}_i) y_i, \quad \mathcal{M} = \{\mathbf{x}_i : (\mathbf{w}^T \mathbf{x}_i) y_i < 0\}$$

Los datos que están mal clasificados

•LMS: **SALIDA NO BINARIA**

$$E(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^{n} (y_i - \mathbf{w}^T \mathbf{x}_i)^2$$

Redes neuronales - Perceptron con bolsillo

No. De datos clasificados correctamente

Incialize $\mathbf{w}_0 = 0, t = 0, t_{max} = 0, \mathbf{w}_{bolsillo} = 0$

repeat

  Escoja $(\mathbf{x}_i, y_i)$ al azar

  if $(\mathbf{w}^T\mathbf{x}_i)y_i > 0$ **then**

    t=t+1;

    if $t > t_{max}$ **then**

      $t_{max} = t$

      $\mathbf{w}_{bolsillo} = \mathbf{w}$

    end if

  else

    $\mathbf{w}_{k+1} = \mathbf{w}_k + \mathbf{x}_i y_i$

    $t = 0$

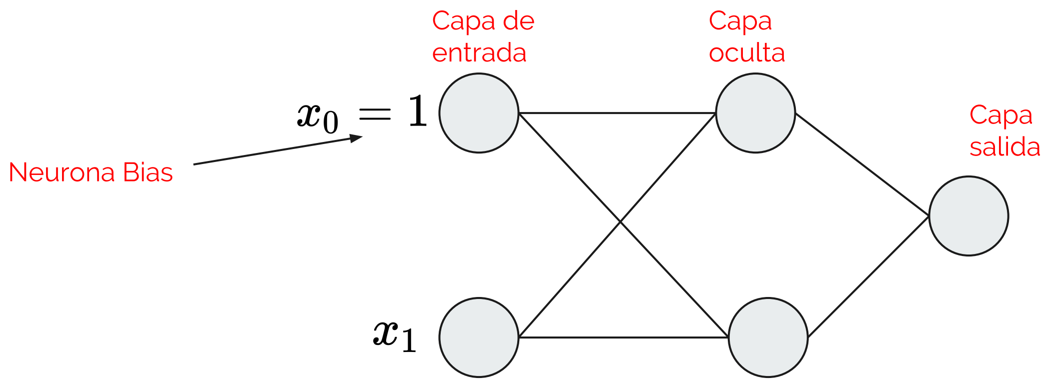
  end if

until Convergencia.

No. Mínimo de buenas clasificaciones

Hay respuesta correcta

Redes neuronales - Forward propagation



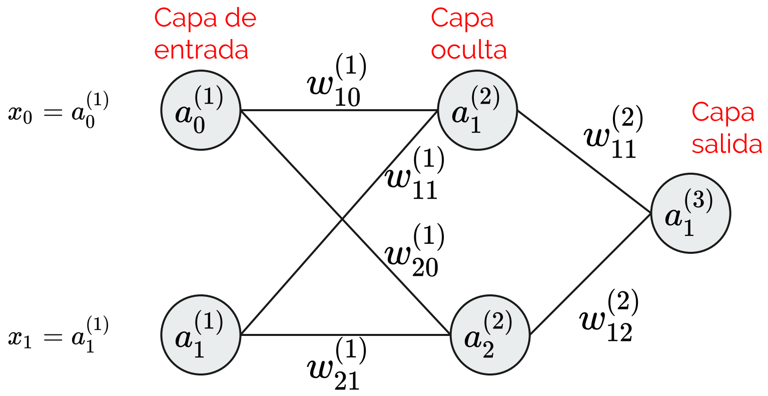
Redes neuronales - Forward propagation

Agreguemos un poco de notación:

Cada neurona tiene una salida $a_k^{(L)}$. El subíndice se refiere al numero de la neurona de esa capa. El superíndice se refiere al número de la capa.

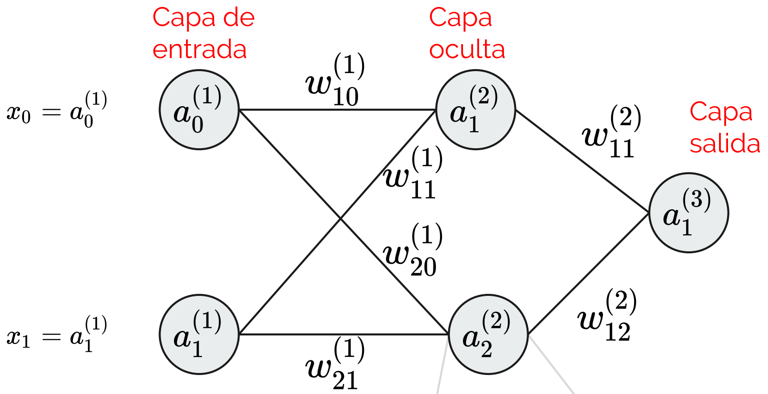
La capa de entrada es la capa 1.

Cada peso $w_{ij}^{(L)}$ se refiere a la conexión de la neurona i de la capa L, con la neurona j de la capa $L+1$



Redes neuronales - Forward propagation

Qué pasa al interior de cada neurona?

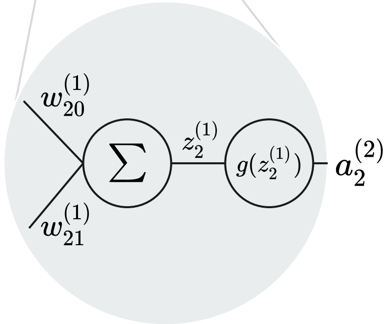


$$z_2^{(1)} = w_{20}^{(1)} a_0^{(1)} + w_{21}^{(1)} a_1^{(1)}$$

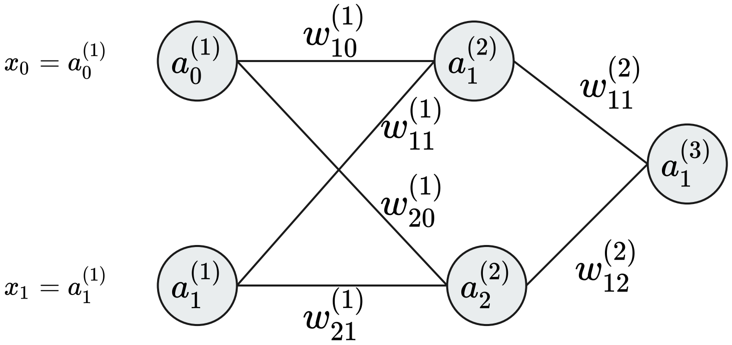
$$z_2^{(1)} = w_{20}^{(1)} x_0 + w_{21}^{(1)} x_1$$

Función de activación g

$$a_2^{(2)} = g(z_2^{(1)})$$



Redes neuronales - Forward propagation



Capa oculta (2)

$$z_1^{(1)} = w_{10}^{(1)} x_0 + w_{11}^{(1)} x_1 \qquad a_1^{(2)} = g(z_1^{(1)})$$

$$z_2^{(1)} = w_{20}^{(1)} x_0 + w_{21}^{(1)} x_1 \qquad a_2^{(2)} = g(z_2^{(1)})$$

Capa de salida

$$z_1^{(2)} = w_{11}^{(2)} a_1^{(2)} + w_{12}^{(2)} a_2^{(2)} \qquad a_1^{(3)} = g(z_1^{(2)})$$

Redes neuronales - Entrenamiento

Ya sabemos cómo se produce la salida de la red. Pero cómo encontramos estos pesos que producirán la salida deseada?

Para esto definimos una función de costo o error, que es lo que vamos a minimizar.

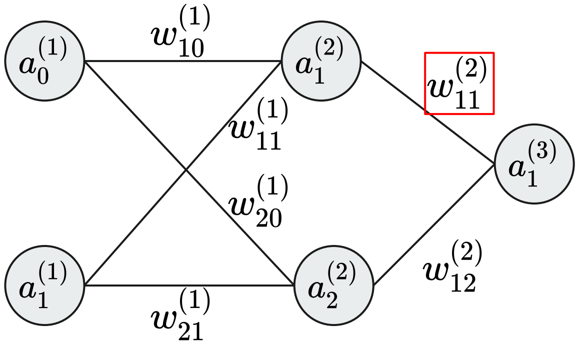
$$E = \frac{1}{2} \sum_m \left[y - a_1^{(3)} \right]^2$$

Si utilizamos el método de gradiente descendiente, cada peso se puede obtener así:

$$w_{ij}^{(L)} = w_{ij}^{(L)} - \alpha \frac{\delta E}{\delta w_{ij}^{(L)}}$$

Redes neuronales - Entrenamiento

Empecemos con los pesos antes de la capa de salida:



$$\frac{\delta E}{\delta w_{11}^{(2)}} = -\sum_m \left[y - a_1^{(3)}\right] \frac{\delta a_1^{(3)}}{\delta w_{11}^{(2)}}$$

$$a_1^{(3)} = g(z_1^{(2)})$$

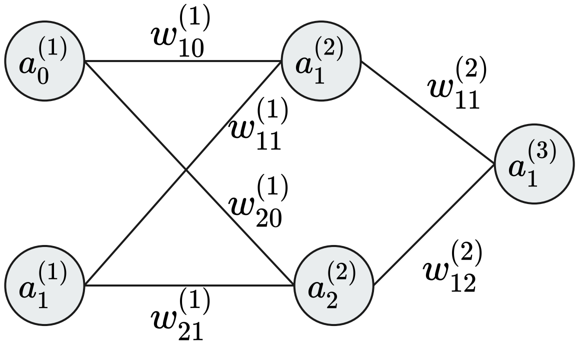
$$\frac{\delta a_1^{(3)}}{\delta w_{11}^{(2)}} = g'\left(z_1^{(2)}\right) \frac{\delta z_1^{(2)}}{\delta w_{11}^{(2)}} \qquad z_1^{(2)} = w_{11}^{(2)} a_1^{(2)} + w_{12}^{(2)} a_2^{(2)}$$

$$\frac{\delta z_1^{(2)}}{\delta w_{11}^{(2)}} = a_1^{(2)}$$

$$\frac{\delta E}{\delta w_{11}^{(2)}} = -\sum_m \underbrace{\left[y - a_1^{(3)}\right] g'\left(z_1^{(2)}\right)}_{\delta_1^{(3)}} a_1^{(2)}$$

Redes neuronales - Entrenamiento

Empecemos con los pesos antes de la capa de salida:

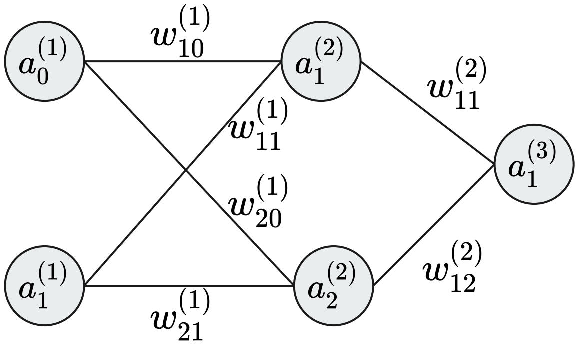


$$\frac{\delta E}{\delta w_{11}^{(2)}} = -\sum_m \delta_1^{(3)} a_1^{(2)}$$

$$\frac{\delta E}{\delta w_{12}^{(2)}} = -\sum_m \delta_1^{(3)} a_2^{(2)}$$

Redes neuronales - Entrenamiento

Ahora sigamos con los pesos de la capa anterior:



$$\frac{\delta E}{\delta w_{10}^{(1)}} = -\sum_m \left[y - a_1^{(3)}\right] \frac{\delta a_1^{(3)}}{\delta w_{10}^{(1)}}$$

$$a_1^{(3)} = g(z_1^{(2)})$$

$$\frac{\delta a_1^{(3)}}{\delta w_{10}^{(1)}} = g'(z_1^{(2)}) \frac{\delta z_1^{(2)}}{\delta w_{10}^{(1)}}$$

$$z_1^{(2)} = w_{11}^{(2)} a_1^{(2)} + w_{12}^{(2)} a_2^{(2)}$$

$$a_1^{(2)} = g(z_1^{(1)}) \qquad a_2^{(2)} = g(z_2^{(1)})$$

$$z_1^{(1)} = w_{10}^{(1)} x_0 + w_{11}^{(1)} x_1 \qquad z_2^{(1)} = w_{20}^{(1)} x_0 + w_{21}^{(1)} x_1$$

0

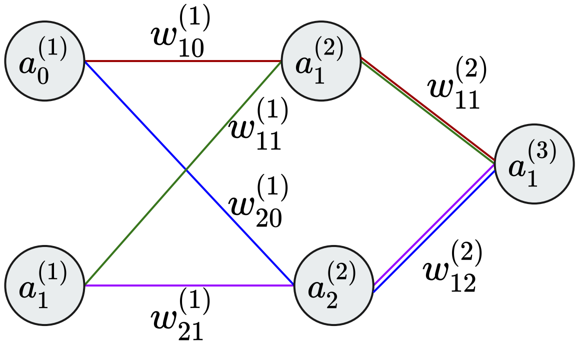
$$\frac{\delta z_1^{(2)}}{\delta w_{10}^{(1)}} = w_{11}^{(2)} * g'(z_1^{(1)}) a_0^{(1)}$$

$$\frac{\delta E}{\delta w_{10}^{(1)}} = -\sum_m \underbrace{\underbrace{\left[y - a_1^{(3)}\right] g'(z_1^{(2)})}_{\delta_1^{(3)}} w_{11}^{(2)} g'(z_1^{(1)})}_{\delta_1^{(2)}} a_0^{(1)}$$

$$\delta_1^{(2)} = \delta_1^{(3)} * w_{11}^{(2)} g'(z_1^{(1)})$$

$$\delta_2^{(2)} = \delta_1^{(3)} * w_{12}^{(2)} g'(z_2^{(1)})$$

Redes neuronales - Backpropagation



$$\frac{\delta E}{\delta w_{11}^{(2)}} = -\sum_m \delta_1^{(3)}\, a_1^{(2)}$$

$$\frac{\delta E}{\delta w_{12}^{(2)}} = -\sum_m \delta_1^{(3)}\, a_2^{(2)}$$

$$\frac{\delta E}{\delta w_{10}^{(2)}} = -\sum_m \delta_1^{(2)}\, a_0^{(1)}$$

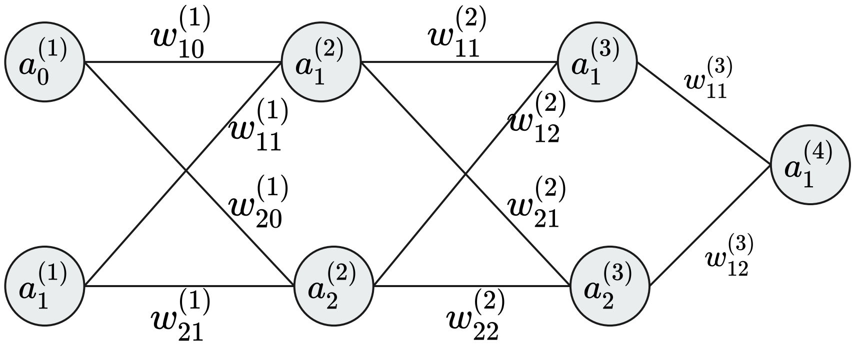
$$\frac{\delta E}{\delta w_{11}^{(2)}} = -\sum_m \delta_1^{(2)}\, a_1^{(1)}$$

$$\frac{\delta E}{\delta w_{20}^{(2)}} = -\sum_m \delta_2^{(2)} a_0^{(1)}$$

$$\frac{\delta E}{\delta w_{21}^{(2)}} = -\sum_m \delta_2^{(2)} a_1^{(1)}$$

Redes neuronales - Backpropagation

Y si tenemos una red más grande?



Para los pesos antes de la capa de salida:

$$\frac{\delta E}{\delta w_{11}^{(3)}} = -\sum_m \delta_1^4 a_1^{(3)}$$

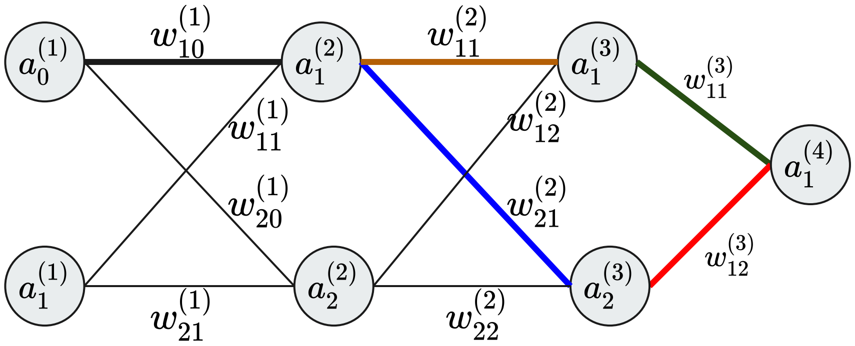
$$\frac{\delta E}{\delta w_{12}^{(3)}} = -\sum_m \delta_1^4 a_2^{(3)}$$

Redes neuronales - Backpropagation

Y si tenemos una red más grande?

$$\frac{\delta E}{\delta w_{10}^{(1)}} = -\sum_m \left[y - a_1^{(4)}\right] \frac{\delta a_1^{(4)}}{\delta w_{10}^{(1)}}$$

$$a_1^{(4)} = g(z_1^{(3)}) = g(w_{11}^{(3)} \underbrace{a_1^{(3)}}_{(1)} + w_{12}^{(3)} \underbrace{a_2^{(3)}}_{(2)})$$



(1) $a_1^{(3)} = g(z_1^{(2)}) = g(w_{11}^{(2)} a_1^{(2)} + w_{12}^{(2)} a_2^{(2)})$

(2) $a_2^{(3)} = g(z_2^{(2)}) = g(w_{21}^{(2)} a_1^{(2)} + w_{22}^{(2)} a_2^{(2)})$

$w_{10}^{(1)}$ sólo afecta a $a_1^{(2)}$

$$a_1^{(2)} = g(z_1^{(1)})$$

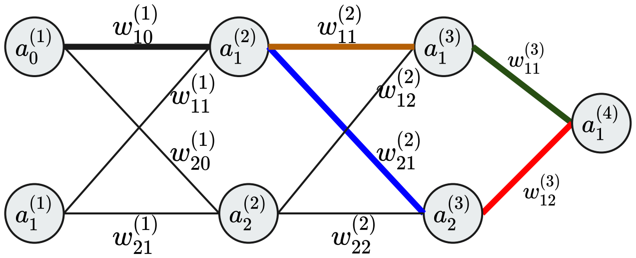
$$z_1^{(1)} = w_{10}^{(1)} x_0 + w_{11}^{(1)} x_1$$

$$\frac{\delta E}{\delta w_{10}^{(1)}} = -\sum_m \left[y - a_1^{(4)}\right] g'(z_1^{(3)}) * [\underline{w_{11}^{(3)} g'(z_1^{(2)}) \frac{\delta z_1^{(2)}}{\delta w_{10}^{(1)}}} + \underline{w_{12}^{(3)} g'(z_2^{(2)}) \frac{\delta z_2^{(2)}}{\delta w_{10}^{(1)}}}]$$

Redes neuronales - Backpropagation

Y si tenemos una red más grande?

$$\frac{\delta E}{\delta w_{10}^{(1)}} = -\sum_m \left[y - a_1^{(4)}\right] g'(z_1^{(3)}) * [\underline{w_{11}^{(3)} g'(z_1^{(2)}) \frac{\delta z_1^{(2)}}{\delta w_{10}^{(1)}}} + \underline{w_{12}^{(3)} g'(z_2^{(2)}) \frac{\delta z_2^{(2)}}{\delta w_{10}^{(1)}}}]$$



$$z_1^{(2)} = w_{11}^{(2)} a_1^{(2)} + w_{12}^{(2)} a_2^{(2)} \qquad z_2^{(2)} = w_{21}^{(2)} a_1^{(2)} + w_{22}^{(2)} a_2^{(2)}$$

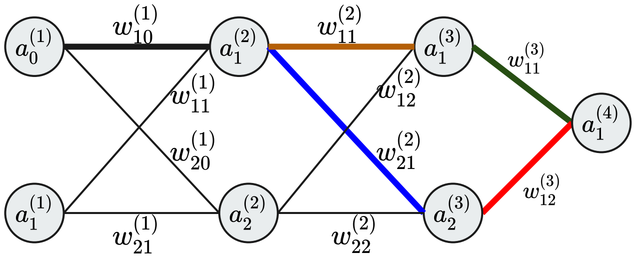
$$\frac{\delta z_1^{(2)}}{\delta w_{10}^{(1)}} = w_{11}^{(2)} g'(z_1^{(1)}) \frac{\delta z_1^{(1)}}{\delta w_{10}^{(1)}} \qquad \frac{\delta z_2^{(2)}}{\delta w_{10}^{(1)}} = w_{21}^{(2)} g'(z_1^{(1)}) \frac{\delta z_1^{(2)}}{\delta w_{10}^{(1)}}$$

$$\frac{\delta z_1^{(2)}}{\delta w_{10}^{(1)}} = w_{11}^{(2)} g'(z_1^{(1)}) a_0^{(1)} \qquad \frac{\delta z_2^{(2)}}{\delta w_{10}^{(1)}} = w_{21}^{(2)} g'(z_1^{(1)}) a_0^{(1)}$$

$$\frac{\delta E}{\delta w_{10}^{(1)}} = -\sum_m \underbrace{\left[y - a_1^{(4)}\right] g'(z_1^{(3)})}_{\delta_1^{(4)}} * [\underbrace{w_{11}^{(3)} g'(z_1^{(2)})}_{\delta_1^{(3)} = \delta_1^{(4)} w_{11}^{(3)} g'(z_1^{(2)})} \underline{w_{11}^{(2)} g'(z_1^{(1)})} a_0^{(1)} + \underbrace{w_{12}^{(3)} g'(z_2^{(2)})}_{\delta_2^{(3)} = \delta_1^{(4)} w_{12}^{(3)} g'(z_2^{(2)})} \underline{w_{21}^{(2)} g'(z_1^{(1)})} a_0^{(1)}]$$

$$\delta_1^{(2)} = \delta_1^{(3)} w_{11}^{(2)} g'(z_1^{(1)}) + \delta_2^{(3)} w_{21}^{(2)} g'(z_1^{(1)})$$

Redes neuronales - Backpropagation



$$\frac{\delta E}{\delta w_{10}^{(1)}} = -\sum_m \delta_1^{(2)} a_0^{(1)}$$

La misma fórmula de la red neuronal más sencilla, solo debemos calcular los deltas acorde al dibujo.

$$\delta_1^{(3)} = \delta_1^{(4)} w_{11}^{(3)} g'(z_1^{(2)}) \qquad \delta_2^{(3)} = \delta_1^{(4)} w_{12}^{(3)} g'(z_2^{(2)})$$

$$\delta_1^{(2)} = \delta_1^{(3)} w_{11}^{(2)} g'(z_1^{(1)}) + \delta_2^{(3)} w_{21}^{(2)} g'(z_1^{(1)})$$