Módulo 3: Aprendizaje Supervisado

3.3. Redes Neuronales (Parte I)

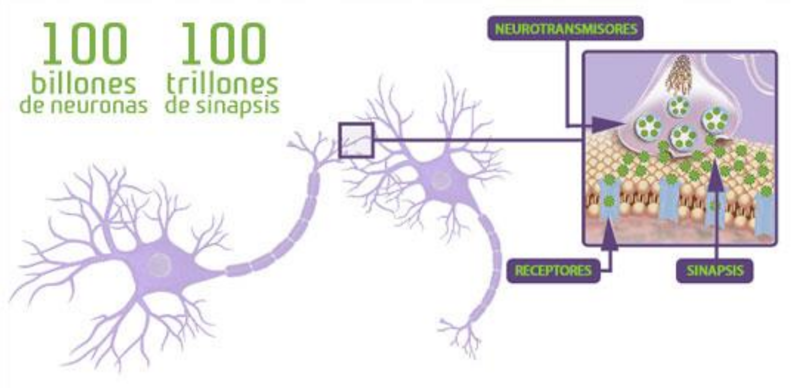
Rafael Zambrano
rafazamb@gmail.com

Redes Neuronales

- Simulan la manera en la que el ser humano procesa la información, a través de neuronas

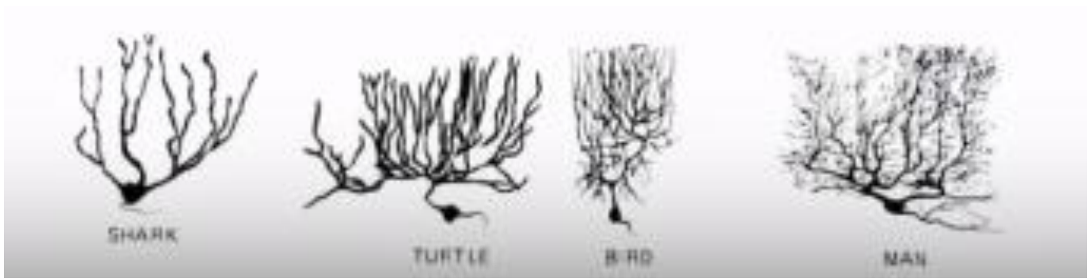
- Una neurona recibe muchas entradas y genera una única salida, que también constituirá una de las muchas entradas de otras neuronas

- La comunicación entre neuronas se realiza a través de una conexión llamada sinapsis, donde se encuentra la memoria



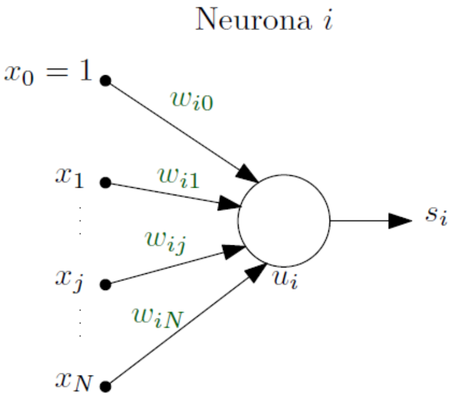
Redes Neuronales

- Entrenar la red neuronal significa alterar sus sinapsis, de forma que aprenda lo que queremos enseñarle.
- Cuando vivimos alguna experiencia, se almacena en nuestro cerebro creando nuevas conexiones, deshaciendo otras, y alterando la ponderación de cada una de esas conexiones.
- Al tratarse de un regresor no lineal, sirven para modelar sistemas no lineales, como simuladores, controladores, clasificadores, reconocimiento de patrones, etc.



Redes Neuronales

- Modelo de neurona:

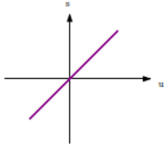


$$u_i = \sum_{j=1}^{N} w_{ij} x_j + w_{i0} = \mathbf{w_i^T x} + w_{i0}$$

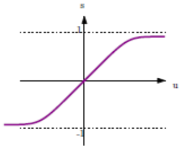
$$s_i = s(u_i)$$

- Función de activación:

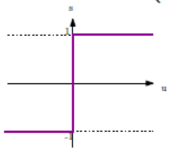
Lineal $s(u) = u$



Tangente hiperbólica $s(u) = tgh(u) = \frac{1-e^{-2u}}{1+e^{-2u}}$

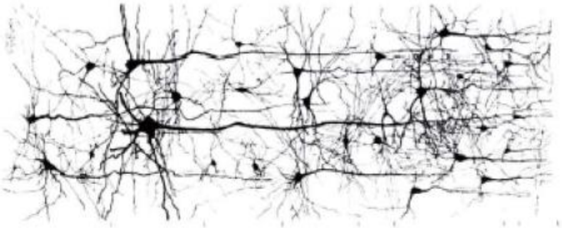


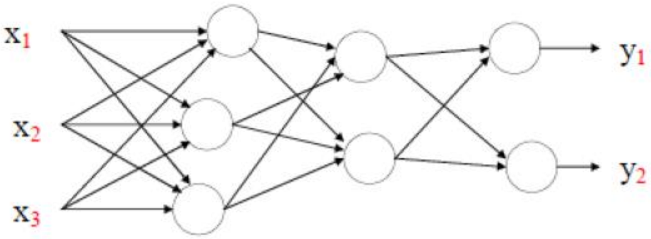
Binaria $s(u) = 0$ si $u < 0; s(u) = 1$ si $u \geq 0$



Redes Neuronales

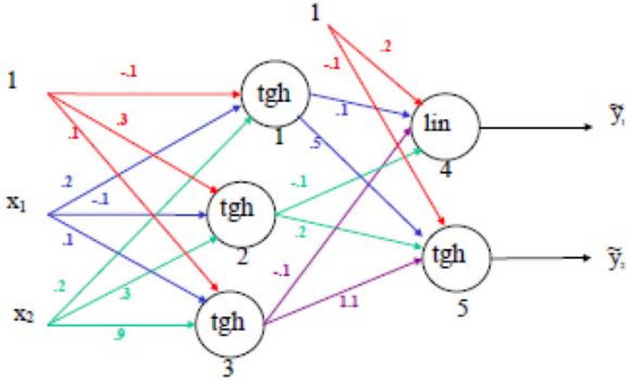
- Red *feedforward*



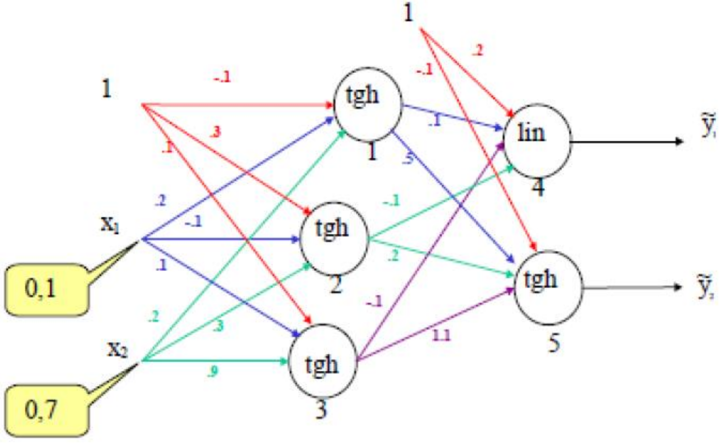


Redes Neuronales

- Ejemplo de operación



$$\underline{\mathbf{x}} = \begin{bmatrix} 0,1 \\ 0,7 \end{bmatrix} \qquad \tilde{\mathbf{y}} = ?$$

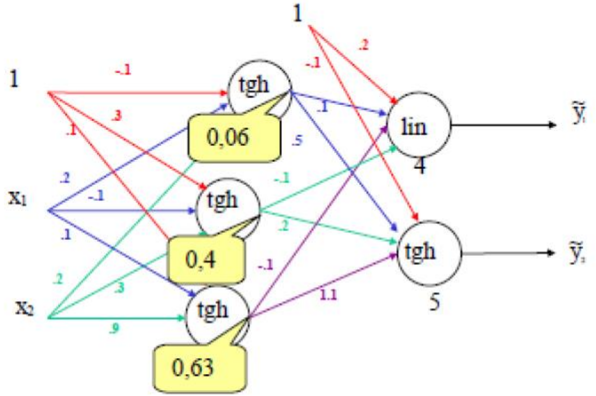


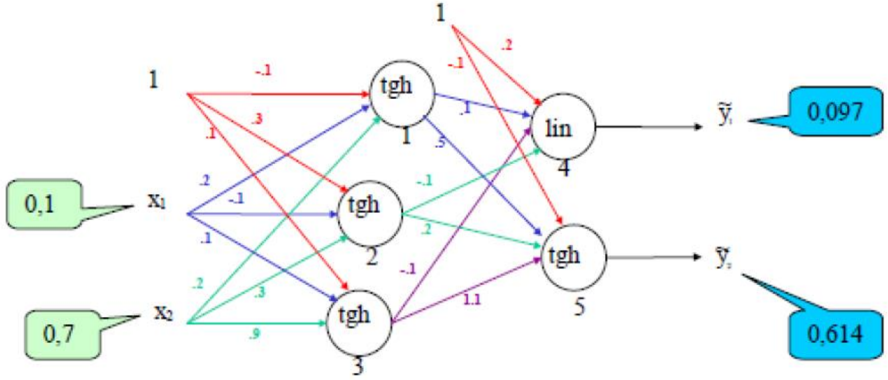
$$u_1 = -0{,}1 + (0{,}2)(0{,}1) + (0{,}2)(0{,}7) = 0{,}06$$

$$v_1 = \mathrm{tgh}(\,0{,}06\,) = 0{,}06$$

Redes Neuronales

- Ejemplo de operación





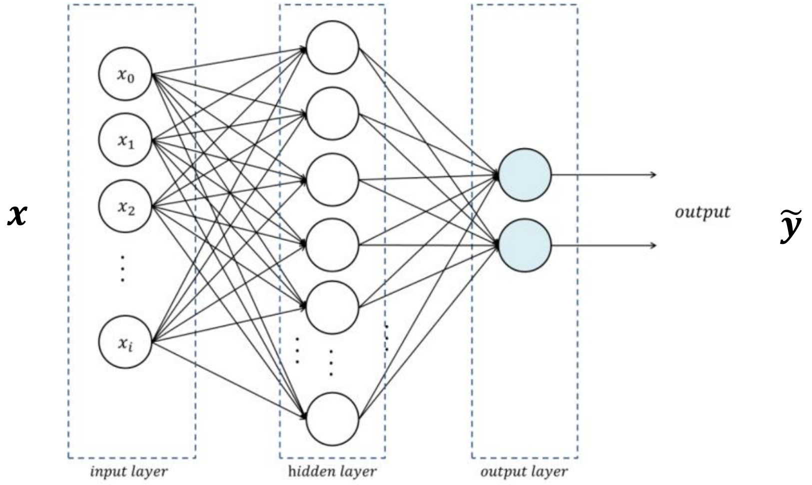
$$u_4 = 0{,}2 + (0{,}1)(0{,}06) + (-0{,}1)(0{,}46) + (-0{,}1)(0{,}63) = 0{,}097$$

$$v_4 = 0{,}097 \quad \text{(linear!)}$$

$$\underline{\mathbf{x}} = \begin{bmatrix} 0{,}1 \\ 0{,}7 \end{bmatrix} \qquad \tilde{\underline{\mathbf{y}}} = \begin{bmatrix} 0{,}097 \\ 0{,}614 \end{bmatrix}$$

Redes Neuronales *feedforward*

- Pueden aproximar relaciones no lineales entre datos de entrada y salida (aproximador universal)



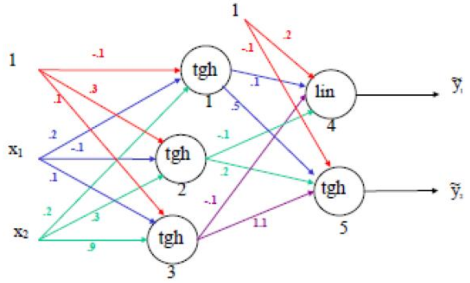
Entrenamiento *backpropagation*

Pares entrada – salida (filas)

$\boldsymbol{x^k}$			$\boldsymbol{y^k}$		
x_1	x_2	…	y_1	y_2	…
2	45	…	3	6	…
…	…	…	…	…	…

$$\boldsymbol{x^1} = \begin{bmatrix} x_1 \\ x_2 \\ \dots \end{bmatrix} = \begin{bmatrix} 2 \\ 45 \\ \dots \end{bmatrix}$$

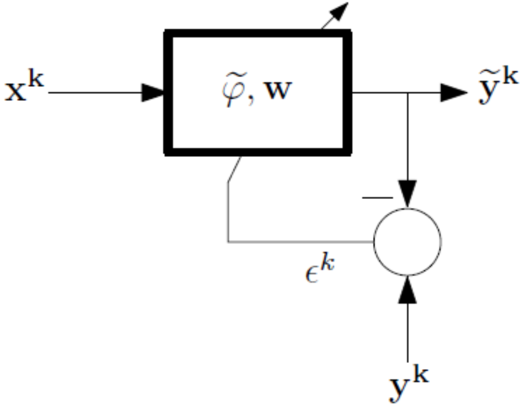
$$\boldsymbol{y^1} = \begin{bmatrix} y_1 \\ y_2 \\ \dots \end{bmatrix} = \begin{bmatrix} 3 \\ 6 \\ \dots \end{bmatrix}$$



$$\epsilon_1 = y_1 - \tilde{y}_1$$

$$\epsilon_2 = y_2 - \tilde{y}_2$$

Aprendizaje: minimización del error en la salida



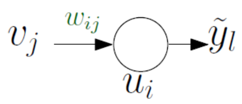
Error a minimizar:

$$\varepsilon^{2^k} = ||\mathbf{y^k} - \tilde{\mathbf{y}}^\mathbf{k}||^2 = \sum_{l=1}^{m} (\mathbf{y}_l^\mathbf{k} - \tilde{\mathbf{y}}_l^\mathbf{k})^2$$ Para las neuronas de salida

$$\epsilon^{2^k} = \epsilon_1^2 + \epsilon_2^2$$

$$F_o = E[\varepsilon^{2^k}] = \frac{1}{P} \sum_{k=1}^{P} \varepsilon^{2^k} = F_o(\mathbf{w}) \geq 0$$ Para todos los pares

Entrenamiento *backpropagation*



- Objetivo: Minimizar $F_o(\boldsymbol{w})$
- Se utiliza el algoritmo del gradiente descendiente: $\boldsymbol{w} \rightarrow \boldsymbol{w} + \Delta\boldsymbol{w}$; $\Delta w_{ij} = -\alpha \frac{\partial F_0}{\partial w_{ij}}$

$$\text{I)}\quad \frac{\partial F_o}{\partial w_{ij}} = \frac{\partial}{\partial w_{ij}} E[\varepsilon^{2^k}] = E[\frac{\partial \varepsilon^{2^k}}{\partial w_{ij}}]$$

$$\text{II)}\quad \frac{\partial \varepsilon^2}{\partial w_{ij}} = \frac{\partial}{\partial w_{ij}} \sum_{l=1}^{m} \varepsilon_l^2 = 2\sum_{l=1}^{m} \varepsilon_l \frac{\partial \varepsilon_l}{\partial w_{ij}} = 2\sum_{l=1}^{m} \varepsilon_l \frac{\partial (y_l - \tilde{y}_l)}{\partial w_{ij}} = -2\sum_{l=1}^{m} \varepsilon_l \frac{\partial \tilde{y}_l}{\partial w_{ij}}$$

$$\boxed{\varepsilon^2 = \sum_{l=1}^{m} \varepsilon_l^2 = \sum_{l=1}^{m} (y_l - \tilde{y}_l)^2}$$

$$= -2\sum_{l=1}^{m} \varepsilon_l \underbrace{\frac{\partial \tilde{y}_l}{\partial u_i}}_{g_{li}} \underbrace{\frac{\partial u_i}{\partial w_{ij}}}_{v_j} = -2\sum_{l=1}^{m} \varepsilon_l \cdot g_{li} \cdot v_j = -2v_j \underbrace{\sum_{l=1}^{m} \varepsilon_l \cdot g_{li}}_{\delta_i} = -2v_j\delta_i$$

$$\text{III)}\quad \Delta w_{ij} = -\alpha \frac{\partial F_o}{\partial w_{ij}} = 2\alpha \frac{1}{P} \sum_{p=1}^{P} v_j \delta_i \rfloor_p = 2\alpha E[v_j \delta_i]$$

Entrenamiento *backpropagation*

- En resumen…

$$v_j \xrightarrow{w_{ij}} \bigcirc_{u_i} \longrightarrow \tilde{y}_l = s(u) = s(v \cdot w)$$

$$\tilde{y} = tgh(u) \Rightarrow \frac{\partial \tilde{y}_l}{\partial u_i} = 1 - \tilde{y}^2$$

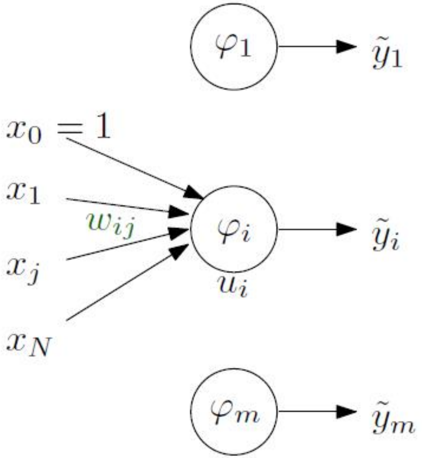
$$\boxed{\Delta w_{ij}^{p} = 2\alpha v_j \delta_i}$$

$$\delta_i = \varepsilon_i \frac{\partial \tilde{y}_l}{\partial u_i}$$

$$v_j = \frac{\partial u_i}{\partial w_{ij}}$$

Redes de una capa

- Para cada par (**x**,**y**)



$$u_i = \sum_{j=0}^{N} w_{ij} x_j$$

$$\tilde{y}_i = \varphi_i(u_i) = \begin{cases} u_i & \text{(neurona lineal)} \\ tgh(u_i) & \text{(neurona tgh)} \end{cases}$$

- Error retropropagado: $\delta_i = \begin{cases} \varepsilon_i \cdot 1 & \text{(neurona lineal)} \\ \varepsilon_i \cdot (1 - \tilde{y}_i^2) & \text{(neurona tgh)} \end{cases}$
- Incremento en cada sinapsis debido al par p: $\Delta w_{ij}^p = 2\alpha x_j \delta_i$

Redes de una capa (ejemplo numérico)

Vamos a implementar una red neuronal con una capa y dos neuronas, la primera de tipo lineal y la segunda de tipo tangente hiperbólica.

Se presenta el par entrada-salida $\{\mathbf{x}; \mathbf{y}\}$, donde $\mathbf{x} = [0.1; 0.7]^T$ e $\mathbf{y} = [0.2; 1]^T$. El valor inicial de las sinapsis es:

$w_{10} = 0.1$

$w_{11} = -0.2$

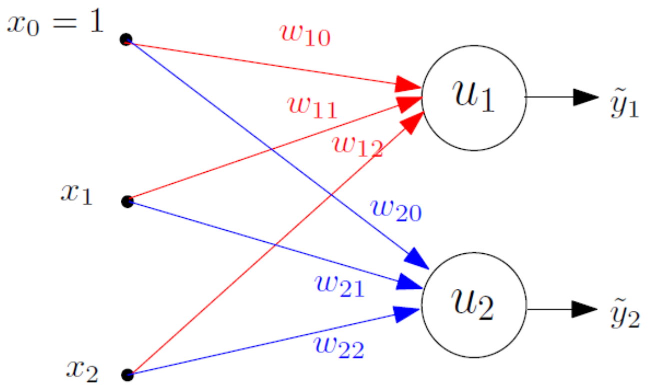
$w_{12} = 0.3$

$w_{20} = 0.5$

$w_{21} = 0.4$

$w_{22} = -0.6$

$\alpha = 0.1$



Vamos a calcular los nuevos valores de las sinapsis tras el primer paso del entrenamiento

Redes de una capa (ejemplo numérico)

Calcula los valores u_1 y u_2

$$u_1 = w_{10} \cdot 1 + w_{11} \cdot x_1 + w_{12} \cdot x_2 = 0.1 \cdot 1 + (-0.2) \cdot 0.1 + 0.3 \cdot 0.7 = 0.29$$

$$u_2 = w_{20} \cdot 1 + w_{21} \cdot x_1 + w_{22} \cdot x_2 = 0.5 \cdot 1 + 0.4 \cdot 0.1 + (-0.6) \cdot 0.7 = 0.12$$

Calcula los valores $\tilde{y}_1$ e $\tilde{y}_2$

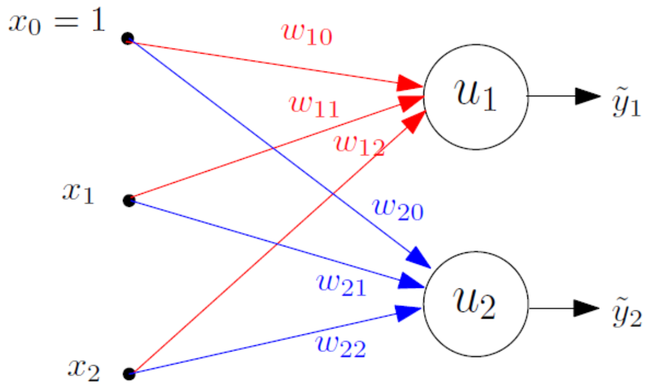
$$\tilde{y}_1 = u_1 = 0.29$$

$$\tilde{y}_2 = tgh(u_2) = tgh(0.12) \approx 0.12$$

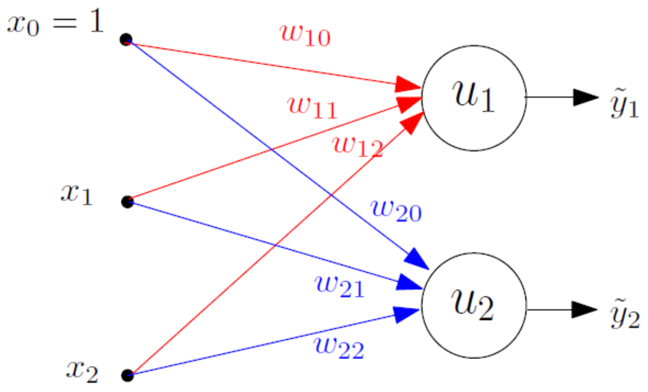
Calcula los errores ε_1 y ε_2

$$\varepsilon_1 = y_1 - \tilde{y}_1 = 0.2 - 0.29 = -0.09$$

$$\varepsilon_2 = y_2 - \tilde{y}_2 = 1 - 0.12 = 0.88$$



Redes de una capa (ejemplo numérico)



Calcular los errores retropropagados δ_1 y δ_2

$\delta_1 = \varepsilon_1 = -0.09$

$\delta_2 = \varepsilon_2(1 - \tilde{y}_2^2) = 0.88 \cdot (1 - 0.12^2) = 0.87$

Calcula el nuevo valor de las sinapsis $w_{ij} \to w_{ij} + \Delta w_{ij}$

$w_{10} \to w_{10} + 2\alpha x_0 \delta_1 = 0.1 + 2 \cdot 0.1 \cdot 1 \cdot (-0.09) = 0.1 - 0.018 = 0.082$

$w_{11} \to w_{11} + 2\alpha x_1 \delta_1 = -0.2 + 2 \cdot 0.1 \cdot 0.1 \cdot (-0.09) = -0.2 - 0.0018 = -0.2018$

$w_{12} \to w_{12} + 2\alpha x_2 \delta_1 = 0.3 + 2 \cdot 0.1 \cdot 0.7 \cdot (-0.09) = 0.3 - 0.0126 = 0.2874$

$w_{20} \to w_{20} + 2\alpha x_0 \delta_2 = 0.5 + 2 \cdot 0.1 \cdot 1 \cdot (0.87) = 0.5 + 0.174 = 0.674$

$w_{21} \to w_{21} + 2\alpha x_1 \delta_2 = 0.4 + 2 \cdot 0.1 \cdot 0.1 \cdot (0.87) = 0.4 + 0.0174 = 0.4174$

$w_{22} \to w_{22} + 2\alpha x_2 \delta_2 = -0.6 + 2 \cdot 0.1 \cdot 0.7 \cdot (0.87) = -0.6 + 0.1218 = -0.4784$

¡Gracias!

Contacto: Rafael Zambrano

rafazamb@gmail.com