Perceptrón multicapa

Alimentación hacia adelante

Verónica E. Arriola-Rios

Facultad de Ciencias, UNAM

14 de octubre de 2020

Funciones no separables linealmente

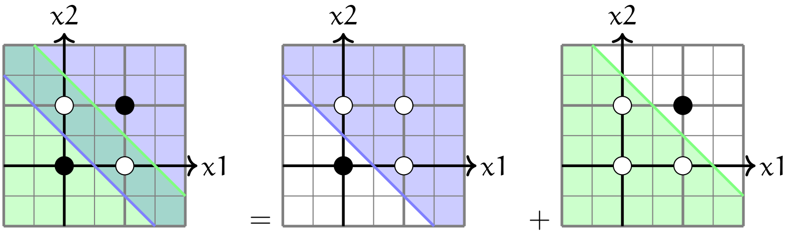
Temas

1. Funciones no separables linealmente
   - XOR
   - Alimentación hacia adelante

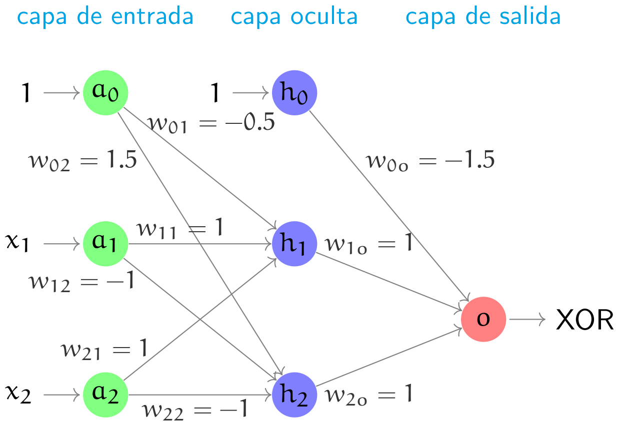
XOR (Minsky & Papert, 1969 → Rumerhart et al. 1986)

- XOR es una compuerta lógica con valores no separables linealmente en el plano.
- Se puede escribir como:

$$\text{XOR} = (x_1 \vee x_2) \wedge \neg(x_1 \wedge x_2) = (x_1 \vee x_2) \wedge (x_1 \text{ NAND } x_2) \tag{1}$$



XOR



Con $W_{<origen><destino>}$.

Temas

1. Funciones no separables linealmente
    - XOR
    - Alimentación hacia adelante

XOR evaluación de la neurona

Para la capa oculta: $$z_j = g\left(\sum_i w_{ij} a_i\right) \qquad h_j = \frac{1}{1 + e^{-z}}$$

En el ejemplo:
$$h_0 \equiv 1$$
$$h_1 = g(-0.5a_0 + 1a_1 + 1a_2)$$
$$h_2 = g(1.5a_0 + -1a_1 + -1a_2)$$

Para la capa de salida: $$z_o = g\left(\sum_j w_{jo} h_j\right) \qquad o = \frac{1}{1 + e^{-z_o}}$$

$$o = g(-1.5h_0 + 1w_{1o} + 1w_{2o})$$

XOR evaluación de la neurona (ejemplo convensión 1)

$$A = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

$$W^{(1)} = \begin{bmatrix} w_{01} & w_{11} & w_{21} \\ w_{02} & w_{12} & w_{22} \end{bmatrix} = \begin{bmatrix} -0.5 & 1 & 1 \\ 1.5 & -1 & -1 \end{bmatrix}$$

$$H = g(W^{(1)}A)$$

$$= \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} = g\left(\begin{bmatrix} -0.5 & 1 & 1 \\ 1.5 & -1 & -1 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}\right) = g\left(\begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}\right) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

XOR evaluación de la neurona (ejemplo convensión 2)

$$A = \begin{bmatrix} a_0 & a_1 & a_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \end{bmatrix}$$

$$W^{(1)} = \begin{bmatrix} w_{01} & w_{02} \\ w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix} = \begin{bmatrix} -0.5 & 1.5 \\ 1 & -1 \\ 1 & -1 \end{bmatrix}$$

$$H = g(AW^{(1)})$$

$$\begin{bmatrix} h_1 & h_2 \end{bmatrix} = g\left(\begin{bmatrix} 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} -0.5 & 1.5 \\ 1 & -1 \\ 1 & -1 \end{bmatrix} \right) = g\left(\begin{bmatrix} 0.5 & 0.5 \end{bmatrix} \right) = \begin{bmatrix} 1 & 1 \end{bmatrix}$$

$$H' = \begin{bmatrix} h_0 & h_1 & h_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \qquad W^{(2)} = \begin{bmatrix} w_{0o} \\ w_{1o} \\ w_{2o} \end{bmatrix} = \begin{bmatrix} -1.5 \\ 1 \\ 1 \end{bmatrix}$$

$$O = [o] = g\left(\begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} -1.5 \\ 1 \\ 1 \end{bmatrix} \right) = g(\begin{bmatrix} 0.5 \end{bmatrix}) = [1]$$

Evaluando sobre varias entradas

¿Qué pasa ahora si queremos evaluar la red sobre varias entradas?

$$X = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{bmatrix} \qquad \text{con sesgos} \rightarrow \qquad X' = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

Veamos sólo la primera capa:

$$H = g(X'W^{(1)}) = g\left(\begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} -0.5 & 1.5 \\ 1 & -1 \\ 1 & -1 \end{bmatrix} \right) = g\left(\begin{bmatrix} -0.5 & 1.5 \\ 0.5 & 0.5 \\ 0.5 & 0.5 \\ 1.5 & -0.5 \end{bmatrix} \right) = \begin{bmatrix} 0 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 0 \end{bmatrix}$$

Entonces, H^T contiene las activaciones para todas las entradas.

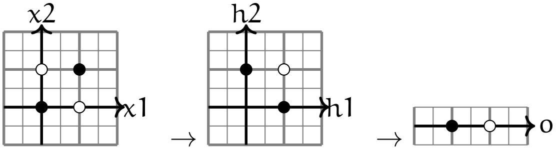
Neuronas como mapeos entre espacios

Observemos la tabla de entradas para la compuerta XOR.

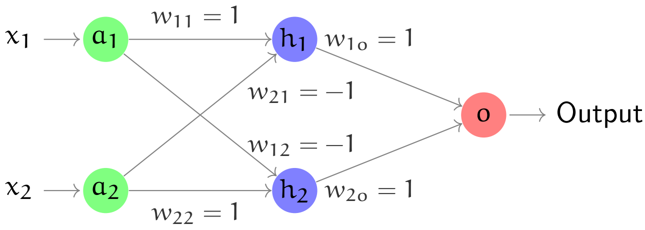
x_1	x_2	h_1	h_2	o
0	0	0	1	0
0	1	1	1	1
1	0	1	1	1
1	1	1	0	0

Podemos interpretar cada capa de la neurona como una función que transforma espacios de varias dimensiones. En este caso:

$$\mathbb{R}^2 \to \mathbb{R}^2 \to \mathbb{R} \tag{2}$$

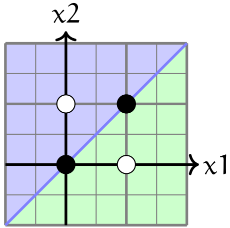


XOR (Utilizando la función umbral)



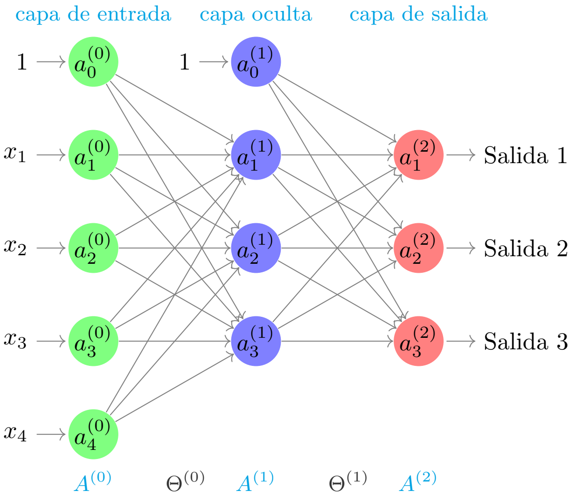
$$g = \begin{cases} 0 & x \leqslant 0 \\ 1 & x > 0 \end{cases}$$

x_1	x_2	z_1	h_1	z_2	h_2	z	o
0	0	0	0	0	0	0	0
0	1	-1	0	1	1	1	1
1	0	1	1	-1	0	1	1
1	1	0	0	0	0	0	0



Modelo computacional de una red neuronal

Red neuronal



One-hot encoding

Por ejemplo:

$$\text{Salida 1} = coche = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{3}$$

$$\text{Salida 2} = casa = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{4}$$

$$\text{Salida 3} = vaca = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{5}$$

Valor de una neurona (propagación hacia adelante)

$$a_j^{(l+1)} = g\left(\sum_i w_{ij}^{(l)} a_i^{(l)}\right) \quad (6)$$

$$g = \frac{1}{1+e^{-x}} \quad (7)$$

$$(8)$$

Para una sola salida:

$$\begin{bmatrix} a_0^{(l+1)} \\ a_1^{(l+1)} \\ \dots \\ a_{n'}^{(l+1)} \end{bmatrix} = g\left(\begin{bmatrix} w_{01}^{(l)} & \dots & w_{n1}^{(l)} \\ \dots & & \\ w_{0n'}^{(l)} & \dots & w_{nn'}^{(l)} \end{bmatrix} \begin{bmatrix} a_0^{(l)} \\ a_1^{(l)} \\ \dots \\ a_n^{(l)} \end{bmatrix}\right) \quad (9)$$

Para varios ejemplares en paralelo:

$$A^{(l+1)} = g(A^{(l)}W) \tag{10}$$

$$\begin{bmatrix} a_0^{(l+1)} & a_1^{(l+1)} & \dots & a_{n'}^{(l+1)} \end{bmatrix} = g\left(\begin{bmatrix} a_0^{(l)} & a_1^{(l)} & \dots & a_n^{(l)} \end{bmatrix} \begin{bmatrix} w_{01}^{(l)} & \dots & w_{0n'}^{(l)} \\ \dots & & \\ & & \\ w_{n1}^{(l)} & \dots & w_{nn'}^{(l)} \end{bmatrix} \right) \tag{11}$$

Referencias I

- Haykin, Simon (2009). *Neural Networks and Learning Machines*. 3rd. Prentice Hall, Pearson.
- *Machine Learning*, Andrew NG, https://www.coursera.org/learn/machine-learning

Licencia