Institución Universitaria
Reacreditada en Alta Calidad
80 Años

Redes Neuronales I

Motivación

Redes Neuronales: Inspiradas por el cerebro

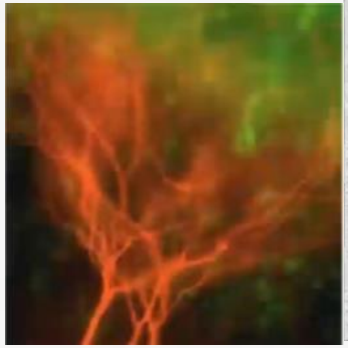
https://www.verywellmind.com/how-brain-cells-communicate-with-each-other-2584397

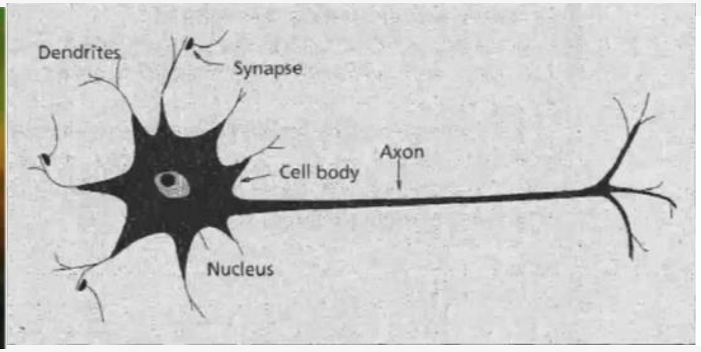
Redes Neuronales: Inspiradas por el cerebro

https://medium.com/@adsactly/is-it-a-bird-is-it-a-plane-biomimicry-in-airplanes-9862d331df2e

Primeras redes

Modelo de Neurona de McCulloch Pitts

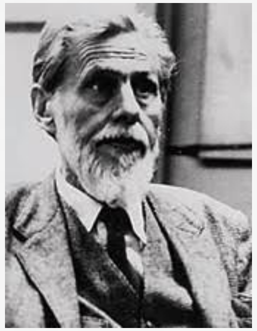


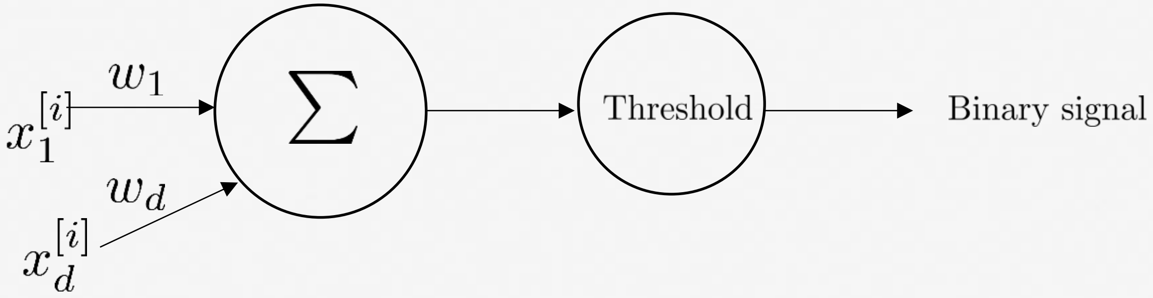


A LOGICAL CALCULUS OF THE IDEAS IMMANENT IN NERVOUS ACTIVITY

WARREN S. McCULLOCH and WALTER H. PITTS 1943

$$z^{[i]} = \mathbf{w} \cdot \mathbf{x}^{[i]}$$

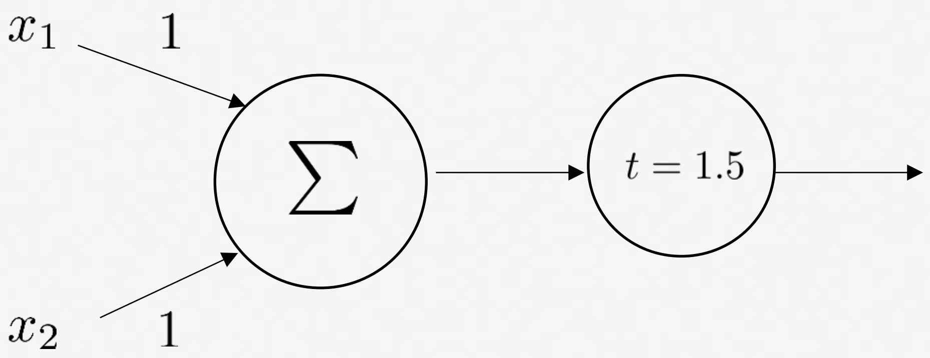




Compuertas lógicas

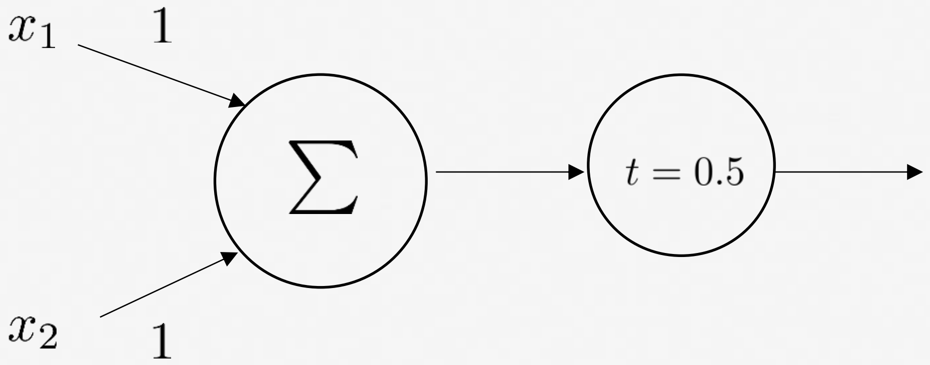
And

x_1	x_2	Out
0	0	0
0	1	0
1	0	0
1	1	1



Or

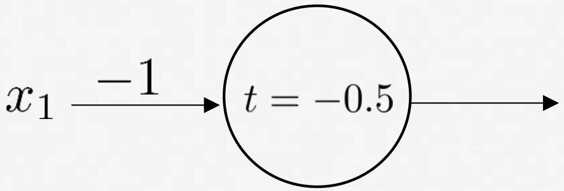
x_1	x_2	Out
0	0	0
0	1	1
1	0	1
1	1	1



Compuertas lógicas

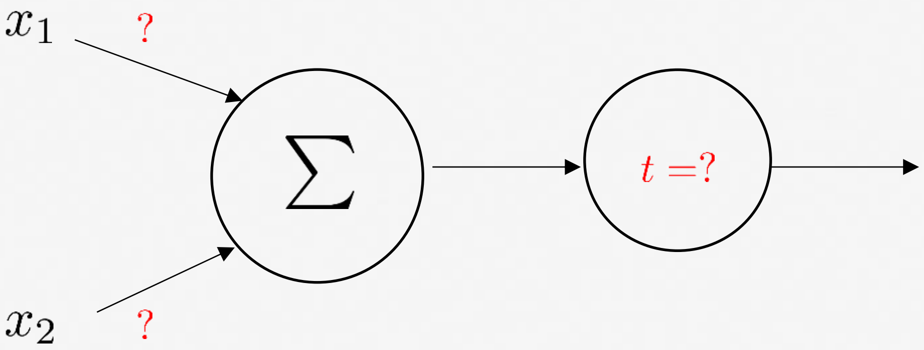
Not

x_1	Out
0	1
1	0



Xor

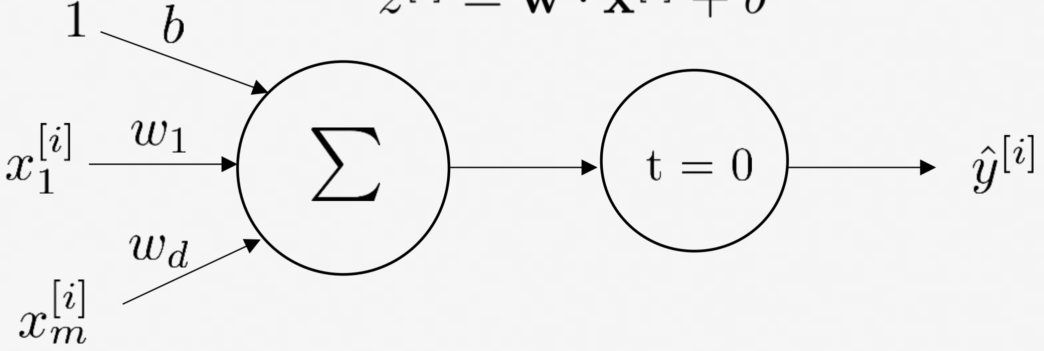
x_1	x_2	Out
0	0	0
0	1	1
1	0	1
1	1	0

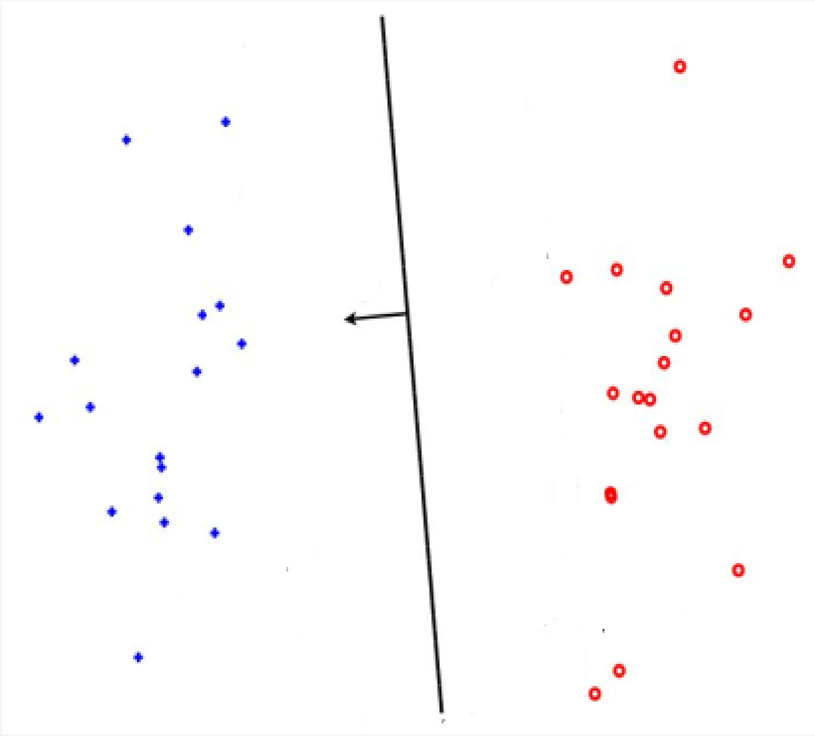


Perceptrón

Establece una regla de aprendizaje para el modelo de neurona

$$z^{[i]} = \mathbf{w} \cdot \mathbf{x}^{[i]} + b$$





$$\mathbf{x}^{[i]} = [1, x_1^{[i]}, \ldots, , x_m^{[i]}] \qquad w_0 = b$$

$$z^{(i)} = \mathbf{w} \cdot \mathbf{x}^{[i]}$$

Perceptrón: Algoritmo

Sea

$$\mathcal{D} = ((\mathbf{x}^{[1]}, y^{[1]}), (\mathbf{x}^{[2]}, y^{[2]}), \ldots, (\mathbf{x}^{[n]}, y^{[n]})) \in (\Re^m \times \{0, 1\})^n$$

Result: $\mathbf{w}, b$
$\mathbf{w} := \mathbf{0} \in \Re^m, \mathbf{b} := 0;$
for $t = 1, \ldots, T$ **do**
  for $i = 1, \ldots, n$ **do**
    cálculo de la salida $\hat{y}^{[i]} := \sigma(\mathbf{x}^{[i]\top}\mathbf{w} + b);$
    cálculo del error $err := (y^{[i]} - \hat{y}^{[i]});$
    actualización de parámetros $\mathbf{w} := \mathbf{w} + err \times \mathbf{x}^{[i]}$, $b := b + err;$
  end
end

Pasos Comunes de Aprendizaje

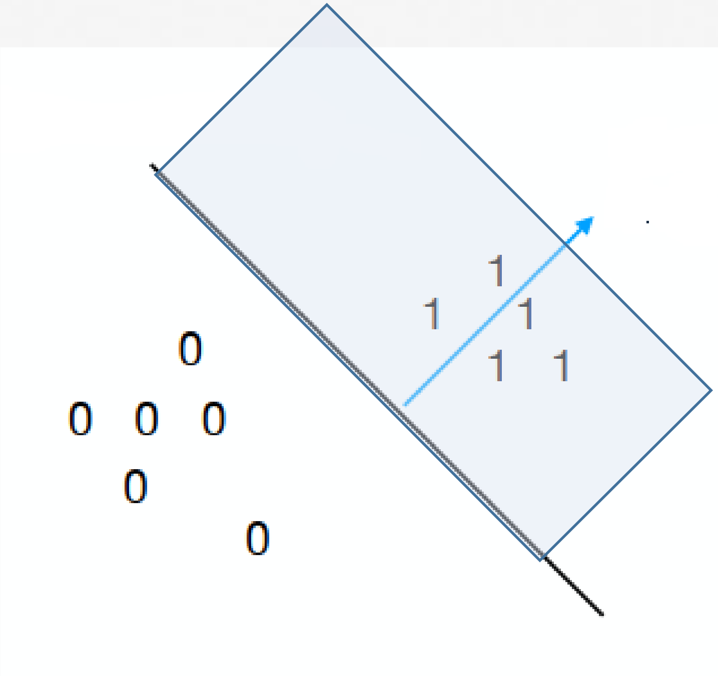
$$\mathcal{D} = \{(\mathbf{x}^{[i]}, y^{[i]}) | i = 1, \dots, n\} \quad \mathbf{x}^{[i]} \in \Re^{m+1} \;\; y^{[i]} \in \{0, 1\}$$

Por cada dato $\left(x^{[i]}, y^{[i]}\right)$:

i) Hacer una predicción
ii) Calcular el error
iii) Actualizar los pesos basado en el error

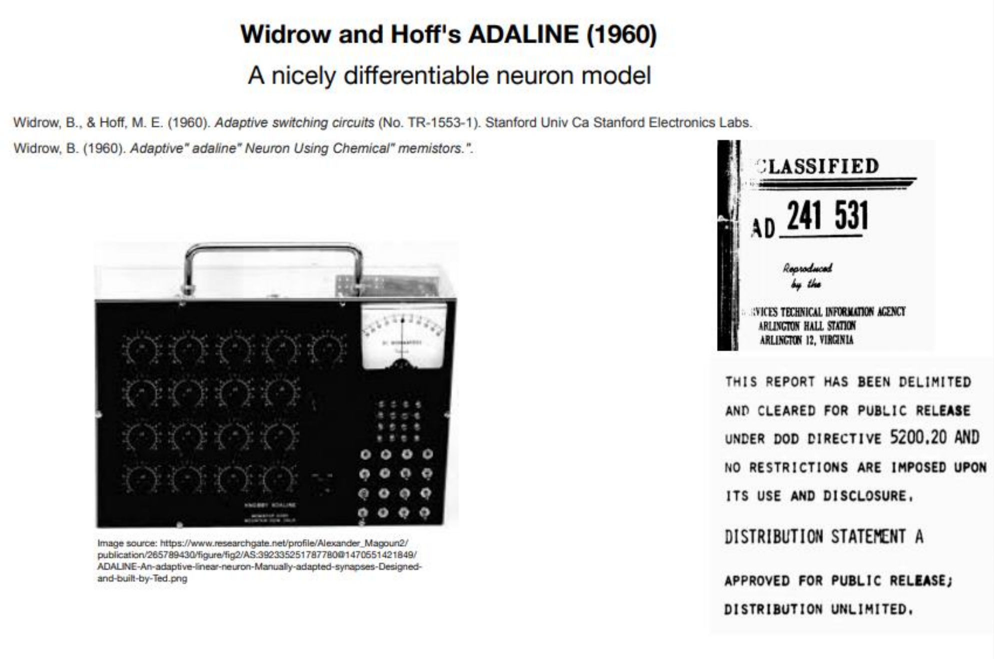
Perceptrón: Interpretación geométrica

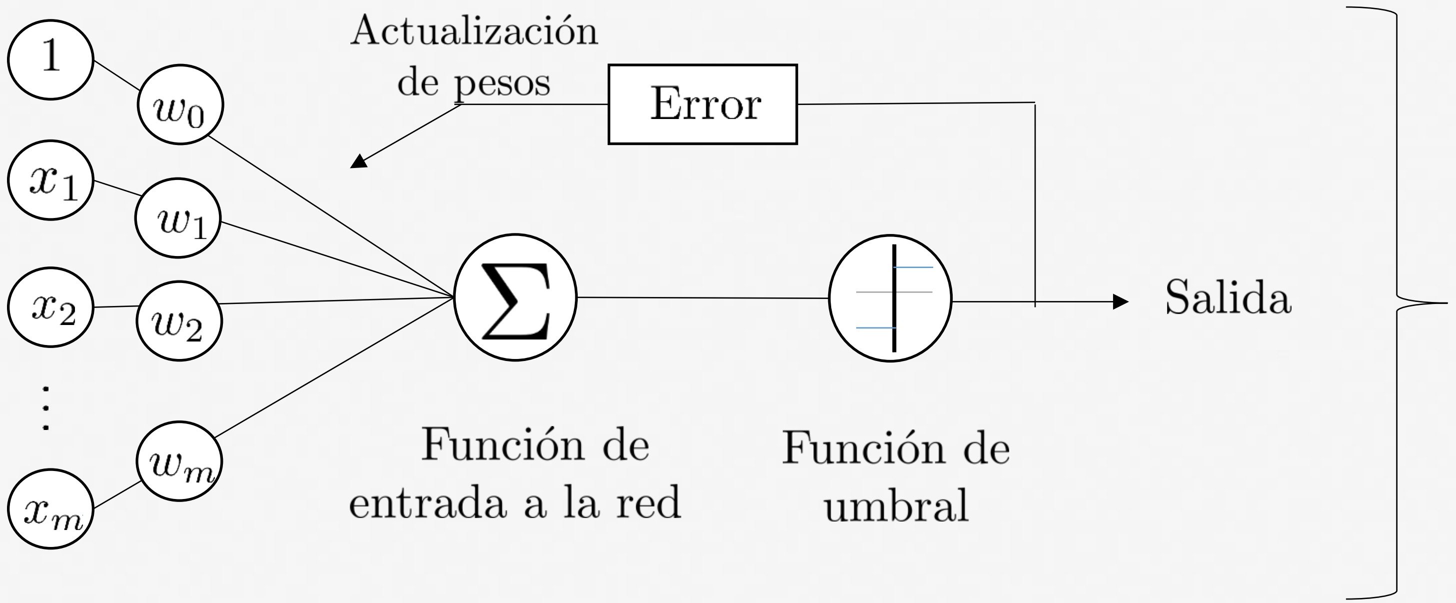
$\mathbf{w} \perp$ boundary



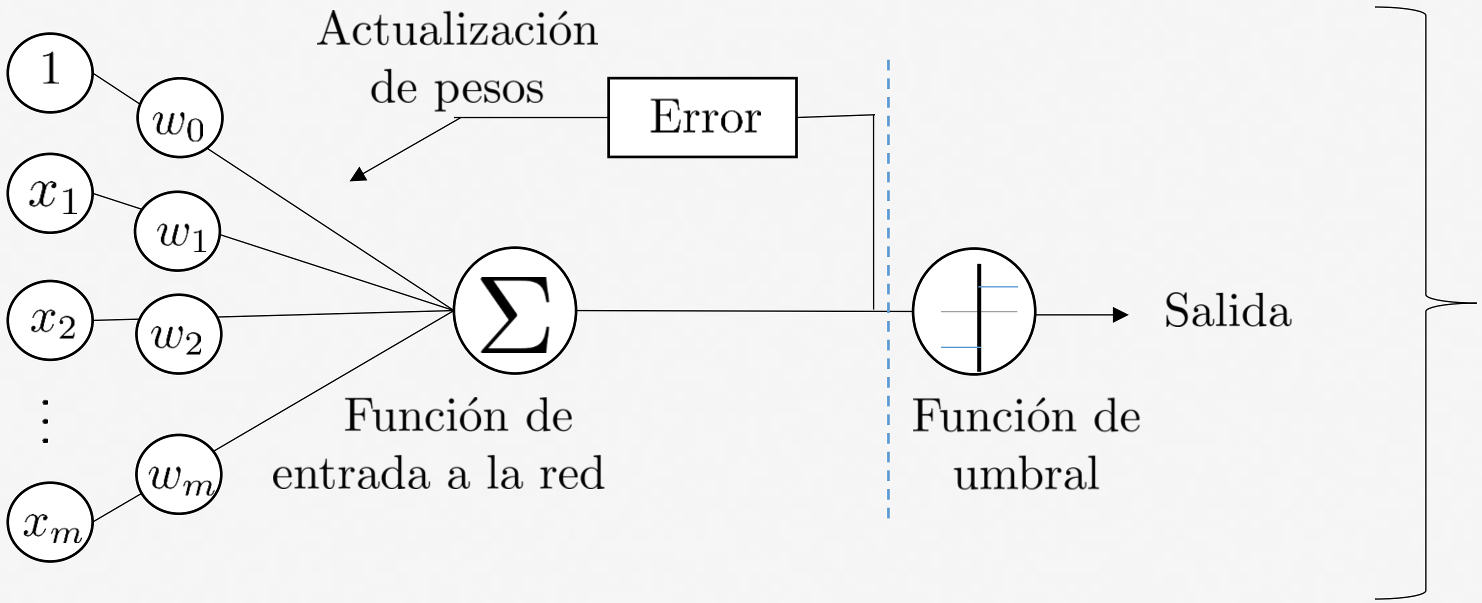
$$\hat{y}^{[i]} = \begin{cases} 0 & \mathbf{w} \cdot \mathbf{x}^{[i]} \leq 0 \\ 1 & \mathbf{w} \cdot \mathbf{x}^{[i]} > 0 \end{cases}$$

Entrenamiento de una red neuronal de una sola capa con gradiente descendente





Perceptrón

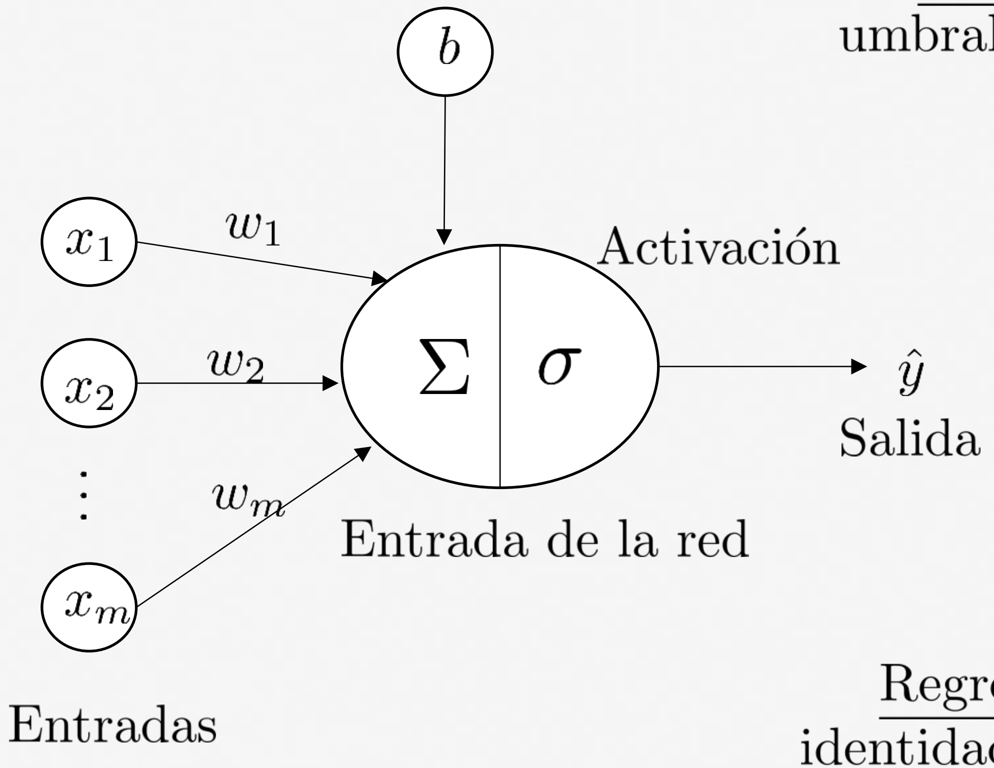


ADAptive LInear NEuron

Regresión Lineal como una red neuronal

Regresión lineal

Perceptrón: la función de activación es la función de umbral. La salida es la etiqueta binaria $\hat{y} \in \{0,1\}$



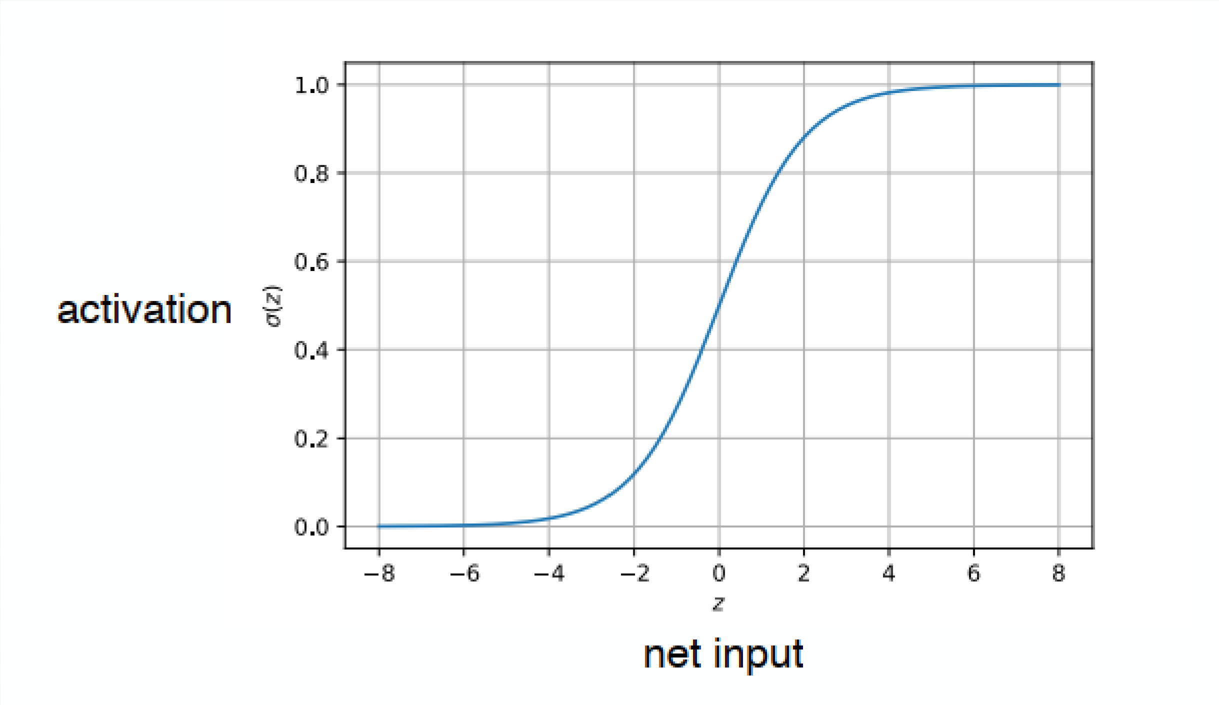
Regresión lineal: la función de activación es la función identidad $\sigma(x) = x$. La salida es el número $\hat{y} \in \Re$

Regresión Logística

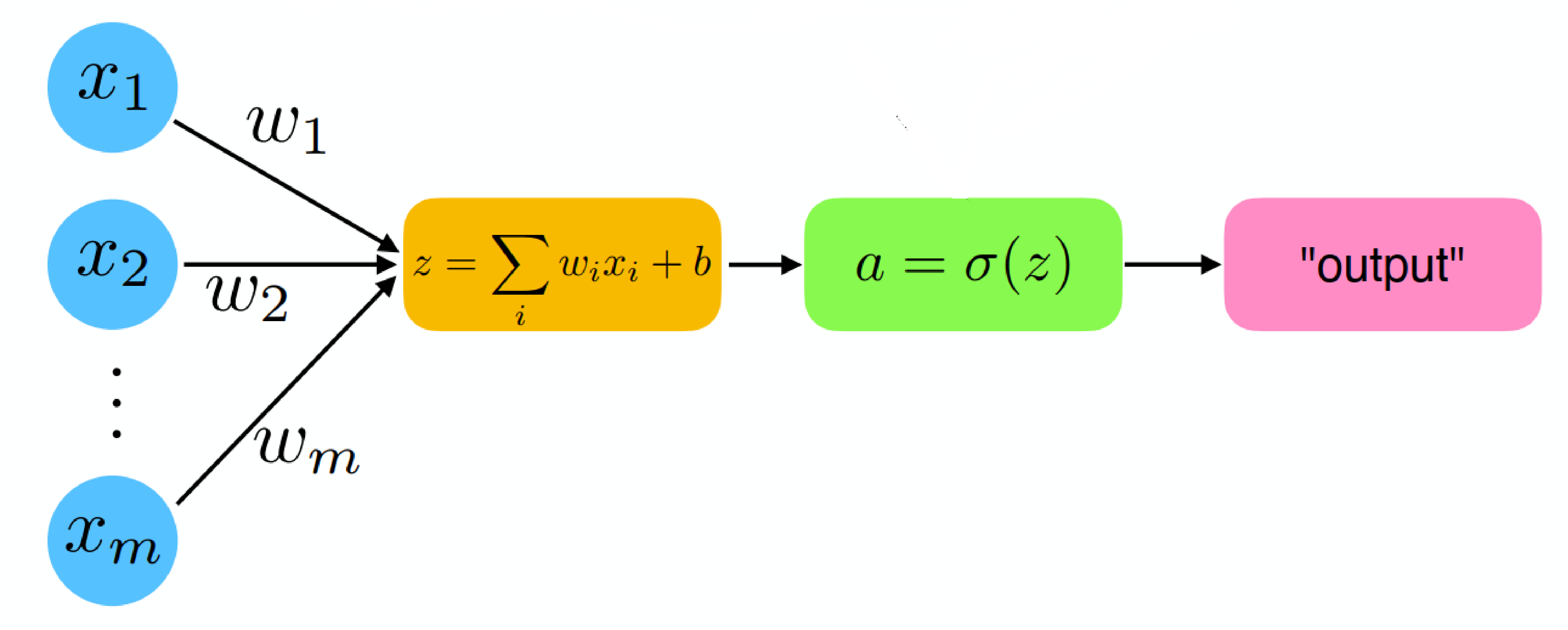
Regresión Logística

Sigmoide

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



Regresión logística para problemas bi-clase $y^{[i]} \in \{0, 1\}$



Para el sigmoide se tiene

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

Regresión logística para problemas bi-clase $y^{[i]} \in \{0, 1\}$

- En ADALINE, la función de activación era una función identidad

$$\sigma(z) = z$$

- Se utilizaba MSE como costo

$$MSE = \frac{1}{n} \sum_{i} \left(y^{[i]} - a^{[i]} \right)^2$$

- Para regresión logística, se utilizará una función de costo diferente

Regresión logística

Dada la salida

$$h(\mathbf{x}) = \sigma\left(\mathbf{w}^{\top}\mathbf{x} + b\right)$$

Donde $h(\mathbf{x})$ o hipótesis es la salida de la función de activación

$$h(\mathbf{x}) = a$$

Podemos calcular la probabilidad posterior o *a posteriori* como

$$P(y|\mathbf{x}) = \begin{cases} h(\mathbf{x}) & y = 1 \\ 1 - h(\mathbf{x}) & y = 0 \end{cases}$$

Institución Universitaria
Reacreditada en Alta Calidad
80 Años

¡MUCHAS GRACIAS!