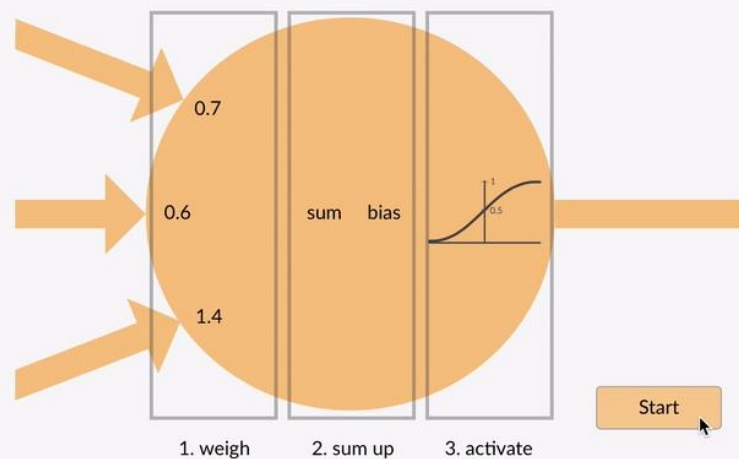




Inteligencia Artificial



Unidad 2: Redes Neuronales

TEMA 3: Algoritmos de IA Moderna-I

Módulo 2: Desde el Perceptrón a las Redes Neuronales

Unidad 2

Redes Neuronales

TEMA 3: Algoritmos de IA Moderna-I

Sesión 11-12

MÓDULO 2: Desde el Perceptrón a las Redes Neuronales



Contenido

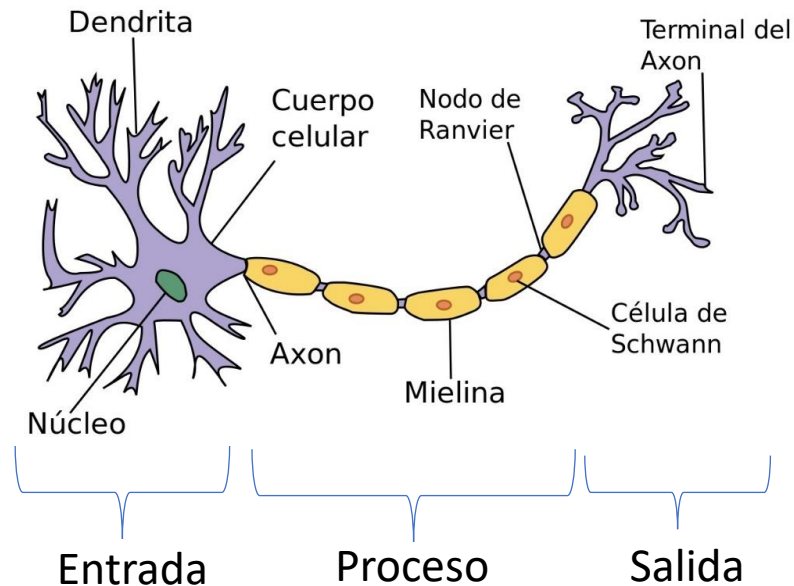
1. Perceptrón vs. Red Neuronal Artificial
2. Estructura de una Red Neuronal Artificial
3. Modelo Estándar de una Red Neuronal Artificial
4. El Perceptrón Simple
5. Redes ADALINE
6. Ventajas / Limitaciones



Preguntas

1. Perceptrón vs. Red Neuronal Artificial

Modelo Biológico: una neurona



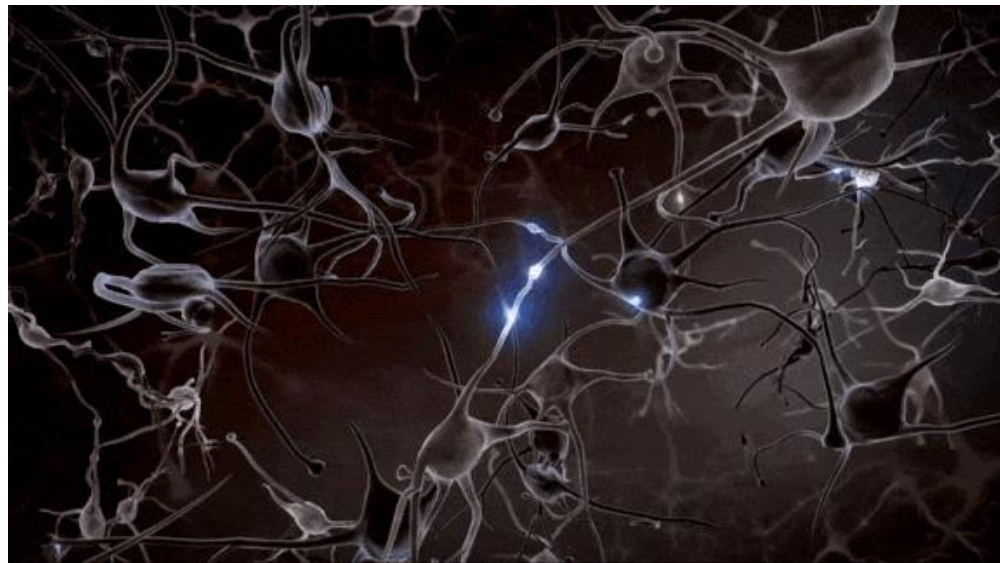
De alguna forma, una neurona es un procesador de información muy simple:

- **Canal de entrada:** dendritas.
- **Procesador:** soma.
- **Canal de salida:** axón.

Las neuronas son las células del sistema nervioso encargadas de transmitir información mediante impulsos nerviosos.

1. Perceptrón vs. Red Neuronal Artificial

Modelo Biológico: Una Red Neuronal

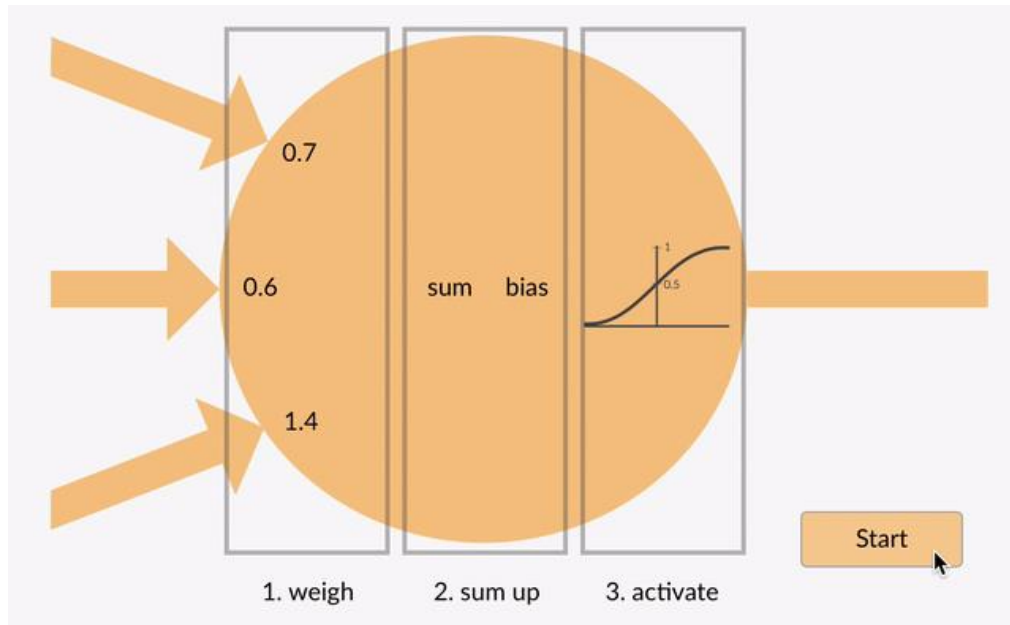


Una **red neuronal** es la unión entre **dos o más neuronas**, estas se relacionan entre si y pueden transmitir la información de manera eficaz.

1. Perceptrón vs. Red Neuronal Artificial

Neurona Artificial

- Es la emulación a través de algoritmos del comportamiento de las neuronas humanas.
- A una sola neurona artificial se le conoce como **Perceptrón**.



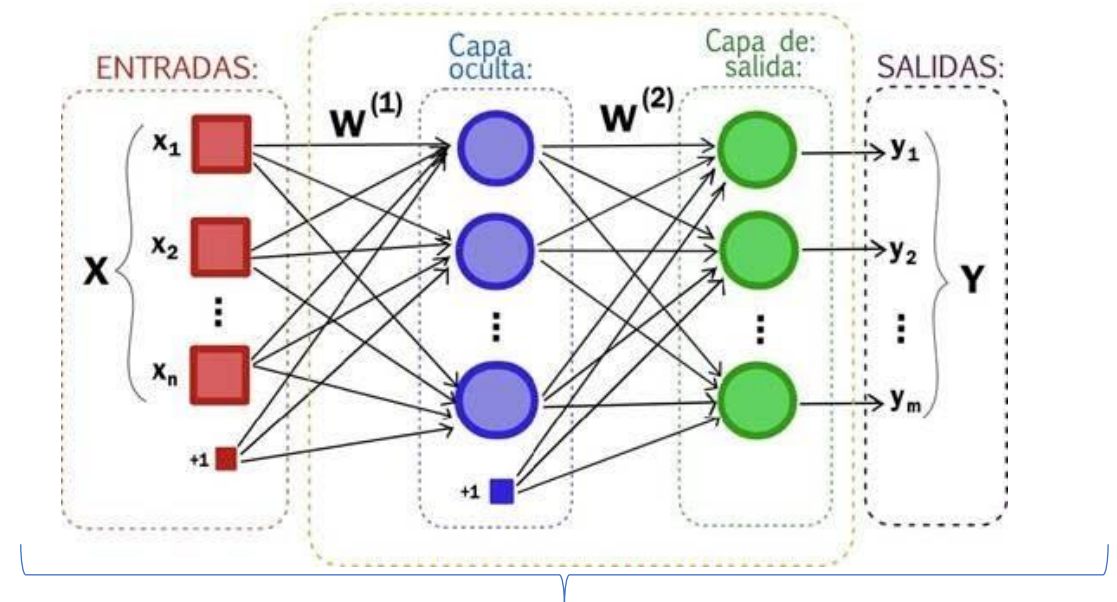
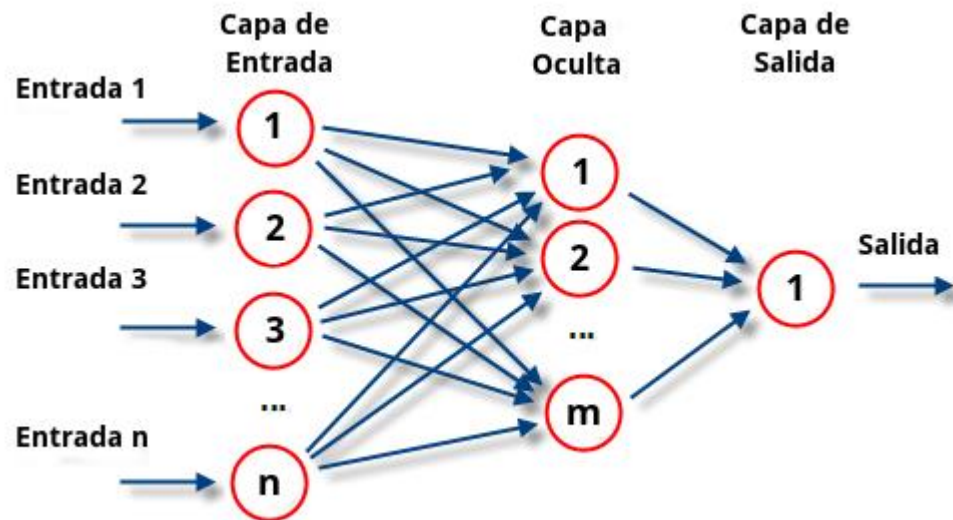
Estructura de un Perceptrón (neurona artificial)

- Un solo **perceptrón** (o neurona artificial) se puede imaginar como una regresión logística.
- Una **regresión logística** es un método estadístico utilizado para predecir clases binarias (o dicotómicas).

1. Perceptrón vs. Red Neuronal Artificial

Red Neuronal Artificial

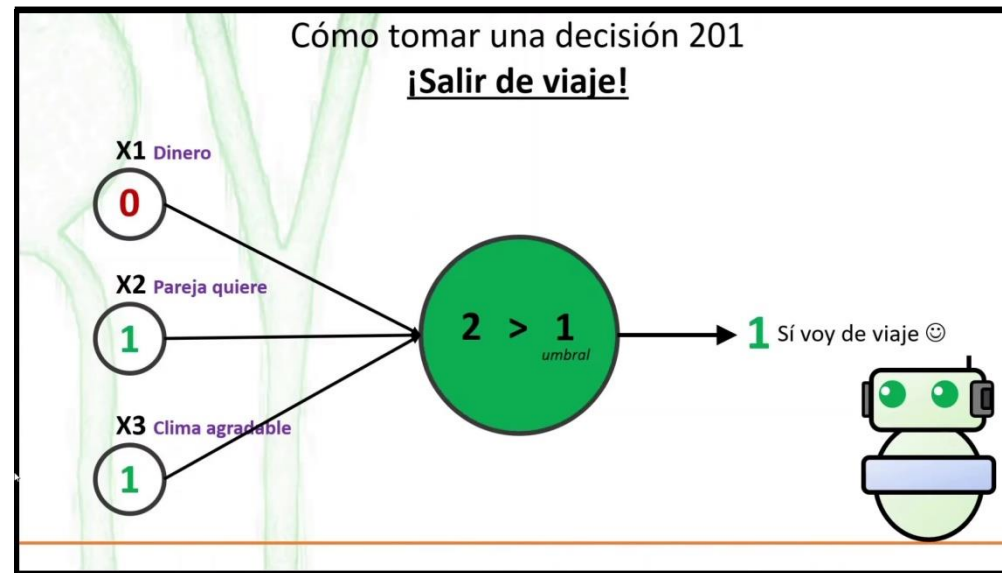
- Las redes neuronales artificiales son un modelo inspirado en el funcionamiento del cerebro humano.
- Una RNA (o ANN por sus siglas en inglés) esta formada por un conjunto de nodos conocidos **como neuronas artificiales o perceptrones** que están conectadas y transmiten señales entre sí (están agrupadas en capas y conectadas mediante vínculos ponderados).
- Estas señales se transmiten desde la entrada hasta generar una salida.



Sistema global de proceso de una Red Neuronal Artificial

2. Estructura de una Red Neuronal Artificial

Red Neuronal Artificial: Cómo aprenden los perceptrones?
El proceso de toma de decisiones



Enlace: <https://youtu.be/T-zSe44XCEk> (Duración: 10 min.)

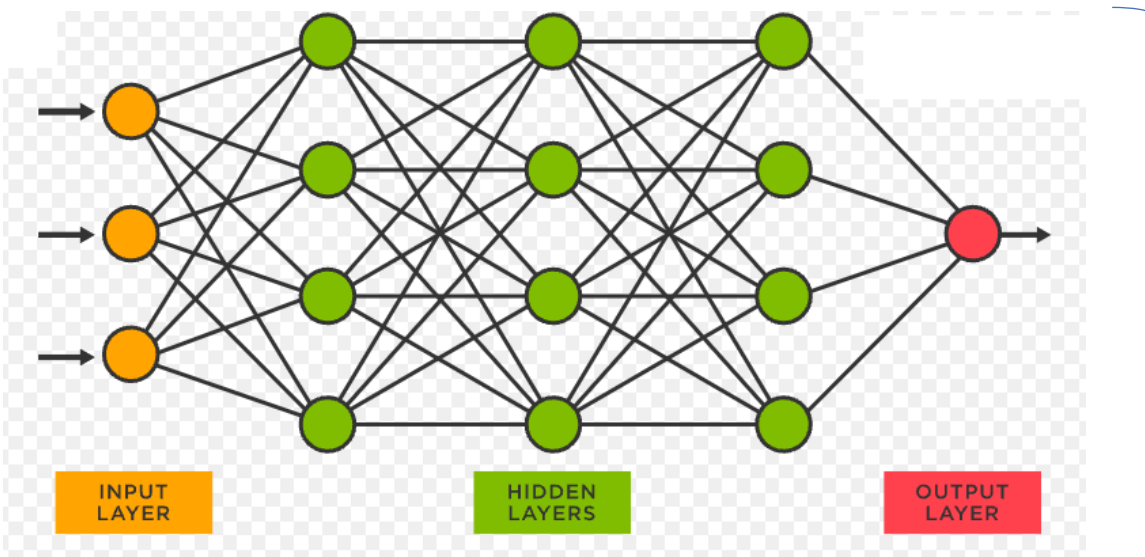
2. Estructura de una Red Neuronal Artificial

Red Neuronal Artificial

OBJETIVO: El objetivo principal de este modelo es **APRENDER** modificándose automáticamente a si mismo de forma que puede llegar a realizar tareas complejas que no podrían ser realizadas mediante la clásica programación basada en reglas.

¿Cómo funciona?

- **ANN** también se conoce como una red neuronal **Feed-Forward** porque las entradas se procesan solo en la dirección de avance:



- ANN consta de 3 capas: Entrada, Oculta y Salida.
 - **La capa de entrada:** acepta los valores de los datos iniciales
 - **La capa oculta:** procesa las entradas
 - **La capa salida:** produce el resultado
- Esencialmente, **cada capa intenta aprender ciertos pesos.**

3. Modelo Estándar de una Red Neuronal Artificial

- El denominado **modelo estándar de neurona artificial** (modelo conexionista) se desarrolló según los principios descritos en Rumelhart y McClelland (1986).
- Siguiendo dichos principios, la i -ésima neurona artificial estándar consiste en:
 1. Un **conjunto de entradas** x_j y unos pesos sinápticos (fuerza de una conexión entre dos neuronas) $w_{i,j}$, con $j = 1, \dots, n$
 2. Una **regla de propagación** h_i definida a partir del conjunto de entradas y los pesos sinápticos. Es decir:

$$h_i(x_1, \dots, x_n, w_{i,1}, \dots, w_{i,n})$$

- La regla de propagación más comúnmente utilizada consiste en combinar linealmente las entradas y los pesos sinápticos, obteniéndose:

$$h_i(x_1, \dots, x_n, w_{i,1}, \dots, w_{i,n}) = \sum_{j=1}^n w_{i,j} x_j$$

- Suele ser habitual añadir al conjunto de pesos de la neurona un parámetro adicional θ_i , que se denomina umbral (bias), el cual se acostumbra a restar al potencial pos-sináptico. Es decir:






$$h_i(x_1, \dots, x_n, w_{i,1}, \dots, w_{i,n}) = \sum_{j=1}^n w_{i,j} x_j - \theta_i$$





3. Modelo Estándar de una Red Neuronal Artificial

3. Una **función de activación**, la cual representa simultáneamente la salida de la neurona y su estado de activación. Si denotamos por y_i dicha función de activación, se tiene:

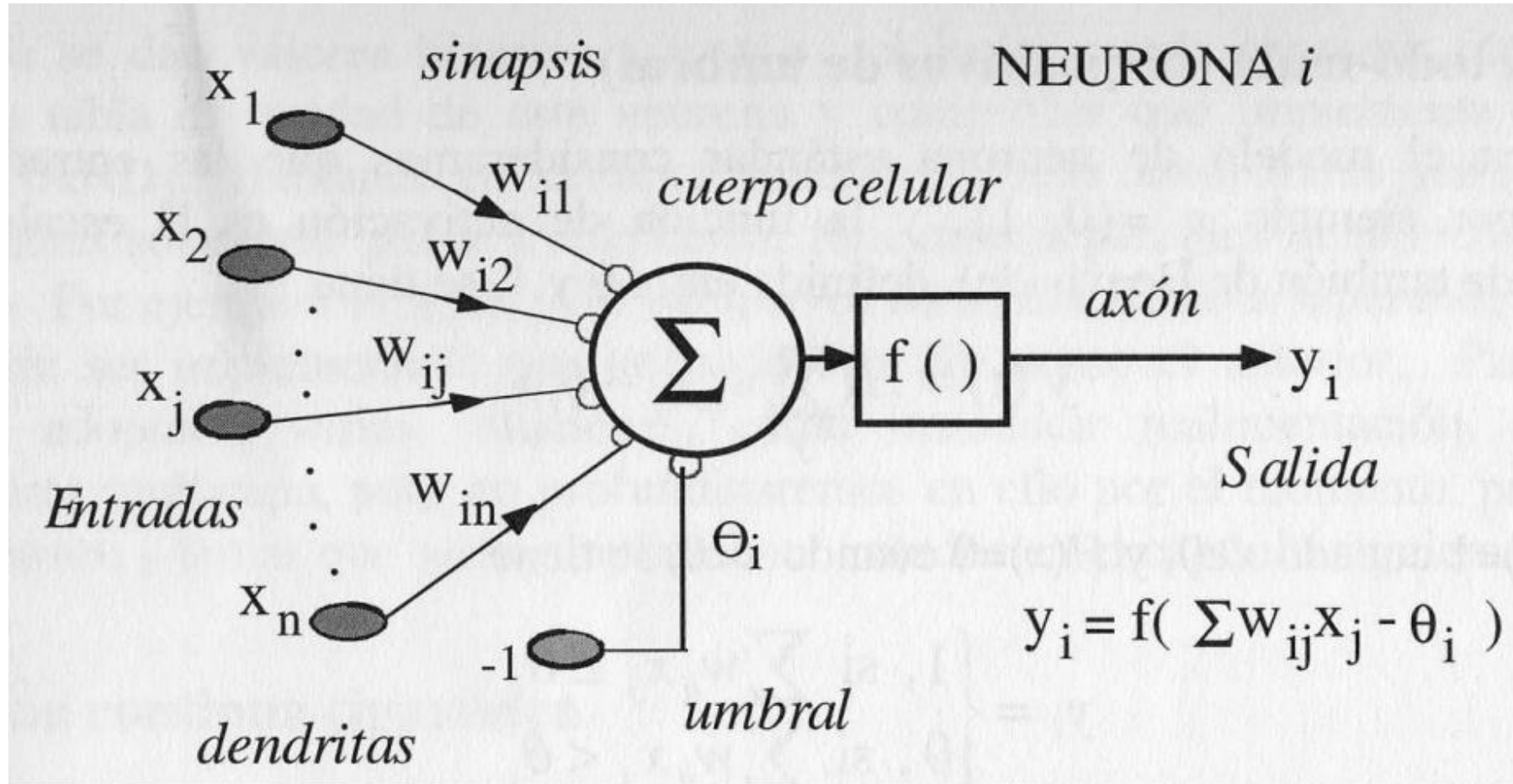
$$y_i = f(h_i) = f\left(\sum_{j=1}^n w_{i,j} x_j - \theta_i\right)$$

Funciones de activación de neurona artificial

Nombre	Relación Entrada /Salida	Icono
Limitador Fuerte	$a = 0 \quad n < 0$ $a = 1 \quad n \geq 0$	
Limitador Fuerte Simétrico	$a = -1 \quad n < 0$ $a = +1 \quad n \geq 0$	
Lineal Positiva	$a = 0 \quad n < 0$ $a = n \quad 0 \leq n$	
Lineal	$a = n$	
Lineal Saturado	$a = 0 \quad n < 0$ $a = n \quad 0 \leq n < 1$ $a = 1 \quad n > 1$	

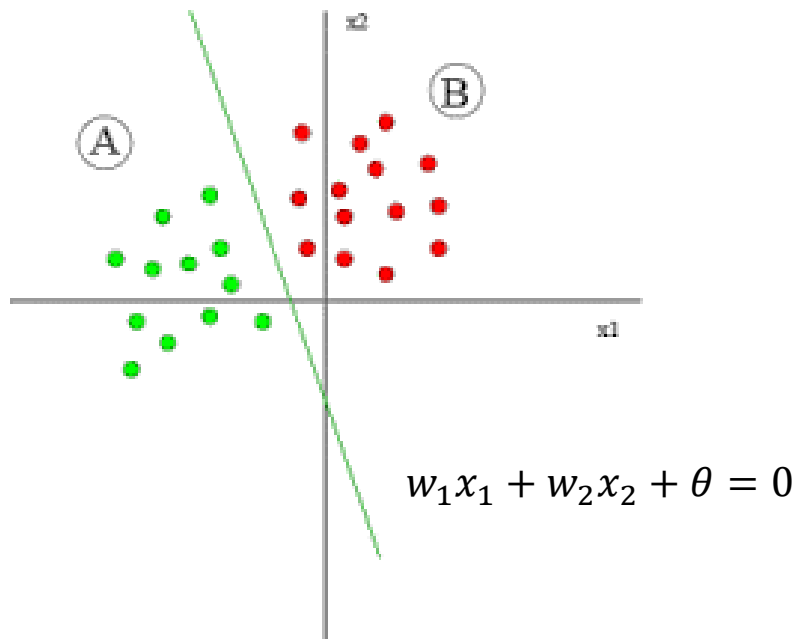
Nombre	Relación Entrada /Salida	Icono
Lineal Saturado Simétrico	$a = -1 \quad n < -1$ $a = n \quad -1 \leq n \leq 1$ $a = +1 \quad n > 1$	
Sigmoidal Logarítmico	$a = \frac{1}{1 + e^{-n}}$	
Tangente Sigmoidal Hiperbólica	$a = \frac{e^n - e^{-n}}{e^n + e^{-n}}$	
Competitiva	$a = 1$ Neurona con n max $a = 0$ el resto de neuronas	

3. Modelo Estándar de una Red Neuronal Artificial



4. El Perceptrón Simple

- ❖ Fue introducido por Frank Rosenblatt en 1962.
- ❖ Se concibió como un sistema capaz de realizar tareas de clasificación de forma automática.
- ❖ A partir de un número de elementos etiquetados, el sistema determina la ecuación del hiperplano discriminante.

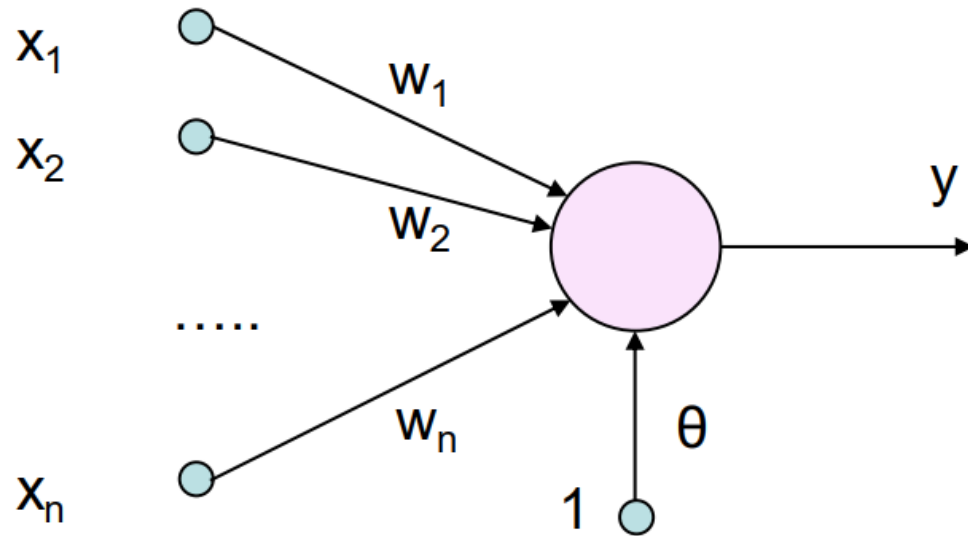


Frank Rosenblatt

4. El Perceptrón Simple

ARQUITECTURA DEL PERCEPTRON SIMPLE

❖ Es un modelo unidireccional compuesto por dos capas de neuronas, una de **entrada** y otra de **salida**.



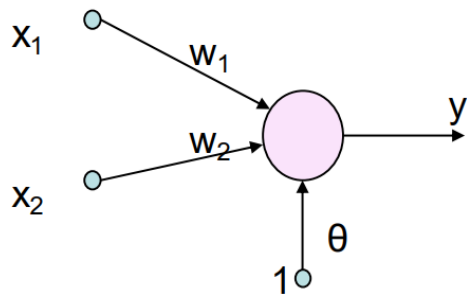
$$y = \begin{cases} +1, & \text{si } w_1x_1 + \dots + w_nx_n + \theta > 0 \\ -1, & \text{si } w_1x_1 + \dots + w_nx_n + \theta \leq 0 \end{cases}$$

4. El Perceptrón Simple

ARQUITECTURA DEL PERCEPTRON SIMPLE

- ❖ El perceptrón equivale a un hiperplano de dimensión $n - 1$ capaz de **separar las clases (es dicotómica)**.
 - Si la salida del perceptrón es +1, la entrada pertenecerá a una clase (estará situada a un lado del hiperplano)
 - Si la salida es -1, la entrada pertenecerá a la clase contraria (estará situada al otro lado del hiperplano)
- ❖ La ecuación del hiperplano es: $w_1x_1 + w_2x_2 + \dots + w_nx_n + \theta = 0$

Ejemplo: Perceptrón Simple 2 dimensiones

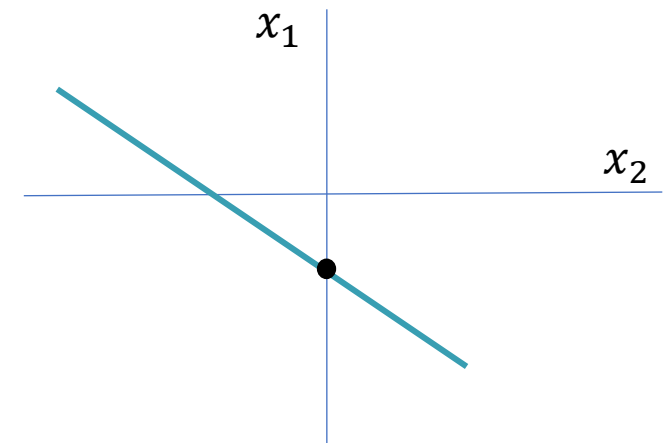


$$y = \begin{cases} +1, & \text{si } w_1x_1 + w_2x_2 + \theta > 0 \\ -1, & \text{si } w_1x_1 + w_2x_2 + \theta \leq 0 \end{cases}$$

La ecuación del hiperplano es:

$$w_1x_1 + w_2x_2 + \theta = 0$$

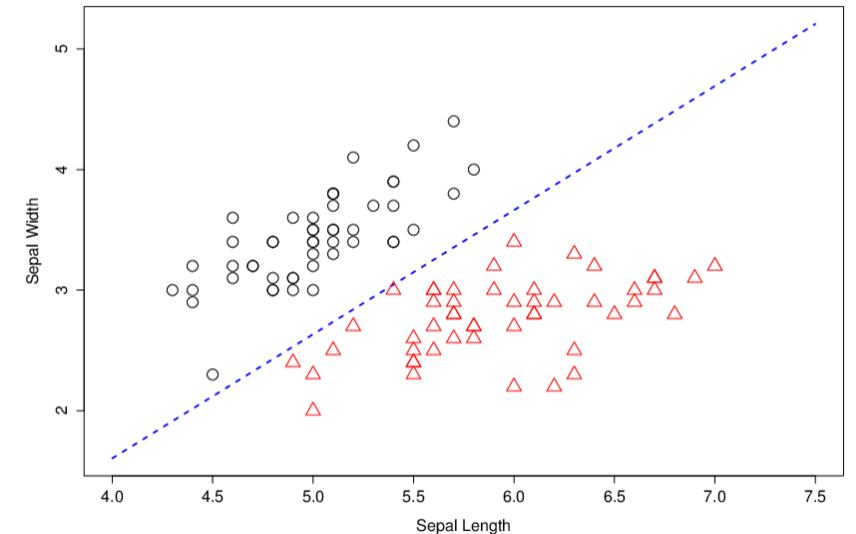
$$x_2 = -\frac{w_1}{w_2}x_1 - \frac{\theta}{w_2}$$



4. El Perceptrón Simple

APRENDIZAJE DE UN PERCEPTRON

- ❖ Se dispone de un conjunto de observaciones (patrones, ejemplos, **datos**) de los que se sabe su categoría o clase (etiqueta).
- ❖ Los ejemplos o datos son puntos en un espacio multidimensional:
 $\mathcal{R}^n: (x_1, \dots, x_n)$
- ❖ Hay que determinar la ecuación del hiperplano que deja a un lado los ejemplos de una clase y al otro lado los de la otra clase.
- ❖ La ecuación del hiperplano se deduce a partir de los ejemplos (características o **datos**).
- ❖ El aprendizaje es un proceso iterativo supervisado.



4. El Perceptrón Simple

APRENDIZAJE

❖ Dado:

- Conjunto de patrones (características)
- Vector de entrada: (x_1, \dots, x_n)
- Salida: $d(x)$

$$d(x) = +1 \quad \text{si } x \in A$$

$$d(x) = -1 \quad \text{si } x \in B$$

❖ Hiperplano discriminante:

$(w_1, \dots, w_n, \theta)$ tales que

$$w_1 x_1 + w_2 x_2 + \dots + w_n x_n + \theta = 0$$

Separa las clases A y B .

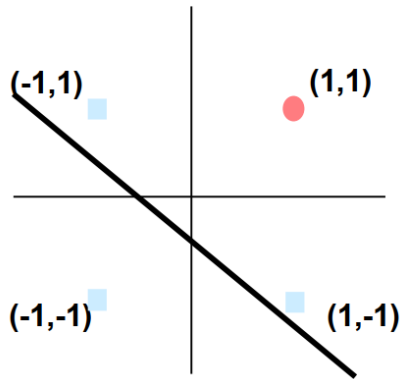
Nota: la función discriminante minimiza la probabilidad de equivocarse al clasificar los individuos en cada grupo.

4. El Perceptrón Simple

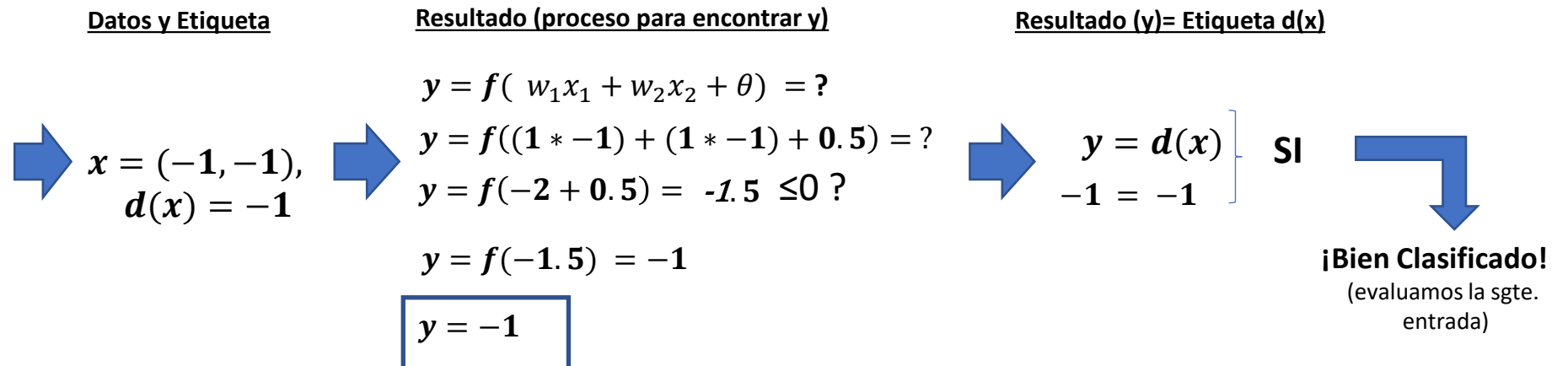
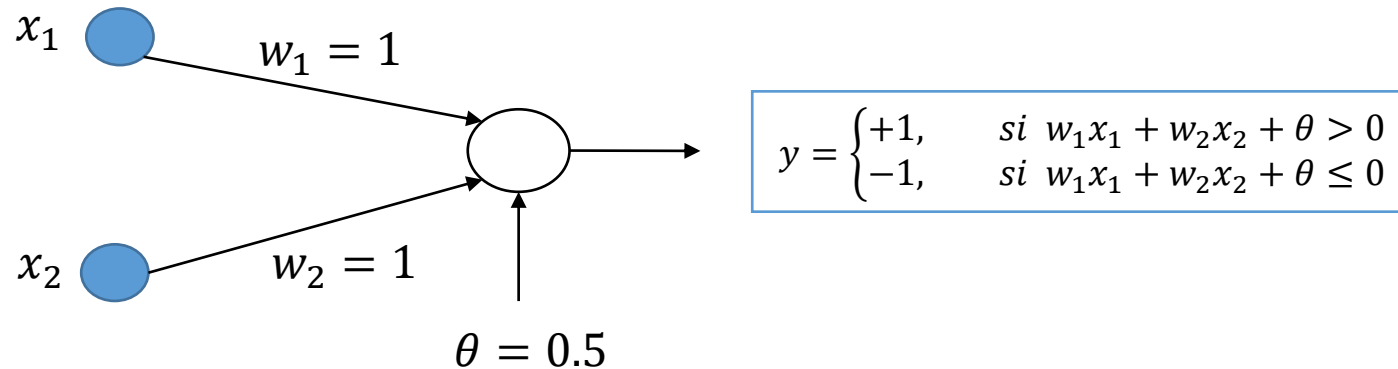
EJEMPLO: Función lógica AND

Datos:

		$d(x)$
x_1	x_2	AND
-1	-1	-1
1	-1	-1
-1	1	-1
1	1	1



Paso #1: Asignamos valores aleatorios para pesos de las entradas y umbral

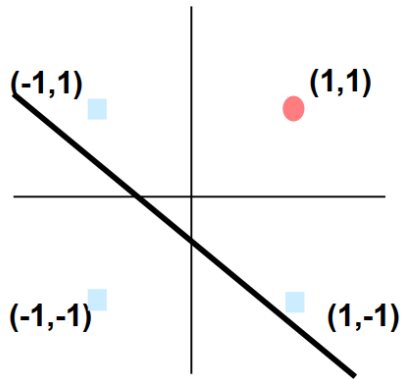


4. El Perceptrón Simple

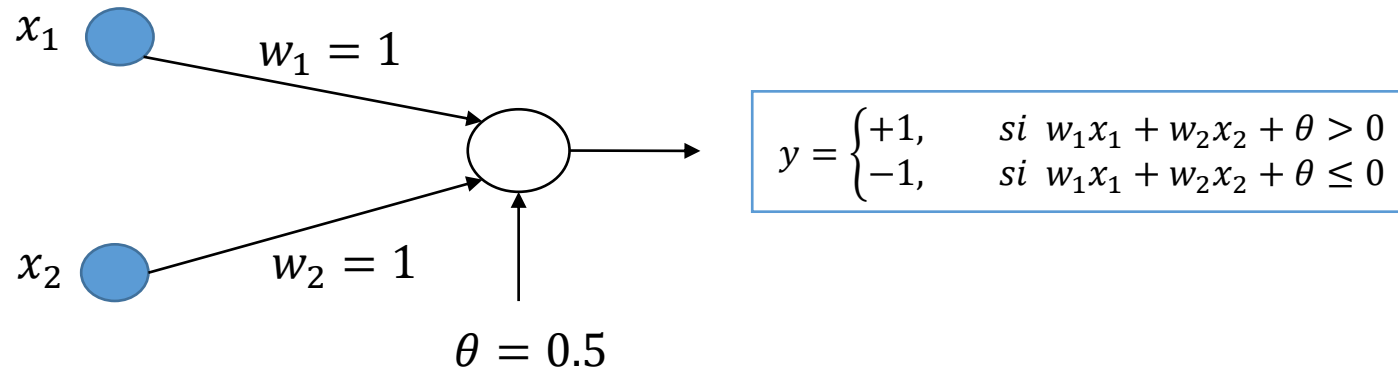
EJEMPLO: Función lógica AND

Datos:

		$d(x)$
x_1	x_2	AND
-1	-1	-1
1	-1	-1
-1	1	-1
1	1	1



Paso #2: Evaluamos la siguiente entrada.



Datos y Etiqueta

$x = (1, -1),$
 $d(x) = -1$

Resultado (proceso para encontrar y)

$$y = f(w_1x_1 + w_2x_2 + \theta) = ?$$

$$y = f((1 * 1) + (1 * -1) + 0.5) = ?$$
$$y = f(0.5) = 0.5 > 0 ?$$

$$y = f(0.5) = 1$$

$$y = 1$$

Resultado (y)= Etiqueta d(x)

$$y = d(x)$$
$$1 = -1$$

NO

¡Mal Clasificado!

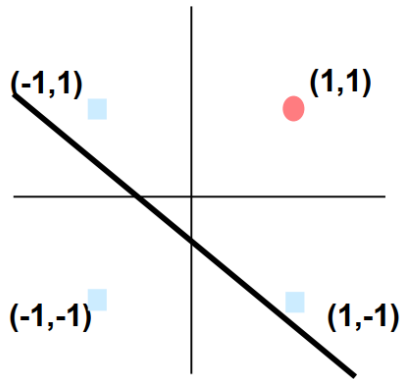
(modificamos pesos y umbral)

4. El Perceptrón Simple

EJEMPLO: Función lógica AND

Datos:

		$d(x)$
x_1	x_2	AND
-1	-1	-1
1	-1	-1
-1	1	-1
1	1	1



Paso #2.1: Modificamos pesos y umbral hasta encontrar una buena clasificación (para la misma entrada).

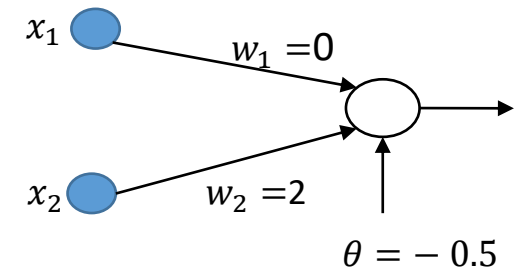
Se conoce: $w_1 = 1, w_2 = 1, \theta = 0.5$

Calculamos nuevos pesos y umbral

$$w_1 = w_1 + d(x) * x_1 = 1 - 1 = 0$$

$$w_2 = w_2 + d(x) * x_2 = 1 + 1 = 2$$

$$\theta = \theta + d(x) = 0.5 - 1 = -0.5$$



$$y = \begin{cases} +1, & \text{si } w_1x_1 + w_2x_2 + \theta > 0 \\ -1, & \text{si } w_1x_1 + w_2x_2 + \theta \leq 0 \end{cases}$$

Datos y Etiqueta

$x = (1, -1),$
 $d(x) = -1$

Resultado

$$y = f(w_1x_1 + w_2x_2 + \theta) = ?$$

$$y = f((0 * 1) + (2 * -1) - 0.5) = ?$$
$$y = f(-2.5) = -2.5 \leq 0 ?$$

$$y = f(-2.5) = -1$$

$$y = -1$$

Resultado = Etiqueta

$$y = d(x)$$
$$-1 = -1$$

SI



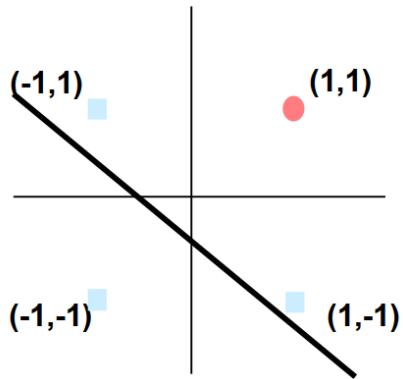
¡Bien Clasificado!
(evaluamos la sgte. entrada)

4. El Perceptrón Simple

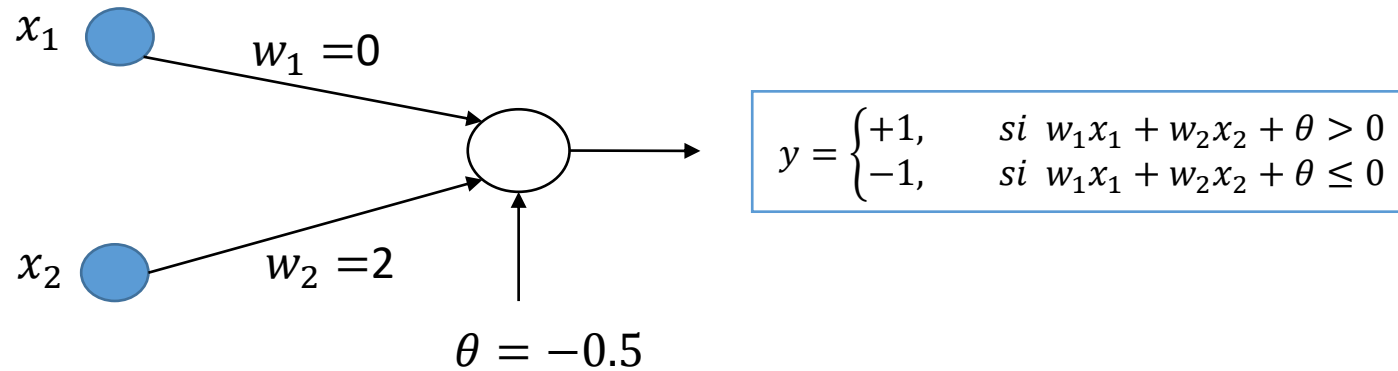
EJEMPLO: Función lógica AND

Datos:

		$d(x)$
x_1	x_2	AND
-1	-1	-1
1	-1	-1
-1	1	-1
1	1	1



Paso #3: Evaluamos la siguiente entrada.



Datos y Etiqueta

$x = (-1, 1),$
 $d(x) = -1$

Resultado (proceso para encontrar y)

$$y = f(w_1x_1 + w_2x_2 + \theta) = ?$$

$$y = f((0 * -1) + (2 * 1) - 0.5) = ?$$

$$y = f(1.5) = 1.5 > 0 ?$$

$$y = f(1.5) = 1$$

$$y = 1$$

Resultado (y)= Etiqueta d(x)

$$y = d(x)$$
$$1 = -1$$

NO



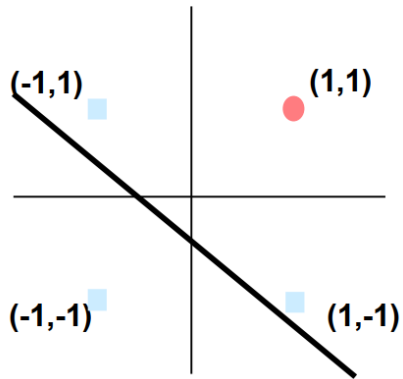
¡Mal Clasificado!
(modificamos pesos y umbral)

4. El Perceptrón Simple

EJEMPLO: Función lógica AND

Datos:

		$d(x)$
x_1	x_2	AND
-1	-1	-1
1	-1	-1
-1	1	-1
1	1	1



Paso #3.1: Modificamos pesos y umbral hasta encontrar una buena clasificación (para la misma entrada).

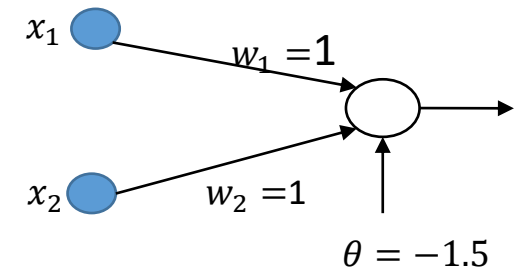
Se conoce: $w_1 = 0, w_2 = 2, \theta = -0.5$

Calculamos nuevos pesos y umbral

$$w_1 = w_1 + d(x) * x_1 = 0 + (-1 * -1) = 1$$

$$w_2 = w_2 + d(x) * x_2 = 2 + (-1 * 1) = 1$$

$$\theta = \theta + d(x) = -0.5 - 1 = -1.5$$



$$y = \begin{cases} +1, & \text{si } w_1x_1 + w_2x_2 + \theta > 0 \\ -1, & \text{si } w_1x_1 + w_2x_2 + \theta \leq 0 \end{cases}$$

Datos y Etiqueta

$$x = (-1, 1), \\ d(x) = -1$$

Resultado

$$y = f(w_1x_1 + w_2x_2 + \theta) = ?$$

$$y = f((1 * -1) + (1 * 1) - 1.5) = ? \\ y = f(-1.5) = -1.5 \leq 0 ?$$

$$y = f(-1.5) = -1$$

$$y = -1$$

Resultado = Etiqueta

$$y = d(x) \\ -1 = -1$$

SI



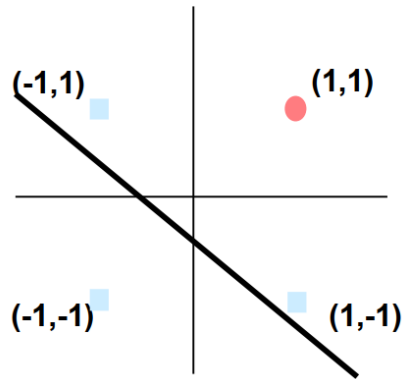
¡Bien Clasificado!
(evaluamos la sgte. entrada)

4. El Perceptrón Simple

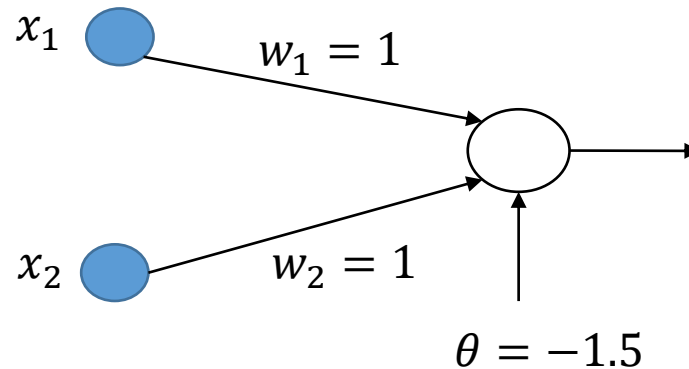
EJEMPLO: Función lógica AND

Datos:

		$d(x)$
x_1	x_2	AND
-1	-1	-1
1	-1	-1
-1	1	-1
1	1	1



Paso #4: Evaluamos la siguiente entrada.



$$y = \begin{cases} +1, & \text{si } w_1x_1 + w_2x_2 + \theta > 0 \\ -1, & \text{si } w_1x_1 + w_2x_2 + \theta \leq 0 \end{cases}$$

Datos y Etiqueta

➡ $x = (1, 1),$
 $d(x) = 1$

Resultado (proceso para encontrar y)

$$\begin{aligned} y &= f(w_1x_1 + w_2x_2 + \theta) = ? \\ y &= f((1 * 1) + (1 * 1) - 1.5) = ? \\ y &= f(0.5) = 0.5 > 0 ? \\ y &= f(0.5) = 1 \\ \boxed{y = 1} \end{aligned}$$

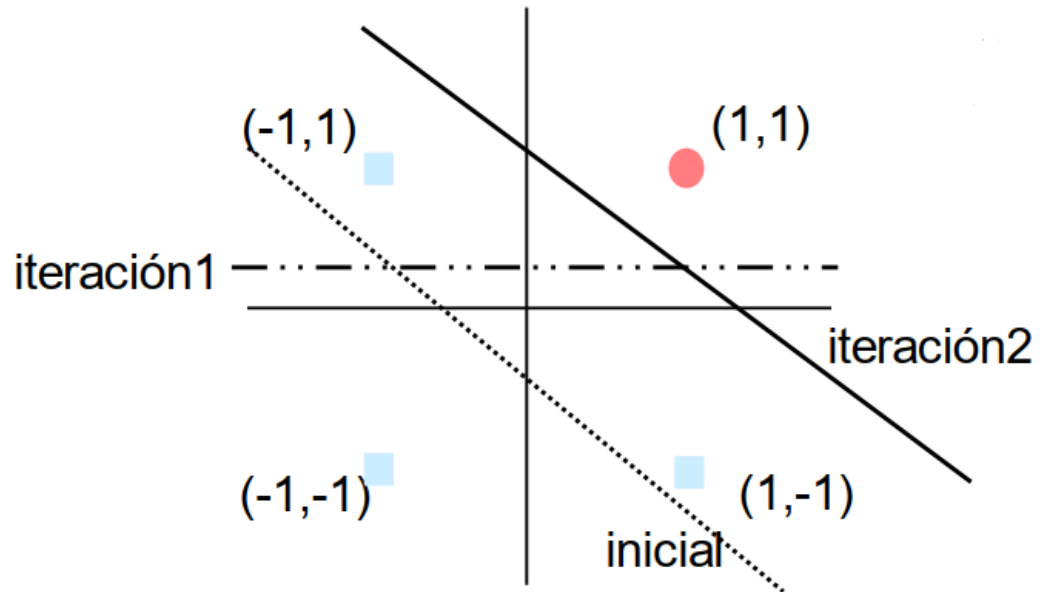
Resultado (y)= Etiqueta d(x)

➡ $y = d(x)$
 $1 = 1$ } SI
➡ ¡Bien Clasificado!

Un hiperplano solución es: $x_1 + x_2 - 1.5 = 0$

4. El Perceptrón Simple

EJEMPLO: Función lógica AND



El hiperplano se mueve de una iteración a otra para clasificar correctamente los patrones (características).

4. El Perceptrón Simple

SEUDOCODIGO

1. Comenzar con valores aleatorios para pesos y umbral
2. Modificación de los pesos y umbral hasta encontrar el hiperplano discriminante
 - a. Seleccionar un ejemplo x del conjunto de entrenamiento
 - b. Se calcula la salida de la red: $y = f(w_1x_1 + w_2x_2, \dots, w_nx_n + \theta)$
 - c. Si $y \neq d(x)$ (clasificación incorrecta) se modifican los pesos y el umbral:
$$w_i(t + 1) = w_i(t) + d(x) * x_i$$
$$\theta(t + 1) = \theta(t) + d(x)$$
 - d. Repetir desde el paso 2.a hasta completar el conjunto de patrones de entrenamiento o hasta alcanzar el criterio de parada.

5. Redes ADALINE

ADALINE (ADaptive Linear NEuron)

fue desarrollado en 1960 por Widrow y Hoff.

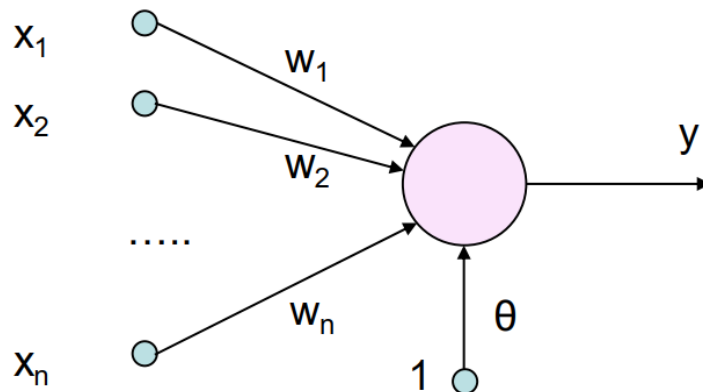


Dr. Bernard Widrow



PhD. Marcian Hoff

- Las entradas pueden ser continuas y se utiliza una neurona similar a la del Perceptrón Simple, igualmente, resulta un caso de respuesta lineal.



$$y = w_1 x_1 + \cdots + w_n x_n + \theta$$

5. Redes ADALINE

APRENDIZAJE: DIFERENCIAS ENTRE PERCEPTRON Y REDES ADALINE

PERCEPTRON

- Utiliza la salida de la función umbral (binaria) para el aprendizaje.
- Sólo se tiene en cuenta si se ha equivocado o no (no cuanto se equivocó).
- Valores de respuesta no continuos.

ADALINE

- Utiliza directamente la salida de la red (real) teniendo en cuenta **cuánto se ha equivocado**.
- Considera el error entre la salida lograda y versus la salida deseada d (etiqueta):
$$|d^p - y^p|$$
- Esta regla se conoce como **REGLA DELTA**. La constante α se denomina **TASA DE APRENDIZAJE**.
$$\Delta w_i = \alpha |d^p - y^p| x_i$$
- Valores de respuesta continuos.
- Se busca minimizar la desviación de la red para todos los patrones (características) de entrada, eligiendo una medida del error global.
- Normalmente se utiliza el error cuadrático medio:

$$E = \frac{1}{2} \sum_{p=1}^m (d^p - y^p)^2$$

5. Redes ADALINE

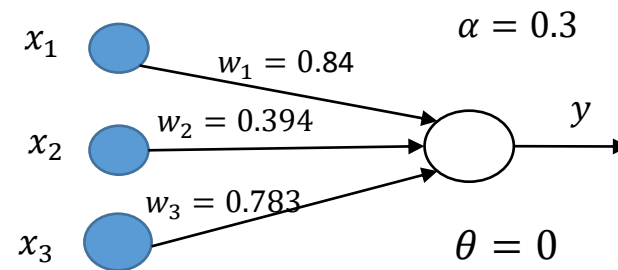
EJEMPLO: Decodificador binario a decimal

Datos



x_1	x_2	x_3	$d(x)$
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

Paso #1: Asignamos valores aleatorios para pesos de las entradas y umbral



Paso #2: Seleccionar un ejemplo x del conjunto de entrenamiento: $x_1 = 0, x_2 = 0, x_3 = 1$

Paso #3: Calcular la salida de la red: $y = f(w_1x_1 + \dots + w_nx_n + \theta)$ y obtener la diferencia.

$$y = 0.84 * 0 + 0.394 * 0 + 0.783 * 1 = 0.783$$

$$E = |d^p - y^p| = |1 - 0.783| = 0.217$$

Paso #4: Para todos los pesos y para el umbral, calcular:

$$w_1 = w_1 + \alpha * E * x_1 = 0.84 + 0.3 * 0.217 * 0 = 0.840$$

$$w_2 = w_2 + \alpha * E * x_2 = 0.394 + 0.3 * 0.217 * 0 = 0.394$$

$$w_3 = w_3 + \alpha * E * x_3 = 0.783 + 0.3 * 0.217 * 1 = 0.8481$$

5. Redes ADALINE

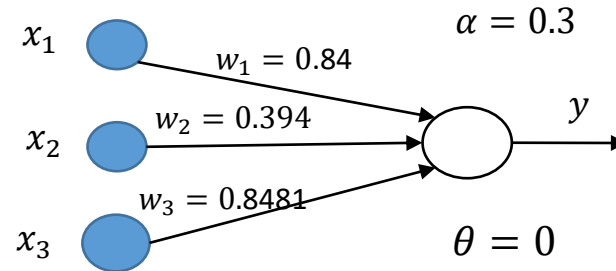
EJEMPLO: Decodificador binario a decimal

Datos



x_1	x_2	x_3	$d(x)$
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

Paso #5: Modificar los pesos y el umbral y repetimos desde el **paso #2** hasta cumplir el criterio de parada.



Paso #2: Seleccionar un ejemplo x del conjunto de entrenamiento: $x_1 = 0, x_2 = 1, x_3 = 0$

Paso #3: Calcular la salida de la red: $y = f(w_1x_1 + \dots + w_nx_n + \theta)$ y obtener la diferencia.

$$y = 0.84 * 0 + 0.394 * 1 + 0.8481 * 0 = 0.394$$

$$E = |d^p - y^p| = |2 - 0.394| = 1.606$$

Paso #4: Para todos los pesos y para el umbral, calcular:

$$w_1 = w_1 + \alpha * E * x_1 = 0.84 + 0.3 * 1.606 * 0 = 0.840$$

$$w_2 = w_2 + \alpha * E * x_2 = 0.394 + 0.3 * 1.606 * 1 = 0.876$$

$$w_3 = w_3 + \alpha * E * x_3 = 0.8481 + 0.3 * 1.606 * 0 = 0.8481$$

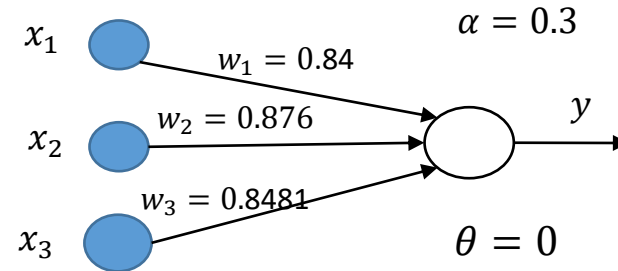
5. Redes ADALINE

EJEMPLO: Decodificador binario a decimal

Datos

x_1	x_2	x_3	$d(x)$
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

Paso #5: Modificar los pesos y el umbral y repetimos desde el **paso #2** hasta cumplir el criterio de parada.



Paso #2: Seleccionar un ejemplo x del conjunto de entrenamiento: $x_1 = 0, x_2 = 1, x_3 = 1$

Paso #3: Calcular la salida de la red: $y = f(w_1x_1 + \dots + w_nx_n + \theta)$ y obtener la diferencia.

$$y = 0.84 * 0 + 0.876 * 1 + 0.8481 * 1 = 1.7241$$

$$E = |d^p - y^p| = |3 - 1.7241| = 1.2759$$

Paso #4: Para todos los pesos y para el umbral, calcular:

$$w_1 = w_1 + \alpha * E * x_1 = 0.84 + 0.3 * 1.2759 * 0 = 0.840$$

$$w_2 = w_2 + \alpha * E * x_2 = 0.876 + 0.3 * 1.2759 * 1 = 1.259$$

$$w_3 = w_3 + \alpha * E * x_3 = 0.8481 + 0.3 * 1.2759 * 1 = 1.231$$

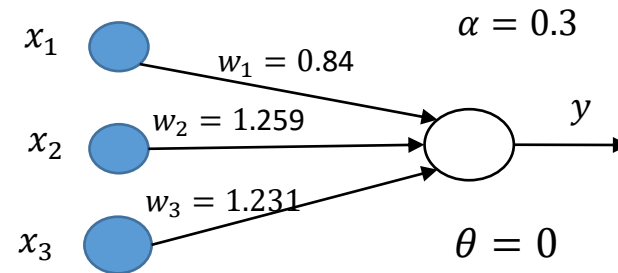
5. Redes ADALINE

EJEMPLO: Decodificador binario a decimal

Datos

x_1	x_2	x_3	$d(x)$
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

Paso #5: Modificar los pesos y el umbral y repetimos desde el **paso #2** hasta cumplir el criterio de parada.



Paso #2: Seleccionar un ejemplo x del conjunto de entrenamiento: $x_1 = 1, x_2 = 0, x_3 = 0$

Paso #3: Calcular la salida de la red: $y = f(w_1x_1 + \dots + w_nx_n + \theta)$ y obtener la diferencia.

$$y = 0.84 * 1 + 0.876 * 0 + 0.8481 * 0 = 0.84$$

$$E = |d^p - y^p| = |4 - 0.84| = 3.16$$

Paso #4: Para todos los pesos y para el umbral, calcular:

$$w_1 = w_1 + \alpha * E * x_1 = 0.84 + 0.3 * 3.16 * 1 = 1.788$$

$$w_2 = w_2 + \alpha * E * x_2 = 1.259 + 0.3 * 3.16 * 0 = 1.259$$

$$w_3 = w_3 + \alpha * E * x_3 = 1.231 + 0.3 * 3.16 * 0 = 1.231$$

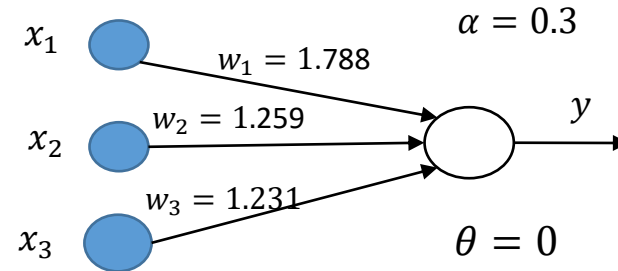
5. Redes ADALINE

EJEMPLO: Decodificador binario a decimal

Datos

x_1	x_2	x_3	$d(x)$
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

Paso #5: Modificar los pesos y el umbral y repetimos desde el **paso #2** hasta cumplir el criterio de parada.



Paso #2: Seleccionar un ejemplo x del conjunto de entrenamiento: $x_1 = 1, x_2 = 0, x_3 = 1$

Paso #3: Calcular la salida de la red: $y = f(w_1x_1 + \dots + w_nx_n + \theta)$ y obtener la diferencia.

$$y = 1.788 * 1 + 1.259 * 0 + 1.231 * 1 = 3.019$$

$$E = |d^p - y^p| = |5 - 3.019| = 1.981$$

Paso #4: Para todos los pesos y para el umbral, calcular:

$$w_1 = w_1 + \alpha * E * x_1 = 1.788 + 0.3 * 1.981 * 1 = 2.382$$

$$w_2 = w_2 + \alpha * E * x_2 = 1.259 + 0.3 * 1.981 * 0 = 1.259$$

$$w_3 = w_3 + \alpha * E * x_3 = 1.231 + 0.3 * 1.981 * 1 = 1.825$$

5. Redes ADALINE

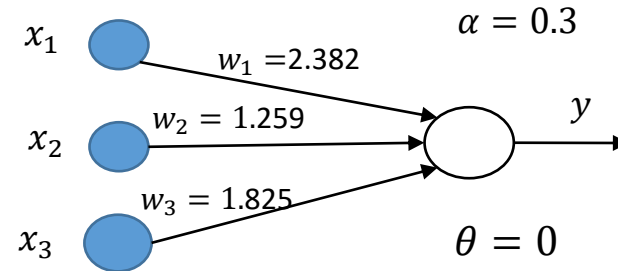
EJEMPLO: Decodificador binario a decimal

Datos

x_1	x_2	x_3	$d(x)$
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7



Paso #5: Modificar los pesos y el umbral y repetimos desde el **paso #2** hasta cumplir el criterio de parada.



Paso #2: Seleccionar un ejemplo x del conjunto de entrenamiento: $x_1 = 1, x_2 = 1, x_3 = 0$

Paso #3: Calcular la salida de la red: $y = f(w_1x_1 + \dots + w_nx_n + \theta)$ y obtener la diferencia.

$$y = 2.382 * 1 + 1.259 * 1 + 1.825 * 0 = 3.641$$

$$E = |d^p - y^p| = |6 - 3.641| = 2.359$$

Paso #4: Para todos los pesos y para el umbral, calcular:

$$w_1 = w_1 + \alpha * E * x_1 = 2.382 + 0.3 * 2.359 * 1 = 3.09$$

$$w_2 = w_2 + \alpha * E * x_2 = 1.259 + 0.3 * 2.359 * 1 = 1.967$$

$$w_3 = w_3 + \alpha * E * x_3 = 1.825 + 0.3 * 2.359 * 0 = 1.825$$

5. Redes ADALINE

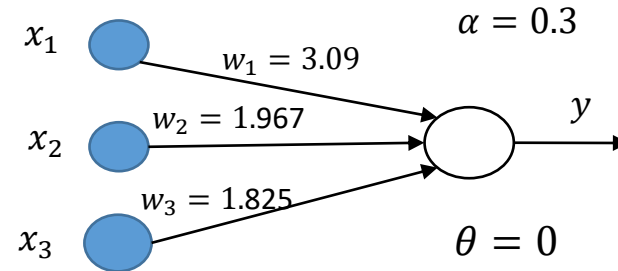
EJEMPLO: Decodificador binario a decimal

Datos

x_1	x_2	x_3	$d(x)$
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7



Paso #5: Modificar los pesos y el umbral y repetimos desde el **paso #2** hasta cumplir el criterio de parada.



Paso #2: Seleccionar un ejemplo x del conjunto de entrenamiento: $x_1 = 1, x_2 = 1, x_3 = 1$

Paso #3: Calcular la salida de la red: $y = f(w_1x_1 + \dots + w_nx_n + \theta)$ y obtener la diferencia.

$$y = 3.09 * 1 + 1.967 * 1 + 1.825 * 1 = 6.882$$

$$E = |d^p - y^p| = |7 - 6.882| = 0.118$$

Paso #4: Para todos los pesos y para el umbral, calcular:

Resultado después de la
primera iteración del entrenamiento

$$w_1 = w_1 + \alpha * E * x_1 = 3.09 + 0.3 * 0.118 * 1 = 3.125$$

$$w_2 = w_2 + \alpha * E * x_2 = 1.967 + 0.3 * 0.118 * 1 = 2.002$$

$$w_3 = w_3 + \alpha * E * x_3 = 1.825 + 0.3 * 0.118 * 1 = 1.860$$

5. Redes ADALINE

EJEMPLO: Decodificador binario a decimal

Visualización de los pesos según iteraciones

Iteración	w1	w2	w3
1	3.13	2.00	1.86
2	3.61	1.98	1.42
3	3.82	1.98	1.2
4	3.92	1.98	1.1
5	3.96	1.99	1.02
6	3.99	2.00	1.01
7	4.00	2.00	1.00
8	4.00	2.00	1.00
9	4.00	2.00	1.00
10	4.00	2.00	1.00

- La tasa de aprendizaje α también puede ser adaptativa.
- Por ejemplo al inicio, el valor puede ser alto, para dar “grandes pasos” de corrección del error y para salir de mínimos locales.
- Sin embargo al final del entrenamiento debe disminuir para hacer correcciones finas.

5. Redes ADALINE

APRENDIZAJE: Seudocódigo

1. Inicializar los pesos y umbral de forma aleatoria.
2. Seleccionar un ejemplo x del conjunto de entrenamiento.
3. Calcular la salida de la red: $y = f(w_1x_1 + \dots + w_nx_n + \theta)$ y obtener la diferencia.

4. Para todos los pesos y para el umbral, calcular:

$$\Delta w_i = \alpha |d^p - y^p| x_i \quad \Delta \theta_i = \alpha |d^p - y^p|$$

5. Modificar los pesos y el umbral del siguiente modo:

$$w_i(t + 1) = w_i(t) + \Delta w_i$$

$$\theta(t + 1) = \theta(t) + \Delta \theta_i$$

6. Repetir para todos los patrones de entrenamiento hasta cumplir el criterio de parada.

6. Ventajas / Limitaciones

VENTAJAS

- El uso del Perceptrón o de las redes ADALINE permite aproximar de manera fácil, cualquier tipo de función o sistemas, sólo conociendo un conjunto de ejemplos (características o entradas).
- El uso del Perceptrón o de las redes ADALINE permite que cualquier sistema (caja negra), se puede representar por una red.

LIMITACIONES

- Estas técnicas poseen grandes limitaciones.
- Sólo pueden resolver sistemas donde los ejemplos o características (entradas) son linealmente separables (por ejemplo no resolverá el XOR Exclusivo para el cual no existe un hiperplano).

x_1	x_2	$d(x)$
-1	-1	1
-1	1	-1
1	-1	-1
1	1	1



Solución: Combinar varios Perceptrones

PREGUNTAS

Dudas y opiniones