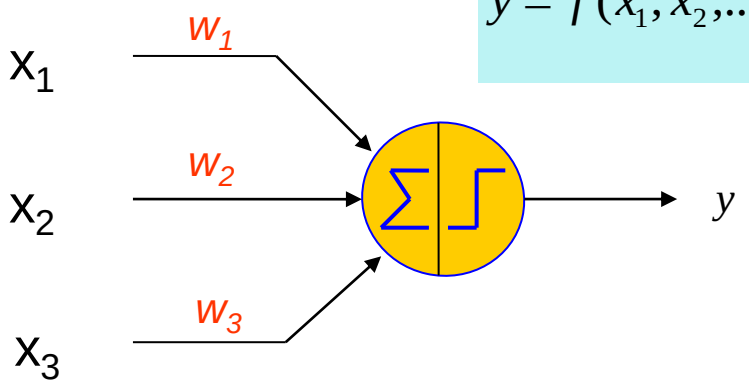
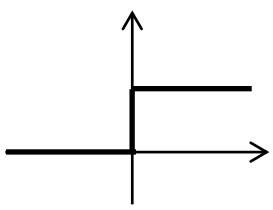
El Perceptrón

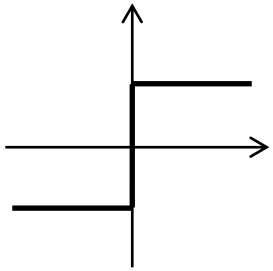
1958 - **El psicólogo Frank Ronsenblant desarrolló un modelo simple de neurona basado en el modelo de McCulloch y Pitts que utilizaba una regla de aprendizaje basada en la corrección del error: Perceptrón**

$$y = f(x_1, x_2, ..., x_n) = \begin{cases} 1 & \text{si} \quad w_1 x_1 + w_2 x_2 + ... + w_n x_n \geq \theta \\ -1 & \text{si} \quad w_1 x_1 + w_2 x_2 + ... + w_n x_n < \theta \end{cases}$$

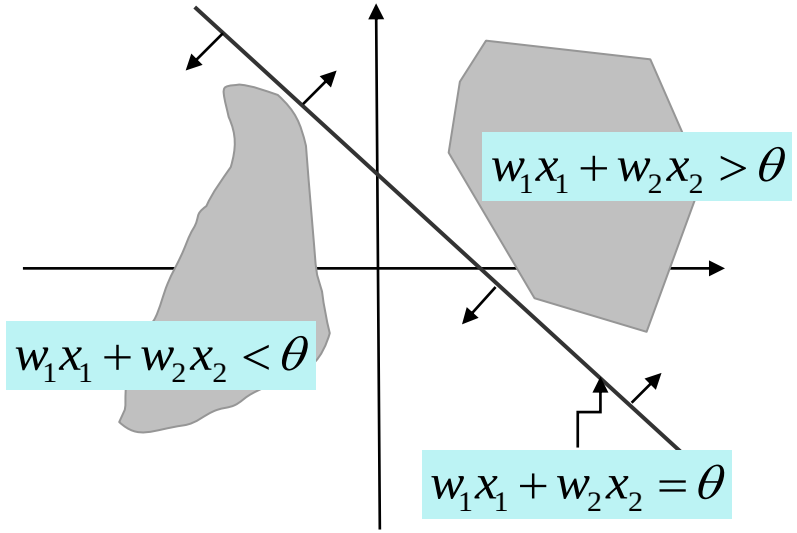




Función paso o *De Heaviside*



Función signo



¿Qué se pretende con el Perceptrón?

Se dispone de la siguiente **información:**

- Conjunto de patrones **{$\mathbf{x}^k$},** $k = 1,2,\ldots,p_1$ **,** de la clase C_1 ($z^k = 1$)
- Conjunto de patrones **{$\mathbf{x}^r$},** $k = p_1+1,\ldots,p$ **,** de la clase C_2 ($z^r = -1$)

Se pretende que el **perceptrón** asigne a cada entrada (patrón x^k) la salida deseada z^k siguiendo un proceso de corrección de error (**aprendizaje**) para determinar los **pesos sinápticos** apropiados

Regla de aprendizaje del Perceptrón:

$$\Delta w_j(k) = \eta(k)\left[z(k) - y(k)\right]x_j(k)$$

error

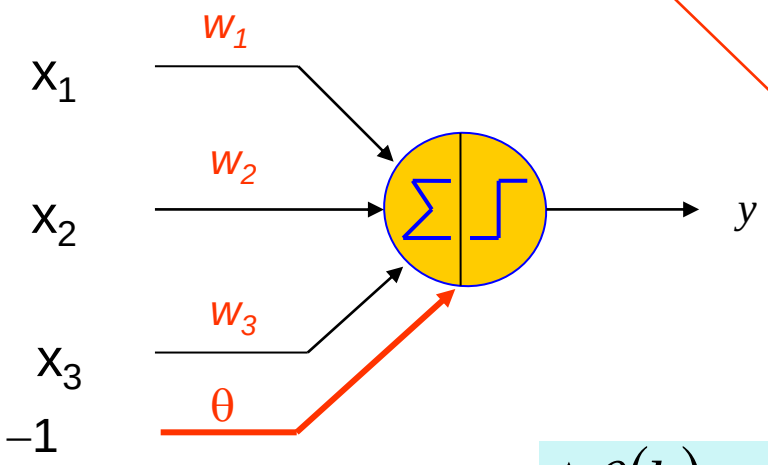
tasa de aprendizaje

$$w_j(k+1) = \begin{cases} w_j(k) + 2\eta x_j(k) & \text{si } y(k) = -1 \text{ y } z(k) = 1, \\ w_j(k) & \text{si } y(k) = z(k) \\ w_j(k) - 2\eta x_j(k) & \text{si } y(k) = 1 \text{ y } z(k) = -1 \end{cases}$$

¿Cómo se modifica el sesgo θ ?

$$w_1 x_1 + w_2 x_2 + \ldots + w_n x_n \geq \theta \quad \Leftrightarrow \quad w_1 x_1 + w_2 x_2 + \ldots + w_n x_n + w_{n+1} x_{n+1} \geq 0$$

w_{n+1} → θ ; x_{n+1} → −1



$$w_1 x_1 + w_2 x_2 + \ldots + w_n x_n + \theta(-1) \geq 0$$

$$\Delta\theta(k) = -\eta(k)\left[z(k) - y(k)\right]$$

Algoritmo del Perceptrón

Paso 0: Inicialización

Inicializar los pesos sinápticos con números aleatorios del intervalo [-1,1].

Ir al paso 1 con k=1

Paso 1: (k-ésima iteración)

Calcular

$$y(k) = \operatorname{sgn}\left(\sum_{j=1}^{n+1} w_j x_j(k) \right)$$

Paso 2: Corrección de los pesos sinápticos

Si $z(k) \neq y(k)$ modificar los pesos sinápticos según la expresión:

$$w_j(k+1) = w_j(k) + \eta\left[z_i(k) - y_i(k)\right]x_j(k), \quad j = 1,2,\ldots,n+1$$

Paso 3: Parada

Si no se han modificado los pesos en las últimas p iteraciones, es decir,

$$w_j(r) = w_j(k), \quad j = 1,2,\ldots,n+1, \; r = k+1,\ldots,k+p$$

parar. La red se ha estabilizado.

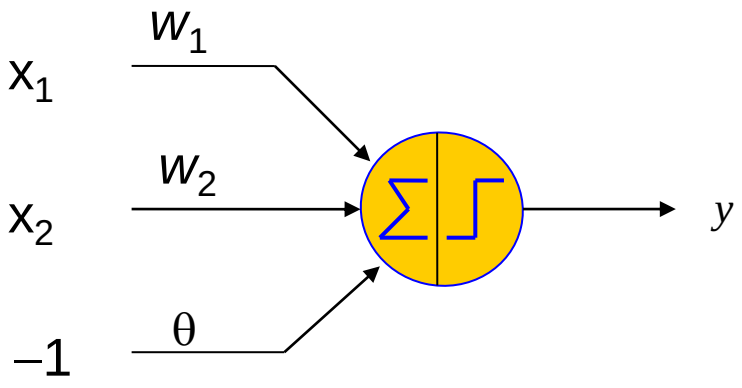
En otro caso, ir al **Paso 1** con k=k+1.

Ejemplo

☺ Diseña un perceptrón que implemente la función **lógica AND**

AND

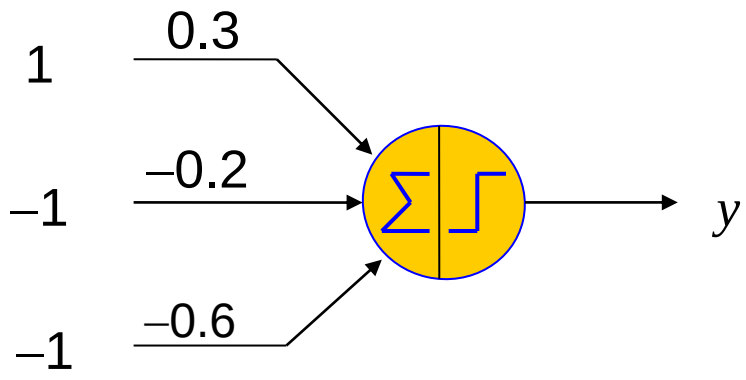
Entradas	Salidas
(1, 1)	1
(1, –1)	–1
(–1, 1)	–1
(–1,–1)	–1

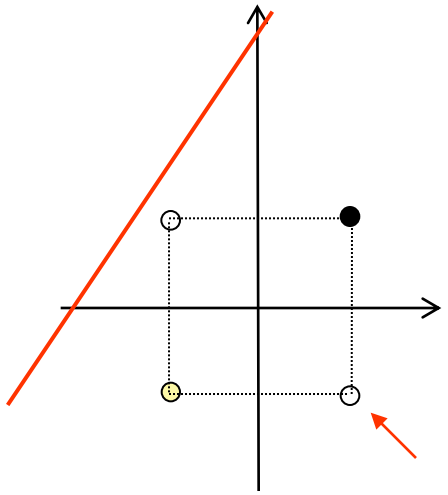


Paso 0: Inicialización aleatoria

$w_1 = 0.4, \quad w_2 = -0.2, \quad \theta = 0.6,$

☺ Diseña un perceptrón que implemente la función **lógica AND**





Paso 1:

Patrón de entrada (1,–1): $h = 0.3(1) - 0.2(-1) - 0.6(-1) = 1.1$

$y = 1$

Paso 2: Corrección de los pesos sinápticos

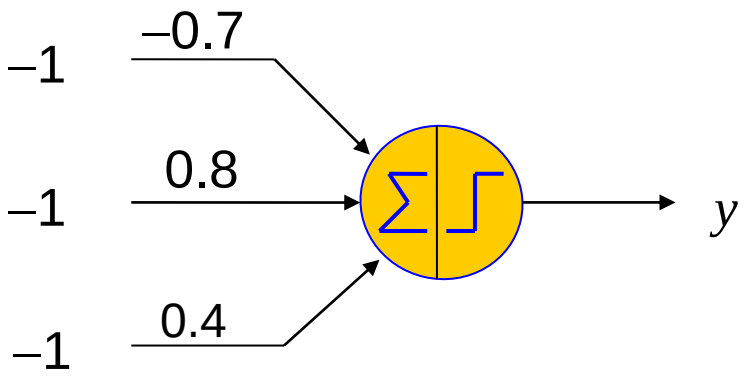
$w_1(1) = w_1(0) - 2\eta 1 = 0.3 - 1 = -0.7$

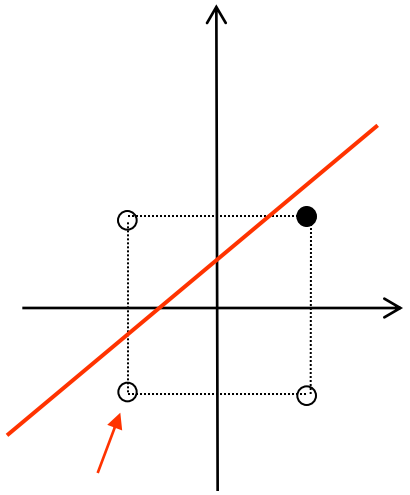
$w_2(1) = w_2(0) - 2\eta(-1) = -0.2 + 1 = 0.8$

$\theta(1) = \theta(0) - 2\eta(-1) = -0.6 + 1 = 0.4$

Elegimos η=0.5

☺ Diseña un perceptrón que implemente la función **lógica AND**





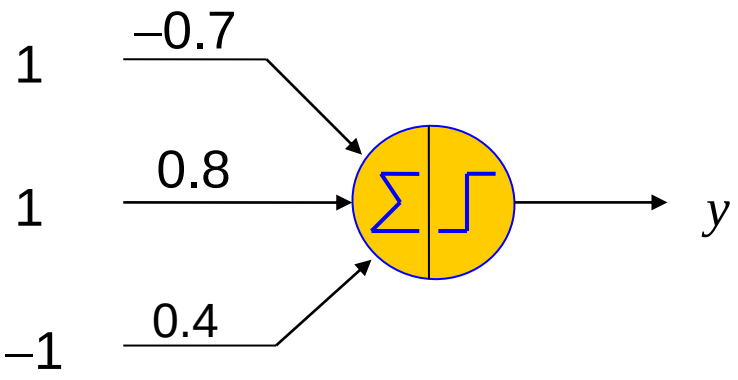
Paso 1:

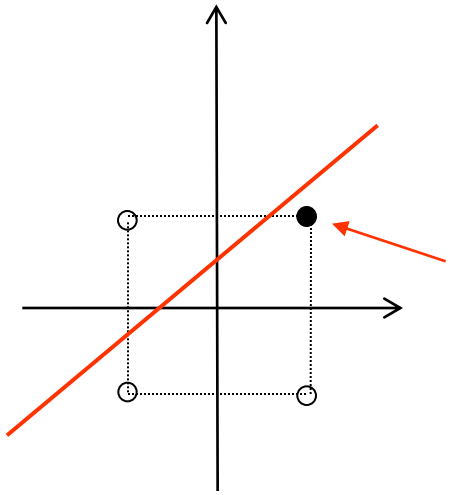
Patrón de entrada (–1,–1): $h = -0.7(-1) + 0.8(-1) + 0.4(-1) = -0.5$

Como $y = -1$ y $z = -1$ la clasificación es **correcta**

$y = -1$

☺ Diseña un perceptrón que implemente la función **lógica AND**





Paso 1:

Patrón de entrada (1,1):

$$h = -0.7(1) + 0.8(1) + 0.4(-1) = -0.3$$

$$y = -1$$

Paso 2: Corrección de los pesos sinápticos

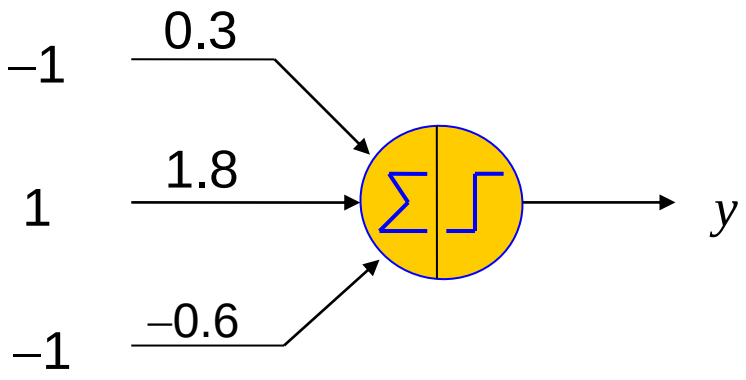
$$w_1(2) = w_1(1) + 2\eta(1) = -0.7 + 1 = 0.3$$

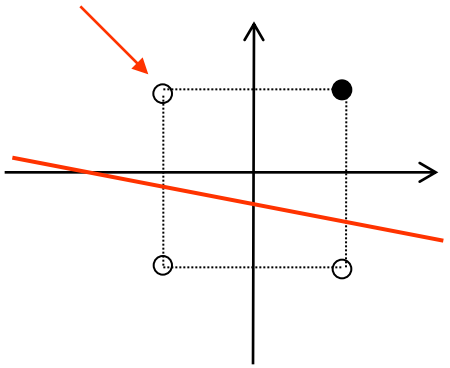
$$w_2(2) = w_2(1) + 2\eta(1) = 0.8 + 1 = 1.8$$

$$\theta(2) = \theta(1) + 2\eta(-1) = 0.4 - 1 = -0.6$$

Elegimos $\eta = 0.5$

☺ Diseña un perceptrón que implemente la función **lógica AND**





Paso 1:

Patrón de entrada (–1,1):

$$h = 0.3(-1) + 1.8(1) - 0.6(-1) = 2.1$$

$$y = 1$$

Paso 2: Corrección de los pesos sinápticos

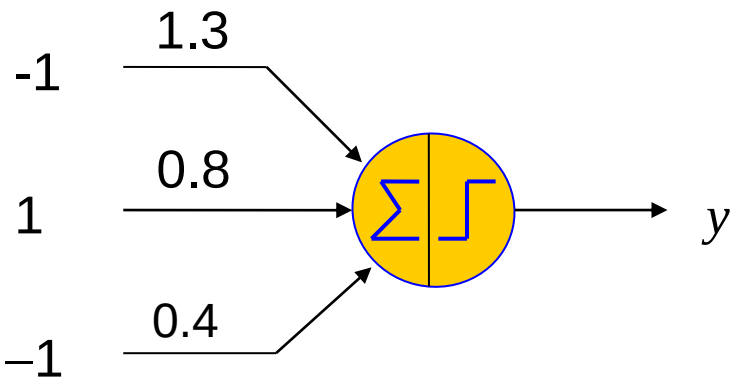
$$w_1(3) = w_1(2) - 2\eta(-1) = 0.3 + 1 = 1.3$$

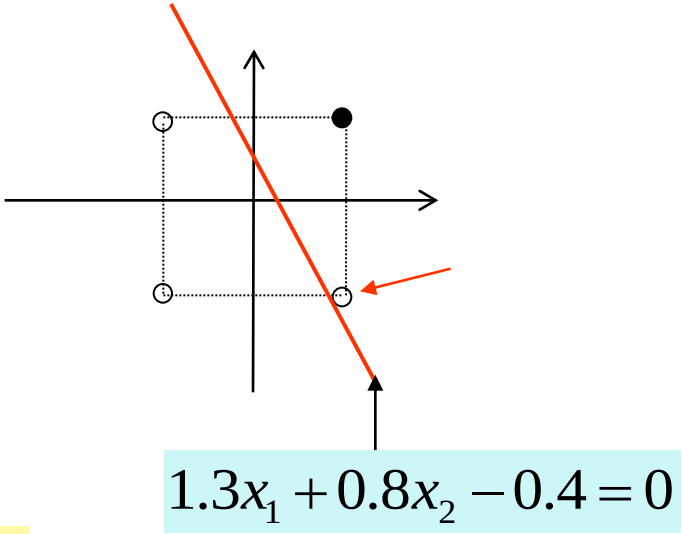
$$w_2(3) = w_2(2) - 2\eta(1) = 1.8 - 1 = 0.8$$

$$\theta(3) = \theta(2) - 2\eta(-1) = -0.6 + 1 = 0.4$$

Elegimos $\eta = 0.5$

☺ Diseña un perceptrón que implemente la función **lógica AND**





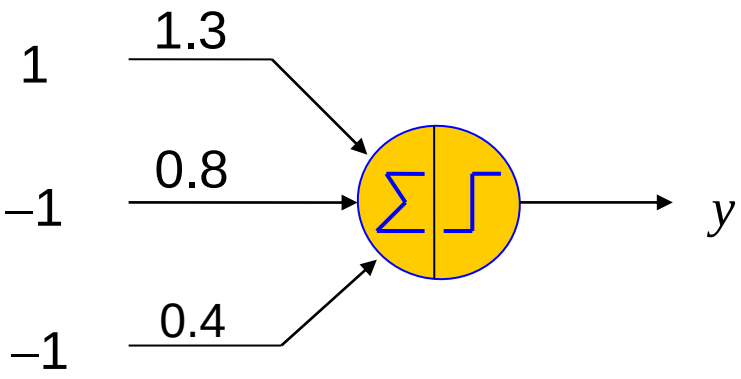
Patrón (1,1): $h = 1.3(1) + 0.8(1) + 0.4(-1) = 2.7$

Patrón (1,–1): $h = 1.3(1) + 0.8(-1) + 0.4(-1) = 0.1$

Patrón (–1,–1): $h = 1.3(-1) + 0.8(-1) + 0.4(-1) = -2.5$

Patrón (–1,1): $h = 1.3(-1) + 0.8(1) + 0.4(-1) = -0.9$

☺ Diseña un perceptrón que implemente la función **lógica AND**



Paso 1:

Patrón de entrada (1,–1):

$$h = 1.3(1) + 0.8(-1) + 0.4(-1) = 0.1$$

$$y = 1$$

Paso 2: Corrección de los pesos sinápticos

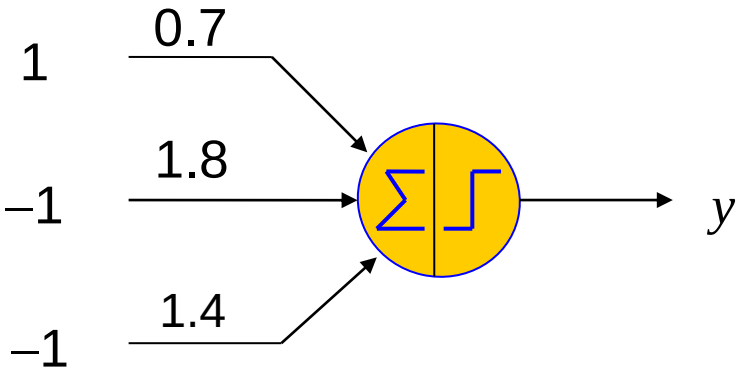
$$w_1(3) = w_1(2) - 2\eta(1) = 1.3 - 1 = 0.7$$

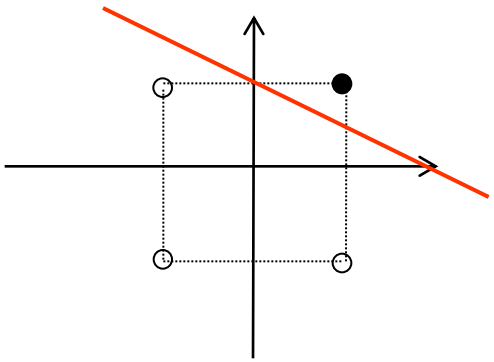
$$w_2(3) = w_2(2) - 2\eta(-1) = 0.8 + 1 = 1.8$$

$$\theta(3) = \theta(2) - 2\eta(-1) = 0.4 + 1 = 1.4$$

Elegimos $\eta = 0.5$

☺ Diseña un perceptrón que implemente la función **lógica AND**





Paso 1:

Patrón de entrada (1,–1):

$$h = 1.3(1) + 0.8(-1) + 0.4(-1) = 0.1$$

$$y = 1$$

Paso 2: Corrección de los pesos sinápticos

$$w_1(3) = w_1(2) - 2\eta(1) = 1.3 - 1 = 0.7$$

$$w_2(3) = w_2(2) - 2\eta(-1) = 0.8 + 1 = 1.8$$

$$\theta(3) = \theta(2) - 2\eta(-1) = 0.4 + 1 = 1.4$$

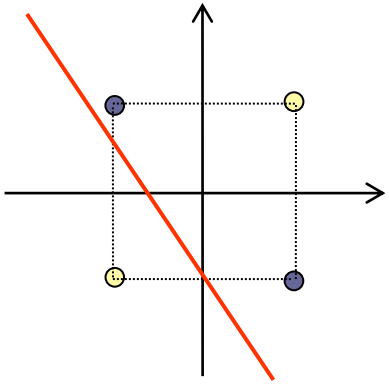
Elegimos $\eta = 0.5$

El Perceptrón

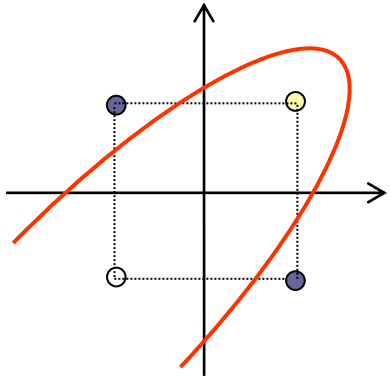
☹ ¿Dado un conjunto cualquiera de patrones de entrenamiento, puede el Perceptrón aprender a clasificarlos correctamente?

Problema XOR

Entradas	Salidas
(1, 1)	–1
(1, –1)	1
(–1, 1)	1
(–1,–1)	–1

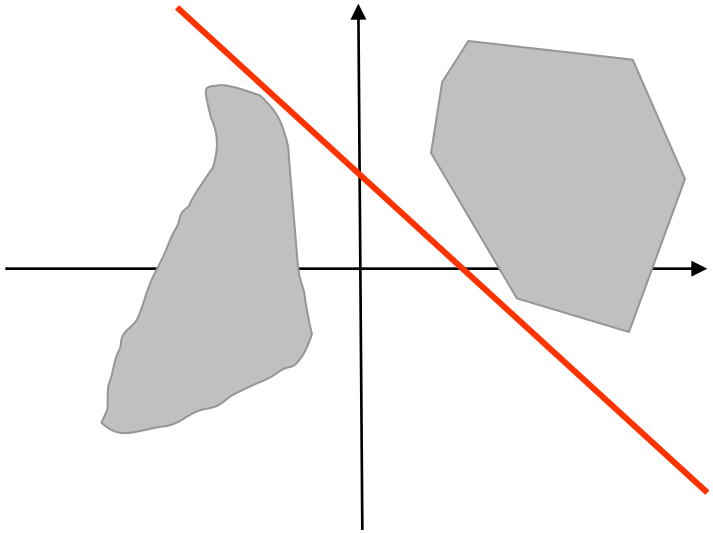


(a)



(b)

Conjuntos separables linealmente



Teorema de convergencia del Perceptrón

Si el conjunto de patrones de entrenamiento con sus salidas deseadas,

$\{\mathbf{x}^1, z^1\}, \{\mathbf{x}^2, z^2\}, \ldots, \{\mathbf{x}^p, z^p\},$

es **linealmente separable** entonces el Perceptrón simple encuentra una solución en un **número finito de iteraciones**

Demostración

Como es **linealmente separable** entonces existen $w_1^*, w_2^*, \ldots, w_{n+1}^*$

θ

$$\sum_{j=1}^{n} w_j^* x_j > w_{n+1}$$ si son de la clase C_1

$$\sum_{j=1}^{n} w_j^* x_j < w_{n+1}$$ si son de la clase C_2