Aprendizado de Máquina

Perceptrons

Profa. Dra. Roseli Aparecida Francelin Romero
SCC - ICMC - USP

2020

Sumário

Perceptron

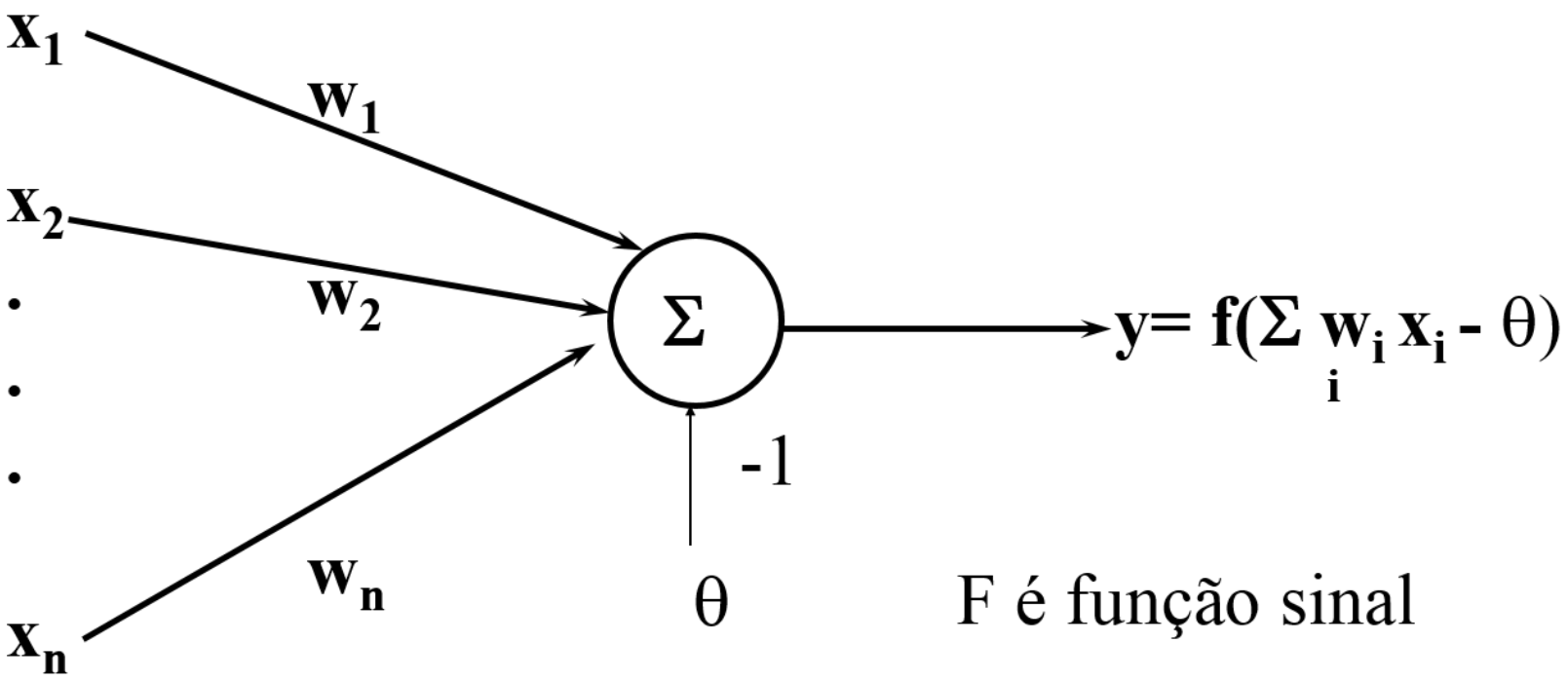


Figura 1: Modelo para representação do *Perceptron*.

Perceptron

- O algoritmo usado para ajustar os parâmetros livres desta rede apareceu num processo de aprendizado desenvolvido por Rosemlbatt (1958, 1962).
- Ele provou que se os padrões usados para treinar são **linearmente separáveis**, então o algoritmo converge e a superfície de decisão tem a forma de um hiperplano entre duas classes.

Perceptron

- É constituído de apenas 1 neurônio e, como tal, limita-se a classificar padrões envolvendo apenas 2 classes, que devem ser linearmente separáveis.
- A regra de decisão é designar x à classe $\mathcal{C}_1$, se a saída é $y = +1$, ou à classe $\mathcal{C}_2$, se a saída é $y = -1$.
- Existem duas regiões separadas pelo hiperplano:
  - $\sum w_i x_i - \theta = 0$
- Se o espaço for o $\mathbb{R}^2$, a região de separação é uma reta.

Sumário

Sumário

Estrutura básica de um neurônio artificial

- **Estado de ativação (saída):** s_j
- **Conexões entre processadores:** w_{ij}
  - a cada conexão existe um peso sináptico que determina o efeito da entrada sobre o processador.
- **Soma:** cada processador soma os sinais de entrada ponderado pelo peso sináptico das conexões
- **Função de ativação:** $s_j = F(net_j)$
  - determina o novo valor do *estado de ativação* do processador.

Funções de transferência

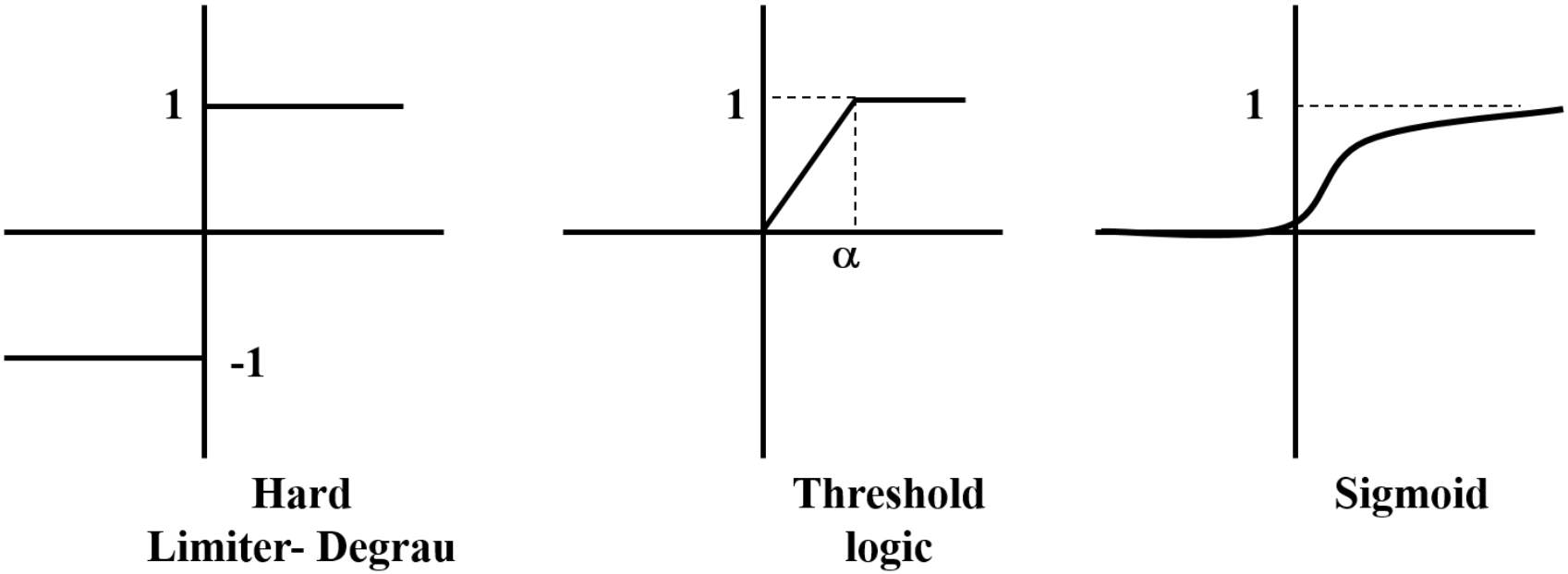


Figura 2: Exemplos de funções de transferência usadas em redes neurais artificiais.

Modelos de neurônios

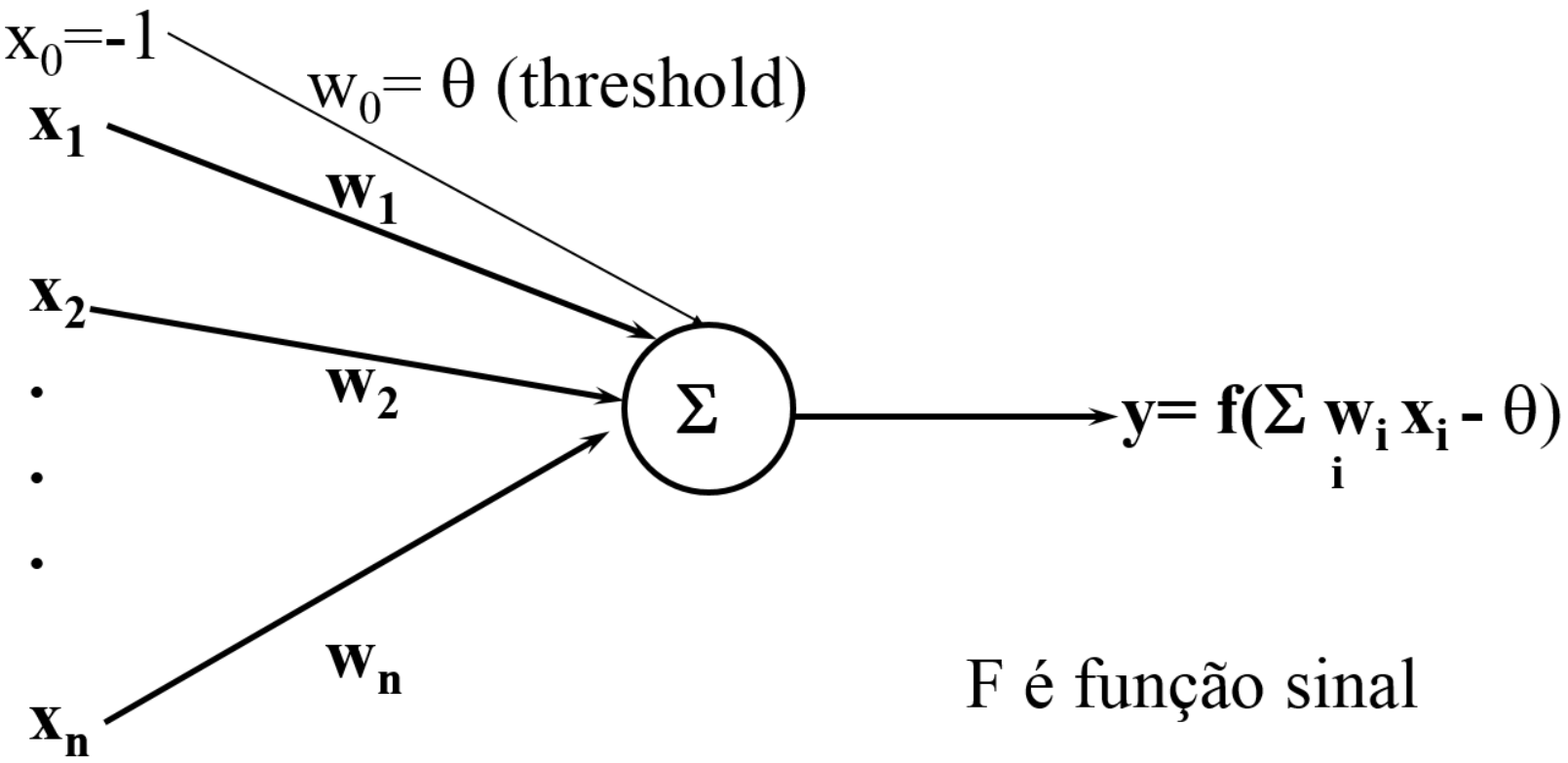


Figura 3: Modelo de um neurônio com $x_0 = -1$.

Modelos de neurônios

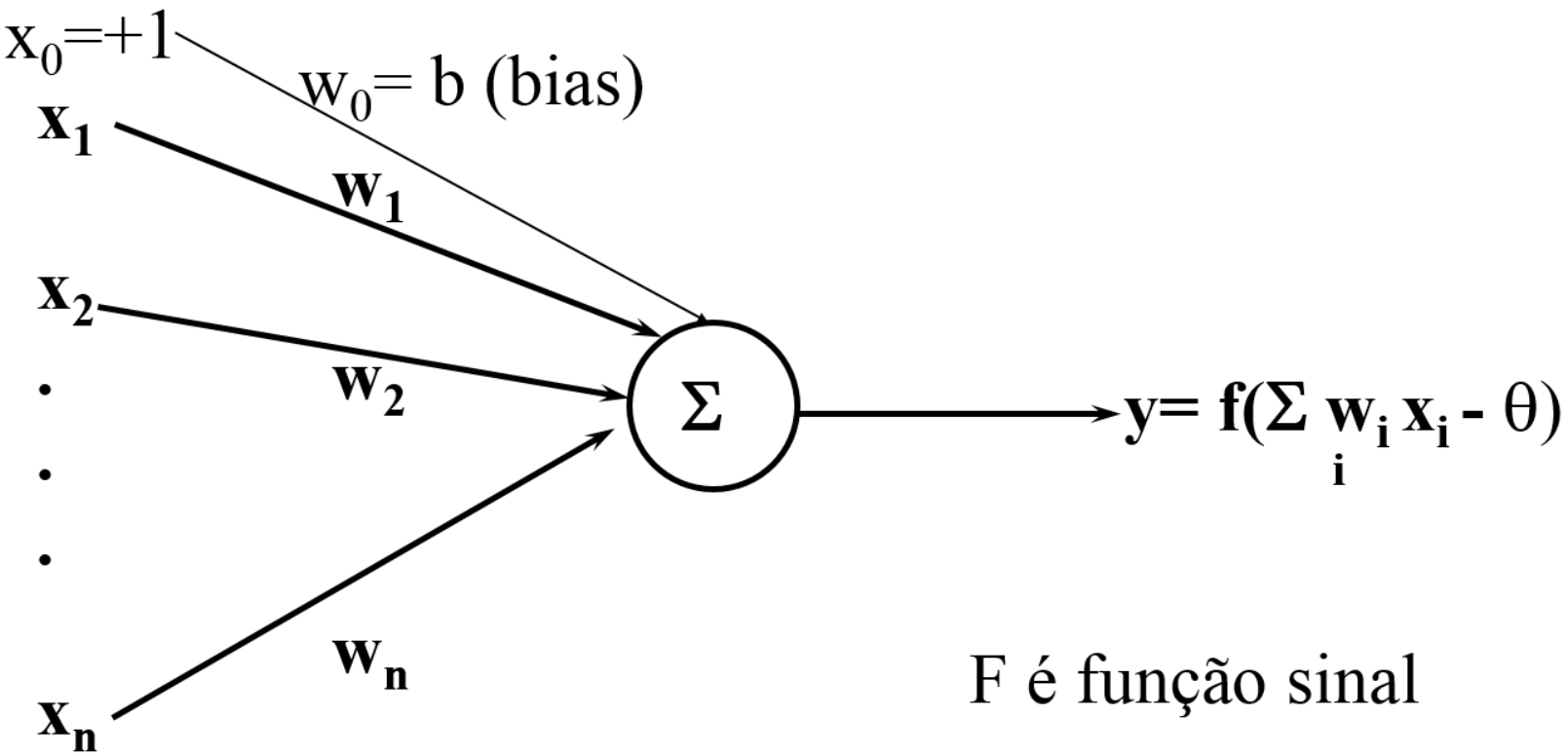


Figura 4: Modelo de um neurônio com $x_0 = +1$.

Sumário

Características básicas

- **Regra de propagação:** $y_j = sgn(\sum_i x_i w_{ij})$
- **Função de ativação:** função sinal
- **Topologia:** uma única camada de processadores.
- **Algoritmo de aprendizado:** $\Delta w_{ij} = \eta x_i (t_j - y_j)$
  - (é do tipo supervisionado)
- **Valores de entrada/saída:** binários $\rightarrow$ $t = 1$ ou $t = -1$

Finalidade do termo *bias*

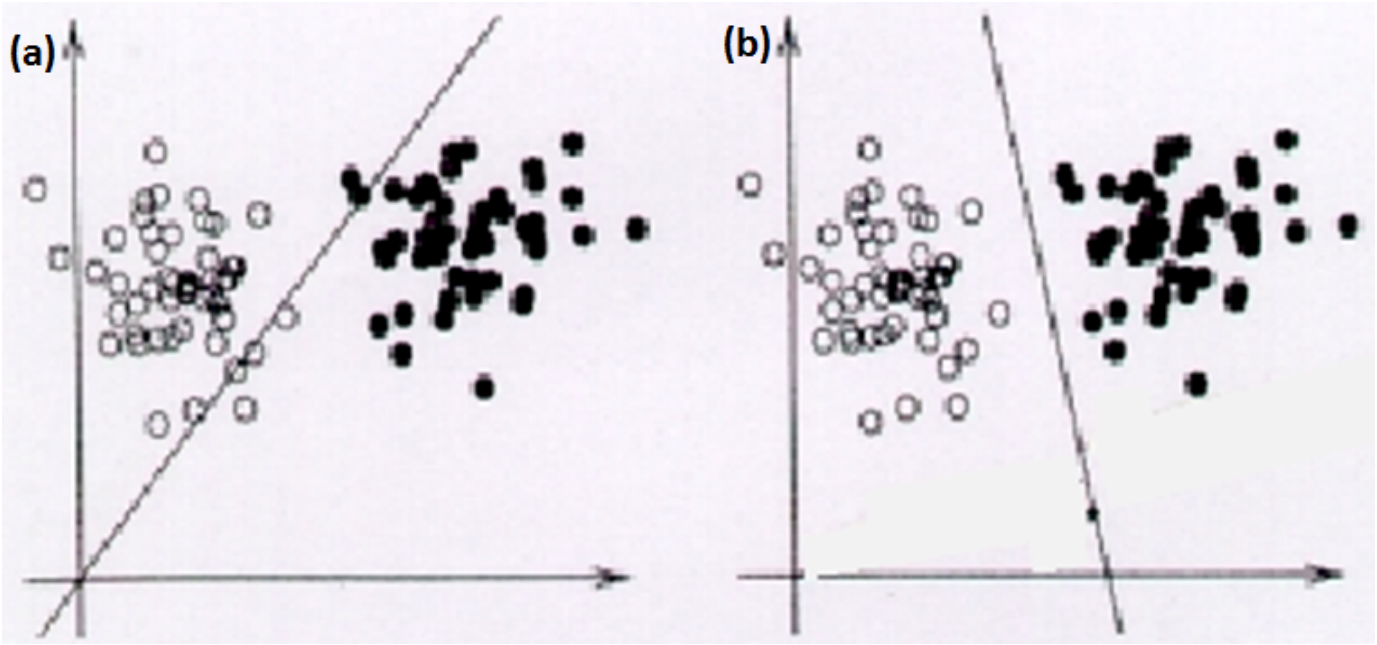


Figura 5: Hiperplano obtido: (a) sem *bias*; (b) com *bias*.

- $\sum_i x_i w_{ij} = 0 \rightarrow$ define um hiperplano passando pela origem.
- $\sum_i x_i w_{ij} + \theta_i = 0 \rightarrow$ desloca o hiperplano da origem.

Sumário

Adaline

- O processo adaptativo do Adaline consiste em utilizar a função de ativação *hard limiter* (saída $+1$ ou -1) e minimizar os pesos usando o algoritmo LMS.

Regra Delta - LMS

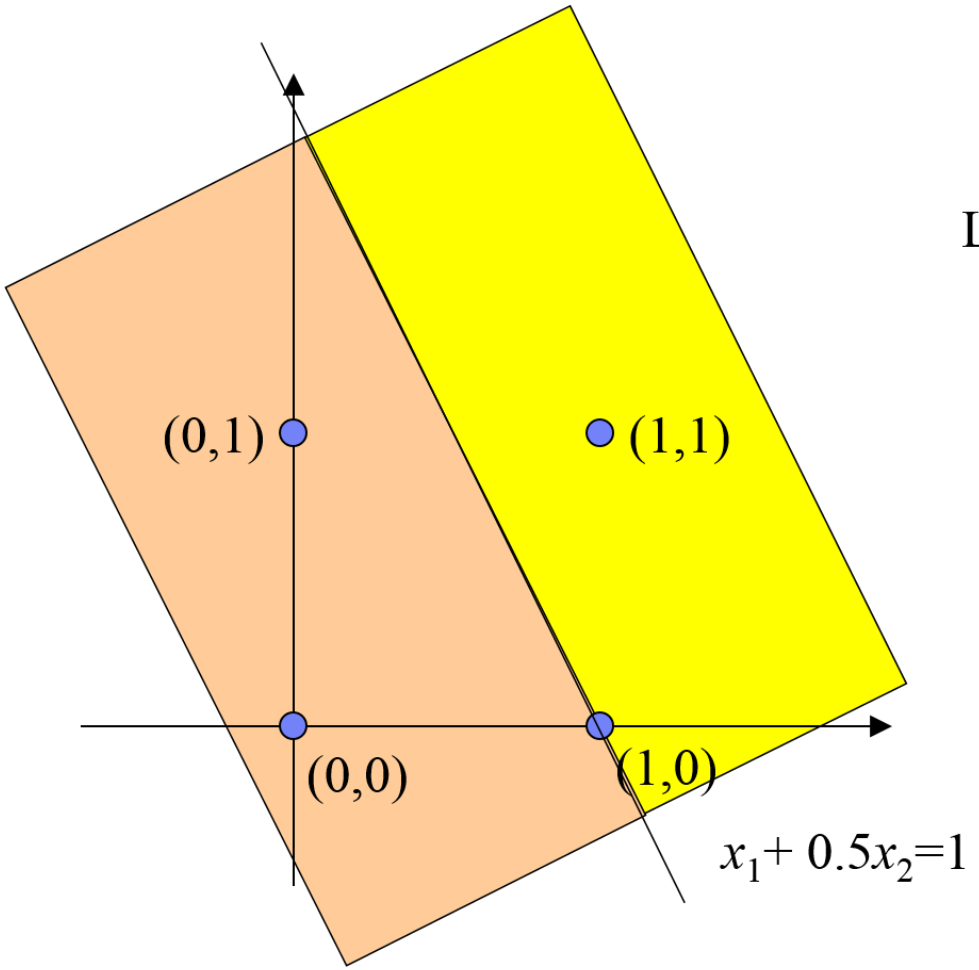
1. Iniciar os pesos sinápticos com valores randômicos pequenos ou iguais a zero.
2. Aplicar um padrão com seu respectivo valor esperado de saída (t_j) e verificar a saída da rede (y_j).
3. Calcular o erro na saída: $E_j = t_j - y_j$
4. Se $E_j = 0$, voltar ao passo 2
   Se $E_j \neq 0$, atualizar os pesos: $\Delta w_{ij} = \eta x_i E_j$
5. Voltar ao passo 2.

Regra Delta - LMS

- **Importante:**
  - Não ocorre variação no peso se a saída estiver correta.
  - Caso contrário, cada peso é incrementado de η quando a saída é maior que o valor-alvo.

$$\Delta w_{ij} = \eta x_i e_j \tag{1}$$

Interpretação geométrica



Linha de Decisão:

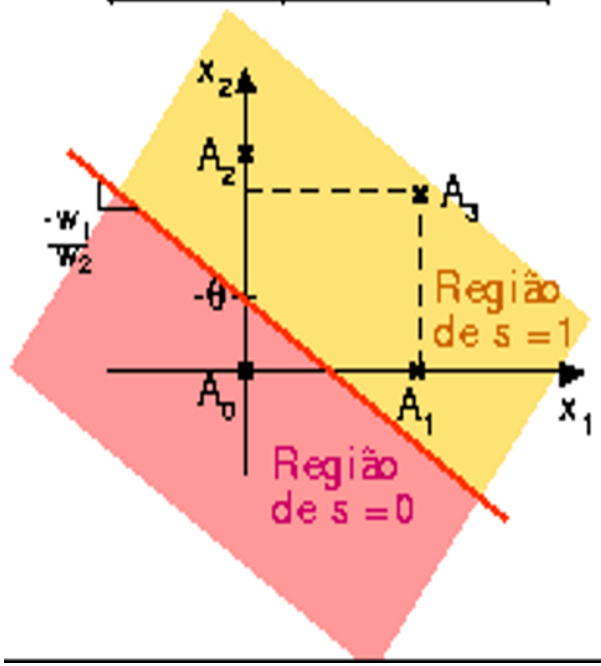
$$x_1 w_1 + x_2 w_2 = -\theta$$

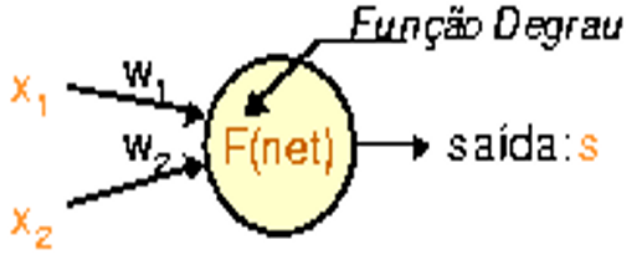
$$x_1 + 0.5\, x_2 = 1$$

Sumário

O problema do OU exclusivo (XOR)

PONTO	X_1	X_2	Saída
A_0	0	0	0
A_1	0	1	1
A_2	1	0	1
A_3	1	1	0



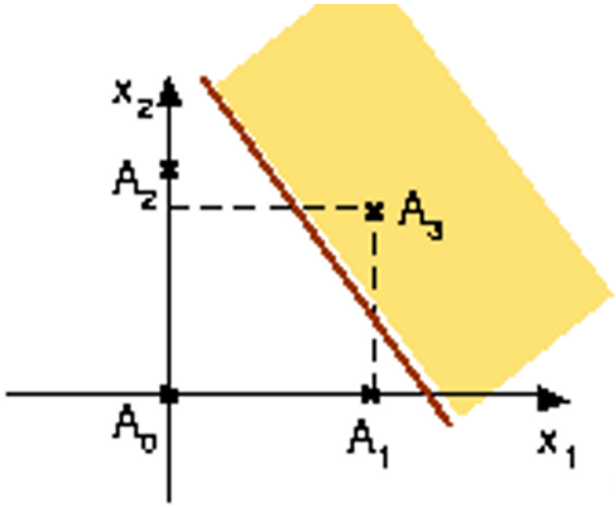


De acordo com a definição do neurônio: $s = F(x_1w_1 + x_2w_2 + \theta)$

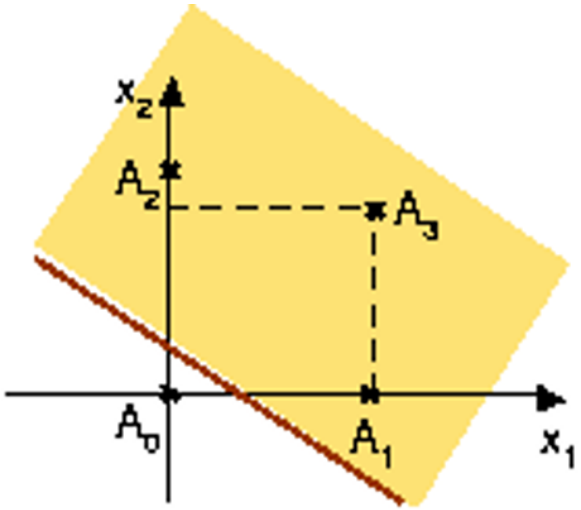
$$net = x_1w_1 + x_2w_2 + \theta \rightarrow \begin{cases} \text{Se net} \geq 0 \rightarrow s = 1 \\ \text{Se net} < 0 \rightarrow s = 0 \end{cases}$$

A rede Perceptron divide o plano $X_1 \times X_2$ *em duas regiões (através da reta net)*

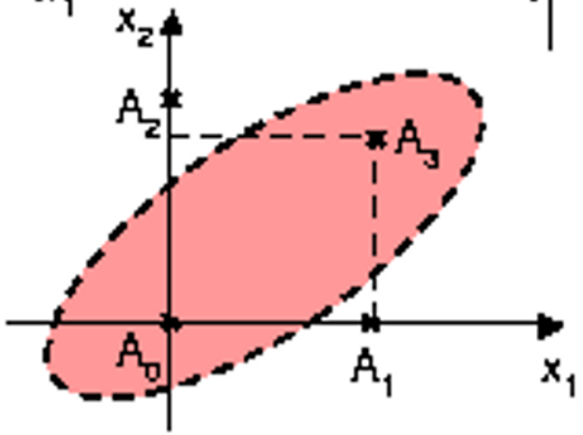
O problema do OU exclusivo (XOR)



Função AND



Função OR



Função OU-Exclusivo

O problema do OU exclusivo (XOR)

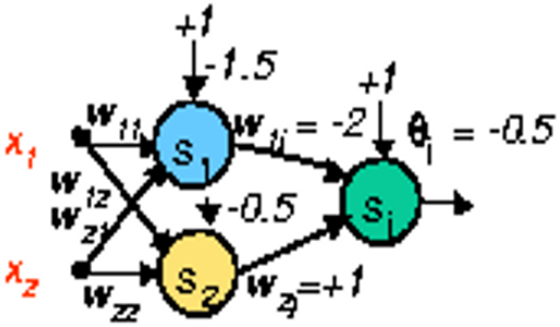
- Mudando-se os valores de w_1, w_2 e θ, muda-se a inclinação da reta.
- Entretanto, é impossível achar uma reta que divida o plano de forma a separar os pontos A_1 e A_2 de um lado e A_0 e A_3 de outro.
- Redes de uma única camada só representam **funções linearmente separáveis**.

O problema do OU exclusivo (XOR)

- Minsky & Papert provaram que esse problema pode ser solucionado adicionando-se uma outra camada intermediária de processadores $\rightarrow$ Multi-Layer Perceptron (MLP).

O problema do OU exclusivo (XOR)

Exemplo:



$w_{11} = w_{12} = w_{21} = w_{22} = +1$

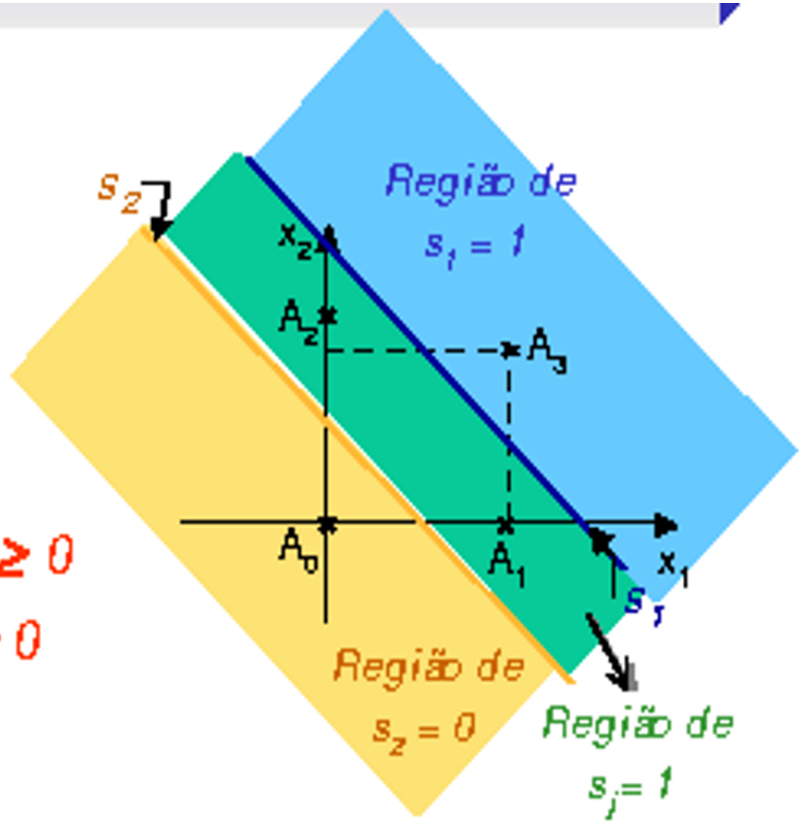
$s_j = 1 \rightarrow s_1 w_{1j} + s_2 w_{2j} + \theta_j \geq 0$

$-2s_1 + s_2 - 0.5 \geq 0$

$-2s_1 + s_2 \geq 0.5$

$\downarrow$

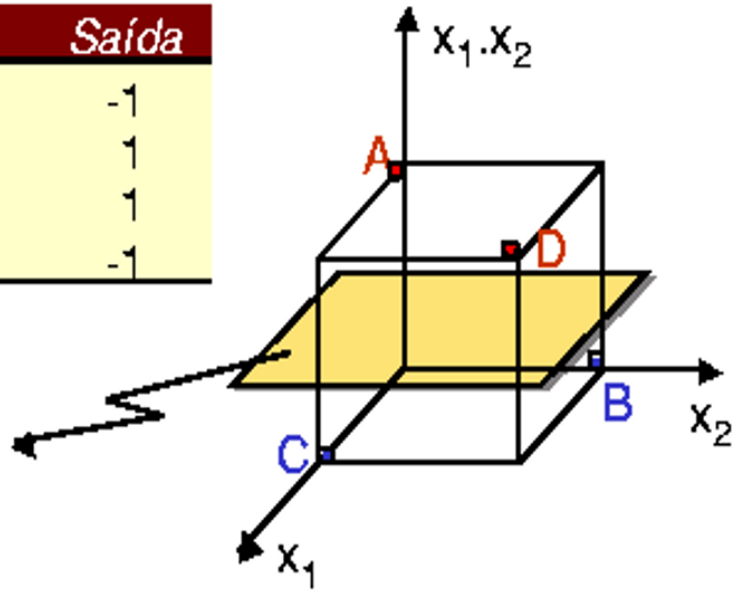
s_1 é inibitório
s_2 é excitatório



O problema do OU exclusivo (XOR)

- *Exemplo do OU-EXCLUSIVO:*
  - J = 2 (número de entradas originais - x_1,x_2)
  - H = 1 (número de entradas adicionais - $x_1.x_2$)

Pontos	Entradas				Saída
A	-1	-1	1	⇨	-1
B	-1	1	-1	⇨	1
C	1	-1	-1	⇨	1
D	1	1	1	⇨	-1



Problema Linearmente Separável

Multi-Layer Perceptron

- Redes de apenas uma camada só representam funções linearmente separáveis.
- Redes de múltiplas camadas solucionam essa restrição.
- O desenvolvimento do algoritmo *backpropagation* foi um dos motivos para o ressurgimento da área de redes neurais.