FUNDAMENTOS DA INTELIGÊNCIA ARTIFICIAL

Aula 02 - Perceptron

Prof. Rafael G. Mantovani

Londrina - PR, Brasil
Outubro, 2018

Universidade Estadual de Londrina - UEL
Departamento de Computação - DC

Roteiro

1. Introdução
2. Perceptron
3. Teorema de Convergência
4. Algoritmo de Treinamento Perceptron
5. Exemplo / Exercício
6. Síntese / Próximas Aulas
7. Referências

Roteiro

1. **Introdução**
2. **Perceptron**
3. **Teorema de Convergência**
4. **Algoritmo de Treinamento Perceptron**
5. **Exemplo / Exercício**
6. **Síntese / Próximas Aulas**
7. **Referências**

Relembrando

- O que vimos na aula passada ?

Relembrando

- Paradigma Conexionista
- Redes Neurais Artificiais
- Inspiração Biológica (estrutura do cérebro)
- Neurônio artificial
- Funções de Ativação
- Topologias
- Algoritmos de Aprendizado

Introdução

- Perceptron (Rosenblatt, 1958):
  - primeira rede neural descrita algoritmicamente
  - Frank Rosenblatt (psicólogo)
  - modelo mais simples de rede neural que existe

Introdução

- Classifica padrões linearmente separáveis

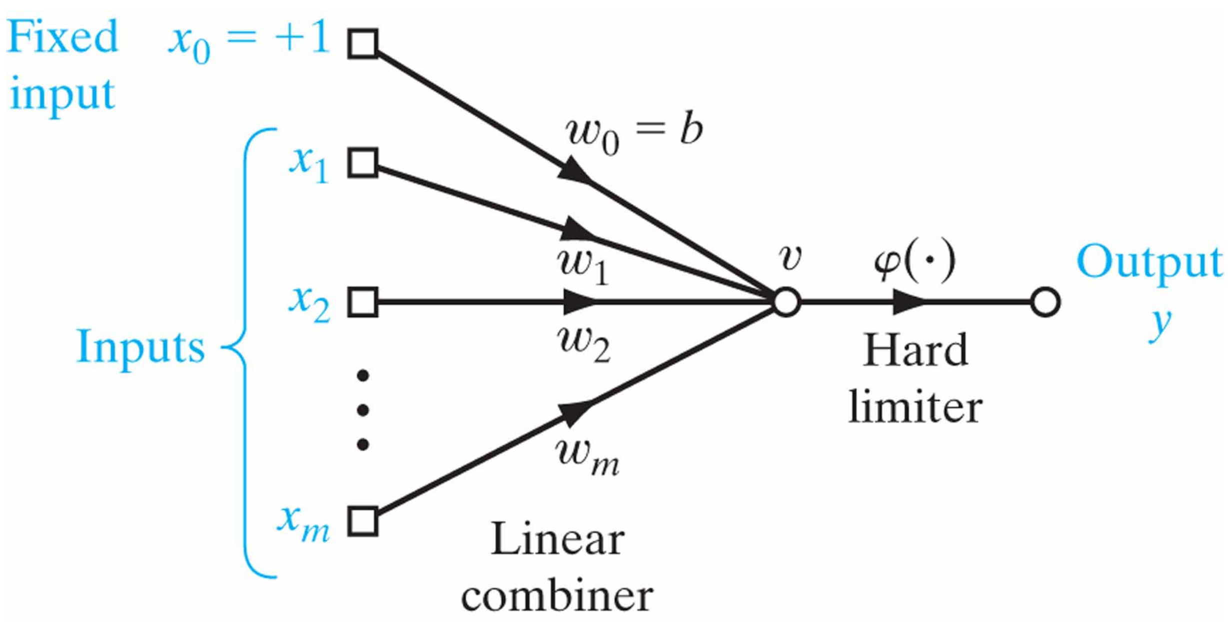
- um único neurônio com pesos sinópticos ajustáveis e bias

- Rosenblatt → algoritmo:
  - Ajuste dos parâmetros livres da rede (pesos sinápticos)
  - provou que se os exemplos utilizados no treino forem linearmente separáveis, o algoritmo converge, posicionando um hiperplano entre as duas classes

Roteiro

Perceptron

- Perceptron → Neurônio de McCulloch-Pitts



Perceptron

- Quais são os elementos manipulados?

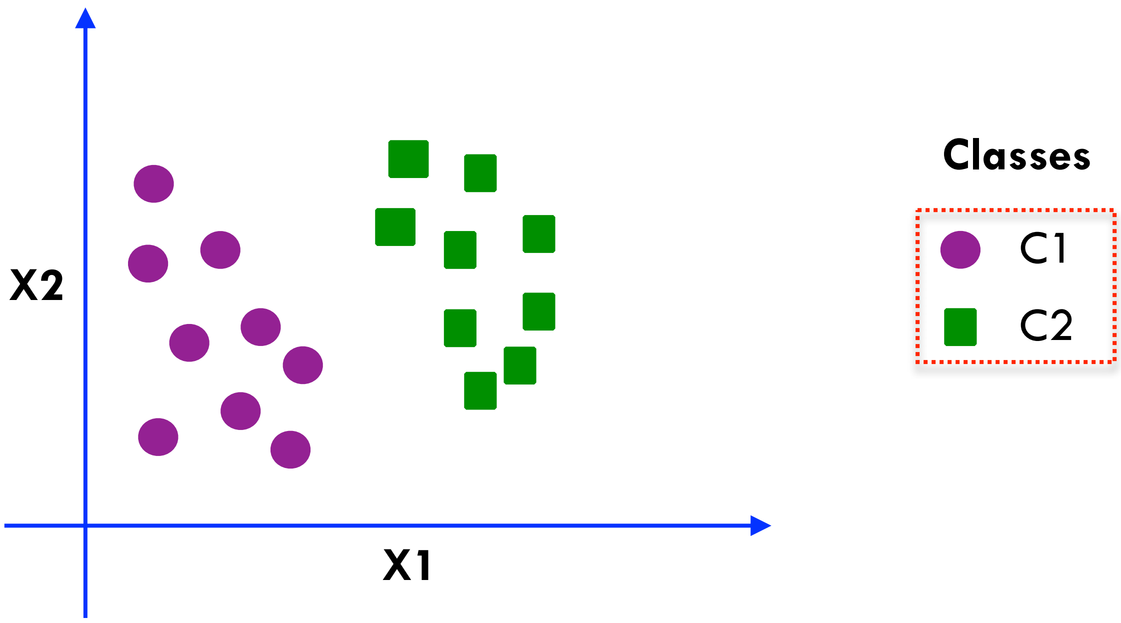
Perceptron

$$v_k = \sum_{j=0}^{m} w_{kj} x_j \qquad \text{e} \qquad y_k = \varphi(v_k)$$

- X são os sinais de entrada
- W são os pesos sinápticos do neurônio k
- v_k é a combinação linear de W e X (entradas)
- b_k é o bias
- $\varphi(.)$ é a função de ativação
- y_k é a saída do neurônio

Perceptron

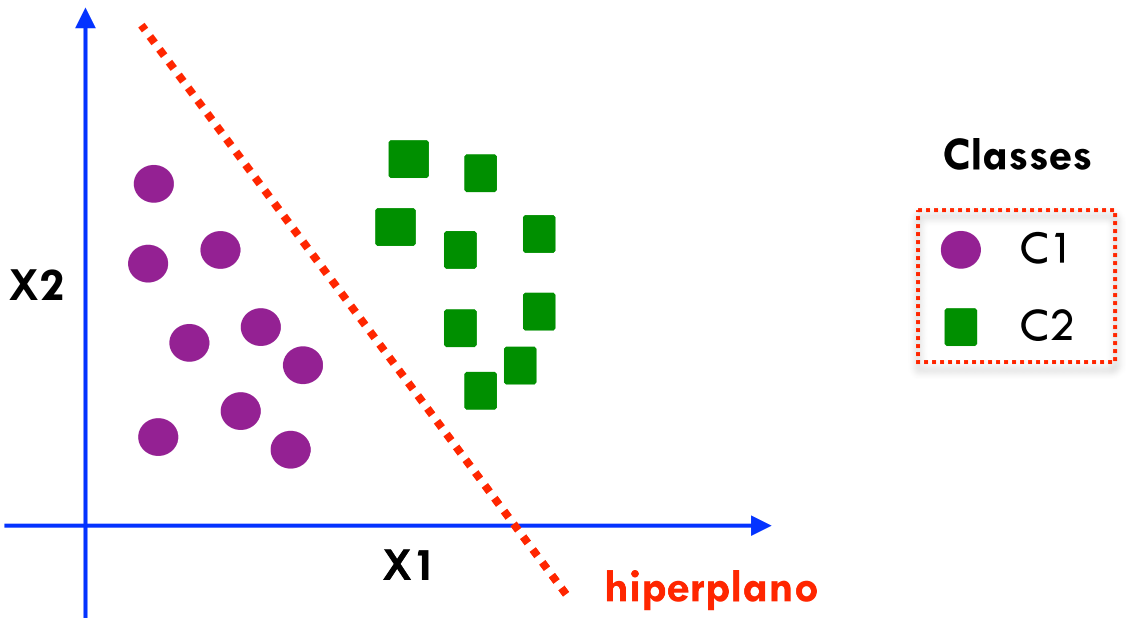
- **Objetivo:** classificar corretamente um conjunto de exemplos X em uma de duas classes C1 ou C2



Classificação

Perceptron

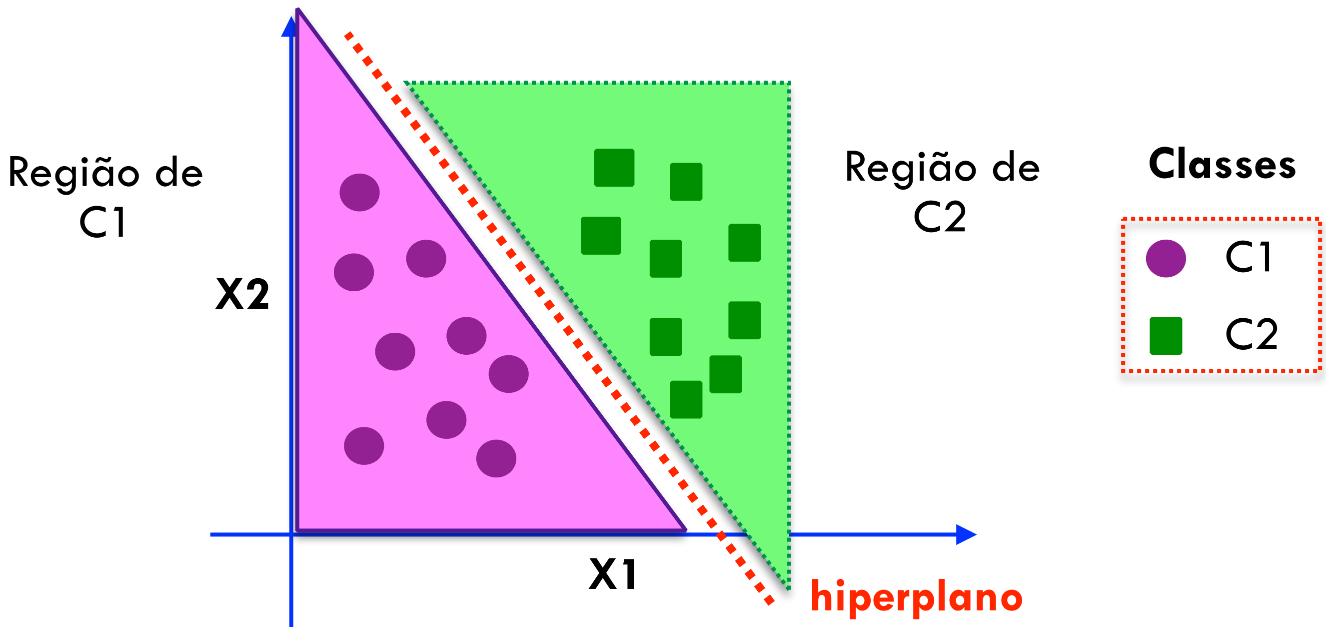
- **Objetivo:** classificar corretamente um conjunto de exemplos X em uma de duas classes C1 ou C2



Classificação

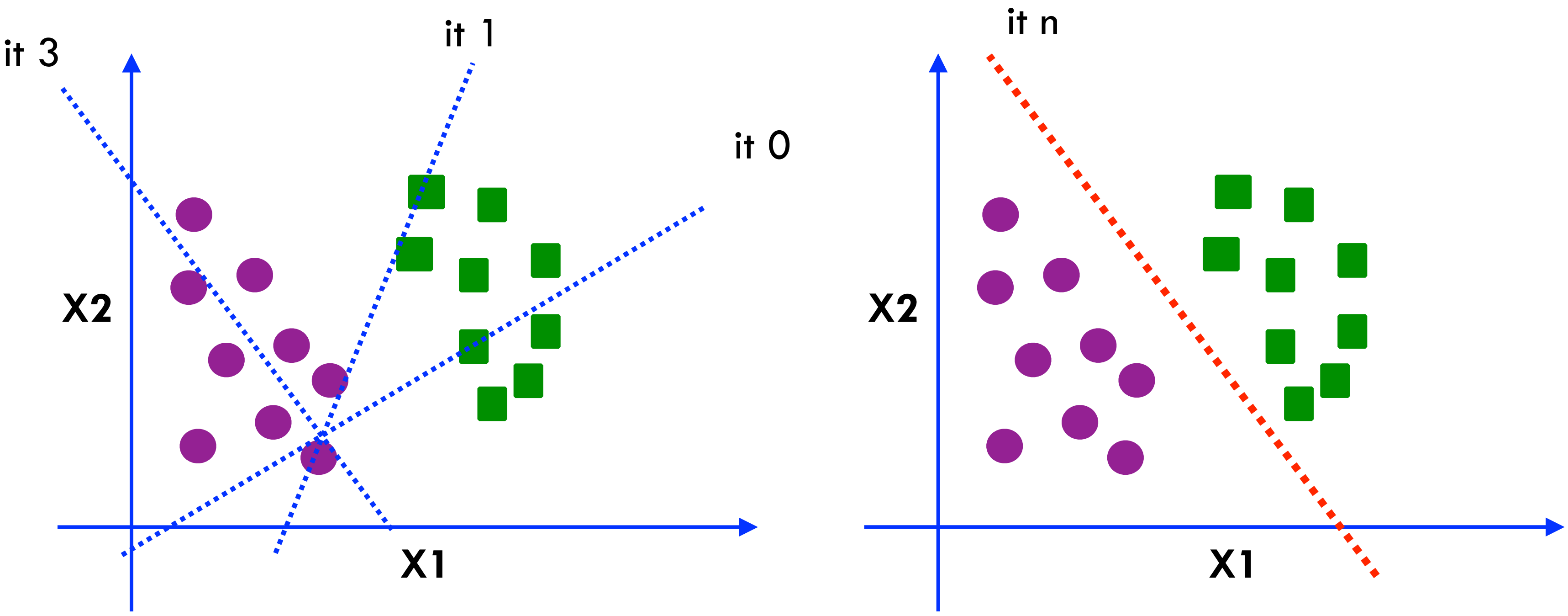
Perceptron

- **Objetivo:** classificar corretamente um conjunto de exemplos X em uma de duas classes C1 ou C2



Perceptron

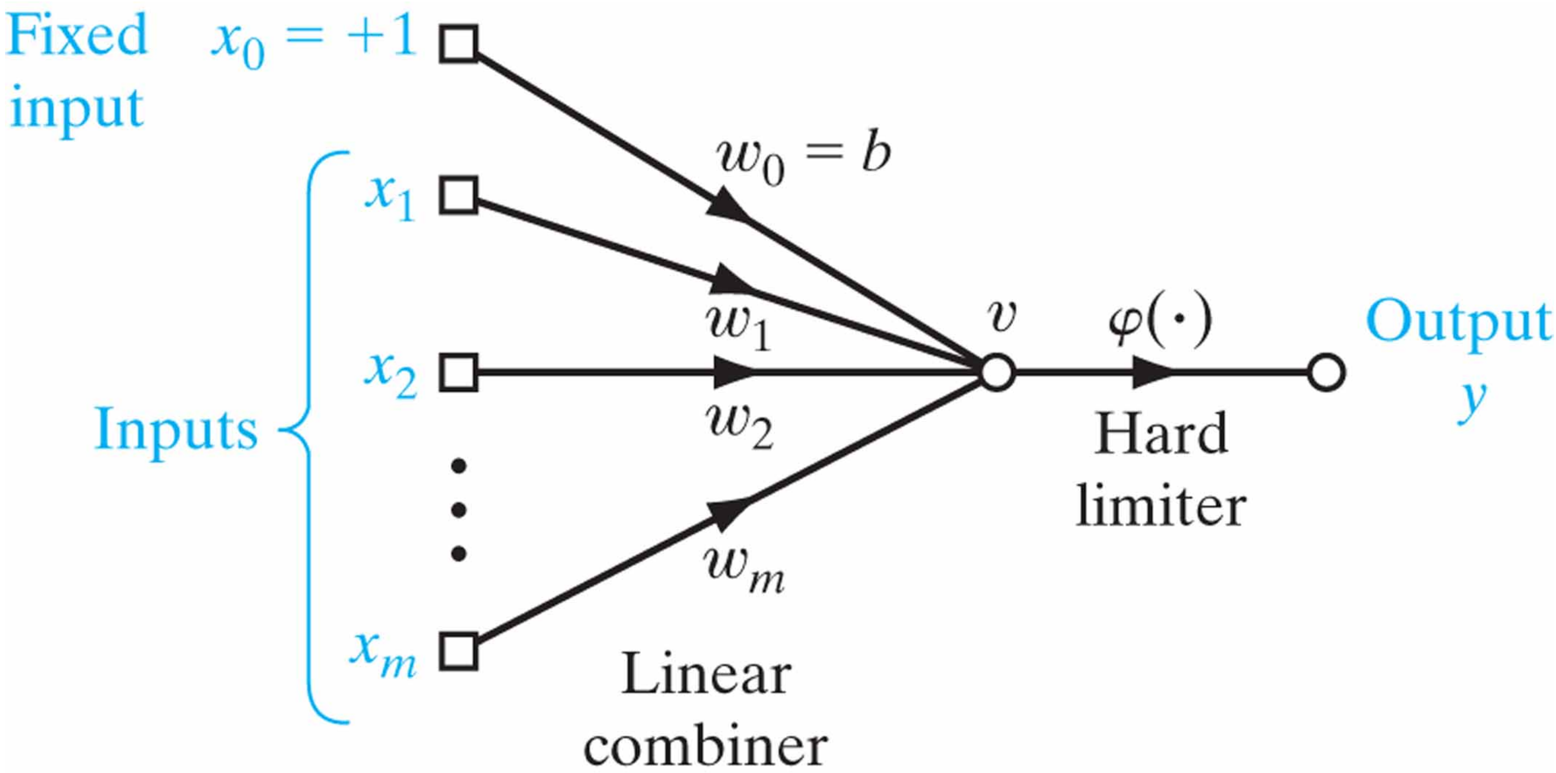
- **Aprendizado:** ajuste iterativo dos pesos sinápticos usando o algoritmo de convergência do perceptron



Roteiro

1. Introdução
2. Perceptron
3. **Teorema de Convergência**
4. Algoritmo de Treinamento Perceptron
5. Exemplo / Exercício
6. Síntese / Próximas Aulas
7. Referências

Teorema de Convergência

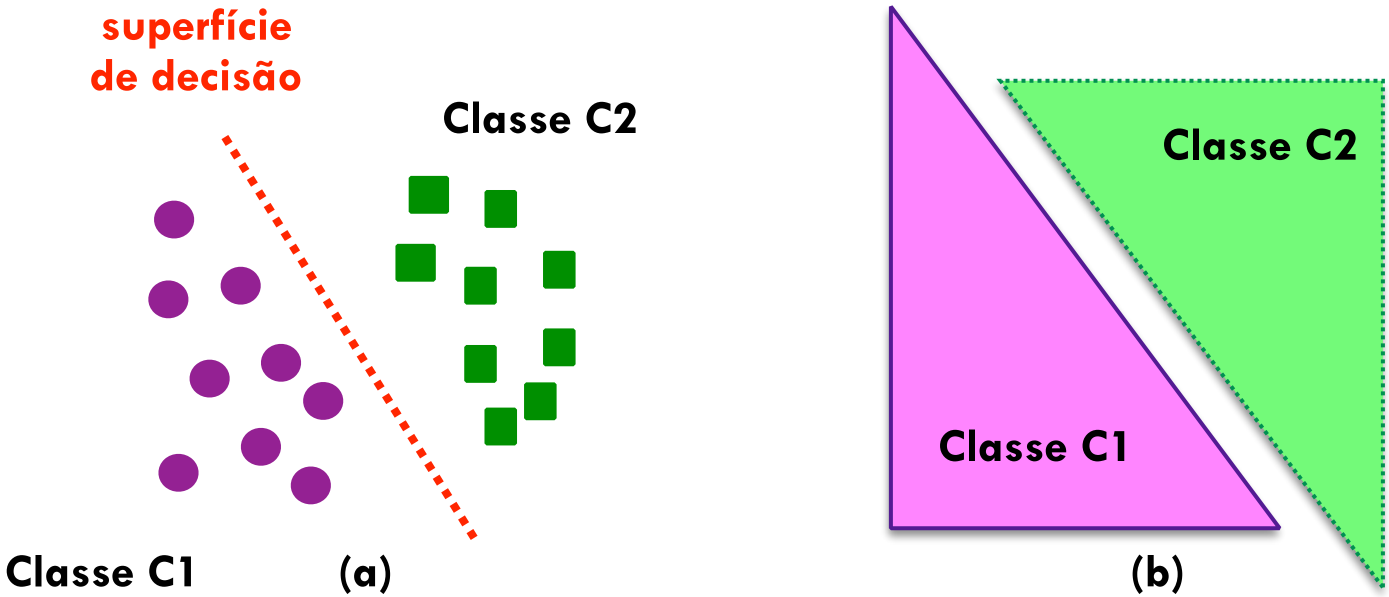


Teorema de Convergência

- **bias b(n):** é um peso w_0 associado a uma entrada +1
- **vetor de entrada X(n):** $[+1, x_1(n), x_2(n), \ldots, x_m(n)]^T$
- **vetor de pesos W(n):** $[b, w_1(n), w_2(n), \ldots, w_m(n)]^T$

$$v(n) = \sum_{i=0}^{m} w_i(n) x_i(n) = \mathbf{w}^T(n)\mathbf{x}(n)$$

Teorema de Convergência



$\mathbf{w}^T\mathbf{x} = 0$ define um hiperplano de separação

$\mathbf{w}^T\mathbf{x} > 0$ para todo vetor $\mathbf{x}$ pertencente à classe

$\mathbf{w}^T\mathbf{x} \leq 0$ para todo vetor $\mathbf{x}$ pertencente à classe

Teorema de Convergência

- Se o n-ésimo vetor $\mathbf{x}(n)$ é corretamente classificado pelo vetor $\mathbf{w}(n)$ na n-ésima iteração do algoritmo, nenhuma correção é feita no vetor de pesos:
  - $\mathbf{w}(n+1) = \mathbf{w}(n)$ se $\mathbf{w}^T\mathbf{x}(n) > 0$ e $\mathbf{x}(n) \ni$ a classe C1
  - $\mathbf{w}(n+1) = \mathbf{w}(n)$ se $\mathbf{w}^T\mathbf{x}(n) \leq 0$ e $\mathbf{x}(n) \ni$ a classe C2

- Caso contrário, o vetor de pesos é atualizado:
  - $\mathbf{w}(n+1) = \mathbf{w}(\mathrm{n})-\eta(\mathrm{n})\mathbf{x}(\mathrm{n})$ se $\mathbf{w}^T\mathbf{x}(n)>0$ e $\mathbf{x}(n) \ni$ classe C2
  - $\mathbf{w}(n+1) = \mathbf{w}(\mathrm{n})+\eta(\mathrm{n})\mathbf{x}(\mathrm{n})$ se $\mathbf{w}^T\mathbf{x}(n) \leq 0$ e $\mathbf{x}(n) \ni$ classe C1

- η é a taxa de aprendizado que controla o ajuste dos pesos
  - hiper-parâmetro do algoritmo
  - parâmetro x hiper-parâmetro

Teorema de Convergência

- A saída do neurônio é computada usando a função sinal sgn(.):

função sinal

$$sgn(v) = \begin{cases} +1 \text{ se } v > 0 \\ -1 \text{ se } v < 0 \end{cases}$$

- Expressamos a saída y(n) de maneira compacta:

$$y(n) = sgn[\mathbf{w}^T(n)\mathbf{x}(n)]$$

Teorema de Convergência

- Regra de Atualização dos Pesos sinápticos:

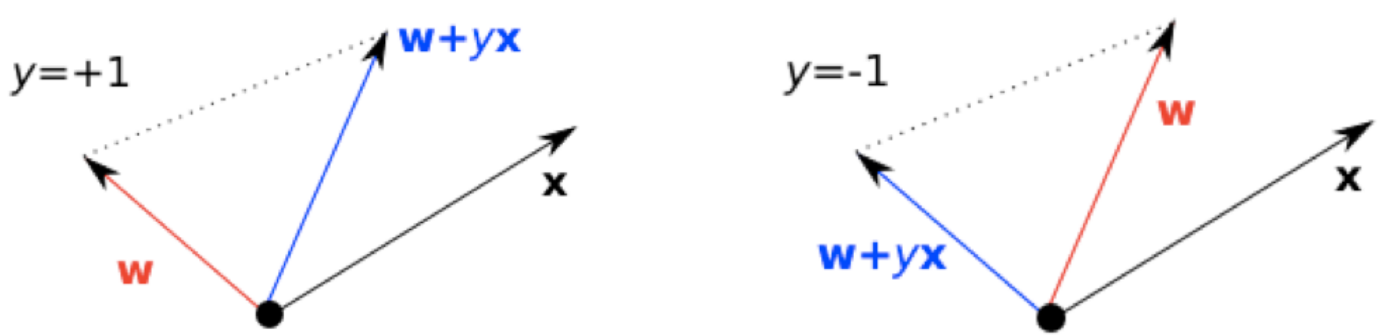
    - $w(n+1) \leftarrow w(n) + \eta\,(d(n) - y(n))\,x(n)$

- $d(n) - y(n)$: sinal de erro

Teorema de Convergência

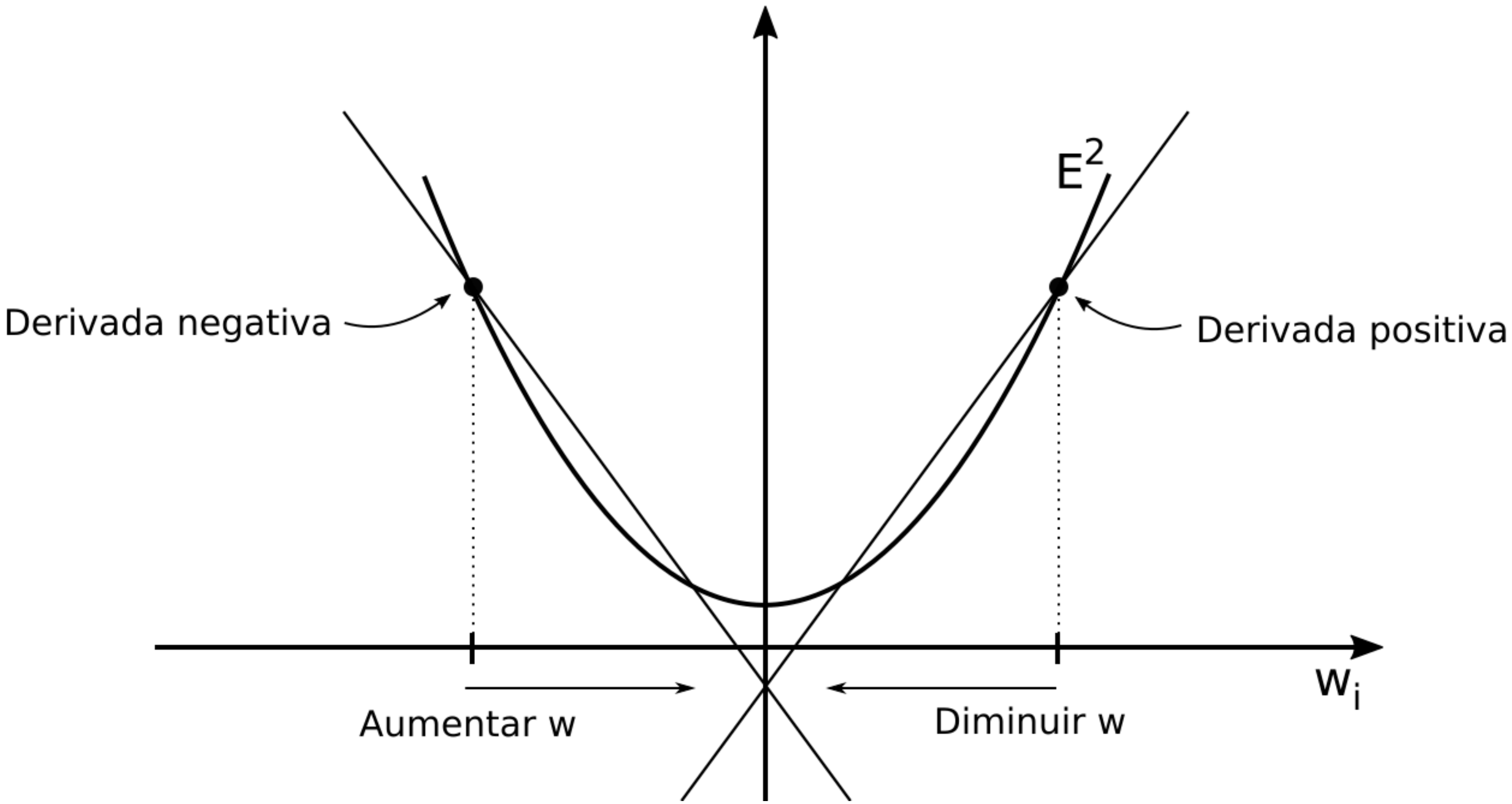
- Os pesos são corrigidos de acordo com o valor do produto interno $\mathbf{w}^T(n)\ \mathbf{x}(n)$

- Se o produto interno, na iteração **n**, tiver um sinal errado, os pesos devem ser ajustados para classificar o exemplo corretamente na iteração n+1



Gradiente descendente

$$E^2 = \left(d(n) - y(n)\right)^2 = \left(d(n) - \mathbf{w}^T(n)\mathbf{x}(n)\right)^2$$



Gradiente descendente

$$\boxed{E^2 = \left(d(n) - y(n)\right)^2 = \left(d(n) - \mathbf{w}^T(n)\mathbf{x}(n)\right)^2}$$

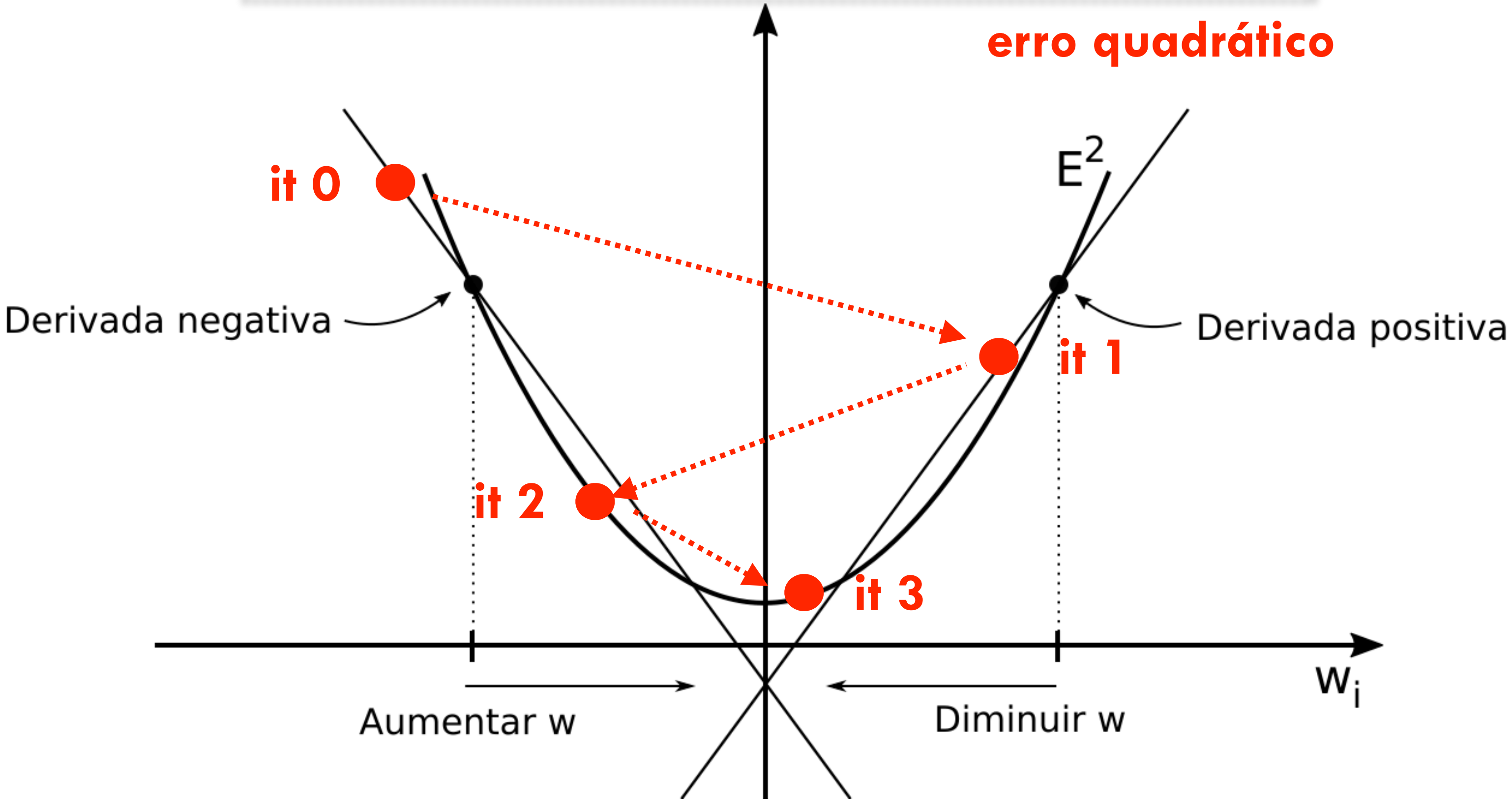
erro quadrático

1. $$w_i(n+1) = w_i(n) - \eta \frac{dE^2}{dw_i}$$

2. $$\frac{dE^2}{dw_i} = \frac{d\left(d(n) - y(n)\right)^2}{dw_i(n)} = 2 \times \left(d(n) - \mathbf{w}^T(n)\mathbf{x}(n)\right) \times -x_i$$

Gradiente descendente

$$E^2 = \left(d(n) - y(n)\right)^2 = \left(d(n) - \mathbf{w}^T(n)\mathbf{x}(n)\right)^2$$



Roteiro

1. Introdução
2. Perceptron
3. Teorema de Convergência
4. **Algoritmo de Treinamento Perceptron**
5. Exemplo / Exercício
6. Síntese / Próximas Aulas
7. Referências

Algoritmo de Treinamento

- Entradas e hiper-parâmetros:
    - **X(n)**: vetor de entrada
    - **W(n)**: vetor de pesos
    - **b** : bias
    - **y(n)**: saída obtida
    - **d(n)**: saída desejada (real)
    - **η**: taxa de aprendizado

- Funcionamento:
    - reduzir o erro entre as saídas esperadas, e as saídas obtidas

Algoritmo de Treinamento

- **Passo 1**. Obter o conjunto de amostras de treinamento
- **Passo 2**. Associar a saída desejada **d(n)** para cada amostra obtida **x(n)**
- **Passo 3**. Iniciar o vetor **w** com valores aleatórios pequenos
- **Passo 4**. Especificar a taxa de aprendizagem $\boldsymbol{\eta}$
- **Passo 5**. Iniciar o contador de número de épocas
  - (época $\leftarrow$ 0)

Algoritmo de Treinamento

- Passo 6. Repita
  - 6.1 erro ← FALSE
  - 6.2 Para todas as amostras de treinamento x(n), d(n), fazer:
    - 6.2.1 n ←epoca
    - 6.2.2 u ← $\mathbf{w}^T(n)\ \mathbf{x}(n)$
    - 6.2.3 y ← sgn (u)
    - 6.2.4 Se y(n) ≠ d(n), então
      - w ←w + $\boldsymbol{\eta}$ (d(n) - y(n)) x(n)
      - erro ← TRUE
  - 6.3 epoca ←epoca + 1
- até que erro ← FALSE

- // Repetir até que todas as amostras de treinamento tenham sido corretamente rotuladas

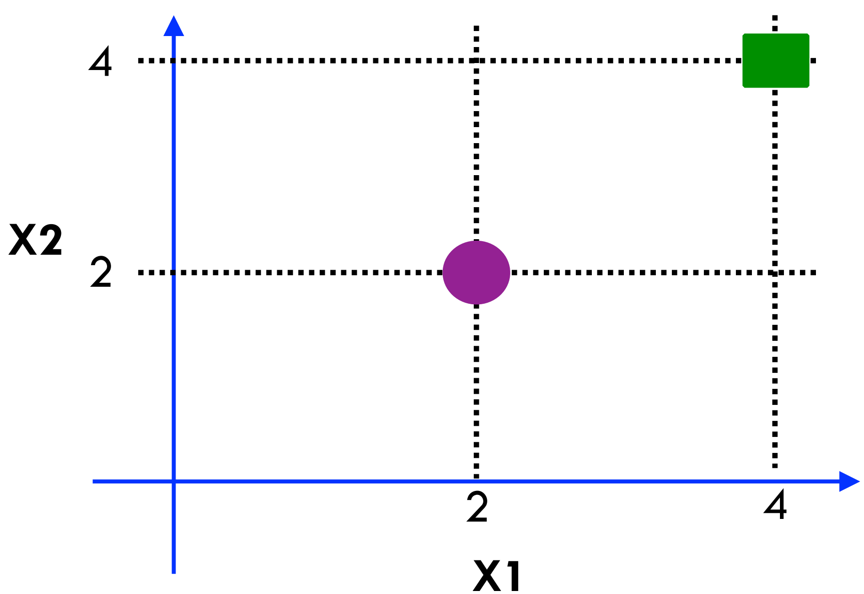
Roteiro

1. Introdução
2. Perceptron
3. Teorema de Convergência
4. Algoritmo de Treinamento Perceptron
5. Exemplo / Exercício
6. Síntese / Próximas Aulas
7. Referências

Exemplo

- Treinar o perceptron para o problema abaixo:
  - w0 = -0.5441, w1 = 0.5562, w2 = 0.4074
  - bias = -1
  - $\eta = 0.1$

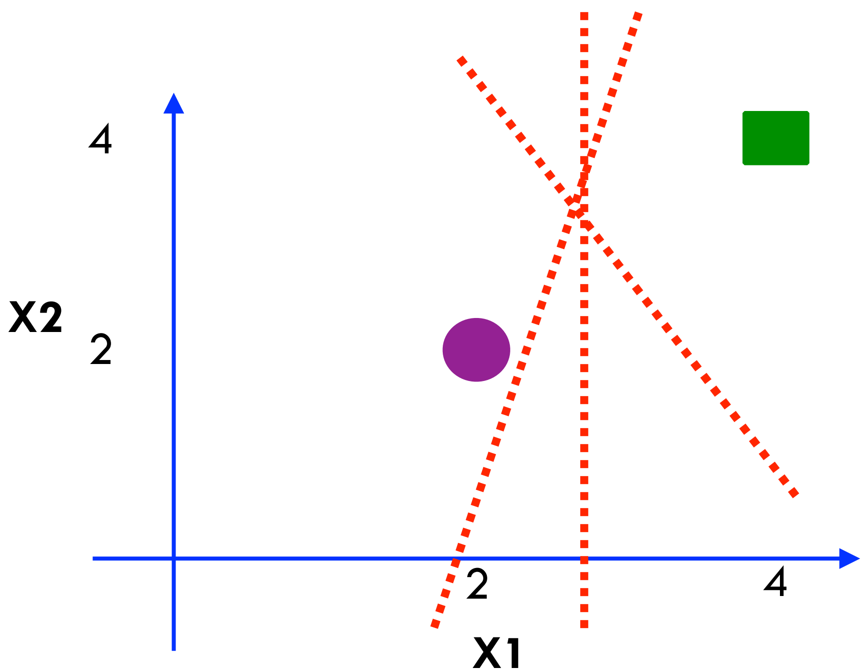
Exemplo	*X1*	*X2*	*Classe*
E1	***2***	***2***	***1***
E2	***4***	***4***	***0***



Exemplo

- Treinar o perceptron para o problema abaixo:
  - w0 = -0.5441, w1 = 0.5562, w2 = 0.4074
  - bias = -1
  - η = 0.1

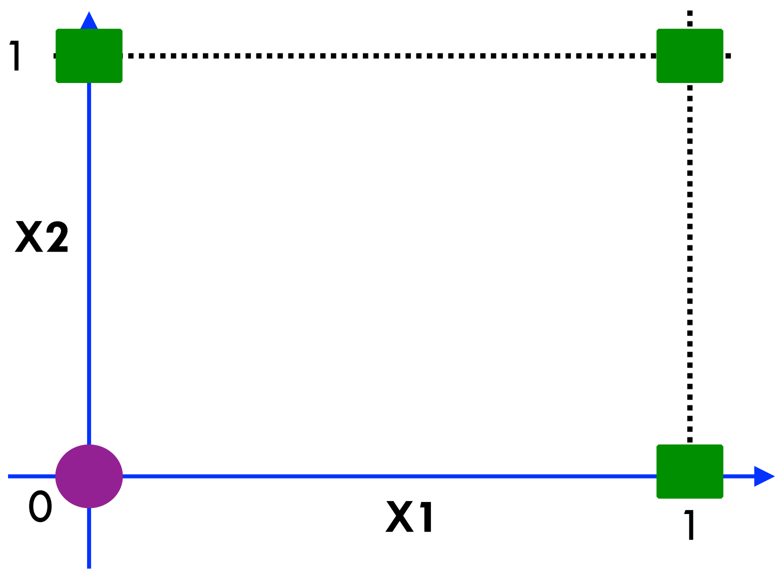
Exemplo	X1	X2	Classe
E1	2	2	1
E2	4	4	0



várias soluções possíveis

Exercício

- Treinar o perceptron para reconhecer o problema lógico OR. Dados:
  - w0 = w1 = w2 = 0.5
  - bias = +1
  - $\eta = 0.1$



X1	X2	D
0	0	0
0	1	1
1	0	1
1	1	1

Roteiro

Síntese/Revisão

- Perceptron
    - um neurônio de McCulloch Pitts
    - bias
    - função de ativação degrau
- Teorema de Convergência
- Algoritmo de Aprendizado do Perceptron
- Exemplo

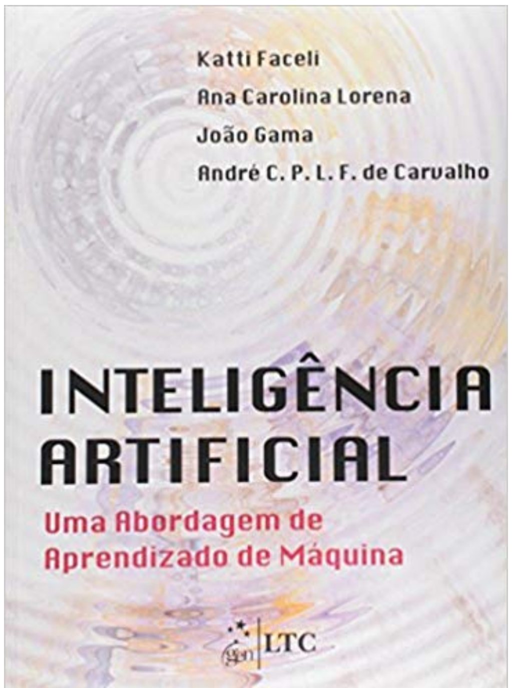
Próxima Aula

- ADALINE x Perceptron Simples
  - implementação dos algoritmos (R/Python)
  - prática: AT01

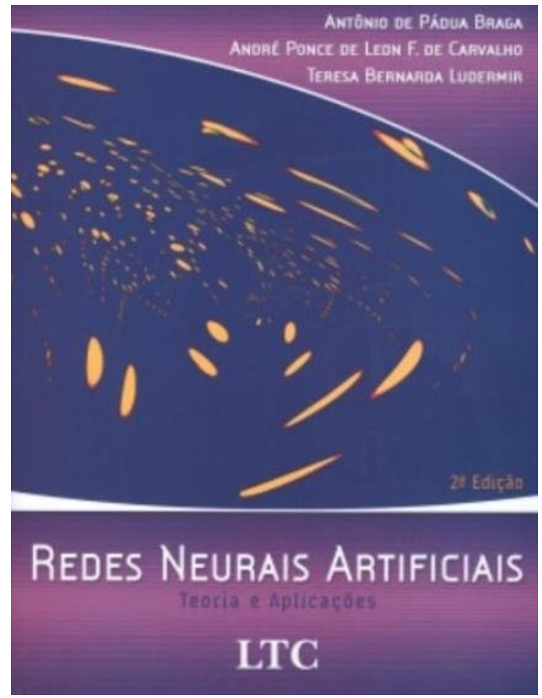
Roteiro

1. Introdução
2. Perceptron
3. Teorema de Convergência
4. Algoritmo de Treinamento Perceptron
5. Exemplo / Exercício
6. Síntese / Próximas Aulas
7. Referências

Literatura Sugerida

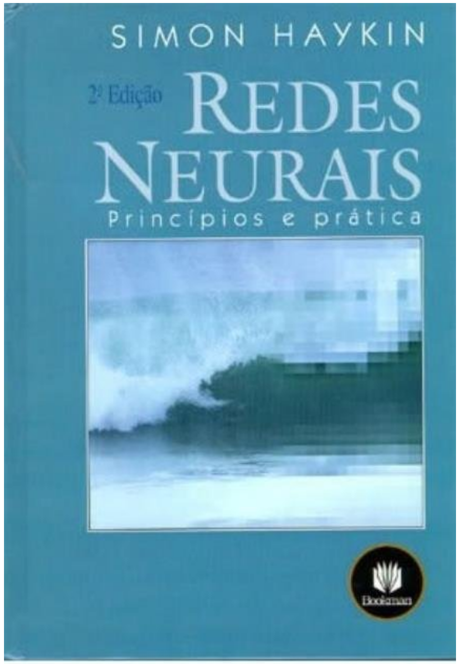


[Faceli et al, 2011]

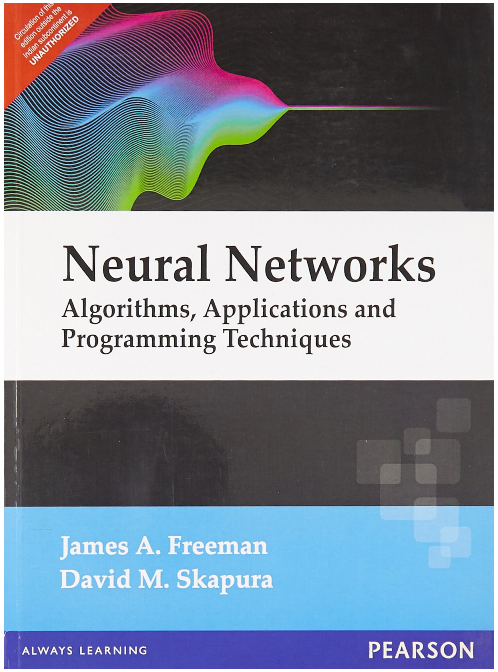


[Braga et al, 2007]

Literatura Sugerida



(Haykin, 1999)



(Freeman & Skapura, 1991)

Perguntas?

Prof. Rafael G. Mantovani

rgmantovani@gmail.com