UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE CIÊNCIAS MATEMÁTICAS E DE COMPUTAÇÃO
Departamento de Ciências de Computação

SCC-5809 - Capítulo 4 Perceptron de Camada Única

João Luís Garcia Rosa[1]

[1]SCC-ICMC-USP - joaoluis@icmc.usp.br

2012

Sumário

Sumário

História

- Nos primórdios das RNA (1943-1958), vários pesquisadores contribuíram:
  - McCulloch e Pitts (1943) introduziram a ideia de redes neurais como máquinas computacionais [3].
  - Hebb (1949) postulou a primeira regra para aprendizado auto-organizado [2].
  - Rosenblatt (1958) propôs o perceptron como o primeiro modelo para aprendizado com professor (supervisionado) [5].
- O perceptron é a forma mais simples de uma RNA usada para classificar padrões *linearmente separáveis*.
- Basicamente, consiste de um único neurônio com pesos sinápticos e bias ajustáveis.
- A prova de convergência do procedimento de aprendizado proposto por Rosenblatt é conhecida como teorema de convergência do perceptron.

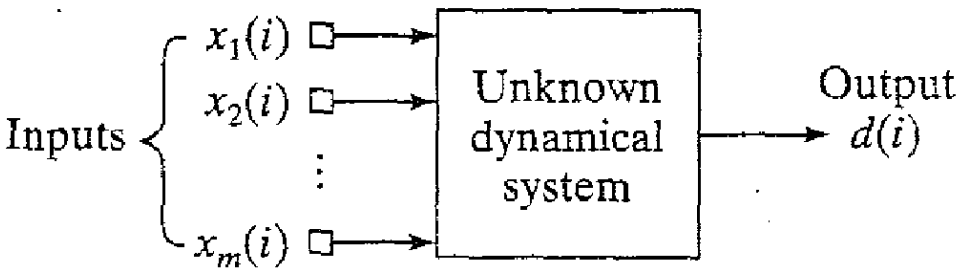
História

- O perceptron construído em torno de um único neurônio é limitado a realizar classificação de padrões com apenas duas classes (hipóteses).
- Expandindo a camada de saída do perceptron para incluir mais que um neurônio, pode-se classificar mais de duas classes.
- Entretanto, essas classes devem ser linearmente separáveis para o perceptron funcionar: superfície de decisão tem a forma de um hiperplano entre as duas classes.
- A regra de decisão é designar x à classe $\mathscr{C}_1$ se a saída for $y = +1$ e à classe $\mathscr{C}_2$ se a saída for -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

Filtro adaptativo

- Considere um sistema dinâmico, uma caracterização matemática do que é desconhecido.
- O que está disponível é um conjunto de dados de entrada-saída rotulados gerados pelo sistema em instantes discretos de tempo a alguma taxa uniforme.
- Quando um estímulo m-dimensional $\mathbf{x}(i)$ é aplicado nos m nós de entrada do sistema, o sistema responde produzindo uma saída escalar $d(i)$, onde $i = 1, 2, ..., n, ...$, como mostrado na figura abaixo [1]:



Filtro adaptativo

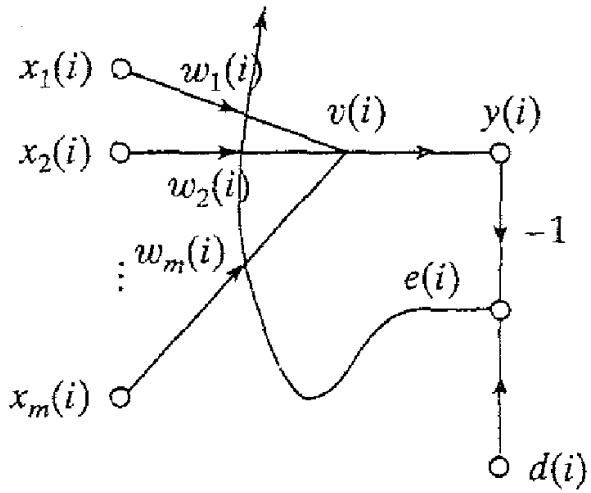
- O comportamento externo do sistema é descrito pelo conjunto de dados

$$\mathfrak{I} : \{\mathbf{x}(i), d(i); i = 1, 2, ..., n, ...\} \quad (1)$$

- onde

$$\mathbf{x}(i) = [x_1(i), x_2(i), ..., x_m(i)]^T \quad (2)$$

- A figura abaixo [1] mostra um grafo de fluxo de sinal do filtro adaptativo.



Filtro adaptativo

- A operação do filtro adaptativo consiste em
  1. *Processo de filtragem*, que envolve a computação de dois sinais:
     - Uma saída $y(i)$ que é produzida em resposta aos m elementos do vetor estímulo $\mathbf{x}(i)$, ou seja, $x_1(i), x_2(i), \ldots, x_m(i)$.
     - Um sinal de erro $e(i)$ obtido pela comparação da saída $y(i)$ com a saída correspondente $d(i)$ produzida pelo sistema desconhecido. $d(i)$ age como uma *resposta desejada* ou *sinal alvo*.
  2. *Processo adaptativo*, que envolve o ajuste automático dos pesos sinápticos do neurônio de acordo com o sinal de erro $e(i)$.

Filtro adaptativo

- Como o neurônio é linear, a saída $y(i)$ é exatamente a mesma do campo local induzido $v(i)$:

$$y(i) = v(i) = \sum_{k=1}^{m} w_k(i)x_k(i) \tag{3}$$

onde $w_1(i)$, $w_2(i)$, ..., $w_m(i)$ são os m pesos sinápticos do neurônio, medidos no tempo i.

- Na forma matricial pode-se expressar $y(i)$ como o produto interno dos vetores $\mathbf{x}(i)$ e $\mathbf{w}(i)$:

$$y(i) = \mathbf{x}^T(i)\mathbf{w}(i) \tag{4}$$

onde

$$\mathbf{w}(i) = [w_1(i), w_2(i), ..., w_m(i)]^T \tag{5}$$

Sumário

Algoritmo LMS

- O algoritmo LMS (*least-mean-square*) é baseado no uso de *valores instantâneos* para a função custo:

$$\xi(\mathbf{w}) = \frac{1}{2} e^2(n) \tag{6}$$

onde $e(n)$ é o sinal de erro medido no tempo n.

$$\frac{\partial \xi(\mathbf{w})}{\partial \mathbf{w}} = e(n) \frac{\partial e(n)}{\partial \mathbf{w}} \tag{7}$$

- Como o algoritmo LMS opera com um neurônio linear:

$$e(n) = d(n) - \mathbf{x}^T(n)\mathbf{w}(n) \tag{8}$$

portanto

$$\frac{\partial e(n)}{\partial \mathbf{w}(n)} = -\mathbf{x}(n) \tag{9}$$

Algoritmo LMS

- E

$$\frac{\partial \xi(\mathbf{w})}{\partial \mathbf{w}(n)} = -\mathbf{x}(n)e(n) \tag{10}$$

- Pode-se escrever

$$\hat{\mathbf{g}}(n) = -\mathbf{x}(n)e(n) \tag{11}$$

onde $\hat{\mathbf{g}}(n)$ é uma *estimativa* do vetor gradiente avaliado no ponto $\mathbf{w}(n)$

- Usando o vetor gradiente, pode-se formular o algoritmo LMS:

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + \eta \mathbf{x}(n)e(n) \tag{12}$$

Algoritmo LMS

- Sumário do algoritmo LMS
  - *Amostra de treinamento*:
    - Vetor do sinal de entrada: $\mathbf{x}(n)$
    - Resposta desejada: $d(n)$
  - *Parâmetro selecionado pelo usuário*: η
  - *Iniciação*: Faça $\hat{\mathbf{w}}(0) = 0$
  - *Computação*: Para $n = 1, 2, ...,$ compute

$$e(n) = d(n) - \hat{\mathbf{w}}^T(n)\mathbf{x}(n) \tag{13}$$

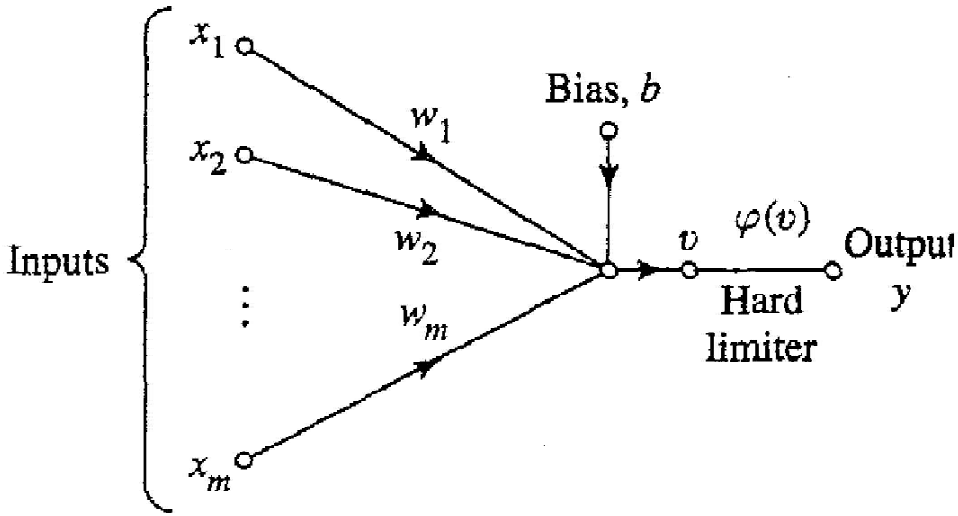
$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + \eta\mathbf{x}(n)e(n) \tag{14}$$

Sumário

Modelo de McCulloch-Pitts

- O perceptron é construído em volta de um neurônio não linear, o modelo de McCulloch-Pitts [1].

$$v = \sum_{i=1}^{m} w_i x_i + b \tag{15}$$



Classificador

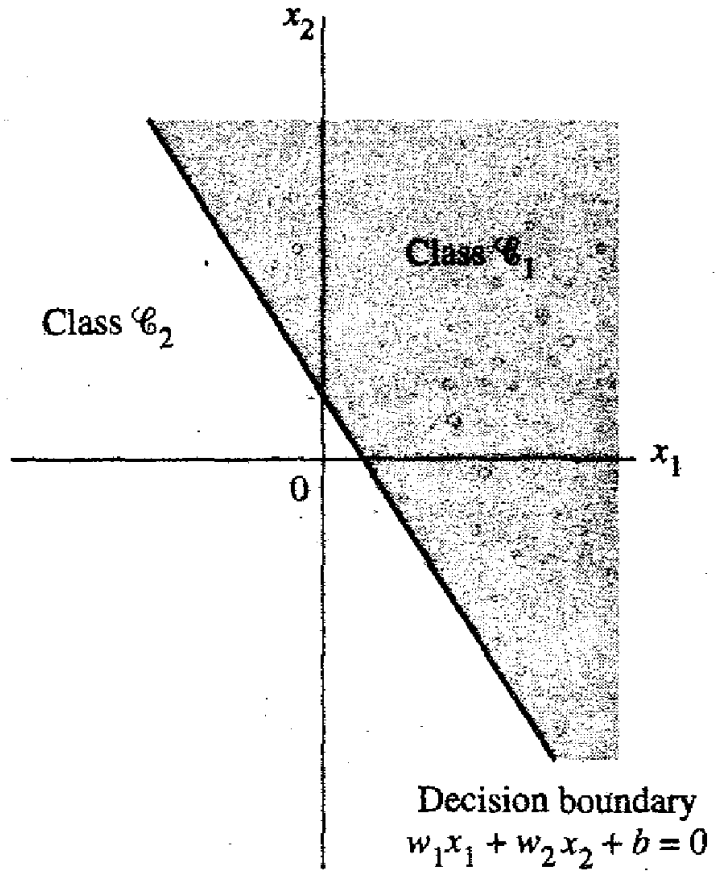
- O objetivo do perceptron é classificar o conjunto de estímulos x_1, x_2, ..., x_m aplicados externamente, à classe $\mathscr{C}_1$ se a saída for $y = +1$ ou à classe $\mathscr{C}_2$ se a saída for -1.
- Na forma mais simples do perceptron, há duas regiões de decisão separadas por um hiperplano definido por

$$\sum_{i=1}^{m} w_i x_i + b = 0 \tag{16}$$

- Veja figura 1.

Hiperplano

Figure 1 : Ilustração do hiperplano (nesse exemplo, uma linha reta) como fronteira de decisão para um problema de classificação de padrões de duas classes e duas dimensões [1].



Sumário

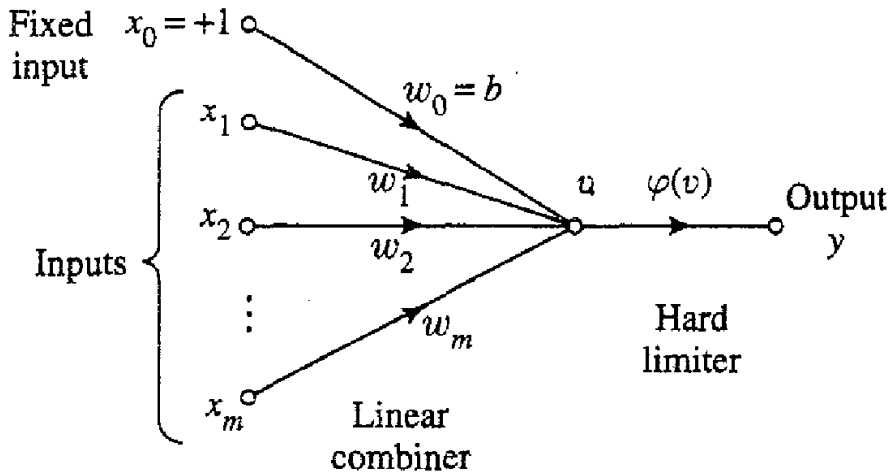
Correção de erros

- Para derivar o algoritmo de aprendizado por correção de erro para o perceptron, considere a figura abaixo [1].
- Vetor de entrada $(m+1)$ por 1:

$$\mathbf{x}(n) = [+1, x_1(n), x_2(n), ..., x_m(n)]^T \tag{17}$$

- Vetor de pesos $(m+1)$ por 1:

$$\mathbf{w}(n) = [b(n), w_1(n), w_2(n), ..., w_m(n)]^T \tag{18}$$

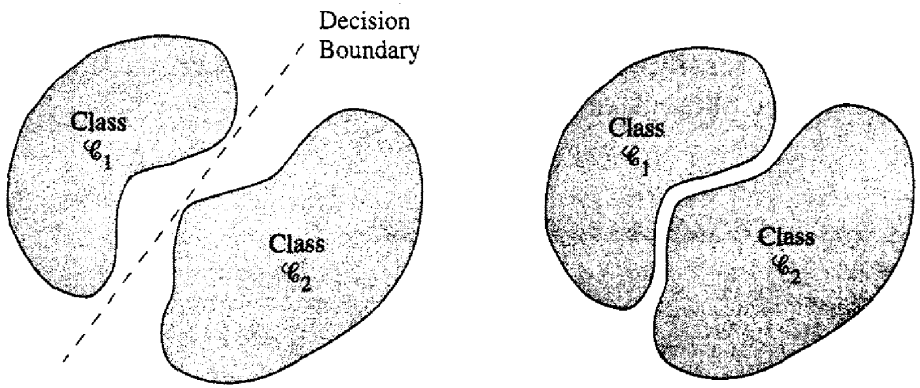


Combinador linear

- A saída do combinador linear é:

$$v(n) = \sum_{i=0}^{m} w_i(n)x_i(n) = \mathbf{w}^T(n)\mathbf{x}(n) \qquad (19)$$

  onde $w_0(n)$ representa o *bias* $b(n)$.
- Para n fixo, a equação $\mathbf{w}^T\mathbf{x} = 0$, plotada num espaço m-dimensional define um hiperplano, como a superfície de decisão entre duas classes diferentes de entradas.
- Para o perceptron funcionar adequadamente, as duas classes $\mathscr{C}_1$ e $\mathscr{C}_2$ devem ser *linearmente separáveis* [1]:



Conjuntos de treinamento

- Seja $\mathscr{X}_1 = \mathbf{x}_1(1), \mathbf{x}_1(2), ...$ o subconjunto de vetores de treinamento que pertencem à classe $\mathscr{C}_1$ e $\mathscr{X}_2 = \mathbf{x}_2(1), \mathbf{x}_2(2), ...$ o subconjunto de vetores de treinamento que pertencem à classe $\mathscr{C}_2$.
- $\mathscr{X} = \mathscr{X}_1 \cup \mathscr{X}_2$.
- Dados os conjuntos $\mathscr{X}_1$ e $\mathscr{X}_2$ para treinar o classificador, o treinamento envolve o ajuste do vetor de pesos $\mathbf{w}$ tal que $\mathscr{C}_1$ e $\mathscr{C}_2$ sejam linearmente separáveis:
  - $\mathbf{w}^T\mathbf{x} > 0$ para todo vetor de entrada $\mathbf{x}$ pertencente à classe $\mathscr{C}_1$.
  - $\mathbf{w}^T\mathbf{x} \leq 0$ para todo vetor de entrada $\mathbf{x}$ pertencente à classe $\mathscr{C}_2$.
- O problema do treinamento é, dados $\mathscr{X}_1$ e $\mathscr{X}_2$, achar um vetor de pesos $\mathbf{w}$ tal que as desigualdades acima sejam satisfeitas.

Treinamento

- O algoritmo pode ser formulado assim:
  1. Se o n-ésimo termo do conjunto de treinamento $\mathbf{x}(n)$ é corretamente classificado por $\mathbf{w}(n)$ computado na n-ésima iteração do algoritmo, nenhuma correção é feita ao vetor de pesos de acordo com a regra
     1. $\mathbf{w}(n+1) = \mathbf{w}(n)$, se $\mathbf{w}^T\mathbf{x}(n) > 0$ e $\mathbf{x}(n)$ pertence à classe $\mathscr{C}_1$.
     2. $\mathbf{w}(n+1) = \mathbf{w}(n)$, se $\mathbf{w}^T\mathbf{x}(n) \leq 0$ e $\mathbf{x}(n)$ pertence à classe $\mathscr{C}_2$.
  2. Caso contrário, o vetor de pesos é atualizado de acordo com a regra
     1. $\mathbf{w}(n+1) = \mathbf{w}(n) - \eta(n)\mathbf{x}(n)$, se $\mathbf{w}^T(n)\mathbf{x}(n) > 0$ e $\mathbf{x}(n)$ pertence à classe $\mathscr{C}_2$.
     2. $\mathbf{w}(n+1) = \mathbf{w}(n) + \eta(n)\mathbf{x}(n)$, se $\mathbf{w}^T(n)\mathbf{x}(n) \leq 0$ e $\mathbf{x}(n)$ pertence à classe $\mathscr{C}_1$.

  onde $\eta(n)$ controla o ajuste aplicado ao vetor de pesos na iteração n.

Convergência

- Se $\eta(n) = \eta > 0$, onde η é independente de n, tem-se uma *regra de adaptação de incremento fixo* para o perceptron.
- Prova-se a convergência para $\eta = 1$.
- Sejam $\mathbf{w}(0) = 0$, $\mathbf{w}^T(n)\mathbf{x}(n) < 0$ e $\mathbf{x}(n) \in \mathscr{X}_1$.
- Ou seja, o perceptron classifica incorretamente os vetores $\mathbf{x}(1)$, $\mathbf{x}(2)$, ..., já que a condição "$\mathbf{w}^T\mathbf{x} \leq 0$ para todo vetor de entrada $\mathbf{x}$ pertencente à classe $\mathscr{C}_2$" é violada.
- Como $\eta(n) = 1$, pode-se escrever

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mathbf{x}(n), \text{ } \textit{para } \mathbf{x}(n) \textit{ pertencente a classe } \mathscr{C}_1 \tag{20}$$

- Dado que $\mathbf{w}(0) = 0$, pode-se iterativamente resolver esta equação para $\mathbf{w}(n+1)$, obtendo o resultado

$$\mathbf{w}(n+1) = \mathbf{x}(1) + \mathbf{x}(2) + ... + \mathbf{x}(n) \tag{21}$$

Convergência

- Como assume-se que $\mathscr{C}_1$ e $\mathscr{C}_2$ são linearmente separáveis, existe uma solução $\mathbf{w}_0$ para o qual $\mathbf{w}^T\mathbf{x}(n) > 0$ para os vetores $\mathbf{x}(1), ..., \mathbf{x}(n)$ pertencentes a $\mathscr{X}_1$.
- Para uma solução fixa $\mathbf{w}_0$, define-se

$$\alpha = min_{\mathbf{x}(n)\in\mathscr{X}_1}\mathbf{w}_0^T\mathbf{x}(n) \tag{22}$$

- Multiplicando ambos os lados da equação 21 pelo vetor linha $\mathbf{w}_0^T$ tem-se

$$\mathbf{w}_0^T\mathbf{w}(n+1) = \mathbf{w}_0^T\mathbf{x}(1) + \mathbf{w}_0^T\mathbf{x}(2) + ... + \mathbf{w}_0^T\mathbf{x}(n) \tag{23}$$

- Retomando a equação 22

$$\mathbf{w}_0^T\mathbf{w}(n+1) \geq n\alpha \tag{24}$$

Convergência

- Dados dois vetores $\mathbf{w}_0$ e $\mathbf{w}(n+1)$, a *desigualdade de Cauchy-Schwarz* estabelece que

$$\| \mathbf{w}_0 \|^2 \| \mathbf{w}(n+1) \|^2 \geq [\mathbf{w}_0^T \mathbf{w}(n+1)]^2 \tag{25}$$

  onde $\| . \|$ corresponde à norma Euclidiana do vetor argumento e o produto interno $\mathbf{w}_0^T \mathbf{w}(n+1)$ é escalar.
- Da equação 24, $[\mathbf{w}_0^T \mathbf{w}(n+1)]^2$ é maior ou igual a $n^2\alpha^2$.
- Da equação 25, $\| \mathbf{w}_0 \|^2 \| \mathbf{w}(n+1) \|^2$ é maior ou igual a $[\mathbf{w}_0^T \mathbf{w}(n+1)]^2$.
- Segue que

$$\| \mathbf{w}_0 \|^2 \| \mathbf{w}(n+1) \|^2 \geq n^2\alpha^2 \tag{26}$$

  ou equivalentemente

$$\| \mathbf{w}(n+1) \|^2 \geq \frac{n^2\alpha^2}{\| \mathbf{w}_0 \|^2} \tag{27}$$

Convergência

- A equação 20 é re-escrita da seguinte forma

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mathbf{x}(k), \text{ para } k = 1, ..., n \text{ e } \mathbf{x}(k) \in \mathscr{X}_1 \quad (28)$$

- Pegando a norma Euclidiana em ambos os lados da equação 28, obtém-se

$$\| \mathbf{w}(k+1) \|^2 = \| \mathbf{w}(k) \|^2 + \| \mathbf{x}(k) \|^2 + 2\mathbf{w}^T(k)\mathbf{w}(k) \quad (29)$$

- Partindo da assunção de que o perceptron classifica incorretamente um vetor de entrada $\mathbf{x}(k)$ pertencente ao subconjunto $\mathscr{X}_1$, tem-se que $\mathbf{w}^T(k)\mathbf{x}(k) < 0$.
- Deduz-se de 29 que

$$\| \mathbf{w}(k+1) \|^2 \leq \| \mathbf{w}(k) \|^2 + \| \mathbf{x}(k) \|^2 \quad (30)$$

ou equivalentemente

$$\| \mathbf{w}(k+1) \|^2 - \| \mathbf{w}(k) \|^2 \leq \| \mathbf{x}(k) \|^2, \ k = 1, ...n \quad (31)$$

Convergência

- Adicionando estas desigualdades para $k = 1, ..., n$ e invocando a condição inicial $\mathbf{w}(0) = 0$, tem-se

$$\| \mathbf{w}(k+1) \|^2 = \sum_{k=1}^{n} \| \mathbf{x}(k) \|^2 \leq n\beta \tag{32}$$

onde β é um número positivo

$$\beta = max_{\mathbf{x}(k) \in \mathscr{X}_1} \| \mathbf{x}(k) \|^2 \tag{33}$$

- A equação 32 estabelece que a norma Euclidiana ao quadrado do vetor de pesos $\mathbf{w}(n+1)$ cresce no máximo linearmente com o número de iterações n.
- Esta equação está em conflito com a equação 27 para valores grandes de n.

Convergência

- Pode-se estabelecer que n não possa ser maior que um valor n_{max} para o qual as equações 27 e 32 sejam satisfeitas com o sinal de igualdade.
- Isto é, n_{max} é a solução da equação

$$\frac{n_{max}^2 \alpha^2}{\| \mathbf{w}_0 \|^2} = n_{max} \beta \tag{34}$$

- Resolvendo para n_{max}, dada um vetor solução $\mathbf{w}_0$:

$$n_{max} = \frac{\beta \| \mathbf{w}_0 \|^2}{\alpha^2} \tag{35}$$

- Provou-se então que para $\eta(n) = 1$ para todo n, e $\mathbf{w}(0) = 0$ e dado que um vetor solução $\mathbf{w}_0$ existe, a regra para adaptar os pesos sinápticos deve terminar depois de no máximo n_{max} iterações.
- Note que, das equações 22, 33 e 35, não há uma solução única para $\mathbf{w}_0$ ou n_{max}.

Teorema da Convergência do Perceptron

- O *teorema da convergência de incremento fixo* para o perceptron pode ser enunciada:

  Sejam os subconjuntos dos vetores de treinamento $\mathscr{X}_1$ e $\mathscr{X}_2$ linearmente separáveis. Sejam as entradas apresentadas ao perceptron originárias desses dois subconjuntos. O perceptron converge depois de n_0 iterações, no sentido de que

  $$\mathbf{w}(n_0) = \mathbf{w}(n_0 + 1) = \mathbf{w}(n_0 + 2) = \ldots \qquad (36)$$

  seja um vetor solução para $n_0 \leq n_{max}$.

Teorema da Convergência do Perceptron

- Considere agora o *procedimento para correção de erro absoluto*, para o qual $\eta(n)$ é variável.
- Seja $\eta(n)$ o menor inteiro para o qual $\eta(n)\mathbf{x}^T\mathbf{x}(n) > |\mathbf{w}^T(n)\mathbf{x}(n)|$.
- Ou seja, se o produto interno $\mathbf{w}^T(n)\mathbf{x}(n)$ na iteração n tem um sinal incorreto, então $\mathbf{w}^T(n+1)\mathbf{x}(n)$ na iteração $n+1$ teria o sinal correto.
- Em outras palavras, cada padrão é apresentado repetidamente ao perceptron até que o padrão seja classificado corretamente.

Teorema da Convergência do Perceptron

- Sumário do Algoritmo de Convergência do Perceptron:
  - Variáveis e parâmetros:
    - Vetor de entrada: $\mathbf{x}(n) = [+1, x_1(n), x_2(n), ..., x_m(n)]^T$
    - Vetor de pesos: $\mathbf{w}(n) = [b(n), w_1(n), w_2(n), ..., w_m(n)]^T$
    - Bias: $b(n)$
    - Resposta real: $y(n)$
    - Resposta desejada: $d(n)$:

$$d(n) = \begin{cases} +1, & \text{se } \mathbf{x}(n) \text{ pertence a classe } \mathscr{C}_1 \\ -1, & \text{se } \mathbf{x}(n) \text{ pertence a classe } \mathscr{C}_2 \end{cases}$$

    - Taxa de aprendizado: η
    - Função *signum*:

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

Teorema da Convergência do Perceptron

- Sumário do Algoritmo de Convergência do Perceptron:
  1. *Iniciação*: Faça $\mathbf{w}(0) = 0$. Então realize as seguintes computações para o passo de tempo $n = 1, 2, ...$
  2. *Ativação*: No passo de tempo n, ative o perceptron aplicando os vetores de entradas contínuas $\mathbf{x}(n)$ e respostas desejadas $d(n)$
  3. *Computação da resposta real*: $y(n) = sgn[\mathbf{w}^T(n)\mathbf{x}(n)]$
  4. *Adaptação do vetor de pesos*: $\mathbf{w}(n+1) = \mathbf{w}(n) + \eta[d(n) - y(n)]\mathbf{x}(n)$
  5. *Continuação*: Incremente o passo de tempo n e vá ao passo 2.

Regra Delta

- Para um perceptron

$$w_i(n+1) = w_i(n) + \eta(d(n) - y(n)).x_i(n) \qquad (37)$$

onde

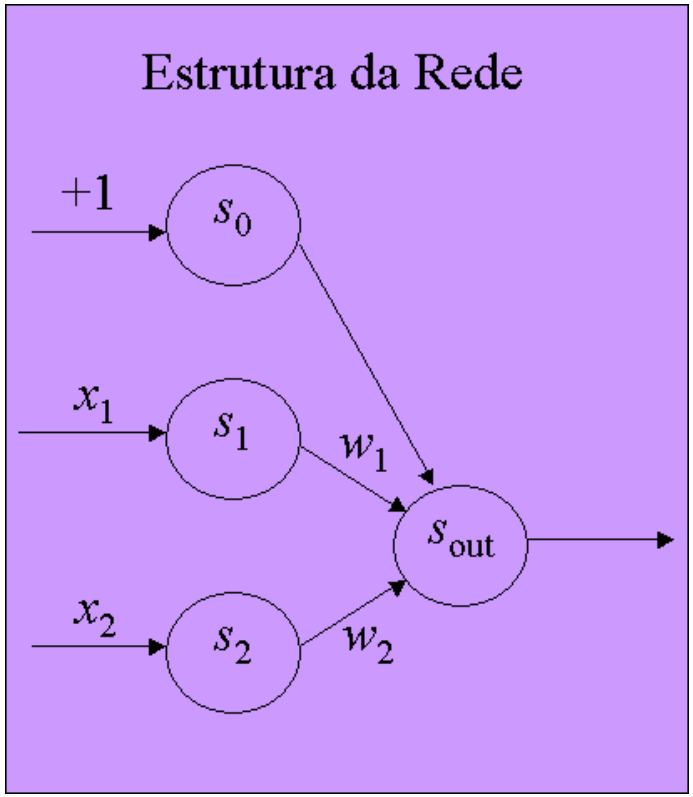
  - η = taxa (ou coeficiente) de aprendizado
  - $d(n)$ = saída desejada
  - $y(n)$ = saída real
  - $\delta = d(n) - y(n)$ = erro
  - $x_i(n)$ = entrada i, $1 \leq i \leq m$

- Regra Delta:

$$\Delta w_i = w_i(n+1) - w_i(n) = \eta \delta x_i \qquad (38)$$

Exemplo [4]

- Simulação do operador lógico AND:



Exemplo [4]

- Pesos iniciais: $w_0 = 0$, $w_1 = 0$, $w_2 = 0$.
- Taxa de aprendizado: $\eta = 0.5$.

AND	x_0	x_1	x_2	t
Entrada 1:	1	0	0	0
Entrada 2:	1	0	1	0
Entrada 3:	1	1	0	0
Entrada 4:	1	1	1	1

Exemplo [4]

- 1^o Ciclo
  - Entrada 1: $s_{out} = f(w_0 x_0 + w_1 x_1 + w_2 x_2) = f(0 \times 1 + 0 \times 0 + 0 \times 0) = f(0) = 0 \longrightarrow s_{out} = t$
  - Entrada 2: $s_{out} = f(w_0 x_0 + w_1 x_1 + w_2 x_2) = f(0 \times 1 + 0 \times 1 + 0 \times 0) = f(0) = 0 \longrightarrow s_{out} = t$
  - Entrada 3: $s_{out} = f(w_0 x_0 + w_1 x_1 + w_2 x_2) = f(0 \times 1 + 0 \times 0 + 0 \times 1) = f(0) = 0 \longrightarrow s_{out} = t$
  - Entrada 4: $s_{out} = f(w_0 x_0 + w_1 x_1 + w_2 x_2) = f(0 \times 1 + 0 \times 1 + 0 \times 1) = f(0) = 0 \longrightarrow s_{out} \neq t$
  - Pesos:
    - $w_0 = w_0 + (t - s_{out})x_0 = 0 + 0.5 \times (1 - 0) \times 1 = 0.5$
    - $w_1 = w_1 + (t - s_{out})x_1 = 0 + 0.5 \times (1 - 0) \times 1 = 0.5$
    - $w_2 = w_2 + (t - s_{out})x_2 = 0 + 0.5 \times (1 - 0) \times 1 = 0.5$

Exemplo [4]

- 2^o Ciclo
  - Entrada 1: $s_{out} = f(w_0 x_0 + w_1 x_1 + w_2 x_2) = f(0.5 \times 1 + 0.5 \times 0 + 0.5 \times 0) = f(0.5) = 1 \longrightarrow s_{out} \neq t$
  - Pesos:
    - $w_0 = w_0 + (t - s_{out})x_0 = 0.5 + 0.5 \times (0 - 1) \times 1 = 0$
    - $w_1 = w_1 + (t - s_{out})x_1 = 0.5 + 0.5 \times (0 - 1) \times 0 = 0.5$
    - $w_2 = w_2 + (t - s_{out})x_2 = 0.5 + 0.5 \times (0 - 1) \times 0 = 0.5$
  - Entrada 2: $s_{out} = f(w_0 x_0 + w_1 x_1 + w_2 x_2) = f(0 \times 1 + 0.5 \times 0 + 0.5 \times 1) = f(0.5) = 1 \longrightarrow s_{out} \neq t$
  - Pesos:
    - $w_0 = w_0 + (t - s_{out})x_0 = 0 + 0.5 \times (0 - 1) \times 1 = -0.5$
    - $w_1 = w_1 + (t - s_{out})x_1 = 0.5 + 0.5 \times (0 - 1) \times 0 = 0.5$
    - $w_2 = w_2 + (t - s_{out})x_2 = 0.5 + 0.5 \times (0 - 1) \times 1 = 0$

Exemplo [4]

- 2^o Ciclo (cont.)
  - Entrada 3: $s_{out} = f(w_0 x_0 + w_1 x_1 + w_2 x_2) = f(-0.5 \times 1 + 0.5 \times 1 + 0 \times 0) = f(0) = 0 \longrightarrow s_{out} = t$
  - Entrada 4: $s_{out} = f(w_0 x_0 + w_1 x_1 + w_2 x_2) = f(-0.5 \times 1 + 0.5 \times 1 + 0 \times 1) = f(0) = 0 \longrightarrow s_{out} \neq t$
  - Pesos:
    - $w_0 = w_0 + (t - s_{out})x_0 = -0.5 + 0.5 \times (1 - 0) \times 1 = 0$
    - $w_1 = w_1 + (t - s_{out})x_1 = 0.5 + 0.5 \times (1 - 0) \times 1 = 1$
    - $w_2 = w_2 + (t - s_{out})x_2 = 0 + 0.5 \times (1 - 0) \times 1 = 0.5$

Exemplo [4]

- 3^o Ciclo
  - Entrada 1: $s_{out} = f(w_0 x_0 + w_1 x_1 + w_2 x_2) = f(0 \times 1 + 1 \times 0 + 0.5 \times 0) = f(0) = 0 \longrightarrow s_{out} = t$
  - Entrada 2: $s_{out} = f(w_0 x_0 + w_1 x_1 + w_2 x_2) = f(0 \times 1 + 1 \times 0 + 0.5 \times 1) = f(0.5) = 1 \longrightarrow s_{out} \neq t$
  - Pesos:
    - $w_0 = w_0 + (t - s_{out})x_0 = -0.5 + 0.5 \times (0 - 1) \times 1 = -1$
    - $w_1 = w_1 + (t - s_{out})x_1 = 1 + 0.5 \times (0 - 1) \times 0 = 1$
    - $w_2 = w_2 + (t - s_{out})x_2 = 0.5 + 0.5 \times (0 - 1) \times 1 = 0$
  - Entrada 3: $s_{out} = f(w_0 x_0 + w_1 x_1 + w_2 x_2) = f(-1 \times 1 + 1 \times 1 + 0 \times 0) = f(0) = 0 \longrightarrow s_{out} = t$
  - Entrada 4: $s_{out} = f(w_0 x_0 + w_1 x_1 + w_2 x_2) = f(-1 \times 1 + 1 \times 1 + 0 \times 1) = f(0) = 0 \longrightarrow s_{out} \neq t$
  - Pesos:
    - $w_0 = w_0 + (t - s_{out})x_0 = -1 + 0.5 \times (1 - 0) \times 1 = -0.5$
    - $w_1 = w_1 + (t - s_{out})x_1 = 1 + 0.5 \times (1 - 0) \times 1 = 1.5$
    - $w_2 = w_2 + (t - s_{out})x_2 = 0 + 0.5 \times (1 - 0) \times 1 = 0.5$

Exemplo [4]

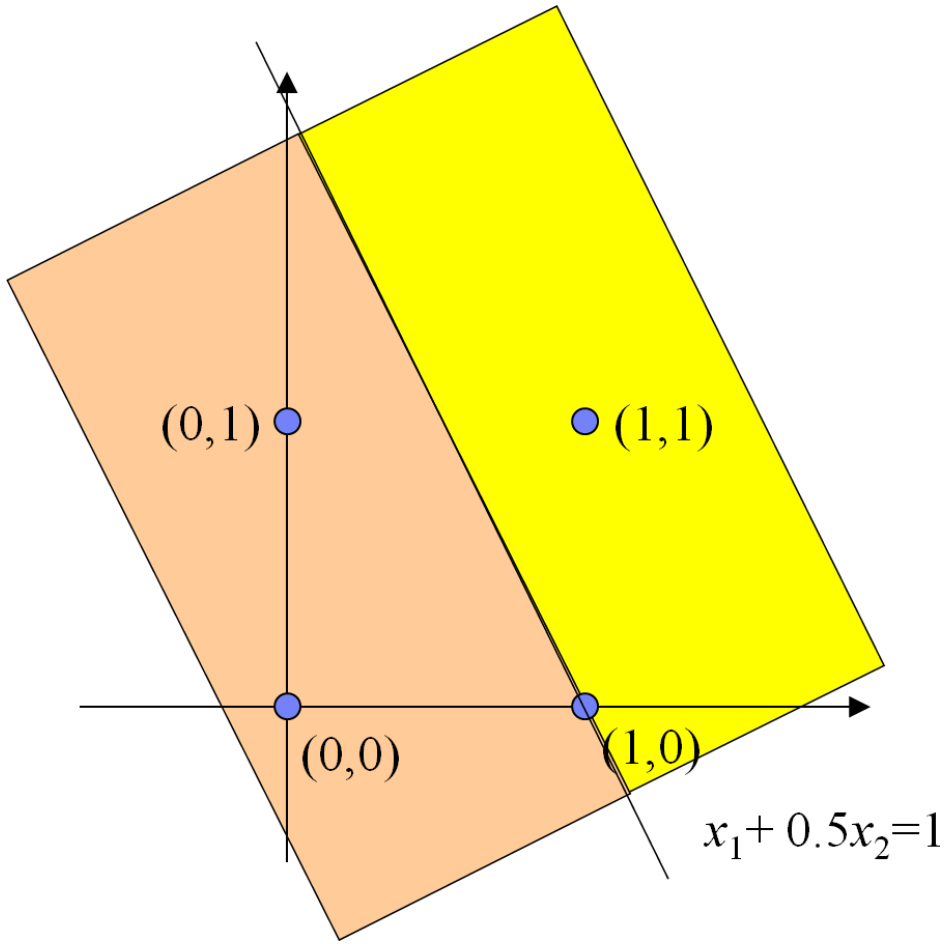
- 4^o Ciclo
  - Entrada 1: $s_{out} = f(w_0 x_0 + w_1 x_1 + w_2 x_2) =$
    $f(-0.5 \times 1 + 1.5 \times 0 + 0.5 \times 0) = f(-0.5) = 0 \longrightarrow s_{out} = t$
  - Entrada 2: $s_{out} = f(w_0 x_0 + w_1 x_1 + w_2 x_2) =$
    $f(-0.5 \times 1 + 1.5 \times 0 + 0.5 \times 1) = f(0) = 0 \longrightarrow s_{out} = t$
  - Entrada 3: $s_{out} = f(w_0 x_0 + w_1 x_1 + w_2 x_2) =$
    $f(-0.5 \times 1 + 1.5 \times 1 + 0.5 \times 0) = f(1) = 1 \longrightarrow s_{out} \neq t$
  - Pesos:
    - $w_0 = w_0 + (t - s_{out})x_0 = -0.5 + 0.5 \times (0 - 1) \times 1 = -1$
    - $w_1 = w_1 + (t - s_{out})x_1 = 1.5 + 0.5 \times (0 - 1) \times 1 = 1$
    - $w_2 = w_2 + (t - s_{out})x_2 = 0.5 + 0.5 \times (0 - 1) \times 0 = 0.5$
  - Entrada 4: $s_{out} = f(w_0 x_0 + w_1 x_1 + w_2 x_2) =$
    $f(-1 \times 1 + 1 \times 1 + 0.5 \times 1) = f(0.5) = 1 \longrightarrow s_{out} = t$

Exemplo [4]

- 5^o Ciclo
  - Entrada 1: $s_{out} = f(w_0 x_0 + w_1 x_1 + w_2 x_2) = f(-1 \times 1 + 1 \times 0 + 0.5 \times 0) = f(-1) = 0 \longrightarrow s_{out} = t$
  - Entrada 2: $s_{out} = f(w_0 x_0 + w_1 x_1 + w_2 x_2) = f(-1 \times 1 + 1 \times 0 + 0.5 \times 1) = f(-0.5) = 0 \longrightarrow s_{out} = t$
  - Entrada 3: $s_{out} = f(w_0 x_0 + w_1 x_1 + w_2 x_2) = f(-1 \times 1 + 1 \times 1 + 0.5 \times 0) = f(0) = 0 \longrightarrow s_{out} = t$
  - Entrada 4: $s_{out} = f(w_0 x_0 + w_1 x_1 + w_2 x_2) = f(-1 \times 1 + 1 \times 1 + 0.5 \times 1) = f(0.5) = 1 \longrightarrow s_{out} = t$
  - Pesos Finais (**solução**):
    - $w_0 = -1$
    - $w_1 = 1$
    - $w_2 = 0.5$

Exemplo [4]: Interpretação geométrica

- *Linha de Decisão*: $x_1 w_1 + x_2 w_2 = -\theta \Longrightarrow x_1 + 0.5 x_2 = 1.$



Bibliografia I

[1] S. Haykin
Neural networks - a comprehensive foundation.
2nd. edition. Prentice Hall, 1999.

[2] D. O. Hebb
The Organization of Behavior: A Neuropsychological Theory.
Wiley, 1949.

[3] W. S. McCulloch and W. Pitts
A logical calculus of the ideas immanent in nervous activity
Bulletin of Mathematical Biophysics, 5, pp. 115-133, 1943.

Bibliografia II

[4] R. A. F. Romero
SCC-5809 Redes Neurais.
Slides e listas de exercícios. Programa de Pós-Graduação em Ciência de Computação e Matemática Computacional. ICMC/USP, 2010.

[5] F. Rosenblatt
The Perceptron: A probabilistic model for information storage and organization in the brain.
Psychological Review, vol. 65, pp. 386–408, 1958.