

Inteligência Artificial

Aula 28- Redes Neurais (Simulação Perceptron) ¹

Sílvia M.W. Moraes

Faculdade de Informática - PUCRS

June 13, 2017

¹Este material não pode ser reproduzido ou utilizado de forma parcial sem a permissão dos autores.

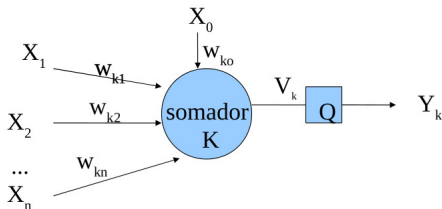
Sinopse

- Nesta aula, mostramos a simulação de uma **rede neural Perceptron**.
- Este material foi construído usando os algoritmos vistos em aula e disponíveis no moodle.

Sumário

1 Introdução à Redes Neurais Perceptron

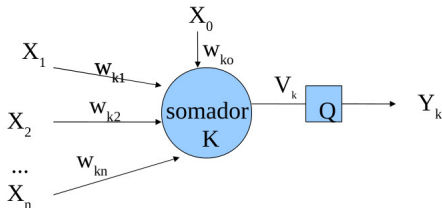
Redes Neurais: Neurônio Artificial



- Entradas: x_1, \dots, x_n : valores do domínio; x_0 : entrada extra, é sempre 1 (bias).
- Pesos sinápticos: armazenam o conhecimento adquirido pela rede.
 - w_{kn} , onde k é o neurônio e o n , a entrada; w_{k0} : peso do bias
- v_k : campo local induzido, corresponde a uma soma ponderada (pelos pesos) das entradas do neurônio

$$v_k = \sum_{i=0}^n w_{ki} \times x_i$$

Redes Neurais: Neurônio Artificial



- y_k : saída do neurônio k é definido pela função de transferência, a qual simula a polaridade da membrana pós-sináptica.
 - $y_k = Q(v_k)$, onde a função de transferência pode ser:
 limiar: $Q(v_k) = \begin{cases} 1 & \text{se } v_k \geq 0 \\ 0 & \text{caso contrário} \end{cases}$

Redes Neurais: Treinamento

- Os ciclos de treinamento de uma rede são medidos em épocas.
- Uma época corresponde a passagem de todos os padrões do conjunto de treino uma vez pela rede.
- Para treinar uma rede são necessárias várias épocas.

Redes Neurais: Treinamento

- **Rede Perceptron (Adaline):**
 - **Algoritmo de Treinamento: Regra Delta**
 - Sendo $X = \{(amostra_1, d_1), (amostra_2, d_2), \dots\}$ o conjunto de treino e η , a taxa de aprendizagem (deve ser positiva).

Inicializa os pesos w da rede com zero

Repetir até encontrar erro zero para todas as amostras{

epocas = epocas + 1

Para cada par de X {

Para cada atributo x_i da amostra, onde $i = 1$ a n {

Para cada neurônio k da rede{

$$v_k = w_{ki} * x_i$$

$$y_k = Q(v_k)$$

$$\text{erro}_k = d_k - y_k$$

$$\Delta w_{ki} = \eta * \text{erro}_k * x_i$$

$$w_{ki} = w_{ki} + \Delta w_{ki}$$

}

}

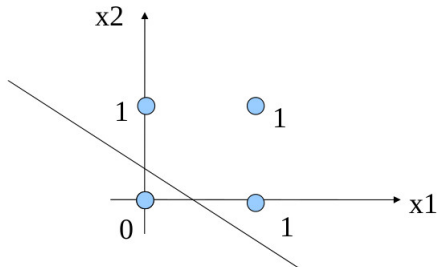
}

}

Redes Neurais: Treinamento

- **Rede Perceptron (Adaline):**
 - Exemplo: OR

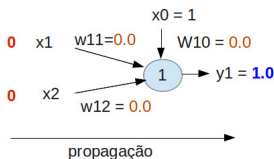
x1	x2	d
0	0	0
0	1	1
1	0	1
1	1	1



Redes Neurais: Treinamento (Época 1)

- **Pesos** inicializados com zero: $w_{10} = 0, w_{11} = 0, w_{12} = 0$
- limiar: $Q(v_k) = \begin{cases} 1 & \text{se } v_k \geq 0 \\ 0 & \text{caso contrário} \end{cases}$, taxa de aprendizagem: $\eta = 1$

x1	x2	d
0	0	0
0	1	1
1	0	1
1	1	1



- $v_1 = x_0 \times w_{10} + x_1 \times w_{11} + x_2 \times w_{12}$
- $v_1 = 1 \times 0 + 0 \times 0 + 0 \times 0 = 0$
- $y_1 = Q(v_1) = Q(0) = 1.0$

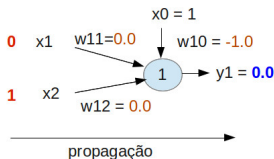
- Ajuste dos pesos (Aplicando a regra delta)

- $erro_1 = d_1 - y_1 = 0 - 1.0 = -1.0$
- $w_{10} = w_{10} + \eta \times erro_1 \times x_0 = 0.0 + 1 \times -1.0 \times 1 = -1.0$
- $w_{11} = w_{11} + \eta \times erro_1 \times x_1 = 0.0 + 1 \times -1.0 \times 0 = 0.0$
- $w_{12} = w_{12} + \eta \times erro_1 \times x_2 = 0.0 + 1 \times -1.0 \times 0 = 0.0$

Redes Neurais: Treinamento (Época 1)

- **Pesos atuais:** $w_{10} = -1.0, w_{11} = 0.0, w_{12} = 0.0$
- limiar: $Q(v_k) = \begin{cases} 1 & \text{se } v_k \geq 0 \\ 0 & \text{caso contrário} \end{cases}$, taxa de aprendizagem: $\eta = 1$

x1	x2	d
0	0	0
0	1	1
1	0	1
1	1	1



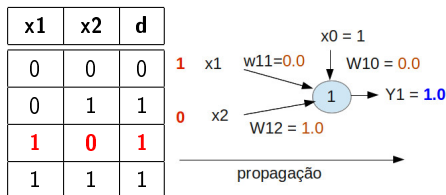
- $v_1 = x_0 \times w_{10} + x_1 \times w_{11} + x_2 \times w_{12}$
- $v_1 = 1 \times -1.0 + 0 \times 0.0 + 1 \times 0.0 = -1.0$
- $y_1 = Q(v_1) = Q(-1.0) = 0.0$

- Ajuste dos pesos (Aplicando a regra delta)

- $erro_1 = d_1 - y_1 = 1 - 0.0 = 1.0$
- $w_{10} = w_{10} + \eta \times erro_1 \times x_0 = -1.0 + 1 \times 1.0 \times 1 = 0.0$
- $w_{11} = w_{11} + \eta \times erro_1 \times x_1 = 0.0 + 1 \times 1.0 \times 0 = 0.0$
- $w_{12} = w_{12} + \eta \times erro_1 \times x_2 = 0.0 + 1 \times 1.0 \times 1 = 1.0$

Redes Neurais: Treinamento (Época 1)

- **Pesos atuais:** $w_{10} = 0.0, w_{11} = 0.0, w_{12} = 1.0$
- limiar: $Q(v_k) = \begin{cases} 1 & \text{se } v_k \geq 0 \\ 0 & \text{caso contrário} \end{cases}$, taxa de aprendizagem: $\eta = 1$

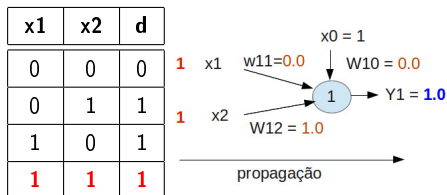


- $v_1 = x_0 \times w_{10} + x_1 \times w_{11} + x_2 \times w_{12}$
- $v_1 = 1 \times 0.0 + 1 \times 0.0 + 0 \times 1.0 = 0.0$
- $y_1 = Q(v_1) = Q(0.0) = 1.0$

- Ajuste dos pesos (Aplicando a regra delta)
 - $erro_1 = d_1 - y_1 = 1 - 1.0 = 0.0$
 - quando o erro é zero, não há ajuste de pesos.

Redes Neurais: Treinamento (Época 1)

- **Pesos atuais:** $w_{10} = 0.0, w_{11} = 0.0, w_{12} = 1.0$
- limiar: $Q(v_k) = \begin{cases} 1 & \text{se } v_k \geq 0 \\ 0 & \text{caso contrário} \end{cases}$, taxa de aprendizagem: $\eta = 1$



- $v_1 = x_0 \times w_{10} + x_1 \times w_{11} + x_2 \times w_{12}$
- $v_1 = 1 \times 0.0 + 1 \times 0.0 + 1 \times 1.0 = 1.0$
- $y_1 = Q(v_1) = Q(1.0) = 1.0$

- Ajuste dos pesos (Aplicando a regra delta)
 - $erro_1 = d_1 - y_1 = 1 - 1.0 = 0.0$
 - quando o erro é zero, não há ajuste de pesos.

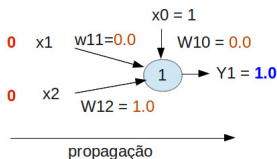
Redes Neurais: Treinamento (Época 1)

- Erro Acumulado para época 1 :
 - Erro Geral:
 - $= \text{abs}(\text{erroEntrada1}) + \text{abs}(\text{erroEntrada2}) + \text{abs}(\text{erroEntrada3}) + \text{abs}(\text{erroEntrada4}) =$
 - $= \text{abs}(-1.0) + \text{abs}(1.0) + \text{abs}(0.0) + \text{abs}(0.0)$
 - $= 2.0$
- Como ainda há erro, o algoritmo prossegue ...

Redes Neurais: Treinamento (Época 2)

- **Pesos atuais:** $w_{10} = 0.0, w_{11} = 0.0, w_{12} = 1.0$
- limiar: $Q(v_k) = \begin{cases} 1 & \text{se } v_k \geq 0 \\ 0 & \text{caso contrário} \end{cases}$, taxa de aprendizagem: $\eta = 1$

x1	x2	d
0	0	0
0	1	1
1	0	1
1	1	1



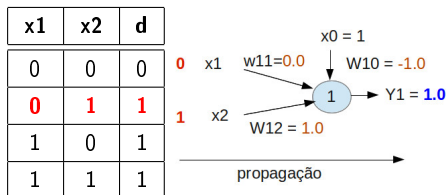
- $v_1 = x_0 \times w_{10} + x_1 \times w_{11} + x_2 \times w_{12}$
- $v_1 = 0 \times 0.0 + 0 \times 1.0 + 1 \times 0.0 = 0.0$
- $y_1 = Q(v_1) = Q(0.0) = 1.0$

• Ajuste dos pesos (Aplicando a regra delta)

- $erro_1 = d_1 - y_1 = 0 - 1.0 = -1.0$
- $w_{10} = w_{10} + \eta \times erro_1 \times x_0 = 0.0 + 1 \times -1.0 \times 1 = -1.0$
- $w_{11} = w_{11} + \eta \times erro_1 \times x_1 = 0.0 + 1 \times -1.0 \times 0 = 0.0$
- $w_{12} = w_{12} + \eta \times erro_1 \times x_2 = 1.0 + 1 \times -1.0 \times 0 = 1.0$

Redes Neurais: Treinamento (Época 2)

- **Pesos atuais:** $w_{10} = -1.0, w_{11} = 0.0, w_{12} = 1.0$
- limiar: $Q(v_k) = \begin{cases} 1 & \text{se } v_k \geq 0 \\ 0 & \text{caso contrário} \end{cases}$, taxa de aprendizagem: $\eta = 1$

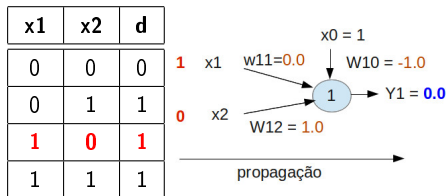


- $v_1 = x_0 \times w_{10} + x_1 \times w_{11} + x_2 \times w_{12}$
- $v_1 = 1 \times -1.0 + 0 \times 0.0 + 1 \times 1.0 = 0.0$
- $y_1 = Q(v_1) = Q(0.0) = 1.0$

- Ajuste dos pesos (Aplicando a regra delta)
 - $erro_1 = d_1 - y_1 = 1 - 1.0 = 0.0$
 - quando o erro é zero, não há ajuste de pesos.

Redes Neurais: Treinamento (Época 2)

- **Pesos atuais:** $w_{10} = -1.0, w_{11} = 0.0, w_{12} = 1.0$
- limiar: $Q(v_k) = \begin{cases} 1 & \text{se } v_k \geq 0 \\ 0 & \text{caso contrário} \end{cases}$, taxa de aprendizagem: $\eta = 1$



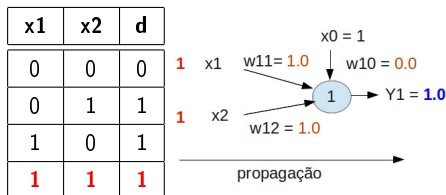
- $v_1 = x_0 \times w_{10} + x_1 \times w_{11} + x_2 \times w_{12}$
- $v_1 = 1 \times -1.0 + 1 \times 0.0 + 0 \times 1.0 = -1.0$
- $y_1 = Q(v_1) = Q(-1.0) = 0.0$

- Ajuste dos pesos (Aplicando a regra delta)

- $erro_1 = d_1 - y_1 = 1 - 0.0 = 1.0$
- $w_{10} = w_{10} + \eta \times erro_1 \times x_0 = -1.0 + 1 \times 1.0 \times 1 = 0.0$
- $w_{11} = w_{11} + \eta \times erro_1 \times x_1 = 0.0 + 1 \times 1.0 \times 1 = 1.0$
- $w_{12} = w_{12} + \eta \times erro_1 \times x_2 = 1.0 + 1 \times 1.0 \times 0 = 1.0$

Redes Neurais: Treinamento (Época 2)

- **Pesos atuais:** $w_{10} = 0.0, w_{11} = 1.0, w_{12} = 1.0$
- limiar: $Q(v_k) = \begin{cases} 1 & \text{se } v_k \geq 0 \\ 0 & \text{caso contrário} \end{cases}$, taxa de aprendizagem: $\eta = 1$



- $v_1 = x_0 \times w_{10} + x_1 \times w_{11} + x_2 \times w_{12}$
- $v_1 = 1 \times 0.0 + 1 \times 1.0 + 1 \times 1.0 = 2.0$
- $y_1 = Q(v_1) = Q(2.0) = 1.0$

- Ajuste dos pesos (Aplicando a regra delta)
 - $erro_1 = d_1 - y_1 = 1 - 1.0 = 0.0$
 - quando o erro é zero, não há ajuste de pesos.

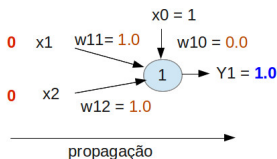
Redes Neurais: Treinamento (Época 2)

- Erro Acumulado para época 2 :
 - Erro Geral:
 - $= \text{abs}(\text{erroEntrada1}) + \text{abs}(\text{erroEntrada2}) + \text{abs}(\text{erroEntrada3}) + \text{abs}(\text{erroEntrada4}) =$
 - $= \text{abs}(-1.0) + \text{abs}(0.0) + \text{abs}(1.0) + \text{abs}(0.0)$
 - $= 2.0$
- Como ainda há erro, o algoritmo prossegue ...

Redes Neurais: Treinamento (Época 3)

- **Pesos atuais:** $w_{10} = 0.0, w_{11} = 1.0, w_{12} = 1.0$
- limiar: $Q(v_k) = \begin{cases} 1 & \text{se } v_k \geq 0 \\ 0 & \text{caso contrário} \end{cases}$, taxa de aprendizagem: $\eta = 1$

x1	x2	d
0	0	0
0	1	1
1	0	1
1	1	1



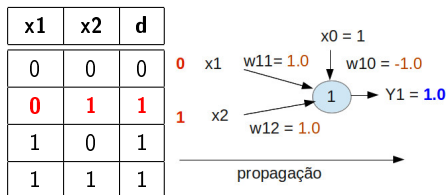
- $v_1 = x_0 \times w_{10} + x_1 \times w_{11} + x_2 \times w_{12}$
- $v_1 = 1 \times 0.0 + 0 \times 1.0 + 0 \times 1.0 = 0.0$
- $y_1 = Q(v_1) = Q(0.0) = 1.0$

- Ajuste dos pesos (Aplicando a regra delta)

- $erro_1 = d_1 - y_1 = 0 - 1.0 = -1.0$
- $w_{10} = w_{10} + \eta \times erro_1 \times x_0 = 0.0 + 1 \times -1.0 \times 1 = -1.0$
- $w_{11} = w_{11} + \eta \times erro_1 \times x_1 = 1.0 + 1 \times -1.0 \times 0 = 1.0$
- $w_{12} = w_{12} + \eta \times erro_1 \times x_2 = 1.0 + 1 \times -1.0 \times 0 = 1.0$

Redes Neurais: Treinamento (Época 3)

- **Pesos atuais:** $w_{10} = -1.0, w_{11} = 1.0, w_{12} = 1.0$
- limiar: $Q(v_k) = \begin{cases} 1 & \text{se } v_k \geq 0 \\ 0 & \text{caso contrário} \end{cases}$, taxa de aprendizagem: $\eta = 1$

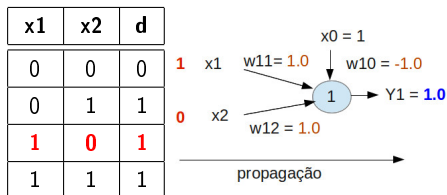


- $v_1 = x_0 \times w_{10} + x_1 \times w_{11} + x_2 \times w_{12}$
- $v_1 = 1 \times -1.0 + 0 \times 1.0 + 1 \times 1.0 = 0.0$
- $y_1 = Q(v_1) = Q(0.0) = 1.0$

- Ajuste dos pesos (Aplicando a regra delta)
 - $erro_1 = d_1 - y_1 = 1 - 1.0 = 0.0$
 - quando o erro é zero, não há ajuste de pesos.

Redes Neurais: Treinamento (Época 3)

- **Pesos atuais:** $w_{10} = -1.0, w_{11} = 1.0, w_{12} = 1.0$
- limiar: $Q(v_k) = \begin{cases} 1 & \text{se } v_k \geq 0 \\ 0 & \text{caso contrário} \end{cases}$, taxa de aprendizagem: $\eta = 1$

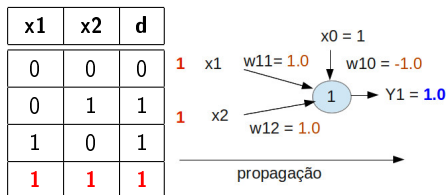


- $v_1 = x_0 \times w_{10} + x_1 \times w_{11} + x_2 \times w_{12}$
- $v_1 = 1 \times -1.0 + 1 \times 1.0 + 0 \times 1.0 = 0.0$
- $y_1 = Q(v_1) = Q(0.0) = 1.0$

- Ajuste dos pesos (Aplicando a regra delta)
 - $erro_1 = d_1 - y_1 = 1 - 1.0 = 0.0$
 - quando o erro é zero, não há ajuste de pesos.

Redes Neurais: Treinamento (Época 3)

- **Pesos atuais:** $w_{10} = -1.0, w_{11} = 1.0, w_{12} = 1.0$
- limiar: $Q(v_k) = \begin{cases} 1 & \text{se } v_k \geq 0 \\ 0 & \text{caso contrário} \end{cases}$, taxa de aprendizagem: $\eta = 1$



- $v_1 = x_0 \times w_{10} + x_1 \times w_{11} + x_2 \times w_{12}$
- $v_1 = 1 \times -1.0 + 1 \times 1.0 + 1 \times 1.0 = 1.0$
- $y_1 = Q(v_1) = Q(1.0) = 1.0$

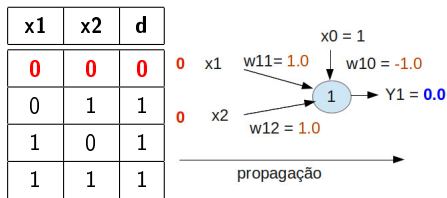
- Ajuste dos pesos (Aplicando a regra delta)
 - $erro_1 = d_1 - y_1 = 1 - 1.0 = 0.0$
 - quando o erro é zero, não há ajuste de pesos.

Redes Neurais: Treinamento (Época 3)

- Erro Acumulado para época 3 :
 - Erro Geral:
 - $= \text{abs}(\text{erroEntrada1}) + \text{abs}(\text{erroEntrada2}) + \text{abs}(\text{erroEntrada3}) + \text{abs}(\text{erroEntrada4}) =$
 - $= \text{abs}(-1.0) + \text{abs}(0.0) + \text{abs}(0.0) + \text{abs}(0.0)$
 - $= 1.0$
- Como ainda há erro, o algoritmo prossegue ...

Redes Neurais: Treinamento (Época 4)

- **Pesos atuais:** $w_{10} = -1.0, w_{11} = 1.0, w_{12} = 1.0$
- limiar: $Q(v_k) = \begin{cases} 1 & \text{se } v_k \geq 0 \\ 0 & \text{caso contrário} \end{cases}$, taxa de aprendizagem: $\eta = 1$

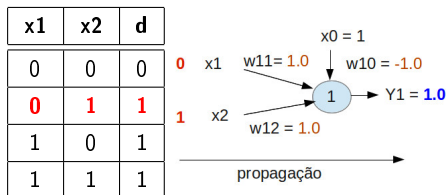


- $v_1 = x_0 \times w_{10} + x_1 \times w_{11} + x_2 \times w_{12}$
- $v_1 = 1 \times -1.0 + 0 \times 1.0 + 0 \times 1.0 = -1.0$
- $y_1 = Q(v_1) = Q(-1.0) = 0.0$

- Ajuste dos pesos (Aplicando a regra delta)
 - $erro_1 = d_1 - y_1 = 0 - 0.0 = 0.0$
 - quando o erro é zero, não há ajuste de pesos.

Redes Neurais: Treinamento (Época 4)

- **Pesos atuais:** $w_{10} = -1.0, w_{11} = 1.0, w_{12} = 1.0$
- limiar: $Q(v_k) = \begin{cases} 1 & \text{se } v_k \geq 0 \\ 0 & \text{caso contrário} \end{cases}$, taxa de aprendizagem: $\eta = 1$

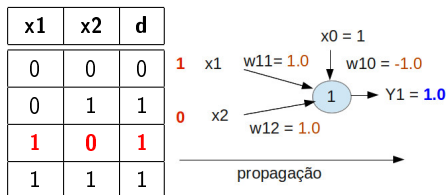


- $v_1 = x_0 \times w_{10} + x_1 \times w_{11} + x_2 \times w_{12}$
- $v_1 = 1 \times -1.0 + 0 \times 1.0 + 1 \times 1.0 = 0.0$
- $y_1 = Q(v_1) = Q(0.0) = 1.0$

- Ajuste dos pesos (Aplicando a regra delta)
 - $erro_1 = d_1 - y_1 = 1 - 1.0 = 0.0$
 - quando o erro é zero, não há ajuste de pesos.

Redes Neurais: Treinamento (Época 4)

- **Pesos atuais:** $w_{10} = -1.0, w_{11} = 1.0, w_{12} = 1.0$
- limiar: $Q(v_k) = \begin{cases} 1 & \text{se } v_k \geq 0 \\ 0 & \text{caso contrário} \end{cases}$, taxa de aprendizagem: $\eta = 1$

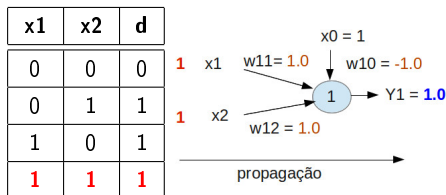


- $v_1 = x_0 \times w_{10} + x_1 \times w_{11} + x_2 \times w_{12}$
- $v_1 = 1 \times -1.0 + 1 \times 1.0 + 0 \times 1.0 = 0.0$
- $y_1 = Q(v_1) = Q(0.0) = 1.0$

- Ajuste dos pesos (Aplicando a regra delta)
 - $erro_1 = d_1 - y_1 = 1 - 1.0 = 0.0$
 - quando o erro é zero, não há ajuste de pesos.

Redes Neurais: Treinamento (Época 4)

- **Pesos atuais:** $w_{10} = -1.0, w_{11} = 1.0, w_{12} = 1.0$
- limiar: $Q(v_k) = \begin{cases} 1 & \text{se } v_k \geq 0 \\ 0 & \text{caso contrário} \end{cases}$, taxa de aprendizagem: $\eta = 1$



- $v_1 = x_0 \times w_{10} + x_1 \times w_{11} + x_2 \times w_{12}$
- $v_1 = 1 \times -1.0 + 1 \times 1.0 + 1 \times 1.0 = 1.0$
- $y_1 = Q(v_1) = Q(1.0) = 1.0$

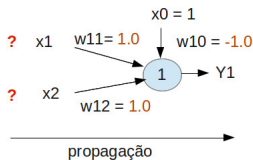
- Ajuste dos pesos (Aplicando a regra delta)
 - $erro_1 = d_1 - y_1 = 1 - 1.0 = 0.0$
 - quando o erro é zero, não há ajuste de pesos.

Redes Neurais: Treinamento (Época 4)

- Erro Acumulado para época 4 :
 - Erro Geral:
 - $= \text{abs}(\text{erroEntrada1}) + \text{abs}(\text{erroEntrada2}) + \text{abs}(\text{erroEntrada3}) + \text{abs}(\text{erroEntrada4}) =$
 - $= \text{abs}(0.0) + \text{abs}(0.0) + \text{abs}(0.0) + \text{abs}(0.0)$
 - $= 0.0$
 - O algoritmo pára.
- Resultado:
 - O algoritmo encontrou os pesos adequados:
 $w_{10} = -1.0, w_{11} = 1.0, w_{12} = 1.0$

Redes Neurais: Generalização

- A rede está pronta para ser testada (ou usada).
 - 1 Carrega-se, na topologia da rede, os pesos encontrados na fase de treinamento.



- 2 Esta fase implementa apenas a propagação. Informa-se novos valores para as entradas e a rede gera a saída correspondente.