Redes Neurais Artificiais

Perceptron

INFORMAÇÃO,

TECNOLOGIA

& INOVAÇÃO

Perceptron

- A primeira rede neural descrita algoritmicamente
- Criado por Frank Rosenblatt, um psicólogo, e inspirou engenheiros, físicos e matemáticos a estudarem redes neurais
- O modelo proposto por Rosenblatt em 1958 como publicado em seu artigo, é válido até hoje

Perceptron

- A forma mais simples de um rede neural utilizada para classificar padrões ditos **linearmente separáveis**

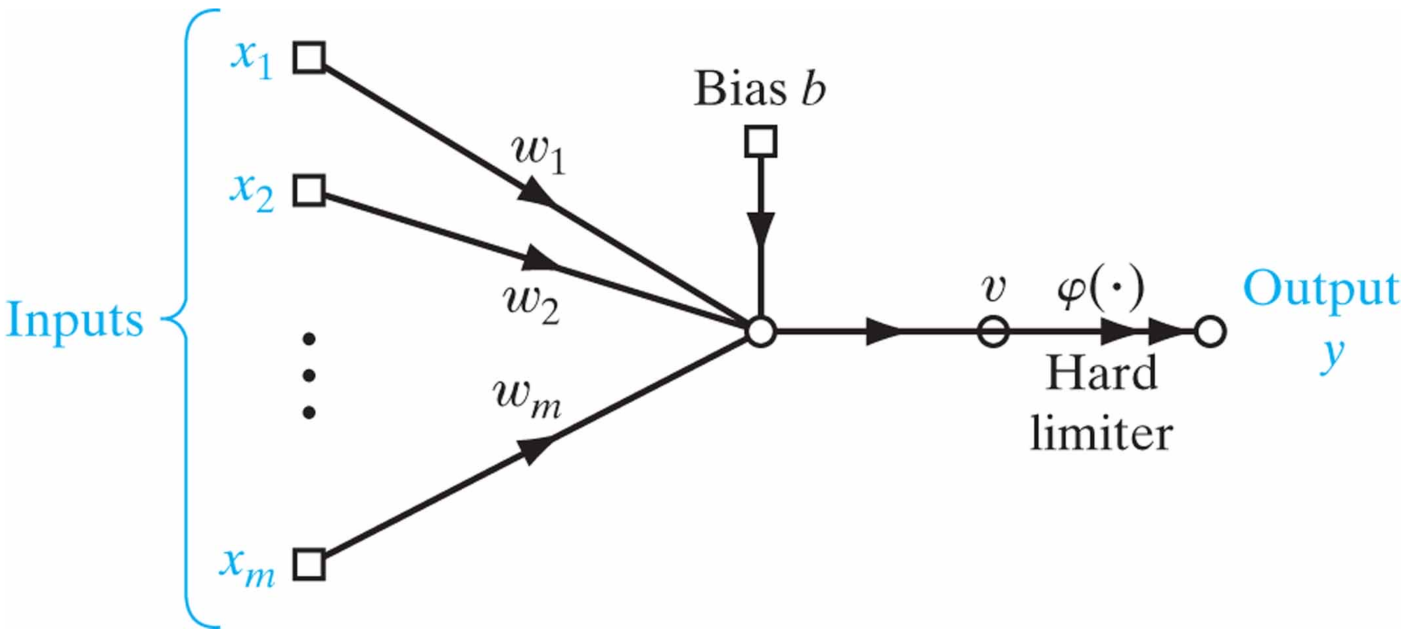
- Consiste de um único neurônio com pesos sinápticos ajustáveis e bias

- Rosenblatt desenvolveu o algoritmo para ajustar os parâmetros livres

- Rosenblatt provou que se os exemplos utilizados no treino pertencerem a classes linearmente separáveis, o algoritmo converge, posicionando um hiperplano entre as duas classes

Perceptron

- O Perceptron de Rosenblatt utiliza o modelo de neurônio de McCulloch-Pitts



Copyright ©2009 by Pearson Education, Inc.
Upper Saddle River, New Jersey 07458
All rights reserved.

Perceptron

- O Perceptron de Rosenblatt utiliza o modelo de neurônio de McCulloch-Pitts
  - Os pesos sinápticos são denotados por $w_1, w_2, \ldots, w_m$
  - As entradas são denotadas por $x_1, x_2, \ldots, x_m$
  - O bias é denotado por b

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

Perceptron

- O objetivo do Perceptron é classificar corretamente um conjunto de exemplos denotados por $x_1, \; x_2, \; ..., \; x_m$ em uma de umas classes $\mathscr{C}_1$ ou $\mathscr{C}_2$

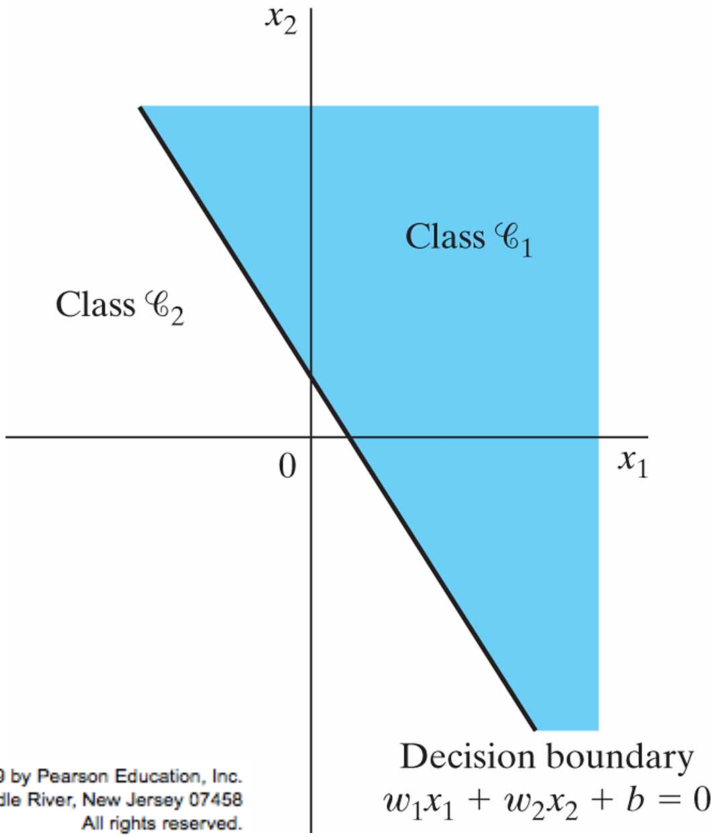
- O ponto representado por $x_1, \; x_2, \; ..., \; x_m$ é classificado como $\mathscr{C}_1$ se a saída y for +1 e como $\mathscr{C}_2$ se a saída y for -1. Há duas regiões separadas por um **hiperplano:**

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

Perceptron

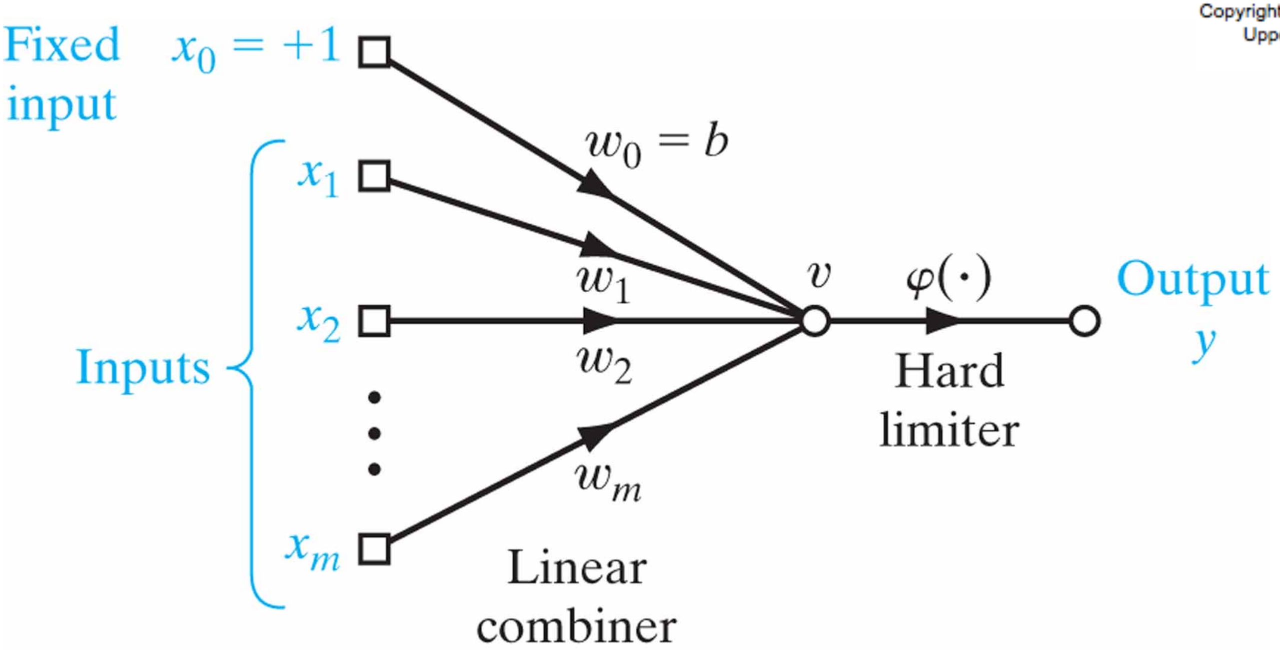
- Ilustração de um hiperplano (linha reta) como fronteira de decisão para um problema de classificação com duas dimensões e duas classes

- Os pesos sinápticos são ajustados em um processo iterativo utilizando o algoritmo de convergência do Perceptron



Copyright ©2009 by Pearson Education, Inc.
Upper Saddle River, New Jersey 07458
All rights reserved.

Teorema de Convergência do Perceptron

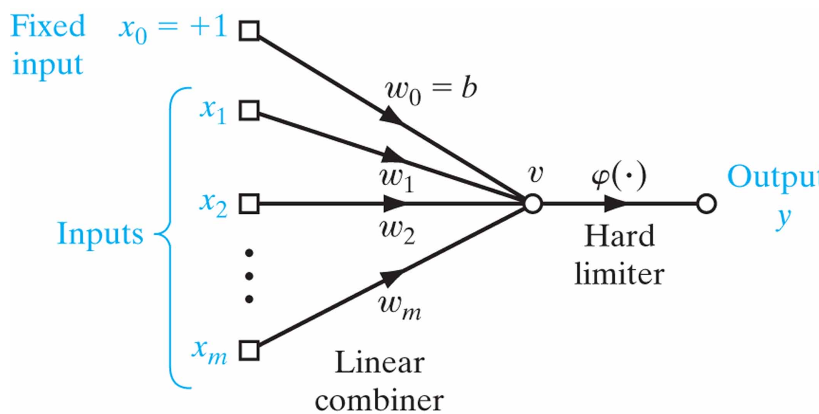


Copyright ©2009 by Pearson Education, Inc.
Upper Saddle River, New Jersey 07458
All rights reserved.

- O bias $b(n)$ é tratado como um peso associado a uma entrada +1
- Vetor de entrada: $\mathbf{x}(n) = [+1, x_1(n), x_2(n), ..., x_m(n)]^T$
- Vetor de pesos: $\mathbf{w}(n) = [b, w_1(n), w_2(n), ..., w_m(n)]^T$

Teorema de Convergência do Perceptron

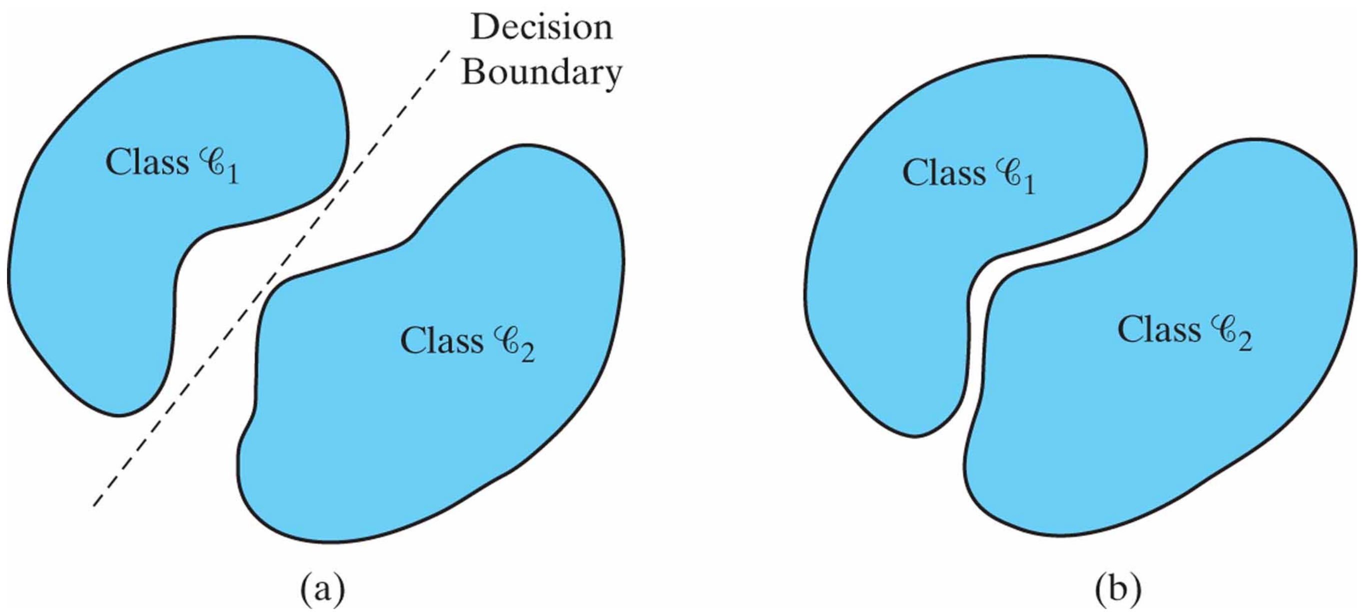
Copyright ©2009 by Pearson Education, Inc.
Upper Saddle River, New Jersey 07458
All rights reserved.



- O bias $b(n)$ é tratado como um peso associado a uma entrada +1
- Vetor de entrada: $\mathbf{x}(n) = [+1, x_1(n), x_2(n), \ldots, x_m(n)]^T$
- Vetor de pesos: $\mathbf{w}(n) = [b, w_1(n), w_2(n), \ldots, w_m(n)]^T$

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

Teorema de Convergência do Perceptron



Copyright ©2009 by Pearson Education, Inc.
Upper Saddle River, New Jersey 07458
All rights reserved.

- $\mathbf{w}^T\mathbf{x} = 0$ define um hiperplano de separação
- $\mathbf{w}^T\mathbf{x} > 0$ para todo vetor $\mathbf{x}$ pertencente à classe $\mathscr{C}_1$
- $\mathbf{w}^T\mathbf{x} \leq 0$ para todo vetor $\mathbf{x}$ pertencente à classe $\mathscr{C}_2$

Teorema de Convergência do Perceptron

- 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 $\mathbf{x}(n)$ pertence a classe $\mathscr{C}_1$
  - $\mathbf{w}(n+1) = \mathbf{w}(n)$ se $\mathbf{w}^T\mathbf{x}(n) \leq 0$ e $\mathbf{x}(n)$ pertence a classe $\mathscr{C}_2$

- Caso contrário, o vetor de pesos é atualizado
  - $\mathbf{w}(n+1) = \mathbf{w}(n) - \eta(n)\mathbf{x}(n)$ se $\mathbf{w}^T(n)\mathbf{x}(n) > 0$ e $\mathbf{x}(n)$ pertence a classe $\mathscr{C}_2$
  - $\mathbf{w}(n+1) = \mathbf{w}(n) + \eta(n)\mathbf{x}(n)$ **se** $\mathbf{w}^T(n)\mathbf{x}(n) \leq 0$ **e** $\mathbf{x}(n)$ **pertence a classe** $\mathscr{C}_1$

- $\eta(n)$ é a taxa de aprendizado que controla o ajuste dos pesos

Teorema de Convergência do Perceptron

- A saída do neurônio é computada utilizando a função sinal $sgn(\cdot)$

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

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

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

Teorema de Convergência do Perceptron

TABLE 1.1 Summary of the Perceptron Convergence Algorithm

Variables and Parameters:

$$\begin{aligned}
\mathbf{x}(n) &= (m+1)\text{-by-1 input vector} \\
&= [+1, x_1(n), x_2(n), \ldots, x_m(n)]^T \\
\mathbf{w}(n) &= (m+1)\text{-by-1 weight vector} \\
&= [b, w_1(n), w_2(n), \ldots, w_m(n)]^T \\
b &= \text{bias} \\
y(n) &= \text{actual response (quantized)} \\
d(n) &= \text{desired response} \\
\eta &= \text{learning-rate parameter, a positive constant less than unity}
\end{aligned}$$

1. *Initialization.* Set $\mathbf{w}(0) = \mathbf{0}$. Then perform the following computations for time-step $n = 1, 2, \ldots$.
2. *Activation.* At time-step n, activate the perceptron by applying continuous-valued input vector $\mathbf{x}(n)$ and desired response $d(n)$.
3. *Computation of Actual Response.* Compute the actual response of the perceptron as

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

where $\operatorname{sgn}(\cdot)$ is the signum function.

4. *Adaptation of Weight Vector.* Update the weight vector of the perceptron to obtain

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta[d(n) - y(n)]\mathbf{x}(n)$$

where

$$d(n) = \begin{cases} +1 & \text{if } \mathbf{x}(n) \text{ belongs to class } \mathscr{C}_1 \\ -1 & \text{if } \mathbf{x}(n) \text{ belongs to class } \mathscr{C}_2 \end{cases}$$

5. *Continuation.* Increment time step n by one and go back to step 2.

Teorema de Convergência do Perceptron

- No algoritmo de convergência, foi utilizada também a resposta desejada $d(n)$ para cada exemplo:

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

- A adaptação dos pesos ocorre de maneira elegante:

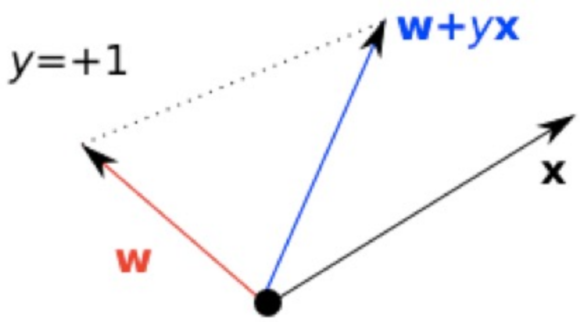
$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta[d(n) - y(n)]\mathbf{x}(n)$$

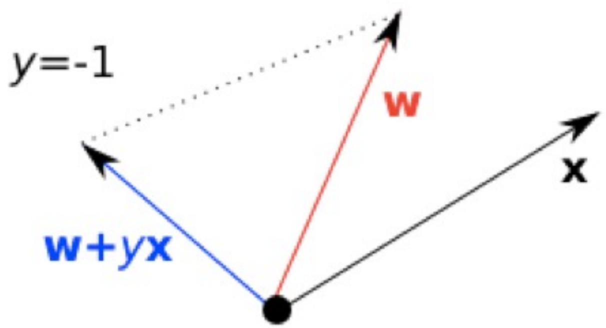
  - η : taxa de aprendizado
  - $d(n) - y(n)$: sinal de erro

Teorema de Convergência do Perceptron

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

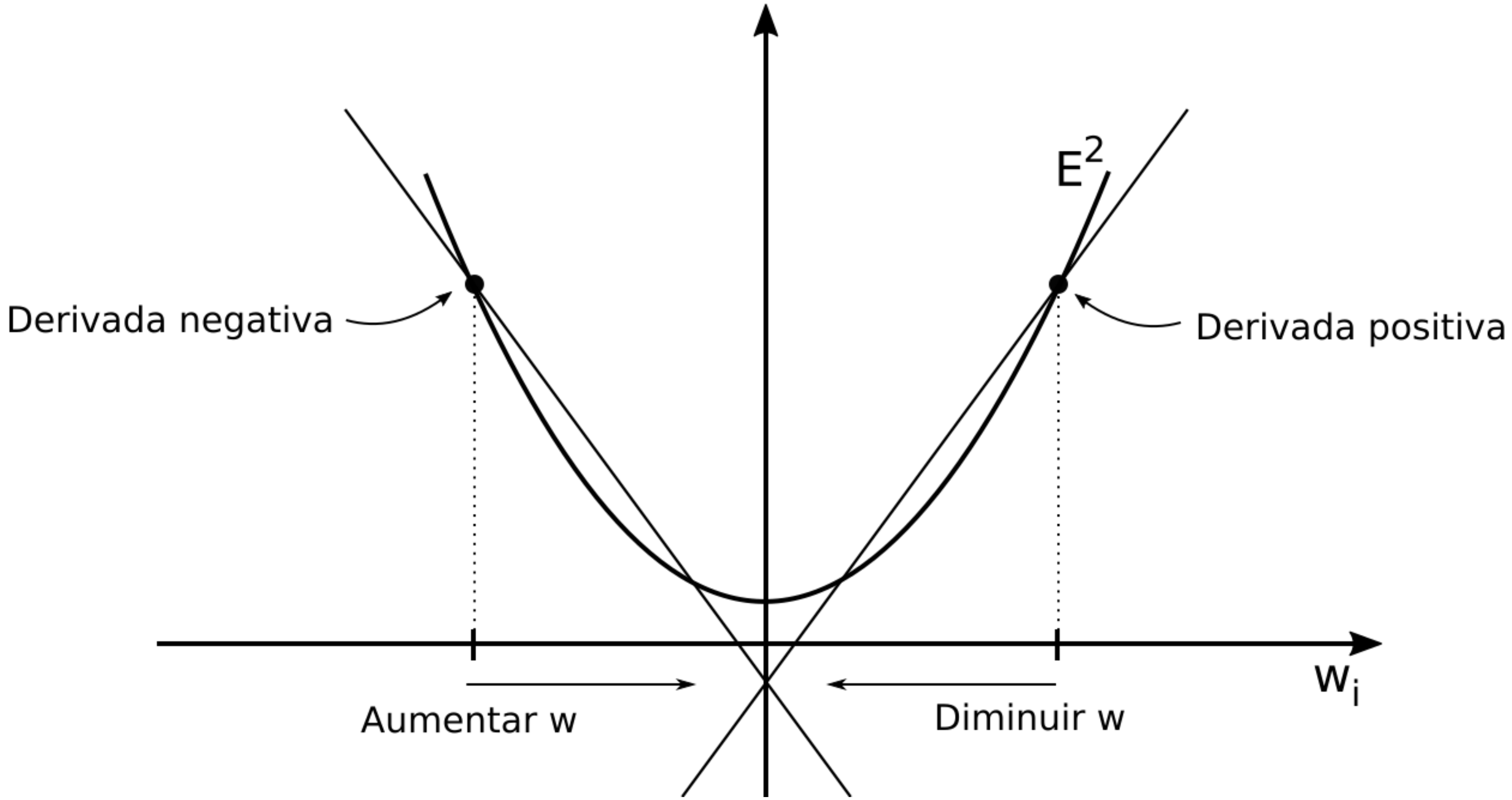
- Se o produto interno, na iteração *n*, tiver um sinal errado, os pesos devem ser ajustados para classificar o exemplo corretamente na iteração *n+1*





Como chegamos nisso?

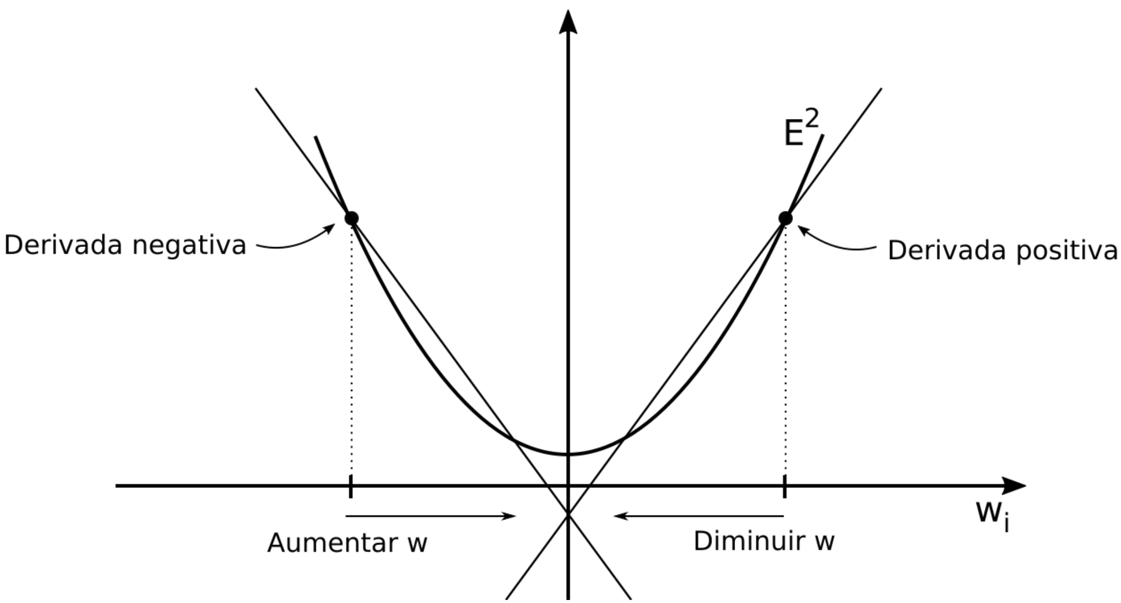
$$E^2 = \left(d(n) - y(n)\right)^2 = \left(d(n) - \mathbf{w}^T(n)\mathbf{x}(n)\right)^2$$



Como chegamos nisso?

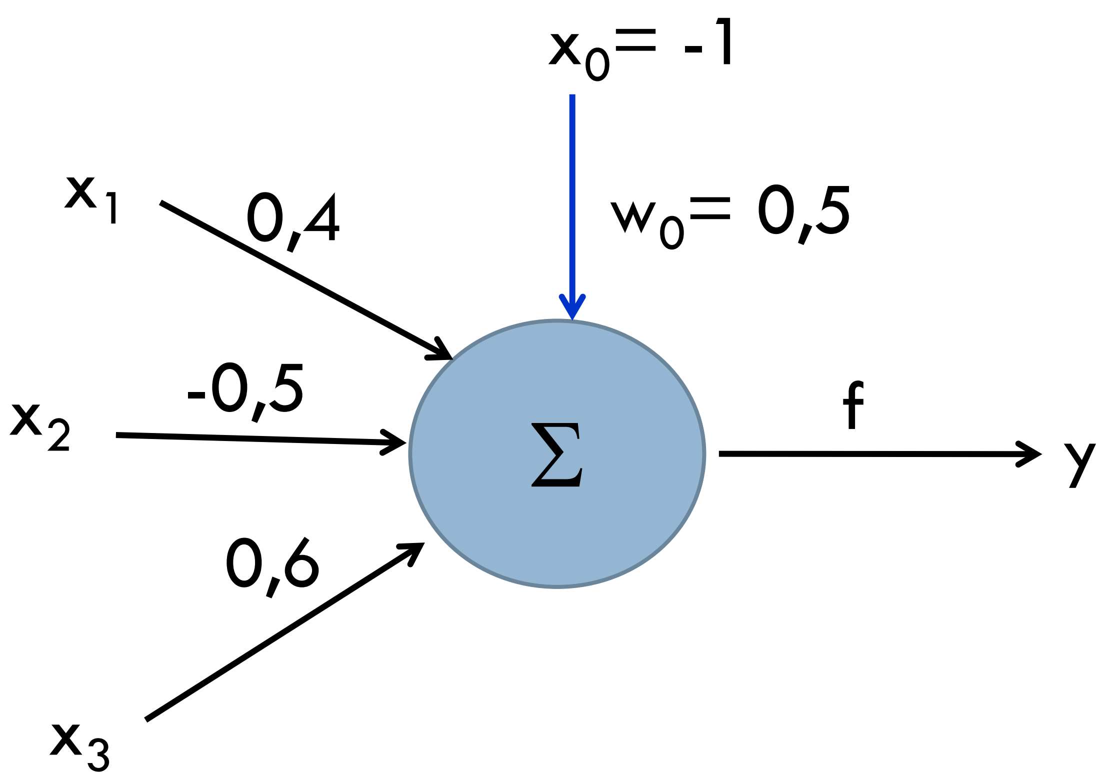
- Gradiente Descendente: $w_i(n+1) = w_i(n) - \eta \dfrac{dE^2}{dw_i}$

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



Exemplo

Dado	X1	X2	x3	Classe
E1	0	0	1	-1
E2	1	0	0	1



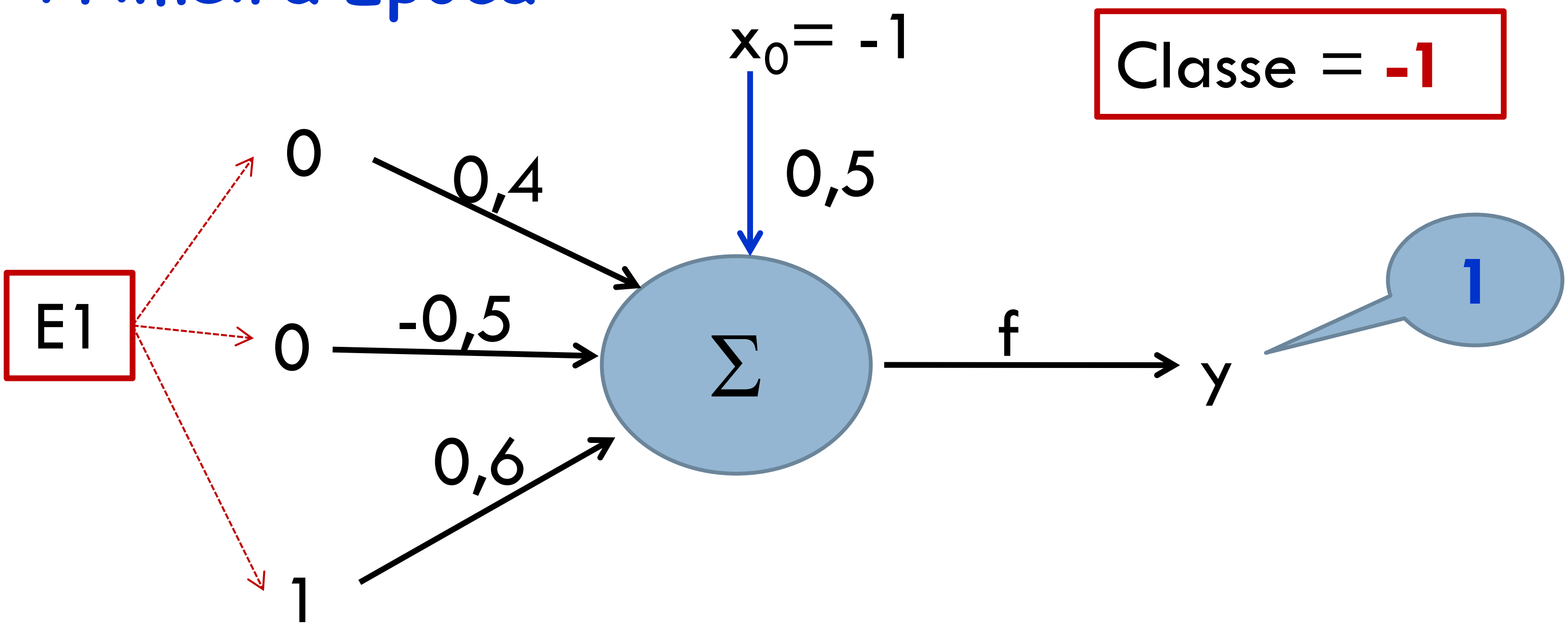
Exemplo

Dado	x1	x2	x3	Classe
E1	0	0	1	-1
E2	1	0	0	1

Aprendizado supervisionado

$$f(X) = \begin{cases} 1 & se \quad X \geq 0 \\ -1 & se \quad X < 0 \end{cases}$$

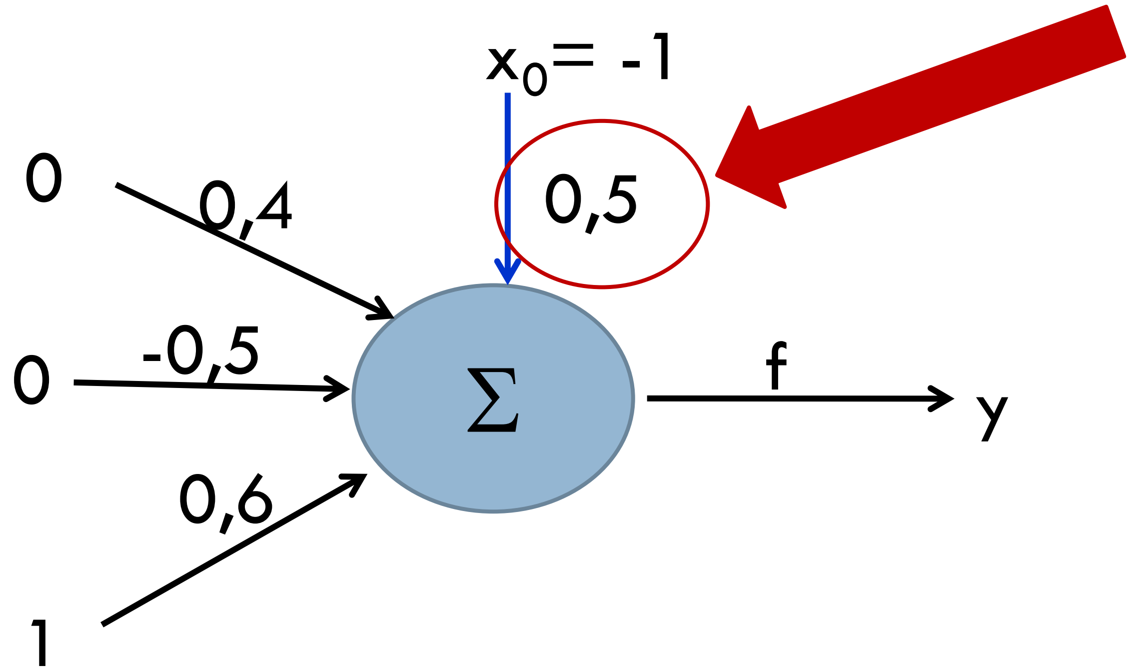
Primeira Época



$$X = (-1 * 0,5) + (0 * 0,4) + (0 * -0,5) + (1 * 0,6) = 0,1$$

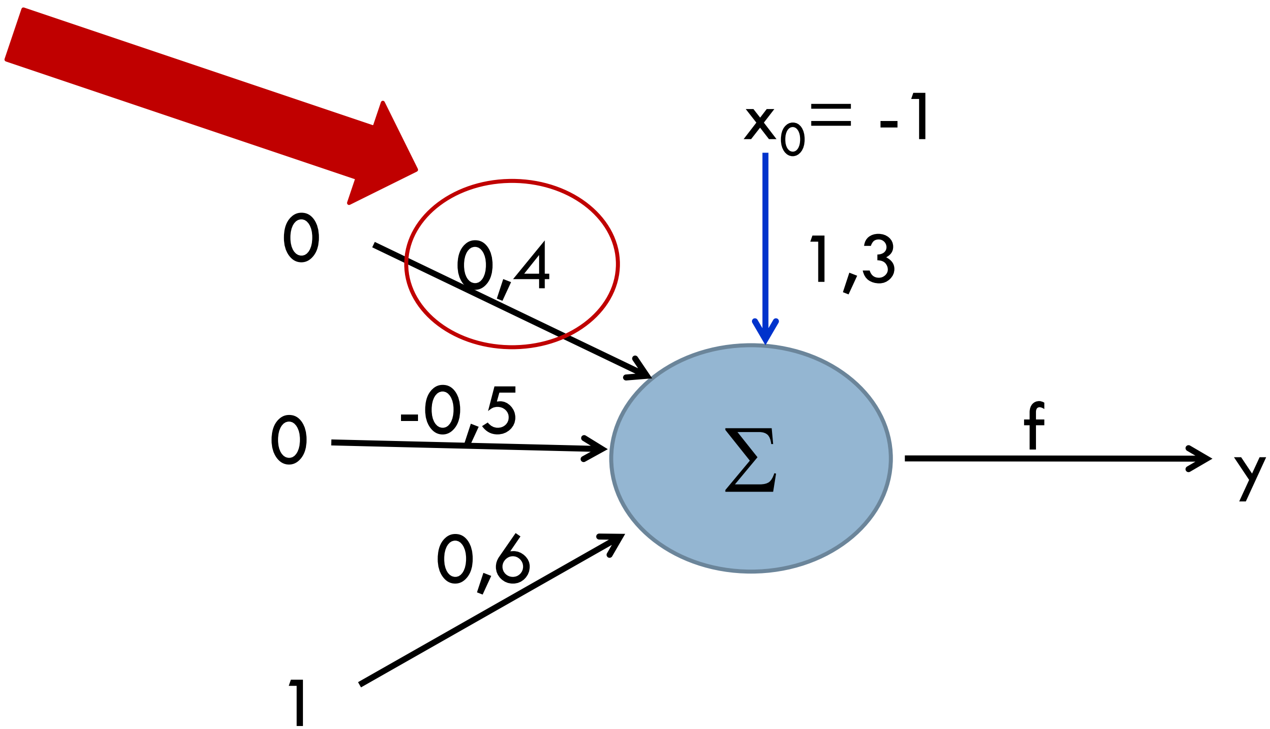
$$f(0,1) = 1$$

$$e = -1 - 1 = -2$$



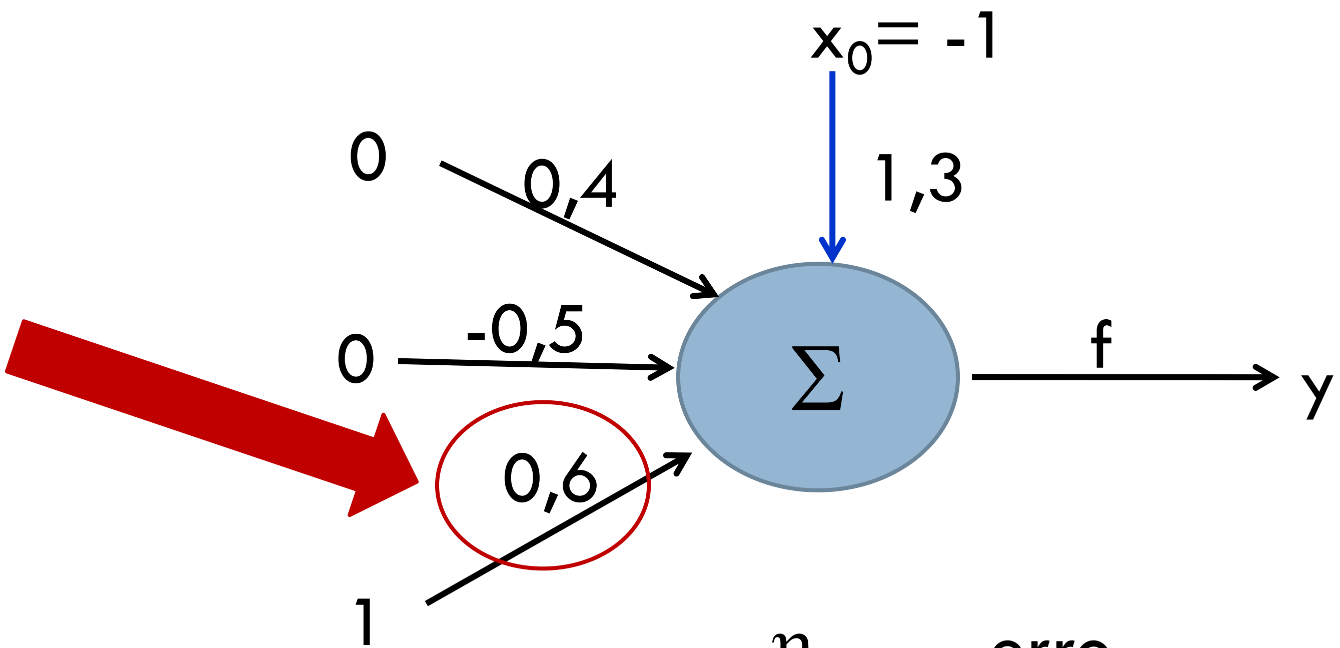
$$w_{0\_novo} = w_{0\_anterior} + (\overbrace{0,4}^{\eta} * \overbrace{(-2)}^{erro} * \overbrace{(-1)}^{x_0}) =$$

$$= 0,5 + 0,8 = 1,3$$



$$w_{1\_novo} = w_{1\_anterior} + (\underbrace{0,4}_{\eta} * \underbrace{(-2)}_{erro} * \underbrace{(0)}_{x_1}) =$$

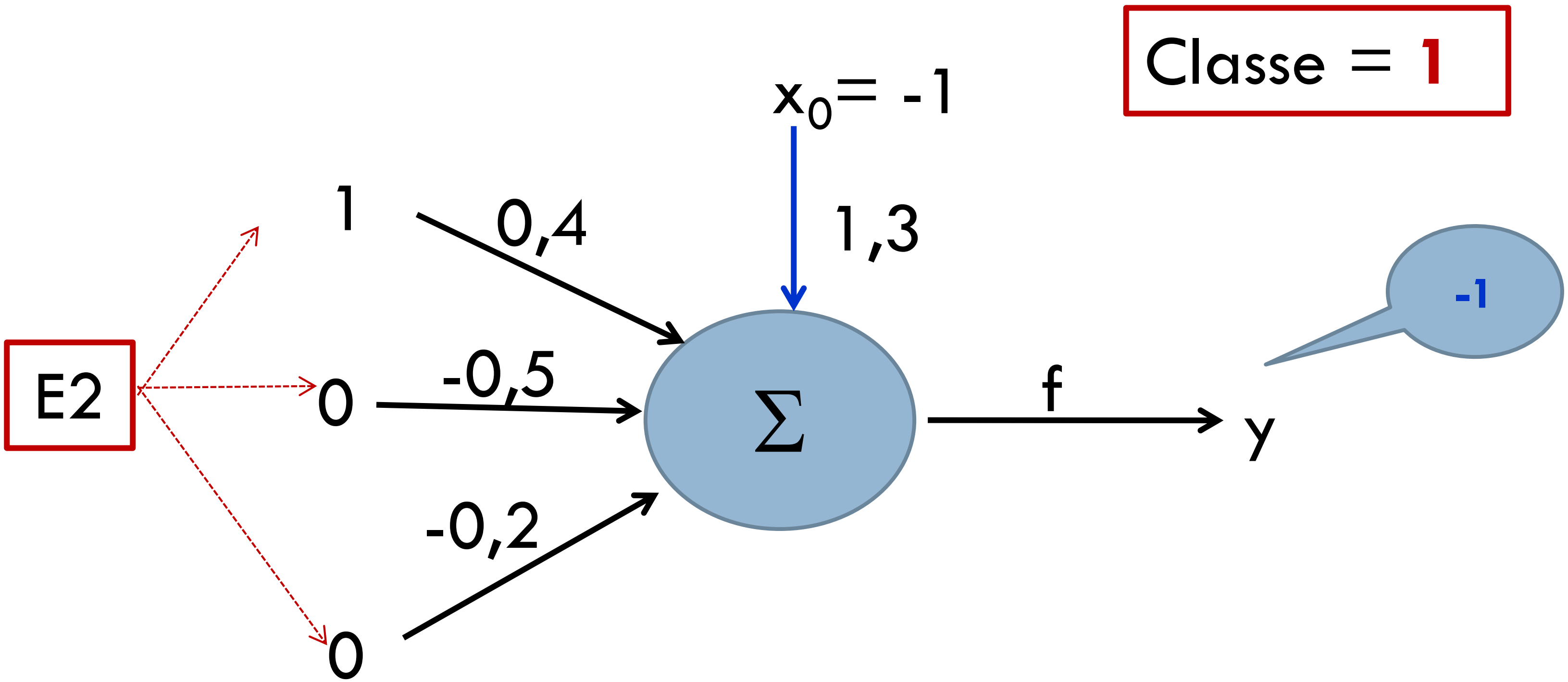
$$= 0,4$$



$$w_{3\_novo} = w_{3\_anterior} + (\overset{\eta}{0{,}4} * \overset{\text{erro}}{(-2)} * \overset{x_3}{(1)}) =$$

$$= 0{,}6 - 0{,}8$$

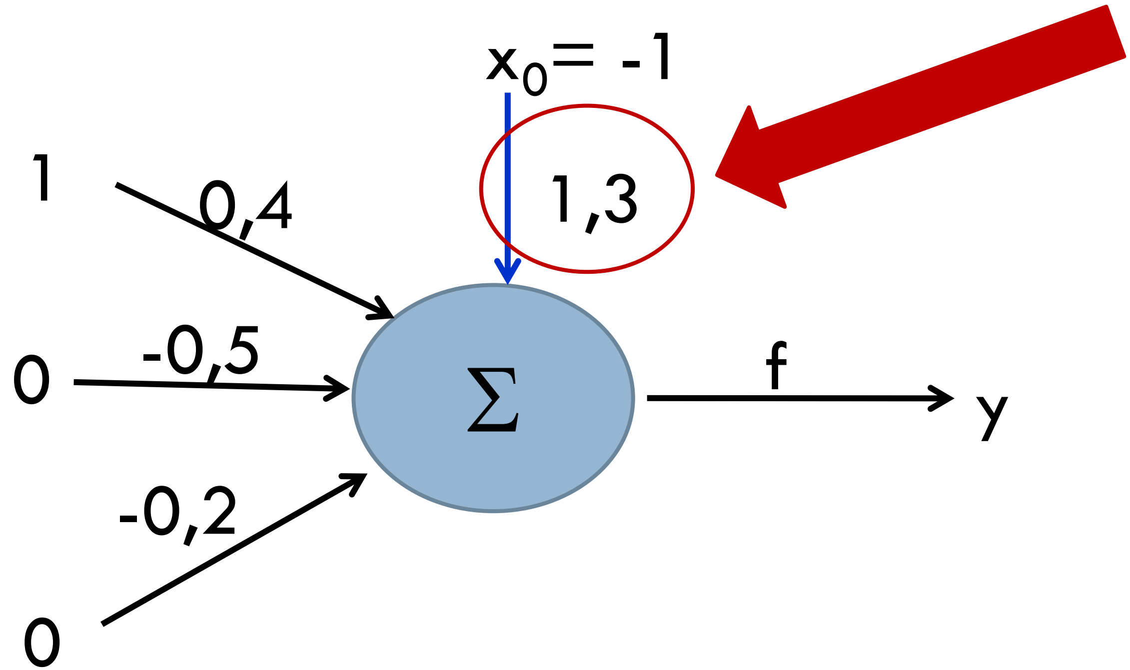
$$= -0{,}2$$



X = (-1 * 1,3) +(1 * 0,4) + (0 * -0,5) + (0 * -0,2) = -0,9

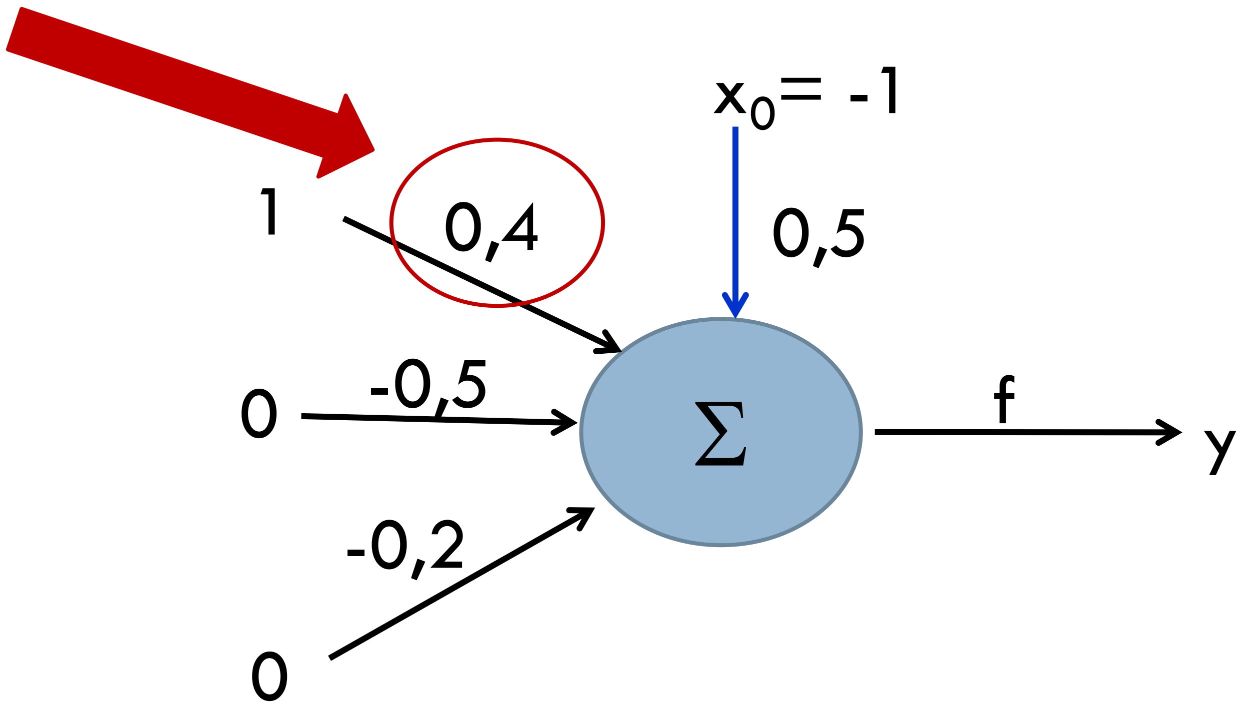
f(-0,9) = **-1**

e = **1** – **(-1)** = 2



$$w_{0\_novo} = w_{0\_anterior} + (\overbrace{0,4}^{\eta} * \overbrace{(2)}^{erro} * \overbrace{(-1)}^{x_0}) =$$

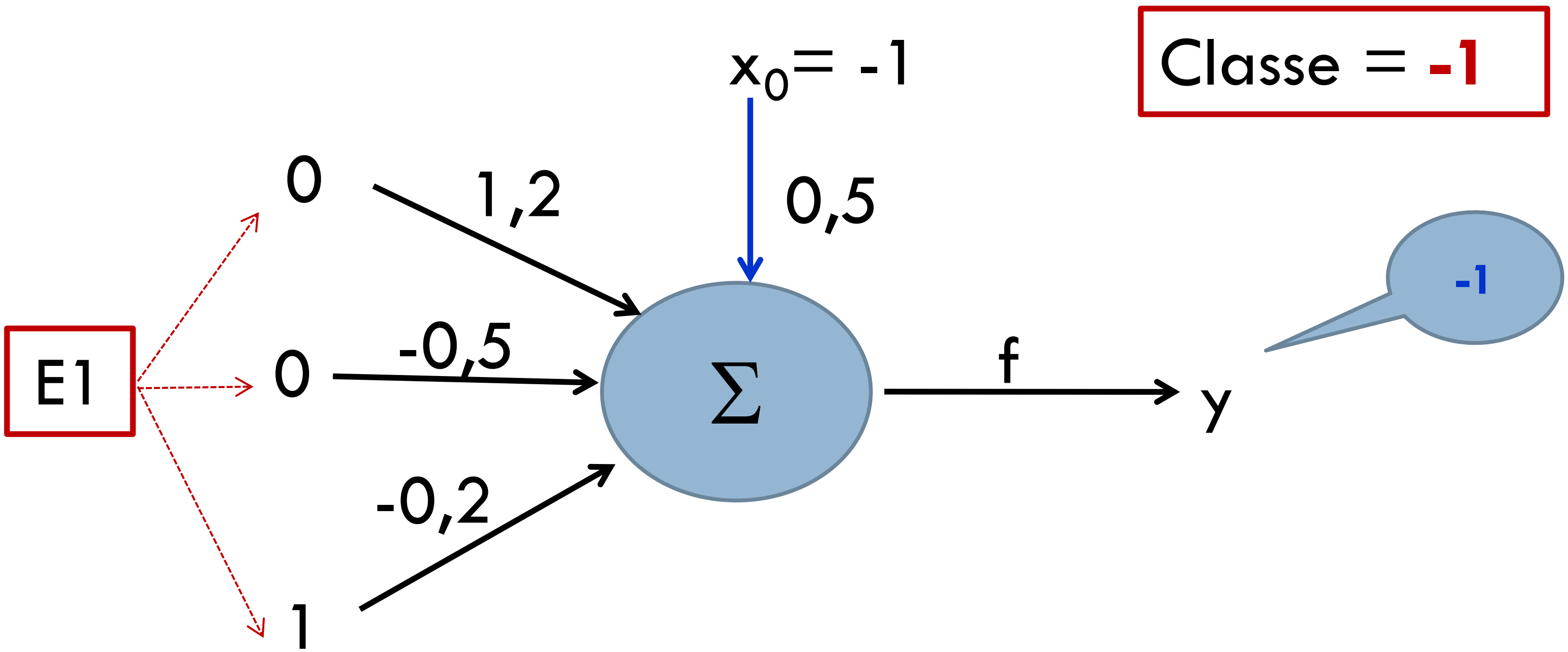
$$= 1,3 - 0,8 = 0,5$$



$$w_{1\text{—}novo} = w_{1\text{—}anterior} + (\overset{\eta}{0{,}4} * \overset{\text{erro}}{(2)} * \overset{x_0}{(1)}) =$$

$$= 0{,}4 + 0{,}8 = 1{,}2$$

Segunda Época

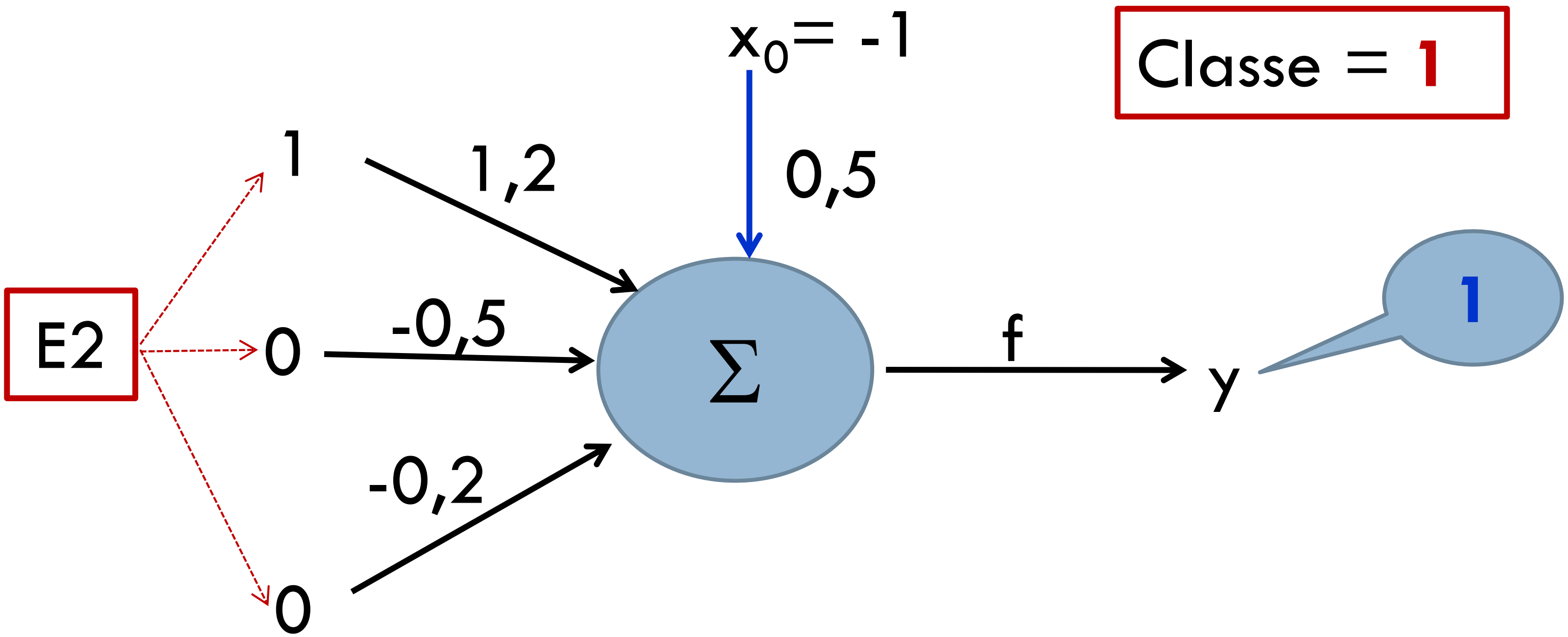


$$X = (-1 * 0{,}5) + (0 * 1{,}2) + (0 * -0{,}5) + (1 * -0{,}2) = -0{,}7$$

$$f(-0{,}7) = -1$$

$$e = -1 - (-1) = 0$$

Segunda Época

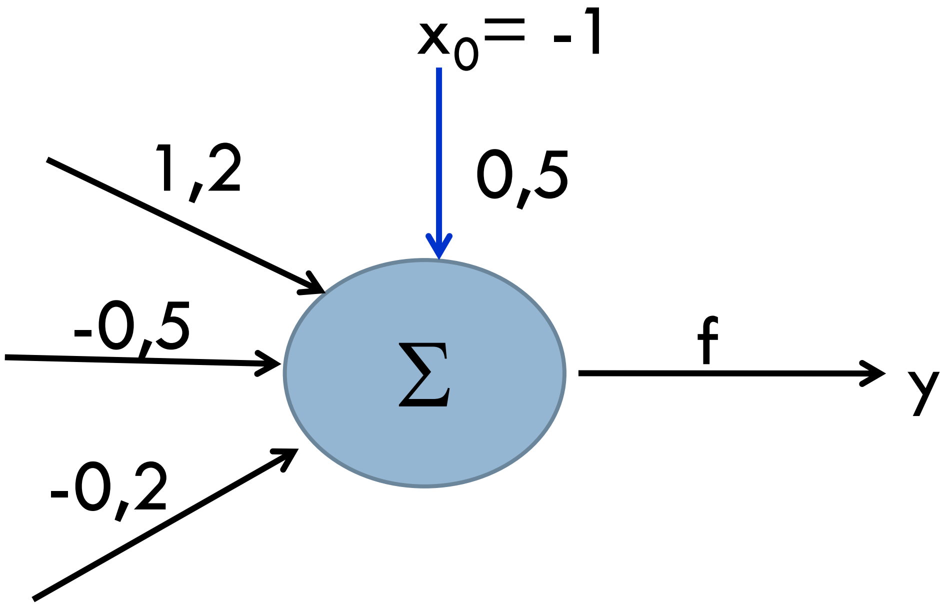


$$X = (-1 * 0{,}5) + (1 * 1{,}2) + (0 * -0{,}5) + (0 * -0{,}2) = 0{,}7$$

$$f(0{,}7) = 1$$

$$e = 1 - 1 = 0$$

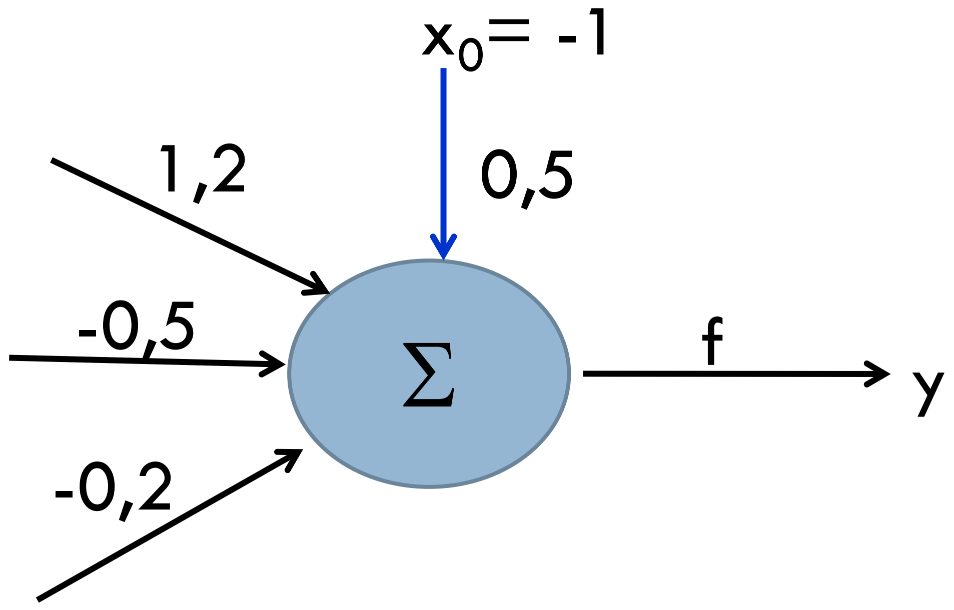
Rede final



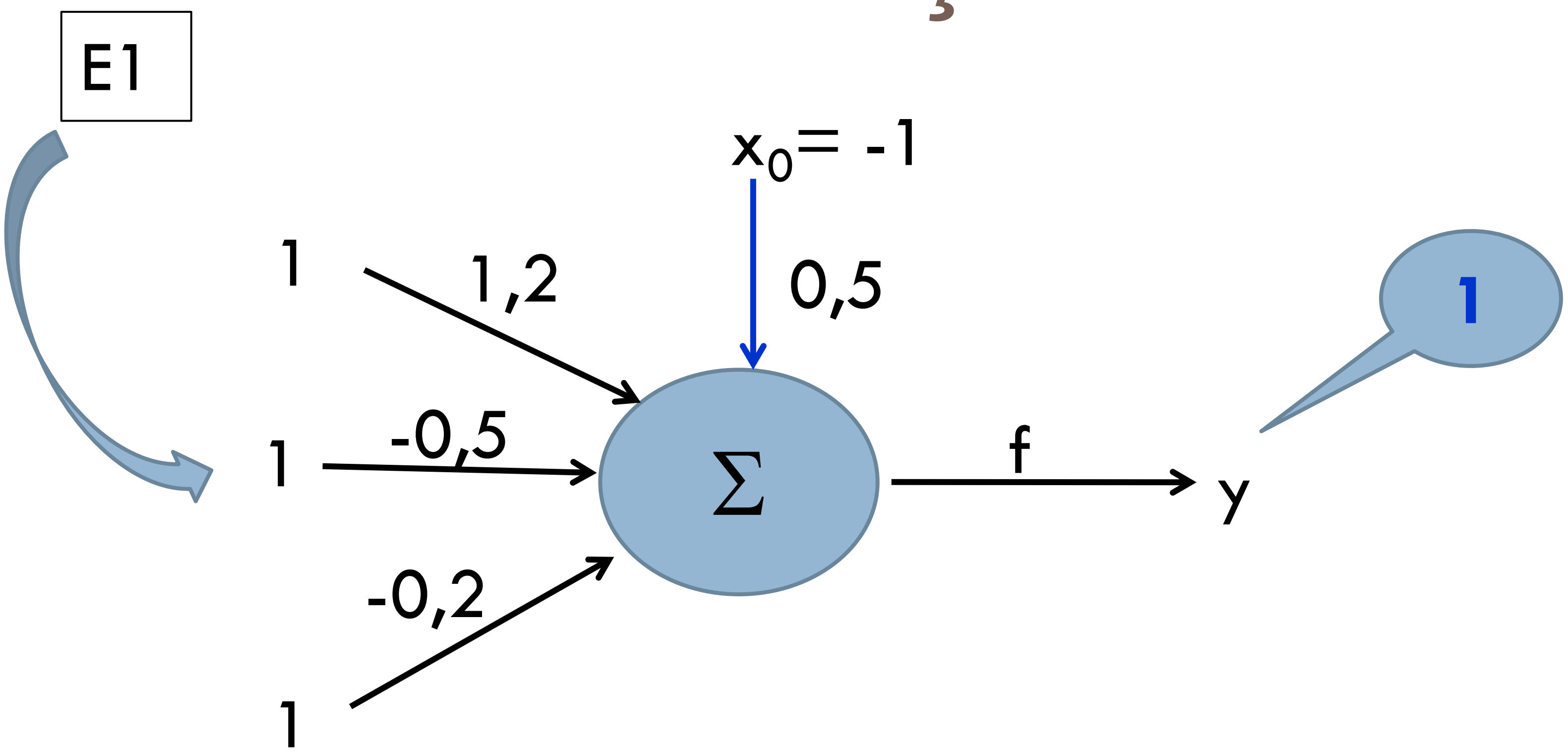
Não ocorreram erros durante a última época.

Classificação

Dado	X1	X2	x3	Classe
E1	1	1	1	?
E2	1	1	0	?
E3	0	1	1	?



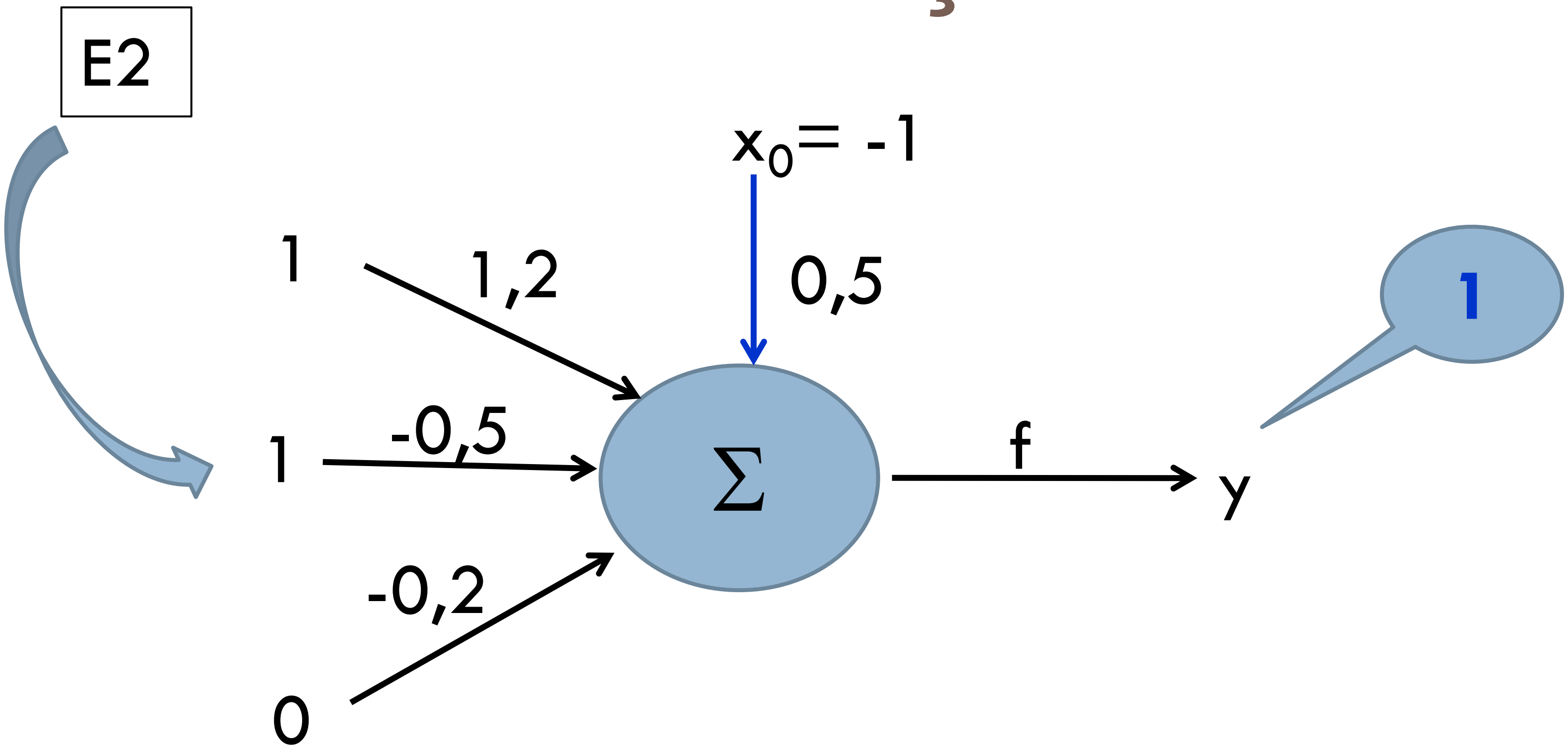
Classificação



$$X = (-1 * 0,5) + (1 * 1,2) + (1 * -0,5) + (1 * -0,2) = 0$$

$$f(0) = \mathbf{1}$$

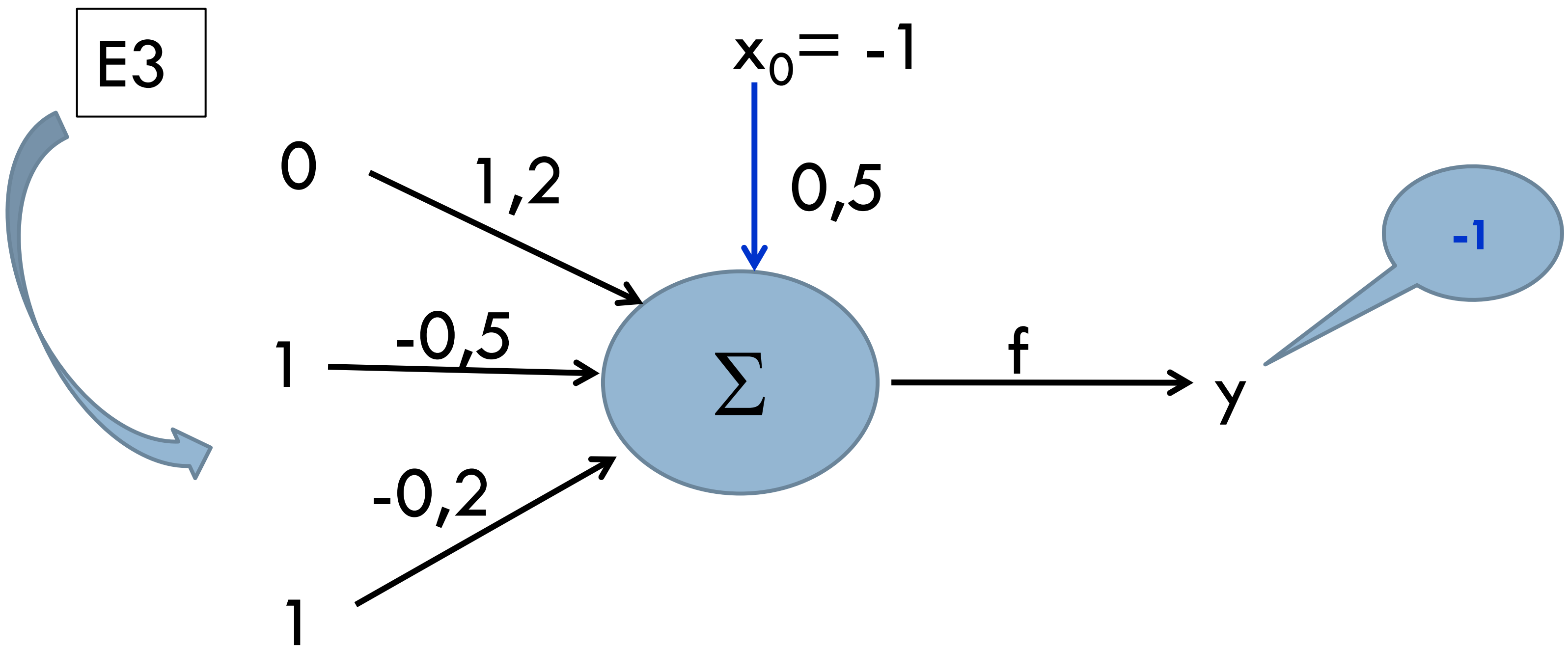
Classificação



$X = (-1 * 0{,}5) + (1 * 1{,}2) + (1 * -0{,}5) + (0 * -0{,}2) = 0{,}2$

$f(0{,}2) =$ **1**

Classificação

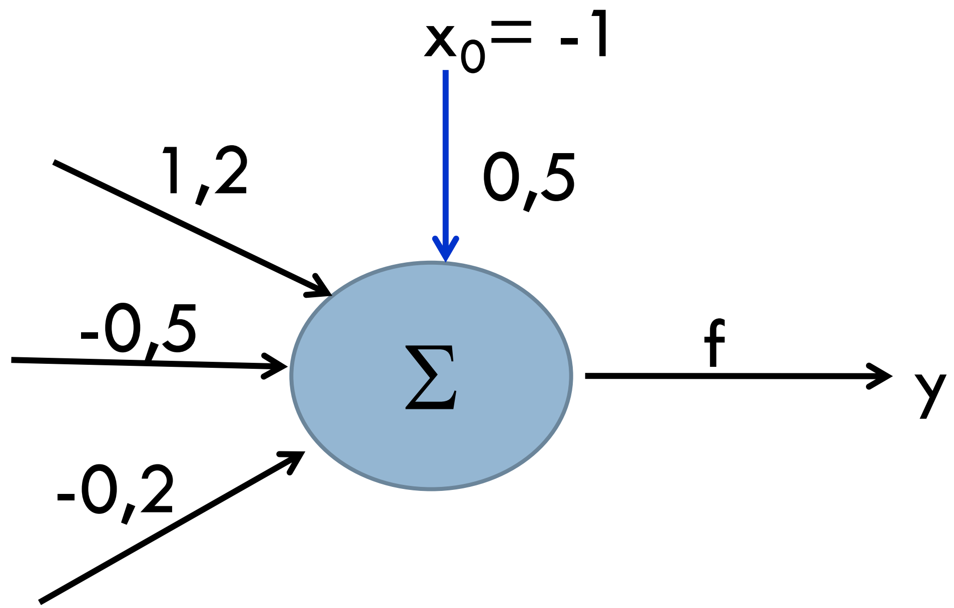


$$X = (-1 * 0{,}5) + (0 * 1{,}2) + (1 * -0{,}5) + (1 * -0{,}2) = -1{,}2$$

$$f(-1{,}2) = \mathbf{-1}$$

Classificação

Dado	**X1**	**X2**	**x3**	**Classe**
E1	1	1	1	**1**
E2	1	1	0	**1**
E3	0	1	1	**-1**



Perguntas?