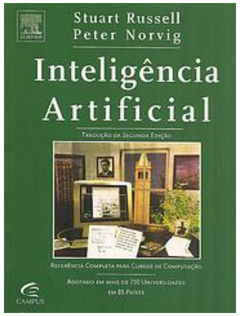
Departamento de Sistemas e Computação – FURB
Curso de Ciência da Computação
Disciplina de Processamento de Imagens

Redes Neurais

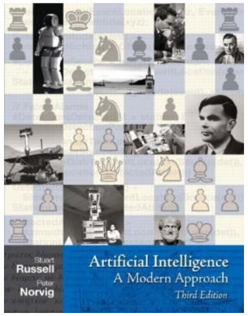
Prof. Aurélio Hoppe
aureliof@furb.br
http://www.inf.furb.br/~aurelio/

Grupo de Processamento de Imagens,
Análise de dados, Robótica e
Simulação computacional

Bibliografia



Stuart Russell e Peter Norvig. **Inteligência artificial**. Rio de Janeiro : Campus, 2004, 1021p.



Stuart Russell e Peter Norvig. Artificial Intelligence: a Modern Approach, 3nd Edition, Prentice-Hall, 2009.

Mitchell, T. **Machine Learning**, McGraw–Hill Science/Engineering/Math, 1997.

Formas de Aprendizado

- **Aprendizado Supervisionado**
  - K-Nearest Neighbor (KNN)
  - **Redes Neurais**
  - Support Vector Machines (SVM)
  - Árvores de Decisão

- Aprendizado Não Supervisionado

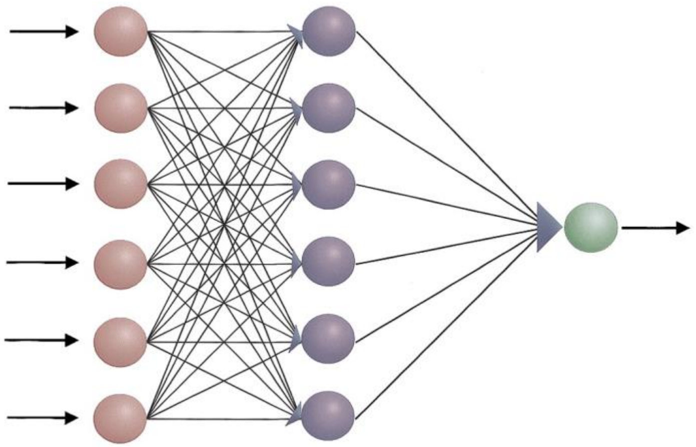
- Aprendizado Por Reforço

Introdução

- **Redes Neurais** podem ser consideradas um paradigma diferente de computação.

- Inspirado na **arquitetura paralela** do cérebro humano.

  – Elementos de processamento simples.
  – Grande grau de interconexões.
  – Interação adaptativa entre os elementos.

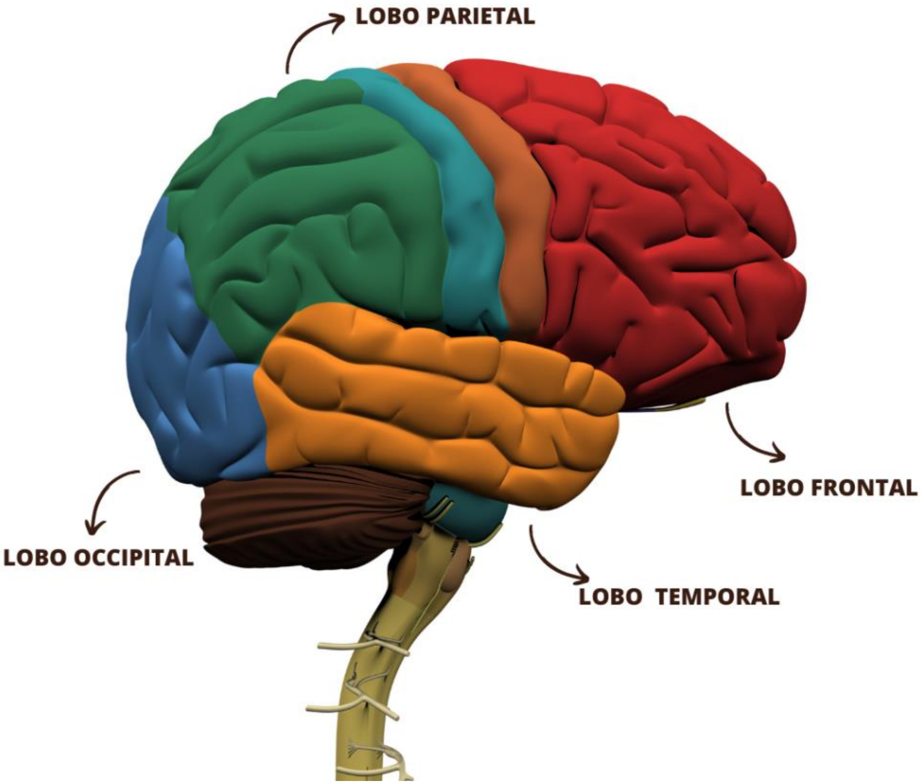


Introdução

- No cérebro, o **comportamento inteligente** é uma propriedade emergente de um grande número de **unidades simples** (ao contrário do que acontece com regras e algoritmos simbólicos).
- Neurônios ligam e desligam em alguns milissegundos, enquanto o hardware atual faz o mesmo em nano segundos.
  - Entretanto, o cérebro realiza tarefas cognitivas complexas (visão, reconhecimento de voz) em décimos de segundo.
- O cérebro deve estar utilizando um **paralelismo massivo**.

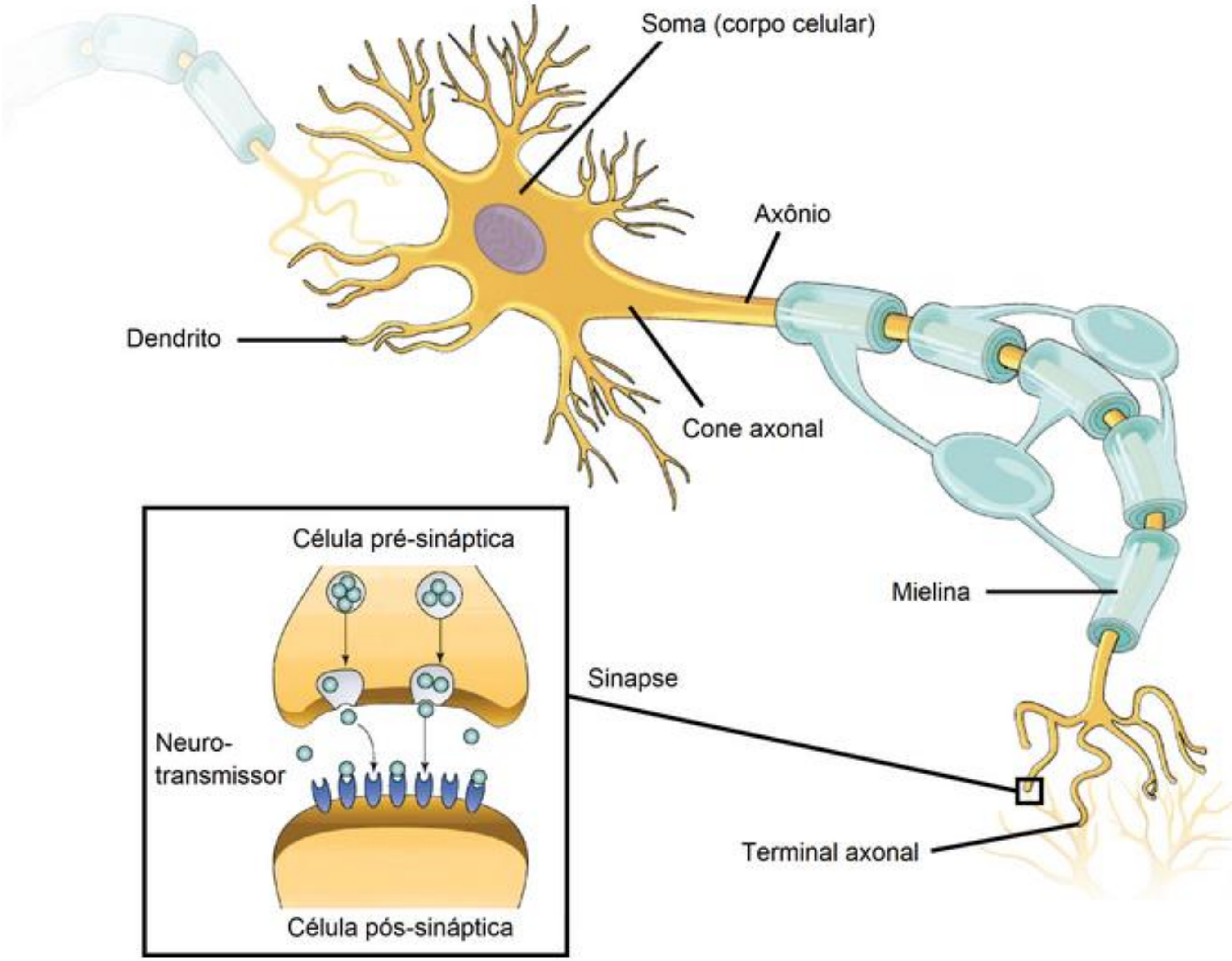
Introdução

- O **cérebro humano** tem sido extensamente estudado, mas ainda não somos capazes de **entender completamente** o seu funcionando.



- O cérebro é **muito complexo**, até mesmo o comportamento de um simples **neurônio** é bem complexo.

Neurônio



Soma (corpo celular)
Axônio
Dendrito
Cone axonal
Célula pré-sináptica
Mielina
Sinapse
Neuro-
transmissor
Terminal axonal
Célula pós-sináptica

Funcionamento de um Neurônio

- Através dos **dentritos**, o neurônio recebe sinais de outros neurônios a ele conectados por meio das **sinapses**.

- Os sinais são acumulados no **corpo** do neurônio.

- Quando a soma dos sinais passa de um certo limiar (~ 50mV) um sinal é propagado no **axônio**.

- As **sinapses** tem um peso que pode ser:
  - excitatório: incrementam a soma dos sinais.
  - inibidor: decrementam.

Introdução

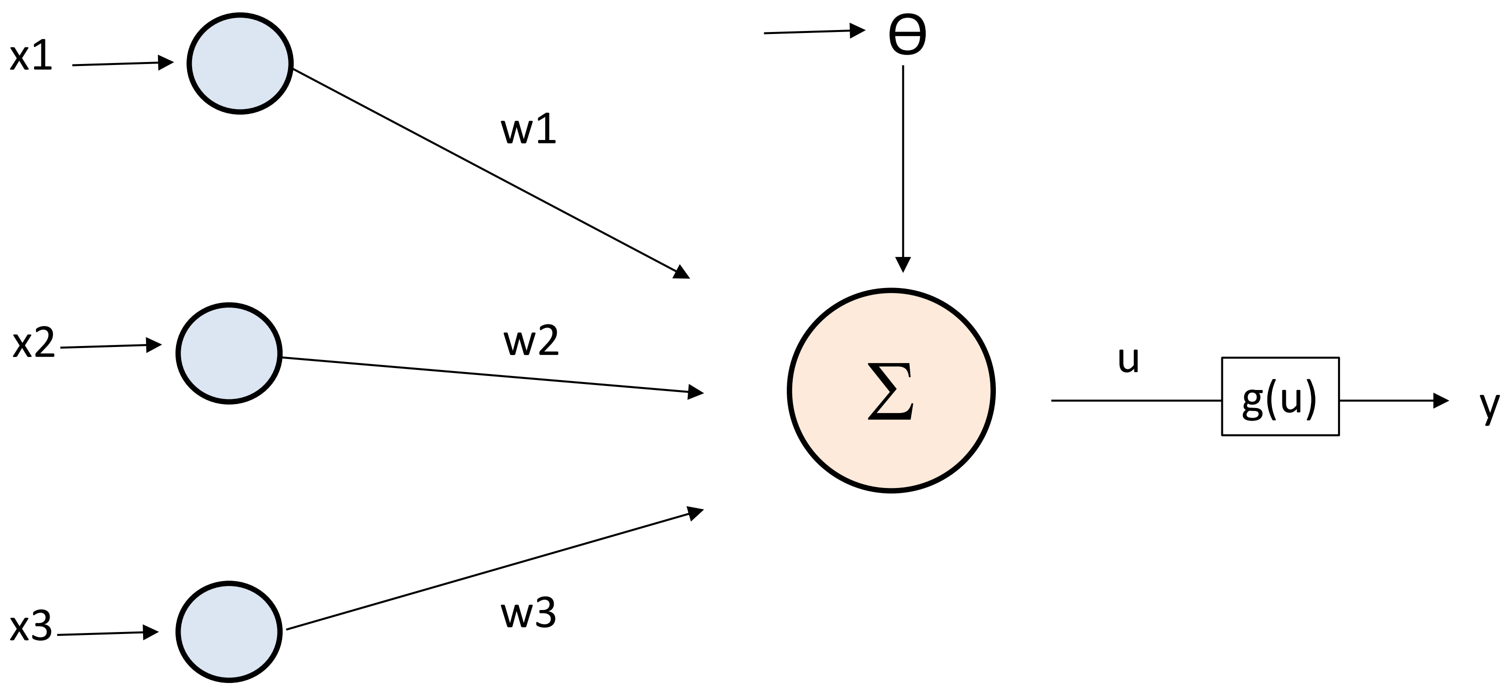
- **Características do Cérebro Humano:**
  - 10^{11} neurônios.
  - Cada neurônio tem em media 10^4 conexões.
  - Milhares de operações por segundo.
  - Neurônios morrem frequentemente e nunca são substituídos.
  - Reconhecimento de faces em aproximadamente 0.1 segundos.

Introdução

- O **cérebro humano** é bom em:
  - Reconhecer padrões,
  - Associação,
  - Tolerar ruídos...

- O **computador** é bom em:
  - Cálculos,
  - Precisão,
  - Lógica.

Introdução

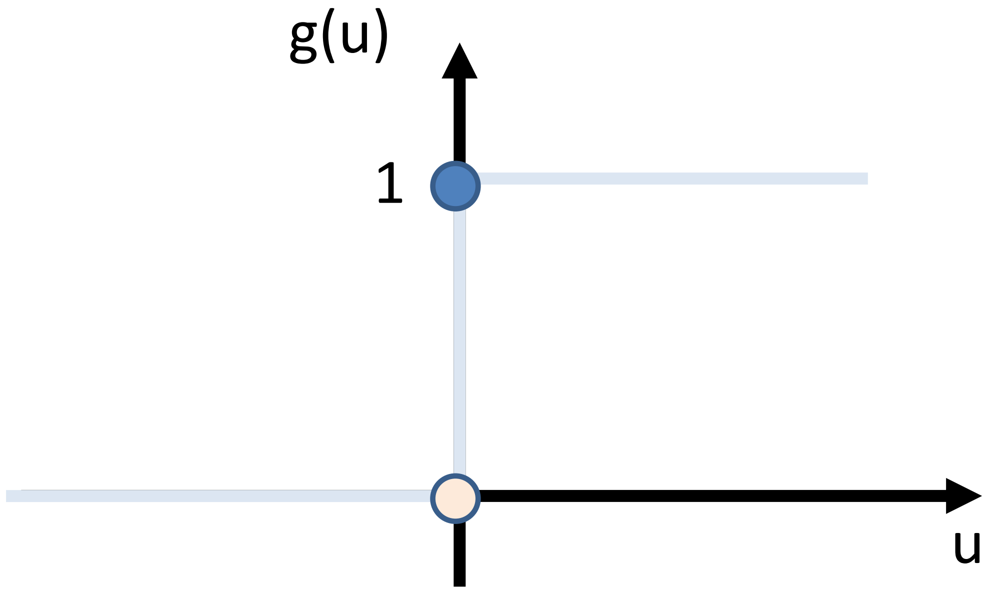


- Neurônio
  - Potencial de ativação { u }
  - Função de Ativação { g }
  - Sinal de saída { y }

Introdução

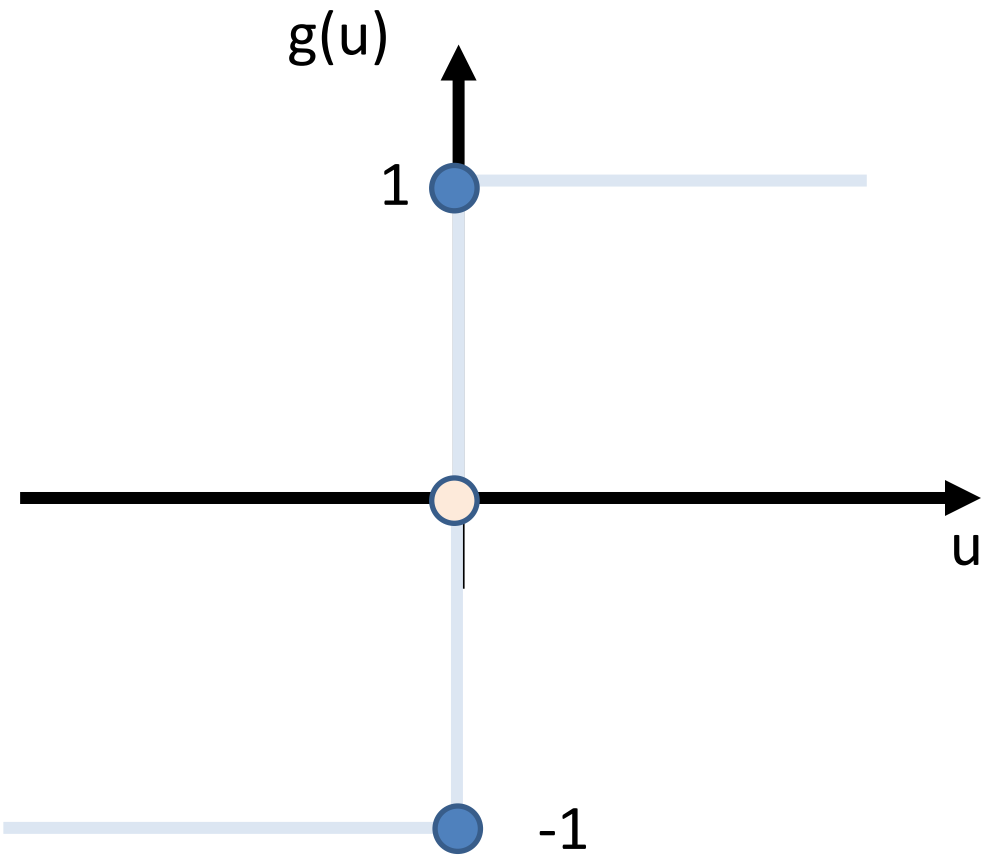
- Funções de ativação
  - Função Degrau

$$G(u) = \begin{cases} 1 & \text{Se } (u \geq 0) \\ 0 & \text{Se } (u < 0) \end{cases}$$



Introdução

- Funções de ativação
  - Função Degrau Bipolar (Função Sinal)

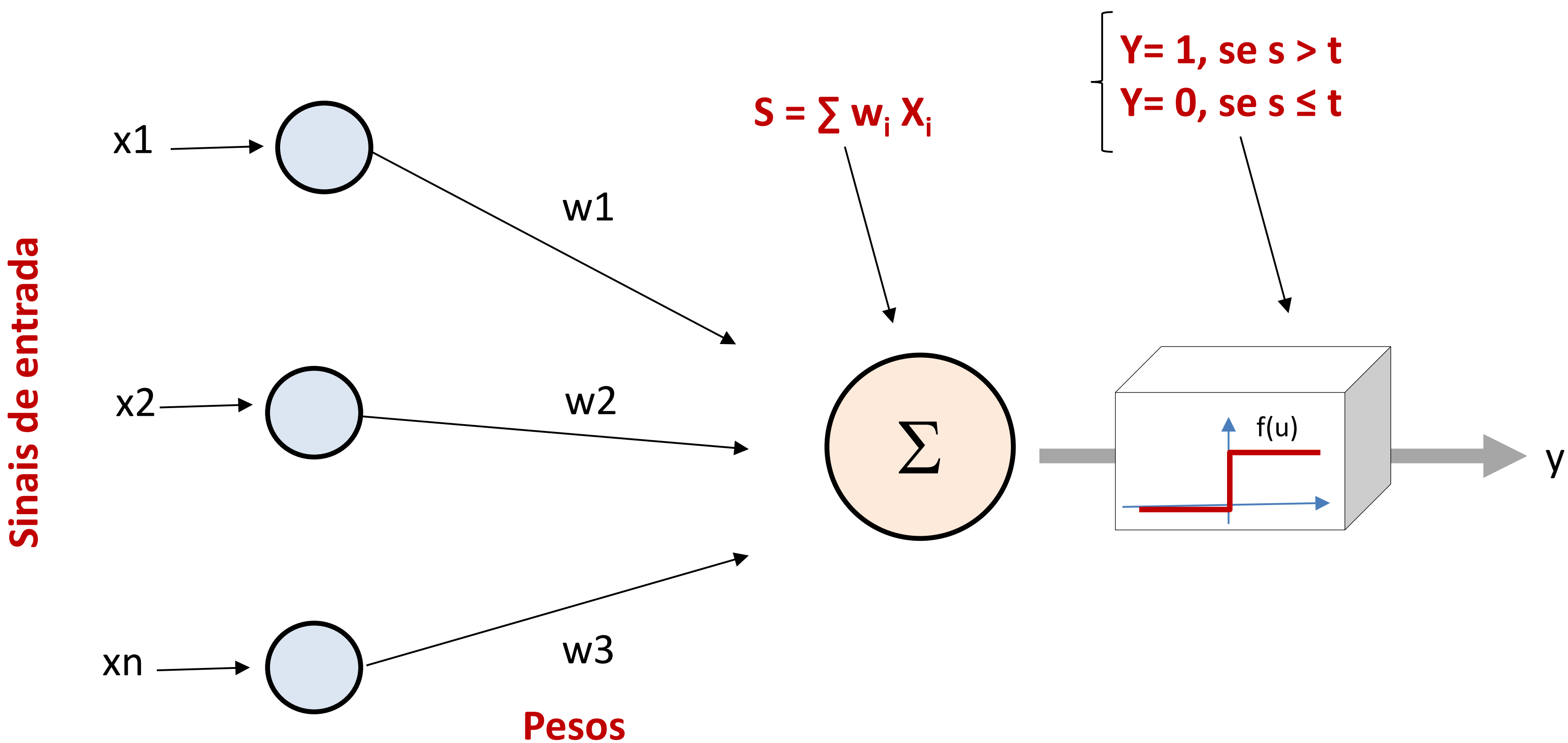


$$G(u) = \begin{cases} 1 & \text{Se } (u \geq 0) \\ -1 & \text{Se } (u < 0) \end{cases}$$

$$G(u) = \begin{cases} 1 & \text{Se } (u > 0) \\ 0 & \text{Se } (u = 0) \\ -1 & \text{Se } (u < 0) \end{cases}$$

Introdução

- Características
  - Operação de uma unidade de processamento (McCullock & Pitts, 1943):



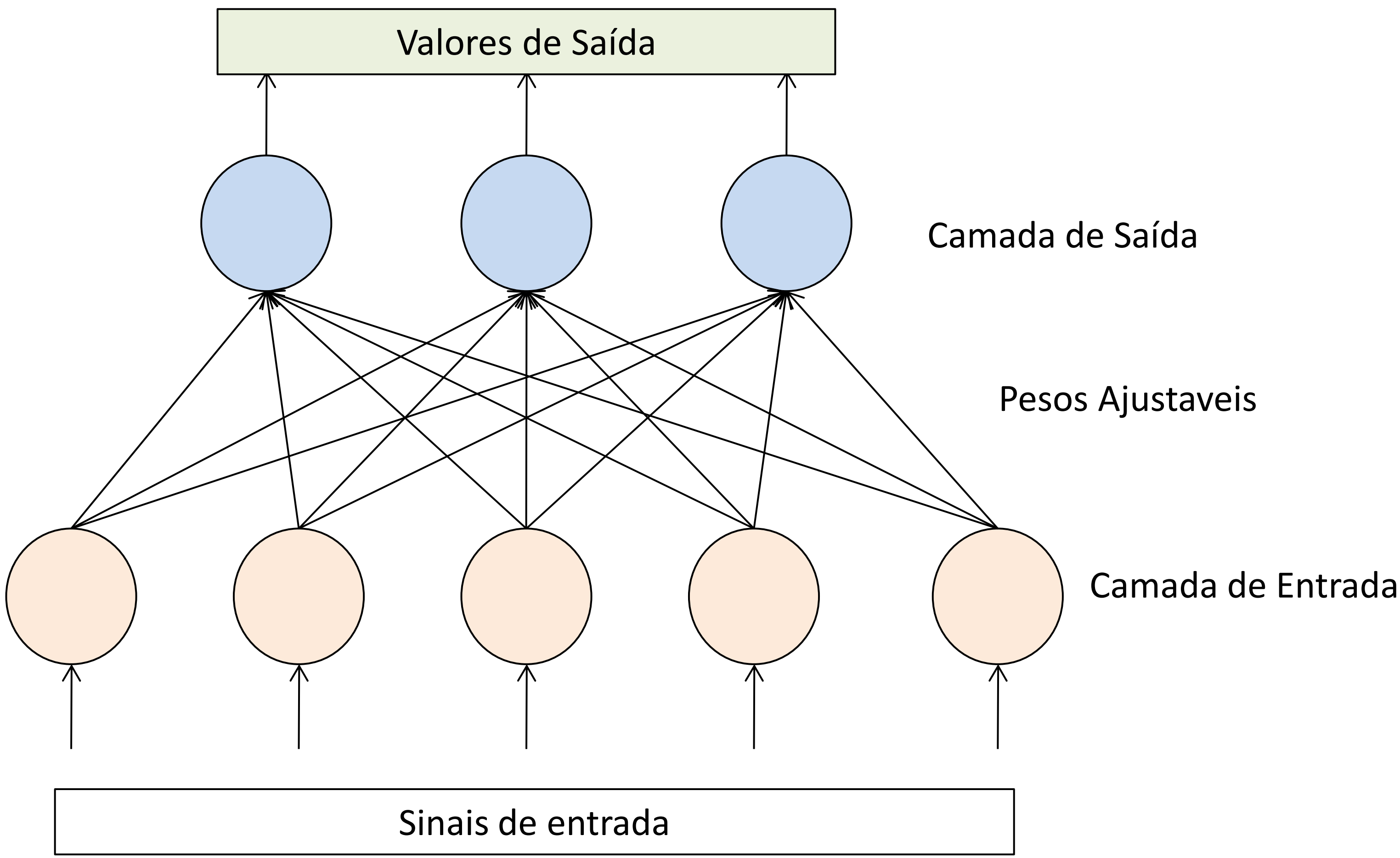
Introdução

- Formas mais básicas de **aprendizado** em Redes Neurais:
  - **Perceptron**: Algoritmo para aprendizagem de redes neurais simples (uma camada) desenvolvido nos anos 50.
  - **Backpropagation**: Algoritmo mais complexo para aprendizagem de redes neurais de múltiplas camadas desenvolvido nos anos 80.

Aprendizagem de Perceptron

- Usa-se um conjunto de **exemplos de treinamento** que dão a saída desejada para uma unidade, dado um conjunto de entradas.
- O objetivo é **aprender pesos** sinápticos de tal forma que a unidade de saída produza a saída correta pra cada exemplo.
- O algoritmo faz atualizações iterativamente até chegar aos **pesos corretos**.

Rede de Perceptrons

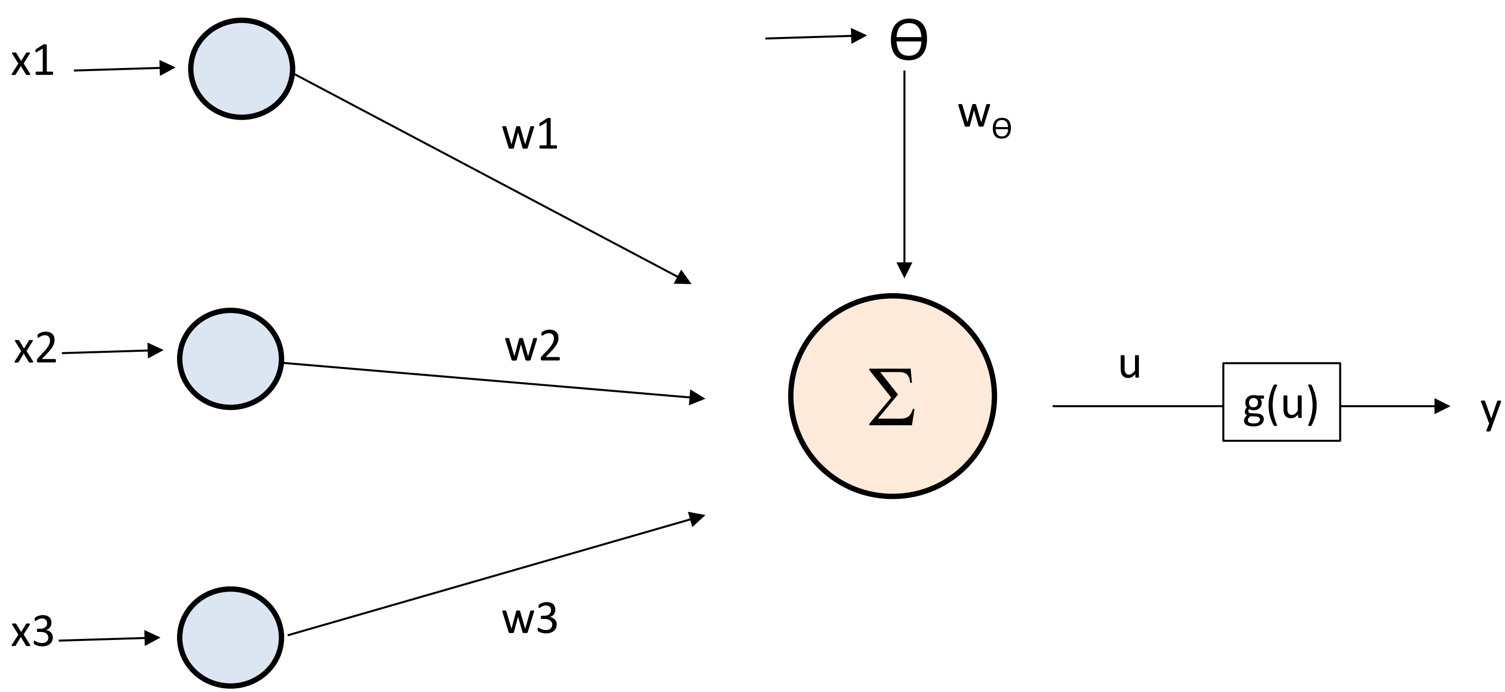


Perceptron

- **Estrutura da Rede**

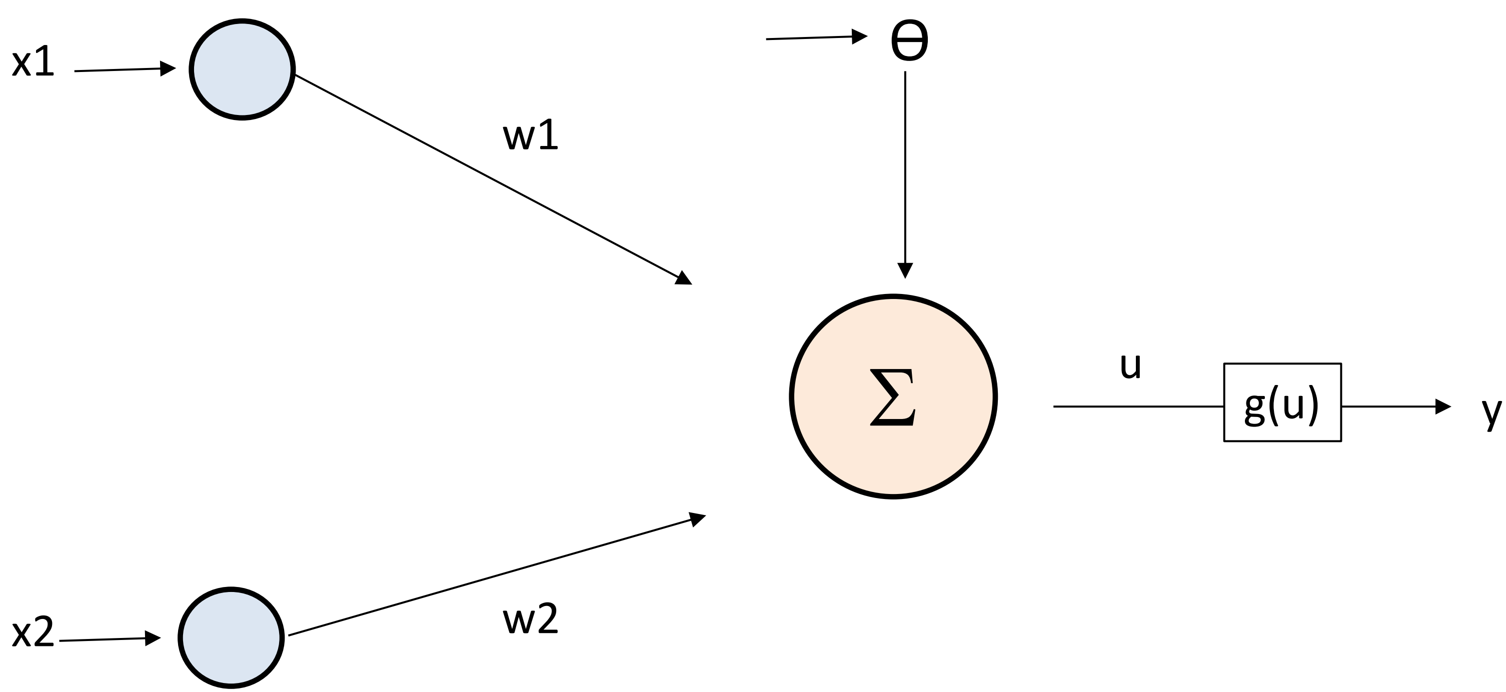
- As entradas normalmente são normalizadas

- Θ



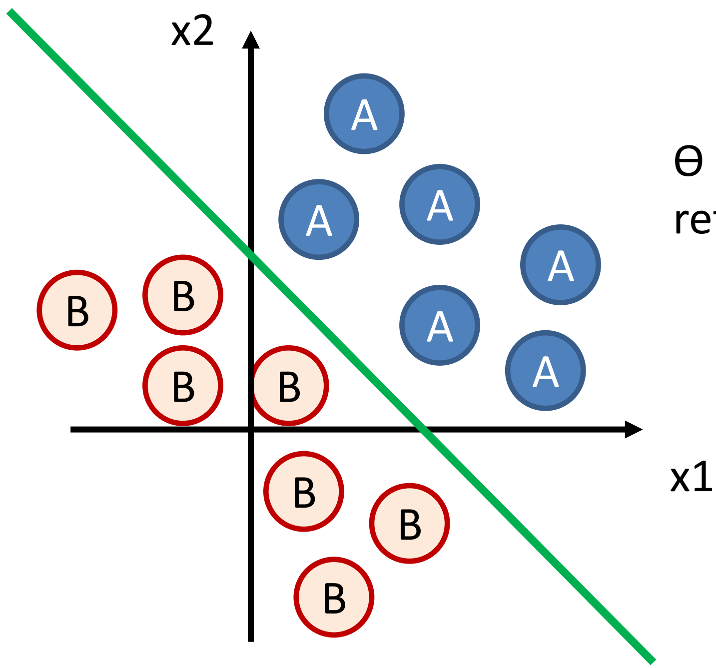
Perceptron

- **Estrutura da Rede** (duas entradas)



Perceptron

- **Classifica (linearmente separável)**
  – Apenas acha a reta que separa as classes

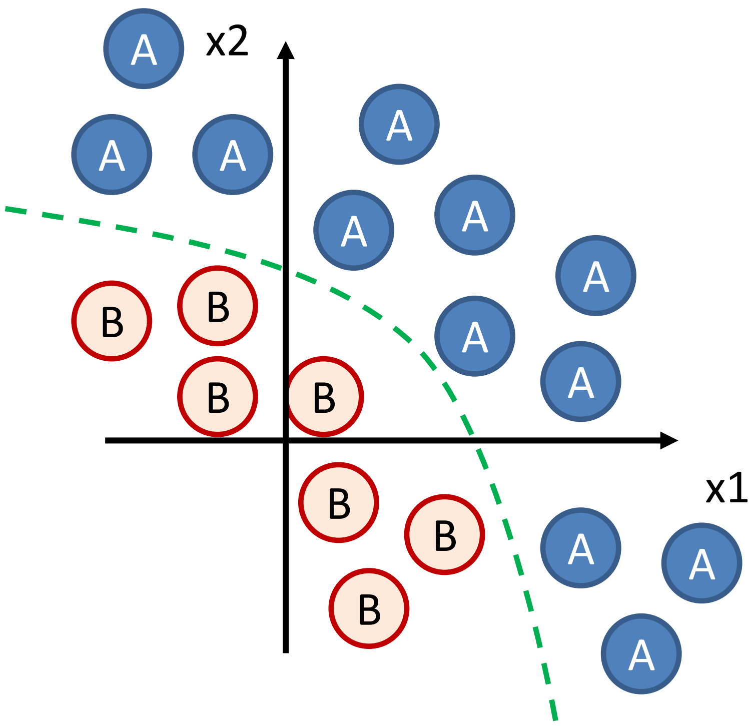


Θ = bias. Se for igual a 0 a reta passa pelo ponto (0,0)

$w_1 * x_1 + w_2 * x_2 - \Theta = 0$

Perceptron

- **Não classifica (não linearmente separável)**

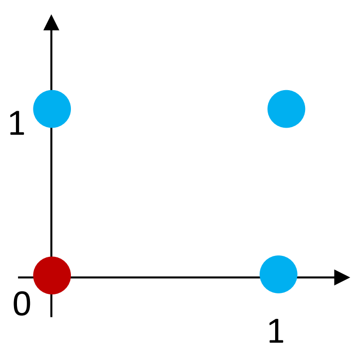


Perceptron

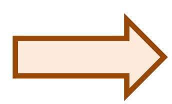
- **Exemplos de classificação**

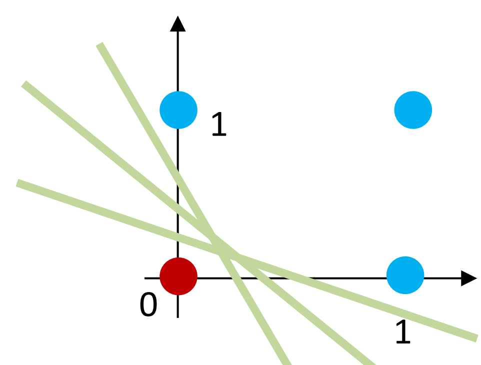
  - OR > linearmente separável

Sinal (x)	Sinal 2 (y)	Saída (x OR Y)
0	0	0 (vermelho)
0	1	1 (azul)
1	0	1 (azul)
1	1	1 (azul)



Representação entrada e saídas





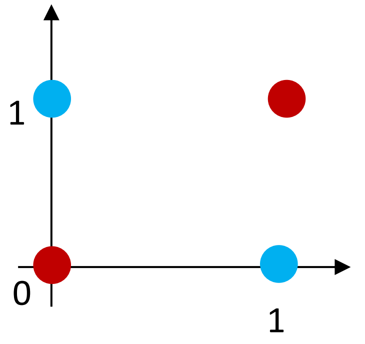
Representação entrada e saídas
(possíveis retas (soluções)

Perceptron

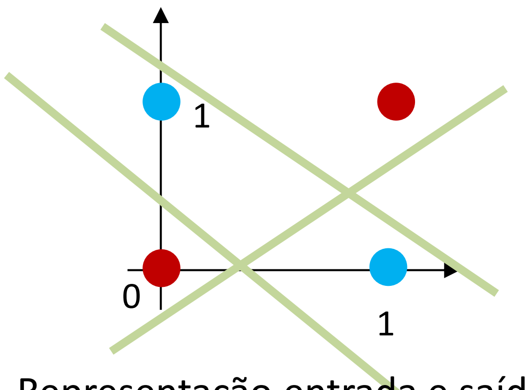
- **Exemplos de classificação**

  - XOR > não linearmente separável

Sinal (x)	Sinal 2 (y)	Saída (x XOR Y)
0	0	0 (vermelho)
0	1	1 (azul)
1	0	1 (azul)
1	1	0 (vermelho)



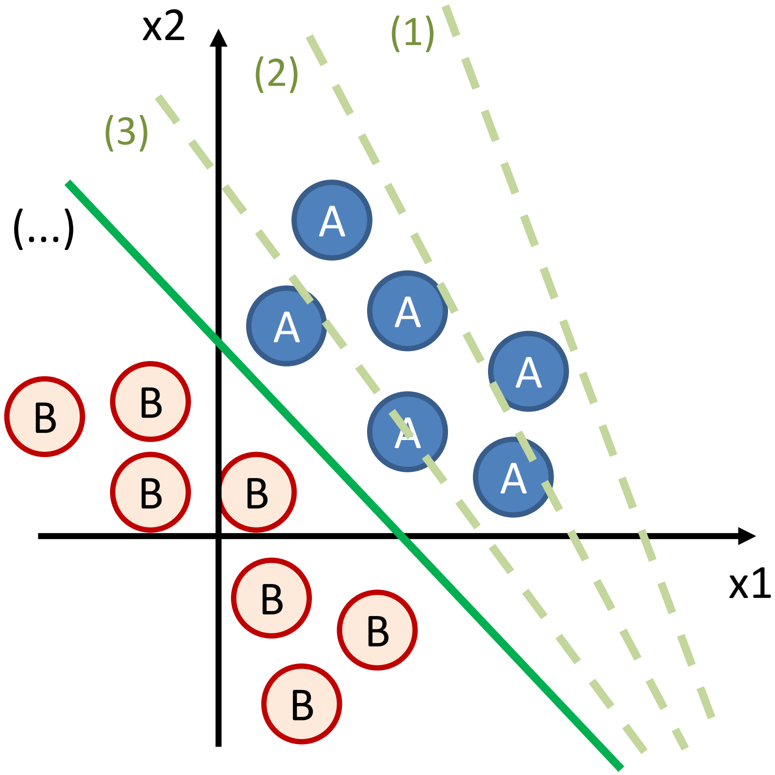
Representação entrada e saídas



Representação entrada e saídas
(impossível separar linearmente

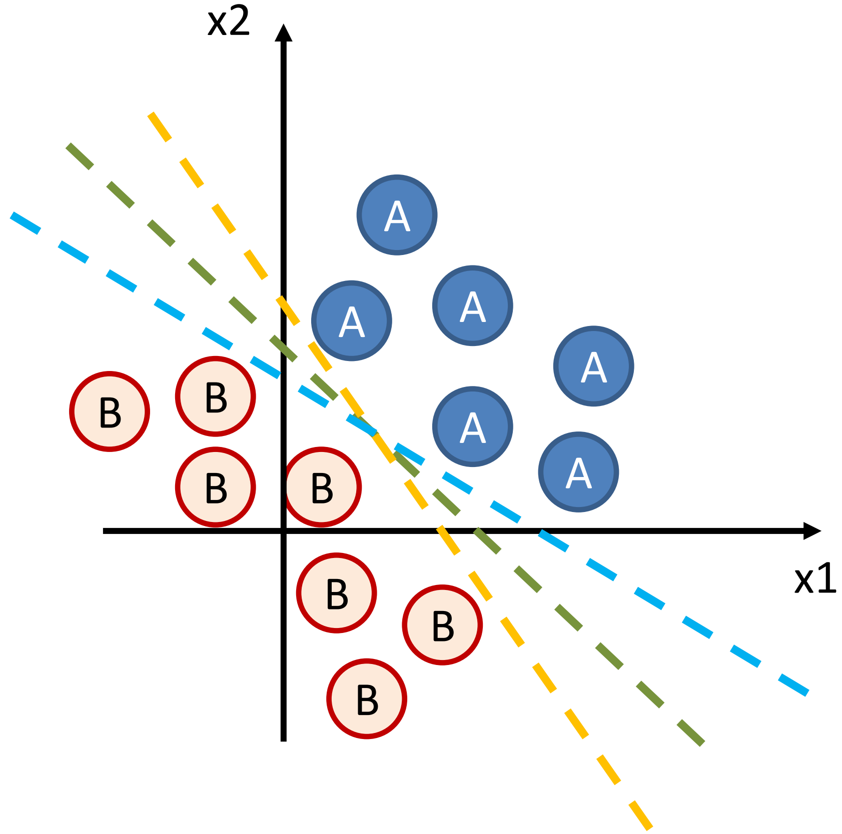
Perceptron

- **Processo de treinamento**



Perceptron

- **Processo de treinamento**



Perceptron

- Algoritmo de Treinamento (Supervisionado)

iniciar todas as conexões com $w_j = 0$;

repita

    para cada padrão de treinamento (x, d)

    faça

        calcular a saída o

        se ($d \neq o$)

      então ajustar pesos

até o erro ser aceitável

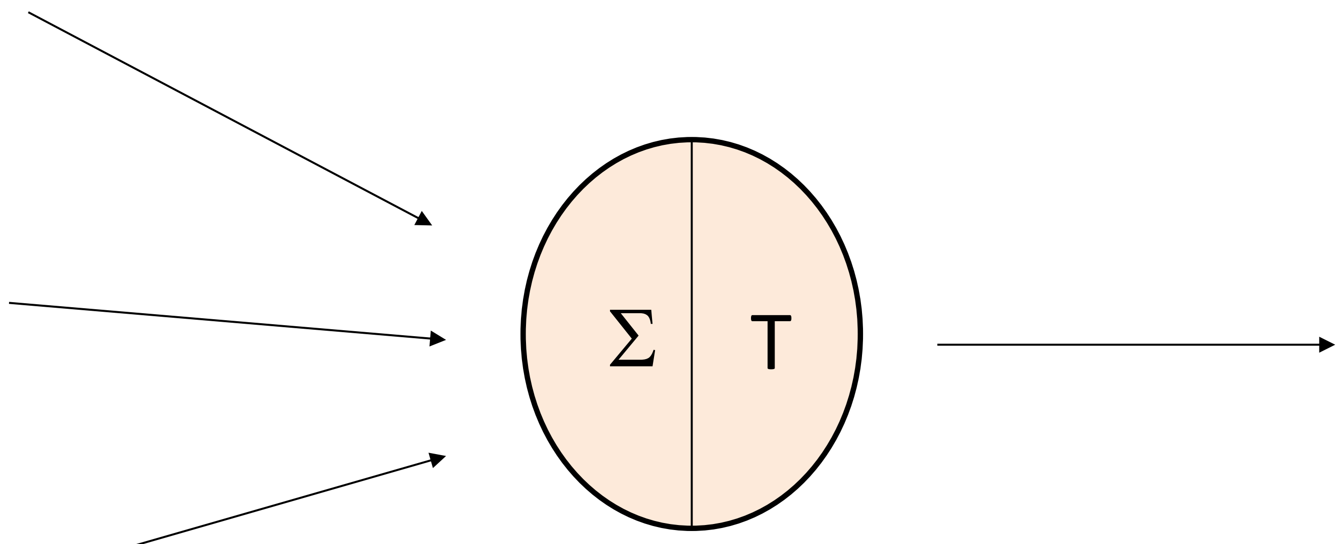
Exemplo de treino

Treinamento:

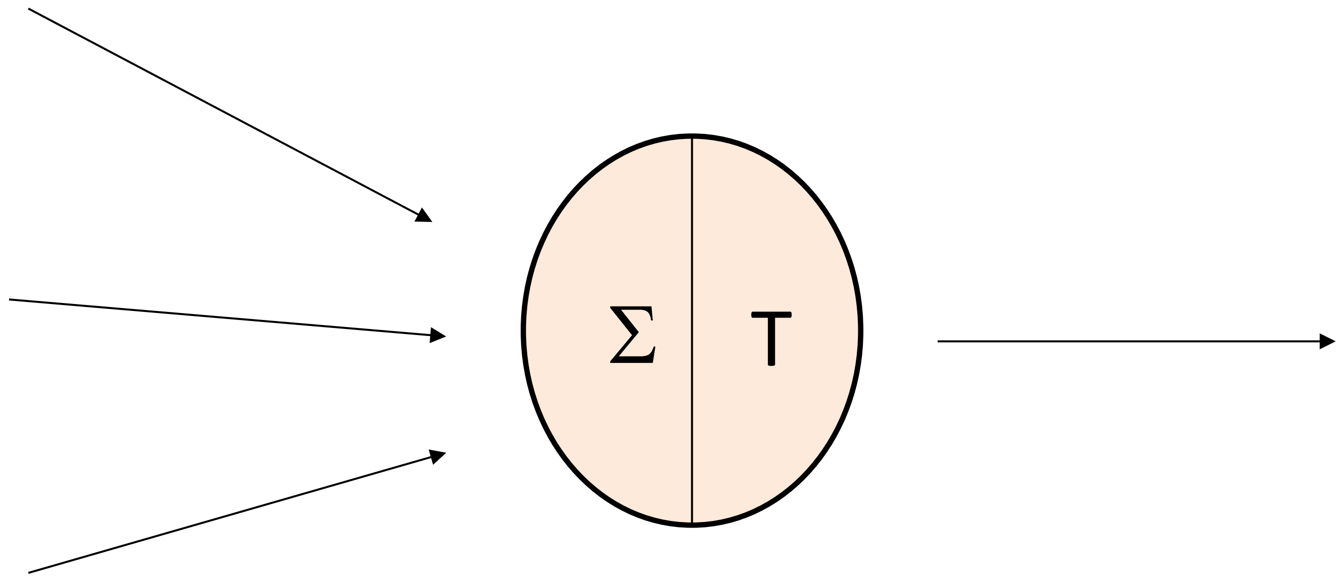
Atleta	**Ciência**	**Física**	**Filosofia**	**Classe**
Neymar	1	0	0	0
Messi	1	0	1	0
Barichello	1	1	0	1
Massa	1	1	1	1

Arquitetura

Perceptron 1 camada

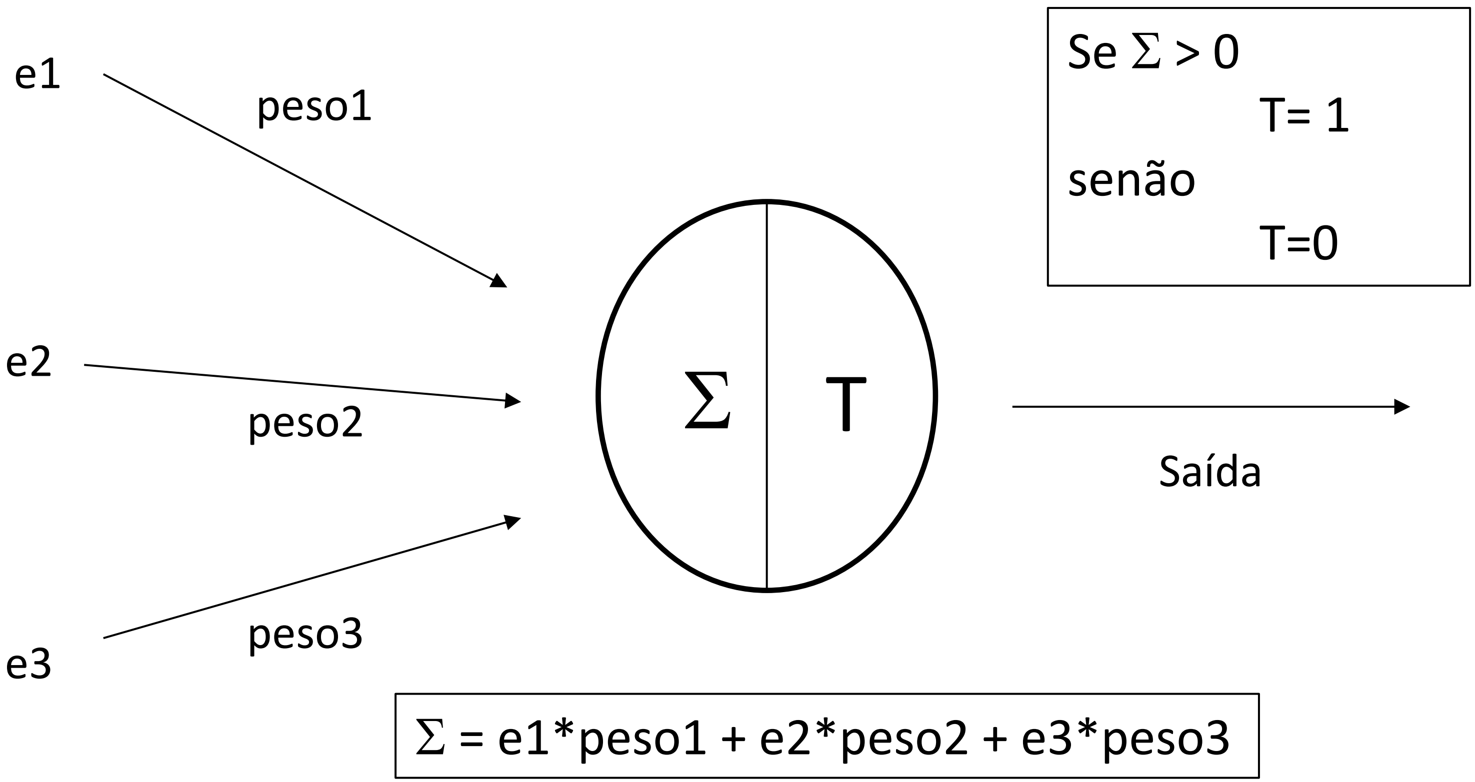


Função soma



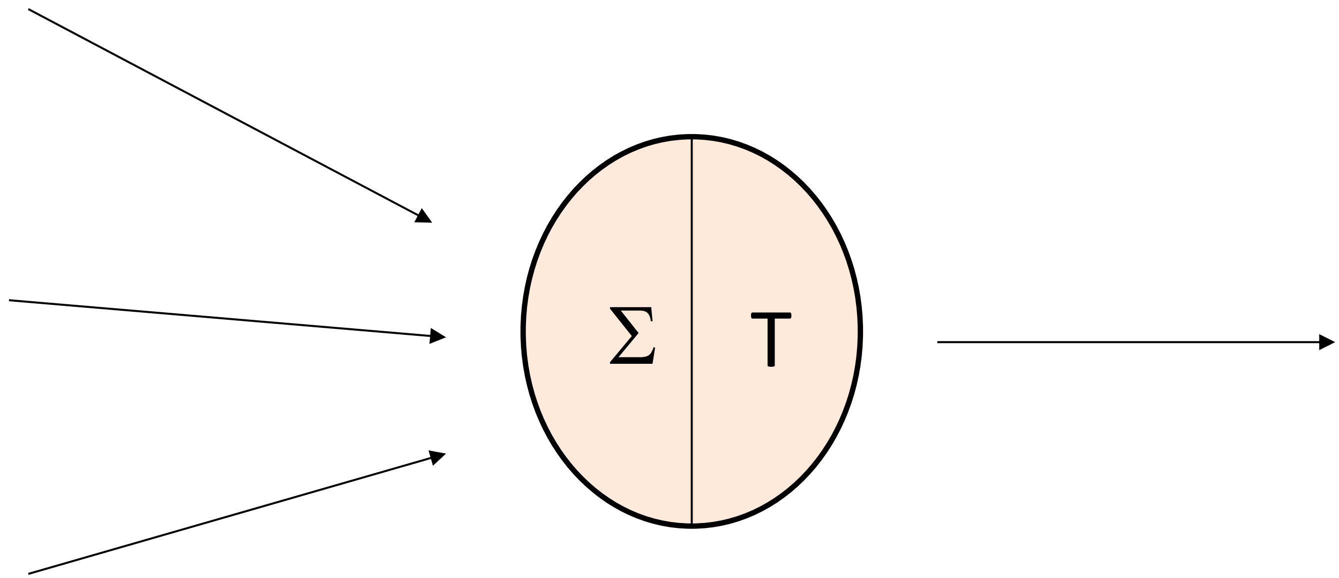
Σ = e1*peso1 + e2*peso2 + e3*peso3

Função de Ativação



Regra de aprendizado

Se a saída estiver errada e for 0
Adicionar a cada peso os sinais de entrada relativas a elas



Se a saída estiver errada e for 1
Subtrair de cada peso os sinais de entrada relativas a elas

Algoritmo

1º passo: iniciar pesos

2º passo: aplicar um padrão e calcular a soma ponderada

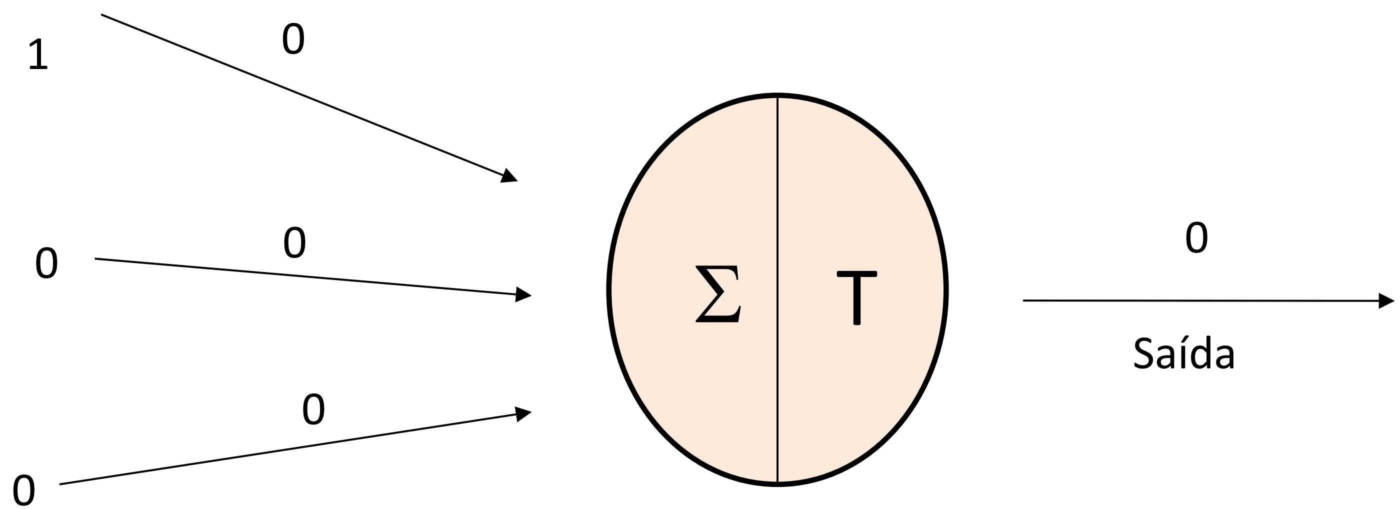
3º passo: passar a soma para a função de transferência

a) Se a saída estiver correta, voltar ao 2º passo
b) Se a saída estiver errada e for 0, adicionar a cada peso os sinais de entrada relativos a elas
c) Se a saída estiver errada e for 1, subtrair de cada peso os sinais de entrada relativos a elas

4º passo: voltar ao 2º passo

Treinamento

		Saída esperada
100	Neymar	0

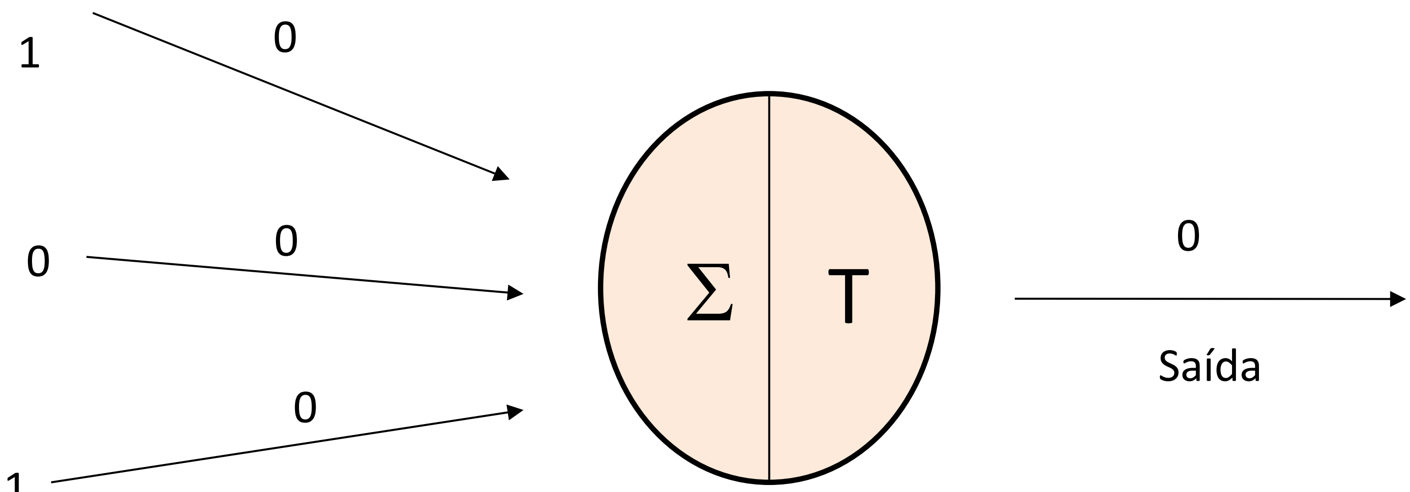


Σ = e1*peso1 + e2*peso2 + e3*peso3

Σ = 1*0 + 0*0 + 0*0

Treinamento

		Saída esperada
101	Messi	0

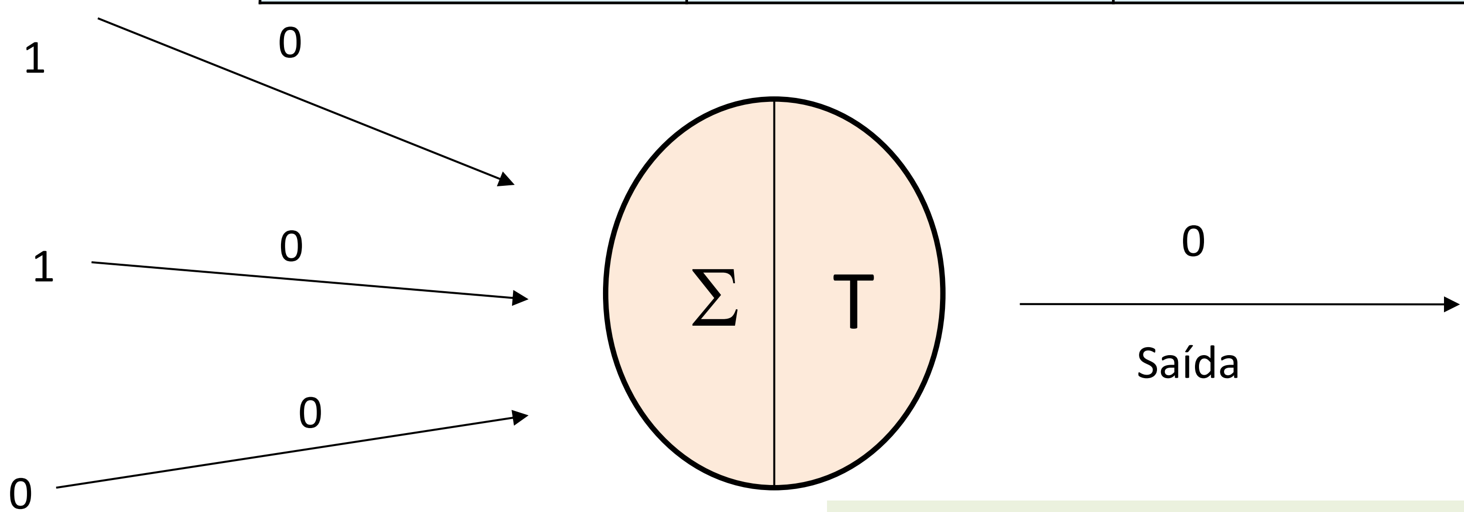


$\Sigma = e1*peso1 + e2*peso2 + e3*peso3$

$\Sigma = 1*0 + 0*0 + 0*1$

Treinamento

		Saída esperada
110	Barichello	1



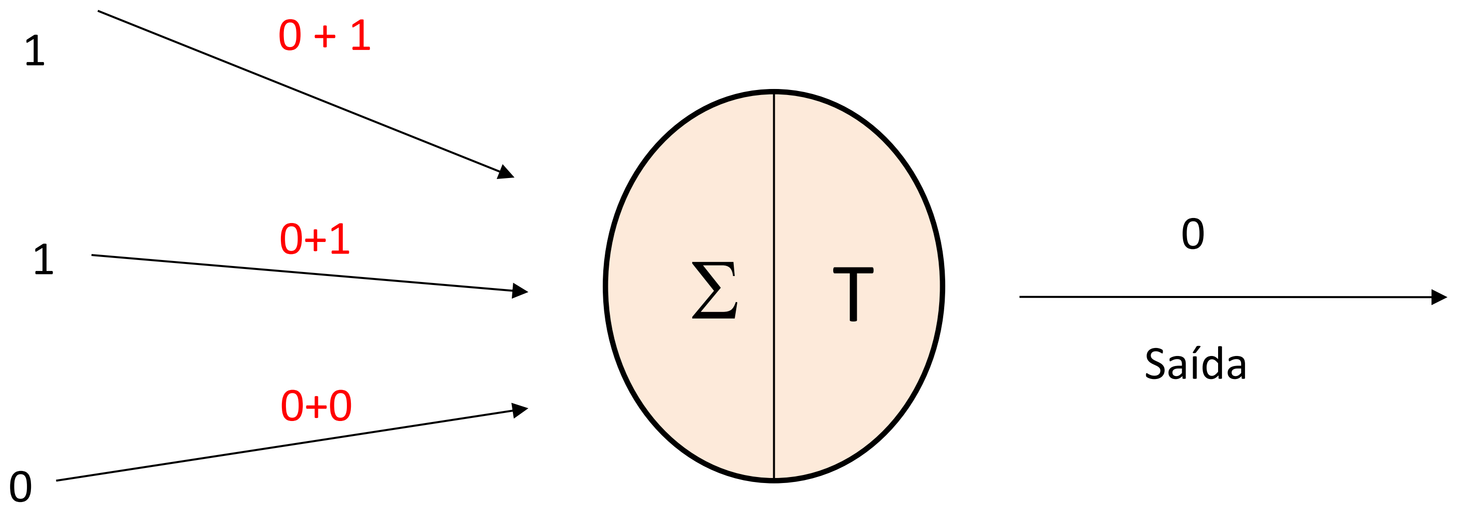
Σ = e1*peso1 + e2*peso2 + e3*peso3

Σ = 1*0 + 1*0 + 0*0

a) Se a saída estiver errada e for 0, adicionar a cada peso os sinais de entrada relativos a elas

Treinamento

		Saída esperada
110	Barichello	1



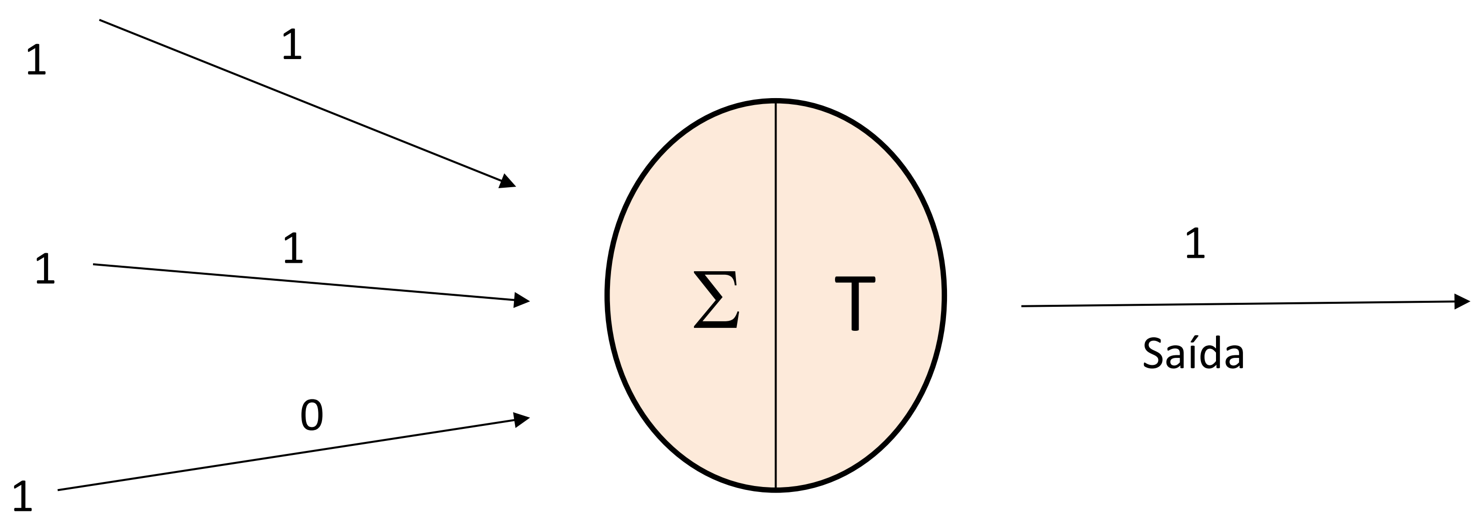
Σ = e1*peso1 + e2*peso2 + e3*peso3

Σ = 1*0 + 1*0 + 0*0

a) Se a saída estiver errada e for 0, adicionar a cada peso os sinais de entrada relativos a elas

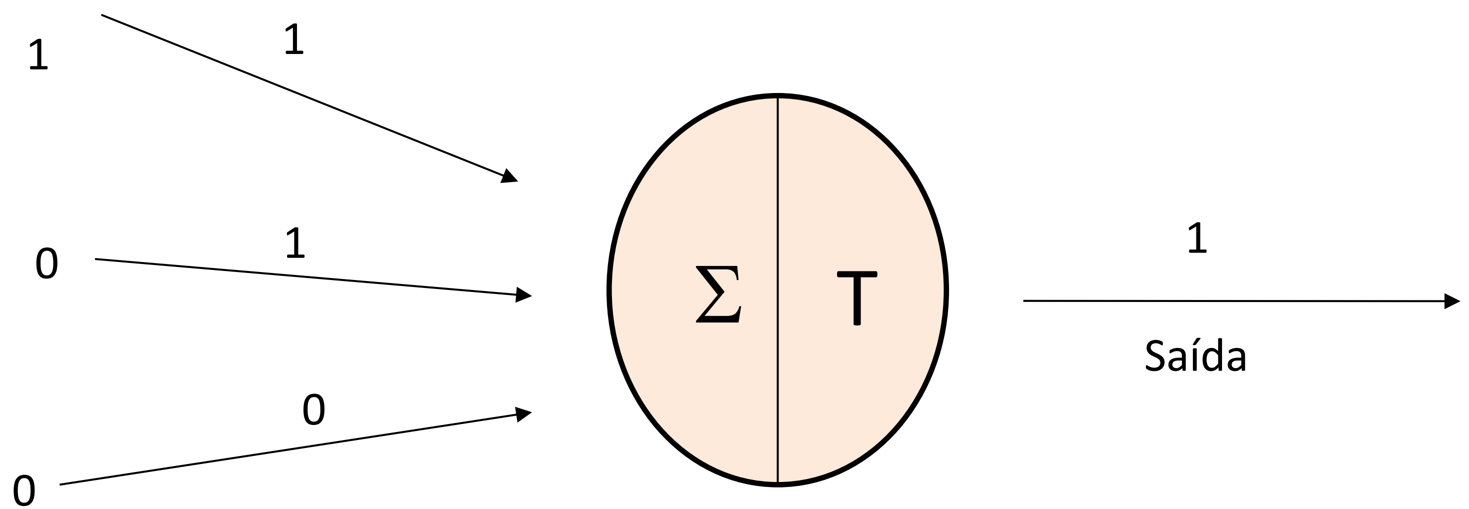
Treinamento

		Saída esperada
111	Massa	1



Treinamento

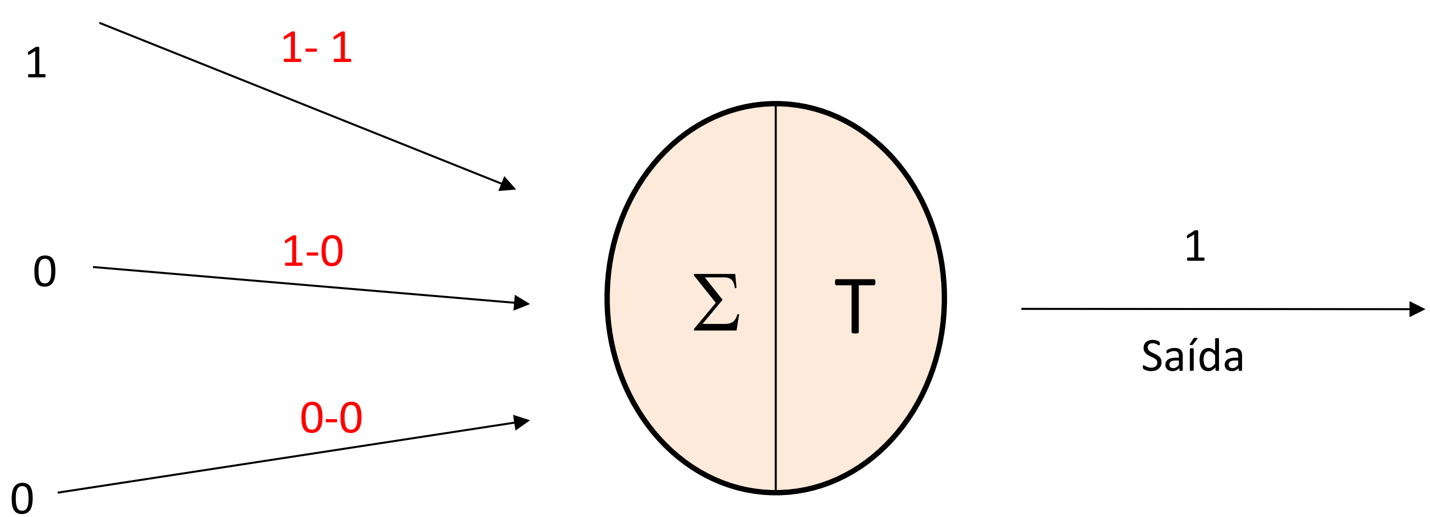
		Saída esperada
100	Neymar	0



b) Se a saída estiver errada e for 1, subtrair de cada peso os sinais de entrada relativos a elas

Treinamento

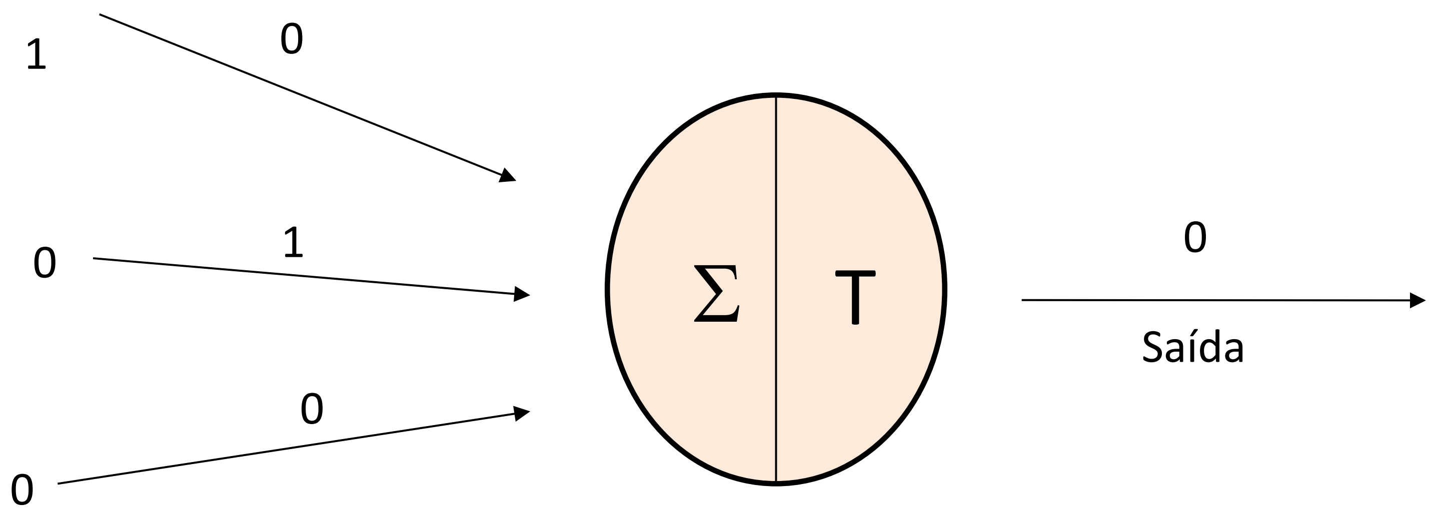
		Saída esperada
100	Neymar	0



b) Se a saída estiver errada e for 1, subtrair de cada peso os sinais de entrada relativos a elas

Treinamento

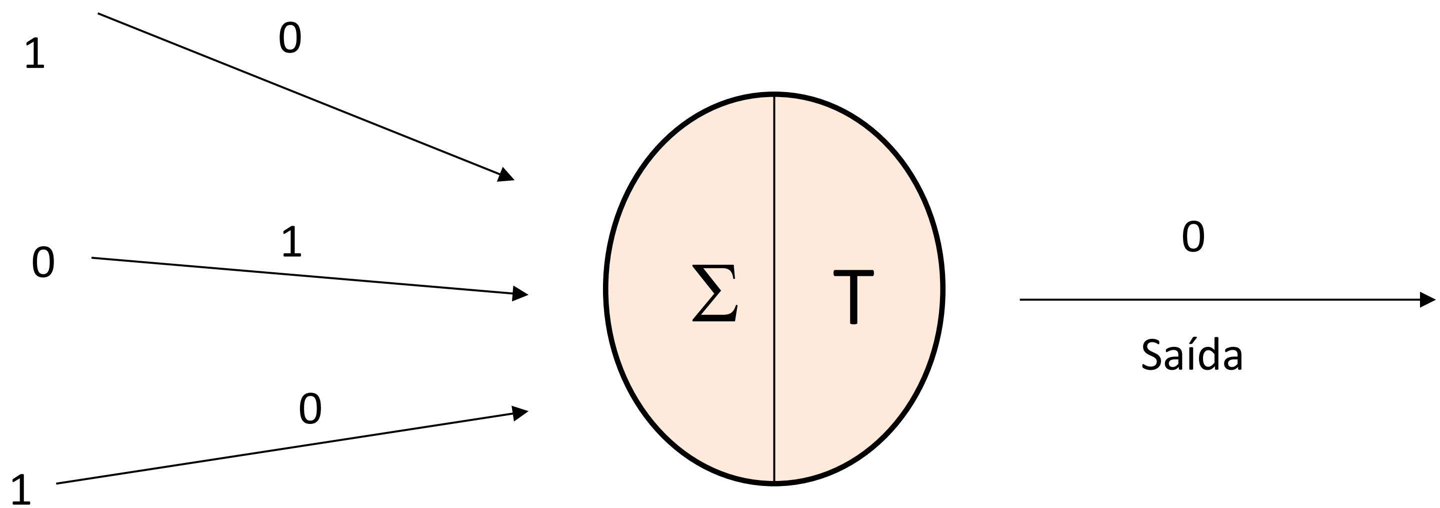
		Saída esperada
100	Neymar	0



b) Se a saída estiver errada e for 1, subtrair de cada peso os sinais de entrada relativos a elas

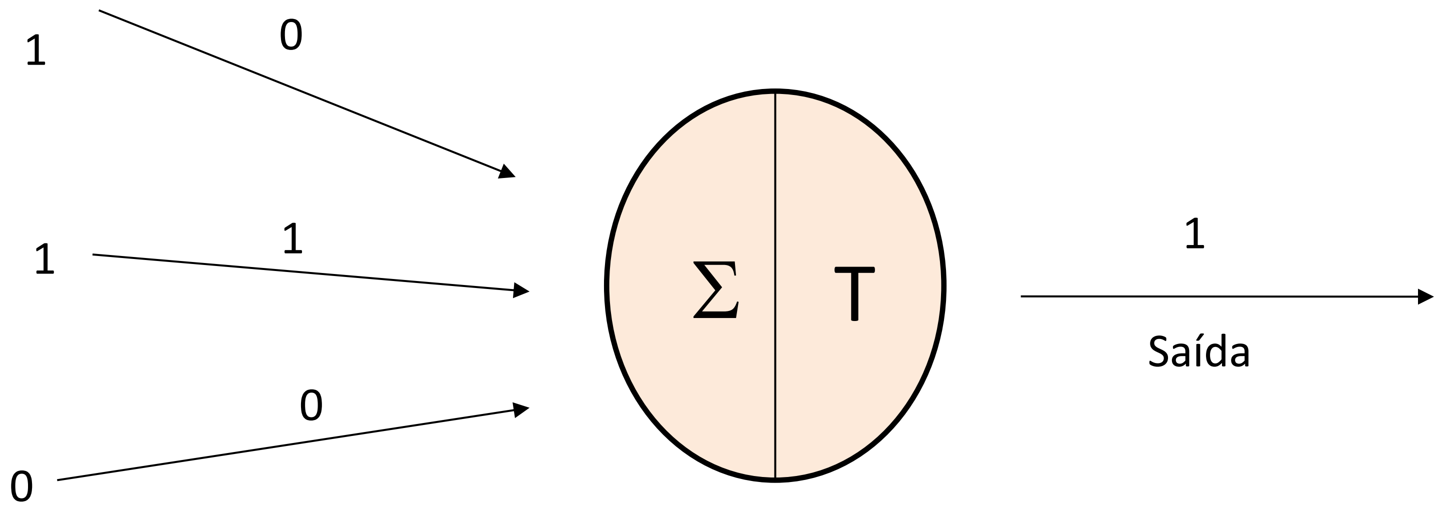
Treinamento

		Saída esperada
101	Messi	0



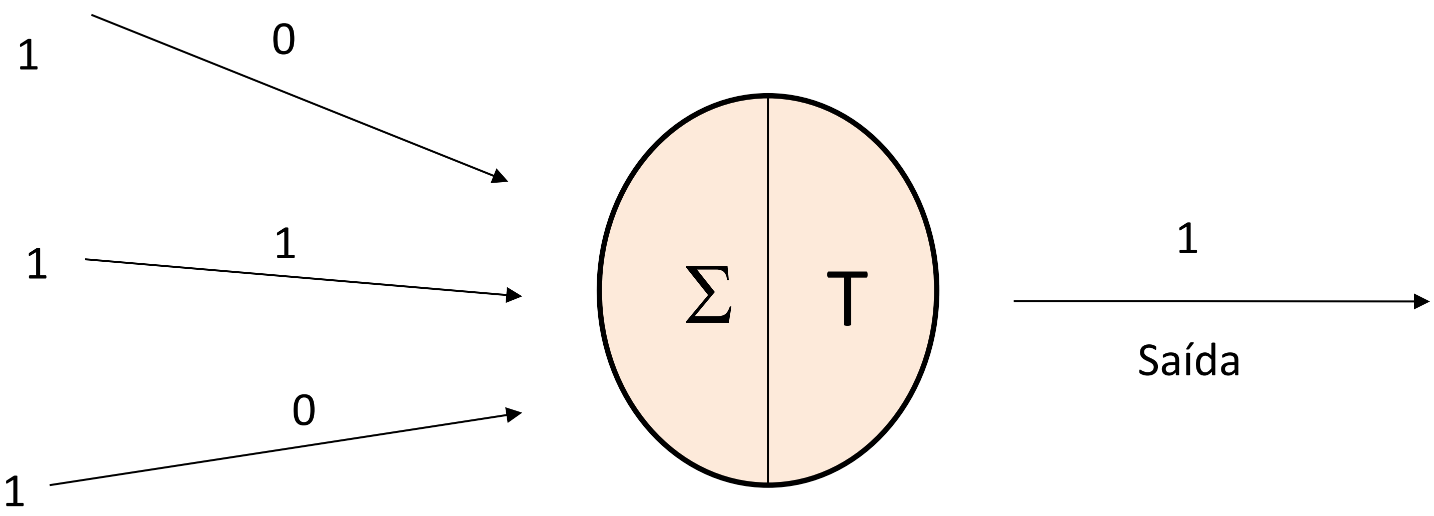
Treinamento

		Saída esperada
110	Barichello	1



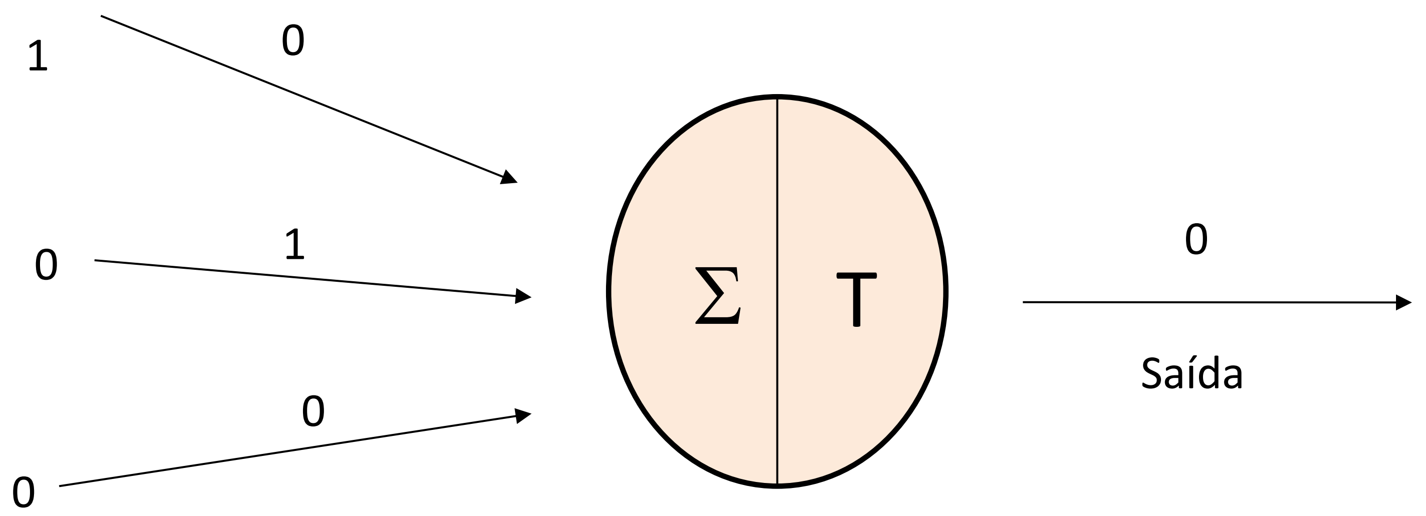
Treinamento

		Saída esperada
111	Massa	1



Treinamento

		Saída esperada
100	Neymar	0



Exemplo de classificação

Após treinamento:

Atleta	**Ciência**	**Física**	**Filosofia**	**Classe**
Jogador 1	0	0	0	0
Jogador 2	0	0	1	0
Piloto 1	0	1	1	1
Piloto 2	0	1	0	1

Aprendizado de Perceptrons

- Para que um perceptron possa **aprender uma função** deve-se mudar o valor dos pesos ajustáveis por uma quantidade proporcional a **diferença entre a saída desejada e atual saída** do sistema.

$$w_i = w_i + \Delta w_i$$

$$\Delta w_i = \eta(t - o)x_i$$

			Saída desejada:	*t*
x_1	x_2	...	x_n	*o*
x_1	x_2	...	x_n	*t*

- *t* = saída desejada.
- *o* = atual saída do perceptron.
- η = Learning rate.

Aprendizado de Perceptrons

- Regra de aprendizado:

$$w_i = w_i + \Delta w_i$$

$$\Delta w_i = \eta(t - o)x_i$$

- Se a saída do perceptron não estiver correta (*t* != *o*):
  - Os pesos w_i são alterados de forma que a saída do perceptron para os novos pesos seja próxima de *t*.

- O algoritmo vai convergir para a correta classificação se:
  - O conjunto de treinamento é linearmente separável.
  - η é suficientemente pequeno.

Aprendizado de Perceptrons

Vetor de entrada $\boxed{X = 0, 0, 0}$

Vetor de pesos $\boxed{W = 0, 0, 0}$

Bias >> $\boxed{\Theta = 1}$

Peso bias >> $\boxed{w_\Theta = 0}$

Função soma

$$u = \left(\sum_{i=0}^{n} X_i \,.\, W_i\right) + \Theta.\, W_\Theta$$

Função de transferência
(Função degrau)

$$y = g(u) = \begin{cases} 0\,,\ u < 0 \\ 1,\ u \geq 0 \end{cases}$$

Algoritmo de aprendizado
(Regra delta)

$$W(i)_{t+1} = w(i)_t + \eta \,.\, E_t.x(i)$$

Cálculo do erro >> $\boxed{E_t = y_d - y}$

$\boxed{\eta = 1}$

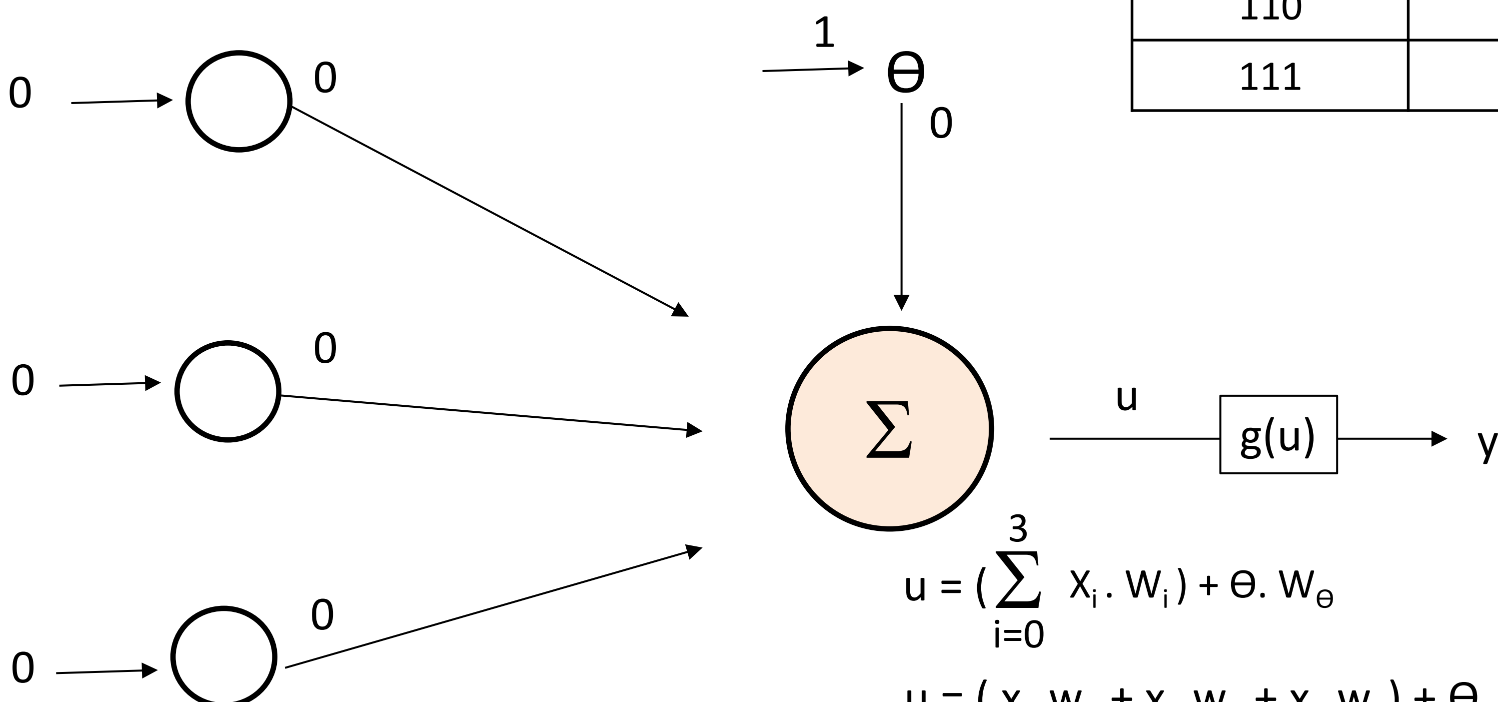
	X
X1	0
X2	0
X3	0

	W
W1	0
W2	0
W3	0

Exemplo
000

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1

$\Theta = 1 \qquad w_\Theta = 0$



$$u = \left(\sum_{i=0}^{3} X_i \,.\, W_i\right) + \Theta .\, W_\Theta$$

$$u = (\, x_1.w_1 + x_2.w_2 + x_3.w_3) + \Theta \,.\, w_\Theta$$

$$u = (0.0 + 0.0 + 0.0) + 1.0$$

$$u = 0 + 0$$

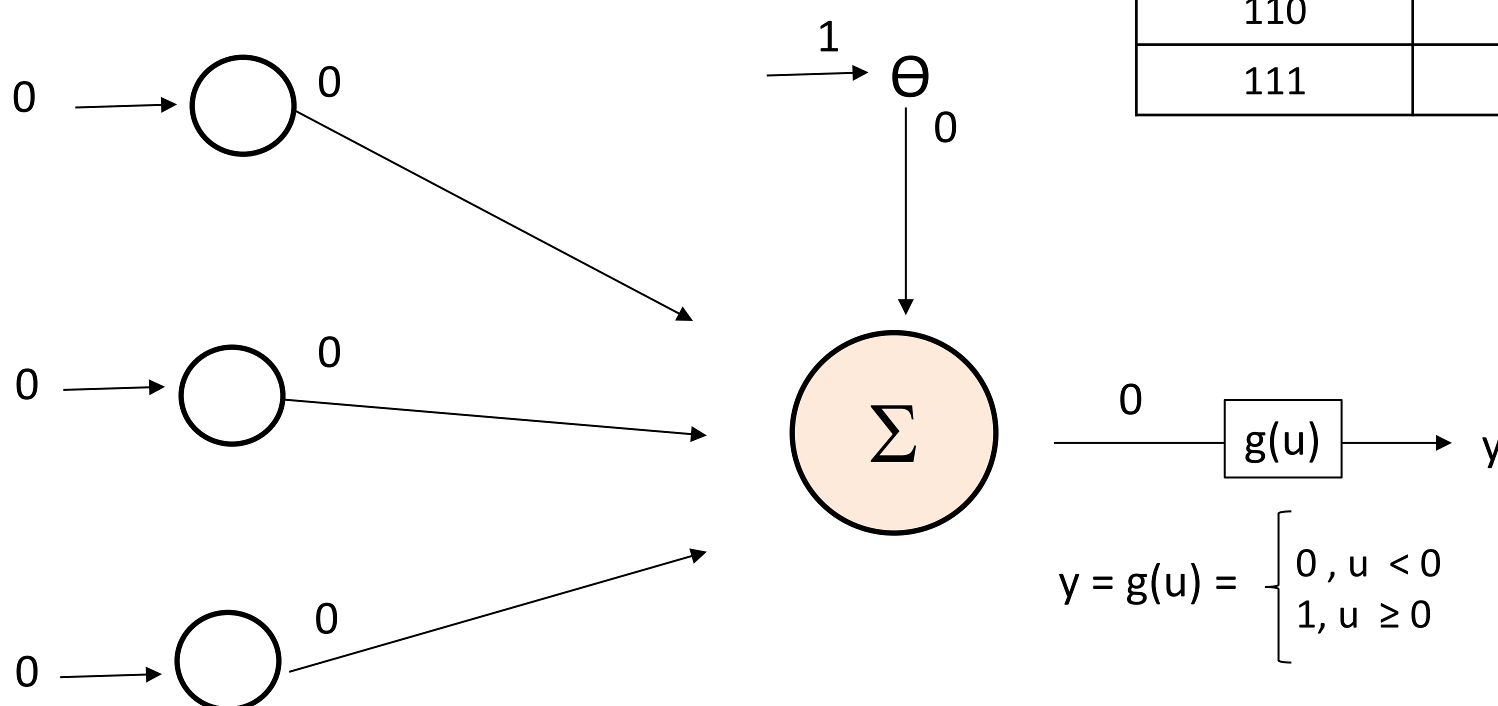
Exemplo
000

	X
X1	0
X2	0
X3	0

	W
W1	0
W2	0
W3	0

$\Theta = 1$ $w_{\Theta} = 0$

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



$$y = g(u) = \begin{cases} 0, & u < 0 \\ 1, & u \geq 0 \end{cases}$$

$$y = g(0) = 1$$

	X
X1	0
X2	0
X3	0

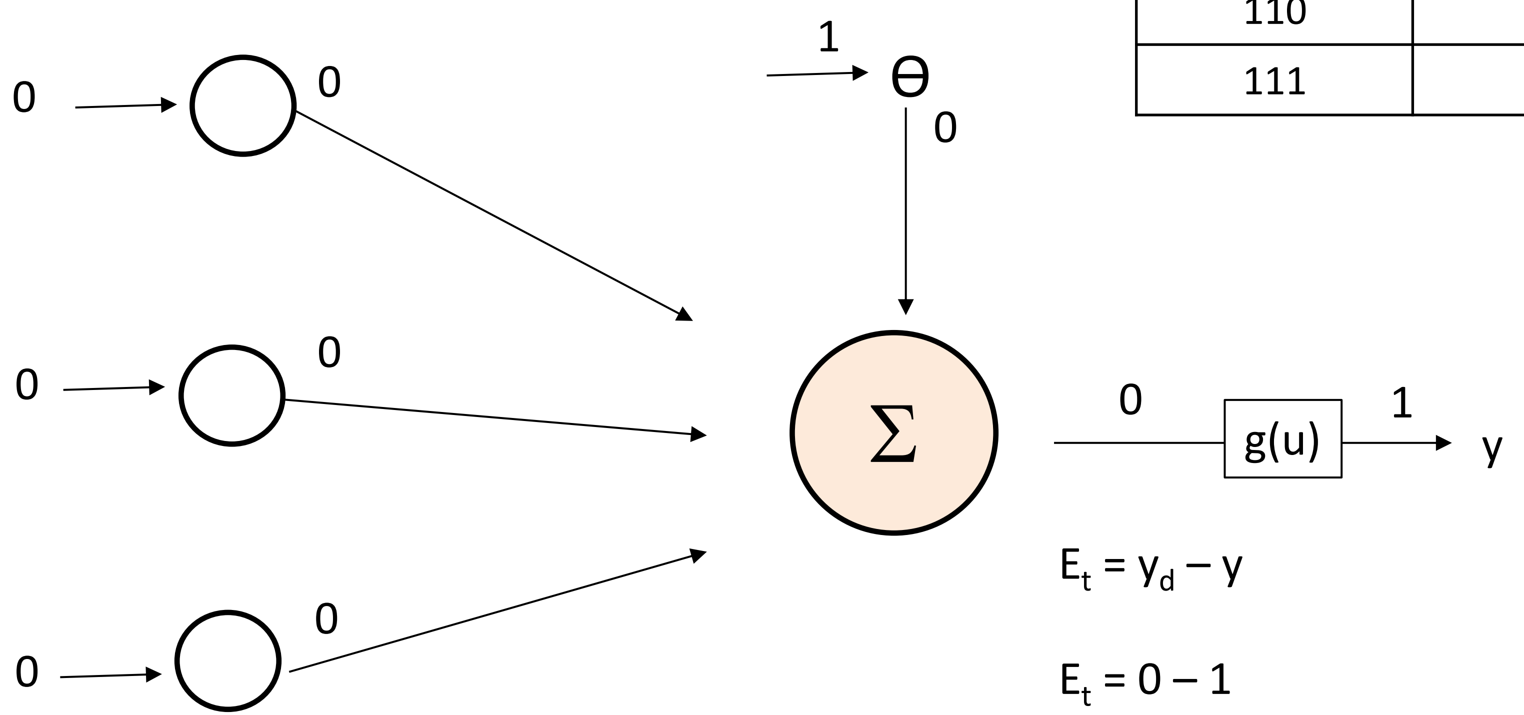
	W
W1	0
W2	0
W3	0

$\Theta = 1 \qquad w_\Theta = 0$

Exemplo

000

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



$E_t = y_d - y$

$E_t = 0 - 1$

$E_t = -1$

	X
X1	0
X2	0
X3	0

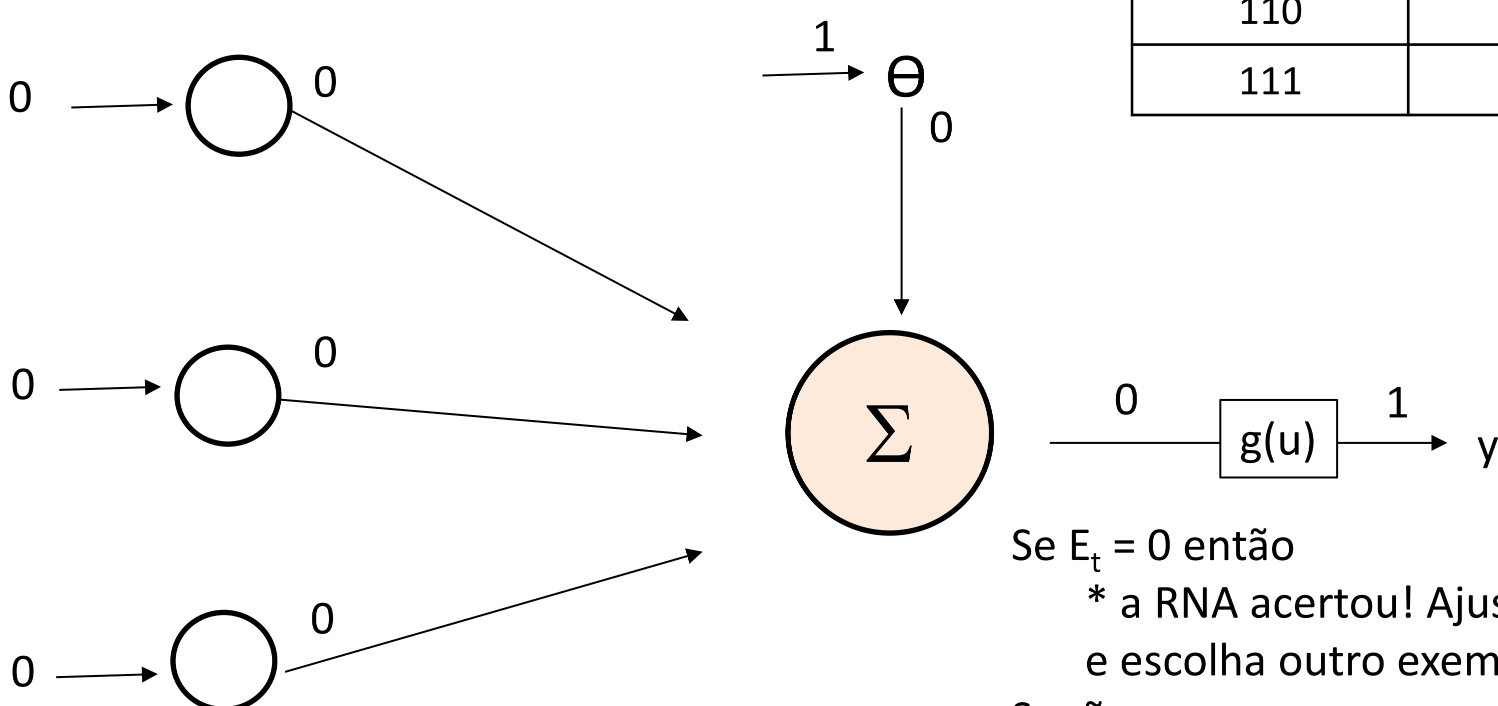
	W
W1	0
W2	0
W3	0

$\Theta = 1 \qquad w_{\Theta} = 0$

Exemplo

000

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



Se E_t = 0 então

* a RNA acertou! Ajuste os pesos e escolha outro exemplo

Senão

* Ajuste os pesos sinápticos e continue no exemplo até acertar

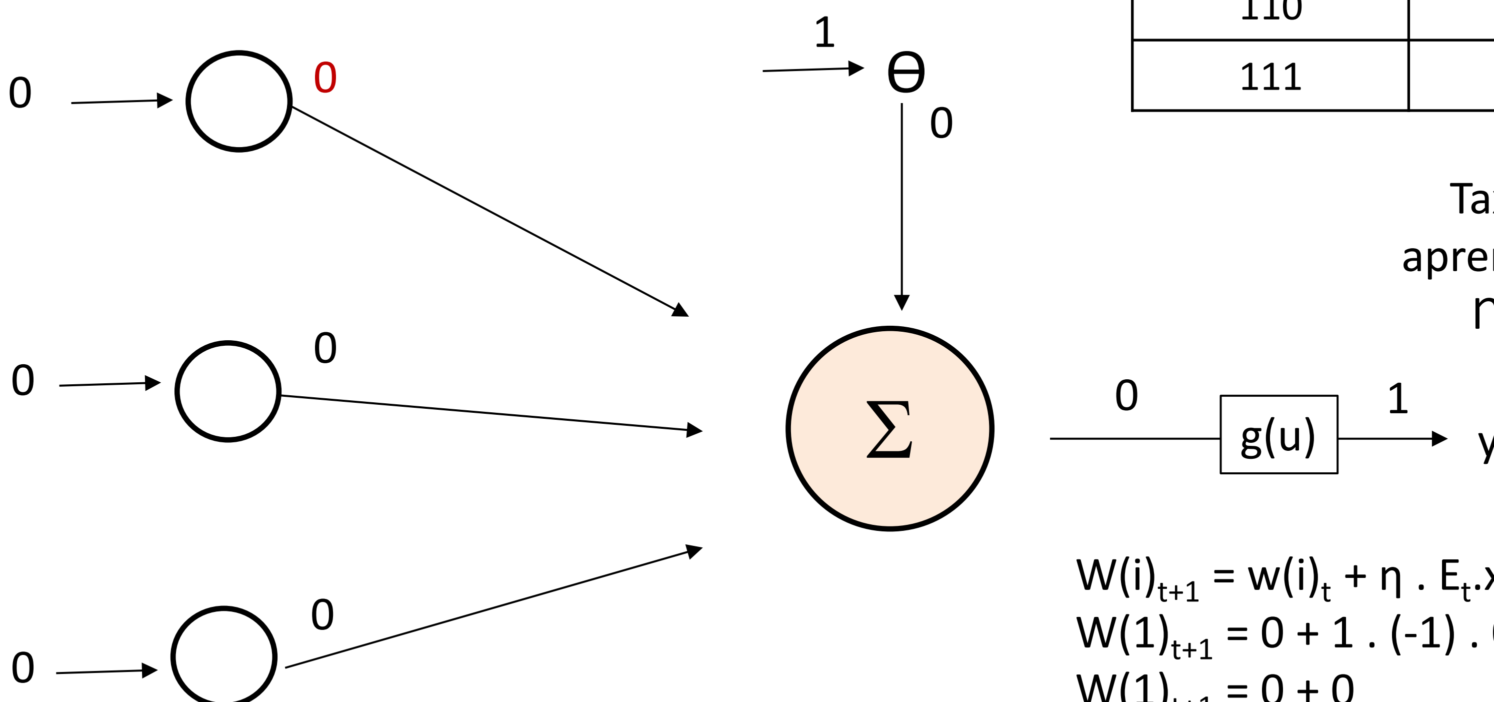
	X
X1	0
X2	0
X3	0

	W
W1	0
W2	0
W3	0

$\Theta = 1 \qquad w_{\Theta} = 0$

Exemplo
000

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



Taxa de aprendizado
$\eta = 1$

$W(i)_{t+1} = w(i)_t + \eta \,.\, E_t.x(i)$
$W(1)_{t+1} = 0 + 1 \,.\, (-1) \,.\, 0$
$W(1)_{t+1} = 0 + 0$

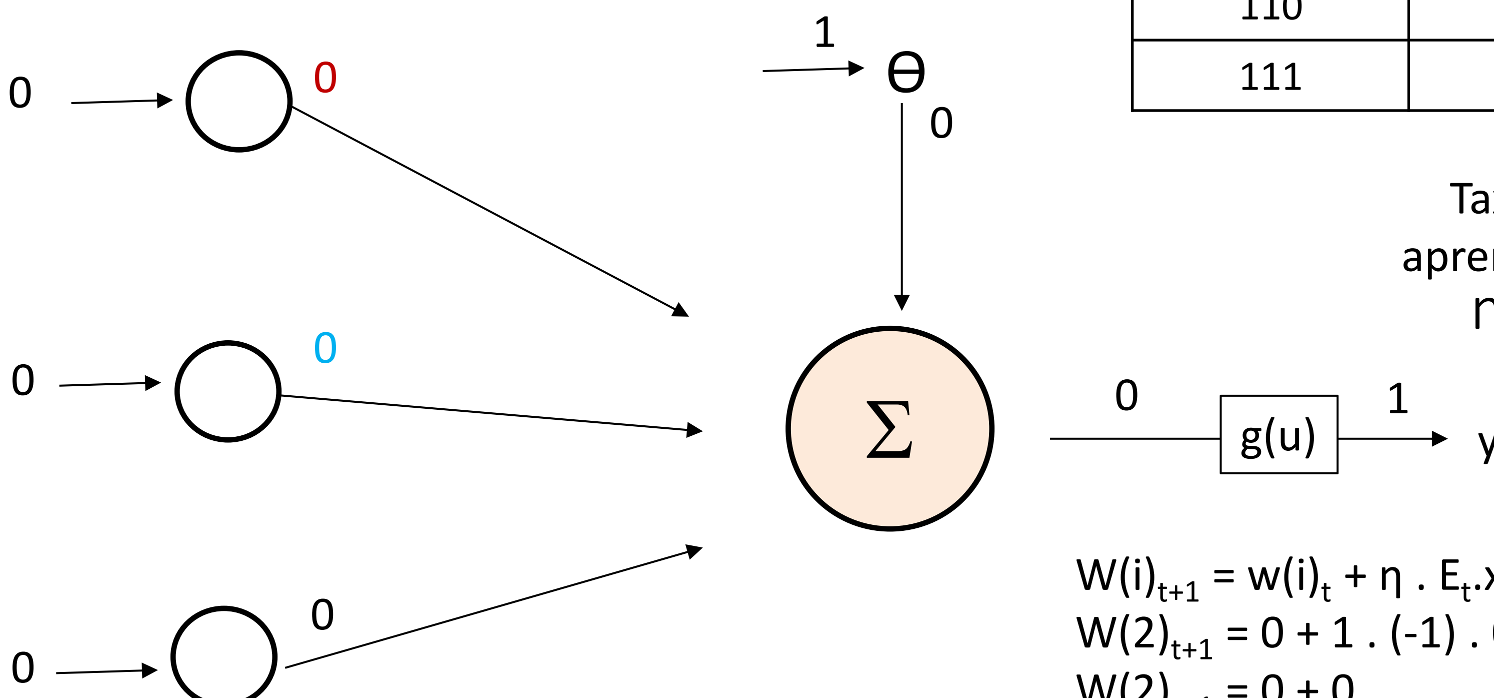
	X
X1	0
X2	0
X3	0

	W
W1	0
W2	0
W3	0

$\Theta = 1 \qquad w_{\Theta} = 0$

Exemplo
000

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



Taxa de aprendizado
$\eta = 1$

$W(i)_{t+1} = w(i)_t + \eta \ . \ E_t . x(i)$
$W(2)_{t+1} = 0 + 1 \ . \ (-1) \ . \ 0$
$W(2)_{t+1} = 0 + 0$

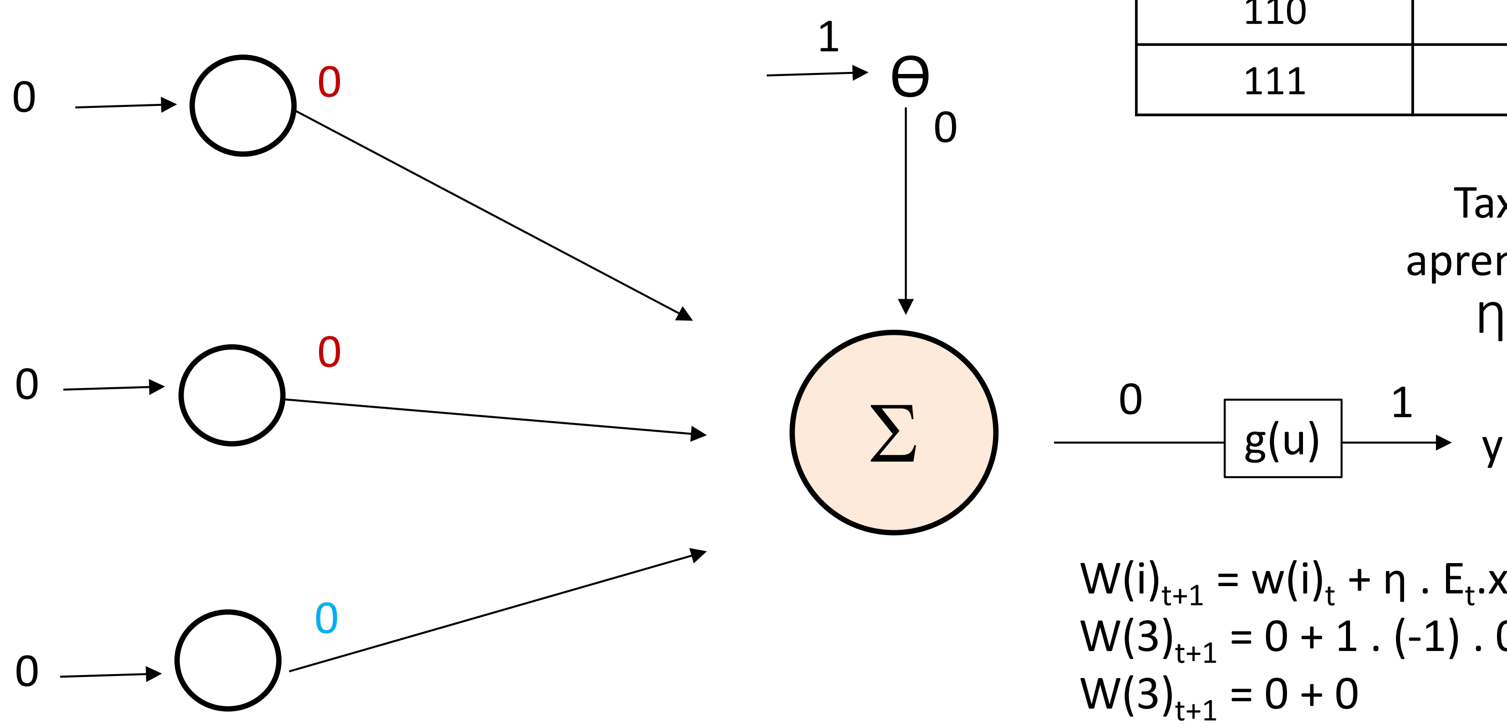
	X
X1	0
X2	0
X3	0

	W
W1	0
W2	0
W3	0

$\Theta = 1 \qquad w_{\Theta} = 0$

Exemplo
000

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



Taxa de aprendizado
$\eta = 1$

$$W(i)_{t+1} = w(i)_t + \eta \; . \; E_t . x(i)$$
$$W(3)_{t+1} = 0 + 1 \; . \; (-1) \; . \; 0$$
$$W(3)_{t+1} = 0 + 0$$

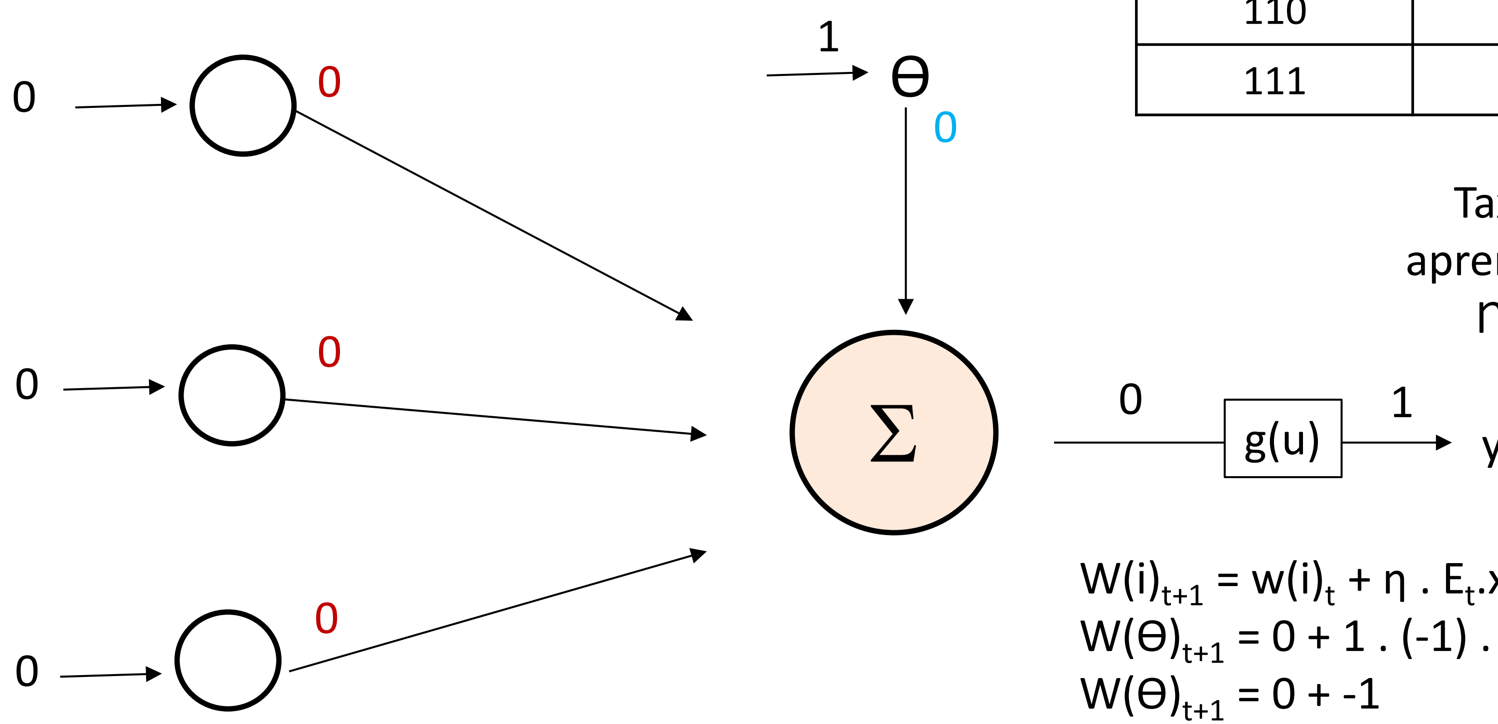
	X
X1	0
X2	0
X3	0

	W
W1	0
W2	0
W3	0

$\Theta = 1 \qquad w_{\Theta} = 0$

Exemplo
000

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



Taxa de aprendizado
$\eta = 1$

$W(i)_{t+1} = w(i)_t + \eta \,.\, E_t.x(i)$
$W(\Theta)_{t+1} = 0 + 1 \,.\, (-1) \,.\, 1$
$W(\Theta)_{t+1} = 0 + -1$

	X
X1	0
X2	0
X3	0

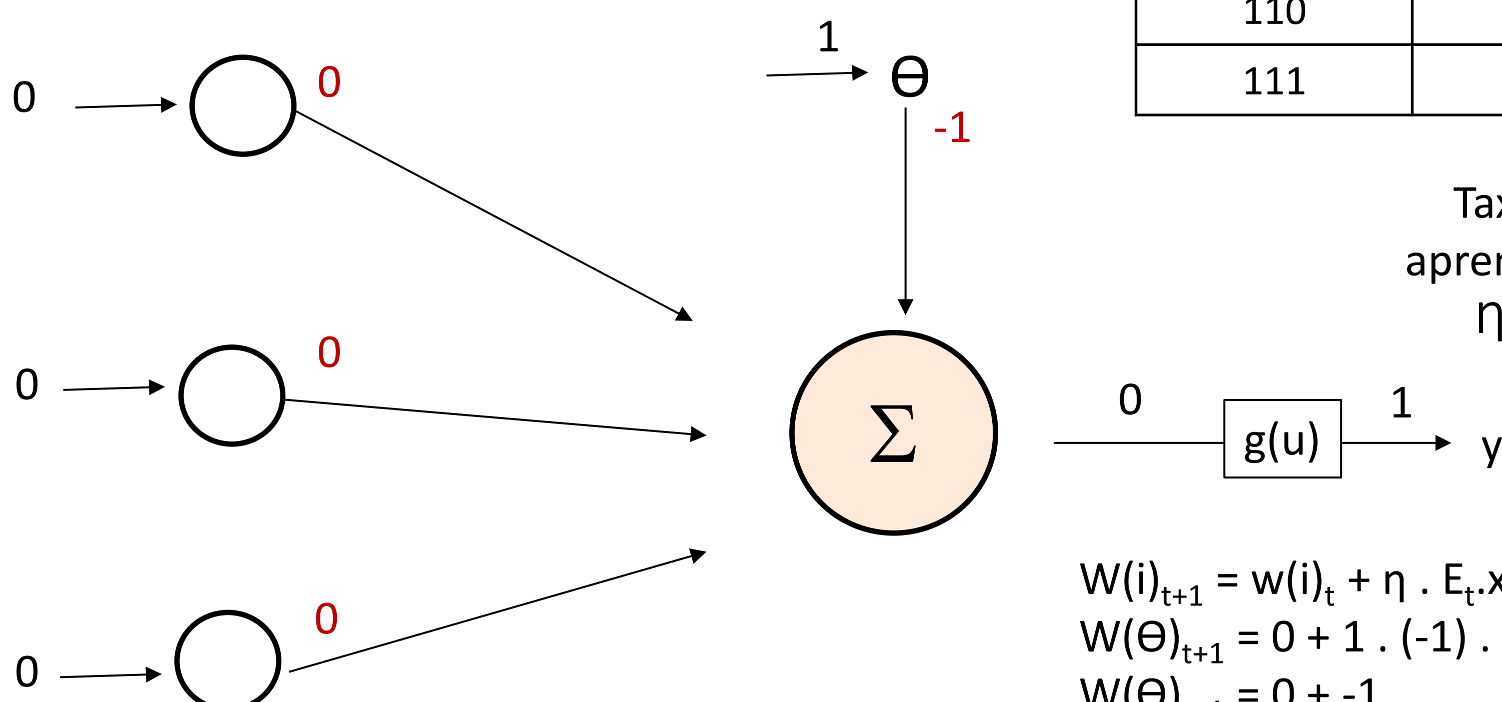
	W
W1	0
W2	0
W3	0

$\Theta = 1 \qquad w_\Theta = 0$

Exemplo

000

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



Taxa de aprendizado
$\eta = 1$

$W(i)_{t+1} = w(i)_t + \eta \;.\; E_t.x(i)$

$W(\Theta)_{t+1} = 0 + 1 \;.\; (-1) \;.\; 1$

$W(\Theta)_{t+1} = 0 + -1$

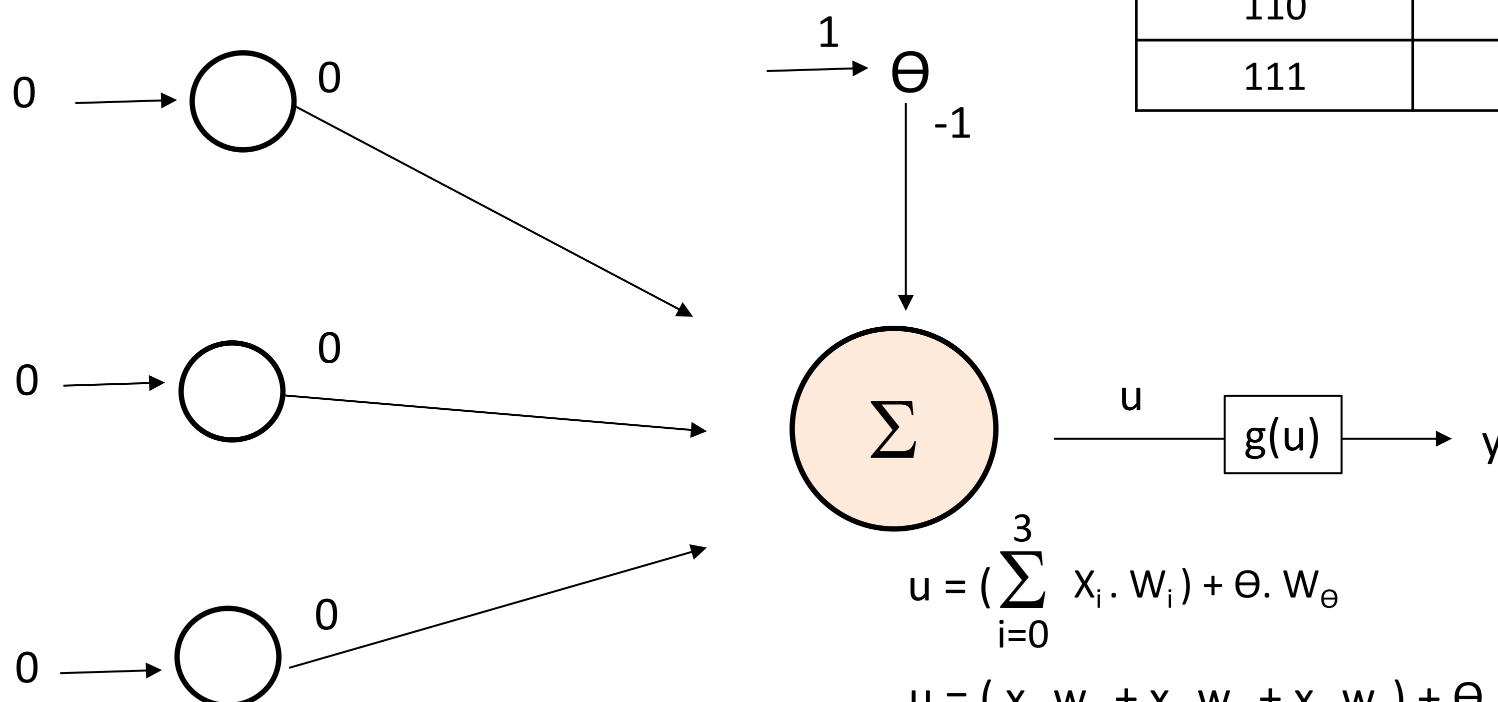
	X
X1	0
X2	0
X3	0

	W
W1	0
W2	0
W3	0

$\Theta = 1 \qquad w_\Theta = -1$

Exemplo
000

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



$$u = \left(\sum_{i=0}^{3} X_i \cdot W_i\right) + \Theta \cdot W_\Theta$$

$$u = (x_1.w_1 + x_2.w_2 + x_3.w_3) + \Theta \cdot w_\Theta$$

$$u = (0.0 + 0.0 + 0.0) + 1.-1$$

$$u = 0 + -1$$

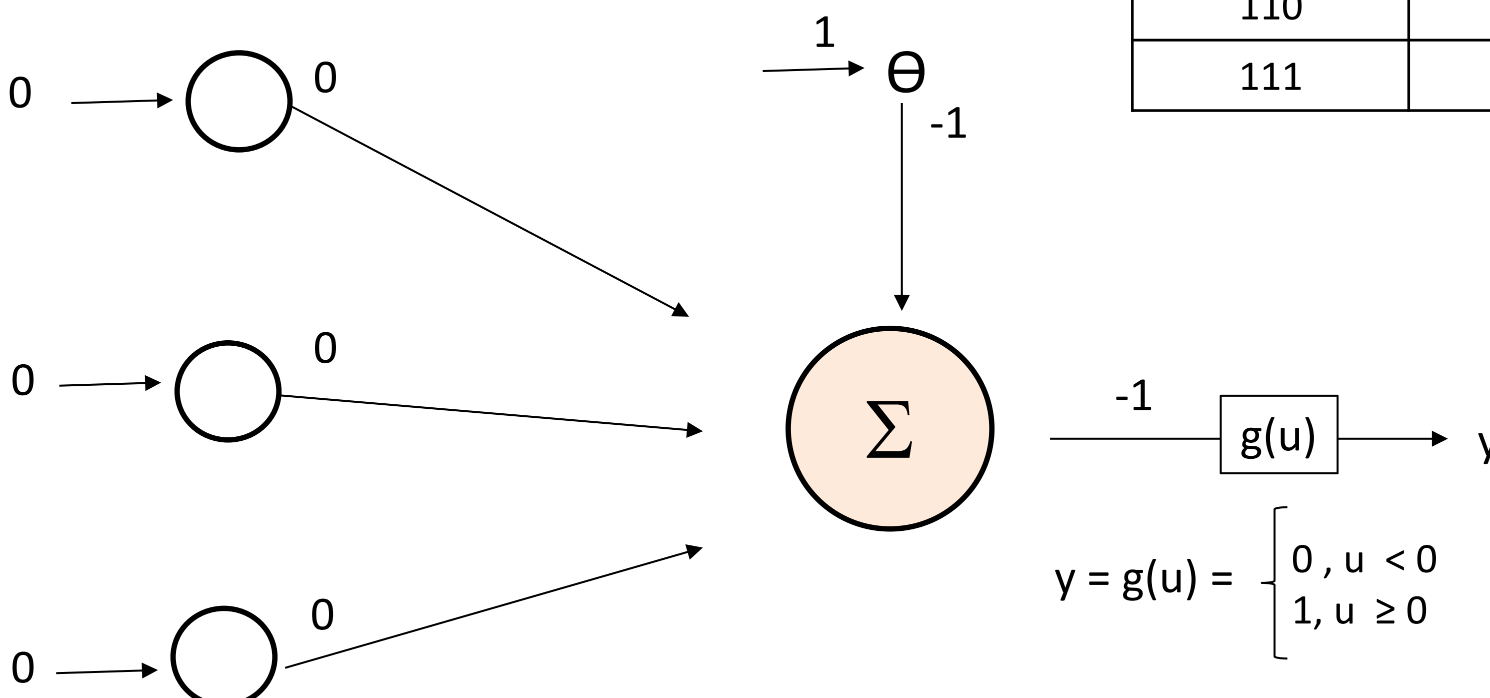
Exemplo
000

	X
X1	0
X2	0
X3	0

	W
W1	0
W2	0
W3	0

$\Theta = 1 \quad w_\Theta = -1$

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



$$y = g(u) = \begin{cases} 0, & u < 0 \\ 1, & u \geq 0 \end{cases}$$

$$y = g(0) = 0$$

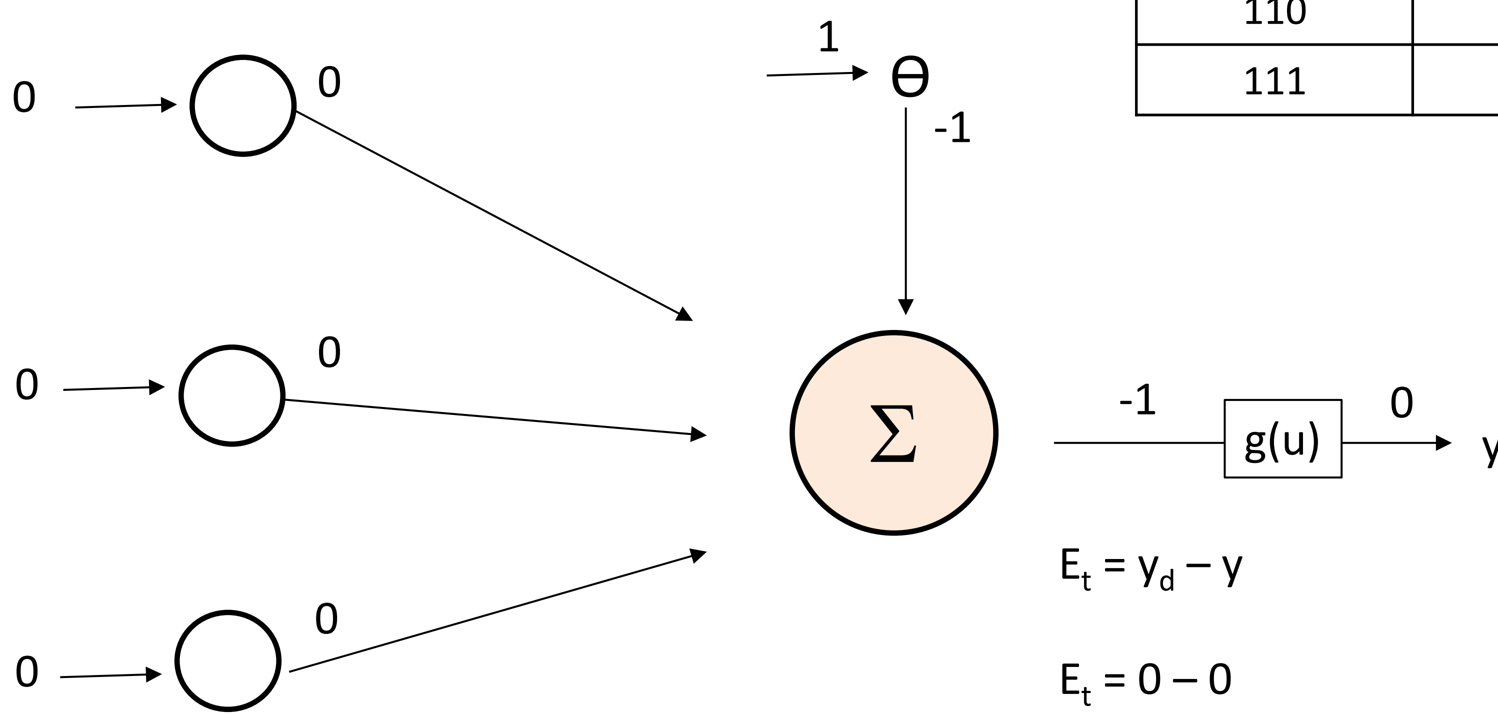
	X
X1	0
X2	0
X3	0

	W
W1	0
W2	0
W3	0

$\Theta = 1 \qquad w_\Theta = -1$

Exemplo
000

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



$E_t = y_d - y$

$E_t = 0 - 0$

$E_t = 0$

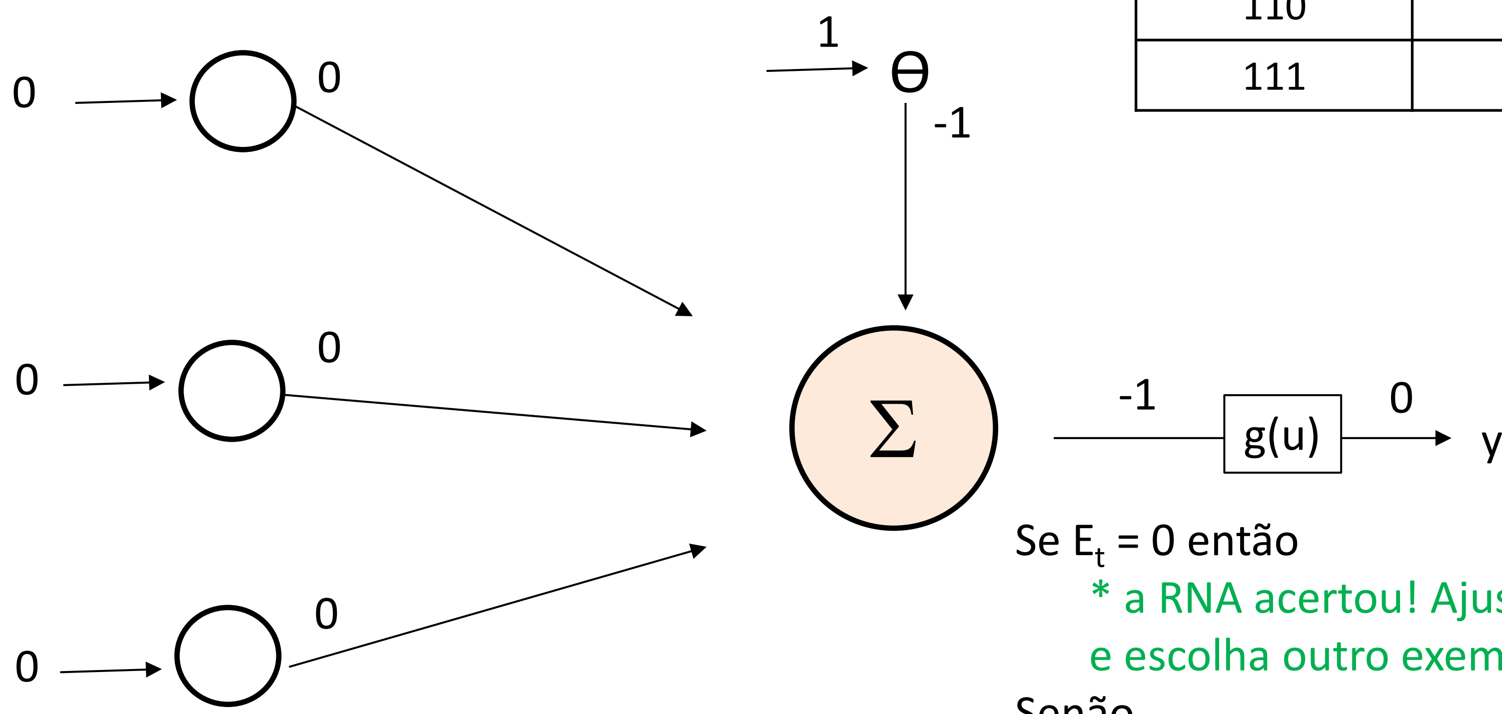
	X
X1	0
X2	0
X3	0

	W
W1	0
W2	0
W3	0

$\Theta = 1 \qquad w_{\Theta} = -1$

Exemplo
000

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



Se E_t = 0 então

* a RNA acertou! Ajuste os pesos e escolha outro exemplo

Senão

* Ajuste os pesos sinápticos e continue no exemplo até acertar

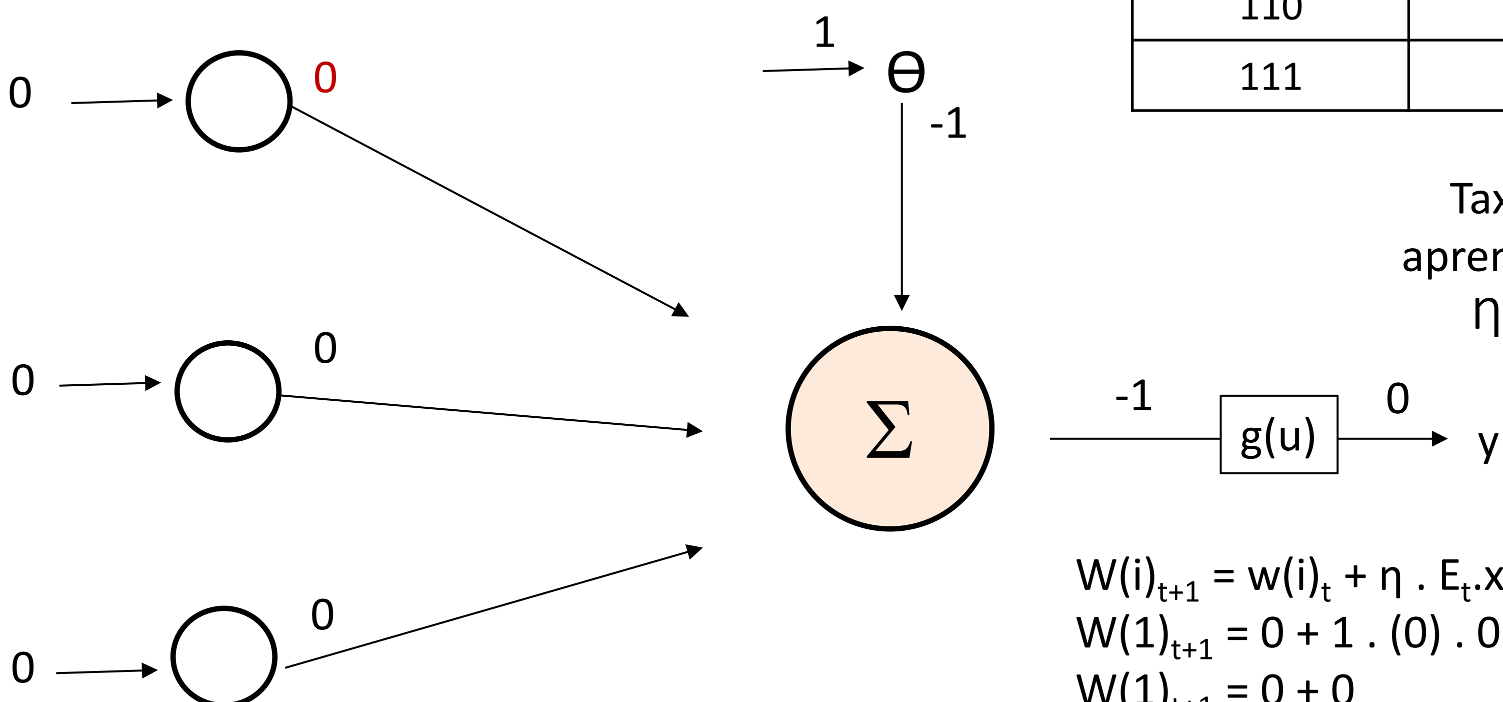
	X
X1	0
X2	0
X3	0

	W
W1	0
W2	0
W3	0

Exemplo
000

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1

$\Theta = 1$ $\quad w_{\Theta} = -1$



Taxa de aprendizado
$\eta = 1$

$$W(i)_{t+1} = w(i)_t + \eta \,.\, E_t . x(i)$$
$$W(1)_{t+1} = 0 + 1 \,.\, (0) \,.\, 0$$
$$W(1)_{t+1} = 0 + 0$$

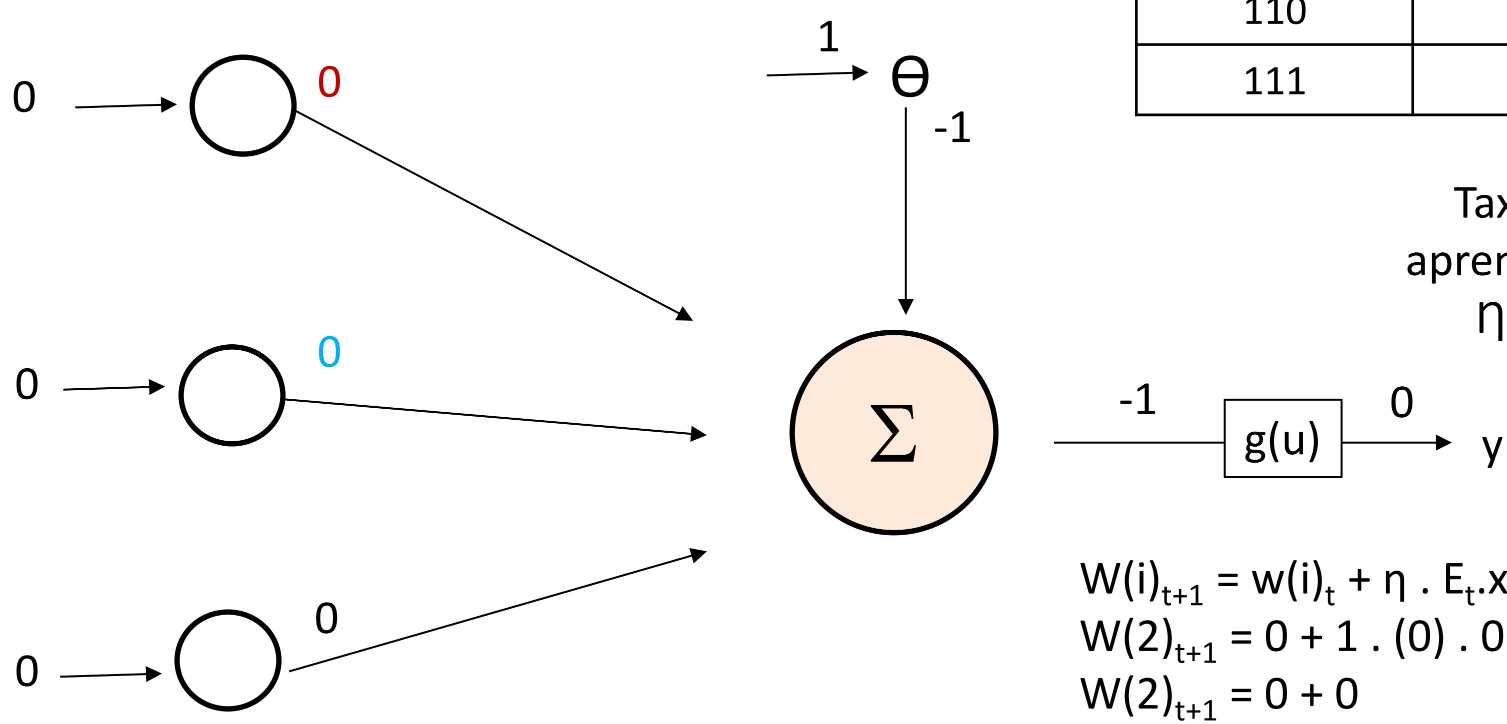
	X
X1	0
X2	0
X3	0

	W
W1	0
W2	0
W3	0

$\Theta = 1 \qquad w_{\Theta} = -1$

Exemplo
000

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



Taxa de aprendizado
$\eta = 1$

$W(i)_{t+1} = w(i)_t + \eta \,.\, E_t.x(i)$
$W(2)_{t+1} = 0 + 1 \,.\, (0) \,.\, 0$
$W(2)_{t+1} = 0 + 0$

	X
X1	0
X2	0
X3	0

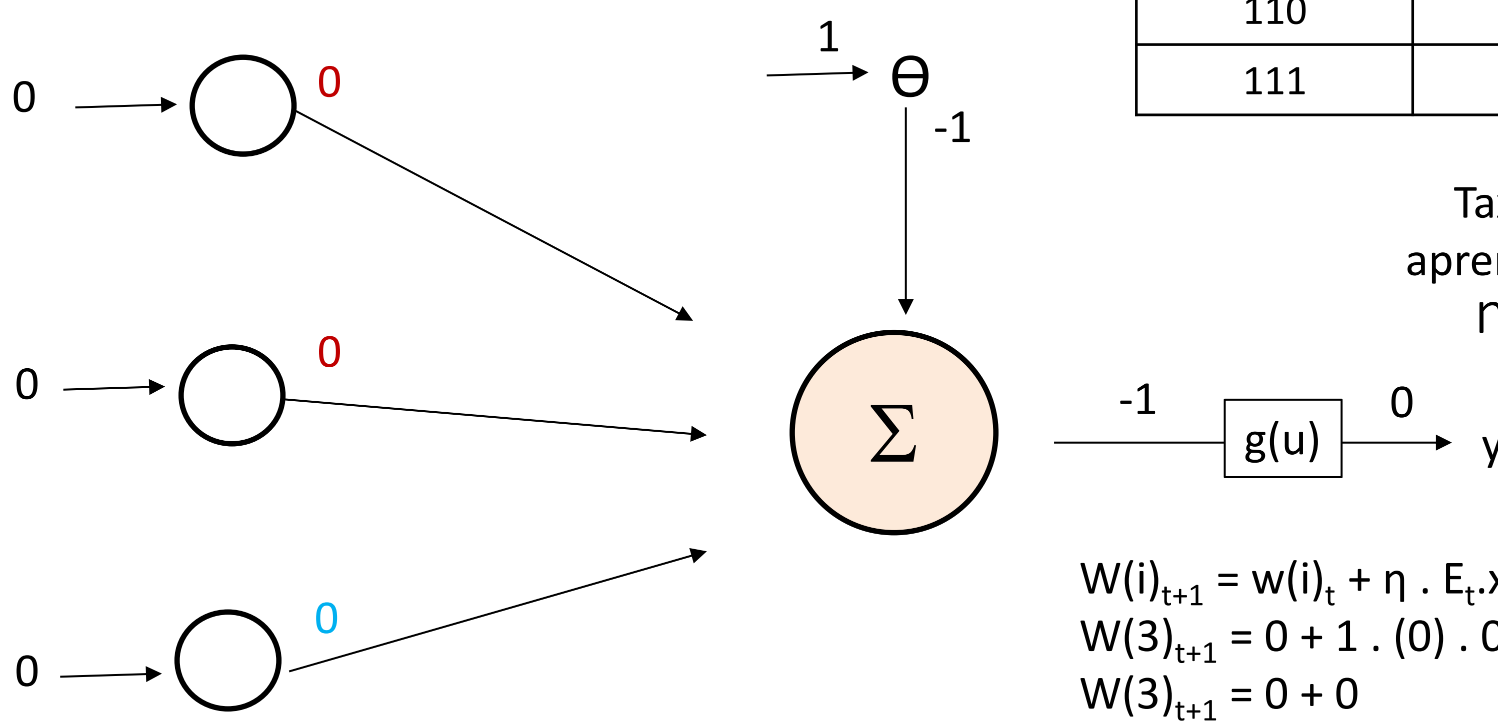
	W
W1	0
W2	0
W3	0

$\Theta = 1 \qquad w_{\Theta} = -1$

Exemplo
000

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1

Taxa de aprendizado
$\eta = 1$



$W(i)_{t+1} = w(i)_t + \eta \;.\; E_t.x(i)$

$W(3)_{t+1} = 0 + 1 \;.\; (0) \;.\; 0$

$W(3)_{t+1} = 0 + 0$

	X
X1	0
X2	0
X3	0

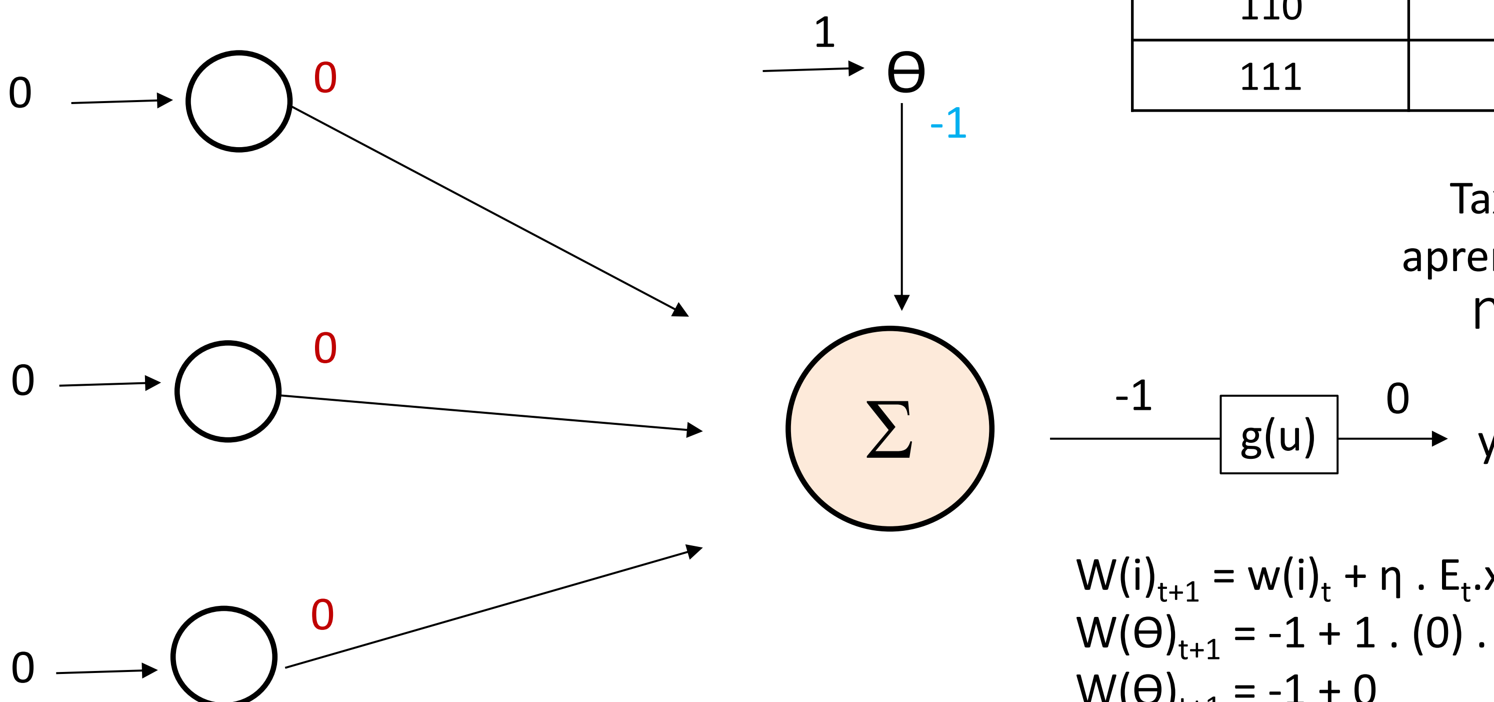
	W
W1	0
W2	0
W3	0

Exemplo

000

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1

$\Theta = 1$ $\quad w_{\Theta} = -1$



Taxa de aprendizado
$\eta = 1$

$$W(i)_{t+1} = w(i)_t + \eta \, . \, E_t . x(i)$$
$$W(\Theta)_{t+1} = -1 + 1 \, . \, (0) \, . \, 1$$
$$W(\Theta)_{t+1} = -1 + 0$$

	X
X1	0
X2	1
X3	1

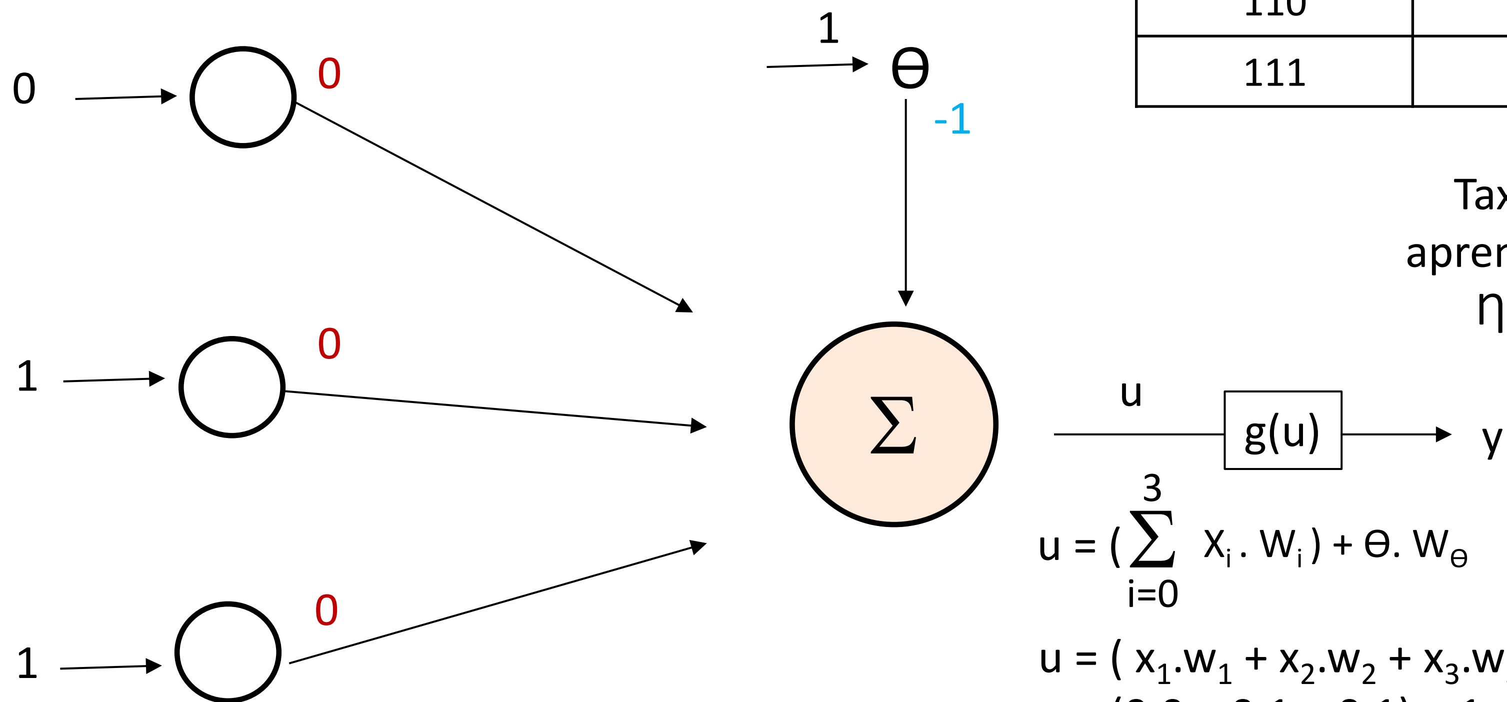
	W
W1	0
W2	0
W3	0

$\Theta = 1 \qquad w_\Theta = -1$

Outro exemplo

011

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



Taxa de aprendizado

$\eta = 1$

$$u = \left(\sum_{i=0}^{3} X_i \,.\, W_i\right) + \Theta .\, W_\Theta$$

$$u = (\, x_1.w_1 + x_2.w_2 + x_3.w_3) + \Theta \,.\, w_\Theta$$

$$u = (0.0 + 0.1 + 0.1) + 1.-1$$

$$u = 0 + -1$$

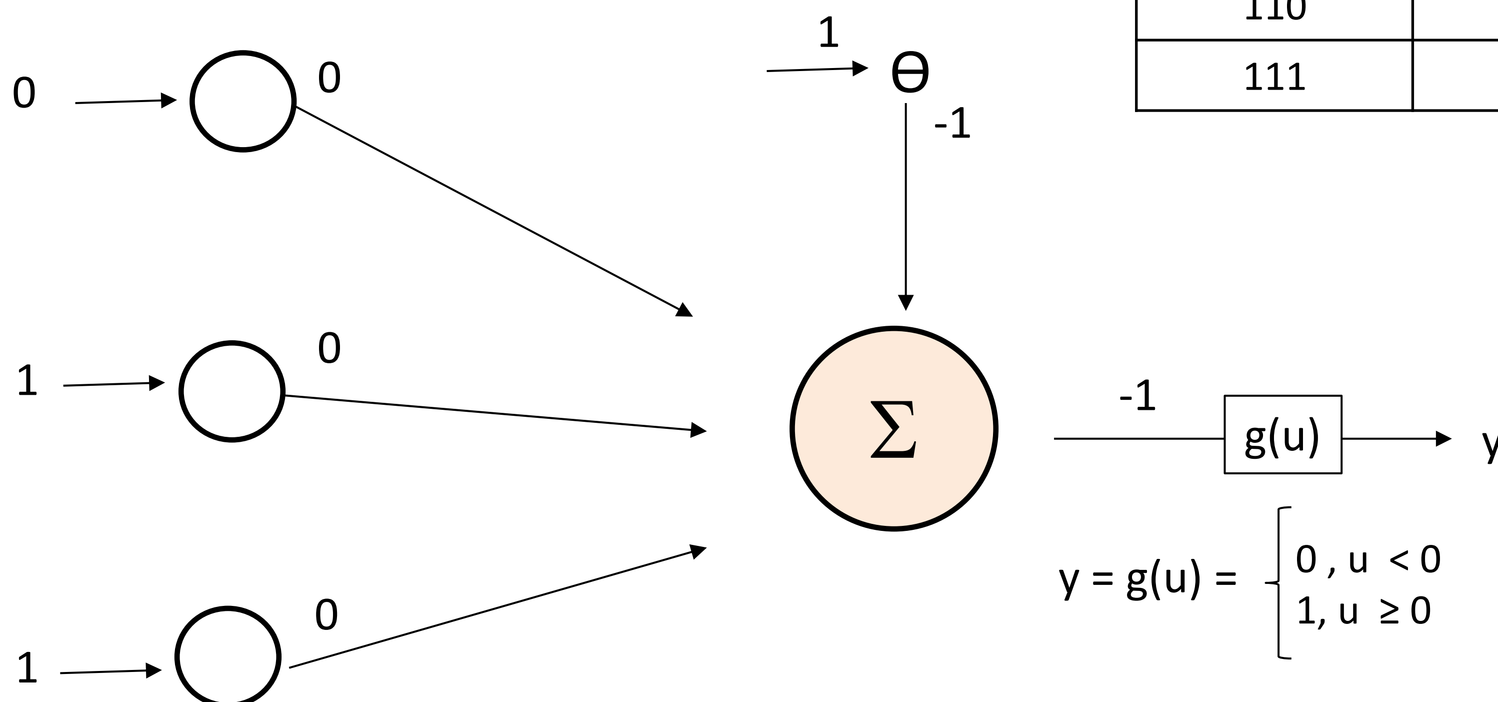
	X
X1	0
X2	1
X3	1

	W
W1	0
W2	0
W3	0

$\Theta = 1 \qquad w_{\Theta} = -1$

Exemplo
011

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



$$y = g(u) = \begin{cases} 0, & u < 0 \\ 1, & u \geq 0 \end{cases}$$

$$y = g(0) = 0$$

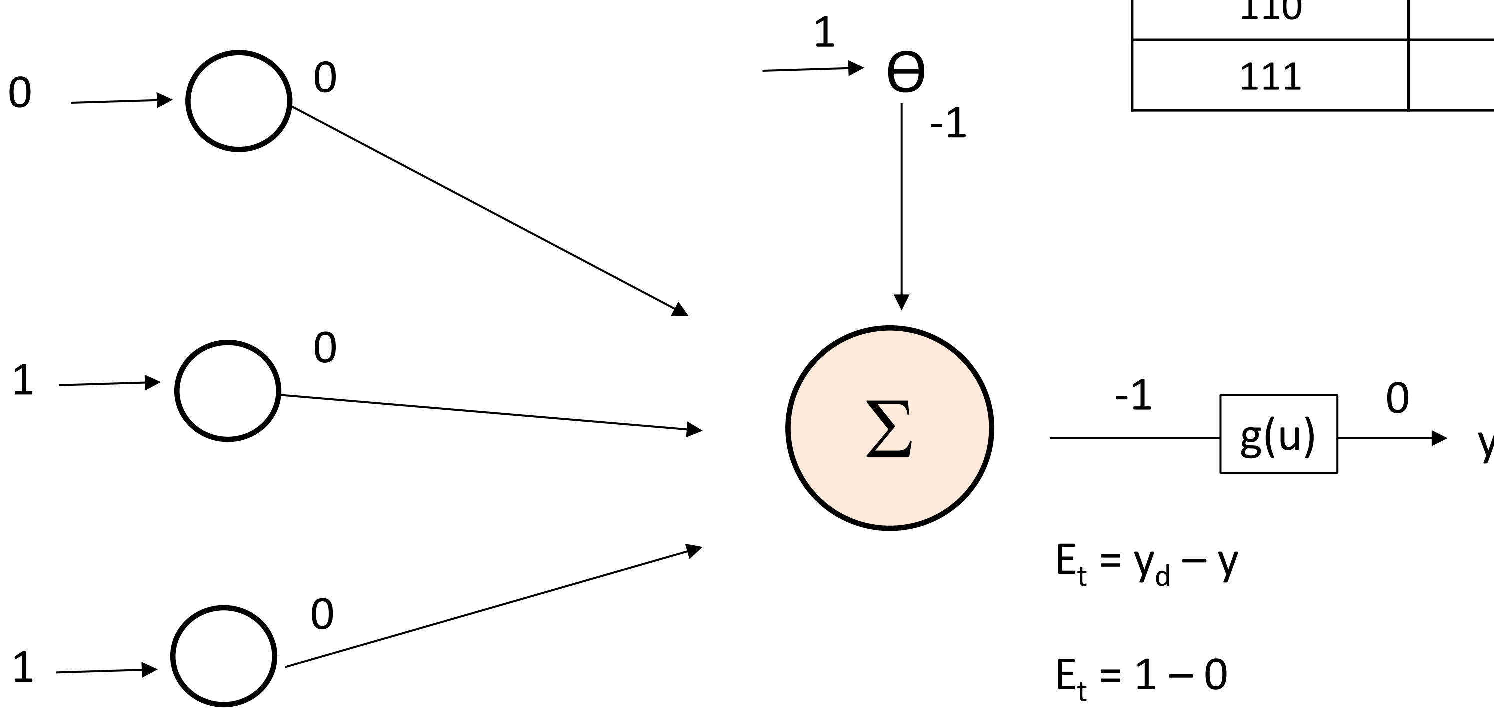
	X
X1	0
X2	1
X3	1

	W
W1	0
W2	0
W3	0

$\Theta = 1 \quad w_\Theta = -1$

Exemplo
000

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



$E_t = y_d - y$

$E_t = 1 - 0$

$E_t = 1$

	X
X1	0
X2	0
X3	0

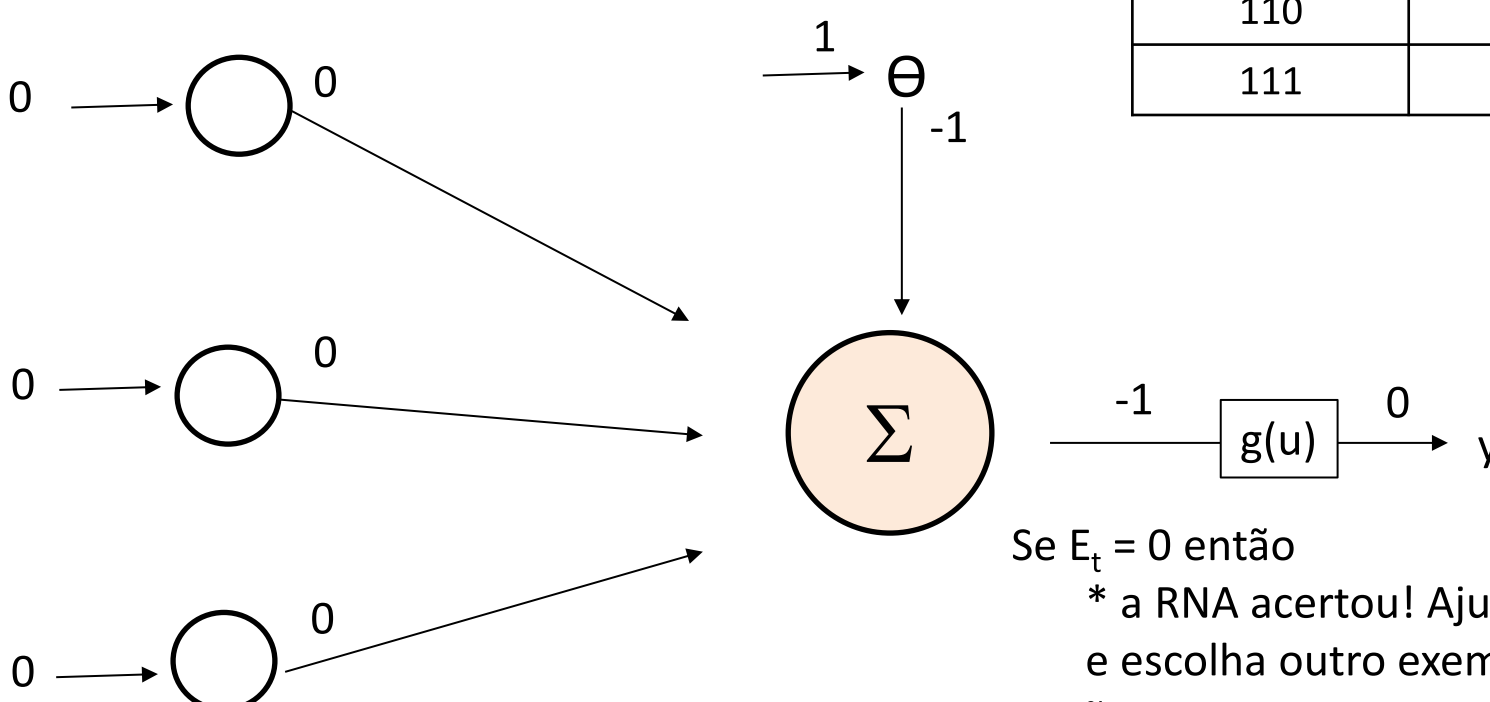
	W
W1	0
W2	0
W3	0

$\Theta = 1$ $w_\Theta = -1$

Exemplo

000

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



Se $E_t = 0$ então

* a RNA acertou! Ajuste os pesos e escolha outro exemplo

Senão

* Ajuste os pesos sinápticos e continue no exemplo até acertar

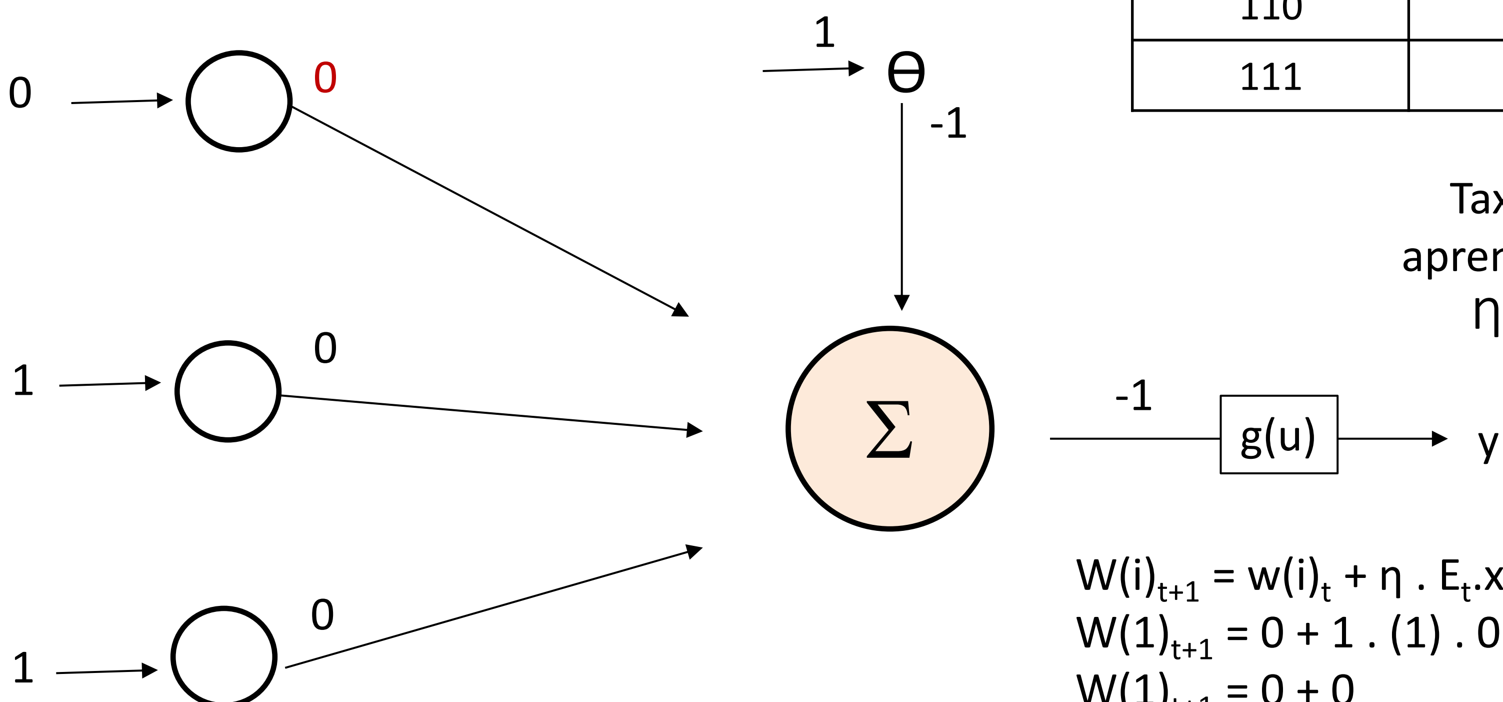
	X
X1	0
X2	1
X3	1

	W
W1	0
W2	0
W3	0

$\Theta = 1 \qquad w_\Theta = -1$

Exemplo
000

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



Taxa de aprendizado
$\eta = 1$

$W(i)_{t+1} = w(i)_t + \eta \cdot E_t . x(i)$

$W(1)_{t+1} = 0 + 1 \cdot (1) \cdot 0$

$W(1)_{t+1} = 0 + 0$

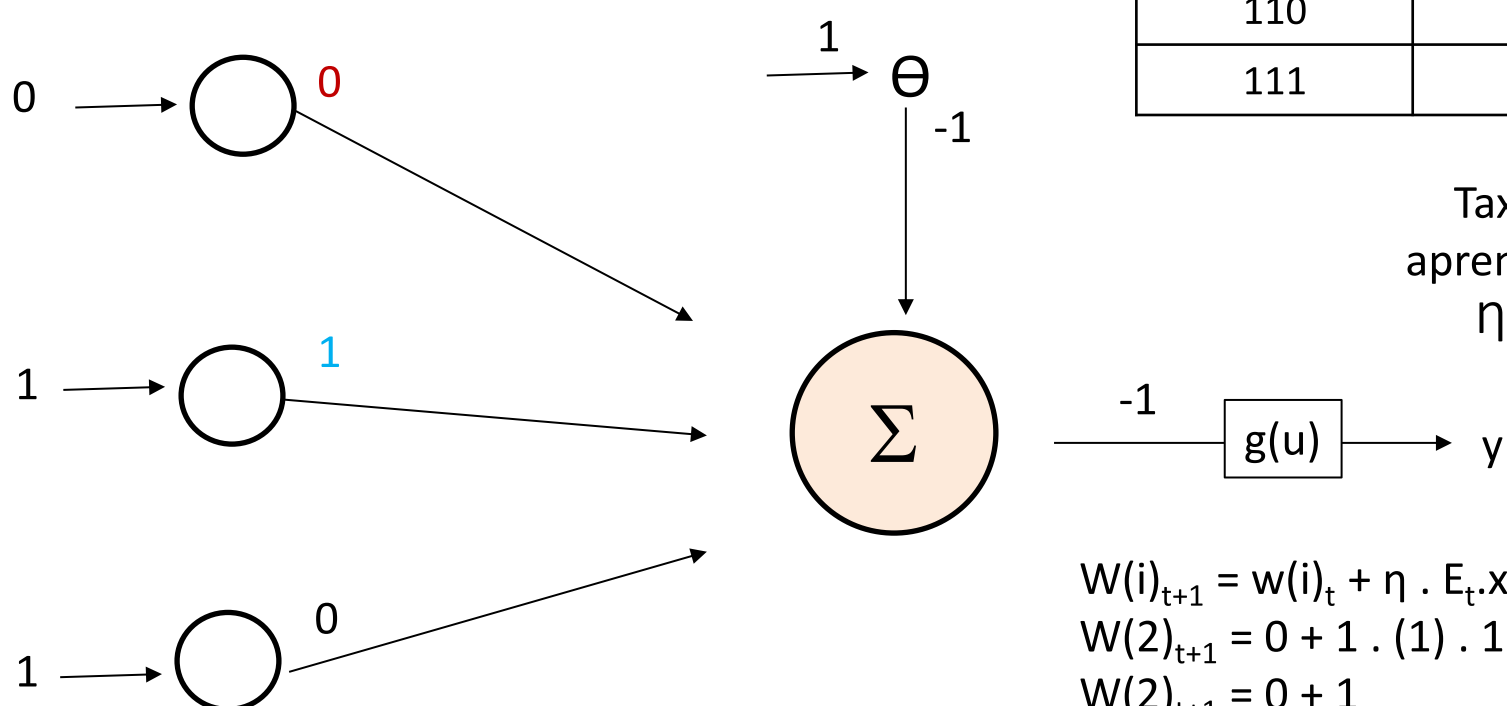
	X
X1	0
X2	1
X3	1

	W
W1	0
W2	0
W3	0

$\Theta = 1 \qquad w_{\Theta} = -1$

Exemplo
011

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



Taxa de aprendizado
$\eta = 1$

$W(i)_{t+1} = w(i)_t + \eta \cdot E_t . x(i)$
$W(2)_{t+1} = 0 + 1 \cdot (1) \cdot 1$
$W(2)_{t+1} = 0 + 1$

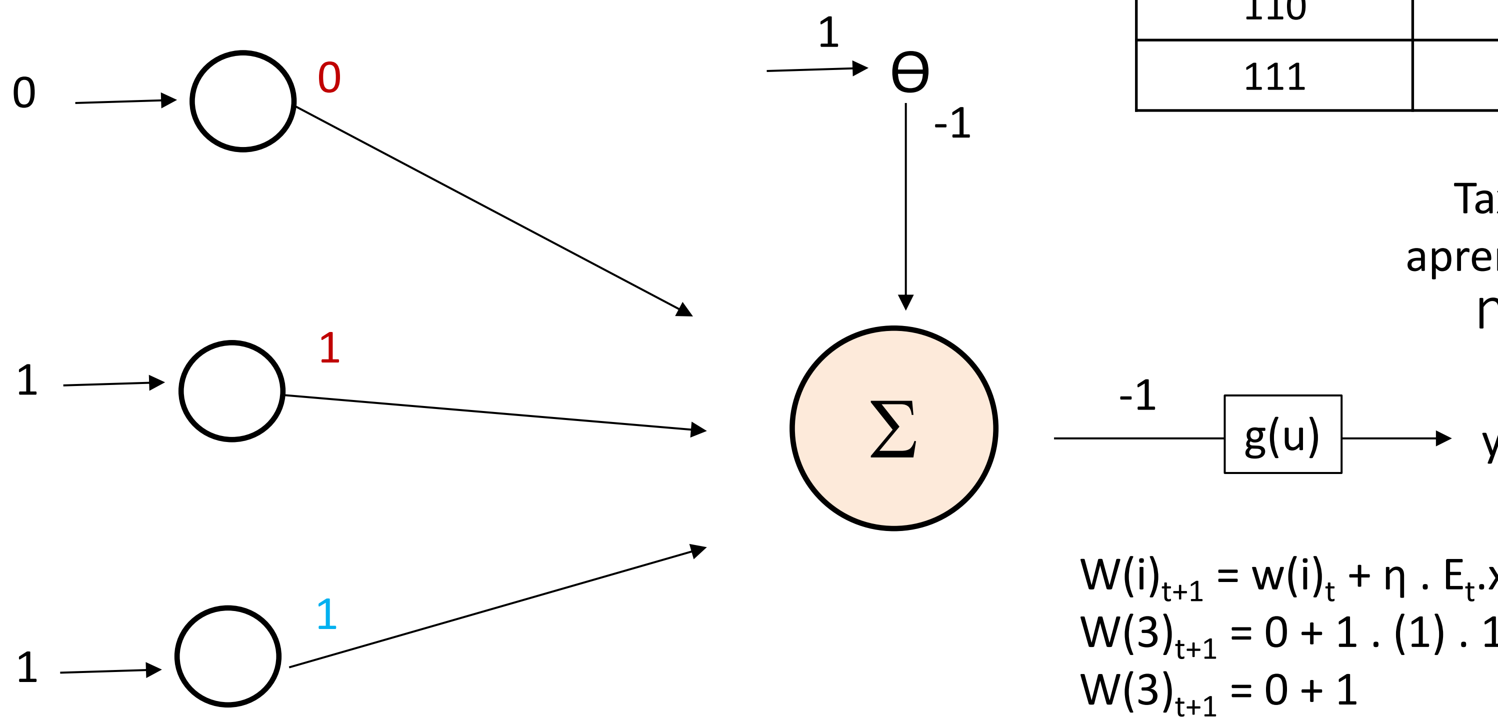
	X
X1	0
X2	1
X3	1

	W
W1	0
W2	0
W3	0

$\Theta = 1 \quad w_{\Theta} = -1$

Exemplo
011

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



Taxa de aprendizado
$\eta = 1$

$W(i)_{t+1} = w(i)_t + \eta \;.\; E_t.x(i)$
$W(3)_{t+1} = 0 + 1 \;.\; (1) \;.\; 1$
$W(3)_{t+1} = 0 + 1$

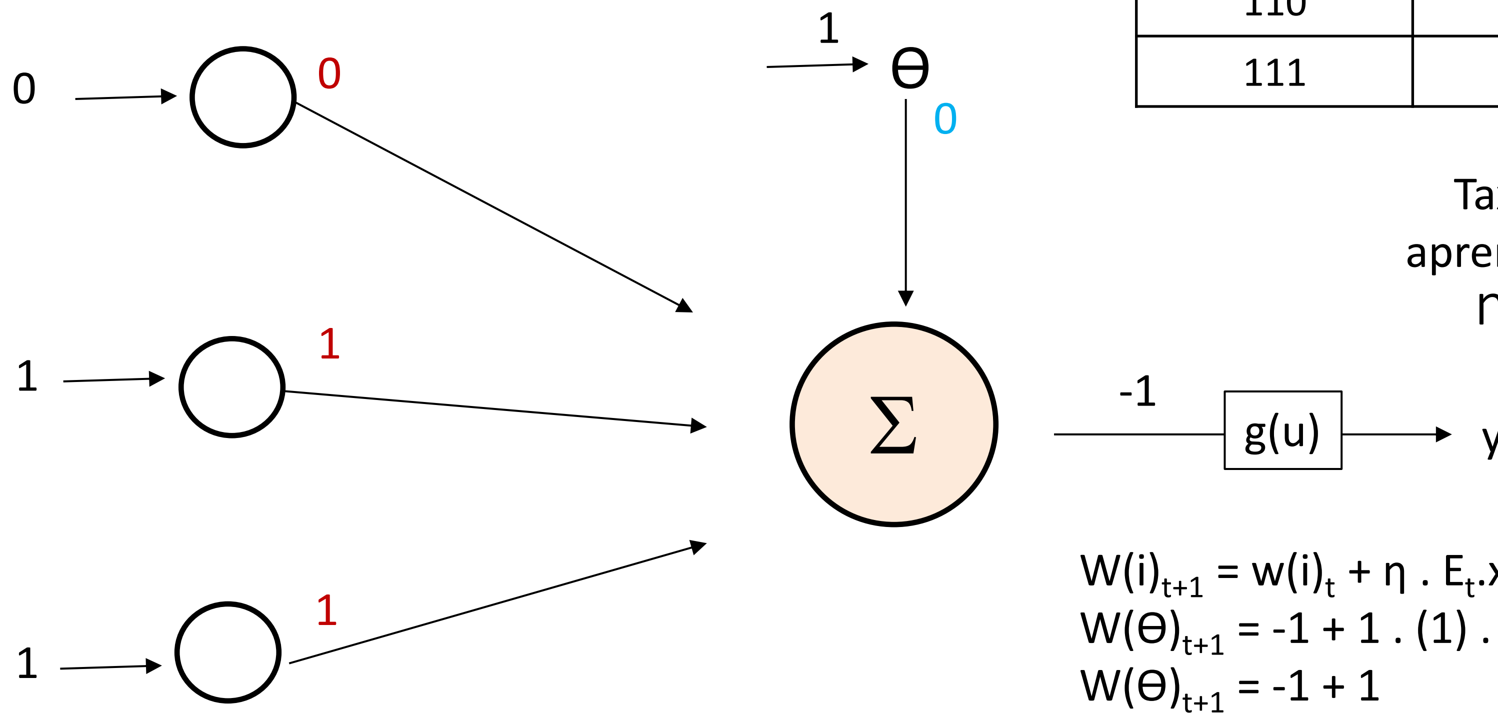
	X
X1	0
X2	1
X3	1

	W
W1	0
W2	0
W3	0

$\Theta = 1 \qquad w_{\Theta} = -1$

Exemplo
011

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



Taxa de aprendizado
$\eta = 1$

$W(i)_{t+1} = w(i)_t + \eta \cdot E_t . x(i)$

$W(\Theta)_{t+1} = -1 + 1 \cdot (1) \cdot 1$

$W(\Theta)_{t+1} = -1 + 1$

	X
X1	0
X2	1
X3	1

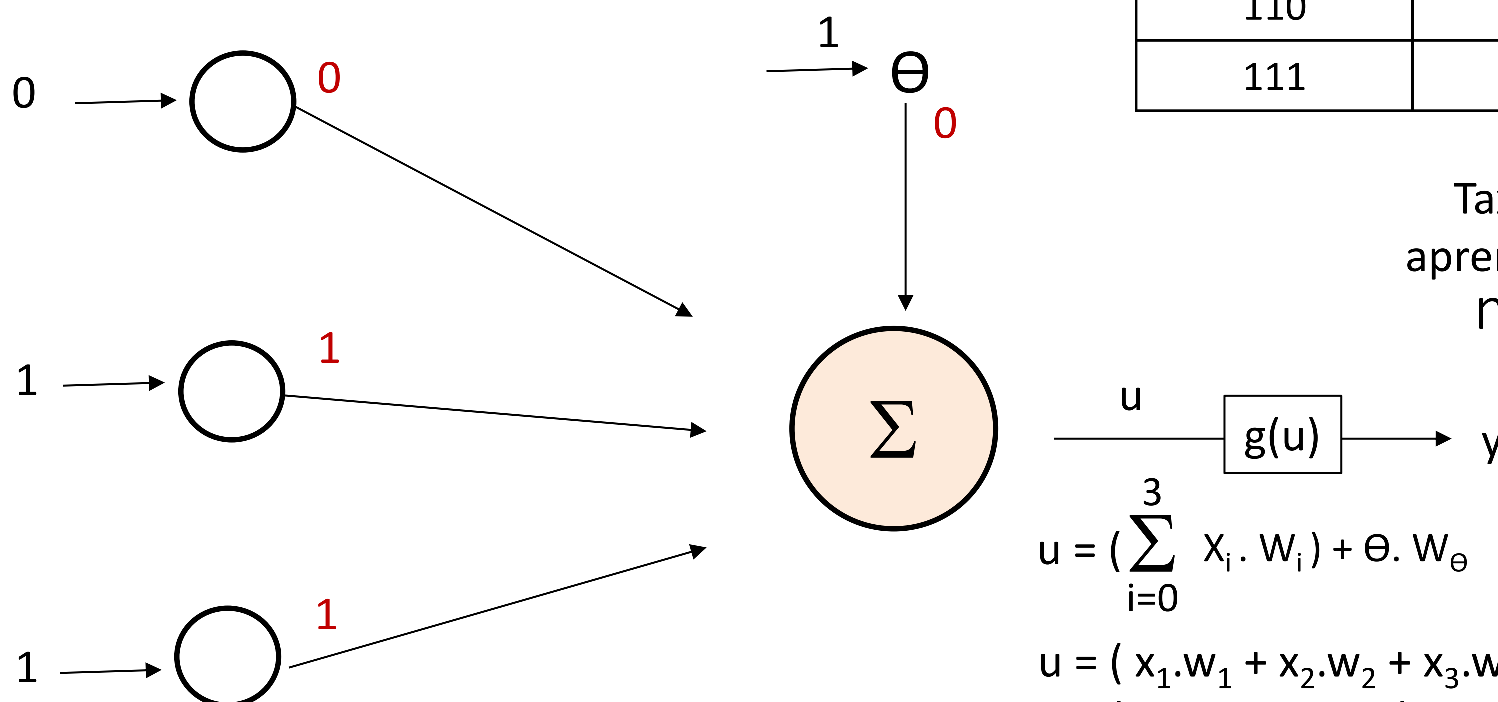
	W
W1	0
W2	1
W3	1

$\Theta = 1$ $\quad w_\Theta = 0$

Outro exemplo

011

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



Taxa de aprendizado

$\eta = 1$

$$u = \left(\sum_{i=0}^{3} X_i \,.\, W_i\right) + \Theta.\, W_\Theta$$

$$u = (\, x_1.w_1 + x_2.w_2 + x_3.w_3) + \Theta \,.\, w_\Theta$$

$$u = (0.0 + 1.1 + 1.1) + 1.0$$

$$u = 2 + 0$$

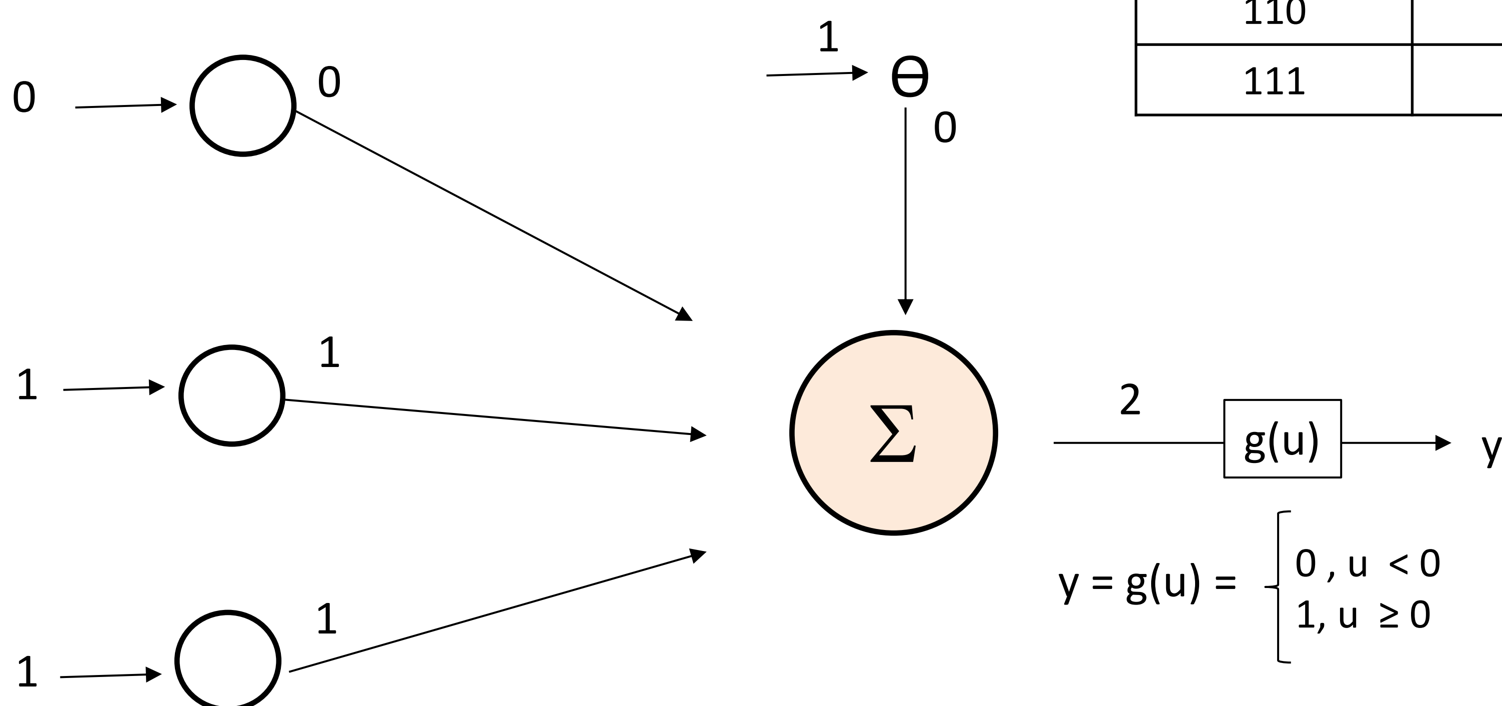
	X
X1	0
X2	1
X3	1

	W
W1	0
W2	1
W3	1

$\Theta = 1 \qquad w_\Theta = 0$

Exemplo
011

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



$$y = g(u) = \begin{cases} 0, & u < 0 \\ 1, & u \geq 0 \end{cases}$$

$$y = g(2) = 1$$

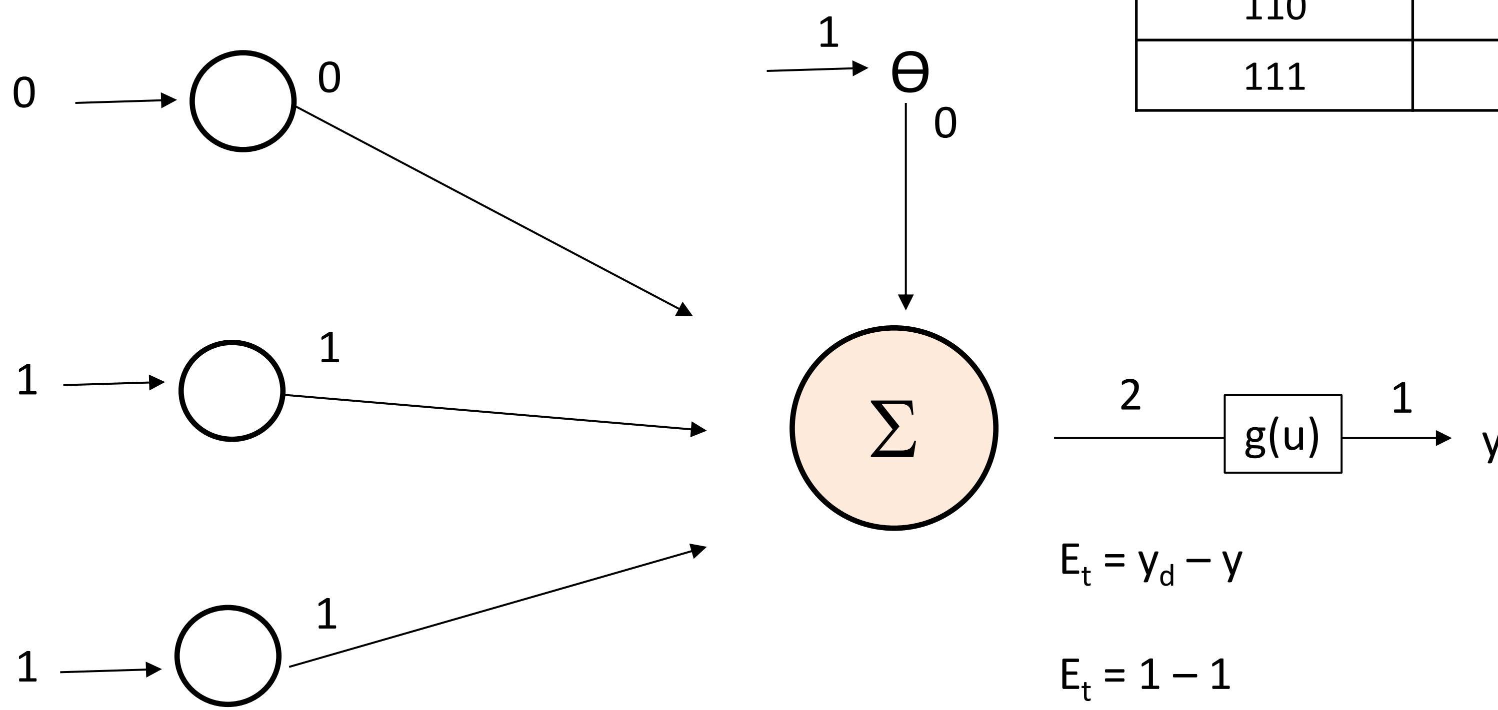
	X
X1	0
X2	1
X3	1

	W
W1	0
W2	1
W3	1

$\Theta = 1 \qquad w_{\Theta} = 0$

Exemplo
011

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



$E_t = y_d - y$

$E_t = 1 - 1$

$E_t = 0$

Exemplo

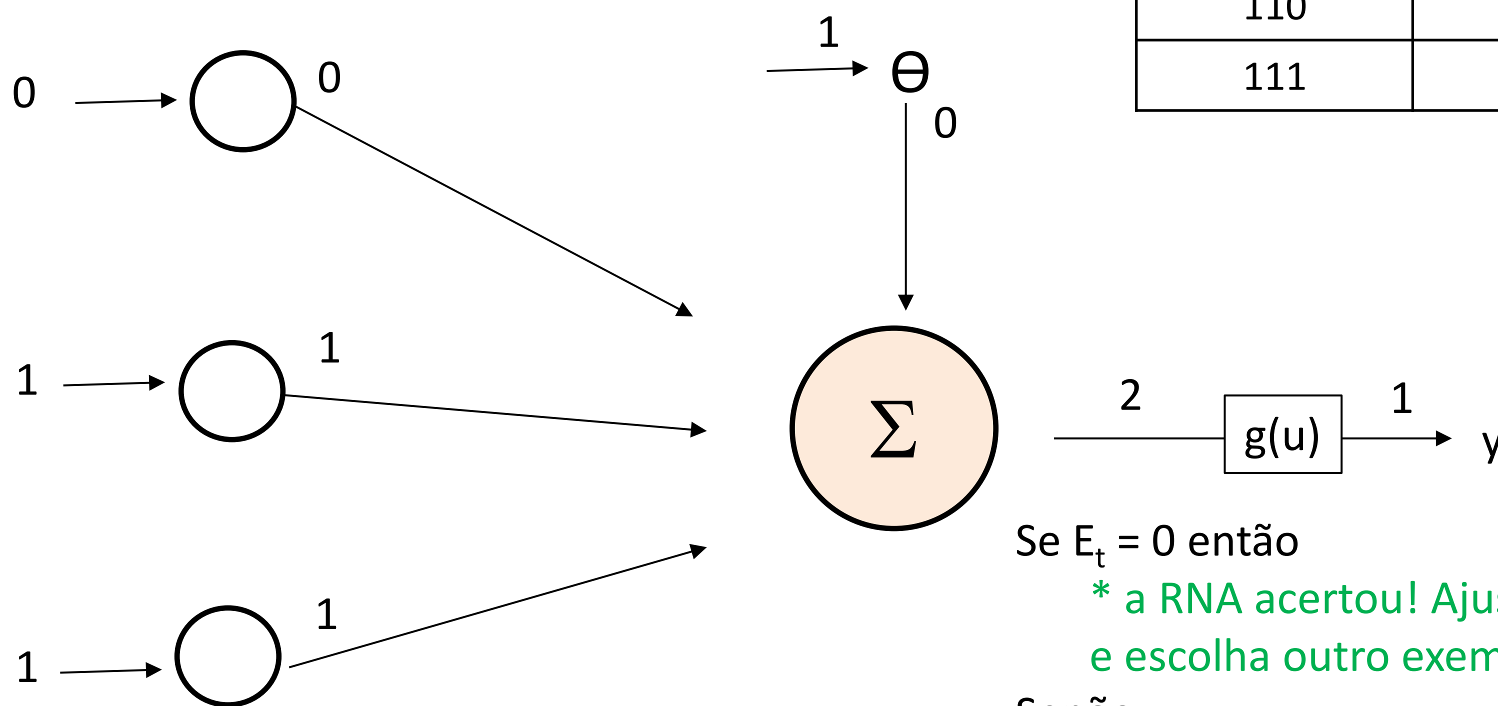
011

	X
X1	0
X2	1
X3	1

	W
W1	0
W2	1
W3	1

$\Theta = 1$ $\quad w_\Theta = 0$

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



Se $E_t = 0$ então

* a RNA acertou! Ajuste os pesos e escolha outro exemplo

Senão

* Ajuste os pesos sinápticos e continue no exemplo até acertar

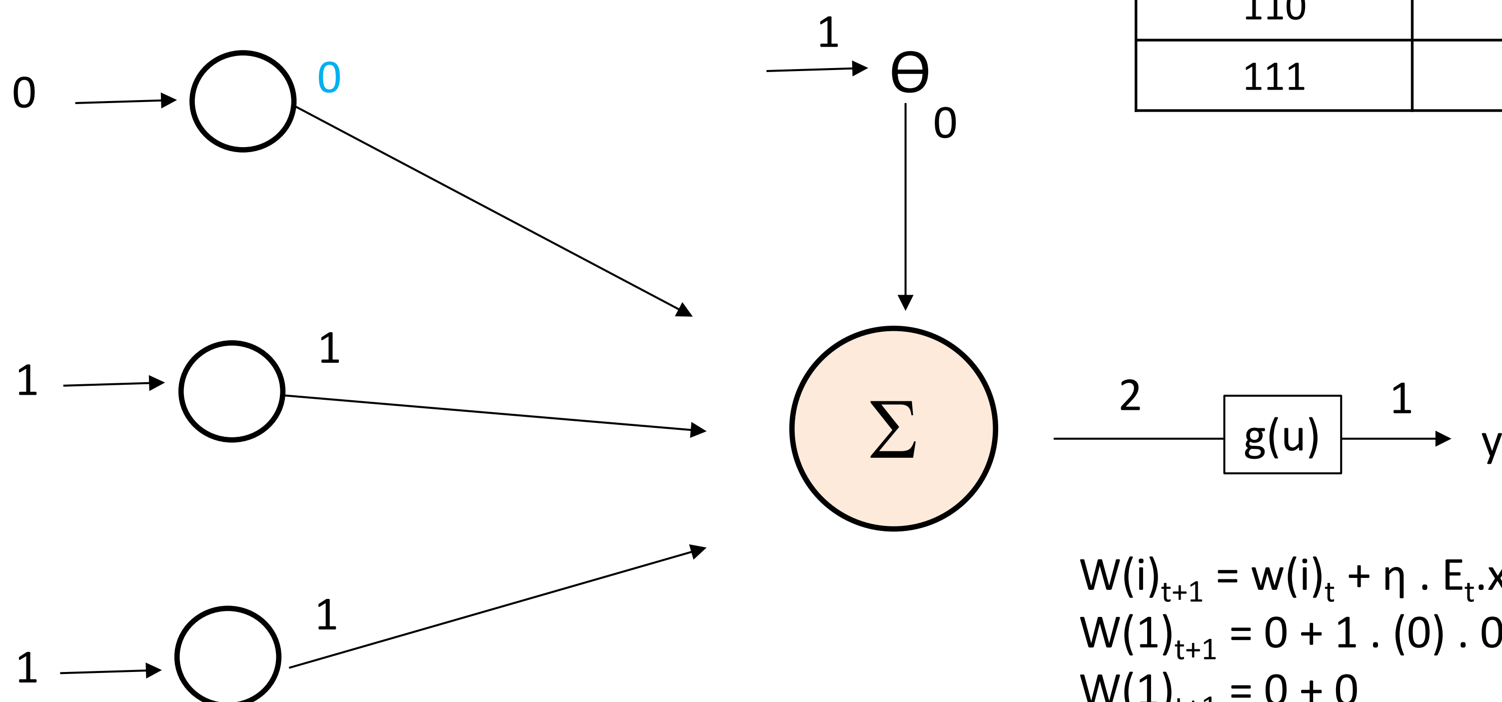
	X
X1	0
X2	1
X3	1

	W
W1	0
W2	1
W3	1

$\Theta = 1 \qquad w_\Theta = 0$

Exemplo
011

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



$W(i)_{t+1} = w(i)_t + \eta \ . \ E_t.x(i)$

$W(1)_{t+1} = 0 + 1 \ . \ (0) \ . \ 0$

$W(1)_{t+1} = 0 + 0$

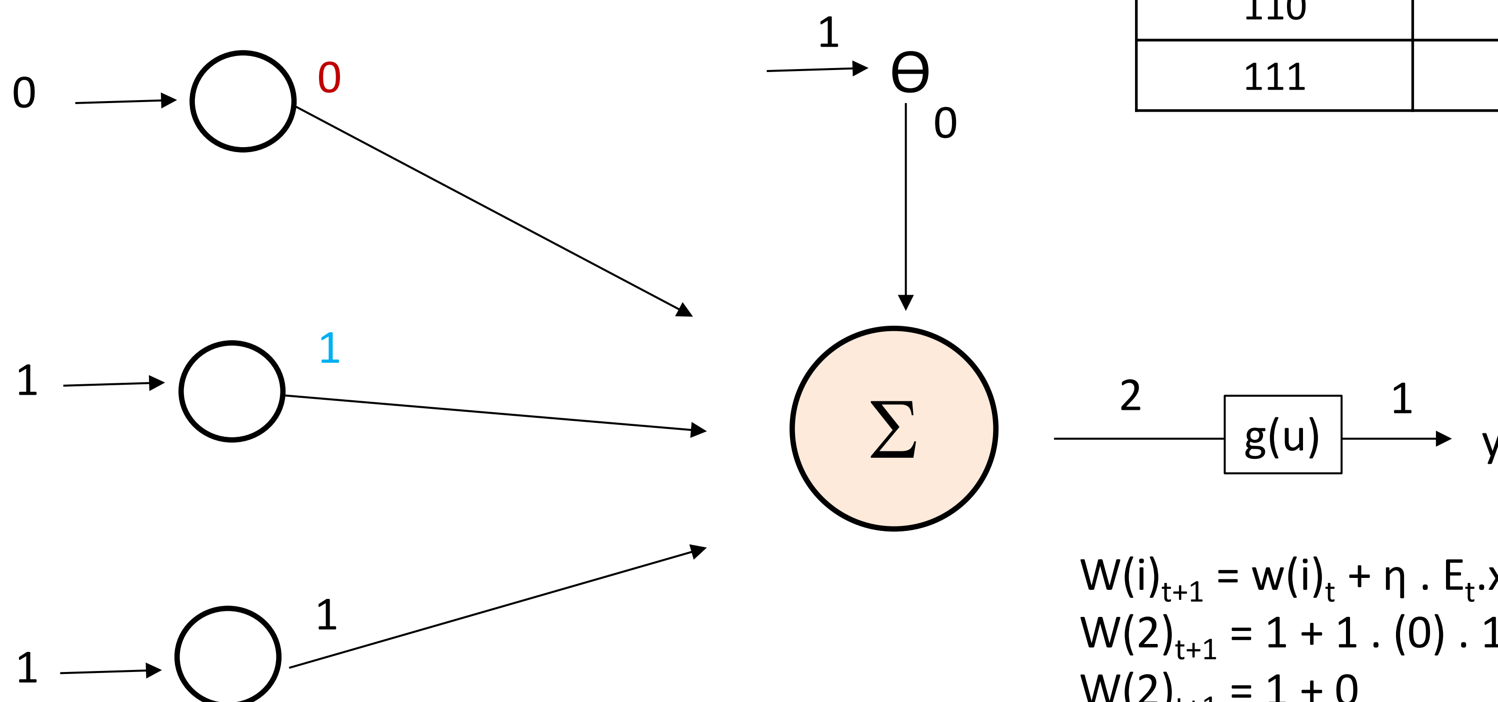
	X
X1	0
X2	1
X3	1

	W
W1	0
W2	1
W3	1

$\Theta = 1 \qquad w_{\Theta} = 0$

Exemplo
011

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



$W(i)_{t+1} = w(i)_t + \eta \,.\, E_t.x(i)$
$W(2)_{t+1} = 1 + 1 \,.\, (0) \,.\, 1$
$W(2)_{t+1} = 1 + 0$

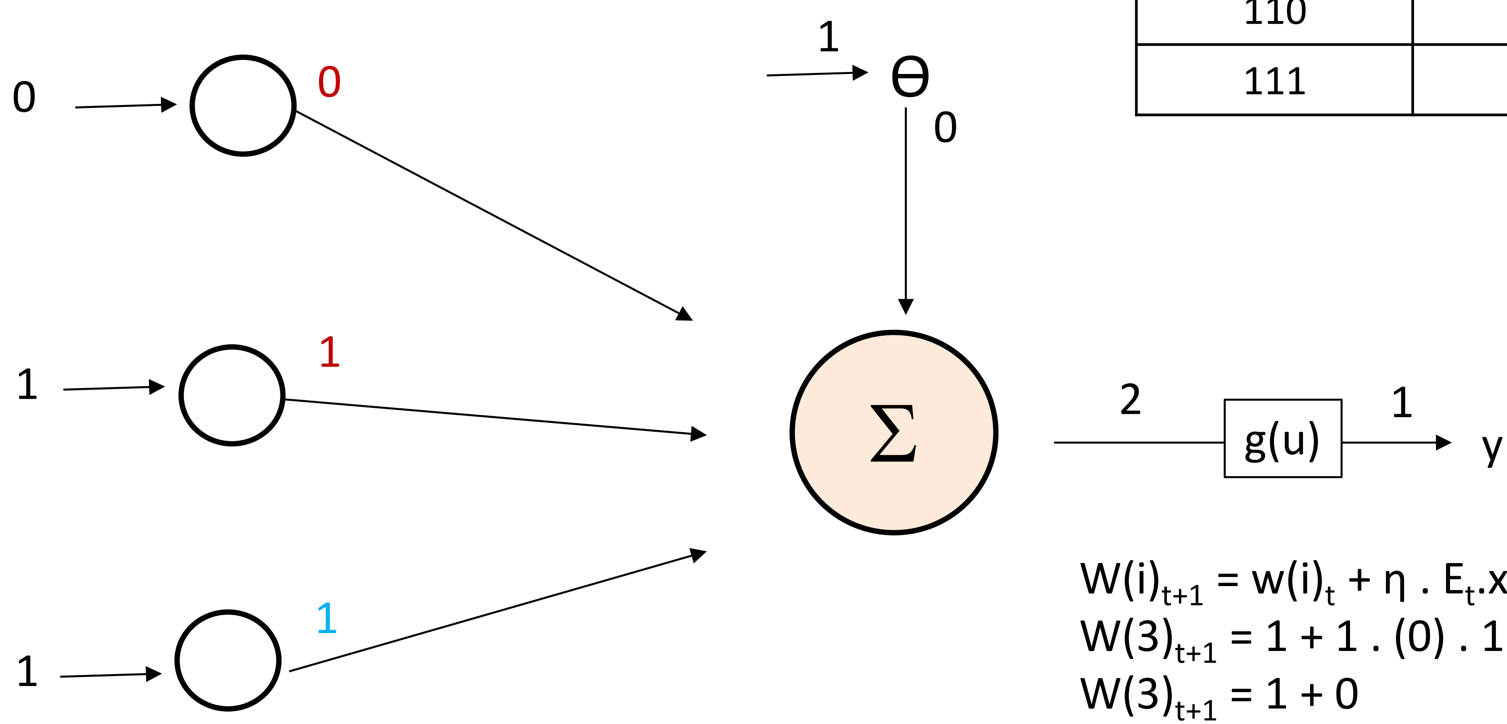
	X
X1	0
X2	1
X3	1

	W
W1	0
W2	1
W3	1

$\Theta = 1 \qquad w_{\Theta} = 0$

Exemplo
011

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



$W(i)_{t+1} = w(i)_t + \eta \,.\, E_t.x(i)$

$W(3)_{t+1} = 1 + 1 \,.\, (0) \,.\, 1$

$W(3)_{t+1} = 1 + 0$

Exemplo

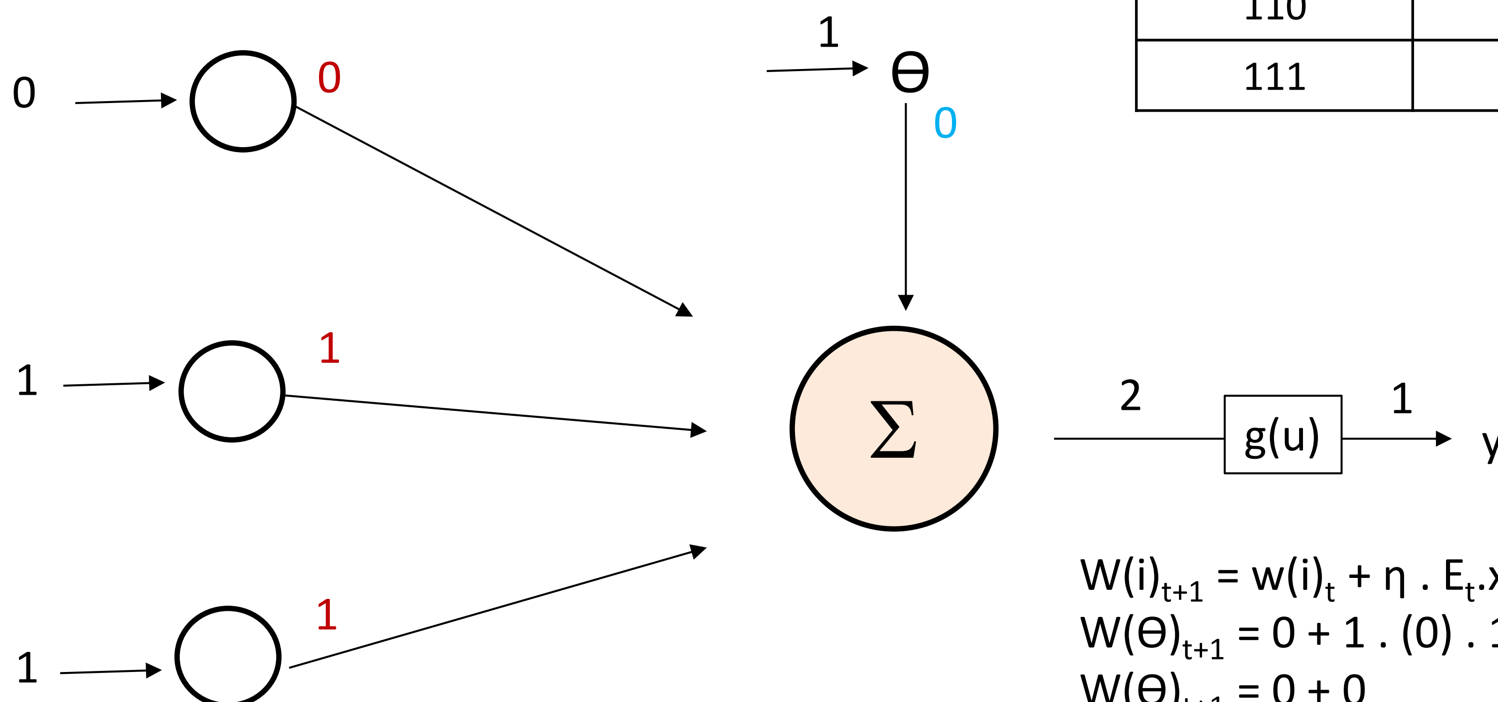
011

	X
X1	0
X2	1
X3	1

	W
W1	0
W2	1
W3	1

$\Theta = 1$ $w_{\Theta} = 0$

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



$W(i)_{t+1} = w(i)_t + \eta \;.\; E_t . x(i)$

$W(\Theta)_{t+1} = 0 + 1 \;.\; (0) \;.\; 1$

$W(\Theta)_{t+1} = 0 + 0$

	X
X1	?
X2	?
X3	?

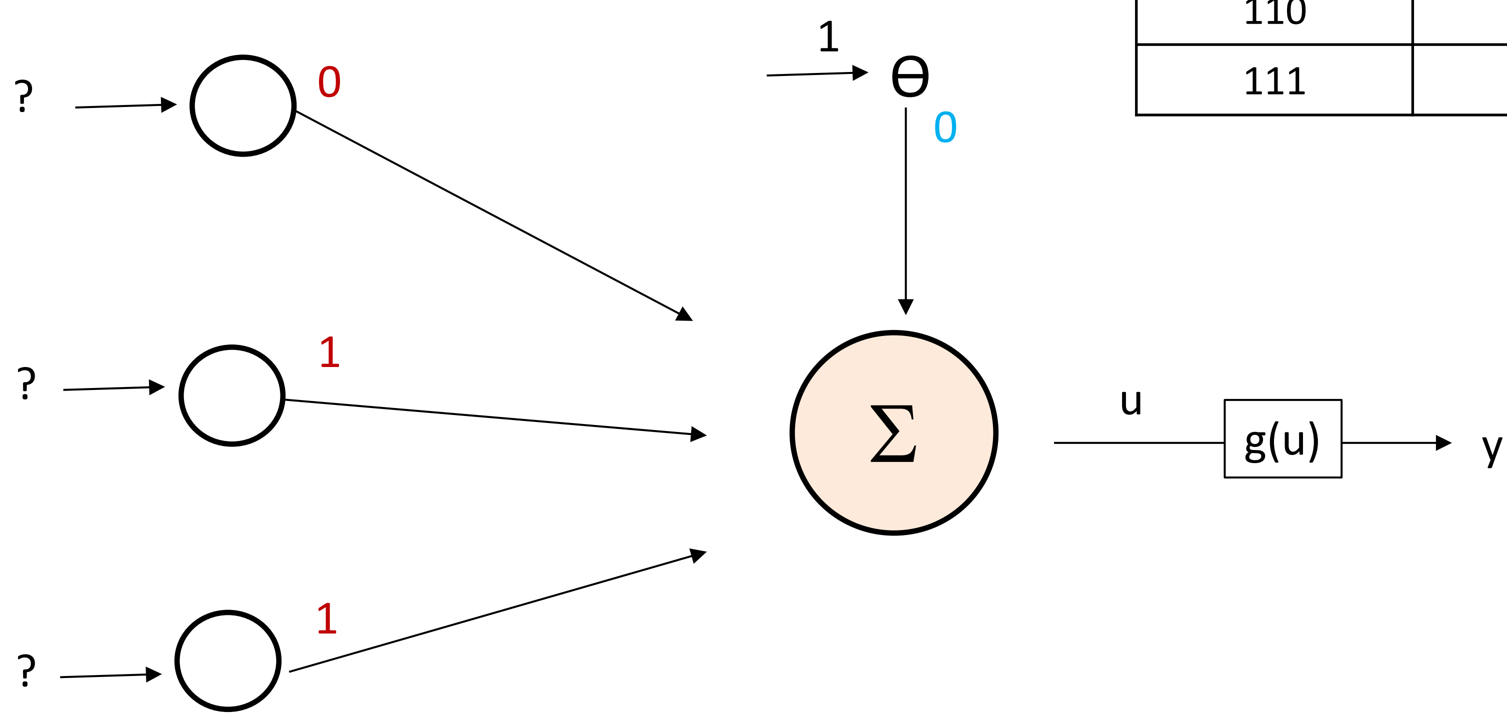
	W
W1	0
W2	1
W3	1

$\Theta = 1 \qquad w_\Theta = 0$

Outro exemplo

????

Atributo	Classe
000	0
001	0
010	1
011	1
100	0
101	0
110	1
111	1



Rede Perceptron

Processo de Treinamento

- η ➔ Constante da taxa de aprendizado ($0 < \eta < 1$)
- y ➔ Valor de saída produzida pelo *Perceptron*
- $d_{(k)}$ ➔ Valor desejado para k-ésima amostra de treinamento
- $x_{(k)}$ ➔ K-ésima amostra de treinamento
- w ➔ Vetor contendo os pesos (inicialmente gerados aleatoriamente)

$$\boxed{W(_k)_{t+1} = w(_k)_t + \eta \,.\, (d(_k) - y).x(_k)}$$

Rede Perceptron

Início {Algoritmo *Perceptron* – Fase de Treinamento}

<1> Obter o conjunto de amostras de treinamento { $\boldsymbol{x}^{(k)}$ };

<2> Associar a saída desejada { $d^{(k)}$ } para cada amostra obtida;

<3> Iniciar o vetor $\boldsymbol{w}$ com valores aleatórios pequenos;

<4> Especificar a taxa de aprendizagem {η};

<5> Iniciar o contador de número de épocas {*época* ← 0};

<6> Repetir as instruções:

<6.1> *erro* ← “inexiste”;

<6.2> Para todas as amostras de treinamento { $\boldsymbol{x}^{(k)}$, $d^{(k)}$ }, fazer:

<6.2.1> $u \leftarrow \boldsymbol{w}^T \cdot \boldsymbol{x}^{(k)}$;

<6.2.2> $y \leftarrow \text{sinal}(u)$;

<6.2.3> Se $y \neq d^{(k)}$

<6.2.3.1> Então $\begin{cases} \boldsymbol{w} \leftarrow \boldsymbol{w} + \eta \cdot (d^{(k)} - y) \cdot \boldsymbol{x}^{(k)} \\ erro \leftarrow \text{"existe"} \end{cases}$

<6.3> *época* ← *época* + 1;

Até que: *erro* = “inexiste”

Fim {Algoritmo *Perceptron* – Fase de Treinamento}

Rede Perceptron

Início {Algoritmo *Perceptron* – Fase de Operação}

<1> Obter uma amostra a ser classificada { $\boldsymbol{x}$ };

<2> Utilizar o vetor $\boldsymbol{w}$ ajustado durante o treinamento;

<3> Executar as seguintes instruções:

<3.1> $u \leftarrow \boldsymbol{w}^T \cdot \boldsymbol{x}$;

<3.2> $y \leftarrow \text{sinal}(u)$;

<3.3> Se $y = -1$

<3.3.1> Então: amostra $\boldsymbol{x} \in$ *{Classe A}*

<3.4> Se $y = 1$

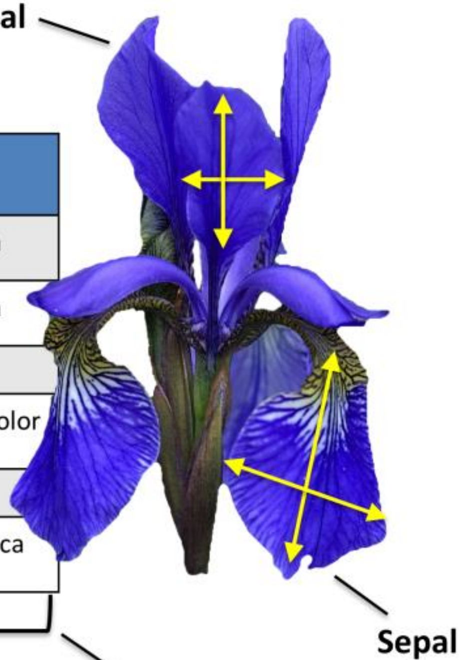
<3.4.1> Então: amostra $\boldsymbol{x} \in$ *{Classe B}*

Fim {Algoritmo *Perceptron* – Fase de Operação}

Iris flower data set

Samples
(instances, observations)

	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
...					
50	6.4	3.5	4.5	1.2	Versicolor
...					
150	5.9	3.0	5.0	1.8	Virginica

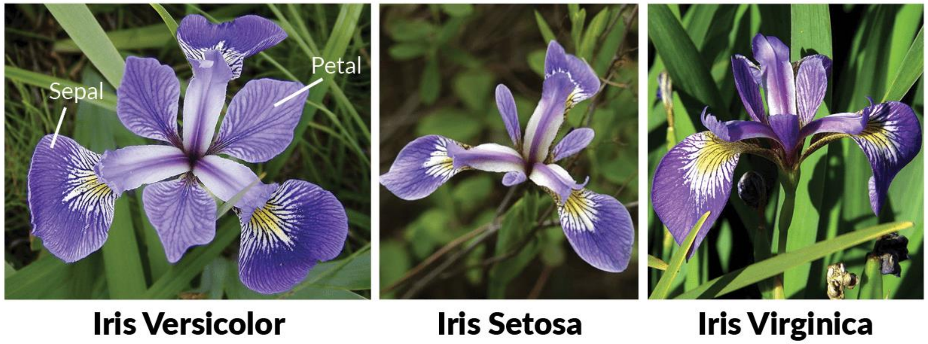


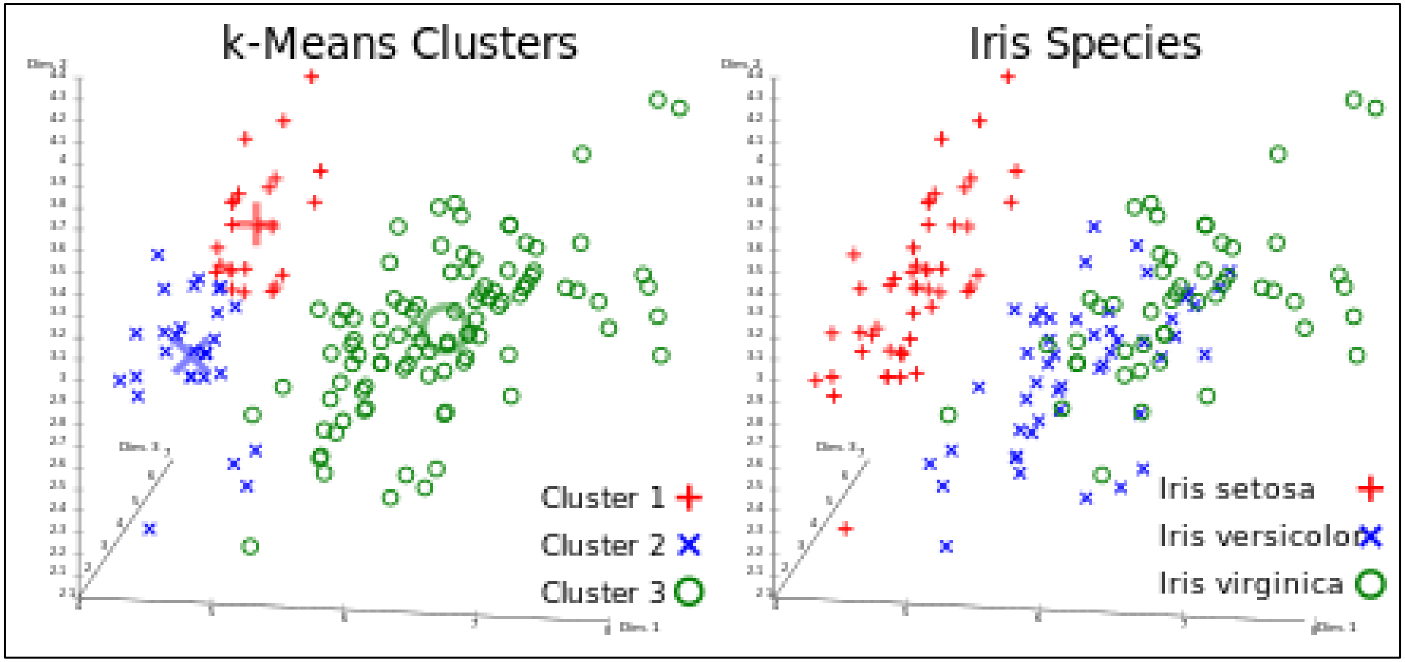
Features
(attributes, measurements, dimensions)

Class labels
(targets)

Iris Versicolor **Iris Setosa** **Iris Virginica**

Iris flower data set





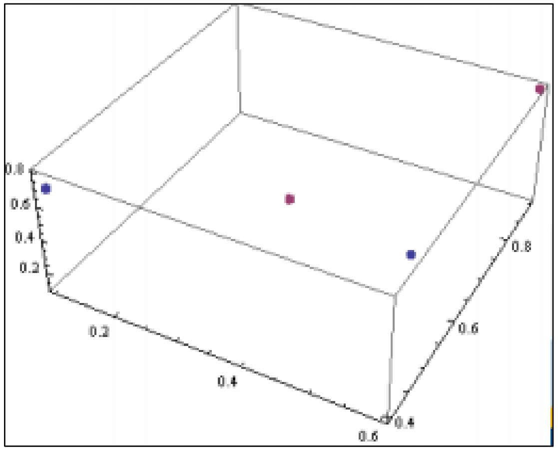
Rede Perceptron

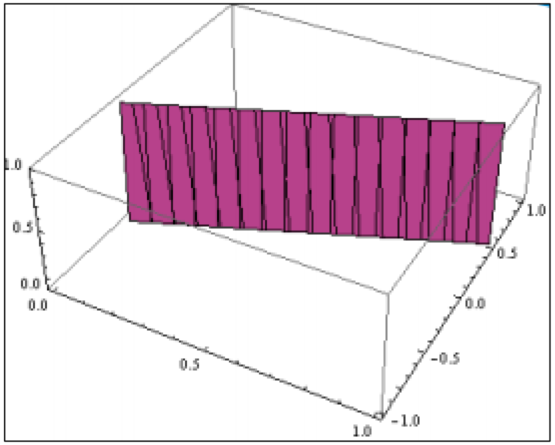
Exemplo de Execução

x1	x2	X3	**Classe** **1 = Setosa** **-1 = Versicolor**
0,1	0,4	0,7	1
0,5	0,7	0,1	1
0,6	0,9	0,8	-1
0,3	0,7	0,2	-1

Iris Setosa

Iris Versicolor

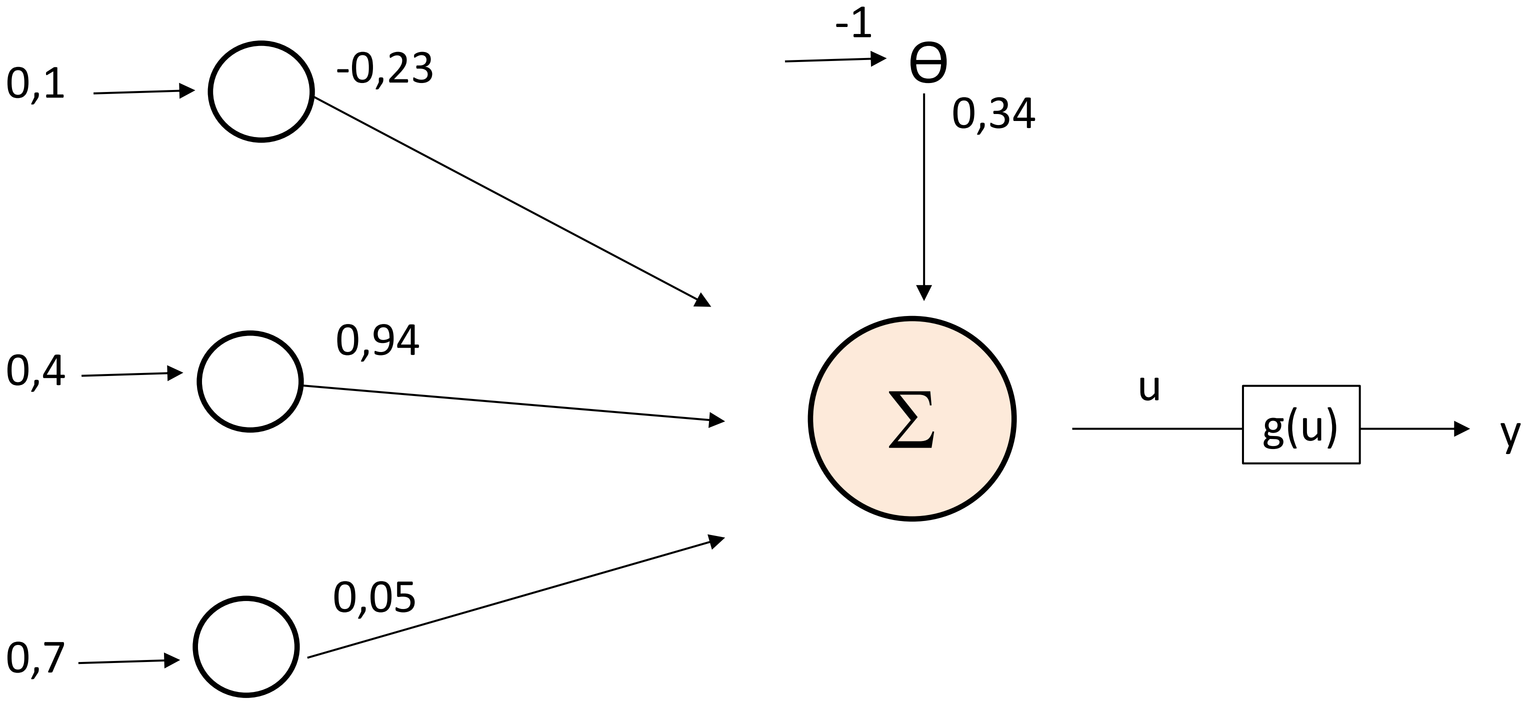




Rede Perceptron

Exemplo de Execução

η=0,05



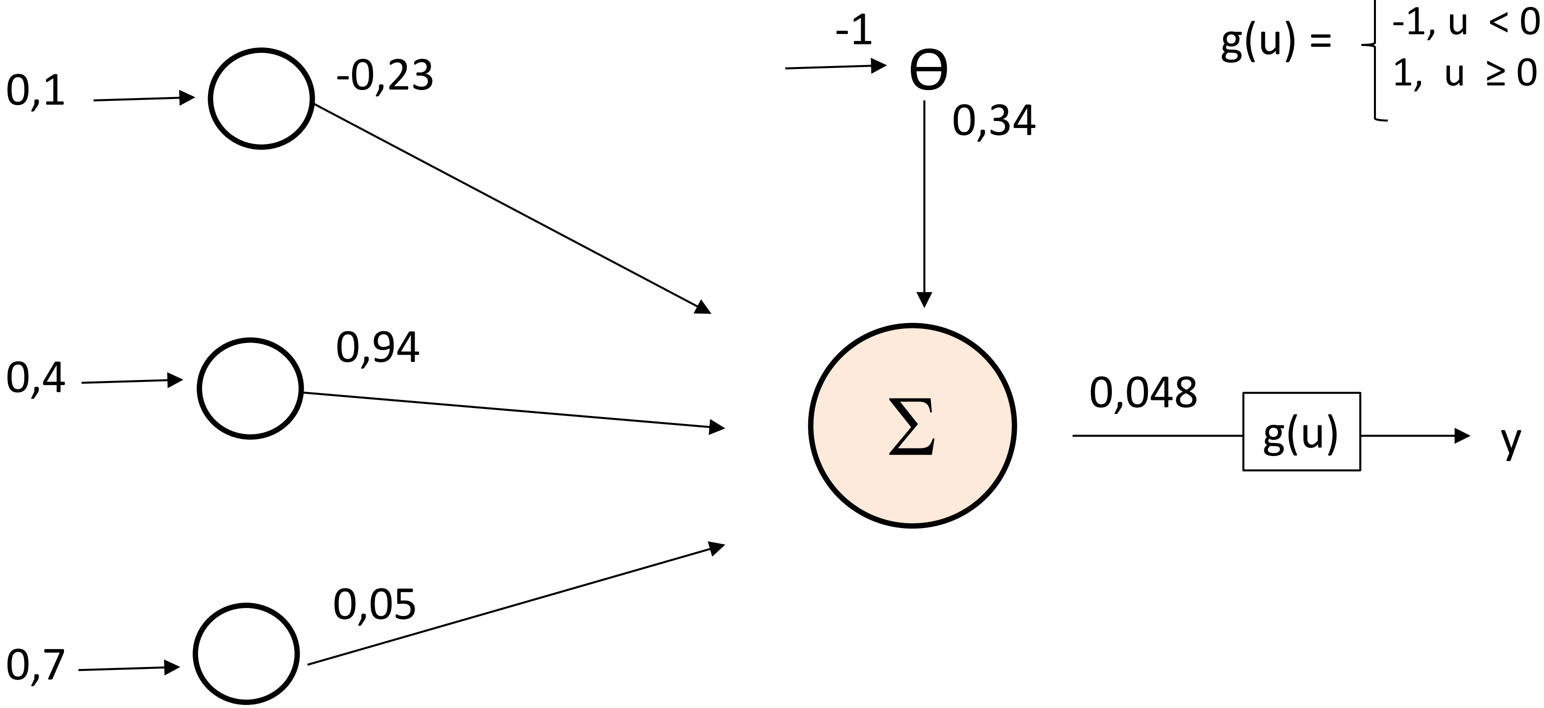
$$W(_k)_{t+1} = w(_k)_t + \eta \ . \ (d(_k) - y).x(_k)$$

Rede Perceptron

Exemplo de Execução

η=0,05

$$g(u) = \begin{cases} -1, & u < 0 \\ 1, & u \geq 0 \end{cases}$$

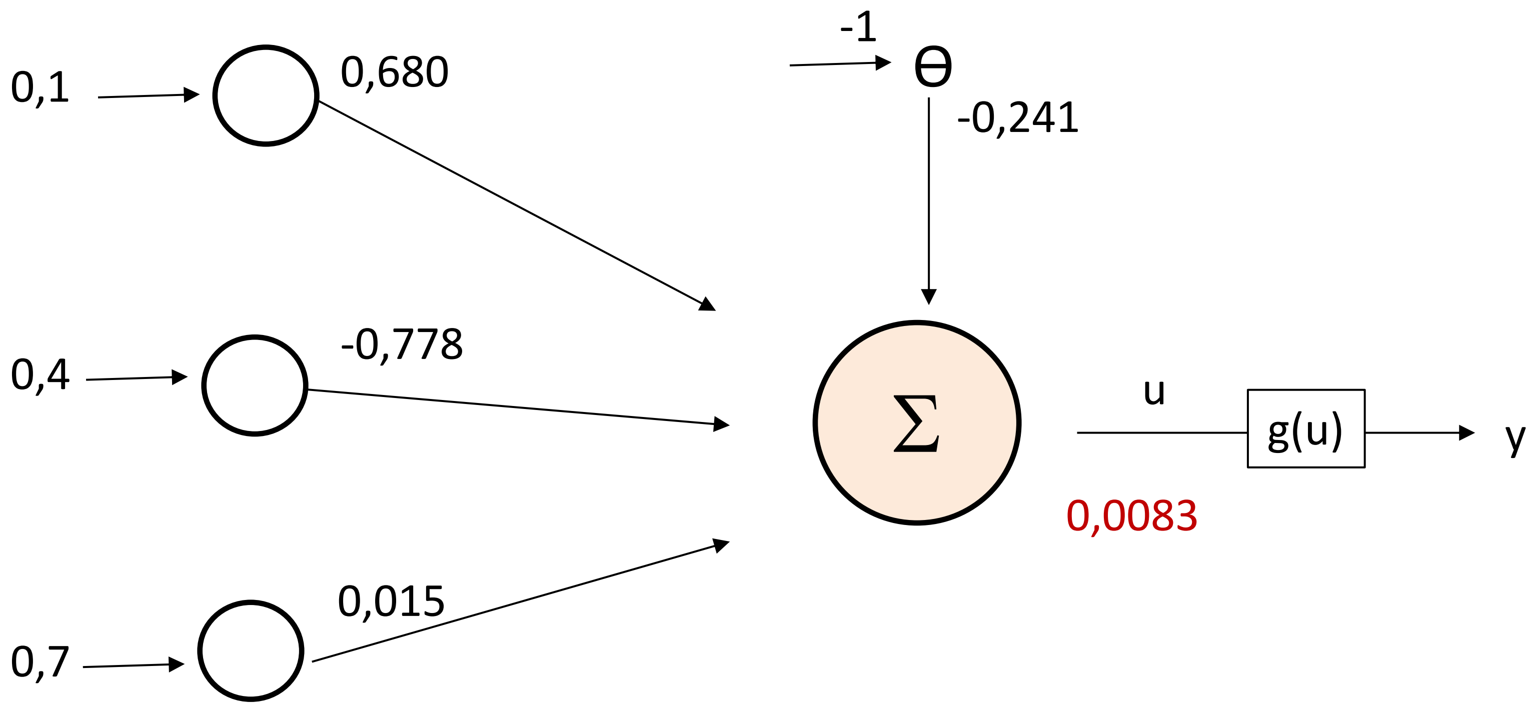


$$W(_k)_{t+1} = w(_k)_t + \eta \,.\, (d(_k) - y).x(_k)$$

Continuar ....

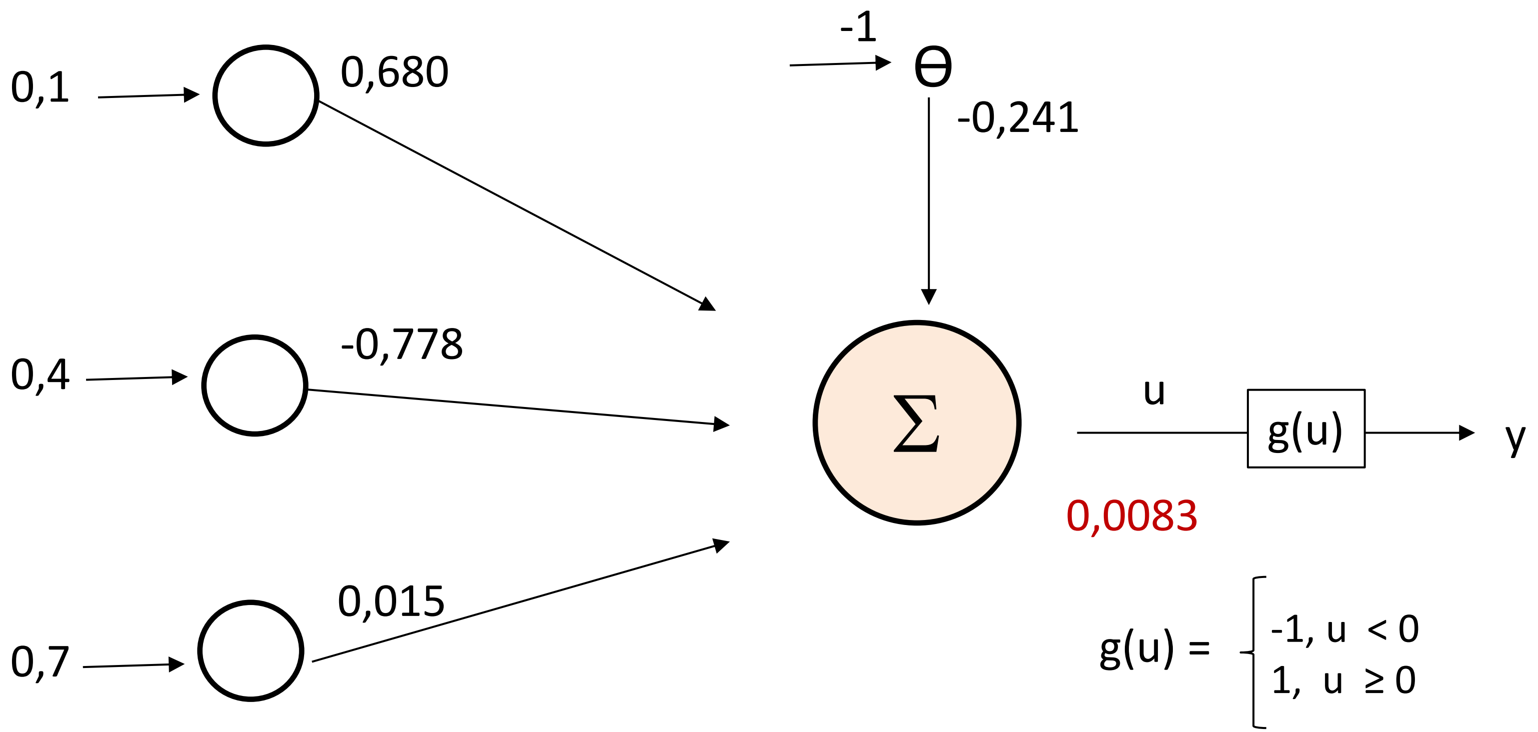
Rede Perceptron

Exemplo de Execução (pesos ajustados)



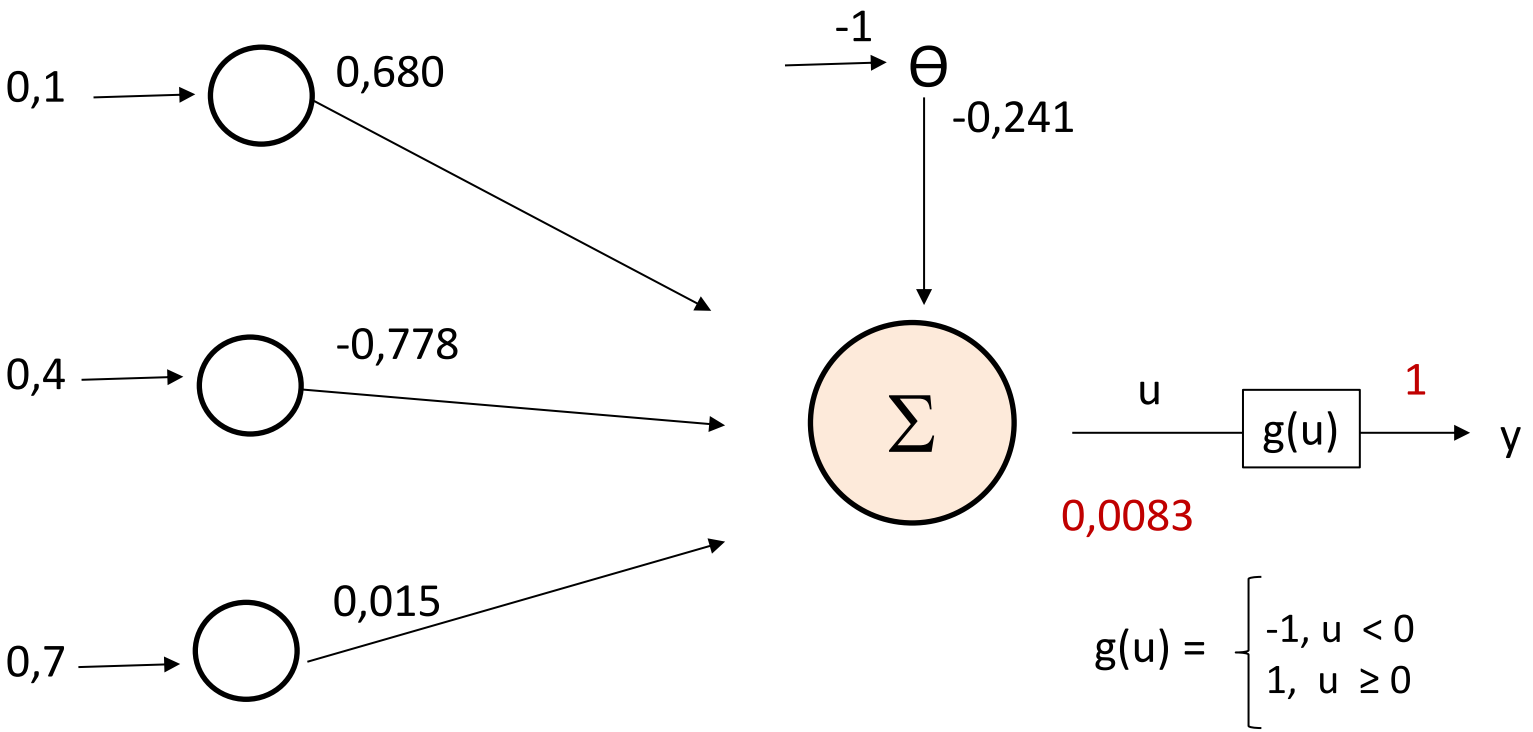
Rede Perceptron

Exemplo de Execução (pesos ajustados)



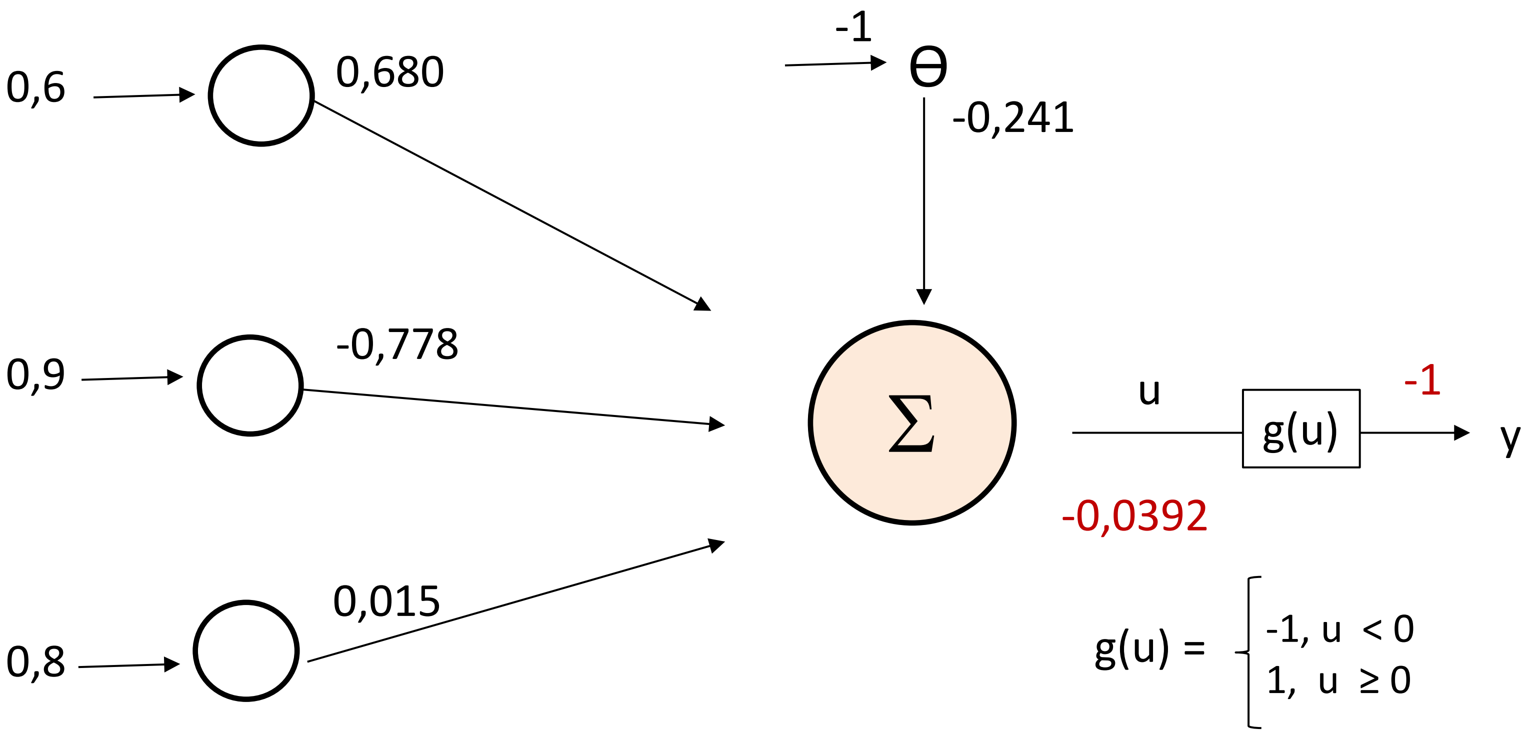
Rede Perceptron

Exemplo de Execução (pesos ajustados)



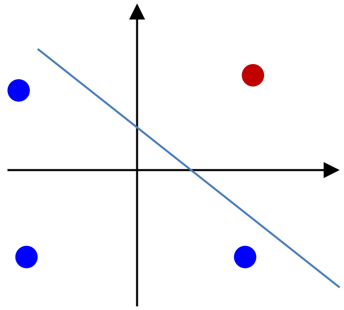
Rede Perceptron

Exemplo de Execução (pesos ajustados)

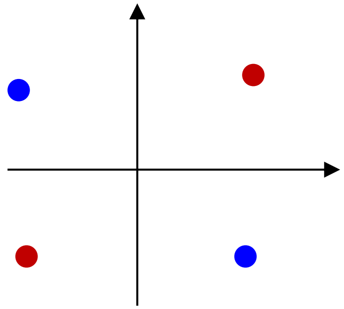


Limitações

- Um único Perceptron consegue resolver somente funções linearmente separáveis.



- Em funções não linearmente separáveis o perceptron não consegue gerar um hiperplano para separar os dados.



Redes Multicamadas

- Perceptrons expressam somente superfícies de decisão linear.

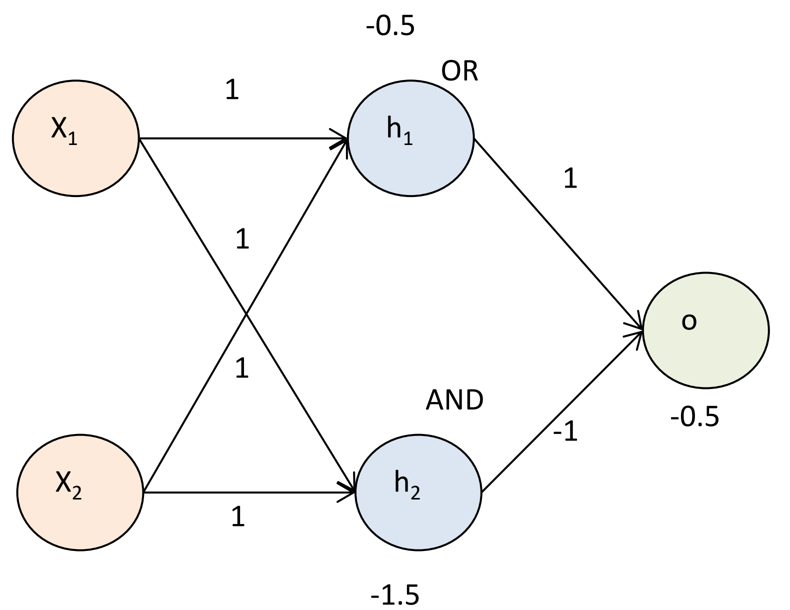
- Entretanto, é possível combinar vários perceptrons lineares para gerar superfícies de decisão mais complexas.

- Dessa forma podemos, por exemplo, gerar uma superfícies de classificação para o operador XOR.

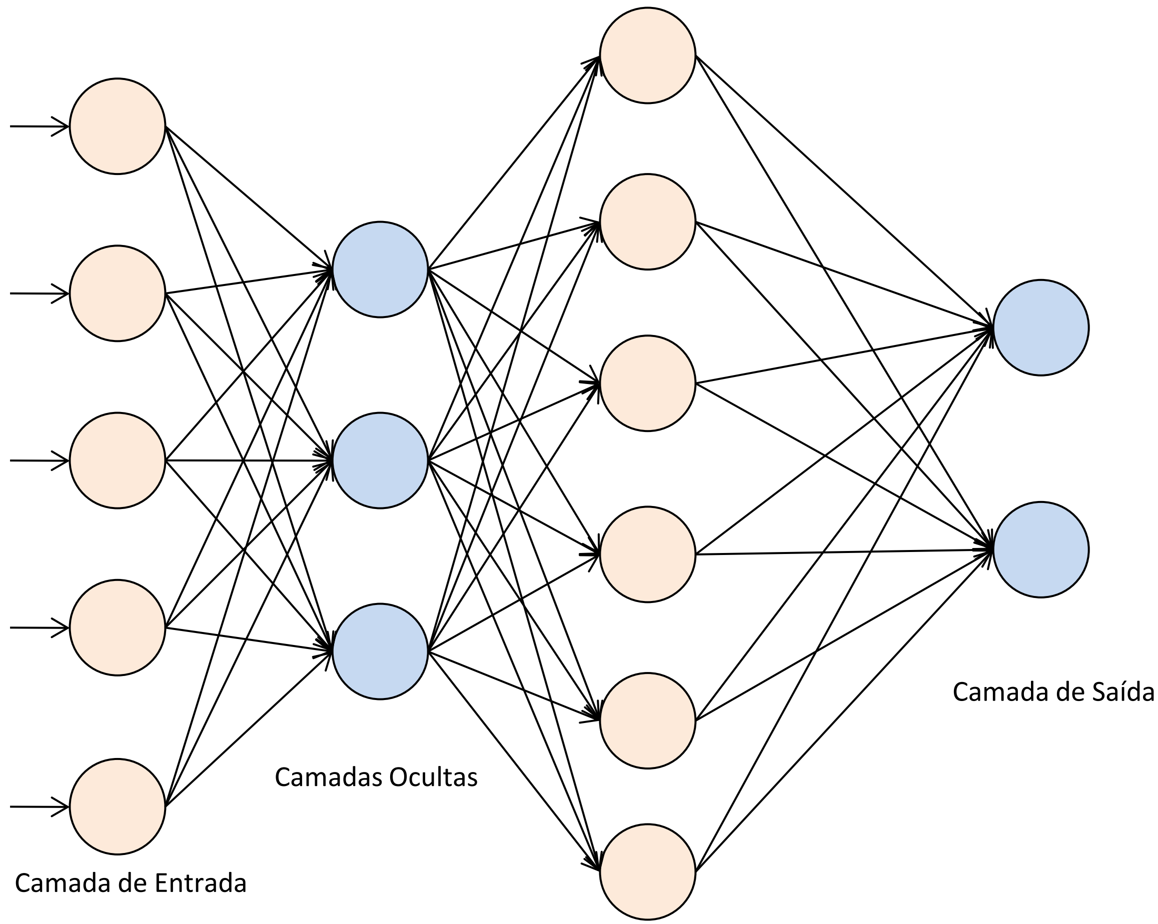
Operador XOR

Operador XOR

A	B	Saída
0	0	0
0	1	1
1	0	1
1	1	0

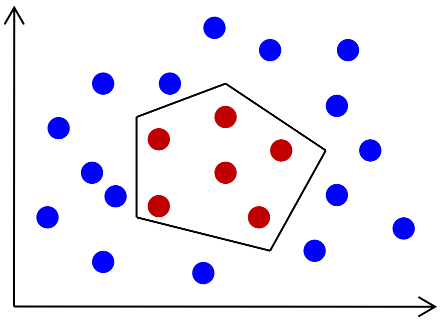


Redes Multicamadas

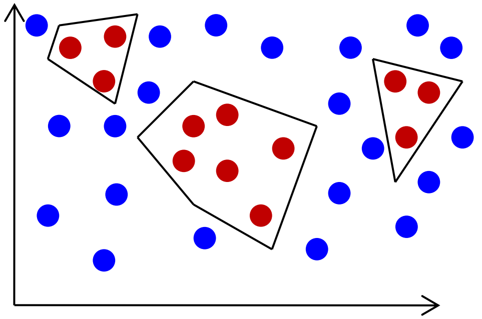


Redes Multicamadas

- Adicionar uma camada oculta a rede permite que a rede possa gerar uma função de convex hull.



- Duas camadas ocultas permite a rede gerar um função com diferentes convex hulls.



Redes Multicamadas

- **Unidades lineares** são capazes gerar **funções lineares**, dessa forma função de uma rede multicamada também será linear.

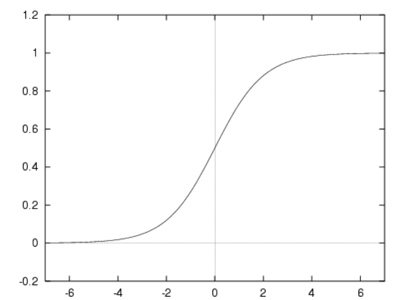
- Entretanto, existem muitas funções que **não podem ser modeladas por funções lineares**.

- Por esse motivo é necessário utilizar uma **outra função de ativação**.

Redes Multicamadas

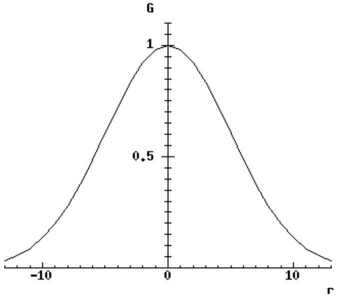
- Funções de ativação mais comuns:
  - Sigmoidal:

$$y = f\left(h = w_0 \cdot 1 + \sum_{i=1}^{n} w_i \cdot x_i ;\, p\right) = \frac{1}{1 + e^{-h/p}}$$



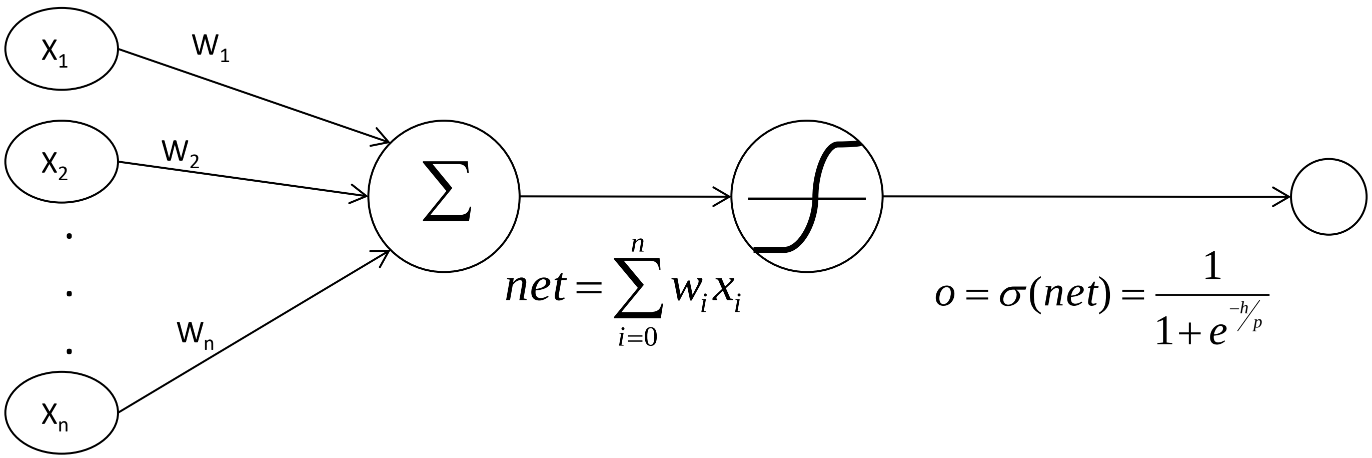
  - Radial (Gausiana):

$$y = f\left(h = \sum_{i=1}^{n} (x_i \cdot w_i)^2 ;\, \sigma = w_0\right) = \frac{1}{2\pi\sigma} e^{-\frac{h^2}{2\sigma^2}}$$



Redes Multicamadas

Sigmoid

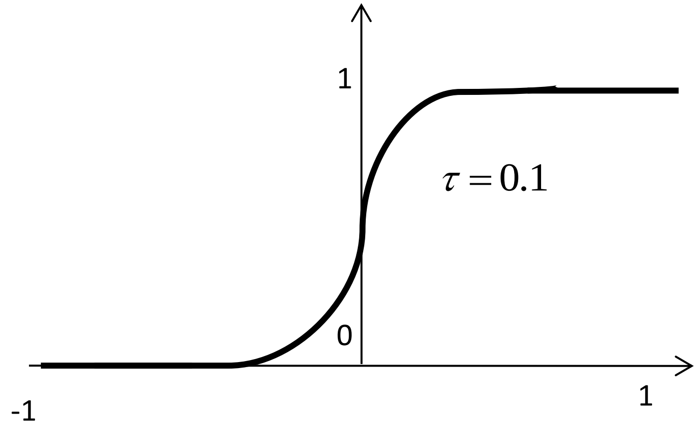


$$net = \sum_{i=0}^{n} w_i x_i$$

$$o = \sigma(net) = \frac{1}{1 + e^{-h/p}}$$

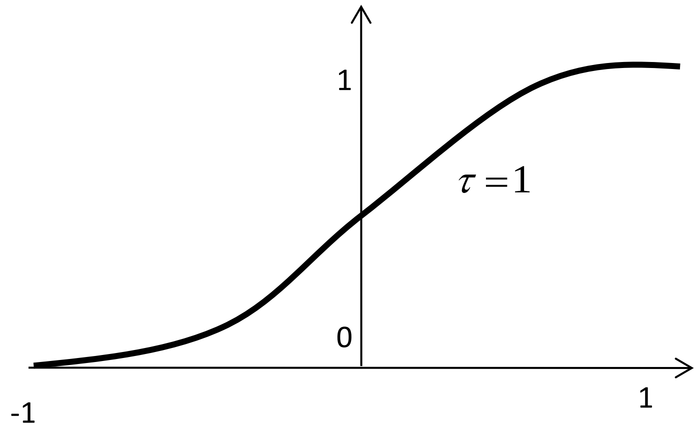
Função Sigmoidal

$$f_i(net_i(t)) = \frac{1}{1 + e^{-(net_i(t)-\alpha)/\tau}}$$



Função Sigmoidal

$$f_i(net_i(t)) = \frac{1}{1 + e^{-(net_i(t) - \alpha)/\tau}}$$



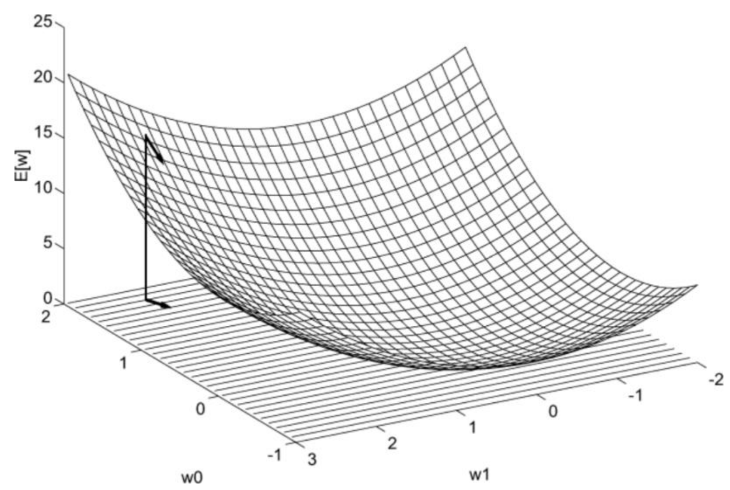
Backpropagation

- **Aprende os pesos para uma rede multicamadas**, dada uma rede com um número fixo de unidades e interconexões.

- O algoritmo backpropagation emprega a **descida do gradiente** para minimizar o erro quadrático entre a saída da rede e os valores alvos para estas saídas.

Descida do Gradiente

- A **descida do gradiente** busca determinar um vetor de pesos que minimiza o erro.



- Começando com um vetor inicial de **pesos arbitrário** e modificando-o repetidamente em pequenos passos.

- A cada passo, o vetor de pesos é alterado na direção que produz a maior queda ao longo da superfície de erro.

Backpropagation

- **Aprende os pesos para uma rede multicamadas**, dada uma rede com um número fixo de unidades e interconexões.

- O algoritmo backpropagation emprega a **descida do gradiente** para minimizar o erro quadrático entre a saída da rede e os valores alvos para estas saídas.

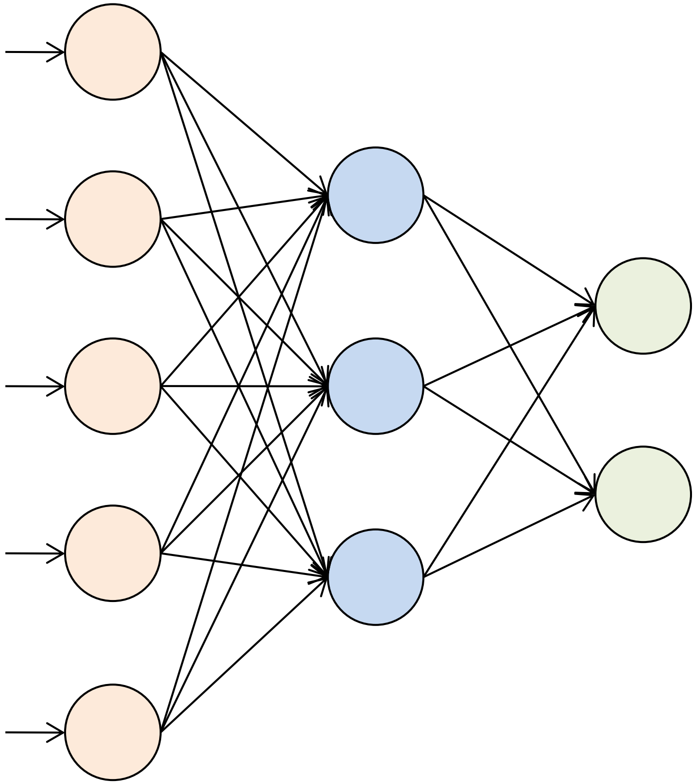
Valor de saida da rede → 0.119 0.059 0.253 0.246

0 0 1 0 ← Valor da saida buscada

Valor de saida da rede	0.119	0.059	0.253	0.246
Valor da saida buscada	0	0	1	0

Erro (E) = (Valor da saida buscada) – (Valor de saida da rede)

Backpropagation



Backpropagation

Inicializa cada peso w_i com um pequeno valor randômico.

Enquanto condição de parada não for atingida **faça**

{

Para cada exemplo de treinamento **faça**

{

Entre com os dados do exemplo na rede e calcule a saída da rede (o_k)

Para cada unidade de saída k **faça**

{

$$\delta_k \leftarrow o_k(1-o_k)(t_k - o_k)$$

}

Para cada unidade oculta h **faça**

{

$$\delta_h \leftarrow o_h(1-o_h) \sum_{k \in outputs} w_{h,k}\delta_k$$

}

Para cada peso w_j da rede **faça**

{

$$w_{i,j} \leftarrow w_{i,j} + \Delta w_{i,j}$$
$$where \quad \Delta w_{i,j} = \eta \delta_j x_{i,j}$$

}

}

}

Backpropagation

- O backpropagation **não é um algoritmo ótimo** e não garante sempre a melhor resposta.

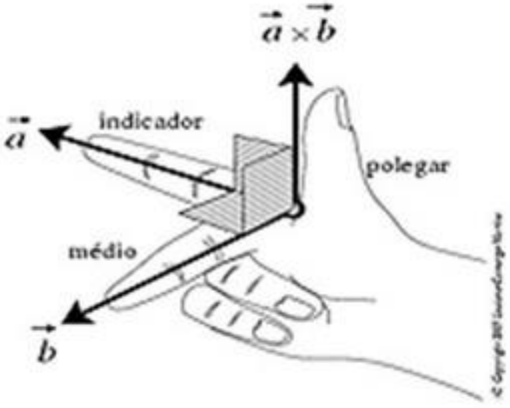
- O algoritmo de descida do gradiente pode ficar preso em um erro **mínimo local**.

- É possível refazer o treinamento variando os valores iniciais dos pesos.

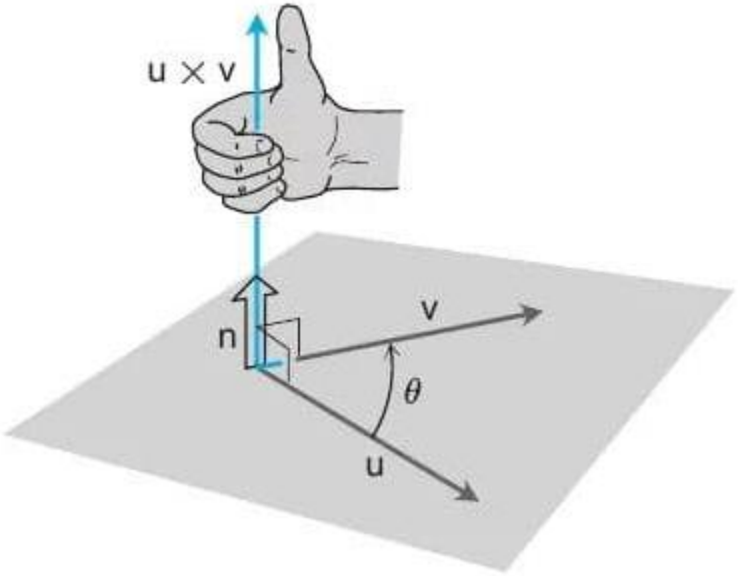
- Backpropagation é o algoritmo de aprendizagem mais comum, porém existem muitos outros.

Revisão de conceitos de álgebra linear

Regra da mão direita



$$\boxed{\vec{c} = \vec{a} \times \vec{b}}$$

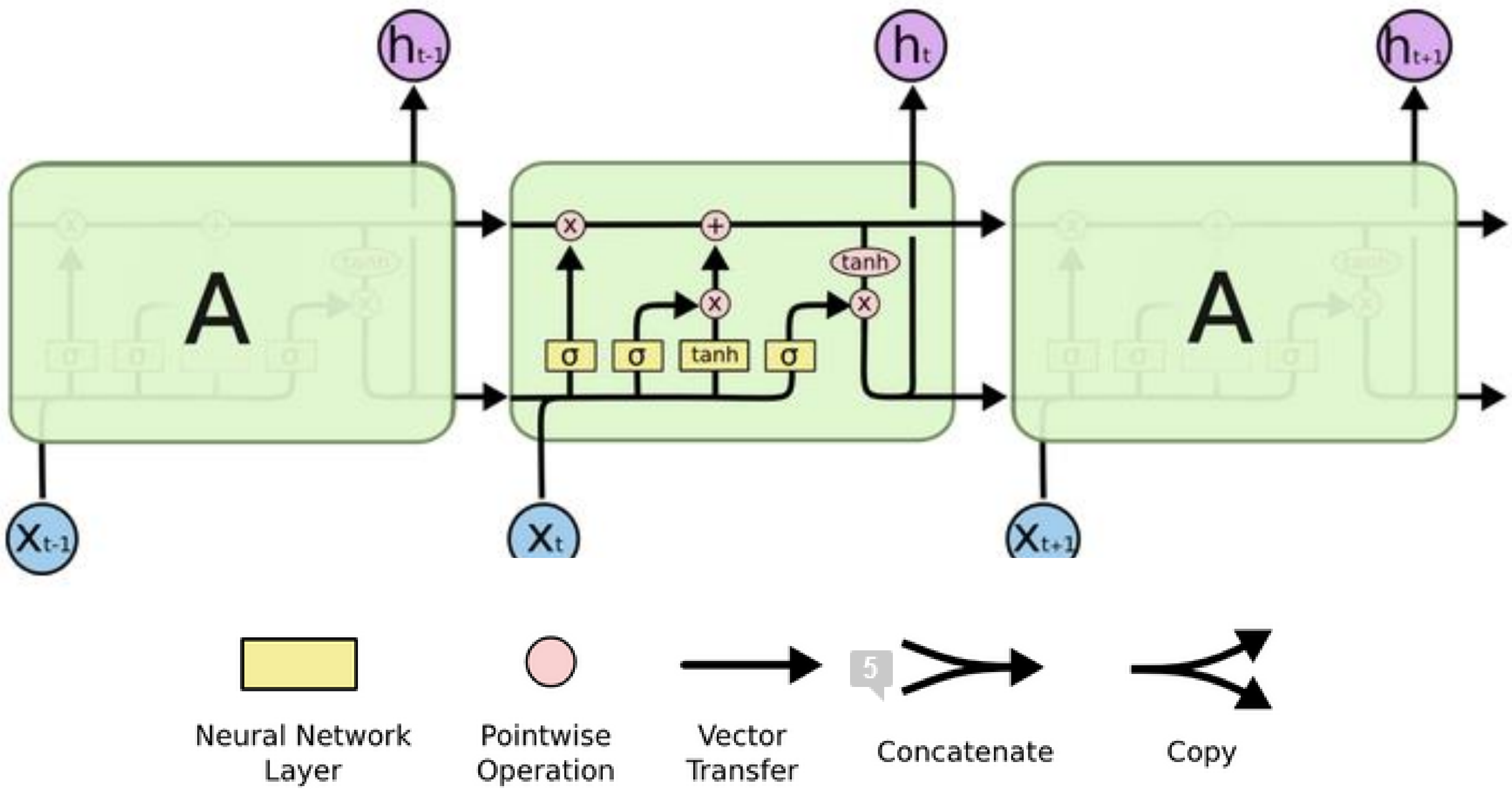


Produto de hadamard

$$\begin{bmatrix} 3 & 5 & 7 \\ 4 & 9 & 8 \end{bmatrix} * \begin{bmatrix} 1 & 6 & 3 \\ 0 & 2 & 9 \end{bmatrix} = \begin{bmatrix} 3\times1 & 5\times6 & 7\times3 \\ 4\times0 & 9\times2 & 8\times9 \end{bmatrix}$$

Long Short Term Memory - LSTM

Hochreiter & Schmidhuber (1997)



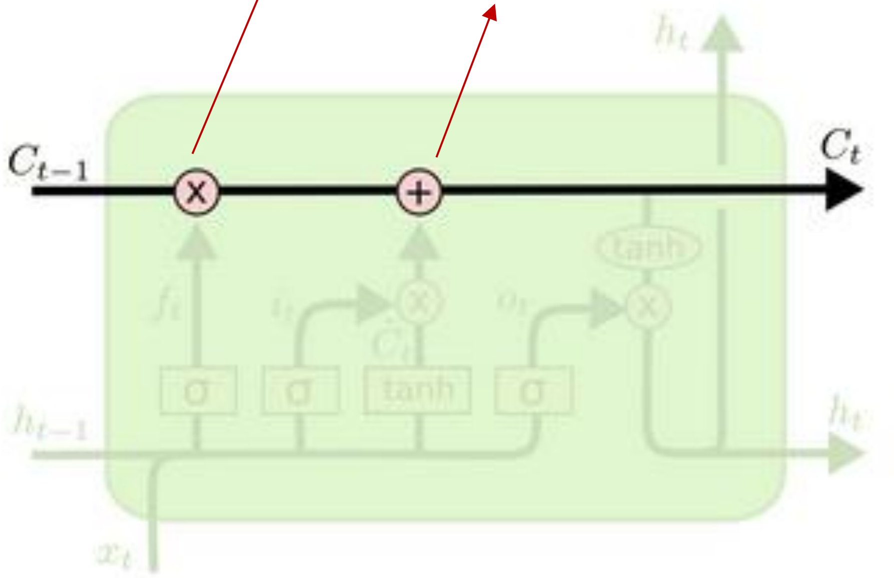
http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Long Short Term Memory - LSTM

[Hochreiter & Schmidhuber (1997)]

Fator de atenuação (esquecimento), n-dimensões

O que se quer lembrar da entrada atual



Canal é o que entra, o que a gente decide que lembra (x), soma o que entrou e sai..

A LSTM cria um cell state onde informações podem fluir de um estado anterior ao próximo, gerando uma espécie de memória na rede

Long Short Term Memory - LSTM

[Hochreiter & Schmidhuber (1997)]

A LSTM possui 3 gates principais

- Forget gate
- Input gate
- Output gate

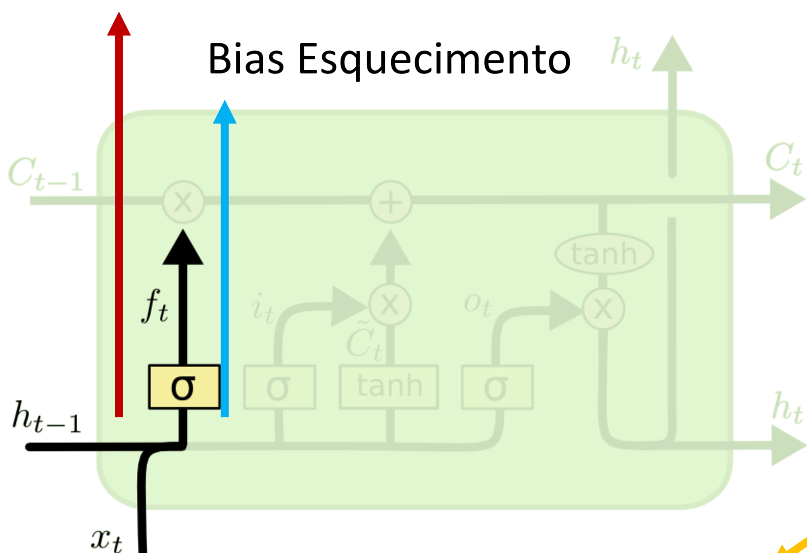
Long Short Term Memory - LSTM

[Hochreiter & Schmidhuber (1997)](#)

O que devo esquecer?

Esquecimento

Bias Esquecimento



Hidden anterior

Entrada atual

Vetor esquecimento

Bias Esquecimento

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right)$$

Vetor Hidden anterior ∪ Vetor entrada atual

Sigmoid aplicada a n-dimensões decide o quanto deve ser esquecido, ou seja, valores próximos de 1 serão mantidos e próximos a 0 serão descartados

Long Short Term Memory - LSTM

[Hochreiter & Schmidhuber (1997)](#)

Sigmoid aplicada a n-dimensões decide o quanto deve ser esquecido, ou seja, valores próximos de 1 serão mantidos e próximos a 0 serão descartados

Bias Esquecimento

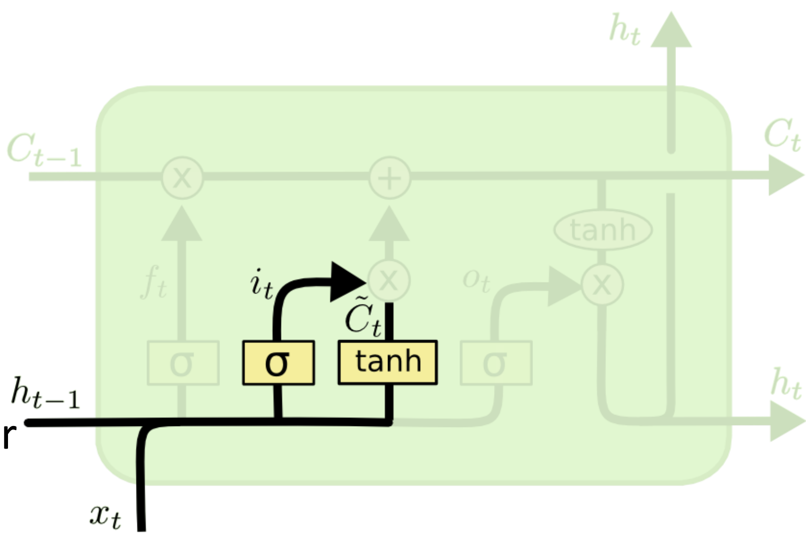
Vetor esquecimento

Vetor Hidden anterior

$\cup$

Vetor entrada atual

Hidden anterior



Entrada atual

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right)$$

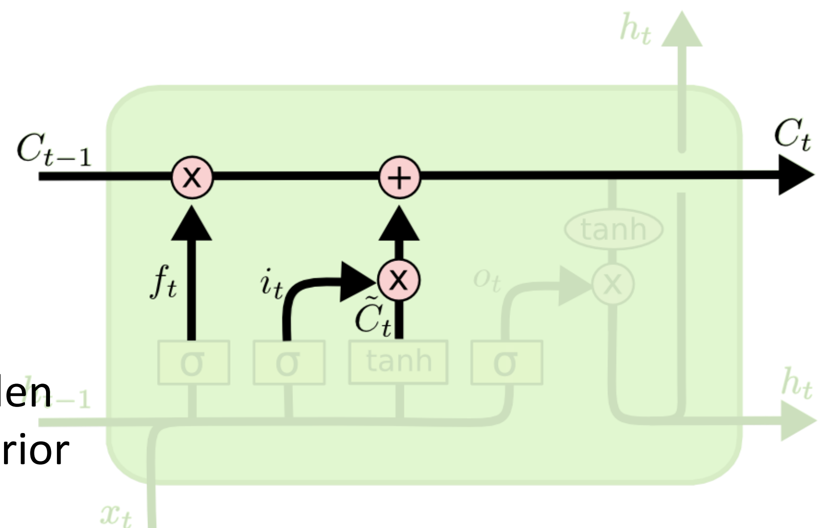
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

C_t calcula candidatos a serem armazenados no cell state

- Input gate decide quais desses valores candidatos devem ser atualizados no cell state

Long Short Term Memory - LSTM

[Hochreiter & Schmidhuber (1997)]



Hidden anterior

Entrada atual

Vetor esquecimento

Modulador esquecimento (+/- importante)

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Entrada

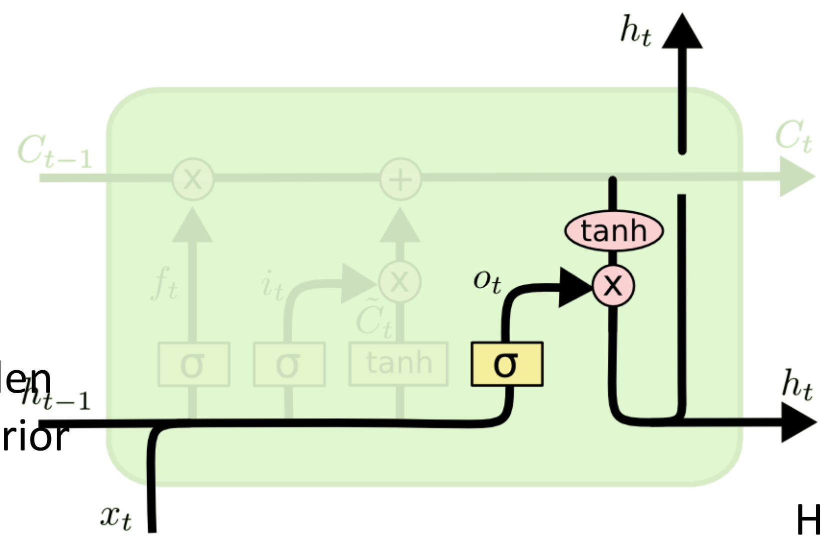
C_t saída que será lembrada no próximo estado

- Cell state – os valores "selecionados" do cell state anterior e do candidato serão combinados para formar o novo cell state

Long Short Term Memory - LSTM

[Hochreiter & Schmidhuber (1997)]

- Novamente uma sigmoide, "seleciona" os elementos do cell state que serão utilizados como output



Hidden anterior

Entrada atual

Vetor esquecimento

Bias Esquecimento

Vetor Hidden anterior
∪
Vetor entrada atual

$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

Hidden state

C_t o canal pode indicar o que pode ser esquecido