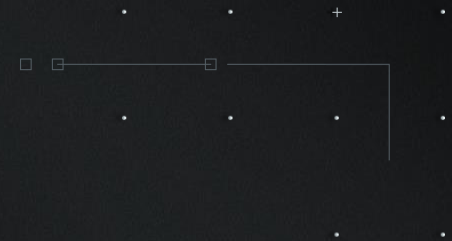




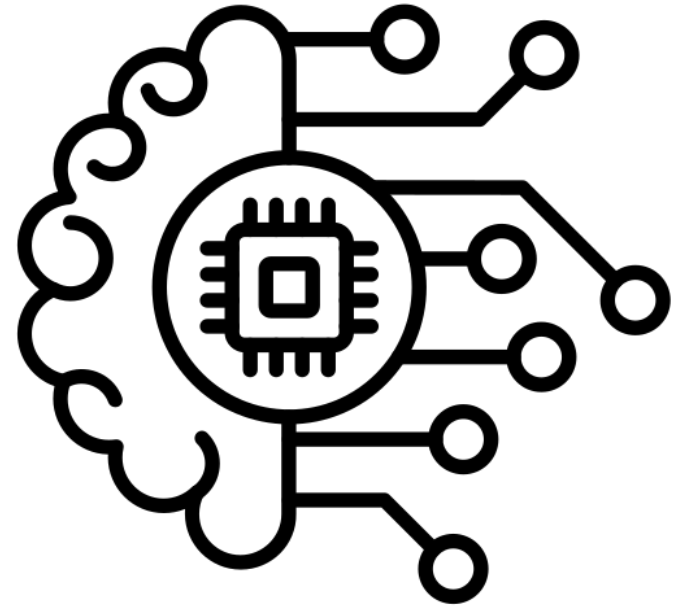
FIAP

– Deep Learning – Introdução a Redes Neurais Artificiais e o Modelo Perceptron

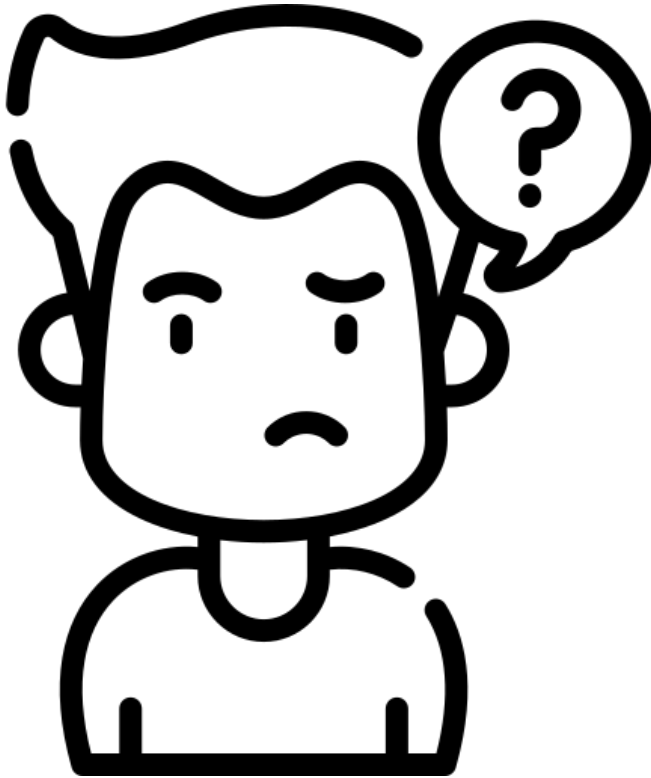


O que é uma Rede Neural Artificial? FIAP

- É um modelo computacional inspirado no funcionamento do cérebro humano;
- É uma técnica de Inteligência Artificial utilizada para tarefas de classificação, reconhecimento de padrões, previsão de valores, entre outras;
- É um conjunto de neurônios artificiais (nós), organizados em camadas, que processam informações por meio de operações matemáticas (funções de ativação);



Quando usar uma rede Neural?



Quando **NÃO** existe um algoritmo que modelo o sistema ou o problema em questão.

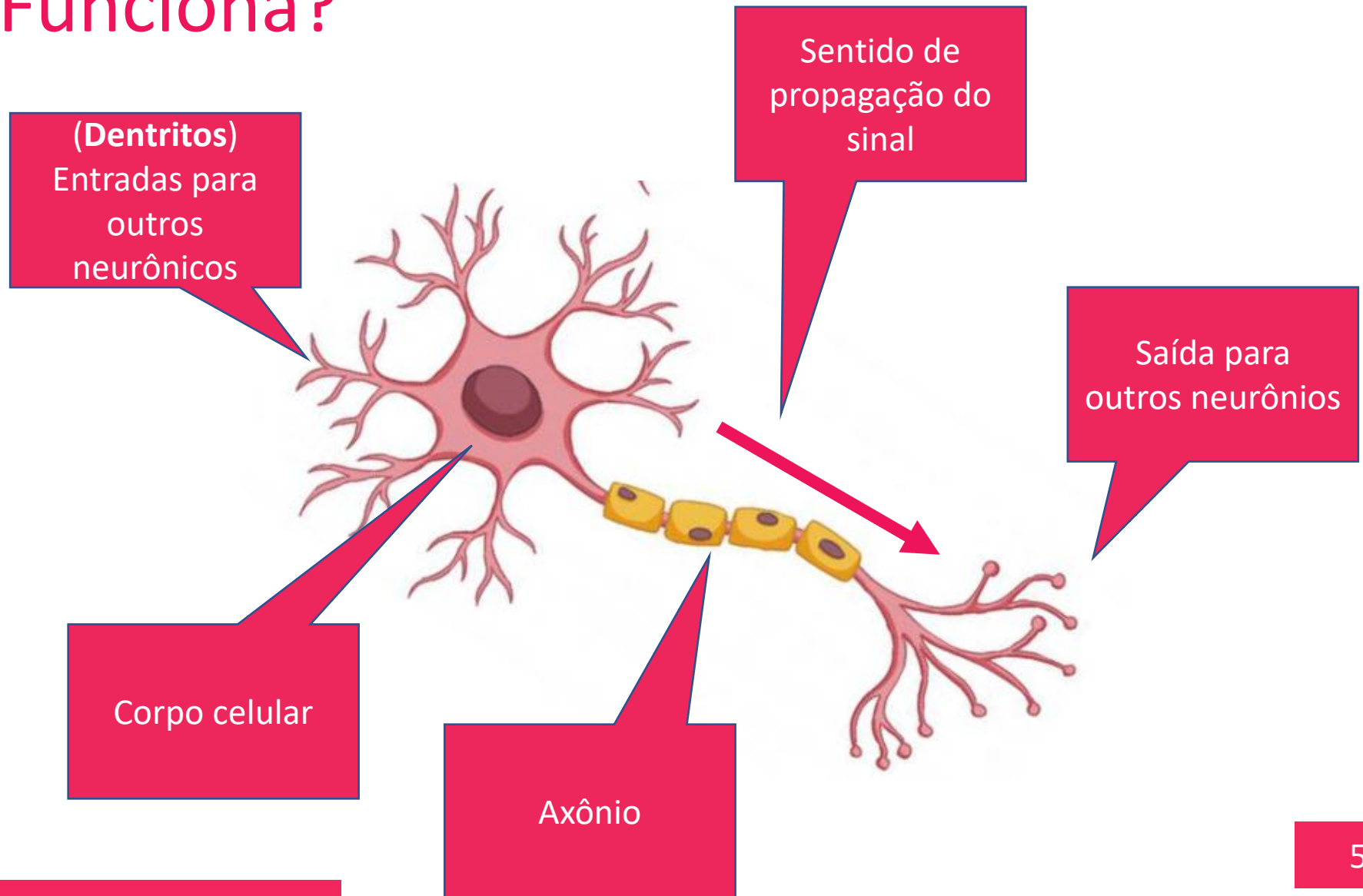


Quando existem **MUITOS** dados para serem analisados



Quando o seu problema é considerado **COMPLEXO**

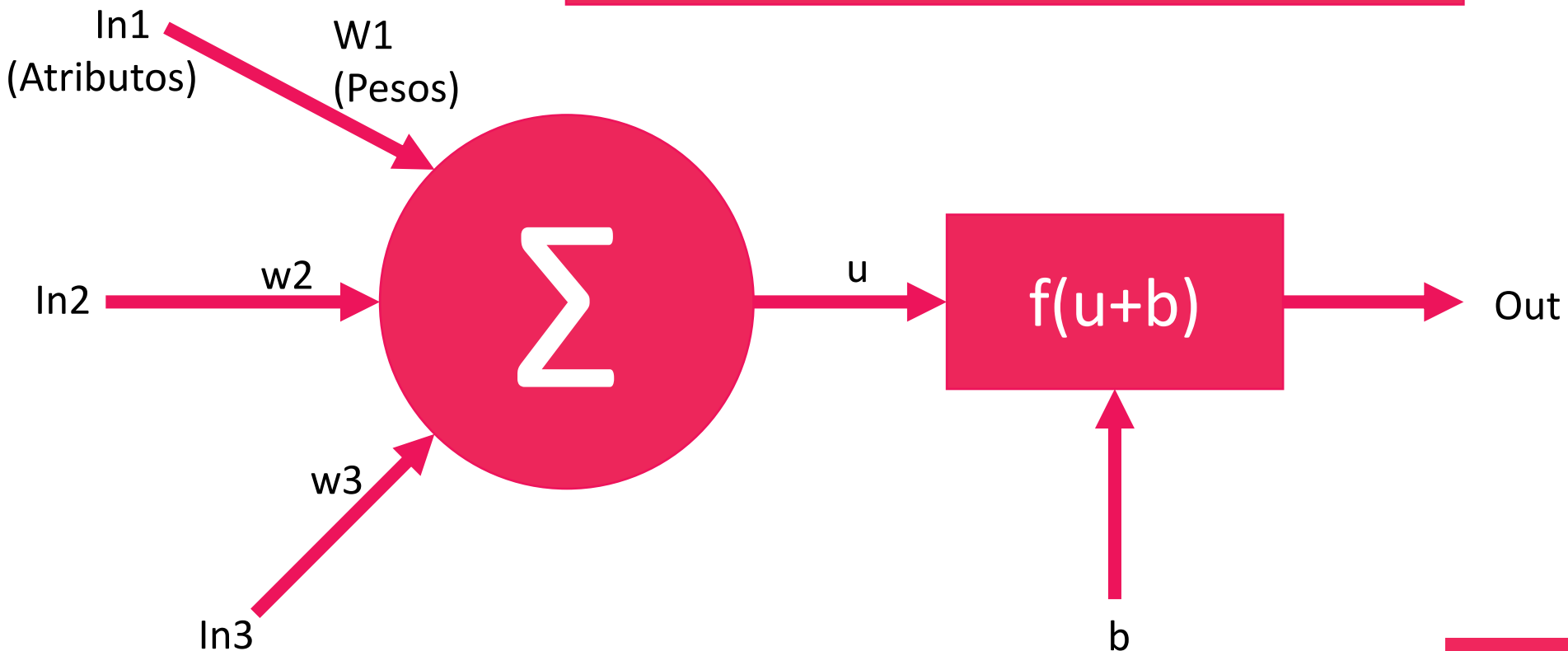
Como uma Rede Neural Artificial Funciona?



Como uma Rede Neural Artificial Funciona?

FIA/P

$$Out = f(\sum (In_i \times w_i) + b)$$



Função de Ativação do Neurônio

Degrau

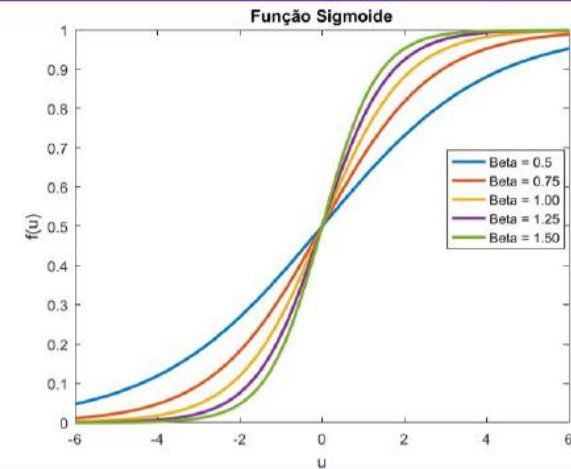
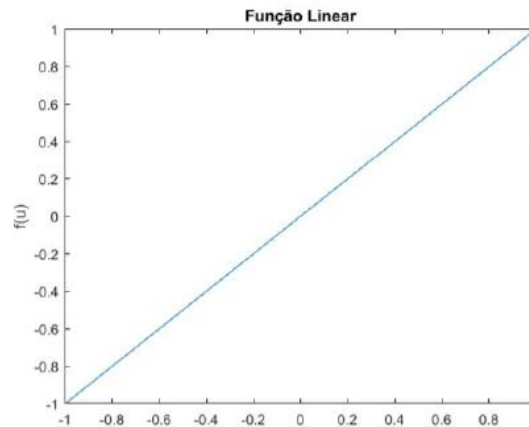
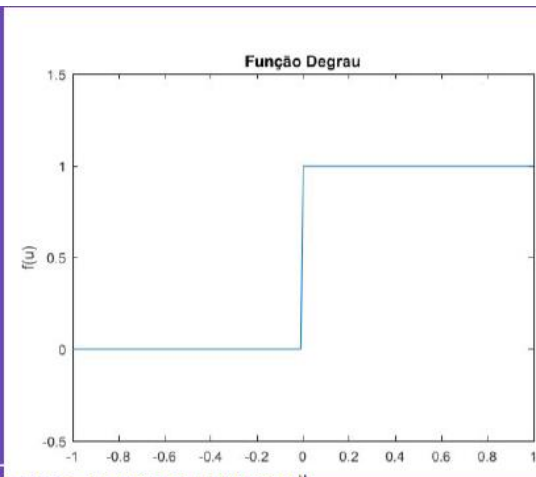
$$f(u) = \begin{cases} 0, & u < 1 \\ 1, & u \geq 1 \end{cases}$$

Linear

$$f(u) = u$$

Sigmóide

$$f(u) = \frac{1}{(1 + e^{-\beta u})}$$



Exercício 01

- Dado uma rede neural de uma camada cujas entradas são 1, 2 e 3, e os pesos são 0.1, 0.2 e 0.3, calcule a saída deste perceptron considerando uma função de ativação degrau e bias = 0.

Entradas:

$$In1 = 1$$

$$In2 = 2$$

$$In3 = 3$$

Pesos:

$$w1 = 0.1$$

$$w2 = 0.2$$

$$w3 = 0.3$$

bias:

$$b = 0$$

$$Out = f(u + b)$$



$$Out = f\left(\sum (In_i \times w_i) + b\right)$$

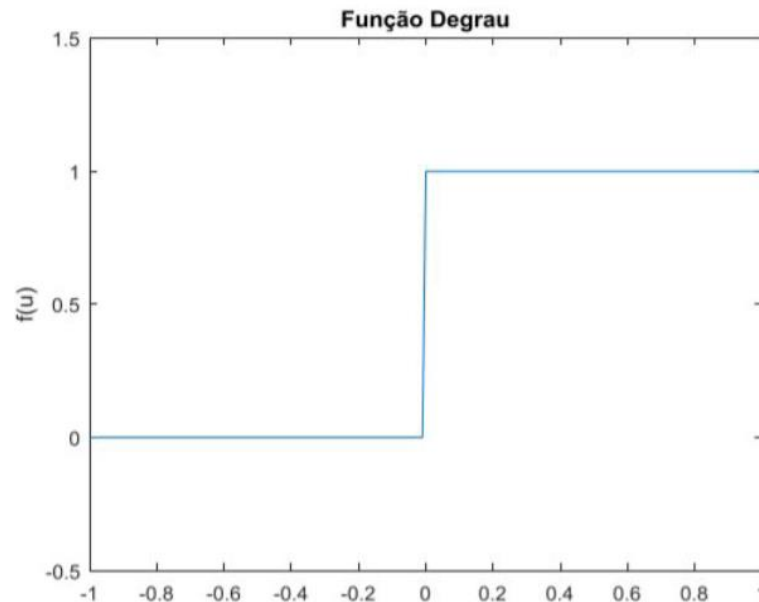
Exercício 01

$$u = 1 * 0.1 + 2 * 0.2 + 3 * 0.3 \therefore u = 1.4$$

$$u + b = 1.4 + 0 \therefore u + b = 1.4$$

Degrau

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



$$f(1.4) = 1$$

Exercício 02

- Dado uma rede neural de uma camada cujas entradas são -1.0, 5.0 e 0.2, e os pesos são 0.11, -0.2 e 4.0, calcule a saída deste perceptron considerando uma função de ativação sigmóide e $\beta = 1.25$.

Entradas:

$In_1 = -1.0$

$In_2 = 5.0$

$In_3 = 0.2$

Pesos:

$w_1 = 0.11$

$w_2 = -0.2$

$w_3 = 4.0$

$$Out = f(u + b)$$



$$Out = f\left(\sum (In_i \times w_i) + b\right)$$

$$u = -1 * 0.11 + 5 * -0.2 + 0.2 * 4 \therefore u = -0.31$$

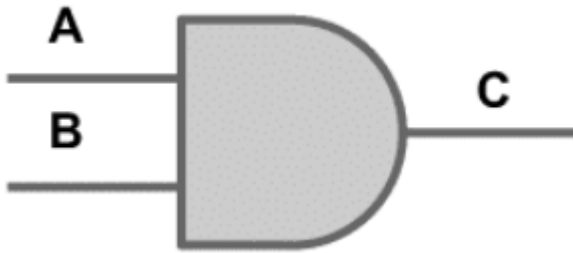
Sigmóide

$$f(u) = \frac{1}{(1 + e^{-\beta u})}$$

$$Out = f(u + b) = \frac{1}{1 + e^{-1,25 * (-0.31)}} \cong 0,40432$$

Ajuste de pesos – Treino da Rede Neural!

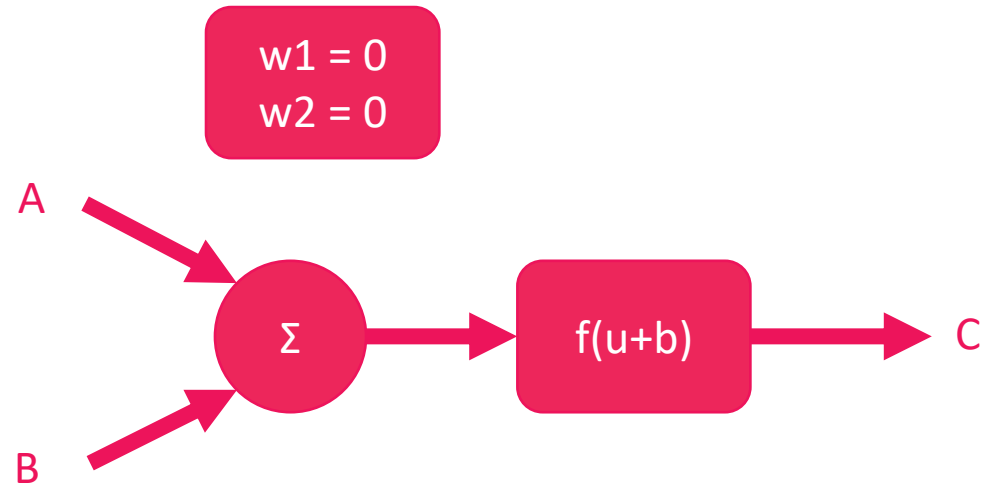
Vamos implementar uma porta lógica tipo AND usando Rede Neural!



A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

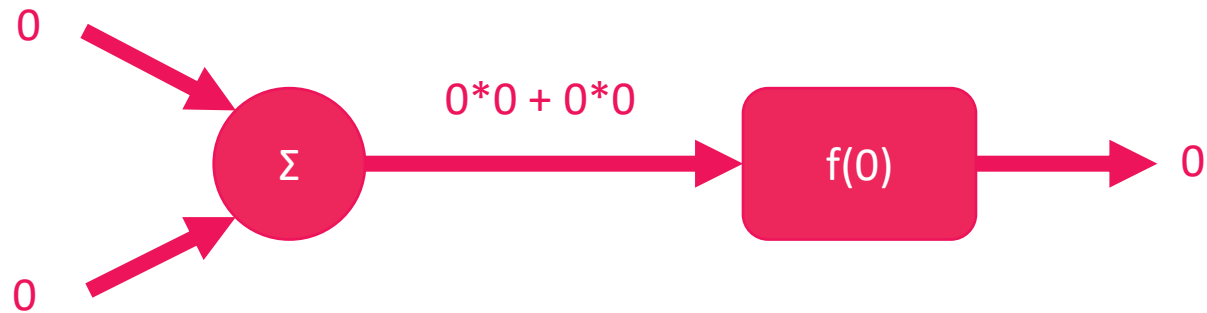
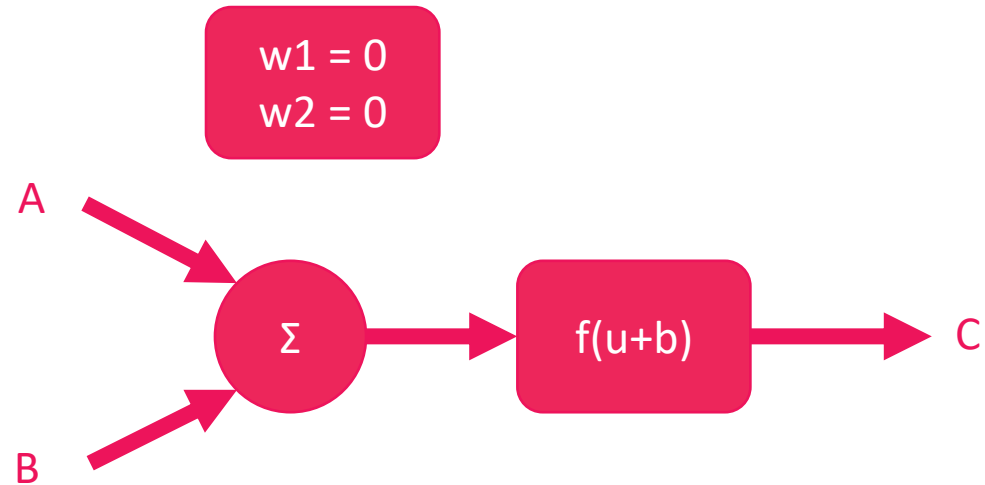
Ajuste de pesos – Treino da Rede Neural!

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1



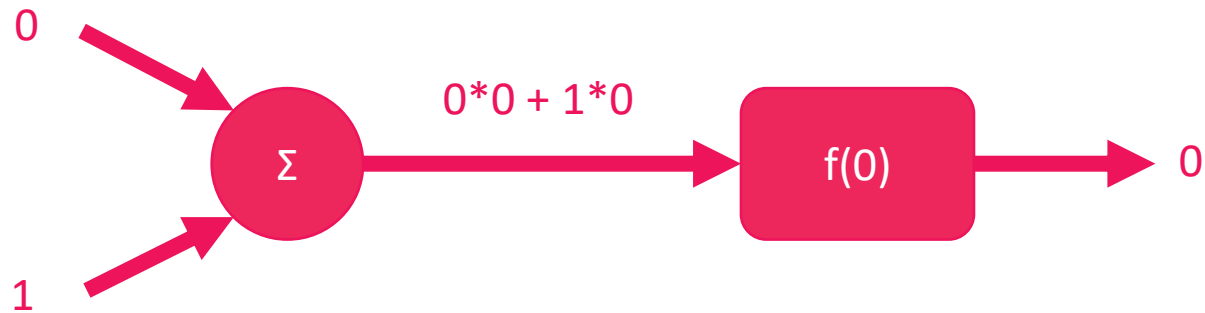
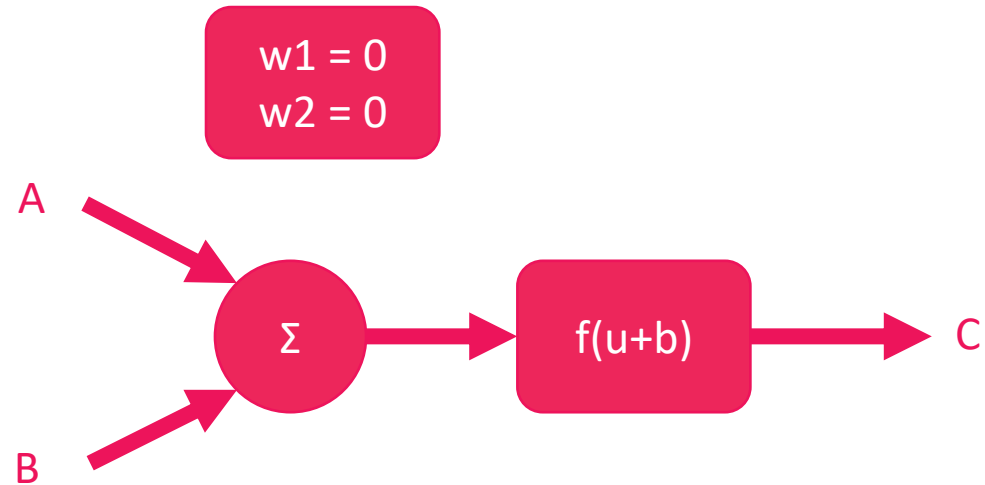
Ajuste de pesos – Treino da Rede Neural!

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1



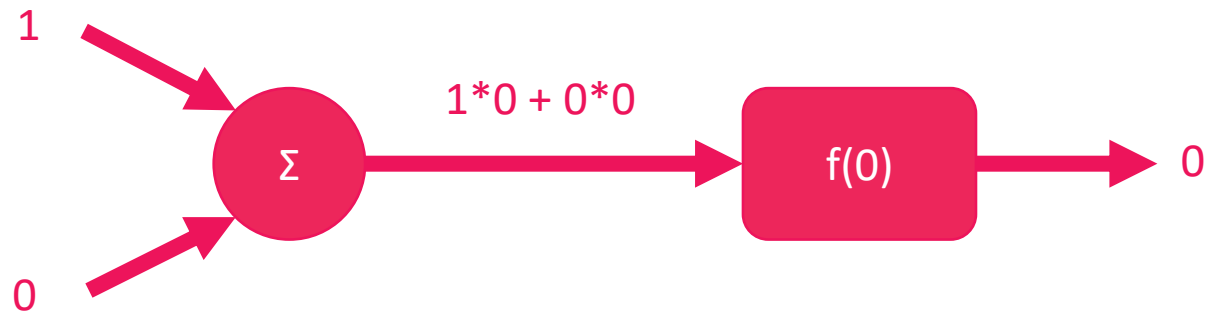
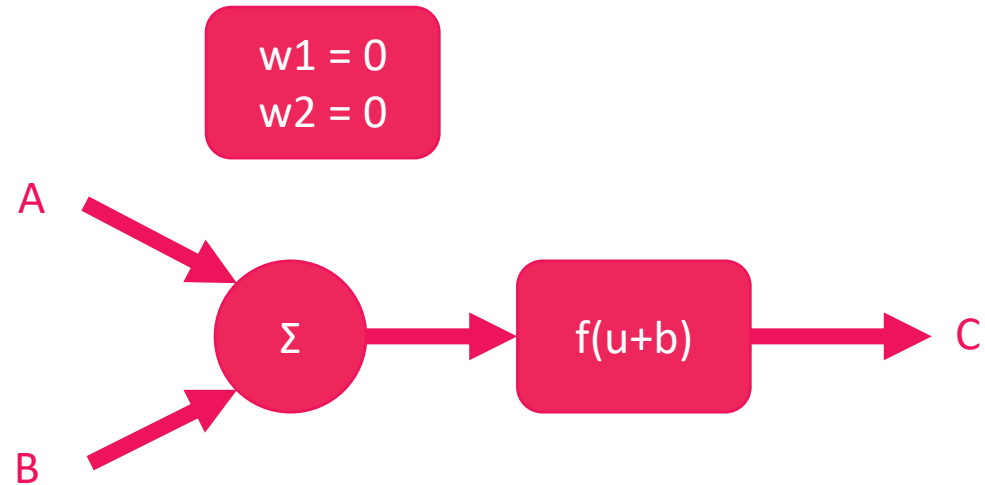
Ajuste de pesos – Treino da Rede Neural!

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1



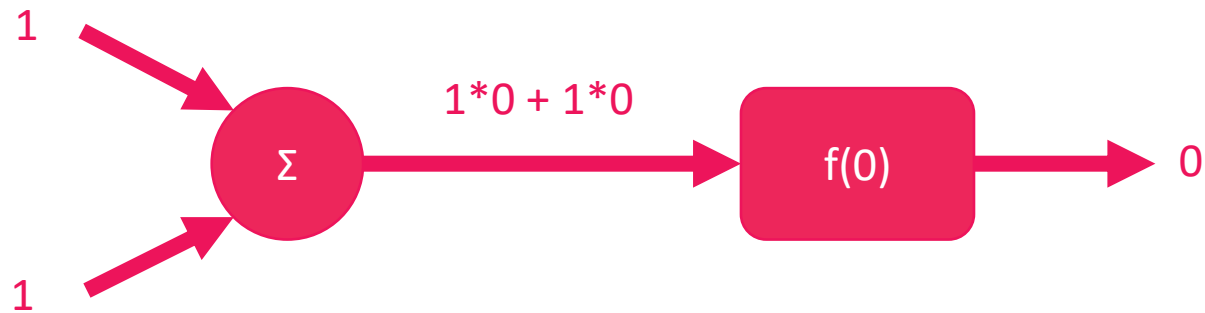
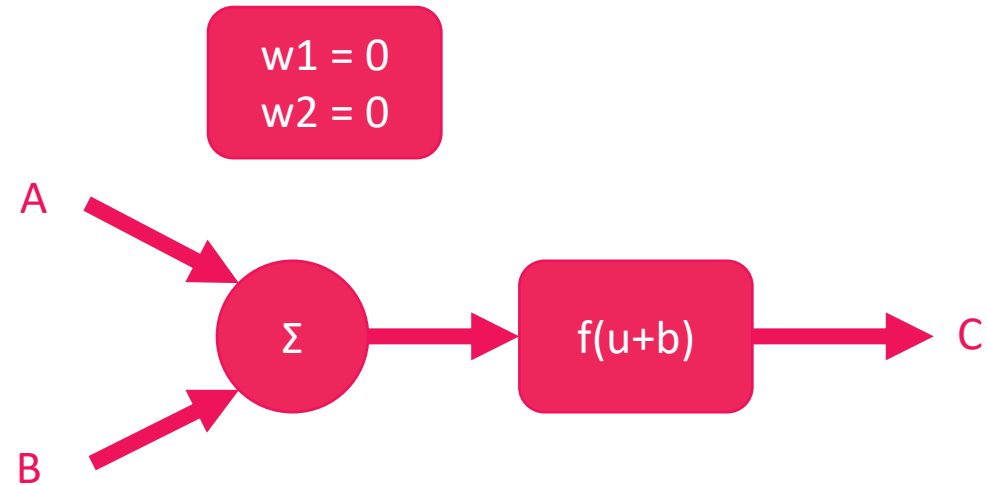
Ajuste de pesos – Treino da Rede Neural!

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

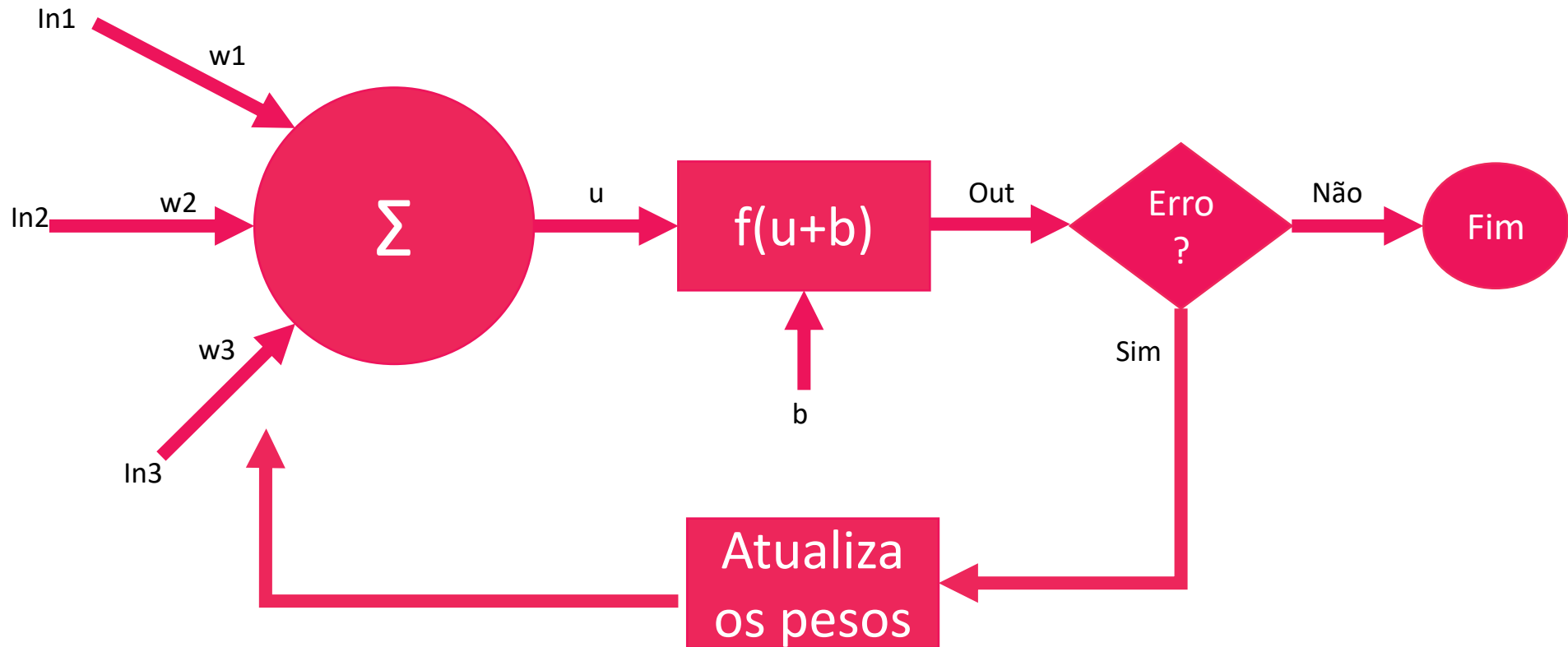


Ajuste de pesos – Treino da Rede Neural!

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

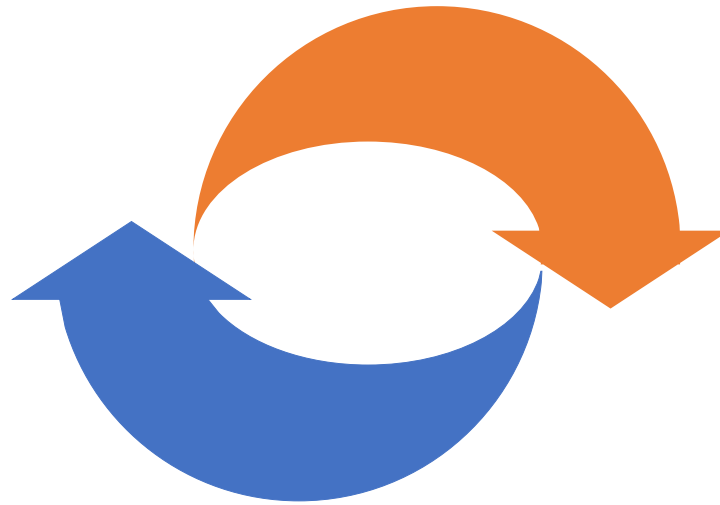


Ajuste de pesos – Treino da Rede Neural!



Ajuste de pesos – Calculado o Erro e atualizando os pesos

$$\text{Erro} = \text{RespostaCorreta} - \text{RespostaCalculada}$$

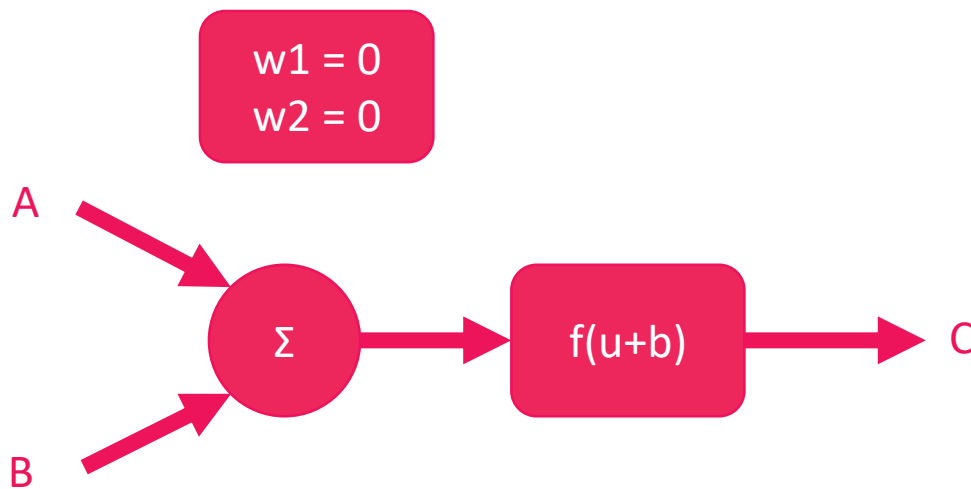


Precisamos atualizar os pesos até os erros serem pequenos ou nulos;

$$\text{pesos}(n + 1) = \text{peso}(n) + (\text{taxaAprendizagem} * \text{entrada} * \text{erro})$$

Ajuste de pesos – Treino da Rede Neural!

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1



Fixando a taxa de aprendizagem em 0.1

❑ Erro = respostaCorreta – respostaCalculada = 1 – 0 = 1

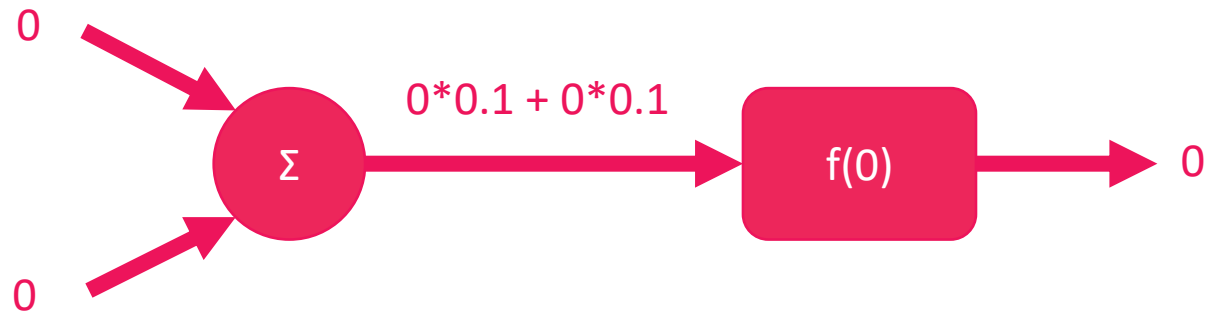
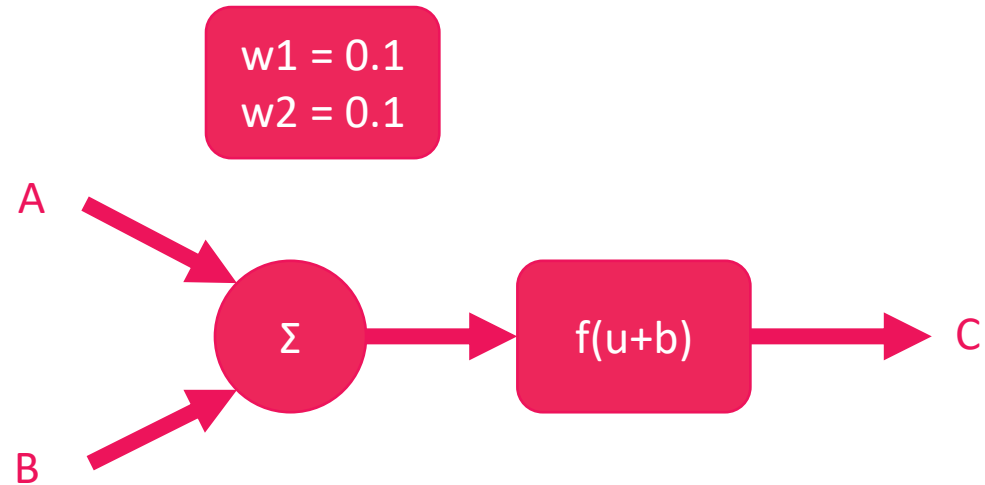
$$pesos(n + 1) = peso(n) + (taxaAprendizagem * entrada * erro)$$

$$A: 0 + (0.1 * 1 * 1) = 0.1$$

$$B: 0 + (0.1 * 1 * 1) = 0.1$$

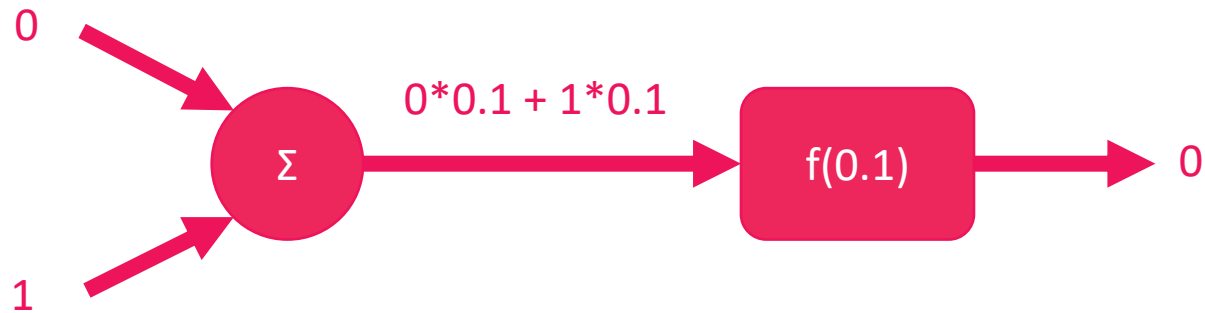
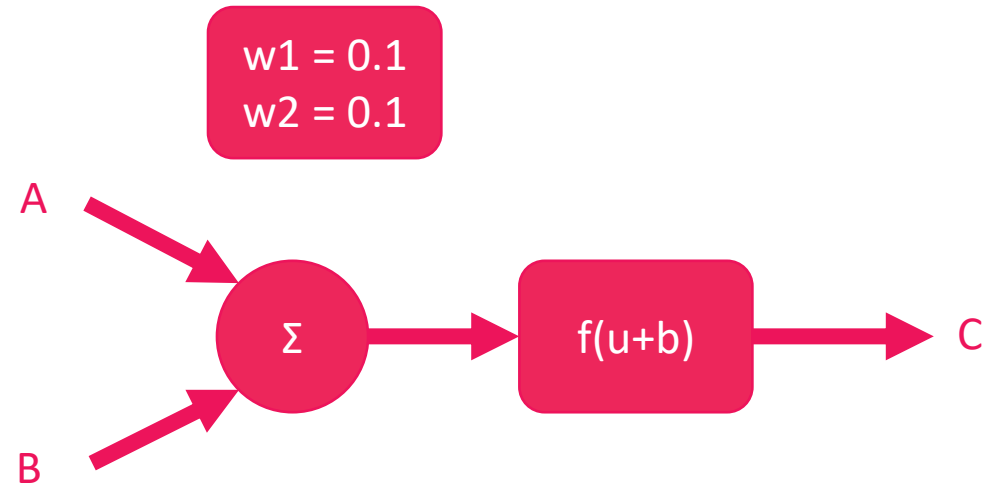
Ajuste de pesos – Treino da Rede Neural!

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1



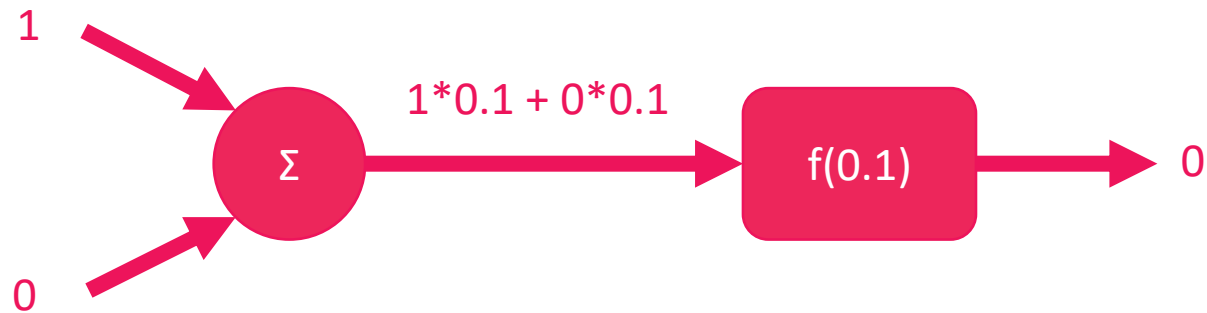
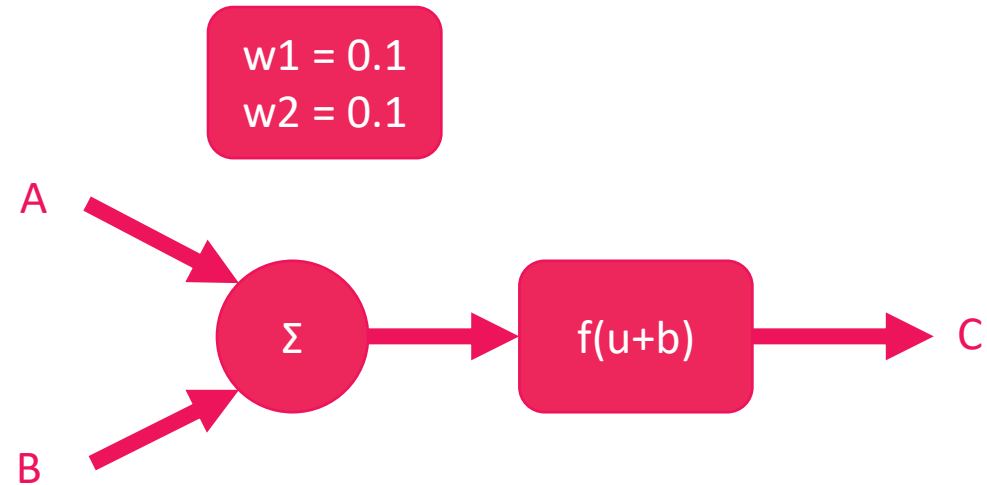
Ajuste de pesos – Treino da Rede Neural!

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1



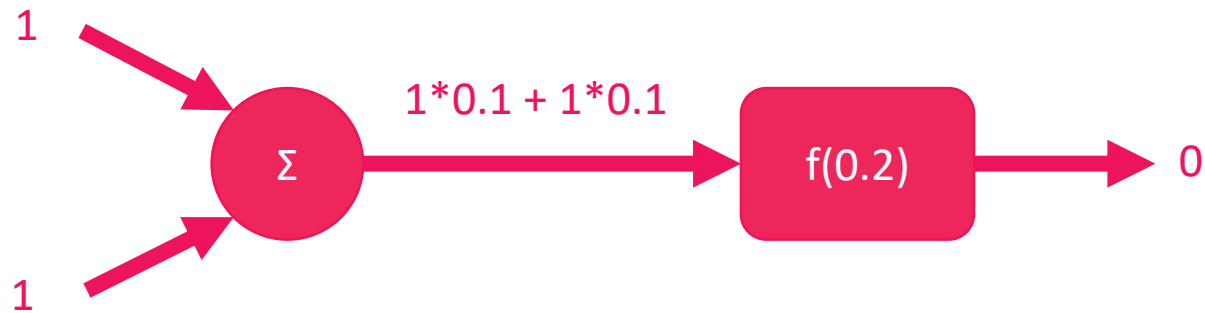
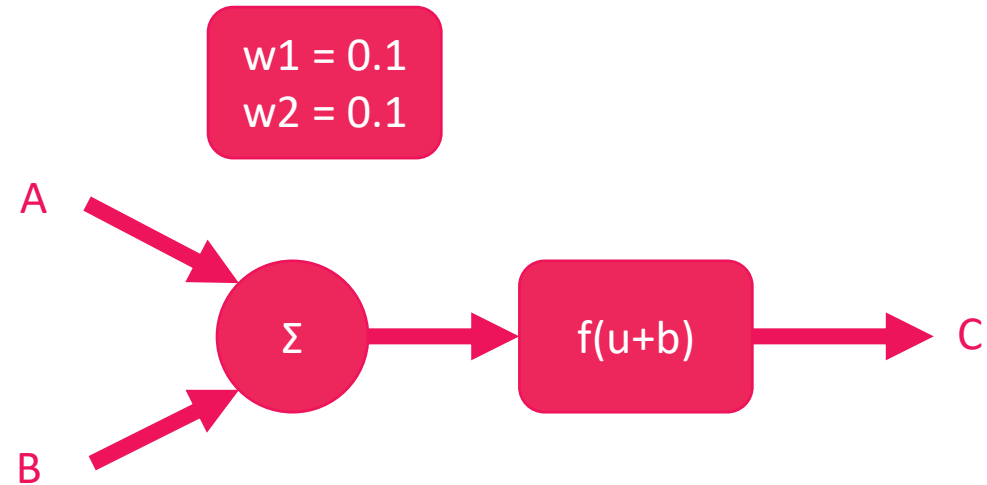
Ajuste de pesos – Treino da Rede Neural!

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1



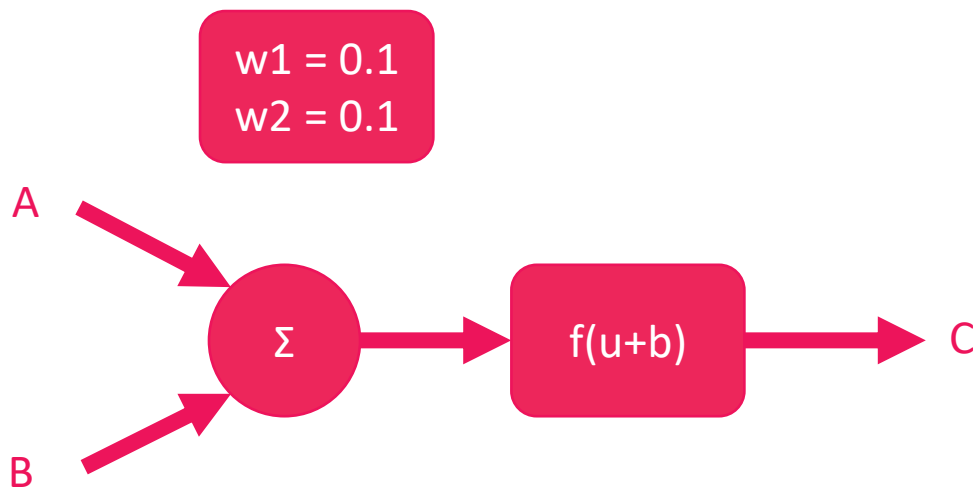
Ajuste de pesos – Treino da Rede Neural!

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1



Ajuste de pesos – Treino da Rede Neural!

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1



Fixando a taxa de aprendizagem em 0.1

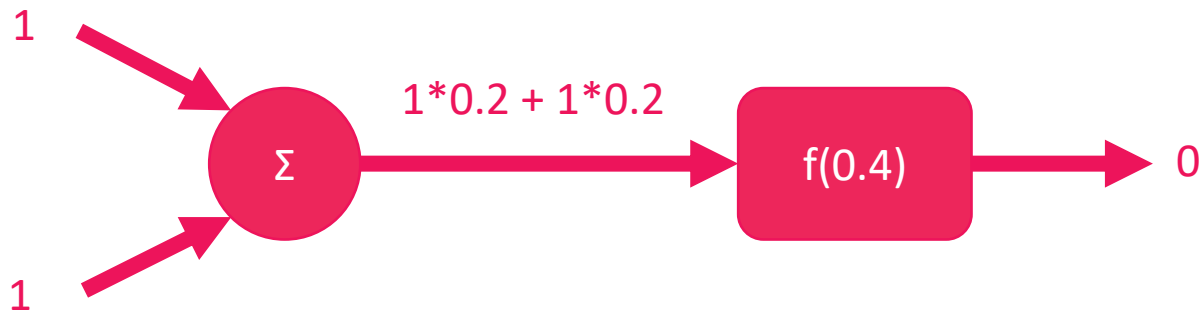
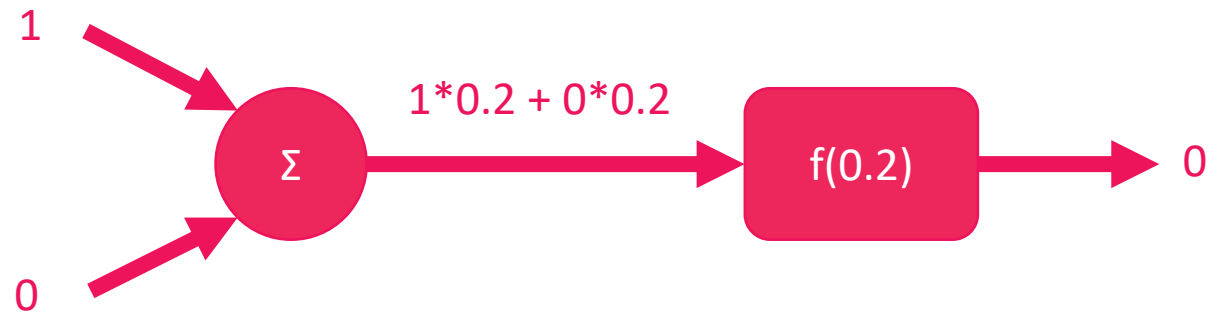
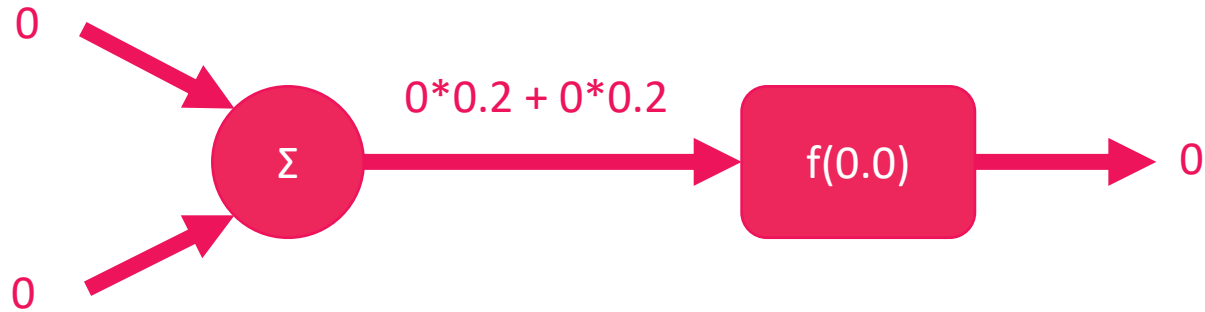
❑ Erro = respostaCorreta – respostaCalculada = 1 – 0 = 1

$$pesos(n + 1) = peso(n) + (taxaAprendizagem * entrada * erro)$$

$$A: 0.1 + (0.1 * 1 * 1) = 0.2$$

$$B: 0.1 + (0.1 * 1 * 1) = 0.2$$

Ajuste de pesos – Treino da Rede Neural!



Exercício 03

- ❑ Encontre os pesos ideais para a Rede Neural Artificial apresentada, mantendo a taxa de aprendizagem de 0.1 e partindo da ultima iteração apresentada.

O conjunto de dados Iris é um dataset clássico de aprendizado de máquina que contém informações sobre três espécies diferentes de flores Iris.

Cada flor é caracterizada por quatro atributos: comprimento da sépala, largura da sépala, comprimento da pétala e largura da pétala.

O objetivo deste laboratório é criar um modelo de **Rede Neural Perceptron** para classificar os dados de acordo com essas três espécies.



Copyright © 2023 Prof. Airtton Y. C. Toyofuku/Yan Coelho

Todos direitos reservados. Reprodução ou divulgação total ou parcial deste documento é expressamente proibido sem o consentimento formal, por escrito, do Professor (autor).