ALGORITMOS GENÉTICOS E REDES NEURAIS

INTRODUÇÃO ÀS REDES NEURAIS ARTIFICIAIS

Prof. Flávio Belizário da Silva Mota
Universidade do Vale do Sapucaí – UNIVAS
Sistemas de Informação

OBJETIVOS DA DISCIPLINA

Compreender os fundamentos de redes neurais artificiais e algoritmos genéticos, desenvolvendo a capacidade de aplicar essas técnicas em problemas de classificação, otimização e planejamento.

PLANEJAMENTO DA DISCIPLINA

Datas	Conteúdo
14/out	Semana de SI
16/out	Semana de SI
21/out	Introdução às RNAs
23/out	MLP com TensorFlow
04/nov	Aprendizagem Supervisionada
06/nov	Aprendizagem Não supervisionada (SOM)
11/nov	Noções de Deep Learning: CNN
13/nov	Prova integrada
18/nov	CNN - MNIST/CIFAR
20/nov	Feriado
25/nov	Algoritmos Genéticos
27/nov	Path-finding com AGs
02/dez	Jogos e planejamento
04/dez	Prova
09/dez	2ª chamada
11/dez	2ª chamada

DISTRIBUIÇÃO DE NOTAS DA DISCIPLINA

Atividade	**Data prevista**	**Valor**
Prova integrada	13/nov	20 pontos
Prova individual	04/dez	40 pontos
Exercícios durante as aulas	Contínuo	40 pontos
Total	—	**100 pts**

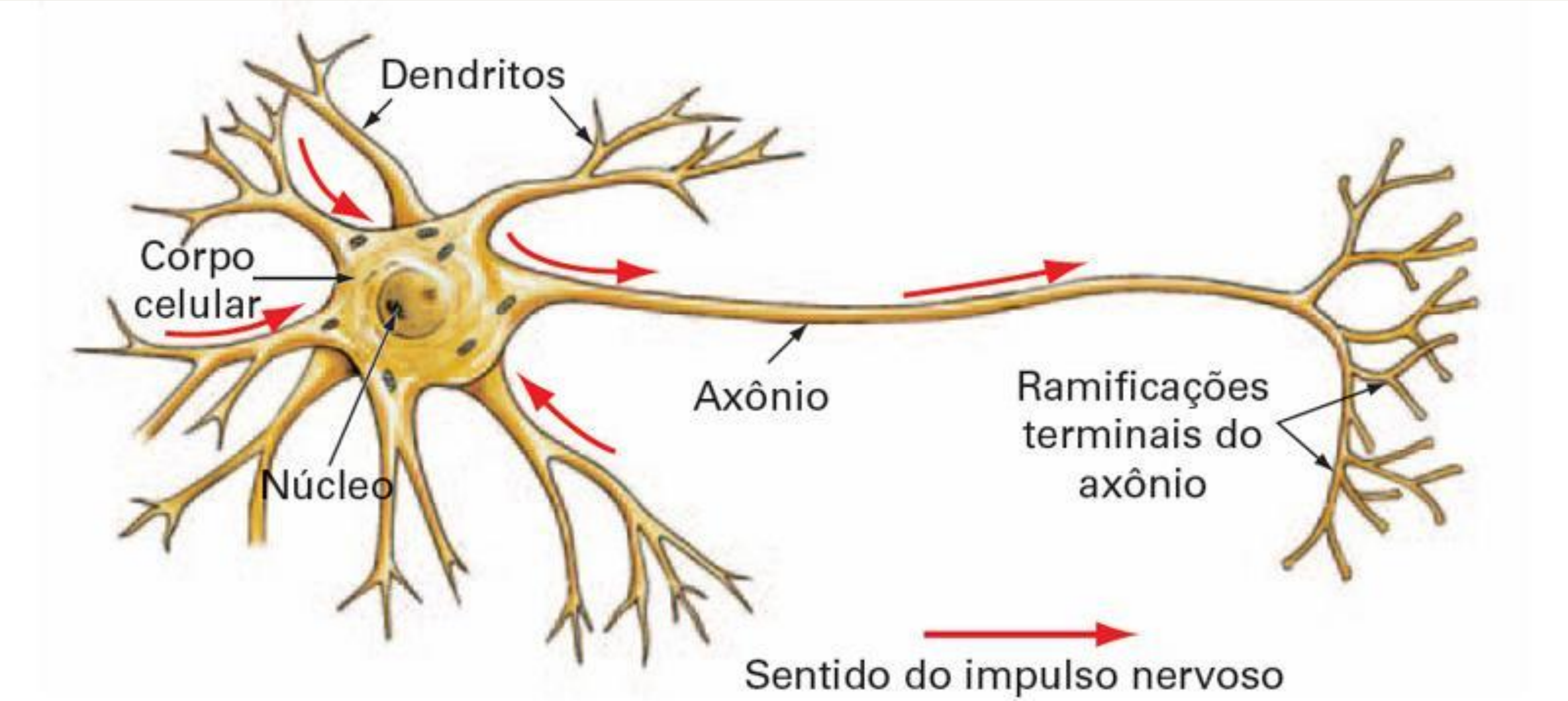
REDES NEURAIS ARTIFICIAIS

- A ciência desde sempre se baseia na observação de fenômenos naturais para construir novas tecnologias
- Pássaros nos inspiraram a voar
- A planta bardana inspirou a criação do velcro



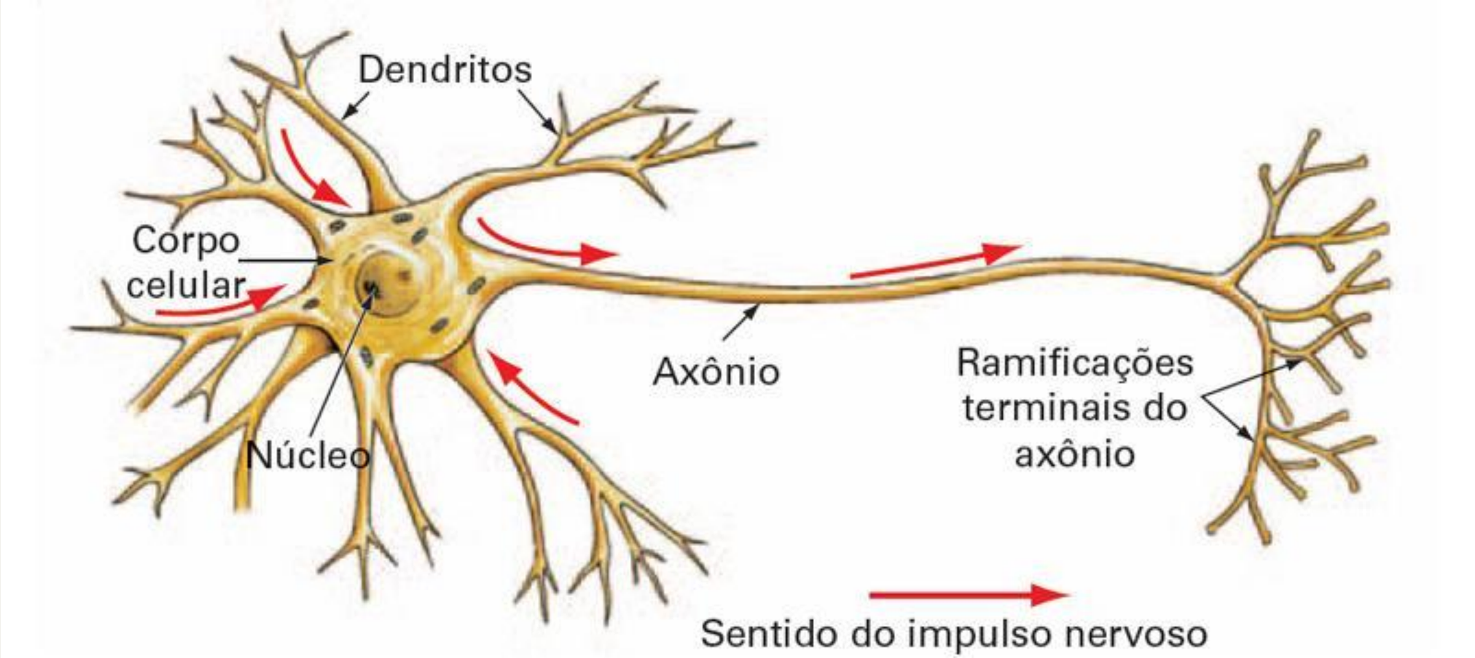

REDES NEURAIS ARTIFICIAIS

- Nesse sentido, as Redes Neurais Artificiais (RNAs) surgiram como uma tentativa de criar **máquinas inteligentes** a partir do funcionamento dos **neurônios biológicos**



NEURÔNIOS BIOLÓGICOS

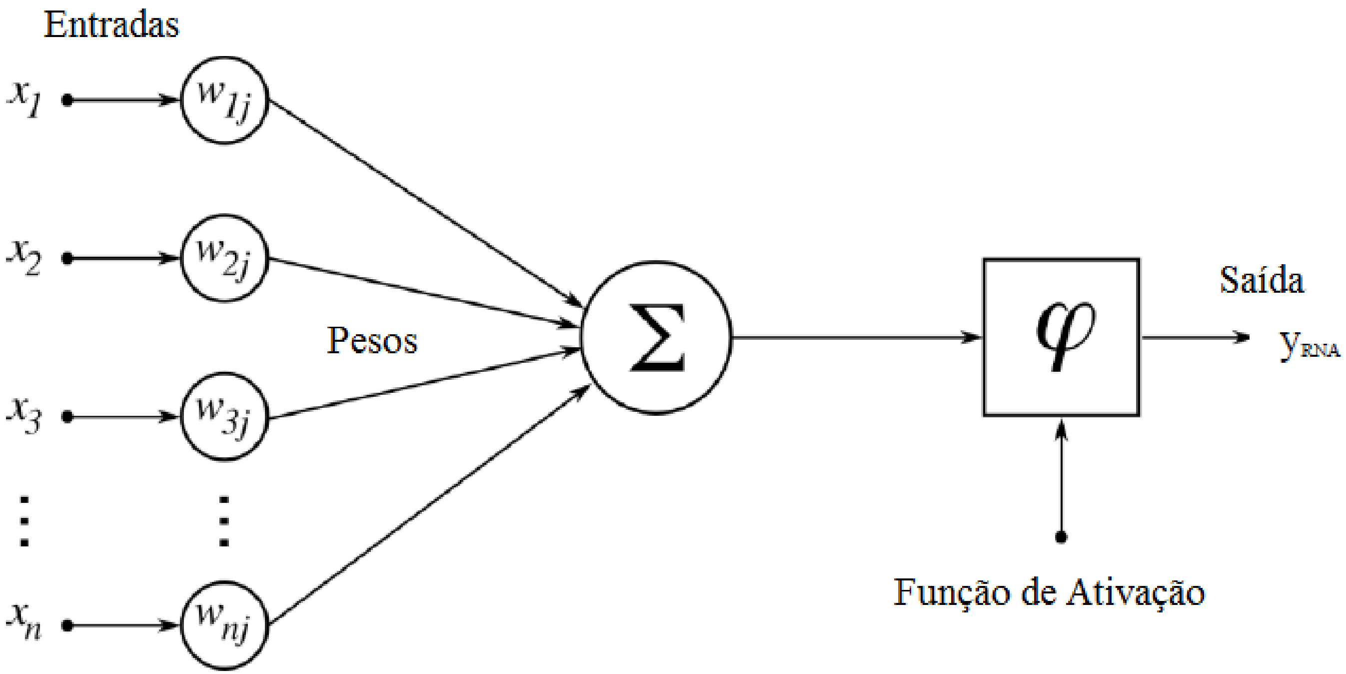
- Um cérebro humano possui mais de 10 bilhões de neurônios, cada qual conectado a milhares de outros neurônios. Essas conexões são as sinapses.
- Composto de um corpo celular que contém:
  - Núcleo
  - Dendritos
  - Axônio



- O neurônio recebe entradas de outros neurônios, a partir dos dendritos e quando esse sinal excede um limiar o neurônio é “ativado”, enviando um pulso elétrico para outros neurônios a partir do axônio.

NEURÔNIOS ARTIFICIAIS

- **Inspirados pelo neurônio biológico, McCulloch e Pitts (1943) propuseram uma representação computacional simplificada, utilizando lógica proposicional.**
- **Em 1957, Frank Rosenblatt propõe o Perceptron, a primeira arquitetura de RNA. Nessa representação, cada conexão de entrada está associada a um peso.**



PESOS

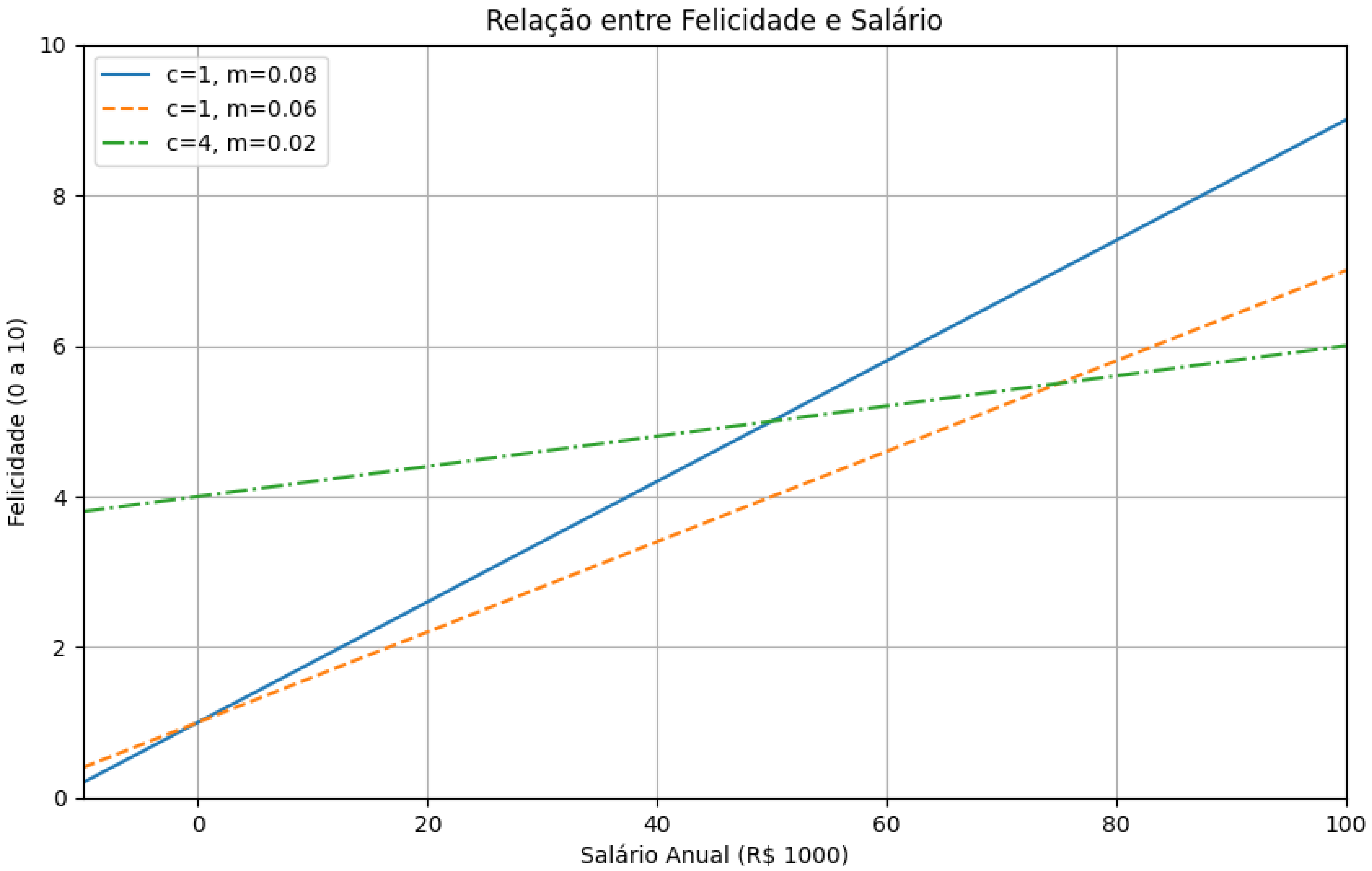
- **Os pesos do neurônio podem ser entendidos como importância de cada entrada, ou o quanto ela colabora para o resultado final. Veja a equação da reta:**

$$y = m \cdot x + c$$

- **Na qual *y* é a variável de saída, *x* é a variável de entrada e *m* e *c* são os parâmetros do coeficiente angular e linear da reta.**
- **Agora, imagine que podemos mapear a felicidade de uma pessoa baseado no salário anual dela usando a equação da reta:**

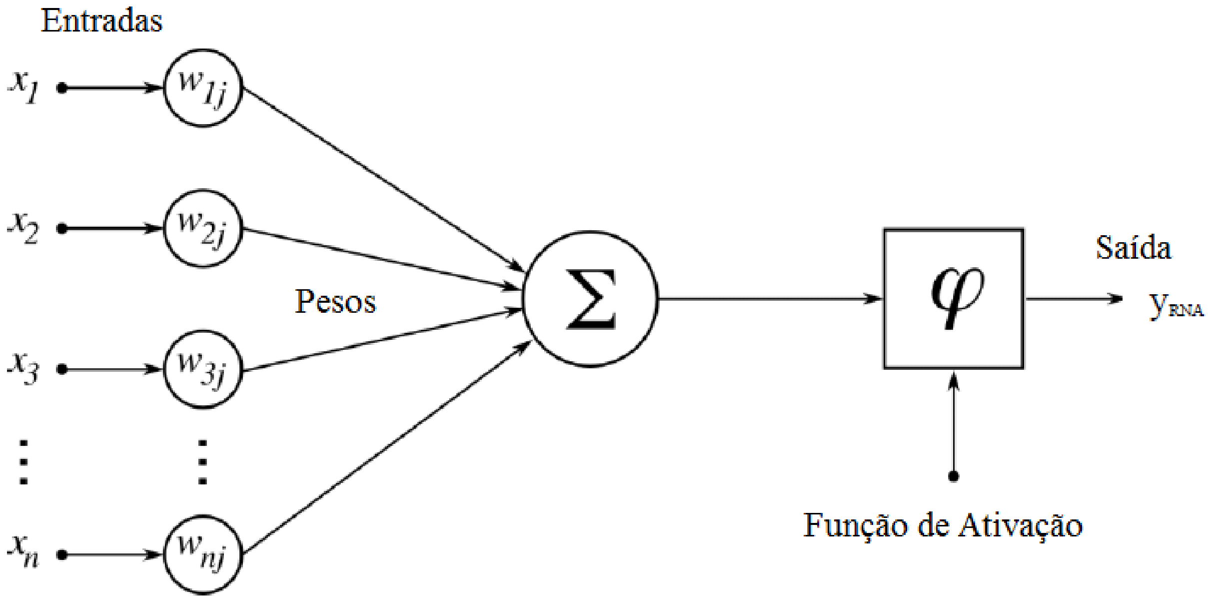
$$\text{felicidade} = m \cdot \text{salario} + c$$

- **Podemos criar um modelo que diz o quanto uma pessoa é feliz baseado no quanto ela ganha simplesmente alterando os valores de *m* e *c***


Relação entre Felicidade e Salário
c=1, m=0.08
c=1, m=0.06
c=4, m=0.02
Felicidade (0 a 10)
Salário Anual (R$ 1000)
0
2
4
6
8
10
0
20
40
60
80
100

PESOS

- **Na analogia anterior, os pesos são o parâmetro *m*, pois eles são capazes de indicar o quanto uma variável de entrada *x* influencia o resultado da saída do neurônio.**
- **Como no Perceptron existem várias entradas e vários pesos, o valor da saída *y* é dada por uma soma da multiplicação dos pesos pelas entradas, uma soma ponderada.**



$$y = \sum_{i=0}^{n} (\text{entrada}_i \times \text{peso}_i)$$

$$y = \sum_{i=0}^{n} (x_i \times w_i)$$

VERIFICANDO O PAGAMENTO DE EMPRÉSTIMO

- A tabela ao lado mostra o quanto uma pessoa tem de renda e quanto ela ainda deve para o empréstimo
- Para calcular uma possibilidade de uma pessoa pagar ou não o empréstimo (solvência), poderíamos aplicar a fórmula anterior linha a linha, considerando as colunas como as nossas entradas

ID	Renda	Dívida
1	R$ 150	-R$ 100
2	R$ 250	-R$ 300
3	R$ 400	-R$ 300
4	R$ 200	-R$ 350

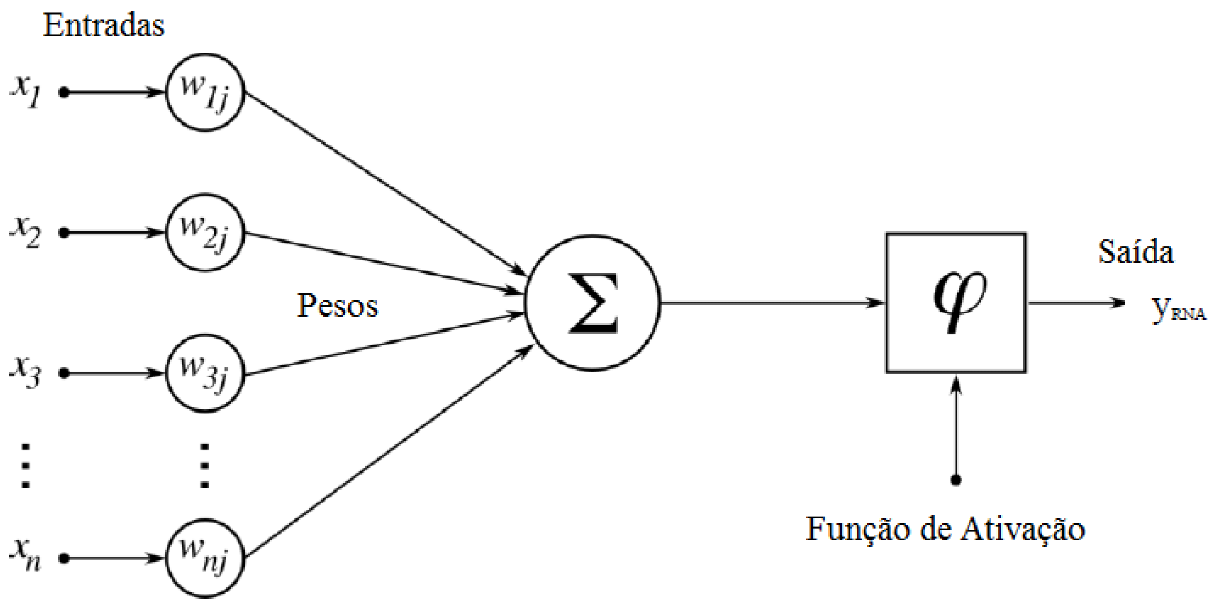
$$\mathrm{solvencia} = \sum_{\mathrm{i}=0}^{\mathrm{n}} (\mathrm{entrada_i} \times \ \mathrm{peso_i})$$

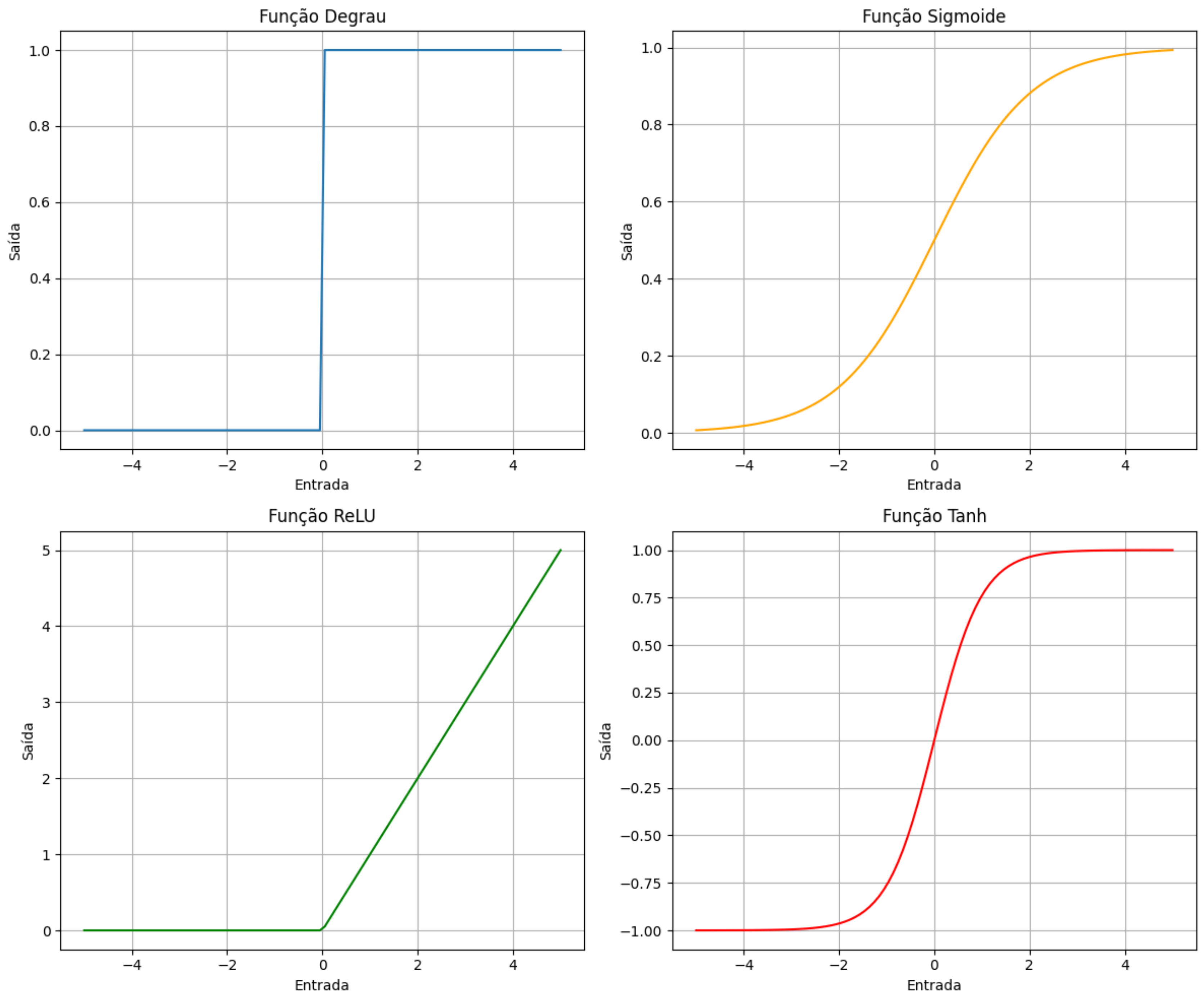
$$solvencia = (renda \times peso\ da\ renda) + (divida \times peso\ da\ divida)$$

- Vamos considerar que a renda pesa mais que a dívida nesse cenário, então assumimos que o peso da renda é 3 o da dívida é 1.

FUNÇÃO DE ATIVAÇÃO

- **Entretanto, os pesos não são a única parte de um Perceptron. O resultado do somatório passa por uma Função de Ativação (assim como no neurônio biológico).**
- **Existem diversas funções de ativação que determinam qual será de fato a saída do Perceptron.**





- As funções de ativação servem para determinar o nível de ativação do neurônio, mapeando o valor resultante do somatório em um valor de saída.
- A **função degrau** mapeia a saída entre 0 e 1, porém com um **limiar fixo**.
- A **função sigmoide** mapeia a saída entre 0 e 1, variando suavemente.
- A função **tangente hiperbólica** mapeia a saída entre -1 e 1.
- A **função ReLU (Rectified Linear Unit)**, mapeia a saída entre 0 e um valor máximo igual ao valor da entrada.

FUNÇÃO DE ATIVAÇÃO

- **Considere um neurônio simples de duas entradas com os seguintes pesos associados:**
  - $w_1 = 0{,}8$
  - $w_2 = 0{,}4$
- **E as seguintes entradas:**
  - $x_1 = 0{,}7$
  - $x_2 = 0{,}9$

$$y = \text{Degrau}\left(\sum_{i=0}^{n}(x_i \times w_i)\right)$$

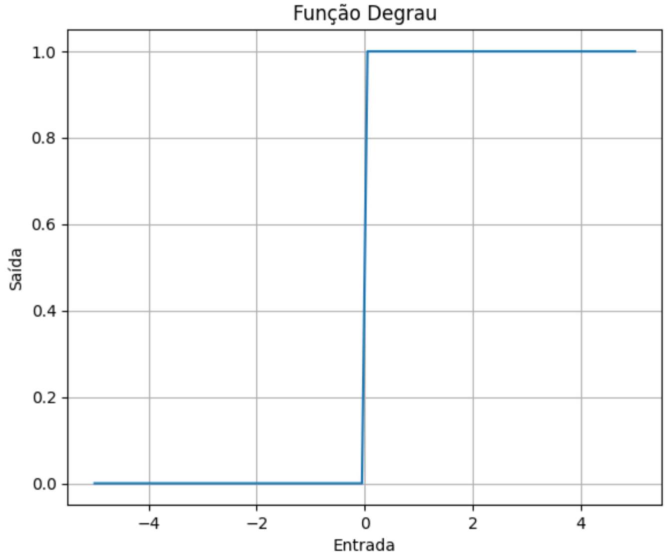
$$y = \text{Degrau}((x_1 \times w_1) + (x_2 \times w_2))$$

$$y = \text{Degrau}((0{,}7 \times 0{,}8) + (0{,}9 \times 0{,}4))$$

$$y = \text{Degrau}(0{,}92)$$

$$y = 1$$

Se usarmos a função degrau com o limiar em 0, significa que a saída do neurônio será igual a 1.



COMO O PERCEPTRON APRENDE?

- **Os pesos de um Perceptron são inicializados de forma aleatória. Então para que ele aprenda a dar uma resposta, como classificar algo como 0 ou 1, ele precisa ajustar os pesos, para ajustar a saída.**
- **Em 1960, Rosenblatt propõe uma fórmula de modificação dos pesos:**

$$\mathrm{w_i} \leftarrow \mathrm{w_i} + (a \times \mathrm{x_i} \times e)$$

Onde:

$\mathrm{w_i}$ **é o peso da conexão *i***

a **é a taxa de aprendizado, sendo 0 < a < 1**

$\mathrm{x_i}$ **é a própria entrada na conexão *i***

e **é o erro produzido na saída da rede**

COMO O PERCEPTRON APRENDE?

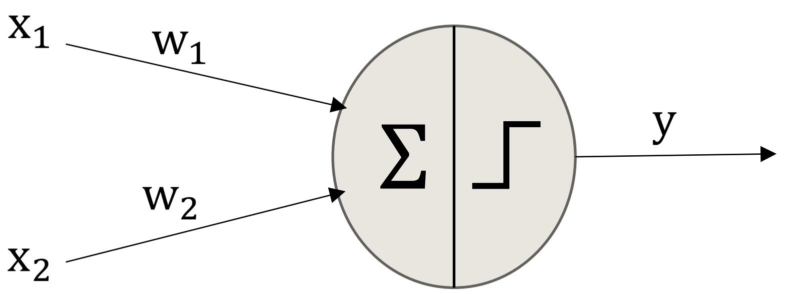
- **Essa modificação será realizada durante o treinamento do Perceptron:**
  - **Primeiro, pessos aleatórios são gerados em cada conexão *i***
  - **Depois, um item de dados do conjunto de treinamento é apresentado ao Perceptron, que vai gerar uma saída usando a função de ativação aplicada à soma ponderada**
  - **Observa-se essa saída: se estiver errada (diferente do que é esperado), os pesos são ajustados**
  - **Depois de ajustados, o próximo item dos dados é apresentado da mesma forma.**
  - **Isso se repete até que todos os dados sejam vistos pelo Perceptron e reinicia até que todos os pesos estejam corretos e os erros zerados. Cada iteração dessa é chamada de época.**

$\text{taxa de aprendizado} = a = 0{,}2$

EXEMPLO – PORTA LÓGICA OU (OR)

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

- A tabela ao lado mostra o funcionamento de uma porta lógica OU com duas entradas.
- Vamos utilizar um Perceptron com duas entradas, dois pesos iniciados de forma aleatória, uma função de ativação Degrau e uma taxa de aprendizado de 0,2.



EXEMPLO – PORTA LÓGICA OU (OR)

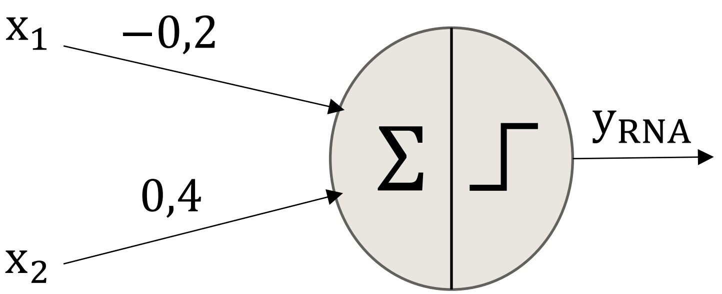
$\text{taxa de aprendizado} = a = 0{,}2$

$w_1 = -0{,}2\,,\, w_2 = 0{,}4$

$x_1 = 0\,,\, x_2 = 0$

$$y_{RNA} = \text{Degrau}\left(\sum_{i=0}^{n}(x_i \times w_i)\right)$$

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1



$$y_{RNA} = \text{Degrau}((x_1 \times w_1) + (x_2 \times w_2))$$

$$y_{RNA} = \text{Degrau}((0 \times -0{,}2) + (0 \times 0{,}4))$$

$$y_{RNA} = \text{Degrau}(0)$$

$$y_{RNA} = 0$$

$$y_{RNA} = y \quad 0 = 0$$

O y esperado é 0 (0 ou 0 é 0), então a saída do Perceptron acertou. Portanto o erro é 0 e não precisamos ajustar os pesos.

EXEMPLO – PORTA LÓGICA OU (OR)

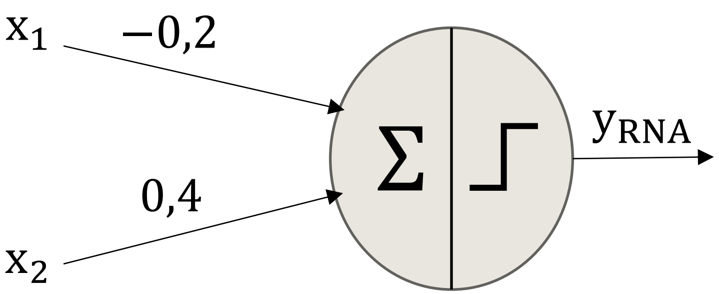
taxa de aprendizado $= a = 0{,}2$

$w_1 = -0{,}2\,, w_2 = 0{,}4$

$x_1 = 0\,, x_2 = 1$

$$y_{RNA} = \text{Degrau}\left(\sum_{i=0}^{n}(x_i \times w_i)\right)$$

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1



$$y_{RNA} = \text{Degrau}((x_1 \times w_1) + (x_2 \times w_2))$$

$$y_{RNA} = \text{Degrau}((0 \times -0{,}2) + (1 \times 0{,}4))$$

$$y_{RNA} = \text{Degrau}(0{,}4)$$

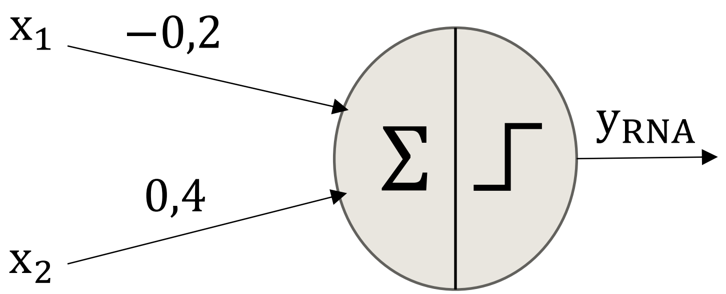
$$y_{RNA} = 1$$

$$y_{RNA} = y \quad 1 = 1$$

O y esperado é 1 (0 ou 1 é 1), então a saída do Perceptron acertou. Portanto o erro é 0 e não precisamos ajustar os pesos.

EXEMPLO – PORTA LÓGICA OU (OR)

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1



$$\text{taxa de aprendizado} = a = 0{,}2$$
$$w_1 = -0{,}2\,, w_2 = 0{,}4$$
$$x_1 = 1\,, x_2 = 0$$

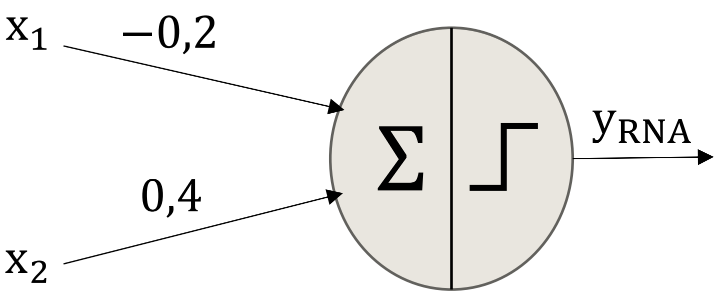
$$y_{RNA} = \text{Degrau}\left(\sum_{i=0}^{n}(x_i \times w_i)\right)$$

$$y_{RNA} = \text{Degrau}((x_1 \times w_1) + (x_2 \times w_2))$$
$$y_{RNA} = \text{Degrau}((1 \times -0{,}2) + (0 \times 0{,}4))$$
$$y_{RNA} = \text{Degrau}(-0{,}2)$$
$$y_{RNA} = 0$$
$$y_{RNA} = y \quad 0 = 1$$

O y esperado é 1 (1 ou 0 é 1), então a saída do Perceptron errou. Portanto precisamos ajustar os pesos.

EXEMPLO – PORTA LÓGICA OU (OR)

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1



$\text{taxa de aprendizado} = a = 0{,}2$

$w_1 = -0{,}2\,, w_2 = 0{,}4$

$x_1 = 1\,, x_2 = 0$

$e = y - y_{RNA}$

$e = 1 - 0$

$e = 1$

$w_i \leftarrow w_i + (a \times x_i \times e)$

Ajustamos o primeiro peso:

$w_1 \leftarrow -0{,}2 + (0{,}2 \times 1 \times 1)$

$w_1 \leftarrow 0$

Ajustamos o segundo peso:

$w_2 \leftarrow 0{,}4 + (0{,}2 \times 0 \times 1)$

$w_2 \leftarrow 0{,}4$

Note que o peso 2 não contribuiu para o erro, então ele não é ajustado. Na próxima execução, esses serão os novos pesos do Perceptron.

EXEMPLO – PORTA LÓGICA OU (OR)

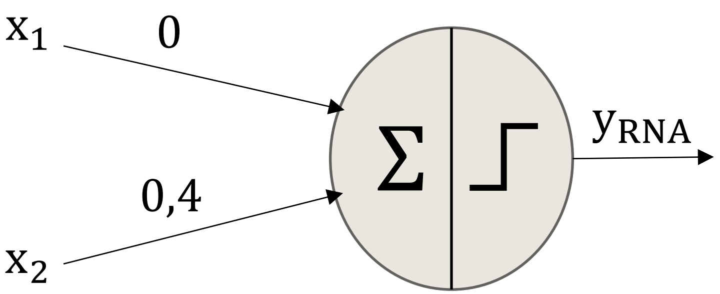
taxa de aprendizado $= a = 0{,}2$

$w_1 = 0 \, , w_2 = 0{,}4$

$x_1 = 1 \, , x_2 = 0$

$$y_{RNA} = \text{Degrau}\left(\sum_{i=0}^{n} (x_i \times w_i)\right)$$

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1



$$y_{RNA} = \text{Degrau}((x_1 \times w_1) + (x_2 \times w_2))$$

$$y_{RNA} = \text{Degrau}((1 \times 0) + (1 \times 0{,}4))$$

$$y_{RNA} = \text{Degrau}(0{,}4)$$

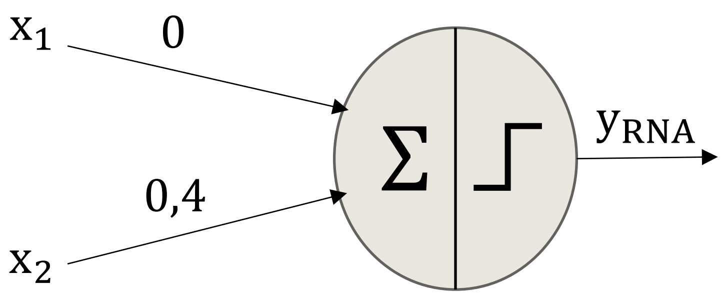
$$y_{RNA} = 1$$

$$y_{RNA} = y \quad 1 = 1$$

O y esperado é 1 (1 ou 1 é 1), então a saída do Perceptron acertou. Portanto não precisamos ajustar os pesos.

EXEMPLO – PORTA LÓGICA OU (OR)

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1



Esse é o fim da primeira **época**. O processo é repetido novamente até que todos os quatro exemplos do dado sejam classificados corretamente.

Quantas épocas são necessárias para que o Perceptron se estabilize? Faça a execução do processo até que não haja ajuste nos pesos.