



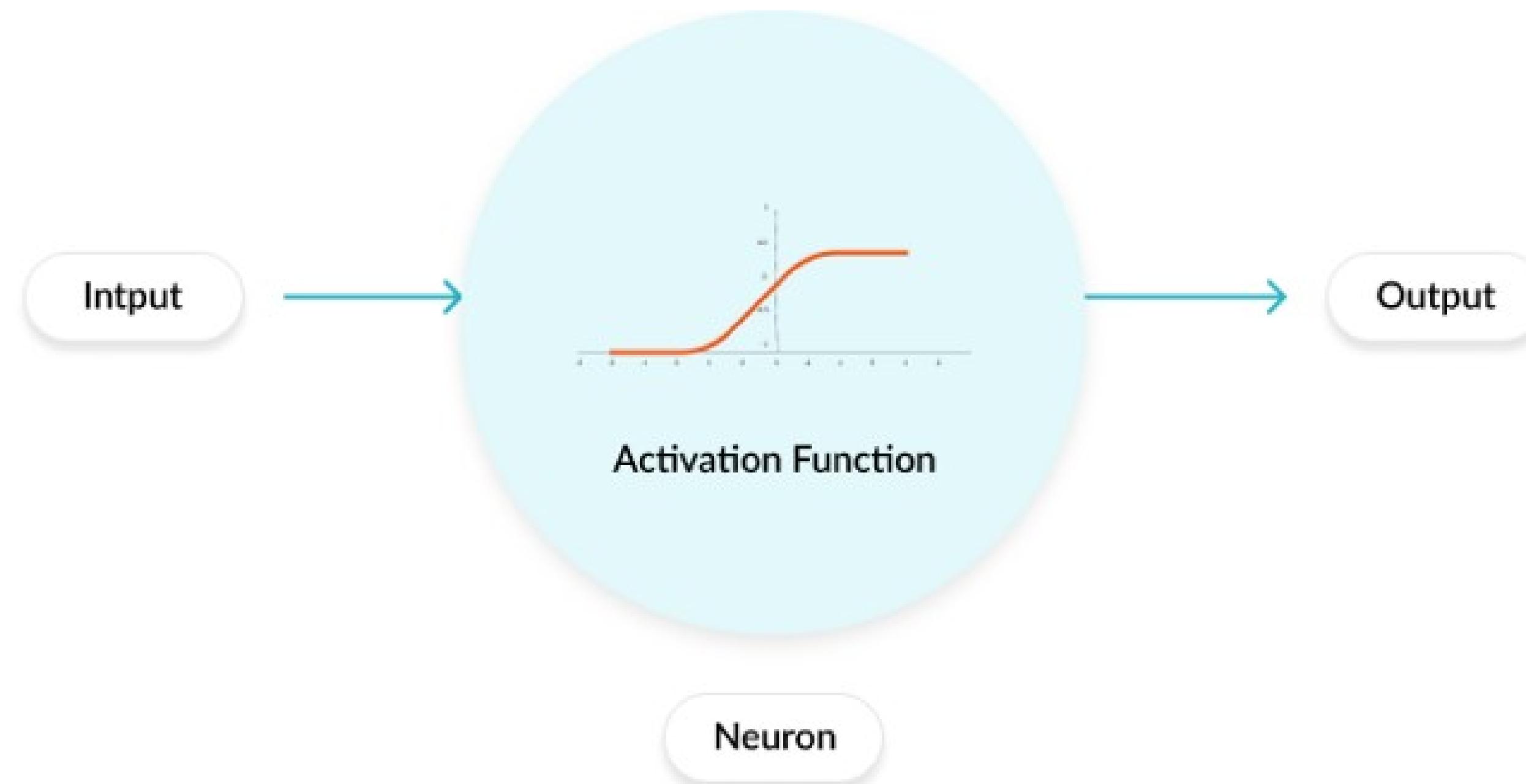
Sistemas Inteligentes Redes Neurais

Semana 11 – Introdução a Redes Neurais e tipos de Aprendizado
Prof. Malga

Redes Neurais e Tipos de Aprendizado

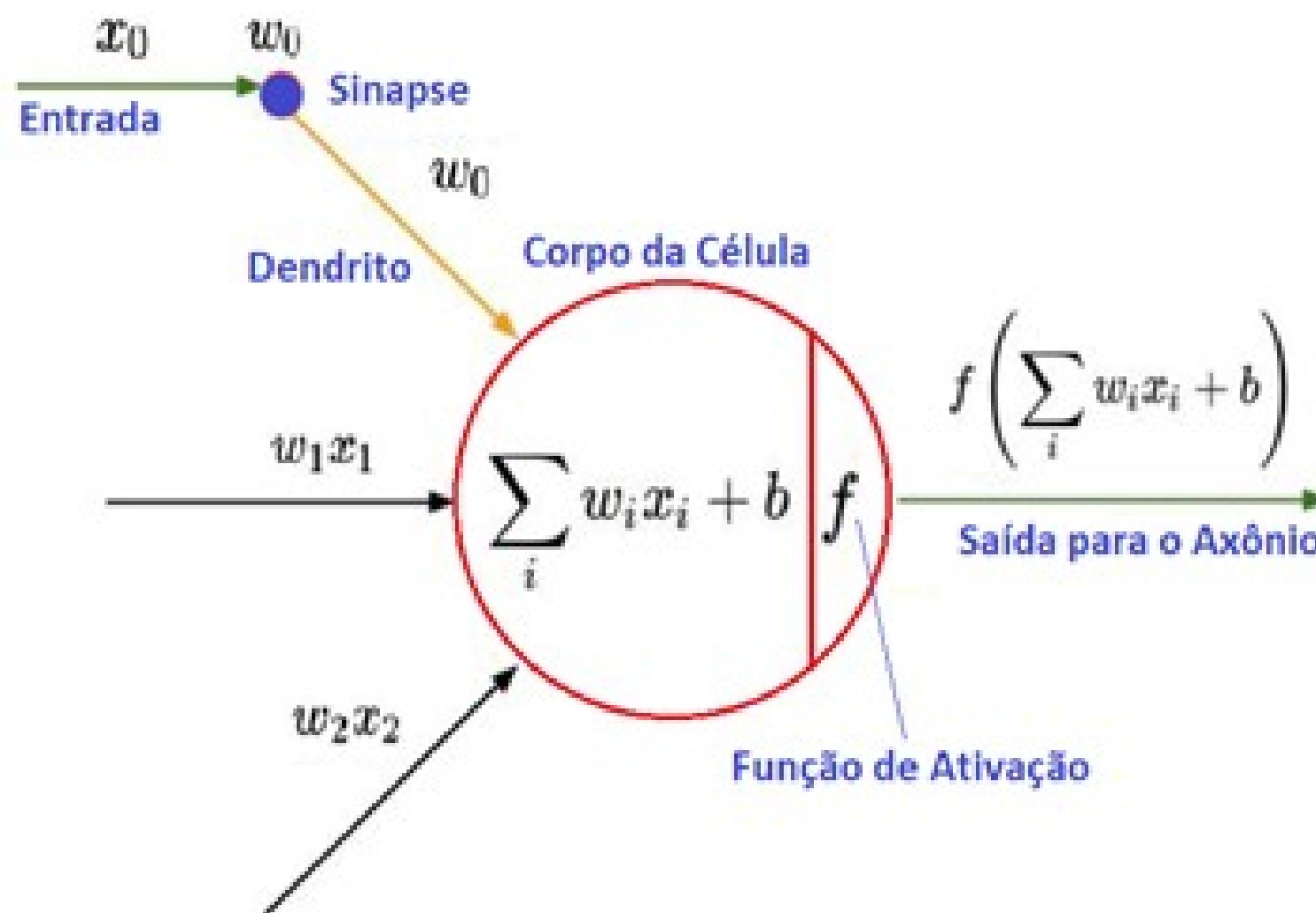


Funções de Ativação



Funções de Ativação

O que é a função de ativação?

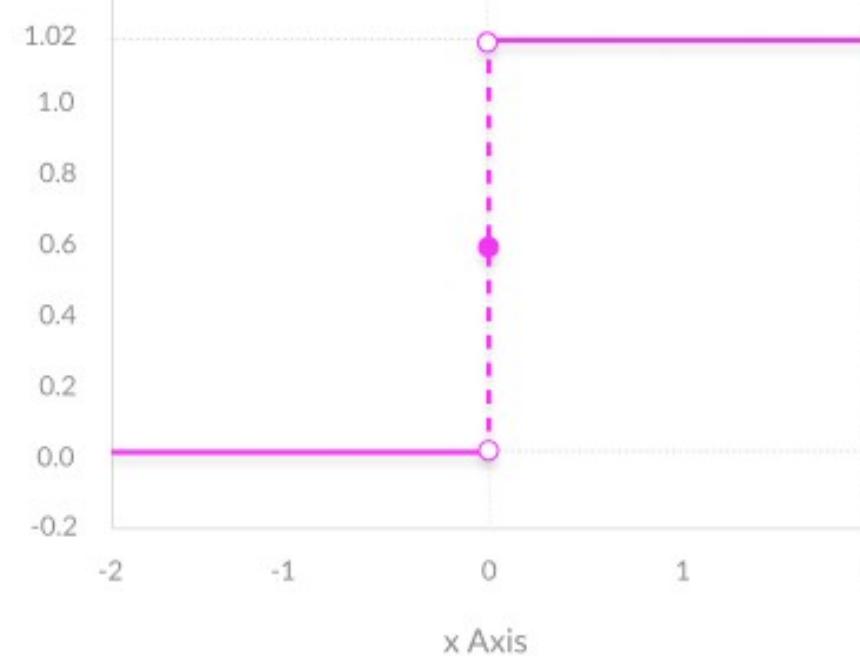


Uma **função de ativação**, nada mais é do que uma **equação matemática que determina o output do seu neurônio**, podendo ser encontrada no final de cada neurônio de uma rede neural. A principal função da função de ativação é *introduzir não-linearidade na saída da unidade*, o que **permite que a rede neural aprenda padrões mais complexos nos dados**.

Funções de Ativação

Funções de Ativação

Função Degrau



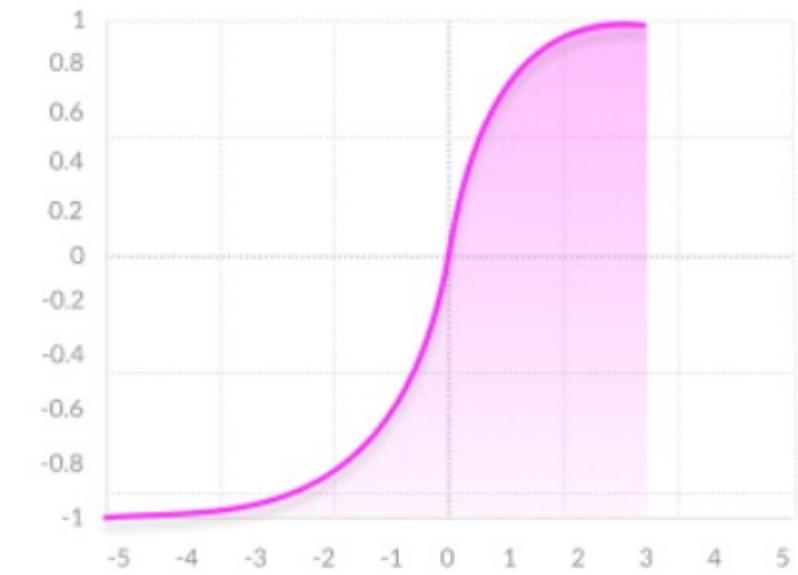
Função Linear



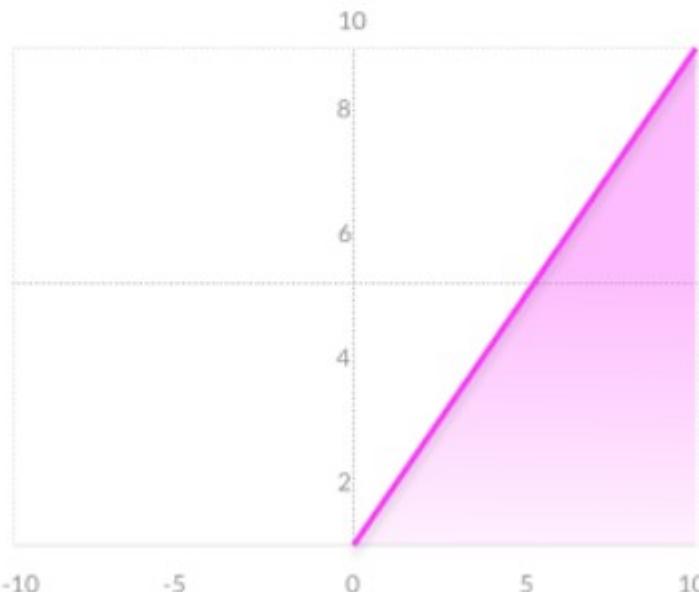
Função Sigmoid



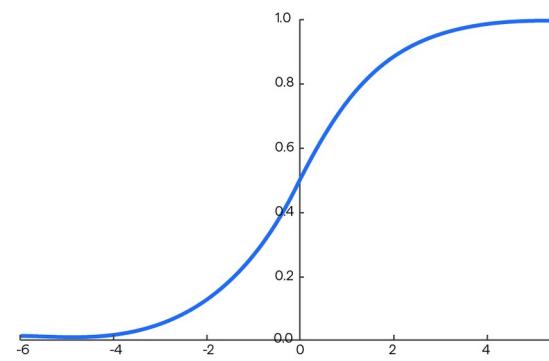
Função Tangente Hiperbólica



Função ReLU

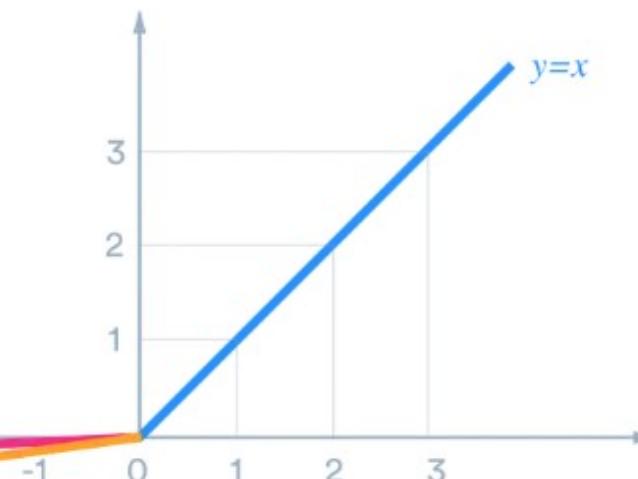


Softmax Function



Leaky ReLU: $y=0.01x$

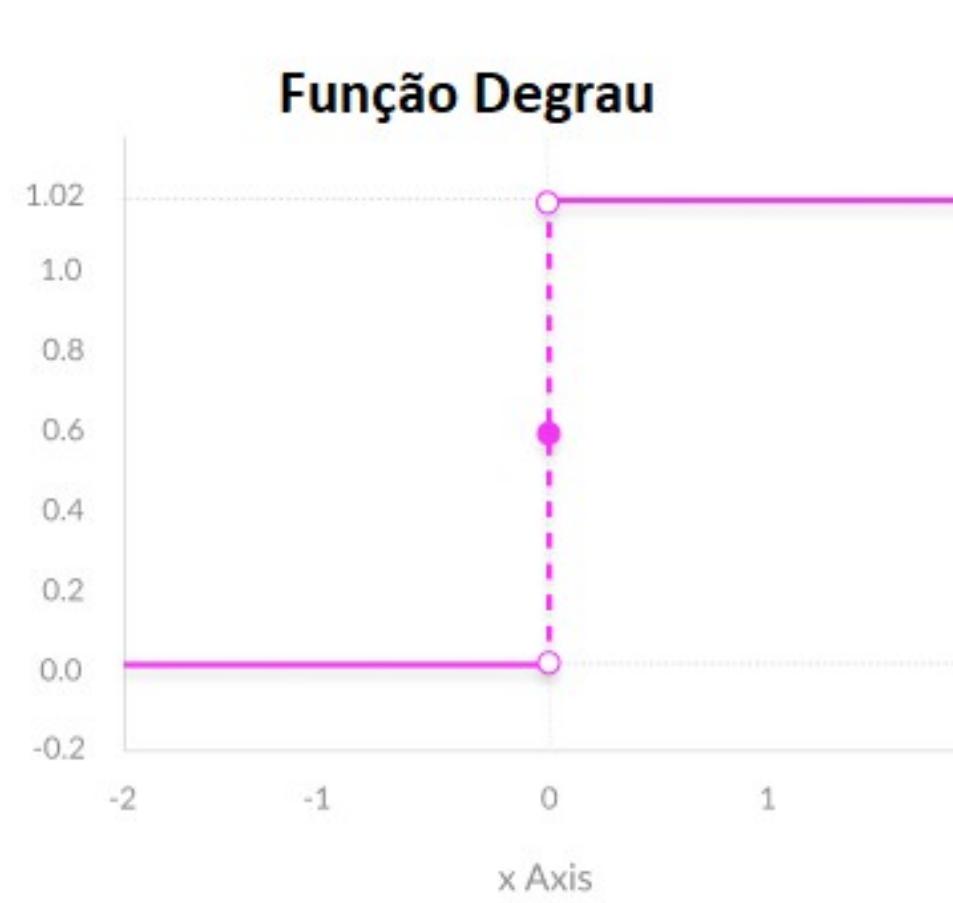
Parametric ReLU: $y=ax$



Funções de Ativação

Função Degrau – (A mais adequada para Perceptron simples)

A **função de degrau** (etapa binária) é extremamente simples. Ela pode ser usada ao criar um classificador binário. Quando simplesmente precisamos dizer sim ou não **para uma única classe**, a função seria a melhor escolha, pois ativaría o neurônio ou deixaria zero.



Funções de Ativação

Função Linear

Essa função recebe a entrada, que é multiplicada por cada peso do neurônio, e emite um output com o valor proporcional ao recebido. É melhor que a binária, pois permite a função emitir sinais com valores além do 0 e 1



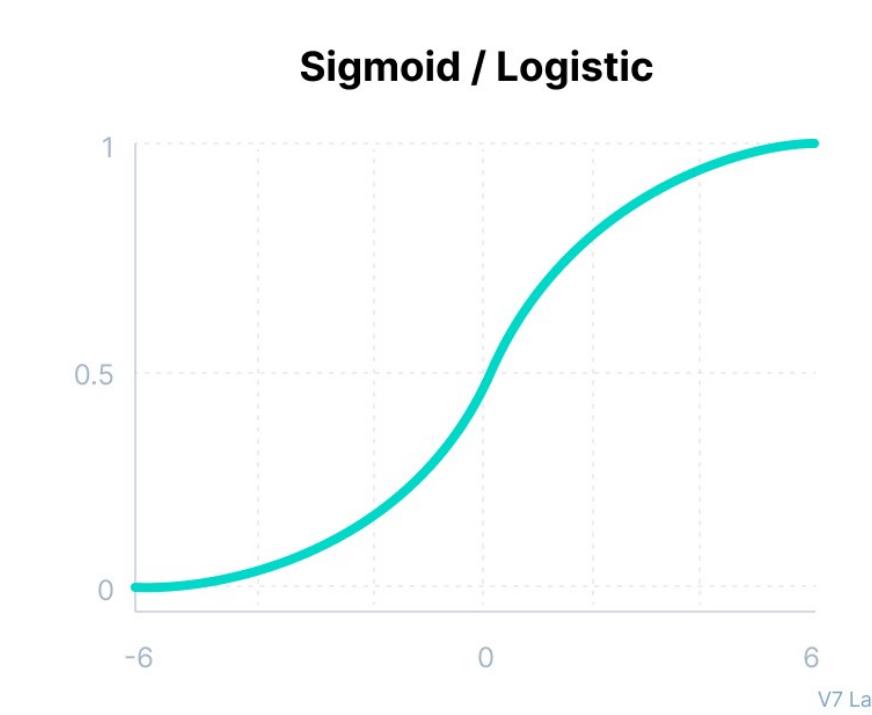
Estudou	Conhecimento prévio	Facilidade de aprendizado	Nome	Resultado
0.8	0.5	0.7	Miguel	1
0.3	0.2	0.8	Bruno	0

Funções de Ativação

Sigmoide

A função Sigmoide é contínua, tendo seus valores variando entre 0 e 1.

A função de ativação Sigmoide é adequada para resolução de problemas que envolvem valores probabilísticos como saída.



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

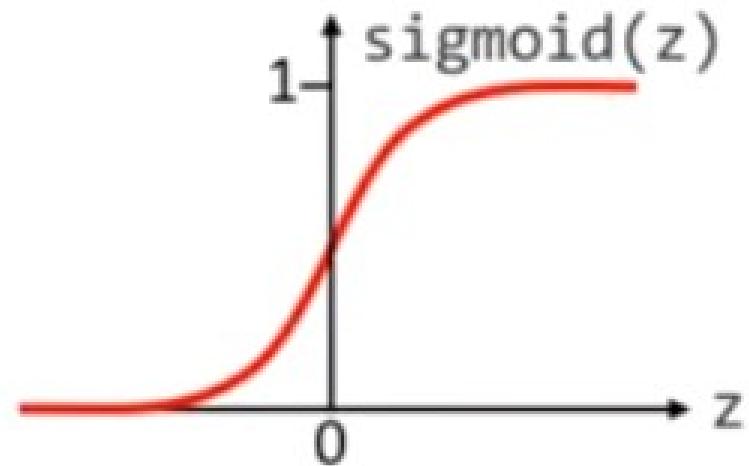
Equação

Graficamente

Funções de Ativação

Softmax

Quando o problema tem mais de uma classe, é aplicado uma Função de Ativação que é uma Generalização da Sigmoidal que é a *Softmax*.



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

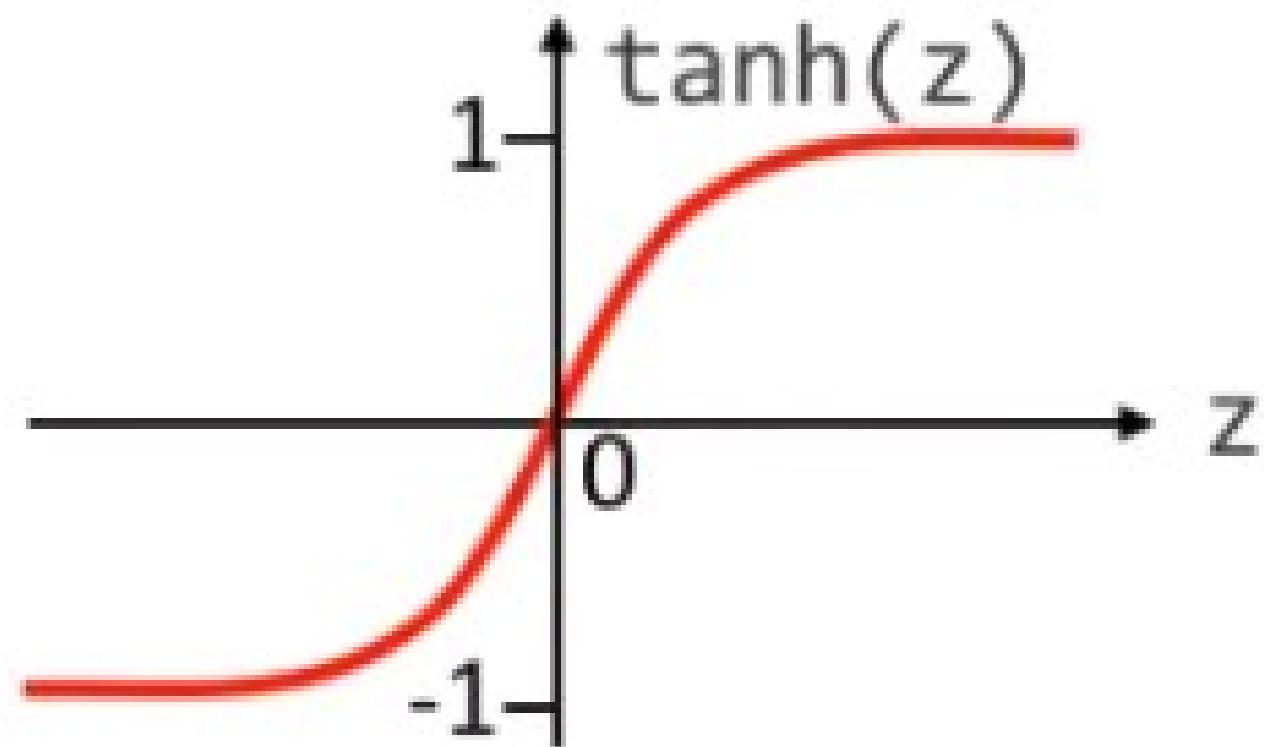
$$\text{Softmax}(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$$

Funções de Ativação

Funções de Ativação – PARA CAMADAS OCULTAS

Tangente Hiperbólico

Semelhante a sigmoide, porém tem uma variação de -1 até 1 .

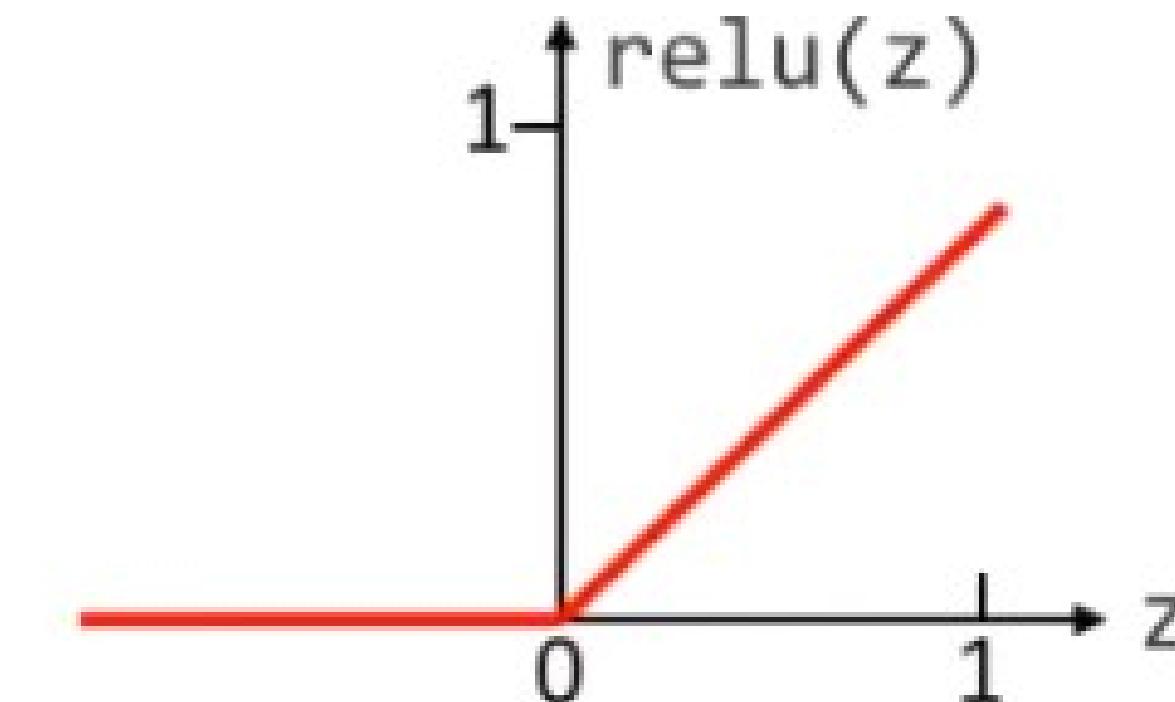


$$f(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})}$$

Funções de Ativação

Unidade Linear Retificada - ReLU

A vantagem da Relu é que a mesma **anula valores negativos** tendo como retorno da derivada sempre o valor 1, ou seja, a inclinação é sempre constante.



$$g(z) = \max(0, z)$$

same as

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

Funções de Ativação

Vantagens da **ReLU**:

A principal vantagem de usar a função ReLU sobre outras funções de ativação é que ***ela não ativa todos os neurônios ao mesmo tempo.***

O que isto significa ?

Se você olhar para a função **ReLU** e a entrada for negativa, ela será convertida em zero e o neurônio não será ativado. Isso significa que, ao mesmo tempo, apenas alguns neurônios são ativados, tornando a rede esparsa e eficiente e fácil para a computação

- **Tipos de Aprendizado**

Após entendermos um pouco sobre as funções de ativação, precisamos entender como se dá o aprendizado de uma rede neural.



• *Tipos de Aprendizado*

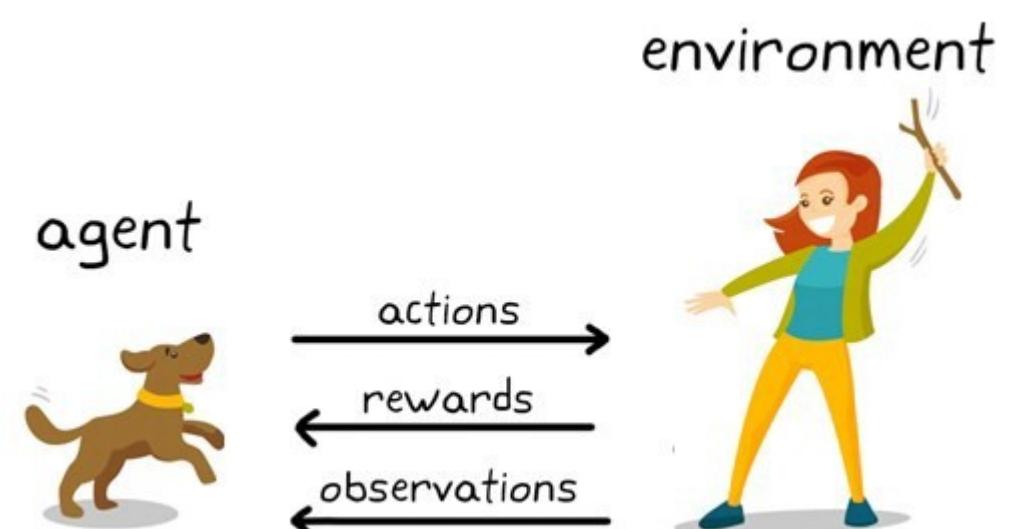
Aprendizado Supervisionado



Aprendizado Não-Supervisionado



Aprendizado Por Reforço



- *Tipos de aprendizado*

Aprendizado Supervisionado

O aprendizado supervisionado em redes neurais é uma abordagem em que o modelo é treinado usando um conjunto de dados rotulados. Isso significa que o conjunto de dados de treinamento inclui pares de entradas e suas correspondentes saídas desejadas (rótulos). O objetivo do modelo é aprender uma relação entre as entradas e as saídas de maneira a poder fazer especificidades específicas para novas entradas não vistas anteriormente



- *Tipos de aprendizado*

Aprendizado Supervisionado

Tecnicamente chamamos as entradas de:
ATRIBUTOS => Vetor de atributos.

As saídas de:
Classes.

- *Tipos de Aprendizado*

Aprendizado Supervisionado

Por exemplo:



Vamos descrever essas frutas através de 4 atributos:

ATRIBUTOS = [COR, PESO, TEXTURA, PH]

- *Tipos de Aprendizado*

Aprendizado Supervisionado

Por exemplo:



Obs: OS dados referentes aos atributos peso, textura e pH são hipotéticos.

$\underline{X}(k)$	Cor	Peso (g)	Textura	pH	$\underline{Y}(k)$	Fruta
$\underline{X}(1)$	Vermelha	113	Lisa	6,8	$\underline{Y}(1)$	Maçã
$\underline{X}(2)$	Laranja	122	Rugosa	4,7	$\underline{Y}(2)$	Laranja
$\underline{X}(3)$	Vermelha	107	Lisa	5,2	$\underline{Y}(3)$	Maçã
$\underline{X}(4)$	Vermelha	98	Lisa	3,6	$\underline{Y}(4)$	Maçã
$\underline{X}(5)$	Laranja	115	Rugosa	2,9	$\underline{Y}(5)$	Laranja
$\underline{X}(6)$	Laranja	120	Rugosa	4,2	$\underline{Y}(6)$	Laranja

Atributos

(entradas)

Classificação

(saída)

Notem que apenas os atributos de cor e textura para este exemplo seriam suficiente (problema de dimensionalidade).

Menos atributos significam redes neurais mais simples e eficientes E menor tempo de treinamento.

• *Tipos de Aprendizado*



Aprendizado Supervisionado

Por exemplo:

Obs: OS dados referentes aos atributos peso, textura e pH são hipotéticos.

$\underline{x}(k)$	Cor	Peso (g)	Textura	pH	$\underline{y}(k)$	Fruta
$\underline{x}(1)$	Vermelha	113	Lisa	6,8	$\underline{y}(1)$	Maçã
$\underline{x}(2)$	Laranja	122	Rugosa	4,7	$\underline{y}(2)$	Laranja
$\underline{x}(3)$	Vermelha	107	Lisa	5,2	$\underline{y}(3)$	Maçã
$\underline{x}(4)$	Vermelha	98	Lisa	3,6	$\underline{y}(4)$	Maçã
$\underline{x}(5)$	Laranja	115	Rugosa	2,9	$\underline{y}(5)$	Laranja
$\underline{x}(6)$	Laranja	120	Rugosa	4,2	$\underline{y}(6)$	Laranja

Notação matemática:

$$\underline{x}(3) = [\text{Vermelha}, 107, \text{Lisa}, 5,2]$$

→ vetor de atributos ou *feature vector*



$$\underline{y}(3) = \text{Maçã}$$

Classe
Saída desejada
Alvo (*target*)

• *Tipos de Aprendizado*

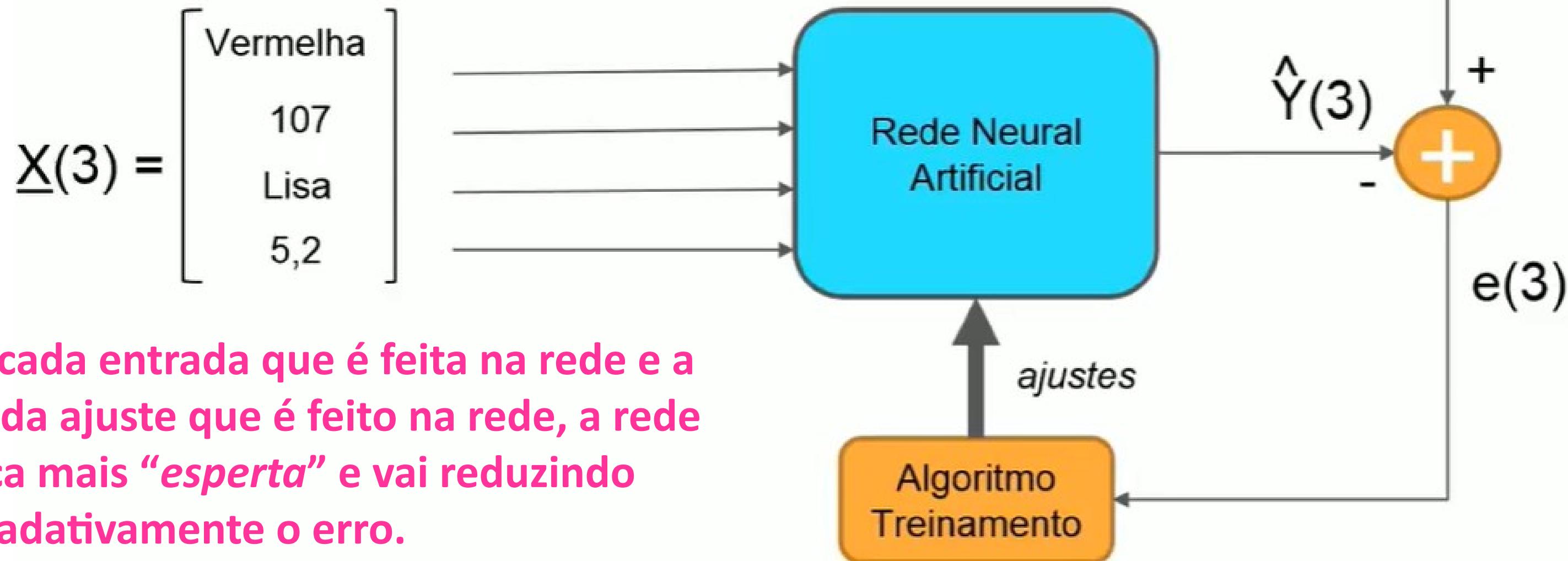


Aprendizado Supervisionado

Por exemplo:

Notação matemática: (\underline{x}, y)

$\hat{Y}(3) = \text{maçã}$



A cada entrada que é feita na rede e a cada ajuste que é feito na rede, a rede fica mais “esperta” e vai reduzindo gradativamente o erro.

• *Tipos de Aprendizado*

Aprendizado Não-Supervisionado

O aprendizado não supervisionado em redes neurais refere-se a um tipo de treinamento de rede em que o modelo é alimentado com dados não rotulados, e o objetivo principal é descobrir padrões intrínsecos ou estruturas nos dados sem orientação externa sobre o que procurar. Em outras palavras, a rede é deixada para explorar e aprender as características básicas dos dados por conta própria.



- *Tipos de Aprendizado*

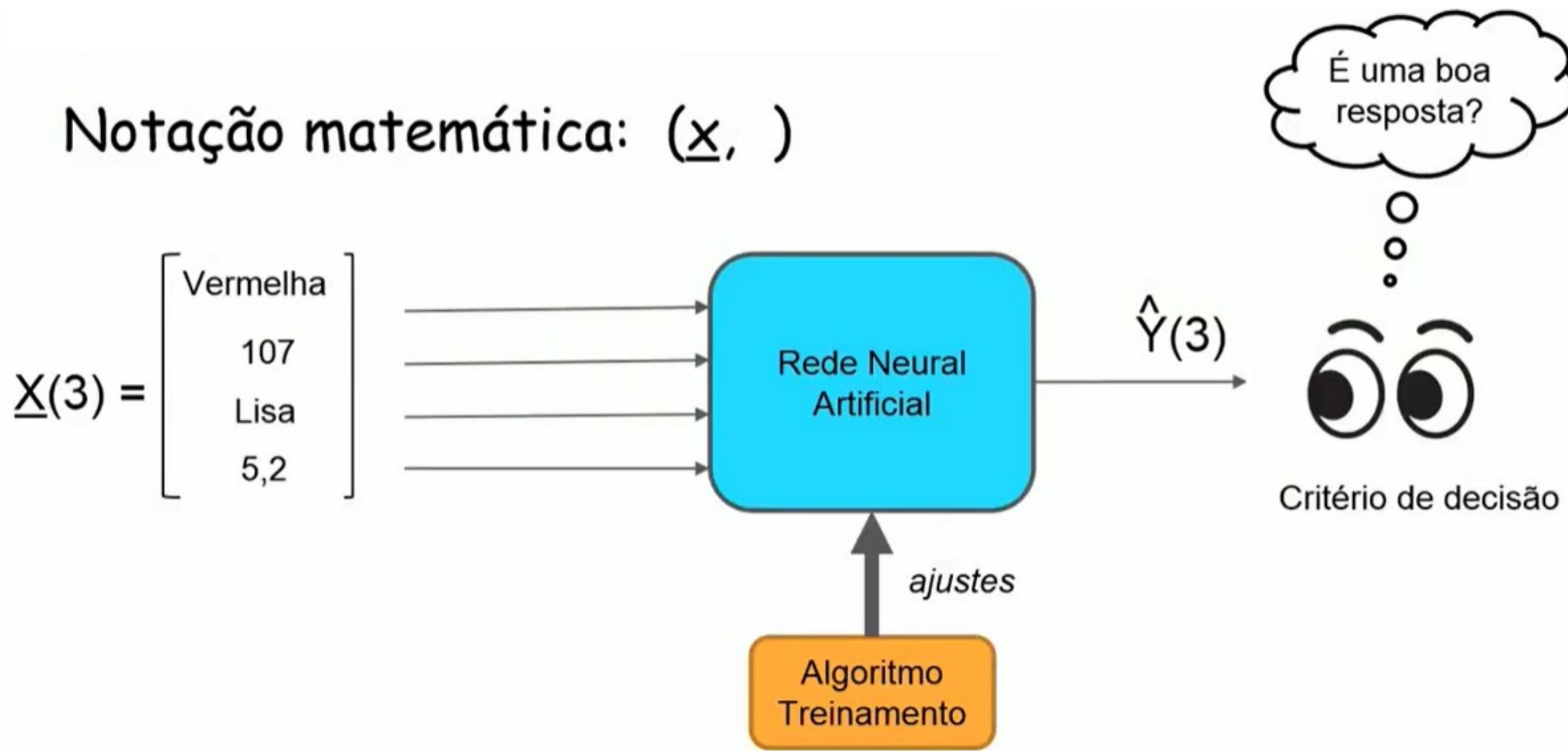
Aprendizado Não-Supervisionado

Agrupamento de Clientes em um Shopping

Suponha que você tenha dados de transações de clientes em uma compra, mas esses dados **não têm rótulos indicados na categoria de cada cliente**. Você deseja segmentar os clientes em grupos diferentes com base em seus **padrões de compra** para poder personalizar estratégias de marketing para cada grupo.

• *Tipos de Aprendizado*

Aprendizado Não-Supervisionado



- *Tipos de Aprendizado*

Aprendizado Por Reforço

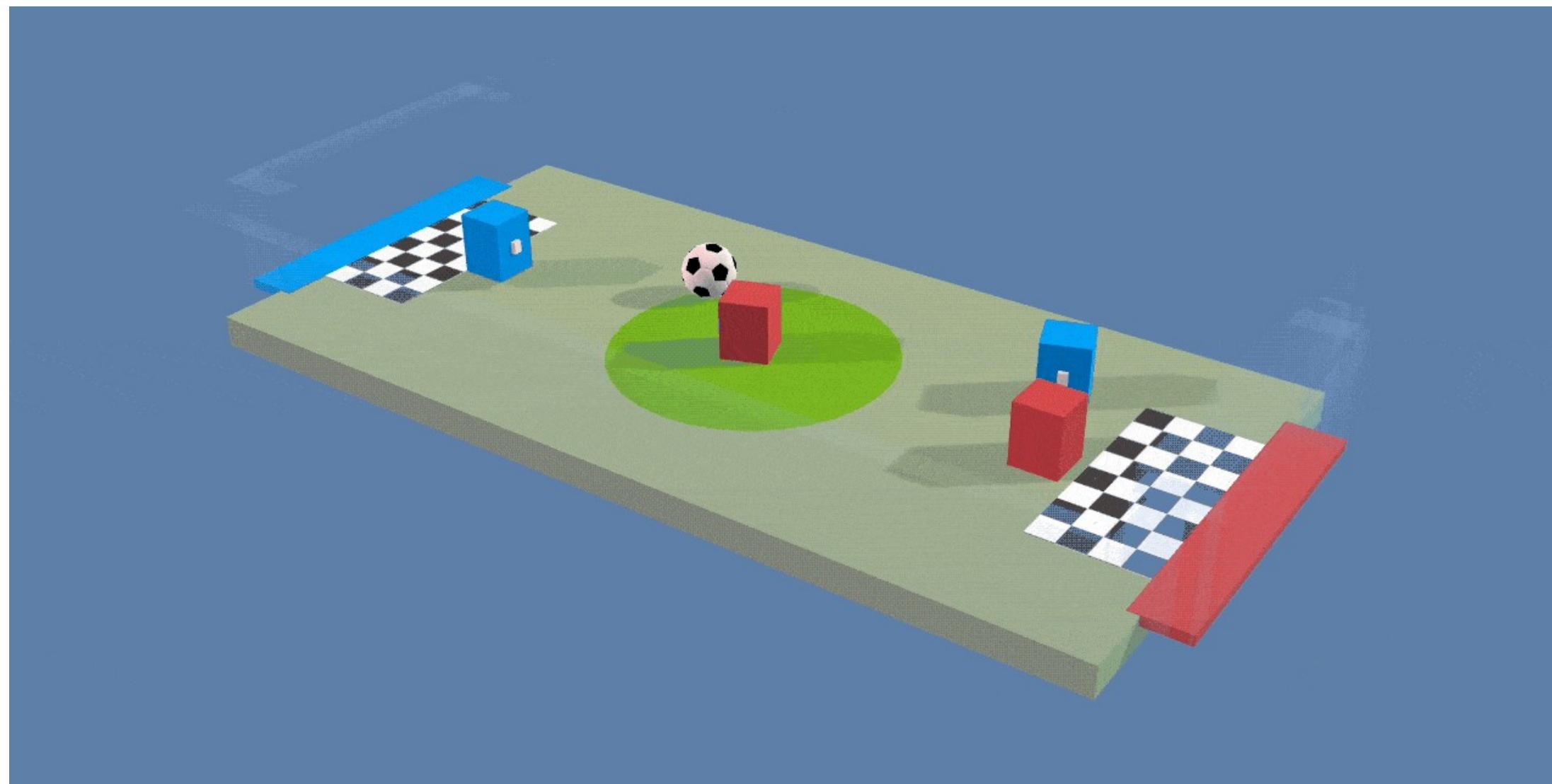
O aprendizado por reforço (RL) é um paradigma de aprendizado de máquina em que **um agente interage com um ambiente dinâmico para aprender a realizar ações que maximizam uma noção de recompensa cumulativa**. Esse tipo de aprendizado é **inspirado na psicologia comportamental**, onde um agente é **incentivado ou desencorajado com base nas consequências de suas ações**.

Em contextos de redes neurais, o aprendizado por reforço pode ser aplicado usando arquiteturas específicas, como Redes Neurais Profundas (Deep Neural Networks), para representar políticas do agente ou para estimar funções de valor.

• *Tipos de Aprendizado*

Aprendizado Por Reforço

É uma técnica que tem como objetivo treinar um **Agente** (programa, código, algoritmo, etc.) a interagir em um **Ambiente** por meio de **Ações** para atingir um **Objetivo**. Por meio de **Recompensas** ou **punições** dadas a esse agente, ele irá aprender quais ações deve executar para aumentar a recompensa e atingir o objetivo.



- Qual das três Abordagens é a Melhor?



A resposta a essa pergunta depende do que você está analisando.
Cada problema possui suas peculiaridades.

Hierarquia do Aprendizado

Conforme visto em estudos anteriores, o aprendizado de máquinas **obedece uma hierarquia segundo os tipos de tarefas de aprendizado.**

Alguns algoritmos utilizam tarefas que induzem ao **modelo preditivo**, ou seja, seguem o paradigma do aprendizado supervisionado.

As tarefas supervisionadas se distinguem pelo tipo de rótulos dos dados, **discreta** no caso de **classificação** e **contínuos** no caso de **regressão Linear**.

Hierarquia do Aprendizado

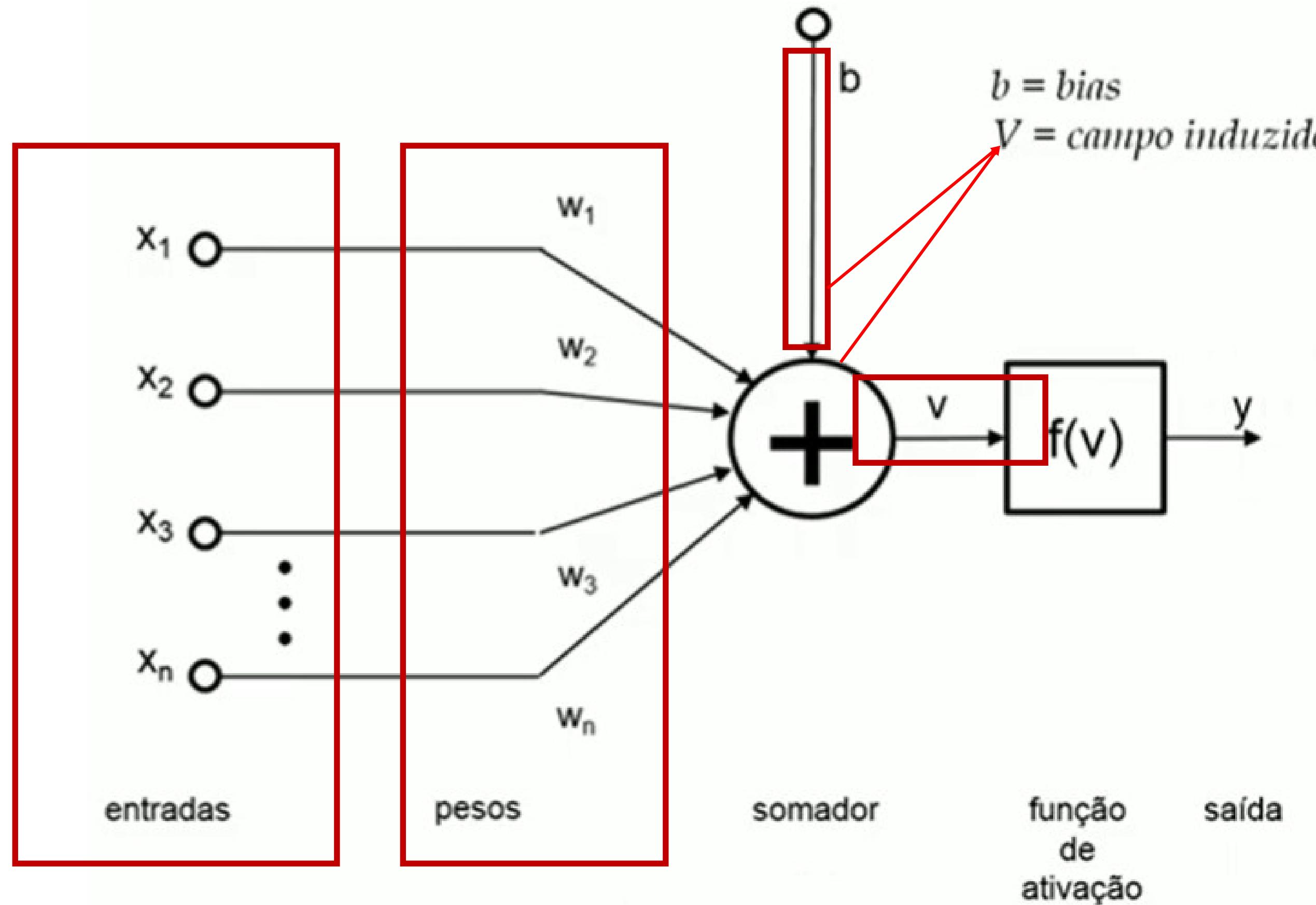
Já alguns algoritmos de aprendizado, utilizam tarefas de **descrição**, não utilizando-se de atributos de saída, sendo assim, seguem o paradigma de aprendizado **não supervisionado**.

Uma tarefa descritiva de **agrupamento** de dados por exemplo, tem por meta encontrar grupos de objetos semelhantes enquanto que as **regras de associação** um grupo de atributo a outro grupo de atributos.

Hierarquia do Aprendizado

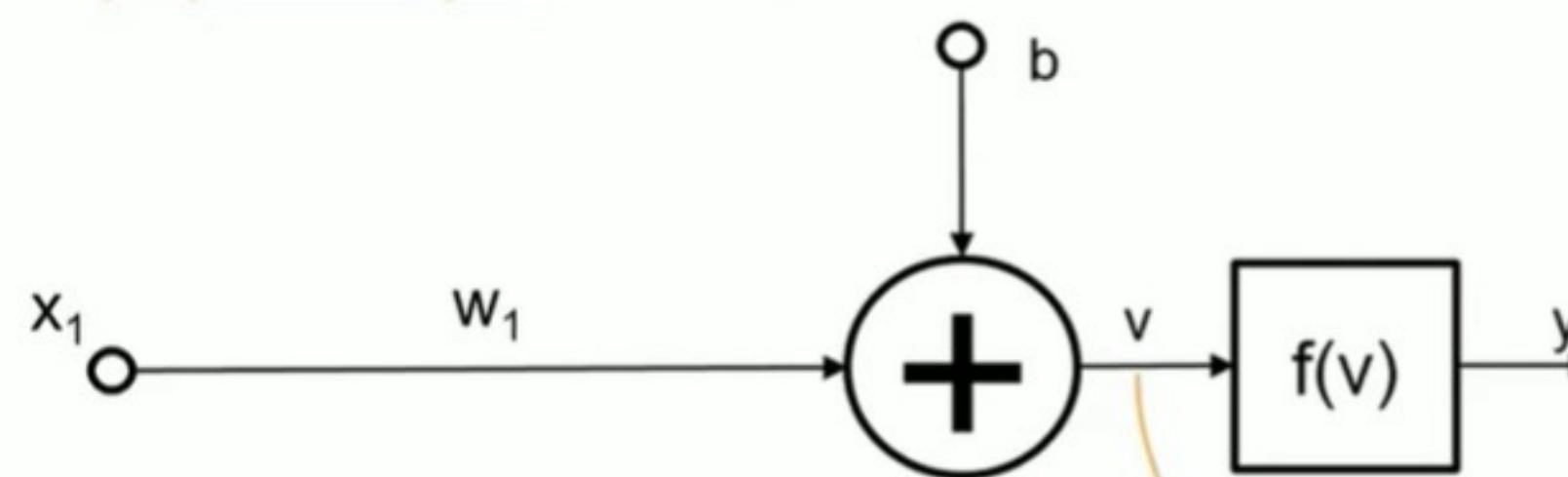


O Papel do Bias e o Modelo Matemático de um Neurônio



Papel do Bias

O "bias" é um elemento que serve para aumentar o grau de liberdade dos ajustes dos pesos.



Para que serve o coeficiente angular?

Serve para girar a reta inteira no sentido horário ou anti-horário

Para que serve o coeficiente linear?

Responsável por posicionar na reta na origem dos eixos do gráfico, acima ou abaixo.

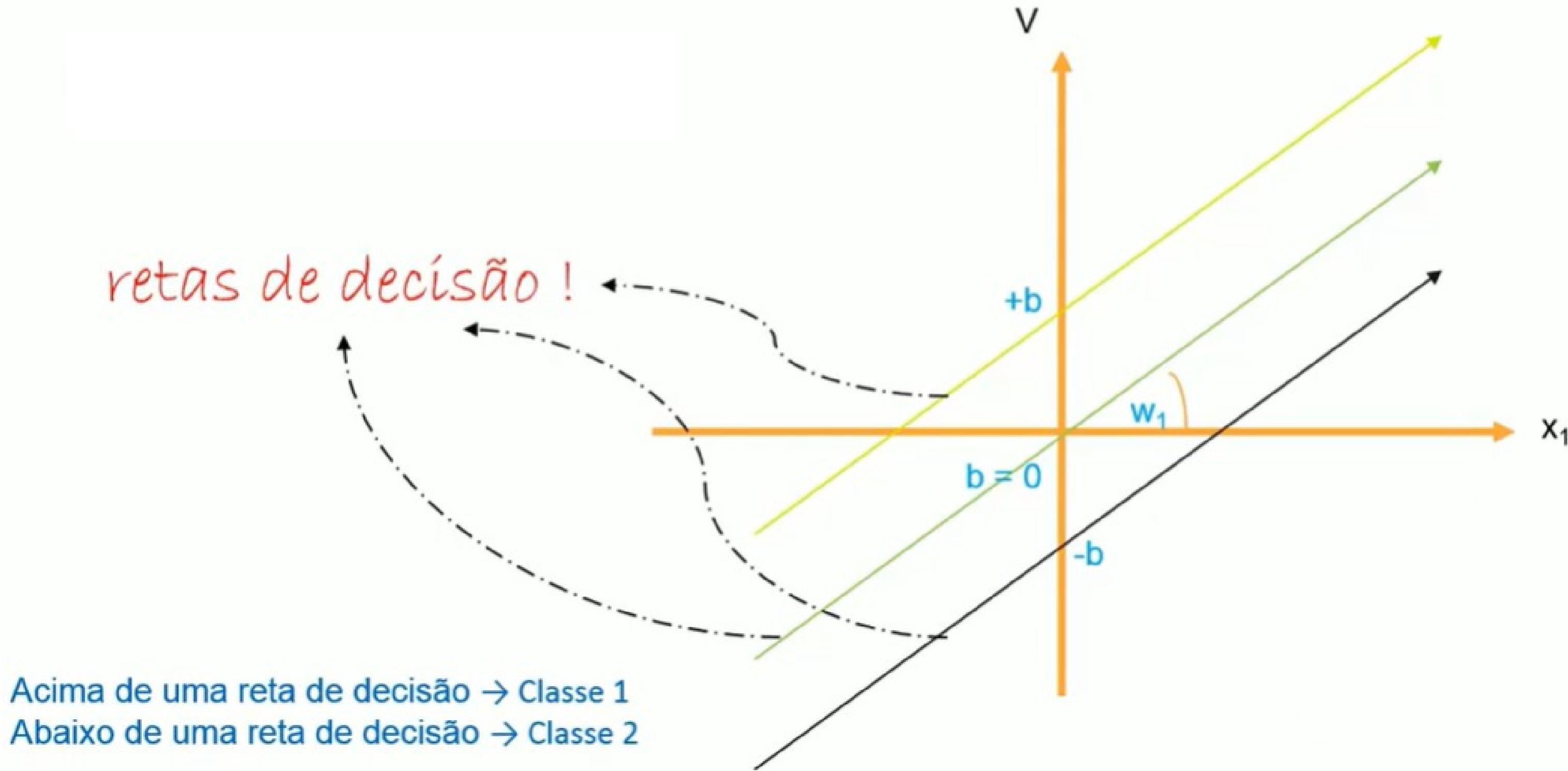
$$v = w_1 x_1 + b \quad (\text{equação da reta})$$

Coeficiente linear

Coeficiente angular

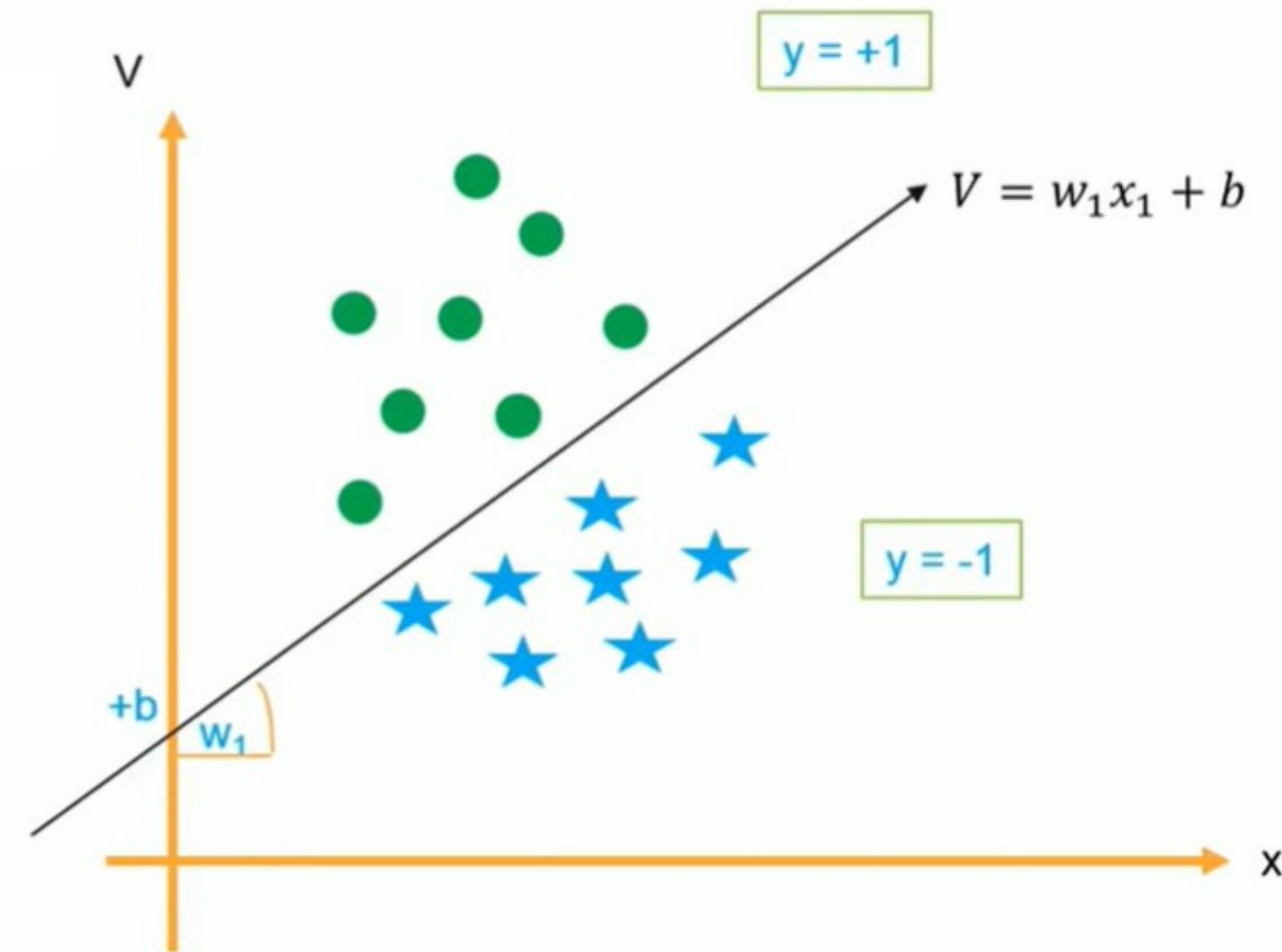
Papel do Bias

retas de decisão !



Papel do Bias

Vamos imaginar que queremos classificar um objeto unidimensional, ou seja, **possui uma única classe**.



Aqui vamos usar pontos verdes e estrelinhas azuis como objetos meramente ilustrativos.

E para classifica-los podemos utilizar qualquer grandeza (temperatura, peso, comprimento). E vamos chamar essa grandeza de x_1 .

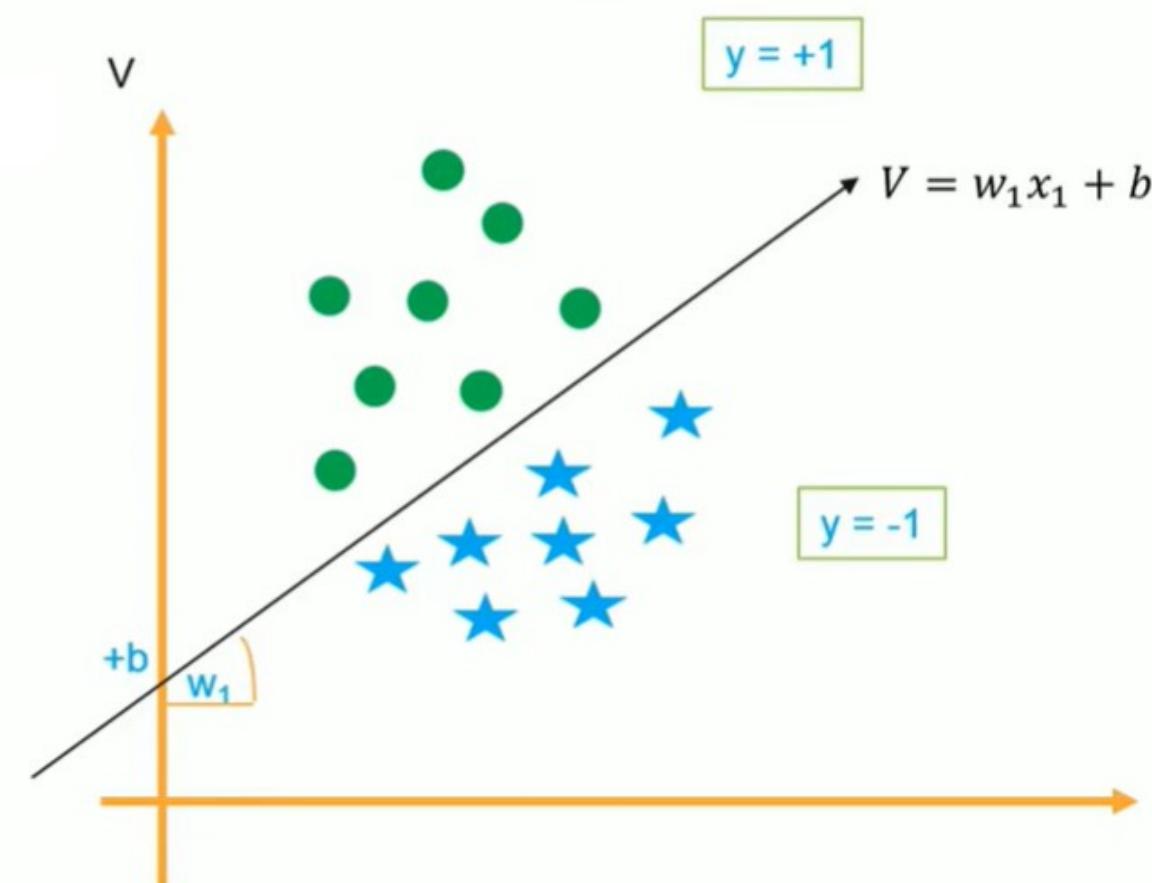
Papel do Bias

Após medir o **atributo x_1** , de todos os objetos a serem classificados, **entregamos** os resultados para nosso **neurônio Perceptron com uma única entrada**.

A cada dado que fornecemos comparamos a resposta dada pelo neurônio com a resposta verdadeira.

Se o algoritmo **acertou** nada acontecerá em **termos de treinamento**.

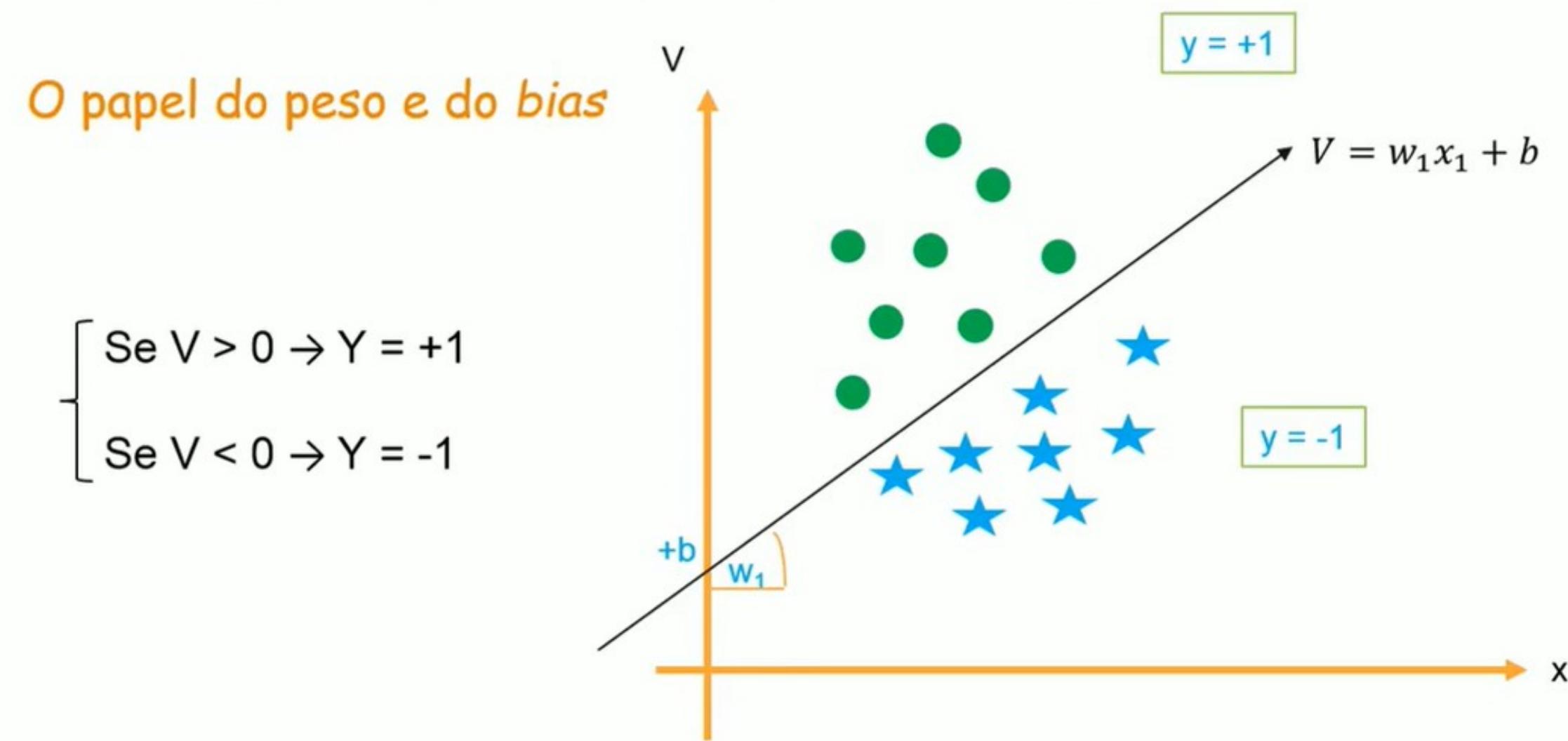
Se o algoritmo **errou** o algoritmo de treinamento **ira ajustar o peso do bias e das sinapses** de modo a diminuir o erro na próxima etapa de treinamento.



Papel do Bias

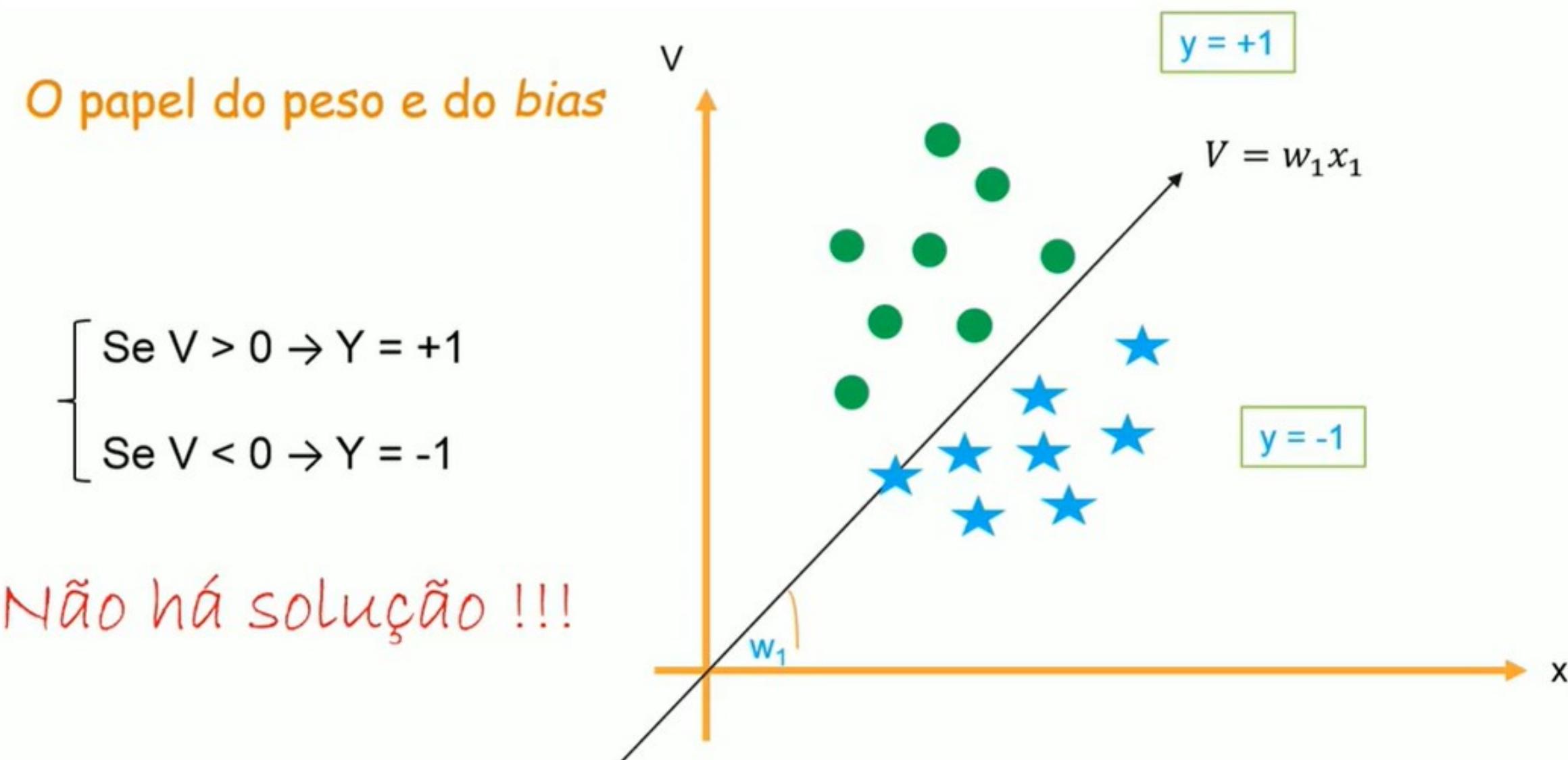
O que isso significa?

Significa que o algoritmo de treinamento irá girar a reta no sentido horário ou anti-horário *E* subir ou descer mais a reta de modo que ao final do treinamento uma reta seja encontrada de modo a separar todos os objetos corretamente.



Papel do Bias

Se neste mesmo exemplo caso o bias fosse nulo!



Tecnicamente então podemos dizer que o Bias aumenta o grau de liberdade do nosso sistema.