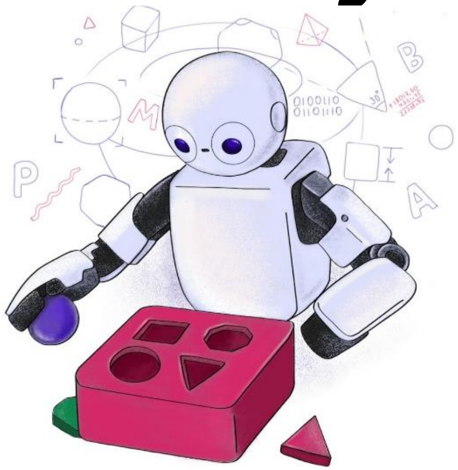
# T320 - Introdução ao Aprendizado de Máquina II:
# *Redes Neurais Artificiais (Parte I)*



Inatel

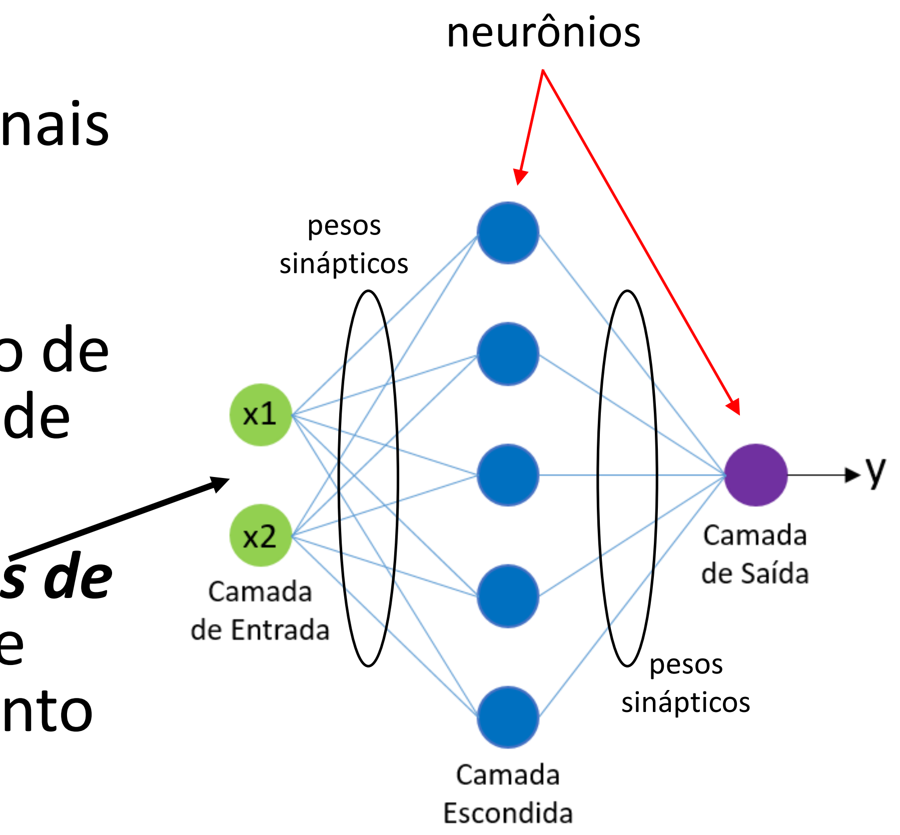
Felipe Augusto Pereira de Figueiredo
felipe.figueiredo@inatel.br

# Introdução

- A partir desta aula, começamos a discutir a respeito de um tópico que parece, inicialmente, não ser relacionado com a disciplina: o cérebro.
- Entretanto, como veremos a seguir, as ideias que discutimos até agora serão úteis na construção de ***modelos matemáticos que aproximam a atividade de aprendizagem do cérebro***.
- E, como veremos, essas ideias que já discutimos, nos ajudarão a entender o funcionamento das ***redes neurais artificiais*** (RNAs).
- Redes neurais artificiais são uma das formas mais populares e efetivas para implementação de sistemas de aprendizado de máquina e mereceriam por sí só uma disciplina em separado.
- Portanto, neste tópico, veremos uma breve visão geral sobre as RNAs.

# Redes Neurais Artificiais

- ***Redes neurais artificiais*** são modelos computacionais inspirados pelo funcionamento do cérebro dos animais.
- Elas são capazes de realizar tarefas de aprendizado de máquina (e.g., regressão e classificação) com grande eficácia.
- RNAs são geralmente apresentadas como ***sistemas de nós (unidades ou neurônios) interconectados***, que geram valores de saída, simulando o comportamento de ***redes neurais biológicas***.
- Esta primeira parte deste tópico, foca nos elementos básicos de construção de uma rede neural, os ***nós*** ou ***neurônios***.



# Algumas aplicações famosas

- RNAs são versáteis, poderosas e escalonáveis, tornando-as ideais para realizar tarefas grandes e altamente complexas de ***aprendizado de máquina***, como por exemplo:
  - Classificar bilhões de imagens (por exemplo, como o Google Images, Facebook, etc. fazem),
  - Serviços de reconhecimento de fala (por exemplo, a Siri da Apple, Alexa da Amazon e Google Assistant),
  - Recomendar vídeos que melhor se adequam ao comportamento de centenas de milhões de usuários todos os dias (por exemplo, YouTube, Netflix),
  - Pilotar um veículo com pouca ou nenhuma intervenção humana.

Google images

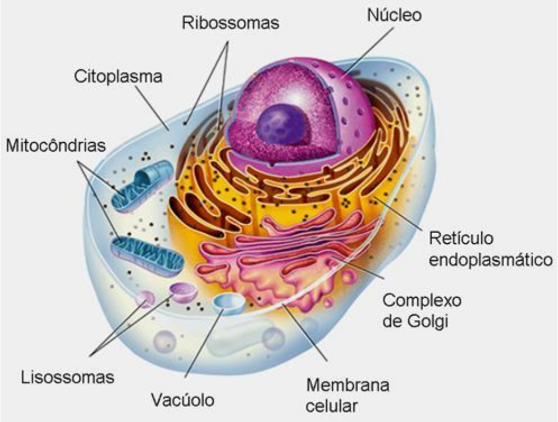
Alexa Google Assistant Siri

YouTube

NETFLIX

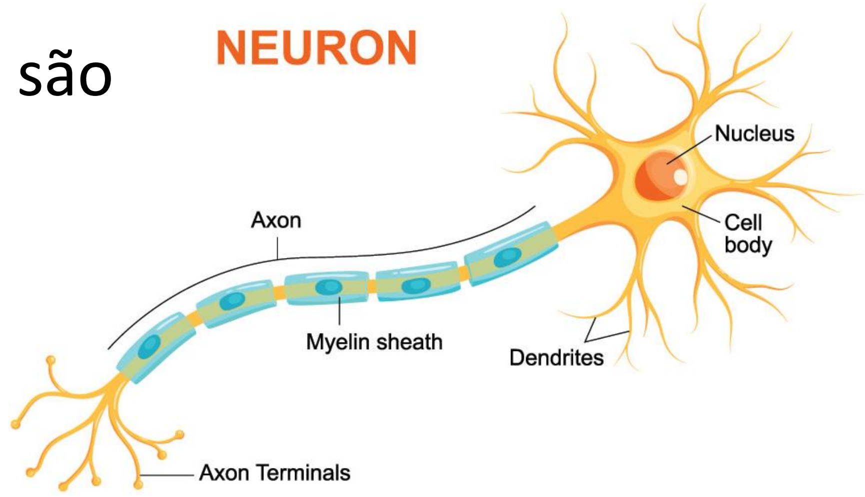
WAYMO

# Um pouco de contexto




- A descoberta da célula em 1665 por Robert Hooke foi importantíssima para que houvesse uma melhor compreensão da estrutura dos seres vivos.
- Podemos considerar a célula como sendo o ***átomo da vida***.
- Células podem ser classificadas em ***procariontes*** e ***eucariontes***.
- Células ***procariontes*** têm uma estrutura simples e não possuem núcleo (e.g., bactérias).
- As células ***eucariontes*** (plantas, animais, fungos, protozoários, algas, e amebas) possuem três partes principais: ***membrana***, ***citoplasma*** e ***núcleo***.
  - A ***membrana*** "delimita a célula", i.e., ela isola seu interior do meio externo.
  - O ***citoplasma*** é o espaço intracelular entre a membrana e o núcleo.
    - Ele é preenchido pelo ***citosol*** onde estão suspensas as ***organelas*** (e.g., mitocôndrias, lisossomos, etc.).
  - Já o ***núcleo*** controla as atividades celulares e armazena a maior parte da informação genética (DNA) da célula.

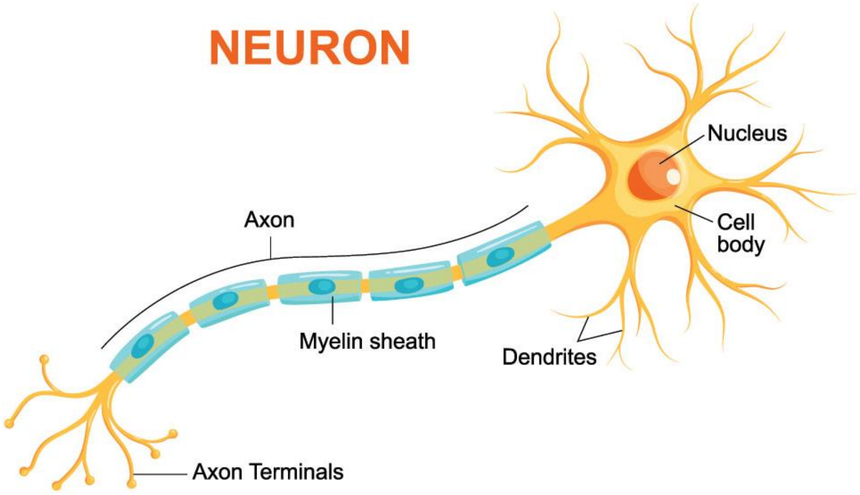
# Um pouco de contexto




- Os ***neurônios*** são células ***eucariontes*** também, mas são células que possuem ***mecanismos eletroquímicos*** característicos.
- Os neurônios apresentam três partes básicas: os ***dendritos***, o ***axônio*** e o ***corpo celular (soma)***.
- Os ***dendritos*** são prolongamentos do neurônio que garantem a ***recepção de estímulos*** de outros neurônios, levando impulsos nervosos em direção ao ***corpo celular***.
- O ***axônio*** é um prolongamento que garante o envio de informação (estímulos) a outros neurônios através de seus ***terminais***.
- Cada neurônio possui apenas um axônio, o qual é, geralmente, mais longo que os dendritos.

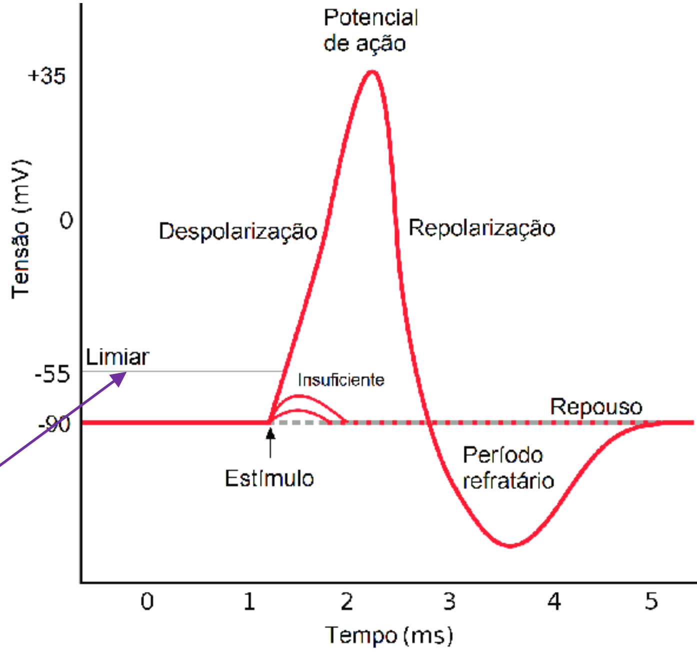
# Um pouco de contexto

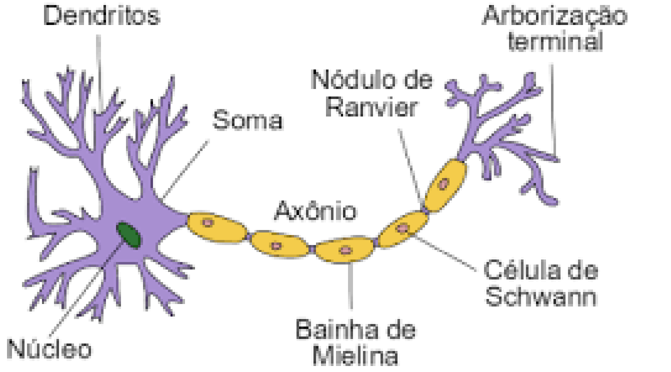
- O ***corpo celular*** (também conhecido como ***soma***) contém o núcleo do neurônio e é responsável por realizar a ***integração*** dos estímulos recebidos pelo neurônio através de seus dendritos.
- Os pontos de contato entre os dentritos de um neurônio e os terminais do axônio de outro neurônio são chamados de ***sinapses***.
- Ou seja, os neurônios se comunicam uns com os outros através das ***sinapses***.
- Sinapses podem ser químicas, as mais comuns, ou elétricas, muito pouco comuns.
- As figuras ao lado mostram o esquema de um ***neurônio*** e uma sinapse química.

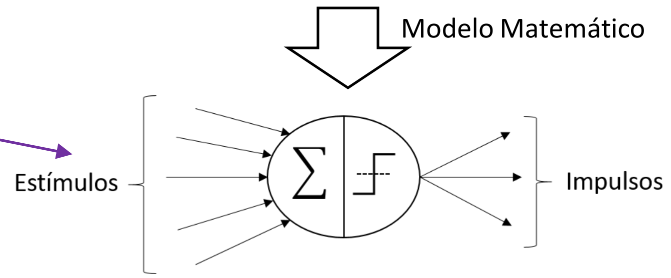



# Um pouco de contexto

- Em termos bem simples, mas lembrando de que existem exceções, nós podemos simplificar o funcionamento do ***neurônio*** como:
  - O neurônio recebe estímulos elétricos, basicamente a partir dos dendritos.
  - Esses estímulos são somados no corpo celular (*soma*).
  - Se a soma dos estímulos exceder um certo ***limiar de ativação***, o ***neurônio*** gera um pulso (ou ***potencial de ação***) que é enviado pelos terminais do axônio a outros neurônios.
- Um ***neurônio*** pode se conectar a até 20.000 outros ***neurônios*** através das ***sinapses***.
- Sinais são passados de ***neurônio*** para ***neurônio*** através de ***reações eletroquímicas***.
- Do ponto de vista do nosso curso, o ***neurônio*** será considerado como um ***sistema com várias entradas e uma ou mais saídas*** onde a comunicação entre neurônios é feita através de sinais elétricos.



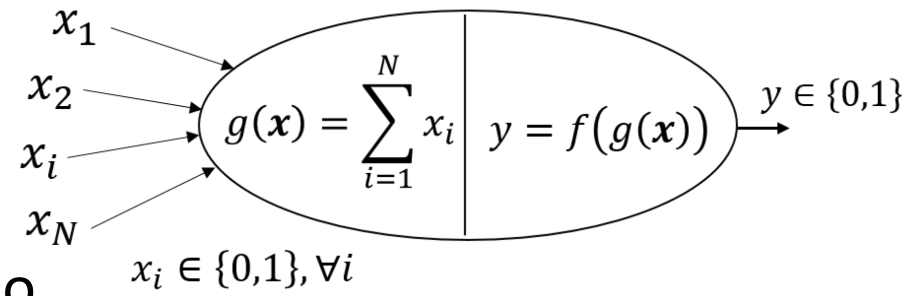
# O Modelo de McCulloch e Pitts

Walter Pitts e
Warren McCulloch

- O final do século XIX e o início do século XX foram períodos fundamentais para o estabelecimento do conhecimento atual do sistema nervoso.
- De posse desse entendimento, em 1943, dois neurocientistas, Warren McCulloch e Walter Pitts apresentam em um artigo científico o primeiro ***modelo computacional*** de um neurônio.
- A partir desse modelo, foi possível estabelecer uma conexão entre o funcionamento de um neurônio e a ***lógica proposicional***.
- ***Lógica proposicional*** se baseia em ***proposições***.
    - Uma ***proposição*** é uma ***sentença declarativa*** ou ***afirmação***, ou seja, é uma sentença que faz uma ***afirmação*** sobre um fato, podendo este ser verdadeiro ou falso.
- O artigo de McCulloch e Pitts fornece *insights* fundamentais sobre como a ***lógica proposicional*** pode ser processada por um neurônio.
- Existe uma correspondência direta entre a lógica proposicional e a lógica Booleana.
    - Podemos pensar em uma ***sentença declarativa*** como sendo uma ***expressão Booleana***
        - 1 ou 1 = 1
        - 1 e 0 = 0
- A partir desta correspondência, a relação com a computação foi direta e natural.

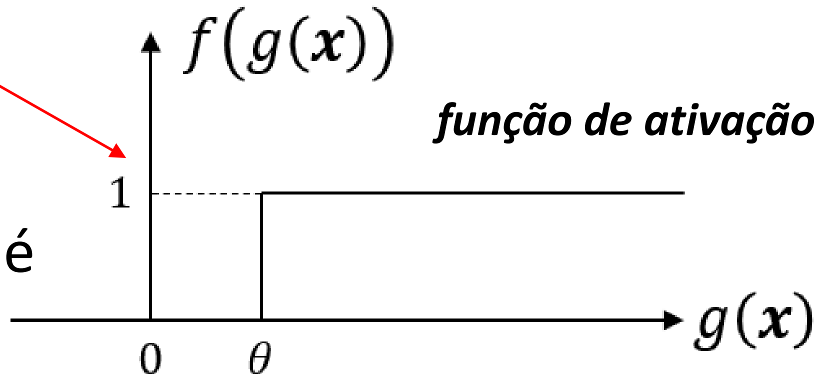
# O Modelo de McCulloch e Pitts

- A figura ao lado apresenta o modelo matemático do ***neurônio*** proposto por McCulloch e Pitts.
- Grosso modo, o ***neurônio*** é ativado (ou disparado) quando a ***combinação linear*** de suas entradas excede o ***limiar de ativação***, $\theta$.
- As premissas do modelo de McCulloch e Pitts (M-P) são:
  - Os valores das entradas, $x_i, \forall i$, ou também chamados de ***sinapses***, são sempre valores booleanos, i.e., '0', ou '1'.
  - As entradas são multiplicadas por pesos unitários (+/- 1).
  - A atividade do ***neurônio*** é um processo do tipo "***tudo ou nada***", ou seja, um processo binário (0 ou 1).
  - Portanto, a ***função de ativação*** do neurônio é uma ***função degrau*** com ***ponto de disparo*** dependente do ***limiar de ativação***, $\theta$.
  - Um certo número de ***sinapses*** deve ser excitado para que o neurônio "dispare".
- O modelo do ***neurônio*** de McCulloch e Pitts nada mais é do que um ***classificador linear com limiar de decisão rígido***, ***pesos unitários*** e ***atributos booleanos***.




$$y = f(g(\boldsymbol{x})) = \begin{cases} 1, \text{se } g(\boldsymbol{x}) \geq \theta \\ 0, \text{se } g(\boldsymbol{x}) < \theta \end{cases}$$

onde $\theta$ é o ***limiar de ativação***.

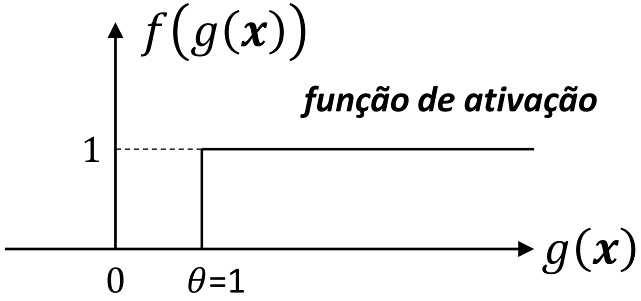


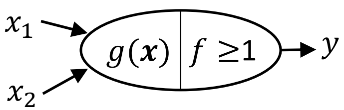
# Exemplos de portas lógicas com o modelo M-P

Podem ser interpretados como problemas de classificação.

| OR | | | |
|---|---|---|---|
| $x_1$ | $x_2$ | $g(\boldsymbol{x})$ | $y$ |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 2 | 1 |

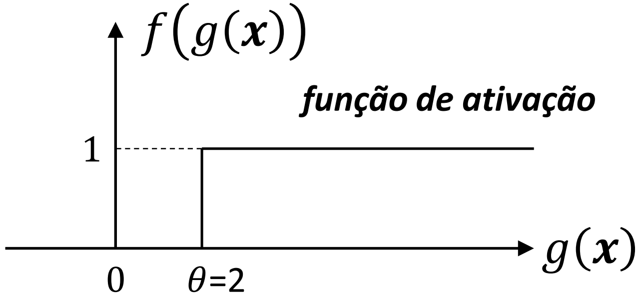
- Qual é o valor do ***limiar de ativação***, $\theta$?
- Analisando-se $g(\boldsymbol{x})$, vemos que o disparo deve ocorrer quando $g(\boldsymbol{x}) \geq 1$, portanto, $\theta = 1$.

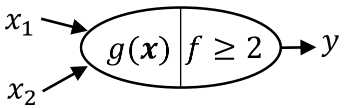



| AND | | | |
|---|---|---|---|
| $x_1$ | $x_2$ | $g(\boldsymbol{x})$ | $y$ |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 2 | 1 |

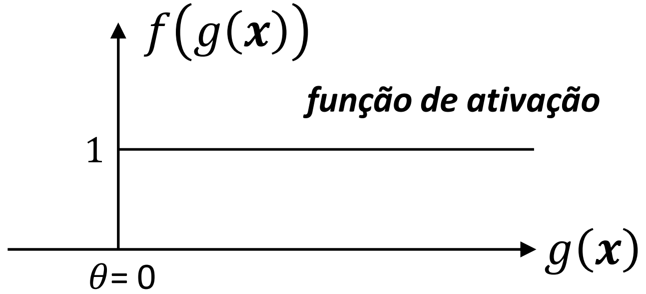
- Qual é o valor do ***limiar de ativação***, $\theta$?
- Analisando-se $g(\boldsymbol{x})$, vemos que o disparo deve ocorrer quando $g(\boldsymbol{x}) \geq 2$, portanto, $\theta = 2$.

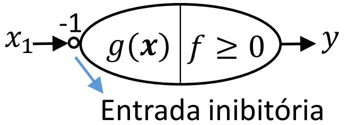



| NOT | | | |
|---|---|---|---|
| $x_1$ | $-x_1$ | $g(\boldsymbol{x})$ | $y$ |
| 0 | 0 | 0 | 1 |
| 1 | -1 | -1 | 0 |

- Qual é o valor do ***limiar de ativação***, $\theta$?
- Analisando-se $x_1$, vemos que para o disparo ocorrer, seu valor deve ser ***negado*** (i.e., multiplicado por -1), e assim, o disparo ocorre quando $g(\boldsymbol{x}) \geq 0$, portanto, $\theta = 0$.




**OBS.**: Entradas inibitórias são entradas que têm seus valores multiplicados por -1.

# Exemplos de portas lógicas com o modelo M-P

- Qual deve ser o valor do ***limiar de ativação***, $\theta$, para a porta lógica XOR?

Sem entradas inibitórias.

| XOR | | | |
|---|---|---|---|
| $x_1$ | $x_2$ | $g(\boldsymbol{x})$ | $y$ |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 2 | 0 |

$0 < g(\boldsymbol{x}) < 2$

$x_1$ como entrada inibitória.

| XOR | | | |
|---|---|---|---|
| $-x_1$ | $x_2$ | $g(\boldsymbol{x})$ | $y$ |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| -1 | 0 | -1 | 1 |
| -1 | 1 | 0 | 0 |

$g(\boldsymbol{x}) \leq -1$ ou $g(\boldsymbol{x}) \geq 1$

$x_2$ como entrada inibitória.

| XOR | | | |
|---|---|---|---|
| $x_1$ | $-x_2$ | $g(\boldsymbol{x})$ | $y$ |
| 0 | 0 | 0 | 0 |
| 0 | -1 | -1 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | -1 | 0 | 0 |

$g(\boldsymbol{x}) \leq -1$ ou $g(\boldsymbol{x}) \geq 1$

$x_1$ e $x_2$ como entradas inibitórias.

| XOR | | | |
|---|---|---|---|
| $-x_1$ | $-x_2$ | $g(\boldsymbol{x})$ | $y$ |
| 0 | 0 | 0 | 0 |
| 0 | -1 | -1 | 1 |
| -1 | 0 | -1 | 1 |
| -1 | -1 | -2 | 0 |

$-2 < g(\boldsymbol{x}) < 0$

- **Resposta**: com um único modelo de M-P, não é possível encontrar um ***limiar de ativação*** que resolva este problema, pois como veremos adiante, este problema não é ***linearmente separável***.
- O modelo de M-P só resolve problemas ***linearmente separáveis***.

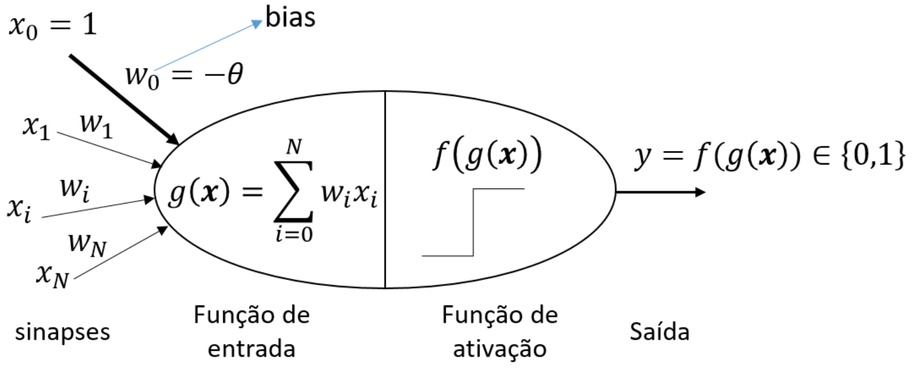
# Tarefa

- **Quiz**: "*T320 - Quiz – Redes Neurais Artificiais (Parte I)*" que se encontra no MS Teams.

# Perceptron

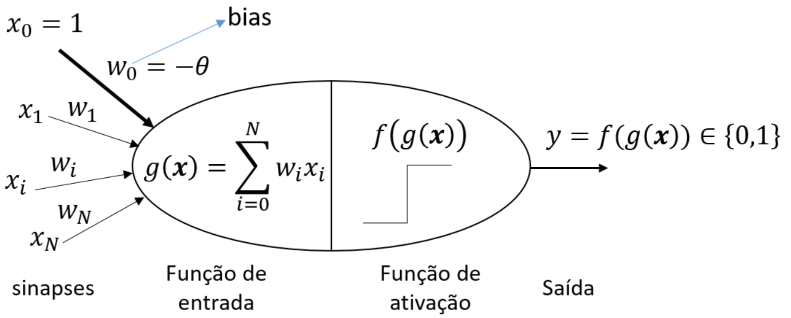
- Em 1958, Frank Rosenblatt, propôs um novo ***modelo computacional mais geral*** que o modelo do ***neurônio*** de McCulloch e Pitts.
- O modelo criado por ele é chamado de ***perceptron*** e é mostrado na figura ao lado.
- O ***perceptron*** é um modelo para ***aprendizado supervisionado*** de ***classificadores binários***, ou seja ***problemas com duas classes***.
- Assim como o modelo de M-P, o ***perceptron*** só é capaz de classificar padrões ***linearmente separáveis***.
- Ou seja, o ***perceptron*** também não resolve o problema da classificação XOR.


Frank Rosenblatt



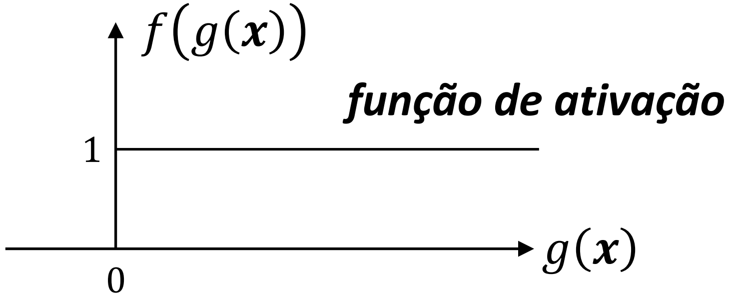
# Perceptron

- Esse novo modelo supera algumas das limitações do modelo de M-P:
  - Introdução do conceito de ***pesos sinápticos*** (uma medida de importância dos atributos) para as entradas (ou ***sinapses***).
  - E um método para que o modelo aprenda os ***pesos***.
- Além disso, as entradas não são mais limitadas a valores booleanos, como no caso do modelo de M-P, suportando ***entradas com valores reais***, o que torna este modelo mais útil e generalizado.
- Assim como no modelo de M-P, a ***função de ativação*** utilizada pelo ***perceptron*** também é a ***função degrau*** com a diferença que aqui ela não mais depende do ***limiar de ativação***, $\theta$.

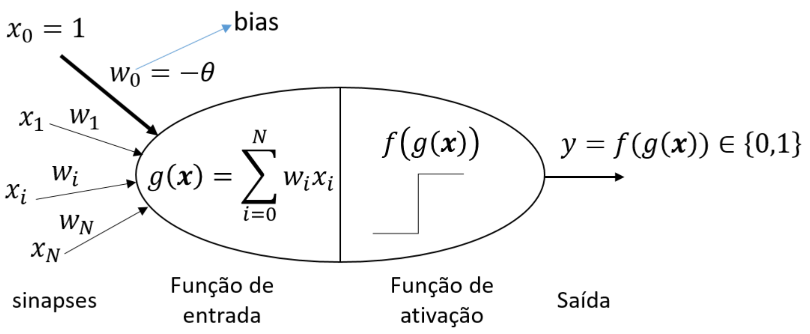



$$y = f\big(g(\boldsymbol{x})\big) = \begin{cases} 1, \text{se } g(\boldsymbol{x}) \geq 0 \\ 0, \text{se } g(\boldsymbol{x}) < 0 \end{cases}$$

Percebam que o ***limiar de ativação***, $\theta$, agora faz parte das entradas e é chamado de ***bias***.



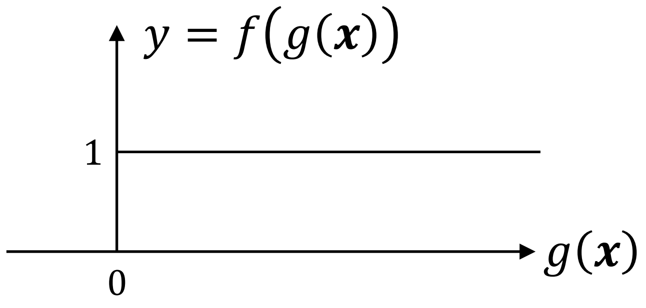
# Perceptron




- A ativação do ***perceptron*** é causada pela ***combinação linear*** dos ***estímulos de entrada*** em relação aos ***pesos sinápticos***.
  - Se a ***combinação linear*** exceder o ***limiar de ativação***, $\theta$, o ***disparo*** ocorre.
  - Isso é expresso por uma ***função de ativação*** do tipo ***degrau***.
- Notem que a ***função de ativação***, $f(.)$, tem a transição para o valor 1 quando $g(\boldsymbol{x}) = 0$ e o ***limiar de ativação*** é controlado, indiretamente, pelo valor do ***peso de bias***, $w_0$.
  - O ***limiar de ativação*** foi absorvido pela combinação linear, $g(\boldsymbol{x})$, e, portanto, podemos usar a ***função de ativação*** com transição fixa em zero, pois agora, ajusta-se o limiar de ativação indiretamente, através da atualização do peso $w_0$.
- Como podemos ver, a ***função discriminante***, $g(\boldsymbol{x})$, do ***perceptron*** tem a forma de um ***hiperplano***

$$g(\boldsymbol{x}) = \sum_{i=0}^{N} w_i x_i. \quad \text{(combinação linear das entradas)}$$

- Portanto, como já sabemos, este tipo de função dá origem a um ***classificador binário*** onde as classes são separadas por uma ***superfície de separação linear***.




Para que tenhamos $y = 1$, então

$$w_0 + \sum_{i=1}^{N} w_i x_i \geq 0 \therefore \sum_{i=1}^{N} w_i x_i \geq -w_0$$

Por exemplo

- Se $w_0 = 1, \sum_{i=1}^{N} w_i x_i \geq -1$.
- Se $w_0 = -1, \sum_{i=1}^{N} w_i x_i \geq 1$.

# Regra de aprendizado do perceptron

- Devido ao fato da ***função degrau***, $f(g(\boldsymbol{x}))$, ter derivada igual a zero em todos os pontos, exceto em $g(\boldsymbol{x}) = 0$, onde ela é indefinida, nós não podemos utilizar o ***gradiente descendente***.
- Entretanto, como aprendemos anteriormente, usamos a ***regra de aprendizado do perceptron*** para treinar o modelo.
- É uma regra ***simples e intuitiva*** para atualização dos pesos do modelo.
- No caso do perceptron, onde $g(\boldsymbol{x})$ é um ***hiperplano***, a regra converge para uma solução perfeita se as classes forem ***linearmente separáveis***:
  - Classes ***suficientemente espaçadas*** e que podem ser separadas por um ***hiperplano***.
- A ***equação de atualização dos pesos*** é definida como

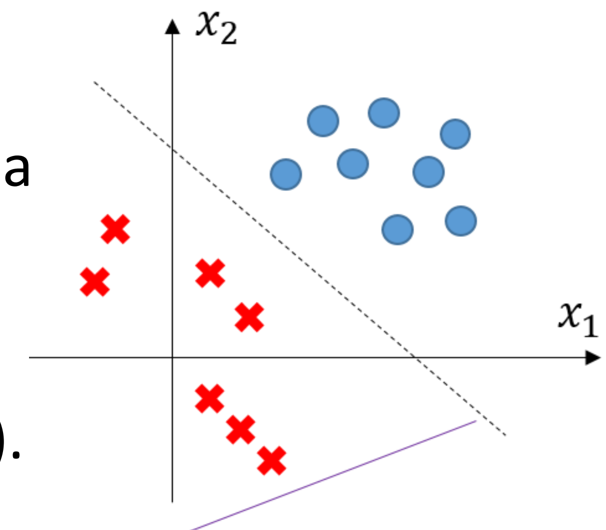
$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \alpha(y - \hat{y})\boldsymbol{x},$$

Equação idêntica a da atualização do gradiente descendente estocástico.

onde $\boldsymbol{w}$ é o vetor de pesos, $\alpha$ é o passo de aprendizagem, $y$ é o valor de saída esperado, $\hat{y}$ é a saída do modelo, i.e., $f(g(\boldsymbol{x}))$, e $\boldsymbol{x}$ é o vetor de atributos.

# Perceptron

- Como podemos perceber, o modelo do ***perceptron*** é idêntico ao ***classificador binário com limiar de decisão rígido***.
- Por definição, o ***perceptron*** sempre utiliza ***superfícies de separação lineares***, ou seja, sempre teremos $g(\boldsymbol{x})$ como sendo a equação de um ***hiperplano***.
- Portanto, teoricamente, ***sem transformação dos atributos***, um ***único perceptron*** só é capaz de ***classificar*** dados que sejam ***linearmente separáveis*** (ou seja, separáveis por um ***hiperplano***).
- A figura ao lado ilustra isso para um caso bidimensional.
- Entretanto, como veremos na sequência, podemos ***combinar os resultados de vários perceptrons*** para criarmos ***superfícies de separação*** que separem dados que não sejam linearmente separáveis sem a necessidade de ***transformar os atributos***, ou seja, de usarmos funções discriminantes, $g(\boldsymbol{x})$, com outros formatos (e.g., polinômios).



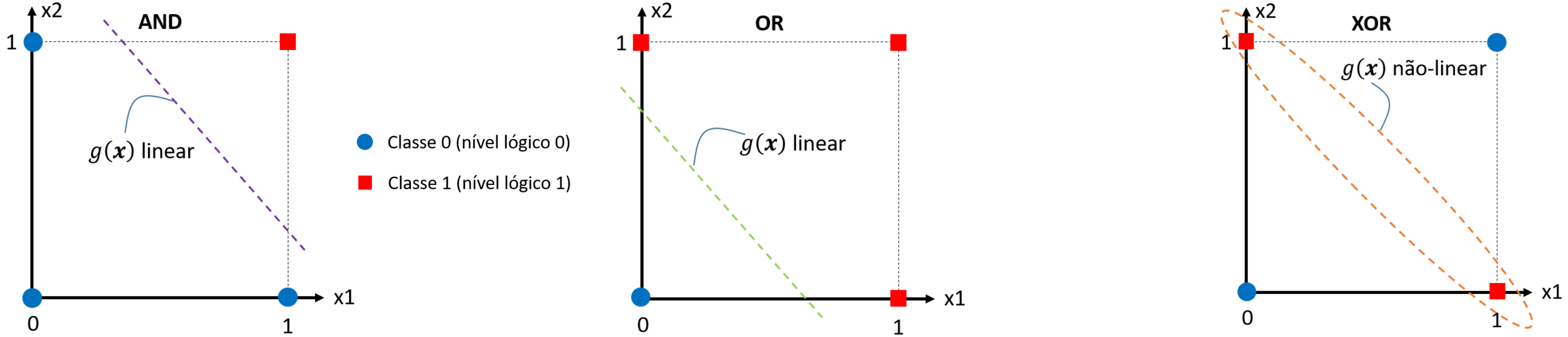

$$g(\boldsymbol{x}) = w_0 + w_1 x_1 + w_2 x_2 = 0$$

A separação das duas classes ocorre onde $g(\boldsymbol{x}) = 0$.

# Perceptron

Exemplo: perceptron_xor_problem.ipynb

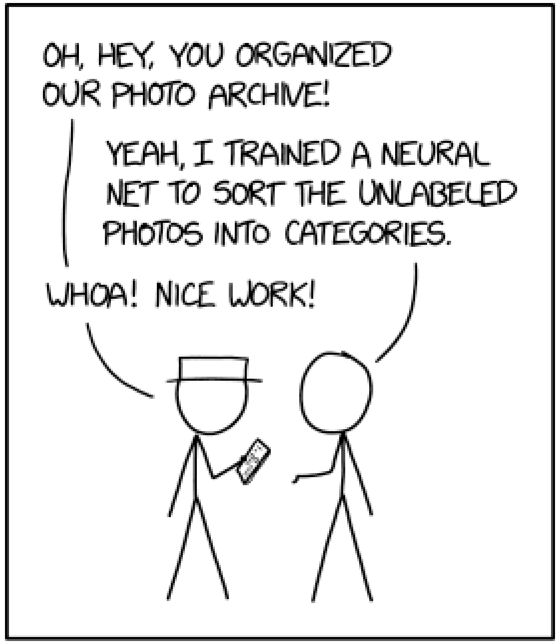
- Por serem ***linearmente separáveis***, as lógicas AND e OR podem ser separadas por um único perceptron (Figuras 1 e 2). Uma simples reta as separa.
- Porém, a lógica XOR ***não é linearmente separável*** e necessita de uma superfície de separação não-linear (Figura 3). ***No mínimo duas retas são necessárias***.
- Como veremos, a ***separação da lógica XOR pode ser obtida combinando-se o resultado de dois perceptrons*** (i.e., dois classificadores lineares), que resultará em uma superfície de separação não linear.



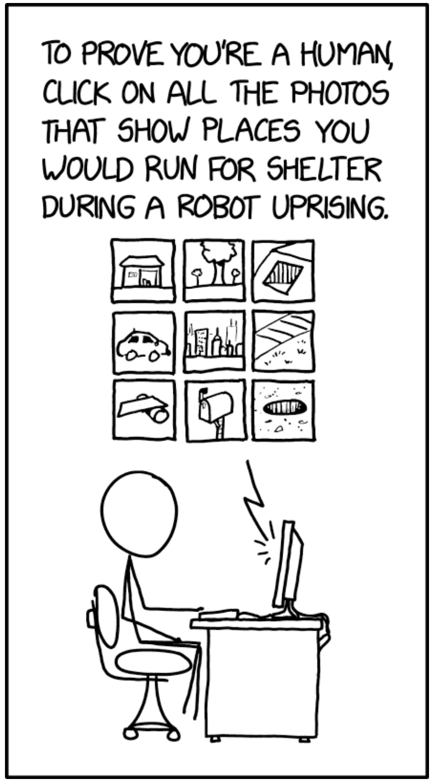
# Tarefas

- **Quiz**: *“T320 - Quiz – Redes Neurais Artificiais (Parte II)”* que se encontra no MS Teams.
- **Exercício Prático**: **Laboratório #6**.
  - Pode ser baixado do MS Teams ou do GitHub.
  - Pode ser respondido através do link acima (na nuvem) ou localmente.
  - Instruções para resolução e entrega dos laboratórios.
  - **Atividades podem ser feitas em grupo, mas as entregas devem ser individuais.**

# Obrigado!

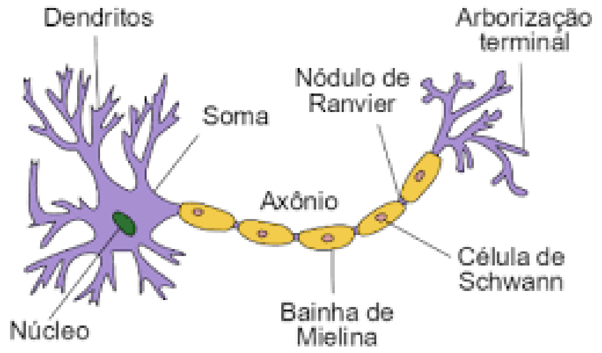



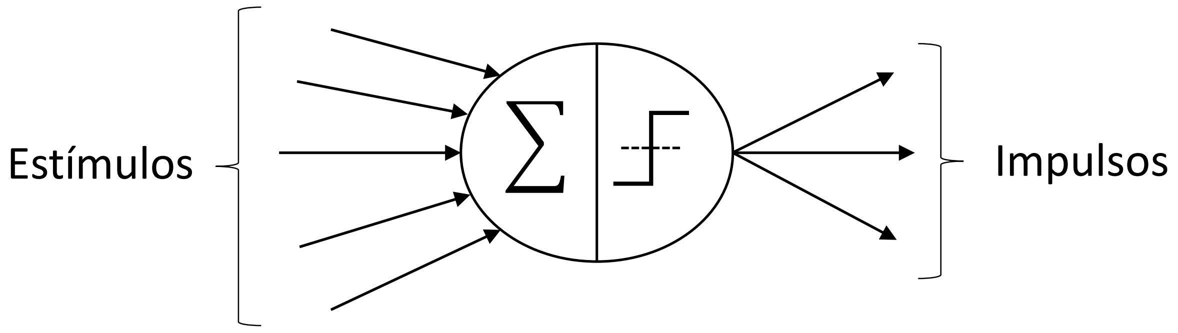

ENGINEERING TIP:
WHEN YOU DO A TASK BY HAND, YOU CAN TECHNICALLY SAY YOU TRAINED A NEURAL NET TO DO IT.

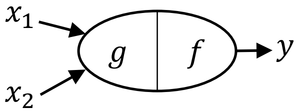


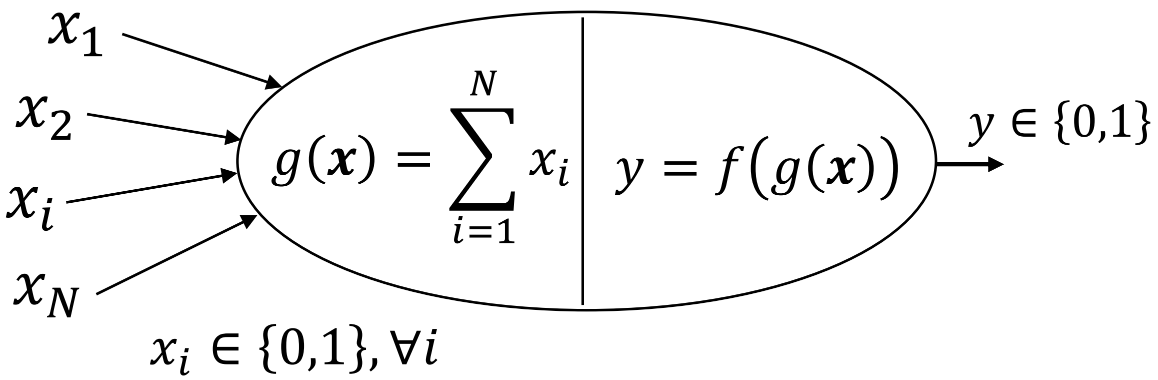


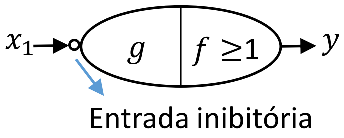
# Figuras

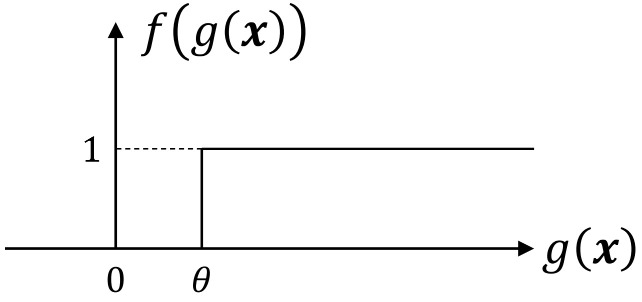

Dendritos
Arborização terminal
Nódulo de Ranvier
Soma
Axônio
Célula de Schwann
Núcleo
Bainha de Mielina


Estímulos
$\sum$
Impulsos



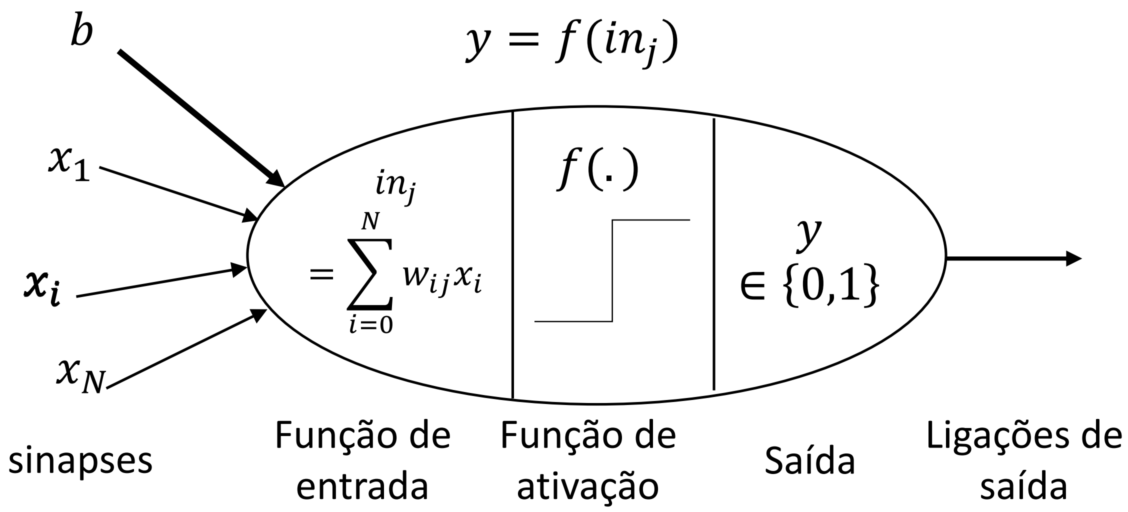



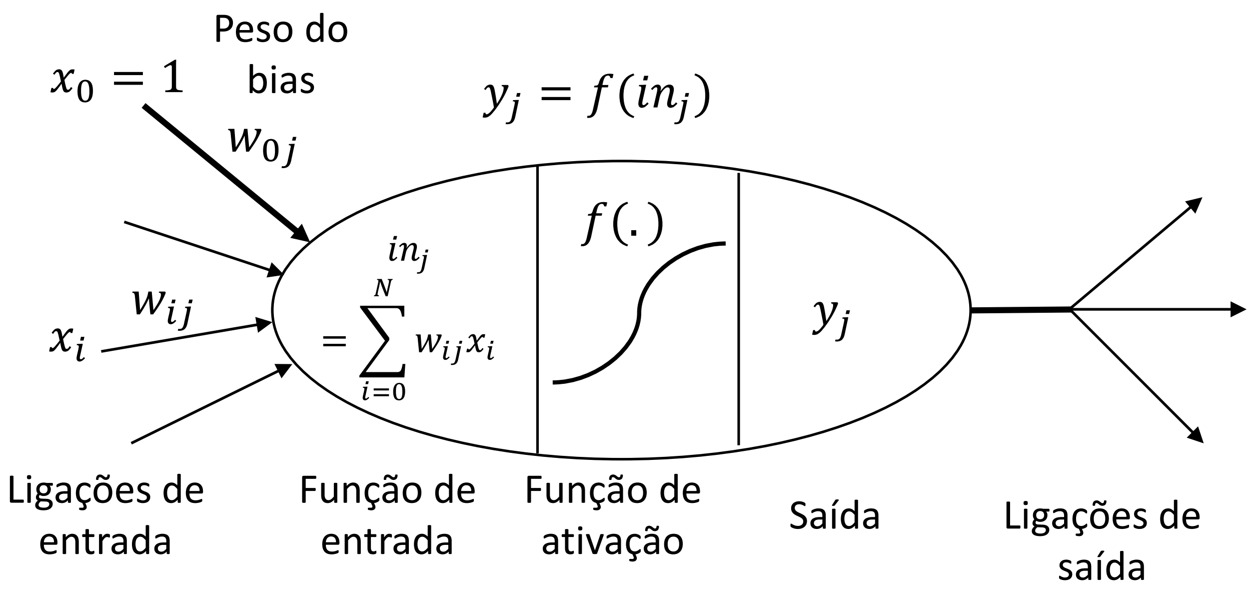




$$y = f\big(g(\boldsymbol{x})\big) = \begin{cases} 1 \text{ se } g(\boldsymbol{x}) \geq \theta \\ 0 \text{ se } g(\boldsymbol{x}) < \theta \end{cases}$$

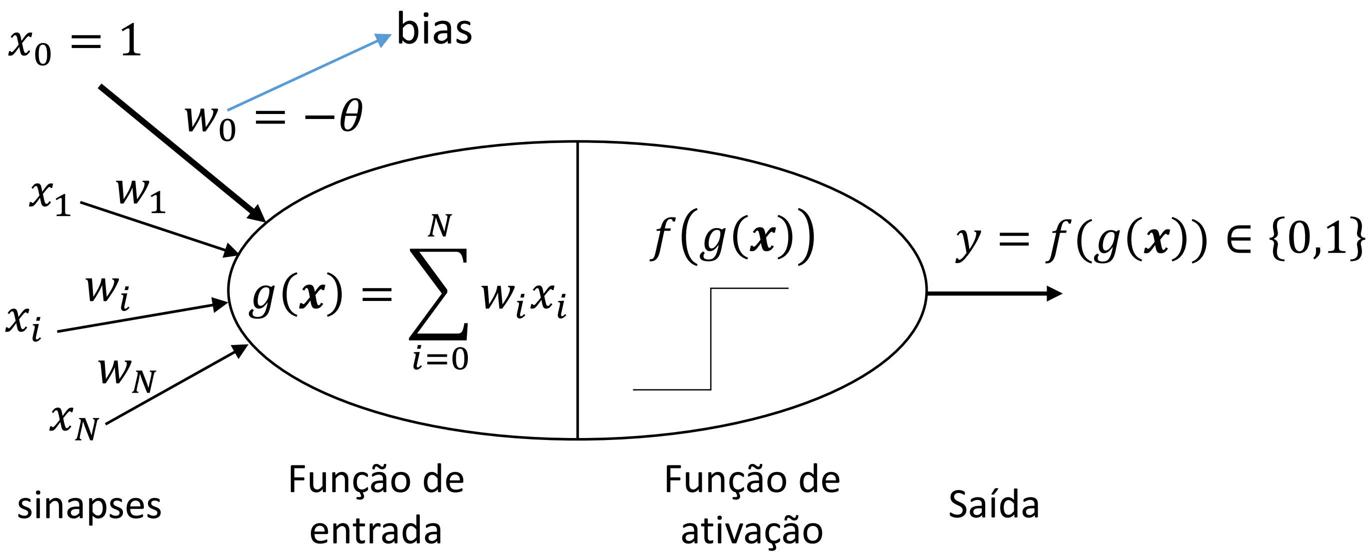
onde $\theta$ é o limiar de decisão.

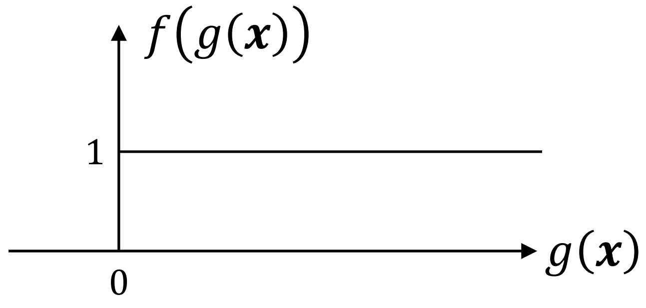


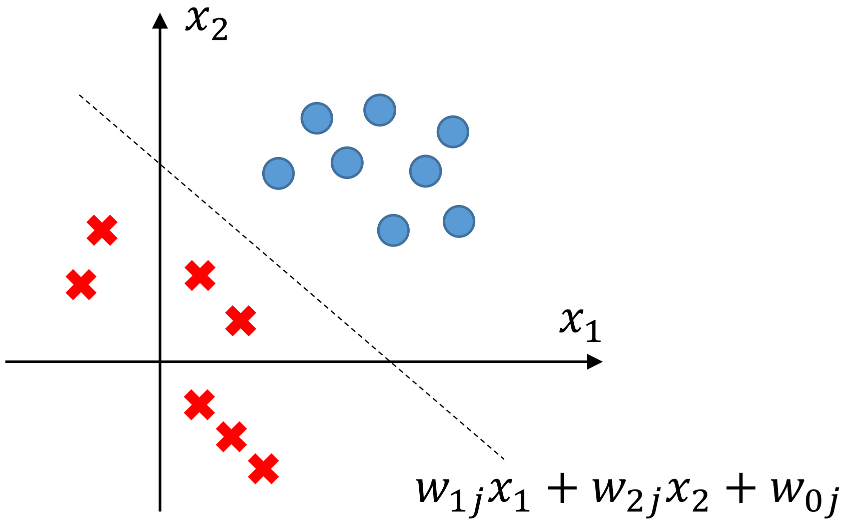
$b$
$y = f(in_j)$
$x_1$
$x_i$
$x_N$
$in_j = \sum_{i=0}^{N} w_{ij} x_i$
$f(.)$
$y \in \{0,1\}$
sinapses
Função de entrada
Função de ativação
Saída
Ligações de saída

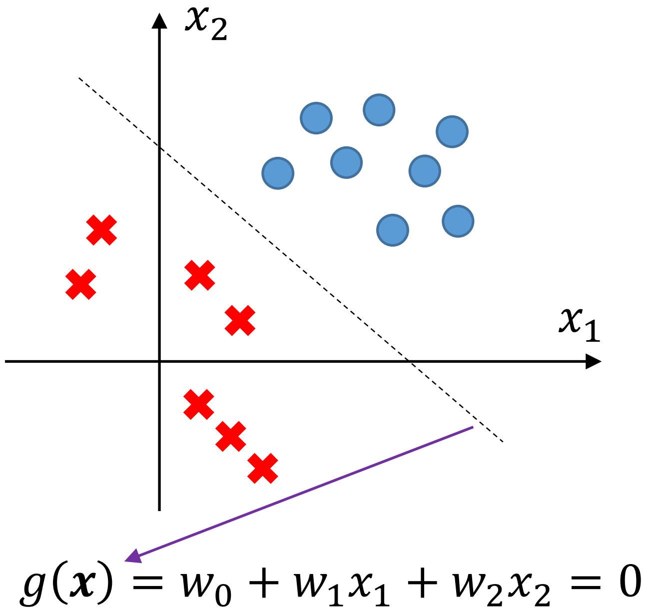


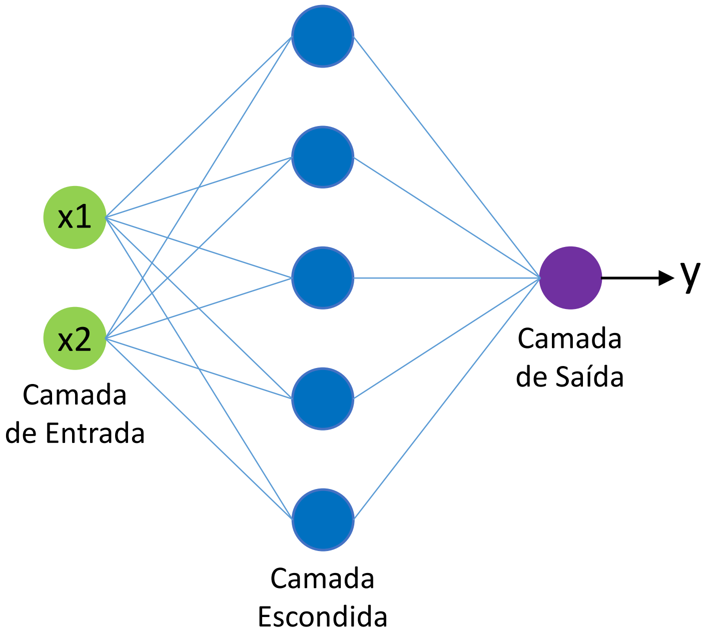
Peso do bias
$x_0 = 1$
$w_{0j}$
$y_j = f(in_j)$
$f(.)$
$w_{ij}$
$x_i$
$in_j = \sum_{i=0}^{N} w_{ij} x_i$
$y_j$
Ligações de entrada
Função de entrada
Função de ativação
Saída
Ligações de saída

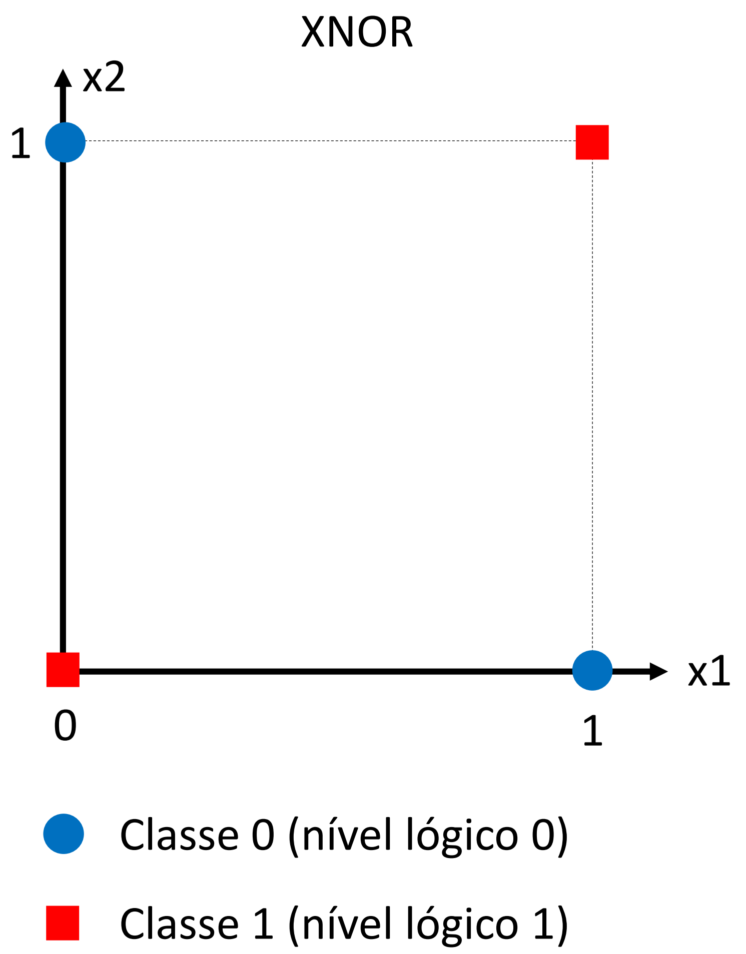

$x_0 = 1$
bias
$w_0 = -\theta$
$x_1$ $w_1$
$x_i$ $w_i$
$x_N$ $w_N$
$g(\boldsymbol{x}) = \sum_{i=0}^{N} w_i x_i$
$f(g(\boldsymbol{x}))$
$y = f(g(\boldsymbol{x})) \in \{0,1\}$
sinapses
Função de entrada
Função de ativação
Saída

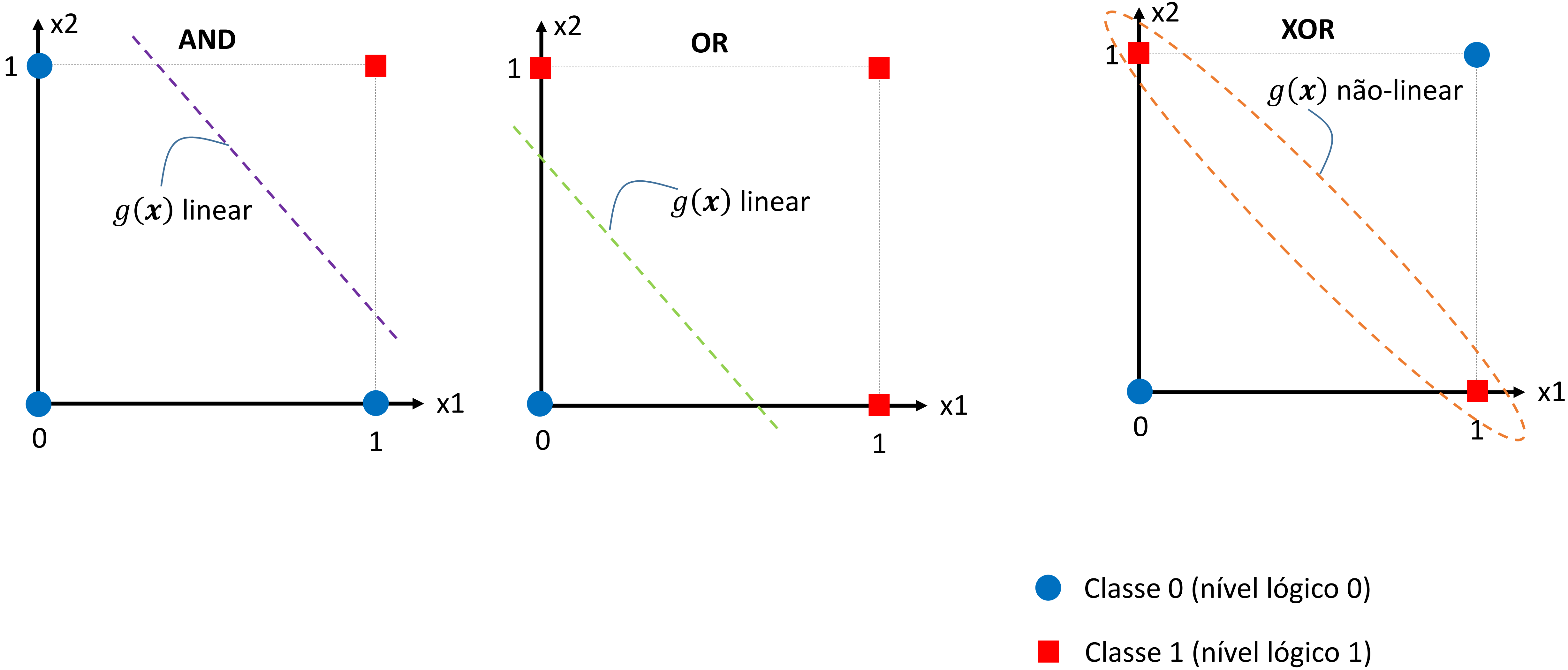

$f(g(\boldsymbol{x}))$
1
0
$g(\boldsymbol{x})$

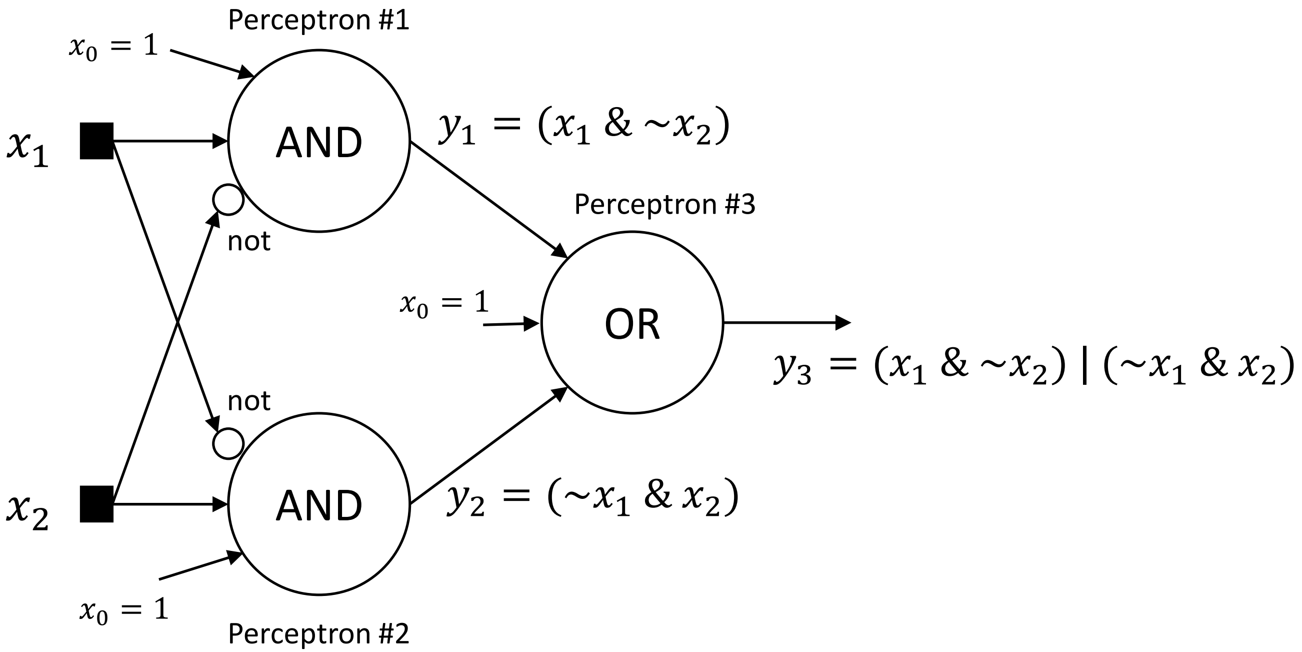

$x_2$
$x_1$
$w_{1j}x_1 + w_{2j}x_2 + w_{0j}$

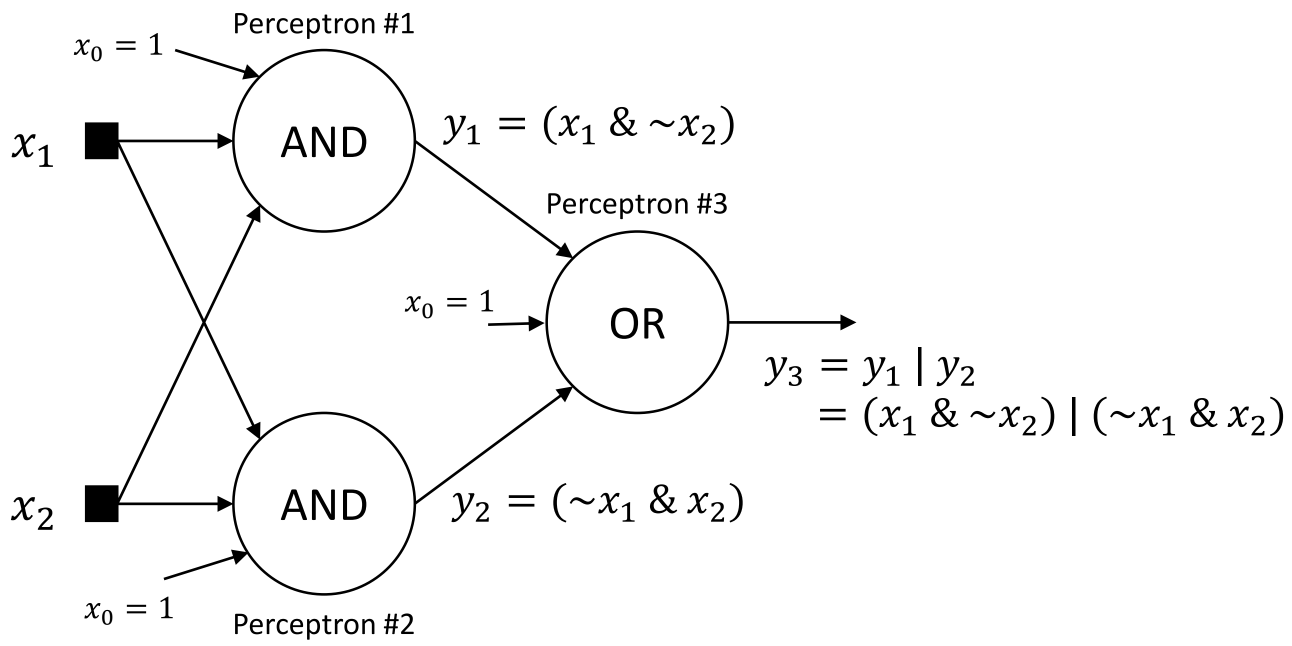

$x_2$
$x_1$
$g(\boldsymbol{x}) = w_0 + w_1x_1 + w_2x_2 = 0$

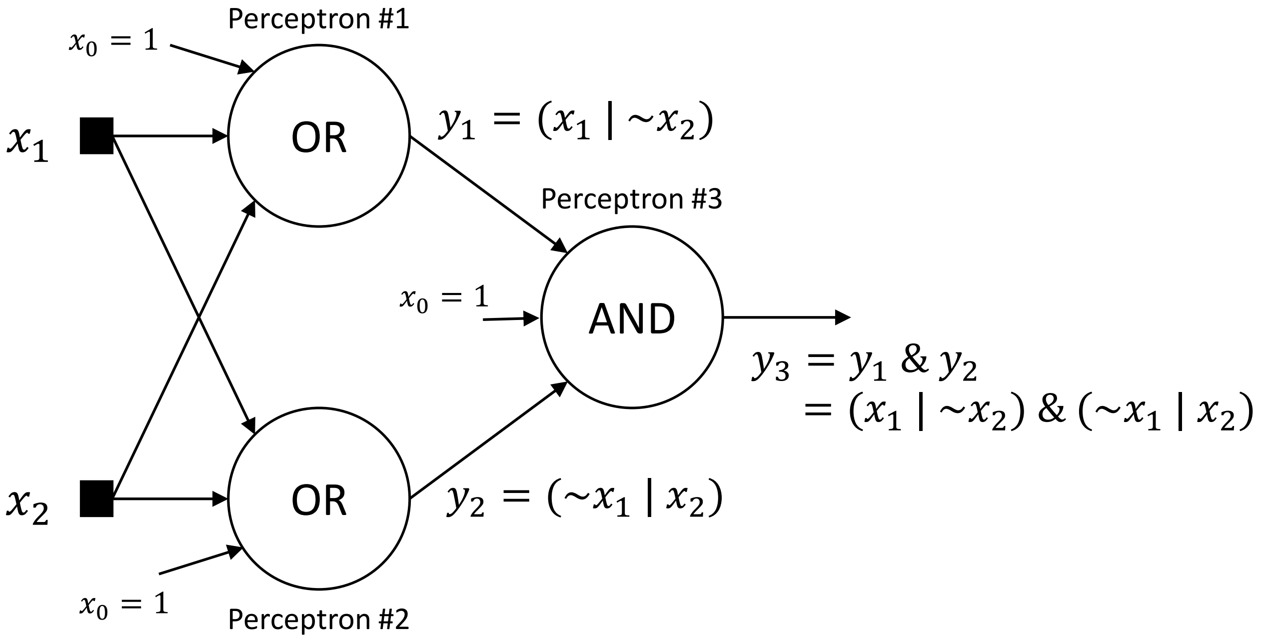

x1
x2
Camada
de Entrada
Camada
Escondida
y
Camada
de Saída


XNOR
x2
1
x1
0
1
Classe 0 (nível lógico 0)
Classe 1 (nível lógico 1)

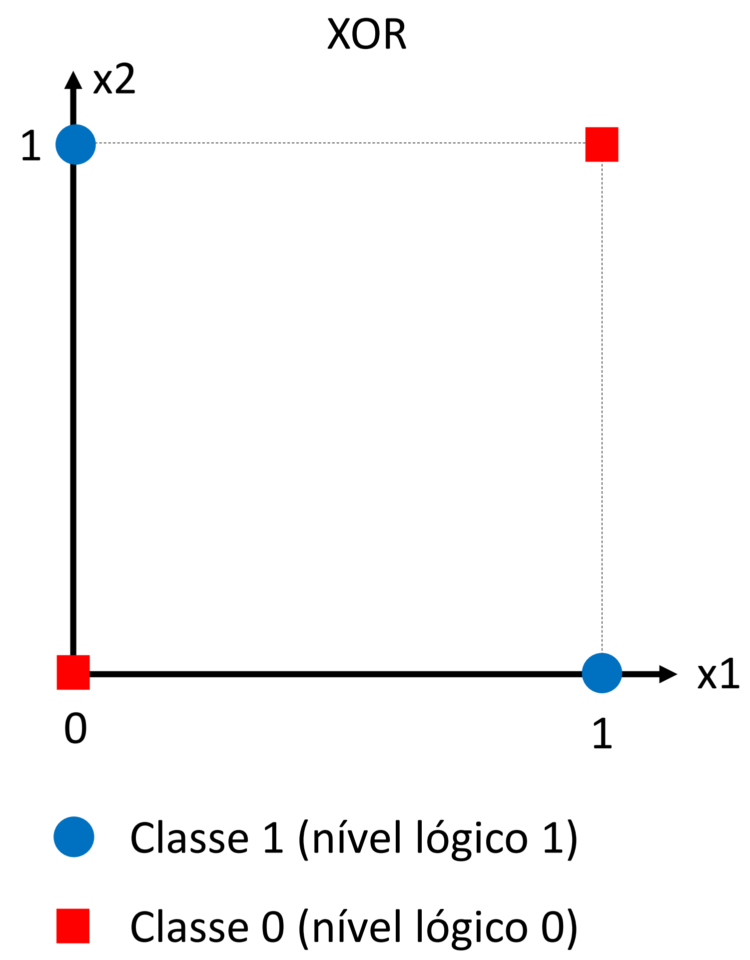

x2
AND
1
$g(\boldsymbol{x})$ linear
0
1
x1
x2
OR
1
$g(\boldsymbol{x})$ linear
0
1
x1
x2
XOR
1
$g(\boldsymbol{x})$ não-linear
0
1
x1
Classe 0 (nível lógico 0)
Classe 1 (nível lógico 1)

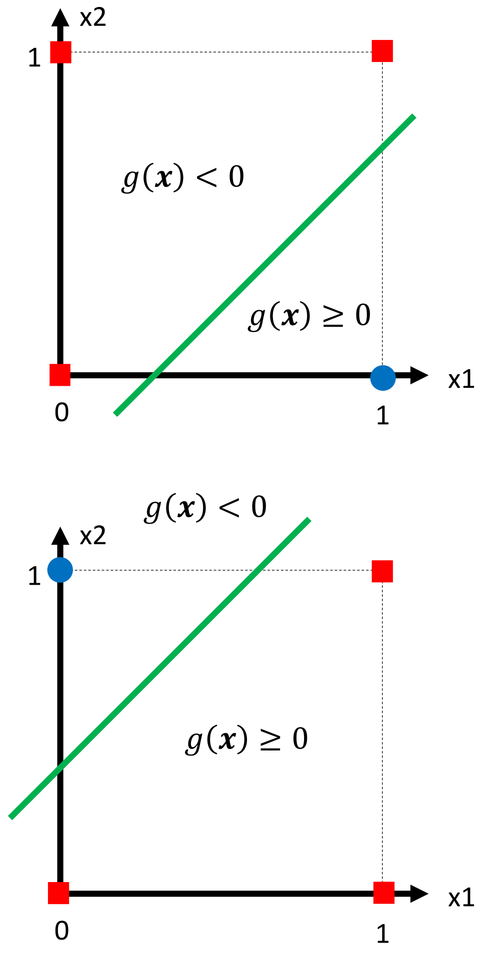

Perceptron #1
$x_0 = 1$
$x_1$
AND
$y_1 = (x_1 \ \& \sim x_2)$
not
Perceptron #3
$x_0 = 1$
OR
$y_3 = (x_1 \ \& \sim x_2) \mid (\sim x_1 \ \& \ x_2)$
not
$x_2$
AND
$y_2 = (\sim x_1 \ \& \ x_2)$
$x_0 = 1$
Perceptron #2


Perceptron #1
$x_0 = 1$
$x_1$
AND
$y_1 = (x_1 \;\&\; \sim x_2)$
Perceptron #3
$x_0 = 1$
OR
$y_3 = y_1 \mid y_2$
$= (x_1 \;\&\; \sim x_2) \mid (\sim x_1 \;\&\; x_2)$
$x_2$
AND
$y_2 = (\sim x_1 \;\&\; x_2)$
$x_0 = 1$
Perceptron #2



Perceptron #1
$x_0 = 1$
$x_1$
OR
$y_1 = (x_1 \mid \sim x_2)$
Perceptron #3
$x_0 = 1$
AND
$y_3 = y_1 \,\&\, y_2$
$= (x_1 \mid \sim x_2) \,\&\, (\sim x_1 \mid x_2)$
$x_2$
OR
$y_2 = (\sim x_1 \mid x_2)$
$x_0 = 1$
Perceptron #2


XOR
x2
1
x1
0
1
Classe 1 (nível lógico 1)
Classe 0 (nível lógico 0)


x2
1
$g(\boldsymbol{x}) < 0$
$g(\boldsymbol{x}) \geq 0$
x1
0
1
$g(\boldsymbol{x}) < 0$
x2
1
$g(\boldsymbol{x}) \geq 0$
x1
0
1