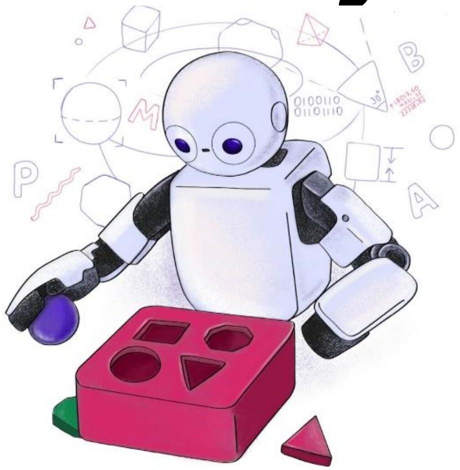
T320 - Introdução ao Aprendizado de Máquina II:

Redes Neurais Artificiais (Parte I)



Inatel

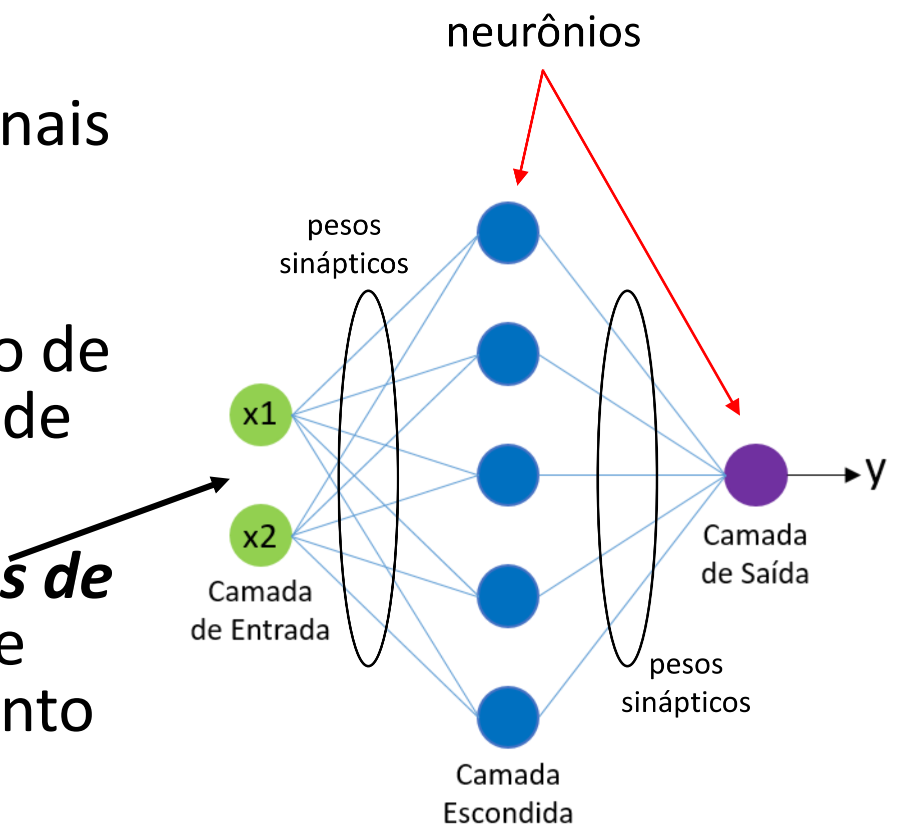
Felipe Augusto Pereira de Figueiredo
felipe.figueiredo@inatel.br

Introdução

- Vamos falar sobre 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 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.
- 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

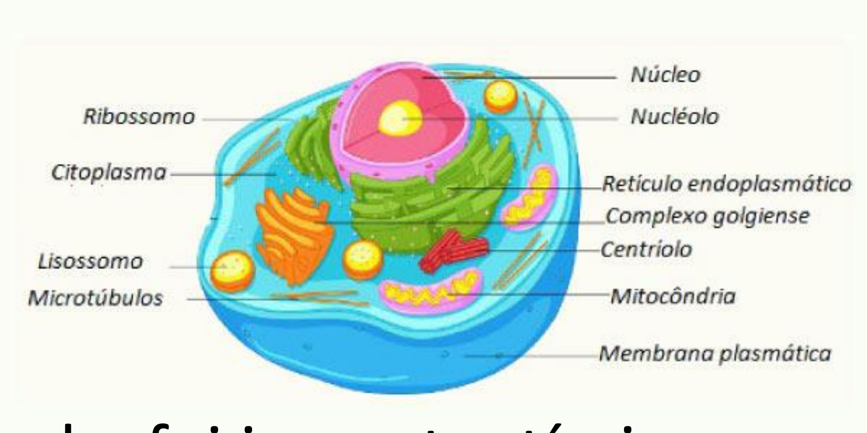
- RNAs são versáteis, poderosas e escalonáveis, tornando-as ideais para realizar tarefas grandes e altamente complexas de ***aprendizado de máquina***, como por exemplo:
  - classificar bilhões de imagens (por exemplo, como o Google Images, Facebook, etc. fazem),
  - serviços de reconhecimento de fala (por exemplo, o Siri da Apple, Alexa da Amazon e Google Assistant da Google),
  - recomendar vídeos que melhor se adequam ao comportamento de centenas de milhões de usuários todos os dias (por exemplo, YouTube, Netflix),
  - ou aprender a vencer o campeão mundial de Go examinando milhões de partidas anteriores e depois jogando contra si mesmo (AlphaGo da DeepMind).

Alexa

Google Assistant

Siri

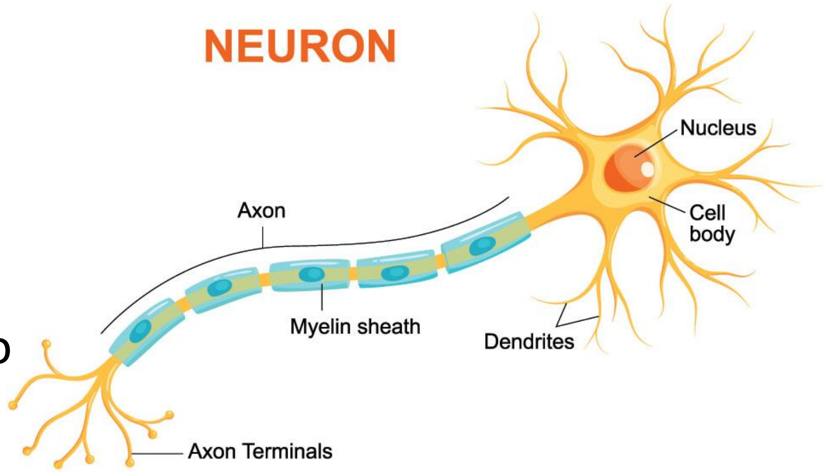
Um pouco de contexto



- 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***.
- As células ***eucariontes*** (plantas, animais, fungos, protozoários e algas) 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***.
- Já o ***núcleo*** abriga a maior parte do material genético (DNA) da célula.
  - Ele regula o metabolismo e armazena as informações genéticas da célula.

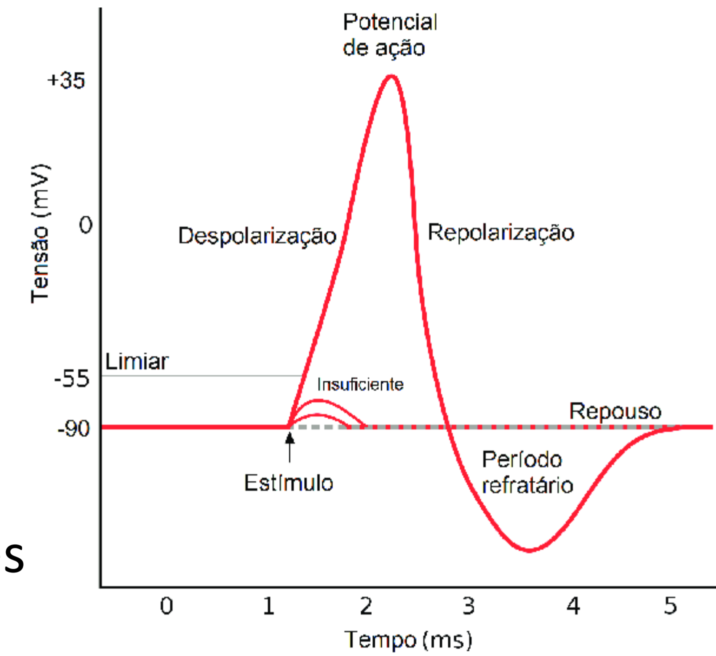
Um pouco de contexto

- Os ***neurônios*** são células ***eucariontes*** também, mas são células que possuem mecanismos elétricos e/ou químicos característicos.
- Os neurônios apresentam três partes básicas: os ***dendritos***, o ***axônio*** e o ***corpo celular***.
- 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.
- 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 locais/pontos de contato entre os dentritos de um neurônio e os terminais do axônio de outro neurônio são chamados de ***sinapses*** e os contatos entre eles de ***contatos sinápticos***.
- Ou seja, os neurônios se comunicam uns com os outros através das ***sinapses***.
- A figura ao lado mostra o diagrama de um ***neurônio***.



Um pouco de contexto

- Em termos simples, mas lembrando de que existem exceções, nós podemos afirmar que:
    - O neurônio recebe estímulos elétricos, basicamente a partir dos dendritos.
    - Esses estímulos são integrados no corpo celular (*soma*).
    - A integração dos estímulos pode levar à geração ou não de uma resposta elétrica enviada pelo axônio a outros neurônios.
- Nós podemos simplificar o funcionamento do ***neurônio*** como:
    - Os neurônios recebem estímulos elétricos.
    - Esses estímulos são integrados.
    - Se a atividade (i.e., integração dos estímulos) exceder certo limiar, o ***neurônio*** gera um pulso (ou potencial de ação).
- O potencial de ação é mostrado na figura ao lado.
- Um ***neurônio*** se conecta com 10 a 100.000 outros ***neurônios*** através das ***sinapses***.
- Sinais são passados de ***neurônio*** para ***neurônio*** através de reações eletro-químicas.
- Do ponto de vista do nosso curso, o ***neurônio*** será considerado como um sistema com várias entradas e uma saída onde a comunicação entre neurônios é feita através de sinais elétricos.



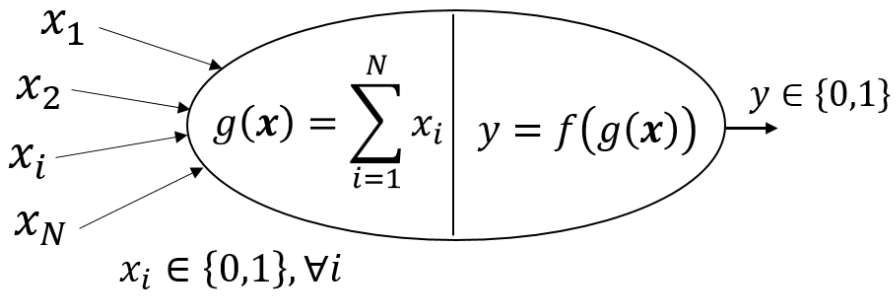
O Modelo de McCulloch e Pitts

- 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, Warren McCulloch e Walter Pitts apresentaram 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*** onde uma proposição é uma sentença declarativa, ou seja, é uma sentença que declara um fato podendo este ser verdeiro ou falso.
  - 1 ou 1 = 1
  - 1 e 0 = 0
- O artigo de McCulloch e Pitts fornece *insights* fundamentais sobre como a ***lógica proposicional*** pode ser processada por um neurônio.
- A partir daí, a relação com a computação foi natural.

Walter Pitts e Warren McCulloch

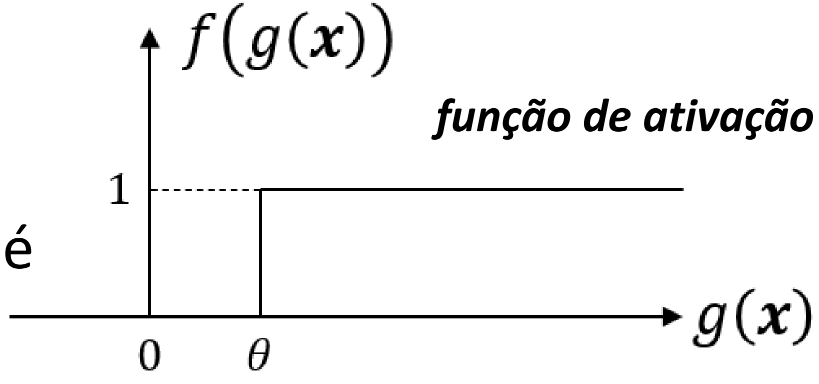
O Modelo de McCulloch e Pitts

- A figura ao lado mostra o modelo matemático do ***neurônio*** criado por McCulloch e Pitts.
- Grosso modo, o ***neurônio*** é ativado (ou disparado) quando uma ***combinação linear*** de suas entradas excede um ***limiar de ativação***.
- As premissas do modelo de McCulloch e Pitts (M-P) são:
  - Os valores das entradas, $x_i, \forall i$, ou também chamadas de ***sinapses***, são sempre valores booleanos, i.e., '0', ou '1'.
  - As entradas são simplesmente somadas.
  - A atividade do ***neurônio*** é um processo do tipo "***tudo ou nada***", ou seja, um processo binário.
  - Portanto, a ***função de ativação*** do neurônio é uma ***função degrau*** com ***ponto de disparo*** dependente do ***limiar de ativação***, θ.
  - Um certo número de ***sinapses*** deve ser excitado num determinado período 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***.



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

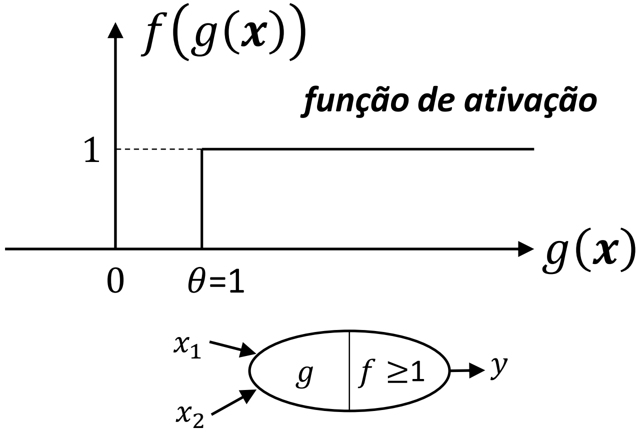
onde θ é o ***limiar de ativação***.



Exemplos com o modelo de McCulloch e Pitts

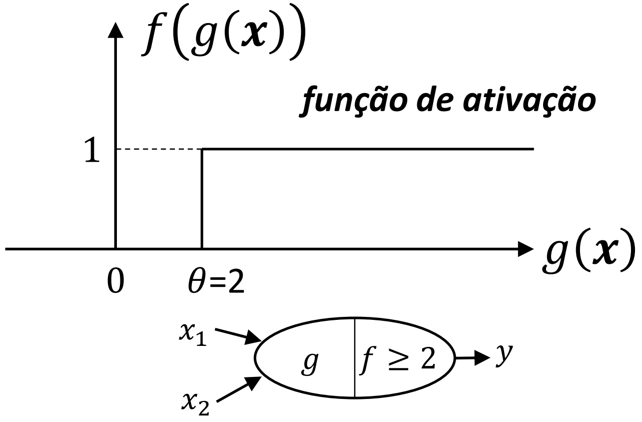
OR			
x1	x2	y	$g(\boldsymbol{x})$
0	0	0	0
0	1	1	1
1	0	1	1
1	1	1	2

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



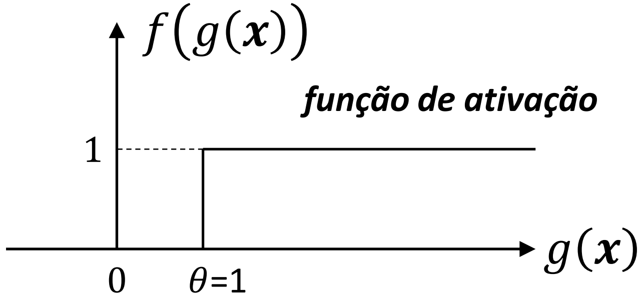
AND			
x1	x2	y	$g(\boldsymbol{x})$
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	2

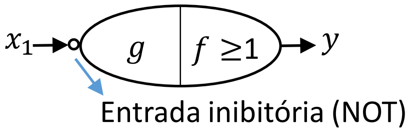
- Qual seria o valor do ***limiar de ativação***, θ?
- Analisando-se $g(\boldsymbol{x})$, vemos que o disparo deve ocorrer quando $g(\boldsymbol{x}) \geq 2$, portanto, $\theta = 2$.



NOT		
x1	y	$g(\boldsymbol{x})$
0	1	0
1	0	1

- Qual seria o valor do ***limiar de ativação***, θ?
- Analisando-se $g(\boldsymbol{x})$, vemos que para o disparo occorer, o valor de x1 deve ser negado, e assim, ele ocorre quando $g(\boldsymbol{x}) \geq 1$, portanto, $\theta = 1$.





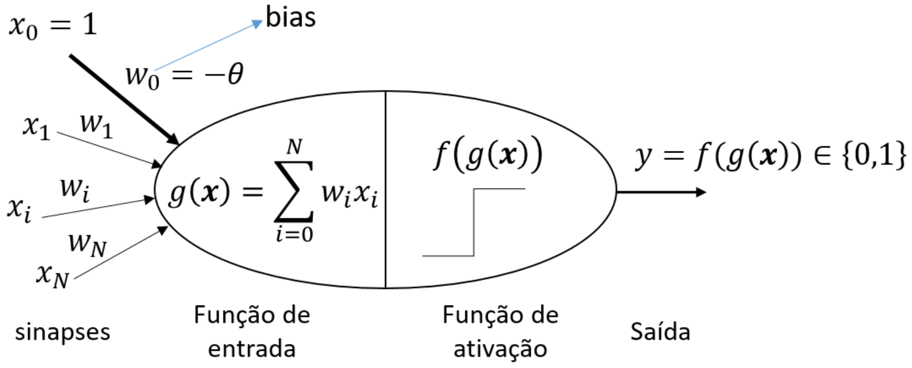
OBS.: Entradas inibitórias são entradas que tem seus valores ***'negados'***.

Tarefa

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

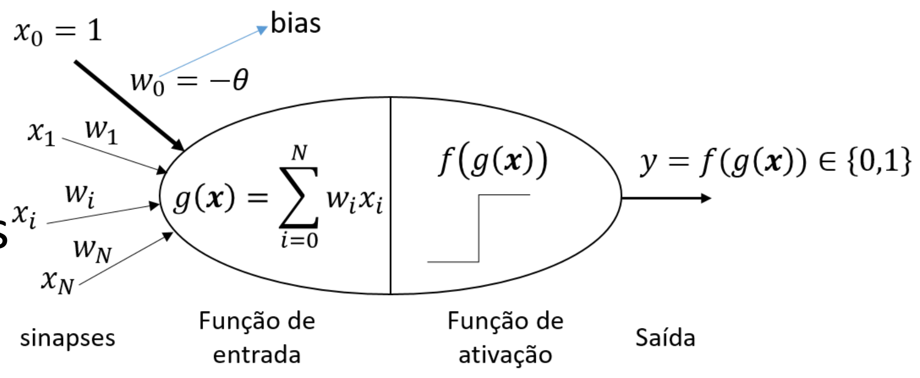
Perceptron

- Em 1958, Frank Rosenblatt, propôs o modelo clássico do ***perceptron***.
- Em 1969, o modelo de Rosenblatt foi cuidadosamente analisado e refinado por Minsky e Papert.
- O modelo criado por eles é chamado de ***perceptron*** e é mostrado na figura ao lado.
- Como veremos a seguir, o modelo do **perceptron**, é um modelo computacional mais geral que o modelo do ***neurônio*** de McCulloch e Pitts.



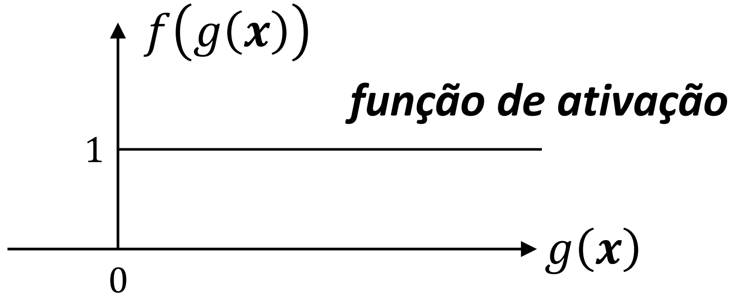
Perceptron

- 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*** θ.

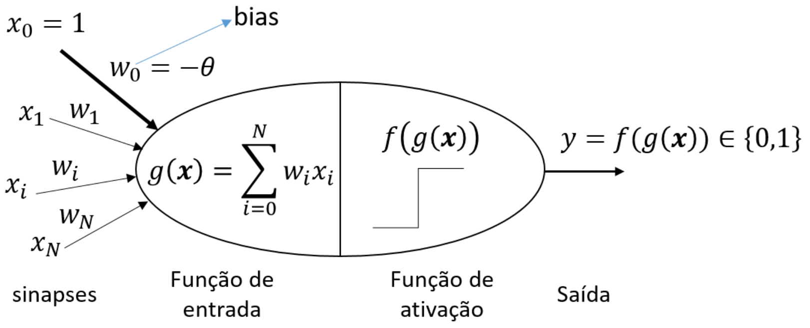


$$y = f\big(g(\boldsymbol{x})\big) = \begin{cases} 1, \text{se } g(\boldsymbol{x}) \geq 0 \\ 0, \text{se } g(\boldsymbol{x}) < 0 \end{cases}$$

Perceba que o ***limiar de ativação*** θ agora faz parte das entradas e é chamado de ***bias***.



Perceptron



- A ideia é que a ativação do ***perceptron*** (causada pelos estímulos de entrada) seja uma ***combinação linear*** dos ***estímulos*** em relação aos ***pesos sinápticos***.
- Se a ativação exceder o ***limiar de ativação***, ocorrerá o ***disparo***.
- Isso é expresso por meio de uma ***função de ativação*** do tipo ***degrau***.
- Notem que a ***função de ativação*** $f(.)$ está centrada "em torno de zero" e o ***limiar de ativação*** é controlado, indiretamente, pelo valor do ***peso do bias***, w_0.
  - O ***limiar de ativação*** foi absorvido pelo somatório, $g(\boldsymbol{x})$, e, portanto, podemos usar a ***função de ativação*** centrada em zero, pois agora, ajusta-se o limiar de ativação indiretamente, através da atualização do peso w_0.
- O tipo de resposta do ***perceptron*** dá origem a um ***classificador binário***, ou seja, para ***problemas com duas classes***.
- As classes são separadas por uma ***superfície de separação linear*** (hiperplano) para o qual a igualdade abaixo é verdadeira.

função discriminante $g(\boldsymbol{x}) = \sum_{i=0}^{N} w_i x_i = 0$ (onde $x_0 = 1$).

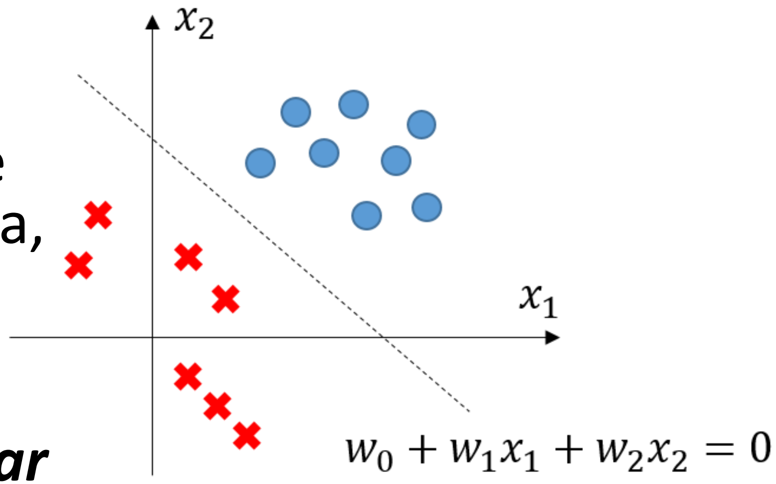
Para que $y = 1$

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

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

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, 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 vários perceptrons*** para criamos uma ***superfície de separação*** que separe dados que não sejam linearmente separáveis sem a necessidade de usarmos funções discriminantes, $g(\boldsymbol{x})$, com ouros formatos (e.g., polinômios).



Regra de aprendizado do perceptron

Exemplo: perceptron_xor_problem.ipynb

- Devido ao fato de que a ***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, não podemos utilizar o ***gradiente descendente***.
- Entretanto, como aprendemos anteriormente, usamos a ***regra de aprendizado do perceptron*** para treinar o modelo.
- Regra simples e intuitiva de atualização dos pesos do modelo
- No caso do perceptron, onde $g(\boldsymbol{x})$ é um hiperplano, 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

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

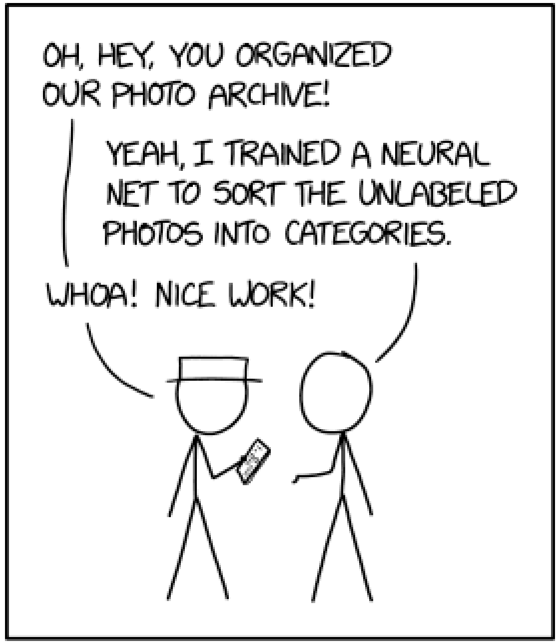
Idêntica à atualização do gradiente descendente estocástico.

onde $\boldsymbol{w}$ é o vetor de pesos, α 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.

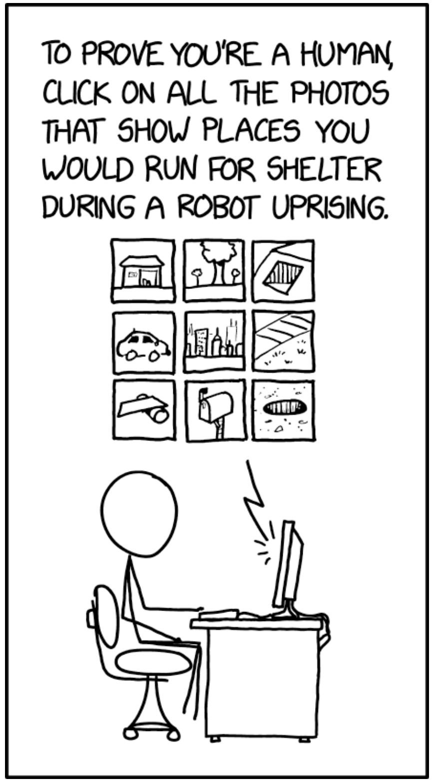
Tarefas

- **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.**

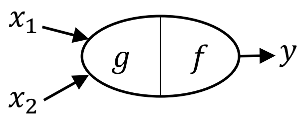
Obrigado!

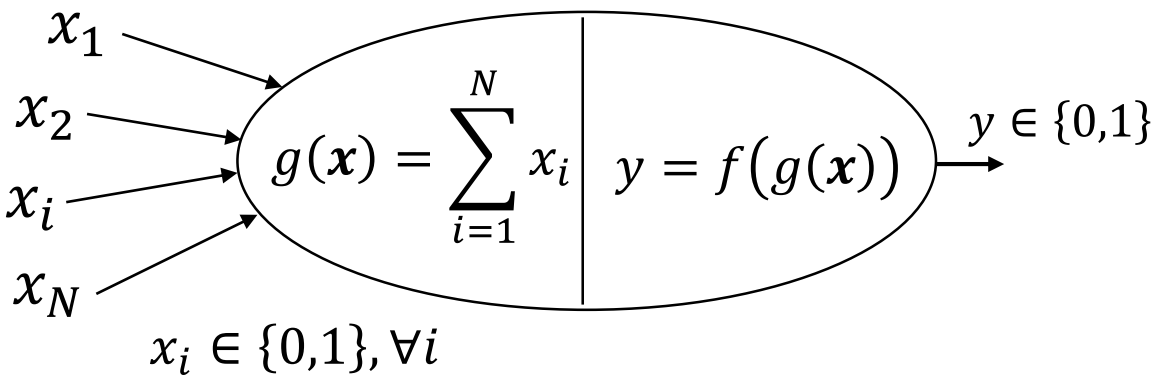


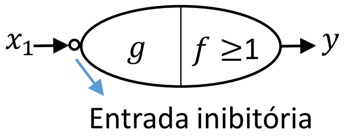
ENGINEERING TIP:
WHEN YOU DO A TASK BY HAND, YOU CAN TECHNICALLY SAY YOU TRAINED A NEURAL NET TO DO IT.

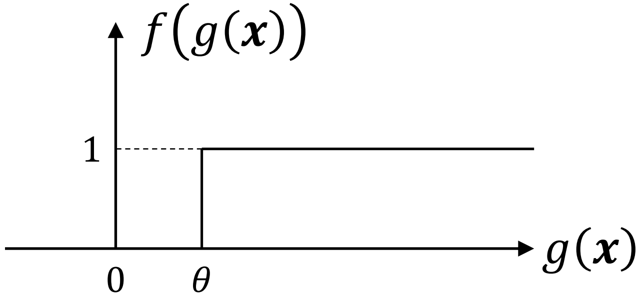


Figuras



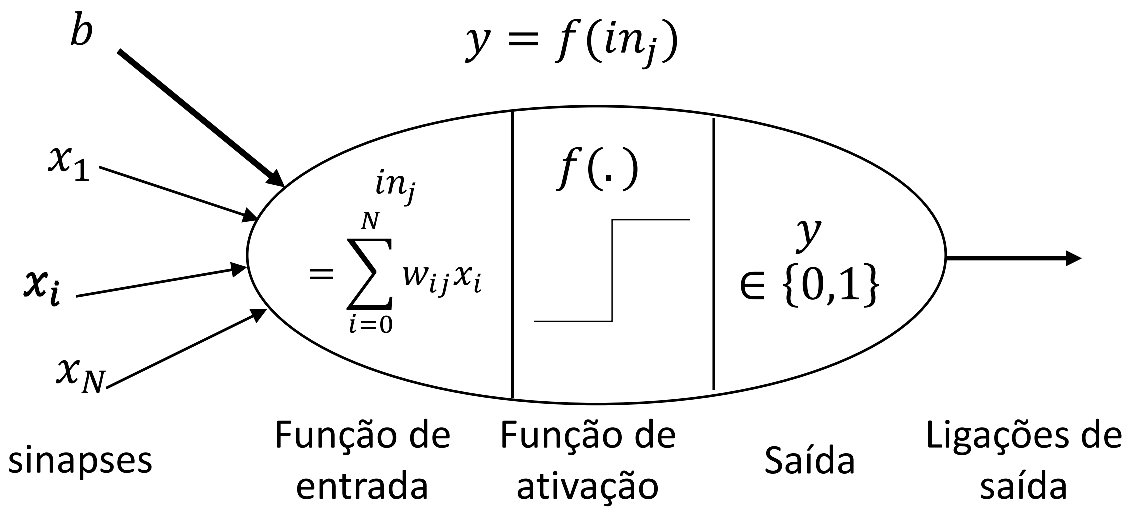


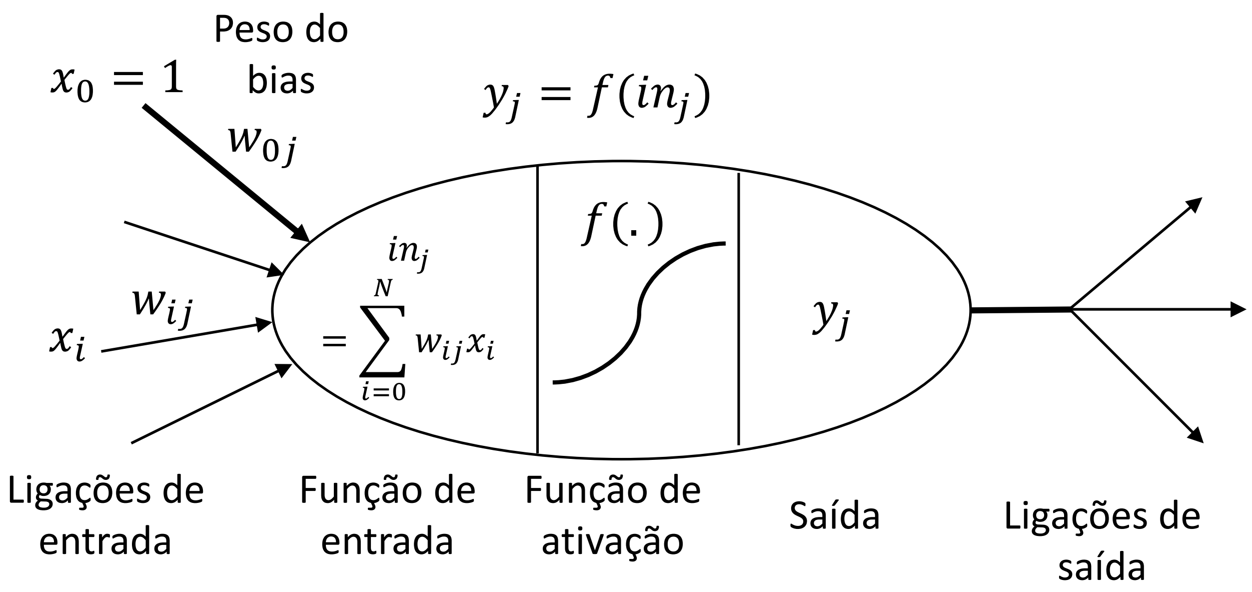


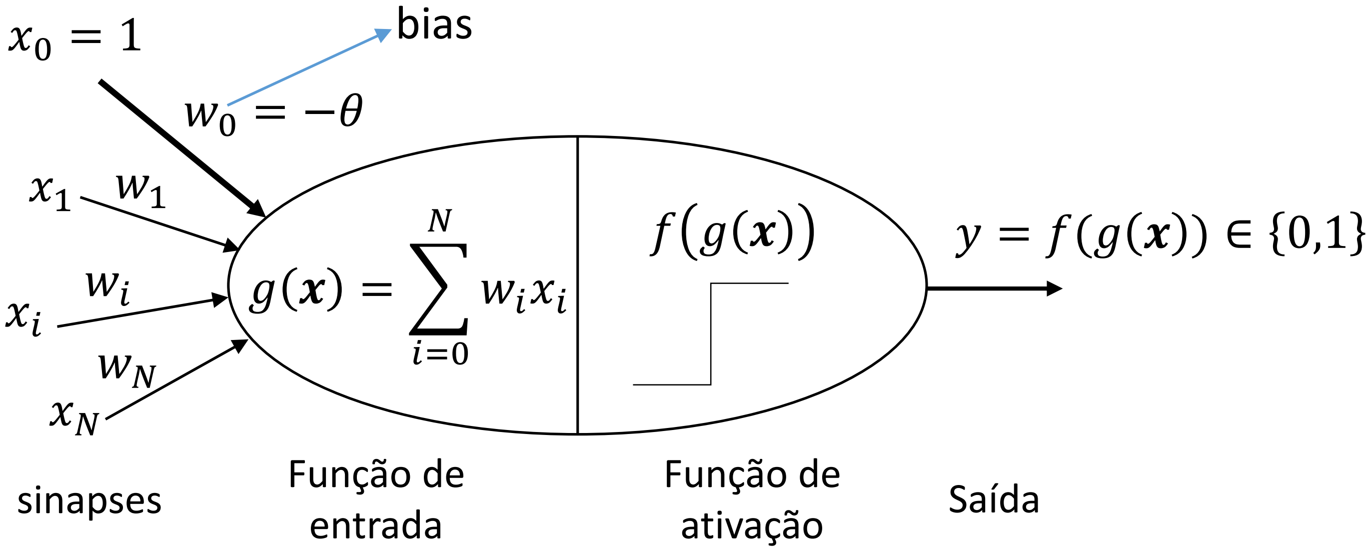


$$y = f\big(g(\boldsymbol{x})\big) = \begin{cases} 1 \text{ se } g(\boldsymbol{x}) \geq \theta \\ 0 \text{ se } g(\boldsymbol{x}) < \theta \end{cases}$$

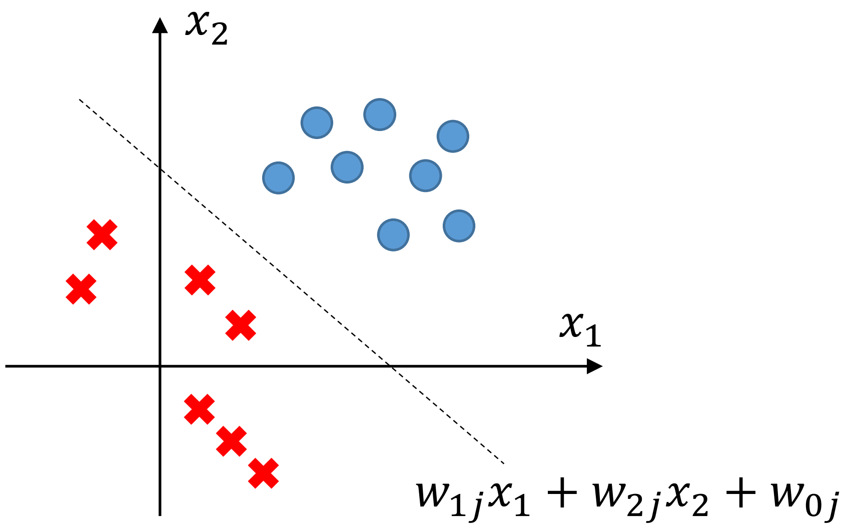
onde θ é o limiar de decisão.

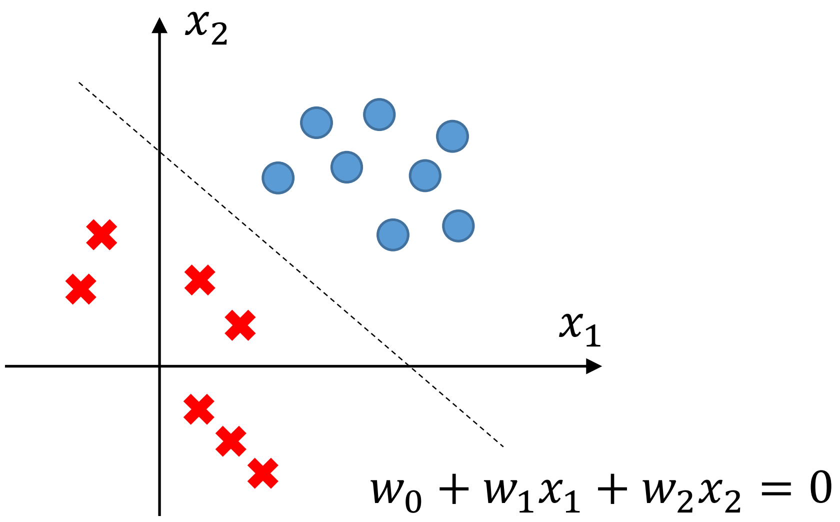

b
$y = f(in_j)$
x_1
x_i
x_N
$in_j = \sum_{i=0}^{N} w_{ij} x_i$
$f(.)$
$y \in \{0,1\}$
sinapses
Função de entrada
Função de ativação
Saída
Ligações de saída

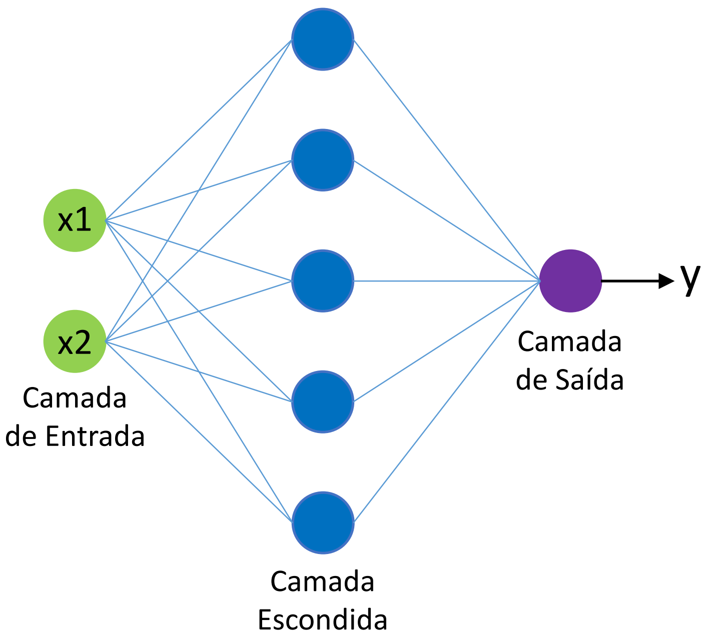

$x_0 = 1$
Peso do bias
w_{0j}
$y_j = f(in_j)$
$f(.)$
$in_j = \sum_{i=0}^{N} w_{ij} x_i$
w_{ij}
x_i
y_j
Ligações de entrada
Função de entrada
Função de ativação
Saída
Ligações de saída

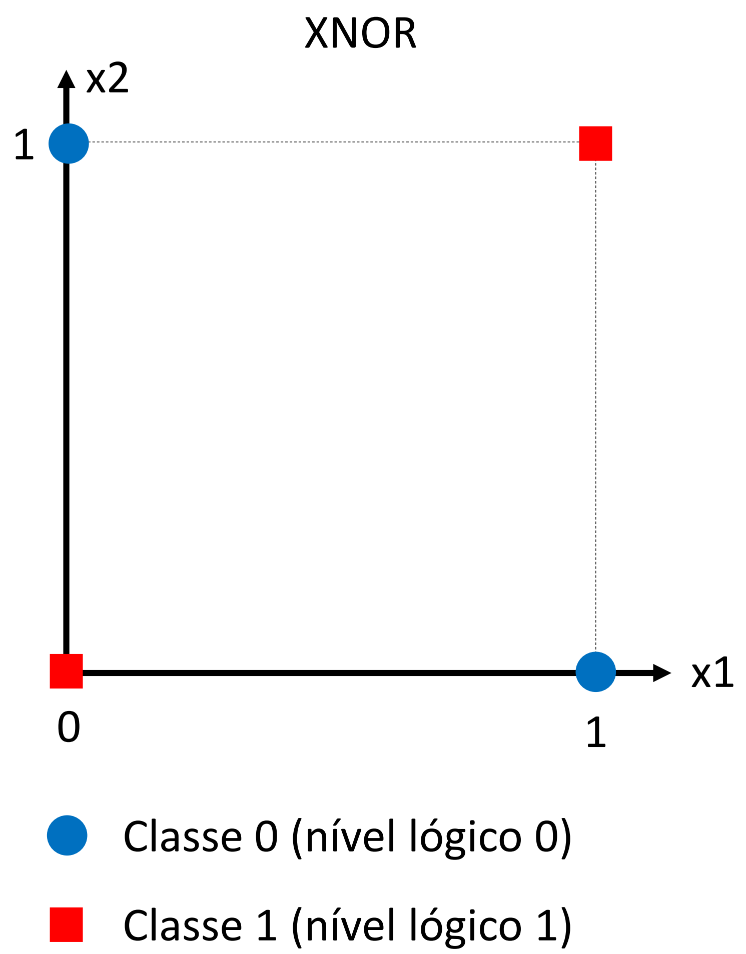

$x_0 = 1$
bias
$w_0 = -\theta$
x_1 w_1
w_i
x_i
w_N
x_N
$g(\boldsymbol{x}) = \sum_{i=0}^{N} w_i x_i$
$f(g(\boldsymbol{x}))$
$y = f(g(\boldsymbol{x})) \in \{0,1\}$
sinapses
Função de entrada
Função de ativação
Saída

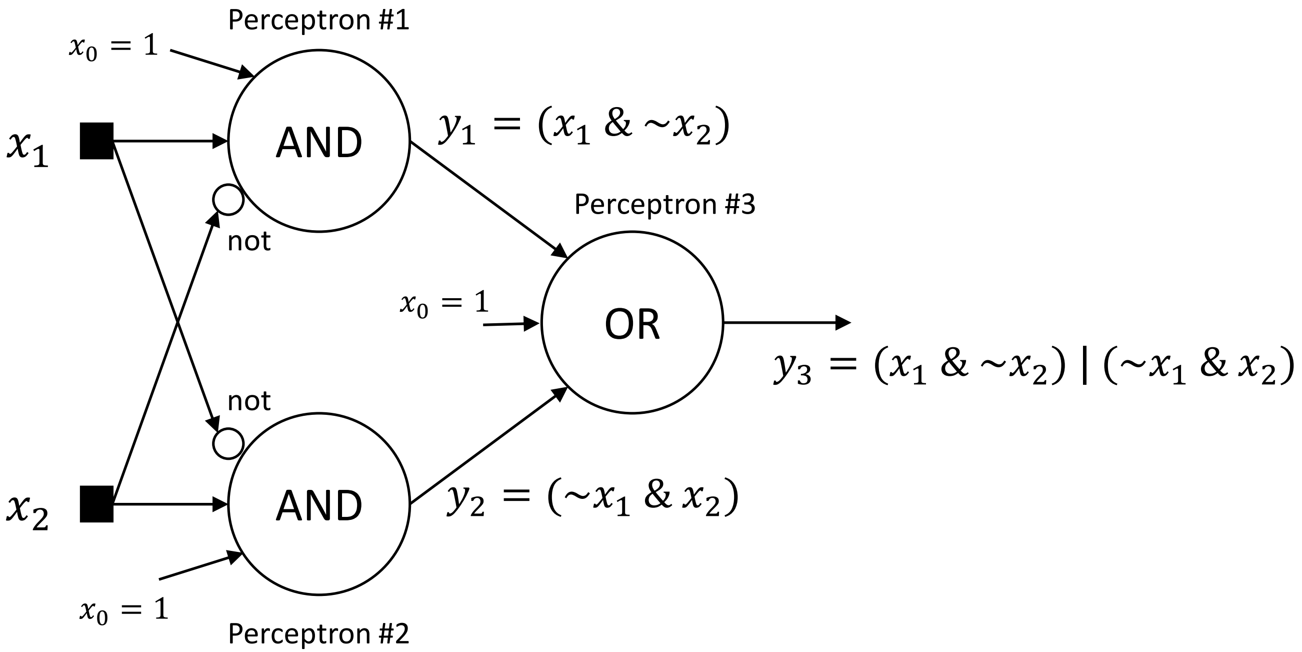

$f(g(\boldsymbol{x}))$
1
0
$g(\boldsymbol{x})$

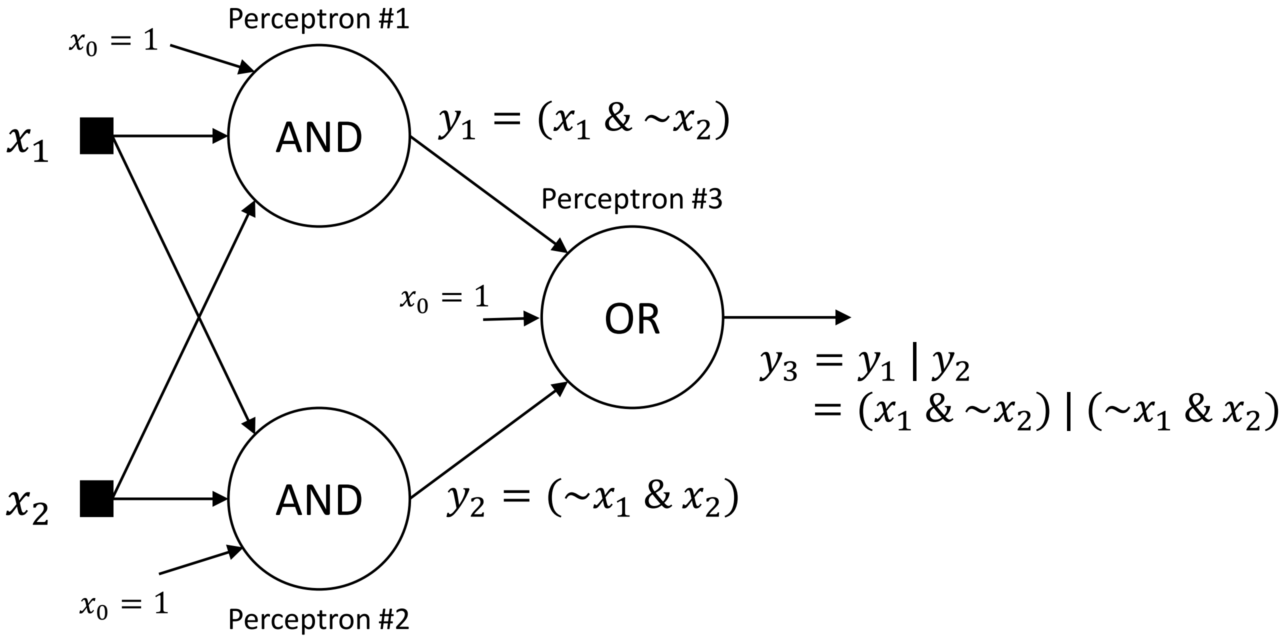

x_2
x_1
$w_{1j}x_1 + w_{2j}x_2 + w_{0j}$


x_2
x_1
$w_0 + w_1x_1 + w_2x_2 = 0$

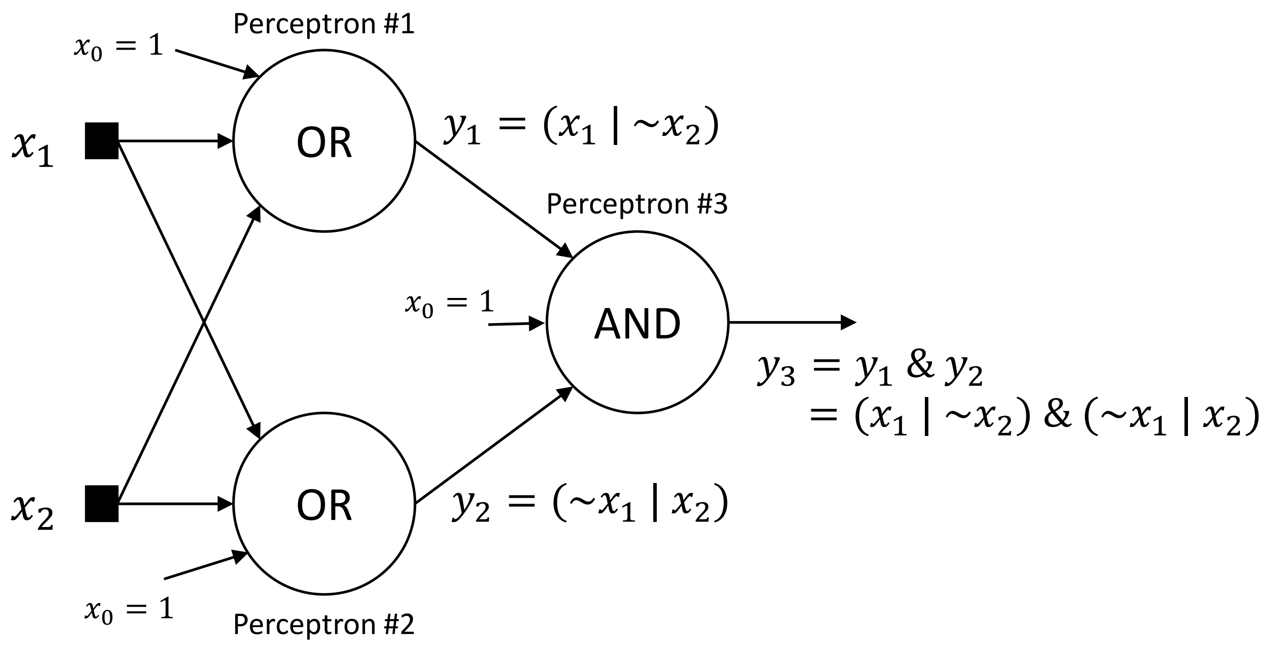

x1
x2
Camada
de Entrada
Camada
Escondida
Camada
de Saída
y


XNOR
x2
1
x1
0
1
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