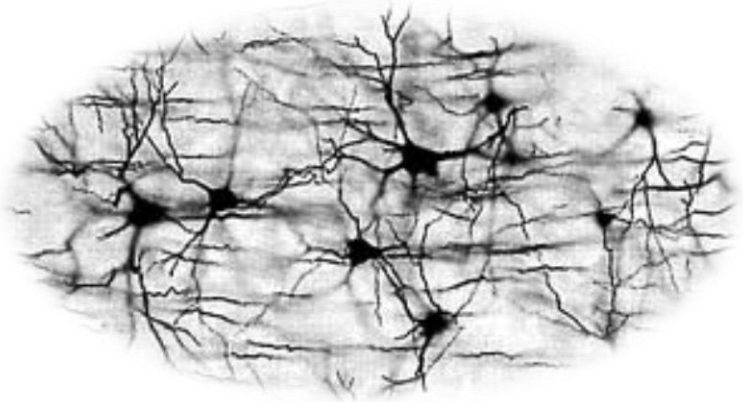
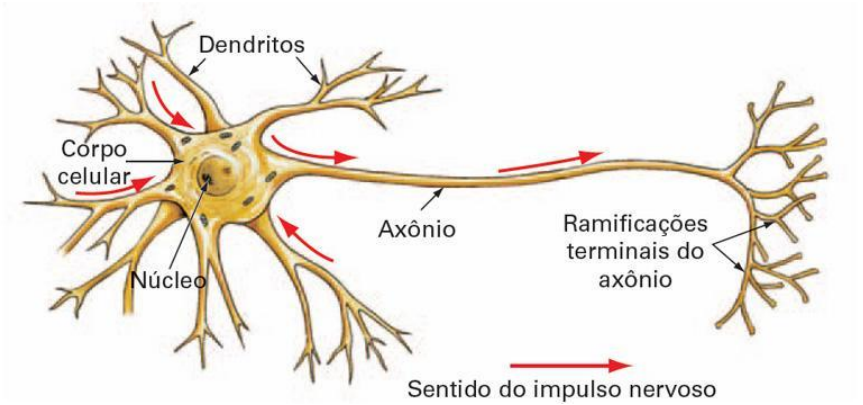
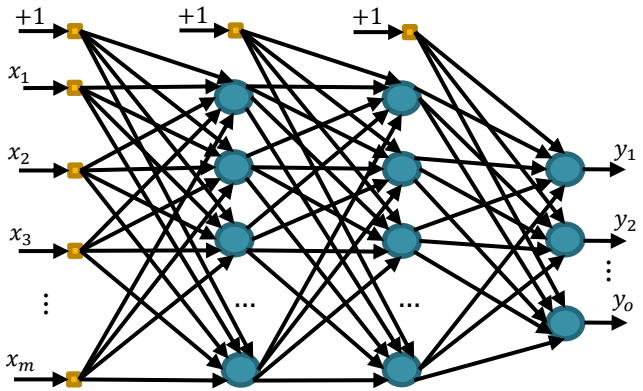
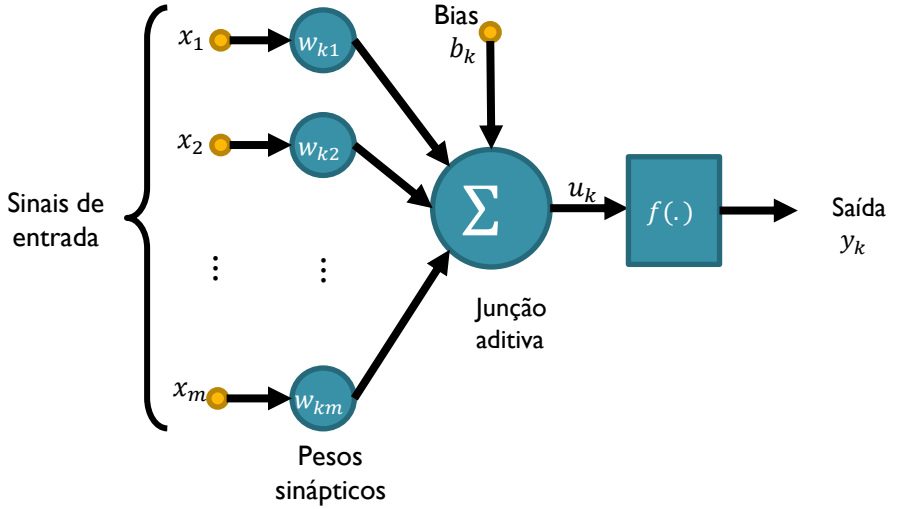
Redes Neurais Artificiais





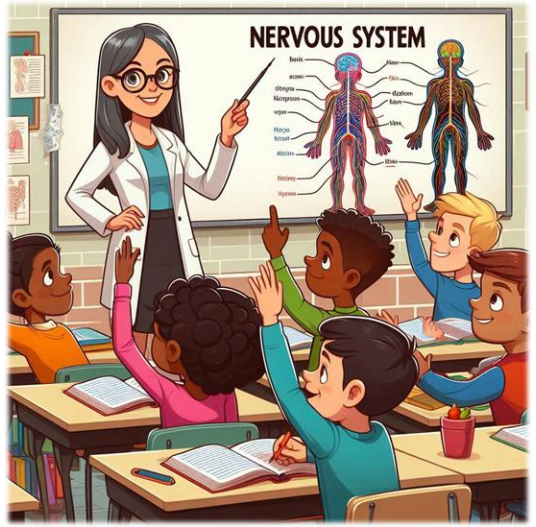


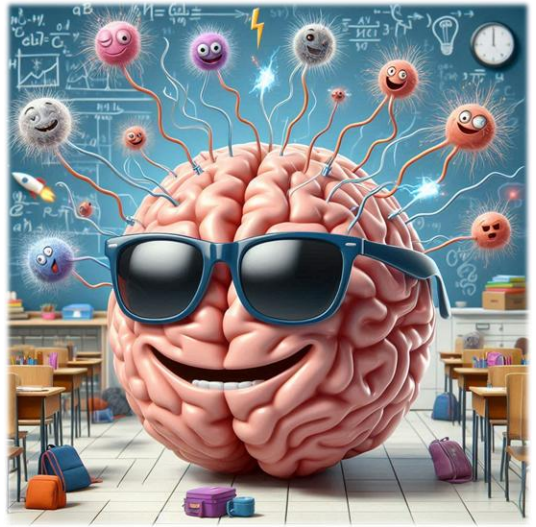


Fabricio Breve – www.fabriciobreve.com

Aula Anterior

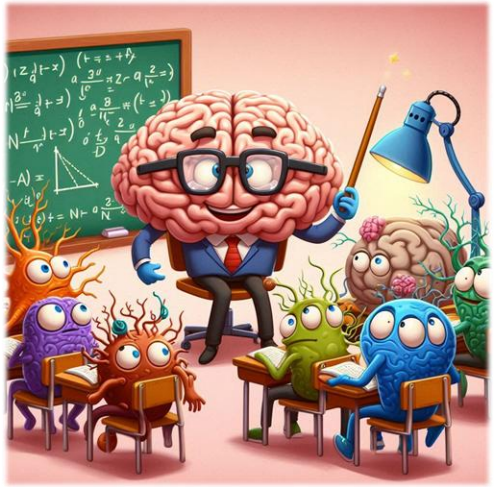
- Introdução à neurocomputação e aos desafios do cérebro.
- Comparação entre capacidades do cérebro e computadores (Deep Blue vs. Kasparov).
- Fundamentos dos neurônios, sinapses e neurotransmissores.
- Estrutura do sistema nervoso (central e periférico).
- Desenvolvimento neural a partir do tubo neural.
- Neuroplasticidade e neurogênese.
- Regiões e funções do cérebro (lobos e hemisférios).
- Transmissão sináptica e sinais elétrico-químicos.
- Organização em camadas e mapas sensoriais.
- Mitos e debates em neurociência.

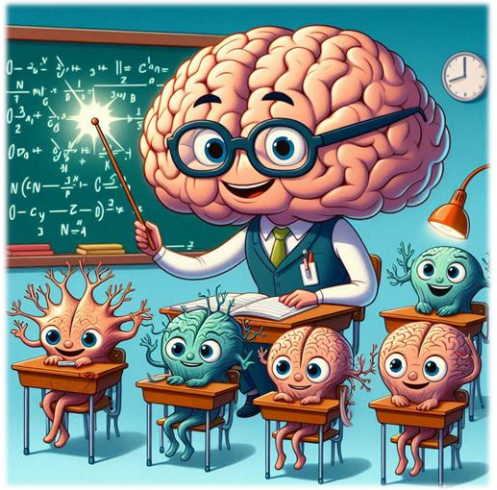




Agenda

- Por que Redes Neurais?
- Redes Neurais Artificiais
- Neurônios
  - Neurônio Natural
  - Neurônios Artificiais
  - Neurônio de McCulloch e Pitts
  - Neurônio Genérico
  - Neurônio Genérico Matematicamente
- Bias
  - Finalidade do Termo Bias
- Funções de Ativação
  - Função Linear
  - Função de Limiar (*Threshold*, *Step*)
  - Função Sigmóide / Logística
  - Função Tangente Hiperbólica
  - Função de Base Radial
  - ReLU (*Rectified Linear Unit*)
  - Outras Funções de Ativação
- Arquiteturas de Redes Neurais
  - Redes *Feedforward* de Uma Única Camada
  - Redes *Feedforward* de Múltiplas Camadas
  - Redes Recorrentes
- Abordagens de Aprendizado
  - Aprendizado Supervisionado
    - Exemplo: Problema de Classificação - Detecção de SPAM
    - Exemplo: Problema de Regressão - Como calcular o preço de uma casa?
    - Treinando uma Rede Neural
    - Função de Custo
    - Capacidade de Generalização
    - *Underfitting* e *Overfitting*
    - Estimando o Desempenho da Rede
    - Validação Cruzada
    - Matriz de Confusão
  - Aprendizado Não Supervisionado
    - Aprendizado Competitivo
    - Exemplo: Separação de Balões
    - Exemplo: Onde abrir pizzarias?
  - Aprendizado por Reforço



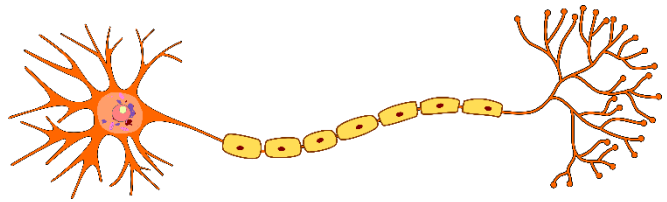


Por que Redes Neurais?

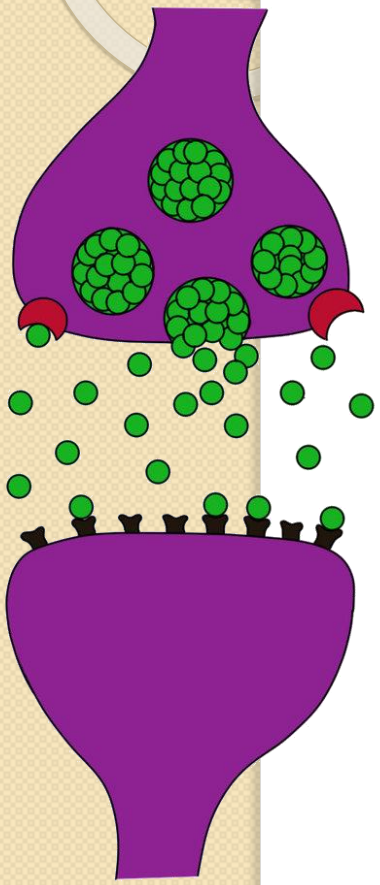
- Computadores convencionais são eficientes em várias áreas.
  - Entretanto, não têm obtido desempenho próximo da natureza em vários domínios de aplicação.
    - Sistema visual humano:
      - Reconhece rosto familiar em ambiente estranho (100~200 metros).
    - Sonar de morcegos:
      - Reconhece alvos (distância e velocidade).
      - Cérebro do tamanho de uma ameixa.

Redes Neurais Artificiais

- Características em comum com o sistema nervoso:
  - O processamento de informações ocorre em muitos elementos simples chamados neurônios (artificiais), nós (*nodes*) ou unidades (*units*).
  - Estes neurônios podem *receber* e *enviar estímulos* de e para outros neurônios e o ambiente.
  - Neurônios podem estar conectados a outros neurônios, portanto formando redes de neurônios ou *redes neurais (neural networks)*.



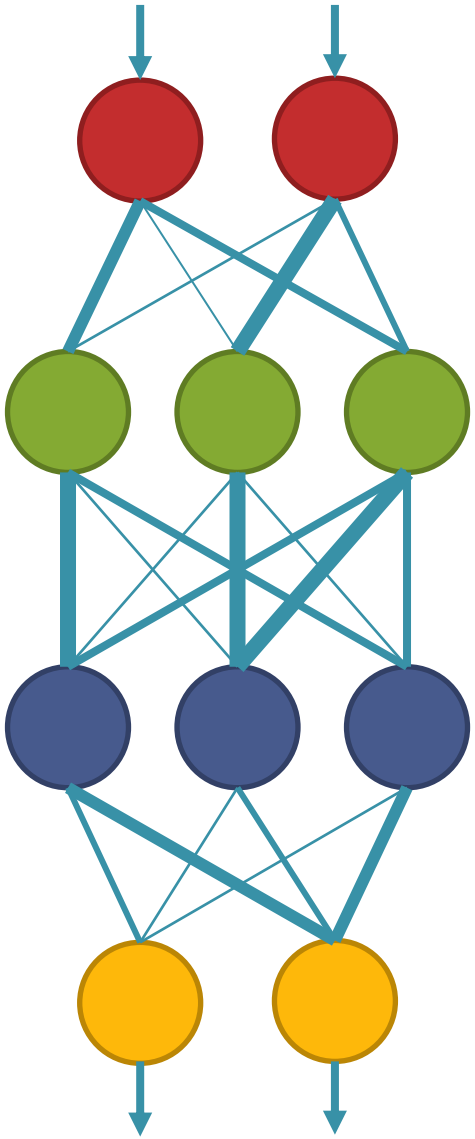
Redes Neurais Artificiais



- Características em comum com o sistema nervoso:
  - Informações (sinais) são transmitidos entre neurônios por elos chamados *sinapses*.
  - A eficiência de uma sinapse, representada por um valor de *peso (weight)* ou *força (strength)* associado, corresponde à informação armazenada no neurônio, e portanto na rede.
  - O conhecimento é adquirido do ambiente por um processo conhecido como *aprendizado (learning)*.
    - Responsável por *adaptar* a força das conexões (valor de peso) ao estímulo do ambiente.

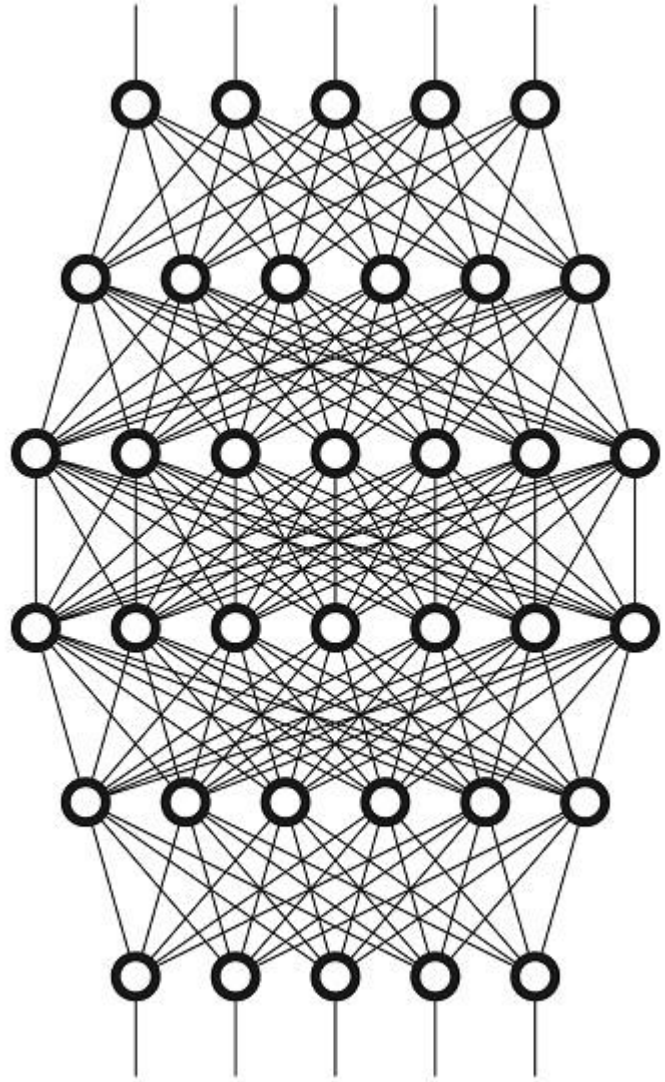
Redes Neurais Artificiais

- Onde o conhecimento está armazenado?
  - Nas forças de conexões (sinápticas) entre unidades que permitem que padrões sejam recriados.
  - Aprendizado é uma questão de encontrar as forças de conexões apropriadas para produzir padrões de ativação satisfatórios sob certas circunstâncias.
    - *Conhecimento distribuído* pelas conexões entre um grande número de neurônios.



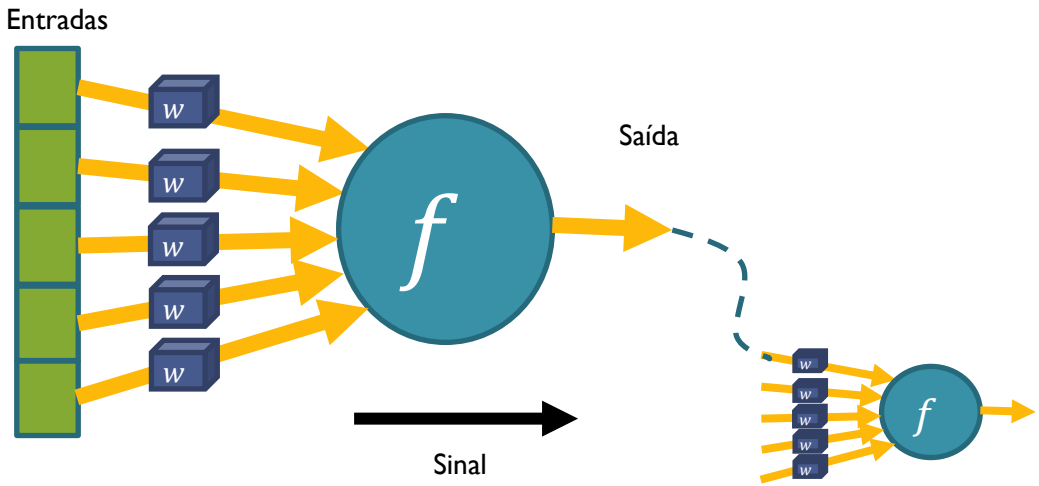
Redes Neurais Artificiais

- São compostas por:
  1. Um conjunto de neurônios artificiais.
     - Chamados nós, unidades ou simplesmente neurônios.
  2. O padrão de conectividade entre os neurônios.
     - Arquitetura ou Estrutura.
  3. Um método para determinar os pesos.
     - Treinamento ou Aprendizado.

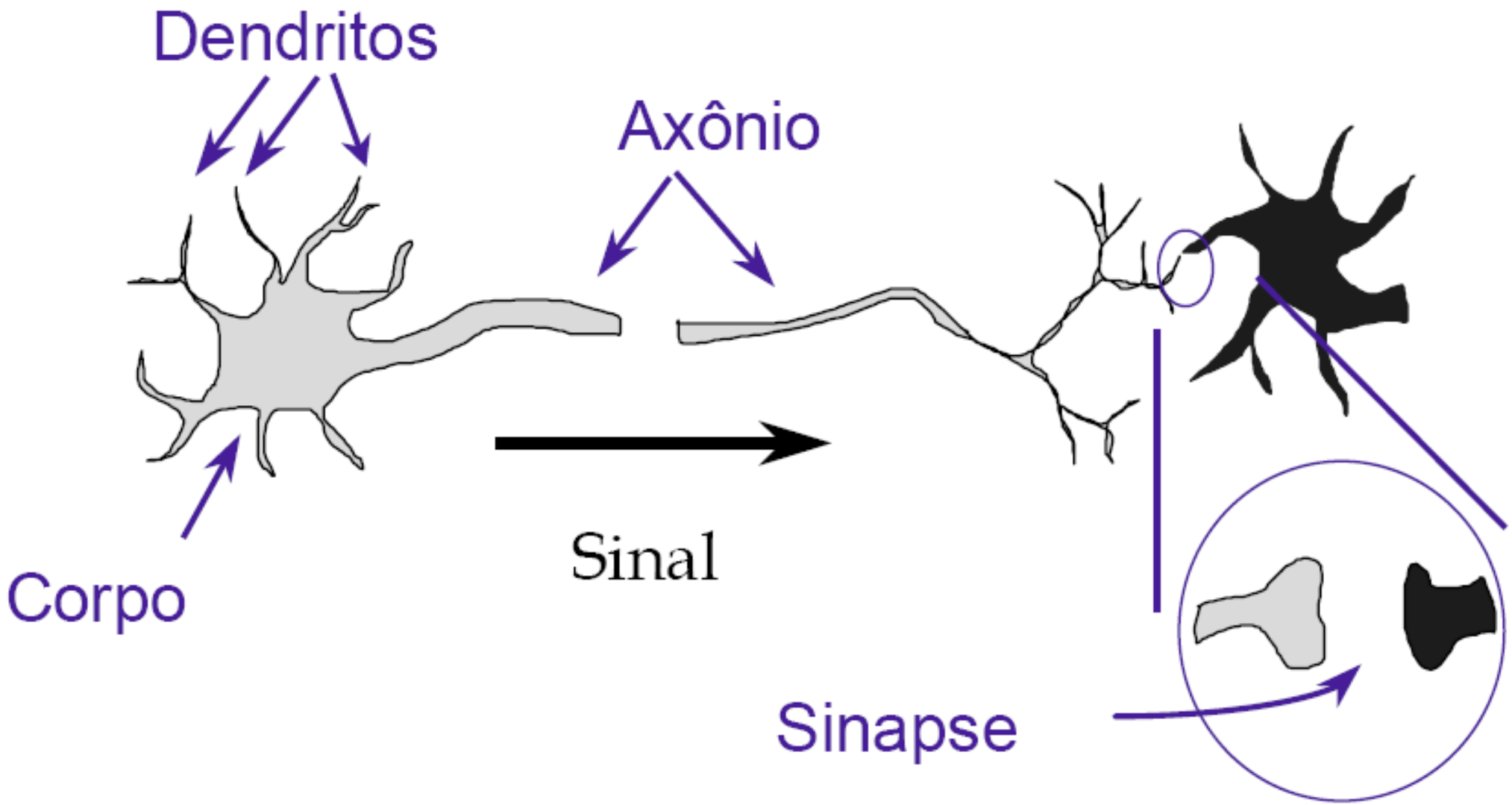


Redes Neurais Artificiais

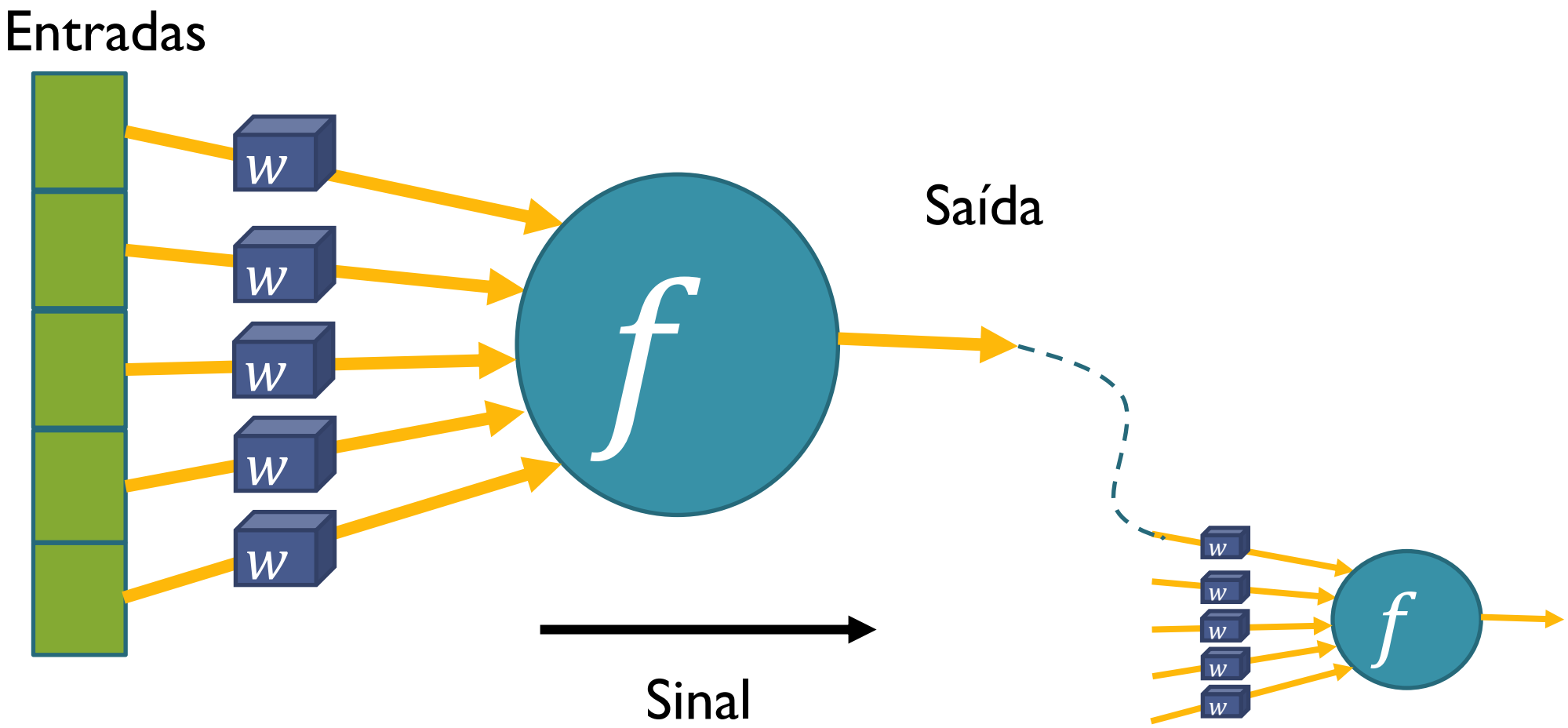
NEURÔNIOS ARTIFICIAIS



Neurônio Natural

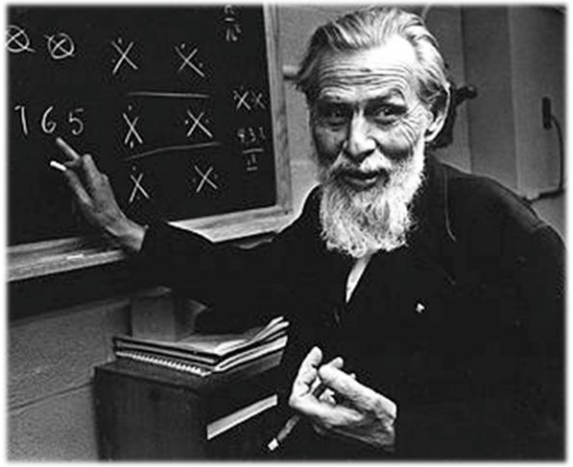


Neurônios Artificiais

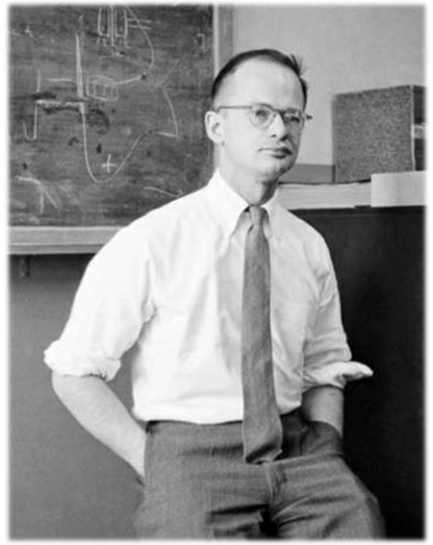


Neurônio de McCulloch e Pitts

- McCulloch e Pitts escreveram um artigo famoso e influente sobre cálculos que poderiam ser feitos com neurônios de dois estados.
  - Umas das primeiras tentativas de entender a atividade do sistema nervoso.
    - Baseada em unidades computacionais neuronais.
  - Modelo bastante abstrato das propriedades fisiológicas dos neurônios e suas conexões.



Warren Sturgis McCulloch [*1898 – †1969]

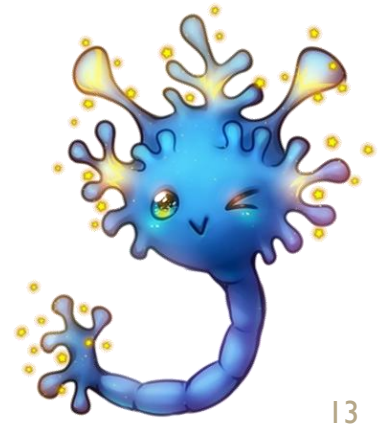


Walter Harry Pitts, Jr. [*1923 – †1969]

McCulloch, W.S., Pitts, W. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics* **5**, 115–133 (1943). https://doi.org/10.1007/BF02478259

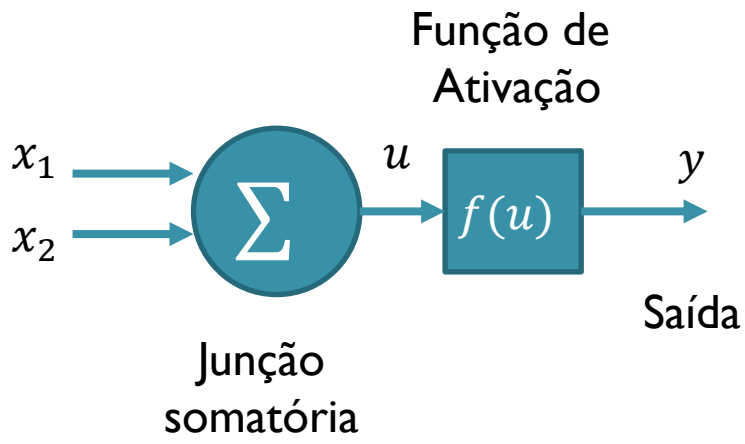
Neurônio de McCulloch e Pitts

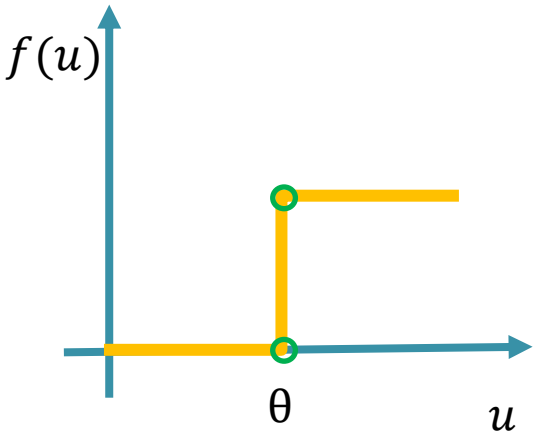
- Neurônio binário:
  - Sua saída pode ter apenas dois estados:
    - 0 ou 1
  - Cada neurônio tem um limiar θ fixo e recebe entrada das sinapses que tem pesos idênticos.
    - Se a soma dos valores de entrada é maior ou igual ao limiar θ o neurônio é ativado (responde com 1).
    - Senão permanece inativo (responde com 0).



Neurônio de McCulloch e Pitts

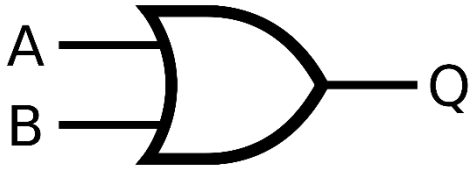
- Modelo bastante simples.
  - Porém já mostra características importantes da maioria dos modelos neuronais.
    - Integração dos estímulos de entrada.
    - Presença de uma função de ativação (limiar).

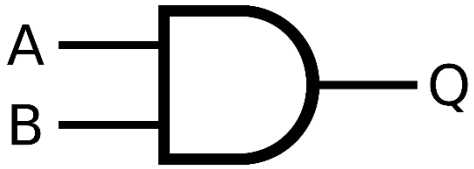




Neurônio de McCulloch e Pitts

- Exemplo:
  - Duas unidades excitativas x_1 e x_2 e limiar $\theta = 1$
    - Equivalente ao operador **OU**
  - Aumente o limiar para $\theta = 2$
    - Equivalente ao operador **E**

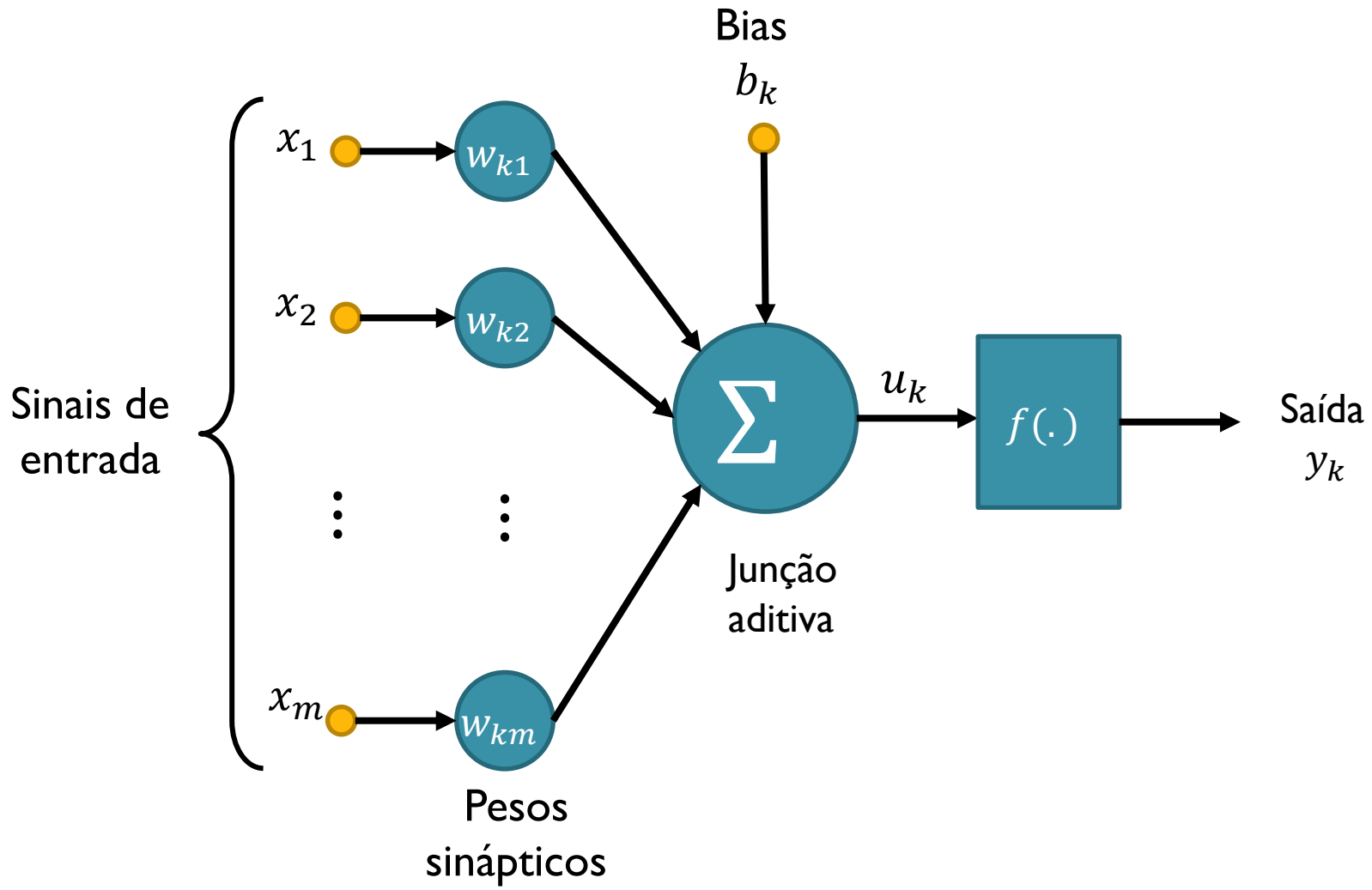




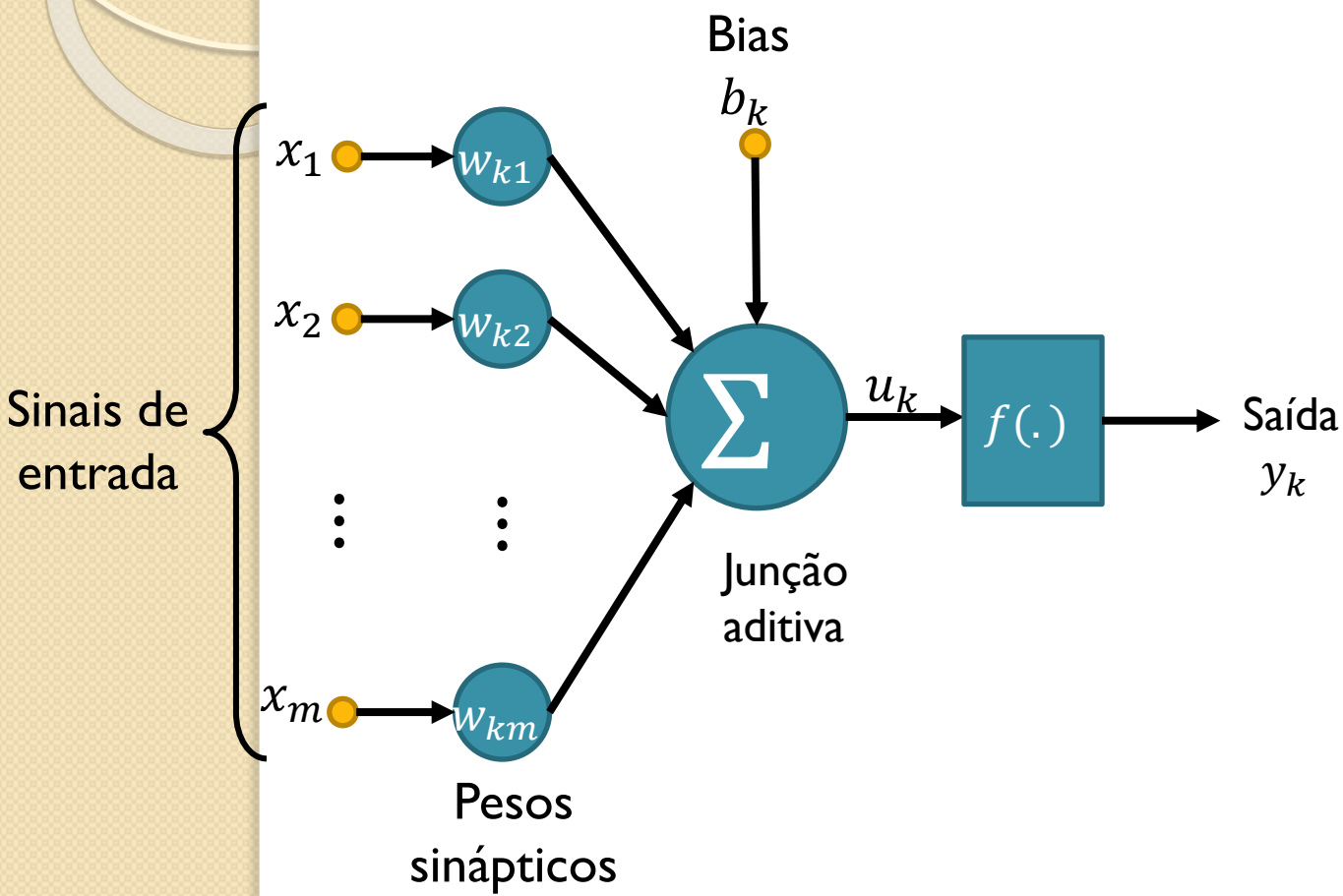
a	b	a OU b
0	0	0
0	1	1
1	0	1
1	1	1

a	b	a E b
0	0	0
0	1	0
1	0	0
1	1	1

Neurônio Genérico



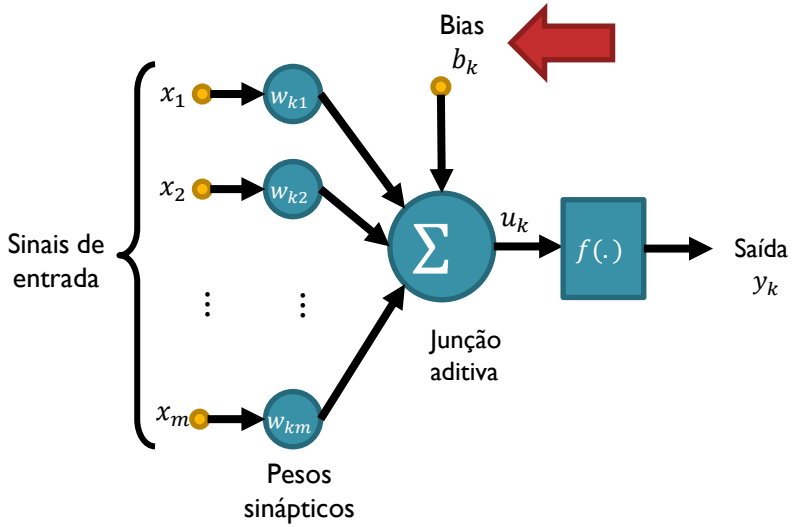
Neurônio Genérico



- w_{kj} é o peso da sinapse entre a entrada x_j e o neurônio k
- A junção de soma faz a soma de todos os sinais de entrada ponderados por seus respectivos pesos e adiciona também o valor do bias b_k
- Função de ativação é usada para limitar a amplitude da saída do neurônio.

Bias

- Aumenta ou diminui a entrada para a função de ativação.
  - Usado no lugar do limiar fixo.
  - *Bias* será ajustado durante o aprendizado.



- Exemplo:
  - No neurônio de McCulloch e Pitts:

$$y = f(u) = \begin{cases} 1 \text{ se } u \geq \theta \\ 0 \text{ caso contrário} \end{cases}$$

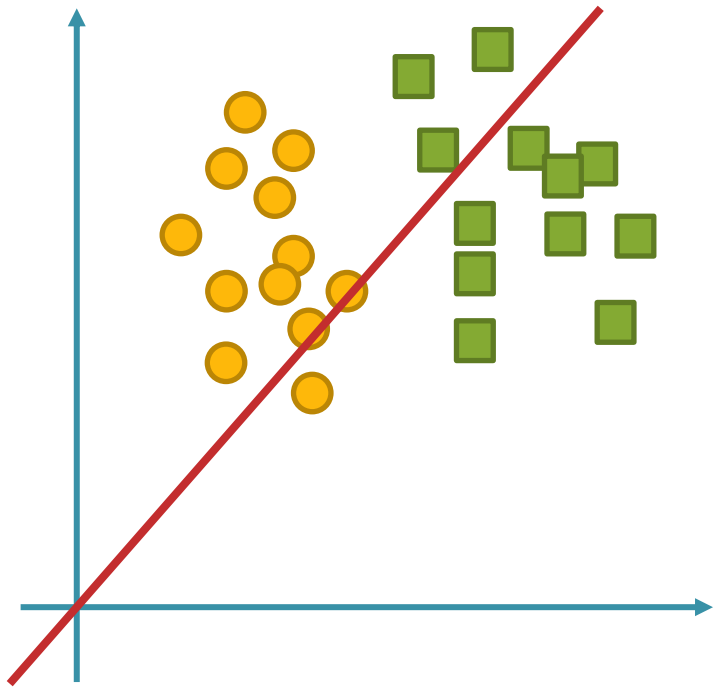
$$u = x_1 + x_2$$

  - Substituindo o limiar pelo bias:

$$y = f(u) = \begin{cases} 1 \text{ se } u \geq 0 \\ 0 \text{ caso contrário} \end{cases}$$

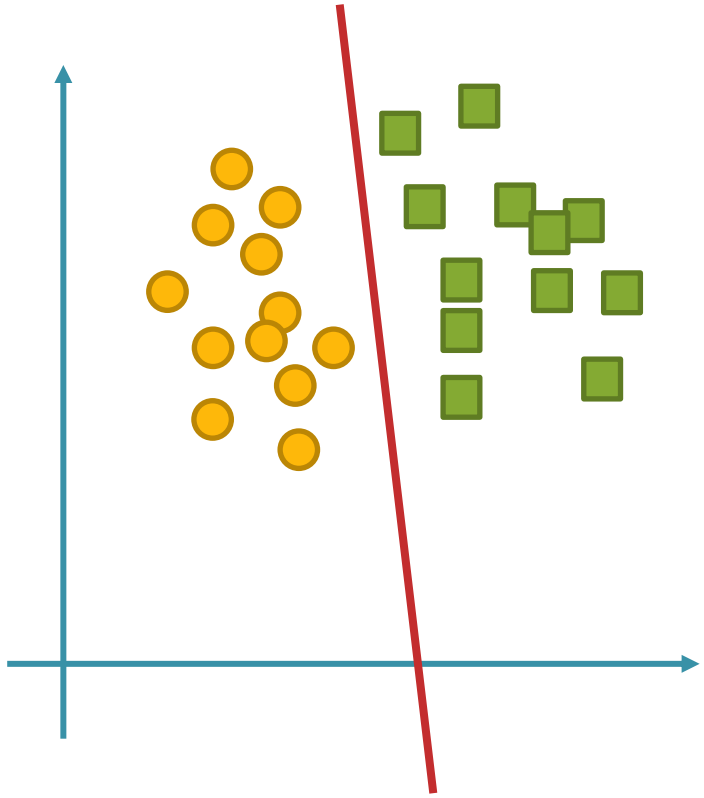
$$u = x_1 + x_2 + b$$

Finalidade do Termo Bias



$$\sum_j x_j w_j = 0$$

Define um hiperplano passando pela origem

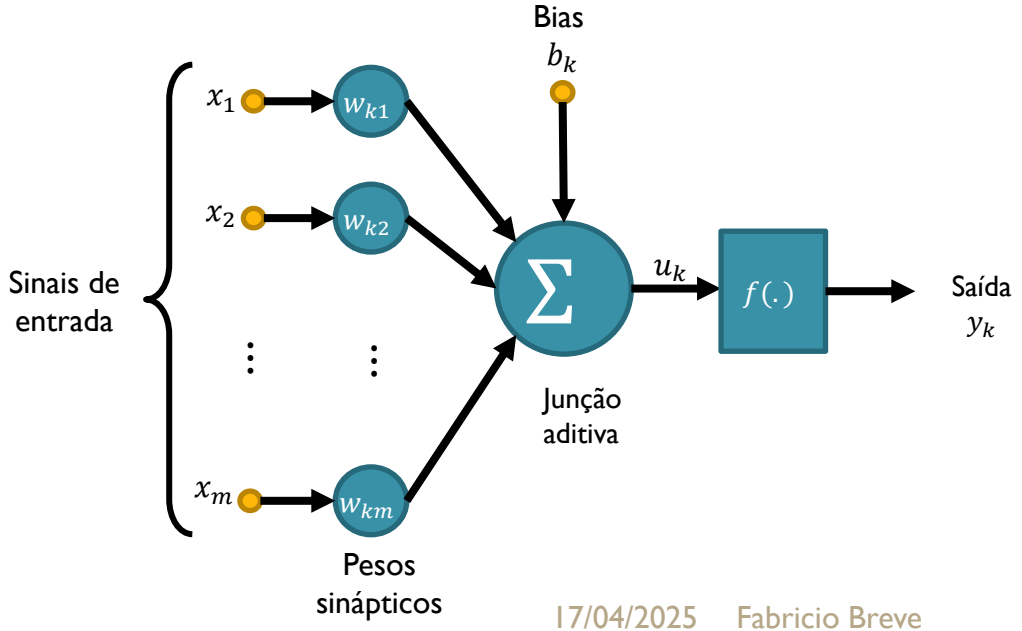


$$\sum_j x_j w_j + b = 0$$

Desloca-se o hiperplano da origem

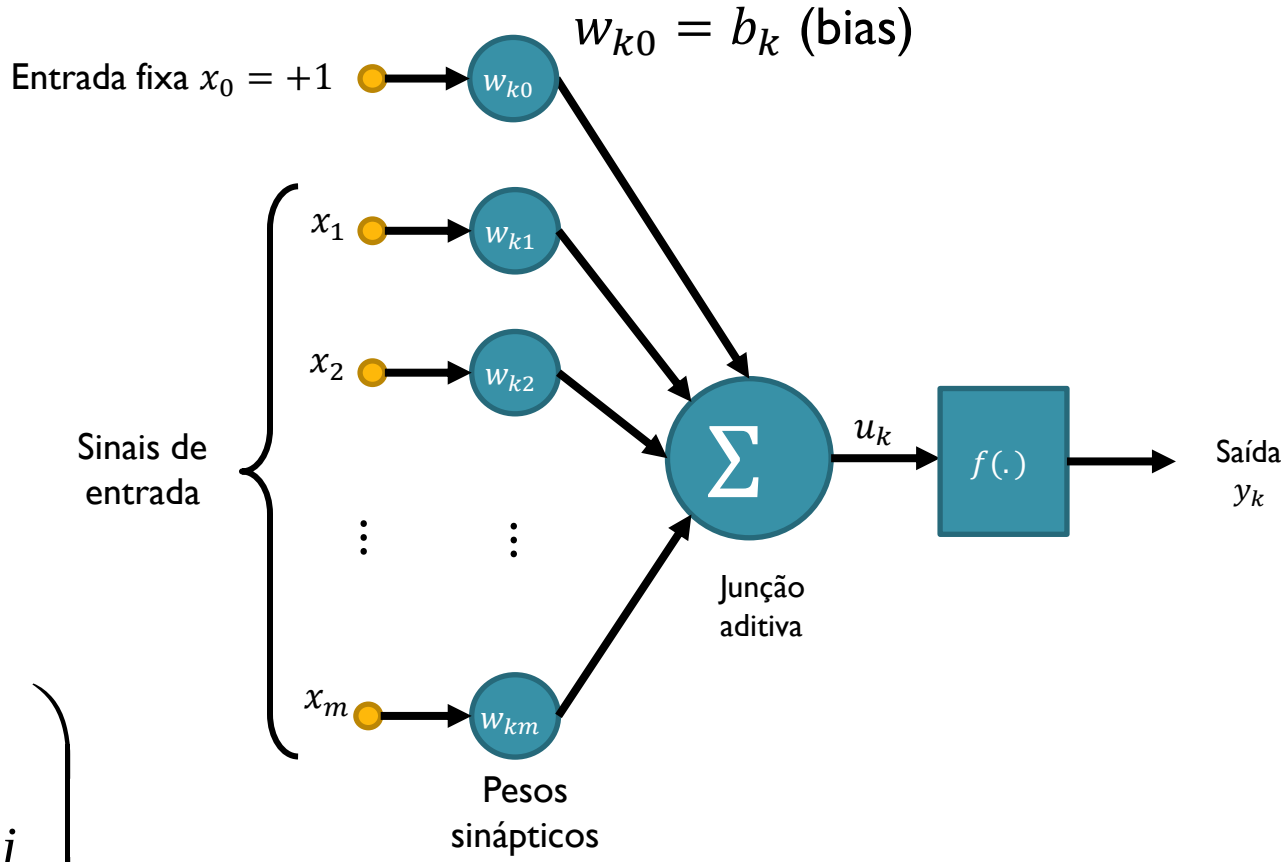
Neurônio Genérico Matematicamente:

$$y_k = f(u_k) = f\left(\sum_{j=1}^{m} w_{kj}x_j + b_k\right)$$



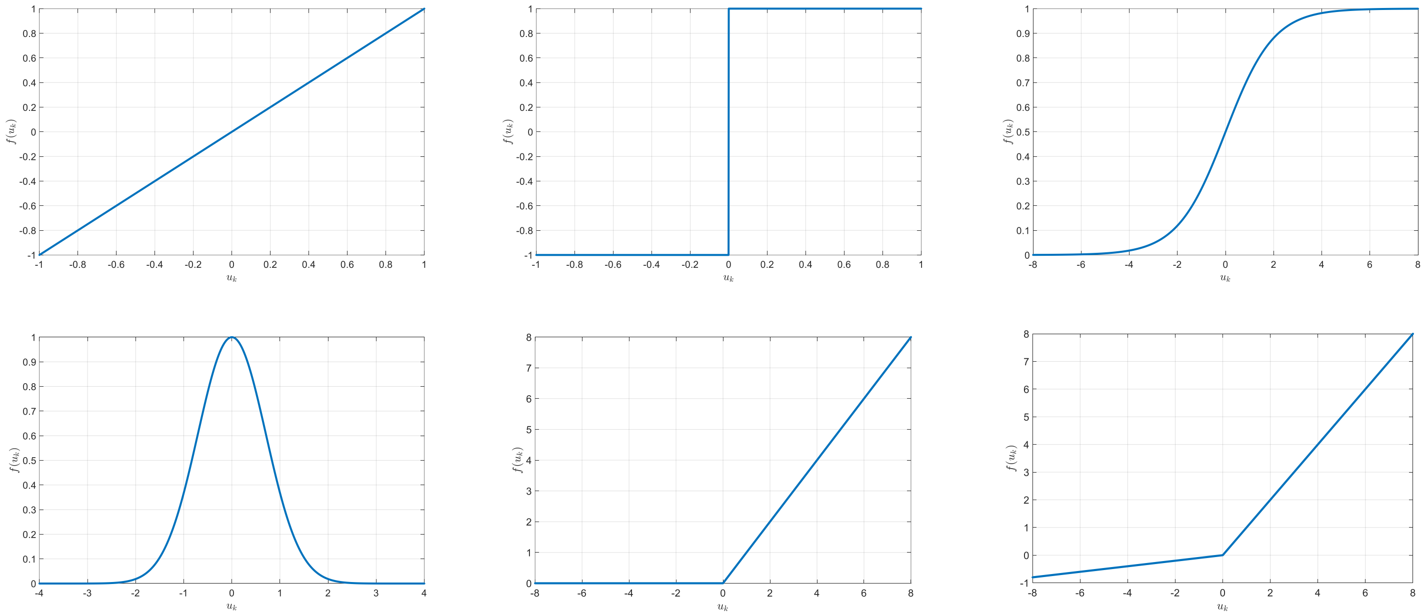
Simplificando...

- Considere bias b_k como sendo um peso w_{k0} de uma conexão de entrada $x_0 = 1$



$$y_k = f(u_k) = f\left(\sum_{j=0}^{m} w_{kj} x_j\right)$$

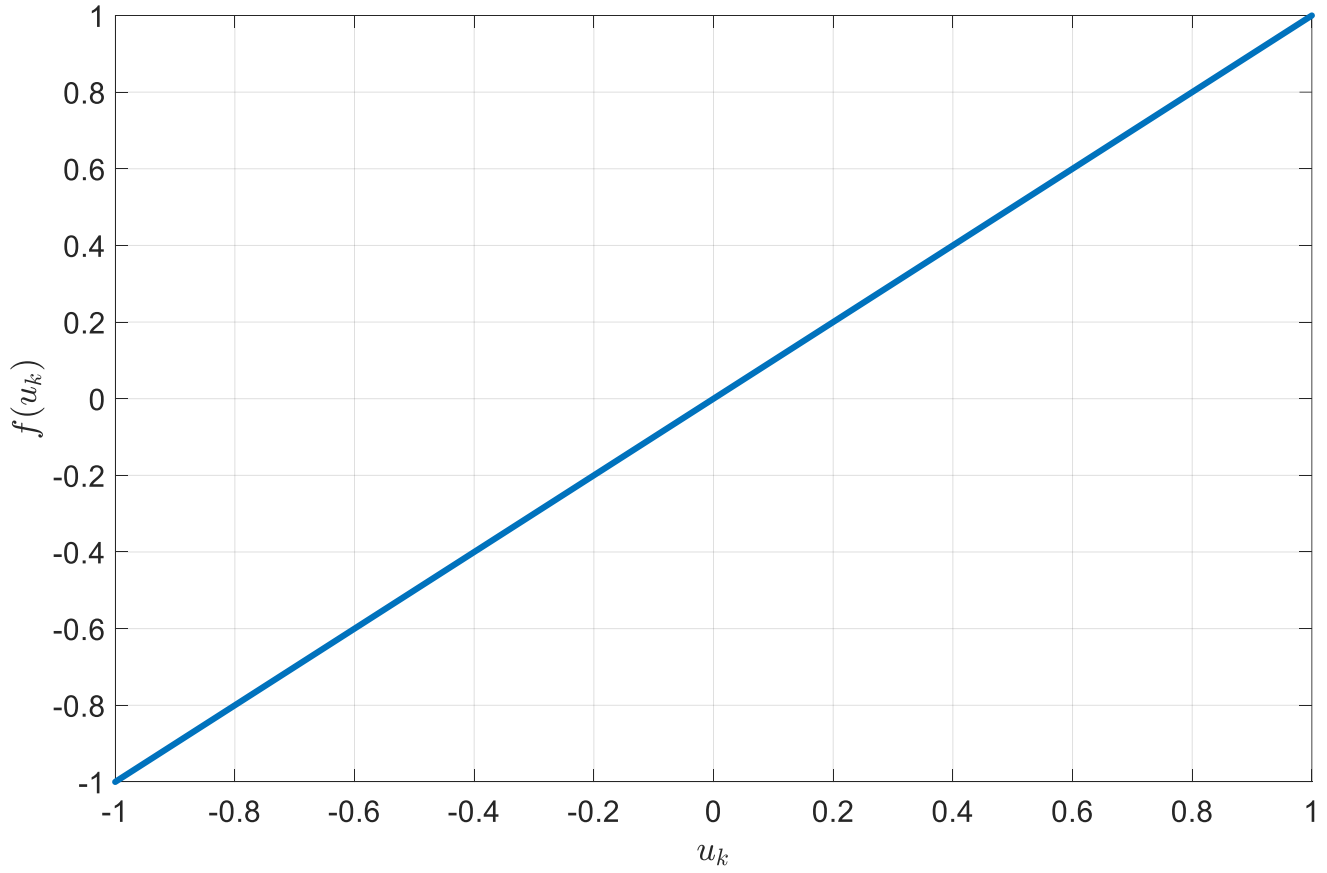
FUNÇÕES DE ATIVAÇÃO



Função Linear

- Relaciona a entrada diretamente com sua saída.

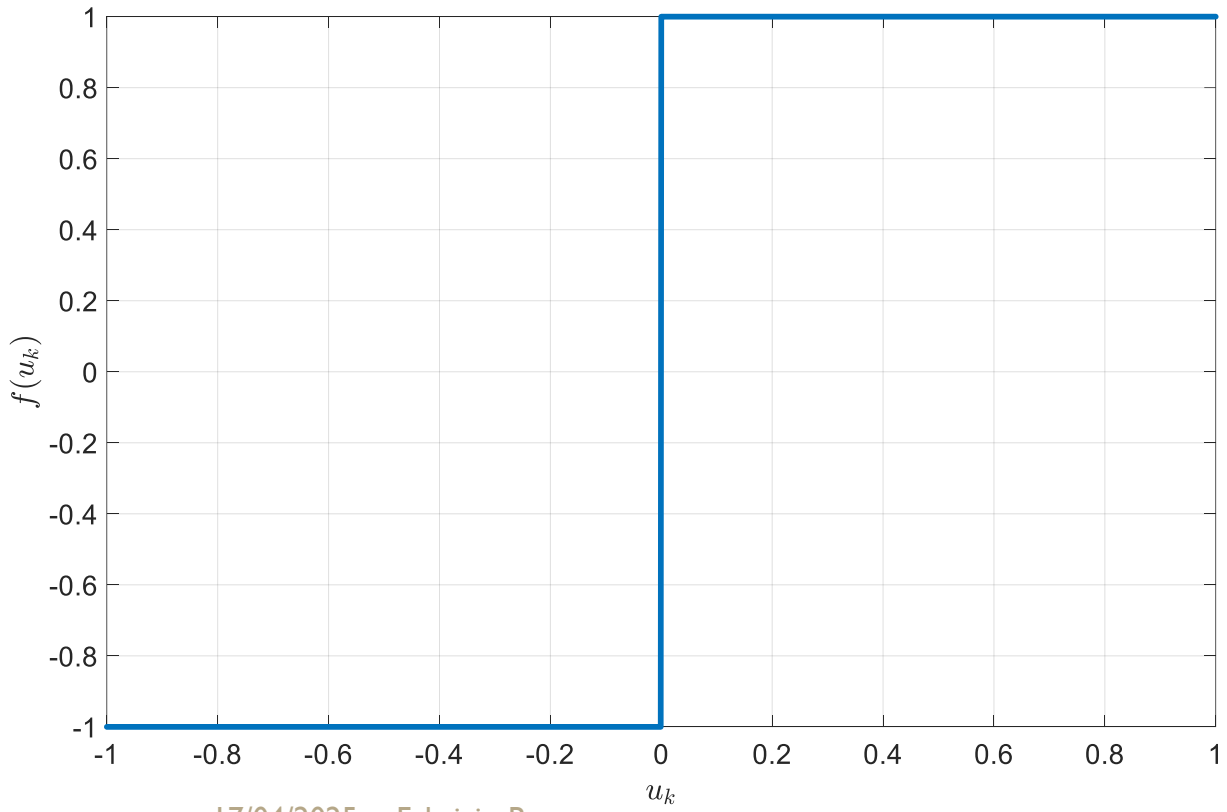
$$f(u_k) = u_k$$



Função de Limiar (*Threshold*, *Step*)

- A saída é 1 se a entrada é maior ou igual um certo limiar θ
- Senão é 0 ou -1
  - Depende de a função ser:
    - Binária $\{0,1\}$
    - Bipolar $\{-1,1\}$
      - Usada no trabalho de McCulloch e Pitts.

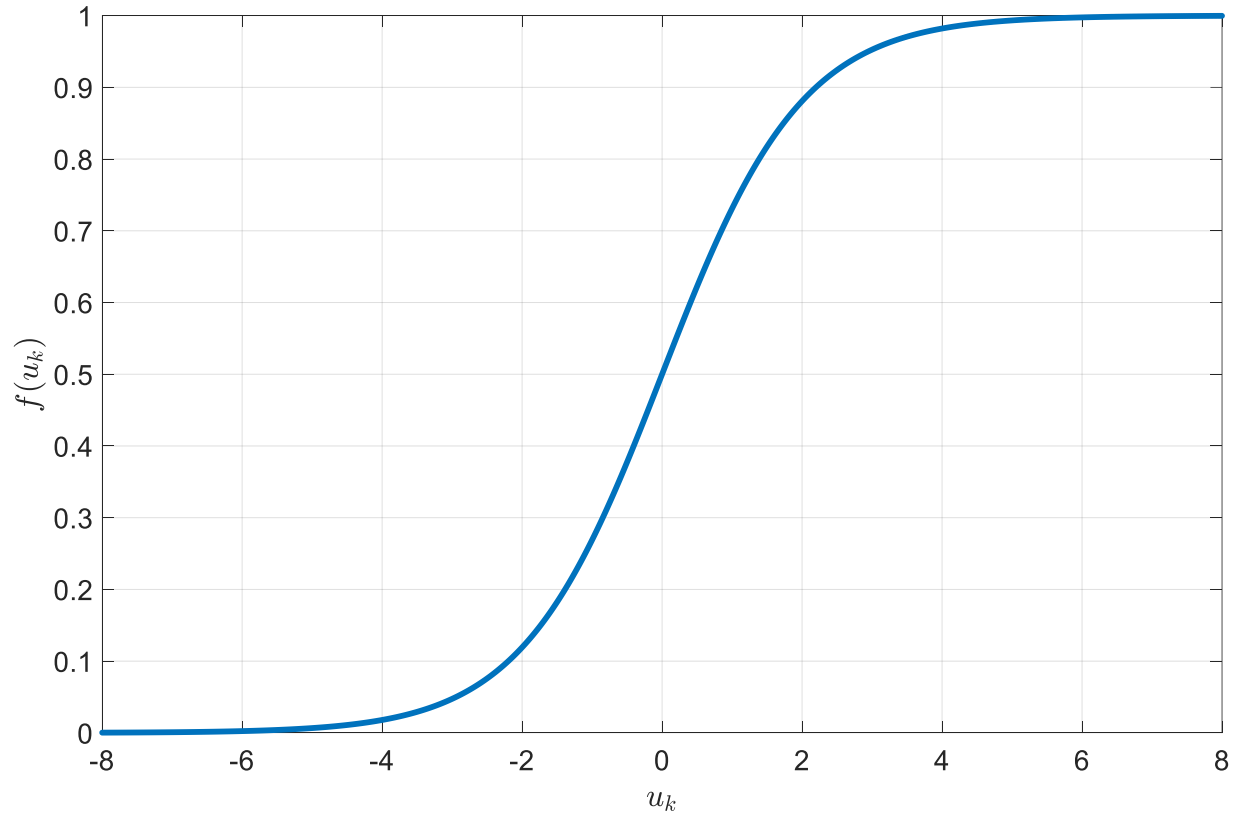
$$f(u_k) = \begin{cases} 1 & \text{se } u_k \geq \theta \\ -1 & \text{se } u_k < \theta \end{cases}$$



Função Sigmóide / Logística

- Função estritamente crescente que oferece um balanceamento adequado entre comportamento linear e não-linear.
- Por muitos anos foi a função de ativação mais popular.
- Em forma de *S*.
- Produz valores de saída entre 0 e 1.

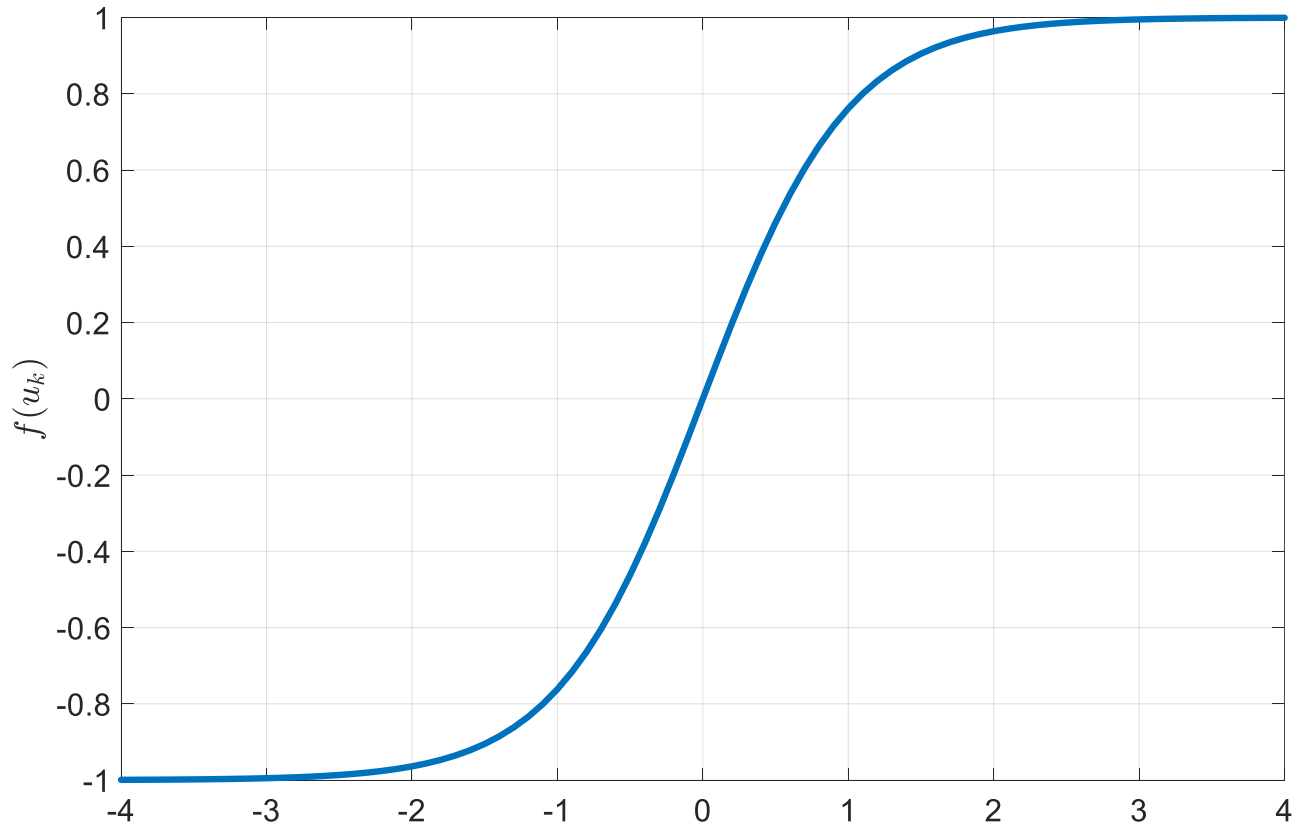
$$f(u_k) = \frac{1}{1+\exp(-u_k)}$$



Função Tangente Hiperbólica

- Similar à função sigmoide logística.
- Porém varia de -1 a 1

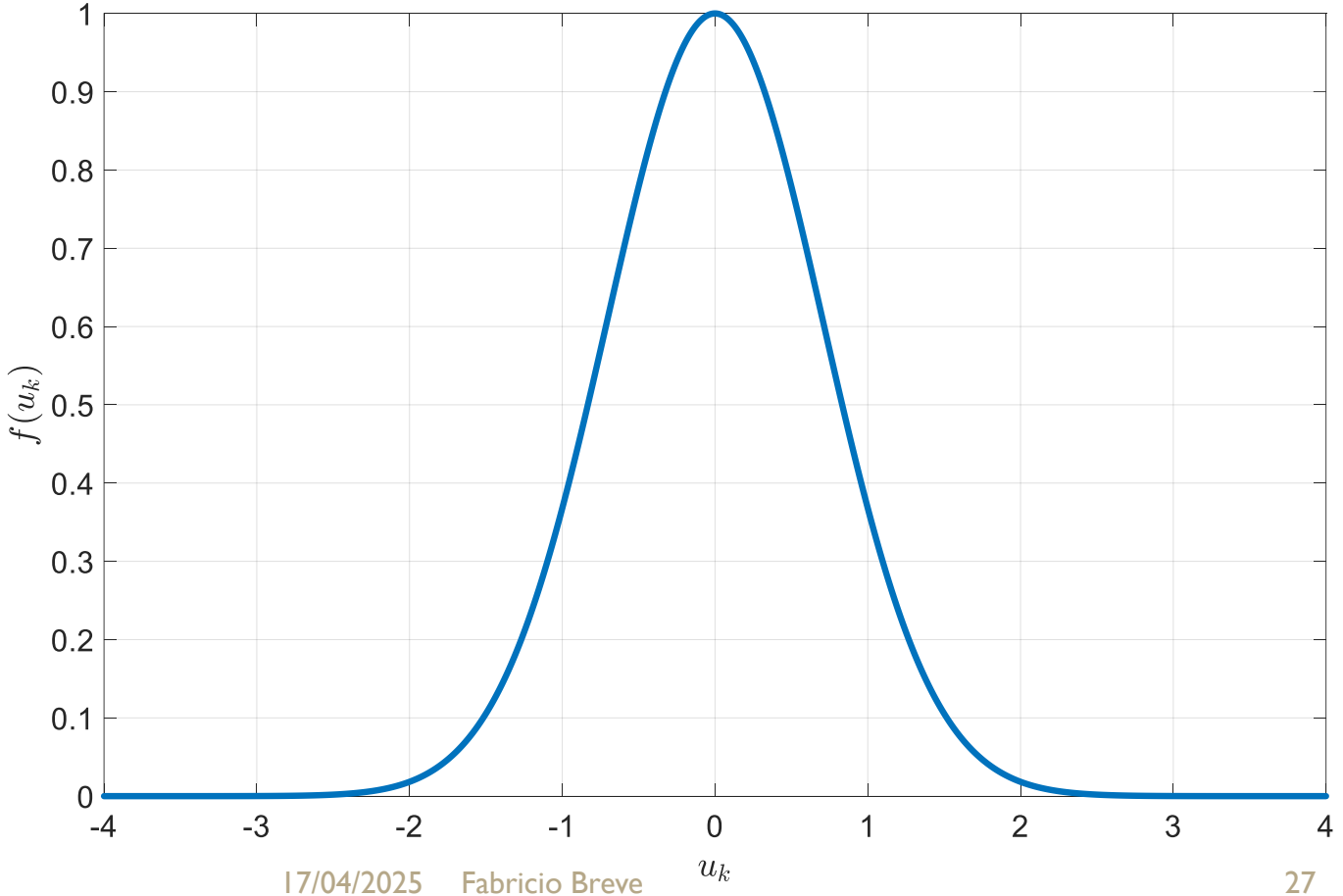
$$f(u_k) = \frac{2}{1 + \exp(-u_k)} - 1$$



Função de Base Radial

- Função não monotônica que é simétrica em torno de sua base.
  - Exemplo: Gaussiana.
    - Principal função de ativação nas *redes neurais de função de base radial*.

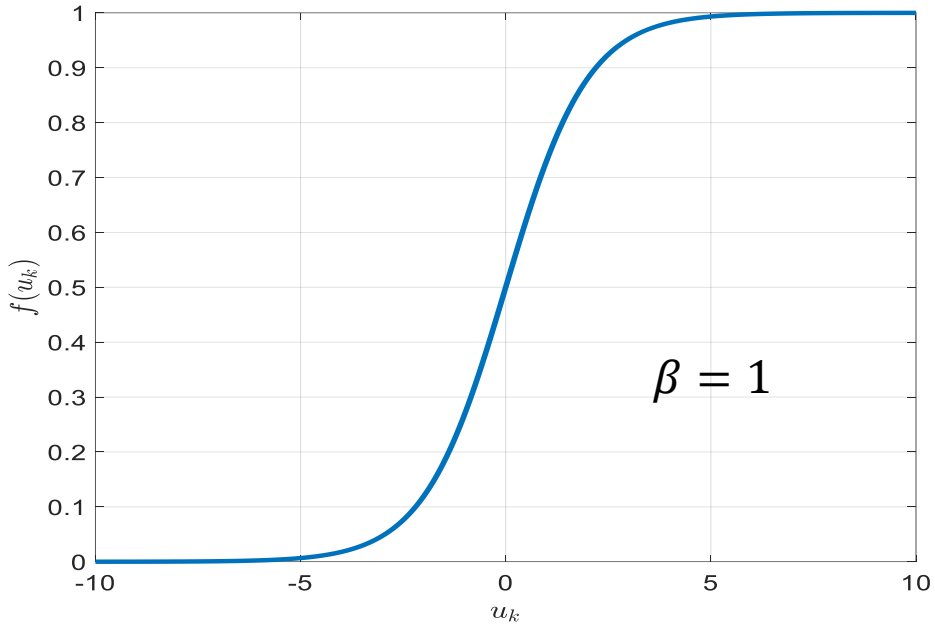
$$f(u_k) = \exp\left(-u_k^{\,2}\right)$$

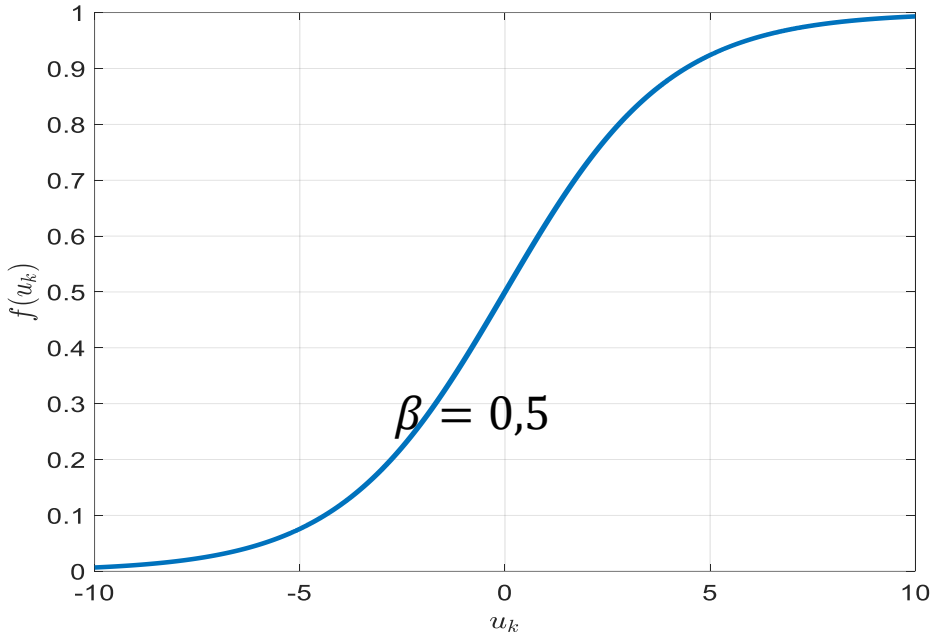


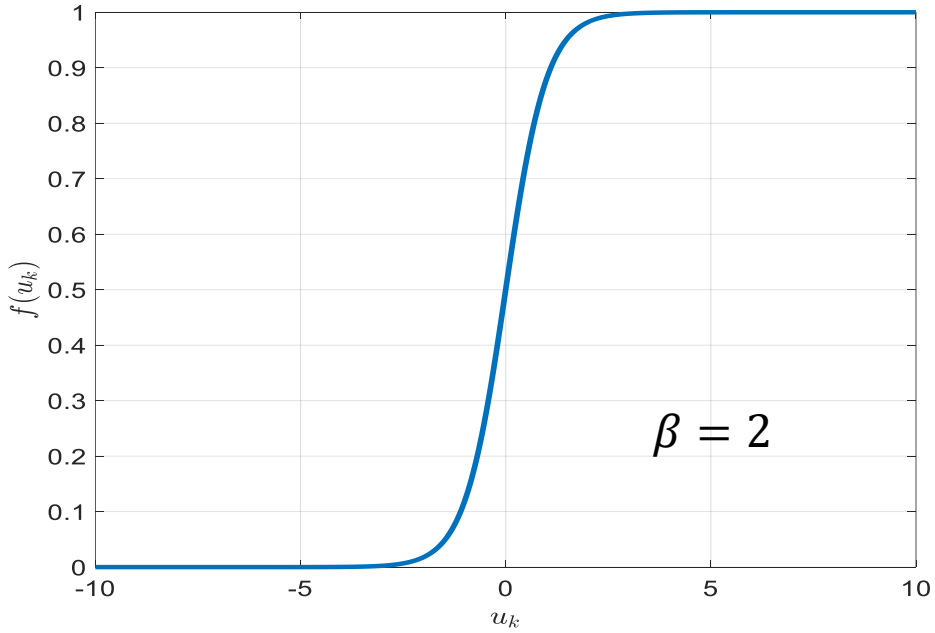
Funções de Ativação

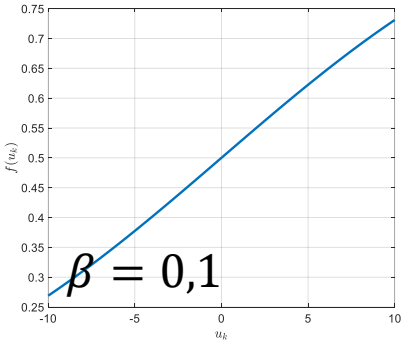
- As funções mostradas e ilustradas mostram apenas a forma geral de tais funções.
  - Estas formas podem ser modificadas.
    - Exemplo:
      - Multiplicando u_k por uma constante β na função logística controlamos a suavidade da função.
        - Valores altos de β tornam a função parecida com uma função de limiar.
        - Valores baixos de β tornam a função parecida com uma função linear.

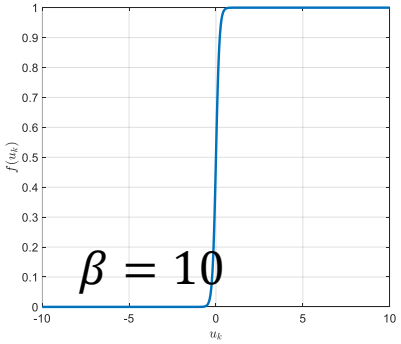
$$f(u_k) = \frac{1}{1+\exp(-\beta.u_k)}$$









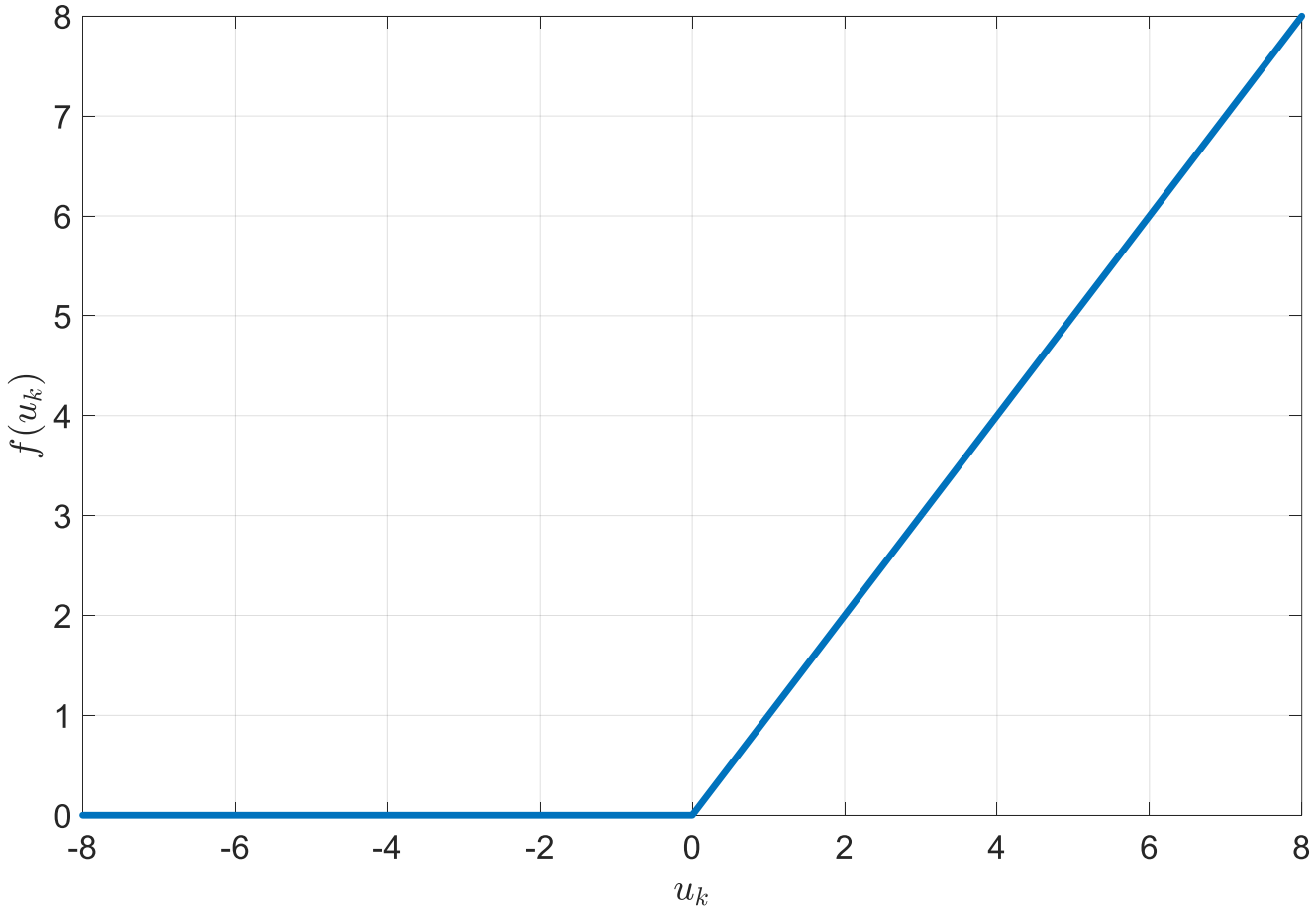


$$f(u_k) = \frac{1}{1 + \exp(-\beta . u_k)}$$

ReLU (Rectified Linear Unit)

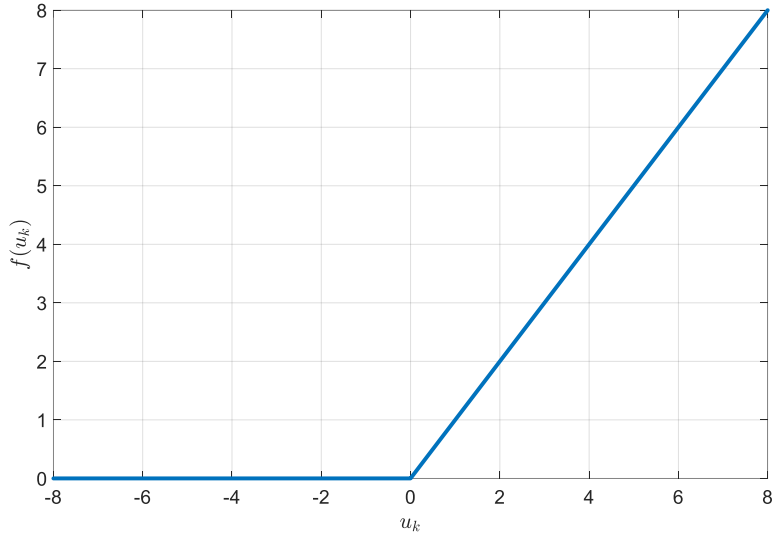
- Se tornou bastante popular com a ascensão das redes de aprendizado profundo (*deep learning*).

$$f(u_k) = \max(0, u_k)$$



ReLU (Rectified Linear Unit)

- Vantagens:
  - Computacionalmente eficiente.
    - Convergência rápida.
  - Não-linear.
    - Permite a retropropagação.
- Desvantagens:
  - Neurônios podem "morrer".
    - Na faixa menor que zero o gradiente é zero, portanto a retropropagação não funciona e não há aprendizado.



Retropropagação é uma forma de aprendizado para redes de múltiplas camadas.

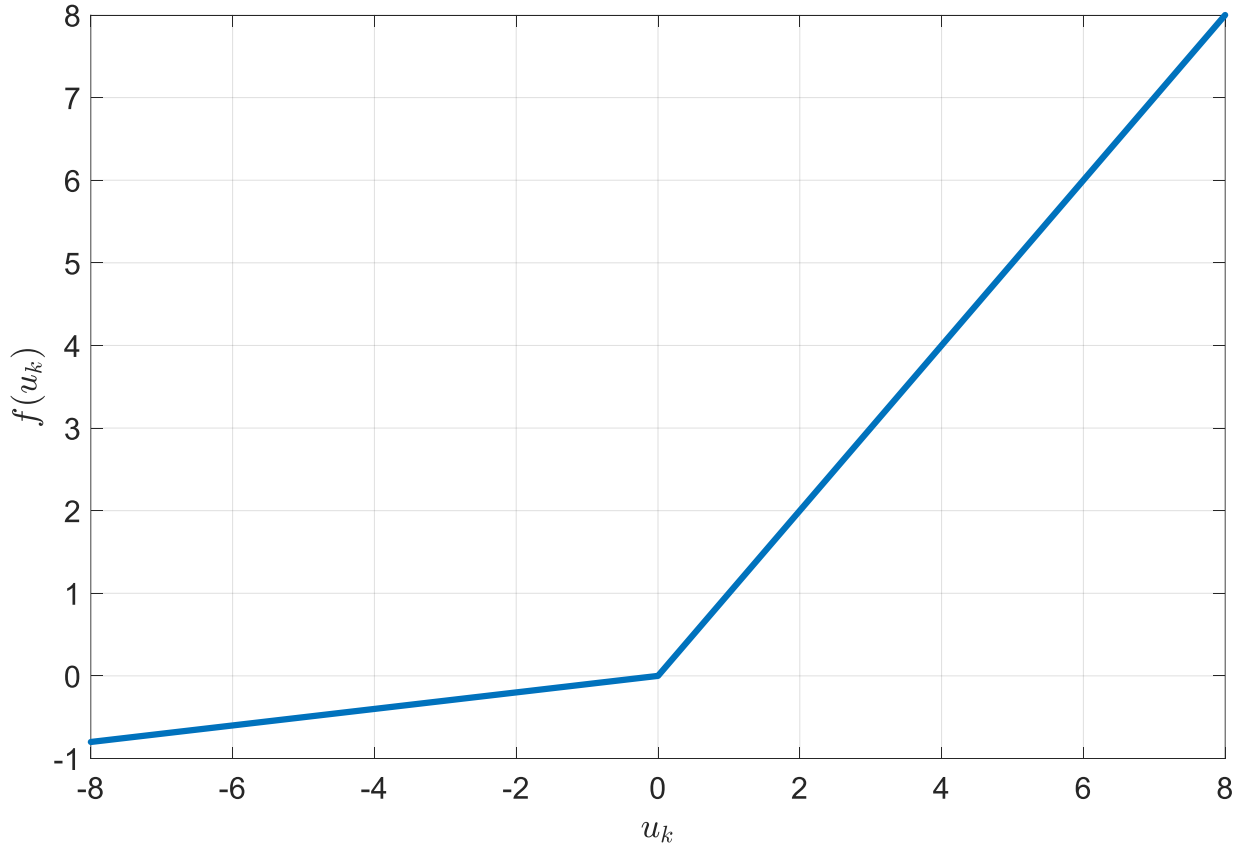
ReLU (Rectified Linear Unit)

- Novas funções de ativação vem sendo propostas para tentar contornar os problemas da ReLU:
  - Leaky ReLU
  - Parametric ReLU
  - ELU (Exponencial Linear Unit)
  - SWISH (Self-Gated Activation Function)

Leaky ReLU

- Vantagem:
  - Resolve o problema dos neurônios mortos da ReLU.
- Desvantagem:
  - Introduz o hiperparâmetro α
    - Normalmente ajustado para $0{,}01$.
      - Mas os resultados podem ser melhores se for ajustado aos dados.

$$f(u_k) = \begin{cases} u_k \text{ se } u_k \geq 0 \\ \alpha u_k \text{ se } u_k < 0 \end{cases}$$



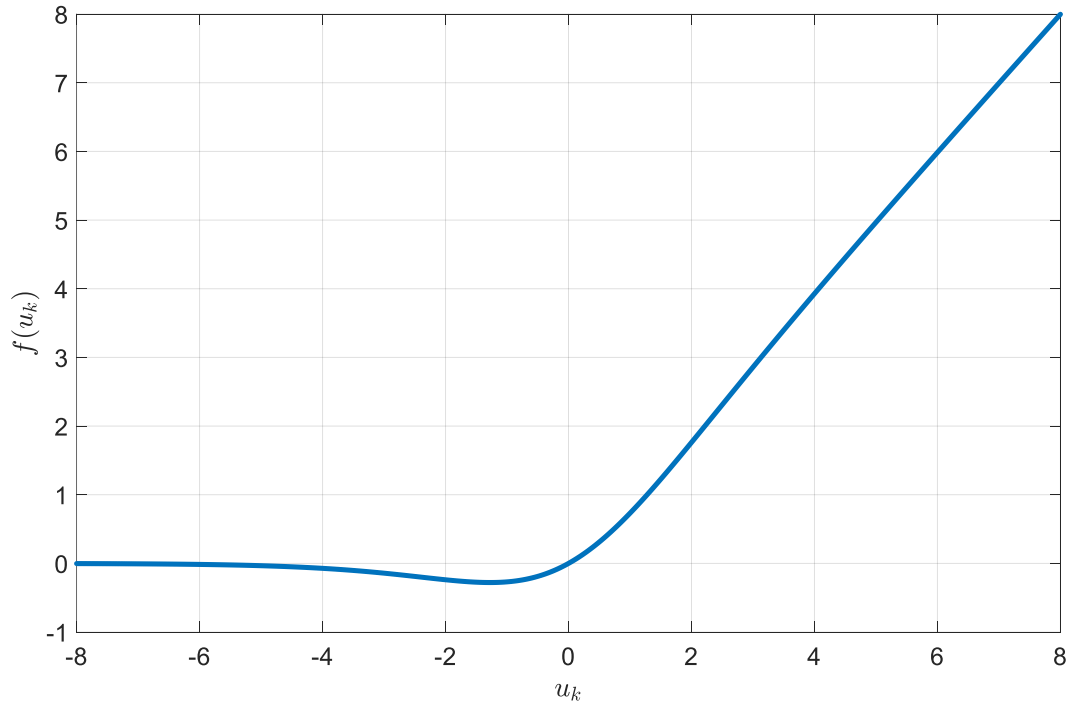
SWISH (Self-Gated Activation Function)

- Uma boa alternativa à ReLU.
  - Em geral consegue resultados melhores ou iguais.
- Vantagens:
  - Não monotônica.
    - Aumenta a capacidade de armazenamento de informação e discriminação do modelo.
  - Ilimitada acima e limitada abaixo.
    - Não satura com valores positivos e tende a uma constante com valores negativos.
  - Suave.
- Desvantagem:
  - Computacionalmente custosa.

$$f(u_k) = u_k \sigma(u_k)$$

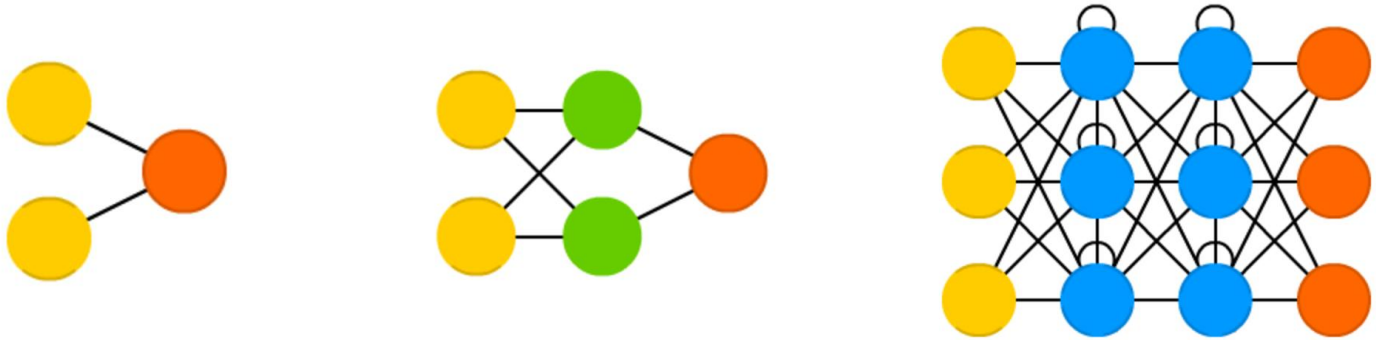
onde σ é a função sigmoide que já vimos:

$$\sigma(u_k) = \frac{1}{1 + \exp(-\beta . u_k)}$$



Redes Neurais Artificiais

ARQUITETURA DE REDES NEURAIS

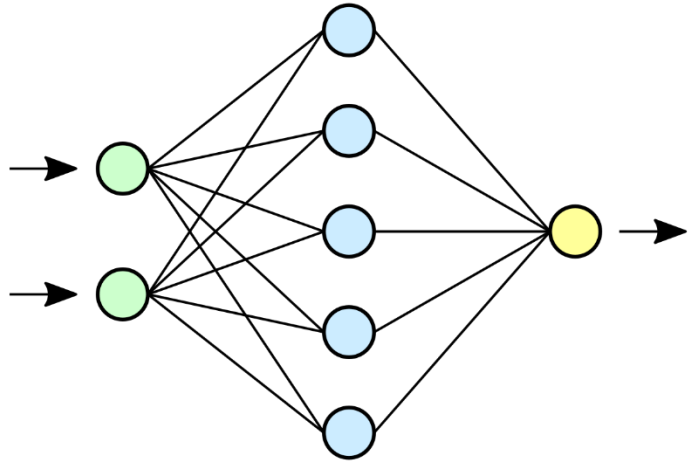


Arquitetura de Redes Neurais

- O comportamento complexo do sistema nervoso se deve à interação de um grande número de neurônios.
  - Um neurônio sozinho não tem sentido.
- Não se sabe muito sobre como essa interconexão é feita.
  - Temos apenas alguns conhecimentos sobre a interconexão em áreas específicas do cérebro, mas pouco sabemos sobre o cérebro como um todo.
    - Sabemos que os neurônios do córtex podem ser divididos em camadas.

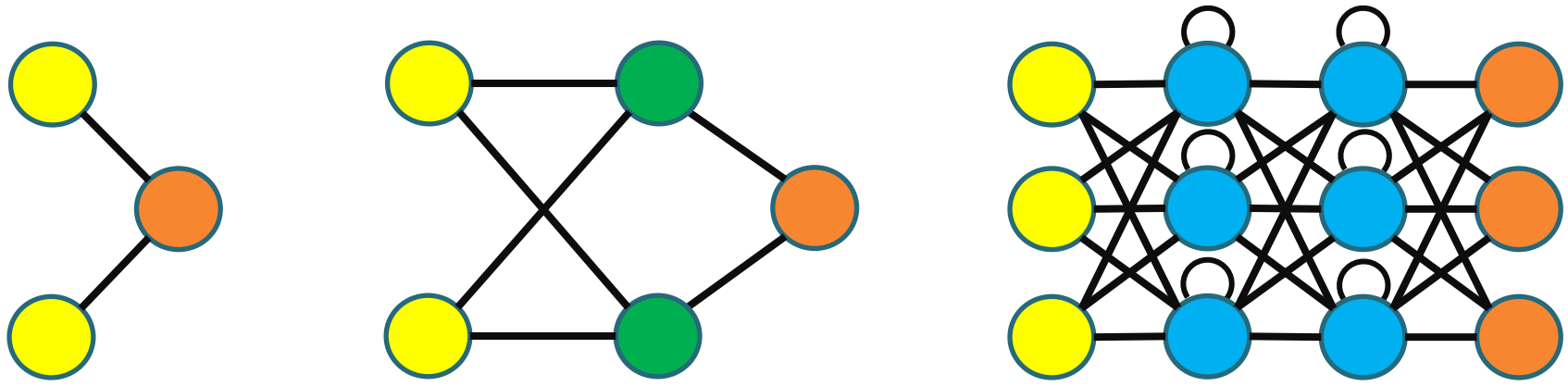
Arquitetura de Redes Neurais

- Redes Neurais Artificiais:
  - Camadas de Neurônios se referem a camadas funcionais.
    - Arquiteturas padronizadas especialmente projetadas com foco no propósito de solução de problemas.
    - Há basicamente três tipos possíveis de camada:
      - Uma camada de entrada:
        - Recebe estímulo do ambiente.
      - Uma ou mais camadas intermediárias (ocultas):
        - Não tem contato direto com o ambiente.
      - Uma camada de saída:
        - Envia saída para o ambiente.



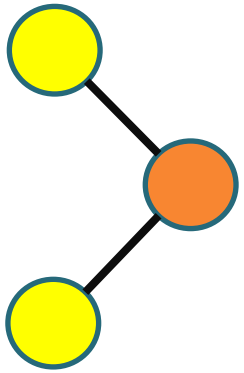
Arquitetura de Redes Neurais

- Existem três tipos básicos de arquitetura:
  - Redes *feedforward* de uma única camada.
  - Redes *feedforward* de múltiplas camadas.
  - Redes recorrentes.



Redes *Feedforward* de Uma Única Camada

- É o tipo mais simples.
- Uma camada de nós de entrada que alimenta uma camada de nós de saída.
  - Camada de entrada:
    - Também é chamada camada sensorial.
      - Apenas recebe o sinal do ambiente e o propaga adiante.
  - Camada de Saída:
    - Elementos de processamento.



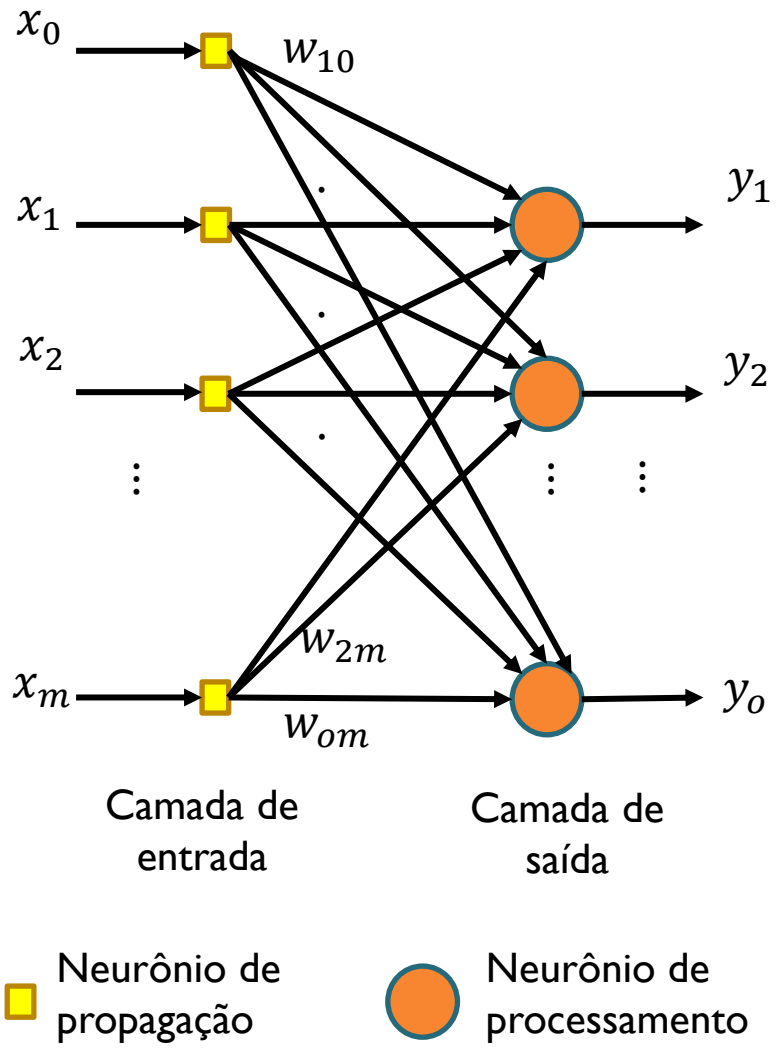
- *Feedforward*
  - Sinal se propaga sempre adiante, da entrada para a saída e nunca o contrário.

Redes *Feedforward* de Uma Única Camada

- Os pesos podem ser escritos usando uma notação matricial.

$$\mathbf{W} = \begin{bmatrix} w_{10} & w_{11} & \cdots & w_{1m} \\ \vdots & \vdots & \ddots & \vdots \\ w_{o0} & w_{o1} & \cdots & w_{om} \end{bmatrix}$$

Lembre-se:
- Neurônios são do tipo genérico.
- X_0 é fixo como $+1$



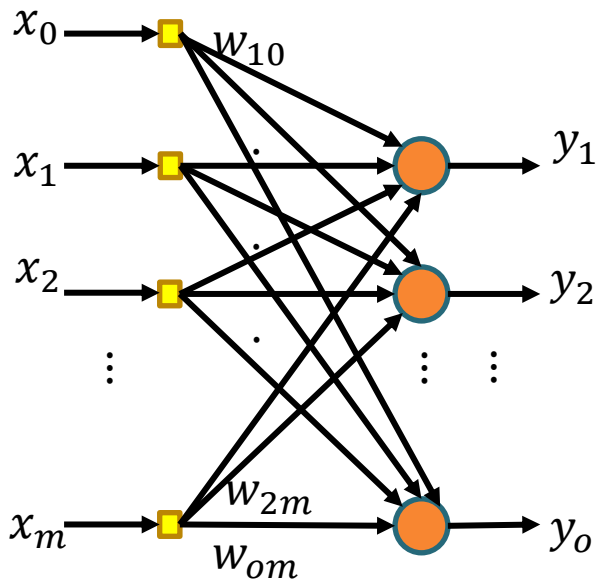
Redes *Feedforward* de Uma Única Camada

- Saída de cada neurônio *i* é dada por:

$$y_i = f(\mathbf{w}_i \mathbf{x}) = f\left(\sum_j w_{ij} x_j\right), j = 0, \ldots, m$$

- Saída da rede toda $\mathbf{y}$ é dada por:

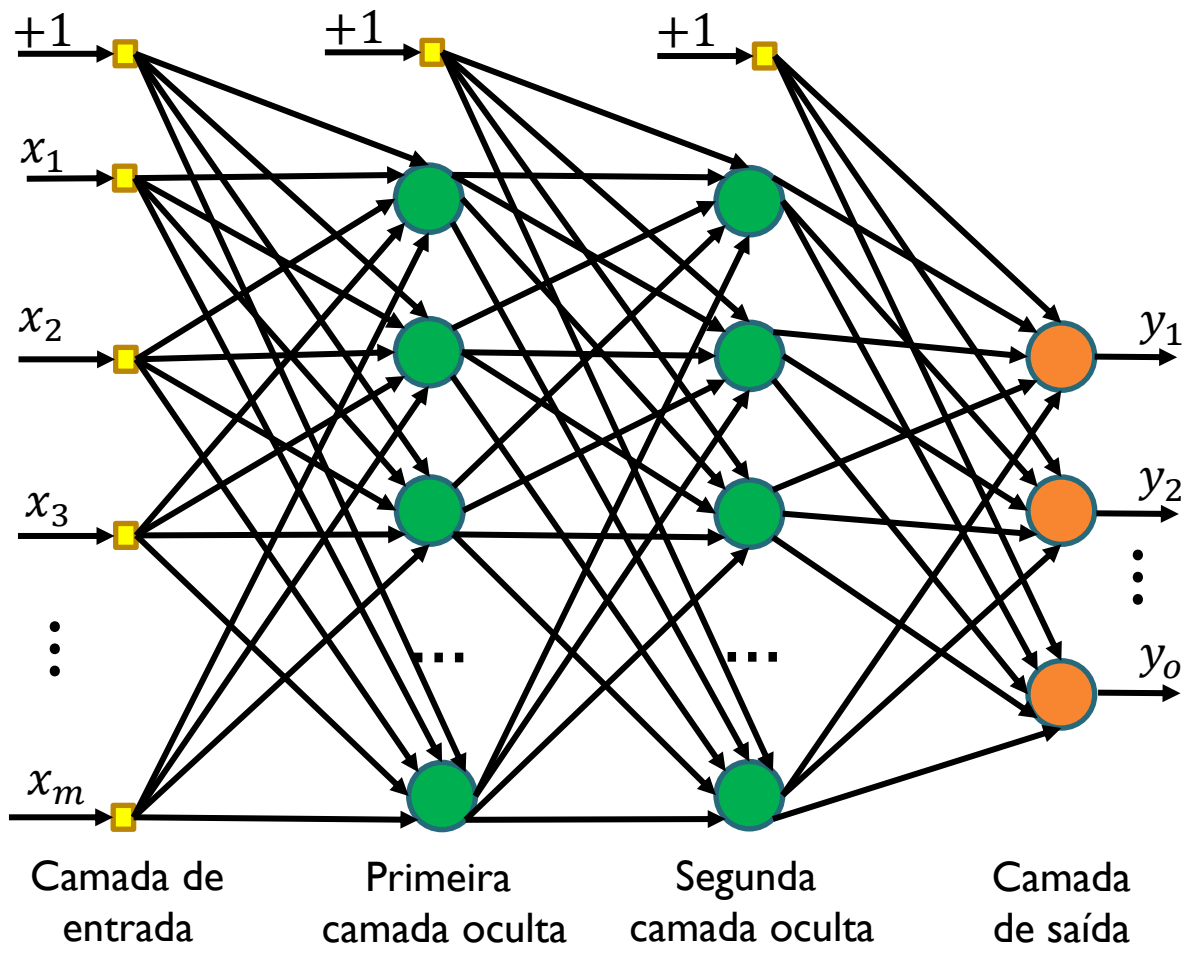
$$\mathbf{y} = f(\mathbf{W}.\mathbf{x})$$



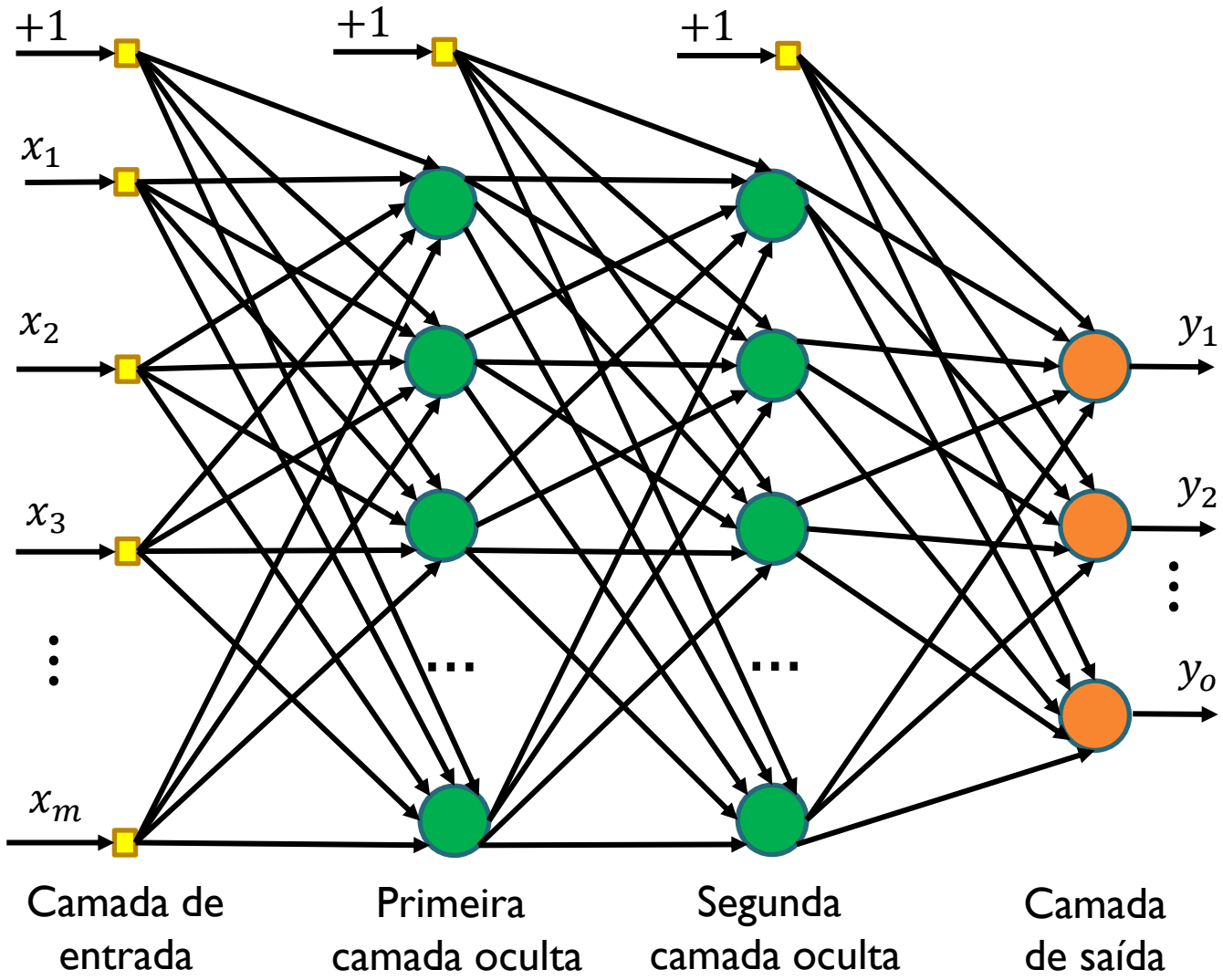
$$\mathbf{W} = \begin{bmatrix} w_{10} & w_{11} & \cdots & w_{1m} \\ \vdots & \vdots & \ddots & \vdots \\ w_{o0} & w_{o1} & \cdots & w_{om} \end{bmatrix}$$

Redes *Feedforward* de Múltiplas Camadas

- Diferem das redes de única camada pela presença de uma ou mais camadas *intermediárias* ou *ocultas*.
- Capacidade de processamento não linear.
- Saída de cada camada é usada como entrada da próxima camada.



Redes *Feedforward* de Múltiplas Camadas



Redes *Feedforward* de Múltiplas Camadas

- Nesse tipo de rede temos uma matriz de peso para cada camada.
  - Notação: $\mathbf{W}^k$ com k indicando a camada.
  - Considerando a figura anterior:
    - $\mathbf{W}^1$ indica a matriz de conexões entre a camada de entrada e a primeira camada oculta.
    - $\mathbf{W}^2$ indica a matriz de conexões entre a primeira camada oculta e a segunda camada oculta.
    - $\mathbf{W}^3$ indica a matriz de conexões entre a segunda camada oculta e a camada de saída.

Redes *Feedforward* de Múltiplas Camadas

- O sinal se propaga camada por camada, da entrada para a saída, portanto a saída da nossa rede em notação de matriz será:

$$\mathbf{y} = \mathbf{f}^3\left(\mathbf{W}^3\mathbf{f}^2\left(\mathbf{W}^2\mathbf{f}^1\left(\mathbf{W}^1\mathbf{x}\right)\right)\right)$$

- $\mathbf{f}^k$ é o vetor de funções de ativação da camada k
  - Nós de camadas diferentes podem ter diferentes funções de ativação.
- $\mathbf{W}^k$ é a matriz de pesos da camada k
- $\mathbf{x}$ é o vetor de entrada.

Redes *Feedforward* de Múltiplas Camadas

- Se os nós das camadas intermediárias tiverem função de ativação linear não há sentido em adicionar mais camadas na rede, pois supondo, por exemplo, $f(x) = x$, a saída seria:

$$\mathbf{y} = \mathbf{f}^3\left(\mathbf{W}^3\mathbf{W}^2\mathbf{W}^1\mathbf{x}\right)$$

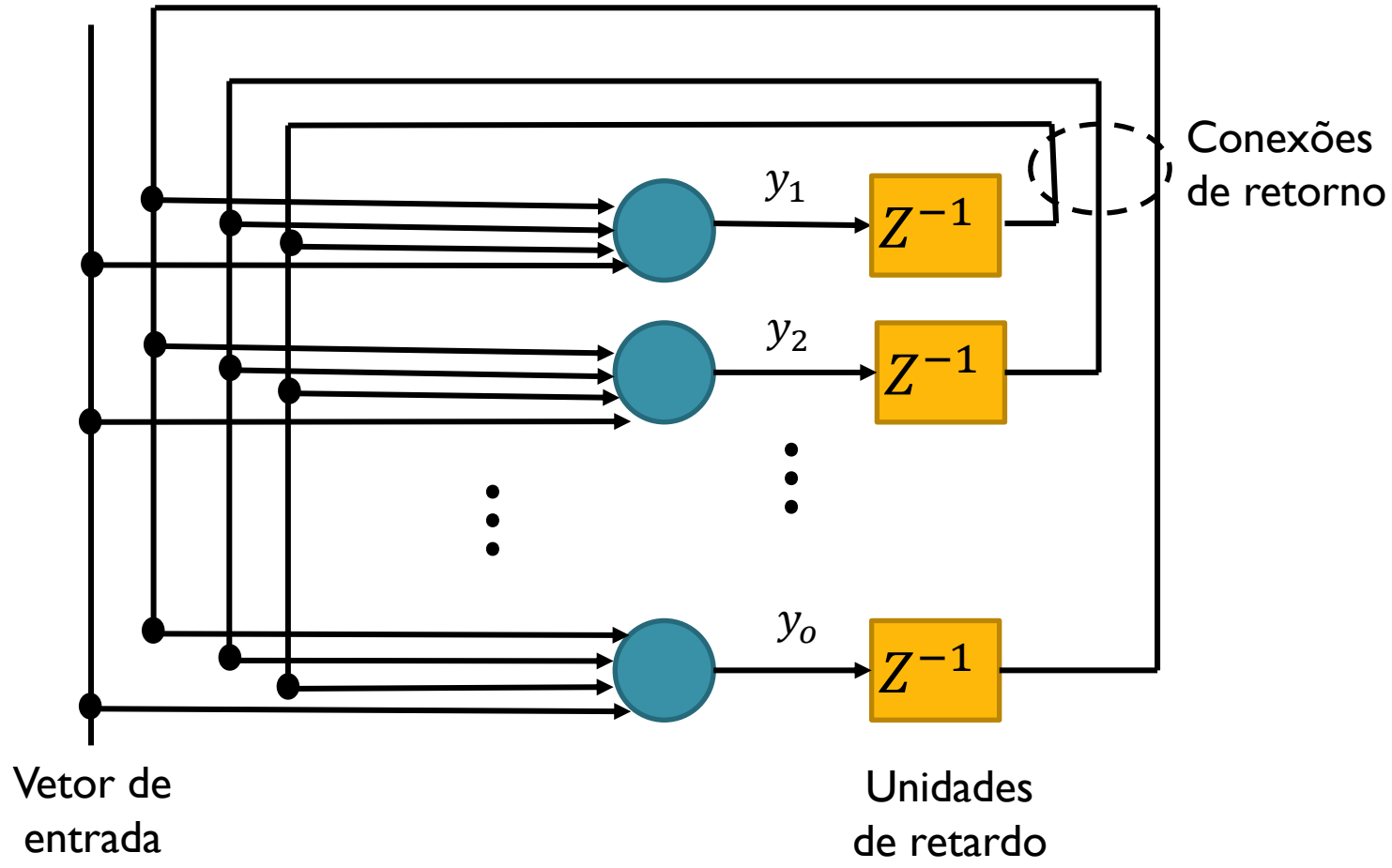
  a qual pode ser reduzida para:

$$\mathbf{y} = \mathbf{f}^3\left(\mathbf{W}\mathbf{x}\right) \text{ onde } \mathbf{W} = \mathbf{W}^3\mathbf{W}^2\mathbf{W}^1$$

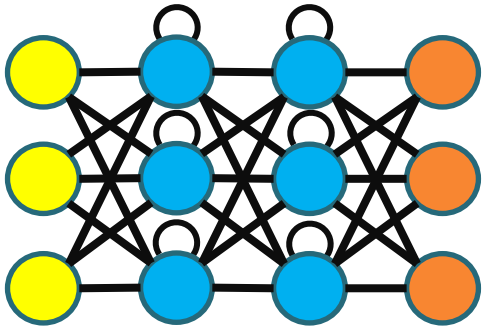
- Portanto, para aumentar a capacidade computacional de uma rede de múltiplas camadas adicionando mais camadas, funções de ativação não-lineares devem ser usadas nas camadas ocultas.

Redes Recorrentes

- Tem ao menos um *loop* recorrente (ou de retorno).
- Pode consistir de uma única camada de neurônios com cada neurônio enviando sua saída de volta para a entrada de outros neurônios.



Redes Recorrentes

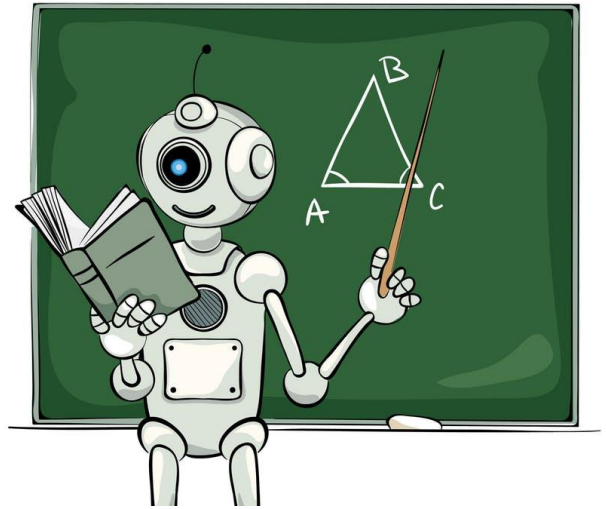


- Laços de realimentação:
  - Impacto profundo na capacidade de aprendizagem da rede e seu desempenho.
- Unidades de retardo:
  - Resulta em comportamento dinâmico não-linear, assumindo que a rede tenha unidades não lineares:

$$y_i(t) = f\left(\mathbf{w}_i.\mathbf{x}(t) + \mathbf{v}_i\mathbf{y}(t-1)\right)$$

Redes Neurais Artificiais

ABORDAGENS DE APRENDIZADO



Abordagens de Aprendizado

Contexto biológico

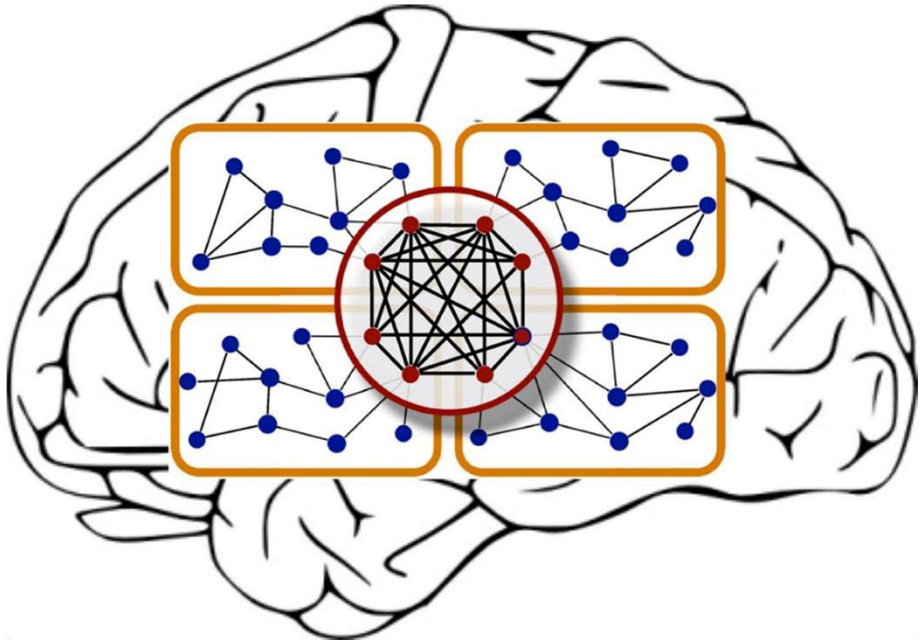
- Acredita-se que o aprendizado envolve a adaptação das forças sinápticas ao estímulo do ambiente.
- Porém a maneira como o aprendizado acontece ainda não é muito clara.

Contexto Computacional

- Aprendizado (ou treinamento) equivale a adaptar (ajustar) os "parâmetros livres" da rede através de um processo de apresentação de estímulos do ambiente.
  - Parâmetros livres:
    - Pesos de conexão de neurônios individuais.
    - E em redes mais sofisticadas:
      - Arquitetura da Rede.
      - Funções de Ativação de Neurônios Individuais.

Abordagens de Aprendizado

- Basicamente feito em dois passos:
  - Apresentação de padrões de entrada para a rede.
  - Adaptação dos parâmetros livres da rede para produzir os padrões de resposta desejados para os dados de entrada.

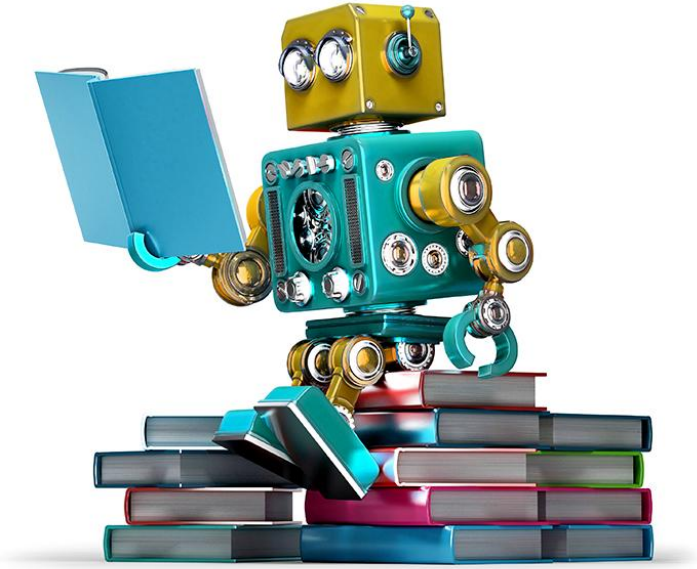


Abordagens de Aprendizado

- Na maioria das redes os pesos são ajustados por uma regra ou algoritmo de aprendizado.
- Depois a rede é aplicada a um novo conjunto de dados.
- Portanto temos duas etapas:
  1. Treinamento da Rede.
  2. Aplicação da Rede.

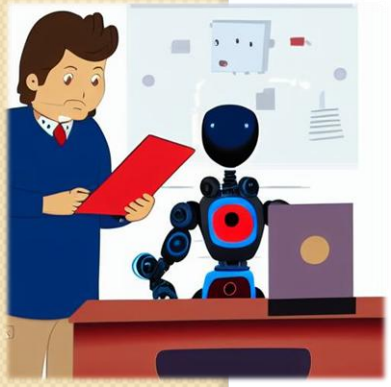
Abordagens de Aprendizado

- Com algoritmos de aprendizado padrão a rede aprende através de um processo iterativo de ajuste de pesos.
- O tipo de aprendizado depende da maneira como os pesos são ajustados:
  1. Aprendizado Supervisionado
  2. Aprendizado Não Supervisionado
  3. Aprendizado por Reforço



Aprendizado Supervisionado

- Caracteriza-se pela presença de um *supervisor* ou *professor.*
- Este conhecimento é representado na forma de amostras de entrada-saída.
- Em geral é usado quando a classe de uma amostra é conhecida a priori.
- Os parâmetros livres da rede são ajustados através da combinação da entrada e do sinal de erro.
  - Sinal de erro é a diferença entre a saída desejada e a saída produzida pela rede.



Exemplo: Problema de Classificação

200 e-mails

Exemplo adaptado de:
https://www.youtube.com/watch?v=IpGxLWOIZy4

Detecção de SPAM

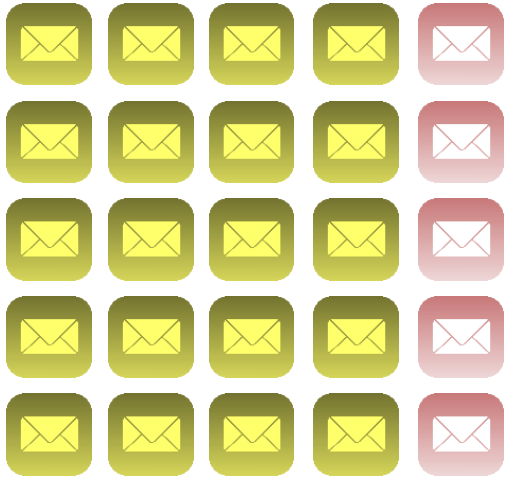
SPAM

Não-SPAM

Classificamos 50 como SPAM
e 150 como Não-SPAM.

Extraindo Atributos

- Quais deste e-mails tem a palavra **"réplica"**? 

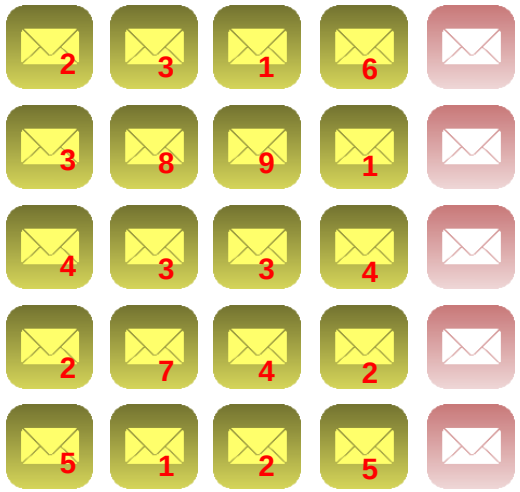


SPAM

Não-SPAM

Extraindo Atributos

- Quais deste e-mails tem a palavra **"réplica"**?
- Quantas vezes essa palavra aparece em cada e-mail?



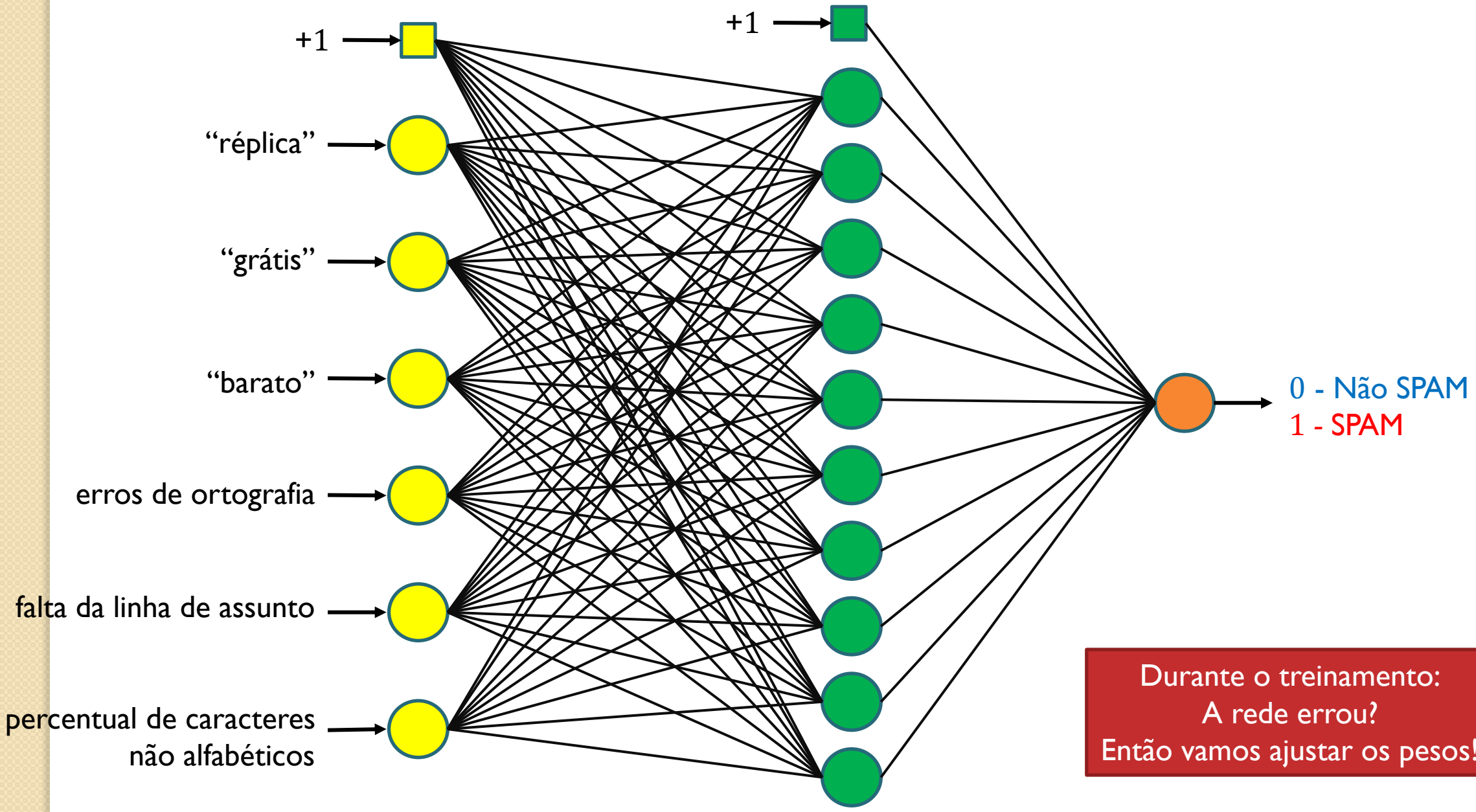
SPAM

Não-SPAM

Treinando uma Rede Neural

- Podemos acrescentar outros atributos.
- Tais atributos podem servir como entrada para uma rede neural.
- A saída da rede pode ser um neurônio que indique se um e-mail é:
  - SPAM (1)
  - Não SPAM (0)
- Uma vez treinada, a rede pode classificar novos e-mails, ainda não vistos.

Atributo	Tipo
"réplica"	inteiro
"grátis"	inteiro
"barato"	inteiro
erros de ortografia	inteiro
falta da linha de assunto	binário
percentual de caracteres não alfabéticos	real
etc.	



Durante o treinamento:
A rede errou?
Então vamos ajustar os pesos!

Normalização

- Pode melhorar o desempenho e estabilidade do treinamento.
  - Usado quando os dados possuem escalas diferentes.
- z-score
  - Método mais utilizado.
  - Centraliza os dados para que tenham média 0
  - Escala os dados para que tenham desvio padrão 1

Não Normalizado:

5,10	3,50	1,40	0,20
4,90	3,00	1,40	0,20
4,70	3,20	1,30	0,20
4,60	3,10	1,50	0,20
5,00	3,60	1,40	0,20
5,40	3,90	1,70	0,40
4,60	3,40	1,40	0,30
5,00	3,40	1,50	0,20
4,40	2,90	1,40	0,20
4,90	3,10	1,50	0,10

Normalizado:

0,82	0,62	-0,46	-0,25
0,14	-1,01	-0,46	-0,25
-0,55	-0,36	-1,39	-0,25
-0,89	-0,68	0,46	-0,25
0,48	0,94	-0,46	-0,25
1,85	1,92	2,31	2,28
-0,89	0,29	-0,46	1,01
0,48	0,29	0,46	-0,25
-1,58	-1,33	-0,46	-0,25
0,14	-0,68	0,46	-1,52

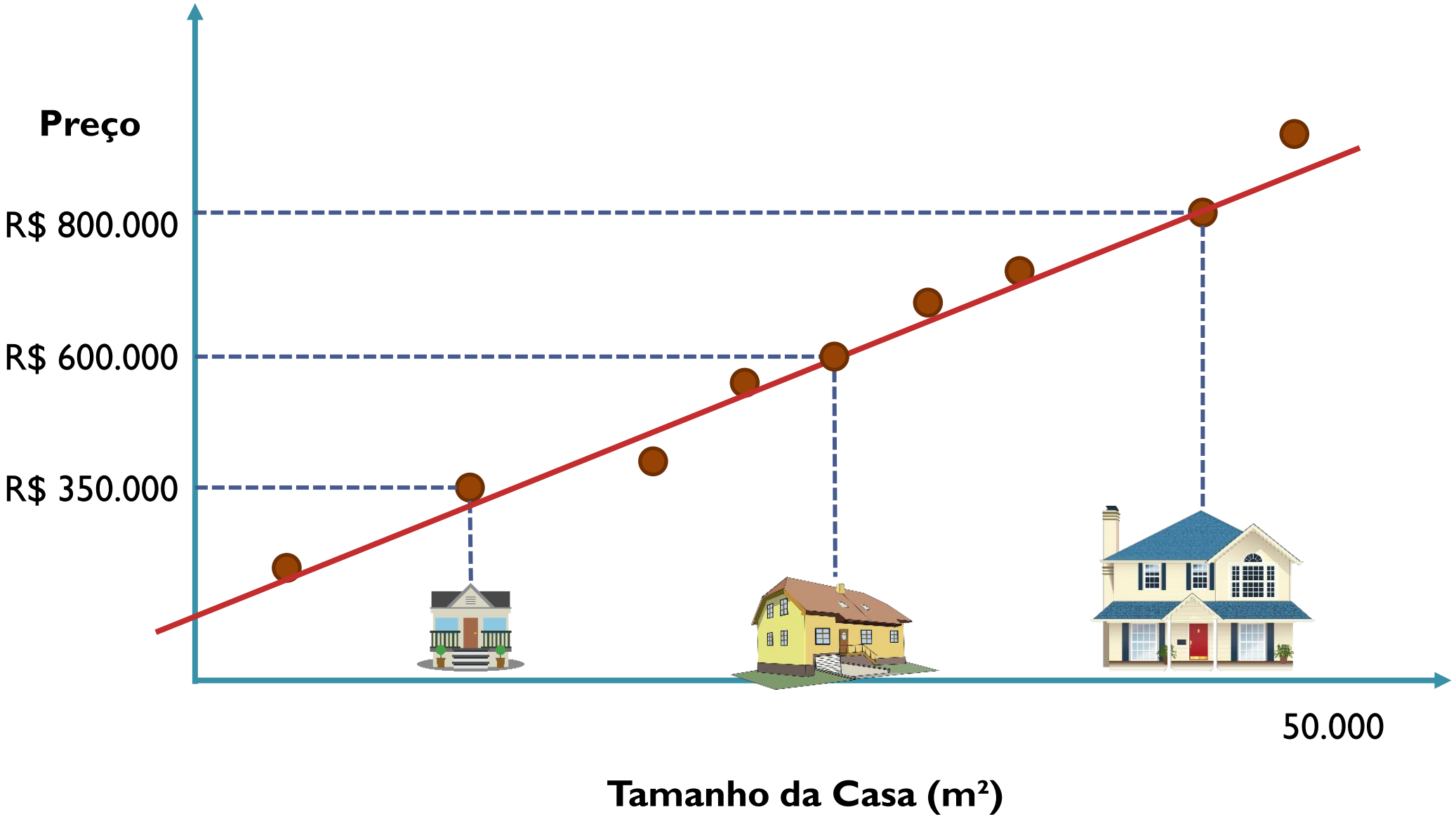
Outro Exemplo: Problema de Regressão
Como calcular o preço de uma casa?

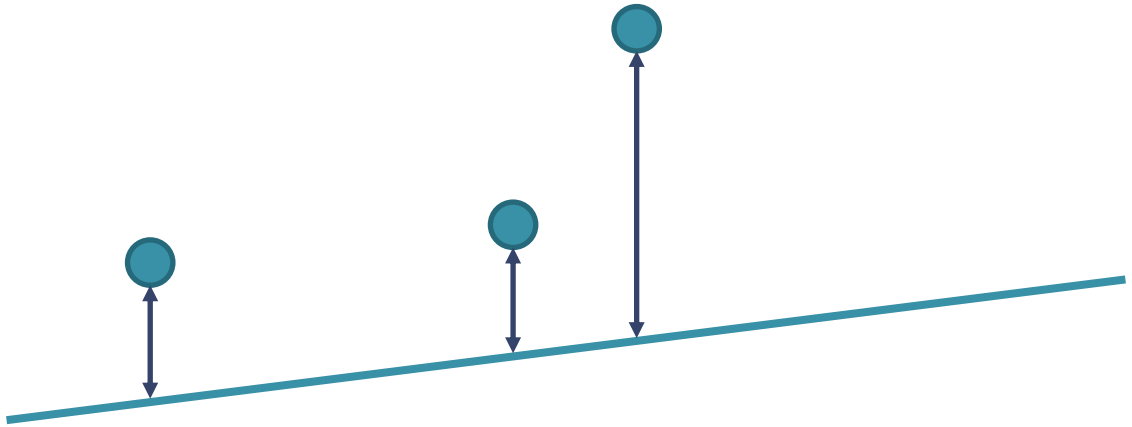
R$ 350.000

?

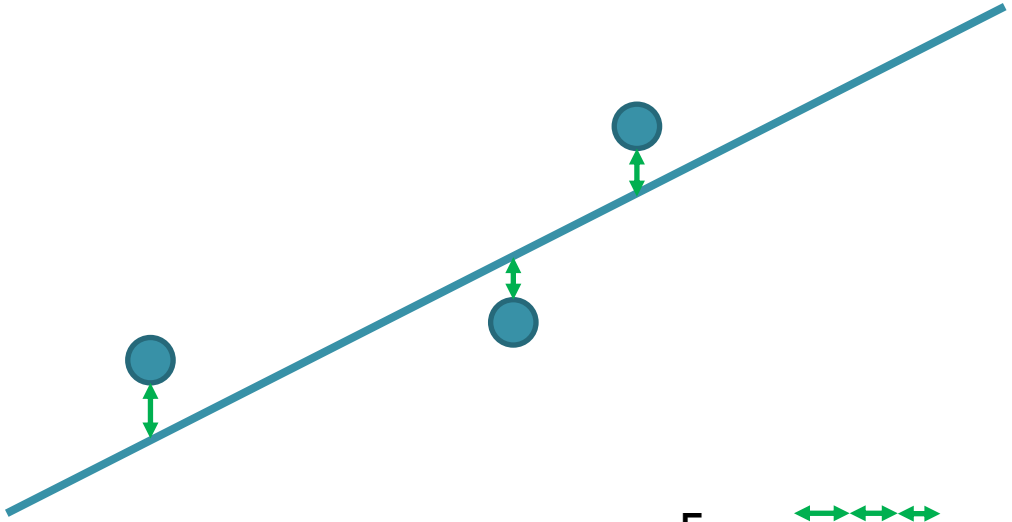
R$ 800.000

Exemplo adaptado de:
https://www.youtube.com/watch?v=IpGxLWOIZy4


Preço
R$ 800.000
R$ 600.000
R$ 350.000
50.000
Tamanho da Casa (m²)



Erro: 

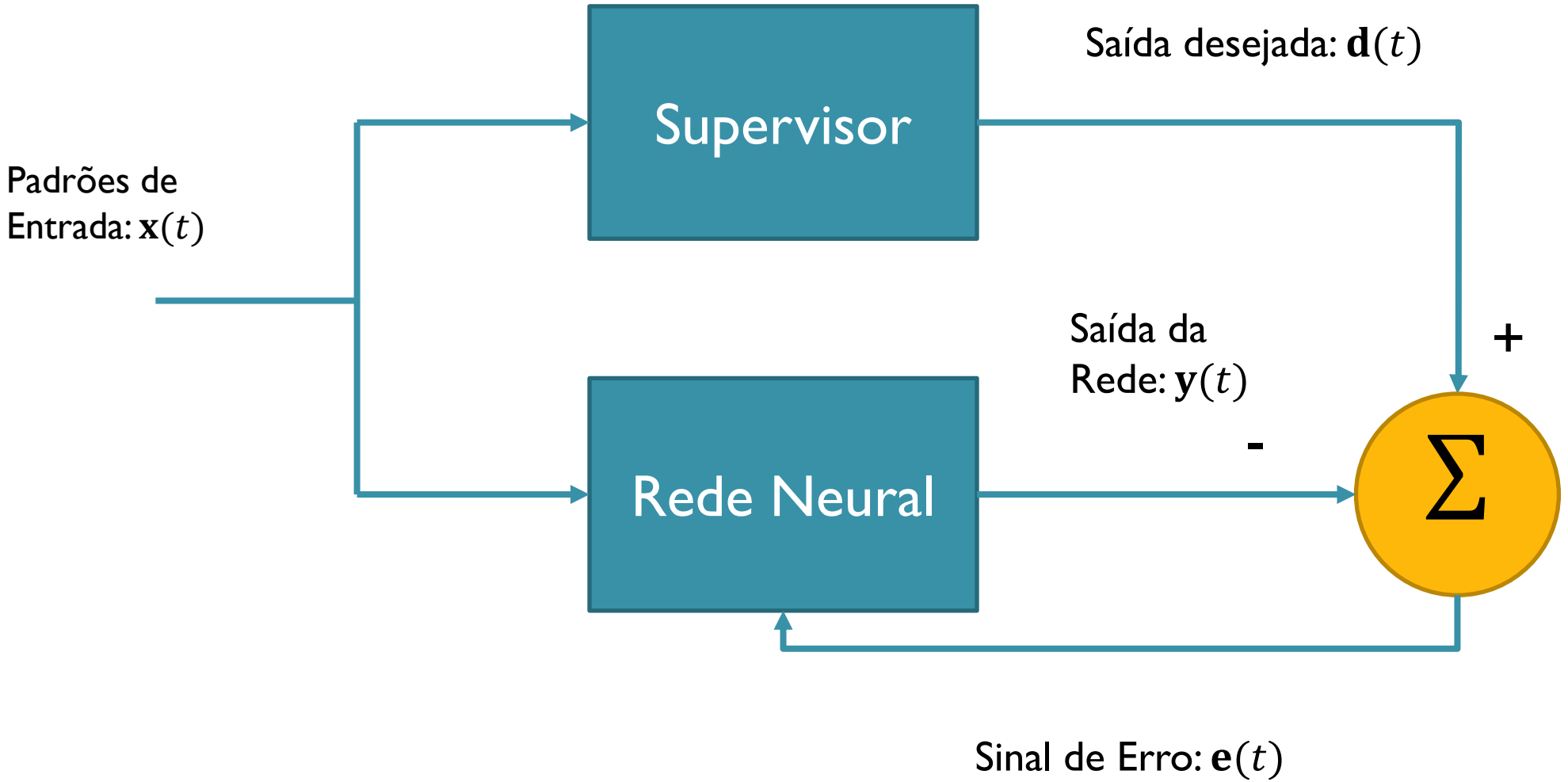


Erro: 

Treinando uma Rede Neural

- Novamente, podemos acrescentar mais atributos:
  - Estado de conservação.
  - Qualidade dos acabamentos.
  - Distância até o centro da cidade.
  - Taxa de criminalidade na região.
  - Qualidade das escolas e hospitais na região.
- A saída da rede pode ser um neurônio que indique o preço estimado para a casa.
- Uma vez treinada, a rede pode prever o preço de outras casas, ainda não vistas.

Aprendizado Supervisionado

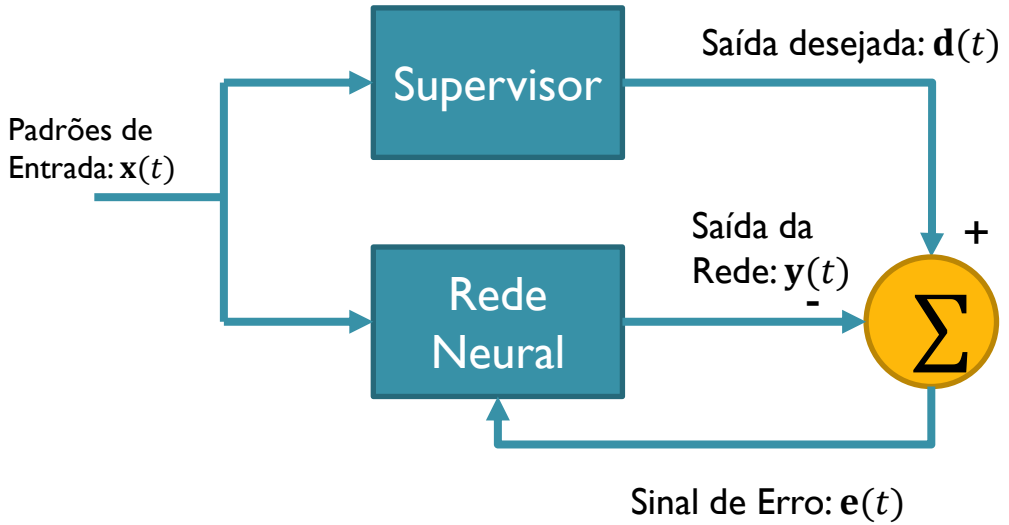


Aprendizado Supervisionado

- Seja j o neurônio de saída de uma rede *feedforward*.
- $\mathbf{x}(t)$ é a entrada desse neurônio.
  - Produzida eventualmente por uma camada intermediária.
- t é o instante de tempo nesse processo iterativo.
- $y_j(t)$ é a saída produzida pelo neurônio j
- A saída desejada é dada por $d_j(t)$

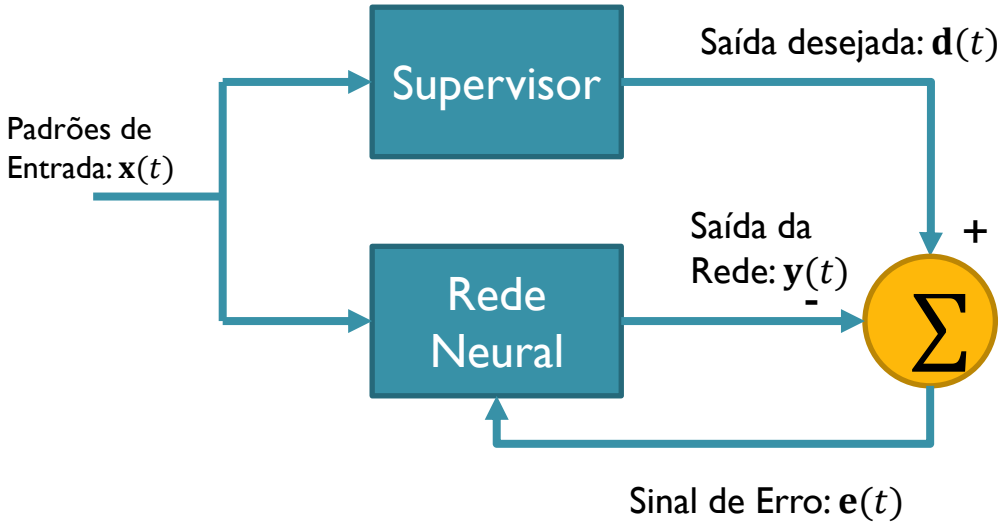
- O sinal de erro $e_j(t)$ é calculado por:

$$e_j(t) = d_j(t) - y_j(t)$$



Aprendizado Supervisionado

- Sinal de Erro:
  - Utilizado como mecanismo de controle para ajustes corretivos no peso do neurônio j
- Objetivo:
  - Tornar a saída da rede mais similar à saída desejada a cada passo no tempo.



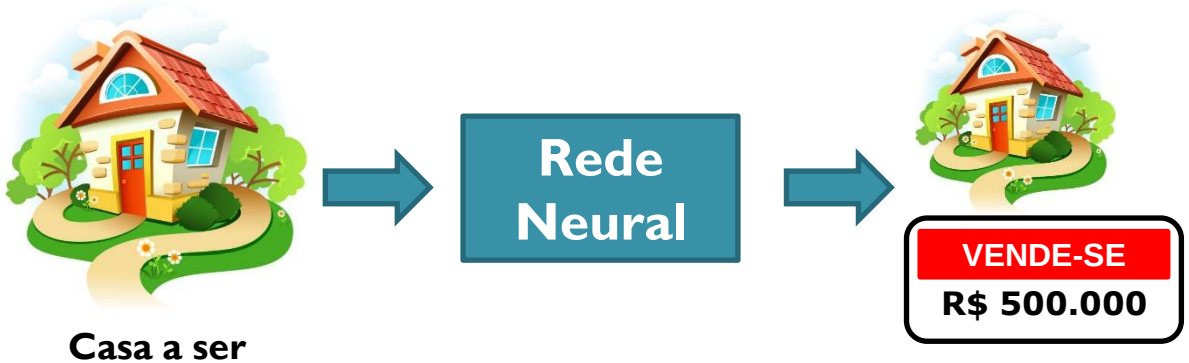
Função de Custo

- Também chamada *índice de performance*, *função de erro*, ou *função objetivo*.
- Mede o desempenho de um modelo de aprendizado de máquina para um determinado conjunto de dados.
- Tipicamente deve ser minimizada.

- Exemplos:
  - Erro médio absoluto:
    - $\Im(t) = {}^{1}/_{m} \sum_{i=1}^{m} |e_i(t)|$
  - Erro médio quadrático:
    - $\Im(t) = {}^{1}/_{2m} \sum_{i=1}^{m} e_i(t)^2$
    - m é a quantidade de exemplos de treinamento.

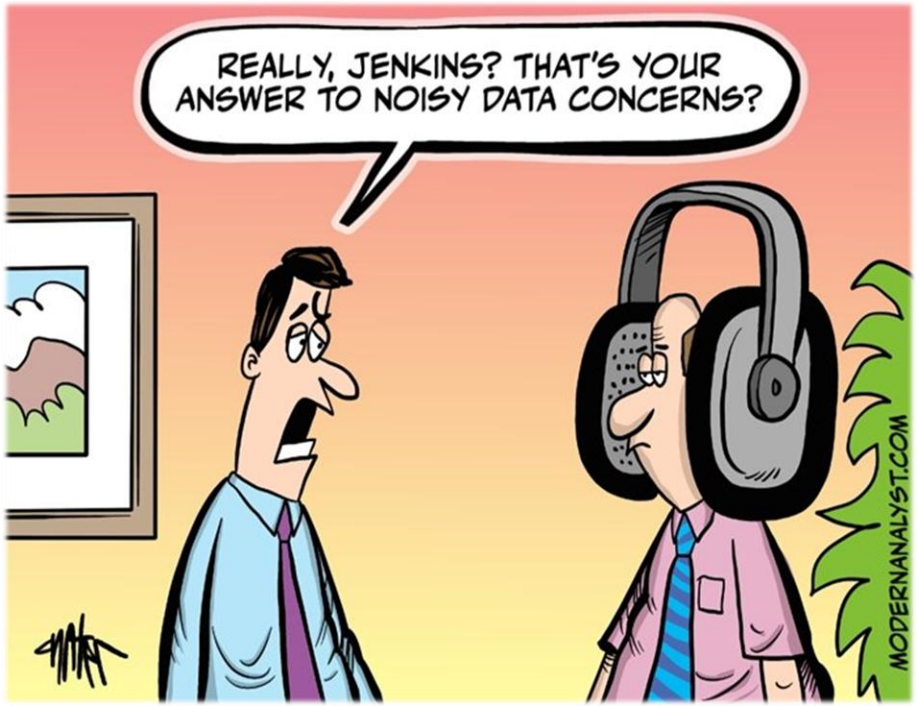
Capacidade de Generalização

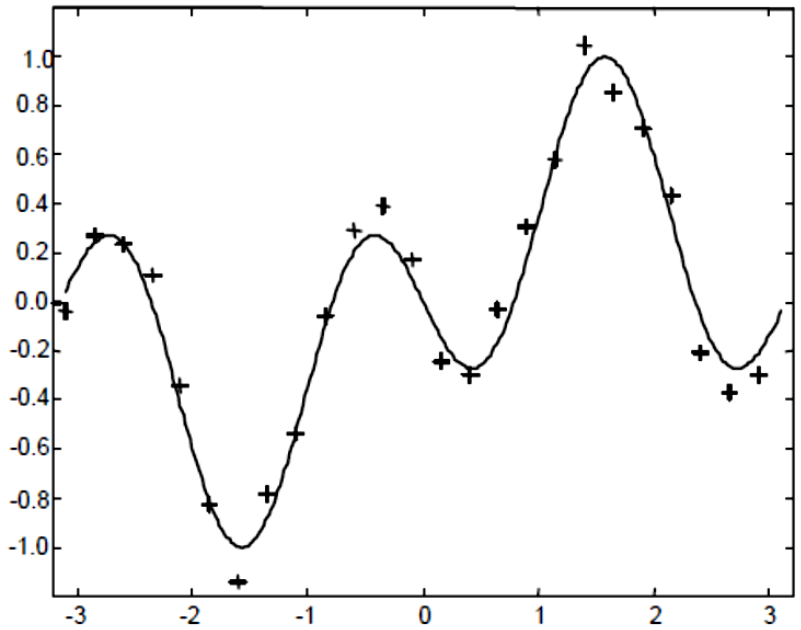
- Quando o processo de aprendizado supervisionado está completo, podemos apresentar uma nova amostra para a rede (ainda não vista) e a rede fornecerá uma saída com um certo grau de precisão.



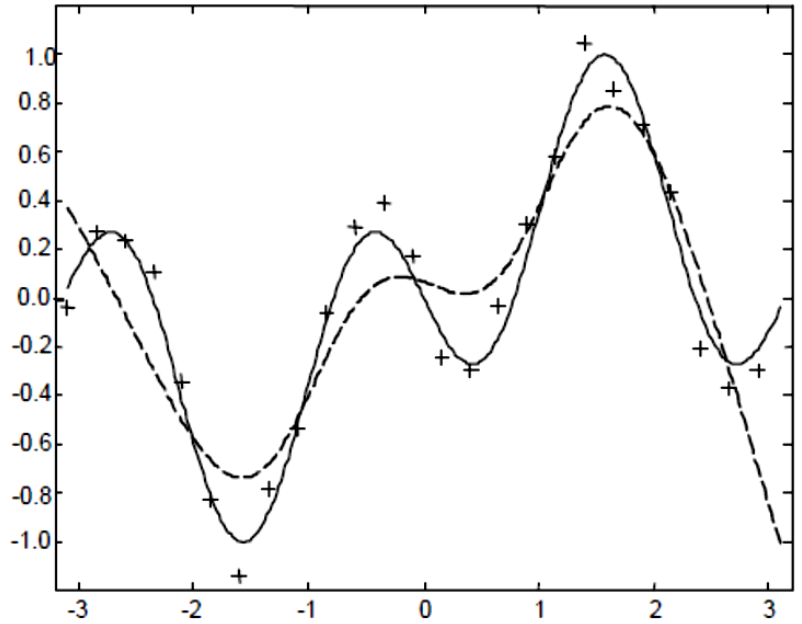
Capacidade de Generalização

- A maioria dos dados do mundo real contém ruído.
  - Dados irrelevantes ou com perturbações.
  - Na figura do próximo *slide*, vemos como o treinamento pode se comportar diante de dados ruidosos.
    - Dados que se desviam um pouco da função que deve ser aproximada.

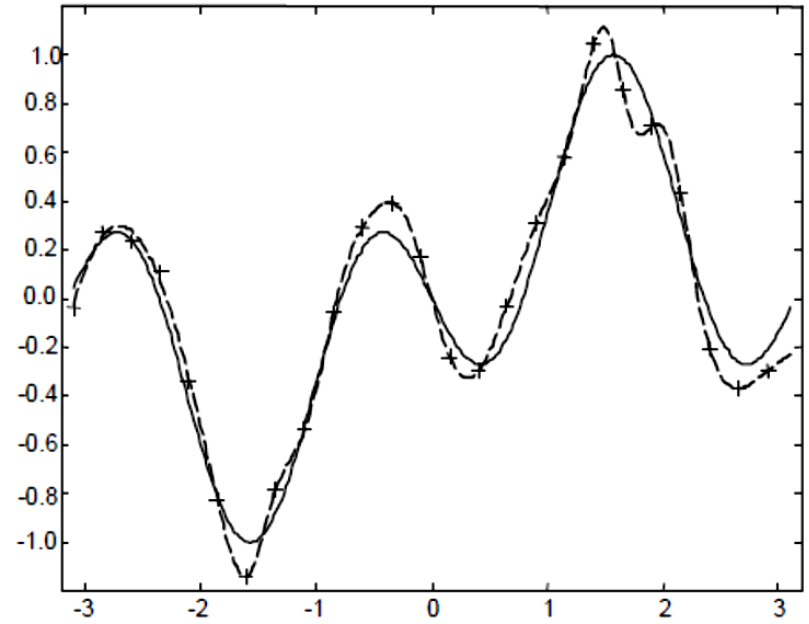




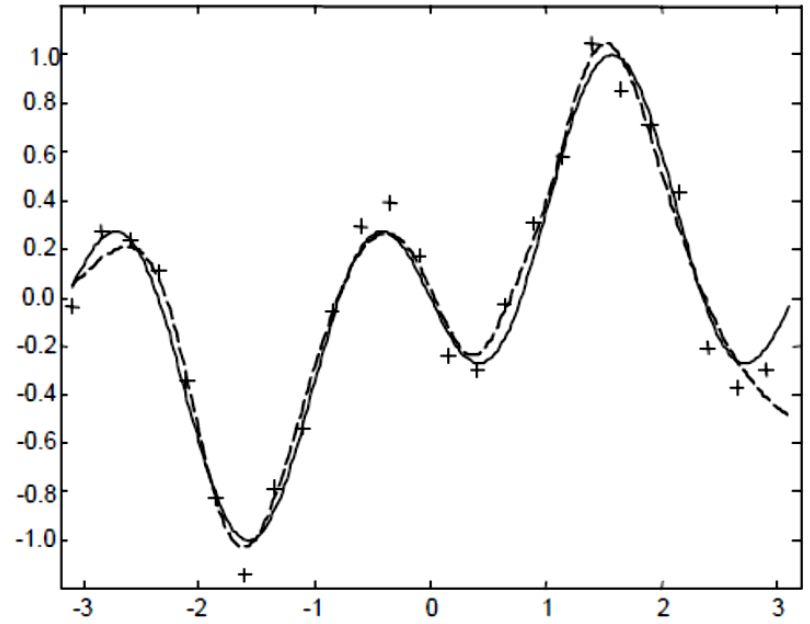
(a)



(b)



(c)



(d)

Função $\sin(x) \times cos(2x)$ com ruído distribuído uniformemente no intervalo $[-0{,}15; 0{,}15]$.

Legenda:
+ padrões de entrada;
— saída desejada;
- - - saída da rede.

(a) Padrões de treinamento com ruído e função original a ser aproximada (linha sólida).
(b) Processo de treinamento interrompido muito cedo, *underfitting* (linha tracejada).
(c) Muito treinamento, *overfitting* (linha tracejada).
(d) Melhor equilíbrio entre qualidade de aproximação e capacidade de generalização.

Capacidade de Generalização

- *Underfitting*:
  - Aproximação insatisfatória.
  - Ocorre quando o treinamento é insuficiente ou a arquitetura da rede não é adequada.
- *Overfitting*:
  - Aproximação perfeita demais para os dados ruidosos.
  - Resultados ótimos para amostras de treinamento, mas ruim para dados não vistos.
- O treinamento ideal é um meio termo entre precisão na aproximação e capacidade de generalização.

Capacidade de Generalização

- Do ponto de vista biológico:
  - O que é mais importante?
    - Decorar uma série de fatos específicos sobre o mundo?
    - Ou tentar extrair alguma regularidade desses fatos?
  - Exemplo:
    - Você quase morreu afogado no mar.
      - Lição aprendida:
        - "Ficar longe da água do mar."
      - Conclusão:
        - Você pode acabar se afogando numa piscina.

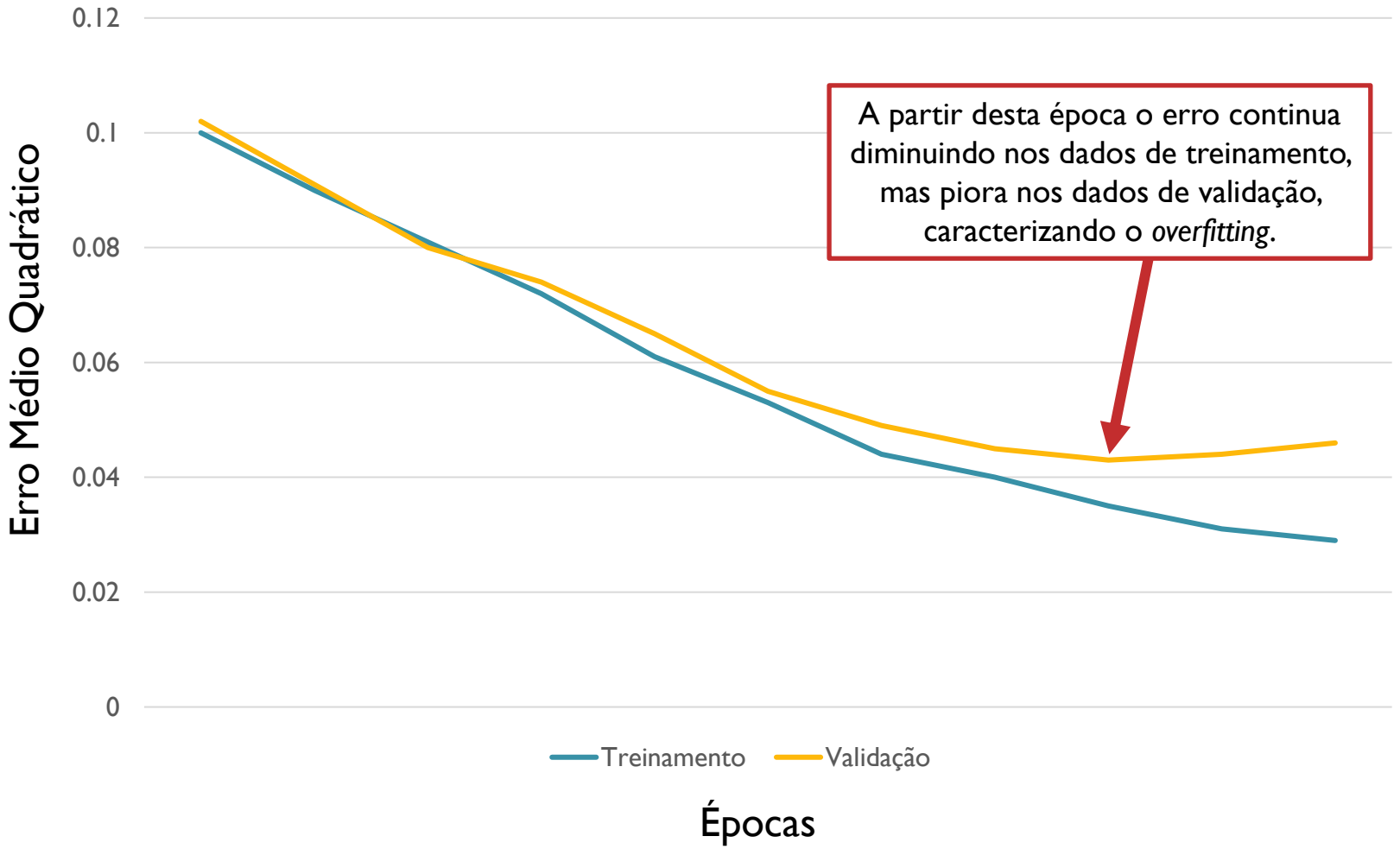
Lloyd's Baia Hotel – Vietri Sul Mare, Salerno, Itália por Fabricio Breve

Piscina no Armação Resort – Porto de Galinhas, Ipojuca, Pernambuco por Fabricio Breve

Reduzindo o *Overfitting*

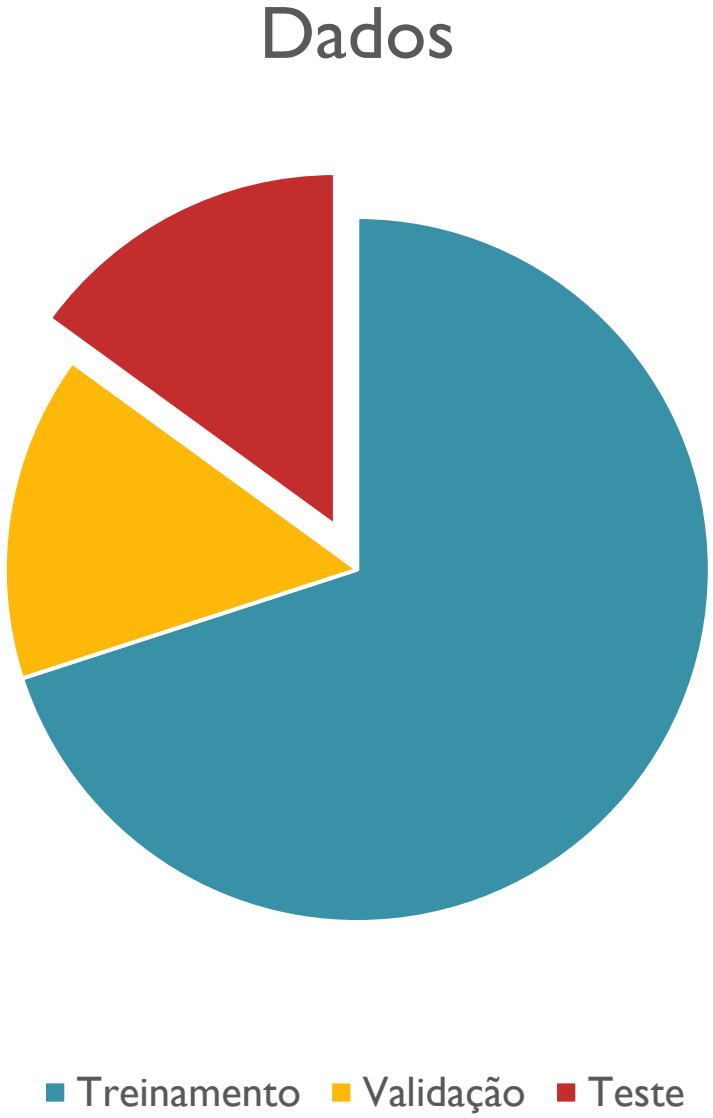
- Pode ser utilizado um subconjunto de validação para testar a capacidade de generalização.
  - Parando o treinamento quando a capacidade de generalização deixa de melhorar.

Overfitting



Uma Abordagem Comum

- Subconjunto de Treinamento:
  - Usados para treinar a rede, sinal de erro usado para ajustar pesos.
- Subconjunto de Validação:
  - Usado para testar a capacidade de generalização da rede a cada época, ou para testar diferentes arquiteturas/algoritmos de rede candidatas.
    - Não é usado para ajustar pesos.
- Subconjunto de Testes:
  - Usado apenas após o treinamento, para testar a capacidade da rede escolhida e treinada em classificar dados que ainda não foram vistos.



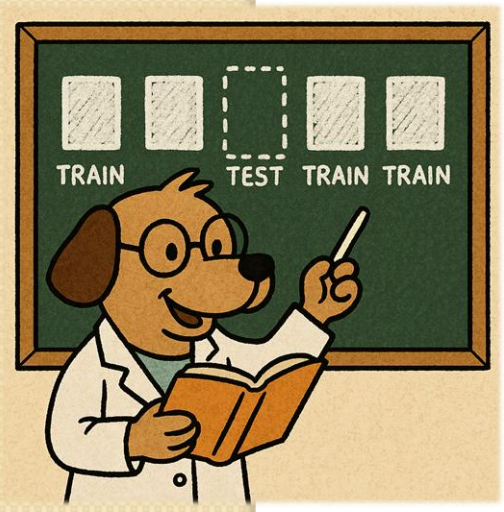

Once you have well-curated and prepared data, you split it up for use in training, testing, and validation.
It might require a ton of work to collect and label that much data, but it may be worth it.
TRAINING
TESTING
VALIDATION

Estimando o Desempenho da Rede

- O subconjunto de testes pode ser usado para estimar o desempenho que a rede teria com novos dados (não vistos), se fosse treinada com o conjunto de dados completo.
  - Os dados de testes não estão sendo usados no treinamento, portanto são dados que não foram vistos pela rede.
  - É uma estimativa pessimista, visto que parte dos dados não foi usada no treinamento.
  - Esta técnica é conhecida como *hold-out*.

Validação Cruzada (*Cross Validation*)

- Divide-se os elementos do conjunto de dados aleatoriamente em *k* grupos (*folds*).
  - Um grupo é usado para teste e o restante é usado para treinamento.
    - Se for usado um subconjunto de validação, ele será um subconjunto extraído dos grupos separados para o treinamento.
    - Este processo é repetido até que todos os grupos tenham sido usados para teste
      - Ou seja, o processo é feito *k* vezes, cada vez com um grupo diferente para teste.
      - Toma-se a média dos erros em cada uma das repetições.

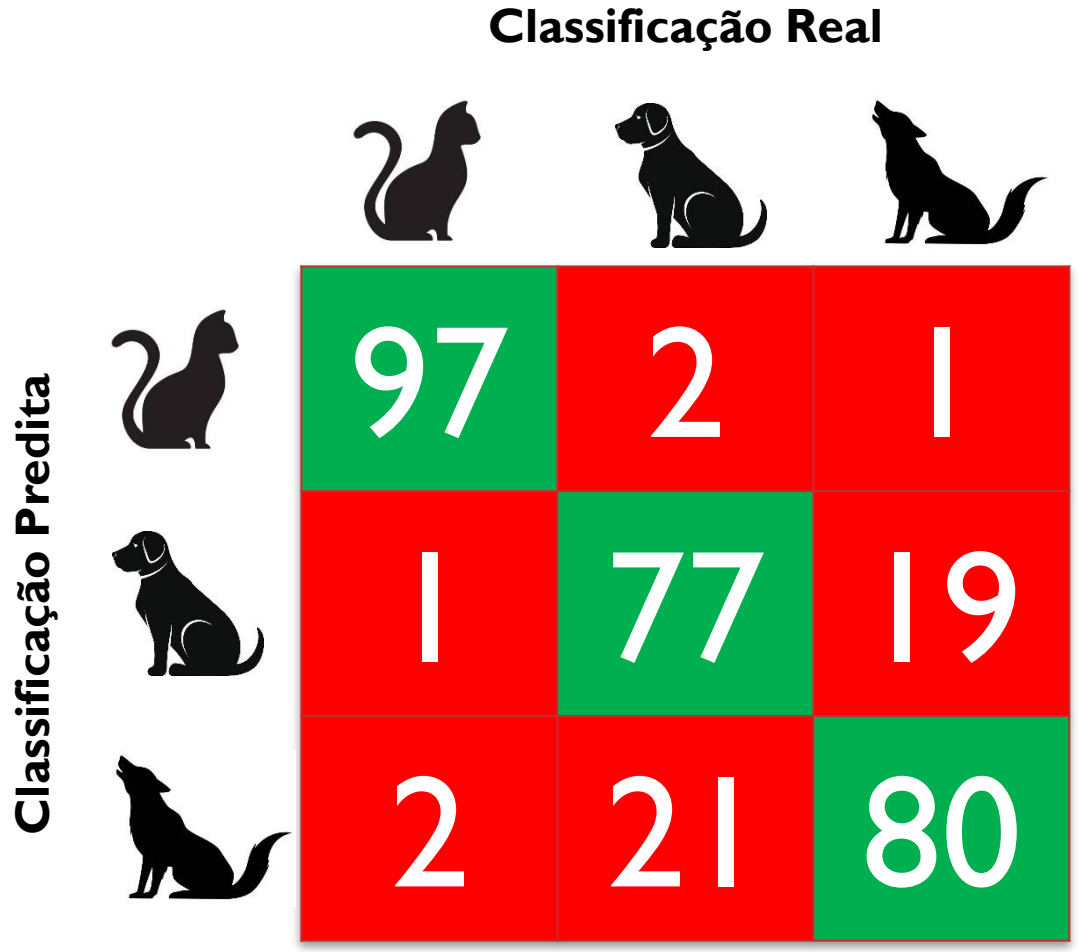


Validação Cruzada (*Cross Validation*)

- **Vantagem**: Fornece uma estimativa melhor do desempenho da rede.
- **Desvantagem**: É mais demorada.
- Dispondo de tempo suficiente, podemos usar *k* = a quantidade de elementos no conjunto de dados.
  - Cada grupo terá um único elemento.
  - Treina-se o modelo com todos os elementos, menos um que será usado para teste, repetindo *k* vezes.
  - *Leave-one-out Cross-Validation*

Matriz de Confusão

- Tabela que compara as previsões de um modelo com os valores reais, permitindo avaliar o desempenho do modelo.
- Fornece algumas informações além do erro médio.
  - Ex.: Permite avaliar se o modelo está confundindo algumas classes específicas.



Exemplo de Matriz de Confusão para um Problema com 300 elementos: 100 gatos, 100 cachorros e 100 lobos

Aprendizado Não Supervisionado

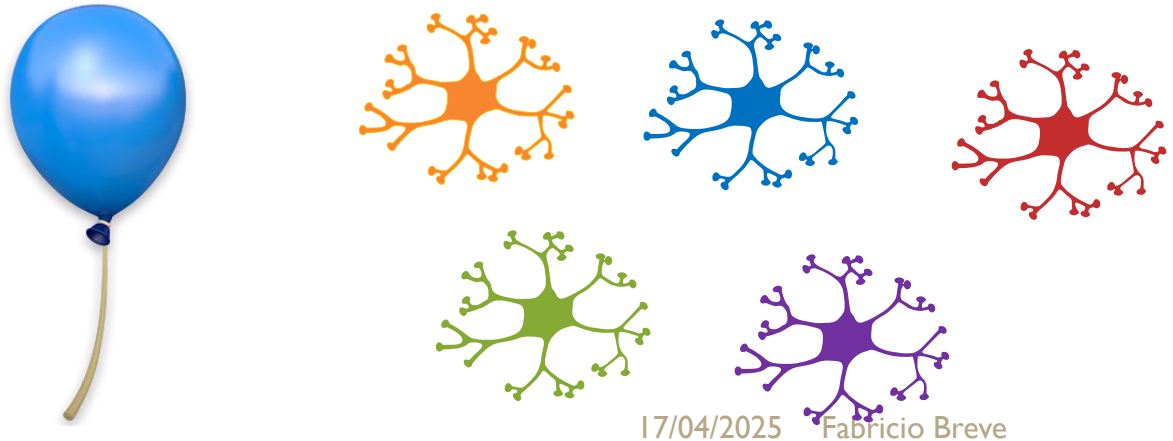
- Também chamado de aprendizado *auto-organizado*.
- Não há um supervisor para avaliar o desempenho da rede em relação aos dados de entrada.
- A idéia é: dado um conjunto de dados, o que pode você pode fazer com ele?
  - Exemplo: alguém te deu uma porção de balões.
    - Você pode separá-las por cor, formato ou qualquer outro atributo que você possa qualificá-los.

Aprendizado Não Supervisionado

- Não há informação de erro.
- A rede se adapta a regularidades estatísticas nos dados de entrada.
  - Desenvolve uma habilidade para criar representações internas para os dados de entrada, e portanto criar novas classes automaticamente.

Aprendizado Não Supervisionado

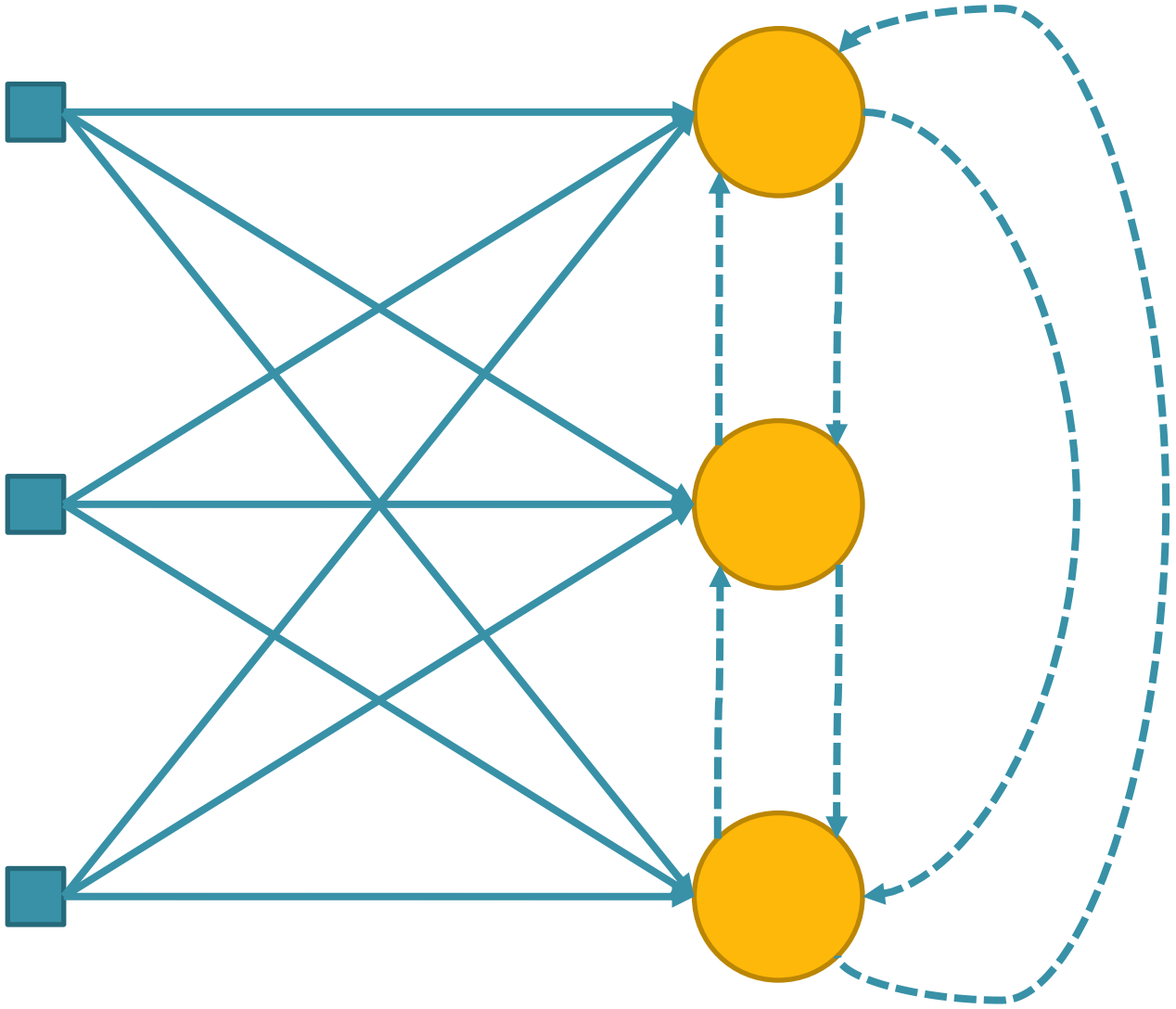
- Aprendizado competitivo:
  - O mais usado em algoritmos auto-organizáveis.
  - Neurônios de saída competem entre si para serem ativados.
    - Apenas um neurônio é ativado a cada iteração.
    - Torna o algoritmo apropriado para descobrir saliências estatísticas que possam ser usadas para classificar os dados de entrada.



Aprendizado Competitivo

- Neurônios se especializam em identificar um certo padrão de entrada.
  - Se tornam *extratores de características* ou *detectores de características* para diferentes classes de padrões de entrada.
- Na forma mais simples, tem uma única camada de saída totalmente conectada.
  - Incluindo conexões laterais entre neurônios.
    - Capaz de inibir ou estimular neurônios vizinhos.

Aprendizado Competitivo



Aprendizado Competitivo

- Para um neurônio j ser o vencedor, a distância entre seu vetor de peso $\mathbf{w}_i$ e um certo padrão de entrada $\mathbf{x}$ precisa ter a menor medida (entre todas os neurônios de saída), dada uma certa métrica.
  - em geral usa-se a Distância Euclidiana.

$$j = \text{argmin}_i \left\| \mathbf{x} - \mathbf{w}_j \right\| \ , \ \forall j$$

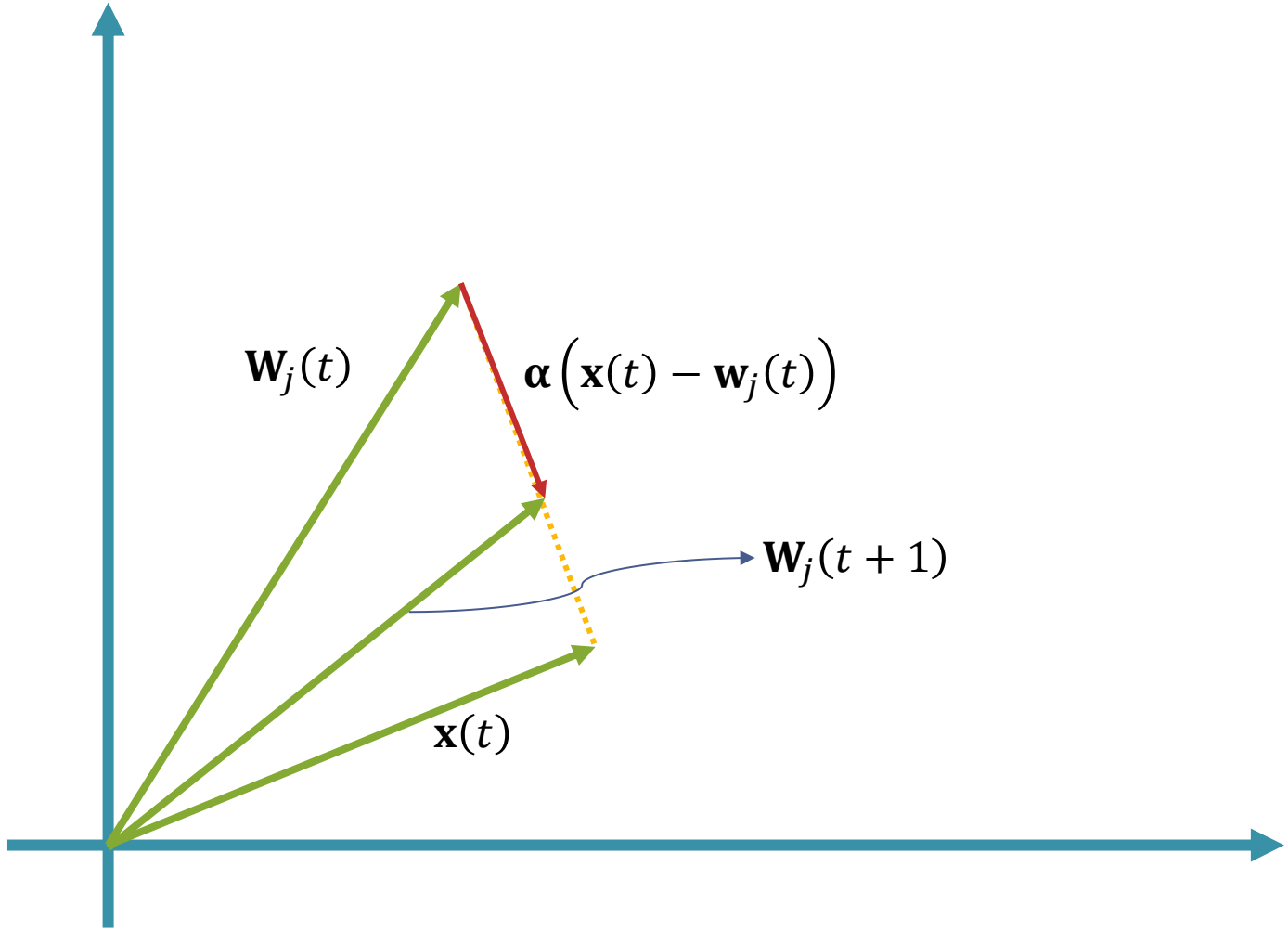
Aprendizado Competitivo

- Se um neurônio não responde a um determinado padrão de entrada, nenhum aprendizado ocorre nele.
- Porém, se ele ganha a competição, um ajuste $\Delta \mathbf{w}_j$ é aplicado ao vetor $\mathbf{w}_j$

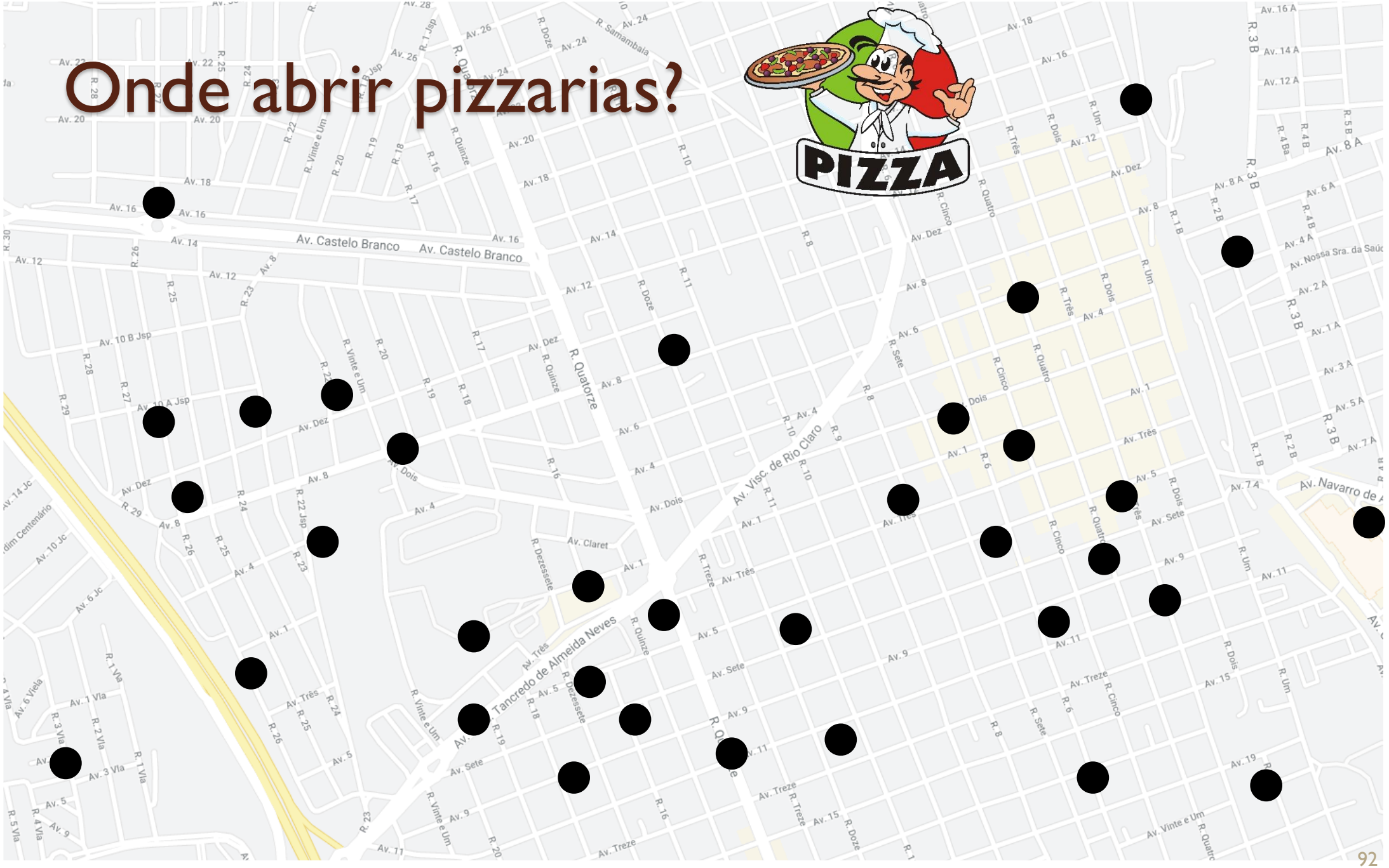
$$\Delta \mathbf{w}_j = \begin{cases} \alpha(\mathbf{x} - \mathbf{w}_j) & \text{se } j \text{ ganha a competição} \\ 0 & \text{se } j \text{ perde a competição} \end{cases}$$

onde α é uma *taxa de aprendizado* que controla o passo de $\mathbf{w}_j$ em direção ao vetor de entrada $\mathbf{x}$

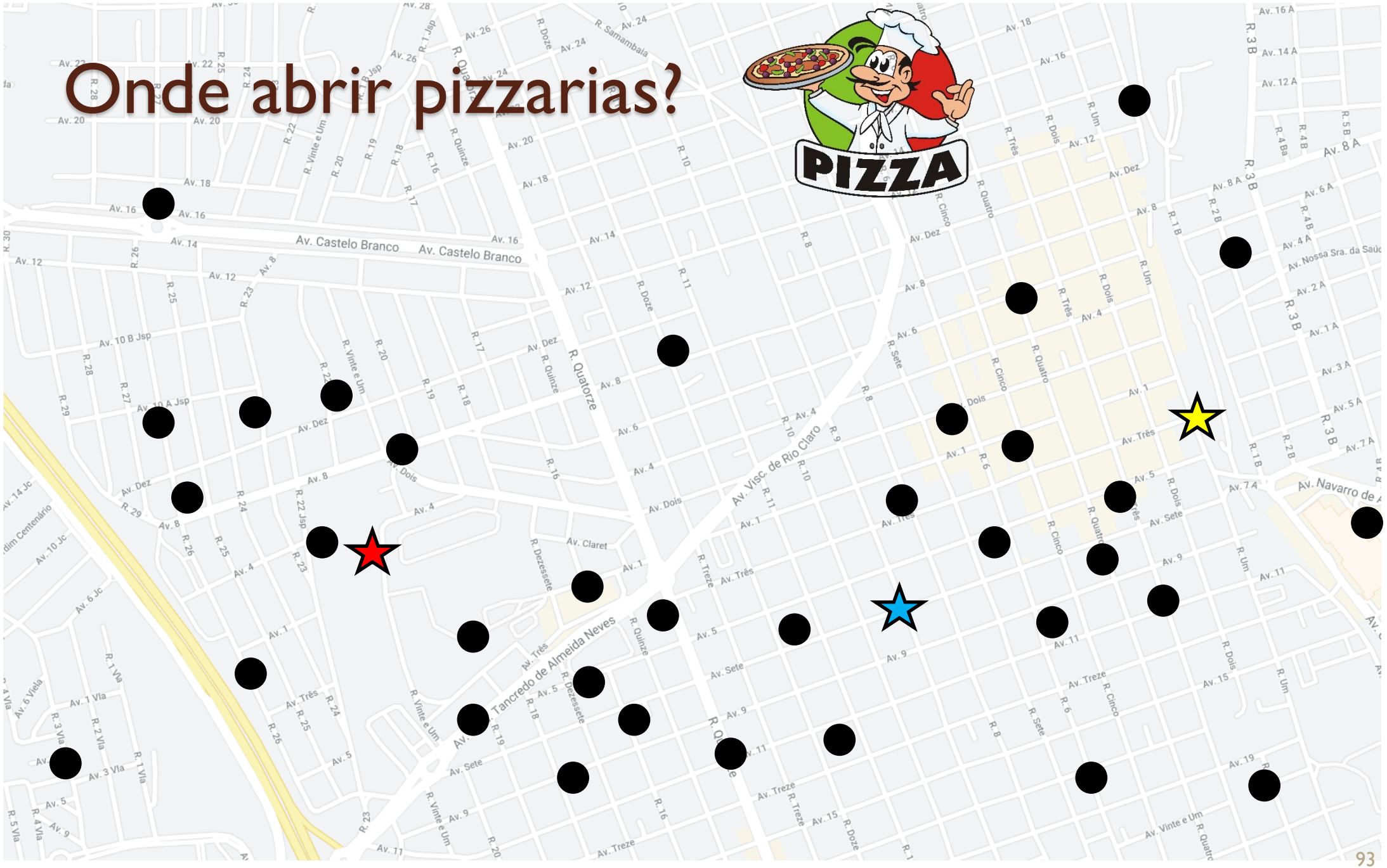
Aprendizado Competitivo



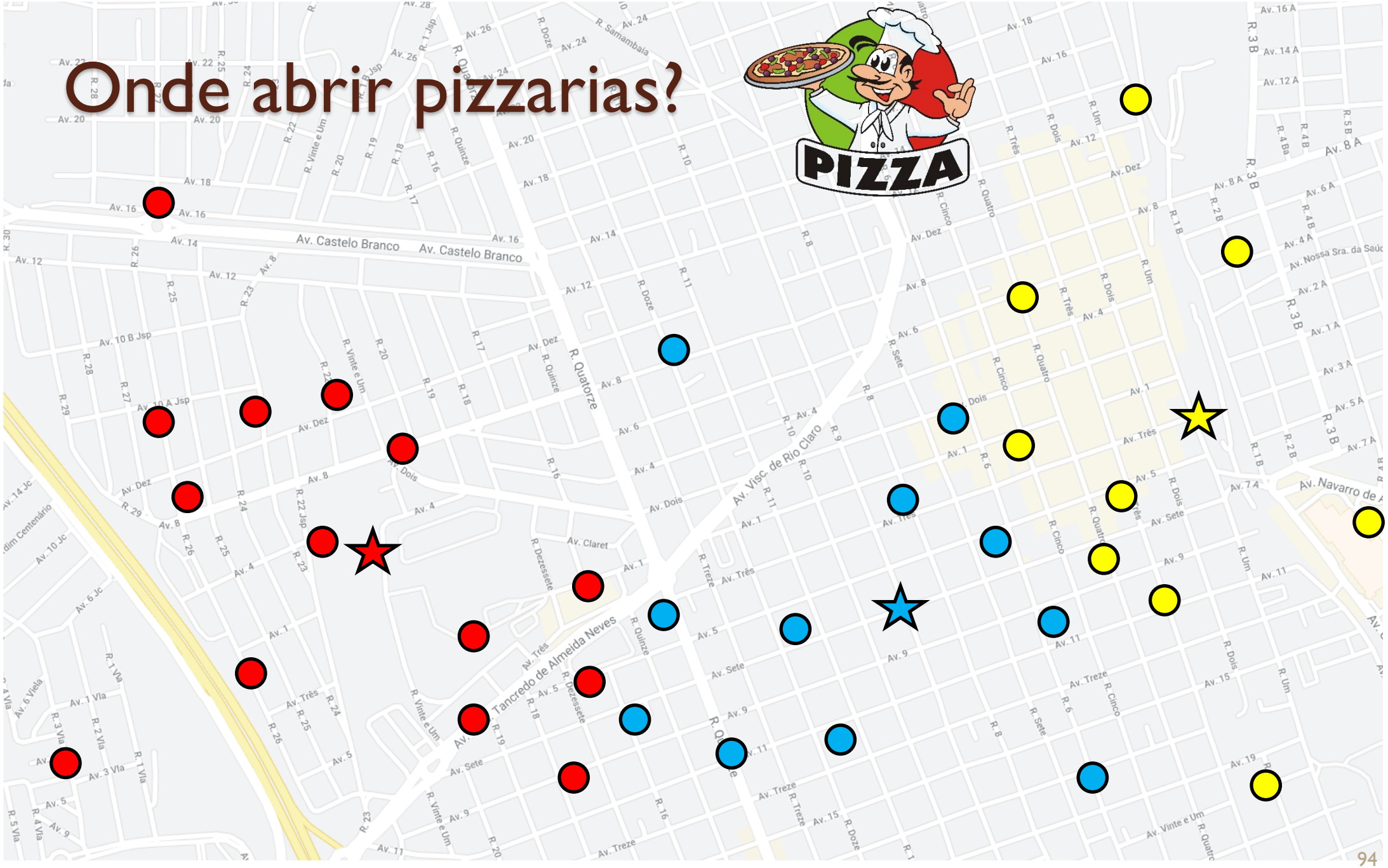
Onde abrir pizzarias?



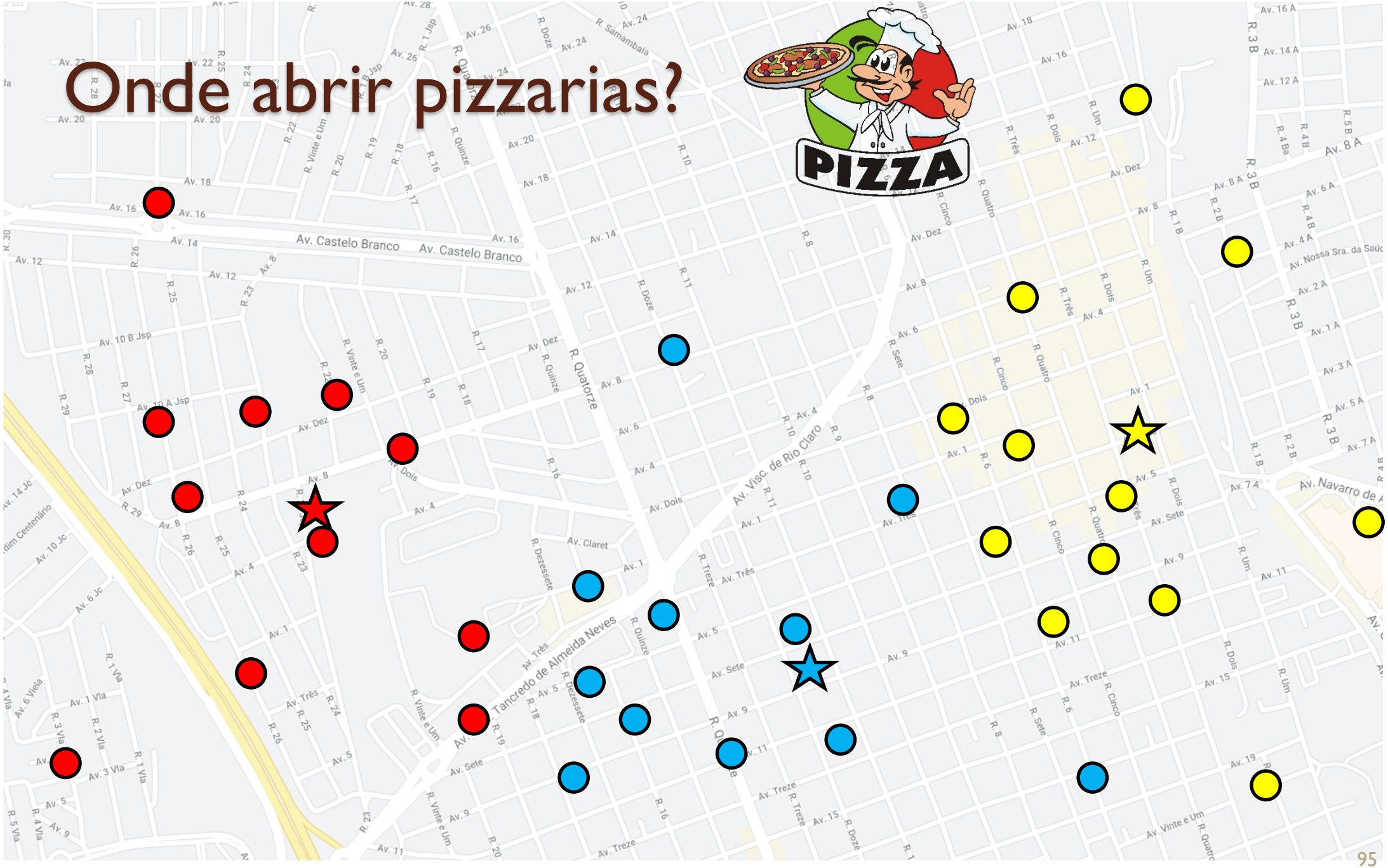
Onde abrir pizzarias?



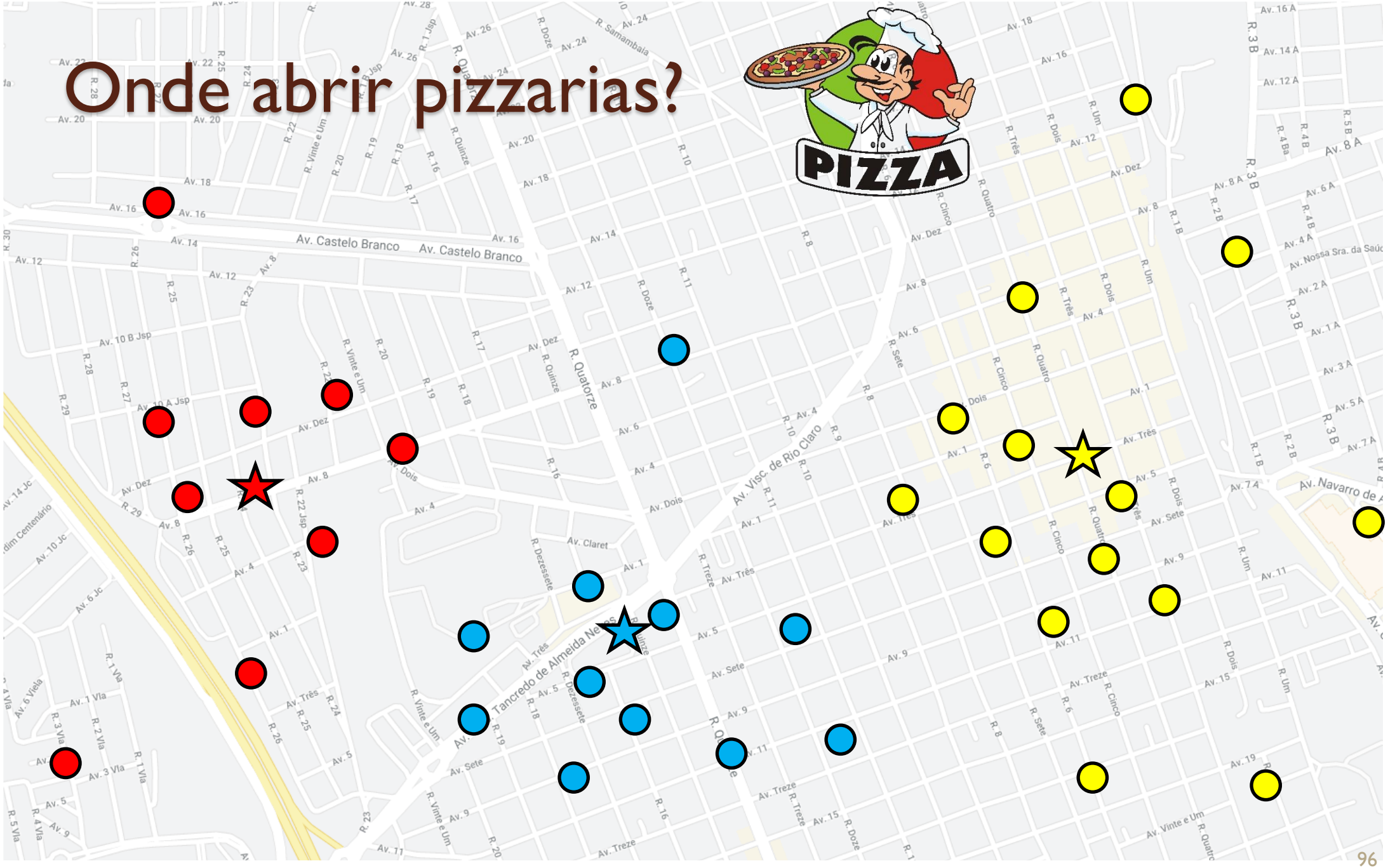
Onde abrir pizzarias?



Onde abrir pizzarias?



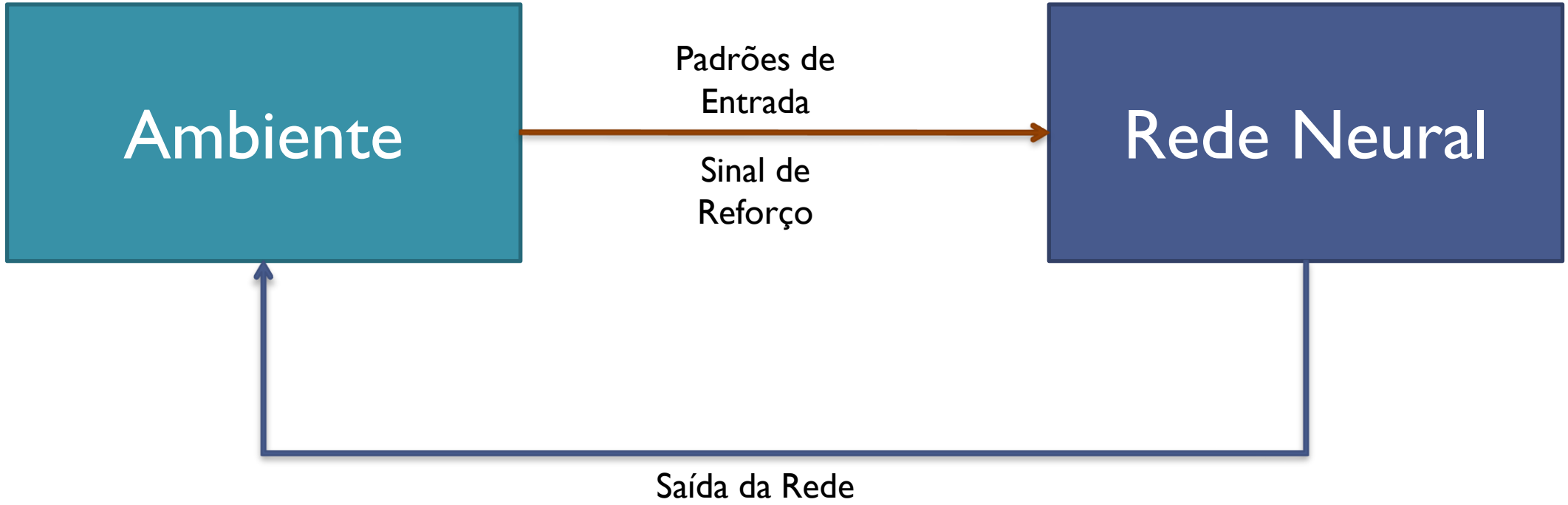
Onde abrir pizzarias?



Aprendizado por Reforço

- Aprende diretamente da interação com o ambiente, porém sem uma supervisão explícita.
- Em geral só recebe uma indicação escalar de quão bem a rede neural está desempenhando.
  - Sinal de recompensa ou punição.

Aprendizado por Reforço

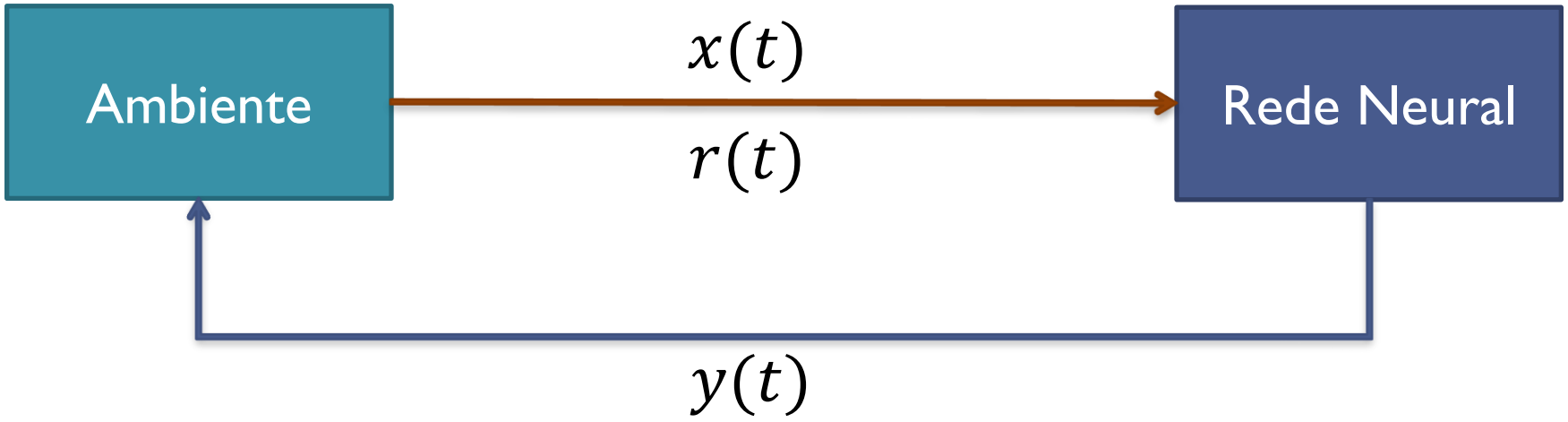


Aprendizado por Reforço

- É dado um objetivo para a rede neural atingir.
  - A rede *tenta* algumas ações (valores de saída).
    - É recompensada ou penalizada.
  - O algoritmo seletivamente retém as saídas que maximizam a recompensa.

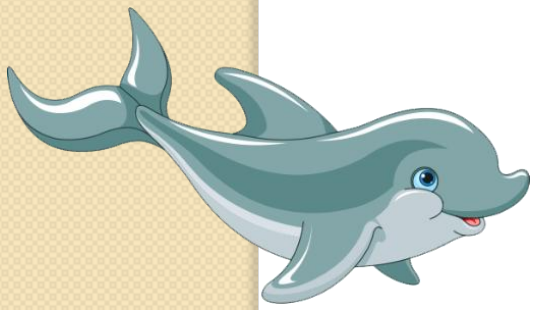
Aprendizado por Reforço

- A cada instante de tempo t, o sistema recebe uma representação do estado do ambiente $\mathbf{x}(t)$ e fornece uma saída $y(t)$, no próximo passo ele recebe a recompensa $r(t+1)$ e se encontra num novo estado $\mathbf{x}(t+1)$
  - Conceito de *busca por tentativa e erro* e *recompensa com atraso*.

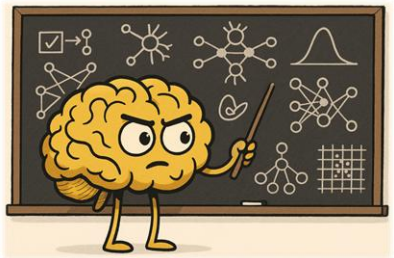


Aprendizado por Reforço

- Exemplos:
  - Simulador de vôo:
    - Você tem de fazer um pouso suave, não há um supervisor.
      - Avião espatifando no solo = reforço negativo.
      - Avião pousando suavemente = reforço positivo.
  - Ensinando um novo truque para um golfinho:
    - Você não pode dizer o que ele tem de fazer.
    - Mas pode recompensá-lo ou puni-lo quando ele fizer corretamente ou erroneamente.
      - Ele vai ter que descobrir o que o fez ser recompensado ou punido.



Próxima Aula



- Perceptron de Uma Única Camada
- Separabilidade Linear
- Perceptron Simples para Classificação de Padrões
- Treinamento
- Perceptron de Múltiplas Saídas para Classificação de Padrões
- Um-versus-Resto
- Função Softmax
- Exemplo de Aplicação
- Saída Desejada
- Ciclos (Épocas)
- Pesos Finais
- Exercício
- Generalização
- Adaline
- Perceptron de Múltiplas Camadas
- Histórico
- Algoritmo de Retropagação de Erro
- Múltiplas Camadas
- Dificuldades de Aprendizado
- Atualização de Pesos
- Redes de Função de Base Radial
- Máquinas de Vetor de Suporte
- Mapas Auto-Organizáveis
- Mapa de Kohonen
- Redes Neurais Profundas
- Outras Redes Neurais
- Aplicações de Redes Neurais Artificiais
- Empresas pioneiras no uso de RNAs
- Conclusão

Bibliografia

- CASTRO, Leandro Nunes. *Fundamentals of Natural Computing: Basic Concepts, Algorithms, And Applications*. CRC Press, 2006.
- CARVALHO, André Ponce de Leon F. de. *Notas de Aula*, 2007.
- BROWNLEE, Jason. *Clever Algorithms: Nature-Inspired Programming Recipes*. Jason Brownlee, 2011.
- HAYKIN, Simon. *Neural Networks and Learning Machines*, 3rd Edition. Prentice Hall, 2008.
- KOVACS, Zsolt L. *Redes Neurais Artificiais: Fundamentos e Aplicações*. Livraria da Física, 2006.
- BISHOP, Christopher M. *Pattern Recognition and Machine Learning*. Springer, 2007.