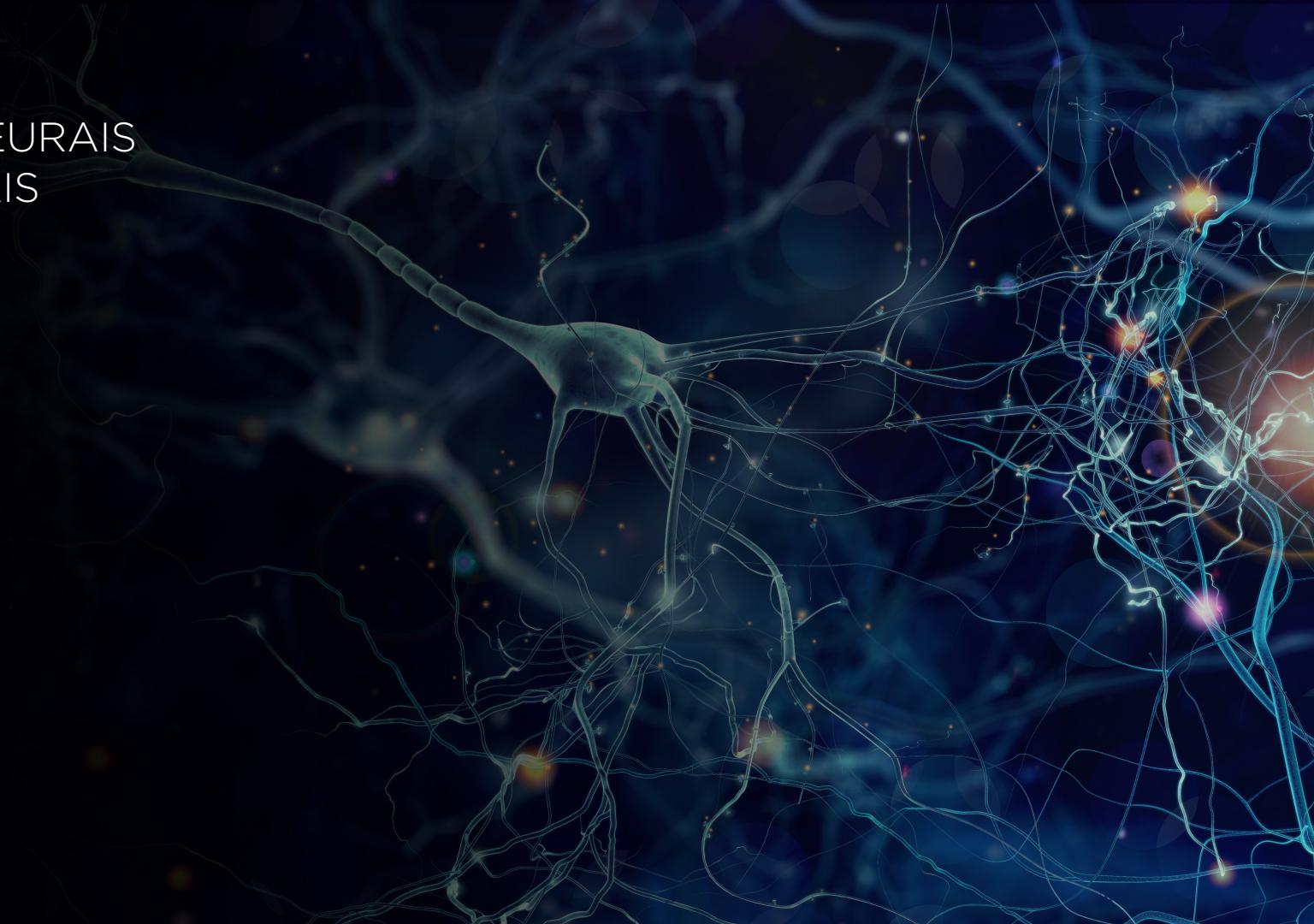


FIAP

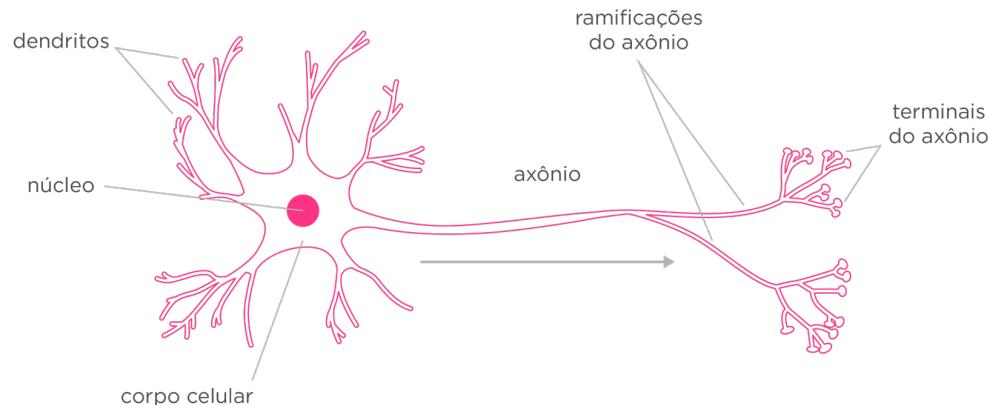
REDES NEURAIS & DEEP LEARNING

REDES NEURAIS ARTIFICIAIS



REDES NEURAIS ARTIFICIAIS

São sistemas inspirados nos neurônios biológicos e na estrutura maciçamente paralela do cérebro, com capacidade de adquirir, armazenar e utilizar conhecimento experimental.



REDES NEURAIS ARTIFICIAIS

Cérebro x Computador:

Item	Computador	Cérebro
Complexidade	<ul style="list-style-type: none">• Estrutura ordenada.• Processamento serializado.	<ul style="list-style-type: none">• 10^{10} Neurônios “processadores” com até 10^4 conexões.
Velocidade de processamento	<ul style="list-style-type: none">• 10.000.000 de operações por segundo.	<ul style="list-style-type: none">• 100 operações por segundo.
Poder computacional	<ul style="list-style-type: none">• Uma operação por vez com 1 ou 2 entradas.	<ul style="list-style-type: none">• Milhões de operações por vez, com milhares de entradas.

REDES NEURAIS ARTIFICIAIS

Cérebro x Computador:

- Aritmética: 1 cérebro = 1/10 calculadora de bolso.
- Visão: 1 cérebro = 1000 supercomputadores.
- Memória de detalhes arbitrários: computador ganha.
- Memória de fatos do mundo real: o cérebro vence.
- Um computador deve ser programado explicitamente.
- O cérebro pode aprender experimentando o mundo.

REDES NEURAIS ARTIFICIAIS

Algumas Curiosidades:

- Nascemos com cerca de 100 bilhões de neurônios.
- Um neurônio pode se conectar a até 100.000 outros neurônios.
- Os sinais “movem-se” via impulsos eletroquímicos.
- As sinapses liberam um transmissor químico - cuja soma faz com que um limiar seja alcançado -, fazendo com que o neurônio “dispare”.
- As sinapses podem ser inibitórias ou excitatórias.

REDES NEURAIS ARTIFICIAIS

Aplicações:

Classificação

- Reconhecimento de caracteres
- Reconhecimento de imagens
- Diagnóstico médico
- Análise de crédito
- Detecção de fraudes

Clusterização

- Agrupamento de sequências de DNA
- Mineração de dados
- Agrupamento de clientes

Previsão/Regressão

- Previsão do tempo (sistemas complexos)
- Previsão financeira (câmbio, bolsa...)
- Previsão de séries Temporais

REDES NEURAIS ARTIFICIAIS

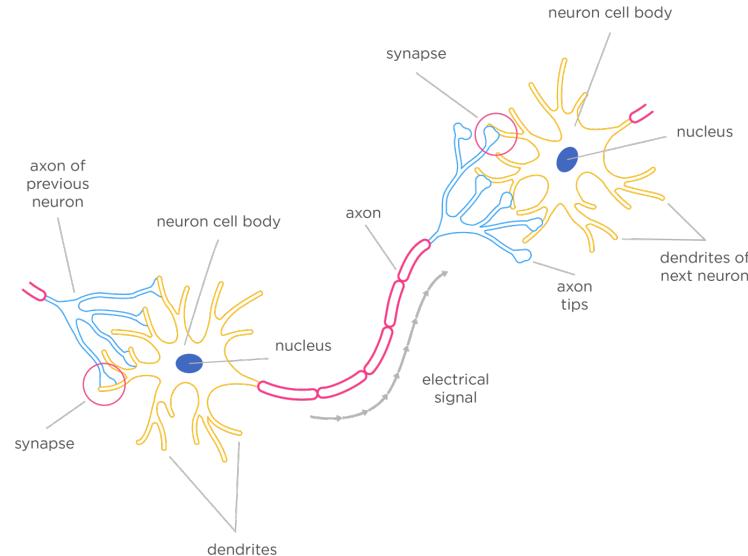
Inspiração biológica:

- O cérebro faz com que tarefas de classificação pareçam fáceis.
- O processo cerebral é realizado por redes de neurônios.
- Cada neurônio é conectado a vários outros neurônios.

REDES NEURAIS ARTIFICIAIS

Inspiração biológica:

- O neurônio recebe impulsos (sinais) de outros neurônios por meio dos seus dendritos.
- O neurônio envia impulsos para outros neurônios por meio do seu axônio.
- O axônio termina num tipo de contato chamado sinapse, que conecta-o com o dendrito de outro neurônio.



REDES NEURAIS ARTIFICIAIS

Inspiração biológica:

- vídeo

REDES NEURAIS ARTIFICIAIS

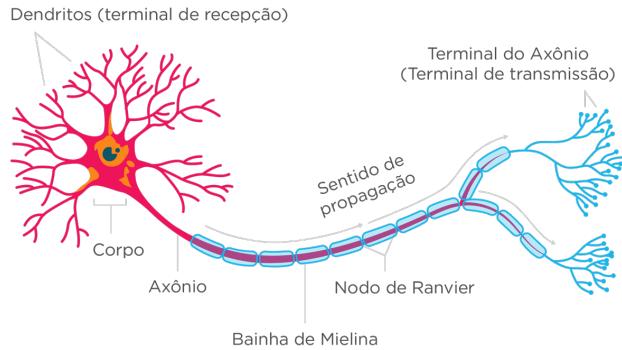
Inspiração biológica - Aprendizado:

- O aprendizado ocorre por sucessivas modificações nas sinapses que interconectam os neurônios em função da maior ou menor liberação de neurotransmissores.
- À medida em que novos eventos ocorrem, determinadas ligações entre neurônios são reforçadas, enquanto outras são enfraquecidas.
- Este ajuste nas ligações entre os neurônios é uma das características das redes neurais artificiais.

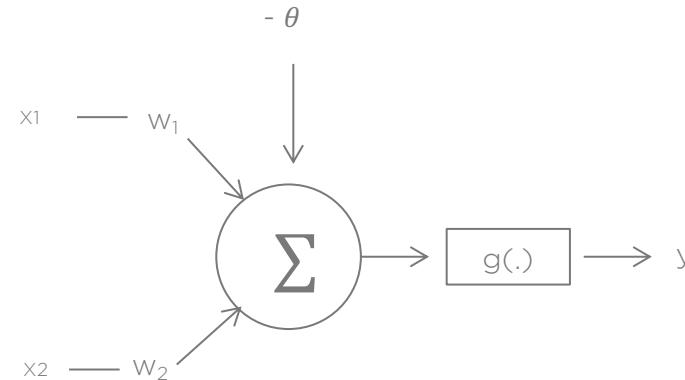
REDES NEURAIS ARTIFICIAIS

Inspiração:

- Neurônio biológico

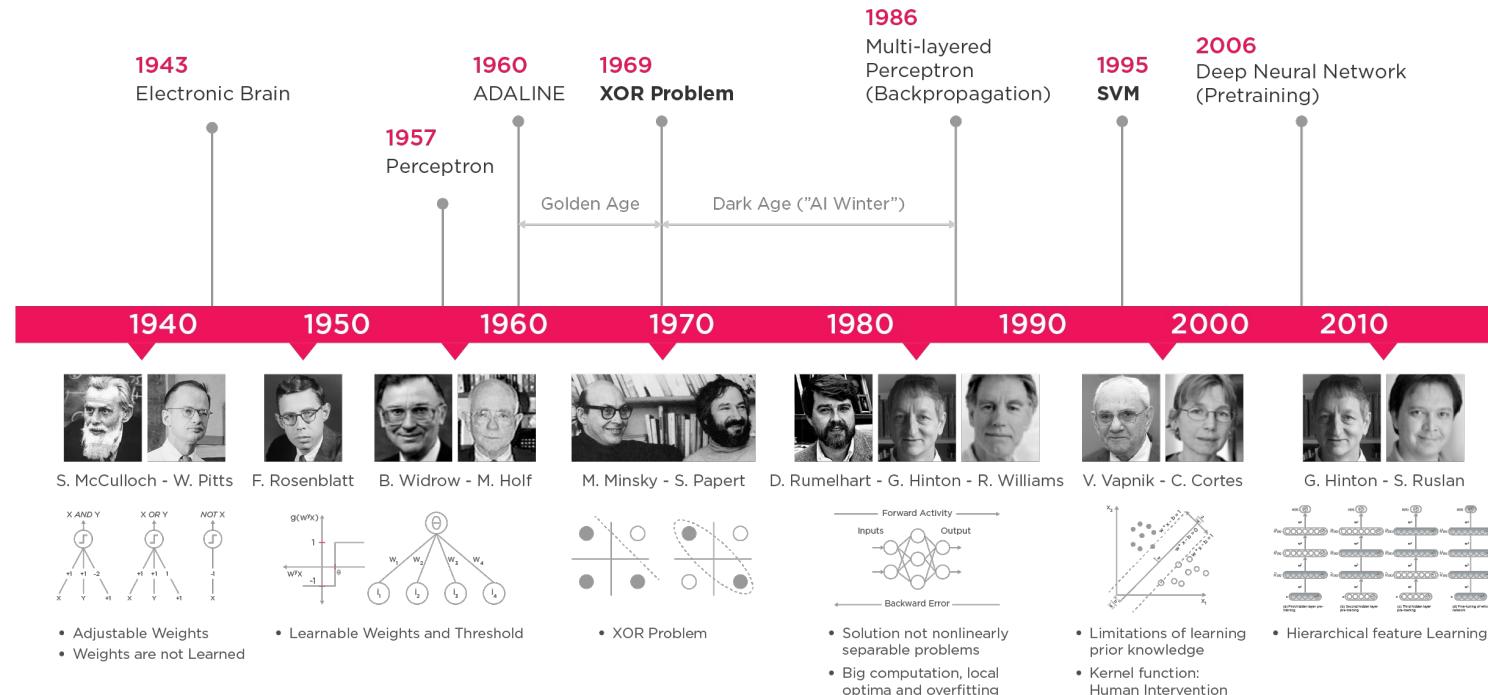


- Neurônio Artificial



REDES NEURAIS ARTIFICIAIS

História :



REDES NEURAIS ARTIFICIAIS

Neurônio de McCulloch e Pitts:

- McCulloch era um psiquiatra e um neuroanatomista. Pitts era um matemático que foi trabalhar com McCulloch na Universidade de Chicago; ambos faziam parte de um dos primeiros grupos do mundo dedicado ao estudo da Biofísica Teórica, criado por Nicolas Rashevsky.
- Em 1943, o conhecimento sobre os neurônios biológicos era muito limitado. As bases iônicas e elétricas da atividade neuronal eram ainda incertas, porém já se sabia da existência de potenciais de ação e da sua natureza “tudo ou nada”.
- McCulloch e Pitts propuseram um modelo de sistema neural em que as unidades básicas, os neurônios, são bastante simples no seu funcionamento.

REDES NEURAIS ARTIFICIAIS

Neurônio de McCulloch e Pitts:

1. A atividade de um neurônio é binária, ou seja, a cada instante o neurônio ou está disparando (atividade = 1) ou não está disparando (atividade = 0);
2. A rede neural é constituída por linhas direcionadas, com pesos ajustáveis, ligando os neurônios. Essas linhas (inspiradas nas sinapses) podem ser excitatórias ou inibitórias (positivas ou negativas);
3. Cada neurônio tem um limiar fixo θ , de maneira que ele só dispara se a entrada total chegando a ele, num dado instante, for maior ou igual a θ ;
4. A chegada de uma única sinapse inibitória num dado instante evita o disparo do neurônio, independentemente do número de sinapses excitatórias que estejam chegando conjuntamente com a sinapse inibitória.

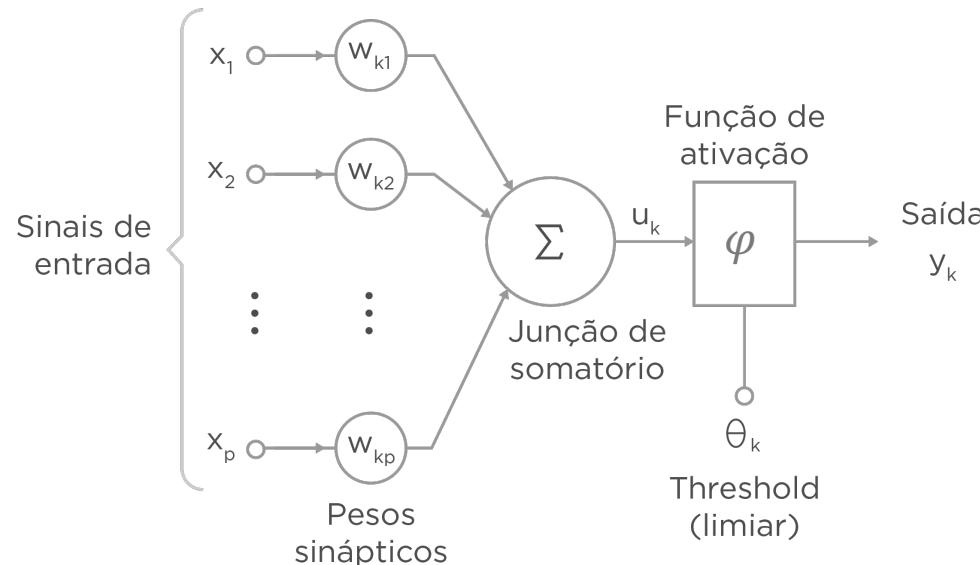
REDES NEURAIS ARTIFICIAIS

Neurônio de McCulloch e Pitts:

- Uma restrição existente no modelo criado é que as redes desenvolvidas só conseguem implementar funções linearmente separáveis, ou seja, aquelas que podem separar os padrões por meio de uma reta.

REDES NEURAIS ARTIFICIAIS

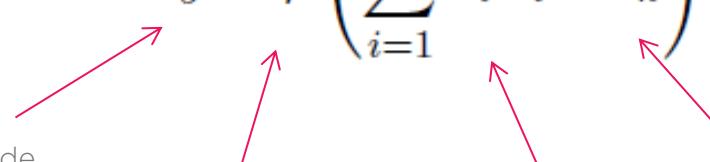
Neurônio de McCulloch e Pitts:



REDES NEURAIS ARTIFICIAIS

Neurônio de McCulloch e Pitts:

A saída y do neurônio de McCulloch-Pitts pode ser equacionada por:

$$y = \varphi \left(\sum_{i=1}^n x_i w_i + \theta_k \right)$$


The diagram shows the mathematical equation for a McCulloch-Pitts neuron. Red arrows point from the text labels below to specific parts of the equation: 'Saída da rede' points to the output variable y ; 'Função de ativação' points to the activation function symbol φ ; 'Porta do limiar' points to the summation term; and 'Limiar (Threshold)' points to the constant term θ_k .

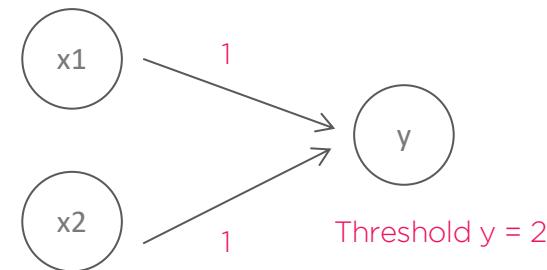
REDES NEURAIS ARTIFICIAIS

Neurônio de McCulloch e Pitts:

Exemplo: implementação de portas lógicas:

A função **AND** resulta na resposta “true” se ambas as entradas são valoradas com “true”; caso contrário a resposta é “false”. Se nós representamos “true” por ‘1’ e “false” por ‘0’, isso nos dá o seguinte conjunto de 4 pares (padrões) de treinamento entrada/saída (alvo):

X1	x2	Y
0	0	0
1	0	0
0	1	0
1	1	1



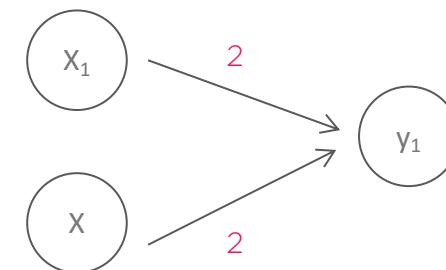
REDES NEURAIS ARTIFICIAIS

Neurônio de McCulloch e Pitts:

Exemplo: Implementação de portas lógicas:

A função OR (threshold = 2) :

X1	x2	Y
0	0	0
1	0	1
0	1	1
1	1	1



REDES NEURAIS ARTIFICIAIS

Aprendizado:

A principal propriedade de uma rede neural é sua capacidade de aprender a partir de estímulos do ambiente e de melhorar seu desempenho por meio do aprendizado.

O processo de aprendizado é iterativo e implica na alteração dos pesos (sinápticos) da rede neural.

REDES NEURAIS ARTIFICIAIS

Aprendizado:

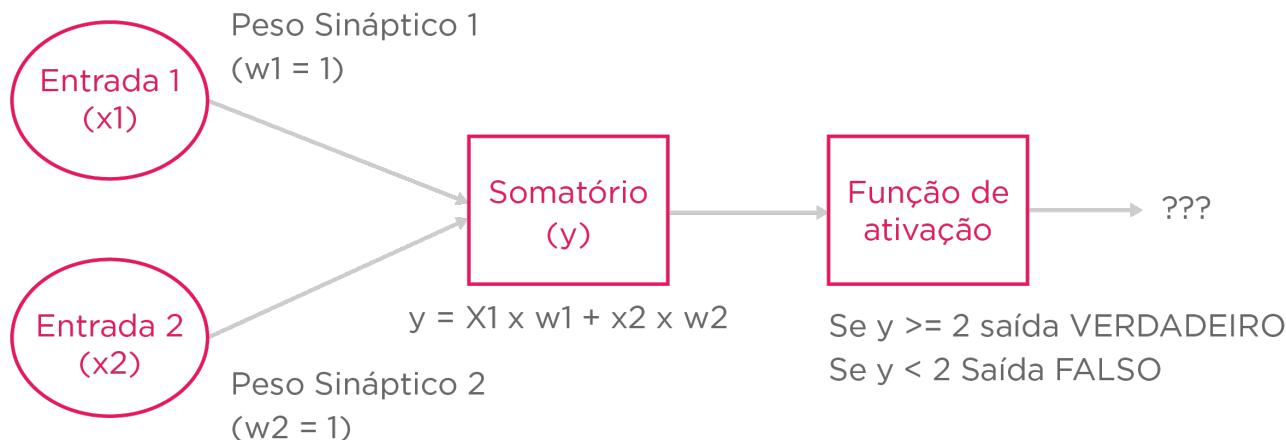
Definição de Mendel e McLaren (1970):

Aprendizado é um processo pelo qual parâmetros livres de uma rede neural são adaptados por meio de um processo de estimulação vindo do ambiente em que a rede está incorporada. O tipo de aprendizado é determinado pela maneira como os parâmetros são alterados.

REDES NEURAIS ARTIFICIAIS

Aprendizado:

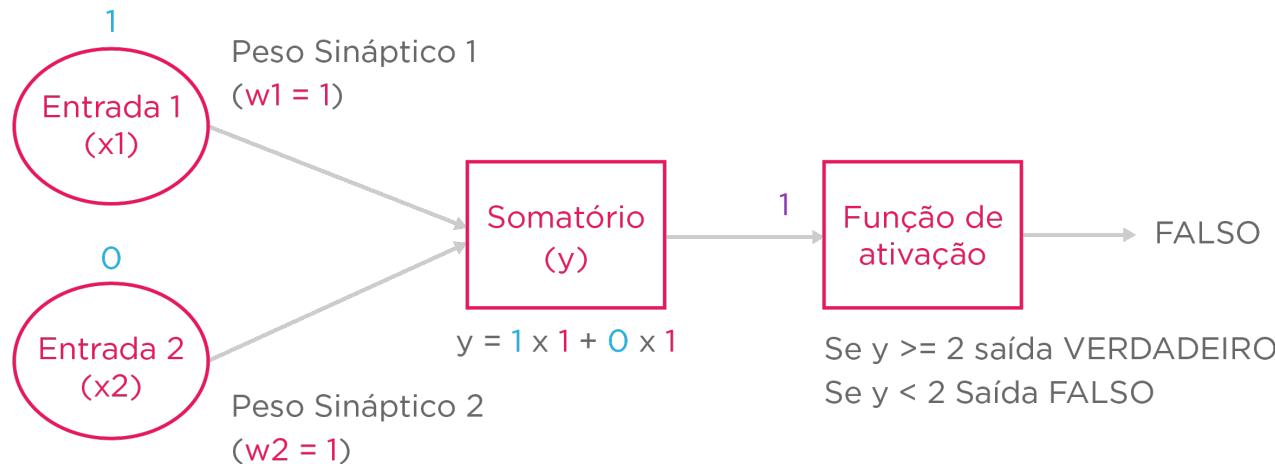
O que queremos:



REDES NEURAIS ARTIFICIAIS

Aprendizado:

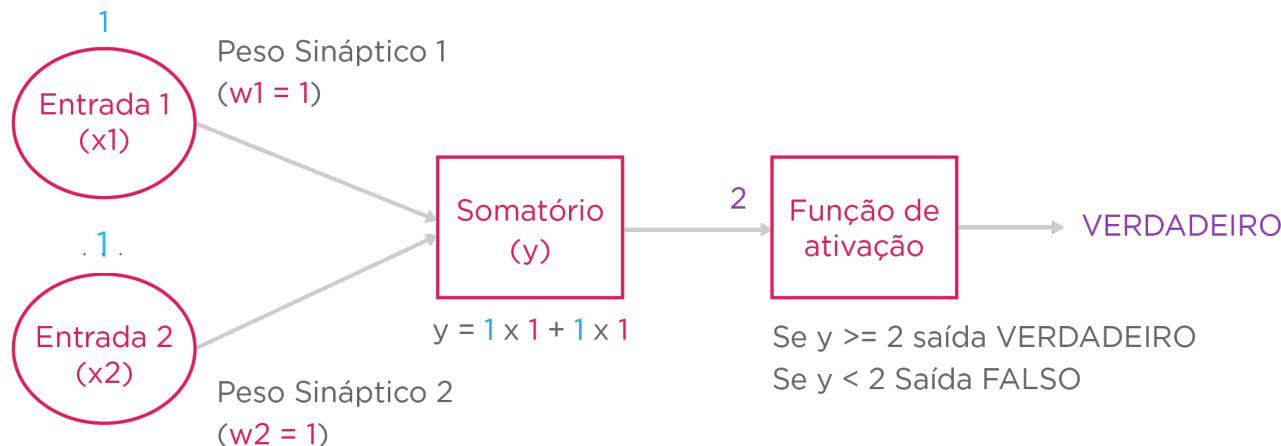
O que queremos:



REDES NEURAIS ARTIFICIAIS

Aprendizado:

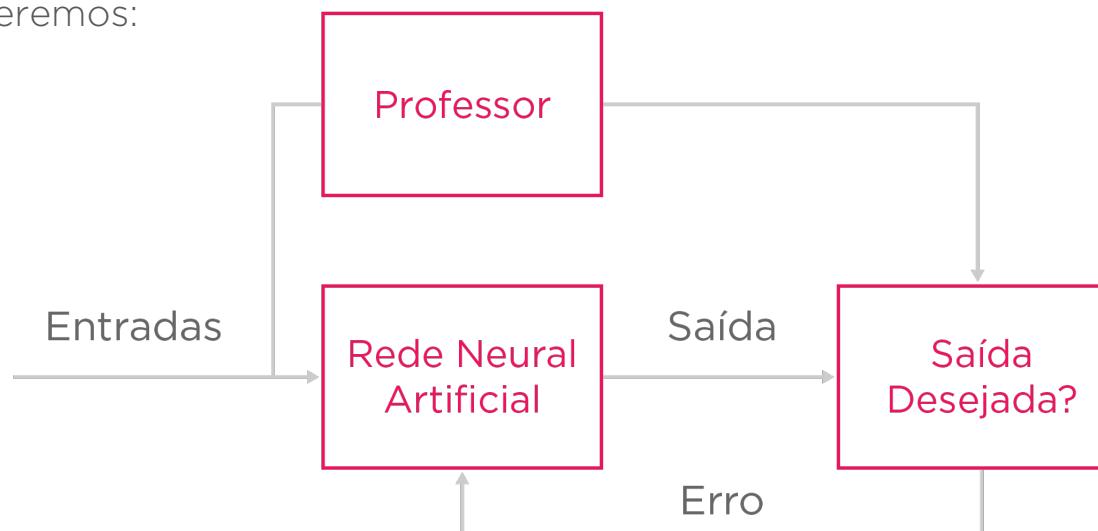
O que queremos:



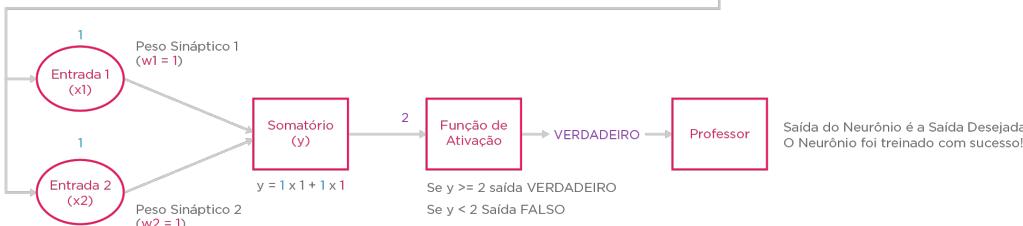
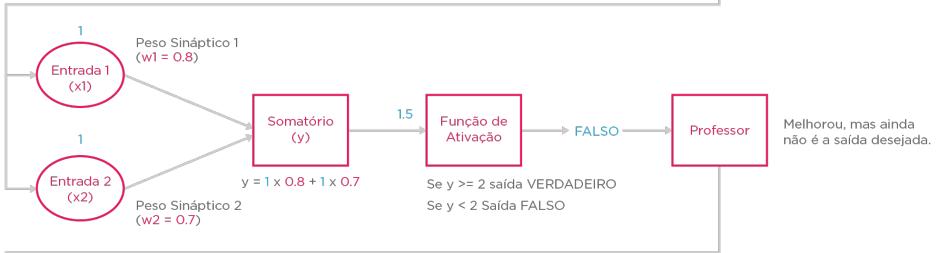
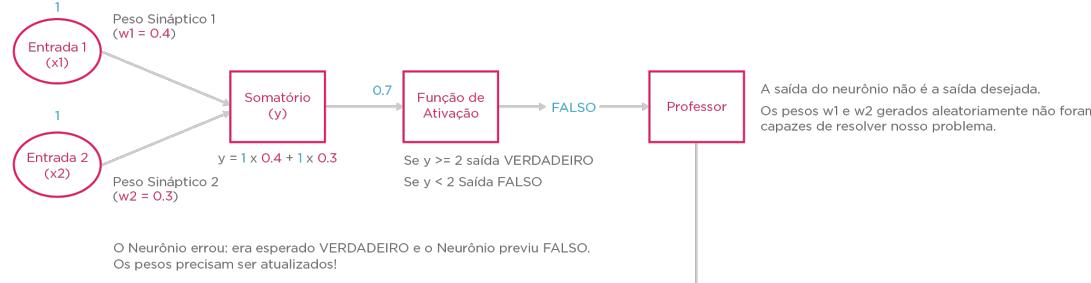
REDES NEURAIS ARTIFICIAIS

Aprendizado:

O que queremos:



REDES NEURAIS ARTIFICIAIS



REDES NEURAIS ARTIFICIAIS

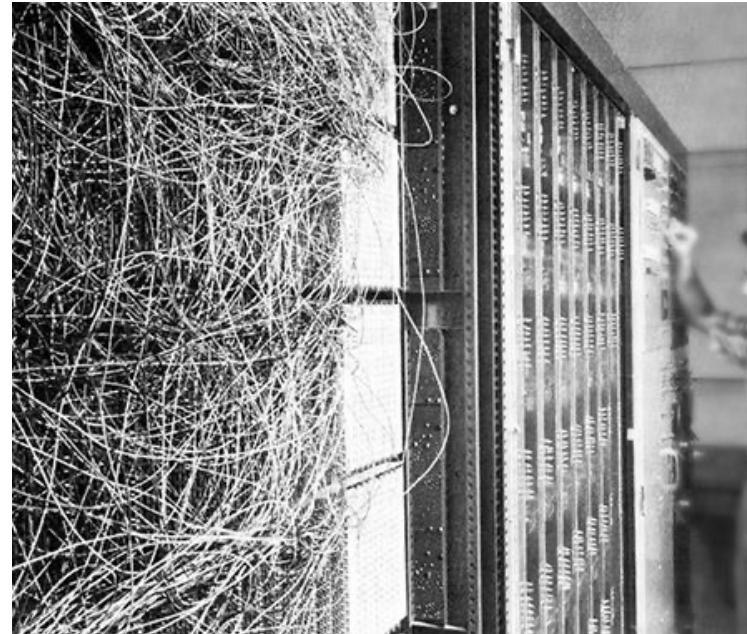
Algoritmos de aprendizagem:

Um conjunto de regras (passos) bem definidas para solucionar o problema de aprendizado é chamada de algoritmo de aprendizado.

- Existem vários algoritmos de aprendizado diferentes.
- Eles diferem-se entre si pela maneira como ajustam (alteram) os pesos.

REDES NEURAIS ARTIFICIAIS

Perceptron:



REDES NEURAIS ARTIFICIAIS

Perceptron:

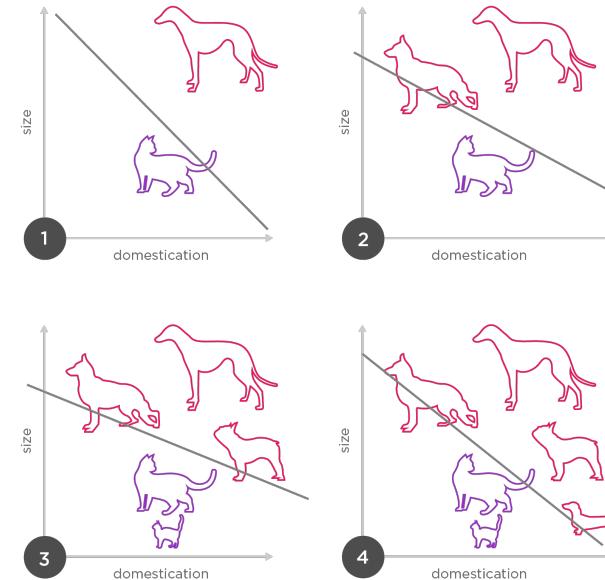
No final da década de 1950, Rosenblatt, na Universidade de Cornell, criou uma genuína rede de múltiplos neurônios do tipo discriminadores lineares e chamou essa rede de perceptron.

Perceptron pode ser visto como o tipo mais simples de rede neural feedforward: um classificador linear.

REDES NEURAIS ARTIFICIAIS

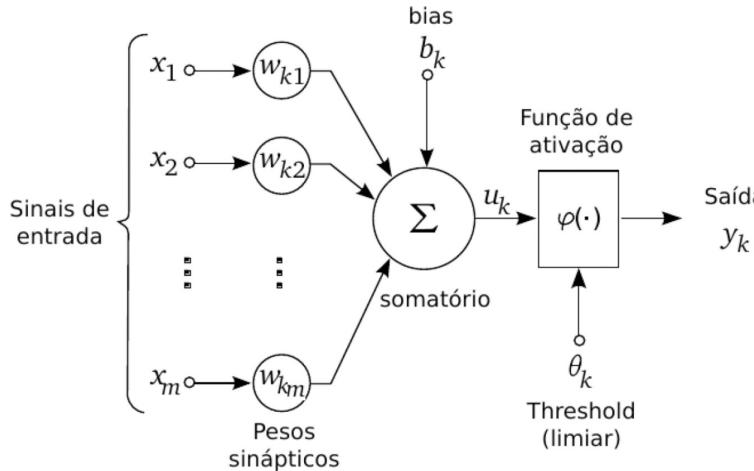
Perceptron:

O Perceptron, proposto por Rosenblatt, é composto pelo neurônio de McCulloch-Pitts, com função de limiar e aprendizado supervisionado. Sua arquitetura consiste na entrada e uma camada de saída.



REDES NEURAIS ARTIFICIAIS

Perceptron:



Considerações:

$$\vec{x} = [1, x_1, x_2, \dots, x_n]$$

$$\vec{w} = [-b, w_1, w_2, \dots, w_n]$$

$$\sum_i w_i x_i = \vec{w} \cdot \vec{x}$$

produto
interno

$$\vec{w}(t+1) = \vec{w}(t) + ? \Rightarrow$$

COMO ATUALIZAR OS PESOS
DA REDE NEURAL?

MAS ANTES...

COMPONENTES DE REDES NEURAIS ARTIFICIAIS

O processo de aprendizagem pode variar de acordo como a Rede Neural é composta, por isso precisamos determinar:

- A função de ativação;
- A arquitetura da Rede Neural
- O otimizador (gradiente);
- A taxa de aprendizagem;
- E a função de erro (também conhecida como função de perda - loss function).

TIPOS DE FUNÇÃO DE ATIVAÇÃO:

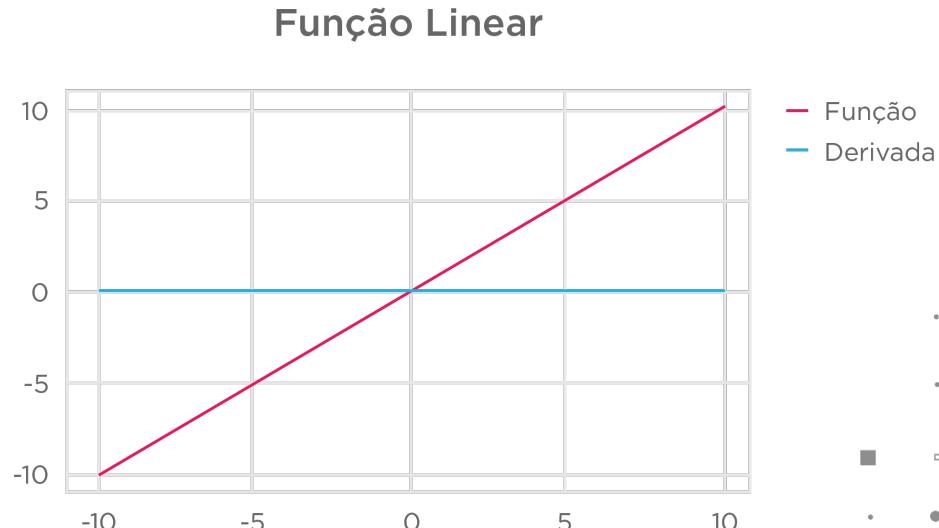
O QUE É A DERIVADA **DE**
UMA FUNÇÃO?

REDES NEURAIS ARTIFICIAIS

Tipos de Função de Ativação:

Função linear:

- A derivada de uma função linear é constante, isto é, não depende do valor de entrada x .



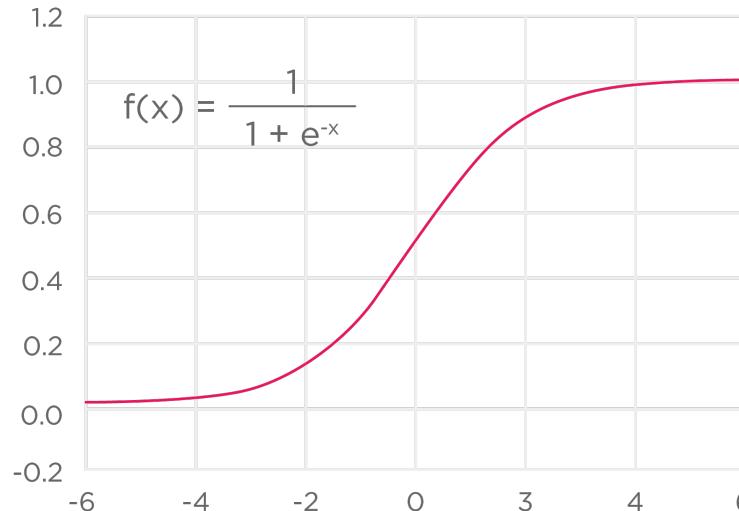
REDES NEURAIS ARTIFICIAIS

Tipos de Função de Ativação:

Função Sigmóide (Sigmoid Function):

- Boa para interpretação da “força” do estímulo do neurônio;
- Função não linear, consegue “afastar” bem os dados no eixo y;
- Pode saturar o processo de treinamento quando os valores são superiores a +5 e -5 e pode haver dificuldades no treinamento quando são próximos de zero;
- Trabalha apenas com valores positivos na saída, o que pode ser um problema.

$$R^n \rightarrow [0,1]$$



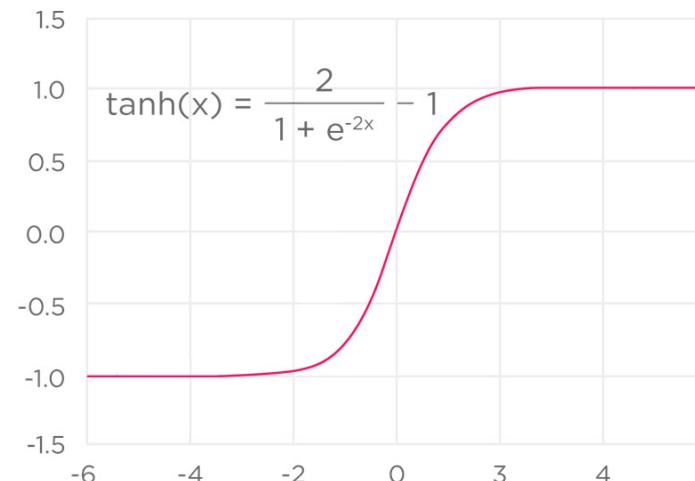
• • • • • + • • • • • . • + REDES NEURAIS ARTIFICIAIS

Tipos de Função de Ativação:

Função Tangente Hiperbólica (tanh):

- Não é centralizada no “zero”;
- É uma versão escalonada da função sigmóide.

$$R^n \rightarrow [-1,1]$$



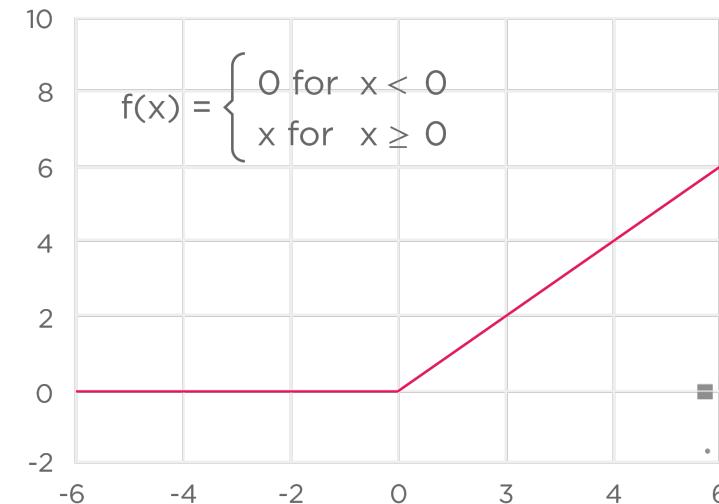
• • • • • + • • • • • . • + REDES NEURAIS ARTIFICIAIS

Tipos de Função de Ativação:

Função ReLU (unidade linear retificada):

- Largamente utilizada em redes neurais convolucionais (Deep Learning);
- Acelera a convergência para a descida do Gradiente;
- Menor complexidade computacional;
- Não ativa todos os neurônios ao mesmo tempo.

$$R^n \rightarrow R_+^n$$

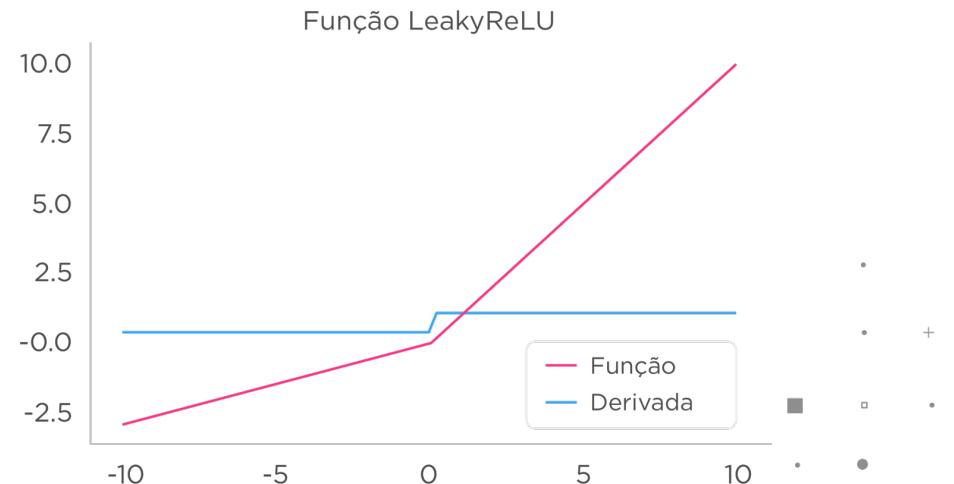


REDES NEURAIS ARTIFICIAIS

Tipos de Função de Ativação:

Função Leaky ReLU:

- Versão melhorada da função ReLU;
- Faz ativação para gradientes cujo valor é menor que zero;
- A principal vantagem de substituir a linha horizontal é remover o gradiente zero.

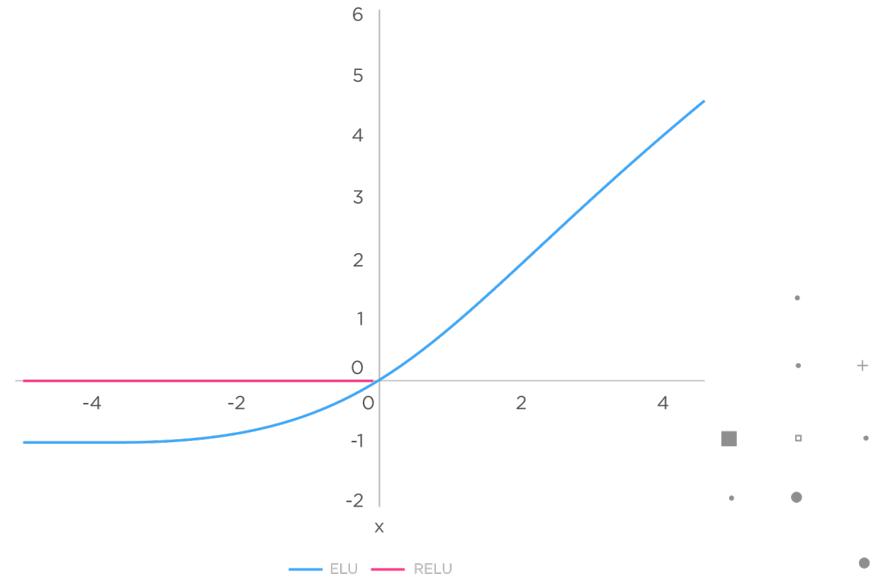


• • • • • + • • • • • . • + REDES NEURAIS ARTIFICIAIS

Tipos de Função de Ativação:

Função ELU (Exponential Linear Unit):

- O ELU é muito semelhante ao RELU, exceto a entradas negativas negativos. Ambos estão na forma de função de identidade para entradas não negativas. Por outro lado, lentamente o ELU se torna suave até que sua saída seja igual a $-\alpha$ enquanto RELU suaviza acentuadamente.



• • • • • + • • • • • . • + REDES NEURAIS ARTIFICIAIS

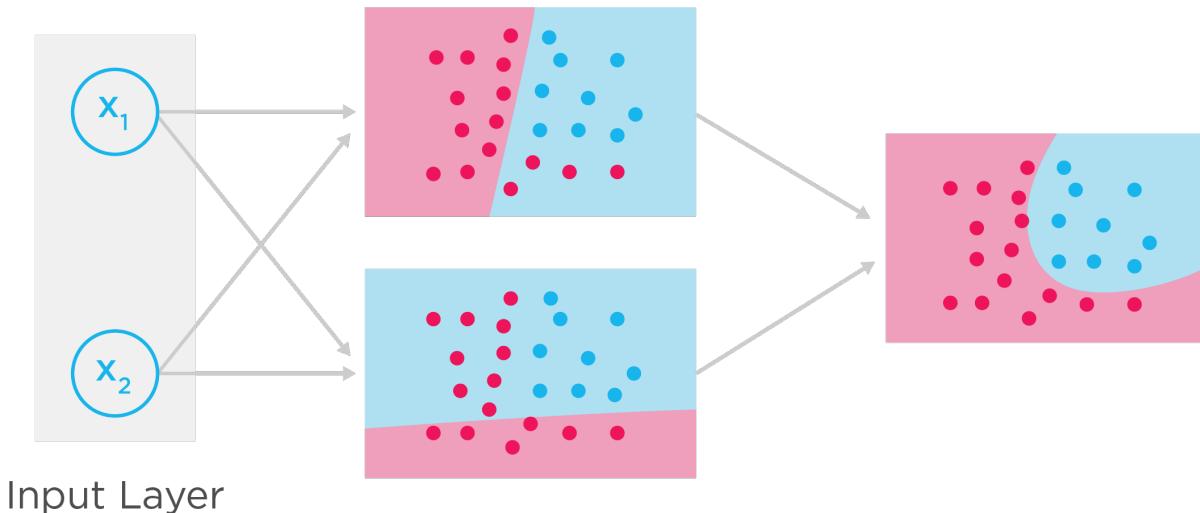
Tipos de Função de Ativação:

- Funções Sigmóide e suas combinações geralmente funcionam melhor no caso de problemas de classificação, principalmente na camada de saída;
- Funções Sigmóide e Tanh, às vezes, são evitadas devido ao problema de Vanishing Gradient em redes neurais recorrentes (veremos mais adiante);
- A função ReLU é uma função de ativação geral e é usada na maioria dos casos atualmente; . .
- Se encontrarmos um caso de neurônios deficientes em nossas redes, a função Leaky ReLU é a melhor escolha;
- Tenha sempre em mente que a função ReLU deve ser usada apenas nas camadas ocultas. . .

REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas:

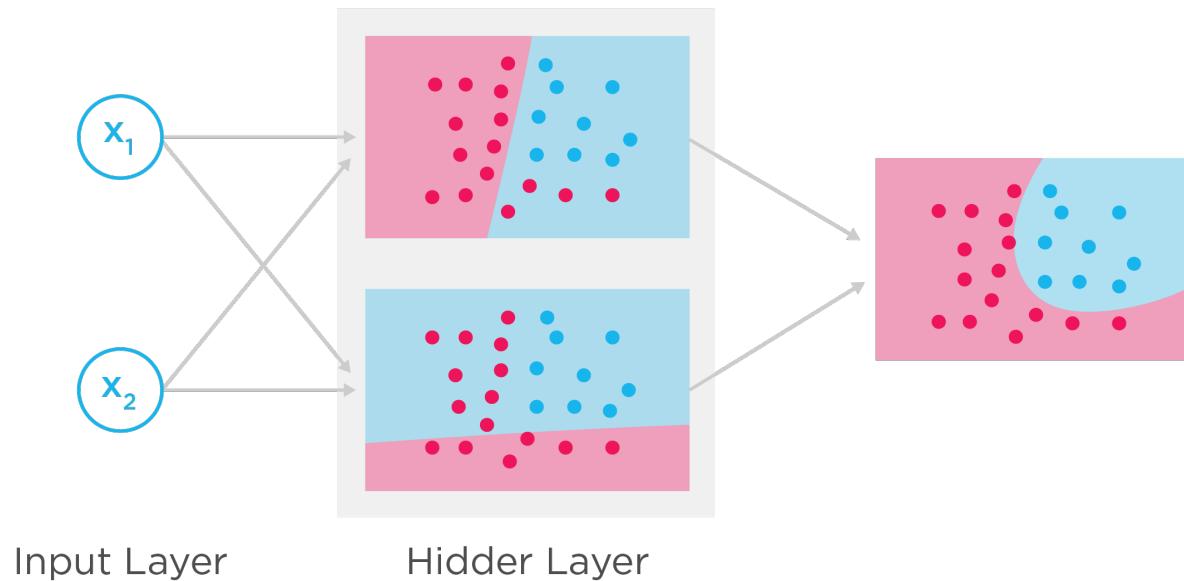
Neural Network



- • • • •
- +
- • • • •
- + REDES NEURAIS ARTIFICIAIS
- +
-

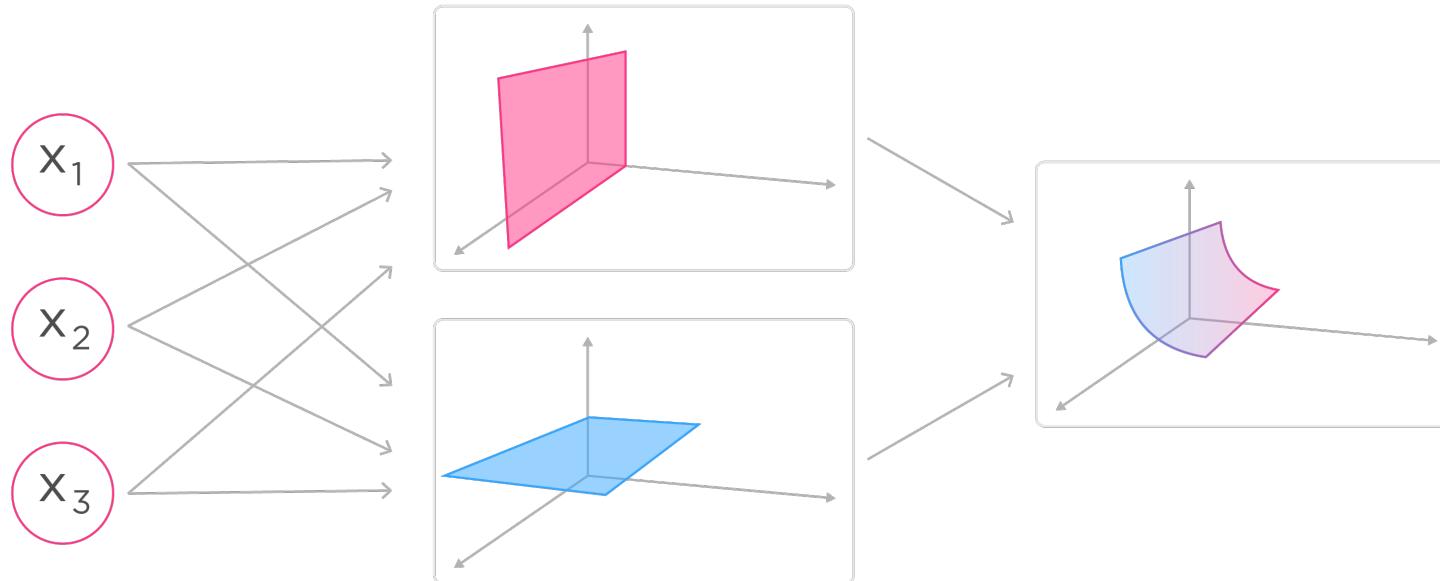
Redes Neurais Multicamadas:

Neural Network



REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas:



TIPOS DE REDES NEURAIS ARTIFICIAIS

A mostly complete chart of

Neural Networks

©2016 Fjodor van Veen - asimovinstitute.org

Backfed Input Cell

Input Cell

Noisy Input Cell

Hidden Cell

Probabilistic Hidden Cell

Spiking Hidden Cell

Output Cell

Match Input Output Cell

Recurrent Cell

Memory Cell

Different Memory Cell

Kernel

Convolution or Pool

Perceptron (P)



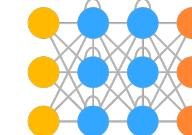
Feed Forward (FF)



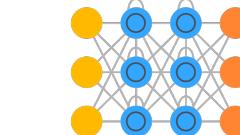
Radial Basis Network (RBF)



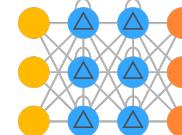
Recurrent Neural Network (RNN)



Long / Short Term Memory (LSTM)



Gated Recurrent Unit (GRU)



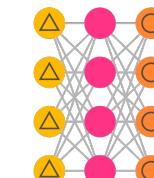
Auto Encoder (AE)



Variational AE (VAE)



Denoising AE (DAE)

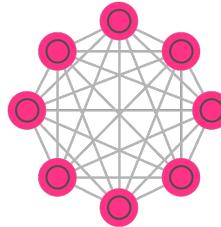


Sparse AE (SAE)

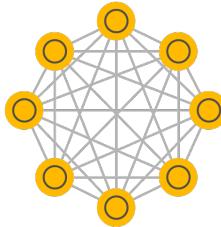


TIPOS REDES NEURAIS ARTIFICIAIS

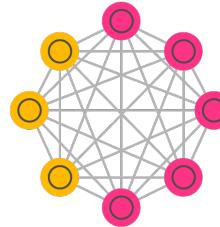
Markov Chain (MC)



Hopfield Network (HN)



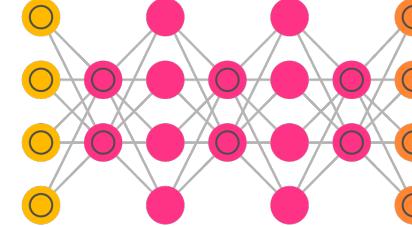
Boltzmann Machine (BM)



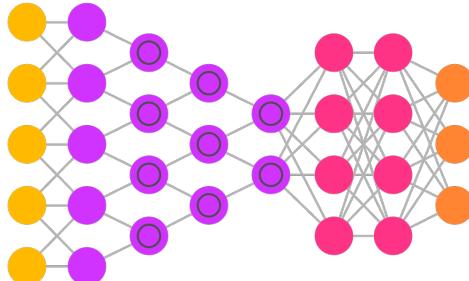
Restricted BM (RBM)



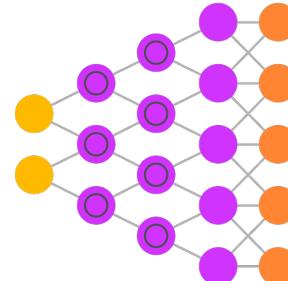
Deep Belief Network (DBN)



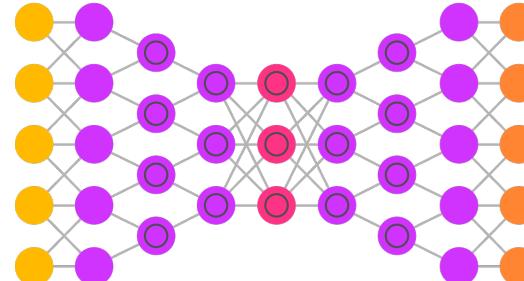
Deep Convolutional Network (DCN)



Deconvolutional Network (DN)

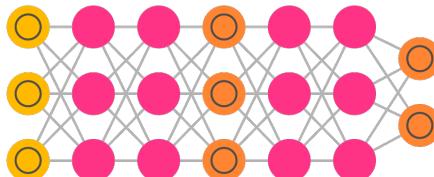


Deep Convolutional Inverse Graphics Network (DCIGN)

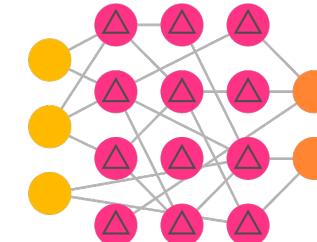


TIPOS REDES NEURAIS ARTIFICIAIS

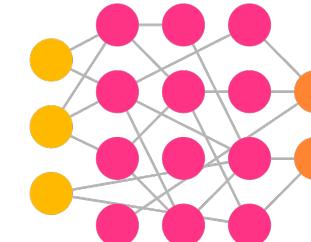
Generative Adversarial Network (GAN)



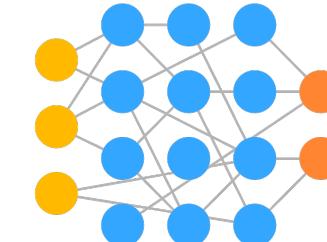
Liquid State Machine (LSM)



Extreme Learning Machine (ELM)



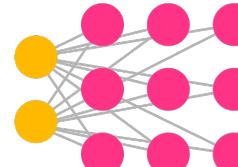
Echo State Network (ESN)



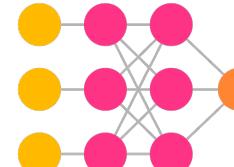
Deep Residual Network (DRN)



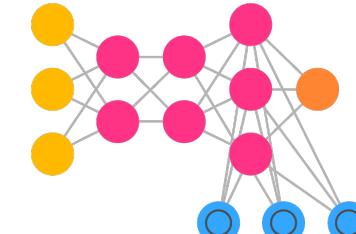
Kohonen Network (KN)



Support Vector Machine (SVM)

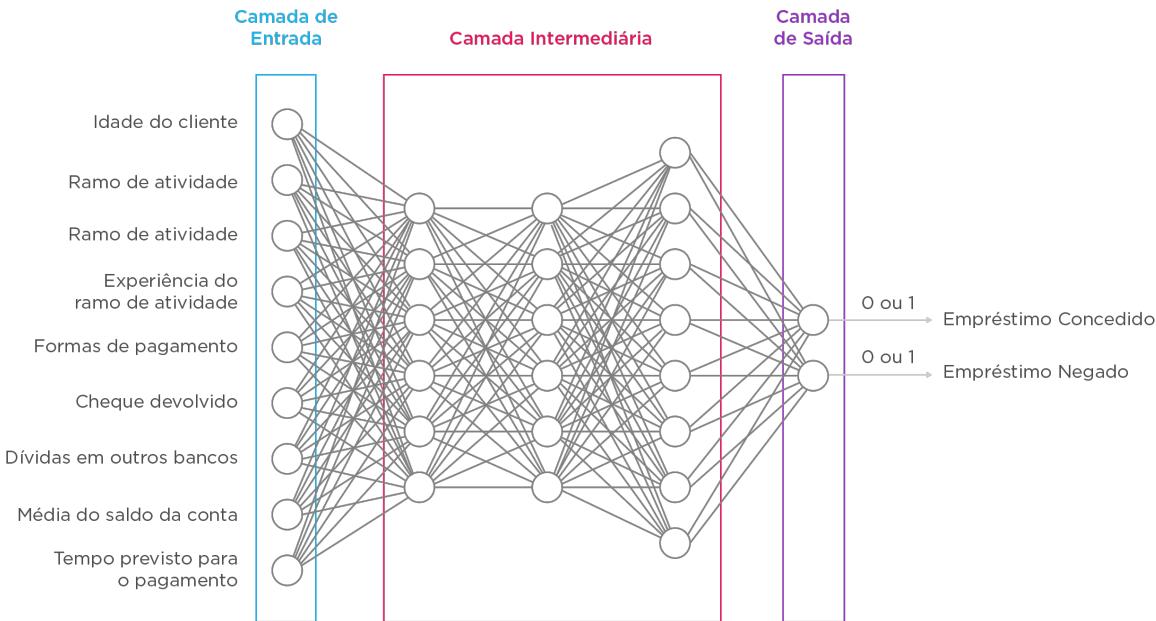


Neural Turing Machine (NTM)



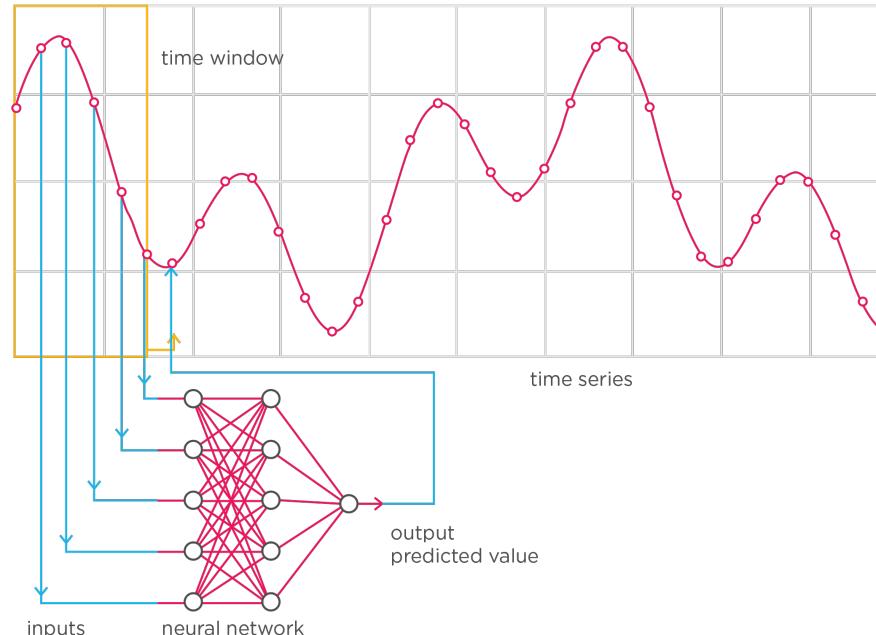
- • • • •
- +
- • • • •
- + TIPOS REDES NEURAIS ARTIFICIAIS
-

- + Multi Layer Perceptron (MLP) :



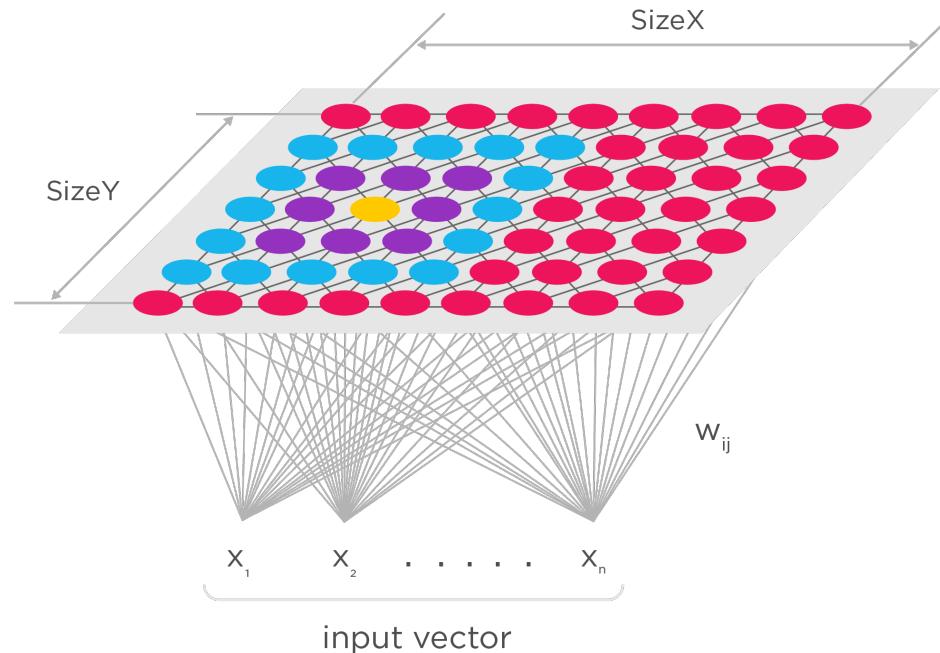
TIPOS REDES NEURAIS ARTIFICIAIS

Redes Recorrentes:



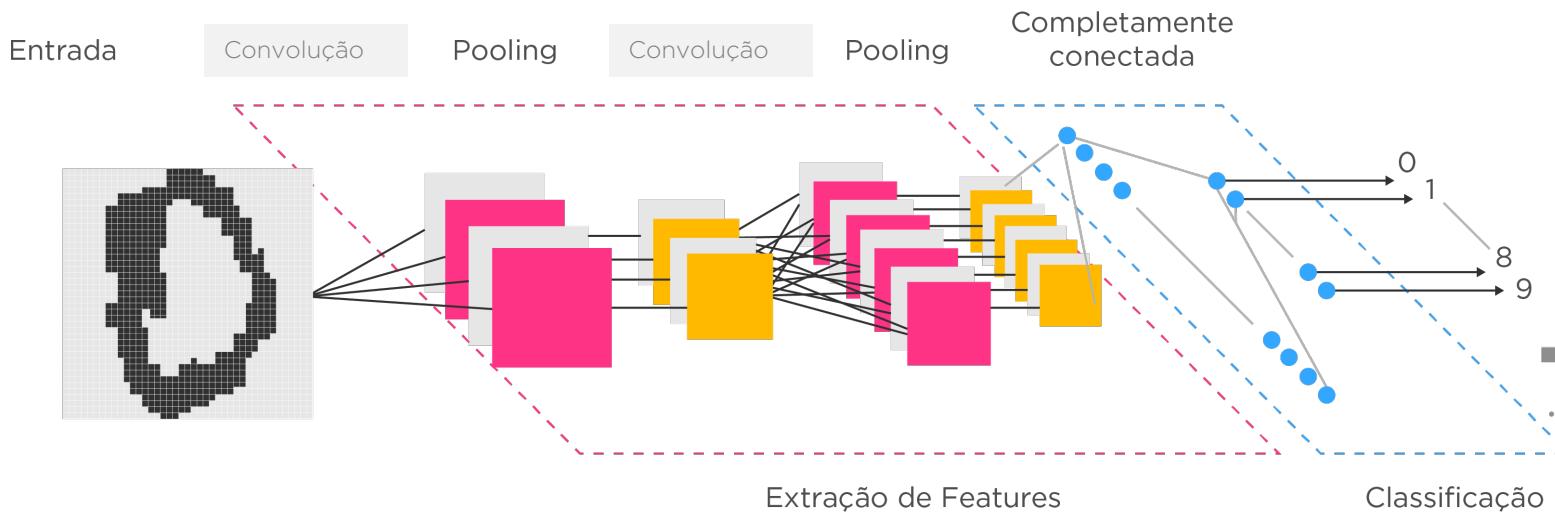
TIPOS REDES NEURAIS ARTIFICIAIS

Self Organized Maps:



- • • • •
- +
- • • • •
- + TIPOS REDES NEURAIS ARTIFICIAIS
-

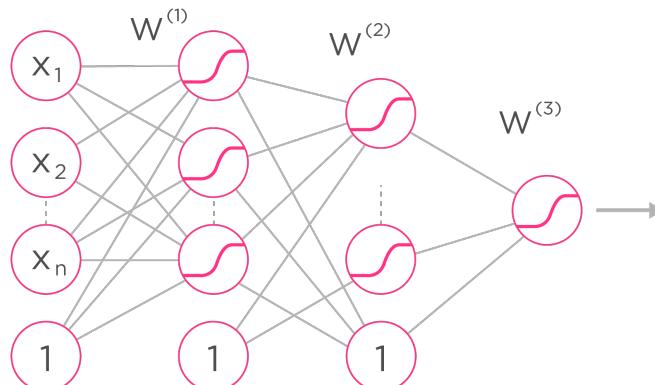
Redes Neurais Convolucionais :



- • • • •
- +
- • • • •
- + REDES NEURAIS ARTIFICIAIS
- +
-

Redes Neurais Multicamadas - Topologia FeedForward

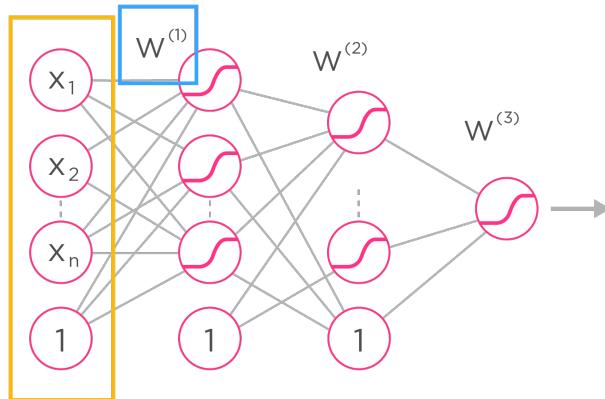
Multi-layer Perceptron



- • • • •
- +
- • • • •
- +
- REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Topologia FeedForward

Multi-layer Perceptron



PREDICTION

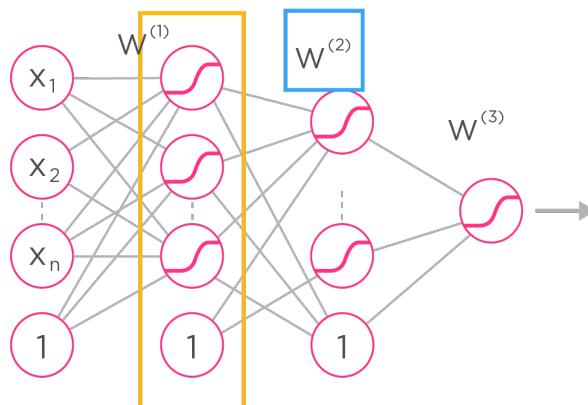
$$\hat{y} =$$

$$\sigma \circ W^{(1)}(x)$$

- • • • •
- +
- • • • •
- + REDES NEURAIS ARTIFICIAIS
- +
-

Redes Neurais Multicamadas - Topologia FeedForward

Multi-layer Perceptron



PREDICTION

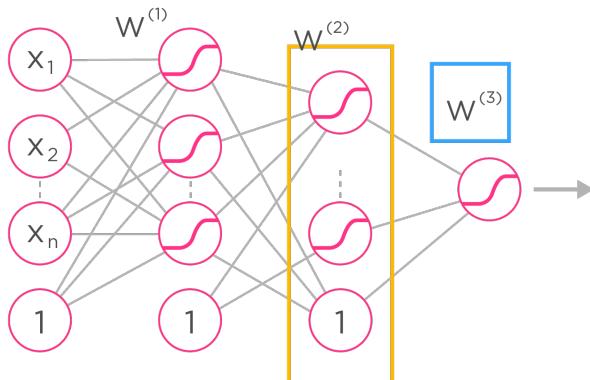
$$\hat{y} =$$

$$\sigma \circ W^{(2)} \circ \sigma \circ W^{(1)}(x)$$

REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Topologia FeedForward

Multi-layer Perceptron



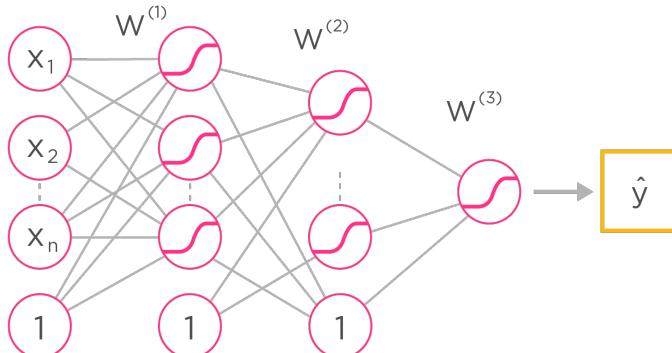
PREDICTION

$$\hat{y} = \sigma \circ W^{(3)} \circ \sigma \circ W^{(2)} \circ \sigma \circ W^{(1)}(x)$$

REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Topologia FeedForward

Multi-layer Perceptron



PREDICTION

$$\hat{y} = \sigma \circ W^{(3)} \circ \sigma \circ W^{(2)} \circ \sigma \circ W^{(1)}(x)$$

- • • • •
- • • • •
- + REDES NEURAIS ARTIFICIAIS

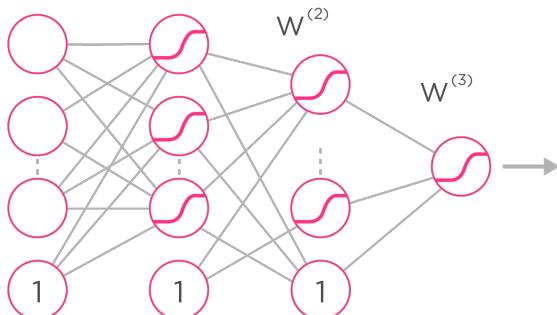
Redes Neurais Multicamadas - Aprendizado

- Mas, agora que temos diversos neurônios na rede neural, como fazer para ajustar todos os seus pesos?

REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Aprendizado

Multi-layer Perceptron

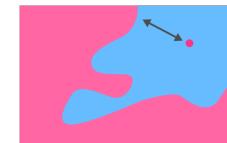


PREDICTION

$$y = \sigma \circ W^{(3)} \circ \sigma \circ W^{(2)} \circ \sigma \circ W^{(1)}(x)$$

ERROR FUNCTION

$$E(W) = -\frac{1}{m} \sum_{i=1}^m y_i \ln(\hat{y}_i) + (1 - y_i) \ln(1 - \hat{y}_i)$$



<https://www.youtube.com/watch?v=GdNmDQRdL2o>

- • • • •
- +
- • • • •
- + REDES NEURAIS ARTIFICIAIS
- +
-

Redes Neurais Multicamadas - Aprendizado

- Como vocês fariam para construir um algoritmo para um robô que quer escalar o pico mais alto/baixo de um conjunto de montanhas?



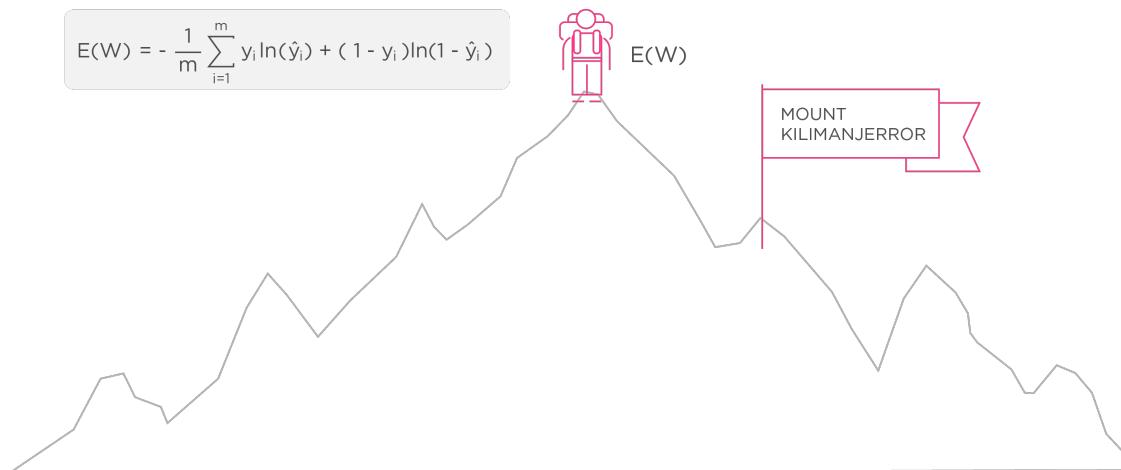
- • • • •
- +
- • • • •
- + REDES NEURAIS ARTIFICIAIS
- +
-

Redes Neurais Multicamadas - Aprendizado

- Descida do Gradiente

Gradient Descent

$$E(W) = - \frac{1}{m} \sum_{i=1}^m y_i \ln(\hat{y}_i) + (1 - y_i) \ln(1 - \hat{y}_i)$$



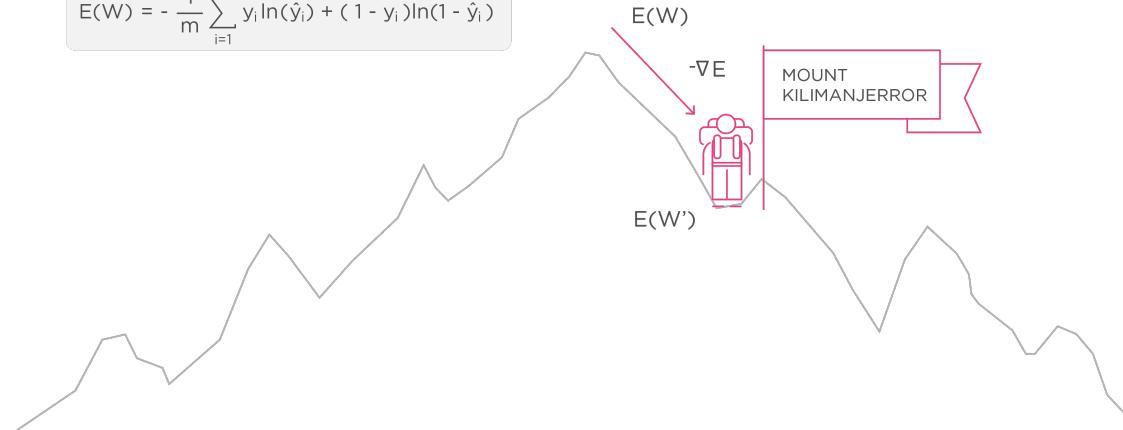
- • • • •
- • • • •
- + REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Aprendizado

- Descida do Gradiente

Gradient Descent

$$E(W) = -\frac{1}{m} \sum_{i=1}^m y_i \ln(\hat{y}_i) + (1 - y_i) \ln(1 - \hat{y}_i)$$



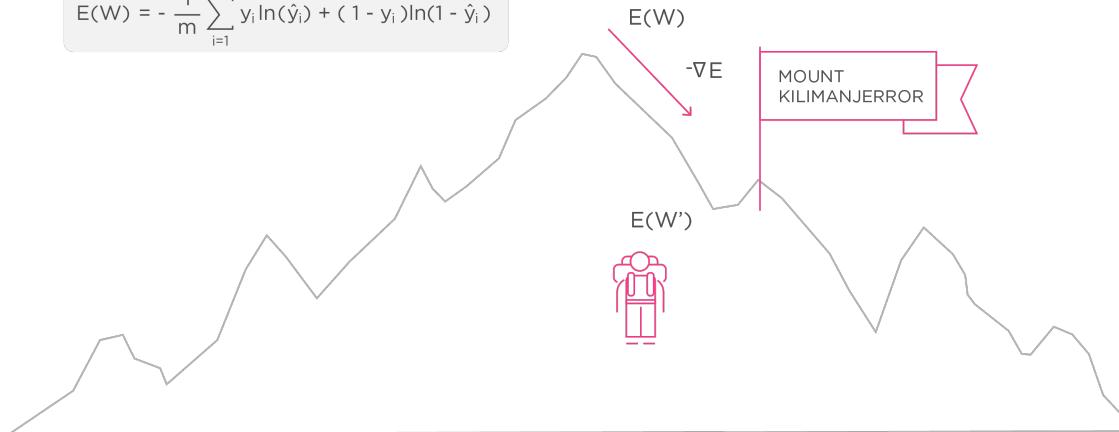
• • • • • + • • • • • . • + REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Aprendizado

- Descida do Gradiente

Gradient Descent

$$E(W) = -\frac{1}{m} \sum_{i=1}^m y_i \ln(\hat{y}_i) + (1 - y_i) \ln(1 - \hat{y}_i)$$



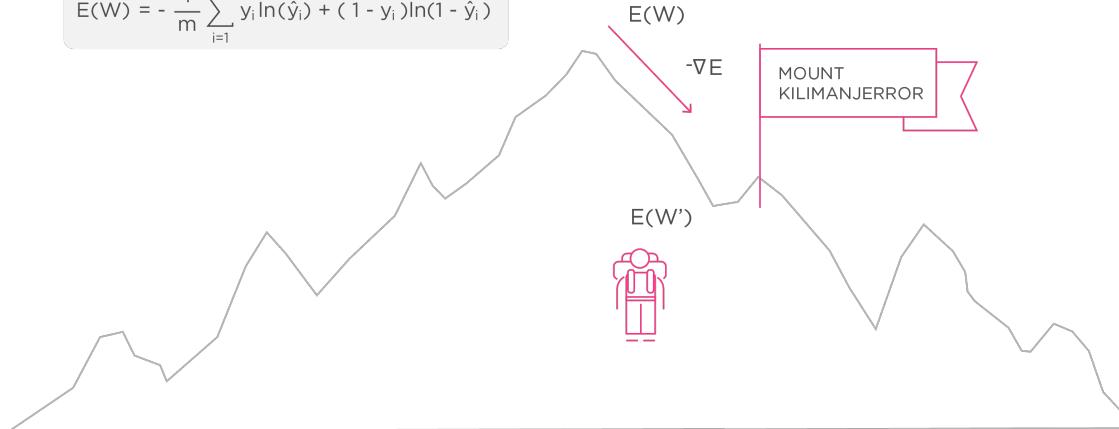
• • • • • + • • • • • . • + REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Aprendizado

- Descida do Gradiente

Gradient Descent

$$E(W) = -\frac{1}{m} \sum_{i=1}^m y_i \ln(\hat{y}_i) + (1 - y_i) \ln(1 - \hat{y}_i)$$



- • • • •
- • • • •
- + REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Aprendizado

- Descida do Gradiente

- A Descida do Gradiente é um algoritmo de otimização usado para encontrar os valores de parâmetros (coeficientes ou se preferir w e b - **weight** e **bias**) de uma função que minimizam uma função de custo. A Descida do Gradiente é melhor usada quando os parâmetros não podem ser calculados analiticamente (por exemplo, usando álgebra linear) e devem ser pesquisados por um algoritmo de otimização.

• • • • • + • • • • • . • + REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Aprendizado

- **Descida do Gradiente**

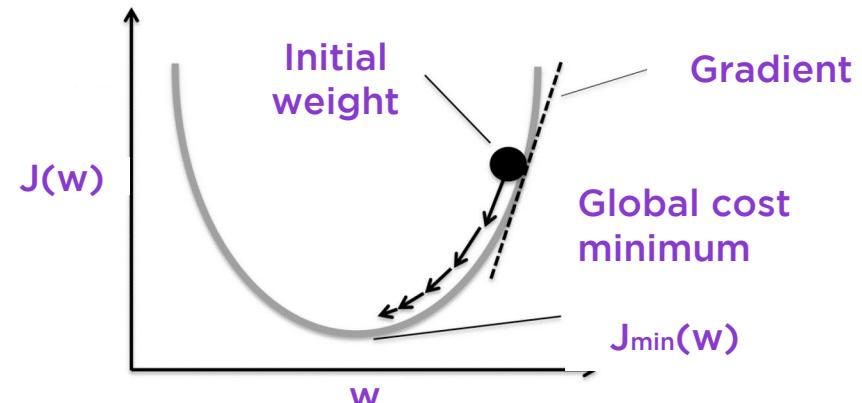
1. Escolha valores iniciais arbitrários para os pesos.
2. Calcule o gradiente da função de custo com respeito a cada peso.
3. Mude os pesos tal que haja um deslocamento pequeno na direção de $-G$.
 $-G \Rightarrow$ Maior taxa de diminuição do erro
4. Repita os passos 2 e 3 até que o erro se aproxime de zero.

REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Aprendizado

- **Descida do Gradiente**

- Uma busca baseada no gradiente descendente determina o vetor de peso que minimiza $J(w)$, começando com um vetor arbitrário inicial;
- Modifique o vetor repetidamente em passos pequenos;



• • • • + • • • . . . • + REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Aprendizado

- Descida do Gradiente - Como calcular:

custo = avaliar (f (coeficiente))

delta = derivado (custo)

coeficiente = coeficiente - (alfa * delta)

- Nossos coeficientes são o vetor de peso w e b;
- Onde f é a função de ativação;
- A função Avaliar calcula o erro da função em relação ao coeficiente;
- A derivada de f previamente calculada retorna a inclinação delta;
- Alfa (ou eta) é a taxa de aprendizagem.

• • • • • + • • • • • . • + REDES NEURAIS ARTIFICIAIS

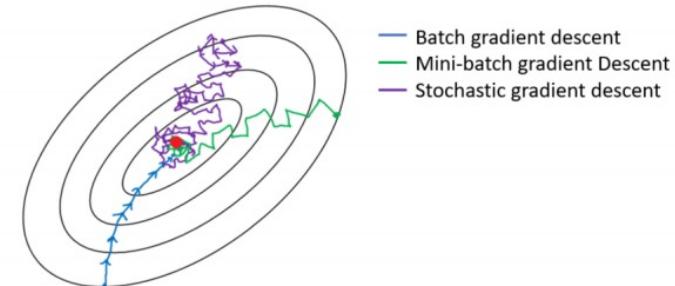
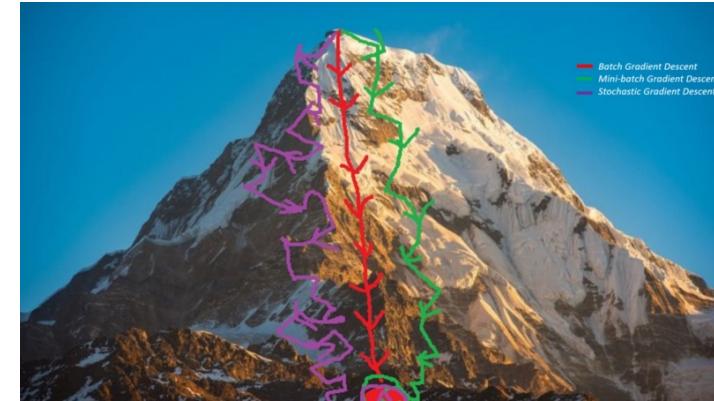
Redes Neurais Multicamadas - Aprendizado

- **Descida do Gradiente - Diferentes algoritmos:**
 - Descida do Gradiente em “Batch”
 - Descida do Gradiente Estocástico
 - Descida do Gradiente em “Mini-Batch”
 - Descida do Gradiente com Adam Optimizer

REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Aprendizado

- Descida do Gradiente - Diferentes algoritmos:
 - Descida do Gradiente em “Batch”
 - Descida do Gradiente Estocástico
 - Descida do Gradiente em “Mini-Batch”



REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Aprendizado

- Descida do Gradiente -
Diferentes algoritmos:

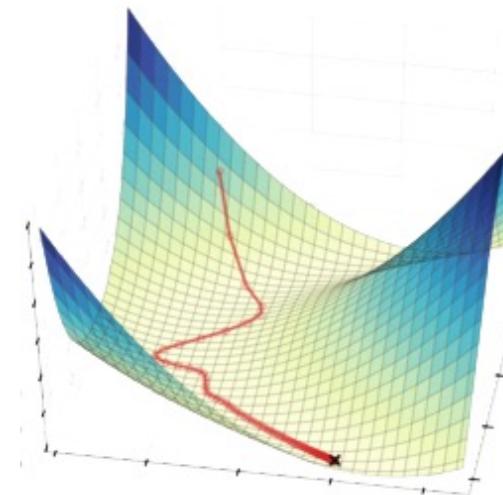
- Descida do Gradiente em “Batch”
- Descida do Gradiente Estocástico
- Descida do Gradiente em “Mini-Batch”

METHOD	ACCURACY	TIME	MEMORY USAGE	ONLINE LEARNING
Batch gradient	○	Slow	High	✗
Stochastic gradient descent	△	High	Low	○
Mini-batch gradient descent	○	Medium	Medium	○

REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Aprendizado

- Descida do Gradiente - Diferentes algoritmos:
 - Descida do Gradiente com Adam Optimizer



<https://www.youtube.com/watch?v=qPKKtvkVAjY>

- • • • •
- • • • •
- + REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Aprendizado

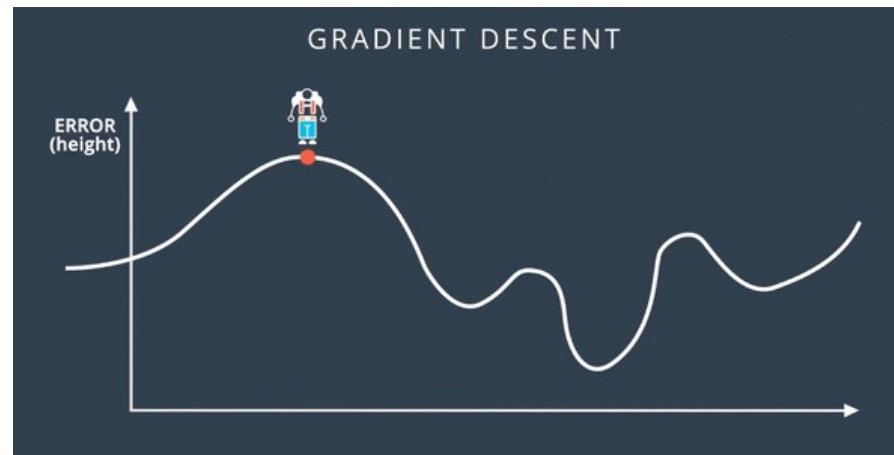
- **Descida do Gradiente - Diferentes algoritmos:**
 - Descida do Gradiente com Adam Optimizer (*Adaptive Moment Estimation*)

De maneira resumida, o Adam Optimizer consegue aumentar (e também diminuir) a velocidade com que “nos deslocamos” ao longo do gradiente, baseado nos passos anteriores.

- • • • •
- +
- • • • •
- + REDES NEURAIS ARTIFICIAIS
- +
-

Redes Neurais Multicamadas - Aprendizado

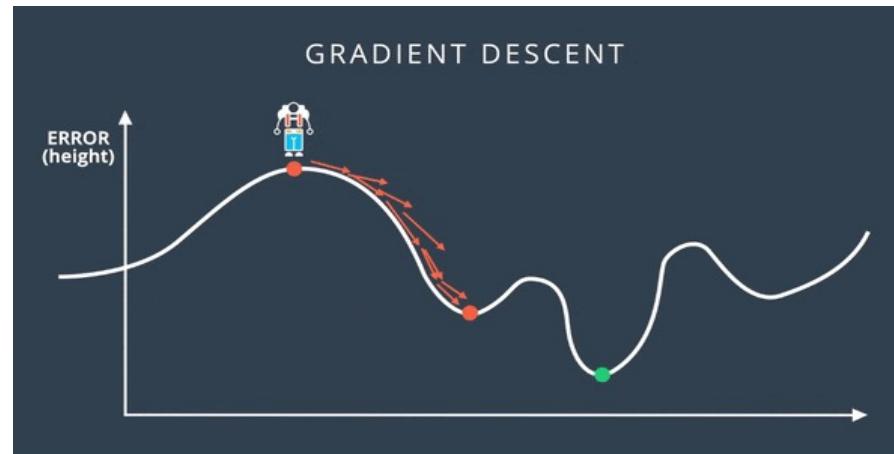
- Descida do Gradiente - Diferentes algoritmos:
 - Descida do Gradiente com Adam Optimizer (*Adaptive Moment Estimation*)



- • • • •
- +
- • • • •
- + REDES NEURAIS ARTIFICIAIS
- +
-

Redes Neurais Multicamadas - Aprendizado

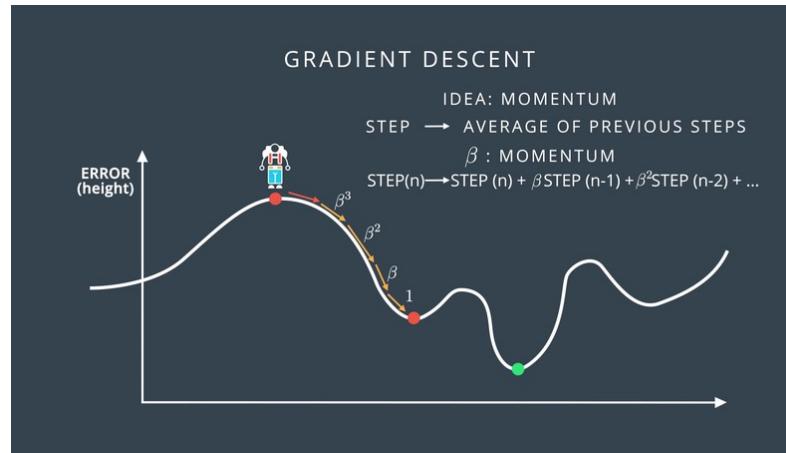
- Descida do Gradiente - Diferentes algoritmos:
 - Descida do Gradiente com Adam Optimizer (*Adaptive Moment Estimation*)



- • • • •
- +
- • • • •
- + REDES NEURAIS ARTIFICIAIS
- +
-

Redes Neurais Multicamadas - Aprendizado

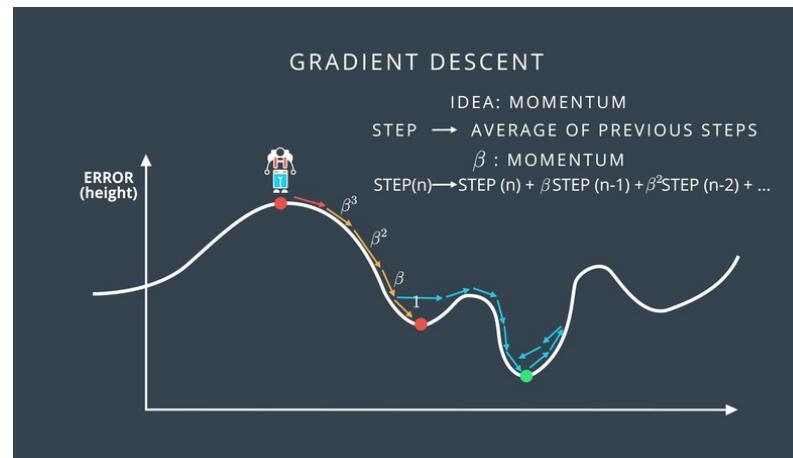
- Descida do Gradiente - Diferentes algoritmos:
 - Descida do Gradiente com Adam Optimizer (*Adaptive Moment Estimation*)



- • • • •
- +
- • • • •
- + REDES NEURAIS ARTIFICIAIS
- +
-

Redes Neurais Multicamadas - Aprendizado

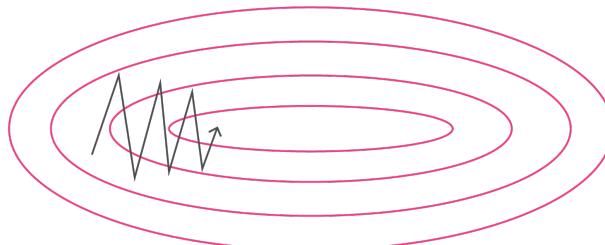
- Descida do Gradiente - Diferentes algoritmos:
 - Descida do Gradiente com Adam Optimizer (*Adaptive Moment Estimation*)



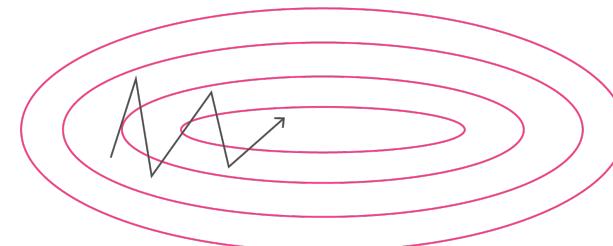
- • • • •
- +
- • • • •
- + REDES NEURAIS ARTIFICIAIS
- +
-

Redes Neurais Multicamadas - Aprendizado

- **Descida do Gradiente - Diferentes algoritmos:**
 - Descida do Gradiente com Adam Optmizer (Adaptive Moment Estimation)



SGD sem momentum



SGD com momentum

• • • • • + • • • • • . • + REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Aprendizado

- Descida do Gradiente - Diferentes algoritmos:

- Vídeo

REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Aprendizado

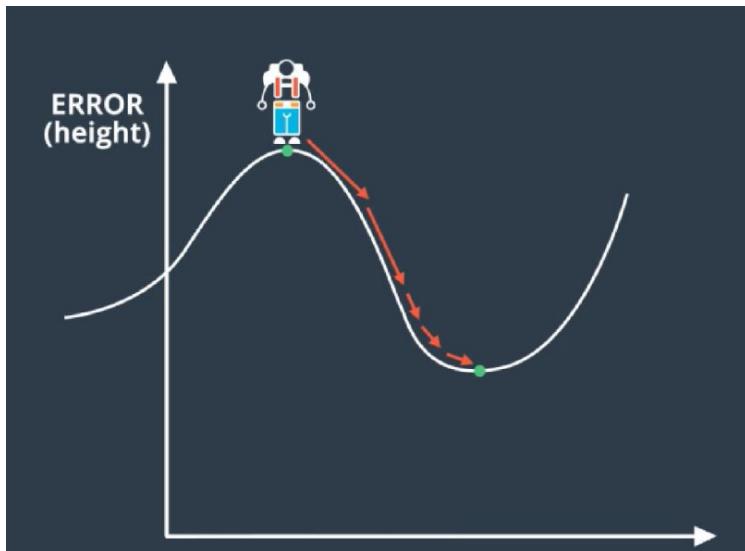
- **Descida do Gradiente - Observações:**

- Dificuldades práticas da estratégia do gradiente descendente:
 1. Convergência para um mínimo local pode ser muito lenta (várias iterações).
 2. Se existir vários mínimos locais na superfície de erro, não haverá garantia que o algoritmo encontrará o mínimo.

- • • •
- +
- • • •
- + REDES NEURAIS ARTIFICIAIS
- +
-

Redes Neurais Multicamadas - Aprendizado

- Taxa de aprendizado adaptativa:



Decreasing Learning Rate

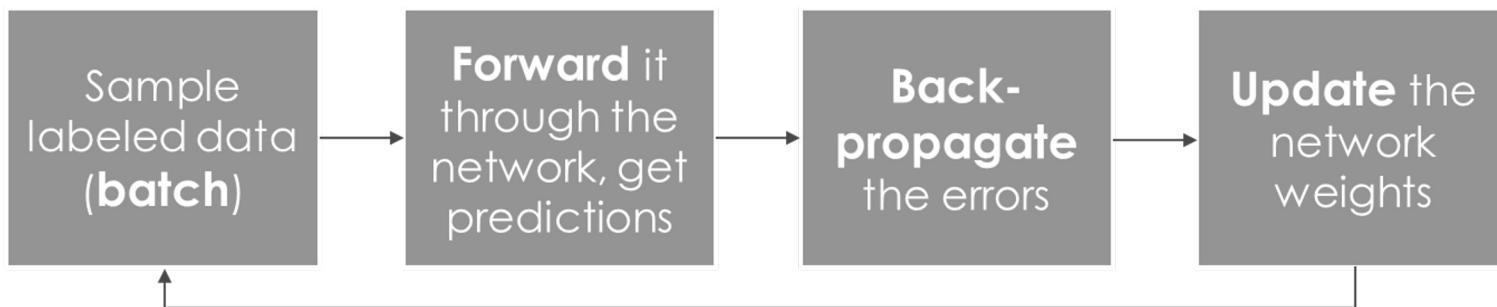
Rule:

- If steep: long steps
- If plain: small steps

REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Aprendizado

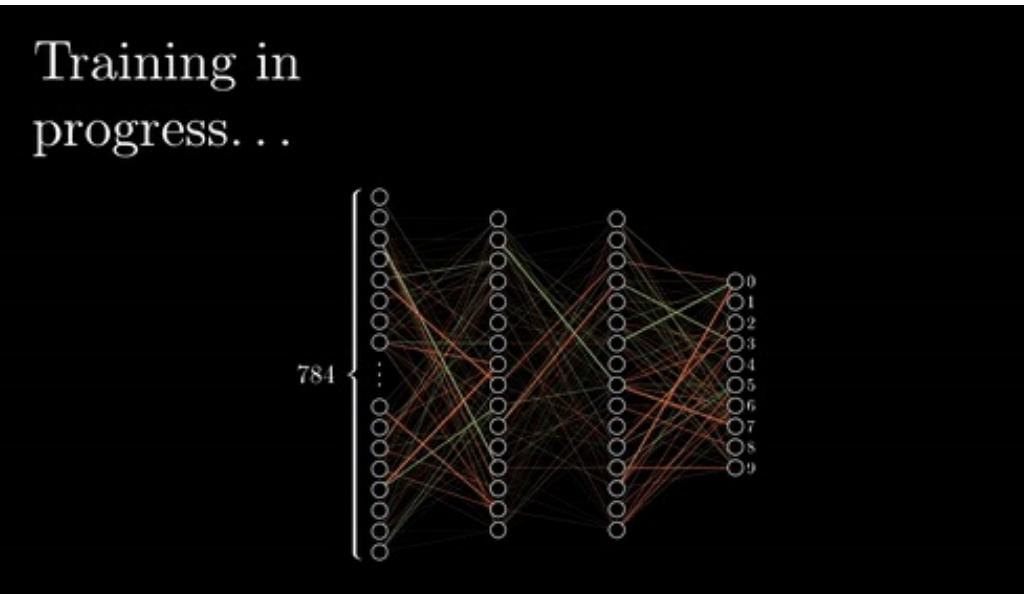
- Backpropagation:



- • • • •
- • • • •
- + REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Aprendizado

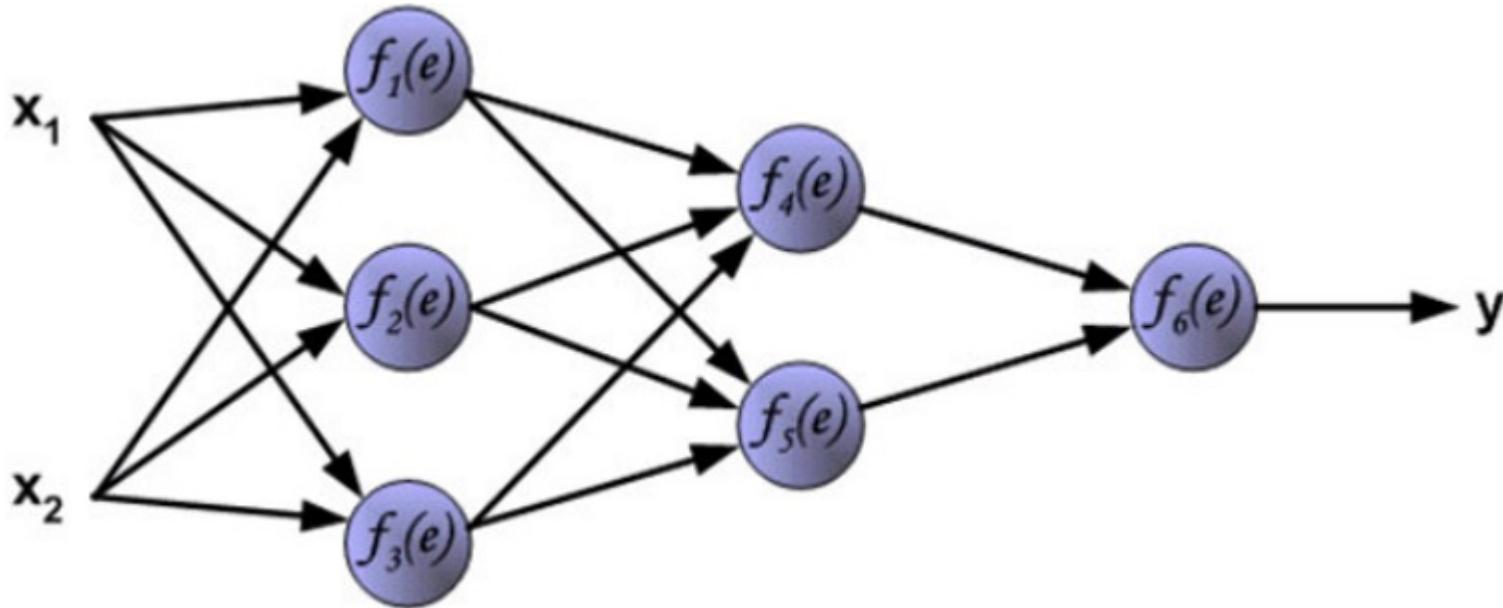
- Backpropagation:



- • • • •
- • • • •
- + REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Aprendizado

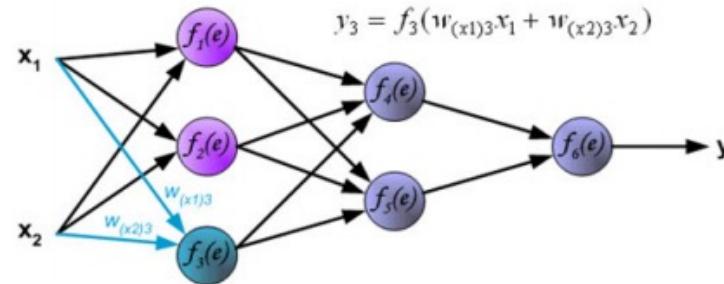
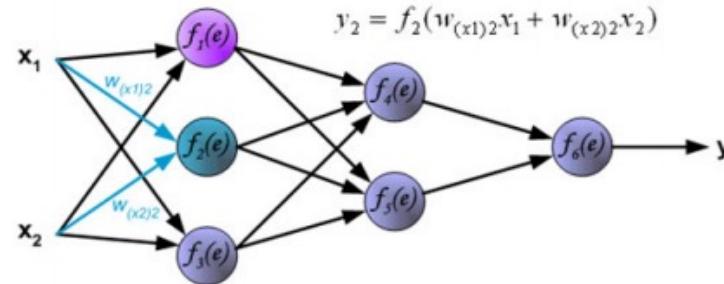
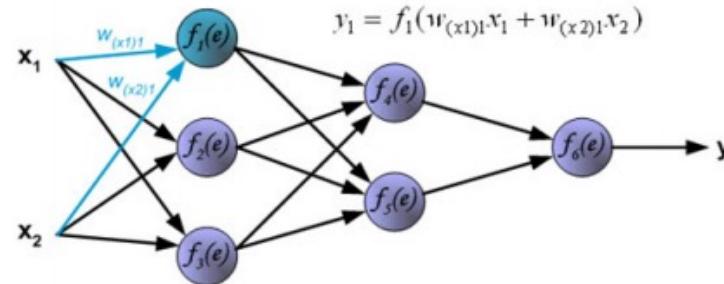
- Backpropagation:



- • • •
- +
- • • • .
- + REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Aprendizado - Backpropagation

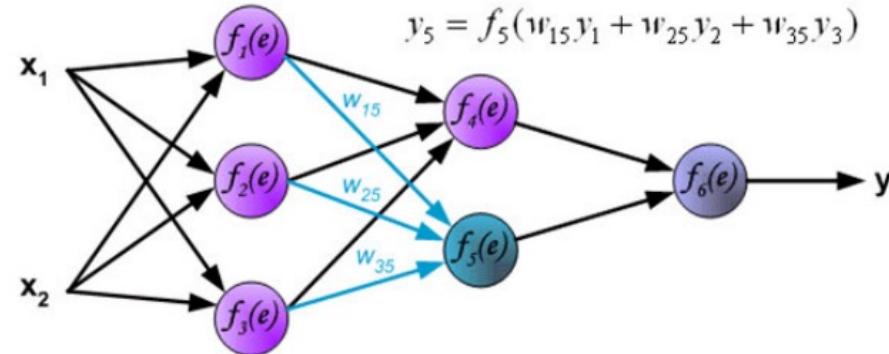
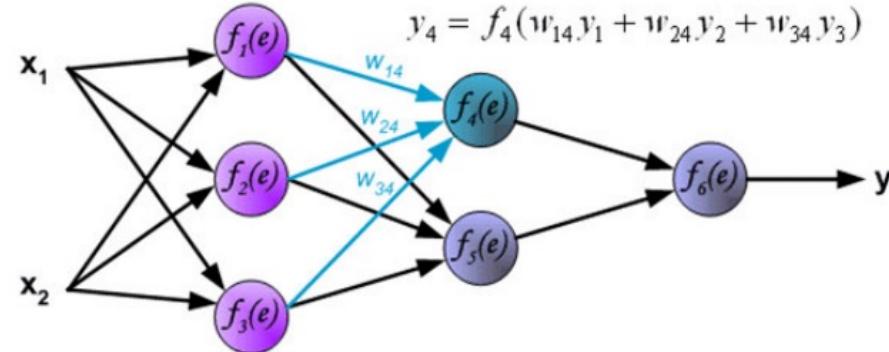
- Passo Forward



REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Aprendizado - Backpropagation:

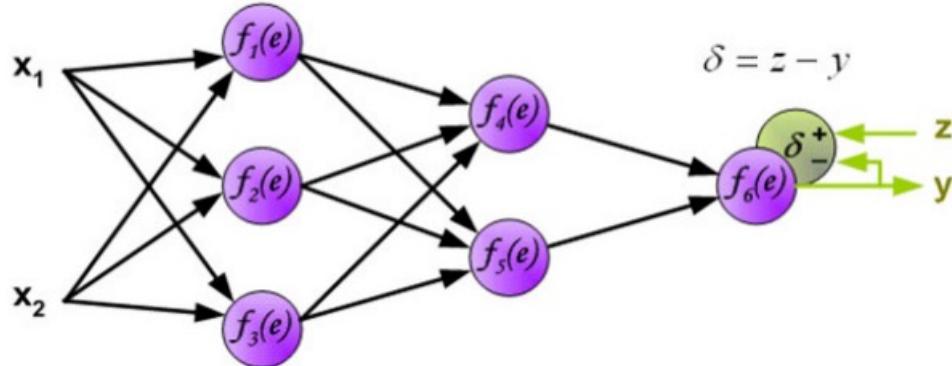
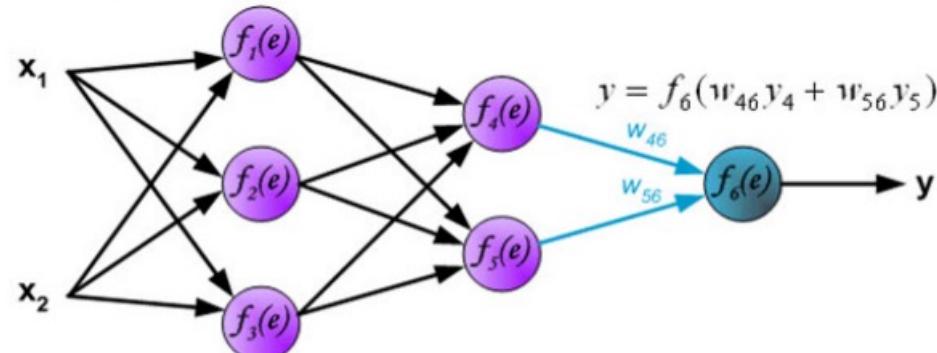
- Passo Forward



REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Aprendizado - Backpropagation

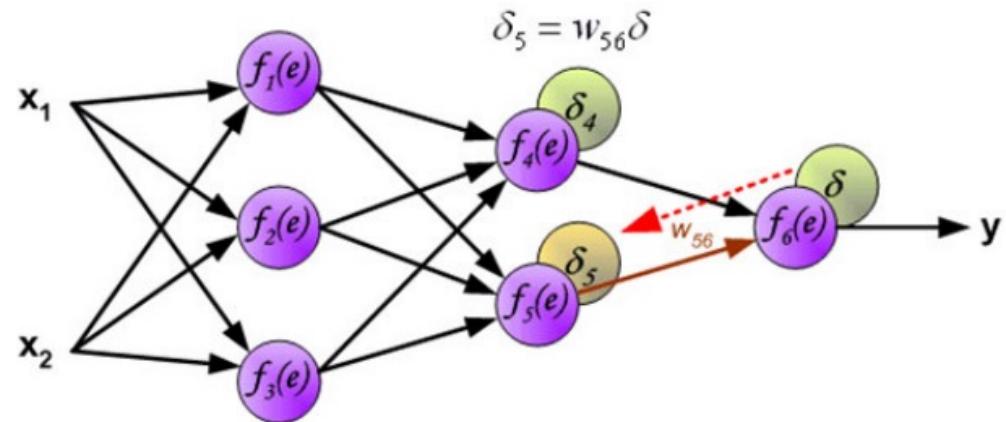
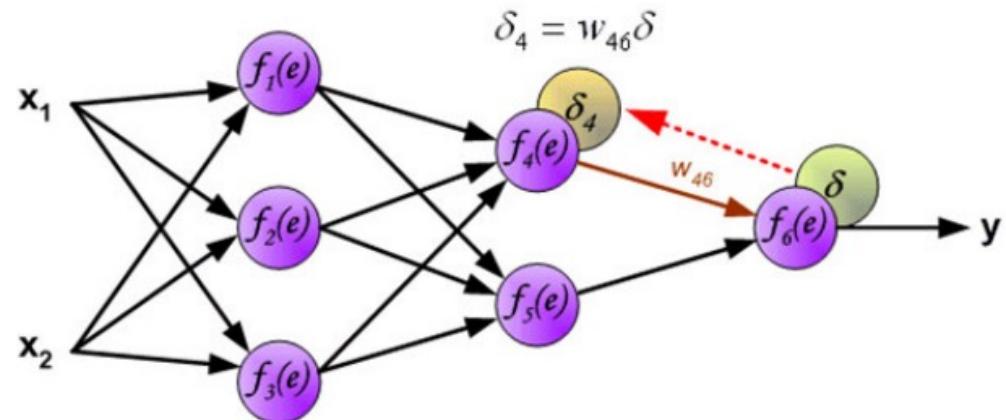
- Passo Forward

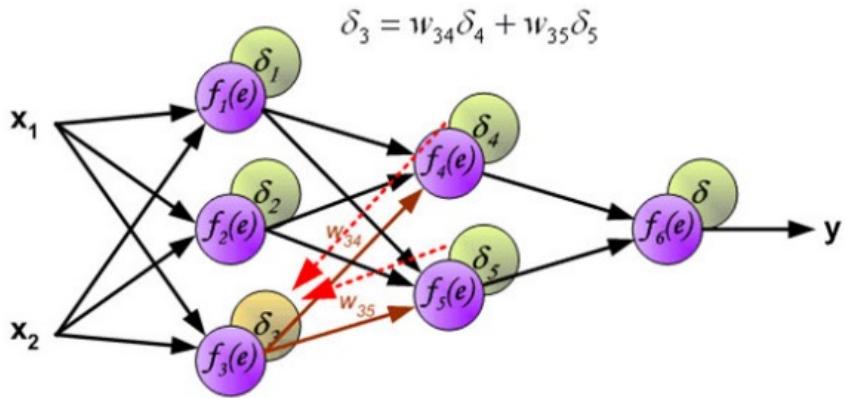
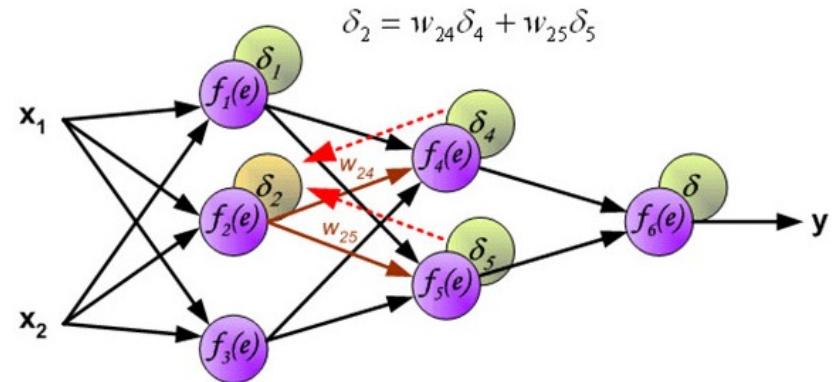
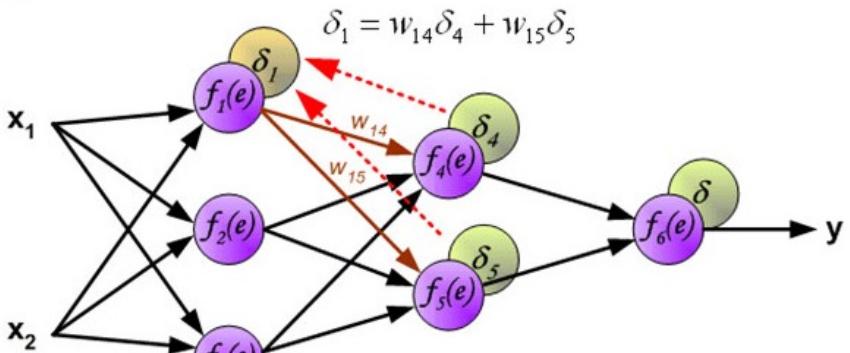


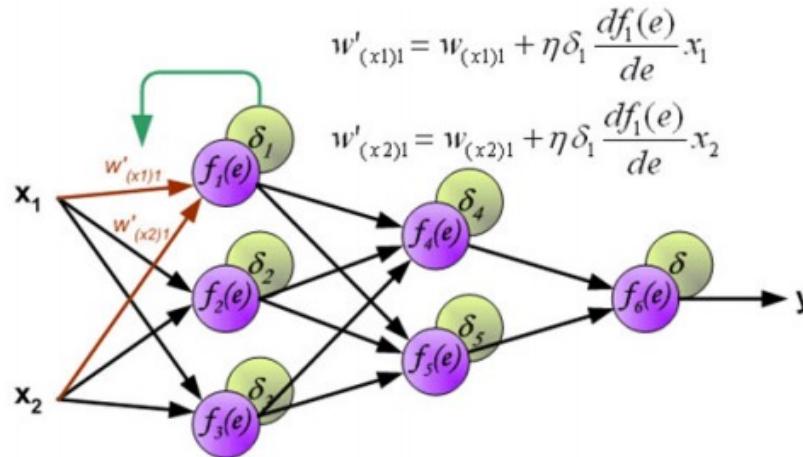
REDES NEURAIS ARTIFICIAIS

Redes Neurais Multicamadas - Aprendizado - Backpropagation

- Passo Backward:

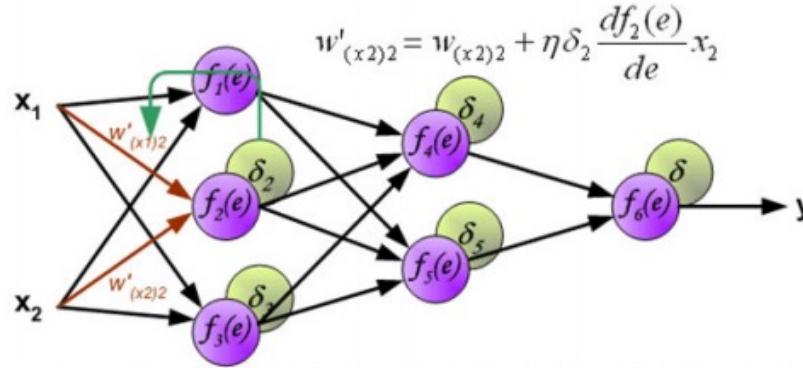






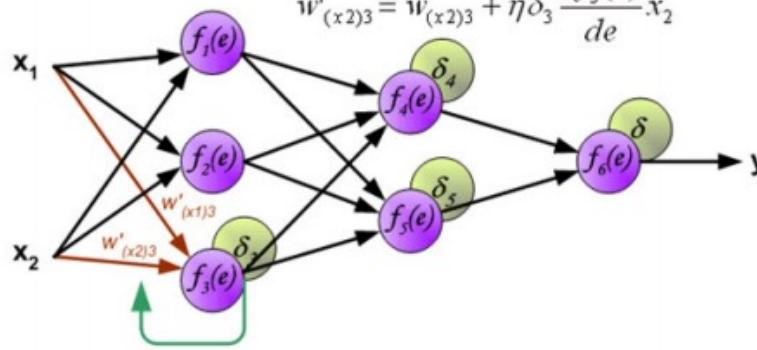
$$w'_{(x1)2} = w_{(x1)2} + \eta \delta_2 \frac{df_2(e)}{de} x_1$$

$$w'_{(x2)2} = w_{(x2)2} + \eta \delta_2 \frac{df_2(e)}{de} x_2$$



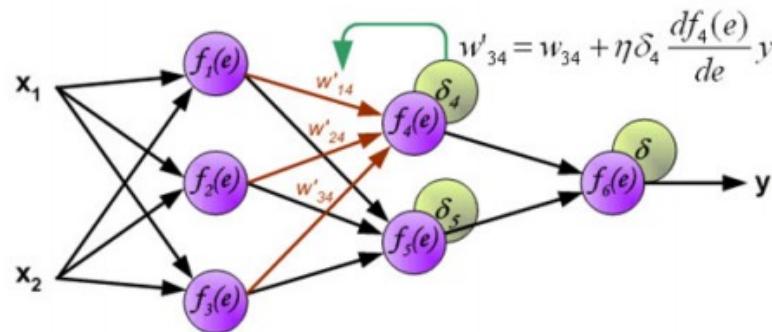
$$w'_{(x1)3} = w_{(x1)3} + \eta \delta_3 \frac{df_3(e)}{de} x_1$$

$$w'_{(x2)3} = w_{(x2)3} + \eta \delta_3 \frac{df_3(e)}{de} x_2$$



$$w'_{14} = w_{14} + \eta \delta_4 \frac{df_4(e)}{de} y_1$$

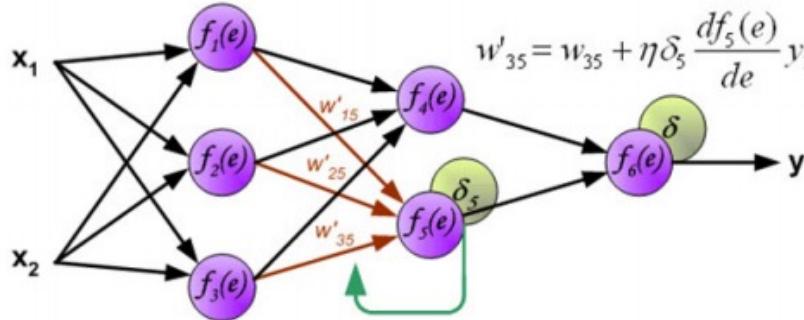
$$w'_{24} = w_{24} + \eta \delta_4 \frac{df_4(e)}{de} y_2$$



$$w'_{15} = w_{15} + \eta \delta_5 \frac{df_5(e)}{de} y_1$$

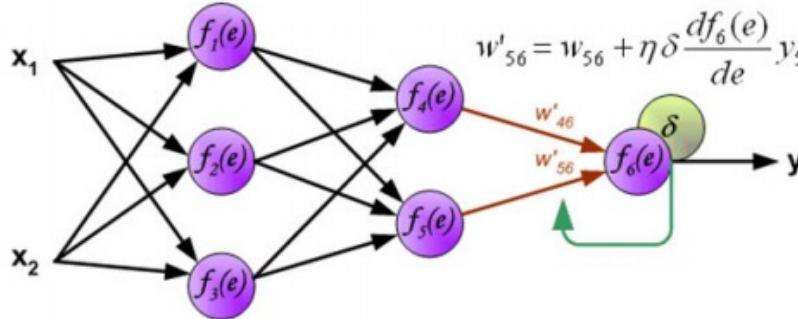
$$w'_{25} = w_{25} + \eta \delta_5 \frac{df_5(e)}{de} y_2$$

$$w'_{35} = w_{35} + \eta \delta_5 \frac{df_5(e)}{de} y_3$$



$$w'_{46} = w_{46} + \eta \delta \frac{df_6(e)}{de} y_4$$

$$w'_{56} = w_{56} + \eta \delta \frac{df_6(e)}{de} y_5$$



REDES NEURAIS
ARTIFICIAIS

HANDS ON #1:
TENSORFLOW PLAYGROUND

REDES NEURAIS ARTIFICIAIS

HANDS ON #2:
**REDE NEURAL MLP PARA
CLASSIFICAÇÃO**

REDES NEURAIS
ARTIFICIAIS

HANDS ON #3:
**REDE NEURAL MLP PARA
REGRESSÃO**

OBRIGADO

FIAP

Copyright © 2020 | Professor Felipe Gustavo Silva Teodoro

Todos os direitos reservados. A reprodução ou divulgação total ou parcial deste documento é expressamente proibida sem consentimento formal, por escrito, do professor(a)/autor(a).

FIAP