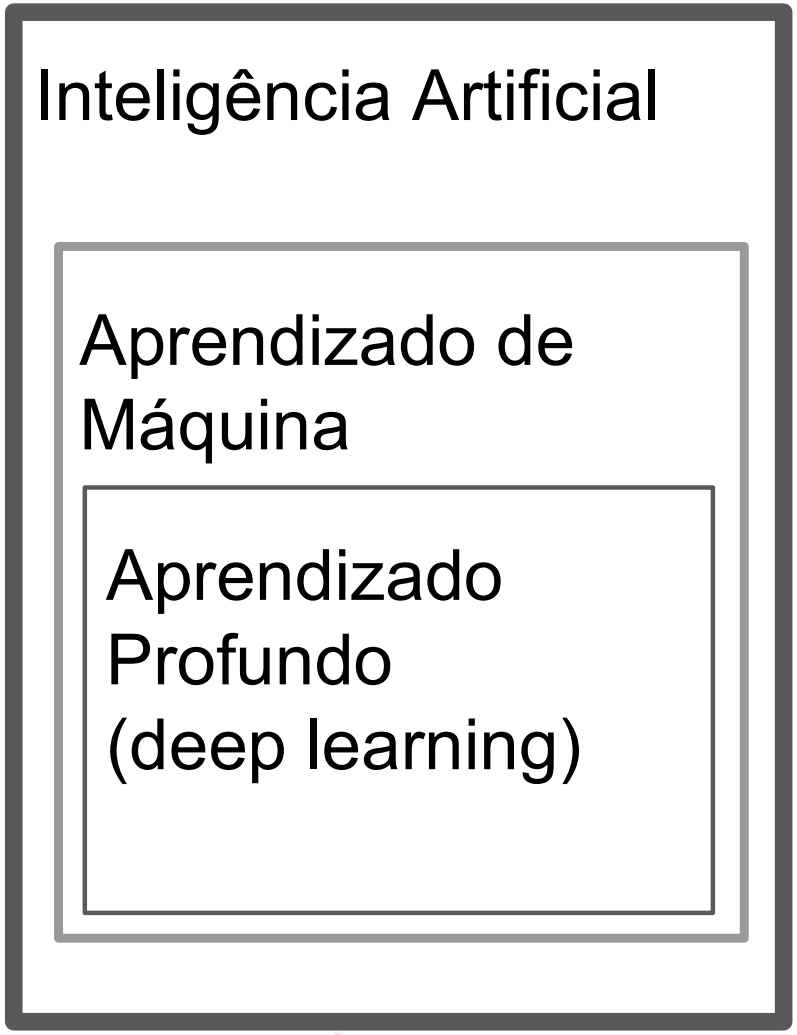
Redes Neurais Artificiais

Prof. Gustavo Willam Pereira

INSTITUTO FEDERAL
Sudeste de Minas Gerais

Conteúdo

1. Introdução
2. Neurônio artificial e funções de ativação
3. Redes multicamadas
4. Algoritmo *backpropagation*
5. Gradient descente (descida do gradiente)
6. Avaliação das RNAs



INSTITUTO FEDERAL
Sudeste de Minas Gerais

O que é inteligência

Na ficção...

Inteligência...

Normalmente, é definida como a capacidade de:

- Adquirir e aplicar conhecimento.
- Razão dedutiva.
- Criatividade.

O que é inteligência artificial?

- IA é a capacidade de realizar as funções que são tipicamente associados com a inteligência humana.

INSTITUTO FEDERAL
Sudeste de Minas Gerais

O que são redes neurais artificiais?

- São sistemas computacionais massivos e paralelos, com propensão de armazenar conhecimento experimental e torna-lo disponível para uso.

- As redes neurais artificiais tentam imitar o funcionamento do cérebro biológico representando-o artificialmente.

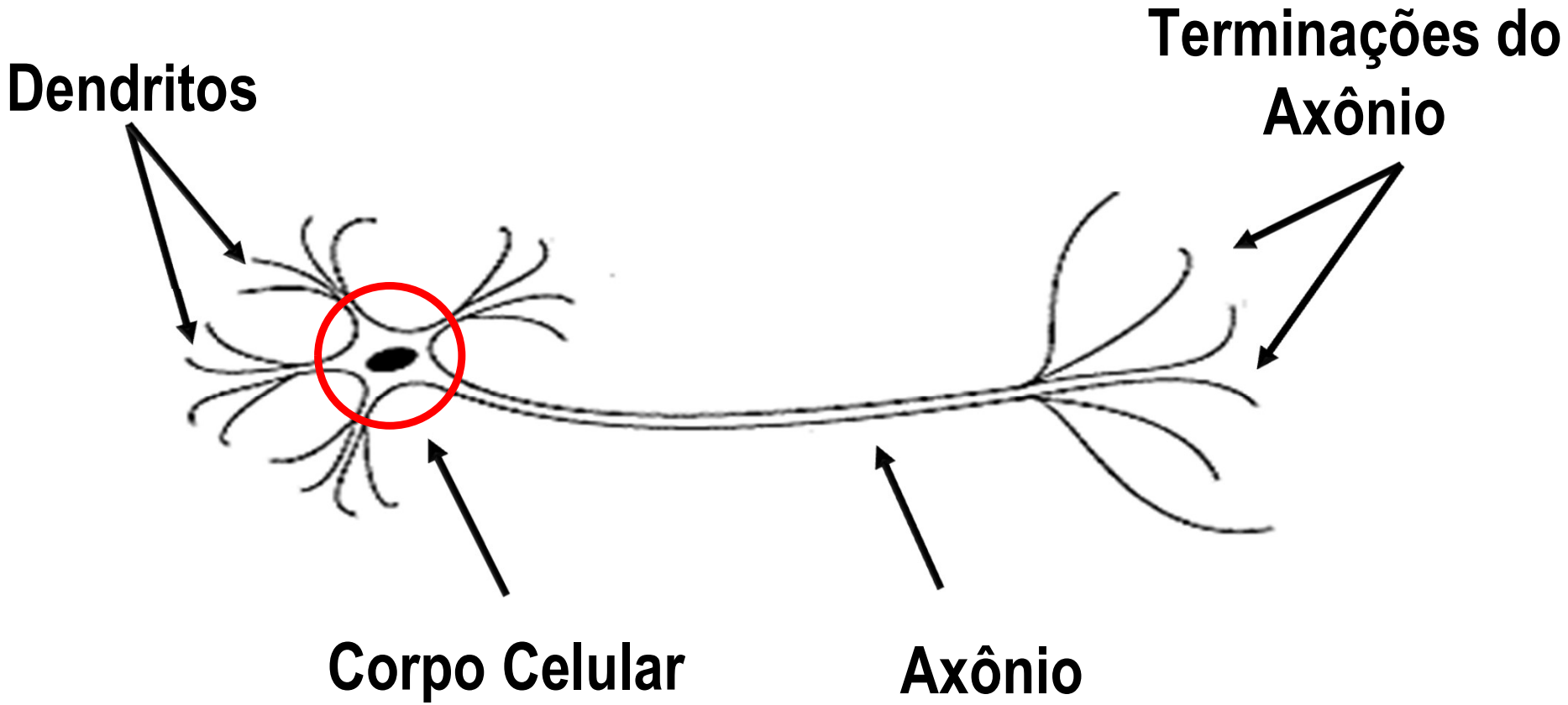
Por que aprender a aplicar RNA?

- Capacidade de aprendizado e generalização.
- Tolerância a falhas devido ao elevado número de conexões entre os neurônios artificiais.
- Tolerância a ruídos nos dados (dados viesados ou outliers)
- Superioridade em relação às estimativas de modelos de regressão.

Características das RNA´s

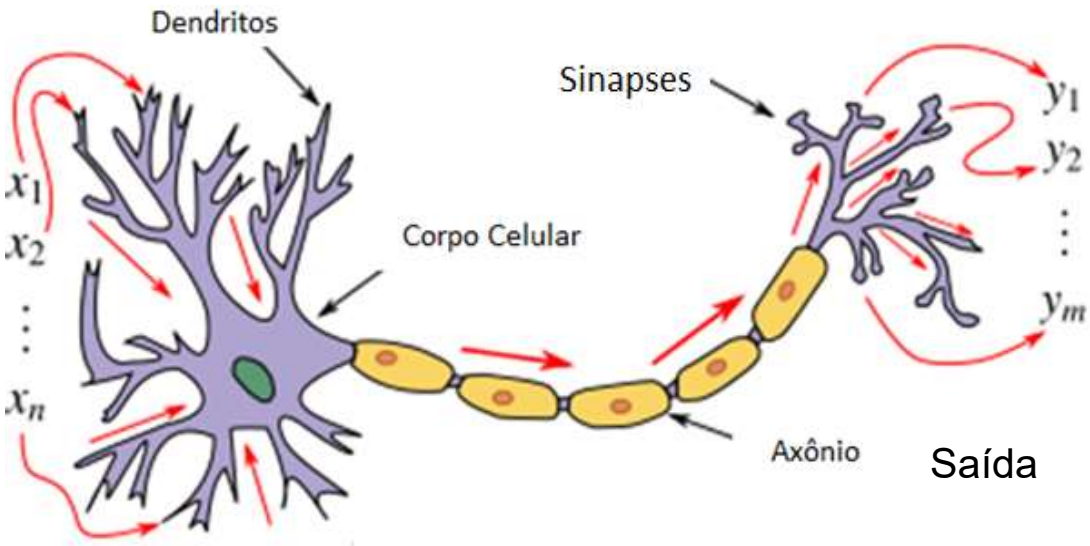
- Lógica de decisão baseada em pesos estatísticos.
- Conseguir distinguir características semelhantes após um processo de “aprendizado”.
- Capacidade de generalizar o que foi aprendido.

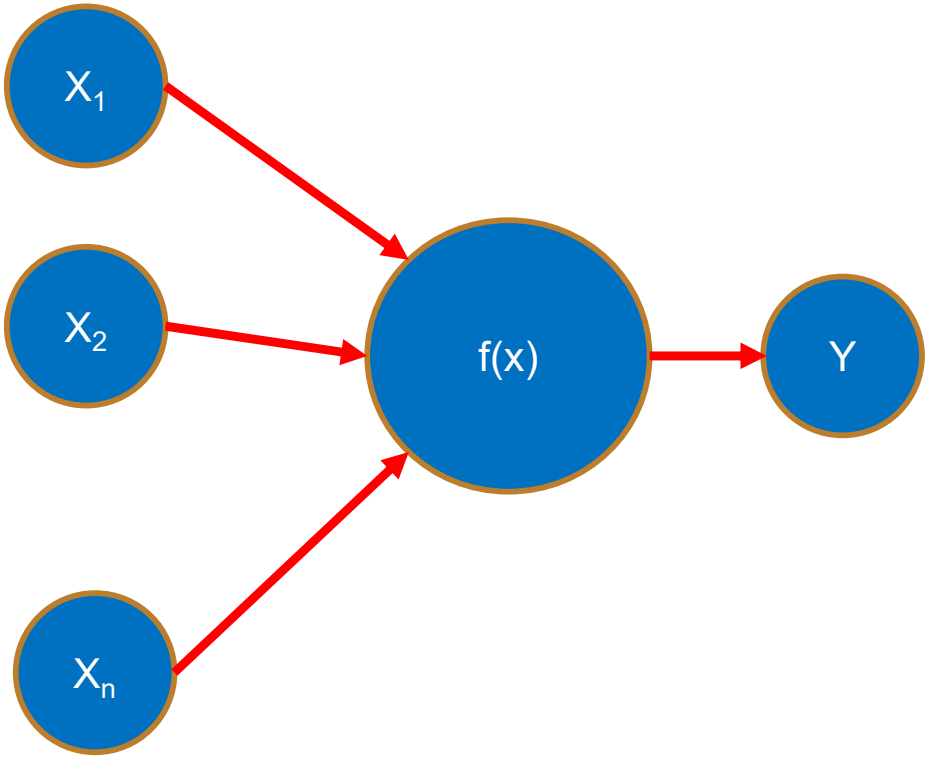
Inspiração biológica



INSTITUTO FEDERAL
Sudeste de Minas Gerais

O Neurônio

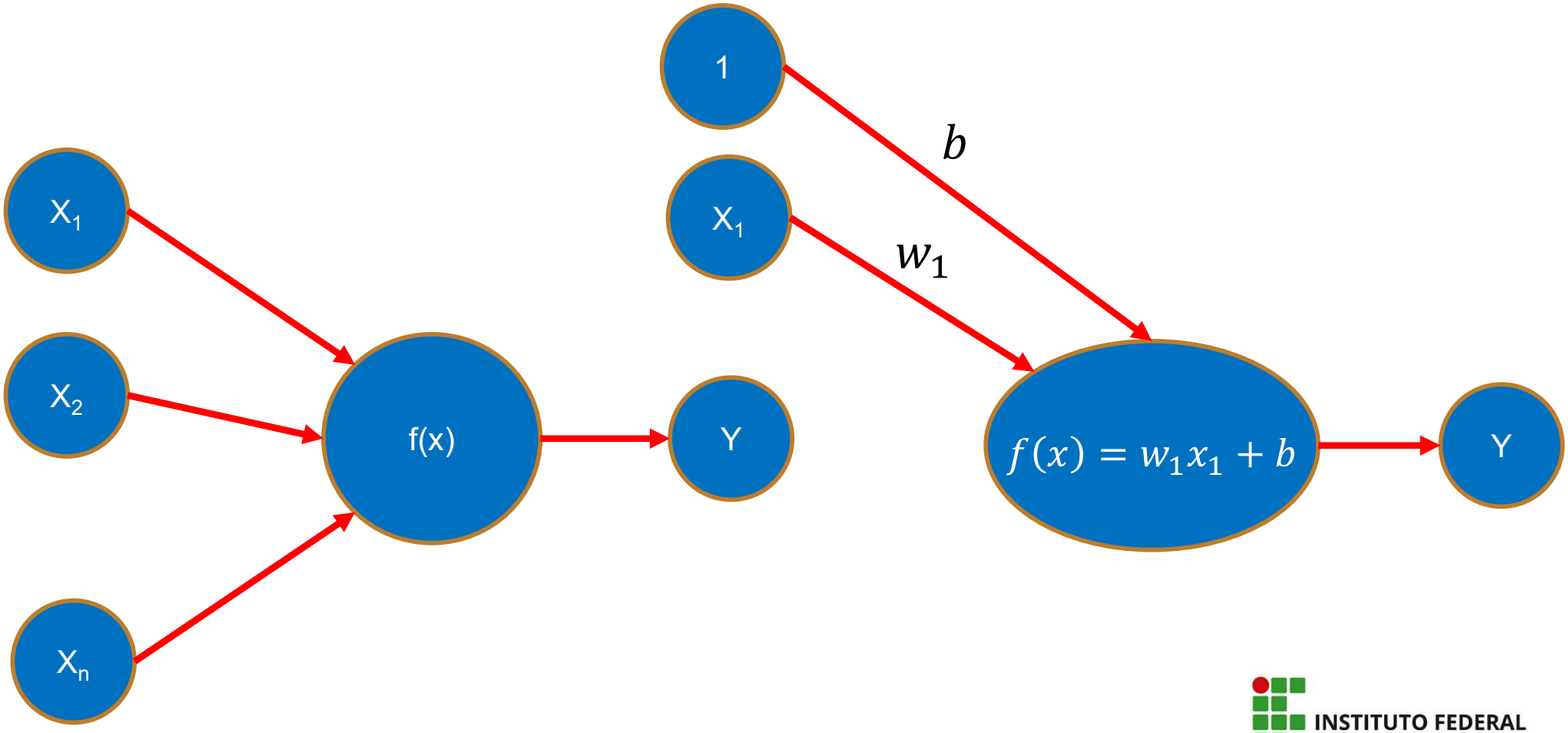




INSTITUTO FEDERAL
Sudeste de Minas Gerais

O Neurônio

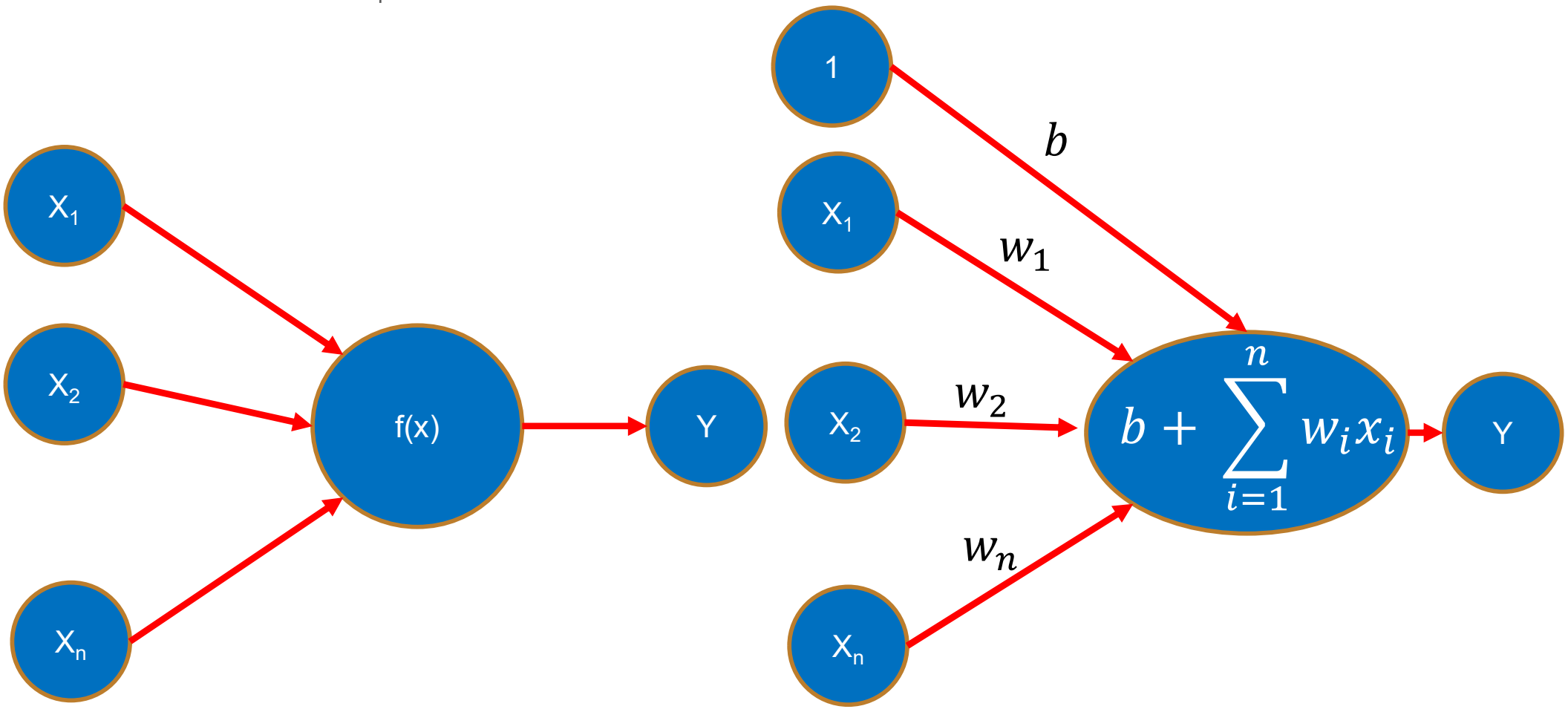
Forma mais simples de uma rede neural



INSTITUTO FEDERAL
Sudeste de Minas Gerais

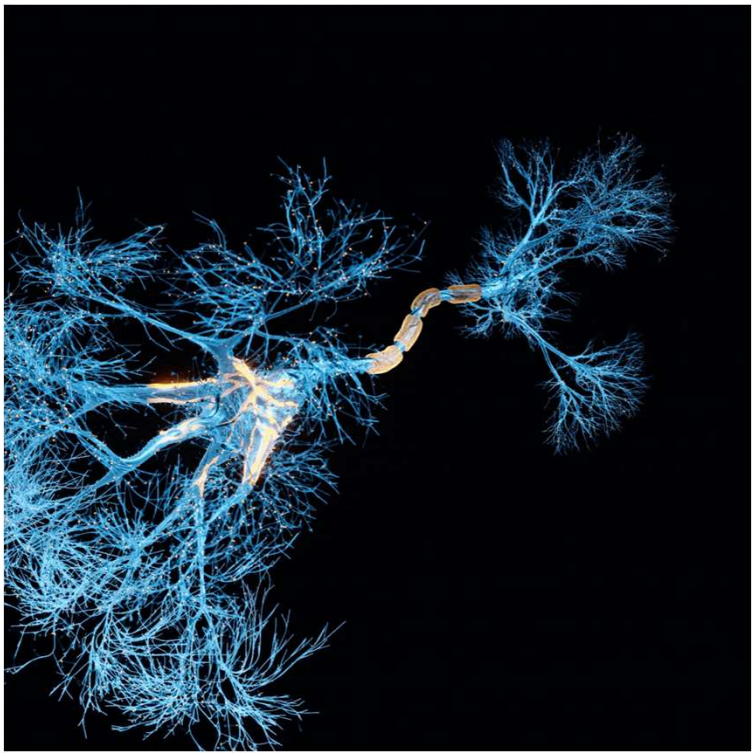
O Neurônio

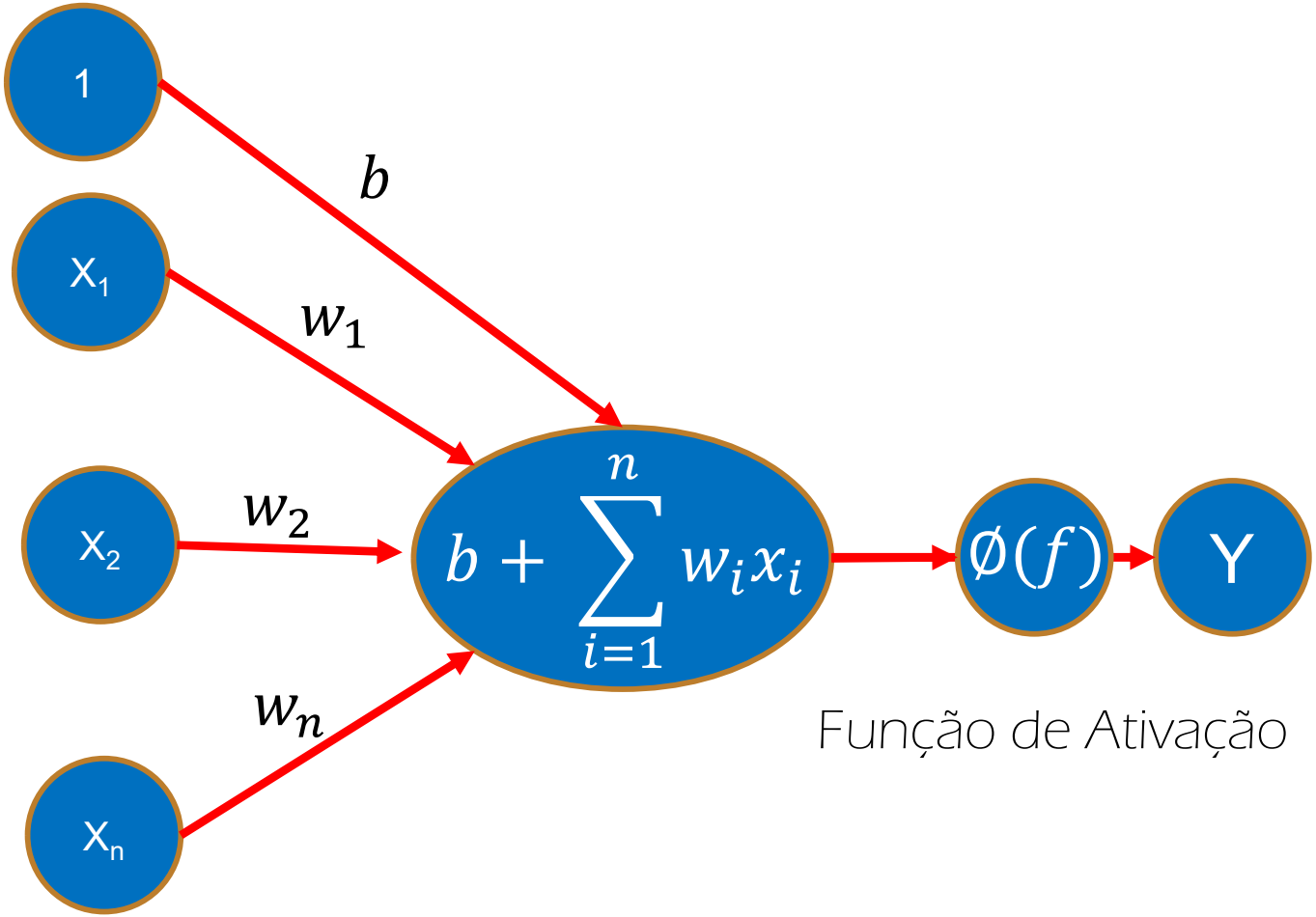
Forma mais simples de uma rede neural



INSTITUTO FEDERAL
Sudeste de Minas Gerais

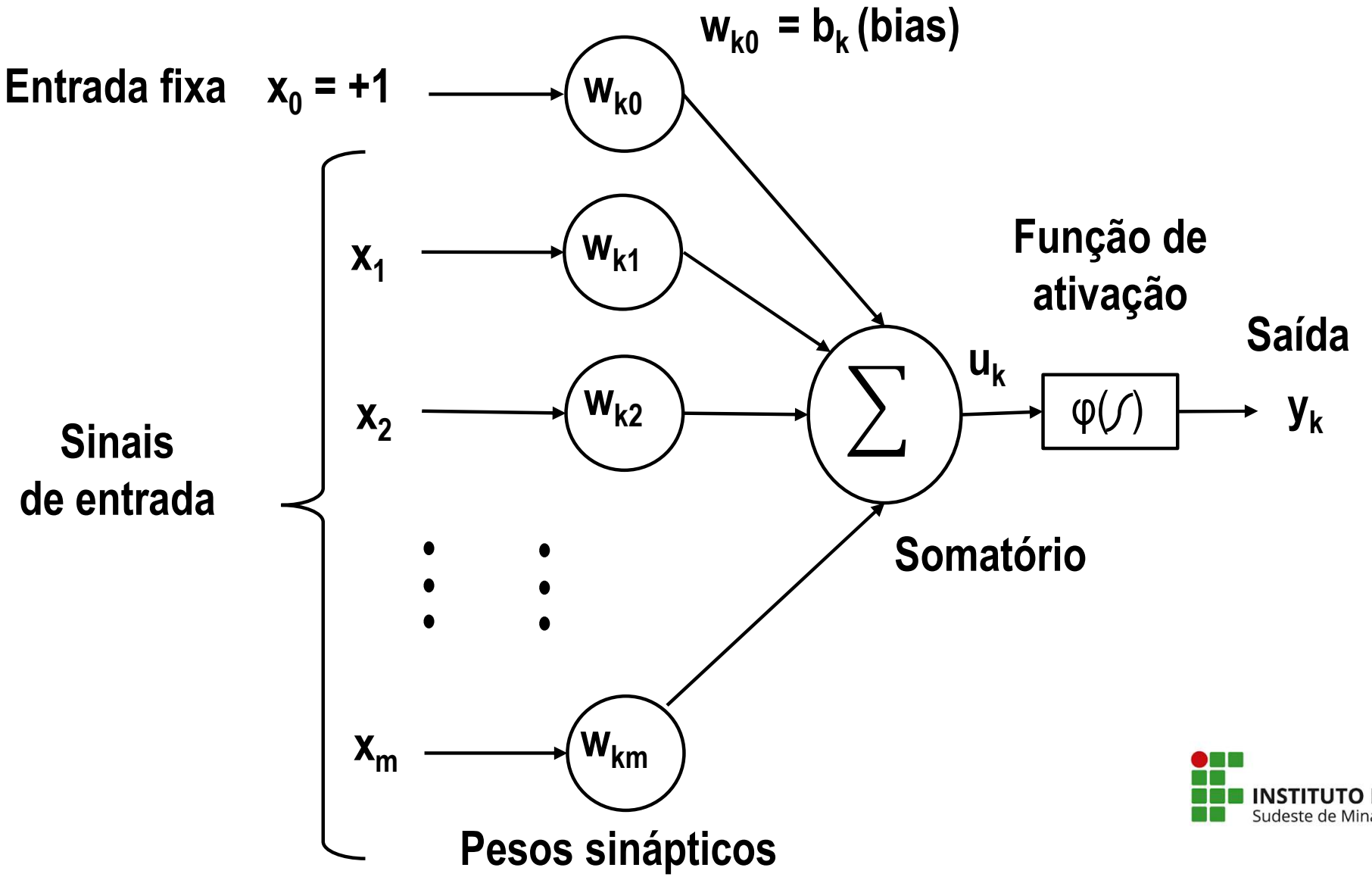
O Neurônio





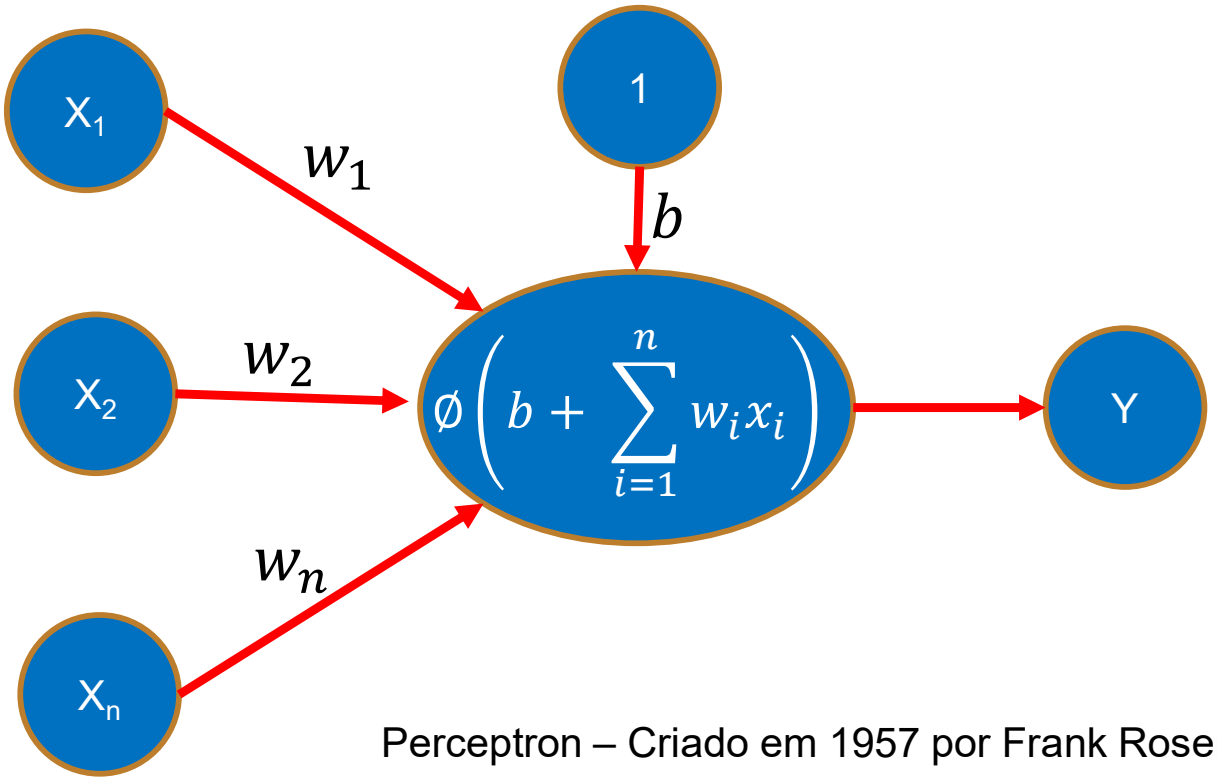
INSTITUTO FEDERAL
Sudeste de Minas Gerais

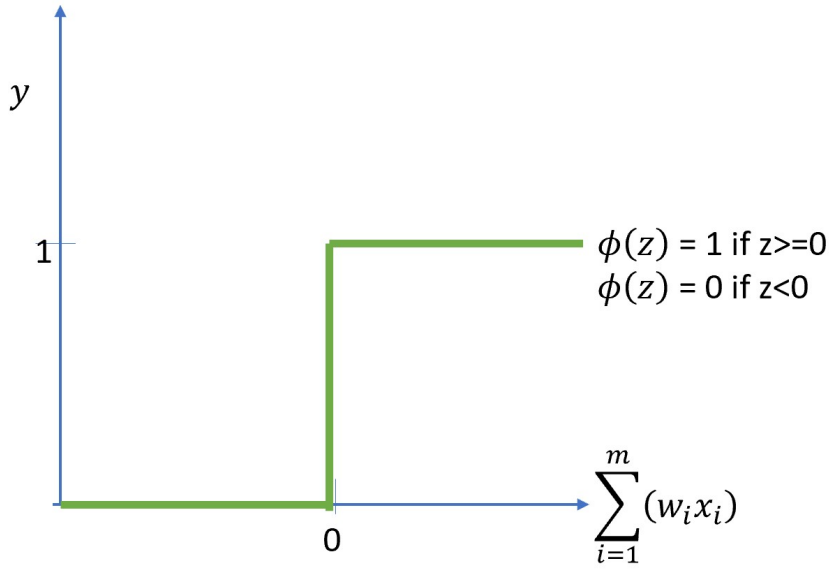
Neurônio Artificial



INSTITUTO FEDERAL
Sudeste de Minas Gerais

Função de Ativação

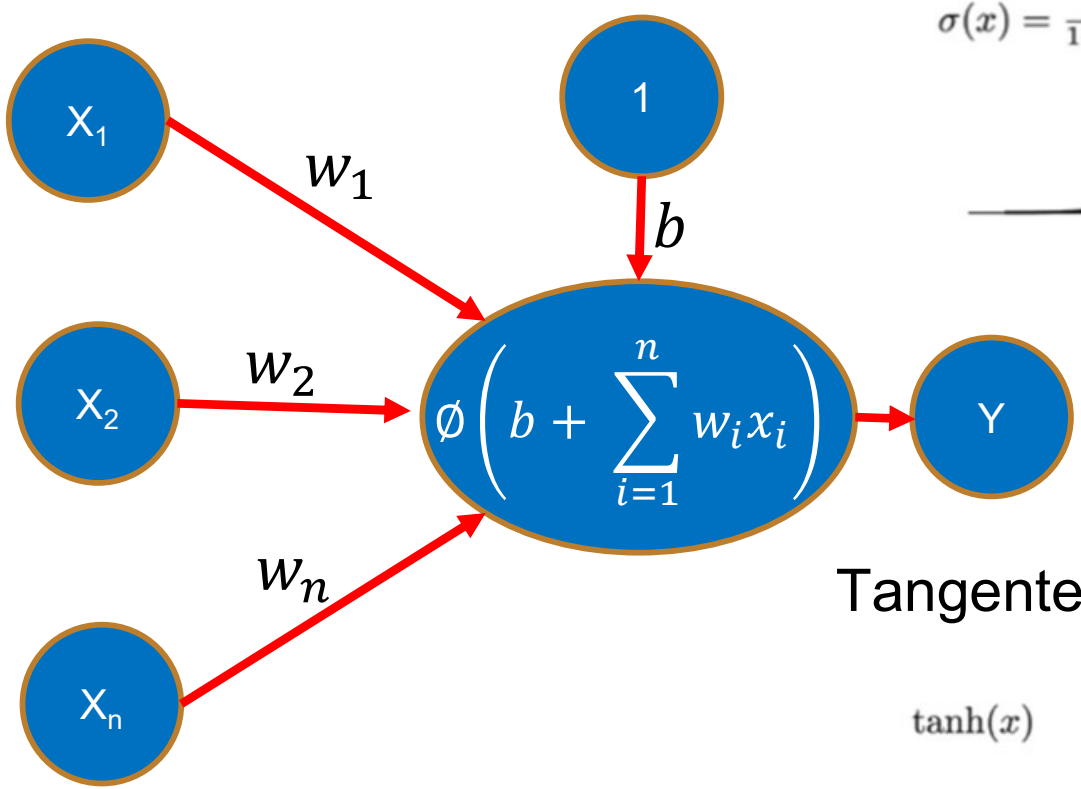




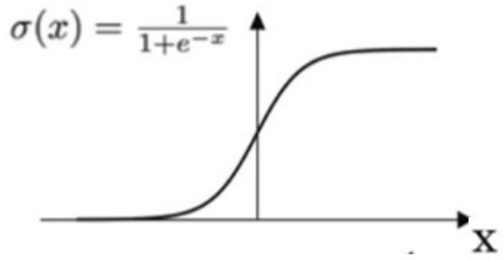
Perceptron – Criado em 1957 por Frank Rosenblatt

INSTITUTO FEDERAL
Sudeste de Minas Gerais

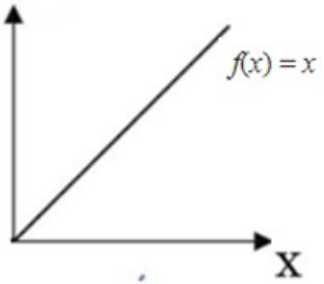
Função de Ativação



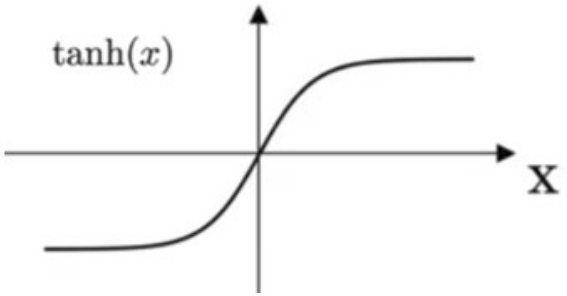
Sigmoide (logística)



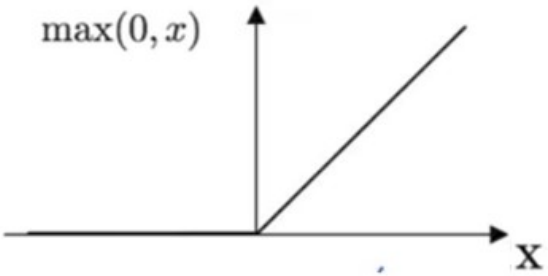
Linear



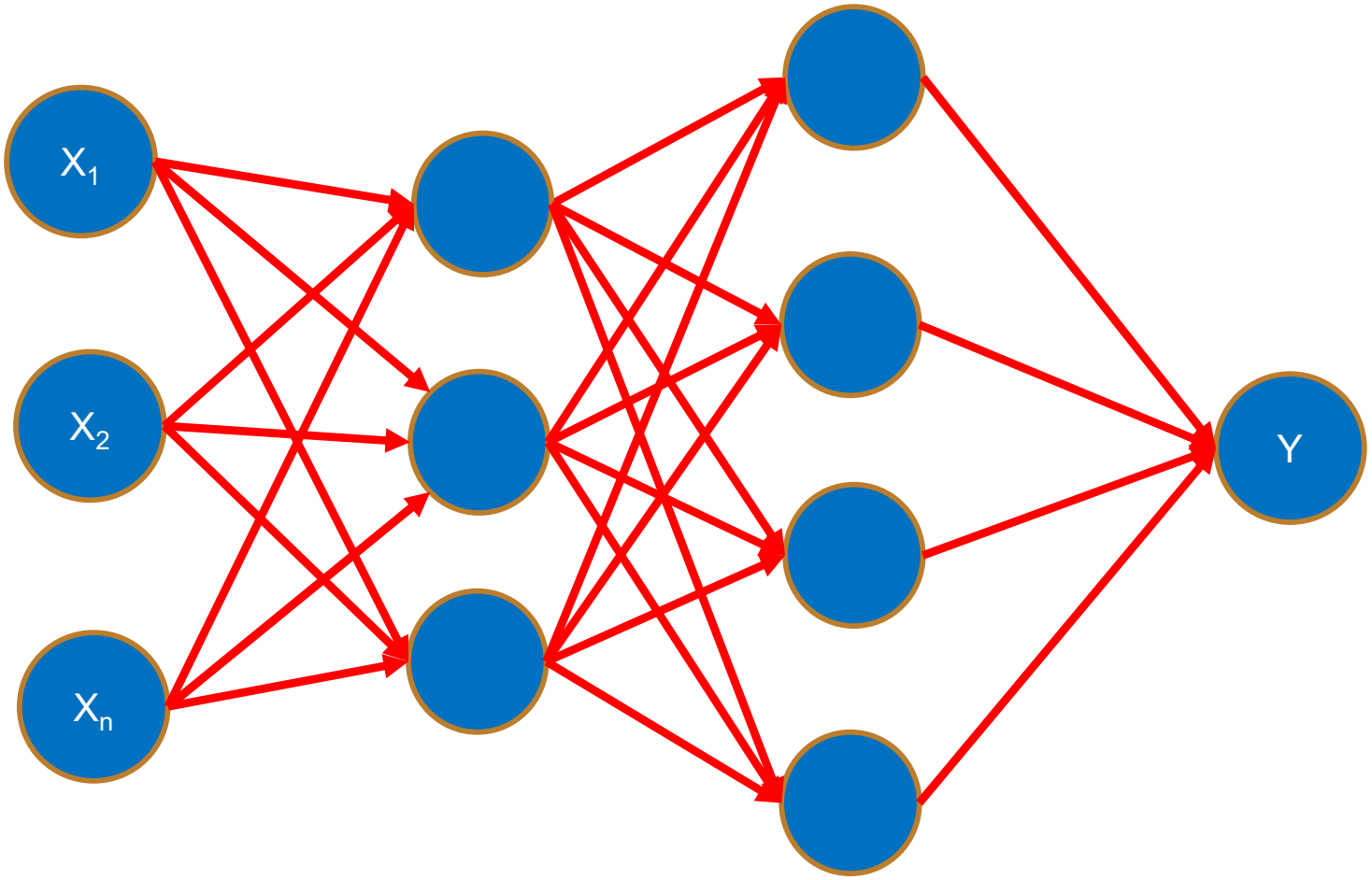
Tangente Hiperbólica



Linear retificada (ReLU)



Rede Neural Artificial



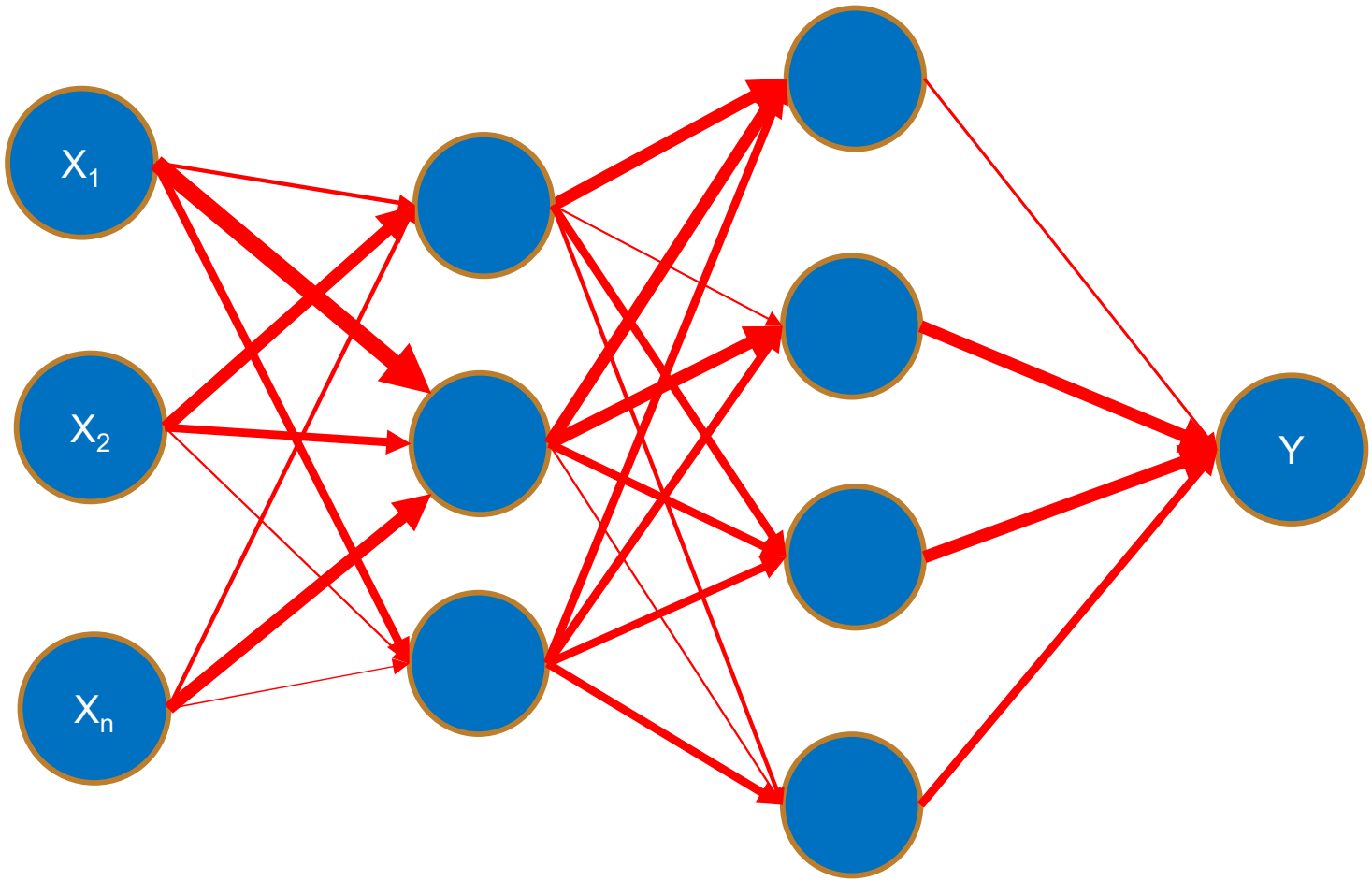
Camada de Entrada

Camadas escondida

Camada de Saída

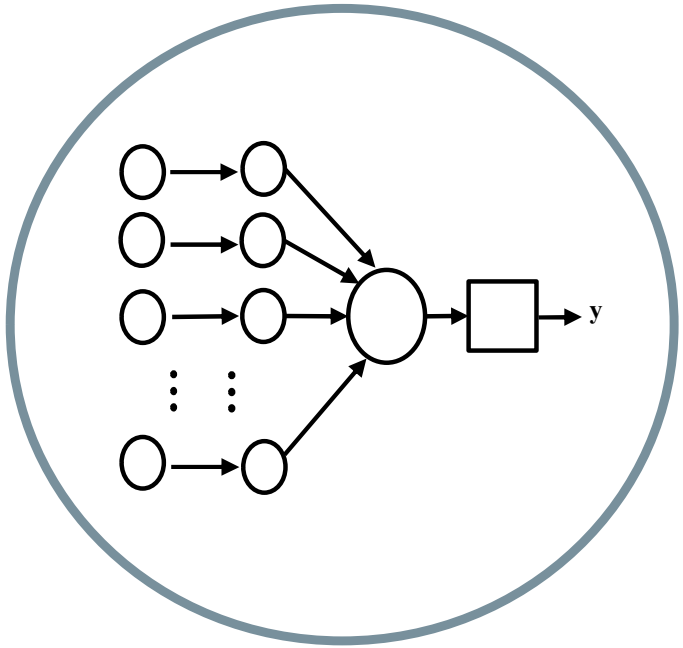
INSTITUTO FEDERAL
Sudeste de Minas Gerais

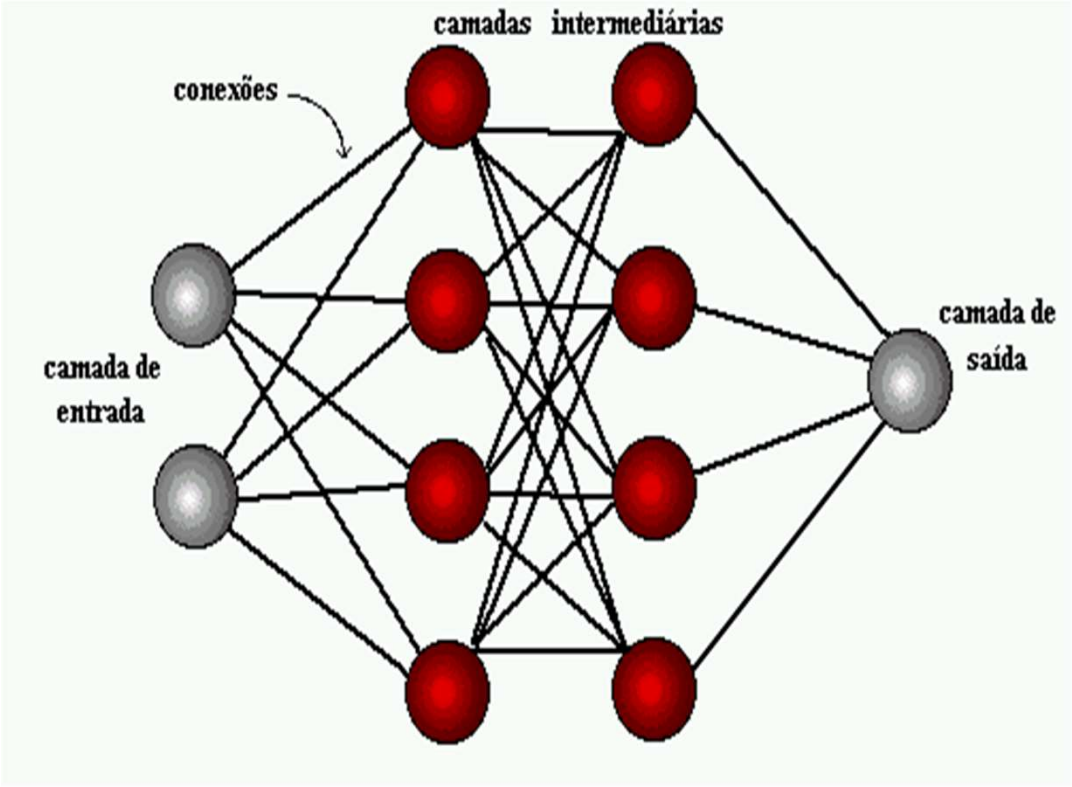
Rede Neural Artificial



INSTITUTO FEDERAL
Sudeste de Minas Gerais

Rede Neural Artificial





https://icmc.usp.br

INSTITUTO FEDERAL
Sudeste de Minas Gerais

Princípio de funcionamento

A operação de um neurônio artificial resume em:

- Sinais são apresentados à entrada (x_1 a x_m);
- Cada sinal é multiplicado por um peso que indica sua influência na saída da unidade (w_k);
- É feito a soma ponderada dos sinais que produz um nível de atividade (u_k);
- A função de ativação $f(u_k)$ tem a função de limitar a saída e introduzir não-linearidade ao modelo;
- O bias b_k tem o papel de aumentar ou diminuir a influência dos valores de entrada.

Princípio de funcionamento

- Matematicamente a saída é expressa por:

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

- Ou considerando o bias como entrada de valor x_0=1 e peso w_{k0}=b_k

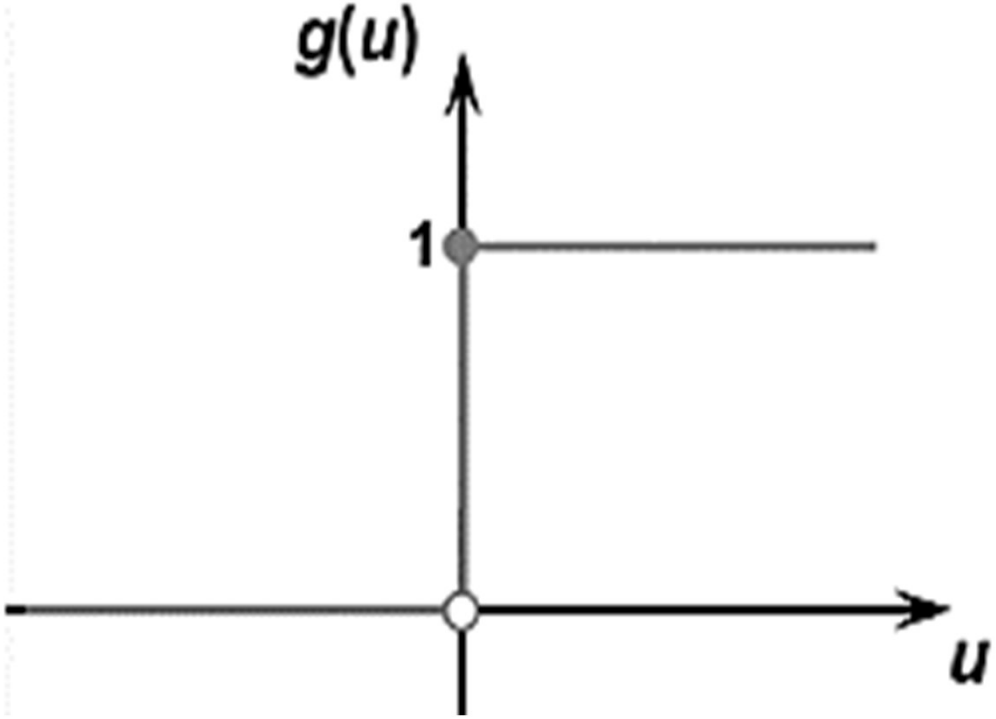
$$y_k = f(u_k) = f\left(\sum_{j=1}^{m} x_i w_i + b_k\right)$$

FUNÇÃO DE ATIVAÇÃO

INSTITUTO FEDERAL
Sudeste de Minas Gerais

Função degrau

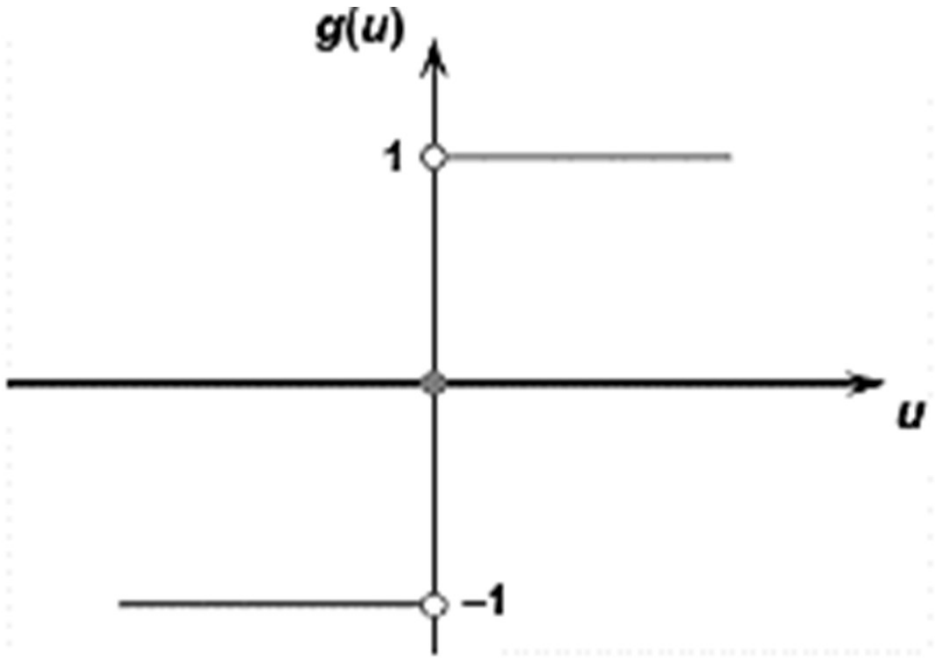
$$f(u) = \begin{cases} 1, \text{ se } u \geq 0 \\ 0, \text{ se } u < 0 \end{cases}$$



INSTITUTO FEDERAL
Sudeste de Minas Gerais

Função degrau bipolar

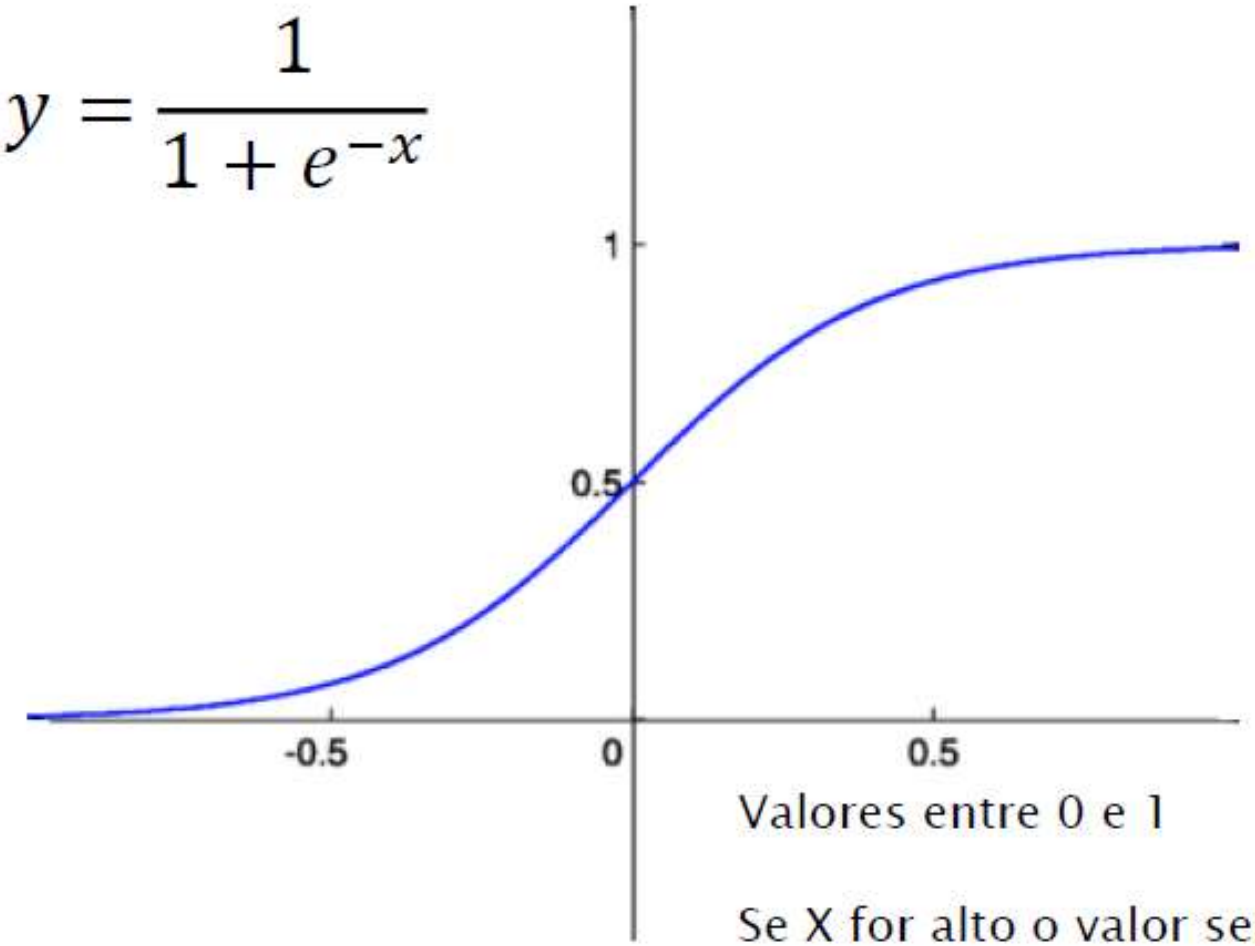
$$f(u) = \begin{cases} 1, \text{ se } u > 0 \\ 0, \text{ se } u = 0 \\ -1, \text{ se } u < 0 \end{cases}$$



INSTITUTO FEDERAL
Sudeste de Minas Gerais

Função sigmoide

$$y = \frac{1}{1 + e^{-x}}$$

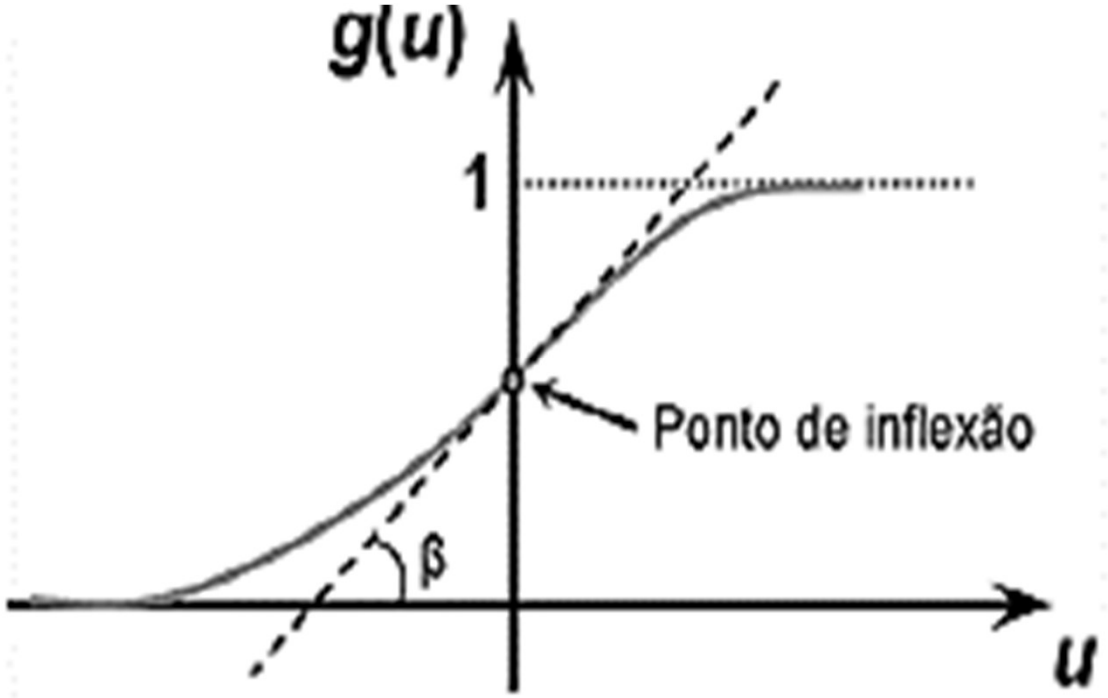


Valores entre 0 e 1

Se X for alto o valor será aproximadamente 1
Se X for pequeno o valor será aproximadamente 0
Não retorna valores negativos

Função logística

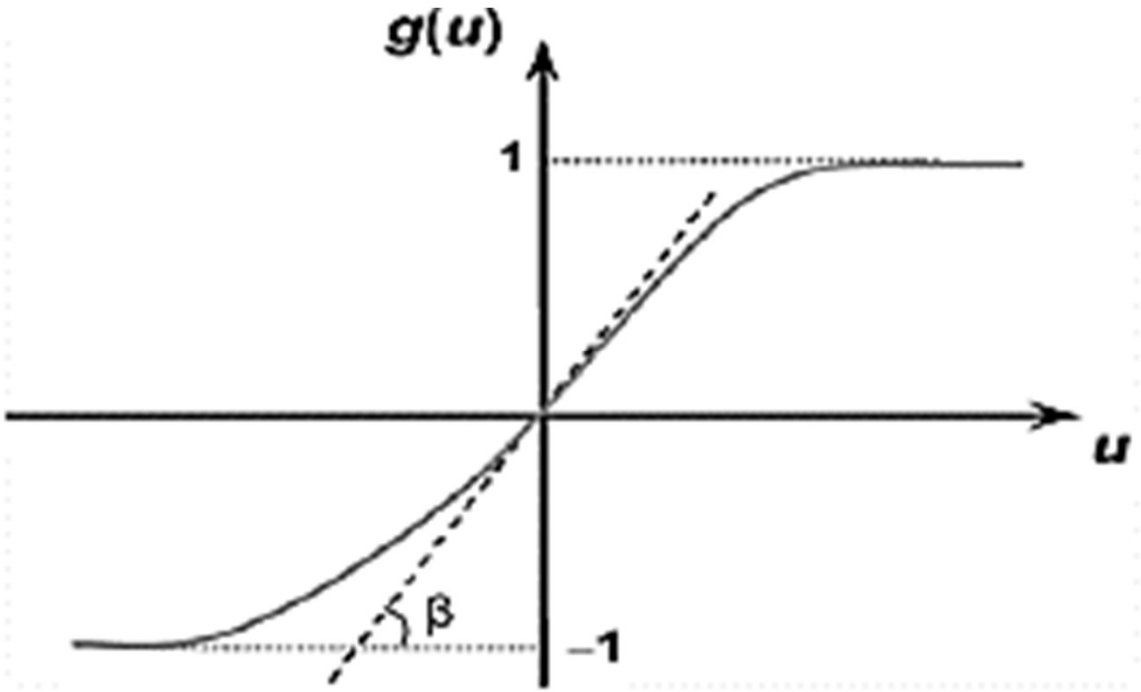
$$f(u) = \frac{1}{1+e^{-u}}$$



INSTITUTO FEDERAL
Sudeste de Minas Gerais

Função tangente hiperbólica

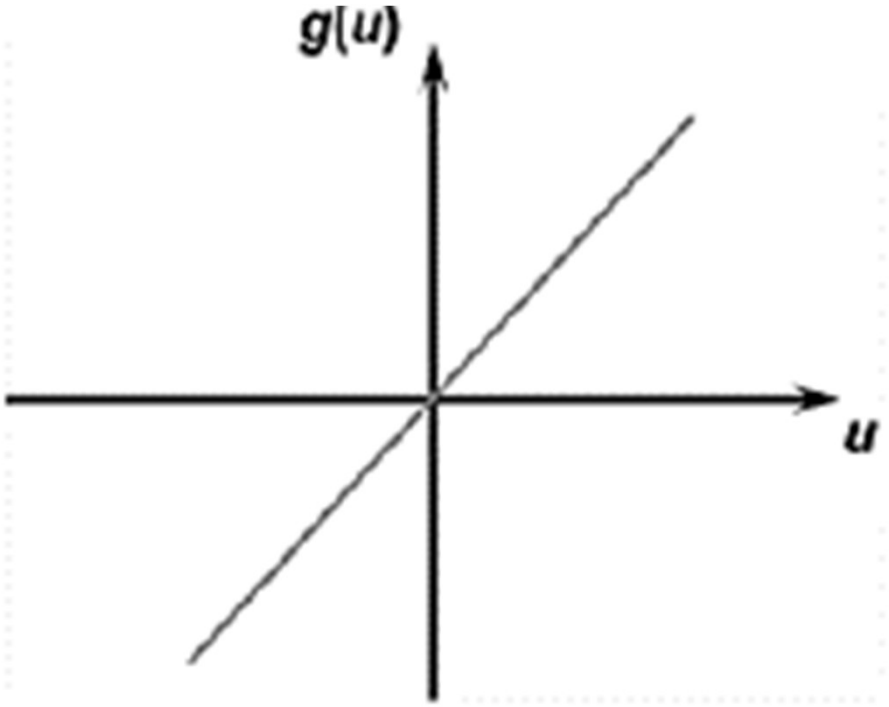
$$f(u) = \frac{1 - e^{-u}}{1 + e^{-u}}$$



INSTITUTO FEDERAL
Sudeste de Minas Gerais

Função linear

$$f(u) = u$$



INSTITUTO FEDERAL
Sudeste de Minas Gerais

ARQUITETURA

INSTITUTO FEDERAL
Sudeste de Minas Gerais

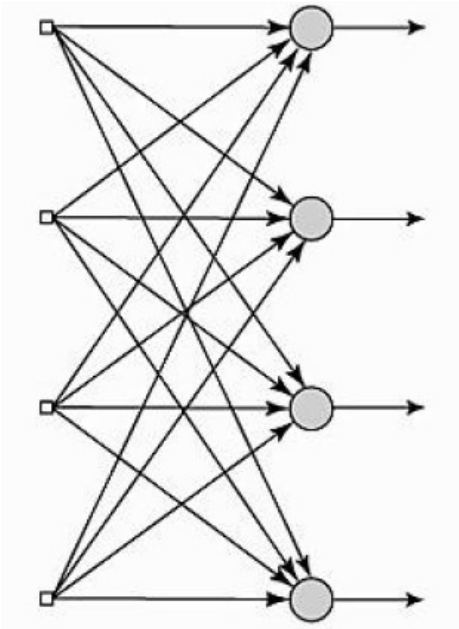
Principais arquiteturas

Em geral, podemos identificar três tipos de arquiteturas de rede fundamentalmente diferentes:

- Redes alimentadas adiante com Camada Única.
- Redes alimentadas diretamente com Múltiplas Camadas.
- Redes Recorrentes.

Arquitetura – Camada Única

- Há somente uma camada de ligações com pesos. As unidades podem ser distinguidas como unidades de entrada e saída.

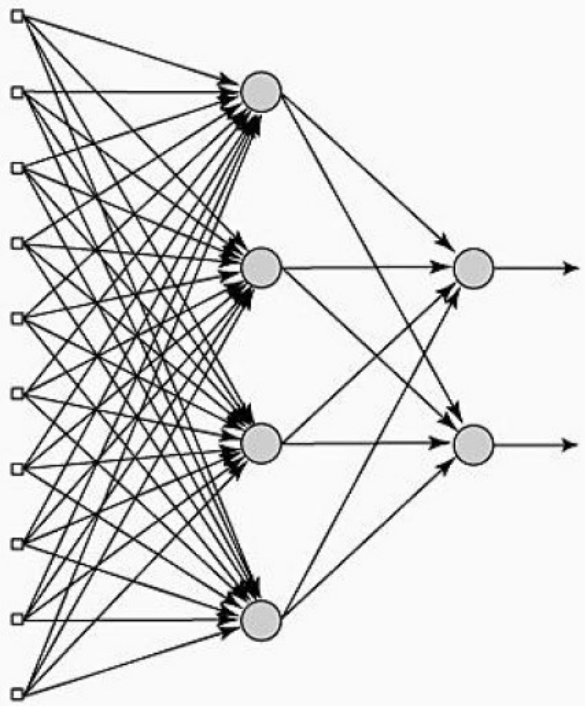


Fonte: Haykin, 2001.

INSTITUTO FEDERAL
Sudeste de Minas Gerais

Arquitetura – Camada Múltipla

- Uma ou mais camadas de nós intermediários às unidades de entrada e saída, chamadas unidades ocultas (*hidden*). (ex: Multilayer Perceptron).

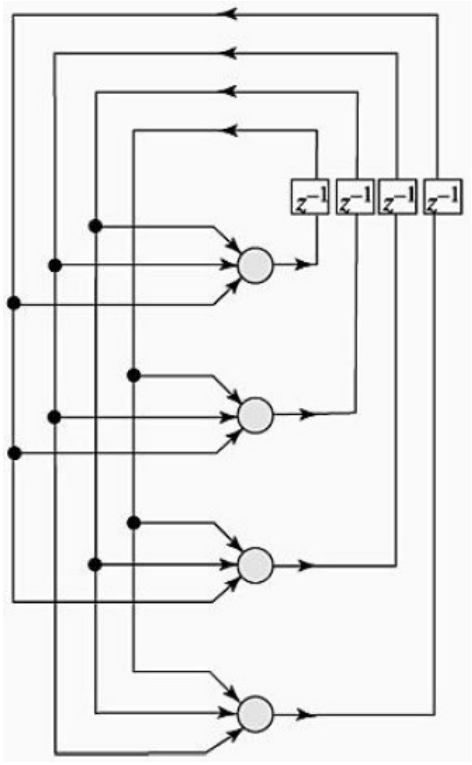


Fonte: Haykin, 2001.

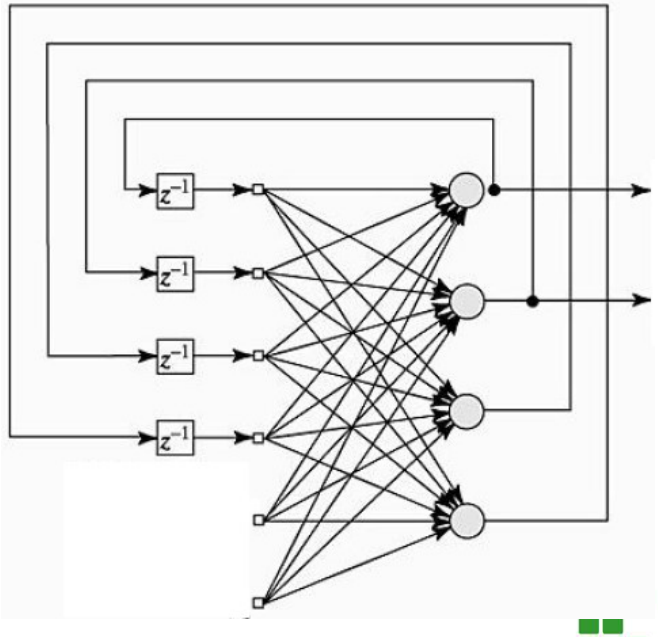
Arquitetura – Recorrentes

- Se distingue de uma rede neural alimentada adiante por ter pelo menos um laço realimentação.

Sem neurônios ocultos



Com neurônios ocultos



INSTITUTO FEDERAL
Sudeste de Minas Gerais

Fonte: Haykin, 2001.

TREINAMENTO

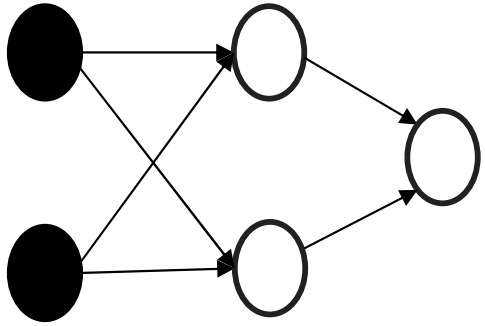
INSTITUTO FEDERAL
Sudeste de Minas Gerais

Como uma RNA aprende?

Exemplos
(entrada – saída)

F	G	H	K
ARVORE	DAP	HT	V4cc
427	7,1	12,7	0,023337981
69	15,75	20,24	0,155035661
323	21,45	24,8	0,351773156
85	6,3	12,3	0,01276902
451	10,1	15,65	0,059425801
504	18,5	26	0,297424335
395	16,1	18,3	0,159896019
497	20,6	28,3	0,425194194
36	5,2	11,24	0,007017407

Configuração



Resposta

Erro

Avaliação

Saída desejada

INSTITUTO FEDERAL
Sudeste de Minas Gerais

Como uma RNA aprende?

- Processo iterativo de ajuste dos parâmetros da rede
- Os **pesos** armazenam o conhecimento adquirido pela rede
- Os algoritmos são agrupados em dois paradigmas principais: **aprendizado supervisionado** e aprendizado não supervisionado

INSTITUTO FEDERAL
Sudeste de Minas Gerais

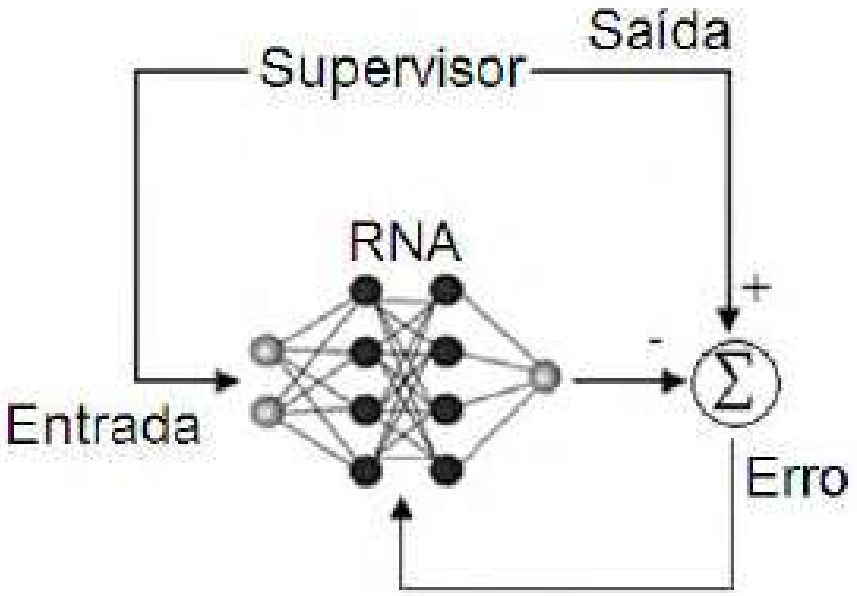
Fonte: Haykin, 2001.

Aprendizado Supervisionado

O método mais comum de treinamento das RNAs

- As entradas e saídas são fornecidas por um supervisor (professor) externo.
- Ajusta-se os parâmetros da rede, encontrando alguma ligação entre os pares de entrada e saída.
- O professor indica um comportamento bom ou ruim da rede.

Aprendizado Supervisionado



A rede tem uma resposta (saída) que é comparada com a saída desejada, recebendo informações do supervisor sobre o erro da resposta atual.

INSTITUTO FEDERAL
Sudeste de Minas Gerais

Aprendizado Supervisionado

O método mais comum de treinamento das RNAs

- As entradas e saídas são fornecidas por um supervisor (professor) externo.
- Ajusta-se os parâmetros da rede, encontrando alguma ligação entre os pares de entrada e saída.
- O professor indica um comportamento bom ou ruim da rede.

DEMONSTRAÇÃO

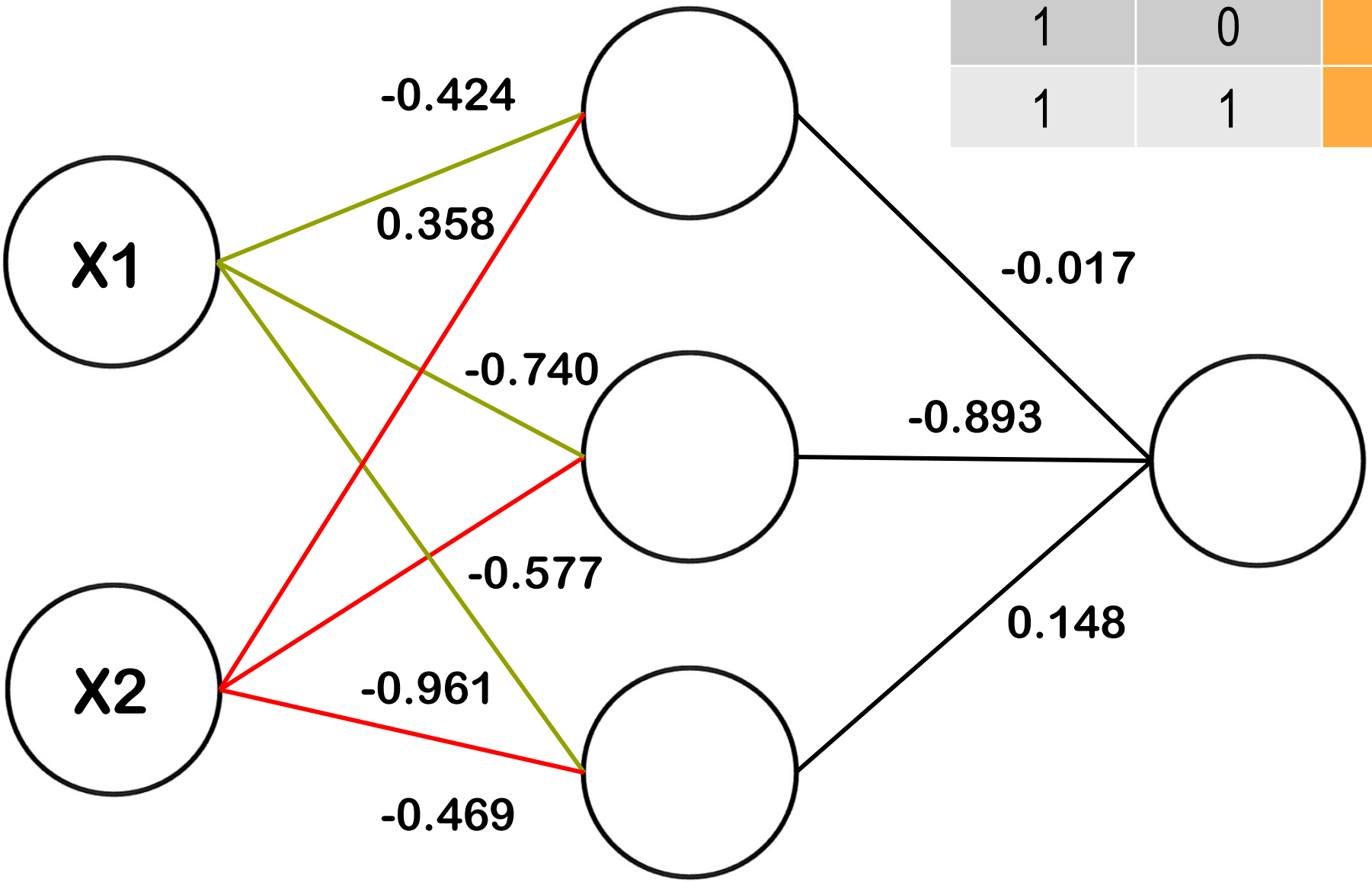
INSTITUTO FEDERAL
Sudeste de Minas Gerais

Operador XOR

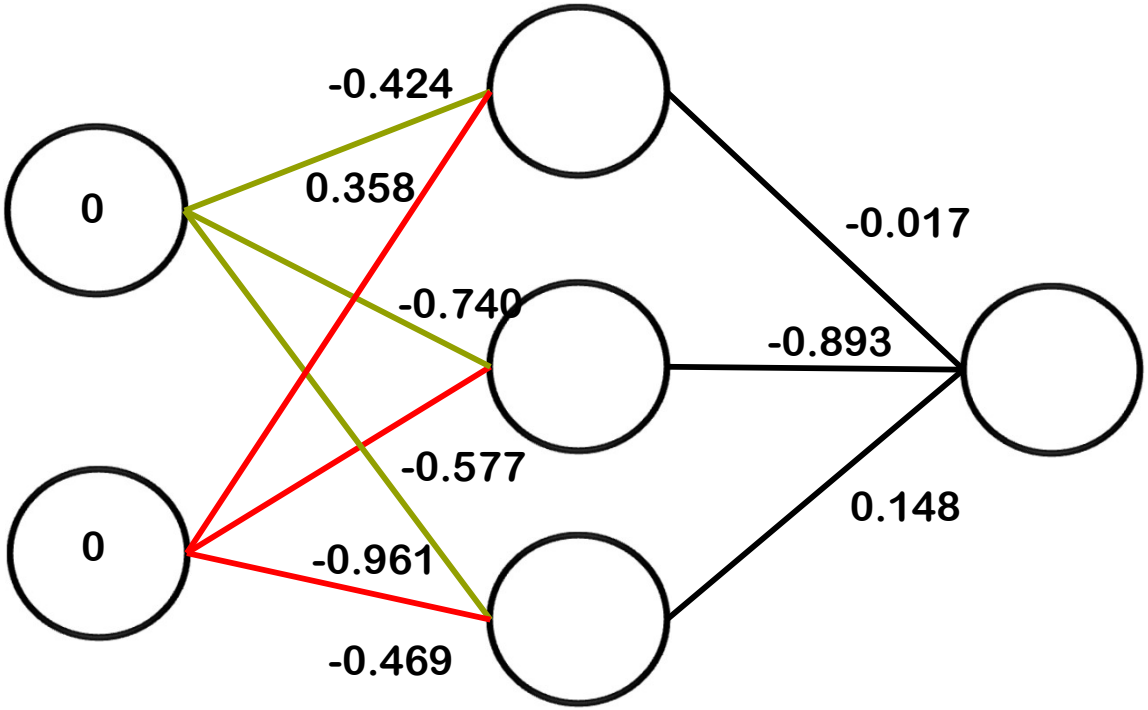
X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

Ativação camada oculta

X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0



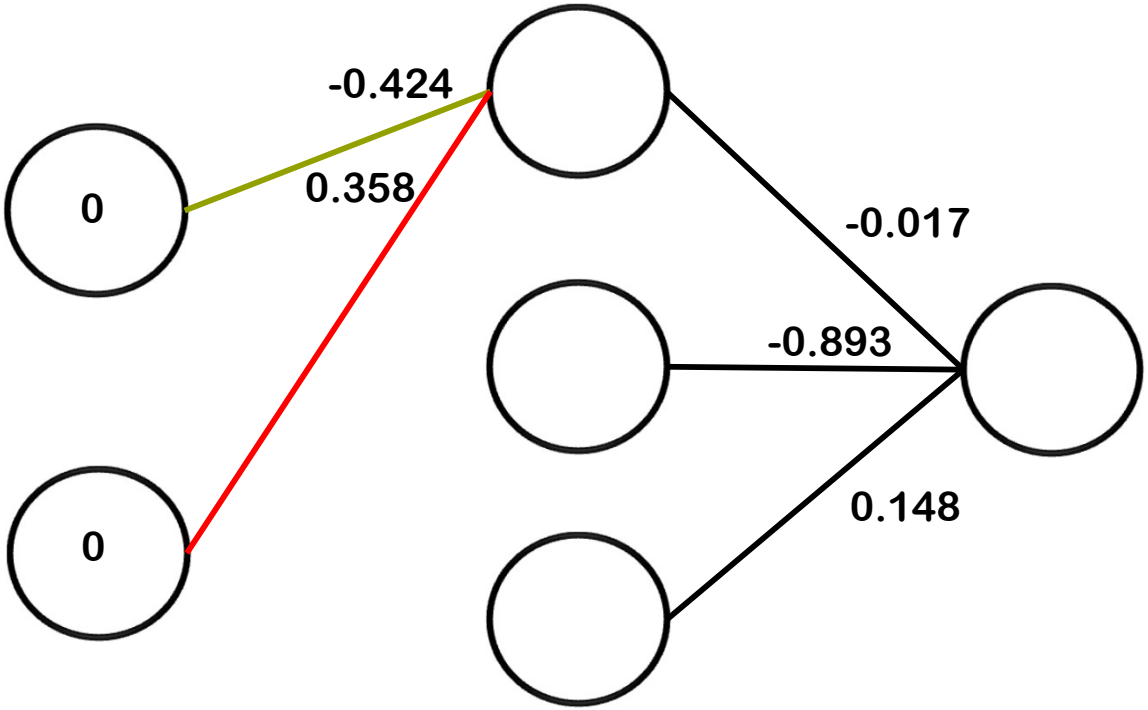
1º Registro



X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

$$Soma = \sum_{i=1}^{n} x_i * w_i$$

1º Registro

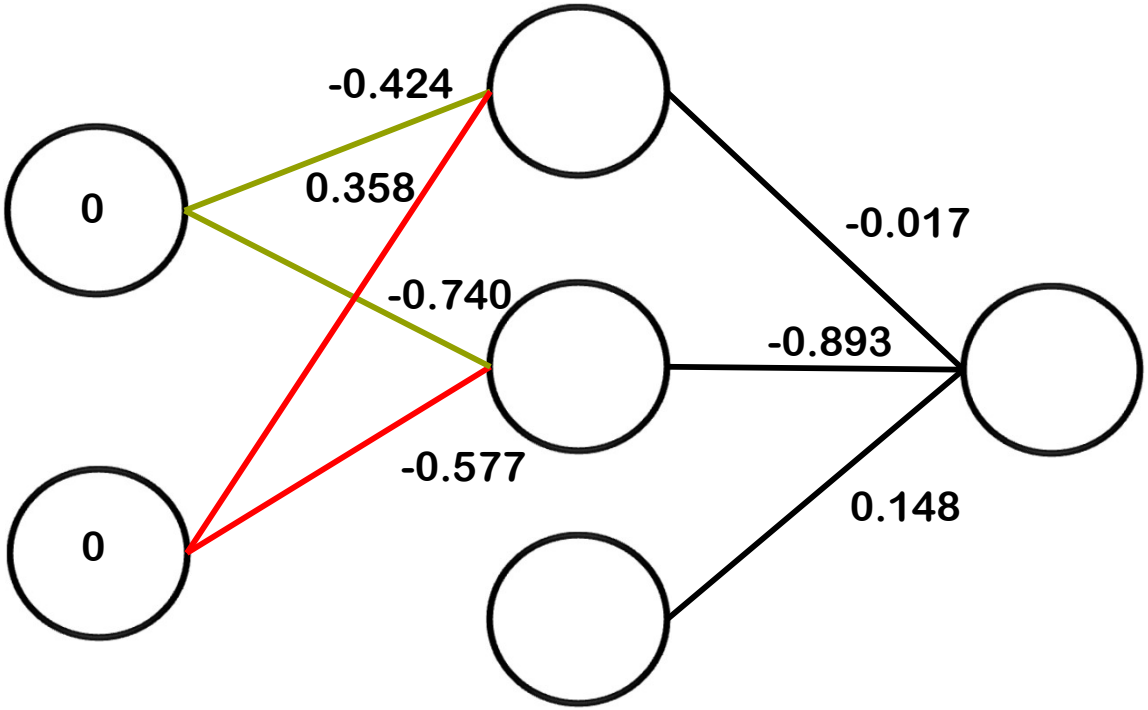


X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

$$Soma = \sum_{i=1}^{n} x_i * w_i$$

u1 = 0 * (-0.424) + 0 * 0.358 = 0

1º Registro



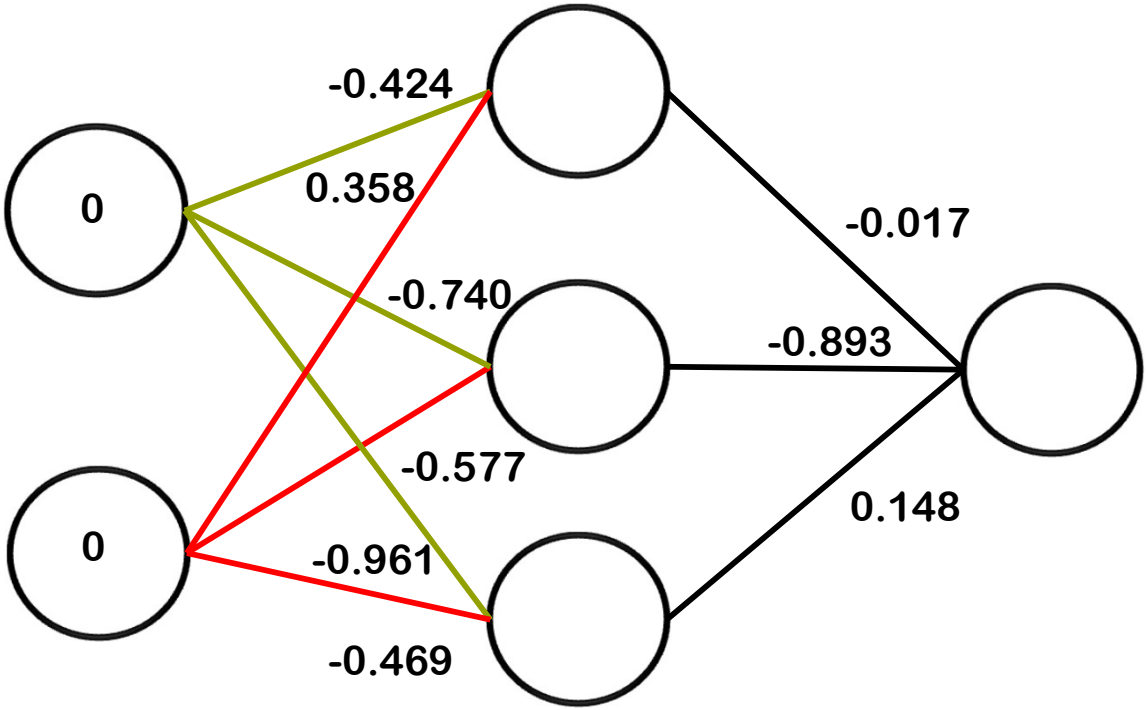
X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

$$Soma = \sum_{i=1}^{n} x_i * w_i$$

u1 = 0 * (-0.424) + 0 * 0.358 = 0

u2 = 0 * (-0.740) + 0 * (-0.577) = 0

1º Registro



X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

$$Soma = \sum_{i=1}^{n} x_i * w_i$$

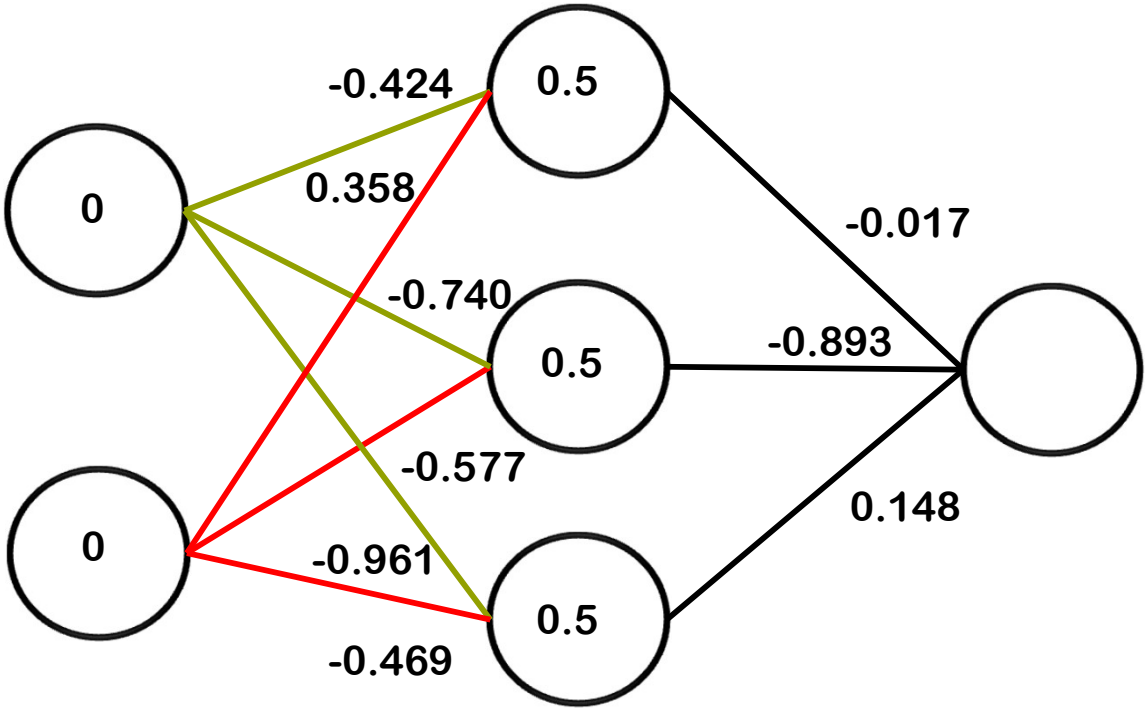
u1 = 0 * (-0.424) + 0 * 0.358 = 0

u2 = 0 * (-0.740) + 0 * (-0.577) = 0

u3 = 0 * (-0.961) + 0 * (-0.469) = 0

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

1º Registro



X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

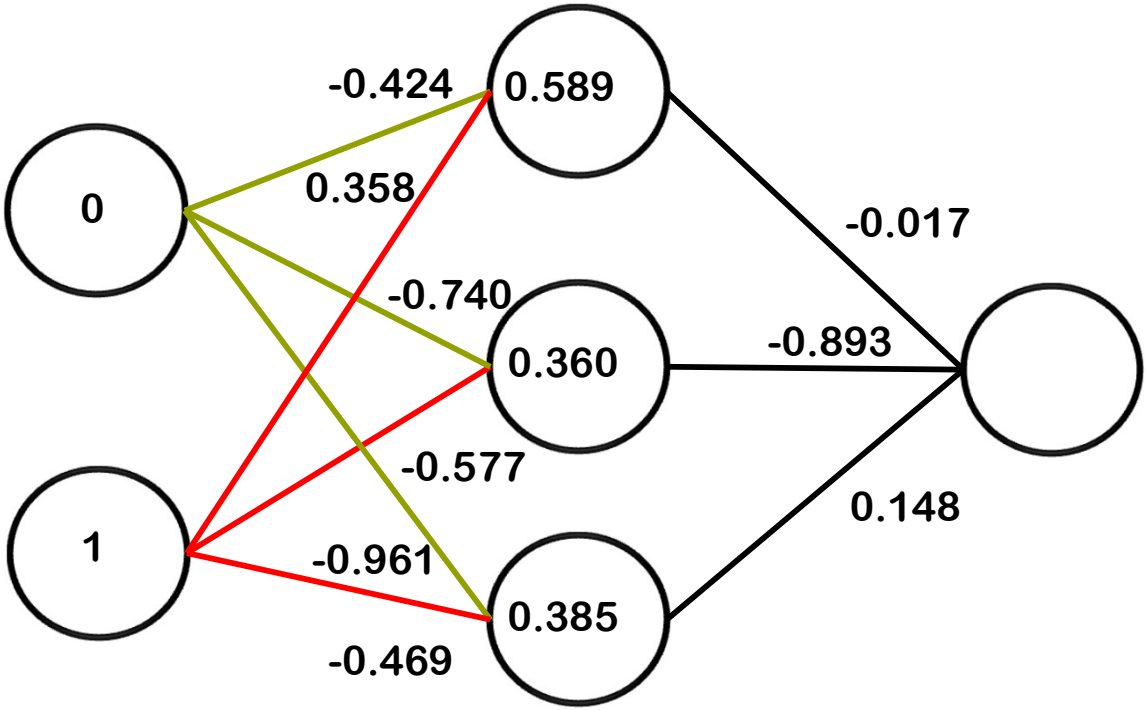
$$Soma = \sum_{i=1}^{n} x_i * w_i$$

u1 = 0 * (-0.424) + 0 * (0.358) = 0 → Ativação = 0.5

u2 = 0 * (-0.740) + 0 * (-0.577) = 0 → Ativação = 0.5

u3 = 0 * (-0.961) + 0 * (-0.469) = 0 → Ativação = 0.5

2º Registro



X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

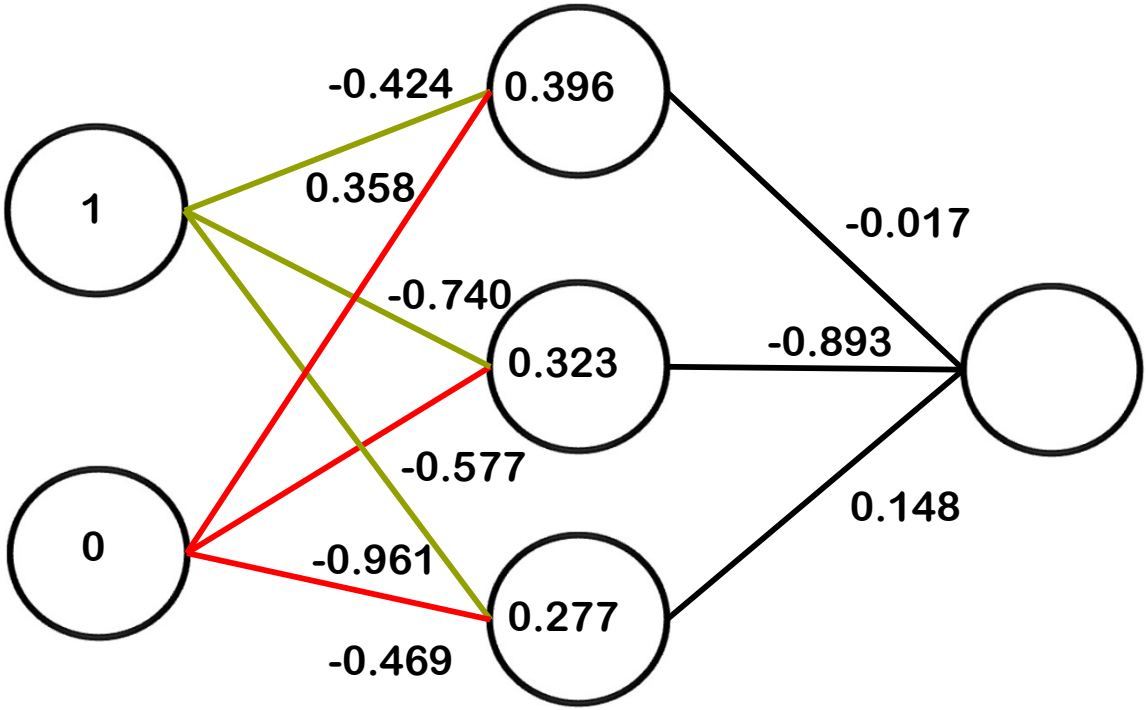
$$f(u) = \frac{1}{1+e^{-u}}$$

u1 = 0 * (-0.424) + 1 * 0.358 = 0.358

u2 = 0 * (-0.740) + 1 * (-0.577) = -0.577

u3 = 0 * (-0.961) + 1 * (-0.469) = -0.469

3º Registro



X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

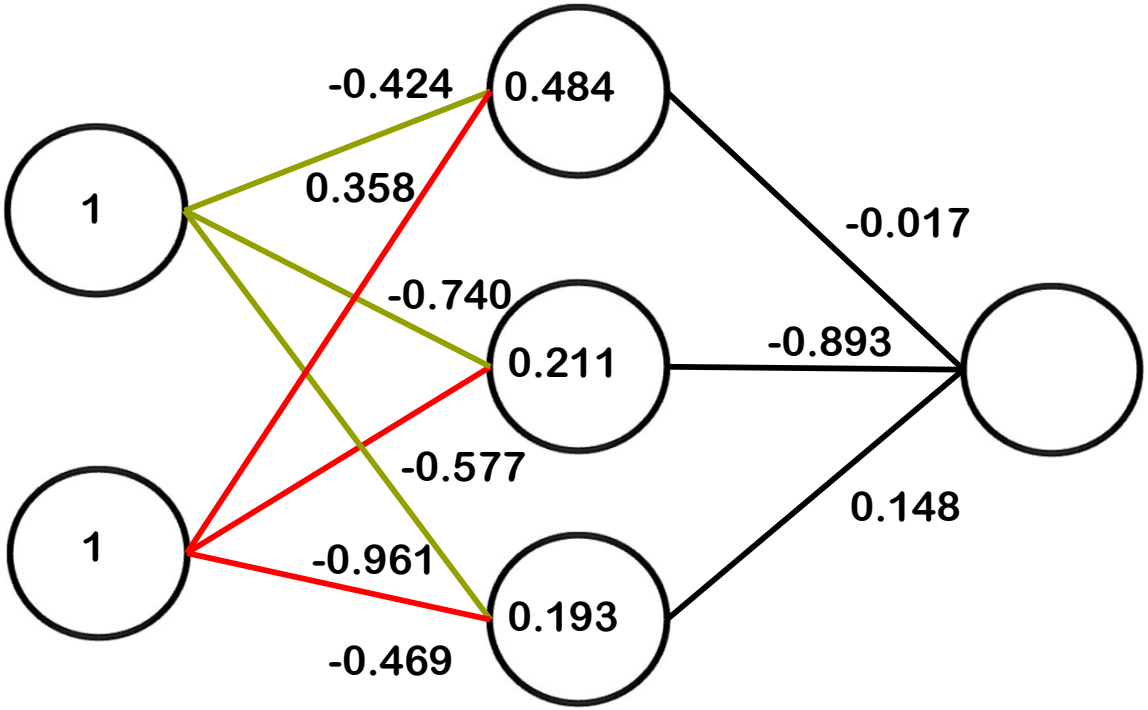
$$f(u) = \frac{1}{1+e^{-u}}$$

u1 = 1 * (-0.424) + 0 * 0.358 = -0.424

u2 = 1 * (-0.740) + 0 * (-0.577) = -0.740

u3 = 1 * (-0.961) + 0 * (-0.469) = -0.961

4º Registro



X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

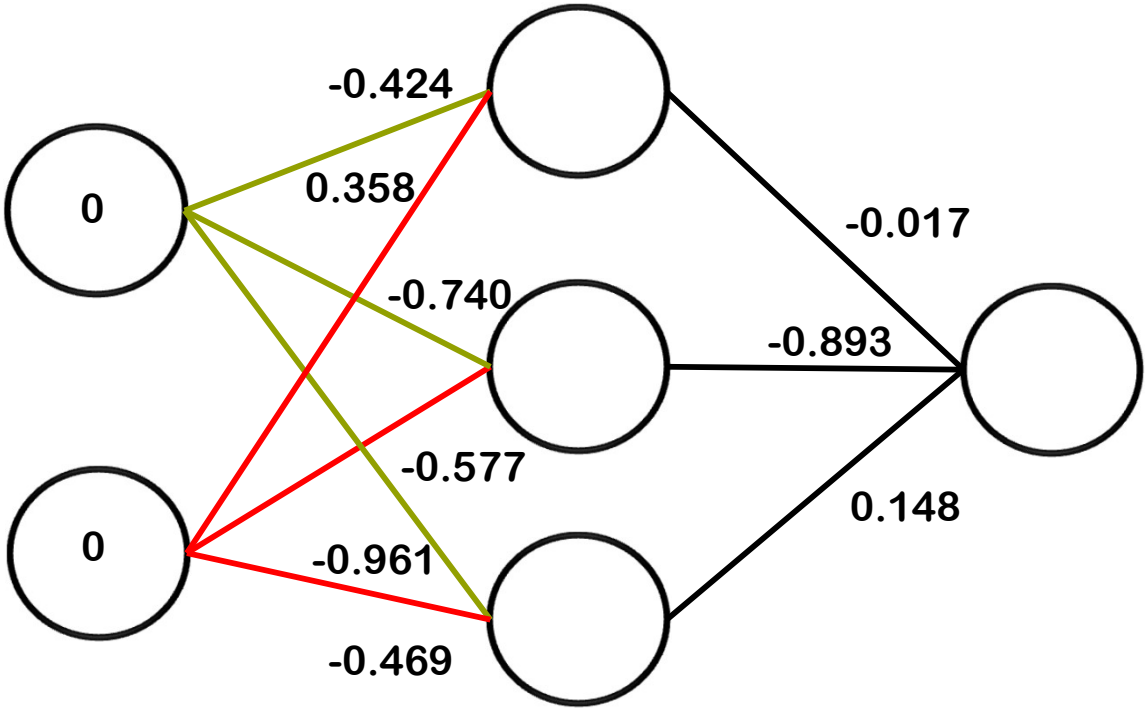
$$f(u) = \frac{1}{1+e^{-u}}$$

u1 = 1 * (-0.424) + 1 * 0.358 = -0.066

u2 = 1 * (-0.740) + 1 * (-0.577) = -1.317

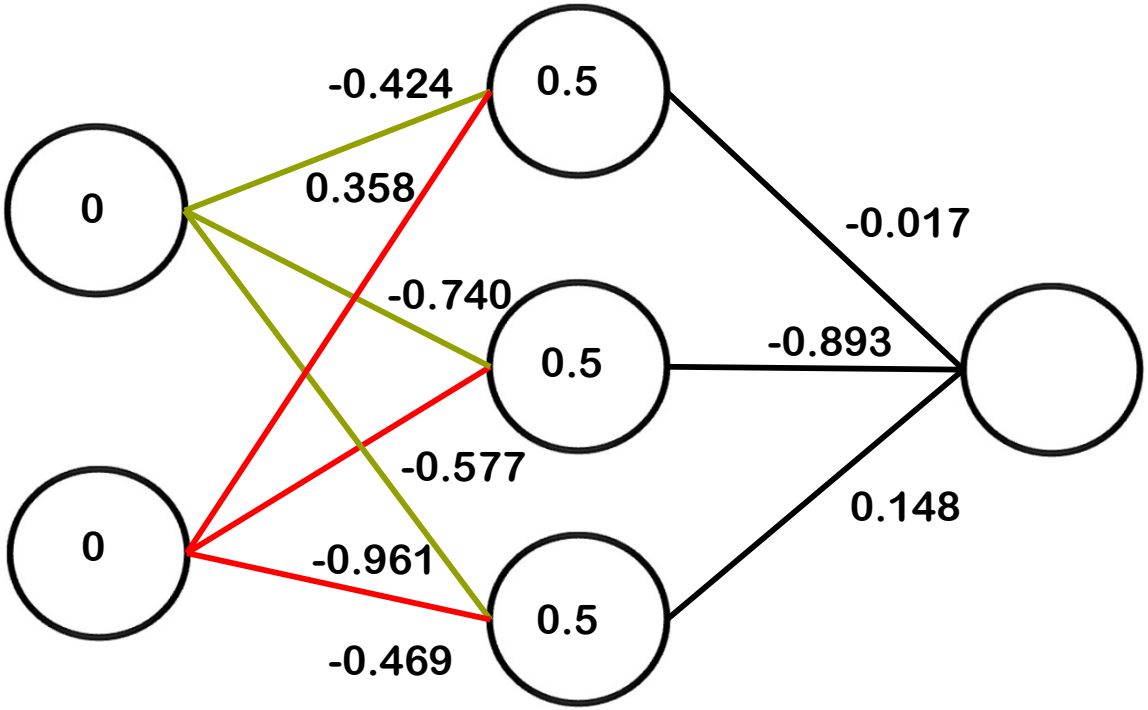
u3 = 1 * (-0.961) + 1 * (-0.469) = -1.430

Ativação camada saída



X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

1º Registro



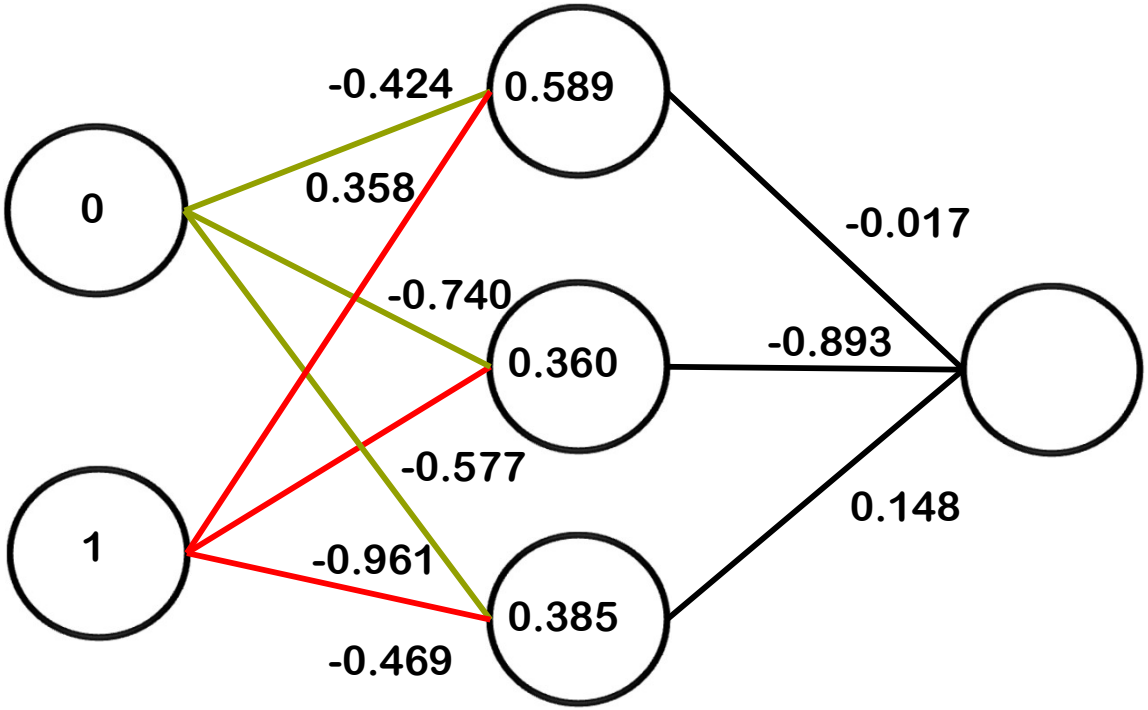
X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

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

u = 0.5 * (-0.017) + 0.5 * (-0.893) + 0.5 * (0.148) = -0.381

Ativação = **0.406**

2º Registro



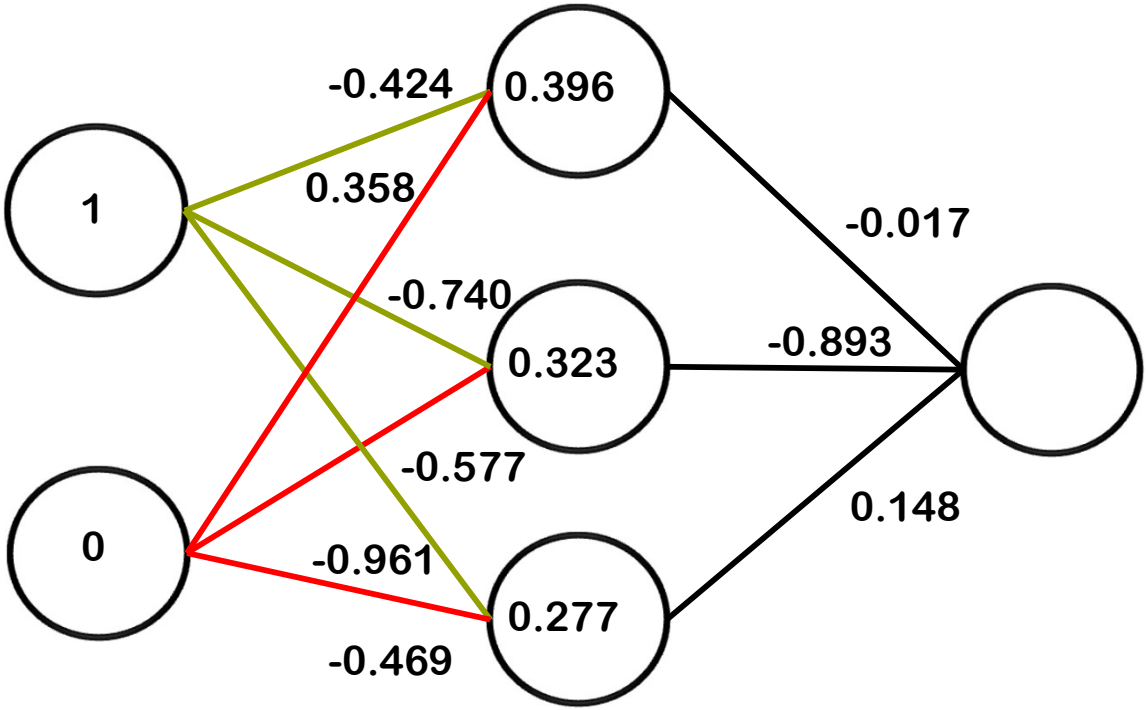
X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

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

u = 0.589 * (-0.017) + 0.360 * (-0.893) + 0.385 * (0.148) = -0.275

Ativação = **0.432**

3º Registro



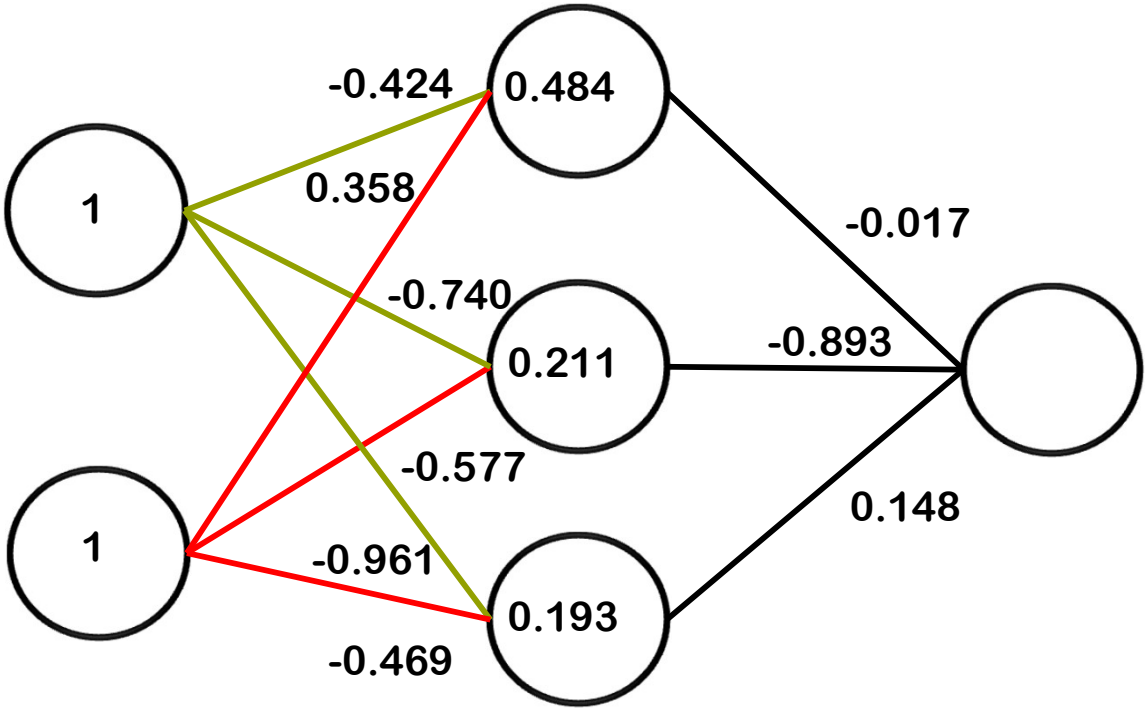
X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

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

u = 0.396 * (-0.017) + 0.323 * (-0.893) + 0.277 * (0.148) = -0.254

Ativação = **0.437**

4º Registro



X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

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

u = 0.484 * (-0.017) + 0.211 * (-0.893) + 0.193 * (0.148) = -0.168

Ativação = **0.458**

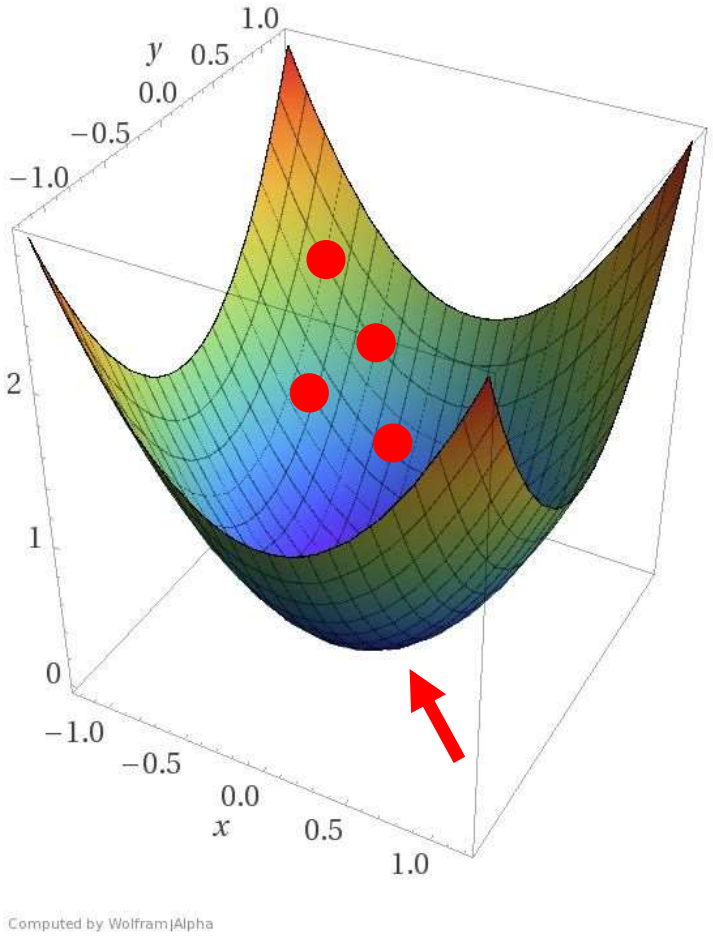
Cálculo do erro

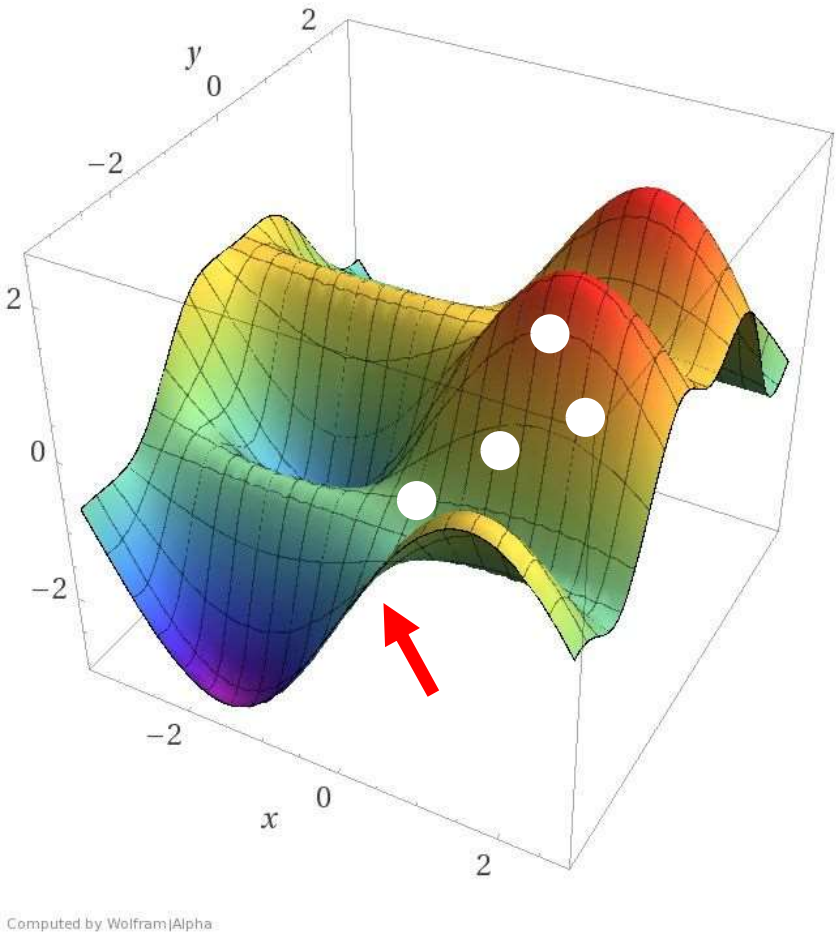
Erro = Resposta correta – Resposta calculada

X1	X2	CLASSE	CALCULADO	ERRO
0	0	0	0.406	-0.406
0	1	1	0.432	0.568
1	0	1	0.437	0.563
1	1	0	0.458	-0.458

Média absoluta = 0.49

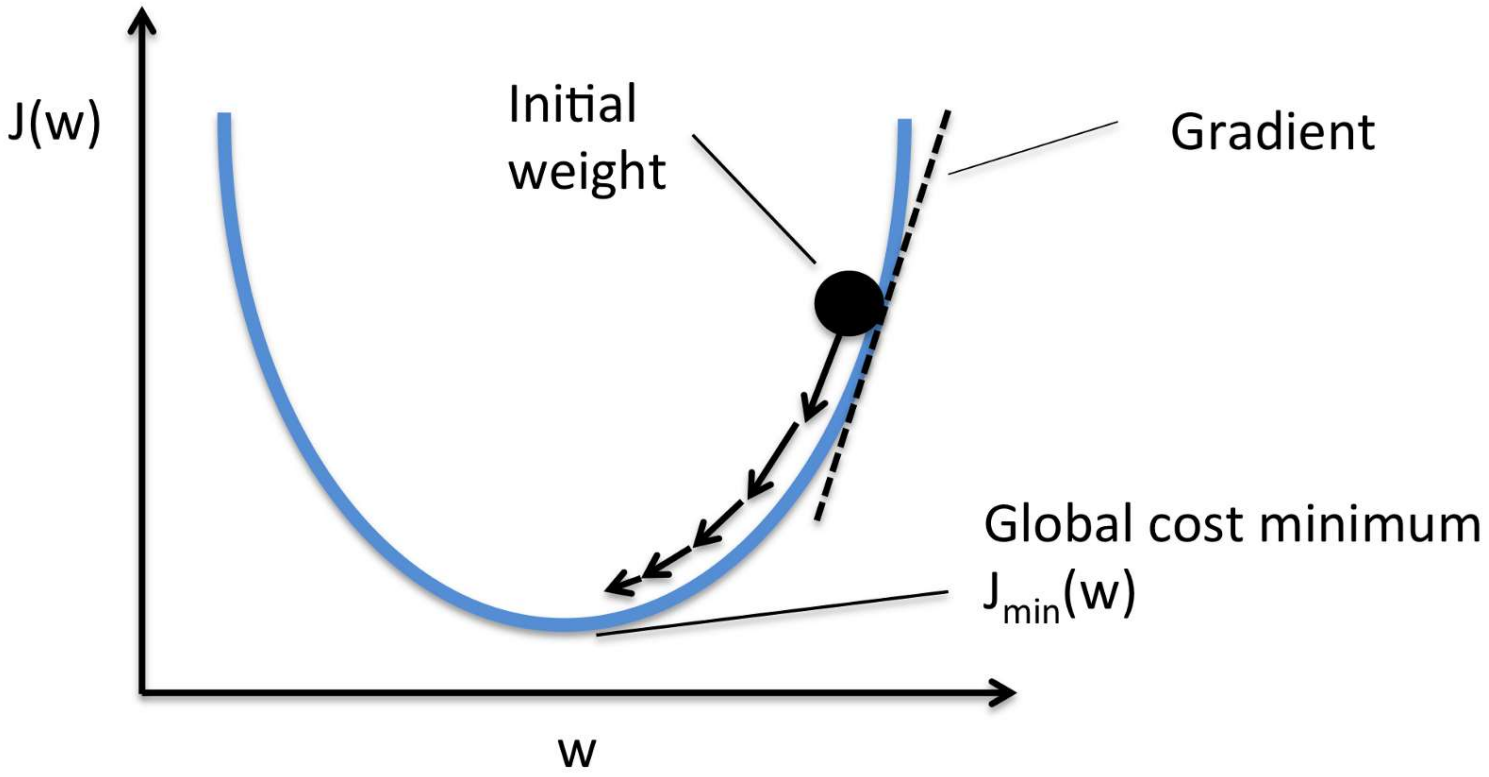
Descida do gradiente (*gradient descent*)





https://blog.paperspace.com

Descida do gradiente (*gradient descent*)



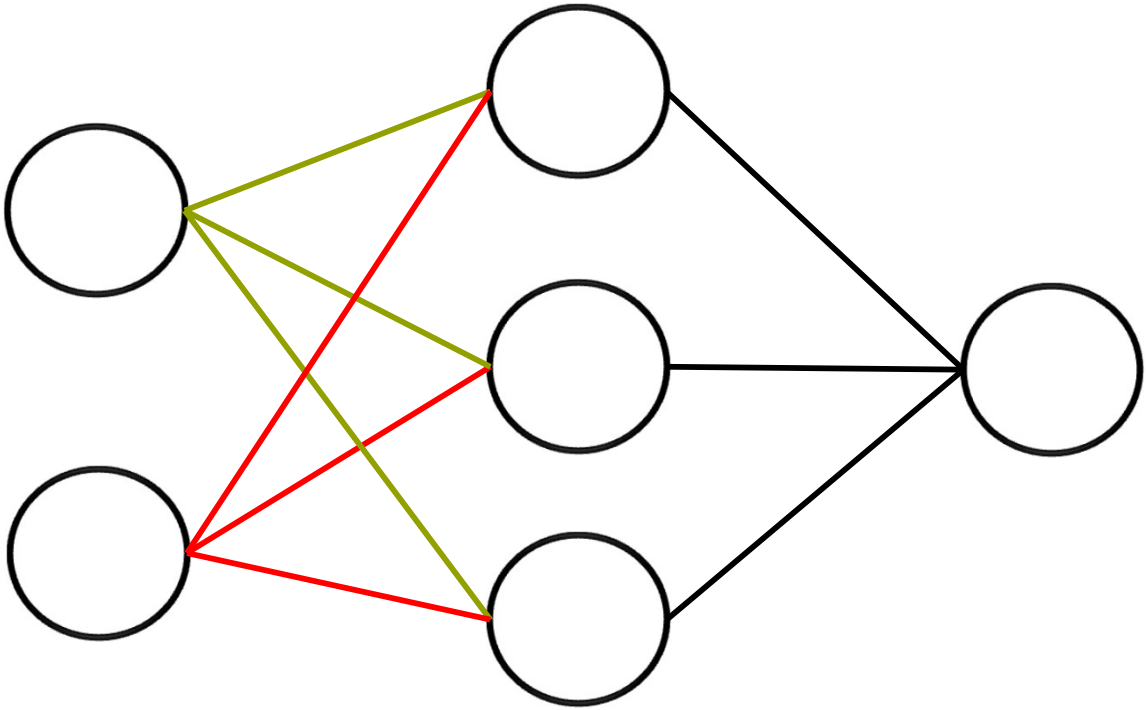
https://hackernoon.com

Descida do gradiente (*gradient descent*)

$$y = \frac{1}{1+e^{-u}} \longrightarrow y' = y * (1 - y)$$

Derivada função de ativação

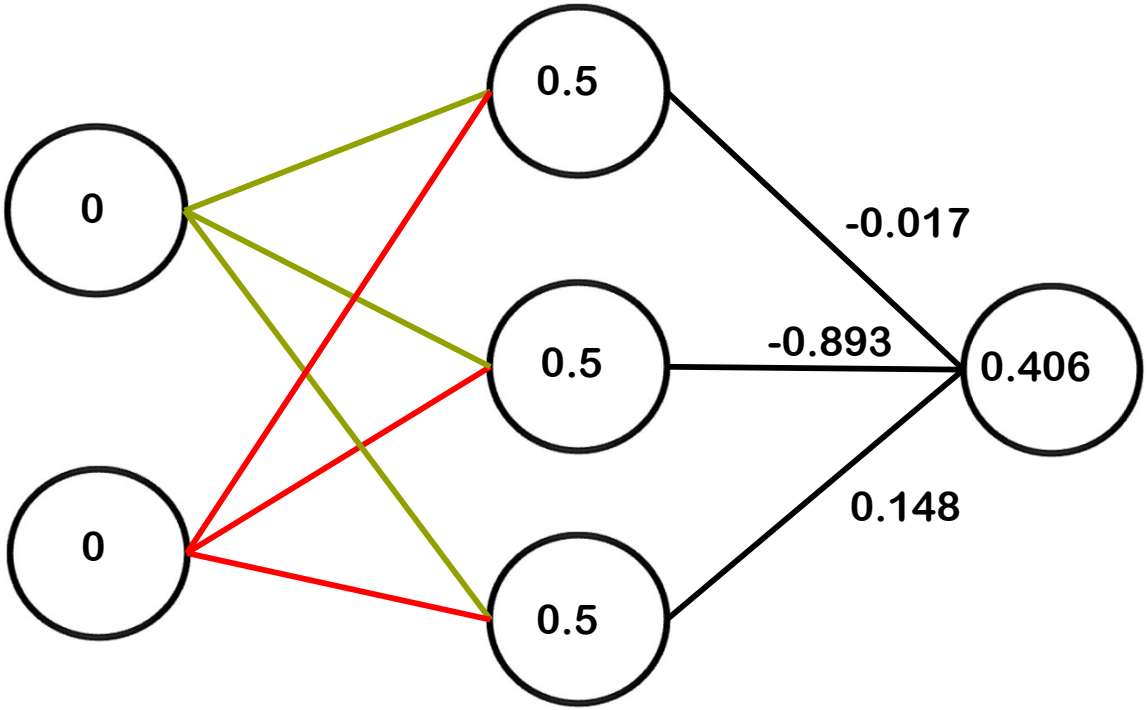
Delta camada saída



X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

Delta saída = Erro * Derivada ativação

1º Registro



X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

Delta saída = Erro * Derivada ativação

Soma = -0.381

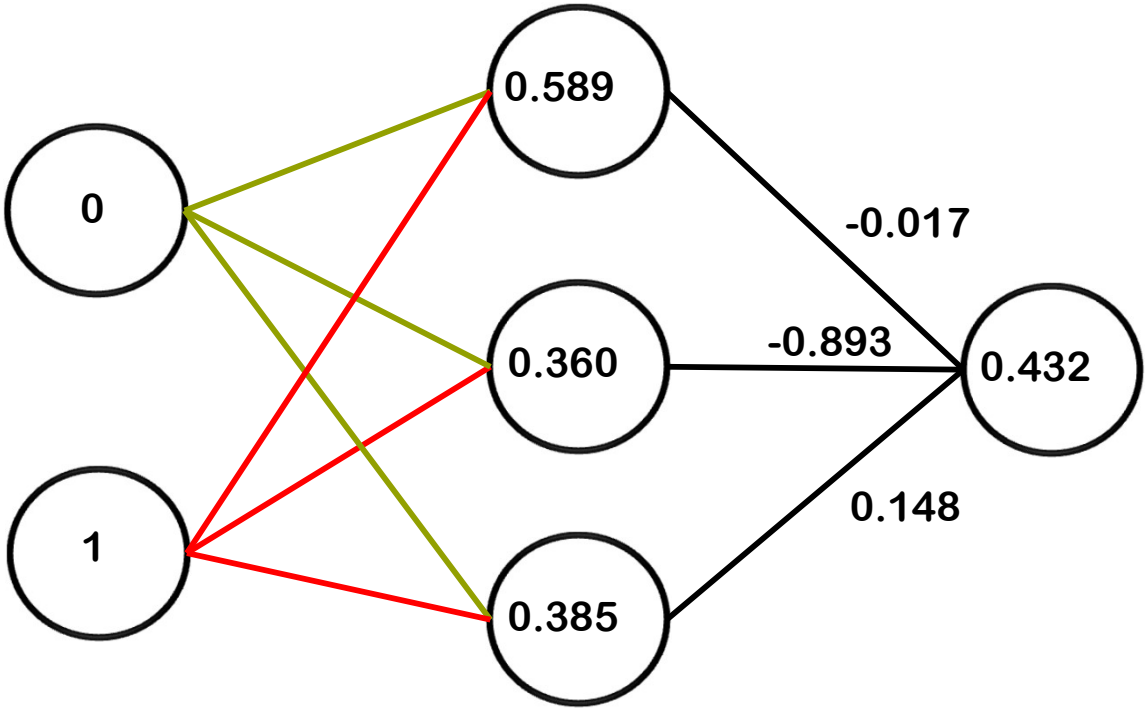
Erro = 0 - 0.406 = -0.406

Ativação = 0.406

Derivada ativação= 0.241

Delta saída = -0.406 * = -0.406 * 0.241 = **-0.098**
(indica a melhor direção para atualizar os pesos)

2º Registro



X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

Delta saída = Erro * Derivada ativação

Soma = -0.274

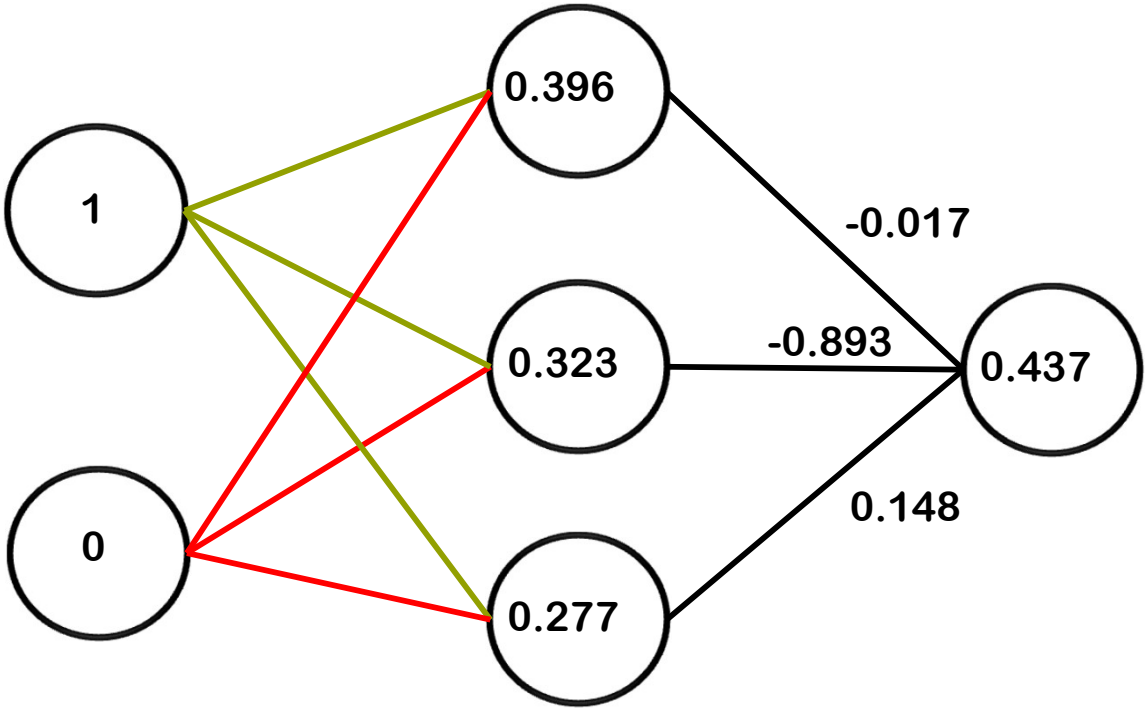
Erro = 1 - 0.432 = 0.568

Ativação = 0.432

Derivada ativação= 0.245

Delta saída = 0.568 * 0.245 = **0.139**

3º Registro



X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

Delta saída = Erro * Derivada ativação

Soma = -0.254

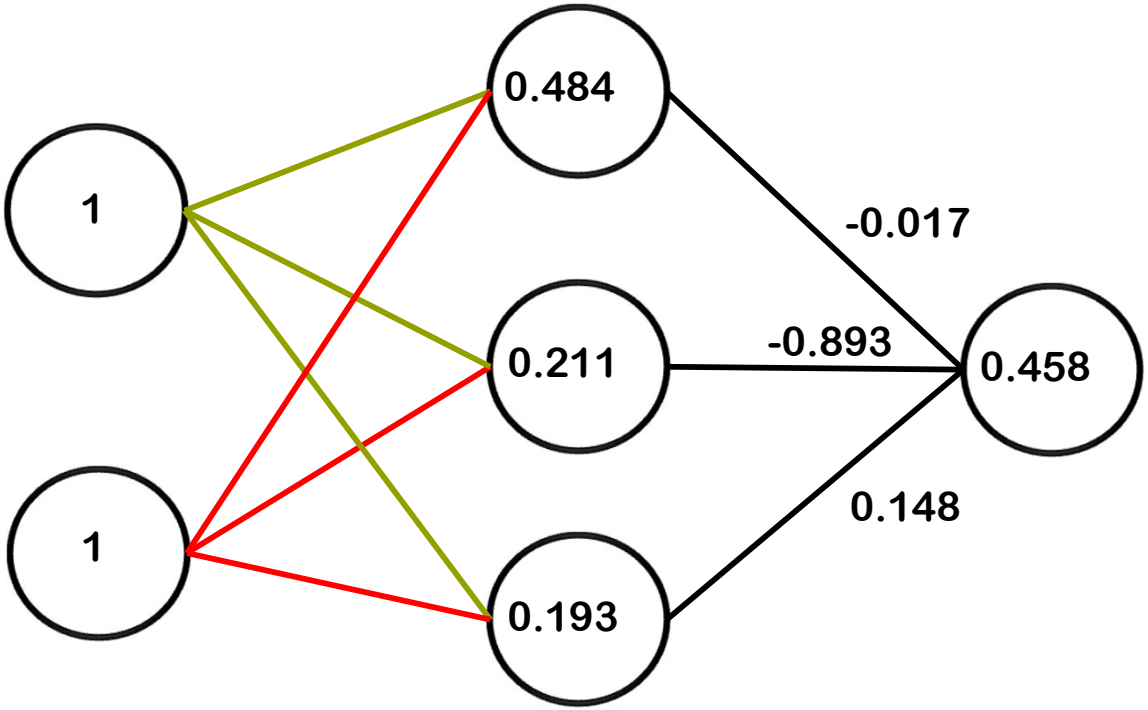
Erro = 1 - 0.437 = 0.563

Ativação = 0.437

Derivada ativação= 0.246

Delta saída = 0.563 * 0.246 = **0.139**

4º Registro



X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

Delta saída = Erro * Derivada ativação

Soma = -0.168

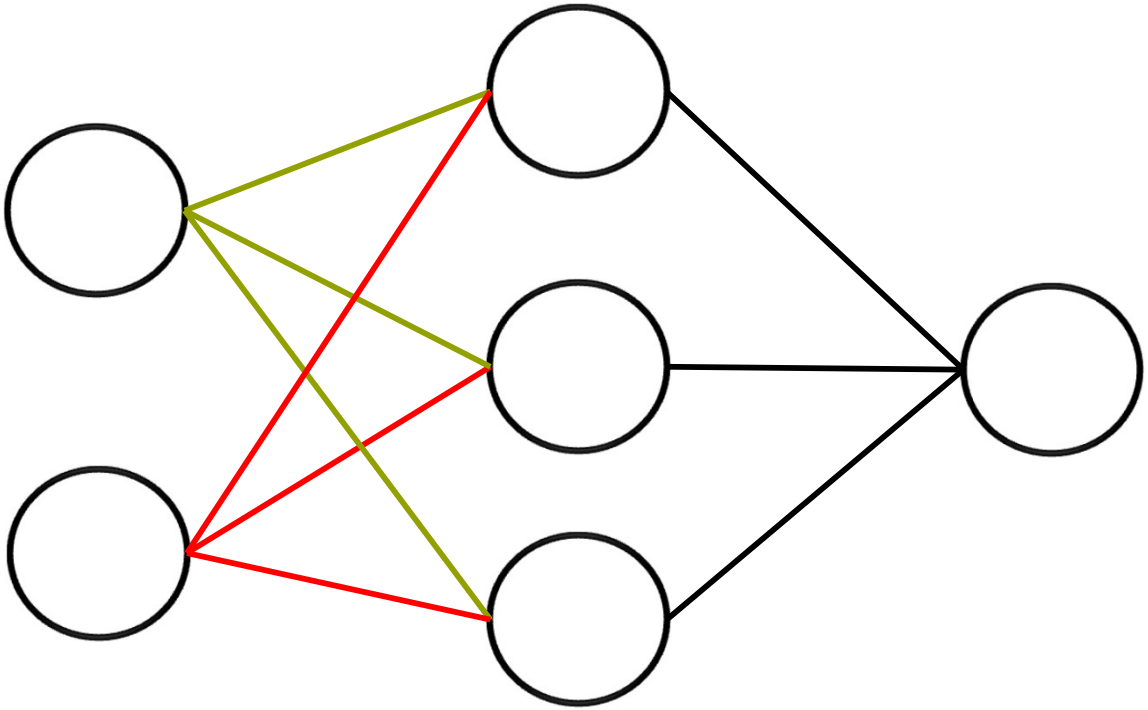
Erro = 0 - 0.458 = -0.458

Ativação = 0.458

Derivada ativação= 0.248

Delta saída = -0.458 * 0.248 = **-0.114**

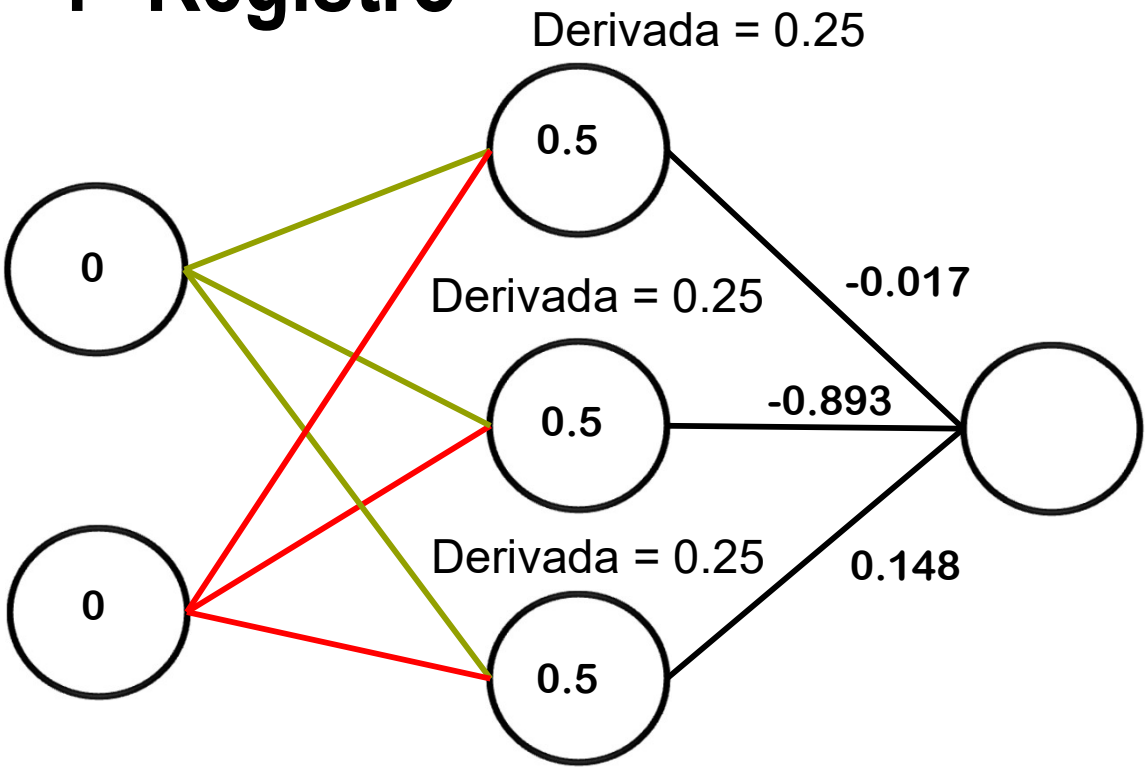
Delta camada oculta



X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

Delta oculta = Derivada ativação * peso * Delta saída

1º Registro



X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

Delta oculta = Derivada ativação * peso * Delta saída

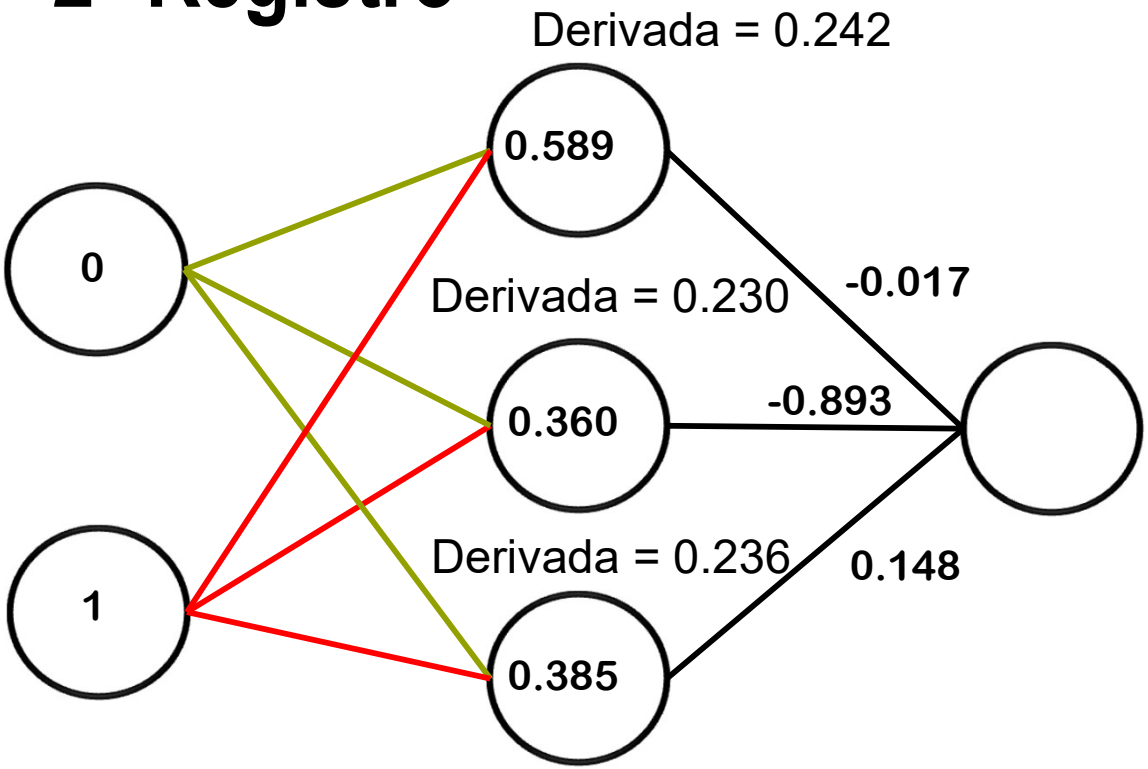
Delta saída = -0.098

Delta oculta 1 = 0.25 * (-0.017) * (-0.098) = **0.000**

Delta oculta 2 = 0.25 * (-0.893) * (-0.098) = **0.022**

Delta oculta 3 = 0.25 * 0.148 * (-0.098) = **-0.004**

2º Registro



X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

Delta oculta = Derivada ativação * peso * Delta saída

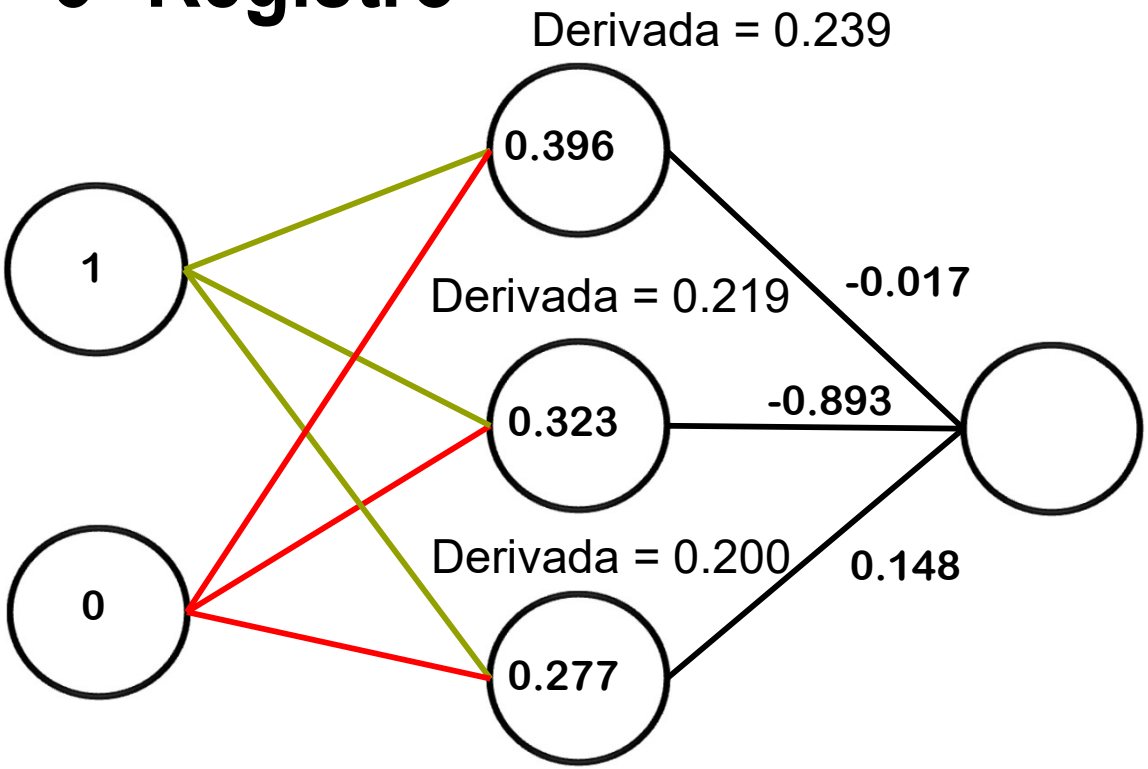
Delta saída = 0.139

Delta oculta 1 = 0.242 * (-0.017) * 0.139 = **-0.001**

Delta oculta 2 = 0.230 * (-0.893) * 0.139 = **-0.029**

Delta oculta 3 = 0.236 * 0.148 * 0.139 = **0.005**

3º Registro



X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

Delta oculta = Derivada ativação * peso * Delta saída

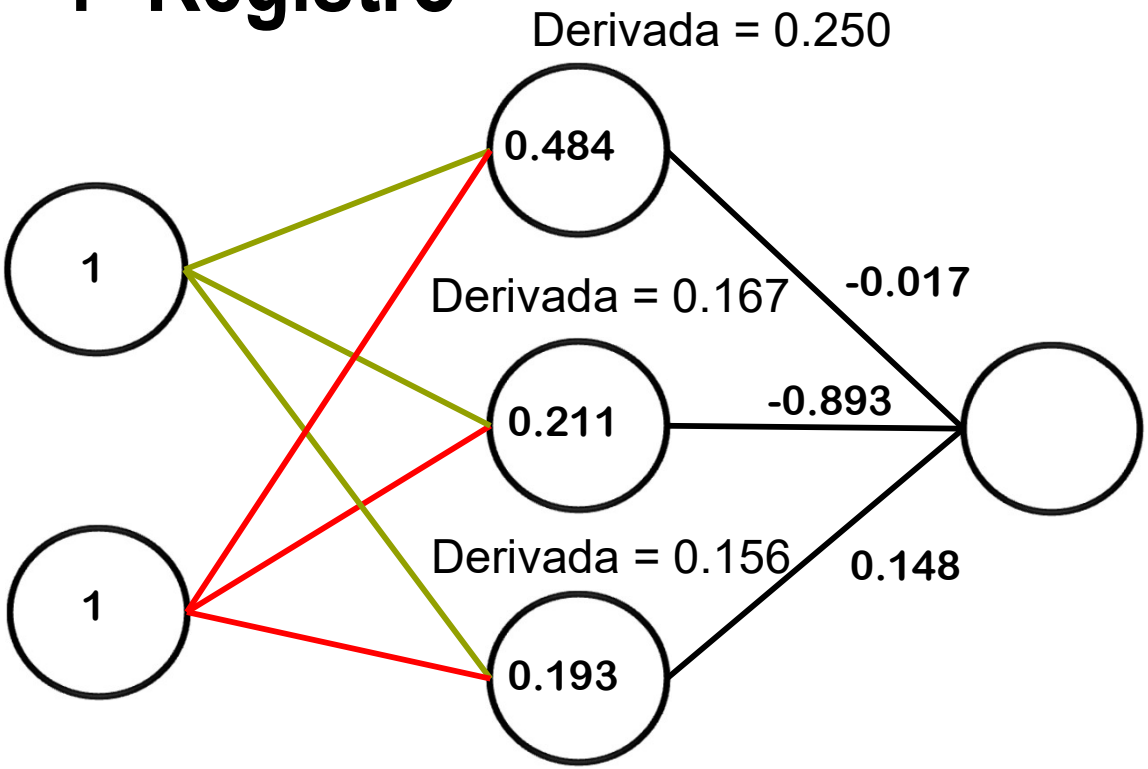
Delta saída = 0.139

Delta oculta 1 = 0.239 * (-0.017) * 0.139 = **-0.001**

Delta oculta 2 = 0.219 * (-0.893) * 0.139 = **-0.027**

Delta oculta 3 = 0.200 * 0.148 * 0.139 = **0.004**

4º Registro



X1	X2	CLASSE
0	0	0
0	1	1
1	0	1
1	1	0

Delta oculta = Derivada ativação * peso * Delta saída

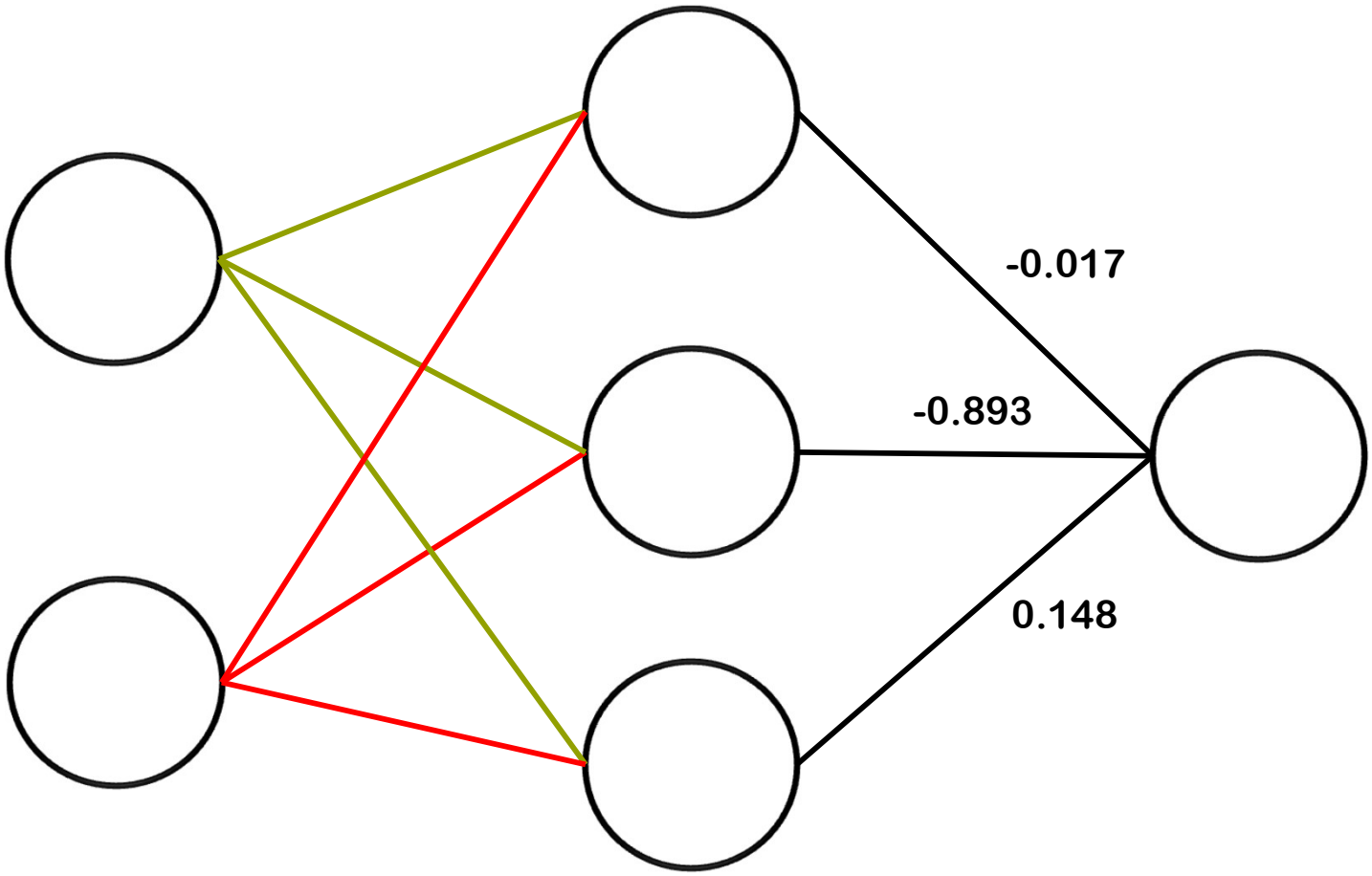
Delta saída = -0.114

Delta oculta 1 = 0.250 * (-0.017) * (-0.114) = **0.000**

Delta oculta 2 = 0.167 * (-0.893) * (-0.114) = **0.017**

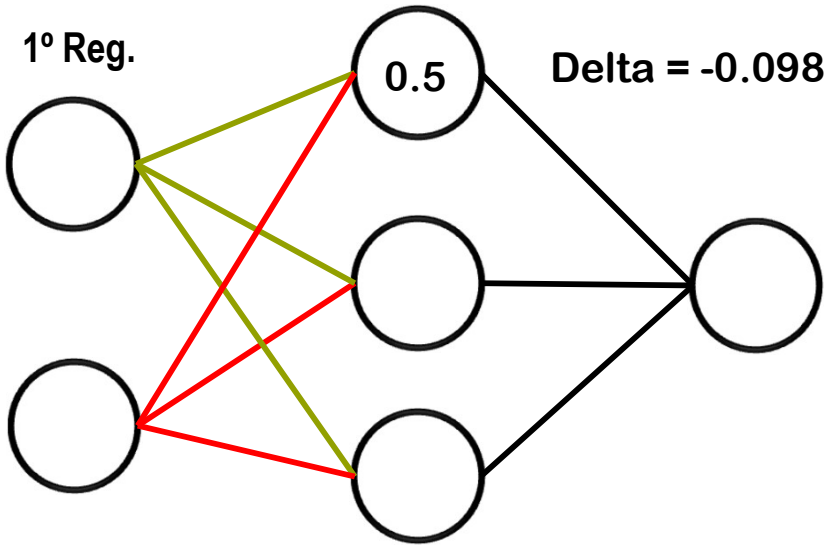
Delta oculta 3 = 0.156 * 0.148 * (-0.114) = **-0.003**

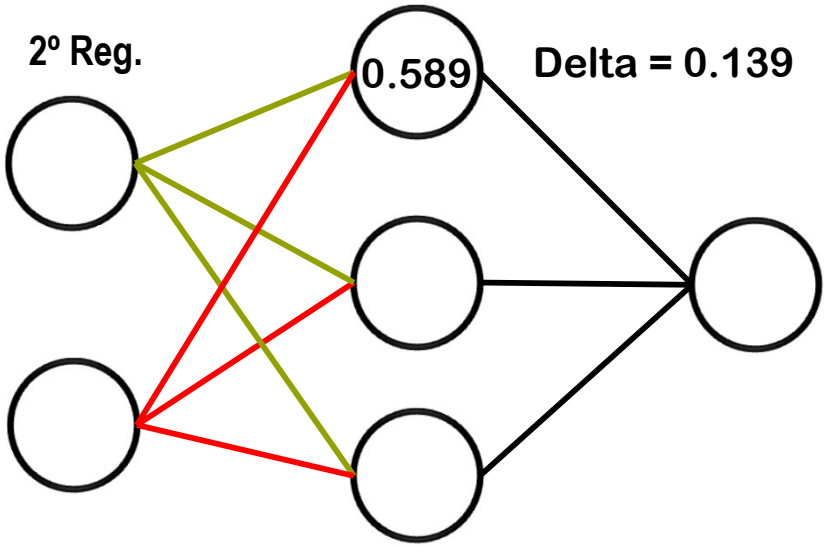
Atualização dos pesos (*backpropagation*)



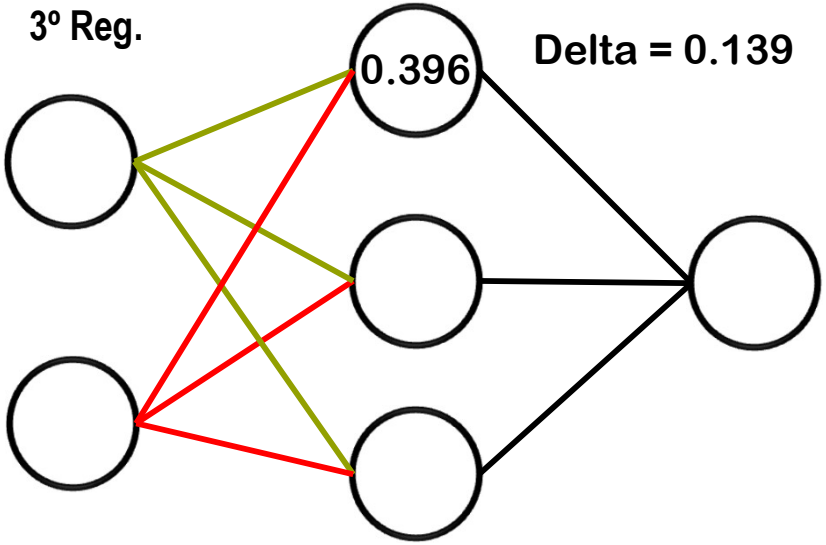
$$Peso_{n+1} = (Peso_n * momento) + (entrada * Delta\ saída * taxa\ de\ aprendizagem)$$

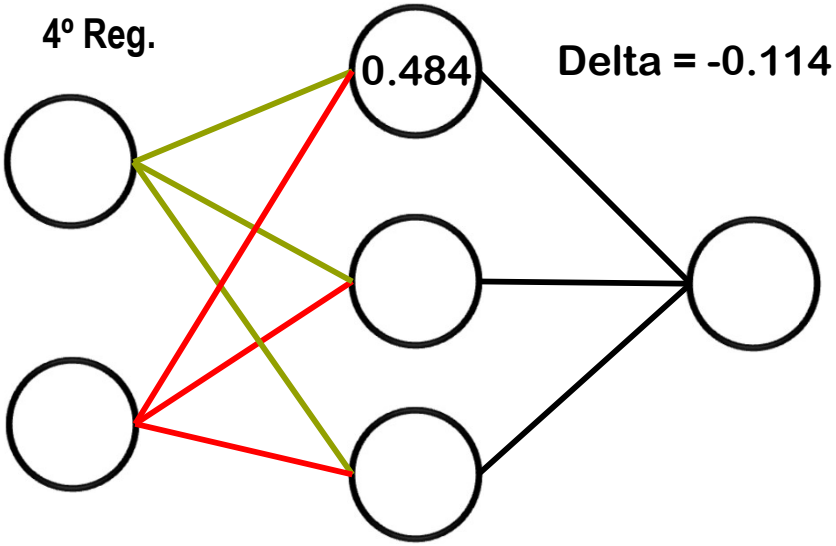
Entrada * Delta saída



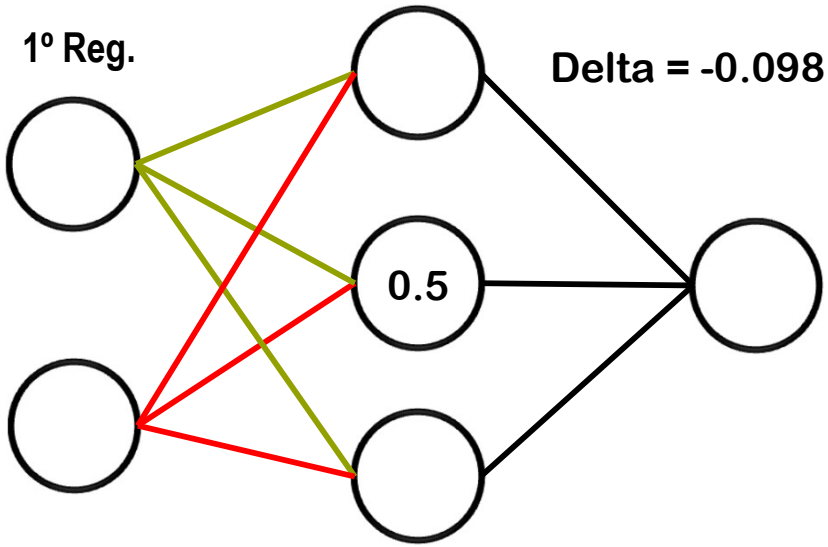


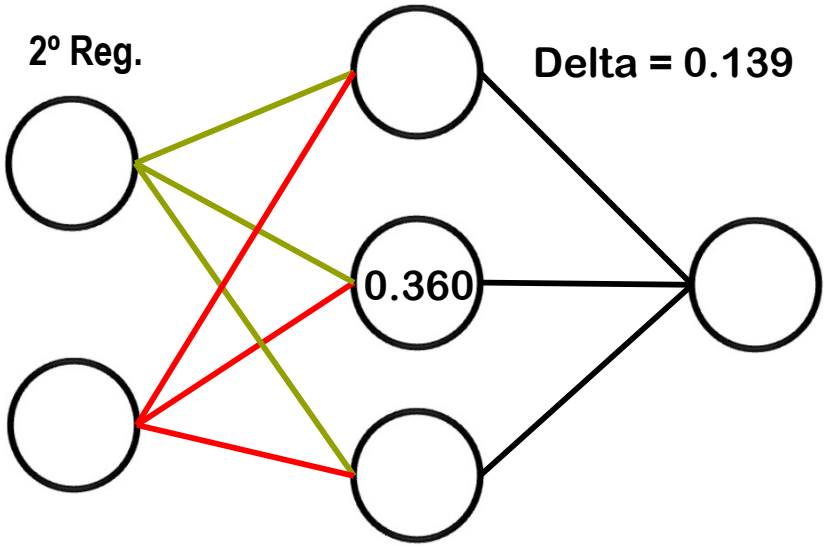
Entrada*delta 1 = 0.5 * (-0.098) + 0.589 * 0.139 + 0.396 * 0.139 + 0.484 * (-0.114) = **0.032**



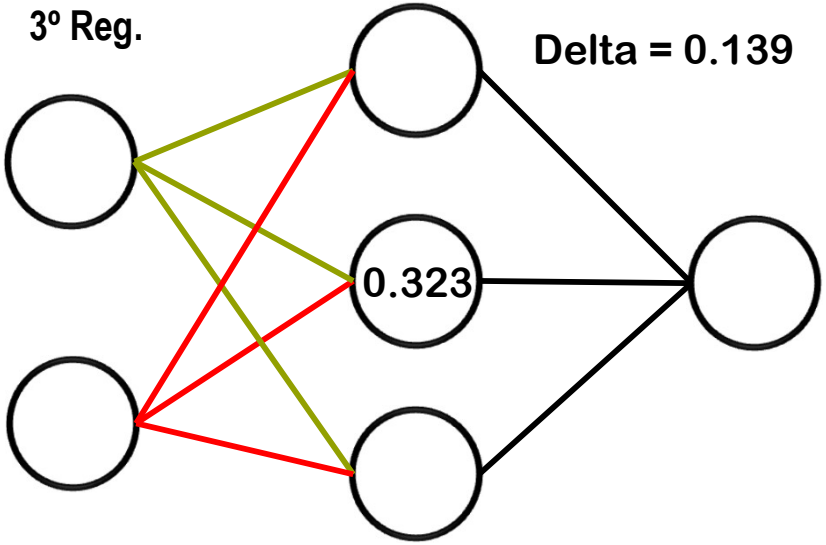


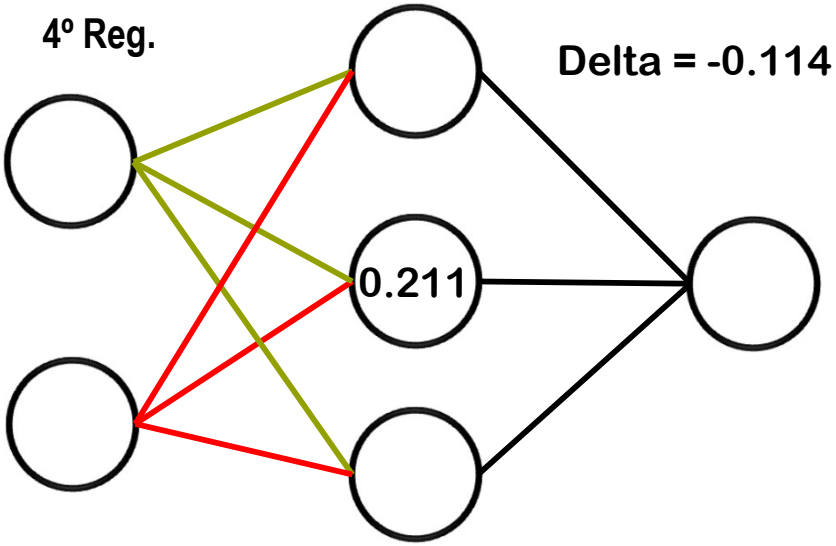
Entrada * Delta saída



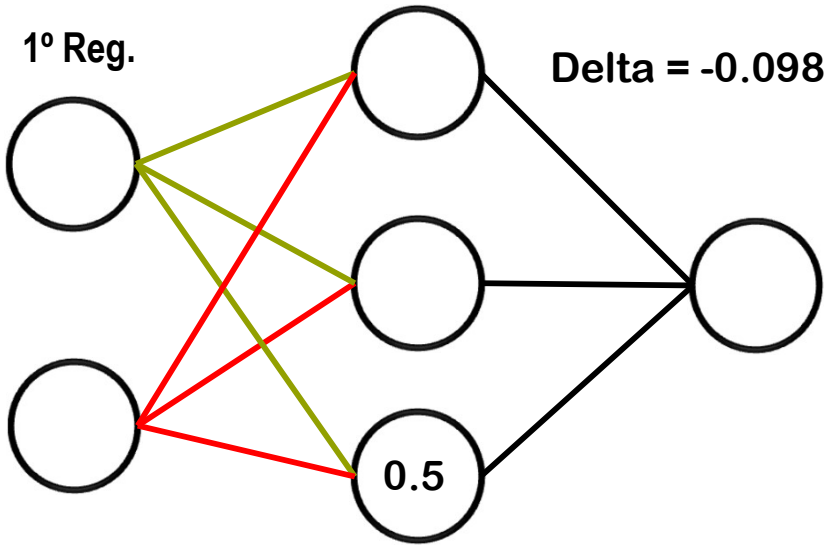


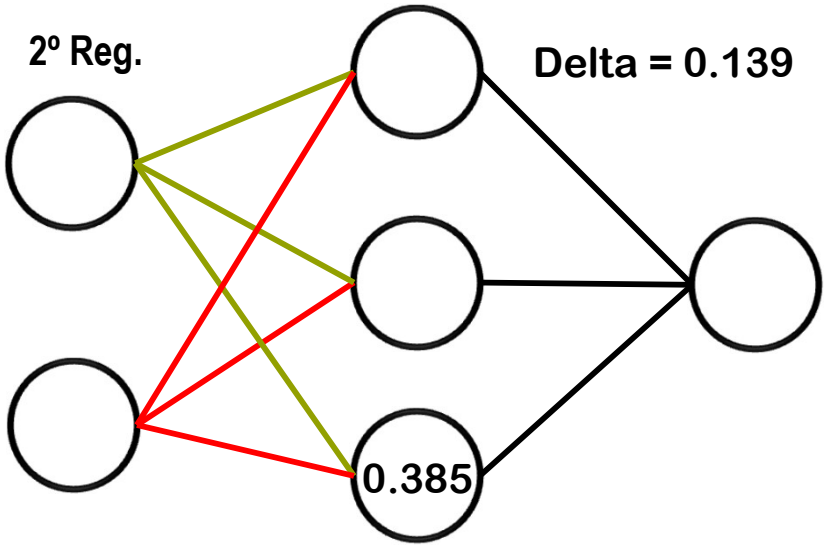
Entrada*delta 2 = 0.5 * (-0.098) + 0.360 * 0.139 + 0.323 * 0.139 + 0.211 * (-0.114) = **0.022**



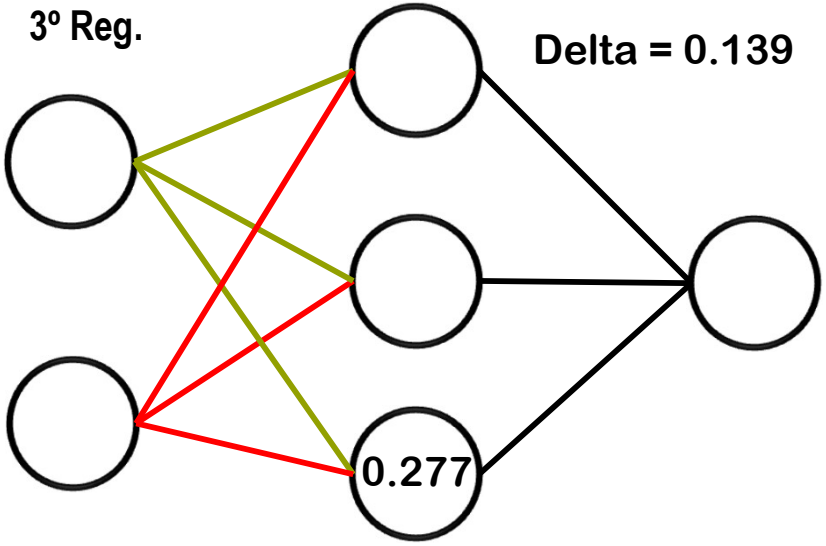


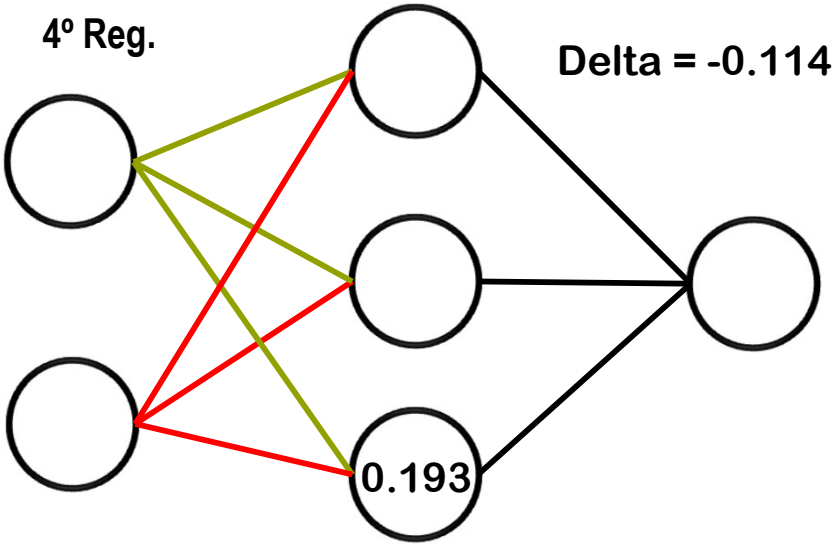
Entrada * Delta saída





Entrada*delta 3 = 0.5 * (-0.098) + 0.385 * 0.139 + 0.277 * 0.139 + 0.193 * (-0.114) = **0.021**

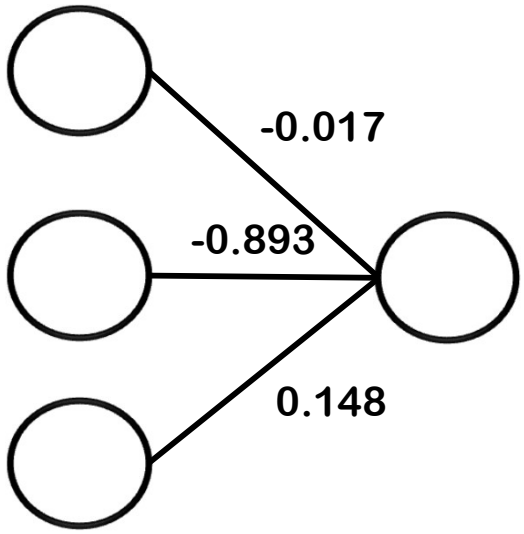




Atualização dos pesos

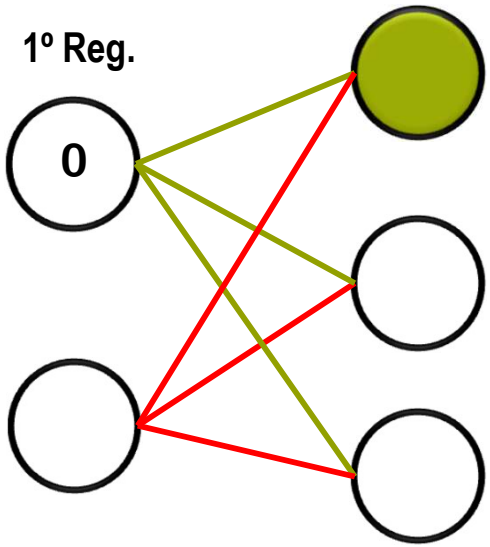
$$Peso_{n+1} = (Peso_n * momento) + (entrada * Delta\ saída * taxa\ de\ aprendizagem)$$

Taxa de aprendizagem = 0.3
Momento = 1
Entrada*delta 1 = 0.032
Entrada*delta 2 = 0.022
Entrada*delta 3 = 0.021

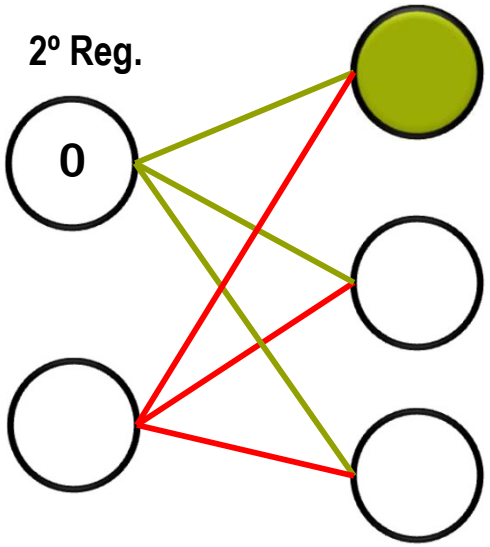


$W_1 = (-0.017*1) + 0.032 * 0.3 = -0.007$
$W_2 = (-0.893 * 1) + 0.022 * 0.3 = -0.886$
$W_3 = (0.148 * 1) + 0.021 * 0.3 = 0.154$

Entrada * Delta oculta (da 1ª Entrada para o 1º neurônio da camada oculta)

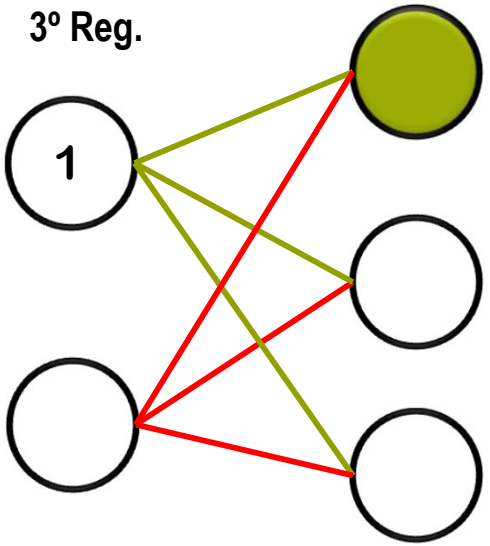


Delta = 0.000

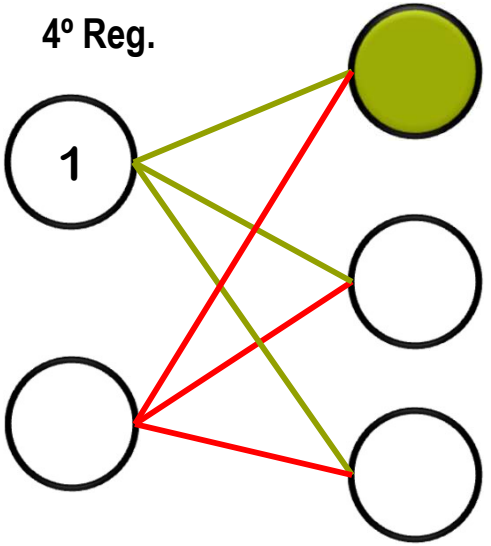


Delta = -0.001

Entrada*delta 1 = 0 * 0.000 + 0 * (-0.001) + 1 * (-0.001) + 1 * 0.000 = **-0.001**

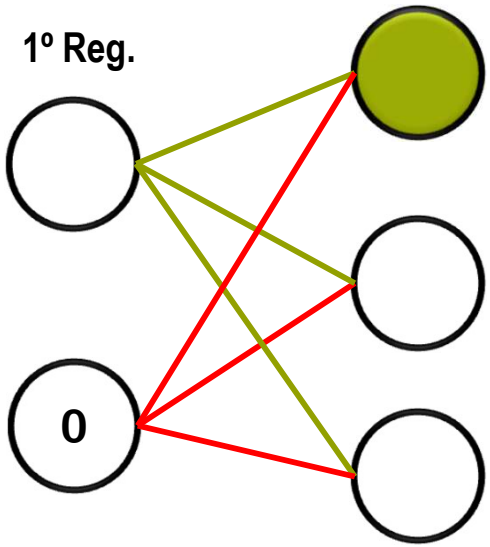


Delta = -0.001

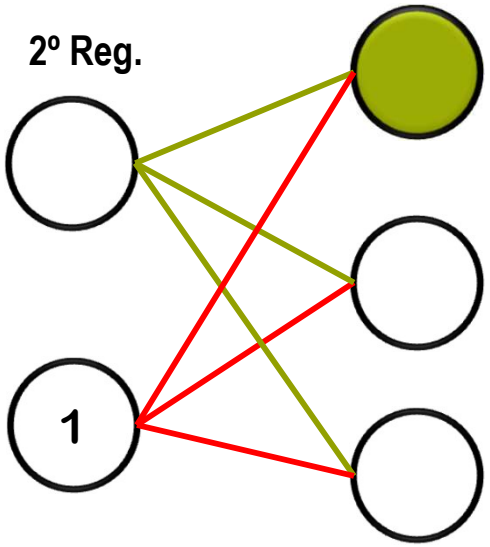


Delta = 0.000

Entrada * Delta oculta (da 2ª Entrada para o 1º neurônio da camada oculta)

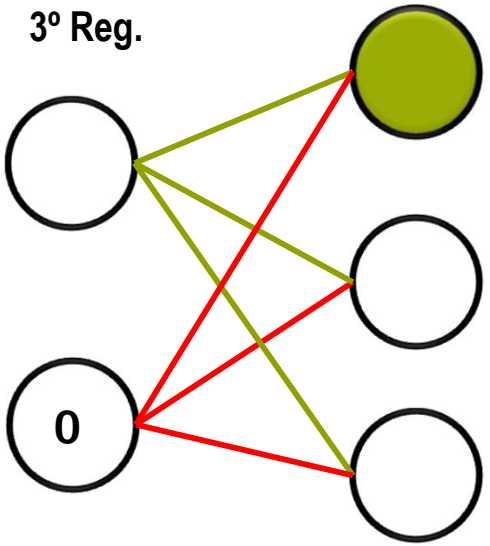


Delta = 0.000

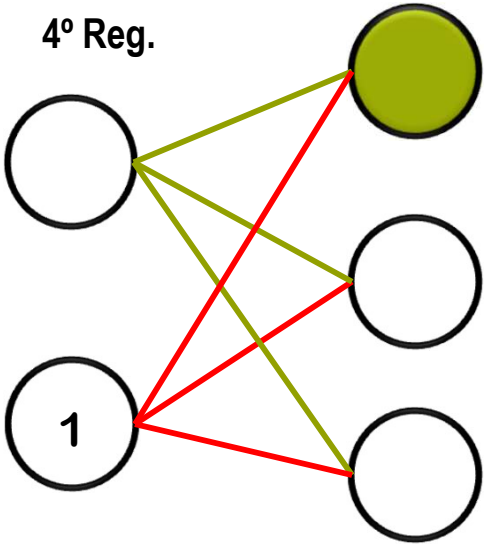


Delta = -0.001

Entrada*delta 1 = 0 * 0.000 + 1 * (-0.001) + 0 * (-0.001) + 1 * 0.000 = **-0.001**

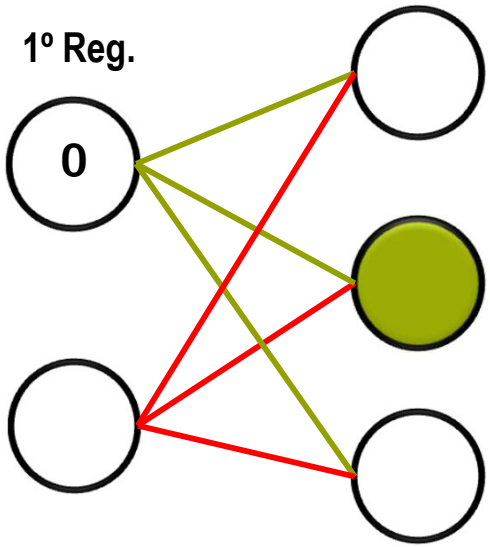


Delta = -0.001

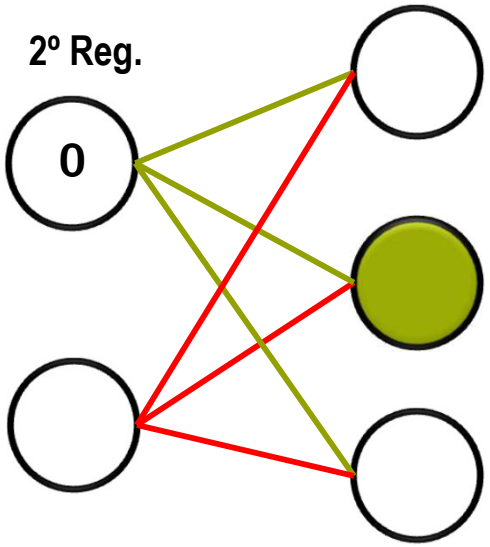


Delta = 0.000

Entrada * Delta oculta (da 1ª Entrada para o 2º neurônio da camada oculta)

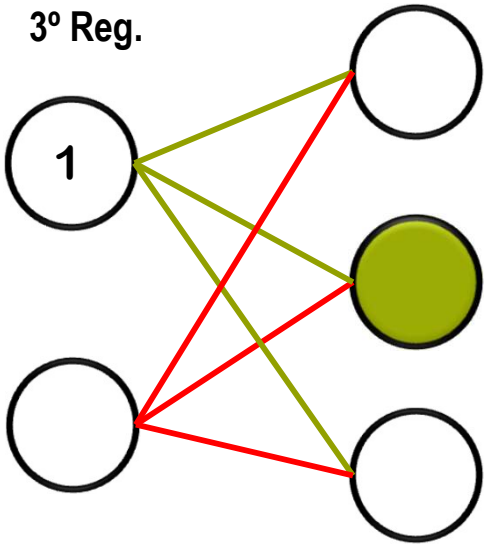


Delta = 0.022

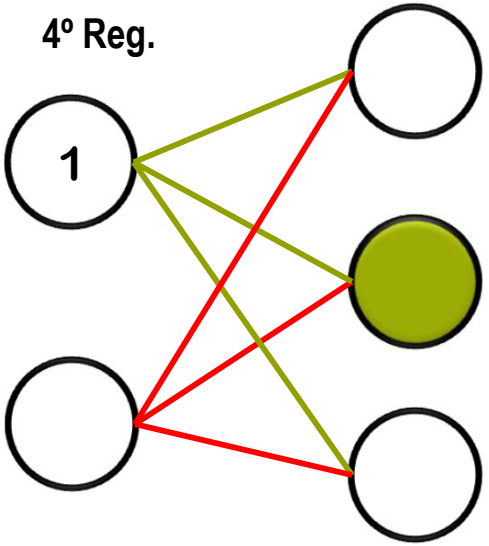


Delta = -0.029

Entrada*delta 2 = 0 * 0.022 + 0 * (-0.029) + 1 * (-0.027) + 1 * 0.017 = **-0.010**

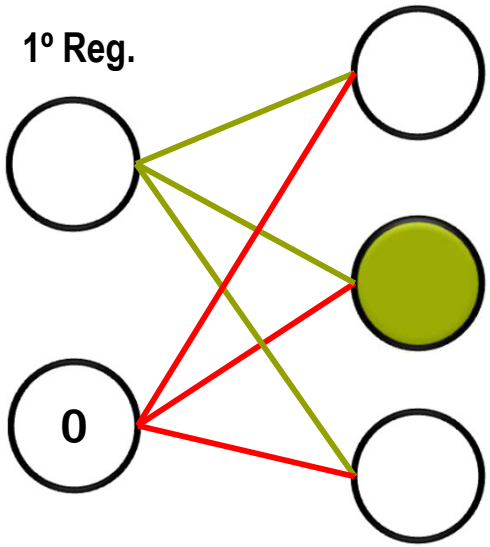


Delta = -0.027

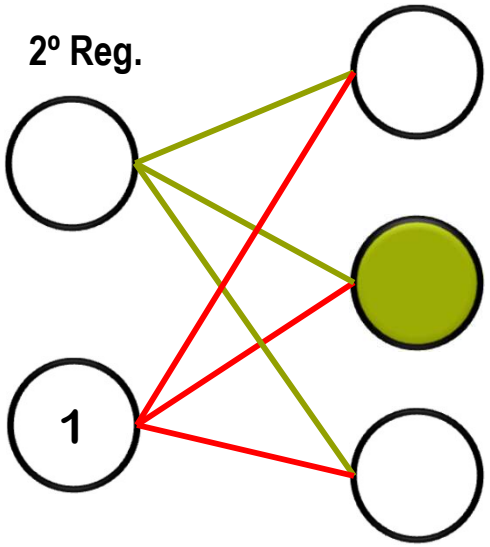


Delta = 0.017

Entrada * Delta oculta (da 2ª Entrada para o 2º neurônio da camada oculta)

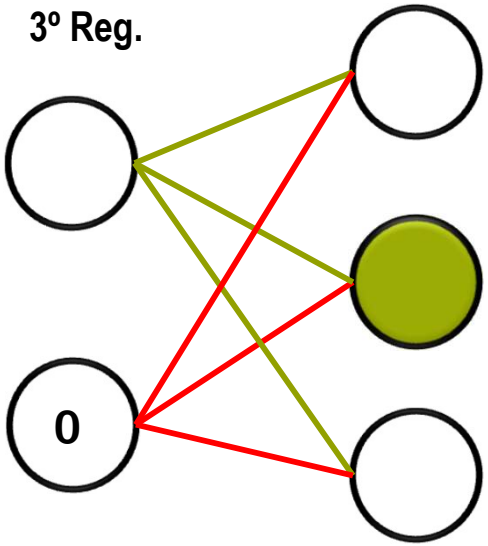


Delta = 0.022

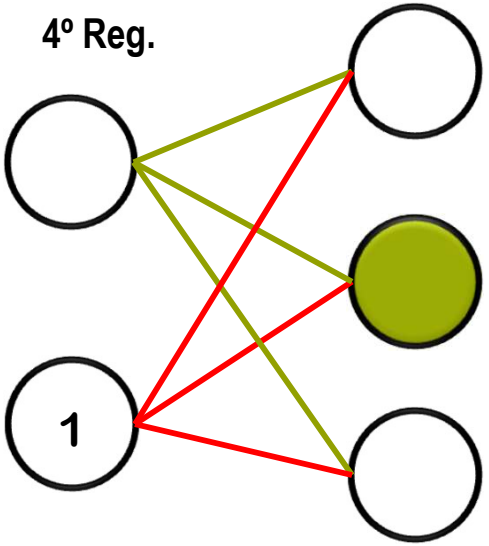


Delta = -0.029

Entrada*delta 2 = 0 * 0.022 + 1 * (-0.029) + 0 * (-0.027) + 1 * 0.017 = **-0.012**

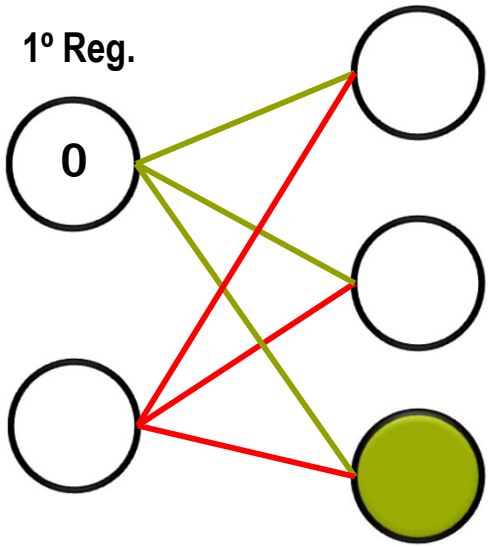


Delta = -0.027

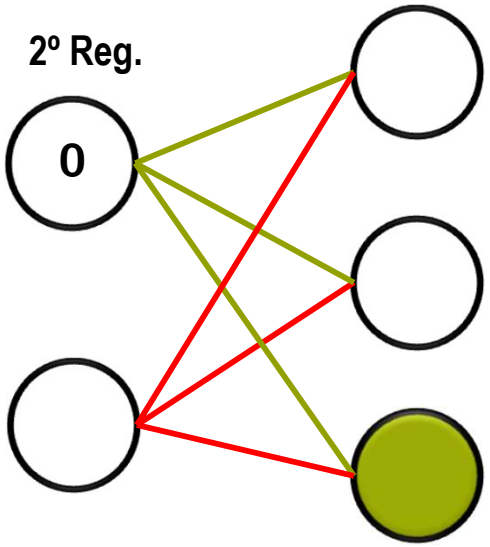


Delta = 0.017

Entrada * Delta oculta (da 1ª Entrada para o 3º neurônio da camada oculta)

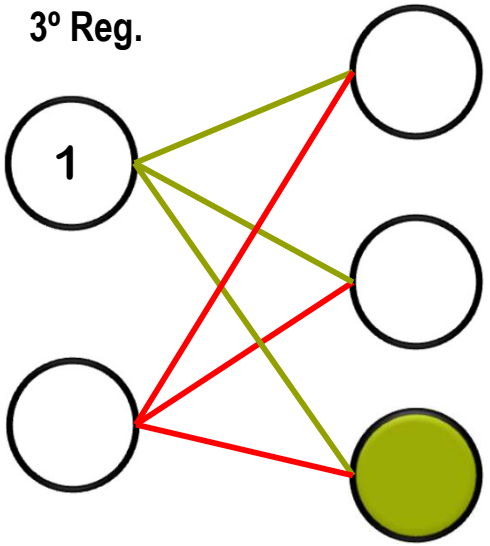


Delta = -0.004

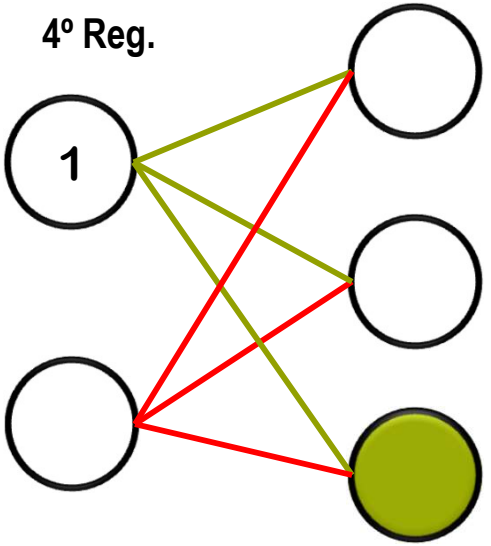


Delta = 0.005

Entrada*delta 3 = 0 * (-0.004) + 0 * 0.005 + 1 * (0.004) + 1 * (-0.003) = **0.001**

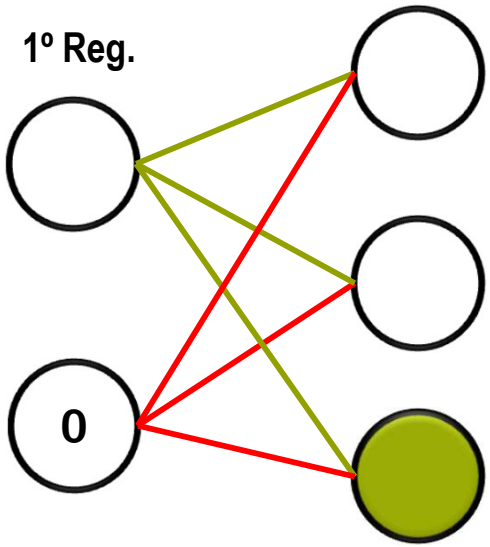


Delta = 0.004

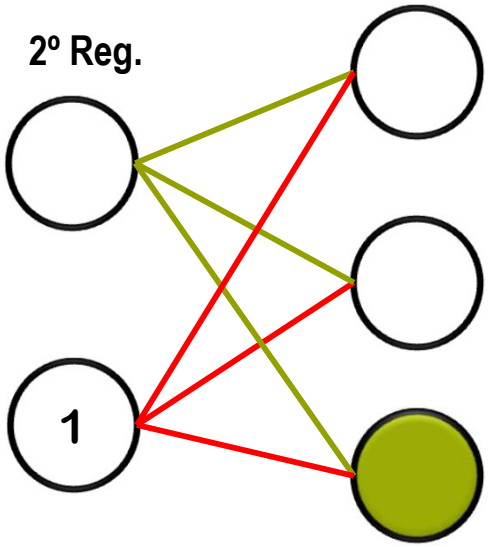


Delta = -0.003

Entrada * Delta oculta (da 2ª Entrada para o 3º neurônio da camada oculta)

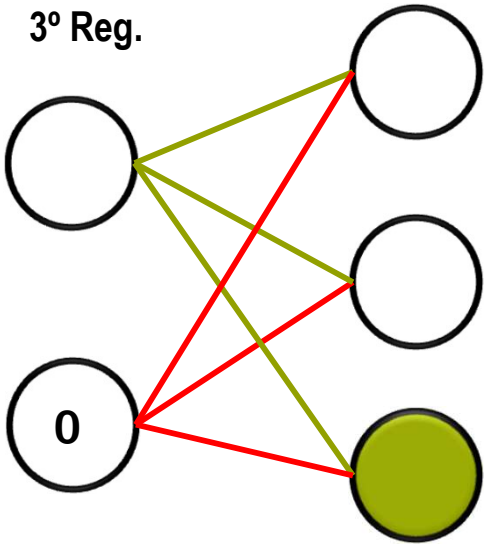


Delta = -0.004

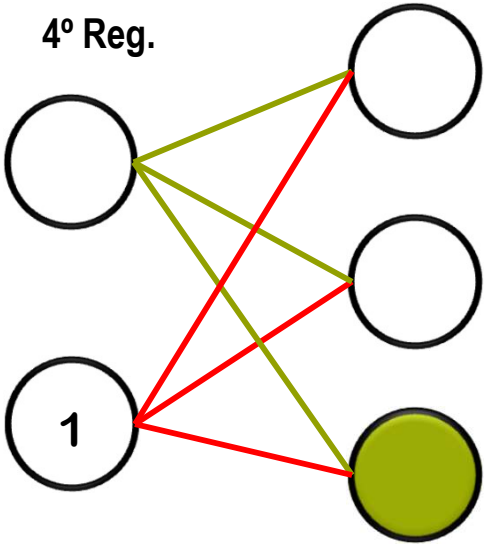


Delta = 0.005

Entrada*delta 3 = 0 * (-0.004) + 1 * 0.005 + 0 * (0.004) + 1 * (-0.003) = **0.002**



Delta = 0.004



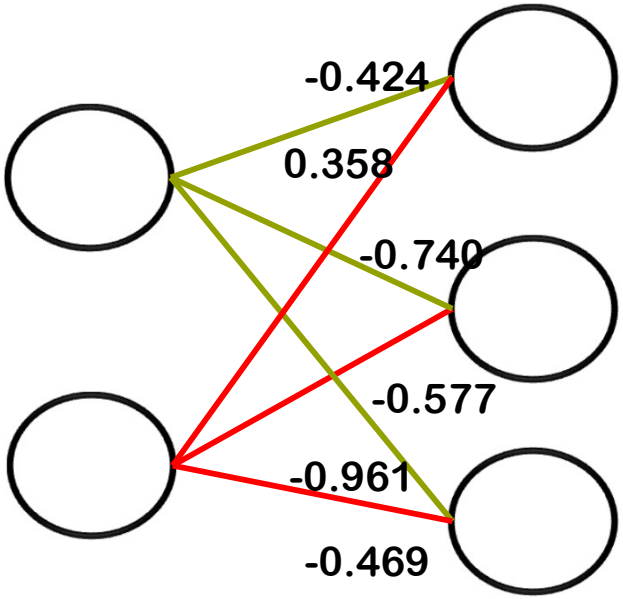
Delta = -0.003

Atualização dos pesos

$$Peso_{n+1} = (Peso_n * momento) + (entrada * Delta\ saída * taxa\ de\ aprendizagem)$$

Taxa de aprendizagem = 0.3
Momento = 1
Entrada*delta n1 = -0.000; -0.000
Entrada*delta n2 = -0.010; -0.012
Entrada*delta n3 = 0.001; 0.002

$W_1 = (-0.424 * 1) + (-0.001 * 0.3) =$ **-0.4243**
$W_2 = (0.358 * 1) + (-0.001 * 0.3) =$ **0.3577**
$W_3 = (-0.740 * 1) + (-0.010 * 0.3) =$ **-0.743**
$W_4 = (-0.577 * 1) + (-0.012 * 0.3) =$ **-0.581**
$W_5 = (-0.961 * 1) + (0.001 * 0.3) =$ **-0.961**
$W_6 = (-0.469 * 1) + (0.002 * 0.3) =$ **-0.468**



Taxa de aprendizagem e momento

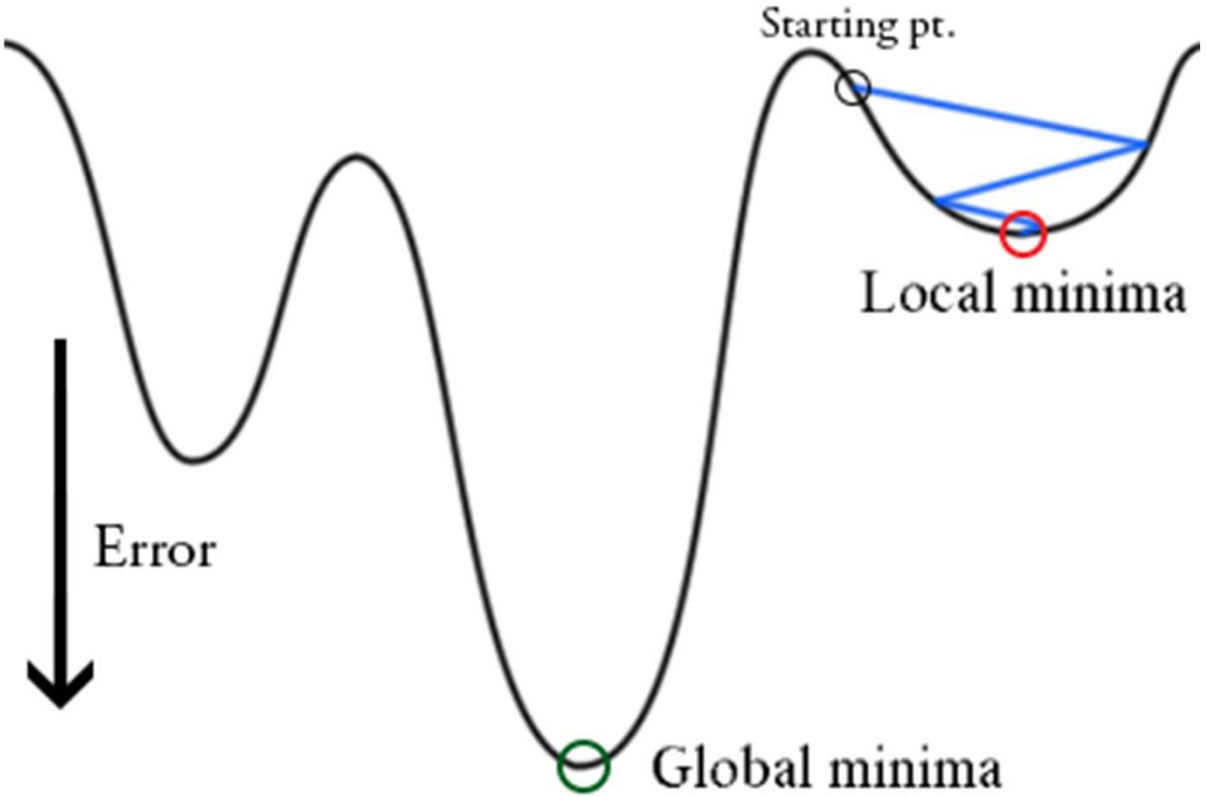
Taxa de aprendizagem:

- Define o quão rápido o algoritmo vai aprender.
- Alto: convergência é mais rápida mas pode perder um mínimo global.
- Baixo: o processo será mais lento mas tem mais chances de chegar no mínimo global.

Momento

- Escapar de mínimos locais.
- Define o quão confiável foi a última alteração.

Momento e taxa de aprendizagem



https://thinkingandcomputing.com

Etapas para desenvolver uma RNA

1 – Coleta de dados

- Os dados devem ser significativos e cobrir amplamente o domínio do problema e também as exceções.

2 – Separação em conjuntos

- Dados treinamento, dados de teste e dados de validação (verificar a performance sob condições reais de utilização).

Etapas para desenvolver uma RNA

3 – Configuração da rede

- Seleção do paradigma neural (Supervisionado ou Não-Supervisionado)
- Definição da topologia da rede
- Definição de parâmetros do algoritmo de treinamento e funções de ativação

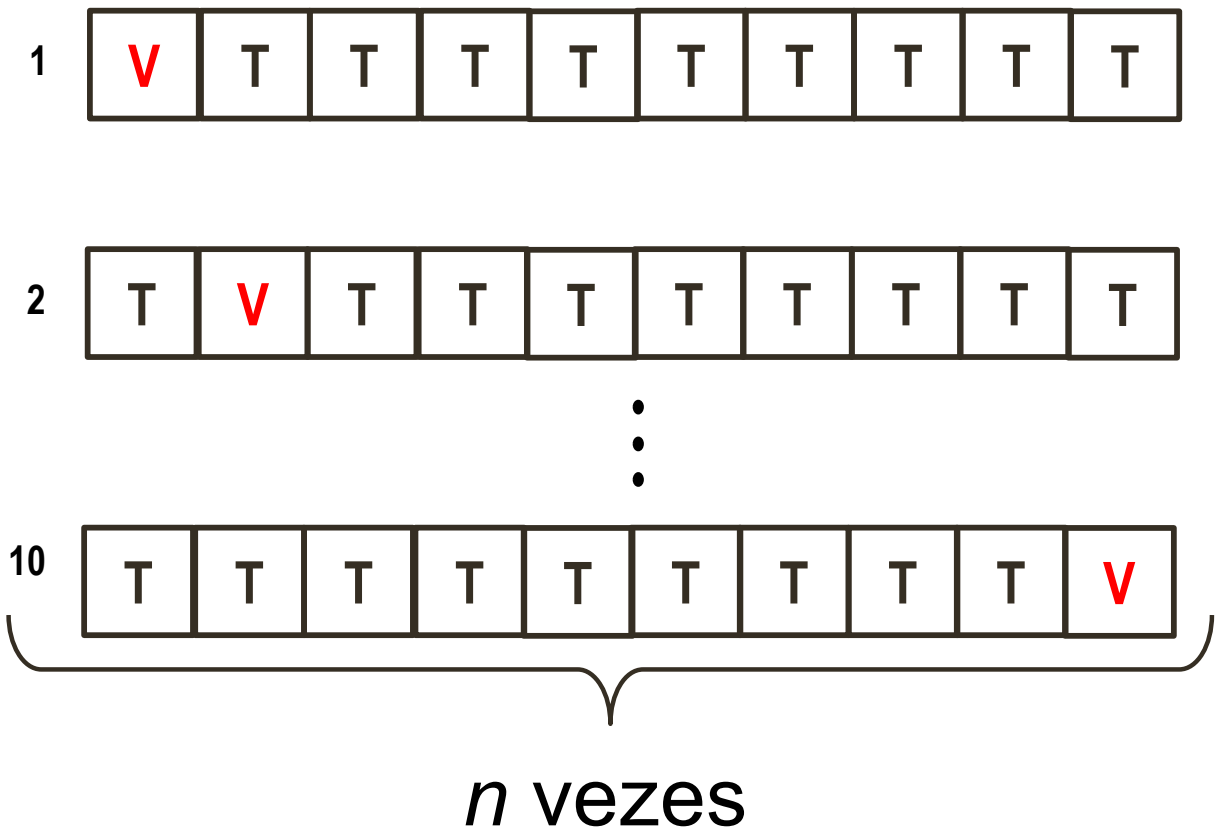
4 – Treinamento

- Emprego do algoritmo de treinamento
- Ajuste do peso das conexões

5 – Validação ou teste

- Determinar a performance da rede, medida nessa fase, é uma boa indicação de sua performance real.

VALIDAÇÃO CRUZADA



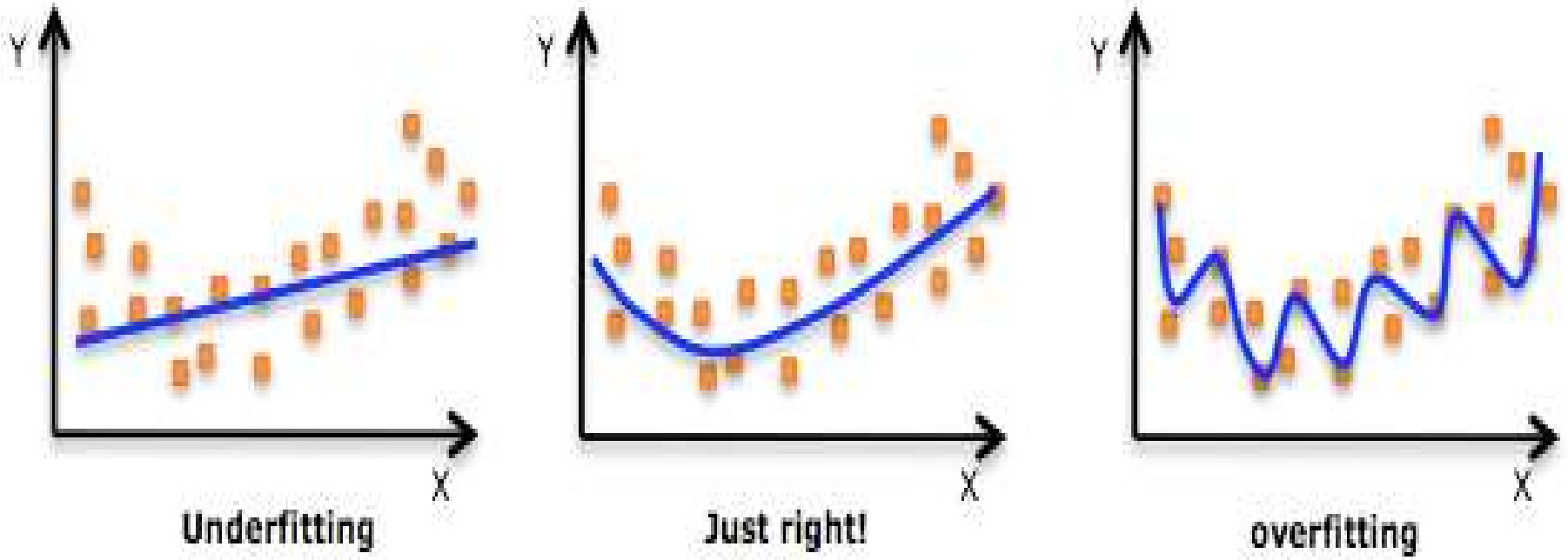
MATRIZ DE CONFUSÃO

	Doente positivo	Doente negativo
Doente positivo	Verdadeiro positivo	Falso negativo
Doente negativo	Falso positivo	Verdadeiro negativo

PROBLEMAS

Principais problemas

- Sobre ajuste (*overfitting*).
- Falta de ajuste (*underfitting*).



- Para simular graficamente as redes neurais você poderá acessar o seguinte link (http://playground.tensorflow.org/). Veja o que acontece quando você adicionar mais camadas (layers), neurônios, dados de entrada, etc.

INSTITUTO FEDERAL
Sudeste de Minas Gerais