TensorFlow
User Group São Paulo

TensorFlow T1|E1

Nessa primeira temporada de Tensorflow 2022 teremos 8 episódios. Temporada será focada no público iniciante, no qual começaremos com os conceitos básicos e chegaremos no nível intermediário.

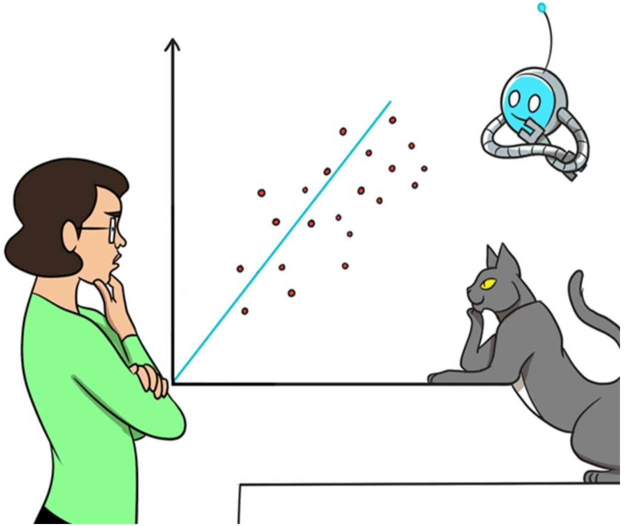
Episódio 1: Introdução redes neurais

Alex Mansano

Pedro Gengo

Vinicius Caridá

Introdução a Redes Neurais

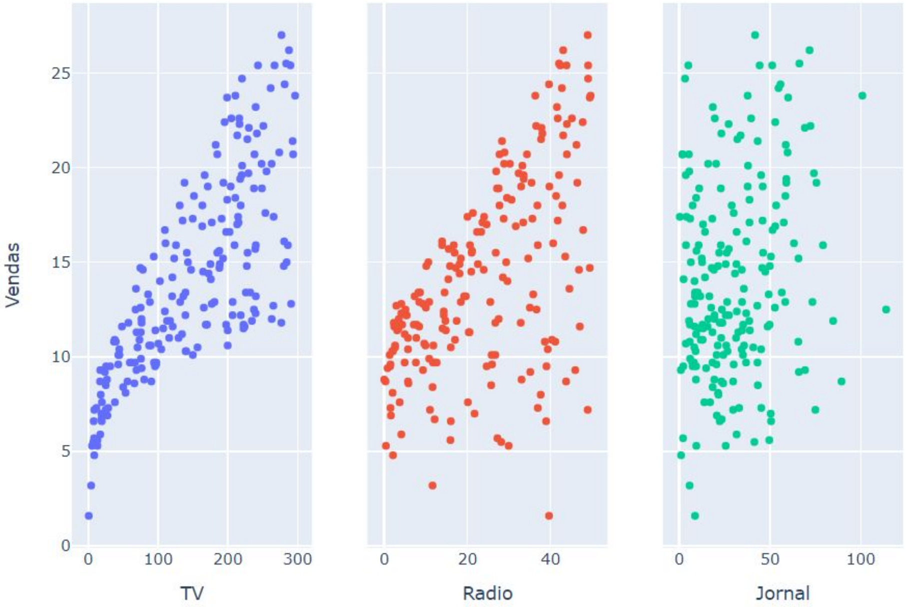


Agenda

- Regressão Linear
  - Definição do problema
  - Definição da regressão linear
  - Erro do modelo
  - Aprendizado do modelo
- Redes Neurais
  - Definição
  - Capacidade de classificação
  - Feedforward
  - Backpropagation

Definição do problema

Trabalhamos em uma empresa de marketing e queremos entender qual a relação entre o valor gasto em propagandas feitas em TV, rádio e jornal e as vendas da empresa.



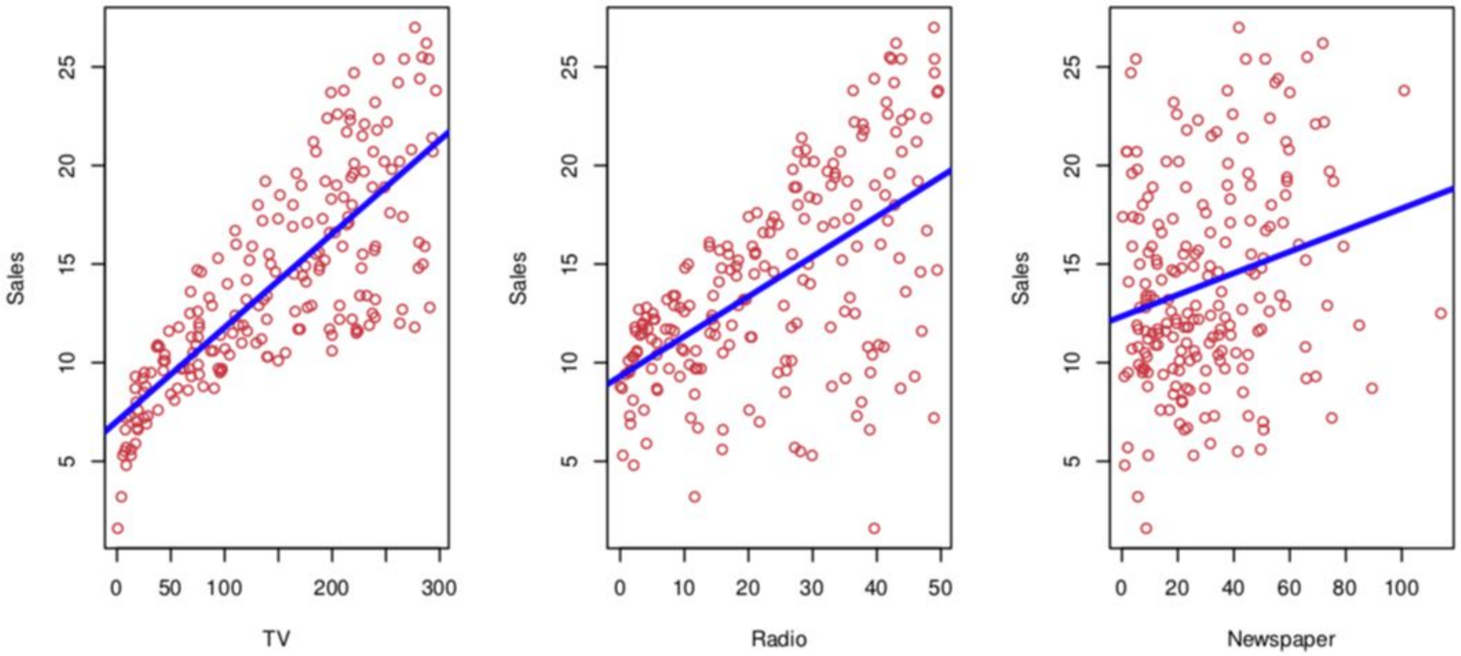
Definição do problema

Se eu investir 200 reais em propaganda na TV, quanto eu espero ganhar em vendas?

Quero encontrar uma “fórmula mágica” que relacione os valores de investimentos nos canais de mídia com o quanto vou vender. Algo assim:

Vendas = 1,234 * TV + 0,763 * Radio + 2,344 * Jornal + 1,17

Definição do problema



Entendendo um pouco mais a equação

Vamos reescrever nossa fórmula:

Vendas = c0 * TV + c1 * Radio + c2 * Jornal + b

Onde,

- c0, c1 e c2 são os coeficientes de cada tipo de mídia
- b é o nosso resultado quando os investimentos forem 0 para TV, Radio e Jornal.

Entendendo um pouco mais a equação

Vendas = 1,234 * TV + 0,763 * Radio + 2,344 * Jornal + 1,17

- Investimento em TV = 0
- Investimento em Radio = 0
- Investimento em Jornal = 0

Venda = ?

Entendendo um pouco mais a equação

Vendas = 1,234 * TV + 0,763 * Radio + 2,344 * Jornal + 1,17

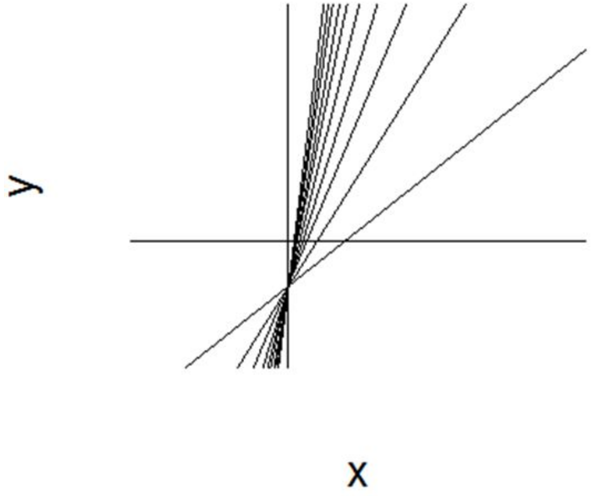
- Investimento em TV = 20
- Investimento em Radio = 100
- Investimento em Jornal = 80

Venda = ?

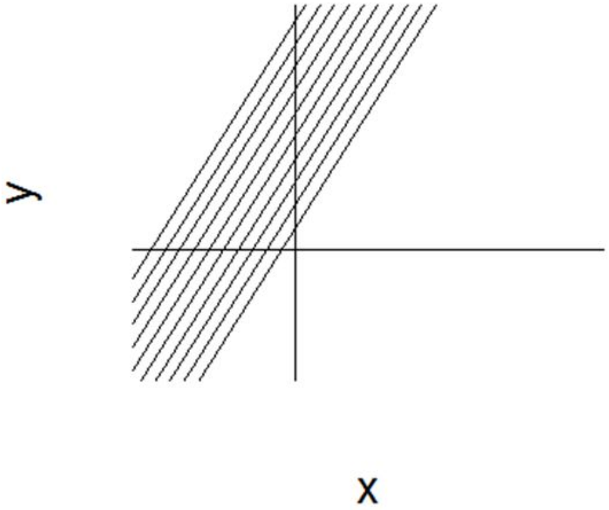
Entendendo um pouco mais a equação

$$Y = c0 * X_0 + b$$

c0 variável, *b* fixo



c0 fixo, *b* variável



Regressão Linear

- Aprendizado supervisionado
- Assume uma dependência linear entre a variável resposta Y e os valores $X_0, X_1, ..., X_n$.
- Assume-se o modelo:

$$Y = c0 * X_0 + c1 * X_1 + \dots + cn * X_n + b$$

Onde os coeficientes c0, c1, .. cn, b são aprendidos pelo modelo.

Mas como encontrar essa fórmula mágica?

Erro do modelo

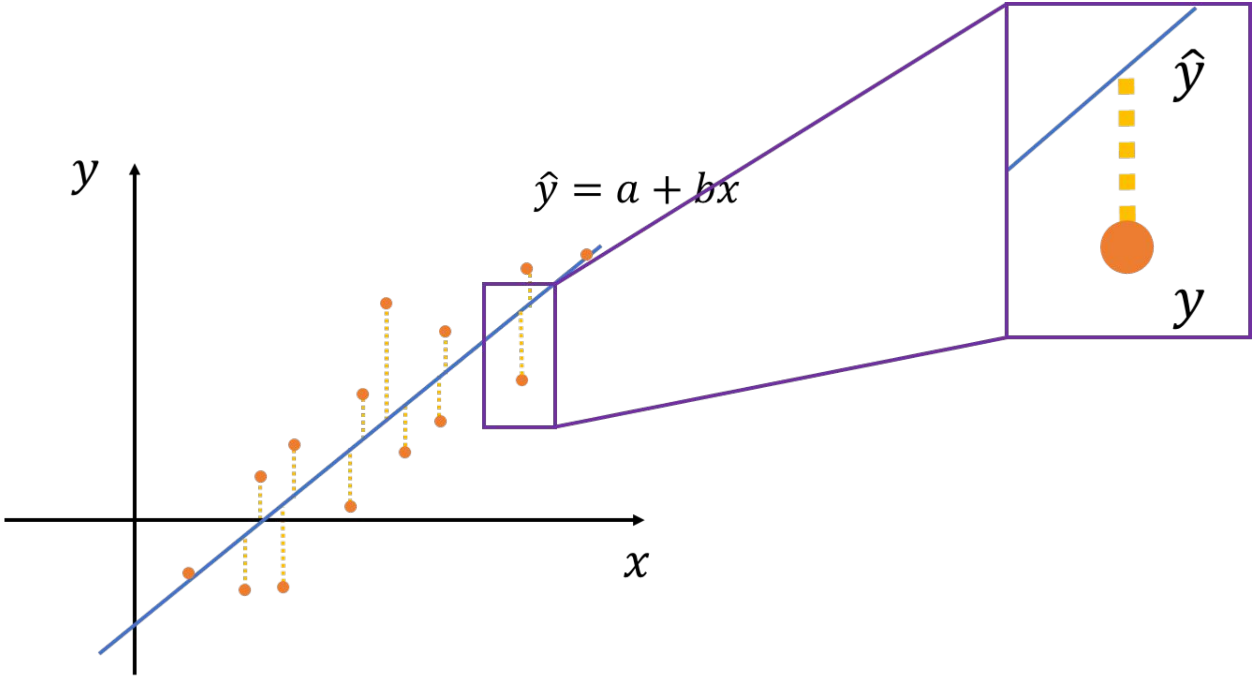
Se estamos querendo APRENDER os coeficientes, muito provavelmente vamos testar alguns valores para eles. Mas como saber qual deles é o melhor?

Erro do modelo

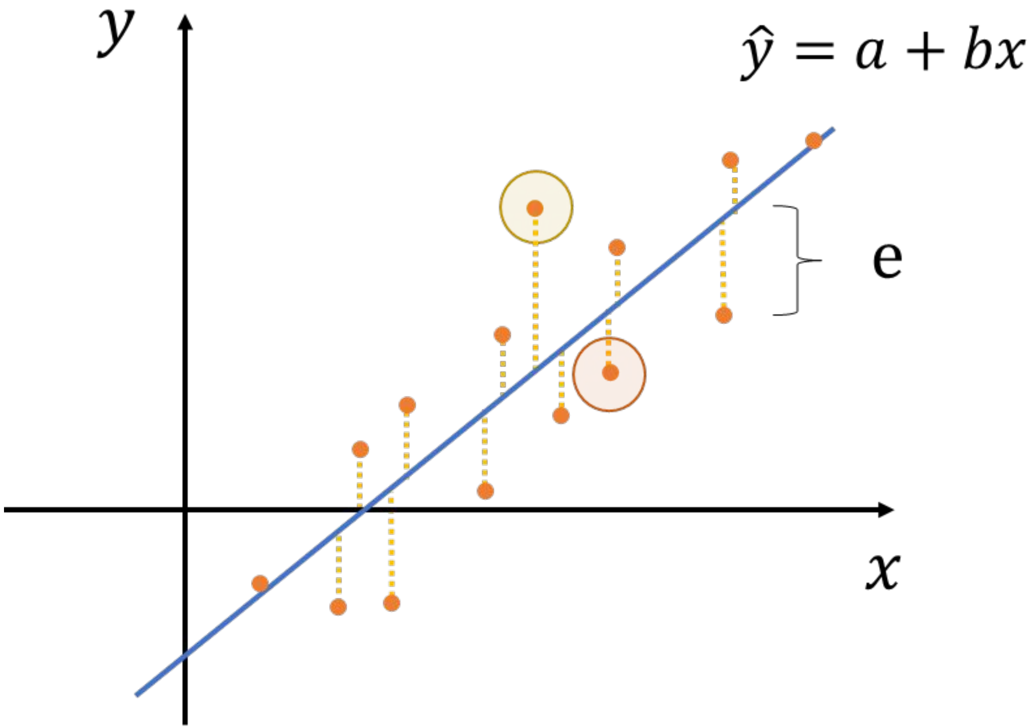
Se estamos querendo APRENDER os coeficientes, muito provavelmente vamos testar alguns valores para eles. Mas como saber qual deles é o melhor?

➔ PRECISAMOS MEDIR O QUANTO ESTAMOS ERRANDO!

Erro do modelo

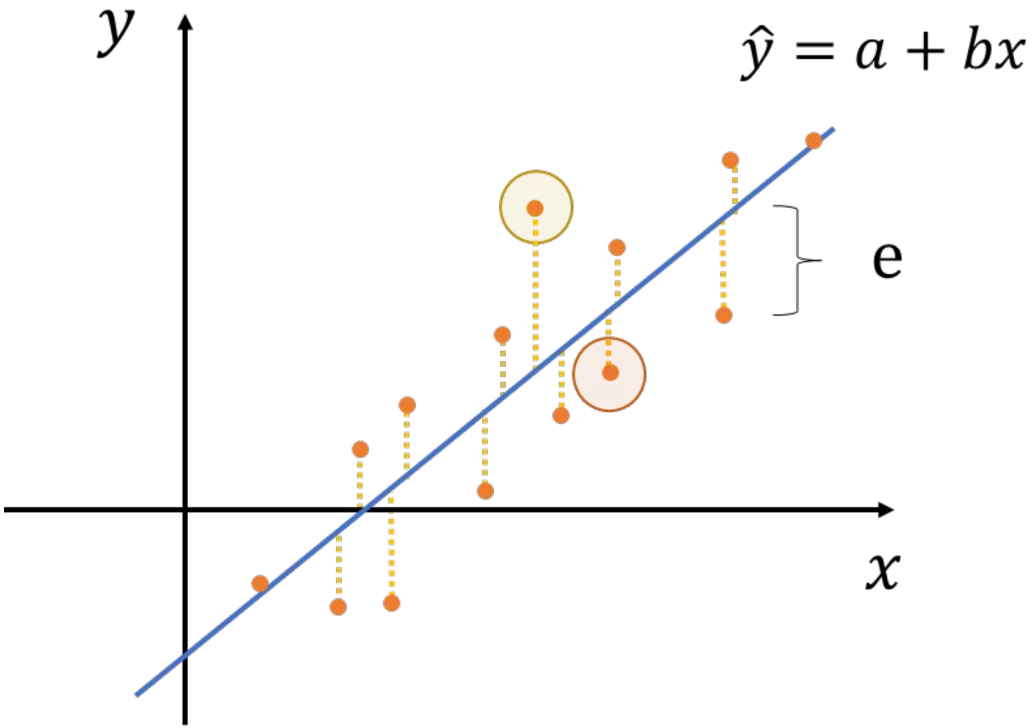


Erro do modelo



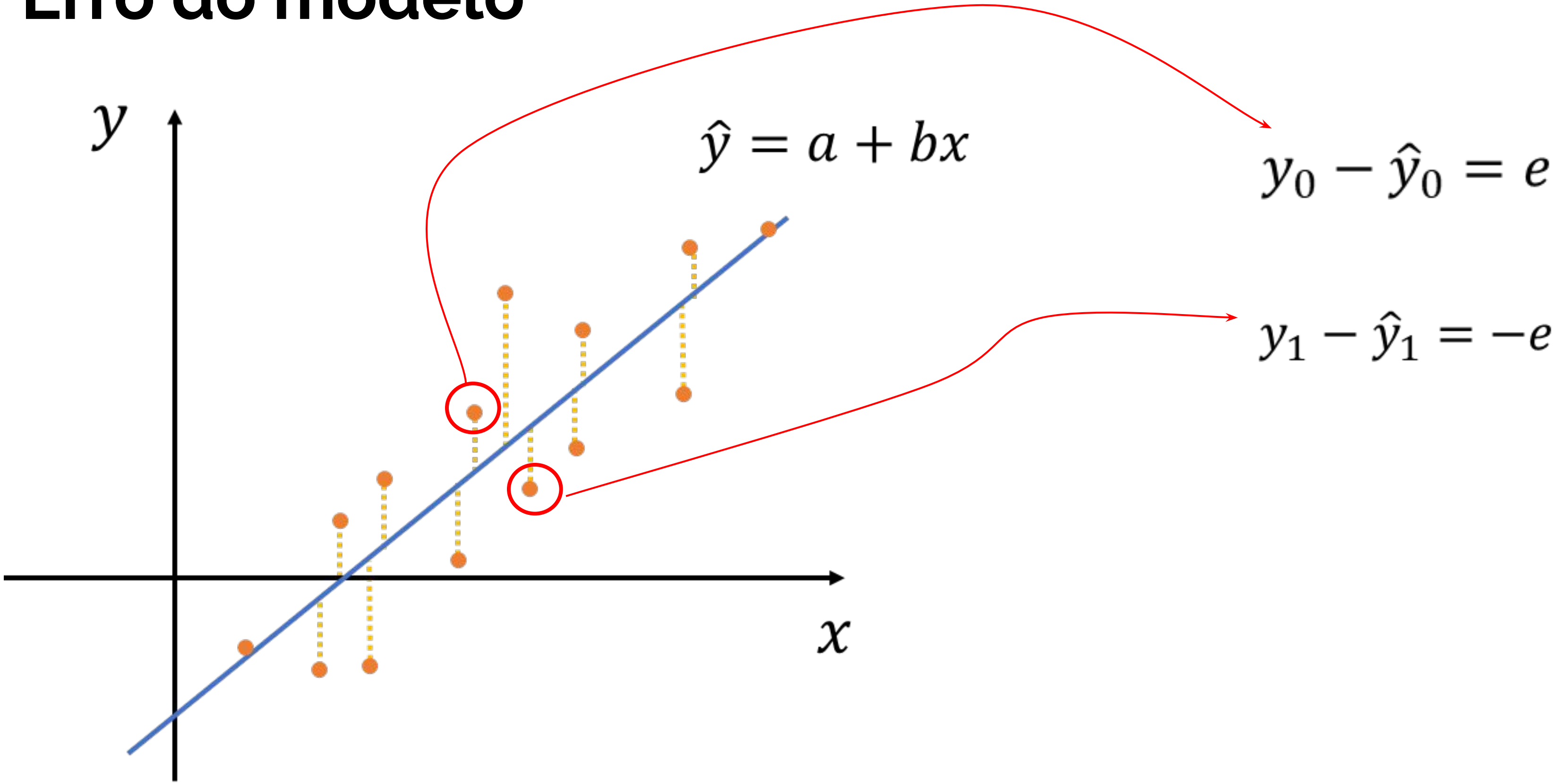
$$e = y - \hat{y}$$
$$e = y - (a + bx)$$

Erro do modelo

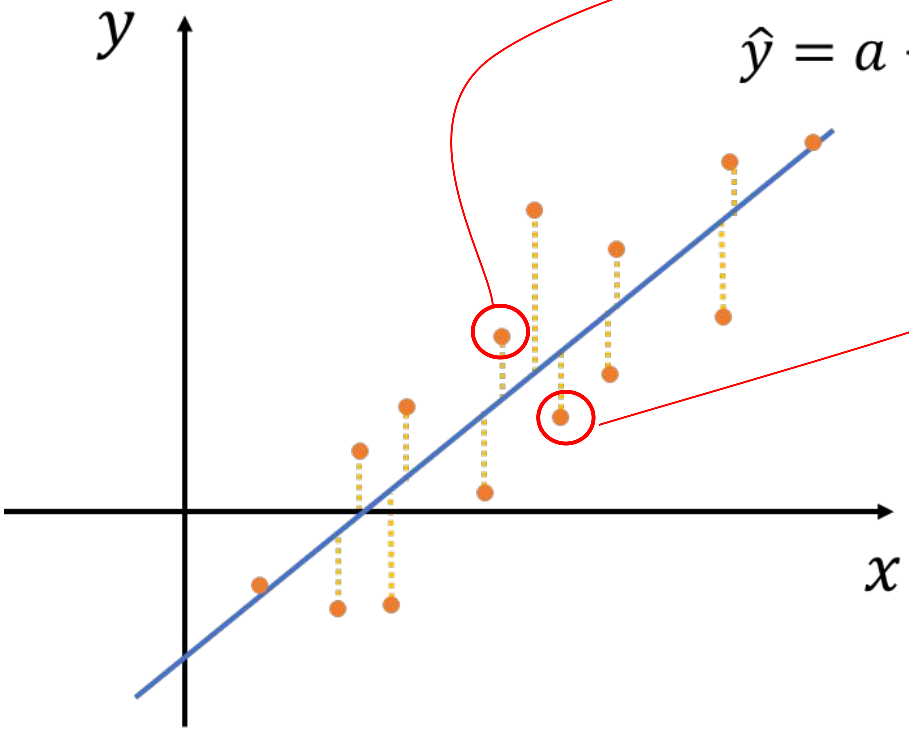


$$
\begin{aligned}
e &= y - \hat{y} \\
e &= y - (a + bx)
\end{aligned}
$$

Erro do modelo



Erro do modelo



$$y_0 - \hat{y}_0 = e$$

$$y_1 - \hat{y}_1 = -e$$

Soma = 0. Está certo? Como corrigir?

Erro do modelo

Erro quadrático médio
R (MSE)

$$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Erro absoluto médio (MAE)

$$\frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Erro do modelo

Equação que queremos encontrar os coeficientes:

Vendas = c0 * TV + c1 * Radio + c2 * Jornal + b

Logo, podemos traduzi-la para:

$$\hat{y} = c_0 * x_0 + c_1 * x_1 + c_2 * x_2 + b$$

Substituindo na equação de erro, temos:

$$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - (c_0 * x_0 + c_1 * x_1 + c_2 * x_2 + b))^2}$$

Como encontrar os valores dos coeficientes?

Aprendizado do modelo

	Investido	Venda
Mês 1	R$ 200	R$ 600
Mês 2	R$ 100	R$ 400
Mês 3	R$ 150	R$ 580

$$\hat{y} = b + c_0 x_0$$

1ª tentativa: chutamos um valor para os coeficientes.

- b = 3
- c0 = 2

$$\hat{y}_1 = 3 + 2 * 200 = 403$$

$$\hat{y}_2 = 3 + 2 * 100 = 203$$

$$\hat{y}_3 = 3 + 2 * 150 = 303$$

Aprendizado do modelo

$$\hat{y} = b + c_0 x_0$$

	Investido	Venda
Mês 1	R$ 200	R$ 600
Mês 2	R$ 100	R$ 400
Mês 3	R$ 150	R$ 580

1ª tentativa: chutamos um valor para os coeficientes.

- b = 3
- c0 = 2

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$$\text{MSE} = \frac{1}{3}((600 - 403)^2 + (400 - 203)^2 + (580 - 303)^2)$$

MSE = 51449

Aprendizado do modelo

$$\hat{y} = b + c_0 x_0$$

	Investido	Venda
Mês 1	R$ 200	R$ 600
Mês 2	R$ 100	R$ 400
Mês 3	R$ 150	R$ 580

- b = 3
- c0 = 2

MSE = 51449

- b = 10
- c0 = 3

MSE = 7533,33

- b = 1
- c0 = 1.2

MSE = 121974,33

Aprendizado do modelo

	Investido	Venda
Mês 1	R$ 200	R$ 600
Mês 2	R$ 100	R$ 400
Mês 3	R$ 150	R$ 580

$$\hat{y} = b + c_0 x_0$$

- b = 3
- c0 = 2

MSE = 51449

- b = 10
- c0 = 3

MSE = 7533,33

- b = 1
- c0 = 1.2

MSE = 121974,33

Aprendizado do modelo

	Investido	Venda
Mês 1	R$ 200	R$ 600
Mês 2	R$ 100	R$ 400
Mês 3	R$ 150	R$ 580

$$\hat{y} = b + c_0 x_0$$

- b = 10
- c0 = 3

MSE = 7533,33

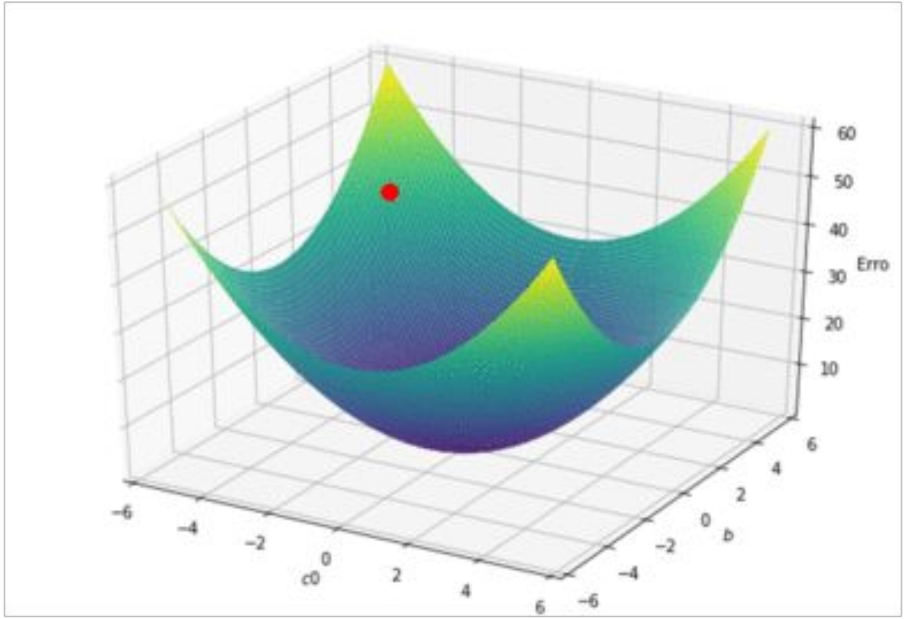
- b = 12
- c0 = 3.5

MSE = 5279

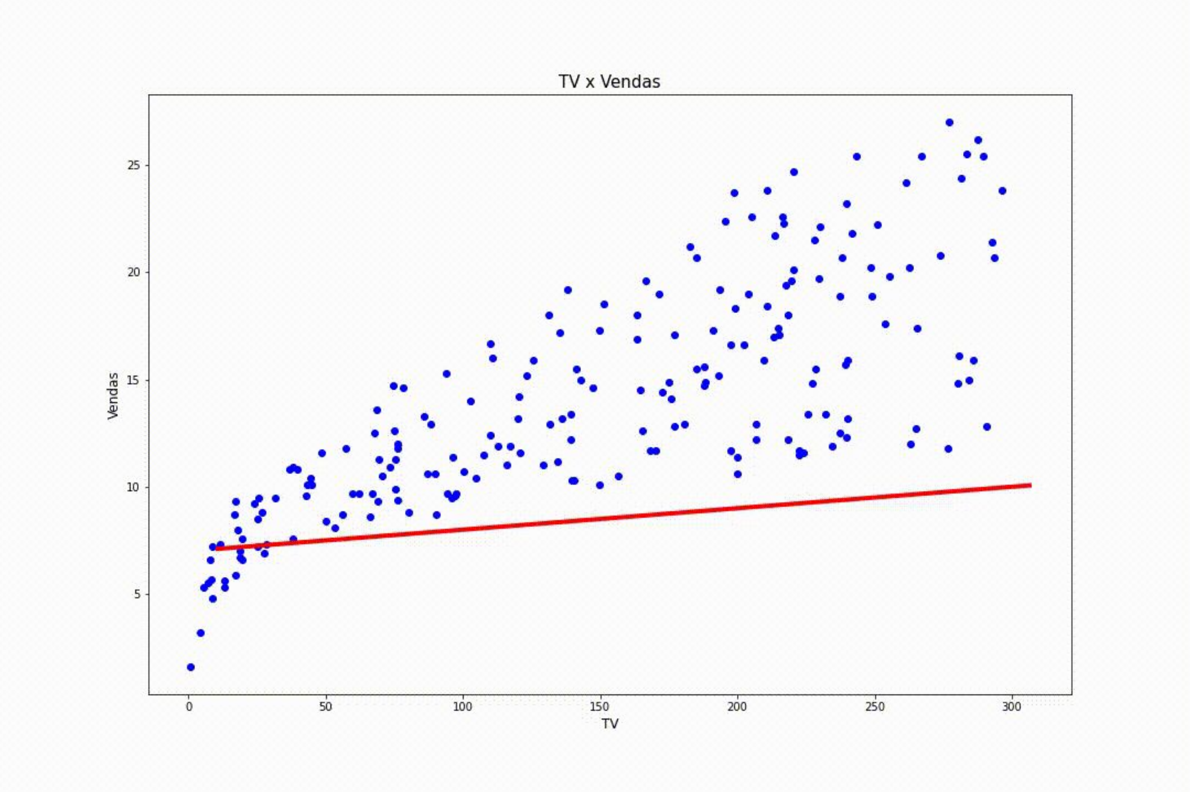
- b = 7
- c0 = 2.5

MSE = 22767,33

Aprendizado do modelo



Aprendizado do modelo

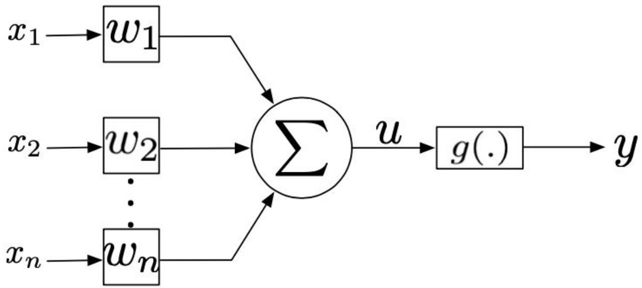


Redes Neurais Artificias

Neurônio

Responsáveis por tarefas simples:

- receber sinais em suas entradas x_i;
- agregá-las por meio de um combinador linear $\sum$;
- aplicar uma função de ativação $g(.)$

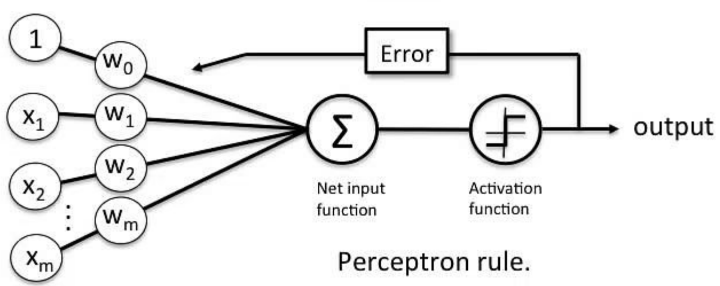


Sua saída pode ser representada por $y = g(u) = g(\sum_{i=1}^{n} x_i w_i)$

Perceptron

Proposto por Frank Rosenblatt em 1957

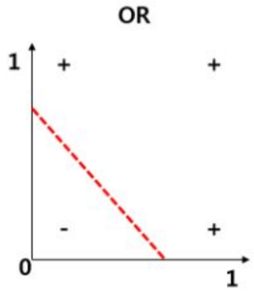
- Única camada é capaz de aprender padrões linearmente separáveis



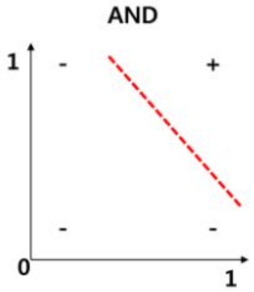
- o algoritmo perceptron aprenderia os coeficientes de peso ótimos
- atualização dos pesos é dada por $\Delta w_j = \lambda * w_j * (y - \hat{y})$

taxa de aprendizado, entre 0 e 1

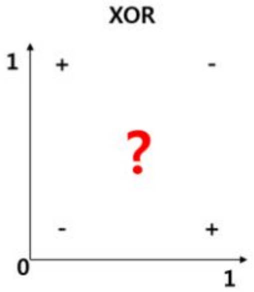
Capacidade de Classificação



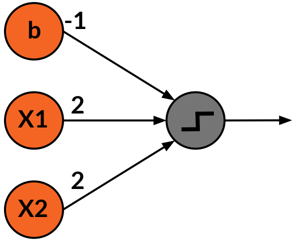
x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

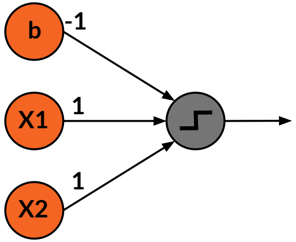


x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

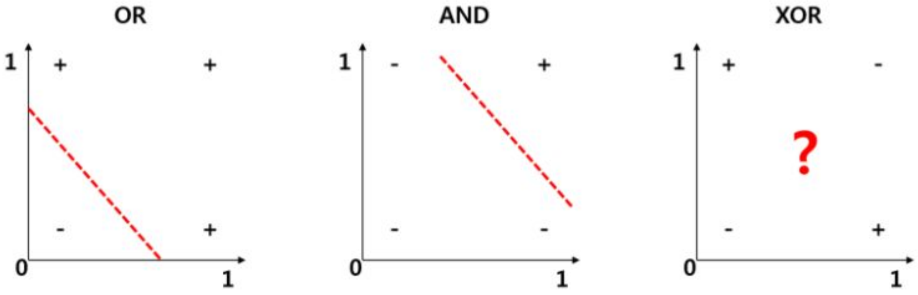


x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0





Capacidade de Classificação



OR

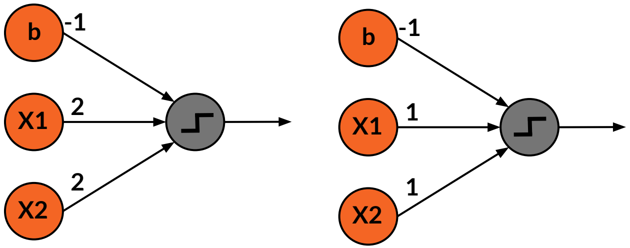
x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

AND

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

XOR

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0



OR

$x_1 = 0$

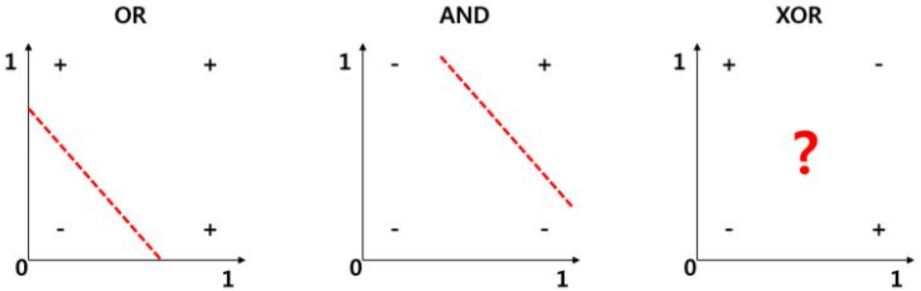
$x_2 = 1$

$$x_1 * 2 + x_2 * 2 - 1$$

$$0 * 2 + 1 * 2 - 1 = 1$$

$$1 > 0 = 1$$

Capacidade de Classificação



OR

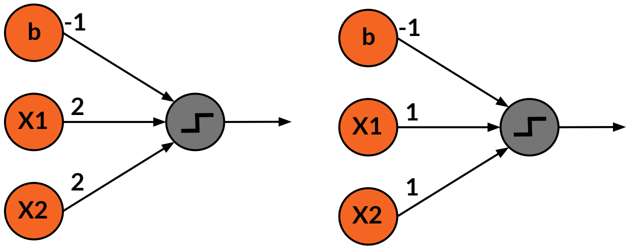
x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

AND

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

XOR

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0



AND

$x_1 = 0$

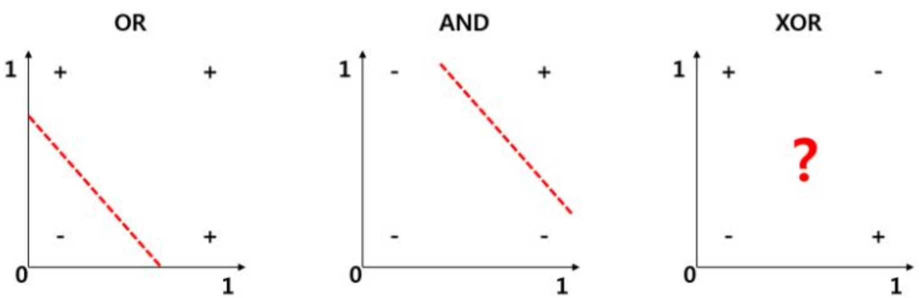
$x_2 = 1$

$$x_1 * 1 + x_2 * 1 - 1$$

$$0 * 1 + 1 * 1 - 1 = 0$$

$$1 > 0 = 0$$

Capacidade de Classificação



OR

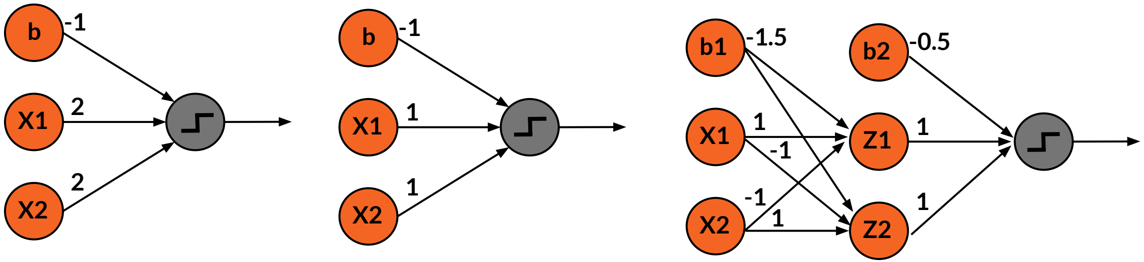
x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

AND

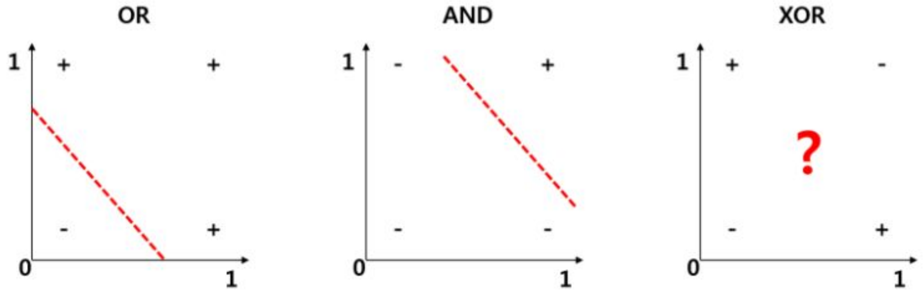
x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

XOR

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0



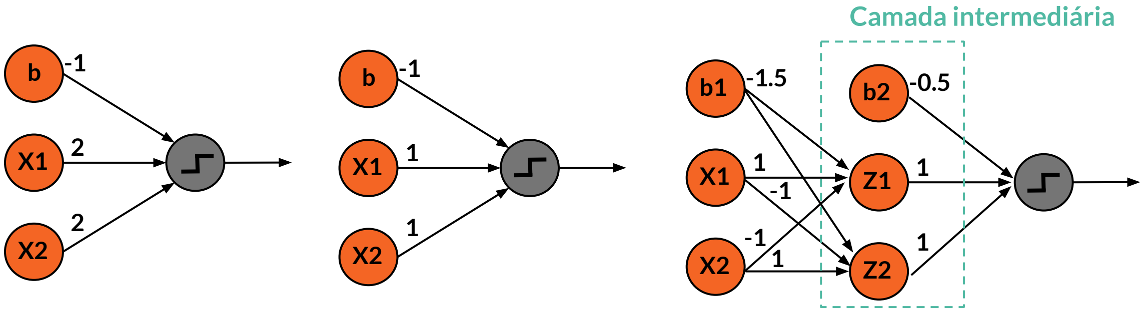
Capacidade de Classificação



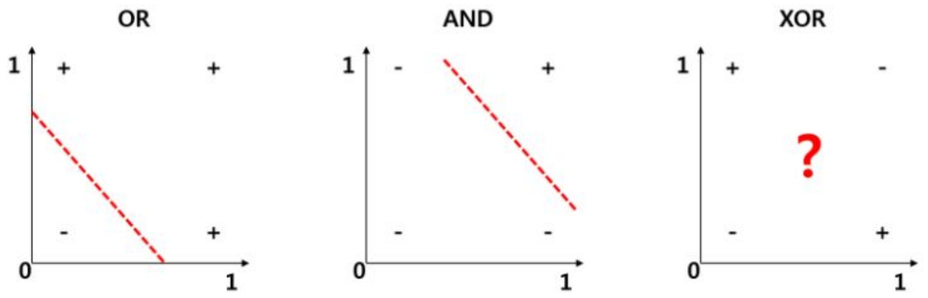
x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0



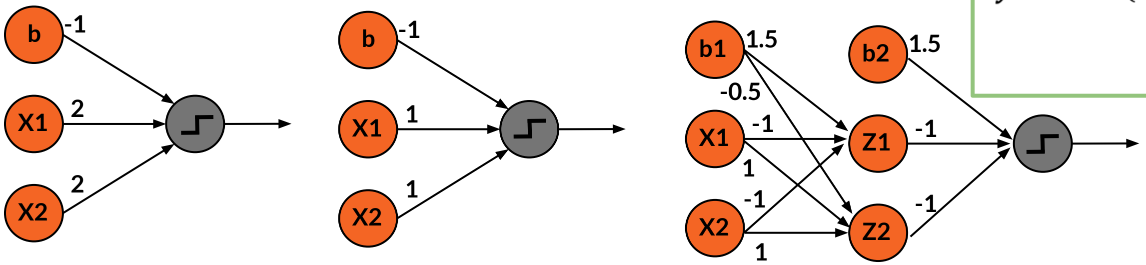
Capacidade de Classificação



x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0



XOR

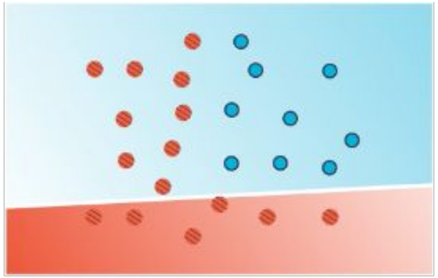
$$x_1 = 0$$
$$x_2 = 1$$

$$z_1 = x_1 * (-1) + x_2 * (-1) + 1.5$$
$$z_1 = 0 * (-1) + 1 * (-1) + 1.5$$
$$z_1 = 0.5$$

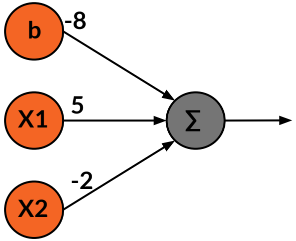
$$z_2 = x_1 * 1 + x_2 * 1 - 0.5$$
$$z_2 = 0 * 1 + 1 * 1 - 0.5$$
$$z_2 = 0.5$$

$$y = z_1 * (-1) + z_2 * (-1) + 1.5$$
$$y = 0.5 * (-1) + 0.5 * (-1) + 1.5$$
$$y = 0.5$$
$$0.5 > 0 = 1$$

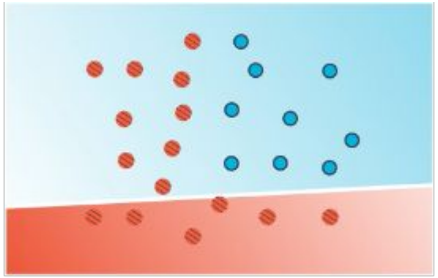
Combinação de Superfícies



$$5x_1 - 2x_2 - 8$$

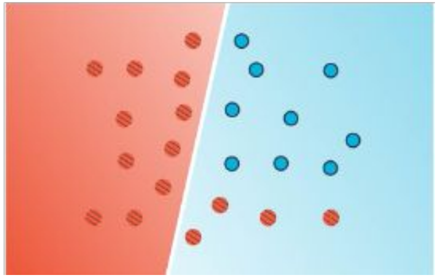


Combinação de Superfícies

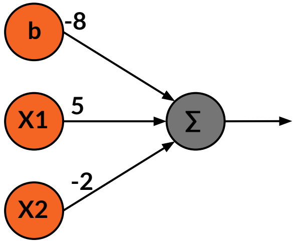


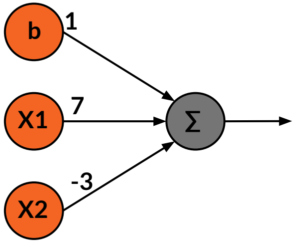
$5x_1 \; - 2x_2 \; - 8$

+

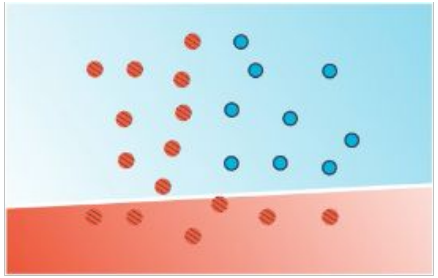


$7x_1 \; - 3x_2 + 1$

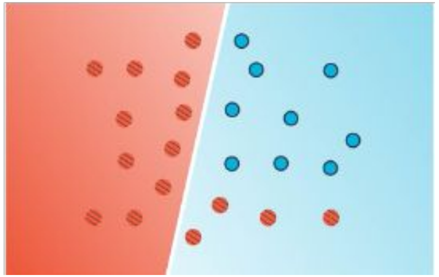




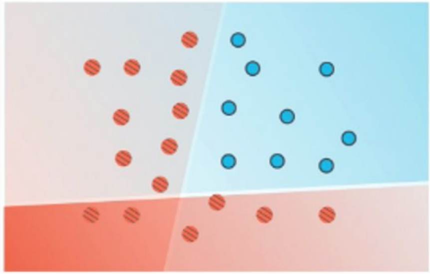
Combinação de Superfícies



+

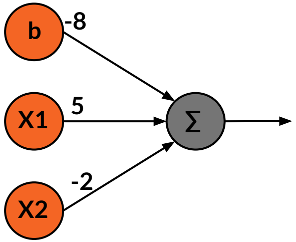


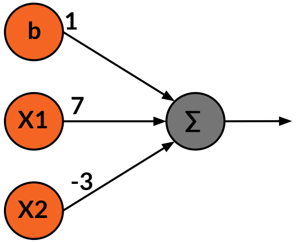
=

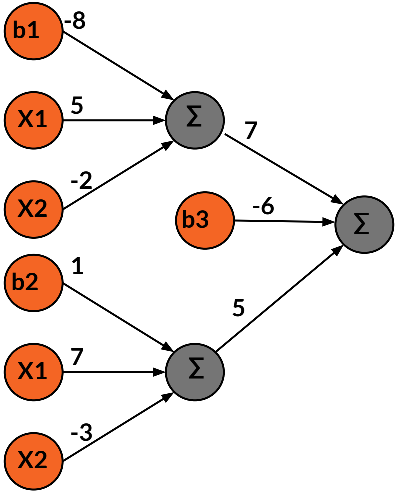


$5x_1 \ - 2x_2 \ - 8$

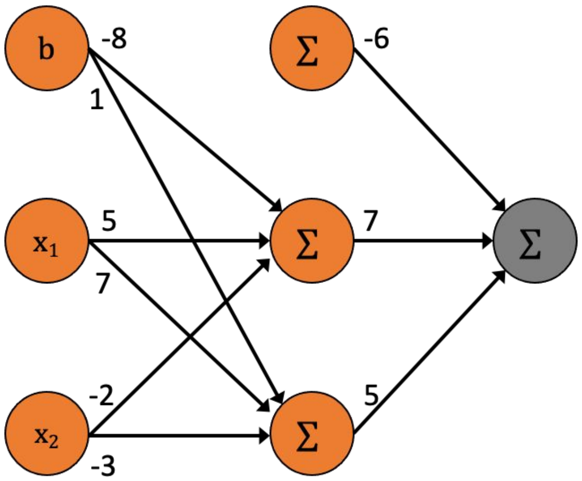
$7x_1 \ - 3x_2 + 1$

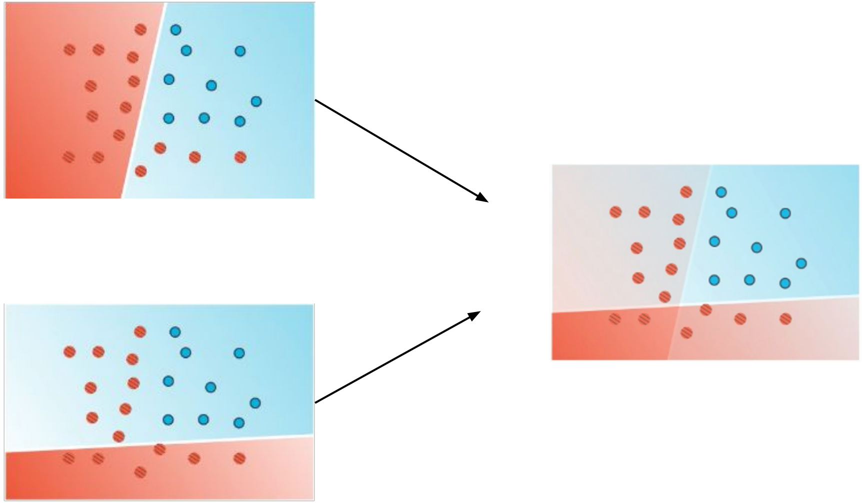






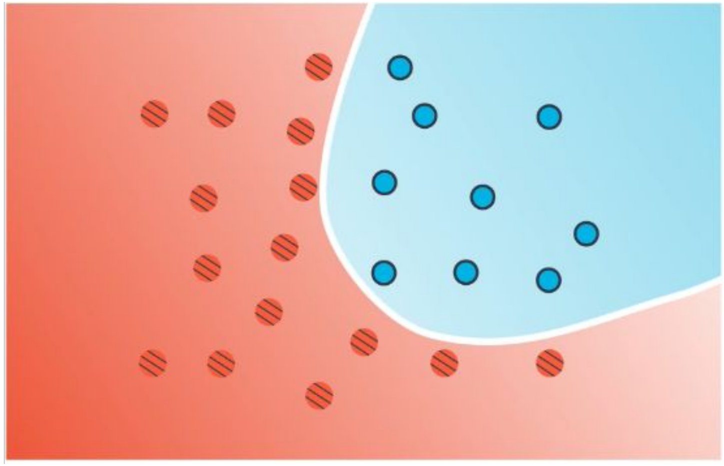
Rede Neural



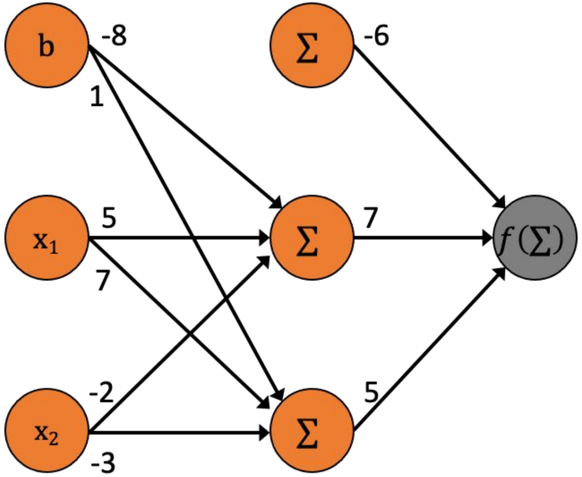


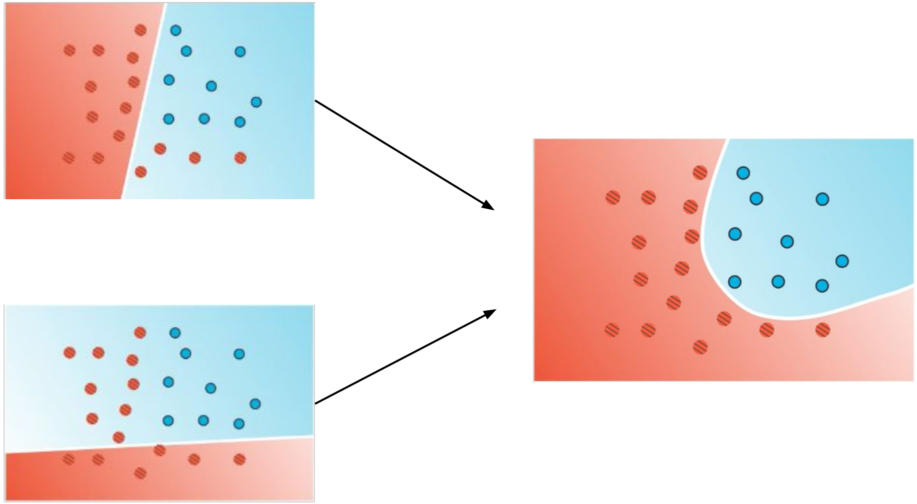
Rede Neural

E os casos não lineares?

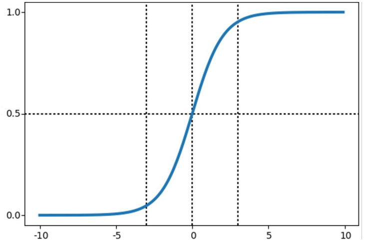


Rede Neural

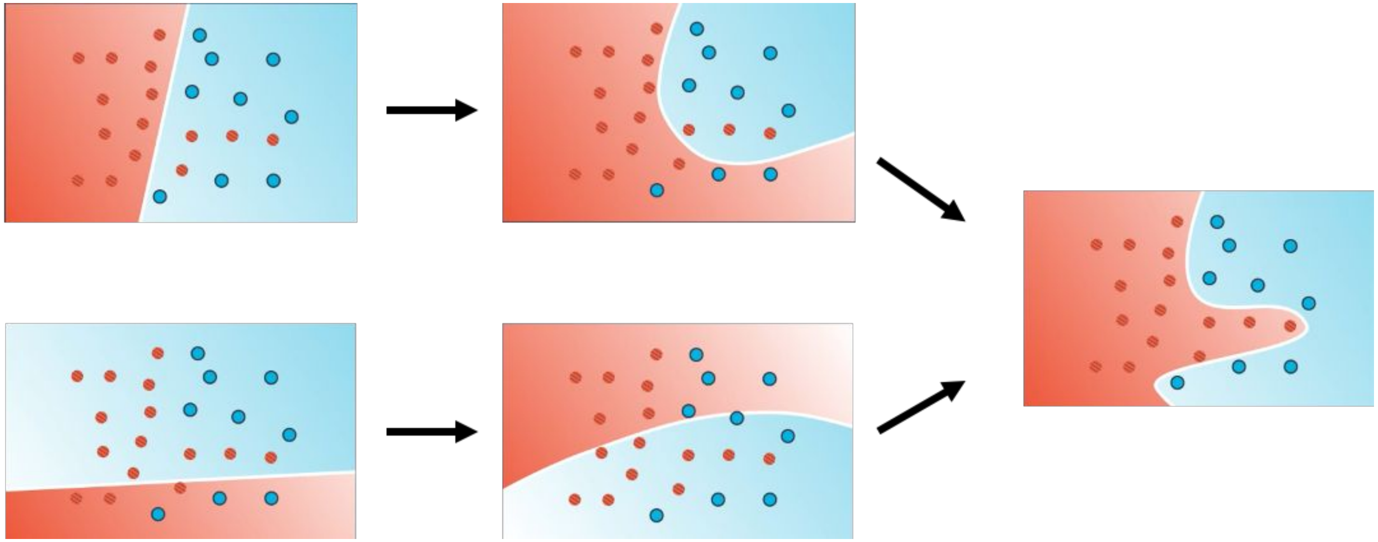




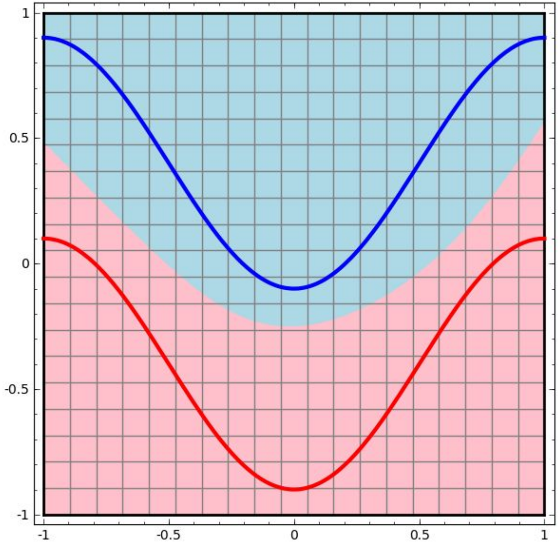
$$f(x) = \frac{1}{1 + e^{-x}} =$$

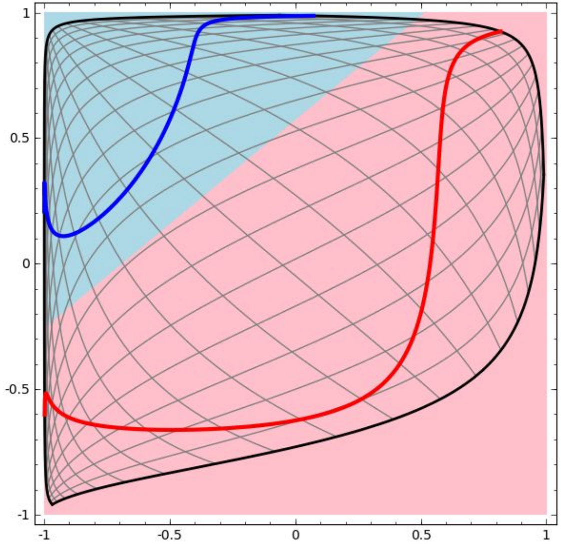


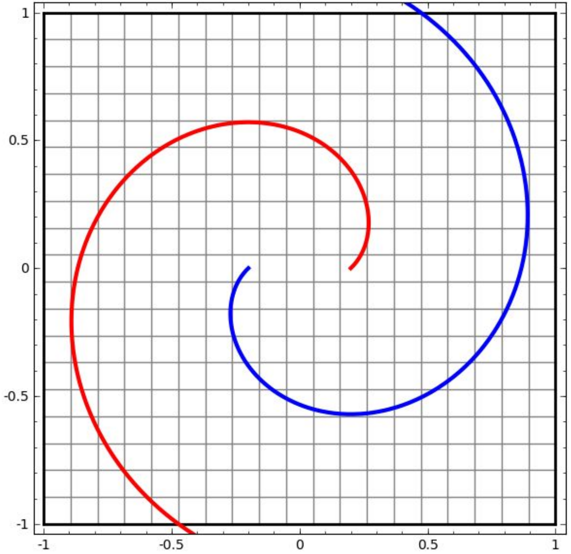
Rede Neural



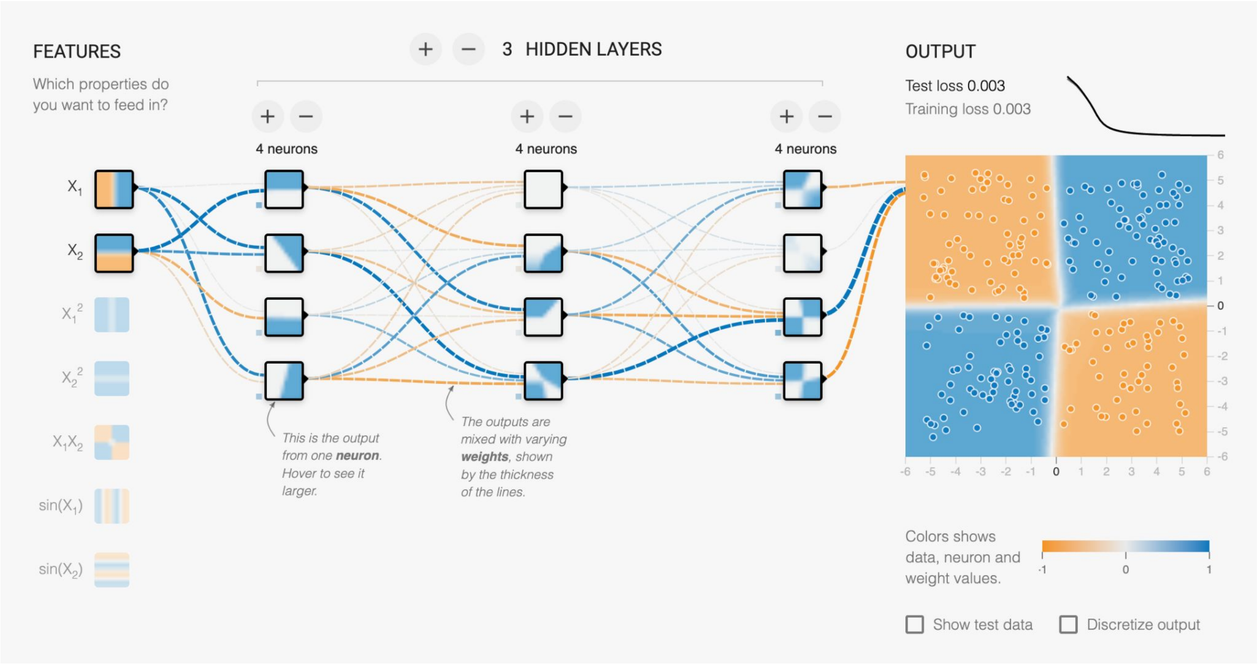
Rede Neural





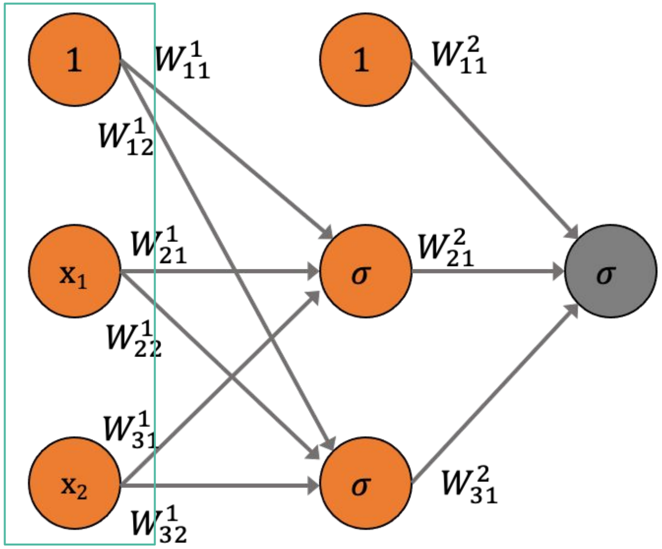


TF Playground



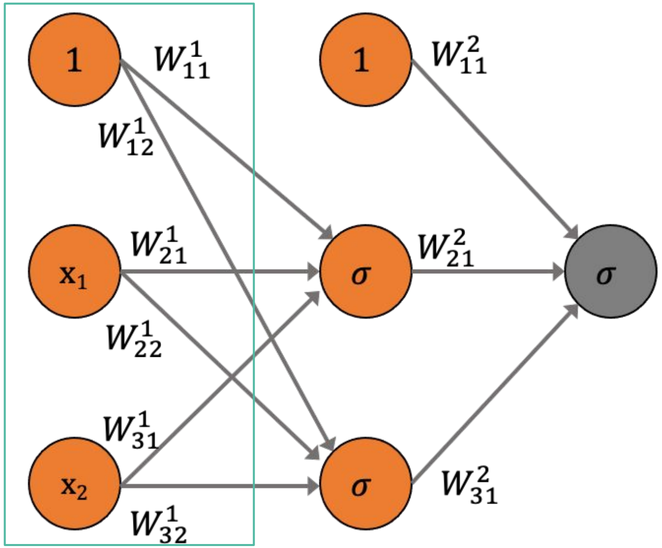
https://playground.tensorflow.org/

Rede Neural - feedforward



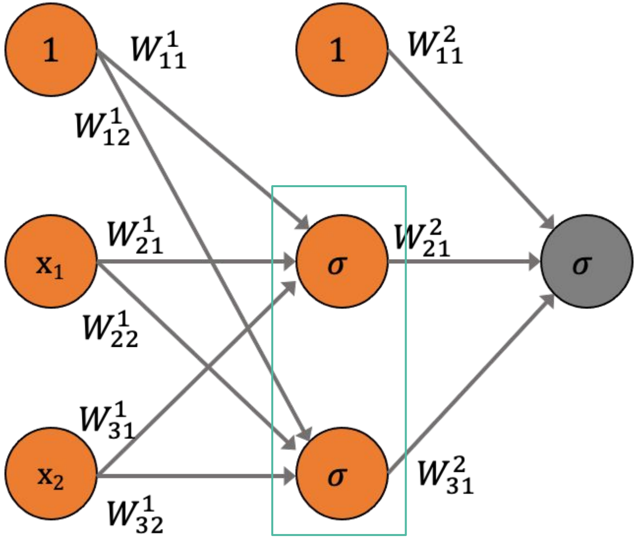
$$\begin{pmatrix} 1 \\ x_1 \\ x_2 \end{pmatrix}$$

Rede Neural - feedforward



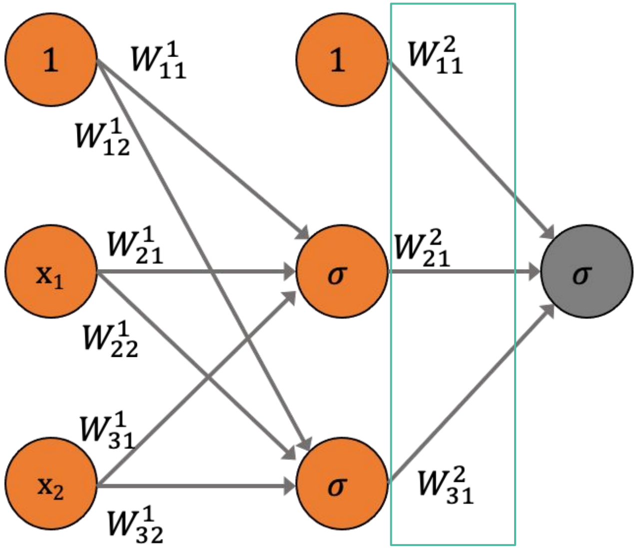
$$\begin{pmatrix} W_{11}^1 & W_{12}^1 \\ W_{21}^1 & W_{22}^1 \\ W_{31}^1 & W_{32}^1 \end{pmatrix} \begin{pmatrix} 1 \\ x_1 \\ x_2 \end{pmatrix}$$

Rede Neural - feedforward



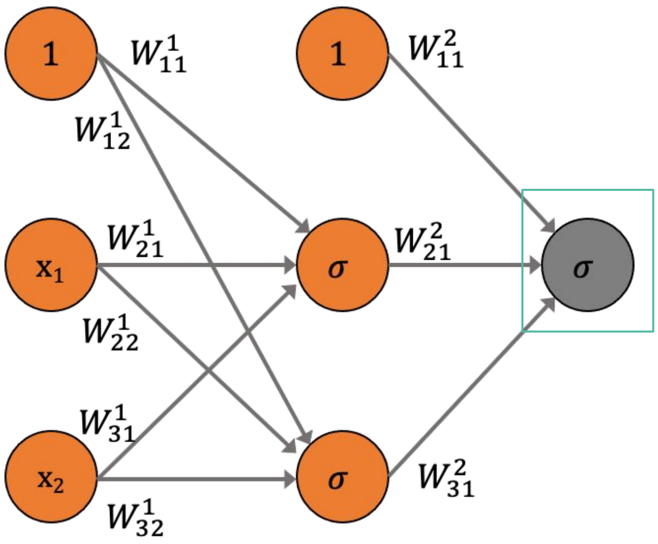
$$\sigma \begin{pmatrix} W_{11}^1 & W_{12}^1 \\ W_{21}^1 & W_{22}^1 \\ W_{31}^1 & W_{32}^1 \end{pmatrix} \begin{pmatrix} 1 \\ x_1 \\ x_2 \end{pmatrix}$$

Rede Neural - feedforward



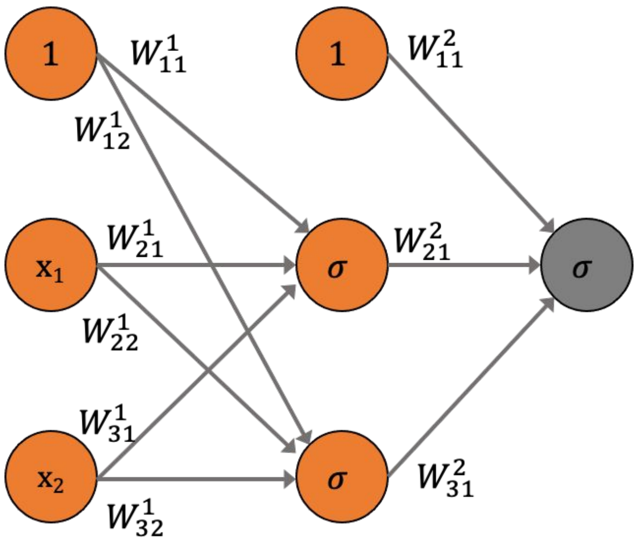
$$\begin{pmatrix} W_{11}^2 \\ W_{21}^2 \\ W_{31}^2 \end{pmatrix} \sigma \begin{pmatrix} W_{11}^1 & W_{12}^1 \\ W_{21}^1 & W_{22}^1 \\ W_{31}^1 & W_{32}^1 \end{pmatrix} \begin{pmatrix} 1 \\ x_1 \\ x_2 \end{pmatrix}$$

Rede Neural - feedforward

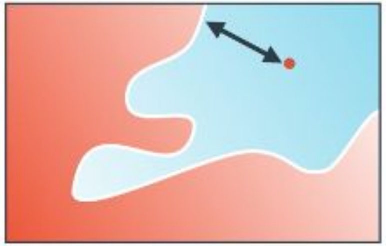


$$\hat{y} = \sigma \begin{pmatrix} W_{11}^2 \\ W_{21}^2 \\ W_{31}^2 \end{pmatrix} \sigma \begin{pmatrix} W_{11}^1 & W_{12}^1 \\ W_{21}^1 & W_{22}^1 \\ W_{31}^1 & W_{32}^1 \end{pmatrix} \begin{pmatrix} 1 \\ x_1 \\ x_2 \end{pmatrix}$$

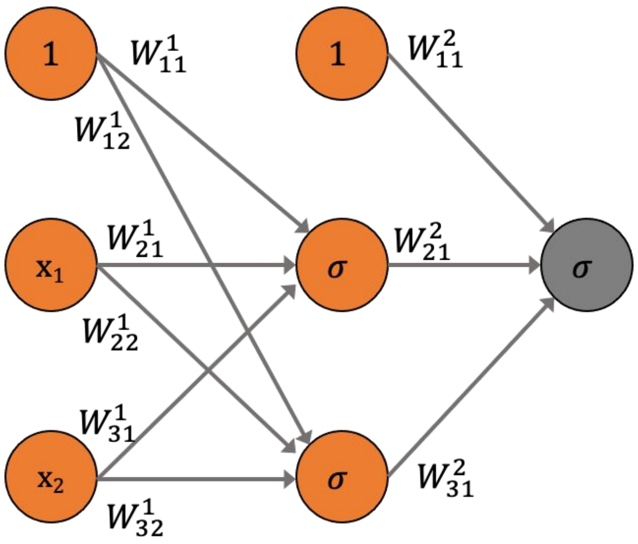
Erro Classificação Binária



Erro = $y \ln(\hat{y}) + (1 - y)\ln(1 - \hat{y})$



Erro Classificação Binária



Erro 1

$y = 0$
$\hat{y} \neq 0$

Erro = $0 \ln(\hat{y}) + (1-0)\ln(1-\hat{y})$

Erro = $\ln(1-\hat{y})$

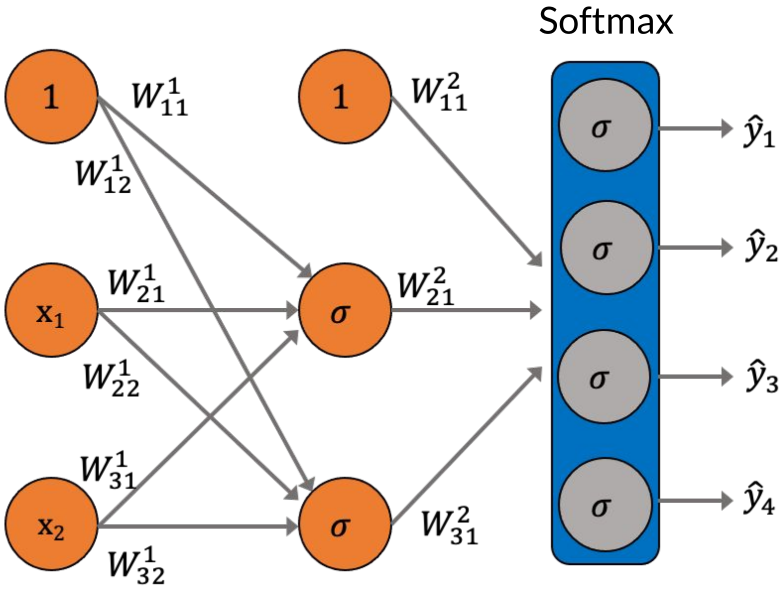
Erro 2

$y = 1$
$\hat{y} \neq 1$

Erro = $1 \ln(\hat{y}) + (1-1)\ln(1-\hat{y})$

Erro = $\ln(\hat{y})$

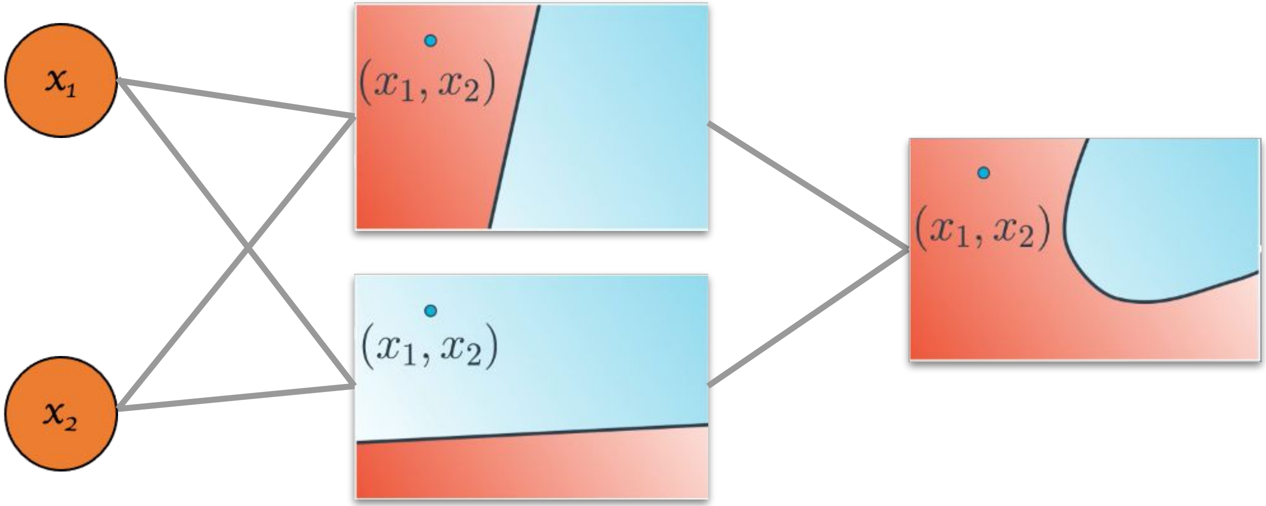
Erro Classificação Multiclasse



$$Softmax(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{M} e^{z_j}}$$

$$Erro = \sum_{j=1}^{M} y_j \log(\hat{y}_j)$$

Ajuste de erro - Backpropagation

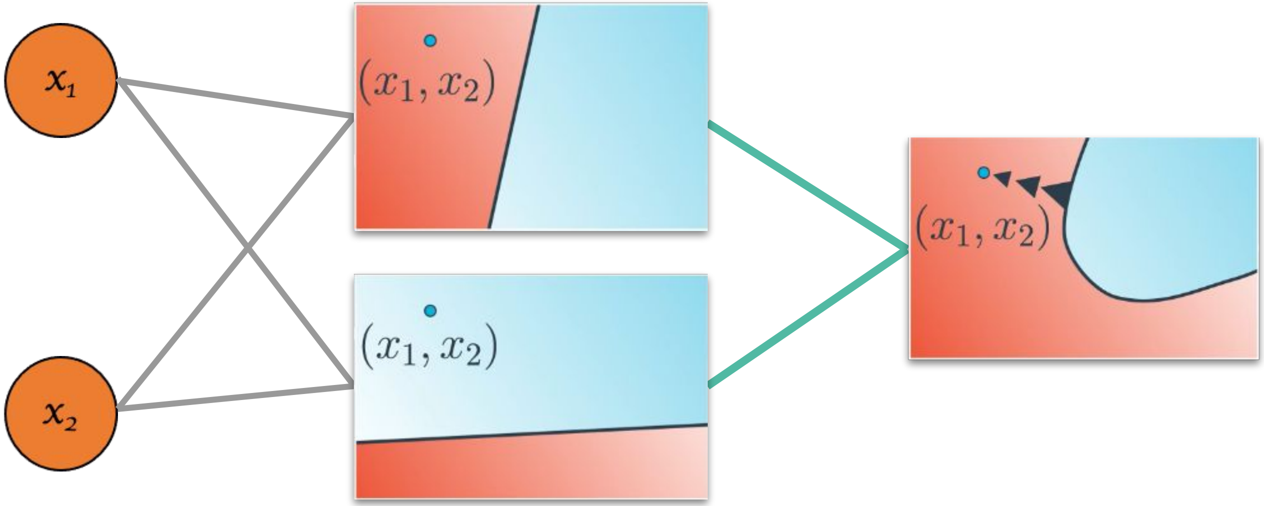


$x = (x_1, x_2)$
$y = 1$

Ajuste de erro - Backpropagation

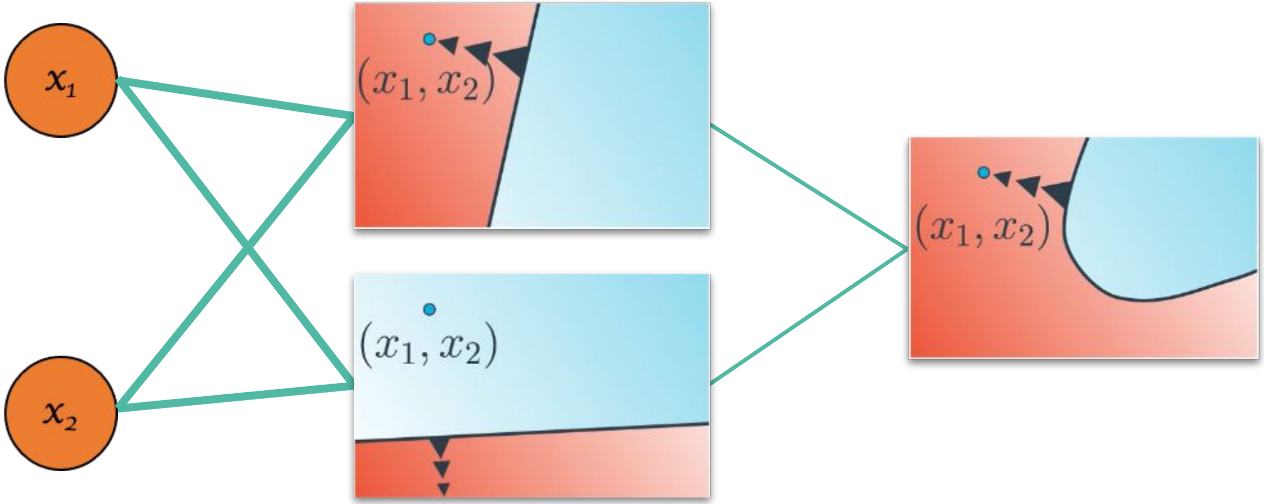
$$x = (x_1, x_2)$$
$$y = 1$$

Ajuste de erro - Backpropagation



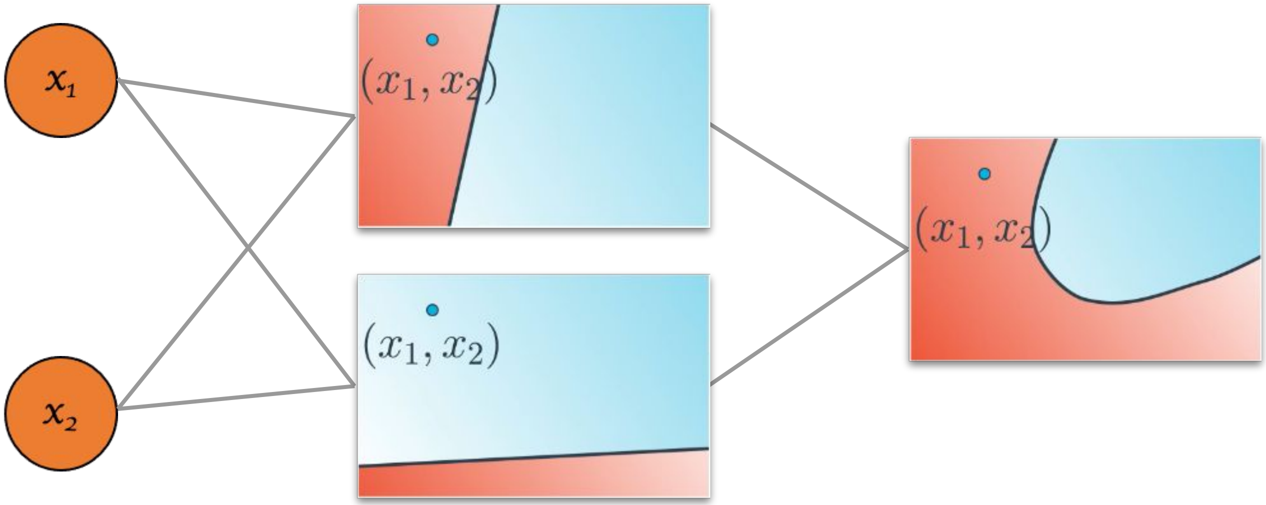
$x = (x_1, x_2)$
$y = 1$

Ajuste de erro - Backpropagation



$x = (x_1, x_2)$
$y = 1$

Ajuste de erro - Backpropagation

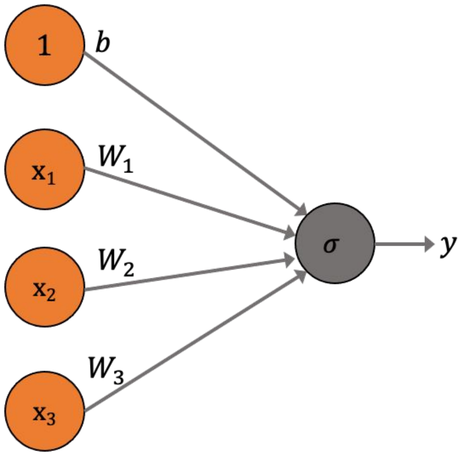


$x = (x_1, x_2)$
$y = 1$

Ajuste de erro - Backpropagation

Como fazer os ajustes?

Utilizando o gradiente do erro para reajustar as superfícies de decisão



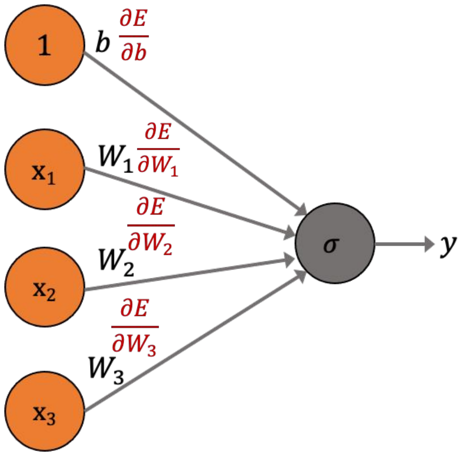
$$W_i' = W_i - \alpha(\nabla E)$$

taxa de aprendizado, entre 0 e 1

Ajuste de erro - Backpropagation

Como fazer os ajustes?

Utilizando o gradiente do erro para reajustar as superfícies de decisão

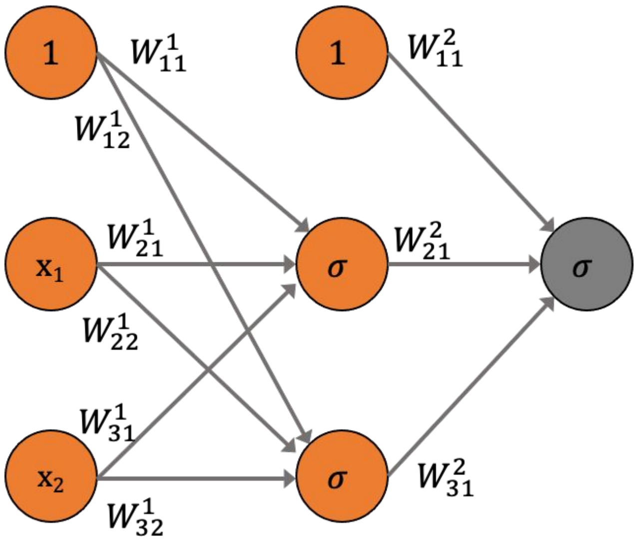


$$W_i' = W_i - \alpha(\nabla E)$$

taxa de aprendizado, entre 0 e 1

$$\nabla E = \left(\frac{\partial E}{\partial b}, \frac{\partial E}{\partial W_1}, \frac{\partial E}{\partial W_2}, \frac{\partial E}{\partial W_3}\right)$$

Ajuste de erro - Backpropagation



$$W^{(1)} = \begin{pmatrix} W^{(1)}_{11} & W^{(1)}_{12} \\ W^{(1)}_{21} & W^{(1)}_{22} \\ W^{(1)}_{31} & W^{(1)}_{32} \end{pmatrix} \quad W^{(2)} = \begin{pmatrix} W^{(2)}_{11} \\ W^{(2)}_{21} \\ W^{(2)}_{31} \end{pmatrix}$$

$$\nabla E = \begin{pmatrix} \frac{\partial E}{\partial W^{(1)}_{11}} & \frac{\partial E}{\partial W^{(1)}_{12}} & \frac{\partial E}{\partial W^{(2)}_{11}} \\ \frac{\partial E}{\partial W^{(1)}_{21}} & \frac{\partial E}{\partial W^{(1)}_{22}} & \frac{\partial E}{\partial W^{(2)}_{21}} \\ \frac{\partial E}{\partial W^{(1)}_{31}} & \frac{\partial E}{\partial W^{(1)}_{32}} & \frac{\partial E}{\partial W^{(2)}_{31}} \end{pmatrix}$$

$$W'^{(k)}_{ij} \leftarrow W^{(k)}_{ij} - \alpha \frac{\partial E}{\partial W^{(k)}_{ij}}$$