INF721

2023/2

UFV

Aprendizado em Redes Neurais Profundas

A7: Backpropagation

Logística

Avisos

- Teste T2: Multilayer Perceptron na próxima aula!

Última aula

- Problemas linearmente não-separáveis
- Multilayer Perceptron (MLP)
- Forward Pass
- Funções de ativação

Plano de Aula

- Grafo computacional
- Backpropagation
  - Gradiente da Regressão Logística
  - Gradiente da MLP

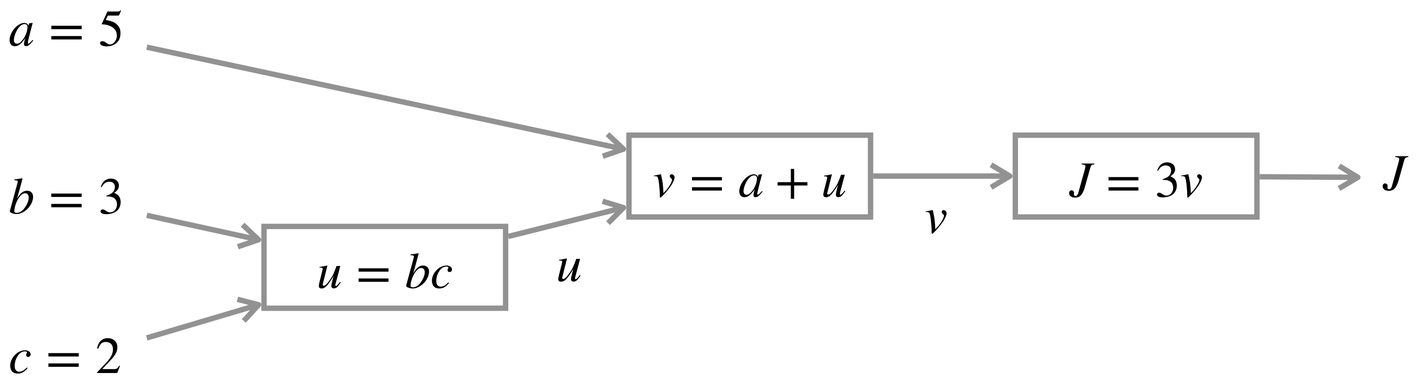
Grafo Computacional

Um grafo dirigido que descreve as expressões matemáticas de uma RNA passo a passo:

- Vértices representam operações
- Arestas representam entrada e saída

$$J(a,b,c) = 3(a + bc)$$

$$u = bc$$
$$v = a + u$$
$$J = 3V$$



Grafo Computacional e RNAs

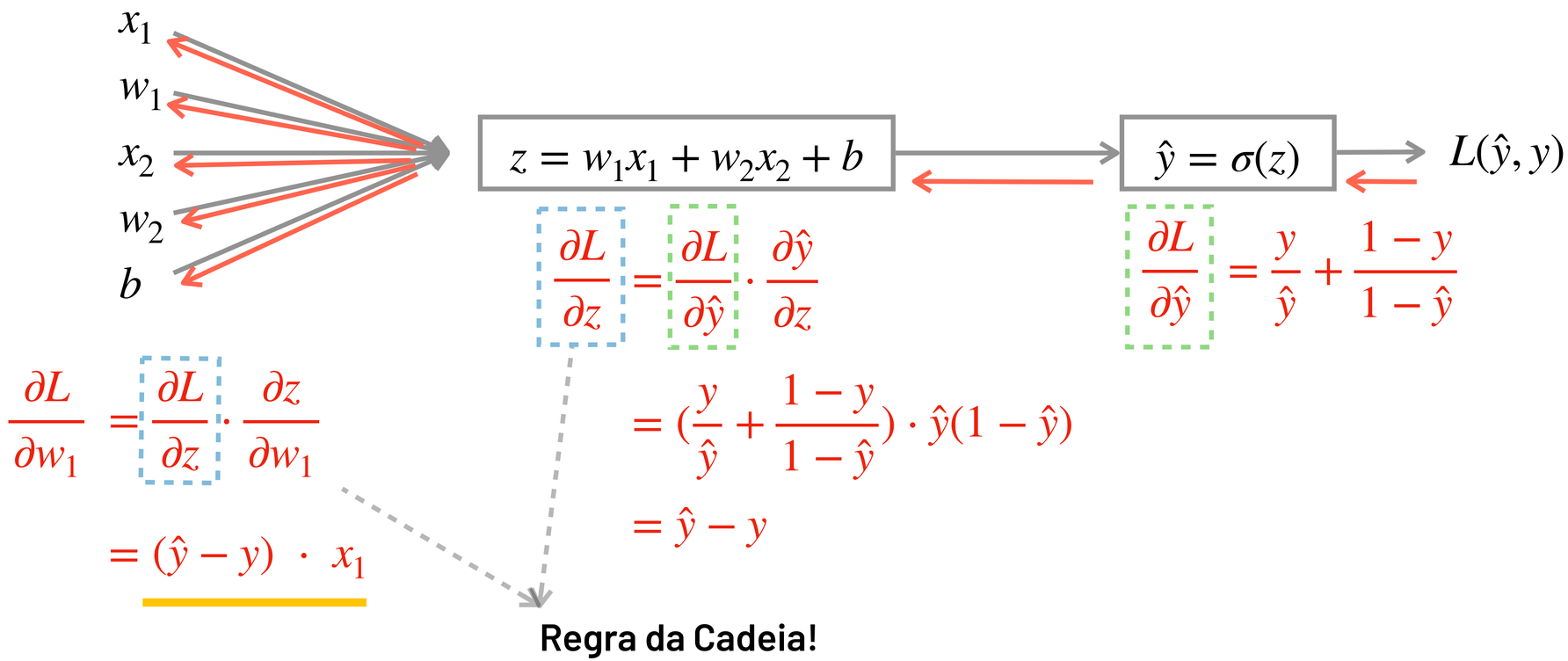
Grafos computacionais nos ajudam a calcular o gradiente de uma função de perda com relação aos pesos de uma RNA

Regressão Logística

$$z = \mathbf{w} \cdot \mathbf{x} + b$$

$$\hat{y} = h(\mathbf{x}) = \frac{1}{1 + e^{-z}}$$

$$L(\hat{y}, y) = -\, y\ log\ \hat{y} + (1 - y)\ log\ (1 - \hat{y}))$$



Retropropagação (Backprop)

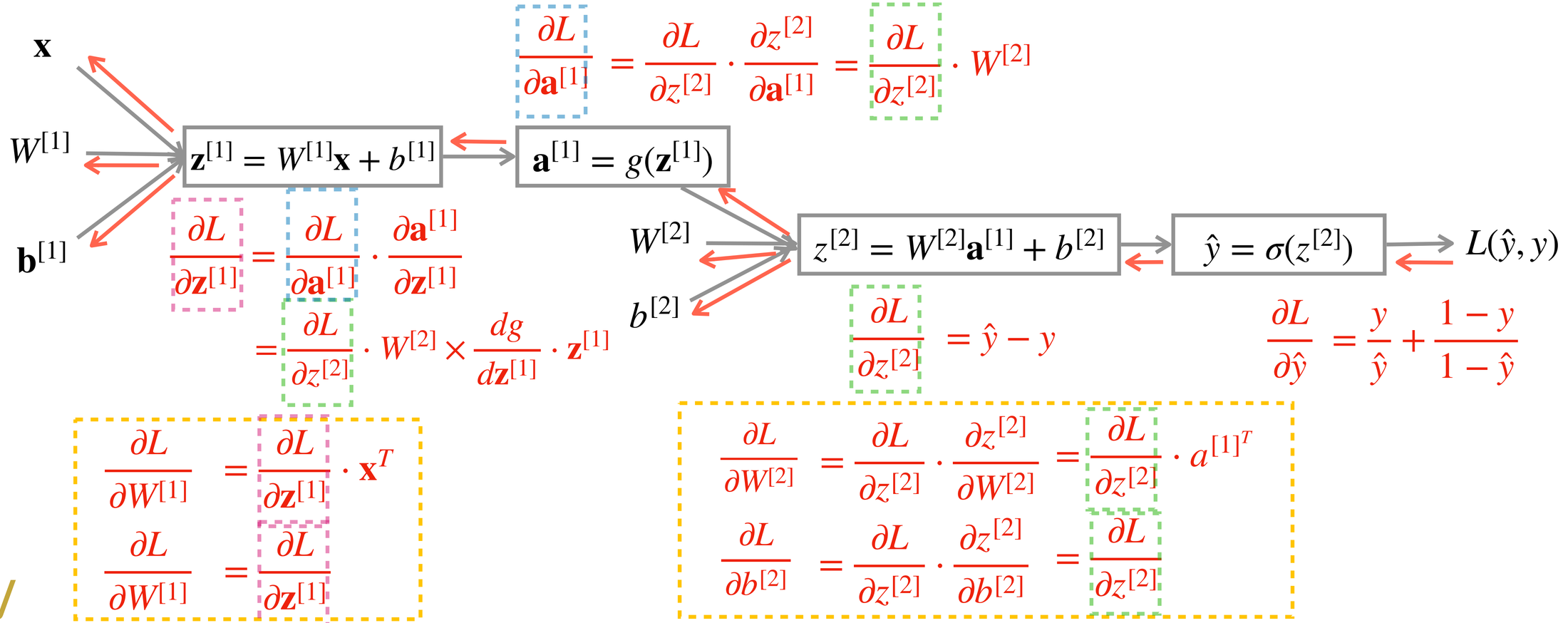
Calcular as derivadas parciais da função de perda com relação aos pesos $W^{[l]}$ e $\mathbf{b}^{[l]}$ para todas as camadas l de trás pra frente com a regra da cadeia.

MLP (1 camada escondidada)

$$\mathbf{z}^{[1]} = W^{[1]}\mathbf{x} + \mathbf{b}^{[1]} \qquad z^{[2]} = W^{[2]}\mathbf{a}^{[1]} + b^{[2]}$$

$$\mathbf{a}^{[1]} = g(\mathbf{z}^{[1]}) \qquad \hat{y} = \sigma(z^{[2]})$$

$$L(\hat{y}, y) = -\, y\ log\ \hat{y} + (1 - y)\ log\ (1 - \hat{y}))$$



Backward Pass

MLP (1 camada)

$$Z^{[1]} = W^{[1]}X + \mathbf{b}^{[1]}$$
$$A^{[1]} = \sigma(Z^{[1]})$$
$$Z^{[2]} = W^{[2]}\mathbf{a}^{[1]} + \mathbf{b}^{[2]}$$
$$\hat{Y} = \sigma(Z^{[2]})$$

Função de Perda

$$L(h) = -\frac{1}{n}\sum_{i=1}^{n}(y_i \; log \; \hat{y}_i + (1 - y_i) \; log \; (1 - \hat{y}_i))$$

Gradiente

$$dZ^{[2]} = \hat{Y} - Y$$
$$dW^{[2]} = \frac{1}{n} dZ^{[2]} \cdot A^{[1]^T}$$
$$db^{[2]} = \frac{1}{n}\sum_{i=1}^{n} dZ^{[2]}_{[:,i]}$$
$$dZ^{[1]} = W^{[2]^T} \cdot dZ^{[2]} \times \frac{dg}{dz^{[1]}} \cdot Z^{[1]}$$
$$dW^{[1]} = \frac{1}{n} dZ^{[1]} \cdot X^T$$
$$db^{[1]} = \frac{1}{n}\sum_{i=1}^{n} dZ^{[1]}_{[:,i]}$$

Inicialização de pesos na MLP

Em RNAs com pelo menos 1 camada escondidada (MLPs), temos que inicializar os pesos com valores aleatórios próximos de zero.

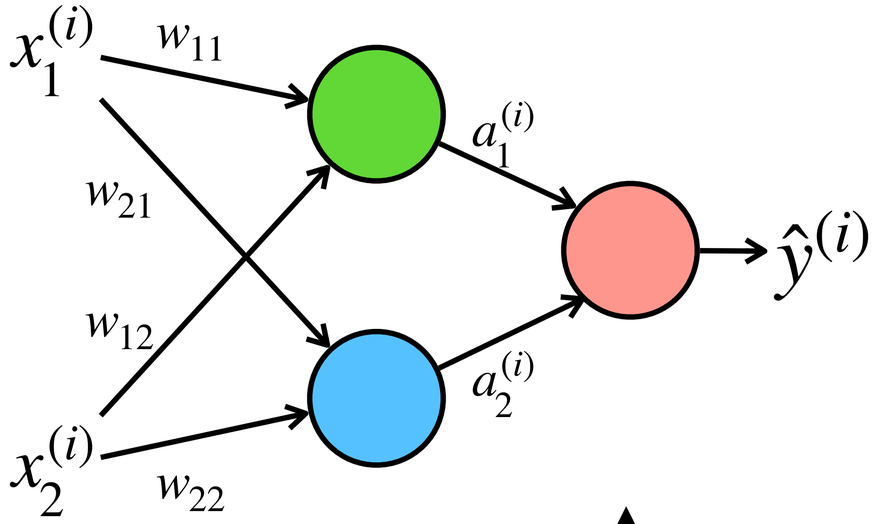
Se inicializarmos com zeros, os neurônios da camada escondida serão iguais!

$$W^{[1]} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \quad b^{[1]} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

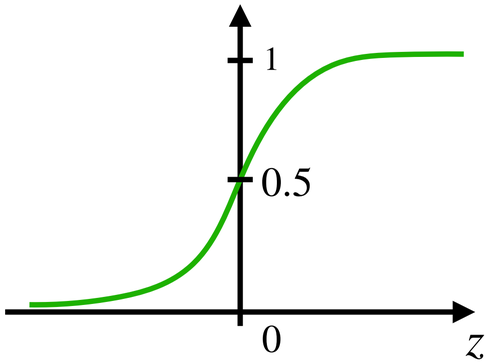
$$W^{[2]} = [0 \quad 0]$$

$$a_1^{(i)} = a_2^{(i)} \longrightarrow dZ_1^{[1]} = dZ_2^{[1]}$$

$$dW = \begin{bmatrix} u & u \\ u & u \end{bmatrix}$$



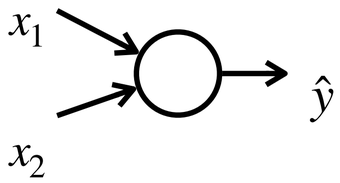
Ns regiões próximas de zero o gradiente é maior!



Redes Neurais Artificiais Profundas

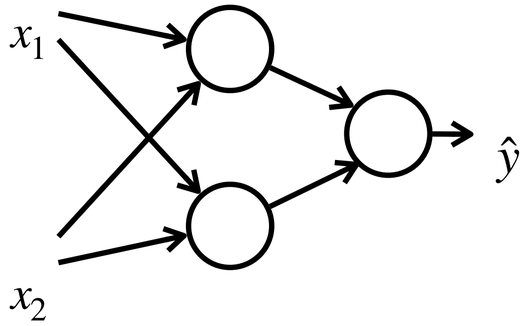
Regressão Logística

RNA de 1 camada (raza)



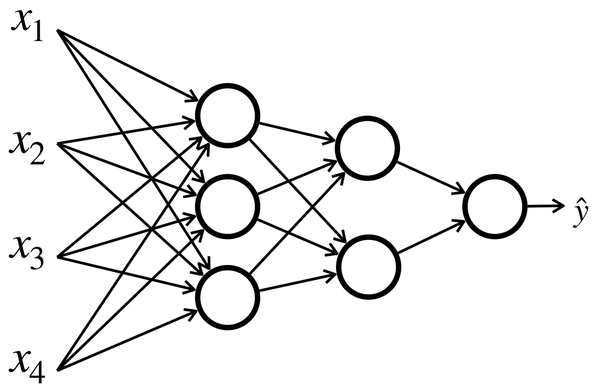
1 camada oculta

RNA de 2 camadas (raza)



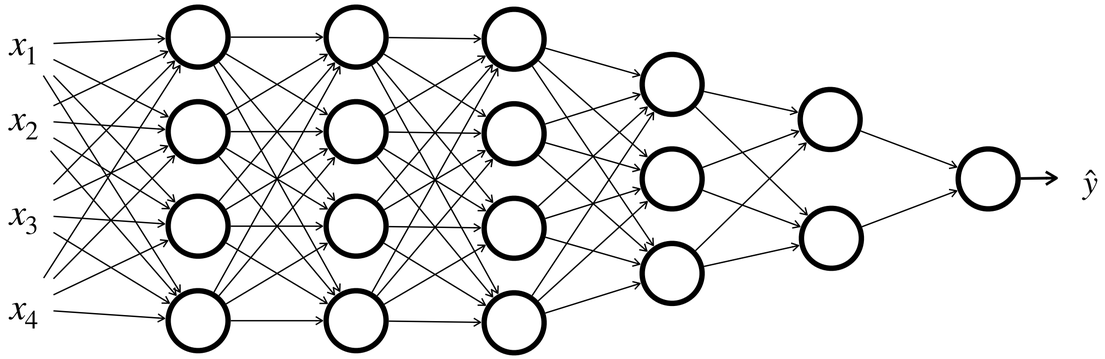
2 camadas ocultas

RNA de 3 camadas (raza)



5 camadas ocultas

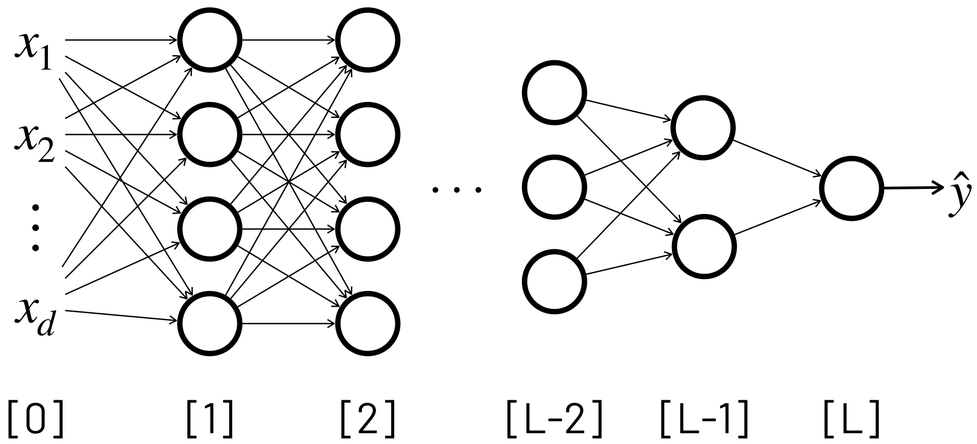
RNA de 6 camadas (profunda)



UFV

Redes Neurais Artificiais Profundas

RNA de L camadas



Para um exemplo $\mathbf{x}$:

$$\mathbf{z}^{[1]} = W^{[1]}\mathbf{x} + \mathbf{b}^{[1]}$$
$$\mathbf{a}^{[1]} = g(\mathbf{z}^{[1]})$$
$$\mathbf{z}^{[2]} = W^{[2]}\mathbf{a}^{[1]} + \mathbf{b}^{[2]}$$
$$\mathbf{a}^{[2]} = g(\mathbf{z}^{[2]})$$
$$\cdots$$
$$\mathbf{z}^{[L]} = W^{[L]}\mathbf{a}^{[L-1]} + b^{[L]}$$
$$\hat{y} = \sigma(\mathbf{z}^{[L]})$$

Regra geral:

$$\mathbf{z}^{[l]} = W^{[l]}\mathbf{a}^{[l-i]} + \mathbf{b}^{[l]}$$
$$\mathbf{a}^{[l]} = g(\mathbf{z}^{[l]})$$

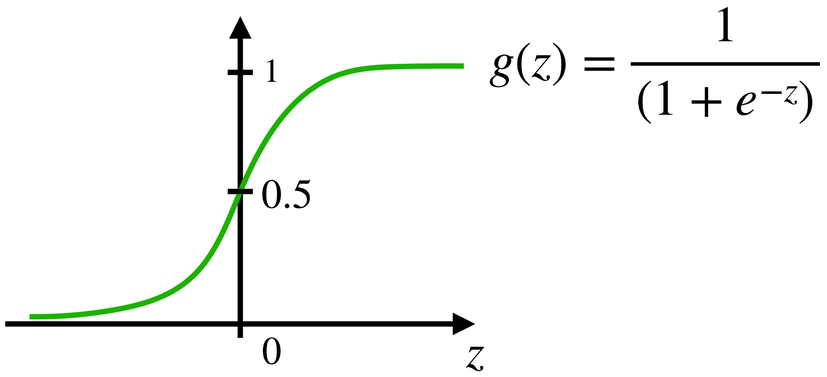
Vetorizado

$$Z^{[l]} = W^{[l]}A^{[l-1]} + \mathbf{b}^{[l]}$$
$$\mathbf{A}^{[l]} = g(Z^{[l]})$$

$$A^{[0]} = X$$
$$A^{[L]} = \hat{Y}$$

Funções de ativação na camada de saída

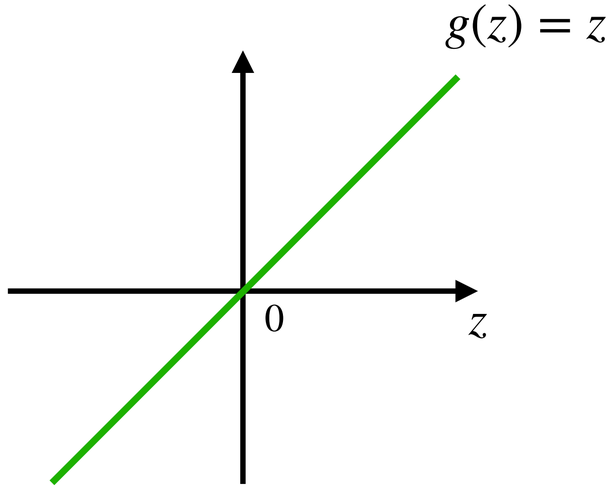
Logística (sigmoide)



$$g(z) = \frac{1}{(1 + e^{-z})}$$

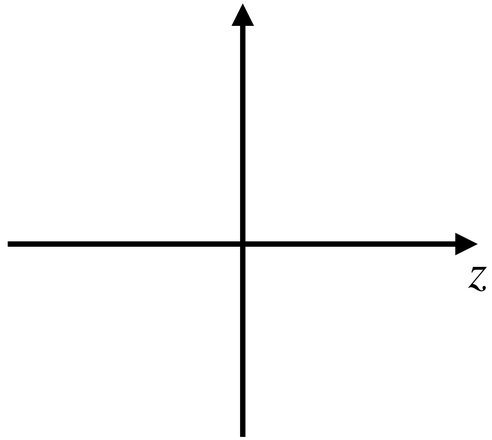
Classificação Binária

Linear



Regressão

?



Classificação Multiclasse

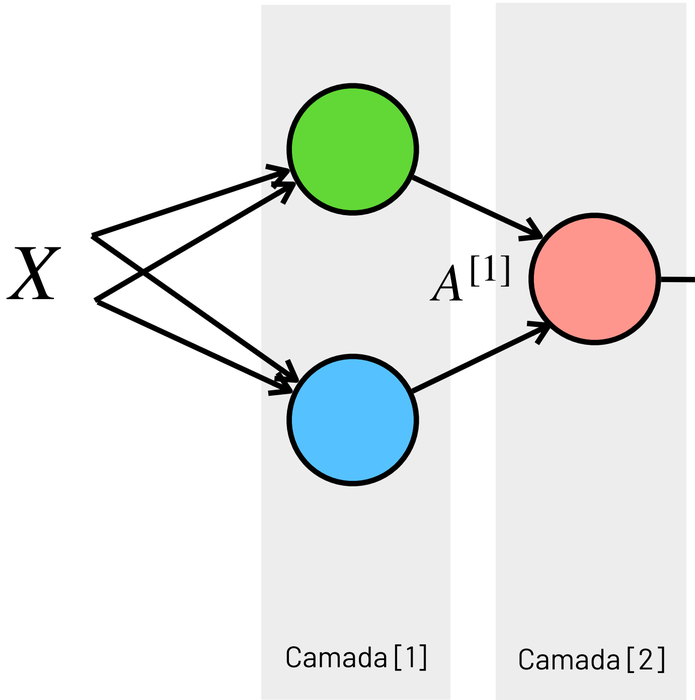
Função de ativação softmax para classificação multiclasse

Hipótese

$$Z^{[1]} = W^{[1]}X + \mathbf{b}^{[1]}$$
$$A^{[1]} = g(Z^{[1]})$$
$$Z^{[2]} = W^{[2]}\mathbf{a}^{[1]} + \mathbf{b}^{[2]}$$
$$\hat{Y} = softmax(Z^{[2]})$$

Softmax

$$g(z) = \frac{e^z}{\sum_{j=i}^{C} e_i^z}$$



$$Z = \begin{bmatrix} 5 \\ 2 \\ -1 \\ 3 \end{bmatrix} \quad e^z = \begin{bmatrix} e^5 \\ e^2 \\ e^{-1} \\ e^3 \end{bmatrix} \quad \sum_{j=i}^{C} e_i^z = 176.3$$

Probabilidades

$$\hat{y} = \begin{bmatrix} 0.842 \\ 0.042 \\ 0.002 \\ 0.114 \end{bmatrix}$$

Classe 0
Classe 1
Classe 2
Classe 3

Próxima aula

A7: MLP em Numpy

Aula prática sobre implementação de redes neurais profundas com Numpy.