Introdução às Redes Neurais

Eduardo Adame

Redes Neurais

20 de agosto de 2025

Revisão: Descida de Gradiente

O que aprendemos:

- Conceitos fundamentais de ML
- Regressão linear
- Descida de gradiente
- Taxa de aprendizado (α)
- SGD vs. Batch GD
- Critérios de convergência

Algoritmo básico:

$$\boldsymbol{\beta}^{(t+1)} = \boldsymbol{\beta}^{(t)} - \alpha \nabla J(\boldsymbol{\beta}^{(t)}) \qquad (1)$$

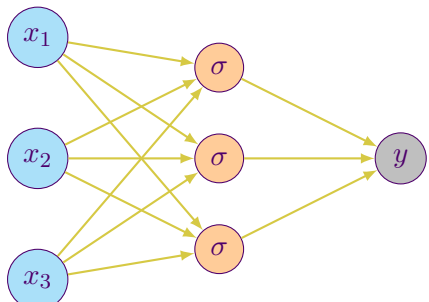
Hoje veremos:

- Como isso se conecta com redes neurais
- O papel do gradiente no treinamento

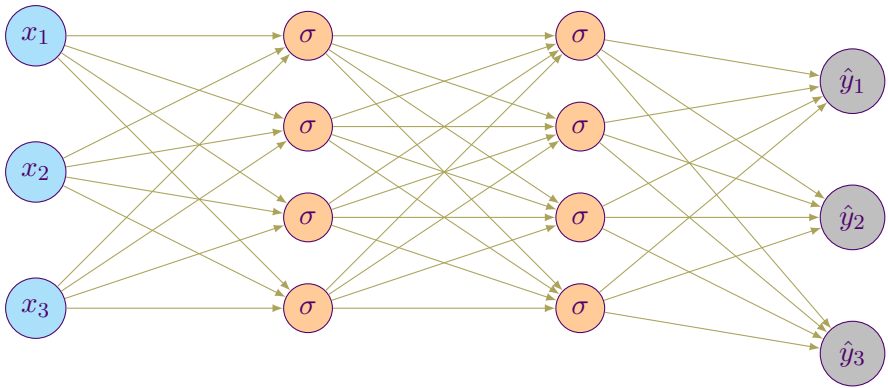
Motivação para Redes Neurais

Inspiração Biológica:

- Usar biologia como inspiração para modelo matemático
- Receber sinais de neurônios anteriores
- Gerar sinais (ou não) de acordo com as entradas
- Passar sinais para próximos neurônios
- Ao empilhar muitos neurônios, podemos criar modelos complexos

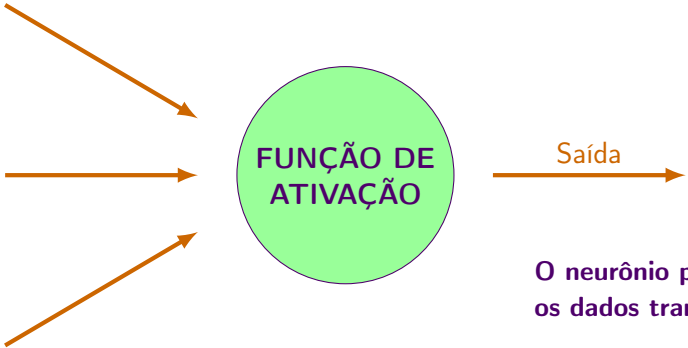


Estrutura de uma Rede Neural (Perceptron)



- Pode ser vista como um motor de computação complicado
- Vamos "treiná-la"usando nossos dados de treino
- Então (esperamos) ela dará boas respostas em novos dados

Visualização Básica do Neurônio



O neurônio produz os dados transformados

Dados da camada anterior

Alguma forma de computação transforma as entradas

Neurônio com Pesos e Bias

$$z = x_1 w_1 + x_2 w_2 + x_3 w_3 + b$$

Notação Vetorial

- **Cálculo do Neurônio**

$$z = b + \sum_{i=1}^{m} x_i w_i \tag{2}$$

$$z = b + \mathbf{x}^T \mathbf{w} \tag{3}$$

$$a = f(z) \tag{4}$$

Nomenclatura:

- z = "net input"(entrada líquida)
- b = "bias term"(termo de viés)
- f = função de ativação
- a = saída para próxima camada

Dimensões:

- $\mathbf{x}$ = vetor de entrada
- $\mathbf{w}$ = vetor de pesos
- b = escalar (bias)
- a = escalar (ativação)

Relação com Regressão Logística

Quando escolhemos a função sigmoid: $f(z) = \frac{1}{1+e^{-z}}$

$$z = b + \sum_{i=1}^{m} x_i w_i = x_1 w_1 + x_2 w_2 + \cdots + x_m w_m + b \tag{5}$$

Então um neurônio é simplesmente uma "unidade"de regressão logística!

pesos	$\Leftrightarrow$	coeficientes
entradas	$\Leftrightarrow$	variáveis
termo de bias	$\Leftrightarrow$	termo constante

Insight Importante

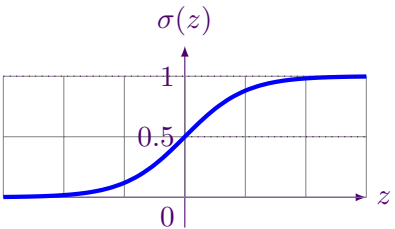
Um neurônio com ativação sigmoid é exatamente uma regressão logística!

A Função Sigmoid

$$\sigma(z) = \frac{1}{1 + e^{-z}} \qquad (6)$$

Propriedades:

- Domínio: $(-\infty, +\infty)$
- Imagem: $(0, 1)$
- Sempre crescente
- Ponto de inflexão em $z = 0$
- $\sigma(0) = 0.5$



Propriedade Importante da Sigmoid

Derivada da Função Sigmoid

$$\sigma(z) = \frac{1}{1+e^{-z}} \tag{7}$$

$$\sigma'(z) = \frac{e^{-z}}{(1+e^{-z})^2} \tag{8}$$

$$= \frac{1}{1+e^{-z}} \cdot \frac{e^{-z}}{1+e^{-z}} \tag{9}$$

$$= \frac{1}{1+e^{-z}} \cdot \left(1 - \frac{1}{1+e^{-z}}\right) \tag{10}$$

$$= \sigma(z) \cdot (1 - \sigma(z)) \tag{11}$$

Exemplo de Computação do Neurônio

Dados de entrada:

- $x_1 = 0.9$
- $x_2 = 0.2$
- $x_3 = 0.3$

Pesos:

- $w_1 = 2$
- $w_2 = 3$
- $w_3 = -1$
- $b = 0.5$

Cálculo:

$$z = 0.9(2) + 0.2(3) + 0.3(-1) + 0.5 \quad (12)$$
$$= 1.8 + 0.6 - 0.3 + 0.5 \quad (13)$$
$$= 2.6 \quad (14)$$

$$f(z) = \frac{1}{1 + e^{-2.6}} \quad (15)$$
$$= 0.93 \quad (16)$$

O neurônio produziria o valor 0.93

Por que Redes Neurais?

Um único neurônio (como regressão logística):

- Só permite uma fronteira de decisão linear
- Limitado a problemas linearmente separáveis
- Não consegue capturar relações complexas

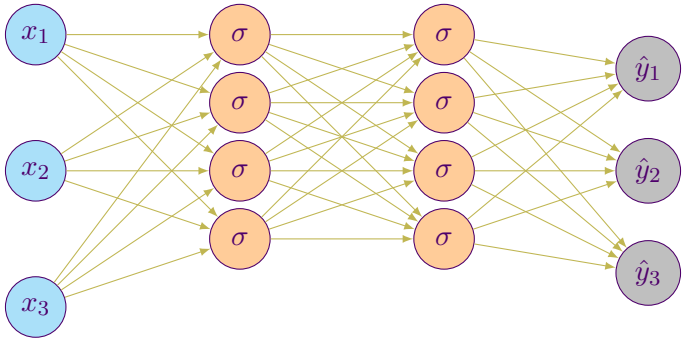
Redes com múltiplos neurônios:

- Podem criar fronteiras não-lineares
- Capturam interações complexas
- Aproximam qualquer função contínua
- Resolvem problemas do mundo real!

Teorema da Aproximação Universal

Uma rede neural com uma camada oculta e suficientes neurônios pode aproximar qualquer função contínua!

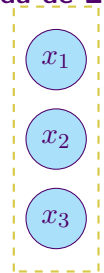
Rede Neural Feedforward



- Feedforward: informação flui em uma direção (entrada $\rightarrow$ saída)
- Cada conexão tem um peso associado
- Cada neurônio tem um bias

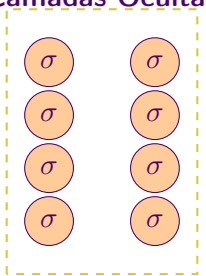
Camadas da Rede Neural

Camada de Entrada



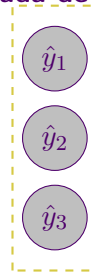
- Recebe os dados
- Não processa
- Apenas repassa

Camadas Ocultas



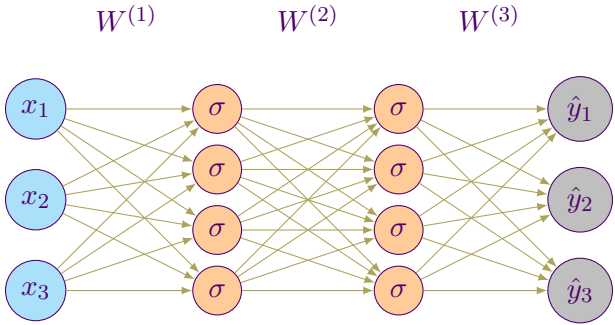
- Processamento
- Extração de features
- Não-linearidade

Camada de Saída



- Produz resultado
- Formato depende do problema
- Classificação ou regressão

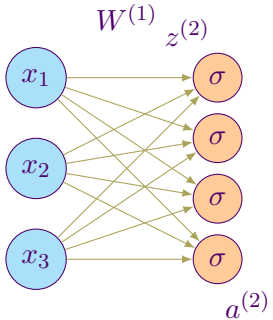
Pesos (Representados por Matrizes)



- $W^{(1)}$: matriz de pesos da entrada para primeira camada oculta (3×4)
- $W^{(2)}$: matriz de pesos entre camadas ocultas (4×4)
- $W^{(3)}$: matriz de pesos da última camada oculta para saída (4×3)

Representação Matricial da Computação

Para a primeira camada oculta:



Cálculo:

$$\mathbf{x} = [x_1, x_2, x_3] \quad (1 \times 3) \tag{17}$$

$$W^{(1)} \text{ é uma matriz } 3 \times 4 \tag{18}$$

$$z^{(2)} = \mathbf{x}W^{(1)} \quad (1 \times 4) \tag{19}$$

$$a^{(2)} = \sigma(z^{(2)}) \quad (1 \times 4) \tag{20}$$

onde $\mathbf{x} = a^{(1)}$ (entrada é a ativação da "camada 0")

Continuando a Computação

Para uma única instância de treinamento

Entrada: vetor $\mathbf{x}$ (vetor linha de tamanho 3)
Saída: vetor $\hat{\mathbf{y}}$ (vetor linha de tamanho 3)

$$z^{(2)} = \mathbf{x}W^{(1)} \qquad a^{(2)} = \sigma(z^{(2)}) \tag{21}$$

$$z^{(3)} = a^{(2)}W^{(2)} \qquad a^{(3)} = \sigma(z^{(3)}) \tag{22}$$

$$z^{(4)} = a^{(3)}W^{(3)} \qquad \hat{\mathbf{y}} = \mathsf{softmax}(z^{(4)}) \tag{23}$$

Múltiplos Pontos de Dados

Processamento em Lote

Na prática, fazemos essas computações para muitos pontos de dados ao mesmo tempo, "empilhando"as linhas em uma matriz. Mas as equações permanecem as mesmas!

Entrada: matriz X (uma matriz $n \times 3$) - cada linha é uma instância
Saída: matriz $\hat{Y}$ (uma matriz $n \times 3$) - cada linha é uma predição

$$Z^{(2)} = XW^{(1)} \qquad A^{(2)} = \sigma(Z^{(2)}) \tag{24}$$

$$Z^{(3)} = A^{(2)}W^{(2)} \qquad A^{(3)} = \sigma(Z^{(3)}) \tag{25}$$

$$Z^{(4)} = A^{(3)}W^{(3)} \qquad \hat{Y} = \text{softmax}(Z^{(4)}) \tag{26}$$

Agora sabemos como redes neurais feedforward fazem computações.

Próximo passo: aprender como ajustar os pesos para aprender dos dados.

O que já sabemos:

- Forward pass (propagação direta)
- Estrutura da rede
- Representação matricial
- Funções de ativação

Próxima aula:

- Backpropagation
- Cálculo de gradientes
- Atualização de pesos
- Treinamento da rede

Obrigado!

Alguma dúvida?

Agora vamos para os exercícios!