UNIVERSIDADE FEDERAL DO MARANHÃO

CAMPUS DE SÃO LUÍS - CIDADE UNIVERSITÁRIA

ENGENHARIA DA COMPUTAÇÃO

REDES NEURAIS

TURMA 1

ATIVIDADE DE REGRESSAO LINEAR

PROFESSOR:THALES LEVI AZEVEDO VALENTE

ALUNOS:

EMANUEL LOPES SILVA - 2021017818

São Luís – MA

24/04/2025

EMANUEL LOPES SILVA - 2021017818

REDES NEURAIS

TURMA 1

ATIVIDADE DE REGRESSÃO LINEAR

Relatório acadêmico apresentado à disciplina de Redes Neurais, no curso de Engenharia da Computação da Universidade Federal do Maranhão, como parte da avaliação do componente curricular, sob a tutoria do Prof.Dr. Thales Valente

São Luís - MA

24/04/2025

SUMÁRIO

RESUMO:

Este trabalho apresenta um estudo aplicado sobre regressão linear utilizando o algoritmo de descida do gradiente, com foco na análise do impacto da taxa de aprendizado e da inicialização dos pesos sobre a convergência da função de custo. Através da implementação prática em Python e da geração de gráficos interpretativos, foram explorados os efeitos de diferentes configurações desses hiperparâmetros. Foram realizados experimentos comparativos que demonstraram como valores inadequados de taxa de aprendizado podem comprometer a estabilidade do treinamento e como diferentes pontos iniciais afetam a trajetória de otimização. Adicionalmente, o estudo estabelece conexões com o treinamento de redes neurais profundas, discutindo a importância de estratégias de inicialização como Xavier e He, além do papel do fine-tuning. Os resultados obtidos reforçam a relevância do estudo da regressão linear como base para a compreensão de modelos mais complexos em aprendizado de máquina.

Palavras-chave: Regressão linear, Descida do gradiente, Inicialização de pesos, Taxa de aprendizado, Redes neurais.

ABSTRACT:

This work presents an applied study on linear regression using the gradient descent algorithm, focusing on analyzing the impact of the learning rate and weight initialization on the convergence of the cost function. Through practical implementation in Python and the generation of interpretive plots, the effects of different hyperparameter configurations were explored. Comparative experiments demonstrated how inappropriate learning rate values can compromise training stability and how different initial points affect the optimization path. Additionally, the study draws connections to deep neural network training, discussing the importance of initialization strategies such as Xavier and He, as well as the role of fine-tuning. The results reinforce the importance of linear regression as a theoretical and practical foundation for understanding more complex machine learning models.

Keywords: Linear regression, Gradient descent, Weight initialization, Learning rate, Neural networks.

1. Introdução

A regressão linear é uma técnica estatística fundamental utilizada para modelar a relação entre uma variável dependente (ou resposta) e uma ou mais variáveis independentes (ou explicativas). Essa relação é representada por uma equação linear, cuja forma mais simples é expressa por $y = \beta 0 + \beta 1x + \varepsilon$, em que y é a variável resposta, x é a variável preditora, β0 é o intercepto, β1 o coeficiente angular, e ε o termo de erro aleatório (TRANMER et al., 2020).

Essa técnica é valorizada pela sua simplicidade interpretativa, robustez teórica e versatilidade prática. Pode ser utilizada para prever valores futuros, interpretar relações entre variáveis, e apoiar a tomada de decisões baseada em dados quantitativos, mesmo em cenários onde o sinal da variável resposta é fraco ou a amostra é limitada (SU; YAN; TSAI, 2012; KILIÇ, 2013).

A forma mais simples é a **regressão linear simples**, onde há apenas uma variável preditora. Quando há múltiplas variáveis explicativas, o modelo torna-se uma **regressão linear múltipla**, oferecendo maior capacidade preditiva (KILIÇ, 2013). Os parâmetros estimados via mínimos quadrados possuem propriedades ótimas, como o fato de serem estimadores lineares insesgados de menor variância.

Além da forma clássica, diversos aprimoramentos têm sido propostos. Por exemplo, o modelo de regressão modal linear propõe estimar a moda condicional de Y dado X, ao invés da média, oferecendo maior robustez em distribuições assimétricas (YAO; LI, 2014). Outro avanço são os métodos de regressão robusta, como a regressão linear baseada em setores (SBLR), que busca minimizar o impacto de outliers sobre a estimativa do plano de regressão (NAGY, 2018).

A regressão linear também é frequentemente usada como base para métodos mais complexos, como modelos de regularização (ridge, lasso), regressões não lineares transformadas, e algoritmos de aprendizado de máquina supervisionado. Em contextos de incerteza na escolha do modelo, abordagens que **combinam múltiplos modelos de regressão linear**, como o método ARMS (*Adaptive Regression by Mixing*), podem ser mais eficazes

que a seleção de um único modelo, pois reduzem a instabilidade e aumentam a precisão das previsões (YUAN; YANG, 2005).

Essas técnicas têm aplicação em áreas tão diversas quanto a economia, medicina, ciências sociais, engenharia e ciência de dados, sendo especialmente úteis para prever valores, identificar padrões e validar hipóteses empíricas (SPÄTH, 1992; SU; YAN; TSAI, 2012).

Neste trabalho, utilizamos a regressão linear simples com descida do gradiente como base para explorar o comportamento de diferentes configurações de taxa de aprendizado (α) e inicialização dos pesos (θ). A partir da implementação dos componentes fundamentais do modelo, foram realizados experimentos comparativos para analisar os efeitos desses parâmetros no processo de aprendizagem. Os resultados são discutidos à luz das propriedades matemáticas do algoritmo e visualizados por meio de gráficos interativos.

2. Fundamentação Teórica

2.1. Regressão Linear

A regressão linear é uma técnica estatística utilizada para modelar a relação entre uma variável dependente Y e uma ou mais variáveis independentes X, assumindo uma estrutura linear entre elas. No caso mais simples, conhecido como regressão linear simples, a equação do modelo é dada por:

$$h\theta(x) = \theta 0 + \theta 1 \quad (1)$$

onde $h\theta(x)$ representa a hipótese do modelo, θ0 é o intercepto e θ1 o coeficiente angular (SU; YAN; TSAI, 2012).

A popularidade do modelo decorre de sua simplicidade interpretativa e da elegância matemática que o sustenta. Os coeficientes estimados podem ser interpretados como a variação esperada em Y para uma unidade de variação em X, mantendo as demais variáveis constantes (SU; YAN; TSAI, 2012). Além disso, a regressão linear é considerada um bloco de construção fundamental para modelos estatísticos mais complexos.

2.2. Função de Custo

A avaliação da qualidade do ajuste de um modelo de regressão linear é realizada por meio da função de custo, geralmente definida como o erro quadrático médio. Essa função quantifica a diferença entre os valores previstos pelo modelo e os valores reais observados. A forma matemática mais comum é:

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h\theta(x^{(i)}) - y^{(i)})^2 \quad (2)$$

Onde m é o número de exemplos de treinamento, $h\theta(x^{(i)})$ é a predição do modelo, e $y^{(i)}$ o valor real correspondente. O fator $\frac{1}{2}$ é utilizado por conveniência na derivação da atualização dos parâmetros (SU; YAN; TSAI, 2012).

Esse tipo de função de custo garante que erros maiores sejam penalizados de forma mais severa, favorecendo ajustes que minimizem discrepâncias significativas.

2.3. Algoritmo de Descida Gradiente

O algoritmo de descida do gradiente é um método iterativo utilizado para minimizar a função de custo e encontrar os valores ótimos para os parâmetros θ. A atualização dos parâmetros ocorre segundo a seguinte regra:

$$\theta := \theta - \alpha \nabla J(\theta) \quad (3)$$

Onde α é a taxa de aprendizado e $\nabla J(\theta)$ representa o gradiente da função de custo com relação aos parâmetros. Essa derivada indica a direção de maior crescimento da função, e, ao subtrairmos esse valor, movemo-nos na direção de maior declínio (HARTSHORN, 2017).

A taxa de aprendizado α é um hiperparâmetro crucial: valores muito baixos tornam o processo lento e sujeitos a mínimos locais; valores altos podem causar oscilações ou até divergência. Da mesma forma, a escolha da **inicialização dos pesos** θ afeta diretamente a

trajetória e a convergência do algoritmo, sendo um conceito intimamente relacionado com práticas como *fine-tuning* em redes neurais profundas (YAO; LI, 2014).

3. Implementação Prática

3.1 Estrutura dos Arquivos Implementados

Para a execução do experimento, foram desenvolvidos e organizados arquivos modulares em Python, cada um responsável por um componente específico do pipeline de regressão linear. Essa separação de responsabilidades favorece a legibilidade, o reuso de código e a facilidade de manutenção.

3.1.1. warm_up_exercise.py

Este módulo contém funções introdutórias para familiarização com operações matriciais básicas utilizando NumPy. Inclui a criação de uma matriz identidade de dimensão 5×5 e funções auxiliares para empilhamento de colunas, simulação de produto matricial e cálculo de erros. Essas funções foram utilizadas para validar o funcionamento de multiplicações vetoriais e construção do vetor de entrada com bias.

3.1.2. plot_data.py

Responsável pela visualização dos dados brutos de entrada. Utiliza a biblioteca matplotlib.pyplot para plotar os dados de treinamento, destacando os pontos no gráfico com marcações vermelhas ('rx'). Essa visualização auxilia na análise exploratória e na compreensão da relação entre a variável independente (população) e a variável dependente (lucro).

3.1.3. compute_cost.py

Contém a implementação da função de custo J(θ) da regressão linear, baseada no erro quadrático médio. Essa função calcula a média dos quadrados dos resíduos entre os valores previstos pelo modelo e os valores reais do conjunto de treinamento. Foi essencial para avaliar o quão bem o modelo se ajustava aos dados e acompanhar a evolução do erro durante a descida do gradiente.

3.1.4. gradient_descent.py

Implementa o algoritmo iterativo de otimização conhecido como descida do gradiente. A função recebe como entrada os dados de treinamento (com coluna de bias), o vetor de saída esperado, os parâmetros iniciais θ, a taxa de aprendizado α e o número de iterações. A cada iteração, os parâmetros θ são atualizados com base no gradiente da função de custo. O script também retorna o histórico dos valores de θ e da função de custo J(θ) em cada passo, permitindo visualizações posteriores da trajetória de otimização.

3.1.5. regressao-linear-ex1.py

O arquivo principal do projeto, regressao-linear-ex1.py, atua como controlador da execução da regressão linear passo a passo, estruturando as chamadas às funções auxiliares implementadas nos módulos separados. A seguir, detalha-se sua estrutura e lógica:

3.1.5.1. Aquecimento e Validação de Operações Matriciais

A primeira parte do main() executa uma série de exercícios introdutórios contidos no módulo warm_up_exercise.py, que incluem:

- Impressão de uma matriz identidade 5×5.

- Criação de um vetor coluna de 1s.
- Empilhamento de uma coluna de bias com dados vetoriais.
- Execução de produto matricial X @ θ.
- Cálculo de erros quadráticos.
- Avaliação do custo médio com base nesses erros.

Essa seção ajuda a testar a correta manipulação de arrays no NumPy, essencial para garantir que a implementação da regressão linear funcione adequadamente.

3.1.5.2. Pré-processamento e Treinamento

Após o aquecimento, o script:

1. Carrega os dados de entrada do arquivo ex1data1.txt, que contém valores de população (input) e lucro (output).
2. Plota os dados usando a função plot_data(x, y), facilitando a análise visual inicial da tendência linear.
3. Adiciona uma coluna de 1s à matriz de entrada para representar o termo de bias (θ0).
4. Inicializa os parâmetros θ com zeros ou valores arbitrários (por exemplo, [8.5, 4.0]).
5. Define os hiperparâmetros: taxa de aprendizado (α = 0.01) e número de iterações (1500).
6. Calcula a função de custo inicial com compute_cost, para validar o ponto de partida.

7. Executa o algoritmo de descida do gradiente com gradient_descent, que retorna os parâmetros otimizados, a curva de custo e o histórico de θ.

3.1.5.3. Visualizações e Análises

O script gera uma série de visualizações para inspecionar os resultados da regressão linear:

- Curva de convergência do custo durante as iterações, mostrando se o treinamento foi eficaz.
- Gráfico da reta de regressão ajustada sobre os dados reais.
- Superfície 3D da função de custo em relação a θ0 e θ1, exibindo o comportamento da função de erro.
- Gráfico de contorno da função de custo, com marcação do mínimo.
- Trajetória dos parâmetros θ sobre o contorno, indicando como o gradiente desceu até o ponto mínimo.
- Visualização 3D da trajetória do gradiente, sobre a superfície de custo.

3.1.5.4. Experimentos Comparativos Interativos

Ao final da execução principal, o script chama duas funções extras:

1. experimento_taxas_aprendizado()
   Solicita três valores de α ao usuário e plota a curva de convergência da função de custo para cada um, comparando velocidade e estabilidade.

2. experimento_inicializacoes()
   Permite ao usuário digitar três inicializações fixas para os pesos θ e compara, junto com três iniciais aleatórias, a trajetória de cada uma no gráfico de contorno da função

de custo.

Esses dois blocos proporcionam uma compreensão profunda sobre os efeitos da escolha dos hiperparâmetros no desempenho e convergência do modelo.

3.2 Geração dos Gráficos

A análise visual foi parte fundamental do projeto, tanto para verificar o comportamento dos parâmetros θ quanto para validar a efetividade da regressão linear aplicada. A seguir, são descritos os principais gráficos gerados ao longo do processo:

- **Curva de Convergência da Função de Custo**

  A função de custo **J(θ)** expressa o erro médio quadrático entre as predições do modelo e os valores reais observados. Ao aplicar o algoritmo de descida do gradiente, o vetor de parâmetros **θ** é atualizado iterativamente para minimizar essa função. O gráfico da curva de convergência mostra **como o valor de J(θ) evolui ao longo das iterações**, sendo, portanto, uma ferramenta essencial para verificar a eficácia do processo de otimização.

  Em teoria, uma curva de custo suave e monotonicamente decrescente indica que o algoritmo está aprendendo de forma estável e eficiente. Já oscilações ou crescimento no custo indicam possíveis problemas, como taxa de aprendizado inadequada ou dados não normalizados. Assim, esse gráfico serve como **indicador visual de convergência e estabilidade do aprendizado**.

- **Ajuste da Reta de Regressão sobre os Dados**

  Após o treinamento do modelo, a reta ajustada foi plotada sobre os dados de entrada. Essa visualização ilustra como o modelo aprendido aproxima os dados reais. A linha é gerada a partir da equação do modelo:

$$h(x) = \theta_0 + \theta_1 \cdot x \qquad (4)$$

O gráfico da reta ajustada sobre os dados permite verificar visualmente o grau de aderência do modelo aos dados observados. Quando a reta passa próxima à maioria dos pontos, é um indicativo de que o modelo conseguiu captar a tendência presente no conjunto de treinamento.

Do ponto de vista metodológico, esse gráfico é a manifestação geométrica da função hipótese, e serve como base para validação qualitativa do modelo ajustado.

- **Superfície 3D da Função de Custo com Trajetória do Gradiente**
  A superfície tridimensional da função de custo $\mathbf{J(\theta_0, \theta_1)}$ representa, de forma geométrica, como o erro varia com diferentes combinações dos parâmetros $\boldsymbol{\theta_0}$ e $\boldsymbol{\theta_1}$. Teoricamente, como a função de custo da regressão linear é convexa, essa superfície assume a forma de uma parábola aberta, com um único ponto de mínimo global.

  Ao sobrepor a trajetória do gradiente sobre essa superfície, é possível visualizar como os parâmetros são atualizados a cada iteração, descendo progressivamente em direção ao vale do mínimo da função. Essa visualização é extremamente valiosa para entender o comportamento geométrico do algoritmo de otimização.

- **Gráfico de Contorno com Trajetória do Gradiente**
  O gráfico de contorno representa a função de custo **em duas dimensões**, como um conjunto de curvas de nível. Cada curva representa todos os pares (θ_0, θ_1) que resultam em um mesmo valor de $\mathbf{J(\theta)}$. Esse tipo de visualização é amplamente utilizado em problemas de otimização para **identificar regiões de baixo custo** e analisar o percurso do algoritmo ao longo da superfície.

  No contexto da regressão linear, a sobreposição da trajetória da descida do gradiente sobre o contorno permite acompanhar o caminho percorrido pelos parâmetros a partir de suas inicializações até a convergência. Essa abordagem permite investigar se o modelo está convergindo corretamente e como a inicialização afeta esse trajeto, oferecendo insights sobre a dinâmica do aprendizado.

Essas representações gráficas não apenas facilitaram a interpretação do comportamento da regressão linear, mas também serviram como base visual para os experimentos comparativos relacionados à taxa de aprendizado e à inicialização dos pesos.

4. Resultados dos Experimentos

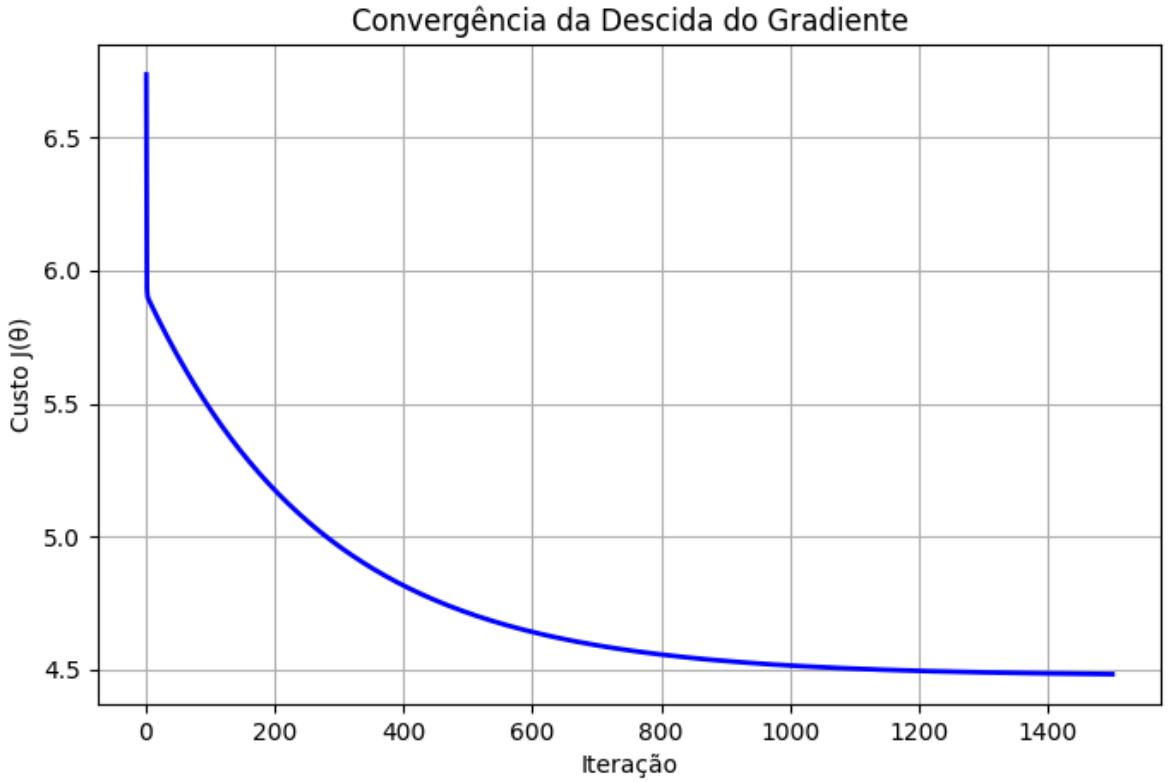
4.1 Análise dos Gráficos de Regressão Linear

Antes da realização dos experimentos comparativos, foram gerados gráficos fundamentais para compreender o funcionamento e a qualidade do ajuste do modelo linear.

- **Gráfico 1**: Curva de Convergência da Função de Custo

A Figura 1 ilustra a evolução do valor da função de custo J(θ) ao longo das iterações do algoritmo de descida do gradiente, considerando uma taxa de aprendizado fixa de α=0,01. A função de custo utilizada corresponde ao erro quadrático médio entre os valores previstos pelo modelo e os valores reais do conjunto de dados.

Figura 1 - Convergência da Função de Custo



Autoria Própria

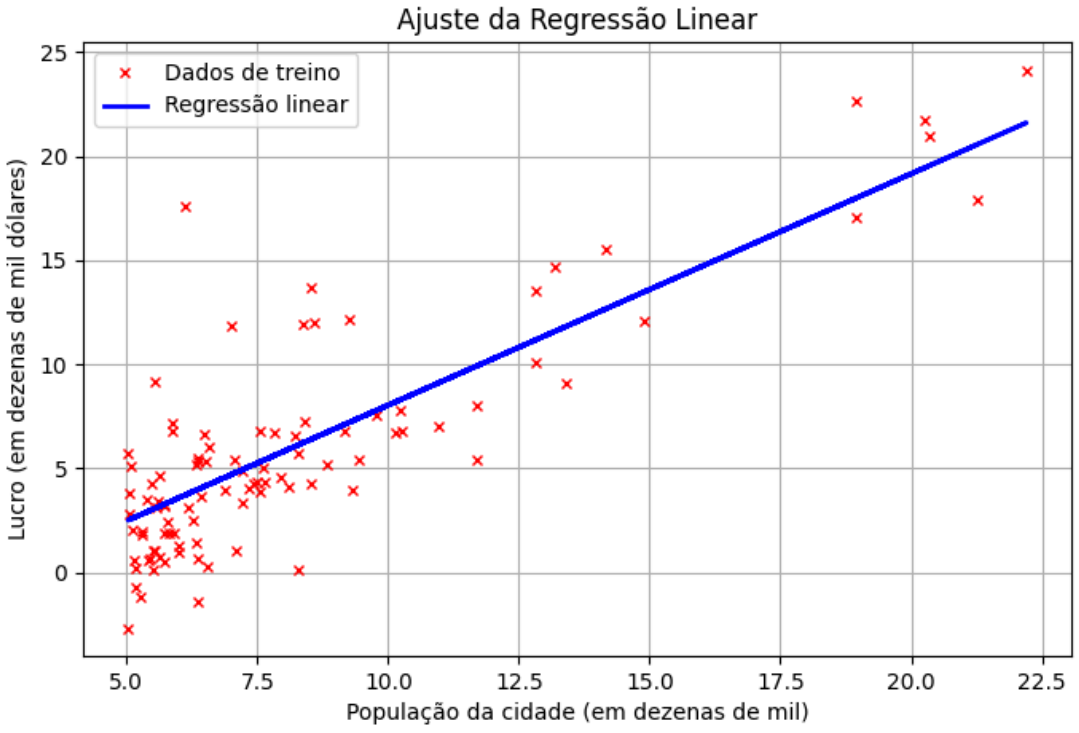
Observa-se que o valor de J(θ) diminui de forma contínua e gradual, até atingir uma região de estabilização, indicando que o algoritmo convergiu para um ponto próximo do mínimo global da função. O comportamento da curva é típico de um processo de otimização bem calibrado: inicia com um declínio acentuado e, posteriormente, desacelera à medida que se aproxima do ótimo.

Esse gráfico é fundamental para a validação empírica do funcionamento da descida do gradiente, pois permite avaliar a eficiência do algoritmo com a taxa de aprendizado escolhida. A ausência de oscilações ou crescimento abrupto na curva confirma que o valor de α foi adequadamente definido, promovendo estabilidade e eficácia no processo de aprendizagem.

- **Gráfico 2**: Ajuste da Reta de Regressão

A Figura 2 apresenta a reta de regressão ajustada sobre os dados de treinamento, resultante da otimização dos parâmetros θ_0 e θ_1 por meio do algoritmo de descida do gradiente. Esta visualização é fundamental para uma avaliação qualitativa do modelo, pois permite observar diretamente o alinhamento da hipótese linear com os dados reais.

Figura 2 - Ajuste da Regressão Linear



Autoria Própria

A reta foi traçada a partir da função preditiva $h(x) = \theta_0 + \theta_1 \cdot x$, onde x representa a variável independente (população da cidade, em dezenas de milhares) e $h(x)$ representa a variável dependente (lucro, em dezenas de mil dólares). Os pontos vermelhos representam as amostras do conjunto de dados, enquanto a linha azul contínua indica a predição gerada pelo modelo ajustado.

Visualmente, observa-se que a reta passa próxima à maioria dos pontos, especialmente no centro da distribuição dos dados, o que sugere que a regressão linear conseguiu capturar a tendência geral da relação entre as variáveis. Embora desvios possam ocorrer para pontos extremos ou com maior variância, o padrão linear predominante é bem representado.

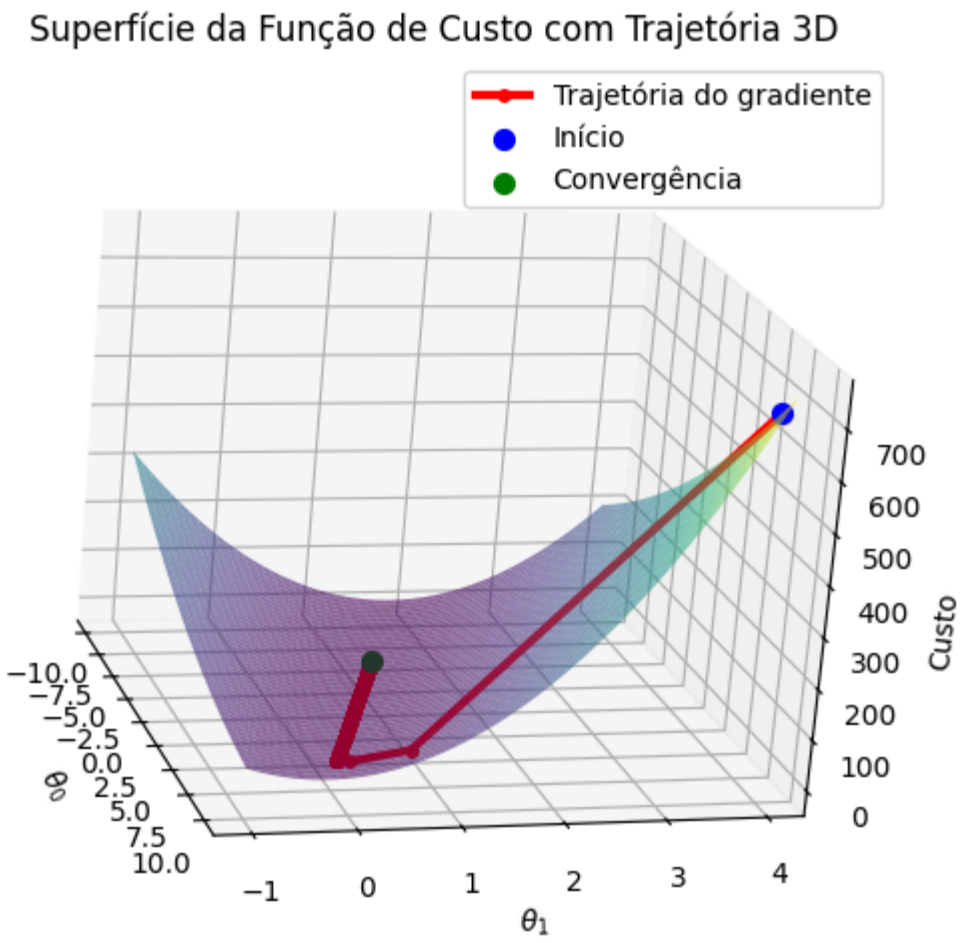
Esse gráfico permite validar se o modelo gerado é coerente com a estrutura dos dados e se pode ser utilizado para fazer previsões razoáveis dentro da faixa observada. Além disso, confirma a capacidade da regressão linear de modelar relações simples de tendência crescente ou decrescente com bom grau de interpretabilidade.

- **Gráfico 3**: Superfície 3D da Função de Custo

A Figura 3 apresenta a representação tridimensional da função de custo $J(\theta_0, \theta_1)$, plotada em função dos dois parâmetros do modelo linear. Essa superfície ilustra geometricamente como o erro do modelo varia conforme diferentes combinações dos coeficientes θ_0 (intercepto) e θ_1 (inclinação). A forma da superfície evidencia que a função de custo da regressão linear é convexa, o que garante a existência de um único mínimo global.

Sobre essa superfície, é sobreposta a trajetória do vetor de parâmetros θ ao longo das iterações do algoritmo de descida do gradiente. Essa linha, geralmente representada em vermelho, indica o caminho percorrido desde a inicialização dos parâmetros até sua convergência. O comportamento descendente e suave da trajetória evidencia que o algoritmo foi capaz de reduzir progressivamente o valor da função de custo, aproximando-se do mínimo global.

Figura 3 - Superfície 3D da Função de Custo



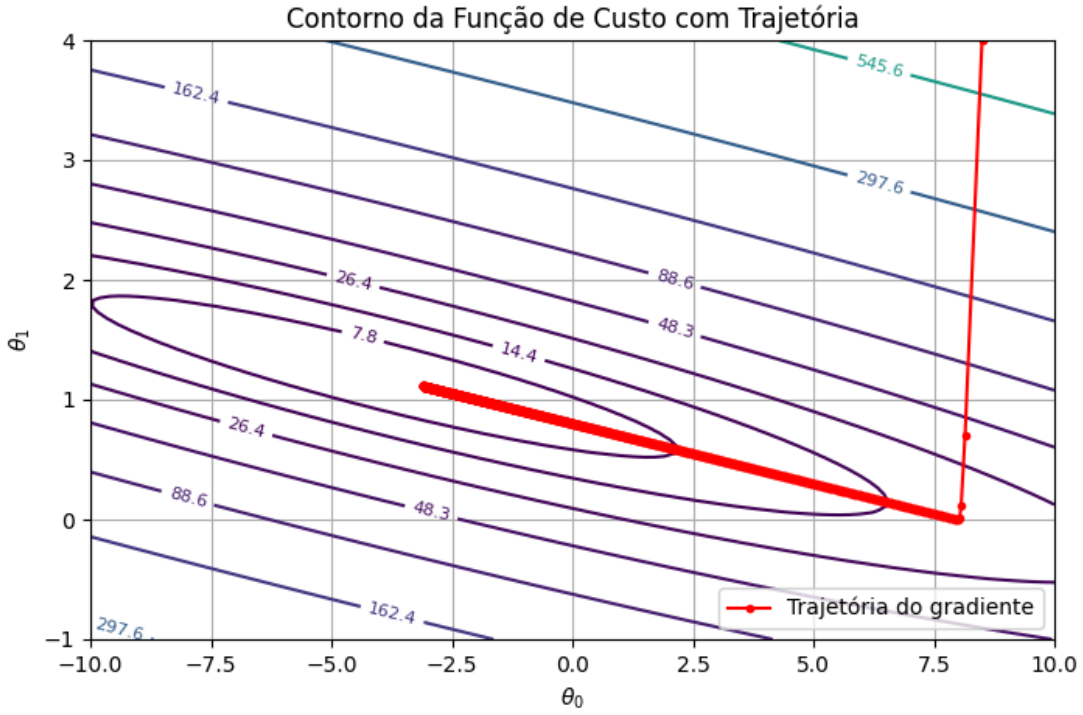
Autoria Própria

Do ponto de vista teórico, essa visualização reforça a compreensão geométrica da descida do gradiente como um processo iterativo de otimização. Ao representar a paisagem da função de custo, o gráfico permite verificar se os parâmetros estão de fato sendo conduzidos para a região de menor erro, além de avaliar visualmente a eficiência da taxa de aprendizado escolhida.

- **Gráfico 4**: Contorno da Função de Custo

A Figura 4 apresenta o gráfico de contorno da função de custo $J(\theta_0, \theta_1)$, também conhecido como gráfico de curvas de nível. Cada linha neste tipo de gráfico representa um conjunto de pares (θ_0, θ_1) que resultam no mesmo valor da função de custo, permitindo a visualização bidimensional da "paisagem" da função de erro.

Figura 4 - Contorno da Função de Custo



Autoria Própria

Sobre esse mapa de níveis, é traçada a trajetória percorrida pelo vetor de parâmetros θ ao longo das iterações do algoritmo de descida do gradiente. A linha pontilhada (ou contínua), geralmente em vermelho, conecta os pontos sucessivos de atualização dos parâmetros, desde a inicialização até a convergência no mínimo global.

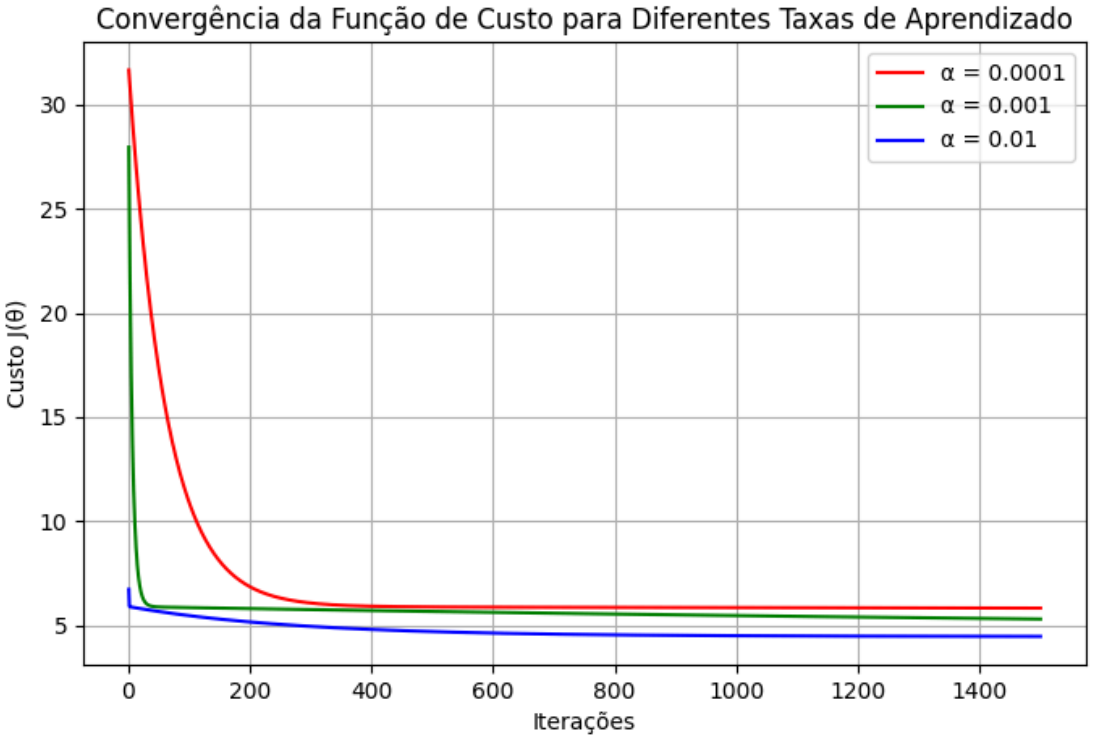
Este tipo de gráfico fornece uma visão clara da dinâmica de otimização do algoritmo: é possível observar se o gradiente está conduzindo os parâmetros na direção correta e se a taxa de aprendizado está produzindo passos suaves e eficientes. Trajetórias que seguem diretamente até o centro das curvas indicam boa calibração, enquanto caminhos tortuosos ou que se desviam sugerem a necessidade de ajustes nos hiperparâmetros.

Do ponto de vista metodológico, o gráfico de contorno complementa a visualização tridimensional da função de custo, oferecendo uma perspectiva mais direta e plana do processo de convergência, e é particularmente útil para comparações entre diferentes inicializações, como será explorado nos experimentos subsequentes.

4.2 Variação da Taxa de Aprendizado (α)

A taxa de aprendizado α é um dos hiperparâmetros mais sensíveis e influentes no desempenho do algoritmo de descida do gradiente. Neste experimento, foi analisado o impacto de três valores distintos de α: 0.0001, 0.001 e 0.01, mantendo-se os demais parâmetros do modelo constantes. O objetivo foi observar como cada configuração influencia a velocidade, a estabilidade e a qualidade da convergência da função de custo J(θ).

Figura 5 - Variação da Taxa de Aprendizado



Autoria Própria

A Figura 5 apresenta a evolução da função de custo J(θ) ao longo das iterações para cada uma das três taxas de aprendizado analisadas. As curvas mostram como a escolha de α afeta diretamente a eficiência do treinamento.

Os resultados apresentados foram:

- **α = 0.0001 (muito pequeno):**
  A curva (vermelha) apresenta uma convergência muito lenta. Apesar da estabilidade, o algoritmo avança de forma extremamente gradual, sendo ineficiente para problemas maiores ou que demandem agilidade.

- **α = 0.001 (intermediário):**
  A curva (verde) mostra um equilíbrio razoável entre estabilidade e velocidade. O modelo converge de forma segura, embora ainda leve mais tempo em relação à taxa maior.

- **α = 0.01 (ideal neste caso):**
  A curva (azul) atinge o menor valor de custo mais rapidamente. O comportamento é estável, sem oscilações ou divergência, indicando que este valor promove uma convergência eficiente e confiável.

4.3 Inicialização dos Pesos (θ inicial):

A inicialização dos parâmetros θ é uma etapa crucial no processo de aprendizado supervisionado, especialmente em algoritmos iterativos como a descida do gradiente. Neste experimento, foram testadas seis diferentes inicializações dos pesos para avaliar sua influência no trajeto de convergência dentro da superfície da função de custo.

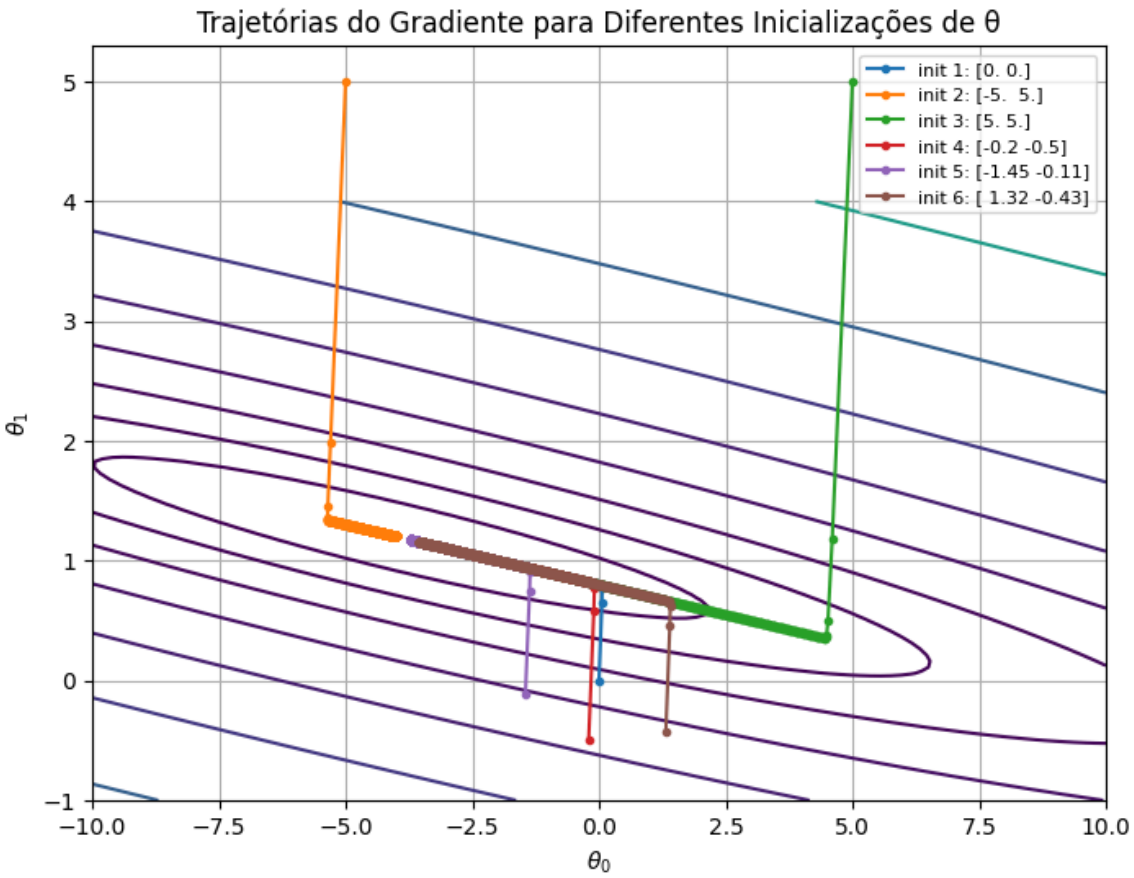
Foram utilizadas:

- **Três inicializações fixas:** [0, 0], [5, 5] e [-5, 5].

- **Três inicializações aleatórias:** amostradas da distribuição normal padrão $N(0, 1)$

A taxa de aprendizado foi mantida constante em α=0,01 para isolar o impacto exclusivamente da inicialização dos parâmetros.

Figura 6 - Diferentes Inicializações de θ



Autoria Própria

A Figura 6 apresenta o gráfico de contorno da função de custo $J(\theta_0, \theta_1)$, sobre o qual estão sobrepostas as trajetórias percorridas pelo vetor de parâmetros θ a partir de cada uma das seis inicializações. Cada linha indica o caminho seguido pela descida do gradiente, com cada ponto representando o valor de θ em uma iteração sucessiva.

É possível discutir os resultados obtidos da seguinte forma:

- **Convergência ao mínimo global:**

  Todas as trajetórias convergem para o mesmo ponto central na superfície de custo, confirmando que a função de custo da regressão linear é convexa e possui um único mínimo global.

- **Impacto da distância inicial:**

  As inicializações fixas mais distantes do mínimo, como [5, 5] e [-5, 5], apresentaram trajetórias mais longas e abruptas, com múltiplas inflexões visíveis no gráfico. Isso indica maior esforço computacional até atingir a convergência.

- **Inicializações aleatórias com trajetórias distintas:**

  As três amostragens aleatórias resultaram em percursos variados:

  - A inicialização init 4 ([-0.20, -0.50]) seguiu um caminho relativamente direto.
  - A init 5 ([-1.45, -0.11]) iniciou mais afastada e desceu rapidamente.
  - A init 6 ([1.32, -0.43]) começou mais próximo do mínimo e convergiu com poucos ajustes.

- **Trajetórias visuais:**

  O gráfico evidencia que, embora todas as trajetórias atinjam o mesmo destino, os caminhos percorridos no espaço de parâmetros variam consideravelmente, reforçando a importância da escolha da inicialização, principalmente em modelos não lineares.

- **Relevância para modelos complexos:**

  Em arquiteturas como redes neurais profundas, onde a função de custo não é convexa, más inicializações podem levar a mínimos locais ou impedir a convergência, tornando estratégias como inicialização de Xavier, He ou normalização dos dados essenciais.

5. Discussão

Embora a regressão linear seja um modelo simples e de natureza convexa, as lições obtidas com seu estudo são diretamente aplicáveis a modelos mais complexos, como redes neurais profundas. A análise da influência da taxa de aprendizado e da inicialização dos parâmetros sobre a convergência do algoritmo de descida do gradiente fornece uma base sólida para compreender o comportamento de redes neurais, nas quais o processo de otimização se torna exponencialmente mais sensível.

5.1. Importância da Inicialização dos Pesos

Em redes neurais profundas, a escolha da inicialização dos pesos é um fator crítico para o sucesso do treinamento. Inicializações inadequadas podem fazer com que o gradiente desapareça (vanishing gradients) ou exploda (exploding gradients) ao longo das camadas, impedindo que os parâmetros se atualizem de maneira eficiente. Ao contrário da regressão linear, onde o custo é uma função convexa com único mínimo global, redes neurais possuem superfícies de erro altamente não convexas, repletas de mínimos locais, planos saddle e regiões de difícil navegação.

5.2. Estratégias de Inicialização: Xavier e He

Para mitigar esses problemas, foram desenvolvidas estratégias de inicialização mais sofisticadas, como:

- **Xavier Initialization** (ou Glorot): Ajusta a variância dos pesos com base na quantidade de entradas e saídas da camada, sendo ideal para funções de ativação como tanh ou sigmoid.

- **He Initialization:** Otimizada para funções ReLU, fornece uma distribuição de pesos que preserva a variância dos gradientes ao longo das camadas, evitando o colapso do sinal durante a propagação.

Essas técnicas buscam garantir que os sinais fluam de maneira estável durante a propagação direta e o backpropagation, melhorando a eficácia do treinamento em redes profundas.

5.3. Conexão com Fine-Tuning

Outro conceito importante nas redes neurais modernas é o fine-tuning, ou ajuste fino dos pesos. Ele se refere à prática de utilizar redes pré-treinadas em grandes conjuntos de dados e ajustar seus pesos em uma nova tarefa com dados específicos. Nesse contexto, a qualidade da inicialização herdada torna-se ainda mais relevante: pesos pré-treinados já estão em regiões do espaço de otimização com boas propriedades, e o fine-tuning atua como uma descida do gradiente restrita e mais eficiente.

Assim, o aprendizado supervisionado adicional funciona como um ajuste local sobre uma base sólida, acelerando a convergência e melhorando a generalização.

5.4. Aplicações das Lições da Regressão Linear

Os experimentos realizados com regressão linear, principalmente os que envolvem a variação da taxa de aprendizado e a inicialização dos parâmetros, ilustram, de forma controlada e visual, os mesmos princípios que regem o comportamento de redes profundas:

- A **taxa de aprendizado** continua sendo um hiperparâmetro crítico, e seu ajuste inadequado pode comprometer o treinamento.

- A **escolha da inicialização** afeta diretamente a dinâmica do gradiente, mesmo em funções de custo simples.

- A **visualização da trajetória do gradiente** no espaço dos parâmetros, embora impraticável em redes reais, é conceitualmente útil para entender a importância da geometria da função de custo.

Portanto, o estudo da regressão linear fornece uma base teórica e prática para compreender problemas mais complexos em aprendizado de máquina. Princípios como boa inicialização, taxa de aprendizado adequada e observação da convergência do custo se

mantêm essenciais à medida que se avança para arquiteturas mais sofisticadas, como redes neurais profundas.

6. Conclusão

Este trabalho teve como objetivo explorar os fundamentos da regressão linear sob a ótica do treinamento supervisionado com descida do gradiente, analisando de forma detalhada os efeitos da taxa de aprendizado e da inicialização dos parâmetros na convergência da função de custo. Através da implementação de funções fundamentais e da geração de representações gráficas, foi possível compreender, de maneira prática, como diferentes configurações impactam diretamente a eficiência e a estabilidade do processo de otimização.

Os experimentos demonstraram que a escolha da taxa de aprendizado α deve ser cuidadosamente calibrada: valores muito baixos resultam em aprendizado lento, enquanto valores excessivamente altos podem provocar instabilidade ou até divergência. Além disso, verificou-se que diferentes inicializações de θ afetam significativamente a trajetória de convergência, ainda que, no caso da regressão linear, todas as trajetórias levem ao mesmo ponto devido à convexidade da função de custo.

As visualizações em gráficos 2D e 3D permitiram uma análise qualitativa do comportamento do algoritmo, tornando evidentes os padrões de descida, às regiões de mínima e a natureza determinística da convergência nesse contexto.

Além da análise prática, foi discutida a relevância desses princípios para o treinamento de redes neurais profundas, nas quais a superfície de erro é altamente não convexa. A boa inicialização de pesos, assim como o ajuste fino de taxas de aprendizado e a aplicação de técnicas como fine-tuning, são estratégias críticas em arquiteturas mais complexas, onde a descida do gradiente enfrenta desafios adicionais.

Conclui-se que, apesar de sua simplicidade, a regressão linear fornece uma base sólida para o entendimento dos princípios de otimização em aprendizado de máquina. Compreender o impacto de cada hiperparâmetro em um modelo básico é essencial para o desenvolvimento de modelos mais robustos e eficazes em tarefas reais, especialmente no contexto das redes neurais modernas.

7. Referências

CHIPMAN, John S. Linear restrictions, rank reduction, and biased estimation in linear regression. Linear Algebra and Its Applications, v. 289, p. 55–74, 1999. DOI: 10.1016/S0024-3795(98)10138-6.

KILIÇ, Selim. Doğrusal regresyon analizi. Journal of Mood Disorders, v. 3, n. 2, p. 90–92, 2013. DOI: 10.5455/jmood.20130624120840.

TRANMER, M.; MURPHY, J.; ELLIOT, M.; PAMPAKA, M. *Multiple Linear Regression* (2nd Edition). Cathie Marsh Institute Working Paper 2020-01. 2020. Disponível em:https://hummedia.manchester.ac.uk/institutes/cmist/archive-publications/working-papers/2020/2020-1-multiple-linear-regression.pdf. Acesso em: 24 abr. 2025.

NAGY, Gábor. Sector Based Linear Regression, a new robust method for the multiple linear regression. Acta Cybernetica, v. 23, p. 1017–1038, 2018. DOI: 10.14232/actacyb.23.4.2018.3.

SPÄTH, Helmuth. Mathematical Algorithms for Linear Regression. Boston: Academic Press, 1992. (Computer science and scientific computing). ISBN 9780126564600.

SU, Xiaogang; YAN, Xin; TSAI, Chih-Ling. Linear regression. WIREs Computational Statistics, v. 4, n. 3, p. 275–294, 2012. DOI: 10.1002/wics.1198.

YAO, W.; LI, L. A new regression model: modal linear regression. *Scandinavian Journal of Statistics*, v. 41, p. 656–671, 2014. DOI: 10.1111/sjos.12054.

YUAN, Z.; YANG, Y. Combining linear regression models: when and how? *Journal of the American Statistical Association*, v. 100, n. 472, p. 1202–1214, 2005. DOI: 10.1198/016214505000000088.

ZEILEIS, A. et al. *strucchange: an R package for testing for structural change in linear regression models*. Technische Universität Wien, 2002. Disponível em: http://cran.R-project.org/. Acesso em: 24 abr. 2025.