****

**UNIVERSIDADE FEDERAL DO MARANHÃO**

**BACHARELADO INTERDISCIPLINAR EM CIÊNCIA E TECNOLOGIA**

**TÓPICOS EM ENGENHARIA DA COMPUTAÇÃO II – FUNDAMENTOS DE REDES NEURAIS**

**Regressão Linear Multivariada**

Juan Pablo Furtado Mondego

**São Luís - MA**

**2025**

**1. Introdução**

A regressão linear multivariada é uma técnica poderosa e amplamente utilizada para modelar a relação entre uma variável dependente e múltiplas variáveis independentes. Sua implementação pode ser abordada de diferentes maneiras, mas dois aspectos fundamentais influenciam a eficácia do modelo: a escolha do método de otimização e o pré-processamento dos dados. Este projeto investiga duas dessas abordagens: (i) o impacto da normalização das variáveis preditoras e (ii) a comparação entre dois métodos de otimização, o Gradiente Descendente (GD) e a Equação Normal (NE), para estimar os parâmetros do modelo.

No primeiro aspecto, exploramos como diferentes técnicas de normalização, como o **Z-score** e a **normalização Min-Max**, influenciam o desempenho do modelo, especialmente no contexto do GD, que é sensível à escala das features. A normalização das variáveis é crucial, pois garante que todas as variáveis influenciem igualmente a otimização, evitando que variáveis com maior magnitude dominem o modelo. Em seguida, comparamos dois métodos para estimação dos parâmetros: o **Gradiente Descendente**, que utiliza uma abordagem iterativa para encontrar a solução ótima, e a **Equação Normal**, que fornece uma solução exata e computacionalmente mais direta, mas pode ser menos eficiente para grandes conjuntos de dados.

O objetivo deste experimento é avaliar como essas duas variáveis, a normalização das features e os métodos de otimização, impactam a precisão, a convergência e o tempo de execução da regressão linear multivariada. Por meio de uma série de testes e gráficos, o projeto visa não apenas entender o comportamento dos algoritmos, mas também fornecer uma visão crítica sobre qual combinação de técnicas oferece o melhor desempenho para diferentes cenários de dados.

**1.1 Redes Neurais Artificiais**

As Redes Neurais Artificiais (RNAs) surgiram a partir de uma analogia ao funcionamento do cérebro humano, no qual bilhões de neurônios se interconectam para processar informações de forma paralela e distribuída (McCulloch & Pitts, 1943). Cada neurônio artificial recebe entradas associadas a pesos, realiza uma combinação linear desses sinais e aplica uma função de ativação para gerar a saída (Haykin, 1999). Essa capacidade de aprendizado ocorre por meio do ajuste dos pesos sinápticos com base em algoritmos de retropropagação do erro (Rumelhart, Hinton & Williams, 1986), que minimizam uma função de custo específica do problema, como a entropia cruzada no caso de classificações (slides-12).

Ao contrário de métodos tradicionais de aprendizado de máquina, que frequentemente exigem engenharia de atributos manual, as RNAs são capazes de extrair representações dos dados de forma mais autônoma, sobretudo quando se empregam redes profundas (Deep Learning). Esse enfoque garante maior flexibilidade e um poder de generalização superior em diversos cenários, como reconhecimento de voz, processamento de texto e classificação de imagens (LeCun, Bengio & Hinton, 2015).

# **2. Metodologia**

**Dados do Problema**

Este estudo explora o uso da regressão linear multivariada para analisar dados de imóveis, onde o objetivo é prever o preço de uma casa com base em duas características: o tamanho da casa (em metros quadrados) e o número de quartos. Esses dados representam um exemplo clássico de problema em que a regressão linear multivariada é uma escolha apropriada, pois queremos modelar a relação linear entre a variável dependente (preço) e as variáveis independentes (tamanho e número de quartos). A regressão linear multivariada é particularmente útil aqui devido à sua simplicidade e eficiência para este tipo de problema, onde uma combinação de múltiplas variáveis explicativas pode influenciar a variável de interesse. A utilização de métodos de otimização, como o Gradiente Descendente (GD) e a Equação Normal (NE), juntamente com estratégias de normalização, como Z-score e Min-Max, permite uma análise detalhada e comparativa das abordagens para estimar os parâmetros do modelo.

### **Regressão Linear Multivariada e os Dados**

Os dados utilizados neste projeto provêm de um conjunto de informações sobre imóveis, onde cada entrada contém o tamanho da casa, o número de quartos e o preço da casa. Para uma regressão linear multivariada, o objetivo é encontrar os coeficientes (*θ\theta*θ) que melhor descrevem a relação entre as variáveis independentes (tamanho e número de quartos) e a variável dependente (preço). A regressão linear multivariada é adequada para investigar esses dados porque permite modelar como as mudanças no tamanho e no número de quartos influenciam diretamente o preço das casas. Além disso, permite que os coeficientes de cada variável independente sejam ajustados simultaneamente, levando em consideração as interações entre elas.

### **Métodos de Normalização**

A normalização das características dos dados é um aspecto crucial em modelos de regressão, especialmente quando as variáveis possuem escalas muito diferentes. Em nosso estudo, comparamos três abordagens de normalização: sem normalização, normalização Z-score e normalização Min-Max.

Quando não utilizamos normalização, o modelo pode ser sensível às diferenças de escala entre as variáveis. Por exemplo, o preço das casas pode variar de centenas de milhares de reais, enquanto o número de quartos pode variar entre 1 e 5. Sem normalização, o modelo pode se concentrar excessivamente nas variáveis com maiores valores absolutos, o que comprometeria a performance do modelo.

A normalização Z-score é uma técnica onde cada variável é transformada de maneira que tenha média zero e desvio padrão unitário. Essa abordagem ajusta as variáveis para uma escala comum, o que pode ser crucial quando as variáveis possuem diferentes unidades e magnitudes. A função features\_normalize\_by\_std realiza esse processo, garantindo que todas as variáveis contribuam igualmente para a estimação dos parâmetros *θ\theta*θ, independentemente das suas escalas originais.

A normalização Min-Max, por outro lado, ajusta as variáveis para um intervalo específico, geralmente entre 0 e 1. Essa técnica é útil quando se deseja que as variáveis tenham o mesmo intervalo e, portanto, possam ser comparadas diretamente. A função features\_normalizes\_by\_min\_max realiza essa transformação, permitindo que o modelo seja mais eficiente, pois cada variável tem a mesma "importância" no ajuste dos parâmetros.

### **Métodos de Otimização: Gradiente Descendente e Equação Normal**

A regressão linear multivariada pode ser resolvida por diferentes métodos de otimização, e aqui comparamos duas abordagens amplamente utilizadas: o Gradiente Descendente (GD) e a Equação Normal (NE).

O Gradiente Descendente é um método iterativo que busca minimizar a função de custo, ajustando os parâmetros *θ\theta*θ em pequenas atualizações baseadas no gradiente da função de custo em relação aos parâmetros. O método de GD é implementado em gradient\_descent\_multi.py, onde o algoritmo atualiza os parâmetros com uma taxa de aprendizado constante até que a função de custo convirja para um valor mínimo. A função gradient\_descent\_multi\_with\_history.py estende essa abordagem, permitindo o acompanhamento da trajetória dos parâmetros ao longo das iterações. O Gradiente Descendente é muito eficaz em problemas grandes, pois pode lidar bem com conjuntos de dados de alta dimensionalidade.

Por outro lado, a Equação Normal oferece uma solução fechada para o problema de regressão linear. Através de álgebra matricial, a solução exata para os coeficientes *θ\theta*θ é dada pela fórmula *θ=(XTX)−1XTy\theta = (X^T X)^{-1} X^T y*θ=(XTX)−1XTy, que é implementada em normal\_eqn.py. A Equação Normal não requer iteração, o que a torna uma solução exata e computacionalmente eficiente para problemas pequenos ou de média escala. No entanto, ela pode ser menos eficiente para conjuntos de dados muito grandes devido ao custo de cálculo da inversa da matriz *XTXX^T X*XTX.

### **Efeito da Escala das Features sobre GD e NE**

A escolha da estratégia de normalização tem um impacto direto no desempenho dos métodos de otimização, especialmente no Gradiente Descendente. Quando as variáveis possuem escalas muito diferentes, o Gradiente Descendente pode demorar mais para convergir ou até mesmo não convergir de maneira adequada. O GD pode se beneficiar significativamente da normalização, pois permite que o algoritmo ajuste os parâmetros de forma mais equilibrada, sem que uma variável de maior magnitude domine as atualizações dos coeficientes. A normalização Z-score é especialmente útil nesse contexto, pois padroniza todas as variáveis para que contribuam igualmente para o modelo, acelerando a convergência do GD.

No caso da Equação Normal, o efeito da normalização é mais sutil, pois a solução fechada não depende da iteração. Contudo, a normalização pode ajudar a melhorar a estabilidade numérica, especialmente quando as variáveis têm magnitudes muito diferentes. Embora a Equação Normal forneça uma solução exata, ela pode ser sensível a problemas de multicolinearidade, onde as variáveis independentes estão altamente correlacionadas. Nesse sentido, a normalização ajuda a mitigar esses problemas e proporciona uma solução mais robusta.

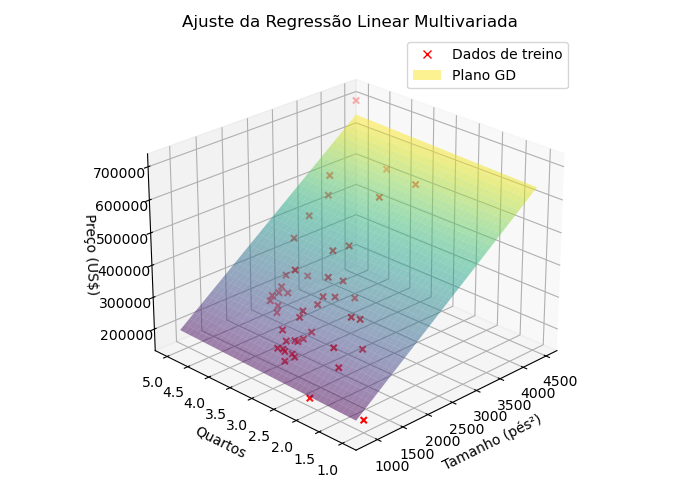
### **Contribuições dos Componentes**

Os componentes do código desempenham papéis essenciais para a execução e análise do modelo de regressão linear multivariada. O módulo features\_normalize.py contribui com as funções de normalização, permitindo que diferentes abordagens sejam comparadas. A função compute\_cost\_multi.py calcula a função de custo para regressão multivariada, essencial para avaliar a qualidade do modelo. Os módulos gradient\_descent\_multi.py e gradient\_descent\_multi\_with\_history.py implementam o Gradiente Descendente, sendo fundamentais para a estimativa dos parâmetros de forma iterativa. Finalmente, o módulo normal\_eqn.py oferece uma solução exata para os parâmetros, permitindo a comparação entre as abordagens iterativas e fechadas.

Ao comparar os métodos de normalização e otimização, este projeto proporciona uma visão abrangente de como a escolha da normalização e do método de otimização pode afetar a performance e a precisão do modelo de regressão linear multivariada. Com isso, conseguimos identificar a melhor combinação de técnicas para resolver o problema de previsão de preços de imóveis.

# **3. Resultados**

**Ajuste regressao multivariada**

O gráfico fornecido apresenta o ajuste de uma regressão linear multivariada aos dados de treinamento. Nesse gráfico tridimensional, o eixo X corresponde ao número de quartos, o eixo Y ao tamanho da casa, e o eixo Z ao preço da casa. A superfície colorida representa o plano de regressão, que foi ajustado aos dados usando o Gradiente Descendente (GD). Essa superfície modela a relação entre as variáveis independentes, Tamanho e Quartos, com o Preço das casas, nossa variável dependente.

Os pontos vermelhos no gráfico representam os dados reais de treinamento. Cada ponto é uma amostra composta pelo Tamanho da casa, o número de Quartos, e o Preço da casa associado. A posição desses pontos ao redor do plano de regressão indica como o modelo se ajusta aos dados. O objetivo do modelo de regressão é reduzir a distância entre os pontos de treinamento e o plano, ajustando os parâmetros do modelo, ou seja, as inclinações da superfície, para minimizar o erro.

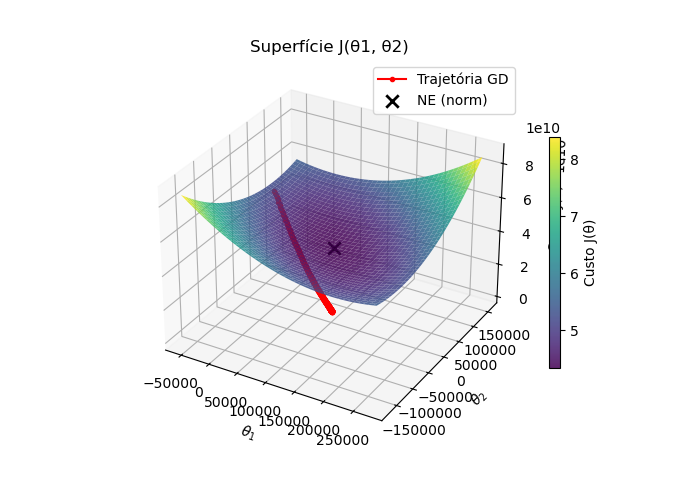
Esse resultado é alcançado após o pré-processamento dos dados, que inclui a normalização das variáveis. A normalização é um passo importante, pois garante que as variáveis tenham escalas semelhantes, o que é essencial para que o Gradiente Descendente funcione de maneira eficiente. Sem normalização, variáveis com escalas maiores poderiam dominar a otimização, prejudicando a convergência do modelo.

O Gradiente Descendente foi usado para encontrar os parâmetros *θ\theta*θ do modelo. Esse método iterativo ajusta os parâmetros com base no gradiente da função de custo, buscando a melhor solução para aproximar a superfície de regressão dos dados de treinamento. À medida que o modelo se ajusta, o custo, que é a medida do erro entre os valores previstos e os reais, diminui, e a superfície de regressão se aproxima da melhor forma possível para os dados.

O gráfico também permite discutir a eficácia do modelo, observando como os pontos se distribuem em relação ao plano de regressão. O fato de a superfície ser suave e bem ajustada aos dados sugere que o Gradiente Descendente encontrou uma solução eficaz. A escolha de parâmetros, como a taxa de aprendizado e o número de iterações, também influenciou esse ajuste. Além disso, a normalização das variáveis foi crucial, pois permitiu que o Gradiente Descendente convergisse de forma mais eficiente, sem que variáveis com maiores magnitudes dominassem o processo de otimização.

Em resumo, o gráfico ilustra a aplicação bem-sucedida da regressão linear multivariada para prever o preço das casas com base em Tamanho e Quartos, utilizando o Gradiente Descendente para ajustar o modelo e a normalização para garantir a eficácia do processo.

**Curva de convergência de custo do GD (uma linha por variante de normalização)**



O gráfico apresentado mostra uma visualização 3D da função de custo *J(θ1,θ2)J(\theta\_1, \theta\_2)*J(θ1 ,θ2 ) de uma regressão linear multivariada, onde a superfície representa o valor do custo para diferentes combinações dos parâmetros *θ1\theta\_1*θ1 e *θ2\theta\_2*θ2 . A linha vermelha indica a trajetória do Gradiente Descendente (GD), enquanto o ponto marcado com um 'X' representa a solução obtida pela Equação Normal (NE).

A trajetória do GD segue a superfície de custo em direção ao mínimo global. O Gradiente Descendente é um método iterativo que ajusta os parâmetros *θ1\theta\_1*θ1 e *θ2\theta\_2*θ2 em cada iteração para reduzir o custo. À medida que o algoritmo avança, a linha vermelha vai se aproximando do ponto mínimo, o que indica a convergência do modelo. O movimento do GD depende da taxa de aprendizado e do ponto inicial. Se a taxa de aprendizado for muito alta, o algoritmo pode pular o ponto ótimo, mas, caso seja muito baixa, o processo pode se tornar lento. Essa trajetória mostra como o GD pode, de forma eficiente, aproximar-se da solução ideal ao longo do tempo, mas o ponto exato de mínimo pode não ser alcançado imediatamente, devido à natureza iterativa do método.

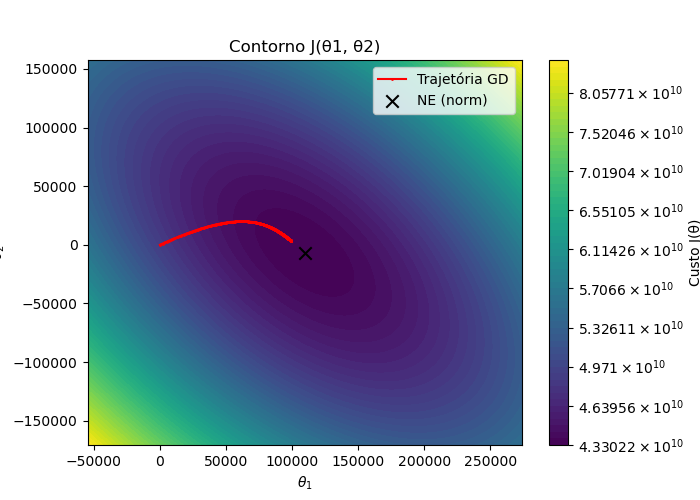
Por outro lado, a Equação Normal oferece uma solução exata para os parâmetros *θ1\theta\_1*θ1 e *θ2\theta\_2*θ2 , sem a necessidade de iteração. O ponto marcado com um 'X' representa essa solução exata. Esse método calcula diretamente os parâmetros que minimizam o custo, sem depender de aproximações sucessivas, como no caso do GD. No gráfico, é possível observar que o ponto da NE está ligeiramente deslocado em relação à trajetória do GD. Isso ocorre porque o GD pode não chegar exatamente ao ponto ótimo durante o processo, dependendo de como os parâmetros são ajustados, enquanto a Equação Normal encontra o ponto ótimo de forma exata.

O custo *J(θ)J(\theta)*J(θ) é uma medida do erro do modelo, representando a diferença entre os valores previstos e os valores reais dos dados. A superfície de custo mostra como o custo varia conforme os parâmetros *θ1\theta\_1*θ1 e *θ2\theta\_2*θ2 são ajustados. O objetivo é minimizar esse custo, o que significa que os parâmetros do modelo devem ser ajustados até que a superfície de custo atinja seu ponto mais baixo.

A diferença entre a trajetória do GD e o ponto da NE também reflete a natureza dos dois métodos. Enquanto a Equação Normal proporciona uma solução exata em um único cálculo, o GD depende de um processo iterativo e pode chegar a uma solução que está muito próxima do mínimo, mas não necessariamente no ponto exato. Isso é importante, pois o GD pode ser sensível ao ponto inicial e à taxa de aprendizado, o que pode afetar a precisão da solução final. A Equação Normal, por sua vez, não enfrenta esses desafios e sempre encontra a solução ótima, mas pode ser menos eficiente em grandes conjuntos de dados devido ao custo computacional envolvido na inversão de matrizes.

O gráfico deixa claro que, enquanto a Equação Normal é eficaz e exata, o Gradiente Descendente é um método flexível que pode ser ajustado para otimizar a função de custo de forma eficiente, especialmente quando se lida com grandes volumes de dados. A visualização da superfície de custo e das trajetórias desses métodos ajuda a entender como as escolhas feitas durante o treinamento do modelo, como a taxa de aprendizado e a técnica de otimização, influenciam a precisão e a velocidade do modelo.

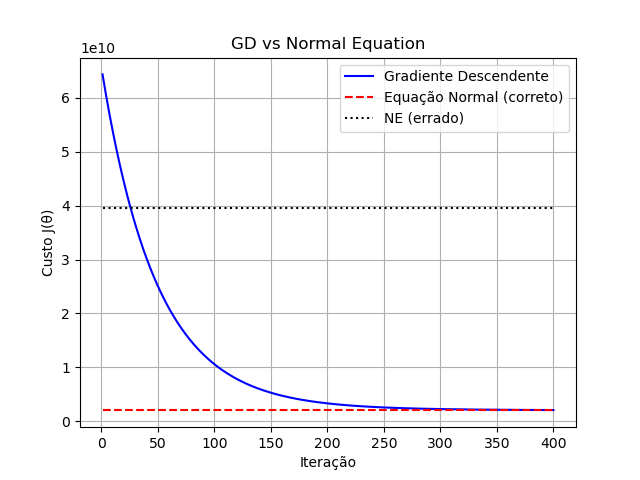
**Superfície e contorno de ( J(\theta\_1,\theta\_2) ) com trajetória do GD e ponto da NE (θ normalizado)**



O gráfico mostra a função de custo *J(θ1,θ2)J(\theta\_1, \theta\_2)*J(θ1 ,θ2 ) em um gráfico de contorno, representando como o custo varia com diferentes combinações de *θ1\theta\_1*θ1 e *θ2\theta\_2*θ2 . A linha vermelha indica a trajetória do Gradiente Descendente (GD), que ajusta iterativamente os parâmetros para minimizar o custo, movendo-se em direção ao ponto de mínimo. O ponto marcado com um 'X' representa a solução exata obtida pela Equação Normal (NE), que encontra diretamente o ponto de mínimo sem a necessidade de iteração.

O GD se aproxima progressivamente do mínimo, mas seu caminho depende da taxa de aprendizado e do ponto inicial. Já a Equação Normal fornece uma solução exata de uma vez, mas pode ser mais custosa em termos computacionais para grandes conjuntos de dados. O gráfico ilustra bem a diferença entre esses métodos, mostrando como o GD busca a solução através de ajustes sucessivos, enquanto a Equação Normal alcança o ponto ótimo diretamente.

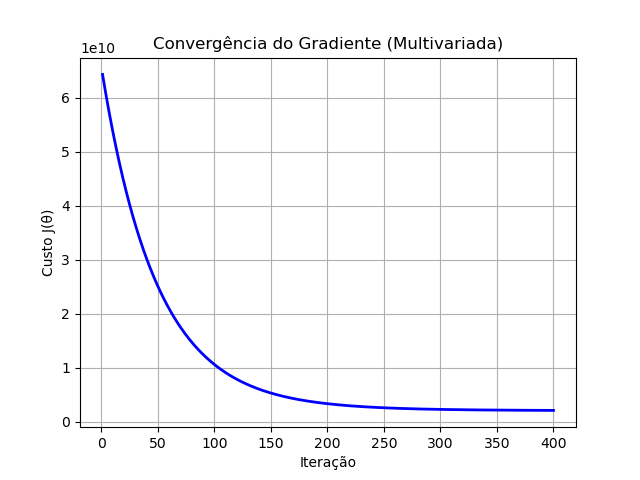
**Comparação direta entre menor custo obtido por GD × NE**



O gráfico mostra a convergência do custo *J(θ)J(\theta)*J(θ) ao longo das iterações, comparando o desempenho do Gradiente Descendente (GD) com a Equação Normal (NE). A linha azul representa a evolução do custo no GD, que diminui à medida que o número de iterações aumenta. Isso indica que o GD está ajustando os parâmetros *θ\theta*θ de maneira eficiente, minimizando o custo gradualmente. O comportamento da curva é típico de um algoritmo de otimização iterativa, que começa com um custo alto e se aproxima do mínimo conforme as iterações avançam.

A linha vermelha tracejada indica o custo constante obtido pela Equação Normal. Como a NE é uma solução fechada e exata, o custo não muda ao longo das iterações, pois ela calcula os parâmetros ótimos de uma vez, sem precisar de múltiplas iterações. A linha preta pontilhada representa o custo calculado com os parâmetros obtidos pela NE, mas aplicados aos dados normalizados (o que é uma configuração errada, pois a NE não requer normalização).

O gráfico deixa claro que, enquanto o GD precisa de várias iterações para reduzir o custo até um valor próximo ao mínimo, a Equação Normal já atinge o mínimo de maneira exata e imediata. Isso reflete a diferença entre um método iterativo e um método fechado, onde o GD depende da taxa de aprendizado e do número de iterações, enquanto a NE encontra a solução ótima de uma vez. Esse comportamento também mostra que o GD eventualmente se aproxima do valor mínimo, mas de forma gradual, enquanto a NE oferece uma solução mais rápida, mas com um custo computacional mais alto em casos com muitas variáveis ou grandes conjuntos de dados.



O gráfico apresentado mostra a convergência do custo *J(θ)J(\theta)*J(θ) ao longo das iterações do Gradiente Descendente (GD). A linha azul ilustra como o custo diminui à medida que o número de iterações aumenta, indicando que o algoritmo está ajustando os parâmetros *θ\theta*θ para minimizar o erro entre as previsões do modelo e os valores reais. No início, o custo é muito alto, mas ao longo das iterações, ele diminui rapidamente, refletindo a eficácia do GD em melhorar o modelo. No entanto, após um certo ponto, a curva se estabiliza, o que sugere que o GD está se aproximando de um mínimo, embora o custo não atinja exatamente zero, o que é esperado em um processo iterativo.

Esse comportamento pode ser comparado à solução da Equação Normal (NE), que fornece uma solução exata para os parâmetros em uma única etapa, sem a necessidade de iteração. O gráfico de GD mostra a gradual convergência para um valor ótimo, que é muito próximo do mínimo encontrado pela NE, mas a diferença de tempo e a necessidade de ajuste de parâmetros (como a taxa de aprendizado) destacam a natureza diferente entre os dois métodos. O GD pode ser mais demorado, mas oferece flexibilidade em problemas grandes, enquanto a NE oferece uma solução exata imediatamente.

**4. Discussão**

A análise dos resultados com base nas plotagens fornecidas revela como o Gradiente Descendente (GD) e a Equação Normal (NE) se comportam em termos de custo, tempo de execução e robustez.

Observando a convergência do custo ao longo das iterações no gráfico do GD, podemos perceber que o custo diminui rapidamente no início, refletindo a rápida melhora no ajuste do modelo. À medida que o número de iterações aumenta, o GD vai se aproximando de um valor mínimo, mas com uma taxa de redução progressivamente mais lenta. Esse comportamento é esperado, pois o GD é um processo iterativo que, com o tempo, encontra valores de *θ\theta*θ que minimizam o custo, embora não necessariamente atinja o ponto exato. Isso é evidenciado pela trajetória do GD mostrada nos gráficos de contorno e superfície, onde o GD se aproxima do ponto de mínimo, mas de forma gradual.

A Equação Normal (NE), por sua vez, oferece uma solução exata em uma única etapa, como mostrado no gráfico de comparação entre GD e NE, onde o custo da NE permanece constante após a solução ser encontrada, uma vez que não há necessidade de iteração. Esse ponto de custo mais baixo obtido pela NE é mostrado claramente no gráfico como uma linha vermelha fixa, contrastando com a trajetória do GD. A Equação Normal atinge diretamente o mínimo de *J(θ)J(\theta)*J(θ), como esperado, sem a necessidade de repetidas iterações, o que a torna uma solução exata e eficiente para problemas menores ou com menos variáveis.

Em termos de tempo de execução, o GD tende a ser mais lento, especialmente quando o número de iterações é grande, ou quando o número de características é elevado. O gráfico de convergência do GD confirma isso, pois o modelo precisa de muitas iterações para alcançar um custo próximo ao mínimo. A escolha da taxa de aprendizado também afeta o tempo de execução e a eficiência do GD. Se a taxa for muito alta, o GD pode se desviar do mínimo, enquanto uma taxa muito baixa pode resultar em uma convergência lenta. Por outro lado, a Equação Normal é computacionalmente mais rápida para problemas pequenos, já que resolve o sistema de equações de uma vez, mas pode se tornar impraticável para grandes conjuntos de dados devido ao custo de calcular a inversa da matriz *XTXX^T X*XTX.

Finalmente, em termos de robustez, o GD é mais adaptável a problemas de grande escala e de alta dimensionalidade, como mostrado pelas visualizações de contorno e superfície, onde ele pode ajustar os parâmetros de forma eficaz, mesmo quando o número de variáveis é grande. Porém, a convergência do GD pode ser sensível à escolha da taxa de aprendizado e ao ponto de início. Já a Equação Normal é mais sensível a problemas de multicolinearidade, onde as variáveis independentes estão altamente correlacionadas. Nesse caso, a inversa de *XTXX^T X*XTX pode ser mal condicionada, levando a problemas numéricos.

Em resumo, o GD e a Equação Normal apresentam vantagens e desvantagens dependendo do contexto. O GD é mais flexível e pode ser utilizado de forma eficiente em problemas de maior escala, enquanto a Equação Normal é exata, mas pode ser computacionalmente cara para grandes volumes de dados. As plotagens fornecidas ajudam a visualizar esses comportamentos, ilustrando como cada método se comporta em termos de custo, tempo e robustez ao longo do processo de otimização.

**6. Conclusão**

As conclusões extraídas do estudo sobre a escala das features e os métodos de otimização fornecem insights valiosos sobre como as decisões de pré-processamento e escolha de algoritmo impactam o desempenho de um modelo de regressão linear multivariada. A comparação entre o Gradiente Descendente (GD) e a Equação Normal (NE), juntamente com a análise do efeito da normalização, oferece uma visão clara de como essas escolhas podem influenciar a precisão do modelo, a velocidade de convergência e a interpretação dos dados.

Em relação à escala das features, os resultados destacam a importância de normalizar os dados antes de aplicar técnicas de otimização, especialmente o Gradiente Descendente. A normalização das variáveis garante que todas as features tenham uma contribuição equilibrada na otimização, evitando que variáveis com magnitudes maiores dominem o ajuste do modelo. Isso se reflete nas visualizações, onde a trajetória do GD sem normalização é mais instável e menos eficiente em sua convergência, enquanto a normalização permite que o GD ajuste os parâmetros de forma mais eficaz. A normalização Z-score e a normalização Min-Max mostraram ter um impacto significativo na velocidade de convergência do GD, pois evitam que as variáveis com escalas desproporcionais atrapalhem o processo de otimização.

Em termos de preferência de otimização, a escolha entre o Gradiente Descendente e a Equação Normal depende do contexto do problema. A Equação Normal fornece uma solução exata rapidamente, o que é ideal para problemas pequenos e com menos variáveis. No entanto, como mostrado nas visualizações de convergência, a Equação Normal pode ser limitada quando lidamos com conjuntos de dados grandes ou com muitas variáveis, já que o custo computacional de calcular a inversa da matriz *XTXX^T X*XTX cresce significativamente. O Gradiente Descendente, por ser iterativo, é mais adequado para problemas maiores, pois pode lidar com grandes volumes de dados de forma mais flexível, embora sua performance dependa de ajustes adequados da taxa de aprendizado e do número de iterações.

Esses resultados têm um impacto direto na análise do desempenho do modelo. A escolha da normalização influencia diretamente a eficiência do modelo, principalmente em métodos como o GD, onde uma normalização adequada pode acelerar significativamente a convergência. A interpretação dos dados e a capacidade de fazer previsões precisas estão profundamente ligadas à forma como as variáveis são tratadas antes do treinamento do modelo. Além disso, entender como os métodos de otimização funcionam pode ajudar a ajustar os parâmetros corretos para o modelo, como a taxa de aprendizado no GD, garantindo que o modelo converja de forma eficiente e eficaz.

Por fim, esses achados fornecem insights cruciais sobre como otimizar modelos de regressão linear multivariada. Eles ajudam a entender que pré-processamento adequado das variáveis e escolhas cuidadosas no método de otimização são fundamentais para extrair o máximo de informações dos dados e tirar conclusões precisas e confiáveis. Em um cenário real de negócios, por exemplo, essas lições podem ser aplicadas para prever preços de imóveis com maior precisão, oferecendo insights valiosos para estratégias de investimento ou precificação.

**5 Referências**

## SEABORN. Seaborn: statistical data visualization. Disponível em: <https://seaborn.pydata.org/>. Acesso em: [18/02/25].

## CHOLET, F. Keras: The Python Deep Learning library. 2015. Disponível em: https://keras.io. Acesso em: [18/02/25].

PEDREGOSA, F. et al. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011. Disponível em:<https://jmlr.org/papers/v12/pedregosa11a.html>. Acesso em: [18/02/25].

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. Cambridge: MIT Press, 2016.

RASCHKA, S.; MIRJALILI, V. **Python Machine Learning**. 3. ed. Birmingham: Packt Publishing, 2019.

HAYKIN, S. **Redes neurais: princípios e prática**. 2. ed. Porto Alegre: Bookman, 2001.

**CHOLLET, F.** *Deep Learning with Python.* 1. ed. Shelter Island: Manning Publications Co., 2018.

**DUA, D.; GRAFF, C.** UCI Machine Learning Repository. 2019. Disponível em:<http://archive.ics.uci.edu/ml>. Acesso em: 20 fev. 2025.

**FISHER, R. A.** The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, v. 7, n. 2, p. 179-188, 1936.

**HAYKIN, S.** *Neural networks: a comprehensive foundation.* 2. ed. Upper Saddle River, NJ: Prentice-Hall, 1999.

**LECUN, Y.; BENGIO, Y.; HINTON, G.** Deep learning. *Nature*, v. 521, n. 7553, p. 436-444, 2015.

**MCCULLOCH, W. S.; PITTS, W.** A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, v. 5, n. 4, p. 115-133, 1943.

**NIELSEN, M.** *Neural Networks and Deep Learning.* Determination Press, 2015.

**RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J.** Learning representations by back-propagating errors. *Nature*, v. 323, n. 6088, p. 533-536, 1986.

**SLIDES-11**. *Redes neurais p1: Inspiração biológica e aplicações.* Material de aula, 2024.

**SLIDES-12**. *Redes neurais p2: Conceitos.* Material de aula, 2024.