instituto superior de engenharia de lisboa

INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA
Mestrado em Matemática Aplicada para a Indústria
Métodos Matemáticos para Inteligência Artificial

Projeto: Método do Gradiente Descendente

Tiago Garcia - A47211
Inês Macedo - A50293
Ricardo Faria - A50834

Docente: Professor Luís Silva
Ano letivo: 2025/2026

1 Introdução

O objetivo deste projeto é estudar e comparar diferentes variantes do método do gradiente aplicadas à regressão linear com regularização L2 (regressão Ridge), utilizando o *dataset California Housing*. São analisadas três abordagens distintas:

- **Batch Gradient Descent** (BGD), que utiliza todo o conjunto de treino em cada atualização dos parâmetros;
- **Stochastic Gradient Descent** (SGD), que atualiza os parâmetros utilizando apenas um exemplo de cada vez;
- **Mini-Batch Gradient Descent**, que realiza atualizações com pequenos subconjuntos fixos (*mini-batches*) do treino.

O estudo experimental analisa o desempenho das três abordagens, a influência da taxa de aprendizagem η, o impacto do tamanho do *batch*, o efeito da regularização λ, a importância da normalização dos dados e a identificação do método mais estável e eficiente.

Pretende-se, assim, compreender o comportamento de cada método, identificar vantagens e limitações e determinar qual apresenta o melhor compromisso entre estabilidade, rapidez de convergência e desempenho final.

2 Conjunto de Dados e Pré-processamento

Foi utilizado o *dataset California Housing*, disponível na biblioteca `scikit-learn`, composto por 20 640 registos de zonas residenciais da Califórnia. Cada exemplo inclui 8 atributos numéricos: rendimento mediano (`MedInc`), idade mediana das habitações (`HouseAge`), número médio de divisões e quartos por habitação (`AveRooms`, `AveBedrms`), população (`Population`), número médio de ocupantes (`AveOccup`) e localização geográfica (`Latitude`, `Longitude`).

A variável alvo (resposta) corresponde ao **valor mediano das habitações**, expresso em centenas de milhares de dólares, apresentando valores bastante diferentes entre as diversas zonas geográficas.

Para reduzir o tempo de execução e facilitar a simulação com vários hiperparâmetros, foi extraída uma amostra aleatória de 1000 exemplos. Esta amostra foi dividida em três subconjuntos disjuntos:

- **treino**: 700 exemplos;
- **validação**: 150 exemplos;
- **teste**: 150 exemplos.

Antes do treino aplicou-se a normalização *Z-score* a todos os atributos de entrada. Para cada atributo x_j foi utilizada a transformação:

$$\tilde{x}_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j},$$

onde μ_j e σ_j correspondem, respetivamente, à média e ao desvio-padrão calculados apenas no conjunto de treino.

A normalização é necessária porque os atributos originais podem estar em escalas muito diferentes (por exemplo, rendimento mediano e latitude têm ordens de grandeza distintas).

Ao colocar todos os atributos numa escala comparável, média próxima de zero e desvio-padrão perto de um, evita-se que certos atributos dominem o gradiente e facilita-se a escolha da taxa de aprendizagem, resultando numa convergência mais estável e eficiente dos métodos de otimização.

3 Modelo e Função de Custo

O modelo considerado neste projeto é o de regressão linear com termo de viés. Dado um exemplo de entrada $\overline{X}_i \in \mathbb{R}^d$, a previsão do modelo é dada por:

$$\hat{y}_i = \overline{W}^\top \overline{X}_i + b,$$

onde $\overline{W} \in \mathbb{R}^d$ é o vetor de pesos do modelo, valores que determinam a influência de cada atributo na previsão, e $b \in \mathbb{R}$ é o termo de viés, uma constante que desloca a função de regressão verticalmente, permitindo um ajuste mais adequado aos dados.

Para ajustar o modelo utilizou-se a função de custo baseada no erro quadrático médio com regularização L2. Num conjunto de treino com n exemplos, a função de custo é definida por:

$$J(\overline{W}, b) = \frac{1}{n} \sum_{i=1}^{n} \Big((\overline{W}^\top \overline{X}_i + b) - y_i \Big)^2 \; + \; \lambda \, \|\overline{W}\|_2^2,$$

onde y_i é o valor real do exemplo i e $\lambda \geq 0$ é o parâmetro de regularização.

O primeiro termo corresponde ao **erro quadrático médio (MSE)**, que mede o erro médio entre as previsões do modelo e os valores reais. O segundo termo corresponde à **regularização L2**, que penaliza pesos demasiado elevados, evitando sobreajuste e tornando o modelo mais estável.

Como esta função de custo é contínua e diferenciável, pode ser minimizada de forma eficiente através das variantes do método do gradiente analisadas nas secções seguintes.

4 Métodos de Otimização: Variantes do Gradiente

Foram implementadas três variantes do método do gradiente para minimizar a função de custo definida na secção anterior. Todas seguem o mesmo princípio: ajustar iterativamente os parâmetros $\overline{W}$ e b na direção oposta ao gradiente do custo. A principal diferença entre os métodos está no número de exemplos utilizados em cada atualização, o que afeta a estabilidade da convergência e o custo computacional.

4.1 Batch Gradient Descent (BGD)

Neste método, o gradiente é calculado utilizando **todos** os exemplos do conjunto de treino. Assim, cada atualização tem o custo computacional de uma passagem completa pelos dados. A atualização tem a forma:

$$\overline{W} \leftarrow \overline{W} - \eta \, \nabla_{\overline{W}} J(\overline{W}, b), \qquad b \leftarrow b - \eta \, \nabla_b J(\overline{W}, b).$$

Apesar de ser o mais estável e de produzir curvas de convergência suaves, o BGD é normalmente o mais lento, sobretudo para conjuntos de dados grandes.

4.2 Stochastic Gradient Descent (SGD)

No SGD, os parâmetros são atualizados exemplo a exemplo. Isto é, para cada par (x_i, y_i), calcula-se o gradiente relativo apenas a esse exemplo. Deste modo, o método é muito mais rápido por iteração, mas introduz ruído nas atualizações, resultando em trajetórias de convergência mais irregulares.

4.3 Mini-Batch Gradient Descent

Este método representa um compromisso entre os dois anteriores. Os dados de treino são divididos em *mini-batches* de tamanho fixo (por exemplo, 16, 32, 64 ou 128), e cada atualização utiliza apenas os exemplos de um desses blocos. O Mini-Batch tende a convergir mais rapidamente do que o BGD e com muito menos ruído do que o SGD, sendo geralmente o método mais eficiente em prática.

5 Resultados Experimentais

Nesta secção apresentam-se os resultados obtidos para as três variantes do gradiente - **BGD**, **SGD** e **Mini-Batch**. O objetivo principal é comparar a estabilidade, a velocidade de convergência e o desempenho final nos conjuntos de validação e teste.

Foram ainda analisados vários fatores que influenciam o comportamento dos métodos, nomeadamente a taxa de aprendizagem, o tamanho do *batch*, o valor da regularização L2 e o impacto da normalização dos dados.

5.1 Comparação entre BGD, SGD e Mini-Batch

Os três métodos foram treinados durante 1000 épocas com os seguintes hiperparâmetros base: $\eta = 0.01$ no BGD (taxas mais altas são adequadas porque o gradiente é calculado sobre todo o conjunto de treino), $\eta = 0.001$ no SGD e no Mini-Batch (valores mais pequenos reduzem o ruído inerente às atualizações), $\lambda = 0.01$ (regularização suficientemente moderada para controlar o sobreajuste sem degradar o ajuste ao treino) e $batch = 32$ (tamanho que oferece bom compromisso entre estabilidade e eficiência computacional). Estes valores foram definidos com base em testes preliminares, tendo demonstrado convergência estável em todos os métodos.

As curvas de convergência revelam comportamentos distintos:

- **BGD**: convergência estável e suave, mas atualizações lentas;
- **SGD**: atualizações rápidas mas ruidosas, com oscilações visíveis;
- **Mini-Batch**: combina rapidez e estabilidade, apresentando o melhor equilíbrio.

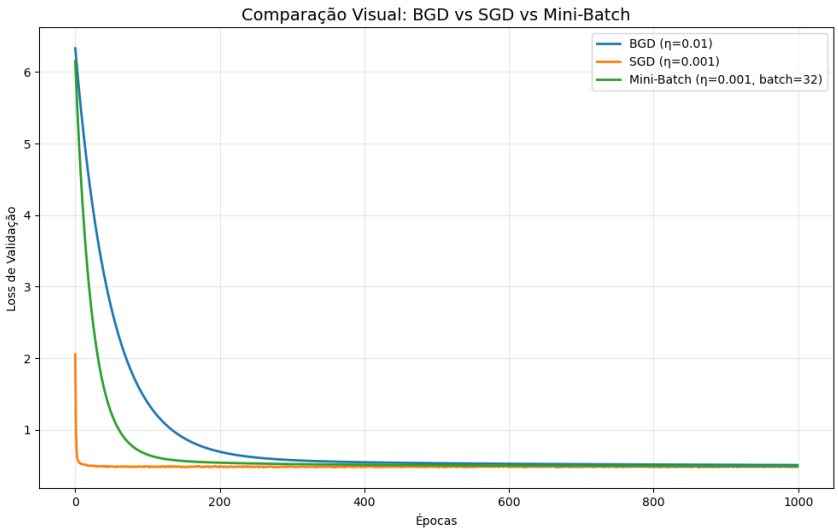

Figura 1: Comparação das curvas de MSE entre BGD, SGD e Mini-Batch.

Os valores finais de MSE no conjunto de teste foram:

Método	MSE no Teste
BGD	0.49972
SGD	0.49141
Mini-Batch	**0.49122**

O Mini-Batch obteve o melhor desempenho global, com uma curva mais estável do que o SGD e convergência mais rápida do que o BGD.

5.2 Efeito da Taxa de Aprendizagem

Foram testadas três taxas de aprendizagem no Mini-Batch: $\eta = 0.01$, $\eta = 0.001$ e $\eta = 0.0001$.

- $\eta = 0.01$: convergência rápida e estável;
- $\eta = 0.001$: convergência mais lenta mas consistente;
- $\eta = 0.0001$: convergência demasiado lenta.

A escolha destes valores permite comparar taxas de aprendizagem que conduzem a uma convergência eficiente (10^{-2} e 10^{-3}) com uma taxa demasiado pequena (10^{-4}), que torna o treino muito lento.

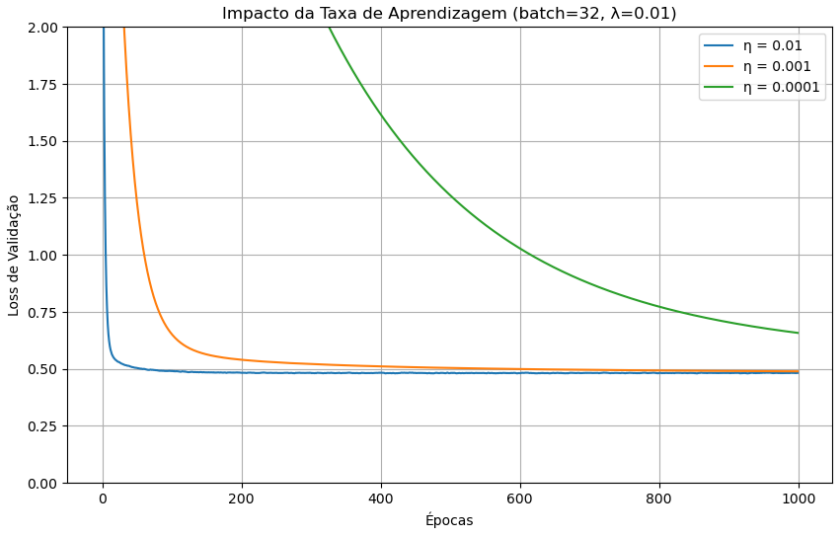

Figura 2: Impacto da taxa de aprendizagem na convergência.

5.3 Influência do Tamanho do *Batch*

Foram testados tamanhos de *batch* iguais a 16, 32, 64 e 128, mantendo $\eta = 0.001$ e $\lambda = 0.01$.

- **16**: convergência rápida;
- **32**: melhor compromisso entre estabilidade e rapidez;
- **64**: curva estável mas convergência ligeiramente mais lenta;
- **128**: comportamento próximo do BGD.

Estes valores foram escolhidos por serem típicos na literatura e permitirem observar claramente o compromisso entre velocidade de convergência e custo computacional.

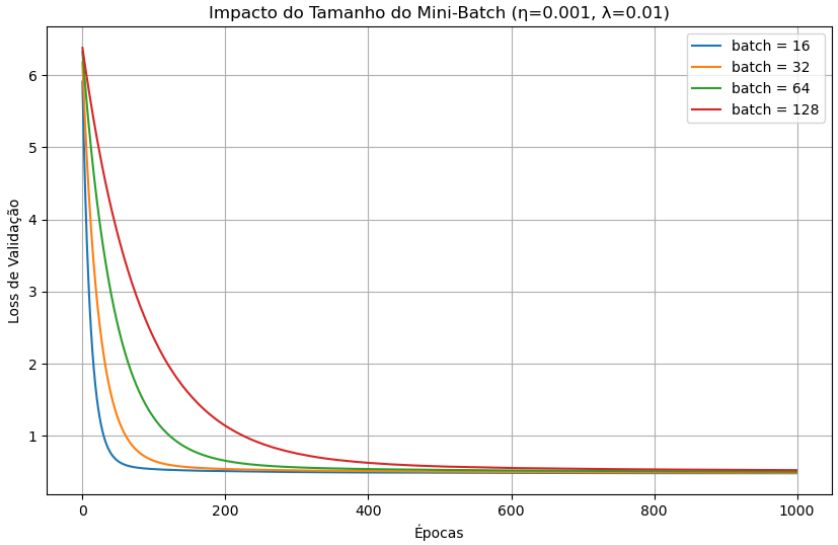

Figura 3: Efeito do tamanho do *batch* na convergência.

5.4 Regularização L2

Foram avaliados quatro valores de regularização: $\lambda = 0$, 0.01, 0.1, mantendo $\eta = 0.001$ e o tamanho do *batch* igual a 32. Estes valores permitem analisar desde a ausência de regularização até penalizações fortes.

- $\lambda = 0$: menor erro do conjunto de treino, mas risco de sobreajuste;
- $\lambda = 0.01$: melhor compromisso treino–validação;
- $\lambda = 0.1$: penalizações fortes, pior desempenho.

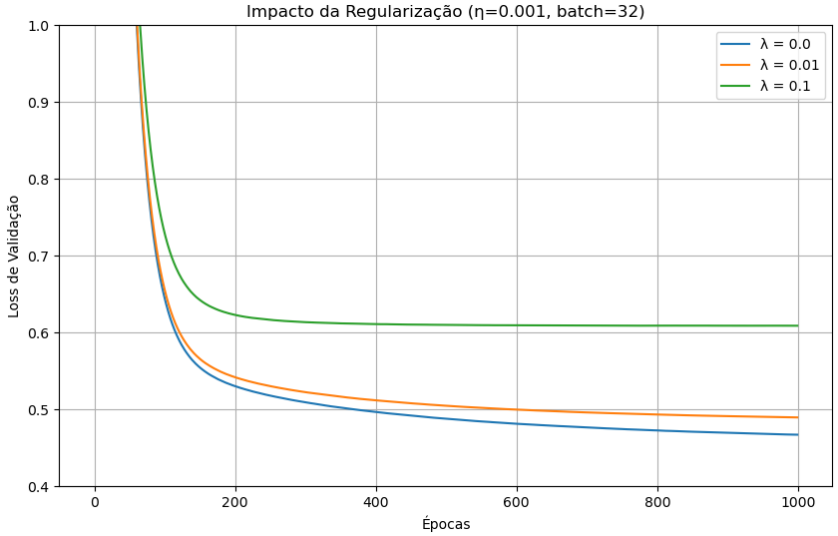

Figura 4: Impacto da regularização L2 na convergência.

5.5 Importância da Normalização

Treinar o modelo sem normalização obrigou a usar uma taxa de aprendizagem extremamente pequena ($\eta = 10^{-7}$), o que levou a uma convergência lenta e a erros mais elevados, ou seja, com mais ruído.

Com normalização *Z-score*, todos os atributos ficam aproximadamente na mesma escala, permitindo taxas maiores e convergência mais estável.

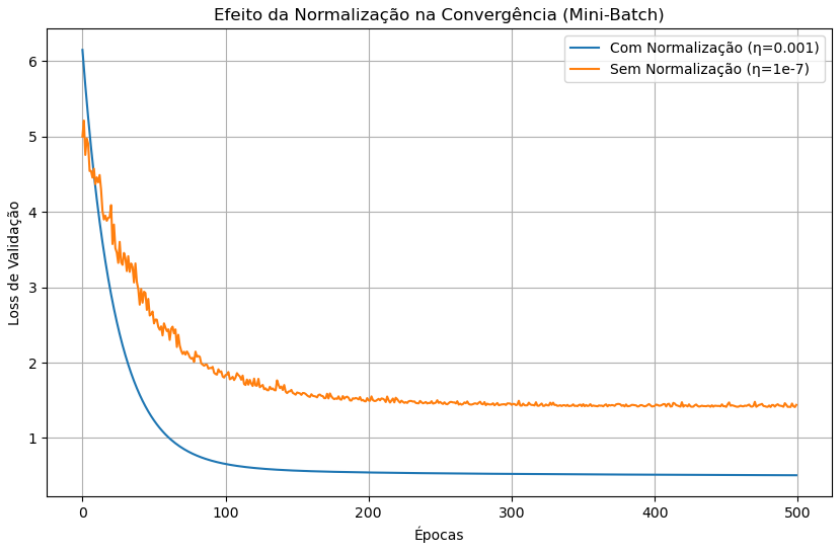

Figura 5: Comparação entre treino com e sem normalização.

6 Conclusão

O estudo realizado permitiu comparar de forma sistemática três variantes do método do gradiente aplicadas à regressão linear com regularização L2: Batch Gradient Descent, Stochastic Gradient Descent e Mini-Batch Gradient Descent.

Os resultados experimentais mostraram diferenças claras entre os métodos. O BGD apresentou o comportamento mais estável, com curvas de convergência suaves, mas revelou-se o mais lento devido ao custo de processamento de uma época completa a cada atualização. O SGD, por outro lado, permitiu atualizações muito rápidas, mas introduziu grande variabilidade no erro, originando convergência irregular. O Mini-Batch demonstrou oferecer o melhor compromisso, combinando a rapidez do SGD com a estabilidade do BGD, e obteve consistentemente o melhor desempenho no conjunto de teste.

Verificou-se também que a normalização dos dados é essencial para garantir uma convergência rápida e estável. Sem normalização, foi necessário utilizar taxas de aprendizagem extremamente pequenas, resultando em tempos de treino muito maiores e desempenhos inferiores.

Relativamente aos hiperparâmetros, observou-se que taxas de aprendizagem demasiado pequenas tornam o processo demasiado lento, enquanto valores excessivamente altos prejudicam a estabilidade. O tamanho do *batch* revelou igualmente impacto significativo, com o valor 32 a oferecer um equilíbrio sólido entre velocidade de convergência e eficiência computacional. A regularização L2 mostrou-se útil para controlar pesos demasiado grandes, sendo $\lambda = 0.01$ o valor que melhor equilibrava o erro de treino e validação.

Em suma, conclui-se que o Mini-Batch Gradient Descent, com normalização adequada e

hiperparâmetros bem ajustados, é o método mais eficiente e robusto para este problema, apresentando um desempenho superior tanto em estabilidade como em velocidade de convergência.