

# Parte 2

Bem vindos!  
Aprendizado Supervisionado

Não esqueça de gravar a aula!



O que você lembra da última aula?

pergunta motivadora para os alunos. a ideia aqui é puxar eles para aula para falar e engajar.

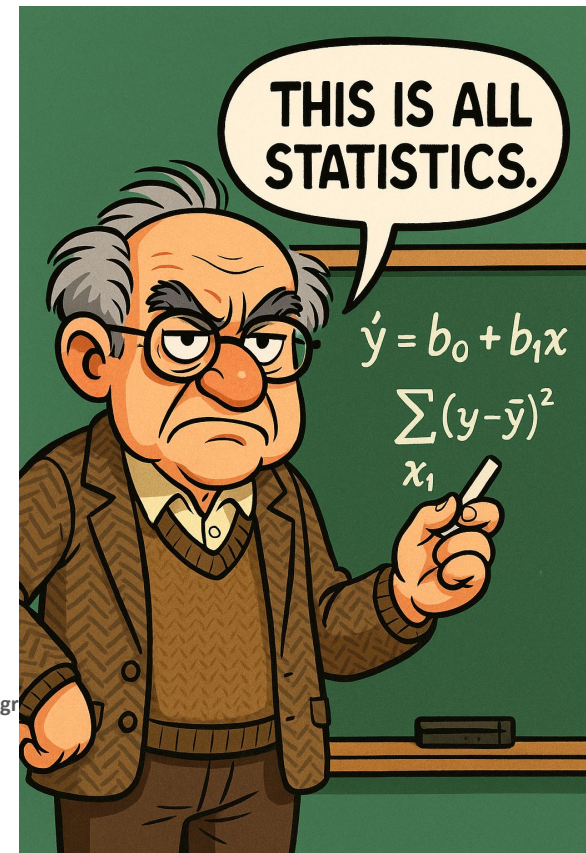
# Regressão Linear!

```
print("Hello Machine Learning!")
```

# Regressão Linear e Machine Learning

## Linha do tempo e raízes da Regressão Linear (RL)

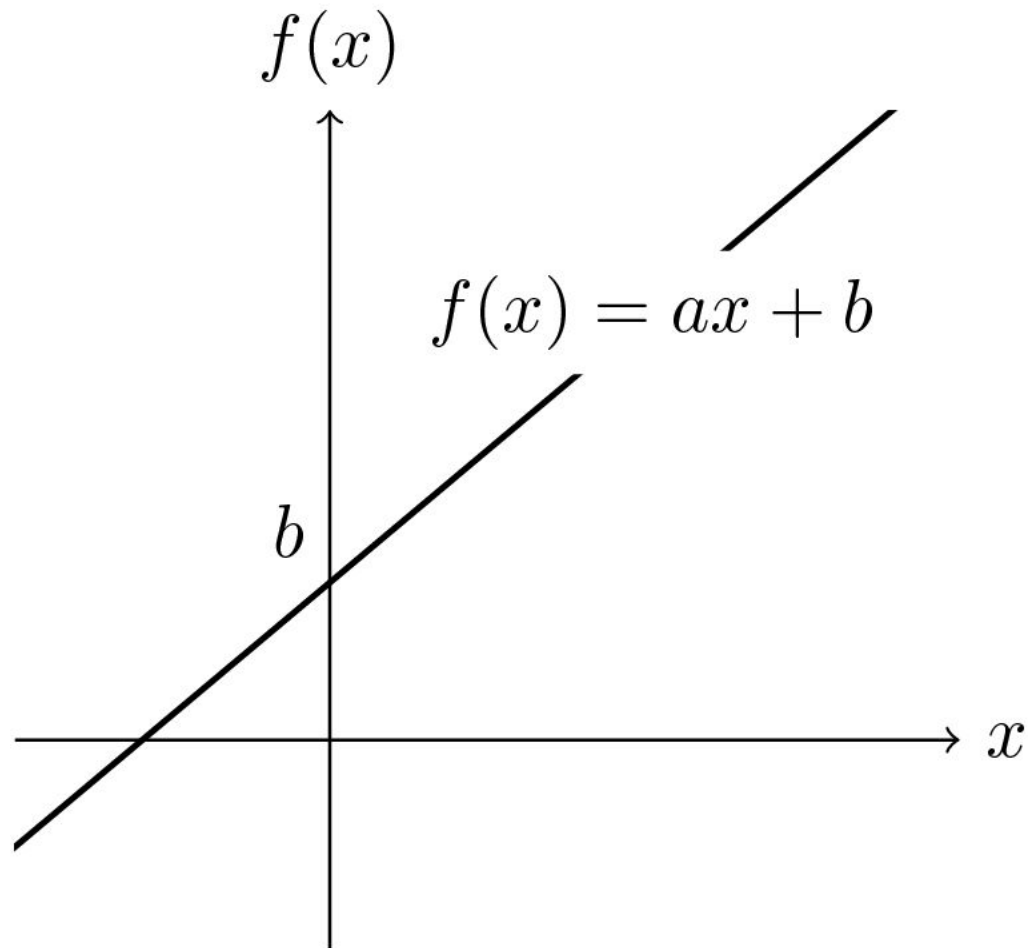
- **1805 – Método dos Mínimos Quadrados (Legendre)**
  - Publicado em *Nouvelles méthodes pour la détermination des orbites des comètes*.
  - Objetivo: ajustar órbitas cometárias — um problema de **astronomia matemática**.
- **1809 – Justificativa Probabilística (Gauss)**
  - Afirma ter usado o método desde 1795 e o fundamenta no erro normalmente distribuído.
  - Campo: **matemática aplicada & estatística nascente**.
- **1877-1886 – Termo “regressão” (Francis Galton)**
  - Observa que a altura dos filhos “regride” à média parental (“regression toward mediocrity”).
  - Surge na **biometria** (estatística aplicada à biologia humana).
- **Início do séc. XX – Formalização Estatística**
  - Karl Pearson, R. A. Fisher, Jerzy Neyman refinam inferência, testes F, ANOVA.
  - RL vira técnica-padrão de **estatística, biologia, agricultura e econometria**.
- **Décadas de 1950-1970 – Computação & Econometria**
  - Mainframes popularizam ajustes de grandes modelos lineares.
  - Economistas chamam de “**modelos econométricos lineares**”; engenheiros falam em “**análise de regressão**”.
- **Décadas de 1980-1990 – IA & Data Mining**
  - Surgem “aprendizado supervisionado” e “algoritmos de treinamento”.
  - RL é rebatizada como “**Linear Regression Learner**” – o “Hello World” do **Machine Learning (ML)**.



# Porque a Regressão Linear é o Melhor Lugar para Começar

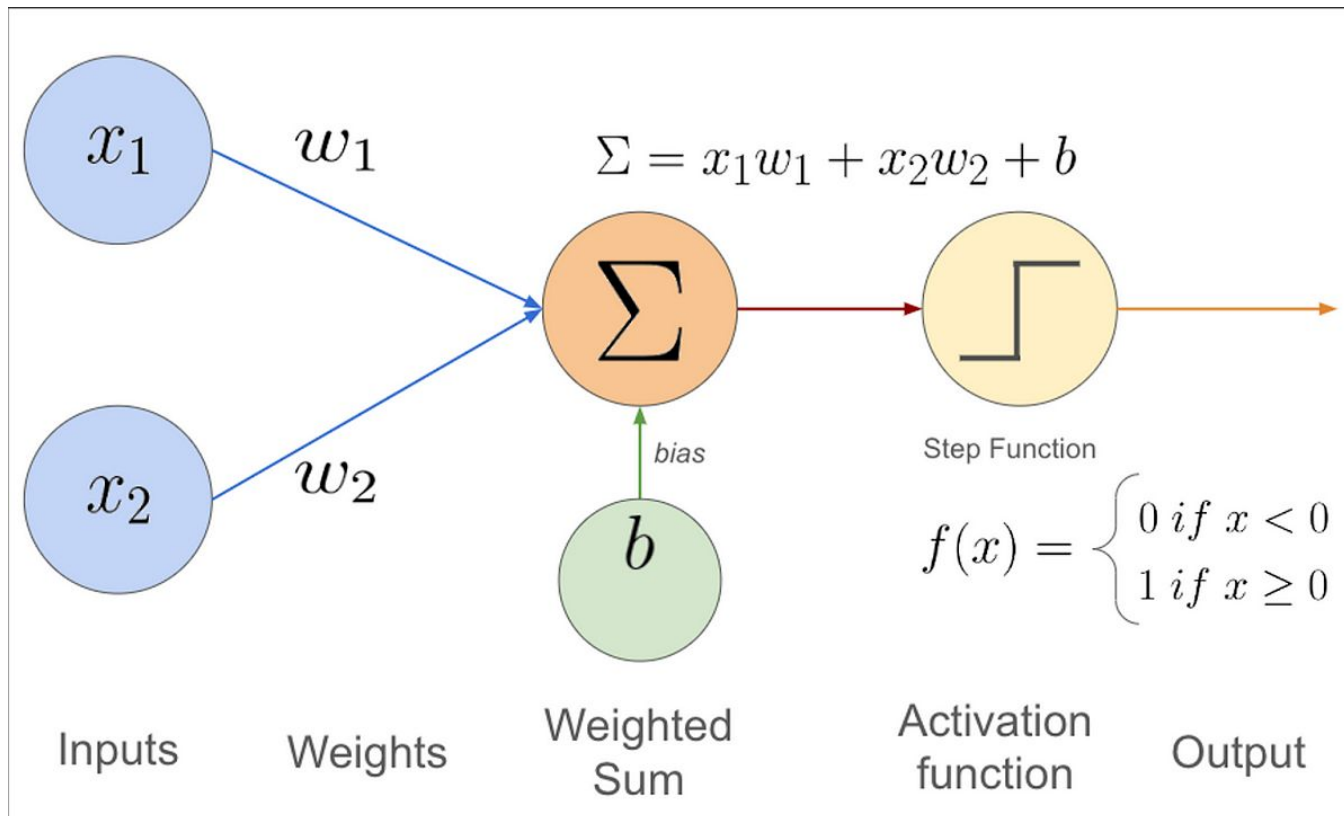
- **“HELLO WORLD” DO MACHINE LEARNING**
- Fórmula simples, conceitos centrais – Introduz parâmetros, função de perda (MSE) e ajuste de modelos sem complexidade matemática excessiva.
- Pipeline de ML completo em miniatura – Permite demonstrar treino / teste, métricas e validação cruzada já na primeira aula.
- Interpretabilidade imediata – Coeficientes mostram o impacto de cada feature, facilitando discussões sobre viés e causalidade.
- Base para técnicas avançadas – Serve de trampolim direto para regularização (Ridge/Lasso) e para classificação (regressão logística).
- Relevância prática + feedback rápido – Modelo treina em segundos e aparece em praticamente todas as áreas de mercado, motivando os alunos desde o início.

# Linear Regression -> Perceptron





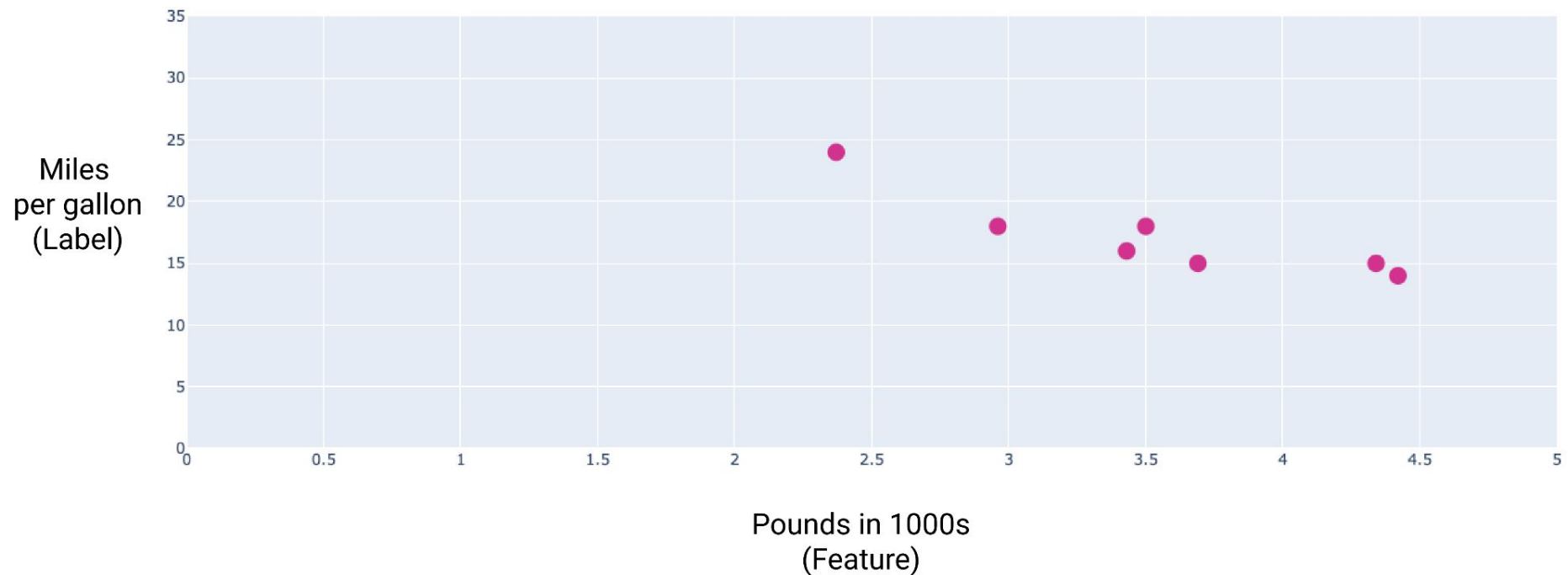
# Linear Regression -> Perceptron



Hoje vamos treinar nosso primeiro  
modelo de Machine Learning!

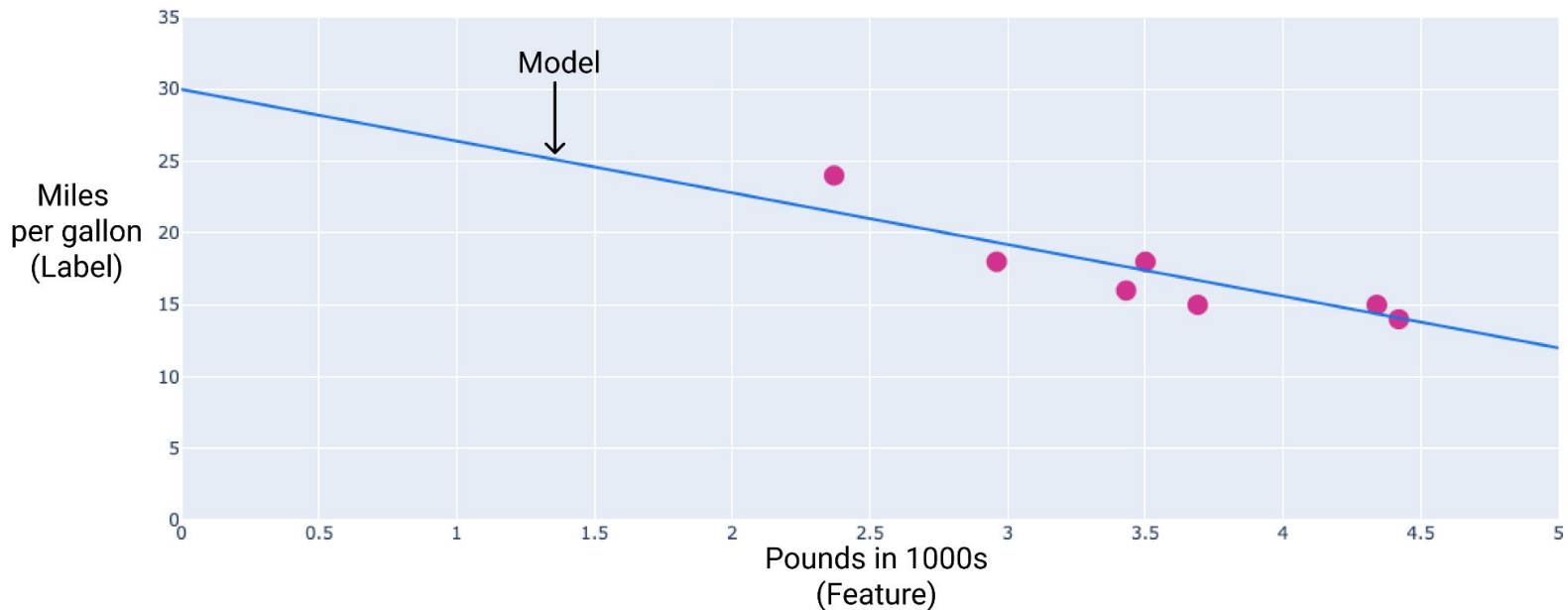
# Regressão Linear

- Estabelecer uma relação (linear) entre variáveis.
- Como você desenharia uma linha que melhor se 'ajusta' a esses pontos?



# Regressão Linear

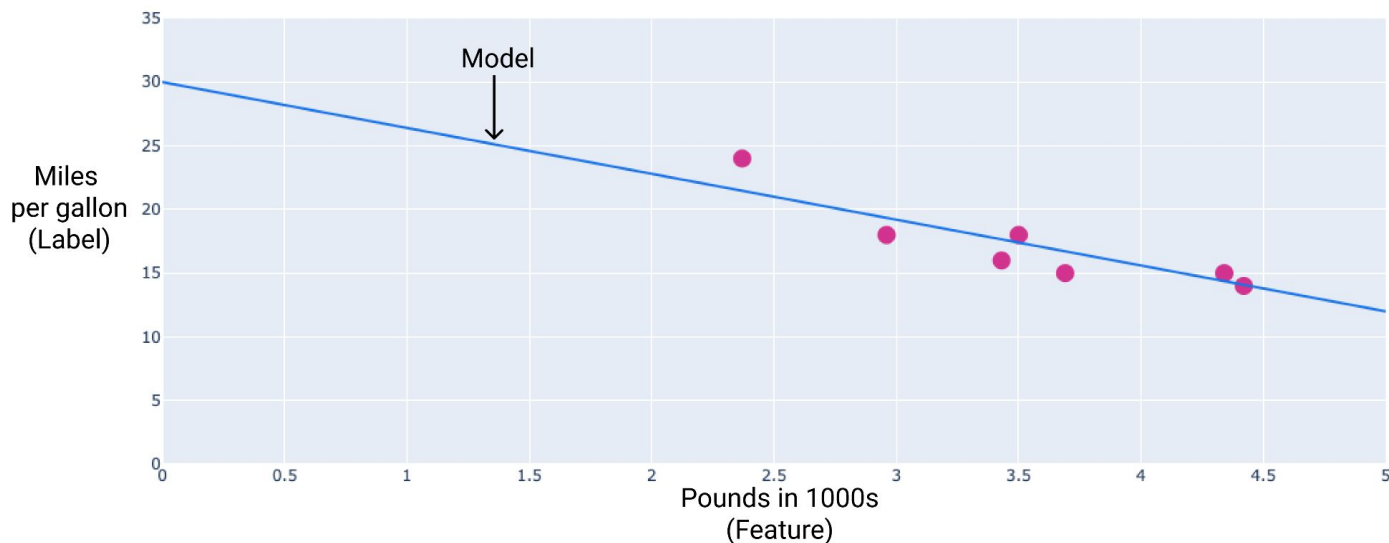
- O modelo é a linha que melhor se ‘ajusta’ aos pontos.
- Com essa linha, conseguimos estender a relação e agora encontrar o consumo de combustível para veículos em qualquer peso que quisermos.



# Equação da Regressão Linear

- $y$  = valor do consumo do combustível
- $m$  = inclinação da curva (multiplicador da variável do peso do veículo)
- $x$  = peso do veículo
- $b$  = ponto onde a reta cruza o valor zero

$$y = mx + b,$$



# Equação da Regressão Linear

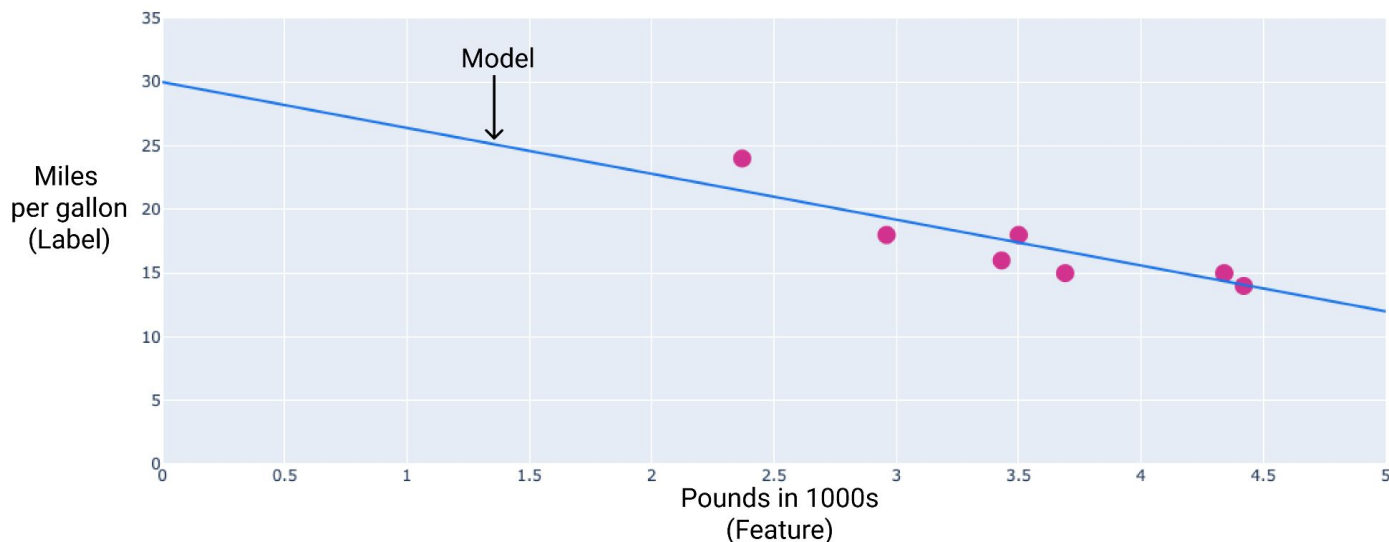
Para este modelo, temos:

- $m = -3.6$
- $b = 30$

$$y = mx + b,$$

Logo, a equação de nosso modelo fica:  $y = (-3.6).x + 30$

Assim, para um dado peso do veículo (x) conseguimos obter seu consumo (y) resolvendo a equação.



# Equação da Regressão Linear

Em Machine Learning, mudamos os nomes dos termos, mas o conceito permanece.

$$y = mx + b \quad \longrightarrow \quad y' = b + w_1 x_1$$

Notação Estatística	Nome Estatístico	Notação Machine Learning	Nome Machine Learning
y	Variável resposta estimada	y'	Saída prevista (output)
m	Coeficiente de regressão	w <sub>1</sub>	Peso (weight)
x	Variável explicativa/preditor	x <sub>1</sub>	Característica (feature)
b	Termo constante/intercepto	b	Viés (bias)
y = mx + b	Equação de regressão linear	y' = b + w <sub>1</sub> x <sub>1</sub>	Modelo linear

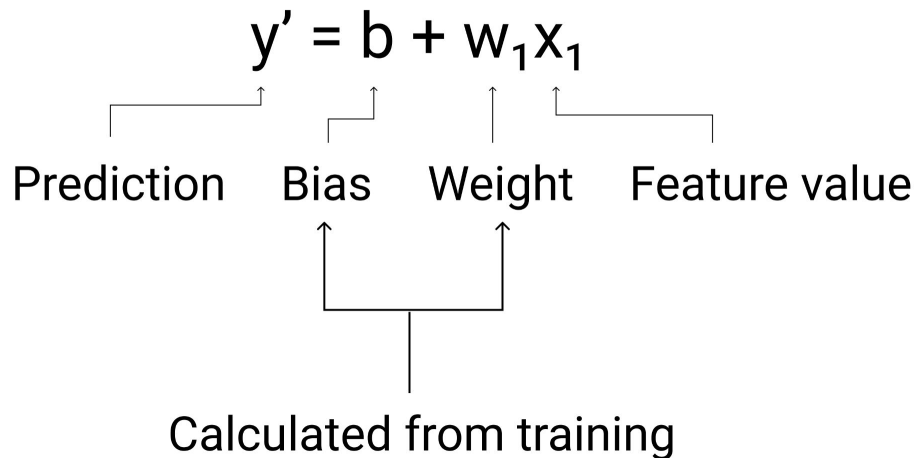
# Equação da Regressão Linear

Definir a equação é encontrar os valores dos parâmetros  $m$  e  $b$ :

$$y = m.x + b \rightarrow y = (-3.6).x + 30$$

Em Machine Learning, encontrar o valor dos parâmetros que melhor se ajusta aos dados, que melhor resolvem o problema, é chamado **treino**.

**Treinar o Modelo = Encontrar o valor dos parâmetros**

$$y' = b + w_1 x_1$$


Prediction    Bias    Weight    Feature value

Calculated from training



# Equação da Regressão Linear

O modelo anterior é simples, contendo apenas uma feature (variável de entrada).

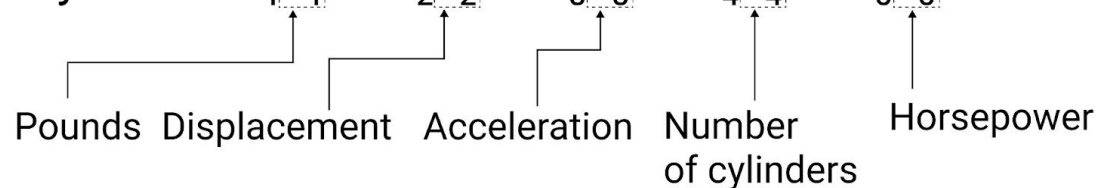
Sabemos que nossa realidade, modelos simples assim não são realidades.

Modelos mais sofisticados podem ser feitos adicionando mais features.

$$y' = b + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5$$

Assim, cada nova feature ganha um peso **w**.

Cada um dos valores de x são diferentes dados de entrada.

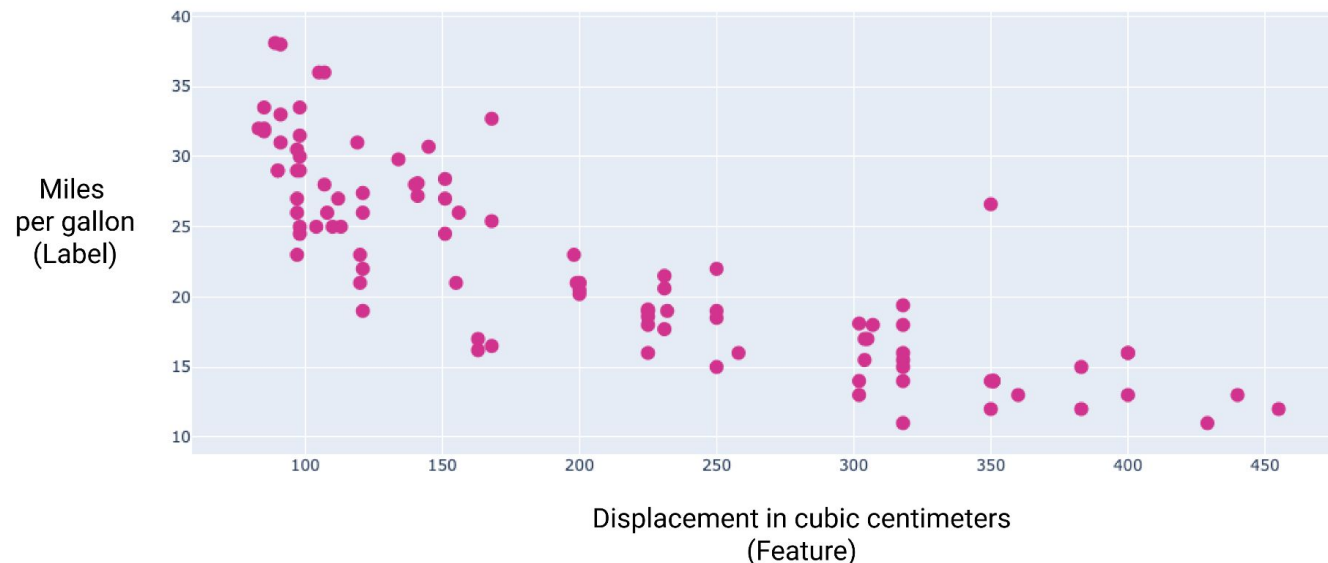
$$y' = b + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5$$


Pounds   Displacement   Acceleration   Number of cylinders   Horsepower

# Equação da Regressão Linear

Treinar o modelo, é encontrar todos os valores de  $\mathbf{w}$  (parâmetros), que encontre a curva que melhor se ajusta ao conjunto de dados.

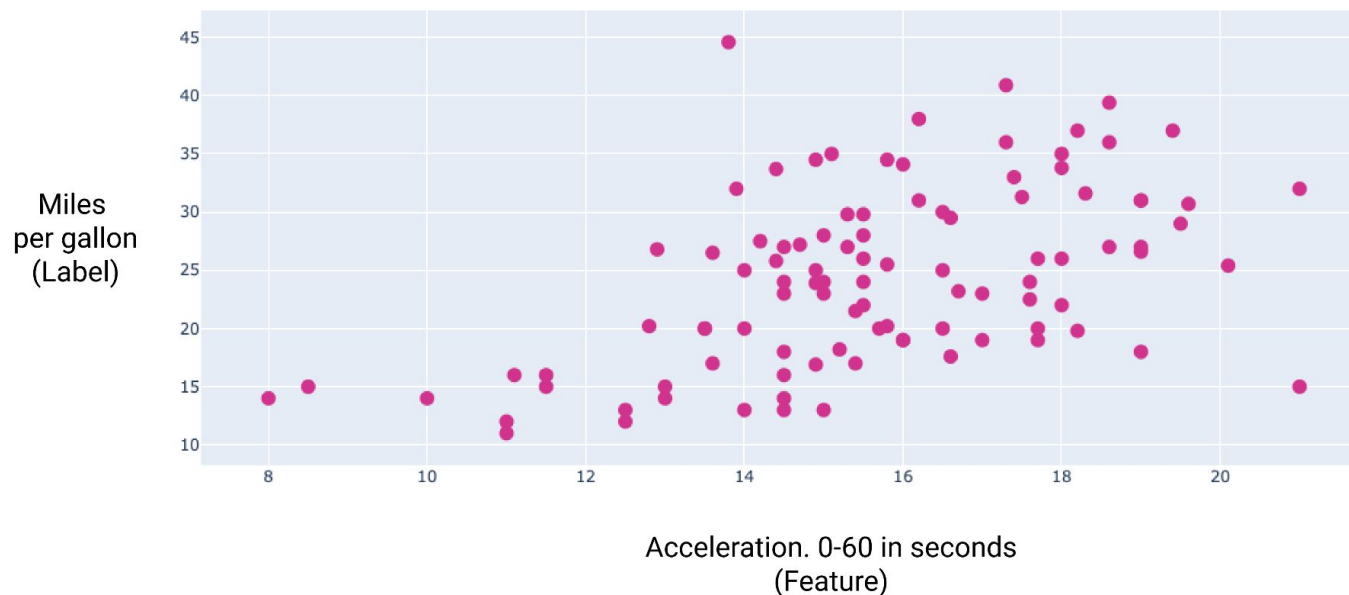
Como este modelo é linear, é recomendado que as outras variáveis também tenham relação linear com a saída



# Equação da Regressão Linear

Treinar o modelo, é encontrar todos os valores de  $\mathbf{w}$  (parâmetros), que encontre a curva que melhor se ajusta ao conjunto de dados.

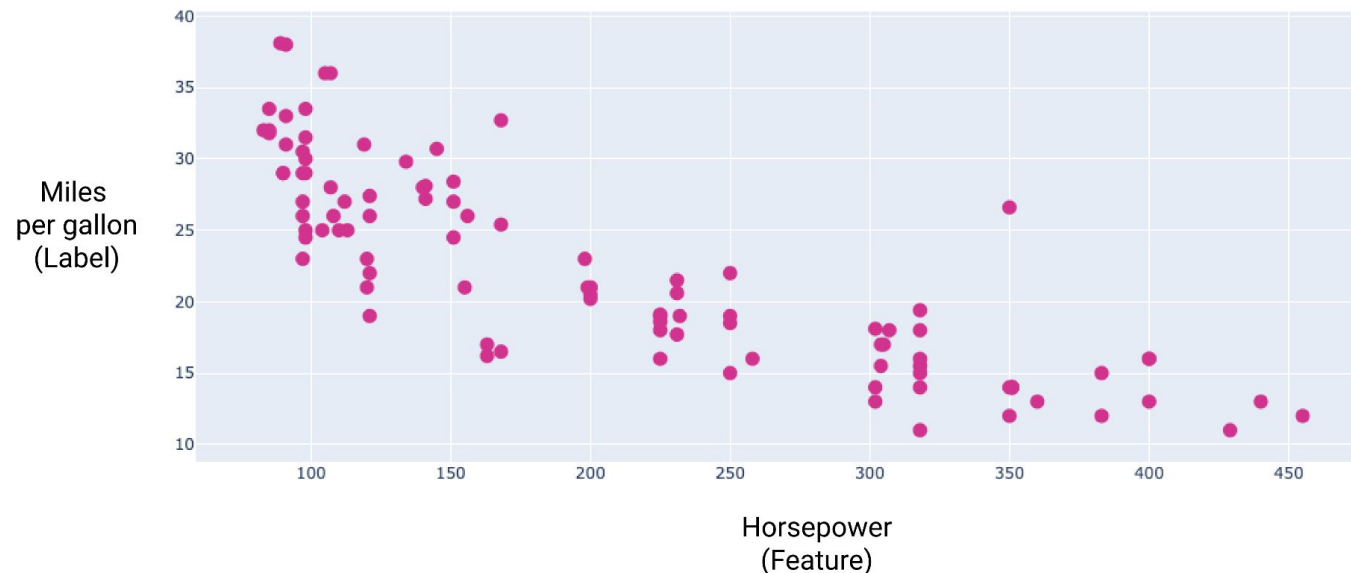
Como este modelo é linear, é recomendado que as outras variáveis também tenham relação linear com a saída



# Equação da Regressão Linear

Treinar o modelo, é encontrar todos os valores de  $\mathbf{w}$  (parâmetros), que encontre a curva que melhor se ajusta ao conjunto de dados.

Como este modelo é linear, é recomendado que as outras variáveis também tenham relação linear com a saída

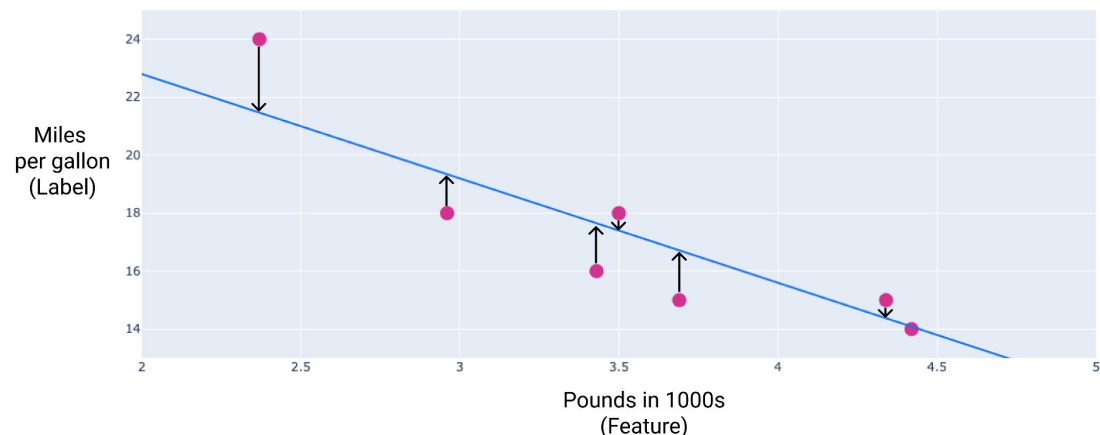



Mas o que significa, matematicamente, melhor se ajustar a curva?

# Função de custo

- A **função de custo** é o que usamos para medir o 'quão bem ajustado a curva está'.
- Em outras palavras, a função de custo mede se **o modelo aprendeu** com os dados.
- Também é conhecida por outros nomes, como **loss function**, e **error function**.
- Para regressão linear, a **loss function** é o **Erro Quadrático Médio (L2)**, cada modelo tem sua função de custo.
- O algoritmo de treino busca pelos parâmetros  $w$  e  $b$ , em  $y=w.x+b$ , de maneira a minimizar o valor da **loss function**.

$$\sum (actual\ value - predicted\ value)^2$$



 Loss lines
  Model

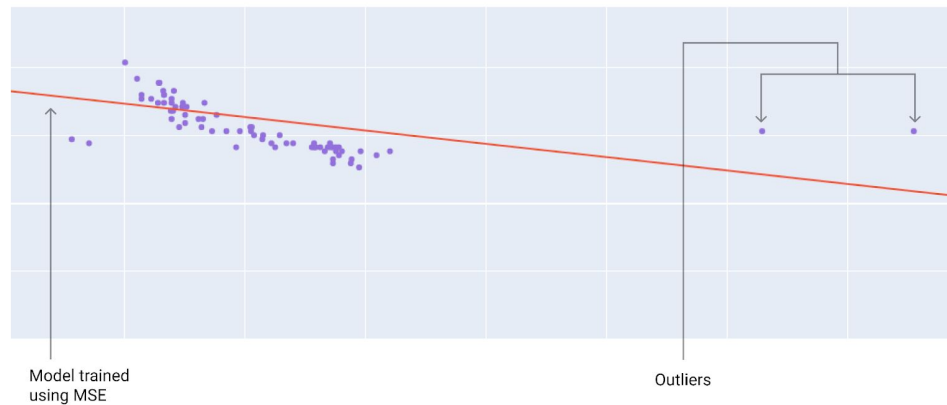
# Função de custo

- Existem diferentes **loss function**, que são diferentes maneiras de medir o erro:

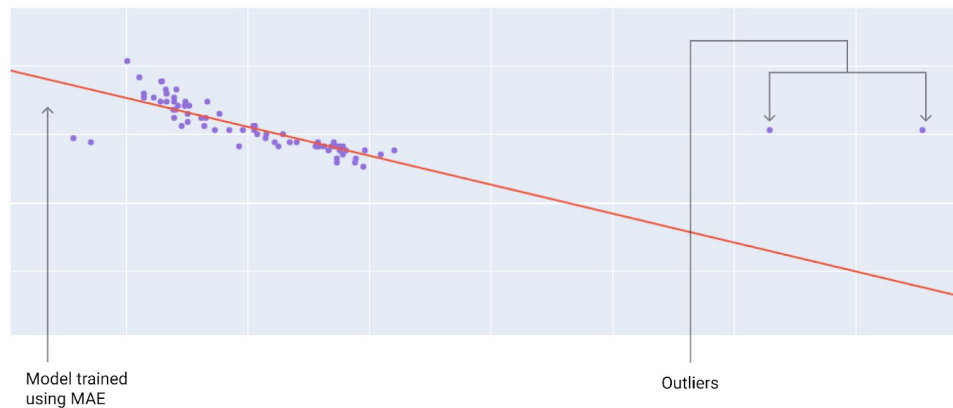
Tipo de Loss	Propósito principal	Equação
Perda $L_1$	Tornar o modelo <b>mais robusto a outliers</b> : cada erro contribui linearmente, então valores extremos não distorcem tanto o ajuste. Boa quando você quer <b>minimizar o desvio absoluto</b> total.	$\sum  \text{valor real} - \text{valor previsto} $
Erro absoluto médio (MAE)	Fornecer uma <b>métrica fácil de interpretar</b> , na mesma unidade da variável-alvo, e ainda manter robustez razoável a outliers. Útil para avaliar desempenho quando você quer saber o erro médio típico.	$\frac{1}{N} \sum  \text{valor real} - \text{valor previsto} $
Perda $L_2$	Priorizar a <b>redução de grandes erros</b> : como o erro é elevado ao quadrado, desvios maiores recebem penalidade bem maior. <b>É mais sensível a outliers</b> .	$\sum (\text{valor real} - \text{valor previsto})^2$
Erro quadrático médio (MSE)	Servir como <b>função-objetivo padrão</b> em regressão e métrica para comparação entre modelos. Mantém as propriedades suaves da $L_2$ , mas normaliza pela quantidade de exemplos, facilitando a leitura do valor e a convergência em otimização por gradiente.	$\frac{1}{N} \sum (\text{valor real} - \text{valor previsto})^2$

# Função de custo

- Um modelo treinado com MSE aproxima o modelo dos outliers.



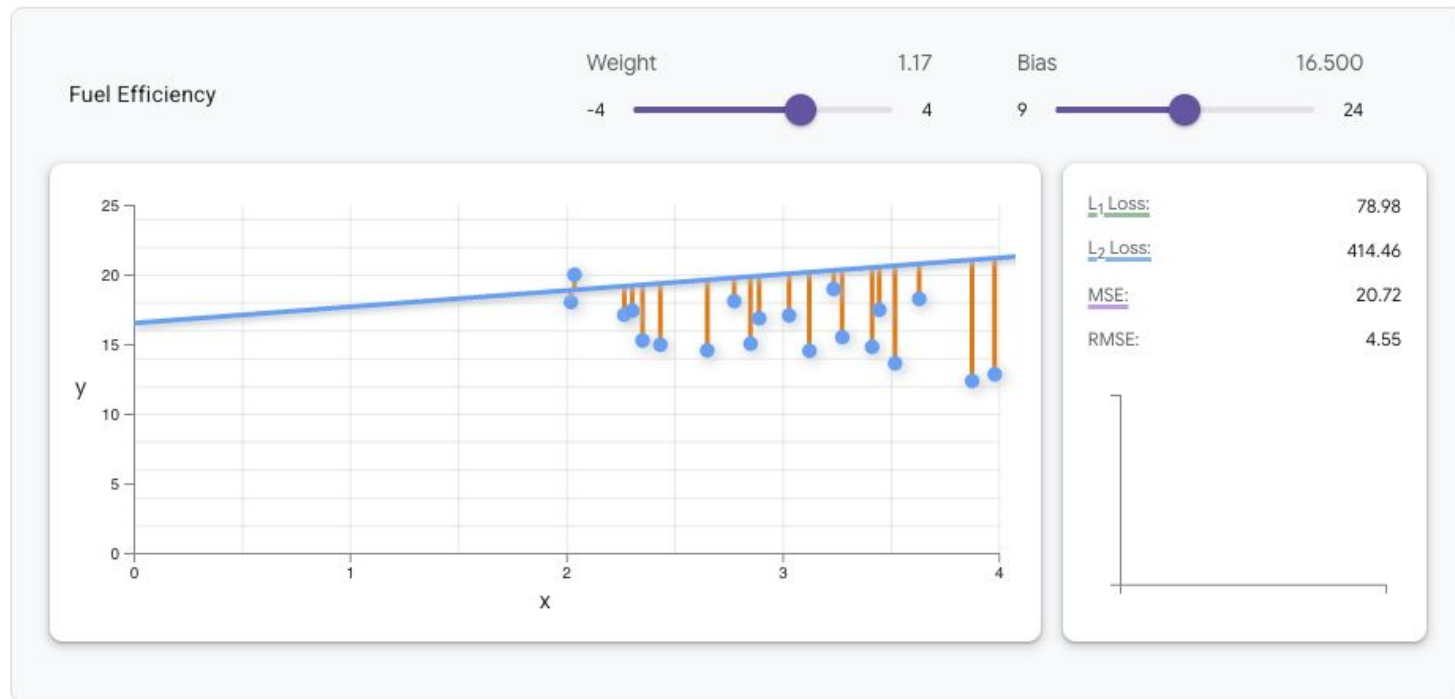
- Um modelo treinado com MAE (Mean Absolute Error) fica mais distante dos outliers.





# Função de custo

Ajustando os parâmetros manualmente



[Link](#)

# Métricas de Desempenho para Regressão

# Função de Custo vs Métrica de Avaliação

**Função de custo = função de perda = função objetivo = loss function**

**Função de custo  $\neq$  Métrica de avaliação**

- Função de perda (loss)
  - Usada durante o treinamento para minimizar o erro
  - Foco em eficiência computacional
  - Exemplo: MSE (Mean Squared Error)
- Métrica de desempenho (metric)
  - Usada para avaliar a qualidade do modelo nos dados de teste
  - Foco em interpretação prática
  - Exemplos:
    - $R^2$ : proporção da variância explicada
    - MAE: erro médio absoluto em unidades reais
    - RMSE: raiz do erro quadrático médio

# Principais Métricas para Regressão

## Erro Quadrático Médio (MSE - Mean Squared Error)

- Calcula a média dos quadrados das diferenças entre as previsões e os valores reais.
- Utilidade: Útil para penalizar erros maiores, destacando discrepâncias entre previsões e valores reais.
- Faixa: De 0 até o infinito (quanto maior, pior).

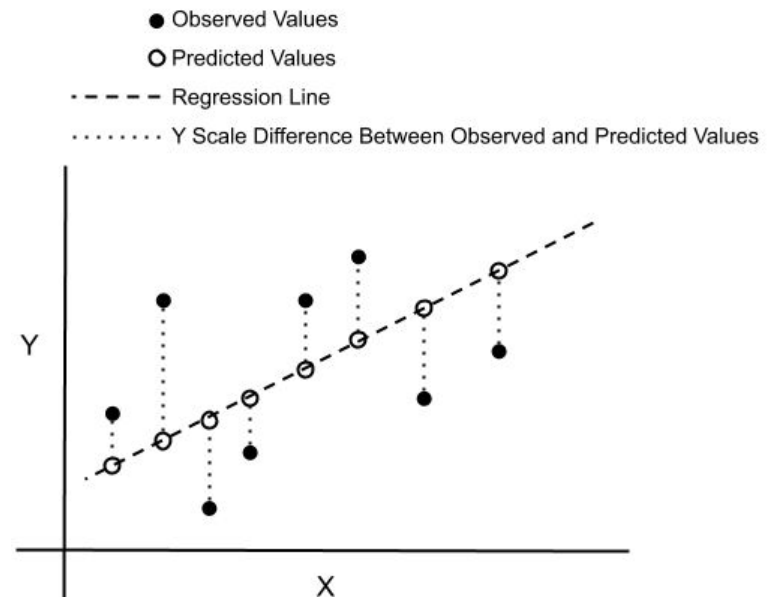
$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2$$

MSE = mean squared error

$n$  = number of data points

$Y_i$  = observed values

$\hat{Y}_i$  = predicted values



# Principais Métricas para Regressão

## Raiz do Erro Quadrático Médio (RMSE - Root Mean Squared Error)

- É a raiz quadrada do MSE e fornece uma medida do erro **em unidades originais**.
- Utilidade: Oferece uma métrica de erro mais facilmente interpretável em comparação com o MSE.
- Faixa: De 0 até o infinito (quanto maior, pior).

$$RMSE = \sqrt{\frac{\sum_{i=1}^N (Predicted_i - Actual_i)^2}{N}}$$

# Principais Métricas para Regressão

## Erro Médio Absoluto (MAE - Mean Absolute Error)

- Calcula a média das diferenças absolutas entre as previsões e os valores reais.
- Utilidade: Menos sensível a valores discrepantes do que o MSE, sendo útil quando se deseja evitar que valores extremos distorçam a métrica.
- Faixa: De 0 até o infinito (quanto maior, pior).

$$MAE = \frac{1}{n} \sum \left| y - \hat{y} \right|$$

Diagram illustrating the MAE formula components:

- Divide by the total number of data points:** Points to the  $\frac{1}{n}$  term.
- Sum of:** Points to the  $\sum$  symbol.
- Actual output value:** Points to the  $y$  term inside the absolute value.
- Predicted output value:** Points to the  $\hat{y}$  term inside the absolute value.
- The absolute value of the residual:** Points to the entire absolute value expression  $|y - \hat{y}|$ .

# MAE vs RMSE

- **MAE (Erro Médio Absoluto):**

- Use o MAE quando você deseja ter uma ideia clara da magnitude média dos erros.
- O MAE é menos sensível a valores discrepantes, pois considera apenas as diferenças absolutas.
- É uma boa escolha quando você quer uma métrica simples e fácil de interpretar, especialmente se houver outliers nos dados.

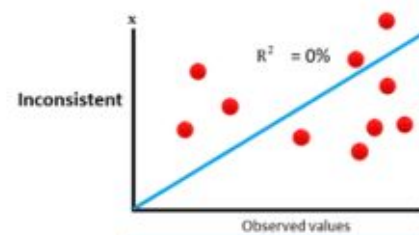
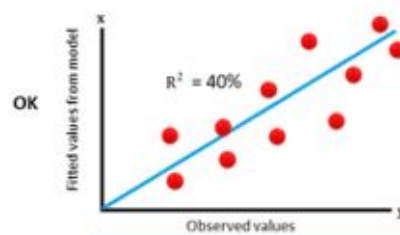
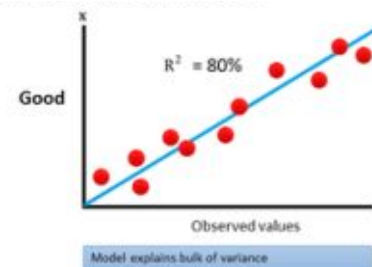
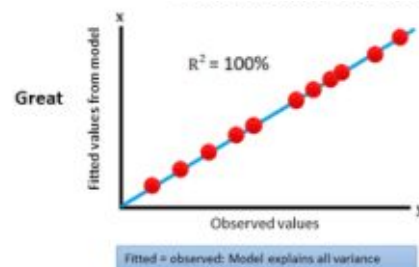
- **RMSE (Raiz do Erro Quadrático Médio):**

- Prefira o RMSE quando erros grandes devem ser penalizados mais fortemente.
- O RMSE amplifica o efeito de grandes erros devido ao processo de elevar ao quadrado as diferenças.
- É útil em situações em que você quer dar mais peso a erros maiores, como quando a precisão é crítica e erros significativos são particularmente indesejáveis.

# Principais Métricas para Regressão

## Coeficiente de Determinação ( $R^2$ - R-squared)

- Avalia a proporção da variabilidade nos dados que é explicada pelo modelo.
- Utilidade: Fornece uma medida da qualidade global do modelo, quanto mais próximo de 1, melhor o ajuste.
- Faixa: De 0 a 1 (quanto maior, melhor).





# Principais Métricas para Regressão

## Erro Percentual Absoluto Médio (MAPE - Mean Absolute Percentage Error)

- Calcula a média das porcentagens das diferenças absolutas entre as previsões e os valores reais.
- Utilidade: Útil quando a precisão relativa é mais importante do que a precisão absoluta.
- **Permite comparar séries com valores absolutos diferentes.**
- Faixa: De 0% até o infinito (quanto maior, pior).

$$MAPE = \frac{1}{n} \sum_{i=1}^n \frac{|A_i - F_i|}{A_i}$$

$A_i$  is the actual value

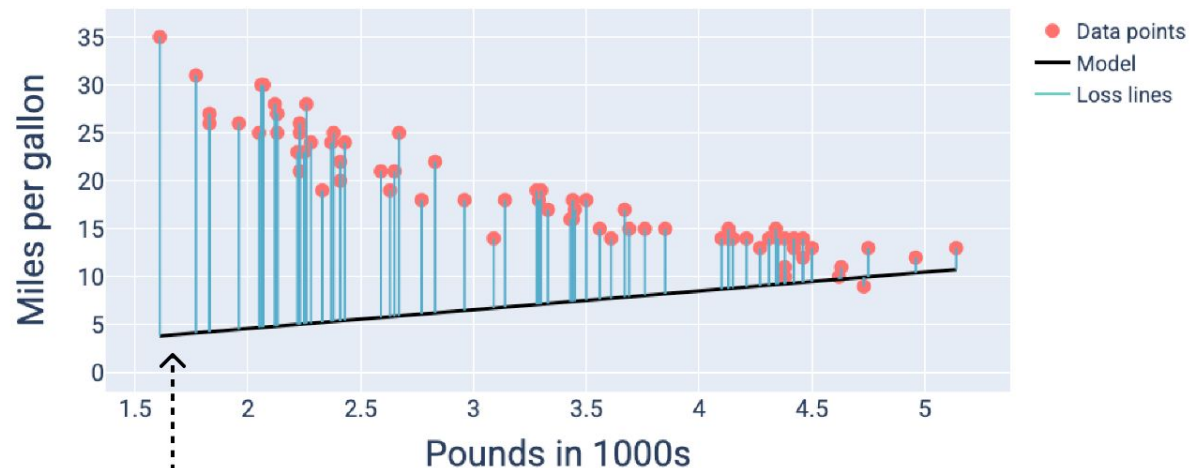
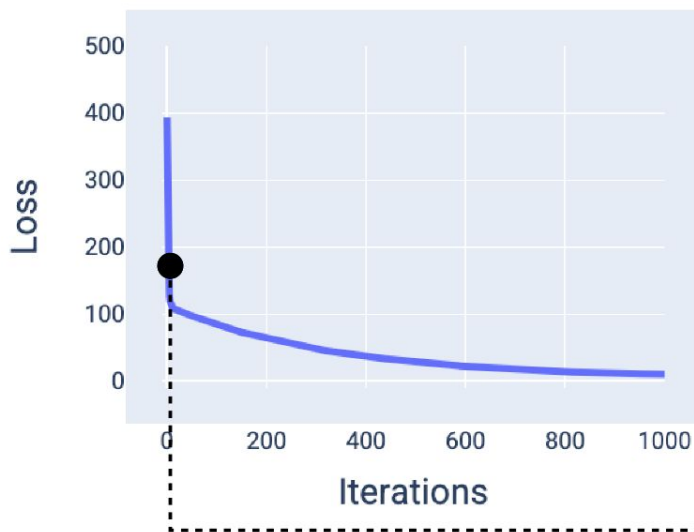
$F_i$  is the forecast value

$n$  is total number of observations

Como o algoritmo de treino encontra os valores dos parâmetros?

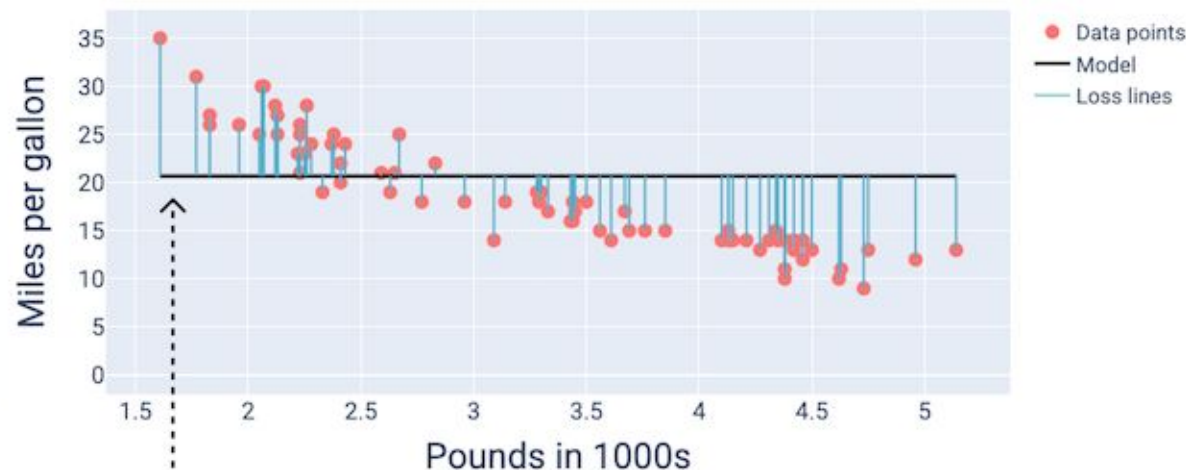
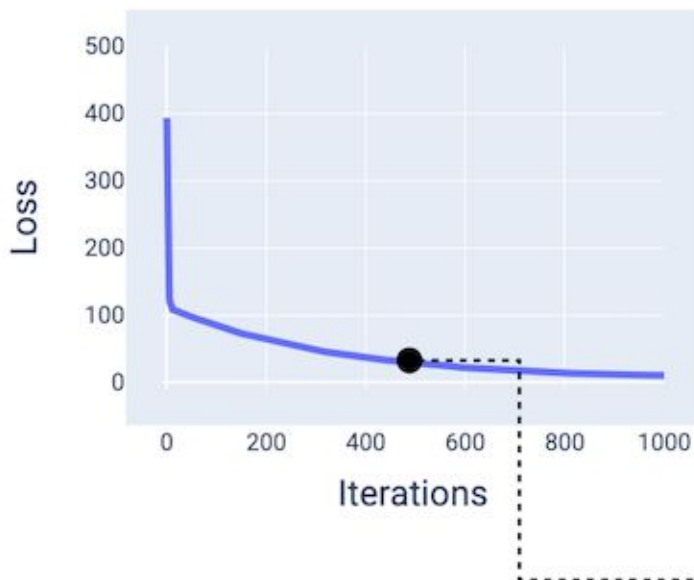
# Regressão Linear: Gradiente Descendente

1. Seta os valores de  $w$  e  $b$  para zero.
2. Usa cálculo para decidir se deve aumentar ou diminuir os valores de  $w$  e  $b$ .
3. Verifica se o valor de loss melhorou.
4. Repete até deixar de melhorar.



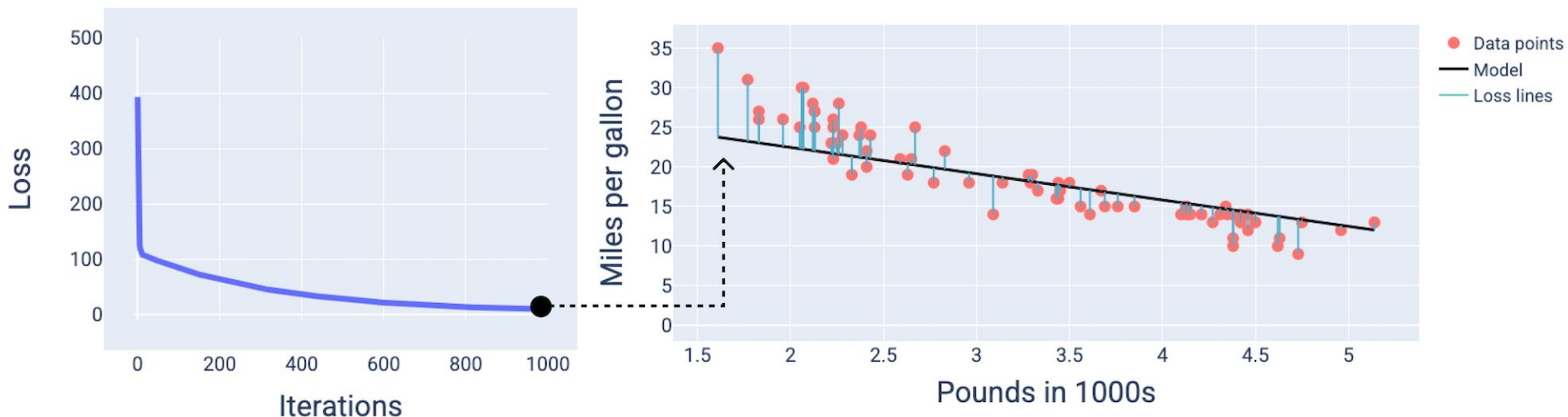
# Regressão Linear: Gradiente Descendente

1. Seta os valores de  $w$  e  $b$  para zero.
2. Usa cálculo para decidir se deve aumentar ou diminuir os valores de  $w$  e  $b$ .
3. Verifica se o valor de loss melhorou.
4. Repete até deixar de melhorar.



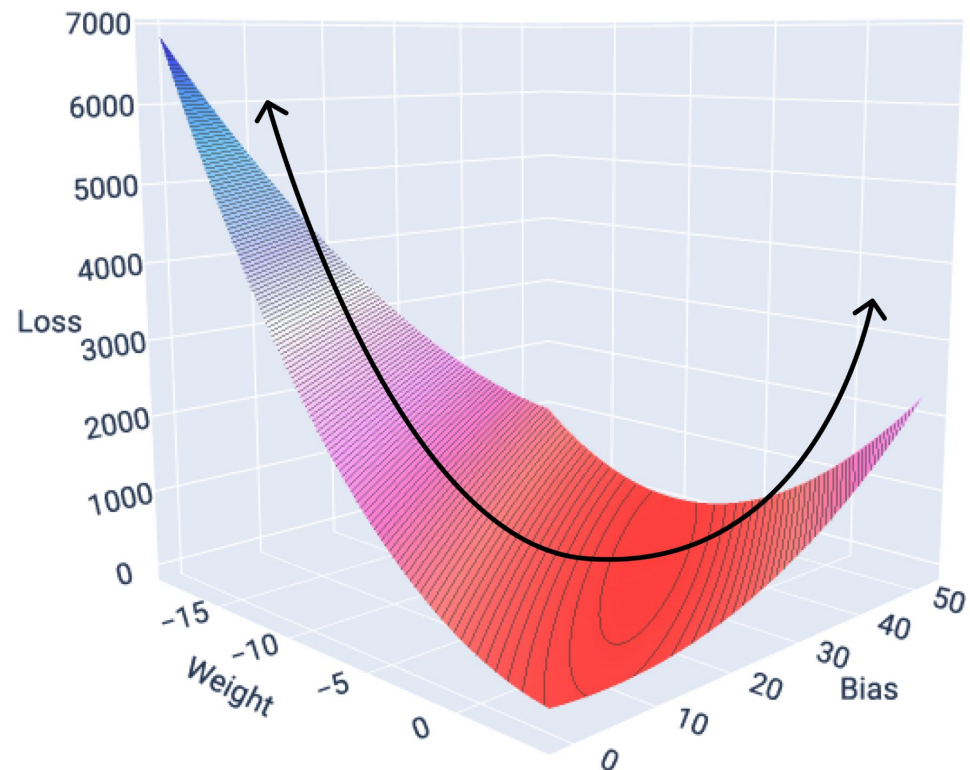
# Regressão Linear: Gradiente Descendente

1. Seta os valores de  $w$  e  $b$  para zero.
2. Usa cálculo para decidir se deve aumentar ou diminuir os valores de  $w$  e  $b$ .
3. Verifica se o valor de loss melhorou.
4. Repete até deixar de melhorar.



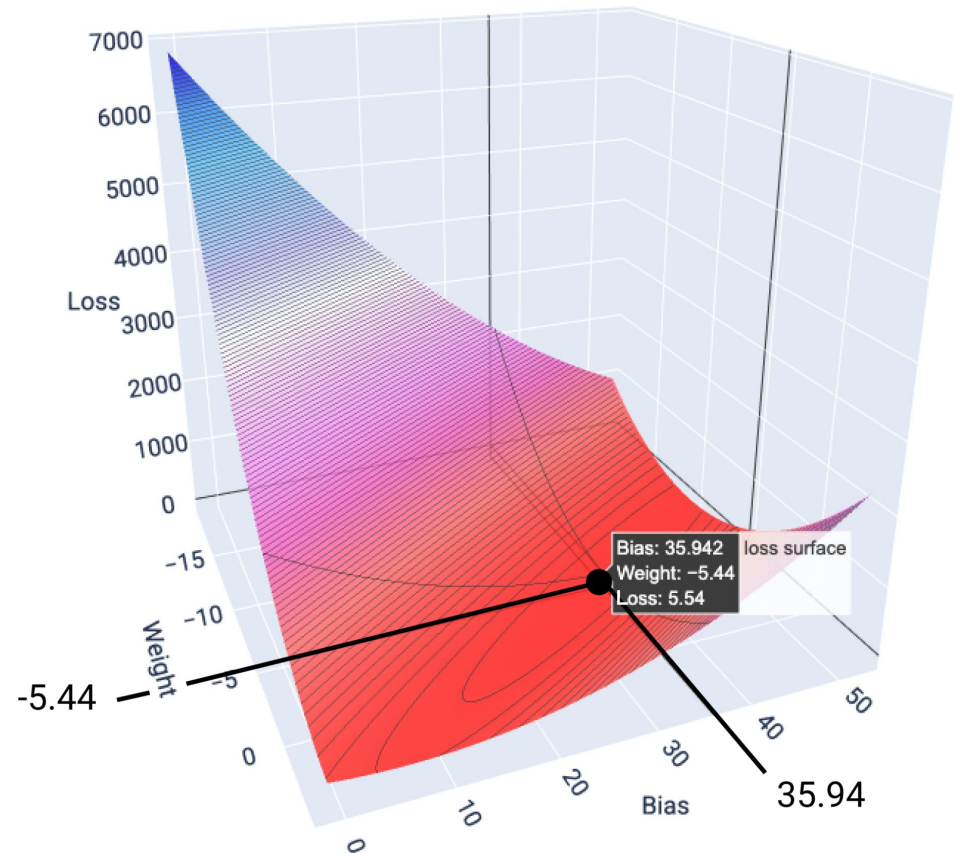
# Convergência da Função de Loss

- Modelo convergir, significa que encontramos os valores de  $w$  (weights) e  $b$  (bias) que minimizam a função Loss.



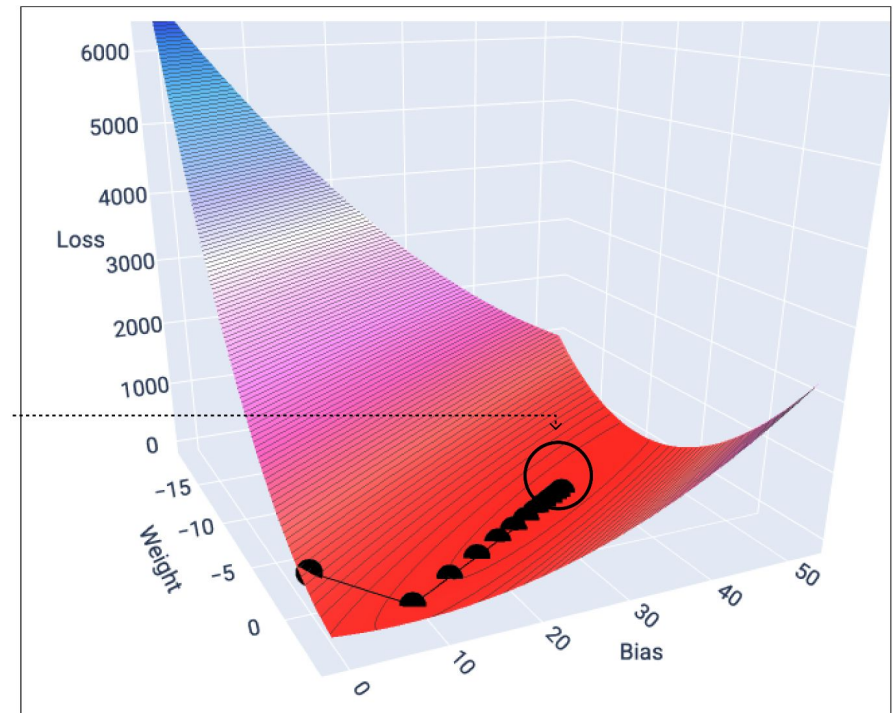
# Convergência da Função de Loss

- Modelo convergir, significa que encontramos os valores de **w (weights)** e **b (bias)** que minimizam a função **Loss**.



# Convergência da Função de Loss

- Modelo convergir, significa que encontramos os valores de **w (weights)** e **b (bias)** que **minimizam a função Loss**.
- Modelos de Regressão Linear convergem porque a superfície da função Loss é convexa e contém um ponto onde os w e b possuem um vale.
- Essa função desenhada na superfície é a mesma que vimos anteriormente.





# Hiperparâmetros

# Hiperparâmetros

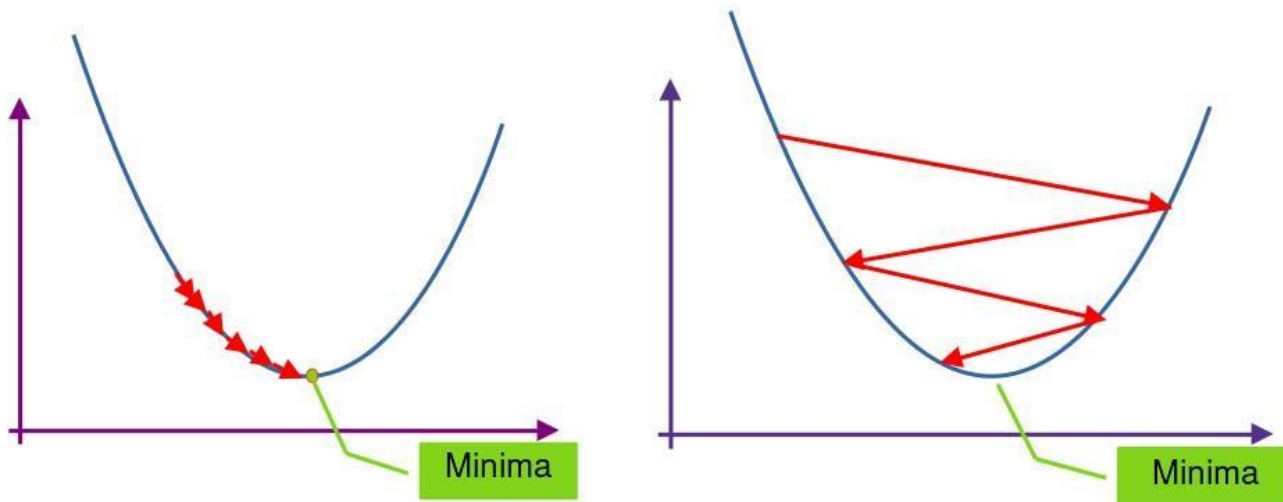
- Hiperparâmetros são variáveis que controlam diferentes aspectos do treinamento. Três hiperparâmetros comuns são:
  - Taxa de aprendizado (learning rate)
  - Tamanho do lote (batch size)
  - Épocas (epochs)
- Em contraste, os parâmetros são variáveis, como os pesos ( $w$ ) e o viés ( $b$ ), que fazem parte do próprio modelo.
- Em outras palavras, os hiperparâmetros são valores que você controla; os parâmetros são valores que o modelo calcula durante o treinamento.

# Hiperparâmetros: Learning Rate

- Controla o tamanho dos passos durante a descida do gradiente.
- Taxa muito baixa: o modelo aprende devagar.
- Taxa muito alta: o modelo nunca converge, fica "saltando" ao redor do mínimo.
- O valor ideal permite que o modelo convirja rapidamente sem instabilidade.

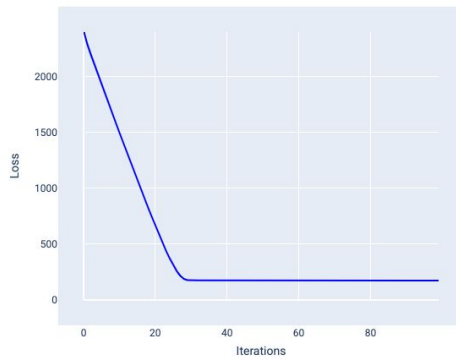
# Hiperparâmetros: Learning Rate

- Controla o tamanho dos passos durante a descida do gradiente.
- Taxa muito baixa: o modelo aprende devagar.
- Taxa muito alta: o modelo nunca converge, fica "saltando" ao redor do mínimo.
- O valor ideal permite que o modelo convirja rapidamente sem instabilidade.

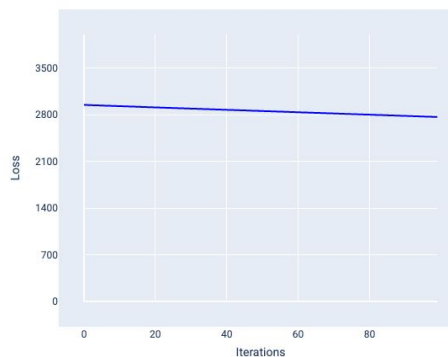


# Hiperparâmetros: Learning Rate

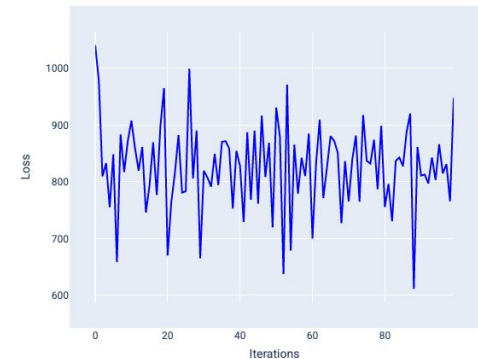
- Controla o tamanho dos passos durante a descida do gradiente.
- Taxa muito baixa: o modelo aprende devagar.
- Taxa muito alta: o modelo nunca converge, fica "saltando" ao redor do mínimo.
- O valor ideal permite que o modelo convirja rapidamente sem instabilidade.



**Bom** Learning Rate



**Baixo** Learning Rate



**Alto** Learning Rate

# Hiperparâmetros: Learning Rate

- A atualização dos pesos e viés é feita multiplicando o gradiente pela taxa de aprendizado.
- A diferença entre os valores antigos e novos dos parâmetros é proporcional à inclinação da função de perda.
- Exemplo: Gradiente = 2.5 e taxa de aprendizado = 0.01  $\rightarrow$  mudança = 0.025
- Uma boa taxa de aprendizado mostra grande melhoria nas primeiras iterações e depois estabiliza.

# Suposições da Regressão Linear

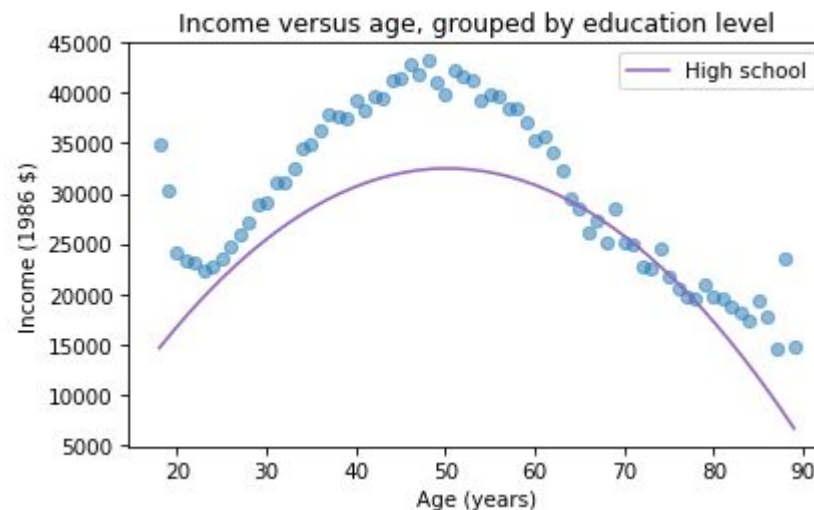
# Suposições da Regressão Linear

- Linearidade
  - A relação entre as variáveis independentes e a variável dependente é linear.
- Independência dos erros (resíduos)x
  - Os erros devem ser independentes entre si (especialmente importante em dados temporais).
- Homoscedasticidade
  - A variância dos erros deve ser constante ao longo de todos os valores previstos.
- Normalidade dos erros
  - Os resíduos devem seguir uma distribuição normal (importante para testes estatísticos e intervalos de confiança).
- Ausência de multicolinearidade
  - As variáveis independentes não devem ser altamente correlacionadas entre si.
- Os erros têm média zero
  - Espera-se que a média dos resíduos seja igual a zero.



# Suposições da Regressão Linear

- Violação da Linearidade
  - O modelo não capta corretamente a relação real entre as variáveis.
  - As previsões e interpretações (como coeficientes e p-valores) se tornam enganosas ou inválidas.
- Exemplo: Você tenta prever a renda com base na idade, mas a relação real é em formato de U (jovens e idosos ganham menos que adultos de meia-idade). O modelo linear vai forçar uma linha reta, errando nas pontas.



# Suposições da Regressão Linear

- Violação da Independência dos Erros

- Comum em dados de séries temporais ou espaciais.
- Leva à subestimação dos erros padrão, aumentando o risco de falsos positivos (erro tipo I).

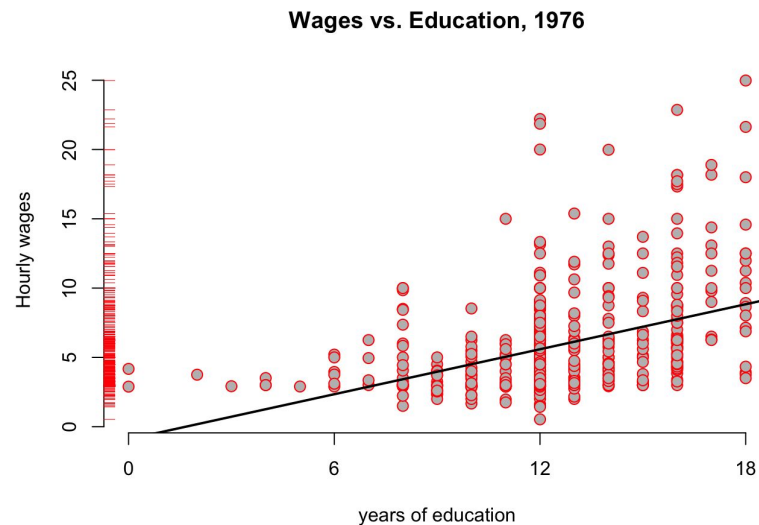
- Exemplo:

Você está prevendo o preço de ações com base nos últimos dias. Como os preços de hoje estão correlacionados com os de ontem, os erros também estarão — o modelo viola independência temporal.

Você não está aprendendo com dados novos. Se os dados são dependentes (ex: a mesma pessoa aparece várias vezes), você não está coletando informações novas e independentes. É como perguntar a mesma coisa para a mesma pessoa várias vezes e contar como se fossem várias respostas diferentes.

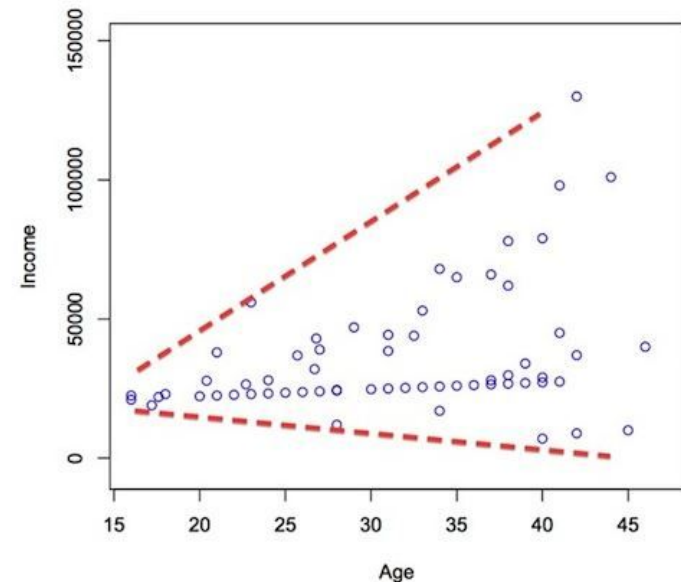
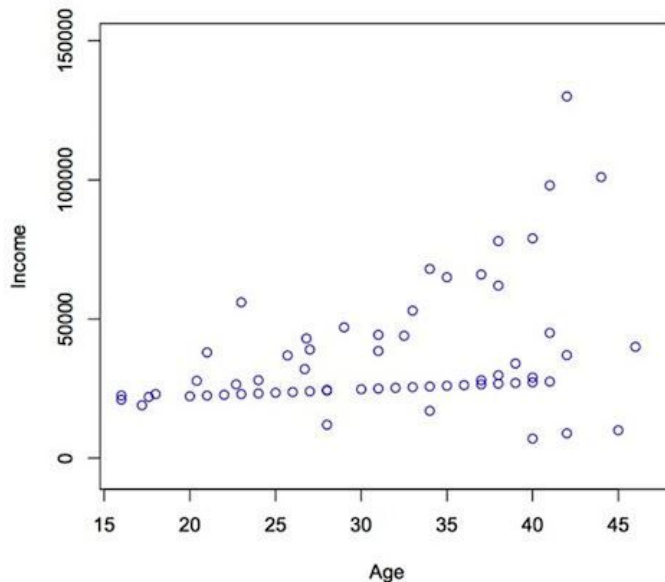
# Suposições da Regressão Linear

- Violação da Homocedasticidade
  - Quando os erros têm variância não constante (heterocedasticidade).
  - Afeta os intervalos de confiança e testes estatísticos, tornando-os não confiáveis.
- Exemplo: Você modela o gasto de clientes com base na renda. Clientes mais ricos têm gastos mais variados. Ou seja, quanto maior a renda, maior a variância dos resíduos — o que distorce a regressão.



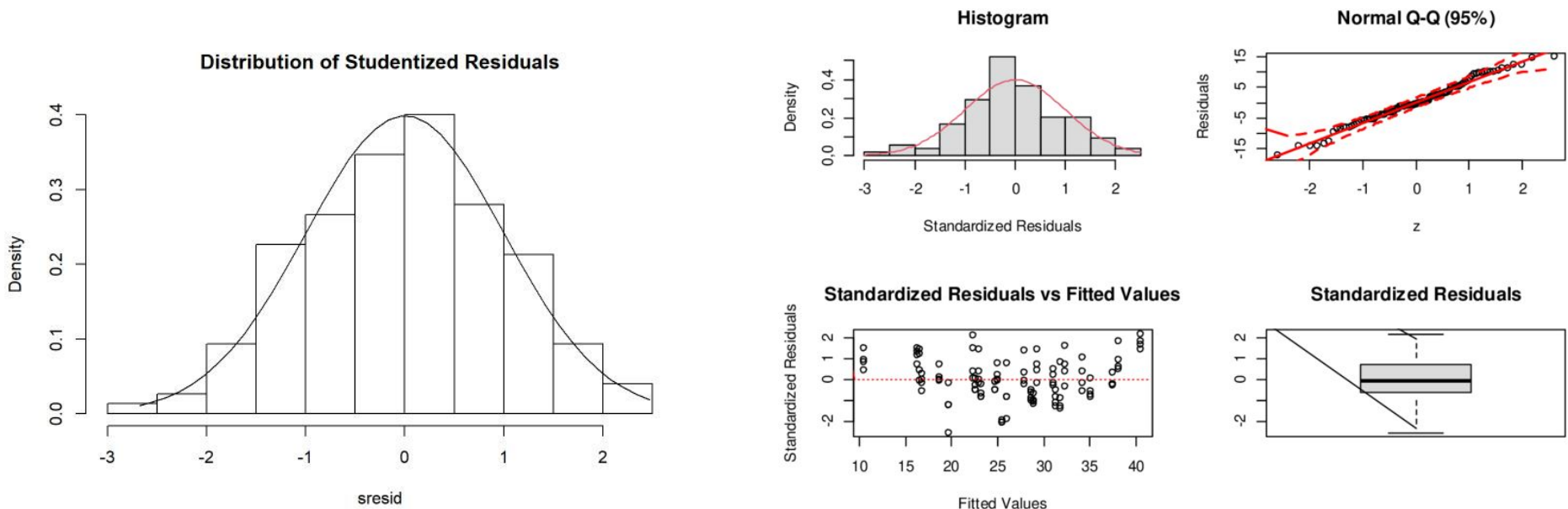
# Suposições da Regressão Linear

- Violação da Homocedasticidade
  - Quando os erros têm variância não constante (heterocedasticidade).
  - Afeta os intervalos de confiança e testes estatísticos, tornando-os não confiáveis.
- Exemplo: Você modela o gasto de clientes com base na renda. Clientes mais ricos têm gastos mais variados. Ou seja, quanto maior a renda, maior a variância dos resíduos — o que distorce a regressão.



# Suposições da Regressão Linear

- Violação da Normalidade dos Erros
  - Impacta principalmente em conjuntos de dados pequenos.
  - Prejudica a validade dos p-valores e intervalos de confiança, que assumem distribuição normal.
- Exemplo: Você treina um modelo com poucos dados e os resíduos seguem uma distribuição muito assimétrica (com muitos valores extremos). Isso afeta os testes estatísticos e pode invalidar conclusões.



# Suposições da Regressão Linear

- Violação da Ausência de Multicolinearidade
  - Dificulta identificar o efeito de cada variável.
  - Causa coeficientes instáveis, erros padrão inflados e interpretação confusa do modelo.
- Exemplo: Você usa como variáveis explicativas o número de quartos e a área construída de uma casa. Como essas variáveis estão fortemente correlacionadas, o modelo não consegue estimar bem os coeficientes.

# Suposições da Regressão Linear

- Violação da Ausência de Endogeneidade
  - Se os resíduos estão correlacionados com as variáveis independentes, os coeficientes estimados serão viesados e inconsistentes.
  - Indica que há uma relação de causa invertida, variável omitida ou medição incorreta.
- Exemplo:
  - Você quer medir o efeito de tomar café (X) na produtividade diária (Y). Pessoas mais cansadas bebem mais café e rendem menos; “cansaço” não está no modelo, vai para o erro (resíduo) e se correlaciona com X. O resultado “mostra” que café reduz produtividade, mas o viés vem do cansaço omitido — logo, a suposição de ausência de endogeneidade é violada.
  - Para estimar o efeito do número de policiais em uma cidade (X) sobre a taxa de criminalidade (Y), você usa regressão OLS. Cidades com mais crime tendem a contratar mais policiais, criando causalidade reversa: crime  $\rightarrow$  mais policiais, ao mesmo tempo que policiais  $\rightarrow$  menos crime. O erro (crimes não explicados pelo modelo) fica positivamente correlacionado com X, viesando o coeficiente e mascarando o real impacto da presença policial.

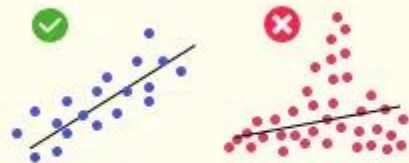
# Suposições da Regressão Linear

Em Resumo

## Assumptions of Linear Regression

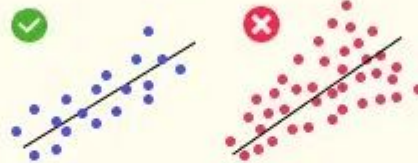
### 1. Linearity

(Linear relationship between Y and each X)



### 2. Homoscedasticity

(Equal variance)



### 3. Multivariate Normality

(Normality of error distribution)



### 4. Independence

(of observations. Includes "no autocorrelation")



### 5. Lack of Multicollinearity

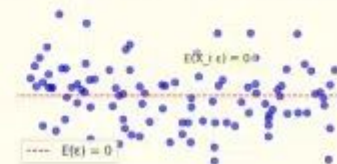
(Predictors are not correlated with each other)

$$X_1 + X_2$$

$$X_1 \sim X_2$$

### 6. Absence of endogeneity

(No correlation between predictors and errors.)





# E se as Suposições da Regressão Linear Falharem?

Quando uma suposição da regressão linear falha

- Os resultados enganam
  - O modelo passa a mostrar relações que não existem ou esconde as que existem de verdade.
- Testes estatísticos perdem sentido
  - P-valores e intervalos de confiança deixam de indicar o que é ou não importante.
- Previsões ficam arriscadas
  - Funciona bem no treinamento, mas erra mais quando aplicado em novas situações.
- Primeiro, descubra a causa
  - Use gráficos simples ou testes rápidos para identificar qual suposição foi quebrada.
- Depois, corrija ou troque de ferramenta
  - Ajuste as variáveis, aplique transformações, adote modelos robustos ou escolha outro método que combine melhor com a natureza dos seus dados.

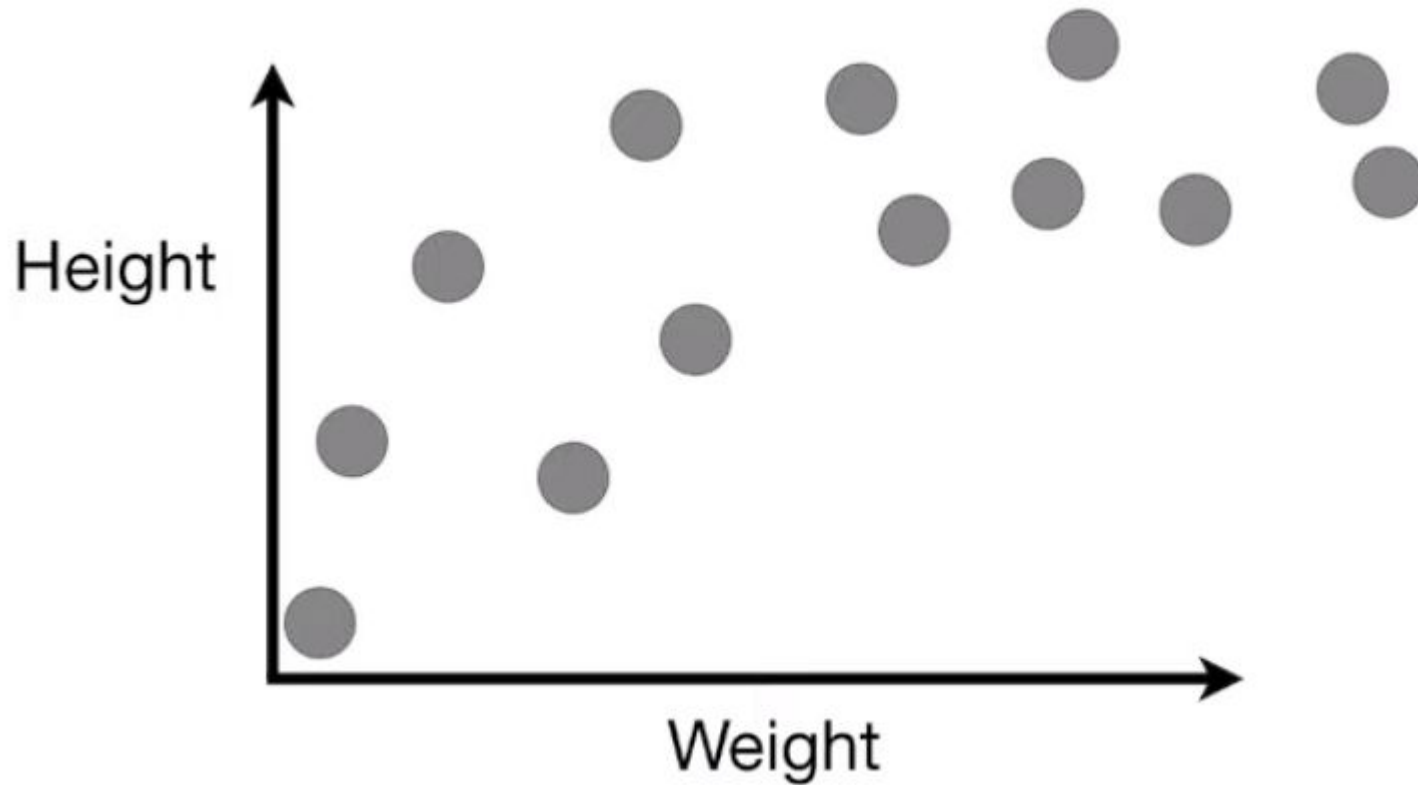
# Treinando um Modelo

## Principais Conceitos

# Underfit, Overfit, Just Right

# Underfitting, Overfitting, Just right

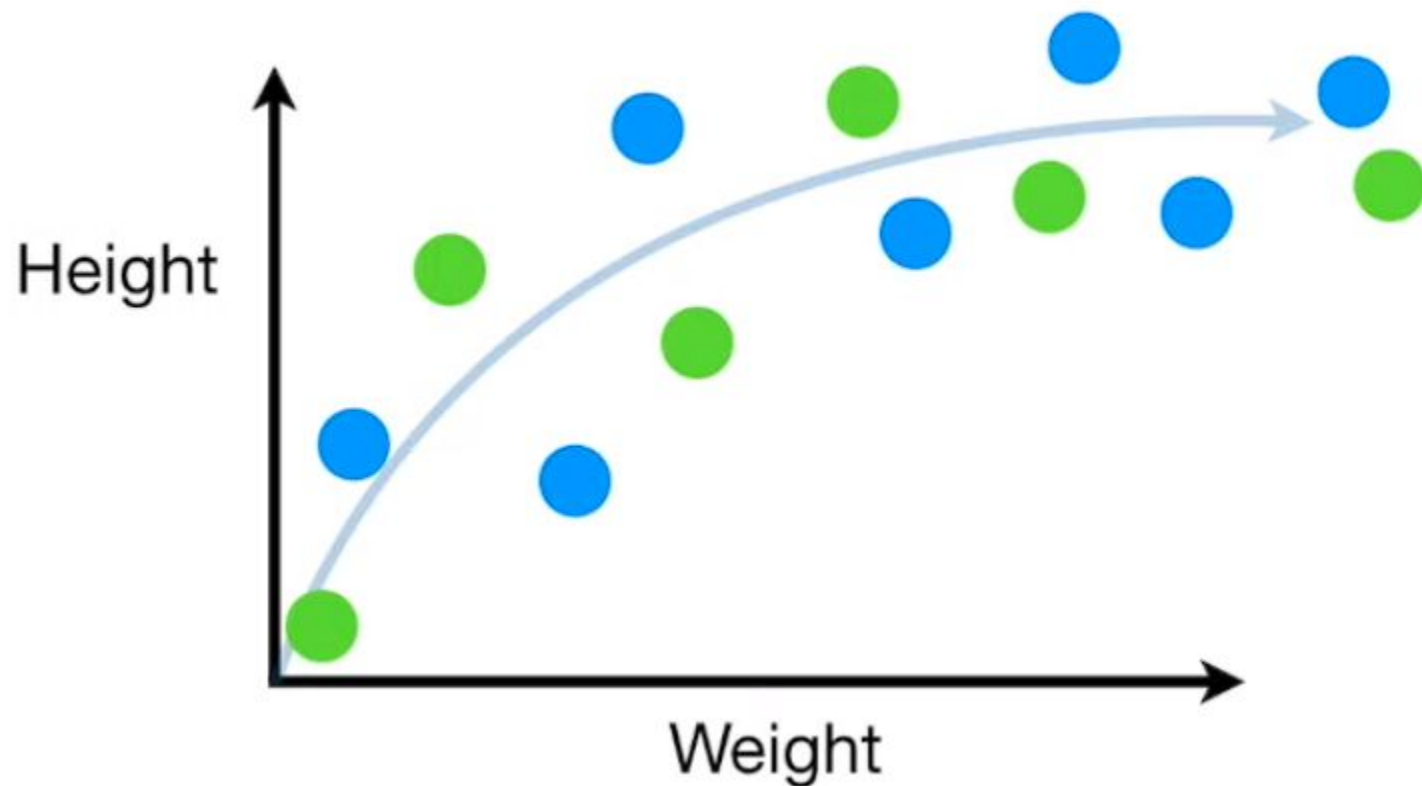
Considerate este dataset.



# Underfitting, Overfitting, Just right

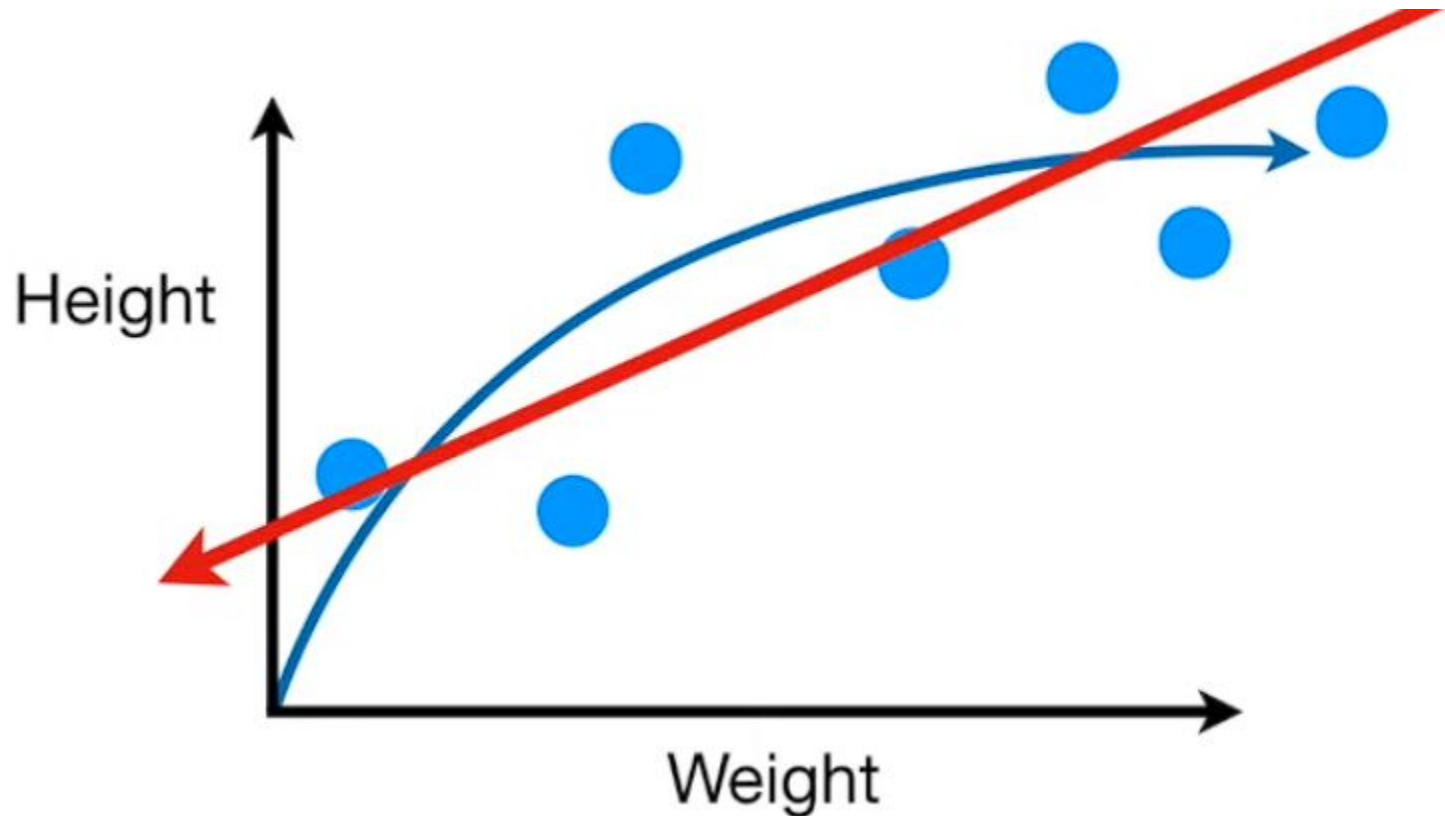
Pontos azuis são dados de treino.

Pontos verdes são dados de teste.



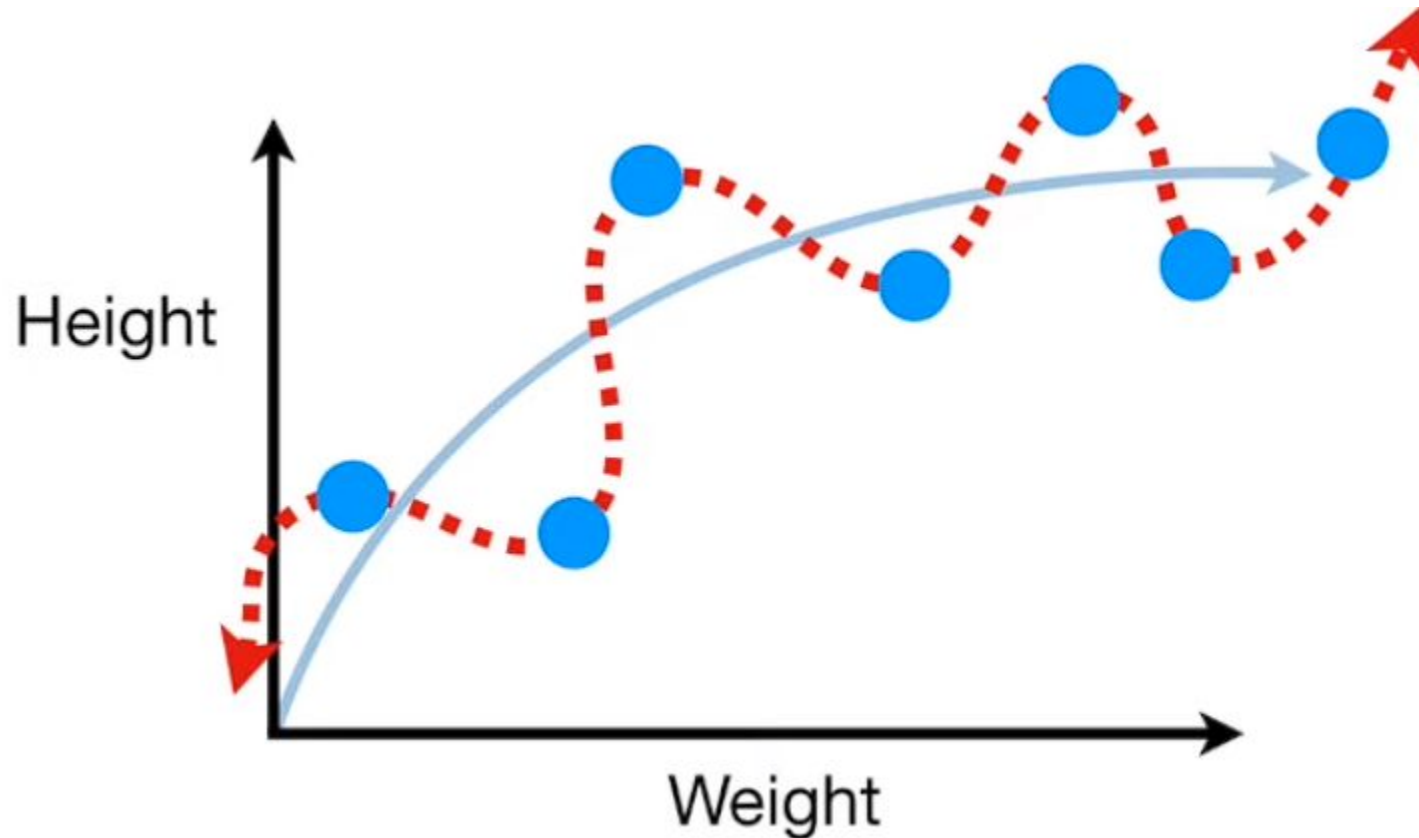
# Underfitting, Overfitting, Just right

Podemos tentar usar um algoritmo **muito simples**, como a regressão linear para modelar o comportamento dos pontos.



# Underfitting, Overfitting, Just right

Podemos tentar usar um algoritmo **complexo**, para modelar o comportamento dos pontos.

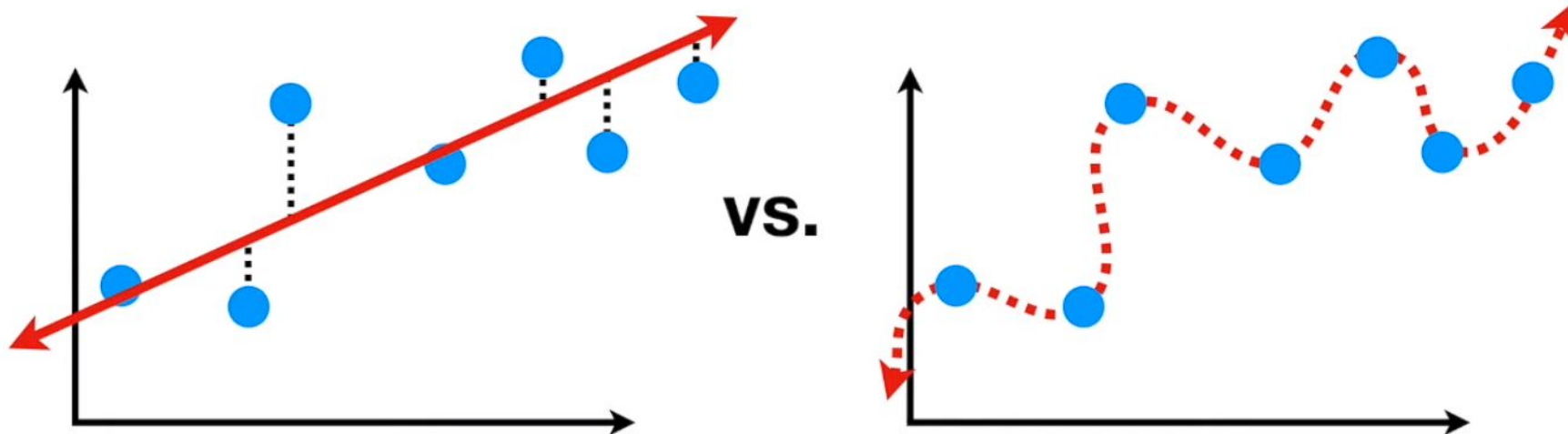


# Underfitting, Overfitting, Just right

Calculamos a performance usando a soma do quadrado das distâncias dos pontos até cada curva.

Para o modelo simples, existe um valor de erro,

Para o modelo complexo, o erro é praticamente zero.

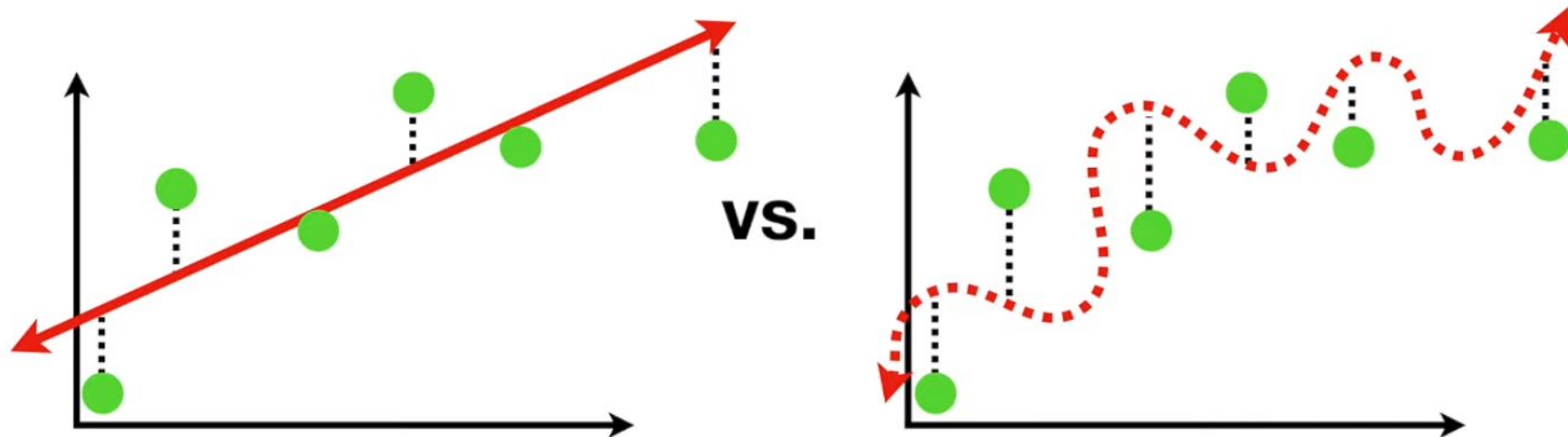




# Underfitting, Overfitting, Just right

Entretanto, ao calcular o valor do erro para o conjunto de teste **o erro para o algoritmo simples é menor do que o erro para o algoritmo complexo.**

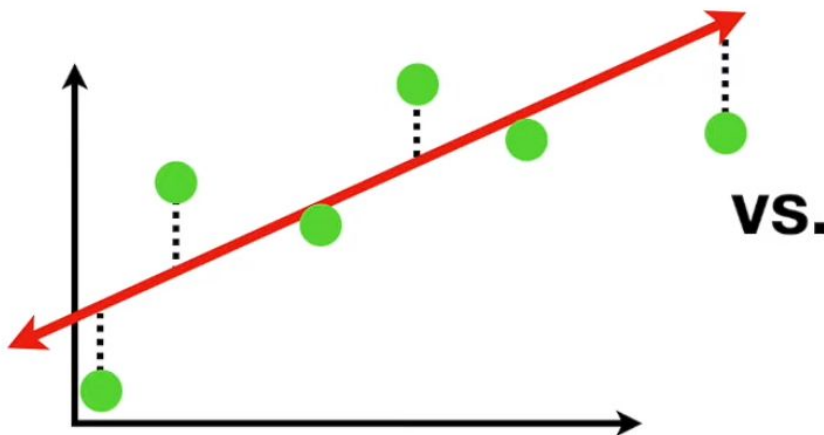
Para diferentes datasets, o modelo **complexo** apresenta maior **variância** no erro, o modelo **simples** apresenta **menor variância**.



# Underfitting, Overfitting, Just right

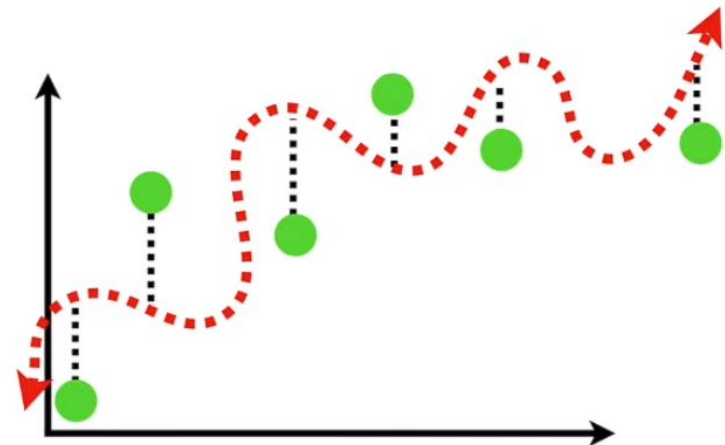
## MODELO SIMPLES

- **High Bias**
- **Low Variance**
- Consegue obter boas previsões, não ótimas, mas consistentes.
- Underfit



## MODELO COMPLEXO

- **Low Bias**
- **High Variance**
- Pode prever bem, ou mal, dependendo do dataset.
- Overfit

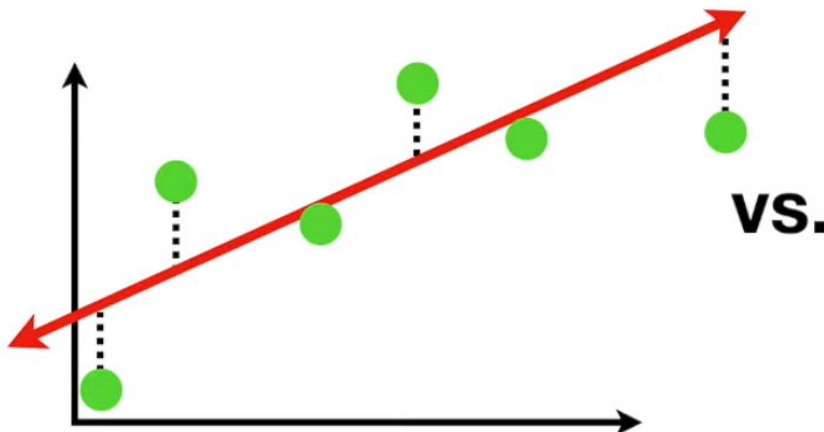


# Bias vs Variance

**Bias:** Erro sistemático introduzido pelo algoritmo:

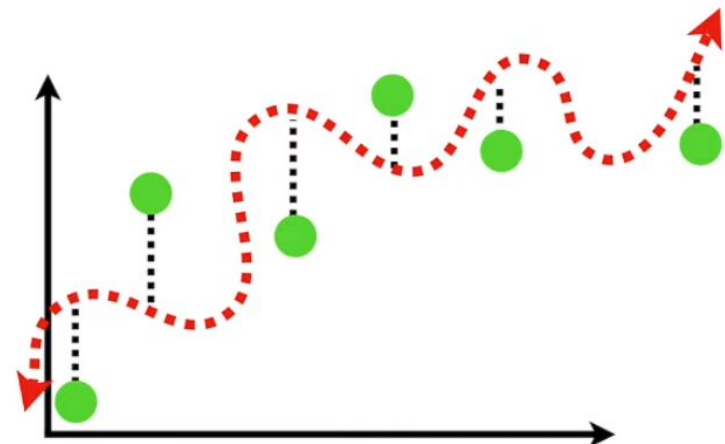
## MODELO SIMPLES

- **High Bias:** Modelo simples demais para capturar variações dos dados, introduzindo um erro sistemático alto. (underfit)



## MODELO COMPLEXO

- **Low Bias:** Modelo mais complexo, capaz de capturar as variações dos dados. Geralmente leva a melhor performance mas pode causar overfitting ao capturar até os ruídos dos dados.

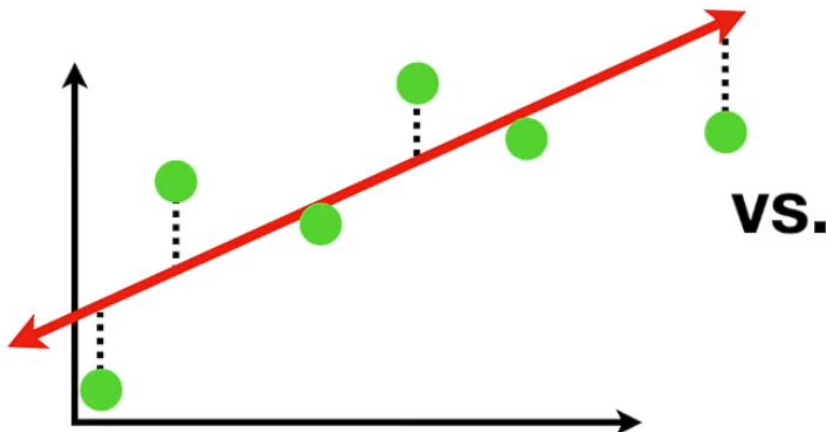


# Bias vs Variance

**Variance:** Variação nas previsões do modelo em relação à média, causada pela sensibilidade do modelo às pequenas mudanças nos dados de treinamento.

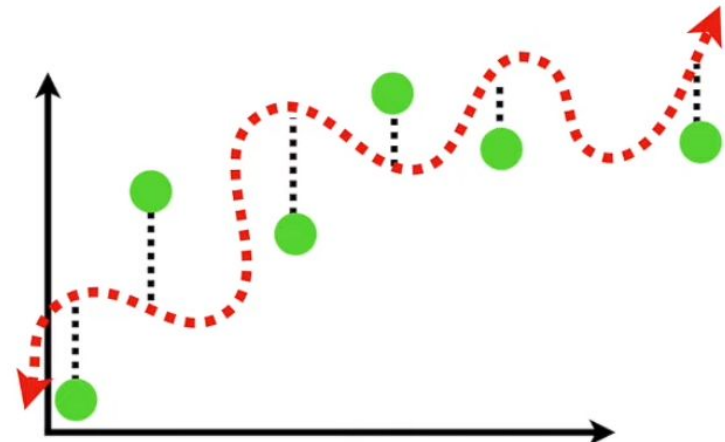
## MODELO SIMPLES

- **Low Variance:** Apesar de apresentar maiores erros, a variação do erro entre diferentes conjuntos de dados é menor.



## MODELO COMPLEXO

- **High Variance:** Apesar de menores erros, a variação do erro entre diferentes conjuntos de dados é maior.



# Underfitting, Overfitting, Just right

**Bias (Viés):** O bias refere-se ao erro introduzido ao simplificar um modelo para se ajustar aos dados. Um modelo com **alto bias** tende a ser muito simples e **não captura as relações subjacentes nos dados**, resultando em **underfitting**, onde o modelo não se ajusta bem nem aos dados de treinamento nem aos de teste.

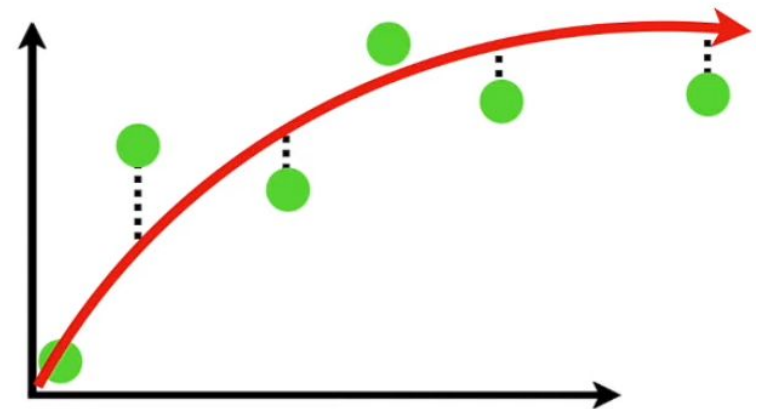
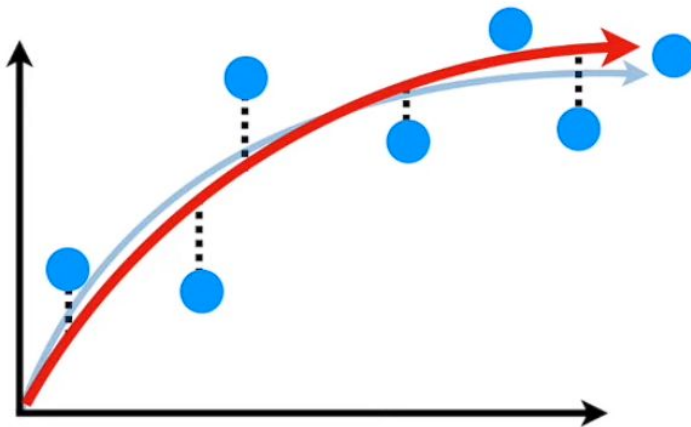
**Variância (Variance):** A variância refere-se à sensibilidade do modelo a pequenas flutuações nos dados de treinamento. Um modelo com alta variância é muito complexo e **se ajusta demais aos dados de treinamento**, incluindo o ruído, o que pode levar ao **overfitting**. Nesse caso, o modelo funciona bem nos dados de treinamento, mas mal em novos dados não vistos, ou seja, possui alta variância para diferentes datasets.

# Bias vs Variance

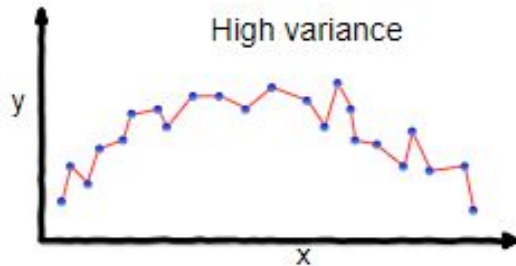
O modelo intermediário entre o mais simples e o mais complexo possui uma boa relação de compromisso (**trade off**) entre variância e bias.

Variância menor que o modelo mais simples.

Bias menor do que o modelo mais complexo.



# Underfitting, Overfitting, Just right

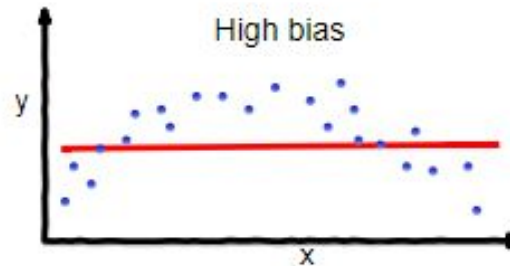


## overfitting

Também chamado de "high variance," ocorre quando um modelo é excessivamente complexo e se ajusta demais aos dados de treinamento, incluindo o ruído.

Um modelo overfit se adapta perfeitamente aos dados de treinamento, mas geralmente tem um desempenho ruim em dados de teste.

Pode ser mitigado com técnicas como regularização e aumento de dados.

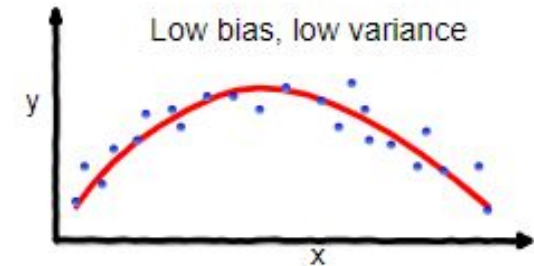


## underfitting

Também conhecido como "high bias," ocorre quando um modelo é muito simples para capturar os padrões nos dados.

Um modelo underfit não se ajusta bem aos dados de treinamento e tende a ter baixo desempenho em dados de teste.

Pode ser causado pela escolha de um modelo muito simples ou falta de dados de treinamento.



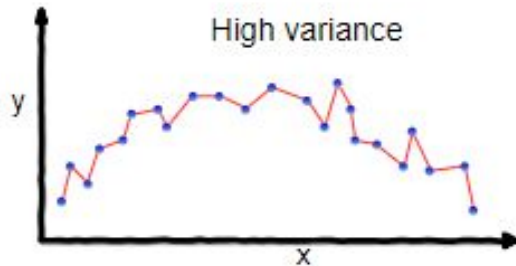
## Good balance

Ocorre quando o modelo captura com precisão as relações nos dados, sem under ou overfitting.

Resulta em um desempenho equilibrado e confiável nos dados de treinamento e teste.

Encontrar um bom ajuste geralmente envolve ajustar a complexidade do modelo e usar técnicas como validação cruzada e regularização para equilibrar bias e variância.

# Underfitting, Overfitting, Just right

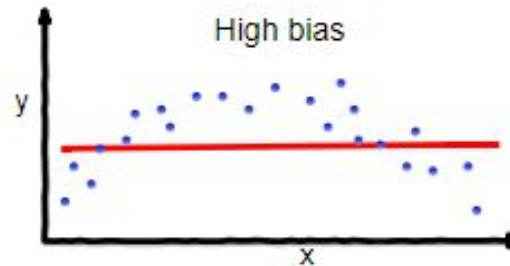


## overfitting

Chamado "high variance", ocorre com modelo excessivamente complexo que se ajusta demais aos dados, incluindo ruído.

Modelo overfit se adapta bem aos dados de treinamento, mas mal em dados de teste.

Mitigado com técnicas como regularização e aumento de dados.

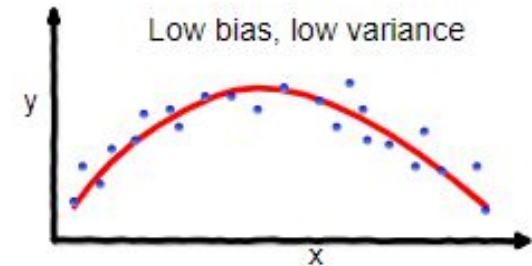


## underfitting

Também chamado "high bias", ocorre quando modelo é muito simples para capturar padrões nos dados.

Modelo underfit não se ajusta bem aos dados de treinamento e tem baixo desempenho em dados de teste.

Causado por escolha de modelo simples ou falta de dados de treinamento.



## Good balance

Ocorre quando o modelo captura com precisão as relações nos dados, sem under ou overfitting.

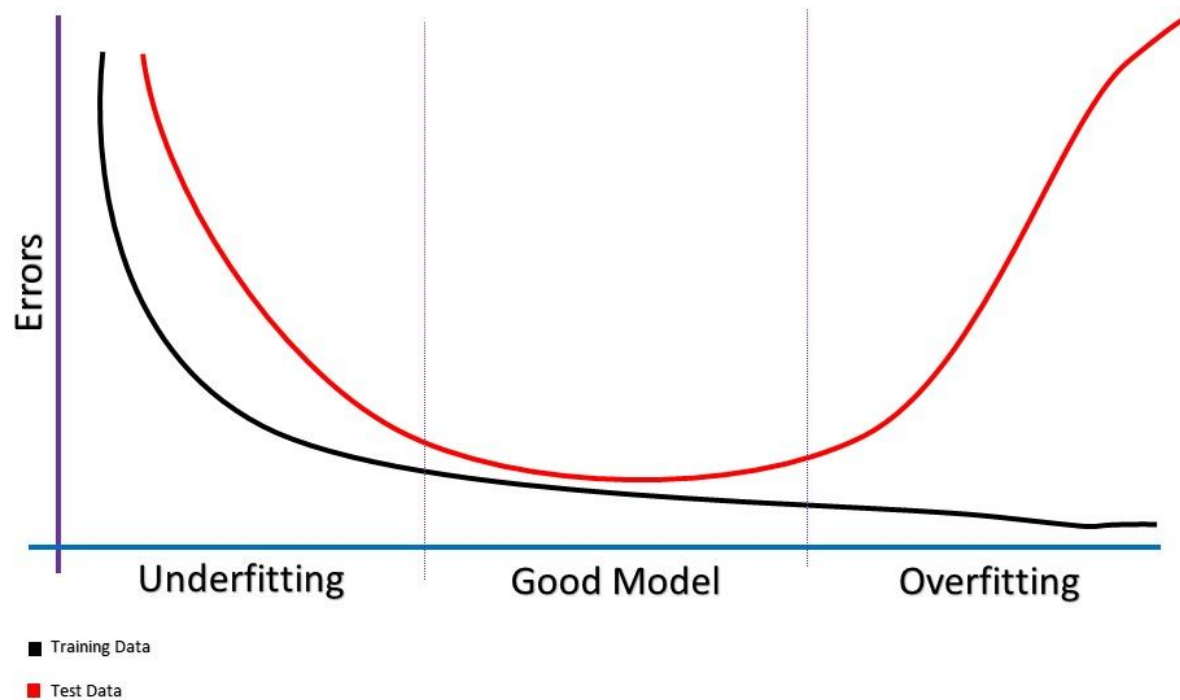
Resulta em desempenho equilibrado e confiável nos dados de treinamento e teste.

Encontrar um bom ajuste envolve ajustar a complexidade do modelo e utilizar técnicas como validação cruzada e regularização para equilibrar bias e variância.



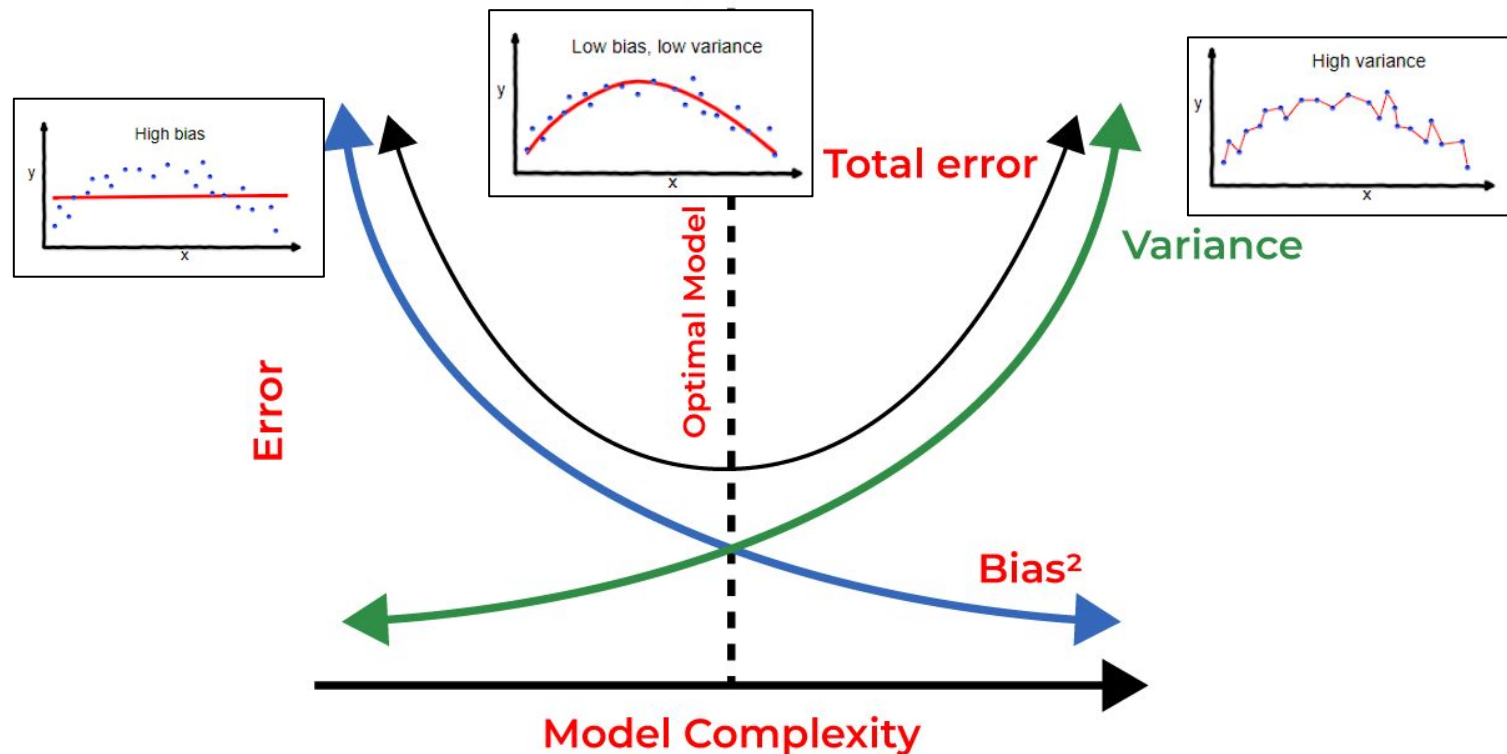
# Underfitting, Overfitting, Just right

O eixo x pode ser visto como complexidade do modelo, ou épocas de treinamento (mais comum para deep learning)



# Underfitting, Overfitting, Just right

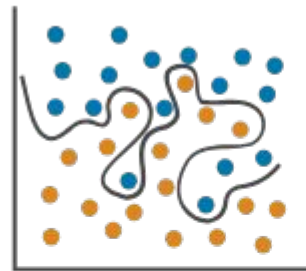
O eixo x pode ser visto como complexidade do modelo, ou épocas de treinamento (mais comum para deep learning)



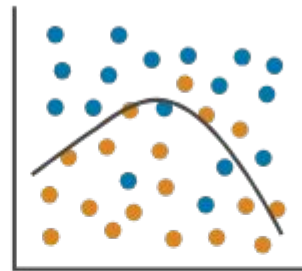
# Underfitting, Overfitting, Just right

Classification

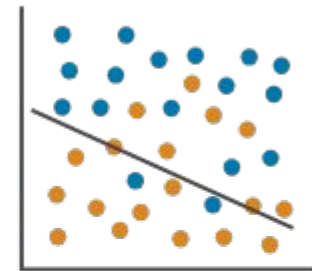
Overfitting



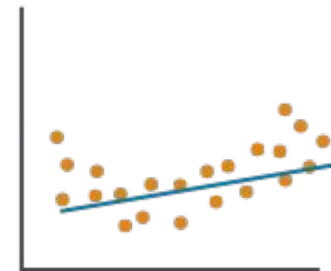
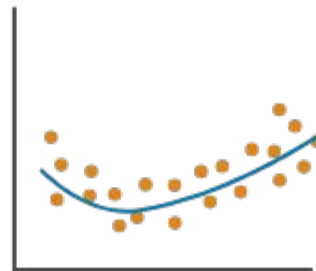
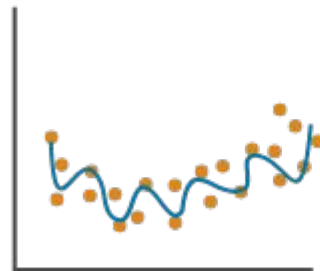
Right Fit



Underfitting



Regression



# Regressão Linear: Underfitting, Overfitting, Just right

## Underfitting na Regressão Linear

- Acontece quando o modelo é **simples demais** para capturar o padrão dos dados.
- Na regressão linear, isso pode ocorrer quando:
  - Você usa **poucas variáveis** (ex: usa apenas  $x_1$ , mas a variável alvo depende de  $x_1$ ,  $x_2$  e  $x_3$ ).
  - A relação real entre as variáveis é **não linear**, e um modelo linear não consegue representar a curva.
- **Sintomas:**
  - **Erro alto no treino e erro alto na validação.**
  - O modelo tem desempenho ruim mesmo nos dados em que foi treinado.
- **Solução:**
  - Usar mais variáveis que podem ajudar a entender melhor a variável de saída.

# Regressão Linear: Underfitting, Overfitting, Just right

## Overfitting na Regressão Linear

- Acontece quando o modelo é **complexo demais** e começa a ajustar o **ruído** dos dados em vez do padrão real.
- Na regressão linear, isso pode ocorrer quando:
  - Você inclui **muitas variáveis**, inclusive irrelevantes.
  - Você **não usa regularização** (como Ridge ou Lasso) em dados com muitas variáveis.
- **Sintomas:**
  - **Erro baixo no treino**, mas **erro alto na validação**.
  - O modelo vai muito bem nos dados de treino, mas generaliza mal para novos dados.
- **Solução:**
  - Usar métodos de regularização L1 (Lasso) ou L2 (Ridge).

# Regularização Lasso (L1)

- Quando há **múltiplas features altamente correlacionadas**, ou seja, que se comportam de forma semelhante, o Lasso tende a:
  - **Selecionar apenas uma** dessas features.
  - **Zerar os coeficientes** das outras, para reduzir a penalização L1.
- Esse comportamento faz com que o Lasso realize **seleção automática de variáveis (feature selection)**:
  - Muitas variáveis terão coeficiente **zero**.
  - Essas variáveis são **ignoradas pelo modelo**.
- Como consequência, o modelo Lasso se torna:
  - **Mais simples e interpretável**.
  - Com **menos variáveis ativas**, o que é uma grande vantagem na análise e explicação dos resultados.

$$\text{RSS}_{\text{lasso}} = \sum_{i=1}^n [y_i - (\mathbf{w} \cdot \mathbf{x}_i + b)]^2, \quad \boxed{+ \alpha \sum_{j=1}^p |w_j|}$$

regularização  $\ell_1$

# Regularização Ridge (L2)

- Na **regularização Ridge (L2)**:
  - A penalização é feita com o **quadrado dos coeficientes**, e não com seus módulos (como no Lasso).
- **Efeito em features altamente correlacionadas**:
  - Podemos ter duas features com **coeficientes parecidos**
  - É improvável que uma feature seja eliminada (coeficiente zero), como acontece no Lasso.
  - A penalização L2 **cresce mais rápido para coeficientes grandes**, então o modelo tende a **distribuir os pesos de forma mais equilibrada** entre as variáveis correlacionadas.
- **Limitações da Ridge**:
  - Não reduz a **sensibilidade a outliers**.
  - Por isso, é **importante limpar o dataset** previamente:
    - Remover outliers.
    - Eliminar variáveis desnecessárias.

Esses cuidados ajudam a melhorar a performance e a interpretabilidade do modelo regularizado com L2.

$$\text{RSS}_{\text{ridge}} = \sum_{i=1}^n [y_i - (\mathbf{w} \cdot \mathbf{x}_i + b)]^2 + \alpha \sum_{j=1}^p w_j^2$$

regularização  $\ell_2$

# Elastic Net – Combinação de L1 e L2

- Combina **Lasso (L1)** e **Ridge (L2)** em um único modelo.
- Traz o **melhor dos dois mundos**:
  - **Seleção de variáveis** do Lasso.
  - **Estabilidade em variáveis correlacionadas** do Ridge.
- A função de custo inclui dois hiperparâmetros:
  - **$\alpha$  (alpha)**: controla o peso total da regularização.
  - **$r$  (l1\_ratio ou mixing parameter)**: controla o equilíbrio entre L1 e L2.
    - $r = 1 \rightarrow$  vira Lasso puro.
    - $r = 0 \rightarrow$  vira Ridge puro.
    - $0 < r < 1 \rightarrow$  combinação das duas.
- Útil quando:
  - Existem **muitas variáveis**, algumas irrelevantes e outras correlacionadas.
  - O Lasso **elimina variáveis demais**.
  - O Ridge **não faz seleção de variáveis**.
- **Desvantagem**:
  - Exige a escolha de **dois hiperparâmetros**  $\rightarrow$  requer **validação cruzada** para encontrar a melhor combinação.
- É uma ótima escolha padrão quando **não se sabe se o Lasso ou o Ridge é mais apropriado**.

$$\text{RSS}_{\text{elasticnet}} = \sum_{i=1}^n [y_i - (\mathbf{w} \cdot \mathbf{x}_i + b)]^2 + \alpha \left( r \sum_{j=1}^p |w_j| + (1 - r) \sum_{j=1}^p w_j^2 \right)$$



# Quando Usar Ridge ou Lasso?

## Lasso (L1) – Seleção de Variáveis

- Quando há **muitas variáveis irrelevantes**.
- Você quer um modelo **mais simples e interpretável**.
- Pode eliminar variáveis automaticamente (**feature selection**).
- Útil quando o número de variáveis é **maior que o número de observações**.

## Ridge (L2) – Todas as Variáveis Importam

- Quando **todas as variáveis são relevantes**, mas podem estar **correlacionadas**.
- Melhor para cenários com **multicolinearidade**.
- Mantém todas as variáveis no modelo (nenhum coeficiente é zerado).
- Útil quando se prioriza **estabilidade e performance preditiva**.

## Dica prática

Se quiser o melhor dos dois mundos, use o **Elastic Net**, que combina L1 e L2!

## O valor do alpha

$$\text{RSS}_{\text{elasticnet}} = \sum_{i=1}^n [y_i - (\mathbf{w} \cdot \mathbf{x}_i + b)]^2 + \alpha \left( r \sum_{j=1}^p |w_j| + (1 - r) \sum_{j=1}^p w_j^2 \right)$$

O que é o alfa ( $\alpha$ )?

- Alfa é o parâmetro de força da regularização.
- Ele controla quanto de penalização será aplicada aos coeficientes do modelo durante o treinamento.
- Aparece na função de custo da Ridge, Lasso e Elastic Net.

## O valor do alpha

$$\text{RSS}_{\text{elasticnet}} = \sum_{i=1}^n [y_i - (\mathbf{w} \cdot \mathbf{x}_i + b)]^2 + \alpha \left( r \sum_{j=1}^p |w_j| + (1 - r) \sum_{j=1}^p w_j^2 \right)$$

- Na regressão linear, buscamos minimizar o erro (a diferença entre os valores previstos e os valores reais).
- A regularização adiciona uma penalização ao modelo quando os coeficientes são muito grandes.
- Alfa controla o quanto nos importamos em manter os coeficientes pequenos versus ajustar bem os dados de treino.

## O valor do alpha

$$\text{RSS}_{\text{elasticnet}} = \sum_{i=1}^n [y_i - (\mathbf{w} \cdot \mathbf{x}_i + b)]^2 + \alpha \left( r \sum_{j=1}^p |w_j| + (1 - r) \sum_{j=1}^p w_j^2 \right)$$

Valor de $\alpha$	O que acontece
$\alpha = 0$	Sem regularização. Regressão linear pura.
$\alpha$ pequeno	Regularização leve. Coeficientes levemente reduzidos.
$\alpha$ grande	Regularização forte. Coeficientes muito reduzidos (alguns zerados no Lasso).
$\alpha$ muito grande	Modelo sofre underfitting — torna-se simples demais.

## O valor do alpha

$$\text{RSS}_{\text{elasticnet}} = \sum_{i=1}^n [y_i - (\mathbf{w} \cdot \mathbf{x}_i + b)]^2 + \alpha \left( r \sum_{j=1}^p |w_j| + (1 - r) \sum_{j=1}^p w_j^2 \right)$$

- Implicações para o negócio
- O alfa equilibra complexidade do modelo vs capacidade de generalização:
  - Alfa muito baixo → risco de overfitting (modelo se ajusta demais ao treino).
  - Alfa muito alto → risco de underfitting (modelo não aprende o suficiente).
- Escolher o alfa ideal é uma decisão crítica:
  - Afeta a precisão.
  - Afeta a interpretabilidade.
  - Afeta a confiabilidade das decisões do modelo (especialmente em áreas reguladas).

## O valor do alpha

$$\text{RSS}_{\text{elasticnet}} = \sum_{i=1}^n [y_i - (\mathbf{w} \cdot \mathbf{x}_i + b)]^2 + \alpha \left( r \sum_{j=1}^p |w_j| + (1 - r) \sum_{j=1}^p w_j^2 \right)$$

Afeta a confiabilidade das decisões do modelo (especialmente em áreas reguladas):

- **Auditoria exige rastreabilidade:** modelos com muitos coeficientes grandes e não interpretáveis dificultam explicações.
- **Regulações como LGPD, GDPR ou Basel II/III** exigem transparência e controle sobre decisões automatizadas.
- Um alfa bem calibrado **ajuda a garantir conformidade** com essas exigências.

## Como escolher o alfa?

Você **não escolhe no chute** — o valor ideal é encontrado por **validação cruzada**:

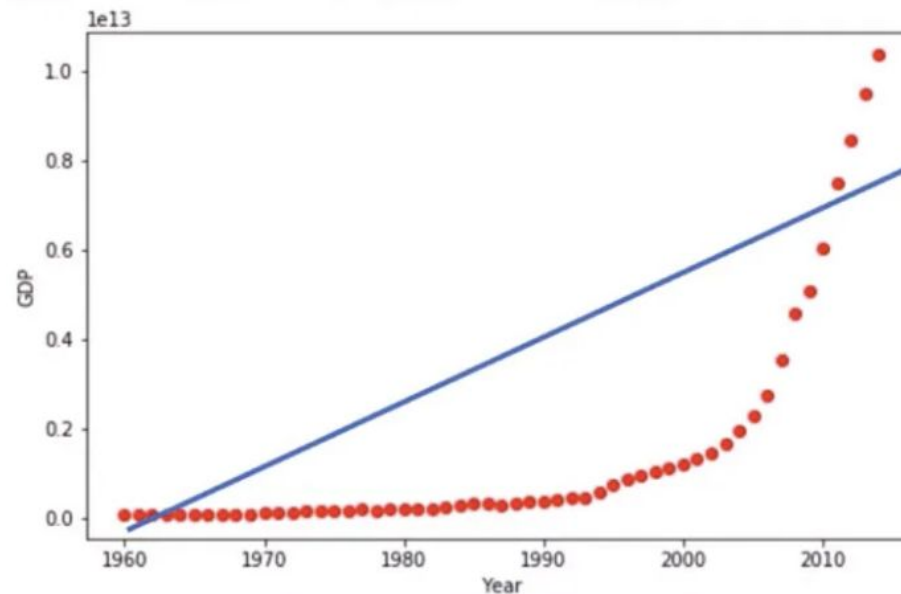
- Teste vários valores (ex: 0.001, 0.01, 0.1, 1, 10...).
- Escolha o que tiver **melhor desempenho em dados não vistos**.

E se as variáveis de entrada não forem lineares?

# Variáveis Não Lineares

Nem tudo é linear...

- Regressão linear assume linearidade nos parâmetros, não necessariamente nas variáveis
- Se a relação não é linear, o modelo linear pode falhar
- Mas ainda podemos usar regressão linear com criatividade!





# Variáveis Não Lineares: Feature Engineering

- Aplicar transformações matemáticas pode "endireitar" a relação
- Exemplos úteis:
  - Curva exponencial  $\rightarrow \log(x)$
  - Curva logarítmica  $\rightarrow \exp(x)$
  - Forma de U  $\rightarrow x^2, x^3$
  - Decaimento  $\rightarrow 1/x$

# Variáveis Não Lineares: Feature Engineering

## Cuidado com overfitting

Mais complexidade = mais risco de overfitting

- Adicionar muitas transformações pode gerar modelos instáveis
- Use validação cruzada para medir o impacto
- Use regularização (Ridge, Lasso) para controlar exageros

# Variáveis Não Lineares: Feature Engineering

## E se nada funcionar?

Se as transformações não resolverem...

- Talvez a regressão linear não seja o melhor modelo
- Considere modelos não lineares:
  - Árvores de decisão / Random Forest
  - Gradient Boosting
  - SVM com kernel
  - Redes neurais

Dúvidas?

# Avaliação



[Link para o Formulário](https://forms.gle/1o4RWqi1xYAY8pfp7)

<https://forms.gle/1o4RWqi1xYAY8pfp7>

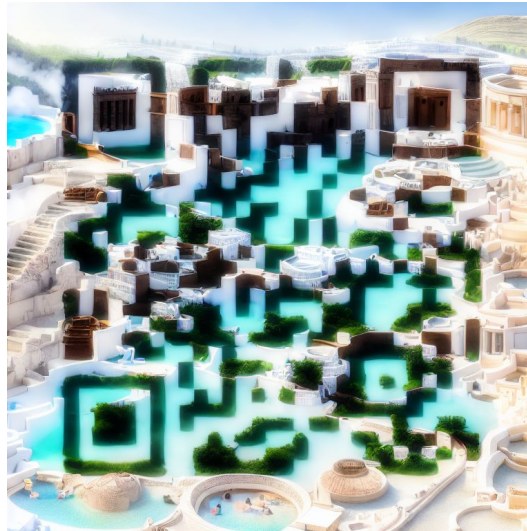
Qual é a aula que está avaliando? \*

- ☐ Aula 1
- ☐ Aula 2
- ☐ Aula 3
- ☐ Aula 4
- ☐ Aula 5



[openart](#)

“deep space as seen from the Hubble telescope”



[huggingface](#)

“Sky view of highly aesthetic, ancient greek thermal baths in beautiful nature”

