ICP363 - Introdução ao Aprendizado de Máquina

Avaliação de Modelos: Bias-Variance, Reamostragem e Métricas

Paulo Mann

Slides adaptados do material original do Prof. João C. P. da Silva

14 de maio de 2026

Introdução

- Dada uma base de dados, existem diversos métodos de aprendizado de máquina que podemos utilizar para construir um modelo que possa prever/classificar novos dados.
- Em princípio, não há um método que possa ser considerado melhor que todos os outros, pois o desempenho de cada um deles pode variar de acordo com a base de dados que temos a nossa disposição.
- Por isso, é importante termos critérios de avaliação robustos para determinar qual modelo vamos utilizar.

Regressão Linear

- Considere que temos:
  - um vetor de entrada com p *features*, $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^p$,
  - um *target* (variável de saída) $y \in \mathcal{Y} \subseteq \mathbb{R}$,
  - uma base de dados $\mathcal{D} = \{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \cdots, (\mathbf{x}^{(n)}, y^{(n)})\}$ com n observações $(\mathbf{x}^{(i)}, y^{(i)})$ amostradas de forma i.i.d.

  Regressão Linear: modelo da família $f(\mathbf{x} \mid \boldsymbol{\theta}) = \boldsymbol{\theta}^\top \mathbf{x}$ que examina a *relação linear* entre features e target.

  Quando uma (ou mais) feature cresce (ou decresce), o target y tende a crescer (ou decrescer) proporcionalmente.

- **Exemplos**
  - Peso $\rightarrow$ Pressão Sanguínea
  - Anúncios $\rightarrow$ Vendas
  - Horas de Estudo $\rightarrow$ Nota

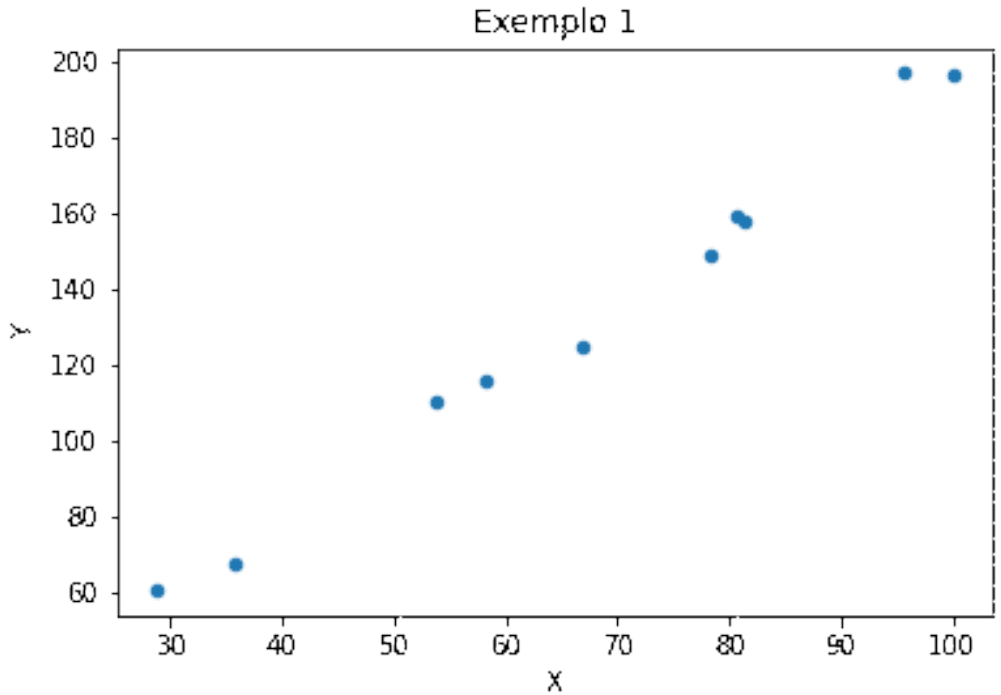
Regressão Linear

- **Objetivo**: determinar a função $\hat{f}$, que dado o valor de entrada x' preveja o valor de saída $y' = \hat{f}(x')$.

X	Y
95.724162408	197.179636092
35.7576189281	67.5906695414
28.8168474238	60.8541328206
99.9584813087	196.907396981
66.8097483121	125.311128524
58.2156926413	115.785784589
53.8210763379	110.762772705
81.2960821704	157.98528569
80.6486970595	159.61941373
78.2528136925	149.003865539

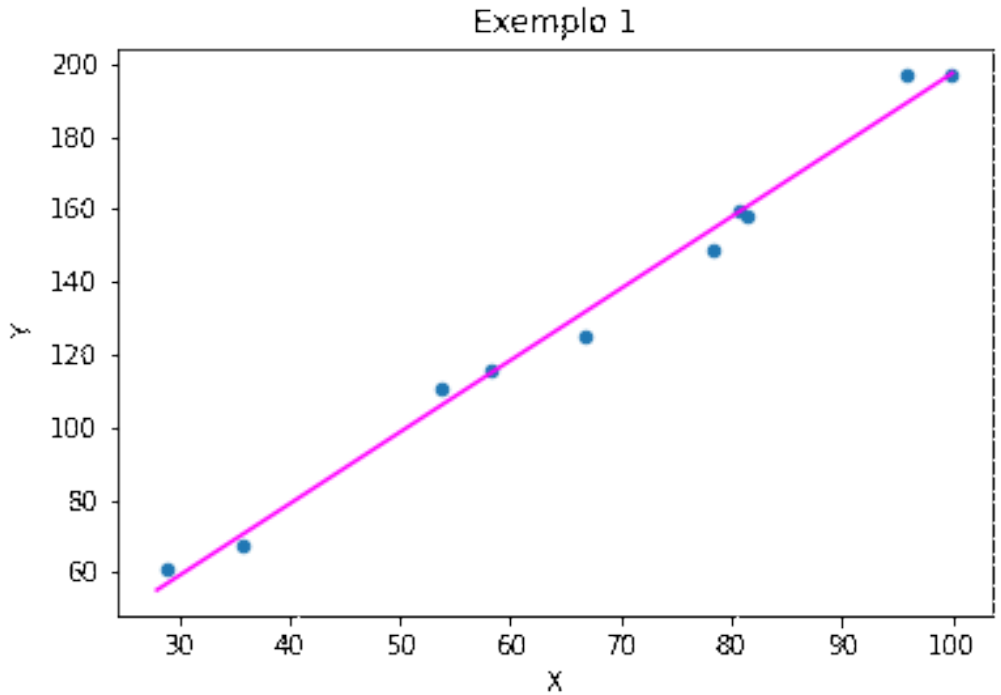
Regressão Linear

- **Objetivo**: determinar a função $\hat{f}$, que dado o valor de entrada x' preveja o valor de saída $y' = \hat{f}(x')$.



Regressão Linear

- **Objetivo**: determinar a função $\hat{f}$, que dado o valor de entrada x' preveja o valor de saída $y' = \hat{f}(x')$.



Escolha do Modelo

Objetivo: determinar a função $\hat{f}$, dado o valor de entrada $\mathbf{x}'$ preveja o valor de saída $y' = \hat{f}(\mathbf{x}')$.

- **Medida**: *Erro Quadrático Médio (Mean Squared Error - MSE)*

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} - \hat{f}(\mathbf{x}^{(i)})\right)^2$$

  Quanto menor o valor de MSE, melhor a previsão que está sendo feita.

- Observe que este erro está sendo medido com relação aos pares $(\mathbf{x}^{(i)}, y^{(i)})$ que **conhecemos**.

- Dizemos que o conjunto de dados (**conhecidos**) $\mathcal{D} = \{(\mathbf{x}^{(1)}, y^{(1)}), \cdots, (\mathbf{x}^{(n)}, y^{(n)})\}$ é o nosso **conjunto de treinamento**.

- E vamos representar seu erro quadrático médio como MSE_{train}.

Escolha do Modelo

Objetivo: determinar a função $\hat{f}$, dado o valor de entrada $\mathbf{x}'$ preveja o valor de saída $y' = \hat{f}(\mathbf{x}')$.

- Mas estamos interessados em ter um erro pequeno com relação aos dados que **não conhecemos**.
- Então, se tivermos um **conjunto de teste** $\mathcal{D}_{\text{test}} = \{(\mathbf{x}^{(t_1)}, y^{(t_1)}), \cdots, (\mathbf{x}^{(t_m)}, y^{(t_m)})\}$, que não foi usado no treinamento, podemos avaliar nosso modelo da seguinte forma:

$$MSE_{\text{test}} = \frac{1}{m} \sum_{i=1}^{m} \left(y^{(t_i)} - \hat{f}(\mathbf{x}^{(t_i)}) \right)^2$$

- Portanto, se tivermos um conjunto de teste a nossa disposição, podemos escolher o modelo que tenha o menor valor de MSE_{test}.
- Caso contrário, escolhemos o modelo que possua o menor valor de MSE_{train}?

Escolha do Modelo

Caso contrário, escolhemos o modelo que possua o menor valor de MSE_{train}? MSE_{train} pequeno não implica em MSE_{test} pequeno

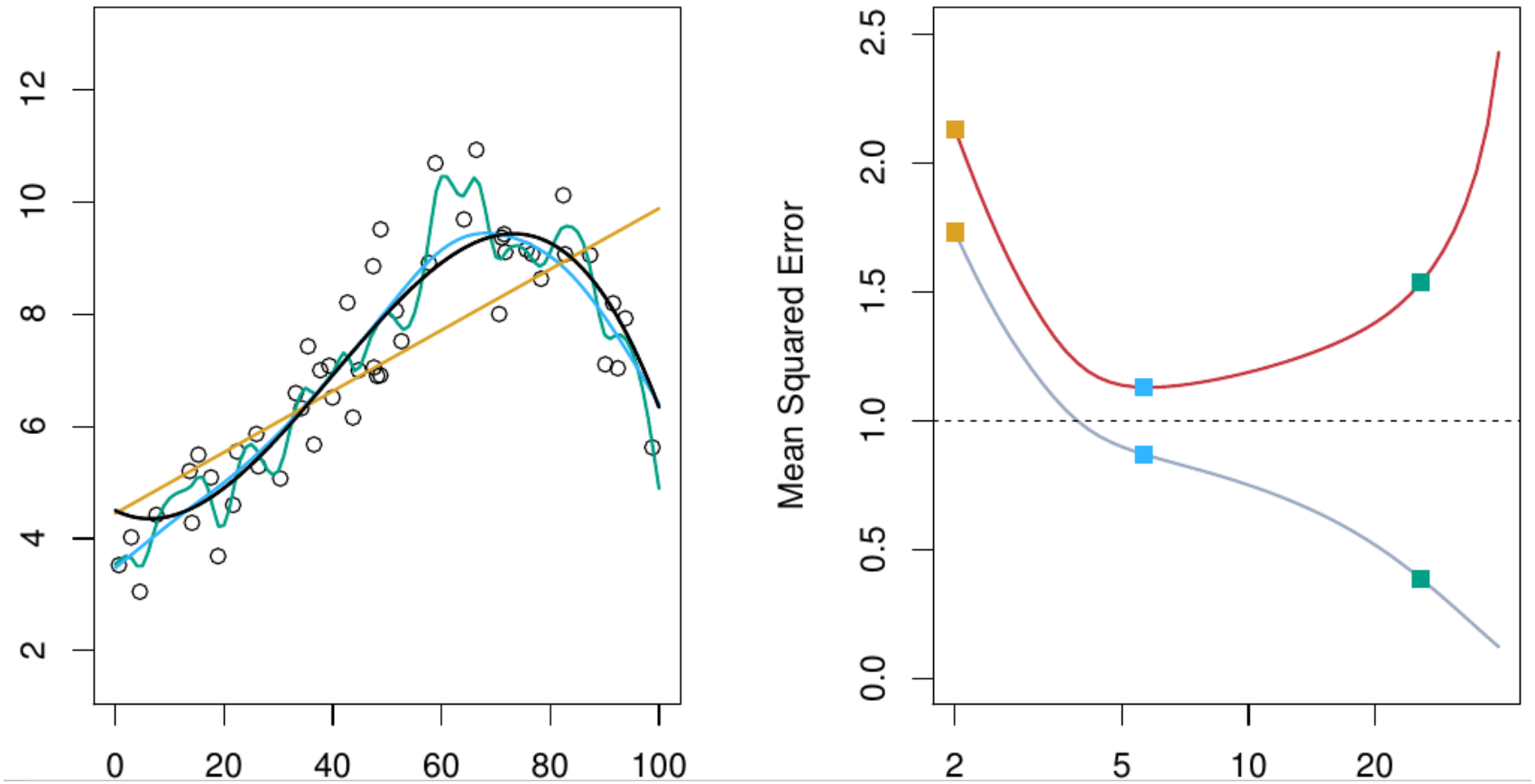


Figura: Fonte: An Introduction to Statistical Learning-James,G. et. al. (2021)

Escolha do Modelo

No gráfico à direita, MSE_{train} cai à medida que o modelo vai se adaptando aos dados, enquanto MSE_{test} começa a piorar em um dado momento - **overfitting**.

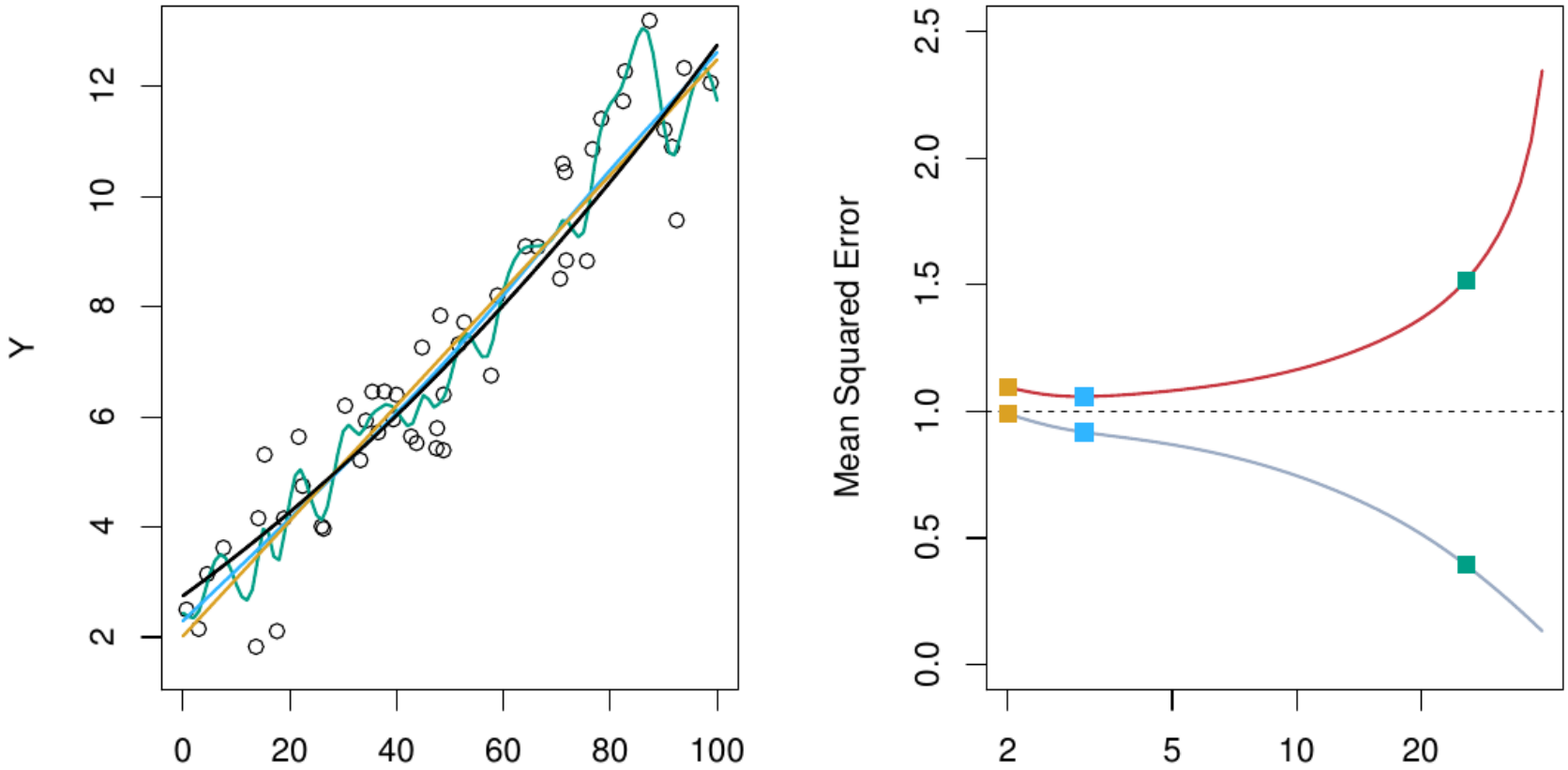


Figura: Fonte: An Introduction to Statistical Learning-James,G. et. al. (2021)

Escolha do Modelo

No gráfico à direita, MSE_{train} cai à medida que o modelo vai se adaptando aos dados, enquanto MSE_{test} começa a piorar em um dado momento - **overfitting**.

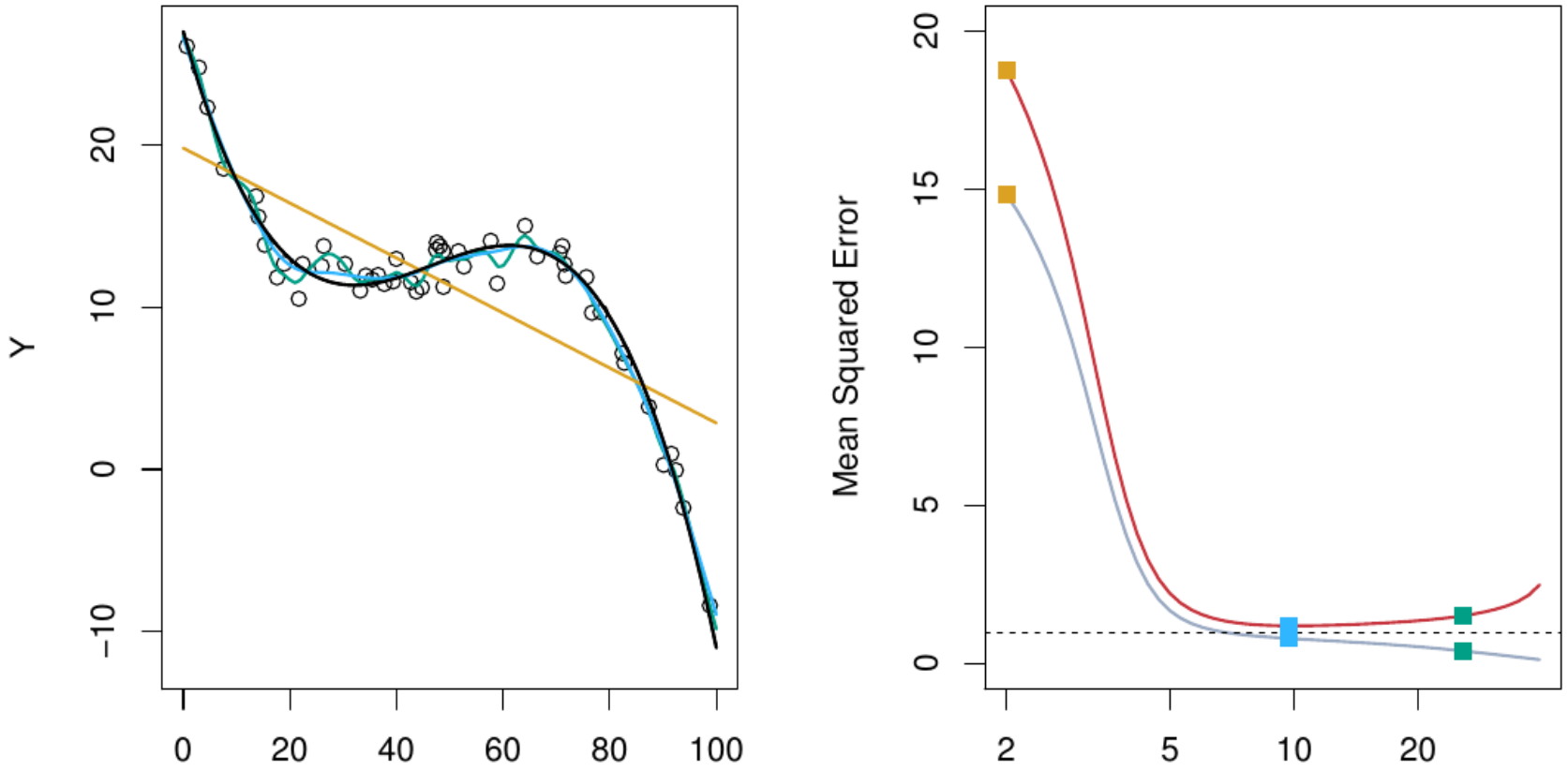


Figura: Fonte: An Introduction to Statistical Learning-James,G. et. al. (2021)

The Bias-Variance Trade-Off

- Para um dado ponto $\mathbf{x}_0$, o valor esperado de MSE_{test} é composto por:

$$\mathbb{E}\Big[(y_0 - \hat{f}(\mathbf{x}_0))^2\Big] = \text{Var}(\hat{f}(\mathbf{x}_0)) + \Big[\text{Bias}(\hat{f}(\mathbf{x}_0))\Big]^2 + \text{Var}(\varepsilon)$$

  - **Variância de $\hat{f}(\mathbf{x}_0)$:** quantidade pela qual $\hat{f}$ mudaria se utilizássemos um conjunto de treinamento diferente. Se ela for alta, pequenas mudanças no conjunto de treinamento implicam grandes mudanças em $\hat{f}$. Quanto mais *flexível* o método, maior a variância.
  - **Viés ao quadrado de $\hat{f}(\mathbf{x}_0)$**: erro introduzido ao modelarmos um problema muito complexo usando um modelo simples. Neste caso, métodos mais *flexíveis* resultam em um viés menor.
  - **Variância do erro ε**: erro irredutível, uma vez que por mais perfeito que seja nosso modelo, ele sempre terá algum erro embutido (vindo de variáveis desconhecidas).

- Métodos mais flexíveis $\Rightarrow$ variância cresce e viés decresce.
- **Bias-Variance Trade-Off**: à medida que variância e viés mudam, impactam o crescimento ou a diminuição do MSE_{test}. O desafio é encontrar um método em que ambas (variância e viés) sejam baixos.

The Bias-Variance Trade-Off

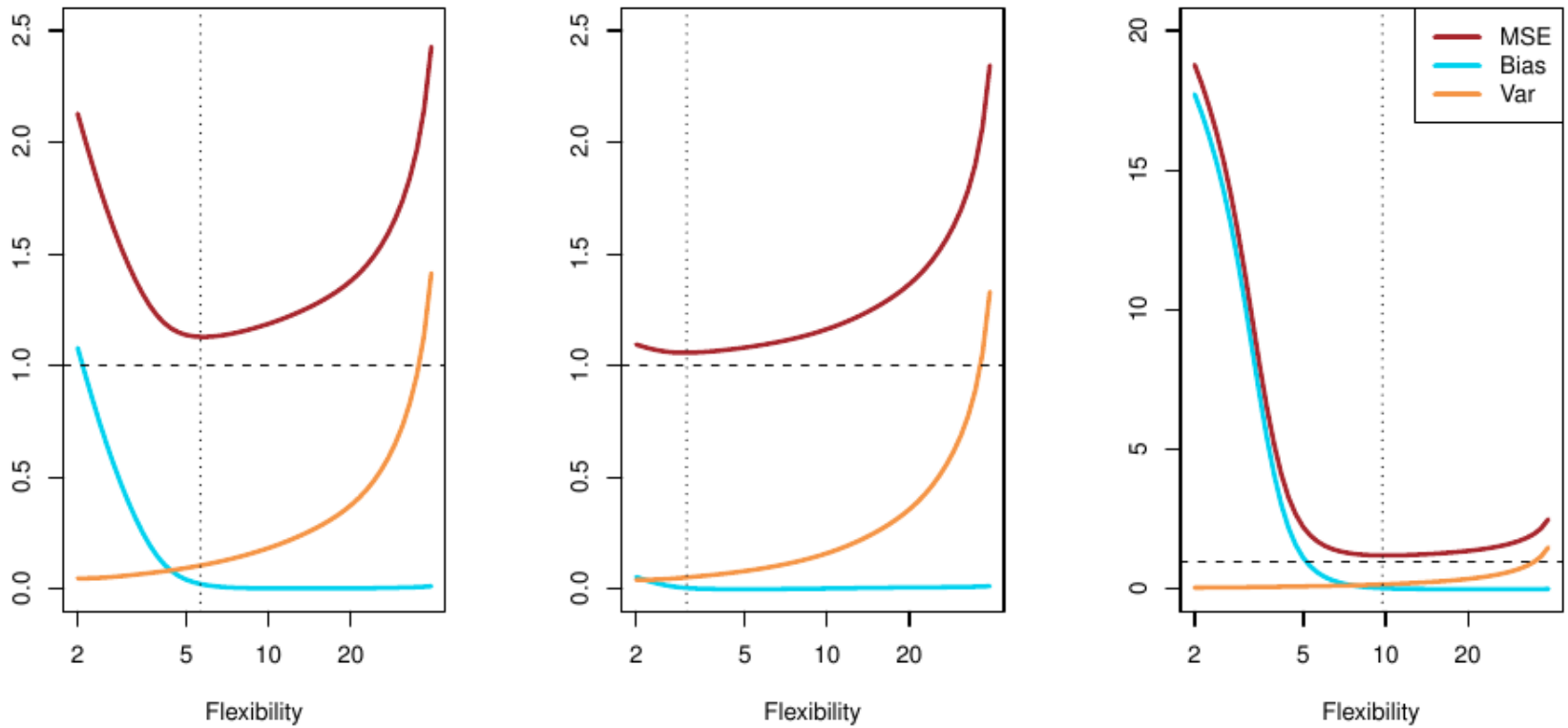


Figura: Fonte: An Introduction to Statistical Learning-James,G. et. al. (2021)

The Bias-Variance Trade-Off

- Para um bom desempenho do modelo em um conjunto de testes, o método de aprendizado deve ter baixa variância e baixo viés.
- Fácil obter baixo viés e alta variância ou alto viés e baixa variância.
- O desafio é encontrar um método que tenha baixo viés e variância.
- Isso não é possível uma vez que normalmente f é desconhecido.

Classificação

- Considere que temos:
  - um vetor de *features* $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^p$,
  - um *target* **qualitativo** $y \in \mathcal{Y} = \{1, 2, \cdots, g\}$ (g classes possíveis),
  - uma base de dados $\mathcal{D} = \{(\mathbf{x}^{(1)}, y^{(1)}), \cdots, (\mathbf{x}^{(n)}, y^{(n)})\}$.
- Podemos estimar o **erro** do classificador $\hat{f}$ determinando a proporção de classificações erradas no conjunto de treinamento (*misclassification error*):

$$\text{erro} = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}\left(y^{(i)} \neq \hat{y}^{(i)}\right), \qquad \hat{y}^{(i)} = \hat{f}(\mathbf{x}^{(i)})$$

com

$$\mathbb{I}\left(y^{(i)} \neq \hat{y}^{(i)}\right) = \begin{cases} 1 & \text{se } y^{(i)} \neq \hat{y}^{(i)} \\ 0 & \text{se } y^{(i)} = \hat{y}^{(i)} \end{cases}$$

- A **accuracy** é o complemento desse erro: accuracy $= 1 -$ erro de classificação. Mas accuracy isolada nem sempre conta a história toda — vamos aos detalhes a seguir.

Matriz de Confusão

Em classificação binária (com convenção $y = 1$ positivo, $y = 0$ negativo), comparamos a predição $\hat{y}^{(i)}$ com o rótulo verdadeiro $y^{(i)}$ em quatro categorias:

	$\hat{y} = 1$ (predito positivo)	$\hat{y} = 0$ (predito negativo)
$y = 1$ (verdadeiro positivo)	**TP** (Verdadeiro Positivo)	**FN** (Falso Negativo)
$y = 0$ (verdadeiro negativo)	**FP** (Falso Positivo)	**TN** (Verdadeiro Negativo)

- **TP**: o modelo previu positivo e o exemplo era de fato positivo.
- **TN**: o modelo previu negativo e o exemplo era de fato negativo.
- **FP**: previu positivo, mas o exemplo era negativo (erro tipo I, "falso alarme").
- **FN**: previu negativo, mas o exemplo era positivo (erro tipo II, "deixou passar").

Exemplo: Matriz de Confusão no Titanic

Suponha que treinamos um classificador para prever a sobrevivência de passageiros do Titanic. Para um conjunto de teste de 12 passageiros, comparamos a verdade $y^{(i)}$ com a predição $\hat{y}^{(i)}$ (positivo = sobreviveu):

ID	Classe	Sexo	Idade	$y^{(i)}$	$\hat{y}^{(i)}$	Tipo
1	1ª	F	25	sim	sim	TP
2	1ª	M	40	não	sim	FP
3	2ª	F	30	sim	sim	TP
4	3ª	M	22	não	não	TN
5	3ª	F	18	sim	sim	TP
6	1ª	M	35	sim	sim	TP
7	2ª	M	50	não	não	TN
8	3ª	M	28	não	não	TN
9	1ª	F	45	sim	sim	TP
10	2ª	F	32	sim	não	FN
11	3ª	M	19	não	não	TN
12	3ª	F	23	sim	não	FN

Contando as 12 linhas: $TP = 5$ (IDs 1, 3, 5, 6, 9), $FP = 1$ (ID 2), $FN = 2$ (IDs 10, 12), $TN = 4$ (IDs 4, 7, 8, 11). A matriz de confusão resultante:

	$\hat{y} = \text{sim}$	$\hat{y} = \text{não}$
$y = \text{sim}$	$TP = 5$	$FN = 2$
$y = \text{não}$	$FP = 1$	$TN = 4$

Acurácia (*accuracy*)

A métrica mais intuitiva é a *accuracy*: proporção de predições corretas em relação ao total.

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

- Equivale a $1 - \frac{1}{n}\sum_{i=1}^{n} \mathbb{I}\left(y^{(i)} \neq \hat{y}^{(i)}\right)$, ou seja, o complemento do *misclassification error* do slide anterior.
- **Cuidado com classes desbalanceadas**: suponha que 99% das amostras são da classe negativa. Um classificador que sempre prevê negativo — sem usar feature nenhuma — tem accuracy de 99%, mas *não detecta nenhum positivo*.
- Em problemas reais (fraude bancária, doenças raras, detecção de spam) accuracy isoladamente esconde o que importa: **como o modelo se sai em cada uma das classes separadamente**.
- Para isso precisamos olhar de forma assimétrica para TP, FP, FN, TN
  $\Rightarrow$ entram *precision* e *recall*.

Precision e Recall

Duas métricas que decompõem o desempenho por tipo de erro:

$$\text{precision} = \frac{TP}{TP + FP} \qquad \text{recall} = \frac{TP}{TP + FN}$$

- **Precision** (precisão) responde: *"dos exemplos que classifiquei como positivos, quantos eram de fato positivos?"*
  Alta precision $\Leftrightarrow$ poucos falsos positivos.
- **Recall** (sensibilidade, *true positive rate*) responde: *"dos exemplos verdadeiramente positivos, quantos consegui detectar?"*
  Alta recall $\Leftrightarrow$ poucos falsos negativos.
- **Trade-off entre precision e recall**: variar o limiar de decisão α (no slide de regressão logística vimos $\hat{y} = 1$ se $\pi(\mathbf{x}) \geq \alpha$) move um contra o outro. α alto $\Rightarrow$ precision alta, recall baixa; α baixo $\Rightarrow$ o oposto.
- **Qual priorizar?**
  - Filtro de *spam*: alta precision (não marcar e-mail legítimo como spam).
  - Diagnóstico de doença grave: alta recall (não deixar de detectar caso positivo).
- **F1-score**: média harmônica das duas, $F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$. Resumo único quando precisamos equilibrar.

Exemplo: Precision e Recall no Titanic

Reaproveitando a matriz de confusão dos 12 passageiros do exemplo anterior:

	$\hat{y} = \text{sim}$	$\hat{y} = \text{não}$
$y = \text{sim}$	$TP = 5$	$FN = 2$
$y = \text{não}$	$FP = 1$	$TN = 4$

- **Accuracy** $= \dfrac{TP + TN}{n} = \dfrac{5+4}{12} = 0{,}75$
  Acertamos 75% dos passageiros.
- **Precision** $= \dfrac{TP}{TP + FP} = \dfrac{5}{5+1} = \dfrac{5}{6} \approx 0{,}833$
  "Dos 6 passageiros que classifiquei como sobreviventes, 5 sobreviveram de fato."
- **Recall** $= \dfrac{TP}{TP + FN} = \dfrac{5}{5+2} = \dfrac{5}{7} \approx 0{,}714$
  "Dos 7 sobreviventes verdadeiros, consegui detectar 5; perdi 2 (FN)."
- **F1-score** $= 2 \cdot \dfrac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = \dfrac{10}{13} \approx 0{,}769$
  Pesa as duas igualmente quando ambas importam.

Conjuntos de Treinamento e Teste

- Não devemos usar toda a nossa base de dados para treinar nosso modelo.
- Precisamos que alguns dados não sejam usados na fase de construção do modelo, para que possamos ajustar os parâmetros do modelo e avaliar o seu desempenho para dados que ele não conhece. Assim, tentamos evitar que nosso modelo fique enviesado.
- Nosso conjunto de dados deve ser dividido em um conjunto usado para treinar nosso modelo (*conjunto de treinamento*) e outro usado para avaliar a performance do modelo (*conjunto de teste ou validação*).
- Implicitamente estamos assumindo que temos um conjunto de dados grande o suficiente para ser considerado como representativo do domínio do nosso problema.
- Se nosso dataset for pequeno (tiver poucas amostras), teremos um conjunto de treinamento pequeno, o que pode impedir a construção do nosso modelo.

Conjuntos de Treinamento e Teste

- Nosso conjunto de dados pode ser dividido em $\frac{2}{3}$ para treino e $\frac{1}{3}$ para teste ou 80% para treino e 20% para teste.
- A divisão dos conjuntos deve ser feita de forma aleatória.
- No caso de problemas de classificação, nosso conjunto de dados pode estar desbalanceado com relação as classes (ou seja, podemos ter mais exemplos de uma classe do que de outra). É desejável que ao dividir nosso conjunto de dados em treinamento e teste, a mesma proporção de exemplos por classe seja preservada em ambos os conjuntos.
- Um problema que esta abordagem apresenta é que usar um único conjunto de teste pode ser insuficiente para se ter uma boa avaliação.
- Para contornar este problema utiliza-se métodos de reamostragem que em geral produzem estimativas de performance melhor do que usar um único conjunto de teste, uma vez que são avaliados diversas versões alternativas dos dados.

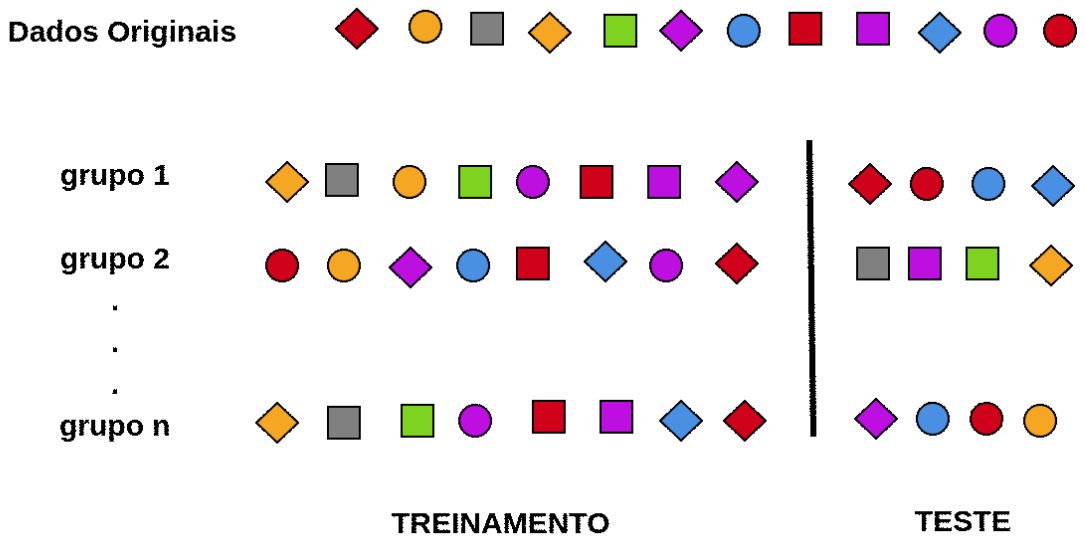
Treinamento, Validação e Teste

- Até aqui falamos de dois conjuntos — mas, se quisermos **selecionar hiperparâmetros** (grau do polinômio, K do KNN, λ de regularização, etc.), avaliar diversas escolhas no conjunto de teste **contamina** esse conjunto: ele deixa de ser uma estimativa honesta do erro em dados novos.
- A solução padrão é dividir os dados em **três** subconjuntos:
  - **Treinamento** ($\mathcal{D}_{\text{train}}$): ajusta os parâmetros $\boldsymbol{\theta}$ do modelo.
  - **Validação** ($\mathcal{D}_{\text{val}}$): compara o desempenho entre diferentes hiperparâmetros e escolhe a melhor configuração.
  - **Teste** ($\mathcal{D}_{\text{test}}$): usado **uma única vez**, ao final, para reportar uma estimativa não enviesada do erro de generalização.
- Proporções típicas: 60%/20%/20%, ou 70%/15%/15%.
- Regra de ouro: o conjunto de teste *só pode ser olhado uma vez*. Se você usar o teste para decidir o que ajustar no modelo, ele vira validação — e você fica sem teste honesto.
- Em datasets pequenos, dividir em três pode deixar cada subconjunto pequeno demais. Para esses casos, a alternativa é **cross-validation** dentro do conjunto de treinamento (próximos slides).

Conjuntos de Treinamento e Teste

Técnicas de reamostragem: o processo de seleção de amostras para o conjunto de treinamento e para o conjunto de teste é repetido várias vezes.

- **Repetição de Divisões em Treinamento e Teste**: esta abordagem simplesmente repete a criação de conjuntos de treinamento e teste várias vezes.



- O número de repetições é importante. Quanto mais amostras o conjunto de teste tiver, maior deverá ser o número de repetições, de modo a diminuir a incerteza da estimativa de performance. Caso o conjunto de dados seja pequeno, a variância de desempenho pode ser grande.

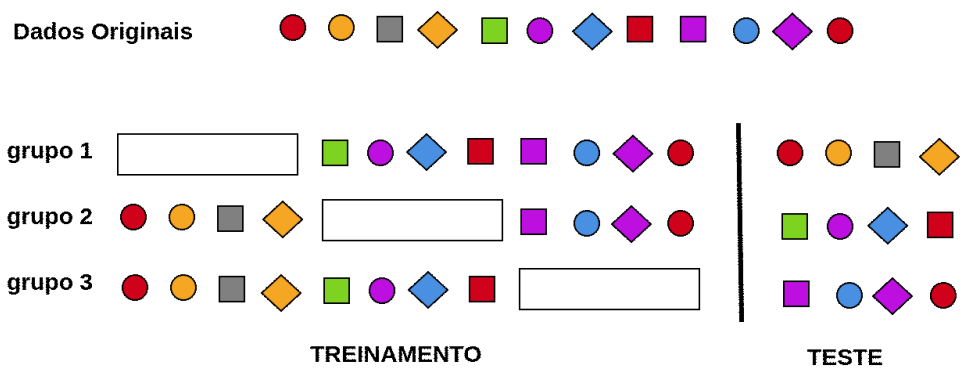
Conjuntos de Treinamento e Teste

Técnicas de reamostragem: o processo de seleção de amostras para o conjunto de treinamento e para o conjunto de teste é repetido várias vezes.

- **K-Fold Cross Validation**: o conjunto de amostras é dividido aleatoriamente em K subconjuntos (*folds*) de mesmo tamanho. A cada iteração, o modelo é treinado em $K-1$ folds e avaliado no fold restante; repete-se até que cada fold tenha sido usado como validação uma vez.
- A performance é a média sobre os K folds:

$$\mathrm{CV}_{(K)} = \frac{1}{K}\sum_{k=1}^{K} \mathrm{Erro}_k,$$

onde Erro_k é o erro avaliado quando o k-ésimo *fold* é a validação.



Conjuntos de Treinamento e Teste

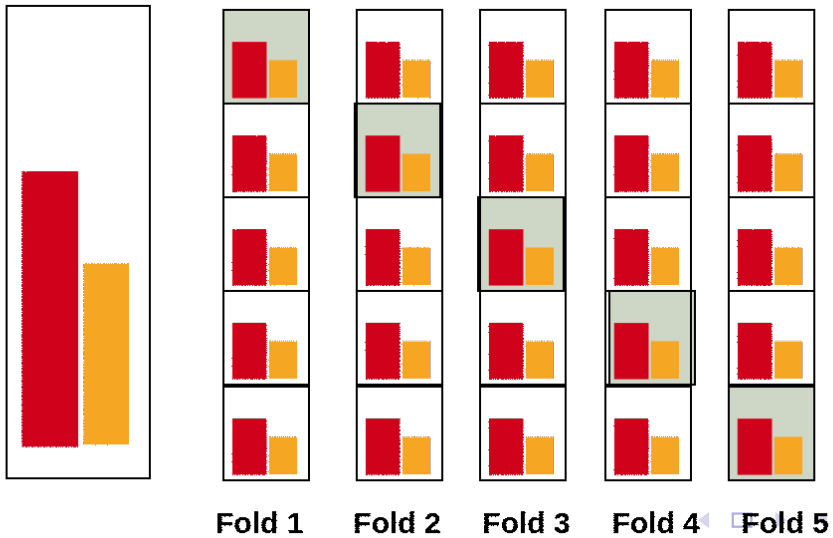
Técnicas de reamostragem: o processo de seleção de amostras para o conjunto de treinamento e para o conjunto de teste é repetido várias vezes.

- **K-Fold Cross Validation**
  - Valores comuns para K são 3, 5 e 10.
  - Avaliar o mesmo modelo com diferentes valores de K e compará-los.
  - O ideal seria treinar o modelo com todo o dataset e avaliar o desempenho em um conjunto que não foi usado no treinamento. Mas isso normalmente não é viável.
  - Uma alternativa é usar o leave-one-out cross-validation (LOOCV), que significa fazer $K = n$, onde n é o número de instâncias no dataset. Computacionalmente é muito custosa.
  - Pode-se comparar a precisão média da classificação para diferentes valores de K com a precisão média da classificação de LOOCV no mesmo conjunto de dados.
  - A diferença entre as pontuações fornece uma aproximação de quão bem um valor K se aproxima da condição de teste de avaliação do modelo ideal.

Conjuntos de Treinamento e Teste

Técnicas de reamostragem: o processo de seleção de amostras para o conjunto de treinamento e para o conjunto de teste é repetido várias vezes.

- **Stratified K-Fold Cross Validation**
  - Usar K-fold Cross Validation no caso do conjunto de dados ser desbalanceado, o modelo pode ser impactado. O Stratified K-Fold Cross Validation procura atacar este problema.
  - No caso de problemas de classificação, a divisão de dados é feita de forma que cada subconjunto tenha a mesma proporção de elementos de cada classe que tem no conjunto todo.



Conjuntos de Treinamento e Teste

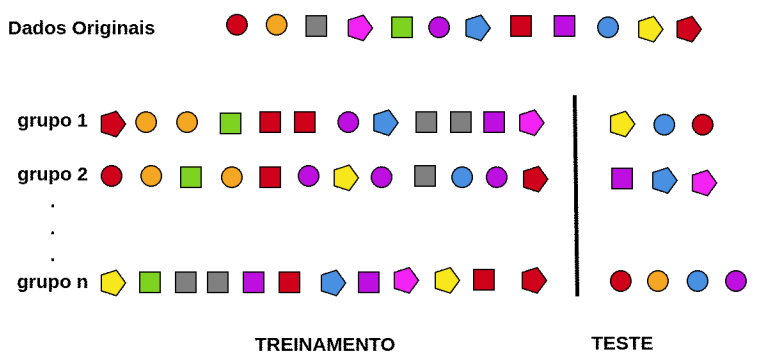
Técnicas de reamostragem: o processo de seleção de amostras para o conjunto de treinamento e para o conjunto de teste é repetido várias vezes.

- **Bootstrap**: do conjunto original $\mathcal{D}$ com n pontos, geramos B conjuntos de treino $\mathcal{D}_1^*, \ldots, \mathcal{D}_B^*$, cada um com n pontos amostrados **com reposição**. Pontos podem aparecer múltiplas vezes ou ficar de fora.

- Os pontos de fora formam o **out-of-bag (OOB)**: $\mathcal{D}_{\text{OOB}}^{(b)} = \mathcal{D} \setminus \mathcal{D}_b^*$. Como $(1 - 1/n)^n \to 1/e$, em média $\approx 36{,}8\%$ ficam OOB e servem de validação "grátis":

$$\text{Erro}_{\text{OOB}} = \frac{1}{B} \sum_{b=1}^{B} \text{Erro em } \mathcal{D}_{\text{OOB}}^{(b)}.$$

- Datasets grandes: $|\mathcal{D}_b^*| < n$ é aceitável (50–80%). Recomenda-se $B \geq 30$ repetições.



ICP363 - Introdução ao Aprendizado de Máquina

Avaliação de Modelos: Bias-Variance, Reamostragem e Métricas

Paulo Mann

Slides adaptados do material original do Prof. João C. P. da Silva

14 de maio de 2026