Machine Learning
Aula 2

Prof. Thiago A. N. De Andrade

Universidade Federal de Santa Maria
Departamento de Estatística

2025-08-18

Aviso aos estudantes

> - Este é um material novo e atualizado, elaborado especialmente para nosso curso **Machine Learning - UFSM 2025.2**. Entretanto, **não se configura em conteúdo original**. É apenas uma compilação resumida de conteúdos presentes nas referências citadas. Em resumo: é indispensável consultar as referências indicadas.
>
> - As imagens não são autorais e os respectivos créditos são reservados aos autores.
>
> - Este material foi integralmente produzido em `R Markdown`, utilizando o pacote `xaringan`, que possibilita a criação de apresentações **ninja**.

Conceitos fundamentais

- *Overfitting* (sobreajuste) e *Underfitting* (subajuste);
- Base de treino vs Base de teste;
- Hiperparâmetros e Ajuste de hiperparâmetros (*Tuning*);
- Base de validação, Validação cruzada (*cross-validation*) e *Leave-One-Out Cross-Validation*

Overfitting e *Underfitting*

Ideias iniciais

- Considere o modelo gerador $y = f(\mathbf{x}) + \varepsilon$, com $\mathbb{E}[\varepsilon] = 0$, $\mathrm{Var}(\varepsilon) = \sigma^2$.
- Chamamos de **erro de teste (ou risco preditivo)** em x_0 a quantidade

$$\mathbb{E}\left[(y_0 - \hat{f}\,(x_0))^2\right] = \underbrace{\sigma^2}_{\text{ruído (irredutível)}} + \underbrace{\left(\mathrm{Bias}[\hat{f}\,(x_0)]\right)^2}_{\text{viés}} + \underbrace{\mathrm{Var}[\hat{f}\,(x_0)]}_{\text{variância}}.$$

- **Underfitting:** baixa flexibilidade, que implica **viés alto**, que resulta em erros altos no treino **e** teste.
- **Overfitting:** flexibilidade excessiva, que implica **variância alta**, que resulta em **erro de treino baixo** e **erro de teste alto**.

De modo formal

Considere dados gerados por um processo $y = f(\mathbf{x}) + \varepsilon$. Um algoritmo escolhe um modelo h pertencente a uma família de modelos $\mathcal{H}$ ($h \in \mathcal{H}$), minimizando o **erro empírico** $\hat{R}_S(h)$ (no treino S).

- Chamamos de **Erro de generalização** (ou risco esperado): $R(h) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[L(y, h(\mathbf{x}))]$. (Ex.: L pode ser MSE em regressão ou erro 0–1 em classificação.). Aqui, a notação $\sim \mathcal{D}$, denota que o valor esperado é calculado com base na distribuição $\mathcal{D}$.

- Note que $R(h)$ não pode ser calculado, uma vez que o processo gerador dos dados não é conhecido.
- Não obstante, o **erro empírico** $\hat{R}_S(h)$ pode ser calculado.
- Chamamos de **lacuna de generalização (gap)** a diferença

$$\text{gap} = R(h) - \hat{R}_S(h).$$

Overfitting (sobreajuste)

Dizemos que um modelo h sofre *overfitting* quando ele obtém **erro empírico muito baixo** (se ajusta demais às particularidades/ruído de S), mas apresenta **erro esperado relativamente alto**, i.e.,

$$\hat{R}_S(h) \text{ muito baixo} \quad \texttt{e} \quad R(h) \text{ alto},$$

que implica grande lacuna

$$\text{gap} = R(h) - \hat{R}_S(\hat{h}).$$

Equivalentemente, existe um modelo h' (tipicamente menos complexo que h) tal que

$$R(h) \; > \; R(h'), \quad \text{apesar de} \quad \hat{R}_S(h) \leq \hat{R}_S(h').$$

Em termos de **viés–variância**, o sobreajuste é caracterizado por **variância alta** (sensibilidade ao conjunto de treino), com viés baixo. Costuma ocorrer quando a **capacidade** da classe $\mathcal{H}$ (graus de liberdade, número de parâmetros, complexidade efetiva) é **grande** em relação ao tamanho/informação dos dados ou quando a **regularização é fraca**.

Sinais práticos de detecção: erro de treino muito baixo, erro de validação/teste bem mais alto e curvas de aprendizado com **grande gap** treino–validação. Em geral, reduzir complexidade ou aumentar regularização melhora o desempenho fora da amostra.

Underfitting (subajuste)

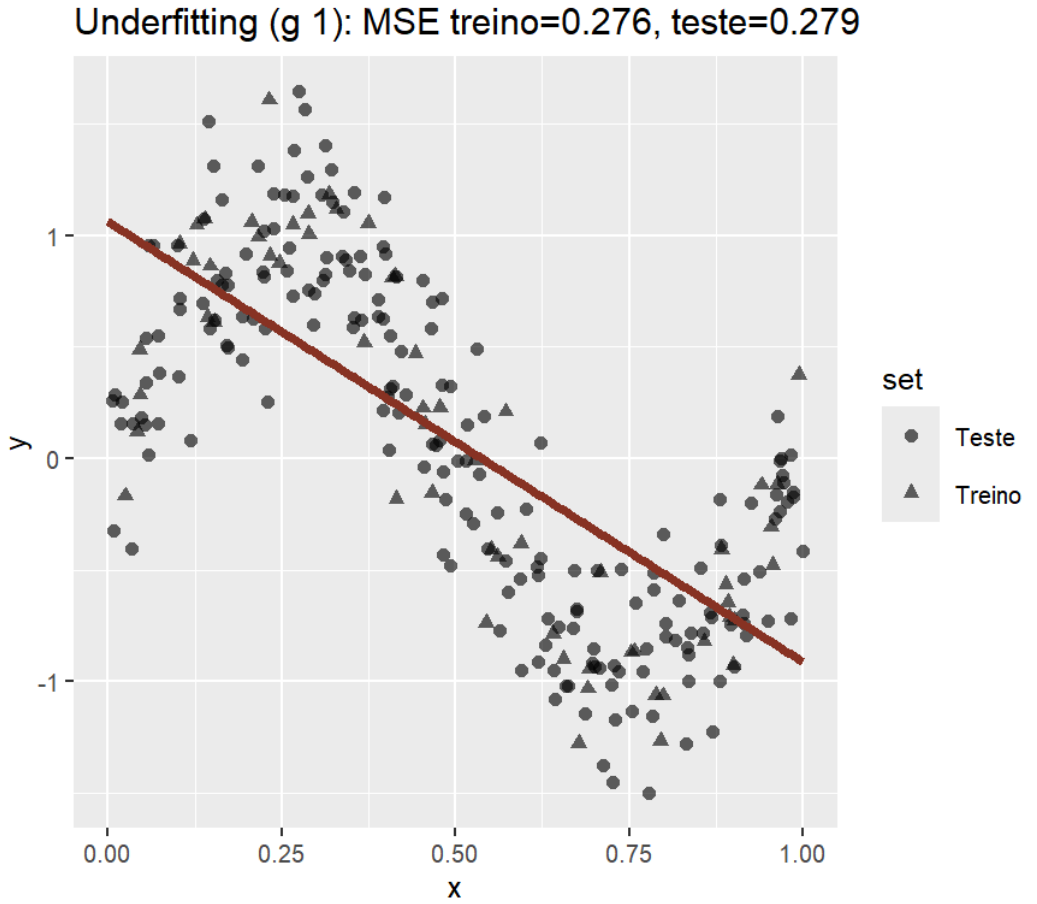
Dizemos que um modelo h sofre *underfitting* quando a classe $\mathcal{H}$ é **pouco flexível** para aproximar f, levando a **erro alto já no treino** e, por consequência, também no teste:

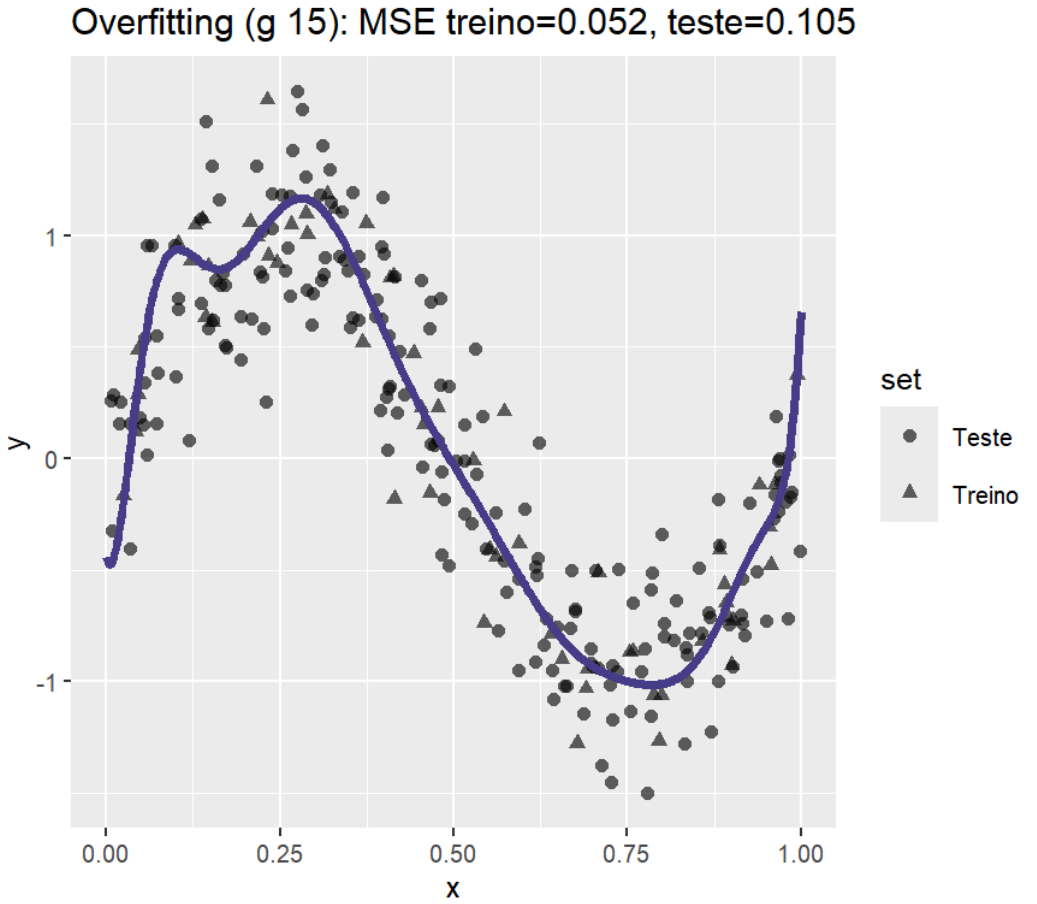
$$\hat{R}_S(h) \text{ alto} \quad \text{e} \quad R(h) \text{ alto},$$

com **lacuna de generalização** pequena (o modelo erra de forma consistente). Em termos de **viés–variância**, o modelo subajuste é caracterizado por **viés alto** (modelo restritivo), usualmente com variância baixa. Ocorre quando a **complexidade é insuficiente** (e.g., polinômio de grau muito baixo, árvore rasa demais) ou quando a **regularização é excessiva**.

Sinais práticos de detecção: erros altos no treino e no teste. Tipicamente, aumentar a complexidade (ou reduzir a regularização) reduz ambos os erros.

> Mais dados, por si só, **não** resolvem se a hipótese é inadequada.


Underfitting (g 1): MSE treino=0.276, teste=0.279
1
0
-1
y
0.00
0.25
0.50
0.75
1.00
x
set
Teste
Treino


Overfitting (g 15): MSE treino=0.052, teste=0.105
1
0
-1
y
0.00
0.25
0.50
0.75
1.00
x
set
Teste
Treino

Base de treino vs Base de teste

Ideias iniciais

- **Objetivos distintos:**
  - **Treino**: base usada para ajustar parâmetros do modelo e, se aplicável, **escolher hiperparâmetros** via **validação/validação cruzada (CV)**.
  - **Teste**: base usada para estimar o **erro de generalização** de um modelo **já finalizado** (hiperparâmetros fixados).
- **Independência:** A base de **teste** deve ser **independente** do processo de ajuste/seleção. Usá-la para *tuning (escolha de hiperparâmetro ótimo)* introduz **viés otimista**.

- **Regra de ouro: Finalize o modelo** (hiperparâmetros escolhidos por validação/CV e parâmetros reestimados no treino completo). **Só então** avalie **uma única vez** o modelo na base de **teste**.

  Qualquer pré-processamento é ajustado no treino/CV e apenas aplicado no teste (sem refitar no teste).

De modo formal

Considere dados gerados por $(\mathbf{X}, Y) \sim \mathcal{D}$. Os conjuntos de **treino** S e **teste** T são amostras independentes e identicamente distribuídas (i.i.d.) de $\mathcal{D}$.

- Escolhemos um modelo $\hat{h} \in \mathcal{H}$ **usando apenas** S (possivelmente com validação/CV), minimizando o **erro empírico** no treino dado por

$$\hat{R}_S(h) = \frac{1}{|S|} \sum_{(\mathbf{x}_i, y_i) \in S} L\big(y_i, h(\mathbf{x}_i)\big).$$

- Após **finalizar** $\hat{h}$, estimamos o **erro de generalização** na **amostra independente** T

$$\hat{R}_T(\hat{h}) = \frac{1}{|T|} \sum_{(\mathbf{x}_i, y_i) \in T} L\big(y_i, \hat{h}(\mathbf{x}_i)\big).$$

Se T for usado para escolher $\hat{h}$, $\hat{R}_T(\hat{h})$ torna-se viesado (otimista) e deixa de refletir o desempenho fora da amostra.

Exemplos no R

```r
library(tidymodels); set.seed(8199510)  # reprodutibilidade

# Exemplo com iris (alvo categórico: Species)
data(iris)

# 80% treino, 20% teste.
# Estratificando para manter a proporção das classes
split_cls <- initial_split(iris, prop = 0.80, strata = Species)

treino_cls <- training(split_cls)
teste_cls  <- testing(split_cls)

# (opcional) Conferir a proporção das classes
treino_cls |> count(Species) |>  mutate(prop = n/sum(n))
teste_cls  |>  count(Species) |>  mutate(prop = n/sum(n))
```

```r
library(tidymodels)

set.seed(8921)

data(mtcars)  # alvo numérico: mpg

# 75% treino, 25% teste. Estratificando por mpg
split_reg <- initial_split(mtcars, prop = 0.75, strata = mpg)

treino_reg <- training(split_reg)
teste_reg  <- testing(split_reg)

# (opcional) checar tamanhos
dim(treino_reg); dim(teste_reg)
```

Hiperparâmetros e Ajuste de hiperparâmetros (*Tuning*)

O que são?

- **Parâmetros**: quantidades fixas e desconhecidas que indexam um modelo estatístico, sendo **estimadas a partir dos dados**. Ex.: coeficientes β em regressão linear.

- **Hiperparâmetros**: são **controles de complexidade/estrutura** definidos **antes** do ajuste dos parâmetros, e.g.: k no K-Nearest Neighbors (k-NN), a profundidade da árvore, λ (penalidade) e α (mistura) no Elastic Net, número de neurônios/camadas em redes neurais.

> Hiperparâmetros **induzem** uma família de modelos $\{\mathcal{H}_\lambda\}_{\lambda \in \Lambda}$ e, portanto, **alteram o equilíbrio entre viés–variância** do estimador.

Problema de seleção (*tuning*)

- Dado $\lambda \in \Lambda$ (hiperparâmetro), o objetivo é minimizar o **erro de generalização**

$$\lambda^\star \in \arg\min_{\lambda \in \Lambda} \; R\big(\hat{h}_\lambda\big), \qquad R(h) = \mathbb{E}_{(X,Y)\sim\mathcal{D}}[L(Y, h(X))].$$

- Como R é desconhecido, usamos um **estimador** via **validação** ou **CV k-fold (validação cruzada com k dobras)**:

$$\widehat{R}_{\mathrm{CV}}(\lambda) = \frac{1}{k} \sum_{j=1}^{k} \frac{1}{|V_j|} \sum_{(x_i, y_i) \in V_j} L\big(y_i, \hat{h}_{\lambda}^{(-j)}(x_i)\big),$$

em que $\hat{h}^{(-j)}$ é o modelo treinado sem a dobra V_j (Maiores detalhes nos slides a seguir).

- Escolha prática: $\hat{\lambda} = \arg\min_{\lambda} \widehat{R}_{\mathrm{CV}}(\lambda)$ ou **regra de 1-SE** (*one-SE rule*, que consiste em preferir modelo mais simples dentro de 1 desvio-padrão do mínimo).

Boas práticas

- **Nunca use a base de teste no tuning** (evita viés otimista).
- O **Pré-processamento** (e.g. escalonamento, *dummies*, PCA) deve ser **ajustado no treino/validação simples ou validação cruzada (CV)** e apenas **aplicado** ao teste (sem *refit* no teste).
- Pesquise λ em **escala log** (e.g.: $[10^{-10}, 1]$).
- Em comparações múltiplas ou *workflows* complexos, pode ser útil **validação cruzada aninhada** (*nested CV*).

Obs: Use escala logarítmica para pesquisar λ:

- A sensibilidade do modelo a esse hiperparâmetro é altamente não linear
- A escala log explora melhor as regiões nas quais pequenas mudanças em λ causam grandes mudanças no modelo.
- Evita testar muitos valores altos para λ, que resultam modelos nulos (coeficientes perto de zero).

Base de validação, Validação cruzada (*cross-validation*) e *Leave-One-Out Cross-Validation*

Conceitos básicos

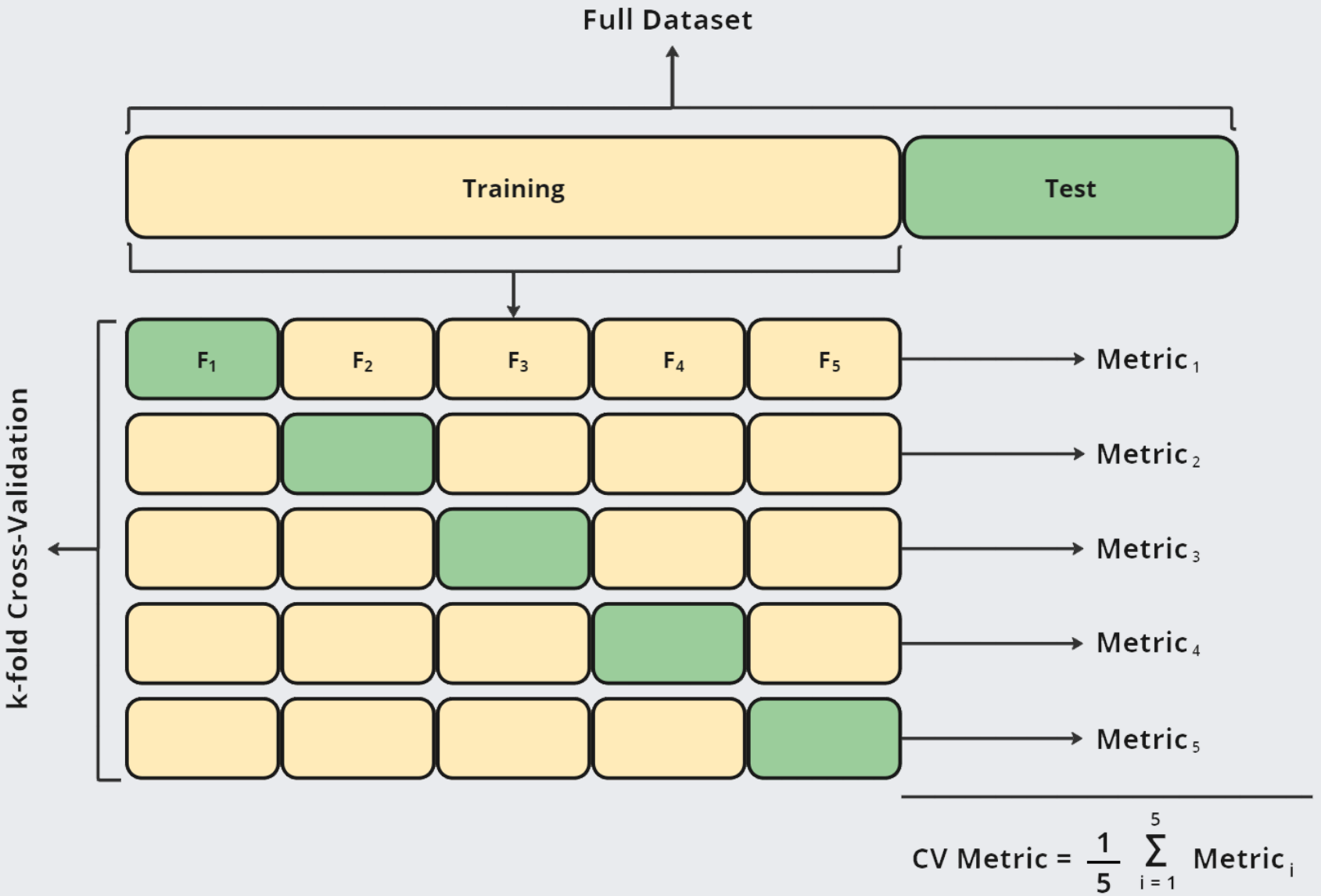
- Dados $(\mathbf{X}, Y) \sim \mathcal{D}$. Queremos estimar o **erro de generalização**: $R(h) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[L\big(y, h(\mathbf{x})\big)]$.
- No treino temos o **erro empírico** (risco amostral): $\hat{R}_S(h) = \frac{1}{|S|}\sum_{(\mathbf{x}_i,y_i)\in S} L\big(y_i, h(\mathbf{x}_i)\big)$.

- **Base de validação (holdout)**: consiste na separaração de um subconjunto dos dados de treino para estimar o erro fora da amostra.
- **Validação cruzada (CV)**: particionar a base de treino em vários subconjuntos, deixando sempre alguma fração para uma espécie de mini teste".
- **Erro estimado na CV**: média de erros obtidos por reamostragens/partições do conjunto de dados, realizada durante a validação cruzada.

k-fold cross-validatio: intuição

- Divida os dados em **treino** e **teste** (por e.g., 80%/20%).
- Considerando os dados de treino (e.g., 80%) subdivida-os em partes iguais (e.g., 4 partes 20%). Cada uma destas partes será usada como base de validação (e.g., cada parte de 20% será a validação).
- Ajuste o modelo no treino e estime o erro na validação.

- **Vantagens:** simples, rápida.
- **Desvantagens:**
  - Alta **variabilidade** da estimativa (depende do *split*).
  - **Desperdício** de dados para ajuste (tendência a **superestimar** o erro do modelo final treinado).


Full Dataset
Training
Test
k-fold Cross-Validation
F1
F2
F3
F4
F5
Metric 1
Metric 2
Metric 3
Metric 4
Metric 5
CV Metric = 1/5 Σ (i = 1 to 5) Metric i

k-fold cross-validation de maneira formal

- Particione os dados em k dobras $V_1, \ldots, V_k$ (tamanhos semelhantes. Estratificar quando aplicável).
- Note que $V_1 + \cdots + V_k = S$.
- Para cada $j = 1, \ldots, k$: ajuste em $S \setminus V_j$ e compute o erro em V_j.
- **Estimador:** $\mathrm{CV}(k) = \frac{1}{k} \sum_{j=1}^{k} \widehat{R}\big(\hat{h}^{(-j)}, V_j\big)$,

em que $\hat{h}^{(-j)}$ é o modelo treinado sem a dobra V_j.

- **Leave-One-Out Cross-Validation (LOOCV)** é o caso $k = n$ (deixa-um-fora).

Refletindo

- Note que **não dá para "calcular" a lacuna de generalização** $gap = R(h) - \hat{R}_S(h)$ **exatamente** (aquela apresentada nos slides anteriores), porque $R(h)$ depende da distribuição desconhecida $\mathcal{D}$. O que fazemos na prática é **estimar** ou **limitar (dar bounds)** essa lacuna. As formas mais usadas são apresentadas a seguir.

1) Estimar com *holdout* (teste independente)

Se você tem um conjunto de **teste** T que **não** participou de treino/tuning:

$$\widehat{\text{gap}} \;=\; \hat{R}_T(\hat{h}) \;-\; \hat{R}_S(\hat{h}).$$

Observe que há uma condição crucial: $\hat{h}$ foi escolhido **sem olhar** para T. Se T for usado no tuning, $\hat{R}_T$ fica otimista (creio que já enfatizamos bastante este aspecto).

2) Estimar com validação cruzada (CV)

Sem teste externo, use CV para aproximar $R(h)$:

$$\widehat{\mathrm{gap}}_{\mathrm{CV}} \;=\; \hat{R}_{\mathrm{CV}}(\hat{h}) \;-\; \hat{R}_S(\hat{h}).$$

Se houve **tuning** com a mesma CV, essa $\widehat{R}_{\mathrm{CV}}$ tende a ficar **otimista**. A correção padrão é **CV aninhada (nested CV)**: a *outer* CV fornece $\widehat{R}_{\mathrm{outer}}$ do *pipeline* completo (incluindo tuning interno). Então,

$$\widehat{\mathrm{gap}}_{\mathrm{nested}} = \frac{1}{k}\sum_{j=1}^{k}\left(\hat{R}_{S^{(j)}_{\mathrm{train}}}(\hat{h}_j) - \widehat{R}^{(j)}_{\mathrm{outer}}\right)$$

em que

- $\hat{R}_{S^{(j)}_{\mathrm{train}}}(\hat{h}_j)$ é o erro no treino externo da dobra (j). Ou seja, é o erro de validação calculado quando a dobra j é usada como conjunto de teste.
- $\widehat{R}^{(j)}_{\mathrm{outer}}$ é o erro no teste externo da dobra j (conjunto $V^{(j)}_{\mathrm{outer}}$.

Regra de bolso, rática e simples: inspecione **curvas de aprendizado** (erro de treino vs. validação por tamanho de amostra). O “gap” observado neste contexto é uma boa *proxy* para (sub/sobre)ajuste.

Referências

- **Regularization and Variable Selection via the Elastic Net**
- **Regression Shrinkage and Selection via the Lasso**
- **Ridge Regression: Biased Estimation for Nonorthogonal Problems**
- **A Survey of Cross-Validation Procedures for Model Selection**

- **The Predictive Sample Reuse Method with Applications**
- **Estimating the Error Rate of a Prediction Rule: Improvement on Cross-Validation**
- **A Leisurely Look at the Bootstrap, the Jackknife, and Cross-Validation**
- **Some Studies in Machine Learning Using the Game of Checkers**

- **Computing Machinery and Intelligence**
- **Machine Learning: an introduction**
- **An Introduction to Statistical Learning**
- **Aprendizado de máquina: uma abordagem estatística**
- **Materiais Curso R**

Não deixe de entrar em contato comigo para tirar suas dúvidas: thiagoan.andrade@gmail.com

Estamos no  @thiagoan.andrade para networking e socializações

Obrigado!

Thanks!