Métodos de Reamostragem

Thiago Ramos

26 de Março, 2024

Objetivo da aula

- Vamos introduzir alguns métodos de reamostragem como **Bootstrap** e **Validação Cruzada**.
- Vamos aprender que tipo de problema esses métodos tentam resolver e qual a abordagem utilizada para resolvê-los.
- Vamos aprender como trabalhar com esses métodos na prática utilizando Python.

Notas de aula e códigos

- Os slides e os Notebooks contendo os códigos que utilizaremos nessa aula estão disponíveis em:



Figura: `https://shorturl.at/ewBIX`

Bootstrap

Bootstrap: motivação

- Seja $X_1, \ldots, X_n \sim P$ uma amostra i.i.d. e θ a mediana de P, isto é, $P(X_i \leq \theta) = P(X_i \geq \theta) = 1/2$. Considere o estimador $\widehat{\theta}_n$ da mediana, dado pela mediana amostral dos dados.
- Queremos responder as seguintes perguntas:
  - Como estimar a variância de $\widehat{\theta}_n$?
  - Como construir um intervalo de confiança de nível $(1 - \alpha)$ para θ?

Bootstrap: desafios

- Seja $X_1, \ldots, X_n \sim P$ e θ a mediana de P, isto é, $P(X_i \leq \theta) = P(X_i \geq \theta) = 1/2$. Considere o estimador $\widehat{\theta}_n$ da mediana, dado pela mediana amostral dos dados.
- Quais são os desafios desse problema?
  - A distribuição P dos dados pode não ter uma forma paramétrica clara.
  - Nem todo estimador tem sua distribuição amostral facilmente definida, mesmo quando se conhece a distribuição original dos dados.

Bootstrap: possível solução

- Uma possível solução seria seguir os seguintes passos:
  1. Para $b \in [1, \ldots, B]$ sorteie uma amostra independente de tamanho n vinda de P, isto é,
  $$X_1^b, X_2^b, \ldots, X_n^b \sim P.$$
  2. Para cada amostra $b \in [1, \ldots, B]$ sorteada no passo anterior, calcule sua respectiva mediana amostral, isto é,
  $$X_1^b, X_2^b, \ldots, X_n^b \to \widehat{\theta}_n^b.$$
  3. Se B é suficientemente grande, então $\{\widehat{\theta}_n^b\}_{b=1}^B$ deve aproximar a distribuição do estimador $\widehat{\theta}_n$ e portanto, conseguimos calcular a variância do estimador e construir intervalos de confiança para θ.

Bootstrap: possível solução

- A lógica da solução anterior está correta, mas na prática ela não funciona.
- O motivo é que supomos que temos acesso à distribuição P. Mas se isso fosse verdade, o problema já estaria resolvido.

Bootstrap: introdução

- O método de Bootstrap nos dá uma forma de adaptar a ideia anterior da prática.
- Ao invés de utilizamos a distribuição P para as $b \in [1, \ldots, B]$ amostragens, utilizamos a distribuição empírica $\widehat{P}_n$, com função de distribuição acumulada:

$$\widehat{F}_n(t) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\left[X_i \leq t\right]$$

Bootstrap: o método

- O método de Bootstrap:
  1. Para $b \in [1, \ldots, B]$ sorteie uma amostra independente de tamanho n vinda de $\widehat{P}_n$, isto é,
  $$X_1^b, X_2^b, \ldots, X_n^b \sim \widehat{P}_n.$$
  Note que isso é equivalente a sortear pontos da amostra original, com repetição.
  2. Para cada amostra $b \in [1, \ldots, B]$ sorteada no passo anterior, calcule sua respectiva mediana amostral, isto é,
  $$X_1^b, X_2^b, \ldots, X_n^b \to \widehat{\theta}_n^b.$$
  3. Utilize a distribuição de $\{\widehat{\theta}_n^b\}_{b=1}^B$ para calcular estatísticas de interesse e construir intervalos de confiança.

Bootstrap: exemplo prático

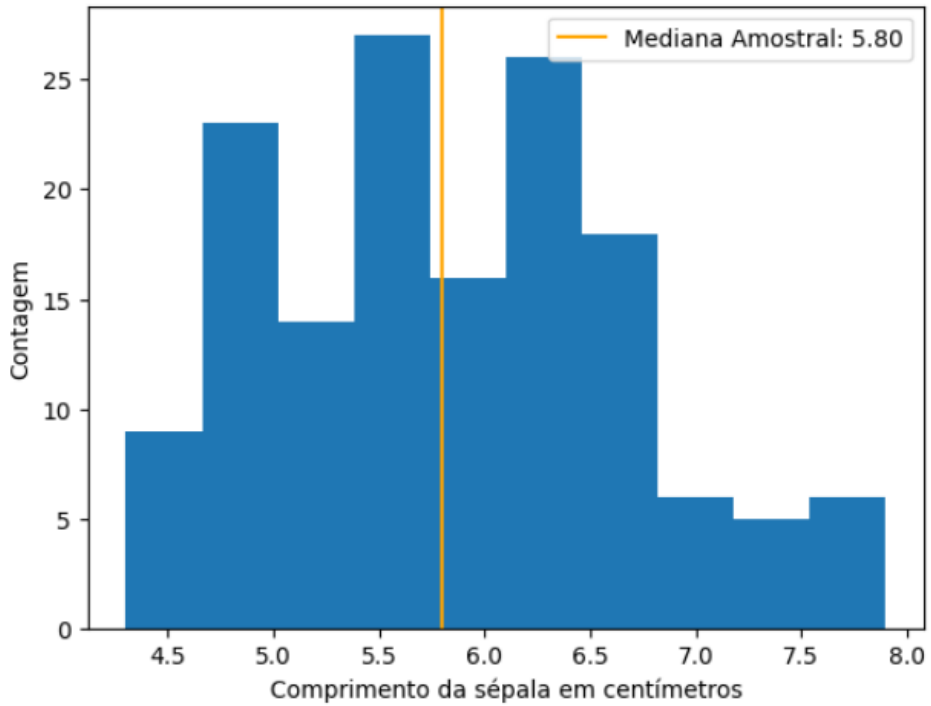
- Vamos usar o dataset Iris, que é um conjunto de dados multivariados introduzido pelo estatístico e biólogo britânico Ronald Fisher e muito utilizado como exemplo para problems de aprendizado de máquinas.
- Este é um conjunto de dados consiste em 3 diferentes tipos de flores íris (Setosa, Versicolour e Virginica), com comprimentos de pétala e sépala armazenados em um vetor de 150×4.

```
df.head()
```

	sepal.length	sepal.width	petal.length	petal.width	variety
0	5.1	3.5	1.4	0.2	Setosa
1	4.9	3.0	1.4	0.2	Setosa
2	4.7	3.2	1.3	0.2	Setosa
3	4.6	3.1	1.5	0.2	Setosa
4	5.0	3.6	1.4	0.2	Setosa

Bootstrap: exemplo prático

- Vamos utilizar o método de Bootstrap para calcularmos a variância da mediana amostral e construir intervalos de confiança para a mediana dos valores de comprimento de sépala.



Bootstrap: exemplo prático

- O código em Python para a implementação do método de Bootstrap é simples:

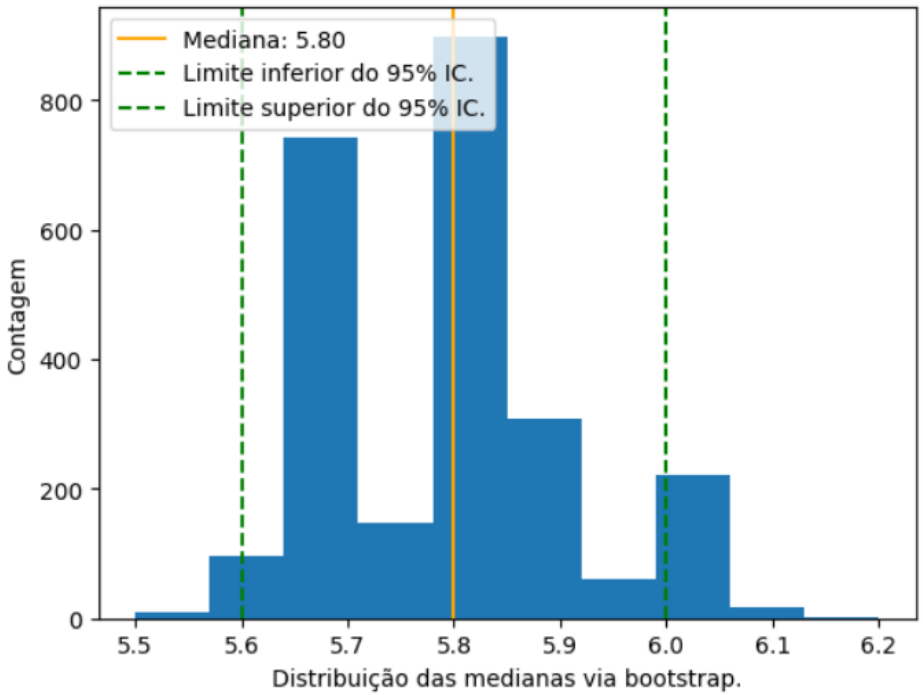
```python
def bootstrap_median(x, B, seed=123):

    rng = np.random.RandomState(seed)
    sample_medians = np.zeros(shape=B)

    for i in range(B):
        boot_sample = rng.choice(x, size=x.shape[0], replace=True)
        bootstrap_replicate = np.median(boot_sample)
        sample_medians[i] = bootstrap_replicate
    return sample_medians
```

Bootstrap: exemplo prático

- Dessa forma é fácil calcular a variância do estimador. Nosso método nos deu o valor de aproximadamente 0.011.
- Além disso conseguimos construir o intervalo de confiança para a mediana. Nosso método nos deu como intervalo de confiança de 95% o intervalo $[5.60, 6.00]$.



s

Bootstrap: Por que funciona?

- Os dados $X_1, \ldots, X_n$ são uma amostra de uma distribuição P que não temos acesso.
- O método de Bootstrap tira conclusões sobre P através de amostras de $\widehat{P}_n$ e não de P.
- Por que essas conclusões deveriam ser validas para P?

Bootstrap: Por que funciona?

- R. Tibshirani conta a origem do nome do método que seu orientador, B. Efron, criou.
- O termo Bootstrap vem do livro " As Aventuras do Barão de Munchausen" de Rudolph Erich Raspe.





Bootstrap: Um gostinho de teoria

Teorema (Desigualdade DKW)

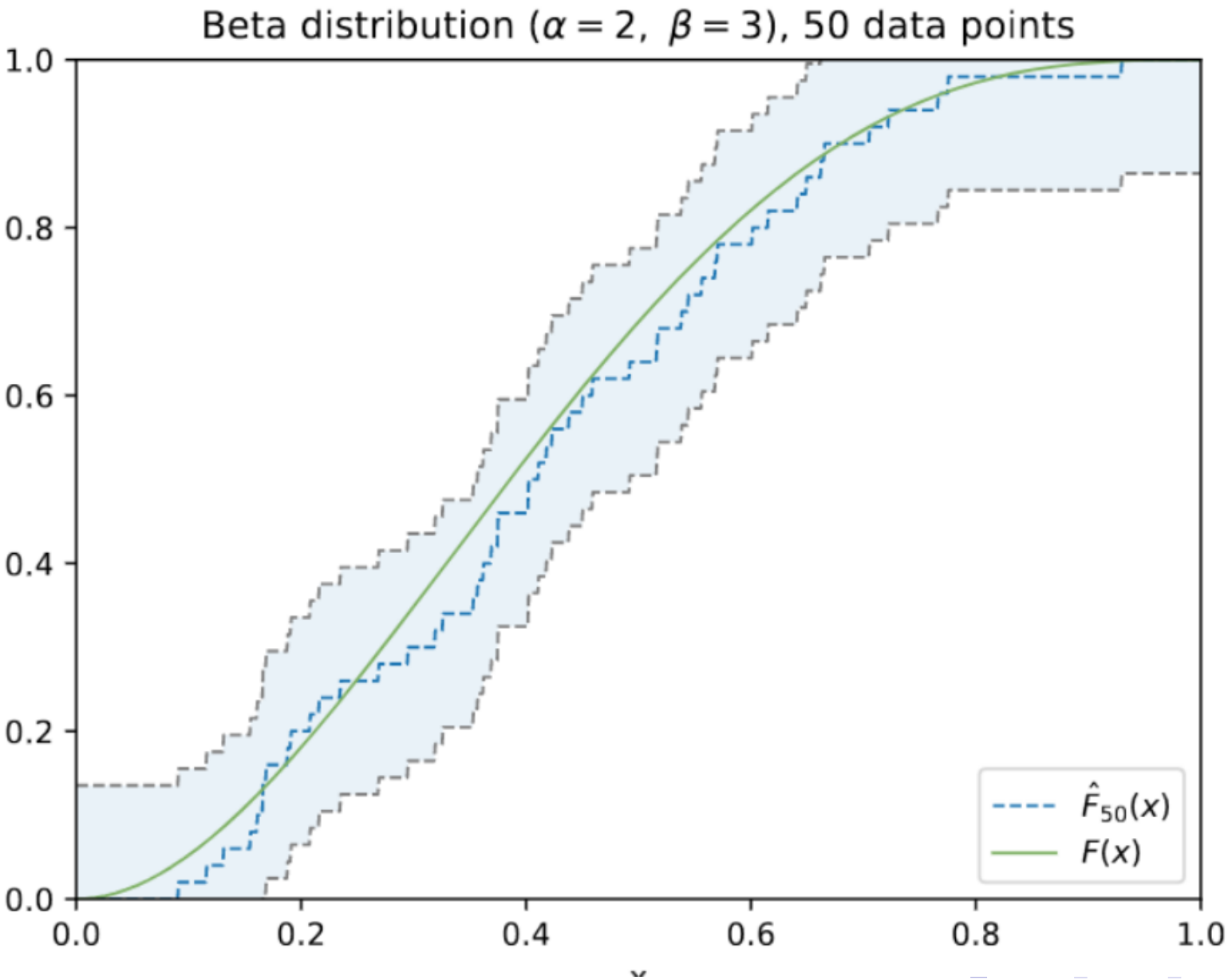
Para qualquer distribuição P e para qualquer $\varepsilon > 0$, temos que

$$P\left[\sup_{t\in\mathbb{R}} \mid F(t) - \widehat{F}_n(t) \mid > \varepsilon\right] \leq 2\exp\left(-2n\varepsilon^2\right).$$

Onde $F(t) = P(X \leq t)$ e $\widehat{F}_n(t) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\left[X_i \leq t\right]$.

- Então, a intuição de por que o bootstrapping funciona reside nos seguintes fatos:
  - Muitos estimadores podem ser formulados como uma função linear da função de distribuição acumulada (CDF).
  - A função de distribuição empírica (EDF) é um bom estimador para a CDF.

Bootstrap: Um gostinho de teoria



Bootstrap: um gostinho da teoria

Teorema (Desigualdade DKW)

Para qualquer distribuição P e para qualquer $\varepsilon > 0$, temos que

$$P\left[\sup_{t\in\mathbb{R}} \mid F(t) - \widehat{F}_n(t) \mid > \varepsilon\right] \leq 2\exp\left(-2n\varepsilon^2\right).$$

Onde $F(t) = P(X \leq t)$ e $\widehat{F}_n(t) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\left[X_i \leq t\right]$.

- A prova desse teorema usa ferramentas importantíssimas na teoria de aprendizado de máquinas, como, Concentração de Medida e Dimensão VC.
- Caso haja interesse da sala, podemos fazer uma aula extra para discutirmos esses conceitos.

Validação Cruzada

Validação Cruzada: motivação

- Seja $(X_1, y_1), \ldots, (X_n, y_n) \sim P$ uma amostra i.i.d.. Suponha que queremos treinar um algoritmo g, para prever y a partir de X.
- Por exemplo, no dataset Iris que utilizamos anteriormente, podemos tentar prever a espécie da planta y (Setosa, Versicolour, and Virginica) utilizando os valores de comprimento e altura das sépalas e pétalas das plantas X.

```
df.head()
```

	sepal.length	sepal.width	petal.length	petal.width	variety
0	5.1	3.5	1.4	0.2	Setosa
1	4.9	3.0	1.4	0.2	Setosa
2	4.7	3.2	1.3	0.2	Setosa
3	4.6	3.1	1.5	0.2	Setosa
4	5.0	3.6	1.4	0.2	Setosa

Validação Cruzada: motivação

- Seja $(X_1, y_1), \ldots, (X_n, y_n) \sim P$ uma amostra i.i.d.. Suponha que queremos treinar um algoritmo g, para prever y a partir de X.
- Queremos responder as seguintes perguntas:
  - Qual metodologia devemos usar para treinar o modelo.
  - Como saber se esse modelo dará bons resultados quando tentarmos prever a espécie de plantas que não estão no dataset original?

Validação Cruzada: possíveis soluções

- Duas soluções intuitivas são:
  1. Treinar o algoritmo utilizando os dados originais e depois estimar o error de predição utilizando os mesmos dados.
  2. Separar os dados originais em dois conjuntos **Treino** e **Teste**. No conjunto **Treino** treinamos o modelo e utilizamos o conjunto **Teste** para estimar o erro de predição (e.g. acurácia).

Motivação: possíveis soluções

- Vamos utilizar o conjunto Iris para comparar as duas alternativas aplicando um modelo de Floresta Aleatória em Python.

```
df.head()
```

	sepal.length	sepal.width	petal.length	petal.width	variety
0	5.1	3.5	1.4	0.2	Setosa
1	4.9	3.0	1.4	0.2	Setosa
2	4.7	3.2	1.3	0.2	Setosa
3	4.6	3.1	1.5	0.2	Setosa
4	5.0	3.6	1.4	0.2	Setosa

Validação Cruzada: exemplo prático

- Seguindo a primeira abordagem, de treinar e avaliar o modelo no mesmo conjunto, obtivemos uma acurácia de 100%, isto é, o modelo não cometeu nenhum erro.

[9]:
```
train_accuracy = accuracy_score(y_train,train_pred)
print("Acurácia no conjunto de treino:",train_accuracy)
```

```
Acurácia no conjunto de treino: 1.0
```

- Agora quando aplicamos algoritmo treinado num conjunto jamais visto (segunda abordagem), obtivemos uma acurácia de 95%.

[11]:
```
test_accuracy = accuracy_score(y_test,test_pred)
print("Acurácia no conjunto de teste:",test_accuracy)
```

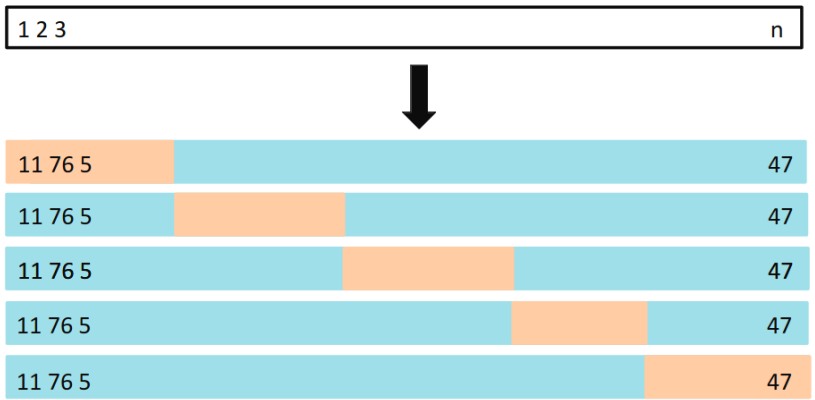
```
Acurácia no conjunto de teste: 0.9555555555555556
```

Validação Cruzada: exemplo prático

- Treinar e avaliar um algoritmo no mesmo conjunto é como aplicar uma prova onde todos os exercícios já foram vistos em aula. É impossível saber se um aluno foi bem porque aprendeu ou se simplesmente decorou todos os exercícios.
- Quando um modelo simplesmente "decora" as possíveis soluções dizemos que houve overfitting.

Validação Cruzada: k-fold

- Podemos diminuir a variância da nossa estimativa da acurácia do modelo aplicando uma técnica chamada de Validação Cruzada em k-folds.
- A estratégia é a seguinte:
  - Divida os dados em k subconjuntos (tipicamente, $k = 5$ ou 10)
  - Para cada $i \in [1, \ldots, k]$:
    1. Treine o modelo em todos os subconjuntos exceto no i-ésimo;
    2. Calcule a acurácia do modelo no i-ésimo subconjunto.
  - Tire a média das k acurácias.



Validação Cruzada: k-fold

- Utilizando essa técnica com 5-folds no problema de previsão de espécies de flores, temos uma acurácia média de 93% que é bem próximo do esperado de 95% no teste.

```
model = RandomForestClassifier(random_state=42)

cv_scores =  cross_val_score(model,X_train,y_train,cv=5,scoring='accuracy')

mean_train_cv_accuracy = train_cv_scores.mean()
print("A acurácia média usando 5-fold validação cruzada no treino é: ",mean_train_cv_accuracy)
```

```
A acurácia média usando 5-fold validação cruzada no treino é:  0.9333333333333333
```

Validação Cruzada: mais um pouquinho de teoria

Teorema (Desigualdade de Hoeffiding)

Para qualquer distribuição P, qualquer $\varepsilon > 0$ e fixado um estimador $g(X) \in \{0, 1\}$, temos que

$$P\left[\mid Err_k(g) - Err(g) \mid > \varepsilon\right] \leq 2\exp\left(-2k\varepsilon^2\right)$$

onde $Err(g)$ é a acurácia na população e $Err_k(g)$ é a acurácia no conjunto de teste, supondo que este tenha k pontos.

- A prova desse teorema usa as mesmas técnicas da desigualdade DKW.
- Esse tipo de resultado é utilizado para provar a capacidade de generalização de modelos clássicos de Aprendizado de Máquina (SVM, Boosting, etc).

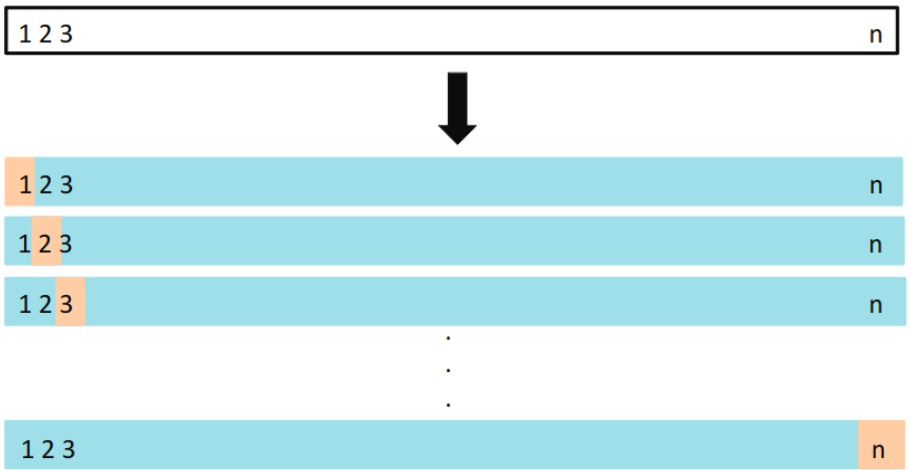
Comparações e extensões

Comparações: Bootstrap e Validação Cruzada

- Seria possível usar bootstrap para estimar erro de teste?
- Usaríamos cada amostra de bootstrap como amostra de treino, os dados de treino como a amostra completa.
- Problema: há interseção entre dados de treino e teste! Isso subestima o erro de teste.
- Alternativa: usar dados de teste como os pontos não sorteados na amostra de bootstrap; mas começa a ficar complicado. Melhor usar k-fold.

Comparações: Validação Cruzada k-fold e Leave-one-Out

- O que acontece se fizermos validação cruzada com $k = n$? Ou seja, para cara $i \in [1, \ldots, n]$
  1. Treine o modelo em todos os pontos, exceto o i-ésimo.
  2. Calcule o erro de teste usando apenas o ponto i.
  3. Tire a média de todos os erros de teste.

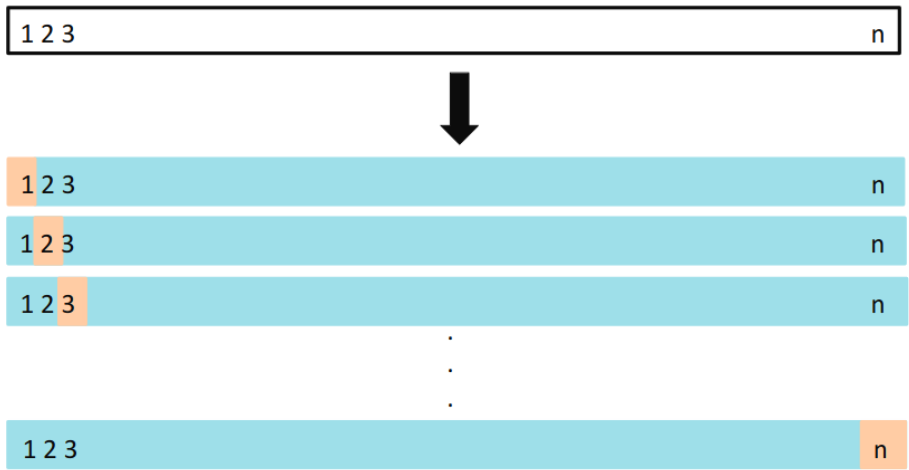


Comparações: Validação Cruzada k-fold e Leave-one-Out

- No contexto de Aprendizado de Máquinas, essa ideia é chamada de Leave-one-Out (LOO)
- Vantagens de LOO sobre conjunto de validação:
  - LOOCV não é aleatório: sempre retorna o mesmo estimador
  - LOOCV usa quase todos os dados de treino: não sobrestima o erro de teste
- Desvantagens:
  - Modelos treinados em dados correlacionados, aumentando a variância da estimativa de teste
  - É computacionalmente intensivo.

Comparações: Leave-one-Out e Bootstrap

- A ideia de apartir de uma amostra de tamanho n, derivar n amostras de tamanho $n-1$, como feito no LOO, é conhecida também como método de Jackknife.
- O método de Jackknife, assim como o de Bootstrap, é utilizado para estimação de parâmetros.
- É bastante utilizado para estimação de viés e de variância de estimadores.



Conclusão

Conclusão

- Se precisamos estimar estatísticas de alguma distribuição e não temos hipóteses paramétricas sobre os dados, utilizamos **Bootstrap**.
- Se precisamos estimar o erro médio que um modelo de aprendizado de máquinas terá na prática, utilizamos **Validação Cruzada**.