Inteligência Artificial – ACH2016

Aula23 – Classificadores: Avaliação de Desempenho

Norton Trevisan Roman
(norton@usp.br)

6 de junho de 2019

Avaliação de Desempenho

Medindo desempenho

- Já temos nosso modelo treinado, então está tudo terminado, certo?

Avaliação de Desempenho

Medindo desempenho

- Já temos nosso modelo treinado, então está tudo terminado, certo?
  - Errado! Como podemos saber se ele é bom realmente?
  - Temos que testá-lo

Avaliação de Desempenho

Medindo desempenho

- Já temos nosso modelo treinado, então está tudo terminado, certo?
  - Errado! Como podemos saber se ele é bom realmente?
  - Temos que testá-lo
- Para isso, contudo, precisamos de uma medida de desempenho
  - Algo que nos diga quão bom nosso modelo é para a tarefa
  - Depende da tarefa em questão

Avaliação de Desempenho

Taxa de Erro

- Para classificação, por exemplo, podemos usar a **taxa de erro**
  - O classificador prediz a classe de cada instância
  - Se estiver correto, conta para o número de sucessos, senão, conta para os erros

Avaliação de Desempenho

Taxa de Erro

- Para classificação, por exemplo, podemos usar a **taxa de erro**
  - O classificador prediz a classe de cada instância
  - Se estiver correto, conta para o número de sucessos, senão, conta para os erros
- A taxa é então a proporção de erros feita em todo o conjunto de instâncias X

$$T_E = \frac{n_E(X)}{n(X)}$$

Avaliação de Desempenho

Taxa de Erro

- Para classificação, por exemplo, podemos usar a **taxa de erro**
  - O classificador prediz a classe de cada instância
  - Se estiver correto, conta para o número de sucessos, senão, conta para os erros
- A taxa é então a proporção de erros feita em todo o conjunto de instâncias X

$$T_E = \frac{n_E(X)}{n(X)} \longleftarrow$$

Número de instâncias classificadas erroneamente em X

Avaliação de Desempenho

Taxa de Erro

- Para classificação, por exemplo, podemos usar a **taxa de erro**
  - O classificador prediz a classe de cada instância
  - Se estiver correto, conta para o número de sucessos, senão, conta para os erros
- A taxa é então a proporção de erros feita em todo o conjunto de instâncias X

$$T_E = \frac{n_E(X)}{n(X)}$$

Número total de instâncias em X

Avaliação de Desempenho

Conjuntos de treino

- Temos agora uma medida de erro e o conjunto de treino. Então podemos ver como o modelo treinado se sai nesse conjunto

Avaliação de Desempenho

Conjuntos de treino

- Temos agora uma medida de erro e o conjunto de treino. Então podemos ver como o modelo treinado se sai nesse conjunto

Fonte: https://twitter.com/soquenaoreal

Avaliação de Desempenho

Conjuntos de treino

- Temos agora uma medida de erro e o conjunto de treino. Então podemos ver como o modelo treinado se sai nesse conjunto

Fonte: https://twitter.com/soquenaoreal

- Qualquer estimativa de desempenho com base nesse conjunto será otimista
  - Em tese, aprendemos como esse conjunto funciona
- Não temos ideia de como o modelo irá se comportar com dados nunca vistos
  - Afinal, já sabemos os resultados no conjunto de treino. Nos interessam os resultados em dados novos

Avaliação de Desempenho

Conjunto de treino: Erro de resubstituição

- Então não devemos medir a taxa de erro no conjunto de treino?

Avaliação de Desempenho

Conjunto de treino: Erro de resubstituição

- Então não devemos medir a taxa de erro no conjunto de treino? Devemos sim

Avaliação de Desempenho

Conjunto de treino: Erro de resubstituição

- Então não devemos medir a taxa de erro no conjunto de treino? Devemos sim
- A taxa de erro no conjunto de treino é chamada de **erro de resubstituição**
    - Porque é calculada pela resubstituição das instâncias de treino no classificador

Avaliação de Desempenho

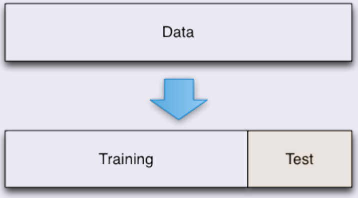
Conjunto de treino: Erro de resubstituição

- Então não devemos medir a taxa de erro no conjunto de treino? Devemos sim
- A taxa de erro no conjunto de treino é chamada de **erro de resubstituição**
  - Porque é calculada pela resubstituição das instâncias de treino no classificador
- Útil para termos uma ideia da adequação do classificador
  - Se ele não se dá bem onde treinou, é um forte indicativo de que esse é o modelo errado

Avaliação de Desempenho

Conjunto de Teste

- Devemos então testar nosso modelo em um **conjunto de teste** (ou *holdout*)
  - Um conjunto independente, com dados não vistos pelo modelo
  - Treinamos no conjunto de treino e testamos no de teste
  - O erro nesse conjunto será nossa estimativa do erro em exemplos não vistos



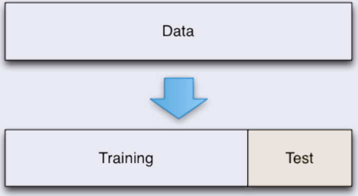
Fonte: [10]

  - Assumindo que ambos os conjuntos são amostras representativas do problema

Avaliação de Desempenho

Conjunto de Teste

- E como construímos este conjunto?
  - Ou tiramos nova amostra aleatória da população
  - Ou separamos aleatoriamente, de forma independente, os dados que já temos em mãos
    - Em geral, 1/3 dos dados é reservado para teste

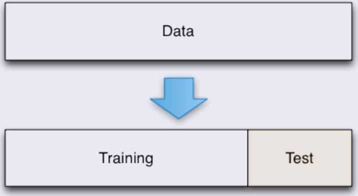


Fonte: [10]

Avaliação de Desempenho

Conjunto de Teste

- E como construímos este conjunto?
  - Ou tiramos nova amostra aleatória da população
  - Ou separamos aleatoriamente, de forma independente, os dados que já temos em mãos
    - Em geral, 1/3 dos dados é reservado para teste



Fonte: [10]

- E medimos a taxa de erro nesse conjunto
  - Ou, alternativamente, a **taxa de sucesso** $T_S = 1 - T_E$

Avaliação de Desempenho

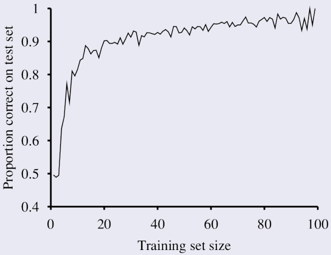
Curva de aprendizagem

- A taxa de sucesso, contudo, dá um resultado final
  - Não nos diz como o algoritmo evoluiu ao longo do treinamento

Avaliação de Desempenho

Curva de aprendizagem

- A taxa de sucesso, contudo, dá um resultado final
  - Não nos diz como o algoritmo evoluiu ao longo do treinamento
- Para esse tipo de informação, temos a **curva de aprendizagem**
  - Ainda treinamos no conjunto de treino e testamos no de teste
  - Só que dessa vez aumentamos gradativamente o conjunto de treino



Fonte: AIMA. R&N.

Avaliação de Desempenho

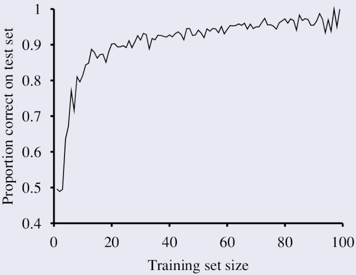
Curva de aprendizagem

- Ex: suponha que temos 100 exemplos no total
  - Dividimos aleatoriamente entre treino e teste

Avaliação de Desempenho

Curva de aprendizagem

- Ex: suponha que temos 100 exemplos no total
  - Dividimos aleatoriamente entre treino e teste
  - Começamos com 1 exemplo no conjunto de treino e 99 no de teste
  - Aumentamos gradativamente o conjunto de treino (reduzindo o de teste), até chegar a 99
  - A cada passo, medimos seu sucesso no conjunto de teste

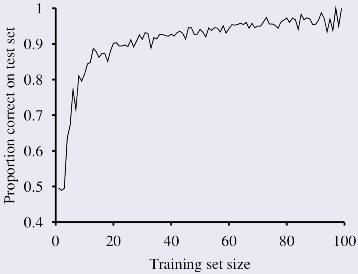


Fonte: AIMA. R&N.

Avaliação de Desempenho

Curva de aprendizagem

- A curva mostra então a variação temporal do desempenho em termos de experiência
  - O quanto aprendemos com a experiência

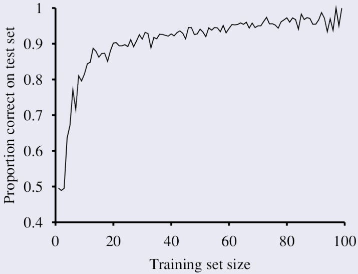


Fonte: AIMA. R&N.

Avaliação de Desempenho

Curva de aprendizagem

- A curva mostra então a variação temporal do desempenho em termos de experiência
  - O quanto aprendemos com a experiência
- Nesse exemplo, mostra a média de 20 repetições
  - Para cada tamanho, em que dividimos aleatoriamente entre treino e teste



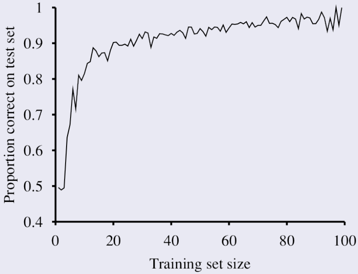
Fonte: AIMA. R&N.

  - E podemos ver o algoritmo se estabilizando, na medida em que cresce o conjunto de treino

Avaliação de Desempenho

Curva de aprendizagem

- E precisa o passo ser sempre de 1 em 1 exemplo?

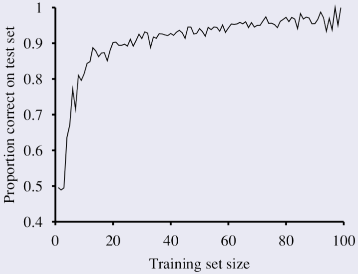


Fonte: Adaptado de AIMA. R&N.

Avaliação de Desempenho

Curva de aprendizagem

- E precisa o passo ser sempre de 1 em 1 exemplo?
  - Não. Podemos fazer de *n* em *n*
  - Ex: Uma geração (algoritmos genéticos), uma época (RNA) etc

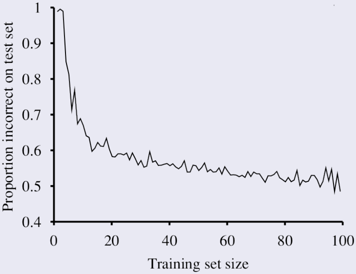


Fonte: Adaptado de AIMA. R&N.

Avaliação de Desempenho

Curva de aprendizagem

- E precisa o passo ser sempre de 1 em 1 exemplo?
  - Não. Podemos fazer de *n* em *n*
  - Ex: Uma geração (algoritmos genéticos), uma época (RNA) etc
- Alternativamente, podemos mostrar a taxa de erro $T_E = 1 - T_S$, em vez de sucesso



Fonte: Adaptado de AIMA. R&N.

Avaliação de Desempenho

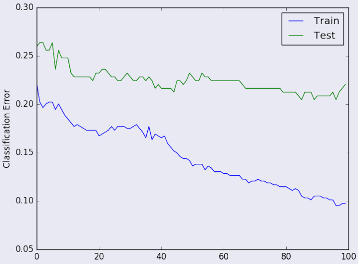
Curva de aprendizagem

- Mas a grande utilidade da curva aparece quando mostramos a taxa de erro (ou de sucesso) tanto no conjunto de treino quanto no de teste

Avaliação de Desempenho

Curva de aprendizagem

- Mas a grande utilidade da curva aparece quando mostramos a taxa de erro (ou de sucesso) tanto no conjunto de treino quanto no de teste
  - Podemos então ver o quão bem o modelo está aprendendo (no conjunto de treino)
  - E o quão bem ele está generalizando (no conjunto de teste)
    - A habilidade de ter bom desempenho em dados inéditos é chamada de **generalização**



Fonte: [9]

Avaliação de Desempenho

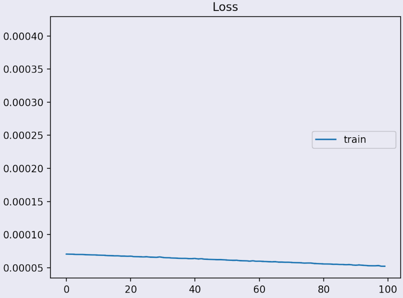
Curva de aprendizagem

- Subajuste (*underfitting*)
  - O modelo não conseguiu ter um erro suficientemente pequeno no treino
    - Não conseguiu aprender com o treino
    - Precisamos apenas da curva no conjunto de treino

Avaliação de Desempenho

Curva de aprendizagem

- Subajuste (*underfitting*)
  - O modelo não conseguiu ter um erro suficientemente pequeno no treino
    - Não conseguiu aprender com o treino
    - Precisamos apenas da curva no conjunto de treino
  - Caracterizada por uma linha plana ou com ruído

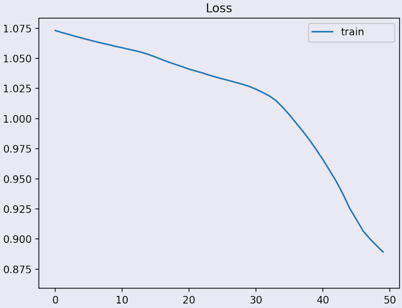


Fonte: Adaptada de [6]

Avaliação de Desempenho

Curva de aprendizagem

- Subajuste (*underfitting*)
  - Ou então por um erro decrescente que não estabiliza
  - Indicando que o modelo ainda pode aprender mais, mas foi interrompido prematuramente



Fonte: Adaptada de [6]

Avaliação de Desempenho

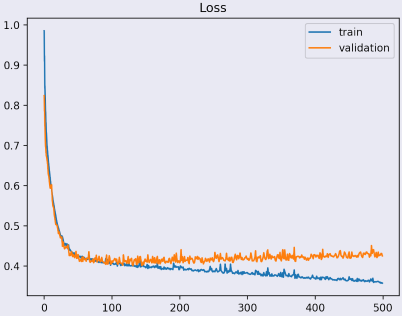
Curva de aprendizagem

- Superajuste (*overfitting*)
  - O modelo aprendeu os dados bem demais
    - Inclusive seu ruído e flutuações aleatórias

Avaliação de Desempenho

Curva de aprendizagem

- Superajuste (*overfitting*)
  - O modelo aprendeu os dados bem demais
    - Inclusive seu ruído e flutuações aleatórias
  - Assim, sua capacidade de generalização reduz
    - O espaço entre erro de treino e teste é grande demais

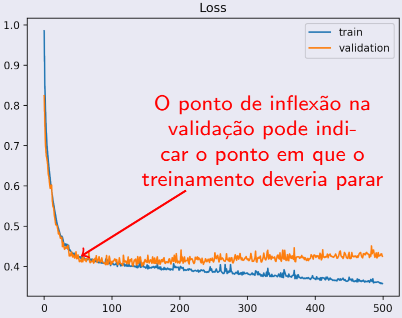


Fonte: [6]

Avaliação de Desempenho

Curva de aprendizagem

- Superajuste (*overfitting*)
  - O modelo aprendeu os dados bem demais
    - Inclusive seu ruído e flutuações aleatórias
  - Assim, sua capacidade de generalização reduz
    - O espaço entre erro de treino e teste é grande demais

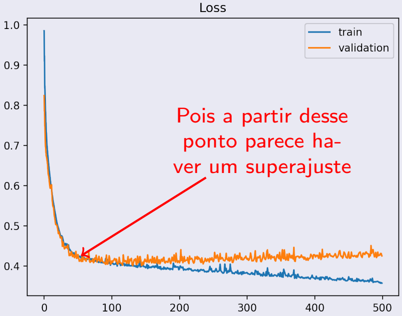


Fonte: [6]

Avaliação de Desempenho

Curva de aprendizagem

- Superajuste (*overfitting*)
  - O modelo aprendeu os dados bem demais
    - Inclusive seu ruído e flutuações aleatórias
  - Assim, sua capacidade de generalização reduz
    - O espaço entre erro de treino e teste é grande demais



Fonte: [6]

Avaliação de Desempenho

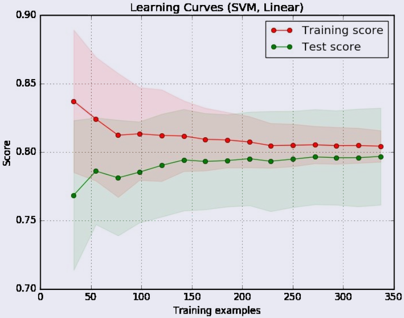
Curva de aprendizagem

- Bom ajuste (*good fit*)
  - Nosso objetivo – entre um subajuste e um superajuste

Avaliação de Desempenho

Curva de aprendizagem

- Bom ajuste (*good fit*)
  - Nosso objetivo – entre um subajuste e um superajuste
  - Identificado por curvas de treino e teste que crescem (ou decrescem, conforme a métrica) até um ponto de estabilidade



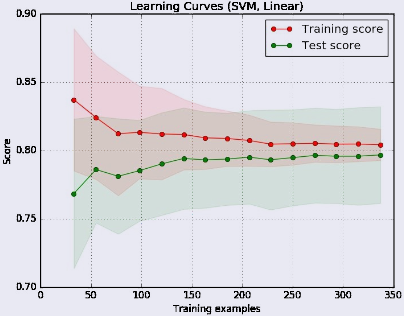
Fonte: `https://stats.stackexchange.com/questions/220827/how-to-know-if-a-learning-curve-from-svm-model-suffers-from-bias-or-variance`

  - Com uma distância pequena entre seus valores finais

Avaliação de Desempenho

Curva de aprendizagem

- Bom ajuste (*good fit*)
  - O erro no treino, contudo, será quase sempre menor que no teste
    - Devemos esperar alguma distância entre as curvas → sua **distância de generalização**

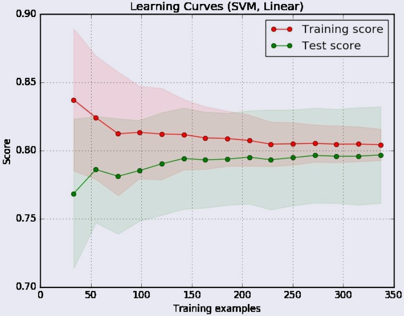


Fonte: `https://stats.stackexchange.com/questions/220827/how-to-know-if-a-learning-curve-from-svm-model-suffers-from-bias-or-variance`

Avaliação de Desempenho

Curva de aprendizagem

- Bom ajuste (*good fit*)
  - O erro no treino, contudo, será quase sempre menor que no teste
    - Devemos esperar alguma distância entre as curvas → sua **distância de generalização**
  - Cuidado!
    - Continuar a treinar um bom ajuste provavelmente levará a um superajuste



Fonte: https://stats.stackexchange.com/questions/220827/how-to-know-if-a-learning-curve-from-svm-model-suffers-from-bias-or-variance

Avaliação de Desempenho

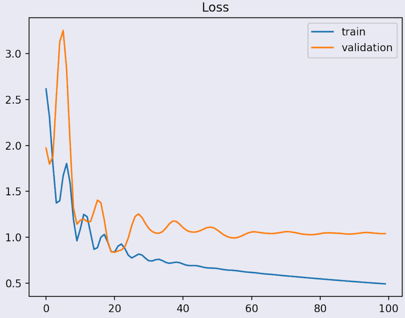
Curva de aprendizagem

- Conjunto de treino não representativo
  - O conjunto de treino possui distribuição diferente da do de teste
    - Talvez por ser pequeno demais, em relação ao de teste

Avaliação de Desempenho

Curva de aprendizagem

- Conjunto de treino não representativo
  - O conjunto de treino possui distribuição diferente da do de teste
    - Talvez por ser pequeno demais, em relação ao de teste
  - Ambas as curvas mostram melhoria
    - Contudo, permanecem bastante separadas



Fonte: [6]

Avaliação de Desempenho

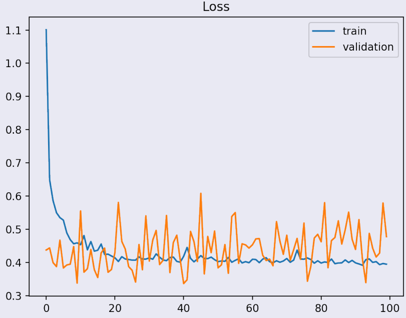
Curva de aprendizagem

- Conjunto de teste não representativo
  - O conjunto de teste não fornece informação suficiente para avaliar o modelo
    - Talvez por ser pequeno demais, em relação ao de treino

Avaliação de Desempenho

Curva de aprendizagem

- Conjunto de teste não representativo
  - O conjunto de teste não fornece informação suficiente para avaliar o modelo
    - Talvez por ser pequeno demais, em relação ao de treino
  - A curva de treino parece se ajustar
    - Contudo, a de teste apresenta ruído em torno da de treino



Fonte: [6]

Avaliação de Desempenho

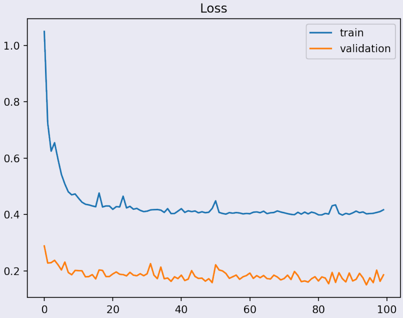
Curva de aprendizagem

- Conjunto de teste não representativo (cont.)
  - Pode também ocorrer que a curva de teste seja menor que a de treino
    - Indicando que o conjunto de teste é mais fácil para o modelo que o de treino

Avaliação de Desempenho

Curva de aprendizagem

- Conjunto de teste não representativo (cont.)
  - Pode também ocorrer que a curva de teste seja menor que a de treino
    - Indicando que o conjunto de teste é mais fácil para o modelo que o de treino



Fonte: [6]

Avaliação de Desempenho

Conjunto de Validação

- Testamos o modelo, e agora?
  - Poderia ser melhor? Não sabemos

Avaliação de Desempenho

Conjunto de Validação

- Testamos o modelo, e agora?
  - Poderia ser melhor? Não sabemos
- Em geral, modelos possuem hiperparâmetros escolhidos de antemão
  - Taxas de aprendizado, cruzamentos, valores de corte etc
  - Como saber se foram a escolha correta?

Avaliação de Desempenho

Conjunto de Validação

- Testamos o modelo, e agora?
  - Poderia ser melhor? Não sabemos
- Em geral, modelos possuem hiperparâmetros escolhidos de antemão
  - Taxas de aprendizado, cruzamentos, valores de corte etc
  - Como saber se foram a escolha correta?
- Fazemos modificações sistemáticas nos hiperparâmetros e testamos novamente o modelo
  - Escolhendo então o melhor conjunto de hiperparâmetros

Avaliação de Desempenho

Conjunto de Validação

- Quando usamos um conjunto de testes com esse fim, o chamamos de **Conjunto de Validação**
  - Pois estamos usando para validar diferentes modelos
  - Fazemos escolhas baseadas no resultado desse conjunto

Avaliação de Desempenho

Conjunto de Validação

- Quando usamos um conjunto de testes com esse fim, o chamamos de **Conjunto de Validação**
  - Pois estamos usando para validar diferentes modelos
  - Fazemos escolhas baseadas no resultado desse conjunto
- E qual o problema com ele?
  - O problema é que o erro relatado nele não é mais independente do modelo testado
  - E é de suma importância que o conjunto de teste não seja usado de modo algum na definição do modelo

Conjunto de Validação: Viés otimista

- O resultado apresentará um viés positivo
  - O erro observado será provavelmente menor do que o erro real em exemplos não vistos
  - Otimista, nesse caso, significa pensar que o erro é menor do que realmente é

Avaliação de Desempenho

Conjunto de Validação: Viés otimista

- O resultado apresentará um viés positivo
  - O erro observado será provavelmente menor do que o erro real em exemplos não vistos
  - Otimista, nesse caso, significa pensar que o erro é menor do que realmente é
- Ex: Digamos que temos 2 modelos m_1 e m_2
  - Ambos com o mesmo erro real (na população) $E_{pop}(m_1) = E_{pop}(m_2) = 0.5$
  - Naturalmente, desconhecemos tal erro

Avaliação de Desempenho

Conjunto de Validação: Viés otimista

- Para simplificar, suponha que testamos o modelo em um único ponto
  - Medimos então seu erro nesse ponto
  - Os erros medidos e_{m_1} e e_{m_2} são então estimativas para $E_{pop}(m_1)$ e $E_{pop}(m_2)$

Avaliação de Desempenho

Conjunto de Validação: Viés otimista

- Para simplificar, suponha que testamos o modelo em um único ponto
  - Medimos então seu erro nesse ponto
  - Os erros medidos e_{m_1} e e_{m_2} são então estimativas para $E_{pop}(m_1)$ e $E_{pop}(m_2)$
- Vamos assumir que ambos e_{m_1} e e_{m_2}:
  - São uniformes em $[0, 1]$ (Seu valor esperado é $\mathbb{E}(e_{m_i}) = 0,5$)
  - São independentes um do outro

Avaliação de Desempenho

Conjunto de Validação: Viés otimista

- Suponha agora que escolhemos um dos modelos $m \in \{m_1, m_2\}$, de acordo com o valor do erro
  - Pegamos aquele em que $e = min(e_{m_1}, e_{m_2})$
  - Ou seja, usamos as estimativas já calculadas para fazer uma escolha

Avaliação de Desempenho

Conjunto de Validação: Viés otimista

- Suponha agora que escolhemos um dos modelos $m \in \{m_1, m_2\}$, de acordo com o valor do erro
  - Pegamos aquele em que $e = min(e_{m_1}, e_{m_2})$
  - Ou seja, usamos as estimativas já calculadas para fazer uma escolha
- Qual a probabilidade $P(e < 0.5)$ de e ser menor do que seu valor esperado real $\mathbb{E}(e_{m_i}) = 0,5$?
  - Lembre que tanto m_1 quanto m_2 tinham erros reais $E_{pop}(m_1) = E_{pop}(m_2) = 0.5$

Avaliação de Desempenho

Conjunto de Validação: Viés otimista

$$
\begin{aligned}
P(e < 0.5) &= P((e_{m_1} < 0.5) \vee (e_{m_2} < 0.5)) \\
&= P(e_{m_1} < 0.5) + P(e_{m_2} < 0.5) - P((e_{m_1} < 0.5) \wedge (e_{m_2} < 0.5)) \\
&= P(e_{m_1} < 0.5) + P(e_{m_2} < 0.5) - P(e_{m_1} < 0.5) \times P(e_{m_2} < 0.5) \\
&= 0,5 + 0,5 - 0,25 = 0,75
\end{aligned}
$$

Avaliação de Desempenho

Conjunto de Validação: Viés otimista

$$
\begin{aligned}
P(e < 0.5) &= P((e_{m_1} < 0.5) \vee (e_{m_2} < 0.5)) \\
&= P(e_{m_1} < 0.5) + P(e_{m_2} < 0.5) - P((e_{m_1} < 0.5) \wedge (e_{m_2} < 0.5)) \\
&= P(e_{m_1} < 0.5) + P(e_{m_2} < 0.5) - P(e_{m_1} < 0.5) \times P(e_{m_2} < 0.5) \\
&= 0,5 + 0,5 - 0,25 = 0,75
\end{aligned}
$$

- Ou seja, a probabilidade de escolhermos um modelo com erro $e < 0.5$, quando seu erro real é 0.5, é de 75%
  - E estaremos achando que o erro é menor do que muito provavelmente é

Avaliação de Desempenho

Treino - Validação - Teste

- Que fazer então?
  - Dividir o conjunto de dados em conjunto de treino, validação e teste

Avaliação de Desempenho

Treino - Validação - Teste

- Que fazer então?
  - Dividir o conjunto de dados em conjunto de treino, validação e teste
- Fazer isso de forma aleatória e independente
  - O conjunto de validação precisa ser diferente do de treino para poder otimizar parâmetros ou escolher melhor o modelo
  - E o de teste precisa ser diferente dos demais para obtermos uma estimativa mais confiável da taxa de erro real

Avaliação de Desempenho

Definindo um modelo

Define($X = \{\vec{x}_1, \ldots, \vec{x}_n\}$)**: modelo + estimativa de erro**

$X_{ts} \leftarrow$ subconjunto aleatório de X
$X_v \leftarrow$ subconjunto aleatório de $X - X_{ts}$
$X_{tr} \leftarrow X - X_{ts} - X_v$
para cada *variação* m_i *do modelo* **faça**
  Treine m_i em X_{tr}
  $e_{i_v} \leftarrow$ erro de m_i em X_v
$m_j \leftarrow$ modelo com menor e_{i_v} do passo anterior
Treine novamente m_j em $X_{tr} \cup X_v$
$e_{m_j} \leftarrow$ erro de m_j em X_{ts}
Treine novamente m_j em $X_{tr} \cup X_v \cup X_{ts}$
retorna m_j *e* e_{m_j}

Avaliação de Desempenho

Definindo um modelo

Define($X = \{\vec{x}_1, \ldots, \vec{x}_n\}$)**: modelo + estimativa de erro**

$X_{ts} \leftarrow$ subconjunto aleatório de X

$X_v \leftarrow$ subconjunto aleatório de $X - X_{ts}$

$X_{tr} \leftarrow X - X_{ts} - X_v$

para cada *variação m_i do modelo* **faça**

  Treine m_i em X_{tr}

  $e_{i_v} \leftarrow$ erro de m_i em X_v

$m_j \leftarrow$ modelo com menor e_{i_v} do passo anterior

Treine novamente m_j em $X_{tr} \cup X_v$

$e_{m_j} \leftarrow$ erro de m_j em X_{ts}

Treine novamente m_j em $X_{tr} \cup X_v \cup X_{ts}$

retorna m_j *e* e_{m_j}

Separa aleatoriamente o conjunto de teste (normalmente 1/3 dos dados)

Avaliação de Desempenho

Definindo um modelo

Define($X = \{\vec{x}_1, \ldots, \vec{x}_n\}$): **modelo + estimativa de erro**

$X_{ts} \leftarrow$ subconjunto aleatório de X

$X_v \leftarrow$ subconjunto aleatório de $X - X_{ts}$

$X_{tr} \leftarrow X - X_{ts} - X_v$

para cada *variação* m_i *do modelo* **faça**

  Treine m_i em X_{tr}

  $e_{i_v} \leftarrow$ erro de m_i em X_v

$m_j \leftarrow$ modelo com menor e_{i_v} do passo anterior

Treine novamente m_j em $X_{tr} \cup X_v$

$e_{m_j} \leftarrow$ erro de m_j em X_{ts}

Treine novamente m_j em $X_{tr} \cup X_v \cup X_{ts}$

retorna m_j e e_{m_j}

Dos dados restantes, separa aleatoriamente o conjunto de validação

Avaliação de Desempenho

Definindo um modelo

Define($X = \{\vec{x}_1, \ldots, \vec{x}_n\}$): **modelo + estimativa de erro**

$X_{ts} \leftarrow$ subconjunto aleatório de X
$X_v \leftarrow$ subconjunto aleatório de $X - X_{ts}$
$X_{tr} \leftarrow X - X_{ts} - X_v$
para cada *variação* m_i *do modelo* **faça**
  Treine m_i em X_{tr}
  $e_{i_v} \leftarrow$ erro de m_i em X_v
$m_j \leftarrow$ modelo com menor e_{i_v} do passo anterior
Treine novamente m_j em $X_{tr} \cup X_v$
$e_{m_j} \leftarrow$ erro de m_j em X_{ts}
Treine novamente m_j em $X_{tr} \cup X_v \cup X_{ts}$
retorna m_j *e* e_{m_j}

O que sobrou é o conjunto de treino

Avaliação de Desempenho

Definindo um modelo

Define($X = \{\vec{x}_1, \ldots, \vec{x}_n\}$)**: modelo + estimativa de erro**

$X_{ts} \leftarrow$ subconjunto aleatório de X
$X_v \leftarrow$ subconjunto aleatório de $X - X_{ts}$
$X_{tr} \leftarrow X - X_{ts} - X_v$
para cada *variação m_i do modelo* **faça**
  Treine m_i em X_{tr}
  $e_{i_v} \leftarrow$ erro de m_i em X_v

$m_j \leftarrow$ modelo com menor e_{i_v} do passo anterior
Treine novamente m_j em $X_{tr} \cup X_v$
$e_{m_j} \leftarrow$ erro de m_j em X_{ts}
Treine novamente m_j em $X_{tr} \cup X_v \cup X_{ts}$
retorna m_j *e* e_{m_j}

Varia cada hiper-parâmetro do modelo, de modo a ajustá-lo

Avaliação de Desempenho

Definindo um modelo

Define$(X = \{\vec{x}_1, \ldots, \vec{x}_n\})$**: modelo + estimativa de erro**

$X_{ts} \leftarrow$ subconjunto aleatório de X
$X_v \leftarrow$ subconjunto aleatório de $X - X_{ts}$
$X_{tr} \leftarrow X - X_{ts} - X_v$
para cada *variação* m_i *do modelo* **faça**
  Treine m_i em X_{tr}
  $e_{i_v} \leftarrow$ erro de m_i em X_v
$m_j \leftarrow$ modelo com menor e_{i_v} do passo anterior
Treine novamente m_j em $X_{tr} \cup X_v$
$e_{m_j} \leftarrow$ erro de m_j em X_{ts}
Treine novamente m_j em $X_{tr} \cup X_v \cup X_{ts}$
retorna m_j *e* e_{m_j}

Treinando cada um no conjunto de treino

Avaliação de Desempenho

Definindo um modelo

Define($X = \{\vec{x}_1, \ldots, \vec{x}_n\}$): **modelo + estimativa de erro**

$X_{ts} \leftarrow$ subconjunto aleatório de X
$X_v \leftarrow$ subconjunto aleatório de $X - X_{ts}$
$X_{tr} \leftarrow X - X_{ts} - X_v$
para cada *variação* m_i *do modelo* **faça**
  Treine m_i em X_{tr}
  $e_{i_v} \leftarrow$ erro de m_i em X_v

$m_j \leftarrow$ modelo com menor e_{i_v} do passo anterior
Treine novamente m_j em $X_{tr} \cup X_v$
$e_{m_j} \leftarrow$ erro de m_j em X_{ts}
Treine novamente m_j em $X_{tr} \cup X_v \cup X_{ts}$
retorna m_j e e_{m_j}

E medindo seu erro no conjunto de validação

Avaliação de Desempenho

Definindo um modelo

Define($X = \{\vec{x}_1, \ldots, \vec{x}_n\}$)**: modelo + estimativa de erro**

$X_{ts} \leftarrow$ subconjunto aleatório de X
$X_v \leftarrow$ subconjunto aleatório de $X - X_{ts}$
$X_{tr} \leftarrow X - X_{ts} - X_v$
para cada *variação m_i do modelo* **faça**
  Treine m_i em X_{tr}
  $e_{i_v} \leftarrow$ erro de m_i em X_v
$m_j \leftarrow$ modelo com menor e_{i_v} do passo anterior
Treine novamente m_j em $X_{tr} \cup X_v$
$e_{m_j} \leftarrow$ erro de m_j em X_{ts}
Treine novamente m_j em $X_{tr} \cup X_v \cup X_{ts}$
retorna m_j *e* e_{m_j}

Seleciona o modelo com o menor erro de validação

Avaliação de Desempenho

Definindo um modelo

Define$(X = \{\vec{x}_1, \ldots, \vec{x}_n\})$**: modelo + estimativa de erro**

$X_{ts} \leftarrow$ subconjunto aleatório de X
$X_v \leftarrow$ subconjunto aleatório de $X - X_{ts}$
$X_{tr} \leftarrow X - X_{ts} - X_v$
para cada *variação* m_i *do modelo* **faça**
  Treine m_i em X_{tr}
  $e_{i_v} \leftarrow$ erro de m_i em X_v
$m_j \leftarrow$ modelo com menor e_{i_v} do passo anterior
Treine novamente m_j em $X_{tr} \cup X_v$
$e_{m_j} \leftarrow$ erro de m_j em X_{ts}
Treine novamente m_j em $X_{tr} \cup X_v \cup X_{ts}$
retorna m_j *e* e_{m_j}

Retreina o modelo com os dados de treino + validação

Avaliação de Desempenho

Definindo um modelo

Define$(X = \{\vec{x}_1, \ldots, \vec{x}_n\})$: **modelo** $+$ **estimativa de erro**

$X_{ts} \leftarrow$ subconjunto aleatório de X
$X_v \leftarrow$ subconjunto aleatório de $X - X_{ts}$
$X_{tr} \leftarrow X - X_{ts} - X_v$
para cada *variação* m_i *do modelo* **faça**
  Treine m_i em X_{tr}
  $e_{i_v} \leftarrow$ erro de m_i em X_v

$m_j \leftarrow$ modelo com menor e_{i_v} do passo anterior
Treine novamente m_j em $X_{tr} \cup X_v$
$e_{m_j} \leftarrow$ erro de m_j em X_{ts}
Treine novamente m_j em $X_{tr} \cup X_v \cup X_{ts}$
retorna m_j *e* e_{m_j}

Com isso maximiza a quantidade de dados usados na geração do modelo a ser testado com dados inéditos

Avaliação de Desempenho

Definindo um modelo

Define($X = \{\vec{x}_1, \ldots, \vec{x}_n\}$)**: modelo + estimativa de erro**

$X_{ts} \leftarrow$ subconjunto aleatório de X
$X_v \leftarrow$ subconjunto aleatório de $X - X_{ts}$
$X_{tr} \leftarrow X - X_{ts} - X_v$
para cada *variação* m_i *do modelo* **faça**
  Treine m_i em X_{tr}
  $e_{i_v} \leftarrow$ erro de m_i em X_v

$m_j \leftarrow$ modelo com menor e_{i_v} do passo anterior
Treine novamente m_j em $X_{tr} \cup X_v$
$e_{m_j} \leftarrow$ erro de m_j em X_{ts}
Treine novamente m_j em $X_{tr} \cup X_v \cup X_{ts}$
retorna m_j *e* e_{m_j}

Mede o erro no conjunto de teste. Essa é a estimativa do modelo para dados não vistos

Definindo um modelo

Define($X = \{\vec{x}_1, \ldots, \vec{x}_n\}$)**: modelo + estimativa de erro**

$X_{ts} \leftarrow$ subconjunto aleatório de X
$X_v \leftarrow$ subconjunto aleatório de $X - X_{ts}$
$X_{tr} \leftarrow X - X_{ts} - X_v$
para cada *variação* m_i *do modelo* **faça**
  Treine m_i em X_{tr}
  $e_{i_v} \leftarrow$ erro de m_i em X_v

$m_j \leftarrow$ modelo com menor e_{i_v} do passo anterior
Treine novamente m_j em $X_{tr} \cup X_v$
$e_{m_j} \leftarrow$ erro de m_j em X_{ts}
Treine novamente m_j em $X_{tr} \cup X_v \cup X_{ts}$
retorna m_j *e* e_{m_j}

A versão final é então retreinada com todos os dados, esperando que isso a deixe melhor (há sempre o perigo de *overfitting*)

Avaliação de Desempenho

Definindo um modelo

Define$(X = \{\vec{x}_1, \ldots, \vec{x}_n\})$**: modelo + estimativa de erro**

$X_{ts} \leftarrow$ subconjunto aleatório de X
$X_v \leftarrow$ subconjunto aleatório de $X - X_{ts}$
$X_{tr} \leftarrow X - X_{ts} - X_v$
para cada *variação m_i do modelo* **faça**
  Treine m_i em X_{tr}
  $e_{i_v} \leftarrow$ erro de m_i em X_v
$m_j \leftarrow$ modelo com menor e_{i_v} do passo anterior
Treine novamente m_j em $X_{tr} \cup X_v$
$e_{m_j} \leftarrow$ erro de m_j em X_{ts}
Treine novamente m_j em $X_{tr} \cup X_v \cup X_{ts}$
retorna m_j e e_{m_j}

Retorna a versão final, com a estimativa do erro

Avaliação de Desempenho

Definindo um modelo

Define($X = \{\vec{x}_1, \ldots, \vec{x}_n\}$)**: modelo + estimativa de erro**

$X_{ts} \leftarrow$ subconjunto aleatório de X
$X_v \leftarrow$ subconjunto aleatório de $X - X_{ts}$
$X_{tr} \leftarrow X - X_{ts} - X_v$
para cada *variação m_i do modelo* **faça**
    Treine m_i em X_{tr}
    $e_{i_v} \leftarrow$ erro de m_i em X_v
$m_j \leftarrow$ modelo com menor e_{i_v} do passo anterior
Treine novamente m_j em $X_{tr} \cup X_v$
$e_{m_j} \leftarrow$ erro de m_j em X_{ts}
Treine novamente m_j em $X_{tr} \cup X_v \cup X_{ts}$
retorna m_j *e* e_{m_j}

Note que essa estimativa foi tomada antes de usar o conjunto de teste para treino

Avaliação de Desempenho

Definindo um modelo

Define($X = \{\vec{x}_1, \ldots, \vec{x}_n\}$)**: modelo + estimativa de erro**

$X_{ts} \leftarrow$ subconjunto aleatório de X

$X_v \leftarrow$ subconjunto aleatório de $X - X_{ts}$

$X_{tr} \leftarrow X - X_{ts} - X_v$

para cada *variação* m_i *do modelo* **faça**

  Treine m_i em X_{tr}

  $e_{i_v} \leftarrow$ erro de m_i em X_v

$m_j \leftarrow$ modelo com menor e_{i_v} do passo anterior

Treine novamente m_j em $X_{tr} \cup X_v$

$e_{m_j} \leftarrow$ erro de m_j em X_{ts}

Treine novamente m_j em $X_{tr} \cup X_v \cup X_{ts}$

retorna m_j *e* e_{m_j}

Ou seja, nenhum viés foi introduzido nessa estimativa

Avaliação de Desempenho

Dados limitados

- Essa metodologia, no entanto, consome dados
  - Nenhum problema, se tivermos muitos

Avaliação de Desempenho

Dados limitados

- Essa metodologia, no entanto, consome dados
  - Nenhum problema, se tivermos muitos
- E se esse não for o caso? Temos então um dilema
  - Para definir um bom modelo, precisamos usar o máximo de dados possível para treino e validação
  - Para obter uma boa estimativa de erro, precisamos usar o máximo possível para teste

Avaliação de Desempenho

Dados limitados

- Essa metodologia, no entanto, consome dados
  - Nenhum problema, se tivermos muitos
- E se esse não for o caso? Temos então um dilema
  - Para definir um bom modelo, precisamos usar o máximo de dados possível para treino e validação
  - Para obter uma boa estimativa de erro, precisamos usar o máximo possível para teste
  - Lembrando que reduzir demais algum dos conjuntos pode torná-lo não representativo da população

Desempenho – Dados limitados

Amostragem estratificada

- Em geral, não temos como dizer se uma amostra é representativa ou não
  - Mas podemos verificar se cada classe no conjunto de dados inteiro está representada, aproximadamente na mesma proporção, nos subconjuntos de treino, validação e teste
  - Procedimento chamado **estratificação**

Desempenho – Dados limitados

Amostragem estratificada

- Em geral, não temos como dizer se uma amostra é representativa ou não
  - Mas podemos verificar se cada classe no conjunto de dados inteiro está representada, aproximadamente na mesma proporção, nos subconjuntos de treino, validação e teste
  - Procedimento chamado **estratificação**
- Ex: Suponha que o conjunto de dados esteja distribuído em 4 classes
  - $n_A = 200$, $n_B = 400$, $n_C = 600$, $n_D = 800$

Desempenho – Dados limitados

Amostragem estratificada

- E que iremos separar $\approx \frac{1}{3}$ para teste. Então escolhemos aleatoriamente:
  - 66 dados de A ($\approx \frac{200}{3}$), 133 de B, 200 de C e 266 de D
  - Num total de 665 ($\approx \frac{2000}{3} = 666,66...$)

Desempenho – Dados limitados

Amostragem estratificada

- E que iremos separar $\approx \frac{1}{3}$ para teste. Então escolhemos aleatoriamente:
  - 66 dados de A ($\approx \frac{200}{3}$), 133 de B, 200 de C e 266 de D
  - Num total de 665 ($\approx \frac{2000}{3} = 666,66\ldots$)
- Note que no conjunto original sobraram
  - 134 em A ($\approx 200 \times \frac{2}{3}$), 267 em B, 400 em C e 534 em D
  - E ambos os conjuntos possuem aproximadamente a mesma distribuição de dados da amostra original

Desempenho – Dados limitados

Repeated Holdout

- Uma estratégia mais geral para reduzir qualquer viés introduzido pela amostra é repetir o processo todo de treino, validação e teste várias vezes
  - Sempre com amostras aleatórias diferentes

Desempenho – Dados limitados

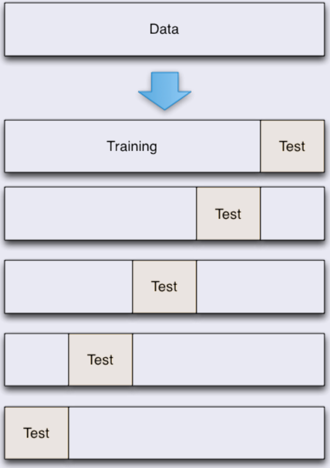
Repeated Holdout

- Uma estratégia mais geral para reduzir qualquer viés introduzido pela amostra é repetir o processo todo de treino, validação e teste várias vezes
  - Sempre com amostras aleatórias diferentes
- Procedimento conhecido como **holdout repetido**
  - A cada iteração, os dados são aleatoriamente separados (possivelmente com estratificação) em conjuntos de treino, validação e teste
  - O erro total será então a média dos erros medidos em cada iteração

Desempenho – Dados limitados

K-fold Cross-Validation

- Variante do *repeated holdout*
  - Divida os dados de treino + validação em k subconjuntos de aproximadamente o mesmo tamanho
  - A cada iteração, separa-se um dos diferentes k subconjuntos para validação, treinando nos demais
  - O resultado final da validação será a média dos k subgrupos



Fonte: [10]

Desempenho – Dados limitados

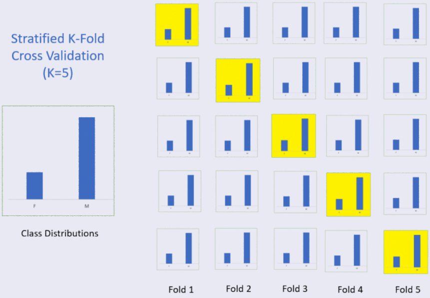
K-fold Cross-Validation

- E temos apenas treino + teste?
  - O procedimento é o mesmo, com ou sem validação
  - Em geral, usa-se $k = 10$, mas outros valores podem ser usados

Desempenho – Dados limitados

K-fold Cross-Validation

- E temos apenas treino + teste?
  - O procedimento é o mesmo, com ou sem validação
  - Em geral, usa-se $k = 10$, mas outros valores podem ser usados
- Os resultados podem ser melhorados com estratificação
  - *Stratified K-fold Cross-Validation*



Fonte: [11]

Desempenho – Dadoslimitados

Leave-one-out Cross-Validation

- Variante do *k-fold*, onde $k = n$, o número de elementos no conjunto de dados
  - A cada iteração separamos um único exemplo para teste, treinando nos demais
  - A estimativa de erro final será a média dos n testes feitos

Desempenho – Dados limitados

Leave-one-out Cross-Validation

- Variante do *k-fold*, onde $k = n$, o número de elementos no conjunto de dados
  - A cada iteração separamos um único exemplo para teste, treinando nos demais
  - A estimativa de erro final será a média dos n testes feitos
- Problema: não pode ser estratificado
  - Imagine um conjunto completamente aleatório, com elementos de 2 classes
    - Em que 50% dos exemplos são da classe A e os outros 50% da classe B

Desempenho – Dados limitados

Leave-one-out Cross-Validation

- O melhor que um classificador conseguirá fazer será predizer a classe majoritária
  - Lembre que os dados são completamente aleatórios → não há qualquer padrão
  - Isso nos leva a uma taxa de erro de 50%

Desempenho – Dados limitados

Leave-one-out Cross-Validation

- O melhor que um classificador conseguirá fazer será predizer a classe majoritária
  - Lembre que os dados são completamente aleatórios $\rightarrow$ não há qualquer padrão
  - Isso nos leva a uma taxa de erro de 50%
- Mas em cada conjunto de treino do *leave-one-out*, a classe oposta à do exemplo de teste é majoritária
  - As predições estarão sempre incorretas, quando testadas
  - A taxa de erro será de 100%

Desempenho – Dados limitados

Bootstrapping

- Método baseado na amostragem dos dados, com reposição, para formação do conjunto de treino

Desempenho – Dados limitados

Bootstrapping

- Método baseado na amostragem dos dados, com reposição, para formação do conjunto de treino
- Considere um conjunto de dados X com n exemplos
  - Para o conjunto de treino X_{tr}, tomamos n amostras de X, com reposição
  - Usamos as instâncias que não foram escolhidas em X para teste, ou seja, $X_{ts} = X - X_{tr}$
    - Por ser com reposição, é bastante provável que algumas instâncias em X não sejam escolhidas para X_{tr}

Desempenho – Dados limitados

Bootstrapping

- Bastante provável quanto?
  - A cada vez, a chance de uma instância particular ser escolhida é $1/n$
  - Terá então uma chance de $1 - \frac{1}{n}$ de não ser escolhida
  - Em n eventos, a chance de não ser escolhida será $\left(1 - \frac{1}{n}\right)^n \approx e^{-1} = 0,368$

Desempenho – Dados limitados

Bootstrapping

- Bastante provável quanto?
  - A cada vez, a chance de uma instância particular ser escolhida é $1/n$
  - Terá então uma chance de $1 - \frac{1}{n}$ de não ser escolhida
  - Em n eventos, a chance de não ser escolhida será $\left(1 - \frac{1}{n}\right)^n \approx e^{-1} = 0,368$
- Então espera-se que, em um conjunto grande, cerca de 36,8% sobrem para teste
  - Com 63,2% escolhidos para treino (contendo repetições)

Desempenho – Dadas limitados

Bootstrapping

- Treinar e testar nesses conjuntos, contudo, nos leva a uma estimativa pessimista do erro
  - Temos apenas 63% dos exemplos para treino (10-*fold* teria 90%)

Desempenho – Dados limitados

Bootstrapping

- Treinar e testar nesses conjuntos, contudo, nos leva a uma estimativa pessimista do erro
  - Temos apenas 63% dos exemplos para treino (10-*fold* teria 90%)
- Para compensar, combinamos as taxas de erro no conjunto de treino e teste
  - Combinamos a otimista com a pessimista
  - A taxa de erro fica então $T_E = 0,632 \times T_{E_{teste}} + 0,368 \times T_{E_{treino}}$

Desempenho – Dados limitados

Bootstrapping

- Repetimos todo o procedimento várias vezes
  - Reconstruindo os conjuntos de treino e teste
  - Executando e testando o método neles
  - Retornando a média dos T_Es como estimativa final de erro

Desempenho – Dados limitados

Bootstrapping

- Repetimos todo o procedimento várias vezes
  - Reconstruindo os conjuntos de treino e teste
  - Executando e testando o método neles
  - Retornando a média dos T_Es como estimativa final de erro
- Embora seja o melhor método para conjuntos muito pequenos, tem problemas
  - Considere o mesmo conjunto de dados de antes (conjunto aleatório com 2 classes)

Desempenho – Dados limitados

Bootstrapping

- Considere agora que temos um método que aprende perfeitamente o conjunto de treino
  - De alguma forma, a existência de repetições levou a um padrão que foi aprendido
  - Assim, $T_{E_{treino}} = 0$

Desempenho – Dados limitados

Bootstrapping

- Considere agora que temos um método que aprende perfeitamente o conjunto de treino
  - De alguma forma, a existência de repetições levou a um padrão que foi aprendido
  - Assim, $T_{E_{treino}} = 0$
- Lembrando que $T_{E_{teste}} = 0.5$
  - Pois o conjunto é totalmente aleatório – não há padrão

Desempenho – Dados limitados

Bootstrapping

- Considere agora que temos um método que aprende perfeitamente o conjunto de treino
  - De alguma forma, a existência de repetições levou a um padrão que foi aprendido
  - Assim, $T_{E_{treino}} = 0$
- Lembrando que $T_{E_{teste}} = 0.5$
  - Pois o conjunto é totalmente aleatório – não há padrão
- Então $T_E = 0,632 \times 0.5 + 0,368 \times 0 = 0,316$
  - 31,6% é bastante otimista, considerando que a taxa real é de 50%

Referências

1. Witten, I.H.; Frank, E. (2005): Data Mining: Practical Machine Learning Tools and Techniques. Elsevier. 2a ed.
2. Russell, S.; Norvig P. (2010): Artificial Intelligence: A Modern Approach. Prentice Hall. 3a ed.
3. Goodfellow, I.; Bengio, Y.; Courville, A. (2016): Deep Learning. MIT Press.
4. Alpaydın, E. (2010): Introduction to Machine Learning. MIT Press. 2 ed.
5. Murphy, K. P. (2012): Machine Learning: A Probabilistic Perspective. MIT Press.
6. `https://machinelearningmastery.com/learning-curves-for-diagnosing-machine-learning-model-performance/`

Referências

7. https://en.wikipedia.org/wiki/Learning_curve
8. https://en.wikipedia.org/wiki/Learning_curve_(machine_learning)
9. https://machinelearningmastery.com/avoid-overfitting-by-early-stopping-with-xgboost-in-python/
10. http://scott.fortmann-roe.com/docs/MeasuringError.html
11. https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/
12. https://www.thoughtco.com/stratified-sampling-3026731