Avaliação da Aprendizagem

Paulo Trigo Silva

Avaliação – "a chave para o sucesso"

Como avaliar?

1. Construir modelo baseado num conjunto de **treino** de **elevada dimensão**.
2. Aplicar o modelo a um conjunto de **teste** de **elevada dimensão**.
3. Analisar o erro resultante da aplicação ao conjunto de treino.

Então que dificuldades existem?

A. Obter conjuntos de treino e teste de **elevada dimensão**.

B. Fazer **análise custo/benefício** independentemente do algoritmo usado.

Paulo Trigo Silva

Taxa de erro e dados (de treino) para construir o modelo

- A "taxa de erro" é, em geral, usada para medir o desempenho
  - de um classificador
- O classificador prevê a classe de cada instância
  - se a previsão estiver correcta considera-se ***sucesso*** senão será ***erro***
- A "taxa de erro" é a proporção de erros num conjunto de instâncias
  - é uma medida do desempenho global do classificador
- ... mas estamos interessados no desempenho sobre novos dados
  - não estamos interessados no erro do conjunto de treino
  - ... pois o modelo foi construído a partir desse mesmo conjunto!
- Assim a questão é saber se "**a taxa de erro de dados antigos é, ou não, um bom indicador da taxa de erro sobre novos dados?**"

Paulo Trigo Silva

Necessidade de existirem conjuntos de treino e de teste

“A taxa de erro de dados antigos é, ou não, um bom indicador da taxa de erro sobre novos dados?”

Se os “dados antigos” foram utilizados no processo de aprendizagem para treinar o classificador, então a resposta é NÃO.

Porquê? Porque qualquer estimativa de erro baseada nos dados de treino será sempre demasiado optimista!

Assim, **para estimar o desempenho da aprendizagem é preciso**:

- calcular a taxa de erro sobre um conjunto de dados que não tenha participado na aprendizagem e que se designa por “**conjunto de teste**”,

- assumir que os conjuntos de treino e o conjunto de teste constituem amostras representativas do problema em causa.

Paulo Trigo Silva

... conjuntos de treino, validação e teste

- O conjunto de teste não pode participar na aprendizagem
  - há métodos com duas fases – *construir* modelo e *afinar* parâmetros
  - ... e.g., construir árvore e realizar poda
  - os dados de teste não podem participar em nenhuma dessas fases
- Os dados usados na aprendizagem
  - não podem usar-se para estimar a taxa de erro sobre dados futuros
- Se aprendizagem tem 2 fases então existem 3 conjuntos de dados
  - conjunto de treino para "construir o modelo", e.g., árvore de decisão
  - conjunto de validação para "afinar parâmetros", e.g., realizar poda
  - conjunto de teste para "estimar taxa de erro sobre dados futuros"
- O mesmo conjunto de treino pode ser usado por vários métodos
- ... devem usar-se diferentes conjuntos de treino, validação e teste

Paulo Trigo Silva

O processo – usando o maior número de dados possível

- O objectivo da avaliação é escolher o método de aprendizagem
  - com menor taxa de erro, i.e., mais adequado ao problema
- Depois de escolhido o método podem usar-se os dados de teste
  - os dados de teste podem juntar-se aos de treino
  - ... maximizando assim a quantidade de dados usadas no modelo final
- Em síntese temos o seguinte processo por cada método a avaliar
  - [1] usar dados de treino para construir o modelo
  - [2] usar dados de validação para afinar parâmetros
  - [3] usar dados de teste para estima taxa de erro sobre dados futuros
- ... comparar as taxas de erro e escolher um (ou mais) métodos
  - [4] unir os dados de treino e de teste
  - [5] construir modelo final (para ser colocado em produção)

... a dificuldade em obter grandes volumes de dados

- Se existe um grande volume de dados disponível tudo é simples
  - consideram-se 3 sub-conjuntos independentes e de grande dimensão
  - ... treino, validação e teste
- A dificuldade acontece com volumes de dados de baixa dimensão
  - muitas vezes os dados, treino e teste, são classificados manualmente
  - ... isto limita a quantidade de dados disponível
- ... como usar da melhor forma um conjunto de dados reduzido?
  - uma parte retira-se para teste; designado por procedimento de *holdout*
  - restante usa-se para treino; se necessário, com separação de validação
- Existe aqui um dilema
  - para aprender usar o maior número de dados no conjunto de treino
  - para estimar erro usar o maior número de dados no conjunto de teste

Paulo Trigo Silva

Validação com reserva (estratificada) – (*stratified*) *holdout*

- Método *holdout* – reservar conjunto teste e usar restante em treino
  - na prática é usual reserva 1/3 para teste (restantes 2/3 para treino)
- ... mas como escolher o conjunto de teste?
  - e.g., uma classe só com exemplos de teste não consegue treinar-se!
- Método *stratified holdout* – tentar salvaguardar representatividade
  - calcular a percentagem de cada classe no *dataset* original
  - construir conjunto de treino mantendo a percentagem de cada classe
  - ... escolher de modo aleatório cada exemplo do conjunto de treino
- e.g., *dataset* com 200 exemplos da classe A, 300 da B e 100 da C
  - a distribuição do *dataset* original é [A: 2/6=1/3, B: 3/6=1/2, C: 1/6]
  - ... reservemos para o conjunto de treino 1/3 dos exemplos (= 200)
  - ... assim, os exemplos de treino por classe serão: [A: 67, B: 100, C: 33]

Paulo Trigo Silva

... validação com reserva repetida – *repeated holdout*

- Mitigar efeito de desvios (*bias*) de uma única escolha de *holdout*
  - adoptando a ideia de *repeated holdout*
- i.e., repetir, em várias iterações, todo o processo de treino e teste
  - com diferentes conjuntos de treino e de teste
- ... em cada iteração considerar novo conjunto de teste
  - considerar um certa proporção para teste e.g., 1/3 do *dataset* original
  - construir cada conjunto de teste usando e.g., *stratified holdout*
  - estimar o erro como a média das taxas de erro das várias iterações
- Nestas abordagens cada conjunto desempenha um único papel
  - i.e., o conjunto de teste nunca se usa para treino
  - ... alternativa é trocar papéis, i.e., treinar com dados teste, testar com dados treino e fazer média dos 2 testes; só viável para partição 50:50
  - ... mas é preferível usar mais de metade dos dados para treino!

Paulo Trigo Silva

Método de validação cruzada (*cross-validation*)

- Variação do método da troca de papéis com mais de 2 partições
  - define-se um número fixo de partições (*folds*)
- Se considerarmos K partições do *dataset* original
  - fazer K iterações de treino e teste
  - escolher, em cada iteração, uma partição de teste ainda não utilizada
  - ... i.e., cada partição é teste em 1 iteração e treino em k – 1 iterações
- O método *threefold (3-fold) cross-validation* considera 3 partições
  - i.e., executa 3 iterações do processo de treino e teste
  - ... em cada iteração tem 2/3 dos dados para treino e 1/3 para teste
- ... em geral utiliza-se o *stratified 3-fold cross-validation*
  - ou seja, garante-se que as partições mantêm proporção da classe

Paulo Trigo Silva

Método "standard" de avaliação da taxa de erro

- É "standard" aplicar o *stratified 10-fold cross-validation*
  - construir, de modo aleatório, 10 partições do *dataset* completo
  - cada partição mantém proporção da classe (relativamente ao *dataset*)
  - usar 9 partições (i.e. 90% do *dataset*) como dados de treino; usar 1 partição como dados de teste e calcular a sua taxa de erro
  - executar aprendizagem 10 vezes variando sempre a partição de teste
  - calcular média das 10 taxas de erro para obter um estimativa global
- Porquê usar 10 partições?
  - resultados experimentais e alguns resultados teóricos suportam o "10"
  - testes também sugerem que abordagem *stratified* melhora resultados
- É também "standard" repetir 10 × *stratified 10-fold cross-validation*
  - e calcular a média dos resultados
  - i.e., executar 100 vezes o algoritmo de aprendizagem em *dataset* com 9/10 da dimensão original; ... é muito pesado fazer uma boa avaliação!

Paulo Trigo Silva

Outro método de avaliação – "*leave-one-out*"

- *leave-one-out* é simplesmente *N-fold cross-validation*
  - onde *N* é a dimensão do *dataset* original
- Ou seja, em cada iteração é deixado apenas 1 exemplo para teste
  - cada aprendizagem executa com todos os restantes exemplos
  - ... e é avaliada como correcta / incorrecta pelo exemplo que foi retirado
  - a média dos resultados das *N* avaliações é a estimativa final do erro
- Vantagem: permite usar o maior conjunto de treino possível
  - obtém o máximo partido de um *dataset* de pequena dimensão
- Vantagem: não exige construir partições aleatórias
  - o método é determinístico pelo que não é necessário repetir o processo
- Desvantagem: elevado custo computacional
  - executar *N* vezes o algoritmo é inviável em *dataset* de grande dimensão

... "*leave-one-out*" – vantagens e desvantagens

- *Recordar*: método é estratificado (*stratified*) se conjuntos de treino
  - mantiverem a proporção das classes em relação ao *dataset* original
- O método "*leave-one-out*" não pode ser estratificado
  - ou pior ainda, garante uma amostra não estratificada!
- O pior cenário (embora muito artificial) do "*leave-one-out*" é
  - um *dataset* de exemplos completamente aleatórios
  - ... com duas classes ambas com o mesmo número de exemplos
- ... com exemplos aleatórios o melhor que um classificador consegue
  - é prever a classe maioritária, logo taxa de erro esperada de 50%
- ... mas, nesse cenário, com o método "leave-one-out"
  - a classe oposta à do exemplo de teste estará sempre em maioria
  - ... assim a previsão será sempre incorrecta, logo taxa de erro de 100%!

Outro método de avaliação – "*bootstrap*"

- Baseado no processo estatístico de amostragem com reposição
  - uma mesma instância pode ser escolhida (amostrada) diversas vezes
- Dado um *dataset* de dimensão *N*
  - seleccionar, do *dataset* original, *N* instâncias
  - ... mas cada instância seleccionada permanece no *dataset* original
  - ... pelo que a mesma instância pode ser seleccionada várias vezes
  - as *N* instâncias seleccionadas constituem o (novo) *dataset* de treino
  - as instâncias que não ocorrem no conjunto de treino serão de teste
- Então qual é a percentagem esperada de exemplos de teste?
  - ou, qual a probabilidade de um exemplo não ser escolhido para treino?

Paulo Trigo Silva

... "*bootstrap*" – que percentagem de exemplos de teste?

- Ou, que probabilidade da instância não ser escolhida para treino?
  - cada instância tem 1/*N* probabilidade de ser escolhida para treino
  - ... logo, 1 – 1/*N* é a probabilidade de não ser escolhida para treino
- ... eventos de escolha são independentes (pois há reposição)
  - logo com *N* escolhas independentes a probabilidade de não ser treino
  - será: $(1 - 1/N)^N \approx e^{-1} \approx 0.368$, com *e* base do logaritmo natural
- Ou seja, dado um *dataset* original o *bootstrap* deve originar
  - um conjunto de teste com 36.8% da dimensão do *dataset* original
  - logo, com 63.2% de exemplos diferentes no conjunto de treino
  - ... recordar que treino e *dataset* original têm mesma dimensão

Paulo Trigo Silva

... "*bootstrap*" – compensar perspectiva pessimista do erro

- No *bootstrap* a estimativa de erro é bastante pessimista
  - pois o treino apenas utiliza 63% dos exemplos do *dataset*
  - ... comparar com os 90% usados no método *10-fold cross-validation*!
- Compensar a perspectiva pessimista combinando com optimismo
  - o mais optimista possível é considerar o erro do conjunto de treino
  - ... compensar fazendo: $erro = 0.632 \times erro_{teste} + 0.368 \times erro_{treino}$
  - i.e., pesar erro de teste (treino) com valor esperado de treino (teste)
- O processo de *bootstrap* é repetido diversas vezes
  - para obter o erro final calcular a média dos erros das várias iterações

Paulo Trigo Silva

... "*bootstrap*" – vantagens e desvantagens

- Dado um *dataset* de pequena dimensão
  - o *bootstrap* é, em geral, o melhor escolhido para estimar o erro
  - ... computacionalmente é menos exigente do que o *leave-one-out*
- O pior cenário (embora muito artificial) do "*leave-one-out*" é
  - tal como no pior cenário do *leave-one-out*
  - um *dataset* de exemplos completamente aleatórios
  - ... com duas classes ambas com o mesmo número de exemplos
- Para qualquer método classificação a verdadeira taxa de erro é 50%
  - mas, um método que memorize (*overfitting*) o conjunto de treino
  - ... terá $erro_{treino} = 0$,
  - i.e., $erro = 0.632 \times erro_{teste} + 0.368 \times 0 \Leftrightarrow erro = 0.632 \times 0.5 + 0 = 0.316$
  - ou seja, um erro de 31.6% que já é "demasiado optimista" face aos 50%

Paulo Trigo Silva

Tipo de erro e custo associado

- Diferentes tipos de erro podem originar diferentes custos
  - e.g., custo de enviar correio promocional para quem não responde é bastante menor do que o negócio perdido por não enviar esse correio
  - e.g., custo de fazer um empréstimo a quem não consegue pagar é bastante maior do que o de recusar um empréstimo a quem iria pagar
- Um problema de classificação com 2 classes (e.g., *yes*, *no*)
  - tem 4 possibilidades: 2 verdadeiras e 2 falsas
- ... as 2 verdadeiras correspondem a classificações correctas
  - verdadeiro positivo (*true positive*), e.g., classificado *yes* e é *yes*
  - verdadeiro negativo (*true negative*), e.g., classificado *no* e é *no*
- ... as 2 negativas correspondem a classificações incorrectas
  - falso positivo (*false positive*), e.g., classificado *yes* e é *no*
  - falso negativo (*false negative*), e.g., classificado *no* e é *yes*

Métricas sobre os tipos de erro (com 2 classes)

Admita-se o conjunto das classes:
{ *SIM*, *NÃO* }

		classe predita (algoritmo)	
		SIM	*NÃO*
classe exacta (humano)	*SIM*	positivo (P)	falso negativo (FN)
	NÃO	falso positivo (FP)	negativo (N)

Taxa de <u>verdadeiros</u> positivos (*true positive rate*) ≡ P / (P + FN)

proporção dos positivos correctamente classificados entre todos os positivos

Taxa dos <u>falsos</u> positivos (*false positive rate*) ≡ FP / (FP + N)

proporção dos negativos incorrectamente classificados entre todos os negativos

Taxa de <u>sucesso</u> (*success rate*) ≡ (P + N) / (P + N + FP + FN)

proporção dos correctamente classificados entre todos os classificados

Taxa de <u>erro</u> (*error rate*) ≡ 1 – "tSucesso" = 1 – [(P + N) / (P + N + FP + FN)]

Matriz de confusão (*confusion matrix*)

- O resultado dos testes à classificação é, em geral, apresentado
  - como uma matriz quadrada com as classes em linha e coluna
  - ... designada por matriz de confusão (*confusion matrix*)
- Cada elemento da matriz de confusão apresenta o número de
  - exemplos de teste onde a classe exacta é a linha e a predita é a coluna
- Um bom resultado no teste corresponde a
  - valores elevados na diagonal principal
  - valores baixos (idealmente zero) nos restantes elementos da matriz

---

Exemplo:

		Classe Predita			
		A	B	C	TOTAL
Classe Correcta	A	88	10	2	100
	B	14	40	6	60
	C	18	10	12	40
	TOTAL	120	60	20	

Qual a taxa de sucesso do teste representado nesta matriz de confusão?

Matriz de confusão e classificador aleatório

		Classe Predita			
		A	B	C	TOTAL
Classe Correcta	A	88	10	2	100
	B	14	40	6	60
	C	18	10	12	40
	TOTAL	120	60	20	

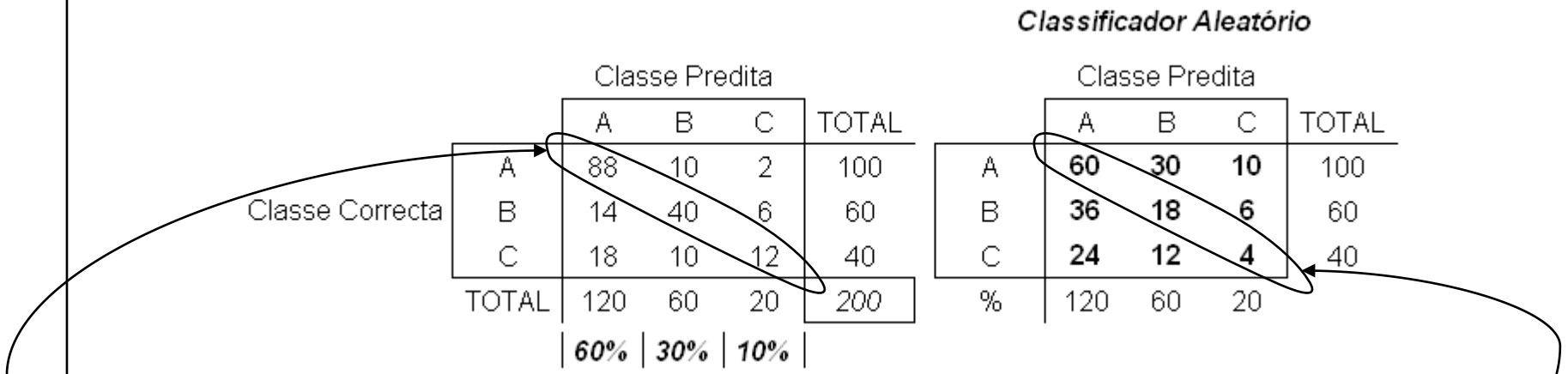
taxa de sucesso =
(88 + 40 + 12) / 200 =
140 / 200 =
70%

- Mas os 70% serão medida adequada do sucesso do classificador?
  - este classificador prediz [A: 120 (60%), B: 60 (30%), C: 20 (10%)]
  - ... mas qual é o sucesso esperado de um classificador aleatório que considere aqueles mesmos totais, i.e., [A: 60%, B: 30%, C: 10%]?
- ... classifica os 100 As (linha 1) mantendo proporção [60%, 30%, 10%]
  - ... e o mesmo para os 60 Bs (linha 2) e para os 40 Cs (linha 3)

Qual será a matriz de confusão desse classificador aleatório?

Paulo Trigo Silva

Matriz de confusão e estatística Kappa



número de sucessos do classificador aleatório = 60 + 18 + 4 = **82**

88 + 40 + 12

número extra de sucessos face ao classificador aleatório = 140 – 82 = **58**

número máximo de sucessos extra seria = 200 – 82 = **118**

o valor da ***estatística Kappa*** *é* = 58 / 118 = **49.2%**

Paulo Trigo Silva

... estatística Kappa

- A estatística Kappa mede a concordância (homogeneidade) entre
  - a previsão obtida pelo classificador e os valores presentes no *dataset*
  - enquanto corrige as concordâncias que se obtêm por mero acaso
- O valor máximo de Kappa (100%) ocorre quando o classificador
  - coincide em todas as previsões com as classes do conjunto de teste
  - i.e., só a diagonal principal da matriz de confusão é diferente de zero
  - ... todas as restantes componentes têm valor zero
  - e as previsões que resultam do acaso não alteram esta concordância
- O valor mínimo de Kappa (0%) ocorre quando o classificador
  - falha em todas as previsões, i.e., diagonal principal toda a zero
  - ou, todas as previsões resultam do acaso
  - ... i.e., tem a mesma soma de diagonal que o classificador aleatório

Classificação sensível ao custo (*cost-sensitive*)

- Xxx
  - xxx

Xxx

- Xxx
  - xxx

Xxx

- Xxx
  - xxx