aprendizagem automática baseada em dados – aspetos gerais; árvores de decisão

Ciências ULisboa

extração de modelos a partir de dados

há uma grande diversidade de formas de aprendizagem

aprendizagem para melhorar o desempenho do agente

em IIA vamos ver apenas algumas das que são baseadas em dados

- a partir de um conjunto de exemplos (casos típicos) o que é possível inferir?

em função da informação que guia a aprendizagem

aprendizagem supervisionada

os exemplos têm caraterísticas (atributos) e o resultado

aprendizagem não supervisionada

os exemplos apenas têm as caraterísticas

- agrupamento
- condicionamento

aprendizagem por reforço

apenas há um sinal (+/-) que pode ser diferido
(ex: 2 pontos recebidos após a vitória num jogo de xadrez)

aprendizagem supervisionada

dado um **conjunto de treino** de *N* exemplos

$$(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_N, y_N)$$

em que $\boldsymbol{x}_i$ é um vetor de atributos e y_i o resultado desse exemplo, dado por uma função desconhecida $y_i = f(\boldsymbol{x}_i)$

pretende-se encontrar uma função *h* que aproxima *f*

h denomina-se a hipótese

"supervisionda" porque se considera que o resultado, y_i, é fornecido por um professor

aprendizagem robusta

interessa obter resultados corretos em mais exemplos do que os usados na aprendizagem – **generalização**

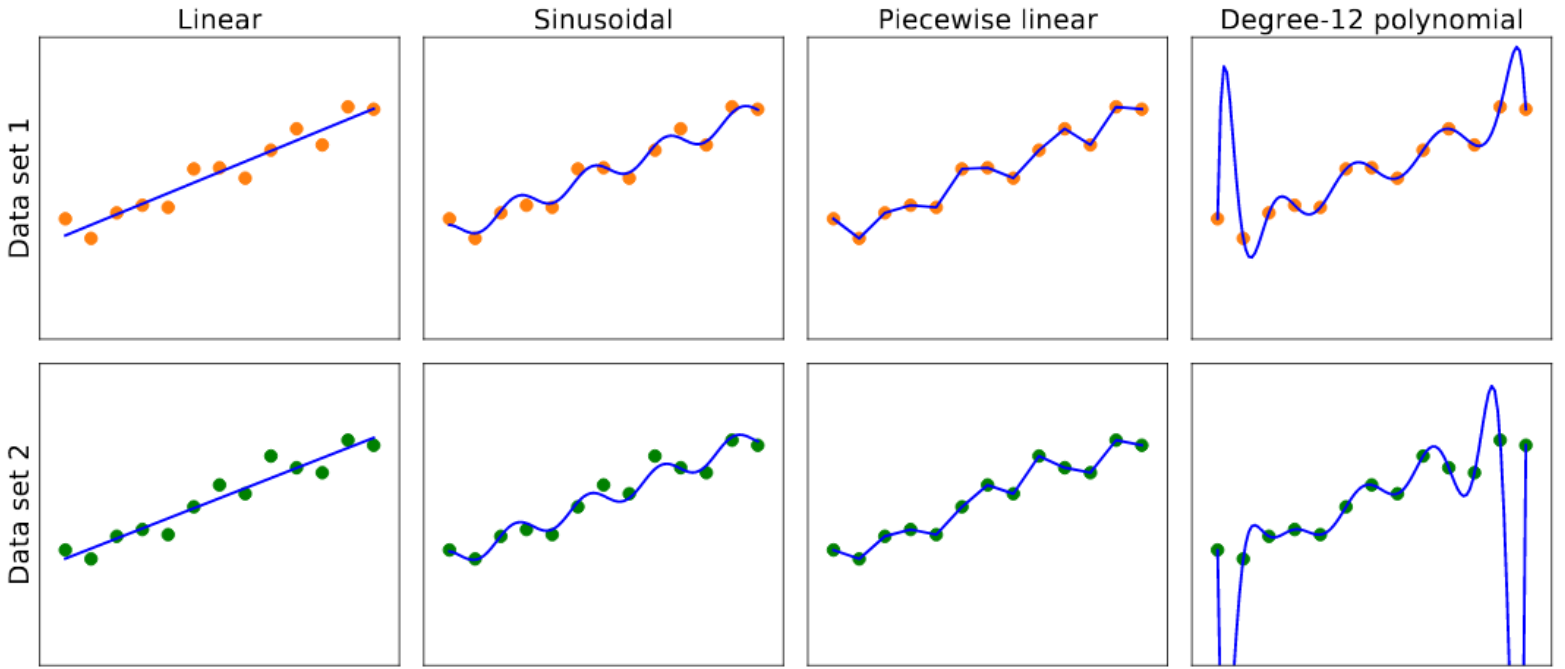
para isso, a avaliação da aprendizagem é feita com um outro conjunto de exemplos, **conjunto de teste**, distinto do conjunto de treino

e isto não é nenhuma indireta sobre a avaliação dos estudantes de IIA!

o espaço de hipóteses

descubra as diferenças nas h_j

dois conjuntos de treino obtidos da mesma função unidimensional $f(x)$



qual a h_j que tem melhor desempenho nos dois conjuntos?

Ciências ULisboa

a propósito de desempenho...

como se comparam desempenhos de diferentes hipóteses?

- precisão (entre 0 e 1)
- erro médio

→ avaliados sobre o conjunto de teste!

e em caso de desempenhos semelhantes de várias hipóteses?

- conhecimento pericial, ou (na sua ausência)
- hipótese mais simples – lâmina de Ockham

tipo de dados – tipo de problema

resultado (*y*)

há diversas outras medidas de desempenho!

- valores discretos – problema de **classificação**

  desempenho:

  precisão = nº ex. corretamente classificados / total ex.

- valores contínuos – problema de **regressão**

  desempenho: erro quadrático médio (varia inv. ao desempenho)

indução de árvores de decisão

um dos modelos mais simples de aprendizagem automática

árvore de decisão produz resultados por

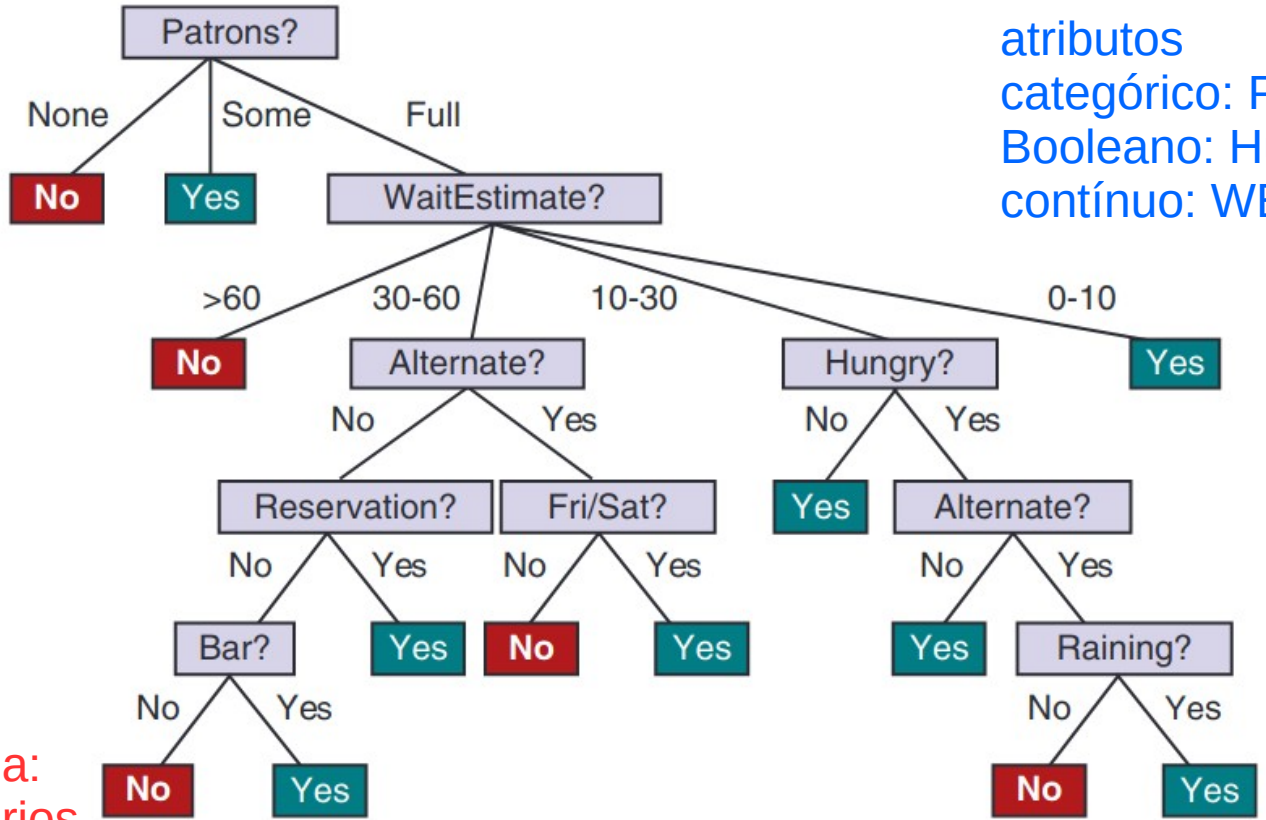
uma série de testes aos valores dos atributos, A_i

- um atributo em cada nó

estrutura em árvore

- as folhas têm os resultados a retornar nas diversas situações

exemplo – espera num restaurante



atributos
categórico: Patrons
Booleano: Hungry
contínuo: WE (txf. categórico)

árvore Booleana:
resultados binários

indução, a partir de exemplos

conjunto de treino

Example	Input Attributes										Output
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
$\mathbf{x}_1$	*Yes*	*No*	*No*	*Yes*	*Some*	*$$$*	*No*	*Yes*	*French*	*0–10*	$y_1 = Yes$
$\mathbf{x}_2$	*Yes*	*No*	*No*	*Yes*	*Full*	*$*	*No*	*No*	*Thai*	*30–60*	$y_2 = No$
$\mathbf{x}_3$	*No*	*Yes*	*No*	*No*	*Some*	*$*	*No*	*No*	*Burger*	*0–10*	$y_3 = Yes$
$\mathbf{x}_4$	*Yes*	*No*	*Yes*	*Yes*	*Full*	*$*	*Yes*	*No*	*Thai*	*10–30*	$y_4 = Yes$
$\mathbf{x}_5$	*Yes*	*No*	*Yes*	*No*	*Full*	*$$$*	*No*	*Yes*	*French*	*>60*	$y_5 = No$
$\mathbf{x}_6$	*No*	*Yes*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Italian*	*0–10*	$y_6 = Yes$
$\mathbf{x}_7$	*No*	*Yes*	*No*	*No*	*None*	*$*	*Yes*	*No*	*Burger*	*0–10*	$y_7 = No$
$\mathbf{x}_8$	*No*	*No*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Thai*	*0–10*	$y_8 = Yes$
$\mathbf{x}_9$	*No*	*Yes*	*Yes*	*No*	*Full*	*$*	*Yes*	*No*	*Burger*	*>60*	$y_9 = No$
$\mathbf{x}_{10}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$$$*	*No*	*Yes*	*Italian*	*10–30*	$y_{10} = No$
$\mathbf{x}_{11}$	*No*	*No*	*No*	*No*	*None*	*$*	*No*	*No*	*Thai*	*0–10*	$y_{11} = No$
$\mathbf{x}_{12}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$*	*No*	*No*	*Burger*	*30–60*	$y_{12} = Yes$

modelo de indução

estratégia dividir para conquistar:

escolher o atributo mais importante primeiro

- “mais importante” – o que mais distingue a classificação dos exemplos

repetir recursivamente

- até só haver folhas (decisões)

sôfrega, mas com um bom critério de escolha do atributo consegue ser uma boa aproximação à árvore ótima

a recursão

4 casos possíveis em cada nó

1. todos os exemplos têm a mesma classe – retorna-a
2. há exemplos de várias classes – continua a recursão
3. não há exemplos – retorna a partição do nó ascendente

   não há exemplos de treino com esta combinação de atributos
4. há exemplos mas já foram testados todos os atributos – retorna a partição das classificações desses exemplos

   há erro, ou ruído nos dados, ou não há informação suficiente

entropia

Claude Shannon

usando o conceito de **entropia** (da teoria da informação)

calculada sobre uma v.a. a partir da distribuição probabilística

$$H(X)=\sum_k P(x_k)\log\frac{1}{P(x_k)}=-\sum_k P(x_k)\log P(x_k)$$

com $\log_2$ a unidade é *bit*

ex. moeda equilibrada (*E*)

$$H(E)=-(0{,}5\log_2 0{,}5+0{,}5\log_2 0{,}5)=1\,\text{bit}$$

ex. moeda falsa (*F*) com 99% probabilidade de cara

$$H(F)=-(0{,}99\log_2 0{,}99+0{,}01\log_2 0{,}01)\approx 0{,}08\,\text{bit}$$

caso do restaurante

total de 12 exemplos

6 em que o resultado (*WillWait*) é *Yes*, $p = 6$

6 em que o resultado (*WillWait*) é *No*, $n = 6$

$$\hat{p}(Yes) = \frac{6}{12} = 0{,}5$$

idem para $\hat{p}(No) = 0{,}5$

logo a incerteza do problema é $H(WillWait) = B(0{,}5) = 1\,\text{bit}$

B(*q*) é a entropia de uma v.a. booleana em que uma das probabilidades é *q*

entropia restante

no caso binário

escolhendo um atributo *A* com *d* valores, definem-se *d* subconjuntos, cada um deles com p_k exemplos positivos e com n_k exemplos negativos, com uma entropia $B(p_k/(p_k+n_k))$

escolhendo esse atributo, a entropia (média) restante é

$$HRestante(A) = \sum_{k}^{d} \frac{p_k+n_k}{p+n} B\left(\frac{p_k}{p_k+n_k}\right)$$

escolha do atributo

o atributo mais importante é o que tiver

menor entropia restante

maior ganho de informação

ex. espera em restaurante:

$$HRestante(Patrons) = \left[\frac{2}{12}B\left(\frac{0}{2}\right) + \frac{4}{12}B\left(\frac{4}{4}\right) + \frac{6}{12}B\left(\frac{2}{6}\right)\right] \approx 0{,}459\,\text{bit}$$

verifica-se que, dos 10 atributos, é o que tem menor HRestante

$$HRestante(Type) = \left[\frac{2}{12}B\left(\frac{1}{2}\right) + \frac{2}{12}B\left(\frac{1}{2}\right) + \frac{4}{12}B\left(\frac{2}{4}\right) + \frac{4}{12}B\left(\frac{2}{4}\right)\right] = 1\,\text{bit}$$

escolhido para a raiz da árvore

overfitting (sobreajuste)

aprendizagem demasiado focada nos exemplos

perde capacidade de generalização

em árvores de decisão, poda-se

- poda prévia – limita a profundidade máxima da árvore

  problema: efeito de horizonte (+1 nível podia ser determinante)

- poda posterior – depois da árvore completa eliminam-se nós de modo a simplificar a árvore

  problema: a árvore completa ser muito grande

poda posterior simples

poda de erro reduzido

começando nas folhas

cada nó é substituído pela sua classe mais frequente

se a precisão estimada não for significativamente afetada,

mantém-se a alteração

simples e rápido!

validação cruzada

vimos que se deve separar o conjunto de exemplos em treino e teste – **holdout cross-validation**

mas para melhorar a identificação do ponto de paragem da aprendizagem e melhor estimar o erro de generalização…

k-fold cross-validation

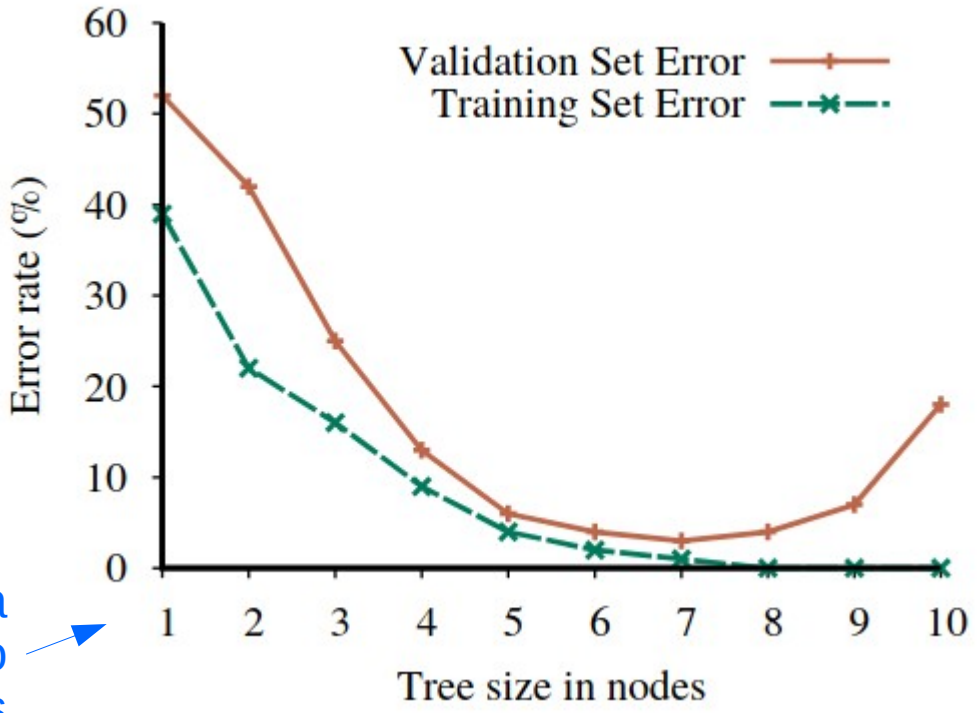
divide-se o conjunto de treino em *k* subconjuntos da mesma #

fazem-se *k* rondas de aprendizagem, uma com cada um dos subconjuntos para validação (teste)

ponto de paragem prematura

no *k*-fold cross-validation

- vai-se verificando o erro médio nos subconjuntos de validação
- quando este erro atinge o mínimo, é o bom ponto de paragem



neste caso da espera no restaurante a dimensão ótima da árvore é 7 nós