Machine Learning I

2. 决策树

WU Xiaokun 吴晓堃

xkun.wu [at] gmail

Contents

餐馆等待问题

假设你慕名跑去一家餐馆

却发现已经爆满。

现在需要做出决定：等还是不等?

假设你慕名跑去一家餐馆

却发现已经爆满。

现在需要做出决定：等还是不等?

- 输出：布尔变量 $WillWait$
- 输入：对状况的描述向量 x

思考：如何描述当前状况，进而对问题建模?

描述向量示例

对输入描述的每个指标通常称为**特征**:

1. $Alternate$: 附近是否有其他合适的替代餐馆
2. Bar: 餐馆是否有舒适的等待区域
3. Fri/Sat: 是否周末
4. $Hungry$: 是否饥饿
5. $Patrons$: 餐馆人数估计（$None, Some, Full$）
6. $Price$: 价格范围（$, $$, $$$）
7. $Raining$: 外面是否下雨
8. $Reservation$: 是否已经订位
9. $Type$: 餐馆类型（$French, Italian, Thai, burger$）
10. $WaitEstimate$: 等待时长估计（0-10, 10-30, 30-60, >60）

思考：问题规模有多大（组合数计算）？

采集样本

Example	Input Attributes										Output
	Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est	*WillWait*
x_1	*Yes*	*No*	*No*	*Yes*	*Some*	*$$$*	*No*	*Yes*	*French*	*0–10*	$y_1 = Yes$
x_2	*Yes*	*No*	*No*	*Yes*	*Full*	*$*	*No*	*No*	*Thai*	*30–60*	$y_2 = No$
x_3	*No*	*Yes*	*No*	*No*	*Some*	*$*	*No*	*No*	*Burger*	*0–10*	$y_3 = Yes$
x_4	*Yes*	*No*	*Yes*	*Yes*	*Full*	*$*	*Yes*	*No*	*Thai*	*10–30*	$y_4 = Yes$
x_5	*Yes*	*No*	*Yes*	*No*	*Full*	*$$$*	*No*	*Yes*	*French*	*>60*	$y_5 = No$
x_6	*No*	*Yes*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Italian*	*0–10*	$y_6 = Yes$
x_7	*No*	*Yes*	*No*	*No*	*None*	*$*	*Yes*	*No*	*Burger*	*0–10*	$y_7 = No$
x_8	*No*	*No*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Thai*	*0–10*	$y_8 = Yes$
x_9	*No*	*Yes*	*Yes*	*No*	*Full*	*$*	*Yes*	*No*	*Burger*	*>60*	$y_9 = No$
x_{10}	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$$$*	*No*	*Yes*	*Italian*	*10–30*	$y_{10} = No$
x_{11}	*No*	*No*	*No*	*No*	*None*	*$*	*No*	*No*	*Thai*	*0–10*	$y_{11} = No$
x_{12}	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$*	*No*	*No*	*Burger*	*30–60*	$y_{12} = Yes$

归纳的核心问题

输入属性所有可能组合的总数：$2^6 \times 3^2 \times 4^2 = 9216$

- 总共观测到的数据数量：12
- 剩下 99.87% 种组合：无法确定正确答案

归纳的核心问题

输入属性所有可能组合的总数：$2^6 \times 3^2 \times 4^2 = 9216$

- 总共观测到的数据数量：12
- 剩下 99.87% 种组合：无法确定正确答案

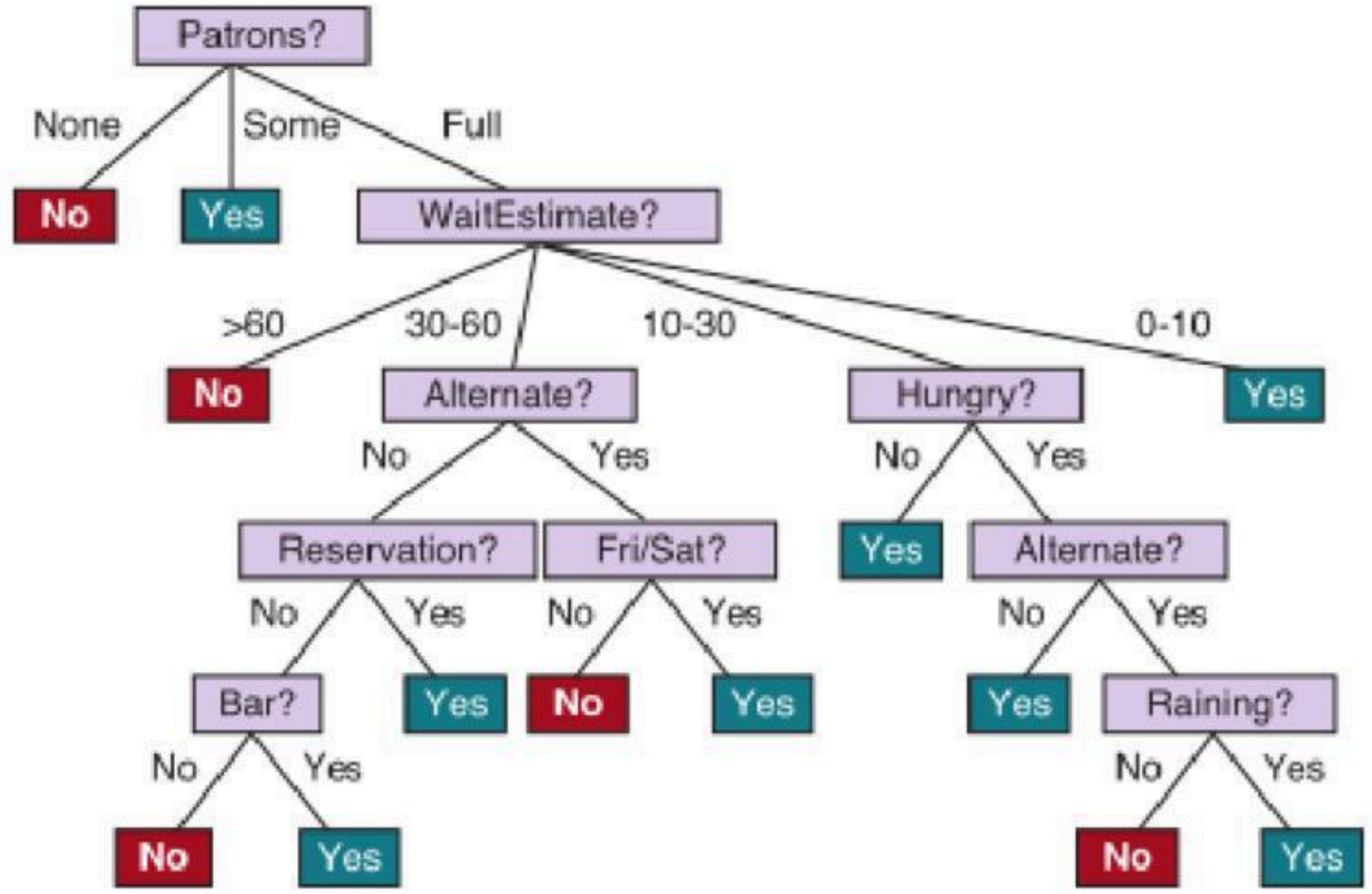
归纳的核心问题：利用非常有限的样本集对智能体行为建模

- 模型必须能够高效、可靠地预测广泛的未知情况
- 故归纳过程必须挖掘决策行为的**内在模式（或逻辑）**

决策树： 表达力

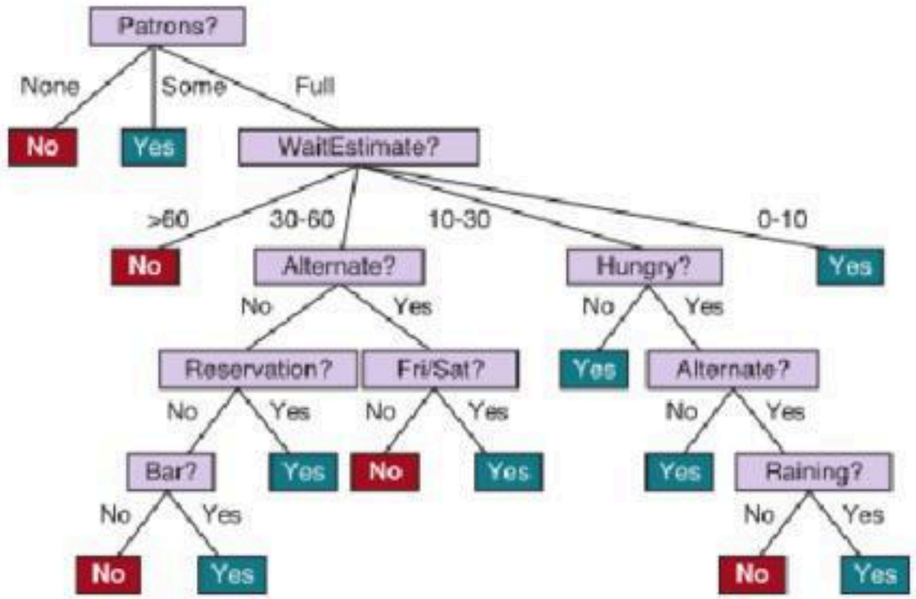
决策树模型

决策树可以看作**函数**：将特征向量映射到单一输出值，即“**决策**”



决策树：工作流程

从根节点出发，经过一系列检测选取分支方向，最终达到叶节点



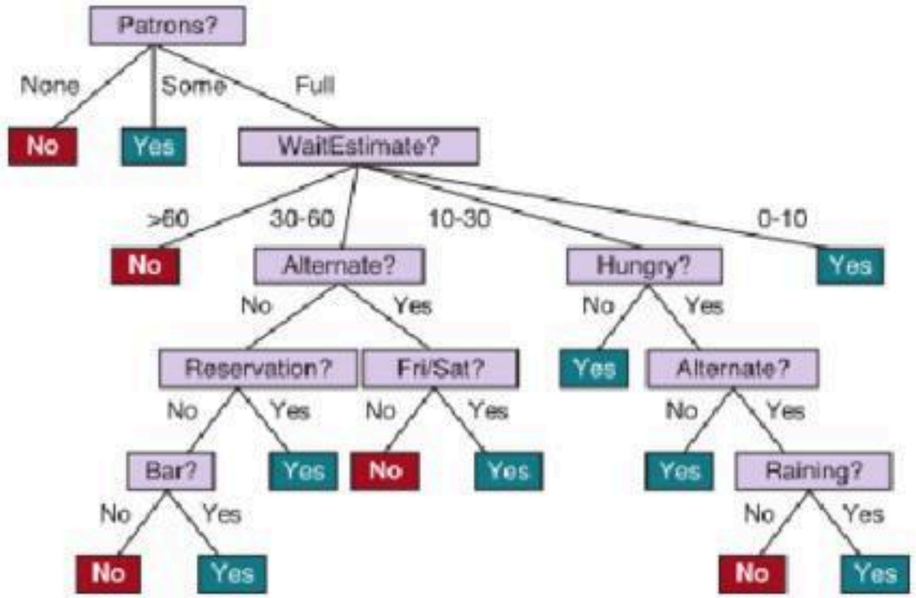
- 内部节点：对某一特征进行取值范围检测
  - 分支：对应特征所有可能取值作为分类标签
- 叶节点：返回最终决策（类别标签）

思考：如何用逻辑语言表示决策树模型?

布尔分类

首先只考虑离散值输入和布尔型输出，即**布尔分类**

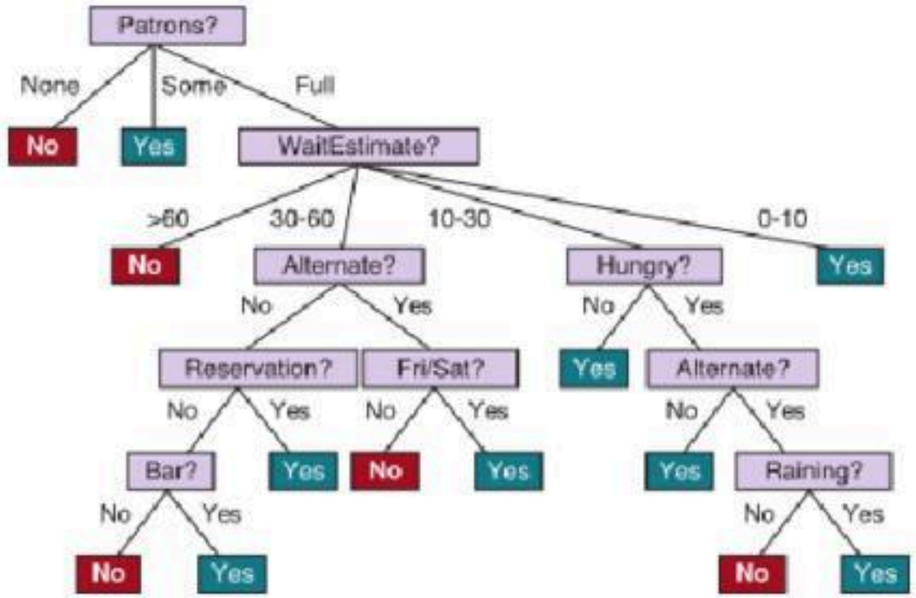
- 广义决策树模型可以处理输入、输出都是连续变量的情况



布尔分类

首先只考虑离散值输入和布尔型输出，即**布尔分类**

- 广义决策树模型可以处理输入、输出都是连续变量的情况



第 j 个样本： x_j, y_j

- $x_{j,i}$：第 j 个样本的第 i 个特征

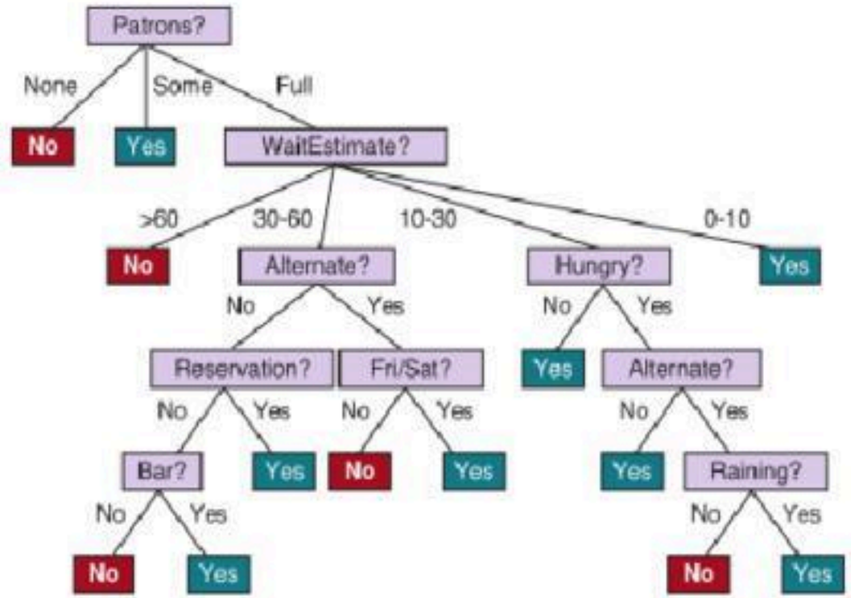
布尔型决策树

布尔型决策树等价以下逻辑声明：

$$Output \Leftrightarrow (Path_1 \vee Path_2 \vee \cdots)$$

- $Path_i \Leftrightarrow (A_a = v_a \wedge A_b = v_b \wedge \cdots)$
  - 即某条根节点到叶节点的**路径**
  - 途径分支由特征值检测决定

思考：此逻辑表达式有何特点？



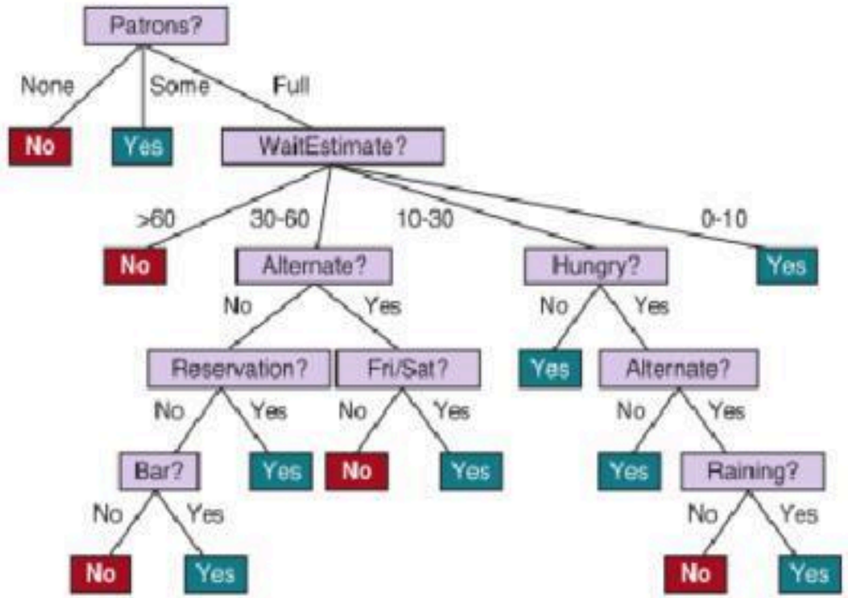
布尔型决策树

布尔型决策树等价以下逻辑声明：

$$Output \Leftrightarrow (Path_1 \vee Path_2 \vee \cdots)$$

- $Path_i \Leftrightarrow (A_a = v_a \wedge A_b = v_b \wedge \cdots)$
  - 即某条根节点到叶节点的**路径**
  - 途径分支由特征值检测决定

思考：此逻辑表达式有何特点？



回顾：此逻辑表达式是**析取范式**

- 推论：任何命题逻辑中的函数都可以表示为布尔型决策树

(*)决策树：可表示性

决策树的应用性非常广泛：可将解决问题的过程表达为简单、易懂的流程步骤

- 从编程逻辑上讲，决策树的本质就是**分支结构**
- 很多使用手册就是写成决策树的形式

(*)决策树：可表示性

决策树的应用性非常广泛：可将解决问题的过程表达为简单、易懂的流程步骤

- 从编程逻辑上讲，决策树的本质就是**分支结构**
- 很多使用手册就是写成决策树的形式

离散变量：组合问题，决策树有可能是**指数级规模**，例如：

- 大多数成立：只在多于半数的输入为真值时输出真
- 奇偶性函数：只在偶数个输入为真值时输出真

(*)决策树：可表示性

决策树的应用性非常广泛：可将解决问题的过程表达为简单、易懂的流程步骤

- 从编程逻辑上讲，决策树的本质就是**分支结构**
- 很多使用手册就是写成决策树的形式

离散变量：组合问题，决策树有可能是**指数级规模**，例如：

- 大多数成立：只在多于半数的输入为真值时输出真
- 奇偶性函数：只在偶数个输入为真值时输出真

连续变量：决策树只能将空间划分成**轴对齐的矩形区域**

- $y > A_1 + A_2$：决策边界是对角线

(*)决策树：似乎并不完美?

那么有没有某种模型，可以高效地表示所有函数类?

(*)决策树：似乎并不完美?

那么有没有某种模型，可以高效地表示所有函数类?

不可能（信息论）：有限位存储只能表达有限个函数

- （操作系统：）达到一定位数后，无法对存储空间进行有效管理

(*)决策树：似乎并不完美?

那么有没有某种模型，可以高效地表示所有函数类?

不可能（信息论）：有限位存储只能表达有限个函数

- （操作系统：）达到一定位数后，无法对存储空间进行有效管理

例如最简单的n位布尔函数：将n个布尔值属性映射到布尔值

- 真值表共2^n行，每行两种可能输出：总共2^{2^n}个不同函数
  - 即使只有20个属性：$2^{2^{20}} = 2^{1,048,576} \approx 10^{300,000}$
  - 存储消耗：百万bit

从样本学习决策树

回到餐馆等待问题

目标：找到能够解释以下样本的最简决策树

- “**最简**”含义：测试数少、最长路径短、决策树浅

Example	Input Attributes										Output
	Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est	*WillWait*
x_1	*Yes*	*No*	*No*	*Yes*	*Some*	*$$$*	*No*	*Yes*	*French*	*0–10*	$y_1 = Yes$
x_2	*Yes*	*No*	*No*	*Yes*	*Full*	*$*	*No*	*No*	*Thai*	*30–60*	$y_2 = No$
x_3	*No*	*Yes*	*No*	*No*	*Some*	*$*	*No*	*No*	*Burger*	*0–10*	$y_3 = Yes$
x_4	*Yes*	*No*	*Yes*	*Yes*	*Full*	*$*	*Yes*	*No*	*Thai*	*10–30*	$y_4 = Yes$
x_5	*Yes*	*No*	*Yes*	*No*	*Full*	*$$$*	*No*	*Yes*	*French*	*>60*	$y_5 = No$
x_6	*No*	*Yes*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Italian*	*0–10*	$y_6 = Yes$
x_7	*No*	*Yes*	*No*	*No*	*None*	*$*	*Yes*	*No*	*Burger*	*0–10*	$y_7 = No$
x_8	*No*	*No*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Thai*	*0–10*	$y_8 = Yes$
x_9	*No*	*Yes*	*Yes*	*No*	*Full*	*$*	*Yes*	*No*	*Burger*	*>60*	$y_9 = No$
x_{10}	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$$$*	*No*	*Yes*	*Italian*	*10–30*	$y_{10} = No$
x_{11}	*No*	*No*	*No*	*No*	*None*	*$*	*No*	*No*	*Thai*	*0–10*	$y_{11} = No$
x_{12}	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$*	*No*	*No*	*Burger*	*30–60*	$y_{12} = Yes$

回到餐馆等待问题

目标：找到能够解释以下样本的最简决策树

- “**最简**”含义：测试数少、最长路径短、决策树浅

Example	Input Attributes										Output
	Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est	*WillWait*
x_1	*Yes*	*No*	*No*	*Yes*	*Some*	*$$$*	*No*	*Yes*	*French*	*0–10*	$y_1 = Yes$
x_2	*Yes*	*No*	*No*	*Yes*	*Full*	*$*	*No*	*No*	*Thai*	*30–60*	$y_2 = No$
x_3	*No*	*Yes*	*No*	*No*	*Some*	*$*	*No*	*No*	*Burger*	*0–10*	$y_3 = Yes$
x_4	*Yes*	*No*	*Yes*	*Yes*	*Full*	*$*	*Yes*	*No*	*Thai*	*10–30*	$y_4 = Yes$
x_5	*Yes*	*No*	*Yes*	*No*	*Full*	*$$$*	*No*	*Yes*	*French*	*>60*	$y_5 = No$
x_6	*No*	*Yes*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Italian*	*0–10*	$y_6 = Yes$
x_7	*No*	*Yes*	*No*	*No*	*None*	*$*	*Yes*	*No*	*Burger*	*0–10*	$y_7 = No$
x_8	*No*	*No*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Thai*	*0–10*	$y_8 = Yes$
x_9	*No*	*Yes*	*Yes*	*No*	*Full*	*$*	*Yes*	*No*	*Burger*	*>60*	$y_9 = No$
x_{10}	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$$$*	*No*	*Yes*	*Italian*	*10–30*	$y_{10} = No$
x_{11}	*No*	*No*	*No*	*No*	*None*	*$*	*No*	*No*	*Thai*	*0–10*	$y_{11} = No$
x_{12}	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$*	*No*	*No*	*Burger*	*30–60*	$y_{12} = Yes$

- 回顾：组合优化问题是NP难，但**启发式搜索**可以高效找到一个合理逼近

算法：决策树

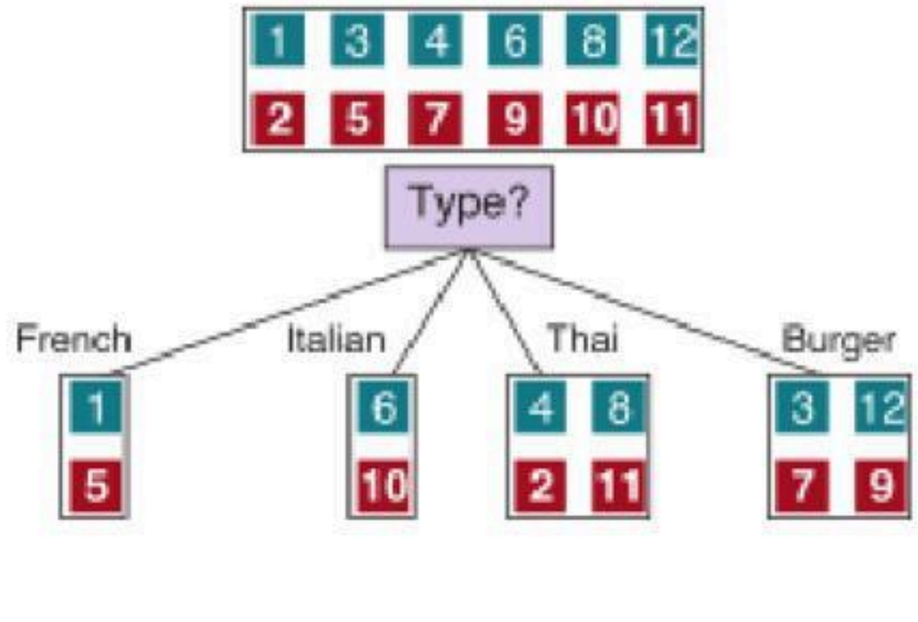
`DECISION-TREE-LEARNING` 基于**贪心、分治**策略

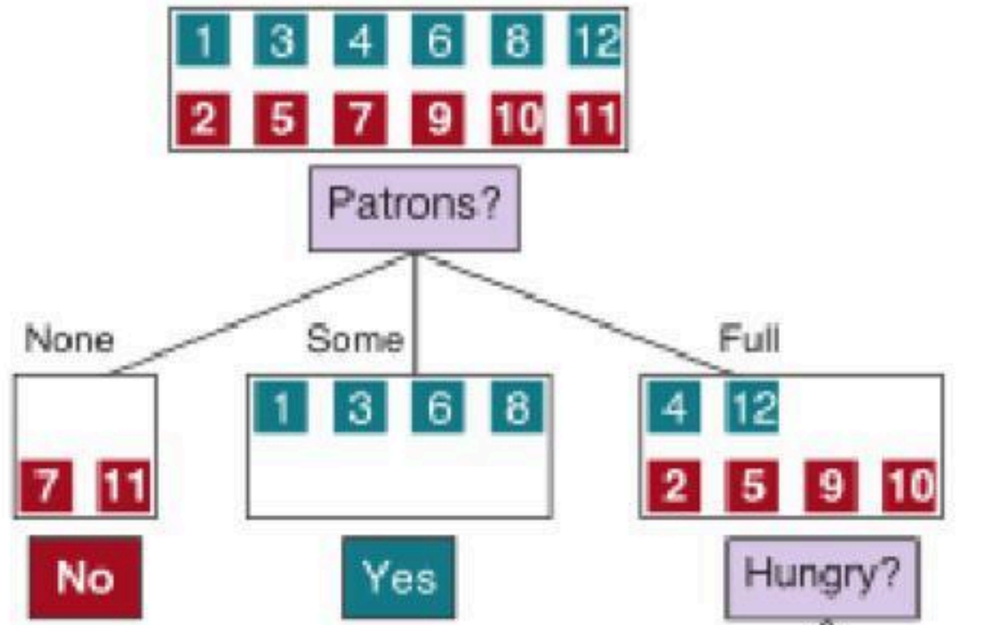
- 每步检测都选择最重要的一个特征，递归解决更小的子问题
  - 思考：如何定义“最重要”？

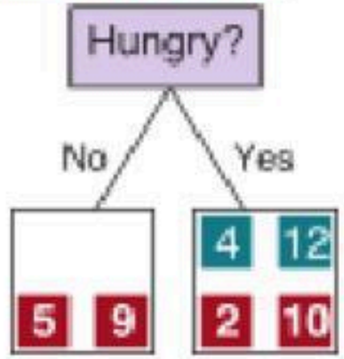
function DECISION-TREE-LEARNING(*examples*, *attributes*, *parent_examples*) **returns** a tree
 if *examples* 是空集 **then return** PLURALITY-VALUE(*parent_examples*)
 else if *examples* 分类结果都相同 **then return** 分类结果
 else if *attributes* 是空集 **then return** PLURALITY-VALUE(*examples*)
 else
  $A \leftarrow \text{argmax}_{a \in attributes}$ IMPORTANCE(*a*, *examples*)
  tree ← 以特征 *A* 为根检测节点的决策树
  for each 特征 *A* 的取值 v_k **do**
   exs ← { *e* : *e* ∈ *examples* **and** *e*._A_ = v_k }

- `PLURALITY-VALUE`：多数胜出，输出判定

特征的重要性：直观理解





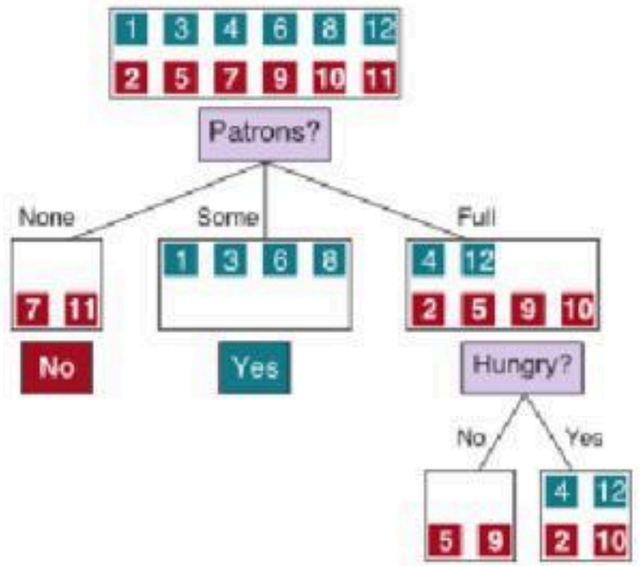


- $Type$ 的效用低：产生的4个分支中，每个分支都不能确定多数
- $Patrons$ 相对比较重要：产生两个决策输出分支，另一分支有简单多数

递归逻辑

算法分支结构注解：每个子问题有4种情况

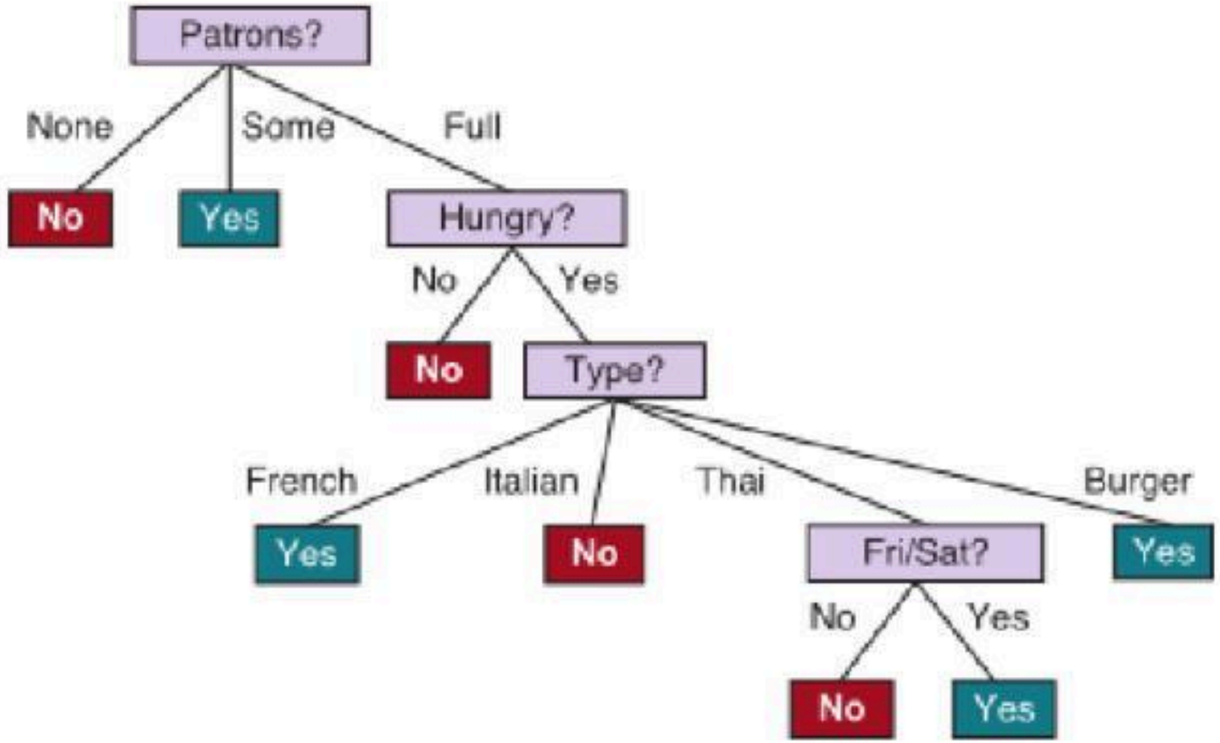
1. 样本耗尽：这一组属性值没有观测样本
   - 无法继续运行：返回父节点的多数投票
2. 所有样本判定一致：输出判定
3. 特征耗尽：剩余样本的描述相同但不同类
   - 无法继续检测：返回多数投票
   - 可能：数据有误、有噪音；问题本身非确定；特征建模不足
4. 样本判定不一致：在剩余属性中选择最优并继续递归



function DECISION-TREE-LEARNING(*examples*, *attributes*, *parent_examples*) **returns** a tree
 if *examples* 是空集 **then return** PLURALITY-VALUE(*parent_examples*)
 else if *examples* 分类结果都相同 **then return** 分类结果
 else if *attributes* 是空集 **then return** PLURALITY-VALUE(*examples*)
 else
  $A \leftarrow \text{argmax}_{a \in attributes}$ IMPORTANCE(*a*, *examples*)
  tree ← 以特征 *A* 为根检测节点的决策树

for each 特征 A 的取值 v_k **do**

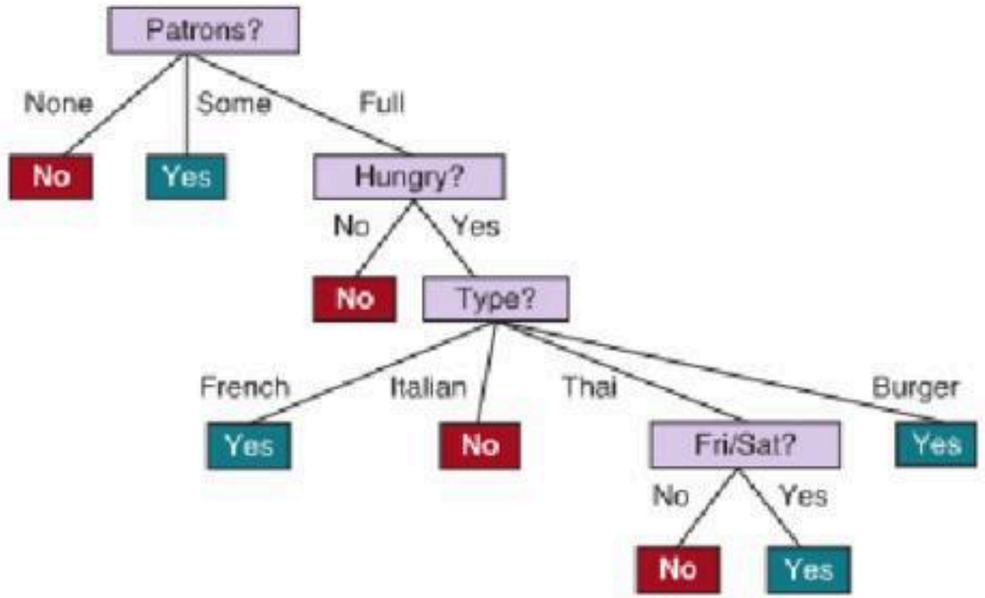
决策树算法输出：餐馆等待问题



注意：算法**只依赖样本本身**，而不考虑与真实函数之间的误差

- 真实函数是未知的，也就无法用于计算：决策树学习算法巧妙地解决了这个问题
- 输出结果不仅对所有样本预测正确，而且比不用任何算法的初始版本简单得多

(*)正确性评价



对此问题来说，只有**当事人**才能做出正确评判

- $Raining$ 和 $Reservation$ 没有被考虑：当前模型已经可以正确解释所有样本
- 决策树可以用于发掘当事人从未意识到的行为习惯
- 样本数不足时，结论可能会出错
  - 当前业界实践的普遍问题：有效样本数量不足

思考：如何客观地评价模型正确性？（未来章节）

(*)学习曲线

抛开实际应用不谈，评测模型的性能可以使用**人造数据**

- 有助于增进对模型的理解，或比较模型之间的差异

(*)学习曲线

抛开实际应用不谈，评测模型的性能可以使用**人造数据**

- 有助于增进对模型的理解，或比较模型之间的差异

为此，随机生成100个样本，并划分到两个集合

- 训练集用于学习模型；测试集用于度量正确性（思考：为什么？）

(*)学习曲线

抛开实际应用不谈，评测模型的性能可以使用**人造数据**

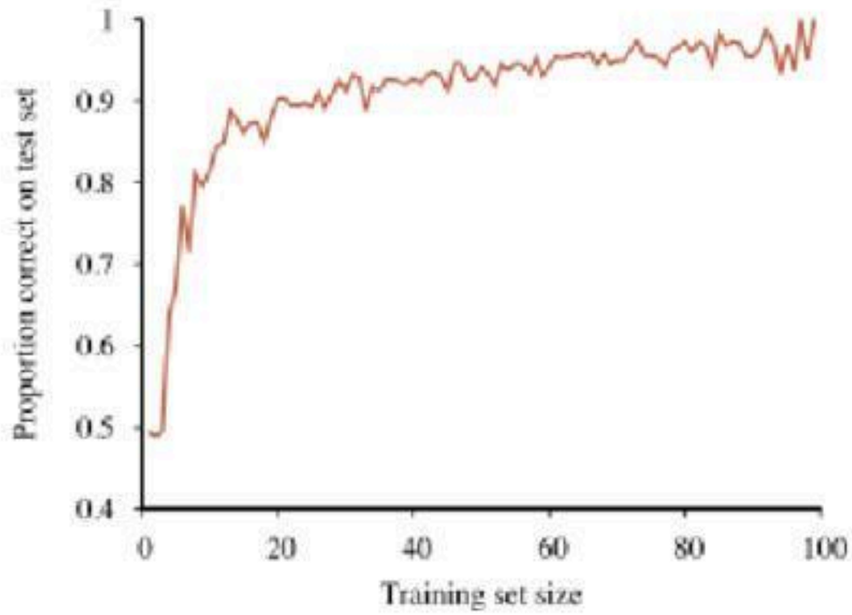
- 有助于增进对模型的理解，或比较模型之间的差异

为此，随机生成100个样本，并划分到两个集合

- 训练集用于学习模型；测试集用于度量正确性（思考：为什么？）

学习曲线：测试集正确率相对训练集大小的曲线

- 关键结论：正确率随着训练集的增大而提升
- 为了平滑曲线：重复随机划分20次，取平均值



ID3算法

ID3算法基本思路

ID3算法是一个经典的决策树学习算法，由Quinlan于1979年提出。

ID3算法的**基本思想**是，以信息熵为度量，用于决策树节点的属性选择

- 通过一个准则找到一个最合适的特征，以这个特征作为根结点；
- 根结点对应了样本集的所有样本，确定一个特征作为根结点，由该特征的各取值形成相应的几条边，引出下一层的结点；
- 每一条边对应了根结点特征的一个取值，根据该特征的不同取值把样本集分成几组，每一组形成一子样本集；
- 子样本集的特征向量中，将根结点已用到的特征删去，每一个子样本集对应一条边下端的结点；
- 满足条件的节点定为叶节点，并确定对应输出，否则递归执行。

信息熵、条件熵

设样本集为$\mathbf{D}$，需分类的类型有K类，设标注是k类的概率为p_k，表示样本集类型的**信息熵（经验估计）**为：

$$H(\mathbf{D}) = -\sum_{k=1}^{K} \frac{c_k}{N} \log \frac{c_k}{N}$$

信息熵、条件熵

设样本集为$\mathbf{D}$，需分类的类型有K类，设标注是k类的概率为p_k，表示样本集类型的**信息熵（经验估计）**为：

$$H(\mathbf{D}) = -\sum_{k=1}^{K} \frac{c_k}{N} \log \frac{c_k}{N}$$

考虑了特征A后需要考虑取值$A = i$的概率，并用**条件熵**度量：

$$H(\mathbf{D}|A) = -\sum_{i\in[1,I]} \frac{N_i}{N} \sum_{k=1}^{K} \frac{c_{ik}}{N} \log \frac{c_{ik}}{N}$$

信息熵、条件熵

设样本集为$\mathbf{D}$，需分类的类型有K类，设标注是k类的概率为p_k，表示样本集类型的**信息熵（经验估计）**为：

$$H(\mathbf{D}) = -\sum_{k=1}^{K} \frac{c_k}{N} \log \frac{c_k}{N}$$

考虑了特征A后需要考虑取值$A = i$的概率，并用**条件熵**度量：

$$H(\mathbf{D}|A) = -\sum_{i\in[1,I]} \frac{N_i}{N} \sum_{k=1}^{K} \frac{c_{ik}}{N} \log \frac{c_{ik}}{N}$$

ID3算法采用**信息增益**做为特征重要性的度量；定义为前两者的差值：

$$G(\mathbf{D}, A) = H(\mathbf{D}) - H(\mathbf{D}|A)$$

例：看电影问题

序号	女友A_1	作业A_2	预习A_3	兴趣A_4	决定
1	女友去	完成	需要	喜欢	看
2	女友去	未完成	需要	不喜欢	不看
3	女友去	未完成	不需要	不喜欢	看
4	女友去	完成	需要	不喜欢	看
5	女友不去	完成	不需要	喜欢	看
6	女友不去	未完成	不需要	喜欢	不看
7	女友不去	完成	需要	喜欢	看
8	女友不去	完成	不需要	不喜欢	不看
9	女友不去	未完成	需要	不喜欢	不看
10	无女友	完成	不需要	喜欢	看
11	无女友	未完成	不需要	喜欢	不看

看电影问题：信息增益1

序号	女友 A_1	作业 A_2	预习 A_3	兴趣 A_4	决定
1	女友去	完成	需要	喜欢	看
2	女友去	未完成	需要	不喜欢	不看
3	女友去	未完成	不需要	不喜欢	看
4	女友去	完成	需要	不喜欢	看
5	女友不去	完成	不需要	喜欢	看
6	女友	未完	不需	喜欢	不

$$
\begin{aligned}
H(\mathbf{D}) &= -\sum_{k=1}^{K} \frac{c_k}{N} \log \frac{c_k}{N} \\
&= -\frac{7}{15} \log \frac{7}{15} - \frac{8}{15} \log \frac{8}{15} \\
&= 0.9966
\end{aligned}
$$

$$
\begin{aligned}
G(\mathbf{D}, A_1) &= H(\mathbf{D}) - H(\mathbf{D}|A_1) \\
&= H(\mathbf{D}) - \sum_{i \in [1,I]} \frac{N_i}{N} \sum_{k=1}^{K} \frac{c_{ik}}{N} \log \frac{c_{ik}}{N} \\
&= 0.9966 - \frac{4}{15}(-\frac{3}{4} \log \frac{3}{4} - \frac{1}{4} \log \\
&\quad - \frac{5}{15}(-\frac{2}{5} \log \frac{2}{5} - \frac{3}{5} \log \frac{3}{5}) \\
&\quad - \frac{6}{15}(-\frac{2}{6} \log \frac{2}{6} - \frac{4}{6} \log \frac{4}{6}) \\
&= 0.0866
\end{aligned}
$$

看电影问题：分支1

同理计算其他特征的信息增益如下

- $G(\mathbf{D}, A_1) = 0.0866$
- $G(\mathbf{D}, A_2) = 0.2876$
- $G(\mathbf{D}, A_3) = 0.027$
- $G(\mathbf{D}, A_4) = 0.0366$

看电影问题：分支1

同理计算其他特征的信息增益如下

- $G(\mathbf{D}, A_1) = 0.0866$
- $G(\mathbf{D}, A_2) = 0.2876$
- $G(\mathbf{D}, A_3) = 0.027$
- $G(\mathbf{D}, A_4) = 0.0366$

显然，对这位同学来说“作业”最重要，因此作为根节点来构建决策树

看电影问题：数据划分1

序号	女友 A_1	作业 A_2	预习 A_3	兴趣 A_4	决定
1	女友去	完成	需要	喜欢	看
4	女友去	完成	需要	不喜欢	看
5	女友不去	完成	不需要	喜欢	看
7	女友不去	完成	需要	喜欢	看
8	女友不去	完成	不需要	不喜欢	不看
10	无女	完成	不需	喜欢	看

序号	女友 A_1	作业 A_2	预习 A_3	兴趣 A_4	决定
2	女友去	未完成	需要	不喜欢	不看
3	女友去	未完成	不需要	不喜欢	看
6	女友不去	未完成	不需要	喜欢	不看
9	女友不去	未完成	需要	不喜欢	不看
11	无女友	未完成	不需要	喜欢	不看
12	无女	未完	需要	喜欢	不

看电影问题：分支2

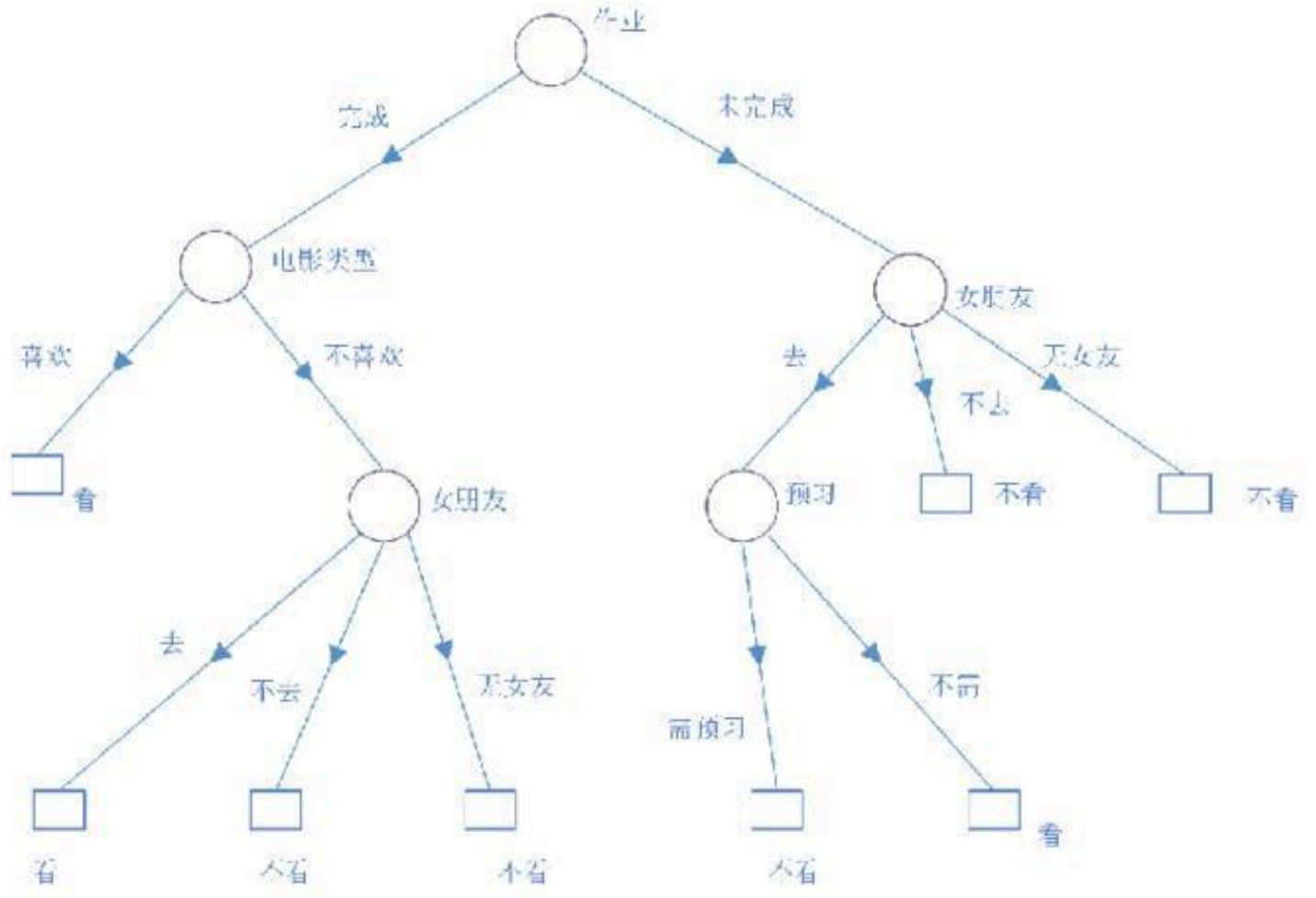
以$\mathbf{D}_2$为例继续计算信息增益

- $G(\mathbf{D_2}, A_1) = 0.31$
- $G(\mathbf{D_2}, A_3) = 0.128$
- $G(\mathbf{D_2}, A_4) = 0.2$

注意：此时特征A_2不需要再考虑

由计算结果得知，“女友”应被选为第二个检测特征

看电影问题：最终输出



问题：给定一个新的输入：女友去、未完成、不需要、喜欢，预测去不去?

(*)度量特征重要性

决策树学习算法的关键：重要性度量

回顾：决策树选取特征的准则是最高“重要性”

决策树学习算法的关键：重要性度量

回顾：决策树选取特征的准则是最高“重要性”

- （信息）理论上可以从**信息增益**的角度度量重要性
- 借助于信息论里的基本量：**熵**

决策树学习算法的关键：重要性度量

回顾：决策树选取特征的准则是最高“重要性”

- （信息）理论上可以从**信息增益**的角度度量重要性
- 借助于信息论里的基本量：**熵**

缺乏信息时（信息不对称），难以做出正确决策：需要**通过操作换取信息**

- 决策树的每一步检测都是有成本的，例如军旗试探，迷雾开图
- 所有信息公开透明时，穷举也能搜到正确决策
  - 为了**付出最少成本**：每一步操作都必须最有价值，即换取最多信息

熵：随机变量不确定性的度量

信息越多，熵值越少：两面一样的硬币没有不确定性，熵值为0

熵：随机变量不确定性的度量

信息越多，熵值越少：两面一样的硬币没有不确定性，熵值为0

- 公平硬币：正反面的概率相同，具有1**比特**的熵
- 四面骰子：2^2 种等概率结果，具有2比特的熵

熵：随机变量不确定性的度量

信息越多，熵值越少：两面一样的硬币没有不确定性，熵值为0

- 公平硬币：正反面的概率相同，具有1**比特**的熵
- 四面骰子：2^2 种等概率结果，具有2比特的熵

给定随机变量 V，取值 v_k 时的概率为 $P(v_k)$，则 V 的**熵**

$$H(V) = \sum_k P(v_k) \log_2 \frac{1}{P(v_k)} = -\sum_k P(v_k) \log_2 P(v_k)$$

熵：计算举例

公平硬币：$H(Fair) = (0.5 \log_2 0.5 + 0.5 \log_2 0.5) = 1$

熵：计算举例

公平硬币：$H(Fair) = (0.5 \log_2 0.5 + 0.5 \log_2 0.5) = 1$

四面骰子：$H(Die4) = -(4 \times 0.25 \log_2 0.25) = 2$

熵：计算举例

公平硬币：$H(Fair) = (0.5 \log_2 0.5 + 0.5 \log_2 0.5) = 1$

四面骰子：$H(Die4) = -(4 \times 0.25 \log_2 0.25) = 2$

假设有一灌铅硬币，99% 的概率出正面

- 理论上这个硬币的不确定性只比公平硬币小一点点，熵值应该非常接近0
- 验证：$H(Loaded) = (0.99 \log_2 0.99 + 0.01 \log_2 0.01) \approx 0.08$

布尔随机变量

为了讨论方便及实用性，定义真值概率为 q 的布尔随机变量的熵

$$B(q) = -q \log_2 q - (1-q) \log_2 (1-q)$$

- $H(Loaded) = B(0.99)$

布尔随机变量

为了讨论方便及实用性，定义真值概率为 q 的布尔随机变量的熵

$$B(q) = -q \log_2 q - (1-q) \log_2 (1-q)$$

- $H(Loaded) = B(0.99)$

假设训练集有 p 个正例和 n 个负例，则整个集合的输出变量的熵

$$H(Output) = B(\frac{p}{p+n})$$

- 餐馆等待数据集：$p = n = 6$，故熵为 $B(0.5) = 1$

剩余总体熵值

选取特征 A 进行检测的**原则**是：系统获得最多的信息，故减少对应量总体熵值

- **选取思路**：检测分支后的剩余总体熵值越低，操作越有意义

剩余总体熵值

选取特征 A 进行检测的**原则**是：系统获得最多的信息，故减少对应量总体熵值

- **选取思路**：检测分支后的剩余总体熵值越低，操作越有意义

假设特征 A 有 d 个不同值：将训练集分割为子集 $E_1, .., E_d$

- 假设E_k 包含 p_k 个正例和 p_n 个负例
  - 如果选取这一分支，还需 $B(p_k/(p_k+n_k))$ 信息才能做出正确决策

剩余总体熵值

选取特征 A 进行检测的**原则**是：系统获得最多的信息，故减少对应量总体熵值

- **选取思路**：检测分支后的剩余总体熵值越低，操作越有意义

假设特征 A 有 d 个不同值：将训练集分割为子集 $E_1, .., E_d$

- 假设E_k 包含 p_k 个正例和 p_n 个负例
  - 如果选取这一分支，还需 $B(p_k/(p_k+n_k))$ 信息才能做出正确决策
- 从训练集中随机选取一个样本，恰巧落在 E_k 中的概率是 $(p_k+n_k)/(p+n)$

剩余总体熵值

选取特征 A 进行检测的**原则**是：系统获得最多的信息，故减少对应量总体熵值

- **选取思路**：检测分支后的剩余总体熵值越低，操作越有意义

假设特征 A 有 d 个不同值：将训练集分割为子集 $E_1, .., E_d$

- 假设E_k 包含 p_k 个正例和 p_n 个负例
  - 如果选取这一分支，还需 $B(p_k/(p_k+n_k))$ 信息才能做出正确决策
- 从训练集中随机选取一个样本，恰巧落在 E_k 中的概率是 $(p_k+n_k)/(p+n)$

故：检测分支后的**剩余总体熵值的期望值**

$$Remainder(A) = \sum_{k=1}^{d} \frac{p_k+n_k}{p+n} B(\frac{p_k}{p_k+n_k})$$

信息增益

检测分支后的**信息增益**：熵值减少的期望值

$$Gain(A) = B(\frac{p}{p+n}) - Remainder(A) = B(\frac{p}{p+n}) - \sum_{k=1}^{d} \frac{p_k + n_k}{p+n} B(\frac{p_k}{p_k + n_k})$$

- 此即决策树学习算法中重要性函数的常见构造方法（**ID3算法**）

信息增益

检测分支后的**信息增益**：熵值减少的期望值

$$Gain(A) = B(\frac{p}{p+n}) - Remainder(A) = B(\frac{p}{p+n}) - \sum_{k=1}^{d} \frac{p_k + n_k}{p+n} B(\frac{p_k}{p_k + n_k})$$

- 此即决策树学习算法中重要性函数的常见构造方法（**ID3算法**）

验证：$Patrons$ 是第一个检测分支的最佳特征选择

- $Gain(Patrons) = 1 - [\frac{2}{12}B(\frac{0}{2}) + \frac{4}{12}B(\frac{4}{4}) + \frac{6}{12}B(\frac{2}{6})] \approx 0.541$
- $Gain(Type) = 1 - [\frac{2}{12}B(\frac{1}{2}) + \frac{2}{12}B(\frac{1}{2}) + \frac{4}{12}B(\frac{2}{4}) + \frac{4}{12}B(\frac{2}{4})] = 0$

决策树的三种基本算法

ID3 (Iterative Dichotomiser)、C4.5、CART (Classification And Regression Tree)

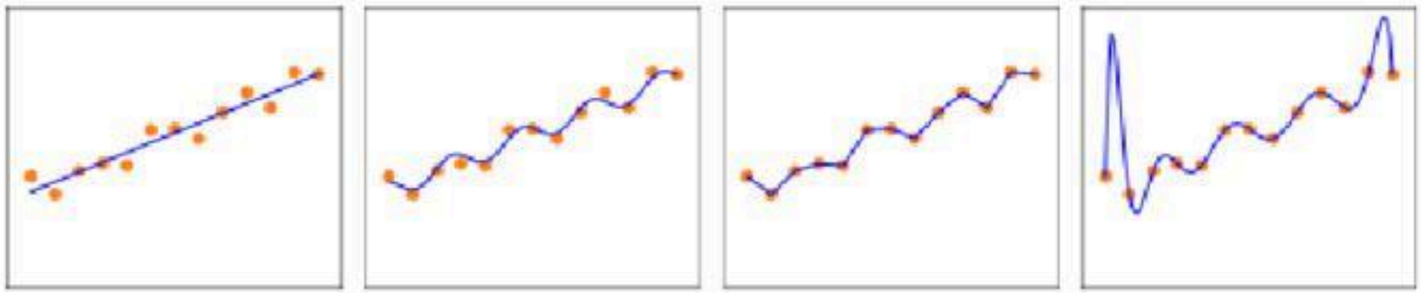
算法	支持模型	树结构	特征选择	连续值	缺失值	剪枝	特征重复使用
ID3	分类	多叉树	信息增益	-	-	-	-
C4.5	分类	多叉树	信息增益率	支持	支持	支持	-
CART	分类/回归	二叉树	基尼指数/均方差	支持	支持	支持	支持

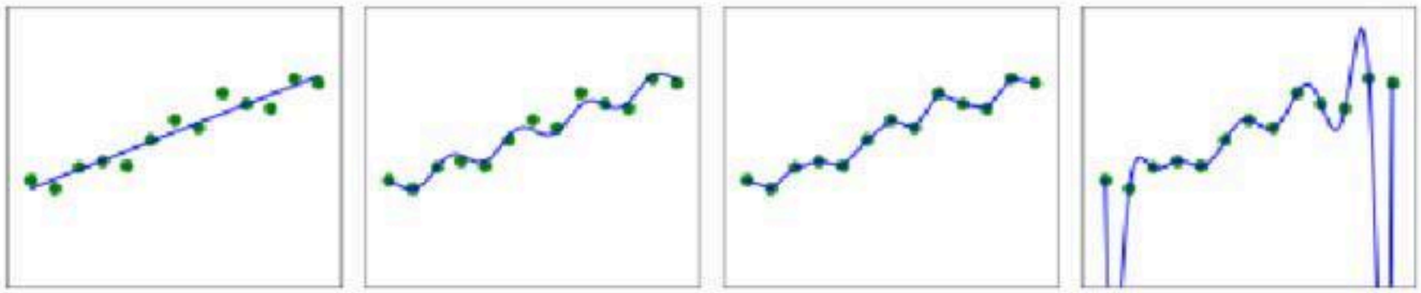
(*)泛化与过拟合

回顾：拟合、过拟合

实现学习算法的**基本要求**：正确性，即拟合训练集

- **优秀**学习算法：对未知数据的泛化性强

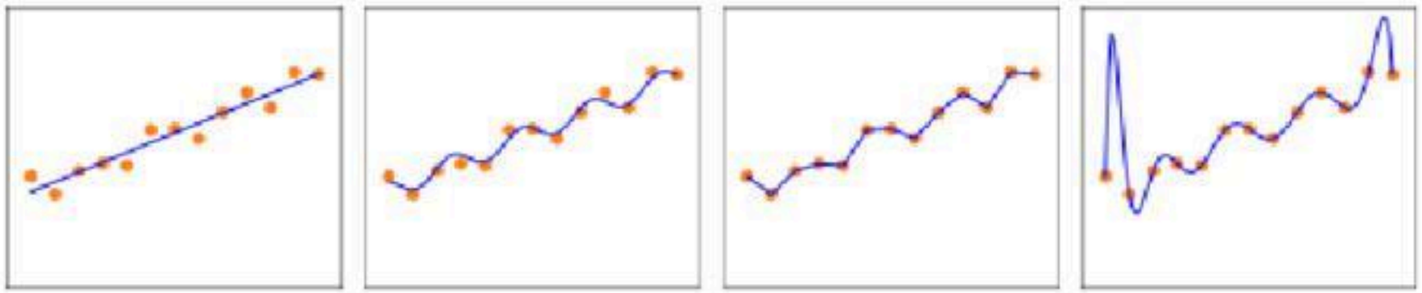


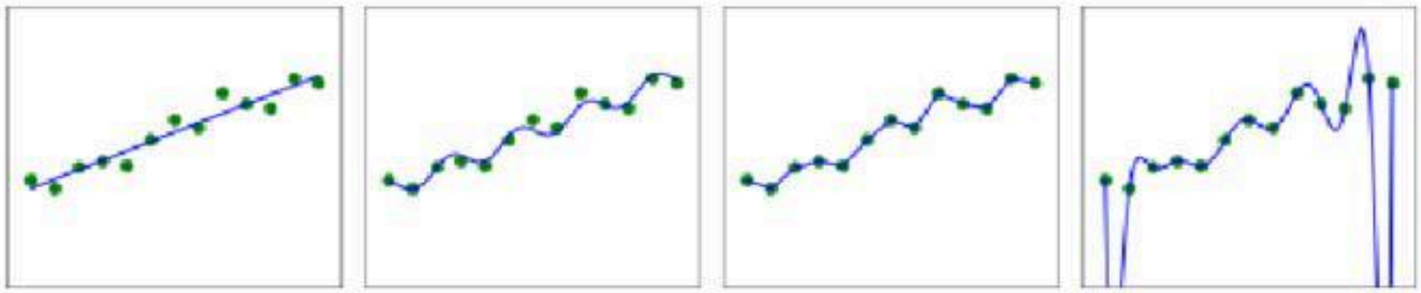


回顾：拟合、过拟合

实现学习算法的**基本要求**：正确性，即拟合训练集

- **优秀**学习算法：对未知数据的泛化性强





过拟合：模型过于关注训练集，即使是**错误、异常数据**

- 随特征数增加而增强；随样本数增加而减弱
- 更大的假设空间更容易拟合和过拟合；某些模型类更容易过拟合

决策树剪枝

决策树建立检测通路，在叶节点输出判定

- 过拟合：关注通往每个数据的检测路径
  - 错误、异常：需要大量检测才能排查，无效路径过长

决策树剪枝

决策树建立检测通路，在叶节点输出判定

- 过拟合：关注通往每个数据的检测路径
  - 错误、异常：需要大量检测才能排查，无效路径过长

决策树剪枝：消除显然低效的检测节点

1. 从算法生成的完整决策树起始
2. 提取一个分支都是叶节点的检测节点
    1. 如果此检测低效（只检测噪音）：
        1. 消除此节点，并替换成最合理的子节点
3. 循环上一步

思考：什么样的特征检测是低效的?

(*)特征检测：低效

假设检测前有 p 个正例和 n 个负例

- 低效意即此检测不能辅助决策树做出更确定的判断
  - 最终判定基于简单多数

(*)特征检测：低效

假设检测前有 p 个正例和 n 个负例

- 低效意即此检测不能辅助决策树做出更确定的判断
  - 最终判定基于简单多数

低信息增益判定法：每个分支的正负例比例与之前相近

- 信息增益接近0

(*)特征检测：低效

假设检测前有 p 个正例和 n 个负例

- 低效意即此检测不能辅助决策树做出更确定的判断
  - 最终判定基于简单多数

低信息增益判定法：每个分支的正负例比例与之前相近

- 信息增益接近0

思考：多大的信息增益足以建立节点?

- 提示：显著性检测

决策树：总结

实际应用中，决策树模型经常是分类问题的**首选解决方案**

- 易于理解，适应大数据，扩展性强
- 贪心搜索容易导致正确率相对不高
- 如果树很深，运行时效率低

决策树：总结

实际应用中，决策树模型经常是分类问题的**首选解决方案**

- 易于理解，适应大数据，扩展性强
- 贪心搜索容易导致正确率相对不高
- 如果树很深，运行时效率低

决策树**不具有稳定性**：仅增加一个新样本就有可能导致整个树的结构改变

- 例如：根节点的检测条件发生改变

Review

本章内容

1. 决策树的表达力
2. 从样本学习决策树
3. 度量特征重要性
4. 泛化与过拟合
5. 实验：决策树算法

重点：模型表达力；决策树构建算法。

难点：度量信息；判定过拟合。

学习目标

1. 理解布尔型决策树等价于析取范式
2. 掌握基于贪心、分治策略学习决策树的方法
3. 理解信息熵的概念；掌握基于信息增益的ID3算法
4. 理解泛化和过拟合的概念
5. 掌握决策树算法的实现方法

问题

1. 简述为什么决策树具有强大的表达力。
2. 简述决策树构建算法及基于信息增益的ID3算法。
3. (*)简述信息熵的定义及意义，并计算抛公平硬币的熵值。