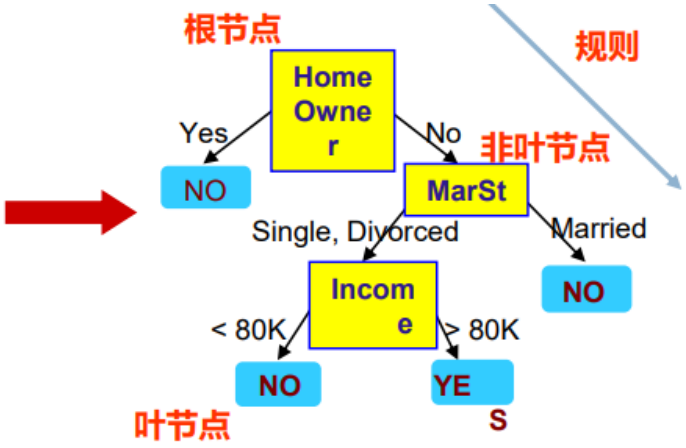
Decision Trees 决策树

什么是决策树 —— 基本概念

- 非叶节点：一个属性上的测试，每个分枝代表该测试的输出
- 叶节点：存放一个类标记
- 规则：从根节点到叶节点的一条属性取值路径

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



建立决策树分类模型的流程

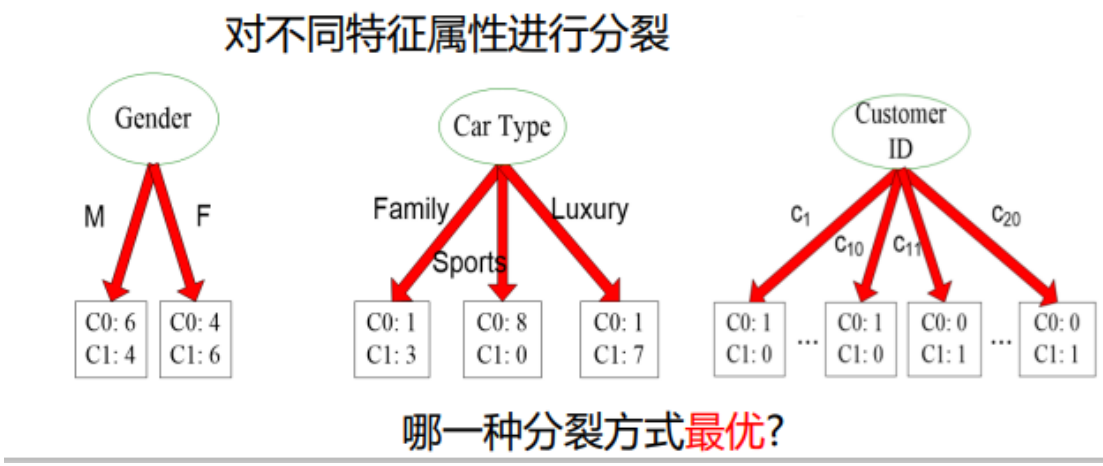
- 模型训练：从已有数据中生成一棵决策树
- 分裂数据的特征，寻找决策类别的路径
- **相同的数据，根据不同的特征顺序，可以建立多种决策树**

如何建立决策树?

基本的决策树学习过程，可以归纳为以下三个步骤：

1. 特征选择：选取对于训练数据有着较强区分能力的特征
2. 生成决策树：基于选定的特征，逐步生成完整的决策树

3. 决策树剪枝：简化部分枝干，避免过拟合因素影响



决策树学习

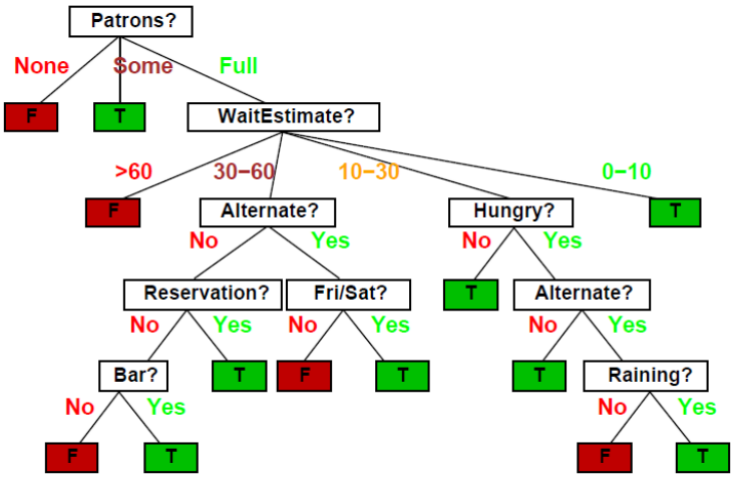
问题：基于以下属性，决定是否在餐厅等待桌子：

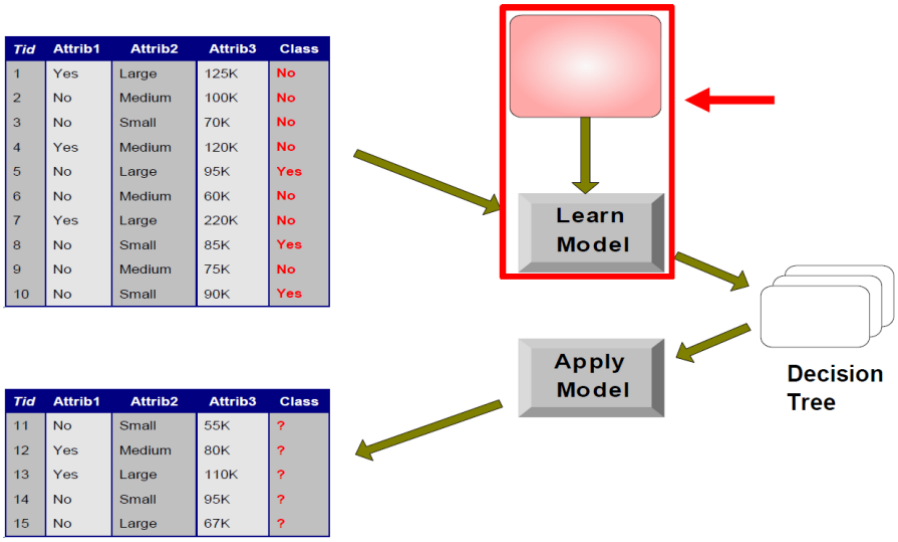
1. Alternate：附近是否有其他选择的餐厅?
2. Bar：是否有一个舒适的酒吧区等待?
3. Fri/Sat：今天是星期五还是星期六?
4. Hungry：我们饿了吗?
5. Patrons：餐厅里的人数（无人、有些人、满座）
6. Price：价格范围（`$、$$、$$$`）
7. Raining：外面是否下雨?
8. Reservation：我们是否预约了?
9. Type：餐厅类型（法国、意大利、泰国、汉堡）
10. WaitEstimate：等待时间的预估值（0-10、10-30、30-60、>60）

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T

假设的一种可能表示

例如，在上述餐厅等待桌子的问题中，我们可以使用决策树来表示假设，该决策树定义了在不同属性值下等待桌子的决策。以下是一个可能的假设树示例：



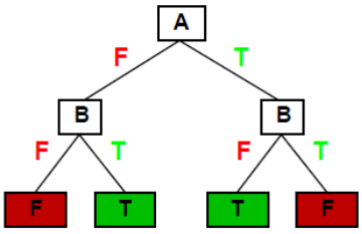


表达能力

决策树可以表示任何输入属性的函数，但使用单条路径来表示每个训练示例的决策树可能会过度拟合数据，无法很好地推广到新的未见过的数据示例。

决策树可以表达输入属性的任何函数。例如，对于布尔函数，函数真值表的每行对应于树中 的一条路径:

A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F



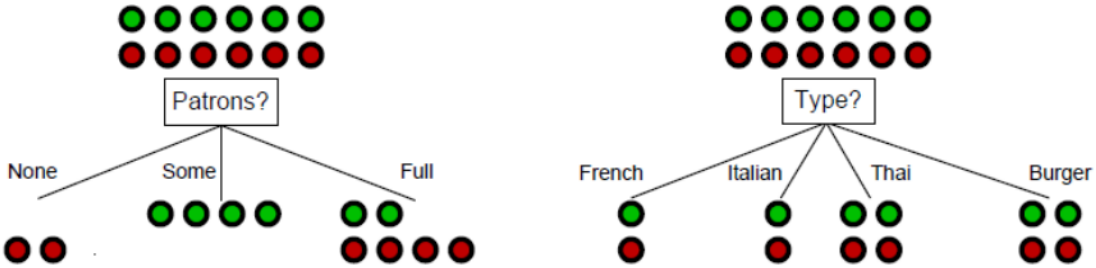
简单来说，针对每个训练示例，可以创建一条路径到叶子节点的一致性决策树（除非函数在输入属性上是非确定性的），但这种决策树可能会过度拟合数据，无法很好地泛化到新的未见过的数据示例。因此，更倾向于找到更紧凑的决策树来提高泛化性能。

决策树学习

目的：找到一个与训练示例一致的小树
想法：（递归）选择"最重要"属性作为（子）树的根

```
function DTL(examples, attributes, default) returns a decision tree
    if examples is empty then return default
    else if all examples have the same classification then return the classification
    else if attributes is empty then return MODE(examples)
    else
        best ← CHOOSE-ATTRIBUTE(attributes, examples)
        tree ← a new decision tree with root test best
        for each value v_i of best do
            examples_i ← {elements of examples with best = v_i}
            subtree ← DTL(examples_i, attributes − best, MODE(examples))
            add a branch to tree with label v_i and subtree subtree
        return tree
```

想法：一个好的属性将示例拆分为（理想情况下）"全正"或"全负"的子集



根据Patron分类是一个更好的选择

信息论在决策树学习中的应用

信息熵：计算数据的不确定性

$$\text{Entropy}(t) = -\sum_{j=1}^{m} p(j|t) \log_2 p(j|t)$$

此时：表示某个节点t （即某个特征）的信息不确定性
$p(j|t)$是节点特征t的属于类别j的样本的比例

- 特点：对于该节点特征t
  - 当样本均匀地分布在各个类别时，熵达到最大值$log(n_c)$, 此时包含的信息最少
  - 当样本只属于一个类别时，熵达到最小值 0, 此时包含的信息最多

对于包含 p 个正例和 n 个反例的训练集，其熵可以用以下公式计算：

$$I(\frac{p}{p+n}, \frac{n}{p+n}) = -\frac{p}{p+n}\log_2 \frac{p}{p+n} - \frac{n}{p+n}\log_2 \frac{n}{p+n}$$

其中，第一项和第二项分别表示正例和反例的占比，$\log_2$ 表示以 2 为底的对数。熵的值越高，表示数据集越不确定。

C1	**0**
C2	**6**

P(C1) = 0/6 = 0　　P(C2) = 6/6 = 1
Entropy = – 0 log 0 – 1 log 1 = – 0 – 0 = 0

C1	**1**
C2	**5**

P(C1) = 1/6　　P(C2) = 5/6
Entropy = – (1/6) $\log_2$ (1/6) – (5/6) $\log_2$ (1/6) = 0.65

C1	**2**
C2	**4**

P(C1) = 2/6　　P(C2) = 4/6
Entropy = – (2/6) $\log_2$ (2/6) – (4/6) $\log_2$ (4/6) = 0.92

特征选择准则一：信息增益

信息增益: 按某个特征划分之后，数据不确定性降低的程度

$$\text{Gain}(m) = \text{Entropy}(p) - (\sum_{i=1}^{k} \frac{|n_i|}{n} \text{Entropy}(i))$$

1. 第一项$\text{Entropy}(p)$表示数据未划分时的信息熵
2. 第二项$\sum_{i=1}^{k} \frac{|n_i|}{n} \text{Entropy}(i)$表示按特征m划分后，数据的信息熵
    1. 按特征m划分后，父节点分裂成k个子节点
    2. n表示父节点的样本个数
    3. ni 表示子节点i的样本个数
    选择准则：选择最大的GAIN 对应的特征m

举例

选择A或B两个特征构造节点，哪种方式好?

特征A	Yes	Yes	No	...	No	No
特征B	No	Yes	Yes	...	Yes	No
类别	C0	C0	C1		C1	C1

$Entropy(p)$

以特征A划分

特征A	Yes	Yes
特征B	No	Yes
类别	C0	C0

$Entropy(A_{Yes})$

特征A	No	...	No	No
特征B	Yes	...	Yes	No
类别	C1		C1	C1

$Entropy(A_{No})$

$$M = \frac{|A_{Yes}|}{n} Entropy(A_{Yes}) + \frac{|A_{No}|}{n} Entropy(A_{No})$$

$$Gain(A) = Entropy(p) - M$$

特征A	Yes	Yes	No	...	No	No
特征B	No	Yes	Yes	...	Yes	No
类别	C0	C0	C1		C1	C1

$Entropy(p)$

以特征B划分

特征A	Yes	No	...	No
特征B	Yes	Yes	...	Yes
类别	C0	C1		C1

$Entropy(B_{Yes})$

特征A	Yes	No
特征B	No	No
类别	C0	C1

$Entropy(B_{No})$

$$M = \frac{|B_{Yes}|}{n} Entropy(B_{Yes}) + \frac{|B_{No}|}{n} Entropy(B_{No})$$

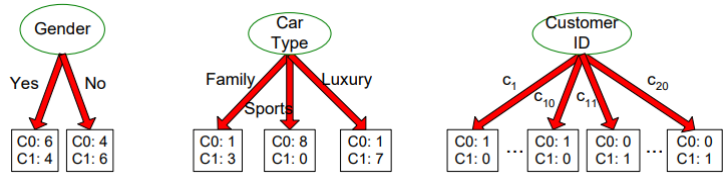
$$Gain(B) = Entropy(p) - M \quad < \quad Gain(A)$$

结论

信息增益能够较好地体现某个特征在降低信息不确定性方面的贡献
信息增益越大，说明信息纯度提升越快，最后结果的不确定性越低

不足

信息增益的局限性，**尤其体现在更偏好可取值较多的特征**
取值较多，不确定性相对更低，因此得到的熵偏低，但不一定有实际意义



特征Customer ID有最大的信息增益，因为每个子节点的熵均为0

回到餐厅的例子

对于训练集，$p = n = 6$，信息熵为 $I(\frac{6}{12}, \frac{6}{12}) = 1$ bit。

考虑属性Patrons和Type（以及其他属性）

$$IG(Patrons) = 1 - [\frac{2}{12} I(0,1) + \frac{4}{12} I(1,0) + \frac{6}{12} I(\frac{2}{6}, \frac{4}{6})] = .541 \text{ bits}$$

$$IG(Type) = 1 - [\frac{2}{12} I(\frac{1}{2}, \frac{1}{2}) + \frac{2}{12} I(\frac{1}{2}, \frac{1}{2}) + \frac{4}{12} I(\frac{2}{4}, \frac{2}{4}) + \frac{4}{12} I(\frac{2}{4}, \frac{2}{4})] = 0 \text{ bits}$$

从12个例子中学到的决策树：