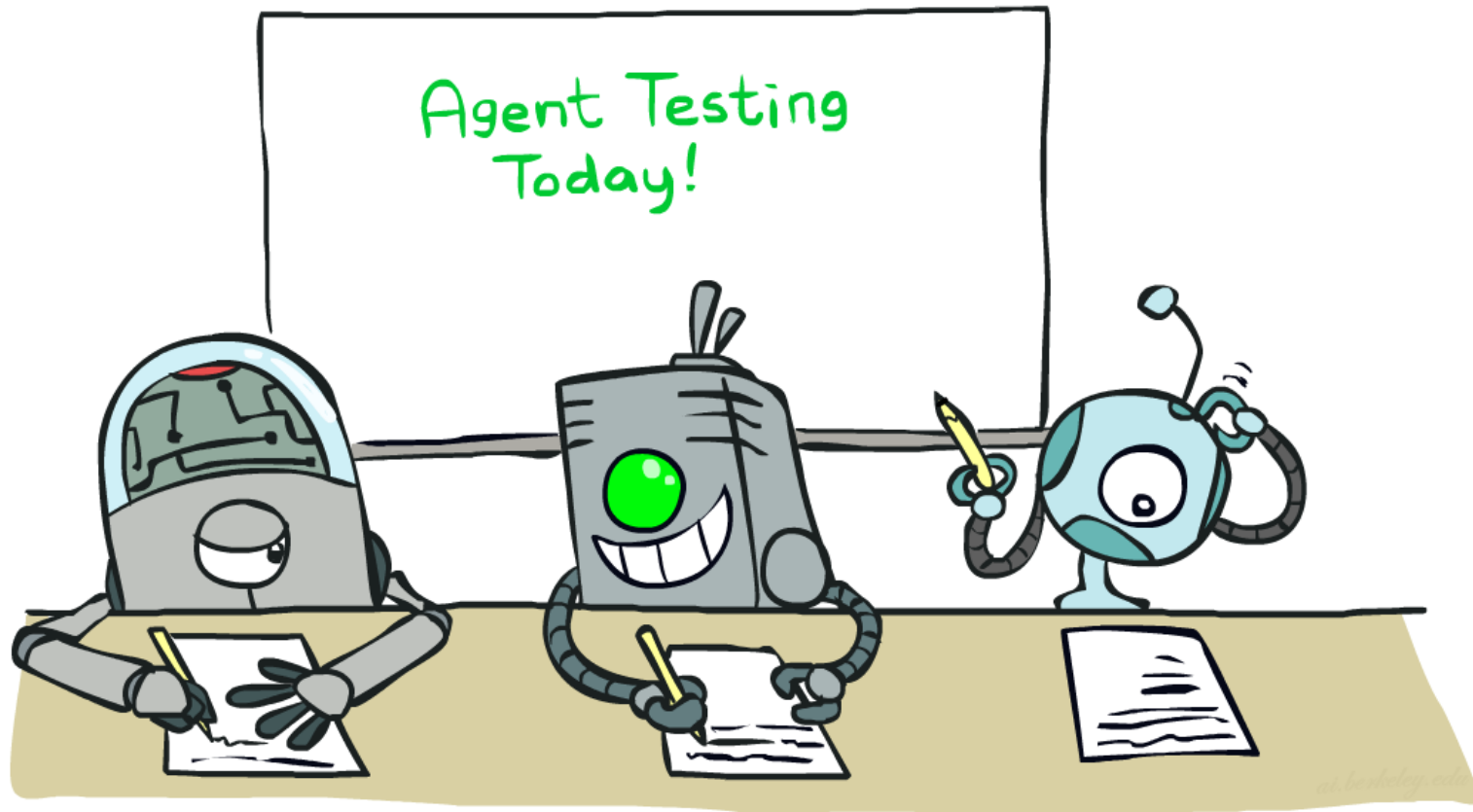


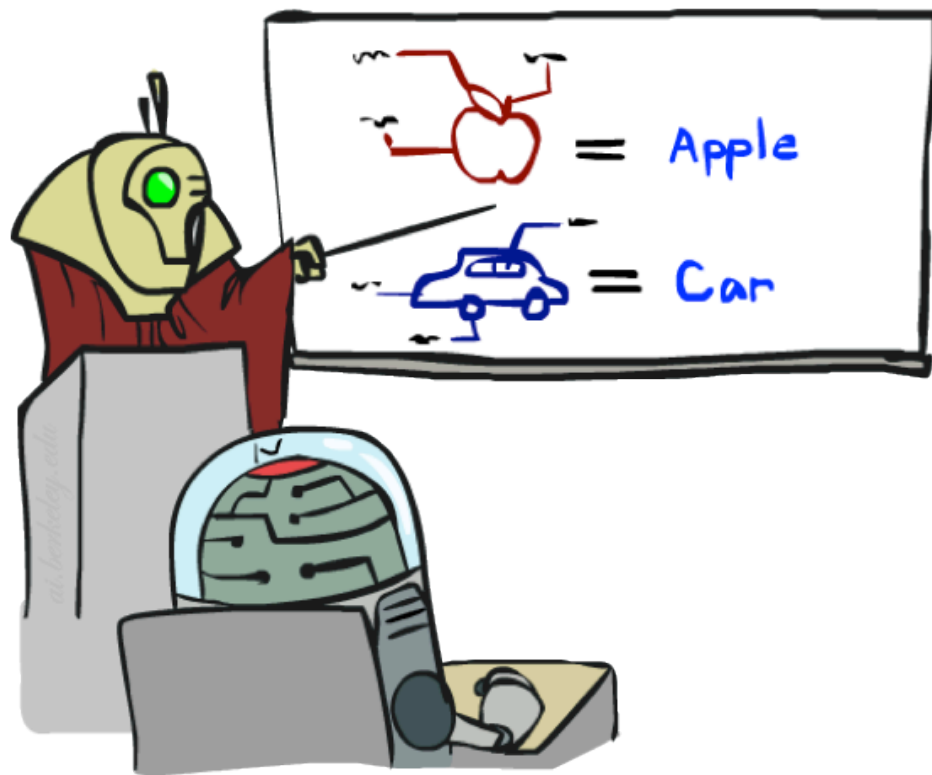
# 第十八章 样例学习

## Learning from examples



# Outline

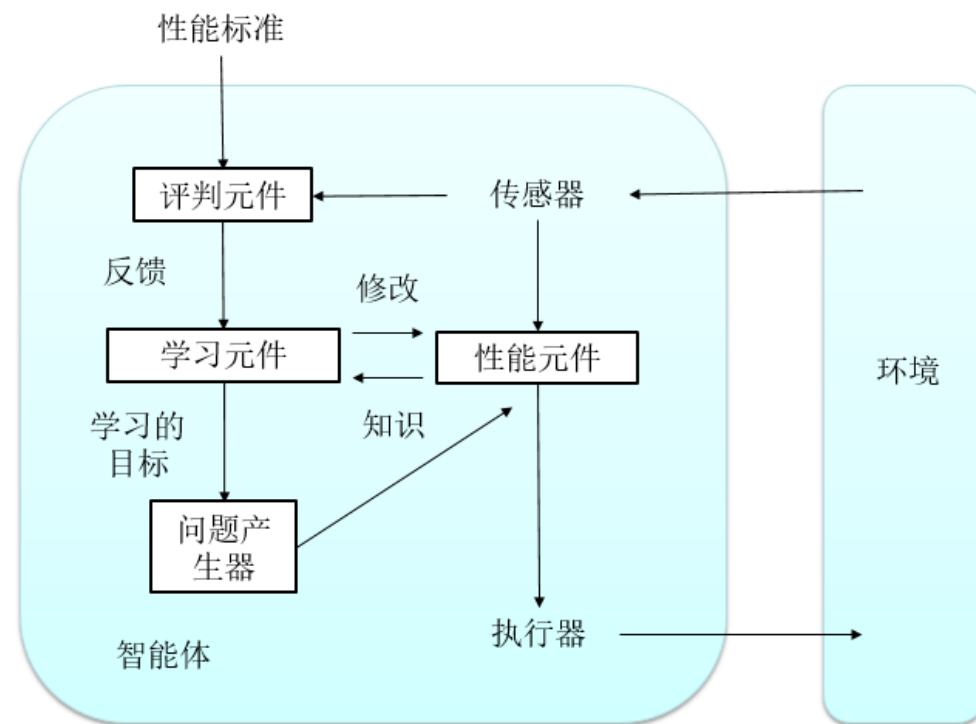
- 18.1 学习形式
- 18.2 监督学习
- 18.3 决策树归纳



# 18.1 学习形式

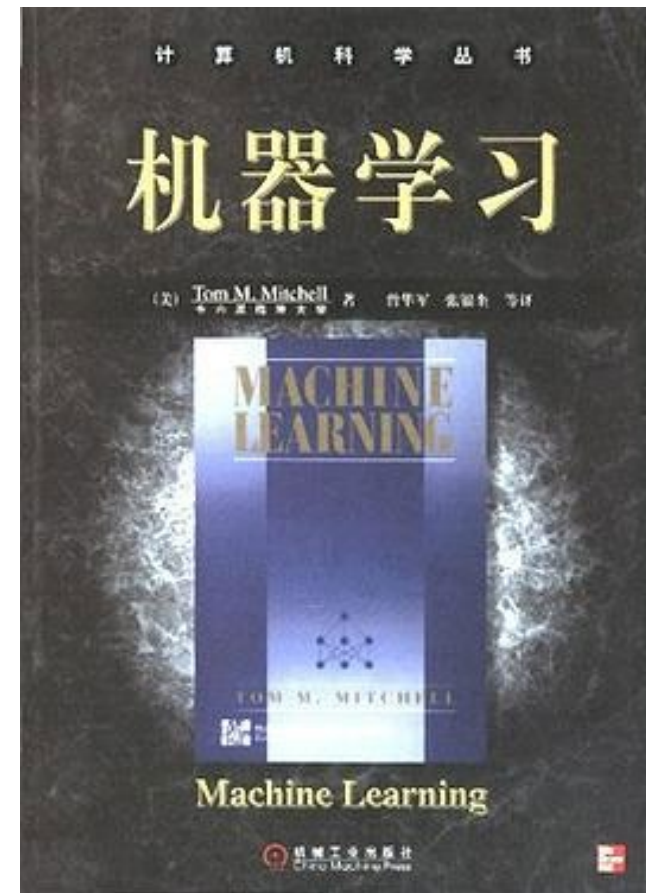
## ► 什么是学习

- 学习是一个过程，通过学习可以对Agent的性能进行改进
- Simon：学习就是系统中的变化，这种变化使系统比以前更有效地去做同样的工作
- 对于未知环境，缺少全知，学习是必要的
- Agent任何部件的性能都可以通过从数据中进行学习来改进



# 学习算法的定义

- **机器学习算法**是一种能够从数据中学习的算法。  
(Machine Learning/Example Learning/Inductive learning)
- “学习”是什么？
  - Mitchell (1997) 提供了一个简洁的定义：  
“对于某类**任务  $T$** 和**性能度量  $P$** ，一个计算机程序被认为可以从**经验  $E$** 中**学习**是指，  
通过经验  $E$  改进后，它在任务  $T$  上由性能度量  $P$  衡量的性能有所提升。



# 18.1 学习形式

- 智能体性能的改进依赖以下四个主要因素：

– 要改进哪个部件？

– 使用什么样的表示法？

– Agent具备什么样的先验知识？

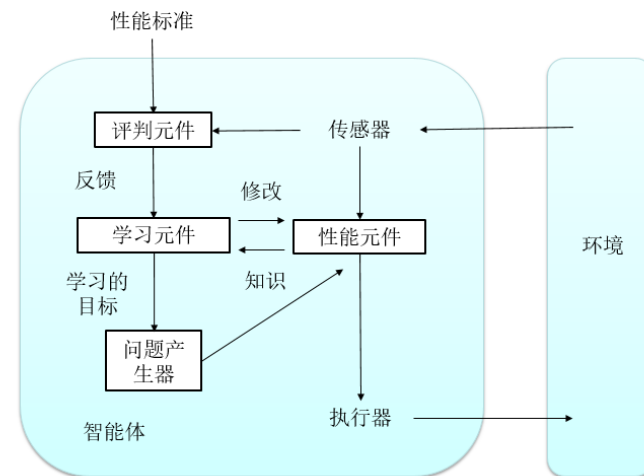
– 学习类型？

有监督学习

无监督学习/自监督学习

半监督学习

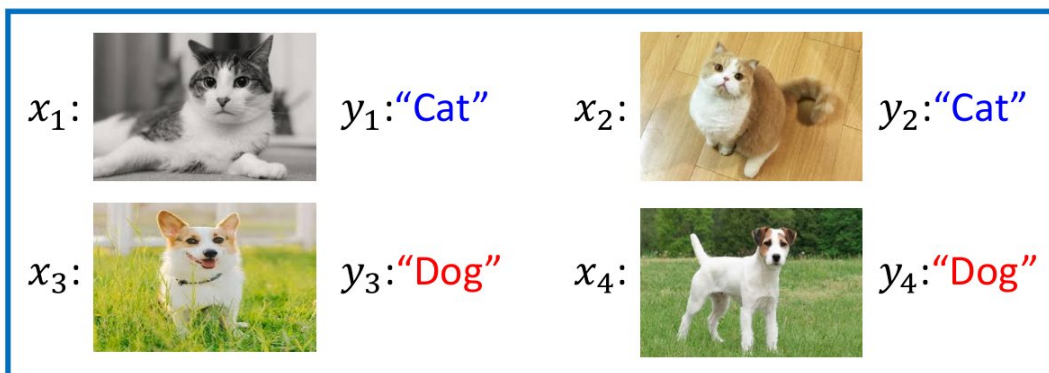
强化学习



- 问题求解Agent：状态空间
- 逻辑Agent：命题逻辑或一阶逻辑
- 不确定推理Agent：贝叶斯网络

# 18.1 学习形式

- 有监督学习 (Supervised learning )
  - 对每一个输入都有一个正确的目标输出，从一组输入 - 输出的实例数据集中，学习出一个“输入-输出”的映射函数： $Y=h(X)$  (模型参数)
  - 当新的数据到来时，可以根据h函数预测结果



Labelled Data

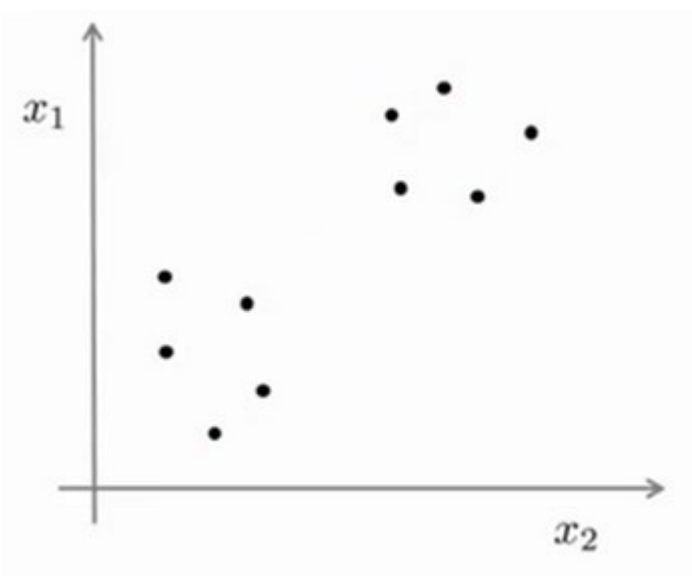
$$h \left( \text{img of orange cat} \right) = \text{"Cat"}$$

# 18.1 学习形式

- 无监督学习 (Unsupervised learning)

- 没有来自外部环境的直接反馈 (输出值), 自组织学习输入的模式
- 需要挖掘数据中的隐含规律
- 最常见的任务是聚类

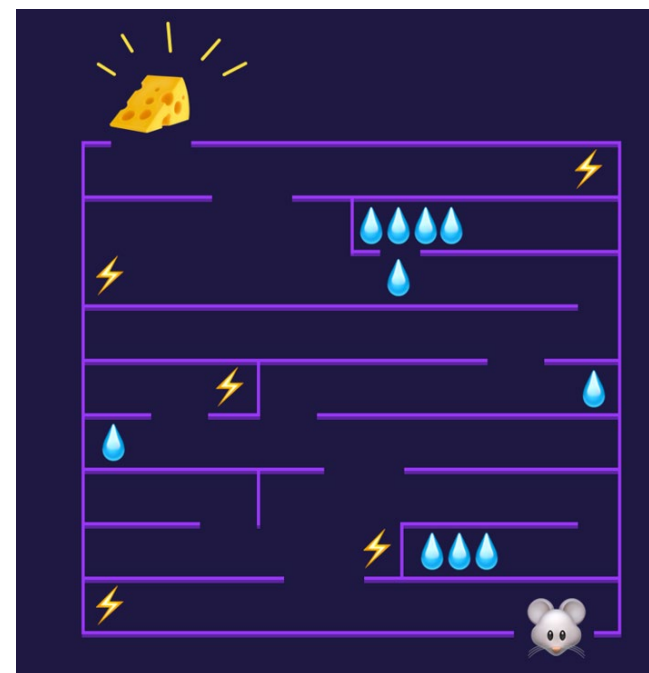
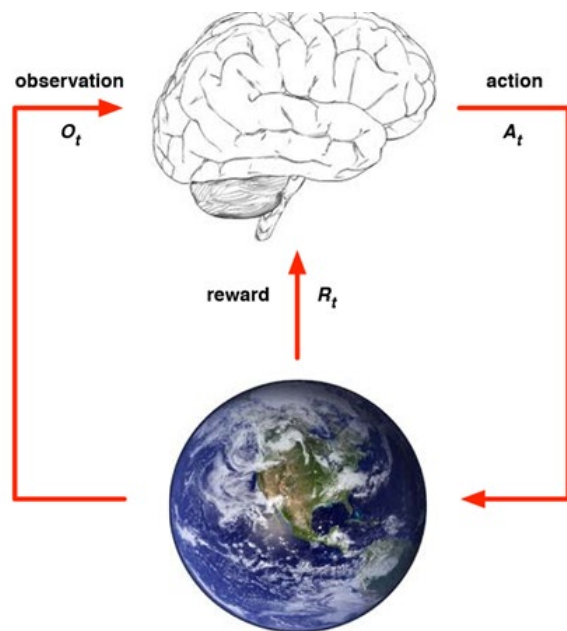
Training set:  $\{x^{(1)}, x^{(2)}, x^{(3)}, \dots, x^{(m)}\}$



## 自监督学习 (Self Supervised learning)

# 18.1 学习形式

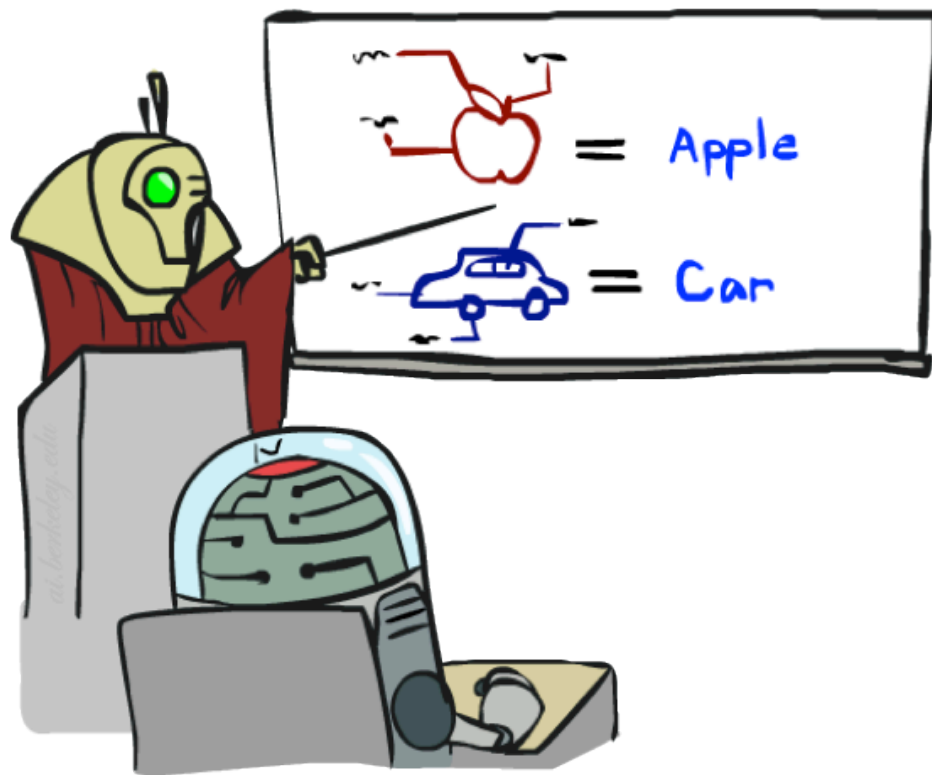
- 强化学习 (Reinforcement learning)
  - 外部环境仅给出一个对当前输出的一个评价 (奖赏或惩罚信号) , 不会给出具体的期望输出
  - 系统从环境学习以使得奖励最大





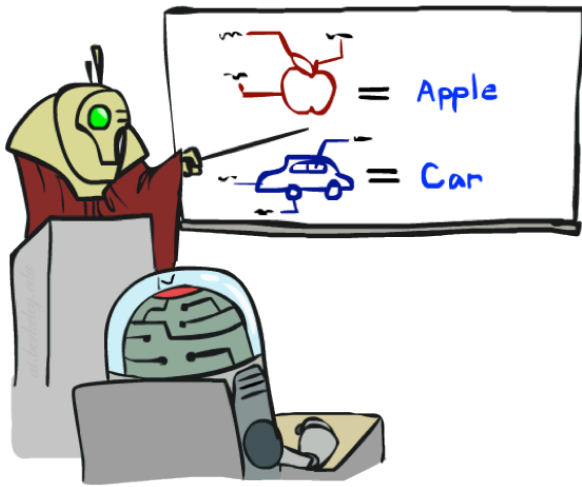
# Outline

- 18.1 学习形式
- 18.2 监督学习
- 18.3 决策树归纳



# 18.2 监督学习

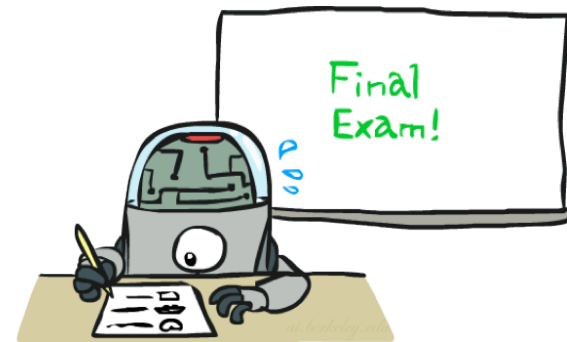
- 给定一个训练集 $N$ 个样本,  $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ , 其中,  $y_i = f(x_i)$ , 真实的 $f$ 是未知的。如果假说空间 $H$ 中存在一个假说或者函数 $h$  (hypothesis), 使得 $h \approx f$ , 就称  $h$  是一致假说 (consistent)



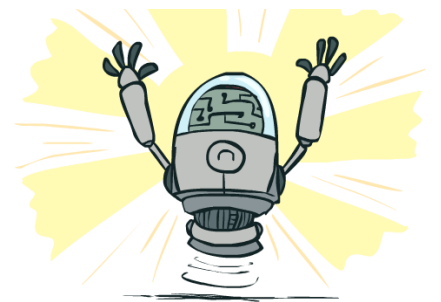
训练阶段



验证阶段



测试阶段



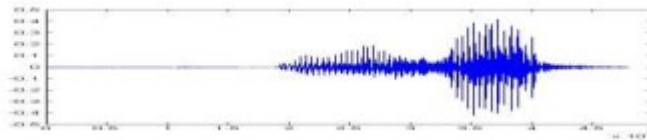
预测未来

# 18.2 监督学习

## ➤ 监督学习

- **分类**学习问题：如果输出 $y$ 是离散的值

语音识别



“How are you”

下围棋



“5-5”

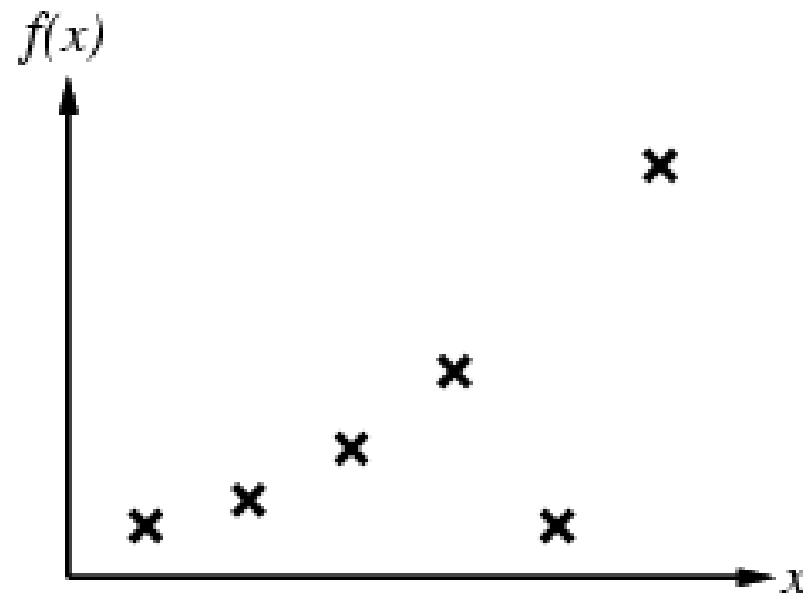
- **回归**学习问题：如果 $y$ 是一个连续的数值

- 预测明天的最高温度

$x, y$ 可以是任何形式的值

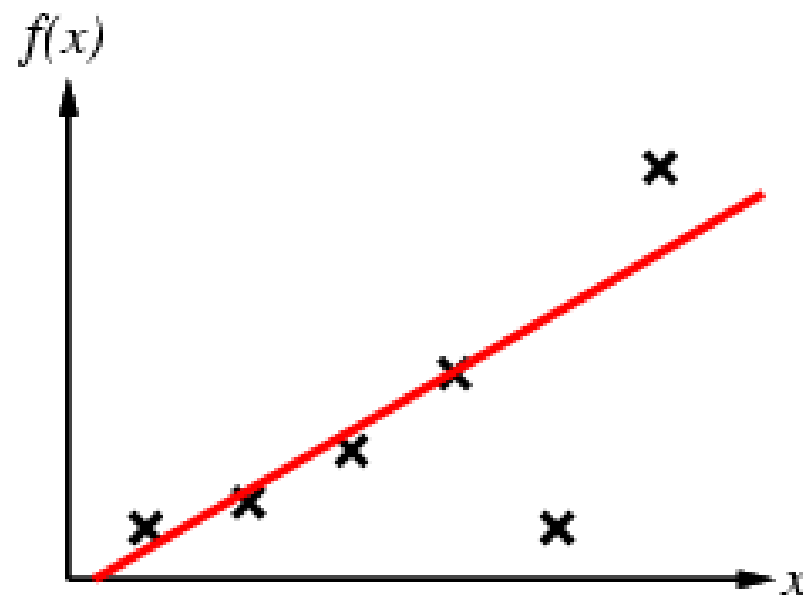
## 18.2 监督学习

- 举例：曲线拟合
  - 在某些数据点上拟合一个单变量函数，样例是 $(x, y)$ 平面上的点，其中 $y=f(x)$
  - 在真实函数 $f$ 未知的情况下，用一个函数 $h$ 逼近它



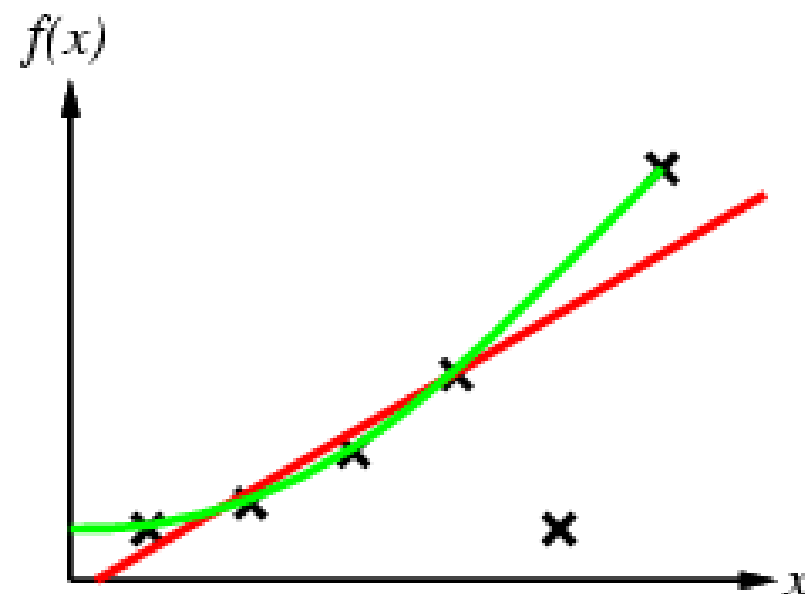
## 18.2 监督学习

- 举例：曲线拟合
  - 在某些数据点上拟合一个单变量函数，样例是 $(x, y)$ 平面上的点，其中 $y=f(x)$
  - 在真实函数 $f$ 未知的情况下，用一个函数 $h$ 逼近它



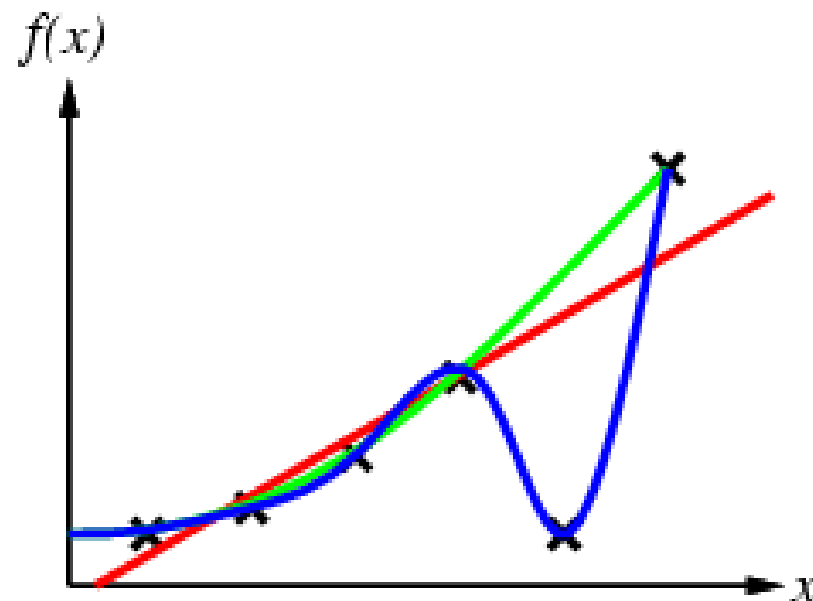
## 18.2 监督学习

- 举例：曲线拟合
  - 在某些数据点上拟合一个单变量函数，样例是 $(x,y)$ 平面上的点，其中 $y=f(x)$
  - 在真实函数 $f$ 未知的情况下，用一个函数 $h$ 逼近它



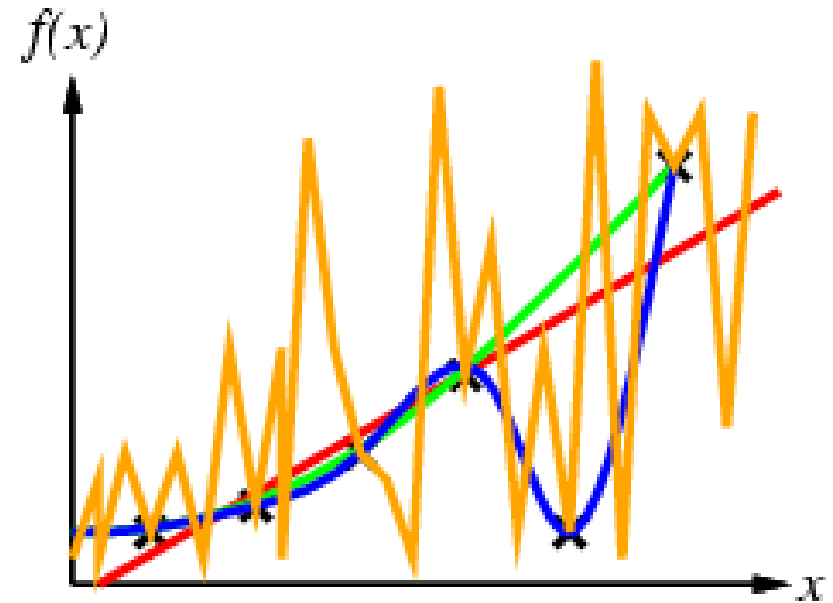
## 18.2 监督学习

- 举例：曲线拟合
  - 在某些数据点上拟合一个单变量函数，样例是 $(x, y)$ 平面上的点，其中 $y=f(x)$
  - 在真实函数 $f$ 未知的情况下，用一个函数 $h$ 逼近它



## 18.2 监督学习

- 举例：曲线拟合
  - 在真实函数 $f$ 未知的情况下，用一个函数 $h$ 逼近它
  - 较好拟合训练数据的复杂假说和更好泛化的简单假说之间存在折中



归纳学习中的基本问题：如何从多个一致假说中抉择？

答案：选择与数据一致的最简单的假说（奥坎姆剃刀）



# 监督学习

归纳偏好 (inductive bias)

Sequence:  $x_1, x_2, x_3, x_4, x_5, \dots$   
1, 2, 3, 4, ?,  $\dots$

$x_5 = 5$ , since  $x_i = i$  for  $i = 1..4$ .

$x_5 = 29$ , since  $x_i = i^4 - 10i^3 + 35i^2 - 49i + 24$ .

Conclusion: We prefer 5, since linear relation involves less arbitrary parameters than 4th-order polynomial.

# 监督学习

Sequence: 14159265358979323846264338327950288419716939937?

Looks random?!

Frequency estimate:  $n$  = length of sequence.  $k_i$  = number of occurred  $i \implies$  Probability of next digit being  $i$  is  $\frac{k_i}{n}$ .

Asymptotically  $\frac{k_i}{n} \rightarrow \frac{1}{10}$  (seems to be) true.

But we have the strong feeling that (i.e. with high probability) the next digit will be 5 because the previous digits were the expansion of  $\pi$ .

Conclusion: We prefer answer 5, since we see more structure in the sequence than just random digits.

**归纳偏好(inductive bias)** 对应了学习算法本身所做出的关于"什么样的模型更好"的假设

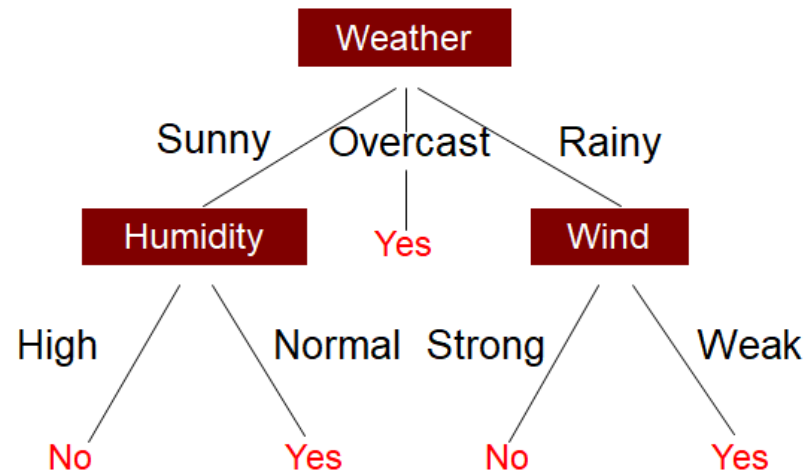
# Outline

- 机器学习
- 18.3 决策树归纳
  - 18.3.1 决策树表示法
  - 18.3.2 决策树的表达能力
  - 18.3.3 从样例归纳决策树
  - 18.3.4 选择测试属性



# Decision Trees

- 决策树归纳是一类最简单的机器学习形式，是一种以样例为基础的归纳学习方法
- 从一组训练数据中学习到的函数，一棵决策树表示一个函数： $y = f(x)$ 
  - 输入 $x$ ：属性值向量
  - 输出 $y$ ：一个决策
- 输入输出可以离散的，也可以连续的
- 布尔分类（二分类）
  - 输入值是离散的，输出为二值的情况
  - 输出为真：正例
  - 输出为假：反例



Play Tennis的决策树

# 决策树学习 - 就餐问题

基于下面的10个属性，决定是否要在餐馆等座位？

1. Alternate（候选）：附近是否有另一家合适的餐馆？
2. Bar（酒吧）：该餐馆中供顾客等候的吧区是否舒适？
3. Fri/Sat（周五/周六）是周五或周六吗？
4. Hungry（饥饿）是否饥饿？
5. Patrons(顾客)：该餐馆中有多少顾客(None, Some, Full)
6. Price（价格）：餐馆的价格范围(\$, \$\$, \$\$\$)
7. Raining（下雨）外面是否在下雨？
8. Reservation（预约）：是否预约过？
9. Type（类型）：餐馆的种类(French, Italian, Thai, Burger)
10. WaitEstimate（等候时间估计）：估计的等候时间(0-10, 10-30, 30-60, >60)

# 基于属性的表示

- 决定是否要在餐馆等座位的实例集 $X_1 \sim X_{12}$
- 实例是通过属性值描述的

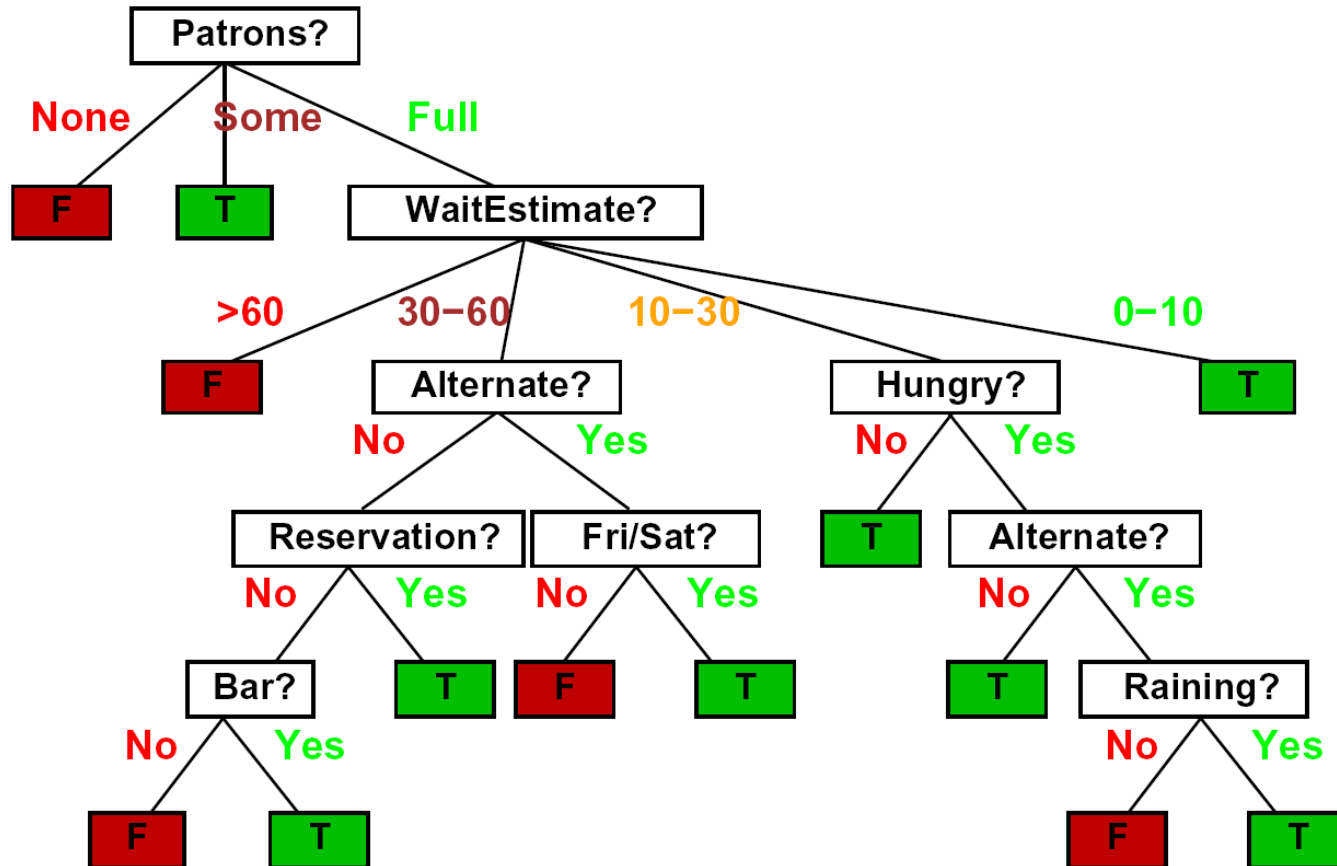
Example	Attributes										Target
	<i>Alt</i>	<i>Bar</i>	<i>Fri</i>	<i>Hun</i>	<i>Pat</i>	<i>Price</i>	<i>Rain</i>	<i>Res</i>	<i>Type</i>	<i>Est</i>	<i>Wait</i>
$X_1$	T	F	F	T	Some	\$\$\$	F	T	French	0-10	T
$X_2$	T	F	F	T	Full	\$	F	F	Thai	30-60	F
$X_3$	F	T	F	F	Some	\$	F	F	Burger	0-10	T
$X_4$	T	F	T	T	Full	\$	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

- 实例分类：正 (T)、负 (F)

输入属性值的可能组合9216种，  
仅通过12个样例学习，对缺失  
的9204个输出值给出预测。

# Decision Trees

- 假设空间的一种可能表示
- 决策树通过把实例从根节点排列到某个叶子节点来分类



- 一棵决策树表示一个函数:  $y = f(x)$

Example	Attributes										Target
	Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est	Wait

- $x = (\text{Yes}, \text{No}, \text{No}, \text{Yes}, \text{Some}, \dots, \text{No})$
- $y = ?$

# Outline

---

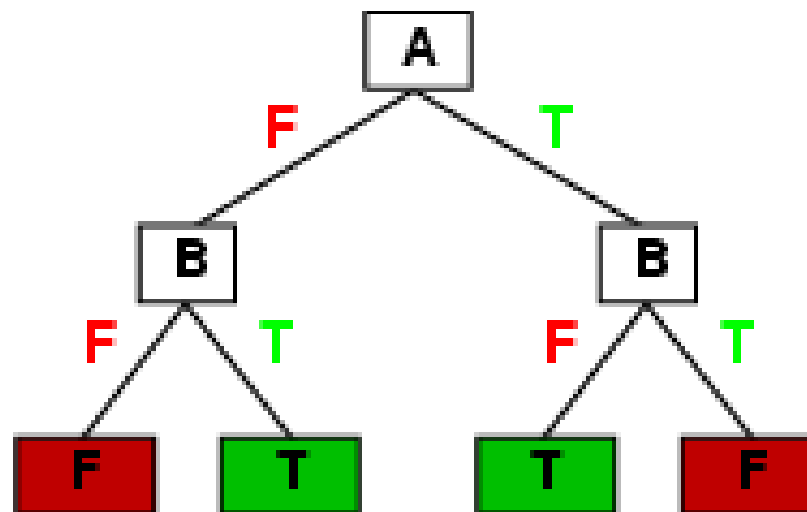
- 18.2 监督学习
- 18.3 决策树归纳
  - 18.3.1 决策树表示法
  - 18.3.2 决策树的表达能力
  - 18.3.3 从样例归纳决策树
  - 18.3.4 选择测试属性



# 决策树的表示能力

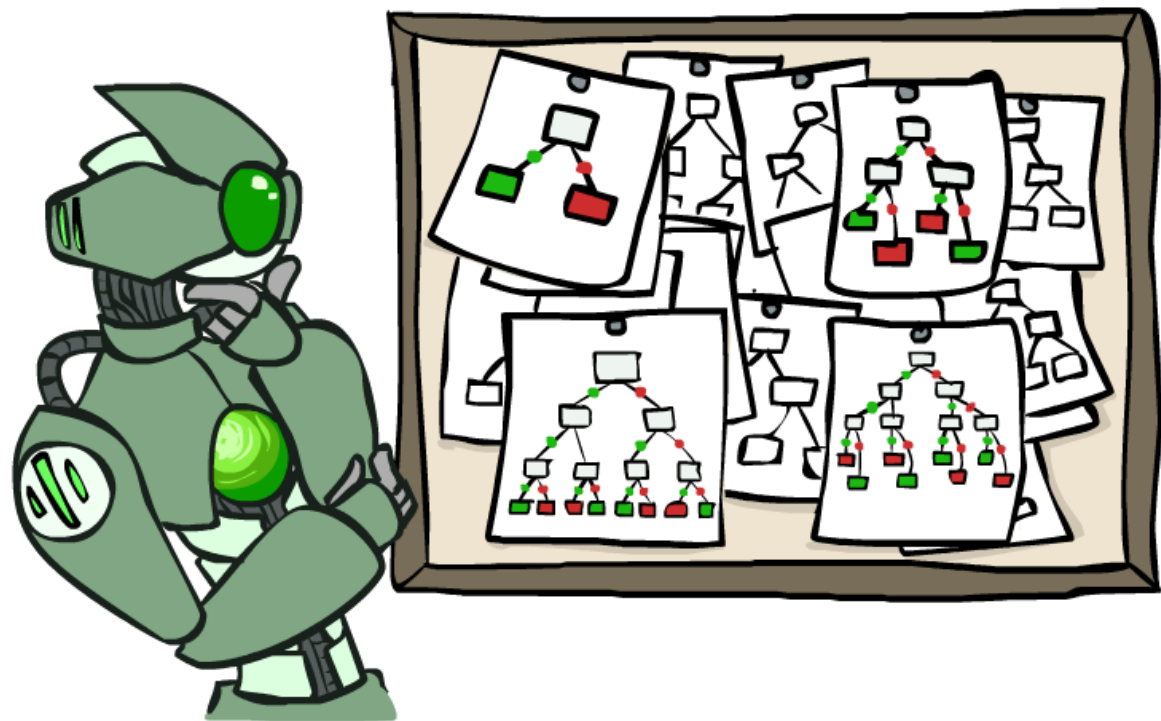
- 决策树能表示输入属性的任何函数
- 对布尔型函数, 真值表中每一行对应树中根到叶节点的一条路径:

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



# 决策树的表示能力

- 对于具有10个布尔属性的饭店例子，有 $2^{1024}$ 或者大约 $10^{308}$ 个候选的函数
- 在如此大的空间中寻找好的假说，需要设计精巧的算法



# 决策树学习要点

---

- **目标**: 找到和训练集一致的**较小的树** (树中所有的路径都很短, 整棵树的规模比较小)
- **思想**: 递归地选择“**最好**”或“**最佳**”的属性作为树或子树的根, 通过较少数量的测试就能得到正确的分类
- **最好**: **分类能力最好**

# Outline

---

- 18.2 监督学习
- 18.3 决策树归纳
  - 18.3.1 决策树表示法
  - 18.3.2 决策树的表达能力
  - 18.3.3 从样例归纳决策树
  - 18.3.4 选择测试属性

# Features and examples

- 就餐问题的样例

- 12 个训练样例
- 10个分类属性
- 目标 $WillWait$ (真:正例, 假:反例)

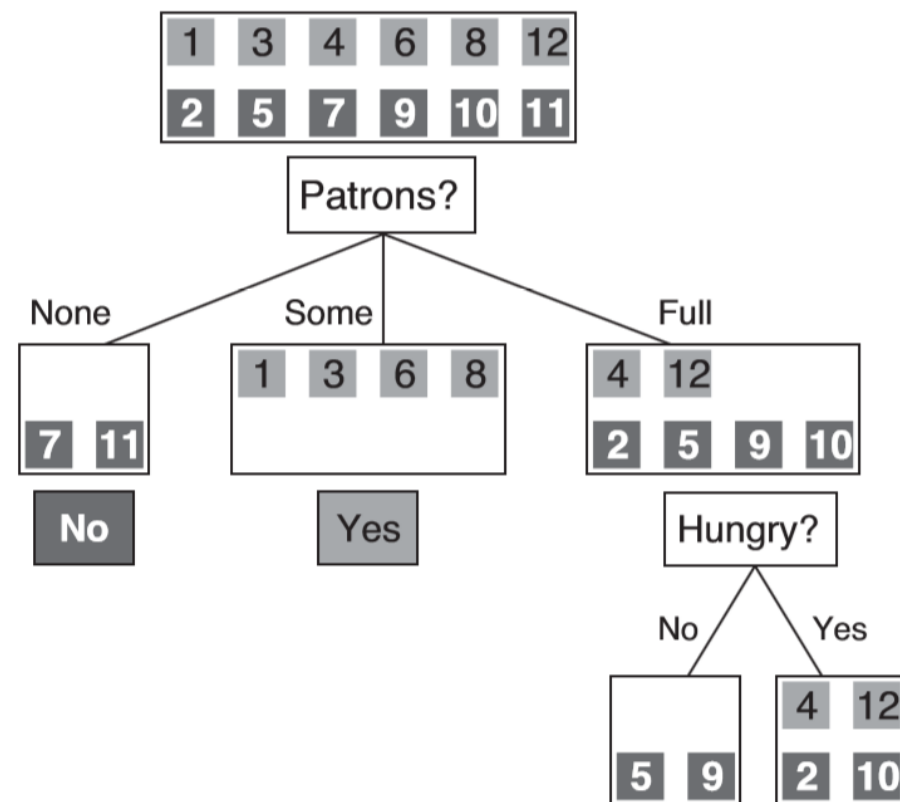
Example	Attributes										Target
	$Alt$	$Bar$	$Fri$	$Hun$	$Pat$	$Price$	$Rain$	$Res$	$Type$	$Est$	$WillWait$
$X_1$	$T$	$F$	$F$	$T$	$Some$	$$$$$	$F$	$T$	$French$	$0-10$	$T$
$X_2$	$T$	$F$	$F$	$T$	$Full$	$\$$	$F$	$F$	$Thai$	$30-60$	$F$
$X_3$	$F$	$T$	$F$	$F$	$Some$	$\$$	$F$	$F$	$Burger$	$0-10$	$T$
$X_4$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

目标：寻找一棵决策树：与样例一致，且规模尽可能小。

即：通过较少的测试达到正确分类。

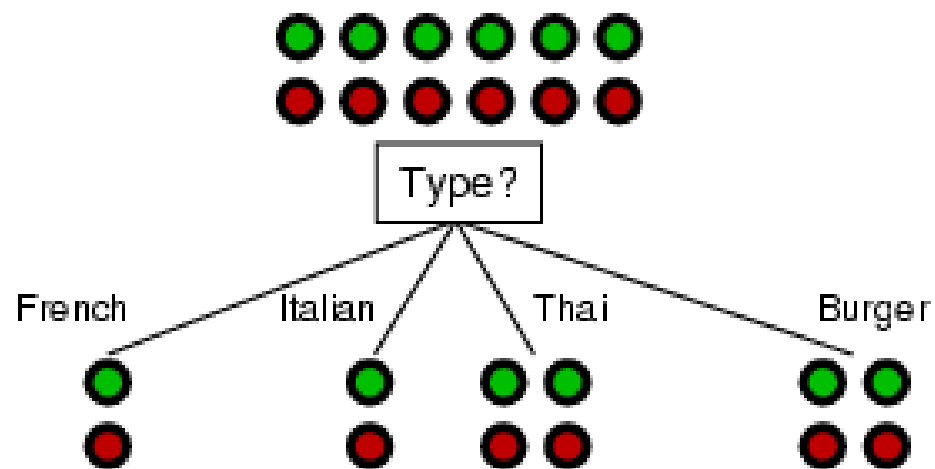
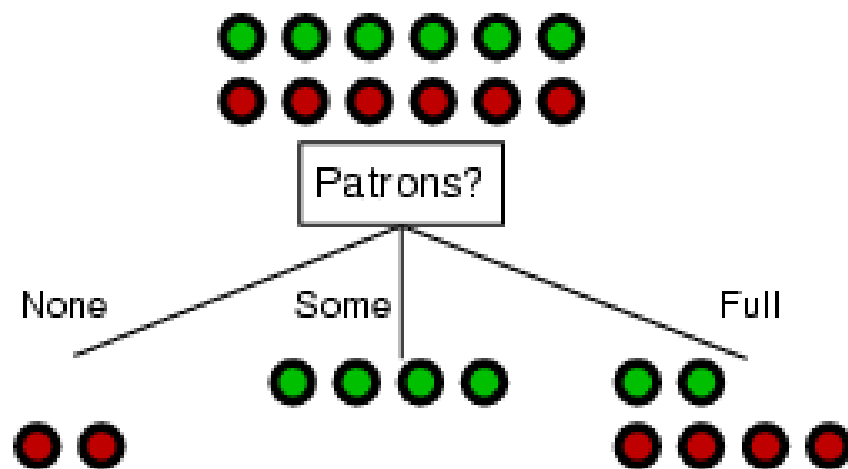
# Decision Tree Learning

- 采用贪婪“分而治之”（Divide- and -conquer）的策略
  - 将问题分解为更小的子问题，这些子问题又可以被递归求解
  - 总是优先测试最重要的属性
  - “最重要的属性”：对于样例分类具有最大差异的属性



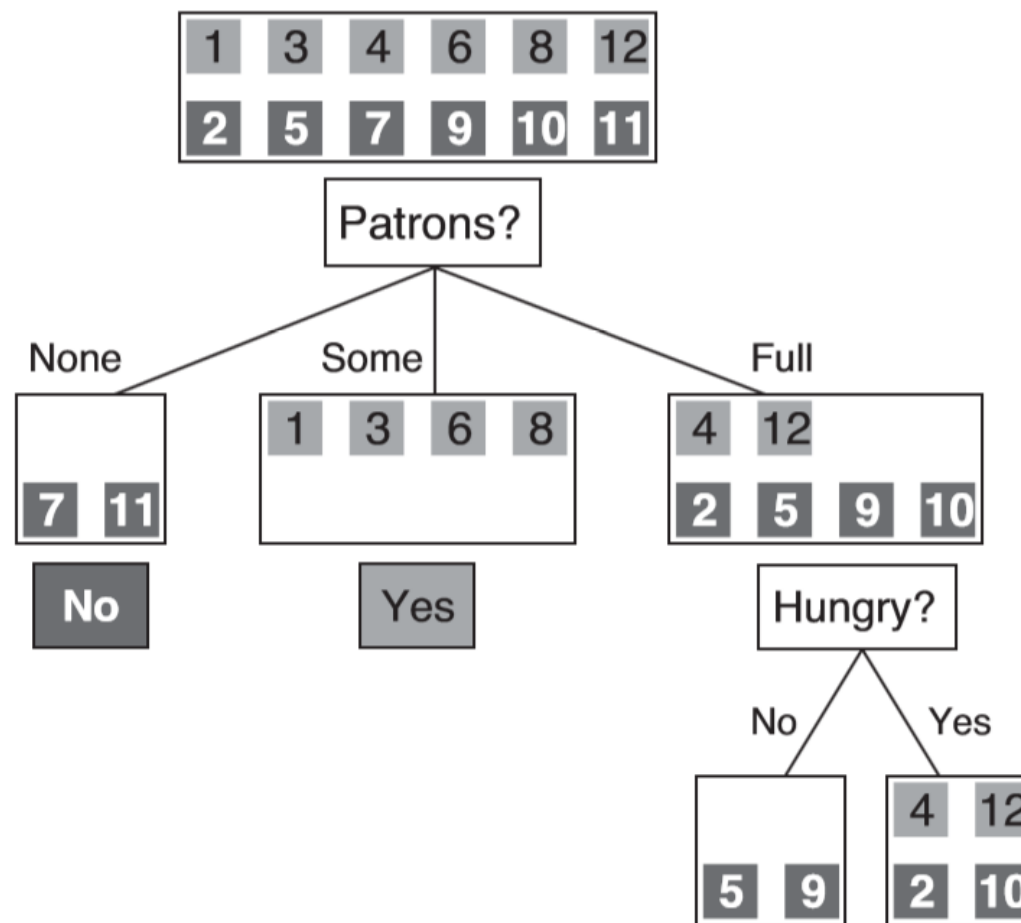
# 选择一个属性

- 采用贪婪“分而治之”（Divide- and -conquer）的策略
  - 总是优先测试最重要的属性
  - “最重要的属性”：对于样例分类具有最大差异的属性



# Decision Tree Learning

- 决策树生成要考虑4种情况：
  - 1.如果剩余样例都是正例（或反例），则返回，可回答Yes或No
  - 2.如果既有正例又有反例，则选择最好属性继续分裂
  - 3.如果没有留下任何样例，则返回一个缺省值（父节点样例中最常见的输出）
  - 4.如果没有属性，返回剩余样例中得票最多的分类。





# Decision Tree Learning

- Aim: find a small tree consistent with the training examples
- Idea: (recursively) choose “most significant” attribute as root of (sub)tree

**function** DECISION-TREE-LEARNING(*examples*, *attributes*, *parent\_examples*) **returns**  
a tree

**if** *examples* is empty **then return** PLURALITY-VALUE(*parent\_examples*)  
**else if** all *examples* have the same classification **then return** the classification  
**else if** *attributes* is empty **then return** PLURALITY-VALUE(*examples*)  
**else**

$A \leftarrow \operatorname{argmax}_{a \in \text{attributes}} \text{IMPORTANCE}(a, \text{examples})$

*tree*  $\leftarrow$  a new decision tree with root test *A*

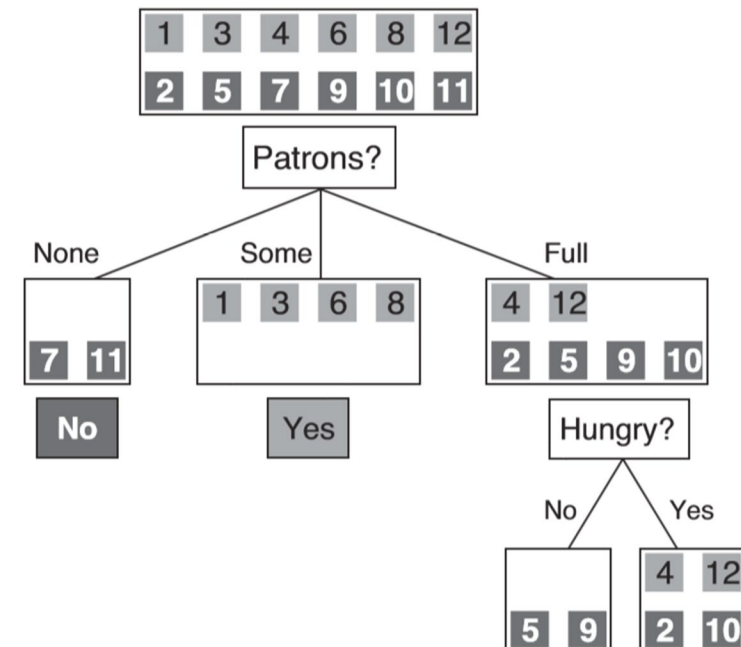
**for each** value  $v_k$  of *A* **do**

$\text{exs} \leftarrow \{e : e \in \text{examples} \text{ and } e.A = v_k\}$

*subtree*  $\leftarrow$  DECISION-TREE-LEARNING(*exs*, *attributes* – *A*, *examples*)

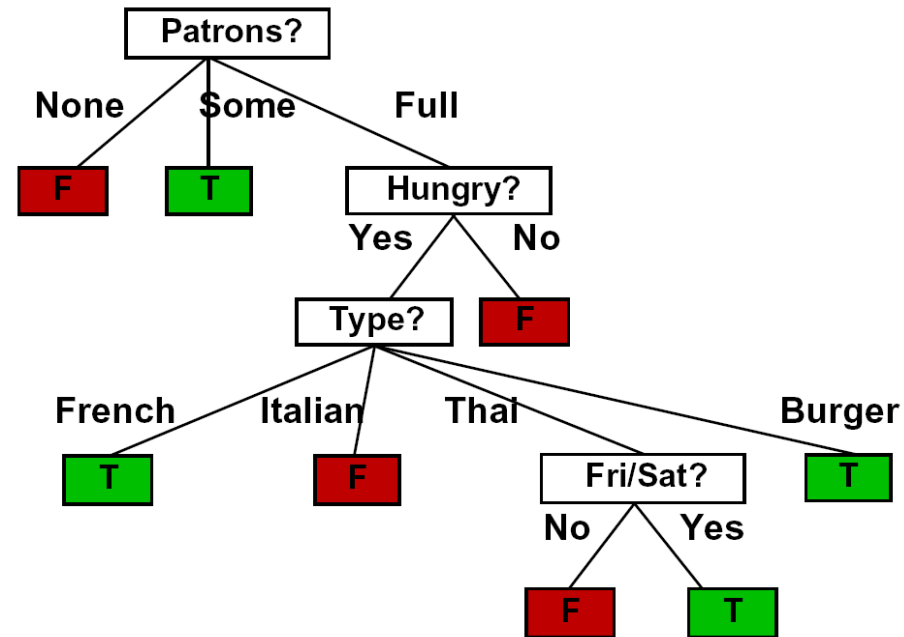
add a branch to *tree* with label ( $A = v_k$ ) and subtree *subtree*

**return** *tree*



# Example: Learned Tree

- Decision tree learned from these 12 examples:



- Substantially simpler than “true” tree
- Also: it’s reasonable

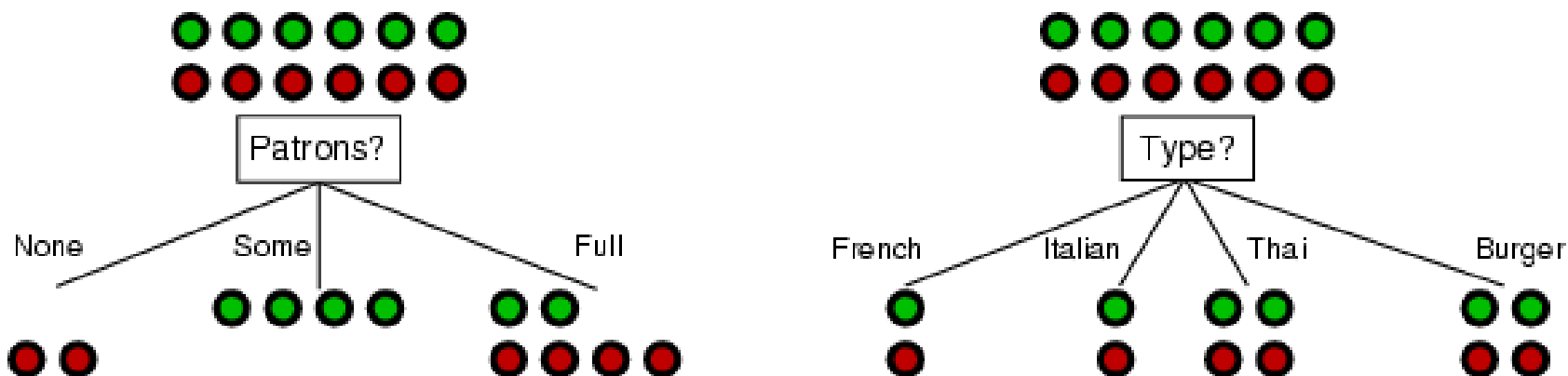
# Outline

---

- 18.2 监督学习
- 18.3 决策树归纳
  - 18.3.1 决策树表示法
  - 18.3.2 决策树的表达能力
  - 18.3.3 从样例归纳决策树
  - 18.3.4 选择测试属性

# 选择一个属性

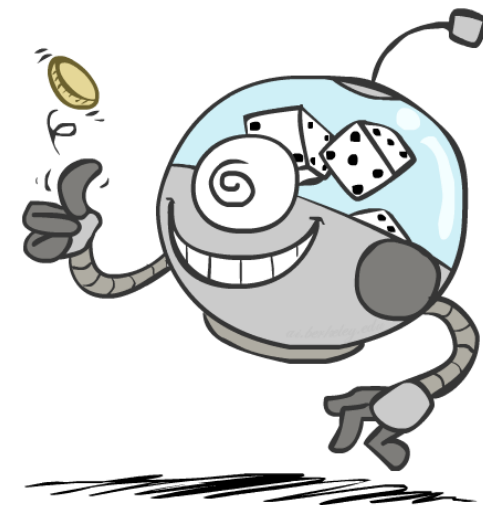
- **思想:** 理想的属性是将实例分为只包含正例或只包含反例的集合



- *Patrons?* 不理想，但是很不錯
- we need a measure of how “good” a split is, even if the results aren’t perfectly separated out

# Entropy and Information

- 形式化度量“相当好”和“真正无用”，使用信息收益的概念定义属性的Importance
- 熵是信息论中的基本量（Shannon, Weaver, 1949）
  - 熵是随机变量的不确定性度量，量化整个概率分布中的不确定性总量
  - 不确定性越小，熵越小
  - 单位: 比特
  - 一般地，设随机变量 $v$ 取值为 $v_k$ 的概率： $P(v_k)$ ，则 $v$ 的熵定义为：



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

# Entropy and Information

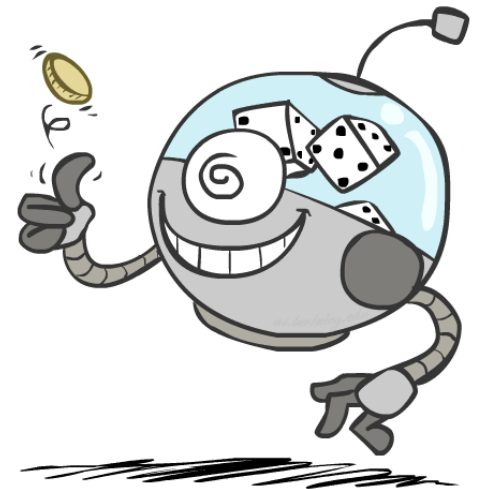
- 熵是信息论中的基本量 (Shannon, Weaver, 1949)

- 熵是随机变量的不确定性度量

- Scale: bits

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

- Answer to Boolean question with prior  $\langle 1/2, 1/2 \rangle$ ? 抛硬币的熵
- Answer to 4-way question with prior  $\langle 1/4, 1/4, 1/4, 1/4 \rangle$ ? 四面色子的熵
- Answer to 4-way question with prior  $\langle 0, 0, 0, 1 \rangle$ ?
- Answer to 3-way question with prior  $\langle 1/2, 1/4, 1/4 \rangle$ ?

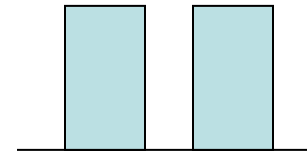


# Entropy

- Also called the **entropy** of the distribution
  - More uniform = higher entropy
  - More values = higher entropy
  - More peaked = lower entropy
- 设布尔随机变量以 $q$ 的概率为真，则可以定义该变量的熵：

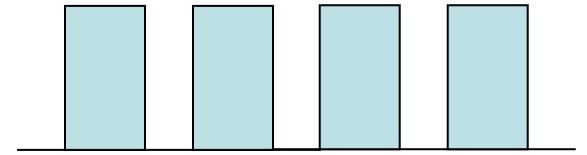
$$B(q) = -(q \log_2 q + (1 - q) \log_2 (1 - q))$$

$\langle 1/2, 1/2 \rangle$

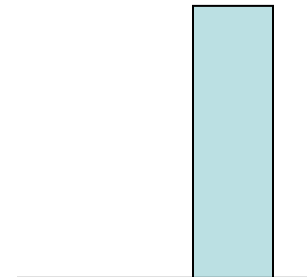


1 bit

$\langle 1/4, 1/4, 1/4, 1/4 \rangle$

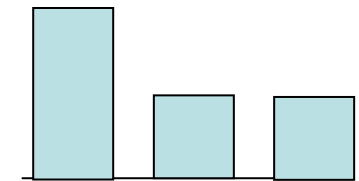


2 bit



0 bits

$\langle 0, 0, 0, 1 \rangle$



1.5 bit

$\langle 1/2, 1/4, 1/4 \rangle$

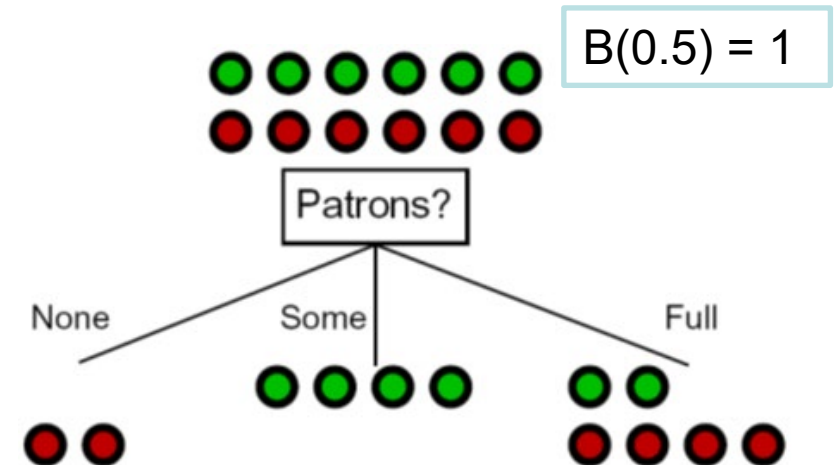
# Entropy

- 设布尔随机变量以 $q$ 的概率为真，则可以定义该变量的熵：

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

- 例如，一个训练集包括 $p$ 个正例样本和 $n$ 个负例样本，目标属性的熵：

$$\begin{aligned} H(Goal) &= B\left(\frac{p}{p+n}\right) \\ &= -\frac{p}{p+n} \log_2 \frac{p}{p+n} - \frac{n}{p+n} \log_2 \frac{n}{p+n} \end{aligned}$$

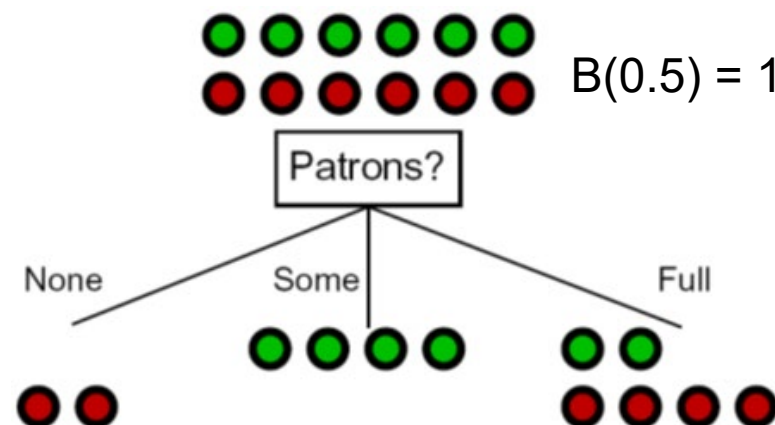




# 信息增益

- **信息增益** (Information Gain) : 一个属性A的信息增益就是由于使用这个属性分裂样例而导致的**期望熵**降低, 即不确定性的减少量

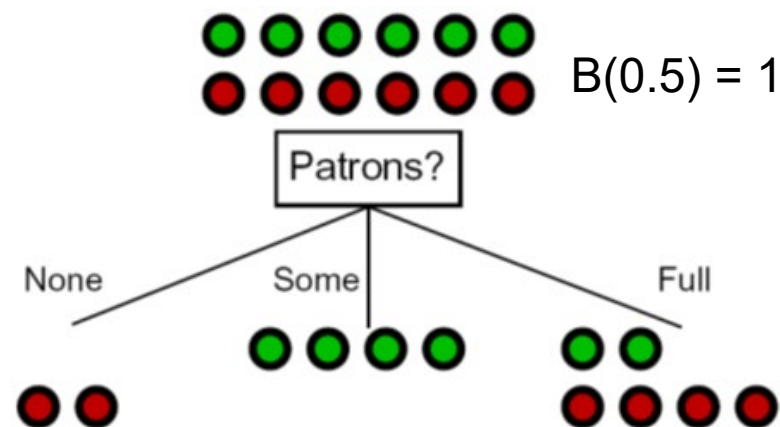
$$Gain(A) = B(\frac{p}{p+n}) - Remainder(A)$$



# 信息增益

- *Remainder(A)*是用属性A测试后剩余的期望熵：一个属性A假定有 $d$ 个不同的取值，根据其取值可以将数据集E分成 $E_1, \dots, E_d$ 子集，每个子集 $E_i$ 包含 $p_i$ 个正例， $n_i$ 个负例，

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



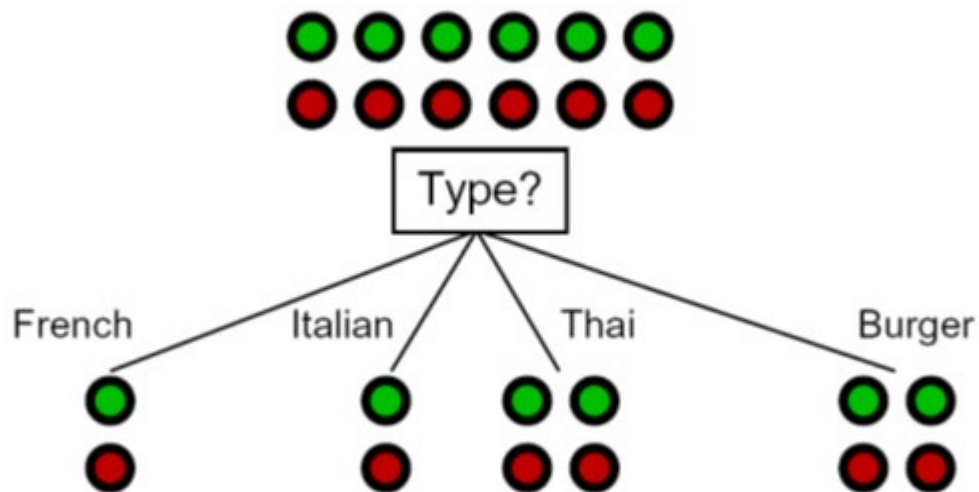
$$Remainder(A) = \frac{2}{12} B\left(\frac{0}{2}\right) + \frac{4}{12} B\left(\frac{4}{4}\right) + \frac{6}{12} B\left(\frac{2}{6}\right)$$

# 信息增益

- 按属性A分裂的信息增益，是熵的期望降低的量

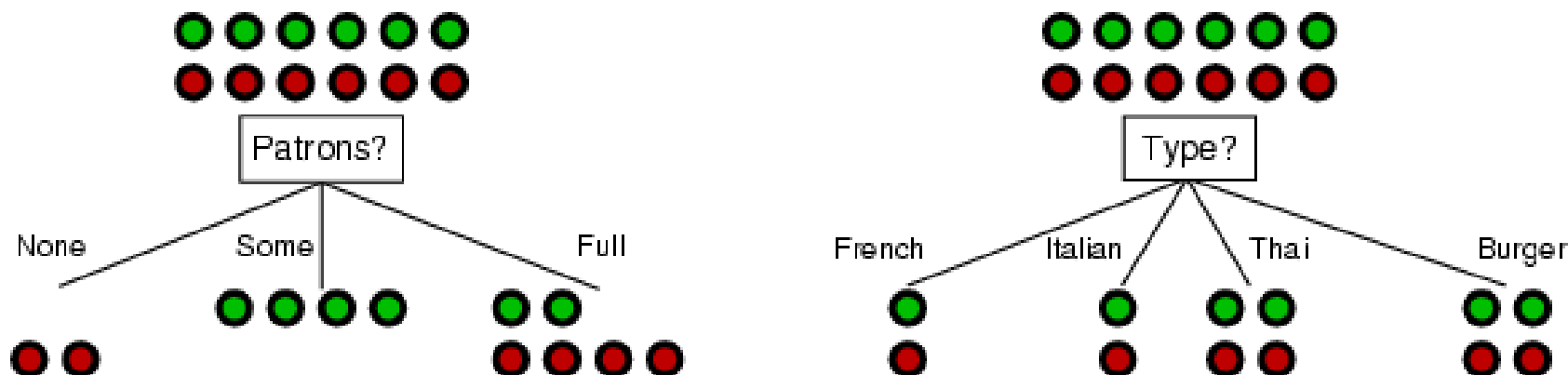
$$Gain(A) = B(\frac{p}{p+n}) - Remainder(A)$$

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



$$Gain(Type) = ?$$

# 信息增益



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

$$Gain(Type) = 1 - \left[ \frac{2}{12} B\left(\frac{1}{2}\right) + \frac{2}{12} B\left(\frac{1}{2}\right) + \frac{4}{12} B\left(\frac{2}{4}\right) + \frac{4}{12} B\left(\frac{2}{4}\right) \right] = 0 \text{ 比特}$$

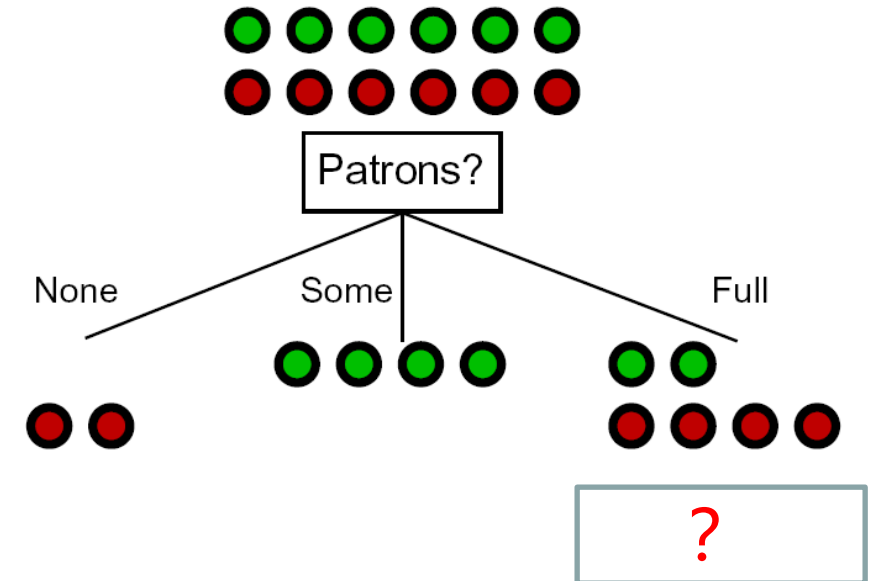
其他的属性的信息增益也可以进行类似的计算

**结论：** Patrons 具有最高的信息增益，被决策树算法选为决策树的根

# Next Step: Recurse

- 继续找下一个属性!
- Two branches are done
- What to do under “full”?
  - See what examples are there...

Example	Attributes										Target
	<i>Alt</i>	<i>Bar</i>	<i>Fri</i>	<i>Hun</i>	<i>Pat</i>	<i>Price</i>	<i>Rain</i>	<i>Res</i>	<i>Type</i>	<i>Est</i>	<i>WillWait</i>
$X_1$	<i>T</i>	<i>F</i>	<i>F</i>	<i>T</i>	<i>Some</i>	<i>\$\$\$</i>	<i>F</i>	<i>T</i>	<i>French</i>	<i>0-10</i>	<i>T</i>
$X_2$	<i>T</i>	<i>F</i>	<i>F</i>	<i>T</i>	<i>Full</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Thai</i>	<i>30-60</i>	<i>F</i>
$X_3$	<i>F</i>	<i>T</i>	<i>F</i>	<i>F</i>	<i>Some</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Burger</i>	<i>0-10</i>	<i>T</i>
$X_4$	<i>T</i>	<i>F</i>	<i>T</i>	<i>T</i>	<i>Full</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Thai</i>	<i>10-30</i>	<i>T</i>
$X_5$	<i>T</i>	<i>F</i>	<i>T</i>	<i>F</i>	<i>Full</i>	<i>\$\$\$</i>	<i>F</i>	<i>T</i>	<i>French</i>	<i>&gt;60</i>	<i>F</i>
$X_6$	<i>F</i>	<i>T</i>	<i>F</i>	<i>T</i>	<i>Some</i>	<i>\$\$</i>	<i>T</i>	<i>T</i>	<i>Italian</i>	<i>0-10</i>	<i>T</i>
$X_7$	<i>F</i>	<i>T</i>	<i>F</i>	<i>F</i>	<i>None</i>	<i>\$</i>	<i>T</i>	<i>F</i>	<i>Burger</i>	<i>0-10</i>	<i>F</i>
$X_8$	<i>F</i>	<i>F</i>	<i>F</i>	<i>T</i>	<i>Some</i>	<i>\$\$</i>	<i>T</i>	<i>T</i>	<i>Thai</i>	<i>0-10</i>	<i>T</i>
$X_9$	<i>F</i>	<i>T</i>	<i>T</i>	<i>F</i>	<i>Full</i>	<i>\$</i>	<i>T</i>	<i>F</i>	<i>Burger</i>	<i>&gt;60</i>	<i>F</i>
$X_{10}$	<i>T</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>Full</i>	<i>\$\$\$</i>	<i>F</i>	<i>T</i>	<i>Italian</i>	<i>10-30</i>	<i>F</i>
$X_{11}$	<i>F</i>	<i>F</i>	<i>F</i>	<i>F</i>	<i>None</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Thai</i>	<i>0-10</i>	<i>F</i>
$X_{12}$	<i>T</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>Full</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Burger</i>	<i>30-60</i>	<i>T</i>



# 总结-选择测试属性

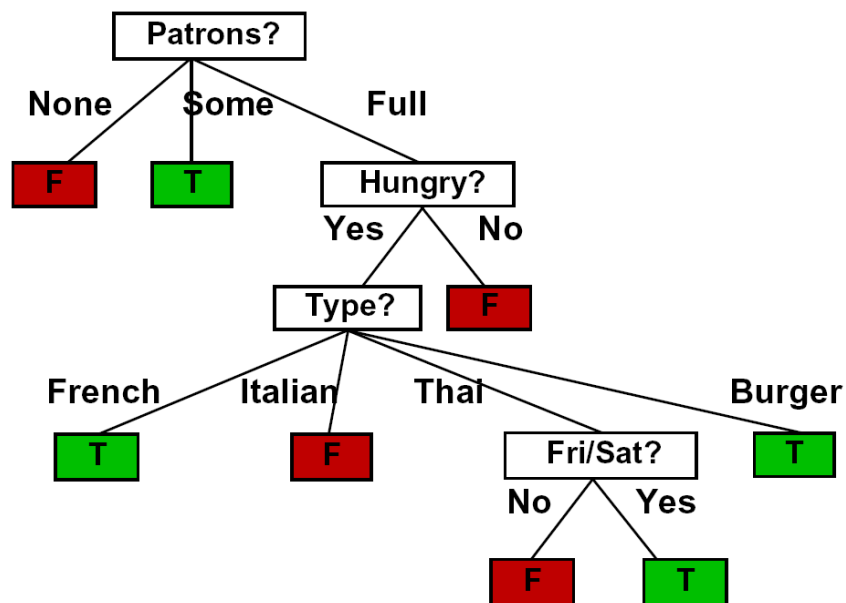
Example	Attributes										Target
	Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est	WillWait
$X_1$	T	F	F	T	Some	\$\$\$	F	T	French	0-10	T
$X_2$	T	F	F	T	Full	\$	F	F	Thai	30-60	F
$X_3$	F	T	F	F	Some	\$	F	F	Burger	0-10	T
$X_4$	T	F	T	T	Full	\$	F	F	Thai	10-30	T
$X_5$	T	F	T	F	Full	\$\$\$	F	T	French	>60	F
$X_6$	F	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

$$H(\text{Goal}) = B\left(\frac{p}{p+n}\right)$$

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

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

$$\text{Gain}(A) = B\left(\frac{p}{p+n}\right) - \text{Remainder}(A)$$





谢谢！

