机器学习: Part 1

- 机器学习概论
- k-近邻分类
- 决策树学习
- 统计学习
- 贝叶斯网络的参数学习

*Slides based on those of Pascal Poupart

什么是机器学习？

一个计算机程序关于某类任务T和性能度量P从经验E中学习，如果以P衡量的它在任务T中的表现，随着经验E而提高。[Mitchell, 1997]

常见学习任务

- 有监督学习(Supervised learning):给定一些输入-输出对阳例，学习一个把输入映射到输出的函数。
- 无监督学习(Unsupervised learning):为样例寻找自然的类
半监督学习包含大量未标注数据和少量标注数据。主要是利用未标注中的信息，辅助标注数据，进行监督学习
- 强化学习(Reinforcement learning):根据一系列奖励或惩罚来决定要做什么

示例

- 跳棋(强化学习):
  - T: 玩跳棋
  - P: 比赛中战胜对手的百分比
  - E: 玩练习游戏来对抗自己
- 手写识别(有监督学习):
  - T: 识别图像中的手写单词
  - P: 正确识别单词的百分比
  - E: 有标签的手写单词数据库
- 客户分析(Customer profiling) (无监督学习):
  - T: 基于交易模式的客户聚类
  - P: 簇的同种性
  - E: 客户交易数据库

表示

- 学习到的信息的表示很重要
  - 决定学习算法如何工作
- 常见表示：
  - 线性加权多项式
  - 命题逻辑
  - 一阶逻辑
  - 贝叶斯网络
  - ...

监督学习

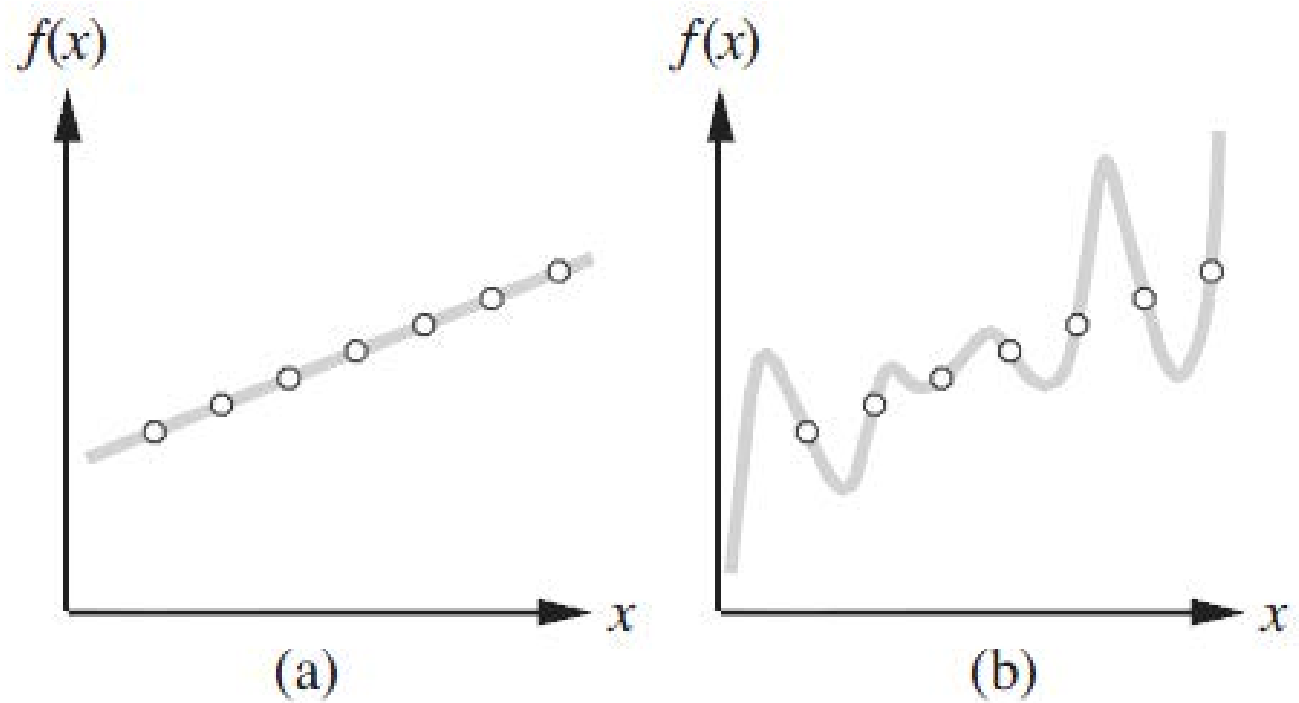
- 定义: 给定一组N个输入-输出对样例的训练集(x_1, y_1), $(x_2, y_2), \ldots, (x_N, y_N)$, 其中每个$y_j$由某个未知函数$y = f(x)$生成，发现逼近真实函数$f$的一个函数$h$。
- 函数h是一个假设。
- 学习是在可能的假设空间中寻找一个即使是在训练集之外的新例子上都表现良好的假设。

分类与回归

- 当输出y的取值范围是有穷的，学习问题称为分类(classification).
- 如果只有两个值，则称为布尔或二元分类。
- 当y是一个数值时，学习问题被称为回归(regression).

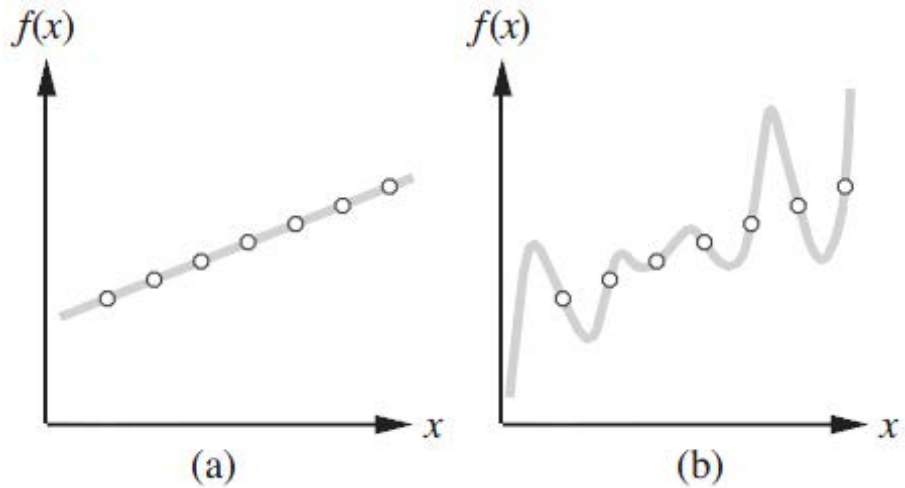
一个回归示例

- 用一个单变量的函数拟合某些数据点
- 一个假设是一致的如果它符合所有数据
- 一个线性假设和一个7阶多项式假设



假设空间

- 假设空间：考虑的所有假设h的集合
- *e.g.*, 多项式集合
- 如何从多个一致的假设中进行选择?
- 偏好与数据一致的最简单的假设。
- 这一原理被称为奥坎姆剃刀(Ockham's razor)，它反对所有形式的复杂性。
- *e.g.*, (a)应优于(b)。

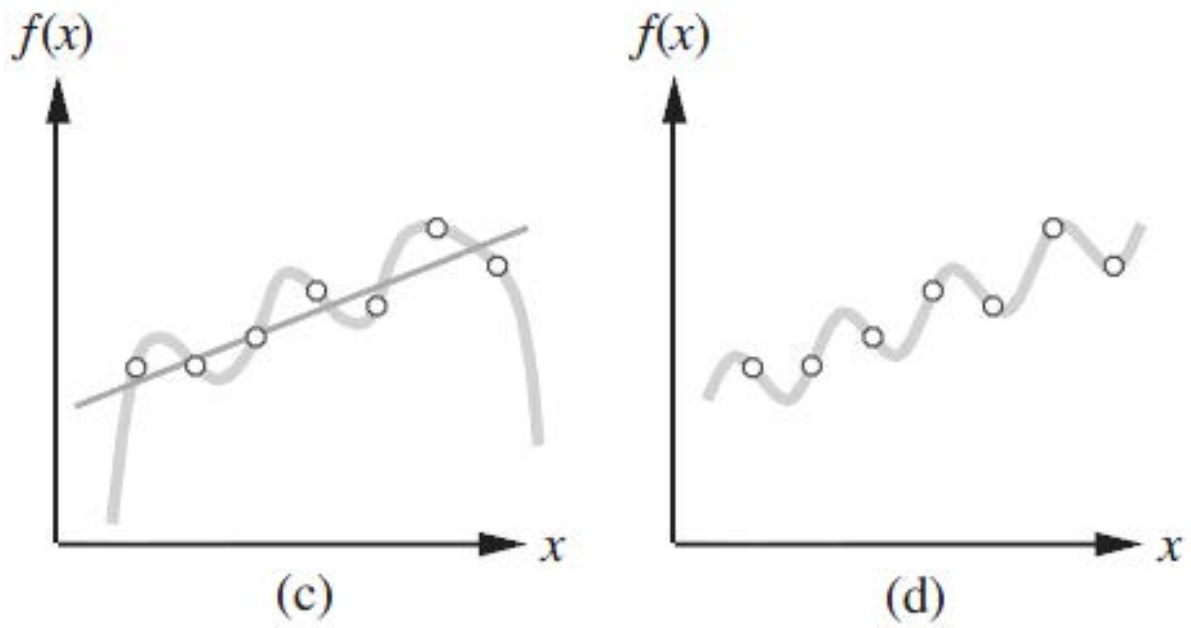


泛化性(Generalization)

- 一个好的假设可以很好地泛化, *i.e.*, 很好地预测未知的例子
- 一般来说，在很好地拟合训练数据的复杂假设和可能更好地泛化的简单假设之间存在权衡

一个示例

- 关于这个数据集没有一致的直线
- 需要一个6阶多项式来精确拟合
- 可以通过一个$ax + b + c\sin(x)$形式的简单函数精确拟合



可实现性

- 找到一致的假设取决于假设空间
- 我们说一个学习问题是可以实现的如果假设空间包含真实函数。
- 然而，我们不能总是判断给定的学习问题是否是可实现的，因为真实函数是未知的。

可实现的

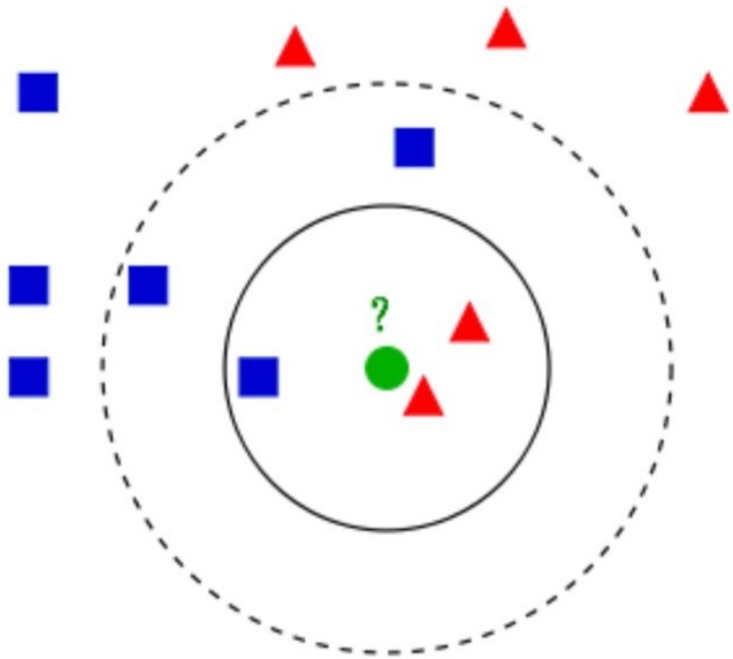
- 既然每个可计算函数都可以由某个图灵机表示，为什么不让H为所有Java程序或图灵机的集合呢？
- 这是因为在假设空间的表达能力和在该空间中找到一个好假设的复杂性之间存在权衡。
- *e.g.*, 将直线拟合到数据很容易；拟合高阶多项式比较困难；而拟合图灵机通常是不可判定的。

k-近邻分类

- 给定一个训练数据集，
- 对新的输入实例，在训练数据集中找到与该实例最邻近的k个实例，
- 这k个实例的多数属于某个类，就把该输入实例分类到这个类中。

示例

- 有两类不同的样本数据，分别用蓝色的小正方形和红色的小三角形表示，
- 而图正中间的那个绿色的圆表示待分类的数据。



- 若k=3, 分类结果为红色；若k=5, 分类结果为蓝色

距离的度量

- 两个n维向量$x = (x_1, x_2, \ldots, x_n)$与$y = (y_1, y_2, \ldots, y_n)$间的闵可夫斯基距离定义为（这里$p \geq 1$）：

$$L_p(x, y) = (\sum_{i=1}^{n} |x_i - y_i|^p)^{\frac{1}{p}}.$$

- 当$p = 2$时，称为欧式距离(Euclidean distance)，即

$$L_2(x, y) = (\sum_{i=1}^{n} |x_i - y_i|^2)^{\frac{1}{2}}.$$

- 当$p = 1$时，称为曼哈顿距离(Manhattan distance)，即

$$L_1(x, y) = \sum_{i=1}^{n} |x_i - y_i|.$$

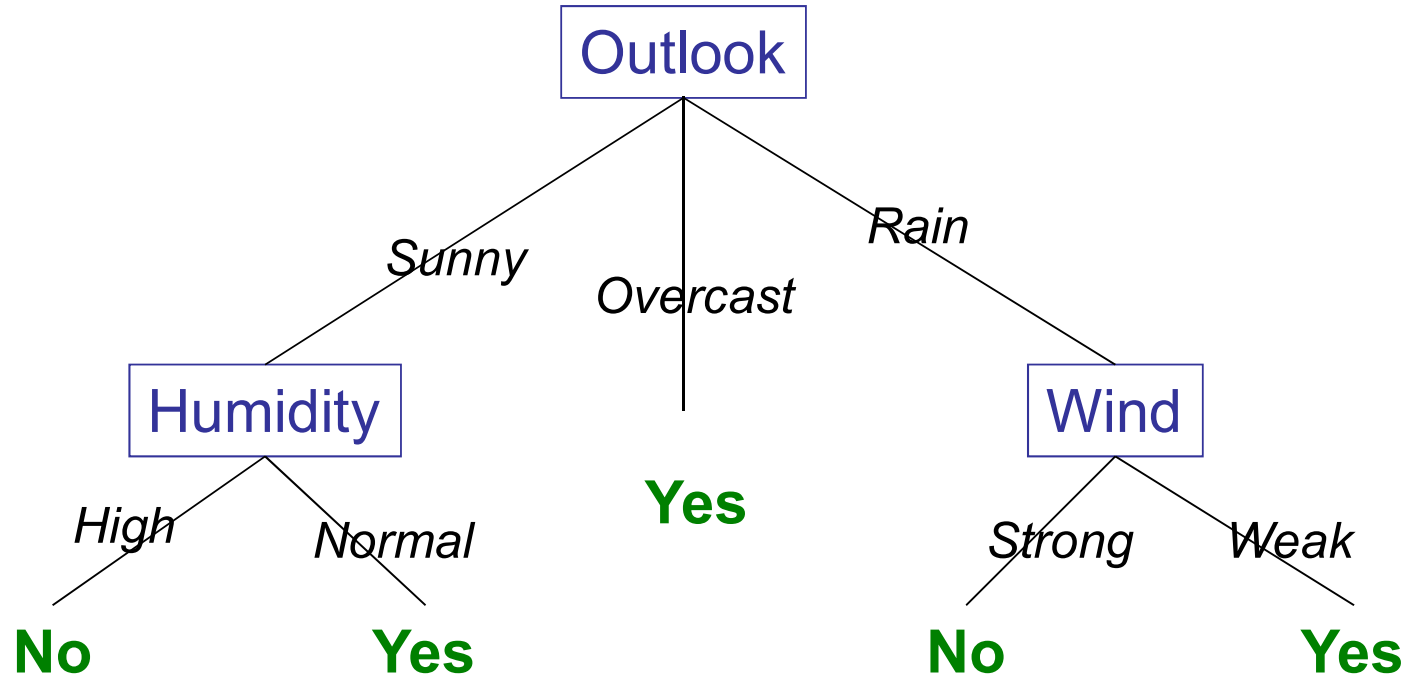
- 当$p = \infty$时，它是各个坐标距离的最大值，即

$$L_\infty(x, y) = \max_i |x_i - y_i|.$$

决策树

- 表示一个函数，该函数将属性值的向量作为输入，并返回一个“决策”（单个输出值）。
- 通过执行一系列测试来做出决策。
- 节点：用属性标记
- 边：用属性值标记
- 叶子：用决策标记

一个例子（打网球）



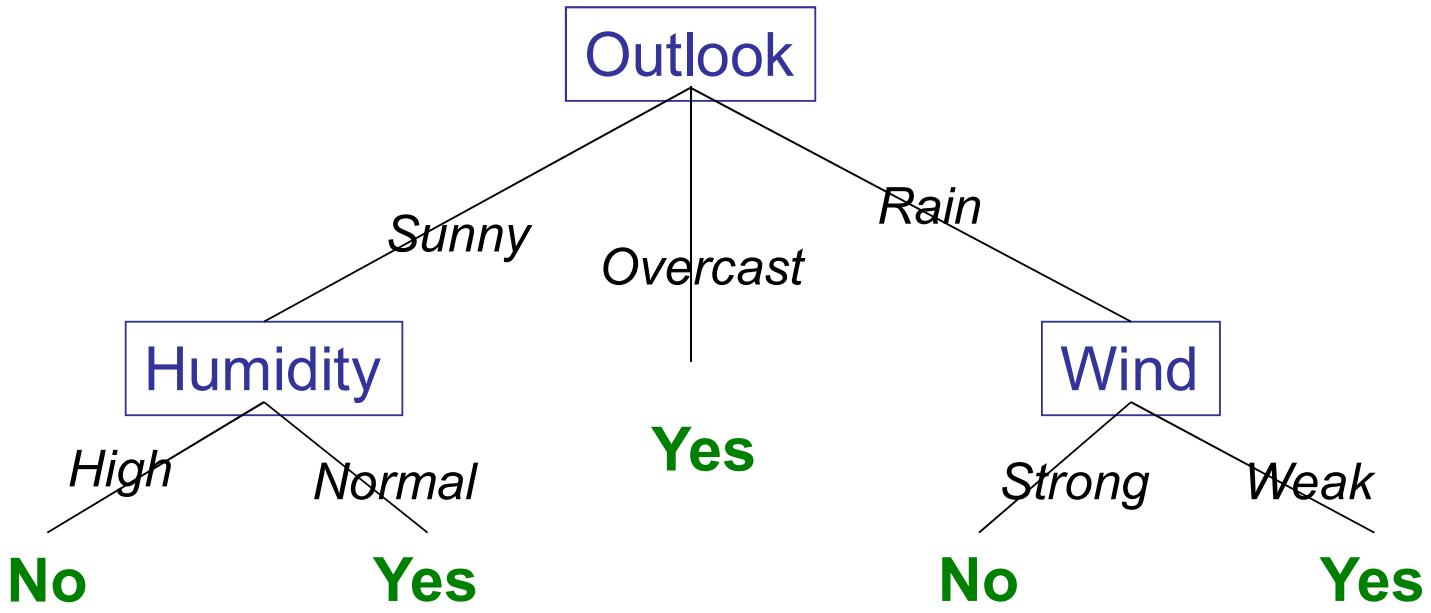
An instance

<Outlook=Sunny, Temp=Hot, Humidity=High, Wind=Strong>

Classification: No

决策树表示

决策树表示属性值约束的合取的析取



$$(\text{Outlook=Sunny} \wedge \text{Humidity=Normal}) \vee (\text{Outlook=Overcast}) \vee (\text{Outlook=Rain} \wedge \text{Wind=Weak})$$

决策树表示

- 任何布尔函数都可以写成决策树
- 通过把真值表中的每一行对应于树中的路径
- 通常可以使用小树
- 然而，有些函数需要指数大的树
- *e.g.*, 多数(majority)函数，其返回真当且仅当超过一半的输入为真

决策树学习

- 目的：找到一棵与训练样例一致的小树
- 思路：选择“最重要”属性作为(子)树的根

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
  $A \leftarrow \text{argmax}_{a \in attributes}$ IMPORTANCE(a, $examples$)
  $tree \leftarrow$ a new decision tree with root test A
  for each value v_k of A **do**
    $exs \leftarrow \{e : e \in examples$ **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 $tree$

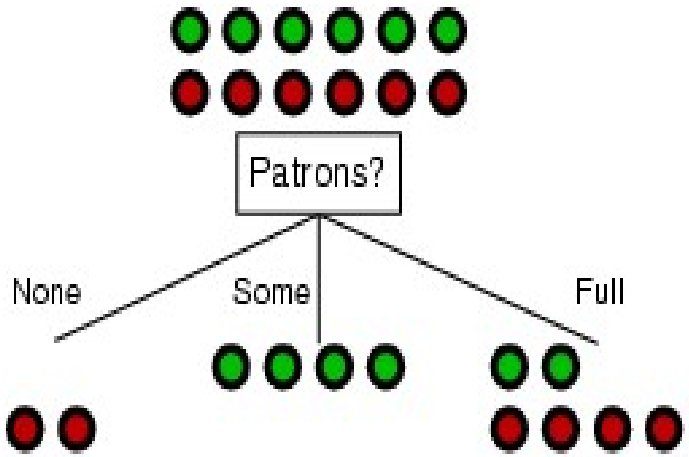
Plurality-value(examples)返回样例的多数分类

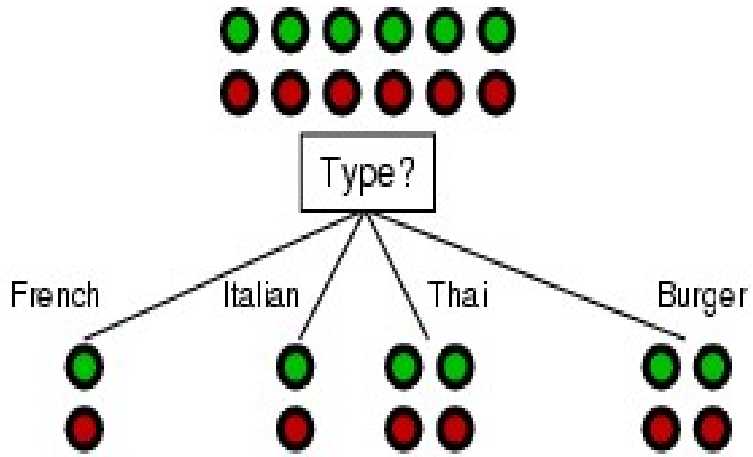
一个例子：餐馆

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

选择属性

- Idea: a good attribute splits the examples into subsets that are (ideally) "all positive" or "all negative"





- *Patrons?* is a better choice

使用信息论

- 我们将使用信息增益(information gain)的概念，它是用信息论的基本概念–熵(entropy)来定义的
- 熵是对随机变量不确定性的度量；信息的获取对应于熵的减少。
- 只有一个值的随机变量没有不确定性，其熵被定义为零。
- 抛一枚公平的硬币具有“1 bit”的熵。
- 扔一枚公平的四边形骰子有2 bits的熵，因为需要2位来描述4个同样可能的结果之一。

熵

- 以概率$P(v_k)$取值V_k的随机变量V的熵：

$$H(V) = -\sum_k P(v_k) \log_2 P(V_k)$$

- 以概率q为真的布尔随机变量的熵：

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

- 如果一个训练集包含p个正例和n个负例，那么目标属性在整个训练集上的熵为

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

信息增益

- 具有d不同值的属性A将训练集E划分为子集$E_1, \ldots, E_d$.
- 每个子集E_k具有p_k个正例和n_k个负例，
- 因此，测试属性A后剩余的期望熵为

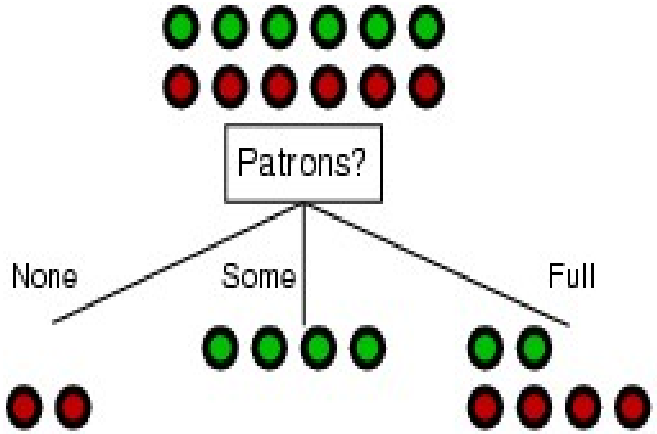
$$Remainder(A) = \sum_{k=1}^{d} \frac{p_k + n_k}{p + n} B(\frac{p_k}{p_k + n_k}).$$

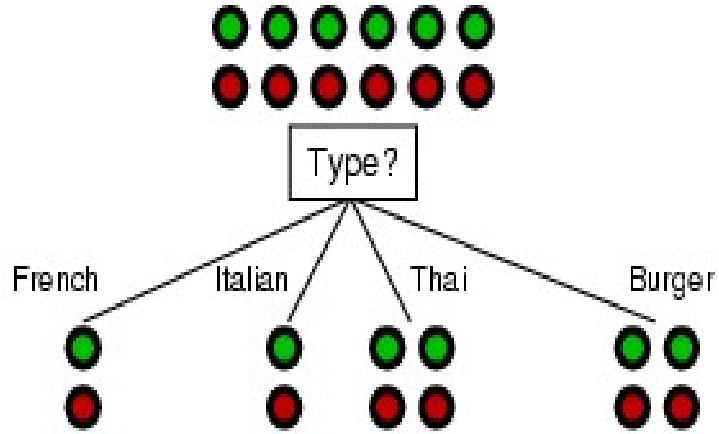
- 对A的属性测试的信息增益（IG）是熵的期望减少：

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

- 选择IG最大的属性

一个示例

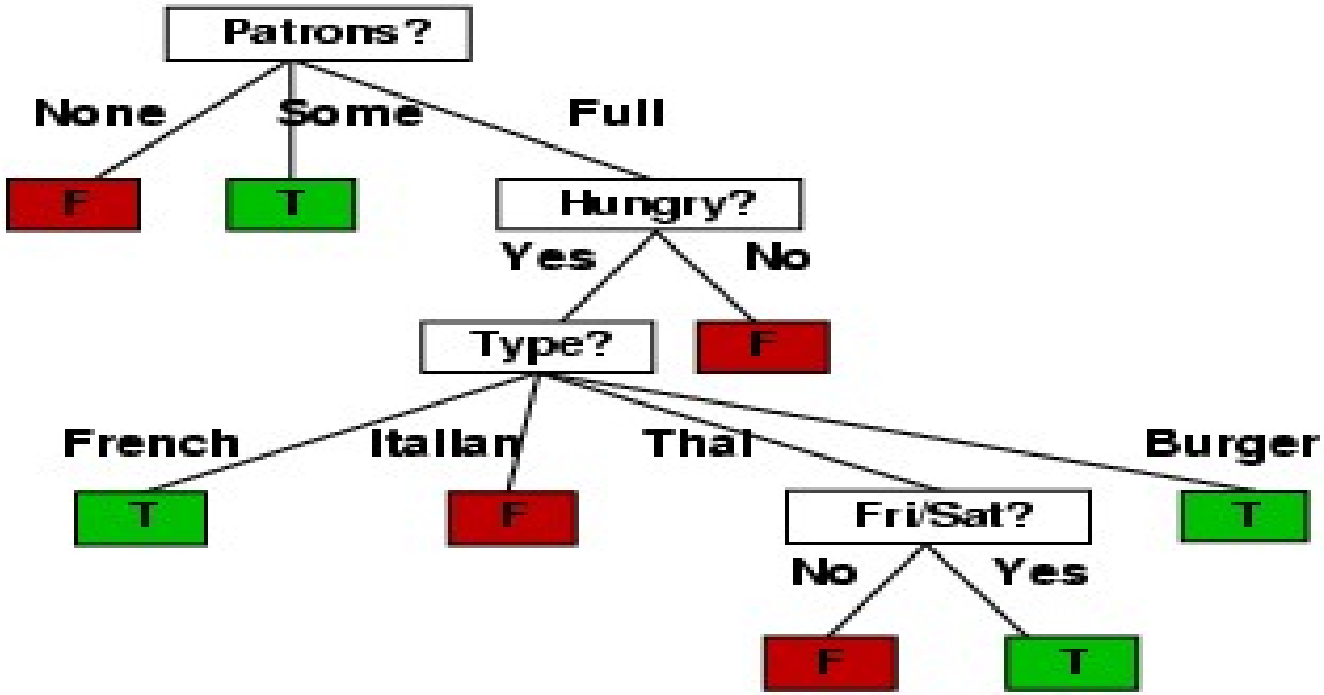




- 对于训练集, $p = n = 6$, $B(6/12) = 1$
- $Gain(Pat) = 1 - [\frac{2}{12}B(\frac{0}{2}) + \frac{4}{12}B(\frac{4}{4}) + \frac{6}{12}B(\frac{2}{6})] \approx 0.541$
- $Gain(Type) = 1 - [\frac{2}{12}B(\frac{1}{2}) + \frac{2}{12}B(\frac{1}{2}) + \frac{4}{12}B(\frac{2}{4}) + \frac{4}{12}B(\frac{2}{4})] = 0$
- 所以Pat是一个更好的分裂属性。
- 事实上，Pat具有任何属性中的最大增益，并将被DTL算法选择为根。

一个示例

Decision tree learned from the 12 examples:



Gini指数

- 熵模型拥有大量耗时的对数运算，基尼指数在简化模型的同时还保留了熵模型的优点。
- 如果一个数据集D包含来自n个类的样本，那么基尼指数定义如下：

$$Gini(D) = \sum_{i=1}^{n} p_i(1-p_i) = 1 - \sum_{i=1}^{n} p_i^2,$$

其中p_i是类i在D中的相对频率。

- 基尼指数反映了从数据集中随机抽取两个样例，其类别标记不一致的概率。
- 因此基尼指数越小，则数据集纯度越高。
- 如果$n = 2$, $Gini(D) = 2p(1-p)$

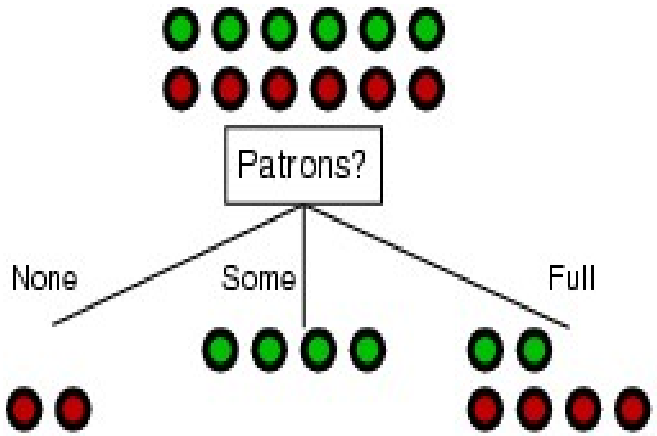
基于Gini指数的属性选择

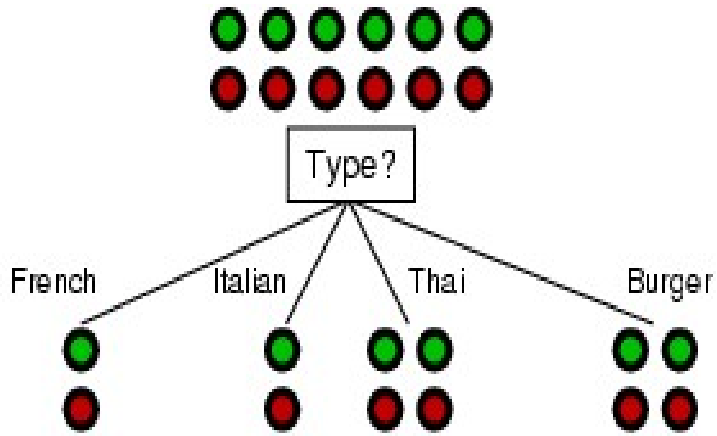
- 假设属性A的取值将D划分为$D_1, \ldots, D_m$

$$Gini(D|A) = \sum_{i=1}^{m} \frac{|D_i|}{|D|} Gini(D_i)$$

- 具有最小$Gini(D|A)$的属性被选为分裂节点的属性

一个示例





- $Gini(D|Pat) = \frac{6}{12} \cdot 2 \cdot \frac{1}{3} \cdot \frac{2}{3} = \frac{2}{9}$
- $Gini(D|Type) = (2 \cdot \frac{2}{12} + 2 \cdot \frac{4}{12}) \cdot 2 \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{2}$
- 所以Pat是一个更好的分裂属性。

学习算法的性能

- 如果学习算法产生的假设能很好地预测未见过的例子的分类，那么它就是好的
- 使用测试集验证性能
  - 收集大量样例
  - 分为2个不相交的集合：训练集和测试集
  - 使用训练集学习假设h
  - 测量测试集中的h正确分类样例的百分比
  - 对随机选择的不同大小的训练集重复2-4步

学习曲线

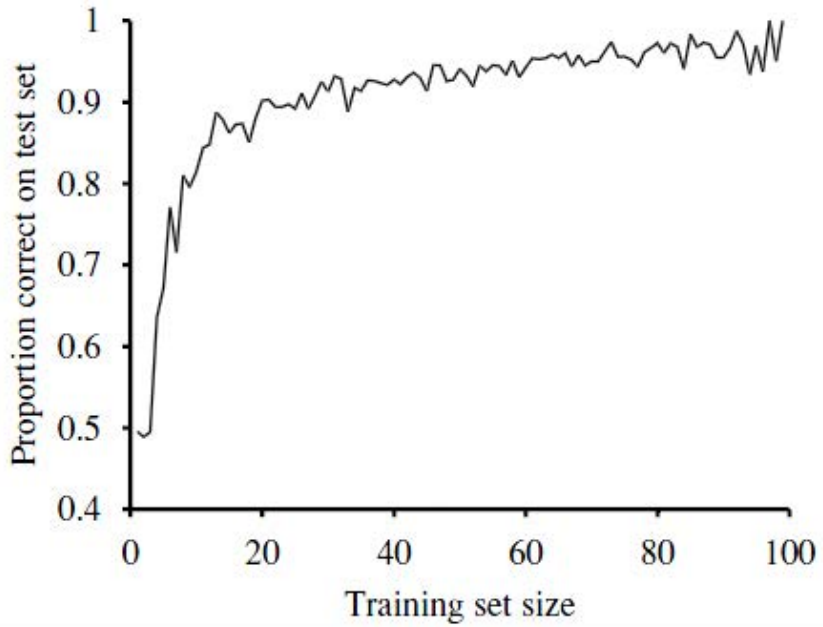


Figure 18.7 A learning curve for the decision tree learning algorithm on 100 randomly generated examples in the restaurant domain. Each data point is the average of 20 trials.

过度拟合(overfitting)

- 决策树不断增长，直到所有训练样例都被完美分类
- 但是如果
  - 数据有噪声
  - 训练集太小，无法给出目标函数的一个代表性样本
- 可能导致过度拟合！
  - 大多数学习算法的常见问题

过度拟合(Overfitting)

- 定义：给定假设空间H，假设$h \in H$过度拟合了训练数据，如果存在另一个假设$h' \in H$，使得h在训练样例上的误差小于h'，但在整个实例分布上h'的误差小于h
- 避免DTL的过度拟合：决策树剪枝：删除不明显相关的节点。

K重交叉验证(Cross-validation)

- 将数据分为两部分，一部分用于训练，另一部分用于测试假设的准确性
- 运行k次实验，每次留出$1/k$的数据进行测试
- 取k轮的平均测试集分数
- k的常用值为5和10

练习

对以下数据集执行DTL，其中D是输出

A	B	C	D
0	0	0	0
0	1	0	0
1	1	0	0
0	0	1	1
1	1	1	1
1	0	0	1
0	1	1	1

糖果示例

- 最喜欢的糖果有两种口味：樱桃(yum), 酸橙(ugh)
- 两种口味的包装相同
- 以不同比例的袋子出售：
  - 100% 樱桃
  - 75% 樱桃+ 25% 酸橙
  - 50% 樱桃+ 50% 酸橙
  - 25% 樱桃+ 75% 酸橙
  - 100% 酸橙
- 你买了一袋糖果，但不知道它的口味比例
- 吃了k个糖果后：
  - 这个袋子的口味比例是多少？
  - 下一个糖果会是什么口味？

糖果示例

- 假设H：关于世界的概率理论
  - h_1: 100% 樱桃
  - h_2: 75% 樱桃+ 25% 酸橙
  - h_3: 50% 樱桃+ 50% 酸橙
  - h_4: 25% 樱桃+ 75% 酸橙
  - h_5: 100% 酸橙
- 数据D：关于世界的证据
  - d_1: 1st candy is cherry
  - d_2: 2nd candy is lime
  - d_3: 3rd candy is lime
  - ...

贝叶斯学习

- 先验: $Pr(H)$
- 似然性: $Pr(d|H)$
- 证据: $d = \langle d_1, d_2, \ldots, d_n \rangle$
- 使用贝叶斯定理计算后验:

$$Pr(H|d) = \alpha Pr(d|H)Pr(H)$$

贝叶斯预测

- 假设我们想对未知量X进行预测(*e.g.*, 下一个糖果的口味)

$$P(X|d) = \sum_i P(X|d, h_i)P(h_i|d) = \sum_i P(X|h_i)P(h_i|d)$$

- 预测是单个假设预测的加权平均值
- 假设是原始数据和预测之间的“中介”

糖果示例

- 假设H:
  - h_1: 100% 樱桃
  - h_2: 75% 樱桃+ 25% 酸橙
  - h_3: 50% 樱桃+ 50% 酸橙
  - h_4: 25% 樱桃+ 75% 酸橙
  - h_5: 100% 酸橙
- 假设先验$P(H) = \langle 0.1, 0.2, 0.4, 0.2, 0.1 \rangle$
- 假设糖果是i.i.d.(独立同分布，identically and independently distributed), *i.e.*, $P(d|h) = \Pi_j P(d_j|h)$
- 假设前10个糖果都有酸橙味:
  - $P(d|h_5) = 1^{10} = 1$,
  - $P(d|h_3) = 0.5^{10} = 0.00097$
  - $P(d|h_1) = 0^{10} = 0$

贝叶斯预测示例

Let $d_1 = \langle lime \rangle$, and $d_2 = \langle lime, lime \rangle$

hypothesis	h_1	h_2	h_3	h_4	h_5
$P(lime\|h_i)$	0	0.25	0.5	0.75	1
$P(h_i)$	0.1	0.2	0.4	0.2	0.1
$P(d_1\|h_i)$	0	0.25	0.5	0.75	1
$P(d_2\|h_i)$	0	0.25^2	0.5^2	0.75^2	1

- $P(d_1) = \Sigma_i P(d_1|h_i)P(h_i) = 0.5$
- $P(lime|d_1) = \frac{1}{P(d_1)}\Sigma_i P(lime|h_i)P(d_1|h_i)P(h_i) = 0.65$
- $P(d_2) = \Sigma_i P(d_2|h_i)P(h_i) = 0.325$
- $P(lime|d_2) = \frac{1}{P(d_2)}\Sigma_i P(lime|h_i)P(d_2|h_i)P(h_i) = 0.654$

使用h_5生成的数据进行贝叶斯预测

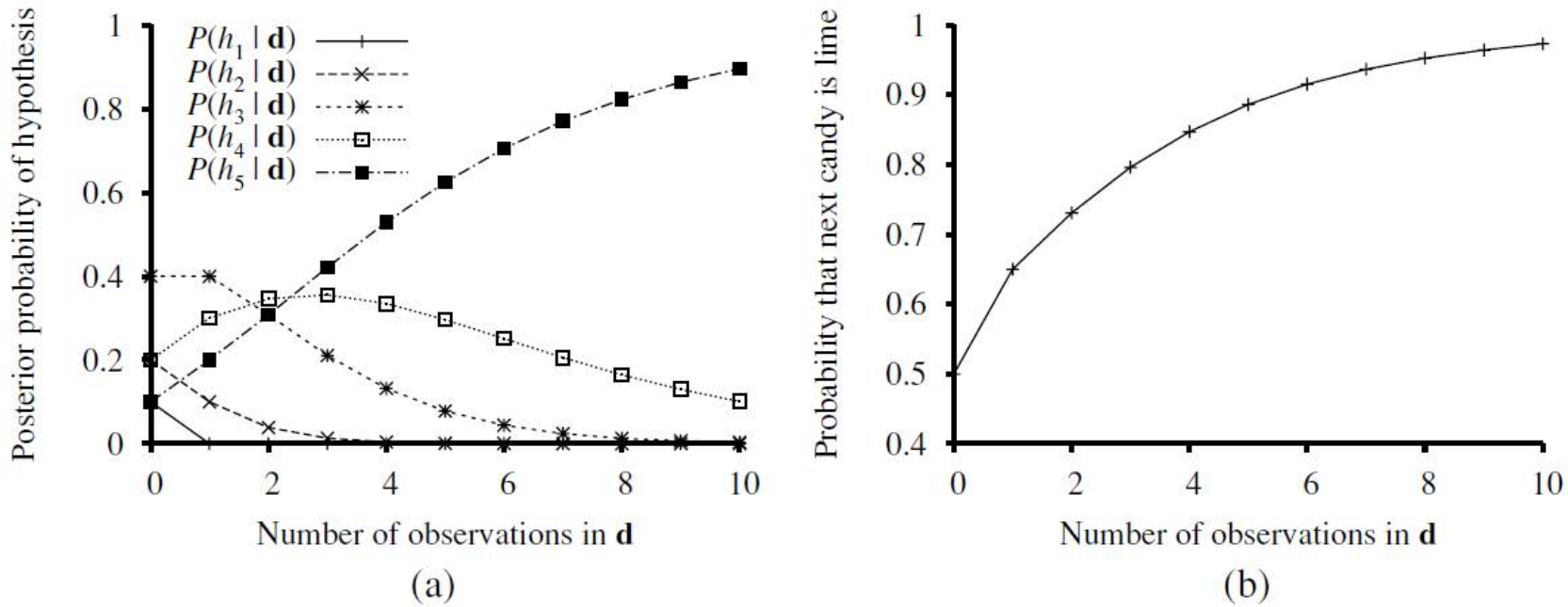


Figure 20.1 (a) Posterior probabilities $P(h_i \mid d_1, \ldots, d_N)$ from Equation (20.1). The number of observations N ranges from 1 to 10, and each observation is of a lime candy. (b) Bayesian prediction $P(d_{N+1} = lime \mid d_1, \ldots, d_N)$ from Equation (20.2).

贝叶斯学习性质

- 最优性（即，在给定先验的情况下，没有其他预测比贝叶斯预测更正确）
- 无过度拟合（所有假设均经过加权和考虑）
- 缺点：
  - 当假设空间很大时，贝叶斯学习可能是难处理的
  - *i.e.*, 假设上的和（或积分）通常是难处理的
- 解决方案：近似贝叶斯学习

极大后验(Maximum a posteriori, MAP)

- 思想: 根据最可能的假设进行预测
  - $h_{\mathsf{MAP}} = \mathsf{argmax}_{h_i} P(h_i|d)$
  - $P(X|d) \approx P(X|h_{\mathsf{MAP}})$
- 而贝叶斯学习基于所有假设进行预测，并根据它们的概率进行加权

糖果示例(MAP)

- Prediction after
  - 1 个酸橙: $h_{\text{MAP}} = h_3$, $Pr(lime|h_{\text{MAP}}) = 0.5$
  - 2 个酸橙: $h_{\text{MAP}} = h_4$, $Pr(lime|h_{\text{MAP}}) = 0.75$
  - 3 个酸橙: $h_{\text{MAP}} = h_5$, $Pr(lime|h_{\text{MAP}}) = 1$
  - 4 个酸橙: $h_{\text{MAP}} = h_5$, $Pr(lime|h_{\text{MAP}}) = 1$
  - ...
- 仅3个酸橙后，它就正确地选择了h_5
- 但如果正确的假设是h_4呢？
- 3个酸橙后，MAP错误地预测h_5
  - MAP 得到$P(lime|h_{MAP}) = 1$
  - 贝叶斯学习得到$P(lime|d) = 0.8$

MAP 性质

- MAP预测不如贝叶斯预测准确，因为它只依赖于一个假设h_{MAP}
- 但随着数据的增加，MAP预测收敛到贝叶斯预测
- 受控过拟合（先验可用于惩罚复杂假设）
- 缺点：寻找h_{MAP}可能是难处理的：
  - $h_{MAP} = \text{argmax}_h P(h|d)$
  - 优化可能很困难

MAP 计算

- 优化：
  - $h_{\text{MAP}} = \text{argmax}_h P(h|d) = \text{argmax}_h P(h)P(d|h) = \text{argmax}_h P(h)\Pi_i P(d_i|h)$
- 乘积导致非线性优化
- 取对数进行线性优化
  - $h_{\text{MAP}} = \text{argmax}_h \log P(h) + \sum_i \log P(d_i|h)$

极大似然(Maximum Likelihood, ML)

- 思想: 通过假设一致先验来简化MAP (*i.e.*, $P(h_i) = P(h_j)$ for all i, j)
  - $h_{\text{MAP}} = \text{argmax}_h P(h)P(d|h)$
  - $h_{\text{ML}} = \text{argmax}_h P(d|h)$
- 仅基于h_{ML} 进行预测:
  - $P(X|d) \approx P(X|h_{\text{ML}})$

ML 性质

- ML预测没有贝叶斯和MAP预测准确，因为它忽略了先验信息并且只依赖于一个假设h_{ML}
- 但是随着数据的增加，ML和MAP预测都会收敛到贝叶斯预测
- 容易过拟合(没有先验惩罚可能利用统计上不重要的数据模式的复杂假设)
- 但寻找h_{ML} 通常比寻找h_{MAP} 容易
  - $h_{\mathsf{ML}} = \mathsf{argmax}_h \sum_i \log P(d_i|h)$

统计学习: 总结

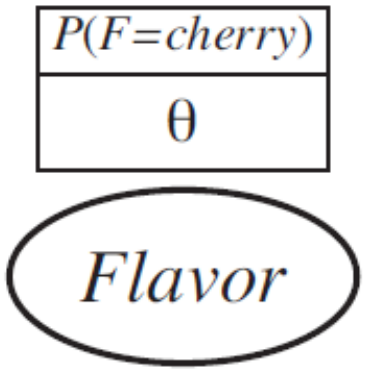
- 给定: 先验$Pr(H)$, 证据$d = \langle d_1, d_2, \ldots, d_n \rangle$, 和似然性$Pr(d|H)$, 对$P(X|d)$ 做预测
- i.i.d. 假设: $P(d|h) = \Pi_j P(d_j|h)$
- 贝叶斯学习: $P(X|d) = \sum_i P(X|h_i)P(h_i|d)$
- MAP学习: $P(X|d) \approx P(X|h_{\mathsf{MAP}})$, where $h_{\mathsf{MAP}} = \mathsf{argmax}_{h_i} P(h_i|d) = \mathsf{argmax}_{h_i} P(h_i)P(d|h_i)$
- ML学习: $P(X|d) \approx P(X|h_{\mathsf{ML}})$, where $h_{\mathsf{ML}} = \mathsf{argmax}_{h_i} P(d|h_i)$
- ML, MAP和贝叶斯预测随着数据的增加而收敛

统计学习

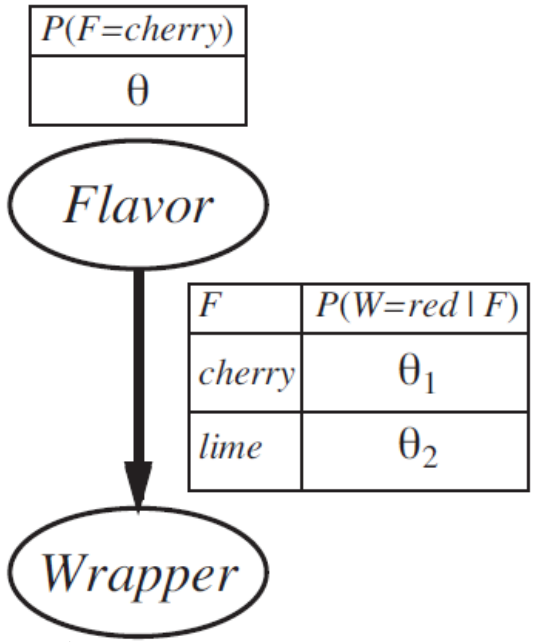
- 完全数据:
  - 当数据有多个属性时，所有属性都是已知的
  - 容易的
- 不完全数据:
  - 当数据具有多个属性时，有些属性是未知的
  - 困难

简单的ML示例

- 假设h_θ
  - $P(cherry) = \theta$ 和$P(lime) = 1 - \theta$
- 数据d:
  - c 个樱桃和l 个酸橙
- $P(d|h_\theta) = \theta^c(1-\theta)^l$
- $\log P(d|h_\theta) = c\log\theta + l\log(1-\theta)$
- $d(logP(d|h_\theta))/d\theta = c/\theta - l/(1-\theta)$
- $c/\theta - l/(1-\theta) = 0 \Rightarrow \theta = c/(c+l)$



更复杂的ML示例



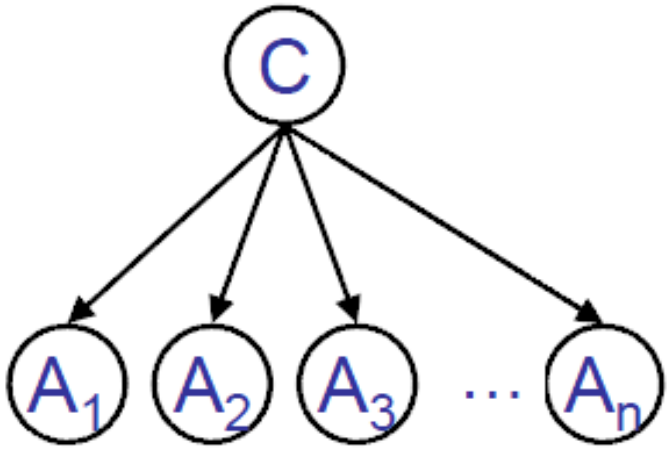
- 假设$h_{\theta,\theta_1,\theta_2}$
- 数据d:
  - c 个樱桃: g_c 个绿的和r_c 个红的
  - l 个酸橙: g_l 个绿的和r_l 个红的
- $P(d|h_{\theta,\theta_1,\theta_2}) = \theta^c(1-\theta)^l\theta_1^{r_c}(1-\theta_1)^{g_c}\theta_2^{r_l}(1-\theta_2)^{g_l}$
- $c/\theta - l/(1-\theta) = 0 \Rightarrow \theta = c/(c+l)$
- $r_c/\theta_1 - g_c/(1-\theta_1) = 0 \Rightarrow \theta_1 = r_c/(r_c+g_c)$
- $r_l/\theta_2 - g_l/(1-\theta_2) = 0 \Rightarrow \theta_2 = r_l/(r_l+g_l)$

Laplace平滑

- 当没有特定结果的样本时，就会发生过拟合的一种重要情况
  - *e.g.*, 到目前为止还没有吃到樱桃
  - $P(cherry) = \theta = c/(c+l) = 0$
  - 零概率是危险的:它们排除了结果
- Solution: Laplace(加1)平滑
  - 所有计数加1
  - $P(cherry) = \theta = (c+1)/(c+l+2) > 0$
  - 在实践中效果更好

朴素贝叶斯模型

- 要基于属性$A_1, \ldots, A_n$预测类C
- 参数:
  - $\theta = P(C = true)$
  - $\theta_{i1} = P(A_i = true | C = true)$
  - $\theta_{i2} = P(A_i = true | C = false)$
- 假设:给定C下A_i's是独立的



一个示例：餐馆

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

朴素贝叶斯学习

- 符号: $p = \#(c), n = \#(-c), p_i^+ = \#(c, a_i)$, $n_i^+ = \#(c, -a_i), p_i^- = \#(-c, a_i), n_i^- = \#(-c, -a_i)$
- $P(d|h) = \theta^p (1-\theta)^n \Pi_i \theta_{i1}^{p_i^+} \theta_{i2}^{p_i^-} (1-\theta_{i1})^{n_i^+} (1-\theta_{i2})^{n_i^-}$
- $\theta = p/(p+n)$, $\theta_{i1} = p_i^+/(p_i^+ + n_i^+)$, $\theta_{i2} = p_i^-/(p_i^- + n_i^-)$,
- $P(C|a_1, \ldots, a_n) = \alpha P(C) \Pi_i P(a_i|C)$
- 选择最可能的类别

朴素贝叶斯与决策树的对比

没有决策树准确，因为真实假设是一个决策树，
无法用朴素贝叶斯模型精确表示。

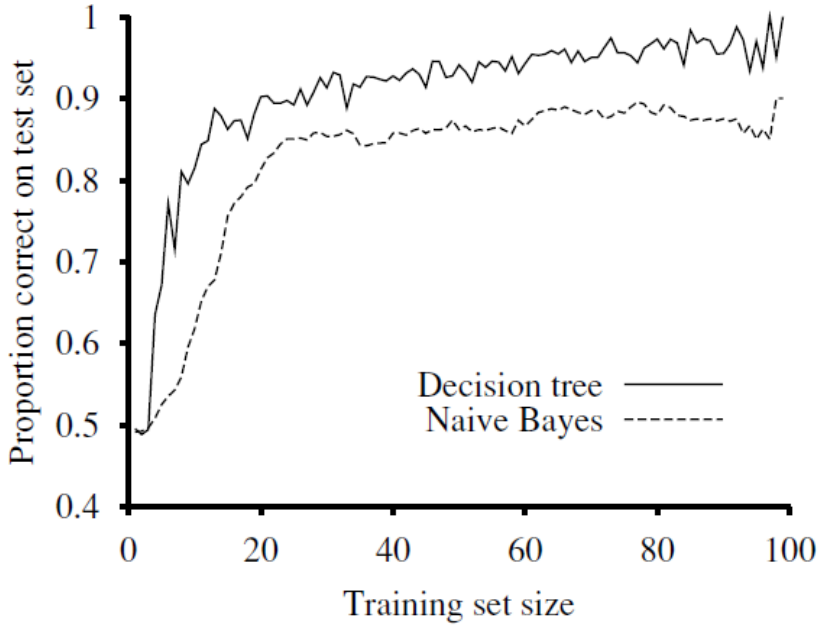


Figure 20.3 The learning curve for naive Bayes learning applied to the restaurant problem from Chapter 18; the learning curve for decision-tree learning is shown for comparison.

贝叶斯网络参数学习(ML)

- Parameters $\theta_{V,pa(V)=v}$:
  - CPTs: $\theta_{V,pa(V)=v} = P(V|pa(V)=v)$
- Data **d**:
  - d_1 : $<V_1=v_{1,1}, V_2=v_{2,1}, \ldots, V_n = v_{n,1}>$
  - d_2 : $<V_1=v_{1,2}, V_2=v_{2,2}, \ldots, V_n = v_{n,2}>$
  - ...
- Maximum likelihood:
  - Set $\theta_{V,pa(V)=v}$ to the relative frequencies of the values of V given the values **v** of the parents of V

$$\theta_{V,pa(V)=v} = \#(V,pa(V)=v) / \#(pa(V)=v)$$

Exercise

对一个新的输入$A = 0, B = 0, C = 1$，朴素贝叶斯分类器将会怎样预测D?

A	B	C	D
0	0	0	0
0	1	0	0
1	1	1	0
1	0	1	1
1	1	0	1
1	0	0	1
0	1	1	1