

---

# 第8章 分类：基本概念


丁钰

[yuding@live.com](mailto:yuding@live.com)

南京农业大学人工智能学院

# 第8章：分类：基本概念

---

- 基本概念 
- 决策树归纳
- 贝叶斯分类方法
- 基于规则的分类
- 模型评估与选择
- 提高分类准确率的技术
- 小结

# 有监督与无监督学习

---

## ■ 有监督学习 (分类)

- 监督：训练数据（观察，测量等）都带有标签，指示观察的类别
- 根据训练集分类新数据

## ■ 无监督学习 (聚类)

- 训练集的类别（标签）未知
- 给定一个观察，测量等的集合，目标是建立数据中存在的数据的类或簇

# 预测问题： 分类与数值预测

---

## ■ 分类

- 预测分类的**类标签**（离散 or 名义）
- 基于训练数据和类标签 构造一个模型，并分类新数据

## ■ 数值预测

- 建连续值函数/模型，预测未知/缺失值 —— 回归分析

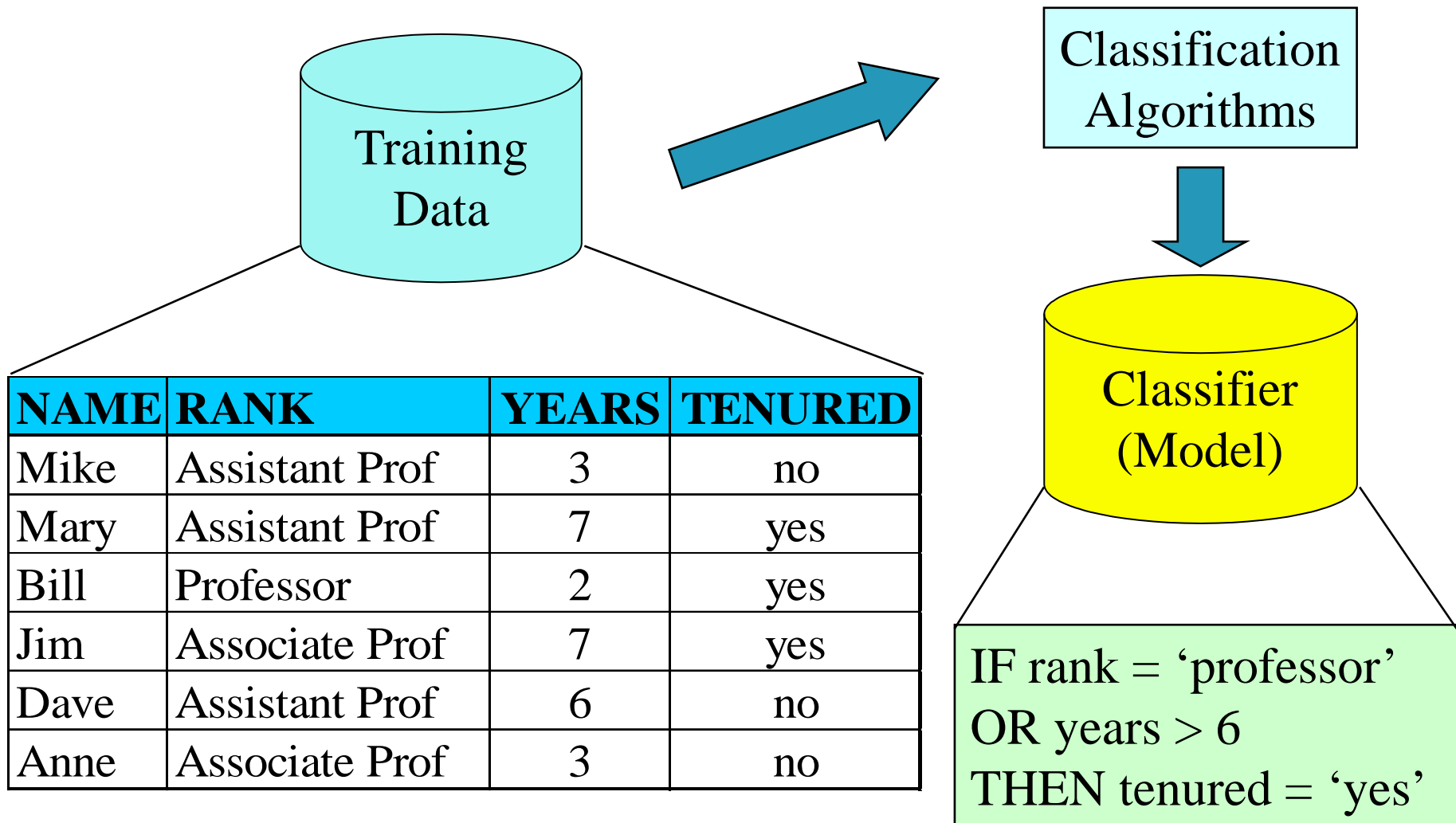
## ■ 典型应用

- 信用卡/贷款审批：信用是否良好？
- 医疗诊断：肿瘤是癌或良性？
- 欺诈检测：交易欺诈？
- 网页分类：这是哪一类？

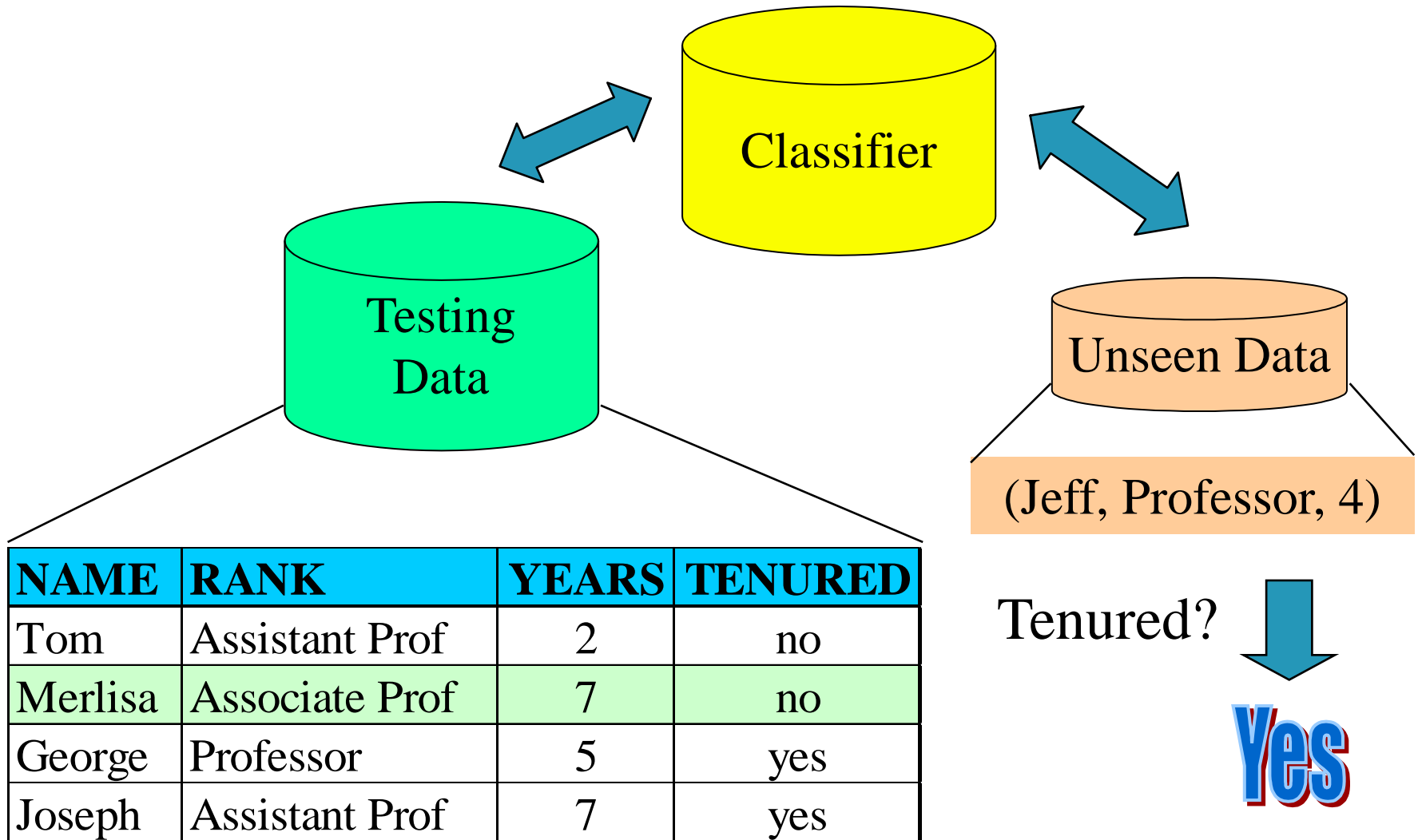
# 分类: 一个两步的过程

- **模型构建:**描述一组预先定义类
  - 假定每个元组/样本 属于一个类, 由类标号属性设定
  - 用于构建模型的元组集合称为训练元组
  - 模型可以表示为分类规则, 决策树, 数学公式
- **模型使用:** 分类将来/未知对象
  - **估计模型的准确率**
    - **测试集:** 独立于训练集的样本 (避免过分拟合 **overfitting**)
    - 比较测试样本的已知标号/由模型预测 (得到) 标号
    - **准确率:** 测试样本集中模型正确预测/分类的样本的比
  - 如果准确率合适, 使用模型来分类标号为未知的样本

# 模型构建




# 利用模型进行预测



# 第8章：分类：基本概念

---

- 基本概念
- 决策树归纳 
- 贝叶斯分类方法
- 基于规则的分类
- 模型评估与选择
- 提高分类准确率的技术
- 小结



# 决策树归纳: 例子

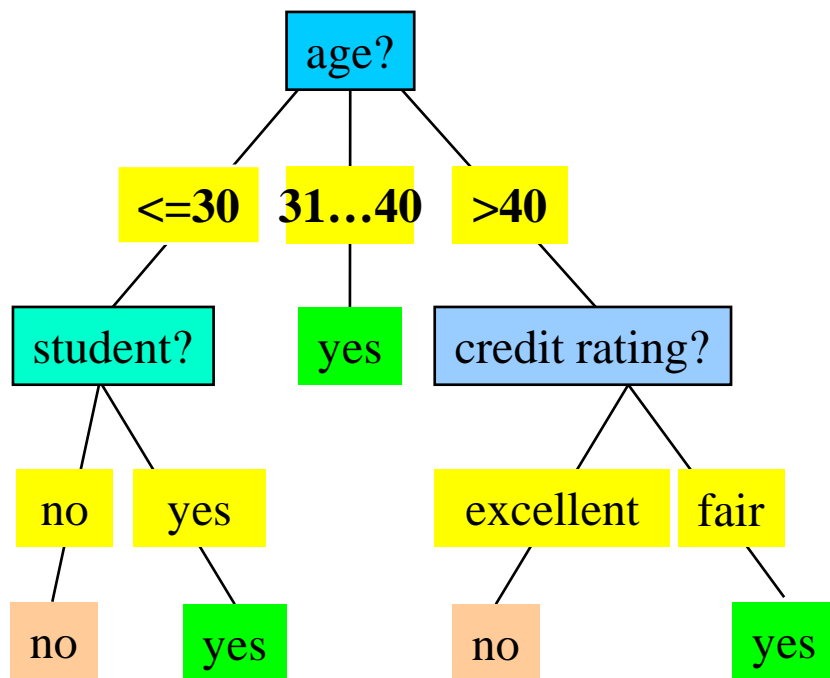
- 决策树: 类似于流程图的树结构
- 训练集: 购买计算机

age	income	student	信誉	购买计算机
<=30	high	no	fair	no
<=30	high	no	excellent	no
31...40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31...40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31...40	medium	no	excellent	yes
31...40	high	yes	fair	yes
>40	medium	no	excellent	no

**内部节点:** 一个属性上的测试

**分枝:** 测试的一个输出 (样本子集)

**叶节点:** 类标签 (用数据子集代表)



# 决策树归纳的算法

- 基本算法 (贪心算法): 根据每个属性, 递归的对样本集进行划分
  - 树构建: 自顶向下递归地分治方式
  - 开始, 所有的训练样本位于根节点
    - 属性是分类属性(若是连续值,事先离散化)
  - 基于选择的属性, 样本被递归地分割 (分割为子集)
  - 基于启发式/统计测来选择测试属性 (例如 信息增益)
- 终止划分的条件
  - 一个给定节点的所有样本属于一个类别
  - 没有属性剩下 (运用多数投票来标记此节点)
  - 没有样本剩下

输出：一棵决策树。

□ P215, 图8.3

方法：

- (1) 创建一个节点 $N$ ;
- (2) **if**  $D$ 中的元组都是同一类 $C$  **then**
- (3)     返回 $N$ 作为叶节点, 以类 $C$ 标记;
- (4) **if**  $attribute\_list$  为空 **then**
- (5)     返回 $N$ 作为叶节点, 标记为 $D$ 中的多数类;     //多数表决
- (6) 使用 $attribute\_selection\_method(D, attribute\_list)$ , 找出“最好”的 $splitting\_criterion$ ;
- (7) 用 $splitting\_criterion$ 标记节点 $N$ ;
- (8) **if**  $splitting\_attribute$ 是离散值的并且允许多路划分 **then**     //不限于二叉树
- (9)      $attribute\_list \leftarrow attribute\_list - splitting\_attribute$ ;     //删除划分属性
- (10) **for**  $splitting\_criterion$ 的每个输出 $j$      // 划分元组并对每个划分产生子树
- (11)     设 $D_j$ 是 $D$ 中满足输出 $j$ 的数据元组的集合;     //一个划分
- (12)     **if**  $D_j$ 为空 **then**
- (13)         加一个树叶到节点 $N$ , 标记为 $D$ 中的多数类;
- (14)     **else** 加一个由  $Generate\_decision\_tree(D_j, attribute\_list)$ 返回的节点到节点 $N$ ;
- end for**
- (15) 返回 $N$ ;

# 属性选择度量

## ■ 属性选择度量

- 分裂规则，决定给定节点上的元组如何分裂
- 具有最好度量得分的属性选定位分裂属性

## ■ 三种度量

- 信息增益、增益率、Gini指标

## ■ 数学符号

- $D$ 为元组的训练集，元组属于 $m$ 个不同的类 $C_i(i=1\dots m)$
- $C_{i,D}$ 是 $D$ 中的 $C_i$ 类的元组集合
- $|C_{i,D}|$ 和 $|D|$ 分别表示各自的元组个数

# 属性选择度量: 信息增益(ID3/C4.5)

- 选择具有最高信息增益的属性
- 令  $p_i$  为  $D$  中的任一元组属于类  $C_i$  概率, 估计为  $|C_{i,D}|/|D|$
- 分类  $D$  中元组需要的期望信息(entropy): 信息熵

$$Info(D) = -\sum_{i=1}^m p_i \log_2(p_i)$$

- (利用  $A$  分裂  $D$  为  $v$  个部分后)分类  $D$  需要的信息为: 条件熵

$$Info_A(D) = \sum_{j=1}^v \frac{|D_j|}{|D|} \times Info(D_j)$$

- 以属性  $A$  分枝得到的信息增益, 在一个条件下, 信息复杂度减少的程度

$$Gain(A) = Info(D) - Info_A(D)$$

# 属性选择: 信息增益

■ **Class P:** 买电脑 = “yes”

■ **Class N:** 买电脑 = “no”

age	income	student	credit_ratings	computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31...40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31...40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31...40	medium	no	excellent	yes
31...40	high	yes	fair	yes
>40	medium	no	excellent	no

$$Info(D) = -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) = 0.940$$

$$\begin{aligned}
 Info_{age}(D) &= \frac{5}{14} \times \left( -\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5} \right) \\
 &\quad + \frac{4}{14} \times \left( -\frac{4}{4}\log_2\frac{4}{4} - \frac{0}{4}\log_2\frac{0}{4} \right) \\
 &\quad + \frac{5}{14} \times \left( -\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5} \right) \\
 &= 0.694 \text{ bits.}
 \end{aligned}$$

$$Gain(age) = Info(D) - Info_{age}(D) = 0.246$$

$$Gain(income) = 0.029$$

$$Gain(student) = 0.151$$

$$Gain(credit\_rating) = 0.048$$

# 计算信息增益-连续值属性

- 令  $A$  为连续属性
- 必须为  $A$  确定一个最佳分裂点 *best split point*
  - 上升序排序  $A$
  - 典型地, 每对相邻值的中点是一个可能的分裂点
    - $(a_i + a_{i+1})/2$  是  $a_i$  与  $a_{i+1}$  的中点
  - 具有最小期望信息需求的点选为  $A$  的分裂点
- Split:
  - $D1$  为  $D$  中元组满足  $A \leq \text{split-point}$ ,  $D2$  是元组满足  $A > \text{split-point}$

# 增益率 (C4.5)

- 信息增益倾向于有大量不同取值的属性（划分更细，更纯）
  - 极端：每个划分子集只有一个样本，即一个类
  - 此时 $\text{Info}(d)=0$
- C4.5 (ID3 后继) 使用增益率来克服这一问题(规范化信息增益)

$$\text{SplitInfo}_A(D) = -\sum_{j=1}^v \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

- $\text{GainRatio}(A) = \text{Gain}(A)/\text{SplitInfo}(A)$
- Ex.  $\text{SplitInfo}_{\text{income}}(D) = -\frac{4}{14} \times \log_2\left(\frac{4}{14}\right) - \frac{6}{14} \times \log_2\left(\frac{6}{14}\right) - \frac{4}{14} \times \log_2\left(\frac{4}{14}\right) = 1.557$ 
  - $\text{gain\_ratio}(\text{income}) = 0.029/1.557 = 0.019$
- 具有最大增益率的属性选为分裂属性



# 基尼指数Gini Index

- 数据  $D$  包含  $n$  类别的样本, gini指标,  $gini(D)$  定义为:

$$gini(D) = 1 - \sum_{j=1}^n p_j^2$$

$p_j$  为类别  $j$  在  $D$  中的频率

- 数据集  $D$  基于属性  $A$  分裂为子集  $D_1$  和  $D_2$ , gini 指标定义为

$$gini_A(D) = \frac{|D_1|}{|D|} gini(D_1) + \frac{|D_2|}{|D|} gini(D_2)$$

- 不纯度减少:  $\Delta gini(A) = gini(D) - gini_A(D)$
- 具有最小  $gini_{split}(D)$  的属性(or不纯度减少最大的) 用于分裂节点 (需要枚举所有可能的分裂情况)

# 计算Gini Index指标

- D 有 9个元组买电脑 = “yes” / 5 个买电脑 = “no”

$$gini(D) = 1 - \left(\frac{9}{14}\right)^2 - \left(\frac{5}{14}\right)^2 = 0.459$$

- 设属性income分裂D为包含10个元组的D<sub>1</sub>: {low, medium} / 4个元组的D<sub>2</sub>

$$\begin{aligned} Gini_{income \in \{low, medium\}}(D) &= \frac{10}{14}Gini(D_1) + \frac{4}{14}Gini(D_2) \\ &= \frac{10}{14} \left( 1 - \left(\frac{7}{10}\right)^2 - \left(\frac{3}{10}\right)^2 \right) + \frac{4}{14} \left( 1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2 \right) \\ &= 0.443 \\ &= Gini_{income \in \{high\}}(D). \end{aligned}$$

**Gini<sub>{low, high}</sub>=0.458; Gini<sub>{medium, high}</sub>=0.450.** 因此{low, medium}/{high}分裂, 由于其有最小的Gini index

- 假设所有属性都是连续值, 需要其他技术, e.g., 聚类, 来获得可能的分裂点

# 比较属性选择度量

- 通常三种度量获得较好的结果
  - 信息增益**Information gain**:
    - 偏向于多值属性
  - 增益率**Gain ratio**:
    - 倾向于不平衡的分裂，其中一个子集比其他小得多
  - **Gini index**:
    - 偏向于多值属性
    - 当类数目较大时，计算困难
    - 倾向于导致大小相等的分区和纯度

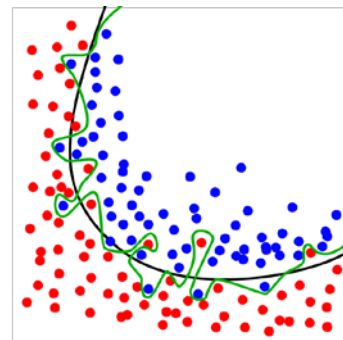
# 其他属性选择度量

- **CHAID**: 一种流行的决策树算法, 基于独立  $\chi^2$  检验的选择度量
- **C-SEP**: 某些情况下比信息增益gini指标更好
- **G-statistic**: 非常近似于  $\chi^2$  分布
- **MDL (最小描述长度)** (i.e., 首选最简单的解):
  - 最佳树为需要最小二进位的树 (1) 编码树, (2) 编码树的异常
- 多元划分 (基于多变量组合来划分)
  - **CART**: 基于属性的线性组合来发现多元划分
- 哪一个是最好的?
  - 大部分可以获得较好结果, 没有一个显著地优于其他

# 过拟合与数剪枝

## ■ 过拟合Overfitting: 一棵归纳的树 可能过分拟合训练数据

- 分枝太多,某些反映训练数据中的异常, 噪音/孤立点
- 对未参与训练的样本的低精度预测



## ■ 两种处理方法

### ■ 先剪枝: 提前终止树构造

- 如果对一个节点的分裂产生低于给定的阈值的度量, 划分停止
- 选择一个合适的阈值很难

### ■ 后剪枝: 从完全生长的树中剪去树枝—得到逐步修剪树

- 例如, 最小化代价复杂度 (树节点个数和错误率的函数)
- 使用不同于训练集的数据来确定哪一个 “best pruned tree”

# 决策树归纳的增强

## ■ 允许连续值属性

- 动态地定义新的离散值属性，其把连续值属性分成离散的区间

## ■ 处理缺失属性值

- 分配属性的最常见值
- 为每一个可能的值分配概率

## ■ 属性构造

- 基于现有的稀少出现的属性创建新的属性，
- 这减少了分散，重复和复制

# 大型数据库中分类

- 分类—被统计学和机器学习研究人员广泛地研究的经典问题
- 可伸缩性:以合理的速度分类由带有数百个属性的百万个样本组成的数据集
- 为什么决策树归纳受欢迎?
  - 相对快的训练速度 (与其他分类方法相比)
  - 转换为简单、易于理解的分类规则
  - 可用 SQL 查询来访问数据库
  - 与其它方法可比的分类精度
- **RainForest** (VLDB'98 — Gehrke, Ramakrishnan & Ganti)
  - Builds an AVC-list (attribute, value, class label)

# 雨林（ RainForest ） 的可扩展性框架

---

- 过去的研究提出了多种决策树算法，但是到目前为止并没有有一种算法在任何数据集合下生成决策树的质量方面超过所有其他的算法
- 雨林算法框架关注于提高决策树算法的伸缩性，该框架可运用于大多数决策树算法，使算法获得的结果与将全部的数据放置于内存所得到的结果一致，但是在运行时可以使用较少的内存。在内存一定的情况下，也可以更好的满足算法的需求。
- 生成的决策树的质量取决于具体的决策树算法，于本框架无关。



# 雨林（RainForest）的可扩展性框架

---

- 在决策树的每个节点，对于每个属性建立并维持 **AVC-list: AVC** (属性-值, 类标号)
- **AVC集(of an attribute  $X$ )**
  - 把训练样本集投影到属性 $X$ 和类标签上，给出属性 $X$ 的每个值上的类标签计数
- **AVC组群 (在节点 $n$ )**
  - 节点 $n$ 上所有预测属性的AVC集合——组群

# Rainforest: 训练集和AVC集

## Training Examples

age	income	student	credit_rating	comp
<=30	high	no	fair	no
<=30	high	no	excellent	no
31...40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31...40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31...40	medium	no	excellent	yes
31...40	high	yes	fair	yes
>40	medium	no	excellent	no

### AVC-set on *Age*

Age	Buy_Computer	
	yes	no
<=30	2	3
31..40	4	0
>40	3	2

### AVC-set on *income*

income	Buy_Computer	
	yes	no
high	2	2
medium	4	2
low	3	1

### AVC-set on *Student*

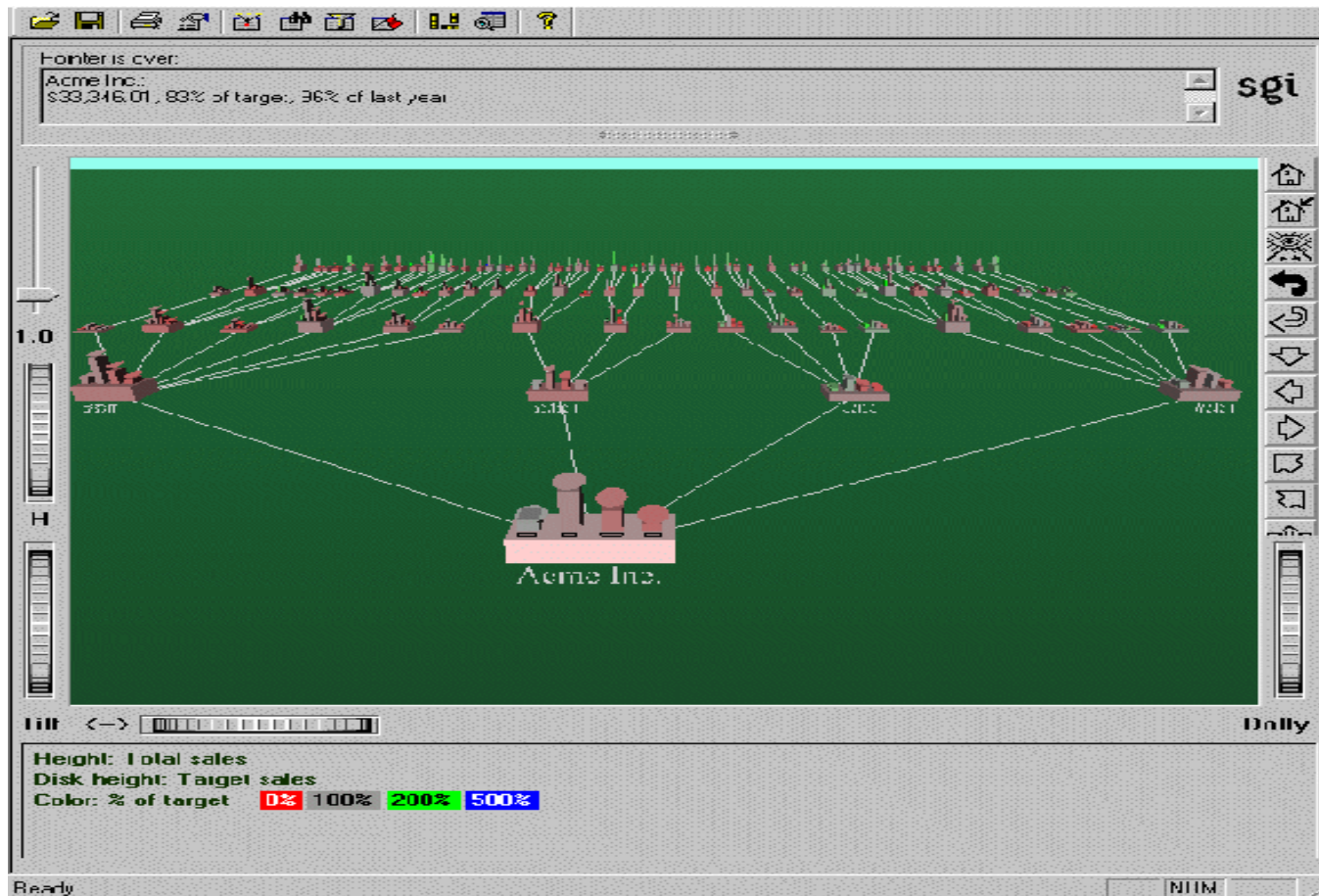
student	Buy_Computer	
	yes	no
yes	6	1
no	3	4

### AVC-set on *credit\_rating*

Credit rating	Buy_Computer	
	yes	no
fair	6	2
excellent	3	3

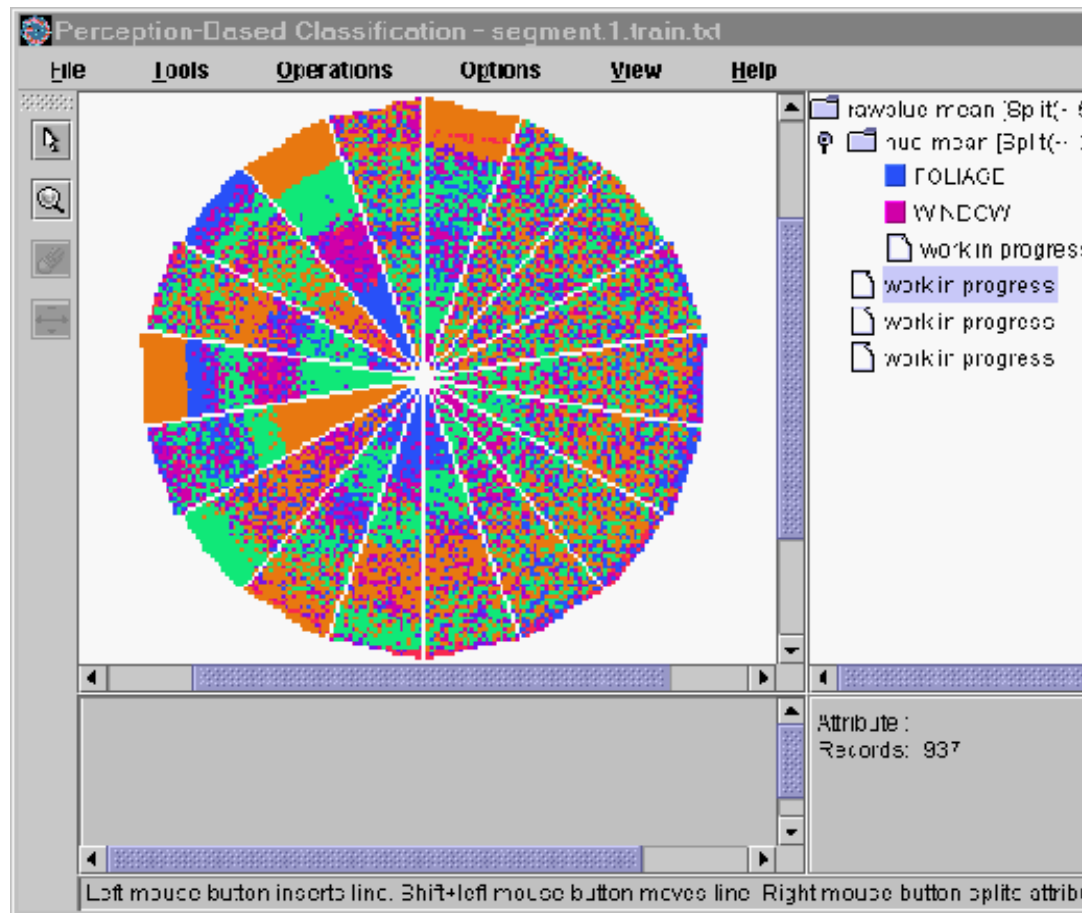
# 自助乐观算法 (BOAT : Bootstrapped Optimistic Algorithm for Tree Construction)

- 使用一个叫做 bootstrapping自助法的统计技术多个更小的样本集（子集），每一个可放入内存
- 每个子集产生一个树，导致多个树
- 考察这些树并用他们构造一个新树 $T'$ 
  - 事实证明， $T'$  非常接近于使用全部数据集构造的树
- Adv: 只要求扫描DB两遍，并且是一个增量算法




# 基于感知的分类（PBC）的交互式视觉挖掘

- ❑ 基于感知的分类器（PCB）：由慕尼黑大学开发（1999）。
- ❑ 一种颜色代表一个类别标签
- ❑ 一个饼代表一个属性（或变量）。
- ❑ 带有随机散布的馅饼意味着分类能力很弱
- ❑ 有明显分区的色条的饼意味着良好的分类能力
- ❑ 人们可以选择一个好的属性，并重新生成新的饼图，以便在随后的层次上进行分类。



# 第8章：分类：基本概念

---

- 基本概念
- 决策树归纳
- 贝叶斯分类方法 
- 基于规则的分类
- 模型评估与选择
- 提高分类准确率的技术
- 小结

# 什么是贝叶斯分类？

---

- 一个统计分类器
  - 进行概率预测（即预测类成员概率）。
- 基于贝叶斯定理的基础
- 性能
  - 一个简单的贝叶斯分类器，即朴素贝叶斯分类器，具有与决策树和经过挑选的神经网络分类器相当的性能。
- 渐进式
  - 每个训练实例都可以逐步增加/减少假设正确的概率--先前的知识可以与观察到的数据相结合
- 理论标准
  - 即使贝叶斯方法在计算上难以解决，它们也能提供一个最佳决策的标准，其他方法可以对照这个标准进行衡量。

# 贝叶斯理论

- 令 $X$  为数据元组：类标签未知
- 令 $H$ 为一个假设在： $X$ 属于类别  $C$
- 分类就是确定  $P(H|X)$ (*后验概率*)
  - 给定观察数据  $X$ ，假设 $H$ 成立的概率
- $P(H)$  (*先验概率*)——最初的概率
  - 例,不管年龄和收入等条件  $X$ 将会购买计算机
- $P(X)$ ：数据元组 $X$ 被观察到的概率
- $P(X|H)$  (可能性)
  - 假设 $H$ 成立，那么观测到样本 $X$ 的概率
  - E.g., 已知 $X$ 购买计算机， $X$  为31..40且中等收入的概率



# 贝叶斯理论 (Bayesian Theorem)

- 给定训练数据  $\mathbf{X}$ , 假设 $H$ 的后验概率  $P(H|\mathbf{X})$ 满足贝叶斯理论

$$P(H|\mathbf{X}) = \frac{P(\mathbf{X}|H)P(H)}{P(\mathbf{X})} = P(\mathbf{X}|H) \times P(H) / P(\mathbf{X})$$

$$p(\text{类别}|\text{特征}) = \frac{p(\text{特征}|\text{类别})p(\text{类别})}{p(\text{特征})}$$

- 预测 $\mathbf{X}$ 属于类别 $C_i$ 当且仅当概率 $P(C_i|\mathbf{X})$ 是所有 $P(C_k|\mathbf{X})$ for all the  $k$  classes最大的
- 实际困难: 需要许多可能性的初步知识, 计算成本显著

# 朴素贝叶斯 (Naïve Bayesian) 分类

- **D**为训练数据集（包含类别标签），并且每个元组表示为一个**n**-维的属性向量 $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$
- 假定有 **m** 个类别  $C_1, C_2, \dots, C_m$
- 分类就是推导最大的后验概率, i.e., the maximal  $P(C_i|\mathbf{X})$ , 可以由贝叶斯理论计算

$$P(C_i | \mathbf{X}) = \frac{P(\mathbf{X} | C_i)P(C_i)}{P(\mathbf{X})}$$

- 由于对所有类 $P(\mathbf{X})$ 是常量，只需要最大化

$$P(C_i | \mathbf{X}) \sim P(\mathbf{X} | C_i)P(C_i)$$

# 朴素贝叶斯分类器的推导

- 一个简单假定: 属性之间相互独立

$$P(\mathbf{X} | C_i) = \prod_{k=1}^n P(x_k | C_i) = P(x_1 | C_i) \times P(x_2 | C_i) \times \dots \times P(x_k | C_i)$$

- 优点: 极大地减少了计算代价, 只需要统计类的分布
- 若 $A_k$  是分类属性
  - $P(\mathbf{x}_k | C_i) = C_i$  类中 $A_k$ 取值为 $\mathbf{x}_k$  的元组数/ $|C_i, D|$  (类 $C_i$  的大小)
- 若 $A_k$ 是连续值
  - 假设 $\mathbf{x}_k$ 服从均值 $\mu$  标准差 $\sigma$  的高斯分布  $g(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$

$$P(\mathbf{X} | C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i})$$

# 朴素贝叶斯分类: 训练数据集

两个类别:

C1:buys\_computer = 'yes'

C2:buys\_computer = 'no'

数据样本

X = (age <=30,

Income = medium,

Student = yes

Credit\_rating = Fair)

age	income	student	credit_rating	comp
<=30	high	no	fair	no
<=30	high	no	excellent	no
31...40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31...40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31...40	medium	no	excellent	yes
31...40	high	yes	fair	yes
>40	medium	no	excellent	no

# Naïve Bayesian Classifier: 例子

- $P(C_i)$ :  $P(\text{buys\_computer} = \text{"yes"}) = 9/14 = 0.643$   
 $P(\text{buys\_computer} = \text{"no"}) = 5/14 = 0.357$
  - Compute  $P(X|C_i)$  for each class
    - $P(\text{age} = \text{"<=30"} | \text{buys\_computer} = \text{"yes"}) = 2/9 = 0.222$
    - $P(\text{age} = \text{"<= 30"} | \text{buys\_computer} = \text{"no"}) = 3/5 = 0.6$
    - $P(\text{income} = \text{"medium"} | \text{buys\_computer} = \text{"yes"}) = 4/9 = 0.444$
    - $P(\text{income} = \text{"medium"} | \text{buys\_computer} = \text{"no"}) = 2/5 = 0.4$
    - $P(\text{student} = \text{"yes"} | \text{buys\_computer} = \text{"yes"}) = 6/9 = 0.667$
    - $P(\text{student} = \text{"yes"} | \text{buys\_computer} = \text{"no"}) = 1/5 = 0.2$
    - $P(\text{credit\_rating} = \text{"fair"} | \text{buys\_computer} = \text{"yes"}) = 6/9 = 0.667$
    - $P(\text{credit\_rating} = \text{"fair"} | \text{buys\_computer} = \text{"no"}) = 2/5 = 0.4$
  - $X = (\text{age} \leq 30, \text{income} = \text{medium}, \text{student} = \text{yes}, \text{credit\_rating} = \text{fair})$ 
    - $P(X|C_i) : P(X|\text{buys\_computer} = \text{"yes"}) = 0.222 \times 0.444 \times 0.667 \times 0.667 = 0.044$   
 $P(X|\text{buys\_computer} = \text{"no"}) = 0.6 \times 0.4 \times 0.2 \times 0.4 = 0.019$
    - $P(X|C_i) * P(C_i) : P(X|\text{buys\_computer} = \text{"yes"}) * P(\text{buys\_computer} = \text{"yes"}) = 0.028$   
 $P(X|\text{buys\_computer} = \text{"no"}) * P(\text{buys\_computer} = \text{"no"}) = 0.007$
- Therefore,  $X$  belongs to class ("buys\_computer = yes")

# 避免零概率问题

- 朴素贝叶斯要求每个条件概率非零. 然而,预测的概率可能为零

$$P(X | C_i) = \prod_{k=1}^n P(x_k | C_i)$$


- Ex. 假定有1000 元组, e=low (0), income= medium (990), and income = high (10)
- Use **Laplacian correction**校准 (or Laplacian estimator估计法)
  - *Adding 1 to each case*  
Prob(income = low) = 1/1003  
Prob(income = medium) = 991/1003  
Prob(income = high) = 11/1003
  - 校准的 “corrected” 概率估计很接近未校准的

# Naïve Bayesian Classifier:评论

- 优点:
  - 分类过程容易执行
  - 在很多情况下都能获得较好的分类效果
- 缺点:
  - 类条件独立性，损失精度
  - 实际中，变量间存在依赖
    - E.g.,医院：患者：简介：年龄，家族病史等  
症状：发烧，咳嗽等疾病：肺癌，糖尿病等
    - 上述属性之间的关系无法用独立性描述
- 解决方案:
  - Bayesian Belief Networks

# 第8章：分类：基本概念

---

- 基本概念
- 决策树归纳
- 贝叶斯分类方法
- 基于规则的分类 
- 模型评估与选择
- 提高分类准确率的技术
- 小结



# 使用IF-THEN 规则分类

- 使用 **IF-THEN** 规则表示知识：
  - **R** : **IF** age = youth AND student = yes **THEN** buys\_computer = yes
  - 规则前件（前提）与规则结论（后件）
- 规则评估：覆盖率（coverage）与准确率（accuracy）
  - $n_{\text{covers}}$  = 规则**R**覆盖的元组数
  - $n_{\text{correct}}$  = **R**正确分类的元组数
  - $\text{coverage}(\mathbf{R}) = n_{\text{covers}} / |\mathbf{D}|$  **D**: 训练数据集
  - $\text{accuracy}(\mathbf{R}) = n_{\text{correct}} / n_{\text{covers}}$

# 使用IF-THEN 规则分类

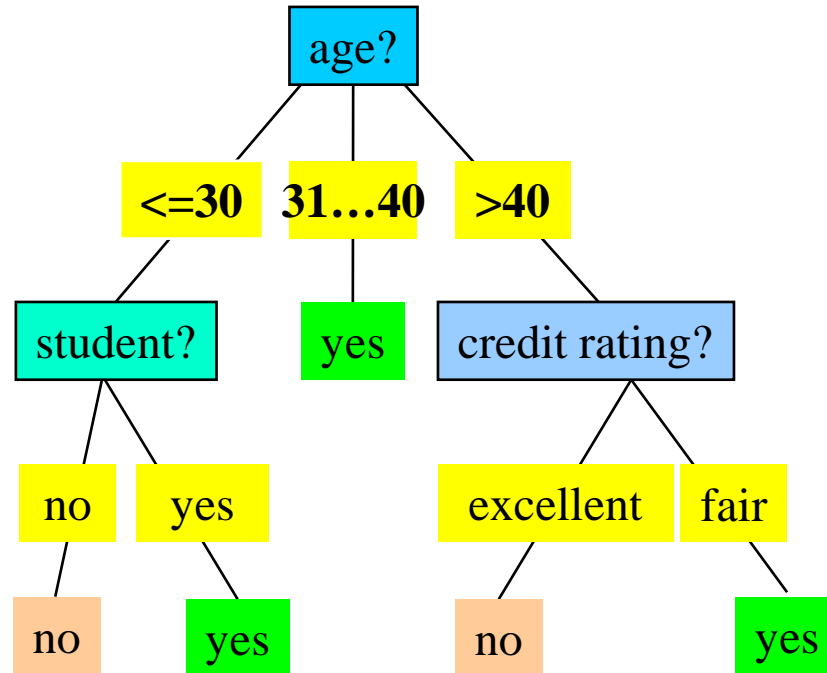
- 需解决的问题：如果超过1条规则被触发，需要解决冲突
  - 规模序Size ordering: 最高优先权赋予 “最苛刻” 的规则（即, 最多属性测试）
  - 基于类的序：每个类的错误分类代价的下降序
  - 基于规则的序（决策表）：根据一些规则的质量度量或由专家建议，规则被组织成一个长的优先级列表
- 若所有规则都不满足——设置默认规则

# 从决策树提取规则

---

- 规则比一棵大的决策树更容易理解
- 从根到每个叶子的路径产生一个规则
- 沿路径的每个属性值对一起形成了一个联合：叶节点形成规则后件
- 规则是互斥的和穷举的
  - 没有冲突规则，每个元组被覆盖

# 从决策树提取规则



- **Example: Rule extraction from our *buys\_computer* decision-tree**

IF *age* = young AND *student* = no

THEN *buys\_computer* = no

IF *age* = young AND *student* = yes

THEN *buys\_computer* = yes

IF *age* = mid-age

THEN *buys\_computer* = yes

IF *age* = old AND *credit\_rating* = excellent THEN *buys\_computer* = no

IF *age* = old AND *credit\_rating* = fair THEN *buys\_computer* = yes

# 顺序覆盖算法的规则归纳

---

- 顺序覆盖算法：
  - 直接从训练数据抽取规则，顺序学习
- 典型的算法：
  - **FOIL, AQ, CN2, RIPPER**
- 规则判断准则：
  - 类 $C_i$ 的规则尽量覆盖 $C_i$ 的元组，不或少覆盖其他类元组

# 顺序覆盖算法的规则归纳

---

## ■ 基本步骤:

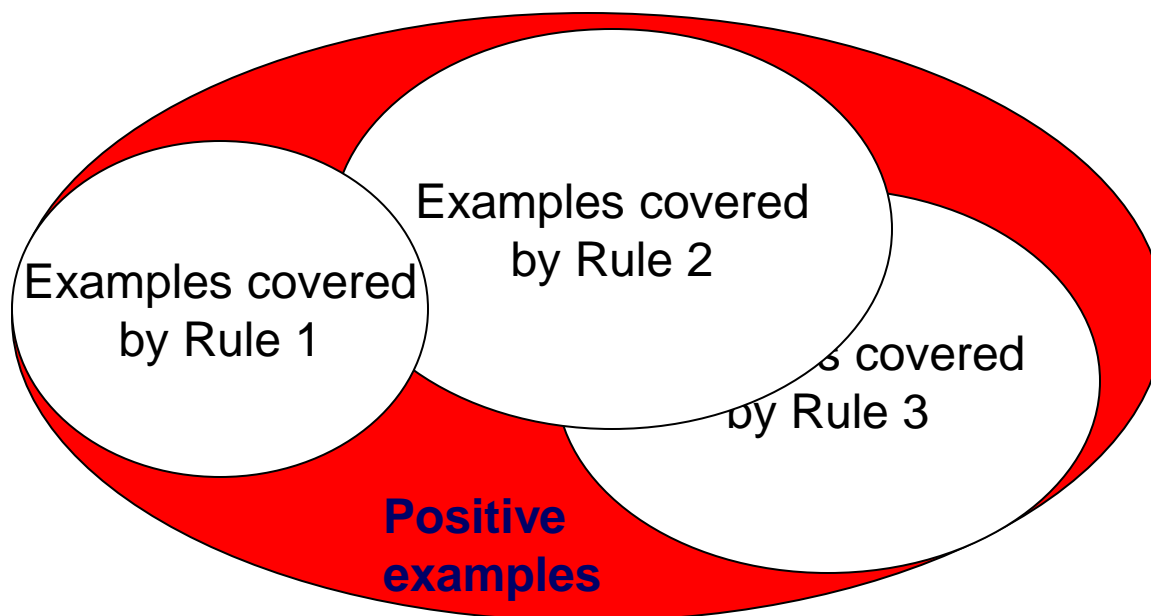
- 一次学习一个规则
- 每学习一个规则，删除此规则覆盖的元组
- 对剩下的元组重复该过程直到终止条件（e.g., 没有训练样本/返回的规则的质量低于用户给定的阈值）

## ■ 与决策树对照:

- 构建决策树，同时学习一组规则

# 顺序覆盖算法的规则归纳

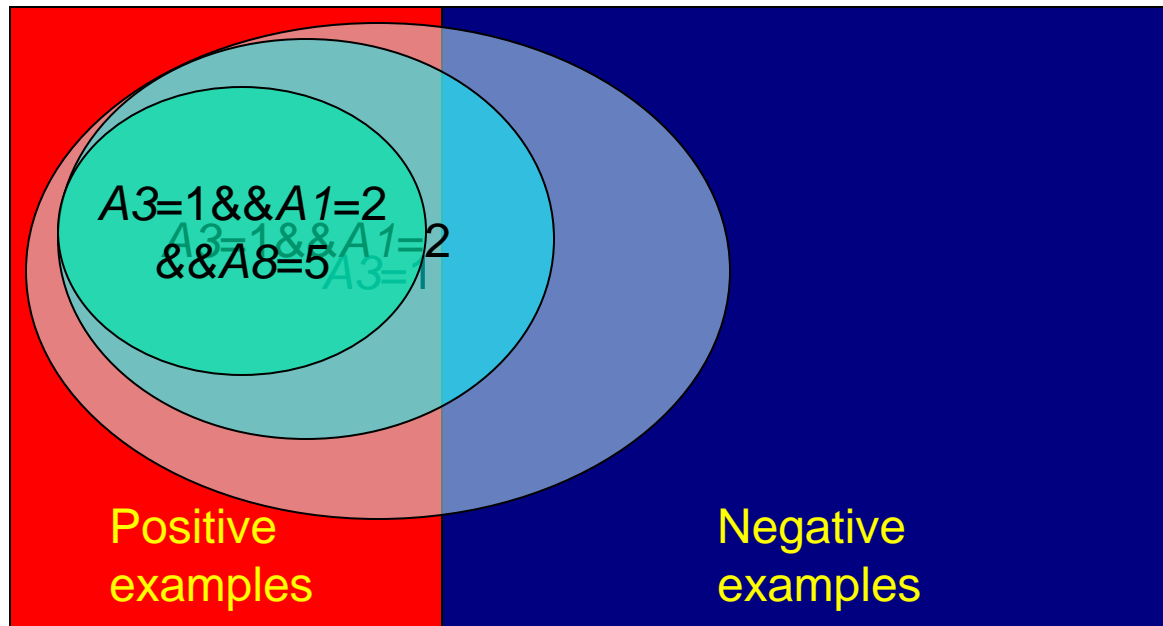
**while** (enough target tuples left)  
    generate a rule  
    remove positive target tuples satisfying this rule



**Positive examples**（正元组）：用于学习规则的类的元组，其余的为负元组

# 规则产生

- To generate a rule (利用度量判断规则的质量)  
    **while**(true)  
        find the best predicate  $p$   
        **if**  $\text{foil-gain}(p) > \text{threshold}$  **then** add  $p$  to current rule  
        **else break**





# 顺序覆盖算法

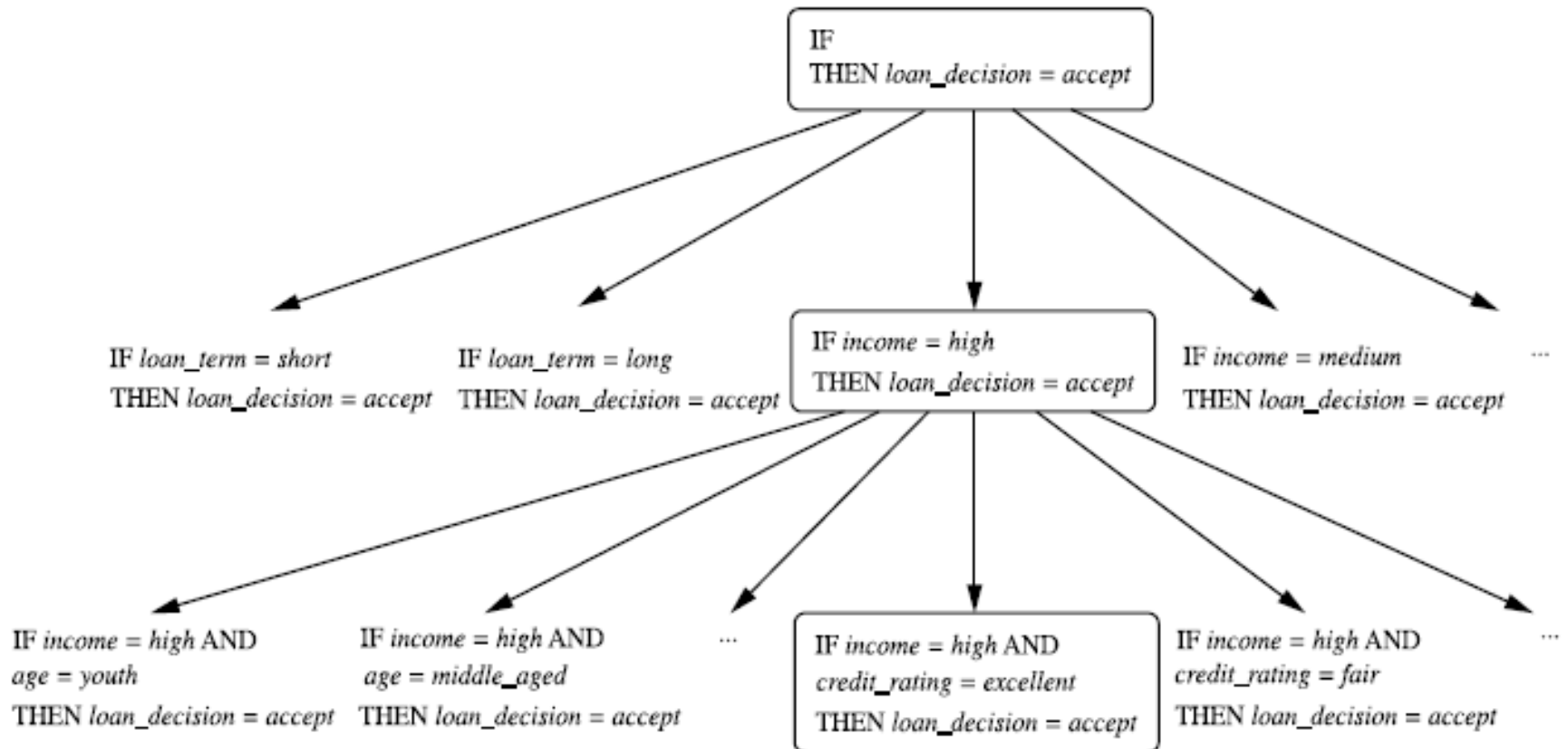
输出：IF-THEN规则的集合。

方法：

- (1)  $Rule\_set = \{\}$ ; // 学习的规则的初始集为空
- (2) **for** 每个类  $c$  **do**
- (3)     **repeat**
- (4)          $Rule = \mathbf{Learn\_One\_Rule} (D, Att\_vals, c) ;$
- (5)         从  $D$  中删除  $Rule$  覆盖的元组;
- (6)     **until** 终止条件满足;
- (7)      $Rule\_set = Rule\_set + Rule;$  // 将新规则添加到规则集
- (8) **endfor**
- (9) 返回  $Rule\_Set$ ;

# 规则学习步骤

- 从可能的最一般的规则开始: **condition = empty**
- 采用贪心的深度优先策略添加新属性到规则中
  - 选择对“规则质量”提高最大的那个属性



# 规则质量度量与剪枝

- 规则质量度量：同时考虑覆盖率和准确率
- FOIL-Gain：评价扩展条件的info\_gain


$$FOIL\_Gain = pos' \times (\log_2 \frac{pos'}{pos' + neg'} - \log_2 \frac{pos}{pos + neg})$$

- FOIL：一阶归纳学习器
  - $pos(neg)$ ：规则覆盖的正（负）元组数
  - 偏向于具有高准确率并覆盖许多正元组的规则
- 基于独立测试集进行规则剪枝，即删除一个属性测试

$$FOIL\_Prune(R) = \frac{pos - neg}{pos + neg}$$

# 第8章：分类：基本概念

---

- 基本概念
- 决策树归纳
- 贝叶斯分类方法
- 基于规则的分类
- 模型评估与选择 
- 提高分类准确率的技术
- 小结

# 模型评价与选择

---

- 评价指标: 怎样度量准确率?
  - 使用测试集（带标签）代替训练集评估准确度
- 估计分类器准确率的方法:
  - 保持
  - 随机二次抽样
  - 交叉验证**Cross-validation**
  - 自助法**Bootstrap**
- 分类器比较: 比较分类器的准确率之差是否偶然
  - 置信区间
  - 代价效益分析和**ROC**曲线

# 分类器评价指标: 混淆矩阵

## 混淆矩阵Confusion Matrix:

实际类\预测类	$C_1$	$\sim C_1$
$C_1$	<b>True Positives (TP)</b>	<b>False Negatives (FN)</b>
$\sim C_1$	<b>False Positives (FP)</b>	<b>True Negatives (TN)</b>

- 感兴趣的类定为“**正类**”或“**阳性类**”，反之则为“**负/阴性类**”
  - **正样本/负样本**
- 可以提供额外的行/列提供“**合计**”和“**识别率**”

实际类\预测类	buy_computer = yes	buy_computer = no	Total
buy_computer = yes	<b>6954</b>	<b>46</b>	7000
buy_computer = no	<b>412</b>	<b>2588</b>	3000
Total	7366	2634	10000

# 分类器评价指标: 准确度, 误差率, 灵敏性, 特效性

A\P	C	~C	
C	<b>TP</b>	<b>FN</b>	<b>P</b>
~C	<b>FP</b>	<b>TN</b>	<b>N</b>
	<b>P'</b>	<b>N'</b>	<b>All</b>

- 分类器准确度或识别率: 测试元组被正确识别的比例

$$\text{Accuracy} = (\text{TP} + \text{TN})/\text{All}$$

- 误差率:  $1 - \text{Accuracy}$  , or

$$\text{Error rate} = (\text{FP} + \text{FN})/\text{All}$$

- 类分布不平衡问题: 感兴趣主类是稀少的, 比如: 欺诈, 或者HIV阳性

- 敏感度: 真正例识别率

$$\text{Sensitivity} = \text{TP}/\text{P}$$

- 特效性: 正确识别负元组的识别率

$$\text{Specificity} = \text{TN}/\text{N}$$

# 灵敏性和特效性

如图所示的医疗数据的混淆矩阵，其中，类标号属性 cancer 的类值为 yes 和 no  
计算其 灵敏度、特效性、准确率

灵敏度：  $90/300 = 30\%$     特效性：  $9560/9700=98.56\%$     准确率  $(9560+90) / 10000 = 96.50\%$

类	yes	no	合计	识别率 (%)
Yes	<b>TP 90</b>	<b>FN 210</b>	<b>300</b>	<b>30.00</b>
no	<b>FP 140</b>	<b>TN 9560</b>	<b>9700</b>	<b>98.56</b>
合计	<b>230</b>	<b>9770</b>	<b>10000</b>	<b>96.40</b>



# 分类器评价指标: 精度、召回、F度量

- **Precision:** 精确度的度量, 正确-被分类器标记为正类的样本中实际上属于“正类”的比例

$$precision = \frac{TP}{TP + FP}$$

- **Recall:** 完全性的度量, 正元组标记为正的百分比 (精度和召回率逆关系)

$$recall = \frac{TP}{TP + FN}$$

- **F measure ( $F_1$  or F-score):** 精度和召回的调和平均值

$$F = \frac{2 \times precision \times recall}{precision + recall}$$

- **$F_\beta$ :** 精确度和召回率的加权量

- Assigns  $\beta$  times as much weight to recall as to precision

$$F_\beta = \frac{(1 + \beta^2) \times precision \times recall}{\beta^2 \times precision + recall}$$

# 分类器评价指标: 例子

真实类\预测类	cancer = yes	cancer = no	Total	Recognition(%)
cancer = yes	<b>90</b>	<b>210</b>	300	30.00 ( <i>sensitivity</i> )
cancer = no	<b>140</b>	<b>9560</b>	9700	98.56 ( <i>specificity</i> )
Total	230	9770	10000	96.40 ( <i>accuracy</i> )

$$Precision = 90/230 = 39.13\%$$

$$Recall = 90/300 = 30.00\%$$

# 评测分类器的正确率:

## Holdout & Cross-Validation Methods

---

### ■ 保持方法

- 给定数据随机分成两个部分
  - 训练集 (e.g., 2/3) 用于模型构造
  - 测试集 (e.g., 1/3) 用于正确率估计
- 随机二次抽样: a variation of holdout
  - 重复保持方法 $k$ 次,  $\text{accuracy}$  = 所有正确率的平均值

### ■ 交叉验证 ( $k$ -fold, $k=10$ 最常用)

- 随机分割数据为 $k$ 互不相交的子集, 每一个大小近似相等; 在 $i$ -th 迭代中, 使用 $D_i$ 为测试集其他的为训练集
- 留一法:  $k$  设置为原始样本数
- 分层交叉验证: 每个部分分层使得每个子集中类分布近似原始数据

# 评测分类器的正确率: 自助法

## ■ 自助法 ( bootstrap )

- 从给定样本中**有放回**的均匀抽样: 每次一个样本被选中, 把它加入训练集并且等可能得被再次选中
- 对于小样本数据, 效果很好

## ■ 多个自助法 (最常用的是 .632)

- 含 $d$ 个样本的数据集有放回抽样 $d$ 次, 产生 $d$ 个样本的训练集, 没有被抽到的样本组成测试集.
- 大约63.2% 的样本被抽中, 剩余的36.8% 形成测试集

$$(1 - 1/d)^d \approx e^{-1} = 0.368$$

- 重复抽样过程 $k$  次, 总体准确率为:

$$Acc(M) = \frac{1}{k} \sum_{i=1}^k (0.632 \times Acc(M_i)_{test\_set} + 0.368 \times Acc(M_i)_{train\_set})$$

# 估计置信区间: 分类器 M1 vs. M2

---

- 假定有两个分类器  $M_1$  and  $M_2$ , 那一个更好?
- 用10折交叉验证获得了  $\overline{err}(M_1)$   $\overline{err}(M_2)$
- 这些平均误差率仅是未来数据总体误差的一种估计
- 2个错误率之间差别如果是偶然的?
  - 使用统计显著性检验
  - 获得估计误差的置信界 ( **confidence limits** )

# 估计置信区间：原假设(null hypothesis)

---

- 执行10折交叉验证
- 假定样本服从 $k-1$ 个自由度的 $t$ 分布( $k=10$ ), 利用 **t-test** (or **student's t-test**)
- 原假设:  $M_1$  &  $M_2$  相同 (即没有区别)
- 如果可以拒绝原假设
  - 可以断定 $M_1$  &  $M_2$  间的不同是统计上显著的
  - 选择具有较低错误率的模型

# 估计置信区间: t-检验

- 当只有一个测试集时: 成对比较
  - 对于10倍交叉验证中第*i*次, 使用相同的样本分割来计算  $err(M_1)_i$  and  $err(M_2)_i$
  - 然后求平均  $\overline{err}(M_1)$   $\overline{err}(M_2)$
  - t-检验 : 计算自由度为  $k-1$  的t统计值

$$t = \frac{\overline{err}(M_1) - \overline{err}(M_2)}{\sqrt{var(M_1 - M_2)/k}}$$

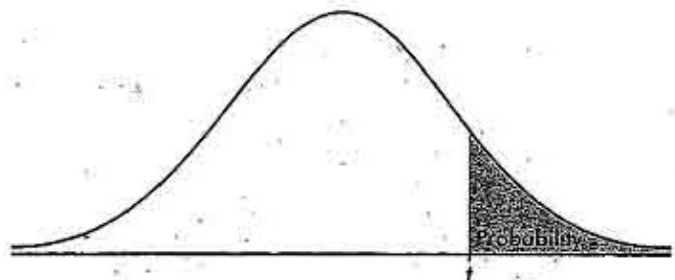
其中

$$var(M_1 - M_2) = \frac{1}{k} \sum_{i=1}^k \left[ err(M_1)_i - err(M_2)_i - (\overline{err}(M_1) - \overline{err}(M_2)) \right]^2$$

若测试的次数不相同  $var(M_1 - M_2) = \sqrt{\frac{var(M_1)}{k_1} + \frac{var(M_2)}{k_2}},$



# 估计置信区间: Table for t-distribution



对称的分布

■ 显著水平 $sig$ :

e.g.,  $sig=0.05$ :  $M_1$  &  $M_2$   
有95%的可能是显著不同的

置信界,  $z = sig/2$

TABLE B: t-DISTRIBUTION CRITICAL VALUES

df	Tail probability $p$											
	.25	.20	.15	.10	.05	.025	.02	.01	.005	.0025	.001	.0005
1	1.000	1.376	1.963	3.078	6.314	12.71	15.89	31.82	63.66	127.3	318.3	636.6
2	.816	1.061	1.386	1.886	2.920	4.303	4.849	6.965	9.925	14.09	22.33	31.60
3	.765	.978	1.250	1.638	2.353	3.182	3.482	4.541	5.841	7.453	10.21	12.92
4	.741	.941	1.190	1.533	2.132	2.776	2.999	3.747	4.604	5.598	7.173	8.610
5	.727	.920	1.156	1.476	2.015	2.571	2.757	3.365	4.032	4.773	5.893	6.869
6	.718	.906	1.134	1.440	1.943	2.447	2.612	3.143	3.707	4.317	5.208	5.959
7	.711	.896	1.119	1.415	1.895	2.365	2.517	2.998	3.499	4.029	4.785	5.408
8	.706	.889	1.108	1.397	1.860	2.306	2.449	2.896	3.355	3.833	4.501	5.041
9	.703	.883	1.100	1.383	1.833	2.262	2.398	2.821	3.250	3.690	4.297	4.781
10	.700	.879	1.093	1.372	1.812	2.228	2.359	2.764	3.169	3.581	4.144	4.587
11	.697	.876	1.088	1.363	1.796	2.201	2.328	2.718	3.106	3.497	4.025	4.437
12	.695	.873	1.083	1.356	1.782	2.179	2.303	2.681	3.055	3.428	3.930	4.318
13	.694	.870	1.079	1.350	1.771	2.160	2.282	2.650	3.012	3.372	3.852	4.221
14	.692	.868	1.076	1.345	1.761	2.145	2.264	2.624	2.977	3.326	3.787	4.140
15	.691	.866	1.074	1.341	1.753	2.131	2.249	2.602	2.947	3.286	3.733	4.073
16	.690	.865	1.071	1.337	1.746	2.120	2.235	2.583	2.921	3.252	3.686	4.015
17	.689	.863	1.069	1.333	1.740	2.110	2.224	2.567	2.898	3.222	3.646	3.965
18	.688	.862	1.067	1.330	1.734	2.101	2.214	2.552	2.878	3.197	3.611	3.922
19	.688	.861	1.066	1.328	1.729	2.093	2.205	2.539	2.861	3.174	3.579	3.883
20	.687	.860	1.064	1.325	1.725	2.086	2.197	2.528	2.845	3.153	3.552	3.850
21	.686	.859	1.063	1.323	1.721	2.080	2.189	2.518	2.831	3.135	3.527	3.819
22	.686	.858	1.061	1.321	1.717	2.074	2.183	2.508	2.819	3.119	3.505	3.792
23	.685	.858	1.060	1.319	1.714	2.069	2.177	2.500	2.807	3.104	3.485	3.768
24	.685	.857	1.059	1.318	1.711	2.064	2.172	2.492	2.797	3.091	3.467	3.745
25	.684	.856	1.058	1.316	1.708	2.060	2.167	2.485	2.787	3.078	3.450	3.725
26	.684	.856	1.058	1.315	1.706	2.056	2.162	2.479	2.779	3.067	3.435	3.707
27	.684	.855	1.057	1.314	1.703	2.052	2.158	2.473	2.771	3.057	3.421	3.690
28	.683	.855	1.056	1.313	1.701	2.048	2.154	2.467	2.763	3.047	3.408	3.674
29	.683	.854	1.055	1.311	1.699	2.045	2.150	2.462	2.756	3.038	3.396	3.659
30	.683	.854	1.055	1.310	1.697	2.042	2.147	2.457	2.750	3.030	3.385	3.646
40	.681	.851	1.050	1.303	1.684	2.021	2.123	2.423	2.704	2.971	3.307	3.551
50	.679	.849	1.047	1.299	1.676	2.009	2.109	2.403	2.678	2.937	3.261	3.496
60	.679	.848	1.045	1.296	1.671	2.000	2.099	2.390	2.660	2.915	3.232	3.460
80	.678	.846	1.043	1.292	1.664	1.990	2.088	2.374	2.639	2.887	3.195	3.416
100	.677	.845	1.042	1.290	1.660	1.984	2.081	2.364	2.626	2.871	3.174	3.390
1000	.675	.842	1.037	1.282	1.646	1.962	2.056	2.330	2.581	2.813	3.098	3.300
∞	.674	.841	1.036	1.282	1.645	1.960	2.054	2.326	2.576	2.807	3.091	3.291
	50%	60%	70%	80%	90%	95%	96%	98%	99%	99.5%	99.8%	99.9%
	Confidence level $C$											



# 估计置信区间: Statistical Significance

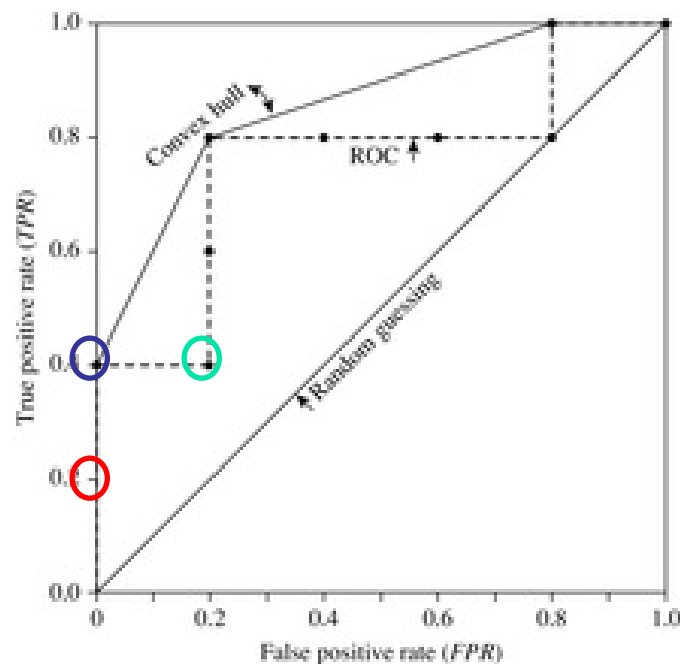
---

- $M_1$  &  $M_2$  是否显著得不同?
  - 计算 $t$ 并选择显著水平 (e.g.  $\text{sig} = 5\%$ )
  - 查询 $t$ -分布表: 找到对应为自由度 $k-1$ 的 $t$ 的值 $z$
  - If  $t > z$  or  $t < -z$ , 那么 $t$ 的值位于拒绝域:
    - 拒绝 $M_1$  &  $M_2$  相似的假设
    - 得出 $M_1$  &  $M_2$  显著不同的结论
  - Otherwise,  $M_1$  &  $M_2$  之间的不同是偶然的, 即选择 $M_1$  &  $M_2$  没有太大区别

# 模型选择: ROC Curves

Tuple #	Class	Prob.	TP	FP	TN	FN	TPR	FPR
1	P	0.90	1	0	5	4	0.2	0
2	P	0.80	2	0	5	3	0.4	0
3	N	0.70	2	1	4	3	0.4	0.2
4	P	0.60	3	1	4	2	0.6	0.2
5	P	0.55	4	1	4	1	0.8	0.2
6	N	0.54	4	2	3	1	0.8	0.4
7	N	0.53	4	3	2	1	0.8	0.6
8	N	0.51	4	4	1	1	0.8	0.8
9	P	0.50	5	4	0	1	1.0	0.8
10	N	0.40	5	5	0	0	1.0	1.0

每个样本被判  
为正的的概率



$$\text{TPR} = \text{TP} / P$$

$$\text{FPR} = \text{FP} / N$$

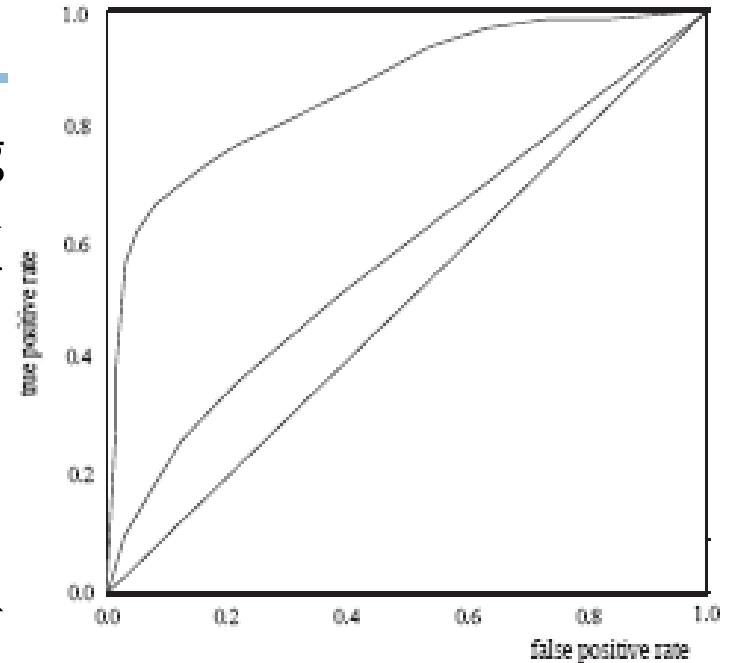
真正例率TPR和假正例率FPR之间的权衡

TPR模型正确标记的正元组的比例。

FPR模型错误标记为正的负元组的比例

# 模型选择: ROC Curves


- ROC (Receiver Operating Characteristics) curves: 图形比较分类模型
- TPR和FPR间的折衷
- ROC曲线下的面积就是模型正确率的度量
- 测试元组递减序排列: 最可能属于正类的排在最顶端
- 越靠近对角线的分类器准确率越低



- 垂直坐标表示the true positive rate
- 水平坐标表示the false positive rate
- 同时显示对角线
- A model with perfect accuracy will have an area of 1.0

# 第8章：分类：基本概念

---

- 基本概念
- 决策树归纳
- 贝叶斯分类方法
- 基于规则的分类
- 模型评估与选择
- 提高分类准确率的技术 
- 小结

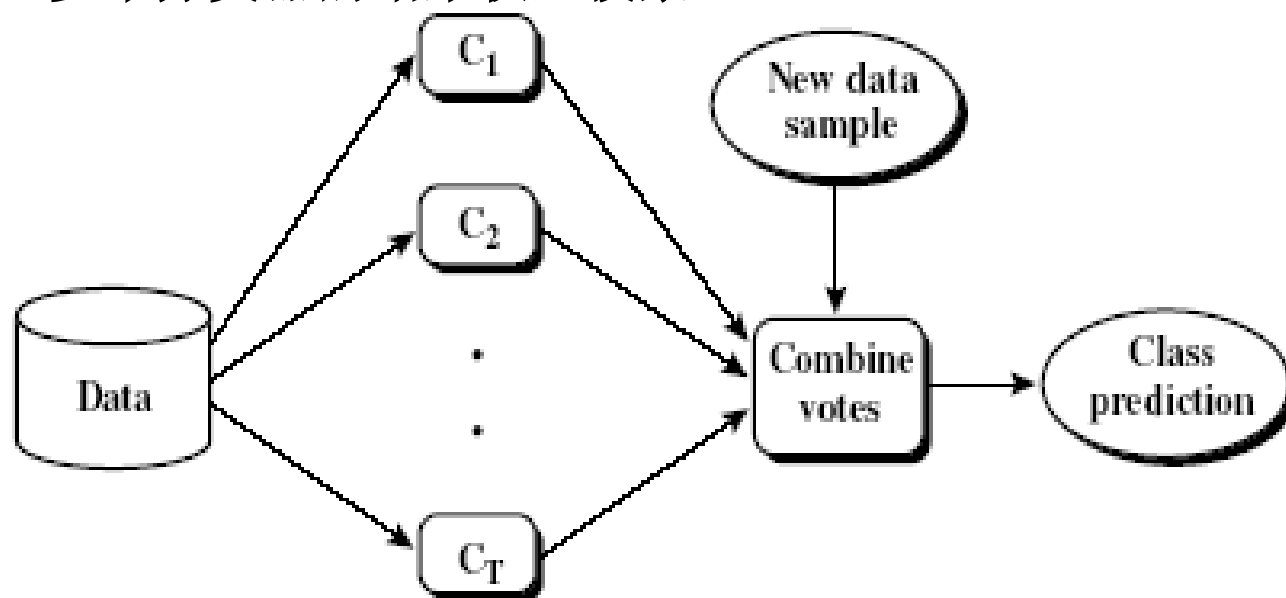
# 组合方法: Increasing the Accuracy

## ■ 集成方法 (Ensemble methods)

- 使用多个模型的组合来提高accuracy
- 组合多个学习的模型,  $M_1, M_2, \dots, M_k$ , 获得提高的模型  $M^*$
- 基于多个分类器的投票

## ■ 常用的组合方法

- 装袋: 多个分类器的结果进行多数表决
- 提升: 多个分类器的结果权重投票



# 装袋Bagging

## ■ 训练

- 给定包含 $d$ 个元组的数据 $D$ ，在第 $i$ 次迭代，从 $D$ 中有放回抽取 $d$ 个样本组成训练集 $D_i$ (i.e., bootstrap)，从 $D_i$ 学习一个分类器 $M_i$

## ■ 分类：分类一个未知样本 $X$

- 每个分类器  $M_i$  给出预测结果
- 装袋分类器 $M^*$  计算投票，把得票最多的类分配给 $X$

## ■ 预测：每个分类器预测的值的平均值

## ■ 正确性Accuracy

- 常常优于 $D$  上单个分类器的正确率
- 对噪音数据：不会很差，更健壮
- Proved improved accuracy in prediction

# 提升Boosting

- 类比：咨询几个医生，在原来的诊断准确性的基础上分配权重，加权诊断的组合为结果
- **Boosting如何工作？**
  - **Weights**分配给每个训练样本
  - 迭代学习一系列分类器
  - 学习 $M_i$ 后，权重更新使得，后续得分分类器 $M_{i+1}$ 更关注于 $M_i$ 错误分类的训练样本
  - 最后的分类器 $M^*$ 组合了每个独立分类器的投票，其中每个分类器的权重是其正确率的函数
- 可以扩充Boosting算法用于数值预测
- 与bagging比较：Boosting倾向于得到更高的准确率，但有过拟合错误分类数据的风险

# Adaboost

- 数据集含  $d$  class-labeled 元组,  $(\mathbf{X}_1, y_1), \dots, (\mathbf{X}_d, y_d)$
- 最初, 每个元组的权重为  $1/d$
- $k$ 轮中产生 $k$ 个分类器, 其中第 $i$ 轮:
  - 从 $D$  有放回抽取训练集  $D_i$  (大小相等)
  - 每个元组被选中的概率基于其权重
  - 分类模型 $M_i$ 学习自 $D_i$
  - 使用 $D_i$ 为测试集计算误差率
  - 如果一个元组被错分, 权重增加, 否则下降
- 误差率:  $err(\mathbf{X}_j)$ 为错误分类元组 $\mathbf{X}_j$ 误差, 分类器 $M_i$ 误差率是元组错误分类的权重和:

$$error(M_i) = \sum_j^d w_j \times err(\mathbf{X}_j)$$

- 分类器 $M_i$ 投票权重为: 
$$\log \frac{1 - error(M_i)}{error(M_i)}$$



# 随机森林

- **Random Forest:**

- 每个分类器为*decision tree*，在每个结点上使用随机选出的属性来分裂产生判定树
- 分类时，每棵树投票得票最多的类返回结果

- 两种构造方法:

- **Forest-RI** (*random input selection*): 每个结点随机选F个属性为分裂的候选，用CART方法产生最大尺寸的树
- **Forest-RC** (*random linear combinations*): 以现有属性的线性组合来产生新属性（降低了单个分类器间的相关性）


- 准确率比得上Adaboost，对误差和孤立点更稳健
- 每次分裂时对选出的候选属性数目不敏感，**faster than bagging or boosting**

# 分类类别不平衡数据集

- 类别不平衡问题.
- 传统的方法假定平衡的类别分布和相等的错误代价在很多应用场景下都不适合
- 二元分类中典型的方法处理不平衡数据:
  - 过采样Oversampling: 对正类数据过/多采样
  - Under-sampling: 随机减少负类的样本
  - 阈值-移动Threshold-moving: 移动判定阈值 $t$ , 使得少数类元组更容易识别, 减少 (昂贵的) 假阴性错误的机会

# 第8章：分类：基本概念

---

- 基本概念
- 决策树归纳
- 贝叶斯分类方法
- 基于规则的分类
- 模型评估与选择
- 提高分类准确率的技术
- 小结 

# Summary (I)

---

- **Classification** is a form of data analysis that extracts **models** describing important data classes.
- Effective and scalable methods have been developed for **decision tree induction**, **Naive Bayesian classification**, **rule-based classification**, and many other classification methods.
- **Evaluation metrics** include: accuracy, sensitivity, specificity, precision, recall,  $F$  measure, and  $F_\beta$  measure.
- **Stratified k-fold cross-validation** is recommended for accuracy estimation. **Bagging** and **boosting** can be used to increase overall accuracy by learning and combining a series of individual models.

# Summary (II)

---

- **Significance tests** and **ROC curves** are useful for model selection.
- There have been numerous **comparisons of the different classification** methods; the matter remains a research topic
- No single method has been found to be superior over all others for all data sets
- Issues such as accuracy, training time, robustness, scalability, and interpretability must be considered and can involve trade-offs, further complicating the quest for an overall superior method

# Scalable Decision Tree Induction Methods

---

- **SLIQ** (EDBT'96 — Mehta et al.)
  - Builds an index for each attribute and only class list and the current attribute list reside in memory
- **SPRINT** (VLDB'96 — J. Shafer et al.)
  - Constructs an attribute list data structure
- **PUBLIC** (VLDB'98 — Rastogi & Shim)
  - Integrates tree splitting and tree pruning: stop growing the tree earlier
- **RainForest** (VLDB'98 — Gehrke, Ramakrishnan & Ganti)
  - Builds an AVC-list (attribute, value, class label)
- **BOAT** (PODS'99 — Gehrke, Ganti, Ramakrishnan & Loh)
  - Uses bootstrapping to create several small samples

# Data Cube-Based Decision-Tree Induction

---

- Integration of generalization with decision-tree induction (Kamber et al.'97)
- Classification at primitive concept levels
  - E.g., precise temperature, humidity, outlook, etc.
  - Low-level concepts, scattered classes, bushy classification-trees
  - Semantic interpretation problems
- Cube-based multi-level classification
  - Relevance analysis at multi-levels
  - Information-gain analysis with dimension + level