# 人工智能
# ARTIFICIAL INTELLIGENCE

主讲：鲍军鹏 博士
西安交通大学电信学院计算机系
电子邮箱：dr.baojp@googlemail.com

# 第六章　机器学习

- 6.1 概述
- 6.2 决策树学习
- 6.3 贝叶斯学习
- 6.4 统计学习
- 6.5 遗传算法
- 6.6 聚类
- 6.7 特征选择与提取
- 6.8 其它学习方法

# 6.1 概述

- 6.1.1 什么是机器学习?

  学习是人类具有的一种重要智能行为，但究竟什么是学习，长期以来却众说纷纭。

- 关于“学习”这一概念的主要观点:
  - 学习是系统改进其性能的过程。这是西蒙的观点。
    - 西蒙的观点：学习就是系统在不断重复的工作中对本身能力的增强或者改进，使得系统在下一次执行同样任务或类似任务时，会比现在做得更好或效率更高。
  - 学习是获取知识的过程。这是从事专家系统研究的人们的观点。
  - 学习是技能的获取。这是心理学家的观点。
  - 学习是事物规律的发现过程。

# 基本的学习形式

- 基本的学习形式有2种：
  - 知识获取和技能求精。
- 例如，我们说某人学过物理。
  - 我们的意思是，此人已经掌握了有关物理学的基本概念，并且理解其含义，同时还懂得这些概念之间以及它们与物理世界之间的关系。
  - 一般地，知识获取可看作学习新的符号信息，而这些符号信息是以有效方式与应用这种信息的能力相适应的。
- 第二类学习形式是通过实践逐步改进机制和认知技能。
- 例如骑自行车或弹钢琴等等。
  - 学习的很多过程都是由改进所学的技能组成。这些技能包括意识的或者机制的协调，而这种改进又是通过反复实践和从失败的行为中纠正偏差来进行的。
- 知识获取的本质可能是一个自觉的过程，其结果产生新的符号知识结构和智力模型。而技能求精则是下意识地借助于反复实践来实现的。人类的学习一般表现尾这两种活动的结合。

# 机器学习的定义

- 至今，还没有统一的“机器学习”定义，而且也很难给出一个公认的和准确的定义。一般认为机器学习是研究如何使用机器来模拟人类学习活动的一门学科。
- 最早的具有学习能力的程序：
  - 1959年美国的塞缪尔(Samuel)设计了一个下棋程序，这个程序具有学习能力，它可以在不断的对奕中改善自己的棋艺。4年后，这个程序战胜了设计者本人。又过了3年，这个程序战胜了美国一个保持8年之久的常胜不败的冠军。

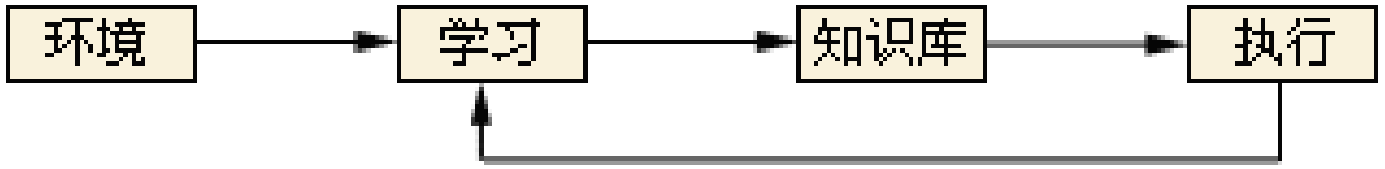
# 机器学习发展简史

- 第一阶段
  - 在50年代中叶到60年代中叶，神经元模型的研究。
- 第二阶段
  - 在60年代中叶至70年代，符号学习的研究。
- 第三阶段
  - 80年代，连接学习的研究与符号学习的进展。
  - 90年代以后，综合多学科、多种方法的知识发现、数据挖掘与机器学习研究

# ML、KDD和DM

- 知识发现(Knowledge Discovering in Database)与数据挖掘(Data Mining)是人工智能、机器学习与(Machine Learning)数据库技术相结合的产物。
  - 1980年,在美国召开了第一届国际机器学习研讨会;1984年,《机器学习》杂志问世。我国于1987年召开了第一届全国机器学习研讨会;1989年成立了以中国科技大学蔡庆生教授为理事长的理事会。
  - KDD一词是在1989年于美国底特律市召开的第一届KDD国际学术会议上正式形成的。
- 1995年,在加拿大召开了第一届知识发现和数据挖掘国际学术会议。由于数据库中的数据被形象地喻为矿床,因此数据挖掘一词很快流传开来。

# 机器学习系统的基本结构




学习系统的基本结构

- 环境向系统的学习部分提供某些信息，
- 学习部分利用这些信息修改知识库，以增进系统执行部分完成任务的效能，
- 执行部分根据知识库完成任务，同时把获得的信息反馈给学习部分。
- 在具体的应用中，环境，知识库和执行部分决定了具体的工作内容，学习部分所需要解决的问题完全由上述3部分确定。

# 机器学习的一般步骤

- 机器学习系统中学习环节的一般过程

# 6.1.2 机器学习方法分类

- 按照有无指导来分：
  - 有监督学习（或有导师学习）、无监督学习（或无导师学习）和强化学习（或增强学习）。
- 按学习方法来分：
  - 有机械式学习、指导式学习、范例学习、类比学习、解释学习。
- 按推理策略来分：
  - 有演绎学习、归纳学习、类比学习、解释学习等。
- 综合多因素的分类：
  - 有人工神经网络学习、进化学习、概念学习、分析学习、基于范例的学习等等。

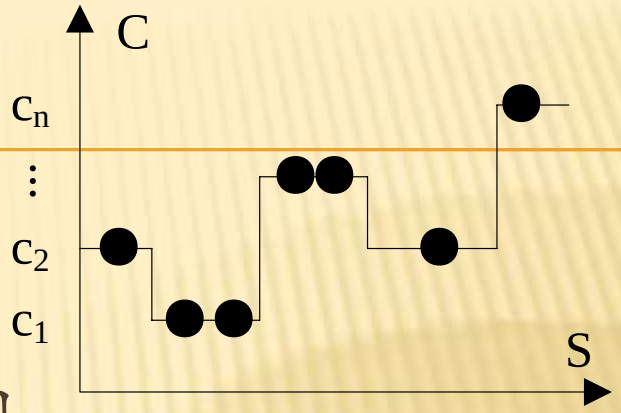
# 6.1.3 机器学习的基本问题

- 机器学习中解决的基本问题主要有：
  - 分类、聚类、预测、联想、优化。
- 令S表示数据空间，Z表示目标空间。
  - 机器学习就是在现有观察的基础上求得一个函数

$$L:S \rightarrow Z,$$

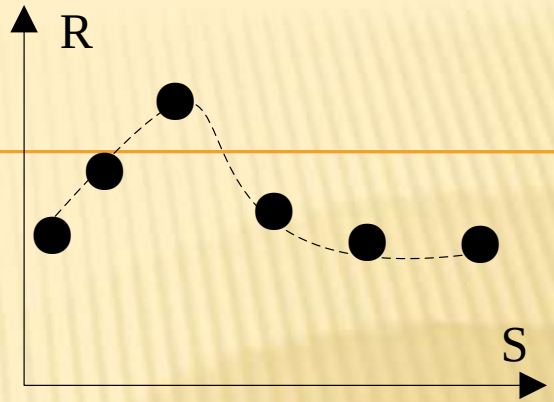
  实现从给定数据到目标空间的映射。
- 不同特征的学习函数实际上表示了不同的基本问题。

# 分类问题




- 目标空间是已知有限离散值空间，
  即， $Z=C=\{c_1,c_2,\ldots c_i,\ldots,c_n\}$
  待求函数就是分类函数（分类器/分类模型）。
- 分类问题所用的训练数据是<D,C>， $D \subset S$ 。
- 由于学习时目标类别已知，所以分类算法都是有监督学习。
- 常用的方法：
  - 决策树方法、贝叶斯方法、前馈神经网络BP算法、支持向量机方法等

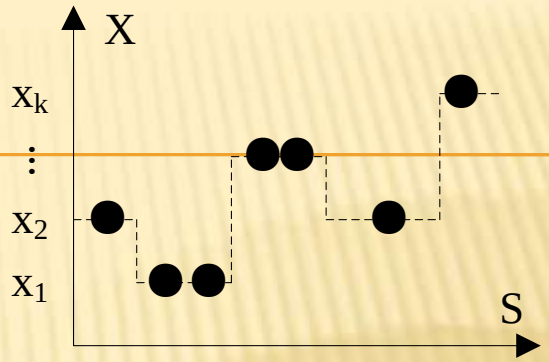
# 预测问题




- 目标空间是连续值空间，待求函数就是回归（拟合）曲线（面）。

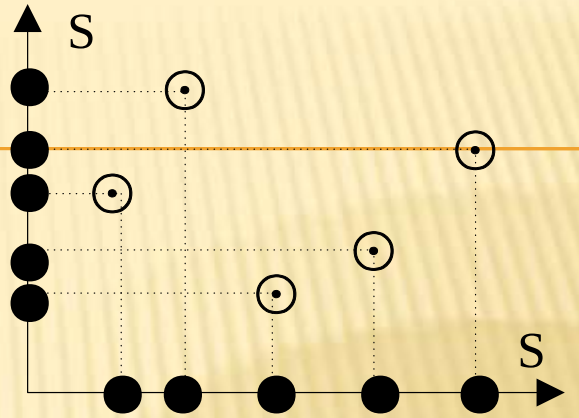
  此时机器学习解决预测问题，也就是求一个数据在目标空间中符合某观测规律的象。
- 预测问题所用的训练数据是<D,R>， $D \subset S$ 。
  - 一般情况下我们事先已知（或者选择了）曲线（面）模型，需要学习的是模型中的参数。
  - 例如已知多项式模型，但是要学习各项的系数。
- 常用的方法：
  - 人工神经网络方法、线性回归、非线性回归、灰色预测模型等。

# 聚类问题




- 目标空间是未知有限离散值空间，
  即，$Z=X=\{x_1,x_2,\ldots,x_k\}$
  待求函数就是聚类函数，也称为聚类模型。
- 聚类问题就是把已知数据集划分为不同子集（类别），并且不同类别之间的差距越大越好，同一类别内的数据差距越小越好。
- 聚类问题所用的训练数据是D（$D \subset S$）。
- 聚类问题要用无监督学习
- 常用的方法：
  - 划分聚类法、层次聚类法、基于密度的聚类、基于网格的聚类、自组织特征映射网络等等。

# 联想问题

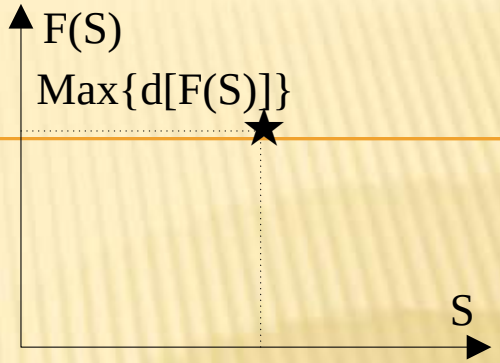



- 目标空间就是数据空间本身，

  即，　　Z=S

  待求函数就是求自身内部的一种映射。
- 联想问题，也称为相关性分析或者关联问题
  - 就是发现不同数据（属性）之间的相互依赖关系。
  - 简单地说，就是可以从事物A推出事物B，即A→B
- 常用的方法：
  - 反馈神经网络、关联规则、回归分析等等。

# 优化问题




- 目标空间是数据空间上的某种函数（用F(S)表示），且学习目标为使对函数F(S)的某种度量d[F(S)]达到极值。
- 解决优化问题，就是在给定数据范围内寻找使某值达到最大（最小）的方法。
- 优化问题一般都有一些约束条件
  - 例如时空资源的限制等等。
  - 典型代表就是NP问题，这也是计算机科学中的一类经典问题。
- 解决优化问题对于提高系统效率，保证系统实用性有重要意义。
- 常用的方法有：
  - 遗传算法、Hopfield神经网络、线性规划方法等等。

# 6.1.4 评估学习结果

- 机器学习一般不要求结果100%正确。
- 评估原则
  - 学习结果的合理性和有效性
    - 模型的泛化能力（Generalization）越强越好。
  - 算法复杂度（Complexity）
    - 时间复杂度、空间复杂度等。
    - 减小时间复杂度常用的思路:
      - 简化问题，降低要求;
      - 用空间换时间;
      - 用并行化算法，提高并行度

# 评估原则

- 模型鲁棒性（Robustness）
  - 就是系统的健壮性，就是系统处理各种非正常数据的能力。
  - 包括
  - 对数据噪声的处理，
  - 对缺失数据及其它包含不完整信息数据的处理，
  - 对错误数据或者含有矛盾数据的处理等等。

# 评估原则

- 模型适应性。
  - 是指对于不同数据，学习模型本身需要做多少人工调整。
  - 我们一般都希望模型本身需要人工指定参数越少越好。
  - 自适应模型并不意味着彻底不需要人工指定的参数。
- 模型描述的简洁性和可解释性。
  - 根据奥坎姆剃刀（Occam's Razor）原则，应该优先选择更简单的假设。
  - 模型描述愈简洁、愈容易理解，则愈受欢迎。

# 机器学习中的测试数据

- 从S中分割出训练数据和测试数据
  - 假设S是已有数据集，并且训练数据和测试数据都遵从同样的分布规律。
- 保留法（Holdout）
  - 取S的一部分（通常为2/3）作为训练数据，剩下的部分（通常为1/3）作为测试数据。
  - 最后在测试数据集上验证学习结果。
- 特点
  - 仅仅使用了部分（2/3）数据训练学习模型，没有充分利用所有的已知数据。
  - 保留法一般用于已知数据量非常巨大的时候。

# 机器学习中的测试数据

- 交叉验证法（Cross Validation）
  - 也称为交叉纠错法
  - 把S划分为k个不相交的子集，即

    $S=\{S_1,S_2,\ldots,S_k\}$，$(S_i \cap S_j=\Phi, 1\le i,j\le k)$
  - 然后取其中一个子集作测试集，剩下数据作训练集。
    - 取$S_i$做测试集，则$S-S_i$就做训练集。
    - 重复k次，把每一个子集都做一次测试集。
    - 于是会得到k个测试结果，最终的测试结果就是这k个测试结果的平均值。
- 特点
  - 交叉验证法还可以再重复多次，每次变换不同的k值或者不同的划分。
  - 交叉验证法充分利用了所有已知数据，可以获得较好的学习结果，但是显然需要更长的训练时间。
  - 交叉验证法一般用于已知数据量不太大的时候。

# 机器学习中的测试数据

- 随机法
  - 随机抽取S中的一部分数据作为测试数据，把剩下的数据作为训练数据。
  - 重复这一过程足够多次。
  - 最终测试结果是所有测试结果的平均值。
- 特点
  - 随机法可以重复无数次，每个数据都可能被充分地用于训练和测试，可以把测试结果的置信区间减小到指定宽度。
  - 随机法中不同的测试集不能看作是对已知数据的独立抽取。而交叉验证法中不同的测试集是独立的，因为一个数据只在测试集中出现一次。

# 度量学习结果有效性的指标

- 误差（Error）
  - 测试数据集T上的误差是

$$Error(T) = \sum_{i=1}^{|T|} P_i \left\| E_i - L_i \right\|$$

其中，$E_i$表示某个数据的理想结果，$L_i$表示该数据的机器学习结果。

  - 常用的误差实际上就是方差

$$Error(T) = \sum_{i=1}^{|T|} P_i \sum_{j=1}^{d} (E_{ij} - L_{ij})^2$$

# 度量学习结果有效性的指标

- 正确率（Accuracy）或错误率（Error Rate）
  - 正确率是被正确处理的数据个数与所有被处理数据个数的比值

$$Accuracy(T) = \frac{|T_{Error<\varepsilon}|}{|T|}$$

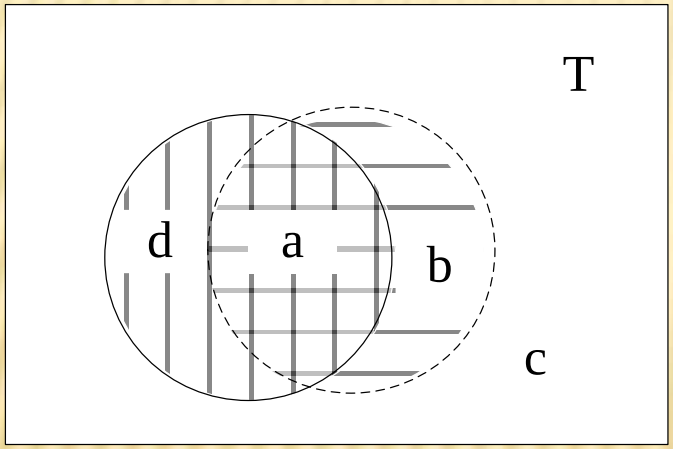
其中$T_{Error}<\varepsilon$表示被正确处理的数据，也就是误差足够小的数据

  - 错误率则是没有被正确处理的数据个数与所有被处理数据个数的比值

$$ErrorRate(T) = \frac{|T| - |T_{Error<\varepsilon}|}{|T|} = 1 - \frac{|T_{Error<\varepsilon}|}{|T|} = 1 - Accuracy(T)$$

# 度量学习结果有效性的指标

- 复合指标
  - 精度（Precision，或称为命中率，准确率）
  - 召回率（Recall，或称为覆盖率）




a：判定属于类且判定正确；
b：判定属于类且判定错误；
c：判定不属于类且判定正确；
d：判定不属于类且判定错误。

T=a+b+c+d

$$precision(T) = \frac{|a|}{|a+b|} \qquad recall(T) = \frac{|a|}{|a+d|} \qquad Accuracy(T) = \frac{|a+c|}{|a+b+c+d|}$$

# 度量学习结果有效性的指标

- $F_\beta$度量（Fβ-Measure）
  - $F_\beta$度量是精度和召回率的调和平均数（Harmonic Mean）

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

其中β是一个大于0的实数，表示精度相对于召回率的权重。

  - 最常用β=1，即$F_1$度量

$$F_1(T) = \frac{2\, precision(T) \times recall(T)}{precision(T) + recall(T)}$$

# 度量学习结果有效性的指标

- 多分类问题学习结果的评判
  - 对于测试集T，目标类别共有k个
- 宏平均法（Macro Average）
  - 思路
    - 先计算各个类别自身的精度和召回率，
    - 然后把各个类别的指标加在一起求算术平均值。
  - 宏平均精度

$$precision_{macro}(T) = \frac{1}{k}\sum_{i=1}^{k} precision_i(T)$$

  - 宏平均召回率

$$recall_{macro}(T) = \frac{1}{k}\sum_{i=1}^{k} recall_i(T)$$

# 度量学习结果有效性的指标

- 微平均法（Micro Average）
  - 把整个测试集看作单分类问题，一次性计算所有个体样本指标的平均值。
  - 微平均精度

$$precision_{micro}(T) = \frac{\sum_{i=1}^{k} |a_i|}{\sum_{i=1}^{k} |a_i| + \sum_{i=1}^{k} |b_i|}$$

  - 微平均召回率

$$recall_{micro}(T) = \frac{\sum_{i=1}^{k} |a_i|}{\sum_{i=1}^{k} |a_i| + \sum_{i=1}^{k} |d_i|}$$

# 6.2 决策树学习

- 决策树学习是应用最广的归纳推理算法之一。它是一种逼近离散值函数的方法。在这种方法中学习到的函数被表示为一颗决策树。学习得到的决策树也能再被表示为多个if-then规则，以提高可读性。

- 决策树学习算法有很多，比如ID3、C4.5、ASSISTANT等等。这些决策树学习方法搜索一个完整表示的假设空间，从而避免了受限假设空间的不足。决策树学习的归纳偏置是优先选择较小的树。

# 6.2.1 决策树表示法

- 决策树通过把实例从根节点排列(sort)到某个叶子节点来分类实例，叶子节点即为实例所属的分类。
- 树上的每一个节点说明了对实例的某个属性(attribute)的测试，并且该节点的每一个后继分枝对应于该属性的一个可能值。
- 分类实例的方法是
  - 从这颗树的根节点开始，测试这个节点指定的属性；
  - 然后按照给定实例的该属性值对应的树枝向下移动；
  - 然后这个过程再以新节点为根的子树上重复。

# 决策树示例

例子：在一个水果的分类问题中，采用的特征向量为：{颜色，尺寸，形状，味道}，其中：

颜色属性的取值范围：红，绿，黄

尺寸属性的取值范围：大，中，小

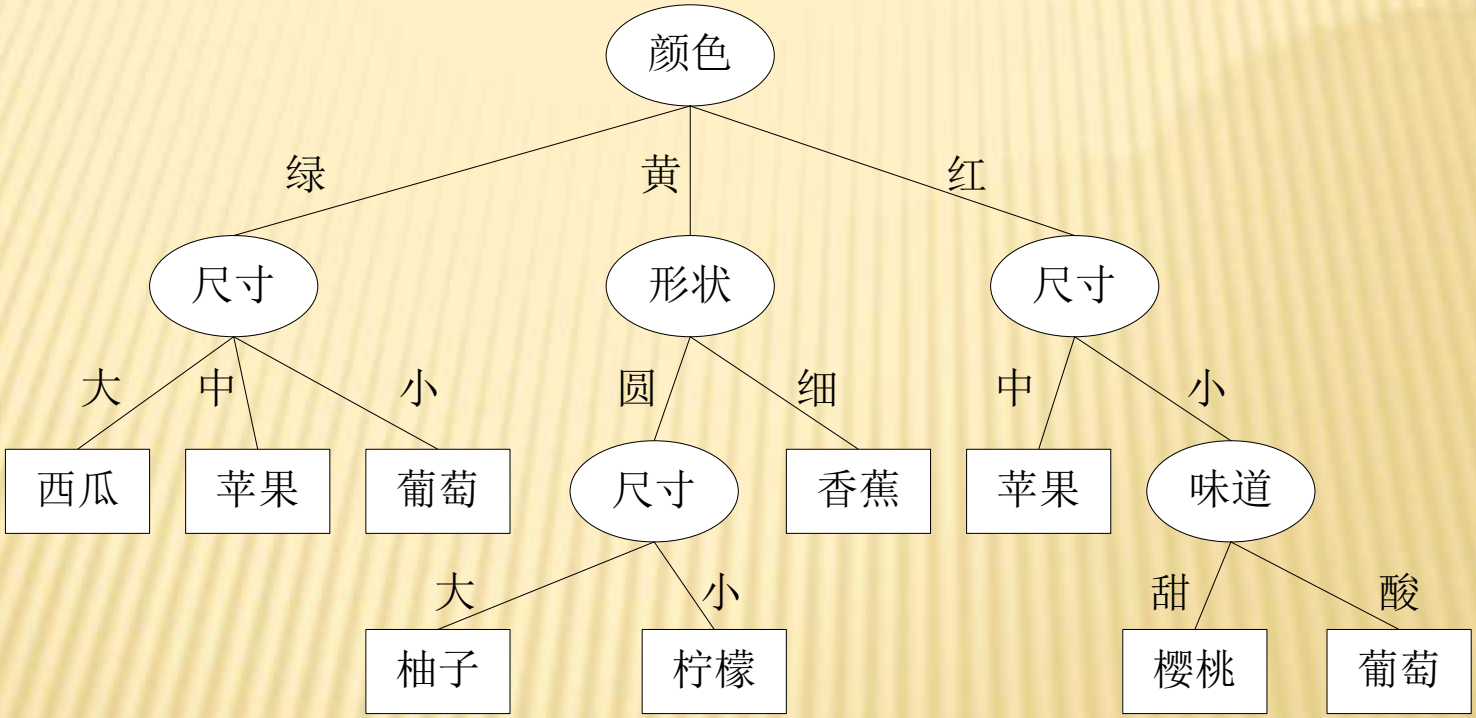
味道属性的取值范围：甜，酸

形状属性的取值范围：圆，细

样本集：一批水果，知道其特征向量及类别

问　题：一个新的水果，观测到了其特征向量，应该将其分类哪一类？

# 决策树示例



# 6.2.2 ID3算法

- 大多数已开发的决策树学习算法是一种核心算法（CLS算法）的变体。该算法采用自顶向下的贪恋搜索遍历可能的决策树空间。这种方法是 ID3 算法 (Quinlan 1986) 和后继的 C4.5(Quinlan 1993)的基础。

# ID3算法

- ID3是一种自顶向下增长树的贪婪算法
  - 在每个节点选取能最好分类样例的属性；
  - 继续这个过程指导这棵树能完美分类训练样例；
  - 或所有的属性都已被使用过。
- 构造过程是从“哪一个属性将在树的根节点被测试”这个问题开始。
  - 分类能力最好的属性被选作树的根节点的测试。
  - 然后为根节点属性的每个可能值产生一个分枝，并把训练样例排列到适当的分枝（也就是，样例的该属性值对应的分枝）之下。
  - 算法从不回溯重新考虑以前的选择。

# 熵（ENTROPY）

- 给定一个样本集D，假设其中的样本来自c个不同的类别，则样本集D的熵为

$$Entropy(D) = \sum_{i=1}^{c} P_i \log_2(\frac{1}{P_i}) = \sum_{i=1}^{c} -P_i \log_2(P_i)$$

其中$P_i$是数据集D中类别为i的样本所占得比例。

熵的最大可能值为$\log_2 c$，最小值为0。

定义0log0=0。

熵反映了D的纯净度，熵越小纯净度越高。

# 信息增益（INFORMATION GAIN）

- 属性A对于数据集D的信息增益Gain(D,A)
  - 由于使用该属性分割数据集D，而导致数据集D期望熵减少的程度。

$$Gain(D,A) = Entropy(D) - \sum_{v \in Values(A)} \left( \frac{|D_v|}{|D|} Entropy(D_v) \right)$$

  - Values(A)是属性A所有可能取值的集合。
  - 如A可以选择为颜色，则Values(A)={红，绿，黄}。
  - $D_v$是D中属性A的值为v的样本的子集，即$D_v$={d|d∈D,d的属性A的值为v}。
  - 在前面的例子中共有4个属性{颜色，尺寸，形状，味道}。

# 决策树结点划分三种情况

- 任意一个结点n，可以出现以下三种情况
  - 结点n中的样本属于同一类，即结点n绝对纯净。此时结点n不可进一步划分。
  - 结点n中的样本不属于同一类，但是不存在任何一个划分可以使其子结点的平均熵低于结点n。此时结点n不可进一步划分。
  - 可以用一个属性对结点n进行划分，从而使结点n的子结点具有更低的平均熵。此时结点n可以进一步划分。

# 确定叶节点

- 一个策略是:
  - 对于每一个可以进一步划分的结点都进行划分，直到得到一个不可划分的子结点，并将该子结点定为叶结点。
  - 该策略可取吗?
- 
- 决策树学习不能绝对追求叶结点的纯净度，否则会产生
- 过学习（过度拟合）。
- 叶结点的纯净度也不能过低，过低则是欠学习。

  - 我们应该在过度拟合与欠学习之间寻求合理的平衡。

# 克服过学习与欠学习的方法

- 测试集方法
  - 将数据样本集合分为训练集与确认集。
  - 根据训练集构建决策树，每展开一层子结点，就将其设为叶结点，并得到一棵决策树，然后采用确认集对所得决策树的分类性能进行统计。
  - 重复上述过程，可以得到决策树在确认集上的学习曲线。
  - 根据学习曲线，选择在确认集上性能最佳的决策树为最终的决策树。
- 阈值方法
  - 在决策树开始训练之前，先设定一个阈值作为终止学习的条件。
  - 在学习过程中如果一个结点满足了终止条件就停止划分，并将其作为叶结点。
  - 终止条件：
    - 可以选择为信息增益小于某阈值；
    - 或结点中样本数量占全体样本数量的比例小于某阈值。

# 决策树的修剪

- 在实践中常用的规则后修剪（Rule Post-Pruning）方法如下：
  - 第一步 从训练集学习决策树，允许过度拟合。
  - 第二步 将决策树转化为等价的规则集合。从根结点到叶子结点的一条路径就是一条规则。
  - 第三步 对每一条规则，如果删除该规则中的一个前件不会降低该规则的分类精度，则可删此前件。
  - 第四步 按照修剪后规则的精度对所有规则排序，最后按照此顺序来选择规则进行分类。

# 叶结点的类别

- 对于叶结点n，如果在该结点对应的样本中，属于第$i$类的样本数量最多，则判该叶结点为第$i$类。

# ID3算法伪代码

- 第一步 创建根结点。
- 第二步 根结点数据集为初始数据集。
- 第三步 根结点属性集包括全体属性。
- 第四步 当前结点指向根结点。
- 第五步 在当前结点的属性集和数据集上，计算所有属性的信息增益。
- 第六步 选择信息增益最大的属性A作为当前结点的决策属性。
- 第七步 如果最大信息增益小于等于0，则当前结点是叶子结点，标定其类别，并标记该结点已处理。执行第十四步。否则执行第八步。
- 第八步 对属性A的每一个可能值生成一个新结点。
- 第九步 把当前结点作为新结点的父结点。
- 第十步 从当前结点数据集中选取属性A等于某个值的数据，作为该值对应新结点的数据集。
- 第十一步 从当前结点属性集中去除属性A，然后作为新结点的属性集。
- 第十二步 如果新结点数据集或者属性集为空，则该新结点是叶子结点，标定其类别，并标记该结点已处理。
- 第十三步 标记当前结点已处理。
- 第十四步 令当前结点指向一个未处理结点。如果无未处理结点则算法结束。否则执行第五步。

# ID3算法特点

- ID3算法的假设空间就是所有可能决策树的集合。
  - 也是一个关于现有属性的有限离散值函数的完整空间。
- ID3算法运用爬山法搜索假设空间，
  - 并未彻底地搜索整个空间，而是当遇到第一个可接受的树时，就终止了。
  - ID3算法实际上用信息增益度量作启发式规则，指导爬山搜索。
- ID3的搜索策略是：
  - 优先选择较短的树，而不是较长的；
  - 优先选择短的树，即复杂度小的决策树，更符合奥坎姆剃刀原则。
  - 也就是优先选择更简单的假设。
- 基本的ID3算法不回溯，对已经做过的选择不再重新考虑。
  - ID3算法收敛到局部最优解，而不是全局最优。
  - 可以对ID3算法得到的决策树进行修剪，增加某种形式的回溯，从而得到更优解。

# 过度拟合问题

- 定义6.3
  - 给定一个假设空间H和一个训练数据集D。对于一个假设h（h∈H），如果存在其它的假设h′（h′∈H），使得在训练数据集D上h的错误率小于h′的错误率，但是在全体可能数据集合上h的错误率大于h′的错误率。
  - 那么假设h就过度拟合（Overfit）了训练数据D。

# 过度拟合问题

- 当训练数据采样太少，不能完全覆盖真实分布时，过度拟合很容易发生。
- 坏处：
  - 严重影响模型的泛化能力，降低模型的实用性能。
- 决策树学习中的过度拟合表现
  - 决策树结点过多，分支过深，
  - 对训练数据可以完美分类，
  - 但是对于非训练数据则精度下降。

# 信息增益度量的问题

- 信息增益度量会偏向于有较多可能值的属性。
  - 特别是当某属性可能值的数目大大多于类别数目时，该属性就会有很大的信息增益。
- 原因
  - 因为太多的可能值把训练数据分割成了非常小的空间。
  - 在每一个小空间内，数据都非常纯净，甚至数据完全一致，熵为0。
  - 这样与未分割之前比，信息增益必然非常大。
- 然而这样的分割显然掩盖了其它有用信息，并未反映真实的数据分布，所以对其它数据就有非常差的分类结果

# 增益比率（GAIN RATIO）度量

- 思路
  - 在信息增益度量的基础上加一个惩罚项来抑制可能值太多的属性
- 定义

$$GainRatio(D,A)=\frac{Gain(D,A)}{SplitInformation(D,A)}$$

- 属性A对数据集D的分裂信息定义为

$$SplitInformation(D,A)=\sum_{v\in Values(A)}-\frac{|D_v|}{|D|}\log_2(\frac{|D_v|}{|D|})$$

- 特点：
  - 对于可能值比较多的属性，由于其分裂信息也比较大，所以最终的增益比率反而可能减小。
  - 但是当分裂信息过小，增益比率会过大，甚至无定义。

# MANTARAS基于距离的度量

- 思路
  - 对于数据集假设存在一个理想划分，使得每一个数据都被正确分类。
  - 那么我们定义一个距离，用以度量其它划分到这个理想划分之间的差距。
  - 于是距离越小的划分自然是越好的划分

# MANTARAS基于距离的度量

- 令A表示把数据集D分为n个子集（类别）的一个划分，B表示把数据集D分为m个子集（类别）的一个划分。
- 划分B对于划分A的条件熵

$$Entropy(B \mid A) = \sum_{i=1}^{n}\sum_{j=1}^{m} -P(A_i B_j)\log_2\left(\frac{P(A_i B_j)}{P(A_i)}\right)$$

- 划分A和划分B的联合熵

$$Entropy(AB) = \sum_{i=1}^{n}\sum_{j=1}^{m} -P(A_i B_j)\log_2 P(A_i B_j)$$

  - 其中$P(A_iB_j)$表示一个数据既在划分A中属于$A_i$类，又在划分B中属于$B_j$类的概率。

# MANTARAS基于距离的度量

- Mantaras定义两个划分A、B间的距离为

$$d(A,B) = Entropy(B \mid A) + Entropy(A \mid B)$$

- 经过规一化之后的距离为

$$d_N(A,B) = \frac{d(A,B)}{Entropy(AB)}$$

- 两种距离定义都满足距离公理
- 特点：
  - 不会偏向可能值较多的属性；
  - 也不会出现增益比率度量的缺陷。

问题：在决策树学习中如何使用上述距离？

本章待续……

# 决策树的修剪

- 修剪的一般原则是使决策树整体的精度提高，或者错误率降低。
- 在实践中常用的规则后修剪（Rule Post-Pruning）方法如下：
  - 第一步 从训练数据学习决策树，允许过度拟合。
  - 第二步 将决策树转化为等价的规则集合。从根结点到叶子结点的一条路径就是一条规则。
  - 第三步 对每一条规则，如果删除该规则中的一个前件不会降低该规则的估计精度，则可删此前件。
  - 第四步 按照修剪后规则的估计精度对所有规则排序，最后按照此顺序来应用规则进行分类。