人工智能——样例学习I

饶洋辉
计算机学院,
中山大学
raoyangh@mail.sysu.edu.cn
http://cse.sysu.edu.cn/node/2471

样例学习：分类

- 预测离散型变量
  - 首先基于一个包含x值，以及离散型的真实y值的训练数据集构建分类模型；然后将该模型用来预测新的只包含x值的测试数据集的y值。
- 典型应用
  - 精准营销 (电子商务)
  - 信用审批 (银行/金融)
  - 医学诊断 (健康医疗)
  - 欺诈/入侵检测 (互联网)

评测指标

- 准确率
- 速度
  - 构建分类模型的时间（训练速度）
  - 使用分类模型的时间（预测速度）
- 鲁棒性
  - 模型在处理噪音和缺失值方面的能力
- 可扩展性
  - 模型用在更大规模的数据集上的能力
- 可解释性

评测指标

评测指标

距离

- 向量 x 与 y 之间的欧式距离用如下公式计算:

$$d(\mathbf{x},\mathbf{y}) = \sqrt{\sum_{k=1}^{n}(x_k - y_k)^2}$$

其中

- n 是向量的维度
- x_k 和 y_k 是 x 与 y 的第 k 项元素

距离

- 欧式距离的衡量公式可以一般化成闵可夫斯基距离（*Minkowski* distance metric）

$$d(\mathbf{x},\mathbf{y}) = \left(\sum_{k=1}^{n} | x_k - y_k |^r \right)^{\frac{1}{r}}$$

- 闵可夫斯基距离的三个常见的例子：
  - r=1: 街区距离（City block distance, L_1 norm）
  - r=2: 欧式距离（Euclidean distance, L_2 norm）
  - r=∞: 极大距离（Supremum distance, L_{max} or L_∞ norm），该距离是两向量对应元素之间差距最大的距离

距离

$$x_i = (x_i^{(1)}, x_i^{(2)}, \cdots, x_i^{(n)})^{\mathrm{T}}$$

- Lp距离：

$$L_p(x_i, x_j) = \left(\sum_{l=1}^{n} | x_i^{(l)} - x_j^{(l)} |^p \right)^{\frac{1}{p}}$$

  - 欧式距离：

$$L_2(x_i, x_j) = \left(\sum_{l=1}^{n} | x_i^{(l)} - x_j^{(l)} |^2 \right)^{\frac{1}{2}}$$

  - 曼哈顿距离：

$$L_1(x_i, x_j) = \sum_{l=1}^{n} | x_i^{(l)} - x_j^{(l)} |$$

  - $L\infty$距离：

$$L_\infty(x_i, x_j) = \max_l | x_i^{(l)} - x_j^{(l)} |$$

距离

- 四个向量的 x 与 y 坐标如下：
  - p1 = <0, 2>
  - p2 = <2, 0>
  - p3 = <3, 1>
  - p4 = <5, 1>

距离

L_1	p1	p2	p3	p4
p1	0.0	4.0	4.0	6.0
p2	4.0	0.0	2.0	4.0
p3	4.0	2.0	0.0	2.0
p4	6.0	4.0	2.0	0.0

L_2	p1	p2	p3	p4
p1	0.0	2.8	3.2	5.1
p2	2.8	0.0	1.4	3.2
p3	3.2	1.4	0.0	2.0
p4	5.1	3.2	2.0	0.0

L_∞	p1	p2	p3	p4
p1	0.0	2.0	3.0	5.0
p2	2.0	0.0	1.0	3.0
p3	3.0	1.0	0.0	2.0
p4	5.0	3.0	2.0	0.0

k-近邻分类

- 随着人们生活水平不断的提高,红酒越来越受到人们的喜爱。红酒的产量越来越大，然而红酒品质鉴定的手段还是仅靠品酒师的人工品尝打分来判定红酒质量的好坏，显然这种鉴定方式难以满足当今市场的需求。现在有不少学者运用一些机器学习的算法来对红酒质量进行预测研究，使得红酒品质鉴定的速度得到大幅提升并且有着较高的准确率。

k-近邻分类

- 对于红酒品质的分类，可以基于红酒的理化指标(例如：酒精的浓度、pH值、糖的含量、非挥发性酸含量、 挥发性酸含量、柠檬酸含量等)作为特征，建立分类模型，然后对红酒品质进行预测。本案例中，我们将使用UCI数据库中的 Wine Quality Data Set 数据集，利用k-近邻分类算法来进行红酒品质的分类。

k-近邻分类

- 我们使用一份包含1599个样本的关于葡萄牙的Vinho Verde葡萄酒数据集。 每个样本包含12个变量，其中最后一个变量quality为预测变量。

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5

- k-近邻分类算法：所有属性（变量）同等重要。
- k指的是选取的和待预测样本距离最近的训练样本数。

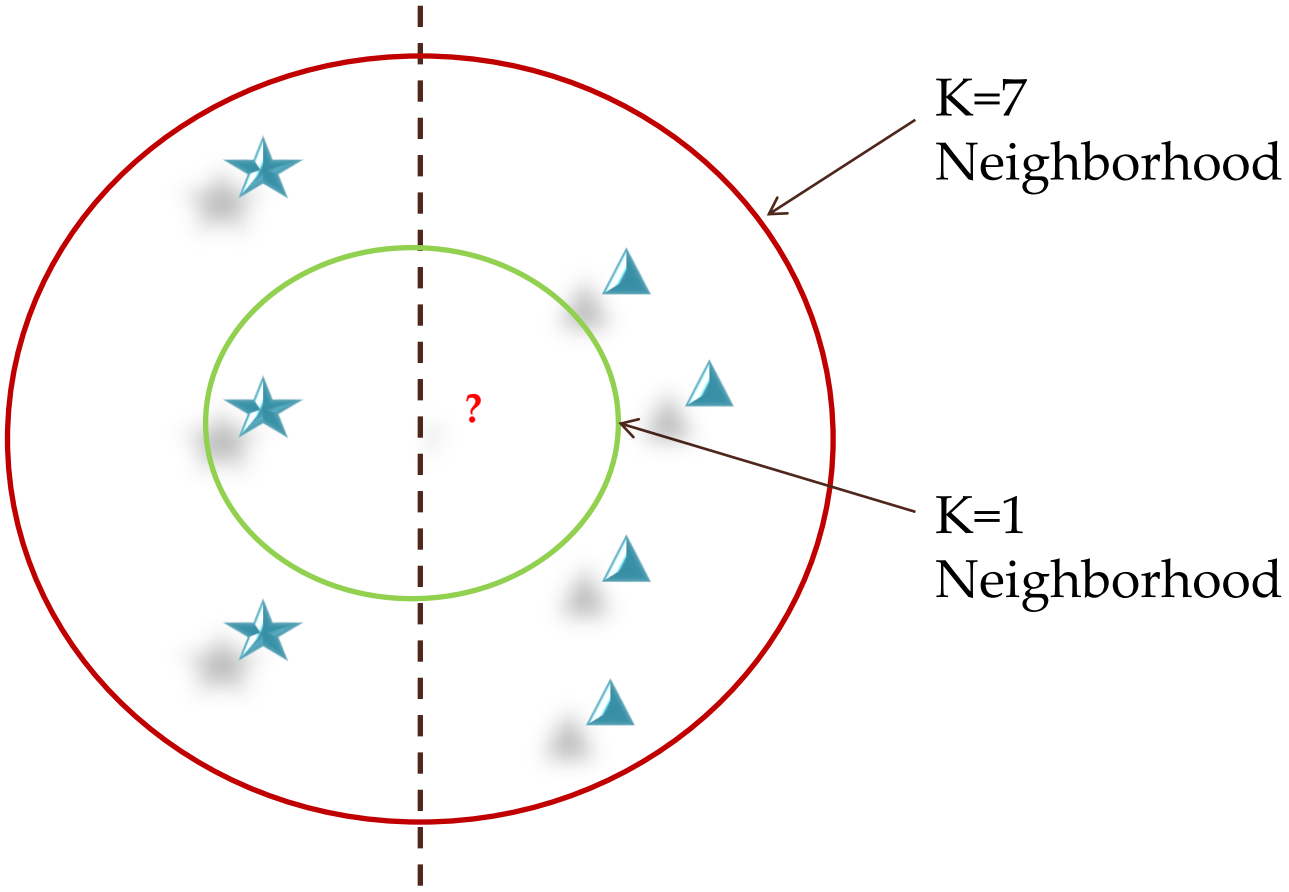
k-近邻分类

- 0、1、2、3为训练集
- 4为测试集
- k = 1, 2, 3, 4
- 采用街区距离

- d(4, 0) = 0.0
- d(4, 1) = 49.1
- d(4, 2) = 25.7
- d(4, 3) = 37.6

变量名称	含义说明
fixed acidity	非挥发性酸含量
volatile acidity	挥发性酸含量
citric acid	柠檬酸
residual sugar	糖含量
chlorides	氯化物
free sulfur dioxide	游离二氧化硫
total sulfur dioxide	总二氧化硫
density	密度
pH	酸碱度
sulphates	硫酸盐
alcohol	酒精浓度
quality	品质， 为预测变量

K-近邻分类



K-近邻分类

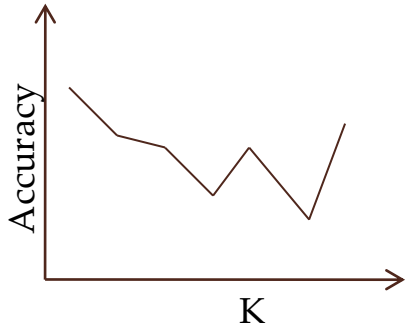
- 工作原理
  - 存在一个样本数据集合，也称作训练样本集，并且样本集中每个数据都存在标签，即我们知道样本集中每个数据与所属分类的对应关系。
  - 输入没有标签的新数据后，将新数据的每个特征与样本集中数据对应的特征进行比较，然后算法提取样本集中特征最相似数据（最近邻）的分类标签。
  - 一般来说，只选择样本数据集中前K个最相似的数据。K一般不大于20，最后，选择K个最近邻中出现次数最多的分类，作为新数据的分类。

K-近邻分类

- 优点
  - 精度高
  - 对异常值不敏感
  - 无数据输入假定
- 缺点
  - 时间复杂度高
  - 空间复杂度高
- 适用数据范围
  - 离散型和连续型

K-近邻分类

- K值确定
  - Non-monotonous impact on accuracy
  - Too Big vs. Too Small
  - Rule of thumbs (经验法)

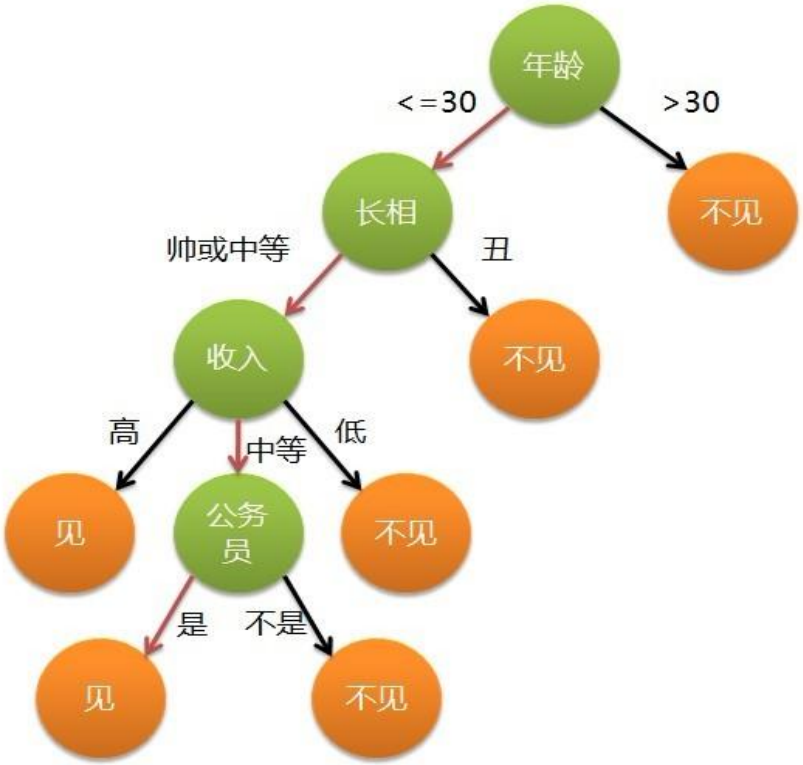


- 特征的选择
  - Different features may have different impact …
- 距离函数确定
  - There are many different ways to measure the distance.
  - Euclidean, Manhattan …
- 复杂度
  - Need to calculate the distance between X′ and all training data.
  - In proportion to the size of the training data.

决策树

- 套用俗语，决策树分类的思想类似于找对象。现想象一个女孩的母亲要给这个女孩介绍男朋友，于是有了下面的对话：
- 女儿：多大年纪了？
  母亲：26。
  女儿：长的帅不帅？
  母亲：挺帅的。
  女儿：收入高不？
  母亲：不算很高，中等情况。
  女儿：是公务员不？
  母亲：是，在税务局上班呢。
  女儿：那好，我去见见。

决策树



决策树

- 一种树状结构的分类模型
- 中间节点：表示基于某个属性进行训练数据集的划分，中间节点中指明该属性的取值
- 分支：用于展示某种划分方式的输出
- 叶子节点：表示按照当前分支得到的训练数据的类分布（class distribution）

决策树

年龄	收入	学生?	信用等级?	是否买电脑
<=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

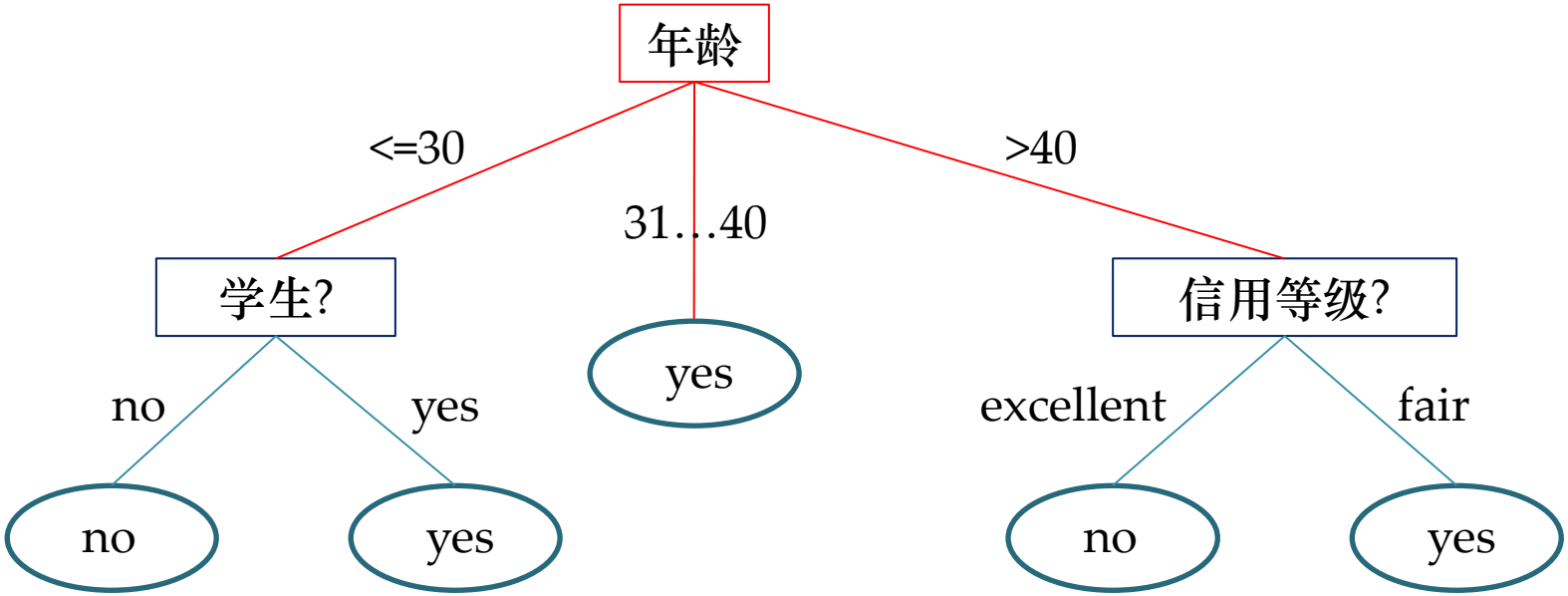
决策树

年龄	收入	学生?	信用等级?	是否买电脑
<=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

决策树

年龄	收入	学生?	信用等级?	是否买电脑
<=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

决策树



决策树

- 建模阶段
  - Tree construction (建树)
    - 首先，所有训练样本都位于根节点位置
    - 基于一定的指标(如：信息增益，基尼系数等)选择属性
    - 根据选择的属性，递归地划分训练样本
  - Tree pruning (剪枝)
    - 识别并删除异常值和噪声影响较大的分支
- 预测阶段
  - 使用构建的树模型预测未知样本

算法

- 基础算法(一种贪心算法)
  - 根据分治的思想，用自顶向下的方法，递归建树。
  - 属性应当是离散的(如果是连续型数据，需要先进行离散化)
  - 首先，所有训练样本都位于根节点位置
  - 基于统计学指标或启发式的方法来对属性进行选择（例如，信息增益，基尼系数等）。
  - (训练) 根据选择的属性，递归地划分训练样本。

算法

- 停止划分的条件
  - 被分到同一个节点内的所有样本都是同一个类别（相同label/class）
  - 所有的属性都已经被用于之前的划分，没有属性可以继续划分——采用多数投票的方法决定该叶子节点的列表
  - 无训练数据

决策树

- 与决策树相关的重要算法包括：
  - CLS, ID3，C4.5，CART
- 算法的发展过程
  - Hunt, Marin和Stone于1966年研制的CLS学习系统，用于学习单个概念。
  - 1979年，Quinlan给出ID3算法，并在1983年和1986年对ID3进行了总结和简化，使其成为决策树学习算法的典型。
  - Schlimmer和Fisher于1986年对ID3进行改造，在每个可能的决策树节点创建缓冲区，使决策树可以递增式生成，得到ID4算法。
  - 1988年，Utgoff在ID4基础上提出了ID5学习算法，进一步提高了效率。
  - 1993年，Quinlan 进一步发展了ID3算法，改进成C4.5算法。
  - 另一类决策树算法为CART，与C4.5不同的是，CART由二元逻辑问题生成，每个树节点只有两个分枝，分别包括学习实例的正例与反例。

决策树

CLS算法

- CLS（Concept Learning System）算法
  - CLS是早期的决策树学习算法。它是许多决策算法的基础
- CLS的基本思想
  - 从一棵空决策树开始，选择某一属性（分类属性）作为测试属性。该测试属性对应决策树中的决策结点。根据该属性的值的不同，可将训练样本分成相应的子集：
    - 如果该子集为空，或该子集中的样本属于同一个类，则该子集为叶结点；
    - 否则该子集对应于决策树的内部结点，即测试结点，需要选择一个新的分类属性对该子集进行划分，直到所有的子集都为空或者属于同一类。

决策树

CLS算法

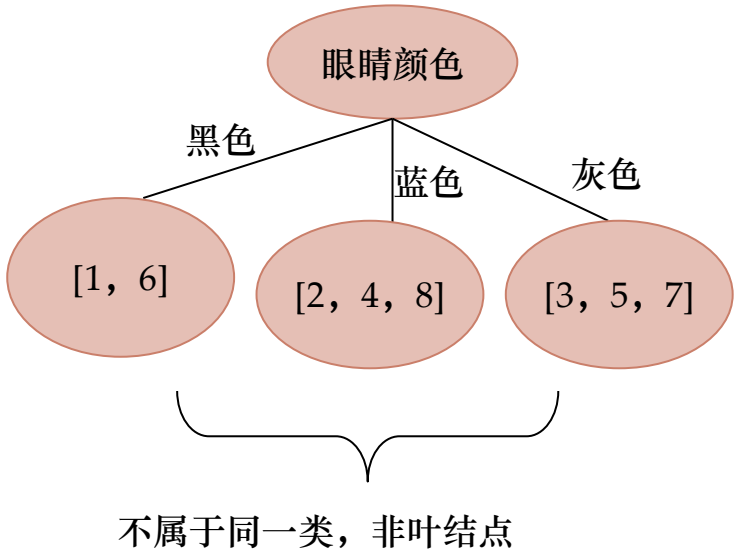
人员	眼睛颜色	头发颜色	所属人种
1	黑色	黑色	黄种人
2	蓝色	金色	白种人
3	灰色	金色	白种人
4	蓝色	红色	白种人
5	灰色	红色	白种人
6	黑色	金色	混血
7	灰色	黑色	混血
8	蓝色	黑色	混血

决策树

CLS算法

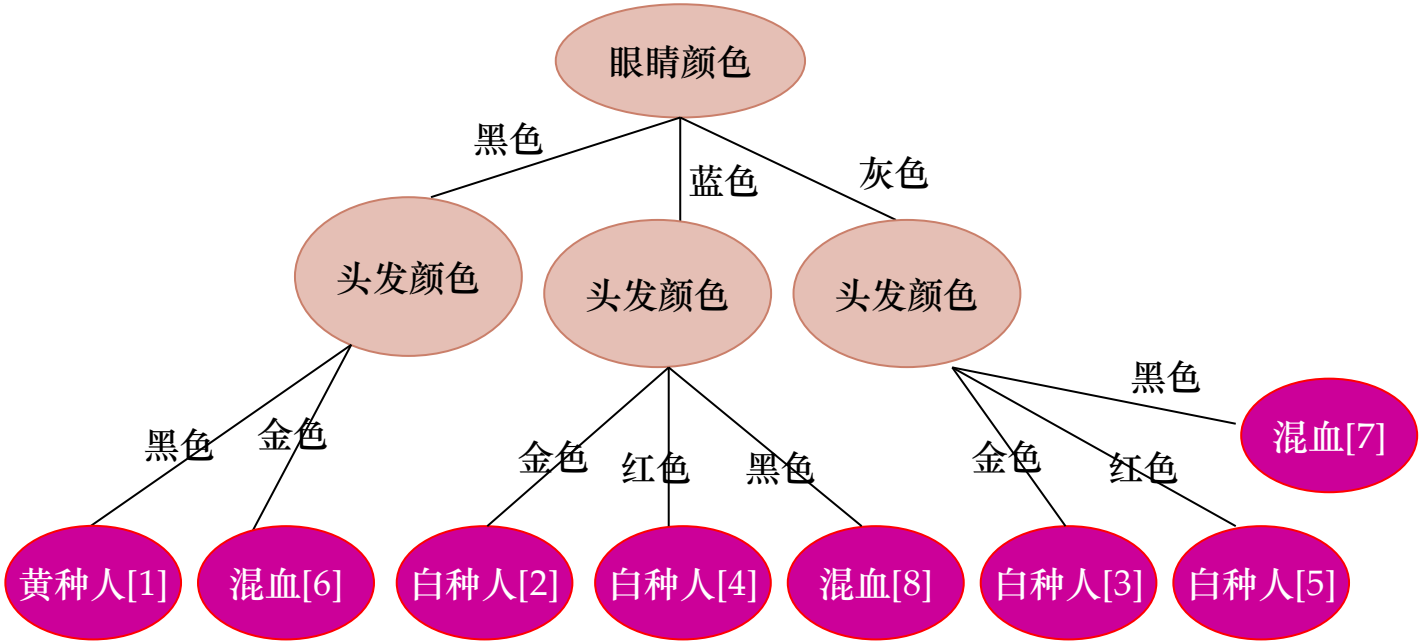
决策树的构建

人员	眼睛颜色	头发颜色	所属人种
1	黑色	黑色	黄种人
2	蓝色	金色	白种人
3	灰色	金色	白种人
4	蓝色	红色	白种人
5	灰色	红色	白种人
6	黑色	金色	混血
7	灰色	黑色	混血
8	蓝色	黑色	混血



决策树

CLS算法


眼睛颜色
黑色
蓝色
灰色
头发颜色
头发颜色
头发颜色
黑色
金色
金色
红色
黑色
黑色
金色
红色
混血[7]
黄种人[1]
混血[6]
白种人[2]
白种人[4]
混血[8]
白种人[3]
白种人[5]

决策树

CLS算法

- 步骤：
  - 生成一颗空决策树和一个训练样本属性表;
  - 若训练样本集 T 中所有的样本都属于同一类，则生成结点 T ，并终止学习算法；否则
  - 根据某种策略从训练样本属性表中选择属性 A 作为测试属性，生成测试结点A;
  - 若A的取值为$v_1, v_2, \ldots, v_m$，则根据 A 的取值的不同，将 T 划分成 m个子集$T_1, T_2, \ldots, T_m$;
  - 从训练样本属性表中删除属性A;
  - 对每个子集递归调用CLS

决策树

CLS算法

- CLS算法问题：
  - 在步骤3中，根据某种策略从训练样本属性表中选择属性*A*作为测试属性。没有规定采用何种测试属性。实践表明，测试属性集的组成以及测试属性的先后对决策树的学习具有举足轻重的影响。

决策树

CLS算法

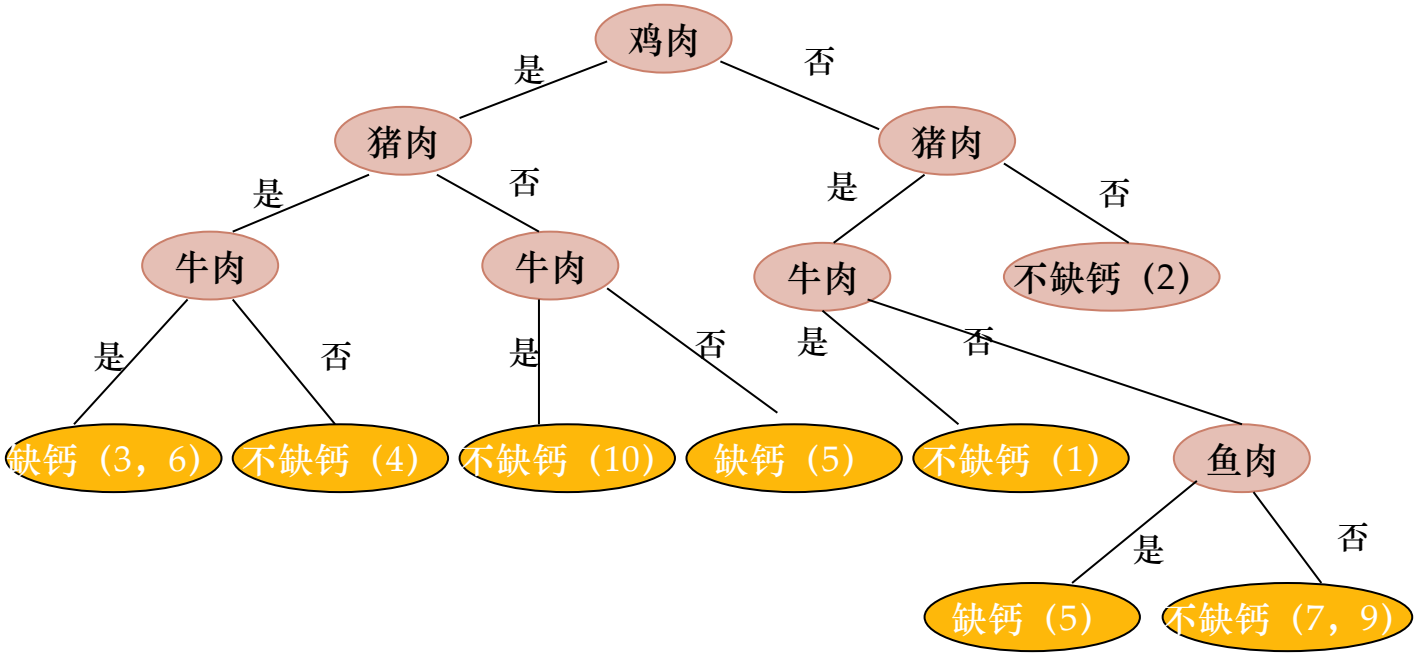
学生膳食结构和缺钙调查表

学生	鸡肉	猪肉	牛肉	羊肉	鱼肉	鸡蛋	青菜	番茄	牛奶	健康情况
1	0	1	1	0	1	0	1	0	1	不缺钙
2	0	0	0	0	1	1	1	1	1	不缺钙
3	1	1	1	1	1	0	1	0	0	缺钙
4	1	1	0	0	1	1	0	0	1	不缺钙
5	1	0	0	1	1	1	0	0	0	缺钙
6	1	1	1	0	0	1	0	1	0	缺钙
7	0	1	0	0	0	1	1	1	1	不缺钙
8	0	1	0	0	0	1	1	1	1	缺钙
9	0	1	0	0	0	1	1	1	1	不缺钙
10	1	0	1	1	1	1	0	1	1	不缺钙

决策树

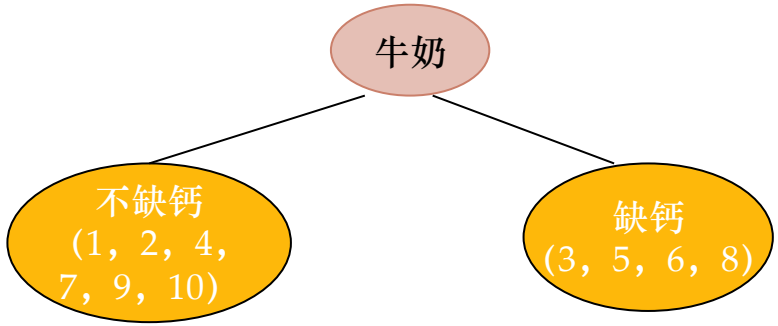
CLS算法

采用不同的测试属性及其先后顺序将会生成不同的决策树



决策树

CLS算法



决策树

ID3算法

- ID3算法是一种经典的决策树学习算法，由Quinlan于1979年提出。
- ID3算法主要针对属性选择问题。是决策树学习方法中最具影响和最为典型的算法。
- 该方法使用信息增益度选择测试属性。
- 当获取信息时，将不确定的内容转为确定的内容，因此信息伴着不确定性。
- 从直觉上讲，小概率事件比大概率事件包含的信息量大。如果某件事情是“百年一见”则肯定比“习以为常”的事件包含的信息量大。

- **如何度量信息量的大小?**

算法

- 如何衡量属性的重要性（importance）？

年龄	收入	学生?	信用等级?	是否买电脑
<=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

信息论

- 假设要为投掷一个8面骰子的结果进行编码. 需要多少个比特？ (bit)

信息论

- 假设要为投掷一个8面骰子的结果进行编码. 需要多少个比特？ (bit)

$$3bits = \log_2 8 = -\sum_{i=1}^{8} \frac{1}{8} \log_2 \frac{1}{8} = -\sum_{i=1}^{8} p(i) \log_2 p(i) = H(X)$$

信息论

- 假设要为投掷一个8面骰子的结果进行编码. 需要多少个比特？ (bit)

$$3bits = \log_2 8 = -\sum_{i=1}^{8} \frac{1}{8} \log_2 \frac{1}{8} = -\sum_{i=1}^{8} p(i) \log_2 p(i) = H(X)$$

- 如果我们希望将投掷这个8面骰子的结果通过某种方式发送给别人，最有效的方式就是将这一信息进行二进制编码【000 – 111】

信息论

- Entropy (熵)
  - represent the expectation of uncertainty for a random variable (用来衡量离散变量的不确定性，如抛硬币、掷骰子)

$$H(X) = -\sum_{x \in X} p(x) \log_2 p(x)$$

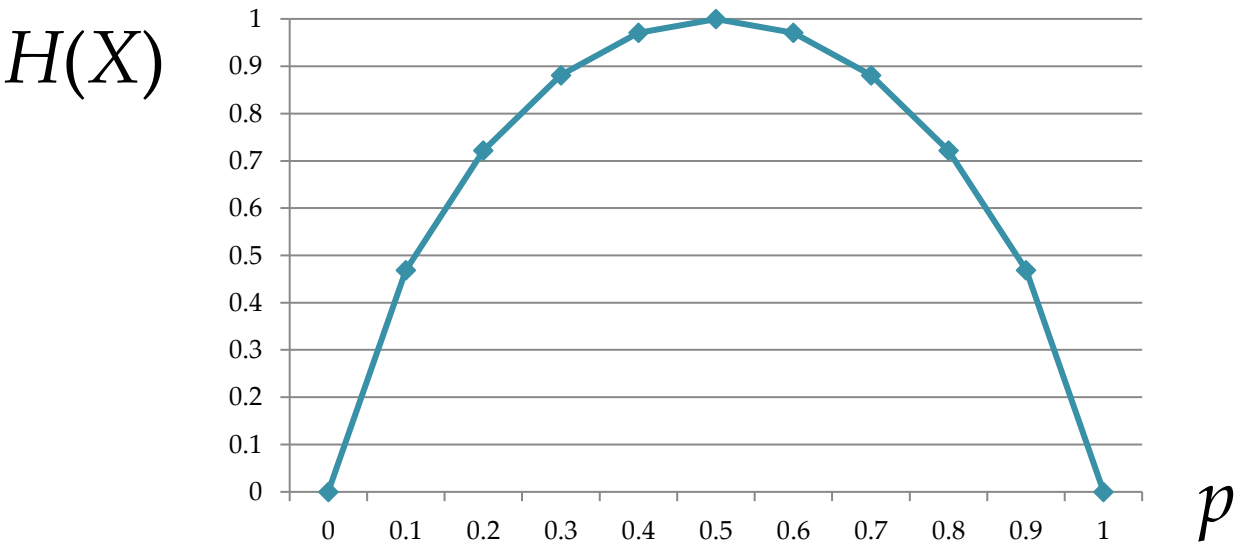
$$= \sum_{x \in X} p(x) \log_2 \frac{1}{p(x)}$$

$$= E\left(\log_2 \frac{1}{p(X)} \right)$$

信息论

- $P(X=1) = p$, $P(X=0) = 1-p$
  - 假设抛一枚硬币，正面朝上的概率为p，反面朝上的概率为$1-p$，则抛这枚硬币所得结果的不确定性（熵值）是p的下述函数：

$$H(X) = -p\log_2 p - (1-p)\log_2(1-p)$$



信息论

- 条件/联合熵

条件熵：
$$H(Y \mid X) = \sum_{x \in X} p(x) H(Y \mid X = x)$$
$$= \sum_{x \in X} p(x) \left[-\sum_{y \in Y} p(y \mid x) \log_2 p(y \mid x) \right]$$
$$= -\sum_{x \in X} \sum_{y \in Y} p(x, y) \log_2 p(y \mid x)$$
$$= -\sum_{x \in X} \sum_{y \in Y} p(x) p(y \mid x) \log_2 p(y \mid x)$$

联合熵：
$$H(X, Y) = -\sum_{x \in X} \sum_{y \in Y} p(x, y) \log_2 p(x, y)$$

信息论

$$
\begin{aligned}
H(X,Y) &= -E_{p(x,y)} \log_2 p(x,y) \\
&= -E_{p(x,y)} \left(\log_2 (p(x) p(y \mid x)) \right) \\
&= -E_{p(x,y)} \left(\log_2 p(x) + \log_2 p(y \mid x) \right) \\
&= -E_{p(x)} \log_2 p(x) - E_{p(x,y)} \log_2 p(y \mid x) \\
&= H(X) + H(Y \mid X)
\end{aligned}
$$

两个离散变量X和Y的联合熵（即，联合出现的不确定性）
= X的熵 + 给定X，出现Y的条件熵
= X的不确定性 + 给定X，出现Y的不确定性

信息论

- Mutual information (互信息)

因为：$H(X,Y)=H(X)+H(Y\mid X)=H(Y)+H(X\mid Y)$

所以：$H(Y)-H(Y\mid X)=H(X)-H(X\mid Y)=I(X;Y)$

两个离散变量X和Y的互信息$I(X;Y)$
衡量的是这两个变量之间的相关度

一个连续变量X的不确定性，用方差$Var(X)$来度量
一个离散变量X的不确定性，用熵$H(X)$来度量
两个连续变量X和Y的相关度，用协方差或相关系数来度量
两个离散变量X和Y的相关度，用互信息$I(X;Y)$来度量

基于信息增益的ID3模型

- Class label: 是否买电脑=“yes/no”
- 用字母D表示类标签，字母A表示每个属性
- $H(D)$=0.940

14个训练样本中，9个买了电脑

$$H(D) = -\frac{9}{14}\log_2\frac{9}{14} - (1-\frac{9}{14})\log_2(1-\frac{9}{14})$$

- $H(D|A=\text{“年龄”})$=0.694

$$H(D|A="\text{年龄}") = \frac{5}{14}\times\left(-\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5}\right)$$

$$+\frac{4}{14}\times\left(-\frac{4}{4}\log_2\frac{4}{4} - \frac{0}{4}\log_2\frac{0}{4}\right) + \frac{5}{14}\times\left(-\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5}\right)$$

基于信息增益的ID3模型

计算D (类标签) 和A (每个属性)的互信息

- $H(D)$=0.940
- $H(D|A=\text{"年龄"})$=0.694

$$g(D,A) = I(D;A) = H(D) - H(D|A)$$

- $g(D,A=\text{“年龄”})$=0.246
- $g(D,A=\text{“收入”})$=0.029
- $g(D,A=\text{“学生?”})$=0.151
- $g(D,A=\text{“信用等级?”})$=0.048

基于信息增益的ID3模型

- 类标签: 是否买电脑=“yes/no”
- 对D和每个属性A，计算互信息(mutual information)
- $H(D)$=0.940
- $H(D|A=\text{“年龄”})$=0.694
- $g(D,A=\text{“年龄”})$=0.246
- $g(D,A=\text{“收入”})$=0.029
- $g(D,A=\text{“学生?”})$=0.151
- $g(D,A=\text{“信用等级?”})$=0.048

“年龄”这个属性的条件熵最小（等价于信息增益最大），因而首先被选出作为根节点

$$g(D,A) = H(D) - H(D|A)$$

基于信息增益的ID3模型

对于下述数据集，采用ID3算法会得到哪个属性最重要?

用户ID	年龄	收入	学生?	信用等级?	是否买电脑
u1	<=30	high	no	fair	no
u2	<=30	high	no	excellent	no
u3	31…40	high	no	fair	yes
u4	>40	medium	no	fair	yes
u5	>40	low	yes	fair	yes
u6	>40	low	yes	excellent	no
u7	31…40	low	yes	excellent	yes
u8	<=30	medium	no	fair	no
u9	<=30	low	yes	fair	yes
u10	>40	medium	yes	fair	yes
u11	<=30	medium	yes	excellent	yes
u12	31…40	medium	no	excellent	yes
u13	31…40	high	yes	fair	yes
u14	>40	medium	no	excellent	no

基于增益率的C4.5模型

- 信息增益（Information gain）的衡量容易偏向那些有大量值的属性
- C4.5 (ID3的一个改进版) 使用了增益率（Gain ratio）克服上述问题（对信息增益正则化）
- 每次选取最大增益率的属性进行划分

基于增益率的C4.5模型

- $\text{GainRatio}_A(D)=\text{Gain}_A(D)/\text{SplitInfo}_A(D)$

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2(\frac{|D_j|}{|D|})$$

- $\text{GainRatio}_{A=\text{“}income\text{”}}(D)=?$

基于增益率的C4.5模型

- $\text{GainRatio}_A(D)=\text{Gain}_A(D)/\text{SplitInfo}_A(D)$

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2(\frac{|D_j|}{|D|})$$

- $\text{GainRatio}_{A=\text{“}income\text{”}}(D)=?$

$$SplitInfo_{A="income"}(D)$$

$$= -\frac{4}{14} \times \log_2(\frac{4}{14}) - \frac{6}{14} \times \log_2(\frac{6}{14}) - \frac{4}{14} \times \log_2(\frac{4}{14})$$

$$= 0.926$$

- $\text{GainRatio}_{A=\text{“}income\text{”}}(D)=0.029/0.926=0.031$

基于Gini指数的CART模型

- 如果一个数据集D包含来自n个类的样本，那么基尼指数，$gini(D)$定义如下：

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

p_j 是类j 在 D中的相对频率。

- 如果 n=2, 那么 $gini(D) = 2p(1-p)$

基于Gini指数的CART模型

- 如果一个数据集 D 被分成两个子集 D_1 和 D_2 大小分别为 N_1 和N_2，数据包含来自 n 个类的样本，则基尼指数 $\text{gini}_{\text{split}}(D)$ 定义如下

$$gini_{split}(D) = \frac{N_1}{N} gini(D_1) + \frac{N_2}{N} gini(D_2)$$

- 具有最小 $\text{gini}_{\text{split}}(D)$的属性被选为分裂节点的属性（对每个属性，需要遍历所有可能的分裂位置点）.

基于Gini指数的CART模型

- 在“是否买电脑”中，D有 9 个样本 “是” 5 个样本 “否”

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

- 属性 *收入* 将D分成：10个在 D_1: {medium,high} 以及4个在D_2：{low}

基于Gini指数的CART模型

- 在“是否买电脑”中，D有 9 个样本 “是” 5 个样本 “否”

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

- 属性 *收入* 将D分成：10个在 D_1: {medium,high} 以及4个在D_2：{low}

$$gini_{income \in \{\text{medium,high}\}}(D) = \frac{10}{14} gini(D_1) + \frac{4}{14} gini(D_2)$$

$$= \frac{10}{14}\left(1 - (\frac{6}{10})^2 - (\frac{4}{10})^2\right) + \frac{4}{14}\left(1 - (\frac{1}{4})^2 - (\frac{3}{4})^2\right)$$

$$= 0.450 = gini_{income \in \{low\}}(D)$$

连续型属性的处理

- 应该如何计算具有**连续值**属性的基尼指数，信息增益?

连续型属性的处理

- 应该如何计算具有**连续值**属性的基尼指数，信息增益?
  - 给定A的 v 个值，那么有 v-1个可能的分裂位置。比如在 A 中，a_i and a_{i+1} 的中点是

$$(a_i + a_{i+1}) / 2$$

决策树

CART的生成

- 回归树的生成
- 设y是连续变量，给定训练数据集：$D=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$
- 假设已将输入空间划分为M个单元$R_1,R_2...R_m$，并且每个单元R_m上有一个固定的输出c_m，回归树表示为：$$f(x)=\sum_{m=1}^{M}c_m I(x\in R_m)$$
- 平方误差来表示预测误差，用平方误差最小准则求解每个单元上的最优输出值 $\sum_{x_i\in R_m}(y_i-f(x_i))^2$
- R_m上的c_m的最优值：$\hat{c}_m=\text{ave}(y_i\mid x_i\in R_m)$

决策树

CART的生成

- 问题：如何对输入空间进行划分？
- 启发式：选择第j个变量$x^{(j)}$和它取的值s，作为切分变量和切分点，定义两个区域：

$$R_1(j,s)=\{x \mid x^{(j)} \leqslant s\} \quad 和 \quad R_2(j,s)=\{x \mid x^{(j)} > s\}$$

- 然后寻找最优切分变量和切分点：

$$\min_{j,s}\left[\min_{c_1}\sum_{x_i \in R_1(j,s)}(y_i-c_1)^2+\min_{c_2}\sum_{x_i \in R_2(j,s)}(y_i-c_2)^2\right]$$

- 且：$\hat{c}_1=\text{ave}(y_i \mid x_i \in R_1(j,s))$ 和 $\hat{c}_2=\text{ave}(y_i \mid x_i \in R_2(j,s))$

- 再对两个区域重复上述划分，直到满足停止条件。

决策树

CART的生成

- 最小二乘回归树生成算法

输入：训练数据集 D；

输出：回归树 $f(x)$.

在训练数据集所在的输入空间中，递归地将每个区域划分为两个子区域并决定每个子区域上的输出值，构建二叉决策树：

（1）选择最优切分变量 j 与切分点 s，求解

$$\min_{j,s}\left[\min_{c_1}\sum_{x_i\in R_1(j,s)}(y_i-c_1)^2+\min_{c_2}\sum_{x_i\in R_2(j,s)}(y_i-c_2)^2\right]$$

遍历变量 j，对固定的切分变量 j 扫描切分点 s，选择使式(
达到最小值的对 (j,s)

决策树

CART的生成

- 最小二乘回归树生成算法

（2）用选定的对 (j,s) 划分区域并决定相应的输出值：

$$R_1(j,s)=\{x \mid x^{(j)} \leqslant s\}, \quad R_2(j,s)=\{x \mid x^{(j)} > s\}$$

$$\hat{c}_m = \frac{1}{N_m} \sum_{x_i \in R_m(j,s)} y_i, \quad x \in R_m, \quad m=1,2$$

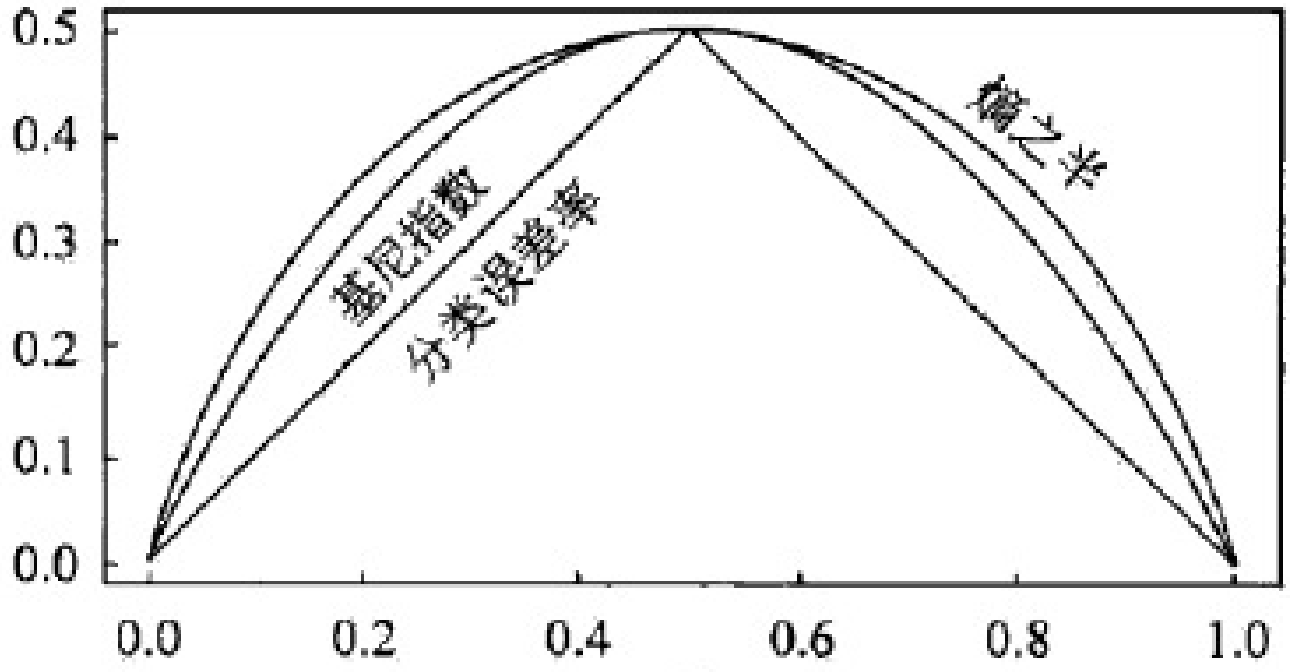
（3）继续对两个子区域调用步骤 (1)，(2)，直至满足停止条件.

（4）将输入空间划分为 M 个区域 $R_1, R_2, \cdots, R_M$，生成决策树：

$$f(x)=\sum_{m=1}^{M} \hat{c}_m I(x \in R_m)$$

决策树

CART的生成



决策树

CART的生成

- CART生成算法
- 输入：训练数据集D，停止计算条件
- 输出：CART决策树
- 从根节点开始，递归对每个结点操作
- 1、设结点数据集为D，对每个特征A，对其每个值a，根据样本点对$A=a$的测试为是或否，将D分为D_1，D_2，计算$A=a$的基尼指数
- 2、在所有的特征A以及所有可能的切分点a中，选择基尼指数最小的特征和切分点，将数据集分配到两个子结点中
- 3、对两个子结点递归调用1，2步骤
- 4、生成CART树

生成分类规则

- 将知识表示为 IF-THEN 形式的规则。
- 对每条从根节点到叶子节点的路径，创建一条规则。
- 从一个节点到下一层节点的一条分支上，每个属性-值对可以形成一个连接。
- 叶子节点代表预测的分类。
- 规则应当容易被人理解 (可解释性)。

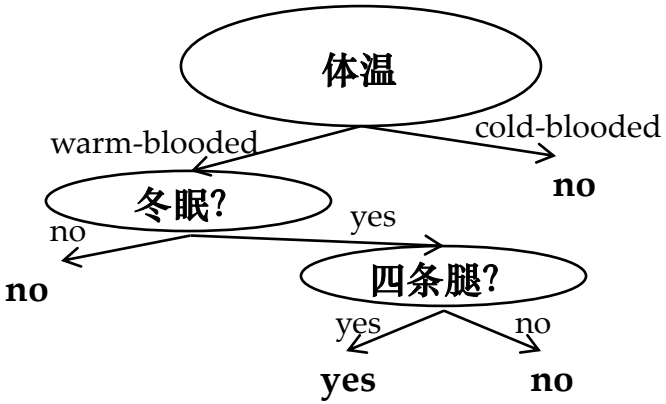
过拟合

- **过拟合问题**
  - 一颗决策树可能对训练样本过拟合，导致在测试样本上没有良好的泛化表现。 (即，在训练集上有良好表现但在新样本上准确率很低)

过拟合

- 训练样本错误为0，将所有不冬眠的温血动物分类为非哺乳动物。

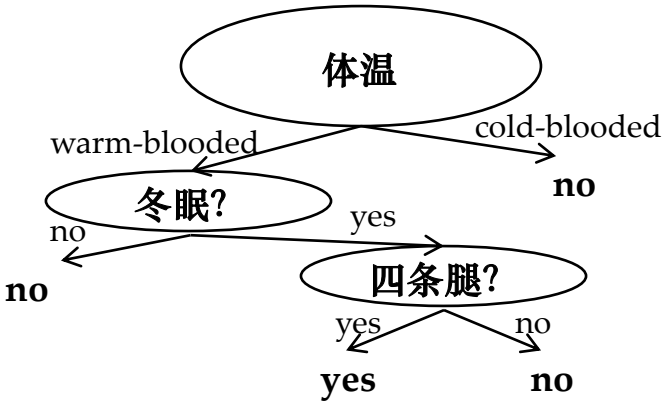
名称	体温	四条腿?	冬眠?	哺乳动物?
salamander	cold-blooded	yes	yes	no
guppy	cold-blooded	no	no	no
eagle	warm-blooded	no	no	no
poorwill	warm-blooded	no	yes	no
platypus	warm-blooded	yes	yes	yes



过拟合

- 训练样本错误为0，将所有不冬眠的温血动物分类为非哺乳动物。

名称	体温	四条腿?	冬眠?	哺乳动物?
salamander	cold-blooded	yes	yes	no
guppy	cold-blooded	no	no	no
eagle	warm-blooded	no	no	no
poorwill	warm-blooded	no	yes	no
platypus	warm-blooded	yes	yes	yes



Humans, elephants and dolphins都分错了

过拟合

- 基于少量训练样本做出分类决策的模型都很容易过拟合。
  - 解决办法：加入更多样本训练模型

名称	体温	四条腿?	冬眠?	哺乳动物?
salamander	cold-blooded	yes	yes	no
guppy	cold-blooded	no	no	no
eagle	warm-blooded	no	no	no
poorwill	warm-blooded	no	yes	no
platypus	warm-blooded	yes	yes	yes
human	warm-blooded	no	no	yes
dolphin	warm-blooded	no	no	yes
elephant	warm-blooded	yes	no	yes

过拟合

名称	体温	胎生?	四条腿?	冬眠?	哺乳动物?
porcupine	warm-blooded	yes	yes	yes	yes
cat	warm-blooded	yes	yes	no	yes
bat	warm-blooded	yes	no	yes	no
whale	warm-blooded	yes	no	no	no
salamander	cold-blooded	no	yes	yes	no
komodo dragon	cold-blooded	no	yes	no	no
python	cold-blooded	no	no	yes	no
salmon	cold-blooded	no	no	no	no
eagle	warm-blooded	no	no	no	no
guppy	cold-blooded	yes	no	no	no

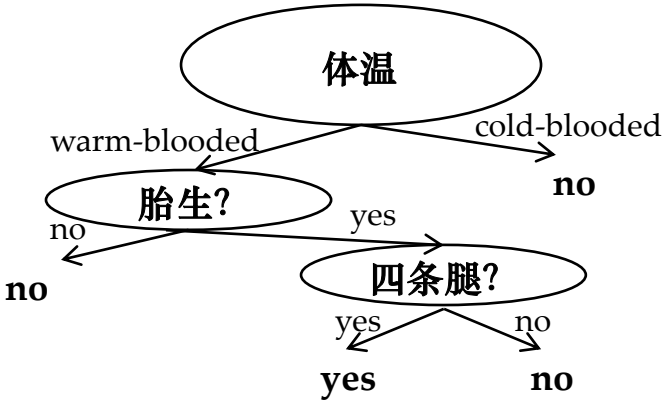
Training set

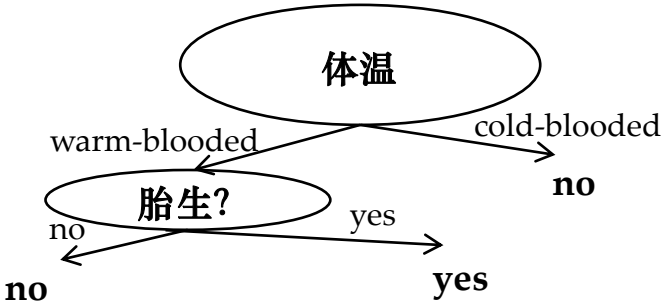
过拟合

名称	体温	胎生?	四条腿?	冬眠?	哺乳动物?
human	warm-blooded	yes	no	no	yes
pigeon	warm-blooded	no	no	no	no
elephant	warm-blooded	yes	yes	no	yes
leopard shark	cold-blooded	yes	no	no	no
turtle	cold-blooded	no	yes	no	no
penguin	warm-blooded	no	no	no	no
eel	cold-blooded	no	no	no	no
dolphin	warm-blooded	yes	no	no	yes
spiny anteater	warm-blooded	no	yes	yes	yes
gila monster	cold-blooded	no	yes	yes	no

Testing set

过拟合





训练集上错误个数 0
测试集上错误率为 30%

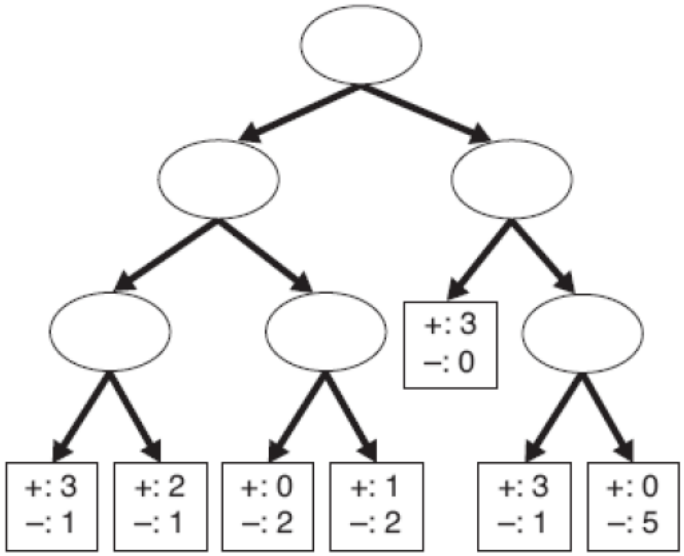
训练集上错误率 20%
测试集上错误率 10%

过拟合

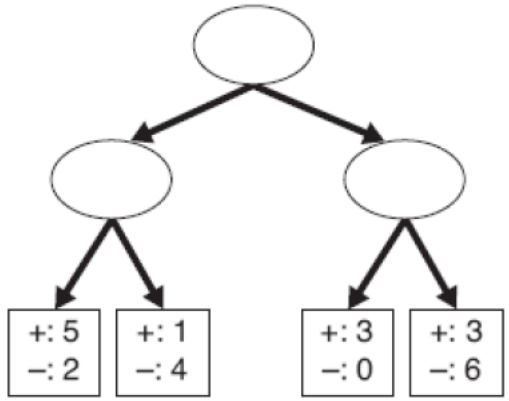
- 训练集的错误可以通过增加模型复杂度减少
  - 当树变得很大，训练集错误率可以持续降低
- 但是，测试集错误率（泛化错误率）会很大，因为模型会拟合一些训练样本的噪声点
  - 太多分支，某些分支会反映出因为噪声和离群值而产生的异常。

融入模型复杂度

- 泛化错误率用以下两个值的和估计：
  - 训练集错误数
  - 模型复杂度的惩罚项
- T_L 的训练集错误数为 $e(T_L)=4/24=0.167$
- T_R 的训练集错误数为 $e(T_R)=6/24=0.25$



Decision Tree, T_L



Decision Tree, T_R

融入模型复杂度

- 在决策树中，定义以下变量
  - L：叶子节点个数
  - n_l：第 l个叶子节点
  - $m(n_l)$：被 n_l 准确分类的训练样本数
  - $r(n_l)$：被 n_l 错误分类的训练样本数
  - $\zeta(n_l)$：n_l的惩罚项
- 决策树的错误率 e_c 可以估计如下:

$$e_c = \frac{\sum_{l=1}^{L}\left(r(n_l)+\zeta(n_l)\right)}{\sum_{l=1}^{L} m(n_l)}$$

融入模型复杂度

- 考虑之前的两棵决策树 T_L 和 T_R.
- 假设对每个叶子节点的惩罚项为 0.5

- T_L 的错误率估计为

$$e_c(T_L) = \frac{4 + 7 \times 0.5}{24} = \frac{7.5}{24} = 0.3125$$

- T_R 的错误率估计为

$$e_c(T_R) = \frac{6 + 4 \times 0.5}{24} = \frac{8}{24} = 0.3333$$

融入模型复杂度

- 基于此惩罚项, 可认为T_L 优于 T_R.
- 对一个二叉树而言, 惩罚项为 0.5 意味着当一个节点分裂为两个节点后能够提升至少一个分类准确数，那就必须分裂。
- 因为增加一个节点，相当于在总体错误加上0.5，比把一个样本错误分类的代价小。

融入模型复杂度

- 假设对于所有叶子节点的惩罚项都为1
- T_L 的错误估计为 0.458.
- T_R 的错误估计为 0.417.
- 基于这个惩罚项，T_R 要优于 T_L.
- 惩罚项为 1 意味着当分裂一个节点能够使多于一个的训练样本被正确分类时，该节点才应该被分裂。

决策树剪枝

- 避免过拟合的两个方法
  - 预剪枝: 提前停止建树-如果分裂节点会导致树的分类准确率低于阈值
    - 难点在于选择一个合适的阈值。
  - 后剪枝: 从一个“建满了”的树上剪枝-获得一系列进一步剪枝了的树
    - 用不同于训练数据的数据集去决定最好的剪枝树是哪个。

决策树的选取方法

- 分出训练集和测试集
- 使用交叉认证，比如，k折交叉认证
  - 把数据集分成k部分
  - 随机选取k-1部分用于训练，在剩下那部分上测试
  - 重复k次

决策树

决策树和归纳算法

- 决策树技术发现数据模式和规则的核心是归纳算法。
- 归纳是从特殊到一般的过程。
- 归纳推理从若干个事实中表征出的特征、特性和属性中，通过比较、总结、概括而得出一个规律性的结论。
- 归纳推理试图从对象的一部分或整体的特定的观察中获得一个完备且正确的描述。即从特殊事实到普遍性规律的结论。
- 归纳对于认识的发展和完善具有重要的意义。人类知识的增长主要来源于归纳学习。

决策树

决策树和归纳算法

- 归纳学习由于依赖于检验数据，因此又称为检验学习。
- 归纳学习存在一个基本的假设：
  - 任一假设如果能够在足够大的训练样本集中很好的逼近目标函数，则它也能在未见样本中很好地逼近目标函数。该假定是归纳学习的有效性的前提条件。

参考资料

- 两个连续变量的线性相关度，用协方差或相关系数来衡量
  - 非线性相关度：Maximal Information Coefficient (MIC). Detecting novel associations in large data sets, *Science*, 2011.
- 两个离散变量的相关度，用互信息度量
  - 互信息倾向于选择取值更多的离散型属性：A framework to adjust dependency measure estimates for chance, *SDM*, 2016.
  - Entropy evaluation based on confidence intervals of frequency estimates: application to the learning of decision trees, *ICML*, 2015.
  - Standardized mutual information for clustering comparisons: one step further in adjustment for chance, *ICML*, 2014.