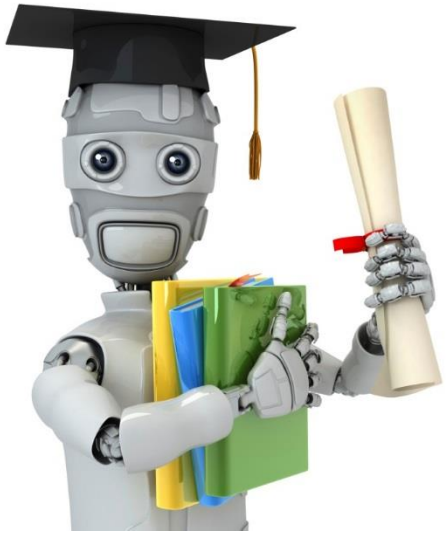


决策树 (Decision Tree)

梁毅雄

yxliang@csu.edu.cn

Machine Learning

Some materials from Xing Eric, Hang Li and others

决策树

- 分类问题一般包括两个步骤:
  - 模型构建（归纳）：通过对训练集合的归纳，建立分类模型
  - 预测应用（推论）：根据建立的分类模型，对测试集合进行测试
- 决策树是一种典型的分类方法:
  - 首先对数据进行处理，利用归纳算法生成可读的规则和决策树
  - 然后使用决策对新数据进行分析
  - 本质上是通过一系列规则对数据进行分类的过程
  - 相关算法： CLS ， ID3，C4.5，CART

决策树

女孩决定是否去相亲：

女儿：多大年纪了？

母亲：26。

女儿：长的帅不帅？

母亲：挺帅的。

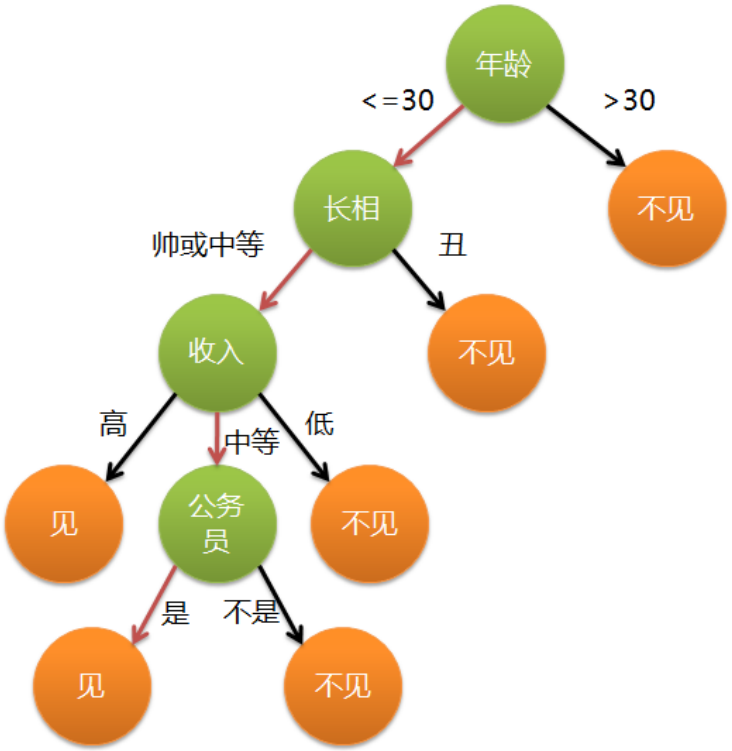
女儿：收入高不？

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。

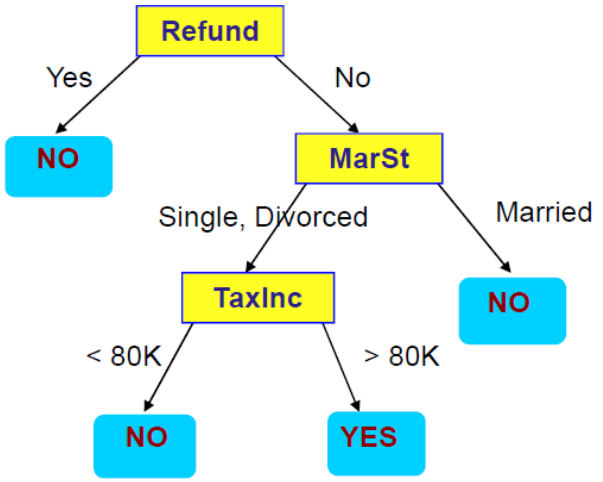
女儿：那好，我去见见。

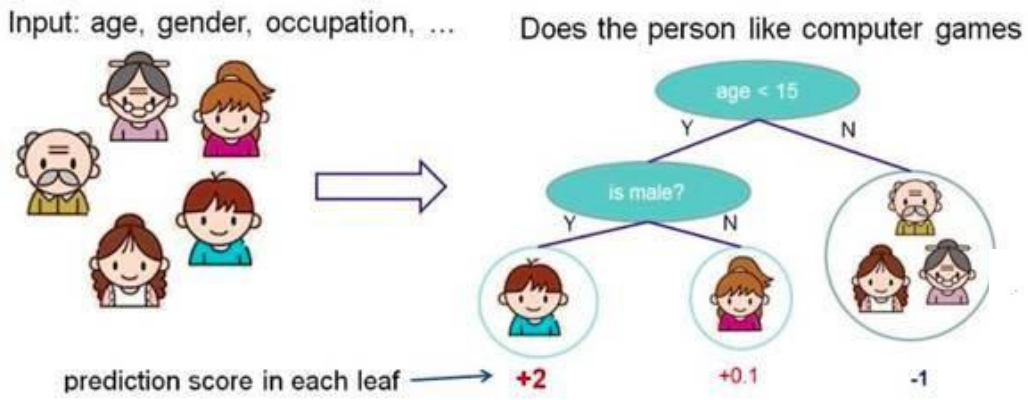


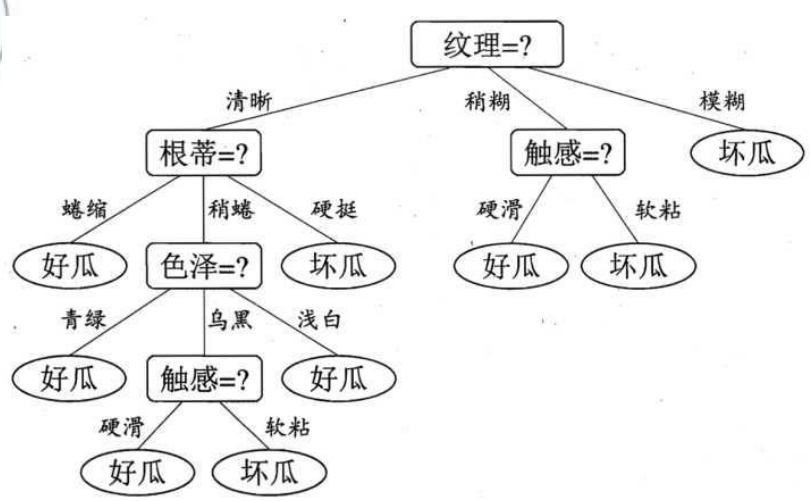
http://www.cnblogs.com/leoo2sk/archive/2010/09/19/decision-tree.html

决策树

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?







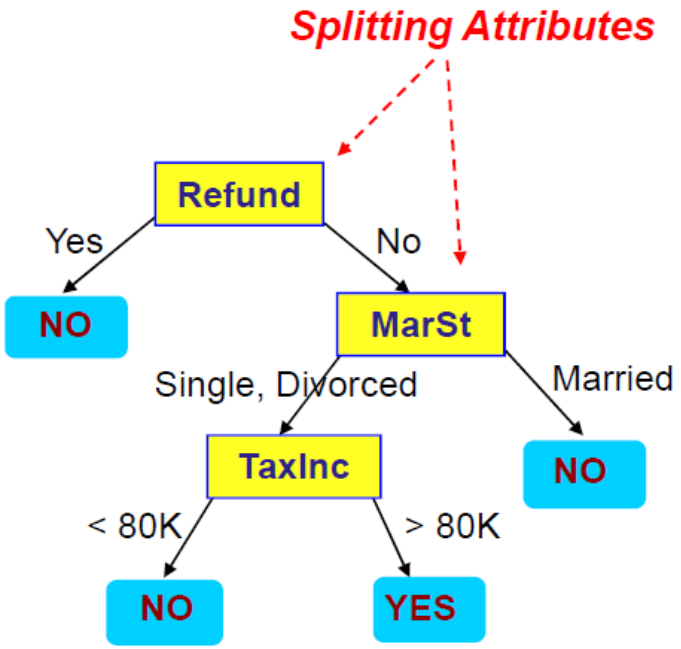
决策树

- 决策树（decision tree）具有树结构:
  - 树的最顶层是根节点，包含所有样本
  - 每个内部节点表示在一个特征（或属性）上的测试，将当前节点的样本划分到不同的子节点中；
  - 每个叶节点代表类或类分布，对应决策结果
  - 使用决策树进行决策的过程就是从根节点开始，测试待分类项中相应的特征属性，并按照其值选择输出分支，直到到达叶子节点，将叶子节点存放的类别作为决策结果

决策树的生成

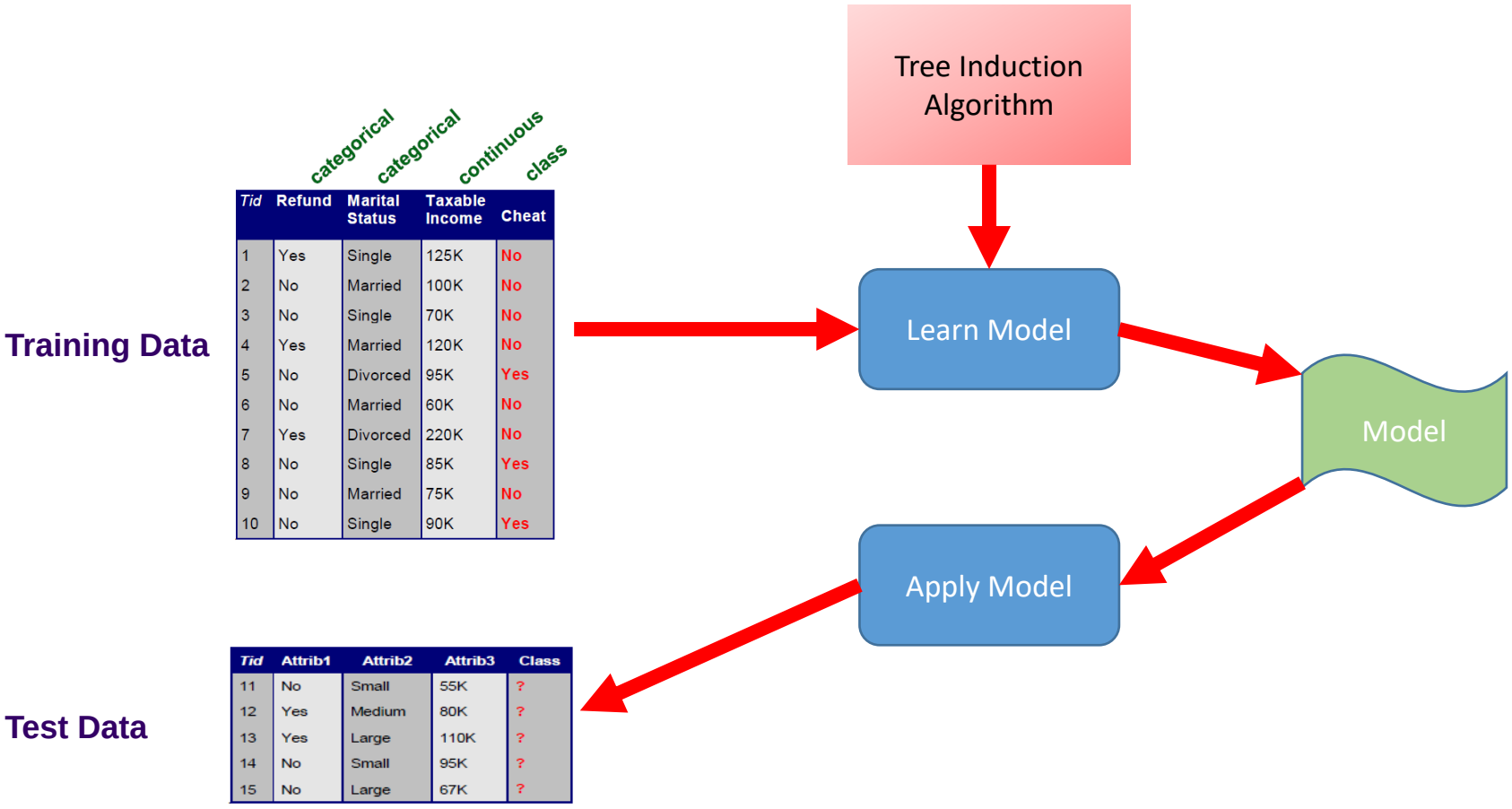
Tid	Refund	Marital Status	Taxable Income	Cheat
	categorical	categorical	continuous	class
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Training Data



Model: Decision T

决策树的生成


Tree Induction Algorithm
Learn Model
Model
Apply Model
Training Data
Test Data
categorical
categorical
continuous
class
Tid Refund Marital Status Taxable Income Cheat
1 Yes Single 125K No
2 No Married 100K No
3 No Single 70K No
4 Yes Married 120K No
5 No Divorced 95K Yes
6 No Married 60K No
7 Yes Divorced 220K No
8 No Single 85K Yes
9 No Married 75K No
10 No Single 90K Yes
Tid Attrib1 Attrib2 Attrib3 Class
11 No Small 55K ?
12 Yes Medium 80K ?
13 Yes Large 110K ?
14 No Small 95K ?
15 No Large 67K ?

决策树CLS算法

- CLS（Concept Learning System）算法
  - CLS算法是早期的决策树学习算法。它是许多决策树学习算法的基础
- 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

- 步骤:
  1. 生成一颗空决策树和一张训练样本属性集;
  2. 若训练样本集D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
  3. 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, 生成测试结点若a的取值为$v_1,v_2,\ldots,v_m$, 则根据a 的取值的不同,将D划分成 m个子集$D_1,D_2,\ldots,D_m$;
  4. 从训练样本属性表中删除属性a;
  5. 转步骤2, 对每个子集递归调用CLS;
- 算法关键：
  - 在步骤3中，如何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

信息增益

- Shannon 1948年提出的信息论理论
- 熵(entropy)：信息量大小的度量，即表示随机变量不确定性的度量
- 事件a_i的信息量$I(a_i)$可如下度量: $I(a_i) = -\log p(a_i)$, 这里$p(a_i)$为事件a_i发生的概率
- 假设有n个互不相容的事件$a_1, \cdots, a_n$,它们中有且仅有一个发生，则其平均的信息量(熵)可如下度量:
  $I(a_1, \cdots, a_n) = \sum_i I(a_i) = -\sum_i p(a_i) \log p(a_i)$

信息增益

- 假设当前样本集D中第k类样本的比例为p_k, 对应的信息熵为:
  $Ent(D) = -\sum_k p_k \log p_k$
- $Ent(D)$越小，表示数据越有序，纯度越高，分类效果越好
- 假设某离散属性a有V个可能值，若采用该属性对样本集来划分，则会产生V个分支，每个分支节点包含的数据记为D^v
- 用属性a对样本集D进行划分，获得的信息增益为:
  $Gain(D, a) = Ent(D) - \sum_v \frac{|D^v|}{|D|} Ent(D^v)$
- 选择具有最大信息增益的属性来划分: $a^* = \arg\max_a Gain(D, a)$ (ID3)

计数	年龄	收入	学生	信誉	归类：买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	中	高	否	良	买
60	老	中	否	良	买
64	老	低	是	良	买
64	老	低	是	优	不买
64	中	低	是	优	买
128	青	中	否	良	不买
64	青	低	是	良	买
132	老	中	是	良	买
64	青	中	是	优	买
32	中	中	否	优	买
32	中	高	是	良	买
63	老	中	否	优	不买
1	老	中	否	优	买

- 计算决策属性的熵
- 决策属性“买？”
- 该属性分两类：买/不买

$$|D^1| = 641,\ |D^2| = 383,$$
$$|D| = |D^1| + |D^2| = 1024$$

$$p_1 = 641/1024 = 0.6260$$
$$p_2 = 383/1024 = 0.3740$$

$$Ent(D) = -\sum_k p_k \log p_k = 0.9537$$

计数	年龄	收入	学生	信誉	归类：买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	中	高	否	良	买
60	老	中	否	良	买
64	老	低	是	良	买
64	老	低	是	优	不买
64	中	低	是	优	买
128	青	中	否	良	不买
64	青	低	是	良	买
132	老	中	是	良	买
64	青	中	是	优	买
32	中	中	否	优	买
32	中	高	是	良	买
63	老	中	否	优	不买
1	老	中	否	优	买

- 计算条件属性的熵
  - 年龄、收入、学生、信誉。
  - 分别计算不同属性的信息增益。

年龄共分三个组：青年、中年、老年

青年买与不买比例为128/256

$|D^{11}| = 128$(买), $|D^{12}| = 256$(不买),

$|D^1| = 384$

$$p_1 = 128/384$$

$$p_2 = 256/384$$

$$Ent(D^1) = -\sum_k p_k \log p_k = 0.9183$$

中年买与不买比例为256/0

$p_1 = 256/256, p_2 = 0/256$

$$Ent(D^2) = 0$$

老年买与不买比例为125/127

$p_1 = 125/252, p_2 = 127/252$

$$Ent(D^3) = 0.9157$$

计数	年龄	收入	学生	信誉	归类：买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	中	高	否	良	买
60	老	中	否	良	买
64	老	低	是	良	买
64	老	低	是	优	不买
64	中	低	是	优	买
128	青	中	否	良	不买
64	青	低	是	良	买
132	老	中	是	良	买
64	青	中	是	优	买
32	中	中	否	优	买
32	中	高	是	良	买
63	老	中	否	优	不买
1	老	中	否	优	买

$$Ent(D^1) = 0.9183$$
$$Ent(D^2) = 0$$
$$Ent(D^3) = 0.9157$$

青年、中年、老年所占比例分别为:

$$\frac{|D^1|}{|D|} = 384/1024 = 0.375,$$
$$\frac{|D^2|}{|D|} = 256/1024 = 0.25,$$
$$\frac{|D^3|}{|D|} = 384/1024 = 0.375$$

年龄属性的信息增益为:

$$Gain(D,a) = Ent(D) - \sum_v \frac{|D^v|}{|D|} Ent(D^v)$$
$$= 0.9537 - (0.375 * 0.9183 + 0.25 * 0 + 0.375 * 0.9157)$$
$$=0.2660$$

计数	年龄	收入	学生	信誉	归类：买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	中	高	否	良	买
60	老	中	否	良	买
64	老	低	是	良	买
64	老	低	是	优	不买
64	中	低	是	优	买
128	青	中	否	良	不买
64	青	低	是	良	买
132	老	中	是	良	买
64	青	中	是	优	买
32	中	中	否	优	买
32	中	高	是	良	买
63	老	中	否	优	不买
1	老	中	否	优	买

年龄属性的信息增益为：

$$Gain(D,a) = Ent(D) - \sum_v \frac{|D^v|}{|D|} Ent(D^v)$$
$$= 0.9537 - (0.375 * 0.9183 + 0.25 * 0 + 0.375 * 0.9157)$$
$$= 0.2660$$

收入属性的信息增益为：

$$Gain(D,a) = Ent(D) - \sum_v \frac{|D^v|}{|D|} Ent(D^v)$$
$$= 0.9537 - 0.9361$$
$$= 0.0176$$

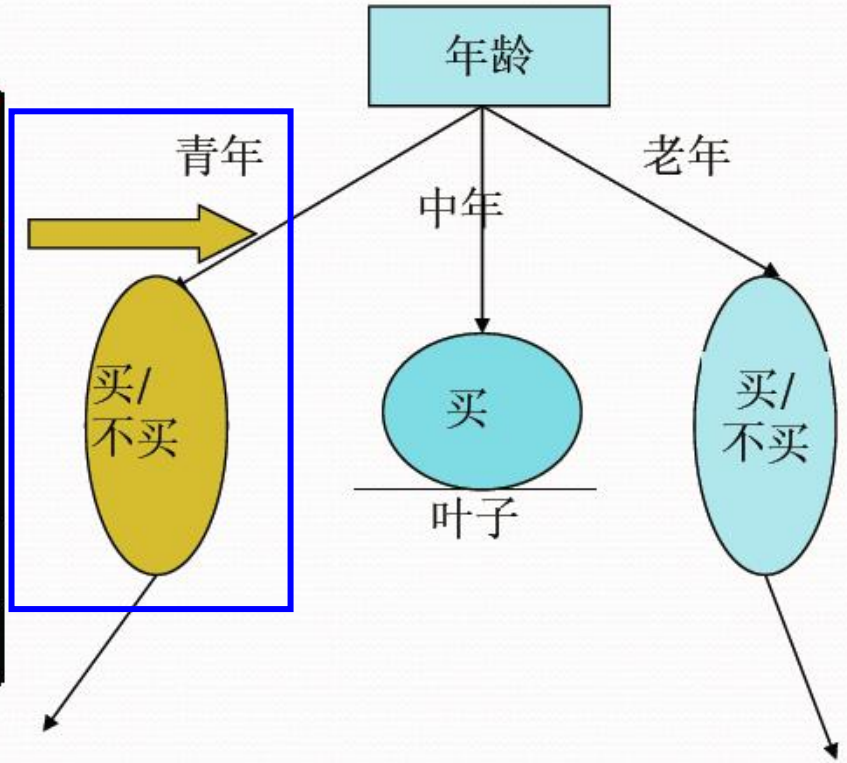
学生属性的信息增益为：

$$Gain(D,a) = Ent(D) - \sum_v \frac{|D^v|}{|D|} Ent(D^v)$$
$$= 0.9537 - 0.7811$$
$$= 0.1726$$

信誉属性的信息增益为：

$$Gain(D,a) = Ent(D) - \sum_v \frac{|D^v|}{|D|} Ent(D^v)$$
$$= 0.9537 - 0.9048$$
$$= 0.0453$$

计数	年龄	收入	学生	信誉	归类：买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	青	中	否	良	不买
64	青	低	是	良	买
64	青	中	是	优	买



下一步对该节点进行处理

计数	年龄	收入	学生	信誉	归类：买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	青	中	否	良	不买
64	青	低	是	良	买
64	青	中	是	优	买

青年买与不买比例为128/256

$p_1 = 128/384$

$p_2 = 256/384$

对应的熵为：

$Ent(D) = -\sum_k p_k \log p_k = 0.9183$

若根据收入属性进行划分高、中、低三类

$Ent(D^1) = 0$

比例为$128/384 = 0.3333$

$Ent(D^2) = 0.9183$

比例为$192/384 = 0.5$

$Ent(D^3) = 0$

比例为$64/384 = 0.1667$

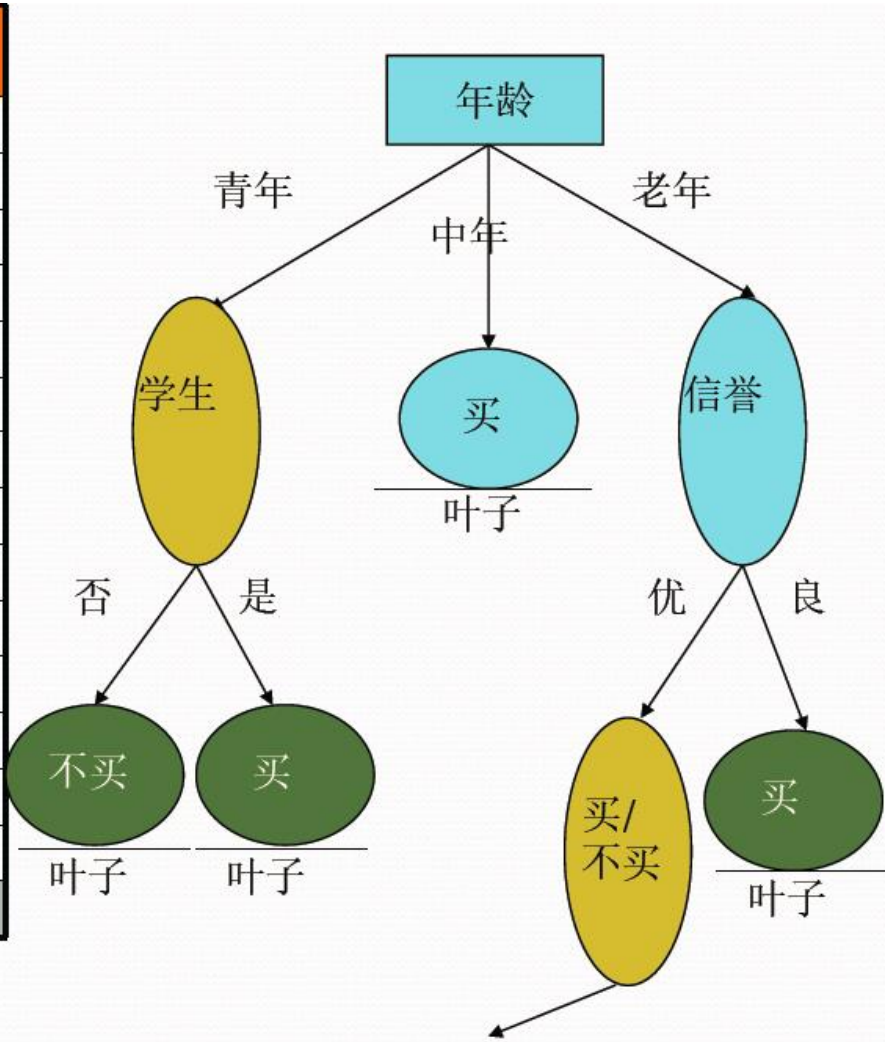
收入属性的信息增益为：

$Gain(D, a) = Ent(D) - \sum_v \frac{|D^v|}{|D|} Ent(D^v)$

$= 0.9537 - (0.3333 * 0 + 0.5 * 0.9183 + 0.1667 * 0)$

$= 0.4591$

计数	年龄	收入	学生	信誉	归类：买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	中	高	否	良	买
60	老	中	否	良	买
64	老	低	是	良	买
64	老	低	是	优	不买
64	中	低	是	优	买
128	青	中	否	良	不买
64	青	低	是	良	买
132	老	中	是	良	买
64	青	中	是	优	买
32	中	中	否	优	买
32	中	高	是	良	买
63	老	中	否	优	不买
1	老	中	否	优	买



ID3决策树生成算法

1. 决定分类属性；
2. 对目前的数据表，建立一个节点N
3. 如果数据库中的数据都属于同一个类，N就是树叶，在树叶上标出所属的类
4. 如果数据表中没有其他属性可以考虑，则N也是树叶，按照少数服从多数的原则在树叶上标出所属类别
5. 否则，根据平均信息期望值GAIN值选出一个最佳属性作为节点N的测试属性
6. 节点属性选定后，对于该属性中的每个值
   - 从N生成一个分支，并将数据表中与该分支有关的数据收集形成分支节点的数据表，在表中删除节点属性那一栏如果分支数据表非空，则运用以上算法从该节点建立子树

ID3决策树应用

原始表

姓名	年龄	收入	学生	信誉	电话	地址	邮编	买计算机
张三	23	4000	是	良	281-322-0328	2714 Ave. M	77388	买
李四	34	2800	否	优	713-239-7830	5606 Holly Cr	78766	买
王二	70	1900	否	优	281-242-3222	2000 Bell Blvd.	70244	不买
赵五	18	900	是	良	281-550-0544	100 Main Street	70244	买
刘兰	34	2500	否	优	713-239-7430	606 Holly Ct	78566	买
杨俊	27	8900	否	优	281-355-7990	233 Rice Blvd.	70388	不买
张毅	38	9500	否	优	281-556-0544	399 Sugar Rd.	78244	买
。。。	。。							
。。。								

ID3决策树应用

原始表

姓名	年龄	收入	学生	信誉	电话	地址	邮编	买计算机
张三	23	4000	是	良	281-322-0328	2714 Ave. M	77388	买
李四	34	2800	否	优	713-239-7830	5606 Holly Cr	78766	买
王二	70	1900	否	优	281-242-3222	2000 Bell Blvd.	70244	不买
赵五	18	900	是	良	281-550-0544	100 Main Street	70244	买
刘兰	34	2500	否	优	713-239-7430	606 Holly Ct	78566	买
杨俊	27	8900	否	优	281-355-7990	233 Rice Blvd.	70388	不买
张毅	38	9500	否	优	281-556-0544	399 Sugar Rd.	78244	买
。。。	。。							
。。。								

整理后的数据表

计数	年龄	收入	学生	信誉	归类：买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	中	高	否	良	买
60	老	中	否	良	买
64	老	低	是	良	买
64	老	低	是	优	不买
64	中	低	是	优	买
128	青	中	否	良	不买
64	青	低	是	良	买
。。。					

ID3决策树应用

整理后的数据表

计数	年龄	收入	学生	信誉	归类：买计算机？
64	青	高	否	良	不买
64	青	高	否	优	不买
128	中	高	否	良	买
60	老	中	否	良	买
64	老	低	是	良	买
64	老	低	是	优	不买
64	中	低	是	优	买
128	青	中	否	良	不买
64	青	低	是	良	买
。。。					

- Data cleaning
  - 删除/减少noise，
  - 补填missing values
- Data transformation
  - 数据标准化（data normalization）
  - 数据归纳（generalize data to higher-level concepts using concept hierarchies）
  - 例如：年龄归纳为老、中、青三类。控制每个属性的可能值不超过七种（最好不超过五种）
- Relevance analysis
  - 对于与问题无关的属性：删
  - 对于属性的可能值大于七种又不能归纳的属性：删

信息增益比

- ID3算法的基本思想是以信息增益选择属性，实际应用中会对可能取值数目较多的属性有所偏好。
  - 如对每个训练样本进行编号，并将该编号作为属性，其信息增益最大，但显然该属性不能作为分类依据
- 信息增益比：

$$\text{Gain\_ratio}(D,a) = \frac{\text{Gain}(D,a)}{IV(a)},$$

其中

$$IV(a) = -\sum_{v=1}^{V} \frac{|D^v|}{|D|} \log \frac{|D^v|}{|D|}$$

为属性a的intrinsic value, a可能的取值越多，$IV(a)$通常也越大。

决策树C4.5的生成算法

输入：训练数据集 D，特征集 A，阈值 ε；

输出：决策树 T.

（1）如果 D 中所有实例属于同一类 C_k，则置 T 为单结点树，并将 C_k 作为该结点的类，返回 T；

（2）如果 $A=\varnothing$，则置 T 为单结点树，并将 D 中实例数最大的类 C_k 作为该结点的类，返回 T；

（3）否则，按式(5.10)计算 A 中各特征对 D 的信息增益比，选择信息增益比最大的特征 A_g；

（4）如果 A_g 的信息增益比小于阈值 ε，则置 T 为单结点树，并将 D 中实例数最大的类 C_k 作为该结点的类，返回 T；

（5）否则，对 A_g 的每一可能值 a_i，依 $A_g=a_i$ 将 D 分割为子集若干非空 D_i，将 D_i 中实例数最大的类作为标记，构建子结点，由结点及其子结点构成树 T，返回 T；

（6）对结点 i，以 D_i 为训练集，以 $A-\{A_g\}$ 为特征集，递归地调用步(1)～步(5)，得到子树 T_i，返回 T_i.

■

决策树的剪枝

- 决策树的剪枝减少决策树的规模（模型复杂度），是处理“过拟合”问题的主要手段
- 通过极小化决策树整体的Cost Complexity函数实现:
  $CC(T) = Err(T) + \lambda R(T)$，
  这里$Err(T)$为树T的错误，$R(T)$为正则项，描述树的复杂度（如树节点的个数），$\lambda \geq 0$为正则化参数
- 如：设树T的叶结点个数为$|T|$, t是树T的叶结点，该叶结点有N_t个样本点，其中k类的样本点有N_{tk}个，$k = 1, 2, \cdots, K$。叶结点t上的经验熵: $H_t(T) = -\sum_{k=1}^{K} \frac{N_{tk}}{N_t} \log \frac{N_{tk}}{N_t}$，则目标函数为：
  $CC(T) = \sum_{i=1}^{|T|} N_t H_t(T) + \lambda |T| = -\sum_{i=1}^{|T|} \sum_{k=1}^{K} N_{tk} \log \frac{N_{tk}}{N_t} + \lambda |T|$

决策树的剪枝算法

- 输入：生成算法产生的整个决策树T，正则参数λ
- 输出：修剪后的决策树T_o
  - Step 1: 计算每个节点的代价
  - Step 2: 递归地从树的叶节点向上回溯
- 设一组叶节点回缩到其父节点之前与之后的代价分别为: $CC(T_B)$和$CC(T_A)$，若$CC(T_A) \leq CC(T_B)$, 则剪枝，返回Step 2, 直到不能继续为止。
- 注意是根据验证集上的代价来决定是否剪枝

CART树

- 分类回归树(Classification and Regression Tree, CART)
  - 分类树：Gini Index
  - 回归树：平方误差最小化
- CART算法由两部分组成
  - 决策树生成
  - 决策树剪枝

- CART vs. ID3
  - 二元划分：二叉树不易产生数据碎片，精确度往往也会高于多叉树
  - 属性选择：
    - Gini Index - 分类树
    - 最小平方残差 - 回归树

基尼系数与CART决策树

- CART决策树采用基尼系数(Gini Index)来选择属性进行划分
- 基尼值:

$$\text{Gini}(D) = \sum_{k=1}^{K} \sum_{k' \neq k} p_k p_{k'} = 1 - \sum_{k=1}^{K} p_k^2$$

- 直观上，基尼值反应了从数据集中任选2个样本，其类别不一致的概率，其值越小，纯度越高
- 基尼系数:

$$\text{Gini\_index}(D, a) = \sum_{v=1}^{V} \frac{|D^v|}{|D|} \text{Gini}(D^v)$$

CART决策树

- 由于CART决策树是二叉树，
  - 问题1：属性取值仍为离散、但超过2种取值的情况如何处理（如年龄属性取值为青年、中年、老年）？
  - 问题2：如何处理属性取值为连续的情况（如月收入、西瓜的密度等）？
- 多属性值的处理：
  - 组合的方式转化成多个二取值问题：如（青年、中老年）、（中年、青老年）、（中青年、老年）等
  - 然后分别计算Gini Index，选择Gini Index最小的二分情况

连续属性的处理

- C4.5决策树和CART决策树均采用将连续属性离散化，最简单的是二分法
- 虽然本质上属性的取值是连续的，但对于有限的采样数据它是离散的
- 给定样本集D和某连续属性a, 假定a在D上出现了n个不同的取值，由大到小排序后记为$\{a^1, a^2, \cdots, a^n\}$.
- 基于阈值t可把D划分为D_t^-和D_t^+两个子集，其中D_t^-包含哪些属性a取值小于t的那部分样本，D_t^+包含哪些属性a取值大于t的那部分样本
- 显然，对于任意的$t \in (a^i, a^{i+1})$，效果相同，因此可遍历如下阈值集合$\{\frac{1}{2}(a^i + a^{i+1}) | 1 \leq i \leq n-1\}$, 根据Gini Index或Gain Ratio来选择最优的阈值t进行离散化

CART决策树的生成算法

- 输入: 训练数据集D，迭代终止条件
- 输出: CART决策树
- 从根节点开始，递归对每个结点进行以下操作构造二叉树
  - 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^-，D^+，计算$A = a$的基尼指数
  - 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
  - 对两个子结点递归调用上面两个步骤，最终生成CART决策树

CART回归树的生成

回归树

- 包括对应输入空间（特征空间）的一个划分和在划分的每个单元上的输出值
- 数学上，回归树可以是一个分段函数，每个叶子节点确定一个分段区间（disjoint），叶子节点的输出为函数在该节点上的值
- 假设CART树T将特征空间划分为$|T|$个区域R_i，并且在每个区域对应的值为b_i, 对应的假设函数为

$$h(x) = \sum_{i=1}^{|T|} b_i \mathbb{I}(x \in R_i)$$

- 两个问题：如何划分区域R_i和如何确定每个区域R_i上对应的值b_i?

CART回归树的生成

- 假设各个区域R_j已知，可以用最小平方损失$\sum_{x^{(i)} \in R_j} (y^{(i)} - h(x^{(i)}))^2 = \sum_{x^{(i)} \in R_j} (y^{(i)} - b_j)^2$来求对应的$b_j$，显然有$b_j = \text{avg}(y^{(i)} | x^{(i)} \in R_j)$
- 为了划分区域，采用启发式的方法：选择第u个属性和对应的值v，作为划分属性和划分阈值，定义两个区域$R_1(u, v) = \{x | x_u \leq v\}$ 和$R_2(u, v) = \{x | x_u > v\}$，然后通过求解下式寻找最优的划分属性和划分阈值

$$\min_{u,v} \left[\min_{b_1} \sum_{x^{(i)} \in R_1(u,v)} (y^{(i)} - b_1)^2 + \min_{b_2} \sum_{x^{(i)} \in R_2(u,v)} (y^{(i)} - b_2)^2 \right]$$

再对两个区域重复上述划分，直到满足停止条件

CART回归树的生成算法

输入：训练数据集

输出：回归树$h(x)$

在训练数据集所在的输入空间中递归的将每个区域划分成两个子区域，并确定在每个区域上输出的值，构造二叉树

- 通过求解下式，求出最优划分属性和对应的最优划分阈值

$$\min_{u,v}\left[\min_{b_1}\sum_{x^{(i)}\in R_1(u,v)}(y^{(i)}-b_1)^2+\min_{b_2}\sum_{x^{(i)}\in R_2(u,v)}(y^{(i)}-b_2)^2\right]$$

- 根据求出的u,v划分区域为$R_1(u,v)=\{x|x_u\le v\}$ 和$R_2(u,v)=\{x|x_u>v\}$，并求解对应的输出值$b_j=\text{avg}(y^{(i)}|x^{(i)}\in R_j),\ j=1,2$

- 继续对两个区域重复上述步骤，知道满足停止条件为止。最终生成回归树T, 包含$|T|$个叶子节点，对应$|T|$个区域，对应的函数为

$$h(x)=\sum_{i=1}^{|T|}b_i\mathbb{I}(x\in R_i)$$

CART决策树的剪枝算法

主要包括2个步骤:

- 从生成算法产生的决策树T_0底端开始不断剪枝，直到T_0的根结点，形成子树序列$\{T_0, T_1, \cdots, T_n\}$
- 通过在独立的验证数据集上对子树序列进行测试，从中选择最优子树

Step 1. 剪枝，形成子树序列

- 计算子树的代价: $CC(T) = Err(T) + \lambda R(T)$, 这里$Err(T)$用Gini Index衡量， $R(T) = |T|$为子树叶子节点的个数
- 给定λ，必可得到训练集上代价最小的树，记为T_λ。将正则化参数λ从零逐渐增大到$+\infty$，即$0 = \lambda_0 \leq \lambda_1 \leq \cdots \leq \lambda_n \leq \infty$, 对应的最优子树序列$\{T_0, T_1, \cdots, T_n\}$是嵌套的

CART决策树的剪枝算法

Step 1. 剪枝，形成子树序列

- 从整体树T_0开始剪枝，对任意内部节点t，以其为单节点树的代价为
  $CC_\lambda(t) = Err(t) + \lambda$
- 以其为根节点的子树T_t的代价为
  $CC_\lambda(T_t) = Err(T_t) + \lambda|T_t|$
- 当λ足够小，有$CC_\lambda(T_t) < CC_\lambda(t)$, 逐步增大$\lambda$, 当$\lambda = \frac{Err(t)-Err(T_t)}{|T_t|-1}$时，$T_t$和$t$具有相同的代价，而$t$的节点更少，因此对$T_t$进行剪枝
- 为此，对T_0中的每个节点t，计算$g(t) = \frac{Err(t)-Err(T_t)}{|T_t|-1}$
- 从T_0中剪枝掉具有最小$g(t)$的节点，得到子树T_1, 同时将最小的$g(t)$置为λ_1, T_1是区间$[\lambda_1, \lambda_2)$的最优子树
- 如此剪枝下去，直到根节点，不断增加λ的值，产生新的区间

CART决策树的剪枝算法

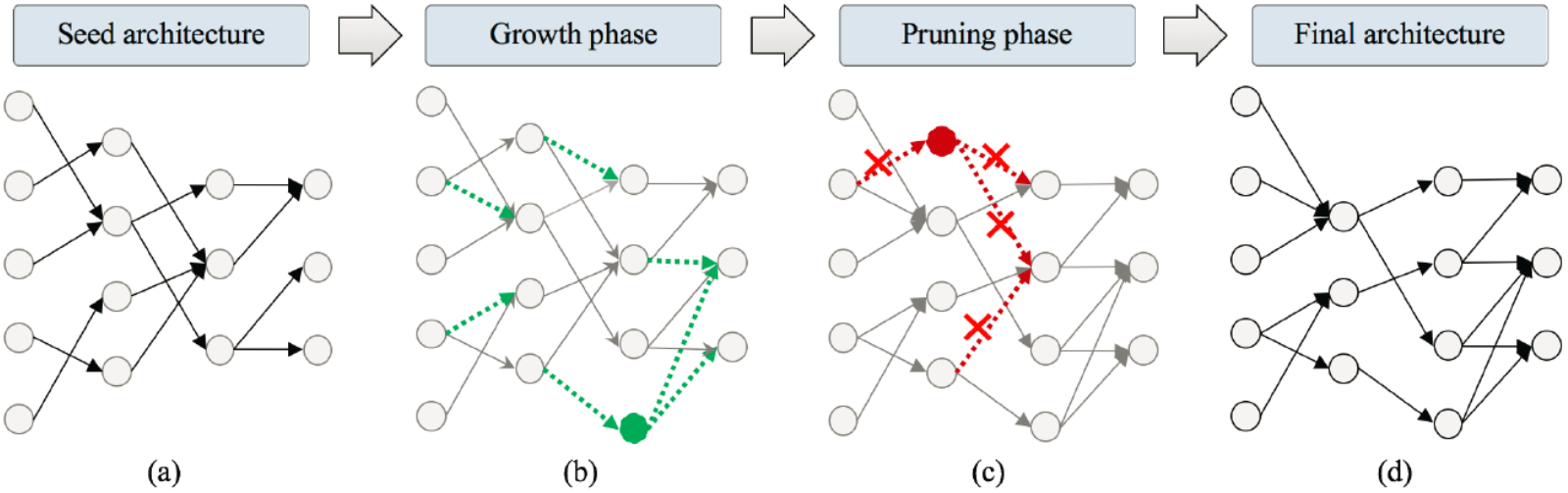
Step 2. 在剪枝得到的子树序列$\{T_0, T_1, \cdots, T_n\}$中通过交叉验证选取最优子树$T_\lambda$

- 利用独立的验证数据集，测试子树序列中各子树的基尼指数或平方误差，最小的决策树就是最优决策树

小结

- Advantages of trees
  - Easy to use, understand
  - Produce rules that are easy to interpret & implement
  - Variable selection & reduction is automatic
  - Do not require the assumptions of statistical models
  - Can work without extensive handling of missing data
- Disadvantages
  - May not perform well where there is structure in the data that is not well captured by horizontal or vertical splits
  - Since the process deals with one variable at a time, no way to capture interactions between variables

深度学习中的网络生长与剪枝



It iteratively tunes the architecture with gradient-based growth and magnitude based pruning of neurons and connections

Dai, Xiaoliang, Hongxu Yin, and Niraj K. Jha. "NeST: A Neural Network Synthesis Tool Based on a Grow-and-Prune Paradigm." *arXiv preprint arXiv:1711.02017* (2017).

Thanks!

Any questions?