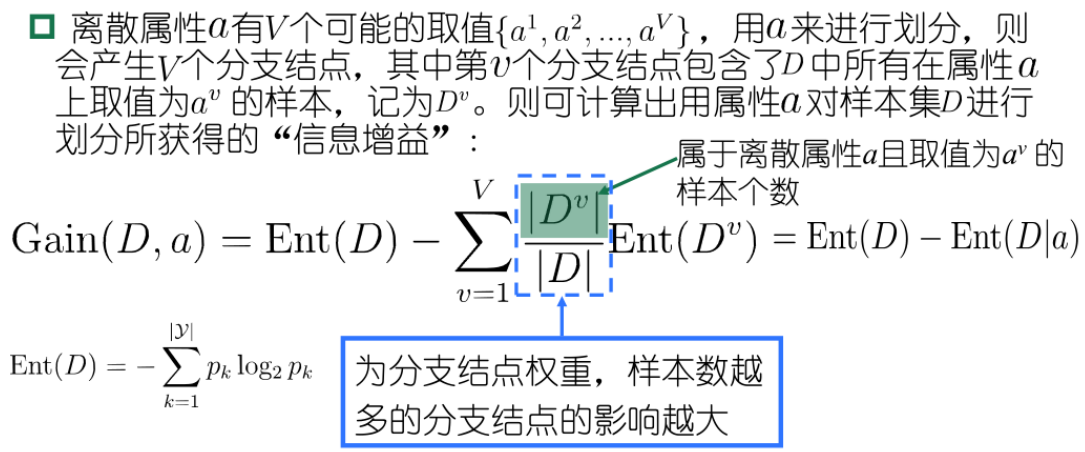
一、属性划分准则

1、信息增益：

（1）、公式：



（2）、含义：一般而言，**信息增益越大**，则意味着使用属性来进行划分所获得的**“纯度提升”越大**（信息增益代表了在一个条件下，信息复杂度（不确定性）减少的程度）

$\mathrm{Ent}(D)$ 的值越小，则 D 的纯度越高，样本的确定性越高

迭代二分器（ID3）决策树学习算法 [Quinlan, 1986] 以 信息增益为准则来选择划分属性

（3）、**信息熵表征的是数据集中的不纯度，信息熵越小表明数据集纯度越大**。ID3 决策树算法就是利用信息增益作为划分数据集的一种方法。信息增益在决策树算法中是用来选择特征的指标，信息增益越大，则这个特征的选择性越好。信息增益做特征选择的优缺点：

优点：

- *信息增益考虑了特征出现与不出现的两种情况，比较全面，一般而言效果不错。*
- *使用了所有样例的统计属性，减小了对噪声的敏感度。*
- *容易理解，计算简单。*

缺点：

- *信息增益考察的是特征对整个系统的贡献，没有到具体的类别上，所以一般只能用来做全局的特征选择，而没法针对单个类别做特征选择。*
- 只能处理离散变量，不能处理连续变量
- *算法天生偏向选择分支多的属性，容易导致 overfitting*
- 是类别越多的特征计算出的信息增益越大，易导致生成的决策树广而浅

偏好：信息增益对可取值数目较多的属性有所偏好，若在属性中存在类似编号这种属性取值数目多而对于决策无意义的属性，则生成的决策树泛化能力会很弱。

缺陷：计算量大，涉及大量的对数运算

2、增益率

增益率定义：

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

其中

$$\mathrm{IV}(a)=-\sum_{v=1}^{V}\frac{|D^v|}{|D|}\log_2\frac{|D^v|}{|D|}$$

称为属性a的“固有值” [Quinlan, 1993]，属性a的可能取值数目越多（即V越大），则$\mathrm{IV}(a)$的值通常就越大

问题：属性的可取类别数目越多，IV 越大，因此，增益率准则对可取值数目较少的属性有所偏好；计算量大，涉及大量的对数运算

优点：引入了属性固有值得概念，消除了信息增益对于属性取值数目多的属性的偏好。

C4.5 [Quinlan, 1993]使用了一个启发式：先从候选划分属性中找出信息增益高于平均水平的属性，再从中选取增益率最高的。

3、 基尼系数

基尼系数代表了模型的不纯度，基尼系数越小，则不纯度越低；CART 作为分类树时，特征属性可以是连续类型也可以是离散类型，但标签属性(或者分类属性)必须是离散类型。

- 数据集D的纯度可用“基尼值”来度量

$$\mathrm{Gini}(D)=\sum_{k=1}^{|\mathcal{Y}|}\sum_{k'\neq k}p_k p_{k'}=\sum_{k=1}^{|\mathcal{Y}|}p_k(1-p_k)=1-\sum_{k=1}^{|\mathcal{Y}|}p_k^2$$

反映了从D 中随机抽取两个样本，其类别标记不一致的概率

假定当前样本集合 D 中第 k 类样本所占的比例为 $p_k(k=1,2,\ldots,|\mathcal{Y}|)$

$\mathrm{Gini}(D)$ 越小，数据集越集中，D 的纯度越高

- 属性 a 的基尼指数定义为：

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

基尼指数（基尼不纯度）：表示在样本集合中一个随机选中的样本被分错的概率

- 应选择那个使划分后基尼指数最小的属性作为最优划分属性，即

$$a_*=\underset{a\in A}{\operatorname{argmin}}\ \mathrm{Gini\_index}(D,a)$$

优点：计算速度快，无对数运算，计算效率高；可以对连续值的属性进行划分。

缺点：趋向于孤立数据集中数量多的类，将它们分到一个树叶中

4、 属性划分策略的选择

大部分时候两种属性划分方法对于生产的决策树基本相同

（1）、若注重计算速度：其实大部分时候它们两种属性划分选择方式所生产的决策树基本相同，但 Gini 值的计算更快一些;

（2）、在机器学习标准库 scikit-learn 中默认的划分方式就是 Gini 指数，默认的决策树是 CART 树;

（3）、但是 Gini 指数的划分趋向于孤立数据集中数量多的类，将它们分到一个树叶中，而熵偏向于构建一颗平衡的树，也就是数量多的类可能分散到不同的叶子中去了。

二、剪枝策略

1、 剪枝的目的：“剪枝”是决策树学习算法对付“过拟合”的主要手段，可通过“剪枝”来一定程度避免因决策分支过多，以致于把训练集自身的一些特点当做所有数据都具有的一般性

质而导致的过拟合。

2、判断决策树泛化性能是否提升的方法

留出法：预留一部分数据用作“验证集”以进行性能评估

3、预剪枝

（1）、原理：

决策树生成过程中，对每个结点在划分前先进行估计，若当前结点的划分不能带来决策树泛化性能提升，则停止划分并将当前结点记为叶结点，其类别标记为训练样例数最多的类别

（2）、优点

- 预剪枝让决策树的很多分支没有展开，降低了过拟合风险。
- 显著减少训练时间和测试时间开销。

（3）、缺点

- 欠拟合风险：有些分支的当前划分虽然不能提升泛化性能，但在其基础上进行的后续划分却有可能导致性能显著提高。预剪枝基于“贪心”本质禁止这些分支展开，带来了欠拟合风险。

4、后剪枝

（1）、原理

先从训练集生成一棵完整的决策树，然后 **自底向上**地对非叶结点进行考察，若将该结点对应的子树替换为叶结点能带来决策树泛化性能提升，则将该子树替换为叶结点。

（2）、优点：后剪枝比预剪枝保留了更多的分支，欠拟合风险小，泛化性能往往优于预剪枝决策树

（3）、缺点：训练时间开销大：后剪枝过程是在生成完全决策树之后进行的，需要自底向上对所有非叶结点逐一考察

三、连续值和缺失值处理

1、 连续值：

- 连续属性离散化(二分法)
  - 第一步：假定连续属性 a 在样本集D上出现 n 个不同的取值，从小到大排列，记为 $a^1, a^2, \ldots a^n$ ，基于划分点 t，可将 D分为子集D_t^-和D_t^+，其中D_t^-包含那些在属性 a 上取值不大于 t 的样本，D_t^+包含那些在属性 a 上取值大于 t 的样本。考虑包含 n-1 个元素的候选划分点集合

$$T_a = \left\{ \frac{a^i + a^{i+1}}{2} \mid 1 \le i \le n-1 \right\}$$

即把区间 $[a^i, a^{i-1})$的中位点 $\frac{a^i + a^{i+1}}{2}$作为候选划分点

- 第二步：采用离散属性值方法，考察这些划分点，选取最优的划分点进行样本集合的划分

$$\begin{aligned}\mathrm{Gain}(D,a) &= \max_{t\in T_a} \mathrm{Gain}(D,a,t)\\ &= \max_{t\in T_a} \mathrm{Ent}(D) - \sum_{\lambda\in\{-,+\}} \frac{|D_t^\lambda|}{|D|}\mathrm{Ent}(D_t^\lambda)\\ &= \max_{t\in T_a} \mathrm{Ent}(D) - \left(\frac{|D_t^-|}{|D|}\mathrm{Ent}(D_t^-) + \frac{|D_t^+|}{|D|}\mathrm{Ent}(D_t^+)\right)\end{aligned}$$

其中 $\mathrm{Gain}(D,a,t)$ 是样本集 D 基于划分点 t 二分后的信息增益，于是，就可选择使 $\mathrm{Gain}(D,a,t)$ 最大化的划分点

若当前结点划分属性为连续属性，该属性还可作为其后代结点的划分属性

2、缺失值

问题 1- 如何在属性缺失的情况下进行划分属性选择?

- $\tilde{D}$ 表示 D 中在属性 a 上没有缺失值的样本子集，$\tilde{D}^v$ 表示 $\tilde{D}$ 中在属性 a 上取值为 a^v 的样本子集，$\tilde{D}_k$ 表示 $\tilde{D}$ 中属于第 k 类的样本子集

  假定为每个样本 $\boldsymbol{x}$ 赋予一个权重 w_x，并定义：

  - 无缺失值样本所占的比例

$$\rho = \frac{\sum_{x\in\tilde{D}} w_x}{\sum_{x\in D} w_x}$$

  - 无缺失值样本中第 k 类所占比例

$$\tilde{p}_k = \frac{\sum_{x\in\tilde{D}_k} w_x}{\sum_{x\in\tilde{D}} w_x} \quad (1\le k\le |\mathcal{Y}|)$$

  - 无缺失值样本中在属性 a 上取值 a^v 的样本所占比例

$$\tilde{r}_v = \frac{\sum_{x\in\tilde{D}^v} w_x}{\sum_{x\in\tilde{D}} w_x} \quad (1\le v\le V)$$

  - 基于上述定义可得推广后的信息增益计算式：

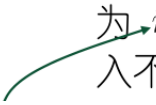
$$\begin{aligned}\mathrm{Gain}(D,a) &= \rho\times\mathrm{Gain}(\tilde{D},a)\\ &= \rho\times\left(\mathrm{Ent}(\tilde{D}) - \sum_{v=1}^{V}\tilde{r}_v\mathrm{Ent}(\tilde{D}^v)\right)\end{aligned}$$

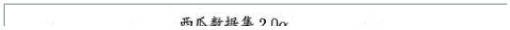
  - 其中

$$\mathrm{Ent}(\tilde{D}) = -\sum_{k=1}^{|\mathcal{Y}|}\tilde{p}_k\log_2\tilde{p}_k$$

问题2：给定划分属性，若样本在该属性上的值缺失，如何对样本进行划分

- 对于问题2
  - 若样本$\boldsymbol{x}$在划分属性a上的取值已知，则将$\boldsymbol{x}$划入与其取值对应的子结点，且样本权值在子结点中保持为w_x
  - 若样本$\boldsymbol{x}$在划分属性a上的取值未知，则将$\boldsymbol{x}$同时划入所有子结点，且样本权值在与属性值a^v对应的子结点中调整为$\tilde{r}_v \cdot w_x$（直观来看，相当于让同一个样本以不同概率划入不同的子结点中去）





缺失值问题可以从三个方面来考虑：

1．在选择划分属性的时候，训练样本存在缺失值，如何处理？

➢ 计算划分损失减少值时，忽略特征缺失的样本，最终计算的值乘以比例（实际参与计算的样本数除以总的样本数）

假如使用ID3算法，那么选择分类属性时，就要计算所有属性的信息增益(Gain)。假设10个样本，属性是a,b,c。在计算a属性熵时发现，第10个样本的a属性缺失，那么就把第10个样本去掉，前9个样本组成新的样本集，在新样本集上按正常方法计算a属性的熵增。然后结果乘9/10（新样本占raw样本的比例），就是a属性最终的熵。

2．分类属性选择完成，对训练样本分类，发现样本属性缺失怎么办？

➢ 将该样本分配到所有子结点中，权重由1变为具有属性a的样本被划分成的子集样本个数的相对比率，计算错误率的时候，需要考虑到样本权重

比如该结点是根据a属性划分，但是待分类样本a属性缺失，怎么办？假设a属性离散，有1,2两种取值，那么就把该样本分配到两个子结点中去，但是权重由1变为相应离散值个数占样本的比例。然后计算错误率的时候，注意，不是每个样本都是权重为1，存在分数。

3．训练完成，给测试集样本分类，有缺失值怎么办？

➢ 分类时，如果待分类样本有缺失变量，而决策树决策过程中没有用到这些变量，则决策过程和没有缺失的数据一样；否则，如果决策要用到缺失变量，决策树也可以在当前结点做多数投票来决定（选择样本数最多的特征值方向）。