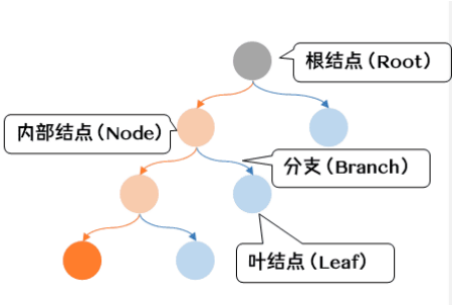
Decision Tree

Decision Tree算法核心思想

决策树是基于已知各种情况(各种特征取值)的基础上，通过构建树形决策结构来进行分析的一种方式，模拟了人类决策的过程。是一种预测模型，代表对象属性与对象值之间的映射关系。

Decision Tree结构



1. 内部结点：表示一个属性的测试(一个特征)
2. 叶结点：表示一种类型(一个类)
3. 分支(有向边)：表示一个测试输出

Decision Tree生长与最优属性的选择

生长流程

从根结点开始，测试待分类项对应的特征属性，并按照其值选择输出分支直至叶子结点。

> 决策树停止生长的三个条件：
>
> 1. 当前结点包含的样本全属于同一类别
> 2. 样本的属性取值都相同或者属性集为空
> 3. 当前结点包含的样本集合为空

最优属性选择

1. **信息熵(Entropy)**：可以用于消除不确定性所需信息量的度量，也是未知事件可能含有的信息量，可以度量样本集合纯度。

   > 数据集D中有y类，其中第k类样本占比为p_k，则信息熵的计算公式为：$Ent(D) = -\sum_{k=1}^{|y|} p_k \log_2 p_k$，$Ent(D)$的值越少则$D$的纯度越高。

2. **信息增益(Information Gain)**：度量的是选择某个属性进行划分时信息熵的变化，描述了一个特征带来的信息量多少。 ID3决策树算法选择该指标

   > $Gain(D,a) = Ent(D) - \sum_{v=1}^{V} \frac{|D^v|}{|D|} Ent(D^v)$，($v$表示第$v$个分支，$\frac{|D^v|}{|D|}$表示第$v$个分支占整个数据集的比例)$Gain(D,a)$也就是**划分前的信息熵-划分后的信息熵**。

3. **信息增益率(Gain Ratio)**：避免信息增益带来的 偏向取值较多的特征 的缺点。 C4.5决策树算法选择该指标

   > $Gain\_Ratio(D,a) = \frac{Gain(D,a)}{IV(a)}$，$IV(a) = -\sum_{v=1}^{V} \frac{|D^v|}{|D|} \log_2 \frac{|D^v|}{|D|}$。离散属性$a$可能取值数目越多则$IV(a)$的值越大。

4. **基尼系数(Gini Index)**：衡量数据的纯度，基尼系数越大表示数据集越不纯。 CART决策树算法选择该指标

   > $Gini(D) = \sum_{k=1}^{|y|} \sum_{k' \neq k} p_k p_{k'} = 1 - \sum_{k=1}^{|y|} p_k^2$，$p_k$表示第$k$类数据占总数据的比例。
   >
   > 属性a的基尼系数：$Gini\_Index(D,a) = \sum_{v=1}^{V} \frac{|D^v|}{|D|} Gini(D^v)$

过拟合与剪枝

可能出现决策树分支过多，导致过拟合。可以通过剪枝主动去掉一些分支来降低过拟合的风险。

预剪枝

在决策树生长过程中，对每个结点在划分前进行估计：如果当前结点的划分不能带来决策树泛化能力的提升，则停止划分并将当前结点标记为叶结点。

1. 训练时间开销降低，测试时间开销降低
2. 过拟合风险降低，欠拟合风险增加

后剪枝

先从训练集生成一颗完整的决策树，然后自底向上地对非叶节点进行考察：如果将该结点对应的子树替换为叶结点能带来决策树泛化能力的提升，则将该子树替换为叶结点。

1. 训练时间开销增加，测试时间开销降低
2. 过拟合风险降低，欠拟合风险基本不变

泛化性能通常优于预剪枝

连续值与缺失值处理

连续值处理

首先，决策树对离散值的处理为：选择一个最合适的特征属性，然后将集合按照这个特征属性的不同值划分为多个子集合，并不断重复。

对连续值处理采用的是**连续属性离散化**，常用的离散化策略是二分法(也是C4.5采用的策略)

1. 对特征的取值进行升序排序
2. 两个特征值之间的中点作为可能的分裂点，将数据集分为两部分，计算每个可能的分裂点的信息增益。(优化算法是：只计算分类属性发生改变的那些特征取值)
3. 选择修正后的信息增益最大的分裂点作为该特征的最佳分裂点
4. 计算最佳分裂点的信息增益率作为该特征的信息增益率

缺失值处理

Q1：如何在有缺失值的情况下进行划分属性的选择

Q2：给定划分属性，如果训练样本在该属性上的值缺失，如何对训练样本进行划分

Q3：训练完成，给测试集样本分类，有缺失值

基本思路：样本赋权，权重划分。