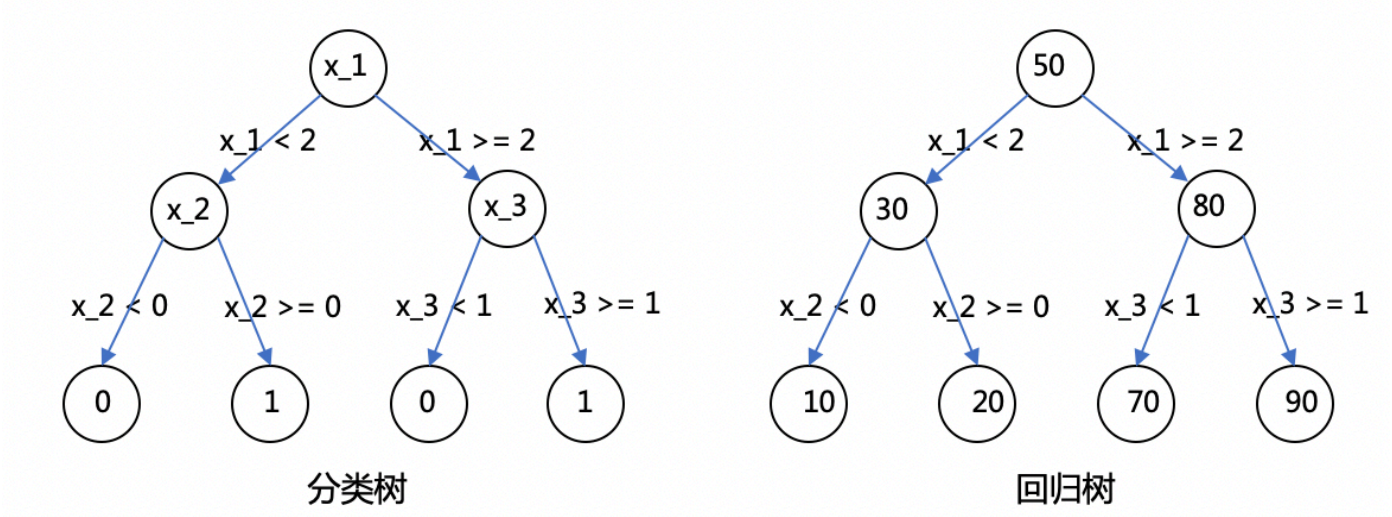
决策树模型

决策树模型分为分类树和回归树，两者都是if-then的树形原理，最大的区别为，对于分类树，最后一层叶子结点才是分类标签，其他时候的结点都不是，而回归树不同，回归树的每一个结点都是待回归属性。直观如下图所示



这里主要介绍决策分类树原理，回归树构造方式差不多。

决策树模型的逻辑是，从根节点出发，对实例的每一个特征进行判断，根据判断结果，将实例分配到其子节点中，此时，每个节点又对应着该特征的一个取值，如此递归的对实例进行判断和分配，直至将实例分配到叶子结点中，其基本楼诚遵循简单且直观的“分而治之”的策略。

本质而言，决策树模型是一个 if - then 的规则集合，根节点到叶子结点的每一条路径构建成了一条规则。这个规则有一个重要的性质——互斥且完备：

- 完备性：每个实例**都**有一条规则路径所覆盖
- 互斥性：每个实例**只**有一条规则路径所覆盖

构建决策树规则路径的依据是条件概率，这一条件概率分布定义在特征空间的一个划分上，决策树的路径就是一个划分单元，决策树分类时将该节点的实例强行分配到条件概率大的一个类别中。

决策树学习

决策树学习的本质是从训练数据集中归纳出一组分类规则，这种决策规则有可能一个也没有，也可能有很多个，这时候需要选择一个与数据集矛盾较小的决策树规则，同时又需要很好的泛化效果。

决策树使用损失函数来表示这一目标，这个损失函数通常是正则化的极大似然函数。决策树的学习策略就是以损失函数为目标函数的最小化。由于从很多决策树中选择决策树是一个NP难问题（多项式复杂程度的非确定性问题，无法按部就班的直接求解，通常使用验证解的方式近似求解），所以决策树学习算法通常是递归的选择特征对数据集进行分割，以达到最好的分类结果。

学习算法原理：构造根节点，将所有训练数据集都放到根节点，选择一个最优特征，将训练数据集分割成子集，使得训练集在当前按条件下有最好的分类。如果这些子集已经可能很好的分类，那么构建叶节点，如果还不能很好的分类，继续对其分割，构造相应的结点，如此递归进行，直至所有训练数据集被基本正确分类，或者没有合适的特征为之。

经过以上过程，决策树可能对训练集有了很好的分类能力，但是对未知数据不一定同样有很好的分类效果，所以，为了避免过拟合现象，还需要对生成的树进行剪枝，将树变得更简单，以实现更好的泛化能力。

从上面可以看出，一个决策树学习算法需要包含特征选择、决策树生成和决策树剪枝过程。常用的学习算法有ID3（Quinlan 在1986年提出），C4.5（Quinlan 在1993年提出）和CART（Breiman 在1984年提出）。

特征选择

特征选择问题

特征选择的目的是选择具有分类能力的特征，如果利用一个特征进行分类与随机分类没有区别，则这个特征就不具备分类能力。

选择哪个特征更好，这就需要一个选择特征的准则，通常选择特征的准则有信息增益与信息增益比（也叫信息增益率）。

在介绍信息增益前，先理解熵与条件熵的概念。

熵

熵常用在信息论和概率统计中，是一种表示随机变量不确定性的度量。

设 X 为一个取有限个值的离散随机变量，其概率分布为：

$$P(X=x_i)=p_i, i=1,2,\cdots,n$$

则随机变量 X 的熵 为

$$H(X)=-\sum_{i=1}^{n} p_i \log p_i$$

从上式能看出熵与 X 的取值无关，只依赖于 X 的分布，所以 X 的熵也可以写为

$$H(p)=-\sum_{i=1}^{n} p_i \log p_i$$

对于上式有：

- 若 $p_i=0$ ，则定义 $0\log 0=0$。
- 对数以 2 为底熵的单位为比特（bit）
- 对数以 e 为底熵的单位为纳特（nat）
- 易验证，$0 \le H(p) \le \log n$

假设一个伯努利分布 X ，

$$P(X=1)=p, P(X=0)=1-p, 0 \le p \le 1$$

熵为

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

显然，当 $p=\frac{1}{2}$ 时，$H(p)=1$，熵最大，随机不确定性越大，当 $p=0$ 或 $p=1$ 时，熵最小。

条件熵

对于随机变量 (X,Y) ，其联合概率分布为

$$P(X=x_i,Y=y_i)=p_{ij}, i=1,2,\cdots,n; j=1,2,\cdots,m$$

条件熵 $H(Y|X)$ 就表示在已知随机变量 X 的条件下随机变量 Y 的不确定性。定义在条件 X 下 Y 的条件概率分布的熵对 X 的数学期望

$H(Y|X)=\sum_{i=1}^{n}P(X=x_i)H(Y|X=x_i)$

信息增益就表示在得知特征 X 的条件下类别 Y 的不确定性程度。

信息增益

熵 $H(X)$ 与条件熵 $H(Y|X)$ 之差称为互信息，决策树学习中信息增益就等价于训练数据集中类与特征的互信息，对于训练数据集 D 和特征 A ，信息增益可以表示为

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

注：前面说过，决策树使用正则化的极大似然函数来作为损失函数，此时的概率不是确定的，而是通过似然函数估计出来的，所以此时的熵 $H(D)$ 和条件熵 $H(D|A)$ 分别称为经验熵和经验条件熵。

信息增益的特征选择方法为：对训练集计算其每个特征的信息增益并比较大小，选择信息增益最大的特征。

信息增益的算法如下：

> 输入：训练数据集 D 和特征 A
>
> 输出：特征 A 对训练数据集 D 的信息增益 $g(D,A)$
>
> （1）设有 k 个类 C_k （$|C_k|$ 为 C_k 的1范数，表示样本个数），计算数据集 D 的经验熵 $H(D)$
>
> $H(D)=-\sum_{k=1}^{K}\frac{|C_k|}{|D|}\log_2\frac{|C_k|}{|D|}$
>
> （2）计算特征 A 对数据集 D 的经验条件熵 $H(D|A)$
>
> $H(D|A)=\sum_{i=1}^{n}\frac{|D_i|}{|D|}H(D_i)=-\sum_{i=1}^{n}\frac{|D_i|}{|D|}\sum_{k=1}^{K}\frac{|D_{ik}|}{|D_i|}\log_2\frac{|D_{ik}|}{|D_i|}$
>
> （3）计算信息增益
>
> $g(D,A)=H(D)-H(D|A)$

信息增益比

以信息增益作为划分训练数据集的特征，存在偏向于选择取值较多的特征的问题，为了矫正这个问题，可以使用信息增益比。

$g_R(D,A)=\frac{g(D,A)}{H_A(D)}$

其中 $H_A(D)=-\sum_{i=1}^{n}\frac{|D_i|}{|D|}\log_2\frac{|D_i|}{|D|}$ ，n 为特征 A 取值的个数。

信息增益比本质： 是在信息增益的基础之上乘上一个惩罚参数。特征个数较多时，惩罚参数较小；特征个数较少时，惩罚参数较大。

不过还是有一个缺点，信息增益比会偏向于选择取值较少的特征，所以出现一种表示样本纯度的方法。

基尼系数

基尼系数的表示为

$Gini(p)=\sum_{k=1}^{K}p_k(1-p_k)=1-\sum_{k=1}^{K}p_k^2$

显然对于二分类问题，基尼系数为

$Gini(p) = 2p(1-p)$

对于给定一个样本集合D，其基尼系数为

$Gini(D) = 1 - \sum_{k=1}^{K}(\frac{|C_k|}{|D|})^2$

同样，条件熵也可以定义为

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

直观而言，$Gini(D)$ 反应了从数据集 D 中随机抽取两个样本，其类别标记不一致的概率，因此，$Gini(D)$ 越小，数据集的纯度越高。

决策树的生成

ID3算法

ID3算法的核心是在决策树各个节点上应用信息增益准则选择特征，递归的构建决策树。

> 输入：训练数据集 D，特征集 A，阈值 ϵ
>
> 输出：决策树 T
>
> （1）若 D 中所有实例都属于同一个类 C_k，则 T 为单节点树，将 C_k 作为该节点的类标记，返回 T
>
> （2）若 $A = \emptyset$，则 T 为单节点树，并将 D 中的实例数最大的类 C_k 作为该节点的类标记，返回 T
>
> （3）否则，计算特征 A 对数据集 D 的**信息增益**，选择信息增益最大的特征 A_g
>
> （4）如果 A_g 的信息增益小于阈值 ϵ，则置 T 为单节点树，并将 D 中实例数最大的类 C_k 作为该节点的类标记，返回 T
>
> （5）否则，对 A_g 的每一可能值 a_i 依 $A_g = a_i$ 将 D 划分为若干非空子集 D_i，将 D_i 中实例数最大的类作为标记，构建子节点，由节点及其子节点构成树 T，返回 T
>
> （6）对第 i 个子节点，以 D_i 为训练集，以 $A - \{A_g\}$ 为特征集，递归调用 (1)~(5)，得到子树 T，返回 T

C4.5算法

C4.5算法与ID3算法类似，只是使用信息增益比替换了信息增益。

> 输入：训练数据集 D，特征集 A，阈值 ϵ
>
> 输出：决策树 T
>
> （1）若 D 中所有实例都属于同一个类 C_k，则 T 为单节点树，将 C_k 作为该节点的类标记，返回 T
>
> （2）若 $A = \emptyset$，则 T 为单节点树，并将 D 中的实例数最大的类 C_k 作为该节点的类标记，返回 T
>
> （3）否则，计算特征 A 对数据集 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 为单节点树，并将 D 中实例数最大的类 C_k 作为该节点的类标记，返回 T

> （5）否则，对 A_g 的每一可能值 a_i 依 $A_g = a_i$ 将 D 划分为若干非空子集 D_i ，将 D_i 中实例数最大的类作为标记，构建子节点，由节点及其子节点构成树 T ，返回 T
>
> （6）对第 i 个子节点，以 D_i 为训练集，以 $A - \{A_g\}$ 为特征集，递归调用 (1)~(5) ，得到子树 T ，返回 T

CART分类树算法

CART算法（classification and regression tree），是一种分类回归树，对于其分类能力而言，整个算法过程与ID3和C4.5相似，不过与ID3和C4.5不同的是，CART使用基尼系数(Gini index)来选择划分属性。

根据训练数据集，从根节点开始，递归的对每个节点进行操作，构成一个二叉决策树。

对于节点的训练数据集 D，计算现有特征对该数据集的基尼系数，此时，每个特征 A 的每个取值 a ，都可以分割为 D^1, D^2 两部分，然后计算 $A = a$ 时的基尼系数。在所有可能的特征 A 和所有可能的切分点 a 中，选择基尼系数最小的特征及其对应的切分点作为最优特征与最优切分点。根据最优特征与最优切分点，从现在节点生成两个子节点，将该节点的训练数据集 D 依特征分配到两个子节点中去。

如此递归到进行，直到满足停止条件，生成CART分类决策树。

算法的停止条件有

- 节点中的样本个数小于设定阈值
- 样本集的基尼系数小于设定阈值
- 没有更多特征可分

CART回归树算法

CART算法既可以用于创建分类树，也可以用于创建回归树，分类是是选择最优的特征，回归实质上就是在特征维度对样本空间进行划分，不过这种特征划分就像前面所说的NP难问题，所以采用启发式的方法解决。典型CART回归树的目标函数为

$\sum_{x_i \in R_m}(y_i - f(x_i))^2$

谷，求解最优的切分变量 j 和最优分点 s 就变成求解下面的目标函数

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

所以我们只要遍历所有特征的的所有切分点，就能找到最优的切分特征和切分点。最终得到一棵回归树。

决策树的剪枝

决策树生成算法递归的生成决策树，理论上可以分类任何的训练数据，但对未知的测试数据分类很可能不准确，即过拟合，需要对决策树进行简化，去掉一些子树或叶节点，即剪枝。

决策树剪枝对基本策略分为预剪枝和后剪枝。

- 预剪枝：在决策树生成过程中，对每个节点在划分前先进行估计，若当前划分不能带来决策树泛化能力的提升，则停止划分。
- 后剪枝：先从训练集生成一颗完整的决策树，然后自下而上的对非叶节点进行考察，若将该节点对应的子树替换成叶节点能提高泛化性能，则进行替换。

预剪枝让很多分支都不会出现，这不仅降低了过拟合的风险，还降低了模型训练过程中的计算开销，但是有些分支的当前划分虽不能提高泛化性能，甚至使得泛化性能下降，但是后续的划分却有可能导致新能提升，所以预剪枝同时也带来的欠拟合的风险。

后剪枝的欠拟合风险相较于预剪枝而言会很小，泛化性能通常也优于预剪枝，但后剪枝是在生成一颗完全决策树后自底向上进行的，因此训练开销上比预剪枝大很多。

附录

附录1: 贷款申请样本数据表

ID	年龄	有工作	有自己的房子	信贷情况	类别
1	青年	否	否	一般	否
2	青年	否	否	好	否
3	青年	是	否	好	是
4	青年	是	是	一般	是
5	青年	否	否	一般	否
6	中年	否	否	一般	否
7	中年	否	否	好	否
8	中年	是	是	好	是
9	中年	否	是	非常好	是
10	中年	否	是	非常好	是
11	老年	否	是	非常好	是
12	老年	否	是	好	是
13	老年	是	否	好	是
14	老年	是	否	非常好	是
15	老年	否	否	一般	否

附录2: 使用信息增益选择最优特征

对附录1中的数据D，根据信息增益选择最优特征，那首先计算经验熵 $H(D)$.

$H(D) = -(\frac{9}{15}log_2\frac{9}{15} + \frac{6}{15}log_2\frac{6}{15}) = 0.971$

以 A_1, A_2, A_3, A_4 分别表示年龄，有工作，有自己房子和信贷情况，可以得到

$g(D, A_1) = H(D) - [\frac{5}{15}H(D_1) + \frac{5}{15}H(D_2) + \frac{5}{15}H(D_3)] = 0.083$

其中 D_1, D_2, D_3 分别为取值为青年、中年、老年的样本。

类似可得

$g(D, A_2) = H(D) - [\frac{5}{15}H(D_1) + \frac{10}{15}H(D_2)] = 0.324$

$g(D, A_3) = 0.420$

$g(D, A_4) = 0.363$

可得 A_3 信息增益最大，为最优特征。

附录3: 使用基尼系数选择最优特征和最优切割点

以 A_1, A_2, A_3, A_4 分别表示年龄，有工作，有自己房子和信贷情况

以 1，2，3 表示青年，中年和老年

以 1，2 表示有工作和有自己房子的是和否

以 1，2，3 表示信贷情况非常好，好喝一般

则

$Gini(D, A_1 = 青年) = \frac{5}{15}(2 \times \frac{2}{5} \times (1 - \frac{2}{5})) + \frac{10}{15}(2 \times \frac{7}{10} \times (1 - \frac{7}{10})) = 0.44$

$Gini(D, A_1 = 中年) = 0.48$

$Gini(D, A_1 = 老年) = 0.44$

$Gini(D, A_2 = 有工作) = 0.32$

$Gini(D, A_3 = 有自己房子) = 0.27$

$Gini(D, A_4 = 非常好) = 0.36$

$Gini(D, A_4 = 好) = 0.47$

$Gini(D, A_4 = 一般) = 0.32$

得

所有特征中划分中，$Gini(D, A_3 = 有自己房子) = 0.27$ 最小，所以 A_3 为最优特征，是有自己房子为最优切分点。

reference

统计学习方法，李航，-第2版，--清华大学出版社，2019

机器学习，周志华，-第1版，--清华大学出版社，2016

end～