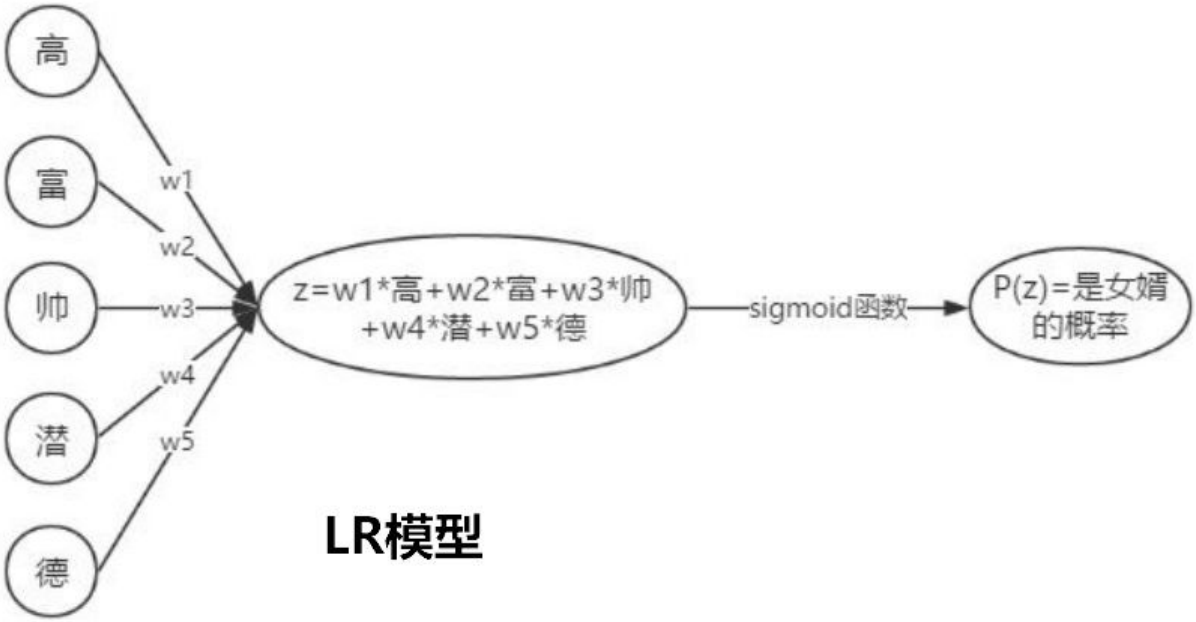
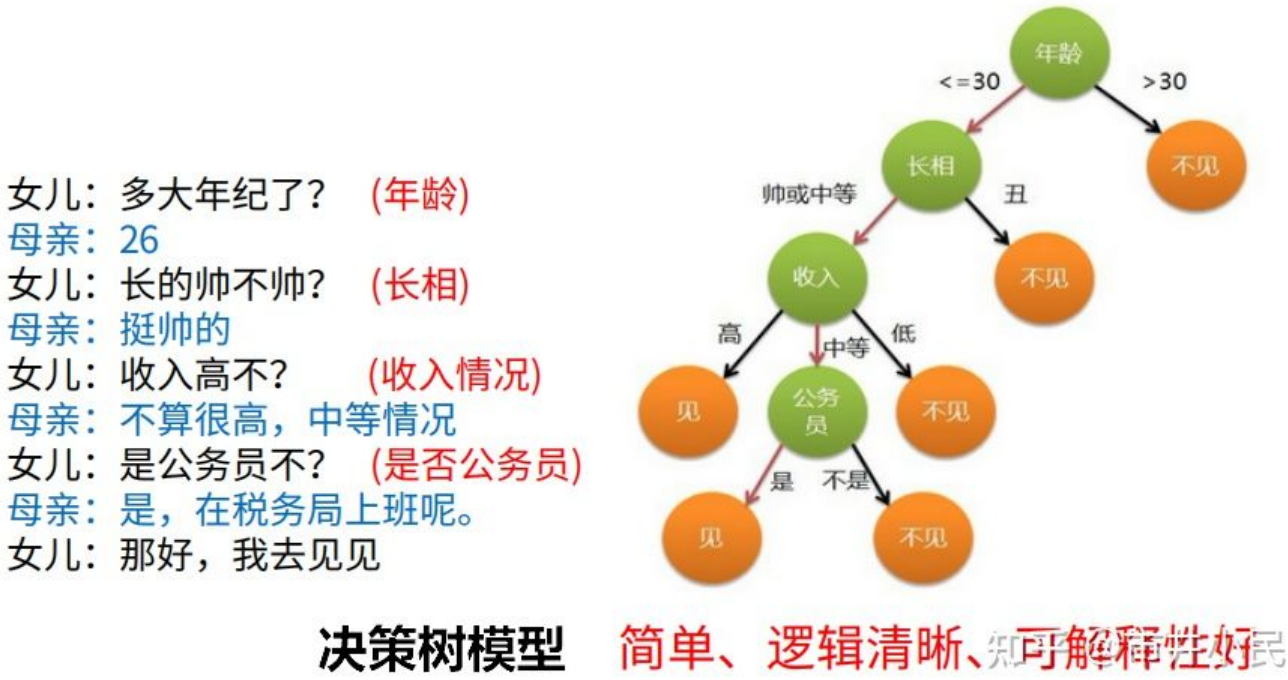
# 机器学习与深度学习面试系列六（决策树）

## 什么是决策树?

这张图片[1]能更好的理解LR模型和决策树模型算法的根本区别以及决策树的思想。我们可以思考一下一个决策（分类）问题：是否去相亲，一个女孩的母亲要给这个女孩介绍对象。




LR模型是一股脑儿的把所有特征塞入模型进行学习，而决策树更像是编程语言中的if-else一样，去做条件判断，这就是根本性的区别。

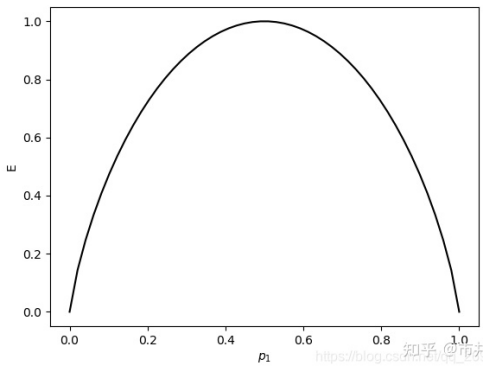
## 决策树的两个基本问题?

1. **树要怎么构造？**从若干不同的决策树中选取最优的决策树是一个NP完全问题，在实际中我们通常会采用启发式学习的方法去构建一棵满足启发式条件的决策树。
2. **树长到什么时候停止？**一棵完全生长的决策树会面临一个很严重的问题，即过拟合，所以我们需要适当控制树的复杂度，来防止过拟合。常见方法有：设置最大树高度、设置最多节点数、剪枝等。

## 信息熵和纯度是什么意思？

信息熵的定义为： $Ent(D) = -\sum_{k=1}^{|y|} p_k log_2 p_k$ ，其中 $p_k$ 表示当前样本集合D中第k类样本所占的比例。

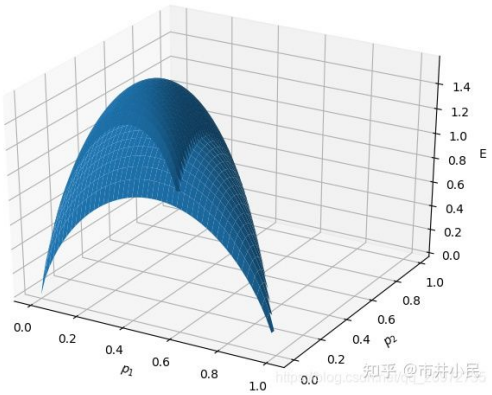
我们可以用二元信息熵的函数图像来直观看一下这个函数的意义，即只有 $p_1$ 和 $p_2$ ，并且 $p_2 = 1 - p_1$ 。




二元信息熵函数图像

可以看到当 $p_1 = 0.5$ 时，其实此时 $p_2 = 1 - p_1 = 0.5$ ，整个信息熵最大。也就是说，在样本集合D中，第1类和第2类样本比例一样，都是0.5，此时我们说这个数据集是“最混乱”的。与此相对应的观点是是此时数据集的纯度是“最低”的。当 $p_1 = 0$ 且 $p_2 = 1$ 或者 $p_1 = 1$ 且 $p_2 = 0$ 时，这时候整个数据集里只有一种分类的数据，我们说数据集的纯度是最高的，信息 很单一明确。综上：信息熵越大，数据集越混乱，纯度越低。

感兴趣的同学可以画出更多元的信息熵损失函数图像，来验证这个结论。这里贴一张别的博主的图，可以看出当三元时， $p_1 = p_2 = p_3 = \frac{1}{3}$ 信息熵最大。




三元信息熵函数图像

## 构造树的启发式方法有哪些?

由第一个问题我们知道，决策树是一种自上而下，对样本数据进行树形分类的过程，由结点和有向边组成。结点分为内部结点和叶结点，其中每个内部结点表示一个特征或属性，叶结点表示类别。从顶部根结点开始，所有样本聚在一起。经过根结点的划分，样本被分到不同的子结点中。再根据子结点的特征进一步划分，直至所有样本都被归到某一个类别(即叶结点)中。构造树的过程，其实就是选择一个特征作为树的内部节点来做分裂的过程。那么所谓构造树的启发式方法，就是如何决定当前选择哪个特征来做内部节点进行分裂。

经典的决策树模型：ID3、C4.5、CART，以他们为例分别来说明它们使用的启发式方法。他们都有一个共同的思想，就是随着树的构造，子节点的纯度越来越高。这从直觉上也很好理解，因为到最后的叶节点是最终的决策（通常为预测分类或回归结果），当一个叶节点的信息纯度很高时，我们才更有信心的说，落入这个叶节点中的样本属于某个分类。

**ID3-最大信息增益。**所谓信息增益，简单的说，就是 划分前的信息熵--划分后的信息熵，它表示的是向纯度方向迈出的“步长”。换句话说，就是我通过某个特质进行分裂的话，在下一层子节点中纯度可以提高多少。ID3选择纯度可以提高最多的那个特征。

$$Gain(D,a) = \underbrace{Ent(D)}_{\text{划分前的信息熵}} - \underbrace{\sum_{v=1}^{V} \underbrace{\frac{|D^v|}{|D|}}_{\text{第v个分支的权重，样本越多越重要}} Ent(D^v)}_{\text{划分后的信息熵}}$$

a表示一个特征，对于每个特征我们都可以计算一次G(D, a)，然后选择最大的那个。其中 $\sum_{v=1}^{V}\frac{D^v}{D}Ent(D^v)$ 又叫特征a对于数据集D的经验条件熵，重点在于理解这个思想，概念无所谓。例子可以参考西瓜书4.2.1节，不再赘述。

**C4.5——最大信息增益比。**ID3使用的信息增益有一个问题：对可取值数目较多的属性有所偏好。举个极端的例子，考虑将“样本编号”作为一个特征，在计算这个特征的信息增益时，显然所有的子节点中都只包含了一个样本，那纯度当然很高了，信息增益也是最大的。如果采用最大信息增益法，那它就会选择“样本编号”这个特征来进行分裂，实际上我们直观上知道，选择这个特征做分裂是没有意义的，因为新的预测样本编号跟之前的都不同，这个树是没有泛化能力的。

为了解决这个问题，C4.5使用的是信息增益率： $Gain\_ratio(D,a)=\frac{Gain(D,a)}{IV(D,a)}$ ，其中： $IV(D,a)=-\sum_{v=1}^{V}\frac{|D^v|}{|D|}log\frac{|D^v|}{|D|}$ 。属性a的可能取值数目越多(即V越大)，则IV(D, a)的值通常就越大。所以信息增益比本质久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。

需注意的是，增益率准则对可取值数目较少的属性有所偏好。因此，C4.5算法并不是直接选择增益率最大的候选划分属性，而是使用了一个启发式：先从候选划分属性中找出信息增益高于平均水平的属性，再从中选择增益率最高的。

**CART——最大基尼(Gini)指数。**首先给出基尼系数的定义： $Gini(D)=\sum_{k=1}^{K}\sum_{k'\neq k}p_kp_{k'}=1-\sum_{k=1}^{K}p_k^2$ ，直观来说， $Gini(D)$ 反映了从数据集 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，则数据集 D 的纯度越高。

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

于是，我们在候选属性集合A中，选择那个使得划分后基尼指数最小的属性作为最优划分属性。

## 树形结构为什么不需要特征归一化?

因为数值缩放不影响分裂点位置，对树模型的结构不造成影响。 按照特征值进行排序的，排序的顺序不变，那么所属的分支以及分裂点就不会有不同。而且，树模型是不能进行梯度下降的，构建树模型寻找最优点时是通过寻找最优分裂点完成的，因此树模型是阶跃的，阶跃点是不可导的，并且求导没意义，也就不需要归一化。

## ID3、C4.5、CART之间的区别与联系是什么?

1. ID3是采用信息增益作为评价标准，但是它对可取值数目较多的属性过于偏好。这种分类的泛化能力是非常弱的。因此，C4.5实际上是对ID3进行优化，通过引入信息增益比，一定程度上对取值比较多的特征进行惩罚，避免ID3出现过拟合的特性，提升决策树的泛化能力。而CART使用的是基尼指数作为评价标准。
2. 从样本类型的角度，ID3只能处理离散型变量，而C4.5和CART都可以处理连续型变量。C4.5处理连续型变量时，通过对数据排序之后找到类别不同的分割线作为切分点，根据切分点把连续属性转换为布尔型，从而将连续型变量转换多个取值区间的离散型变量。而对于CART，由于其构建时每次都会对特征进行二值划分，因此可以很好地适用于连续性变量。
3. 从应用角度，ID3和C4.5只能用于分类任务，而CART(Classification and Regression Tree，分类回归树)从名字就可以看出其不仅可以用于分类，也可以应用于回归任务。
4. 从缺失值处理角度，ID3对样本特征缺失值比较敏感，而C4.5和CART可以对缺失值进行不同方式的处理。例如C4.5对于含缺失值的样本以不同的概率划分到多个子节点中去。
5. 从树的结构来说，ID3和C4.5可以在每个结点上产生出多叉分支（多叉树），且每个特征在层级之间不会复 用，而CART每个结点只会产生两个分支，因此最后会形成一颗二叉树，且每个特征可以被重复使用。
6. 从过拟合处理角度来说，ID3和C4.5通过剪枝来权衡树的准确性与泛化能力，而CART 直接利用全部数据发现所有可能的树结构进行对比（可参考李航《统计机器学习》第二版5.5.2）。

## CART如何构建回归树?

CART回归树与分类树的主要区别在于分裂节点的启发式方法不同，分类树是基于基尼指数（提高子节点信息纯度）的，而回归树基于最小子节点的均方差。

CART回归树是二叉树，通过不断将特征进行分裂。比如当前树结点是基于第j个特征值进行分裂的，设该特征值小于s的样本划分为左子树，大于s的样本划分为右子树。

$$R_1(j,s) = \{x|x^{(j)} \leq s\} \; and \; R_2(j,s) = \{x|x^{(j)} > s\}$$

而CART回归树实质上就是在该特征维度对样本空间进行划分，而这种空间划分的优化是一种NP难问题，因此，在决策树模型中是使用启发式方法解决。典型CART回归树产生的目标函数为：

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



因此，当我们为了求解最优的切分特征j和最优的切分点s，就转化为求解这么一个目标函数：

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

所以，在每一次的划分中，选择切分变量（splitting variable）和切分点（splitting point）时（也就是选择 feature 和将该 feature space 一分为二的split），使得模型在训练集上的 mse 最小，也就是每片叶子的MSE之和最小。( $c_i$ 为叶节点的平均值)

在预测时，返回样本落入的叶子结点中所有训练样本的平均值即可。

## 决策树如何剪枝?

决策树的剪枝基本策略有 预剪枝 (Pre-Pruning) 和 后剪枝 (Post-Pruning)。

- **预剪枝**：其中的核心思想就是，在每一次实际对结点进行进一步划分之前，先采用验证集的数据来验证如果划分是否能提高划分的准确性。如果不能，就把结点标记为叶结点并退出进一步划分；如果可以就继续递归生成节点。缺点：有些分支的当前划分虽不能提升泛化性能、甚至可能导致泛化性能暂时下降，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给预剪枝决策树带来了欠拟含的风险。
- **后剪枝**：后剪枝则是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泛化性能提升，则将该子树替换为叶结点。一般情形下，后剪枝决策树的欠拟合风险很小，泛化性能往往优于预剪枝决策树。但后剪枝过程是在生成完全决策树之后进行的，并且要自底向上地对树中的所有非叶结点进行逐一考察，因此其训练时间开销比未剪枝决策树和预剪枝决策树都要大得多.

例子可以参考西瓜书4.3节。

## 参考

1. ^ https://github.com/NLP-LOVE/ML-NLP/blob/master/Machine%20Learning/3.Desition%20Tree/Desition%20Tree.md