决策树学习笔记（一）：特征选择

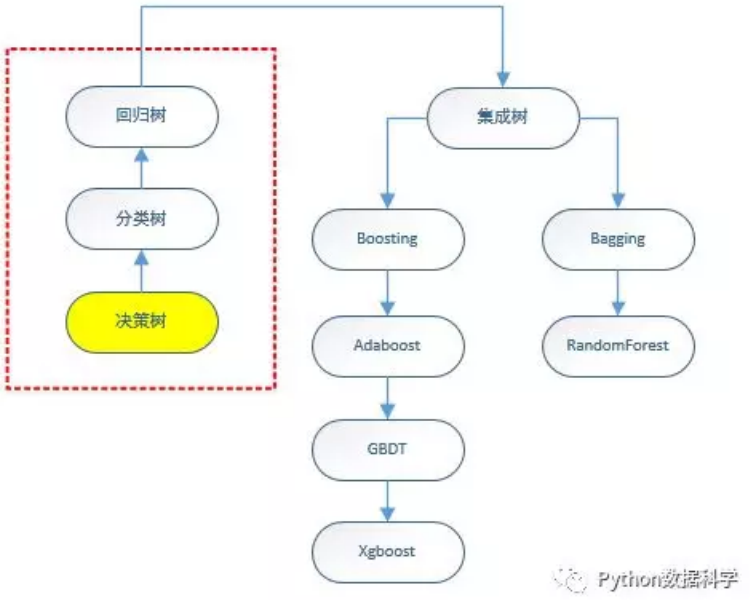
原创 wLsq Python数据科学 2019-01-06

收录于话题
#Python数据科学 60 #机器学习 18

> 作者：xiaoyu
> 介绍：一个半路转行的数据挖掘工程师

相信很多朋友已经对决策树很熟悉了，决策树是机器学习中的一种基本的可用于分类与回归的方法，它是一些集成学习如GBDT，XGboost等复杂模型的基础。这些高级模型比如XGboost可以非常好地拟合数据，在数据挖掘比赛以及工业界中都有着非常出色的表现，受到了无数爱好者的追捧。

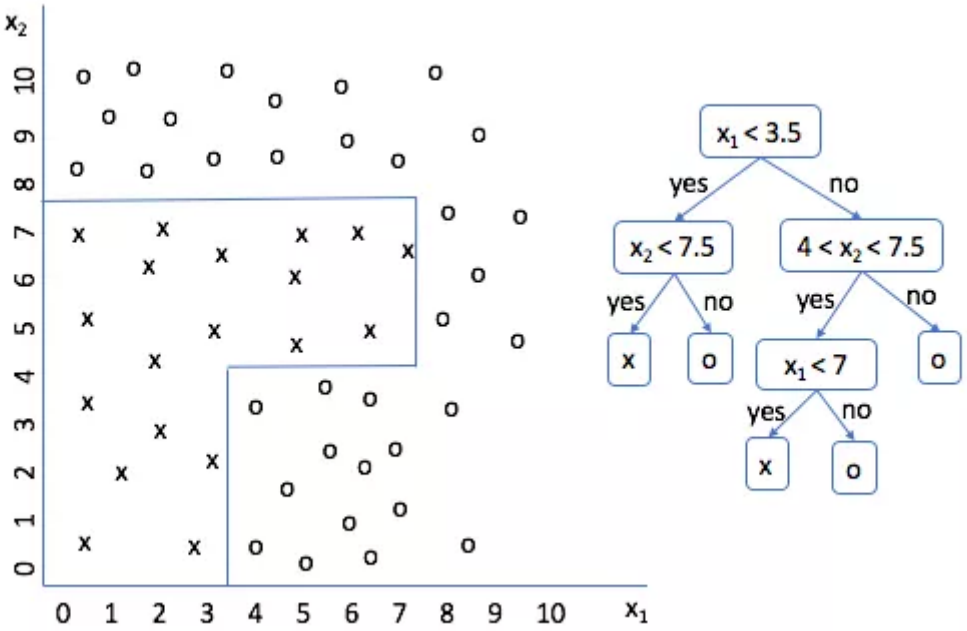
有的朋友可能觉得XGboost很牛逼，上来就要学GBDT，XGboost。我并不建议这么做，我个人对整个树模型的学习过程总结了一个流程：



本篇开始进入树模型系列，从最简单的决策树开始，按照上面这个学习流程，循序渐进，直到复杂模型Xgboost。

决策树概述

不同于逻辑回归，决策树属于**非线性模型**，可以用于分类，也可用于回归。它是一种树形结构，可以认为是if-then规则的集合，是以实例为基础的归纳学习。基本思想是自顶向下，以信息增益（或信息增益比，基尼系数等）为度量构建一颗度量标准下降最快的树，每个内部节点代表一个属性的测试，直到叶子节点处只剩下同一类别的样本。它的决策流程如下所示：



决策树的学习包括三个重要的步骤，**特征选择**，**决策树的生成**以及**决策树的剪枝**。

- **特征选择**：常用的特征选择有信息增益，信息增益比，基尼系数等。

- **生成过程**：通过计算信息增益或其它指标，选择最佳特征。从根结点开始，递归地产生决策树，不断的选取局部最优的特征，将训练集分割成能够基本正确分类的子集。
- **剪枝过程**：首先定义决策树的评价指标，对于所有的叶子结点，累加计算每个叶子结点中的（样本数）与其（叶子节点熵值）的乘积，以叶子数目作为正则项（它的系数为剪枝系数）。然后计算每个结点的剪枝系数，它的大概含义是删除该结点的子树，损失不变的前提下，正则项系数的值为多少，这个值越小说明该子树越没有存在的必要。依次选取剪枝系数最小的结点剪枝，得到决策树序列，通过交叉验证得到最优子树。

特征选择

对于特征选择，常用的特征选择指标有**信息增益，增益率，基尼指数**。

信息增益 (Information Gain)

提到信息增益必须先解释一下什么是“信息熵”，因为信息增益是基于信息熵而定义的。我们先给出**信息熵**的公式定义：

$$H(D) = -\sum_{k=1}^{K} p_k log_2 p_k$$

D：表示**当前**的数据集合。
k：表示当前数据集合中的第k类，也就是我们目标变量的类型。

最后，解释下信息熵公式的由来：

$$H(X) = -\sum_{x \in \mathcal{X}} p(x) \log p(x)$$

信息论之父克劳德·香农，总结出了信息熵的三条性质：

- **单调性，即发生概率越高的事件，其所携带的信息熵越低。极端案例就是“太阳从东方升起”，因为为确定事件，所以不携带任何信息量。从信息论的角度，认为这句话没有消除任何不确定性。**
- **非负性，即信息熵不能为负。这个很好理解，因为负的信息，即你得知了某个信息后，却增加了不确定性是不合逻辑的。**
- **累加性，即多随机事件同时发生存在的总不确定性的量度是可以表示为各事件不确定性的量度的和。写成公式就是：**

事件 $X = A, Y = B$ 同时发生，两个事件相互独立 $p(X = A, Y = B) = p(X = A) \cdot p(Y = B)$ ，

那么信息熵 $H(A, B) = H(A) + H(B)$ 。

香农从数学上，严格证明了满足上述三个条件的随机变量不确定性度量函数具有唯一形式：

$$H(X) = -C \sum_{x \in \mathcal{X}} p(x) \log p(x)$$

其中的 C 为常数，我们将其归一化为 $C = 1$ 即得到了信息熵公式。

补充一下，如果两个事件不相互独立，那么满足

$H(A, B) = H(A) + H(B) - I(A, B)$ ，其中 $I(A, B)$ 是互信息（mutual information），代表一个随机变量包含另一个随机变量信息量的度量，这个概念在通信中用处很大。

有很多朋友不是很明白为什么要选择这个指标来度量。其实简单来说，可以将信息熵理解为**“不纯度”**的意思。纯度高意味着在数据集里我们要分类的某一种类型占比很高，纯度低意味着分类的各个类型占比近似很难区分。

举一个例子，比如我们想分类A和B，用公式量化来体现就是：

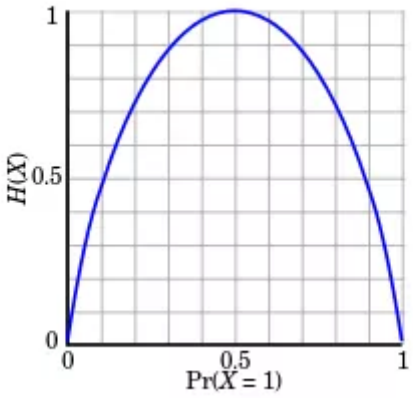
1）如果分类结果中A和B各占50%，那么意味着分类结果很失败，这无异于随机地乱猜，完全没起到分类效果，公式计算结果如下：

$$
\begin{aligned}
H(D) &= -(p_1 log_2 p_1 + p_2 log_2 p_2) \\
&= -(\frac{1}{2} log_2 \frac{1}{2} + \frac{1}{2} log_2 \frac{1}{2}) \\
&= 1
\end{aligned}
$$

2）如果分类结果中A占比100%，B占比0%或者B占比100%，A占比0%时，那么意味着分类很成功，因为我们成功地区分了A和B，我们就说此时的纯度很高，公式计算结果如下：

$$
\begin{aligned}
H(D) &= -(p_1 log_2 p_1 + p_2 log_2 p_2) \\
&= -(\frac{2}{2} log_2 \frac{2}{2} + \frac{0}{2} log_2 \frac{0}{2}) \\
&= 0
\end{aligned}
$$

信息熵的取值范围为0~1，值越大，越不纯，相反值越小，代表集合纯度越高，如下图所示。



上面信息熵只能代表不纯度，并不能代表信息量。但既然我们有了不纯度，我们只要将分类前后的不纯度相减，那就可以得到一种**“纯度提升值”**的指标，我们把它叫做**"信息增益"**，公式如下：

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

$$H(D|A) = \sum_{i=1}^{n} \frac{|D_i|}{|D|} H(D_i)$$
$$= -\sum_{i=1}^{n} \frac{|D_i|}{|D|} \sum_{k=1}^{K} \frac{|D_{ik}|}{|D_i|} log_2 \frac{|D_{ik}|}{|D_i|}$$

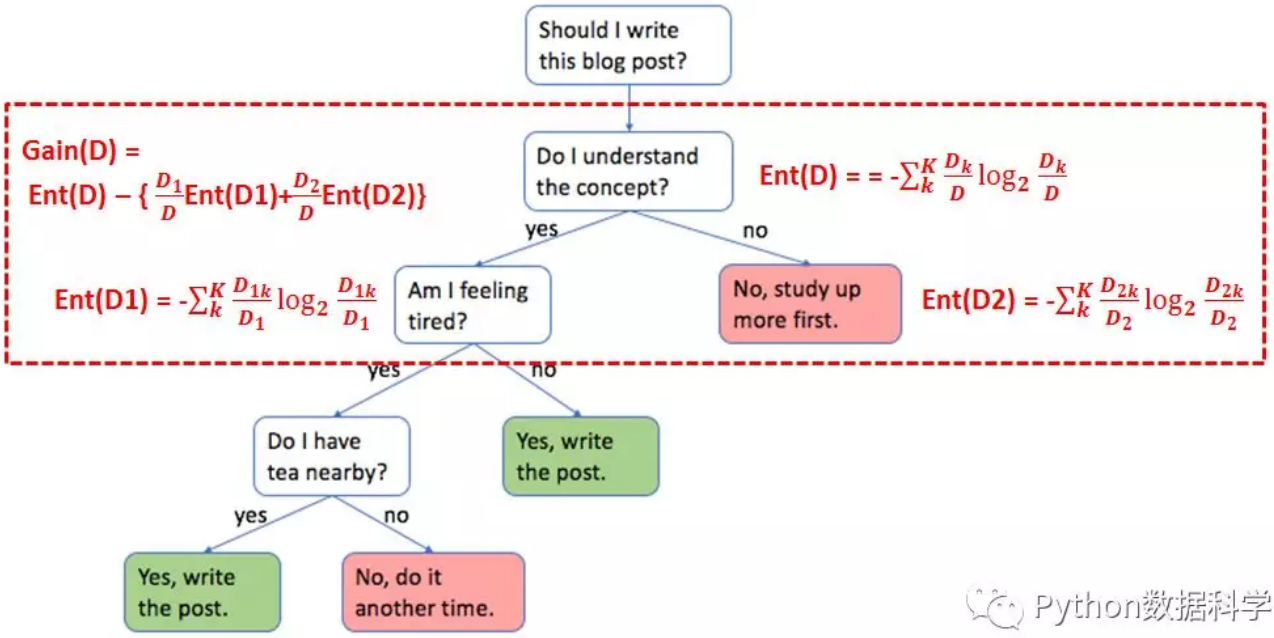
基于特征A对数据集D的划分：

- H(D|A)：表示基于特征A（分类后）的信息熵，也称条件熵。
- Di：基于特征A对数据集D划分的子集。
- |Di|/|D|：考虑到特征分类子集的数据量不同，给每个子集赋予了权重。
- n：为特征A的分类总数，即有多少个Di。

基于目标分类对特征类别Di数据集的划分：

- Dik：每个特征分类子集Di中目标分类后的子集。
- K：为目标分类的类别总数，即有多少个Dik。
- |Dik|/|Di|：每个特征分类子集中，各目标分类子子集所占比例。

简单来说，先基于特征A进行划分，再基于目标变量进行划分，这是一个嵌套的过程。举一个例子说明，红色框内代表决策树中的其中一个分类过程，按照“是否理解内容”这个特征分成两类，树的父集和子集信息熵都已经标出，因此信息增益Gain就可以计算出来。



上面只是描述了针对一个特征的计算信息增益的过程。在实际决策过程中，我们需要将所有的特征分类后的信息增益都计算一遍，然后选择其中最大的一个最为本次的分类特征。因此也就达到了特征选择的目的。ID3算法使用信息增益的方法来选择特征。

> 从这个过程，我们可以发现：最开始选择的特征肯定是提供信息量最大的，因为它是遍历所有特征后选择的结果。因此，**按照决策过程中特征从上到下的顺序，我们也可以将特征的重要程度进行排序。**这也就解释了为什么树模型有feature_importance这个参数了。

增益率（信息增益比）

信息增益可以很好的度量特征信息量，但却在某些情况下有一些弊端，举一个例子说明。

比如对于编号这个特征，我们知道一般编号值都是各不相同的，因此有多少个编号就需要分为多少类。由于每一个分类中只有一个编号值，即纯度已经最大，所以导致编号这个特征的信息增益最大，而实际上它并不是最优的特征，这样选择决策树也显然不具备泛化能力。

这正是信息增益的一个弊端：**对可取值数目较多的属性有所偏好。**因为信息增益反映的是给定一个条件以后不确定性减少的程度，必然是分得越细的数据集确定性更高，也就是条件熵越小，信息增益越大。

为了减少这种偏好带来的不利影响，**“增益率”**这个指标诞生了。替代地，增益率通过引入一个被称作分裂信息(Split information)的项来惩罚取值较多的Feature，分裂信息用来衡量Feature分裂数据的广度和均匀性。增益率定义如下：

$$Gain\_ratio(D,a) = \frac{Gain(D,A)}{Split\_Information(D,A)}$$

$$Split\_Information(D,A) = -\sum_{i=1}^{n} \frac{|D_i|}{|D|} log_2 \frac{|D_i|}{|D|}$$

- Gain_ratio(D,A)：代表基于特征A的增益率。
- Split_Information(D,A)：代表对属性A信息增益的平衡项。也可以理解为特征A的一种内在属性。

因此，属性A的取值数目越多，平衡项的值越大，增益率也就越小，这也反映出增益率对取值数据较少的属性有所偏好。C4.5算法就是利用增益率来选择特征。

基尼指数

与信息增益和增益率类似，基尼指数是另外一种度量指标，由CART决策树使用，其定义如下：

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

对于二类分类问题,若样本属于正类的概率为 p，则基尼指数为：

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

对于给定的样本集合D，其基尼指数定义为：

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

其中Ck是D中属于第k类的样本子集。

如果样本集合D被某个特征A是否取某个值分成两个样本集合D1和D2，则在特征A的条件下，集合D的基尼指数定义为：

$$Gini(D,A) = \frac{|D_1|}{|D|}Gini(D_1) + \frac{|D_2|}{|D|}Gini(D_2)$$

基尼指数Gini(D)反应的是集合D的不确定程度，跟熵的含义相似。Gini(D,A)反应的是经过特征A划分后集合D的不确定程度。所以决策树分裂选取Feature的时候，要选择使基尼指数最小的Feature，但注意信息增益则是选择最大值，这个值得选取是相反的。

再看看下图，其实基尼指数,熵，误分类率的曲线非常接近。

图 5.7 显示二类分类问题中基尼指数 Gini(p)、熵（单位比特）之半$\frac{1}{2}H(p)$和分类误差率的关系．横坐标表示概率 p，纵坐标表示损失．可以看出基尼指数和熵之半的曲线很接近，都可以近似地代表分类误差率．

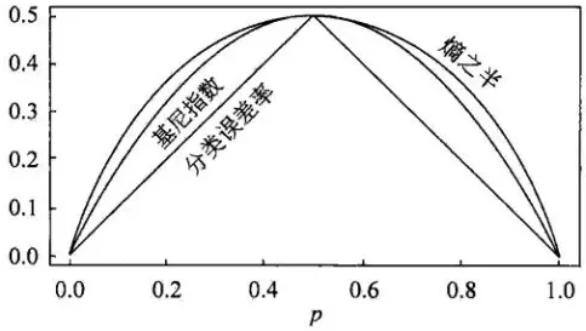


图 5.7　二类分类中基尼指数、熵之半和分类误差率的关系

总结

本篇介绍了决策树中的一个非常重要的步骤：**特征选择。**分别介绍了三种选择度量指标，信息增益，增益率，基尼指数。这三种指标也分别对应着三种算法ID3，C4.5，CART。

下一篇将会介绍三种算法，也包括分类和回归的算法流程和Python实现。最后介绍决策树的剪枝，如何处理缺失值，以及决策树的优缺点和应用。

参考：

[1] 统计学习方法，李航

[2] 机器学习，周志华

[3] http://leijun00.github.io/2014/09/decision-tree/

[4] https://www.jianshu.com/p/ca21d2392e73