昵称: 做梦当财神
园龄: 2年5个月
粉丝: 28
关注: 26
+加关注

< 2020年3月 >

日	一	二	三	四	五	六
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	1	2	3	4
5	6	7	8	9	10	11

搜索

找找看
谷歌搜索

常用链接

我的随笔
我的评论
我的参与
最新评论
我的标签

随笔分类

jupyter notebook(2)
Linux(2)
MongoDB(29)
MySQL(1)
Numpy(9)
Oracle(12)
Pandas(18)
Python(22)
Python数据分析与挖掘(22)
机器学习(41)
爬虫(2)

随笔档案

2019年8月(3)
2019年7月(13)
2019年6月(2)
2019年5月(6)
2019年4月(7)
2019年3月(11)
2019年2月(1)
2019年1月(8)
2018年12月(1)
2018年9月(6)
2018年7月(4)
2018年6月(5)
2018年5月(4)
2018年4月(20)

决策树算法原理(CART分类树)

决策树算法原理(ID3，C4.5)

CART回归树

决策树的剪枝

在决策树算法原理(ID3，C4.5)中，提到C4.5的不足，比如模型是用较为复杂的熵来度量，使用了相对较为复杂的多叉树，只能处理分类不能处理回归。对这些问题，CART(Classification And Regression Tree)做了改进，可以处理分类，也可以处理回归。

1. CART分类树算法的最优特征选择方法

ID3中使用了信息增益选择特征，增益大优先选择。C4.5中，采用信息增益比选择特征，减少因特征值多导致信息增益大的问题。CART分类树算法使用基尼系数来代替信息增益比，基尼系数代表了模型的不纯度，基尼系数越小，不纯度越低，特征越好。这和信息增益（比）相反。

假设K个类别，第k个类别的概率为p_k，概率分布的基尼系数表达式：

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

如果是二分类问题，第一个样本输出概率为p，概率分布的基尼系数表达式为：

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

对于样本D，个数为|D|，假设K个类别，第k个类别的数量为$|C_k|$，则样本D的基尼系数表达式：

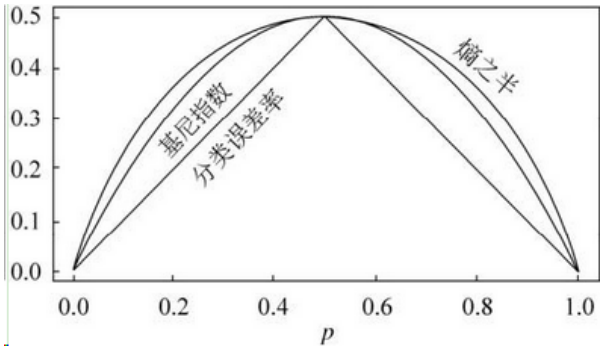
$$Gini(D)=1-\sum_{k=1}^{K}(\frac{|C_k|}{|D|})^2$$

对于样本D，个数为|D|，根据特征A的某个值a，把D分成|D1|和|D2|，则在特征A的条件下，样本D的基尼系数表达式为：

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

比较基尼系数和熵模型的表达式，二次运算比对数简单很多。尤其是二分类问题，更加简单。

和熵模型的度量方式比，基尼系数对应的误差有多大呢？对于二类分类，基尼系数和熵之半的曲线如下：



基尼系数和熵之半的曲线非常接近，仅在45度角附近误差稍大。因此，基尼系数可以做为熵模型的一个近似替代。

CART分类树算法每次仅对某个特征的值进行二分，而不是多分，这样CART分类树算法建立起来的是二叉树，而不是多叉树。

2. CART分类树算法对连续特征和离散特征的处理

CART分类树算法对连续值的处理，思想和C4.5相同，都是将连续的特征离散化。唯一区别在选择划分点时，C4.5是信息增益比，CART是基尼系数。

具体思路：m个样本的连续特征A有m个，从小到大排列a_1，a_2，……，a_m，则CART取相邻两样本值的平均数做划分点，一共取m-1个，其中第i个划分点T_i表示为：$T_i = (a_i + a_{i+1})/2$。分别计算以这m-1个点作为二元分类点时的基尼系数。选择基尼系数最小的点为该连续特征的二元离散分类点。比如取到的基尼系数最小的点为a_t，则小于a_t的值为类别1，大于a_t的值为类别2，这样就做到了连续特征的离散化。

注意的是，与ID3、C4.5处理离散属性不同的是，如果当前节点为连续属性，则该属性在后面还可以参与子节点的产生选择过程。

CART分类树算法对离散值的处理，采用的思路：不停的二分离散特征。

在ID3、C4.5，特征A被选取建立决策树节点，如果它有3个类别A1,A2,A3，我们会在决策树上建立一个三叉点，这样决策树是多叉树。

CART采用的是不停的二分。会考虑把特征A分成{A1}和{A2,A3}、{A2}和{A1,A3}、{A3}和{A1,A2}三种情况，找到基尼系数最小的组合，比如{A2}和{A1,A3}，然后建立二叉树节点，一个节点是A2对应的样本，另一个节点是{A1,A3}对应的样本。由于这次没有把特征A的取值完全分开，后面还有机会对子节点继续选择特征A划分A1和A3。这和ID3、C4.5不同，在ID3或C4.5的一颗子树中，离散特征只会参与一次节点的建立。

3. CART分类树算法具体流程

CART分类树建立算法流程，之所以加上建立，是因为CART分类树算法有剪枝。

算法输入训练集D，基尼系数的阈值，样本个数阈值。

输出的是决策树T。

算法从根节点开始，用训练集递归建立CART分类树。

(1)、对于当前节点的数据集为D，如果样本个数小于阈值或没有特征，则返回决策子树，当前节点停止递归。

(2)、计算样本集D的基尼系数，如果基尼系数小于阈值，则返回决策树子树，当前节点停止递归。

(3)、计算当前节点现有的各个特征的各个特征值对数据集D的基尼系数，对于离散值和连续值的处理方法和基尼系数的计算见第二节。缺失值的处理方法和C4.5算法里描述的相同。

(4)、在计算出来的各个特征的各个特征值对数据集D的基尼系数中，选择基尼系数最小的特征A和对应的特征值a。根据这个最优特征和最优特征值，把数据集划分成两部分D1和D2，同时建立当前节点的左右节点，做节点的数据集D为D1，右节点的数据集D为D2。

(5)、对左右的子节点递归的调用1-4步，生成决策树。

对生成的决策树做预测的时候，假如测试集里的样本A落到了某个叶子节点，而节点里有多个训练样本。则对于A的类别预测采用的是这个叶子节点里概率最大的类别。

4. CART回归树建立算法

CART回归树

CART回归树和CART分类树的建立类似，这里只说不同。

(1)、分类树与回归树的区别在样本的输出，如果样本输出是离散值，这是分类树；样本输出是连续值，这是回归树。分类树的输出是样本的类别，回归树的输出是一个实数。

(2)、连续值的处理方法不同。

(3)、决策树建立后做预测的方式不同。

分类模型：采用基尼系数的大小度量特征各个划分点的优劣。

回归模型：采用和方差度量，度量目标是对于划分特征A，对应划分点s两边的数据集D1和D2，求出使D1和D2各自集合的均方差最小，同时D1和D2的均方差之和最小。表达式为：

$$\min_{A,s}\left[\min_{c_1}\sum_{x_i\in D_1(A,s)}(y_i-c_1)^2+\min_{c_2}\sum_{x_i\in D_2(A,s)}(y_i-c_2)^2\right]$$

其中，c1为D1的样本输出均值，c2为D2的样本输出均值。

最新评论

阅读排行榜

评论排行榜

推荐排行榜

对于决策树建立后做预测的方式，CART分类树采用叶子节点里概率最大的类别作为当前节点的预测类别。回归树输出不是类别，采用叶子节点的均值或者中位数来预测输出结果。

5、CART树算法的剪枝

CART树的生成：基于训练数据集，递归构建二叉决策树。CART树的剪枝：用验证数据集对生成的树进行剪枝并选择最优子树，损失函数最小作为剪枝的标准。

CART分类树的剪枝策略在度量损失的时候用基尼系数；CART回归树的剪枝策略在度量损失的时候用均方差。

决策树很容易对训练集过拟合，导致泛化能力差，所以要对CART树进行剪枝，即类似线性回归的正则化。CART采用后剪枝法，即先生成决策树，然后产生所有剪枝后的CART树，然后使用交叉验证检验剪枝的效果，选择泛化能力最好的剪枝策略。

剪枝损失函数表达式：$C_\alpha(T_t) = C(T_t) + \alpha|T_t|$

α为正则化参数(和线性回归的正则化一样)，$C(T_t)$为训练数据的预测误差，$|T_t|$是子树T叶子节点数量。

当α = 0时，即没有正则化，原始生成的CART树即为最优子树。当α = ∞时，正则化强度最大，此时由原始的生成CART树的根节点组成的单节点树为最优子树。当然，这是两种极端情况，一般来说，α越大，剪枝剪的越厉害，生成的最优子树相比原生决策树就越偏小。对于固定的α，一定存在使得损失函数$C_\alpha(T_t)$最小的唯一子树。

剪枝的思路：

对于位于节点t的任意一颗子树T_t，如果没有剪枝，损失函数是：

$$C_\alpha(T_t) = C(T_t) + \alpha|T_t|$$

如果将其剪掉，仅保留根节点，损失函数是：

$$C_\alpha(T) = C(T) + \alpha$$

当α = 0或α很小，$C_\alpha(T_t) < C_\alpha(T)$，当α增大到一定程度时

$$C_\alpha(T_t) = C_\alpha(T)$$

当α继续增大时不等式反向，即满足下式：

$$\alpha = \frac{C(T) - C(T_t)}{|T_t| - 1}$$

T_t和T有相同的损失函数，但T节点更少，因此可以对子树T_t进行剪枝，也就是将它的子节点全部剪掉，变为一个叶子结点T。

交叉验证策略：

如果我们把所有节点是否剪枝的值α都计算出来，然后针对不同α对应的剪枝后的最优子树做交叉验证。这样可以选择最好的α，有了这个α，用对应的最优子树作为最终结果。

有了上面的思路，CART树的剪枝算法：

输入是CART树建立算法得到的原始决策树T。

输出是最优决策树T_α。

算法过程：

(1)、初始化$\alpha_{min} = \infty$，最优子树集合ω = {T}。

(2)、从叶子结点开始自下而上计算内部节点 t 的训练误差损失函数$C_\alpha(T_t)$（回归树为均方差，分类树为基尼系数），叶子节点数$|T_t|$，以及正则化阈值$\alpha = min\{\frac{C(T)-C(T_t)}{|T_t|-1}, \alpha_{min}\}$，更新$\alpha_{min} = \alpha$

(3)、得到所有节点的α值得集合M。

(4)、从M中选择最大的值α_k，自上而下的访问子树 t 的内部节点，如果$\frac{C(T)-C(T_t)}{|T_t|-1} \leq \alpha_k$时，进行剪枝。并决定叶子节点 t 的值。如果是分类树，这是概率最高的类别，如果是回归树，这是所有样本输出的均值。这样得到α_k对应的最优子树T_k

(5)、最优子树集合$\omega = \omega \cup T_k$，$M = M - \{\alpha_k\}$。

(6)、如果M不为空，则回到步骤4。否则就已经得到了所有的可选最优子树集合ω。

(7)、采用交叉验证在ω选择最优子树T_α。

6. CART算法小结

算法	支持模型	树结构	特征选择	连续值处理	缺失值处理	剪枝
ID3	分类	多叉树	信息增益	不支持	不支持	不支持
C4.5	分类	多叉树	信息增益比	支持	支持	支持
CART	分类回归	二叉树	基尼系数 均方差	支持	支持	支持

CART算法缺点：

(1)、无论ID3，C4.5，CART都是选择一个最优的特征做分类决策，但大多数，分类决策不是由某一个特征决定，而是一组特征。这样得到的决策树更加准确，这种决策树叫多变量决策树(multi-variate decision tree)。在选择最优特征的时，多变量决策树不是选择某一个最优特征，而是选择一个最优的特征线性组合做决策。代表算法OC1。

(2)、样本一点点改动，树结构剧烈改变。这个通过集成学习里面的随机森林之类的方法解决。

7. 决策树算法小结

这里不纠结ID3、C4.5、CART，这部分来自scikit-learn英文文档。

优点：

1. 简单直观，生成的决策树很直观。
2. 基本不需要预处理，不需要提前归一化和处理缺失值。
3. 使用决策树预测的代价是$O(\log_2 m)$。m为样本数。
4. 既可以处理离散值也可以处理连续值。很多算法只是专注于离散值或者连续值。
5. 可以处理多维度输出的分类问题。
6. 相比于神经网络之类的黑盒分类模型，决策树在逻辑上可以很好解释。
7. 可以交叉验证的剪枝来选择模型，从而提高泛化能力。
8. 对于异常点的容错能力好，健壮性高。

缺点：

1. 决策树算法非常容易过拟合，导致泛化能力不强。可以通过设置节点最少样本数量和限制决策树深度来改进。
2. 决策树会因为样本发生一点的改动，导致树结构的剧烈改变。这个可以通过集成学习之类的方法解决。
3. 寻找最优的决策树是一个NP难题，我们一般是通过启发式方法，容易陷入局部最优。可以通过集成学习的方法来改善。
4. 有些比较复杂的关系，决策树很难学习，比如异或。这个就没有办法了，一般这种关系可以换神经网络分类方法来解决。
5. 如果某些特征的样本比例过大，生成决策树容易偏向于这些特征。这个可以通过调节样本权重来改善。

分类: 机器学习

做梦当财神
关注 - 26
粉丝 - 28
+加关注

1 0

« 上一篇： K-Means聚类算法原理
» 下一篇： CART回归树

相关博文：
· 决策树算法原理(下)
· 决策树模型 ID3/C4.5/CART算法比较
· 决策树
· 2.决策树（DecisionTree）-ID3、C4.5、CART比较
· 决策树简单介绍（一） 决策树理论基础
» 更多推荐...

最新 IT 新闻:
· 微软Universal Print云端打印解决方案已转入私有预览
· 刘强东："非典"时京东还是小公司，特别能体会疫情对企业的影响
· Gitee遭受DDoS攻击，官方建议不要在hosts里绑定IP地址
· 疫情下的口罩"疯狂制造"：物料飚涨40倍，3天回本
· Facebook在Microsoft Store上永久删除其桌面应用程序
» 更多新闻...