决策树的理论与编程实践

倪杰

2021 年 3 月 7 日

摘要

决策树是一种常见的机器学习方法。以分类问题为例，对其类别的判定要基于一系列的判断或者“子决策”。这一系列的子决策构成整个决策树从顶到底的一条“路径”。决策树学习算法最著名的代表有 ID3，C4.5 和 CART。

目录

1 决策树节点的划分与生成 **1**

1.1 信息熵与信息增益 2

1.2 信息增益率 2

1.3 基尼指数 2

2 决策树基本算法实现思路 **2**

3 决策树的剪枝 **3**

4 简单的代码实现 **3**

1 决策树节点的划分与生成

一般的，一棵决策树包含一个根节点，若干个内部节点和若干个叶节点。叶节点对应于决策结果，其他每个节点则对应于一个属性测试。每个节点包含的样本集合根据属性测试的结果被划分到子节点中。决策树学习的关键在于如何选择最优划分属性。一般而言，我们希望分支样本尽可能属于同一类别，即节点的纯度越来越高。

1.1 信息熵与信息增益

假设当前样本集合 D 中第　类样本所占的比例为 p_k, 则 D 的信息熵定义为

$$Ent(D) = \sum_{k=1}^{|y|} p_k log_2 p_k \tag{1}$$

Ent(D) 的值越小，则 D 的纯度越高在分支节点根据属性 a 划分前后，可以通过”信息增益”衡量划分所得的纯度提升

$$Gain(D,a) = Ent(D) - \sum_{v=1}^{|V|} \frac{|D^v|}{|D|} Ent(D^v) \tag{2}$$

一般而言，信息增益越大，则意味着使用该属性进行划分所获得的“纯度提升”越大。因此，我们可用信息增益来进行决策树的划分属性选择，即在当前节点选择属性 $arg\,\underset{a\in A}{max}\, Gain(D,a)$

1.2 信息增益率

信息增益准则对可取值数目较多的属性有偏好，增益率可以减少这种偏好带来的不利影响

$Gainratio(D,a) = \frac{Gain(D,a)}{IV(a)}$, 其中 $IV(a) = -\sum_{v=1}^{V} \frac{|D^v|}{|D|} log_2 \frac{|D^v|}{|D|}$

1.3 基尼指数

基尼值:

$Gini(D) = \sum_{k=1}^{|y|} \sum_{k'\neq k} p_k p_{k'}$

基尼指数定义为:

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

2 决策树基本算法实现思路

类似贪心，每一步根据信息增益或其他标准选择划分属性和划分值。

采用深度或者广度遍历选择划分节点

划分中止条件:

子样本全部属于同一类别-标记为这个类别

子样本属性取值全部相同-选子样本里标记最多的类
(深度高于给定的值 d/划分节点个数多于值 n)
(子样本个数小于给定的值 m)
……

3 决策树的剪枝

决策树往往会一直跑到终止条件，产生过多的节点，会产生过拟合的现象。决策树的剪枝算法可以有效的防止这个问题。剪枝分为预剪枝和后剪枝。简而言之，预剪枝就是在划分节点前就根据验证集精度是否提升来判断是否有必要进行这个划分。后剪枝就是在决策树生成后自底向上判断每个节点是否有必要划分，若无则替换为叶节点。

4 简单的代码实现

```
from sklearn.datasets import load_iris
from sklearn.model_selection  import cross_val_score
import sklearn.tree as st
import random
import sklearn.utils as us
iris = load_iris()
x = iris.data
y = iris.target
x,y=us.shuffle(x,y,random_state=7)
dt=st.DecisionTreeClassifier(criterion='entropy',max_depth=10)
scores = cross_val_score(dt, x, y, cv=10, scoring='accuracy')
print(scores.mean())
```