



浙江大学数学科学学院

前沿数学专题讨论 PR01

28 Sept 2022

Submitted To:

张南松老师

22-23 秋冬学期

Submitted By :

吴凡

农业工程 + 统计学

Contents

1 问题背景	2
2 基本算法	4
3 ID3 算法模型	4
4 C4.5 算法模型	6
5 CART 决策树模型	7
6 决策树的剪枝	9
7 代码实例	9

1 问题背景

小明看过以下 6 部电影，请你根据这 6 部电影的特点来推断小明是否喜欢看某部电影。

小明观影记录

序号	片名
1	疯狂动物城
2	美国队长2
3	龙珠Z：复活的弗利萨
4	速度与激情8
5	战狼II
6	赛尔号大电影6：圣者无敌

为建立决策树模型，引入 2 部小明没看过的电影作为对照。



决策树模型的任务：从给定的训练数据集学得一个模型对新示例进行分类。可看作对“当前示例属于这类吗？”这个问题的“决策”或“判定”过程。

这个决策的过程通过决策树来完成。决策过程中提出的每个判定问题都是对某个属性的”测试”。

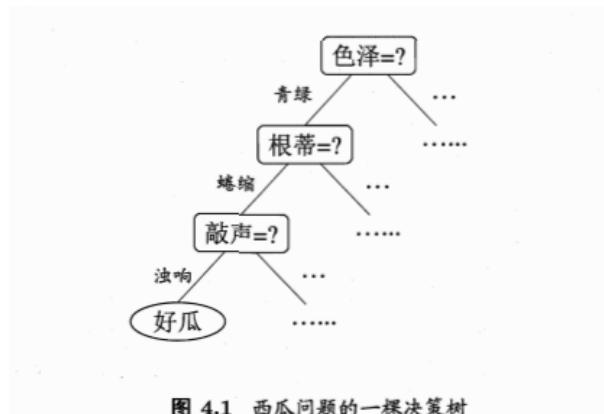


图 4.1 西瓜问题的一棵决策树

决策树学习的目的：产生一棵泛化能力强，即处理未见示例能力强的决策树。

2 基本算法

决策树模型基本流程遵循简单且直观的“分而治之”策略。

算法流程如下：

```

输入: 训练集  $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$ ;
       属性集  $A = \{a_1, a_2, \dots, a_d\}$ .
过程: 函数 TreeGenerate( $D, A$ )
1: 生成结点 node;
2: if  $D$  中样本全属于同一类别  $C$  then
3:   将 node 标记为  $C$  类叶结点; return
4: end if
5: if  $A = \emptyset$  OR  $D$  中样本在  $A$  上取值相同 then
6:   将 node 标记为叶结点, 其类别标记为  $D$  中样本数最多的类; return
7: end if
8: 从  $A$  中选择最优划分属性  $a_*$ ;
9: for  $a_*$  的每一个值  $a_*^v$  do
10:   为 node 生成一个分支; 令  $D_v$  表示  $D$  中在  $a_*$  上取值为  $a_*^v$  的样本子集;
11:   if  $D_v$  为空 then
12:     将分支结点标记为叶结点, 其类别标记为  $D$  中样本最多的类; return
13:   else
14:     以 TreeGenerate( $D_v, A \setminus \{a_*\}$ ) 为分支结点
15:   end if
16: end for
输出: 以 node 为根结点的一棵决策树

```

3 ID3 算法模型

一般而言，随着划分过程不断进行，我们希望决策树的分支结点所包含的样本尽可能属于同一类别，即结点的“纯度”(purity)越来越高。

“信息熵”为度量样本集合纯度常用的一种指标，假定当前样本集合 D 中第 k 类样本所占的比例为 $p_k (k = 1, 2, \dots, |y|)$ ，则 D 的信息熵定义为

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

$Ent(D)$ 的值越小，则 D 的纯度越高。

假定离散属性 a 有 V 个可能的取值 $\{a^1, a^2, \dots, a^V\}$, 若使用 a 来对样本集 D 进行划分, 则会产生 V 个分支结点, 其中第 v 个分支结点包含了 D 中所有在属性 a 上取值为 a^v 的样本, 记为 D^v , 可计算出 $Ent_a(D)$, 于是可计算出用属性 a 对样本集 D 进行划分所获得的”信息增益”

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

一般而言, 信息增益越大, 则意味着使用属性 a 来进行划分所获得的”纯度提升”越大, 因此可选择属性 $a_* = argmax_{a \in A} Gain(D, a)$

举例:

小明观影记录					
序号	片名	类型	产地	票房	是否观影
1	龙珠Z:复活的弗利萨	动漫	日本	低1,100万	1
2	美国队长3	科幻	美国	低124,637万	1
3	疯狂动物城	动漫	美国	低153,035万	1
4	速度与激情8	动作	美国	高267,098万	1
5	战狼2	动作	中国	高548,461万	1
6	赛尔号大电影6：圣者无敌	动漫	中国	低9758.2万	1
7	星级特工	科幻	法国	低32,047万	0
8	叶问3	动作	中国	低77,031万	0

$$Ent(D) = -(\frac{6}{8} * ln\frac{6}{8} + \frac{2}{8} * ln\frac{2}{8}) = 0.8112$$

$$Ent(D) = \frac{3}{8} * (-\frac{2}{3} * ln\frac{3}{3} - \frac{1}{3} * ln\frac{1}{3}) + \frac{2}{8} * (-\frac{1}{2} * ln\frac{1}{2} - \frac{1}{2} * ln\frac{1}{2}) + \frac{3}{8} * (-\frac{2}{3} * ln\frac{2}{3} - \frac{1}{3} * ln\frac{1}{3}) = 0.5944$$

$$\text{因此, } Gain(D, \text{产地}) = Ent(D) - Ent(D) = 0.2168$$

$$\text{同理可求得, } Ent(D) = 0.3444 \quad Ent(D) = 0.6887$$

$$Gain(D, \text{类型}) = Ent(D) - Ent(D) = 0.4668$$

$$Gain(D, \text{票房}) = Ent(D) - Ent(D) = 0.1225$$

因此, $Gain(D, \text{产地}) > Gain(D, \text{类型}) > Gain(D, \text{票房})$

属性”产地”的信息增益最大, 于是它被选为划分属性, 随后, 决策树学习算法对每个分支结点做进一步划分。



上述过程为决策树算法中最常用的算法模型——ID3 算法模型。

4 C4.5 算法模型

按照信息增益准则， $Gain(D, a) = Ent(D) - \sum_{v=1}^V \frac{|D^v|}{|D|} Ent(D^v)$ ，对可取值数目较多的属性有所偏好，如在“小明观影”的例子中，若把“序号”作为划分属性，此时， $Ent(D) = 6 * \frac{1}{8} * (-\frac{1}{1} * \ln \frac{1}{1}) + 2 * \frac{1}{8} * (-\frac{0}{1} * \ln \frac{0}{1}) = 0$ ，则信息增益非常大，而，若模型以“序号”作为划分属性，显然，此时这些分支结点的纯度已达最大，然而，这样的决策树显然不具有泛化能力，无法对新样本进行有效预测。

C4.5 算法模型不直接使用信息增益，而是使用“增益率”(gain ratio) 来选择最优划分属性。

$$Gain_{ratio}(D, a) = \frac{Gain(D, a)}{IV(a)} \quad (3)$$

其中， $IV(a) = -\sum_{v=1}^V \frac{|D^v|}{|D|} \log_2 \frac{|D^v|}{|D|}$ 为属性 a 的“固有值”，属性 a 的可能取值数目越多，则 IV(a) 的值通常会越大。

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。

5 CART 决策树模型

CART 决策树模型采用“基尼系数”(Gini index) 来选择划分属性

$$Gini(D) = 1 - \sum_{k=1}^{|y|} p_k^2 \quad (4)$$

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， $Gini(D)$ 越小，则数据集 D 的纯度越高。

Gini计算示例

$$GINI = 1 - \sum_{i=1}^c p(i)^2$$

C1	0	$P(C1) = 0/6 = 0$	$P(C2) = 6/6 = 1$
C2	6	$Gini = 1 - P(C1)^2 - P(C2)^2 = 1 - 0 - 1 = 0$	

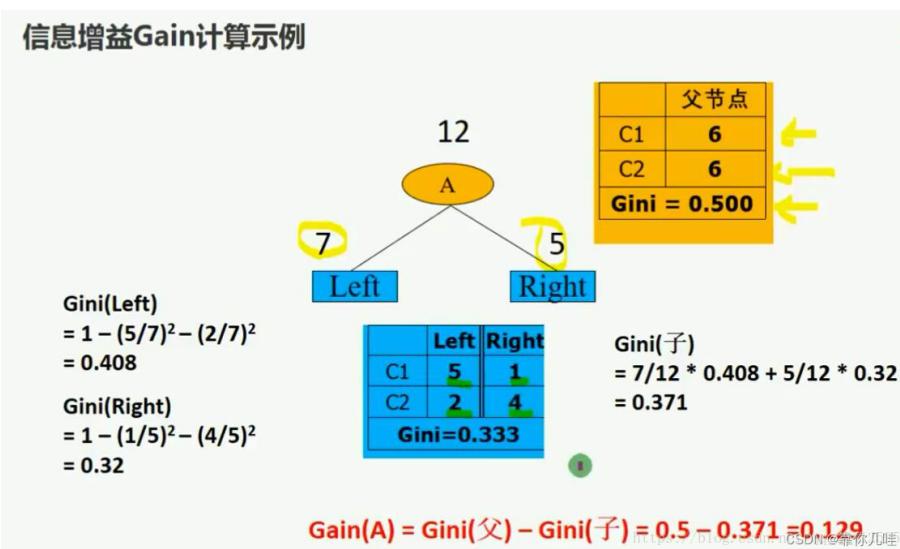
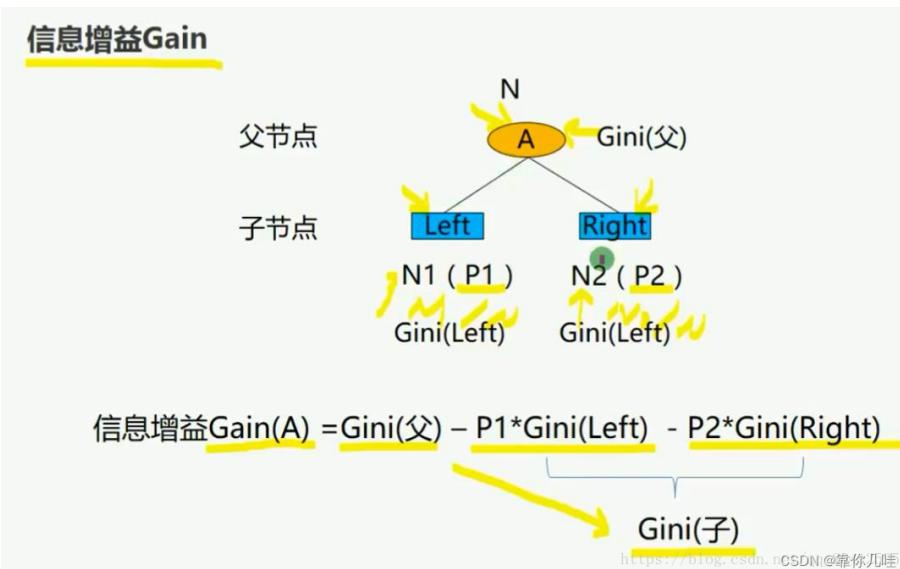
C1	1	$P(C1) = 1/6$	$P(C2) = 5/6$
C2	5	$Gini = 1 - (1/6)^2 - (5/6)^2 = 0.278$	

C1	2	$P(C1) = 2/6$	$P(C2) = 4/6$
C2	4	$Gini = 1 - (2/6)^2 - (4/6)^2 = 0.444$	<small>https://blog.csdn.net/CSDNi@靠你几壁</small>

属性 a 的基尼指数定义为：

$$Gini_{index}(D, a) = \sum_{v=1}^V \frac{|D^v|}{|D|} Gini(D^v) \quad (5)$$

于是，选择使得划分后基尼指数最小的属性作为最优划分属性，即 $a_* = argmin Gini_{index}(D, a)$



6 决策树的剪枝

决策树生成算法递归地产生决策树，直到不能继续下去为止，这样产生的树往往会出现过拟合现象。产生过拟合的原因在于学习时过多地考虑如何提高对训练数据的正确分类，从而构建出过于复杂的决策树。解决这个问题的办法是对已生成的决策树进行简化。

在决策树学习中将已生成的树进行简化的过程称为剪枝。具体是从已生成的树上裁掉一些子树或叶节点，并将其根节点或父节点作为新的叶节点，从而简化分类树模型。

剪枝策略分为预剪枝 (prepruning) 和后剪枝 (post-pruning)。

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，若当前节点的划分不能带来决策树泛化性能提升，则停止划分并将当前节点标记为叶节点。

后剪枝：先从训练集中生成一棵完整的决策树，然后自底向上地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带来决策树泛化能力的提升，则将该子树替换成叶节点。

7 代码实例

使用银行营销数据集,这些数据共包括 16 个特征,输出标签有 2 个(“是”、“否”),即客户是否已订阅定期存款。我们将采用其中的三个特征 “年龄”、“年平均余额”以及 “该月与该客户最后一次联系的日期” 对客户是否订阅定期存款来进行预测。

属性信息

- age 年龄 (数字)
- job 职务: 职务类型 (类别: “管理员”, “蓝领”, “企业家”, “女佣”, “管理”, “退休”, “自雇”, “服务”, “学生”, “技术员”, “待业”, “未知”)
- marital 婚姻: 婚姻状况 (类别: “已婚”, “离婚”, “单身”, “未知”)
- education 教育 (类别: “未知”, “中学”, “小学”, “高等教育”)
- default 违约: 信用违约吗? (类别: “否”, “是”)
- balance 收支: 年平均余额
- housing 住房: 有住房贷款吗? (分类: “否”, “是”)
- loan 贷款: 有个人贷款吗? (类别: “否”, “是”)

等等

输出类别信息: 客户是否已定阅?

```
import pandas as pd

bm = pd.read_csv('./bank/bank.txt', sep=';')

X = bm[['age', 'balance', 'day']].values[:200, :]
y = bm['y'].values[:200]

bm.head()
```

将数据集划为 70% 作为训练集, 30% 划分为测试集。

	age	job	marital	education	default	balance
0	30	unemployed	married	primary	no	1787
1	33	services	married	secondary	no	4789
2	35	management	single	tertiary	no	1350
3	30	management	married	tertiary	no	1476
4	59	blue-collar	married	secondary	no	0

5 rows × 17 columns

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=1, stratify=y)
```

使用 scikit-learn 训练一个决策树

```
from sklearn.tree import DecisionTreeClassifier

# 采用gini系数进行度量，树的最大深度设置为4
tree = DecisionTreeClassifier(criterion='gini',
                               max_depth=4,
                               random_state=1)

# 训练
tree.fit(X_train, y_train)
```

DecisionTreeClassifier() 函数参数:

```
class sklearn.tree.DecisionTreeClassifier(criterion='gini', splitter=
                                         'best', max_depth=None,
                                         min_samples_split=2,
                                         min_samples_leaf=1,
                                         min_weight_fraction_leaf=0.0,
                                         max_features=None, random_state=
                                         None, max_leaf_nodes=None,
                                         min_impurity_decrease=0.0,
                                         min_impurity_split=None,
                                         class_weight=None, presort=False)
```

criterion: 默认 gini, 即 CART 算法。

max depth: 决策树最大深度，默认值是'None'. 一般数据比较少或者特征少的时候可以不用管这个值，如果模型样本数量多，特征也多时，推荐限制这个最大深度，具体取值取决于数据的分布。常用的可以取值 10 到 100 之间，常用来解决过拟合。

random state: 是用来设置决策树分枝中随机模式的参数，在高维度时 sklearn 决策树的特征随机性会很明显，低维度的数据（比如鸢尾花数据集），随机性几乎不会显现。高维数据下我们设置 random state 并配合 splitter 参数可以让模型稳定下来，保证同一数据集下是决策树结果可以多次复现，便于模型参数优化。

计算模型准确率：

```
from sklearn.metrics import accuracy_score
y_train_pre = tree.predict(X_train)
y_test_pre = tree.predict(X_test)
print("training accuracy: %.3f" % accuracy_score(y_train_pre, y_train))
print("testing accuracy: %.3f" % accuracy_score(y_test_pre, y_test))
```

Output:

training accuracy: 0.907

testing accuracy: 0.817

使用 Graphviz 库进行对决策树进行可视化：

```
from pydotplus import graph_from_dot_data
from sklearn.tree import export_graphviz

# 生成 .dot 文件
dot_data = export_graphviz(tree,
                           filled=True,
                           rounded=True,
                           class_names=['yes',
                                         'no'],
                           feature_names=['年龄',
                                         '收支',
                                         '联系日期'],
                           out_file=None)
```

```
# export_graphviz(tree, out_file='tree.dot', feature_names=['age', 'balance', 'duration'])

# 将 .dot 文件存为图像文件
graph = graph_from_dot_data(dot_data)

# 将图像文件写入
graph.write_png('tree.png')
```

