

**模式识别大作业**

题 目 决策树及其优化算法

学 院 信息科学与工程

专 业 控制科学与工程

组 员 王 琨

指导教师 赵海涛

**完成日期： 2019 年 12 月11日**

**目 录**

[1.算法原理 2](#_Toc26974643)

[2.经典决策树算法 2](#_Toc26974644)

[2.3 ID3算法 2](#_Toc26974645)

[2.2 C4.5算法 4](#_Toc26974646)

[2.3 CART算法 5](#_Toc26974647)

[2.4 C5.0算法 6](#_Toc26974648)

[2.5 总结 7](#_Toc26974649)

[3.优化算法——随机森林 8](#_Toc26974650)

[3.1随机森林的概述 8](#_Toc26974651)

[3.2 随机森林算法 9](#_Toc26974652)

[4.面向大数据的决策树算法 10](#_Toc26974653)

[4.1 对大数据的集中式算法——SLIQ 10](#_Toc26974654)

[4.2 对大数据的分布式算法——PLANET 11](#_Toc26974655)

[4.3 面向流数据的算法——VFDT 12](#_Toc26974656)

[4.4 总结 12](#_Toc26974657)

[5.总结与展望 12](#_Toc26974658)

[参考资料 13](#_Toc26974659)

**模式识别作业报告——决策树及其优化算法**

组员：王 琨

## ****算法原理****

决策树（Decision Tree）是一种简单但是广泛使用的分类器，属于有监督学习（Supervised Learning）。决策树是一种树形结构，其中每个内部节点表示一个属性上的判断，每个分支代表一个判断结果的输出，最后每个叶节点代表一种分类结果。通过训练数据构建决策树，可以高效的对未知的数据进行分类。决策树有两大优点：

（1）决策树模型可以读性好，具有描述性，有助于人工分析；

（2）效率高，决策树只需要一次构建，反复使用，每一次预测的最大计算次数不超过决策树的深度。

## 经典决策树算法

决策树本质上是寻找最纯净的划分方法，这个最纯净在数学上叫纯度，纯度通俗点理解就是目标变量要分得足够开。另一种理解可以认为是分类误差率的一种衡量。实际决策树算法往往用到的是不纯度的公式。不纯度的选取有多种方法，每种方法也就形成了不同的决策树方法，下面的三种经典决策树算法中，ID3算法使用信息增益作为不纯度；C4.5算法使用信息增益率作为不纯度；CART算法使用基尼系数作为不纯度。

### 2.3 ID3算法

ID3算法以信息论为基础，以信息熵和信息增益为衡量标准，从而实现对数据的归纳分类。在ID3算法中，选择信息增益最大的属性作为当前的特征对数据集分类。ID3算法是以信息熵和信息增益作为衡量标准的分类算法。通过不断的选择特征对数据集不断划分。

#### 2.3.1 信息熵

熵的概念主要是指信息的混乱程度，变量的不确定性越大，熵的值也就越大，熵的公式可以表示为：

为D中第k类样本所占的比例。设a为离散属性：

信息增益越大，意味着使用属性a来进行划分所获得的“纯度”提升越大。

#### 2.3.2 递归构建决策树

从数据集构造决策树算法所需的子功能模块工作原理如下：得到原始数据集，然后基于最好的属性值划分数据集，由于特征值可能多于两个，因此可能存在大于两个分支的数据集划分，第一次划分之后，数据将被向下传递到树分支的下一个节点，在此节点在此划分数据，因此可以使用递归的原则处理数据集。

递归结束的条件，是程序完全遍历所有划分数据集的属性，或者每个分支下的所有实例都具有相同的分类，如果所有实例具有相同的分类，则得到一个叶子节点或者终止块，任何到达叶子节点的数据必然属于叶子节点的分类。

相关例子：根据以下属性对食堂的饭菜是否好吃进行划分。

**表2.1 食堂饭菜的相关属性**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 编号 | 荤素搭配 | 味道咸淡 | 色泽搭配 | 价格高低 | 香气类型 | 口感软硬 | 好吃 |
| 1 | 荤素 | 适中 | 丰富 | 实惠 | 诱人 | 软 | 是 |
| 2 | 荤 | 适中 | 一般 | 实惠 | 诱人 | 软 | 是 |
| 3 | 荤 | 适中 | 丰富 | 实惠 | 诱人 | 软 | 是 |
| 4 | 荤素 | 适中 | 一般 | 实惠 | 诱人 | 软 | 是 |
| 5 | 素 | 适中 | 丰富 | 实惠 | 诱人 | 软 | 是 |
| 6 | 荤素 | 淡 | 丰富 | 稍贵 | 平淡 | 硬 | 是 |
| 7 | 荤 | 淡 | 丰富 | 实惠 | 平淡 | 硬 | 是 |
| 8 | 荤 | 淡 | 丰富 | 实惠 | 平淡 | 软 | 是 |
| 9 | 荤 | 淡 | 一般 | 稍贵 | 平淡 | 软 | 否 |
| 10 | 荤素 | 咸 | 糟糕 | 实惠 | 刺激 | 硬 | 否 |
| 11 | 素 | 咸 | 糟糕 | 很贵 | 刺激 | 软 | 否 |
| 12 | 素 | 适中 | 丰富 | 很贵 | 刺激 | 硬 | 否 |
| 13 | 荤素 | 淡 | 丰富 | 稍贵 | 诱人 | 软 | 否 |
| 14 | 素 | 淡 | 一般 | 稍贵 | 诱人 | 软 | 否 |
| 15 | 荤 | 淡 | 丰富 | 实惠 | 平淡 | 硬 | 否 |
| 16 | 素 | 适中 | 丰富 | 很贵 | 刺激 | 软 | 否 |
| 17 | 荤素 | 适中 | 一般 | 很贵 | 平淡 | 软 | 否 |

判断饭是否好吃：|y|=2，正例，反例

计算根节点的信息熵：

计算当前属性集合{口感软硬 色泽搭配 荤素搭配 味道咸淡 香气类型 价格高低}中每个属性的信息增益。

荤素搭配={荤素 荤 素}

（荤素）= {1 4 6 10 13 17}，正例，反例，

(荤) = {2 3 7 8 9 15}，正例，反例，

(素) ={5 11 12 14 16}，正例，反例，

同理可得：

，,

可以看出，“价格高低”的信息增益最大，则第一个根节点为“价格高低”，向下进行三个分支，根据分类后自左向右分别进行分支计算，如对左一，按照“实惠”分类后得到一组数据，再对该组数据进行基于（实惠）计算各属性的信息增益、基于（稍贵）计算各属性的信息增益，用同样的方法逐层计算来进行分类。

由于按照上面的算法会生成很深的树，所有容易产生过拟合现象。ID3算法还有其他一些缺点，对于具有很多值的属性它是非常敏感的，不能处理具有连续值的属性，不能处理属性具有缺失值的样本。

### 2.2 C4.5算法

因为ID3采用信息增益准则对可取值数目较多的属性有所偏好，食堂根据一系列的属性将学生的餐饮消费水平进行划分，虽然饭卡卡号这一属性具有很高的信息增益，可以很好的划分类别，但是一旦出现新样本它的表现就很不理想。为克服该问题，C4.5对ID3算法进行了改进，使用信息增益率来选择最优划分属性。

除此之外，ID3算法不能解决连续变量问题，对于一些连续型的变量时，ID3只能对每一个不同值的变量进行分开计算，这就会使决策树中产生过多的分支。而C4.5算法也可以解决这一问题，同过合理的寻找某一域值，让小于等于这个域值的数据放在决策树的左边，大于这个域值的数据放在决策树的右边。

信息增益指的是划分前后熵的变化，可以用下面的公式表示：

其中，为属性a的固有值：

### 2.3 CART算法

CART假设决策树是二叉树,等价于递归地二分每个特征，将输入空间即特征空间划分为有限个单元，并在这些单元上确定预测的概率分布，也就是在输入给定的条件下输出的条件概率分布。

CART算法由决策树生成和决策树剪枝两部组成。

#### 2.3.1 决策树生成

基于训练数据集生成决策树，生成的决策树要尽量大。CART决策树的生成就是递归地构建二叉决策树的过程。CART决策树既可以用于分类也可以用于回归。对分类树而言，CART用Gini系数最小化准则来进行特征选择，生成二叉树。

基尼值：

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值越小，数据集D的纯度越高。

属性a的基尼指数：

在侯选属性集合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，即:

在CART算法中，Gini不纯度表示一个随机选中的样本在子集中被分错的可能性。Gini不纯度为这个样本被选中的概率乘以它被分错的概率。当一个节点中所有样本都是一个类时，Gini不纯度为零。

基尼指数Gini(D)表示集合D的不确定性，基尼指数Gini(D , a)表示经过a分割后集合D的不确定性。基尼指数越大，样本的不确定性也就越大。

#### 2.3.2 决策树剪枝

用验证数据集对已生成的树进行剪枝并选择最优子树，这时损失函数最小作为剪枝的标准。剪枝能够解决决策树过拟合问题，去除多余的节点，提高泛化性能。而裁剪枝叶的策略对决策树正确率的影响很大，主要有两种裁剪策略：

（1）预剪枝：在决策树的生成过程中，对每个结点在划分前先进行估计，若当前结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结点标记为叶节点。

（2）后剪枝：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节点进行考察，若将该结点对应的子树替换为叶节点或是将一个子树完全替代另外一棵子树能带来决策树泛化性能提升，则进行替换。

### 2.4 C5.0算法

C5.0是来自[C4.5](https://cse-wiki.unl.edu/wiki/index.php/Decision_Trees,_Overfitting,_and_Occam's_Razor" \l "C4.5)的同一创建者的[决策树](https://cse-wiki.unl.edu/wiki/index.php/Decision_Trees,_Overfitting,_and_Occam's_Razor" \l "C4.5)算法。因此，它自然会在C4.5上扩展和改进。但是，C5.0的源代码是封闭的，对于公众是不可得到的。它是在C4.5算法做了一些改进，C5.0主要增加了对Boosting的支持，它同时也用更少地内存。

从C4.5到C5.0的改进如下：

（1）更快的速度：C5.0比C4.5快。

（2）内存使用效率更高：C5.0的内存使用效率更高。

（3）较小的树：C5.0获得的结果与C4.5非常相似，但构建的决策树却小得多。

（4）相似的精度：C5.0获得与C4.5相似的精度。

（5）Boosting支持：可以使决策树更准确。

（6）权重：使用C5.0，可以权衡不同的属性和错误分类类型，可以构造分类器以最大程度地减少预期的错误分类成本而不是错误率。

### 2.5 总结

对以上经典的决策树算法进行比较：

ID3算法以信息增益为准则选择划分属性，采用贪心算法最大化信息增益的值，自顶向下进行搜索，对单个属性进行多叉划分，为属性的所有取值都建立一个分支。这种划分方式简单直观、易于解释，但是无法处理连续数值型的属ID3倾向于选择水平数量较多的变量，可能导致训练得到一个庞大且深度浅的树；另外输入变量必须是分类变量（连续变量必须离散化）；最后无法处理空值。

C4.5算法使用信息增益率选择属性，消除了信息增益指标导致的多值属性不应有的优势。C4.5算法相对于ID3算法进行了改进，在决策树的构造过程中对树进行剪枝，能够对非离散数据和不完整数据进行处理。

CART 既可以用于分类问题，也可以用于回归问题。回归树中，使用平方误差最小化准则来选择特征并进行划分；分类树中，使用 Gini指数最小化准则来选择特征并进行划分。无论是回归还是分类问题，无论特征是离散的还是连续的，无论属性取值有多个还是两个，内部节点只能根据属性值进行二分。

CART分类算法以基尼系数替代熵，使用Gini不纯度作为度量标准，最小化不纯度而不是最大化信息增益。不断将当前的样本集分为两个子集，使得每个非叶子节点都有两个分支，最后产生一棵二叉决策树。

相比ID3算法，二叉划分的适用范围更广，可以较好地处理数值型的属性，但是使用这种方法划分离散值属性时会造成决策树深度增加，划分数值型属性时则需要大量的排序和计算。CART与C4.5算法是非常相似的，但是CART支持预测连续的值。CART用训练集和交叉验证集不断地评估决策树的性能来修剪决策树，从而使训练误差和测试误差达到一个很好地平衡点。

分类

信息增益

ID3

信息增益率

分类

C4.5

决策树

回归

分类

Gini系数

CART

误差平方和

模型树

回归树

回归

**图2.1 经典决策树对比图**

其中，三个划分属性的对比：

信息增益和信息增益率对比来看，之所以引入了信息增益比，是由于信息增益总是偏向于选择取值较多的属性。信息增益比在此基础上增加了一个罚项，解决了这个问题。

Gini系数和熵对比来看，这两个都可以表示数据的不纯度，但Gini 指数的计算不需要对数运算，更加高效；Gini指数更偏向于连续属性，熵更偏向于离散属性。

## 3 优化算法——随机森林

Bagging算法是一种重要的集成学习算法。Bagging方法最初是为了改善分类器的不稳定性。它的核心思想是使用可放回的抽样，反复采样产生许多数据集，并使用它们分别训练产生一簇决策树，以投票或者其他方式产生最终的预测结果。

当针对的是分类问题时，从直观角度看，可将每棵决策树都是一个分类器，那么对于一个输入样本，N棵树会有N个分类结果。而随机森林集成了所有的分类投票结果，将投票次数最多的类别指定为最终的输出，这就是一种最简单的Bagging思想。

### 3.1随机森林的概述

随机森林是组合分类器算法，是通过集成学习的思想将多棵树集成的一种法。它采用分类回归树CART作为元分类器，通过Bagging方法生成相互之间有差异的不同训练样本集。在单棵树的构建中，随机选择属性对节点样本集进行划分。

它的本质属于机器学习的一大分支——集成学习方法。其中，“随机”表示 一个是随机有放回的选取数据样本，一个是随机选取特征。

该算法尽管单个树的分类能力弱化了，但是可有通过大量的树来进行判断，得到一个较优的分类。随机森林从直观上看类似于投票，当判断食堂饭好不好吃时，森林里的每一棵树都比作学生，它们分别给出判断好吃与不好吃，虽然可能有些个体的判断不是很准，但是最后能够根据投票多的结果来得出食堂饭好不好吃的结论。

### 3.2 随机森林算法

随机森林的具体算法步骤如下：

（1）假设要构建的森林规模大小为k。在训练样本集中通过袋装法(Bagging)生成k个自助样本集。

（2）一个自助样本集用于构建一棵分类树，k个自助样本集则产生k个分类树。设样本的特征数为M，单棵树的生长过程如下：在树的每个内部节点处从M个特征中随机挑选m个特征作为候选特征(m<<M)，按照节点不纯度最小的原则从这m个候选特征中选择一个最优特征对节点进行分裂生长。让每一棵分类树充分生长直到每个叶子节点的不纯度达到最小，不对树进行剪枝。

（3）依据构建好的多个CART树分类器来预测新的未知样本，未知样本的分类结果根据每个树分类器投票结果的简单多数投票法来决定。

其中，对于每棵树都有放回的随机抽取训练样本，测试集即为该树抽取样本之后剩余的样本，称为袋外样本(Out-Of-Bag,OOB)。OOB数据可以用来估计组合分类器的泛化误差，或者用于估计单个特征的重性。由袋外样本做测试集造成的误差称为袋外误差errOOB。OOB误分率是随机森林的泛化误差的一个无偏估计，这是Breiman通过实验证明得到的。

在一棵树中对于m个特征都计算一次，就可以算出m个特征在该树中的重要程度。每个特征在多棵数中出现，取这个特征值在多棵树中的重要程度的均值即为该特征在森林中的重要程度：

其中ntree表示再森林中出现的次数。表示第t棵树中属性值改变后的袋外误差，表示第t棵树中正常值的袋外误差。

得到了所有特征在森林中的重要程度。将所有的特征按照重要程度排序，去除森林中重要程度低的部分特征，得到新的特征集。这时相当于我们回到了原点，这算是真正意义上完成了一次迭代。

## 4 面向大数据的决策树算法

现在各个行业的数据量越来越大，内存容量和处理器速度成了限制的条件，同时还伴有特征数量多的问题，以上三种经典的决策树算法不再适用。无法对大数据集进行处理，因此要对算法进行改进，将已经成功应用于规模相对较小的数据的成熟算法进行适当扩展，或者开发适合于大数据分析的新算法

### 4.1 对大数据的集中式算法——SLIQ

在SLIQ(Supervised Learning InQuest)算法中，最重要思想是在决策树的构造过程中采用了预排序和广度优先两种技术，减少对磁盘的读写次数以提升算法效率。使用Gini指数代替增益比率，选择具有最低Gini指数的值进行分裂，Gini指数比增益比率性能更好，且计算方便。Gini指数最大的特点是计算时只需考虑类值在被划分时每一部分的分布情况通过前面可知，ID3算法不能够充分处理缺失数据和噪音数据，而SLIQ算法能够弥补ID3算法存在的这些不足。

#### 4.1.1预排序

预排序的过程使用了两个数据结构来实现，分别是属性列表和类别列表。对于连续属性来说，在每个内部节点寻找其最优分裂标准的时候，都需要对训练集按照该属性的取值进行排序，而排序是个很浪费时间的操作。为此，SLIQ算法采用了预排序的技术，以便能够消除在决策树的每个节点对数据集进行排序的操作。所谓预排序，就是针对每个属性的取值，遍历所有节点，把所有的记录按照从小到大的顺序进行内部排序。具体实现时，需要为训练集数据的每个属性创建一个属性列表，为每个元组的类别创建一个类别列表。如果某升序节点分裂为两个节点，它的子节点的属性值在属性表仍是升序序列。因此，在数据挖掘的数据准备阶，属性值只需排序一次。完成预排序，数据的初始化就结束。算法实现时需要有足够的内存来保存类别列表。

#### **4.1.2广度优先策略**

在C4.5中，树的构造是按照深度优先策略完成的，需要对每个属性列表在每个节点处都进行一遍扫描，费时很多。SLIQ采用广度优先策略构造决策树，即在决策树的每一层只需对每个属性列表扫描一次，就可以为当前决策树中每个叶子节点找到最优分裂标准，因此在一定程度上具有良好的随记录个数和属性个数增长的可扩展性。

然而它仍然存在如下缺点：

（1）由于需要将类别列表存放于内存，而类别列表的长度与训练集的长度是相同的，这就一定程度上限制了可以处理的数据集的大小。

（2）由于采用了预排序技术，而排序算法的复杂度本身并不是与记录个数成线性关系，因此使得SLIQ算法不可能达到随记录数目增长的线性可扩展性。

#### 4.1.3 Gini指标

在 SLIQ 算法中，采用了Gini指标来方便运算。 如果给定数据集合D，则GINI（D）为：

若D划分为D1和D2，则有：

#### 4.1.4 剪枝

SLIQ采用了MDL后剪枝方法，其目标是生成一棵描述长度最小的决策树。根据MDL原理，最好的编码模型描述数据代价最小。若模型M对数据集D进行编码，那么描述代价Cost为：

其中，Cost（M|D）表示模型M对数据集D编码的编码代价，单位bits；Cost（M）表示描述模型本身所需的编码长度。

### 4.2 对大数据的分布式算法——PLANET

PLANET并不开源。它是Google开发的可扩展的分布式计算框架，用于大数据样本下树模型的训练。PLANET算法的核心是一个控制器，它初始化并控制了整个树的构建过程。

### 4.3 面向流数据的算法——VFDT

VFDT算法是流式决策树算法的先驱，它利用了Hoeffding不等式，在每个叶子节点都保存了相应的统计信息，通过不断地将叶子节点替换为中间决策节点来生成决策树。当新样本到达后，在树的每个节点进行测试，最终到达叶子点，并更新统计信息。算法是一种基于Hoeffding树的对数据流模型进行分类的决策树方法。Hoeffding树在最坏情况下能在恒定时间内对一个实例进行学习，而且所构建的决策树的准确率不差于其他算法。VFDT系统则是在Hoeffding树的基础上发展而来的，它对每一个实例只扫描一次，不会对它们保存，所以适合大量数据流的挖掘，VFDT最主要的贡献在于用Hoeffding不等式来估算样本的数量。

VFDT是面向流式数据的决策树算法，每个样本至多只被处理了一次，相同情况下计算时间远远少于传统的决策树。

### 4.4 总结

数据挖掘的研究工作越来越火热，决策树作为数据挖掘中求解分类问题的有效的方法之一仍有很多可以改进，比如分裂属性如何选择、选择属性的次序、分裂的数目确定、产生树的结构、决策树的停止规则、训练数据集的大小和对树如何进行剪枝等问题。

## 5. 总结与展望

结合课堂中学习到的决策树知识，在整理改文档过程中，我通过看论文和博客了解到常规决策树算之间的区别与联系；随后，结合当下大数据的背景，去了解相应的优化算法。通过阅读这些会发现单就决策树而言，它并不独立，具体步骤中会和k折交叉验证还有Hoeffding不等式等知识有紧密的关联，需要有较好的数学功底。

这次作业增进了我对模式识别的认识，让我对今后模式识别的学习有了更加清晰的看法，不仅要深入理解学到的知识，还应该对其中提到的有关方法进行学习才能拓宽知识面；另外，实验能够事半功倍。感谢赵老师的教学与帮助。

### 参考资料

[1]张华成.基于SLIQ决策树算法的研究[J].现代计算机(专版),2009(10):54-56+83.

[2]张薇.一种基于改进SLIQ决策树分类算法的应用研究[J].苏州大学学报(工科版),2010,30(01):72-77.

[3]张棪,曹健.面向大数据分析的决策树算法[J].计算机学,2016,43(S1):374-379+383.

[4]李贞贵. 随机森林改进的若干研究[D].厦门大学,2013.

[5]刘寿强,祁明.分布式数据流决策树VFDT分类算法研究[J].现代计算机(专业版),2016(36):3-7+14.

[6]李航.统计学习方法[M],清华大学出版社,2012.3,ISBN 978-7-302-27595-4

[7]<https://www.cnblogs.com/bourneli/archive/2013/03/15/2961568.html>

[8]<https://www.jianshu.com/p/36dd7cc9286a>

[9]https://blog.csdn.net/icefire\_tyh/article/details/52081879

[10]<https://www.jianshu.com/p/60f875cf0e8e>

[11]<https://blog.csdn.net/qq_34106574/article/details/82016442>

[12]<https://blog.csdn.net/August1226/article/details/80729908#%E3%80%90%E5%8F%82%E8%80%83%E9%93%BE%E6%8E%A5%E3%80%91>

[13]<https://www.cnblogs.com/bourneli/archive/2013/03/11/2954060.html>

[14]<https://scikit-learn.org/stable/modules/tree.html#tree-algorithms-id3-c4-5-c5-0-and-cart>

[15]<https://blog.csdn.net/qq_34420188/article/details/71079763>

[16]https://blog.csdn.net/cg896406166/article/details/83796557

[17]<https://www.cnblogs.com/libin123/p/9985918.html>