第43卷 第6A期
2016年6月

计 算 机 科 学
Computer Science

Vol. 43 No. 6A
June 2016

面向大数据分析的决策树算法

张 棪 曹 健
(上海交通大学计算机科学与工程系 上海 200240)

摘 要 决策树作为机器学习中的一个预测模型,因其输出结果易于理解和解释,而被广泛应用于各个领域,成为了学术界研究的热点。随着数据产生速度的剧增,由于内存容量和处理器速度等限制,常规的决策树算法无法对大数据集进行处理,因此需要对决策树算法的实现进行针对性的处理。首先阐述了决策树的基本算法和优化方法,在此基础上结合大数据带来的挑战,分类比较了各类针对性算法的优缺点,并介绍了支撑这些算法运行的平台。最后讨论了面向大数据的决策树算法的未来发展方向。

关键词 决策树,大数据,机器学习
中图法分类号 TP18 **文献标识码** A

Decision Tree Algorithms for Big Data Analysis

ZHANG Yan CAO Jian
(Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China)

Abstract As a predictive model in machine learning, decision tree algorithm is widely used in various fields and has been a hot research topic due to its easily understandable result. Since the speed of data generation increases explosively, conventional decision tree algorithms cannot deal well with large datasets due to the limitation of memory capacity and processor speed, and thus novel implementation is needed urgently. Based on classic implementation and optimization method of decision tree, and challenges brought by big data, we compared various kinds of corresponding algorithms. Platforms for big data algorithms were then introduced and possible directions in the future were discussed in the end.

Keywords Decision tree, Big data, Machine learning

1 引言

1.1 决策树算法的基本概念及其意义

决策树是机器学习中的一个树状预测模型,通常其内部节点表示在一个属性上的测试,而叶子节点代表最终的类别。决策树模型被用来解决许多基本问题,诸如多阶段决策、表查找[34]、最优化[35,36]等,它很自然地还原了做决策的过程,将复杂的决策过程拆分成了一系列简单的选择,因而能直观地解释决策的整个过程。

在机器学习和数据挖掘领域,不乏一些如专家决策系统等需要展示或阐明处理过程的场景,大多数复杂的机器学习算法由于计算过程晦涩难懂,无法满足这类需求,而决策树模型正是最合适的解决方案。有调查表明决策树算法是经常被使用的机器学习算法之一[2],已经成功应用在医疗、广告、交通、金融等领域[62-65]。学术界对决策树技术的研究也一直不断发展,出现了各种衍生算法及相关的优化[1,8,13,19,20,26,28]。

1.2 大数据对决策树的挑战

随着信息技术的发展大量的信息被数字化成数据并由计算机进行处理分析。在互联网、金融、医学等一些领域,许多数据集每天可以产生上亿条记录,此外,手机、智能可穿戴设备等电子产品配备了各种传感器,它们也产生大量的数据。而网络和存储技术的发展使得数据的传播和存储变得更加便捷。

越来越多行业需要对大量信息进行分析和处理,在大数据中找到有用的信息,挖掘其中蕴藏的规律并加以利用。因此,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,成为了目前的研究焦点。

对于决策树算法而言,大数据集的特征数量多,其中不乏一些冗余、低质量甚至无关的特征,而对它们的处理不仅消耗了大量计算资源,导致树的结构臃肿,还使预测的准确性存在下降的风险。因此,对数据集的特征进行筛选,找出更符合要求的那部分特征进行机器学习是提高效率和准确率的一个重要手段,其具体实现方法则是一个很大的挑战。

大数据的高样本容量直接导致了决策树无法在内存中完全构建,还会消耗大量的运算时间。为了高效地处理这些数据,较为合理的办法是利用集群的分布式存储和带宽资源,使用 MapReduce[49] 或 MPI[48] 等任务并行的方法来降低运算负载。如何利用机器学习的其他技术把决策树构建算法扩展以进行并行计算,是大数据时代决策树算法以及其他机器学习

本文受国家自然科学基金(61272438,61472253),上海市科委项目(14511107702,15411952502)资助。
张 棪(1992—),男,硕士生,主要研究方向为大数据分析,E-mail:zy1992@sjtu.edu.cn;曹 健(1972—),男,博士后,教授,主要研究方向为服务计算、网络计算、大数据分析。

方法的可行方法。

而在一些特定的场景下，在短时间内高速产生大量的数据，如社交网络的媒体数据、电商网站上的交易记录、人们日常的通话记录等。这些数据无法完全被放到内存或者磁盘中，只能被观察到有限次（通常只有一次），一般算法的处理能力远低于数据产生的速率。对于这些流数据，如何更高效地实时处理并利用它们，也是一个亟待解决的问题。

本文介绍了决策树构造的基本方法，结合相应的问题进行了分析和比较，重点对面向大数据的优化进行了研究，介绍了相关的算法和计算平台。第 2 节介绍了决策树的基本算法及其优化方法；第 3 节具体介绍和对比了一些面向大数据的算法和优化方法；第 4 节简要介绍了支撑大数据决策树算法的平台和框架；最后简要分析了现今大数据环境下决策树算法存在的一些潜在问题和未来发展的研究方向。

2 决策树的基本方法

构造决策树的过程与人做决策的行为模式相似。给定一个数据集 S，其包含多种属性的值以及所属的分类，首先要做的是使用一些统计方法选择一个属性 A 作为根节点，根据属性 A 的值将数据集 S 划分为多个子集。在这些子集上重复上述过程，直到满足一个特定的终止条件，如子集中超过一定比例的数据都属于同一个类别，最终叶子节点就代表它对应的集合中大部分数据的类别。

2.1 经典算法

ID3 算法[1]使用信息熵[4]选择属性，采用贪心算法最大化信息增益(Info Gain)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性。

C4.5 算法[5]使用信息增益率(Gain Ratio)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。

CART 算法[3]使用 GINI 不纯度作为度量标准，采用二分递归分割的方法，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。相比于 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。

此外还有一些如 OC1 算法[14]等关于多变量决策树或斜决策树的研究[13,14]。

2.2 基本优化方法和性能比较

在决策树学习中，树的结构（大小和深度等）对决策准确率有很大影响，过拟合不仅会影响预测的准确性，还会导致决策规则复杂难懂。决策树的结构主要由不纯度的度量指标和后剪枝的方法决定。早期的研究[15,16]对这两方面采取的常见方法做了介绍和评估，利用一些数据集上的实验结果表明了它们对决策树大小和预测准确度的影响。

2.2.1 度量指标的选取

度量指标主要分为下面 3 类：

(1)信息增益(Info Gain)又叫做相对熵或者 KL 散度，在决策树学习理论中一般特指信息熵的差，用来表示不纯度的绝对值之差。

(2)信息增益率(Gain Ratio)是信息增益的一个标准化处理，它考虑到了多值属性取值个数（即子分支数）的影响，相比于信息增益而言，可以消除多值属性不应有的优势。

(3)代价函数(Cost Function)不局限于信息熵，而是根据实际的经验，关注了错误判断类别所带来的代价，在文献[6,7]等的新算法和其他过程文献[9-11]中被广泛使用。

关于度量指标选取方面的方法，Mingers[16]和 Buntine[41]分别在多个数据集上做过实验比较。

2.2.2 后剪枝方法

剪枝是指用直接叶子节点替换掉一些子树，叶子的类用替换掉的子树中包含的大多数训练样本的类表示。它可以降低树结构的复杂性，改善过拟合的情况以提高预测准确性，快速预测结果以及简化决策过程。后剪枝最初由 Breiman 提出[3]，相对于存在视野效果问题的前剪枝，它指一棵决策树初次构造完毕之后进行的剪枝。

最早提出的基于代价复杂度的剪枝方法[3]首先自底向上依次用叶子节点替换掉一些子树，生成一簇决策树，然后对这些剪枝过的决策树进行评估比较，选出一个最好的作为最终的结果。最小化错误数的剪枝方法[19]则更为简单，直接测试原始决策树的每一棵子树，一旦替换成叶子节点便能够减少错误数就执行。使用上述两种方法剪枝，除了生成决策树的过程中使用的训练数据集之外，还需要提供一个额外的测试数据集。而悲观错误剪枝方法[20]使用原始训练样本集合，根据概率论和数理统计的知识，对每一棵子树的错误率作估计从而判断是否剪枝，因而不需要额外的数据集。

常见的后剪枝方法还有临界值剪枝方法[15]、最小错误剪枝法[22]等。文献[21]对这些方法作了具体的介绍和比较，它们的性能比较也可以参考 Mingers 等人[15]的研究。

2.3 基于集成学习的多决策树构造

单一决策树分类器处理问题的能力有限，也较容易出现过拟合的情况，利用多算法或多分类器一起产生一个更好结果的方法就是集成学习[44]的基本思想。Dietterich 在文献[43]中分类介绍了包含随机化、Bagging、Boosting 等集成学习的方法。

Bagging 方法最初是为了改善分类器的不稳定性[23]。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。基于这个原理，Domingos 提出了 MetaCost 算法[24]，预测的结果通过求解最小化特定的代价函数得到。Lin 和 Mclean 使用的算法[25]是一个包含决策树的混合分类器，训练的数据集是一样的，不同的是各个分类器使用了不同的模型，其中包含了 C5.0 决策树和神经网络等。著名的随机森林算法[26]本质上也是 Bagging 算法。

Boosting 是一种迭代使用弱学习分类器组合成一个强学习分类器的思想。AdaBoost[27]是 Boosting 方法中经典的例子，它通过控制训练样本的权重生成强学习分类器。每一轮对训练样本进行预测，被正确分类的样本降低权重，错误分类的样本提高权重，在下一轮中提高了那些之前被错误分类的样本对即将新加入的弱分类器的影响，从而使总体的分类器能更好地适配训练集的特征。提升决策树最早由 Yoav 提出[29]，衍生出的算法还有 Bonsai 提升决策树[30]等，其中梯度提升决策树算法(GBDT)利用了梯度提升法的思想，使用决策回归树作为分类器已经被广泛使用[31,32]，不仅在 KDD CUP 等竞赛中出现，在工业界也有应用[33]。

Dietterich 在文献[42]中以 C4.5 算法为基础，以图的形式直观比较了包含 Boosting 和 Bagging 的集成学习方法的性能。

3 面向大数据的决策树算法

大数据环境下样本数量巨大，同时往往会伴有特征数量多的问题，进行特征选择能规避无关或冗余的特征带来的影响，从而减少运行时间并提高预测准确度，是应用各种机器学习算法之前一个很重要的环节。在决策树的构建中，选取出真正相关的特征还简化了模型，保留了决策树模型易于解释的优点。

优化扩展各种决策树的学习算法以应对大数据的挑战是一个很重要的研究方向，许多研究者活跃在这个方向并且提出了各种优化方法，这些方法的思路大致分为两个方向：

1）为了规避内存大小限制而在磁盘上进行的集中式算法。

2）使用特别的平台或架构进行计算的分布式并行算法。

前者的瓶颈在于磁盘读写的速度相比内存而言慢得太多，而后者的瓶颈在于分布式节点的通信负载以及如何将算法的核心任务分割成可以并行执行的子任务。

此外，面临越来越多的流数据，研究人员也一直在满足面向流数据机器学习算法的基本需求的前提下进行各种优化。

3.1 特征选择

特征选择是指在原特征集合中找出一个子集，使得后续的机器学习算法在这个子集上能更好地进行。对特征按照某一个评价指标进行排名是基本的方法，它的过程简单而且便于扩展，在实际应用中也表现得不错，因而许多特征选择算法使用排名的方式作为主要或者辅助的选择机制。而排名所用的评价指标是最关键的因素，根据这个指标的不同计算方式，特征选择算法大致分为两类：

1）筛选器(Filter)，直接通过特征集合内部的统计信息等来衡量，是一个预处理的过程。

2）封装器(Wrapper)，实际上是一个分类器，用选取的特征子集对样本用后续要使用的机器学习进行分类，用准确率来衡量。

对于筛选器而言，它是独立于后续分类算法的一个过程，常用的评价指标有相关系数、样本距离、信息增益等。

相关系数用来衡量向量之间的线性相关度。用它选择特征时，为了使特征具有代表性，特征子集所包含的特征与类别的相关度要高；为了减少特征之间的冗余，特征之间相互相关度要低。但是使用相关系数的一个局限是它默认了各个样本之间的线性关系。

用距离进行特征选择时，选取的方法和聚类相似，同一类别的样本之间距离要尽可能小，而不同类别的样本之间的距离要尽可能大，其中常用的有欧几里得距离和马氏距离。

对于封装器而言，评价指标其实就是选取的特征子集的模型在测试样本上的准确率，它在评价的过程中用了后续要使用的分类算法来进行分类，使选出来的特征能更适应这个分类算法，可以看作是具体算法的一个嵌入式特征选取过程。相比于筛选器而言，它的计算量更大，而且通用性较差。

3.2 集中式算法

数据集过大的一个直接影响就是无法完全放进内存进行计算，以至于数据结构存放在磁盘中。决策树构建过程中大量的读写操作使得磁盘的读写速度成为一个很大的瓶颈，针对这种情况，许多研究对决策树构造算法的基本过程进行了优化，以减少对数据的读写操作。

Mehta 等人[50]提出了 SLIQ 算法，使用预排序和广度优先划分节点等高明的技巧进行优化，减少对磁盘的读写次数以提升算法效率。

传统的算法中，在决策树的每个节点寻找最优划分策略时都需要对样本根据每个属性进行排序，其中包含了大量重复的排序操作，可以使用预排序进行优化。预排序的过程如图 1 所示，使用了两个数据结构来实现，分别是属性列表(Attribute List)和类别列表(Class List)。

属性列表〈值，类别列表索引〉是对每个属性分别建立的一张按值排序的表，记录样本在这个属性上的值以及它在类表中的索引，其由于总体积较大可以写回磁盘。

类别列表〈类别，叶子节点〉只有一张，必须常驻内存，记录了每个样本所属的类别以及所对应的叶子节点，使得任何时候都能快速得到每个样本所对应的划分方式。

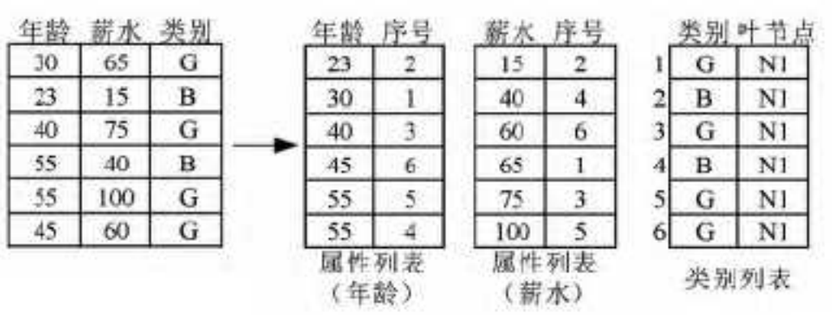


图 1 SLIQ 算法预排序过程[50]

注：左图为训练样本，右图分别是预排序后两个属性的属性列表和类别列表。

与传统算法不同，SLIQ 还使用了广度优先的策略来进行决策树的生长，避免了深度优先策略引起的每个节点都要遍历属性列表的问题，从而只需在决策树的每一层对每个属性列表遍历一次就可以找到最佳的划分方式。

SLIQ 中类别列表的大小和训练样本的容量成正比，随着样本容量增大，类别列表的大小会增大，导致无法驻留在内存，因此 Shafer 等人[53]在 SLIQ 的基础上提出了 SPRINT 算法，去掉了类别列表，将类别信息直接记录在属性列表中，每个条目变成了〈值，类别，样本 ID〉。这样虽然增大了属性列表的体积，但是在遍历它以寻找最优划分方式的过程中不需要再去根据索引查找类别列表的内容了，决策树节点的划分直接表现在了属性列表的分割，如图 2 所示，每个属性列表也被分割成了两个，分别保存两个分支的样本所对应的信息。

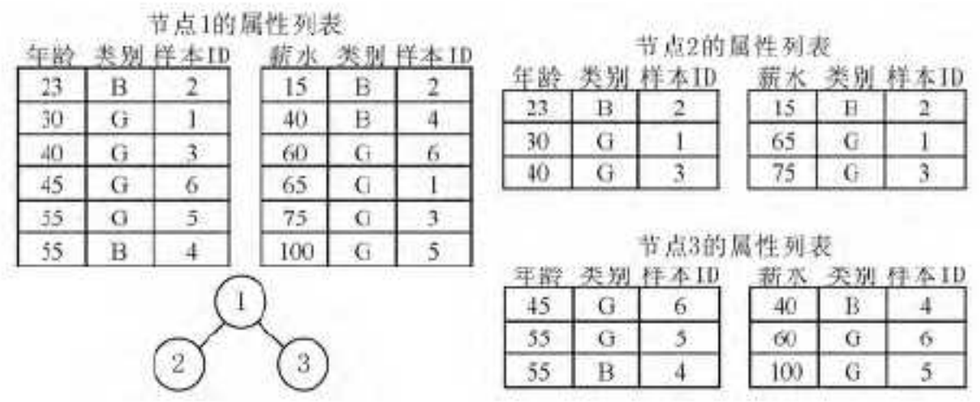


图 2 SPRINT 算法节点划分过程[53]

文献[52]提出的 CLOUDS 算法对数值型属性的分割点进行采样，通过估计的方法找到一个相对最好的划分方式。相比于 SPRINT 或 C4.5 等算法遍历所有样本中相应属性的值而言，CLOUDS 算法简化了计算步骤，且实验结果表明它又不失准确度。

为了能使最复杂的运算过程能完全在内存中进行以减少

磁盘读写次数，Johannes Gehrke 等人提出了 RAINFOREST 算法[54]和 BOAT 算法[55]。其中 RAINFOREST 算法在选择最佳划分方式时，把属性、值和类等所需要的信息压缩到一个叫做 AVC group 的数据结构里，通常情况下这个压缩后的数据结构不会很大，可以被放进内存里计算。而 BOAT 算法则对原始数据集进行了采样，生成一个可以放进内存的子集进行训练，最后再与原始数据集对比修正误差。

3.3 分布式算法

早期 Bradford 等人[51]基于 ccNUMA 架构，对 C4.5 算法进行扩展，并行地扫描整个数据集加快处理器从存储器读取数据的过程，以提高整体的运行速度，相比使用单一处理器性能有所提高。

PLANET[45]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。控制器往往是一台单独的计算机，可以接入一个计算集群，从而使用 MapReduce 来进行分布式计算。为了控制和协调树的构建，控制器需要维护模型文件(M)，一系列的 MapReduce 任务参与构建决策树的不同部分，在任意一个计算节点，这个模型文件表示决策树当前的状态，记录了到当前为止已经构造完毕的部分。

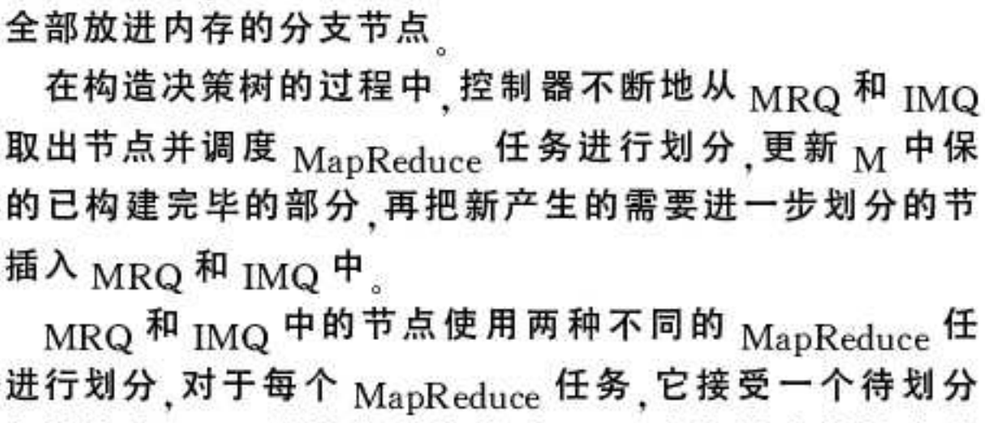
有了模型文件，控制器就可以判断当前应该在哪个节点进行划分操作了，划分后的分支节点被存放在两个不同的队列中：

MapReduce 队列(MRQ)，这个队列存放包含的样本总量超过内存大小的分支节点。

In-Memory 队列(IMQ)，这个队列存放包含的总样本可以全部放进内存的分支节点。

在构造决策树的过程中，控制器不断地从 MRQ 和 IMQ 中取出节点并调度 MapReduce 任务进行划分，更新 M 中保存的已构建完毕的部分，再把新产生的需要进一步划分的节点插入 MRQ 和 IMQ 中。

MRQ 和 IMQ 中的节点使用两种不同的 MapReduce 任务进行划分，对于每个 MapReduce 任务，它接受一个待划分节点的集合(N)、训练样本的集合(D)、决策树当前的状态(M)这 3 个参数作为输入。MRQ 中的节点使用 MR_ExpandNodes 任务来划分，产生分支节点；而 IMQ 中的节点使用 MR_InMemory 进行划分，直接构建完这一部分的子树。图 3 示出了该算法构建一棵树的基本过程，样例设置内存中至多可以放入 25 个样本。



左图为构建的决策树，其中方框中标记的是节点的划分条件，边上标注了各个分支中的样本数量；右图显示了算法执行的过程，以及 MRQ 和 IMQ 两个队列的变化情况

图 3 PLANET 算法示意图[45]

PLANET 并不开源，它的主要技术细节可以参考文献[45]，文献中有对 PLANET 使用的 MR_ExpandNodes 任务的具体介绍，Yin 等人[46]基于此实现了一个开源的 OpenPlanet 算法。

此外，当下流行的结合集成学习方法的决策树模型，也可以使用分布式的解决方案来应对大数据集的问题。把 GBDT 转化成 MapReduce 模型十分直观，Ye 等人[47]基于 GBDT 算法分别使用了 MapReduce 和 OpenMPI 两种方式，提出了两种不同的分布式决策树算法。

3.4 面向流数据的算法

VFDT 算法[56]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，最终到达叶子节点，并更新统计信息。构造过程如表 1 所列，其中 n_{leaf} 表示在 $leaf$ 节点上的样本数，$n_{\min}$ 表示一次生长操作需要的最小样本数。

表 1 HTInduction 算法

输入：I 是一个训练样本；HT 是当前状态的决策树
输出：下一个状态的决策树
1. 找到样本 I 在 HT 中所属于的叶子节点 leaf，并更新 leaf 中的统计信息
2. $n_{leaf} \leftarrow n_{leaf} + 1$
3. if $n_{leaf} \bmod n_{min} = 0$ and leaf 上有不同类别的样本 then
4. 对每个属性 X_i 计算度量指标 $G_{leaf}(X_i)$，包括不划分的情况 $X_\varnothing$
5. 找到 G_{leaf} 最大的两个属性 X_a 和 X_b
6. 计算 Hoeffding 上界
7. if $X_a \neq X_\varnothing$ and $G_{leaf}(X_a) - G_{leaf}(X_b) > \epsilon$ then
8. 在 leaf 上用 X_{-a} 属性进行划分
9. 在新划分的每个叶子节点上更新相应的统计信息
10. endif
11. endif

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。

但是 VFDT 中既没有处理连续数值型属性的手段，也无法解决实际应用中概念漂移所带来的准确度下降的问题。因此，Hulten 等人[57]基于 VFDT 提出了 CVFDT 算法，用一个 slide-window 来监测数据流，在叶子节点可能会产生概念漂移时，使用新的备选子树来代替原先的旧子树，以应对预测性能下降的问题。Gama 等人[58]则提出了 VFDTc 算法来支持数值型属性的处理，同时也提高了预测的准确性。

4 支撑大数据决策树算法的平台

Apache 软件基金会(ASF)是专门为支持开源软件项目而办的一个非营利性组织，有许多大数据机器学习平台都诞生于不同的公司或者社区，最后成为 Apache 旗下的重要项目，如最初由 AMPLab 开发的 Spark，以及由 BackType 开发经 Twitter 开源的 Storm 等。

Apache Mahout 是 Apache Hadoop 上传统的机器学习系统。Hadoop 是一个比较早的分布式系统基础架构，用户可以利用其 API 方便地在上层开发分布式程序，利用大量集群的计算和存储能力来解决问题，其中 MapReduce 并行计算框架是 Hadoop 的一个核心模块。早期的版本没有实现其他高级语言的接口，但是随着 Python、R 等脚本语言在科学计算领域的兴起以及机器学习的发展，出现了诸如 RHadoop、Oryx、OxData 等简单易用的工具和平台。

Apache Spark 是一个开源的集群计算系统，是当今大数据领域最活跃的开源项目之一。其中 MLlib 是 Spark 的核心

机器学习库，实现了常用的机器学习算法，还包括了数据的生成和测试功能。此外，Spark 还提供了 Scala、Java、Python 和 R 等多种高级编程语言的接口以及详细的文档，用户可以很快学会如何使用。早期版本的 Spark 不支持树结构，但是在开发者们的努力下各种新特性包括对决策树的支持被加入到 Spark 中。近年来，树学习算法在 Spark 上已经成熟，Manish Made 等人在 2014 年 Spark 峰会上详细介绍了使用 MLlib 进行决策树构建的过程。

Apache Storm 是一个免费、开源的分布式实时计算系统。Storm 简化了面向庞大规模数据流的处理机制，从而在实时处理领域扮演着重要角色。除了用于实时分析外，Storm 也可用于在线机器学习。随着越来越多流式数据的产生，诸如 VFDT[56] 等在线生成决策树的算法也不断发展，对于这些实时性要求高的决策树生成而言，Storm 是一个很好的选择。

另一个流数据机器学习框架是 Massive Online Analysis (MOA)[59]，它包含了一些广为人知的在线算法，诸如一些流数据分类器、聚类算法等。它提供了图形化、命令行以及 Java 的 API 供用户使用，用户可以简单地进行自定义扩展，实现自己的机器学习算法。此外，除了 S4[60]、Debellor[61] 等免费的平台以外，还有一些商用的流数据处理引擎，如 IBM 的 InfoSphere Streams、微软的 StreamInsight 以及 TIBCO 的 StreamBase 等。

结束语　本文介绍了决策树算法的基本方法和优化，对现有的面向大数据的决策树算法进行了研究和分析，分类比较了各自的特点。我们认为大数据环境下，决策树算法有以下潜在的挑战和发展方向。

(1)属性质量问题。在实际应用中，数据质量参差不齐，属性缺失的问题普遍存在，对决策树算法的准确性有很大的影响。文献[37]把获取属性所需要的代价作为特征选择的指标，用以选取最佳的特征子集进行学习，而文献[38]则根据属性缺失程度计算了置信度作为辅助信息，用来描述最后的预测结果的可信程度。如何对待数据中缺失的属性一直是机器学习领域中的一个热门研究课题，在决策树构造过程中也是一个值得深入研究的方向。

(2)正负样本比例。实际应用中经常会遇到训练样本中各个类别的样本数量相差悬殊的情况，这时候大多数决策树算法可能会导致小样本数量的类别的特征在训练过程中被分类器忽略。在这种情况下，如何使决策树模型能在大量其他样本中学习到稀少类型的特征是一个值得研究的问题。

(3)决策树增量更新。虽然决策树的构造一般离线完成，但是在巨大的数据量面前，如何增量地更新决策树也是一个很好的研究方向。现在除了面向流数据的主要算法之外，还有其他一些方法对增量更新的问题作了相应的改进[39,40]。

(4)变化的决策模型。一些数据规律会随时间变化，对于相同的模型，新的数据也许无法很好地符合旧的参数。此外，比起模型的参数随时间变化，更为复杂的问题是模型本身也随着时间变化。现实生活中的数据往往不是单一模型就能完美描述的，其规律也会随着时间和环境的变化而变化，那么如何根据当前的数据变化趋势为未来的数据选择合理的模型也将会是非常有意义的研究。

参 考 文 献

[1] Quinlan J R. Induction of decision trees[J]. Machine Learning, 1986,1(1):81-106

[2] Seni G, Elder J F. Ensemble methods in data mining: improving accuracy through combining predictions[J]. Synthesis Lectures on Data Mining and Knowledge Discovery, 2010, 2(1): 1-126

[3] Breiman L, Friedman J, Stone C J, et al. Classification and regression trees[M]. CRC Press, 1984

[4] Shannon C E. A note on the concept of entropy[J]. Bell System Tech. J, 1948, 27: 379-423

[5] Quinlan J R. C4. 5: Programming for machine learning[M]. Morgan Kauffmann, 1993

[6] Ferri C, Flach P, Hernández-Orallo J. Learning decision trees using the area under the ROC curve[C]// ICML. 2002: 139-146

[7] Ling C X, Yang Q, Wang J, et al. Decision trees with minimal costs[C]// Proceedings of the Twenty-first International Conference on Machine Learning. ACM, 2004: 4-8

[8] Lomax S, Vadera S. A survey of cost-sensitive decision tree induction algorithms[J]. ACM Computing Surveys (CSUR), 2013, 45(2): 227-268

[9] Moret S L, Langford W T, Margineantu D D. Learning to predict channel stability using biogeomorphic features[J]. Ecological Modelling, 2006, 191(1): 47-57

[10] Kretowski M, Grzes M. Evolutionary induction of mixed decision trees[J]. International Journal of Data Warehousing and Mining, 2007, 3(4): 68-82

[11] Fan W, Stolfo S J, Zhang J, et al. AdaCost: misclassification cost-sensitive boosting[C]// ICML. 1999: 97-105

[12] Murthy S K, Kasif S, Salzberg S. A system for induction of oblique decision trees[J]. Journal of Artificial Intelligence Research, 1994, 2(1): 1-32

[13] Barros R C, Cerri R, Jaskowiak P, et al. A bottom-up oblique decision tree induction algorithm[C]// 2011 11th International Conference on Intelligent Systems Design and Applications (ISDA). IEEE, 2011: 450-456

[14] Wickramarachchi D C, Robertson B L, Reale M, et al. HHCART: An Oblique Decision Tree[J]. arXiv preprint arXiv: 1504.03415, 2015

[15] Mingers J. An empirical comparison of pruning methods for decision tree induction[J]. Machine learning, 1989, 4(2): 227-243

[16] Mingers J. An empirical comparison of selection measures for decision-tree induction[J]. Machine Learning, 1989, 3(4): 319-342

[17] Elouedi Z, Mellouli K, Smets P. A pre-pruning method in belief decision trees[C]// The Ninth International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems IPMU. 2002, 1: 579-586

[18] Trabelsi S, Elouedi Z, Mellouli K. Pruning belief decision tree methods in averaging and conjunctive approaches[J]. International Journal of Approximate Reasoning, 2007, 46(3): 568-595

[19] Quinlan J R. Simplifying decision trees[J]. International Journal of Man-machine sSudies, 1987, 27(3): 221-234

[20] Quinlan J R, Cameron-Jones R M. FOIL: A midterm report[C]// Machine Learning: ECML-93. Springer Berlin Heidelberg, 1993: 1-20

[21] Breslow L A, Aha D W. Simplifying decision trees: A survey[J]. The Knowledge Engineering Review, 1997, 12(1): 1-40

[22] Cestnik B, Bratko I. On estimating probabilities in tree pruning

[C]// Machine Learning-EWSL-91. Springer Berlin Heidelberg, 1991:138-150

[23] Breiman L. Bagging predictors[J]. Machine Learning, 1996, 24(2):123-140

[24] Domingos P. Metacost: A general method for making classifiers cost-sensitive[C]// Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 1999:155-164

[25] Lin F Y, McClean S. The Prediction of financial distress using a cost sensitive approach and prior probabilities[C]// Proceedings of the 17th International Conference on Machine Learning (ICML'00). 2000

[26] Breiman L. Random forests[J]. Machine Learning, 2001, 45(1): 5-32

[27] Freund Y, Schapire R E. A decision-theoretic generalization of on-line learning and an application to boosting[J]. Journal of Computer and System Sciences, 1997, 55(1):119-139

[28] Friedman J H. Greedy function approximation: a gradient boosting machine[J]. Annals of Statistics, 2001, 29(5):1189-1232

[29] Freund Y, Mason L. The alternating decision tree learning algorithm[C]// ICML. 1999, 99:124-133

[30] Gligorov V V, Williams M. Efficient, reliable and fast high-level triggering using a bonsai boosted decision tree[J]. Journal of Instrumentation, 2013, 8(2):6

[31] Ayaru L, Ypsilantis P P, Nanapragasam A, et al. Prediction of Outcome in Acute Lower Gastrointestinal Bleeding Using Gradient Boosting[J]. PloS one, 2015, 10(7):e0132485

[32] Lombardo L, Cama M, Conoscenti C, et al. Binary logistic regression versus stochastic gradient boosted decision trees in assessing landslide susceptibility for multiple-occurring landslide events: application to the 2009 storm event in Messina (Sicily, southern Italy)[J]. Natural Hazards, 2015, 79(3):1-28

[33] He X, Pan J, Jin O, et al. Practical lessons from predicting clicks on ads at facebook[C]// Proceedings of 20th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. ACM, 2014:1-9

[34] Haralick R M. The table look-up rule[J]. Communications in Statistics-Theory and Methods, 1976, 5(12):1163-1191

[35] Hartmann C R P, Varshney P K, Mehrotra K G, et al. Application of information theory to the construction of efficient decision trees[J]. IEEE Transactions on Information Theory, 1982, 28(4):565-577

[36] Knuth D E. Optimum binary search trees[J]. Acta Informatica, 1971, 1(1):14-25

[37] Zhang S, Qin Z, Ling C X, et al. Missing is useful: missing values in cost-sensitive decision trees [J]. IEEE Transactions on Knowledge and Data Engineering, 2005, 17(12):1689-1693

[38] Gu Q, Zhang Y, Cao J, et al. A confidence-based entity resolution approach with incomplete information[C]// 2014 International Conference on Data Science and Advanced Analytics (DSAA). IEEE, 2014:97-103

[39] Utgoff P E, Berkman N C, Clouse J A. Decision tree induction based on efficient tree restructuring [J]. Machine Learning, 1997, 29(1):5-44

[40] Gama J, Fernandes R, Rocha R. Decision trees for mining data streams[J]. Intelligent Data Analysis, 2006, 10(1):23-45

[41] Buntine W, Niblett T. A further comparison of splitting rules for decision-tree induction[J]. Machine Learning, 1992, 8(1):75-85

[42] Dietterich T G. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization[J]. Machine Learning, 2000, 40(2):139-157

[43] Dietterich T G. Ensemble methods in machine learning[M]// Multiple Classifier Systems. Springer Berlin Heidelberg, 2000:1-15

[44] Opitz D, Maclin R. Popular ensemble methods: An empirical study[J]. Journal of Artificial Intelligence Research, 1999:169-198

[45] Panda B, Herbach J S, Basu S, et al. Planet: massively parallel learning of tree ensembles with mapreduce[J]. Proceedings of the VLDB Endowment, 2009, 2(2):1426-1437

[46] Yin W, Simmhan Y, Prasanna V K. Scalable regression tree learning on Hadoop using OpenPlanet[C]// Proceedings of third international workshop on MapReduce and its Applications Date. ACM, 2012:57-64

[47] Ye J, Chow J H, Chen J, et al. Stochastic gradient boosted distributed decision trees[C]// Proceedings of the 18th ACM Conference on Information and Knowledge Management. ACM, 2009:2061-2064

[48] Gropp W, Lusk E, Doss N, et al. A high-performance, portable implementation of the MPI message passing interface standard [J]. Parallel Computing, 1996, 22(6):789-828

[49] Dean J, Ghemawat S. MapReduce: simplified data processing on large clusters[J]. Communications of the ACM, 2008, 51(1): 107-113

[50] Mehta M, Agrawal R, Rissanen J. SLIQ: A fast scalable classifier for data mining[M]// Advances in Database Technology (EDBT'96). Springer Berlin Heidelberg, 1996:18-32

[51] Bradford J P, Fortes J A B. Characterization and parallelization of decision-tree induction[J]. Journal of Parallel and Distributed Computing, 2001, 61(3):322-349

[52] Ranka S, Singh V. CLOUDS: A decision tree classifier for large datasets[J]. Knowledge Discovery and Data Mining, 1998:2-8

[53] Shafer J, Agrawal R, Mehta M. SPRINT: A scalable parallel classifier for data mining[C]// Proc. 1996 Int. Conf. Very Large Data Bases. 1996:544-555

[54] Gehrke J, Ramakrishnan R, Ganti V. RainForest-A framework for fast decision tree construction of large datasets[C]// VLDB. 1998, 98:416-427

[55] Gehrke J, Ganti V, Ramakrishnan R, et al. BOAT—optimistic decision tree construction[C]// ACM SIGMOD Record. ACM, 1999, 28(2):169-180

[56] Domingos P, Hulten G. Mining high-speed data streams[C]// Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2000: 71-80

[57] Hulten G, Spencer L, Domingos P. Mining time-changing data streams[C]// Proceedings of the seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2001:97-106

(下转第 383 页)

[5] Arora S,Chana I. A survey of clustering techniques for Big Data analysis[C] // 5th International Conference Confluence The Next Generation Information Technology Summit (Confluence). IEEE,2014:59-65

[6] Nagpal P B,Mann P A. Survey of Density Based Clustering Algorithms[J]. International Journal of Computer Science and its Applications,2011,1(1):313-317

[7] Xu R,Wunsch D. Survey of clustering algorithms,Neural Networks[J]. IEEE Transactions,2005,16(3):645-678

[8] Yadav C,Wang S,Kumar M. Algorithm and approaches to handle large Data-A Survey[J]. Eprint Arxiv,2013,361-363(3):1117-1181

[9] Shirkhorshidi A S, Aghabozorgi S, Wah T Y, et al. Big Data Clustering:A Review[M] // Computational Science and Its Applications(ICCSA 2014). Springer International Publishing, 2014:707-720

[10] Wu X,Zhu X,Wu G Q,et al. Data mining with Big Data[J]. IEEE Transactions on Knowledge and Data Engineering,2014,26(1):97-107

[11] Aggarwal C C,Reddy C K. Data Classification:Algorithms and Applications[C] //CRC Press. 2014

[12] Vadgasiya M G, Jagani J M. An enhanced algorithm for improved cluster generation to remove outlier's ratio for large datasets in data mining[P]. Development,2014

[13] Ng R T,Han J. CLARANS:A method for clustering objects for spatial data mining[J]. IEEE Trans. Knowl. Data Eng.,2002,14(5):1003-1016

[14] Kaufman L,Rousseeuw P J. Finding Groups in Data:An Introduction on Cluster Analysis[M]. John Wiley and Sons,1990

[15] Ng R T,Han J. CLARANS:A method for clustering objects for spatial data mining[J]. IEEE Trans. Knowl. Data Eng.,2002,14(5):1003-1016

[16] Zhang T,Ramakrishnan R,Livny M. BIRCH:An efficient data clustering method for very large database[C] // SIGMOD Conference. 1996:103-114

[17] Zhang T,Ramakrishnan R,Livny M. BIRCH:An efficient data clustering method for very large database[C] // SIGMOD Conference. 1996:103-114

[18] Guha S,Rastogi R. CURE:An efficient clustering algorithm for large database[J]. Inf. Syst.,2001,26(1):35-58

[19] Bu F,Chen Z,Zhang Q,et al. Incomplete Big Data Clustering Algorithm Using Feature Selection and Partial Distance[C] // 5th International Conference on Digital Home (ICDH). IEEE,2014:263-266

[20] Kim B J. A Classifier for Big Data[M] //Convergence and Hybrid Information Technology. Springer Berlin Heidelberg,2012:505-512

[21] Dhillon I S,Modha D S. A data-clustering algorithm on distributed memory multiprocessors[M] // Large-Scale Parallel Data Mining. Springer Berlin Heidelberg. 2000:245-260

[22] Stoffel K,Belkoniene A. Parallel k/h-means clustering for large data sets[M] // Euro-Par'99 Parallel Processing. Springer Berlin Heidelberg,1999:1451-1454

[23] Nagesh H S,Goil S,Choudhary A. A scalable parallel subspace clustering algorithm for massive data sets[C] // International Conference on Parallel Processing,2000. IEEE,2000:477-484

[24] Ng M K, H Zhe-xue. A Parallel k-Prototypes Algorithm for Clustering Large Data Sets in Data Mining[J]. Intelligent Data Engineering and Learning,1999:263-290

[25] Davidson I,Satyanarayana A. Speeding up k-means clustering by bootstrap averaging[J]. IEEE Data Mining Workshop on Clustering Large Data Sets,2003,133(12):982-992

[26] Farnstrom F,Lewis J,Elkan C. Scalability for clustering algorithms revisited[J]. ACM SIGKDD Explorations Newsletter,2000,2(1):51-57

[27] Domingos P,Hulten G. A general method for scaling up machine learning algorithms and its application to clustering[C] // ICML. 2001:106-113

[28] Cui X,Zhu P,Yang X,et al. Optimized Big Data K-means clustering using MapReduce[J]. The Journal of Supercomputing,2014,70(3):1249-1259

[29] Zhao Y,Chen Y,Liang Z,et al. Big Data Processing with Probabilistic Latent Semantic Analysis on MapReduce[C] // International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery. 2014:162-166

[30] Younghoon K, Kyuseok S, Min-Soeng K, et al. DBCUREMR:An efficient density-based clustering algorithm for large data using MapReduce[J]. Information Systems,2014,42:15-35

[31] Jianqiang D, Fei W, Bo Y. Accelerating BIRCH for clustering large scale streaming data using CUDA dynamic parallelism [M]. Intelligent Data Engineering and Automated Learning-IDEAL 2013. Springer Berlin Heidelberg,2013:409-416

(上接第 379 页)

[58] Gama J,Rocha R,Medas P. Accurate decision trees for mining high-speed data streams[C] // Proceedings of the ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM,2003:523-528

[59] Bifet A, Holmes G,Kirkby R,et al. Moa:Massive online analysis [J]. The Journal of Machine Learning Research,2010,11:1601-1604

[60] Neumeyer L, Robbins B, Nair A, et al. S4:Distributed stream computing platform[C] // 2010 IEEE International Conference on Data Mining Workshops (ICDMW). IEEE,2010:170-177

[61] Wojnarski M. Debellor:a data mining platform with stream architecture[M] // Transactions on Rough Sets IX. Springer Berlin Heidelberg,2008:405-427

[62] Fan C Y,Chang P C,Lin J J,et al. A hybrid model combining case-based reasoning and fuzzy decision tree for medical data classification[J]. Applied Soft Computing,2011,11(1):632-644

[63] Wu M C,Lin S Y,Lin C H. An effective application of decision tree to stock trading[J]. Expert Systems with Applications,2006,31(2):270-274

[64] Kim J W,Lee B H,Shaw M J,et al. Application of decision-tree induction techniques to personalized advertisements on internet storefronts[J]. International Journal of Electronic Commerce,2001,5(3):45-62

[65] Zeitouni K, Chelghoum N. Spatial decision tree-application to traffic risk analysis[C] // ACS/IEEE International Conference on Computer Systems and Applications. IEEE,2001:203-207