基于 MapReduce 范式的 Apache Spark 异构集成分类器大数据分类

Hamidreza Kadkhodaei, Amir Masoud Eftekhari Moghadam, Mehdi Dehghan

摘要

在这个大数据时代，如何高效、准确地处理大规模数据已经成为一个具有挑战性的问题。作者提出一种有能力处理大数据集的，采用了 Boosting 技术的异构集成学习模型命名为 DHBoost，并改写成 Spark 的 MapReduce 范式。本项目在代码未开源的情况下使用 SparkML 库进行复现，并在自己搭建的小集群上进行实验。

关键词：集成分类器；boosting；MapReduce；大数据；Apache Spark；Apache Hadoop

1 引言

如今，数据生成的速度增长如此之快，几乎每两年世界上的数据量就会翻一番。由于这种大数据的庞大体积和固有的复杂性，传统的数据处理方法已不适合处理。在一台机器上存储和处理大数据几乎是不可能的。因此，对大数据处理速度的需求日益增长，许多研究人员致力于并行处理，以提高算法的速度。值得注意的是，除了并行求解外，还需要修改算法和方法来应对大数据问题。

分类是一种有监督的机器学习，它从正确分类的数据集建立模型。传统的分类算法，构建一个整体的泛化的分类器以普遍优于其他分类算法在某种程度上是不可实现的，这意味着每种分类算法在不同数据集上的表现不同。解决这一问题的一个好办法是考虑多种分类算法，并通过融合它们的输出来构建一个混合模型，以提高最终模型的预测性能。大数据背景下的分类问题是一个具有挑战性的问题，需要考虑其准确性和效率。

2 相关工作

2.1 前人工作

许多集成分类器是受到 Boosting 方法的启发。其中最受欢迎的是 1997 年 Freund 提出的 AdaBoost.M1，它支持多类数据集，并已应用于各种各样的应用程序[1]。目前对于加速 AdaBoost.M1 算法已经有许多工作提出。Lazarevic 提出了一种并行版本的 Boosting，但是它是为紧密耦合的共享内存系统设计的，因此不能应用于分布式云计算环境[2]。

Fan 将样本分布在几台机器上并采用分布式采样，用一小部分的数据集训练 AdaBoost 分类器得到弱学习器[3]。Gambs 提出两种方法不需要显式地共享数据集的分布式 Boost 算法：BIBOOST 和 MULTBOOST，主要用于保护分布式数据的隐私性[4]。Palit 提出了一种 AdaBoost.M1 并行化的 MapReduce 方法[5]。首先，将数据集划分为一些分区，在 Map 阶段，从每个分区中的实例训练一个 Boosting 分类器。在 Reduce 阶段，将生成的分类器进行组合以构建最终的集成分类器。

梯度提升分类器 (GBT) 是由 Friedman 在 2001 年提出的。GBT 的基础学习器是决策树。在每次迭代中，新的分类器拟合上一次迭代的残差，从而提升模型。通过计算损失函数来检测残差，并将每次

迭代的结果进行聚合得到最终模型。GBT 在分类和回归任务中都非常高效，被包括在许多机器学习包中，如 R, Python Scikit-learn 和 Spark MLlib。与许多其他增强方法一样，GBT 对噪声实例和异常值非常敏感。随着树木数量的增加，它们也容易过拟合。

受 GBT 的启发，2016 年 Chen 推出了 XGBoost[6]。它能较好地控制过拟合问题，且训练时间较短。LightGBM 是另一种基于 GBT 的分类器[7]。其目的是排除小梯度的训练实例，只使用剩下的部分来估计信息增益。因此，它可以比其他基于 GBT 的分类器训练得更快。然而，到目前为止的工作都没有尝试使用多个学习算法而不是单个学习算法的 boost 基分类器。

2.2 HBoost

作者在 2020 年针对小数据集提出了一种基于异构增强的分类器，其步骤如图 1所示[8]。其目的是使用多种学习算法作为增强和修剪生成的分类器的基本分类器，以增加多样性。每一种学习算法都会产生许多增强的分类器。然后根据分类器的精度对分类器进行排序，并对同一分类级别的分类器进行分组修剪，以增加分类器的多样性。最后，将剩下的分类器组合起来生成最终的输出。与最先进的集成方法 (如 Bagging、AdaBoost 和堆栈泛化) 相比，HBoost 具有更好的准确性。

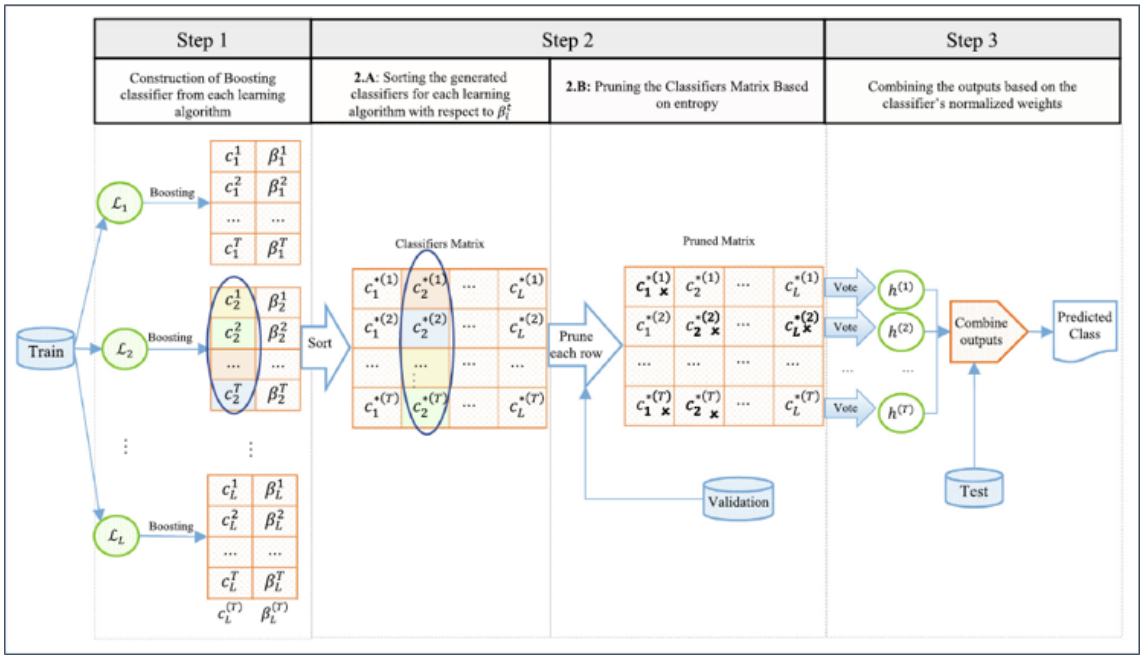

图 1: HBoost 步骤

3 本文方法

复现的论文扩展了 HBoost，通过使用 MapReduce 编程范式和 Apache Spark 框架将任务分布到集群节点上，使其能够处理大数据集，并命名为 DHBoost。该方法与 HBoost 的不同之处在于:(1) 它在训练每个基分类器时不考虑整个训练样本，这意味着首先将整个训练样本划分为一些不相交的区域，并为每个区域分配一个学习算法，该算法使用区域内的样本进行训练，并使用 AdaBoost.M1 生成 T 个基分类器。(2) 使用不同的策略对分类器池进行剪枝。其步骤可以用图 2表示。

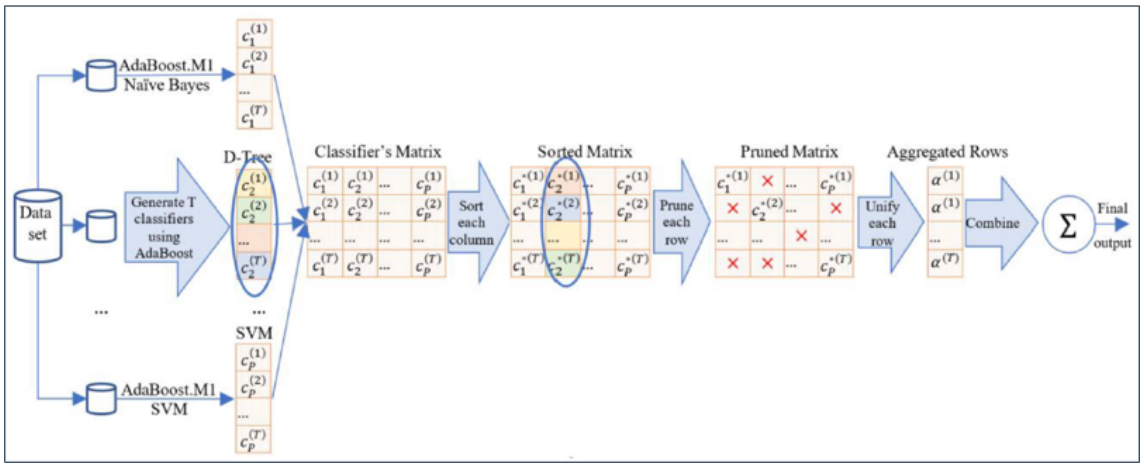

图 2: DHBoost 步骤

3.1 数据分区、训练

数据集存储在分布式文件系统上，例如 HDFS。首先将训练数据切分成若干块，以提高并行性和数据可拓展性。切分的数据块可以与使用的基学习算法数目相同，每一个数据划分分配一个学习算法并使用 AdaBoost.M1 方法进行集成学习。假设将数据集被划分为 P 个分区，每个分区使用 AdaBoost.M1 方法迭代训练 T 次，则最后形成一个 P × T 个分类器的分类器池和对应的权重矩阵。

3.2 分类器排序、修剪

修剪生成的分类器池可以增加分类器之间的多样性，提高最终模型的预测性能。这也使最终的模型更简单。在应用剪枝步骤之前，我们将分类器分类到一些组中，并对每个组进行剪枝，以在其分类器之间达到更好的多样性。为此目的，每个分区生成的分类器按照分配的权重按降序排序。分类器的排序和分组背后的动机是使每个分区的相似分类器在同一排序级别中对齐。这意味着每个分区中最准确的分类器归到同一组中，并相互比较。

使用文献中常见的熵测度进行对分类器多样性进行评估。与 HBoost 不同，DHBoost 中使用贪心算法获得可接受的熵值。也就是说，在每一行中都选择权重最高最准确的分类器，这样熵就会增加。这个过程一直持续到没有更多的分类器可以增加熵。尽管这种方法不能保证最大的熵，但它在合理的时间内给出了一个可接受的答案。

3.3 分类器集成输出

对分类器矩阵进行修剪后，对剩下的分类器应用加权投票的形式将输出聚合为单个输出。

4 复现细节

4.1 与已有开源代码对比

复现的论文未提供源码。根据作者在文中的描述采用 Scala 并调用 Spark.ml 库编写 Spark 程序。原文使用 Spark 版本 2.4.5，Hadoop 版本 2.7，Scala 版本 2.11，由于低版本不支持某些机器学习库，复现使用的版的本为更新的版本，将在 4.2 小节列出；数据集中，原文使用了 UCI7 个开源数据集，复现中选用其中 Susy 和 Higgs-True 两个数据集进行实验；原文使用基学习算法为：朴素贝叶斯、决策树、逻辑回归和支持向量机，迭代次数为 100，复现工作中选用了决策树、逻辑回归和支持向量机作为基学习算法，迭代次数也没有 100 次这么多，因为观察到往后的效果提升较小；数据切分中，考虑到使用小集群进行实验，将数据块分太多反而会降低运算效率，原文将数据切分成 16 ~1024 块，而复现工

作中只是简单地将数据切分为与基学习算法数目一样；原文中使用了信息熵进行剪枝，复现工作中简单的使用了权重大小进行剪枝。最后和原文一样使用 SparkML 内置的分类算法 RandomForest 和 GBT 进行比较。

4.2 实验环境搭建

如图 3所示在 VMware ESXi 上创建了 5 台虚拟机 gaia0 gaia4 作为本次实验的集群环境。每个节点分配 4 个 CPU，20GB 内存以及 200GB 的磁盘空间。安装的虚拟机版本为 Centos7.5，实验环境使用 JDK1.8，Scala2.13.10，Hadoop3.3.4，Spark3.3.1。实验中 Spark 运行模式均使用 yarn 模式。

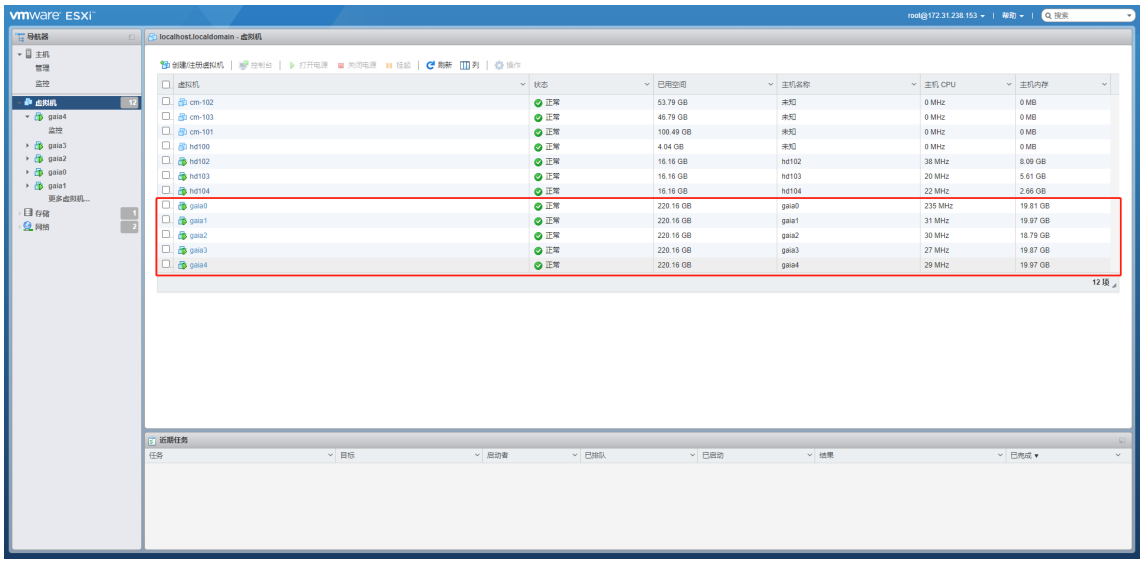

图 3: 集群搭建

以 gaia0 作为主节点部署 Hadoop 和 Spark，并配置开启相对应的服务如 HDFS、History Sever。工作目录为 hdfs://user/chandler/recurrence。图 4为部署的 HDFS 服务的 Web 界面，图 5为部署的 Spark 历史服务器，可以监控到各个任务的执行进度和结果。

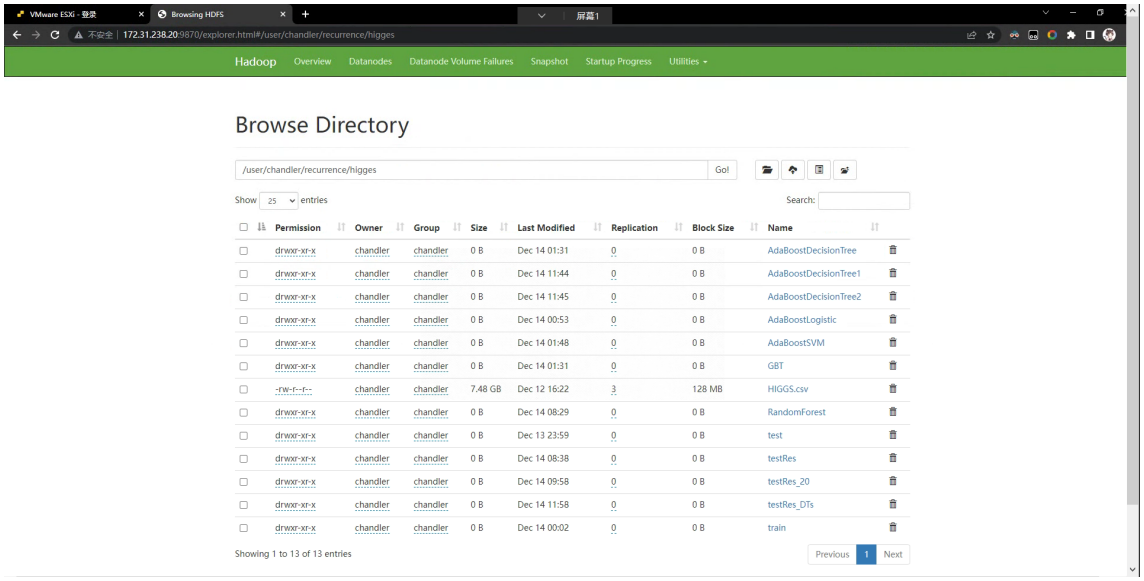

图 4: HDFS

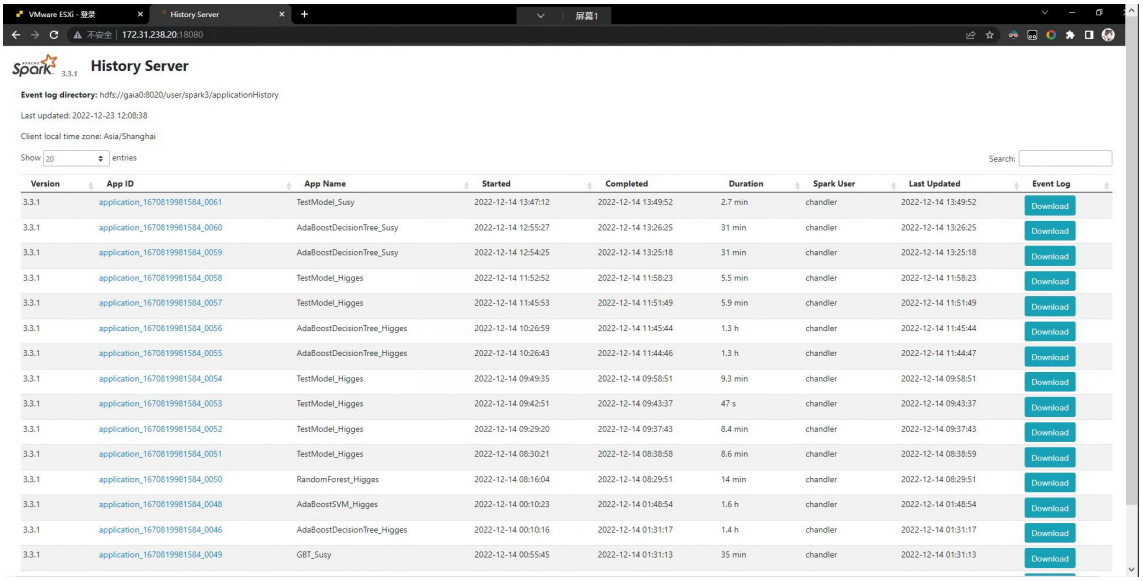

图 5: historysever

4.3 创新点

结果中发现 AdaBoost 方法应用在决策树 DecisionTree（DT）上较为稳定且效果较好，增加了对 AdaboostDT 进行剪枝、集成的实验。

5 实验结果分析

5.1 模型训练

图 6a 是在 Susy 数据集中使用 AdaBoost.M1 方法，使用决策树为基学习算法迭代训练的收敛曲线。图 6b 是在 Higge 数据集中使用 AdaBoost.M1 方法，使用决策树为基学习算法迭代训练的收敛曲线。可以观察到，随着迭代次数的增加，集成的模型准确率不断增加。

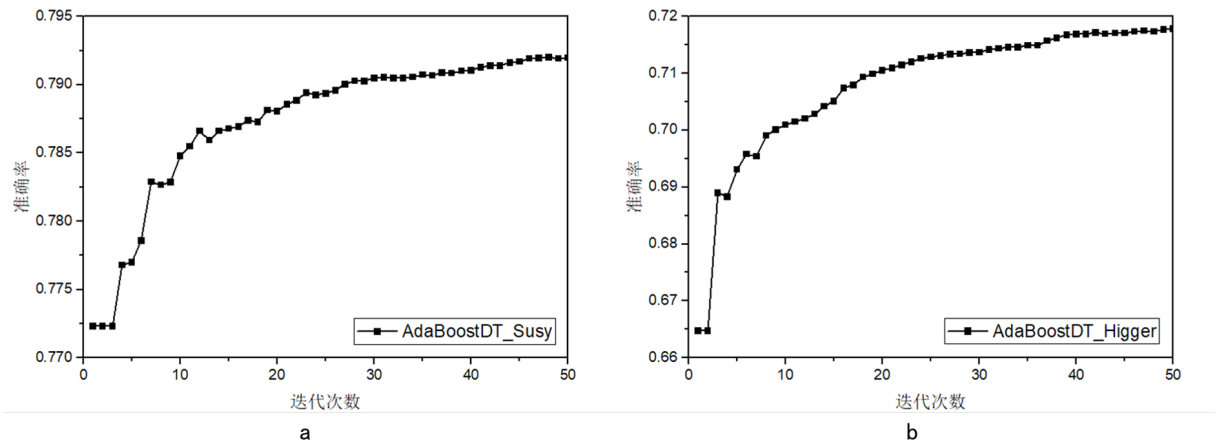

图 6: a. Susy 数据集训练收敛曲线　　b. Higge 数据集训练收敛曲线

5.2 准确率对比

表 1是最终的准确率结果。其中 MyDHBoost 是本次复现的结果，MyDHBoost0 是未经剪枝的结果，DHBoost 是原文给出的结果。AdaBoostDT、AdaBoostLogistic 和 AdaBoostLinearSVC 分别是未经聚合输出的三个分类器，AdaBoostDTAgg 是增加的对 AdaBoostDT 的剪枝集成输出模型，在 Higge 数据集中由于使用逻辑回归和支持向量机的方法准确率与使用决策树的准确率相差较大，故单纯使用决策树进行集成的效果会更加好，而在 Susy 数据集中则相反。在两个数据集中，复现的 demo 准确率大

表 1: 各模型准确率对比

数据集	准确率								
	MyDHBoost	MyDHBoost0	DHBoost	spark_GBT	spark_RF	AdaBoostDT	AdaBoostLogistic	AdaBoostSVC	AdaBoostDTAgg
Susy	0.7902	0.7896	0.8012	0.7934	0.7751	0.7928	0.7886	0.7876	0.7893
Higge	0.6867	0.6865	0.7364	0.7052	0.6599	0.7012	0.6642	0.6642	0.7087

于 Spark 内置的 RF 分类器但是小于 Spark 内置的 GBT 分类器；Suys 数据集中，复现的 demo 准确率与原文的相近；对生成的分类器矩阵剪枝后模型准确率有了一定的上升，初步验证其理论的正确性。

对结果进行初步分析，性能不如原文主要在于：小集群运行，性能有限，数据分块少。选用的基学习算法较少，选用的学习算法对选用的数据集分布不友好。

6 总结与展望

本次复现按照自己对原文的理解和对 Scala、Spark 的理解复现了 DHBoost，比较简陋简单，结果上与原文工作存在一定的差距。对于改进方向，可以考虑改变数据分块策略。DHBoost 进行数据分块时，简单的使用 mapReparationWithIndex 算子对数据进行分块，属于块抽样，会改变数据分布函数，这对大数据来说是不必甚至不利的。考虑使用 RSP(Random Sample Partition) 技术，每个机器学习算法训练具有相同数据分布的数据块，而且能够处理无限规模的数据。复现工作中，为了实现其思想而简单的在小集群上进行运行。但是原文工作目的是为了解决大数据问题，大集群计算是需要的。在其基础上的时间开销会有明显的区别。

参考文献

[1] FREUND Y, SCHAPIRE R E. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting[J/OL]. Journal of Computer and System Sciences, 1997, 55(1): 119-139. https://www.sciencedirect.com/science/article/pii/S002200009791504X. DOI: https://doi.org/10.1006/jcss.1997.1504.

[2] LAZAREVIC A, OBRADOVIC Z. Boosting algorithms for parallel and distributed learning[J]. Distributed and parallel databases, 2002, 11(2): 203-229.

[3] FAN W, STOLFO S J, ZHANG J. The application of AdaBoost for distributed, scalable and on-line learning[C]//Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining. 1999: 362-366.

[4] GAMBS S, KÉGL B, AÏMEUR E. Privacy-preserving boosting[J]. Data Mining and Knowledge Discovery, 2007, 14(1): 131-170.

[5] GARCÍA-GIL D, LUENGO J, GARCÍA S, et al. Enabling smart data: noise filtering in big data classification[J]. Information Sciences, 2019, 479: 135-152.

[6] CHEN T, GUESTRIN C. Xgboost: A scalable tree boosting system[C]//Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. 2016: 785-794.

[7] KE G, MENG Q, FINLEY T, et al. Lightgbm: A highly efficient gradient boosting decision tree[J]. Advances in neural information processing systems, 2017, 30.

[8] KADKHODAEI H R, MOGHADAM A M E, DEHGHAN M. HBoost: A heterogeneous ensemble classifier based on the Boosting method and entropy measurement[J]. Expert Systems with Applications, 2020, 157: 113482.