

MapReduce 环境下处理多类别不平衡数据的 改进随机森林算法

姚 立, 张曦煌

(江南大学 物联网工程学院, 江苏 无锡 214122)

摘 要: 针对传统 MapReduce 环境下的随机森林算法在处理多类别不平衡数据问题时仍然以全局最优值作为划分点, 忽略了少数类对分类准确率的影响, 本文提出了一种 MapReduce 环境下处理多类别不平衡数据的改进随机森林算法(MR-RF-SHDSE). 该算法利用分层采样方法在各个类别中进行样本抽样, 并以 HDDT 决策树作为基学习器以弱化数据偏置给分类准确率带来的影响, 最后计算决策树的 GMean 值和不合度值, 利用调和平均值作为衡量标准对决策树进行选择集成. 通过实验证明, 相比其他算法, MR-RF-SHDSE 能够有效提高了对多类别不平衡数据集的分类准确率.

关键词: MapReduce; 随机森林; 分层采样; HDDT 决策树; 选择集成

中图分类号: TP301.6

文献标识码: A

文章编号: 1000-7180(2018)11-0139-06

An Improved Random Forest Algorithm for Multi-Class Imbalanced Data Problem Under Map Reduce

YAO Li, ZHANG Xi-huang

(School of IOT Engineering, Jiangnan University, Wuxi, 214122, China)

Abstract: Because the traditional random forest algorithm under the MapReduce still takes the global optimal point as the dividing point when dealing with the multi-class imbalance data problem, ignoring the influence of the minority class on the classification accuracy rate, this paper presents an improved random forest algorithm (MR-RF-SHDSE) for dealing with multi-class imbalance data under MapReduce. This algorithm uses the stratified sampling method to sample the samples in each category, and uses the HDDT decision tree as the learner to weaken the impact of data bias on the classification accuracy. Finally, the GMean value and the disagreement measure value of the decision tree are calculated, we use harmonic mean as a metric to select decision trees. Experiments show that compared with other algorithms, MR-RF-SHDSE can effectively improve the classification accuracy of multi-class imbalanced data sets.

Key words: mapreduce; random forest; stratified sampling; hellinger distance decision tree; selective ensemble

1 引言

随机森林是目前流行的一种简单高效的算法. Fernández-Delgado^[1]等曾经指出随机森林的综合表现最好, 然而 Zhou^[2]等认为从所有基学习器中选择某些基学习器可以比所有的基学习器的学习效果更好, 这为进一步提升随机森林的分类效果指明了

方向.

目前国内外众多学者对不平衡数据有一定的研究, 然而很少有学者研究了在分布式环境下多类别不平衡算法的问题. 由于 MapReduce^[3]的数据划分会大大加剧了数据的不平衡性, 所以很有必要对此问题进行研究.

本文提出了一种 MapReduce 环境下处理多类

别不平衡数据的改进随机森林算法 (MR-RF-SHDSE), 该算法利用分层采样 (Stratified sampling) 方法在各个类别中进行样本抽样, 并以 HD-DT 多分类决策树作为基学习器, 最后针对训练出来的决策树计算其 GMean 值与不合度值, 提出了一种选择集成方法 (Selective Ensemble). 通过实验证明, 相比其他算法, MR-RF-SHDSE 能够有效提高了对多分类不平衡数据集的分类准确率.

2 MR-RF-SHDSE 算法

2.1 融入分层抽样的 bootstrap 方法

传统的面向平衡数据集的 bootstrap 算法往往有效, 然而面对海量不平衡数据时, 由于 bootstrap 的随机性, 大约有 36.8% 的数据未能采样得到^[4], 又由于 MapReduce 对数据集又再次进行的划分, 所以这会加剧数据集的不平衡. 分层抽样^[5]是指将数据空间划分为多个层次, 然后在每层中随机抽样. 在这里我们以类别作为层次, 通过在数据分布较多的类别中抽取较多的样本, 而在数据分布少的类别上抽取较少的样本, 从而使得样本具有较强的代表性.

2.2 处理多类别不平衡数据集的决策树集成方法

在 MapReduce 环境下, 根据随机森林的算法步骤, 每一个 Mapper 中都会产生一定数量的决策树. 不同的 Mapper 中的决策树之间有一定的差异, 同一 Mapper 下的决策树之间也有一定的差异, 这是由于在采样数据的时候使用了分层 bootstrap 采样的数据、采样方法和在构建决策树的时候随机进行了属性选择, 这两种方法保证了随机森林的多样性, 从而能够学习到不同子空间的数据规律. 然而由于随机性, 在采样数据和属性选择的时候并没有考虑到数据之间和属性之间的相关性, 所以并不能考虑各个决策树预测准确性和各个决策树之间的差异性. 由于基分类器之间的差异性和基分类器自身的准确性是影响集成算法泛化性能的两个重要因素^[2]. 所以, 要考虑如何找到一棵树, 使得该树对于不平衡数据有良好的精度并且这棵树与其他决策树之间有一定的差异性.

通常利用袋外数据以及精度值来衡量决策树的强度. 然而由于现在面对的是不平衡数据集, 这种评价标准是有缺陷的. 针对不平衡数据, 为了更好地评价决策树分类器的效果, 文本采用 GMean 值来衡量决策树的强度. 以二分类为例, 存在以下混淆矩阵:

表 1 二分类混淆矩阵

真实情况	预测结果	
	正例	反例
正例	TP	FN
反例	FP	TN

则 $GM(i, j) = \sqrt{sensitivity \times specificity}$ (1)

式中, $sensitivity = \frac{TP}{TP + FN}$,

$$specificity = \frac{TN}{FN + TN}$$

然而以上是二分类的情况, 针对多分类的情况就要将其拓展, 其拓展后的公式如下所示^[6]:

$$GM = \frac{2}{C(C-1)} \sum_{1 \leq i, j \leq C \text{ 且 } i \neq j} \overline{GM(i, j)} \quad (2)$$

式中, $\overline{GM(i, j)} = \frac{GM(i, j) + GM(j, i)}{2}$ (3)

可以看出, 公式(3) 中将少数类和多数类进行了调换, 认为多数类也是值得关注的, 然后求其平均值, 但是因为我们只想关注少数类, 所以对以上的公式做了改进, 改进后的公式如下所示.

$$GM = \frac{2}{C(C-1)} \sum_{1 \leq i, j \leq C \text{ 且 } i \neq j} GM(i, j) \quad (4)$$

文献[7] 中曾提出了多种多样性度量方案, 文本采用其中的不合度量 (disagreement measure) 来衡量两棵决策树之间相似度. 其中不合度量的定义如下所示:

对于给定数据集 $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$ 和两个分类器 C1 和 C2, 它们的分类效果如表 2 所示:

表 2 分类器 C1 和 C2 的分类结果联表

	C1 分类正确数量	C1 分类错误数量
C2 分类正确数量	A	B
C2 分类错误数量	C	D

则 $Sim(C1, C2) = Dis(C1, C2) = \frac{B + C}{A + B + C + D}$ (5)

通过公式(4) 和公式(6), 我们可以得到一棵决策树的强度和任意两棵决策树之间的相似度. 为了能够找到一个 Mapper 下有一定差异性的并且有一定强度的决策树, 本文提出如下计算步骤:

步骤一: 首先对于一个 Mapper 下的所有决策树的强度和多样性按公式(6) 和(7) 归一化:

$$TS_i = \frac{GM_i - GM_{\min}}{GM_{\max} - GM_{\min}} \quad (6)$$

$$TD_i = \left(\sum_{j=1, i \neq j}^T \frac{Sim(i, j) - Sim(i)_{\min}}{Sim(i)_{\max} - Sim(i)_{\min}} \right) / (T-1) \quad (7)$$

步骤二:为了衡量决策树 DT_i 在强度空间和相似度空间的总体平均分布程度,计算决策树 DT_i 强度和相似度的调和平均数:

$$HM_i = 2 / \left(\frac{1}{TS_i} + \frac{1}{TD_i} \right) \quad (8)$$

步骤三:将 Mapper 下所有计算出来的决策树的调和平均值从高到低排序,按阈值 m 选出前 m 个决策树作为最终的决策树集合。

综上所述,结合强度和相似度,决策树的集成策略的伪代码如下所示:

SHDSE 算法:

输入:分区数据集 S ,决策树集成数量阈值 m

输出:决策树集成集合

1、从分区数据集 S 上按分层抽样获得训练集合 T 与验证数据集 V 和决策树数据量 n

2、For i to n :

3、 $baggingData \leftarrow T$; $outBaggingData \leftarrow T$ // 按分层抽样获取 $bagging$ 数据集与袋外数据集

4、 $DT_i \leftarrow buildHDDT(baggingData)$ // 构建决策树

5、 $DT_i - GMean \leftarrow calcGMean(outBaggingData)$ // 计算决策树的强度 $GMean$ 值

6、End for

7、 $new\ matrix[n][n]$ // 用于保存决策树相似度值

8、For i to n :

9、For j to n :

10、if $(i \neq j)$:

11、 $matrix[i][j] \leftarrow calcSim(i, j, V)$ // 利用验证数据集 V 计算两棵决策树的相似度值

12、End for

13、End for

14、For i to n :

15、 $DT_i - TS \leftarrow calcTS(DT - GMean - List)$ // 归一化强度

16、 $DT_i - TD \leftarrow calcTD(i, matrix)$ // 归一化相似度

17、End for

18、For i to n :

19、 $HM_i = calc(DT_i - TS, DT_i - TD)$ // 计算平均调和值

20、End for

21、Sort($HM - List$) // 按平均调和值降序排序选出前 m 棵决策树

22、返回决策树集成集合

2.3 MR-RF-SHDSE 模型构建预测流程

根据 MapReduce 模型,将模型的构建预测流程分为三个阶段。

(1)初始阶段:在该阶段中,由 MapReduce 自动进行

数据划分,然后将划分后的数据分派到各个 Mapper 中。

(2)Map 阶段:

输入: $\langle key, value \rangle$ 键值对, key 为二进制流, $value$ 为 MapReduce 框架划分的的数据集

输出: $\langle key', value' \rangle$ 键值对, key' 为测试数据集的索引, $value'$ 为决策树的预测结果。

1. $treeList \leftarrow getTreeListBySHDSE()$ // 按照 SHDSE 算法生成决策树集合。

2. $testData \leftarrow context.getCacheFiles();$ // 利用 MapReduce 框架的 $cacheFile$ 功能读入测试数据。

3. For i to $testData.length$:

4. For n to $treeList.length$:

5. $key' \leftarrow testData_i.index$;

6. $value' \leftarrow treeList_n.predict(testData_i)$;

7. $context.write(key', value')$;

8. End for

9. End for

(3)Reduce 阶段:

输入: $\langle key', value' \rangle$ 键值对, key' 为测试数据集的索引, $value'$ 为所有 map 中决策树对同一条数据的预测结果集合。

输出: $\langle key'', value'' \rangle$ 键值对, key'' 为测试数据索引, $value''$ 为预测结果值。

1. For i to $value'.length$:

2. $hashMap.put(value'_i, value'_i.count ++)$; // 利用 HashMap 类存储预测类别的数量

3. End for

4. $key'' \leftarrow key'$ // 测试数据的索引

5. $value'' \leftarrow max(hashMap, value).key$ // 获取数量最多的那个类别

6. $context.write(key'', value'')$;

3 实验与分析

3.1 实验环境

本文的实验环境由 4 台集群结点组成,选取 1 台作为 NameNode,另外三台作为 WorkNode。每个节点的配置如下:硬件配置为 Intel Core i5-2500CPU、内存 16G 和 1TB 硬盘;软件配置为 CentOS6.5, JDK8u151, Hadoop 版本 CDH5.4.5;4 个节点都连接在 100Mbps 的交换机上。

3.2 实验评价指标

因为实验所使用的数据都是不平衡数据集,本文采用 $GMean$ 值和 $f-\beta$ -measure 值作为衡量分类的准确率指标。其中 $GMean$ 值如公式 4 所示。

而 $f-\beta$ -measure 指标的指标如下式所示:

$$f-\beta\text{-measure} =$$

$$\frac{(1+\beta^2)(positive \times sensitivity)}{(\beta^2 \times positive)+sensitivity} \quad (11)$$

式中, $sensitivity = \frac{TP}{TP + FN}$; $positive = \frac{TP}{TP + FP}$; β 的取值范围是 $[0, +\infty]$. 在本章实验中 β 取值为 1.

3.3 实验数据集

本文的实验数据集采用了 KDD Cup 1999 dataset(KCD)和 Poker Hand dataset(PHD). KCD 数据集一共有 4 898 431 条数据,41 个属性,有 5 个类别分别是 DOS,NOR,POR,R2L 和 U2R,它们的数量之比为 3883370:972781:41102:1126:52;PHD 数据集一共有 1 000 000 条数据,有 10 个属性,有 0 到 9 共 10 个类别,所有类别的数量之比为 501209:422498:47622:21121:3885:1996:1424:230:12:3. 为了能够较为清楚显示算法在多类别情况下的评价指标. 本章实验将 KCD 和 PHD 两个数据集人为地分为三类多分类数据集,这个三类数据集的最大不平衡比例由高到低,每个数据集类别比例可以从数据集介绍中知晓. 通过实验,比较经典算法与本文提出的算法在不同数据集上的指标差异.

表 3 实验数据集

数据集	实例数量	最大不平衡比例 (IR)%	
KCD	DOS/NOR/U2R	4 856 203	74 680. 19
	NOR/R2L/U2R	973 959	18 707. 32
	DOS/NOR/POR	4 297 253	94. 48
	KCD_ALL	4 898 431	74 680. 19
PHD	0/7/8/9	501 454	167 069. 66
	0/1/6/7	925 361	2 179. 16
	4/5/6	7305	2. 72
	PHD_ALL	1 000 000	167 069. 66

3.4 实验对比算法以及参数设置

为了验证本文的算法效果,分别选取了传统 MR-RF 算法、基于欠采样的 MR-RF-US 算法^[8]、基于过采样的 MR-RF-OS 算法^[8]、基于 HDDT 决策树的 MR-RF-HD 算法以及文本提出的 MR-RF-SHDSE 算法. 由于过采样算法和欠采样算法都是基于二分类的算法,本文采用“一对一”投票的方法将其拓展成多分类算法.

MR-RF,MR-RF-US 和 MR-RF-OS 这三种算法使用 CART 决策树作为基分类器. 并且这 5 种算法中决策树的最大深度设置为无限,设置每种算法每次选取属性的数量为 $\log_2(M)+1$,M 表示的所有属性的数量,设置决策树的数量为 200. 本文采用 5

次 5 折交叉验证,所有结果为其均值.

3.5 实验结果分析

3.5.1 默认设置下各个数据集的指标对比

表 4 和表 5 记录了在各个实验数据集上的 GMean 值和 f- β -measure,粗体表示最高值. 从表中可以总结出以下结论.

(1)MR-RF-SHDSE 在 7 个数据集上的两项指标最高. 而由于 PHD-4/5/6 数据集相比其他数据集而言相对平衡,所以不及传统的 MR-RF 算法,但是其值相差不大.

(2)由于 MR-RF-OS 算法是对数据集进行随机过采样,从而会增加数据集的数据量,尤其是当数据集中数据量较多的类别作为少数类时,使用的内存会超过最大内存,从而导致内存溢出(oom).

(3)由于 MR-RF-US 算法抛弃了多数类的数据集,对多数类与少数类的训练数据数量进行了平衡,从而能够使 MR-RF-US 算法在两种指标上相比 MR-RF 算法有一定的提升,但是由于 MR-RF-US 算法在训练时抛弃了部分数据,使得算法未能学习到被抛弃数据的内在规律,所以其指标相比 MR-RF-HD 和 MR-RF-SHDSE 较低.

(4)MR-RF-SHDSE 算法相比 MR-RF-HD 算法而言其 GMean 平均值和 f- β -measure 平均值分别高出了 1. 30%和 0. 966%,从而验证了本文提出的算法有效性.

表 4 各个数据集下 GMean 值

数据集	MR-RF	MR-RF-US	MR-RF-OS	MR-RF-HD	MR-RF-SHDSE
DOS/NOR/U2R	0. 424	0. 866	oom	0. 912	0. 921
NOR/R2L/U2R	0. 468	0. 894	oom	0. 901	0. 914
DOS/NOR/POR	0. 814	0. 919	oom	0. 928	0. 940
KCD_ALL	0. 373	0. 830	oom	0. 896	0. 909
0/7/8/9	0. 336	0. 899	oom	0. 909	0. 918
0/1/6/7	0. 533	0. 912	oom	0. 910	0. 922
4/5/6	0. 929	0. 904	0. 892	0. 905	0. 918
PHD_ALL	0. 335	0. 836	oom	0. 868	0. 881

表 5 各个数据集下 f- β -measure 值

数据集	MR-RF	MR-RF-US	MR-RF-OS	MR-RF-HD	MR-RF-SHDSE
DOS/NOR/U2R	0. 401	0. 786	oom	0. 908	0. 919
NOR/R2L/U2R	0. 459	0. 876	oom	0. 899	0. 908
DOS/NOR/POR	0. 788	0. 895	oom	0. 909	0. 919
KCD_ALL	0. 367	0. 825	oom	0. 885	0. 893
0/7/8/9	0. 329	0. 883	oom	0. 903	0. 910
0/1/6/7	0. 499	0. 893	oom	0. 879	0. 883
4/5/6	0. 913	0. 871	0. 875	0. 897	0. 908
PHD_ALL	0. 318	0. 815	oom	0. 853	0. 862

3.5.2 不同数量的 mapper 下各个算法之间的指标对比

随着 mapper 数量的增加,每个 mapper 上数据量也随之减少,在上一个实验中由于过采样的而出现的内存溢出现象没有出现.通过表 6 和表 7 可以看出,MR-RF 两项指标下降较快,主要是因为 MR-RF 算法本身对不平衡数据敏感;又因为随着每个 mapper 上的数据量减少,MR-RF-US 使用了欠采

样的算法,从而进一步导致算法不能够学习到数据集的内在规律,所以各项指标所有下降;MR-RF-OS 算法指标虽然有所下降,但是差异不大;MR-RF-HD 和 MR-RF-SHDSE 算法总体上表现较为平稳,主要原因仍然是 HDDT 决策树对不平衡数据不敏感的. MR-RF-SHDSE 在不同数量的 mapper 下各个指标最高,从而验证了该算法的优越性.

表 6 不同 mapper 数量下 KCD_ALL 数据集和 PHD_ALL 数据集的 GMean 值表

		12 mappers	18 mappers	24 mappers	30 mappers	36 mappers
KCD_ALL	MR-RF	0.379	0.364	0.352	0.348	0.340
	MR-RF-US	0.817	0.813	0.801	0.798	0.789
	MR-RF-OS	0.828	0.821	0.819	0.815	0.812
	MR-RF-HD	0.893	0.891	0.897	0.889	0.892
	MR-RF-HDSSE	0.903	0.898	0.903	0.897	0.899
PHD_ALL	MR-RF	0.330	0.327	0.319	0.310	0.301
	MR-RF-US	0.832	0.831	0.823	0.816	0.813
	MR-RF-OS	0.840	0.836	0.833	0.829	0.828
	MR-RF-HD	0.865	0.861	0.863	0.859	0.857
	MR-RF-HDSSE	0.883	0.879	0.881	0.882	0.881

表 7 不同 mapper 数量下 KCD_ALL 数据集和 PHD_ALL 数据集下 f-β-measure 值表

		12 mappers	18 mappers	24 mappers	30 mappers	36 mappers
KCD_ALL	MR-RF	0.362	0.360	0.339	0.329	0.325
	MR-RF-US	0.809	0.803	0.796	0.799	0.779
	MR-RF-OS	0.829	0.820	0.816	0.813	0.805
	MR-RF-HD	0.884	0.886	0.879	0.881	0.878
	MR-RF-HDSSE	0.899	0.896	0.902	0.898	0.896
PHD_ALL	MR-RF	0.326	0.322	0.310	0.305	0.291
	MR-RF-US	0.827	0.823	0.818	0.817	0.808
	MR-RF-OS	0.834	0.835	0.825	0.819	0.810
	MR-RF-HD	0.850	0.845	0.851	0.847	0.845
	MR-RF-HDSSE	0.859	0.853	0.858	0.860	0.851

3.5.3 不同数量的 mapper 下各个算法之间的运行时间对比

图 1 和图 2 是 5 种算法在不同数据集上的运行时间比较图.任务完成的总时间为任务初始化时间、模型建立时间和任务调度时间之和.可以看出随着 mapper 数量的增加,运行时间随之减少,但是到达一定数量的 mapper 之后运行时间相差不大,主要是因为随着 mapper 数量的增多,花在任务调度上的时间也随之增加.从两张运行时间比较图中也可以明显看出 MR-RF-HD 算法与 MR-RF 算法在两种数据集上运行时间最少;虽然 PHD_ALL 数据集

的数量相比 KCD_ALL 数据集的数量少,但是因为 MR-RF-US 和 MR-RF-OS 是基于二分类问题的算法,需要将多分类拆解为二分类,随着数据集类别的增多,一定会增加算法运行时间,所以可以明显看出在 PHD_ALL 数据集上 MR-RF-US 和 MR-RF-OS 算法相比其他算法来说花费的时间更长,而在 KCD_ALL 数据集上,由于类别较少,但数据较多,所以也需要花费一定的时间进行计算.最后因为 MR-RF-SHDSE 算法改进了决策树的选择策略,需要评价决策树的强度和计算决策树之间的相似度,所以其运行时间比 MR-RF-HD 算法与 MR-RF 算法长,

但是相比另外两种算法却低得多. 从本实验中可以看出本文提出算法的可行性.

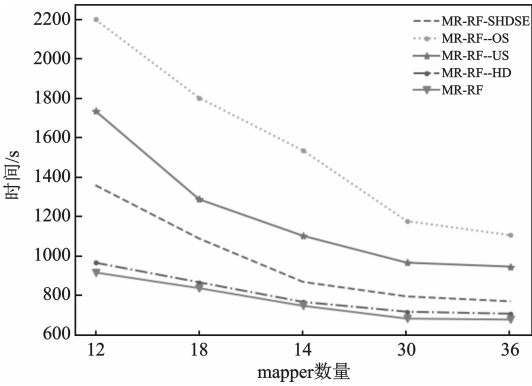


图 1 不同 mapper 数量下 KCD_ALL 数据集处理时间对比图

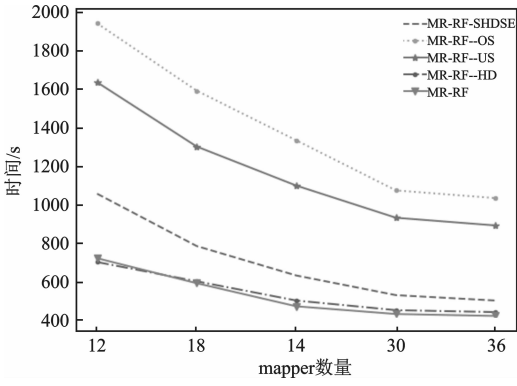


图 2 不同 mapper 数量下 PHD_ALL 数据集处理时间对比图

4 结束语

本文提出了一种 MapReduce 环境下处理多类别不平衡数据集的改进随机森林算法. 由于数据的不平衡性,为了避免由于随机采样而忽略少数类,采用了基于分层抽样的 bootstrap 方法;针对传统的 MR-RF 算法中使用 CART 决策树仍然以全局最优点作为划分点,使用了 HDDT 决策树以减少数据偏置给分类准确率带来的影响;利用 GMean 值和不合度值作为树的强度值和相似度值,提出了基于调和平均值的决策树选择集成方法. 最后通过公开的

数据集实验证明了该算法的有效性与可行性,具有现实意义. 在下一步的工作中,将本文提出的算法应用到真实的系统环境之中,以发挥该算法的作用.

参考文献:

[1] Fernández-Delgado M, Cernadas E, Barro S, et al. Do we need hundreds of classifiers to solve real world classification problems[J]. Journal of Machine Learning Research, 2014, 15(1): 3133-3181.

[2] Zhou Z H, Wu J, Tang W. Ensembling neural networks; many could be better than all[J]. Artificial intelligence, 2002, 137(1-2): 239-263.

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

[4] Hernández-Lobato D, Martínez-Muñoz G, Suárez A. Out of Bootstrap Estimation of Generalization Error Curves in Bagging Ensembles[M]// Intelligent Data Engineering and Automated Learning - IDEAL 2007. Springer Berlin Heidelberg, 2007:47-56.

[5] Liu T, Wang F, Agrawal G. Stratified Sampling for Data Mining on the Deep Web[J]. Frontiers of Computer Science, 2012, 6(2):179-196.

[6] Espíndola R P, Ebecken N F F. On extending f-measure and g-mean metrics to multi-class problems[J]. Data Mining VI Data Mining Text Mining &. Their business applications, 2014.

[7] 周志华. 机器学习[M]. 北京:清华大学出版社,2016.

[8] Herrera F. On the use of MapReduce for imbalanced big data using Random Forest[J]. Information Sciences An International Journal, 2014, 285 (C): 112-137.

作者简介:

姚立 男,(1991-),硕士研究生. 研究方向为数据挖掘、机器学习. E-mail:yl_jnu@126.com.

张曦煌 男,(1962-),博士,教授. 研究方向为嵌入式系统、数据挖掘、网络协议、图形与图像处理.