基于层次聚类的不平衡数据加权过采样方法

夏 英 李刘杰 张 旭 裴海英

(重庆邮电大学计算机科学与技术学院 重庆 400065)

摘 要 不平衡数据对传统分类算法的性能有一定影响,使得少数类的识别率降低。过采样是处理不平衡数据集的常用方法之一,其主要思想是通过增加少数类样本,使得少数类与多数类的数量能够在一定程度上达到平衡,但现有的过采样方法存在合成重叠样本以及过拟合的问题。文中提出一种基于层次聚类的不平衡数据加权过采样方法WOHC(Weighted Oversampling method based on Hierarchical Clustering)。该方法首先使用层次聚类算法对少数类进行聚类,将少数类样本划分为多个类簇,然后计算出类簇的密度因子来确定各类簇的采样倍率,最后根据每个类簇中样本与多数类边界的距离确定采样权重。利用该方法采样并结合C4.5算法在多个数据集上进行分类实验,结果表明使用该方法能够使分类算法在F-measure和G-mean指标上分别提升7.6%和5.8%,体现了该方法的有效性。

关键词 不平衡数据,层次聚类,过采样,重叠样本

中图法分类号 TP301 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2019.04.004

Weighted Oversampling Method Based on Hierarchical Clustering for Unbalanced Data

XIA Ying LI Liu-jie ZHANG XU BAE Hae-young

(School of Computer Science and Technology, Chongqing University of Posts and Telecommunications, Chongqing 400065, China)

Abstract Imbalanced data affect the performance of traditional classification algorithms to some extent, leading to a lower recognition rate for minority classes. Oversampling is one of the common methods for processing Imbalanced datasets. Its main idea is to increase the number of minority class samples so that the number of minority classes and majority classes can be balanced to a certain extent. Existing oversampling methods have problems of synthesis of overlapping samples and overfitting. This paper proposed a weighted oversampling method based on hierarchical clustering for Imbalanced data, named WOHC. It uses hierarchical clustering algorithm to divide the minority class samples into several clusters first, then it calculates the clusters' density factors to determine the sampling rate of each cluster, and finally determines the sampling weights according to the distance between the minority classes and the boundary of majority classes. In the experiments, WOHC method is adopted for oversampling and C4.5 algorithm is combined to perform the classification experiment on several datasets. Results show that the proposed method can improve the performance of algorithm by 7.6% and 5.8% on F-measure and G-mean respectively, which indicates the effectiveness of the method.

Keywords Imbalanced data, Hierarchical clustering, Oversampling, Overlapping sample

1 引言

不平衡数据集是指数据集中的某一类样本数量远远少于其他类的样本数量,其中数量占多数的类称为多数类,而占少数的类称为少数类。在现实生活中有很多不平衡数据的分类应用场景,如软件缺陷预测[1]、交通事故判定[2]、医疗诊断[3]、文本分类[4]、信用卡诈骗预测[5]等。在这些领域中,人们往往更加关心的是少数类数据,但传统分类算法适合于各类样本数量相对平衡的数据集,当处理不平衡数据时,传统分类算法可能会由于考虑整体分类准确率而忽视少数类,导致少数类样本的识别率下降。对于不平衡数据分类问题,可以从算法和数据采样两个层面寻找解决方案。其中,算法层面主要是通过集成学习[6]和代价敏感学习[7]等方法提高分类性能;数据采样层面主要是采用过采样(即增加少数类样本)和欠采样(即减少多数类样本)的方法来改善不同类样本数量的不平衡性。相比之下,数据采样层面的方法更有利于提升模型的泛化能力,而且过采样方法由于不会丢失样本信息而更具优势。但是,过采样方法存在合成重叠样本和过拟合问题。本文针对这一问题,提出一种基于层次聚类的加权过采样方法,以便有效识别少数类进而提升少数类的分类效果。

2 问题的提出及解决思路

目前已有很多学者针对过采样算法展开研究并取得了一系列成果。其中,SMOTE[8]算法是经典的过采样算法之一,

到稿日期:2018-09-23 返修日期:2018-12-21 本文受国家自然科学基金(41571401)资助。

夏 英(1972-),女,博士,教授,主要研究方向为数据库与数据挖掘;**李刘杰**(1993-),男,硕士生,主要研究方向为数据挖掘,E-mail:S160201032@stu.cqupt.edu.cn(通信作者);**张 旭**(1981-),男,博士,副教授,主要研究方向为数据挖掘、大数据分析;**裴海英**(1948-),男,博士,教授,主要研究方向为数据库、空间信息处理。

在此基础上扩展了一系列算法。SMOTE 算法首先在少数类样本中选择一个种子样本,然后再选择 K 近邻个同类样本中的一个,最终在这两者之间采用线性插值的方式产生合成样本,合成方法如式(1)所示:

$$s=a+\beta\cdot(a-b) \tag{1}$$

其中,a 是种子样本,b 是 K 近邻的同类样本,β 是介于 0 到 1 之间的随机数。SMOTE 能够增加少数类样本的数量,从而改善数据样本的不平衡性,但是这样的样本合成方式没有考虑到少数类内部数据分布不集中的情况[9]。如图 1 所示,三角形表示少数类样本,分布在圆形所表示的多数类样本的两侧。利用 SMOTE 合成样本,选择阴影三角形作为种子样本。假设 K 为 5,由于种子样本周边的同类样本数不足 5,因此会将分布在多数样本另一侧的同类样本视为邻近样本,合成样本时便会产生实心三角形所示的重叠样本。分类器在训练过程中学习到重叠样本所在区域的属性信息时,由于其所在区域的样本主要为多数类,可能导致在最终分类时将其错误地划分为多数类对象,并没有发挥出合成样本的作用。

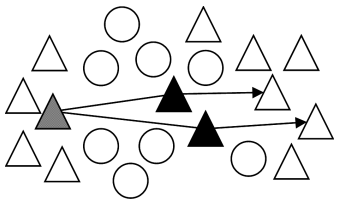

图 1　合成样本重叠实例

Fig. 1　Instance of synthesize sample overlapping

Safe-level SMOTE[10]方法会检测每个少数类样本邻近的同类样本的数量,并据此为每个少数类样本设置安全等级,只对安全等级高于一定阈值的样本进行采样。这样虽然避免了噪音样本的生成,但可能导致合成样本过于集中在少数类内部而出现过拟合现象。DS-SMOTE[11]在采样时,区分样本的稀疏密集区域,针对少数类稀疏区域进行采样,避免过拟合现象的发生。Cluster-SMOTE[12]首先使用 K-Means 方法将少数类划分为若干个类簇,然后再分别对这些类簇进行过采样,避免了重叠样本的生成。DB-SMOTE[13]以少数类样本距类中心的距离为标准进行采样,选择边界样本作为种子样本以避免过拟合情况的发生,但合成样本时仍有可能产生重叠样本。

针对上述过采样方法中存在的合成重叠样本和过拟合问题,本文提出一种基于层次聚类的加权过采样方法 WOHC。一方面,该方法通过层次聚类对少数类进行聚类,在聚类过程中考虑多数类的空间分布,防止形成的类簇中包含多数类样本;在生成的类簇中进行样本合成,避免重叠样本的产生。另一方面,为了防止过拟合,该方法对每一个类簇的密集程度进行判断,避免在过于密集的类簇中合成样本,并根据少数类样本与多数类边界样本的距离赋予少数类样本不同的采样权重,靠近多数类边界的样本采样权重高,远离多数类边界的样本采样权重低,从而增加少数类边界样本的数量,有利于分类器识别少数类样本的分类边界。

3　基于层次聚类的加权过采样处理步骤

3.1　少数类簇发现

从原始不平衡数据中获得少数类簇并确定其在特征空间中的分布是进行过采样处理的基础。WOHC 采用层次聚类[14]对少数类样本进行处理,确定少数类的特征空间分布。传统的层次聚类算法首先将数据集中的每一个样本看作一个类簇,接着合并距离最近的两个类簇,形成一个新的类簇,将总的类簇数目减一,然后继续合并最近的两个类簇,直到满足某个终止条件为止。终止条件一般为提前设定好的类簇数,或者距离阈值,即当类簇数达到一定的值,或者最近的两个类簇之间的距离超过某个阈值时,聚类终止。

为了避免重叠样本的生成,利用层次聚类对少数类进行聚类,除了要考虑两个少数类簇之间的距离,还要考虑多数类样本的分布。如果多数类样本与这两个少数类簇的距离均小于这两个少数类簇之间的距离,则表明合并后的少数类簇合成样本时会产生重叠样本,对应的少数类簇不应被合并。

在对少数类进行聚类前,先对多数类进行层次聚类,得到多数类簇集合 C^{maj}。利用 C^{maj} 中的类簇代表多数类,判断多数类与少数类之间的距离。基于层次聚类的少数类簇发现算法 MDH(Minority cluster discovery based on hierarchical clustering)的描述如算法 1 所示。

算法 1　基于层次聚类的少数类簇发现算法 MDH

输入:多数类簇集合 $C^{maj}=\{C_1^{maj},C_2^{maj},\cdots,C_n^{maj}\}$;少数类样本集 $D_{min}=\{x_{min}^1,x_{min}^2,x_{min}^m\}$;距离阈值 T

输出:少数类簇 $C^{min}=\{C_1^{min},C_2^{min},\cdots,C_N^{min}\}$

1. D=0;//初始化类簇间最小值
2. $C^{min}=D_{min}$;//初始化少数类簇集合 C^{min}
3. While D< T
4.　flag=1;//初始化类簇合并标志
5.　Mat=d(C^{min}) //计算类簇间距离矩阵
6.　D=Mat(i^*,j^*) //求得类簇间的最小距离及其对应类簇编号 i^* 与 j^*
7.　for p=1,2,…,n do //每个多数类簇
8.　if(d(C_p^{maj},$C_{i^*}^{min}$)<D && d(C_p^{maj},$C_{j^*}^{min}$)<D)
9.　flag=0;
10.　Unreach($C_{i^*}^{min}$,$C_{j^*}^{min}$)//设置两类簇不可达
11.　end if
12.　end for
13.　if flag ==1
14.　$C_{i^*}^{min}=C_{i^*}^{min}\cup C_{j^*}^{min}$;
15.　$C_{j^*}^{min}=\varnothing$;
16.　end if
17. end while

C^{min} 表示类簇集合。步骤 2 将少数类中的每个样本初始化为一个类簇。步骤 3 表明当类簇距离 D 小于距离阈值 T 时聚类继续进行。步骤 4 对合并类簇的标志进行初始化。步骤 5－6 首先计算类簇间的距离,并选择类簇间的最小距离 $M(i^*,j^*)$ 赋值给 D。步骤 8 通过距离判断检测少数类 $C_{i^*}^{min}$ 与

C_j^{min}之间是否存在多数类，步骤 13－15 进行类簇的合并。

距离阈值 T 是进行类簇合并的关键条件，为了更好地估计距离阈值，定义距离阈值如下：

$$T=d_{mean}*f \tag{2}$$

设 d_m^i 表示某个少数类样本与其他同类样本距离从小到大排序的中位距离，而 d_{mean} 表示这些中位距离的平均值，选择中位距离的平均值作为基准值可以避免噪声样本的干扰。f 是距离调节因子，可对距离阈值进行调节，具体取值范围将在实验结果与分析中进行讨论。

3.2 基于密度因子的采样倍率

通过层次聚类获得了若干少数类簇，由于每个类簇的样本数和特征空间范围不尽相同，因此每一个类簇的密集程度也不尽一致。采用 SMOTE 算法在密集的类簇中合成样本时，由于种子样本与邻近样本之间的距离较近，导致合成样本与原始样本的相似性较高，虽然增加了少数类样本的数量，但并没有增加样本的多样性，仍有可能造成过拟合现象。因此针对每个少数类簇，根据其密集程度来确定采样倍率，对密集的类簇设置较低的采样倍率，对稀疏的类簇设置较高的采样倍率，这样有利于避免过拟合现象。

类簇的密集程度可以通过类簇中样本之间的距离和样本的数量来度量。对于第 i 个类簇，首先计算类簇中两两样本的距离，再计算这些距离的平均值 $dist_i$，并结合类簇样本数量确定类簇的密度因子 den_i，如式(3)所示：

$$den_i \frac{n_i}{dist_i}(i=\{1,2\cdots,N\}) \tag{3}$$

其中，n_i 表示第 i 个类簇的样本数量。每个类簇的采样倍率与该类簇的密度因子成反比，进行归一化后，可得到每个类簇的采样倍率 W_i，如式(4)所示：

$$W_i=\frac{1/den_i}{\sum_{i=1}^{N}1/den_i} \tag{4}$$

其中，N 表示少数类簇个数。最终根据总的多数类与总的少数类的数量差值 num_{need} 以及各类簇的采样倍率 W_i，来确定每个类簇的采样数量 num_i。

$$num_i=num_{need}*W_i \tag{5}$$

3.3 采样权重计算

每个样本的采样权重是依据该样本与多数类边界的距离所确定的。距离近的样本的采样权重高，距离远的样本的采样权重低，以有效扩展少数类决策边界。相较于非边界多数类样本，边界多数类样本与少数类样本之间的距离更近，因此可以根据与少数类样本的邻近关系确定多数类边界。记某个少数类簇中第 q 个样本点为 α_q，其邻近的 M 个多数类样本的集合 $N_{maj}(\alpha_q)$，该类簇的每个少数类样本都有一个邻近多数类样本集合 $N_{maj}(\alpha_q)$，但每一个少数类样本的邻近集合都只是多数类边界的一部分，需要通过合并这些多数类样本集合为 $N_{maj}(\alpha_q)$来形成多数类边界。因此取这些邻近多数类样本集合的并集，记为 B_{maj}，表示对应少数类簇的多数类边界。

确定多数类边界后，通过少数类簇中样本到多数类边界中各点的距离关系来分配权重。少数类簇样本 α_q 与多数类边界各点 $b\in B_{maj}$ 的距离总和表示为 td_q，而类簇中样本点的权重与距离成反比，因此类簇中各样本的采样权重可由式(6)计算得到：

$$P_q=\frac{1/td_q}{\sum 1/td_q} \tag{6}$$

其中，P_q 表示某个类簇中的第 q 个少数类样本的采样权重。

4 基于层次聚类的加权过采样算法(WOHC)综述与分析

4.1 WOHC 算法

基于层次聚类的加权过采样算法首先对少数类样本进行聚类来确定其分布情况，然后根据各类簇的密度因子确定每个类簇的采样倍率，接着计算出少数类与多数类边界的距离来分配采样权重，最终结合采样倍率和采样权重对每一个少数类簇使用 SMOTE 算法进行过采样。具体如算法 2 所示。

算法 2 基于层次聚类的加权过采样算法(WOHC)

输入：少数类样本集合 $D_{min}=\{x_{min}^1,x_{min}^2,\cdots,x_{min}^m\}$；多数类簇集合 C^{maj}；距离阈值 T；少数类邻近多数类个数阈值 M

输出：合成少数类样本与原始样本的数据集合 sample

```
1. den=Ø;//密度因子集合初始化
2. td=Ø;//与边界样本的距离集合初始化
3. sample=Ø;//合成样本集合初始化
4. C^min=MDH(C^maj,D_min, T)//调用少数类发现算法获得少数类簇集合 C^min
5. for i=1,2,…,N do//对少数类簇集合的每个类簇
6.     dist_i=avg(C_i^min) //求解类簇样本间的平均欧拉距离
7.     den_i=n_i/dist_i//利用式(3)求得密度因子
8.     den=den ∪den_i//添加当前密度因子到 den
9. end for
10. W=Norm(den) //利用式(4)归一化得到采样倍率集合 W
11. for i =1,2,…,N do //对少数类簇集合里的每个类簇
12.    B_maj=NM(C_i^min) //获得类簇多数类边界集合
13.    for a_q∈C_i^min//类簇的每个样本
14.       td_q=sum(a_q,B_maj) //计算与边界集合的距离
15.          td=td∪ td_q//添加当前边界样本距离到 td
16.    end for
17.    P=Norm(td) //利用式(6)归一化得到采样权重集合 P
18.    num_i=num_need * W_i//利用式(5)确定每个类簇的采样数量
19.    sample_i=Synt(SMOTE,P,num_i) //利用 SMOTE 进行样本合成
20.    sample=sample∪sample_i
21. end for
```

步骤 1－3 分别对密度因子集合、少数类样本与对应多数类边界距离集合、合成样本集合进行初始化。步骤 4 调用少数类簇发现算法，获得少数类簇 C^{min}。步骤 5－10 首先利用式(3)计算类簇间的平均距离并求得了对应的密度因子 den_i，利用式(4)归一化后获得了采样倍率集合 W。步骤 12 获得了对应类簇的少数类边界样本集合 B_{maj}^i。步骤 14 计算出各少数类与边界样本的距离之和 td_q 后，利用式(6)进行归一化，得到该类簇中各样本的采样权重集合 P。在步骤 18－20 利用采样倍率与采样权重结合 SMOTE 算法进行样本合成。

4.2 WOHC 算法分析

少数类簇发现算法使用层次聚类，聚类中需要计算少数类簇之间的距离和多数类与少数类簇之间的距离。假设多数类簇数目为 N_{maj}，当前少数类簇数目为 N_{min}。每个少数类簇都需要与其他的少数类簇计算距离，因此计算少数类簇两两距离的时间复杂度为 $O(N_{min}^2)$，将两两距离以矩阵存储，之

后从距离矩阵中找到最小距离对应的两个少数类簇,每个多数类簇都需要与这两个类簇计算距离,共有 N_{maj} 个多数类簇,计算 N_{maj} 次,因此得到多数类簇与少数类簇之间的距离矩阵的时间复杂度为 $O(N_{maj})$。然后根据类簇之间的距离关系确定是否合并,更新少数类簇个数 N_{min},执行下一次合并,可能进行的合并次数为 N_{maj},因此该步骤的时间复杂度为 $O(N_{min}(N_{maj}+N_{min}^2))$。

采样倍率的设置依赖于类簇的密度因子,确定密度因子时需要计算类簇中两两样本之间的距离。假设第 i 个类簇的样本数量为 n_i,则获得两两距离的时间复杂度为 $O(n_i^2)$,因此该步骤的时间复杂度为 $O(n_i^2)$。

设置采样权重时需要计算每个类簇中的样本与多数类之间的距离来确定多数类边界。第 i 个少数类簇的样本数量为 n_i,多数类样本的数量为 n_{maj},则对于第 i 个类簇需要完成 $n_i * n_{maj}$ 次运算,因此该步骤的时间复杂度为 $O(n_i * n_{maj})$。

从上述 3 个步骤的时间复杂度分析可知,WOHC 算法的时间复杂度应由时间复杂度最大的部分表示,因此最终的时间复杂度为 $O(N_{min}(N_{maj}+N_{min}^2))$。而对于空间复杂度,由于均采用矩阵存储样本或者类簇之间的距离,因此空间复杂度为 $O(n_{min}^2+n_{maj}^2)$,其中 n_{min} 和 n_{maj} 分别表示少数类和多数类样本的数量。

5　实验结果与分析

5.1　评价指标

传统分类算法常采用混淆矩阵进行性能分析评价,但不平衡分类算法的性能通常采用 F-measure[15] 与 G-mean[16] 这两个指标。F-measure 是准确率和召回率的调和平均值,其值接近两数的较小者。在不平衡数据分类中,只有当准确率和召回率的值均较大时,F-measure 才会增大。G-mean 表示对数据正类与负类的召回率的几何平均值(将少数类视为正类,多数类视为负类),可以同时考量分类器对两类数据的分类效果。准确率(Precision)和召回率(Recall)可以在如表 1 所列的混淆矩基础上通过式(7)和式(8)分别得到。F-measure 通过式(9)计算得到,其中调节参数 β 表示两个参数的重要性,一般取值为 1。G-mean 可以直接由混淆矩阵得出,如式(10)所示。

表 1　混淆矩阵

Table 1　Confusion matrix

	预测正类	预测负类
实际正类	TP	FN
实际负类	FP	TN

$$Precision=\frac{TP}{TP+FP} \tag{7}$$

$$Recall=\frac{TP}{TP+FN} \tag{8}$$

$$F\text{-}measure=\frac{(1+\beta^2)*Recall*Precision}{\beta^2*Recall+Precision} \tag{9}$$

$$G\text{-}mean=\sqrt{\frac{TP}{TP+FN}\times\frac{TN}{TN+FP}} \tag{10}$$

5.2　实验数据集

实验数据取自 UCI 数据集[17],表 2 列出了数据集的特征数、各类样本数量以及不平衡度。本文重点讨论二分类情况,由于 wine 数据集中存在 3 类样本,因此将前两类样本视为同一类样本,给予相同的类别标签,第 3 类样本单独作为一类样本。ecoli 数据集将其中的类别标签为“pp”的样本作为少数类,其余的作为多数类;vehicle 数据集将其中标签为“van”的样本作为少数类,剩余的作为多数类。

表 2　数据集

Table 2　Datasets

数据集	特征数	样本总数	不平衡度
pima	8	768	1∶1.87
blood	4	748	1∶3.20
haberman	3	306	1∶2.78
ionoshpere	35	351	1∶1.78
breast	10	699	1∶1.90
wine	13	178	1∶1.83
vehicle	17	846	1∶3.25
ecoli	7	336	1∶4.98
yeast	8	1484	1∶9.10

5.3　实验结果分析

实验为 Matlab 环境,将 WOHC,SMOTE,Safe-level SMOTE 和 DB-SMOTE 4 种采样算法分别结合稳定的分类算法 C4.5[18] 进行实验对比;为了验证采样的有效性,以 C4.5 未结合任何采样方法的分类结果作为对比基准线。采用 10 折交叉验证作为评估方法,即将每份数据集等分为 10 份,依次选择其中的一份作为验证集,剩余的 9 份作为训练集,求 10 次结果的平均值。实验结果如表 3 和表 4 所列,其分别列出了各算法在不同数据集上的 *F-measure* 和 *G-mean* 的值,表格加粗项表示性能第一的分类方法,并以柱状图在图 2 和图 3 中进行了直观展示。

表 3　各算法在不同数据集上的 *F-measure* 值对比

Table 3　Comparison of *F-measure* values of each algorithm on different datasets

数据集	C4.5	SMOTE+C4.5	Safe-level SMOTE+C4.5	DB-SMOTE+C4.5	WOHC+C4.5
pima	0.5996	0.5586	0.5695	0.5888	**0.6001**
blood	0.3839	**0.4127**	0.4011	0.3968	0.4075
haberman	0.3228	0.3776	0.3694	0.3863	**0.4071**
ionoshpere	0.7353	0.7414	0.7699	**0.7756**	0.7562
breast	0.7076	0.7063	0.6968	0.6854	**0.7196**
wine	0.9021	0.9326	0.9599	0.9204	**0.9785**
vehicle	0.9234	0.9251	0.9266	0.9179	**0.9353**
ecoli	0.7299	0.7246	**0.7468**	0.7139	0.7352
yeast	0.3758	0.434	0.4752	0.4523	**0.4838**

表 4　各算法在不同数据集上的 *G-mean* 值对比

Table 4　Comparison of *G-mean* values of each algorithm on different datasets

数据集	C4.5	SMOTE+C4.5	Safe-level SMOTE+C4.5	DB-SMOTE+C4.5	WOHC+C4.5
pima	0.6605	0.6519	0.6584	0.6852	**0.6855**
blood	0.5617	**0.5918**	0.5531	0.5796	0.5827
haberman	0.4860	0.5436	0.5356	0.5481	**0.5682**
ionoshpere	0.7835	0.7963	0.8196	**0.8230**	0.8086
breast	0.7627	0.7520	0.754	0.748	**0.7650**
wine	0.9199	0.9531	0.9604	0.9558	**0.9810**
vehicle	0.9635	0.9637	0.9635	0.9619	**0.9657**
ecoli	0.9077	0.9018	**0.9117**	0.9011	0.9018
yeast	0.5442	0.6084	0.6429	0.6351	**0.6757**

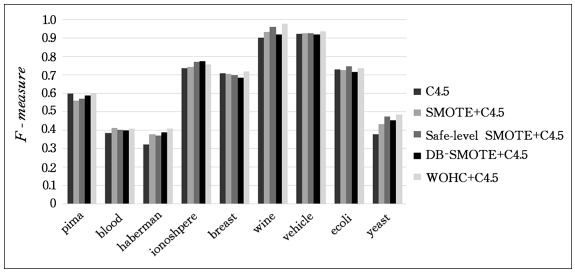

图 2 各种分类方法的 *F-measure* 值对比

Fig. 2 Comparison of *F-measure* values of each classification method

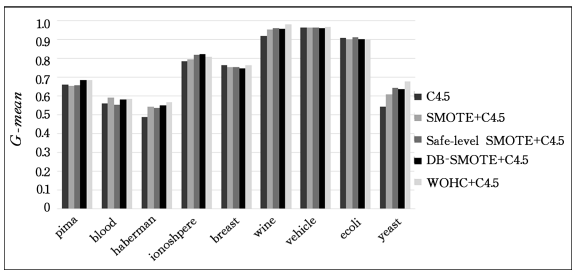

图 3 各种分类方法的 *G-mean* 值对比

Fig. 3 Comparison of *G-mean* values of each classification method

从图 2 和图 3 中可以看出，采样方法结合 C4.5 在绝大多数情况下的结果均优于只使用 C4.5 的分类结果，无论是 *F-measure* 值还是 *G-mean* 值。而在 *F-measure* 值上，WOHC 算法在 9 个数据集上获得了 6 次第一，在 *G-mean* 值上其同样获得 6 次第一，均领先于其他算法，这表明该算法能够提升对少数类数据的分类效果。相比 SMOTE 和 Safe-level SMOTE 等算法，WOHC 算法的性能有较大提升，这主要是因为 WOHC 算法避免了过拟合现象的出现。DB-SMOTE 考虑了通过在边界样本上进行样本合成，来扩大少数类决策边界；但 WOHC 不仅考虑到了边界样本的重要性，同时也考虑了少数类的分布情况，减少了重叠样本的生成，避免了噪音的干扰，因此相比于 DB-SMOTE 有了更进一步的提升。而对于不平衡度较高的数据集如 yeast，WOHC 算法仍有不错的表现，这说明其对于各种不平衡度的数据集具有良好的适应性。

5.4 实验参数选择

WOHC 算法的性能一定程度上受到实验参数的影响。其中主要涉及到的参数包括少数类簇发现算法中的距离调节因子 f，以及确定多数类边界的邻近多数类个数阈值 M。

距离调节因子 f 用于控制聚类中的类簇合并。该值太小会导致少数类簇数量过多，而每个类簇中样本数量较少，使得合成样本的多样性降低，造成过拟合；而该值过大则会使得合并的类簇中包含多数类样本，导致合成样本时重叠样本的出现。

对于邻近多数类个数阈值 M，其取值不应过大，因为过大的取值会使得处于内部的非边界多数类样本也作为边界样本的一部分，使得边界样本对少数类样本权重分配的灵敏性降低，导致少数类之间的采样权重差异减小。

为了确定 M 和 f 的最佳取值范围，以 wine，ecoli，haberman 为测试数据集。对于 M，以 2 为初始值，步长为 1，进行了 5 组实验，实验结果如表 5 所列。其中，当 M 为 5 时 wine 的 *G-mean* 取得最大值，而 ecoli 与 haberman 则是在 M 分别取 3 与 4 时取得最大 *G-mean* 值，由此可知 3～5 这一区间为 M 较为合适的取值范围。将表 5 转化为折线图的形式，如图 4 所示。对于 f，以 0.2 为初始值，步长为 0.2，进行了 5 组实验，实验结果如表 6 所列。其中当 f 取 0.6 时 wine 取得 *G-mean* 的最大值，同理 ecoli 与 haberman 分别在 f 取 0.8 与 1 时取得 *G-mean* 的最大值，0.6～1 为 f 的参考取值范围，最终将结果以折线图的形式展示，如图 5 所示。

表 5 不同 M 值下的 *G-mean* 值

Table 5 *G-mean* values at different M

M	wine	ecoli	haberman
2	0.9412	0.9314	0.5047
3	0.9554	**0.9429**	0.5373
4	0.9772	0.9287	**0.5590**
5	**0.9857**	0.9102	0.4928
6	0.9336	0.9166	0.5497
7	0.9421	0.9014	0.5239

表 6 不同 f 值下的 *G-mean* 值

Table 6 *G-mean* values at different f

f	wine	ecoli	haberman
0.2	0.9521	0.9368	0.5167
0.4	0.9601	0.9355	0.5777
0.6	**0.9757**	0.9456	0.5749
0.8	0.9695	**0.9578**	0.5419
1.0	0.9683	0.9245	**0.6114**
1.2	0.9378	0.9359	0.5973

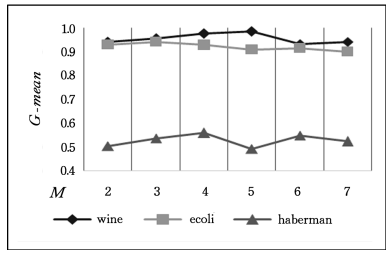

图 4 不同 M 值下的 *G-mean* 值

Fig. 4 *G-mean* values at different M

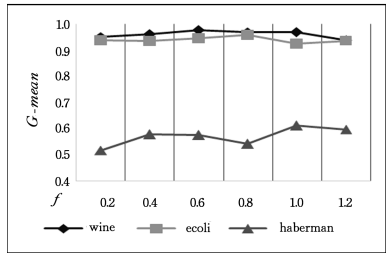

图 5 不同 f 值下的 *G-mean* 值

Fig. 5 *G-mean* values at different f

结束语 过采样是进行不平衡数据集分类的有效方法之一。针对过采样方法中存在的样本重叠和过拟合等问题，本文提出一种基于层次聚类的加权过采样方法 WOHC。该方法通过聚类确定少数类的样本分布，避免重叠样本的产生，同时根据每个少数类簇的密集程度确定其采样倍率，最后通过计算少数类与邻近多数类之间的距离来确定采样权重，从而拓展样本的边界区域。实验验证了该算法在 F-measure 和

G-mean 指标上比同类方法具有更好的表现,能够有效提升不平衡数据的分类效果,但其中关于参数的选择,需要通过多次实验进行确定,因此如何根据数据集自适应地确定参数的选择是未来的一个研究方向。

参 考 文 献

[1] MALHOTRA R,KHANNA M. An empirical study for software change prediction using imbalanced data[J]. Empirical Software Engineering,2017,22(6):1-46.

[2] JEONG H,JANG Y,BOWMAN P J,et al. Classification of motor vehicle crash injury severity: A hybrid approach for imbalanced data[J]. Accident Analysis & Prevention,2018,120:250-261.

[3] JIANG J ,LIU X ,ZHANG K ,et al. Automatic diagnosis of imbalanced ophthalmic images using a cost-sensitive deep convolutional neural network[J]. BioMedical Engineering OnLine,2017,16(1):132.

[4] LI Y,GUO H,ZHANG Q,et al. Imbalanced text sentiment classification using universal and domain-specific knowledge[J]. Knowledge-Based Systems,2018,160:1-15.

[5] DAL P A . Learned lessons in credit card fraud detection from a practitioner perspective[J]. Expert Systems with Applications,2014,41(10):4915-4928.

[6] TANG B,HE H. GIR-based Ensemble Sampling Approaches for Imbalanced Learning[J]. Pattern Recognition,2017,71:306-319.

[7] BIAN J,PENG X G,WANG Y,et al. An Efficient Cost-Sensitive Feature Selection Using Chaos Genetic Algorithm for Class Imbalance Problem[J]. Mathematical Problems in Engineering,2016,2016(6):1-9.

[8] CHAWLA N V,BOWYER K W,HALL L O,et al. SMOTE: synthetic minority over-sampling technique[J]. Journal of Artificial Intelligence Research,2002,16(1):321-357.

[9] HE H,GARCIA E A. Learning from Imbalanced Data[J]. IEEE Transactions on Knowledge & Data Engineering,2009,21(9):1263-1284.

[10] BUNKHUMPORNPAT C,SINAPIROMSARAN K,LURSINSAP C. Safe-Level-SMOTE: Safe-Level-Synthetic Minority Over-Sampling Technique for Handling the Class Imbalanced Problem[C]// Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining. Springer-Verlag,2009:475-482.

[11] WANG J H,DUAN B Q. Research on a density based SMOTE method[J]. CAAI Transactions on Intelligent Systems,2017(6):865-872. (in Chinese)
王俊红,段冰倩. 一种基于密度的 SMOTE 方法研究[J]. 智能系统学报,2017(6):865-872.

[12] CIESLAK D A,CHAWLA N V,STRIEGEL A. Combating imbalance in network intrusion datasets[C]// IEEE International Conference on Granular Computing. IEEE,2006:732-737.

[13] LIU Y X,LIU S M,LIU T,et al. A new oversampling algorithm DB-SMOTE[J]. Computer Engineering and Applications,2014,50(6):92-95. (in Chinese)
刘余霞,刘三民,刘涛,等. 一种新的过采样算法 DB-SMOTE[J]. 计算机工程与应用,2014,50(6):92-95.

[14] VOORHEES E M. Implementing agglomerative hierarchic clustering algorithms for use in document retrieval[J]. Information Processing & Management,1986,22(6):465-476.

[15] CHEN S,GUO G D,CHEN L F. Unbalanced data classification method based on clustering fusion[j]. Pattern Recognition and Artificial Intelligence,2010,23(6):772-780. (in Chinese)
陈思,郭躬德,陈黎飞. 基于聚类融合的不平衡数据分类方法[J]. 模式识别与人工智能,2010,23(6):772-780.

[16] MATHEW J,PANG C K,LUO M,et al. Classification of Imbalanced Data by Oversampling in Kernel Space of Support Vector Machines[J]. IEEE Transactions on Neural Networks and Learning Systems,2018,29(9):4065-4076.

[17] UCI Machine Learning Repository [EB/OL]. http://archive.ics.uci.edu/ml/index.php.

[18] BOMBARA G,VASILE C I,PENEDO F,et al. A Decision Tree Approach to Data Classification using Signal Temporal Logic[C]// International Conference on Hybrid Systems: Computation and Control. ACM,2016:1-10.