计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.190600132

一种新的不均衡关联分类算法

崔 巍 贾晓琳 樊帅帅 朱晓燕
西安交通大学计算机科学与技术学院 西安 710049
(wwwcuiwei@stu.xjtu.edu.cn)

摘 要 基于规则的分类算法具有分类性能好、可解释性强的优点,得到了广泛的应用。然而已有的基于规则的分类算法没有考虑不均衡数据的情况,从而影响了其对不均衡数据的分类效果。文中提出了一种新的不均衡关联分类算法 ACI。首先生成所有的关联规则,然后使用不均衡规则裁剪方法进行规则裁剪。最后,将剩余规则存储到 CR 树中,用于新实例的分类。在 27 个公开数据集上的实验结果表明,提出的不均衡关联分类算法在不均衡数据集上比基准算法的分类效果更好。

关键词:分类;关联规则;不均衡数据
中图法分类号 TP312

New Associative Classification Algorithm for Imbalanced Data

CUI Wei, JIA Xiao-lin, FAN Shuai-shuai and ZHU Xiao-yan
School of Computer Science and Technology, Xi'an Jiaotong University, Xi'an 710049, China

Abstract The rule-based classification algorithms, which have good classification performance and interpretability, have been widely used. However, the existing rule-based classification algorithms do not consider the case of imbalanced data, thus affect their classification effect on imbalanced data. In this paper, a new associative classification algorithm ACI for imbalanced data is proposed. Firstly, all the association rules are generated. Then, the rules are pruned by an imbalanced rule pruning method. Finally, the remaining rules are saved in a CR Tree for new instance classification. Experimental results on 27 public data sets show that the proposed algorithm performs better than the compared algorithms.

Keywords Classification, Association rule, Imbalanced data

1 引言

基于规则的分类算法具有分类效能好、可解释性强等优点,得到了广泛的关注和应用。常见的基于规则的分类算法有 ID3[1],C4.5[2],Ripper[3-4]等。另外,还有多种关联分类算法被提出,包括 CBA[5],CMAR[6],MMAC[7],ACWV[8]等。关联分类算法将关联规则挖掘和分类结合在一起,从数据集中挖掘出大量的用于分类的规则,并提取出最有效的规则,构建高效的分类器。这些关联分类算法在处理均衡数据集时表现出了相对其他分类算法的优势。但在处理不均衡数据时,往往表现不佳,尤其是对少数类的预测性能较差。

现实世界中的分类数据大部分都是不均衡的[9],且少数类往往更重要,因此研究针对不均衡数据的分类算法就显得尤为重要。目前针对不均衡数据的分类算法有两种,一种是对数据进行预处理,将不均衡数据转换为均衡数据,常用的方法包括随机欠采样[10]和随机过采样[10]。随机欠采样通过随机减少多数类实例,使得类分布基本均衡。该方法丢弃了很多具有潜在价值的数据,影响了分类性能。随机过采样通过增加少数类实例,使得类分布基本均衡。该方法破坏了原始数据的分布,也会对分类性能产生影响。另一种方法是从算法角度进行改进,主要采用对样本设置权值[11]、调整分类边界[12]以及集成学习[13-14]等措施。通过改进算法可使算法更加重视少数类,从而降低对少数类的误分率。

本文从算法角度对在均衡数据集上表现优异的关联分类算法进行改进,提出了一种新的不均衡关联分类算法 ACI。ACI 包括 4 个主要步骤:规则生成、规则裁剪、规则存储和新实例分类。ACI 不仅保留了关联分类算法分类效果好、可解释性强的优点,同时能够更好地处理不均衡数据的分类。

2 不均衡关联分类算法

2.1 相关术语

设数据集 $D=\{d_i \mid i=1,2,\cdots,n\}$,$d_i=\langle a_j \mid j=1,2,\cdots,m\rangle$,其中 a_m 是实例 d_i 的类标签,一般用 cl 表示。

项:由属性及属性值组成的二元组,如$\langle a_1,1\rangle$,其中 a_1 是属性,1 是属性 a_1 的取值。

项集:由一个或多个项组成的集合,如$\langle a_1,1\rangle \wedge \langle a_2,1\rangle \wedge \langle a_4,1\rangle$。

规则:由两部分组成,左部为项集,右部为项,如:$\langle a_1,1\rangle \wedge \langle a_2,1\rangle \wedge \langle a_4,1\rangle \rightarrow \langle a_6,1\rangle$

关联分类规则:规则右部的项对应的属性为类标签 cl 的特殊规则,如:$\langle a_1,1\rangle \wedge \langle a_2,1\rangle \wedge \langle a_4,1\rangle \rightarrow \langle cl,1\rangle$

覆盖[15]:如果一个实例 d_i 满足规则 r 的限制,即 d_i 中属性的取值与规则 r 属性的取值一致,则称规则 r 覆盖实例 d_i。

支持度[16]:项集 I 的支持度是指与 I 中属性取值一致的

基金项目:国家自然科学基金(61402355,61502378)
This work was supported by the National Natural Science Foundation of China(61402355,61502378).
通信作者:朱晓燕(zhu.xy@xjtu.edu.cn)

实例所占的比例，规则 r 的支持度是指 r 覆盖的实例占总实例的比例。

置信度[16]：规则 r 的支持度与其左部的项集的支持度的比值。

2.2　算法描述

ACI 算法是在关联分类算法中引入不均衡数据处理，使得关联分类算法适用于不均衡数据的分类。ACI 算法主要由规则生成、规则裁剪、规则存储以及新实例的分类 4 部分组成，其框架如图 1 所示。

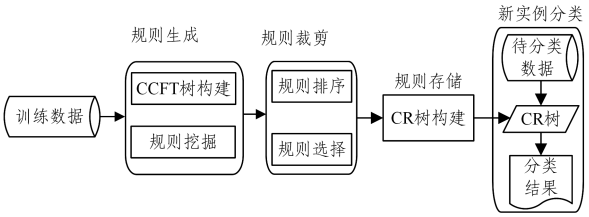

图 1　不均衡关联分类算法的框架图

Fig. 1　Diagram of imbalanced association classification algorithm

2.2.1　规则生成

给定训练数据集 D，最小支持度[17]阈值 $minsup$，最小置信度[17]阈值 $minconf$ 和最小卡方值[6]阈值 $minK$。ACI 在生成关联规则时，首先构造 CCFP 树[8]，然后使用 CCFP-Growth 算法[8]在 CCFP 树中挖掘关联规则。

1）CCFP 树构建

构建 CCFP 树时，首先构建项头表 H。遍历数据集 D，找出所有支持度大于 $minsup$ 的频繁项，并将频繁项按照支持度从大到小排列，存入 H。H 中的一条记录对应一个频繁项及其支持度。

然后，创建一棵只包含一个空的根节点 R 的树 T。再将 D 中的实例依次插入到 T 中。插入一条实例 d_i 时，从 R 开始，将 R 设为当前节点 Current，依次遍历 H 中的每个频繁项。

对于当前的频繁单项集 I_p，如果 d_i 不包含 I_p，则接着遍历下一个频繁项 I_q。如果 d_i 包含 I_q，且 Current 的子节点 Node 中包含 I_q，则将该子节点 Node 设置为当前节点 Current，然后继续遍历 H，处理下一个频繁项。如果 Current 的子节点中不包含 I_q，则需要为 I_q 重新构建一个节点 N 作为 Current 的子节点，并将 N 设置为 Current。树 T 中的每个节点记录该节点对应的项，即从根节点 R 到该节点组成的项集对应的各个类标签出现的次数。依次进行，直到 H 遍历完，CCFP-Tree 则构建完成。在 CCFP 树中具有相同属性的实例可以共享前缀，在节省空间的同时，还可以方便后续的规则生成。

2）关联规则挖掘

CCFP 树构建好之后，需要使用 CCFP-Growth 算法在 CCFP 树中挖掘所有满足 $minsup$，$minconf$ 和 $minK$ 的分类关联规则。挖掘过程使用递归的方式进行。从叶节点开始，先挖掘所有包含叶节点的规则，然后将叶节点从 CCFP-Tree 中删除，接着递归地挖掘包含叶节点的规则，直至到达根节点。

2.2.2　规则裁剪

使用 CCFP-Growth 算法生成所有满足最小支持度、置信度和卡方值阈值限制的关联规则之后，就可以进行规则裁剪。在进行规则裁剪时，ACI 使用一种不均衡规则裁剪方法。该方法首先将生成的规则根据支持度、置信度和卡方值进行排序，得到排好序的规则集合 R。然后对 R 中的规则依次进行遍历和选择：对每一条规则 r，遍历数据集 D，将 r 覆盖的实例对应的规则覆盖条数加 1。如果 r 没有覆盖任何实例，则将 r 从 R 中删除。为了处理类不均衡问题，ACI 对不同的类设置不同的规则覆盖条数阈值 $limits$。当覆盖该实例的规则条数超过对应类的 $limit$ 时，将实例从 D 中删除。在同一条路径上的规则都满足短规则比长规则的支持度和置信度大，因此短规则排在长规则前面。

2.2.3　规则存储

规则裁剪之后，ACI 将剩余的规则存储到 CR 树[6]中。CR 树是一棵前缀树，具有相同属性的规则共享前缀。使用 CR 树存储规则不仅可以节约空间，还可以从 CR 树方便地得到规则。在同一条路径上的规则，长规则所覆盖的实例是短规则覆盖实例的子集。另外，从 CR 树可以很容易地得到规则的排序，CR 树的构建和 CCFP 树的构建过程类似，都需要使用项头表，并从空树开始将规则依次插入树中。

2.2.4　新实例的分类

对于新实例 d，根据所有覆盖新实例的关联规则来确定 d 的类标签。如果所有覆盖 d 的关联规则具有相同的类标签，则新实例的类标签就是这些关联规则的类标签。

如果覆盖 d 的关联规则具有不同的类标签，则根据类标签将关联规则分组，具有相同类标签的规则归为一组。分别计算每组规则的加权卡方值，将 d 分类为具有最大加权卡方值[6]的规则组对应的类标签。加权卡方值的具体计算方法如下：

$$\sum \frac{\chi^2\chi^2}{\max \chi^2} \tag{1}$$

其中，$\max \chi^2$ 是对每条规则 $r=X\rightarrow c$ 计算式(2)的值。

$$\max \chi^2 = \left(\min\{\sup(X),\sup(c)\} - \frac{\sup(X)\sup(c)}{|T|}\right)^2 |T|e \tag{2}$$

$$e=\frac{1}{\sup(X)\sup(c)}+\frac{1}{\sup(X)(|T|-\sup(c))}+\frac{1}{(|T|-\sup(X))\sup(c)}+\frac{1}{(|T|-\sup(X))(|T|-\sup(c))} \tag{3}$$

3　算法实现及复杂度分析

3.1　算法实现

ACI 算法的实现分为两部分，第一部分是模型构建，包括规则生成、规则裁剪和规则存储，构建的模型即存储最终裁剪之后关联规则的 CR 树；第二部分是基于 CR 树模型的新实例分类。算法 1 给出了模型构建的实现过程的伪代码。该过程的输入包括训练数据集 D，以及 4 个阈值限制 $minsup$，$minconf$，$minK$ 和 $limits$。其输出是一个存储了最终用于后续分类的关联规则的 CR 树。

算法 1　ACI 模型构建过程

输入：D 为训练数据集；minsup 为支持度阈值；minconf 为置信度阈值；mink 为卡方值阈值；limits 为最少规则条数

输出：CR-tree 为存储裁剪之后规则的树

```
1. T. H=HeaderTable(D,minsup)
2. R=Node(Null);
3. T. root=R
4. for each d ∈D do
5.      将 d 插入 T
6. end for
7. α=Null
```

```
8. RS=CCFPGrowth(T,minsup,minconf,mink,null)
9. Rules=RulePrune (RS,limits)
10. for each rule∈Rules
11.     将 rule 插入 CR-tree
12. end for
13. Return CR-tree
```

ACI 模型构建过程伪代码第 1－6 行是 CCFP 树的构建，其中第 1 行构建项头表 H，第 2 行和第 3 行创建一棵只包含一个空的根节点的树 T。第 4－6 行将训练集 D 中的实例依次插入 T。第 8 行是规则生成，第 9 行是规则裁剪，第 10－12 行是规则存储。

在 ACI 的模型构建过程中，有 3 个主要的函数：HeaderTable、CCFPGrowth 和 RulePrune，分别完成项头表构建、规则挖掘和规则裁剪。其中项头表构建相对比较简单，这里不给出其伪代码。

CCFFP-Growth 函数的伪代码如算法 2 所示。算法的输入为构建好的 CCFP 树，支持度、置信度和卡方值阈值，以及后缀项集。输出为挖掘得到的关联规则集合。

算法 2 CCFPGrowth 函数伪代码

```
输入：T；minsup；minconf；mink；α
输出：RS；规则集合
1. RS=Null
2. If T 只包含单个路径 P then
3.   for P 中节点的每个组合 do
4.     γ=β∪α
5.     for 每个类标签 cl
6.     产生一条规则 r=γ－>cl
7.     if r 满足 minsup，minconf and minK then
8.       RS=RS∪{r}
9.     end if
10.    else
11.    for T 的项头表 H 中的每个项 a_f do
12.      β=α_f∪α
13.      构造 β 的条件 CCFP 树 Tree_β；
14.      if Tree_β≠∅ then
15.      RS=RS ∪CCFPGrowth(Tree_β，minsup，minconf，minK，β)；
16.    endif
17.  endfor
18. endif
19. return RS
```

CCFPGrowth 函数第 1 行将规则集合 RS 初始化为空。当 CCFP 树只有一条路径时，第 2－10 行挖掘出单路径上的满足条件的所有规则并存入 RS。当 CCFP 树包含不止一条路径时，第 12－18 行从项头表 H 最下面的频繁项开始，依次挖掘包含当前频繁项的规则并存入 RS 中。

ACI 算法为不同类别实例设置不同的规则覆盖阈值。通过调高少数类的阈值，产生更多的覆盖少数类的实例，以提高少数类的预测效果。ACI 的规则裁剪函数 RulePrune 的伪代码如算法 3 所示。RulePrune 函数的输入是裁剪前的规则集合 RS、训练数据集 D 以及不同类别的规则覆盖条数阈值 *limits* 数组。输出为裁剪之后的规则集合 Rules。

算法 3 RulePrune 函数伪代码

```
输入：RS 为裁剪前的规则集合；D 为训练数据集；limits 为各类别覆盖条数阈值
输出：Rules 为裁剪后的规则集合
1. Rules=Null
2. R=sort(RS)
3. for d∈D do
4. d. CoverCount=0
5. endfor
6. while R≠Null and D≠Null do
7. for r∈R do
8. Insts=D 中被 r 覆盖的实例集合
9.     If|Insts|>0 then
10.      R=R－r
11.      Rules=Rules+r
12.      for d∈ Insts
13.        d. CoverCount++
14.        if d. CoverCount>=limits[r. cl]
15.          D=D－d
16.        endif
17.      end for
18.   endif
19. endwhile
20. returnRules
```

RulePrune 函数第 1 行将规则集合 Rules 初始化为空，第 2 行对 RS 中的规则进行排序并存入 R 中。排序时，首先比较两条规则的支持度，支持度越大，则规则排序越靠前。如果支持度相等，则比较两条规则的置信度，置信度越大越靠前。如果两条规则的支持度和置信度都相等，则先生成的规则排序优于后生成的规则。第 3－5 行将 D 中每条实例的规则覆盖条数初始化为 0。第 6－19 行循环处理每条规则，直到 R 为空或者 D 为空。对 R 中的每条规则 r，如果 r 覆盖 D 中的每条实例，则将 r 存入 Rules，并从 R 中删除。对 r 覆盖的每条实例 d，将其规则覆盖条数加 1，如果达到了 *limits* 的限制，则将 d 从 D 中删除。

算法 4 给出了对新实例分类函数的伪代码。Classify 函数的输入是裁剪之后的规则集合 Rules、待分类实例 d 以及类标签集合 C。输出是为 d 预测的类标签 *classlabel*。

算法 4 ACI 分类过程

```
输入：Rules 为关联规则集合；d 为待分类实例；C 为类标签集合
输出：classlabel 为 d 的类标签
1. for cl∈C
2. cl. weight=0
3. end for
4. for r∈Rules and r 覆盖 d
5.     r. weight=χ²χ²/max χ²
6.     cl=r 的类标签
7.     cl. weight=cl. weight+r. weight
8. end for
9. classlabel=max_{cl∈C} cl. weight
10. return classlabel
```

Classify 函数第 1－3 行将每个类标签的权重初始化为 0。第 4－8 行遍历规则集合 Rules，对 Rules 中覆盖 d 的每条规则 r，计算该规则的权重，并将权重加到 r 对应的类标签 cl 的权重中。第 9－10 行，将权重最大的类标签返回作为预测 d 的类标签。

3.2 算法复杂度分析

算法的时间复杂度主要取决于 3 部分。第 1 部分为关联

规则挖掘,本文算法采用的是 CCFPGrowth 函数,该函数分为两大步,第一步是构建 CCFP 树,其需要首先遍历数据集生成 HeaderTable,这一步的复杂度为 $O(N)$,其中 N 为数据集实例数;接着需要将每条实例插入到 CCFP 树中,这一步的时间复杂度也为 $O(N)$。第二步是生成规则,其复杂度取决于 CCFPGrowth 函数。CCFPGrowth 函数的时间复杂度为 $O(N\log(N)+N(N-1)/2)$。

第 2 部分是规则生成完后,对规则进行裁剪,包括规则排序和遍历。假设规则条数为 M,则这部分的时间复杂度为 $O(M\log(M))$。从而整个构建分类器的时间复杂度为 $O(N\log(N)+\frac{N(N-1)}{2}+M\log(M))=O(N^2)$。

第 3 部分是对新实例的分类,主要是对规则的遍历。因为裁剪后的规则条数不会超过 M,因此该部分的时间复杂度为 $O(M)$。

综上,ACI 算法的时间复杂度为 $O(N^2)$。

空间复杂度主要在 CCFP 树结构上,其复杂度为 $O(N)$。

4　实验及结果分析

4.1　实验设置及评估指标

为了验证 ACI 算法的有效性,使用了 27 个不均衡数据集进行实验。这些数据集均是二分类问题,来源于 keel 数据集[18],其详细信息如表 1 所列。

表 1　数据集

Table 1　Dataset

编号	数据集	实例数	类别数	属性个数	少数类实例数	多数类实例数
1	abalone-17_vs_7-8-9-10D	2338	2	9	58	2280
2	abalone-20_vs_8-9-10D	1916	2	9	26	1890
3	abalone-21_vs_8D	581	2	9	14	567
4	abalone-3_vs_11D	502	2	9	15	487
5	car-goodD	1728	2	7	69	1659
6	ecoli-0-1-4-6_vs_5D	280	2	7	20	260
7	ecoli-0-1-4-7_vs_2-3-5-6D	336	2	8	29	307
8	ecoli-0-1-4-7_vs_5-6D	332	2	7	25	307
9	ecoli-0-1_vs_2-3-5D	244	2	8	24	220
10	ecoli-0-1_vs_5D	240	2	7	20	220
11	ecoli-0-2-3-4_vs_5D	202	2	8	20	182
12	ecoli-0-2-6-7_vs_3-5D	224	2	8	22	202
13	ecoli-0-3-4-6_vs_5D	205	2	8	20	185
14	ecoli-0-3-4-7_vs_5-6D	257	2	8	25	232
15	ecoli-0-3-4_vs_5D	200	2	8	20	180
16	ecoli-0-4-6_vs_5D	203	2	7	20	183
17	ecoli-0-6-7_vs_3-5D	222	2	8	22	200
18	ecoli-0-6-7_vs_5D	220	2	7	20	200
19	flare-FD	1066	2	12	43	1023
20	glass-0-1-4-6_vs_2D	205	2	10	17	188
21	glass-0-1-5_vs_2D	172	2	10	17	155
22	glass-0-4_vs_5D	92	2	10	9	83
23	glass-0-6_vs_5D	108	2	10	9	99
24	led7digit-0-2-4-5-6-7-8-9_vs_1D	443	2	8	37	406
25	yeast-0-2-5-6_vs_3-7-8-9D	1004	2	9	99	905
26	yeast-0-2-5-7-9_vs_3-6-8D	1004	2	9	99	905
27	yeast-0-3-5-9_vs_7-8D	506	2	9	50	456

基准方法选用了两个关联分类算法 CMAR[6] 和 ACWV[8] 以及两个常用的不均衡学习算法,即使用决策树算法 J48[19] 做基分类器的 Bagging[20] 和 Boosting[20]。

分类性能评价指标使用了针对不均衡数据分类的 AUC[21]、少数类的 Recall[22] 以及 F1[23]。AUC 是 ROC 曲线下的面积,经常被用来评估一个二元分类器的好坏。Recall 即召回率,反映了有多少正类被正确识别出来,计算式为:

$$Recall=\frac{TP}{TP+FN} \tag{4}$$

其中,TP 是真正的正类实例数目,FN 是被预测为负类,而实际上是正类的实例数目。Precision[22] 为查准率,反映了预测为正类的实例中有多少是真正的正类,计算式为:

$$Precision=\frac{TP}{TP+FP} \tag{5}$$

F1 反映了 Recall 和 Precision 的综合性能,其计算公式如下:

$$F_\alpha=\frac{(\alpha^2+1)PR}{\alpha^2P+R} \tag{6}$$

其中,P 为 Precision,R 为 Recall,当 $\alpha=1$ 时即为 $F1$。

4.2　实验结果及分析

ACI 与基准方法在 27 个数据集上的 $w/d/l$ 结果如表 2 所列。

表 2　ACI 和基准算法各个指标上的 win/draw/loss

Table 2　win/draw/loss on each indicator of ACI and benchmark algorithm

基准算法	*AUC*	*Recall*	*F1*
	w/d/l	*w/d/l*	*w/d/l*
CMAR	16/9/2	15/11/1	8/12/7
ACWV	10/12/5	13/14/0	4/11/12
Bagging+J48	20/1/6	20/2/5	15/3/9
Boosting+J48	9/3/15	20/6/1	15/3/9

表中 w(win)指的是使用 Wilcoxon 符号秩检验[24] 得到 ACI 算法显著优于对比的基准算法的数据集个数,l(loss)指 ACI 算法显著差于对比算法的数据集个数,而 d(draw)指的是 ACI 算法和对比算法没有显著性差异的数据集个数。

从表 2 可以看出与关联分类算法相比,在 AUC 上,ACI 在 16 个和 10 个数据集上显著优于 CMAR 和 ACWV,在 9 个和 12 个数据集上与二者没有显著差异,只在 2 个和 5 个数据

集上差于二者。在 Recall 上，ACI 在 15 个和 13 个数据集上显著优于 CMAR 和 ACWV，在 11 个和 14 个数据集上与二者没有显著差异，只在 1 个和 0 个数据集上差于二者。综合两个指标，ACI 算法比关联分类算法 CMAR 和 ACWV 算法在不均衡数据集上表现更好。

从表 2 还可以看出与不均衡学习算法相比，在 AUC 上，ACI 在 20 个和 9 个数据集上显著优于 Bagging 和 Boosting，在 2 个和 3 个数据集上与二者没有显著差异，在 6 个和 15 个数据集上差于二者。在 Recall 上，ACI 在 20 个数据集上显著优于 Bagging 和 Boosting，在 2 个和 6 个数据集上与二者没有显著差异，只在 5 个和 1 个数据集上差于二者。综合两个指标，ACI 算法相比不均衡学习算法 Bagging 和 Boosting 在不均衡数据集上表现更好。

综上，相比基准算法，ACI 对不均衡数据的分类性能更好。

ACI 对不均衡数据处理的核心操作是为不同类别实例设置不同的规则覆盖条数限制，以增加覆盖少数类实例的规则。而 CMAR 在进行规则裁剪时没有区别多数类和少数类。表 3 给出了 ACI 和 CMAR 在不同数据集上得到的覆盖少数类实例的规则条数。

表 3 ACI 和 CMAR 的少数类规则数对比

Table 3 Comparison of number of rules for minority class between ACI and CMAR

数据集	规则总数	ACI 少数类规则数	CMAR 少数类规则数
abalone-17_vs_7-8-9-10D	1 730	51	21
abalone-20_vs_8-9-10D	1 300	20	10
abalone-21_vs_8D	976	24	8
abalone-3_vs_11D	764	4	2
car-goodD	712	45	16
ecoli-0-1-4-6_vs_5D	272	18	9
ecoli-0-1-4-7_vs_2-3-5-6D	548	30	15
ecoli-0-1-4-7_vs_5-6D	397	29	13
ecoli-0-1_vs_2-3-5D	408	21	11
ecoli-0-1_vs_5D	196	17	8
ecoli-0-2-3-4_vs_5D	432	20	10
ecoli-0-2-6-7_vs_3-5D	320	18	9
ecoli-0-3-4-6_vs_5D	440	22	11
ecoli-0-3-4-7_vs_5-6D	564	24	12
ecoli-0-3-4_vs_5D	464	20	10
ecoli-0-4-6_vs_5D	220	22	11
ecoli-0-6-7_vs_3-5D	320	18	9
ecoli-0-6-7_vs_5D	152	18	9
flare-FD	20 586	131	54
glass-0-1-4-6_vs_2D	512	4	2
glass-0-1-5_vs_2D	0	0	0
glass-0-4_vs_5D	1 280	6	3
glass-0-6_vs_5D	1 216	6	3
led7digit-0-2-4-5-6-7-8-9_vs_1D	768	40	20
yeast-0-2-5-6_vs_3-7-8-9D	752	34	17
yeast-0-2-5-7-9_vs_3-6-8D	1 200	74	37
yeast-0-3-5-9_vs_7-8D	448	12	6

从表 3 可以看出，ACI 算法确实得到了比 CMAR 更多的少数类规则。而正是由于少数类上用于分类的规则数的增多，降低了少数类的误分率。

4.3 参数分析

本文提出的算法针对多数类和少数类实例分别使用不同的规则覆盖条数限制阈值 *limit*。本节取不同的 *limits* 值来分析其分类效果的影响。在多数类实例的规则覆盖条数取值为 1，少数类实例的规则覆盖条数取值分别为 1，2，3，4，5，6，7 时，分类结果如图 2 所示。

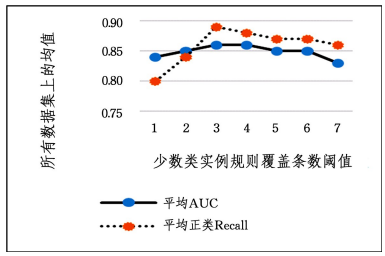

图 2 规则覆盖条数阈值对 ACI 分类效果的影响

Fig. 2 Influence of rule number threshold on classification effect of ACI

从图 2 可以看出，在少数类实例规则覆盖条数不断增大的过程中，少数类的 *Recall* 和 *AUC* 都呈现先增加后减少的趋势。可以看出，在阈值设置为 3 时，分类效果取得最优值。

结束语 本文提出了针对不均衡数据的关联分类算法 ACI。ACI 在进行规则裁剪时，为不同类别的实例设置不同的规则覆盖条数阈值。通过为少数类设置较大的阈值，以增加覆盖少数类的规则，从而提高少数类的分类效果。在 27 个不均衡数据集上的实验结果表明，与 4 个基准算法相比，ACI 在 3 个分类结果评价指标上取得了更好的性能。

参 考 文 献

[1] HIERONS R. Machine learning. Tom M. Published by McGraw-Hill, Maidenhead, U. K., International Student Edition, 1997. ISBN: 0-07-115467-1, 414 pages. Price: U. K. £22. 99, soft cover[J]. Software Testing Verification & Reliability, 2015, 9(3): 191-193.

[2] SALZBERG S L J M L. C4. 5: Programs for Machine Learning by J. Ross Quinlan. Morgan Kaufmann Publishers, Inc., 1993 [J]. Machine Learning, 1994, 16(3): 235-240.

[3] RAJPUT A. J48 and JRIP Rules for E-Governance Data [J]. IJCSS, 2011, 5(2): 201.

[4] FÜRNKRANZ J, WIDMER G. Incremental Reduced Error Pruning[C]// Machine Learning Proceedings. 1994: 70-77.

[5] HU K, LU Y, ZHOU L, et al. Integrating classification and association rule mining: A concept lattice framework[C]// International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing. Springer. 1999: 443-447.

[6] LI W, HAN J, PEI J. CMAR: Accurate and efficient classification based on multiple class-association rules[C]// Proceedings IEEE International Conference on Data Mining, 2001 (ICDM 2001). IEEE, 2001: 369-376.

[7] THABTAH F A, COWLING P, PENG Y. MMAC: A New Multi-Class, Multi-Label Associative Classification Approach [C]// IEEE International Conference on Data Mining. 2004.

[8] ZHU X, SONG Q, JIA Z. A Weighted Voting-Based Associative Classification Algorithm [J]. The Computer Journal, 2010, 53(6): 786-801.

[9] GANGANWAR V. An overview of classification algorithms for imbalanced datasets [J]. International Journal of Emerging Technology and Advanced Engineering, 2012, 2(4): 42-47.

[10] HE H, MA Y. Imbalanced learning. Foundations, algorithms, and applications[M]. Wiley-IEEE Press, 2013.

[11] ZHOU Z H, LIU X Y. On multi-class cost-sensitive learning [C]// National Conference on Artificial Intelligence. 2006.

[12] WU G, CHANG E Y. KBA: Kernel boundary alignment consi-

dering imbalanced data distribution[J]. IEEE Transactions on Knowledge & Data Engineering, 2005(6): 786-795.
[13] BREIMAN L. Bagging predictors [J]. Machine Learning, 1996, 24(2): 123-140.
[14] ZAREAPOOR M, SHAMSOLMOALI P. Application of credit card fraud detection: Based on bagging ensemble classifier[J]. Procedia computer science, 2015, 48(2015): 679-685.
[15] WITTEN I H, FRANK E, HALL M A, et al. Data Mining: Practical machine learning tools and techniques[M]. Morgan Kaufmann, 2016: 70-71.
[16] 韩家炜，坎伯. 数据挖掘：概念与技术[M]. 北京：机械工业出版社，2012：158-159.
[17] DEORA C S, ARORA S, MAKANI Z. Comparison of Interestingness Measures: Support-Confidence Framework versus Lift-Irule Framework[J]. International Journal of Enginnering Research & Applications, 2014, 3(2): 208-215.
[18] ALCALÁ-FDEZ J, FERNÁNDEZ A, LUENGO J, et al. KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework[J]. Journal of Multiple-Valued Logic & Soft Computing, 2011, 17: 255-287.
[19] PATIL T R, SHEREKAR S. Performance analysis of Naive Bayes and J48 classification algorithm for data classification[J]. International Journal of Computer Science and Applications, 2013, 6(2): 256-261.
[20] QUINLAN J R. Bagging, boosting, and C4.5 [C]// AAAI/IAAI. 1996: 725-730.
[21] LOBO J M, JIMÉNEZ-VALVERDE A, REAL R. AUC: a misleading measure of the performance of predictive distribution models[J]. Global Ecology and Biogeography, 2008, 17(2): 145-151.
[22] DAVIS J, GOADRICH M. The relationship between Precision-Recall and ROC curves[C]// Proceedings of the 23rd International Conference on Machine Learning. ACM, 2006: 233-240.
[23] POWERS D M. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation[J]. Journal of Machine Learning Technology, 2011, 2(1): 37-63.
[24] WILCOXON F, KATTI S, WILCOX R A. Critical values and probability levels for the Wil-coxon rank sum test and the Wilcoxon signed rank test[J]. Selected Tables in Mathematical Statistics, 1970, 1: 171-259.

CUI Wei, born in 1994, postgraduate. His main research interests include machine learning and data mining.

ZHU Xiao-yan, born in 1983, Ph.D, associate professor, is a member of China Computer Federation. Her main research interests include machine learning and data mining.

（上接第 487 页）

将 GM(1,1)与 SVM 组合模型运用到阿拉尔市人口数量预测研究中，得到阿拉尔市未来五年的人口数量分别为 35.38 万人、35.78 万人、36.29 万人、36.66 万人和 37.09 万人。由于《新疆生产建设兵团十三五规划》中关于各师市人口聚集的有相关政策的要求，阿拉尔市在十三五期间需向西部贫困山区招工落户 10 万人，2018 年招工人数为 2 万余人，从而导致 2018 年预测结果与阿拉尔市公布的人口数据之间存在差距。因此，阿拉尔市实际人口数量与预测值之间相差 2 万余人，5 年之后的 2022 年阿拉尔市实际人口数量约为 40 万余人，达到了预期的人口聚集目标。

参 考 文 献

[1] 郭雪峰，黄健元，王欢. 改进的灰色模型在流动人口预测中的应用[J]. 统计与决策，2018(8)：76-79.
[2] 蒋若凡，姜玉梅，李菲雅. 基于灰色 PSO-BP 人口预测模型的研究与应用[J]. 西北人口，2011，32(3)：23-26.
[3] 龙会典，严广乐. 基于改进的 GM(1,1)—Markov 链组合模型广东省单位 GDP 能耗预测[J]. 数理统计与管理，2017，36(2)：200-207.
[4] 李凯，张涛. 基于组合插值的 GM(1,1)模型背景值的改进[J]. 计算机应用研究，2018，35(10)：2994-2999.
[5] 吴文泽，张涛. GM(1,1)模型的改进及应用[J]. 统计与决策，2019(9)：15-18.
[6] 徐丽丽，李洪，李劲. 基于灰色预测和径向基网络的人口预测研究[J]. 计算机科学，2019，46(Z1)：431-435.
[7] 解伟，潘文明，王成化，等. 基于支持向量机的省级电网中长期投资规模预测模型研究[J]. 工业技术经济，2019(8)：154-160.
[8] 贾娜，郭佳欣，花军，等. 采用支持向量机算法对金刚石锯片锯切木材表面粗糙度的预测[J]. 东北林业大学学报，2019，47(10)：85-89.
[9] 徐路路，王芳. 基于支持向量机和改进粒子群算法的科学前沿预测模型研究[J]. 情报科学，2019，37(8)：22-28.
[10] 宋晓华，祖丕娥，伊静，等. 基于改进 GM(1,1)和 SVM 的长期电量优化组合预测模型[J]. 中南大学学报(自然科学版)，2012，43(5)：1803-1807.
[11] BATES J M, GRANGER C W. Combination of Forecasts[J]. Operational Res-Ouart, 1969, 20(4): 451-468.
[12] LIU S L, HU Z Q, CHI X K. The research of power load forecasting method on combination forecasting model[J]. Information Science and Engineering, 2010(26).
[13] 吴静敏，左洪福，陈勇. 基于免疫粒子群算法的组合预测方法[J]. 系统工程理论方法应用，2006，15(3)：229-233.

XU Xiang-yan, born in 1990, master, lecturer. Her main research interests include intelligent optimization algorithm and so on.

HOU Rui-huan, born in 1986, master, lecturer. His main research interests include nonparametric statistics and so on.