文章编号:1672-3961(2017)01-0015-07 DOI:10.6040/j.issn.1672-3961.0.2016.304

基于多次随机欠采样和POSS方法的软件缺陷检测

方昊,李云*
(南京邮电大学计算机学院,江苏 南京 210003)

摘要: 为了解决因软件缺陷数据存在数据不平衡问题限制了分类器的性能,将POSS(pareto optimization for subset selection)特征选择算法和随机欠采样技术引入到软件缺陷检测中,并利用支持向量机(support vector machine,SVM)构建预测模型。试验结果表明,通过多次随机欠采样可以有效地解决软件缺陷数据不平衡问题,同时使用POSS方法对目标子集进行双向优化,从而提高分类的准确率,其结果要优于Relief、Fisher、MI(mutual information)特征选择算法。

关键词: 软件缺陷检测;不平衡性;数据采样;特征选择
中图分类号: TP391 **文献标志码:** A

Random undersampling and POSS method for software defect prediction

FANG Hao, LI Yun*
(College of Computer, Nanjing University of Posts and Telecommunications, Nanjing 210003, Jiangsu, China)

Abstract: In order to solve the problem of imbalance distribution in software defect prediction, POSS(pareto optimization for subset selection) feature selection and random undersampling was applied in this paper, and SVM was used to build the prediction model. The experimental results showed that the problem could be solved effectively by using multiple random undersampling, and the POSS method was treated subset selection as a bi-objective optimization, which could improve the accuracy of classification, the effectiveness of proposed method was verified by comparing with Relief、Fisher、MI(mutual information).

Key words: software defect prediction; class imbalance; data sampling; feature selection

0 引言

软件缺陷检测对于降低软件开发成本和保证软件系统[1]质量具有重要作用。项目经理可以根据前期的缺陷预测模型合理地分配有限的时间和人力资源。根据软件缺陷的相关工作,可以将其分为三种类型[2]:(1)评估软件系统中缺陷的数量;(2)挖掘缺陷的关联性;(3)对软件模块分类,即有缺陷和无缺陷两类。

本研究主要关注第三种类型,该类型是利用机器学习方法将软件模块分类,即有缺陷和无缺陷两类。现在已有很多机器学习方法应用到软件缺陷检测中,如集成学习[3]、神经网络[4]、类比方法[5]等。这些方法都是通过训练历史数据集来构建分类器,并预测未知的软件模块的缺陷倾向[6]。基于机器学习的软件缺陷检测通常分为三个步骤,首先对从软件中得到的原始数据进行特征提取以及对数据进行预处理,形成一个格式化的软件数据集;然后对该数据集进行特征选择,剔除不相关和冗余的特征

收稿日期:2016-07-22; 网络出版时间:2016-12-19 15:30:33
网络出版地址:http://www.cnki.net/kcms/detail/37.1391.T.20161219.1530.002.html
基金项目:江苏省自然科学基金资助项目(BK20131378, BK20140885);广西高校云计算与复杂系统重点实验室资助项目(15206)
作者简介:方昊(1989—),男,江苏宿迁人,硕士研究生,主要研究方向为特征选择. E-mail:15150662912@163.com
* 通讯作者:李云(1974—),男,安徽望江人,教授,博士,主要研究方向为机器学习与模式识别. E-mail:liyun@njupt.edu.cn

以提高分类性能和分类效率;最后使用该数据集训练一个分类算法来预测软件缺陷。然而,随着现代软件系统的规模和复杂度不断增加,提取出的特征维度数不断增加,其中包含了大量冗余特征,从而影响了分类性能以及降低了分类准确率。特征选择作为基于机器学习的缺陷检测的一个关键步骤,其目的是提高分类性能和准确率[7],现已成为研究的热点。

特征选择算法广泛应用于多个领域,其目标是从特征集合中挑选出满足一定评价准则的最优特征子集的过程,通过去除不相关特征和冗余特征,从而实现降维[8]。特征选择算法具有以下作用:第一,避免维度灾难的发生;第二,减少构建学习器的代价;第三,提高学习器的准确率;第四,增强可理解性。根据以往的研究[9],在软件缺陷检测中,特征选择是解决冗余性和无关性最有效的方法之一。目前特征选择算法主要分为四类:过滤器、封装器、嵌入式和组合式[10]。过滤式特征选择算法独立于学习器,根据特征固有的性质来决定每个特征的重要性,具有较高的时间效率,如 Fisher[11]、Relief[12]等;封装式的算法采用学习器来评价特征子集的性能,主要依赖于不同的搜索策略,如前向搜索、后向搜索、随机搜索等,准确率较高,但是依赖特定的学习器;嵌入式算法是在构建学习器的过程中进行特征选择,在效率和准确率上是过滤式和封装式的折中,如 SVMRFE[13];组合式的算法[14]先用过滤式算法去除一些特征,再对剩下的特征集合采用封装式算法查找出最优子集,同时利用了过滤式和封装式的优点。而特征选择的结果又有三种形式:特征权重,特征排序和特征子集[15]。其中特征权重可以根据权重大小转化为特征排序;特征权重和特征排序可以通过设定一定的阈值转化为特征子集。当前,多种特征选择算法已经广泛应用于软件领域中。MARCHIORI 等人使用信息增益降低了特征维度,在保证了分类性能的同时加快了软件缺陷模型的学习速度[16]。RODRIGUEZ 等人在软件缺陷数据集上应用特征选择算法,并得出使用更少的特征可以提高分类准确率的结论[17]。FORMAN 等人提出由特征排序和特征子集组合而成的混合特征选择算法[18]。

另一方面,软件数据集中存在着不平衡的特点。所谓不平衡指的是数据集类间分布存在明显的不平衡的模式分类问题。具体地说就是某些类的样本数量远远少于其他类。在软件缺陷的数据集中,具有少量样本的类称为小类,相应的具有大量样本的类称为大类。在软件领域中,存在缺陷的软件总是占少数的。在一个庞大的软件数据集中,可能少数有几个软件样本是存在缺陷的,而绝大多数软件样本是正常的。然而,为了提高软件的质量,这少数的几个存在缺陷的软件正是本研究需要识别出来的。因此,在进行软件缺陷检测时,对不平衡数据的处理也是一大研究热点。目前已经提出了很多解决不平衡问题的方法,比较主流的方法是数据采样,包括随机欠采样(random undersampling, RUS)和随机过采样(random oversampling),上述两种方法是通过增加少数类样本或者减少多数类样本来构建一个相对平衡数据集。随机欠采样是非常简单且有效的采样方法,并不像其他复杂的采样算法,仅仅是通过随机地剔除多数类样本来构建平衡数据集。虽然随机欠采样有可能会丢失大量的重要信息的样本,但是缩短了模型的训练时间。相反,随机过采样不丢失样本信息,而是通过复制少数类样本来达到数据的平衡,这样很容易导致过拟合,并且增加了模型的训练时间。本研究采用多次随机欠采样来弥补丢失随机欠采样中重要样本的缺点,并且将每次采样的结果进行特征选择,得出若干个特征序列,最终采用均值法选取最终的特征序列。

1 设计方法

为了解决不平衡问题并且提高分类性能,提出了基于多次随机欠采样与 POSS[19]特征选择算法相结合的算法。其大致流程如图 1 所示,主要包括两个部分:(1)评估方案模块,其核心部分为学习方案。评估方案模块是针对学习方案的结果进行性能的评估,其中学习方案主要由数据预处理、特征选择、学习算法三个部分组成。(2)软件缺陷检测模块,是根据学习方案选择最佳性能的分类器,最终预测出软件系统的软件缺陷倾向。下面,对上述两个模块做简单介绍。

1.1 评估方案模块

评估方案模块是软件缺陷预测框架的基础部分,该部分首先实现对数据集的划分,即将历史数据划分为适合机器学习的训练样本和测试样本。再将训练样本作为学习方案模块输入,该模块是评估方案的核心部分,最后训练出分类模型。通常采用十次交叉验证来验证该分类模型的性能。学习方案部分中通过对数据的预处理、特征选择以及学习算法来实现对分类模型的构建,学习方案主要由以下三个部分组成。

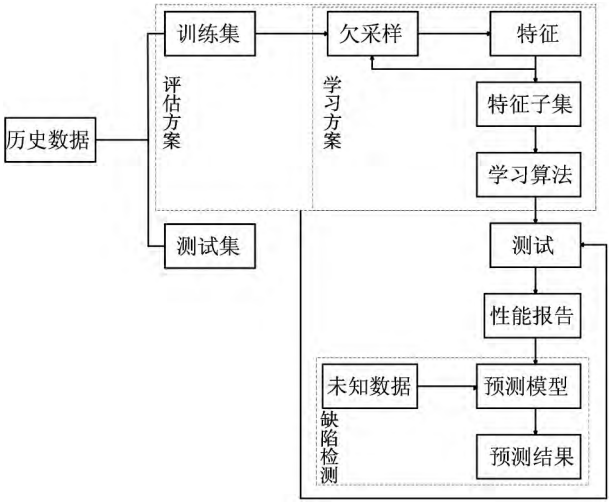

图 1　软件缺陷预测框架图

Fig. 1　Software defect prediction framework

(1) 数据预处理。数据预处理作为机器学习步骤的第一步,对最后的分类性能有着重要的影响。该步骤会对训练数据进行离群点、缺失值处理。由于软件数据不平衡的特点,本研究在该步骤采用多次随机欠采样的方法。该方法在有效地解决不平衡问题的同时也弥补了随机欠采样可能丢失重要样本信息以及破坏样本分布的缺点。

(2) 特征选择。从软件中提取出的原始数据集中的特征并不一定都是针对缺陷检测的。在进行软件缺陷检测时,可能只需要其中的部分特征。过多的特征可能会造成数据冗余从而影响分类性能和分类效率。因此,特征选择也是软件缺陷检测的一个关键步骤。本研究采用了四种特征选择算法对原始样本进行特征筛选: 互信息(MI)[10]、Relief[12]、Fisher[11] 以及 POSS[19]。

(3) 学习算法。本研究采用 SVM 作为分类算法,其原因是分类性能不受特征子集的影响[21]。SVM 作为一个性能较好的二分类算法,被广泛用于在软件工程和数据挖掘领域。

1.2　软件缺陷检测模块

软件缺陷检测模块主要包括构建分类器和缺陷预测两部分。首先,根据由评估方案得出来的性能报告,通过对分类模型的不断测试,确定一个最佳的学习方案,并构建一个相对最优的分类器; 然后利用该分类器对未知的软件数据集进行缺陷预测,得到最终的缺陷检测报告。

2　多次随机欠采样和 POSS 方法

2.1　POSS 算法

本研究使用了一种新型的特征选择算法 POSS,该方法是基于帕累托最优的方法并对其改进,从而获得相对最优的特征子集[20]。该方法被 ZHOU 在文献[20]中提出,并被广泛使用。其大致思想为: 假设给定数据集 $V=\{X_1,X_2,\cdots,X_n\}$(n 表示特征数量),判别函数 f 以及正整数 k。该特征选择算法是寻找一个特征子集 $S\subseteq V$,并在约束条件 $|S|\leqslant k$ 的情况下,判别函数取得最优值。其中 $|\cdot|$ 表示数据集的大小。其定义如式(1)所示:

$$\arg\min_{S\subseteq V} f(S),\ \text{s.t.}\ |S|\leqslant k, \tag{1}$$

式(1)可以看作有两个优化目标: (1)优化判别函数 f,即 $\arg\min_{S\subseteq V} f(S)$; (2)同时要保证 $|S|$ 最小,即 $\min_{S\subseteq V}\max\{|S|-k,0\}$。但一般情况下,想要取得更好的判别函数值,即 $f(S)$ 值,需要更大的样本数量,所以可以说这两个优化目标在一定程度上是矛盾的。然而,POSS 方法可以有效地实现上述两个优化目标。其具体步骤描述如下:

在该算法中,使用 n 维二元向量 $s=\{0,1\}^n$ 来表示某一个特征是否被选中,称 s 为特征选择的一种解决方案。即当 $s_i=1$ 时表示第 i 个特征将会被选到特征子集 S 中,反之将不会被选入。接着,为所有可能的解决方案 s 定义两个属性 o_1,o_2。前者表示评价值,后者表示稀疏值,具体如式(2)所示:

$$\begin{aligned} s.o_1 &= \begin{cases} +\infty, & s=\{0\}\ \text{or}\ |s|\geqslant 2k; \\ f(s), & \text{otherwise}, \end{cases} \\ s.o_2 &= |s|, \end{aligned} \tag{2}$$

当 o_1 的值趋向于 $+\infty$ 时,则说明该方案效果较差,可予以舍弃。然后再引入一个函数 $I: \{0,1\}^n\rightarrow \mathbf{R}$[22]。该函数决定了两个方案是否可作比较,也就是说如果它们的隔离函数值不相等,则不能比较,如果相等,对于方案 s' 和方案 s,若 $s'.o_1\leqslant s.o_1$,则说明 s' 弱优于 s。若 $s'.o_2\leqslant s.o_2$ 且 $s'.o_1<s.o_1$,则 s' 优于 s。但是如果 s 既不优于 s',s' 也不优于 s,则说明他们之间不可比较。具体算法如算法 1。

算法 1:

输入: $v=\{x_1,x_2,\cdots,x_n\}$,标准函数 f 和正整数 k

参数: 迭代次数 T 和隔离函数 $I: \{0,1\}^n\rightarrow\mathbf{R}$。

输出: 最多为 k 个 v 中的子集

过程:

(1) 令 $s=\{0\}^n$ 和 $P=\{s\}$

(2) 令 $t=0$

(3) While $t<T$ do

(4) 均匀随机的从 P 中抽取 s

(5) 以 $\frac{1}{n}$ 的概率翻转 s 中的一位,产生 s'

(6) if $\nexists z\in P$,若 $I(z)=I(s')$ 并且 $((z.o_1<s'.o_1\wedge z.o_2<s'.o_2)$ or $(z.o_1\leqslant s'.o_1\wedge z.o_2\leqslant s'.o_2))$

(7) 则 $Q=\{z\in P\mid \boldsymbol{I}(z)=\boldsymbol{I}(s')\wedge s'.o_1\leqslant z.o_1\wedge s'.o_2\leqslant z.o_2\}$

(8) $P=(P\backslash Q)\cup\{s'\}$

(9) end if

(10) $t=t+1$

(11) end while

Return $\arg\min_{s\in P,|s|\leqslant k} f(s)$

2.2 基于多次欠采样的特征选择算法

本研究提出的算法是针对不平衡数据集的特征选择算法。其伪代码如算法2所示。该方法的整体分为两个步骤。

(1) 利用随机欠采样技术来获得平衡的数据集。在本研究工作中,采样比为50:50,即采样后的正类样本数和负类样本数的比率。

(2) 应用特征选择技术处理采样后的数据,并且根据预测能力(分数)来排序特征。本研究主要对四种特征选择算法进行比较分析。

由于单次随机欠采样可能会删除重要的样本,因此,为了减小其带来的偏差,本研究采用多次随机欠采样以及多次特征选择,即重复上述两个步骤 K 次($K=10$),最后利用投票法选择出相对最优的特征子序列。

算法2

输入:

1. 具有 F^j 数据集 D, $j=1,\cdots,m$;

2. 每个样本 $X\in D$ 可以被分为两个类别中的一个类 $c(x)\in\{\text{fp},\text{nfp}\}$;

3. 特征选择算法 $\omega=\{\text{MI},\text{Relief},\text{Fisher},\text{POSS}\}$;

4. 数据采样技术 RUS;

输出:

特征序列;

For $i=1$ to k do

利用 RUS 处理数据集 D 并获得平衡数据集 D_i;

使用 ω 对 D_i 中的 F^j 排序,并得到排序后的特征 $\omega(F^j)$, $j=1,\cdots,m$;

End

利用均值法对 k 次不同的 $\omega(F^j)$ 排序,并获得最终的特征序列。

3 试验结果分析

3.1 试验准备

3.1.1 数据集

本研究中的数据集使用了美国宇航局 MDP 项目中使用的数据集中的两个。该项目是由美国宇航局提供的一个软件度量库。每个数据集中的样本都来自于美国宇航局的软件系统或者子系统,包含每个组成模块的静态代码度量和相应的缺陷标记数据。这些项目通过一个 bug 跟踪系统的记录数来记录模块的缺陷数。类不平衡的比例一般为 10% ~ 20%。软件的模块数即样本个数为 498 ~ 2 109。本研究所使用的数据集名称、属性个数、样本数量、存在缺陷的样本数量以及每个数据集的不平衡比率如表1所示。

表1 NASA 数据集描述

Table 1 The description of NASA's dataset

数据集名称	属性个数	样本(模块)数量/个	存在缺陷的样本数量/个	不平衡比例/%
cm1	21	498	49	11.0
kc1	21	2 109	326	18.3

3.1.2 评价准则

对于数据不平衡问题,通常情况下应关注小类样本的分类准确率。因此,F-Measure[23]、G-Means[24]、Balance[25]、ROC[26]曲线等常用于评价数据不平衡问题的分类性能。由于需要同时考虑到大类样本和小类样本的分类准确率,本研究使用 G-Mean 和 Balance 作为评价准则。下面分别介绍上述两个评价指标。

(1) G-Means

$$\text{G-Means}=\sqrt{\text{PR}\times\text{NR}},$$

$$\text{PR}=\text{TP}/(\text{TP}+\text{FN}),\ \text{NR}=\text{TN}/(\text{TN}+\text{FP}),$$

其中: PR 是正类样本的分类准确率; NR 是负类样本的分类准确率; TP 是被正确分类的正类样本数量; FN 是被错误分类的正类样本数量; TN 是被正确分类的负类样本的数量; FP 是被错误分类的负类样本的数量。

(2) Balance

$$\text{Balance}=1-\left(\sqrt{(0-\text{PF})^2+(1-\text{PD})^2}\right)/\sqrt{2},$$

$$\text{PF}=\text{FP}/(\text{TN}+\text{FP}),$$

$$\text{PD}=\text{TP}/(\text{TP}+\text{FN}),$$

其中: PF(false positive rate, PF)为误报率[25],也称为假阳率,是错误预测为有缺陷模块数与实际无缺陷模块数的比; PD 为召回率(recall)[24],表示正确预测为有缺陷数的模块与真实有缺陷的模块数的比值。一般认为 PD = 1, PF = 0 是理想点,即所有的模块都可以被正确分类。Balance 是用来计算真实的(PF,PD)点到(0,1)点的欧式距离[20]。

3.1.3 特征选择算法

本研究主要使用 POSS 特征选择与其他三种常

用的特征选择算法进行比较,这三种特征选择算法分别为: Relief、Fisher、MI。本小节对其他三种特征选择算法做简单的介绍。

Fisher 特征选择算法是一个广泛使用的特征选择算法。该种算法将每个特征的费希尔值作为权重,主要思想是对于每一个特征,在不同的样本上的均值相差越大,方差之和越小,则被赋予的权重越大。

Relief 特征选择算法是根据样本在不同的特征上的假设间隔为特征赋予不同的权重。假设间隔是指保持样本分布不变的情况下决策面所能移动的最大距离。对于每个特征而言,样本的假设间隔越大,则该特征被赋予的权重越大。

互信息(MI)是信息论里对有用信息的度量,起初是作为衡量两个信号的关联的尺度,后来引申为两个随机变量的关联程度。基于互信息的特征选择算法常用于计算特征与类别之间的相关性,其互信息越大,表示特征对分类越有效。互信息也可以计算特征之间的相关性,其值越大,则表示两个特征越相关。

3.2　本研究设计和结果分析

本研究包括两个部分。第一部分对比不同特征选择算法的性能; 第二部分验证了多次随机欠采样与单次随机欠采样在不平衡数据中的性能。特征选择算法分别使用了 POSS、Relief、Fisher、MI。分类算法使用支持向量机,其 sigmoid 参数设置为 2,损失函数 C 设置为 32,参数计算的方法使用 SMO 算法。

3.2.1　不同的特征选择算法比较

本研究比较的特征选择算法都使用了基于多次欠采样的特征选择算法。每个特征选择算法均分别取 3,4,5,6,7,8 这 6 个特征得到不同的特征子集。对所有特征子集在上述提到的两个数据集中测试其分类性能。其中 KC1 的本研究结果如图 2 及图 3 所示,CM1 的本研究结果如图 4 及图 5 所示。

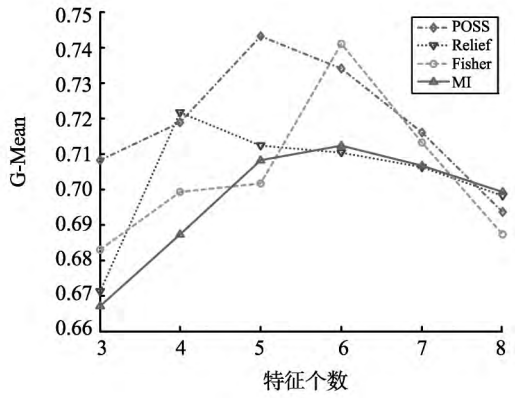

图 2　在 KC1 数据集上四种不同的特征选择算法 G-Mean 结果的比较

Fig. 2　In the KC1 data set and compare four algorithms results with different feature selection in G-Mean method

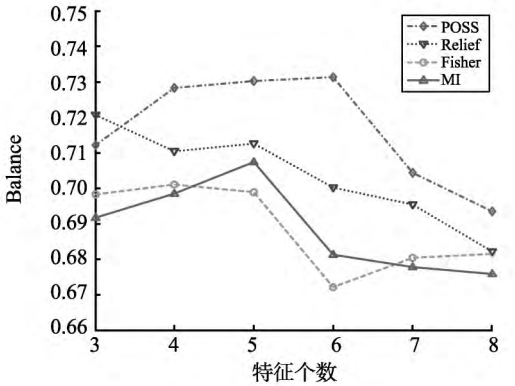

图 3　在 KC1 数据集上四种不同的特征选择算法 Balance 结果的比较

Fig. 3　In the KC1 data set and compare four algorithms results with different feature selection in Balance method

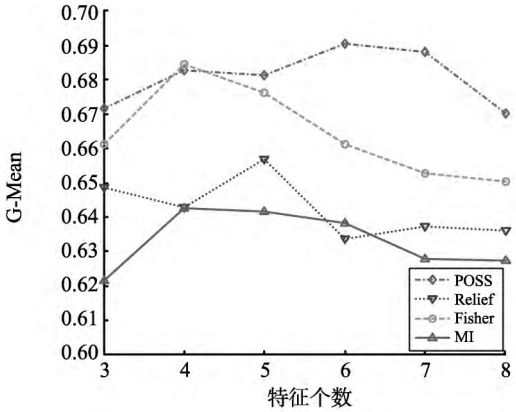

图 4　在 CM1 数据集上四种不同的特征选择算法 G-Mean 结果的比较

Fig. 4　In the CM1 data set and compare four algorithms results with different feature selection in G-Mean method

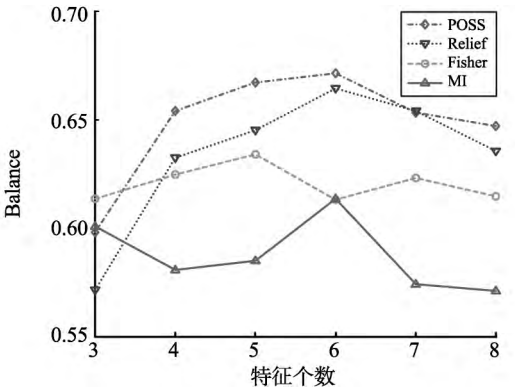

图 5　在 CM1 数据集上四种不同的特征选择算法 Balance 结果的比较

Fig. 5　Comparison of four algorithms results with different feature selection in Balance method

试验结果显示,在大部分情况下,当选取 5 个特征时,四种特征选择算法性能可以达到最好。在不同特征选择对比的情况来看,POSS 特征选择算法在大多数情况下的性能都优于其它三种特征选择算法,该特征选择算法在 KC1 数据集上选取 5 个特征达到最优,而在 CM1 上在之前的 5 个特征的基础上多选一个特征效果可以达到最优。

3.2.2 不同的采样策略比较

本研究验证多次欠采样对不平衡数据的处理性能。由于随机欠采样可能破坏样本的分布或者可能删除带有重要信息的样本,提出基于多次随机欠采样的 POSS 方法用以弥补单次随机欠采样的缺陷。本研究采用 1:1 比例的随机欠采样策略,即随机选择与小类样本数量相同的大类样本。特征选择算法使用 POSS,特征数量在 KC1 数据集中选择 5 个特征,在 CM1 中选择 6 个特征的特征子集作为训练样本。试验结果如图 6 和图 7 所示。

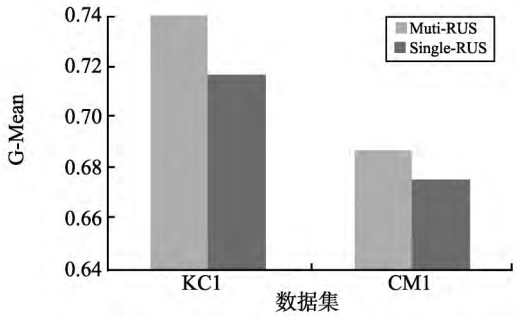

图 6 单次随机欠采样和多次随机欠采样在不同数据集中的 G-Mean 结果比较

Fig. 6 Comparison of G-Mean results in different data sets with single random under sampling and multiple random under sampling

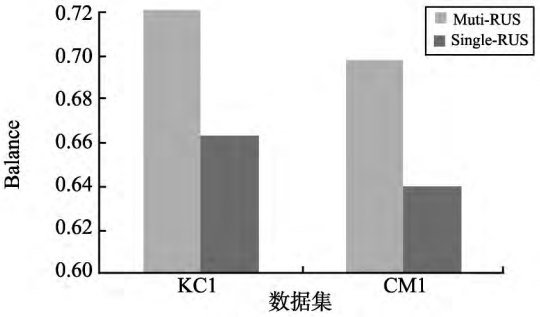

图 7 单次随机欠采样和多次随机欠采样在不同数据集中的 Balance 结果比较

Fig. 7 Comparison of Balance results in different data sets with single random under sampling and multiple random under sampling

结果显示,多次随机欠采样能有效地弥补单次随机欠采样中有可能删除重要样本的缺点,对不平衡数据有良好的泛化能力,在处理不平衡数据的问题上能取得相对较好的效果[27]。

4 结语

本研究提出一种基于多次随机欠采样和 POSS 特征选择算法,并将其应用到软件缺陷检测中。该方法是把子集的选择作为双目标优化问题,同时满足子集个数的约束。POSS 方法的可行性已经得到了理论上的证明。试验证明,提出的方法在软件缺陷检测中能比其他方法取得更好的效果,并且具有一定的实用性。算法的不足之处在于采用单一的采样比,不能从整体上表现出算法的有效性。这也是在以后工作中需要改进的地方。

参考文献:

[1] SONG Q, JIA Z, SHEPPERD M, et al. A general software defect-proneness prediction framework [J]. IEEE Transactions on Software Engineering, 2011, 37(3): 356-370.

[2] MUNSON J C, KHOSHGOFTAAR T M. Regression modelling of software quality: empirical investigation [J]. Information and Software Technology, 1990, 32(2): 106-114.

[3] ZHENG J. Cost-sensitive boosting neural networks for software defect prediction [J]. Expert Systems with Applications, 2010, 37(6): 4537-4543.

[4] KHOSHGOFTAAR T M, SELIYA N. Analogy-based practical classification rules for software quality estimation [J]. Empirical Software Engineering, 2003, 8(4): 325-350.

[5] CHIDAMBER S R, KEMERER C F. A metrics suite for object oriented design [J]. IEEE Transactions on Software Engineering, 1994, 20(6): 476-493.

[6] KHOSHGOFTAAR T M, GAO K, NAPOLITANO A. An empirical study of feature ranking techniques for software quality prediction [J]. International Journal of Software Engineering and Knowledge Engineering, 2012, 22(2): 161-183.

[7] GAO K, KHOSHGOFTAAR T M, WANG H, et al. Choosing software metrics for defect prediction: an investigation on feature selection techniques [J]. Software: Practice and Experience, 2011, 41(5): 579-606.

[8] KHOSHGOFTAAR T M, GAO K, NAPOLITANO A, et al. A comparative study of iterative and non-iterative feature selection techniques for software defect prediction [J]. Information Systems Frontiers, 2014, 16(5): 801-822.

[9] BOEHM B W, PAPCCIO P N. Understanding and controlling software costs [J]. IEEE Transactions on Software Engineering, 1998, 14(10): 1462-1477.

[10] 姚旭,王晓丹,张玉玺. 特征选择综述 [J]. 控制与决策, 2012,27(2): 161-166.
YAO Xu, WANG Xiaodan, ZHANG Yuxi. Survey of feature selection methods [J]. Control and Decision, 2012, 27(2): 161-166.

[11] GU Q, LI Z, HAN J. Generalized fisher score for feature selection [C] // Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence, UAI 2011. Bar-

celona, Spain: AUAI Press, 2011: 266-273.

[12] ROBNIK-SIKONJA M, KONONENKO I. Theoretical and empirical analysis of ReliefF and RReliefF [J]. Machine Learning, 2003, 53(1-2): 23-69.

[13] GUYON I, WESTON J, BARNHILL S, et al. Gene selection for cancer classification using support vector machines [J]. Machine Learning, 2002, 46(1-3): 389-422.

[14] LIU H, YU L. Toward integrating feature selection algorithms for classification and clustering [J]. IEEE Transactions on Knowledge and Data Engineering, 2005, 17(4): 491-502.

[15] WOZNICA A, NGUYEN P, KALOUSIS A. Model mining for robust feature selection [C] //Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Beijing, China: ACM, 2012: 913-921.

[16] JONG K, MARCHIORI E, SEBAG M, et al. Feature selection in proteomic pattern data with support vector machines [C] //Proceedings of the 2004 Symposium on Computational Intelligence in Bioinformatics and Computational Biology. La Jolla, USA: IEEE, 2004: 41-48.

[17] RODRIGUEZ D, RUIZ R, CUADRADO-GALLEGO J, et al. Detecting fault modules applying feature selection to classifiers [C] //Proceedings of the 2007 IEEE International Conference on Information Reuse and Integration. Las Vegas, USA: IEEE, 2007: 667-672.

[18] FORMAN G. An extensive empirical study of feature selection metrics for text classification [J]. Journal of Machine Learning Research, 2003, 3(2): 1289-1305.

[19] QIAN C, YU Y, ZHOU Z H. Subset Selection by Pareto Optimization [C] //Proceedings of the Advances in Neural Information Processing Systems 28 (NIPS 2015). Montreal, Canada: NIPS, 2015: 1774-1782.

[20] 徐燕, 李锦涛, 王斌, 等. 基于区分类别能力的高性能特征选择方法 [J]. 软件学报, 2008, 19(1): 82-89.
XU Yan, LI Jintao, WANG Bin, et al. A high performance feature selection method based on classification [J]. Journal of Software, 2008, 19(1): 82-89.

[21] 马衍庆. 基于机器学习的网络流量识别方法与实现 [D]. 济南: 山东大学, 2014.
MA Yanqing. Internet traffic classification and identification based on machine learning [D]. Jinan: Shandong University, 2014.

[22] YU Y, YAO X, ZHOU ZH. On the approximation ability of evolutionary optimization with application to minimum set cover [J]. Artificial Intelligence, 2012, 180-181(2): 20-33.

[23] HUANG Y J, POWERS R, MONTELIONE G T. Protein NMR recall, precision, and F-measure scores (RPF scores): structure quality assessment measures based on information retrieval statistics [J]. Journal of the American Chemical Society, 2005, 127(6): 1665-1674.

[24] ZHAO Z, GUO S, XU Q, et al. G-means: a clustering algorithm for intrusion detection [C] //Proceedings of the Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). [S. l.]: Springer, 2009, 5506: 563-570.

[25] WANG S, YAO X. Using class imbalance learning for software defect prediction [J]. IEEE Transactions on Reliability, 2013, 62(2): 434-443.

[26] METZ C E. Basic principles of ROC analysis [J]. Seminars in Nuclear Medicine, 1978, 8(4): 283-298.

[27] KUBAT M, MATWIN S. Addressing the curse of imbalanced training sets: one-sided selection [C] //Proceedings of the Fourteenth International Conference on Machine Learning. Stanford, USA: ICML, 2000: 179-186.

(编辑: 宋艳)