文章编号:1000 - 2375(2017)05 - 0550 - 08

基于 SSDBSCAN 的跨项目缺陷预测数据筛选方法

伍蔓[1,2,3],张建升[1,2],马传香[1,2],安格格[3],余啸[3]

(1. 湖北大学计算机与信息工程学院,湖北 武汉 430062;2. 湖北省教育信息化工程研究中心,湖北 武汉 430062;
3. 武汉大学软件工程国家重点实验室,计算机学院,湖北 武汉 430072)

摘要:针对跨项目软件缺陷预测中大量不相关的跨项目数据损害了缺陷预测模型性能的问题,提出了一种基于 SSDBSCAN(semi-suppervised density-based clustering)的跨项目缺陷预测数据筛选方法——SSDBSCAN filter. 首先,SSDBSCAN filter 结合少量带类标号的本项目历史数据、跨项目历史数据和大量不带类标号的本项目数据;然后,利用 SSDBSCAN 算法对这些数据进行聚类发现子簇;最后,收集子簇中的跨项目数据,不属于任何簇的跨项目数据被作为噪声数据而丢弃. 实验使用 15 个公开的 PROMISE 数据集,3 种分类器和 4 种性能度量指标. 实验结果表明,相比于目前已有的 Burak filter 和 DBSCAN filter 方法,SSDBSCAN filter 在提高了预测率的同时降低了误报率,且 G-measure 与 AUC 度量值更佳.

关键词:跨项目缺陷预测;数据筛选;SSDBSCAN

中图分类号:TB324. 1　　**文献标志码**:A　　**DOI**:10.3969/j.issn.1000-2375.2017.05.021

A data filtering method based on SSDBSCAN for cross-project defect prediction

WU Man[1,2,3], ZHANG Jiansheng[1,2], MA Chuanxiang[1,2], AN Gege[3], YU Xiao[3]

(1. School of Computer Science and Information Engineering, Hubei University, Wuhan 430062, China;
2. Educational Informationalization Engineering Research Center of Hubei Province, Wuhan 430062, China;
3. State Key Lab of Software Engineering, Computer School, Wuhan University, Wuhan 430072, China)

Abstract: Since massive irrelevant cross-project data degrades the performance of cross-project defect prediction model in cross-project defect prediction, a data filtering method based on SSDBSCAN (i. e., SSDBSCAN filter) was proposed for cross-project defect prediction. Firstly, the method combined limited amount of labeled within-project data, cross-project data and unlabeled within-project data. Secondly, sub-clusters were found by using SSDBSCAN algorithm. Finally, cross-project data in the sub-clusters was collected and some cross-project data which did not belong to any sub-clusters was discarded as noisy data. The experiments were conducted on 15 public PROMISE datasets using three different classifiers with four performance metrics. Compared with Burak Filter and DBSCAN Filter, the experimental results showed that SSDBSCAN Filter increased *PD* value, reduced *PF* value and achieved higher G-measure and *AUC* values.

Key words: academic cross-project defect prediction; data filter; SSDBSCAN

0　引言

软件缺陷是计算机软件或程序中存在的某个破坏正常运行能力的问题、错误以及隐藏的功能缺陷.

收稿日期:2017 - 01 - 28
基金项目:湖北省自然科学基金(2011CDB072)和湖北大学《数据挖掘》精品课程经费资助
作者简介:伍蔓(1996 -),女,本科生; 马传香(1971 -),女,通信作者,教授,博士,主要研究方向为数据挖掘、云计算,E-mail: mxc838@ hubu. edu. cn

随着软件系统在工程应用中的不断扩大,软件缺陷导致的经济损失日益增加. 因此,高效的软件缺陷预测技术越来越受到研究者的关注. 目前,已有很多高效的软件缺陷预测方法被提出[1-4],但它们通常基于项目内缺陷预测,当有足够的历史数据可用来建立缺陷预测模型时,本项目缺陷预测效果最佳. 但对于一些新的项目来说,项目内的历史数据非常有限,本项目缺陷预测很难顺利进行. 跨项目缺陷预测通过利用一个或多个已有的其他项目(跨项目)的历史数据来训练预测模型,然后将模型应用到本项目的缺陷预测中,从而解决本项目没有足够的历史数据来训练模型的问题. Zimmermann 使用 12 个项目构建 622 个跨项目组合,对这些跨项目缺陷预测模型的性能进行评估,发现由于跨项目数据与本项目数据的数据分布不同,大量不相关的跨项目数据损害了缺陷预测模型的性能[5]. 因此在使用跨项目数据构建缺陷预测模型之前,对跨项目数据进行数据筛选,找出与本项目数据最相关的数据尤为必要.

近年来,有研究人员提出了一些跨项目缺陷预测数据筛选方法. Turhan 提出了 Burak Filter,利用 K 近邻过滤器筛选跨项目数据中与本项目数据最相似的数据作为训练样本[6]. 实验证明,使用 Burak Filter 筛选的跨项目数据构建的预测模型,其误报率大大降低. Peter 提出了 Peter Filter 跨项目数据筛选方法,不同于 Burak Filter 方法仅使用了 K 近邻算法, Peter Filter 将 K 近邻算法与 K-Means 聚类算法相结合[7]. 实验结果表明 Peter Filter 方法取得了更佳的预测性能. Burak Filter 和 Peter Filter 方法操作简单、容易理解,但其筛选数据的效果容易受到设定的 K 值的影响,若 K 值设定太小,会丢失大量相关的数据实例;若 K 值设定太大,近邻中又可能包含太多的不相关的数据实例. 随后,聚类算法渐渐被运用到跨项目数据筛选中. Kawata 等尝试使用更为先进的 DBSCAN 算法进行跨项目数据的筛选,提出了基于 DBSCAN 的跨项目数据筛选方法 (DBSCAN Filter) [8]. DBSCAN Filter 将跨项目数据和本项目数据混合,依据 DBSCAN 算法进行聚类,最后丢弃不包含本项目数据的子簇,收集剩余子簇的跨项目数据. 实验选取 56 个项目数据,将 DBSCAN Filter 与 Peter Filter 和 Burak Filter 两种方法进行了对比. 结果表明,该方法在 AUC 和 G-measure 度量上明显优于 Burak Filter 和 Peter Filter 方法. 但是 DBSCAN Filter 对邻域密度阈值和阈值半径这两个参数的定义是敏感的,若选取不当将造成聚类质量下降. 且由于参数是全局唯一的,当簇之间的密度相差较大时,容易造成差别很大的聚类.

近年来,一些基于迁移学习的跨项目预测方法被提出,Ma 提出了 TNB (Transfer Naive Bayes) 跨项目缺陷预测方法,依据跨项目数据与本项目数据的相似度赋予跨项目数据不同的权重,然后在已加权的跨项目数据基础上构建预测模型[9]. Jing 指出源公司数据与目标公司数据度量属性通常是异构的,因此提出了本公司和跨公司数据统一度量表示方法,并将典型相关分析(CCA)算法运用到跨项目缺陷预测中[10].

以上的跨项目缺陷预测方法都是基于本项目没有任何历史数据的情况. 事实上,如果项目内存在少量带类标号的历史数据,这些数据不足以进行基于项目内缺陷预测,但可以充分利用这些数据进一步提升跨项目缺陷预测模型的性能.

Turhan 尝试将本项目历史数据与跨项目历史数据混合,使用混合数据进行缺陷预测模型的构建,实验结果表明当本项目历史数据极少时,混合数据构建的缺陷预测模型也能达到很好的性能[11]. Chen 提出了名为 DTB(Double Transfer Boosting)的跨项目缺陷预测方法[12]. 该方法利用少量本项目历史数据,通过缩小跨项目数据和本项目数据的数据分布差异,实现了较好的跨项目缺陷预测性能. 当本项目数据的极少时,DTB 方法构建的跨项目缺陷预测模型的性能甚至可以与本项目缺陷预测模型相媲美. Yu 等人针对 DTB 方法中会出现负迁移的问题提出了基于多源动态 TrAdaBoost 算法的跨项目缺陷预测方法[13]. 该方法利用少量本项目历史数据,通过从多个跨项目数据中学习多个弱缺陷预测模型,然后结合这些模型组合为一个强缺陷预测模型,依赖多个跨项目数据避免了负迁移,实现了较好的性能.

SSDBSCAN 算法是一种最新的半监督聚类算法,仅需要一个领域密度阈值参数就能自动发现簇结构,适用于没有全局密度参数的情况,且聚类过程不需要用户干预. 相比于 DBSCAN 算法,当簇之间的密度相差较大时,SSDBSCAN 算法也能依据自然簇的密度发现自然簇结构. 为了提升跨项目数据筛选能力,本文基于 SSDBSCAN 算法提出了跨项目数据筛选方法——SSDBSCAN Filter. 本文中提出的 SSDBSCAN Filter 同样能充分利用少量带有类标号的本项目数据,进一步去除跨项目中的不相关数据,

从而提升预测模型性能. 实验结果表明,相比前两种数据筛选方法(Burak Filter 和 DBSCAN Filter),SSDBSCAN Filter 显著提高了预测模型的性能.

接下来的内容共分为五个部分:第一部分是对算法部分的相关约定;第二个部分是对 SSDBSCAN Filter 的详细说明;第三部分为实验数据集、实验环境、性能度量指标以及实验步骤等的介绍;第四部分为实验结果分析;第五部分为对全文的总结和对未来的展望.

1 相关约定

假设跨项目历史数据集为 $D^{CP}=\{(x_1^{CP},c_1^{CP}),(x_2^{CP},c_2^{CP}),\cdots,(x_n^{CP},c_n^{CP})\}$,其中 x_i^{CP} 表示第 i 个实例,n 表示跨项目历史数据集中实例个数,c_i^{CP} 表示实例的类标号,且 $c_i^{CP}\in\{Y,N\}$,Y 表示实例有缺陷,N 表示实例无缺陷. 假设无类标号的本项目数据集为 $D^{WP}=\{x_1^{WP},x_2^{WP},\cdots,x_m^{WP}\}$,其中 x_i^{WP} 表示第 i 个实例,m 表示无类标号的本项目数据集中实例个数. 假设带类标号的本项目历史数据集为 $D^L=\{(x_1^L,c_1^L),(x_2^L,c_2^L),\cdots,(x_k^L,c_k^L)\}$,其中 x_i^L 表示第 i 个实例,k 表示带类标号的本项目历史数据集中实例个数,c_i^L 表示样本 x_i^L 的类标号,$c_i^L\in\{Y,N\}$. 假设总数据集为 D,$D=\{D^{CP},D^{WP},D^L\}$.

定义 1 ε 邻域. 若实例 $x\in D$,实例 x 的 ε 邻域是指以 x 为中心,以为 ε 半径的空间.

定义 2 邻域密度阈值. 若实例 $x\in D$,且为核心对象,则邻域密度阈值 MinPts 即为 x 的 ε 邻域所需包含的对象的最小数目.

定义 3 核心距离. 若实例 $x\in D$,x 的核心距离 $cDist(x)$ 为满足邻域密度 MinPts 的最小半径值.

定义 4 边界权值. 若实例 $p,q\in D$, $rDist(p,q)=\max\{cDist(p),cDist(q),dist(p,q)\}$,$rDist(p,q)$ 即为实例 p,q 边界权值,边界权值表示使两个实例 p、q 密度相连的最小半径值. 公式中 $dist(p,q)$ 表示 p 和 q 两个实例间的欧氏距离.

2 SSDBSCAN Filter

为了充分利用少量带有类标号的本项目数据,进一步提升缺陷预测模型的性能,我们提出了 SSDBSCAN Filter 用于跨项目数据筛选. SSDBSCAN Filter 首先将本项目数据和跨项目数据混合. 本项目数据中包含少量带有类标号的数据和大量不带有类标号的数据;然后开始聚类过程,每次迭代以带有类标号且未被访问的本项目数据为起点,扩充过程中不断更新邻域各点到簇中各实例的边界权值,每一次迭代时寻找邻域中边界权值最小实例添加进簇. 边界权值是由 SSDBSCAN 算法给出的实例连接到簇中的最短路径,详细定义参见定义 4. 由于文章篇幅原因,本文不对为何每次将边界权值最小实例添加进簇的原因进行详细描述,详见文献[14];当出现不同类标号的本项目数据,则停止簇的扩充,循环结束,以另一个未被访问过的带类标号本项目数据实例为

算法 1:SSDBSCAN Filter

输入:跨项目历史数据集 D^{CP}、无类标号本项目数据集 D^{WP}、带类标号本项目历史数据集 D^L、邻域密度阈值 *MinPts*

输出: 筛选后的跨项目数据集

```
1. 将 D^CP、D^WP 以及 D^L 混合于 D
2. 标记所有 D 中实例为 unvisited   3. for each unvisited 实例 p in D^L
4.      for each unvisited 实例 q in D
5.          key[q] = +∞ ;
6.      end for
7. key[p]←0 ;
8. 创建簇 C;
9. Q←D;
10. while Q≠∅
11. {
12.     找到 Q 中 key 值最小实例 r;
13.     if r∈D^L 且 c_p^L≠c_r^L
14.         输出簇 C 中与本项目数据类标号一致的跨项目数据实例,将簇中实例均标记为 visited,break;
15.     将 r 插入簇中,并将 r 从 Q 中移除;
16.     for each 实例 x in Q
17.         if rDist(r,x) < key[x]
18.             key[x]←rDist(r,x);
19.     end for
20. }
21. end for
```

起点继续另一个簇的扩充;随后,当所有带有类标号的本项目数据都包含在簇中,聚类过程终止;最后,聚类生成的子簇中包含大量数据分布相近的实例,收集子簇中的与本项目数据一致的跨项目数据,即得到筛选后的跨项目数据.

算法 1 显示了 SSDBSCAN filter 的伪代码. 该方法在首行将本项目与跨项目数据相混合,并将所有数据实例标记. 第 2 行将所有实例标记为 unvisited,每次循环开始时均选择未被标记实例. 接下来 3-23 行是簇的生成与扩充的过程. 第 3 ～8 行初始化各实例的边界权值 *key*,每次建立簇后,带有类标号的本项目数据因边界权值 *key* 最小首先被选中,从而作为簇扩充的起点. 簇的扩充工作从第 10 步的 while 循环开始,每次找到边界权值最小的实例加入簇中. 一旦出现类标号不同的本项目数据实例,即表明两个需要被分离的簇即将被连接,此时停止簇的扩充,收集簇中与本项目类标号一致的跨项目数据,并跳出循环过程,操作过程如 13 ～ 14 行所示,否则便将实例添加进簇. 第 16 ～21 行利用边界权值计算公式更新邻域内实例边界权值 *key*,以推动循环的继续进行. 整个过程重复执行,直到所有带有类标号的本项目数据实例都被添加到簇中,程序结束. 最后得到筛选后的跨项目数据,没有进入簇中的跨项目数据被作为噪声数据而舍弃.

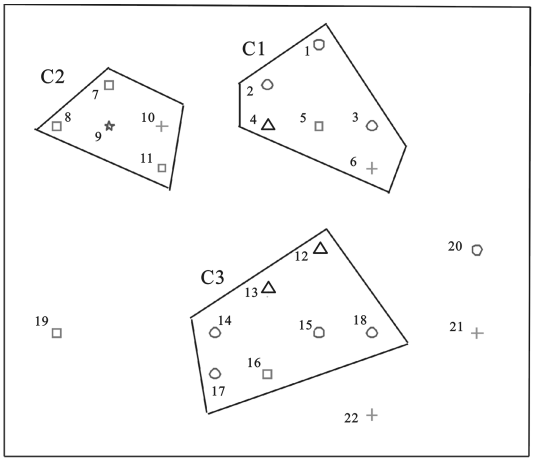

图 1　SSDBSCAN filter 结果图

SSDBSCAN filter 的聚类结果如图 1 所示,我们约定:“○”表示类标号为 Y 的跨项目数据,“□”表示类标号为 N 的跨项目数据,“△”表示类标号为 Y 的本项目数据,“☆”表示类标号为 N 的本项目数据,“ + ”表示无类标号的本项目数据. 依据 SSDBSCAN filter 进行聚类得到 3 个子簇:C1、C2、C3. 聚类过程由带类标号的本项目数据实例 4、9、12、13 开始. 聚类完成之后,子簇中可能包含的实例有相同类标号的本项目实例、相同类标号的跨项目实例、不同类标号的跨项目实例以及无类标号的本项目实例. 我们仅收集子簇中与本项目实例类标号相同的跨项目实例. 因此,簇 C1 的筛选结果保留跨项目数据实例 1、实例 2 和实例 3,实例 5 因类标号与本项目数据不同而被舍弃. 簇 C2、C3 的数据筛选同 C1,簇中实例的类标号应与带类标号的本项目数据一致,因此舍弃跨项目实例 5 和跨项目实例 16. 子簇之外的实例 19、20、21 和 22 为噪声数据. 最后筛选出的跨项目实例有:实例 1、实例 2、实例 3、实例 7、实例 8、实例 11、实例 14、实例 15、实例 17 和实例 18.

3　实验方法

3.1　实验数据与环境　为了验证本文方法 SSDBSCAN filter 在跨项目软件缺陷预测的性能,本文从 PROMISE 库[15]选取 15 个常用公开数据集,数据集详细信息如表 1 所示. 实验均在 Intel(R) Core(TM) i5-4210M CPU @ 2.60 GHz,内存 4 GB 的 PC 机上完成的. 采用 Java 语言的开发平台 Eclipse,并使用了 Weka 工具包.

表 1　实验数据

数据集	模块数	缺陷数	缺陷率/%	代码来源	数据集	模块数	缺陷数	缺陷率/%	代码来源
ant	125	20	16.0	Open-source	prop6	660	66	10.0	Proprietary
arc	234	27	11.5	Academic	redactor	176	27	15.3	Academic
camel	339	23	3.8	Open-source	synapse	157	16	10.2	Open-source
elearn	64	5	7.8	Academic	system	65	9	13.8	Open-source
jedit	272	90	33.1	Open-source	tomcat	858	77	9.0	Open-source
log4j	135	34	25.2	Open-source	xalan	723	110	15.2	Open-source
lucene	195	91	46.7	Open-source	xerces	162	77	47.5	Open-source
poi	237	141	59.5	Open-source					

3.2 实验度量标准 为了评价跨项目缺陷预测模型的性能,本文选择了四种常用的度量指标:*PD*、*PF*、*G-measure* 以及 *AUC*. 除 *PF* 之外,其他度量指标值越大表示模型性能越佳. 这些度量值首先要用到表2的混淆矩阵. 混淆矩阵中 *TP* 指被正确预测的有缺陷模块数;*TN* 指被正确预测的无缺陷模块数;*FP* 指被错误预测为有缺陷的无缺陷模块数;*FN* 指被错误预测为无缺陷的有缺陷模块数. 混淆矩阵定义如下:

表2 混淆矩阵

	预测为有缺陷	预测为无缺陷
有缺陷模块	TP(True Positive)	FN(False Positive)
无缺陷模块	FP(False Positive)	TN(True Positive)

1) *PD* (预测率)是指正确被预测为有缺陷的模块数与实际有缺陷的模块数的比率.

$$PD = \frac{TP}{TP + FN} \tag{1}$$

2) *PF*(误报率)是指正确被预测为有缺陷的模块数与预测为有缺陷的模块数的比率.

$$PF = \frac{TP}{TP + FP} \tag{2}$$

3) *G-measure* 是一种常用的综合评价预测率和误报率的度量方法. *G-measure* 值介于0～1之间,值越大则性能越佳.

$$G\text{-}measure = \frac{2 * PD * (1 - PF)}{PD + (1 - PF)} \tag{3}$$

4) *AUC* 值(Area Under ROC Curve)是常用的度量模型好坏的标准. 为了权衡误报率和预测率,常采用图形化方法 ROC(Receiver Operating Characteristic)曲线,*AUC* 值即为 *ROC* 曲线下方的面积,提供了评价模型平均性能的一般方法. *AUC* 度量值介于0.5～1.0之间,较大的 *AUC* 代表了较好的性能.

$$AUC = \int_0^1 PD\mathrm{d}PF \tag{4}$$

3.3 实验步骤 实验共采用15个 PROMISE 数据集,将其中1个作为本项目数据,剩余14个作为跨项目数据,共15种数据集组合. 为便于模型的性能度量,实验数据均带有类标号.

我们将本文方法与 Burack Filter 方法[5]和 DBSCAN Filter[9]对比. 对于 Burak Filter 方法,每个本项目实例从跨项目数据中选取 *K* 个最近邻实例组成训练数据集,与文献[3]一致,取 $K = 10$. 对于 DBSCAN Filter,相比于文献[7],由于使用数据集不同,本文中将参数调整为 MintPts = 5, radius = 5. 考虑到公平性,SSDBSCAN Filter 的参数 MintPts 同样设为5. 3种数据筛选方法均使用每种组合全部的本项目数据和跨项目数据. 由于 SSDBSCAN Filter 可以使用少量的带有类标号的本项目数据,因此将本项目数据进一步划分为两个部分:10%带类标号的本项目数据和90%不带类标号的本项目数据,应用于 SSDBSCAN Filter.

构建跨项目缺陷预测模型时,我们采用3种分类方法:朴素贝叶斯、随机森林和逻辑回归在 Weka 上实现. SSDBSCAN Filter 已使用10%的本项目数据的类标号,为公平起见,在对模型进行性能度量时,使用剩余的90%本项目数据. 实验重复进行20次,取性能度量的均值.

4 实验结果分析

表3、图2～图3显示了本文提出的方法与 Burak Filter 和 DBSCAN Filter,在 ant、arc、camel 等15个 PROMISE 数据集和3种预测模型上运行20次的4个度量指标的平均值.

4.1 PD 与 PF 结果分析 表3记录了 *PD* 与 *PF* 的实验结果. 为了使结果更加清晰,表格首先被划分为3个区域,从左到右依次为3种数据筛选方法:Burak Filter、DBSCAN Filter 和 SSDBSCAN Filter. 紧接着,实验使用了3种最常用的构建预测模型的方法:朴素贝叶斯(NB)、随机森林(RF),以及逻辑回归(LR). 在使用朴素贝叶斯构建预测模型的情况下,SSDBSCAN Filter 的 *PF* 值在大部分数据集上均有下降,相比于 Burak Filter 方法,*PF* 值下降幅度达到4.6%,此外过半数的数据集 *PD* 值提高;在使用随机森林构建预测模型的情况下,SSDBSCAN Filter 在13个数据集上 *PF* 值降低,10个数据集上 *PD* 值升高,

直接显示出性能的优越;在使用逻辑回归构建预测模型的情况下,约半数数据集显示,SSDBSCAN Filter 在降低 *PF* 值的同时提高了 *PD* 值. 综合表 3 可以看出,SSDBSCAN Filter 区域的最优值居多,且选择 3 种分类器的任意一种构建缺陷预测模型,15 组数据的 *PD*、*PF* 度量指标的平均结果均在 SSDBSCAN Filter 处取得最佳. 结果表明,相比于 Burak Filter 和 DBSCAN Filter,SSDBSCAN Filter 在 *PD*、*PF* 度量上显示出其优势.

表 3　跨项目缺陷预测 *PD* 与 *PF* 结果对比

序号	数据	Burak Filter						DBSCAN Filter						SSDBSCAN Filter					
		NB		RF		LR		NB		RF		LR		NB		RF		LR	
		PD	PF	PD	PF	PD	PF	PD	PF	PD	PF	PD	PF	PD	PF	PD	PF	PD	PF
1	ant	**0.829**	0.537	0.795	0.713	0.759	0.630	0.768	0.539	0.768	0.808	**0.786**	**0.580**	0.813	**0.485**	**0.821**	**0.663**	**0.786**	0.625
2	arc	0.829	0.593	0.881	0.739	0.871	0.664	0.824	0.594	0.881	0.815	0.881	0.625	**0.838**	**0.592**	**0.895**	**0.699**	**0.886**	**0.586**
3	camel	0.889	0.879	0.957	0.964	0.938	0.965	0.875	0.792	0.964	**0.964**	0.944	**0.965**	**0.902**	**0.704**	0.957	**0.964**	**0.944**	**0.965**
4	elearn	0.842	0.705	0.895	0.932	0.807	0.708	0.860	0.704	0.895	0.932	0.807	0.708	**0.877**	**0.703**	**0.912**	**0.931**	**0.842**	**0.507**
5	jedit	0.779	0.337	0.779	0.286	0.769	0.311	0.775	0.345	0.779	0.427	**0.811**	**0.252**	**0.783**	**0.329**	**0.795**	**0.272**	0.803	0.273
6	log4j	**0.860**	**0.342**	0.835	0.374	0.793	0.437	0.835	0.374	0.818	0.429	**0.802**	**0.409**	0.843	0.372	**0.860**	**0.342**	0.796	0.469
7	lucene	**0.657**	**0.374**	0.669	0.336	0.651	0.356	0.646	0.383	**0.785**	0.464	0.657	0.353	0.651	0.378	0.764	**0.333**	**0.696**	**0.338**
8	poi	**0.545**	**0.338**	**0.728**	**0.298**	0.634	0.413	0.521	0.354	0.663	0.346	0.648	0.396	0.531	0.352	**0.728**	0.305	**0.675**	**0.386**
9	prop6	**0.803**	0.531	0.847	0.771	0.886	0.870	0.801	**0.370**	0.643	**0.403**	0.880	0.856	0.796	0.517	**0.875**	0.653	**0.889**	**0.755**
10	redactor	0.747	0.372	0.880	0.510	**0.854**	**0.482**	**0.759**	**0.338**	0.867	0.545	0.848	0.483	**0.759**	**0.338**	**0.892**	**0.475**	**0.854**	0.515
11	synapse	0.759	0.508	0.787	**0.910**	**0.887**	**0.634**	0.773	0.438	0.887	**0.910**	0.851	0.775	**0.787**	**0.367**	**0.887**	**0.910**	0.858	0.575
12	system	**0.828**	**0.485**	0.745	0.754	0.810	0.579	**0.828**	**0.485**	**0.879**	0.819	0.845	**0.391**	**0.828**	**0.485**	0.862	**0.660**	**0.882**	0.485
13	tomcat	0.854	0.591	0.909	0.819	**0.911**	**0.764**	0.850	**0.577**	**0.916**	**0.818**	0.905	0.792	**0.858**	0.604	0.915	**0.818**	0.909	0.771
14	xalan	**0.820**	**0.565**	**0.858**	0.705	0.863	**0.661**	0.817	0.574	0.846	0.724	**0.865**	**0.661**	0.818	**0.565**	0.857	**0.688**	0.857	0.669
15	xerces	**0.579**	**0.425**	0.772	0.671	0.703	0.297	0.566	0.439	0.759	0.241	**0.717**	**0.283**	0.566	0.439	**0.779**	**0.221**	**0.717**	**0.283**
	average	0.775	0.505	0.822	0.652	0.809	0.585	0.767	0.487	0.823	0.643	0.816	0.569	**0.777**	**0.482**	**0.853**	**0.596**	**0.826**	**0.547**

4.2　G-measure 与 AUC 结果分析　图 2 ~图 3 使用盒图描述了本文方法与其他方法在 15 个数据集上的 G-measure 与 AUC 值结果对比. 盒图由 5 个数值点组成:最大值、上四分位数、中位数、下四分位数、最小值. 少数偏移整体的离群点单独绘出. 利用盒图可以清晰地识别出数据的偏差和尾重,以及异常值. 图 2 与图 3 将盒图划分为 3 个区域,从左到右依次为 3 种预测模型:朴素贝叶斯(NB),随机森林(RF)和逻辑回归(LR). 每个区域将 SSDBSCAN Filter 与其他两种算法进行对比.

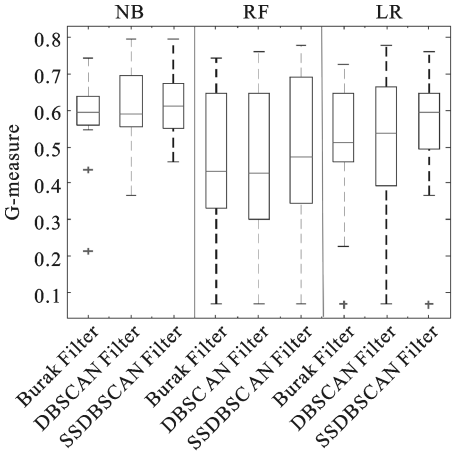

图 2　G-measure 结果对比

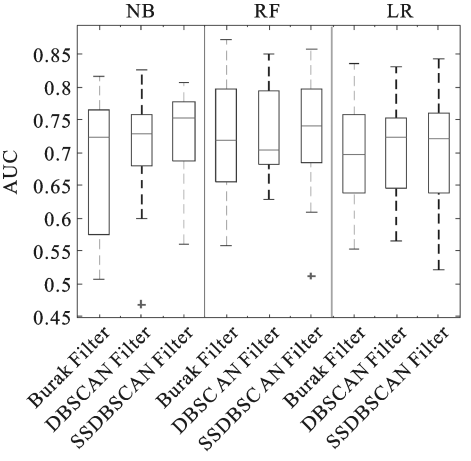

图 3　AUC 结果对比

从图 2 可以看出,对于朴素贝叶斯分类器,SSDBSCAN Filter 的 G-measure 值的中位数高于其他两种方法,最小值与最大值均有相应提升. Burak Filter 的数据最集中,但是相对而言 SSDBSCAN Filter 使得 G-measure 值有一段幅度的提升;对于随机森林分类器,SSDBSCAN Filter 的 G-measure 值的最小值和下

四分位数与其他两种方法基本持平,但从中位数、上小四分位数和最大值的角度来看,G-measure 性能较佳;对于逻辑回归分类器,SSDBSCAN Filter 的 G-measure 值分布集中,且中位数、最小值、下四分位数明显高于其他方法,显示出最佳的性能.

由图 3 可见,在朴素贝叶斯作为分类器的情况下,SSDBSCAN Filter 的 AUC 度量中位数明显提高,上下四分位数之间的差距缩小;对于随机森林,AUC 值的中位数明显高于其余两种方法,最小值也有一定的提升. 结合图 1 和图 2 可知,相比于 Burak Filter 和 DBSCAN Filter, SSDBSCAN Filter 总具有较好的 G-measure 与 AUC 性能.

综上所述,从选择的 4 种性能度量 PD、PF、G-measure 和 AUC 的结果来看,在大部分的数据集上,SSDBSCAN Filter 提高预测率的同时降低了误报率. 且使用不同的分类方法:朴素贝叶斯、随机森林或逻辑回归,筛选后的跨项目数据所构建的预测模型总具有较好的性能度量结果. 显然, SSDBSCAN Filter 最大限度地利用了本项目数据信息,较好地解决了大量不相关的跨项目数据影响预测的问题,显著提高了跨项目缺陷预测性能.

4.3 显著性检验 统计学研究中,p 值常用于反映两者间差别有无统计学意义. p 值越小,越有理由认为对比事物间存在差异. $p>0.05$ 称"不显著";$p\leqslant 0.05$ 称"显著",$p\leqslant 0.01$ 称"非常显著". 从显著性检验的角度,我们将 SSDBSCAN Filter 与 Burak Filter、DBSCAN Filter 分别进行 p 检验,得到以下结果:

表 4 p 值结果对比

p	Burak-Filter			DBSCAN Filter		
	NB	RF	LR	NB	RF	LR
G-measure	0.018	<0.001	0.001	0.010	0.001	0.020
AUC	0.004	0.010	0.010	0.005	0.003	0.010

由上表易知,结果中所有 p 值均 $\leqslant 0.05$,少部分 p 值 $\leqslant 0.01$,因此有理由认为本文提出的 SSDBSCAN Filter 与 Burak Filter、DBSCAN Filter 的差别具有显著的统计学意义.

5 结论

本文中针对跨项目数据中存在大量不相关数据的问题,提出了 SSDBSCAN Filter 用于跨项目数据筛选. 该方法的优势在于可利用少量带有类标号的本项目数据进一步提升筛选效果. 实验结果表明,相比于前人提出的 Burak Filter 和 DBSCAN Filter,利用本文中提出的 SSDBSCAN Filter 所筛选的跨项目数据,在朴素贝叶斯、随机森林、逻辑回归这 3 种分类器上所构建的跨项目缺陷预测模型的性能均有明显提高.

在后续工作中,我们将收集更多的数据来验证我们提出方法的通用性. 同时,将进一步考虑采用更为高效的半监督聚类算法对跨项目数据进行数据筛选.

6 参考文献

[1] 李勇. 结合欠抽样与集成的软件缺陷预测[J]. 计算机应用,2014,34(8):2291-2294.

[2] 王培,金聪,葛贺贺. 面向软件缺陷预测的互信息属性选择方法[J]. 计算机应用,2012,32(6):1738-1740.

[3] Liu M, Miao L, Zhang D. Two-stage cost-sensitive learning for software defect prediction[J]. IEEE Transactions on Reliability, 2014, 63(2): 676-686.

[4] Jing X Y, Ying S, Zhang Z W, et al. Dictionary learning based software defect prediction[C]// Proceedings of the 36th International Conference on Software Engineering. New York: ACM,2014: 414-423.

[5] Zimmermann T, Nagappan N, Gall H, et al. Cross-project defect prediction: a large scale experiment on Cross-project defect prediction: a large scale experiment on data vs. domain vs. process[C]// Proceedings of the the 7th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering. New York: ACM, 2009: 91-100.

[6] Turhan B, Menzies T, Bener A B, et al. On the relative value of cross-company and within-company data for defect prediction[J]. Empirical Software Engineering, 2009, 14(5): 540-578.

[7] Peters F, Menzies T, Marcus A. Better cross company defect prediction[C]// Mining Software Repositories (MSR). San Francisco: IEEE, 2013: 409-418.

[8] Kawata K, Amasaki S, Yokogawa T. Improving Relevancy Filter Methods for Cross-Project Defect Prediction[M]// Applied Computing & Information Technology. Springer International Publishing, 2016: 1-12.

[9] Ma Y, Luo G, Zeng X, et al. Transfer learning for cross-company software defect prediction[J]. Information and Software Technology, 2012, 54(3): 248-256.

[10] Jing X, Wu F, Dong X, et al. Heterogeneous cross-company defect prediction by unified metric representation and cca-based transfer learning[C]// Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering. New York: ACM, 2015: 496-507.

[11] Turhan B, MiSiRLi A T, Bener A. Empirical evaluation of the effects of mixed project data on learning defect predictors [J]. Information and Software Technology, 2013, 55(6): 1101-1118.

[12] Chen L, Fang B, Shang Z, et al. Negative samples reduction in cross-company software defects prediction[J]. Information and Software Technology, 2015, 62: 67-77.

[13] Yu Xiao, Liu Jin, Fu Mandi, et al. A Multi-Source TrAdaBoost Approach for Cross-Company Defect Prediction[C]// The 28th International Conference on Software Engineering & Knowledge Engineering. San Francisco Bay, California, USA, 2016: 237-242.

[14] Lelis L, Sander J. Semi-supervised density-based clustering[C]// 2009 Ninth IEEE International Conference on Data Mining. Miami: IEEE, 2009: 842-847.

[15] Boetticher G, Menzies T, Ostrand T. PROMISE Repository of empirical software engineering data[J]. West Virginia University, Department of Computer Science, 2007.

(责任编辑　江津)

(上接第 541 页)

推论 2.2 的证明　在定理 2.1 中取 $S(x)=x, U=R^m$,则满足定理(2.1)中条件,从而结论成立.

最后,考虑模型(8)在 $p=1$ 时的情形,即

$$Y_n=\varphi(Y_{n-1})+\varepsilon_n\omega(Y_{n-1}) \tag{8}$$

推论 2.3　设模型(2.5)满足

(i) $\{\varepsilon_n, n=1,2,\cdots\}$关于一维 Lebesgue 测度有绝对连续部分,

(ii) $\varphi(x)$是局部有界的函数,$\omega(x)>0$,且 $\omega(x)$和$\frac{1}{\omega(x)}$都是局部有界函数,

那么定理 2.1 的结论成立.

推论 2.3 的证明　在定理 2.1 中令 $U=R\times(0,\infty)$, $h(z,y)=z_1+z_2y$, $Z=(z_1,z_2)\in U, y\in R$,映射 $T:R\to U$, $T(x)=(\varphi(x),\omega(x))$,记 $\Phi(x,y)=h(T(x),y)$,易验证 $\Phi(x,y)$满足定理 2.1 的条件,故推论 2.3 的结论成立.

3　参考文献

[1] An H Z, Chen M. Nonlinear time series analysis[M]. Shanghai:Shanghai Science and Technology Press,1998.

[2] An H Z, Chen S G. A note on the ergodicity of nonlinear autoregressive models[J]. Statist Probab Lett, 1997,34:365-372.

[3] Brockwell P J, Povis R A. Time series analysis: theory and methods[M]. New York: Springer-verlag, 1998.

[4] Fonseca G,Tweedie R L. Stationary measure for mon-irreducible non-continuous markovchains with time series applications [J]. Statist Sini,2001,12(1):651-660.

[5] 盛昭瀚 . 非线性时间序列模型的稳定性分析遍历性理论与应用[M]. 北京:科学出版社,1993.

[6] 张韧, 张绍义. 非线性自回归序列的平稳解及其矩阵的存在性[J]. 数学物理学报,2013(2): 260-266.

(责任编辑　赵燕)