Volume 3, Issue 8, August 2018 ISSN (online): 2456-0006

International Journal of Science Technology Management and Research

Available online at: www.ijstmr.com

Automatic Summarization of Bug Reports and Bug Triage classification

Prajakta Kokate.
PG Student,Dept.of Compute
L.G.N.Sapkal COE,Anjaneri,Nashik

N.R.Wankhade
Asst.Prof ,Dept. of Computer
L.G.N.Sapkal COE,Anjaneri,Nashik

Abstract- In software industries bugs are very critical aspects and it cannot be avoided. Bug triage is a necessary step in an software company, in this step a correct developer should be assigned for fixing the bug. To handle bag triage manually is very expensive and time consuming process. In manually bag triage human perform the bug triage. So existing techniques uses text classification. In this techniques automatic bug triage is performed. But still there is an problem of large data i.e so there is a need the data should be reduced which will increase the quality of the data. At the same time, we used instance selection and feature selection methods to perform data reduction. The sequence for applying instance selection and feature selection should be predicted, and to know the order we extract the attributes from the bug data sets. Even the task of domain specific bugs is performed and accurate developer assignment is performed in this system.

Keywords— Mining Bug repositories, bug data reduction, attribute extraction, instance and feature selection, Developer Recommendation.

I. INTRODUCTION

For the storage purpose software repositories mainly used in current software development. In that software repository it consist of source code, emails, bugs and specifications. Human triagers are used to manually perform the bug triage which is very expensive and even takes long span of time. Software projects in an company uses bug repositories and this repositories consist of bug data which helps developers to handle bug updates according to the status of bug fixing. There are two challenges which are associated with the bug data that may affect the effectual use of bug repositories they are huge scale and the low quality of data. Two typical characteristics of low-quality bugs are noise and redundancy. Both of these characteristics affect the bug triage process. So in this paper we mainly focus two major issues are the large data and low quality. This two issue need to be solved to facilitate the bug handling process. In authors system they combine existing techniques of instance selection and feature selection to simultaneously reduce the bug dimension and the word dimension which improves the quality of the bug data. When the bug reports are reduced then the accurate developer assignment is performed using the weight estimations as a result it gives proper developer is assigned to a bug which increases the quality of the bug triage. And even domain specific task is performed in this system which helps to easily perform the accurate developer assignment. In domain specific bugs are categorized as per the domain. This is performed using different techniques.

II. RELATED WORK

Sandusky et al. [1] outline a bug report network to observe the need among bug reports. In addition to studying relationships between bug reports, Hong et al. [2] construct a developer social network to study the relationship among developers based on the bug data in Mozilla project. This developer social network is useful to recognize the developer community and the project evolution. By mapping bug priorities to developers, Xuan et al. [3] recognize the developer prioritization in open source bug repositories. The developer prioritization can differentiate developers and assist work in software maintenance. To study the excellence of bug data, Zimmermann et al. [4] design questionnaires to developers and users in three open source projects. Based on the study of questionnaires, they differentiate what makes a good bug report and instruct a classifier to recognize whether the quality of a bug report should be improved. Duplicate bug reports weaken the quality of bug data by delaying the cost of managing bugs. To identify duplicate bug reports, Wang et al. [5] propose a natural language processing approach by matching the execution information; Sun et al. [6] suggested a duplicate bug detection approach by optimizing a retrieval function on multiple features. Cubrani and Murphy [7] first

IJSTMR © 2018| All Rights Reserved

plan the crisis of automatic bug triage to reduce the cost of manual bug triage. They used text classification methods to identify associated developers. Anvik and Murphy [8] expand

above effort to decrease the task of bug triage by creating development-oriented recommenders. Jeong et al. [9] searched that over 37 percent of bug reports have been reassigned in manual bug triage. They put forward a tossing graph technique to decrease reassignment in bug triage. In software engineering, defect prediction is a kind of work on software metrics. To recover the data quality, Gao et al. [10] inspect the methods on feature selection to hold imbalanced defect data. Shivaji et al. [11] advises a framework to study multiple feature selection algorithms and eliminate noise features in classification-based defect prediction. In addition to feature selection in defect prediction, Kim et al. [12] present how to calculate the noise resistance in defect prediction and how to detect noise data. Additionally, Bishnu and Bhattacherjee [13] practice the defect data with quad tree based k-means clustering to help defect prediction.

III. PROPOSED SYSTEM

In software industry when bugs arise in software then bug triage is used to fix that bug. In this process accurate developer is assigned for fixing the particular bug. But manual bug triage process is very lengthy and expensive process. So to avoid time and cost in manual bug triage, automatic bug triage is used in which text classification techniques are used. The problem which is addressed for this is the large bug dataset. And the large bug dataset affects the quality of the bug datasets. So to reduce the bug dataset we use feature selection and instance selection techniques as shown in block diagram. This techniques reduces the bug data in both bug and word dimensions. And even we want to know the order of applying the instance selection and feature selection for this the attributes of the historical bug datasets are extracted. This gives us the reduced and quality bug dataset. Even would like to built the domain specific system for which the domain relevance class labels are generated. And the necessary thing which need to be improved is the accuracy of the developer assignment. Correct developer assignment would increase the quality of the bug triage which is the main aim of the system. So weight estimation is performed for assigning the accurate developer.

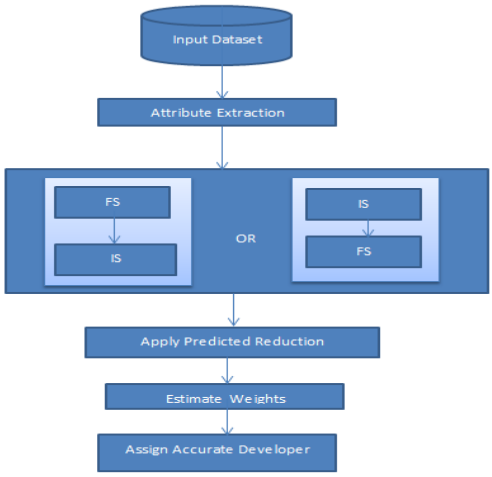

Fig -1: Proposed System

IV. MATHEMATICAL MODEL

Let the system be described by S,

S={D, AE, BR, FS,IS, NB, CR }

Where

S: is a System.

D: Set of Dataset.

AE: Attributes Extraction .

BR: Bigdata Reduction

FS: Feature Selection .

IS: Instance Selection .

NB: Naive Bayes .

CR: Classified Results

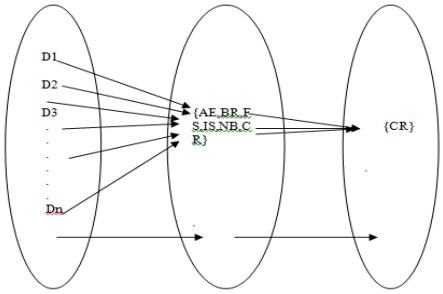

Fig 2: Vein Diagram

AE: Attributes Extraction .

BR: Bigdata Reduction

FS: Feature Selection .

IS: Instance Selection .

NB: Naive Bayes .

CR: Classified Results.

State diagram

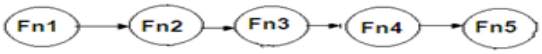

Fig 3:State diagram

Fn1: Attributes Extraction .

Fn2: Bigdata Reduction .

Fn3: Feature Selection

Fn4: Instance Selection

Fn5: Classification

V. ALGORITHM

Following are the steps of the algorithm:

Step 1.Input Dataset

Step 2.Extract the attributes from the dataset.

Step 3.To perform the data reduction use feature selection and instance selection

Step 4.To predict the order of applying IS and FS use binary classifier

Step 5.Apply the predicted order.

Step 6.Perform the weight estimation

Step 7.Assign accurate developer according to weights.

IV. RESULTS

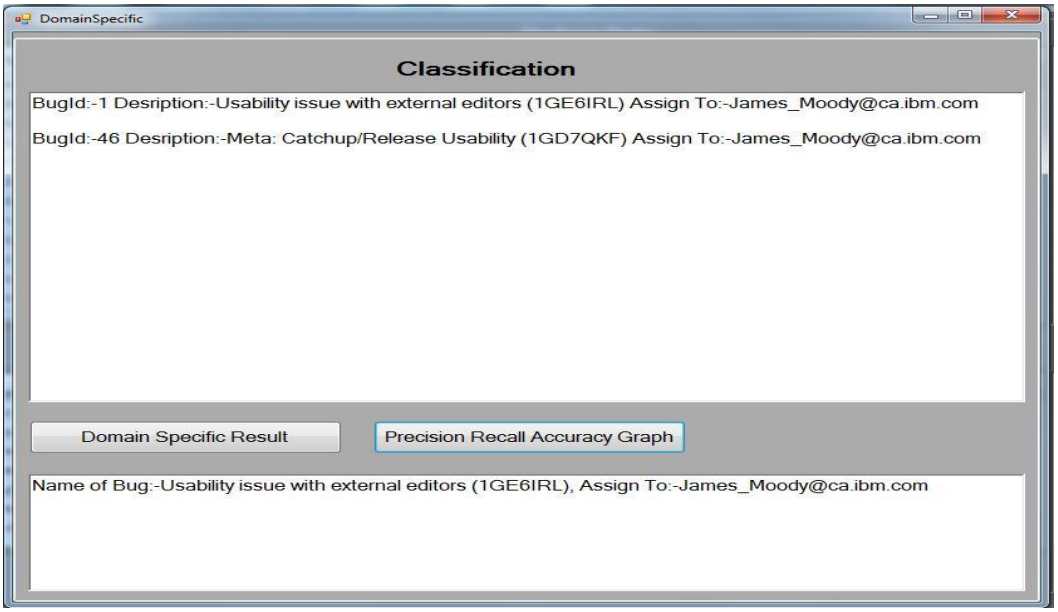

Fig-4: classification

Precision recall Accuracy graph as shown in fig

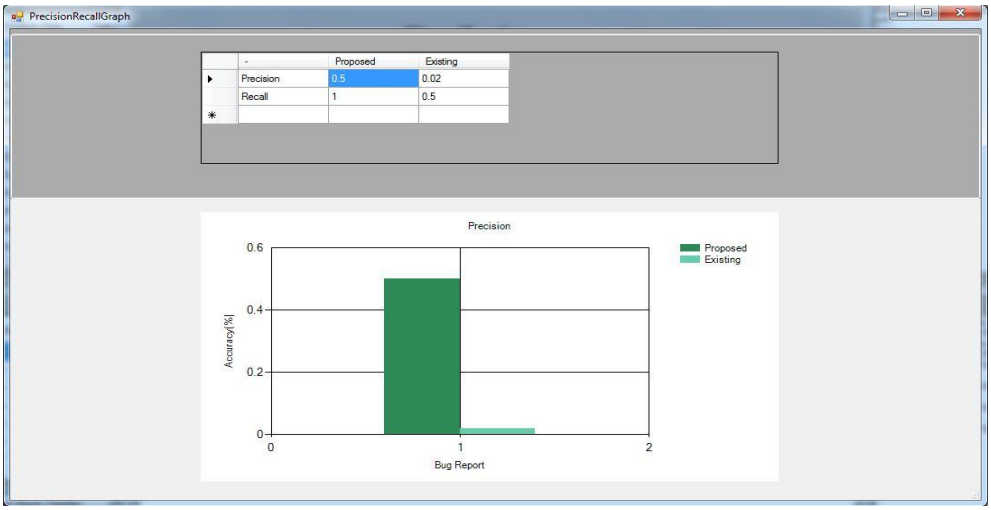

Fig-4: classification

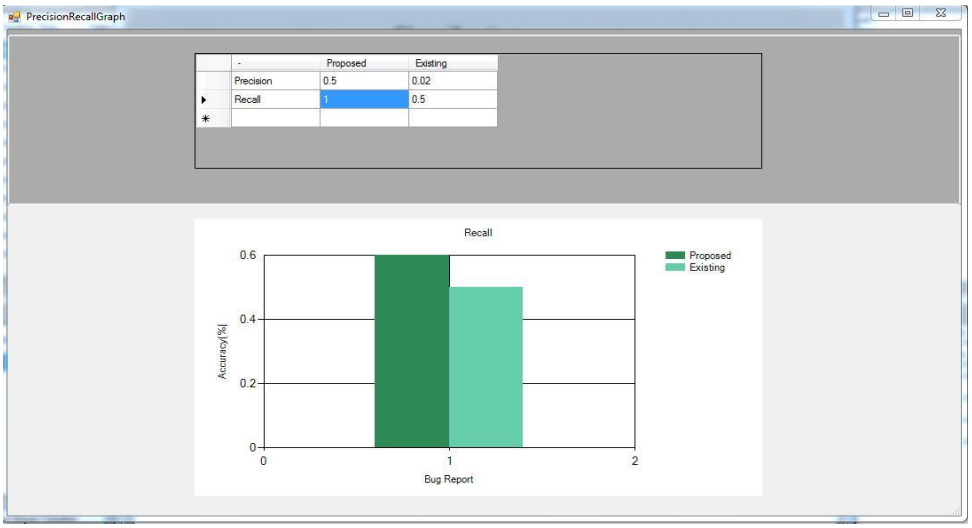

Fig-4: classification

CONCLUSION

In this paper, we use method instance selection and feature selection at the same time, which decrease the degree of the data and increases the value of bug data. And also the prediction order is known by extracting the attributes of the bug data sets. They perform the experiment of the data reduction for bug triage in bug repositories of two large open source projects such as Eclipse and Mozilla. They provide an technique on data processing which forms the reduced and high-quality bug data in software development and maintenance.

REFERENCES

1. R. J. SANDUSKY, L. GASSER, AND G. RIPOCHE, "BUG REPORT NETWORKS:VARIETIES, STRATEGIES, AND IMPACTS IN AN F/OSS DEVELOPMENT COMMUNITY," IN PROC. 1ST INTL. WORKSHOP MINING SOFTW. REPOSITORIES, MAY 2004, PP. 80–84.
2. Q. Hong, S. Kim, S. C. Cheung, and C. Bird, "Understanding a developer social network and its evolution," in Proc. 27th IEEEInt. Conf. Softw. Maintenance, Sep. 2011, pp. 323–332.
3. J. Xuan, H. Jiang, Z. Ren, and W. Zou, "Developer prioritization in bug repositories," in Proc. 34th Int. Conf. Softw. Eng., 2012, pp. 25–35.
4. T. Zimmermann, R. Premraj, N. Bettenburg, S. Just, A. Schr€oter,and C. Weiss, "What makes a good bug report?" IEEE Trans.Softw. Eng., vol. 36, no. 5, pp. 618–643, Oct. 2010.

   X. Wang, L. Zhang, T. Xie, J. Anvik, and J. Sun, "An approach to detecting duplicate bug reports using natural language and execution information," in Proc. 30th Int. Conf. Softw. Eng., May 2008,pp. 461–470.
5. C. Sun, D. Lo, S. Khoo, and J. Jiang, "Towards more accurate retrieval of duplicate bug reports," in Proc. 26th IEEE/ACM Int.Conf. Automated Softw. Eng., 2011, pp. 253–262.
6. D. _Cubrani_c and G. C. Murphy, "Automatic bug triage using text categorization," in Proc. 16th Int. Conf. Softw. Eng. Knowl. Eng.,Jun. 2004, pp. 92–97.
7. J. Anvik, L. Hiew, and G. C. Murphy, "Who should fix this bug?" in Proc. 28th Int. Conf. Softw. Eng., May 2006, pp. 361–370.
8. G. Jeong, S. Kim, and T. Zimmermann, "Improving bug triage with tossing graphs," in Proc. Joint Meeting 12th Eur. Softw. Eng.Conf. 17th ACM SIGSOFT Symp. Found. Softw. Eng., Aug. 2009,pp. 111–120.
9. K. Gao, T. M. Khoshgoftaar, and A. Napolitano, "Impact of data sampling on stability of feature selection for software measurement data," in Proc. 23rd IEEE Int. Conf. Tools Artif. Intell., Nov.2011, pp. 1004–1011.
10. S. Shivaji, E. J. Whitehead, Jr., R. Akella, and S. Kim, "Reducing features to improve code change based bug prediction," IEEETrans. Soft. Eng., vol. 39, no. 4, pp. 552–569, Apr. 2013.
11. S. Kim, H. Zhang, R. Wu, and L. Gong, "Dealing with noise in defect prediction," in Proc. 32nd ACM/IEEE Int. Conf. Softw. Eng.,May 2010, pp. 481–490.
12. P. S. Bishnu and V. Bhattacherjee, "Software fault prediction using quad tree-based k-means clustering algorithm," IEEE Trans. Knowl. Data Eng., vol. 24, no. 6, pp. 1146–1150, Jun. 2012.