International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 02 Issue: 04 | July-2015 www.irjet.net p-ISSN: 2395-0072

Bug Triage with Bug Data Reduction

Pankaj Gakare[1], Yogita Dhole[2], Sara Anjum[3]

[1] Asst. Professor, Department of E & TC, DMIETR, Maharashtra, India
[2] M. Tech. Student, Department of CSE, AMR Institute of Technology, Adilabad, India
[3] Asst. Professor, Department of CSE, AMR Institute of Technology, Adilabad, India

---

Abstract - ***The process of fixing bug is bug triage, which aims to correctly assign a developer to a new bug. Software companies spend most of their cost in dealing with these bugs. To reduce time and cost of bug triaging, we present an automatic approach to predict a developer with relevant experience to solve the new coming report. In proposed approach we are doing data reduction on bug data set which will reduce the scale of the data as well as increase the quality of the data. We are using instance selection and feature selection simultaneously with historical bug data. We have added a new module here which will describe the status of the bug like whether it assigned to any developer or not and it is rectified or not.***

Key Words: *Bug, Bug triage, data reduction, Instance selection, Data Mining.*

1. INTRODUCTION

A bug repository plays an important role in managing software bugs. Many open source software projects have an open bug repository that allows both developers and users to submit defects or issues in the software, suggest possible enhancements, and comment on existing bug reports.

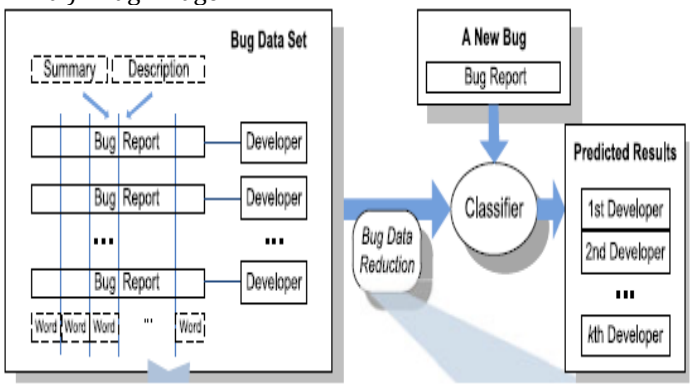
For open source large-scale software projects, the number of daily bugs is so large which makes the triaging process very difficult and challenging [2]. Software companies spend over 45 percent of cost in fixing bugs .There are two challenges related to bug data that may affect the effective use of bug repositories in software development tasks, namely the large scale and the low quality. In a bug repository, a bug is maintained as a bug report, which records the textual description of reproducing the bug and updates according to the status of bug fixing [1].

Primary contribution of this paper is as follow:
Here in this paper we are using feature selection and instance selection with historical data for reducing the bug data in bug repository so that we get quality data as well as low scale data. We are also adding a graph module for representing the bug report's.

Section II describes the architecture of the proposed system. The details of instance selection, feature selection, historical data use and graph module is given in section III and the summary is concluded in section IV.

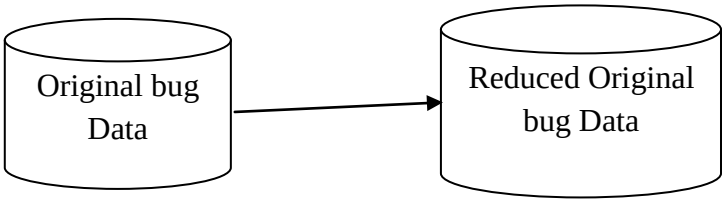
2. ARCHITECTURE

a.) Bug Triage



Aim of bug triage is to assign a developer for bug fixing. Once a developer is assigned to a new bug report he will fix the bug or try to rectify it. He will give the status related to bug whether it is rectified or not [1].

b) Data Reduction
Here we are reducing the bug data by using instance and feature selection so that we get low scale as well as quality data.



3. INSTANCE SELECTION

- Instance selection methods associated with data mining tasks such as classification and clustering
  - It's a nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in

data. Choosing a subset of data to achieve the original purpose of a data mining application as if the whole data is used.

- The ideal outcome of instance selection is model independent.

$$P(M_s) \cong P(M_w)$$

Evaluation measures:

- Direct Measure:
  - Keep as much resemblance as possible between the selected data and the original data.
  - Ex) Entropy, moments, and histograms.
- Indirect Measure
  - For example, a classifier can be used to check whether instance selection results in better, equal, or worse predictive accuracy.
  - Conventional evaluation methods in sampling, classification, and clustering can be used in assessing the performance of instance selection.
  - Ex) Precision, recall.

3.1 FEATURE SELECTION

- It select a minimum set of features such that the probability distribution of different classes given the values for those features is as close as possible to the original distribution given the values of all features [1].
- Reduce # of patterns in the patterns, easier to understand.
- Create new attributes that can capture the important information in a data set much more efficiently than the original attributes.
- Use the smallest representation which is enough to solve the task.

Heuristic methods:

- Step-wise forward selection
- Step-wise backward elimination
- Create new attributes that can capture the important information in a data set much more efficiently than the original attributes
- Three general methodologies:
  - Feature extraction
  - domain-specific
  - Mapping data to new space (see: data reduction)

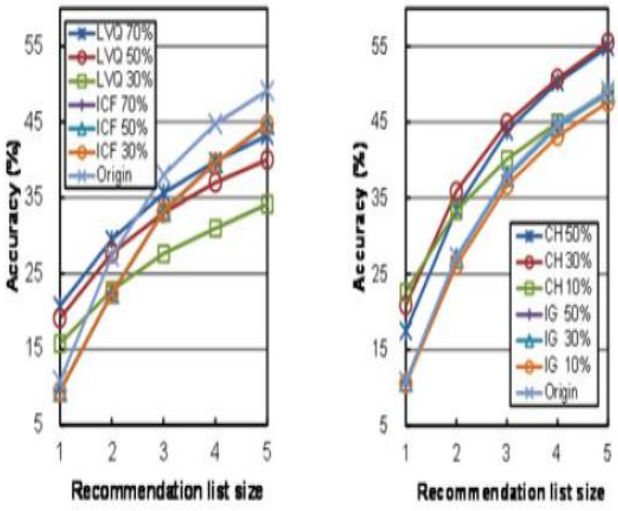


Figure:1: Instance selection in Mozilla and feature selection in Mozilla

3.2 GRAPH MODULE

This module show's four part's as follow:

1) Firstly it will show how many bugs are not assigned to any developer. It will give complete status about the bugs to the admin so that he will come to know which bugs are not assigned yet.

2) Secondly it will show how many bugs are not assigned to any developer. It will give complete status about the bugs to the admin so that he will come to know which bugs are assigned.

3) Thirdly it will show how many bugs are rectified by the developer's. It will give complete status about the bugs to the admin so that he will come to know which bugs are rectified completely.

4) Fourthly it will show how many bugs are not rectified by the developer's. It will give complete status about the bugs to the admin so that he will come to know which bugs are not rectified yet.

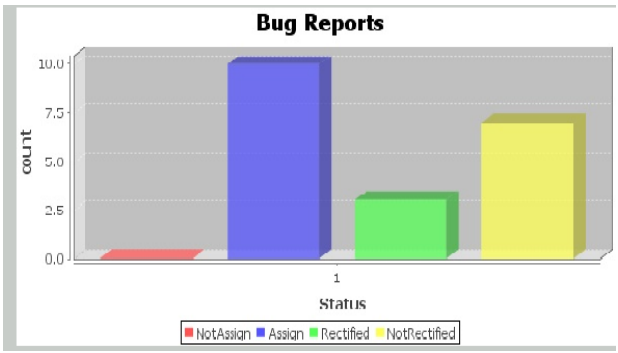


Figure 2: Bug Reports

3.3 Historical data:

This is also used for reducing the bug data. Here we will enter the date for the the last accessed bug's and the data which we get we will use it for data reduction [3].

4. CONCLUSION

In this paper we have focused on reducing bug data set in order to have less scale of data and quality data. For that we have used feature selection and instance selection techniques of data mining as well as we have used historical data. Our experimental results showed that this data reduction technique will give quality data as well as it will reduce the data scale. We have added new module in this paper than the earlier which will give various details related to the bugs to administrator in graphical format.

In future work, we plan on improving the results of data reduction more in bug triage to explore how to prepare a high quality bug data set.

REFERENCES

[1]Jifeng Xuan, He Jiang, Yan Hu, Zhilei Ren, Weiqin Zou, Zhongxuan Luo, and Xindong Wu," Towards Effective Bug Triage with Software Data Reduction Techniques" ieee transactions on knowledge and data engineering, vol. 27, no. 1, january 2015

[2] Mamdouh Alenezi and Kenneth Magel, Shadi Banitaan "Efficient Bug Triaging Using Text Mining" © 2013 academy publisher

[3]Francisco Servant "Supporting Bug Investigation using History Analysis" 978-1-4799-0215-6/13 c 2013 IEEE

[4]Pamela Bhattacharya, Iulian Neamtiu, Christian R. Shelton, "Automated, Highly- Accurate, Bug Assignment Using Machine Learning and Tossing Graphs", May 2, 2012

[5] K. Balog, L. Azzopardi, and M. de Rijke, "Formal models for expert finding in enterprise corpora," in Proc. 29th Annu. Int. ACM SIGIR Conf. Res. Develop. Inform. Retrieval, Aug. 2006, pp. 43–50.

[6] 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.

[7] H. Brighton and C. Mellish, "Advances in instance selection for instance-based learning algorithms," Data Mining Knowl. Discovery, vol. 6, no. 2, pp. 153–172, Apr. 2002. A. K. Uysal and S. Gunal, "A novel probabilistic feature selection method for text classification," *Knowledge-Based Systems*, vol. 36, no. 0, pp. 226–235, 2012.

[8] 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.

[9] A. Lamkanfi, S. Demeyer, E. Giger, and B. Goethals, "Predicting the severity of a reported bug," in Proc. 7th IEEE Working Conf. Mining Softw. Repositories, May 2010, pp. 1–10.

