EFFECTIVE BUG DATA REDUCTION TECHNIQUE INSTANCE SELECTION & FEATURE SELECTION

Abdul Kadir[1] ,Jayprakash Chaurasia[2] ,Rohit Gujar[3] ,Lalit Harkhani [4],Rajashree Gadhave[5]
Student[1,2,3,4],Faculty[5]
Pillai HOC College of Engineering & Technology,
Rasayani, Raigad, INDIA.

Abstract: Eclipse and Firefox are open source projects which consist of open source bug repositories. Bugs reported to these repositories are mostly non-technical and correct class cannot be assigned to these bugs. Triaging of bugs is a tedious task and it also consumes large amount of time. Some developers are connoisseurs of GUI while others in Java applications. If a particular bug is assigned to respective developer, then good amount of time could be saved .But, assigning correct bug to respective developer is quite difficult for triaged without knowing the actual class, to which the bug belongs. Here in this paper, we have surveyed the enlisted reference papers for bug triaging in which the various triaged bug reports are assigned to the respective developers using data reduction techniques and also there are two important algorithms which are namely Instance selection and Feature selection that will help to classify bugs.

IndexTerms - Text mining, textual classification, software repositories, open source software projects, triaging, feature extraction.

I. INTRODUCTION

Data mining is the process of extracting useful information through data analysis. It's also known as knowledge discovery. Useful knowledge can be obtained as a result of data mining which can be used to cut costs, increase revenues or both. There have been two types of data, categorical and numerical, used for mining purpose like integer, decimal, float, char, varchar2 etc.It's not possible to do data mining on data that is not numerical or categorical. Most of the time, enterprise data found is non numerical and non-categorical. For the success of business, extracting knowledge from this unstructured data can be difficult hence it has to be processed using text mining techniques, so that it can be processed by data mining algorithms and techniques. Some techniques used for text mining are Information extraction, information retrieval and natural language processing techniques.Process of assigning items in a collection to predefined classes or categories is called Classification. The basic goal of classification is the accurate prediction of target class for each case in data. e.g., applications of loan can be classified into high, medium or low risks on the basis of classification model. Status monitoring can be applied to determine when the bug arrives to developer and when developer actually starts working on it and by what time the developer will be done with fixing this particular bug.This will help to monitor not just status of bug and the condition through which it is going , but it will also help to earn some fairness for the developer as his credentials will be generated based on this approach.

II. MINING SOFTWARE REPOSITORIES

Understanding constantly evolving software systems is a very tiring task. History of software systems have been maintained in software repositories. Evolution of software systems can be documented by artifacts called Software repositories which often contain data from years of development of a software project.
Examples of repositories of software's are:
Runtime Repositories: Large organized storages which contain development logs about application usage on deployment sites and useful information of its execution are one of many examples of runtime repositories.
Historical Repositories: Bug databases, source code repositories and archived communication logs are some examples of historical repositories.
Code Repositories: Examples of code repositories are Google code and codeforge.net which store source code of various open source projects.
A process of software repository analysis that does discover significant and interesting information hidden in these repositories is known as MSR. It does process and analyze he huge software engineering data to detect interesting patterns in this data. It's an open field as in what can be mined and what can be learned from practice. Mining can be done on all kind of software repositories.

III. PROBLEM FORMULATION

Triaging of bugs and assigning a developer to fix them is a daunting and time consuming task. Developers, generally, happen to be expert in some certain area. For example few developers could be expert in GUI while some could be in pure java functionality etc; hence assigning a specific bug to appropriate developer could save time. It can also help to maintain the interest level of the developers by assigning bugs according to their interest. It's not an easy task to assign right bug to the right developer for triager without knowing the actual class a bug belongs to. Technique that classifies open source software bugs using the summary and description of the bugs provided by the bug reporters is proposed in this research.

IV. LITERATURE SURVEY

Following data represents already implemented techniques for software bugs classification:

1) **Towards effective bug triage with software data reduction techniques** JifengXuan , He Jiang , Yan Hu , ZhileiRen , WeiqinZou , Xindong Wu

This paper highlights the two important algorithms, Instance and Feature algorithms that deals with mining repositories. These algorithms are combination of multiple algorithms that we can use for mining process . Naïve Bayes algorithm will be used for performing text classification.

2) Mapping bug reports to closely connected files : A ranking model , a fine- grained feature evaluation

XinYe ,RazvanBunescu , Chang Liu
This paper let us know that the source files happen to be in large contain compared to actual bug files present in it. Bug files could be way much smaller compared to whole source files. Hence, ranking approach has been implemented to source files to rank them according to their relevance.

3) Micheal W. Godfrey, Olga Baysal and Robin Cohen developed a model for automatic assignment of bugs to developers for fixation using vector space model .

In this paper, the authors have proposed a specimen of the intelligent system that instinctively conducts the bug assignment. They have employed the vector space model to infer information about the developer's expertise from the history of the previously fixed bugs. The vector model is used to retrieve the title and the description from the report to build a vector which later can be used to find similar reports by mining the data in the bug repository. In order to create an efficient bug triage model, the authors conducted a survey wherein they collected a feedback from the developers regarding their previous bug fixing experience, their satisfaction with the bug assignment, whether they were successful and confident in handling bugs in the past, etc. The overall information provided them the initial estimates for the proposed model. This in turn helped them to implement the specimen model and test it within a software team working on the maintenance activities.

4) Lei Xu , Lian Yu , Jingtao Zhao , Changzu Kong , Huihui Zhang put forth a technique which used data mining techniques to automatically classify bugs of web- based applications by predicting their bug type.

The authors have put forth that the debug strategy acquaints us with the erroneous part of the source code. Once the errors are found then it is very easy for the developers to fix them. The determined association rules help to predict files that usually change together such as functions and variables.

V. PROBLEM SOLUTION

To obtain the solution for the bugs that are generated, we can implement our two important algorithms, instance selection and feature selection. These algorithms will help to reduce the time amount taken for resolving the generated bugs.

VI. INPUT DATA

The data of software's such as Eclipse and Mozilla Firefox happen to be obtained from bugzilla -an open bug repository
Datasets of bug reports are obtained. This data is divided into training and testing groups, experiments are performed on different set of data from these groups. In this Project, we are using our Bug Repository and we will also make our own compiler for taking input. First we have to compile file and take input from there.

VII. PRE-PROCESSING

Data preprocessing is the most important step of data mining . Raw data can be obtained from bug repositories which cannot be directly used for training the classification algorithm. Hence, the data needs to be pre-processed to make it functional for training purpose. Data pre-processing is a monotonous step of data mining and important as well. Stop-words dictionary and regular expression rules are used to filter redundant and irrelevant words and filter the punctuations respectively. Stemming algorithm is used to stem the vocabulary.

VIII. FEATURE SELECTION

After implementing bunch of words approach on data, the result that is obtained has very large dimensionality. Many of these dimensions are not related to text categorization and thus in turn result in reducing the attainment of the classifier. Feature selection is the process that helps to reduce the intensity of the obtained vocabulary. In this technique, best k terms out of the whole vocabulary are elected which contribute to accuracy and efficiency.

IX. There are a number of feature selection techniques such as Document Frequency (DF),Chi-Square Testing, Information Gain (IG), Term Frequency Inverse Document Frequency (TFIDF). In this research, we will use feature selection algorithm.

Algorithm and it's brief explanation :

Applied to bugs obtained recently:
IG, CH, SU,RF.
DEMONSTRATION:
Input : Create new department and new bug pattern.
Output: Generate new department in database and define new bug pattern in new department **.**
D(I) : Instant selection department
D(F) : Feature Selection department
P(f) : bug PatternSteps are as followed:
1.If department D (I) not exist;
2.Go to feature selection
3.Create D (F) and send to database;
4.Define bug pattern p(f)
5.P(f) should be save in new D (F)
6.Terminate D(F);
7.Repeat 2 to 6 if again new department has to come.
8.end.

X. INSTANCE SELECTION

Instance selection uses a methodology which is used for reducing the dimension of vocabulary obtained after applying to bunch of words. As most of the dimensions are related to our pre-defined bugs and result in reducing the performance of the classifier, hence to decrease the time, the process of Instance selection is used which chooses the best k terms out of the complete vocabulary which contribute to accuracy and efficiency. This selection is fast instance of feature selection.
Algorithm and it"s explanation:
Applied sequentially:
ICF, LVQ, DROP, POP
DEMONSTRATION:
Input: training set T with n stop words and m bug report,
Reduction order IS -> FS
Final number n(f) of words,
Final number m(I) of bug report,
Output: reduced, triage and send data set T (FI) to matching department.
Steps are as followed:

1. Admin (project manager) apply IS to n stop words T and calculate objective values for all words;
2. Select the top n(I) words of T and generate a training set T(I);
3. Filter bugs F(I) and send to suitable department D (I);
4. Terminate IS when new department to come;
5. end

XI. CLASSIFIER MODELING

An automated process to find some metadata about a document is considered as "Text classification." It has been used in various areas like document indexing by suggesting its categories in a "content management system", "spam filtering", "automatic help desk requests" sorting etc.
For bug classification, Naïve Bayes text classifier is used in this research. Naïve Bayes classifier is based on Bayes" theorem with maverick assumption and is a probabilistic classifier. It implies that the classifier speculates that any feature of a class is unassociated to the presence or absence of any other feature.

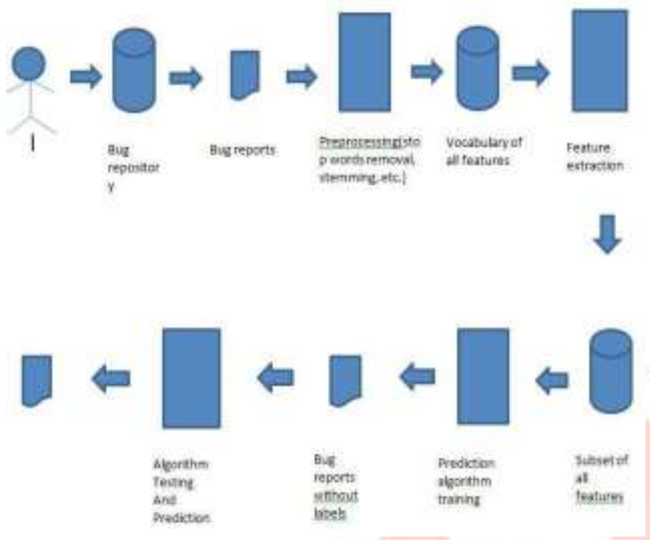
Bug repositor y
Bug reports
Preprocessing(sto p words removal, stemming, etc.)
Vocabulary of all features
Feature extraction
Subset of all features
Prediction algorithm training
Bug reports without labels
Algorithm Testing And Prediction

Fig: Block diagram illustrating classification of bugs

XII. ARCHITECTURE DIAGRAM

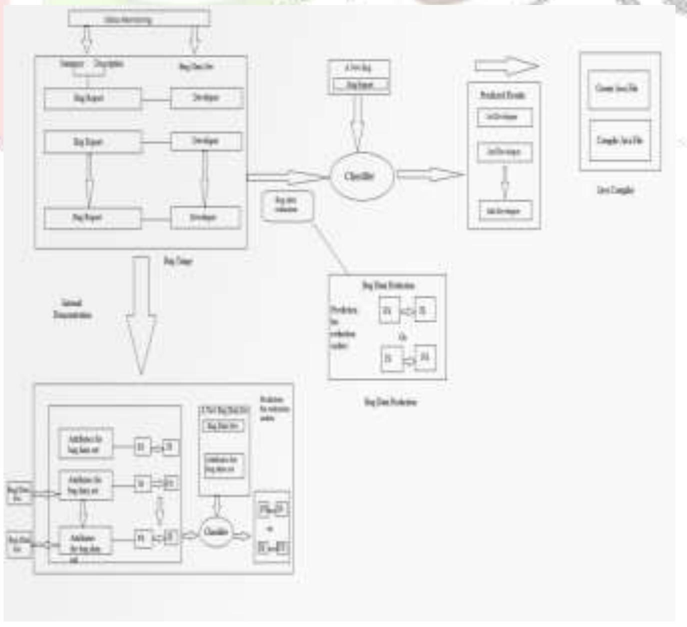

Fig: Architectural diagram

XIII. Modules and Description

This project shall require the implementation of following modules:
Authentication users, Products, Bug details, View, Admin and Logs.

Authentication users:

The Bug Tracking System first activates the login form. Here the user is prompted to enter the "user name" and „password" and our system starts the authentication process in which the username and password are verified with the help of existing username and password in the database. If the password matches then it is allowed to the main page else it warns the user for Invalid User name and password. After successful authentication, the system activates menus. The activity log has also been prepared for failures.

Bug Details:

The user is provided with the facility for adding bugs, updating the existing bugs in this module. As the number of bugs for a product can be very large, this system is equipped with efficient filtering. The user can filter the bugs based on priorities, databases, operating systems and status. After the user applies filter the list of bugs are displayed from the database.
More modules might be added as and when needed in to the stage of project development.

XIV. CONCLUSION

In open source bug repositories, bugs are reported by users numerous times. Triaging of these bugs is a repetitive and protracted task. If some proper class is assigned to these bugs ,then they could be easily allotted to a relevant developer and thus bugs can be fixed efficiently. Still, as reporters of these bugs are mostly non-technical it would be unfeasible for them to assign correct class to these bugs. In this research multinomial Naïve Bayes text classifier is used for classifying software bugs. Instance selection algorithm and Feature selection algorithm are used for bug triage. Maximum accuracy at prediction can be obtained using this system. Bug triage is an expensive step of software maintenance in both labor and time cost. In this project, advantages like reduction in bug data sets and improved data quality can be obtained because of combination of instance and feature selection.

XV. FUTURE WORK

The main challenge would be performing classification for numerous numbers of domains that could be tedious but important process. This will help teams of developers under one roof to work on their separate domains with fairness and utilization of time will be done properly.

References

[1] A. Hotho, A. Nürnberger and G. Paaß, "A Brief Survey of Text Mining," vol. 20, GLDV Journal for Computational Linguistics and Language Technology, 2005, pp. 19-62.

[2] A. E. Hassan, "The Road Ahead for Mining Software Repositories," IEEE Computer society, pp. 48-57, 2008.

[3] S. Diehl, H. C. Gall and A. E. Hassan, "Special issue on mining software repositories," in Empirical Software Engineering An InternationalJournal © Springer Science+Business Media, 2009.

[4] O. B. Michael and G. C. Robin, "A Bug You Like: A Framework for Automated Assignment of Bugs.," IEEE 17th international conference, 2009.

[5] C. Zhang, H. Joshi, S. Ramaswamy and C. Bayrak, "A Dynamic Approach to Software Bug Estimation," in SpringerLink, 2008.

[6] L. Yu, C. Kong, L. Xu, J. Zhao and H. Zhang, "Mining Bug Classifier and Debug Strategy Association Rules f or Web-Based Applications," in 08 Proceedings of the 4th international conference on Advanced Data Mining and Applications , 2008.

[7] N. Jalbert and W. Weimer, "Automated Duplicate Detection for Bug Tracking Systems," in IEEE computer society, 2008

[8] T. Bruckhaus, C. X. Ling, N. H. Madhavji and S. Sheng, "Software Escalation Prediction with Data Mining," in Data Mining, Fifth IEEEInternational Conference, 2006.

[9] [Online]. Available: https://bugzilla.mozilla.org/.

[10] [Online].Available:https://bugs.eclipse.org/bugs/.

[11] [Online].Available:https://en.wikipedia.org/wiki/Naive_Bayes_classifier.