System for Effective Data Processing Using Flaw Traige

Nikita Hiwase, Prof. RoshniTalmale
TGPCET, Mohgaon, Nagpur.

Abstract :- A Software bug is an error, flaw, failure or fault in a computer program or system that causes it to produce an incorrect or unexpected result. When bugs arise, we have to fix them which is not easy. Most of the companies spend 40% of cost to fixing bugs. The process of fixing bug is bug triage or bug assortment. Triaging this incoming report manually is error prone and time consuming .software company pays most of their cost in dealing with these bugs. In this paper we classifying the bugs so that we can determine the class of the bug at which class that bug is belongs and after applying the classification we can assign the particular bug to the exact developer for fixing them. This is efficient. In this paper we are using combination of two classification techniques , naïve bayes (NB) and k nearest neighbor(KNN).In modern days company uses automatic bug triaging system but in Traditional manual Triaging system is used which is not efficient and taking too much time .For triaging the bug we require bug detail which is called bug repository. In this paper we also reducing the bug dataset because if we having more data with unused information which causes problem to assigning bugs. For implementing this we use instance selection and feature selection for reducing bug data. This paper describe the whole procedure of bug allotment from starting to end and at last result will show on the basis of graph .Graph represents the maximum possibility of class means at which class the bug will belongs.

Keywords*-bug triage, bug data reduction, bugs classification technique (NB &KNN)*

*****

Introduction

MINING programming storehouses is an interdisciplinary space, which plans to utilize information mining to manage programming designing issues [33]. In cutting edge programming advancement, programming storehouses are vast scale databases for putting away the yield of programming improvement, e.g., source code, bugs, messages, and determinations. Conventional programming examination is not totally reasonable for the huge scale and complex information in programming archives [31]. Information mining has developed as a promising intends to handle programming information (e.g., [7], [32]). By utilizing information mining systems, mining programming archives can reveal intriguing data in programming vaults and tackle certifiable programming issues. A bug vault (a run of the mill programming storehouse, for putting away points of interest of bugs), assumes an imperative part in overseeing programming bugs. Programming bugs are unavoidable and settling bugs is costly in programming improvement. Programming organizations spend more than 45 percent of expense in settling bugs [39].

Expansive programming ventures send bug storehouses (additionally called bug following frameworks) to bolster data accumulation and to help engineers to handle bugs [9], [14]. In a bug vault, a bug is kept up as a bug report, which records the printed depiction of recreating the bug and redesigns as indicated by the status of bug altering [64]. A bug archive gives an information stage to bolster numerous sorts of assignments on bugs, e.g., shortcoming expectation [7], bug limitation [2], and re-opened bug examination [63]. In this paper, bug reports in a bug storehouse are called bug information. There are two difficulties identified with bug information that may influence the powerful utilization of bug archives in programming improvement errands, in particular the substantial scale and the low quality.

In this paper we using the classification to assigning the bug which is automatic process and by using this bug fixing is done quickly as compare to the manual assignment. Depending on graph we can assign the bug easily, here two graphs are use each one is represent maximum possibility of class and these two graphs are describe the output of two classifier techniques.

One of the most important reasons why bug triaging is such a lengthy process is the difficulty in selection of the most competent developer for the bug kind. The bug triager, the person who assigns the bug to a developer, must be aware of the activities (or interest areas) of all the developers in the project. Bug triaging normally takes 8 weeks to resolve a bug if the developer, to whom the bug report is assigned, could not resolve it, it is assigned to another developer. This would consume both time and money. Thus, it is really important on part of bug triager to assign the bug report to a developer who could successfully fix the bug without need of any tossing. Hence, the job of bug triager is really crucial [2].

A Framework for Automated Assignment of Bugs

Olga Baysal et al. [18] present a system for robotized task of bug. They proposed system which can finish up an engineer's level of ability by following the historical backdrop of the bugs beforehand determined by this designer. Inclination elicitation implies the issue of building up a choice emotionally supportive network which is fit for creating suggestions to a client, which then help with basic leadership. Olga Baysal et al. [18] Approach utilizes inclination elicitation [19] to learn designer preferences in altering bugs inside a given framework. Olga Baysal et al. [18] apply a vector space model to prescribe specialists for fix that bugs. At the point when another bug report arrives, the framework consequently relegates it to the right designer by considering his or her mastery, current workload, and inclinations.

RELATED WORK

Programming advancement includes numerous difficulties. One such test is the procedure of bug following. Bugs during the time spent programming advancement are unavoidable. Nonetheless, they are to be followed and altered with some concentrate else they are showed into the last item and that prompts disappointment of programming frameworks. An blunder, deficiency, disappointment, botch, blemish can be called as "bug" concerning a product framework. SDLC has numerous stages.

The improvement group may confer botches in any stage. Be that as it may, when comes to coding it gets surfaced once execution is tried. The majority of the bugs can be stayed away from by having right examination and outline according to client necessities. As indicated by [6] bugs in programming framework cause the framework to come up short. It means that such programming framework can't meet the quality and desires of the customer. At the point when programming framework can meet the practical prerequisites given by the client, which is said to be a quality item.

At the point when quality issue emerges, client gets disappointed. As indicated by [7] a bug in programming framework is not a mischance bug that happens because of particular reason. Following bugs has numerous stages and it has its own particular life cycle as appeared in fig. 1.

Figure: Bug Tracking Life Cycle

As can be found in fig. 1, a bug following is cyclic in nature. Right from its underlying finding, until it is determined or altered for all time it has its lifecycle with certain stages. As bugs cause frameworks to act mistakenly bug following frameworks lessen the probability of having bugs in the framework long time. Rapidly the product improvement group can settle bugs and close them with appropriate bug following framework. Bug following enhances quality of programming being produced. Indeed, even in the wake of conveying programming items bugs might be uncovered and still the attaching framework is to be utilized to settle those bugs speedier. At the point when programming framework is being produced bug following is critical furthermore difficult undertaking [14]. Generally bug following is made straightforward. It means that numerous a long time down the line, programming engineers utilized spreadsheets to store bug points of interest and alter them. In any case, utilizing programming like spreadsheet is not powerful considering cutting edge offices, for example, messaging; RSS channels, and change warning through SMS et cetera. In any case, it is extremely valuable to utilize a spreadsheet application like Excel for little scale ventures with less many-sided quality and less heading. As spreadsheets need in security and not complex with regards to database operations, now a day's kin are utilizing social databases to store bug reports and present them through a bug following framework front end. Bug following framework helps QA groups in programming improvement cycles.

Programming improvement organizations surely utilize a bug following framework as it is vital. Bug following framework is truly required while building up each product item the same number of engineers don't keep up satisfactory documentation concerning client necessities through the whole life cycle of the item [8].

There are numerous advantages of utilizing a bug following framework. As determined in [6], clear correspondence can be given by a bug database. At the point when contrasted with casual messages and so on elegantly composed reports with principles clarify the bugs and the needs exceptionally well. In any case, it is difficult to track bugs. It is a dull assignment as determined in [6]. Numerous bug following frameworks are as of now accessible. They are either open source or business. Best bug following framework among them can be utilized to satisfy client necessities.

At the point when bug following frameworks are utilized numerous variables are to be considered. While utilizing bug following framework the variables to be watched are application setup, reporting procedure, and notice technique. Before utilizing a bug following framework, it must be set up. The understood bug management application which is open source is "Bugzilla". In numerous open source items, for example, Linux, Overshadowing, and Mozilla it is being utilized. There may be setup issues with open source items. As per [10] free bug following frameworks like Bugzilla take more opportunity to setup and don't show up client agreeable. This is on the grounds that its setup procedure is extremely confounded. Clients feel it troublesome while introducing its server.

4.1 PROPOSED SYSTEM

• We introduce the issue of information lessening for bug triage. This issue means to increase the information set of bug triage in two viewpoints, to be specific

a) To at the same time lessen the sizes of the bug measurement and the word measurement.

b) to enhance the precision of bug triage.

• We propose a mix way to deal with tending to the issue of information diminishment. This can be seen as a use of example choice and highlight determination in bug vaults.

• We manufacture a parallel classifier to foresee the request of applying case choice and highlight choice. As far as anyone is concerned, the request of applying occurrence determination and highlight choice has not been explored in related spaces.

By utilizing the programmed bug triage approach the time and cost is spared. This framework is executed by utilizing the Naive bayes grouping strategy. The task of bug to designer is done naturally. Nature of bug triage is enhanced utilizing this new framework, in light of the fact that here we utilize classifier for execution. We are utilizing occasion and highlight determination for information decrease. Occurrence determination and Feature choice with Naïve Bayes and KNN grouping for bug triage.

4.2 System Architecture

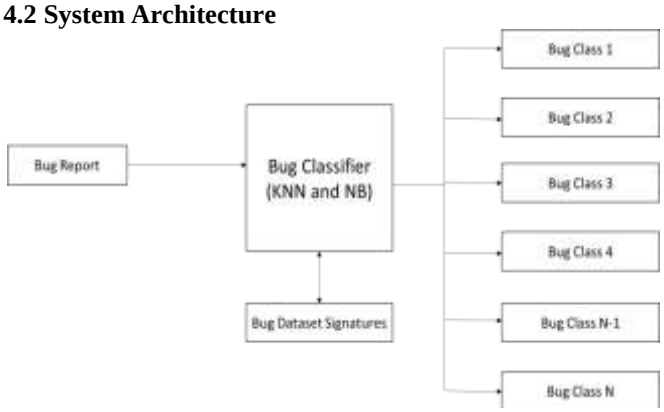

Figure: Architecture

4.3 Algorithm

4.3.1 Algorithm Preprocess (Data D)

Step 1: Read Data into Array
Step 2: Remove All Stop words

$$\Sigma\, i = 0 \mid \phi\, n \neq stop(i)$$

Step 3: Remove Redundancy from Array

$$\Sigma\, i = 0 \mid \phi\, n \neq repeat(i)$$

Step 4: Remove all Special Symbol and digits.
Step 5: Write back

4.3.2 Algorithm Classification (Data D)

Step 1: Read Data into Array
Step 2: Call Preprocess (D)
Step 3: Calculate Word count

$$\Sigma\, i = 0 \mid \phi\, i = i + 1\; if\; a[i] \in final[i]$$

Step 4: Calculate Frequency
Tf = number of occurrences / total words
Step 5: Calculate Normalized TF
NTF = sum of Tf / number of classes
Step 6: Generate Decision Matrix
Step 7: Calculate Final max class value and classify.

4.4 Module

Module1: Dataset Generation- First module of this project is dataset generation. all types of classes are inserted in this dataset. we introduce 6 types of classes. Bad practice, Correctness, dodgy code, Experimental, Internationalization, and Malicious code Vulnerability.
If bug belongs to this classes it automatically assign to that class of developer which is efficient for fixing the bug.
Module2: Prepossessing second module of this project is prepossessing. In this module we remove all stop words which is not useful for detecting the class.
After prepossessing we will apply the classifier on that preprocessed data for detecting the class.

Module3: Bug Classification: This is third module "bug Classification". Two classification algorithms are use for classifying the bug. NB(naïve bayes) and KNN(K nearest neighbor).In this project we are using the Combination of these two algorithm due tothis, it generate the better result.
Module 4: Bug allotment: last module is Bug allotment where we are allotting the bug to the particular developer depending upon their class.

RESULT

Basic Login Page:

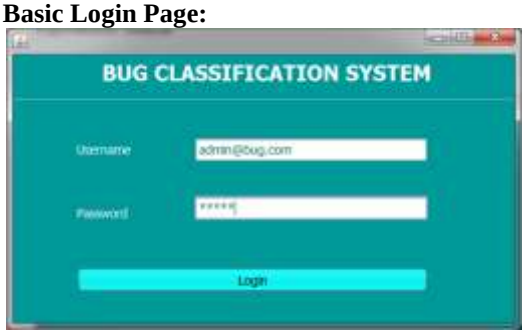

Fig: Role Based Login Page

This is login page of Admin, after entering the Id and password admin can go to the Main page.

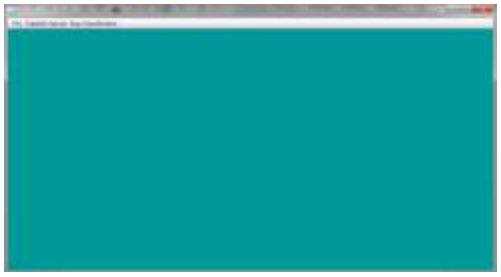

Fig: Main Page

Four subparts of main page are File, Dataset Upload and Bug Classification. In file there we can add the new user after Registration. In Dataset upload , we can add the new class through Add new class and we can also update the class.4th part is bug classification where we can classify the bug.

Fig User registration

In this module we register the user with her details, that useful in system database to detect bug and unauthorized user.

I **Fig Add class**

n this module we entry the different bug classes that used by system to detect bug and unauthorized user.

Fig update class

Fig Tester home

This is Tester Home page, only Tester is responsible for submit the bug with bug details.

Fig. Graph based on KNN

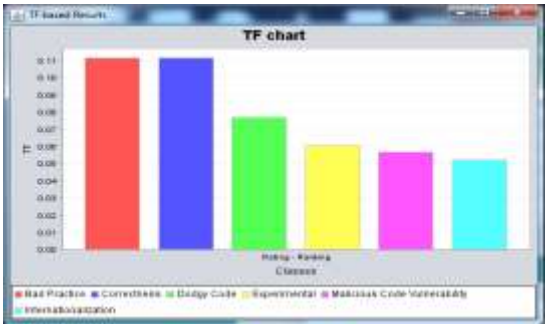

Fig.Graph based on KNN &NB

Here two Graphs are Present 1st one is based on word count which support KNN Classifier and 2nd is based on Term Frequency which support NB and KNN classifier. This Second graph shows the output of classifier.

Conclusion and Future Scope

Conclusion

Bug triage is a costly stride of programming upkeep in both work cost and time cost. In this paper, we join highlight choice with occasion determination to diminish the size of bug information sets and enhance the information quality. To decide the request of applying example determination and highlight choice for another bug information set, we separate traits of every bug information set and prepare a prescient model in view of verifiable information sets. We experimentally examine the information lessening for bug triage in bug storehouses of FindBugs Database of Bugs. Our work gives a way to deal with utilizing strategies on information handling to shape diminished and brilliant bug information in programming advancement and upkeep. We also provide a system that can classify bugs with the help of KNN and NB classifier system. Bug Triage with automated classification is the main objective of proposed system.

In this paper we introduced the concept of bug classification for allotting thebug to appropriate developer for fixing it and for this purpose we used thecombination of two algorithms NB and KNN. Wealso introduced the reduction process of bug database by applying prepossessing.

Future Scope

Bug tracking systems are an important part of how teams in open source interact with their user communities. This interaction goes beyond users simply submitting bugs. Many follow-up questions are posed to the reporters of bugs and Often, if a reporter does not play an active role in the discussion of the bug, little progress is made. Our results highlight the importance of effectively and efficiently engaging the user community in bug fixing activities, and keeping them up-to-date about the status of a bug. We believe that our results will help to form the design of new bug tracking systems that will aim at eliciting the right information from Users and facilitating communication between end users and developers as well as among developers. An integration and active participation of users in bug tracking will result in bugs being fixed faster and more efficiently.

REFERENCES

[1] JifengXuan, He Jiang, Yan Hu, ZhileiRen, WeiqinZou, ZhongxuanLuo, and Xindong Wu, ―Towards Effective Bug Triage with Software Data Reduction Techniques,‖ IEEE Transactions, Volume 27, NO. 1, JANUARY 2015.

[2] Anjali, Sandeep Kumar Singh, ―Bug Triaging: Profile Oriented Developer Recommendation‖, International Journal of Innovative Research in Advanced Engineering (IJIRAE) ISSN: 2349-2163, Volume 2, Issue 1, January 2015.

[3] S. Artzi, A. Kie_zun, J. Dolby, F. Tip, D. Dig, A. Paradkar, and M. D.Ernst, "Finding bugs in web applications using dynamic test generation and explicit-state model checking," IEEE Softw., vol. 36, no. 4, pp. 474–494, Jul./Aug. 2010.

[4] PankajRana, Asst. Prof. Saurabh Sharma "Review of Bug Serverity Prediction Techniques Using Data Mining"Volume 5, Issue 6,June2015 .

[5] 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.

[6] J. Anvik and G. C. Murphy, "Reducing the effort of bug report tri- age: Recommenders for development-oriented decisions," ACM Trans. Soft. Eng. Methodol., vol. 20, no. 3, article 10, Aug. 2011.

[7] 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.

[8] PankajGakare, YogitaDhole,SaraAnjum, ―Bug Triage with Bug Data Reduction‖, International Research Journal ofEngineering and Technology (IRJET) e-ISSN: 2395 -0056,Volume 02 Issue 04, July 2015.

[9] C. C. Aggarwal and P. Zhao, "Towards graphical models for text processing," Knowl. Inform. Syst., vol. 36, no. 1, pp. 1–21, 2013.

[10] D. Cubranic and G. C. Murphy, "Automatic bug triage using text categorization," in Proc. 16th Int. Conf. Softw. Eng. Knowl. Eng., Jun. 2004, pp. 92–97.

[11] J. Anvik and G. C. Murphy, ―Reducing the effort of bug report triage: Recommenders for development-oriented decisions,‖ ACM Trans. Softw. Eng. Methodol., vol. 20, no. 3, pp. 10:1–10:35, Aug. 2011.

[12] A. K. Farahat, A. Ghodsi, M. S. Kamel, "Efficient greedy feature selection for unsupervised learning," Knowl. Inform. Syst., vol. 35, no. 2, pp. 285–310, May 2013.

[13] K. Gao, T. M. Khoshgoftaar, and A. Napolitano, "Impact of data sampling on stability of feature selection for software measure- ment data," in Proc. 23rd IEEE Int. Conf. Tools Artif. Intell., Nov. 2011, pp. 1004–1011.

[14] J. A. Olvera-Lopez, J. A.Carrasco-Ochoa, J. F. Martınez-Trinidad, and J. Kittler, "A review of instance selection methods," Artif. Intell. Rev., vol. 34, no. 2, pp. 133–143, 2010.

[15] C. Sun, D. Lo, S. C. 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.