IJETIR

ISSN (Online) 2583-0554

International Journal of Emerging Technologies and Innovative Research (IJETIR)

Volume 4, Issue 6, June 2024

Impact Factor: 5.731

Predicting Software Defect Complexity and Accuracy using Bug Tracking and Clustering

Swetha. S[1] and A.Poongodi[2]

MCA Student, Department of Computer Applications[1]

Professor, Department of Computer Applications[2]

Vels Institute of Science, Technology & Advanced Studies (VISTAS), Chennai, Tamil Nadu, India

22304241@vistas.ac.in and poongodimca1979@gmail.com

Abstract: *Many open sources, free and commercial bug tracking tools have been developed and are currently under development. There are number of issues are related to software projects are daily increasing and the developers are started to use bug tracking systems in that order to manages the bug reports. The industry needs that the criteria to select the best system tool among the available set of system tools which will helps to fix and track the progressive report of bug fixes. While, collection of useful information from the large and not organized set of these reports is still difficult problem because there are various bug tracking systems are provide the data via many resources like web interfaces. We try to present these comprehensive classification criteria to manage the reviews for available tools and propose a new modified tool for the bug tracking and reporting system. It also helps in reporting the bugs which arefounded by that process, assigning the bug to the developer for monitoring and fixing the progress of bug fixing by various graphical/charting facility and status updates. It also providing the reliability of bug prediction and tries to find the bugs for complexity measurements, and allows to distributing.*

Keywords: SVM, Association Rule, CNN

I. INTRODUCTION

The main purpose of this project Bug Tracking System project is to deal with providing online support to the software engineers who are facing the bugs or errors in software technologies. This project can maintain project details, developer details and tester details. Bug Tracking System is the system which enables to detect the bugs. It does not find the bugsbut provides the full information regarding bugs detected. Bug Tracking System allows the user of it who wants to know about a provide information to the identified bugs. Theengineers develop the project as per client requirements. The tester will identify the bugs in the testing phase.

Whenever the tester facing number of bugs then he adds the bug id and information in the database. The tester informs to project manager and developer. The bug details in the database table are accessible to project manager and developer. When a client puts request or orders for a product to be developed. The system/Project manager is responsible to adding users in the Bug Tracking System and assigning projects to the users. This project provides bug information includes the bug id, bug name, bug priority, project name, bug location, bug type.

This all processes are continuously executing until all the bugs are resolved in this system. This system also able to help for bug report is sending to the System/project manager and the developer as far as the bug is getting identified. It makes easy to anyone who really needs to know about the bug can learn of it soon after it is reported. Bug Tracking System plays an main role in the testing phase. But it supports assigning projects for the developer, tester. ThisBug Tracking System maintaining the different users and it provides separate environments for project manager, developer tester.

IJETIR
ISSN (Online) 2583-0554
International Journal of Emerging Technologies and Innovative Research (IJETIR)
Volume 4, Issue 6, June 2024
Impact Factor: 5.731

II. LITERATURE SURVEY

An Eye Tracking Research on Debugging Strategies towards Different Types of Bugs Fei Peng; Chunyu Li; Xiaohan Song; Wei Hu; Guihuan Feng,2023

Debugging is one of the important links in software quality assurance. Generally, the debugging performance of people adopting different debugging strategies varies enormously. Although there are studies discussing debugging strategies, little research analyzes the impactof these strategies towards different types of bugs. In this paper, the experiments conductedon 20 participants suggest that there do exist differences on the eye movement data of those successful and failed debugging samples. Specifically, concerning data flow bugs, it is beneficial to pay attention to the changes of variables, nevertheless, it is more important to watch the code and understand their logical structure when dealing with control flow bugs. We believe it can help programmers find defects more efficiently by combining this conclusion and the error message provided by the compiler.

The Eclipse and Mozilla defect tracking dataset: A genuine dataset for mining bug information Ahmed Lamkanfi; Javier Pérez; Serge Demeyer,2023

The analysis of bug reports is an important subfield within the mining software repositories community. It explores the rich data available in defect tracking systems to uncover interesting and actionable information about the bug triaging process. While bug data is readily accessible from systems like Bugzilla and JIRA, a common database schema and a curated dataset could significantly enhance future research because it allows for easier replication. Consequently, in this paper we propose the Eclipse and Mozilla Defect Tracking

Dataset, a representative database of bug data, filtered to contain only genuine defects (i.e.,no feature requests) and designed to cover the whole bug-triage life cycle (i.e., store all intermediate actions). We have used this dataset ourselves for predicting bug severity, for studying bug-fixing time and for identifying erroneously assigned components. Sharing these data with the rest of the community will allow for reproducibility, validation and comparison of the results obtained in bug-report analyses and experiments.

III. METHODOLOGY

Data collection

Data collection is a main integral part of implementing mining techniques, for this challenge and data recording systems was used. Data recording system provides route data to excel in tabular form. "Data record based on a digital processor which is used by the built-in sensor oran external instrument and sensors associated with position data of the time of the electronic device can automatically collect and records data of 24 hours. This was the main and most important benefits of data recorder. It was used to collect route data from local stations at to adevoted lab PC, then copy the transferred weather data to an Excel spreadsheet and recorded on daily basis along with monthly basis to identify data.

Data pre-processing

The data preprocessing is the next step of Machine Learning after collection of data. Thus, the data is pre-processed to remove noise and unwanted data. Pretreatment means concentrating the removal of other unwanted variables from the data, while the data preprocessing includes these steps:

Data scrubbing: it's the stage where noise and irrelevant data is removed. Data cleaning procedures are implemented to fill out missing values and to eliminate noise in recognizing outliers and to correct data irregularities.

Data integration: it's recognized as the data conversion; in this stage, the suitable form of data is converted for the procedure of Machine Learning by reduction of data and construction of attributes.

Analysis of Result

The values obtained from different routes were analyzed depending on the mining results.

- User authentication

IJETIR
ISSN (Online) 2583-0554
International Journal of Emerging Technologies and Innovative Research (IJETIR)
Volume 4, Issue 6, June 2024
Impact Factor: 5.731

- Data preprocessing
- Data analysis
- User details
- Predict bug
- Result set

BugHerd:

BugHerd is a web-based application for collecting and managing website feedback and tracking bugs. The software uses a sidebar to collect feedback from a website, which then pushes tasks to a Kanban style task management system. Originally launched to users in2011, it is owned by Splitrock Studio and is based in Melbourne, Australia.

Users can provide feedback on a website using a browser extension or Javascript snippet. Once installed, any user can point, click and offer a comment on an issue, leaving the comment as a task directly on the page. The task created will contain metadata such as a screenshot, the ability to assign a priority, a person to execute the task and relevant browser data and operating system information.

The task (or bug) is then managed from a Kanban board style task management system, with editable columns such as backlog, to do, doing and done. BugHerd is used by people who build websites such as web developers and designers, as well as people responsible for website content, such as marketers and project managers.

The company took a hiatus in 2015, going into maintenance mode after failing to achieve expected growth numbers and not achieving unicorn startup status. In 2019 a new CEO, Stephen Neville, was appointed to reinvigorate the company due to a persistent customerbase.

Bugherd returned to active development in 2019 and actively re-hired a number of original team members to breathe life back into the dormant product. The company is one of the tech products owned and developed by startup studio, Splitrock Studio.

Mantis Bug Tracker:

Mantis Bug Tracker is a free and open source, web-based bug tracking system. The most common use of MantisBT is to track software defects. However, MantisBT is often configured by users to serve as a more generic issue tracking system and project management tool.

The name **Mantis** and the logo of the project refer to the insect family Mantidae, known for the tracking of and feeding on other insects, colloquially referred to as "bugs". The name of the project is typically abbreviated to either **MantisBT** or just **Mantis**.

An event-driven plug-in system was introduced with the release of version 1.2.0. This plug-in system allows extension of MantisBT through both officially maintained and third party plug-ins. As of November 2013, there are over 50 plug-ins available on the MantisBT-plugins organization on GitHub.

Prior to version 1.2.0, a third party plug-in system created by Vincent Debout was availableto users along with a variety of different plug-ins. This system was not officially supported bythe MantisBT project and is incompatible with MantisBT 1.2.0 and later.

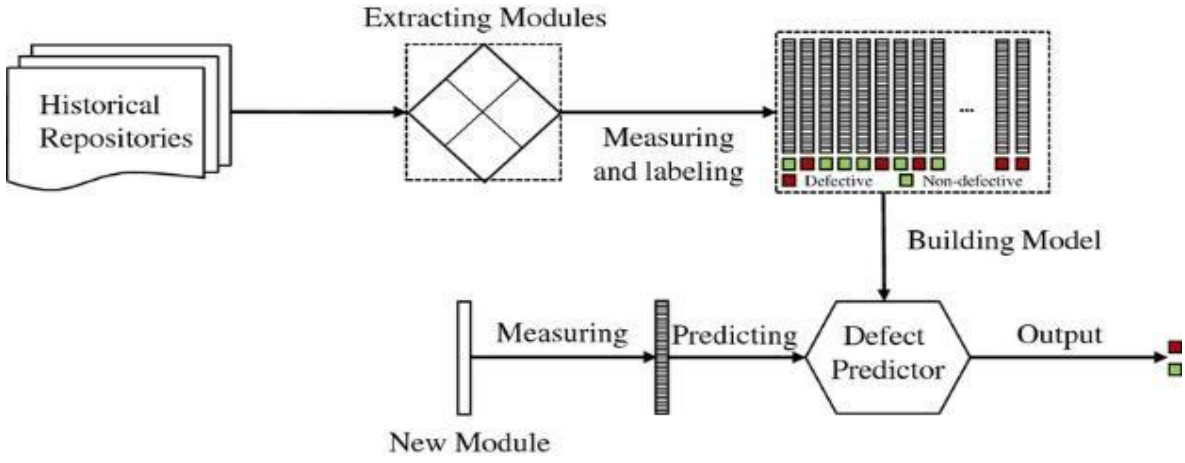
MantisBT is mainly written in PHP and uses SQL to interface with databases. The web-based user interface of MantisBT is written using HTML which is styled and presented using CSS. The UI also uses the jQuery client-side JavaScript library to provide optional features such as Ajax and JSON powered dynamic page content.

Development tools and build scripts are written predominantly in Python with some Shell script and PHP.

MantisBT's codebase dates back to a time when PHP's support for object-oriented programming principles was in its infancy. As of version 1.2.0, the majority of the MantisBT codebase still uses procedural programming principles, however some sections have been converted to make use of PHP 5's new object model.

IJETIR
ISSN (Online) 2583-0554
International Journal of Emerging Technologies and Innovative Research (IJETIR)
Volume 4, Issue 6, June 2024
Impact Factor: 5.731

FLOWCHART



IV. EXPERIMENTAL RESULTS

5/4/24, 11:35 PM proposed1 - Jupyter Notebook

```
In [10]: trace = go.Histogram(
             x = data.defects,
             opacity = 0.75,
             name = "Bugs Classification",
             marker = dict(color = 'green'))

         hist_data = [trace]
         hist_layout = go.Layout(barmode='overlay',
                                 title = 'Defects',
                                 xaxis = dict(title = 'True - False'),
                                 yaxis = dict(title = 'Frequency'),
         )
         fig = go.Figure(data = hist_data, layout = hist_layout)
         iplot(fig)
```

Fig: 1 Training data set for X and Y Axis

V. CONCLUSION AND FUTURE SCOPE

In this project we have reviewed on the technologies which are being used for finding and improving bug tracking system. Further we have introduced different techniques used to implement them. Present methods include database server and admin information. Later onwe use SVM and CNN are used for comparative study in terms of accuracy and storing of structure data into the database etc. This comparison will help us in building our system more

convenient and useful. From the research we have proposed the system which will predict time required for particular task.

The proposed work can be extended using advanced text pre-processing techniques for optimizing the clustering and classification work, and also modern text clustering and Classification algorithms can be implemented and compared with the proposed algorithm. We also plan to extend it by performing feature selection on our factors. Employing feature selection may improve the performance of our models since it removes redundant factors

Our results show that blocking bugs take longer to be fixed than non-blocking bugs; however it is unclear if blocking bugs require more effort and resources than non-blocking bugs. To tackle this question, we plan to link bug reports with information from the version control systems, leverage metrics at commit level and perform a quantitative analysis that may help us to confirm or refute our intuition that blocking bugs indeed require more effort.

REFERENCES

[1] Current challenges in automatic software repair. Goues, Claire Le, Forrest, Stephanie and Weimer, Westley. New York: Springer Science+Business Media, 2013.

[2] Yuan Tian, David Lo, Chengnian Sun, "Information Retrieval Based Nearest Neighbour Classification for Fine Grained Bug Severity Prediction", WCRE, 2012, 2013 20th Working Conference on Reverse Engineering (WCRE), 2013 20th Working Conference on Reverse Engineering (WCRE) 2012, pp. 215-224, doi:10.1109/WCRE.2012.31

[3] http://www.bugzilla.org/

[4] Shaffiei, Zatul Amilah, Mudiana Mokhsin, and Saidatul Rahah Hamidi. "Change and Bug Tracking System: Anjung Penchala Sdn. Bhd." Change 10.3 (2010).

[5] http://www.techopedia.com/definition/13698/tokenization

[6] Valdivia Garcia, Harold, and Emad Shihab. "Characterizing and predicting blocking bugs in open source projects." Proceedings of the 11th Working Conference on Mining Software Repositories. ACM, 2014.

[7] N. Kumar Nagwani and S. Verma, "CLUBAS: An Algorithm and Java Based Tool for Software Bug Classification Using Bug Attributes Similarities," Journal of Software Engineering and Applications, Vol. 5 No. 6, 2012, pp. 436-4. doi: 10.4236/jsea.2012.56050.

[8] http://istqbexamcertification.com/what-are-the-software-development-life-cycle-sdlc- phases/