Reproduction of "Predicting the Severity of a Reported Bug"

Andrina Vincenz
andrina.vincenz@uzh.ch
University of Zurich
Zurich, Switzerland

David Stalder
david.stalder@uzh.ch
University of Zurich
Zurich, Switzerland

ABSTRACT

Background. Bug fixing is a fundamental part of software maintenance. Bugs have different levels of severity according to their threat to the system and urgency to fixing.

Aim. The aim is to be able to have models which are able to identify the severity of bug reports.

Method. After identifying the system, bug reports will be extracted from bugzilla including the severity of the bug. This data is used to train two classifiers: SGDClassifier and Multinomial Naïve Bayes. The following measurements are used to identify the quality of the model: precision, recall and f-measures.

Conclusion. This study will show if it is possible to generate models, which can do the classification of bug reports automatically with certainty. This can help software engineers in practice to classify their bug reports and focus on the most urgent.

1 INTRODUCTION

Bug reports represent a fundamental part of software maintenance by documenting the malfunction of a software part. A large number of bug reports by users and developers are collected on a daily basis in bug tracking systems like Bugzilla. These reports are then manually prioritized according to their severity and classified into 6 groups, ranging from a request to enhancement to a critical level resulting in crashes, loss of data or severe memory leak. The level of severity indicates the impact of the bug on the successful execution of a software system.

However, the manual assignment of the severity level to a bug report takes time and resources. In combination with the increase of daily bug reports the classification introduces a set of challenges such as misclassification of severity or the increase of cost. The ability of the automated severity prediction of bug reports by a tool reduces errors in misclassification and manual work as well as enhance the software quality.

In order to automate the severity predication of bug reports, different classification and information retrieval techniques based on textual descriptions have been used previously. Techniques range from Nearest Neighbours [5] [1] [4], Naïve Bayes [1] [4] to Support Vector Machines [3] [1] [4]. While some work focuses on automatically analyzing the text of past bug reports and the recommendation of severity labels [5] [1] [4] other work creates a classification-based approach to create a bug priority recommender [3].

For this project, we intend to conduct a reproduction of the paper Predicting the severity of a reported bug" [2] by Lamkanfy et al. It was originally written in 2010, which we consider as too old for such a current topic.

The research goal of this paper was that they tested if it is possible to predict the severity of a reported bug by analysing its textual description by using a text mining algorithm. They came up with subsidiary research questions which need to be included to find an explanation of their research goal.

- Concerning potential indicators of the severity of a bug, the first research question was: "Which terms in the textual descriptions of a bug report could serve as good indicators of the severity?" [2]
- Concerning the length of a bug report, the second research question was: "Which (text) fields in the bug reports serve as the best prediction basis? The one-line summary which briefly focusses on the problem or the longer full description which includes more detail?" [2]
- Concerning the scope of the training period, their third research question was: "How many samples must be collected before one can make a reliable predictor?" [2]

The structure of the paper is as follows. In the second section "Background" we explain all the necessary background knowledge on bug triaging and text mining. In the third section "Case Study" we first explain the approach step by step, then we present the measures we used to process and evaluate the data and finally we present the results. In the fourth Section "Subsidiary Research Questions" we answer step by step the subsidiary research questions. In the fifth section "Threats to Validity", we need to identify factors that could threaten our results and the measures we took to minimize the risk of bad results. In the sixth section "Related Work" we discuss related work of other researchers and their findings to compare it with our findings. Finally, in the seventh section "Conclusion and Future Work" we summarize all our findings and mention future work to be done in conjunction with our research.

REFERENCES

[1] Q.D. Soetens T. Verdonck A. Lamkanfi, S. Demeyer. 2011. Comparing mining algorithms for predicting the severity of a reported bug. *2011 15th European Conference on Software Maintenance and Reengineering* (2011), 249–258. https://doi.org/10.1109/CSMR.2011.31

[2] S. Giger B. Goethals A. Lamkanfi, S. Demeyer. 2010. Predicting the severity of a reported bug. *7th Working Conference on Mining Software Repositories, Cape Town, South Africa, 2 May 2010 - 3 May 2010* (2010), 1–10. https://doi.org/10.1109/MSR.2010.5463284

[3] O. Maqbool J. Kanwal. 2010. Managing open bug repositories through bug report prioritization using SVMs. *Proceedings of the 4th international conference on open-source systems and technologies* (2010), 1–7. https://doi.org/ICOSST

[4] V.B. Singh K.K. Chaturvedi. 2012. Determining bug severity using machine learning techniques. *CSI sixth international conference on software engineering* (2012), 1–6. https://doi.org/10.1109/CONSEG.2012.6349519

[5] Sun C. Tian Y., Lo D. 2013. DRONE: predicting priority of reported bugs by multi-factor analysis. *2013 IEEE International Conference on Software Maintenance* (2013), 200–209. https://doi.org/10.1109/ICSM.2013.31