Knowledge-Based Systems 000 (2016) 1–25

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

# Linear and non-linear heterogeneous ensemble methods to predict the number of faults in software systems

Santosh Singh Rathore, Sandeep Kumar*

*Department of Computer Science and Engineering, Indian Institute of Technology Roorkee, Roorkee, India*

ARTICLE INFO

*Article history:*
Received 4 August 2016
Revised 25 October 2016
Accepted 18 December 2016
Available online xxx

*Keywords:*
Software fault prediction
Ensemble methods
Heterogeneous ensemble
Prediction of number of faults

ABSTRACT

Several classification techniques have been investigated and evaluated earlier for the software fault prediction. These techniques have produced different prediction accuracy for the different software systems and none of the technique has always performed consistently better across different domains. On the other hand, software fault prediction using ensemble methods can be very effective, as they take the advantage of each participating technique for the given dataset and try to come up with better prediction results compared to the individual techniques. Many works are available for classifying software modules being faulty or non-faulty using the ensemble methods. These works are only specifying that whether a given software module is faulty or not, but number of faults in that module are not predicted by them. The use of ensemble methods for the prediction of number of faults has not been explored so far. To fulfill this gap, this paper presents ensemble methods for the prediction of number of faults in the given software modules. The experimental study is designed and conducted for five open-source software projects with their fifteen releases, collected from the PROMISE data repository. The results are evaluated under two different scenarios, intra-release prediction and inter-releases prediction. The prediction accuracy of ensemble methods is evaluated using absolute error, relative error, prediction at *level l*, and measure of completeness performance measures. Results show that the presented ensemble methods yield improved prediction accuracy over the individual fault prediction techniques under consideration. Further, the results are consistent for all the used datasets. The evidences obtained from the prediction at *level l* and measure of completeness analysis have also confirmed the effectiveness of the proposed ensemble methods for predicting the number of faults.



## 1. Introduction

As the demand of high performance and efficient software systems is growing day by day, their complexity is also increasing continuously. The increased complexity makes the quality assurance task difficult. On the other hand, ensuring the high quality of software is a costly task and requires ample amount of resources. Additionally, many reports suggested that testing a software project thoroughly and completely is very difficult with the limited testing resources [1]. Therefore, many earlier works focused on the prioritization of quality assurance activities. Knowing the parts of software that are more likely to be fault-prone before the testing phase can help in allocating testing resources efficiently and optimally. Software fault prediction aims to solve this kind of problem. It involves prediction of the fault prone modules prior to the testing phase using some underlying characteristics of software. It uses software metrics (e.g. Lines of Code, McCabe Cyclomatic Complexity) and software faults obtained from the previous releases of the project or similar projects to build fault prediction model and uses this model to predict faults in the future releases of the project.

Several classification techniques have been successfully applied to software fault prediction over the last few decades. This includes techniques such as decision tree [2], naive Bayes [3], logistic regression [4], support vector machine [5], discriminant analysis [6], factor analysis [7], fuzzy classification [8], random forest [9], Bayesian network [10], artificial neural networks [11], etc., which are mainly applied for predicting software modules being faulty or non-faulty. Some other techniques such as, decision tree regression [12], negative binomial regression [13], Poisson regression [14], genetic programming [15] have been used for predicting the number of faults and fault density. The analysis of these techniques revealed that each technique has different prediction capabilities and the performance of these techniques is not consistent across dif-

* Corresponding author.
*E-mail addresses:* sunnydec@iitr.ac.in (S.S. Rathore), sgargfec@iitr.ac.in (S. Kumar).

http://dx.doi.org/10.1016/j.knosys.2016.12.017

ferent software systems and varies from dataset to dataset [16,17]. Recently, a number of researchers have analyzed ensemble methods for software fault prediction and reported better performance of ensemble methods over the individual participating fault prediction techniques [18–21].

The center idea of ensemble method is to utilize the prediction capabilities of several weak learning techniques for the given dataset and come up with better prediction results by combining their prediction outputs [22]. The idea of using the ensemble method is attractive, since the final output is based on combination of the outputs of different techniques that leads to the more accurate prediction. Ensemble method helps to alleviate the problem of finding the global minima for the given input function over the individual techniques. The problem with classification techniques is to find the global minima of the given function. Even, with the sufficient training dataset in-hand, it requires a significantly large number of computational resources to determine the global minima [18]. Ensemble method eliminates this problem by averaging or aggregating the local optimal solutions of several weak learning techniques for the given input function [18]. It also solves the small sample size problem by incorporating multiple techniques to reduce the potential risk of overfitting the training dataset. These benefits of ensemble methods attracted a significant amount of research using ensemble methods for the binary class classification of software faults. However, use of ensemble method for the prediction of number of faults remained unexplored so far. Locating and predicting the number of faults accurately can help the software tester to streamline the testing efforts to be applied in the later phases of software development. Therefore, evaluation of the effectiveness of ensemble methods is required for the prediction of number of faults in software systems.

Various works (e.g., [18,21,23]) on the use of homogeneous ensemble methods such as bagging, boosting for software fault prediction are available for predicting software module being faulty or non-faulty. However, in the area of ensemble methods for software fault prediction, very few works are available that presented ensemble methods for combining the outputs of different participating learning techniques. In homogeneous ensemble methods, we can only harness the capability of one learning technique. It might be better to use more than one learning techniques since different techniques can perform best locally and combining their outputs can help in achieving overall better accuracy [22,24]. Additionally, using more than one techniques increase the generalization of prediction. However, very few works including a rudimentary study by us that is under publication explore this for the software fault prediction, especially for the prediction of number of faults. Hence, the novelty of this work lies in presenting heterogeneous ensemble methods that combine different learning techniques for the prediction of number of faults. This paper investigates and evaluates the capabilities of ensemble methods for the prediction of number of faults in the given software system. Following are the contributions of the paper:

- This paper expands the use of ensemble methods for the prediction of number of faults unlikely the earlier works on ensemble methods that focused on predicting software modules as faulty or non-faulty.
- This paper investigates the usage of both heterogeneous ensemble methods as well as homogeneous ensemble methods for the prediction of number of faults.
- We present two linear combination rules and two non-linear combination rules for combining the outputs of the base learners in the ensemble.
- In addition, we assess the performance of ensemble methods under two different scenarios, intra-release prediction and inter-releases prediction.

In this paper, we perform an experimental study for five software projects with their fifteen releases, available in the PROMISE data repository [25] and evaluate the fault prediction performance. In this paper, we aim to find empirical evidences to answer the following research questions:

**RQ1:** *How do ensemble methods perform in the prediction of number of faults?*

This research question aims to evaluate the performance of ensemble methods for the prediction of number of faults. Earlier reported studies of ensemble methods for software fault prediction performed prediction in the form of predicting software modules being faulty or non-faulty. On the other hand, we build and evaluate ensemble methods for the prediction of number of faults.

**RQ2:** *How do heterogeneous ensemble methods perform in comparison to homogeneous ensemble methods?*

We investigate the use of heterogeneous ensemble methods for the prediction of number of faults and compare their performance with homogeneous ensemble methods such as bagging and boosting.

**RQ3:** *Does selection of combination rules affect the performance of ensemble methods?*

We assess two linear combination rules based approaches and two non-linear combination rules based approaches to combine the base learner outputs for ensemble methods.

**RQ4:** *Is accuracy of inter-releases fault prediction comparable with intra-release fault prediction?*

We evaluate the performance of presented ensemble methods under two different scenarios, intra-release prediction and inter-releases prediction by performing experiments using five-fold cross-validation on fifteen software fault datasets.

The rest of the paper is organized as follows. Section 2 reviews the work related to the prediction of number of faults and ensemble methods. Section 3 presents an overview of our proposed ensemble method. The details of the experimental setup are provided in the Section 4. The results are presented in Section 5. Section 6 presents the comparative analysis of ensemble methods with the individual participating fault prediction techniques and with homogeneous ensemble methods. Section 7 examines the applicability of proposed ensemble methods to the Eclipse project. Section 8 presents the discussion of the results. The possible threats to the validity are given in Section 9, followed by the conclusions in Section 10.

## 2. Related work

Prediction of number of faults in a given software system using machine learning and statistical techniques has been done by a number of researchers earlier. Most of these studies include the techniques such as generalized linear regression [26], Poisson regression [14], negative binomial regression [13], genetic programming [27], neural network [20], etc.

Grave et al. [26] evaluated generalized linear regression (GLR) for predicting number of faults in the given software system. The experimental investigation has been carried out over a large telecommunication system having various software change metrics. The results revealed that module age, changes made to a module, and age of the change metrics together produced the best fault prediction accuracy. While, other used software metrics such as size of a module and complexity were not significantly correlated with fault-proneness. The study did not provide any evaluation of GLR technique for the prediction of number of faults.

Ostrand et al. presented a negative binomial regression (NBR) model for the number of fault and fault density prediction in the given software systems [28]. The presented fault prediction model was built and evaluated for two industrial software systems having different software file characteristics and LOC metrics. The re-