Method level bug prediction: An overview

SATTI. SONIYA
Master's in the specialization of Data Science at JNTUGV, College of Engineering, Vizianagaram.
Email: soniyaanu80@gmail.com

What is a bug?

A bug is a flaw, mistake, or failure in the program or system being developed that leads to unanticipated outcomes.

What is software bug prediction?

The modules that are defect-prone and need thorough testing are identified using software defect prediction. In this manner, it is possible to use the testing resources effectively while yet violating the limitations.

Why software defect prediction is important?

To boost software reliability, software defect prediction has grown to be an important area of study. The usage of program defect predictions helps developers to spot prospective issues and to make the most use of testing resources to increase program dependability.

Context

Researchers have been working to forecast the presence of software defects for many years. The studies attempt to forecast problems in various software structures, ranging from single files (Units) and groups of files (Packages) as well as larger systems (Modules). Very few research have examined the possibility of predicting issues with methods, which are even smaller structures.

Different levels of bugs predictions

Bug predictions can be done at various levels like File level, package level, Module level, and Method level

File-level bug prediction: Bug predictions at the file level concentrate on specific source code files inside a project.

Package-level bug prediction: This makes it easier to spot packages that are more likely to have issues in them.

Module-level bug prediction: Within a software system, modules represent logical units of functionality. At the module level, bug predictions concentrate on evaluating each module's level of complexity, interdependencies, and previous bug data.

Method-level bug prediction: Within a module, methods are discrete operations or steps that carry out specific responsibilities. At the method level, bug predictions are made by looking at variables including method complexity, code metrics, parameter usage, and previously reported bugs for each method. The likelihood of defects in particular approaches can be predicted by looking at these parameters.

Method-level bug prediction approach was applied to a large-scale project named Avro available at the Git hub repository. Ran Mo, Shaozhi Wei, Qiong Feng, and Zengyang Li [1] first provided a set of code metrics and history measures to be the attributes for prediction; next, they labeled data samples (i.e., methods) as being bug-prone or not; last, they proposed a set of code metrics and history measurements; To automate the calculation of the aforementioned history measures and code metrics, they developed the Fine-grained Code Metrics and History Measures Extractor (FCHE)[1]. We used ten-fold cross-validation to develop and test bug prediction models using Random Forest. For this work, we conduct predictions by using a Weka tool, which is a collection of machine learning algorithms. To assess the effectiveness of our bug prediction model at the method level, we applied AUC or Area Under the ROC Curve.

WEKA - Waikato environment for knowledge analysis

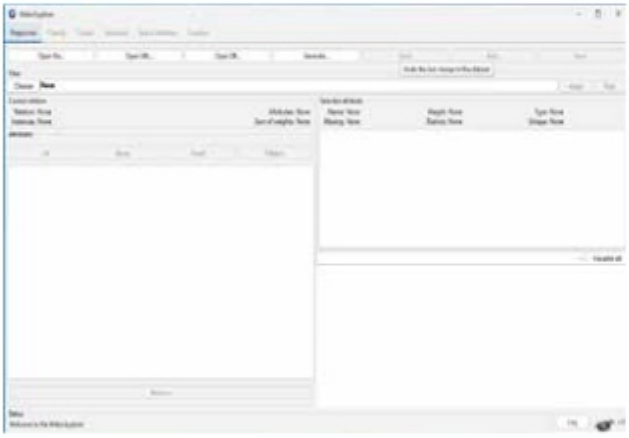
About Weka

Weka is an ensemble of machine learning algorithms. It has tools for preprocessing, classification, Clustering, association, selection of attributes and visualization. Weka tool is easy to preprocess any data from Files, URLs, and databases. And this tool is used to compare our results with other classifiers very fastly and accurately.

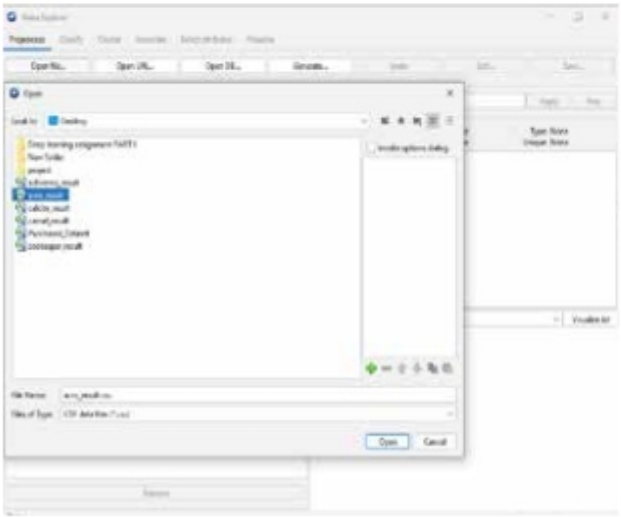
https://doi.org/10.5281/zenodo.8250366

For this work start the WEKA GUI Chooser application

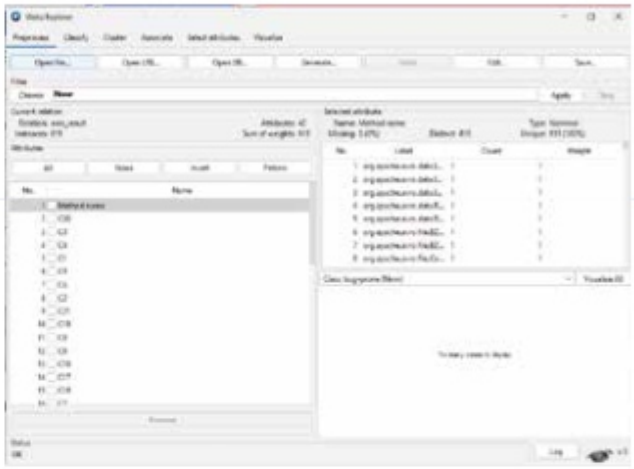
Then click on the explorer button, when opening the explorer, preprocess tab is activated. Preprocess is the first step in machine learning, then select a file and process it and make it ready for applying the machine learning algorithms.



Click on the open file button. A directory window opens



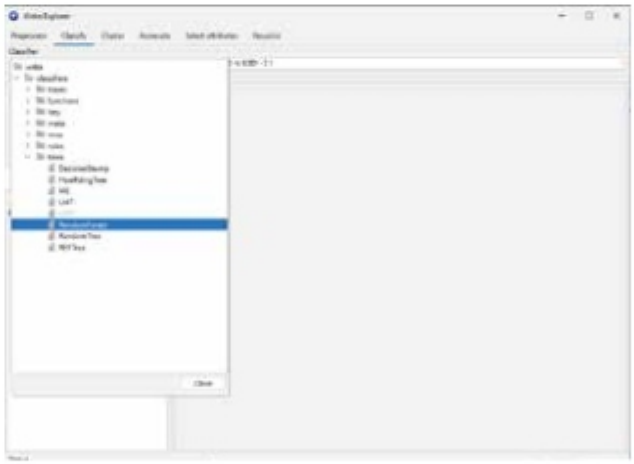
After opening the file, the **Current relation** sub-window the name of the file that is loaded and there are 619 instances and the dataset contains 42 attributes.



The avro dataset contains 42 attributes which include Method name, code metrics(21) and history measures(19), and bug-prone which are displayed on the left side.

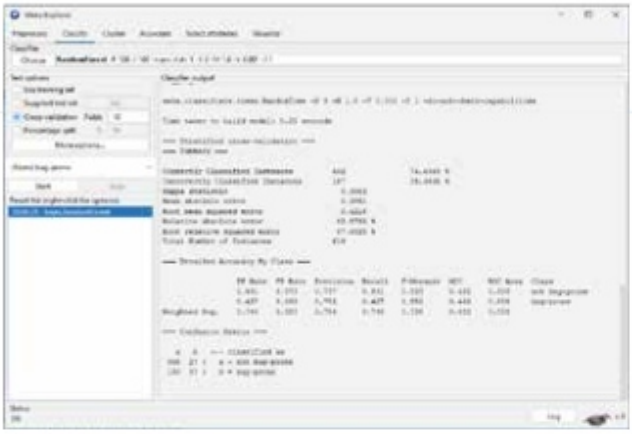
Test data

Click on the Classify tab, and select Random Forest to create models for method-level bug prediction. A random forest is a collection of decision trees that may be used to create classification prediction models. Random forest uses decision trees to improve accuracy and reduce over-fitting.



Click on the start button to start classification With Random Forest, we employed the ten-fold cross-validation method to train and validate bug prediction models. The cross-validation technique divides the original data into a training set and a test set to determine how well a predictive model performs in practice. The original dataset will be randomly split into 10 equal-sized sections for ten-fold cross-validation. The technique will

then be applied iteratively ten times, with nine subsets being utilized for training and one for validation. A subset will only be utilized precisely once as the validation data during each iteration. A single estimation would then be presented using the results of 10 folds.



To assess the accuracy of our bug prediction models at the technique level, we utilized AUC or Area Under ROC Curve. When prediction models are used on uneven data, transitional measures like Accuracy may not perform very well. The Area Under ROC Curve (AUC) could accurately depict the performance of prediction models developed on unbalanced data, which is similar to our dataset where the distributions of two classes (bug-prone or not bug-prone) are imbalanced.

Results

Going into the analysis of these results shows the Roc area as 0.808 and According to the prior studies, an AUC value larger than or equal to 0.7 indicates the best prediction.

According to the confusion matrix, the correctly classified instances are 462 and the incorrectly classified instances are 157.

As per the results, the Area under the Roc curve gives the best prediction results when the dataset is imbalanced.

Dataset	Precision	recall	F-measure	MCC	ROC area
Avro_ result	0.754	0.746	0.724	0.432	0.808

Comparing with other datasets

Dataset	Precision	recall	F-measure	MCC	ROC area
calcite_ result	0.652	0.649	0.644	0.297	0.710
flume_ result	0.725	0.721	0.693	0.368	0.800
zookeeper_ result	0.697	0.694	0.695	0.389	0.793

A common metric for unbalanced datasets is the AUC-ROC. The performance of the model in separating positive and negative samples is better indicated by a higher AUC-ROC score.

Accuracy may not be the most reliable metric for assessing model performance when working with datasets that are imbalanced, where the distribution of samples across classes is noticeably unbalanced.

Comparison of different classifiers in terms of AUC

Classifiers	Datasets			
	Avro_result	calcite_ result	flume_ result	zookeeper_ result
Logistic	0.714	0.658	0.736	0.746
Bayes	0.704	0.624	0.735	0.725
Decision tree	0.720	0.660	0.682	0.714

All AUC scores are higher than 0.5, which means that the proposed classifiers could achieve acceptable results. The Random Forest algorithm had the best result with the given datasets.

References

[1] "An exploratory study of bug prediction at the method level," *An exploratory study of bug prediction at the method level - ScienceDirect*, Dec. 07, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0950584921002330?via%3Dihub

[2] "Software defect prediction using cost-sensitive neural network," *Software defect prediction using a cost-sensitive neural network - ScienceDirect*, Apr. 30, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S1568494615002720

About the Authors

Satti Soniya, currently pursuing her Master's in the specialization of Data Science at JNTUGV, college of Engineering, Vizianagaram. She is completed her B.Tech at West Godavari Institute of Science and Engineering College. She is good at Python, Deep Learning and Machine learning. She is always interested to learn new technologies and better approaches to upgrade her skills.