International Journal of Scientific Research & Engineering Trends
Volume 7, Issue 2, March-April-2021, ISSN (Online): 2395-566X

A Review on Software Fault Detection using a Classification Model with Dimensionality Reduction Technique

Research Scholar Devangi Paneri
Department of Computer Engineering,
V.V.P Engineering College,
Rajkot, India
15cse101@gardividyapith.ac.in

Research Scholar Mansi Chauhan
Department of Computer Engineering,
Gujrat Technological University,
Rajkot, India
mansih1998@gmail.com

***Abstract*- Software plays the most important role in every organization it requires high-quality software. If a fault happens in this system then it causes high financial costs and affects people's lives. So, it is important to develop fault-free software. Sometimes, a single fault can cause the entire system to frailer. So in the SDLC life cycle Fault prediction at an early stage is the most important activity it helps in effectively utilize the resources for better quality assurance. So before delivering the software to market it is important to identify defects in the software because it increases the customer satisfaction level. here in this survey paper present an ensemble approach to identifying fault before delivering the software. Ensemble classifier improved classification performance compared to the single classifier. So improved the accuracy the new algorithm is proposed that is "improved random forest" it works with random forest classifier with filter-based feature selection method. The feature selection method reduces the dimensionality and selects the best subset of features and gives that subset to the random forest classifier. The experiment carried the public NASA dataset of the PROMISE repository.**

***Keywords*- Data mining, machine learning, software fault, dimensionality reduction technique, improved Random forest.**

I. INTRODUCTION

Software engineering is an emerging area in data mining. Software is used by every end-user and every organization. If failure happens in the software then it causes high financial costs and affects people's lives. So the quality of software must be maintained. Failure happens in software because of the ambiguities in requirements, design, code, and testing cases. In the SDLC life cycle testing process of software, quality is more expansive and requires huge resources but machine learning techniques are giving promising ways to predict software fault in the early stage. A software defect detection model is used for identifying the faulty component.

To detect the fault Different classification models are used such as random forest, support vector machine, Neural Network, Genetic programming, Naïve Bayes approach, fuzzy logic, decision trees, and also different feature selection techniques are used to reduce the dimensionality like ranking method, filter method, wrapper method, embedded method, and correlation-based method.

These methods are used to preprocess the data. Software matrices are used to detect the fault and matrix compute from the source code. Matrix-like size metrics, object-oriented metrics, and complexity metrics are used. Identifying the fault before the testing improves the software quality. Hear Public NASA datasets are used for software defect detection. PROMISE repository includes several public NASA datasets. The model used historical data for training the classifier.

The main objective of the paper is to design a new hybrid approach for software defect detection that is "Improved random forest." it is the combination of Random forest classifier and feature subset method. Data preprocessing perform to identify the most relevant feature. Classification of fault data measure using the accuracy, ROC, Area of ROC curve, recall, precision.

II. RELATED WORK

Feature selection is a technique used to select an optimal feature subset from the original input features according to a specific criterion. It is a kind of data preprocessing technique for selecting a subset of the best software metrics before the construction of a defect model and it reduces the redundant and irrelevant features without much loss of the facts and details [1].

Burcu Yalciar and Merve ozdes [2] are compared seven machine learning algorithms that are Bagging, Random Forests (RF), Multilayer Perceptron (MLP), Radial Basis Function(RBF), Naive Bayes, Multinomial Naive Bayes, and Support Vector Machine (SVM).with four NASA datasets which are PC1, CM1, KC1, and KC2. Out of 7 machine learning algorithms, Random forest gives the best output.

Kulamala Vinod Kumar, Priyanka Kumari, Avishikta Chatterjee, and Durga Prasad Mohapatra [3] compared Random forest with 5 models such as SVM (linear), SVM(degree 2), SVM(RBF kernel), Decision tree, BPNN With ar1,ar3,ar5 dataset the results is that Random forest model predicts better faulty modules then others also measures accuracy and F-measure metrics of all models.

Shamsuddeen M. Abubakar, Zahraddeen Sufyanu, Abubakar M. Miyim has gives a brief study of The three most important Feature selection techniques are filter, wrapper, and embedded. The embedded technique is more powerful than others [4].

Kalai Magal. R, Shomona Gracia Jacob, build a new algorithm that is improved random forest with combine random forest and correlation-based feature selection method. Correlation-based Feature Subset Selection algorithm selects the optimal subset of features. The optimal features are fed as a part of Random Forest classification to give better accuracy in software defect prediction.

The accuracy of the Random Forest classifier for PC1 data is 92.953%. For PC2 data the Random Forest classifier performance in terms of accuracy is 98.346% For PC3 data the Random Forest classifier performance in terms of accuracy is 89.208% and for PC4 data is 88.267% . Where The accuracy of the Improved Random Forest classifier for PC1 data is 94.5%. For PC2 data the Improved Random Forest classifier performance in terms of accuracy is 98.5%. For PC3 data the Improved Random Forest classifier performance in terms of accuracy is 89.6% and for PC4 data is 90.6% using WEKA tools.

II. FEATURE SELECTION APPROACHES

All the attributes which are given in the data set are known as "Features" or "characteristics of the data set.[6] Feature selection is a data preprocessing technique for selecting a subset of the best software metrics before the construction of a defect model. Feature selection has been widely used in software engineering to remove irrelevant metrics (i.e., metrics that do not share a strong relationship with the outcome) and correlated metrics [7].

So selecting the best subset of features from the given set of features to remove the redundant and irrelevant features without much loss of the facts and details it's called Feature selection is the mechanism.

The selection of features can be achieved in two ways, feature ranking and features subset selection. Feature subset selection can be divided into three models: filters, wrappers and embedded. All feature selection models have their advantages and drawbacks.

1. Ranking Method:
Feature Ranking is the process of ordering the features by the value of some scoring function, which usually measures feature-relevance [24].

The ranking method is the simple method for feature selection to select the k highest-ranked features according to S. This method is usually not optimal, but often preferable to others, It is the computationally efficient only calculation and sorting of n scores [24].

2. Subset Method:
The filter approach can be used to select the best subset of features from high-dimensional datasets without using a learning algorithm. Filter-based feature selection methods are faster compared to the wrapper and embedded method but the classifier accuracy is not ensured [8]. Whereas, In wrapper methods, a search Strategy iteratively adds or removes features from the data to search the best possible features subset that maximizes accuracy[7]. but it is highly complex and time-consuming but it gives good accuracy compared to the filter method. Where Embedded approach combines the qualities of filter and wrapper methods.

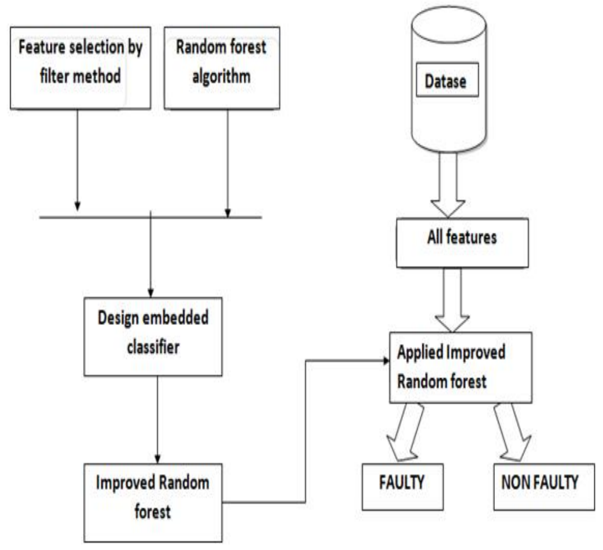
A decision tree is also be considered an embedded method, construction of the tree and the selection of the features are interleaved. The selection of the feature in each iteration is usually done by a simple filter or ranker. it performs feature selection during the modeling algorithm's execution [7].

A hybrid approach is a combination of both filter and wrapper-based methods. Here, a filter approach selects a candidate feature set from the original feature set and the candidate feature set is defined by the wrapper approach. It exploits the advantages of both these approaches [8].

Filter methods are use ANOVA, personal's correlation, LDA, and chi-square methods to select a useful set of features. Where wrapper method is used Forward Selection, Backward Elimination, Recursive Feature Elimination, and embedded method use LASSO regression it means Least Absolute Shrinkage and Selection Operator.

III. PROPOSED WORK

Fig 1. Shows the proposed framework of the system. in NASA dataset there are many features but all features are not useful to detect the fault in the system some feature are irrelevant and some are noisy so applied some feature selection method with classifier. To hear, in embedded classifier one feature selection method is used that is filter method with random forest classifier.


Feature selection by filter method
Random forest algorithm
Datase
All features
Design embedded classifier
Applied Improved Random forest
Improved Random forest
FAULTY
NON FAULTY
Fig 1. Proposed Work.

1. Design of Embedded Classifier:
(The Embedded classifier is designed by adding a new algorithm in the weka interfaced with NetBeans.) The algorithm is implemented by filter-based Feature subset selection algorithm with Random Forest algorithm to give better accuracy.

The performance evaluation is done by Repeatable Randomly train-test splits to obtain better accuracy. The filter method selects the most optimal subset of features and passes to the Random Forest thus it takes the optimal subset of features and gives better accuracy in classifying the faulty and non-faulty software.

IV. DATASET AND TOOLS

To evaluate the proposed framework we used the real-world software NASA dataset. Datasets are freely available online. The dataset consists of the software metrics.

1. Software Metrics:
Software Metrics measure a different aspect of the software complexity used for analyzing and increase software quality.

There are two types of metrics: process metrics and product metrics. Product metrics are used for the software defect detection model. Product metrics include size metrics, complexity metrics, and object-oriented metrics.

2. Size Metrics:

2.1 **LOC (Line of code):** This measure quantifies software complexity. It measures the length of the program by counting the number of lines in the program. It is counting the total length of code including the statement, comment, Blank Space. i.e. the higher the LOC higher the bug density.

2.2 **Number of Statement:** It counts the total number of a statement in the program. The statement includes continuing, do, if, for, constructor call, return, switch, try and catch, while, finally, break, pre-post increment or decrement, methods call.

2.3 **Number of Comment:** It counts the total number of comment lines in the program. There are two types of comment multiple-line comment and single- line comment. A good program includes more comments in the program. If comment less than 30% means the program is not good explain.

3. Complexity Metrics:
Count the complexity of the program there are two types of matrices used.

3.1 **McCabe Complexity:** It is counted as cyclomatic complexity. it measures quantitative of the linear independent path from source code. Cyclomatic complexity number V(G) is given below:

$$V(G) = E - N + P$$

Where,
E = total number of edges in graphs G
N = total number of nodes in graphs G
P = total number of disconnected parts of graph G.

3.2 **Halstead Complexity:** It measures the operands and operator of the program code. it is highly correlated to the number of lines of code. The measures are: n1=count of the unique operator n2=count of unique operands N1=total occurrence of operator N2=total occurrence of operands.

4. Object-oriented Metrics:
Object-oriented metrics useful in the design and implementation phase;

4.1 **Weighted Method per class (WMC):** This measure counts a number of the method per class.

4.2 **Depth of Inheritance Tree (DIT):** This measures the count number of a class level in the tree. hear root class is considered as Zero. If the length of the inheritance tree is higher than the behavior prediction of that class is complex.

4.3 **Number of Children (NOC):** These measures count the number of immediate subclasses in the tree. If a class has many children then requires more testing.

4.4 **Coupling between Object classes (CBO):** These measures count the number of other classes in which class is coupled. This type of coupling is achieved by inheritance, arguments passing, return types, method calls, field accesses, and exceptions.

4.5 **Response for a Class (RFC):** The number of local methods plus the number of non-local methods called by local methods.

V. PERFORMANCE EVALUATION

The main aim of most classifiers is to perform binary classification, i.e., Faulty or Non- Faulty. The performance measures used are accuracy, confusion metrics, Area under the ROC curve.

1. Accuracy:
Accuracy is simply the rate of correct classification. It is the ratio of the number of correct predictions upon a total number of predictions.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

2. Area under ROC Curve (AUC):
To measure the area under the curve Receiver operating characteristics (ROC) is the plot. The receiver operating characteristics (ROC) curve is a graphical representation of the performance of the binary classifier. The curve is created by the true positive rate against the false-positive rate at the different threshold values. AUC is given better results for software defect detection.

3. Confusion Matrix:
The confusion matrix shows the performance of the classifier. it known as the error matrix. The confusion table represents the true value identity from the set of test data. The confusion matrix has basic categories which are True positive, True Negative, False Positive, False Negative. Table 5.1 shows the confusion table.

From the confusion table another evaluation Measures Count which are:

3.1 Recall: Recall is the ratio between, correctly predicted positive observations are divided by the all observations in the actual class and it can be defined as below:

$$Recall = \frac{TP}{TP + FN}$$

3.2 Precision: Precision is the ratio between correctly predicted positive observations is divided by all observations in the predicted class and it can be defined as below:

$$Precision = \frac{TP}{TP + FP}$$

3.3 F-measure: - F-measure is the weighted harmonic mean of the precision and recall of the test.

$$Precision = \frac{2 * Recall * Precision}{Recall + Precision}$$

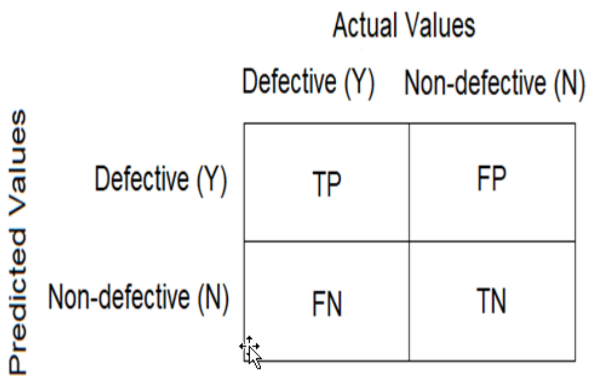


Fig 2. Confusion Matrix.

VI. CONCLUSION

This study proposed a feature selection-based Random Forest model for software defect prediction. Filter-based feature subset selection technique was used to select significant features which were helpful to predict defects in software modules.

The experimental results revealed that the predictive capability of the proposed approach is better or at least comparable with other approaches. This research discloses the effectiveness of the feature subset selection- based Random Forest approach in predicting defective software modules and suggests that the proposed model can be useful in predicting defective software based on the important attributes.

REFERENCES

[1] Shamsuddeen M. Abubakar, Zahraddeen Sufyanu, Abubakar M. Miyim, A Survey of Feature Selection Methods for Software Defect Prediction Models, fudma Journal of Sciences (FJS), Vol. 4, March 2020, pp 62-68.
[2] Burcu Yalciar, Merve ozdes "Software Defect Estimation Using Machine Learning Algorithms" IEEE (UBMK'19) 4rd International Conference on Computer Science and Engineering, pp. 487- 491, 2019.
[3] Kulamala Vinod Kumar, Priyanka Kumari, Avishikta Chatterjee, and Durga Prasad Mohapatra, "Software Fault Prediction Using Random Forests", Springer, pp. 95-103, 2020,
[4] Shamsuddeen M. Abubakar, Zahraddeen Sufyanu, Abubakar M. Miyim, A Survey of Feature Selection Methods for Software Defect Prediction Models, fudma Journal of Sciences (FJS), Vol. 4, March 2020, pp 62-68.
[5] Kalai Magal. R, Shomona Gracia Jacob, Improved Random Forest Algorithm for Software Defect Prediction through Data Mining Techniques, International Journal of Computer Applications (0975 – 8887), Vol.117, PP 18-22, 2015.

[6] Amit Kumar Jakhar and Kumar Rajnish, Software Fault Prediction with Data Mining Techniques by Using Feature Selection Based Models, International Journal on Electrical Engineering and Informatics, Volume 10, pp 447-465, September 2018.
[7] Shamsuddeen M. Abubakar, Zahraddeen Sufyanu, Abubakar M. Miyim, A survey of feature selection methods for software defect prediction models, FUDMA Journal of Sciences (FJS) Vol. 4 No. 1, March 2020, pp 62 -68.
[8] Nazrul Hoque Mihir Singh Dhruba K. Bhattacharyya, an ensemble feature selection method for classification, springer, 10 October 2017, pp 1-14.
[9] K. Gao, T. M. Khoshgoftaar, H. Wang, and N. Seliya, "Choosing software metrics for defect prediction: An investigation on feature selection techniques," Softw.-Practice Exper., 2011, 41(5), 579–606.
[10] A Soleimani, F. Asdaghi, "An AIS Based Feature Selection Method for Software Fault Predictions" in Iranian conference on intelligent system (ICIS) IEEE, Iran, 2014.