iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 5 **Issue:** VIII **Month of publication:** August 2017

DOI: http://doi.org/10.22214/ijraset.2017.8084

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor:6.887
Volume 5 Issue VIII, August 2017- Available at www.ijraset.com

Review on Machine Learning Framework for Software Defect Prediction

Jamshiya P K
Student, Department of Computer Science and EngineeringJyothi Engineering College, Thrissur, India

Abstract: ***Software defect prediction is the process of tracing defective components in software prior to the start of testing phase. Defect prediction leads to low cost, development time, reduced rework effort, increased customer satisfaction and more reliable software. Therefore, defect prediction practices are important to achieve software quality and to learn from past mistakes. A predictive model is constructed by using machine learning approaches and classified them into defective and non-defective modules. Machine learning techniques help developers to obtain useful information after the classification and enable them to analyze data from different perspectives. The various classification methods used are Naive Bayes, SVM, J48 and Random forest. These classification methods in defect prediction system can provide high prediction performance and better accuracy. Parameters selected for performance comparison includes Accuracy, Mean absolute error and F- measure.***
Keywords: Software Defect Prediction; Machine Learning; Classification; Naive Bayes; J48; SVM; Random Forest.

I. INTRODUCTION

A fault or defect in any software system can cause the software system to fail to perform what it's actually supposed to perform. Knowing the possible causes of software defects early, could help on planning, controlling and executing software development activities. The Software Quality Assurance is a set of activities that ensures software system to meet a specific quality level. Most business organizations targeted towards customer satisfaction and profitable growth, are being met through increasing use of software. A minor defect, or even inefficiency, in the software may lead to not only loss of money and effort, but loss of customer base. Using software defect prediction model, one can able to identify potential defect prone software, predict number of defects, Identify possible causes of defects.

Machine learning techniques can be used to analyse data from different viewpoints and enable developers to retrieve useful information. Classification is a data mining and machine learning approach, useful in software bug prediction which categorizes software modules into defective or non-defective. The prediction is then made based on this classification results. The various classification methods used are Naïve Bayes, Random Forest, J48 and SVM. These methods for software defect prediction are analysed based on parameters such as accuracy, F-measure and Mean absolute error.

Waikato Environment for Knowledge Analysis (Weka) is used to generate the result. WEKA, developed at the University of Waikato in New Zealand, is open-source data mining software in Java. It is comprised of a collection of algorithms for data mining tasks, including data pre-processing, association mining, classification, regression, clustering, and visualization together with graphical user interfaces for easy access to these functions. Nowadays, WEKA is recognized as a landmark system in data mining and machine learning.

II. RELATED WORK

Rajni Jindal, Ruchika Malhotra and Abha[1] Jain proposed a system that uses radial basis function of neural network to predict the defects at various levels of severity. The proposed tool will first extract the relevant information from PITS database using a series of text mining techniques. After extraction, the tool will then predict the defect severities using machine learning techniques. Qunbao song et al [3] developed a general software defect proneness prediction framework that consists of two components such as scheme evaluation and defect prediction. At the scheme evaluation phase, the performances of the various learning schemes are assessed with historical data to determine whether a certain learning scheme performs sufficiently well for prediction purposes or to select the best from a set of competing schemes. At the defect prediction phase, according to the performance report of the first stage, a learning scheme is selected and used to construct a prediction model and predict software defect. Okutan and Yildiz[4] used Bayesian networks to determine the probabilistic influential relationships among software metrics and defect proneness. In addition

to the metrics used in promise data repository they defined two more metrics such as NOD for number of developers and LOCQ for source code quality.

III. PROPOSED SYSTEM

The proposed framework of software defect prediction consists of training data set and user input as test dataset. The most common components that we used are Instances, different classifiers and methods for evaluation. A training data set is essential for the working of the system since the system rely on supervised learning. Supervised learning is the machine learning task of deducing a function from labelled training data. The training data is comprised of a set of training examples. A supervised learning algorithm examines the training data and generates an inferred function, which can be used for mapping new examples. An ideal state will allow for the algorithm to correctly determine the class labels for unseen instances. This requires the learning algorithm to generalize from the training data to hidden situations in a reasonable way. The training data obtained from promise data repository is the training example. It consists of the class label and its corresponding value. Naive Bayesian, Random Forest, SVM and J48 classifiers are supervised learning algorithms. They learn from the provided training examples.

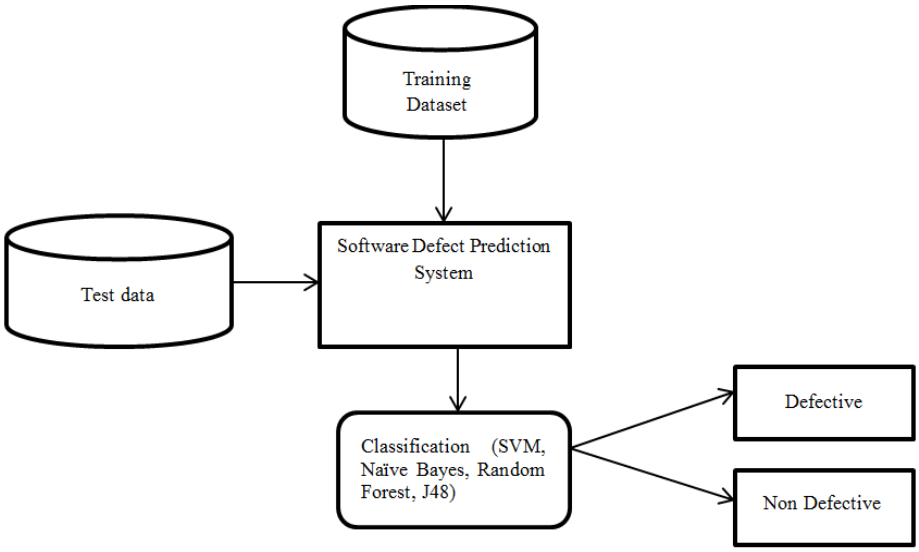


Fig.2 Proposed model

When a new instance with same attributes as in training data with different values other than those in the training example comes, these algorithms correctly classify the new instance based on the generalization created from the training set. Naive Bayes, Random forest, SVM and J48 decision tree are classifiers.

A. Classification

Classification is the problem of recognizing to which of a set of groups a new observation belongs, on the basis of a training set of data containing observations whose category membership is known. Classification is considered as an instance of supervised learning. That is learning when a training set of correctly identified observations are available.

B. Random Forest

Each tree is trained on approximately 2/3rd of the total training data. Cases are taken at random with replacement from the original data. This section will be the training set for developing the tree.

Out of all the predictor variables some predictor variables (K) are selected at random and the best split on these K is used to split the node. K is square root of the total number of all predictors for classification .The value of k is held constant during the forest development. In a standard tree, each split is created after examining every variable and picking the best split from all the variables.

For each tree, using the remaining data, calculate the misclassification rate which is the out of bag (OOB) error rate. Combine error from all trees to define overall OOB error rate for the classification.

Each tree gives a classification which is said to be "votes" for that class. The forest picks out the classification having the most votes over all the trees in the forest. The vote will be YES or NO for a binary dependent variable and we count up the YES votes. This is the RF score and the percentage of YES votes obtained is the predicted probability.

C. J48

J48 algorithm uses pruning method for the tree construction. Pruning is a technique that reduces size of tree by removing over fitting data, which leads to poor accuracy in predictions. The J48 algorithm recursively classifies data until it has been categorized as perfectly as possible. This technique gives maximum accuracy on training data. The overall concept is to build a tree that provides balance of flexibility and accuracy.

D. SVM

SVM is a supervised machine learning algorithm that can be used for the purpose of classification. It is used to train a model that assigns new unseen objects into a particular category. This is achieved by creating a linear separation of the feature space into two categories. Based on the features in the new unseen objects, it places an object "above" or "below" the separation plane, leading to a categorization (e.g. defective or non-defective).

E. Naives Bayes

Naive Bayes classification algorithm is based on baye's theorem. It is assumed that given the class variable, the value of a particular feature is independent of the value of any other feature. It is considered that these features contribute independently to the probability regardless of any possible correlations.

IV. EXPERIMENTAL EVALUATION

This section describes the performance of proposed software defect prediction model. The performance of the different classifier is evaluated using the native methods in Weka. Percentage of correctly classified instances, incorrectly classified instances, precision, recall and confusion matrix is obtained by the cross validate model.

Datasets	Naives Bayes	Random Forest	J48
Ant	81.0738	82.4161	79.3289
KC1	82.4427	86.4027	84.1603
PC4	86.9192	90.8506	89.564
Eclipse_JDT_Core	82.6479	84.9549	81.8455

V. CONCLUSION

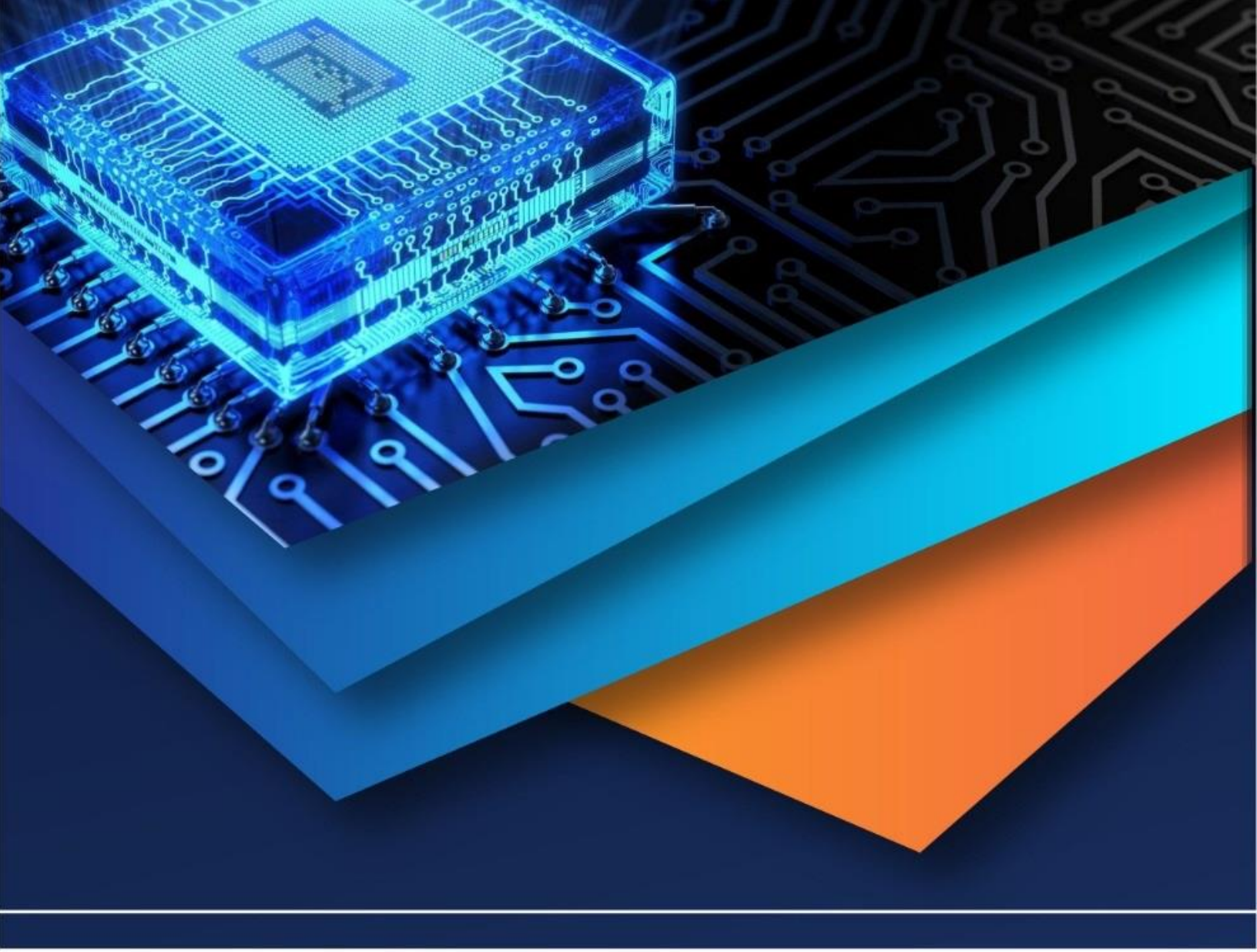
In this paper, we have presented a machine learning framework for software defect prediction. The framework involves evaluation and prediction using various classification algorithms. Different learning schemes are evaluated and the best one is found out based on performance measures such as accuracy, mean absolute error and F- measure. The datasets for the experiment was taken from the promise data repository. The best learning scheme is used to build a predictor with all historical data and the predictor is finally used to predict defect on the new data. Waikato Environment for Knowledge Analysis (Weka) is used for generating the result. The various classification methods used are Naive Bayes, SVM, J48 and Random forest.

VI. ACKNOWLEDGMENT

First and foremost, I express my thanks to The Lord Almighty for guiding me in this endeavour and making it a success. I take this opportunity to express my cordial obligation to all respected personalities, who had guided encouraged and helped me in the successful finalization of this work.

REFERENCES

[1] Rajni Jindal, Ruchika Malhotra, Abha Jain, "Software Defect Prediction Using Neural Networks," International conference on reliabilityinfocom technologies and optimization, 2014 IEEE.
[2] Saiqa Aleem1 , Luiz Fernando Capretz and Faheem Ahmed, "Benchmarking Machine Learning Techniques For Software Defect Detection," International Journal of Software Engineering & Applications (IJSEA), Vol.6, No.3, May 2015.
[3] Qinbao Song, Zihan Jia, Martin Shepperd, Shi Ying, and Jin Liu, "A General Software Defect-Proneness Prediction Framework," Ieee Transactions On Software Engineering, Vol. 37, No. 3, May/June 2011.
[4] Ahmet Okutan·Olcay Taner Yıldız, "Software defect prediction using Bayesian networks," Empir Software Eng (2014) 19:154–181.
[5] Reena P, Binu Rajan, "Software Defect Prediction System –Decision Tree Algorithm With Two Level Data Preprocessing," International Journal of Engineering Research & Technology (IJERT) Vol. 3 Issue 3, March – 2014.
[6] Zi Yuan1, Lili Yu1, Chao Liu , Linghua Zhang, "Predicting Bugs in Source Code Changes with Incremental Learning Method," Journal Of Software, Vol. 8, No. 7, July 2013.
[7] M. Karthikeyan, Dr. S. Veni "Software Defect Prediction Using Improved Support Vector Machine Classifier," International Journal of Mechanical Engineering and Technology (IJMET) Volume 7, Issue 5, October 2016.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY