Industrial Engineering Journal
ISSN: 0970-2555
Volume : 52, Issue 7, July : 2023

A NOVEL APPROACH FOR MACHINE LEARNING-BASED DEFECT PREDICTION FOR SOFTWARE

M.SUSMITHA SRIVALLI, MCA, DCA, DVR & Dr.Hima Shekar MIC College of Technology, A.P., India.

MOUNIKA.S , Assistant Professor, Dept.of AI & IT, DVR & Dr.Hima Shekar MIC college of Technology, A.P., India.

Abstract— software has become increasingly vital in our daily lives. Software engineering research is centered on defect prediction. Successful software development requires better communication between data mining and software engineering. Software Engineering is a comprehensive domain since it requires a tight communication between system stakeholders and delivering the system to be developed within a determinate time and a limited budget. Delivering the customer requirements include procuring high performance by minimizing the system. Thanks to effective prediction of system defects on the front line of the project life cycle, the project's resources and the effort or the software developers can be allocated more efficiently for system development and quality assurance activities. All in all, the results of ensemble learners category consisting of Random Forests (RF) and Bagging in defect prediction is pretty much its counterparts.

INTRODUCTION

Software defect prediction is a pre-testing technique that estimates where bugs will show up in the code. The purpose of software defect prediction research is to identify potentially flawed parts of a programme before it reaches the testing phase. The primary benefit of these prediction models is that they need more testing time and money. may be directed to the modules most prone to errors. However, only a few mobile app-specific software defect prediction algorithms currently exist. It is common practise to utilise defect prediction algorithms to probe the impact domain in software (clustering, neural networks, statistical methods, and machine learning models). This research aims to examine and compared various ML (machine learning) algorithms for software bug prediction. Developing a software system is an arduous process which contains planning, analysis, design, implementation, testing, integration and maintenance. A software engineer is expected to develop a software system on time and within limited the budget

which are determined during the planning phase. During the development process, there can be some defects such as improper design, poor functional logic, improper data handling, wrong coding, etc. and these defects may cause errors which lead to rework, increases in development and maintenance costs decrease in customer satisfaction. A defect management approach should be applied in order to improve software quality by tracking of these defects. In this approach, defects are categorized depending on the severity and corrective and preventive actions are taken as per the severity defined. Studies have shown that 'defect prevention' strategies on behalf of 'defect detection' strategies are used in current methods. Using defect prevention strategies to reduce defects generating during the software development the process is a costly job. It requires more effort and leads to increases in project costs.

LITERATURE REVIEW

Basili et al (1996) [2] have utilized calculated relapse to inspect what the impact of the set-up of item situated plan measurements is on the expectation of shortcoming inclined classes. Khoshgoftaar et al (1997) [3] have utilized the neural organization to group the modules of enormous media transmission frameworks as flaw inclined or not and compared it to a non-parametric discriminant model. To identify software problems. Ceylan et al. (2006) [4] proposed a decision tree, radial bias function and multilayer perceptron based model which was applied on three huge companies in Turkey and detected the flaws in the software, which lead the companies to make necessary changes. Elish et al. (2008) [5] applied SVM on the four NASA data sets [12] against other machine learning algorithms and realizedSVM's performance was superior to that of others. In this experiment, the study of Malhotra et. al. (2015) [10] have been guiding us while deciding to select which machine learning algorithms we have used for defect prediction in software systems. They categorized the machine learning algorithms based on distinct learners such as Ensemble Learners, Bayesian Learners, Neural Networks and SVM. According to these categories, we selected seven different machine learning algorithms to estimate software defect.

RELATED WORK

1. Bayesian Learners:

• Naive Bayes: Naive Bayes which is one of the most commonly used algorithms for classifying problems is simple probabilistic classifier and is based on Bayes Theorem

Multinomial Naïve Bayes: Multinomial Naive Bayes classifier is obtained by enlarging Naive

Bayes classifier. Differently from the Naive Bayes classifier, a multinomial distribution is used for each features.

2) Ensemble Learners:

• Bagging: This algorithm which is introduced by Leo Bremen and also called Bootstrap Aggregation is one of the ensemble methods. In this approach, N sub-samples of data from the training sample are created and the predictive model is trained by using these subset data. Sub-samples are chosen randomly with replacement. As a result, the final model is an ensemble of different models.

• Random Forest: Random Forest algorithms which also called random decision forest is an ensemble tree-based learning algorithm. It makes a prediction over individual trees and selects the best vote of all predicted classes over trees to reduce overfitting and improve generalization accuracy. It is also the most flexible and easy to use for both classification and regression.

3) Neural Networks:

• Multilayer perceptron: Multilayer Perceptron which is one of the types of Neural Networks comprises of one input layer, one output layer and at least one or more hidden layers. This algorithm transfers the data from the input layer to the output layer, which is called feedforward. For training, the backpropagation technique is used. One hidden layer with (attributes + classes) / 2 units are used for this experiment. Each dataset has 22 attributes and 2 classes which are false and true. We determined the learning rate as 0.3 and momentum as 0.2 for each dataset.

• **Radial Basis Function Network (RBFN):** Radial Basis Function Network includes an input vector for classification, a layer of RBF neurons, and an output layer which has a node for each class. Dot products method is used between inputs and weights and for activation sigmoidal activation functions are used in MLP while in RBFN between inputs and weights Euclidean distances method is used and as activation function, Gaussian activation functions are used.

4) Support Vector Machines: Support vector machine (SVM) is a supervised machine learning method capable of both classification and regression. It is one of the most effective and simple methods used in classification. For classification, it is possible to separate two groups by drawing decision boundaries between two classes of data points in a hyperplane. The main objective of this algorithm is to find optimal hyperplane

PROPOSED WORK

There are a great variety of studies which have developed and applied statistical and machine learning based models for defect prediction in software systems. Basil etal. (1996) [1] have used logistic regression in order to examine what

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 52, Issue 7, July : 2023

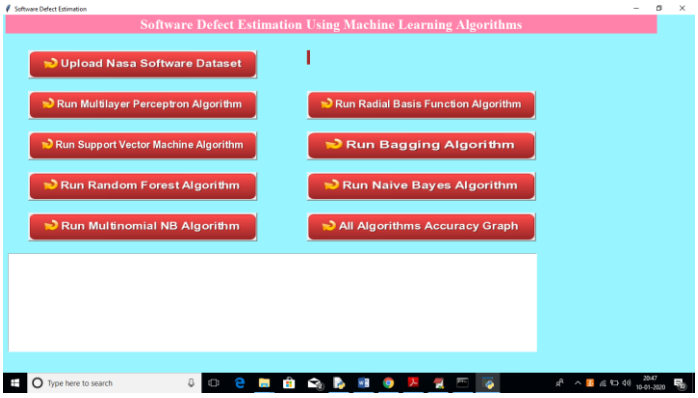
the effect of the suite of object-oriented design metrics is on the prediction of fault-prone classes. Khoshgoftaaretal. (1997)[7] have used the neural network in order to classify the modules of large telecommunication systems as fault-prone or not and compared it with a non-parametric discriminant model. The results of their study have shown that compared to the non-parametric discriminant model, the predictive accuracy of the neural network model had a better result. Then in 2002 [6], they made a case study by using regression trees to classify fault-prone modules of enormous telecommunication systems.

The proposed system includes SVM, Multilayer Perceptron, Run Bagging algorithm, Naive Bayes algorithm, Random Forest algorithm, Multinomial NB and Radial Basis Functions to solve the class misbalancing problem which causes in the decreasing performance of defect prediction. The dataset has been trained and spitted according to the constraints and using the accuracies has been defined in order to measure the defect estimation capability of various algorithms proposed.

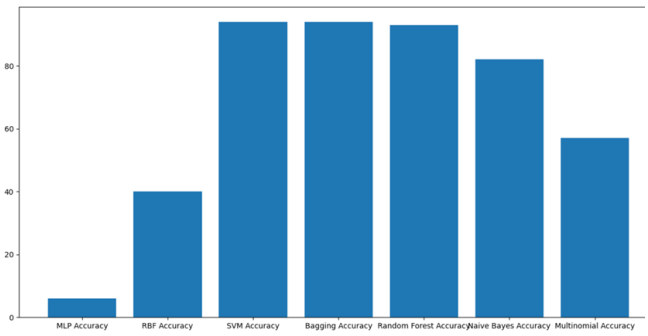
Advantages of proposed system:

1. Predicted model is used for evaluating the performance measures.
2. We can apply various datasets in this project. But we are using NASA datasets in our project.
3. Software defects are classified to the extent.
4. Advance measures can be taken on selection of algorithm
5. Provides Better results.
6. Identify defects in the early stage of the project which in turn results in Customer loyalty.

SAMPLE SCREENSHOTS



Industrial Engineering Journal
ISSN: 0970-2555
Volume : 52, Issue 7, July : 2023



CONCLUSION

In this paper, we propose a seven machine learning algorithms are used to predict defectiveness of software systems before they are released to the real environment and/or delivered to the customers and the best category which has the most capability to predict the software defects are tried to find while comparing them based on software quality metrics which are accuracy, precision, recall and F-measure. We carry out this experimental study with four NASA datasets which are PC1, CM1, KC1 and KC2. These datasets are obtained from public PROMISE repository. The results of this experimental study indicate that tree-structured classifiers in other words ensemble learners which are Random Forests and Bagging have better defect prediction performance compared to its counterparts. Especially, the capability of Bagging in predicting software defectiveness is better. When applied to all datasets, the overall accuracy, precision, recall and FMeasure of Bagging is within 83,7-94,1%, 81,3-93,1%, 83,7- 94,1% and 82,4-92,8% respectively.For PC1 dataset, Bagging outperforms all other machine learning techniques in all quality metric. However, Naive Bayes outperforms Bagging in precision and F-Measure while Bagging outperforms it in accuracy and recall for CM1 dataset. Random Forests outperforms all machine learning techniques in all quality metrics for KC1 dataset. Finally, for KC2 dataset, MLP outperforms all machine learning techniques in all quality metrics for KC2 dataset.

References

[1] Victor R Basili, Lionel C. Briand, and Walcelio L Melo. ´ A validation of object-oriented design metrics as quality indicators. IEEE Transactions on software engineering, 22(10):751–761, 1996.

[2] EvrenCeylan, F OnurKutlubay, and Ayse B Bener. Software defect identification using machine learning techniques. In 32nd EUROMICRO Conference on Software Engineering and Advanced Applications (EUROMICRO'06), pages 240–247. IEEE, 2006.

[3] Karim O Elish and Mahmoud O Elish. Predicting defect-prone software modules using support vector machines. Journal of Systems and Software, 81(5):649– 660, 2008.

[4] Norman Fenton, Paul Krause, and Martin Neil. Software measurement: Uncertainty and causal modeling. IEEE software, 19(4):116–122, 2002.

[5] Lan Guo, Yan Ma, Bojan Cukic, and Harshinder Singh. Robust prediction of fault-proneness by random forests. In 15th International Symposium on Software Reliability Engineering, pages 417–428. IEEE, 2004.

[6] Taghi M Khoshgoftaar, Edward B Allen, and Jianyu Deng. Using regression trees to classify fault-prone software modules. IEEE Transactions on reliability, 51(4):455–462, 2002.

[7] Taghi M Khoshgoftaar, Edward B Allen, John P Hudepohl, and Stephen J Aud. Application of neural networks to software quality modeling of a very large telecommunications system. IEEE Transactions on Neural Networks,

8(4):902–909, 1997. [8] Sunghun Kim, Hongyu Zhang, Rongxin Wu, and Liang Gong. Dealing with noise in defect prediction. In 2011 33rd International Conference on Software Engineering (ICSE), pages 481–490. IEEE, 2011.

[9] Yan Ma, LanGuo, and Bojan Cukic. A statistical framework for the prediction of fault-proneness. In Advances in Machine Learning Applications in Software Engineering, pages 237–263. IGI Global, 2007.

[10] Ruchika Malhotra. A systematic review of machine learning techniques for software fault prediction. Applied Soft Computing, 27:504–518, 2015.

Author's profile:

M.SUSMITHA SRIVALLI, as MCA student in the department of DCA at DVR & DR. HS MIC COLLEGE OF TECHNOLOGY, Kanchikacherla, NTR District. She has completed BSC in SRI DURGA MALLESWARA SIDDHARTHA MAHILA KALASALA From KRISHNA UNIVERSITY. His areas of interests are Networks, Machine Learning and Cyber Security.

Mrs. MOUNIKA.S completed her Bachelor of Technology in Computer Science and Engineering. She completed her Masters of Technology in

Computer Science and Engineering from JNTU KAKINADA UNIVERSITY. Currently working as an Assistant Professor in the department of AI&IT at DVR & DR HS MIC COLLEGE OF TECHNOLOGY (Autonomous), Kanchikacherla (NTR Dist, AP). Her areas of interest are Data Mining, Cloud Computing and Machine Learning & Networks.