THINK INDIA JOURNAL

ISSN:0971-1260
Vol-22-Issue-14-December-2019

Machine Learning Techniques On Historical Software Bugs For Prediction Of Software Bugs

D. Siri
Research Scholar
Shri JJT University
Rajasthan
Assistant professor
Dept of IT, MRECW
dharmapuri.siri@gmail.com

Dr. Prasadu Peddi
Assistant Professor
Shri JJT University
Rajasthan

Dr. D. Murali
Prof & Hod
Vemu Institute Of Technology, P Kothakota, Chittoor Ap

ABSTRACT:

With the growing complexities of the software, the number of potential bugs is also increasing rapidly. These bugs hinder the rapid software development cycle. Bugs, if left unresolved, might cause problems in the long run. Also, without any prior knowledge about the location and the number of bugs, managers may not be able to allocate resources in an efficient way. Software Bug Prediction (SBP) is an important issue in software development and maintenance processes, which concerns with the overall of software successes. This is because predicting the software faults in earlier phase improves the software quality, reliability, efficiency and reduces the software cost. This paper presents a software bug prediction model based on machine learning (ML) algorithms. Three supervised ML algorithms have been used to predict future software faults based on historical data. These classifiers are Naïve Bayes (NB), Decision Tree (DT) and Artificial Neural Networks (ANNs). The evaluation process showed that ML algorithms can be used effectively with high accuracy rate. Furthermore, a comparison measure is applied to compare the proposed prediction model with other approaches. The collected results showed that the ML approach has a better performance.

Keywords: Software bug prediction, faults prediction, prediction model, machine learning;

INTRODUCTION:

When there repeatedly exists a software failure in system through time it automatically leads to software defect. Software defect are an error that are introduced by software developer and stakeholders. The main objective of software defect prediction is to improve the quality, minimized cost and time of software products. Software defect is also referred to as bug can

ISSN:0971-1260
Vol-22-Issue-14-December-2019

be defined as shortage in the software product that causes the software not to perform its task as the programmer and customer needed. Machine Learning is one of the most vital and motivating area of research with the objective of finding meaningful information from huge data sets. The basic purpose of machine learning is to extract useful pattern from the data, mining data may be structured format (example. multiple data base) or text mining: unstructured data (example, natural language document).

LITERATURE REVIEW:

Malhotra, R (2014) presented a good systematic review for software bug prediction techniques, which using Machine Learning (ML). The paper included a review of all the studies between the period of 1991 and 2013, analyzed the ML techniques for software bug prediction models, and assessed their performance, compared between ML and statistic techniques, compared between different ML techniques and summarized the strength and the weakness of the ML techniques.

Gupta and K. Saxena, D.L et al () developed a model for object-oriented Software Bug Prediction System (SBPS). The study combined similar types of defect datasets which are available at Promise Software Engineering Repository. The study evaluated the proposed model by using the performance measure (accuracy). Finally, the study results showed that the average proposed model accuracy is 76.27%.

Singh and Chug et al () discussed five popular ML algorithms used for software defect prediction i.e. Artificial Neural Networks (ANNs), Particle Swarm Optimization (PSO), Decision Tree (DT), Naïve Bayes (NB) and Linear Classifiers (LC). The study presented important results including that the ANN has lowest error rate followed by DT, but the linear classifier is better than other algorithms in term of defect prediction accuracy, the most popular methods used in software defect prediction are: DT, BL, ANN, SVM, RBL and EA, and the common metrics used in software defect prediction studies are: Line Of Code (LOC) metrics, object oriented metrics such as cohesion, coupling and inheritance, also other metrics called hybrid metrics which used both object oriented and procedural metrics, furthermore the results showed that most software defect prediction studied used NASA dataset and PROMISE dataset.

Grishma BR et al () investigated root cause for fault prediction by applying clustering techniques and identifies the defects occurs in various phases of SDLC. In this research they used COQUALMO prediction model to predict the fault in a software and applied various clustering algorithms like k-means, agglomerative clustering, COBWEB, density based scan,

ISSN:0971-1260
Vol-22-Issue-14-December-2019

expectation maximization and farthest first. Implementation was done using Weka tool. Finally they conclude that k-means algorithm works better when compared with other algorithms

Software Defect: A software defect is an error, bug, flaw, fault, malfunction or mistakes in software that causes it to create an erroneous or unpredicted outcome. Faults are essential properties of a system. They appear from design or manufacture, or external environment. Software flaws are programming errors which cause different performance compared with anticipation. The majorities of the faults are from source code or deign, some of them are from the incorrect code generating from compilers. For software developers and clients, software faults are a danger problem. Software defects not merely decrease software quality, increase costing but also delay the development schedule. Software fault predicting is proposed to solve this sort of trouble. SDP can efficiently progress the effectiveness of software testing and direct the allocation of resources. To develop quality software, software flaws have to be detected and corrected at early phase of SDLC.

MACHINE LEARNING CONCEPT:

One of the main differences between how people and computer work is that human, performing any kind of activity usually simultaneously expand efforts to improve the way they perform it. This is to say, human can perform any tasks with the versatility means while not machines. Machine learning algorithms 'learn' to predict outputs based on previous examples of relationships between input data and outputs. The models that are constructed based on the relationship between input and output slowly improved by testing and its predictions and correcting when wrong[9]; according to Tom Mitchell definition, the machine learns with respect to particular tasks T, performance P and Experience E.

Machine Learning Tasks A system that is used to understand the concept and its environment using a simplified interpretation of the environment using model is called cognitive system. The step that we pass to construct the model is known as inductive learning. The Cognitive system is able to combine its experience by constructing new structures is patterns. The constructed model and pattern by cognitive system is called machine learning. Models that are described as predictive since it can be used to predict the output of a function (target function) for a given value in the function's domain while informative pattern are characterized only describes the portion of data. Machine learning task classified into four that are supervised, unsupervised, semi-supervised and reinforcement learning; however, the

ISSN:0971-1260
Vol-22-Issue-14-December-2019

most known task is supervised and un-supervised learning. In section a & section b below, we looked some of supervised and un-supervised learning below.

i. Supervised Learning It is the machine learning task of inferring a function from labeled training data. The training data consists of a set of training examples. In supervised learning, each example is a pair consisting of an input object and a desired output value. It has two known supervised learning tasks, classification and regression. Classification concerns on building predictive model for function with discrete range, while regression concern on continuous range model building. A lot of researchers in machine learning are focused on supervised learning. The most common supervised machine learning methods includes concept learning, classification, rule learning, instance based learning, Bayesian learning, liner regression, neural network and support vector machine.

ii. ii. Unsupervised Learning This is also called learning from observation. In unsupervised learning the system has to explore any patterns based only on the common properties of the example without knowing how many or even if there are any patterns. The most common method in unsupervised learning's is association rule mining, sequential pattern mining and clustering.

Machine Learning in Software Engineering: The Machine learning is not a difficult science. Machine learns automatically from training data and it generate summaries of data or existing systems in a smaller form while the software engineer could able to use machines in order to minimize the system development phases time and cost consumptions. One thing that overcomes machine learning integrating with software engineering is developing machine learning based solution to software engineering problems. The data and the complexity of pattern in software engineering needs pre-processed before applying machine learning method, this is similar with other applications. Component-based approach fault prevention methods have tremendous attentions from developers now a time, which reduce the development cost and time as well as to improve the quality and reliability of the projects by breaking the project in to separate components, however it is only practical for finding the problems in the quality of individual components. One of the solution to resolve the problem is through machine learning approach to software engineering.

ISSN:0971-1260
Vol-22-Issue-14-December-2019

Table: SE (software engineering) tasks and applicable ML methods

SE Tasks	Applicable types of Learning methods
Requirement Engineering	AL, BBN, LL, DT
Rapid Prototyping	GP
Component Reuse	IBL
Cost/effort prediction	IBL (CBR), DT
Defect Prediction	BBN
Test Oracle generation	AL
Test data Adequacy	CL
Validation	AL

In order to estimate the quality of software we used an attribute such as software development effort, software reliability and productivity of programmer to predict the important of model in software engineering. Early software quality prediction to enhance the performance of system via machine learning techniques (case based reasoning and fuzzy logic) was studied. According to researches, Software engineering problems that are ready to resolved by machine learning, which are software measurement selection, defect prediction models, software reuse qualification, software requirements gathering, software quality estimation, project management, software testing, we also called it an application of machine learning in software engineering. Among software engineering problems that we listed, researcher chooses software defect prediction via machine learning methods to conduct this study.

Machine Learning: Machine learning provides ability predict the occurrence of an event based on historical data without being explicitly programmed. Machine learning focuses on the output variable and try to look for patterns within the data and predicts the final outcome. There are two kinds of machine learning algorithms. Supervised, Unsupervised machine learning algorithms. In supervised algorithms, we will have an output variable decided upfront and we try to look for pattern similarities between the dependent and independent variables. In un supervised machine learning algorithms, one can pass the data and predict the final outcome variable.

Machine Learning in Software Defect Prediction:

The field of machine learning has been growing rapidly, producing a variety of learning algorithms for different applications. As we discussed in section 2.5.4.1 one of the

ISSN:0971-1260
Vol-22-Issue-14-December-2019

application of machine learning is software engineering. The ultimate value of those algorithms is to a great extent judged by their success in solving real-world problems. Therefore, algorithm reproduction and application to new tasks are crucial to the progress of the field. However, various machine learning researchers currently publish for software fault prediction model development. Now, we categorizing successful software defect model into three that are based on classification, clustering and ensemble methods**.**

Defect Prediction Using Machine Learning Algorithms: Predicting the bug occurrence and understanding the process flow of SDLC are the key factors which leads to success criteria. Defect prediction using Machine Learning algorithm can be applied for any stage of SDLC such as , identification of current problems, plan, design, build, test, deploy and maintain and also any model type of SDLC such as Waterfall Model, Agile Model, Iterative Model, V-Shaped Model, Big Bang Model and Spiral Model. Machine learning algorithms and statistical analysis actually helps to guess that a code is buggy. Improved and better approaches in the development will increase the quality of the software**.**

METHODOLOGY:

Machine Learning Algorithms used: This subsection deals with the concept of the algorithms used in our approach.

Regression is one of the machine learning methods to derive the relationship between variables of interest, i.e. regression enables the determination of best fit curve to the data in hand. Many regression methods are known and depending on the nature and the number of both independent and dependent variables, a suitable method can be chosen. The regression works to minimize the sum of the squares of the errors (least squares). This amounts to saying that the regression techniques provide the best fit curve globally, though there may be some discrepancies locally. But, the performance of regression models on the test data may not chime with that of training data. It has been shown that the linear model is the close approximation for the bug predictions 18. Moreover, the linear model is the simplest of all the models known. So, we resort to a linear model on the carefully chosen metrics and their weights determined using the proposed algorithm. But instead of using the regression coefficients directly in the model, we use the differences in the coefficients of determination (R square) as the weights.

Selection of metrics: The input to any regression model is a good set of metrics (independent variables) which can determine the dependent variable to the highest accuracy achievable. We choose metrics after a careful evaluation of their coefficients of determination (R square)

ISSN:0971-1260
Vol-22-Issue-14-December-2019

in simple linear regressions. Taking the actual bug count as the dependent variable and the metric in hand as the independent variable, we perform simple linear regression to determine the coefficient of determination. Once the coefficients of determination of every metric are obtained through linear regressions, we sort the metrics in the decreasing order of their coefficients of determination

Perform multiple regression: Linear multiple regression is a good machine learning method to obtain the best fit curve for the training data set. Taking the 'n' chosen metrics as the independent variables and the actual bug count as the dependent variable, we perform a linear multiple regression. The coefficient of determination obtained will be used in the further steps to determine the marginal R square values, which in turn will be used to determine the weights of the metrics in the final model

Compute marginal R square values: In order to find the weights of the 'n' chosen metrics, we perform 'n' linear multiple regressions taking all the 'n' chosen metrics except one (a total of (n-1)) as the independent variables and the actual bug count as the dependent variable. The differences in the coefficients of determination (marginal R square) obtained in the current regressions and that obtained in the previous step are determined and these values when normalized in the range of 0 to 1 give the weights of the metrics left out in the current regressions. Higher marginal R square value of a metric indicates that the error in the regression is more because this metric was not considered. This, in turn, means that the contribution of the metric in the bug prediction is high. So, we use these normalized marginal squares as the weights of the metrics.

Logistic Regression Logistic regression method solves classification problems. It is meant for predicting the likelihood of an entity belonging one class or another class. There are many such examples like marketing effort to know about whether customers purchase or non-purchase, creditworthiness for paying EMI is high or low, in insurance whether there is high or low risk of accident claim. The logistic regression follows a s-shaped curve to the predict the feature of the a binary response variables. In this approach complex optimized equation is obtained by converting from the logistic equation to the OLS-type equation. The Eq. (1) which is derived from probabilistic approach is given below. P is the probability for Y=1 and 1-P is the probability of getting Y=0;

$$\ln\left(\frac{P}{1-P}\right)=a+bX \qquad (1)$$

ISSN:0971-1260
Vol-22-Issue-14-December-2019

P can also be obtained from the regression equation. The regression equation given in Eq. (2) calculates the expected probability for a given value of X for Y=1.

$$P = \frac{\exp(a+bX)}{1+\exp(a+bx)} = \frac{e^{a+bx}}{1+e^{a+bx}} \qquad (2)$$

Decision tree & Random Forest: Decision tree is a one of supervised learning algorithms that is widely used in classification problems with a fixed target variable. This algorithm can be applied to both categorical and continuous input and output variables. In this technique, the given data set is divided into two or more consistent sets based on most significant input variables which act as differentiator. Random Forest is a ensemble classifier that is also meant for (2) (1) classification, and regression. It constructs the number of decision trees on different sub-samples of the dataset and takes average to get the predictive accuracy and also controls over fitting of the model. For classification its output is based on mode of the classes and for regression it uses mean of the individual trees.

Artificial Neural Networks (ANNs): ANNs are networks inspired by biological neural networks. Neural networks are non-linear classifier which can model complex relationships between the inputs and the outputs. A neural network consists of a collection of processing units called neurons that are work together in parallel to produce output. Each connection between neurons can transmit a signal to other neurons and each neuron calculates its output using the nonlinear function of the sum of all neuron's inputs.

Naïve Bayes (NB): NB is an efficient and simple probabilistic classifier based on Bayes theorem with independence assumption between the features. NB is not single algorithms, but a family of algorithms based on common principle, which assumes that the presence or absence of a particular feature of the class is not related to the presence and absence of any other features

Results and Discussion:

The section provides understanding about the performance study of the proposed predictive algorithm. The performance study of the proposed technique is performed over different performance factors. The obtained performances according to the evaluated parameters are compared with a traditional classifier Bayesian using WEKA machine learning tool and their performance is reported.

ISSN:0971-1260
Vol-22-Issue-14-December-2019

Accuracy: The accuracy is a measurement of the data model for finding the amount of correctly classified data using the input samples. The performance of the algorithm in terms of accuracy can be evaluated using the following formula.

$$accuracy\ \% = \frac{total\ correctly\ classified\ data}{total\ input\ datasets} X100$$

Accuracy rate of both the experiments of each data sets

Features	CM1	JM1	MC1	MC2	PC3	PC4
All	83.68	84.34	85.32	86.78	87.84	90.34
4	87.5	90.9	87.6	89.2	74.8	81.9
5	85.48	86.83	88.45	89.09	79.76	83.23
6	84.12	86.45	79.45	86.78	84.11	92.89
7	86.54	87.34	88.99	93.24	95.78	96.32
8	87.32	88.98	90.76	78.87	90.98	91.92
9	79,78	80.45	87.34	88.89	92.89	97.89

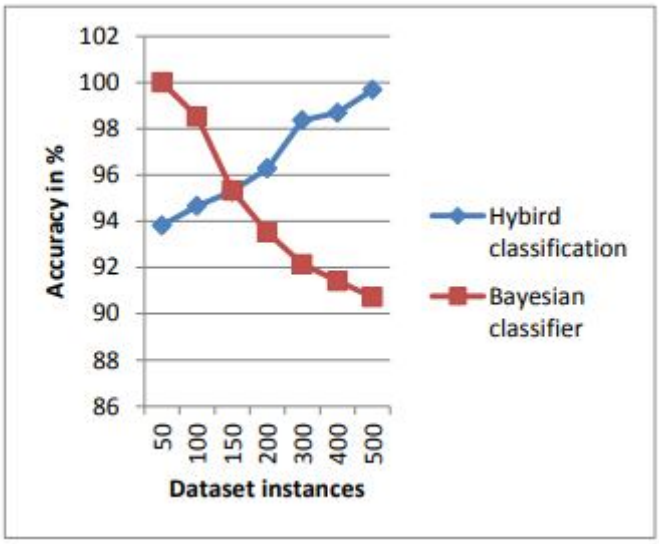

Figure: Accuracy

The performance of the proposed hybrid classifier and the traditional Bayesian classifier is compared using the figure 2. In this diagram the X axis shows the training samples in the dataset and the Y axis shows the obtained accuracy in terms of percentage

ISSN:0971-1260
Vol-22-Issue-14-December-2019

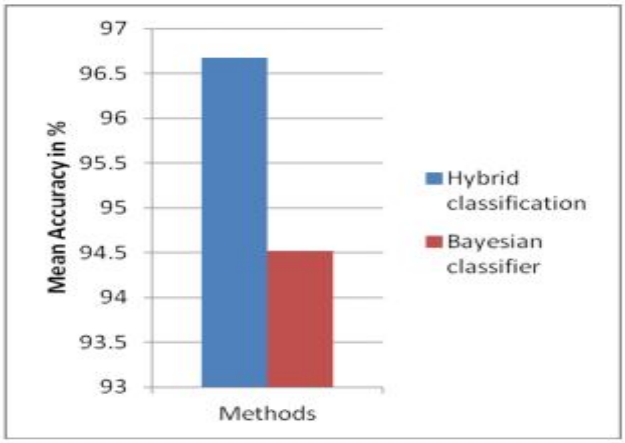

Figure: Mean accuracy

This table includes the proportion of positive modules which are also predicted as positive. By analyzing the results, it is found that the feature selection model has more capability for defect prediction than all features, with five of six data sets

The key aim of the proposed investigation and solution development is to accomplish an accurate and efficient prediction algorithm. That algorithm is used with software engineering to minimize the efforts of programmer and tester, additionally improve or maximize the production of good quality software solutions. This chapter provides the summary of the entire effort made for proposed solution development and additionally the future extension of the work is also suggested

Conclusion:

Now a day the development of software based system are increasing from the previous years due to its benefit. However, the quality of the system is required before it is delivered to end users. All of the machine learning classifiers have shown almost similar results. Using the bug Level criteria, we are able to classify the files with more than 90% accuracy. Based on the results obtained from the systematic review, we conclude that the machine learning techniques have the ability for predicting software fault proneness and can be used by software practitioners and researchers. However, the application of the machine learning techniques in software fault prediction is still limited and more number of studies should be carried out in order to obtain well formed and generalizable results. We provide future guidelines to practitioners and researchers based on the results obtained in this work. In order to enhance the software quality, we have various quality metrics such as software testing, CMM and ISO standards. Currently software testing becomes more and more important in the software reliability. Software defects prediction can effectively improve the efficiency of

ISSN:0971-1260
Vol-22-Issue-14-December-2019

software testing and guide the allocation of resources. For the error-prone modules, we should spend more resource and time. However, some of the algorithms were able to generate 100 % accuracy with train and test datasets. Hence, It is decided to settled down with Machine learning algorithms. If we increase the dataset size ANN will be a good option to proceed further.

REFERENCES:

1. Malhotra, R (2015), "A systematic review of machine learning techniques for software fault prediction", Applied Soft Computing, vol. 27, PP: 504-518.
2. Kaur, A., I. Kaur (2014), "Empirical Evaluation of Machine Learning Algorithms for Fault Prediction", Lecture Notes on Software Engineering, vol. 2, no. 2.
3. Sandeep D and R. S, "Case Studies of Most Common and Severe Types of Software System Failure " International Journal of Advanced Research in Computer Science and Software Engineering vol. 2, pp. 341-347 August 2012
4. Venkata U and R. A, "Empirical Assessment of Machine Learning based Software Defect Prediction Techniques " Proceedings of the 10th IEEE International Workshop on Object-Oriented Real-Time Dependable Systems 2005.
5. R. Malhotra, "Comparative analysis of statistical and machine learning methods for predicting faulty modules," Applied Soft Computing 21, (2014): 286-297
6. T. Gyimothy, R. Ferenc and I. Siket, "Empirical Validation of Object-Oriented Metrics on Open Source Software for Fault Prediction," IEEE Transactions On Software Engineering, 2005.
7. M. C. Prasad, L. Florence and A. Arya, "A Study on Software Metrics based Software Defect Prediction using Data Mining and Machine Learning Techniques," International Journal of Database Theory and Application, pp. 179-190, 2015.
8. Chug, Anuradha, and Shafali Dhall. "Software defect prediction using supervised learning algorithm and unsupervised learning algorithm." (2013): 5-01.
9. Venkata U and R. A, "Empirical Assessment of Machine Learning based Software Defect Prediction Techniques " Proceedings of the 10th IEEE International Workshop on Object-Oriented Real-Time Dependable Systems 2005
10. Rajkumar G and K.Alagarsamy, "The Most Common Factors For The Failure Of Software Development Project," vol. 11, pp. 74-77, January 2013

ISSN:0971-1260
Vol-22-Issue-14-December-2019

11. George T, Ioannis K, Ioannis P, and Ioannis V, "Modern Applications of Machine Learning " Proceedings of the 1st Annual SEERC Doctoral Student Conference vol. 1, 2006
12. Yogesh S, Pradeep K, and O. S, "A Review of Studies On Machine Learning Techniques," International Journal of Computer Science and Security vol. 1, pp. 70-84
13. Nguyen V, "The Application of Machine Learning Methods in Software Verification and Validation " MSc, Master of Science in Engineering, The University of Texas at Austin, Texas 2010.
14. Saiqa A, Luiz F, and F. A, "Benchmarking Machine Learning Techniques for Software Defect Detection," International Journal of Software Engineering & Applications, vol. 6, pp. 11-23, May 2015

