International Journal of Marketing and Technology
Vol. 14 Issue 09, Sep 2024
ISSN: 2249-1058 Impact Factor: 6.559
Journal Homepage: http://www.ijmra.us, Email: editorijmie@gmail.com
Double-Blind Peer Reviewed Refereed Open Access International Journal - Included in the International Serial Directories Indexed & Listed at: Ulrich's Periodicals Directory ©, U.S.A., Open J-Gate as well as in Cabell's Directories of Publishing Opportunities, U.S.A

Accuracy of Software Defect Prediction using Machine Learning on Test and Source Metrics

Prathyusha Nama, Test Architecture Manager, Align Technology Inc., USA

Abstract: -Focus of the paper is on predicting software defects (SD) based on Machine learning (ML)techniques which is a challenging research area because of unbalanced nature of data sets.In general, a set of design metrics is used for fault prediction and for identifying thefault-proneness in software and recent studies shows that ML techniques isbeing applied for defect prediction. However, some ML techniques cannot produce theneeded results when dealing with unbalanced dataset and the results produced are not certainlyinferred by the developers by observing these factors, we propose a unique approach for faultpredictionwhich is based on feature selection technique which improves the overallperformance by using attribute selection when predicting defects.The ML concentrates on the algorithms entirely centred on statistical methods and data mining techniques for classifying and predicting the defects and these statistical methods followed are quite similar to regression based methods which we used earlier to the ML. When more data is available, ML algorithms behave as dynamic algorithms to improve their performancesignificantly.The GB ML technique is providing good accuracy compared to other DT and SVM technique. In this model is simulated python language and calculated simulation parameter i.e. precision, recall and accuracy.

Keywords: -Software Defects, Accuracy, Precision, Recall

I. INTRODUCTION

With growing demand and technology, the software industry is rapidly evolving. Since humans do most of the software development, defects will inevitably occur. In general, defects can be defined as undesired or unacceptable deviations in software documents, programs, and data [1].Defects may exist in requirements analysis because the product manager misinterprets the customer's needs, and as a result, this defect will also carry on to the system design phase. Defects may also occur in the code due to inexperienced coders. Defects significantly impact software quality, such as increased software maintenance costs, especially in healthcare, and aerospace software defects can have serious consequences. If the fault is detected after deployment, it causes an overhead on the development team as they need to re-design some software modules, which increases the development costs. Defects are nightmares for reputed organizations. Their reputation is affected due to customer dissatisfaction and hence reduces its market share. Therefore, software testing has become one of the main focuses of industrial research [2]. With the rise in software development and software complexity, the number of defects has increased to the extent that traditional manual methods consume much time and become inefficient. The rise of machine learning has made automatic classification of defects a research hotspot.

The life cycle and the phases of the software which is the basic framework used by the managers, developers and the project management team is so called as the software development life cycle (SDLC). It is equipped with needed tasks to be performed in each and every phases during the development of the software.

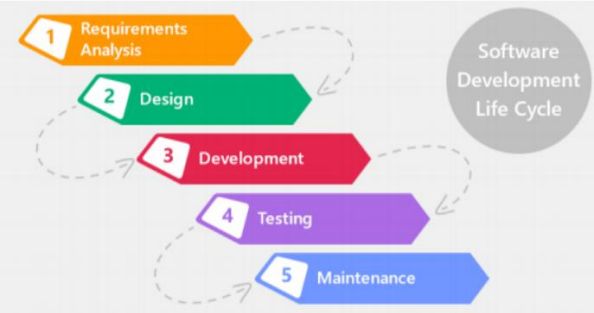

Fig. 1:Life cycle of software development

Figure 1 describes various phases of software development process and its activities. The SDLC followed by the development team illustrates a practice and the procedures for developing software of improved quality as well as the methods used in the development process. The phase of SDLC needs to be followed strictly by the team members to satisfy the customer and to meet their highly expected demands. The model is well-known to perform verification and validation. Various phases and the process and activities involved during SDLC is shown in figure. 1. Requirement analysis is the first phase and that needs to be first step which is to be followed in developing

International Journal of Marketing and Technology
Vol. 14 Issue 09, Sep 2024
ISSN: 2249-1058 Impact Factor: 6.559
Journal Homepage: http://www.ijmra.us, Email: editorijmie@gmail.com
Double-Blind Peer Reviewed Refereed Open Access International Journal - Included in the International Serial Directories Indexed & Listed at: Ulrich's Periodicals Directory ©, U.S.A., Open J-Gate as well as in Cabell's Directories of Publishing Opportunities, U.S.A

the software also called as requirement engineering. In this phase, the various requirement of the software needs to be gathered from the client and we need to explain the boundaries of the system to the client with a clear feature description and the do's and don'ts of the system and the target system functionalities which help us to develop a sophisticated document which give rise to a system requirement specification [5].
The developers have to check whether the customer demands can be satisfied by performing an analysis of the proposed system and its needed functionalities to ensure that the application can be practically implemented by conducting the feasibility study of the system and in the next phase the needed requirements are gathered from the user [6, 7]. During this phase, the developers need to sit with the clients discussing the features that the system provide and what the system is intended for and its potential benefits. This phase is also accompanied with documents such as user interface requirements, technical and functional requirements, design requirements plan along with non-functional requirements [8].

II. SOFTWARE QUALITY AND SD PREDICTION

2.1 Concept of Software

Since analysts often can't distinguish between software defects and programming faults, errors, and failure, this article utilizes IEEE 729-1983 (Standard Glossary of Software Engineering Terminology) to characterize defects as, From the inside of the product, the defects are mistakes and errors in the maintenance or development of the product item. From an external perspective, a defect is the violation or failure of the framework/system to accomplish specific capacities [9, 10]. The description of the concepts that are easily mistaken with defects is as follows

1. Fault: The software doesn't perform according to the client's expectations and runs in an unsuitable internal state. We can view it as a defect that can prompt software errors and is regarded as dynamic behavior.

2. Failure: It refers to the outputs that the software generates at runtime, which the client doesn't accept. For instance, if theexecution capacity is lost, and the client's capabilities are not met, the framework can't meet the fixed asset's execution necessities.

3. Error: It is introduced by individuals and changed over into faults under specific conditions. It exists in the whole software life cycle, including error information in the software design, data structure, code, requirements analysis, and other carriers [11]. The quality of software relies upon the number of defects. An excessive number of defects lead to reduced client satisfaction, consuming organization assets and expenses, and slower testing. To spare the costs, improving test productivity is critical to managing defects.

2.2 Software Defect Prediction

The error in the algorithm or in the software program will not permit the software product to satisfy the user requirements and because of that, the user expectations and the software requirement standards are not maintained by the product and is also called as software defect. The error in the software sometimes produce unexpected outcome and cause software malfunctioning too. There are several ways to define the defects produced by the software: -

- The defect or the bug in the application may be caused or created due to some mistake of the programmer.
- If there is a deviation in the expected result produced by the software and it is not producing the result specified or defined in Specification document, then it is considered as a defect.
- Failing to satisfy the end user expectations is considered as defect in the software and it is mainly due to the bugs arising in the program or the methods used when the product is developed.

The foremost thing when developing a piece of software is to produce it with great eminence and with high excellence. Superiority of the software is measured by the degree to which the piece of code meets the requirements specified in the requirement specification document. [12].

III. PROPOSED METHODOLOGY

The distribution in machine learning builds the module based on the training dataset with a classification algorithm. This learning can be categorized into all three possible classification algorithms. In a supervised learning class, labeled data is present at the beginning.
In semi-supervised learning, some of the class labels are known. Whereas in unsupervised learning no class label for the entire dataset.
Once the training phase is finished, features are extracted from the data based on term frequency, and then the classification technique is applied.

International Journal of Marketing and Technology
Vol. 14 Issue 09, Sep 2024
ISSN: 2249-1058 Impact Factor: 6.559
Journal Homepage: http://www.ijmra.us, Email: editorijmie@gmail.com
Double-Blind Peer Reviewed Refereed Open Access International Journal - Included in the International Serial Directories Indexed & Listed at: Ulrich's Periodicals Directory ©, U.S.A., Open J-Gate as well as in Cabell's Directories of Publishing Opportunities, U.S.A

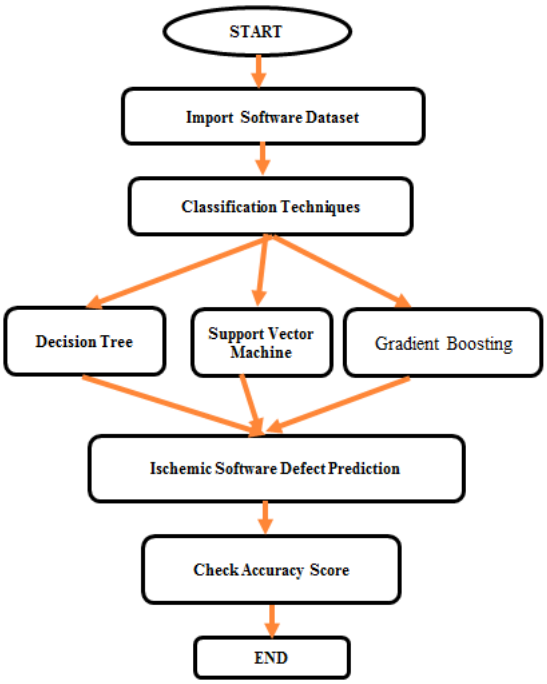

Fig. 2: Flow chart of Proposed Methodology

DT:-

A DT is a choice help instrument that utilizes a tree-like model of choices and their potential results, including chance occasion results, asset expenses, and utility. It is one method for showing a calculation that just holds back restrictive control explanations. DT are ordinarily utilized in tasks research, explicitly in choice examination, to assist with recognizing a technique probably going to arrive at an objective, but at the same time are a well-known device in ML.

GB:-

GB calculation is one of the most remarkable calculations in the field of AI. As we realize that the blunders in AI calculations are extensively characterized into two classifications for example Inclination Error and Variance Error. As inclination supporting is one of the helping calculations limiting predisposition mistake of the model is utilized.

SVM:-

In ML, SVM are directed learning models with related learning calculations that examine information for grouping and relapse examination.

To isolate the two classes of data of interest, there are numerous conceivable hyperplanes that could be picked. Our goal is to find a plane that has the greatest edge, for example the greatest distance between data of interest of the two classes. Boosting the edge distance gives some support so future information focuses can be grouped with more certainty.

IV. SIMULATION RESULTS

In this test case, we considered other standard classification scheme suchas Support vector machine (SVM), decision tree (DT) and Gradient Boosting(GB) classifier.

Data set:KC1 dataset contains total 2109 modules which are given in the table 1 as the predicted outcome using SVMclassifier in terms of defective (D) and non-defective (ND) classes.

Table 1: KC1 Prediction for SVM

Actual class	Predicted class	
	ND	D
D	205	129
ND	203	1572

It is really tough to find a better separation between two groups. Certainly, the results obtained from the tests may overlay with one another. Sometimes, the defective module will be correctly classified as True Positive instances (TP)) and also in certain cases the defect will be classified negative instances(FN). Occasionally, without defect willbe correctly classified as True negatives (TN) but sometimes, without defect will be classified as Positive (FP).

The above analysis shows that, when the threshold is low at a point both true positive fraction (TP) and sensitivity will rise and at times FP will also increase and therefore a decrease will be there in TN and in specificity.

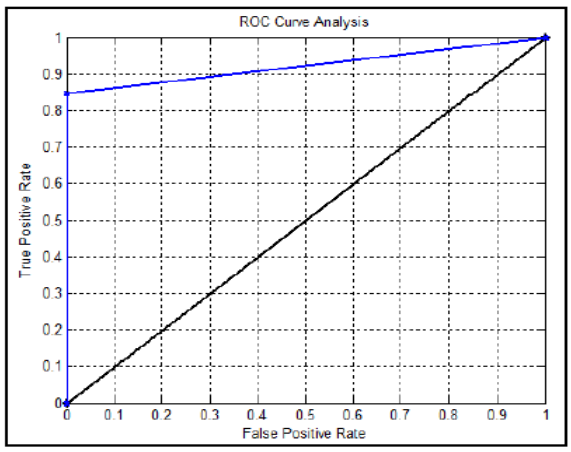

Fig. 3: ROCAnalysis of KC1 using SVM

International Journal of Marketing and Technology
Vol. 14 Issue 09, Sep 2024
ISSN: 2249-1058 Impact Factor: 6.559
Journal Homepage: http://www.ijmra.us, Email: editorijmie@gmail.com
Double-Blind Peer Reviewed Refereed Open Access International Journal - Included in the International Serial Directories Indexed & Listed at: Ulrich's Periodicals Directory ©, U.S.A., Open J-Gate as well as in Cabell's Directories of Publishing Opportunities, U.S.A

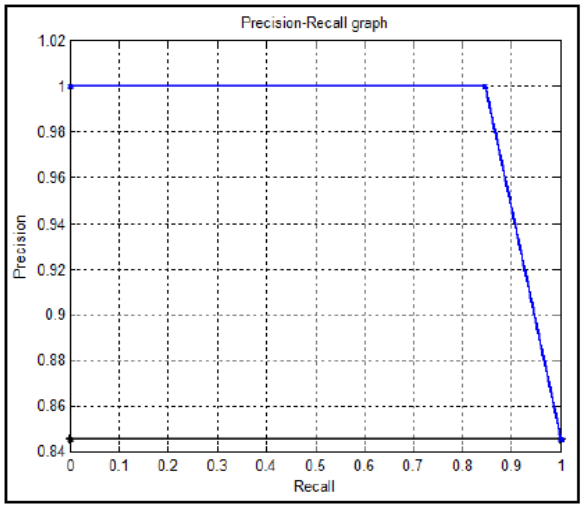

Fig. 4: Precision and RecallAnalysis of KC1 using SVM

Table II: KC1 Prediction for DT

Actual class	Predicted class	
	ND	D
D	178	201
ND	68	1662

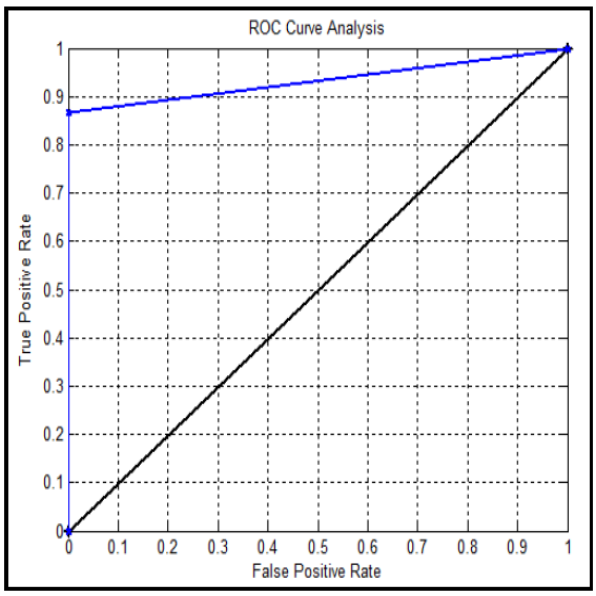

Fig. 5: ROCAnalysis of KC1 using DT

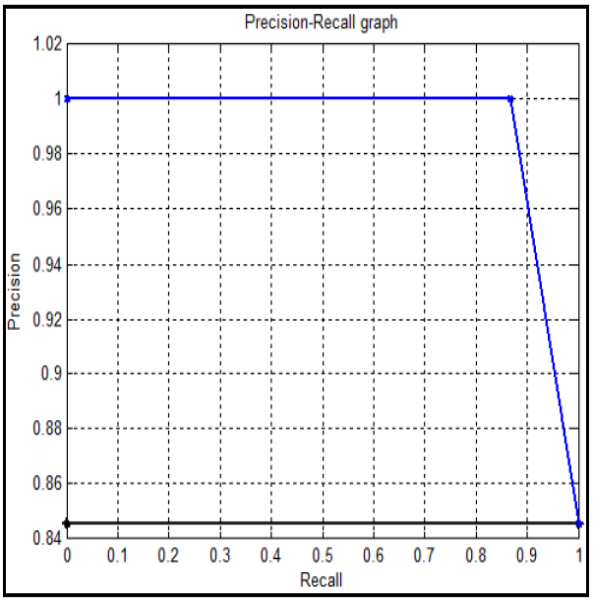

Fig. 6: Precision and RecallAnalysis of KC1 using SVM

Table III: Table 1: KC1 Prediction for GB

Actual class	Predicted class	
	ND	D
D	326	0
ND	228	1555

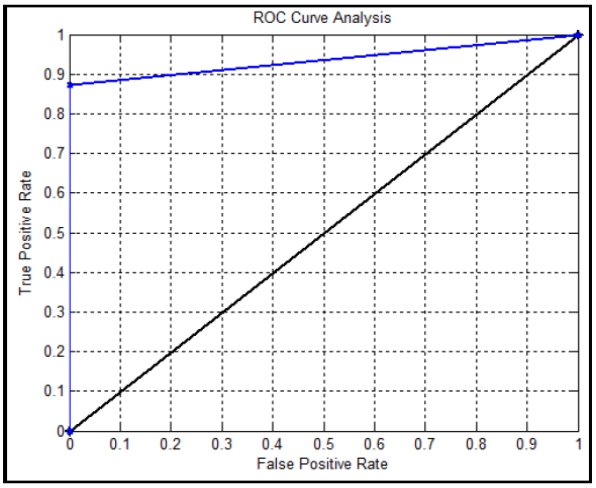

Fig. 7: ROCAnalysis of KC1 using GB

International Journal of Marketing and Technology
Vol. 14 Issue 09, Sep 2024
ISSN: 2249-1058 Impact Factor: 6.559
Journal Homepage: http://www.ijmra.us, Email: editorijmie@gmail.com
Double-Blind Peer Reviewed Refereed Open Access International Journal - Included in the International Serial Directories Indexed & Listed at: Ulrich's Periodicals Directory ©, U.S.A., Open J-Gate as well as in Cabell's Directories of Publishing Opportunities, U.S.A

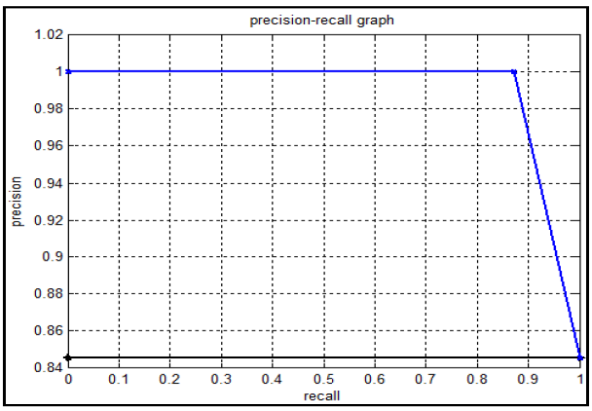

Fig. 8: Precision and RecallAnalysis of KC1 using GB

Table IV: Comparison Result

Technique	Precision	Recall	Accuracy
SVM	68.33%	77.45%	84.25%
DT	72.35%	81.67%	87.24%
KNN	78.34%	87.21%	89.18%

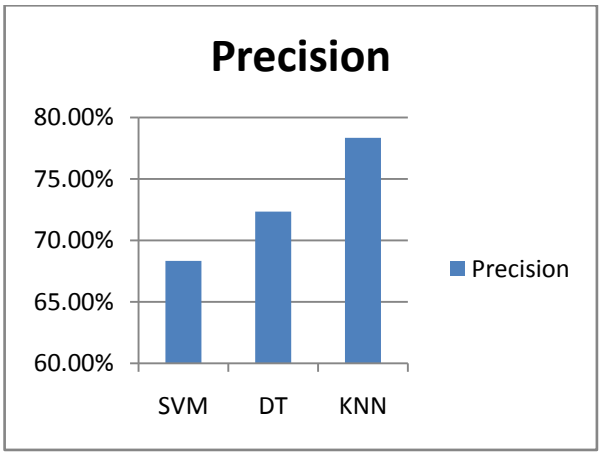

Fig. 9: Graphical Represent of Precision

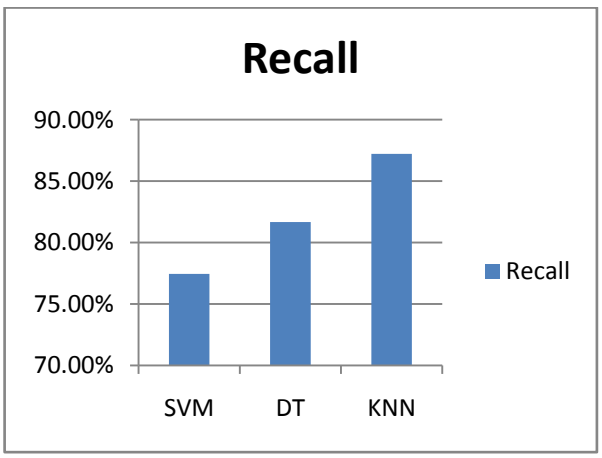

Fig. 10: Graphical Represent of Recall

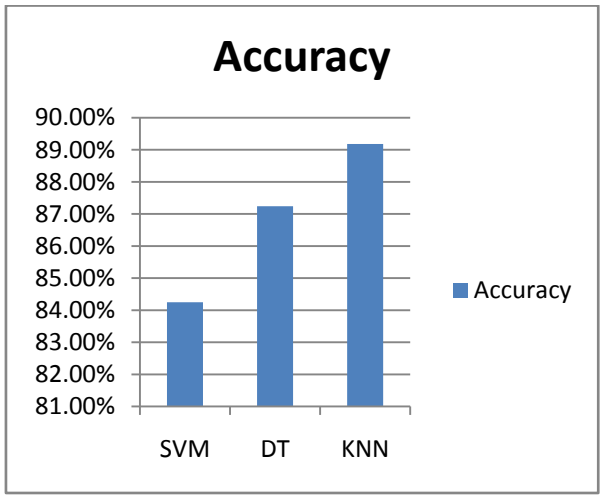

Fig. 11: Graphical Represent of Accuracy

V. CONCLUSION

Our main intention is to use Machine learning based methods to cultivate an automated process framework or a prototype for predicting the defects in the software. The model mends the superiority of the software and identifies the fault prone modules by taking metric data into consideration.Usage of ML techniques is model construction and to identify and to forecast the defects in the software. To understand how metrics can be used by the developers to control, track and understand the software process and its ongoing activities.

Then software defect prediction automatically replaces cumbersome traditional methods that sometimes lead to errors, and a system is implemented using DT, SVM and GB ML technique. The SVM achieves 84.25% accuracy, DT achieves 87.24% accuracy and GB achieves 89.18% accuracy. It is clearly that the GB ML technique is provides good accuracy compared to SVM and GB ML technique.

REFERENCES

[1] IqraMehmood, Sidra Shahid, HameedHussain, Inayat Khan,Shafiq Ahmad, ShahidRahman, NajeebUllah and Shamsul Huda, "A Novel Approach to Improve Software Defect Prediction Accuracy Using Machine Learning", IEEE Access 2023.

[2] L.-Q. Chen, C. Wang, and S.-L. Song, ''Software defect prediction basedon nested-stacking and heterogeneous feature selection,'' *Complex Intell.Syst.*, vol. 8, no. 4, pp. 3333–3348, Aug. 2022

[3] M. Pavana, L. Pushpa, and A. Parkavi, ''Software fault prediction usingmachine learning algorithms,'' in *Proc. Int. Conf. Adv. Elect. Comput.Technol.*, 2022, pp. 185–197

[4] A. Al-Nusirat, F. Hanandeh, M. K. Kharabsheh, M. Al-Ayyoub, andN. Al-Dhfairi, ''Dynamic detection

of software defects using supervisedlearning techniques,'' *Int. J. Commun. Netw. Inf. Secur.*, vol. 11, no. 1,pp. 185–191, Apr. 2022.

[5] R. Bahaweres, F. Agustian, I. Hermadi, A. Suroso, and Y. Arkeman,''Software defect prediction using neural network basedSMOTE,'' in *Proc.7th Int. Conf. Electr. Eng., Comput. Sci. Informat. (EECSI)*, Oct. 2020,pp. 71–76

[6] N. Li, M. Shepperd, and Y. Guo, ''A systematic review of unsupervisedlearning techniques for software defect prediction,'' *Inf. Softw. Technol.*,vol. 122, Jun. 2020, Art. no. 106287.

[7] Y. Qiu, Y. Liu, A. Liu, J. Zhu, and J. Xu, ''Automatic featureexploration and an application in defect prediction,'' *IEEE Access*,vol. 7, pp. 112097–112112, 2019.

[8] A. Alsaeedi and M. Z. Khan, ''Software defect prediction using supervisedmachine learning and ensemble techniques: A comparative study,''*J. Softw. Eng. Appl.*, vol. 12, no. 5, pp. 85–100, 2019.

[9] C. Manjula and L. Florence, ''Deep neural network based hybrid approachfor software defect prediction using software metrics,'' *Cluster Comput.*,vol. 22, no. S4, pp. 9847–9863, Jul. 2019.

[10] R. Jayanthi and L. Florence, ''Software defect prediction techniques usingmetrics based on neural network classifier,'' *Cluster Comput.*, vol. 22,no. S1, pp. 77–88, Jan. 2019.

[11] A. Hammouri, M. Hammad, M. Alnabhan, and F. Alsarayrah, ''Softwarebug prediction using machine learning approach,'' *Int. J. Adv. Comput. Sci.Appl.*, vol. 9, no. 2, pp. 78–83, 2018.

[12] N. Kalaivani and R. Beena, ''Overview of software defect predictionusing machine learning algorithms,'' *Int. J. Pure Appl. Math.*, vol. 118,pp. 3863–3873, Feb. 2018.

[13] M. A. Memon, M.-U.-R. Magsi, M. Memon, and S. Hyder, ''Defectsprediction and prevention approaches for quality software development,''*Int. J. Adv. Comput. Sci. Appl.*, vol. 9, no. 8, pp. 451–457, 2018.

[14] E. Naresh and S. P. Shankar, ''Comparative analysis of the various datamining techniques for defect prediction using the NASA MDP datasets forbetter quality of the software product,'' *Adv. Comput. Sci. Technol.*, vol. 10,no. 7, pp. 2005–2017, 2017.

[15] D. Kumar and V. H. S. Shukla, ''A defect prediction model for softwareproduct based on ANFIS,'' *Int. J. Sci. Res. Devices* vol. 3, no. 10,pp. 1024–1028, 2016.

[16] P. Mandal and A. S. Ami, ''Selecting best attributes for software defectprediction,'' in *Proc. IEEE Int. WIE Conf. Electr. Comput. Eng.*, Dec. 2015,pp. 110–113.

[17] M. C. M. Prasad, L. F. Florence, and A. Arya, ''A study on software metricsbased software defect prediction using data mining and machine learningtechniques,'' *Int. J. Database Theory Appl.*, vol. 8, no. 3, pp. 179–190,Jun. 2015.

[18] A. Chug and S. Dhall, ''Software defect prediction using supervised learningalgorithm and unsupervised learning algorithm,'' in *Proc. 4th Int. Conf.Confluence Next Gener. Inf. Technol. Summit*, Noida, 2013, pp. 173–179.