International Conference on Intelligent Technologies & Science - 2021
(ICITS-2021)

# Software Quality Improvement using Machine Learning Techniques on Open Source Projects

**Karishma Sahni[1], Dr. Ganesh Chandra[2]**
Computer Science and Engineering,
Goel Institute of Technology & Management, Lucknow, India
karishmasahni903@gmail.com

**Abstract: Software reliability is a most important and integral part of the software quality and management. Software industries have been adopting various traditional methods to find the bugs and the software reliability. Many companies release various trail versions and are made available to the general public in form of various releases like the beta version and other and records the various bugs reported by users and also based on the feedback of the users. We will use the machine learning models like ensemble learning with both boosting and bagging techniques, decision tree models, KNN, Linear regression models. All the models will be trained on the SPSS 14.0 dataset and then the multdimensional result analysis is performed to obtained the best model to predict the software reliabity.**

**Keywords: ANSI, Machine Learning, Software Defect, SRGM**

## 1. Introduction:

Business applications which are critical in nature require reliable software, but developing such software's is a key challenge which our software industry faces today. With the increasing complexity of the software these days, achieving software reliability is hard to achieve. The primary goal of software reliability modeling is to find out the probability of a system failing in given time interval or the expected time span between successive failures

Software reliability is defined as the probability of failure-free software operation for a specified period of time in a specified environment (ANSI definition). Software reliability modeling has gained a lot of importance in the recent years. Criticality of software in many of the present day applications has led to a tremendous increase in the amount of work being carried out in this area. The use of intelligent neural network and hybrid techniques in place of the traditional statistical techniques have shown a remarkable improvement in the prediction of software reliability in the recent years. Among the intelligent and the statistical techniques it is not easy to identify the best one since their performance varies with the change in data. SRGM has been used for predicting and estimating number of errors remaining in the software

**ISO 9126 defines software quality as "the totality of features and characteristics of software product that bears on its ability to satisfy stated or implied needs"**

**While ISO 25000 takes the following approach to quality**

**"Capability of software products to satisfy stated and implied needs when used under specified conditions"**

## 2. Related Work:

"Machine Learning Approach for Quality Assessment and Prediction in Large Software Organizations" (Rakesh Rana, and Miroslaw Staron) [1], *Importance of software is being rising day by day and its complexity such as* measuring, maintaining and increasing software quality. Software metrics provide a quantitative means to measure and thus control various attributes of software systems. Assessing software quality early in the development process is essential to identify and allocate resources where they are needed most. Software quality, mainly the attributes related to dependability are even more important when developing software for systems deemed as safety, business and/or mission critical. Using machine learning techniques in conjunction to ISO/IEC 15939 measurement information model we can model overall quality of given software module/product/project. An algorithm or calculation combining one or more base and/or derived measures with associated decision criteria. It is based on an understanding of, or assumptions about, the expected relationship between the component measures and/or their behavior over time. Models produce estimates or evaluations relevant to defined information needs. The scale and measurement method affect the choice of analysis techniques or models used to produce indicators. As software becomes more integral part of our daily lives and its complexity increases - monitoring, assessing and improving software quality also becomes ever more important.

"Machine Learning-based Software Quality Prediction Models: State of the Art" (Hamdi A. Al-Jamimi and Moataz Ahmed) [2], *Quantification of parameters that affecting the quality of software and machine learning techniques being used to predict the quality of software. Software quality means a degree and satisfaction of the customers identified needs.*ML techniques have been utilized in many different problem domains as this field concentrates on building algorithms. Support Vector Machine, SVM for predicting the software fault-proneness modules. SVM being employed to the maintenance effort prediction. Bayesian network (BN) has been used in various studies in SWE area due its ability to integrate both empirical data and expert opinions. Neural networks (NN) as a tool for predicting the software faults. The relationship between the internal and external Quality attributes is surrounded with impression and uncertainty, different studies in the literature have made attempts to utilize the capability of fuzzy logic (FL) to estimate the software

quality. current software quality prediction modelling approaches utilizing the two major sources of knowledge for building the models. Pre-requisite to effective combination of knowledge sources is the transparency of the model. The survey revealed that none of the current models address the model transparency issue.

**3. Machine Learning:**

In machine learning, it is used catch on and learns properties from the data. The learning and prediction performance will effect on the quantity and quality of data set. Machine learning is a scientific discipline of exploring the construction and study of techniques that allow computer programs to learn from data without explicitly programmed (Smola & Vishwanathan, 2008). Basically, machine learning provides computer programs with capabilities to imitate the human learning process. This process is to observe a phenomenon and to generalize from the observations (Japkowicz & Shah, 2011). Machine learning is typically divided into three main categories: supervised and unsupervised learning. In unsupervised learning, algorithms are used to learn predictors from unlabeled data. Meanwhile, supervised learning learns prediction models based on a set of input data with label information. In supervised learning, the outputs can be real numbers in regression or class labels in classification. As classifying an input into two or more classes, supervised learning is sometimes known as classification. There are a variety of classification techniques that have been widely exploited in the literature for labeling software instances as defective or defect-free.

**3.1. Types of machine learning:**

**3.1.1 Supervised learning:**

Supervised learning is trained a labeled data. Supervised learning generally used in applications where historical data predict like future use. This technique further divided into two categories as regression and classification. In regression the label is continuous quantity. On the other hand, in classification the label is discrete.

**3.1.2 Unsupervised learning:**

Unsupervised learning used unlabeled data that has no historical labels.

**3.1.3 Reinforcement learning:**

It is used for gaming, navigation and robotics. This type of learning has three types of primary components: the learner, the environment and actions.

**3.2 Machine learning techniques:**

A machine learning algorithms are developed to build machine learning models and important machine learning process.

A machine learning algorithms are developed to build machine learning models and important machine learning process. Three classifier are :

- decision tree,
- naïve bayes
- support vector machine.

**4. Gradient Boosting**

Boosting is an ensemble modeling technique which attempts to build a strong classifier from the number of weak classifiers. It is done building a model by using weak models in series. Firstly, a model is built from the training data. Then the second model is built which tries to correct the errors present in the first model. This procedure is continued and models are added until either the complete training data set is predicted correctly or the maximum number of models are added.

Gradient Boosting is a popular boosting algorithm. In gradient boosting, each predictor corrects its predecessor's error. In contrast to Adaboost, the weights of the training instances are not tweaked, instead, each predictor is trained using the residual errors of predecessor as labels.

There is a technique called the Gradient Boosted Trees whose base learner is CART (Classification and Regression Trees).

The below diagram explains how gradient boosted trees are trained for regression problems.

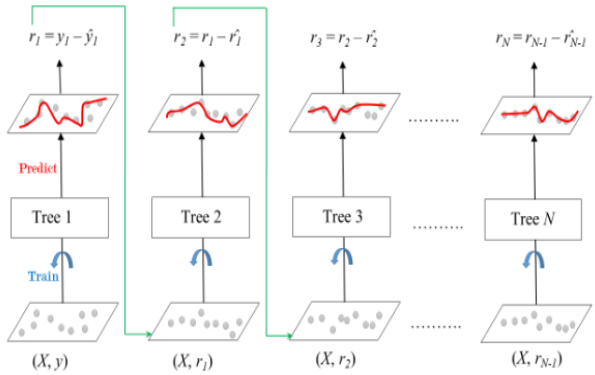


**Fig. 1. Gradient Boosted Trees for Regression.**

The ensemble consists of N trees. Tree1 is trained using the feature matrix X and the labels y. The predictions labelled y1(hat) are used to determine the training set residual errors r1. Tree2 is then trained using the feature matrix X and the residual errors r1 of Tree1 as labels. The predicted results r1(hat) are then used to determine the residual r2. The process is repeated until all the N trees forming the ensemble are trained.There is an important parameter used in this technique known as Shrinkage.

Shrinkage refers to the fact that the prediction of each tree in the ensemble is shrunk after it is multiplied by the learning rate (eta) which ranges between 0 to 1. There is a trade-off between eta and number of estimators, decreasing learning rate needs to be compensated with increasing estimators in order to reach certain model performance. Since all trees are trained now, predictions can be made.

Each tree predicts a label and final prediction is given by the formula,
Y (pred) = y1 + (eta * r1) + (eta * r2) + ..... + (eta * rN)

**5. AdaBoost Regressor**

AdaBoost was the first really successful boosting algorithm developed for the purpose of binary classification. AdaBoost is short for Adaptive Boosting and is a very popular boosting technique which combines multiple "weak classifiers" into a single "strong classifier". It was formulated by Yoav Freund and Robert Schapire. They also won the 2003 Gödel Prize for their work.

**Algorithm:**

1. Initialise the dataset and assign equal weight to each of the data point.
2. Provide this as input to the model and identify the wrongly classified data points.
3. Increase the weight of the wrongly classified data points..
4. if (got required results)
5. Goto step 5
6. else Goto step 2
7. End

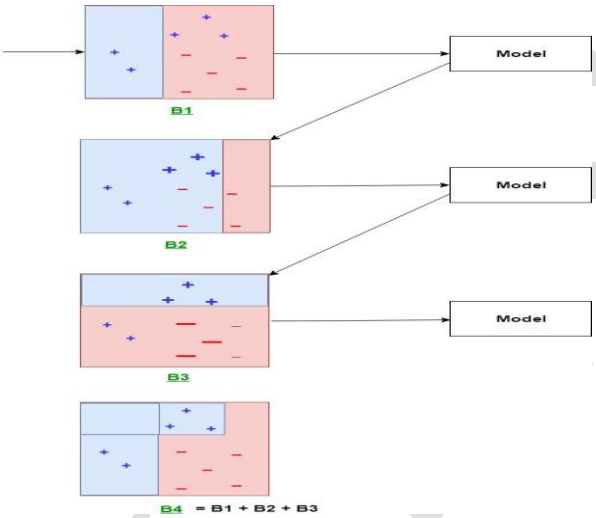


**Fig. 2. Adaboost regression model**.

Above diagram explains the AdaBoost algorithm in a very simple way. Let's try to understand it in a step wise process:

- **B1** consist of 10 data points which consist of two types namely plus(+) and minus(-) and 5 of which are plus(+) and other 5 are minus(-) and each one has been assigned equal weight initially. The first model tries to classify the data points and generates a vertical separator line but it wrongly classifies 3 plus(+) as minus(-).
- **B2** consists of the 10 data points from the previous model in which the 3 wrongly classified plus(+) are weighted more so that the current model tries more to classify these pluses(+) correctly. This model generates a vertical separator line which correctly classifies the previously wrongly classified pluses(+) but in this attempt, it wrongly classifies two minuses(-).
- **B3** consists of the 10 data points from the previous model in which the 3 wrongly classified minus(-) are weighted more so that the current model tries more to classify these minuses(-) correctly. This model generates a horizontal separator line which correctly classifies the previously wrongly classified minuses(-).
- **B4** combines together B1, B2 and B3 in order to build a strong prediction model which is much better than any individual model used.

The block diagram of the proposed work is shown in the figure 3. The open source data set is acquired and different regression and classification models are used to predict the software quality. We will use five different datasets for validating the model. Before using the data set to train model the dataset is pre-processed for skewness and the missing values.

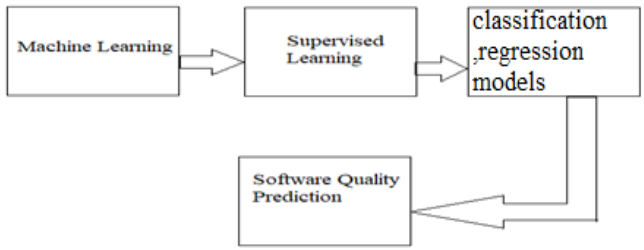


**Fig. 3: Adaboost regression model**.

**6. Result and Discussion:**

The proposed model is implemented by using the python language. Python is a dynamically typed oops based language. It is majorly used in the field of AI, deep learning, machine learning, data science and analytics, as well in field of web devolopment. The python provides various preprocessed api for the implemention of machine learning models. We have acquired 5 different open source dataset which we will use to devolop the software quality assurance model. The data set is orginally software failure data, which is made availabe by Musa (1979). The data set is of SPSS 14.0 availabe on http:// www.spss.com. The data set contains the cumulative failure time and inter failure time. the dataset is stored in comma seprated files which is read via the pandas (python data analysis) library. Then the data is visualized by using matplotlib library. The figure 4 shows the graph of the different data set used.

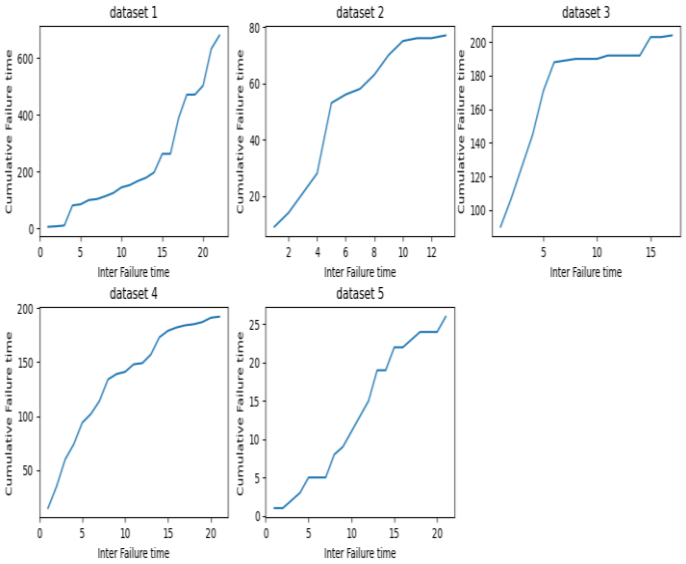


**Fig. 4. Correlation of the Different Data Set**

The data set is divided into 80:20 ratio as train and test dataset. The data is used to train the various available machine learning models. The machine learning models are used from the scikit-learn library. We used different models like KNeighbors, linear regression, lasso, ridge, svm, decision tree, random forest, adaboost, gradientboost. All the model is trained by using the first data and the predictions are made. The result shows that the ensemble models are having better accuracy compared to other machine learning models when we used the first dataset for training and testing .the prediction accuracy is as shown in the figure 5.

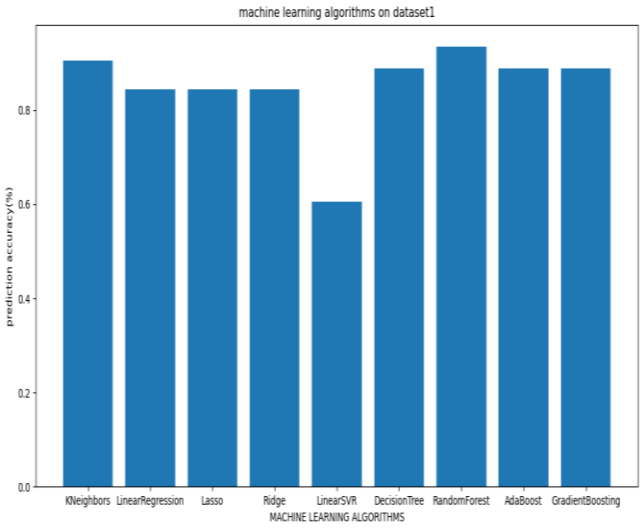


**Fig. 5. Result of machine learning models on first data set**

When the models were trained by using the second data set and the predictions were made. The result shows that the ensemble models are having better accuracy compared to other machine learning models when we used the second dataset for training and testing .the prediction accuracy is as shown in the figure 6.

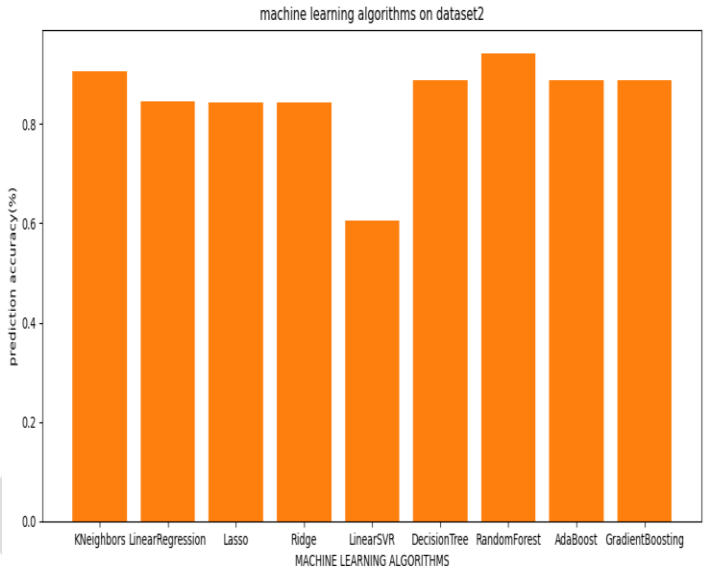


**Fig. 6: Result of machine learning models on second data set**

When the models were trained by using the third data set and the predictions were made. The result shows that the ensemble models are having better accuracy compared to other machine learning models when we used the third dataset for training and testing purpose .the prediction accuracy is as shown in the figure 7.

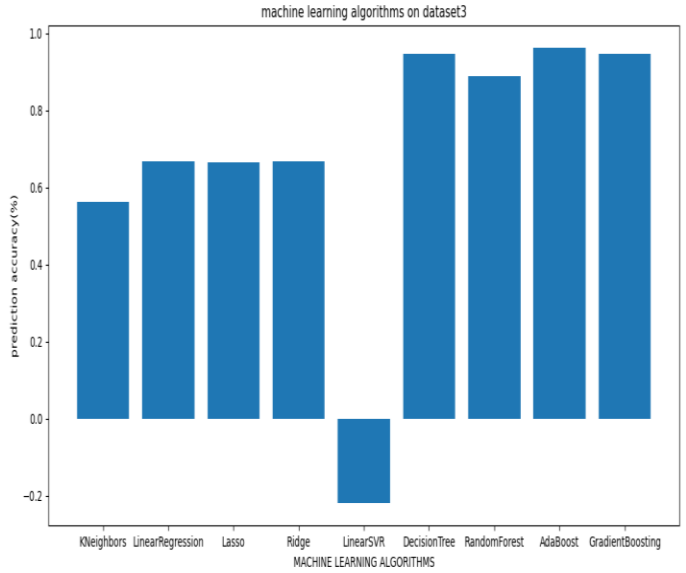


**Fig. 7: Result of machine learning models on third data set**

When the models were trained by using the fourth data set and the predictions were made. The result shows that the

kneighbors models are having better accuracy compared to other machine learning models when we used the fourth dataset for training and testing .the prediction accuracy is as shown in the figure 8.

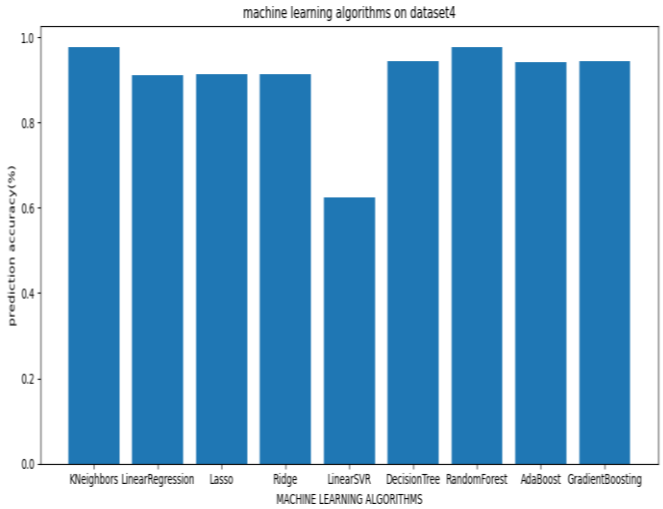


**Fig. 8. Result of machine learning models on fourth data set**

When the models were trained by using the fifth data set and the predictions were made. The result shows that the kneighbors models are having better accuracy compared to other machine learning models when we used the fifth dataset for training and testing .the prediction accuracy is as shown in the figure 9.

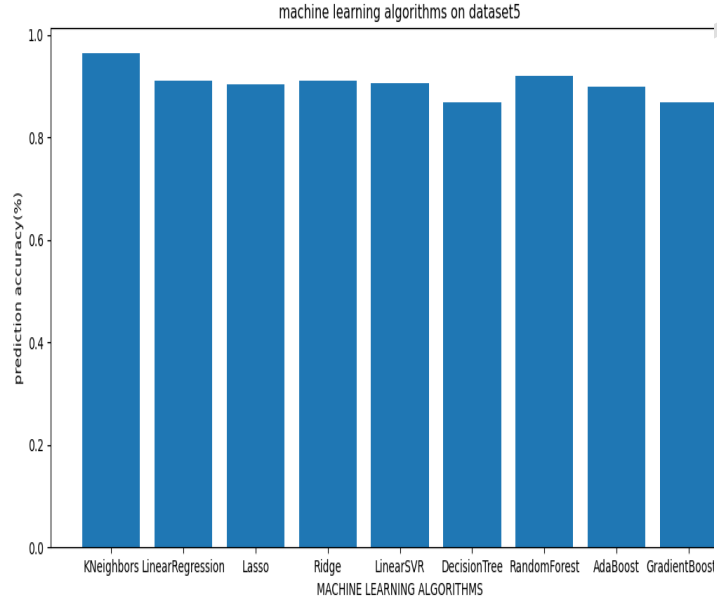


**Fig. 9 Result of machine learning models on fifth data set**

The figure 10 shows the prediction accuracy of the machine learning models on all the data set. The under fit model is svm and best fitted models were Kneighbors, and ensemble models like Random forest, Adaboost, Gradientboost algorithms.

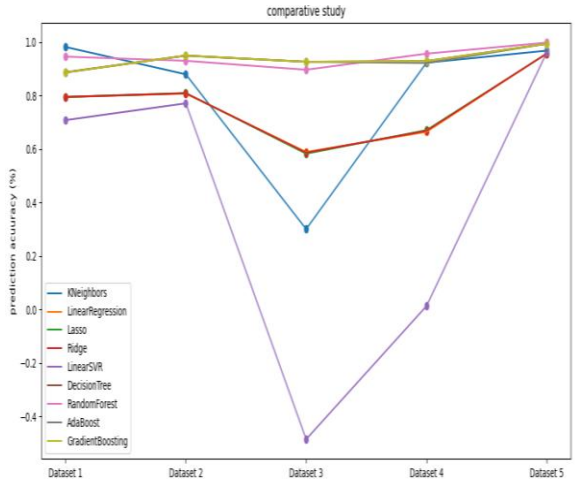


**Fig. 10: Result of machine learning models on all data set**

## 7. Conclusion:

Software reliability is the integral part of software quality. Software quality has been a major concern for the companies from past. The software reliability is unpredictable due to its probabilistic nature but with the help of machine learning models we have developed a prediction models for the prediction of software reliability. The ensemble models including both the bagging and boosting have outperformed all the other models. The ensemble learning models used were random forest, Adaboost, Gradientboost algorithms. The KNeighbors algorithm also have shown good results on the open source software reliability dataset. The SPSS 14.0 open source dataset was used to train the machine learning models. Software reliability changes frequently from software to software very quickly hence it is very difficult to predict the software reliability. We can improve the software reliability prediction by using the proposed methodology and a proper pre-processing of the dataset.

## References:

[1]. Rakesh Rana, and Miroslaw Staron (Issue 2015), "Machine Learning Approach for Quality Assessment and Prediction in Large Software Organizations," IEEE Transaction on https://ieeexplore.ieee.org/document/7339243.

[2]. Hamdi A. Al-Jamimi and Moataz Ahmed(Issue 2013),"Machine Learning-based Software Quality Prediction Models: State of the Art," IEEE Transaction on https://ieeexplore.ieee.org/document/6579473/

[3]. C. SenthilMurugan S. Prakasam "A Literal Review of Software Quality Assurance," International Journal of Computer Applications (0975 – 8887) Volume 78 – No.8, September 2013

[4]. Jyoti Devi, Nancy Seghal "Review of Improving Software Quality using Machine Learning Algorithms," IJCSMC, Vol. 6, Issue. 3 March 2017.

[5]. A. Sheta and D. Rine, "Modeling Incremental Faults of Software Testing Process Using AR Models ", the Proceeding of 4th International Multi-Conferences on Computer Science

and Information Technology (CSIT 2006), Amman, Jordan. Vol. 3. 2006.
[6]. D. Sharma and P. Chandra, "Software Fault Prediction Using MachineLearning Techniques," Smart Computing and Informatics. Springer, Singapore, 2018. 541-549.
[7]. R. Malhotra, "Comparative analysis of statistical and machine learning methods for predicting faulty modules," Applied Soft Computing 21, (2014): 286-297 [5] Malhotra, Ruchika. "A systematic review of machine learning techniques for software fault prediction." Applied Soft Computing 27 (2015): 504-518.
[8]. D'Ambros, Marco, Michele Lanza, and Romain Robbes. "An extensive comparison of bug prediction approaches." Mining Software Repositories (MSR), 2010 7th IEEE Working Conference on. IEEE, 2010.
[9]. Gupta, Dharmendra Lal, and Kavita Saxena. "Software bug prediction using object-oriented metrics." Sādhanā (2017): 1-15..
[10]. M. M. Rosli, N. H. I. Teo, N. S. M. Yusop and N. S. Moham, "The Design of a Software Fault Prone Application Using Evolutionary Algorithm," IEEE Conference on Open Systems, 2011.
[11]. T. Gyimothy, R. Ferenc and I. Siket, "Empirical Validation of ObjectOriented Metrics on Open Source Software for Fault Prediction," IEEE Transactions On Software Engineering, 2005.
[12]. Singh, Praman Deep, and Anuradha Chug. "Software defect prediction analysis using machine learning algorithms." 7th International Conference on Cloud Computing, Data Science & EngineeringConfluence, IEEE, 2017.
[13]. 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.
[14]. Okutan, Ahmet, and Olcay Taner Yıldız. "Software defect prediction using Bayesian networks." Empirical Software Engineering 19.1 (2014): 154-181.
[15]. Bavisi, Shrey, Jash Mehta, and Lynette Lopes. "A Comparative Study of Different Data Mining Algorithms." International Journal of Current Engineering and Technology 4.5 (2014).