JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

SOFTWARE BUG PREDICTION USING MACHINE LEARNING

Varshini Balanagu
Department of Information Technology
S.R.K.R. Engineering College(A)
SRKR Marg, Bhimavaram.

Tanuku Sai Lalitha Jyothi
Department of Information Technology
*S.R.K.R. Engineering College(A)***SRKR Marg, Bhimavaram.**

Vadlamuru Rithika Naga Anusha
Department of Information Technology
*S.R.K.R. Engineering College(A)***SRKR Marg, Bhimavaram.**

Rudraraju P D Chandini
Department of Information Technology
*S.R.K.R. Engineering College(A)***SRKR Marg, Bhimavaram.**

Dr.Bh. V. S. R. K. Raju
M.Tech,Ph.D,Professor, IT-HOD
Department of Information Technology
S.R.K.R. Engineering College(A)
SRKR Marg, Bhimavaram.

Abstract— **The entire success of software is impacted by software bug prediction (SBP), a crucial part of the software development and maintenance life cycles. It is necessary to anticipate issues in order to increase the software's dependability, efficiency, and cost. In spite of the fact that a number of methods have been put out in the literature, it isdifficult to create a reliable bug prediction model. In this paper, a machine learning (ML) based prediction method for software issues is introduced. Algorithms for guided machine learning that are based on historical data have been usedto predict software errors. Two of the classifiers are Support Vector Machine and Naive Bayes (NB). How accurate anduseful ML methods can be employed was demonstrated through the evaluation process. Included is the utilization of a comparative measure.**

Keywords— Naïve Bayes; Convolutional Neural Network; Decision Tree; Regression; Support Vector Machine; Fault Prediction.

I. INTRODUCTION

A software flaw is a flaw that makes a program crash or produce false results. The issue arises from faulty or insufficient logic. An error, omission, flaw, or defect that has the potential to lead to a failure or a departure from the expected results is referred toas a bug. Because they lower program quality, dependability, and efficiency while raising software costs, software failures must be anticipated.

Regression techniques have recently been used to reduce the sum of squared errors. However, this linear model is not very effective in detecting all kinds of defects. As a result, using that program is challenging because it will eventually produce more defects.

In the suggested approach, machine learning is used to pinpoint the modules where issues might be given priority. As a result, it improves the software's quality, effectiveness, and performance.

In this, a data set is used to anticipate software problems using machine learning techniques, and the accuracy of each data set isthen determined. The test case with the lowest accuracy rate should be changed because it has more bugs, and the test case withthe highest accuracy rate should be utilized without restriction.

II. LITERATURE SURVEY

Many researches have been conducted utilizing machine learning approaches to forecast software bugs.

Regression [9] is one of the machine learning techniques for determining the relationship between relevant variables; specifically, regression allows for the selection of the curve that best fits the available data. There are numerous regression techniques that can be used, and the type and quantity of independent and dependent variables will determine which technique is best. The goalof the regression is to reduce the errors' squared sum (least squares). In essence, this means that, despite any local variations, theregression algorithms produce the curve with the best overall fit. [11] Yet, it's possible that the performance of the regression modelson the test data will differ from that of the training data. The linear model has been demonstrated to be a good approximation forbug predictions. The linear model is also the simplest model that is currently available. Hence, using the carefully selected measurements and their weights calculated using the suggested algorithm, we resort to a linear model. Nevertheless, we use thevariations in the coefficients of determination (Rsquare) as the weights in the model rather than the regression coefficients directly.

A defect prediction model aims to find the components of new quality control techniques. An R(x) score is assigned to each module in the model, which stands for expected risk or relative error-proneness. The QA team [12] can choose the modules with thehighest scores (those displaying the most risk) for additional treatment, such as code reviews or unit tests, after ranking the modules using this score. Each sort of treatment has a different level of work involved. Although it could be difficult to quantify,it is unreasonable to assume that each file will necessitate the same amount of treatment labour. Based on the confusion matrix.It is based upon confusion matrix,

Radj(x) = R(x).RAFRAF =1-E(x)Emax

where the risk adjustment factor (RAF) is a treatment effort-sensitive variable. The ML abilities of SBP are assessed using ML classifiers. The Naïve Bayes (NB) classifier was discussed in the paper. Three separate datasets are used to test the discussed ML classifiers.

To enhance performance, an experiment [3] was run on the SBP model based on clusters. The researcher improves the performance after using the strategy, increasing recall from 31.6% to 79.2% and precision from 65.8% to 78.6%. The accuracyof the k-mean based clustering method used to determine how bug-prone object-oriented programming is was determined to be62.4%.

Disadvantages: Given that cluster demands better technology and design, it is pricey.

Deep learning gives convolutional neural networks [4] more consideration for proper feature extraction when utilised for bug prediction [5]. We can get great accuracy with this technique [4]. With this methodology, the train and test set software is first converted into abstract syntax trees. Each tree's selected nodes are then converted into token vectors. These token vectors, whichwere transformed into numerical vectors during the encoding stage, are provided to CNN. Convolutional Neural Networks generate structural and semantic program features that are combined with those currently employed for bug prediction. To ascertain whether the program has problems [13], the logistic regression model is then applied to the features.

Disadvantages: The absence of location and direction encoding in the prediction of objects is a drawback.

An experiment [6] was conducted on the SBP model based on the Decision Tree to improve performance. The outcome value inthe leaf of a "decision tree," a hierarchical prediction model [18], is reached by using observations as branches. A decision tree's building blocks are called decision nodes, and they are just a branch and a leaf node. The prediction accuracy of the model is 73%.

Disadvantages: The structure of the decision tree could be altered as a result of data changes. More duration and complexity. Thestructure of the decision tree could be altered as a result of data changes. More difficulty and time commitment.

Here, we'll invent hyperparameter optimization [7] for a machine learning model to forecast bugs [8]. Five projects employ these methods. The outcomes what was As was seen in each case, divide the datasets into a tuning set and an experiment set. Hyperparameters are tuned by employing a tuning set. Comparison of customised and default hyperparameter values for a machine learning model's prediction error [14]. To compare, cross-validation is employed. It is determined whether the tuning procedure results in higher accuracy by comparing it to the student's ttest. Also, it should be highlighted that the outcomes of hyper parameter optimization utilizing an Artificial Immune Network beat those of employing the hyper parameters that were used by default. The accuracy rate for the model is 73.9%.

III. SYSTEM IMPLEMENTATION (METHODOLOGY)

In this study, Naive Bayes (NB) and Support Vector Machine, two supervised machine learning algorithms, will be examined and evaluated (SVM). The study reveals the effectiveness and power of ML algorithms in predicting software defects in addition to comparing the various ML methods.

Supervisory machine learning techniques seek to create an inferring function that can be used to anticipate the values of outputsfrom new input data by identifying the connections and dependencies between the known inputs and known outputs of the labelled training data.

The data is collected from a software repository before being entered into the aforementioned methods, and the dataset shouldhave the following tabulated attributes from Table - 1.

Here, in Fig - 1, is the system architecture for software bug prediction. As depicted in Fig - 1, the dataset has been cleaned anddivided into training and testing data. Later on, this data is made available to the ML algorithms.

Table 1: Attributes in the dataset.

ATTRIBUTE	DESCRIPTION
loc	McCabe’s line count of code
V(g)	McCabe “cyclomatic complexity”
ev(g)	McCabe “essential complexity”
iv(g)	McCabe “design complexity”
n	Halsted total operators + operands
v	Halsted “volume”
l	Halsted “program length”
d	Halsted “difficulty”
i	Halsted “intelligence”
e	Halsted “effort”
b	Halsted blank
t	Halsted’s time estimator
IOCode	Halsted’s line count
IOComment	Halsted’s count of lines of comments
IOBlank	Halsted’s count of blank lines
IOCodeAndComment	Count of code and comments
Uniq_Op	Unique operators
Uniq_Opnd	Unique Operands
total_Op	Total operators
total_Opnd	Total operands
branchCount	Count of flow graph
problems	Module has/has not come or more reported defects

***Data Cleaning**:*

The process of data cleaning includes removing irrelevant parts of data, cleans the data and fill the empty parts of data using replacing with null values, replacing with mean values. For data cleaning, there are built in functions in pandas like isnull(), isnull.sum(), fillna(s.means()) through which we can verify whether data is cleaned or not and also to clean the data, we can gofurther for data preprocessing. By observing our attributes in the dataset, some attributes are not necessary for our analysis. So,we are dropping them.

Univariate Analysis:

Uni means “one”. It is the most straightforward type of data analysis since it just considers one variable, ignores any relationships or correlations between the data, and focuses mostly on description [17]. It takes the data, condenses it, and looks for patterns. Histograms are used in univariate analysis to analyze and display the frequency distribution. Plotting histograms in pandas are very easy and straightforward.

***Bivariate Analysis*:**

Bivariate analysis will be the next step. In this quantitative analysis, two variables are being examined. Finding an empirical relationship between them is the goal. The basic hypothesis of connection can be tested with the aid of bivariate analysis. Box plot, which is part of the Seaborn package, is utilized for this. To discover data linkage,

additional charts including the swarm plot, bar plot, and point plot are utilized.

Training and Testing data:

By providing various software features and an output that indicates if a defect is present or not, the data is trained. After training,the system is put to the test by being presented with various software features to see if it can identify software bugs. Using the train test split function in the sklearn package, which divides arrays into random train and test subsets, it is possible to divide data into training and testing. The dataset for this project is divided into 80% training data and 20% testing data.

Applying Naïve Bayes Model (NB):

A straightforward probabilistic classifier known as a naive bayes classifier is built using the Bayes theorem, naïve independence assumptions from Bayesian statistics, and classification rules. The naïve bayes model is easy to build and particularly useful for very large data sets. Even though Naïve Bayes is simple, it has been shown to outperform even more complicated classification techniques. Supervised learning is used to create predictions. It creates frequency tables using the practice data

set. Then, a straightforward Bayesian equation is used to calculate the posterior probability for each class. The set of forecasts'posterior probability has the highest value of any class. The project's naive bayes function, which is a part of the Sklearn package, will be used to identify the dataset's flaws.

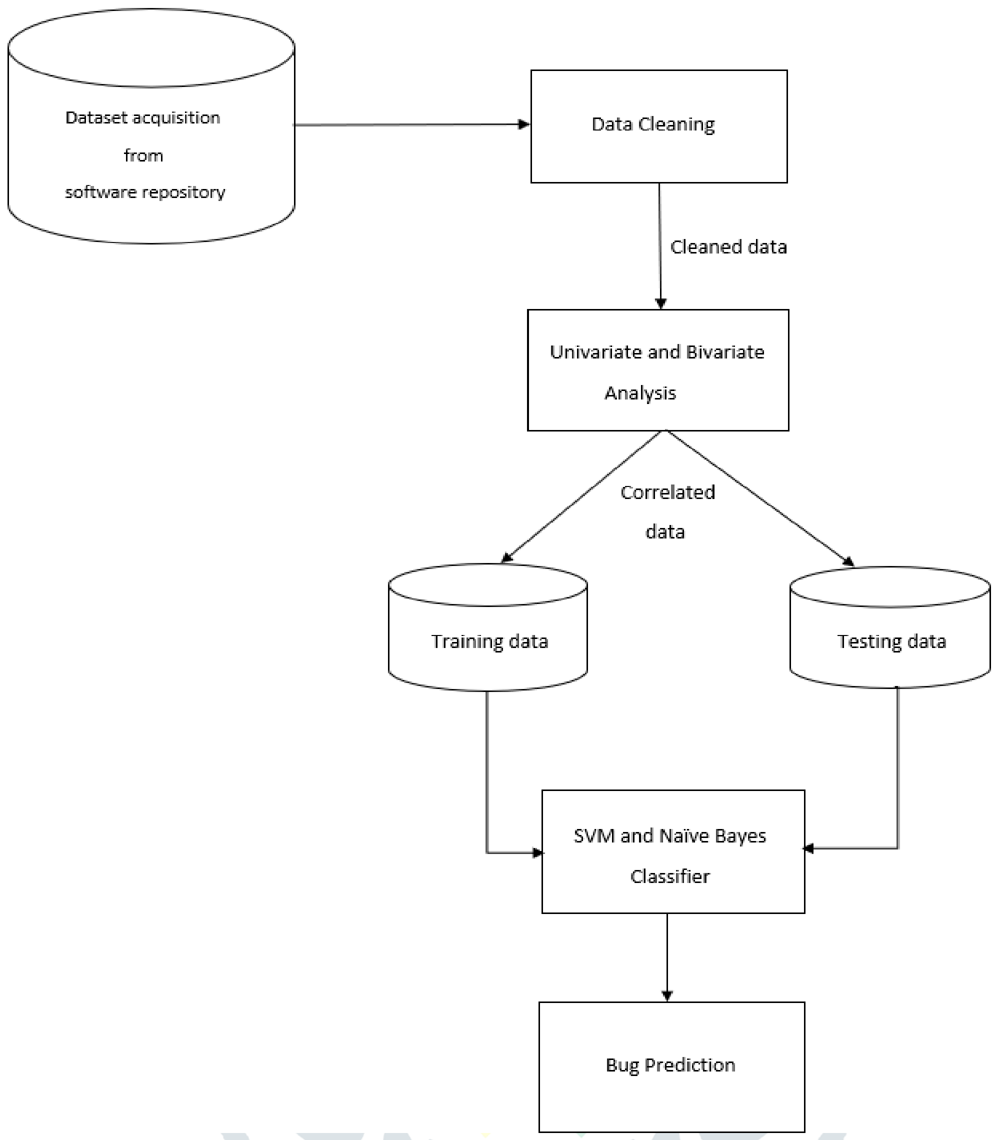


Fig. 1: System architecture of software bug prediction.

Applying Support Vector Machine (SVM):

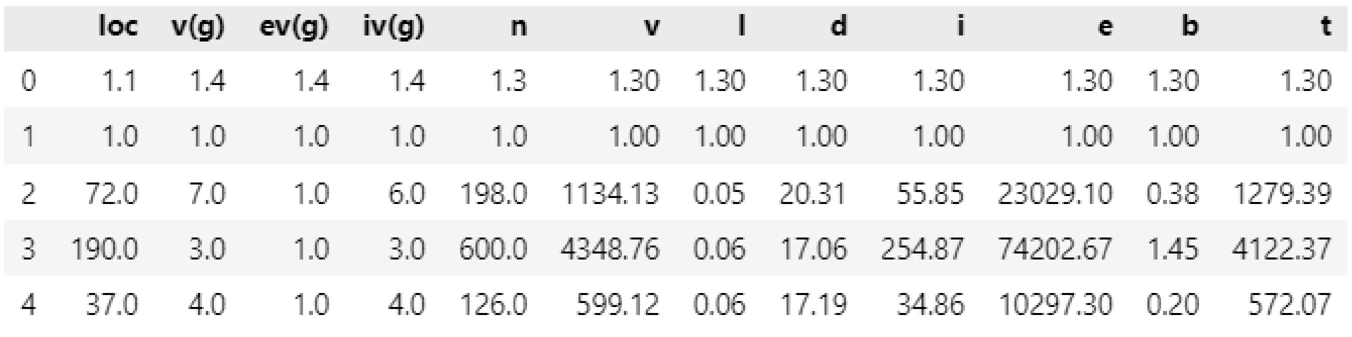
Classification or regression issues can be handled using the "Support Vector Machine" method of supervised machine learning. Yet, it is most typically used in classification-related situations. Each data point [15] is represented by the SVM technique as a pointin an n-dimensional space, where n is the number of features, and the value of each feature is a value at a specific position. Thehyperplane that effectively divides the two classes is then found, and classification is then completed. When there are no outliersin the data, the SVM approach performs better. [16] The dataset's issues will be found with the help of the project's support vector classifier, which is a component of the Sklearn package.

Front end Integration:

By using Flask, we integrate our machine learning model with front-end. In this, we give input values, it gives output as defectsare yes or no which means present or not.

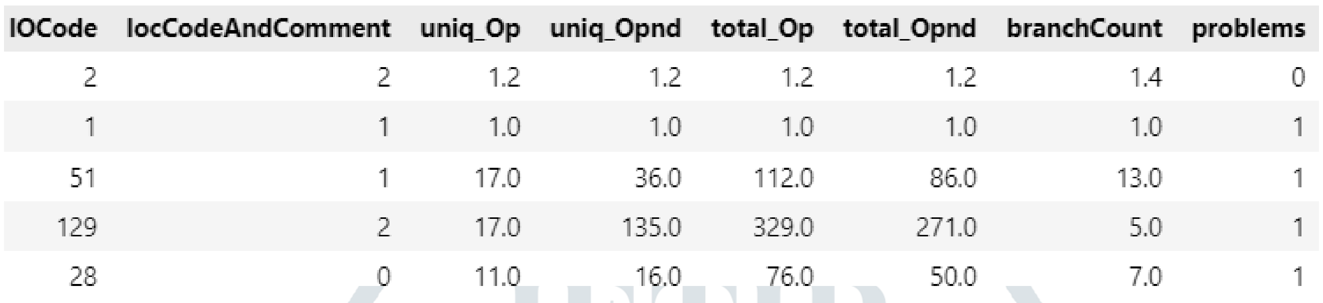
IV. EXPERIMENTS & RESULTS

Our model predicts the existence of bug in a software. After training the model with some of the machine learning algorithms we choose the respective algorithms i.e, Naïve Bayes and Support vector machine which gives better accuracy. Now we testedthe model with test data and based on the output accuracy we concluded, if accuracy is more for testing data the input softwarecontains less number of bugs and if accuracy is less, then the input software contains more number of bugs.



	loc	v(g)	ev(g)	iv(g)	n	v	l	d	i	e	b	t
0	1.1	1.4	1.4	1.4	1.3	1.30	1.30	1.30	1.30	1.30	1.30	1.30
1	1.0	1.0	1.0	1.0	1.0	1.00	1.00	1.00	1.00	1.00	1.00	1.00
2	72.0	7.0	1.0	6.0	198.0	1134.13	0.05	20.31	55.85	23029.10	0.38	1279.39
3	190.0	3.0	1.0	3.0	600.0	4348.76	0.06	17.06	254.87	74202.67	1.45	4122.37
4	37.0	4.0	1.0	4.0	126.0	599.12	0.06	17.19	34.86	10297.30	0.20	572.07

Fig. 2(a): Dataset after performing data cleaning.



lOCode	locCodeAndComment	uniq_Op	uniq_Opnd	total_Op	total_Opnd	branchCount	problems
2	2	1.2	1.2	1.2	1.2	1.4	0
1	1	1.0	1.0	1.0	1.0	1.0	1
51	1	17.0	36.0	112.0	86.0	13.0	1
129	2	17.0	135.0	329.0	271.0	5.0	1
28	0	11.0	16.0	76.0	50.0	7.0	1

Fig. 2(b): Dataset after performing data cleaning.

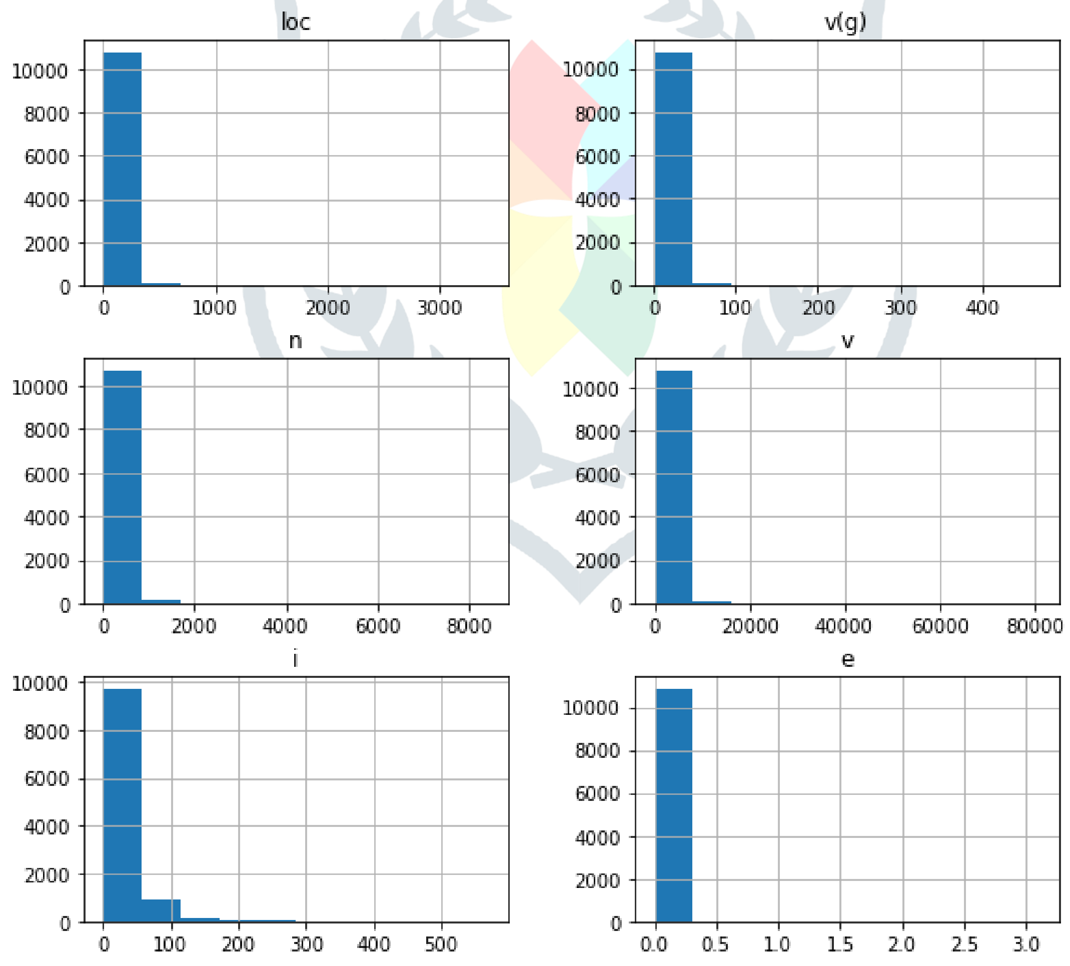


Fig. 3(a): Univariate Analysis (To find out the patterns in the data).

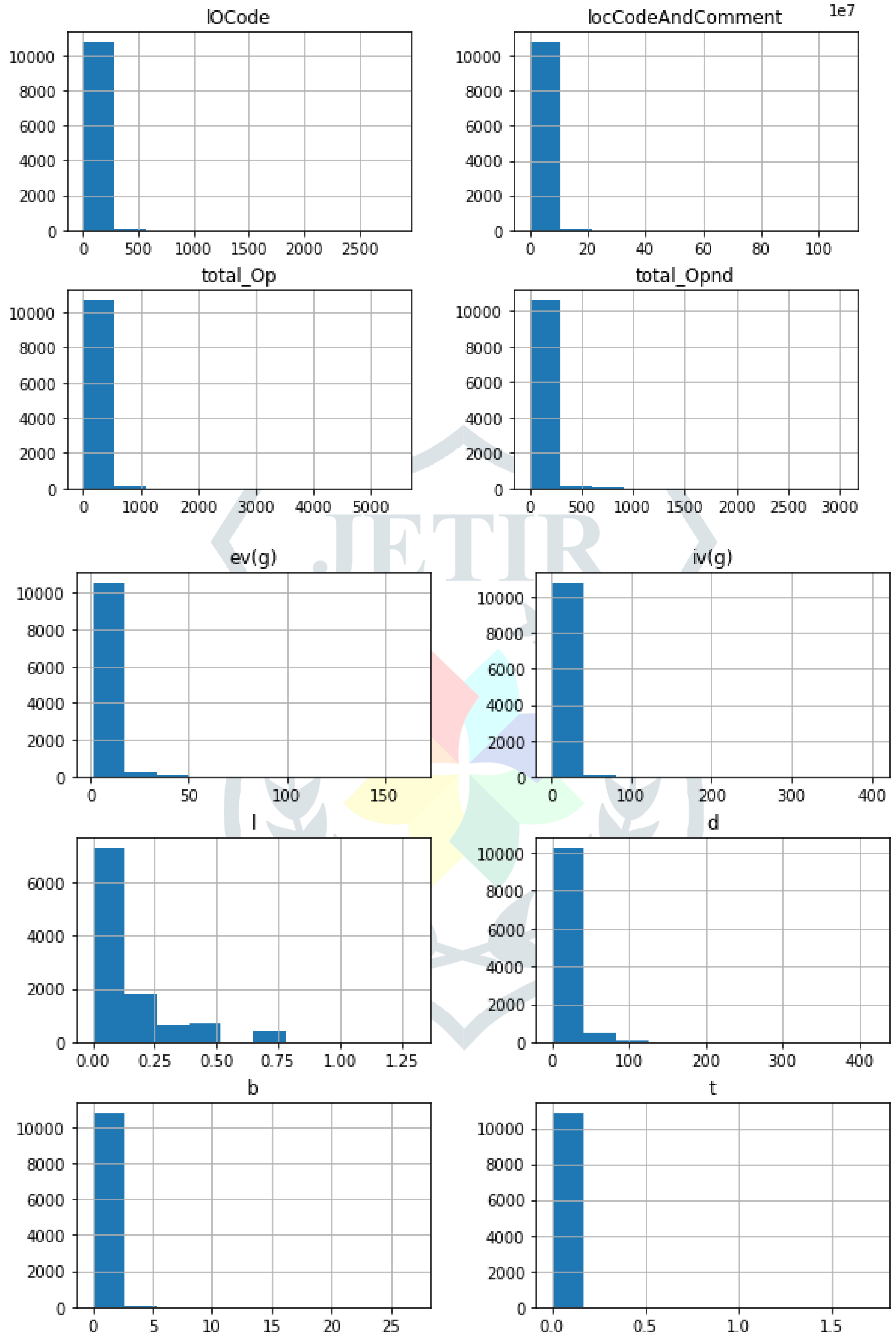


Fig. 3(b): Univariate Analysis (To find out the patterns in the data).

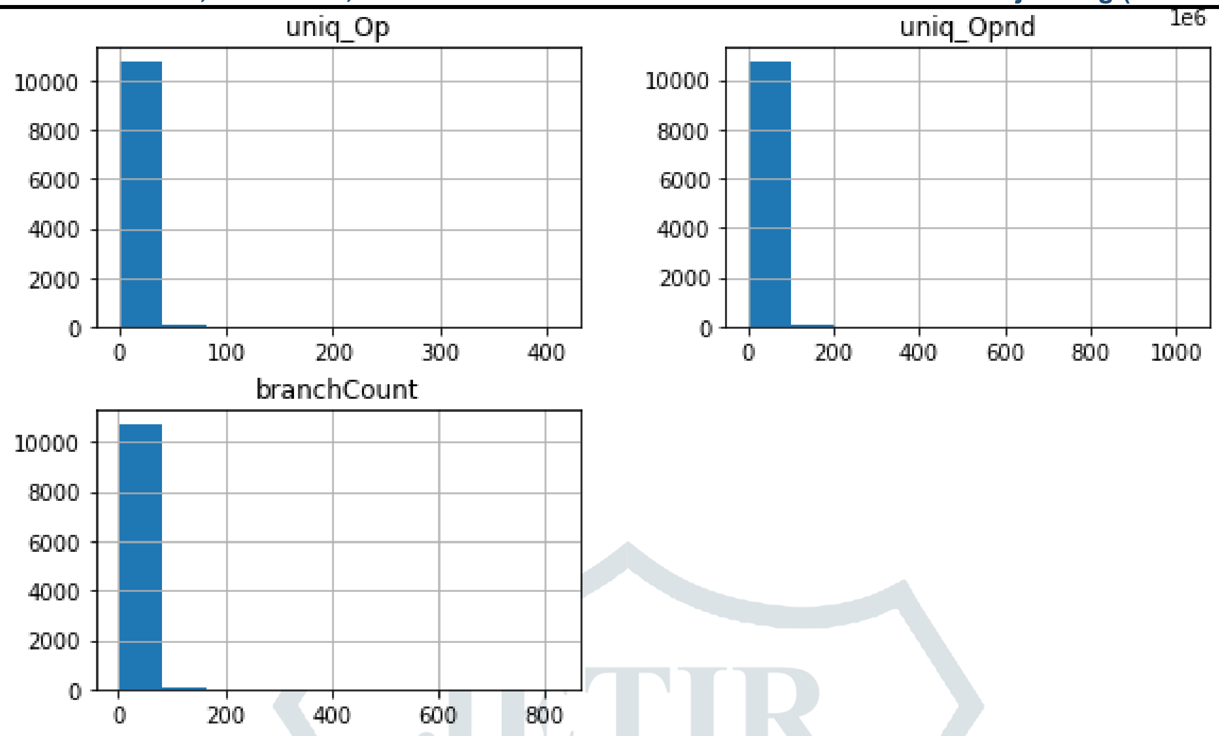
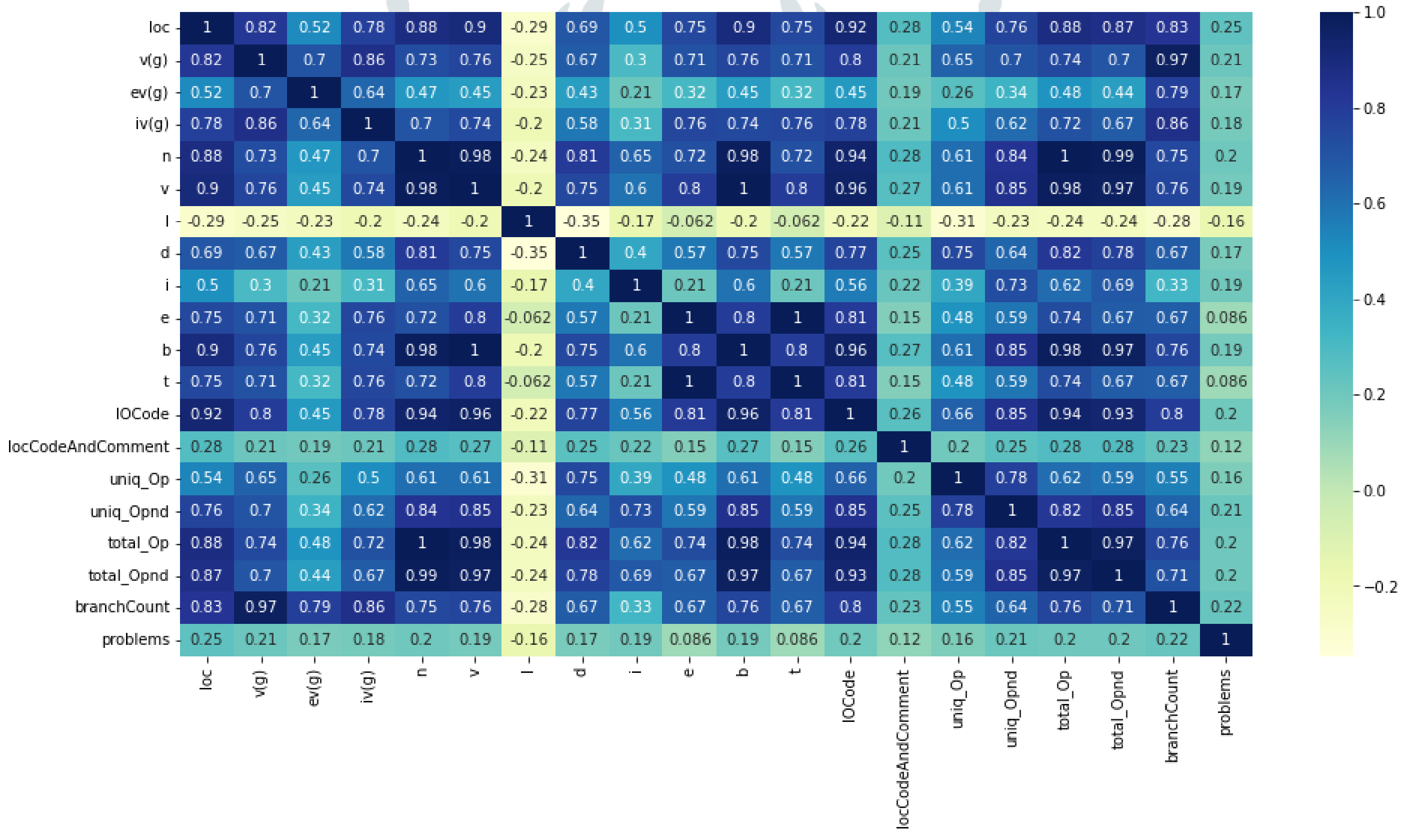


Fig. 3(c): Univariate Analysis (To find out the patterns in the data).



	loc	v(g)	ev(g)	iv(g)	n	v	l	d	i	e	b	t	lOCode	locCodeAndComment	uniq_Op	uniq_Opnd	total_Op	total_Opnd	branchCount	problems
loc	1	0.82	0.52	0.78	0.88	0.9	-0.29	0.69	0.5	0.75	0.9	0.75	0.92	0.28	0.54	0.76	0.88	0.87	0.83	0.25
v(g)	0.82	1	0.7	0.86	0.73	0.76	-0.25	0.67	0.3	0.71	0.76	0.71	0.8	0.21	0.65	0.7	0.74	0.7	0.97	0.21
ev(g)	0.52	0.7	1	0.64	0.47	0.45	-0.23	0.43	0.21	0.32	0.45	0.32	0.45	0.19	0.26	0.34	0.48	0.44	0.79	0.17
iv(g)	0.78	0.86	0.64	1	0.7	0.74	-0.2	0.58	0.31	0.76	0.74	0.76	0.78	0.21	0.5	0.62	0.72	0.67	0.86	0.18
n	0.88	0.73	0.47	0.7	1	0.98	-0.24	0.81	0.65	0.72	0.98	0.72	0.94	0.28	0.61	0.84	1	0.99	0.75	0.2
v	0.9	0.76	0.45	0.74	0.98	1	-0.2	0.75	0.6	0.8	1	0.8	0.96	0.27	0.61	0.85	0.98	0.97	0.76	0.19
l	-0.29	-0.25	-0.23	-0.2	-0.24	-0.2	1	-0.35	-0.17	-0.062	-0.2	-0.062	-0.22	-0.11	-0.31	-0.23	-0.24	-0.24	-0.28	-0.16
d	0.69	0.67	0.43	0.58	0.81	0.75	-0.35	1	0.4	0.57	0.75	0.57	0.77	0.25	0.75	0.64	0.82	0.78	0.67	0.17
i	0.5	0.3	0.21	0.31	0.65	0.6	-0.17	0.4	1	0.21	0.6	0.21	0.56	0.22	0.39	0.73	0.62	0.69	0.33	0.19
e	0.75	0.71	0.32	0.76	0.72	0.8	-0.062	0.57	0.21	1	0.8	1	0.81	0.15	0.48	0.59	0.74	0.67	0.67	0.086
b	0.9	0.76	0.45	0.74	0.98	1	-0.2	0.75	0.6	0.8	1	0.8	0.96	0.27	0.61	0.85	0.98	0.97	0.76	0.19
t	0.75	0.71	0.32	0.76	0.72	0.8	-0.062	0.57	0.21	1	0.8	1	0.81	0.15	0.48	0.59	0.74	0.67	0.67	0.086
lOCode	0.92	0.8	0.45	0.78	0.94	0.96	-0.22	0.77	0.56	0.81	0.96	0.81	1	0.26	0.66	0.85	0.94	0.93	0.8	0.2
locCodeAndComment	0.28	0.21	0.19	0.21	0.28	0.27	-0.11	0.25	0.22	0.15	0.27	0.15	0.26	1	0.2	0.25	0.28	0.28	0.23	0.12
uniq_Op	0.54	0.65	0.26	0.5	0.61	0.61	-0.31	0.75	0.39	0.48	0.61	0.48	0.66	0.2	1	0.78	0.62	0.59	0.55	0.16
uniq_Opnd	0.76	0.7	0.34	0.62	0.84	0.85	-0.23	0.64	0.73	0.59	0.85	0.59	0.85	0.25	0.78	1	0.82	0.85	0.64	0.21
total_Op	0.88	0.74	0.48	0.72	1	0.98	-0.24	0.82	0.62	0.74	0.98	0.74	0.94	0.28	0.62	0.82	1	0.97	0.76	0.2
total_Opnd	0.87	0.7	0.44	0.67	0.99	0.97	-0.24	0.78	0.69	0.67	0.97	0.67	0.93	0.28	0.59	0.85	0.97	1	0.71	0.2
branchCount	0.83	0.97	0.79	0.86	0.75	0.76	-0.28	0.67	0.33	0.67	0.76	0.67	0.8	0.23	0.55	0.64	0.76	0.71	1	0.22
problems	0.25	0.21	0.17	0.18	0.2	0.19	-0.16	0.17	0.19	0.086	0.19	0.086	0.2	0.12	0.16	0.21	0.2	0.2	0.22	1

Fig. 4: Bivariate Analysis using HeatMap (To visualize the strength of correlation among variables).

Based on the characteristics listed in Table 1, we evaluate the effectiveness of the software bug prediction system. A final resultis obtained, which can be referred to as the accuracy of the supplied input, after all computations based on the qualities have been determined. After that, the accuracy is plotted on a hyper-plane to show whether or not the input has flaws.

Table 2: Accuracy of the predicted models for given training dataset

Machine learning model	Accuracy in percentage
Naïve Bayes (NB)	80.42%
Support vector machine (SVM)	81.80%

Table 3: Comparison of different ML methods applied

Machine learning model	Accuracy in percentage
Clustering	62.4%
Decision Tree	73%
Random Forest Classifier	78.39%
Naïve Bayes (NB)	**80.42%**
Support vector machine (SVM)	**81.80%**

V. CONCLUSION & FUTURE WORK

Software bug prediction is a method that creates a prediction model from previous data to foretell potential software problemsin the future. With the aid of various datasets, metrics, and performance measurements, several strategies have been proposed. This study evaluated how machine learning techniques were used to the challenge of predicting software bugs. Machine learning methods NB and SVM have both been applied. Two actual testing/debugging datasets are used to put the evaluation method into effect. Measures for recall, accuracy are used to construct the experimental data. Findings show that ML approaches can be used to predict upcoming software issues. Although the suggested approaches improve software bug prediction, there are still a number of areas that need more study. In a later project, we might use other ML methods and provide a thorough comparison of them. Also, one method to improve the prediction model's accuracy is to include new software metrics during the learning phase.

References

[1] Analysis on Detecting a Bug in a Software using Machine Learning - Rashmi P, Prashanth Kambli, International Journal ofRecent Technology and Engineering (IJRTE) ISSN: 2277- 3878 (Online), Volume-9 Issue-2, July 2020.

[2] Machine Learning Techniques for Software Bug Prediction: A Systematic Review - Syahana Nur'Ain Saharudin, Koh TiengWei and Kew Si Na, 14 Nov 2020.

[3] Feidu Akmel, Ermiyas Birihanu, Bahir Siraj "A Literature Review Study of Software Defect Prediction using Machine Learning Techniques," IJERMT, ISSN: 2278-9359 (Volume6, Issue-6), June 2017.

[4] Amruthalakshmi, Prashanth Kambli, Naresh E, "Recognition of Handwritten Digits Using Convolutional Neural Network and Linear Binary Pattern," International Journal of Innovative Technology and Exploring Engineering, ISSN: 2278-3075, Volume-9, Issue-1 November 2019.

[5] J. Li, P. He, J. Zhu and M. R. Lyu, "Software Defect Prediction via Convolutional Neural Network," 2017.

[6] Awni Hammouri, Mustafa Hammad, Mohammad Alnabhan, Fatima Alsarayrah, "Software Bug Prediction using Machine Learning Approach," IJACSA, Vol. 9, No. 2, 2018.

[7] H. Osman, M. Ghafari and O. Nierstrasz, "Hyperparameter optimization to improve bug prediction accuracy," 2017 IEEE Workshop on Machine Learning Techniques for Software Quality Evaluation (MaLTeSQuE), Klagenfurt, 2017, pp. 33-38, doi: 10.1109/MALTESQUE.2017.7882014.

[8] Sultan Alamri, Bushra Mumtaz, F Faiza Khan, Summrina Kanwal, "Hyper-Parameter Optimization of Classifers, Using an Artificial Immune Network and Its Application to Software Bug Prediction," IEEE Access (Volume: 8), 2020.

[9] Shruthi Puranika, Pranav Deshpandea, K Chandrasekarana "An Innovative Method For Machine Learning To Forecast Bugs ," 2016.

[10] Aqsa Rahim, Muhammad Abbas "Software Defect Prediction using Naïve Bayes classifier", 2021 International Bhurban conference on Applied sciences and technologies (IBCAST).

[11] Bh. V. S. R.K. Raju Et Al "Sustainable and reliable healthcare automation and digitization using deep learning technologies", Journal of Scientific and Industrial Research (JSIR),2023 , ISSN: 0975-1084.

[12] Bh. V. S. R.K. Raju Et Al "GW-CNNDC: Gradient Weighted CNN Model for Diagnosing Covid-19 UsingRadiography X- Ray Images", Measurement:Sensors, Elsevier, 2023 , ISSN: 2665-9174.

[13] Bh. V. S. R.K. Raju Et Al "A Novel Technique of Threshold Distance-Based Vehicle Tracking System for Woman Safety", Intelligent System Design, Lecture Notes in Networks and Systems, Springer, India -2022, Pages:567-577, ISSN: 2665-9174.

[14] Bh. V. S. R.K. Raju Et Al "Handling Emotional Speech: a prosody based data augmentation technique for improvingneutral speech trained ASR systems", International Journal of Speech Technology, Sep -2021, ISSN: 1381-2416.

[15] Bh. V. S. R.K. Raju Et Al "A Novel Technique For Prediction Of Coronary Artery Disease From Human Fundus Images Using Machine Learning Approach", International Journal For Innovative Engineering And Management Research
Elsevier SSRN-2020, Pages: 655-661, ISSN: 2456-5083.

[16] Bh. V. S. R.K. Raju Et Al "Recognition of Human Being Through Handwritten Digits Using Image Processing Techniques And Ai", International Journal For Innovative Engineering And Management Research, Dec-2020, pages: 69-74, ISSN: 2456- 5083.

[17] Bh. V. S. R.K. Raju Et Al "An Experimental Analysis Of Secure-Energy Trade-Off Using Optimized Routing Protocol
In Modern-Secure-Wsn", Eai Endorsed Transactions On12 Scalable Information Systems, Issue, Mar-2020, Pages: 1-7, ISSN: 2032-9407.

[18] Bh. V. S. R.K. Raju Et Al "A Novel Robotic aid for physically challenged Implemented using Image Processing", Journal of Engineering Research and Application, IJERA, Feb-2022, Pages:53-57, ISSN: 2248-9622.