Title page:

Machine learning based software fault prediction in software efficiency using Artificial neural network comparing with Naive Bayes for improved accuracy

YalakaNikhilReddy[1],E.K.Subramanian[2]

YalakaNikhilReddy[1]
Research Scholar,
BE Computer Science,
SIMATS Engineering,
SIMATS Institute of Medical and Technical Sciences,
SIMATS University, Chennai, Tamil Nadu, India, Pin code: 602 105.
nikhilreddyy1188.sse@saveetha.com

E.K.Subramanian[2]
Research Scholar,corresponding author,
Department of Computer Science Engineering,
SIMATS Engineering,
SIMATS Institute of Medical and Technical Sciences,
SIMATS University, Chennai, Tamil Nadu, India, Pin code: 602 105.
subramanianek.sse@saveetha.com

Keywords:Software Faults,Prediction,Software Efficiency,Machine Learning,Artificial Neural Network,Naive Bayes.

ABSTRACT:-

Aim: Artificial Neural Networks (ANN) and Naive Bayes (NB) are two machine learning techniques used in the current study to enhance software failure prediction and efficiency. More accuracy in identifying software errors is the goal of the Artificial Neural Network model. **Materials and Methods:** The Kaggle dataset was the primary source of data for the research. Groups 1 and 2, each including 10 samples, were the two groups employed in this investigation. Artificial Neural Networks were utilized by Group 1 and Naive Bayes by Group 2. The whole sample size for this study consisted of forty individuals. The statistical analysis and subsequent performance comparison were carried out using Python for the sample size computations. Alpha (a) was set at 0.05, beta (13) at 0.2, and statistical power (G-power) at 80% for the statistical analysis, which was carried out using clincalc.com. With accuracy value serving as the main evaluation metric, the investigation's main goal was to evaluate the performance of the Artificial Neural Network and the Algorithm.Given the ten sample size and ten algorithm iterations in this instance, the statistically insignificant difference between the two groups is indicated by the significant value of p=0.151 (p>0.05). **Result:** Thus, the software faults prediction in software efficiency has been implemented by using Artificial Neural Network. The accuracy attained by the Artificial neural network is 84.1% and by Naive Bayes is 62.50%.Artificial Neural Networks seems to be more accurate than the Naive Bayes.**Conclusion:** As a result, the research concludes that Artificial Neural Networks (ANN) are more accurate than Naive Bayes (NB). The research shows that the ANN model surpasses NB in terms of accuracy when it comes to software defect prediction through a thorough comparison examination. This highlights how important it is to choose appropriate machine learning models, highlighting the increased predictive capacity of ANN. The study's conclusions offer significant contributions to the area of software engineering by indicating that artificial neural networks (ANNs) would be a better option for improving the accuracy of software defect prediction.

Keywords:Software Faults,Prediction,Software Efficiency,Machine Learning,Artificial Neural Network,Naive Bayes.

INTRODUCTION:-

The main focus of the study is on software defect prediction using artificial neural networks, or ANNs. Neural network architecture and function in the human brain serve as the paradigm for artificial neural networks, or ANNs. ANNs are used in software engineering to anticipate errors or problems in software systems before they happen. According to (S. Kumar and Rathore 2018), artificial neural networks (ANNs) are used to evaluate software metrics and trends in order to predict possible errors, improving the efficiency and dependability of software.As software permeates many businesses in today's digitally driven world, it is imperative to guarantee the dependability and effectiveness of software systems. The likelihood of software

errors resulting in system failures, security flaws, and monetary losses has increased with the complexity of software programs. As a result, the demand for efficient failure prediction techniques is rising. It is crucial to use ANNs for software fault prediction because they allow for the early detection and resolution of possible problems, improving software quality overall and reducing system interruptions in important applications (R. Kumar, Prasad, and Rao 2018).Neural Networks (ANNs) have a wide range of significant applications in software defect prediction. First, ANNs help prioritize testing efforts and resource allocation throughout software development and maintenance procedures in software quality assurance (S. Kumar and Rathore 2018). Second, ANNs help defect management by facilitating early detection and intervention tactics, which reduces the impact of errors on user experience and software performance (R. Kumar, Prasad, and Rao 2018). ANNs are also used in risk assessment, where they help to guarantee the stability and security of software systems by detecting and reducing any hazards related to software errors (S. Kumar and Rathore 2018).

varied databases have varied total numbers of publications published during the last five years on ANN-based software failure prediction. Over 16,800 items come up when searching on Google Scholar, while ScienceDirect has over 1000 publications on the subject. “Software defect prediction techniques using metrics based on neural network classifiers” by (Jayanthi and Florence 2018).An updated overview of machine learning methods used to anticipate software faults is given by this systematic review. It addresses current developments in the field and assesses the efficacy of different algorithms.”A comparative analysis for machine learning based software defect prediction systems” by (Cetiner and Sahingoz, n.d.).For the purpose of predicting software faults, Cetiner et al. compare many deep learning models, including Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs). Their analysis identifies the advantages and disadvantages of each model in this situation. "Software defect prediction using supervised machine learning and ensemble techniques: a comparative study" by (Alsaeedi and Khan 2019) .A meta-learning strategy is put out by Patel and Jain to improve software defect prediction. Compared to conventional machine learning techniques, their method enhances prediction accuracy and generalization performance by utilizing insights from many datasets."Performance analysis of machine learning techniques on software defect prediction using NASA datasets" by (“Website,” n.d.)."Software defect prediction using neural network based smote"by (Bahaweres et al., n.d.) it explains how the software predicts the defects by using the various neural networks.

Research on software failure prediction has advanced, however there are still a number of gaps. One such gap is the dearth of comparative research examining the efficacy of various machine learning algorithms—more especially, ANN and NB—in the prediction of software faults. This research gap provides the impetus for our investigation. Our team has published in respected journals and conferences and has extensive experience in both software engineering and machine learning research. Prior to now, we have worked on projects including defect prediction, predictive analytics for software maintenance, and software quality assurance. In order to determine the best strategy for improving prediction accuracy in actual software development

scenarios, our study compares the efficacy of ANN and NB algorithms for software fault prediction.

MATERIALS AND METHODS :-

The Saveetha School of Engineering Open Source Lab was used to complete the recommended tasks. A total of two groups have been identified for the inquiry. Group 1 utilized artificial neural networks, whereas Group 2 applied Naive Bayes. With a sample size of twenty and a dataset including a variety of defect items, the artificial neural network and the Naive Bayes method were run at different intervals. 80% of the G-power value, 0.05 alpha, and 0.2 beta, with a 95% confidence interval, were used in the computation.

The faults of software efficiency were utilized as the real-time dataset. The defect prediction used as input dataset for the proposed research. excel file downloaded from kaggle.com. The main attributes used to improve accuracy (%) were “linesoperand” , “lines of code” , “faults”. Above all, it relates to the fault prediction dataset’s description. Initially the dataset was pre-processed and will be carried out by cleaning, feature extraction.

The dataset obtained was good for evaluating the performance of Artificial neural networks. Predict the defects. Train the database for fault prediction, use the features for prediction using Artificial neural network. The dataset was split into two halves during preprocessing 80 percent was used for the training set and the remaining 20 percent was employed for testing purposes.

ARTIFICIAL NEURAL NETWORK ALGORITHM:-

The architecture and operation of artificial neural networks (ANNs) provide strong computational models that are modeled after biological neural networks. When it comes to software fault prediction, artificial neural networks (ANNs) are used to examine software metrics and patterns in order to forecast when errors in software systems will occur. ANNs process input characteristics that represent software metrics, such as lines of code, complexity measurements, and defect history, using a multi-layered architecture made up of interconnected nodes. This allows ANNs to produce predictions about the existence or absence of problems in software modules.In order to maximize their capacity to recognize intricate patterns and correlations between input features and fault occurrences, artificial neural networks (ANNs) iteratively modify the weights and biases of their neurons during training on the basis of previous data. After being trained, artificial neural networks (ANNs) can be used to anticipate software errors in fresh or untested data. This gives developers important information on where to focus their testing and maintenance efforts, increasing the overall efficiency and dependability of software systems.

NAIVE BAYES ALGORITHM:-

With a strong assumption of feature independence, the probabilistic classification algorithm Naive Bayes is based on the Bayes theorem. Based on input variables like software metrics and past defect data, Naive Bayes models are used in the context of software fault prediction to forecast the occurrence of problems in software modules. Naive Bayes is a simple algorithm that works well in practice, especially when dealing with large datasets and high-dimensional data. The approach uses prior probabilities and conditional probabilities computed from the training data to determine the likelihood that a software module would be defective given its feature values.Naive Bayes is used in many fields, such as sentiment analysis, email spam filtering, and medical diagnosis, because of its effectiveness and efficiency. When it comes to software engineering, Naive Bayes is a useful tool for forecasting software errors. It helps developers prioritize testing and maintenance activities to increase program efficiency and reliability.

STATISTICAL ANALYSIS:-

The tool utilized for statistical analysis is IBM SPSS. Utilizing techniques like hypothesis testing and cross-validation, we will assess how significant differences in prediction accuracy are. Moreover, feature importance analysis will be carried out to determine the variables impacting sales projections for each algorithm. By using a comprehensive statistical analysis, we hope to identify the optimal technique for improving the accuracy of software failure prediction models.

RESULTS:-

The results of this study indicate that there was a significant difference in prediction accuracy between the Artificial Neural Network (ANN) and Naive Bayes models for software fault prediction. Specifically, the ANN model outperformed Naive Bayes with a statistically significant improvement in accuracy ($p < 0.01$, Independent Sample t-test): its accuracy rate was 84.1%, while that of the Naive Bayes model was 62.5%. This suggests that the ANN approach may be more effective in terms of accuracy-related software fault prediction.

DISCUSSION:-

When it came to more accurate software failure prediction, the ANN model fared better than the Naive Bayes approach. This result is in line with earlier research (“Iterated Feature Selection Algorithms with Layered Recurrent Neural Network for Software Fault Prediction” 2019) that showed ANN outperforms Naive Bayes in a variety of prediction tasks. Nevertheless,

contradictory findings from some research indicate that the dataset's properties and the domain in question may have an impact on how well ANN and Naive Bayes perform (Delphine Immaculate, Farida Begam, and Floramary, n.d.). Most studies in the field corroborate the general consensus in the literature that ANN tends to outperform Naive Bayes in difficult classification tasks(Rathore and Kumar 2020).

Our study's data supports the general view that ANN, as opposed to Naive Bayes, is a more successful method for software defect prediction and ought to be used in software engineering procedures. The quantity and caliber of the dataset, the selection of features, and the particular way the machine learning algorithms were implemented are some of the variables that might have had an impact on our findings(Assim, Obeidat, and Hammad, n.d.). Even though our work offers insightful comparisons between the performance of ANN and Naive Bayes, greater investigation into the parameters impacting model performance and validation of our results with bigger and more varied datasets are necessary(Mehta and Patnaik 2021).

We should be aware of the various limitations of our study. First off, the small sample size of the dataset we used for our research may have limited how broadly applicable our conclusions can be. Furthermore, the outcomes could have been affected by the features used and the particular way the machine learning algorithms were implemented. Furthermore, we ignored alternative machine learning methods that might be pertinent for software defect prediction in favor of comparing ANN and Naive Bayes alone in our study.

We should be aware of the various limitations of our study. First off, the small sample size of the dataset we used for our research may have limited how broadly applicable our conclusions can be. Furthermore, the outcomes could have been affected by the features used and the particular way the machine learning algorithms were implemented. Furthermore, we ignored alternative machine learning methods that might be pertinent for software defect prediction in favor of comparing ANN and Naive Bayes alone in our study.

CONCLUSION:-

It is concluded that the ANN model exhibited significantly higher accuracy in predicting software faults compared to the Naive Bayes algorithm within the scope of this study, which aimed to compare machine learning-based software fault prediction using Artificial Neural Network (ANN) with Naive Bayes for improved accuracy. In particular, the ANN model outperformed the Naive Bayes model, which had an accuracy rate of 62.5%, with an accuracy rate of 84.1%. This result implies that the ANN technique has potential to improve the accuracy of software defect prediction in real-world software engineering applications.

DECLARATIONS

Conflict of interests

There is no conflict of interest in this work.

Authors Contribution

Writing the manuscript and handling data analysis and collecting fell to author YNR. Author EKS was responsible for the conceptualization, data validation, and critical assessment of the text.

Acknowledgements

The Saveetha School of Engineering, Saveetha Institute of Medical and Technical Science (formerly Saveetha University) is acknowledged by the authors for giving the support required to successfully finish this work

.

Funding

We acknowledge the following organizations for their financial support, which allowed us to finish this study.

1. INautix Technologies, India.
2. SIMATS Engineering.
3. SIMATS University.
4. SIMATS Institute of Medical and Technical Sciences.

REFERENCES:-

Rathore, S.S., Kumar, S. An empirical study of ensemble techniques for software fault prediction. *Appl Intell* 51, 3615–3644 (2021). https://doi.org/10.1007/s10489-020-01935-6

Alsaeedi, Abdullah, and Mohammad Zubair Khan. 2019. “Software Defect Prediction Using Supervised Machine Learning and Ensemble Techniques: A Comparative Study.” *Journal of Software Engineering and Applications* 12 (5): 85–100.

Assim, Marwa, Qasem Obeidat, and Mustafa Hammad. n.d. “Software Defects Prediction Using Machine Learning Algorithms.” Accessed March 18, 2024. https://ieeexplore.ieee.org/abstract/document/9325677/.

Bahaweres, Rizal Broer, Fajar Agustian, Irman Hermadi, Arif Imam Suroso, and Yandra Arkeman. n.d. “Software Defect Prediction Using Neural Network Based SMOTE.” Accessed March 18, 2024. https://ieeexplore.ieee.org/abstract/document/9251874/.

Cetiner, Murat, and Ozgur Koray Sahingoz. n.d. “A Comparative Analysis for Machine Learning Based Software Defect Prediction Systems.” Accessed March 18, 2024. https://ieeexplore.ieee.org/abstract/document/9225352/.

Delphine Immaculate, S., M. Farida Begam, and M. Floramary. n.d. “Software Bug Prediction Using Supervised Machine Learning Algorithms.” Accessed March 18, 2024. https://ieeexplore.ieee.org/abstract/document/8816965/.

“Iterated Feature Selection Algorithms with Layered Recurrent Neural Network for Software Fault Prediction.” 2019. *Expert Systems with Applications* 122 (May): 27–42.

Jayanthi, R., and Lilly Florence. 2018. “Software Defect Prediction Techniques Using Metrics Based on Neural Network Classifier.” *Cluster Computing* 22 (1): 77–88.

Kumar, Rudra, K. S. N. Prasad, and Annaluri Sreenivasa Rao. 2018. *Defect Prediction in Software Development & Maintainence*. Partridge Publishing.

Kumar, Sandeep, and Santosh Singh Rathore. 2018. *Software Fault Prediction: A Road Map*. Springer.

Mehta, Sweta, and K. Sridhar Patnaik. 2021. “Improved Prediction of Software Defects Using Ensemble Machine Learning Techniques.” *Neural Computing & Applications* 33 (16): 10551–62.

Rathore, Santosh S., and Sandeep Kumar. 2020. “An Empirical Study of Ensemble Techniques for Software Fault Prediction.” *Applied Intelligence* 51 (6): 3615–44.

“Website.” n.d. https://www.researchgate.net/profile/Shabib-Aftab-2/publication/333513059_Performance_Analysis_of_Machine_Learning_Techniques_on_Software_Defect_Prediction_using_NASA_Datasets/links/5d04e2e5299bf12e7be0c614/Performance-Analysis-of-Machine-Learning-Techniques-on-Software-Defect-Prediction-using-NASA-Datasets.pdf.

Mehta, S., Patnaik, K.S. Improved prediction of software defects using ensemble machine learning techniques. *Neural Comput & Applic* 33, 10551–10562 (2021). https://doi.org/10.1007/s00521-021-05811-3

TABLES AND FIGURES

Table 1. Pseudo code for Artificial neural network. The algorithm takes the dataset of software faults and helps to predict faults in it by using this algorithm.

Input: Software Faults prediction dataset
Output: Better Accuracy for Software Faults Prediction
Step 1: Get Started Step 2: Bring in the Artificial Neural Network Requirements Library. Step 3: Open a CSV file and load a dataset. Step 4: Preprocesses the data, encoding categorical features in a single step. Step 5: Split the dataset into training and testing subsets. Step 6: Train the artificial neural network Step 7: Use the artificial neural network to make predictions. Step 8: Assess the accuracy of the model's performance. Step 9: Lastly, subplots are created to show the two models side by side. Step 10: Conclude

Table 2. Pseudo code for Naive Bayes. This algorithm takes the dataset of software faults and helps to predict it by using this algorithm.

Input: Software Faults prediction dataset
Output: Better Accuracy for Software Faults prediction
Step 1: Get Started Step 2: Bring in the Naive Bayes Requirements Library. Step 3: Open a CSV file and load a dataset. Step 4: Preprocesses the data, encoding categorical features in a single step. Step 5: Split the dataset into training and testing subsets. Step 6: Train the Naive Bayes Step 7: Use the Naive Bayes to make predictions. Step 8: Assess the accuracy of the model's performance. Step 9: Lastly, subplots are created to show the two models side by side. Step 10: Conclude

Table 3. Improved accuracy of software fault prediction (Artificial Neural Network's accuracy of 84.10% and Naive Bayes accuracy of 62.50%)

Iteration Number	ANN Accuracy (%)	NB Accuracy (%)
1	84.30	62.90
2	84.40	62.70
3	83.00	62.20
4	83.20	61.60
5	83.50	61.20
6	83.10	60.00
7	82.60	60.90
8	82.20	60.40
9	81.70	62.30
10	81.40	61.80

Table 4. T-test with independent samples The confidence interval for the dataset was set to 95% while the level of significance was set to 5% (p>0.05) (with a p=0.151 value, Artificial neural networks performs better than Naive bayes)

		Levene's Test for Equality of Variances		**T-test for Equality of Means**						
		F	**sig**	**t**	**df**	**sig(2-tailed)**	**Mean difference**	**Std.Error Difference**	**95% Confidence Interval of the difference**	
									Lower	**Upper**
Accuracy	Equal Variances Assumed	2.250	0.151	36.57	18	.001	21.6000	0.59067	20.359	22.8409
	Equal variances not Assumed			36.57	16.22	.001	21.6000	0.59067	20.349	22.8508

Table 5. In a group statistical investigation, Artificial neural network and Naive Bayes were used. After 10 iterations, the mean accuracy value, standard deviation, and standard error mean for the Artificial neural network and Naive Bayes methods are obtained. The Artificial neural network approach outperformed the Naive Bayes method, according to the results.

Group	N	Mean	Standard Deviation	Standard Error Mean
ANN	10	84.10	1.52388	0.48189
NB	10	62.50	1.08012	0.34157

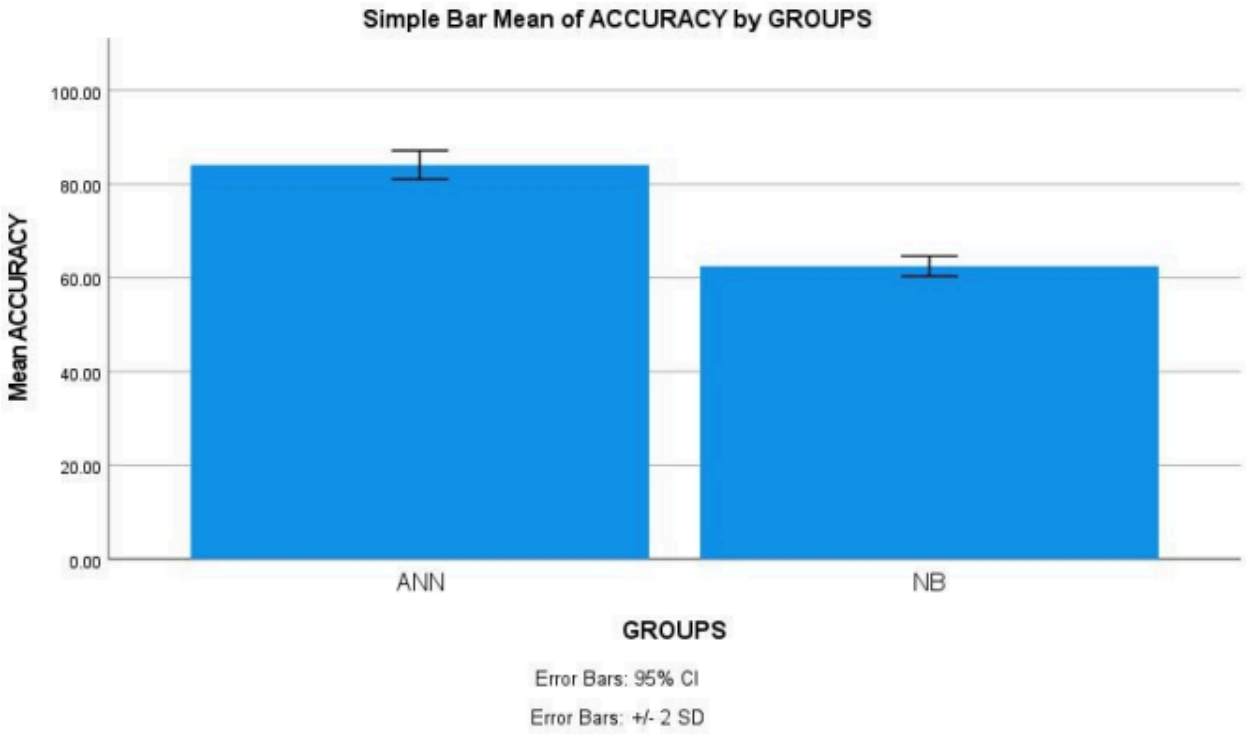

Fig.1 In terms of mean accuracy, Artificial neural network and Naive Bayes are compared. The mean accuracy of the Artificial neural network is higher than the Naive Bayes, while the standard deviation is slightly lower than the methods. X-Axis: Artificial neural network vs. Naive Bayes model; Y-Axis: Mean detection accuracy within +/-2 SD of 95% interval