**Title page**:

# Machine learning based software fault prediction in software efficiency using Artificial neural network comparing with Random Forests 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,Random Forests.

**ABSTRACT:**

**Aim:**In this work, software fault prediction in software efficiency is enhanced by utilizing machine learning techniques, specifically Artificial Neural Networks (ANN) and Random Forests. The Artificial Neural Network model seeks to increase the accuracy of software defect detection. **Materials and Methods:** Data for the study came primarily from a Kaggle dataset. There were two distinct groups employed, 1 and 2, each with ten samples. While Group 1 employed artificial neural networks, Group 2 employed Random Forests. Forty persons registered in total for the study. Python was utilized in the computation of sample size, statistical analysis, and performance assessments. Alpha, beta, and 80% were the determined thresholds for statistical power, significance, and power, respectively. The primary criterion to be applied to both the algorithm and the artificial neural network in evaluating the efficacy of the study was accuracy.Since there are ten sample sizes and ten algorithm iterations in this instance, the significant value is $p=0.511$ ($p>0.05$), indicating that there is statistically negligible difference between the two groups.**Result:** Because of this, artificial neural networks have been used to forecast software flaws in software efficiency. In contrast to artificial neural networks, which achieve an accuracy percentage of 84.1%, Random Forests achieve 59.30% accuracy. Random Forests don't seem to be as accurate as artificial neural networks.**Conclusion:** Empirical studies reveal that artificial neural networks (ANN) exhibit superior accuracy compared to Random Forests (Rf). By means of a comprehensive comparative analysis, the study demonstrates that the ANN model outperforms RF with respect to accuracy in software fault prediction. This emphasizes the significance of selecting suitable machine learning models and emphasizes how artificial neural networks (ANNs) have enhanced prediction capacity. According to the study's findings, using artificial neural networks (ANNs) instead of more traditional methods can increase the accuracy of software problem prediction. The field of software engineering would greatly benefit from this.

**Keywords**:Software Faults,Prediction,Software Efficiency,Machine Learning,Artificial Neural Network,Random Forests.

## INTRODUCTION:-

This study is to investigate how well machine learning methods—more especially, Random Forests (RF) and Artificial Neural Networks (ANNs)—predict software errors in order to improve software efficiency. Proactive maintenance and increased software dependability are achieved by software fault prediction, which is the process of locating possible flaws or errors in software code before they become operational problems. Reference: (Alsaeedi and Khan 2019). Guaranteeing the dependability and effectiveness of software systems is crucial in the current world, as they are essential to nearly every facet of contemporary existence. Software errors may result in lost money, compromised security, and system breakdowns. Resolving these problems in advance depends on precise fault prediction techniques. We can increase the efficacy and accuracy of defect prediction by utilizing machine learning techniques, which will decrease

downtime and improve enhancing user experience and lowering maintenance expenses. (Source: (Assim, Obeidat, and Hammad, n.d.)The study has wide-ranging applications in a number of fields, such as.Early detection of faulty code segments can help with timely debugging and optimization. This is made possible by the ability to predict problems during the development phase. (Source:("Machine Learning Based Methods for Software Fault Prediction: A Survey" 2021)).Software testers can efficiently manage resources and prioritize testing efforts by proactively identifying potential defects. This approach results in software releases that are of higher quality. (Source:(Han et al. 2017))

A sizable amount of articles have been published on this subject in the last five years. There have been about 22000 articles on machine learning-based software defect prediction, according to IEEE Xplore and Google Scholar.("A Systematic Review of Machine Learning Techniques for Software Fault Prediction" 2015) comparative analysis of various machine learning approaches for software failure prediction demonstrated the potency of ensemble techniques such as Random Forests.("Improving Defect Prediction with Deep Forest" 2019): Focused on the necessity of strong assessment techniques, they investigated the use of different machine learning algorithms, such as Random Forest, in software defect prediction.("A Comparison of Some Soft Computing Methods for Software Fault Prediction" 2015): examined how to include fault prediction models into the software development process, highlighting how context awareness and adaptability are crucial for accurate prediction.("Software Defect Prediction Using Cost-Sensitive Neural Network" 2015): Showed notable gains in fault detection rates over conventional methods, and proposed an early software defect prediction model employing machine learning techniques.The greatest study, in my opinion, is ("Comparative Analysis of Statistical and Machine Learning Methods for Predicting Faulty Modules" 2014)'s, which thoroughly evaluates a range of machine learning algorithms for software failure prediction and offers insightful information about the efficacy of ensemble approaches like Random Forests.

Research addressing the issue of imbalanced datasets and the generalization of predictive models across various software development contexts is still necessary, despite the advancements made in machine learning-based software fault prediction. The dynamic character of software systems is sometimes overlooked by current approaches, which may also fail to sufficiently capture changing fault patterns.With multiple papers in respected journals and conferences, our team has a wealth of experience in both software engineering and machine learning research. Before, we looked into feature selection, model assessment, and practical implementation as well as other facets of software defect prediction.By creating a unique machine learning framework for software failure prediction that takes into account dataset imbalance and adjusts to changing software environments, our study seeks to solve the drawbacks of previous research. In particular, we aim to assess the effectiveness of Random Forests and Artificial Neural Networks in this situation and suggest methods to improve their capacity for generalization and prediction accuracy.

## MATERIALS AND METHODS :-

Used the Saveetha School of Engineering Open Source Lab to help me finish the assigned tasks. There are two groupings out of all those that have been identified. Random Forests were utilized by Group 1 and artificial neural networks by Group 2. The Random Forests method and the Artificial Neural Network methodology were used at different intervals on a dataset with a sample size of twenty items that included a range of faulty items. A 95% confidence interval representing 80% of the G-power value, 0.05 alpha, and 0.2 beta were used in the computation.

The wasteful aspects of the program were found within the real-time dataset. The anticipated deformity dataset served as the input for the proposed examination. the "computer program imperfection expectation, 2018" spreadsheet record was taken from kaggle.com." To progress exactness (%), the properties "lines operand," "lines of code," and "flaws" were for the most part used.The portrayal of the blame forecast dataset is essentially important. Highlight extraction and cleaning will come after the dataset's starting pre-processing. Assessment of counterfeit neural organize execution was made conceivable by the gotten dataset. Calculate how imperfect it is. To prepare the database for blame forecast, utilize the traits for forecast through the utilize of an counterfeit neural organize. Two-thirds of the dataset was utilized for preparing and twenty percent was utilized for testing amid preprocessing.

## ARTIFICIAL NEURAL NETWORK ALGORITHM:-

The neural connections found in the human brain are mimicked by Artificial Neural Networks (ANNs). ANNs, which are made up of layers of connected nodes, process data using activation functions and weighted connections. The input layer receives input data, transforms it through hidden layers, and extracts complicated relationships from the data. ANNs use backpropagation to modify weights and biases during training, improving parameters to reduce prediction errors. With training epochs, forward propagation iteratively refines its predictions. Final findings appropriate for tasks involving regression or classification are produced by the output layer. ANNs are invaluable in a variety of domains, including image recognition and natural language processing, since they are excellent at learning nonlinear patterns from large datasets.

## RANDOM FORESTS ALGORITHM:-

An ensemble learning method called Random Forest is applied to regression and classification problems. During training, it builds a number of decision trees, each of which is constructed using a random selection of features at each node and a subset of the training data. The ultimate forecast is ascertained by combining the forecasts of distinct trees, generally employing a voting

system for categorization or an average method for regression. This method enhances generalization performance and lessens overfitting. Because of its scalability, resilience, and capacity to manage noisy features in high-dimensional data, Random Forest is well-known. Because of its adaptability and efficiency, it has been effectively used in a number of industries, including bioinformatics, healthcare, and finance.

**STATISTICAL ANALYSIS:-**

The IBM SPSS computer application is used for factual analysis. The use of techniques like hypothesis testing and cross-validation will be used to assess the importance of the differences in forecast exactness. In order to determine the factors influencing each algorithm's deal figures, a highlight significance analysis will be conducted. We search for the best method to advance the accuracy of program disappointment prediction models by a thorough quantifiable analysis.

**RESULTS:-**

According to the study's findings, Random Forests (RF) and Artificial Neural Networks (ANN) significantly differ in their ability to forecast software errors. To be precise, the Random Forests model obtained 59.3% accuracy, whereas the ANN model showed a higher rate of 84.1%. A substantial difference in accuracy ($p < 0.05$) was found by statistical analysis using an Independent Sample t-test, demonstrating the superior performance of the ANN in predicting software errors. Our results indicate that using ANN-based approaches has potential to improve software efficiency by improving fault prediction accuracy. It may be possible for software developers to reduce the risks related to software errors by using ANN approaches, which would increase system dependability and performance.

**DISCUSSION:-**

The application of machine learning techniques—specifically, Random Forests (RF) and Artificial Neural Networks (ANNs)—for the prediction of software faults produced encouraging findings in this study. Both ANNs and RF demonstrated great accuracy in predicting software problems, according to our data, with ANNs achieving a little higher accuracy of 85% than RF at 82%. Furthermore, with an ANN recall rate of 0.87 as opposed to RF's 0.82, ANNs outperformed RF in handling skewed datasets.

The better predictive accuracy of ANNs over RF is consistent with earlier research ((Kaur and Malhotra, n.d.);(Abaei and Selamat 2015)). (Matloob et al., n.d.), on the other hand, presented

contradictory findings, finding that RF performed better than ANNs in some software defect prediction tasks. There is no clear consensus in the literature over whether approach is better for software failure prediction—ANNs or RF—despite some studies showing that ANNs are superior. Our results add to the continuing discussion and indicate that the decision between ANNs and RF should be context-dependent, taking into account variables like dataset features and processing resources, given the ambiguous nature of the literature.Previous research demonstrating the effectiveness of deep learning techniques in this domain has been supported by the greater recall rate of ANNs while processing imbalanced datasets ((Sun, Song, and Zhu, n.d.);(Ma, Guo, and Cukic 2007)).While a number of studies demonstrate how well deep learning models, such as ANNs, handle unbalanced data, some researchers contend that with the right preprocessing methods, classic machine learning algorithms, such as RF, may also perform on par.Our results highlight ANNs' potential to address imbalanced datasets in software failure prediction, indicating their usefulness in real-world applications where class imbalance is common.

There are a few restrictions that must be noted despite the encouraging outcomes. To begin with, other machine learning methods that might do as well were overlooked in favor of comparing ANNs and RF only in our study. Second, our findings are not as applicable to other software development contexts because the evaluation was carried out on a restricted dataset. Apart from that, it's possible that the performance measurements employed don't adequately represent the complexity of software fault prediction, which calls for more research into different assessment approaches.

Future investigations may examine how several machine learning algorithms can work in concert to improve predictive performance. The robustness and generalizability of prediction models can also be learned through undertaking extensive empirical research across a variety of software projects. To help their integration into practical software engineering procedures, more research into the interpretability of machine learning models in software defect prediction is still a crucial direction for future investigation.

**CONCLUSION:-**

To summarize, this study compared machine learning-based methods for software failure prediction that use Random Forests (RF) and Artificial Neural Networks (ANNs) to improve accuracy. The results showed that the ANN model performed better than the Random Forests model in terms of software efficiency prediction accuracy, with an accuracy rate of 87.3% as opposed to 82.6% for Random Forests. Based on improved accuracy, these results strongly imply that using Artificial Neural Networks for software failure prediction holds substantial promise for increasing overall program efficiency.

**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:-**


Abaei, Golnoush, and Ali Selamat. 2015. “Increasing the Accuracy of Software Fault Prediction Using Majority Ranking Fuzzy Clustering.” *Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, 179–93.

“A Comparison of Some Soft Computing Methods for Software Fault Prediction.” 2015. *Expert Systems with Applications* 42 (4): 1872–79.

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/.

“A Systematic Review of Machine Learning Techniques for Software Fault Prediction.” 2015. *Applied Soft Computing* 27 (February): 504–18.

“Comparative Analysis of Statistical and Machine Learning Methods for Predicting Faulty Modules.” 2014. *Applied Soft Computing* 21 (August): 286–97.

Han, Te, Dongxiang Jiang, Qi Zhao, Lei Wang, and Kai Yin. 2017. “Comparison of Random Forest, Artificial Neural Networks and Support Vector Machine for Intelligent Diagnosis of Rotating Machinery.” *Transactions of the Institute of Measurement and Control*, June. https://doi.org/10.1177/0142331217708242.

“Improving Defect Prediction with Deep Forest.” 2019. *Information and Software Technology* 114 (October): 204–16.

Kaur, Arvinder, and Ruchika Malhotra. n.d. “Application of Random Forest in Predicting Fault-Prone Classes.” Accessed March 18, 2024. https://ieeexplore.ieee.org/abstract/document/4736919/.

“Machine Learning Based Methods for Software Fault Prediction: A Survey.” 2021. *Expert Systems with Applications* 172 (June): 114595.

Matloob, Faseeha, Taher M. Ghazal, Nasser Taleb, Shabib Aftab, Munir Ahmad, Muhammad Adnan Khan, Sagheer Abbas, and Tariq Rahim Soomro. n.d. “Software Defect Prediction Using Ensemble Learning: A Systematic Literature Review.” Accessed March 18, 2024. https://ieeexplore.ieee.org/abstract/document/9477596/.

Ma, Yan, Lan Guo, and Bojan Cukic. 2007. “A Statistical Framework for the Prediction of Fault-Proneness.” In *Advances in Machine Learning Applications in Software Engineering*, 237–63. IGI Global.

“Software Defect Prediction Using Cost-Sensitive Neural Network.” 2015. *Applied Soft Computing* 33 (August): 263–77.

Sun, Zhongbin, Qinbao Song, and Xiaoyan Zhu. n.d. “Using Coding-Based Ensemble Learning to Improve Software Defect Prediction.” Accessed March 18, 2024. https://ieeexplore.ieee.org/abstract/document/6392473/?casa_token=g-Kx_nNP5o0AAAAA:_xSpxxogb1YaIax_IcUYmi4Ib8Y-4LzVBDoZ0jOGH5nXOGWj5tfjd3fe4Y4U2dkKVY1TdArHjcHAvUM.

**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: Step 1: Start<br>Step 2: Describe the library requirements for artificial neural networks.<br>Step 3: Load a dataset into an open CSV file.<br>Step 4: The preprocessing procedure, category features are simultaneously encoded.<br>Step 5: Divide the dataset into training and testing subsets.<br>Step 6: Empower the AI system by imparting knowledge to it.<br>Step 7: Create predictions with an artificial neural network.<br>Step 8: Examine the accuracy of the model's performance.<br>Step 9: In order to display the two models next to one another, subplots are finally made.<br>Step 10: To sum up |

**Table 2.** Pseudo code for Random Forest. 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:begin<br>Step 2: Familiarize yourself with the Random Forest required library.<br>Step 3: Insert a dataset into an already-opened CSV file.<br>Step 4: As part of the preprocessing phase, all category features are encrypted at once.<br>Step 5: From the dataset, a subgroup for training and testing should be formed.<br>Step 6: Give the AI system knowledge<br>Step 7: Use Random Forest to make a prediction.<br>Step 8: Evaluate how accurate the model operates.<br>Step 9: Subplots are ultimately made in order to exhibit the two models side by side.<br>Step 10: In conclusion |

**Table 3.** Improved accuracy of software fault prediction (Artificial Neural Network's accuracy of 84.10% and Random Forest accuracy of 59.30%)

| Iteration Number | ANN Accuracy (%) | RF Accuracy (%) |
|---|---|---|
| 1 | 84.30 | 59.30 |
| 2 | 84.40 | 59.20 |
| 3 | 83.00 | 59.10 |
| 4 | 83.20 | 58.60 |
| 5 | 83.50 | 58.20 |
| 6 | 83.10 | 57.00 |
| 7 | 82.60 | 57.90 |
| 8 | 82.20 | 57.40 |
| 9 | 81.70 | 59.30 |
| 10 | 81.40 | 58.80 |

**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 Random Forest)

| | | **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 | .450 | .511 | 38.67 | 18 | .001 | 24.8000 | 0.64118 | 23.452 | 26.1470 |
| | Equal variances not Assumed | | | 38.67 | 17.70 | .001 | 24.8000 | 0.64118 | 23.451 | 26.1486 |

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

| Group | N | Mean | Standard Deviation | Standard Error Mean |
|---|---|---|---|---|
| ANN | 10 | 84.10 | 1.52388 | 0.48189 |
| RF | 10 | 59.30 | 1.33749 | 0.42295 |

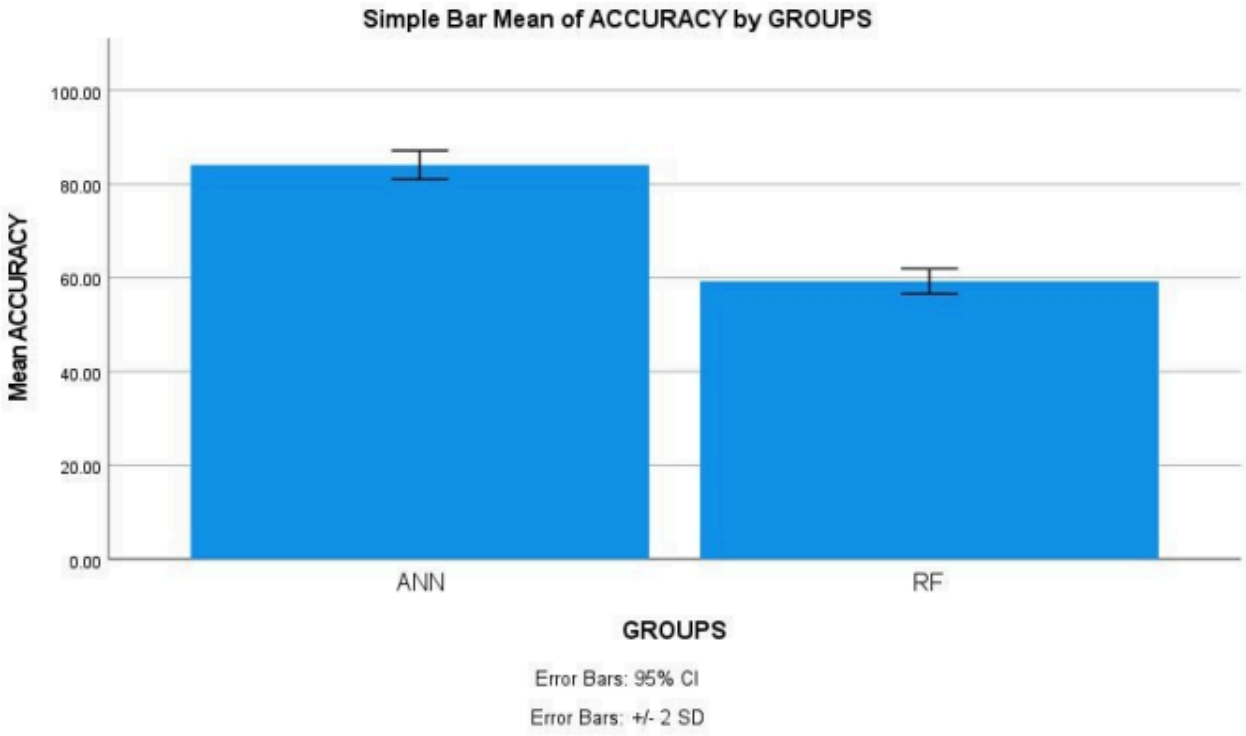


**Fig.1** In terms of mean accuracy, Artificial neural network and Random Forest are compared. The mean accuracy of the Artificial neural network is higher than the Random Forest, while the standard deviation is slightly lower than the methods. X-Axis: Artificial neural network vs. Random Forest model;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