International Journal For Advanced Research In Science & Technology
A peer reviewed international journal www.ijarst.in
ISSN: 2457-0362

SOFTWARE DEFECT PREDICTION USING MACHINE LEARNING

Kuncham Ramya [1], Kadiyala Siva Rama Krishna[2], Chembeti Mahendra [3], Chittela Manohar [4], Kola Thirupalu [5], Arnepalli Madhu[6]
[1] Asst.Professor,CSE(Artificial Intelligence) Department,ABRCET,Kanigiri, Andhra Pradesh, India.
[2,3,4,5,6] B.Tech Student, CSE(Artificial Intelligence) Department, ABRCET, Kanigiri, Andhra Pradesh, India.

ABSTRACT

The end users who are using the software and its products is vastly increased when comparedto the earlier days. As we are seeing that the technology has evolved a lot and it has deliveredan extraordinary technology named artificial intelligence. Identifying defects in a software in the current time can be held with Software Development Life Cycle (SDLC) and it stays a fundamental and crucial task. In the present days, a few instances of flawed and non-defective modules are used to construct prediction models which utilize machine learning techniques.
To address the software modules, software metrics were used as input to these machine learning algorithms. In order to detect the defects in a software, few powerful machine learning techniques are implemented and in existing system the algorithm named Random Forest (RF)gives an adequate accuracy.
But we need to identify the defects in a software using machine learning methods with better model which must give some improved accuracy when compared with RF.
So here in this paper we are using extra trees classifier and hybrid model to identify the defectsin a software.

Introduction:

To improve the reliability of software few measuring techniques are used. Software Defect Prediction (SDP) has a lot of test experience in computer science for finding flaws. Mostly in the present scenario, the curiosity amongst the individuals has increased a lot because majority of the devices contains software programs which have become important to its customers because it includes appealing features, and buyers want to access them without having to learn anything. Yet, the focus of interest was that it evolved into a communal requirement whereby individuals can connect and share knowledge. In the recent decade, people have been focusing on application frames, where performance enhancement is seen as the most important aspect of client functioning. Software performance has remained a perplexing subject, yielding insufficient outcomes for commercial and facilities services, owing to the clear tremendous growth of utilization. Throughout the growth phase, firms frequently use fault diagnostic patterns and similar methodologies to help in anticipating defects, estimating effort, assessing software dependability,risk assessment, and so on. With a given huge dataset, a controlled machine learning prediction computation is employed.

Following that, the algorithm learns from the training sample and develops instructions for predicting the class name for a new data set. Using math equations to construct and improve the indicator work is one of the learning stages. There is a certain intake esteem and a specific yield esteem in that interaction training set. A generally known result is used to assess the correctness ofa typical Machine Learning (ML) calculation.

Some concepts may be to establish a gathering of associates at a specified location for casual contact amongst data users. Software quality may be enhanced by predicting failure areas. Defect detection is a method of generating models that are used early in the contact process to identify problematic frames such as units or categories. It is accomplished by classifying components as defect-prone or not. Severalmethods have been used to recognize the defective modules, the most well-known of which are Cat Boost, XG Boost, SVM, Light Gradient Boosting, RF classifier and Hybrid algorithm. Each classifying item, also known as the connection between characteristics and the training dataset class mark, is placed downon the classifier technique and analyzed using the target order equations. Such parameters will also be used to choose the names of future data classes. These complicated data may be classified in this way byusing categorization algorithms and classifiers. Identifying software problems, discovering the defect, and acknowledging it is a difficult work for specialists. The main purpose is to divide the software data into defective and non-defective datasets as a paradigm for identifying issues. The input software dataset is supplied to the classifier in this method, and the client knows the real class values. Prior to this graph, metric strategies centered on need and setup yielded long-term outcomes. Regardless, the design of methodologies and the accuracy of forecasts remain a challenge that must be addressed. Software defectprediction aims to forecast defect-prone software systems by utilizing a few essential elements of the software project. It's usually done by creating forecasting models for known projects using project attributes reinforced with defective data, and then using these forecasting models to anticipate defects forunknown projects. SDP is based on the idea that if a project is created in a defect-prone environment, every module created in a similar environment with comparable job characteristics would be troublesomeas well. The purpose of software defect prediction is to anticipate defect-prone software modules based on a set of baseline software project characteristics. Creating a classification algorithm for a known project utilizing project attributes supplemented with incorrect information, and then applying the prediction system to new projects to foresee issues, is a common method. If a program generated in a given environment causes flaws, then every component created in a restricted sequence with identical project parts would produce errors as well. The RF ensemble technique performs well, but our major aim is to improve the accuracy gained and more precisely detect the defective software. As a result, we devised a method that uses a hybrid algorithm and an extra trees classifier to provide more accuracy thanprevious ensemble methods.

Literature Survey:

1. Individual Classifier for Software Defect Prediction:

Software Defect Prediction via Attention-Based Recurrent Neural Network: Individuals' use of software has expanded dramatically in recent years as compared to previous years. Artificial intelligence has evolved in tandem with the rapid advancement of technology.

Software Defect Prediction (SDP) remains a basic and crucial function in the underlying period of the Software Development Life Cycle (SDLC). A lot of experiments have been going on in the last several days to detect the program's quality, which leads to giving the software a guaranteed quality. SDP predicts thelikelihood of software flaws at the start of the software development process, making it easier to identify and address them, as well as reducing issues that may arise later. This will help with the software's overall nature. In recent years, a number of machine learning algorithms (ML) have built prediction models using examples of defective and non-defective modules. In order to address the software modules, software metrics were used as input to these ML algorithms. ML techniques such as Cat Boost, XG Boost (Extreme Gradient Boosting), Support Vector Machine (SVM), Light Gradient Boosting Machine (Light GBM), and Random Forest are used in this research to discover software faults (RF). These analyses are carried out on three independent software defect datasets. We use the SMOTE technique to address this difficulty in most software defect datasets with imbalanced output results (count of defect and not defect too much fluctuate). We model each dataset separately and fine- tune the hyper parameters for each dataset to achieve the best accuracy. Cat Boost Classifier is the bestmethod for software defect datasets, according to our research. It provides the greatest accuracy score for two of the three datasets. For the remaining dataset, the Random Forest Classifier has the highest accuracy. The algorithm with the lowest accuracy for all three datasets is Support Vector Machine. In this study, we will go over our dataset, pre-processing approaches, modelling, assessment, and model comparison in greater detail, as well as compare our work to that of other authors.

2. Within- vs. Cross-project Software Defect Prediction:

In this method there using Cat Boost, XG Boost (Extreme Gradient Boosting), Support Vector Machine (SVM), Light Gradient Boosting Machine (Light GBM) it will take lot of time to get the good accuracy. Its complex to write the code for each Algorithm. In Our method using python machine learning techniques and using different algorithms and Hybrid algorithm to predict the Software quality.

Existing System:

♦ In the existing system, implementation of machine learning algorithms is bit complex to build dueto the lack of information about the data visualization.

♦ Mathematical calculations are used in existing system for model building this may takes the lot oftime and complexity.

♦ To overcome all this, we use machine learning packages available in the Scikit-learn library likeboosting techniques.

Drawbacks for Existing System:

- In this existing method to using machine learning algorithms of software defect prediction itwill work like as a High complexity.

- The visualization of using techniques and Packages is Library and boosting it will come asTime consuming.

Proposed System for Software Defect Prediction:

➢ Proposed several machine learning models to classify the software detection, but none haveadequately addressed this misdiagnosis problem.

➢ Also, similar studies that have proposed models for evaluation of such accident severity mostlydo not consider the heterogeneity and the size of the data.

➢ Therefore, we propose a xgboost, catboost, extra tree classifier and performing classifier testsbased.

Block Diagram for Proposed System:

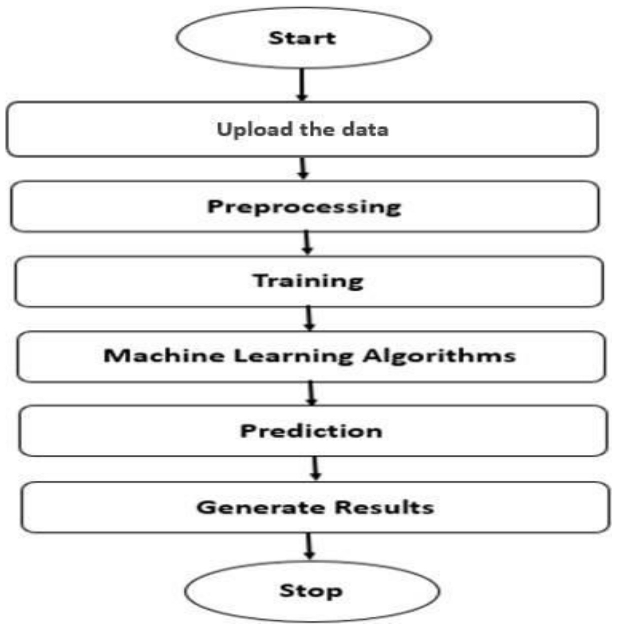
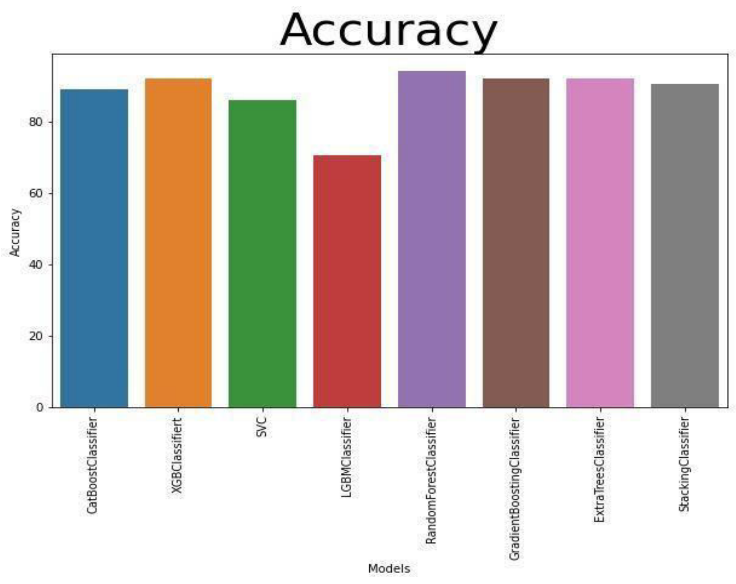


Figure.1: Block Diagram of Software Defect Prediction**.**

RESULTS

Everyone can't perform very complex statistical calculations with accuracy making statistical analysis a time-consuming and costly process. Statistical software has become a very important tool for companies to perform their data analysis. The software uses Artificial Intelligence and Machine Learning to perform complex calculations, identify trends and patterns, and create charts, graphs, and tables accurately within minutes. In Statistical we plot the each algorithm got different accuracy value so show the plot in the build the bar plot in the shoes below figure



CONCLUSION

Software faults can degrade the quality of software, causing problems for both customers and developers. As software designs and technology have become more intricate, manual programme identification has become a challenging and time-consuming task. As a result, autonomous software detection has been a hotspot for industrial research in recent years. The purpose of this research is to use data-mining techniques to predict software flaws. Furthermore, this problem has grown in importance as a research area, with numerous approaches being explored to increase the effectiveness of detecting software flaws or anticipating defects in some way.

In this study, we use machine learning to try to solve the problem. Using three datasets from the NASA Promise dataset repository, we examine the performance of state-of-the-art machine learning approaches. We develop predictions using a variety of algorithms and can detect poor software. This is accomplished in a user-friendly environment using Python programming and machine learning techniques such as Cat Boost, Gradient Boosting, Light GBM, Random Forest, and Ml Xtend to construct a hybrid model and extra trees classifier, both of which outperform with higher accuracy.

In Hybrid Model: All three Accuracy, Precision, and Recall had maximum equal values.These three indicators indicate how well our model performs.

Accuracy: The accuracy of a machine learning model is a metric for determining which model is the best at recognizing relationships and patterns between variables in a dataset based on theinput, or training, data.

Precision: The accuracy of the model's positive forecast. The number of true positives dividedby the total number of positive predictions is known as precision (i.e., the number of true positives plus the number of false positives).

Recall: Recall is a metric that measures how many correct positive predictions were made out of all possible positive predictions. Unlike accuracy, which simply comments on the accurate positive predictions out of all positive predictions, recall shows where positive predictions weremissed.

If we have the same values, it suggests that our prediction is mainly correct.

REFERENCES

[1].G. Fan, X. Diao, H. Yu, K. Yang, and L. Chen, Software Defect Prediction via Attention-Based Recurrent Neural Network, Hindawi, 2019.
[2].E. A. Felix, and S. P. Lee, Predicting the number of defects in a new software version, PLOS ONE (2020).
[3].Z. Xu, J. Liu, X. Luo, Z. Yang, Y. Zhang, P. Yuan, Y. Tang, and T. Zhang, Inform. Softw.

Technol. **106**, 182–200 (2019).

[4].Z. Xu, J. Xuan, J. Liu, and X. Cui, Cross project defect prediction via balanced distributionadaptation based transfer learning. Journal of Computer Science and Technology 34(5), 1039–1062 (2019).
[5].R. Malhotra, and S. Kamal, Neurocomputing 343, 120–140 (2019).

[6].R. Malhotra, and S. Kamal, Tool to handle imbalancing problem in software defect prediction using oversampling methods. In: 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), IEEE, 2017, pp. 906–912.
[7].R. Malhotra, and J. Jain, Int. J. Softw. Eng. Knowl. Eng. 31, 193–215 (2021).

[8].D. Ryu, J. Baik, Effective multi-objective naïve Bayes learning for cross-project defect prediction. Appl. Soft Comput. 49, 1062 (2016).
[9].C. Shan, B. Chen, C. Hu, J. Xue, and N. Li, Software defect prediction model based on LLE and SVM. In: Proceedings of the Communications Security Conference (CSC ’14), 2014, pp. 1–5.
[10]. Z. R. Yang, A novel radial basis function neural network for discriminant analysis.

IEEE Trans. Neural Netw. 17(3), 604–612 (2006).

[11]. K. Han, J. -H. Cao, S. -H. Chen, and W. -W. Liu, A software reliability prediction method based on software development process. In: 2013 International Conference on Quality, Reliability, Risk, Maintenance, and Safety Engineering (QR2MSE). IEEE, 2013, pp. 280–283.