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

Enhancing Software Quality using Machine Learning on Open Source Projects

Nitin Kumar[1], Dakshita Joshi[2]
Computer science and Engineering Department,
Bansal Institute of Engineering and Technology, Lucknow
Nitink.srivastava529@gmail.com

Abstract: Software quality and management include software reliability as a crucial and essential component. Different conventional techniques have been adopted by the software industry to discover defects and improve programme reliability. Numerous businesses provide numerous beta versions and other releases that are accessible to the general public. These releases keep track of the various bugs that users report and are also based on user feedback. We'll employ machine learning methods like decision tree models, KNN, linear regression models, ensemble learning with both boosting and bagging strategies. The SPSS 14.0 dataset will be used to train all of the models, and the multidimensional result analysis will be used to determine which model is best for predicting software reliability.

Keywords: ANSI, Machine Learning, Software Defect, SRGM

1. Introduction:

Software that is trustworthy is necessary for business applications that are vital in nature, yet creating such software is a major difficulty that our software industry currently faces. Nowadays, software complexity is rising, making it challenging to achieve software reliability. The main objective of software reliability modelling is to determine the likelihood that a system will fail within a certain time frame or the anticipated time gap between failures.

According to the ANSI definition, software reliability is the likelihood that software will operate without errors for a predetermined amount of time in a predetermined environment. In recent years, modelling software reliability has become much more significant. The importance of software in many modern applications has greatly increased the quantity of work being done in this field. The prediction of software reliability has significantly improved in recent years with the introduction of intelligent neural networks and hybrid techniques in place of the classic statistical methodologies. It is difficult to choose the most intelligent or statistical technique because their performance changes depending on the data. For estimating and projecting the number of software defects still present, SRGM has been used.

Software quality, according to ISO 9126, is the "totality of features and qualities of a software product that have an impact on its capacity to satisfy stated or implied needs."

While ISO 25000 adopts the stance that software solutions must be able to satisfy explicit and implicit needs when utilised in accordance with predetermined guidelines,

"Machine Learning Approach for Quality Assessment and Prediction in Large Software Organizations" is a related piece of work (2). (Miroslaw Staron and Rakesh Rana) [1], Software's importance and complexity, including how to measure, maintain, and improve software quality, are expanding daily. Software metrics offer a quantitative way to assess and subsequently manage various software system properties. To determine and spend resources where they are most required, it is crucial to evaluate software quality early in the development process. When creating software for systems regarded to be mission, commercial, or safety critical, dependability-related qualities become even more crucial. In conjunction with the ISO/IEC 15939 measurement information model, machine learning techniques can be used to model the overall quality of a certain software module, product, or project. a formula or algorithm that combines a number of basic and/or derived measures with the relevant decision criteria. It is predicated on knowledge of, or suppositions regarding, the anticipated relationship between the component measures and/or their evolution through time. Models generate projections or assessments pertinent to specified information requirements. The choice of analysis methods or models used to develop indicators is influenced by the scale and measuring approach. Software quality monitoring, evaluation, and improvement are becoming increasingly crucial as software becomes a more significant part of our daily lives and as its complexity rises.

"State of the Art: Machine Learning-based Software Quality Prediction Models" (Moataz Ahmed and Hamdi A. Al-Jamimi) [2], Quantification of factors influencing software quality and machine learning methods used to forecast software quality Software quality refers to the extent and satisfaction of the needs that consumers have defined. As this subject focuses on creating algorithms, ML techniques have been used to a wide range of problem domains. For predicting the software fault-proneness modules, use the support vector machine (SVM). SVM is being used to forecast maintenance effort. Due to its capacity to incorporate both empirical data and professional judgments, the Bayesian network (BN) has been used in numerous research in the SWE field. The use of neural networks (NN) as a technique for software error prediction. The connection between internal and external factors Different research in the literature have attempted to use the capability of fuzzy logic (FL) to measure the software quality. Quality attributes are surrounded by impression and uncertainty. The two main sources of knowledge are used by contemporary software quality prediction modelling techniques to generate the models. The transparency of the model is a need for the efficient fusion of knowledge sources. The study found that none of the existing models address the problem of model transparency.

2. Related Work:

Bibi S., Tsoumakas G., Stamelos I., Vlahavas I.(2006)[1] involved AI procedure for figuring out the quantity of deformities viz. called Regression through Classification (RvC). A near exploratory investigation of many AI calculations was done to assess this methodology. Pekka Forselious gathered the information, whose applications were kept up with by Finland bank. The structure created gave the possibility to identify shortcomings that were generally not distinguishable.

Norman Fenton et.al.(1999) [2], have portrayed a probabilistic model for programming deformity expectation. The point here is to plan a model which is a blend of different structures that might be frequently relaxed, with accessible proof being developed of programming so the work should be possible in more regular and productive way than it was recently finished. Here a basic survey of various programming measurements and factual models and the cutting edge has been completed. In the greater part of the models utilized for expectation of deformity size and intricacy measurements is utilized.

Other live on the testing information, quality improvement process or potentially a multivariate style is followed. To confirm this methodology, Graphical likelihood models (otherwise called Bayesian Belief Networks) has been utilized. To foster the likelihood model, emotional judgment of experts, project administrators who are capable has been used to anticipate the model for blunder. This has been utilized all through the improvement life pattern of the project.This model can not exclusively be utilized for surveying continuous undertakings, yet in addition for investigating the potential impacts of a scope of programming process improvement exercises. In the event that expenses can be related with process enhancements, and advantages surveyed for the anticipated improvement in programming quality, then the model can be utilized to help trustworthy decision making for SPI (Software Process Improvement).

Ahmet Okutan, et.al.(2012)[3], proposed an original strategy utilizing Bayesian organizations to investigate the connections among programming measurements and imperfection inclination. Nine informational collections from Promise information vault has been utilized and show that RFC, LOC, and LOCQ are more successful on imperfection inclination. Additionally proposition for two additional measurements, for example Gesture for the quantity of designers and LOCQ for the source code quality has been given. At long last eventually, peripheral deformity likelihood of the product, its successful metrices and their connections has been examined.

Mrinal Singh Rawat et. al.(2012)[4], distinguished causative variables which thusly propose the solutions for further develop programming quality and efficiency. They showed how the different deformity expectation models are executed bringing about decreased size of imperfections. They introduced the utilization of different AI procedures for the product shortcoming expectation issue. The unfussiness, ease in model adjustment, client acknowledgment and forecast precision of these quality assessment procedures exhibit its functional and handy attraction. These demonstrating frameworks can be utilized to accomplish convenient issue expectations for programming parts as of now a work in progress, giving significant experiences into their quality. The product quality affirmation group can then use the expectations to utilize accessible assets for acquiring savvy dependability upgrades.

3. Methodology:

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

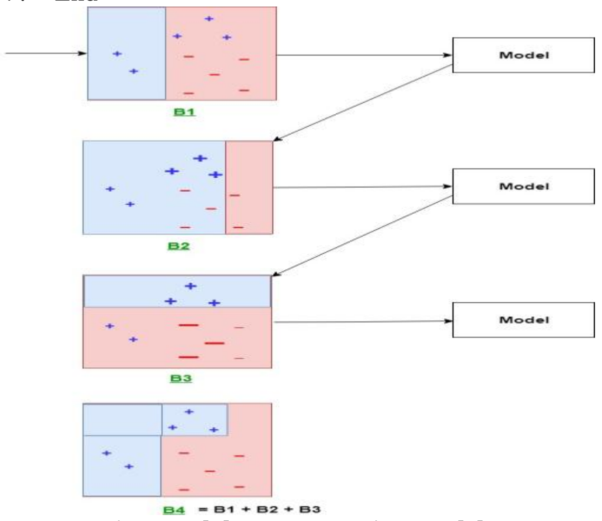

Fig. 1. Adaboost regression model.

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 2. 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.

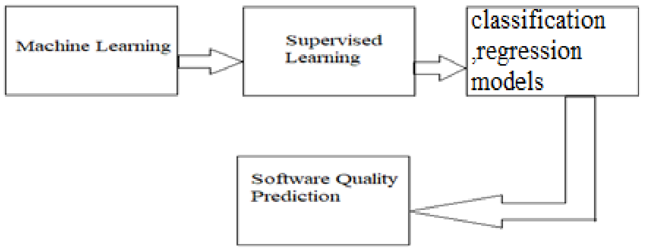
Machine Learning
Supervised Learning
classification ,regression models
Software Quality Prediction

Fig. 2: 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 3 shows the graph of the different data set used.

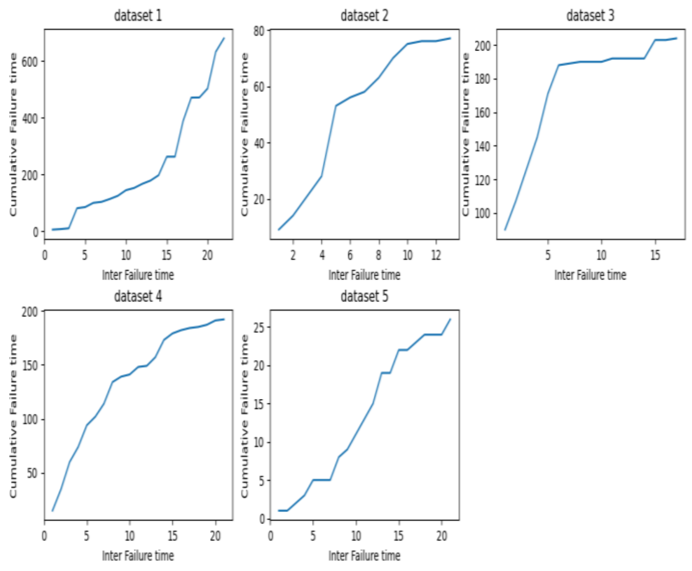

Fig. 3. Correlation of the Different Data Set

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 4.

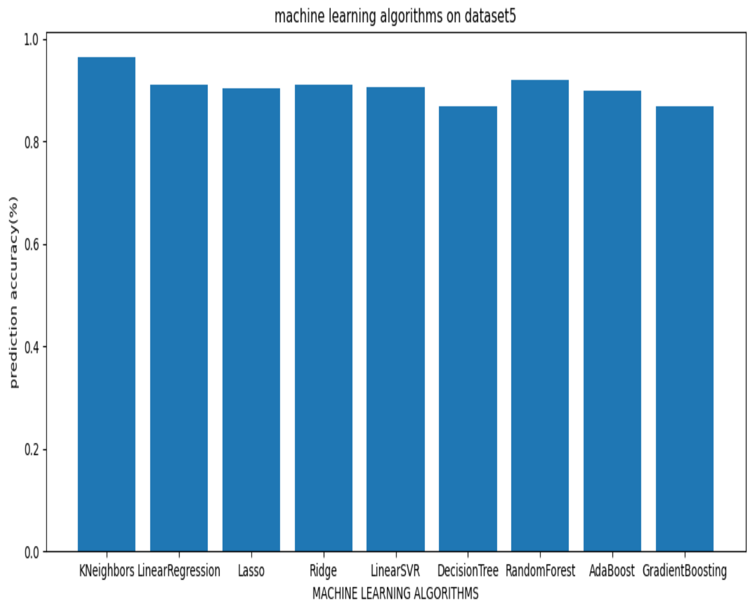

Fig. 4. Result of machine learning models on fifth data set

The figure 5 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.

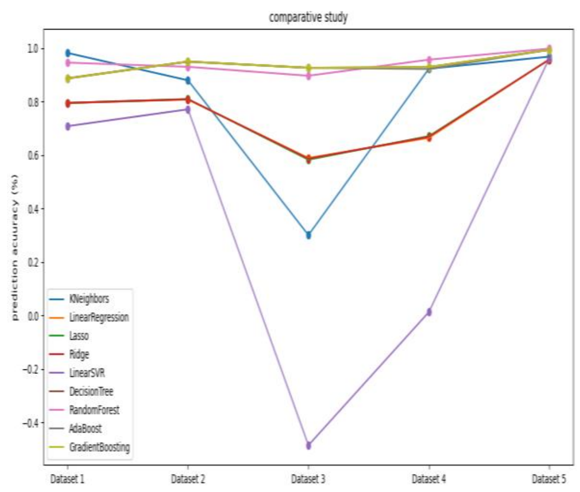

Fig. 5: Result of machine learning models on all data set

7. Conclusion:

Software quality is fundamentally dependent on software reliability. Software quality has long been a top priority for businesses. Because software dependability is probabilistic, it is unpredictable, but we have built prediction models for software reliability with the aid of machine learning models. The bagging and boosting ensemble models have excelled all other models in terms of performance. The methods employed for the ensemble learning were random forest, adaboost, and gradientboost. On the open source software reliability dataset, the KNeighbors algorithm has also produced positive results. The machine learning models were trained using data from the SPSS 14.0 open source software. Software reliability is particularly difficult to anticipate because it varies drastically from software to software on a regular basis. By utilising the suggested methodology and conducting a suitable dataset pre-processing, we may enhance the software dependability prediction.

References:

[1] Bibi S., Tsoumakas G., Stamelos I., Vlahavas I.(2006), "Software Defect Prediction Using Regression via Classification", IEEE International Conference on Computer Systems and Applications, pp.330 – 336.

[2] Norman Fenton, Paul Krause and Martin Neil, (1999), “A Probabilistic Model for Software Defect Prediction”, For submission to IEEE Transactions in Software Engineering.

[3] Ahmet Okutan, Olcay Taner Yıldız,(2012) “Software defect prediction using Bayesian networks”, Empirical Software Eng (2014) 19:154–181 © Springer Science+Business Media, LLC.

[4] Mrinal Singh Rawat, Sanjay Kumar Dubey,(2012) “Software Defect Prediction Models for Quality Improvement: A Literature Study”, IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 5, No 2, pp 288-296.

[5] Supreet Kaur, and Dinesh Kumar, “Software Fault Prediction in Object Oriented Software Systems Using Density Based Clustering Approach”, International Journal of Research in Engineering and Technology (IJRET) Vol. 1 No. 2 March, (2012)ISSN: 2277-4378

[6] Xiao-dong Mu, Rui-hua Chang, Li Zhang, “Software Defect Prediction Based on Competitive Organization Co-Evolutionary Algorithm”, Journal of Convergence Information Technology(JCIT) Volume7, Number5, (2012).

[7] N. Fenton and M. Neil (2008) “Using Bayesian networks to predict software defects and reliability”, Proc. IMechE Vol. 222 Part O: J. Risk and Reliability, pp 702-7.

[8] Jie Xu, ²Danny Ho and ¹Luiz Fernando Capretz, “An Empirical Study On The Procedure Drive Software Quality Estimation Models”, International journal of computer science & information Technology (IJCSIT) Vol.2, No.4, (2010).

[9] Manu Banga, “Computational Hybrids Towards Software Defect Predictions”, International Journal of Scientific Engineering and Technology Volume 2 Issue 5, pp : 311-316, (2013)

[10] Mohamad Mahdi Askari and Vahid Khatibi Bardsiri (2014), “Software Defect Prediction using a High Performance Neural Network”, International Journal of Software Engineering and Its Applications Vol. 8, No. 12 (2014), pp. 177-188.

[11] Kamaljit Kaur (2012), “Analysis of resilient back-propogation for improving software process control” International Journal of Information Technology and Knowledge Management July-December 2012, Volume 5, No. 2, pp. 377-379.

[12] Mrs.Agasta Adline, Ramachandran. M(2014), “Predicting the Software Fault Using the Method of Genetic Algorithm”, International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, Vol. 3, Special Issue 2,, pp 390-398.

[13] R. Li, L. Zhou, S. Zhang, H. Liu, X. Huang, and Z. Sun, ``Software defect prediction based on ensemble learning," in Proc. 2nd Int. Conf. Data Sci. Inf. Technol. (DSIT), Jul. 2019, pp. 1 6, doi: 10.1145/3352411.3352412.

[14] F. Yucalar, A. Ozcift, E. Borandag, and D. Kilinc, ``Multiple-classi ers in software quality engineering: Combining predictors to improve software fault prediction ability," Eng. Sci. Technol., Int. J., vol. 23, no. 4, pp. 938 950, Aug. 2020, doi: 10.1016/j.jestch.2019.10.005.

[15] H. Aljamaan and A. Alazba, ``Software defect prediction using tree-based ensembles," in Proc. 16th ACM Int. Conf. Predictive Models Data Anal. Softw. Eng. (PROMISE). New York, NY, USA: Association for Computing Machinery, Nov. 2020, pp. 1 10, doi: 10.1145/3416508.3417114.

[16] L. Goel, M. Sharma, S. K. Khatri, and D. Damodaran, ``Defect prediction of cross projects using PCA and ensemble learning approach," in Micro-Electronics and Telecommunication Engineering Lecture Notes in Networks and Systems, vol. 106, D. Sharma, V. Balas, L. Son, R. Sharma, and K. Cengiz, Eds. Singapore: Springer, 2020, doi: 10.1007/978-981-15-2329-8_31.