

# Experimental Investigation of Machine Learning Techniques for Predicting Software Quality

Asst. Prof. V.Lakshmi Chaitanya, Nikhitha Sutraye, A.Sai Praveena, U.Naga Niharika,  
P.Ulfath, D.P.Rani  
Department of CSE,  
Santhiram Engineering College,  
Nandyal

**Abstract-**There are several points in the software development process when estimating software quality is useful. Quality assurance planning and benchmarking are two potential applications. Multiple criterion linear programming and quadratic programming are two approaches that have been utilised in prior research to estimate software quality. In addition, we tested out C5.0, SVM, and a Neural network to determine how best to estimate quality. The precision of these research is rather poor. The purpose of this research was to enhance estimate precision by using important properties of a large dataset. In order to improve accuracy, we used a feature selection technique and a correlation matrix. We have also tried our hands at several newer techniques that have proven effective in other prediction challenges. Xgboost, Random Forest, and Decision Tree are only few of the machine learning algorithms used to analyse the data and draw conclusions about the software's quality and its relationship to its development qualities. Results from experiments demonstrate that machine learning algorithms can accurately predict the quality of software.

**Keywords-**Software quality,SVM,Decision tree,Random forest,Navie Bayes,Accuracy,Bagging Classifier,Recall,Precision.

## I. INTRODUCTION

Application software may include bugs that stem from the software development process itself, including requirements analysis, definition, and testing. Therefore, estimating the quality of software is a process that must be performed at different points in its development. Project-based quality assurance practise planning and benchmarking are two potential applications. In addition, the defect density is one of the most prominent measures of software quality. Two studies employing defect quantity from the ISBGS dataset to predict software quality have been conducted, and they are very similar to one another. In the first experiment, both MCLP and MCQP were applied to the dataset and compared.

Using the formula: (number of minor defect + 2\*number of major defect + 4\*number of severe defect), we were able to categorise the overall quality. We required either high or low quality levels. The effectiveness of MCLP and MCQP on the ISBSG database was evaluated with the use of the k-fold cross-validation method. The dataset utilised was Release 10 Dataset (which was made available in January 2007) and it included 4,017 entries and 106 characteristics. The dataset was reduced to 374 records and 11 characteristics after preprocessing. The same database was used once again for another investigation. If the programme has no severe faults, no more than one major defect, and no more than ten minor defects, then it was considered to be of excellent quality. All the rest are automatically categorised as low-quality. 746 projects and

53 characteristics were retained from the original dataset after the cleaning procedure. Classification was performed with C5.0, SVM, and a Neural network. Rashid utilised CBR for software quality estimation as an example in a more practical research. To learn, CBR is a machine learning model that uses data from prior trials. Estimates are made after inputs such the amount of lines of code, the number of functions, the degree of complexity, the kind of development, and the experience level of the programmers involved are made. Disturbance is determined using either the Euclidean distance (ED) or the Manhattan distance (MD).

If the estimated error is less than 10%, the data will be stored. There is a limit on the total number of user inputs that may be collected. The accuracy of an estimate also depends on how near the values in the database are. Biary categorization was used to make quality estimates in these investigations. We made an effort to enhance these prediction models by factoring in size in terms of function points and used a four-tiered categorization system. New categorization approaches that have proven effective in various prediction tasks have been tested by us.

## II. RELATED WORK

**"Software quality metrics in quality assurance to study the impact of external factors related to time."**An analysis of software quality measures is presented here. Quality and software quality are defined, and their respective scopes are discussed. Contrast is drawn

between software metrics and quality assurance. The significance of software quality is also addressed. Both the benefits and drawbacks of comparing software metrics are outlined. Perspectives on software quality are also provided. Metrics for software give a numerical foundation for foreseeing and planning software development cycles. As a result, it is simple to regulate and enhance software quality. Measuring software quality has recently gained attention since it helps increase efficiency. Metrics for measuring software quality are organised into categories.

Measurements may be broken down into "process," "product," and "project" categories. This article takes a look at software engineering to determine the need of software metrics, and it also evaluates the impact software metrics have had on improving software quality and dependability. Several metrics for keeping up with repairs are detailed. We also discuss the factors that influence software quality.

As a result, software metrics have come to the fore as a means to improve quality and hence, productivity. Contrasting points of view on software quality are the subject of this study. Software quality case studies are also described. As we gain expertise with more software measures, we can further refine our results. Gains in quality and dependability might be enormous as a result of such experiences. Costs and automated data collecting for software metrics are also covered.

**"Software defect prediction: do different classifiers find the same defects."**Over the last decade, several models for predicting defects have been published. The classifiers used by these models are said to perform similarly, with models seldom exceeding the predicted performance ceiling of about 80% recall. We assess the degree of uncertainty generated by four different classifiers and look into the specific flaws that they anticipate. We conduct a sensitivity study to evaluate several classifiers, including Naive Bayes, RPart, and SVM, for their ability to anticipate flaws in NASA, open source, and commercial datasets.

Each classifier's defect predictions are recorded in a confusion matrix, and the prediction uncertainty of each is compared. Each of these four classifiers finds unique flaws, while having comparable prediction performance values. It has been shown that certain classifiers are more reliable than others when trying to anticipate failures. Based on our findings, it is clear that some classifiers are able to identify a narrow group of errors.

However, some classifiers' predictions are constant while others' are inconsistent. We draw the conclusion that non-majority voting decision-making procedures in classifier ensembles are most likely to provide the best outcomes in defect prediction.

### III. METHODOLOGY

Using a python function, the contents of the excel files produced after preprocessing were read and assigned to a variable. One variable was set to include the desired property (Quality level), while the other was set to have the other chosen characteristics. Starting with the figures in figures 3 and 4, we may get the correlation matrix. As seen in this matrix, the values of the qualities' correlations with one another are shown. It has been shown that the number of defects correlates highly with software quality in both datasets. A correlation between the time and money spent on software's development and its final quality is to be anticipated. According to Figure 4, time and money are the second and third most critical factors, respectively.

Second, the feature significance table illustrates the target class's association with the chosen classes. The Number of Defects in the dataset was shown to be the most influential factor among the best quality estimation parameters. To put the models into action, we turned to the scikit-learn module in Python. The ratio of training to test data was %33-%67. As a means of calculation, we turned to the scikit-learn library's multi-class classification methods. Figure 5 from the EBSPM dataset demonstrates that the defect process is the most critical, but that the functional size of a programme has a nearly same impact on its quality. In addition, time and money also play crucial roles in determining quality.

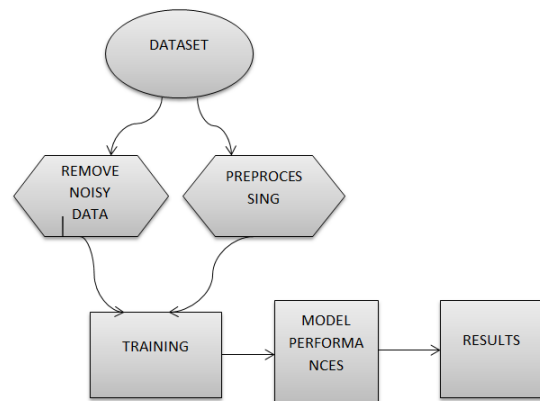


Fig 1. Workdown structure

Figure 6 shows how this is not the case with the ISBGS dataset, where defect amount once again ranks as the most relevant feature. The other characteristics are almost as influential, but they do not compare in importance to defect count. High-quality and low-quality software were identified in the two prior studies with which this one most closely compares. At the edges, this may cause problems with making the right choices. Because of this, we decided to categorise the quality into four levels. Prediction requires the use of multi-class prediction algorithms due to the presence of four distinct categories. The scikit-learn

library's algorithms, in particular those used in multi-class prediction challenges, served our needs well. Bernoulli Naive Bayes, Gaussian Naive Bayes, KNeighbors Decision Tree, Random Forest, XGBoost, Bagging, Logistic Regression, Ridge Classifier, and Nearest Centroid are some examples.

#### IV. RESULT AND DISCUSSION

The creator of this project use a number of machine learning techniques to make inferences about the quality of software. These include the Random Forest, Decision Tree, Gradient Boosting, Bagging Classifier, Logistic Regression, Bernays, and CNN. Because the author's two datasets utilised in the project implementation are not publicly accessible, I am only utilising the dataset(s) that students provide me. Files related to this dataset may be found in the Dataset folder.

By using the training model we can predict the quality of the software, by uploading the related dataset and above image represents the accuracy of each algorithm which are implemented in our project



Fig 4.Run Machinelearning algorithm

#### V. CONCLUSION

In this study, we use the Scikit-learn toolkit to test out several categorization methods on two datasets. New algorithms that allow for several classes have been tested by us. These methods provide accuracies of 92.28 percent on the EBSPM Dataset and 92.22 percent on the ISBSG Dataset. To an acceptable level, multiclass quality prediction might be obtained, especially when compared to directly similar research done in the past.

#### REFERENCES

- [1] V Lakshmi Chaitanya, "Machine Learning Based Predictive Model for Data Fusion Based Intruder Aler tSystem", journal of algebraic statistics, Vol. 13, no. 2, pages. 2477-2483, June2022.
- [2] V. Lakshmi Chaitanya, "Apriorivs Genetic algorithms for Identifying Frequent Item Sets", International journal of Innovative Research & Development, Vol.3,no.6,pages.249-254,June2014.
- [3] M. Sharmila Devi, Farooq Sunar Muhammad, D. Bhavana, D. Sukanya, TV. Sai Thanusha, M.Chandrakala, P. Venkata Swathi "Machine Learning Based Classification and Clustering Analysis of Efficiency of Exercise Against Covid-19 Infection", Journal of Algebraic Statistics, Vol. 13, no. 3, pages. 112-117, June2022.
- [4] M. Sharmila Devi, "A comparative Study of Classification Algorithm for Printed Telugu Character Recognition", International Journal of Electronics Communication and Computer Engineering, Vol.3,no.3,pages .633-641,2012.
- [5] B.Swarajya Lakshmi, "Fire detection using Image processing", Asian Journal of Computer Science and Technology ISSN: 2249-0701 Vol.10 No.2, 2021, pp.14-19, 2021.
- [6] B.Swarajya Lakshmi,—"Identity-Based Proxy-Oriented Data Uploading and Remote Data Integrity checking in Public Cloud", International Journal of Research Vol.5,no.22,pages.744-

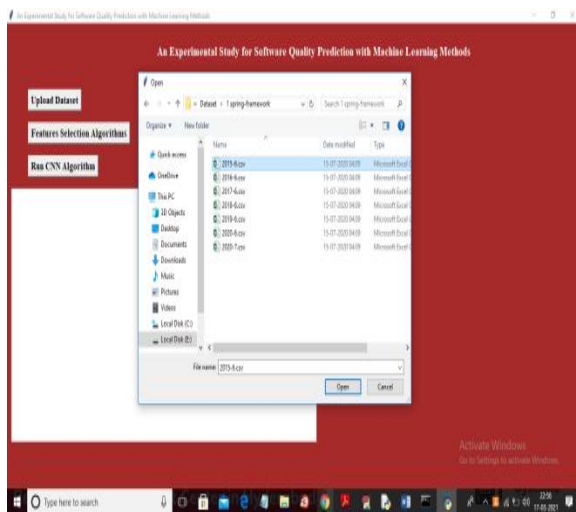


Fig 2.User upload dataset

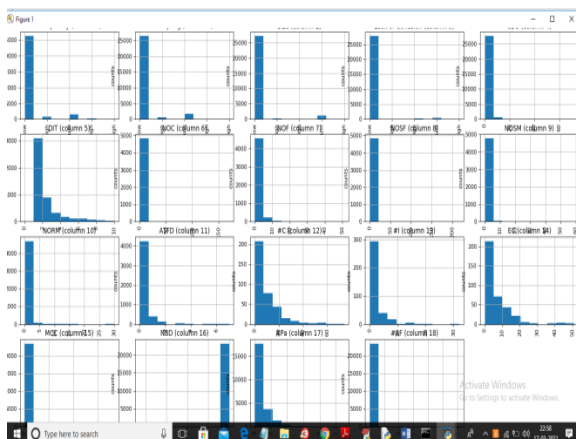


Fig 3.Plotting subgraphs

- 757,2018.
- [7] Sunar mohammed Farooq, "Static Peers for Peer-to-Peer Live Video Streaming", International Journal of Scientific Engineering and Technology Research, Vol.05, No.34, Pages:7055-7064, October-2016.
- [8] Farooq Sunar Mahammad, P Bhaskar, A Prudvi, N Yugandhar Reddy, P Jaswanth Reddy, "Prediction of Covid-19 Infection Based on Lifestyle Habits Employing Random Forest Algorithm", Journal of Algebraic Statistics, Vol.13, No.3, pages.40-45, June 2022.
- [9] Sunar Mohammed Farook, K Nageswara Reddy, "Implementation of Intrusion Detection Systems for High Performance Computing Environment Applications", International Journal of Scientific Engineering and Technology Research, Vol.04, No.41, Pages:8958-8963, October 2015.
- [10] MV Subramanyam, "Automatic feature based image registration using SIFT algorithm", Conference of 2012 Third International Conference on Computing, Communication and Networking Technologies (ICCCNT'12), pages. 1-5, July 2012.
- [11] MV Subramanyam, Mahesh, "Feature based image registration using steerable filters and Harris algorithm", pages. 95-99, January 2012.
- [12] MV Subramanyam, K Satya Prasad, PV Gopi Krishna Rao, "Robust control of steam turbine system speed using improved IMC tuned PID controller", Procedia Engineering, Vol.38, Pages. 1450-1456, January 2012.
- [13] MV Subramanyam, Giri Prasad, "A New Approach for SAR Image Denoising", International Journal of Electrical and Computer Engineering, Vol.5, No.5, Pages. 984-991, October 2015.