A AN EXPERIMENTAL STUDY FOR SOFTWARE QUALITY PREDICTION WITH MACHINE LEARNING METHODS

SYED SHAZAD[1], KAMBHAM SALIVAHANA REDDY[2]

[1]MTech Student, Department of **CSE**, **Global College of Engineering and Technology**, kadapa, AP.

[2]Assistant professor, Department of **CSE**, **Global College of Engineering and Technology**, kadapa, AP.

ABSTRACT:

Software quality estimation is an activity needed at various stages of software development. It may be used for planning the project's quality assurance practices and for benchmarking. In earlier previous studies, two methods (Multiple Criteria Linear Programming and Multiple Criteria Quadratic Programming) for estimating the quality of software had been used Also, C5.0, SVM and Neutral network were experimented with for quality estimation. These studies have relatively low accuracies. In this study, we aimed to improve estimation accuracy by using relevant features of a large dataset. We used a feature selection method and correlation matrix for reaching higher accuracies. In addition, we have experimented with recent methods shown to be successful for other prediction tasks. Machine learning algorithms such as Xgboost, Random Forest and Decision Tree are applied to the data to predict the software quality and reveal the relation between the quality and development attributes. The experimental results show that the quality level of software can be well estimated by machine learning algorithms.

Keywords: SVM, ML, AI, software quality, multiple criteria.

DOI Number: 10.14704/nq.2022.20.12.NQ77049 **NeuroQuantology 2022; 20(12): 662-667**

1. INTRODUCTION:

Software applications may contain defects, originating from requirements analysis, specification and other activities conducted in the software development. Therefore, software quality estimation is an activity needed at various stages

STAGES:

[1]. It may be used for planning the project based quality assurance practices and for benchmarking. In addition, the number of defects per unit is considered one of the most important factors that indicate the quality of the software.

[2].There is two directly comparable studies on software quality prediction using defect quantities in ISBGS dataset. In the first study, the two methods (MCLP and MCQP) were experimented with the dataset and the results were compared

[3]. The quality level was classified according to: number of minor defect + 2*number of major defect + 4*number of extreme defect. The quality of level was to be either high or low. They used k-fold cross-validation technique to measure MCLP and MCQP's performance on the ISBSG database. Release 10 Dataset (released in January 2007) which contained 4,017 records and 106 attributes was used. After preprocessing, 374 records and 11 attributes remained in the dataset. In another study, the same data set was used again.

eISSN 1303-515

[4]. The software belonged to high quality class if it fulfills the following requirements: the extreme defects exist or the number of major defects is more than 1 or the number of minor defects is more than 10. The rest are assumed to belong to low quality class. After preprocessing, 746 projects and 53 attributes remained in the dataset. They used C5.0, SVM and Neutral network for classification. As an example to a more application oriented study Rashid et al.

[5] Used case based reasoning (CBR) for software quality estimation. CBR is a machine learning model which performs the learning process using the results of the previous experiments. Line of code, number of function, difficulty level, and development type and programmers experience are entered and these attributes are used for estimation. The deviation is calculated by using Euclidian distance (ED) or The Manhattan distance (MD). If the error in estimation is less than 10% then the record is saved to the database. Number of inputs that can be obtained from the user is limited. Also, it is necessary to have close values in the database in order to estimating precise values.

2. LITERATURE SURVEY

Software quality metrics in quality assurance to study the impact of external factors related to time:

Software quality assurance is a formal process for evaluating and documenting the quality of the work products during each stage of the software development lifecycle. The practice of applying software metrics to operational factors and to maintain factors is a complex task. Successful software quality assurance is highly dependent on software metrics. It needs linkage the software quality model and software metrics through quality factors in order to offer measure method for software quality assurance. The contributions of this paper build an appropriate method of Software quality metrics application in quality life cycle with software quality assurance. Design: The purpose approach defines some software metrics in the factors and discussed several software quality assurance model and some quality factors measure method. Methodology: This paper solves customer value evaluation problem are: Build a framework of combination of software quality criteria. Describes software metrics. Build Software quality metrics application in quality life cycle with software quality assurance. Results: From the appropriate method of Software quality metrics application in quality life cycle with software quality assurance, each activity in the software life cycle, there is one or more QA quality measure metrics focus on ensuring the quality of the process and the resulting product. Future research is need to extend and improve the methodology to extend metrics that have been validated on one project, using our criteria, valid measures of quality on future software project.

Software defect prediction: do different classifiers find the same defects:

During the last 10 years, hundreds of different defect prediction models have been published. The performance of the classifiers used in these models is reported to be similar with models rarely performing above the predictive performance ceiling of about 80% recall. We investigate the individual defects that

eISSN 1303-515

four classifiers predict and analyse the level of prediction uncertainty produced by these classifiers. We perform a sensitivity analysis to compare the performance of Random Forest, Naïve Bayes, RPart and SVM classifiers when predicting defects in NASA, open source and commercial datasets. The defect predictions that each classifier makes is captured in a confusion matrix and the prediction uncertainty of each classifier is compared. Despite similar predictive performance values for these four classifiers, each detects different sets of defects. Some classifiers are more consistent in predicting defects than others. Our results confirm that a unique subset of defects can be detected by specific classifiers. However, while some classifiers are consistent in the predictions they make, other classifiers vary in their predictions. Given our results, we conclude that classifier ensembles with decision-making strategies not based on majority voting are likely to perform best in defect prediction.

A Knowledge Discovery Case Study of Software Quality Prediction:

Software becomes more and more important in modern society. However, the quality of software is influenced by many un-trustworthy factors. This paper applies MCLP model on ISBSG database to predict the quality of software and reveal the relation between the quality and development attributes. The experimental result shows that the quality level of software can be well predicted by MCLP Model. Besides, several useful conclusions have been drawn from the experimental result.

Evidence-based software portfolio management:

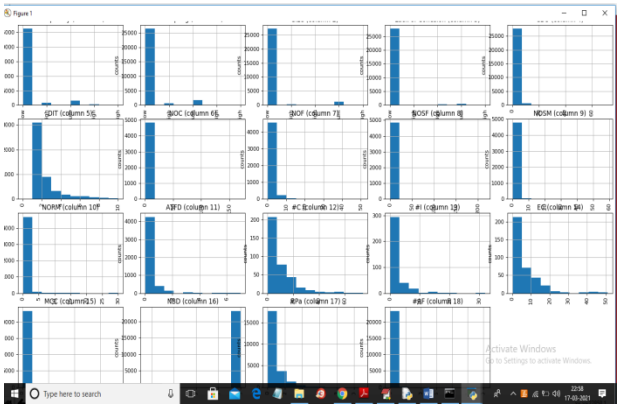
In this paper we describe and evaluate a tool for Evidence-Based Software Portfolio Management (EBSPM) that we developed over time in close cooperation with software practitioners from The Netherlands and Belgium. Objectives: The goal of the EBSPM-tool is to measure, analyze, and benchmark the performance of interconnected sets of software projects in terms of size, cost, duration, and number of defects, in order to support innovation of a company's software delivery capability. The tool supports building and maintaining a research repository of finalized software projects from different companies, business domains, and delivery approaches. Method: The tool consists of two parts. First, a *Research Repository*, at this moment holding data of for now 490 finalized software projects, from four different companies. Second, a *Performance Dashboard*, built from a so-called *Cost Duration Matrix*. Results: We evaluated the tool by describing its use in two practical applications in case studies in industry. Conclusions: We show that the EBSPM-tool can be used successfully in an industrial context, especially regarding its benchmarking and visualization purposes.

3. METHODOLOGY

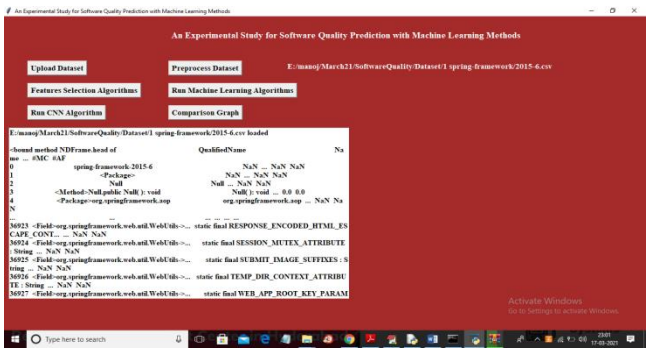
Secondly, feature importance table reveal the relationship of the target class with other selected classes. Among the best of the factors selected to estimate the quality, it is revealed that the most influencing attribute of the dataset is the Number of Defects. We used Python scikit-learn library for implementing the models. Training and test data were separated by %33-%67 ratio. Multi-class classification

eISSN 1303-515

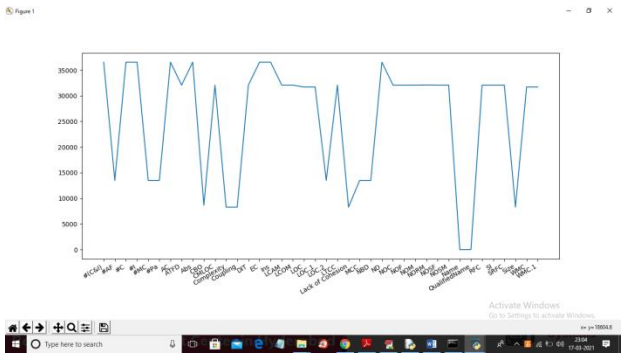
algorithms in the scikit-learn library was used for estimation.



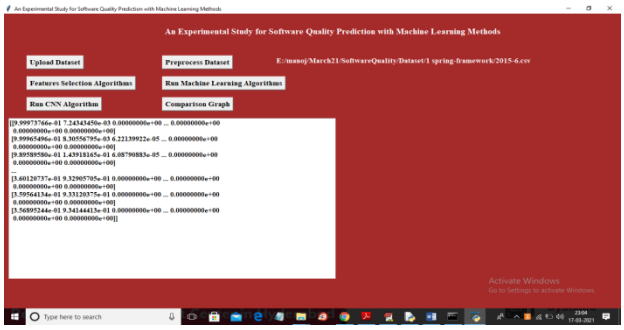
In above graph we can see each graph represents one column from dataset and from that columns its counting each distinct value from and plot in that graph for example in second graph NOC columns 3 different values and its plotting 3 different bars with count and no close above graph to get below screen



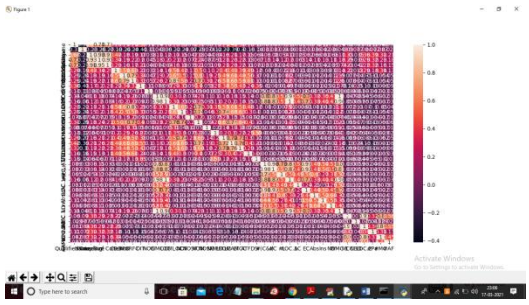
In above screen displaying values from dataset and we can see dataset contains NAN (missing values) and string non numeric values and we need to replace all missing and non-numeric values with their count so click on 'Preprocess Dataset' button



In above graph x-axis represents column names and y-axis represents total missing values counts in that column and now close above graph to get below screen

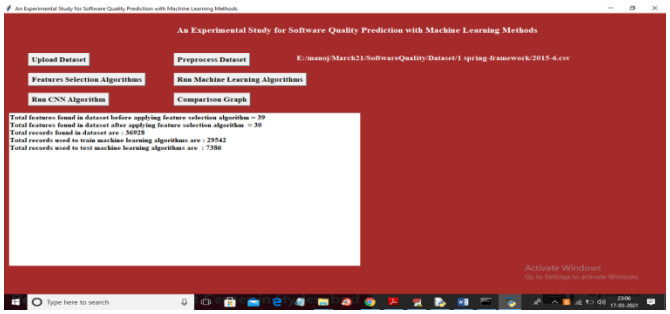


In above screen all missing an string values are replace with numeric values and now click on 'Features Selection Algorithms' button to select important features from dataset and then split dataset into train and test part

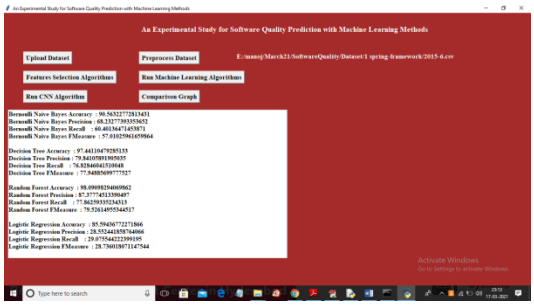


In above graph the box which contains value >0.5 will be consider as important attributes and now close above graph to get below screen

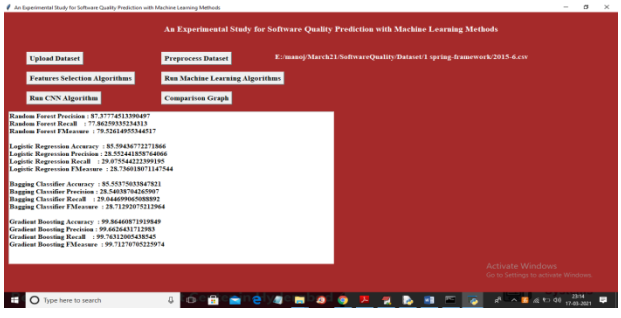
eISSN 1303-515



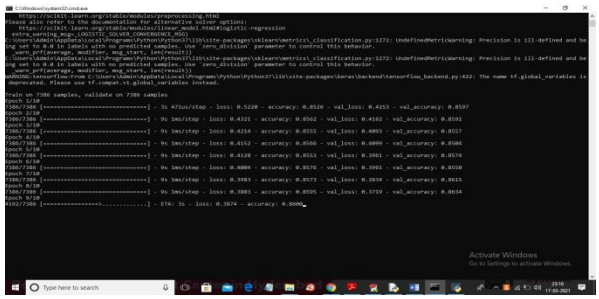
In above screen before applying feature selection algorithm dataset contains 39 features/columns and after applying PCA feature selection we got 30 important features and dataset contains 36928 records and application using 7386 records for testing and 29542 records for training and now both train and test dataset is ready and now click on 'Run Machine Learning Algorithms' button to run all machine learning algorithms



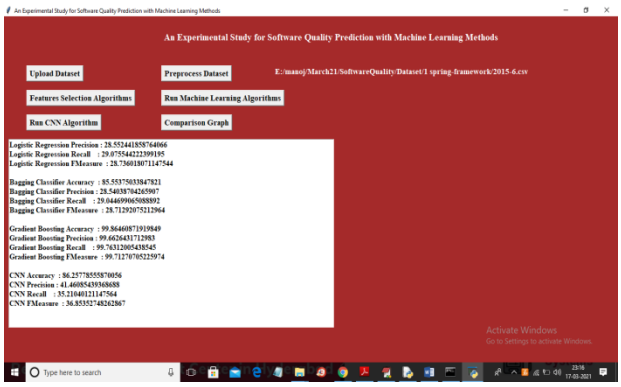
In above screen we can precision, recall, accuracy and fscore for all algorithms



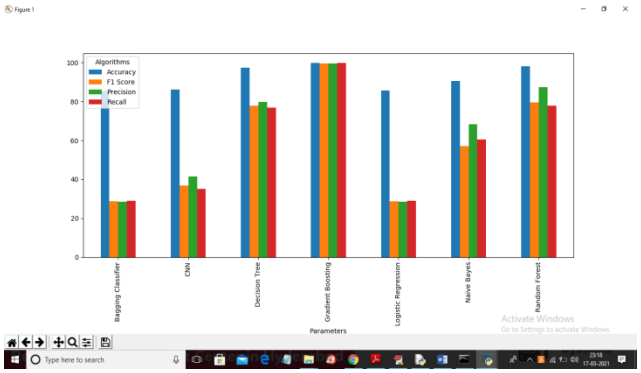
Now click on 'Run CNN Algorithm' button to run CNN algorithm and to get below screen



In above screen to train CNN we took 10 iterations or epoch and at each epoch accuracy get better and loss get reduce and after 10 iterations will get below screen



In above screen we got output values for CNN also and now click on 'Comparison Graph' button to get below screen



In above graph we are plotting accuracy, precision, recall and accuracy for each algorithm

CONCLUSION

eISSN 1303-515

In this paper we have experimented classification algorithms using Scikit-learn library on two dataset. We have experimented with recent algorithms that support multi-class classification. The accuracies achieved by using these algorithms are 92.28% on EBSPM Dataset and 92.22% on ISBSG Dataset. In comparison to previous directly comparable studies, acceptable level multiclass quality prediction could be achieved.

REFERANCES

[1] Vijay, T. John, D. M. G. Chand, and D. H. Done. "Software quality metrics in quality assurance to study the impact of external factors related to time." International Journal of Advanced Research in Computer Science and Software Engineering, 2017.

[2] D. Bowes, T. Hall, and J. Petrić, "Software defect prediction: do different classifiers find the same defects?." Software Quality Journal, 26(2), 2018, pp. 525-552.

[3] X. Wang, Y. Zhang, L. Zhang and Y. Shi, "A Knowledge Discovery Case Study of Software Quality Prediction: ISBSG Database," 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, Toronto, ON, 2010, pp. 219-222.

[4] X. Wang, Y. Zhang, L. Zhang and Y. Shi, "A Knowledge Discovery Case Study of Software Quality Prediction Based on Classification Models: ISBSG Database," The 11th International Symposium on Knowledge Systems Sciences (KSS 2010), 2010

[5] E. Rashid, S. Patnaik, and V. Bhattacherjee, "Software quality estimation using machine learning: Case-Based reasoning technique, " International Journal of Computer Applications, 2012

[6] www.isbsg.org

[7] https://goverdson.nl/

[8] H. Huijgens,"Evidence-based software portfolio management: a tool description and evaluation", 20th International Conference on Evaluation and Assessment in Software Engineering (EASE '16), 201