Industrial Engineering Journal
ISSN: 0970-2555
Volume : 51, Issue 10, July-December: 2022

MACHINE LEARNING TECHNIQUES FOR SOFTWARE QUALITY ASSESSMENT AND PREDICTION: AN EXPERIMENTAL STUDY

Dr. BTR Naresh Reddy[1] , B. Bhagyalaxmi[2]

[1] Professor, Department of Computer Applications, Aurora's PG College (MBA), Uppal, Hyderabad
Email**:** reddynareshbtr@gmail.com

[2]Assistant Professor, Department of Computer Applications, Aurora's PG College (MBA), Uppal, Hyderabad
Email: laxmibbaghya@gmail.com

ABSTRACT

Software development requires the activity of software quality estimate at different phases. It could be utilized for benchmarking and for organizing the project's quality assurance procedures. Two techniques (Multiple Criteria Linear Programming and Multiple Criteria Quadratic Programming) for assessing software quality had been applied in earlier research. For quality estimation, experiments were also conducted with C5.0, SVM, and Neutral network. The accuracy of these investigations is comparatively low. Our goal in this work was to use pertinent information from a huge dataset to increase estimation accuracy. To achieve improved accuracies, we employed a correlation matrix and feature selection technique. We have also experimented with new techniques that have been successful for other prediction problems. Xgboost, Random Forest, and Decision Tree are a few machine learning algorithms that are used on data to forecast software quality and show the relationship between quality and development variables. The experimental findings demonstrate that machine learning algorithms are capable of accurately estimating the software quality level.

Keywords: Xgboost, Random Forest And Decision Tree, Machine Learning, Software Quality, Assessment, Prediction

1. INTRODUCTION

Defects in software applications can arise from requirements analysis, definition, and other software development tasks. Consequently, software quality estimate is a task that is required at different phases.
Phases:

1. It can be utilized for benchmarking and for organizing project-based quality assurance procedures. Furthermore, one of the key indicators of the software's quality is thought to be the quantity of flaws per unit

2. The ISBGS dataset's defect numbers are used in two directly comparable studies on software quality prediction. In the first study, the dataset was used to test the two approaches (MCLP and MCQP), and the outcomes were compared.

3. The number of minor defects plus two times the number of major defects plus four times the number of extreme defects was used to categorize the quality level. There would be two quality levels: high and low. On the ISBSG database, they measured MCLP and MCQP's performance using the k-fold cross-validation technique. The dataset known as Release 10, which was made available in January 2007, comprised 4,017 records and 106 attributes. The dataset contained 374 records and 11 attributes after preprocessing. The identical data set was used once more in a different investigation.

4.If the program satisfies the following criteria, it is considered high quality: either there are serious flaws, or there are more than one big defect or ten minor problems. The remaining ones are thought to be of lower quality. 53 attributes and 746 projects were still included in the dataset following preprocessing. For classification, they employed C5.0, SVM, and Neutral Network. As an illustration of a study that was more application-focused, Rashid et al.

5. Employed case-based reasoning (CBR) to estimate the quality of software. A machine learning model called CBR uses the findings of earlier experiments to carry out the learning process. The estimation process takes into account the line of code, number of functions, difficulty level, development kind, and programmers' experience, all of which are entered. The Manhattan distance (MD) or the Euclidian distance (ED) are used to compute the deviation. The record is stored in the database if the estimation error is less than 10%. There is a maximum amount of inputs that can be collected from the user. Additionally, in order to estimate precise values, the database must have close values.

2.LITERATURE SURVEY AND RELATED WORK

Software quality metrics in quality assurance to study the impact of external factorsrelated 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.

2.2. 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 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, R Part 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.

2.3 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.

3.EXISTING SYSTEM

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 . 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.

Disadvantages:
Low accuracy

4. PROPOSED SYSTEM

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.

Advanatages:
High accuracy than existing methods.

5. METHODOLOGIES

MODULE

Tensorflow:
TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.
TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

Numpy:
Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.
It is the fundamental package for scientific computing with Python. It contains various features including these important ones:
A powerful N-dimensional array object.
Sophisticated (broadcasting) functions.
Tools for integrating C/C++ and Fortran code.
Useful linear algebra, Fourier transform, and random number capabilities Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows Numpy to seamlessly and speedily integrate with a wide variety of databases.

Pandas:
Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Matplotlib:
Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.
For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

Scikit – learn:

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.
Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

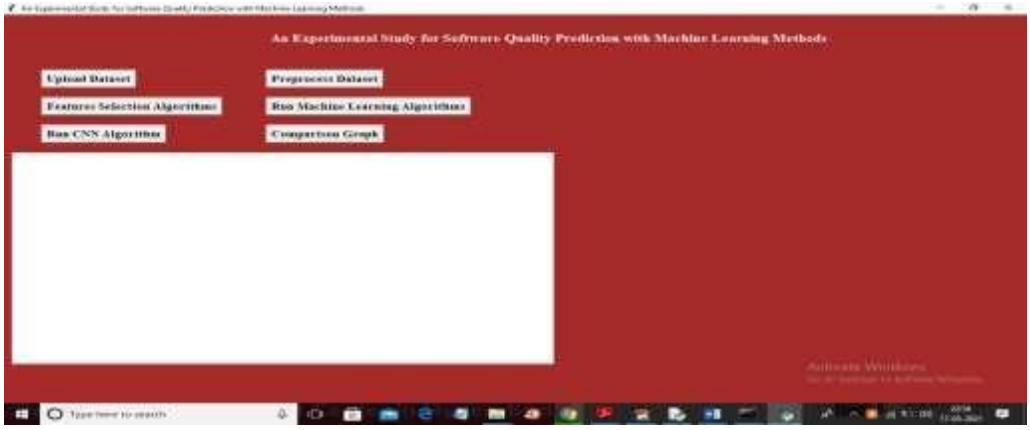
Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

Python is Interactive − you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

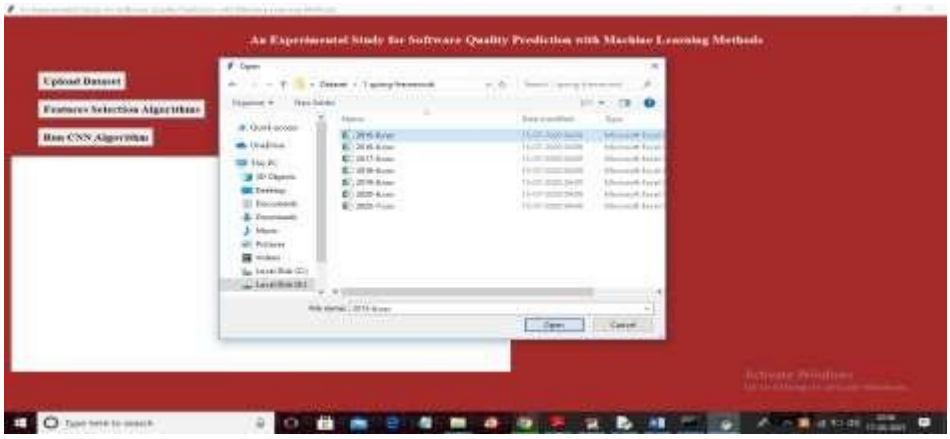
Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

6. RESULTS AND DISCUSSION

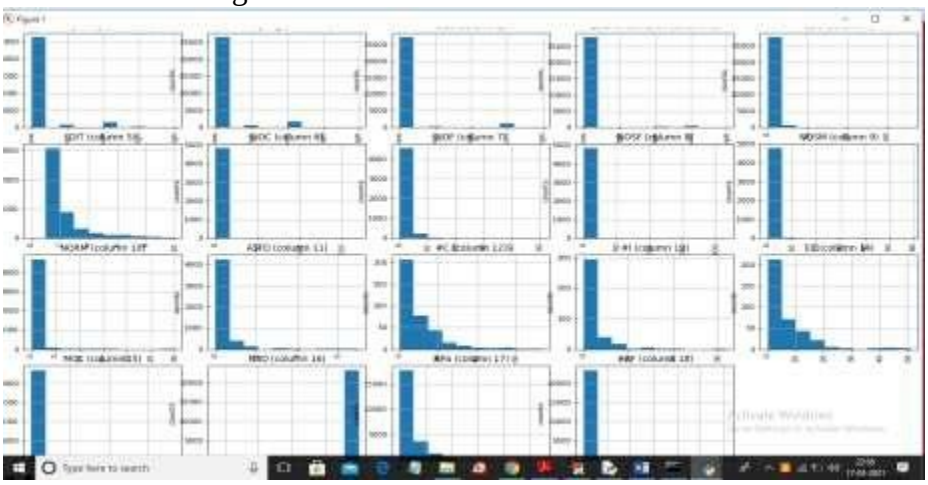


In above screen click on ‘Upload Dataset’ button to and upload dataset

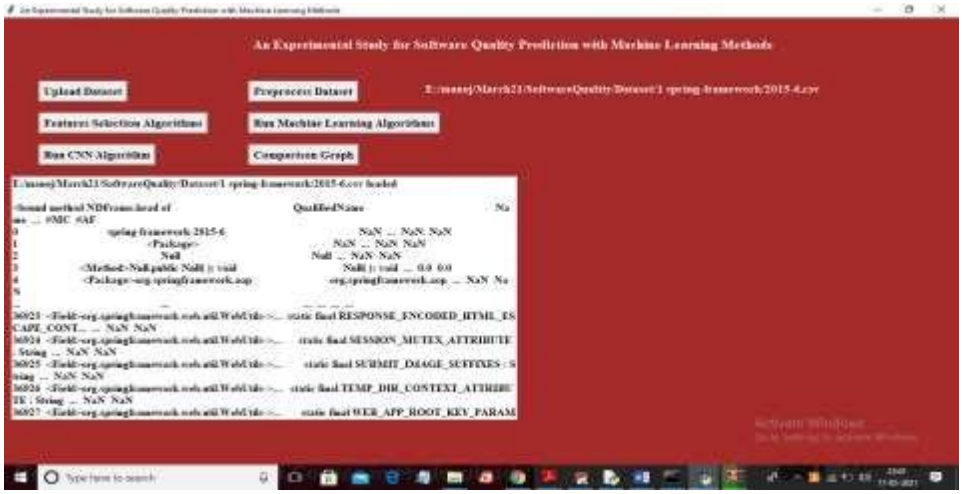
Industrial Engineering Journal
ISSN: 0970-2555
Volume : 51, Issue 10, July-December: 2022



In above screen selecting and uploading '2015-6.csv' dataset file and then click on 'Open' button to load dataset and to get below screen



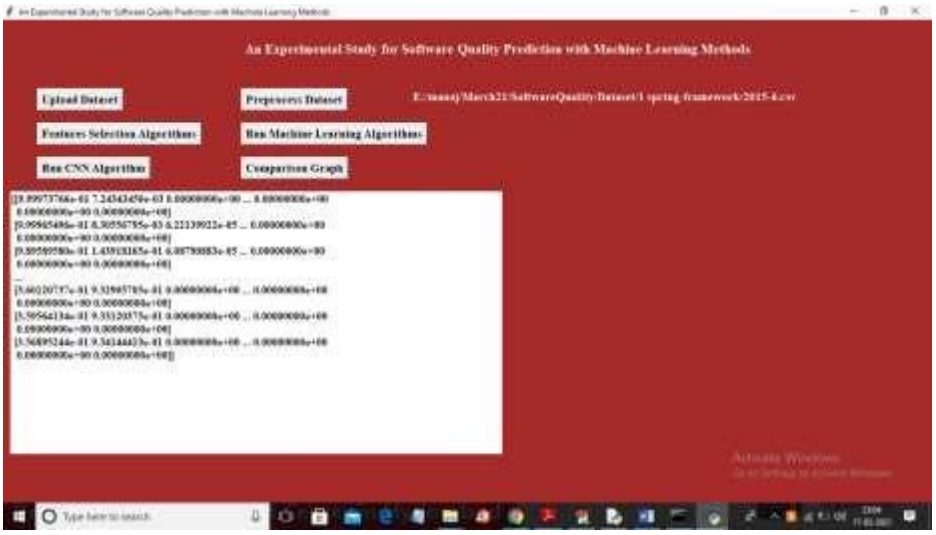
In above graph we can see each graph represents one column from dataset and from that columns its counting each distinct value from and plot in that graph for example in second graph NOC columns 3 different values and its plotting 3 different bars with count and no close above graph to get below screen



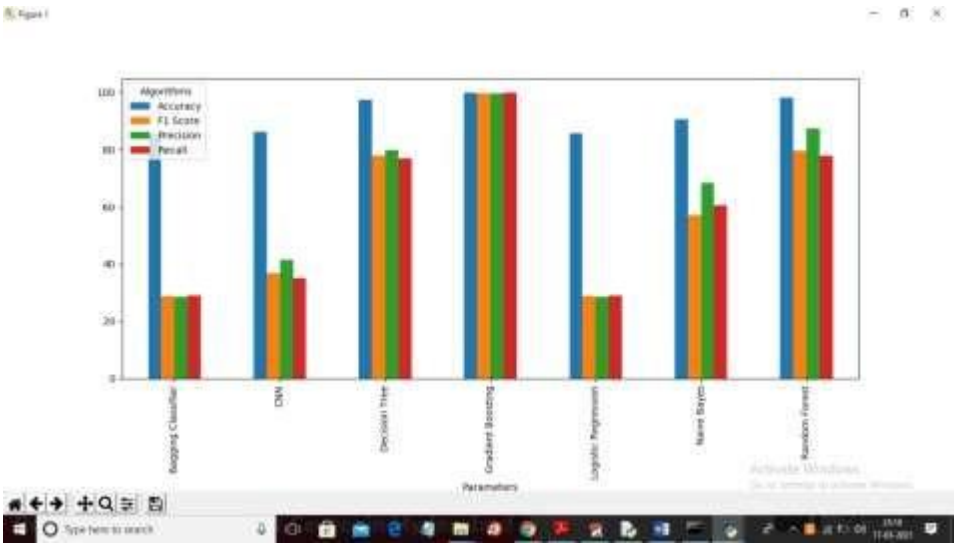
In above screen displaying values from dataset and we can see dataset contains NAN

(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 we are plotting accuracy, precision, recall and accuracy for each algorithm

7. CONCLUSION AND FUTURE SCOPE

In this project, we used the Scikit-learn toolkit to experiment with two datasets' worth of categorization techniques. We have experimented with new multi-class classification algorithms. With the EBSPM Dataset and the ISBSG Dataset, these algorithms yielded accuracy levels of 92.28% and 92.22%, respectively. It was possible to attain an acceptable degree of multiclass quality prediction when compared to earlier directly similar research.

REFERENCES

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