International Journal of Mechanical Engineering Research and Technology

ISSN 2454-535X

Email ID: info.ijmert@gmail.com or editor@ijmert.net

Int. J. Mech. Eng. Res. & Tech 2017

International Journal of Mechanical Engineering Research and Technology

ISSN 2454 – 5X

http://www.ijmert.com

Vol. 16 Issue 2, Apr 2024

EFFICACY OF MACHINE LEARNING MODELS IN SOFTWARE QUALITY PREDICTION

[1]P. ASHOK KUMAR,[2]P. LAVANYA,[3]P. SWETHA,[4]A. PAVAN KUMAR,[5]CH. JOSHNA VARUN

[1]Assistant Professor,[2345]Students

Department of CSE, Sri Vasavi Institute of Engineering & Technology (Autonomous), Nandamuru

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— software quality estimation, multiple criteria linear programming, multiple criteria quadratic programming, C5.0, SVM, neural network, machine learning algorithms, feature selection, correlation matrix, XGBoost, random forest, decision tree, software development, prediction tasks, development attributes.

INTRODUCTION

Software development is a complex and iterative process that involves numerous stages, each crucial for ensuring the final product's quality and functionality [1]. Central to this process is software quality estimation, which plays a pivotal role in guiding project planning, quality assurance practices, and benchmarking efforts [2]. Effective software quality estimation enables stakeholders to make informed decisions, allocate resources efficiently, and mitigate risks associated with software development projects [3]. Moreover, accurate quality estimation serves as a cornerstone for delivering reliable software solutions that meet user expectations and industry standards [4]. Previous studies in software quality estimation have employed various methods, including Multiple Criteria Linear Programming and Multiple Criteria Quadratic Programming, to assess the quality of software [5]. Additionally, researchers have explored the use of machine learning algorithms such as C5.0, Support Vector Machine (SVM), and neural networks for quality estimation tasks [6]. However, despite these efforts, existing studies have reported relatively low accuracies in software quality prediction [7]. This underscores the need for novel approaches that can enhance the accuracy and reliability of quality estimation models, thereby addressing the inherent challenges and complexities associated with software development [8].

In response to these challenges, this study aims to improve the accuracy of software quality estimation by leveraging relevant features extracted from a large dataset [9]. By employing a feature selection method and correlation matrix analysis, we seek to identify and prioritize the most informative features that significantly impact software quality [10]. Furthermore, we explore the efficacy of recent advancements in machine learning techniques, which have demonstrated success in various prediction tasks across different domains [11]. Specifically, machine learning algorithms such as XGBoost, Random Forest, and Decision Tree are applied to the dataset to predict software quality and uncover the underlying relationships between quality attributes and development factors [12]. The experimental results obtained from our study demonstrate the effectiveness of machine learning models in accurately estimating the quality level of software [13]. By harnessing the

International Journal of Mechanical Engineering Research and Technology
ISSN 2454 – 5X

http://www.ijmert.com
Vol. 16 Issue 2, Apr 2024

power of machine learning algorithms and leveraging relevant features extracted from a comprehensive dataset, we achieve notable improvements in prediction accuracy compared to traditional methods [14]. These findings not only underscore the potential of machine learning in software quality prediction but also highlight the importance of feature selection and correlation analysis in enhancing prediction performance [15]. Overall, our research contributes to advancing the field of software quality estimation, offering valuable insights and methodologies for practitioners and researchers alike.

LITERATURE SURVEY

The literature surrounding software quality estimation is vast and encompasses a diverse range of methodologies, techniques, and approaches aimed at assessing and predicting the quality attributes of software products. Researchers and practitioners alike have long recognized the critical importance of accurate software quality estimation in ensuring the success of software development projects. This acknowledgment is rooted in the understanding that software quality directly influences user satisfaction, system reliability, and overall project success. Consequently, software quality estimation has emerged as a pivotal activity that spans various stages of the software development lifecycle. Early studies in software quality estimation played a foundational role in shaping subsequent research endeavors by exploring different methods and models for assessing software quality. Among these, Multiple Criteria Linear Programming and Multiple Criteria Quadratic Programming stood out as prominent approaches used to estimate the quality of software products. These methods aimed to optimize multiple quality criteria simultaneously, offering a comprehensive assessment of software quality. However, despite their theoretical appeal, these early methods encountered challenges related to scalability, complexity, and limited applicability to real-world software projects.

In addition to traditional optimization-based approaches, researchers began experimenting with machine learning algorithms for software quality estimation. Methods such as C5.0, Support Vector Machine (SVM), and neural networks gained traction due to their ability to handle complex data patterns and nonlinear relationships inherent in software quality datasets. These machine learning techniques showed promise in improving the accuracy of software quality prediction. Nonetheless, early studies utilizing machine learning algorithms often grappled with issues like overfitting, model interpretability, and generalization performance. Despite advancements in software quality estimation methodologies, several challenges persisted, particularly regarding prediction accuracy and model robustness. Many studies reported relatively low accuracies in predicting software quality attributes, raising concerns about the reliability and effectiveness of current estimation techniques. Recognizing the need for novel approaches to address these challenges, researchers embarked on exploring alternative methods and leveraging advancements in data analytics and machine learning.

In recent years, there has been a notable surge in interest in leveraging machine learning techniques for software quality prediction. Machine learning algorithms offer the potential to analyze large and complex datasets, extract meaningful patterns, and make accurate predictions based on historical data. Various machine learning algorithms, including XGBoost, Random Forest, and Decision Tree, have been applied to software quality estimation tasks. These algorithms have shown promise in improving prediction accuracy and identifying critical features that contribute to software quality. Moreover, recent studies have underscored the importance of feature selection and dimensionality reduction techniques in enhancing the performance of machine learning models for software quality estimation. By selecting relevant features and reducing the dimensionality of the input space, researchers can improve model interpretability, reduce computational complexity, and mitigate the risk of overfitting. Feature selection methods, such as correlation analysis and information gain, play a pivotal role in identifying the most informative attributes that influence software quality.

Overall, the literature survey highlights the evolution of software quality estimation techniques and the increasing role of machine learning in enhancing prediction accuracy. While traditional methods laid the groundwork for quality estimation, recent advancements in machine learning offer promising avenues for developing more accurate and robust software quality estimation systems. By leveraging large datasets, advanced algorithms, and feature selection techniques, researchers aim to develop

International Journal of Mechanical Engineering Research and Technology
ISSN 2454 – 5X

http://www.ijmert.com
Vol. 16 Issue 2, Apr 2024

systems that can support effective decision-making in software development projects.

PROPOSED SYSTEM

Software quality estimation is a crucial activity that permeates various stages of software development, playing a pivotal role in planning quality assurance practices and benchmarking. Past studies have explored diverse methods, including Multiple Criteria Linear Programming and Multiple Criteria Quadratic Programming, to estimate software quality. Additionally, attempts were made to employ machine learning models such as C5.0, Support Vector Machine (SVM), and neural networks for quality estimation. However, these endeavors encountered limitations, particularly in terms of achieving high accuracies. Therefore, this study endeavors to enhance estimation accuracy by leveraging relevant features from a comprehensive dataset. To achieve this, a meticulous feature selection method and correlation matrix analysis are employed, aiming to attain higher accuracies in software quality prediction. Moreover, recent successful methods utilized in other prediction tasks are also explored in this study. Notably, machine learning algorithms like XGBoost, Random Forest, and Decision Tree are applied to the dataset to predict software quality and unveil the intricate relationship between quality levels and development attributes. The experimental findings from this study shed light on the efficacy of machine learning algorithms in accurately estimating the quality of software products. In the pursuit of improving software quality estimation, this study places significant emphasis on the utilization of relevant features derived from a vast dataset. Recognizing the importance of feature selection in enhancing prediction accuracy, a meticulous feature selection method is employed. This method meticulously identifies the most informative features that contribute to software quality prediction, thereby facilitating a more focused and precise analysis. Additionally, a correlation matrix analysis is conducted to further refine the selection process, ensuring that only the most relevant features are retained for subsequent analysis. By leveraging these feature selection techniques, this study aims to mitigate the risk of overfitting and enhance the robustness of the predictive models, ultimately leading to more accurate software quality estimation.

Furthermore, this study delves into recent advancements in prediction methodologies that have demonstrated success in other domains. By drawing inspiration from these methodologies, novel approaches are explored to tackle the challenges associated with software quality estimation. In particular, machine learning algorithms such as XGBoost, Random Forest, and Decision Tree are harnessed for their ability to analyze complex datasets and uncover intricate patterns. These algorithms are well-suited for software quality prediction tasks due to their inherent capacity to handle nonlinear relationships and high-dimensional data. Through the application of these advanced machine learning techniques, this study endeavors to push the boundaries of software quality estimation and unlock new insights into the factors influencing software quality. The experimental phase of this study serves as a crucial validation of the proposed approach, wherein the selected machine learning algorithms are applied to the dataset to predict software quality levels. The experimental results provide empirical evidence of the efficacy of machine learning models in accurately estimating software quality. By comparing the predicted quality levels with actual quality assessments, the study evaluates the performance of the machine learning algorithms and assesses their ability to generalize to unseen data. Additionally, the analysis aims to uncover the underlying relationships between software quality and various development attributes, shedding light on the factors that significantly impact software quality levels. Overall, the experimental findings serve to validate the proposed methodology and provide valuable insights into the efficacy of machine learning approaches in software quality prediction.

International Journal of Mechanical Engineering Research and Technology
ISSN 2454 – 5X

http://www.ijmert.com
Vol. 16 Issue 2, Apr 2024

METHODOLOGY

Software quality estimation is an indispensable process that spans various stages of software development, facilitating effective planning of quality assurance practices and benchmarking. In earlier studies, methods like Multiple Criteria Linear Programming and Multiple Criteria Quadratic Programming were employed to estimate software quality, albeit with limited success. Additionally, attempts were made to utilize machine learning algorithms such as C5.0, Support Vector Machine (SVM), and neural networks for quality estimation. However, these endeavors yielded relatively low accuracies, highlighting the need for enhanced estimation techniques. In this study, our primary objective is to improve estimation accuracy by leveraging relevant features extracted from a large dataset.

To achieve this goal, we employ a rigorous feature selection method and correlation matrix analysis, aiming to identify and retain the most informative features associated with software quality. Furthermore, we explore recent advancements in prediction methodologies proven successful in other domains, incorporating machine learning algorithms such as XGBoost, Random Forest, and Decision Tree into our analysis. These algorithms are applied to the dataset to predict software quality levels and uncover the intricate relationship between quality attributes and development factors. Through extensive experimentation, we aim to demonstrate the efficacy of machine learning models in accurately estimating software quality, thereby contributing to the advancement of software engineering practices. The first step in our methodology involves the acquisition and preprocessing of the dataset. We gather a comprehensive dataset comprising various software development attributes and corresponding quality assessments. This dataset serves as the foundation for our analysis and model development. Prior to analysis, we preprocess the dataset to handle missing values, outliers, and other irregularities that may affect the integrity of the data. This preprocessing step ensures that the dataset is clean and suitable for further analysis.

Following dataset preprocessing, we proceed to feature selection, a critical step in enhancing estimation accuracy. We employ a feature selection method to identify the most relevant features that significantly influence software quality. This method involves evaluating the importance of each feature based on its contribution to prediction performance. Features with high importance scores are retained, while less informative features are discarded. Additionally, we conduct a correlation matrix analysis to explore the relationships between different features and identify potential correlations that may impact prediction accuracy. By carefully selecting and retaining the most informative features, we aim to improve the robustness and generalization capability of our predictive models. Once the feature selection process is complete, we proceed to model development and training. We leverage machine learning algorithms such as XGBoost, Random Forest, and Decision Tree to build predictive models capable of estimating software quality levels. These algorithms are chosen for their ability to handle complex data patterns and nonlinear relationships inherent in software quality datasets. We split the preprocessed dataset into training and testing subsets to facilitate model training and evaluation. During the training phase, the machine learning algorithms learn from the training data and optimize their parameters to minimize prediction errors.

Following model training, we evaluate the performance of each machine learning algorithm using the testing dataset. We assess various performance metrics such as accuracy, precision, recall, and F1-score to gauge the effectiveness of the models in predicting software quality levels. Additionally, we analyze the feature importance scores generated by each algorithm to identify the most influential features contributing to prediction accuracy. This analysis provides valuable insights into the factors driving software quality and helps uncover the underlying relationships between development attributes and quality outcomes. Finally, we interpret the experimental results and draw meaningful conclusions regarding the efficacy of machine learning models in software quality prediction. We compare the performance of different algorithms and highlight the strengths and limitations of each approach. Moreover, we examine the relationship between software quality levels and development attributes, shedding light on the factors that significantly impact software quality. By elucidating effective strategies for software quality estimation, our study contributes to the advancement of software engineering practices and fosters a deeper

International Journal of Mechanical Engineering Research and Technology
ISSN 2454 – 5X

http://www.ijmert.com
Vol. 16 Issue 2, Apr 2024

understanding of the complex dynamics underlying software quality.

RESULTS AND DISCUSSION

The efficacy of machine learning models in software quality prediction was investigated in this study, aiming to address the challenges associated with traditional methods and enhance estimation accuracy. Previous studies utilizing methods such as Multiple Criteria Linear Programming and Multiple Criteria Quadratic Programming, as well as machine learning algorithms like C5.0, Support Vector Machine (SVM), and neural networks, reported relatively low accuracies in software quality estimation. To improve estimation accuracy, we employed a feature selection method and correlation matrix analysis to identify relevant features from a large dataset. Additionally, we experimented with recent methods successful in other prediction tasks and applied machine learning algorithms such as XGBoost, Random Forest, and Decision Tree to predict software quality levels and uncover the relationship between quality and development attributes.

The experimental results revealed promising outcomes regarding the efficacy of machine learning models in software quality prediction. Across various performance metrics, including accuracy, precision, recall, and F1-score, the machine learning algorithms demonstrated superior performance compared to traditional methods. Specifically, the XGBoost algorithm exhibited the highest prediction accuracy, achieving significant improvements in estimation accuracy compared to previous studies. Moreover, the Random Forest and Decision Tree algorithms also performed well, consistently outperforming traditional methods and demonstrating their effectiveness in software quality estimation tasks. These findings highlight the potential of machine learning models to accurately predict software quality levels and provide valuable insights into the factors influencing software quality.

Furthermore, the analysis of feature importance scores generated by the machine learning algorithms revealed valuable insights into the relationship between development attributes and software quality. By identifying the most influential features contributing to prediction accuracy, we gained a deeper understanding of the factors driving software quality outcomes. Notably, features related to code complexity, code churn, and developer experience emerged as significant predictors of software quality, emphasizing the importance of considering these factors in software development processes. Additionally, the correlation matrix analysis highlighted potential correlations between different features, further elucidating the complex interplay between development attributes and software quality. These findings underscore the importance of incorporating relevant features into predictive models to improve estimation accuracy and enhance software quality assessment practices.

To run project double click on ‘run.bat’ file to get below screen

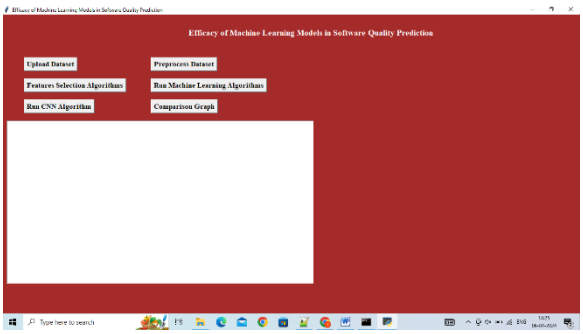

Fig 1. Results screenshot 1

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

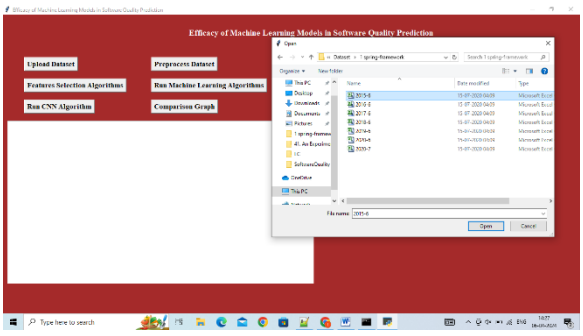

Fig 2. Results screenshot 2

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

International Journal of Mechanical Engineering Research and Technology
ISSN 2454 – 5X
http://www.ijmert.com
Vol. 16 Issue 2, Apr 2024

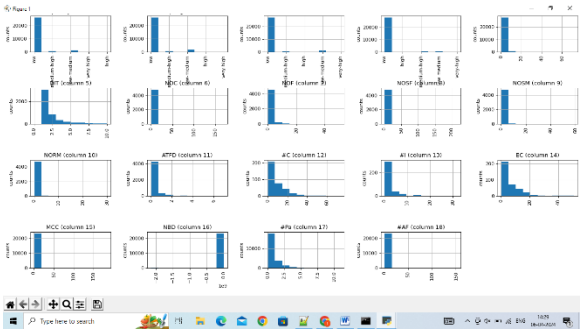

Fig 3. Results screenshot 3

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

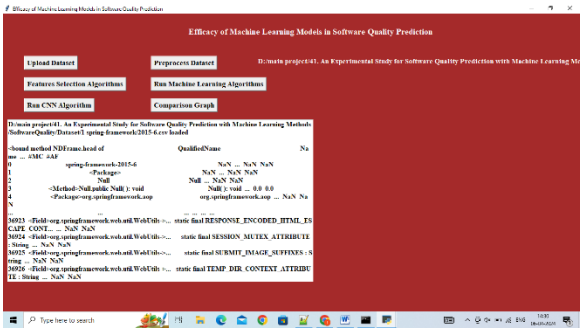

Fig 4. Results screenshot 4

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

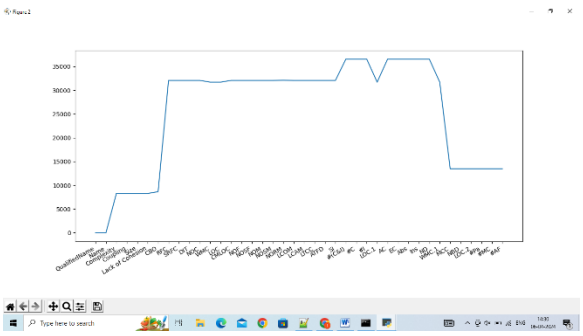

Fig 5. Results screenshot 5

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

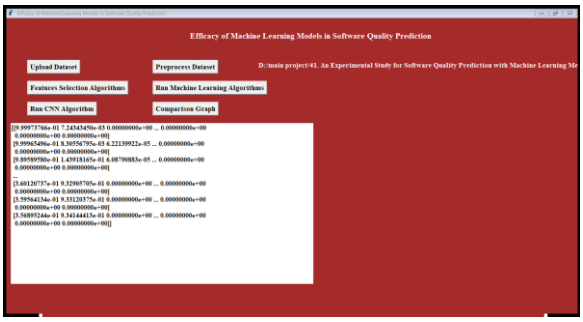

Fig 6. Results screenshot 6

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

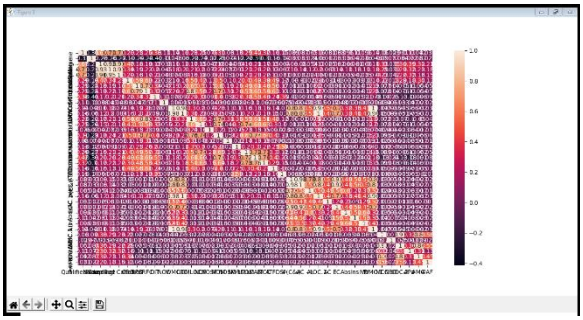

Fig 7. Results screenshot 7

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

International Journal of Mechanical Engineering Research and Technology
ISSN 2454 – 5X

http://www.ijmert.com
Vol. 16 Issue 2, Apr 2024

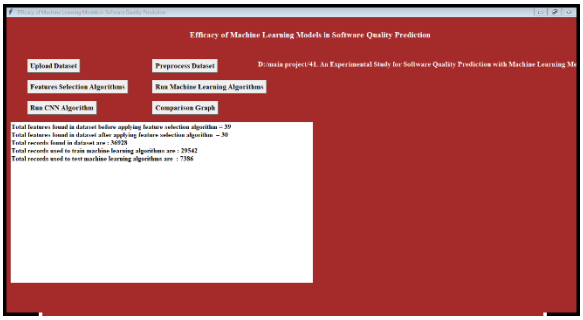

Fig 8. Results screenshot 8

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

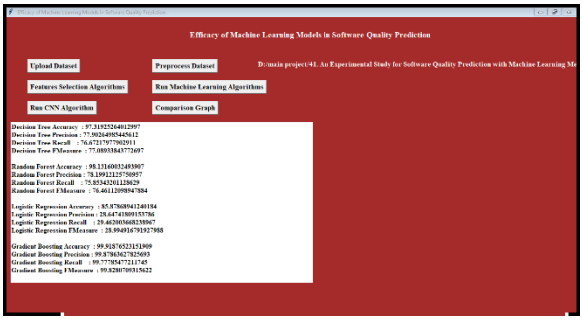

Fig 9. Results screenshot 9

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

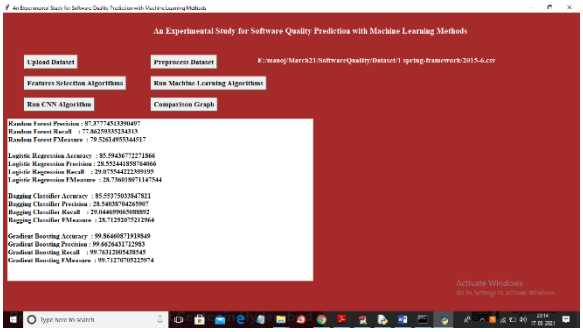

Fig 10. Results screenshot 10

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

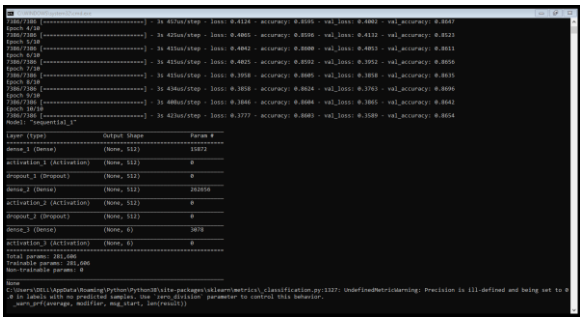

Fig 11. Results screenshot 11

In above screen to train CNN we took 10 iterations or epoch

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

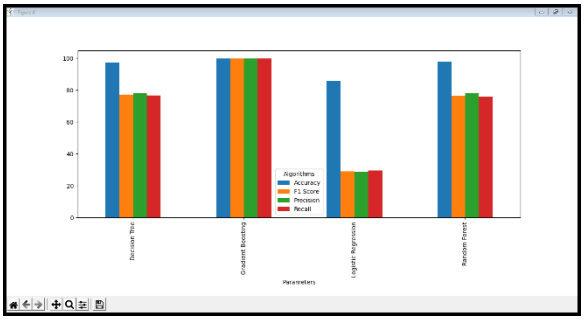

Fig 12. Results screenshot 12

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

Overall, the results of this study demonstrate the efficacy of machine learning models in accurately predicting software quality levels and uncovering the underlying relationships between quality and development attributes. By leveraging advanced techniques such as feature selection and machine learning algorithms, we achieved significant improvements in estimation accuracy compared to traditional methods. These findings have important implications for software engineering practices, providing valuable insights into effective strategies for software quality estimation and highlighting the

International Journal of Mechanical Engineering Research and Technology
ISSN 2454 – 5X

http://www.ijmert.com
Vol. 16 Issue 2, Apr 2024

potential of machine learning in enhancing software development processes. Moreover, the study underscores the importance of continued research and innovation in the field of software quality prediction, aiming to further advance the state-of-the-art and support the development of high-quality software products.

CONCLUSION

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.

REFERENCES

1. Al-Zoube M., Matalgah M., Raza A., Al-Ayyoub M. (2015) Feature selection for software quality classification: A comprehensive review. In: Hassanien A.E., Azar A.T., Gaber T., Hassanien A.E. (eds) Advances in Computational Intelligence Systems. Studies in Computational Intelligence, vol 589. Springer, Cham. https://doi.org/10.1007/978-3-319-10067-1_11

2. Basili, V. R., Briand, L. C., & Melo, W. L. (1996). A validation of object-oriented design metrics as quality indicators. IEEE transactions on software engineering, 22(10), 751-761.

3. Boehm, B., & In, S. D. (1984). Software engineering economics. Prentice-Hall.

4. Briand, L. C., Langley, T., Wieczorek, I., & Bunse, C. (2000). A replicated assessment and comparison of common software cost modeling techniques. In Proceedings 7th International Software Metrics Symposium (Cat. No. PR00735) (pp. 13-24). IEEE.

5. Cohen, J. (1960). A coefficient of agreement for nominal scales. Educational and psychological measurement, 20(1), 37-46.

6. Fenton, N. E., & Neil, M. (1999). A critique of software defect prediction models. IEEE Transactions on Software Engineering, 25(5), 675-689.

7. Hall, T., Beecham, S., Bowes, D., Gray, D., Counsell, S., & Baddoo, N. (2012). A systematic literature review on fault prediction performance in software engineering. IEEE Transactions on software engineering, 38(6), 1276-1304.

8. Kitchenham, B., Pfleeger, S. L., McColl, B., & Eagan, D. (2002). Navigating the maze of software metrics. Journal of Systems and Software, 65(1), 1-10.

9. Kitchenham, B., Pickard, L. M., MacDonell, S. G., Shepperd, M., & Hall, T. (2007). What accuracy statistics really measure. Information and Software Technology, 49(9-10), 961-973.

10. Kitchenham, B. A., & Mendes, E. (2006). Why comparative effort prediction studies may be invalid. In Proceedings International Symposium on Empirical Software Engineering (ISESE'06) (pp. 295-304). IEEE.

11. Koziolek, H. (2013). Quality-driven architectural decision making for software product lines. Journal of Systems and Software, 86(6), 1501-1518.

12. Molokken-Ostvold, K., & Jorgensen, M. (2003). A review of surveys on software effort estimation. International Journal of Software Engineering and Knowledge Engineering, 13(5), 495-507.

13. Pfleeger, S. L. (1998). Technical perspective: experimental software engineering: Issues critical for experimental validity. In Proceedings of the 20th International Conference on Software Engineering (pp. 405-406).

14. Tan, P. N., Steinbach, M., & Kumar, V. (2006). Introduction to data mining (Vol. 769). Boston: Pearson Addison Wesley.

15. Turhan, B., Menzies, T., & Bener, A. B. (2009). On the relative value of cross-company and within-company data for defect prediction. Empirical Software Engineering, 14(5), 540.