International Journal on Emerging Technologies **12**(2): 262-265(2021)

ISSN No. (Print) : 0975-8364
ISSN No. (Online) : 2249-3255

Machine Learning Algorithms for Software Assessment

Archana Chaudhary Thakur*
Assistant Professor, School of Computer Science & IT,
Devi Ahilya Vishwavidyalaya, Indore, (Madhya Pradesh), India.

(Corresponding author: Archana Chaudhary Thakur)

(Received 24 June 2021, Revised 22 July 2021, Accepted 24 August 2021)
(Published by Research Trend, Website: www.researchtrend.net)

ABSTRACT: Software maintainability is the extent to which a software can be understood, enhanced, or modified to adapt the present environment. It is an important quality measure according to ISO standards. It has been a major concern from decades and even in present times. Object-Oriented methodologies are used to develop several open-source software. Various researchers have employed statistical techniques on metric data excerpted from software to assess software quality. Various researchers have employed machine learning algorithms for fault prediction, reliability prediction, threshold estimation, but assessing software maintainability is still a great challenge now days. Machine learning algorithms can be explored to assess software maintainability. The contribution of the present work is to use machine learning algorithms for estimating software maintainability. In the present work machine learning algorithms are used for assessing the maintainability of an open-source software. The present work also emphasizes upon the relationships amongst different maintainability metrics.

Keywords: Object-Oriented methodologies, ISO, Machine learning, Open- Source Software, Maintainability, Design metrics.

I. INTRODUCTION

The ISO standards suggest software quality using eight features. One of the vital measures for assessing a software is software quality. According to ISO standards software maintainability is one of the most important quality features since decades. Object-oriented methodologies are used for creating various open-source software today. Therefore, in order to assess the quality of any open-source software, it is necessary to assess the maintainability of open-source software. When software applications are designed, the design metrics are attained. Design metrics can be employed as essential indicators to assess software maintainability [12]. Machine learning is a field of artificial intelligence [22]. Software applications improve themselves through experience using machine learning algorithms. Machine learning emphasizes on developing predictive models [18, 19]. These predictive models are used for training various software applications. Machine learning suit symbolize a set of application software that learn from a specified set of data and attain predictions on the novel dataset based upon its learning experience [20, 21]. Machine learning algorithms are trained with the help of an example dataset in order to make predictions on novel datasets [23]. Various categories of machine learning algorithms have achieved greater influence for example supervised learning, reinforcement learning, unsupervised learning, semi-supervised learning algorithms etc.

Machine learning algorithms are used on various diverse software applications for quality assessment [13,14]. Earlier various metrics were observed for different software applications for building predictive models. Different software metrics have also attained greater influence in maintainability prediction [9, 15, 16]. Various research works have also been conducted employing some design metrics from various suites. Software maintainability was also observed for entire software using entire metric suite [16]. Some of the well-known Object-Oriented metric suites are the MOOD suite, CK suite, and the Martin suite. In the present work, complete metric suite that is Martin suite of metrics is excerpted from a famous open-source charting application j free chart employing the J Depend tool [9]. Machine learning algorithms were then employed on the excerpted dataset. The present work establishes the relationships amongst design metrics recommended by Martin [8] and maintainability.

The present work is organized as follows. The related work is presented in Section II. The methodology used is described in Section III. Section IV presents results and discussions. Section V presents the conclusions drawn.

II. RELATED WORK

It was proposed that different metrics can be used as important measures for predicting maintainability [2]. The Maintainability Index (MI) is a software metric which evaluates the maintainability of a software code [17]. MI is a code metric exercised to assess the maintainability of a software application [2]. Various research works in the past used MI metrics and had attained significant attention [3]. The usage of MI metric was also validated in various research works [10]. The MI metric can be employed to verify the maintainability of existing code and was subsequently adapted to Object-Oriented paradigm also but with some restrictions [2]. The MI metric was validated and it was used practically in determining the maintainability of a software application [1]. It was also suggested that MI can be used to assess the quality of any source code. Machine learning predictive models were developed using design metrics and has attained great importance because of the benefits it offers. The designers can suggest changes in software design or code using machine learning predictive models and improve the software maintainability of their software applications [2]. The relationships amongst various metrics and software maintainability were assessed in [2]. It was observed that various size metrics have great impacton software maintainability [2]. Machine learning neural network algorithm was applied for approximating software quality using Object-Oriented metrics [4]. Modelling technique was suggested to develop maintainability prediction models employing various metrics excerpted from different software applications [5]. Artificial Neural Network algorithm was exercised for predicting maintainability using Object-Oriented metrics [6]. The use of different metrics for maintainability assessment using statistical methods was examined in literature [1, 2, 3]. A comparative analysis was conducted amongst three metric suites for fault prediction in Object-Oriented systems [11]. It was observed that the prediction models designed exercising the Martin suite were more correct as compared to models designed employing the CK and MOOD suites. But none of the above research works have exercised machine learning algorithms for assessing software maintainability. Hence the present work examines an open-source software jfreechart for software quality / maintainability assessment using machine learning algorithms.

III. METHODOLOGY USED

In the present work maintainability aspect is analyzed for jfreechart. It is a famous charting software popular amongst the software fraternity. The major focus of the present work is to examine an open-source java-based software that has grown with multiple versions [7]. This present work estimates the maintainability of an open-source software with the Maintainability Index (MI) metric [2]. MI is basically a source code metric which is exercised to assess the maintainability of a software [2]. In the present work the Martin metrics are considered as independent variables.

Machine learning algorithms are useful in predicting the maintainability of a software application. In the present work, the machine learning algorithms Linear Regression and Gradient Boosting Regression are used for assessing the maintainability. The methodology used in the present work is shown in steps below -

- Select Linear Regression algorithm for maintainability assessment.
- Input Independent variable Xis and evaluate respective Yi.
- In case of large errors use Gradient Boosting Regression algorithm to reduce the errors.

IV. RESULTS AND DISCUSSIONS

The MI values attained from different jfreechart versions had very indigent MI. In the present work various design metrics like concrete classes, abstract classes, afferent coupling, efferent coupling, abstractness, instability, distance and MI are examined for fifty jfreechart versions. It is an important observation that most of the jfreechart versions have less MI. It is also observed that MI decreases continuously from the first version till the last version.

A. Linear Regression

It is a linear model which assumes linear relationship amongst the input variables (X) and the single output variable (Y).The algorithm predicts a dependent variable value (Y) based upon independent variable(s) X. In other words, Y can be evaluated from a linear grouping of the input variables (X).When there exists only one input variable (X), the technique is termed as simple linear regression. When there exist numerous input variables, the method is termed as multiple linear regression. Various methods can be exercised to train or prepare the linear regression equation from data, the most common of which is called Ordinary Least Squares.

Fig. 1 shows simple graph between MI and Independent factor. The results obtained after applying the Linear Regression algorithm are shown in Fig. 1. The results in Fig. 1 are scattered that means the regression curve doesn't form a straight line which advocates that the model has many errors. Some goodness of fit test is conducted to zero-in upon the errors in the numerically quantified form. The Mean Squared Error(MSE) is observed as 5.09 and the Root Mean Squared Error (RMSE) value observed is 2.17. These values reflect the fact that there are many errors in the Linear Regression model.

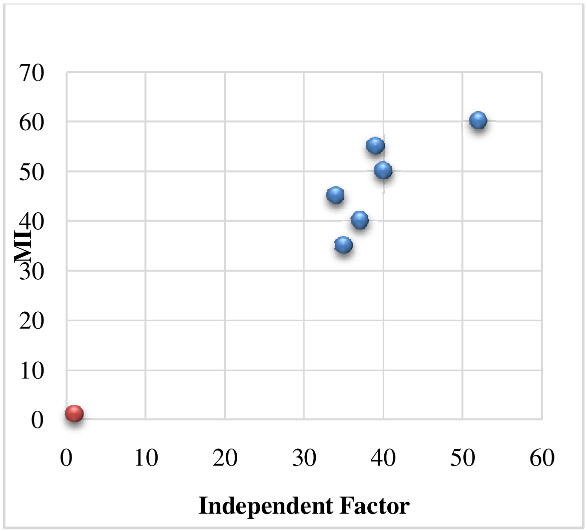

Fig. 1. Linear Regression Curve.

B. Gradient Boosting Regression Algorithm

Gradient Boosting Regression is one of the famous methods exercised with predictive algorithms. It is an effective algorithm which changes a comparatively unfortunate hypotheses problem into a very virtuous hypothesis. Back propagation algorithm is used to approximate or lessen errors. In the present work, this optimization algorithm was exercised to lessen the errors attained from Linear Regression model. After applying this algorithm, errors were lessened to a greater extent. The Mean Squared Error (MSE) observed was 2.30 and the Root Mean Squared Error (RMSE) value observed was 1.10. So, the errors attained using this model were significantly less as compared to the Linear Regression model.

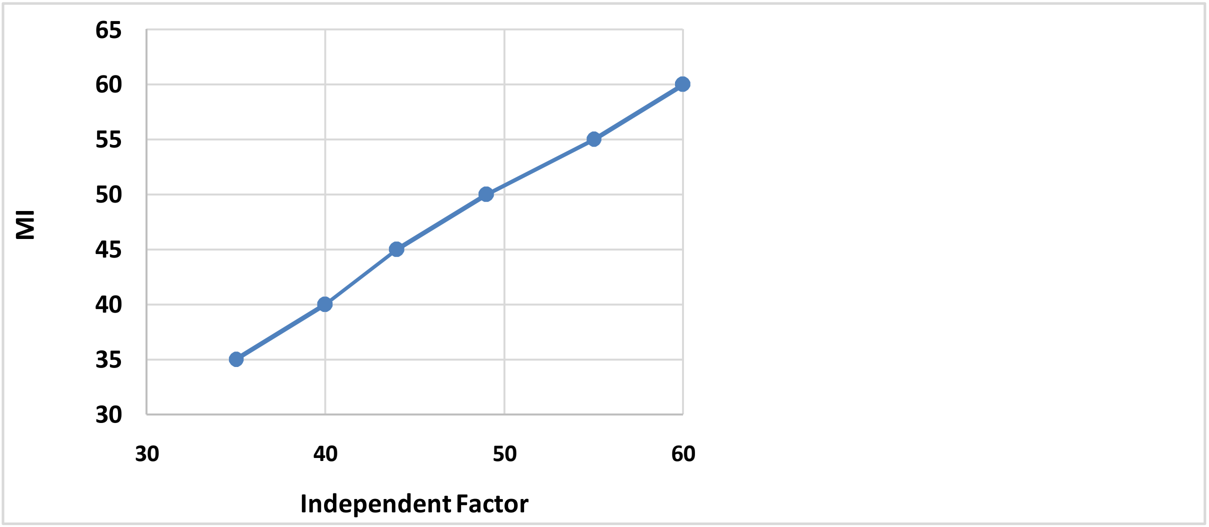

Fig. 2. After applying Gradient Boosting Regression.

In order to minimize the errors generated from Linear Regression, Gradient Descent Boosting algorithm is applied. It is clear from Fig. 2 that leaving two cases (slightly deviated from the straight line), a significant model is generated that implies the errors are minimized to a greater extent. The MSE attained now is 2.36 and the RMSE valueis1.12.The errors attained using Gradient Descent Boosting model are certainly less as compared to the Linear Regression model (Fig. 1).

V. CONCLUSIONS

Software quality assessment for maintainability feature was conducted using machine learning algorithms viz. Linear Regression, Gradient Descent Boosting. The work was conducted for assessing maintainability for open-source software. The results showed that many errors existed in Linear Regression model. The errors were significantly reduced after applying Gradient Boosting Regression model. Hence the future scope of the work could be exploring machine learning algorithms for assessing other software quality measures such as portability, scalability, usability, flexibility and much more.

REFERENCES

[1]. Welker, K.D. (2001). The software maintainability index revisited. Journal of DefenseSoftwareEngineering,18-21.

[2]. Oman, P., & Hagemeister, J., (1992). Metrics for assessing a software system's maintainability. In: Proceedings of the IEEE International Conference on Software Maintenance, 337-344.

[3]. Oman, P., & Hagemeister, J.(1994). Constructing and testing of Polynomials predicting software maintainability. Journal of Systems and Software, *24*(3), 251-266.

[4]. Thwin, M.M.T., & Quah, T-S. (2005). Application of neural networks for software quality prediction using object-oriented metrics. *Journal of Systems and Software*, *76*(2),147-156.

[5]. Zhou, Y., & Lueng,H.(2007). Predicting object-oriented software Maintain ability using multivariate adaptive regressionsplines. *Journal of Systems and Software*, *80*(8),1349-1361.

[6]. Aggarwal, K., Singh, Y., Kaur, A. & Malhotra, R.(2008). Application of artificial neural network for predicting maintainability using object- oriented metrics. *World Academy of Science, Engineering and Technology, 2*, 3552-3556.

[7]. Loy, M., Niemeyer, P., & Leuck, D. (2020). Learning Java. O'Reilly Media, Inc Publishers.

[8]. Martin, R. (2003). Agile Software Development Principles: Principles, Patterns and Practices, Prentice Hall.

[9]. Emanuel, A.W. R., Wardoyo, R., & Istiyanto, J. E.(2011). Modularity Index Metrics for Java-Based Open Source Software Projects. *International Journal of Advanced Computer Science and Applications*, *2*(11), 52-58.

[10]. Halstead, M. H. (1977). Elements of Software Science. Elsevier Science Ltd.

[11]. Elish, M.O., Al-Yafei, A.H.A., & Al-Mulhem, M.S. (2011). Empirical comparison of three metrics suites for fault prediction in packages of object-oriented systems: a case study of eclipse. Advances in Engineering Software, *42*(10), 852-859.

[12]. IEEE Standard Glossary of Software Engineering Terminology, Report IEEE Std 610.12-1990, IEEE, 1990.

[13]. Malhotra, R., & Bansal, A. (2012). Fault Prediction Using Statistical and Machine Learning Methods for Improving Software Quality. JIPS, *8*, 241-262.

[14]. Jaiswal, A.,&Malhotra, R. (2018). Software reliability prediction using machine learning techniques. *International Journal of System Assurance Engineering and Management*, *9*(1),230-244.

[15]. Padhy, N.,Panigrahi, R., & Neeraja, K. (2021). Threshold estimation from software metrics by using evolutionary techniques and its proposed algorithms, models. Evolutionary Intelligence,1-15.

[16]. Madhwaraj K.G., & Amirthavalli, M. (2019). Applying Machine Learning Techniques to Predict the

Maintainability of Open Source Software. *International Journal of Engineering and Advanced Technology*, *8*(5S3), 192-195.
[17]. Madhwaraj K.G., & Amirthavalli, M. (2019). Applying Machine Learning Techniques to Predict the Maintainability of Open Source Software. *International Journal of Engineering and Advanced Technology, 8*(5S3), 192-195.
[18]. Chaudhary, A., Kolhe, S., & Kamal, R. (2016). A hybrid ensemble for classification in multiclass datasets: An application to oil seed disease dataset. *Computers and Electronics in Agriculture*, *124*, 65–72.
[19]. Chaudhary, A., Kolhe, S., & Kamal, R. (2016). An improved random forest classifier for multi-class classification. *Information Processing in Agriculture*, *3*(4), 215-222.
[20]. Chaudhary, A., Kolhe, S., & Kamal, R. (2020). A particle swarm optimization based ensemble for vegetable crop disease recognition. *Computers and Electronics in Agriculture*, *178*, 1-7.
[21]. Thakur, A., & Thakur, R. (2018). Machine Learning Algorithms for Intelligent Mobile Systems. International *Journal of Computer Sciences and Engineering*, *6*(6), 1257–1261.
[22]. Chaudhary, A. (2020). Performance Enhancement Method for Machine Learning Algorithm. International *Journal of Innovative Technology and Exploring Engineering*, *9*(11), 320-322.
[23]. Thakur, A. (2020). Crop Disease Recognition using Machine Learning Algorithms. *International Journal of Innovative Technology and Exploring Engineering, 9*(11), 164-166.

How to cite this article: Thakur A. C. (2021). Machine Learning Algorithms for Software Assessment. *International Journal on Emerging Technologies, 12*(2): 262–265.