Predicting Software Maintainability in Object-Oriented Systems Using Ensemble Techniques

Hadeel Alsolai
[1]Computer Science and Information system
Princess Nourah Bint Abdulrahman University
Riyadh, Saudi Arabia
[2]Computer and Information Sciences
University of Strathclyde
Glasgow, United Kingdom
hadeel.alsolai@strath.ac.uk

Supervisor :Dr.Marc Roper
Computer and Information Sciences
University of Strathclyde
Glasgow, United Kingdom
Marc.roper@strath.ac.uk

Co-Supervisor :Dr.Dua' Nassar
Computer Science and Information system
Princess Nourah Bint Abdulrahman University
Riyadh, Saudi Arabia
DANassar@pnu.edu.sa

Abstract— Prediction of the maintainability of classes in object-oriented systems is a significant factor for software success, however it is a challenging task to achieve. To date, several machine learning models have been applied with variable results and no clear indication of which techniques are more appropriate. With the goal of achieving more consistent results, this paper presents the first set of results in an extensive empirical study designed to evaluate the capability of bagging models to increase accuracy prediction over individual models. The study compares two major machine learning based approaches for predicting software maintainability: individual models (regression tree, multilayer perceptron, k-nearest neighbors and m5rules), and an ensemble model (bagging) that are applied to the QUES data set. The results obtained from this study indicate that k-nearest neighbors model outperformed all other individual models. The bagging ensemble model improved accuracy prediction significantly over almost all individual models, and the bagging ensemble models with k-nearest neighbors as a base model achieved superior accurate prediction in both datasets. This paper also provides a description of the planned programme of research which aims to investigate the performance over various datasets of advanced (ensemble-based) machine learning models.

Keywords— individual models, bagging ensemble model, software maintainability, prediction, Object-oriented systems.

I. INTRODUCTION

Prediction of software maintainability has received much attention in recent years due to its essential role in managing maintenance resource and controlling future maintenance effort. It is also an essential task supporting software quality assurance activities. However, the development of an accurate model to predict software maintainability of OO system is difficult to accomplish.

Utilization of individual machine learning models has been investigated in several studies to predict software maintainability[1-8]. Recently, ensemble models have been applied across a wide range of software engineering problem domains, such as fault prediction, to increase accuracy prediction over individual models [9]. Nevertheless, far too little attention has been paid to use ensemble models in software maintainability domain.

This programme of research adopts an empirical approach to evaluate the performance of bagging ensemble models against individual models for predicting software maintainability of OO systems. To date we have carried out two experiments to compare individual models, namely, regression tree (RT), multilayer perceptron (MLP), k-nearest neighbors (KNN) and m5rules (M5Rules), and a bagging ensemble model. These models are evaluated on the well-known public dataset collected from OO system maintenance activities, namely QUES (Quality Evaluation System). To the best of our knowledge, this the only study that investigates the capability of bagging ensembles on the QUES datasets. However, these are relatively old datasets and the aim of this research is to further explore the use ensemble approaches on more recent datasets and systems.

The remainder of this paper is organized as follows: Section II describes related work and summarizes research conducted in software maintenance prediction. Section III discusses the research questions and contributions. Section IV describes the individual prediction models that we used as a base model for the bagging ensemble. Section V presents the bagging ensemble models. Section VI explains the initial empirical study design. Section VII demonstrates results and answers the research questions. Section VIII defines the planned proggrame of research. The authors conclude this paper with a summary and lessons learned in Section IX.

II. RELATED WORK

Key to the prediction of software maintainability is the determination of software maintenance measurements which are notoriously hard to capture due to the problems in estimating the effort associated with the maintenance task. Software maintenance effort depends on various aspects of software maintenance operation, such as corrective, adaptive, emergency or perfective [10], and is a function of the subtle interplay between the maintenance change and the software being modified. A considerable amount of literature has investigated different types of software maintenance

measurement: corrective maintenance effort [11]; adaptive maintenance effort [12]; maintainability index [13] and maintenance time[14]. A common maintenance effort measure used in many studies is based on changes made in the maintenance process and determines maintenance effort by computing the number of modifications made per class during the maintenance period[1-7, 15-17]. A higher number of changes indicates greater maintenance effort.

A range of OO metrics have been used in the various studies to try and capture the concept of software maintainability: Chidamber and Kemerer [18, 19]; Oman and Hagemeister [20]; Li and Henry [21]; Coleman and Ash et al [22]; Welker and Oman [23]; Genero and Piattini et al [24]; Misra [13]; Yuming and Baowen [25]. Many of these metrics have been validated only in a limited number of studies, and some of them have been published but never applied. Several studies have found a robust relationship between software maintainability and OO metrics[1-3, 13, 26, 27], although Thwin and Quah [26] conclude that this relationship is complicated, nonlinear and low accuracy.

Among the wide variety of OO metrics, Chidamber and Kemerer (C&K) [18, 19] provided a foundation of OO metrics by introducing six metrics: depth of the inheritance tree (DIT), number of children (NOC), response for a class (RFC), lack of cohesion of methods (LCOM), coupling between objects (CBO) and weighted method per class (WMC). Li and Henry [21] extended their work by using all C&K metrics except CBO and introducing additional independent ones: message-passing coupling (MPC), abstract data type (ADT), number of methods (NOM), lines of code (LOC), number of semicolons in a class (SIZE1) and number of properties (SIZE2), as well as CHANGE metrics as the dependent variable to predict software maintainability by calculating the number of lines changed in the class during maintenance process.

A number of studies have employed individual machine learning models to predict software maintainability based on historical data of OO systems on the QUES dataset specifically: Bayesian network [1]; Multivariate adaptive regression splines [2]; TreeNet [3]; Mamdani-based model [4]; Group Method of Data Handling model [5]; Artificial neural network and genetic algorithm [6]; Neuro-Genetic algorithm [7]; Hybrid neural network and fuzzy logic approach [8]. Furthermore, two research studies have investigated the application of ensemble models in software maintainability: Aljamaan et al. [15] developed a heterogeneous ensemble model by selecting the best base model in training; and Elish et al. [17] evolved their work by generalizing a heterogeneous ensemble to include averaging, weighted averaging and best base in training as well. The results confirm that both of these heterogeneous ensemble models are at least as accurate as individual ones.

The most obvious gap from previous studies is that a wide variety of individual models were constructed to predict software maintainability, while only two studies were performed using heterogenous ensemble models (i.e. those which combine different types of models). Furthermore, homogeneous ensemble models (which combine models of the same type) have never been used in any study that employed on QUES dataset. Overall, it is clear that the application of ensemble models in predicting software maintainability is limited and there is no evidence of the performance of homogenous models on the QUES dataset specifically. Consequently, we select bagging ensemble models, because it is recommended to apply on the small training set, such as QUES dataset, to decrease the variance between the base models that causes the unstable problem [28].

III. RESEARCH QUESTIONS AND CONTRIBUTION

The primary objective is to predict software maintainability accurately by evaluating the capability of bagging ensemble models to increase or decrease accuracy prediction over individual models. Therfore, our objective in this paper focus on answering the following research questions:

RQ1) What is the best performing individual model to predict software maintainability?

RQ2) How much can a bagging ensemble model increase or decrease the performance of individual models?

In order to make a contribution in both the machine learning and software maintainability fields, it is necessary to investigate the capability of the various machine learning models on a range of datasets. Therefore, homogeneous ensemble models (bagging) and some novel individual models (KNN and M5Rules) are chosen in our empirical study.

IV. INDIVIDUAL PREDICTION MODELS

This section presents a summary description of the most well-known regression models that have been applied to various machine learning domain. Some of these models, such as RT and KNN are considered amongst the best ten data mining models [29], while MLP is the best neural networks in term of extreme learning machine [30]. M5rules is categorized from the tree that generates a set of rules accurately [31]. In our empirical study, we used these models as a base model for the bagging ensemble model.

- **Regression Tree (RT)** is constructed by using the binary recursive partitioning process that divides dataset recursively into partitions or branches by selecting at each stage the independent variables that have the lowest minimum sum of the squared deviations from the mean [32]. RT is an unstable algorithm as any small changes in the training set can lead to considerable changes in the model's prediction [33].
- **Multilayer Perceptron (MLP)** is an artificial neural network built from several layers of nodes (neurons) that combine weighted inputs and generate an output defined by a non-linear activation function. The MLP uses backpropagation to construct the neural model from historical training data [34].

- **K-Nearest Neighbors (KNN)** is computed by choosing the closest neighbors in the training data to generate the target data. The algorithm categorizes data by selecting the majority class of k the closest neighbors in the training set based on Euclidean (or other specified) distance [35].
- **M5rules(M5Rules)** is built by creating a decision list for regression problems to apply a separate-and-conquer strategy. This is an iterative process that constructs a model tree by utilizing the M5 algorithm and selecting the best leaf to transform into a rule. M5Rules differs slightly from the RT in that M5Rules has a regression model in its nodes to predict a value, while the RT has only a constant fitted mean in its nodes, therefore each one can be used to make different predictions [31].

V. ENSEMBLE PREDICTION MODELS

This section provides a brief clarification of the bagging ensemble model used in our empirical study. The bagging ensemble model is one of the most common ensemble models that combine a set of multiple models to generate a final prediction. Because the QUES dataset has limited records it can lead to an unstable model so we decided to perform bagging which can address this situation [28].

- **Bootstrap aggregating (Bagging)** is an ensemble technique to enhance model prediction by integrating several models of the same type. The bagging algorithm starts by generating further data for training from an initial training set iteratively with the replacement, and creates an equal weight of the models. After that, the bagging algorithm combines the results of these models by using voting or averaging. Bagging can improve the prediction of unstable models, such as RT and is also recommended to use with data training sets, (such as QUES) to decrease the variance between the base models [28]. Three parameters are required to be determined in the bagging algorithm:
- Base model: the individual model to be used as a base model in the bagging algorithm.
- Ensemble size: the number of the individual models to be created in the bagging algorithm.
- Training set size: the size for each bagging algorithm as a percentage of the training set [36].

VI. INITIAL EMPIRICAL STUDY DESIGN

The primary goal of this empirical study is to investigate the efficiency of the bagging ensemble model in terms of accurate prediction as compared to an individual model. To accomplish this goal, we carried out two experiments to predict software maintainability as follows:

- The first experiment compares different well-known individual regression models (RT, MLP, KNN, and M5Rules) to compare their accuracy.
- The second experiment evaluates the accuracy of homogeneous ensemble models against each individual model in the first experiment using individual regression models from the first expirement as a base model.

A. Dataset

We utilized the QUES object-oriented software datasets proposed by Li and Henry [10]. QUES is publicly available, accurate, validated and widely used in software maintainability prediction studies. Based on the Classic-Ada tool the QUES dataset consists of class-level metrics data was gathered from 71 classes. The independent variables consist of five of the metrics published by Chidambar and Kemerer [19]: WMC, DIT, NOC, RFC, and LCOM, along with four metrics published by Li and Henry [10]: MPC, DAC, NOM, SIZE2, and one traditional lines of code size metric SIZE1. The dependent metric is CHANGE that measures software maintainability by identifying the number of lines changed per class from 3 years of software maintenance. A high number of changes refers to the high maintenance effort [2].

B. Prediction accuracy measures

Since OO software maintainability prediction is a regression problem, we used the de facto standard prediction accuracy measures that well-known and most frequently used in regression problem, namely: the magnitude of relative error (MRE), pred measures (Pred (q)) and absolute residual (Ab.Res). In addition, several evaluation measures are used in this paper generated from the above equations, such as, the standard deviation of the absolute residuals (SD. Ab.Res) that computes the amount of variation, and maximum number of magnitude of relative error (MAX. MRE).

Ten-fold cross-validation is used in this paper to evaluate and compare between prediction models and R is used to build these models [37].

VII. RESULTS AND ANALYSİS

Table I summarizes the results obtained from applying the individual and bagging models on QUES dataset. Bold values in the table indicate the best results among each experiment, which are the lowest (MAX. MRE, MMRE and SD. Ab.Res) or the highest (Pred(.25) and Pred(.30)) depending on the measure. Among individual models, KNN achieved the best MMRE value that reaches to (0.43), while MLP and RT were the worst MMRE value. In addition, KNN produced significantly better Pred(.25), Pred(.30) and SD. Ab.Res than other individual models. However, M5Rules achieved the best MAX. MRE value. After building a bagging ensemble on each individual model, KNN outperformed all others models in all accuracy predictions except SD. Ab.Res. Overall, the findings from Table I reports that the bagging ensemble model enhances the accuracy over the individual models. These findings confirmed with the previous study, showing that the bagging ensemble models have a high potential to build more accurate predictions than individual models [28], [38]. The

notable improvement of prediction accuracy in individual model verified the bagging ensemble model improves performance effectively when it applies to the limited amount of datasets, such as in this case the QUES dataset, which has only 71 rows [28].

TABLE I: THE SUMMARY RESULT OF QUES DATASET.

Model Name	Individual models				Bagging ensemble model			
	RT	MLP	KNN	M5rules	Bag (RT)	Bag (MLP)	Bag (KNN)	Bag (M5rules)
MAX. MRE	6.59	7.36	2.86	**1.94**	1.77	0.67	**0.54**	1.18
MMRE	0.68	0.76	**0.43**	0.55	0.30	0.20	**0.10**	0.33
Pred (.25)	0.32	0.25	**0.56**	0.28	0.56	0.72	**0.90**	0.51
Pred (.30)	0.39	0.38	**0.59**	0.37	0.69	0.77	**0.93**	0.62
SD. Ab.Res	18.91	32.15	**18.52**	19.99	16.04	**8.48**	12.35	13.68

Fig. 1 illustrates a histogram of Pred(.25) to investigate the comparison between individual models and bagging ensemble models on QUES dataset. The results from Fig. 1 provide confirmatory evidence that the bagging model increased the performance dramatically overall individual models.

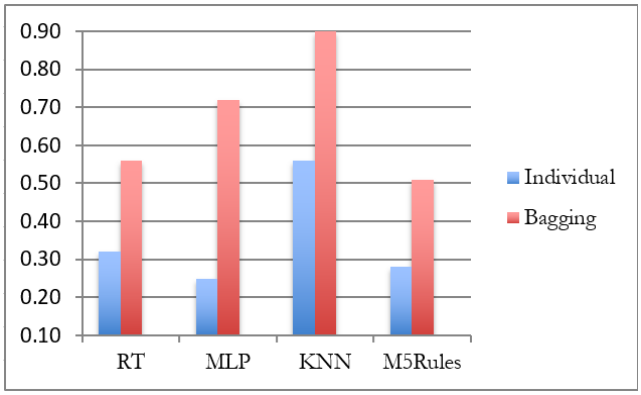

Fig. 1. Pred (.25) for each models apply on QUES dataset.

Fig. 2 employs a box plot of MRE to enable a visual comparison of different prediction models in QUES dataset. The upper line and lower line of the box represents "whiskers". The middle horizontal line across box presents middle quartile, which is MMRE value for prediction models. The upper quartile is the line between upper whiskers and middle quartile, where the lower quartile is the line between lower whiskers and middle quartile. It clearly demonstrates the improved accuracy achieved by applying bagging ensemble models on various individual models.

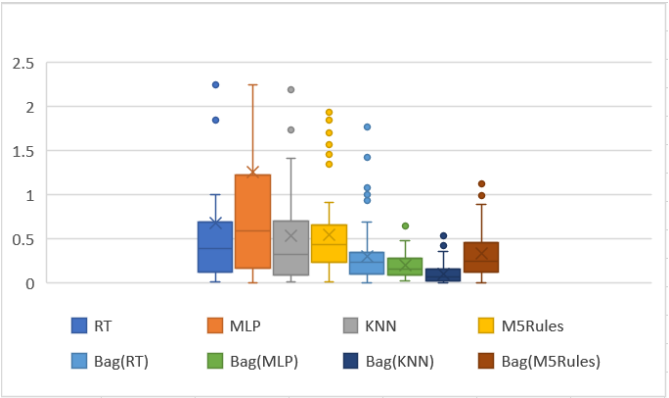

Fig. 2. Box plot of MRE for models in QUES dataset.

VIII. RESEARCH PLAN

Further investigation of the prediction of software maintainability is planned using the research plan, as described below:

- Identify and source a number of representative data sets for software maintainability either from open source data sets hosted in PROMISE [39] (for example) or by extraction from exiting open source projects proposed in repositories such as Github or SourceForge.
- Construct a range of advanced machine learning models (i.e. homogenous and heterogenous ensemble models) from existing individual models to predict software maintainability.
- On the identified datasets, compare the prediction performance between the ensemble models over individual models, and study the gained performance using the most commonly used evaluation measures in software maintainability such as accuracy and f-measure for classification problems, and MMRE and pred for regression problems.
- Draw conclusions from the conducted empirical studies to evaluate the prediction performance of machine learning models in software maintainability.

IX. CONCLUSION

Prior studies have documented the effectiveness of machine learning models in predicting software maintainability of object-oriented systems. However, these studies have used a wide variety of individual models and paid limited attention on ensemble models. This paper has identified a programmed of research to investigate these approaches and presented the results of an initial empirical study to assess the impact of bagging ensemble models over individual models in predicting software maintainability.

The results concluded that among individual models, KNN has achieved the best prediction accuracy in QUES dataset, and the bagging ensemble model has improved the prediction accuracy magnificently over almost all individual models. The bagging ensemble with KNN as a base model particularly has outperformed all other models in both datasets. These findings extend our knowledge of the capability of ensemble model to advance accuracy prediction of individual models and provide a basis for exploring this on a wider range of datasets.

ACKNOWLEDGEMENTS

The authors would like to acknowledge Princess Nourah bint Abdulrahman University for supporting various development and providing several facilities in this research paper.

REFERENCES

[1] C. van Koten and A. R. Gray, "An application of Bayesian network for predicting object-oriented software maintainability," Information and Software Technology, vol. 48, pp. 59-67, 1// 2006.

[2] Y. Zhou and H. Leung, "Predicting object-oriented software maintainability using multivariate adaptive regression splines," Journal of Systems and Software, vol. 80, pp. 1349-1361, 8// 2007.

[3] M. O. Elish and K. O. Elish, "Application of TreeNet in Predicting Object-Oriented Software Maintainability: A Comparative Study," in 2009 13th European Conference on Software Maintenance and Reengineering, 2009, pp. 69-78.

[4] M. A. Ahmed and H. A. Al-Jamimi, "Machine learning approaches for predicting software maintainability: a fuzzy-based transparent model," IET Software, vol. 7, pp. 317-326, 2013.

[5] R. Malhotra and A. Chug, "Application of Group Method of Data Handling model for software maintainability prediction using object oriented systems," International Journal of System Assurance Engineering and Management, vol. 5, pp. 165-173, 2014.

[6] L. Kumar and S. Rath, "Predicting Object-Oriented Software Maintainability using Hybrid Neural Network with Parallel Computing Concept," in Proceedings of the 8th India Software Engineering Conference, 2015, pp. 100-109.

[7] L. Kumar, D. K. Naik, and S. K. Rath, "Validating the Effectiveness of Object-Oriented Metrics for Predicting Maintainability," Procedia Computer Science, vol. 57, pp. 798-806, 2015/01/01 2015.

[8] L. Kumar and S. K. Rath, "Software maintainability prediction using hybrid neural network and fuzzy logic approach with parallel computing concept," International Journal of System Assurance Engineering and Management, vol. 8, pp. 1487-1502, November 01 2017.

[9] C. W. Yohannese, T. Li, M. Simfukwe, and F. Khurshid, "Ensembles based combined learning for improved software fault prediction: A comparative study," in 2017 12th International Conference on Intelligent Systems and Knowledge Engineering (ISKE), 2017, pp. 1-6.

[10] "IEEE Standard for Software Maintenance," IEEE Std 1219-1993, pp. 1-45, 1993.

[11] A. De Lucia, E. Pompella, and S. Stefanucci, "Assessing effort estimation models for corrective maintenance through empirical studies," Information and Software Technology, vol. 47, pp. 3-15, 1/1/ 2005.

[12] F. Fioravanti and P. Nesi, "Estimation and prediction metrics for adaptive maintenance effort of object-oriented systems," IEEE Transactions on Software Engineering, vol. 27, pp. 1062-1084, 2001.

[13] S. C. Misra, "Modeling Design/Coding Factors That Drive Maintainability of Software Systems," Software Quality Journal, vol. 13, pp. 297-320, 2005.

[14] R. K. Bandi, V. K. Vaishnavi, and D. E. Turk, "Predicting maintenance performance using object-oriented design complexity metrics," IEEE Transactions on Software Engineering, vol. 29, pp. 77-87, 2003.

[15] H. Aljamaan, M. O. Elish, and I. Ahmad, "An Ensemble of Computational Intelligence Models for Software Maintenance Effort Prediction," in Advances in Computational Intelligence: 12th International Work-Conference on Artificial Neural Networks, IWANN 2013, Puerto de la Cruz, Tenerife, Spain, June 12-14, 2013, Proceedings, Part I, I. Rojas, G. Joya, and J. Gabestany, Eds., ed Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 592-603.

[16] S. K. Dubey, A. Rana, and Y. Dash, "Maintainability prediction of object-oriented software system by multilayer perceptron model," SIGSOFT Softw. Eng. Notes, vol. 37, pp. 1-4, 2012.

[17] M. O. Elish, H. Aljamaan, and I. Ahmad, "Three empirical studies on predicting software maintainability using ensemble methods," Soft Computing, vol. 19, pp. 2511-2524, 2015.

[18] S. R. Chidamber and C. F. Kemerer, Towards a metrics suite for object oriented design vol. 26: ACM, 1991.

[19] S. R. Chidamber and C. F. Kemerer, "A metrics suite for object oriented design," IEEE Transactions on software engineering, vol. 20, pp. 476-493, 1994.

[20] P. Oman and J. Hagemeister, "Metrics for assessing a software system's maintainability," in Software Maintenance, 1992. Proceedings., Conference on, 1992, pp. 337-344.

[21] W. Li and S. Henry, "Object-oriented metrics that predict maintainability," The Journal of Systems & Software, vol. 23, pp. 111-122, 1993.

[22] D. Coleman, D. Ash, B. Lowther, and P. Oman, "Using metrics to evaluate software system maintainability," Computer, vol. 27, pp. 44-49, 1994.

[23] K. D. Welker, P. W. Oman, and G. G. Atkinson, "Development and application of an automated source code maintainability index," Journal of Software: Evolution and Process, vol. 9, pp. 127-159, 1997.

[24] M. Genero, M. Piattini, E. Manso, and G. Cantone, "Building UML class diagram maintainability prediction models based on early metrics," in Software Metrics Symposium, 2003. Proceedings. Ninth International, 2003, pp. 263-275.

[25] Y. Zhou and B. Xu, "Predicting the maintainability of open source software using design metrics," Wuhan University Journal of Natural Sciences, vol. 13, pp. 14-20, February 01 2008.

[26] M. M. T. Thwin and T.-S. Quah, "Application of neural networks for software quality prediction using object-oriented metrics," Journal of Systems and Software, vol. 76, pp. 147-156, 5// 2005.

[27] C. Jin and J.-A. Liu, "Applications of support vector mathine and unsupervised learning for predicting maintainability using object-oriented metrics," in Multimedia and Information Technology (MMIT), 2010 Second International Conference on, 2010, pp. 24-27.

[28] M. Skurichina and R. P. Duin, "Bagging for linear classifiers," Pattern Recognition, vol. 31, pp. 909-930, 1998.

[29] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, et al., "Top 10 algorithms in data mining," Knowl. Inf. Syst., vol. 14, pp. 1-37, 2007.

[30] J. Tang, C. Deng, and G. B. Huang, "Extreme Learning Machine for Multilayer Perceptron," IEEE Transactions on Neural Networks and Learning Systems, vol. 27, pp. 809-821, 2016.

[31] G. Holmes, M. Hall, and E. Prank, "Generating Rule Sets from Model Trees," Berlin, Heidelberg, 1999, pp. 1-12.

[32] W.-Y. Loh, "Classification and regression trees," Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, vol. 1, pp. 14-23, 2011.

[33] R.-H. Li and G. G. Belford, "Instability of decision tree classification algorithms," in Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, 2002, pp. 570-575.

[34] J. B. Bradley, "Neural networks: A comprehensive foundation: S. HAYKIN. New York: Macmillan College (IEEE Press Book) (1994). v + 696 pp. ISBN 0-02-352761-7," Information Processing & Management, vol. 31, p. 786, 1995/09/01/ 1995.

[35] D. W. Aha, D. Kibler, and M. K. Albert, "Instance-based learning algorithms," Machine Learning, vol. 6, pp. 37-66, January 01 1991.

[36] L. Breiman, "Bagging Predictors," Machine Learning, vol. 24, pp. 123-140, August 01 1996.

[37] R. Ihaka and R. Gentleman, "R: A Language for Data Analysis and Graphics," Journal of Computational and Graphical Statistics, vol. 5, pp. 299-314, 1996/09/01 1996.

[38] D. W. Opitz and R. Maclin, "Popular ensemble methods: An empirical study," J. Artif. Intell. Res.(JAIR), vol. 11, pp. 169-198, 1999.

[39] T. Menzies, Krishna, R., Pryor, D. (2016). The Promise Repository of Empirical Software Engineering Data. Available: http://openscience.us/repo.