Evaluating Imputation Methods to Improve Data Availability in a Software Estimation Dataset

Sreekumar P. Pillai, T. Radha Ramanan, S.D. Madhu Kumar

Abstract--- Missing of partial data is a problem that is prevalent in most of the datasets used for statistical analysis. In this study, we analyzed the missing values in ISBSG R1 2018 dataset and addressed the problem through imputation, a machine learning technique which can increase the availability of data. Additionally, we compare the performance of three imputation methods: Classification and Regression Trees (CART), Polynomial Regression (PR), Predictive Mean Matching (PMM), and Random Forest (RF) applied to ISBSG R1 2018 dataset available from International Standards Benchmarks Group. Through imputation, we were able to increase data availability by four times. We also evaluated the performance of these methods against the original dataset without imputation using an ensemble of Linear Regression, Gradient Boosting, Random Forest, and ANN. Imputation using CART can increase the availability of the overall dataset but only at the loss of some predictive capability of the model. However, CART remains the option of choice to extend the usability of the data by retaining rows that are otherwise removed from the dataset in traditional methods. In our experiments, this approach has been able to increase the usability of the original dataset to 63%, but with 2 to 3% decrease in its overall predictive performance.

Keywords--- Software Effort Estimation, Software Cost Estimation, Effort Prediction, Gradient Boosting Machines, Generalized Linear Model, Artificial Neural Networks, Random Forests, Missing Data Imputation, Ensemble Models.

I. INTRODUCTION

Missing of partial data is a problem that is prevalent in most of the datasets used for statistical analysis. Most of the regression and machine learning techniques cannot handle missing data. Traditionally, the approach has been to remove rows containing missing values altogether. This creates further challenges, especially in a data-sparse domain like Software Effort estimation. The second problem is that the removal of rows creates bias in the dataset, especially if the missing values are not at random (1). Datasets collated by non-profit organizations also contain a considerable percentage of missing values (2), and it is pertinent to solve this problem from the perspective of other domains too.

Estimating Software Effort accurately has always been a challenge to project managers and research scholars. In the last couple of decades, many methods have been put up by research scholars and scientist's hat could be classified as Algorithmic models, Expert Judgment models, and AI-based models (3). In the last decade, ANN has come to the forefront in software estimation in terms of accuracy. The effort to develop a software product is a function of many factors, such as its size and many environmental factors.

In a previous study that we conducted, we had arrived at a minimal variable subset from literature research and subsequently undertaken a comparison of the output of estimation models using these subsets. All these methods, except manual methods, expect the dataset to be complete with no missing values.

While software effort estimation is critical an activity, research still struggles to get hold of data of adequate quality to develop accurate models. Many companies fail to do a precise estimate or keep a record of the actual effort consumed in a project. Availability of reliable data is very poor in the Software Engineering domain, and scholars and practitioners struggle to get hold of data.

To address this challenge, the International Software Benchmark Group (ISBSG) provides the Software Project dataset for research and development. We use ISBSG R1 2018 dataset for our analysis. Even with the minimal variable subset, we identified from the ISBSG dataset; we would be able to use only 15% of the original dataset by removing rows with NA values.

In this study, we analyzed the missing values in the ISBSG dataset through imputation using three different methods: Classification and Regression Trees (CART), Polynomial Regression (PR), Predictive Mean Matching (PMM), and Random Forest (RF).

We went further to evaluate all these approaches through an ensemble model of Generalized Linear Regression, Gradient Boosting, Random Forest, and ANN. Our analysis evidence that imputation methods do not provide any significant advantages to the predictive accuracy of the model. Imputation using CART can simulate the original distribution of the dataset.

However, for experiments with Machine Learning techniques where the amount of data directly impacts the performance of the model, CART provides the best option to retain rows otherwise removed from the dataset in traditional methods. This approach has been able to increase the usability of the original dataset to 63%, but with no improvement in the performance accuracy.

Manuscript received September 16, 2019.

Sreekumar P. Pillai, Research Scholar, School of Management Studies, NIT Calicut, Kozhikode, Kerala, India.(e-mail: sreekumar.pillai@hotmail.com)

Dr.T. Radha Ramanan, Associate Professor, School of Management Studies, NIT Calicut, Kozhikode, Kerala, India.(e-mail: radha_ramanan@nitc.ac.in)

Dr.S.D. Madhu Kumar, Professor, Dept. of Computer Science, NIT Calicut, Kozhikode, Kerala, India. (e-mail: madhu@nitc.ac.in)

II. BACKGROUND

Many studies have been attempted in the last decade to address the problem of missing values in Datasets through imputation.

Data ignoring approaches does not address the challenge especially in data sparse domains and it is pertinent to explore the opportunities to impute features and increase the availability of the overall data rather than removing the rows completely.

Scholars have been reporting varied results due to imputation. Cartwright (4) has commented that a better estimate for certain missing values for features will not ensure a better overall estimate of the dataset.

2.1 The ISBSG Dataset

The data that we used in this study is from ISBSG (Release 1, 2018). Of late, this dataset has been used in many studies, ISBSG aims to provide IT industry data to better software processes and products, as stated on their website.

This dataset contains information on 8261 projects executed globally and collected from 26 different countries spread globally. The projects collated in the dataset were implemented in 32countries (5).

A systematic mapping study was undertaken by Guevara et al. (2), and they listed the 20 most frequently used variables in Software Effort Estimation studies as evident in the literature. We have considered a subset of these variables that represent the software estimation domain.

III. PROBLEM STATEMENT

Software Effort Estimation is one of the core functions in Software Engineering since it serves to establish processes to manage the production of an intangible asset. All downstream activities, including the sales and promotional activity of software, depends on its estimation.

Despite this, any research into improving the estimation process is hampered by many factors. While quantitative methods are employed in many aspects of engineering, there is a dearth of data from the actual implementation.

Additionally, organizations are reluctant to share project management data to avoid sharing their productivity figures and to comply with confidentiality imposed by regulatory bodies. Research into Software Effort Estimation is faced with challenges in getting quality data (6).

Non-profit organizations such as the International Standards Benchmark Group (ISBSG) (5) consolidates project management data from around the globe and their Release 1 2018 consists of data related to 8261projects from 1989 to 2015.

Unfortunately, missing data is a challenge that software datasets also face as in many other domains. For the most critical variables, there is an average 30% of missing values which limits its use especially in the case of Machine Learning models that performs well with large data volumes (7).

It is pertinent to address the problem of missing data so that the sparse data that is available can be put to optimum use.

IV. DATA SELECTION

The variable selection has been based on our survey of systematic studies conducted previously on the subject (2), (8).

From these, we identified a subset of variables and added a few others that have the most pertinent impact on the effort variable based on our knowledge of the domain and their relevance from an industry perspective.

We have not considered any derived variables other than the primarily observed project attributes.

- **Data Quality Rating:** ISBSG uses two attributes ratings on, Data Quality, and Unadjusted Function Points to denote the reliability of the sizing measure. Projects are classified from A to D (from where the submitter satisfy all criteria specified by ISBSG, to where the data has severe shortcomings). ISBSG recommends only projects classified as A or B to be considered for statistical analysis. In our study, all data rows rated as C or D were removed.
- **Unadjusted Function Point Rating:** Only projects classified as A or B are considered for this research as recommended by ISBSG and other scholars (2).
- **Count Approach:** The counting approach is the technique used to measure the size of a project. The following NESMA, IFPUG 4+, COSMIC, Mark II, IFPUG old, Dreger, Albrecht (6), are retained. All these are different versions of FPA and are retained. Dreger (9) is merely a guide to IFPUG counts. All these variants are included.
- **Recording Method:** Additionally, ISBSG records whether their volunteering organization has collected the information first hand or derived them from other parameters, through the “Recording Method” attribute in their dataset. We have used only data that has been mentioned as directly observed and recorded, in the dataset.

4.1 Rationale for Variable Selection

The section details our rationale for the choice of the 7 independent and the dependent variable

- **Functional Size:** The functional size refers to the size of the software developed and has the most impact on the software development effort but is not linearly related to the project effort (10).
- **Development Type:** The development type variable groups the projects in the ISBSG dataset into the nature of development.
- **Language Type:** The language type variable refers to the generation of the language employed to develop a software product. The programming languages are classified into five distinct generations based on their evolution and how significantly they impact the function of software productivity.

- **Project Elapsed Time:** The project duration in calendar months impacts the effort required to develop software. The relation between software effort and schedule is non-linear as concluded by empirical studies (11).
- **Team Size Group:** The size of the team is represented by the team size group. The size of the team has an indirect impact on the cost due to dysfunctions associated with having a large team.
- **Value Adjustment Factor:** The VAF represents an adjustment factor to scale the functional size to accommodate for differences in the environment under which software is produced. VAF is derived as part of the Function Point Analysis (FPA) estimation process and can vary from 0.65 to 1.35.
- **Industry Sector:** The industry sector is a grouping variable that classifies the ISBSG projects across18 distinct domain areas. The complexity of software products/features needed by industry domains are different and reflect its characteristics. Software productivity also varies across industry domains (12).
- **Work Summary Effort:** The summ.weffort variable represents the effort required to produce soft-ware of defined size and is the dependent variable. Since above 90% of the cost of software development is dependent on human effort, the cost of software production depends on this variable.

V. MISSING VALUE ANALYSIS AND IMPUTATION

Our analysis of missing values in the original dataset with the selected variables is shown in Figure1. The team.size.gp variable has the maximum number of missing values with 36% NAs. 21.729% of the dataset has no missing values. Of the 64% available, there is data missing across four variables and offers potential opportunity for data imputation.

Standard imputation has been performed on ISBSG dataset by Sentas and Angelis (13), Shepperd and Cartwright (14), (4), Song et al. (1), Shera et al. (15), and recently by Mittas et al. (16). We performed imputation of missing values using Burren's, 'Multivariate Imputation' method (17) as implemented in the mice 'R' package.

Within the ISBSG dataset version we used, specific projects have captured the unadjusted Function Points, and there are others that captured Adjusted FP along with VAF. We removed rows only where both these values are non-existent. In cases where two of these values are present, we have derived the missing value based on equation (1). This helps us retain close to 30% of the rows missing VAF in the original dataset.

Data imputation is done for Language Type (lang.type) and Team Size Group (team.size.gp) variables based on multivariate imputation further evidenced in the papers by Cartwright (4), and recently by Lee (18).

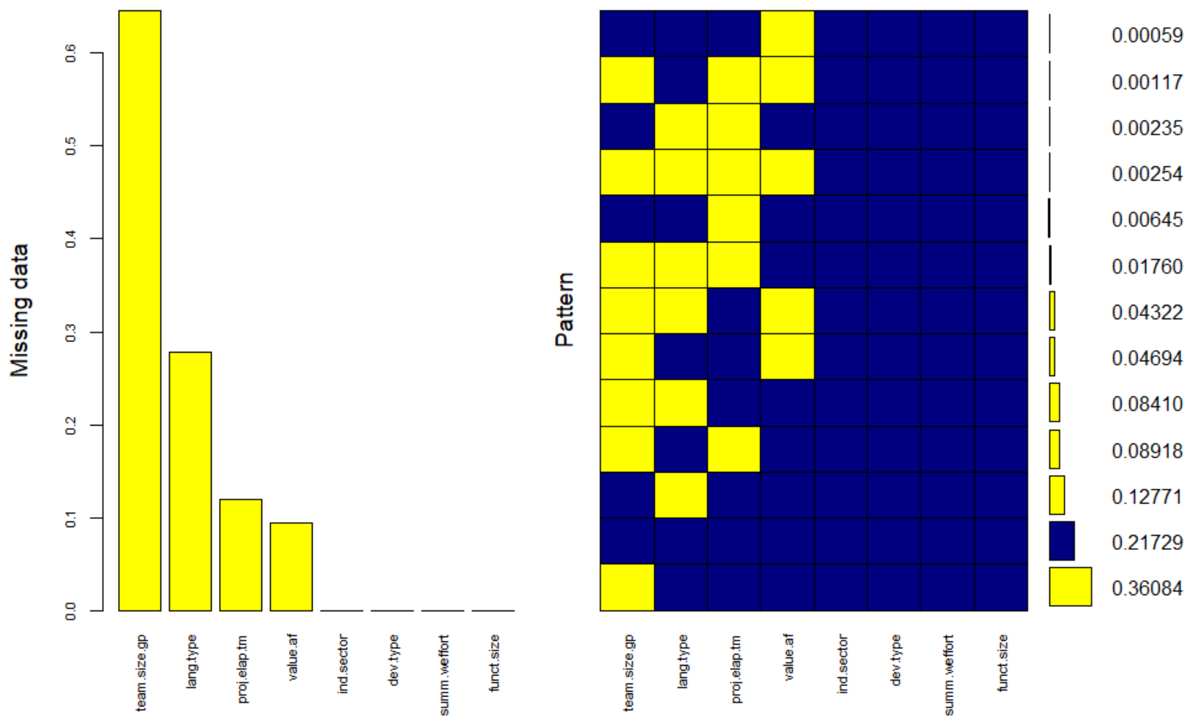

Figure 1: Chart of Missing Data

5.1 *Imputation Methodology*

The multiple imputation techniques were developed by Rubin (19) and Little (20), and overcomes the challenges faced from missing values across most types of datasets. Traditional imputation covers the imputation and error minimization in a single step. The multiple imputations work in three steps: imputation across 'm' datasets, analysis of the error, and finally, the pooling of the results from these 'm', observations. Vaan Burren implemented multiple imputations using chained equations in the MICE package. One condition of multiple imputations is that the data should be missing completely at random (MCAR).

In the case of ISBSG dataset, the data is collated from all over the world (5) so this assumption is ascertained. In multiple imputations, the number of iterations is a critical factor to reckon. Rubin suggests a minimum of 5 repetitions for adequate accuracy; however, experts relate this with the percentage of missing values in the dataset. A 30% missing data on average would require 30 imputations (11).

The imputation can be performed using a variety of methods. We implemented Predictive Mean Matching (PMM) for numerical variables, Cartesian and Regression Trees (CART), Polytomous Regression (Polyreg), and Random Forest (RF) for categorical variables.

The predictive mean matching metric was proposed by Rubin and Little and improves the standard pre-diction line by adding appropriate random noise to the predicted value. A random value is then from a normal distribution with zero mean and the standard deviation of the original dataset. The assumption is that the missing values follow the same distribution as the available rows.

The CART methods are not influenced by outliers and deals with skewed distributions well making it suitable for non-linear variables. CART has been recommended by many authors (4). Multiple imputations for different algorithms have been explored in detail by Vaan (17). Polytomous regression is also known as multinomial logistic regression or nominal regression. This models relationships between a polytomous dependent variable and other independent variables. They have multiple unordered categories and, are called multinomial.

VI. DATA PRE-PROCESSING

The Descriptive statistics for the selected continuous variables in our dataset after pre-processing steps are shown in Table 1, under the data heading “Original”.

6.1 Removing Outliers

To avoid unrealistic inferences regarding outliers from univariate detection, we used a multivariate outlier detection procedure using cook’s distance (21), (22). This approach considers all numerical predictor variables collectively, to decide whether an observation is an outlier. A computational procedure towards this is also provided by Kannan and Manoj (23). It computes the influence of each data point (row) on the predicted outcome as in Equation 5.

We considered observations that have a cook’s distance greater than 4 times the mean as influential outliers and these were removed from the datasets before imputation (24).

6.2 Normalization

As recommended by Peterson and other scholars, on statistical analysis of software projects, Scatter plots are employed for univariate analysis of their data (8), (25). We used multivariate normality-based outlier detection (as discussed in the previous section) to identify and remove outliers. Further, we normalized the data using Box Cox normalization.

Table 1: Descriptive Statistics of the data for each of the Experiments

Variable	mean	sd	median	mad	min	max	range	skew	kurtosis	se
Original (n = 1111)										
value.af	1.01	0.08	1.00	0.00	0.65	1.31	0.66	-0.18	3.73	0.00
funct.size	459.78	793.18	229.00	212.01	6.00	14656.00	14650.00	8.00	108.17	23.80
proj.elap.tm	8.97	7.25	7.10	4.60	0.30	87.00	86.70	3.57	25.52	0.22
summ.weffort	5371.42	9802.71	2387.00	2521.90	17.00	134211.00	134194.00	5.48	44.72	294.10
CART (n = 4630)										
value.af	1.01	0.07	1.00	0.00	0.65	1.31	0.66	0.87	5.60	0.00
funct.size	307.82	654.58	126.00	127.50	3.00	16148.00	16145.00	10.50	186.49	9.62
proj.elap.tm	7.87	6.46	6.00	4.45	0.10	87.00	86.90	2.59	14.65	0.09
summ.weffort	3884.11	7829.75	1595.00	1699.06	4.00	134211.00	134207.00	6.34	58.37	115.07
Polyreg (n = 4630)										
value.af	1.01	0.07	1.00	0.00	0.65	1.31	0.66	0.87	5.60	0.00
funct.size	307.82	654.58	126.00	127.50	3.00	16148.00	16145.00	10.50	186.49	9.62
proj.elap.tm	7.90	6.44	6.00	4.45	0.10	87.00	86.90	2.48	13.91	0.09
summ.weffort	3884.11	7829.75	1595.00	1699.06	4.00	134211.00	134207.00	6.34	58.37	115.07
Random Forest (n = 4630)										
value.af	1.00	0.07	1.00	0.00	0.65	1.30	0.66	0.87	5.60	0.00
funct.size	307.80	654.58	126.00	127.50	3.00	16148.00	16145.00	10.50	186.49	9.62
proj.elap.tm	8.00	6.47	6.00	4.40	0.10	87.00	86.90	2.50	13.79	0.10
summ.weffort	3884.10	7829.75	1595.00	1699.10	4.00	134211.00	134207.00	6.34	58.37	115.07

CART=Classification and Regression Trees, Polyreg=Polytomous Regression, RF=Random Forests, value.af=Value Adjustment Factor, funct.size=Functional Size, proj.elap.tm=Project Elapsed Time, summ.weffort=Summary Work Effort.

6.3 Homogeneous Levels and ANOVA

The ANOVA test for the categorical variables shows the presence of homogeneous groups within the dataset. We analyzed the homogeneous groups using the multiple comparison procedure proposed by Donoghue (26), (27).

Three variables in our dataset had homogeneous groups within their levels, and they were consolidated to form distinct groups. The levels in the lang.type variable was reduced to two levels based on the ANOVA and post-hoc tests conducted to (2GL) (4GL), and (3GL) (ApG) since there existed no significant difference between the mean of these groups.

The dev.type variable was reduced from three levels to two:" New development" and" Enhancement". The one-way ANOVA test done on the ind.sector variable showed a statistically significant difference be-tween 14 industry sectors. The other 7 were re-grouped into the identified primary groups. This helped us balance the number of levels and get significant ones consolidated appropriately.

VII. METHODOLOGY

The steps we adopted to create the samples, the modeling and verification methodology is provided in Figure 2. In the first step, we developed and tuned these model hyper parameters for optimum performance individually, using the grid search method and the best performing one selected for the lowest MMRE.

As the next step, based on the best hyper parameters, we created stack ensembles of these four models, from a grid of all the 12 possible combinations of models. For each imputation method, we obtained 12models, and the best group was selected based on the lowest MMRE.

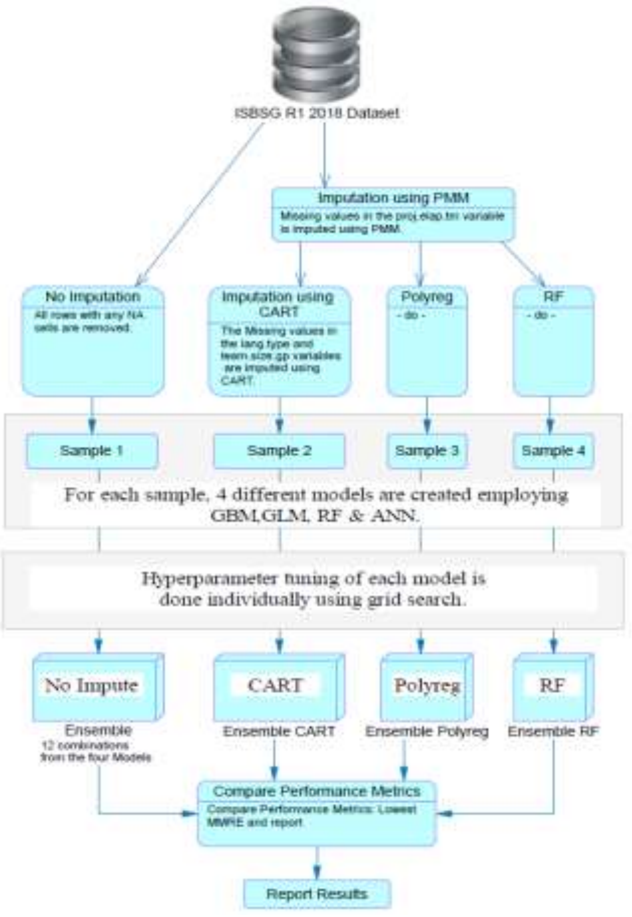
ISBSG R1 2018 Dataset
Imputation using PMM
Missing values in the proj.elap.tm variable is imputed using PMM.
No Imputation
All rows with any NA cells are removed.
Imputation using CART
The Missing values in the lang.type and team.size.gp variables are imputed using CART.
Polyreg
- do -
RF
- do -
Sample 1
Sample 2
Sample 3
Sample 4
For each sample, 4 different models are created employing GBM,GLM, RF & ANN.
Hyperparameter tuning of each model is done individually using grid search.
No Impute
CART
Polyreg
RF
Ensemble
12 combinations from the four Models
Ensemble CART
Ensemble Polyreg
Ensemble RF
Compare Performance Metrics
Compare Performance Metrics: Lowest MMRE and report
Report Results

Figure 2: The Imputation and Validation Steps

We created four samples from the ISBSG dataset. One dataset that we called original had the rows for all NA values removed. For the other three samples replicated from the root dataset, we imputed theproj.elap.tm variable using the PMM (17) method since this is a continuous variable. For the two categorical variables: team.size.gp and lang.type, we employed three different methods Classification and Regression Trees (CART), Polytomous Regression (Polyreg) and Random Forest (RF) thus creating 4 total samples (28). The descriptive statistics of each of these samples is provided in Table 1.

7.1 *Modeling and Testing*

We tested the performance of the imputed dataset using an ensemble model to rule out bias from a single algorithm conflicting with the imputation method used. Additionally, the literature has established that ensembles of different models give better prediction accuracy against individual models (29). The4 different models employing Generalized Linear Model (30), Gradient Boosting Machines (GBM) (31), Random Forests (RF), (32) and Artificial Neural Network (ANN), (33) were combined to form the ensemble. We used the train and test sets to build and test our models by splitting the dataset using the 80:20 ratio. We made the model using the training set reserving the test set to validate the model accuracy within dependent data.

7.2 *Measuring Model Accuracy*

The accuracy of the models is evaluated and reported using the following nomenclature from the literature. For any single project, Magnitude of Relative Error (MRE): Measures the absolute estimation accuracy and is defined as (11):

$$MRE = \frac{|ActualEffort - PredictedEffort|}{ActualEffort} \quad (1)$$

Mean Magnitude of Relative Error (MMRE): The Mean Magnitude of Relative Error (MMRE) is expressed as given:

$$MMRE = \frac{1}{n}\sum_{i=1}^{n} \frac{|ActualEffort - PredictedEffort|}{ActualEffort} \quad (2)$$

Prediction (PRED): For n projects, the prediction at level p is defined as:

$$PRED(p) = \frac{k}{n} \quad (3)$$

k is the number of projects where the MRE is less than or equals p. Desirable MMRE and PRED thresholds were first published by (34), and subsequently by other scholars (35).

VIII. RESULTS

Table 2 summarizes our findings and reports the accuracy metrics of each of the imputation methods and the number of data observations against which the training has been done. The no-impute sample had only 686 observations in the training set against 2800 in the case of others. The imputation methods help increase data availability by 4 times. In the case of the no-impute sample, there is a very high probability of labels in the testing data set against which the model wouldn't be trained on, which can cause the machine learning algorithm to fail at random, affecting the stability of the model.

Innoimpute and CART based imputation, the best accuracy is provided by an ensemble of all four models. In the case of Polyreg and RF imputation, an ensemble of just two algorithms provide the best accuracy.

We evidence that while imputation considerably increases the availability of data (in our case raising it to 64% of the initially available, imputation does not necessarily improve the overall performance of the dataset as commented by other scholars (28). The study by Song et al. (36) on imputation by K-NN algorithm concludes the negative performance of imputation when the overall percentage of missingness exceeds 40%.

While there is a trade-off in terms of predictive accuracy for the additional availability of data, CART remains a viable option to increase data availability for data consuming algorithms in the case of Ma-chine Learning and Deep Learning. The representational capability of the Pmm-CART imputation has increased by 2.8% as seen by the increased R2value. Our study also reiterates that CART performs best compared to other models we have evaluated.

Table 2: Summary of Results

ID	Stack	Model	MSE	RMSE	MMRE	MAE	R2	NOBS
Error Metrics								
1	1,2,3,4	No Impute	0.110	0.332	0.047	0.224	0.818	686
2	1,2,3,4	Pmm-CART	0.162	0.402	0.058	0.212	0.841	2867
3	2,3	Pmm-RF	0.200	0.447	0.066	0.236	0.805	2870
4	2,3	Pmm-Polyreg	0.193	0.439	0.066	0.215	0.834	2868
Prediction								
ID	**Stack**	**Model**	**PRED(5)**	**PRED(10)**	**PRED(15)**	**PRED(20)**	**PRED(25)**	**PRED(30)**
1	1,2,3,4	No Impute	0.594	0.895	0.977	0.995	1.000	1.000
2	1,2,3,4	Pmm-CART	0.547	0.847	0.948	0.973	0.987	0.994
3	2,3	Pmm-RF	0.501	0.801	0.919	0.968	0.985	0.991
4	2,3	Pmm-Polyreg	0.499	0.800	0.919	0.971	0.989	0.995

CART=Classification and Regression Trees, Polyreg=Polytomous Regression, RF=Random Forests, Stack:1=Gradient Boosting Machine, 2=Generalized Linear Model, 3=Random Forests,4=Artificial Neural Networks.

IX. CONCLUSION

The results of our experiments on the ISBSG dataset to improve data availability is promising towards optimum utilization of data but this comes at the cost of the predictive accuracy of the overall model. Among the models that we compared: CART, Polyreg and Random Forest, CART shows the best performance that is near to the non-imputed dataset. We recommend the use of CART in instances where data retention is the priority.

There are opportunities for further experiments in imputation using other algorithms, chained equations or different iterations to improve accuracy. As the next step, we propose research towards identifying new techniques and experiments with other datasets to increase data availability in the software estimation domain.

REFERENCES

1. Q. Song, M. J. Shepperd, X. Chen, J. Liu, Can k -NN Imputation Improve the Performance of C4. 5 With Small Software Project Data Sets? A Comparative Evaluation, *Journal of Systems and Software* 81 (2008) 1–31.
2. F. Gonzalez-Ladron-de Guevara, M. Fernandez-Diego, C. Lukan, The usage of ISBSG data fields in software effort estimation: A systematic mapping study, *The Journal of Systems and Software* 113 (2016) 188–215.
3. T. Menzies, Z. Chen, J. Hihn, K. Lum, Best Practices in Software Effort Estimation, *Ieee Transactions on Software Engineering* 1 (2006) 1.
4. M. Cartwright, Data Imputation Techniques for Software Engineering: Case for Support, *IEEE Transactions on Soft-ware Engineering* (2003) 1–8.
5. ISBSG, Demographics of Development & Enhancement Repository, *Technical Report, ISBSG*, 2017.
6. R. Jeffery, M. Ruhe, I. Wieczorek, using public domain metrics to estimate software development effort, *Proceedings -7th International Software Metrics Symposium*, 2001 (2001) 16 –27.
7. M. Jørgensen, M. J. Shepperd, A Systematic Review of Software Development Cost Estimation Studies, *IEEE Transactions on Software Engineering* 33 (2007) 33–53.
8. K. Petersen, Measuring and Predicting Software Productivity: A Systematic Map and Review, *Information and Soft-ware Technology* 53 (2011) 317–343.
9. J. B. Dreger, Function Point Analysis (Prentice Hall advanced reference series), *Prentice Hall*, 1989.
10. M. Jørgensen, B. Boehm, S. Rifkin, Software development effort estimation: Formal models or expert judgment? *IEEE Software* 26 (2009) 14–19.
11. W. Rosa, R. Madachy, B. Boehm, B. Clark, C. Jones, J. Mcgarry, J. Dean, Improved Method for Predicting Software Effort and Schedule, *International Cost Estimating and Analysis Association (ICEAA)* (2014).
12. A. Trendowicz, J. Munch, Chapter 6 Factors Influencing Software Development Productivity-State-of-the-Art and Industrial Experiences, *Advances in Computers* 77 (2009) 185–241. URL: https://www.ebook. de/de/product /8322757/advances incomputers volume77.html.
13. P. Sentas, L. Angelis, Categorical missing data imputation for software cost estimation by multinomial logistic regression, *Journal of Systems and Software* 79 (2006) 404–414.
14. M. Shepperd, M. Cartwright, Predicting with sparse data, *IEEE Transactions on Software Engineering* 27 (2001) 987–998.
15. S. K. Sehra, J. Kaur, Y. S. Brar, N. Kaur, Analysis of Data Mining Techniques for Software Effort Estimation, 2014 11th *International Conference on Information Technology: New Generations* (2014) 633–638.
16. N. Mittas, E. Papatheocharous, L. Angelis, A. S. Andreou, Integrating non-parametric models with linear components for producing software cost estimations, *Journal of Systems and Software* 99 (2015) 120–134. URL: http: //dx.doi.org/10.1016/j. jss.2014.09.025.
17. S. van Buuren, K. Groothuis-Oudshoorn, S. Van Buuren, K. Groothuis-Oudshoorn, mice: Multivariate Imputation by Chained Equations in R, *Journal of Statistical Software VV* (2011). URL:http://www.jstatsoft.org/.

18. J. B. Taeho Lee, Taewan G, MND-SCEMP: An empirical study of a software cost estimation modeling process in the defense domain, *Empirical Software Engineering* 19 (2014) 213–240.
19. P. Taylor, D. B. Rubin, D. B. Rubin, Multiple Imputation after 18 + Years, *American Statistical Association* 91 (2012)37–41.
20. J. Roderick, A Test of Missing Completely at Random for Multivariate Data with Missing Values Author (s), *American Statistical Association* 83 (2010) 1198–1202. URL:http://www.jstor.org/stable/2290157.
21. R. D. Cook, Detection of Influential Observation in Linear Regression, *Technometrics* 19 (1977) 15–18.
22. X. Y. Leandro L. Minku, Ensembles and locality: Insight on improving software effort estimation, *Information and Software Technology 55* (2013) 1512–152.
23. K. S. Kannan, K. Manoj, Outlier detection in multivariate data, *Applied Mathematical Sciences* 9 (2015) 2317–2324.
24. L. Lavazza, G. Valetto, Requirements-based estimation of change costs, *Empirical Software Engineering* 5 (2000) 229–243.
25. R. J. Yeong-Seok Seo, Doo-Hwan Bae, AREION: Software effort estimation based on multiple regressions with adaptive recursive data partitioning, *Information and Software Technology* 55 (2013) 1710–1725.
26. J. R. Donoghue, Implementing Shaffer’s multiple comparison procedure for a large number of groups, *Wiley Online* 1998 (1998) i–38.
27. S. Graves, H.P. Piepho, L. Sundar, D.-R. Maintainer, L. Selzer, Package ’multcompView’ Visualizations of Paired Com-parisons, R package http://CRAN.R-project.org/package=multcompView (2015).
28. S. Van Buuren, Flexible Imputation of Missing Data Stef Van Buuren, 1st ed., *CRC Press, Netherlands*, 2012.
29. R. Polikar, Ensemble based systems in decision making, Circuits and Systems Magazine, *IEEE* 6 (2006) 21–45.doi:10.1109/MCAS.2006.1688199.arXiv:arXiv:1011.1669v3.
30. T. Nykodym, T. Kraljevic, A. Wang, A. Bartz, Generalized Linear Modeling with H2O, November, H2O, 2017. URL:http://www.h2o.ai/wp-content/uploads/2018/01/GLM-BOOKLET.pdf.
31. V. K. Ayyadevara, Gradient Boosting Machine, Pro Machine Learning Algorithms (2018) 117–134. URL:http://link.springer.com/10.1007/978-1-4842-3564-56.
32. P. Geurts, D. Ernst, L. Wehenkel, extremely randomized trees, Machine Learning 63 (2006) 3–42.
33. A. Candel, E. Ledell, Deep Learning with H2O, June, H2o, 2018.
34. H. S.D. Conte, V.Y. Shen, Software Engineering Metrics and Models, benjamin/c ed., Benjamin/Cummings, Menlo Park, 1986, 1986.
35. D. Port, M. Korte, Comparative Studies of the Model Evaluation Criterions Mmre and Pred in Software Cost Estimation Research, Proceedings of the Second ACM-IEEE *International Symposium on Empirical Software Engineering and Measurement* (2008) 51–60. URL:http://doi.acm.org/10.1145/1414004.141401.
36. L. Song, L. L. Minku, X. Yao, the impact of parameter tuning on software effort estimation using learning machines, Proceedings of the 9th *International Conference on Predictive Models in Software Engineering - PROMISE* ’13 (2013)1–10.