Received: August 2, 2021. Revised: September 18, 2021.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Empirical Analysis of Software Effort Preprocessing Techniques Based on Machine Learning

Robert Marco[1]* **Sakinah Sharifah Syed Ahmad[2]** **Sabrina Ahmad[2]**

[1]*Department of Informatics, Universitas Amikom Yogyakarta, Yogyakarta, Indonesia*
[2]*Faculty of Information & Communication Technology, Universiti Teknikal Malaysia Melaka, Malaysia*
* Corresponding author's Email: *robertmarco@amikom.ac.id*

Abstract: In the subject of software effort estimation, missing data is a significant issue that leads to information loss and bias in data analysis. The majority of data preprocessing procedures are simple reuse approaches built for numerical data, which presents a problem when missing data and irrelevant features are associated with categorical variables. The purpose of this paper is to evaluate and compare the performance of the proposed technique with the k-nearest neighbor imputation (kNNI) technique, random forest imputation, and multiple imputation by chained equations in terms of error and accuracy in the ISBSG dataset when missing data is present. This study relied on five machine learning approaches as its foundation with hyperparameter tuning using grid search. The results show that the three imputation methods have almost the same performance. However, the combination of kNNI, Genetic feature selection, and classification and regression tree (CART) yielded better results than other combinations of methods with MAE (0.015), RMSE (0.037) and R-squared (0.804) values.

Keywords: Missing data, Software effort preprocessing technique, Imputations, Machine learning.

1. Introduction

The field of software effort estimation (SEE) is a key factor in the successful development of software engineering [1]. Software success is highly dependent on consistency and accuracy to avoid inaccuracies and ambiguity in software development efforts [2, 3]. Unfortunately, the data also present a challenging problem for effort estimation [4]. Because, building an SEE model the main input is to use a software project dataset [5]. Although a large number of software projects are available, the value of the effort collected is usually damaged by data tampering due to human participation.

However, problems arise when building the SEE model there are missing data. In software engineering, missing data is a severe issue since it can lead to information loss and bias in data analysis [6]. For instance, more than 40% of the variables in the International Software Benchmarking Standard Group (ISBSG) dataset are missing [7]. Missing data in key software attributes is a common occurrence that can lead to erroneous estimates and poor prediction ability [8, 9], and can have a negative impact on the learning process, leading to erroneous conclusions [10].

Little and Rubin (1989), mentions that there are three categories for missing value patterns, such as missing completely at random (MCAR), missing at random (MAR), and nonignorable missing (NIM) [11]. However, there have been many previous studies in developing data imputation methods based on artificial intelligence systems. In most cases, there are two methods for dealing with missing data: the embedded technique and the preprocessing method [12].

In addition, the empirical study of Yadav and Roychoudhury (2018), mentions a package of popular imputation methods available in R such as: k-nearest neighbor imputation (kNNI), Multiple imputations by chained equation (MICE), and random forest imputation (MissForest) [13]. The fundamental issue with imputation approaches is that they are more difficult to handle when missing data occurs in categorical characteristics. According to Idri et al (2015), despite the fact that most software project data sets consist of mostly categorical features with many missing values, but most of the imputation techniques are applied to numeric data types [14].

kNNI is a well-known and computationally simple method for missing data imputation that uses observations in the environment to account for missing values [15-18]. In addition, kNN can predict categorical or numeric attributes using the value of mode/mean or median [19]. Several previous studies investigated the imputation of kNN in software engineering datasets with categorical attributes in a classical way, mainly using classical numbers or intervals and overlapping measures to evaluate the similarity between software projects [20]. This technique has proven to be the most frequent imputation technique for datasets in the SEE field. However, MICE and missForest they may be ignored as imputation methods in this field

When targeting estimation accuracy, much effort has been devoted to improving machine learning (ML) methods [17]. For empirical validation, the ML algorithm is routinely tested on the SEE datasets. The use of data preprocessing (DP) techniques as a fundamental step in helping to increase machine learning performance [15, 16, 21].

To the author knowledge, there is very limited research focusing on DP techniques in the SEE literature. In many situations DP techniques, such as missing data imputation [15, 16, 22, 23], and feature selection [16,24-26] has been considered a necessary step for case-based reasoning (CBR), artificial neural networks (ANN), support vector regression (SVR) [27], while for Other ML methods, such as random forest (RF), classification and regression tree (CART) and k-nearest neighbor (kNN), they may be ignored.

Sidra Tariq et al (2020) revealed that the DP phase was still limited to the sample data set, which included handling missing values and feature selection [28]. Strike et al (2001) simulated several incomplete data sets and discovered that imputing missing data with Z-score normalization produced the best regression model [29]. However, missing values must be discarded, if possible, to eliminate biases which can alter the accuracy of ML predictions [29]. In contrast, in Huang et al (2017), the list deletion in SEE has become less frequent because it reduces the exhaustiveness of the data and therefore makes it less appropriate for the application of the derived model [16].

Proper data preprocessing improves the accuracy prediction in the end. However, machine learning techniques can have a negative effect on the prediction of performance without properly considering their features [15, 28]. According to Huang et al (2017), that DP techniques are an effective choice for effort estimation [16]. Several previous studies combining multiple scheme on DP techniques were examined first, but their impact on the ML approach was not investigated.

The aim of this paper is to determine whether the imputation approach can be applied effectively to the DP technique for attributes containing mostly categorical attributes with many missing values, and in particular the relative performance of the imputation approach and how imputation changes the relationship between data properties. We tested three imputation approaches: kNNI, MICE, and missForest with data sets of more than 40% missing values. Then evaluate the performance of the approach by comparing the performance results of the imputation method using five predictor methods in machine learning (such as multilayer perceptron (MLP), SVR, kNN, CART, and RF).

Based on the analysis of previous empirical studies on several data preprocessing techniques for the ML method. The remainder of this paper is structured as follows: Section 2 presents a related work on the application of missing data techniques in SEE, Section 3 presents missing imputations techniques, Section 4 presents experimental design, Section 5 presents the experimental results and analysis, Section 6 concludes this work and points out future research directions.

2. Related work

A mapping study conducted by Idri et al (2015), they classified 35 papers related to the treatment of missing data in software engineering repositories. In their study, six criteria were taken, such as: research type, research approach, missing data technique, missing data type, data type, and purpose. The results of the study explained that 94% of the papers selected ISBSG were frequently used datasets, 54% of the selected papers only treated MCAR, 63% were dedicated to dealing with missing numerical and categorical data, and 97% of selected papers were treated with imputation techniques [14]. Table 1, shows several previous studies in the field of SEE that investigated missing data in categorical and numerical attributes explaining information about the investigated SEE techniques, missing data techniques used, and missing data mechanisms handled in each study.

Based on the results of the summary of findings in Table 1, that analogy-based estimation and regression are the most studied effort estimation in the context

Table 1. Related work of the application of missing data techniques in SEE

ID	Effort technique	Imputation methods	Mechanism	Findings
[18]	ABE	Toleration, deletion and kNNI	MCAR, MAR, NIM	The results show that fuzzy analogy produces more accurate estimates in terms of standard accuracy measures. Another finding, that of kNN imputation, yields accurate estimates rather than toleration or deletion.
[16]	ABE	MI, kNNI	MCAR	The findings also suggest that combining Z-score normalization, kNN imputation, and mutual information-based feature weighting for analogy-based effort estimate is a viable option.
[29]	Regression	LD, MI, hot-deck imputation	MCAR, MAR, NIM	The simplest technique, listwise deletion, is an missing data strategies work well (with a bias value always less than 0.03). The best results are obtained by employing a hot-deck imputation with euclidean distance.
[7]	Regression	LD, MI, SRPI, FIML	MCAR, MAR	For missing at random (MCAR) data, FIML works well. Also, for MD mechanisms other than MCAR, state that LD, MMSI, and SRPI were demonstrated to produce biased findings (MMRE values for LD of 48%, MI of 61%, SPRI of 68%, and FIML of 74%).
[15]	CBR, ANN, CART	LD, MI	MCAR	In terms of increasing predictive accuracy and resilience, LD is favoured over MI for data sets with a substantial number of missing values. CBR is a better fit for feature selection than ANN or CART. In our investigation, ANN performed worse than the other two ML approaches.
[30]	Regression	Toleration, kNNI, MI	MCAR	Although MI improves accuracy over toleration, kNN produced the greatest results. MMRE = 155.1 percent for toleration, 62.1 percent for MI, and 28.2 percent for kNN in the first data set. No model could be developed for the toleration data in the second data set (MMRE values for SMI of 112% and kNN of 78%).
[31]	Decision tree	NNSI (kNN single imputation), MI	MCAR, MAR, NIM	BAMI is a suggested ensemble missing data approach that incorporates kNN and multi imputation. The empirical findings reveal that BAMI is the better technique overall, with an excess error rate of 8.9% with 10.6% for NNSI.
[23]	C4.5	kNNI, toleration	MCAR, MAR, NIM	C4.5 prediction accuracy can be improved via kNN imputation. The missingness mechanism affects tolerance and kNN, nevertheless, the pattern and proportion of missing data have a significant detrimental impact on prediction accuracy, especially when the percentage of missing data approaches 40%
[32]	Regression	LD, MI, EM, RI, MLR	MCAR, MAR, NIM	The efficiency of the suggested method is shown by the comparison of MLR with other strategies for dealing with missing data in various patterns and the percentage of missing data with standard deviation values: MLR = 178.410, EM = 214.848, MI = 269.624, RI = 255.035, LD = 353.526)
[33]	ABE	kNNI	MCAR, MAR, NIM	The results of an ensemble kNN imputation were compared to those of a single GS-kNN imputation and kNN without optimization. Result E-kNNI outperformed to other methods, according to our findings.
[34]	ABE (1NN), CART	MI	MCAR	According to our findings, 1NN and CART perform worse on smaller data sets than on complete data sets. As a result, using these approaches without taking into account the magnitude of the data is not advised.
[35]	RT, RBF, MLP	kNNI	MCAR	For ISBSG, which is anticipated to be more heterogeneous, approaches like kNN are more frequently among the best, whereas RT is more frequently among the best for PROMISE. The MAE of these approaches did not differ statistically significantly according to the Friedman test (statistic Ff = 0.2612 F(4,48)=2.565).

Abbreviations: Mean/mode imputation(MI); Listwide deletion(LD); similar response pattern imputation(SRPI); full information maximum likelihood(FIML); expectation maximization imputation (EM); regression imputation (RI); multinomial logistic regression (MLR); Regression tree (RT); Radial basis function (RBF); Analogy based estimation (ABE) of missing data [16], kNNI is the most investigated imputation technique, found to be the best in most cases [6,16,33], and MCAR is the most effective omission mechanism widely used in the field of SEE [14], and data preprocessing as an effective approach in dealing with missing data [15, 16, 23, 35].

3. Software effort preprocessing techniques

3.1 Ordinal Encoding

Ordinal Encoding assigns for each category a unique number code [36] and as every category is displayed as a single input, the advantage is that the problem space dimensions do not increase [37]. Give value f for the i-th object is x_{if} and f have status sorted M_f which presented rank $\{1,2,\dots,M_f\}$. Replace each x_{if} be ranked accordingly $r_{if} = \{1,2,\dots,M_f\}$. Map the range of each attribute to [0, 1] so that each attribute has the same weight. Use z_{if} to present attributes r_{if} from the i-th object.

$$z_{if} = \frac{r_{if} - 1}{M_f - 1} \quad (1)$$

Dissimilarity is then calculated using one of the distance measurements, namely the Manhattan distance, formulated as follows:

$$d(x_j, x_j) = \sum_{n=1}^{N} |x_i - x_j| \quad (2)$$

Difference matrix (or object-per-object structure): This stores a set of approximations available for all pairs of n objects. These are often represented by tables *n-by-n*. Where $d(x_i, x_j)$ is the measured inequality or "difference" between objects x_i and x_j. The similarity value for the ordinal attribute can be interpreted from dissimilarity as $sim(x_i, x_j) = 1 - d(x_i, x_j)$.

3.2 Data normalization

Data normalization (DN) changes the value of a feature according to preset guidelines to ensure that every scaled feature has the same influence [38]. In our study, we will use the interval [0,1] as a scaling target, as in Eq. (3) below:

$$[0,1] Interval\ on\ x_k = \frac{x_k - \mathrm{x}_{min}}{\mathrm{x}_{max} - \mathrm{x}_{min}} \quad (3)$$

After the scaling is done, then the unstructured data can be normalized using the Z-score parameter, according to Eq. (4) [15, 16, 39]:

$$Z - score\ x_k = \frac{x_k - \bar{x}}{std(x)} \quad (4)$$

Where, x is the feature column in the data matrix. x_k represents the k-th value of x_{min} corresponds to the minimum values, while x_{max} corresponds to the maximum values in x. While x and $std(x)$ are calculated using the mean and standard deviation x.

3.3 Missing data imputation techniques

3.3.1. K-nearest neighbors imputation

The k-nearest neighbors imputation (kNNI) based analogy algorithms that have been shown to be an efficient method of estimating missing values of the attributes in various software engineering dataset [22, 23, 30]. The kNNI approach has been extended to imputation of missing data across multiple dataset [40]. kNNI imputation is generally the right choice when we do not have prior knowledge of the distribution of data [41].

3.3.2. Multiple imputation by chained equations

In the statistical literature, multiple imputations by chained equation (MICE) from Raghunathan et al (2001) have emerged as one of the principle methods of dealing with missing information, sometimes known as the "full condition specificities" or "sequential regressions multiple imputations" [42]. According to Schafer and Graham (2002), are explaining that missing values are calculated on the basis of the observed values of a particular individual and of the relations for other participants observed in the data provided that the observed variables are contained in the imputation model. This approach can reduce nonresponse bias and increase strength [43].

3.3.3. Random forest imputation

Random forest is a well-known machine learning technique that has been used to solve problems like regression and classification [44]. Currently, the random forest approach has been expanded as an iterative imputation method (dubbed "MissForest") to handle the problem of missing data [45]. There are two crucial parameters in the missForest function in the R package missForest. The first, ntree, is the number of trees per forest, and the second, mtry, is the total number of variables sampled at random within each tree split. The random forest original goal was to choose these two parameters internally by looking for Out-Of-Bag (OOB) issues.

3.4 Feature selection

The use of genetic algorithms (GA) as feature selection (FS) uses a parallel search random strategy, directed to the search for high fitness points, i.e. the point at which the function to be minimized or maximized has a relatively low or high value [26], with the Genetic operators as follows [46]: selection, the process of selecting individuals from a population for further breeding using the roulette wheel selection method. The probability of selecting a specific individual h_i is defined as:

$$P(h_i) = \frac{F(h_i)}{\sum_{i=1}^{p} F(h_i)} \qquad (5)$$

Where, $F(h_i)$ is the fitness value of h_i . Meanwhile, a single point crossover operator will be used for crossover. The crossover point *i* is chosen at random so that one parent contributes the first bit of *i* and the second parent contributes the remaining bits. Furthermore, each individual has a p_m chance of mutating.

3.5 Machine learning methods in study

3.5.1. Support vector regression

Support vector regression (SVR) is a new generation of machine learning techniques that can be used to model predictive data. SVR is a support vector machines based approach [47]. For pedagogical reasons, this is linear function f, which is presented in the Eq. (6).

$$f(x) = \langle w, x \rangle + b \; with \; w \epsilon \mathbb{R}^d, b \epsilon \mathbb{R} \qquad (6)$$

Where $\langle .., .. \rangle$ represents the dot product in $\mathbb{R}^d$. For case nonlinear regression $f(x) = \langle w, \emptyset(x) \rangle + b$, where $\emptyset$ represented as some nonlinear function that can map the input space to a higher dimensional feature space $\mathbb{R}^d$. ε-SV, to optimize it is selected weight vector w and threshold b.

3.5.2. Multilayer perceptron

Because of its rapid operation, ease of construction, and smaller training set needs, the multilayer perceptron (MLP) is the most often used feed-forward neural network [48, 49]. Each neuron j in the hidden layer adds up the input signals x_i that are impinging on it after multiplying them by their connection weights w_{ji} . The following is a description of each neuron output:

$$y_j = f\left(\sum w_{ji} x_i\right) \qquad (7)$$

Where f represents the activation function. E is the sum of the squared differences in the output neurons desired and actual values [48, 49]:

$$E = \frac{1}{2} \sum_j (y_{dj} - y_j)^2 \qquad (8)$$

Where y_{dj} is the desired value of output neuron j and y_j is the neuron's actual output. Depending on the training algorithm used, each w_{ji} weight is adjusted to minimize the value *E*.

3.5.3. Classification and regression tree

Classification and regression tree (CART) builds a binary decision tree by breaking down data sets such that the data in the subsets is purer than the data in the parent set. For example, in a regression problem, suppose (x_n, y_n) represents the *n*-th example, where x_n is the *n*-th sample vector of the independent variable and y_n is the value of the dependent variable. If there are a total of *N* samples, CART calculates the best separation so that the following is maximized for all possible *S* separations [50].

$$\Delta R(s^*, t) = argmax \Delta R(s, t), s \in S \qquad (9)$$

Where $\Delta R(s, t) = R(t) - R(t_L) - R(t_R)$ is an increase in the estimated replacement for separation *s* from *t*.

3.5.4. K-narest neighbor

The k-nearest neighbor (kNN) approach groups objects based on the feature space's nearest training sample. kNN is a sort of example-based learning, often known as lazy learning, in which functions are only estimated locally and all calculations are postponed until classification and regression are completed [51]. d is the Euclidean distance between two d data points $d(x, y)$. where N denotes the number of effort drivers that describe a specific historical project. The illustration in Eq. (10) [52].

$$d(x, y) = \sum_{i=1}^{N} \sqrt{(x_i - y_i)^2} \qquad (10)$$

Received: August 2, 2021. Revised: September 18, 2021.

3.5.5. Random forest

For regression purposes, a random forest (RF) is created by developing a tree that relies on a random vector λ and performs numerical data rather than class labels with specifies that the tree predictor $h(x, \lambda)$. The predictor's output is $h(x)$ and the actual work value is Y. The common mean square error is calculated for each numerical predictor $h(x)$ as:

$$E_{x,Y}(Y - h(x))^2 \tag{11}$$

The RF predictor is modeled by calculating the mean value obtained in k trees $h(x, \lambda_k)$.

3.6 Error metrics

The metrics that are utilized for evaluation are mean absolute error (MAE), root mean square error (RMSE), and R-squared (R²). The model is better if its MAE and RMSE values are low, and its R² value is high.

$$MAE = \frac{1}{m}\sum_{i=1}^{m}|X_i - Y_i| \tag{12}$$

$$RMSE = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(X_i - Y_i)^2} \tag{13}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{m}(X_i - Y_i)^2}{\sum_{i=1}^{m}(\bar{Y} - Y_i)^2} \tag{14}$$

4. Experiment design

The DP approach in our study is used for handling attributes that mostly contain categorical attributes with many missing values, with the DP stage, including: categorical conversion, missing data imputation, data normalization, and feature selection used in our study. The complete experimental technique is depicted in Fig. 1. As a first step, we will do a categorical conversion. The use of the ISBSG dataset is a heterogeneous dataset type, so to overcome categorical data using ordinal encoding. The next step is to solve missing data using imputation method, in our study compare 3 imputation methods, such as: kNNI, MICE, and missForest. Next, it will be scaled into the range[0, 1]. After that, it is followed by the determination of "NA or NAN" which will automatically be assigned to each particular column value that has an empty string, while for the numeric column it will be filled with "0". Meanwhile, for the feature selection method, we compare 3 methods, such as GA (genetic algorithm), LVF (low variance filter) and CF (correlation filter). The test results from all data splits are aggregated after 10 iterations to calculate the error metrics.

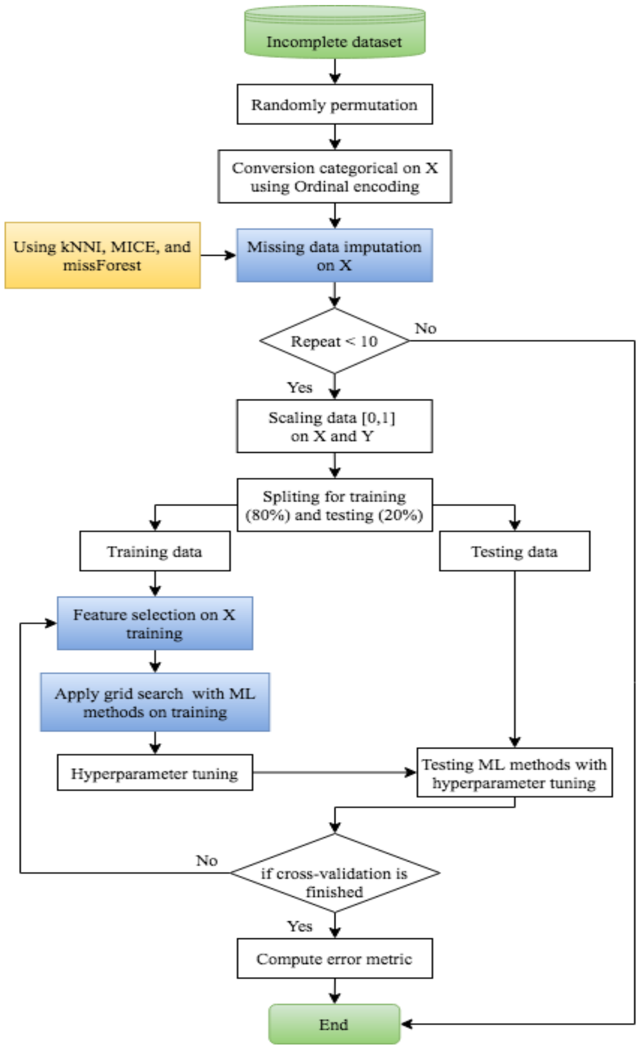
Incomplete dataset
Randomly permutation
Conversion categorical on X using Ordinal encoding
Using kNNI, MICE, and missForest
Missing data imputation on X
Repeat < 10
No
Yes
Scaling data [0,1] on X and Y
Spliting for training (80%) and testing (20%)
Training data
Testing data
Feature selection on X training
Apply grid search with ML methods on training
Hyperparameter tuning
Testing ML methods with hyperparameter tuning
No
if cross-validation is finished
Yes
Compute error metric
End

Figure. 1 Experiment procedur

It should be noted that, our study will use 2 subsets such as training as parameter setting (80%), and testing as predictive evaluation of the training method (20%) at random. We used five ML methods for the experiment. Meanwhile, to set the parameter values of a single technique using a grid search, show in Table 2 for more details.

5. Result and discussion

In this study, we used the publicly available ISBSG dataset from the PROMISE repository for empirical testing. The results of the first analysis carried out were to investigate the features, type, and total missing values in ISBSG which are show in Table 3.

Table 2. Grid search spaces for each ML technique parameter values

ML	Parameter Values

SVR	Kernel: RBF
	Kernel Parameter: {0.0001, 0.001, 0.01, 0.1}
	Degree: {3, 4, 5, 6, 7, 8, 9}
	Deviation tolerated: {0.001, 0.01, 0.1}
	Learning rate: {0.01, 0.02, 0.03, 0.04, 0.05}

MLP	Kernel: RBF
	Minimum instance per leaf: {1, 2, 3, 4, 5}
	Max iter: {100, 500, 1000}
	Momentum: {0.1, 0.2, 0.3, 0.4, 0.5}
	Random state: {1, 2, 3, 4, 5}
	Hidden layer sizes: {1, 3, 5, 9}

CART	Minimal Size for Split: {1, 2, 3, 4, 5} Minimal Leaf Size: {1, 2, 3, 4, 5} Confidence: 0.25 Minimal Gain: 0.1 Number of Prepruning: 3 Maximal Depth: 20 Number of Trees: 10 (*for RF)
kNN	K: {1, 2, 3, 5} Similarity measure: Euclidean Distance

RF	Number of Trees: 125
	Max depth of the tree: {100, 200, 300}
	Min samples split: {3, 5, 7, 9}
	Min sample leaf: {3, 4, 5, 6, 7}

Table 3. Descriptive ISBSG10

ID	Feature	Type	Total missing (%)
1	Functional Size	Ratio	0
2	Value Adjustment Factor	Ratio	33.29
3	Project Elapse Time	Ratio	30.77
4	Development Type	Nominal	0
5	Business Area Type	Nominal	77.31
6	Client Server	Nominal	71.95
7	Development Platform	Nominal	50.84
8	Language Type	Nominal	25.21
9	First OS	Nominal	68.17
10	Maximum Team Size	Ratio	57.77
11	Normalised Level 1 Work Effort	Ratio	0

5.1 Performance under categorical conversion scheme

In this study, for the SEE method that handle numerical and categorical features separately, we use proximity measures as an example procedure for handling ordinal features on a training project from Eq. (1) with manhattan distance from Eq. (2). Categorical feature sorting has no influence on distance in this way. Our preprocessing may yield ordering information among categorical data, which can be misleading because SEE approaches can only handle numeric features. Meanwhile, a one-hot encoding/dummy variable that employs the values 0 Eq. (1) to signal the absence (presence) of categorical feature values is used in preprocessing methods. However, the method has the potential to significantly increase the number of features.

Table 4. Descriptive statistics under MDI scheme

ID	kNNI		MICE		MissForest	
	Mean	std	Mean	std	Mean	std
1	0.061	0.099	0.061	0.099	0.061	0.099
2	0.602	0.130	**0.606**	0.121	0.603	0.130
3	0.131	**0.097**	0.126	**0.097**	0.131	0.095
4	0.230	0.421	0.230	0.421	0.230	0.421
5	**0.769**	0.362	0.756	0.285	0.522	**0.477**
6	**0.750**	0.365	0.659	0.450	0.558	**0.453**
7	0.349	0.309	**0.357**	0.307	0.258	**0.341**
8	0.361	0.438	**0.371**	0.418	0.342	**0.442**
9	0.406	0.349	0.454	0.297	**0.477**	**0.394**
10	0.022	0.037	**0.029**	**0.040**	0.026	0.039
11	0.045	0.081	0.045	0.081	0.045	0.081

5.2 Performance under missing data imputation scheme

Marginal mean and standard deviation of test error in terms of standardized accuracy when using the three missing data imputation (MDI) techniques are show in Table 4. From the table, the results show that MICE may be more effective than kNNI and MissForest (based on mean/average). The ISBSG repository has a serious problem with missing values. Our data analysis suggests that MissForest may not provide a strong estimate because it doesn't exist. MissForest is only slightly better under MICE and kNNI. MICE is somewhat superior than kNNI in terms of imputation of missing data, according to the marginal effect. MissForest greater standard deviation, on the other hand, suggests that in some cases, MissForest accuracy may be superior to kNNI and MICE. The properties of the ISBSG data, which have high skewness values and strong kurtosis (see Table 6), also influence MDI impact. To further explore the impact of MDI, it is necessary to combine DP to use other imputation methods. However, kNNI, MICE and missForest have nearly the same performance results.

The relationship between two variables using the pearson correlation method. Correlation ranges between -1 and 1. A number close to or equal to 0 indicates that the two variables have little or no linear relationship. The linear relationship, on the other hand, is stronger the closer it gets to 1 or -1. In the next step, a heatmap is constructed to show the correlation between all the features, and its graph is

Received: August 2, 2021. Revised: September 18, 2021.

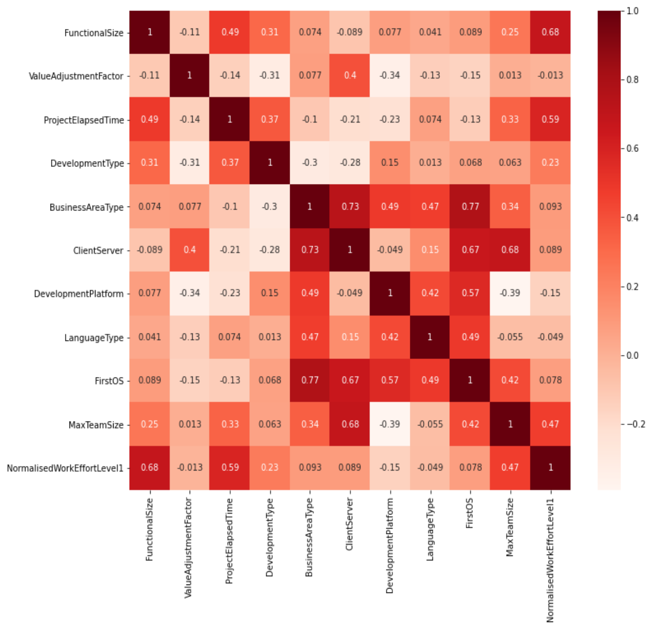

Figure. 2 Correlation heatmap dataset

Table 5. Performance evaluation under MDI scheme for each ML methods

MDI	ML	Testing Error Metrics			
		MAE	RMSE	R^2	SD
kNNI	SVR	0.062	0.070	0.316	0.059
	MLP	0.052	0.085	-0.027	0.052
	CART	**0.018**	**0.042**	**0.751**	**0.073**
	kNN	0.022	0.044	0.718	0.063
	RF	0.025	0.045	0.710	0.053
MICE	SVR	0.053	0.062	0.450	0.070
	MLP	0.048	0.076	0.183	0.065
	CART	0.021	0.045	0.711	**0.071**
	kNN	**0.016**	**0.038**	**0.797**	0.065
	RF	0.025	0.045	0.707	0.051
Miss Forest	SVR	0.066	0.074	0.237	0.058
	MLP	0.047	0.070	0.308	0.071
	CART	**0.019**	**0.040**	**0.774**	**0.074**
	kNN	0.021	0.044	0.726	0.060
	RF	0.024	0.044	0.718	0.054

given in Fig. 2.

The following is the error metric value of the test error under the three MDI scheme for each ML method show in Table 5.

In our imputation method assumptions, all test data values are missing. Because the imputation method returns a prediction column for each column of test data. In the training process, the imputation method is trained with the default hyperparameters on the training data. Meanwhile, in the machine learning method as an estimator, we use a hyperparameter tuning technique using a grid search. The actual value in the test data column is estimated during the imputation process. The model with the lowest MAE and RMSE gives more precise results. while, for the value of R^2 and standard deviation (SD) if it has a higher value then it gives better accuracy results. For the performance of the three imputation methods that take into account the values of MAE, RMSE, R^2, and SD have been given in Table 5.

As shown in the table, the kNNI-CART has the best performance as indicated by the values of MAE(0.018), RMSE(0.042), R^2(0.751), and SD(0.073). Meanwhile, kNNI-SVR and kNNI-MLP had the poor performance. Meanwhile, kNNI-kNN and kNNI-RF have almost the same good performance. For the MICE imputation method, it shows that MICE-kNN has the best performance with MAE(0.016), RMSE(0.038), and R^2(0.797) values. Although MICE-CART has the highest value on SD(0.071) on the other hand, indicating that in some cases, the accuracy of MICE-CART may be superior to that of the MICE-kNN method. Meanwhile, MICE-CART and MICE-RF almost have the same performance. Meanwhile, MICE-MLP has the poor performance. Meanwhile, missForest-CART has the best performance with values of MAE(0.019), RMSE(0.040), R^2(0.774), and SD(0.074). While missForest-SVR and missForest-MLP had the worst performance. Meanwhile, missForest-kNN and missForest-RF have almost the same good performance. Although, it is important that all three MDI scores are nearly equal in performance. The imputation method shows that CART, kNN, and RF have the best performance compared to SVR and MLP. Whereas MLP and SVR have the poor performance in this regard. MLP and SVR were consistently negatively affected by all three imputation methods, and were less sensitive to a small number of missing values. Meanwhile, CART, kNN, and RF are sensitive and have resistance to outliers.

We compared imputation methods on data sets containing mostly categorical attributes with many missing values more than 40% missing data with three different imputation methods: kNNI, MICE, and missForest. To compare and evaluate the accuracy of the suggested ML algorithms for each imputation approach. In addition, statistical results were evaluated using a 5-fold cross-validation procedure. When, MAE and RMSE are low, the algorithm error is also low, on the contrary, R^2 and SD values must have the highest values. In full, it will be show in Fig. 3 and Fig. 4, where the point that has the lowest value (MAE and RMSE) is the method that has the best accuracy value. While, R^2 and SD show in Fig. 5, and Fig. 6 show a comparison of the methods, if they have the highest value as the method that has the best performance.

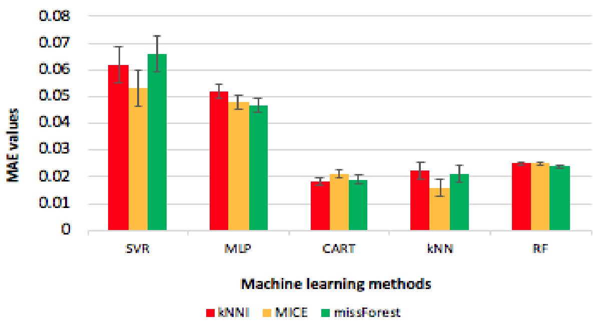

Figure. 3 MAE imputation error of MDI and ML methods

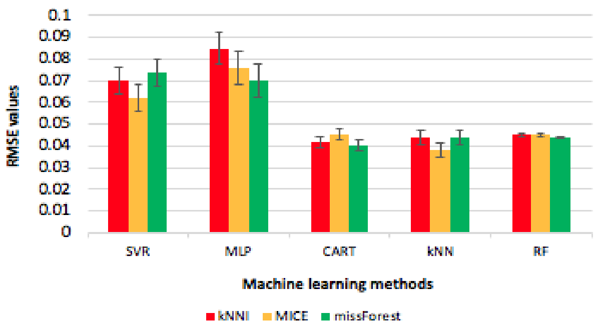

Figure. 4 RMSE imputation error of MDI and ML methods

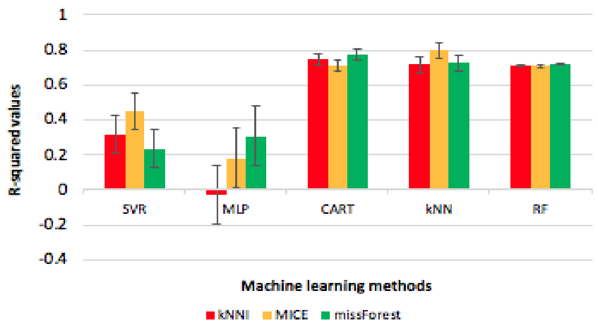

Figure. 5 R^2 imputation accuracy of MDI and ML methods

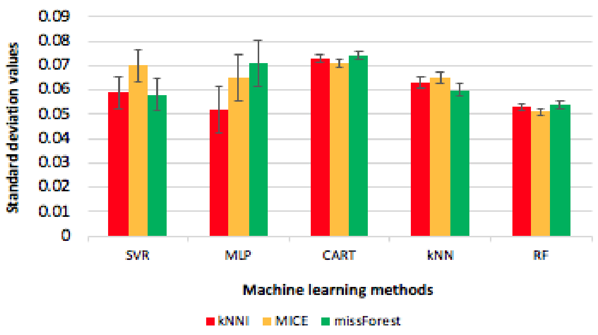

Figure. 6 SD imputation accuracy of MDI and ML methods

Table 6. Descriptive statistics under DN scheme

ID	Mean	Skew	Kurt	std
1	0.061	4.103	22.457	0.099
2	0.596	**0.124**	1.906	0.138
3	0.137	2.380	9.810	0.109
4	0.230	1.284	**-0.349**	0.421
5	0.810	-1.593	**0.542**	0.393
6	0.797	-1.490	**0.224**	0.402
7	0.301	**0.666**	-1.062	0.359
8	0.352	**0.618**	-1.621	0.478
9	0.384	**0.413**	-1.158	0.379
10	0.025	14.249	247.123	0.055
11	0.045	5.860	50.078	0.081

5.3 Performance under data normalization scheme

In this study, we use data normalization (DN) which is used as a scale of 0 and 1 values. The results of the study by Mensah et al (2018) found that normalized Z-score [0,1] resulted in the best prediction accuracy than box-cox and log transform for all dataset [53]. Huang et al (2017), the use of the [0, 1] scheme slightly outperformed the [-1, 1] scheme [15, 16]. And the Z-score scheme can improve the overall estimation accuracy. The results of data analysis show that for the marginal mean, skewness, kurtosis and standard deviation on standardized accuracy with the Z-score scheme, show in Table 6.

The results show that the schema [0, 1] for the ISBSG dataset is highly skewed, although there are some features that are less skewed. The ISBSG subset is therefore a high-order non-normality. If the number is larger than +1 or fewer than -1, the distribution is severely skewed, according to a standard skewness rule of thumb. The basic criterion for kurtosis is that if the number exceeds +1, the distribution is excessively peaked. A kurtosis of less than -1, on the other hand, denotes a distribution that is too flat. A distribution that exceeds these parameters in terms of skewness and/or kurtosis is considered abnormal.

5.4 Performance under feature selection scheme

The goal of feature selection (FS) utilizing GA is to determine the best option among a set of possible solutions. A population is a collection of solutions. Vectors, such as chromosomes or individuals, make up the population. A gene is the name given to each item in the vector. Chromosomes are represented as binary strings of 1 and 0 in the suggested method. The number 1 denotes feature selection, while the number 0 denotes non-selection. Population size = 952, generations = 40, crossover rate = 0.5, mutation rate = 0.05, two-point crossover randomly selected,

Table 7. Performance evaluation under FS scheme for each ML Methods

Testing Error Metrics	ML Methods	kNNI			MICE			MissForest		
		LVF	**CF**	**GA**	**LVF**	**CF**	**GA**	**LVF**	**CF**	**GA**
MAE	SVR	0.063	0.069	0.048	0.054	0.059	0.062	0.063	0.066	0.056
	MLP	0.039	0.044	0.039	0.046	0.044	0.042	0.039	0.039	0.033
	CART	0.020	0.023	**0.015**	0.017	0.023	0.022	0.024	0.025	0.021
	kNN	0.023	0.034	0.024	**0.016**	0.029	0.021	0.028	0.027	0.023
	RF	0.028	0.025	0.025	0.027	0.028	0.025	0.026	0.025	**0.024**
RMSE	SVR	0.071	0.080	0.059	0.063	0.073	0.068	0.071	0.077	0.064
	MLP	0.085	0.092	0.075	0.084	0.080	0.065	0.085	0.090	0.063
	CART	0.043	0.051	**0.037**	0.051	0.047	0.049	0.052	0.058	0.043
	kNN	0.045	0.071	0.048	**0.038**	0.066	0.045	0.057	0.056	0.047
	RF	0.050	0.047	0.044	0.048	0.052	0.045	0.047	0.048	**0.043**
R^2	SVR	0.294	0.098	0.510	0.434	0.243	0.356	0.296	0.166	0.427
	MLP	-0.027	-0.180	0.205	-0.002	0.092	0.395	-0.018	-0.141	0.431
	CART	0.739	0.636	**0.804**	0.636	0.680	0.656	0.618	0.517	0.733
	kNN	0.706	0.289	0.666	**0.797**	0.392	0.711	0.542	0.563	0.679
	RF	0.639	0.686	0.718	0.671	0.611	0.716	0.680	0.677	**0.732**
SD	SVR	0.059	0.041	0.071	0.070	0.048	0.060	0.062	0.050	0.066
	MLP	0.004	0.025	0.022	6.938	0.045	0.044	0.000	0.023	0.051
	CART	0.072	0.067	**0.076**	0.067	0.069	0.068	0.066	0.060	0.072
	kNN	0.063	0.032	0.051	**0.065**	0.039	0.056	0.046	0.053	0.054
	RF	0.051	**0.061**	0.056	0.050	0.048	0.055	0.053	0.053	**0.054**

roulette wheel selection, and elitism replacement were the GA settings. Because the outcomes are dependent on the population randomly produced by the GA algorithm, we ran 10 simulations for dataset.

The use of GA for subset selection can be a potential candidate when suitable parameters can be selected. However, sequential feature selection mostly selects 'Functional Size', 'Project Elapsed Time', 'Development Type', 'Business Area Type', 'Language Type', 'Client Server', 'First OS', and 'Max Team Size' ' features from the ISBSG dataset. GA, on the other hand, selects the majority of the features from the data. The population is 952, divided into 40 generations. At the 0.05 level, the mutations followed a uniform distribution. GA-based feature selection, which necessitates a 5-fold cross-validation step in order to find the optimal features.

In this study, we compare GA feature selection with low variance filter (LVF) and correlation filter (CF). LVF is a useful dimension reduction algorithm. Removing low variance features, that is, low variance filtering feature selection. Calculate the variance for each of the sample's feature values. Filter if it is less than the threshold (delete). All zero-variance features are disabled by default. In the delete low-variance feature method, the feature with a lower variance than $p(1-p)$ will be removed. In this study, only select columns that have a variance higher than 0.006. Meanwhile, the Pearson correlation is the most commonly used method for CF. First, we'll create a Pearson correlation heatmap and examine the correlation between the independent and output variables. Only features with a correlation of more than 0.5 (in absolute values) with the output variable will be considered. For the comparison results of the feature selection method, see Table 7.

As shown in the table, the lowest MAE (0.015) and RMSE (0.037) values were obtained using the CART algorithm for the kNNI imputation methods and GA feature selection. With R^2 (0.804) and SD (0.076) having the highest value, this shows that CART has the best performance. Even though, it is significant that these three feature selection methods have values that are close to the same. In combining the feature selection and imputation methods, it shows that CART has the best performance than SVR and MLP. While, for RF (MAE: 0.024; RMSE: 0.043; R^2: 0.732; SD: 0.054) have almost the same performance as CART, by using a combination of missForest as MDI and GA as FS. Meanwhile, kNN only had the best performance on the combination of MICE as MDI and LVF as FS (MAE: 0.016; RMSE: 0.038; R^2: 0.797; SD: 0.065). Thus, it can be concluded that CART, kNN, and RF have almost the same performance, both using a combination (LVF, CF, and GA as FS) with (kNNI, MICE, and missForest as MDI). Whereas MLP and SVR have the worst performance in this regard. Surprisingly, the results reveal that feature selection has no effect on the accuracy of the MLP and SVR approaches. Applying feature selection to MLP as a whole can

Received: August 2, 2021. Revised: September 18, 2021.

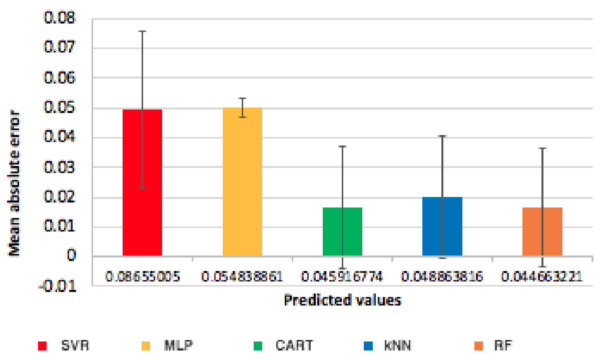

Figure. 7 Compare ML methods using MAE and prediction values

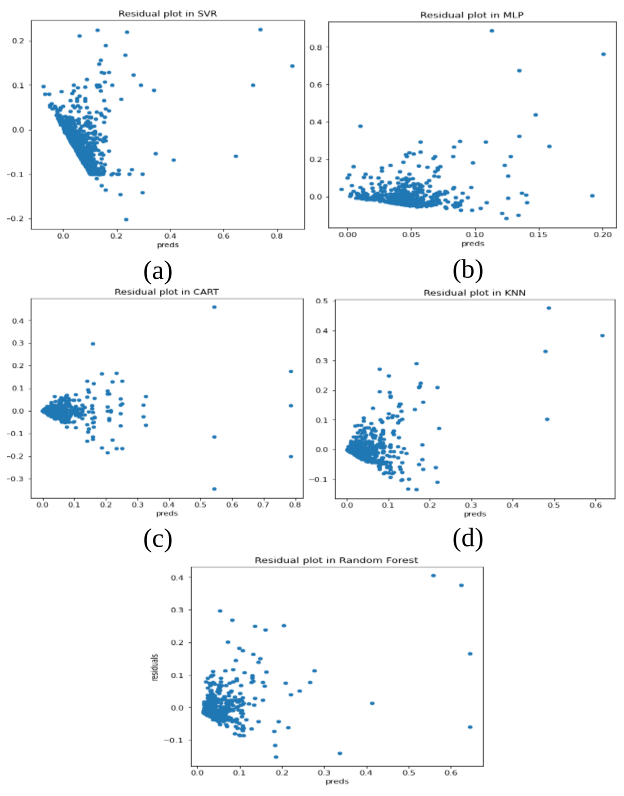

Figure. 8 Residual plot: (a) SVR, (b) MLP, (c) CART, (d) kNN, and (e) RF

lower the number of hidden nodes in data sets with a small number of features, reducing its approximation ability. This could be due to the limitations of the specific feature selection approach used in this study. SVR, on the other hand, is ineffective in dealing with outliers in training data, which are common in practical applications. As a result, some outliers result in poor regression.

This research demonstrates that using LVF or CF for feature selection does not always increase effort estimation accuracy. On the other hand, when suitable parameters can be chosen, using GA for subset selection can be a viable option. GA does well in a more complete search to find the optimal solution [54].

5.5 Overall performance ML

We conducted extensive simulation studies to evaluate each ML method. Based on Fig. 7, provides a calculation of the mean absolute error between the list of expected and predicted values. The perfect mean absolute error value is 0.0, which means that all predictions exactly match the expected values.

A graph is created to provide a basic statistical picture of the data showing the comparative value of the five ML performance. All variations on the graph were reviewed using the median on the MAE and predictive values. The results showed that the CART method had the best performance as indicated by the lowest MAE value, followed by RF and kNN. While the SVR and MLP methods have poor performance. However, in this study, it is shown that the dataset can significantly affect the accuracy of ML performance.

Based on Fig. 8, it shows that some of the characteristics of a good residual plot lie in the CART method, which is followed by kNN and RF. This can be seen because it has a high point density close to the origin and a low point density from the origin. Residue indicates the quality of the model. If the expected residual value is not close to 0, this implies that the model is systematically biased towards over or under prediction. In addition, if the residuals contain a pattern, it is likely that the model fails to explain some of the relationships in the data and is, therefore, qualitatively inconsistent. So, it is also necessary to check whether the residuals are normally distributed, homoscedastic, or heteroscedastic.

6. Conclusion

We learned that the imputation approach can be applied effectively to the DP technique (with four stages in the context of SEE) for attributes containing mostly categorical attributes with many missing values. His paper has empirically evaluated and compared the effectiveness of missing imputation of the three imputation methods (kNNI, MICE, and missForest) and the three feature selection methods (GA, LVF, and CF) aimed at addressing missing data and irrelevant features. An empirical study was conducted using the ISBSG dataset. The ML model was assessed using error metrics (MAE, RMSE, R^2, and SD).

The results, suggest that kNNI and MICE may be more effective than MissForest (based on mean). MissForest is only slightly better under kNNI and MICE with larger standard deviation values, on the other hand, indicating that in some cases, MissForest's accuracy may be superior. However, kNNI, MICE and missForest generally have almost the same performance results.

Meanwhile, for the ML model test on the imputation method the results show that kNNI-CART (MAE: 0.018; RMSE: 0.042; R^2: 0.751; SD: 0.073) and missForest-CART (MAE: 0.019; RMSE: 0.040; R^2: 0.774; SD: 0.074) has the best performance accuracy improvement. On the other hand, MICE-kNN (MAE: 0.016; RMSE: 0.038; R^2: 0.797; SD: 0.065) and missForest-RF (MAE: 0.024; RMSE: 0.044; R^2: 0.718; SD: 0.054) had the best performance accuracy improvement. Whereas MLP and SVR have poor performance in this regard.

Finally, for the performance of the feature selection GA works well in a more complete search to find the optimal solution. On the other hand, the use of LVF or CF for feature selection does not always improve the accuracy of the effort estimate. Nevertheless, it can be concluded that the application of three feature selection methods to the ML model such as: CART, kNN, and RF can improve performance accuracy. Surprisingly, the results reveal that feature selection has no effect on the accuracy of the MLP and SVR approaches.

As future work, more empirical studies can be carried out to further support the findings of this study and to gather knowledge using other datasets. Investigating other types of imputation methods, or performing combinations using ensemble learning, is another direction for future work. It is also important to investigate the effectiveness of other feature selection methods or reinforce with optimization parameters on the software effort estimation.

Conflicts of interest

The authors declare no conflict of interest.

Author contributions

Conceptualization, R. Marco; methodology, R. Marco; validation, S. S. S. Ahmad and S. Ahmad; formal analysis, R. Marco, S. S. S. Ahmad and S. Ahmad; investigation, R. Marco; resources, R. Marco, S. S. S. Ahmad and S. Ahmad; data curation, R. Marco; writing—original draft preparation, R. Marco; writing—review and editing, S. S. S. Ahmad and S. Ahmad; visualization, R. Marco; supervision, S. S. S. Ahmad and S. Ahmad; funding acquisition, R. Marco.

References

[1] Y. Mahmood, N. Kama, and A. Azmi, "A systematic review of studies on use case points and expert-based estimation of software development effort", *Journal of Software: Evolution and Process*, Vol. 32, No. 7, pp. 1–20 , 2020.

[2] M. Usman, K. Petersen, J. Börstler, and P. S. Neto, "Developing and using checklists to improve software effort estimation: A multi-case study", *Journal of Systems and Software*, Vol. 146, pp. 286–309, 2018.

[3] S. Ezghari and A. Zahi, "Uncertainty management in Software effort estimation using a consistent fuzzy analogy-based method", *Journal of Applied Soft Computing*, Vol. 67, pp. 540–557, 2018.

[4] M. Azzeh, D. Neagu, and P. I. Cowling, "Fuzzy grey relational analysis for software effort estimation", *Journal of Empirical Software Engineering*, Vol. 15, No. 1, pp. 60–90, 2010.

[5] A. Özkaya, E. Ungan, and O. Demirörs, "Common practices and problems in effort data collection in the software industry", In*: Proc. of International Conf. on Software Measurement,* Nara, Japan, pp. 308–313, 2011.

[6] I. Abnane, A. Idri, and A. Abran, "Fuzzy case-based-reasoning-based imputation for incomplete data in software engineering repositories", *Journal of Software: Evolution and Process*, Vol. 32, No. 9, pp. 1–25, 2020.

[7] I. Myrtveit, E. Stensrud, and U. H. Olsson, "Analyzing data sets with missing data: An empirical evaluation of imputation methods and likelihood-based methods", *Journal of Transactions on Software Engineering*, Vol. 27, No. 11, pp. 999–1013, 2001.

[8] S. K. Sehra, Y. S. Brar, N. Kaur, and S. S. Sehra, "Research patterns and trends in software effort estimation", *Journal of Information and Software Technology*, Vol. 91, pp. 1–21, 2017.

[9] F. A. Amazal, A. Idri, and A. Abran, "An Analogy-Based Approach to Estimation of Software Development Effort Using Categorical Data", In*: Proc. of International Conf. on Software Measurement,* Rotterdam, Netherlands, pp. 252–262, 2014.

[10] L. Gondara and K. Wang, "MIDA: Multiple imputation using denoising autoencoders", *Journal of Advances in Knowledge Discovery and Data Mining,* Vol. 10939, pp. 260–272, 2018.

[11] R. J. A. Little and D. B. Rubin, "The Analysis of Social Science Data with Missing Values", *Journal of Sociological Methods and Research*, Vol. 18, No. 2–3, pp. 292–326, 1989.

[12] P. J. G. Laencina, J. L. S. Gómez, A. R. F. Vidal, and M. Verleysen, "K nearest neighbours with mutual information for simultaneous

classification and missing data imputation", *Journal of Neurocomputing*, Vol. 72, No. 7–9, pp. 1483–1493, 2009.

[13] M. L. Yadav and B. Roychoudhury, "Handling missing values: A study of popular imputation packages in R", *Journal of Knowledge-Based Systems*, Vol. 160, pp. 104–118, 2018.

[14] A. Idri, I. Abnane, and A. Abran, "Systematic mapping study of missing values techniques in software engineering data", In: *Proc. of International Conf. on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing,* Takamatsu, Japan, 2015.

[15] J. Huang, Y. F. Li, and M. Xie, "An empirical analysis of data preprocessing for machine learning-based software cost estimation", *Journal of Information and Software Technology,* Vol. 67, pp. 108–127, 2015.

[16] J. Huang, Y. F. Li, J. W. Keung, Y. T. Yu, and W. K. Chan, "An empirical analysis of three-stage data-preprocessing for analogy-based software effort estimation on the ISBSG data", In: *Proc. of International Conf. on Software Quality, Reliability and Security,* Prague, Czech Republic, pp. 442–449, 2017.

[17] L. L. Minku and X. Yao, "A Principled Evaluation of Ensembles of Learning Machines for Software Effort Estimation Categories and Subject Descriptors", In: *Proc. of International Conf. on Predictive Models in Software Engineering,* Banff Alberta, Canada, 2011.

[18] A. Idri, I. Abnane, and A. Abran, "Missing data techniques in analogy-based software development effort estimation", *Journal of Systems and Software*, Vol. 117, pp. 595–611, 2016.

[19] G. E. A. P. A. Batista and M. C. Monard, "A study of k-nearest neighbour as an imputation method", *Journal of Frontiers in Artificial Intelligence and Applications*, Vol. 87, pp. 251–260, 2002.

[20] J. Li, A. A. Emran, and G. Ruhe, "Impact Analysis of Missing Values on the Prediction Accuracy of Analogy-based Software Effort Estimation Method AQUA", In: *Proc. of International Conf. on Empirical Software Engineering and Measurement (ESEM),* Madrid, Spain, pp. 126–135, 2007.

[21] S. A. N. Alexandropoulos, S. B. Kotsiantis, and M. N. Vrahatis, "Data preprocessing in predictive data mining", *Journal of The Knowledge Engineering Review,* Vol. 34, pp. 1–33, 2019.

[22] A. Idri, I. Abnane, and A. Abran, "Support vector regression-based imputation in analogy-based software development effort estimation", *Journal of Software: Evolution and Process,* Vol. 30, No. 12, pp. 1–23, 2018.

[23] Q. Song, M. Shepperd, X. Chen, and 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,* Vol. 81, No. 12, pp. 2361–2370, 2008.

[24] A. Jović, K. Brkić, and N. Bogunović, "A review of feature selection methods with applications" , In: *Proc. of International Conf. on Information and Communication Technology, Electronics and Microelectronics,* Opatija, Croatia, pp. 1200–1205, 2015.

[25] M. Hosni, A. Idri, and A. Abran, "Investigating Heterogeneous Ensembles with Filter Feature Selection for Software Effort Estimation", In: *Proc. of International Conf. On Software Process and Product Measurement,* Gothenburg, Sweden, No. 2, 2017.

[26] P. L. Braga, A. L. I. Oliveira, and S. R. L. Meira, "A GA-based Feature Selection and Parameters Optimization for Support Vector Regression Applied to Software Effort Estimation", In: *Proc. of International Conf. On on Applied computing,* Fortaleza, Brazil, pp. 1788–1792, 2008..

[27] Y. Mahmood, N. Kama, A. Azmi, A. S. Khan, and M. Ali, "Software effort estimation accuracy prediction of machine learning techniques: A systematic performance evaluation", *Journal of Software: Practice and Experience*, 2021.

[28] S. Tariq, M. Usman, and A. C. M. Fong, "Selecting best predictors from large software repositories for highly accurate software effort estimation", *Journal of Software: Evolution and Process* , Vol. 32, No. 10, pp. 1–19, 2020.

[29] K. Strike, K. E. Emam, and N. Madhavji, "Software cost estimation with incomplete data", *Journal of Transactions on Software Engineering,* Vol. 27, No. 10, pp. 890–908, 2001.

[30] M. H. Cartwright, M. J. Shepperd, and Q. Song, "Dealing with missing software project data" , In: *Proc. of International Conf. On Enterprise Networking and Computing in Healthcare Industry,* Sydney, Australia, pp. 154–165, 2003.

[31] B. Twala and M. Cartwright, "Ensemble imputation methods for missing software engineering data", In: *Proc. of International Conf. On Software Metrics Symposium,* Como, Italy, pp. 271–280, 2005.

[32] P. Sentas and L. Angelis, "Categorical missing data imputation for software cost estimation by multinomial logistic regression", *Journal of*

Systems and Software, Vol. 79, No. 3, pp. 404–414, 2006.

[33] I. Abnane, M. Hosni, A. Idri, and A. Abran, "Analogy Software Effort Estimation Using Ensemble KNN Imputation", In: *Proc. of International Conf. on Software Engineering and Advanced Applications*, Kallithea, Greece, pp. 228–235, 2019.

[34] E. Kocaguneli, T. Menzies, J. Hihn, and B. H. Kang, "Size doesn't matter? On the value of software size features for effort estimation", In: *Proc. of International Conf. on Predictive Models in Software Engineering*, Lund, Sweden, pp. 89–98, 2012.

[35] L. L. Minku and X. Yao, "Ensembles and locality: Insight on improving software effort estimation", *Journal of Information and Software Technology* , Vol. 55, No. 8, pp. 1512–1528, 2013.

[36] S. Viaene, G. Dedene, and R. A. Derrig, "Auto claim fraud detection using Bayesian learning neural networks", *Journal of Expert Systems with Applications*, Vol. 29, No. 3, pp. 653–666, 2005.

[37] E. F. Norris, S. Vahid, and C. Hand, "Evaluating the Impact of Categorical Data Encoding and Scaling on Neural Network Classification Performance: The Case of Repeat Consumption of Identical Cultural Goods", *Journal of Communications in Computer and Information Science*, Vol. 311, pp. 343–0352, 2012.

[38] L. Angelis and I. Stamelos, "A Simulation Tool for Efficient Analogy Based Cost Estimation", *Journal of Empirical Software Engineering*, Vol. 5, No. 1, pp. 35–68, 2000.

[39] S. G. K. Patro and K. K. Sahu, "Normalization: A Preprocessing Stage", *Journal of arXiv*, pp. 20–22, 2015.

[40] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. B. Altman, "Missing value estimation methods for DNA microarrays", *Journal of Bioinformatics*, Vol. 17, No. 6, pp. 520–525, 2001.

[41] A. Choudhury and M. R. Kosorok, "Missing Data Imputation for Classification Problems", *Journal of arXiv*, pp. 1–27, 2020.

[42] T. Raghunathan, J. Lepkowski, J. V. Hoewyk, and P. Solenberger, "A multivariate technique for multiply imputing missing values using a sequence of regression models", *Journal of Survey methodology*, Vol. 27, No. 1, pp. 85–96, 2001.

[43] J. L. Schafer and J. W. Graham, "Missing data: Our view of the state of the art", *Journal of Psychological Methods*, Vol. 7, No. 2, pp. 147–177, 2002.

[44] L. Breiman, "Random forests", *Journal of Machine Learning*, Vol. 45, pp. 1–122, 2001.

[45] D. J. Stekhoven and P. Bühlmann, "Missforest-Non-parametric missing value imputation for mixed-type data", *Journal of Bioinformatics*, Vol. 28, No. 1, pp. 112–118, 2012.

[46] F. Tan, X. Fu, Y. Zhang, and A. G. Bourgeois, "A genetic algorithm-based method for feature subset selection", *Journal of Soft Computing* , Vol. 12, No. 2, pp. 111–120, 2008.

[47] A. Corazza, S. D. Martino, F. Ferrucci, C. Gravino, and E. Mendes, "Investigating the use of Support Vector Regression for web effort estimation", *Journal of Empirical Software Engineering*, Vol. 16, No. 2, pp. 211–243, 2011.

[48] A. Subasi, "Application of adaptive neuro-fuzzy inference system for epileptic seizure detection using wavelet feature extraction" , *Journal of Computers in Biology and Medicine* , Vol. 37, No. 2 , pp. 227–244 , 2007.

[49] E. D. Übeyli, "Combined neural networks for diagnosis of erythemato-squamous diseases", *Journal of Expert Systems with Applications*, Vol. 36, No. 3, pp. 5107–5112, 2009.

[50] P. C. Pendharkar, G. H. Subramanian, and J. A. Rodger, "A probabilistic model for predicting software development effort", *Journal of Transactions on Software Engineering*, Vol. 31, No. 7, pp. 581–588, 2005.

[51] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression", *Journal of American Statistician*, Vol. 46, No. 3, pp. 175–185, 1992.

[52] M. Hosni, A. Idri, A. Abran, and A. B. Nassif, "On the value of parameter tuning in heterogeneous ensembles effort estimation", *Journal of Soft Computing* , Vol. 22, No. 18, pp. 5977–6010, 2018.

[53] S. Mensah, J. Keung, M. F. Bosu, and K. E. Bennin, "Duplex output software effort estimation model with self-guided interpretation", *Journal of Information and Software Technology* , Vol. 94, pp. 1–13, 2018.

[54] B. K. Khotimah, M. Miswanto, and H. Suprajitno, "Optimization of feature selection using genetic algorithm in naïve Bayes classification for incomplete data", *Journal of Intelligent Engineering and Systems*, Vol. 13, No. 1, pp. 334–343, 2020.