Proceedings of the Federated Conference on Computer Science and Information Systems pp. 1015–1024
DOI: 10.15439/2018F95
ISSN 2300-5963 ACSIS, Vol. 15

Improved Analogy-based Effort Estimation with Incomplete Mixed Data

Ibtissam Abnane and Ali Idri
Software Project Management Research Team,
ENSIAS, University Mohammed V, Rabat, Morocco
{Ibtissam_abnane, ali.idri }@um5.ac.ma

***Abstract*—Estimation by analogy (EBA) is one of the most attractive software effort development estimation techniques. However, one of the critical issues when using EBA is the occurrence of missing data (MD) in the historical data sets. The absence of values of several relevant software attributes is a frequent phenomenon that may cause inaccurate EBA estimations. The MD can be numerical and/or categorical. This paper evaluates four MD techniques (toleration, deletion, k-nearest neighbors (KNN) imputation and support vector regression (SVR) imputation) over four mixed data sets. A total of 432 experiments were conducted involving four MD techniques, nine MD percentages (from 10% to 90%), three missingness mechanisms (MCAR: Missing Completely at Random, MAR: Missing at Random and NIM: Non-Ignorable Missing) and four data sets. The evaluation process consists of four steps and uses several accuracy measures such as standardized accuracy (SA) and prediction level (Pred).**

The results suggest that EBA with imputation techniques achieved significantly better SA values over EBA with toleration or deletion regardless of the mechanism of missingness. Moreover, no particular MD imputation technique outperformed the other techniques overall. However, according to Pred and other accuracy criteria, EBA with SVR was the best, followed by KNN imputation; we also found that toleration instead of deletion improves the accuracy of EBA.

***Index Terms*—Estimation by analogy, missing data, imputation.**

I. Introduction

SOFTWARE development effort estimation (SDEE) is the process of predicting the effort required to develop a software system. It is a challenging and substantial activity when managing a software project. The challenge arises due to the complex relationship between effort and various software attributes related to the personal, product, and/or platforms used in the project [1], [2].

Machine learning (ML) based estimation techniques are gaining increasing attention in SDEE research, as they can model the complex relationship between effort and software attributes (cost drivers), especially when this relationship is not linear and does not seem to have any predetermined form [2]. Estimation by Analogy (EBA) is one of the most attractive ML techniques in the SDEE field, and is essentially a form of Case-Based Reasoning (CBR)[3]. The idea of analogy based estimation is to determine the effort of the new project as a function of the known efforts from similar historical projects. Wen et al. [2] carried out a systematic literature review of ML SDEE techniques published between 1991 and 2010 and found that EBA is the most investigated ML technique in SDEE (37% of selected studies).

The intensive and increasing use of EBA is due to its several advantages including simplicity, mimicking human reasoning, ease to understand and no assumption is made about the form of the relationship [1], [4]–[10]. Moreover, EBA can handle both quantitative and qualitative data [5]–[7], [11], [12]. Nonetheless, several studies pointed out that EBA still has some limitations such as dealing with missing data (MD) which is a widespread problem that can affect the ability to use data to construct effective EBA techniques [3], [8], [13], [14]. However, little attention has been given to handling missing data in EBA [3]. In fact, the mapping study of Idri et al.]21] found that until 2015, only one paper was published to deal with MD in EBA.

In a prior work [8], we evaluated two EBA techniques in terms of SA criterion on seven data sets when used in conjunction with three MD techniques (toleration, deletion and KNN imputation method), different missingness mechanisms (MCAR, MAR and NIM) and nine percentages of MD (from 10% to 90%) [8]. This was with the aim of determining whether the KNN imputation method, instead of deletion and toleration techniques, could improve the performance of EBA when predicting software development effort with incomplete data. The findings suggest that EBA using KNN imputation outperformed EBA using deletion or toleration regardless of the missingness mechanism and the MD percentage.

However, the study [8] has three limitations: (1) it only dealt with numerical data , (2) it used one imputation technique (KNN), and (3) it used one accuracy criterion (SA), which is insufficient to conclude about EBA accuracy [15], [16].

Thus, this paper improves our previous work with: (1) the use of both numerical and categorical data, 2) the use of a new imputation technique: Support Vector Regression (SVR), in addition to KNN, and (3) the use of a set reliable accuracy criteria (e.g.. Pred(0.25), Mean Absolute Error (MAE), Mean Balanced Relative Error (MBRE), Mean Inverted Balanced Relative Error (MIBRE) and Logarithmic Standard Deviation (LSD)), in addition to SA, in order to investigate if they would confirm the findings of [8].

Therefore, this study carry out an empirical evaluation of EBA using four MD techniques: toleration, deletion, KNN imputation, and SVR imputation with different percentages (from 10% to 90%) and three missingness mechanisms (MCAR, MAR and NIM) over four mixed datasets including

both numerical and categorical attributes (ISBSG R8, COCOMO81, USP05_FT and USP05_RQ).
Toward this aim, four research questions were discussed (RQs):

- (RQ1) Is there evidence that the use of KNN and SVR imputations rather than toleration/deletion improves the accuracy of EBA in terms of SA when using mixed datasets?
- (RQ2) Is there evidence that SVR imputation instead of KNN imputation would improve EBA accuracy measured in terms of SA?
- (RQ3) Is there evidence that the missingness mechanism and the MD percentage affect the accuracy of EBA over mixed datasets?
- (RQ4) Does the performance of EBA in terms of Pred(0.25), MAE, MBRE, MIBRE and LSD confirm the findings of SA?

The structure of the paper is the following: Section II presents the concepts of MD and EBA. Section III describes the data sets used. Section IV presents the empirical design which includes the process of generate MD and the empirical evaluation process. The results are presented and discussed in Section V. Section VI concludes by discussing the findings as well as some directions for future work.

II. BACKGROUND

This section presents the concepts of MD and an overview of the software effort estimation by analogy process we used in this study.

A. Concepts of MD

This section gives an overview of the different missingness mechanisms (i.e., different ways in which data can be missing) and the different MD techniques.

1) Missigness mechanism

Understanding the missing data mechanism is a key stage in comprehending the impact of the missing data on a specific analysis, or missing data methods [17], [18]. Rubin's classification of Missing Data Mechanisms has been regarded as being "fundamental to the modeling of incomplete data" [19] and is in common use in the field of missing data. Little and Rubin classified missing data mechanisms as[17]:

- **Missing completely at random (MCAR)** is when the probability that an observation is missing does not depend on either the observed or the missing values.
- **Missing at Random (MAR)** means that the probability that an observation is missing depends only on the values of the observed data.
- **Non-Ignorable Missing (NIM):** means that the missing data mechanism is related to the missing values.

2) MD techniques

There are three approaches to this problem: MD deletion technique, MD toleration techniques, and MD imputation techniques [13], [20].

a) Toleration

MD toleration technique is an embedded strategy in which analysis is performed directly on the data set with MD[8], [18]. Despite its simplicity, toleration is not a reliable approach, sometimes even providing estimates that are less efficient than estimation from deletion technique [8], [17].

b) Deletion technique

Deletion is the most commonly used technique for dealing with missing data among researchers [8], [21]. It omits all cases with missing values from the analysis and only includes those cases for which all measurements are observed. This method has many advantages since it is easy to use. Also, it produces unbiased estimates for the parameters if the assumption that the data are MCAR holds. Nevertheless, deletion is not generally recommended since omitting cases with MD would result in a significant loss in power and precision due to the reduced sample size. Moreover, if the MCAR assumption does not hold, this method can result in biased parameter estimates as it is ignoring potential systematic differences between the complete and incomplete cases. Consequently, deletion can only be justified if the missing data mechanism in operation is MCAR and the MD percentage is small [22].

c) Imputation technique

MD imputation replaces missing values by suitable estimates and then applies standard complete-data methods to the filled in data [17].This method is attractive to practitioners because the resulting completed data can be handled using standard software for rectangular data sets. Imputation uses available data to impute the missing data and hence, an important characteristic of a good imputation method is that it makes good use of information in the incomplete cases. Moreover, it is important to take into account the missingness mechanism while using imputation technique [8], [18].

B. Effort estimation by analogy: An Overview

EBA is based on the use of historical information from completed projects with known effort [10]. It is based on the following affirmation: *similar software projects have similar costs*. It has been deployed as follows. EBA has been proposed since a long time as a valid alternative to effort estimation by parametric effort estimation and/or expert judgment [23]. In 1997, it has been presented in the form of a detailed estimation methodology and has been applied on a set of historical software projects data sets [10]. EBA consists of three steps:

1) Identification of cases: each project is described by a set of attributes that are believed to be significant in determining similarity and can influence effort.
2) Retrieval of similar cases: several distance metrics can be used to calculate how much the new target project differs from the other projects based on their attribute values. In this study, we used the Euclidean and the overlap distances for numerical and categorical software attributes respectively [10], [24].
3) Case adaptation: involves two phases in order to generate an estimate of the new project. First, we decide on the number of similar projects and second we define the

adaptation strategy. The number of analogues (*k)* refers to the number of most similar projects used to generate the estimation. Several studies in SDEE analyzed the impact of the number of analogues [9], [25], [26]. This study varied the number of analogues between 2 and 5. The second phase consists on selecting the adaptation strategy to provide an effort estimate. We used the arithmetic mean [10], arithmetic median [27] and inverse ranked weighted mean [28].

In order to select the best variant of ABE, we varied the adaptation strategy and the number of analogues as described above and chose the best configuration of ABE, i.e. having the lowest Mean of Absolute Error (MAE).

III. DATA DESCRIPTION

This study uses four available data sets: ISBSG repository (release 8), COCOMO81[23], USP05_FT[9] and USP05_RQ[9]. Table I provides an overview of these datasets, including number of attributes (numerical and categorical), observations, and previous use. The minimum, mean and maximum of effort and size are given.

Since the aim of this study is to deal with missing numerical and categorical data, the solution adopted was to use (1) all the attributes for the COCOMO81 data set, (2) 11 attributes for USP05_FT and USP05_RQ data sets, and (3) 20 attributes for the ISBSG data set. The attributes chosen for USP05_FT, USP05_RQ and ISBSG data sets are the results of our previous studies related to software effort estimation [4][7][8] . Table III shows the attributes chosen for ISBSG, USP05 and COCOMO81 data sets, where (N) and (C) indicate numerical and categorical attributes respectively.

IV. EXPIREMENT DESIGN

This section describes the experimental process used in this study. It consists of four main steps: data removal, complete data set generation, EBA evaluation using SA, and EBA evaluation using Borda count method based on Pred(0.25), MAE, MBRE, MIBRE and LSD. A similar process was followed in [8]. The study was designed to apply EBA with nine percentages of incomplete mixed data (from 10% to 90%), three different MD mechanisms (MCAR, MAR and NIM), and four MD techniques (toleration, deletion, KNN imputation, and SVR imputation) over four data sets. Hence, the experimental design consists in evaluating 9 percentages × 4 MD techniques × 3 MD mechanisms × 4 datasets = 432 different effort estimation experiments.

A. Step 1: Data removal

The first step in the experimental process requires a complete data set to work with. For this purpose, we first preprocessed the four datasets by deleting cases with MD to obtain the corresponding seven complete data sets. We then used the complete datasets to artificially generate MD by mimicking the different mechanisms. A similar approach was followed in [8], [20]. By combining the four datasets, three missingness mechanisms and nine percentages, we obtained 4×3×9 =108 incomplete data sets at this stage.

B. Step 2: Complete data set generation

This step uses four MD techniques: toleration, deletion, KNN imputation and SVR imputation to generate complete datasets from those of Step 1. After applying the four MD techniques on the 108 incomplete datasets of step 1, we obtained 432 complete datasets.

a) Deletion

Under the deletion technique, projects with missing values at any attribute are omitted in the experiments.

b) Toleration

The toleration technique uses a special value *NULL* to replace a missing value in a data set. A similar approach was used in [8], [13], [14]. Hence, the following operations on *NULL* are defined for distance metrics:

(P_1) $\delta(\text{b}, \text{NULL}) = \delta(\text{NULL}, \text{b}) = (\text{NULL}, \text{NULL}) = \text{NULL}$

(P_2) W+ NULL = NULL + w = w

where δ is the distance used in Classical Analogy (e.g. Euclidean distance).

It can be seen from the above discussion that the effect of the *NULL* is to ignore the participating attributes that have MD in searching similar objects. Therefore, the more *NULLs* in the data set, the fewer attributes will be participating in searching analogues through similarity measures.

c) KNN imputation

Figure 1 shows the KNN imputation (KNNI) process. KNN imputation belongs to the analogy based algorithms; it is computationally simple and has proven to be effective approach to estimate missing values of attributes in different software engineering datasets [21]. Using KNN for imputation requires adapting the three analogy steps of Section II.B: (1) Identification of cases, (2) Retrieval of similar projects, and (3) Case adaptation. In the following, we present how we adapted each step to serve the imputation process.

The identification of cases step mainly aims to calculate the distance between each incomplete project and the complete projects. The most similar complete projects were used as source analogues in the imputation process. To determine the distance between the incomplete project and the complete projects, we use a combination of two distance measures. Hence, the distance between an incomplete case P_i and a complete case P_j is calculated using Equation (1).

$$\mathbf{d}(\mathbf{P_i}, \mathbf{P_j}) = \mathbf{d_n}(\mathbf{P_i}, \mathbf{P_j}) + \mathbf{d_c}(\mathbf{P_i}, \mathbf{P_j}) \qquad (\mathbf{1})$$

where:

- $d_n(P_i, P_j)$is the Euclidean distance used to calculate the similarity between P_i and P_j taking into consideration only the numerical attributes. It is calculated using Equation (2):

$$d_n(P_i, P_j) = \sqrt{\sum_{l=1}^{n}(P_{il} - P_{jl})^2} \qquad (2)$$

- $d_c(P_i, P_j)$ is the hamming distance used to evaluate the similarity between P_i and P_j taking into consideration only the categorical attributes. The formula of $d_c(P_i, P_j)$ is given by Equation (3):

$$d_c(P_i, P_j) = \sum_{l=1}^{n} \delta(P_{il}, P_{jl}) \qquad (3)$$

TABLE I Selected Software Attributes from ISBSG, USP05, and COCOMO81 Data Sets. (N: numerical and C: categorical)

ISBSG		USP05	COCOMO81	
Value adjustment factor (N)	Reference table approach (C)	Data file (N)	KDSI (N)	VEXP (C)
Maximum team size (N)	Recording method (C)	Data entry (N)	RELAY (C)	LEXP (C)
User-based business units (N)	Development platform (C)	Data output (N)	DATA (C)	MODP (C)
User-based locations (N)	Programming language (C)	Unadjusted function point (N)	CPLX (C)	TOOL (C)
User-based concurrent users (N)	Used methodology (C)	Internal complexity (C)	TIME (C)	SCED (C)
Input count (N)	Development technique (C)	Language (C)	STOR (C)	MODE (C)
Output count (N)	Organization type (C)	Tools (C)	VIRT (C)	
Enquiry count (N)	Business area type (C)	Applications experience (C)	TURN (C)	
Interface count (N)	Application type (C)	Database systems (C)	ACAP (C)	
Measurement technique (C)		Methodology (C)	AEXP (C)	
		Application type (C)	PCAP (C)	

TABLE II Description Statistics of the Selected Data Sets

Data set	#of Projects	#of attributes	Effort					
			Min	Max	Median	Mean	Skewness	Kurtosis
USP05RQ	102	11	0.5	50	3	8.05	2.01	3.60
USP05FT	58	11	0.5	24	1	3.21	3.03	8.84
ISBSG	89	20	24	36286	2101	3779.52	3.49	14.74
COCOMO81	63	17	6	11400	98	683.44	4.47	21.87

where $\delta(P_{i1}, P_{j1})$ is the hamming distance between P_{i1} and P_{j1}. The Hamming distance between two sets of binary digits is the number of corresponding binary digit positions that differ divided by the number of comparisons made [31]

The case adaptation step matches the imputation phase. First, we decide on the number of analogous projects, *k*. we varied k from 1 to 5. Thereafter, to impute the missing values, we had to decide also on the adaptation strategy. For numerical attributes, we choose the weighted mean since it allows the higher similar projects to have more influence than the lower ones. For categorical attributes, we imputed a missing value with the attribute value of the most similar project to the incomplete project.

d) SVR imputation

Support vector machine has been developed by [32] and it is a supervised learning approach based on statistical theory. It has been gaining popularity due to its attractive features and promising empirical performance. Based on the structural risk minimization (SRM) principle, SVM is able to control the complexity of the model and its generalization ability, which can be used for solving two-class or multi-class classification and regression problems in various fields [33]–[35]. SVM possesses many advantages including fast-learning, global optimization, and excellent generalization abilities due to minimizing the tradeoff between the complexity of the model and its generalization ability compared with other approaches such as artificial neural networks [36], linear regression and radial Basis functions neural networks (RBFNs) [37].

With the introduction of Vapnik's ε-insensitivity loss function, the regression model of SVMs, called support vector regression (SVR), has received increasing attention to solve nonlinear regression problems. The investigation of SVR for software development effort estimation was originally carried out by Oliveira [37]. They found that SVR outperforms both linear regression and RBFNs for software effort estimation over NASA data set.

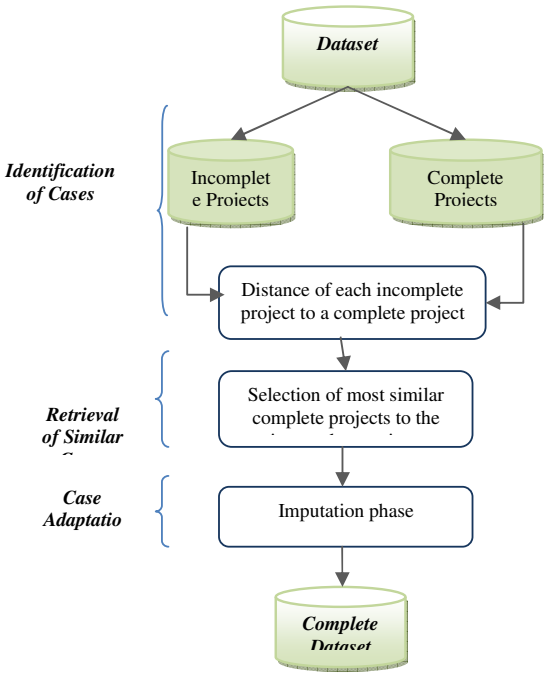

Figure 1 KNN imputation process

The main challenge when dealing with SVR is how to solve the dual problem [38]. The traditional quadratic programming (QP) algorithms solvers are slow, particularly for large problems[34], [35]. In addition, those algorithms can be complex, subtle, and difficult for an engineer to implement [34].

Specific algorithms were developed in order to make easier the use of SVR, such as Vapnik's chunking [39] and Osuna's decomposition [40]. Those algorithms make the training of SVR possible by breaking the large QP problem into a series of smaller QP ones and optimizing only a subset of training data patterns at each step. The subset of training data patterns optimized at each step is called the working set. Thus, these approaches are categorized as the working set methods. Based on the idea of the working set methods, [34] proposed the Sequential Minimal Optimization (SMO) algorithm that selects the size of the working set as two and uses a simple analytical approach to solve the reduced smaller QP problems. Thereafter, [38] ascertained inefficiency associated with Platt's SMO and suggested a modified version of SMO that can solve the SVR QP problem without any extra matrix storage and without using numerical QP optimization steps at all. Hence, this work uses the SMO-SVR algorithm of [38]. Moreover, An important factor that influences the performance of SVR is how to adequately select model parameters (C, ε, γ), which play an important role for a good generalization performance [41]. This paper uses a selection methodology based on Particle Swarm Optimization (PSO) to search global solutions of the optimal parameters (ε, C, γ) [42], [43].

Before proceeding to imputation, SVR transforms categorical variable into numerical ones. In fact, SVR maps each possible value for a categorical attribute into a number.

Unlike KNN imputation technique, SVR imputation (SVRI) requires building a model for each missing value in the dataset. Fig.2 shows the imputation method based on SMO-SVR. Let X be a N×D matrix of N projects described by d *attributes*. For each attribute i, we construct:

1. A complete dataset, Complete_X_i containing all projects P_j for which the values $x_{j,i}$ were not missing.
2. An incomplete dataset, Incomplete_X_i, containing all projects P_j for which the values $x_{j,i}$ were missing

Next, the i^{th} attribute X_i is set as the dependent attribute of Complete_X_i and Incomplete_X_i datasets. Then, the SMO-SVR model is trained using Complete_X_i. Firstly, the PSO algorithm is used to determine the optimal values of ε, C and γ. Next, those optimal values were used to train the SMO-SVR model. Finally, the missing values were imputed by the SMO-SVR model by using the Incomplete_X_i as the test set.

C. Step 3: EBA accuracy evaluation using SA

The accuracy of EBA was assessed using the Jackknife method in which the target project is excluded from the historical dataset and its effort estimation is calculated using the actual effort values of the remaining projects[44].

The accuracy of EBA was assessed in four steps:

1) Evaluation using SA

The first evaluation step aims to compare the accuracy of EBA with random guessing using the Standardized Accuracy (SA) suggested by Shepperd and MacDonell [45]. SA evaluates how good a SDEE technique is in comparison to random guessing. It is based on the Mean of Absolute Error (MAE) and is defined by Equation (4):

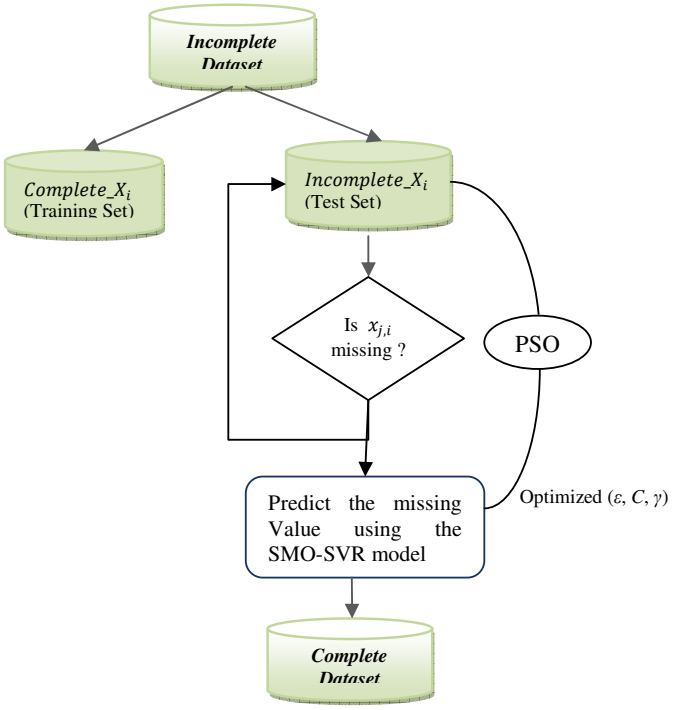

Figure 2 SVR imputation process

$$SA_{P_i} = 1 - \frac{MAE_{P_i}}{\overline{MAE}_{p0}} \times 100 \quad (4)$$

where MAE_{P_i} is defined to be the MAE of the estimation method P_i and MAE_{P_0} is the mean of a large number of (in our case 1000) random guessing. In the random guessing procedure, a training instance is randomly chosen with equal probability from the training set and its effort value is used as the estimate of the test instance. SA gives us an idea of how good an estimation method is in comparison to random guessing. Since the term MAE_{P_i} is in the nominator, the higher the SA values, the better an estimation method is.

The interpretation of SA is that the ratio represents how much better it is as a predictive model (P_i) than the mean or random guessing (P_0). A value close to zero is discouraging and a negative value would be worrisome. The positive sign of SA means the predictive models are better than mean or random guessing. Meanwhile the negative sign is shows how bad the predictive models are against the mean as an estimator. Unlike MRE-based error measures which have been criticized for being biased and favoring underestimation, SA is an unbiased and standardized accuracy measure.

2) Hypothesis testing

The second step aimed to statistically investigate the significance of the results found in step 1. Awareness about statistical validation of the published results had increased among machine learning researchers, in particular is software effort estimation [46].

Hypothesis testing is the process of inferring from a sample whether or not a given statement about the population appears to be true [47]. The first step in hypothesis testing is

establishing a null hypothesis. The null hypothesis is typically a statement contrary to what the researcher is attempting to confirm; we assume the null hypothesis to be true, and use data to try and refute it [47]. The statement that the researcher would like to prove is called the alternative hypothesis. We often establish a significance level (i.e. α-levels). It is is a limit on how unusual a result we will accept. An α-level of 0.05 means that if our observations from our collected data would occur less than 5% of the time given that the null hypothesis is true, then we will reject the null hypothesis [47].

However, null hypothesis testing is not sufficient to analyze and interpret data. A more refined goal of statistical analysis is to provide an evaluation of certainty or uncertainty of the size of an effect [45],[48]. The American Psychological Association (2001) has suggested that researchers report the confidence interval for research data. A confidence interval is an inference to a population in terms of an estimation of sampling error. More specifically, it provides a range of values that fall within the population with a level of confidence of 100*(1 - α) % (i.e. the level of confidence is 95% for α =0.05). Confidence intervals (CIs) offer much more information and allow us to move beyond dichotomous thinking and adopt an "*estimation thinking"*. Estimation thinking focuses on how big an effect is; this is usually more valuable than knowing whether or not the effect is zero, which is the focus of dichotomous thinking. Confidence intervals convey information about magnitude and precision of effect simultaneously, keeping these two aspects of measurement closely linked [49]–[51].

This study used the Wilcoxon statistical test which is a non-parametric procedure used to test whether there is sufficient evidence that the median of two probability distributions differ in location [52]. All statistical tests were two-sided and performed at α=0.05 significance level. Confidence intervals were calculated using Hodges-Lehmann estimates of shift [53], [54]. To adjust for multiple testing, we used the Holm-Bonferroni method [55].

3) Effect size results

To verify how meaningful is the improvement and how important are the results, the effect size criterion defined by Equation (5) was used:

$$\mathbf{\Delta} = \frac{MAR - \overline{\mathrm{MAR}_{\mathrm{P}_0}}}{SP_0} \quad (5)$$

where SP_0 is the sample standard deviation of the random guessing. The Δ values can be interpreted in terms of the categories proposed by Cohen [56] of small (around 0.2), medium (around 0.5) and large (around 0.8). A medium or large value of Δ indicates an acceptable degree of confidence on the model predictions over random guessing.

D. Step 4: EBA accuracy evaluation using Borda Count

The use of SA enabled us to explore the influence of the missingness mechanisms and MD percentages and techniques on the prediction ability of the EBA technique. In fact, SA determines if EBA is reasonable (i.e., is actually predicting, and how much better is it than random guessing), but does not evaluate its accuracy [16]. Thus, SA alone is not sufficient to conclude about EBA accuracy and should be used with other metrics [15], [16].

Hence, we evaluated The EBA technique using a set of reliable accuracy measures that are believed to be less sensitive to bias and asymmetry. These measures are: Pred(0.25), Mean Absolute Error (MAE), Mean Balanced Relative error (MBRE), Mean Inverted Balanced Relative Error (MIBRE) and logarithmic standard deviation (LSD) as shown in Equations (8)–(12), respectively. Using a set of accuracy measures would ensure that different aspects are captured and would give more confidence in the results obtained compared with using only one accuracy measure. Similar approach was used in [57]–[59].

We evaluated the EBA variants (i.e. with four MD techniques and three missingness mechanisms) according to those performance measures and used Borda counting method to rank them over the four datasets in order to identify which variants were the most accurate. The Borda count method was used for the first time in SDEE by Azzeh et al. [59] and then by Idri et al. [58] and it allows to take into consideration different aspects of prediction performance since it is based on five performance measures.

$$AE_i = |e_i - \hat{e}_i| \quad (6)$$

$$MRE = \frac{AE_i}{e_i} \quad (7)$$

$$Pred(0.25) = \frac{100}{N}\sum_{i=1}^{N} \begin{cases} 1 \; if \; MRE_i \leq 0.25 \\ 0 \; otherwise \end{cases} \quad (8)$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n} AE_i \quad (9)$$

$$MBRE = \frac{1}{N}\sum_{i=1}^{N} \frac{AR_i}{\min(e_i, \hat{e}_i)} \quad (10)$$

$$MIBRE = \frac{1}{N}\sum_{i=1}^{N} \frac{AR_i}{\max(e_i, \hat{e}_i)} \quad (11)$$

$$LSD = \sqrt{\frac{\sum_{i=1}^{n}(\lambda_i + \frac{s^2}{2})^2}{n-1}} \quad (12)$$

where:

- e_i and $\hat{e}_i$ are the actual and predicted effort for the *i*th project.
- $\lambda_i = \ln(e_i) - \ln(\hat{e}_i)$
- s^2 is an estimator of the variance of the residual λ_i.

V. RESULTS

This section presents and discusses the experimental results when evaluating the accuracy of EBA using MD.

A. EBA Evaluation using SA (RQs 1-3)

The first objective is to investigate the effect of the MD techniques on the accuracy of EBA in terms of SA. Figures 3a-c show the median SA values for EBA applied to the four data sets with three mechanisms of missingness, different MD percentages and four MD techniques. In general, we notice that the SA values decrease as the MD percentage increases regardless the mechanism of missingness.

For the MCAR mechanism, we observe that the imputation techniques outperformed toleration and deletion (SA values at 10% of MD: 41% for SVRI, 41% for KNNI, 37% for toleration and 34% for deletion). Moreover, Fig.3-a shows that toleration outperformed deletion (SA values at 10% of MD: 37% for toleration and 34% for deletion). Note that SVR and KNN imputations performed almost the same.

For MAR mechanism, Fig.3-b shows that the imputation techniques have the same performance (SA values at 10% of MD: 35% for SVRI, 34% for KNNI). Moreover, they both outperformed toleration and deletion (SA values at 10% of MD: 31% for Toleration, 31% for Deletion). Moreover, we notice that toleration and deletion gave the same performance.

As for NIM mechanism, we notice that the imputation techniques gave similar performances (SA values at 10% of MD: 32% for SVRI and 32% for KNNI). Moreover, toleration and deletion performed the same (SA values at 10% of MD: 26% for toleration and 27% for deletion).

1) Hypothesis testing

In the previous section, we found that imputation techniques (KNN or SVR) instead of toleration and deletion improved the performance of EBA. This section investigates whether this improvement is statistically significant. Moreover, we investigate whether the improvement varies with the mechanisms of missingness. To do that, we compared the median of SA values across data sets for each MD percentage using the Wilcoxon t-test. We drew the following hypothesis:

NH1: The prediction performance of EBA is not affected by the MD technique.

NH2: The prediction performance of EBA when using MD techniques is not affected by the mechanism of missingness.

Each null hypothesis was evaluated separately for the MD techniques and the three missingness mechanisms. Tables II and III sum up the results of the Wilcoxon t-test conducted to evaluate NH1 and NH2 respectively, where p(α) denotes the p-value of the Wilcoxon test, α' denotes the significance level corrected by Holm-Bonferroni correction and CI denotes the confidence interval. Table II shows the results of Wilcoxon test on NH1. We notice that the difference between SVR and KNN imputations is not significant regardless of the mechanism of missingness. The confidence intervals also reflect this finding. We notice that for MCAR and MAR mechanisms, the confidence intervals have negative values. This means that no imputation technique is always superior to the other. For MCAR mechanism, we observe that the difference between KNN and SVR is between -1.679 and 0.416. This means that KNN outperformed SVR by 1.679 and SVR outperformed KNN by 0.416. The case of MAR is the similar to MAR. However, under NIM, we observe that SVR outperformed KNN and the magnitude of the difference is between 0.484 and 0.7

Table II also shows that imputation techniques often significantly outperformed toleration and deletion. We notice that the magnitude of the difference is larger with deletion compared to toleration. The improvement given by the imputation techniques over toleration/deletion is larger when using MCAR or NIM compared to MAR. This is due to the fact that, under MCAR mechanism, imputation techniques outperformed largely toleration and deletion. Moreover, under NIM mechanism, toleration and deletion gave the worst results (negative values of SA). Under MAR mechanism, the performance of imputation techniques decreased and the toleration/deletion gave acceptable performance which explains the small CI.

To evaluate the impact of the missingness mechanisms on the accuracy of EBA, Table III reports the results of the statistical tests of NH2; it can be noticed that:

- The difference between MCAR and MAR is significant when using SVR, KNN and toleration. However, the difference is not significance when using deletion.
- The difference between MCAR and NIM is in general significant.
- The difference between MAR and NIM is significant for SVR, toleration and deletion. However, it is not significant in the case of KNN.

2) Effect size results

To ensure that the results are not generated by chance and to assess if there is effect improvement over random guessing, we evaluate the effect size measured by means of Equation (5). Table VII reports the median values of the effect size Δ of EBA using the four MD techniques under three mechanisms of missingness and nine MD percentages across the four datasets where the baseline method is random guessing.

From Table VII, we notice that the Δ values are higher than 0.5, which means that the results obtained by EBA in terms of SA are more likely not to be due to chance under:

- MCAR mechanism when using: 1) imputation, 2) toleration/deletion with MD percentage less than 80%.
- MAR mechanism when using: 1) KNN imputation with MD percentage less than 90%, 2) SVR imputation, toleration or deletion with MD percentage less than 80%.
- NIM mechanism when using: 1) SVR/KNN imputation with MD percentage less than 90%, 2) toleration with MD percentage less than 60%, 3) deletion with MD percentage less than 70%.

B. EBA Evaluation using Borda Count (RQ4)

Although SA results would confirm if EBA outperform random guessing, they are not sufficient to conclude about the accuracy of EBA [15], [16]. This section evaluates EBA with four MD techniques and three missingness mechanisms using a set of reliable performance measures as explained in Section V.C. Thereafter, we rank the four variants of EBA (i.e. with the four MD techniques) using the Borda count method. Table VI shows the ranking over the four datasets.

We notice that EBA with SVR imputation generally outperformed the other EBA variants except for USP05FT under MAR and USP05RQ under NIM.

Moreover, we notice that toleration generally improves the EBA accuracy compared to deletion except for ISBSG under MAR/NIM and USP05FT under NIM.

Furthermore, we compared the four EBA variants across the four datasets. Table VII shows the results of this ranking. We notice that EBA with SVR imputation was the best, followed by KNN imputation, toleration and lastly deletion regardless the missingness mechanisms.

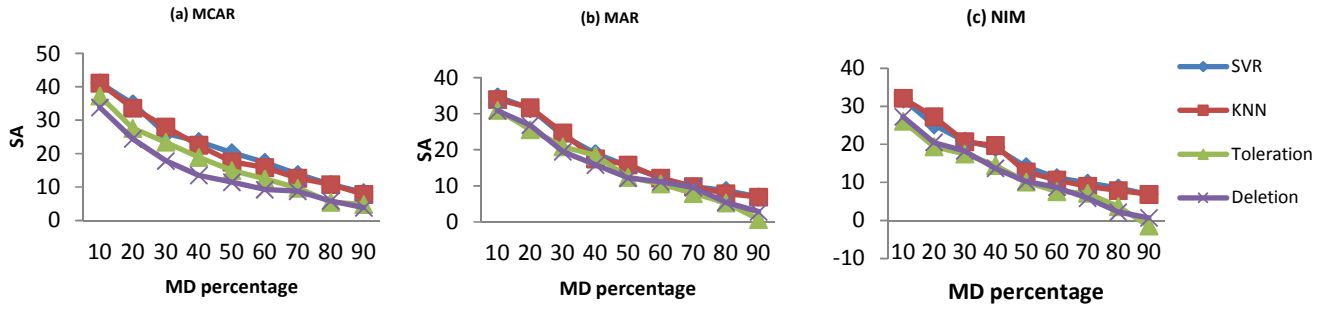

Figure 3(a-c) SA values of Classical Analogy applied to four data sets with three mechanisms of missingness, different MD percentages and four MD techniques

TABLE III Results of Wilcoxon test of NH2

		KNN				Toleration				Deletion			
		p(α)/ α'	Z	CI		p(α)/ α'	Z	CI		p(α)/ α'	z	CI	
				Min	Max			Min	Max			Min	Max
MCAR	SVR	0.123/0.05	1.540	-1.679	0.416	0.008/0.0167	2.666	3.688	5.783	0.008/0.01	2.666	5.097	9.408
	KNN					0.008/0.0125	2.666	2.996	4.911	0.008/0.0083	2.666	4.879	9.060
	Toleration									0.011/0.025	2.547	1.012	4.491
MAR	SVR	0.499/0/025	0.676	-0.766	0.362	0.008/0.0125	2.666	1.842	4.759	0.008/0.01	2.666	1.780	4.217
	KNN					0.011/0.0167	2 .547	1.264	4.814	0.008/0.0083	2.666	1.387	4.438
	Toleration									0.767/0.05	0.296	-1.299	1.230
NIM	SVR	0.790/0.05	0.757	0.484	0.700	0.008/0.0167	2.666	3.530	6.160	0.008/0.01	2.666	3.262	5.508
	KNN					0.008/0.0125	2.666	2.836	6.782	0.008/0.083	2.666	2.509	6.069
	Toleration									0.515/0.025	0.652	-0.732	1.206

TABLE IV Results of Wilcoxon test of NH1

		MAR				NIM			
		p(α)/ α'	Z	CI		p(α)/ α'	Z	CI	
				Min	Max			Min	Max
SVR	MCAR	0.008/0.0167	2.666	2.126	5.292	0.008/0.025	2.666	3.088	7.783
	MAR					0.038/0.05	2.073	0.004	3.222
KNN	MCAR	0.008/0.0125	2.666	1.909	5.071	0.008/0.025	2.666	2.855	6.884
	MAR					0.086/0.05	1.718	-0.200	3.108
Toleration	MCAR	0.008/0.05	2.666	1.121	4.135	0.008/0.025	2.666	3.350	8.043
	MAR					0.008/0.0167	2.666	1.828	4.558
Deletion	MCAR	0.314/0.05	1.007	-1.050	2.001	0.021/0.025	2.310	0.381	3.919
	MAR					0.008/0.0167	2.666	1.900	4.194

TABLE V Median values of effect size of EBA across the four datasets based on comparison with random guessing baseline method

	MCAR				MAR				NIM			
	SVRI	KNNI	Toleration	Deletion	SVRI	KNNI	Toleration	Deletion	SVRI	KNNI	Toleration	Deletion
10%	-2.34	-2.43	-2.1	-2.06	-2.28	-2.49	-1.98	-1.93	-2.09	-2.4	-1.76	-1.76
20%	-2.15	-1.89	-1.82	-1.53	-1.95	-2.07	-1.49	-1.68	-1.84	-1.72	-1.49	-1.49
30%	-2	-1.58	-1.53	-1.27	-1.74	-1.6	-1.27	-1.32	-1.48	-1.33	-1.19	-1.28
40%	-1.62	-1.27	-1.27	-0.91	-1.45	-1.51	-1.08	-1.18	-1.19	-1.28	-0.82	-1.01
50%	-1.29	-0.99	-1.04	-0.82	-1.04	-1.16	-0.81	-0.94	-1.06	-0.96	-0.63	-0.76
60%	-1.08	-0.89	-0.9	-0.67	-0.85	-0.94	-0.71	-0.83	-0.82	-0.75	-0.48	-0.63
70%	-0.83	-0.8	-0.65	-0.63	-0.69	-0.67	-0.55	-0.7	-0.71	-0.64	-0.46	-0.43
80%	-0.64	-0.71	-0.4	-0.43	-0.45	-0.63	-0.4	-0.4	-0.59	-0.56	-0.2	-0.16
90%	-0.56	-0.55	-0.28	-0.26	-0.35	-0.47	-0.06	-0.18	-0.4	-0.47	-0.09	-0.05

TABLE VI Borda Count ranking of the four MD techniques under the three missingness mechanisms.

	EBA											
	MCAR				MAR				NIM			
	SVRI	KNNI	Toleration	Deletion	SVRI	KNNI	Toleration	Deletion	SVRI	KNNI	Toleration	Deletion
ISBSG	1	3	2	4	1	3	4	2	1	2	4	3
COCOMO81	1	2	3	4	1	3	2	4	1	2	3	4
USP05FT	1	2	3	4	2	1	3	4	1	2	4	3
USP05RQ	1	2	3	4	1	3	2	4	2	1	3	4

TABLE VII Borda Count ranking of the four MD techniques under the three missingness mechanisms across data sets.

	SVRI	KNNI	Toleration	Deletion
MCAR	1	2	3	4
MAR	1	2	3	4
NIM	1	2	3	4

VI. Conclusion and Future Work

This study evaluated EBA using four MD techniques: toleration, deletion, KNN imputation, and SVR imputation with different percentages (from 10% to 90%) and three missingness mechanisms (MCAR, MAR and NIM) on four datasets (ISBSG R8, COCOMO81, USP05_FT and USP05_RQ) with mixed data (numerical and categorical).

four research questions RQs 1-4 have been discussed. The findings when answering RQs 1-4 are as follows:

(RQ1): Is there evidence that the use of KNN and SVR imputations rather than toleration/deletion improves the accuracy of EBA in terms of SA when using mixed datasets?

We found that EBA with imputation techniques achieved significantly better SA values over EBA with toleration or deletion regardless of the mechanism of missingness.

EBA with toleration provided significantly better SA over EBA with deletion when the missingness mechanism is MCAR. However, under MAR or NIM mechanisms, the improvement provided by toleration over deletion is not significant. Moreover, EBA outperformed random guessing when using imputation techniques. However, when using toleration/deletion at high percentages of MD, EBA underperformed random guessing.

(RQ2): Is there evidence that SVR imputation instead of KNN imputation would improve EBA accuracy measured in terms of SA?

In terms of SA, we found that the performance difference between the imputation techniques KNN and SVR was not significant.

(RQ3): Is there evidence that the missingness mechanism and the MD percentage affect the accuracy of EBA measured in terms of SA over mixed datasets?

We found that EBA with MCAR instead of MAR achieved significantly better SA values when using imputation or toleration. However, when using deletion, the improvement of EBA with MCAR instead MAR is not significant. EBA with MCAR instead of NIM achieved significantly better SA values regardless of the MD technique used. EBA with MAR instead of NIM achieved significantly better SA values when using toleration/deletion. However, when using imputation, the difference is not significant.

(RQ4): Does the performance of EBA in terms of Pred(0.25), MAE, MBRE, MIBRE and LSD confirm the findings of SA?

When using the Borda count based on five accuracy criteria, we found that EBA with SVR was the best, followed by KNN imputation. We also notice that toleration instead of deletion improves the accuracy of EBA. These findings confirm those of [14], [15] stating that different measures capture different aspects of EBA performance: SA determines if a technique is reasonable (i.e., is actually predicting, and how much better is it than random guessing) while other accuracy metrics measure how close a prediction is to its correct value.

Future work aims to confirm our finding with different variants of EBA techniques (e.g. Fuzzy Analogy) and other software effort estimation techniques. Moreover, other imputation techniques may give different results.

References

[1] S. K. Sehra, Y. S. Brar, N. Kaur, and S. S. Sehra, "Research patterns and trends in software effort estimation," *Inf. Softw. Technol.*, vol. 91, p. , 2017.

[2] J. Wen, S. Li, Z. Lin, Y. Hu, and C. Huang, "Systematic literature review of machine learning based software development effort estimation models," *Inf. Softw. Technol.*, vol. 54, no. 1, pp. 41–59, 2012.

[3] A. Idri, F. A. Amazal, and A. Abran, "Analogy-based software development effort estimation: A systematic mapping and review," *Inf. Softw. Technol.*, vol. 58, pp. 206–230, 2014.

[4] F. A. Amazal, A. Idri, and A. Abran, "Improving Fuzzy Analogy based Software Development Effort Estimation," in *21st Asia-Pacific Software Engineering Conference (APSEC)*, 2014, pp. 1–4.

[5] F. A. Amazal, A. Idri, and A. Abran, "An analogy-based approach to estimation of software development effort using categorical data," in *Joint Conference of the International Workshop on Software Measurement and the International Conference on Software Process* and Product Measurement, 2014, pp. 252–262.

[6] A. Idri and A. Abran, "Evaluating software project similarity by using linguistic quantifier guided aggregations," *Proceedings Joint 9th IFSA World Congress and 20th NAFIPS International Conference,* Volume: 1, 2001, pp. 470 - 475.

[7] A. Idri, F. A. Amazal, and A. Abran, "Accuracy Comparison of Analogy-Based Software Development Effort Estimation Techniques," *Int. J. Intell. Syst.*, vol. 31 (2), pp. 128–152, 2016.

[8] A. Idri, I. Abnane, and A. Abran, "Missing data techniques in analogy-based software development effort estimation," *J. Syst. Softw.*, vol. 117, pp. 595–611, 2016.

[9] J. Li, G. Ruhe, A. Al-Emran, and M. M. Richter, "A flexible method for software effort estimation by analogy," *Empir. Softw. Eng.*, vol. 12, no. 1, pp. 65–106, 2007.

[10] M. Shepperd and C. Schofield, "Estimating Software Project Effort Using Analogies," *IEEE Trans. Softw. Eng.*, vol. 23, no. 12, pp. 736–743, 1997.

[11] A. Idri and I. Abnane, "Fuzzy Analogy Based Effort Estimation: An Empirical Comparative Study," in *IEEE International Conference on Computer and Information Technology (CIT)*, 2017, pp. 114–121.

[12] F. A. Amazal, A. Idri, and A. Abran, "Software Development Effort Estimation Using Classical and Fuzzy Analogy: a Cross-Validation Comparative Study," *Int. J. Comput. Intell. Appl.*, vol. 13, no. 3, p.

1450013, 2014.
[13] J. Li, A. Al-Emran, and G. Ruhe, “Impact Analysis of Missing Values on the Prediction Accuracy of Analogy-based Software Effort Estimation Method AQUA,” First Int. Symp. Empir. Softw. Eng. Meas. (ESEM 2007), pp. 126–135, 2007.
[14] I. Abnane and A. Idri, “Evaluating Fuzzy Analogy on Incomplete Software Projects data,” in IEEE Symposium Series on Computational Intelligence (SSCI), 2016.
[15] M. Azzeh and A. B. Nassif, “A hybrid model for estimating software project effort from Use Case Points,” Appl. Soft Comput. J., pp. 1–9, 2016.
[16] A. Idri, I. Abnane, and A. Abran, “Evaluating Pred(p) and standardized accuracy criteria in software development effort estimation,” J. Softw. Evol. Process, no. September, 2017.
[17] R. J. A. Little and D. . Rubin, “Statistical Analysis with Missing Data,” Wiley, New York., 1987.
[18] D. . Little, R.J.A., Rubin, “Analysis of social science data with missing values,” Sociol. Methods Res., pp. 292–326, 1989.
[19] G. Molenberghs and M. G. Kenward, Missing Data in Clinical Studies, vol. 61. John Wiley & Sons, 2007.
[20] 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,” J. Syst. Softw., vol. 81, no. 12, pp. 2361–2370, 2008.
[21] A. Idri, I. Abnane, and A. Abran, “Systematic Mapping Study of Missing Values Techniques in Software Engineering Data,” in International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), 2015 16th IEEE/ACIS, 2015, pp. 1–8.
[22] J. Schafer, Analysis of Incomplete Multivariate Data. 1997.
[23] B. W. Boehm, “Software Engineering Economics,” IEEE Trans. Softw. Eng., vol. SE-10, no. 1, 1984.
[24] L. C. Briand, K. El Emam, D. Surmann, I. Wieczorek, and K. D. Maxwell, “An assessment and comparison of common software cost estimation modeling techniques,” Proc. 21st Int. Conf. Softw. Eng. - ICSE ’99, pp. 313–322, 1999.
[25] E. Mendes, “A Comparative Study of Cost Estimation Models for Web Hypermedia Applications,” Empir. Softw. Eng., vol. 8, no. 2, pp. 163–196, 2003.
[26] M. Azzeh and Y. Elsheikg, “Learning Best K analogies from Data Distribution for Case-Based Software Effort Estimation,” in The Seventh International Conference on Software Engineering Advances, 2012, no. 2, pp. 341–347.
[27] L. Angelis and I. Stamelos, “A Simulation Tool for Efficient Analogy Based Cost Estimation,” Empir. Softw. Eng., vol. 5, no. 1, pp. 35–68, 2000.
[28] G. K. Michelle, M. Cartwright, and L. Chen, “Experiences Using Case-Based Reasoning to Predict Software Project Effort,” no. Ml, pp. 1–22, 2000.
[29] A. Idri, A. Abran, and T. M. Khoshgoftaar, “Investigating soft computing in case-based reasoning for software cost estimation,” International Journal of Engineering Intelligent Systems for Electrical Engineering and Communications, 10 (3), 2002. p. 147-157.
[30] A. Idri, A. Zahi, and E. Mendes, A. Zakrani, “Software Cost Estimation Models Using Radial Basis Function Neural Networks”, Mensura-IWSM: Software Process and Product Measurement , 2007, pp 21-31.
[31] S. Yenduri, “an Empirical Study of Imputation Techniques for Software Data Sets,” Louisiana State, 2005.
[32] V. N. Vapnik, The Nature ofStatistical Learning Theory. New York, 1995.
[33] S. K. Shevade, S. S. Keerthi, C. Bhattacharyya, and K. R. K. Murthy, “Improvements to the SMO algorithm for SVM regression,” IEEE Trans. Neural Networks, vol. 11, no. 5, pp. 1188–1193, 2000.
[34] J. C. Platt, “Sequential minimal optimization: A fast algorithm for training support vector machines,” Adv. Kernel MethodsSupport Vector Learn., vol. 208, pp. 1–21, 1998.
[35] A. Smola and B. Scholkopf, “A tutorial on support vector regression,” Stat. Comput., vol. 14, no. 3, pp. 199–222, 2004.
[36] X. Chen, Q. Zhou, and H. Xiao, “Combination of Support Vector Regression with Particle Swarm Optimization for Hot-spot temperature prediction of oil-immersed power transformer,” Prz. Elektrotechniczny, no. 8, pp. 172–176, 2012.
[37] A. L. I. Oliveira, “Estimation of software project effort with support vector regression,” Neurocomputing, vol. 69, no. 13–15, pp. 1749–1753, 2006.
[38] S. S. Keerthi, S. K. Shevade, C. Bhattacharyya, and K. R. K. Murthy, “Improvements to platt’s SMO algorithm for SVM classifier design,” 1999.
[39] V. N. Vapnik, “An overview of statistical learning theory.,” IEEE Trans. Neural Netw., vol. 10, no. 5, pp. 988–99, 1999.
[40] E. Osuna, R. Freund, and F. Girosi, “Training support vector machines: an application to face detection,” in IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1997, pp. 130–136.
[41] H. Hsieh, T. Lee, and T.-S. Lee, “A Hybrid Particle Swarm Optimization and Support Vector Regression Model for Financial Time Series Forecasting,” Int. J. Bus. Adm., vol. 2, no. 2, pp. 48–56, 2011.
[42] C. W. Hsu, C. . Chang, and C. J. A. Lin, “A practical guide to support vector classification.,” 2003.
[43] Q. Zong, W. Liu, and L. Dou, “Parameters selection for SVR based on PSO,” in 6th World Congress on Intelligent Control and Automation, 2006, no. 1, pp. 2811–2814.
[44] E. Kocaguneli and T. Menzies, “Software effort models should be assessed via leave-one-out validation,” J. Syst. Softw., vol. 86, no. 7, pp. 1879–1890, 2013.
[45] M. Shepperd and S. MacDonell, “Evaluating prediction systems in software project estimation,” Inf. Softw. Technol., vol. 54, no. 8, pp. 820–827, 2012.
[46] J. Demšar, “Statistical Comparisons of Classifiers over Multiple Data Sets,” J. Mach. Learn. Res., vol. 7, pp. 1–30, 2006.
[47] G. M. Foody, “Classification accuracy comparison: Hypothesis tests and the use of confidence intervals in evaluations of difference, equivalence and non-inferiority,” Remote Sens. Environ., vol. 113, no 8, pp. 1658–1663, 2009.
[48] S. Greenland et al., “Statistical tests, P values, confidence intervals, and power: a guide to misinterpretations,” Eur. J. Epidemiol., vol. 31, no. 4, pp. 337–350, 2016.
[49] C. J. Geyer, “Nonparametric Tests and Confidence Intervals,” In Pract., pp. 1–14, 2003.
[50] G. Cumming and S. Finch, “Inference by eye: Confidence intervals and how to read pictures of data.,” Am. Psychol., vol. 60, no. 2, pp. 170–180, 2005.
[51] M. J. Gardiner and D. G. Altman, Statistics with confidence: confidence intervals and statistical guidelines. 1989.
[52] D. Sheskin, Handbook of Parametric and Non-parametric Procedures. CRC Press, 1997.
[53] E. Lehmann, “Nonparametrics: Statistical methods based on ranks,” Prentice Hall New Jersey, 1998.
[54] J. L. Hodges and E. L. Lehmann, “Estimates of Location Based on Rank Tests,” Ann. Math. Stat., 1963.
[55] H. Abdi, “1 Overview 2 Preliminary : The different meanings of alpha,” Encycl. Res. Des., pp. 1–8, 2010.
[56] J. Cohen, “Quantitative Methods in Psychology,” Psychol. Bull., vol. 112, no. 1, pp. 155–159, 1992.
[57] M. Hosni, A. Idri, A. Abran, and A. B. Nassif, “On the value of parameter tuning in heterogeneous ensembles effort estimation,” Soft Computing, Springer Berlin Heidelberg, pp. 1–34, 2017.
[58] A. Idri, M. Hosni, and A. Abran, “Improved Estimation of Software Development Effort Using Classical and Fuzzy Analogy Ensembles,” Appl. Soft Comput., vol. 49, pp. 990–1019, 2016.
[59] M. Azzeh, A. B. Nassif, and L. L. Minku, “An empirical evaluation of ensemble adjustment methods for analogy-based effort estimation,” J. Syst. Softw., vol. 103, pp. 36–52, 2015.