Indonesian Journal of Electrical Engineering and Computer Science
Vol. 27, No. 2, August 2022, pp. 1138~1150
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v27.i2.pp1138-1150

Cluster-based fuzzy regression trees for software cost prediction

Assia Najm[1], Abdelali Zakrani[2], Abdelaziz Marzak[1]
[1]Department of Mathematics and Computer Science, Faculty of Sciences Ben M'sik, Hassan II University, Casablanca, Morocco
[2]Department of Computer Science Engineering, Ecole Nationale Superieure d'Arts et Metiers, Casablanca, Morocco

Article Info

Article history:

Received Aug 30, 2021
Revised Jun 15, 2022
Accepted Jun 23, 2022

Keywords:

Decision tree
Fuzzy k-means
Regression tree
Software development cost prediction
Software development effort estimation

ABSTRACT

The current paper proposes a novel type of decision tree, which is never used for software development cost prediction (SDCP) purposes, the cluster-based fuzzy regression tree (CFRT). This model uses the fuzzy k-means (FKM), which deals with data uncertainty and imprecision. The tree expansion is based on the variability measure by choosing the node with the highest value of granulation diversity. This paper outlined an experimental study comparing CFRT with four SDCP methods, notably linear regression, multi-layer perceptron, K-nearest-neighbors, and classification and regression trees (CART), employing eight datasets and the leave-one-out cross-validation (LOOCV). The results show that CFRT is among the best, ranked first in 3 datasets according to four accuracy measures. Also, according to the Pred(25%) values, the proposed CFRT model outperformed all the twelve compared techniques in four datasets: Albrecht, constructive cost model (COCOMO), Desharnais, and The International Software Benchmarking Standards Group (ISBSG) using LOOCV and 30-fold cross-validation technique.

Corresponding Author:

Assia Najm
Department of Mathematics and Computer Science, Faculty of Sciences Ben M'sik, Hassan II University Casablanca, Morocco
Email: assia.najm@gmail.com

1. INTRODUCTION

A growing body of literature on software project management recognizes the importance of software development cost prediction (SDCP). Accurate models assist in efficiently controlling resources, timing, and budgeting even in the earlier phase of the software development process. However, many limitations or conditions could negatively affect the performance of SDCP models and therefore give inaccurate estimations. There are many agile and non-agile effort prediction techniques [1], where researchers propose new methods or optimization for improving existing ones. Generally, decision trees (DTs) are one of the computational intelligence techniques which are largely employed for cost prediction [2], [3].

A DT consists of one root node, different leaves, and branch nodes, where each branch constitutes an option among many options. The effort estimation for a test project is done by traversing the DT nodes according to the project's attributes values. Once the project hits the leaf node, the estimation is performed based on the instances located at this leaf node [4]. Effort estimation using DT has many significant advantages. First, DTs can be employed for classification and regression tasks (RTs) [3]. Second, DTs are known to be easily comprehensible thanks to their ability to provide a meaningful bit of knowledge about the function(f) that models the inherent hidden relation between features and effort. Third, DTs can also be used for feature selection to reduce misleading data in the SDCP models [5]. Despite the previously mentioned advantages, classical DTs methods suffer from a particular weakness that hinders their ability to make robust estimations. First, DTs are very sensitive to the data used in model construction because numerical features with sharp boundaries are susceptible to their values. Therefore any change in input values impacts the decision value [6]. Second, crisp DTs cannot handle uncertainty and imprecision in the acquired data and the obtained estimates. Third, DT suffers from the absence of locality due to the crisp boolean nature of tree splits. Two close projects are treated separately by opposite sides of tree branches and hence have distinct effort values [7].

Subsequently, many studies were performed to avoid the limitations mentioned. Consequently, several techniques were employed with DT methods to surmount the traditional DTs challenges. Mainly, according to Najm *et al.* [8], the fuzzy-logic (FL) is the most employed technique with DT, and the resulting trees are then called fuzzy decision trees (FDTs).

The FDTs include the fuzzy set theory that enables partial membership into the tree's nodes. The FDTs differ from the classical DTs in the fuzzy restriction of splitting criteria and the inference approach [9]. The fuzzy DT approach has several attractive features. First, the knowledge representation is enhanced thanks to fuzzy splits. Second, fuzzification introduces a continuity constraint at the split boundaries that maintain the locality notion where two close projects pursue equal treatment and gain similar effort estimates. Third, FDT perfectly solves the issues related to species preciseness arising from uncertain data or generated by imprecise models. Overall, FDTs have the potential to simulate human reasoning thanks to their ability to perform decisions with high imprecise and volatile datasets besides solving the uncertainty derived from sharp interval boundaries. However, FDTs suffer from some serious issues. First, there are challenges in defining proper fuzzy set partitions and their suitable membership function (MF) [10]. Because the fuzzification closely depends on the dataset used. Therefore, a suitable fuzzy set and MF will produce reasonable effort estimations, while a worse choice will yield inaccurate model estimates. Second, tree growth based on a single attribute selection is also perceived as a conceptual restriction [11], affirmed in [12].

This paper proposes a novel type of regression tree that handles these shortcomings, stated as cluster-based fuzzy regression trees (CFRT) [11]. Essentially, CFRT is conceived to handle the uncertainty management issues and assumed to address the classical FDTs challenges. The essential advantages of using CFRT are:

- Decreasing the imprecision in the prediction model by involving the fuzzy k-means (FKM) during the tree growth of CFRT. In our proposed CFRT model, a project can belong to many nodes at each tree level, which increases the probability of any project being captured by the correct node at each level of the tree. It is also beneficial in handling the uncertainty's sources, given that the CFRT introduces the FM of projects in tree nodes. Plus, it produces imprecise estimates that offer practitioners the possibility of clearly quantifying the uncertainties in the estimates.
- Surmounting traditional FDT limitations. First classical FDT performs estimation based on a subset of pertinent attributes, which is considered a conceptual drawback because considering more than one attribute may result in good trees and enhanced accuracy [11] as the CFRT, where predictions are made based on the whole set. Second, CFRT does not suffer from the problem of finding a proper fuzzy set and suitable MF, which is done thanks to its particular structure based on the FKM method.
- Maintaining the quality of the historical projects because the generated estimates are directly fitted with the project attribute. The generated estimate of the fuzzy inference is in the form of a single numerical value. Which is more advantageous in enhancing the accuracy of the estimates by preventing the defuzzification step that transforms the fuzzy estimate into a crisp one.

For example, we illustrate the difference between CFRT and traditional FDT with the following example, where we perform a software effort estimation using Kemerer datasets and the leave-one-out cross-validation (LOOCV) method.

A decision tree using classification and regression trees (CART) Figure 1 forms a binary tree. Each node of a tree specifies a condition based on one of the features. Each branch corresponds to possible values or range of values this variable may take, where the decision represents the value of a predicted variable determined based on data in the particular terminal node.

While a decision tree using CFRT Figure 2 is based mainly on the FKM algorithm, which is used for clustering the dataset D into k distinct clusters (k=3 in this example) while each cluster is represented by its center Ci. The tree growth is performed by repetitively splitting the most heterogeneous node from all resulting nodes. In this example, one tree level exists due to the minimum size of the Kemerer dataset and the k=3. Each node of a tree contains more than one attribute. The prediction of a project pi is made as follows: At each tree level, we determine its membership to each node based on each node center ci. After that, we choose the correspondent pathway until the terminal nodes (leaves) are reached (see subsection 2.1.3 for more details about calculating the estimated effort value). By using CART algorithm as shown in Figure 1 we get the following values of errors: MAE=140.5, Pred (0.25)=26.66, MMRE=0.79, MdMRE=0.57, SA=0.37, delta=1.49, MIBRE=0.46, MBRE=1.21. By using CFRT algorithm as shown in Figure 2 we get the following values of errors: MAE=105.88, Pred (0.25)=46.66, MMRE=0.56, MdMRE= 0.27, SA=0.53, delta=1.99, MIBRE=0.32, MBRE=0.82.

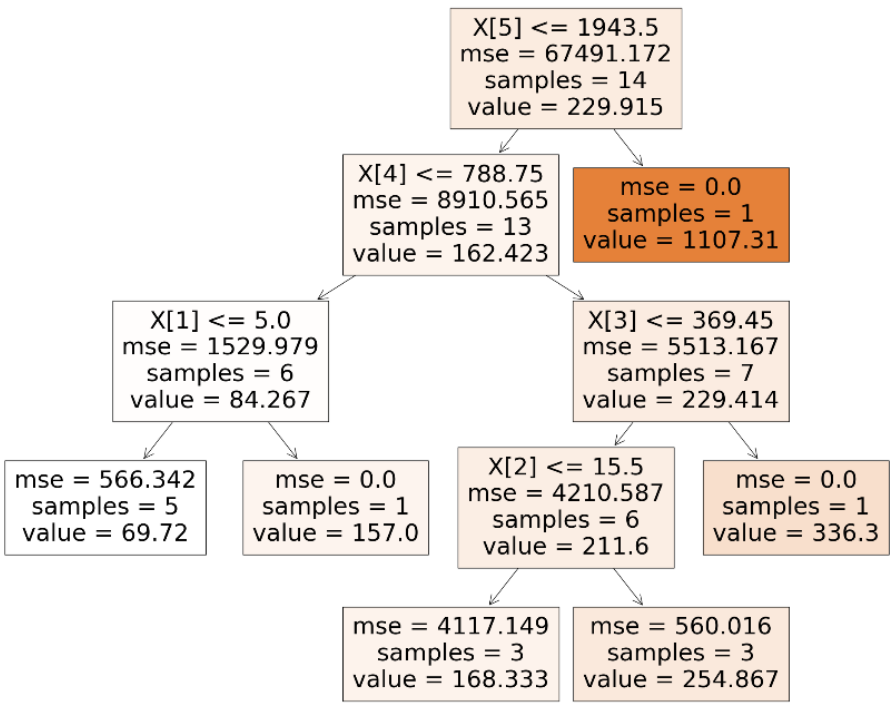
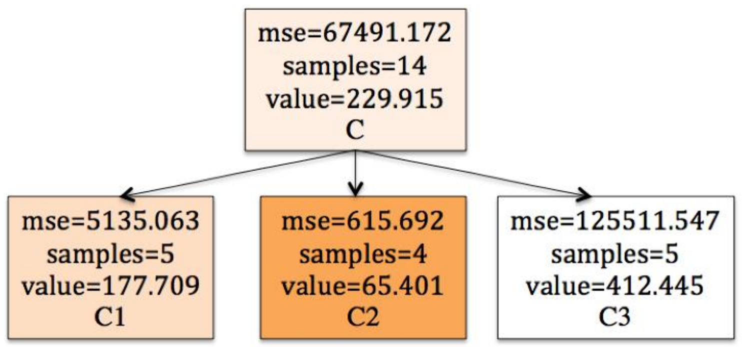


Figure 1. The expansion of the tree using the CART algorithm



C, C1, C2, and C3: Clusters' centers
C: [1.07142857, 2.42857143, 14.28571429, 195.6, 995.91428571, 995.14285714]
C1: [1.00002916, 2.86804511, 19.42883188, 192.43131932, 909.98489297, 880.03159406]
C2: [1.24718259, 2.78542626, 5.68655844, 44.85964467, 322.97302454, 309.92936167]
C3: [1.00000053, 1.60833782, 15.6254328, 327.961146, 1648.00395, 1688.41186]

Figure 2. The expansion of the tree nodes using CFRT with k=3

We notice that CFRT produces an enhanced accuracy compared to the CART algorithm thanks to its particular structure based on more than one attribute, which is considered a conceptual advantage that results in good trees and then an enhanced accuracy (Pred=46.66). Nevertheless, as seen in Figure 1, CART performs estimation based on a choice of a single attribute, which is considered a conceptual drawback and yet reduces accuracy (Pred=26.66). Therefore, this paper proposed the CFRT model, which is more advantageous in managing uncertainty issues and surmounting the FDT challenges. The performance of the CFRT technique is then compared to four popular ML techniques: linear regression, multilayer perceptron (MLP), KNN, and CART. According to Wen *et al.* [13], KNN, MLP, and DT are the most employed methods with 37, 26, and 17%. Also, we note that tuning the hyperparameters of the SDCP methods can significantly affect their performance. We used the grid search (GS) method [14].

Hence, we perform the analysis in light of the following research questions:

a) ReQu1: Does the CFRT technique perform better than the four techniques according to the SA measure using LOOCV?
b) ReQu2: Does the CFRT present the most precise (accurate) technique compared to the four SDCP methods using LOOCV?
c) ReQu3: Does CFRT significantly outperform the other SDCP methods according to the statistical test?.

Regarding related works, Najm *et al.* [8] have performed a systematic literature review on decision trees-based SDCP. They outlined 50 studies published from 1985 to 2019. We noted the main results:

– Generally, decision trees perform better than case-based reasoning (CBR), CHAID, MLP, and RBFN models. Moreover, traditional DTs are unable to manage uncertainties.
– FL and regression are the most employed techniques with DT. Moreover, according to Pred(25) and MMRE, the best technique is FL (92.56% MMRE median).

Several works have been done to manage the SDCP model uncertainties. Hence, the researchers explain the uncertainty's sources or propose solutions to manage imprecision in software effort prediction environments. Mainly, in [5], [15], [16], the authors investigated the use of the fuzzy ID3 decision tree in SDCP by incorporating the principle of fuzzy theory. Idri and Elyassami [15], the experiments show that based on Pred(25) and MMRE, the Fuzzy ID3 outperforms crisp ID3, CART, and C4.5 over the TUKUTUKU dataset. Moreover, in Elyassami [16], the use of the ISBSG dataset enhances the performance. Also, in Elyassami and Idri [17] the authors are interested in testing the accuracy of fuzzy C5 and the effect of the confidence factor of pruning on the accuracy of fuzzy C5 using the ISBSG dataset [18]. The comparison was made utilizing fuzzy ID3, and the results show that increasing the confidence factor of pruning produces the best estimates of fuzzy C5.

Additionally, other studies focus on enhancing the accuracy of FDT by incorporating a hybrid DT model with FL. Which is the case for [19]-[21]. The authors in [20] suggested a fuzzy scheme (framework) that integrates FL with CART and CHAID. Besides, a fuzzy inference system (FIS) for defuzzification of generated rules. The suggested framework was applied to Desharnais, constructive cost model (COCOMO), and ISBSG datasets and the subsets available in the first stage of software development. The results revealed that the framework gives accurate results and can be used even in the first stages of software development.

By carefully analyzing the studies on FDTs, we affirmed that they deal with uncertainty management issues, a severe challenge of SCDP models. Also, they are enhanced compared to the crisp version. Nevertheless, the proposed models are still suffering from some weaknesses. For example, in [16], the author does not compare with other SDCP methods, which hinders the validity of the results. Also, in [5], the authors compare the proposed models just with the crisp version of the models or with just one model like in [17], [18], which is not enough for generalizing the results and affirming their superiority. Also, experimentation made in [15] is still limited due to using just one dataset and the absence of other Non-DT models in the comparison.

Some existing works use CFDT in classification and regression problems but not for software cost prediction concerns. For example, Chiu *et al.* [22] have proposed an enhanced CFDT method that uses an adjusted version of the FKM algorithm and the linear model in the leaf nodes where its parameters are determined using the recursive SVD-based least squares estimator. The empirical results indicate that the enhanced model generated a small (mean square) error compared to the initial model. Generally, the optimization calculation techniques of FKM are highly susceptible to the initially chosen parameters. Also, there is a risk of local extrema problems resulting from the optimization of cluster criteria. That is why some optimization algorithms exist in the literature, even inspired by nature, like particle swarm optimization [23], bee colony, and firefly method.

Alternatively, algorithms inspired by the biological domain, such as genetic algorithms [24], and artificial immune networks [25], were used to achieve global optimization of the Fuzzy k-means method. The use of GA with FKM in building decision trees (GCFDT) was previously employed for reactive ion etching (RIE) modeling as a data mining approach [26]. The main finding of this paper is that the accuracy of GCFDT outperforms that of classical CFDT and C4.5 algorithms. Also, Gadomer and Sosnowski [27] have proposed a fuzzy random forest with CFDT and proved that the proposed model performs better than the singular « CFDT » model and C4.5 rev. 8 classifiers. Therefore, in this paper, we proposed the CFRT model which is more advantageous in managing uncertainty issues and surmounting the FDT challenges.

The remainder of this study is organized as follows. Section 2 explains the method used, the overall design of the proposed "CFRT" model, and a detailed description of the used SDCP methods and the used optimization technique and datasets, while the results are explained in section 3. At last, section 4 concludes the study and provides information about future work.

2. METHOD

2.1. C-fuzzy regression trees: the tree growth and expansion

In this study, the construction of the tree blocks as shown in Figure 3 is based mainly on the FKM algorithm used for clustering the dataset D into k distinct clusters, while each cluster is represented by its center vi (or prototype). The variability criterion controls tree growth, which measures the data heterogeneity according to the target value y (effort) in a certain node (a cluster is perceived as a tree node). Let H $=\{H_1,....H_k\}$be a set of heterogeneity criteria of the corresponding k clusters. The node N_j with the highest H_k is chosen to be more refined (we refer the reader to [1] for more details about the calculation of H_i).

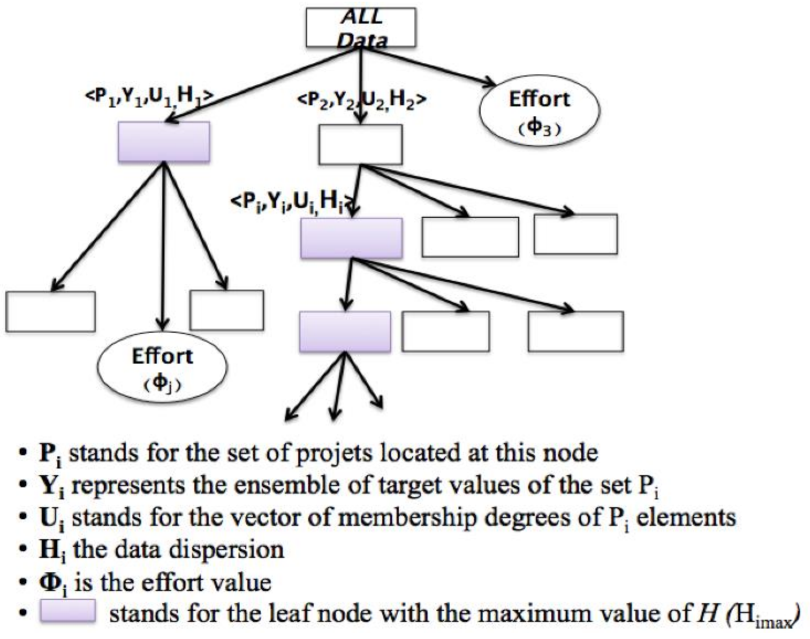

ALL Data
<P1,Y1,U1,H1>
<P2,Y2,U2,H2>
Effort (Φ3)
<Pi,Yi,Ui,Hi>
Effort (Φj)
• Pi stands for the set of projets located at this node
• Yi represents the ensemble of target values of the set Pi
• Ui stands for the vector of membership degrees of Pi elements
• Hi the data dispersion
• Φi is the effort value
• ▭ stands for the leaf node with the maximum value of H (Himax)

Figure 3. The expansion of the tree nodes with k=3

2.1.1. Clustering technique: fuzzy K-means method

Generally, the tree's growth is based on constructing fuzzy clusters through the fuzzy K-means method (FKM). The FKM is a widespread method of information granules that Bezdek described in [28]. It is a technique that helps assign each input in the dataset to more than one cluster. Essentially, the FKM is an optimization procedure where we perform a recursive updating of the membership matrix and cluster centers until specific stopping criteria are reached.

2.1.2. The criterion of node splitting

The overall tree growth is performed by measuring the data diversity accrued at the tree nodes. The split of tree nodes is made through the variability criterion, and we choose precisely the node in which the diversity criterion is maximum and split it into k subsequent nodes. We note that each dataset pattern is assigned to the clusters with distinct membership grades.

Besides, we mention two stopping criteria conditions that depend on the nature of node patterns. The first condition depends on the number of elements located at the node. We split the node if the number of its elements exceeds the clusters' number (k). The second condition depends on the data structure index. We refer the reader to [11] for more details about the splitting criteria and the quantization of this index. We stop expanding the node once the number of patterns (projects) is less than k (number of clusters) or the data structure ability is too weak.

2.1.3. Software cost prediction using the tree

Generally, we use the regression tree to predict the software development cost ($\hat{y}$) from a training data or software project (p_i). For this purpose, we calculate the membership degrees for each tree node (or cluster) φ_{ij}. We choose the correspondent pathway until the terminal nodes (leaves) are reached. At each tree level, we determine the path t_0 based on $t_0 = \mathrm{argmax}_k \varphi_k(\mathrm{pi})$. We repeat this procedure at each tree level. Finally, the estimated effort value ($\hat{y}$) calculation at the terminal node is equivalent to the ϕ_i value of the (1).

$$\phi_i = \frac{\sum_{(p_k, y_k) \in P \times Y_{ii}} \varphi_{ik} \times y_k}{\sum_{(p_k, y_k) \in P \times Y_{ii}} \varphi_{ik}} \quad (1)$$

2.2. SDCP techniques and optimization approach

We introduce in this subsection all SDCP methods used in comparison with the CFRT technique. Mainly the linear regression, multi-layer perceptron, k-nearest neighbor, and CART. Besides, the employed optimization technique.

2.2.1. Linear regression

LR is among the popular (famous) SDCP methods [2]. The LR estimates the effort required by building a linear equation. This linear equation models the relationship between the input variables (p_{j1}, …, p_{jn}) and the output (y_j).

2.2.2. Multi-layer perceptron

MLP was used efficiently to deal with many problems like time-series forecast, classification, and regression [29]. MLP is a neural network composed of an input layer, hidden layers (>=1), and an output layer that reflects the output variable (the cost). The performance of the MLP network depends on various parameters, which are: the number of hidden layers (H), their neurons, and the value of training epochs (E) plus the learning rate (L) and the momentum. These parameters were selected carefully in this study by performing a GS method.

2.2.3. K-nearest neighbor

KNN is one of the SDCP methods employed for classification and regression problems [30]. It is a kind of case-based reasoning, which is based on the following logic: identical inputs (projects) have a potentially identical output (effort/cost) [31]. Consequently, the cost prediction of a novel unseen project depends on the costs of analog historical projects. The KNN includes three phases: case recognition, similar projects' extraction, and adaption of cases. We note that in this paper, we firstly made a min-max normalization to the project features to have a similar impact when performing the prediction; also, we used the Euclidian distance to compute the analogy between the new project and the available projects in the dataset. And then, we average the costs of the N identical projects.

2.2.4. CART

CART [32] is an ML technique that uses the project attributes to construct a binary tree to deal with classification problems and regression issues. To get any prediction of any project, we must traverse the nodes of the constructed tree by starting from the root node and then choosing its subsequent nodes according to the project attributes' values. Then, the cost prediction is calculated from the values available in the final reached node (leaf).

2.2.5. Optimization technique: grid search

The performance of any ML method depends essentially on the chosen hyperparameters. We note that, for each technique, the selected hyperparameters change depending on the used dataset. Therefore, an optimization process is applied to get the optimal parameter settings that generate the best accuracy. In this study, we used the GS. The GS method uses the cross-validation technique to determine the most optimal values from a set of parameters by testing all possible combinations and selecting the best one. We refer the reader to [33] for more details about the GS principal phases.

2.3. Experimental design

This subsection gives insight into the practical design. Mainly, it briefly describes the employed datasets, the performance criteria, and the evaluation process description. In addition, the employed statistical test and the configuration of the SDCP methods.

2.3.1. Datasets

The characteristics of datasets significantly impact the evaluation of SDCP methods like the size of the dataset, the nature of its features, the presence of outliers, and missing values [34]. We note that the eight datasets used in this paper originated from two repositories: the repository PRedictOr Models In Software Engineering (PROMISE) [35] and the International Software Benchmarking Standards Group (ISBSG) repository [36]. Table 1 gives more details about the eight used datasets. The number of projects and features besides the statistics related to each dataset's target value (Effort).

Table 1. Datasets descriptive statistics

Dataset	Size	Nb. Features	Effort						
			Min	Max	Mean	Std	Median	Skewness	Kurtosis
Albrecht	24	7	0.50	105.20	21.88	28.42	11.45	2.30	4.67
Atkinson	16	14	109.00	912.00	456.06	241.07	427.00	0.47	-0.66
COCOMO	252	13	6.00	11,400.00	683.44	1,810.62	98.00	4.39	20.51
China	499	15	26.00	54,620.00	3,921.05	6,480.86	1,829.00	3.93	19.31
Desharnais	77	8	546.00	23,940.00	4,833.91	4,188.19	3,542.00	2.04	5.30
ISBSG	151	6	24.00	60,270.00	5,039.13	8,471.11	2,449.00	4.17	21.10
Kemerer	15	6	23.20	1,107.31	219.25	263.06	130.30	3.07	10.59
Miyazaki	48	7	5.60	1,586.00	87.48	228.76	38.10	6.26	41.36

2.3.2. Evaluation methods and criteria

We notice that, for all the eight datasets mentioned, we used the LOOCV, where the estimation of one project is performed by leaving this project out of the training set and making the prediction. Also, the standard accuracy (SA) and the effect size (Δ) measures were employed as evaluation criteria. Generally, a better model has a positive SA value, whereas a poorly predictive model has an SA value near zero, and negative SA values signify poor model predictions. The effect size measure (Δ), as explained in [37], was used to check whether the model estimates were produced by chance or not.

Three categories are defined as follows: large (for value close to 0.8), medium (for value close to 0.5), and small (for value close to 0.2). The large and medium categories mean that the model performs good predictions. Nevertheless, the SA measure reflects only the acceptance of the model, so it is not convenient enough to conclude the method's accuracy [38], [39]. Therefore, we used other criteria in (4-8) in conjunction with SA, which are characterized by being unbiased and insensitive to dissymmetry: mean absolute error (MAE), mean balanced relative error (MBRE), mean inverted balanced relative error (MIBRE) and logarithmic standard deviation (LSD).

We used the Borda counting technique to point out the most accurate methods as described in [40].

$$AbE_i = |y_i - \widehat{y_i}| \tag{2}$$

$$MRE = \frac{AbE_i}{y_i} \tag{3}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n} AbE_i \tag{4}$$

$$MBRE = \frac{1}{N}\sum_{i=1}^{N} \frac{AR_i}{\min(y_i,\widehat{y_i})} \tag{5}$$

$$MIBRE = \frac{1}{N}\sum_{i=1}^{N} \frac{AR_i}{\max(y_i,\widehat{y_i})} \tag{6}$$

$$LSD = \sqrt{\frac{\sum_{i=1}^{n}(\delta_i + \frac{z^2}{2})^2}{n-1}} \tag{7}$$

$$\delta_i = \ln(y_i) - \ln(\widehat{y_i}) \tag{8}$$

Where y_i is the actual cost, $\widehat{y_i}$ reflects the estimated cost of project number i, and z^2 reflects the residual variance estimate.

2.3.3. Statistical testing

The statistical test is still required to evaluate the significant difference between methods [41]. The statistical test used in this study is the Wilcoxon test, with a significance level of 0.05. This test aims to check if the evaluated SDCP methods are statistically different by using their absolute error (AbE). One reason why this test has been used is that the distribution of AbEs is not normal.

2.3.4. SDCP techniques configuration

To tune the hyperparameter values of all SDCP methods, we perform a series of GS evaluations by checking every possible parameter combination listed in Table 2. These parameters originated from the available studies assessing ML methods' performance [42], [43]. Hence, to search for the optimal parameter of each method using the GS technique, we evaluate all eventual combinations as shown in Table 2 based on the value of MAE. Then the best hyperparameter is the one that produces minimal MAE.

Table 2. Grid search hyperparameters

Method	Parameters
KNN	K={1,2,4,8,12}
MLP	Learning rate={0.01, 0.02, 0.03, 0.04, 0.05}
	Momentum={0.1, 0.2, 0.3, 0.4, 0.5}
	Layers={3, 5, 9, 16} Epochs={100, 500, 1000, 200}
CART	Minimum instances per leaf (I) = {1, 2, 3, 4, 5}
	The minimum number of samples required to split an internal node (S)= {1, 2, 4, 6,8,10}
	Max depth (M) = {1, 2, 4, 6, 8}

3. RESULTS AND DISCUSSION

We present in this section all the experimental results attained from the evaluation of the CFRT and the remaining four SDCP methods. Firstly, we tune the SDCP methods using GS, and after that, we used SA and Δ measures to assess their performance. Table 3 shows the values of SA and the effect size measures resulting from the empirical evaluation of the five SDCP methods throughout eight datasets.

According to Table 3, all SDCP methods perform better than randomly guessing over all datasets (positive value of SA measure). Nevertheless, some methods perform better than randomly guessing but with low SA values: (SA=0.19 for LR with the Kemerer dataset and SA=0.26 for MLP in Albrecht).

- CFRT has the highest values of SA in COCOMO (SA=88%), Albrecht (SA=77%), and Atkinson (53%).
- MLP has the highest values of SA in COCOMO (72%), ISBSG (64%), and china (58%).
- LR has the highest values of SA in ISBSG (64%), Albrecht (59%), and china (54%).
- CART has the highest values of SA in COCOMO (95%) and China (74%).
- KNN has the highest values of SA in COCOMO (89%) and Albrecht (65%).

The Δ values are all large for all methods in all datasets (Δ>0.74). So, the totality of estimations was not produced by chance:

- CFRT with COCOMO (9.04), China (8.93), and ISBSG (4.80).
- LR with China (10.3), ISBSG (6.3), and Desharnais (5.37).
- MLP with china (11.5), COCOMO (7.6), and ISBSG (6.5).
- CART with COCOMO (14.14), COCOMO (10.06), and ISBSG (5.05).
- KNN with China (10.6), COCOMO (9.2), and ISBSG (5.5).

Table 3. Evaluation of the five SDCP techniques in terms of SA and effect size, bold values reflect the highest SA in the corresponding dataset using LOOCV

Datasets	LR		MLP		CART		KNN		CFRT	
	SA	|Δ|	SA	|Δ|	SA	|Δ|	SA	|Δ|	SA	|Δ|
Albrecht	0.591	2.960	0.261	1.274	0.504	2.637	0.655	3.327	**0.771**	4.058
Atkinson	0.416	2.587	0.494	3.056	0.390	2.489	0.414	2.647	**0.532**	3.437
China	0.548	10.388	0.586	11.589	**0.746**	14.144	0.547	10.647	0.462	8.93
COCOMO	0.446	4.484	0.729	7.620	**0.957**	10.061	0.891	9.291	0.882	9.04
Desharnais	**0.504**	5.371	0.408	4.149	0.424	4.421	0.448	4.855	0.437	4.850
ISBSG	**0.641**	6.344	**0.641**	6.533	0.496	5.065	0.515	5.538	0.460	4.809
Kemerer	0.196	0.741	**0.492**	1.945	0.382	1.452	0.463	1.742	0.473	1.800
Miyazaki	0.504	1.830	0.371	1.390	0.587	2.152	0.476	1.688	0.435	1.553

Moreover, we perform a Borda counting technique to rank methods as shown in Table 4. Using the LOOCV method, we mention that no method from the five-used SDCP was ranked first overall datasets. Nevertheless, CFRT was rated first in three datasets (the Albrecht, Atkinson, and Kemerer) and the CART method, which is also ranked number one in three datasets: China, Desharnais, and Miyazaki. In contrast, KNN and MLP were ranked first just in COCOMO and ISBSG, respectively, whereas LR may be worst because it was ranked last in three datasets (the Atkinson, COCOMO, and Kemerer). The results indicate that all compared SDCP performed better than randomly guessing (positive SA), besides CFRT and CART are among the best (each one is rated first in 3 datasets), but typically no method is the most precise and reliable, and the best regarding its accuracy.

To explain the reached findings, we employed the Wilcoxon test. Table 5 shows the p-values:

- Concerning the Atkinson, china, Desharnais, ISBSG, and Miyazaki datasets: according to p-value, we notice that the difference in performance between CFRT and other methods is not statistically significant. So, the difference in ranks observed in Table 4 between all methods is insignificant in these datasets.
- Regarding the Albrecht dataset: CFRT significantly outperformed MLP, CART, and LR. Also for COCOMO: CFRT significantly outperformed LR, and MLP (p-value=0). The same finding is depicted by Borda counting as shown in Table 4.
- For Kemerer: the difference between CFRT and other techniques is not significant except for LR, where CFRT outperforms it significantly (confirm the results found in Table 4).

3.1. Comparison between CFRT and other SDCP methods

We compared CFRT and other techniques presented in the literature to analyze the accuracy of CFRT in SDCP effectively. We used the techniques reported in articles [44]–[46]. However, these papers used a different version of the dataset and validation methods besides different accuracy measures. Therefore, to make a fair comparison, we make various experimental designs using the same datasets, validation method, and accuracy measures as those used in the comparison papers. Tables 6 and 7 present the comparison results between CFRT and other methods on different used datasets using Pred (25%) and MMRE measures.

Table 4. The rank of SDCP methods using the Borda counting method

Method	Albrecht	Atkinson	China	COCOMO	Desharnais	ISBSG	Miyazaki	Kemerer
	CFRT	CFRT	CART	KNN	CART	MLP	CART	CFRT
	KNN	MLP	KNN	CART	KNN	CART	KNN	KNN
LOOCV	LR	KNN	MLP	CFRT	LR	KNN	LR	CART
	CART	CART	LR	MLP	CFRT	LR	CFRT	MLP
	MLP	LR	CFRT	LR	MLP	CFRT	MLP	LR

Table 5. Statistical significance (P-Value)

Datasets	LR	MLP	CART	KNN
Albrecht	**0.056**	**0.018**	**0.036**	0.132
Atkinson	0.174	0.390	0.174	0.071
China	0.931	0.684	1.0	0.954
COCOMO	**0.000**	**0.000**	0.196	0.999
Desharnais	0.853	0.343	0.414	0.546
ISBSG	0.999	0.999	0.990	0.857
Kememer	**0.027**	0.299	0.165	0.511
Miyazaki	0.465	0.167	0.990	0.558

Table 6. The values of Pred(25%) over all datasets

Réf	Methods	Validation technique	ALBRECHT	CHINA	COCOMO	DESHARNAIS	ISBSG	KEMERER	MIYAZAKI
[3]	LR	LOOCV	33	76	10	30		13	25
	SVR-PLOY		21	**93**	27	25		40	23
	SVR-RBF		42	90	31	34		**53**	**46**
	MLP		46	86	13	32		13	33
	M5P		25	79	32	32		2	10
	CA		33	38	26	36		33	31
	FA		45	38	46	26		47	42
[4]	SVR_BFE	LOOCV	42.857	83.333	36.842	39.13	31.111	20	28.571
	SVR_Boruta		28.571	75.333	34.211	34.783	24.444	0	42.857
	SVR		42.857	81.333	36.842	30.435	31.111	20	42.857
CFRT (our proposed model)		LOOCV	**58.33**	**58.33**	27.254	**70.238**	**37.662**	26.490	46.666
[5]	RF	30% holdout	42.86		51.31	43.48	33.33	66.67 (loocv)	
	RT		28.57		15.79	43.48	26.67		
CFRT (our proposed model)		30% holdout	**50**	**50**		**60.5**	**50**	**39.1**	

As shown in Table 6, The proposed CFRT model outperformed all the twelve compared techniques (LR, MLP, M5P, CA, FA, RF, RT, SVR, SVR-PLOY, SVR-RBF, SVR_BFE, and SVR_Boruta) in four datasets: Albrecht, COCOMO, Desharnais, and ISBSG using LOOCV and 30 fold cross-validation techniques. While in the Kemerer dataset, CFRT outperformed 9 out of 12 techniques except (RF, SVR_RBF, and FA). Concerning the Miyazaki dataset, CFRT outperformed 8 out of 12 techniques except (SVR_RBF, FA, SVR_Boruta, and SVR).

Table 7. The values of MMRE(25%) overall datasets

Réf	Methods	Validation technique	ALBRECHT	CHINA	COCOMO	DESHARNAIS	ISBSG	KEMERER	MIYAZAKI
[5]	RF	30% holdout	0.73		0.97	**0.42**	**1.17**	**0.57 loocv**	
	RT		0.97		2.74	0.52	3.71		
	CFRT (our proposed model)	30% holdout	**0.394**	**0.394**		**0.844**	0.570	**1.458**	
[4]	SVR_BFE	LOOCV	**0.548**	**0.17**	1.242	0.462	**1.092**	1.334	1.367
	SVR_Boruta		0.566	0.242	1.524	**0.457**	1.559	1.671	**0.527**
	SVR		0.583	0.187	1.375	0.467	1.478	1.235	0.553
	CFRT (our proposed model)	LOOCV	0.782	0.782	1.020	**0.677**	0.684	2.05	**0.593**

According to MMRE values (Table 7), CFRT outperformed RT in Albrecht, COCOMO, and ISBSG. Also, it outperformed RF in Albrecht, COCOMO, with similar MMRE in Desharnais, ISBSG, and Kemerer. In addition, CFRT outperformed SVR, SVR_Boruta, and SVR_BFE in COCOMO Kemerer, while in Miyazaki, it outperformed SVR_BFE only. Finally, we conclude from these comparison results that CFRT is a promising and competitive technique to the others existing in the literature, so it can effectively be used for SDCP purposes.

3.2. Threats to validity

The findings of this study are not definitive, and they are also subject to some criticism. Mainly, they are related to the internal and external threats to validity. Which are detailed as follows:

3.2.1. Internal validity

The 10-fold cross-validation used in the grid search to select the best hyperparameters may threaten the resulting conclusions of this paper even though many studies use the same cross-validation [29] and the grid search method for parameter selection [14], [33]. This can have a significant impact on the obtained results. Additionally, the chosen validation method may be perceived as another threat. However, in this study, we surmount it using the LOOCV method, which is considered a stable method (even if the experimentation is repeated, the same results are produced), contrary to other validation methods [47]. Another reason behind the use of LOOCV is its behavior of producing reduced bias and a more significant variance of predictions [48]. Moreover, we limited this study to handle just numeric features, not categorical ones. Therefore, taking into account, categorical features remain, since many popular software databases include categorical attributes.

3.2.2. External validity

In this study, we deal with the following two external validity issues: the first is about the probability of generalizing the empirical results, while the second concerns the experimental design. The chosen datasets have the following characteristics: They differ in their sizes and the number of features, they include projects from several domains, the data (projects) originated from distinct organizations, and they were gathered at various periods. Therefore, the used datasets allow us to make a suitable conclusion thanks to their typical characteristics and number (eight datasets).

4. CONCLUSION

This paper proposes a new method of FRTs: the CFRT. The comparison is made with four SDCP methods, notably LR, MLP, KNN, and CART employing eight datasets and the LOOCV method. Firstly, we assessed the performance of the five SDCP methods using the SA measure. After that, we calculated the effect size to verify that the estimations were not generated by chance. Furthermore, due to the insufficiency of the SA criterion to make any conclusion concerning the accuracy of the compared techniques, we used some reliable criteria to perform the Borda counting method that enables us to rank the SDCP methods according to their accuracy, and finally to validate and make a justification of the attained findings we perform a statistical Wilcoxon test.

We intend from this comparative study to answer the three previously asked questions: i) ReQu1: Does the CFRT technique perform better than the four techniques according to the SA measure using LOOCV? Using LOOCV and according to SA values, no specific method performed better than the others over whole datasets. ii) ReQu2: Does the CFRT present the most precise technique compared to the four SDCP methods by using LOOCV? We mention that no method from the five-used SDCP was rated one over all datasets. However, CFRT and CART are among the best (each one is rated first in 3 datasets) and iii) ReQu3: Does CFRT significantly outperform the other SDCP methods according to the statistical test? The performance of CFRT compared to the other techniques differs depending on the dataset and the SDCP method employed.

In summary, affirming that a specific method performs better in any context is a problematic issue, considering that distinct findings were obtained when changing the context of estimation, such as distinct scenarios of evaluations, distinct datasets, and distinct SDCP methods. However, according to the performed analysis, CFRT is among the best. It is rated first in 3 datasets according to four accuracy measures: MAE, MIBRE, MBRE, and LSD. Also, according to the Pred(25%) values, the proposed CFRT model outperformed all the twelve compared techniques (LR, MLP, M5P, CA, FA, RF, RT, SVR, SVR-PLOY, SVR-RBF, SVR_BFE, and SVR_Boruta) in four datasets: Albrecht, COCOMO, Desharnais, and ISBSG using LOOCV and 30 fold cross-validation techniques. Therefore, CFRT remains a promising SDCP method. Regarding future work, the proposed model of "CFRT" can be optimized using an optimization algorithm such as an "optimized artificial immune network (Opt-ainet)" to have an improved model, which will then be applied for SDCP concerns, and more specifically for agile projects.

REFERENCES

[1] A. Kaushik, D. Kr. Tayal, and K. Yadav, "A comparative analysis on effort estimation for agile and non-agile software projects using DBN-ALO," *Arab. J. Sci. Eng.*, vol. 45, no. 4, pp. 2605–2618, Apr. 2020, doi: 10.1007/s13369-019-04250-6.

[2] A. B. Nassif, L. F. Capretz, D. Ho, and M. Azzeh, "A treeboost model for software effort estimation based on use case points," in *2012 11th International Conference on Machine Learning and Applications*, 2012, pp. 314–319, doi: 10.1109/ICMLA.2012.155.

[3] E. Mendes, "A comparative study of cost estimation models for web hypermedia applications," *Empirical Software Engineering*, vol. 8, no. 2, pp. 163-196, 2003.

[4] A. Najm, A. Zakrani, and A. Marzak, "Decision trees based software development effort estimation: a systematic mapping study," in *2019 International Conference of Computer Science and Renewable Energies (ICCSRE)*, Agadir, Morocco, Jul. 2019, pp. 1–6, doi: 10.1109/ICCSRE.2019.8807544.

[5] A. Idri and S. Elyassami, "Applying fuzzy ID3 decision tree for software effort estimation," *IJCSI Int. J. Comput. Sci.*, vol 8, no. 4, p. 8, 2011, doi: 10.14569/SpecialIssue.2011.010314.

[6] F. Gasir, Z. Bandar, and K. Crockett, "Elgasir: An algorithm for creating fuzzy regression trees," in *2009 IEEE International Conference on Fuzzy Systems*, Aug. 2009, pp. 332–337, doi: 10.1109/FUZZY.2009.5277128.

[7] A. Suarez and J. F. Lutsko, "Globally optimal fuzzy decision trees for classification and regression," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 12, pp. 1297–1311, Dec. 1999, doi: 10.1109/34.817409.

[8] A. Najm, A. Zakrani, and A. Marzak, "Systematic review study of decision trees based software development effort estimation," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 11, no. 7, 2020, doi: 10.14569/IJACSA.2020.0110767.

[9] V. A. Gupta and S. Soni, "Review of fuzzy decision tree: an improved decision making classifier," *International Journal of Computer Science and Engineering*, vol. 1, no. 9, pp. 1-5, 2014.

[10] H. Zheng, J. He, Y. Zhang, G. Huang, Z. Zhang, and Q. Liu, "A general model for fuzzy decision tree and fuzzy random forest," *Computational Intelligence*, vol. 35, no. 2, pp. 310–335, May 2019, doi: 10.1111/coin.12195.

[11] W. Pedrycz and Z. A. Sosnowski, “C–fuzzy decision trees,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 35, no. 4, pp. 498–511, Nov. 2005, doi: 10.1109/TSMCC.2004.843205.

[12] L. Castin and B. Frenay, "Clustering with decision trees: divisive and agglomerative approach," in *26th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2018, pp. 455-460.

[13] J. Wen, S. Li, Z. Lin, Y. Hu, and C. Huang, "Systematic literature review of machine learning based software development effort estimation models," *Information and Software Technology*, vol. 54, pp. 41–59, 2012, doi: 10.1016/j.infsof.2011.09.002.

[14] X. Ma, Y. Zhang, and Y. Wang, "Performance evaluation of kernel functions based on grid search for support vector regression," in *2015 IEEE* 7th *International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM)*, Jul. 2015, pp. 283–288, doi: 10.1109/ICCIS.2015.7274635.

[15] A. Idri and S. Elyassami, "A fuzzy decision tree to estimate development effort for web applications," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 1, no. 3, 2011, doi: 10.14569/SpecialIssue.2011.010314.

[16] S. Elyassami, "Investigating effort prediction of software projects on the ISBSG dataset," *International Journal of Artificial Intelligence & Applications (IJAIA)*, vol. 3, no. 2, pp. 121–132, Mar. 2012, doi: 10.5121/ijaia.2012.3210.

[17] S. Elyassami and A. Idri, "Evaluating software cost estimation models using fuzzy decision trees," *Recent Adv. Knowl. Eng. Syst. Sci. WSEAS Press*, p. 6, 2013.

[18] S. Elyassami and A. Idri, "Fuzzy model for an early estimation of software development effort," *Int. J. Appl. Math. Inform.*, vol. 7, no. 3, p. 9, 2013.

[19] A. S. Andreou and E. Papatheocharous, "Software cost estimation using fuzzy decision trees," in *2008 23rd IEEE/ACM International Conference on Automated Software Engineering, L'Aquila*, Sep. 2008, pp. 371–374, doi: 10.1109/ASE.2008.51.

[20] E. Papatheocharous and A. S. Andreou, "A hybrid software cost estimation approach utilizing decision trees and fuzzy logiC," *International Journal of Software Engineering and Knowledge Engineering*, vol. 22, no. 03, pp. 435–465, May 2012, doi: 10.1142/S0218194012500106.

[21] E. Papatheocharous and A. S. Andreou, "Classification and prediction of software cost through fuzzy decision trees," in *International Conference on enterprise information systems*, 2009, vol. 24, pp. 234–247, doi: 10.1007/978-3-642-01347-8_20.

[22] H.-W. Chiu, C.-S. Ouyang, and S.-J. Lee, "Improved c-fuzzy decision trees," in *2006 IEEE International Conference on Fuzzy Systems, Vancouver*, 2006, pp. 1763–1768. 10.1109/FUZZY.2006.1681944.

[23] S. Lalwani, H. Sharma, S. C. Satapathy, K. Deep, and J. C. Bansal, "A survey on parallel particle swarm optimization algorithms," *Arabian Journal for Science and Engineering,* vol. 44, no. 4, pp. 2899–2923, Apr. 2019, doi: 10.1007/s13369-018-03713-6.

[24] J. R. Velasco, S. Lopez, and L. Magdalena, "Genetic fuzzy clustering for the definition of fuzzy sets," in *Proceedings of 6th International Fuzzy Systems Conference*, 1997, vol. 3, pp. 1665–1670, doi: 10.1109/FUZZY.1997.619790.

[25] L. Liu and W. Xu, "Fuzzy data clustering using artificial immune network," *In International Conference on Intelligent Computing*, 2006, pp. 195-205, doi: 10.1007/978-3-540-37275-2_26.

[26] S. K. Shukla and M. K. Tiwari, "GA guided cluster based fuzzy decision tree for reactive ion etching modeling: a data mining approach," *IEEE transactions on semiconductor manufacturing,* vol. 25, no. 1, pp. 45–56, Feb. 2012, doi: 10.1109/TSM.2011.2173372.

[27] Ł. Gadomer and Z. A. Sosnowski, "Fuzzy random forest with c–fuzzy decision trees," in *IFIP International Conference on Computer Information Systems and Industrial Management*, 2016, pp. 481–492, doi: 10.1007/978-3-319-45378-1_43.

[28] J. C. Bezdek, "Pattern recognition with fuzzy objective function algorithms," Boston, MA: Springer US, 1981, doi: 10.1007/978-1-4757-0450-1.

[29] A. L. I. Oliveira, P. L. Braga, R. M. F. Lima, and M. L. Cornélio, "GA-based method for feature selection and parameters optimization for machine learning regression applied to software effort estimation," *Inf. Softw. Technol.,* vol. 52, no. 11, pp. 1155–1166, Nov. 2010, doi: 10.1016/j.infsof.2010.05.009.

[30] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *Am. Stat.,* vol. 46, no. 3, pp. 175–185, Aug. 1992, doi: 10.1080/00031305.1992.10475879.

[31] E. Mendes, I. Watson, C. Triggs, N. Mosley, and S. Counsell, "A comparison of development effort estimation techniques for Web hypermedia applications," in *Proceedings Eighth IEEE Symposium on Software Metrics*, 2002, pp. 131–140, doi: 10.1109/METRIC.2002.1011332.

[32] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, "Classification and regression trees," Belmont, Calif.: Wadsworth Int'l Group, 1984.

[33] M. Hosni, A. Idri, A. Abran, and A. B. Nassif, "On the value of parameter tuning in heterogeneous ensembles effort estimation," *Soft Computing,* vol. 22, no. 18, pp. 5977–6010, Sep. 2018, doi: 10.1007/s00500-017-2945-4.

[34] A. Bakır, B. Turhan, and A. B. Bener, "A new perspective on data homogeneity in software cost estimation: a study in the embedded systems domain," *Software Quality Journal,* vol. 18, no. 1, pp. 57–80, Mar. 2010, doi: 10.1007/s11219-009-9081-z.

[35] T. Menzies *et al.,* "The promise repository of empirical software engineering data," *Promisedata,* Jun. 2012.

[36] C. Lokan, T. Wright, P. Hill, and M. Stringer, "Organizational benchmarking using the ISBSG data repository," *IEEE Softw.,* vol. 18, no. 5, pp. 26–32, Oct. 2001, doi: 10.1109/52.951491.

[37] J. Cohen, "Quantitative methods in psychology," *Psychological Bulletin,* vol. 112, no. 1, pp. 155–159, 1992, doi: 10.1037/0033-2909.112.1.155.

[38] M. Azzeh and A. B. Nassif, "A hybrid model for estimating software project effort from use case points," *Appl. Soft Comput.,* vol. 49, pp. 981–989, Dec. 2016, doi: 10.1016/j.asoc.2016.05.008.

[39] E. Kocaguneli, T. Menzies, and E. Mendes, "Transfer learning in effort estimation," *Empir. Softw. Eng.,* vol. 20, no. 3, pp. 813–843, Jun. 2015, doi: 10.1007/s10664-014-9300-5.

[40] 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, May 2015, doi: 10.1016/j.jss.2015.01.028.

[41] J. Demsar, "Statistical comparisons of classifiers over multiple data sets," *The Journal of Machine learning research*, vol. 7, pp. 1-30, 2006.

[42] P. L. Braga, A. L. I. Oliveira, G. H. T. Ribeiro, and S. R. L. Meira, "Bagging predictors for estimation of software project effort," in 2007 *International Joint Conference on Neural Networks*, Aug. 2007, pp. 1595–1600, doi: 10.1109/IJCNN.2007.4371196.

[43] L. L. Minku and X. Yao, "Ensembles and locality: Insight on improving software effort estimation," *Inf. Softw. Technol.,* vol. 55, no. 8, pp. 1512–1528, Aug. 2013, doi: 10.1016/j.infsof.2012.09.012.

[44] A. Idri, I. Abnane, and A. Abran, "Evaluating Pred(p) and standardized accuracy criteria in software development effort estimation," *J. Softw. Evol. Process,* vol. 30, no. 4, p. e1925, Apr. 2018, doi: 10.1002/smr.1925.

[45] A. Zakrani, M. Hain, A. Namir, and Faculte des Sciences Ben M'sik, "Software development effort estimation using random forests: an empirical study and evaluation," *Int. J. Intell. Eng. Syst.,* vol. 11, no. 6, pp. 300–311, Dec. 2018, doi: 10.22266/ijies2018.1231.30.

[46] A. Zakrani, M. Hain, and A. Idri, "Improving software development effort estimating using support vector regression and feature selection," *IAES International Journal of Artificial Intelligence (IJ-AI),* vol. 8, no. 4, p. 399, Dec. 2019, doi: 10.11591/ijai.v8.i4.pp399-410.

[47] 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, Jul. 2013, doi: 10.1016/j.jss.2013.02.053.

[48] E. Kocaguneli, T. Menzies, and J. W. Keung, "On the value of ensemble effort estimation," *IEEE Trans. Softw. Eng.*, vol. 38, no. 6, pp. 1403–1416, Nov. 2012, doi: 10.1109/TSE.2011.111.

BIOGRAPHIES OF AUTHORS

Assia Najm is a Ph.D. student at Hassan II university at Casablanca, department of Mathematics and Computer Sciences. She received an engineer's degree in telecommunication and technology of information from INPT, Rabat, Morocco, in 2014. Her research interests include software cost estimation, software metrics, fuzzy logic, decision trees, and optimization. She can be contacted at email: assia.najm@gmail.com.

Abdelali Zakrani is an associate professor at Hassan II university at Casablanca, He received the B.Sc. degree in Computer Science from Hassan II University, Casablanca, Morocco, in 2003, and his DESA degree (M.Sc.) and PhD. in the same major from University Mohammed V, Rabat, in 2005 and 2012 respectively. His research interests include software cost estimation, software metrics, fuzzy logic, neural networks, decision trees. He can be contacted at email: zakrani@gmail.com.

Abdelaziz Marzak is currently a professor of computer science at the Faculty of Science Ben M'sik Casablanca, where he is the head of technologies of information and communication. He has several publications on the Web and databases. He received a Ph.D. in computer science in 2000 and a Ph.D. in computer science in 1997 both from Mohammed V University at Rabat, Morocco. He can be contacted at email: marzak@hotmail.com.