Stacking regularization in analogy-based software effort estimation

Anupama Kaushik¹ · Prabhjot Kaur¹ · Nisha Choudhary¹ · Priyanka¹

Accepted: 12 November 2021
© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2021

Abstract
Analogy-based estimation (ABE) estimates the effort of the current project based on the information of similar past projects. The solution function of ABE provides the final effort prediction of a new project. Many studies on ABE in the past have provided various solution functions, but its effectiveness can still be enhanced. The present study is an attempt to improve the effort prediction accuracy of ABE by proposing a solution function SABE: Stacking regularization in analogy-based software effort estimation. The core of SABE is stacking, which is a machine learning technique. Stacking is beneficial as it works on multiple models harnessing their capabilities and provides a better estimation accuracy as compared to a single model. The proposed method is validated on four software effort estimation datasets and compared with the already existing solution functions: closet analogy, mean, median and inverse distance weighted mean. The evaluation criteria used are mean magnitude of relative error (MMRE), median magnitude of relative error (MdMRE), prediction (PRED) and standard accuracy (SA). The results suggested that the SABE showed promising performance for almost all the evaluation criteria when compared with the results of the earlier studies.

Keywords Analogy-based estimation · Stacking · Software effort estimation · Machine learning

1 Introduction
Cost estimation is a methodology of forecasting the expense of executing a project with a given framework. A cost estimate is a summary of all costs involved, from commencement to culmination (duration of the project). This pays for every item required for the project, from supplies to manpower, and estimates a total amount deciding the cost of a project. Cost estimation can be utilized for determining the performance of a project. Accurate cost estimation leads to a successful project, and inaccurate estimations result in a project failure. Analogy-based estimation (ABE) is a cost-evaluation method used for software projects. It was commonly used and researched as an alternative to multiple regression models and expert judgment techniques over the last decennium (Shepperd and Schofield 1997).

Portraying the ABE approach in brief: Initially, the project to be evaluated is kept alongside the projects, similar in characteristics and present in historical archive. After that, one or more identical project is discovered from the archive by a predefined similarity function. At last, to generate the final estimation, heuristic function is applied to predict the effort of a new project based on the information of the retrieved projects.

In this paper, the authors proposed a method SABE (Stacking regularization in analogy-based software effort estimation). The authors utilized stacked generalization which is a prevalent concept related to any knowledge feeding scheme from one generalizer to another afore the final approximation is made (Wolpert 1992). It is a machine learning technique which couples the capabilities of various heterogeneous models and provides better estimate than a single model. The two techniques used in designing SABE are polynomial regression and LASSO (Least Absolute Shrinkage and Selection Operator) regression. Polynomial regression is a specialized version of a linear regression in which a polynomial equation shows the relationship between the objective variable and the independent variables in data, and modeled with a curvi-
linear relationship (Ostertagová 2012). LASSO is a kind of regression analysis that uses shrinkage. It performs variable selection and regularization so as to increase the prediction accuracy and interpretability of the statistical model it generates (Tibshirani 1996). More details on these models are described in the next section. Regression models have been long used in software estimation studies by various authors (Jørgensen 2004), (Yücalar et al. 2016), (Nassif et al. 2019), (Anandhi and Chezian 2014) and provided good cost estimates. The machine learning approaches are employed not only for predictions in software engineering, but they are also utilized in various other fields such as text document clustering and COVID-19 predictions (Abualigah and Hanandeh 2015; Abualigah and Khader 2017; Abualigah et al. 2019, 2021).

The remaining portion of this paper is structured according to the following: Section 2 reviews the relevant research; Section 3 presents a succinct observation regarding the context of this study; Section 4 addresses the proposed Stacking regularization in analogy-based software effort estimation (SABE). Section 5 describes the dataset and evaluation criteria. Section 6 provides the experimental evaluation and results. Section 7 presents the statistical performance assessment of the proposed technique, and Sect. 8 finally concludes the paper.

2 Related work

A variety of research (Kumar et al. 2020) have been performed on numerous techniques that are acclimatized to estimate the software cost. In fuzzy analogy software effort estimation (FASEE), Ezghari and Zahi (Ezghari and Zahi 2018) suggested a new methodology to the uncertainty management. The proposal introduced consistency criteria in the estimation model in order to increase the data quality and infer a consistent possibility distribution, called Consistent fuzzy analogy software effort estimation (C-FASEE). Idri et al. (Idri et al. 2016) proposed a model using Missing Data (MD) technique. They explored MD strategies with classical analogy and fuzzy analogy. The researchers used three MD methods (toleration, elimination and imputation techniques for K-nearest neighbors), three missing mechanisms (MCAR: Missing completely at random, MAR: Missing at random, NIM: Non-ignorable missing) and MD percentages from 10 percent to 90 percent. Their findings revealed that regardless of MD technique, the data set used, missing mechanism or MD percentage, fuzzy analogy provided more accurate estimates in terms of the standard accuracy measure (SA) than the classical analogy. Their analysis showed that the utilization of k-nearest neighbors imputation may increase the predictability of analogy-based techniques than toleration or deletion. A model on analogy-based software effort estimation is proposed by Benala and Mall (Benala and Mall 2018) using differential evolution. They investigated the efficacy of the differential evolution (DE) algorithm by applying five mutation strategies to optimize the feature weights of homogeneous attribute functions of analogy-predicated estimation (ABE). The empirical analysis is designated as DABE: differential evolution in analogy-based software development effort estimation by them. Singh et al. (Singh et al. 2018a) presented an incipient variant of DE utilizing homeostasis adaption-based mutation operator (HABMO) and applied it for software cost estimation. They used mean magnitude of relative error (MMRE), mean magnitude of relative error relative to the estimate (MMER), mean squared error (MSE) and root mean squared error (RMSE) as the evaluation matrices. They found the proposed technique worked best in comparison with different variants of DE. Azzeh et al. (Azzeh et al. 2015) presented a model for analogy-based effort estimation, entitled, An empirical evaluation of ensemble adjustment methods for analogy-based effort estimation. In their paper, they investigated the potential of ensemble learning for variants of adjustment methods used in analogy-based effort estimation. Their results were subjected to statistical paramounity testing, and showed consequential plausible amendments in predictive performance where ensemble methods were applied. As a case study, Effendi et al. (Effendi et al. 2019) utilized a student desk platform in their paper to expound a use case point method. They utilized the data from the actual software development in their work and adjusted the effort using the use case point process. The effort computed for three separate applications by using the use case point was compatible with the actual result. Idri et al. (Idri et al. 2016) used classical and fuzzy analogy ensembles for estimation of software development effort. They conducted the study on 100/60 variants of classical/fuzzy analogy techniques over seven datasets. Using the Scott–Knott statistical test, these variants were clustered and ranked. The results revealed that there was no strongest single classical or fuzzy analogy technique for all the datasets, and the ensembles that were built were conventionally better than the single technique. Phannachitta (Phannachitta 2020) proposed an innovative approach by introducing a combined effort adapter for analogy estimation. They cumulated gradient boosting machine algorithm and conventional adaptation technology based on productivity adjustment. They found their technique was at par in comparison with the already existing effort estimation techniques used in the study. Zima (Zima 2015) presented a case-based reasoning model of cost estimation at the preliminary stage of a construction project. In the paper, it was postulated that the benefits of the model presented were its flexibility and the ease of calculation. Singh et al. (Singh et al. 2018b) utilized Improved environmental adaption model with real parameter (IEAM-RP) to estimate the software effort. They used
NASA dataset to evaluate their technique. The experimental results demonstrated the effectiveness of IEAM-RP. Suresh Kumar et al. (2021) proposed a gradient boosting regressor model. The model is compared with stochastic gradient descent, k-nearest neighbor, decision tree, bagging regressor, random forest regressor, Ada-boost regressor, and gradient boosting regressor. The authors evaluated the model using mean absolute error (MAE), mean square error (MSE), root mean square error (RMSE), and $R^2$. They showed the results on Cocomo81 and China datasets.

Above are the few experimental studies on software cost estimation and modern techniques are integrated from time to time. In this article, the authors have consequently endeavored to incorporate a software cost estimation methodology SABE (Stacking regularization in analogy-based software effort estimation). The SABE method has not been used up till now for analogy-based estimation as per the current knowledge of the authors.

### 3 Background techniques

#### 3.1 Stacking

Stacking (infrequently known as Stacked Generalization) is an ensemble algorithm of machine learning. It integrates the results from several models of machine learning. First proposed by David Wolpert (Wolpert 1992) in 1992, the key purpose is to minimize the error of generalization. As per his view, it is a more sophisticated variant of cross-validation.

The core concept behind the framework of stacking is to render forecasts utilizing one or more first-stage models and their estimates as feature, to match up to one or more second-level models. The principle of stacking is shown in Fig. 1.

#### 3.2 Polynomial regression

Polynomial regression is a form of linear regression, where the association between the independent $x$ variable and the dependent $y$ variable is formulated as a polynomial $n^{th}$ degree. The first proposal was created in 1815 (Stigler 1974; Gergonne 1974) for an experiment with polynomial regression. This is a special case of linear regression where the data with a curvilinear relationship between target variable and independent variables are compiled in a polynomial equation. The value of the target variable changes, with the predictor ($s$), in a curvilinear relationship uniformly. The fundamental objective of regression analysis is for modeling the expected value of the dependent variable $y$ according to the value of the independent variable $x$. The authors utilized Eq.(1) in simple regression:

$$y = p + qx + e$$  \hspace{1cm} (1)

where $y$ is the dependent variable on $x$, $p$ is $y$ intercept, $q$ defines the slope, $e$ is the rate of error. The above equation is conventionally model for $n^{th}$ term in equation 2:

$$y = p + qx + q2x^2 + ... + qnx^n$$  \hspace{1cm} (2)

As the regression function is linear in terms of unknown variables, these models are consequently linear from the estimation perspective.

### 3.3 LASSO regularization

#### 3.3.1 Regularization

The regularization is considered to make things regular or acceptable. It regularizes or decreases the coefficient toward zero. It is a methodology practiced by introducing an external penalty term to the error function to adjust the process. It makes the optimal solution unique. The supplemental term regulates the exceedingly shifting function to avert extreme values from being taken by the coefficients. This technique avoids the risk of overfitting and improves model interpretability.

#### 3.3.2 LASSO regression

LASSO regression is a regularization technique used over regression methods for accurate predictions. It was published in 1986 (Santosa and Symes 1986) in the literature of geophysics and afterward was rediscovered and popularized separately by Robert Tibshirani in 1996 (Tibshirani 1996). LASSO regression implements L1 regularization that integrates a penalty equivalent to the absolute value of the coefficient’s magnitude. This method of regularization will result in sparse models containing fewer coefficients; certain coefficients can be zero and omitted from the model. Larger penalties lead to coefficient values that are more proximate to zero, which is ideal for making simpler models. Equation 3 indicates the cost feature of LASSO regression.

$$\sum_{i=1}^{M} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{M} \left( y_i - \sum_{j=0}^{p} w_j x_{ij} \right)^2 + \lambda \sum_{j=0}^{p} |w_j|$$  \hspace{1cm} (3)

where $\lambda$ is the shrinkage amount:

1. When $\lambda=0$, no parameters are eliminated. The calculation is equivalent to that of linear regression.
2. When $\lambda$ rises, the coefficients are gradually set to zero and discarded.
3. As there is an increase in $\lambda$, the bias grows.
4. As there is a decrease in $\lambda$, the variance grows.
3.4 Analogy-based effort estimation

The analogous estimation is a method utilized for calculating specific parameters for future operation by utilizing historical data values. In analogy-based effort estimation, the effort of a new project is calculated based on the past development experience of similar projects. The ABE method broadly consists of a four-step procedure (Aamodt and Plaza 1994):

1. First, extract the most similar cases to the current project from the historical database.
2. Second, reuse the knowledge of these similar projects in effort estimation of the current project.
3. Third, evaluate the solution function and find the effort of the current project.
4. Fourth, preserve the solution for future estimations.

3.4.1 Similarity function

The similarity degree is determined by the similarity function between the projects. Four similarity functions are utilized in this paper, namely the Euclidean distance, Manhattan distance, Jaccard distance and Minkowski distance. The predilection of similarity measure determines the option of k-nearest neighbors. The most acknowledged variant of computing distance is the Euclidean distance. Euclidean distance is:

$$d(P_1, P_2) = \sqrt{\sum_{k=1}^{n} (P_{1k} - P_{2k})^2}$$  \hspace{1cm} (4)

where $P_1$ is the targeted project and $P_2$ is the historical project and $n$ is the number of attributes. Distance from Manhattan which is shown in Eq.(5) is a metric in which the distance between the two points is the sum of their Cartesian coordinates’ absolute differences, where $P_1$ is the targeted project and $P_2$ is the historical project and $n$ is the number of attributes.

$$d(P_1, P_2) = \sum_{k=1}^{n} |P_{1k} - P_{2k}|$$ \hspace{1cm} (5)

Coefficient of Jaccard distance formula is a metric used to explain the distinctions between the samples. It can be shown...
Stacking regularization in analogy-based software effort estimation

The distance from Minkowski given in (7) is a simplified statistical version of Euclidean distance and Manhattan distance.

\[
d(m, n) = \sqrt[\alpha]{\sum_{i=1}^{j} |x_{m,i} - x_{n,i}|^\alpha}
\]

In the above equation, Minkowski distance between data record \( m \) and \( n \) is represented by \( d(m, n) \). \( j \) is the total number of variables of \( x \), \( i \) is the variable’s index, and \( \alpha \) is the Minkowski metric order.

### 3.4.2 k-nearest neighbors

The number of \( k \) — most proximate neighbors is a critical parameter impacting the estimation of a new project. \( k = 1 \) to 5 is used in this article, as it is perceived in several researches (Benala and Mall 2018; Jørgensen et al. 2003; Huang and Chiu 2006)

#### 3.4.3 Solution functions

Once \( k \) most similar projects are chosen, certain statistics are estimated depending upon the chosen similar projects. This is called solution function, and it is used in the final forecast of a new project. During experimental analysis, the following assessment methods are utilized by the author as the substructure for the solution function: the closet analogy (most similar project) (Walkerden and Jeffery 1999), the mean of chosen similar projects (Shepperd and Schofield 1997), the median of chosen similar projects (Angelis and Stamelos 2000) and the inverse distance weighted mean (Kadoda et al. 2000). The mean is the average or a quantified “central” value of the efforts of \( k \) most homogeneous projects, where \( k > 1 \).

The median is the median of the efforts of the most homogeneous projects, where \( k > 2 \). When the number of the most similar projects increases, the median becomes a more reliable figure compared with the mean.

The inverse distance-weighted mean (IWM) indicates that the importance of more related projects is greater than the less homogeneous projects. It can be defined as (8):

\[
\hat{c}_p = \frac{\sum_{k=1}^{n} Sim(P, P_k) C_{p_k}}{\sum_{i=1}^{n} Sim(P, P_k) C_{p_k}}
\]

### 4 Proposed work

For decades, incorrect cost estimate of the project has plagued software developments. Weak forecasts have not only resulted in projects exceeding budget and timeline but also, in many cases, being entirely terminated. The quality to accurately estimate software development time, cost, and manpower, changes as more incipient methodologies supersede old ones. Consequently, an effective and precise software cost estimation model is highly required in software project management. The proposed framework SABE (Stacking regularization in analogy-based software effort estimation) is discussed in this section. The conventional analogy-based estimation approach is first considered to produce an unadjusted retrieval effort when a new target project comes to be predicted. The primary aim of the adjustment is to locate the ‘update’ that transforms the effort from the projects retrieved into the target effort.

In SABE framework, the model is fed with a series of projects. One project is kept as a test project and the remaining projects as historical projects. After applying the similarity functions as discussed in Sect. 3.4.1, the closest analogy projects are determined. These projects are now subjected to a solution function as explained in Sect. 3.4.3, and Stacking Regularization (SR) algorithm. The effort computed by both the procedures is then adapted to provide the final SABE solution function. The flowchart of SABE is given in Fig. 2. The code for the proposed model and other methods of comparative adaptation are implemented in Python. The algorithm of effort prediction and adjustment through SABE is as follows:

**Step 1:** Start with Data Preprocessing.
**Step 2:** Project number \( P_i \) is exempted from the dataset as a test project, and the remainder projects are considered as historic.

| Input | Output |
|-------|--------|
| \( d_1(P_1, P_{a_1}) \) | \( d_c(P_1, P_{a_1}) \) |
| \( d_1(P_2, P_{a_2}) \) | \( d_c(P_2, P_{a_2}) \) |
| \( d_1(P_3, P_{a_3}) \) | \( d_c(P_3, P_{a_3}) \) |
| \( \ldots \) | \( \ldots \) |
| \( d_1(P_n, P_{a_n}) \) | \( d_c(P_n, P_{a_n}) \) |

Table 1 Difference table (Azzeh, 2011)

\( \text{Sim}(P, P_k) \) shows the similarity between projects \( P_k \) and \( P \), and \( C_{p_k} \) denotes the cost of the most similar project \( P_k \).
Fig. 2 Flowchart of SABE

**Step 3**: The analogy projects $P_a$ most similar to the test project $P_t$ are retrieved using various similarity functions as discussed in 3.4.1 for each k-nearest neighbors ($k = 1 \ldots 5$). Equation (9) depicts the Euclidean distance.

$$SM(P_t, P_j) = \sqrt{\sum_{k \in N} (P_t, P_j)^2}, j = 1, 2, 3, \ldots, n \quad (9)$$

where SM indicates measure of similarity, $N$ is the count of predictor attributes, $P_t$ and $P_j$ are projects under investigation. This step is repeated with the rest of the similarity functions, i.e., Manhattan distance, Minkowski distance and Jaccard distance.

**Step 4**: The closest analog projects, i.e., $P_a$, are utilized to perform two variety of operations indicated under step 4.1 and step 4.2.

**Step 4.1(a)**: Fed closest analogies to the solution function as discussed in Sect. 3.4.3 to find effort of the project to be estimated. The effort computed through one of the solution functions is depicted in Eq.(10).

$$Effort_s = \frac{\sum_{a=1}^{K} SM(P_t, P_a) \times Size(P_a)}{\sum_{a=1}^{K} SM(P_t, P_a)} \quad (10)$$

where $SM(P_t, P_a)$ indicates measure of similarity between $P_t$ and $P_a$, $Size(P_a)$ is the size of the project $P_a$ as given in the dataset.

**Step 4.1(b)**: Make the difference table as shown in Table 1 (Azzeh 2011), taking difference between the predicted effort using Eq.(10) and the test project effort along with all predictable attributes as shown in Table 1.

$$d_k(P_t, P_a) = P_{tk} - P_{ak}, k = 1, 2, 3, \ldots, N \quad (11)$$

$$d_e(P_t, P_a) = P_{te} - P_{ae} \quad (12)$$

where $d_k$ is the difference between $t^{th}$ project $P_t$ and its closest analogy $P_a$ at $k^{th}$ attribute, $d_e$ is the difference between $t^{th}$ project $P_t$ and its closest project $P_a$ at effort attribute.

**Step 4.2**: Provide the closest analogies to the Stacking Regularization (SR) algorithm, which constructed SABE dependent adaptation mechanism as illustrated in Fig. 4 to estimate $Effort_{SR}$.

**Step 5**: The predicted difference table and estimated effort obtained from the SR algorithm is used to adapt and update the goal project effort as given in Eq.(13).

$$Effort(P_t) = Effort_{SR} \pm d_e(P_t, P_a) \quad (13)$$

**Step 6**: Final calculation of MRE, MMRE, MdMRE, PRED and SA is performed.

**Step 7**: End.

The process of SR algorithm is described as below:
Stacking regularization in analogy-based software effort estimation

Step 1: Read each test project and its corresponding similar projects obtained using similarity functions for training.

Step 2: Set polynomial regression features as degree from 2 to 8.

Step 3: Set LASSO regression features as \( \text{eps} = 0.0001(\text{Length of the path}), n_{\text{alphas}} = 1(\text{number of alphas along the regularization path}) \) and normalization=True.

Step 4: Perform stacking by first using polynomial regression as the first stage model for each test project and using its similar projects.

Step 5: Output from polynomial regression is fed to the second stage model of stacking, LASSO regression with cross-validations = 10 to get the final effort for each test project.

Step 6: Output from LASSO regularization is treated as predicted effort from SR algorithm.

5 Dataset description and evaluation criteria

5.1 Dataset description

For the intent of SABE model evaluation, four datasets have been utilized by authors which are Cocomo81, NASA93, Maxwell and China datasets (http://promise.site.uottawa.ca/SERepository/datasets-page.html). The Cocomo81 dataset consists of 63 projects. Each project contains 15 cost drivers, the software size measured in KDSI (kilo delivered source instructions), the actual effort and the mode of development. The NASA93 dataset includes 93 NASA projects from different centers for various years. It comprises of 24 attributes containing 15 standard Cocomo-I attributes, 7 others describing the project; one lines of code measured in KLOC and the actual effort. The Maxwell dataset contains 62 software projects with 26 features each including software size, duration and effort. The China dataset consists of 499 projects with 19 features where 18 are independent variables and 1 is a dependent variable. These datasets are publicly available at http://openscience.us/repo/(2019). Table 2 exhibits the concise information and the explanatory statistics of the datasets.

5.2 Evaluation criteria

For the assessment and comparison of the precision of the analogy-based effort estimation model, the following prevalent evaluation criteria is employed. The magnitude of error correlated with estimated effort is kenned as absolute error \( AE \). It is the distinction between a certain project’s estimated effort and actual effort. This is represented in Eq.(14), where \( \alpha_i \) is the actual effort and \( \hat{\alpha}_i \) is predicted effort.

\[
AE_i = |\alpha_i - \hat{\alpha}_i| \tag{14}
\]

As seen in Eq.(15), magnitude of relative error (MRE) measures the total error percentage to actual effort. It can be evaluated by dividing absolute error by \( \alpha_i \) which is actual effort.

\[
MRE_i = \frac{AE_i}{\alpha_i} \tag{15}
\]

MMRE is mean of magnitude of relative error. For the \( n \) number of projects, it is the average of MRE. MMRE can be interpreted in mathematical terms as in Eq.(16), where \( n \) is number of projects.

\[
\text{MMRE} = \frac{1}{n} \sum_{i=1}^{n} MRE_i \tag{16}
\]

\( PRED(x) \), the output predictor, is the percentage of predictions that fell inside the actual kenned value \( x \), expressed in Eq.(18), where \( n \) is number of projects and \( D_i \) can be calculated by:

\[
D_i = \begin{cases} 
1 & \text{if } MRE_i \leq 0.25 \\
0 & \text{otherwise}
\end{cases} \tag{17}
\]

\[
PRED(x) = \frac{100}{n} \sum_{i=1}^{n} D_i \tag{18}
\]

MMRE and PRED are rendered on MRE. Because of their asymmetric distribution, the MRE-predicated quantifications have intrinsic drawbacks as performance metrics, biasing against presage model that they are. Therefore, MMRE and PRED are also underestimated prejudiced measures (Angelis and Stamelos 2000; Kadoda et al. 2000; Boehm 1984). Another measure was also published, named as the mean absolute error (MAE), which is not prejudiced and does not show asymmetric distribution as MMRE does. The MAE (Gergonne 1974) is measured using an average of absolute errors as shown in Eq.(19):

\[
\text{MAE} = \frac{1}{N} \sum_{i=1}^{N} AE_i \tag{19}
\]

MAE became challenging to decipher since its residuals are not standardized. Consequently, an incipient metric called standardized accuracy (SA) as given in Eq.(20) was stated by Shepperd and MacDonell (Shepperd and MacDonell 2012). The SA ratio is specifically used to assess how a predictive model outperforms a random guessing baseline and engenders accurate estimates. If not, the model of prediction cannot
be considered subsidiary. The SA value proximate to zero suggests poorer reliability, and a negative value is often deemed unsatisfactory.

\[
SA = 1 - \frac{MAR}{MAR_{P_0}}
\]  

where \(MAR\) (mean absolute residual) is calculated as:

\[
MAR = \frac{\sum_{i=1}^{m} |Actual\ Effort - Estimated\ Effort|}{m}
\]  

\(MAR\) is the MAR of the proposed technique and \(\overline{MAR}_{P_0}\) is the mean of MARs obtained through large runs of random guessing (Shepperd and MacDonell 2012). \(m\) is the total number of projects.

### 6 Experimental evaluation

The authors define in this segment the experimental setup that is utilized in the experiments. The authors have omitted the various consequences of the features as a preprocessing phase by normalizing in the interval [0, 1] utilizing the min–max normalization method (Kocaguneli and Menzies 2013). In the current study, leave-one-out (LOO) (Kocaguneli and Menzies 2013) is used for generating train and test datasets. The experiment with the proposed SABE framework is carried out using all the four datasets and all the evaluation criteria as discussed in Sect. 5.1 and Sect. 5.2, respectively. To further explain clearly the SABE framework, Project 2 from Cocomo dataset is selected with attributes 0.88, 1.16, 0.85, 1, 1.06, 1, 1.07, 1, 0.91, 1, 0.9, 0.95, 1.1, 1 and 1. These values are normalized. This project is the test project, and the rest of the projects are the historic projects. The similarity functions are then applied on the test project and the historic projects. The five best similar projects are retrieved. These projects then undergo two separate operations as explained in Step 4.1 and Step 4.2. The output obtained from Step 4.1 and Step 4.2 is then adapted using Eq. (13) to get the final estimated effort of the test project. The estimated effort and the actual effort are then subjected to performance metrics MMRE, MdMRE, PRED and SA. Tables 3, 4, 5 and 6 depict the results on the four datasets. On each dataset, all the four similarity functions are applied and the best \(k\)-nearest neighbors are retrieved; they are then submitted to solution functions and the SABE technique by varying \(k\) values from 1 to 5. The experimental findings are also used to obtain the most desirable ABE framework in terms of \(k\) value, similarity measure and solution function with all the performance metrics.

Apart from the results of SABE technique, Tables 3, 4, 5 and 6 also depict the results of CA, mean, IWM and median solution functions for all the four datasets.

By using the results of MMRE, MdMRE, PRED, and SA for Coomoco81, NASA93, Maxwell and China datasets, the best values are highlighted for each evaluation criteria. A prediction technique is considered better if it has low MRE, and MMRE values, and high PRED, and SA values. The highlighted train and test values from the Tables 3, 4, 5 and 6 show that the best solution function for all the datasets and similarity functions irrespective of the \(k\) values, is the proposed SABE framework. For example, from Table 3 for Cocomo dataset with Euclidean similarity, the best testing value for MMRE is found at \(k = 3\), for MdMRE it is at \(k = 5\), for PRED it is at \(k = 1\), and for SA it is at \(k = 5\) and all at SABE solution function.

Few of the results are depicted graphically also. Figures 3, 4, 5 and 6 depict the performance of all the solution functions at \(k = 3\) and at Euclidean similarity. The graphs for Cocomo, NASA and Maxwell datasets show the performance of SABE is at par over all the evaluation criteria. In China dataset, SABE is performing best for MMRE and MdMRE criteria, and there is less variation for PRED and SA criteria. But as depicted from the results of Table 6 on China dataset, it can be seen that SABE is outweighing PRED and SA criteria for Jaccard, Manhattan and Minkowski similarity functions.

The researchers Benala and Mall (Benala and Mall 2018) contributed various ABE models, and they concluded that the best ABE configuration is found at Euclidean similarity, with \(k = 3\) and mean solution function. The authors have compared the performance of SABE with Benala and Mall models (Tables 13-15) (Benala and Mall 2018) on Cocomo81, NASA93 and China datasets. The proposed approach is compared with the six models (ABE, GA-ABE,
### Table 3: Cocomo

| Similarity  | K | Solution | MMRE | MdMRE | PRED | SA  |
|-------------|---|----------|------|-------|------|-----|
|             |   |          | Train | Test  | Train | Test | Train | Test | Train | Test |
| Euclidean   | 1 | CA       | 1.6193 | 1.9748 | 0.7802 | 0.9256 | 0.0937 | 0.0645 | 0.3988 | 0.1026 |
|             |   | SABE     | 0.0677 | 0.078 | 0.0238 | 0.0659 | **1.00** | **1.00** | 0.992 | 0.9207 |
|             | 2 | Mean     | 1.2925 | 1.597 | 0.6392 | 0.9305 | 0.2812 | 0.0645 | 0.4626 | 0.1649 |
|             |   | IWM      | 1.3531 | 0.4012 | 0.7013 | 0.9275 | 0.2812 | 0.0967 | 0.4178 | 0.1462 |
|             |   | SABE     | **0.033** | 0.1012 | 0.0168 | 0.0798 | 0.9655 | 0.931 | 0.997 | 0.9709 |
|             | 3 | Mean     | 0.3392 | 0.4245 | 0.5997 | 0.3333 | 0.1937 | 0.2645 | 0.508 | 0.5659 |
|             |   | IWM      | 1.3995 | 0.5197 | 0.6022 | 0.3414 | 0.1625 | 0.1967 | 0.4921 | 0.4553 |
|             |   | Median   | 0.4795 | 0.162 | 0.6693 | 1.0566 | 0.1625 | 0.1967 | 0.4909 | 0.4662 |
|             |   | SABE     | 0.0568 | 0.0702 | **0.011** | 0.069 | 0.9677 | 0.9375 | **0.9974** | 0.9721 |
|             | 4 | Mean     | 1.2498 | 0.7595 | 0.6992 | 1.4673 | 0.125 | 0.0322 | 0.5105 | 0.1893 |
|             |   | IWM      | 1.2424 | 3.0035 | 0.7217 | 1.5394 | 0.125 | 0.0322 | 0.4848 | 0.1764 |
|             |   | Median   | 0.942  | 0.2573 | 0.7092 | 0.9764 | 0.1562 | 0.129 | 0.4099 | 0.2545 |
|             |   | SABE     | 0.0525 | 0.0828 | 0.0169 | 0.0647 | **1.00** | **1.00** | 0.9964 | 0.9681 |
|             | 5 | Mean     | 1.2365 | 2.4305 | 0.6195 | 1.5172 | 0.0937 | 0.129 | 0.0322 | 0.4675 |
|             |   | IWM      | 1.2329 | 2.4016 | 0.6625 | 1.6933 | 0.125 | 0.0322 | 0.4675 | 0.2167 |
|             |   | Median   | 0.942  | 0.2573 | 0.7092 | 0.9764 | 0.1562 | 0.129 | 0.4099 | 0.2545 |
|             |   | SABE     | 0.0515 | 0.0752 | 0.0157 | 0.0481 | 0.9677 | 0.9655 | **0.9974** | 0.9765 |

| Manhattan   | 1 | CA       | 1.4431 | 1.3877 | 0.7581 | 0.8048 | 0.125 | 0.129 | 0.4039 | 0.2626 |
|             |   | SABE     | 0.182 | 0.084 | 0.0175 | 0.069 | 0.9677 | 0.875 | **0.9975** | 0.9753 |
|             | 2 | Mean     | 1.0989 | 2.0539 | 0.6392 | 0.8125 | 0.25 | 0.0645 | 0.4627 | 0.1896 |
|             |   | IWM      | 1.1529 | 2.2902 | 0.6878 | 0.6432 | 0.2812 | 0.0645 | 0.4226 | 0.1702 |
|             |   | SABE     | **0.0686** | 0.0915 | 0.0168 | 0.0668 | 0.9032 | 0.9375 | 0.9968 | 0.9727 |
|             | 3 | Mean     | 1.1195 | 2.4378 | 0.5672 | 1.1047 | 0.0937 | 0.0645 | 0.5091 | 0.3701 |
|             |   | IWM      | 1.1305 | 2.6637 | 0.6057 | 1.0365 | 0.0937 | 0.0967 | 0.4928 | 0.2561 |
|             |   | Median   | 1.1561 | 1.9735 | 0.6913 | 0.8962 | 0.0937 | 0.0967 | 0.291 | 0.1768 |
|             |   | SABE     | 0.0699 | 0.0938 | 0.0168 | 0.0157 | **1.00** | 0.875 | 0.9974 | 0.9788 |
|             | 4 | Mean     | 1.1443 | 2.2563 | 0.6992 | 1.4019 | 0.125 | 0.0645 | 0.5167 | 0.4011 |
|             |   | IWM      | 1.1653 | 2.3663 | 0.7282 | 1.4677 | 0.1562 | 0.0322 | 0.4933 | 0.3937 |
|             |   | Median   | 0.9423 | 1.7527 | 0.6728 | 0.8143 | 0.125 | 0.129 | 0.4133 | 0.3576 |
Table 3 continued

| Similarity | K   | Solution | MMRE  | MdBRE | PRED  | SA    |
|------------|-----|----------|-------|-------|-------|-------|
|            |     |          | Train | Test  | Train | Test  | Train | Test  |
| SABE       | 0.0718 | 0.0951 | 0.0168 | 0.0657 | 0.9677 | 0.9062 | 0.9974 | 0.9649 |
| Mean       | 1.2601 | 2.3011 | 0.6971 | 1.0981 | 0.0625 | 0.0645 | 0.4886 | 0.3216 |
| Median     | 0.9574 | 1.6755 | 0.7649 | 0.7679 | 0.0312 | 0.3225 | 0.3067 | 0.3088 |
| SABE       | 0.0741 | 0.0973 | 0.0168 | 0.0657 | 0.9354 | 0.9062 | 0.9974 | 0.9799 |
| Minkowski  |     |          |       |       |       |       |       |       |
| CA         | 1.1693 | 2.0449 | 0.7802 | 0.9444 | 0.0937 | 0.0449 | 0.3988 | 0.1048 |
| Mean       | 1.2857 | 2.6298 | 0.6392 | 0.9764 | 0.2812 | 0.0967 | 0.4629 | 0.1681 |
| Median     | 1.1977 | 2.1619 | 0.6913 | 1.0566 | 0.0625 | 0.0967 | 0.2897 | 0.1661 |
| SABE       | 0.0772 | 0.0965 | 0.0168 | 0.0657 | 0.9354 | 0.9062 | 0.9973 | 0.9758 |
| CA         | 1.1817 | 2.0449 | 0.7802 | 0.9444 | 0.0937 | 0.0449 | 0.3988 | 0.1048 |
| Mean       | 1.2857 | 2.6298 | 0.6392 | 0.9764 | 0.2812 | 0.0967 | 0.4629 | 0.1681 |
| Median     | 1.1977 | 2.1619 | 0.6913 | 1.0566 | 0.0625 | 0.0967 | 0.2897 | 0.1661 |
| SABE       | 0.0772 | 0.0965 | 0.0168 | 0.0657 | 0.9354 | 0.9062 | 0.9973 | 0.9758 |
| CA         | 1.1817 | 2.0449 | 0.7802 | 0.9444 | 0.0937 | 0.0449 | 0.3988 | 0.1048 |
| Mean       | 1.2857 | 2.6298 | 0.6392 | 0.9764 | 0.2812 | 0.0967 | 0.4629 | 0.1681 |
| Median     | 1.1977 | 2.1619 | 0.6913 | 1.0566 | 0.0625 | 0.0967 | 0.2897 | 0.1661 |
| SABE       | 0.0772 | 0.0965 | 0.0168 | 0.0657 | 0.9354 | 0.9062 | 0.9973 | 0.9758 |
| CA         | 1.1817 | 2.0449 | 0.7802 | 0.9444 | 0.0937 | 0.0449 | 0.3988 | 0.1048 |
| Mean       | 1.2857 | 2.6298 | 0.6392 | 0.9764 | 0.2812 | 0.0967 | 0.4629 | 0.1681 |
| Median     | 1.1977 | 2.1619 | 0.6913 | 1.0566 | 0.0625 | 0.0967 | 0.2897 | 0.1661 |
| SABE       | 0.0772 | 0.0965 | 0.0168 | 0.0657 | 0.9354 | 0.9062 | 0.9973 | 0.9758 |
| CA         | 1.1817 | 2.0449 | 0.7802 | 0.9444 | 0.0937 | 0.0449 | 0.3988 | 0.1048 |
| Mean       | 1.2857 | 2.6298 | 0.6392 | 0.9764 | 0.2812 | 0.0967 | 0.4629 | 0.1681 |
| Median     | 1.1977 | 2.1619 | 0.6913 | 1.0566 | 0.0625 | 0.0967 | 0.2897 | 0.1661 |
| SABE       | 0.0772 | 0.0965 | 0.0168 | 0.0657 | 0.9354 | 0.9062 | 0.9973 | 0.9758 |

Bold values represent the best solution function for all the datasets and similarity functions irrespective of the K values.

DABE-3, PSO-ABE, SADE-ABE, JADE-ABE) from Benala and Mall’s work.

For Cocomo81 dataset, the best MMRE test value is given by SABE-ABE which is 0.016, whereas the best MMRE test value for SABE is 0.0137 at Jaccard similarity and at k=3. In PRED, the best value is given by DABE-3 which is 0.816, for SABE the best PRED value is 1.00 at k=1,4 and using Euclidean similarity. For MdMRE, it is given by PSO-ABE which is 0.021, whereas for SABE the best value is 0.0157 using Manhattan similarity and at k=3. The best SA value is given by DABE-3 which is 98.940, whereas using SABE it is 99.75 at k=4 using Jaccard similarity.

For NASA93 dataset, the best MMRE test value is given by GA-ABE which is 0.009, whereas the best MMRE test value for SABE is 0.0211 at Minkowski similarity and at k=5. In PRED, the best value is given by ABE which is 0.839, for SABE the best PRED value is 1.00 at k=4,5 and using Jaccard similarity. It is also found at k=4, with Manhattan similarity and at k=5 with Minkowski similarity. For MdMRE, it is given by GA-ABE which is 0.009, whereas for SABE the best value is 0.0089 using Euclidean similarity and at k=1. The best SA value is given by DABE-3 which is 94.234, whereas using SABE it is 99.22 at k=5 using Minkowski similarity.

For CHINA dataset, the best MMRE test value is given by PSO-ABE which is 0.010, whereas the best MMRE test value for SABE is 0.0116 at Jaccard similarity and at k=4. In PRED, the best value is given by ABE which is 0.868, for SABE the best PRED value is 1.00 at k=1,2,3,4,5 by using Euclidean, Manhattan and Minkowski similarity measures. For MdMRE, it is given by DABE-3 which is 0.039, whereas for SABE the best value is 0.0127 using Manhattan similarity and at k=3. The best SA value is given by DABE-3 which is 96.509, whereas using SABE it is 99.62 at k=5 using Manhattan similarity.

All the analyses performed above concluded that SABE is providing the best results for mostly all the models and evaluation criteria except for GA-ABE at MMRE for NASA93 dataset, and for PSO-ABE at MMRE for China dataset. But the best performance of SABE is not restricted to a single configuration. It is giving good results at different similarity measures and at different k values. This is further confirmed using statistical analysis.
## Table 4 NASA

| Similarity | $k$ | Solution | MMRE | MdMRE | PRED | SA |
|------------|-----|----------|------|-------|------|----|
|            |     |          | Train | Test  | Train | Test | Train | Test |
| Euclidean  | 1   | CA       | 0.5972 | 2.6799 | 0.2887 | 0.5821 | 0.234 | 0.1086 | 0.5699 | 0.522 |
|            |     | SABE     | 0.0713 | 0.0362 | 0.0445 | 0.0089 | 0.0978 | 0.9565 | 0.9604 | **0.9901** |
|            | 2   | Mean     | 0.3573 | 0.9007 | 0.6   | 0.7632 | 0.2765 | 0.1521 | 0.4376 | 0.6093 |
|            |     | IWM      | 1.5157 | 1.5746 | 0.5917 | 0.7656 | 0.2404 | 0.1304 | 0.4875 | 0.6146 |
|            |     | SABE     | 0.0857 | 0.0708 | 0.0609 | 0.0163 | 0.9545 | 0.9574 | 0.9801 | 0.991 |
|            | 3   | Mean     | 0.2687 | 0.617  | 0.6716 | 0.7586 | 0.2553 | 0.2304 | 0.7015 | 0.7793 |
|            |     | IWM      | 0.3396 | 0.3001 | 0.6423 | 0.7146 | 0.1702 | 0.1904 | 0.6866 | 0.736 |
|            |     | Median   | 0.2573 | 0.0359 | 0.3243 | 0.5664 | 0.3404 | 0.3086 | 0.776  | 0.7786 |
|            |     | SABE     | **0.0211** | **0.0234** | 0.072 | 0.0271 | 0.9328 | **0.9787** | 0.9778 | 0.99 |
| Jaccard    | 1   | CA       | 29.6726 | 4.697  | 10.4795 | 0.8928 | 0.0212 | 0.0652 | 0.0439 | 0.0189 |
|            |     | SABE     | 0.1302 | 0.0503 | 0.0638 | 0.0312 | 0.8297 | 0.9565 | 0.9776 | 0.9886 |
|            | 2   | Mean     | 18.7283 | 9.8978 | 6.7806 | 0.9121 | 0.0851 | 0.0652 | 0.0074 | 0.0795 |
|            |     | IWM      | 15.0173 | 10.4523 | 6.5494 | 0.9317 | 0.0851 | 0.0869 | 0.0181 | 0.0783 |
|            |     | SABE     | 0.1165 | 0.0508 | 0.0588 | 0.0277 | 0.851  | 0.9565 | 0.9764 | 0.9895 |
|            | 3   | Mean     | 14.423  | 8.364  | 5.6034 | 0.8813 | 0.0851 | 0.0869 | 0.0078 | 0.0668 |
|            |     | IWM      | 12.0727 | 8.6593 | 5.6034 | 0.8821 | 0.1489 | 0.0652 | 0.0304 | 0.0678 |
|            |     | Median   | 10.6561 | 6.6932 | 3.9236 | 0.8353 | 0.0851 | 0.1086 | 0.0332 | 0.0154 |
|            |     | SABE     | 0.0858 | 0.0599 | 0.044  | **0.044** | 0.017 | 0.9148 | 0.9565 | 0.9826 | 0.9909 |
| Manhattan  | 1   | CA       | 13.7776 | 6.8011 | 5.289  | 0.8902 | 0.1489 | 0.1304 | 0.02   | 0.0514 |
|            |     | SABE     | 12.3934 | 6.9039 | 4.5456 | 0.8949 | 0.1702 | 0.1086 | 0.0194 | 0.0519 |
|            | 2   | Mean     | 18.7283 | 9.8978 | 6.7806 | 0.9121 | 0.0851 | 0.0652 | 0.0074 | 0.0795 |
|            |     | IWM      | 15.0173 | 10.4523 | 6.5494 | 0.9317 | 0.0851 | 0.0869 | 0.0181 | 0.0783 |
|            |     | Median   | 10.6561 | 6.6932 | 3.9236 | 0.8353 | 0.0851 | 0.1086 | 0.0332 | 0.0154 |
|            |     | SABE     | 0.0852 | 0.0592 | 0.0529 | 0.0182 | **0.9361** | **0.00** | **0.9829** | **0.9912** |
|            | 3   | Mean     | 20.5229 | 9.2866 | 7.2791 | 0.8078 | 0.1489 | 0.1521 | 0.0213 | 0.0336 |
|            |     | IWM      | 21.0114 | 9.8992 | 6.8732 | 0.6773 | 0.1489 | 0.1956 | 0.0314 | 0.0465 |
|            |     | Median   | 9.2402  | 6.775  | 1.2291 | 0.9027 | 0.0851 | 0.0434 | 0.0554 | 0.0413 |
|            |     | SABE     | **0.0852** | **0.0284** | 0.0529 | 0.0182 | **0.9361** | **0.00** | **0.9829** | **0.9912** |

Note: MDRE stands for Mean Deviation Relative Error, and SA stands for the correlation coefficient.
Table 4 continued

| Similarity | k  | Solution | MMRE | MdMRE | PRED | SA |
|------------|----|----------|------|-------|------|----|
|            |    |          | Train| Test  | Train| Test| Train| Test|
| Median     |    |          | 0.4438| 0.3513| 0.5579| 0.4404| 0.4521| 0.6074| 0.5926|
| SABE       |    | 0.0421   | 0.9921| 0.0577| 0.0254| 0.9148| 1.00  | 0.9804| 0.9903|
| k=5 Mean   |    | 0.9469   | 1.5989| 0.5968| 0.6913| 0.234 | 0.1956| 0.4912| 0.442 |
| IWM        |    | 0.4721   | 1.0034| 0.3461| 0.5914| 0.2978| 0.1739| 0.4314| 0.4516|
| Median     |    | 1.0678   | 1.4628| 0.5311| 0.6227| 0.234 | 0.2391| 0.5472| 0.5914|
| SABE       |    | 0.0853   | 0.0667| 0.0496| 0.0218| 0.9361| 0.9782| 0.9827| 0.9912|
| Minkowski  | k=1| CA       | 1.4365| 2.7628| 0.5   | 0.6912| 0.234 | 0.1086| 0.4464| 0.0924|
|            |    | SABE     | 0.1188| 0.0756| 0.0605| 0.0248| 0.8297| 0.9565| 0.9783| 0.9893|
| k=2 Mean   |    | 1.3923   | 1.901 | 0.6011| 0.7632| 0.234 | 0.1304| 0.4353| 0.107 |
| IWM        |    | 1.4897   | 1.5349| 0.5901| 0.7674| 0.2978| 0.1086| 0.3909| 0.216 |
| SABE       |    | 0.1264   | 0.0809| 0.0785| 0.0292| 0.851 | 0.9347| 0.9776| 0.9881|
| k=3 Mean   |    | 1.355    | 1.6207| 0.7724| 0.7694| 0.234 | 0.1304| 0.4936| 0.5666|
| IWM        |    | 1.348    | 1.2811| 0.6137| 0.7131| 0.1489| 0.1304| 0.4909| 0.4247|
| Median     |    | 0.4566   | 1.0474| 0.3333| 0.5664| 0.3191| 0.1304| 0.56  | 0.58  |
| SABE       |    | 0.118    | 0.0406| 0.0685| 0.0234| 0.8723| 0.9782| 0.9793| 0.9904|
| k=4 Mean   |    | 1.5893   | 1.7759| 0.7465| 0.7404| 0.234 | 0.1086| 0.5346| 0.2693|
| IWM        |    | 1.8293   | 1.6313| 0.6651| 0.5863| 0.2765| 0.2608| 0.546  | 0.2426|
| Median     |    | 0.5166   | 1.0221| 0.4763| 0.5579| 0.2978| 0.1304| 0.5986| 0.18  |
| SABE       |    | 0.0909   | 0.0505| 0.0594| 0.0279| 0.9361| 0.9565| 0.9803| 0.9896|
| k=5 Mean   |    | 1.4761   | 1.5942| 0.956 | 0.6715| 0.1702| 0.1739| 0.458  | 0.235 |
| IWM        |    | 1.5173   | 1.4125| 0.7425| 0.5625| 0.2765| 0.2391| 0.4245| 0.2626|
| Median     |    | 0.4885   | 0.9804| 0.3888| 0.657 | 0.3191| 0.1521| 0.5466| 0.1901|
| SABE       |    | 0.0824   | 0.0211| 0.0381| 0.0117| 0.8936| 1.00 | 0.9862 | 0.9922|

Bold values represent the best solution function for all the datasets and similarity functions irrespective of the K values

7 Statistical analysis

Statistical analysis helps us to find the appropriateness of one model to another (Kitchenham and Mendes 2009), (Mitas and Angelis 2008). From the discussion in Sect. 6, it is evident that the SABE solution function is providing the best results but now using statistical analysis this will be further confirmed. Also it will help to find the best similarity function and k value to be chosen for SABE. The datasets for software cost estimation studies do not follow any particular distribution so nonparametric statistical tests are used (Kaushik et al. 2016). All these tests are performed on KEEL (Knowledge Extraction based on Evolutionary Learning) tool (Alcalá-Fdez et al. 2011), and the statistical analysis is done on MMRE metric.

Initially, the authors compared all the solution functions using Friedman test (Hodges and Lehmann 2012). This test computes the statistical differences among the solution functions. It provides the lowest rank to the best solution function. The Friedman test is computed by:

\[
FT = \frac{(q - 1) \left[ \sum_{j=1}^{q} \hat{M}_j^2 - \left( \frac{qn^2}{2} \right) (qn + 1)^2 \right]}{\left[ \left( qn(n+1)(2qn+1) \right) / 6 \right] - (1/q) \sum_{i=1}^{n} \hat{M}_i^2}
\]

where \( q \) is the total number of techniques, \( \hat{M}_i \) is the total rank of the \( i^{th} \) data-set, and \( \hat{M}_j \) is the total rank of the \( j^{th} \) technique. This test is conducted for all the solution functions at \( k=3 \) with Euclidean similarity on all the four datasets. The authors have chosen \( k=3 \), for the Friedman test as it is considered as the optimal value and it is covering all the solution functions (Benala and Mall 2018). The null hypothesis assumed that all the solution functions performed equally. The results of the Friedman test is given in Table 7.

Here, \( N \) is the number of input datasets, the degree of freedom \((df)\) is 3 as there are four solution functions to compare. The standard \( \chi^2 \) value with 3 \( df \) and significance value \( \alpha = 0.05 \) is 7.815. The null hypothesis is rejected as the \( \chi^2 \) value listed in the test statistic (Table 7) is more than 7.815 and the \( p-value \) is less than 0.05. So, it is deduced that all the solution functions are different. Fig. 7 shows the average
Table 5  Maxwell

| Similarity | k | Solution | MMRE | MdMRE | PRED | SA |
|------------|---|----------|------|-------|------|----|
|            |   |          | Train | Test  | Train | Test |
| Euclidean  | k=1| CA       | 0.094| 0.0599| 0.0574| 0.0615| 0.2258| 0.2666| 0.5784| 0.5537|
|            |   | SABE     | 0.0042| 0.0028| 0.0026| 0.0014| 1.00  | 1.00  | 0.999  | 0.9984|
|            | k=2| Mean     | 0.0807| 0.053 | 0.0568| 0.0556| 0.129 | 0.3   | 0.6997 | 0.7692|
|            |   | IWM      | 0.0812| 0.0549| 0.0567| 0.0406| 0.1935| 0.3   | 0.6832 | 0.7172|
|            |   | SABE     | 0.0037| 0.0029| 0.0017| 0.0013| 1.00  | 1.00  | 0.9991 | 0.9983|
|            | k=3| Mean     | 0.0715| 0.0353| 0.0501| 0.0362| 0.329 | 0.3666| 0.822  | 0.9782|
|            |   | IWM      | 0.0856| 0.0555| 0.0566| 0.0432| 0.129 | 0.2333| 0.8214 | 0.9241|
|            |   | Median   | 0.0743| 0.046 | 0.0503| 0.0412| 0.2058| 0.2   | 0.8476 | 0.9434|
|            |   | SABE     | 0.0047| 0.0029| 0.0026| 0.002 | 1.00  | 1.00  | 0.9989 | 0.9983|
| Jaccard    | k=1| CA       | 0.1552| 1.8261| 0.8122| 0.7825| 0.1935| 0.1333| 0.3107 | 0.4047|
|            |   | SABE     | 0.0054| 0.0031| 0.0029| 0.0018| 1.00  | 1.00  | 0.9989 | 0.9982|
|            | k=2| Mean     | 2.6104| 2.1184| 0.9081| 0.7066| 0.0645| 0.1   | 0.0904 | 0.1966|
|            |   | IWM      | 2.6207| 2.1183| 0.9081| 0.7066| 0.0645| 0.1   | 0.0923 | 0.1965|
|            |   | SABE     | 0.0057| 0.0031| 0.0037| 0.0021| 1.00  | 1.00  | 0.9989 | 0.9982|
|            | k=3| Mean     | 3.0696| 2.018 | 0.9311| 0.7868| 0.0645| 0.1   | 0.0904 | 0.1966|
|            |   | IWM      | 3.1044| 2.0283| 0.9311| 0.7869| 0.0645| 0.1   | 0.0923 | 0.1965|
|            |   | Median   | 2.707 | 2.0612| 0.9634| 0.811 | 0.1935| 0.0666| 0.1631 | 0.2112|
|            |   | SABE     | 0.0051| 0.0035| 0.002 | 0.0021| 1.00  | 1.00  | 0.9989 | 0.9981|
| Manhattan  | k=1| CA       | 0.0838| 0.0575| 0.0475| 0.0553| 0.2258| 0.3333| 0.451  | 0.4709|
|            |   | SABE     | 0.005| 0.003 | 0.0022| 0.0017| 1.00  | 1.00  | 0.999  | 0.9981|
|            | k=2| Mean     | 0.0798| 0.051 | 0.0544| 0.0485| 0.1612| 0.3733| 0.4066 | 0.5757|
|            |   | IWM      | 0.0826| 0.0544| 0.0549| 0.0461| 0.2258| 0.2666| 0.387  | 0.5137|
|            |   | SABE     | 0.0046| 0.0036| 0.0021| 0.0016| 1.00  | 1.00  | 0.999  | 0.9982|
|            | k=3| Mean     | 0.0749| 0.0495| 0.0477| 0.0442| 0.129 | 0.2   | 0.4347 | 0.5403|
|            |   | IWM      | 0.0755| 0.0499| 0.0489| 0.0455| 0.129 | 0.2   | 0.4347 | 0.5403|
|            |   | Median   | 0.066 | 0.0441| 0.0494| 0.0412| 0.1612| 0.2666| 0.4586 | 0.5798|
|            |   | SABE     | 0.0039| 0.0026| 0.0021| 0.0013| 1.00  | 1.00  | 0.9991 | 0.9984|
|            | k=4| Mean     | 0.0709| 0.0447| 0.0541| 0.0419| 0.2258| 0.2666| 0.4682 | 0.5806|
|            |   | IWM      | 0.0696| 0.0432| 0.0553| 0.0346| 0.2258| 0.3333| 0.489  | 0.5486|
| Similarity | k | Solution | MMRE |  | MdMRE |  | PRED |  | SA |  |
|-----------|---|----------|------|---|------|---|------|---|-----|---|
|           |   |          | Train| Test| Train| Test| Train| Test| Train| Test|
| Median    |   |          | 0.065| 0.0414| 0.0559| 0.0329| 0.2258| 0.3666| 0.4386| 0.5636|
| SABE      |   |          | 0.0034| 0.0028| **0.0019**| 0.0017| **1.00**| **1.00**| 0.999| 0.9983|
| k=5       |   | Mean     | 0.0745| 0.0441| 0.05| 0.0357| 0.1612| 0.3333| 0.4772| 0.605|
|           |   | IWM      | 0.077| 0.0449| 0.0521| 0.0342| 0.129| 0.3| 0.4829| 0.5754|
|           |   | Median   | 0.0713| 0.0417| 0.0503| 0.0366| 0.2258| 0.2666| 0.3861| 0.5091|
|           |   | SABE     | 0.005| 0.0032| 0.0021| 0.0015| **1.00**| **1.00**| 0.9989| 0.9983|
| Minkowski |   | CA       | 0.0945| 0.066| 0.0574| 0.0626| 0.2258| 0.2666| 0.3777| 0.4264|
| k=1       |   | SABE     | **0.004**| 0.003| **0.0015**| 0.0018| **1.00**| **1.00**| **0.9991**| 0.9982|
|           |   | k=2 Mean | 0.0837| 0.0516| 0.058| 0.048| 0.1612| 0.3333| 0.3802| 0.5783|
|           |   | IWM      | 0.0827| 0.053| 0.0573| 0.0365| 0.1935| 0.3666| 0.3698| 0.5302|
|           |   | SABE     | 0.0044| 0.0031| 0.0021| 0.0015| **1.00**| **1.00**| 0.9989| 0.9983|
|           |   | k=3 Mean | 0.0831| 0.0553| 0.0589| 0.0442| 0.0967| 0.2| 0.415| 0.5446|
|           |   | IWM      | 0.0874| 0.0555| 0.0577| 0.0418| 0.0967| 0.2333| 0.4154| 0.5192|
|           |   | Median   | 0.0745| 0.0461| 0.0503| 0.0412| 0.2258| 0.2666| 0.4471| 0.5727|
|           |   | SABE     | 0.004| 0.0031| 0.0021| 0.0023| **1.00**| **1.00**| 0.9989| 0.9981|
|           |   | k=4 Mean | 0.0771| 0.0478| 0.0541| 0.0419| 0.129| 0.2666| 0.4357| 0.578|
|           |   | IWM      | 0.0754| 0.0473| 0.0552| 0.0343| 0.129| 0.3| 0.4605| 0.5437|
|           |   | Median   | 0.0734| 0.0418| 0.0562| 0.0309| 0.1935| 0.3333| 0.3974| 0.5688|
|           |   | SABE     | 0.0044| **0.0025**| 0.0018| **0.0013**| **1.00**| **1.00**| **0.9991**| **0.9984**|
|           |   | k=5 Mean | 0.0756| 0.0461| 0.057| 0.0369| 0.0967| 0.3333| 0.4609| 0.5862|
|           |   | IWM      | 0.0788| 0.0486| 0.0586| 0.0314| 0.0645| 0.4| 0.4618| 0.5455|
|           |   | Median   | 0.0729| 0.0415| 0.0543| 0.0397| 0.1935| 0.3| 0.3875| 0.5242|
|           |   | SABE     | 0.0042| 0.0034| 0.0021| 0.0014| **1.00**| **1.00**| 0.999| 0.9982|

Bold values represent the best solution function for all the datasets and similarity functions irrespective of the K values.

As per the Holm test statistics, the hypothesis is rejected for mean and IWM but it is accepted for Median. This shows that there is not much significant difference between SABE and the median solution function. Next, Holm post hoc test (Holm 1979) is conducted to find the differences among the techniques with SABE as the control method. Its test statistics is given in Table 8.

Here, N is the number of input datasets, the degree of freedom (df) is 3 as there are four similarity functions to compare. The null hypothesis is accepted as the χ² value listed in the test statistic (Table 10) is less than 7.815 and the p-value is more than 0.05. So, it is deduced that all the similarity functions performed equally and there are no significant differences.

Figure 8 shows the average ranks of all the similarity functions as given by Friedman rank test. It shows that the Minkowski similarity function has the lowest rank of 1.875, so this similarity function is taken as the control method. The statistical analysis is further performed to find the best k value for SABE. The results considered for analysis are MMRE test values for all the k values with Minkowski similarity function and SABE solution function. The Friedman test results to know which is the best similarity function and the best k value for SABE. In order to find the best similarity function, the authors have chosen the best MMRE test values for all the datasets. First, the Friedman test is performed. The null hypothesis assumed that all the similarity functions performed equally. The results of Friedman test for similarity function are given in Table 10.

Here, N is the number of input datasets, the degree of freedom (df) is 3 as there are four similarity functions to compare. The null hypothesis is accepted as the χ² value listed in the test statistic (Table 10) is less than 7.815 and the p-value is more than 0.05. So, it is deduced that all the similarity functions performed equally and there are no significant differences.

Figure 8 shows the average ranks of all the similarity functions as given by Friedman rank test. It shows that the Minkowski similarity function has the lowest rank of 1.875, so this similarity function is taken as the control method. The statistical analysis is further performed to find the best k value for SABE. The results considered for analysis are MMRE test values for all the k values with Minkowski similarity function and SABE solution function. The Friedman test results
Table 6  China

| Similarity  | k   | Solution | MMRE  | MdMRE  | PRED  | SA     |
|-------------|-----|----------|-------|--------|-------|--------|
|             |     |          | Train | Test   | Train | Test   |
| Euclidean   | k=1 | CA       | 0.1127| 0.1143 | 0.0575| 0.0566 |
|             |     | SABE     | **0.0216** | **0.0275** | **0.0116** | **0.013** | **1.00** | **1.00** |
|             |     | Mean     | 0.1169| 0.1078 | 0.0772| 0.0823 |
|             |     | IWM      | 0.1195| 0.1106 | 0.0813| 0.0846 |
|             |     | SABE     | 0.0226| 0.0279 | 0.0119| 0.0133 |
|             |     | Mean     | 0.1115| 0.0907 | 0.0768| 0.0524 |
|             | k=2 | IWM      | 0.1137| 0.1108 | 0.0761| 0.0828 |
|             |     | Median   | 0.1161| 0.0956 | 0.063 | 0.0606 |
|             |     | SABE     | 0.0226| 0.0278 | 0.0126| 0.0133 |
|             |     | Mean     | 0.1196| 0.1058 | 0.0713| 0.0849 |
|             |     | IWM      | 0.1123| 0.1083 | 0.0733| 0.0855 |
|             |     | Median   | 0.1119| 0.1002 | 0.0711| 0.0744 |
|             |     | SABE     | 0.0223| 0.0282 | 0.0127| 0.0133 |
|             | k=3 | Mean     | 0.1096| 0.1061 | 0.0721| 0.0854 |
|             |     | IWM      | 0.1133| 0.1091 | 0.0787| 0.0868 |
|             |     | Median   | 0.1128| 0.102  | 0.0642| 0.0725 |
|             |     | SABE     | 0.0219| 0.0279 | **0.0109** | **0.013** | **1.00** | **1.00** | **0.9934** | **0.9959** |
| Jaccard     | k=1 | CA       | 7.1901| 5.33   | 1.7325| 1.3428 |
|             |     | SABE     | 0.0912| 0.149  | 0.0348| 0.0344 |
|             | k=2 | Mean     | 2.6104| 2.1184 | 0.9081| 0.7066 |
|             |     | IWM      | 2.6207| 2.1183 | 0.9081| 0.7066 |
|             |     | Median   | 0.1125| 0.1604 | 0.0357| 0.046 |
|             |     | SABE     | 0.0223| 0.0282 | 0.0127| 0.0133 |
|             | k=3 | Mean     | 3.0696| 2.018  | 0.9311| 0.7868 |
|             |     | IWM      | 3.1044| 2.0283 | 0.9311| 0.7869 |
|             |     | Median   | 2.707 | 2.0612 | 0.9634| 0.811 |
|             |     | SABE     | 0.0697| 0.1266 | 0.0332| 0.0356 |
|             | k=4 | Mean     | 2.7083| 1.9209 | 0.9356| 0.6958 |
|             |     | IWM      | 2.7211| 1.9285 | 0.9356| 0.7165 |
|             |     | Median   | 2.3147| 1.9227 | 0.9253| 0.7458 |
|             |     | SABE     | 0.01126| **0.01161** | **0.0408** | **0.0302** | **0.86** | **0.9633** | **0.9812** |
| Manhattan   | k=1 | CA       | 0.126 | 0.109  | 0.0602| 0.0571 |
|             |     | SABE     | 0.022 | 0.0278 | **0.011** | **0.013** | **1.00** | **1.00** | **0.9934** | **0.996** |
|             | k=2 | Mean     | 0.1159| 0.1101 | 0.0758| 0.0814 |
|             |     | IWM      | 0.1184| 0.1135 | 0.0808| 0.0863 |
|             |     | SABE     | 0.0227| **0.0274** | **0.0125** | **0.0132** | **1.00** | **1.00** | **0.9931** | **0.9961** |
|             | k=3 | Mean     | 0.1125| 0.1062 | 0.077 | 0.0855 |
|             |     | IWM      | 0.1153| 0.1088 | 0.0771| 0.082 |
|             |     | Median   | 0.1158| 0.1038 | 0.0633| 0.0627 |
|             |     | SABE     | 0.0221| 0.0277 | 0.0116| **0.0127** | **1.00** | **1.00** | **0.9932** | **0.996** |
|             | k=4 | Mean     | 0.1094| 0.1118 | 0.0733| 0.0875 |
|             |     | IWM      | 0.1117| 0.1161 | 0.0761| 0.0913 |
|             |     | Median   | 0.1112| 0.1051 | 0.0765| 0.084 |
Table 6 continued

| Similarity | k   | Solution | MMRE Train | MMRE Test | MdMRE Train | MdMRE Test | PRED Train | PRED Test | SA Train | SA Test |
|------------|-----|----------|------------|-----------|-------------|------------|------------|-----------|-----------|---------|
|            |     | SABE     | 0.0221     | 0.0278    | 0.0118      | 0.0133     | **1.00**   | **1.00**  | 0.9934    | 0.996   |
| Minkowski  | k=5 | Mean     | 0.1079     | 0.1097    | 0.0683      | 0.0867     | 0.9641     | 0.9596    | 0.9213    | 0.9056  |
|            |     | IWM      | 0.1106     | 0.1133    | 0.0722      | 0.0865     | 0.9641     | 0.9596    | 0.9197    | 0.9054  |
|            |     | Median    | 0.1077     | 0.1086    | 0.0603      | 0.0741     | 0.9641     | 0.9677    | 0.9212    | 0.9043  |
|            |     | SABE     | **0.0202** | 0.0278    | 0.0111      | 0.0132     | **1.00**   | **1.00**  | 0.9937    | **0.9962** |
|            |     | k=1      | CA         | 0.1144    | 0.124       | 0.0599     | 0.0697     | 0.9362    | 0.9233    | 0.9131  | 0.8998  |
|            |     | SABE     | 0.0227     | 0.0279    | **0.0111**  | **0.0132** | **1.00**   | **1.00**  | 0.9933    | **0.9961** |
|            | k=2 | CA         | 0.117      | 0.1116    | 0.0835      | 0.0832     | 0.9561     | 0.9516    | 0.9168    | 0.9037  |
|            |     | IWM      | 0.12       | 0.1142    | 0.0808      | 0.0856     | 0.9402     | 0.9516    | 0.9175    | 0.9013  |
|            |     | SABE     | 0.0224     | 0.0279    | 0.0119      | 0.0141     | **1.00**   | **1.00**  | 0.9935    | 0.996   |
|            | k=3 | Mean     | 0.1121     | 0.1054    | 0.0741      | 0.0756     | 0.9521     | 0.9596    | 0.9186    | 0.9084  |
|            |     | IWM      | 0.1142     | 0.1077    | 0.0746      | 0.0789     | 0.9402     | 0.9556    | 0.9179    | 0.9092  |
|            |     | Median    | 0.1148     | 0.1068    | 0.0603      | 0.0576     | 0.9362     | 0.9596    | 0.9161    | 0.9074  |
|            |     | SABE     | 0.0222     | 0.0275    | **0.0113**  | **0.013**  | **1.00**   | **1.00**  | 0.9935    | **0.9961** |
|            | k=4 | Mean     | 0.1138     | 0.1046    | 0.074       | 0.0788     | 0.9681     | 0.9637    | 0.9186    | 0.9394  |
|            |     | IWM      | 0.1163     | 0.107     | 0.0755      | 0.0816     | 0.9641     | 0.9596    | 0.9176    | 0.9105  |
|            |     | Median    | 0.1123     | 0.1034    | 0.0704      | 0.0803     | 0.9681     | 0.9596    | 0.9191    | 0.9105  |
|            |     | SABE     | **0.0217** | 0.028     | 0.0112      | 0.0138     | **1.00**   | **1.00**  | 0.9936    | **0.9959** |
|            | k=5 | Mean     | 0.1142     | 0.1037    | 0.0778      | 0.0811     | 0.9641     | 0.9596    | 0.9255    | 0.9139  |
|            |     | IWM      | 0.1179     | 0.1064    | 0.0804      | 0.082      | 0.9641     | 0.9596    | 0.927     | 0.9144  |
|            |     | Median    | 0.1145     | 0.1049    | 0.066       | 0.0719     | 0.9681     | 0.9556    | 0.9192    | 0.9074  |
|            |     | SABE     | 0.0228     | 0.0279    | 0.0117      | 0.013      | **1.00**   | **1.00**  | 0.9934    | 0.9959  |

Bold values represent the best solution function for all the datasets and similarity functions irrespective of the K values.

Fig. 3 Comparison of solution functions with Cocomo dataset.
for this are given in Table 11. The hypothesis assumed here is all the $k$ values performed equally. The degree of freedom (df) is 4 corresponding to 5 $k$ values. The null hypothesis is accepted again as the $\chi^2$ value listed in the test statistic (Table 11) is less than 9.488 and the p-value is more than 0.05. So, it is deduced that all the $k$ values performed equally and there are no significant differences. The ranks as calculated by Friedman’s test is depicted in Fig. 9, and it shows that $k=5$ has the lowest rank of 2.5. So, $k=5$ can be considered as the best value with Minkowski similarity function and SABE solution function.

8 Conclusion

In this work, Stacking regularization in analogy-based software effort estimation (SABE) is proposed for software projects. The technique SABE is a solution function which
is evaluated alongside other solution functions like the closest analogy (CA), mean, median and the inverse distance-weighted mean (IWM). The technique is tested on four software estimation datasets, i.e., Cocomo, NASA, Maxwell and China. It is found that the SABE solution function provided the best results both in experimental evaluations as well as in statistical validations. The four similarity functions used are Euclidean, Jaccard, Manhattan and Minkowski. The k most proximate neighbors from 1 to 5 are considered in implementing the technique. The results of SABE are also compared with other techniques. It is found that SABE worked best maximum number of times for the evaluation criteria in comparison with models proposed by the earlier studies. The research has its own limitations too. The SABE technique uses stacking, which comes with its own price. Stacking involves training multiple base-models to predict the target variable. So, the stacked models take longer time to train than simpler models and require more memory. The choice of the base models used can also affect the results of the proposed technique. The base models used in the study are polynomial regression and lasso regression. The future work can include replacing these models with other base models which will take less time to train and provide more

Table 8 Holm test statistics

| Solution Functions   | Z      | Holm | Hypothesis (α=0.05) |
|----------------------|--------|------|---------------------|
| SABE vs Mean         | 1.9170 | 0.025| Rejected            |
| SABE vs IWM          | 3.0124 | 0.0166| Rejected            |
| SABE vs Median       | 1.6431 | 0.05 | Accepted            |

Table 9 Wilcoxon signed rank test statistics for SABE vs. median

| Solution Functions   | Rank Positive (R+) | Rank Negative (R-) | Hypothesis (α=0.05) | p-value   |
|----------------------|--------------------|--------------------|---------------------|-----------|
| SABE vs Median       | 10.0               | 0.0                | Rejected            | 0.04461   |

Table 7 Friedman rank test statistics for solution function

|                |        |        |
|----------------|--------|--------|
| N              | 4      |        |
| \( \chi^2 \)  | 9.3    |        |
| df             | 3      |        |
| \( p-value \)  | 0.0255 |        |

Table 10 Friedman rank test statistics for similarity function

|                |        |        |
|----------------|--------|--------|
| N              | 4      |        |
| \( \chi^2 \)  | 2.775  |        |
| df             | 3      |        |
| \( p-value \)  | 0.4276 |        |
Stacking regularization in analogy-based software effort estimation

Fig. 7 Average ranking of Friedman test for solution functions

Fig. 8 Average ranking of Friedman test for similarity functions

Fig. 9 Average ranking of Friedman test for \( k \) values

Table 11 Friedman rank test statistics for \( k \) values

|  |  |
|---|---|
| \( \chi^2 \) | 0.95 |
| df | 4 |
| \( p – value \) | 0.9172 |

efficient results. These models can be chosen from the current state-of-the-art machine learning approaches.

Author Contributions All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Dr. Anupama Kaushik, Dr. Prabhjot Kaur, Ms. Nisha Chaudhary and Ms. Priyanka Aggarwal. The first draft of the manuscript was written by Dr. Anupama Kaushik, and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Declarations

Conflict of interest All the authors declare that there is no conflict of interest in publishing this paper.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

Aamodt A, Plaza E (1994) Case-based reasoning: foundational issues, methodological variations, and system approaches. AI Comm 7(1):39–59
Abualigah L, Diabat A, Sumari P, Gandomi AH (2021) A novel evolutionary arithmetic optimization algorithm for multilevel thresholding segmentation of covid-19 ct images. Processes 9(7):1155
Abualigah LM, Khader AT (2017) Unsupervised text feature selection technique based on hybrid particle swarm optimization algorithm with genetic operators for the text clustering. J Supercomput 73(11):4773–4795
Abualigah LMQ, Hanandeh ES (2015) Applying genetic algorithms to information retrieval using vector space model. Int J Comput Sci Eng Appl 5(1):19
Abualigah LMQ et al (2019) Feature selection and enhanced krill herd algorithm for text document clustering. Springer, Newyork
Alcalá-Fdez J, Fernández A, Luengo J, Derrac J, García S, Sánchez L, Herrera F (2011) Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. J Multiple-Val Log Soft Comput 17
Anandhi V, Chezian RM (2014) Regression techniques in software effort estimation using cocomo dataset. In: 2014 international conference on intelligent computing applications, IEEE, pp 353–357
Angelis L, Stamelos I (2000) A simulation tool for efficient analogy based cost estimation. Emp Softw Eng 5(1):35–68
Azzeh M (2011) Model tree based adaption strategy for software effort estimation by analogy. In: 2011 IEEE 11th International Conference on Computer and Information Technology, IEEE, pp 328–335
Azzeh M, Nassif AB, Minku LL (2015) An empirical evaluation of ensemble adjustment methods for analogy-based effort estimation. J Syst Softw 103:36–52
Benala TR, Mall R (2018) Dabe: Differential evolution in analogy-based software development effort estimation. Swarm Evol Comput 38:158–172
Boehm BW (1984) Software engineering economics. IEEE Trans Softw Eng 1:4–21
Effendi A, Setiawan R, Rasjidi ZE (2019) Adjustment factor for use case point software effort estimation (study case: student desk portal). Procedia Comput Sci 157:691–698
Ezghari S, Zahi A (2018) Uncertainty management in software effort estimation using a consistent fuzzy analogy-based method. Appl Soft Comput 67:540–557
Gergonne J (1974) The application of the method of least squares to the interpolation of sequences. Historia Math 1(4):439–447
Hodges J, Lehmann EL (2012) Rank methods for combination of independent experiments in analysis of variance. In: Selected Works of EL Lehmann, Springer, pp 403–418
Holm S (1979) A simple sequentially rejective multiple test procedure. Scandinavian journal of statistics pp 65–70
Huang SJ, Chiu NH (2006) Optimization of analogy weights by genetic algorithm for software effort estimation. Inf Softw Technol 48(11):1034–1045
Idri A, Hosni M, Abran A (2016) Improved estimation of software development effort using classical and fuzzy analogy ensembles. Appl Soft Comput 49:990–1019
Jørgensen M (2004) Regression models of software development effort estimation accuracy and bias. Emp Softw Eng 9(4):297–314
Jørgensen M, Indahl U, Sjøberg D (2003) Software effort estimation by analogy and “regression toward the mean.” J Syst Soft. 68(3):253–262
Kadoda G, Cartwright M, Chen L, Shepperd M (2000) Experiences using case-based reasoning to predict software project effort. In: Proceedings of the EASE 2000 conference, Keele, UK, Citeseer
Kaushik A, Tayal DK, Yadav K, Kaur A (2016) Integrating firefly algorithm in artificial neural network models for accurate software cost predictions. J Softw Evol Proc 28(8):665–688
Kitchenham B, Mendes E (2009) Why comparative effort prediction studies may be invalid. In: Proceedings of the 5th international Conference on Predictor Models in Software Engineering, pp 1–5
Kocaguneli E, Menzies T (2013) Software effort models should be assessed via leave-one-out validation. J Syst Softw 86(7):1879–1890
Kumar PS, Behera HS, Kumari A, Nayak J, Naik B (2020) Advancement from neural networks to deep learning in software effort estimation: Perspective of two decades. Comput Sci Rev 38(100):288
Mittas N, Angelis L (2008) Comparing cost prediction models by resampling techniques. J Syst Softw 81(5):616–632
Nassif AB, Azzeb M, Idri A, Abran A (2019) Software development effort estimation using regression fuzzy models. Comput Intell Neurosci 2019
Ostertagová E (2012) Modelling using polynomial regression. Procedia Eng 48:500–506
Phannachitta P (2020) On an optimal analogy-based software effort estimation. Inf Softw Technol 125(106):330
Santosa F, Symes WW (1986) Linear inversion of band-limited reflection seismograms. SIAM J Sci Stat Comput 7(4):1307–1330
Shepperd M, MacDonell S (2012) Evaluating prediction systems in software project estimation. Inf Softw Technol 54(8):820–827
Shepperd M, Schofield C (1997) Estimating software project effort using analogies. IEEE Trans Softw Eng 23(11):736–743
Singh SP, Singh VP, Mehta AK (2018) Differential evolution using homeostasis adaption based mutation operator and its application for software cost estimation. J King Saud Univ Comp Inf Sci
Singh T, Singh R, Mishra KK (2018) Software cost estimation using environmental adaptation method. Procedia Comp Sci 143:325–332
Stigler SM (1974) Gergonne’s 1815 paper on the design and analysis of polynomial regression experiments. Historia Math 1(4):431–439
Tibshirani R (1996) Regression shrinkage and selection via the lasso. J Roy Stat Soc Ser B Methodol 58(1):267–288
Walkerden F, Jeffery R (1999) An empirical study of analogy-based software effort estimation. Emp Softw Eng 4(2):135–158
Wilcoxon F (1992) Individual comparisons by ranking methods. In: Breakthroughs in statistics, Springer, pp 196–202
Wolpert DH (1992) Stacke generalizedation. Neural Netw 5(2):241–259
Yeclar F, Kiline D, Borandag E, Ozclift A (2016) Regression analysis based software effort estimation method. Int J Softw Eng Knowl Eng 26(05):807–826
Zima K (2015) The case-based reasoning model of cost estimation at the preliminary stage of a construction project. Procedia Eng 122:57–64

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.