ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/337538341

Machine Learning Models for Software Cost Estimation

Conference Paper · September 2019

DOI: 10.1109/3ICT.2019.8910327

CITATIONS
5

READS
959

2 authors, including:

Mustafa Hammad
University of Bahrain
63 PUBLICATIONS **302** CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Network Routing View project

All content following this page was uploaded by Mustafa Hammad on 02 December 2019.

The user has requested enhancement of the downloaded file.

2019 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT)

Machine Learning Models for Software Cost Estimation

Mahmood Mohd Al Asheeri
Department of Computer Science
University of Bahrain
Sakheer, Bahrain
20003621@stu.uob.edu.bh

Mustafa Hammad
Department of Computer Science
University of Bahrain
Sakheer, Bahrain
mhammad@uob.edu.bh

***Abstract*— Software cost estimation is a critical task in software projects development. It assists project managers and software engineers to plan and manage their resources. However, developing an accurate cost estimation model for a software project is a challenging process. This paper builds a software cost estimation model using machine learning approach. Different machine learning algorithms are applied to two public datasets to predict the software cost in the early stages. Results show that machine learning methods can be used to predict software cost with a high accuracy rate.**

Keywords—Machine Learning, Cost Estimation, Prediction.

I. INTRODUCTION

Software cost estimation is a critical stage that is being done in the initial phases of software development process. The aim of such a process is to have a better future sight of the project progress and its phases. Another main objective is to have clear project details and specifications to assist stakeholders in managing the project in terms of human resources, assets, software, data and even in the feasibility study. Accurate estimation results with definitely helps the project manager to do better estimation for the project cost, the time required for various project phases and resources or assets. However, the inaccuracy may result from the project cost estimation process that will certainly affect the project delivery. A project with wrong or imprecision evaluation will face issues with delivery timing, resources required, budget or even in quality or operational side and sometimes the project may fail or aborted. Hence, the cost estimation is a significant part of the software projects and so it continues to be a complex issue in the software engineering field [1]. Therefore, many studies and researches have been conducted for the purpose of enhancing and improving the estimation process and get more accurate and dependable results.

On the other hand, recently Machine Learning (ML) techniques become very essential in software studies. In many scientific researches, ML methods are being used and executed most likely in the various fields, however, depending on the research nature and objectives one or more of the methods will be selected. As the process of software cost estimation is rapidly evolving which may include technology advances, team skills and experience, and tools and programming languages available, it gives superiority to ML techniques than some other methods that may stick to statistical and mathematical work [2]. Hence, ML can be a suitable technique to build the proposed model due to the ability to learn from historical data and adapt the wide variations that join software project development.

In this paper, 13 ML techniques are being evaluated and assessed through five main statistical indices. Data used in this study is collected from the public that is available on Internet. It contains practical software engineering data. Datasets can be downloaded from (http://tunedit.org/repo/promise/effortprediction) which is made publicly available to encourage and improve the cost estimation work in software engineering. At the end of this work, it can be concluded which ML methods are applicable to be used in software cost estimation process. Moreover, the process of assessing and comparing the scored results among the used ML methods will specify the best and most appropriate ML that can be used with minimum error rate.

The rest of this paper is divided into related work that concentrates on reviewing the software cost estimation and ML techniques. In the next section, the used ML methods, datasets, and evaluation criteria are explained in depth. Before ending with the conclusion and future work section, the experimental results are being compared and analyzed in details.

I. RELATED WORK

This section is divided into two main subsections. In the first section, the software cost estimation process is discussed from different angles. The second section is reviewing the ML methods usage and their implementations.

A. Software Cost Estimation Review

Many studies proposed different models for estimating software cost. Several models have proposed and built to find alternatives, enhance or support existing models. Constructive Cost Model (COCOMO) is considered as one of the most known models in the field of software cost estimation. For the purpose of improving COCOMO II accuracy model [3], Langsar et al. proposed a method to optimize the parameters used in COCOMO II model. The proposed model is capable of candling improper and unclear inputs in an efficient way and so improves software reliability. In 2011, a study [4] aimed to examine the results of applying fuzzy logic on COCOMO II and FL-COCOMO II and its effect on cost estimation. The research focused on SCE model and incorporating fuzzy logic to assess the inaccuracy of software attributes. The study' outcomes showed from applying various datasets that FL-COCOMO II model scored better estimation outcomes than the COCOMO II based on different assessment criteria.

Litoriya et al. [5] conducted an analysis of the cost drivers that affect directly the cost estimation model accuracy and a substitution process has done for those drivers with nearest

978-1-7281-3012-5/19/$31.00 ©2019 IEEE

values to show and prove the decrease in the software cost using Agile COCOMO II. In 2017, Saljoughinejad and Khatibi [6] proposed a new study based on COCOMO model to enhance and improve the accuracy of the software cost estimation process. The COCOMO model has been selected due to its flexibility and applicability to various types of projects. During the study, an analysis of cost drivers has been done using different meta-heuristic algorithms. The improvement process was based on the effective selection of the factors and coefficients used in the model. The improvement was on comprises cost drivers and coefficients estimations. Results showed that explicit superiority once a comparison is done between the proposed model and COCOMO or other models. Chen et al. [7], focused on software cost estimation using different models such as COCOMO and how it can be improved by using the WRAPPER feature in DM. It is basically depending on Feature Subset Selection (FSS), where it concentrates on mainly the most promising fields in the dataset and neglects the other to speed up the processing time. So by using the WRAPPER feature, they concluded that COCOMO model can be improved and its results could be more efficient. Khalifelu and Gharehchopogh [8], presented various software cost estimation models founded on data mining methods for the purpose of choosing appropriate Artificial Intelligence (AI) techniques that are needed in new projects. Their main aim of all experiments was to asset and compare different data mining approaches with intermediate COCOMO models with respect to the prediction's accuracy. The achieved results were promised and noticeable.

Beside many researches done on the COCOMO model and how to enhance it, there are other studies conducted trying to propose new models and algorithms that may add value to the field. Whigham et al. [9], suggested a baseline model that is essential to be used for all projects that are being developed and software effort estimation study is required. It is very important to compare the results with the baseline when a new or existing method is carried out to examine the prediction process. Their proposed model called Automatic Transformations Linear Model (ATLM) that could be used as a baseline for the comparison process between the software effort estimation methods.

Sarro et al. [10] introduced a new effort estimation algorithm that is based on a combination of Confidence Interval Analysis and assessment of the Mean Absolute Error (MAE). The developed algorithm has been tested and evaluated based on three different factors, however, the results were very promised. The experiments were done on the dataset that is collected from more than 700 software projects. Recently, Masoudi-Sobhanzadeh et al. [11] proposed a novel software model for feature selection. It can be applied in many fields including designing drug, biology, and image processing. The model starts by selecting a subset of features/factors based on optimizations algorisms to be transmitted later to the classifiers or learners. The learners are SVM, ANN and Decision Tree, which can be applied to regressions and classification datasets. Two types of optimization algorithms and the three classifiers can be applied by researchers to any dataset to use this model which is called *FeatureSelect*. The *FeatureSelect* has been tested on 8 different datasets in size and nature were great results have been observed.

B. Machine Learning Methods Review

On the other hand, machine learning algorithms considered to be vital in nowadays studies. ML techniques are widely used and their outcome results are dependable and reliable in many researches. In 2018, a deep study [12] analyzed 25 releases containing hundreds of classes to test the effort indicators. The study concluded that out of 18 machine learning algorithms used, IBk, KStar, Additive Regression, and Multi-Layer Perceptron were capable to estimate the test effort precisely. Moreover, the work in [13] proposed a prediction model to estimate the duration of the software processes using ML algorithms. Two training models which are Levenberg–Marquardt (LM) and Bayesian regularization back propagation (BR) used to test and evaluate FFNN and RBNN algorithms. The comparison between the two models showed that BR outcomes are slightly better. Also, BR is more desirable as its application is cost effective. Yeha and Deng [14] proposed a framework to predict the software product life cycle using two machine learning algorithms. The work presented a more precise and generalizable model for product cost estimation.

In 2019, research has been done on breast cancer [15] trying to build models for visualizing and detecting analytical signs of breast cancer survival rate using ML algorithms. To determine the important aspects of breast cancer survival rate, prediction algorithms were developed using the extreme boost, decision tree, neural networks, random forest support vector machine, and logistic regression. All the algorithms scored very high and close outcomes and the highest one was random forest which can be concluded that those methods could be used as predictive models in breast cancer studies. Some major challenges usually evolve software cost estimation process such as factors related to technology and creativity. In addition, some uncertain problems can happen during the implementation like lack of resources or increasing the cost of OS. Due to that, Kumari and Pushkar [1] introduced a hybrid algorithm to better estimate the SCE based on a combination of COA-Cuckoo and KNN. The hybrid algorithm runs on 6 different datasets and evaluated using 8 examined criteria. The overall results show improved accuracy on cost estimation.

Pospieszny et al. [2] developed an approach based on machine learning algorithms to limit the gap between recent researches and real implementations. The achieved results for the proposed model were very accurate and authors believe that it gives more realistic results in terms of software effort and duration estimation for practical projects. Furthermore, Pandey [16] did an analysis study on most of the techniques and methods used in software cost estimation. He tried to mention some of the main advantages and drawbacks of each one. He concluded that all project factors are important and critical to evaluate and estimate the cost of a project and they can differ in their important and influence from one project to another. The main factors that should be included in estimation metrics are qualitative: team experience, development environment, and culture; quantitative factors are project size and the available resources. Başkeleş et al. [17] carried out many experiments on software effort estimation using different machine learning methods based on three main dissimilar datasets. As a conclusion, they have

noticed that parametric models are inadequate for software effort estimation process.

Some other studies focused more on the cost estimation environment and other related factors like software development cycle associated with each project. For example, in 2018, Rahikkala et al. [18] introduced a study on the role of organizational phenomena and how its various factors can improve and influence the process of software cost estimation. Most of the researches focus on the development and improvement of the SCE including technical factors and methodologies without mentioning or analyzing the impact of the environment or organizational factors. By conducting a case study and quantitative research, the authors concluded that senior management responsibility is essential in making a significant estimation, whereas the daily follow up is not required. They also found that no significant individual factors that may affect directly the estimation process. Another work [19] conducted on Agile Development Life Cycle. Due to its high success rates and because of its rapid nature capability to adopt changes, Agile Development Cycle became very popular and widely used since the late 90s. Vyas, Bohra, Lamba, and Vyas, did a great job in 2017 by doing a survey to highlight the most trends related to the Agile Software Development (ASD) process and how it relates to the cost estimation. The authors were able to identify the factors to be included or not in the estimation process in order to get a more accurate and true cost for projects.

II. METHODOLOGY

In this section, the work methodology is discussed through the illustration of the used ML algorithms. Then the used datasets and its structure is explained. Lastly, the discussion is focused on the evaluation criteria.

A. Machine Learning Algorithms

In this section, a quick and brief overview is done on the ML methods that have been used in building prediction models.

Random Forest is a machine learning algorithm constructed on decision tree algorithms. It operates like a group of constructed decision trees that works independently where each tree is using a distinctive part of the dataset. In each tree, it consists of two randomization levels. The first one is called “bagging” or bootstrap of aggregation and the other level is at each node of the decision tree [20]. REPTree is an abbreviation of Reduced Error Pruning Tree. The tree is being built in a fast and learnable way depending on the gained information. REPTree is another kind of decision trees that uses regression tree, which can create many trees in various rounds or iterations. Then, out of all the generated trees, the best one is being selected. To do the pruning process for the tree, the mean square error is measured based on the tree predictions [21]. M5P method is another tree model that is being constructed based on Quinlan’s M5 algorithm. Originally, in addition to adding the linear regression method to the tree leave nodes, M5 model is also based on the conventional decision tree. The trained data is used by the algorithm to form the nodes and represent the decision tree model [22]. The ZeroR algorithm is one of the simplest classifiers. It considers all required potential values and the attributes being targeted. Based on the provided data and using the target attribute, the required output will always be found. This classifier does not have a rule that depends on the untargeted attribute [23].

The Decision table classifier is a classification model used in prediction studies. Its idea is similar to decision trees or neural networks, where it involves a hierarchal table in which each row is a top level being broken down to construct another new table. Its structure is very close to dimensional stacks [24]. Input Mapped Classifier works like a wrapper that specifies the mismatches between test and training data by trying to build a relation between data used in training which the classifier has been constructed based on and the received test cases or instances [25]. Additive Regression classifier improves the performance of classifiers that are based on regression. Each iteration done uses the residuals generated from previous iterations. It can overcome overfitting problem but it takes extra time [26]. IBK stands for Instance-Bases Learning with parameter K. Its will-known name is K-nearest Neighborhood (KNN), while it is used in Weka software as IBK. It determines the number of nearest neighbors to be used in the instance classification process [27]. In 2009, K-Star classifier was first introduced by Hussain Aljazzar. K means the number of shortest paths that can be found between a selected group of data points in a given graph [28].

Gussian Processes classifier can be considered as an example of non-parametric algorithms. It is used to deduce a distribution of random variables collection over functions. It attempts to find the similar points within the distribution to forecast the value [29]. Linear regression is a ML algorithm that is classified under supervised learning. It predicts values based on the independently provided attributes. It is widely used in finding relationships between datasets attributes and prediction studies. So this classifier tries to find a linear relationship between the input values and the one to be predicted [30]. Multi-Layer Perceptron is a neural network algorithm that consists of three main layers which are input layer, at least one hidden layer, then an output layer. Depending on the dataset and the problem, one or more nodes can compose the output layer. The input signals spread forward within the network while error signals spread backward. To reduce the error, some weight adjustments are being made [31]. SMOreg means Sequential Minimal Optimization that is an enhanced model of the SMO method which is based on support vector machine (SVM) used for regression. For un-linear prediction, SMOreg can be used in an efficient way. In SMOreg algorithm, some efficiency problems can be generated as a result of having one threshold [31].

B. Datasets and Evaluation Criteria

The datasets used in this work are prepared for software engineering experiments that are available publically. Two datasets will be used to test and compare the ML techniques, the first dataset is called **Usp05-ft** and the second is **Usp05**. Dataset (**Usp05-ft**) contains 76 instances and consists of the 15 attributes as described in Table 1.

In addition to the 15 attributes, the second dataset **(Usp05)** contains 203 instances and consists of two more attributes which are 16 and 17 as illustrated in Table 1.

TABLE I: DATASETS ATTRIBUTES DESCRIPTION

Item No.	Attribute	Description	Type
1	ID	Object ID	Positive integer
2	Effort	The actual total hours expended on the implementing process	Positive integer
3	IntComplx	The level of the Internal Calculation's complexity	1 to 5 that mean low to very high
4	DataEn	Total Number of Data-Entry Items	Positive number
5	DataFile	Total Number of Data-Files Accessed	Positive number
6	DataOut	Total Number of Data-Output Items	Positive number
7	UFP	Unadjusted Function Point Count	Positive number
8	Lang	Language Used	Language name
9	Tools	Used Platforms and Development Tools	Tools/platform name
10	ToolExpr	Language and Tool Experience Level	Range of number of months of experience
11	AppExpr	Applications Experience Level	1 to 5 which means low to very high
12	TeamSize	Size of the developing team	Range of min. to max. number
13	DBMS	Used DBMS	Database system name
14	Method	The used Implementation Methodology	e.g. OO, JAD
15	AppType	The used Architecture	e.g. C/S, Centered
16	ObjType	Type of the Object	PJ-project, FT-feature, RQ-requirement
17	Funct%	Percentage of Functionality of features or requirements	1 to 7

Going through Table 1, the main and most important attribute in this study is the second one which is "Effort". It will be used in the prediction process by the ML algorithms where the actual value in the dataset will be compared with the predicted one. Around 50% of the used attributes are integers, whereas the remaining are mostly strings describing the used Language, Database, Tools, and Platform. Datasets also contain some other significant attributes which are considered very important in development cycles like the size of the team, implementation methodology, and experience level.

To evaluate the ML algorithms performance after applying them on the used datasets, five basic statistical indices were used as performance and assessment criteria. The indices [32] [33] are Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Relative Absolute Error (RAE), Root Relative Squared Error (RRSE), and Correlation Coefficient (R^2). Basically, they are measuring the error rate between the actual effort within the dataset and the predicted effort using the ML algorithm. Assuming A is the actual effort, $\tilde{A}$ is the predicted effort, $\bar{A}$ is the mean of A, and n is the number of instances. The used measures can be calculated as the following formulas:

$$MAE = \frac{1}{n}\sum_{j=1}^{n} |A_j - \tilde{A}_j| \quad (1)$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{j=1}^{n}\left(A_j - \tilde{A}_j\right)^2} \quad (2)$$

$$RAE = \frac{\sum_{j=1}^{n} |A_j - \tilde{A}_j|}{\sum_{j=1}^{n} |A_j - \bar{A}_j|} \quad (3)$$

$$RRSE = \sqrt{\frac{\sum_{j=1}^{n}\left(A_j - \tilde{A}_j\right)^2}{\sum_{j=1}^{n}\left(A_j - \bar{A}_j\right)^2}} \quad (4)$$

$$R^2 = 1 - \frac{\sum_{j=1}^{n}\left(A_j - \tilde{A}_j\right)^2}{\sum_{j=1}^{n}\left(A_j - \bar{A}_j\right)^2} \quad (5)$$

III. EXPERIMENTAL RESULTS

Weka tool version 3.8 has been used to evaluate the used algorithms. The 10 folds' cross-validation technique is used to train the data on 90% and test it on the remaining 10% until the whole data (100%) is being used as a test data through 10 cycles. Six types of classifiers have been used in Weka to test the 13 algorithms. The used classifiers or algorithms are Random Forest, REPTree, M5P, ZeroR, Decision Table, Input Mapped Classifier, Additive Regression, IBK, KStar, Gussian Processes, Linear Regression, Multilayer Perceptron, and SMOreg. The 13 methods have been tested on two datasets, the first one is (**Usp05-ft**) with 15 attributes and 76 instances. The second one (**Usp05**) is with 17 attributes and 203 instances.

Table 2 shows the results of applying the 13 algorithms on the dataset (**Usp05-ft**). The evaluation and comparison were done on 5 statistical error measures. The last row in Table 2 represents the average values for each measurement criteria.

The Results in Table 2 show that there is a very close relationship between the predicted value and the actual one using Random Forest method. R^2 values are ranged from -1 to 1. As the result is more closed to 1 as the relation is stronger. The correlation coefficient for Random Forest is the highest one which is 0.8441. Moreover, Random Forest also scored the best results for the remaining measurement criteria. The other methods scored closed results to Random Forest such as IBK in MAE and RAE %, Multilayer-perception in RMSE and RRSE %.

TABLE II: PERFORMANCE RESULTS ON USP05-FT DATASET

Method	R^2	MAE	RMSE	RAE	RRSE
Random Forest	0.8441	2.5025	4.8546	41.7323	55.6778
REPTree	0.7607	3.2553	5.6754	54.2868	65.0918
M5P	0.705	3.2359	6.2428	53.9628	71.5994
ZeroR	-0.264	5.9965	8.7191	100	100
Decision Table	0.7137	3.4225	6.241	57.0738	71.579
Input-Mapped-Classifier	-0.264	5.9965	8.7191	100	100
Additive-Regression	0.7136	3.2441	6.5709	54.0993	75.363
IBK	0.7853	2.5132	5.8021	41.9101	66.5451
KStar	0.7797	2.7272	6.0219	45.4795	69.0658
Gussian-Processes	0.7604	2.8814	5.6809	48.0511	65.1554
Linear-Regression	-0.264	5.9965	8.7191	100	100
Multilayer-Perceptron	0.7979	2.8173	5.6413	46.9824	64.7004
SMOreg	0.7504	2.6597	6.2764	44.3547	71.9853
Average	**0.5244**	**3.6345**	**6.5511**	**60.6102**	**75.1356**

As shown in Table 2, all types of statistical measurements used to test and evaluate the error rate, Random Forest scored the best results in all the indices with the lowest error rate for the four types (MAE, RMAE, RAE, and RRSE) and highest correlation coefficient R^2. However, the worst results have been scores by three algorithms that shared the same scored number in all performance measurements which are ZeroR, Input Mapped Classifier, and Linear Regression. Methods that scored worst results where the RAE% and RRSE% reached 100% were not capable to generate valid prediction values, as the predicted effort was static for all the inputs. The last row in the table shows the average between the worst and best values.

As an example, Figure 1 represents the distribution of the values for the best results scored by the Random Forest algorithm. The x-axis shows the number of instances within the dataset, whereas the y-axis shows the predicted and actual values. This structure is applied to the remaining figures in this paper. The points in both figures show the differences between the actual effort points and the predicted ones using the Random Forest model. It is noticeable that the predicted values are closed to the actual ones. However, some predicted values are far from the dataset effort values which leads to increasing the error rate.

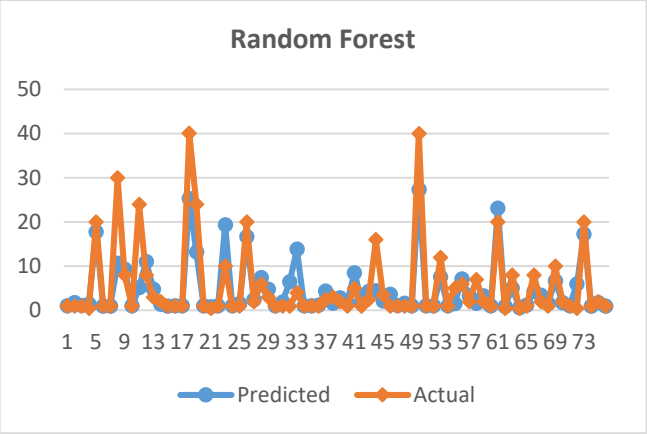

Fig. 1: Predicted and actual values distribution using Random Forest

Table 3 represents the performance measurement results after applying the 13 algorithms on the second dataset (**Usp05**). Two more attributes have been added to the dataset and the number of instances is almost three times than the first dataset. The last row is showing the average value for each measurement criteria.

TABLE III: PERFORMANCE RESULTS ON USP05 DATASET

Method	R^2	MAE	RMSE	RAE	RRSE
Random Forest	0.4319	8.1464	31.8046	60.5244	92.3641
REPTree	0.6296	10.0367	29.757	74.569	86.4177
M5P	0.4628	9.9475	30.6858	73.9061	89.1151
ZeroR	-0.311	13.4596	34.4339	100	100
Decision Table	0.5118	8.7527	30.1391	65.0297	87.5273
Input-Mapped-Classifier	-0.311	13.4596	34.4339	100	100
Additive-Regression	0.558	8.445	28.2736	62.7431	82.1098
IBK	0.2504	8.7266	38.9912	64.8355	113.2349
KStar	0.3626	7.3318	32.1284	54.4729	93.3046
Gussian-Processes	0.4978	8.2471	29.659	61.2729	86.1333
Linear-Regression	-0.311	13.4596	34.4339	100	100
Multilayer-Perceptron	-0.022	15.7926	36.513	117.3334	106.038
SMOreg	0.4962	8.8851	29.5932	66.0134	85.9422
Average	**0.2496**	**10.3608**	**32.3728**	**76.9770**	**94.0144**

As shown in Table 3, it can be observed that the best results are between 3 algorithms, which are REPTree, KStar and Additive Regression, while the worst error results have been scored either by IBk or Multilayer Perception. Regarding the R^2, the results are the same as in the first experiment as the worst outcomes have been scored by ZeroR, Linear Regression, and Input Mapped Classifier.

IV. CONCLUSION AND FUTURE WORK

In this work, 13 ML algorithms have been evaluated using two datasets. The evaluation criteria used in this work are R^2, MAE, RMAE, RAE, and RRSE. The aim of the proposed model is to predict the effort using dataset attributes and compare them with the actual effort in order to measure the error using different criteria. The higher the value of R^2 the better result, for the rest of the measurement criteria, the lower value means a better result. Random Forest achieved the best results in the first experiment using (**Usp05-ft**) dataset and another three models which are REPTree, Additive Regression and Kstar scored the best results using (**Usp05**) dataset. ZeroR method scored the worst results using the first dataset while some other methods including Multilayer Perceptron, IBk, and Linear Regression didn't perform well on the second dataset.

In the future, more datasets can be included in the study to have a wider angle and more variety in the inputs which will be reflected to have a better estimation and more accurate results. Moreover, some other ML algorithms can be tested and involved in upcoming studies in order to cover all available machine learning methods.

REFERENCES

[1] S. Kumari and S. Pushkar, "Cuckoo search based hybrid models for improving the accuracy," *Microsystem Technologies,* vol. 24, no. 12, pp. 4767-4774, 2018.

[2] P. Pospieszny, B. Czarnacka-Chrobot and A. Kobylinski, "An effective approach for software project effort and duration estimation with machine learning algorithms," *The Journal of Systems & Software,* vol. 184, 2017.

[3] K. Langsar, R. Sarno and Sholiq, "Optimizing Effort Parameter of COCOMO II Using Particle Swarm Optimization Method," *TELKOMNIKA,* vol. 16, no. 5, p. 2208–2216, 2018.

[4] I. Attarzadeh and S. H. Ow, "Improving Estimation Accuracy of the COCOMO II Using an Adaptive Fuzzy Logic Model," in *2011 IEEE International Conference on Fuzzy Systems*, Taipei, Taiwan, 2011.

[5] R. Litoriya, N. Sharma and D. A. Kothari, "Incorporating Cost driver substitution to improve the Effort using Agile COCOMO II," in *2012 CSI Sixth International Conference on Software Engineering*, 2012.

[6] R. Saljoughinejad and V. Khatibi, "A New Optimized Hybrid Model Based On COCOMO to Increase the Accuracy of Software Cost Estimation," *Journal of Advances in Computer Engineering and Technology,* vol. 4, no. 1, pp. 27-40, 2018.

[7] Z. Chen, T. Menzies, D. Port and B. Boehm, "Feature Subset Selection Can Improve Software Cost Estimation Accuracy," *ACM SIGSOFT Software Engineering Notes,* vol. 30, no. 4, 2005.

[8] Z. A. Khalifelu and F. S. Gharehchopogh, "Comparison and evaluation of data mining techniques with algorithmic models in software cost estimation," *Procedia Technology,* pp. 65-71, 2012.

[9] P. A. WHIGHAM, C. A. OWEN and S. G. MACDONELL, "A Baseline Model for Software Effort Estimation," *ACM Transactions on Software Engineering and Methodology,* vol. 24, no. 3, 2015.

[10] F. Sarro, A. Petrozziello and M. Harman, "Multi-objective Software Effort Estimation," in *ACM 38th IEEE International Conference on Software Engineering*, 2016.

[11] Y. Masoudi-Sobhanzadeh, H. Motieghader and A. Masoudi-Nejad, "FeatureSelect: a software for feature selection based on machine learning approaches," *BMC Bioinformatics,* vol. 20, no. 170, pp. 1-17, 2019.

[12] V. Vig and A. Kaur, "Test effort estimation and prediction of traditional and rapid release models using machine learning algorithms," *Journal of Intelligent and Fuzzy Systems,* vol. 35, no. 2, p. 1657–1669, 2018.

[13] A. Khalid, M. A. Latif and M. Adnan, "An Approach to Estimate the Duration of Software Project through Machine Learning Techniques," *Gomal University Journal of Research,* vol. 33, no. 1, pp. 1-13, 2017.

[14] T.-H. Yeha and S. Deng, "Application of machine learning methods to cost estimation of product life cycle," *International Journal of Computer Integrated Manufacturing,* vol. 25, no. 4/5, p. 340–352, 2012.

[15] M. D. Ganggayah, N. A. Taib, Y. C. Har, P. Lio and S. K. Dhillon, "Predicting factors for survival of breast cancer patients using machine learning techniques," *BMC Medical Informatics and Decision Making,* pp. 1-17, 2019.

[16] P. Pandey, "Analysis of the Techniques for Software Cost Estimation," in *2013 Third International Conference on Advanced Computing and Communication Technologies (ACCT)*, Rohtak, India, 2013.

[17] B. Başkeleş, B. Turhan and A. Bener, "Software Effort Estimation Using Machine Learning Methods," in *2007 22nd International Symposium on Computer & Information Sciences*, 2007.

[18] J. Rahikkala, S. Hyrynsalmi, V. Leppänen and I. Porres, "The Role of Organisational Phenomena in Software Cost Estimation: A Case Study of Supporting and Hindering Factors," *e-Informatica Software Engineering Journal,* vol. 12, no. 1, p. 167–198, 2018.

[19] M. Vyas, A. Bohra, D. C. Lamba and A. Vyas, "A Review on Software Cost and Effort Estimation Techniques for Agile Development Process," *International Journal of Recent Research Aspects,* vol. 5, no. 1, 2018.

[20] S. A. Woznicki, J. Baynes, S. Panlasigui, M. Mehaffey and A. Neale, "Development of a spatially complete floodplain map of the conterminous United States using random forest," *Science of the Total Environment,* vol. 647, p. 942–953, 2019.

[21] Kalmegh and Sushilkumar, "Analysis of WEKA Data Mining Algorithm REPTree, Simple Cart and RandomTree for Classification of Indian news," *International Journal of Innovative Science, Engineering & Technology,* vol. 2, no. 2, 2015.

[22] S.-a. Blaif, S. Moulahoum, R. Benkercha, B. Taghezouit and A. Saim, "M5P model tree based fast fuzzy maximum power point tracker," *Solar Energy,* vol. 163, pp. 405-424, 2018.

[23] T.Rajasekaran, P.Jayasheelan and K. S. Preethaa, "Predictive Analysis in Agriculture to Improve the Crop Productivity using ZeroR algorithm," *International Journal of Computer Science and Engineering Communications,* vol. 4, no. 2, pp. 1397-1401, 2016.

[24] Becker and B. G., "Visualizing decision table classifiers," in *Proceedings IEEE Symposium on Information Visualization*, 1998.

[25] "Class InputMappedClassifier," 2019. [Online]. Available: http://weka.sourceforge.net.

[26] "Additive Regression," 2019. [Online]. Available: https://www.cs.waikato.ac.nz/ml/weka/.

[27] gerardnico, "Machine Learning - K-Nearest Neighbors (KNN) algorithm - Instance based learning," 2017. [Online]. Available: https://gerardnico.com/.

[28] Universität Konstanz, "K* Algorithm (K Star)," 2019. [Online]. Available: https://www.sen.uni-konstanz.de/.

[29] M. Krasser, "Gaussian processes," 2018. [Online]. Available: http://krasserm.github.io.

[30] geeksforgeeks, "ML | Linear Regression," 2019. [Online]. Available: https://www.geeksforgeeks.org.

[31] P. Singh and S. Agrawal, "Node Localization in Wireless Sensor Networks Using the M5P Tree and SMOreg Algorithms," in *2013 5th International Conference on Computational Intelligence and Communication Networks*, 2013.

[32] J. Wang, L.-C. Yu, K. R. Lai and X. Zhang, "Dimensional Sentiment Analysis Using a Regional CNN-LSTM Model," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, 2016.

[33] C. Escalante-Sandoval and L. Amores-Rovelo, "Regional monthly runoff forecast in southern Canada using ANN, K-means, and L-moments techniques," *Canadian Water Resources Journal / Revue canadienne des ressources hydriques,* vol. 42, no. 3, pp. 205-222, 2017.

View publication stats