EasyChair Preprint

№ 9439

Random Forests for Predicting Software Effort Estimation Based on Use-Case Points Analysis

Ne'Mah Alrababa'H and Ahmed Banimustafa

December 9, 2022

Random Forests for Predicting Software Effort Estimation Based on Use-Case Points Analysis

Ne'meh AlRababeh [1, a)] and Ahmed BaniMustafa[2, b)]

[1] *Software Engineering Department, Isra University, Airport Road, Amman, Jordan.*
[2] *Data Science and Artificial Intelligence Department, Isra University, Airport Road, Amman, Jordan.*
[b)] *Corresponding author: banimustafa@gmail.com*

Abstract. Software estimation is vital for the success of software engineering projects. However, predicting software effort is difficult due to the complexity, intangibility, and diversity of software solutions and its involved expertise and underlying technology. This paper aims at enhancing the accuracy of software estimation using a data mining approach that combines Random Forests Regression with Use-Case Points analysis that is typically used in estimating effort in object-oriented software engineering projects. The experimental results of applying our proposed approach have demonstrated a significant improvement in the prediction accuracy of software effort estimation when compared to Use-Case Points estimation based on R-Squared (R2) and other metrics such as Mean Absolute Error (MAE), and the Mean Squared Error (MSE).

Keywords: Software Effort Estimation, Machine Learning, Data Mining, Random Forests, Use-Case Points.

INTRODUCTION

Software applications are expanding to span additional domains, architectures, and platforms. As the cost of software development rises owing to inflation, salary increases, and other factors, the demand for smarter, and more innovative software are greater than ever, necessitating the development of better, yet less expensive, and faster software development.

Software estimation is the process of approximating the effort, cost, time, and resources needed for building a software solution. It aims at developing software solutions efficiently within time, at lower cost and with less effort while meeting specified requirements. Software estimation is one of the most crucial factors for the success of software development projects. However, software estimation is particularly difficult due to the intangible nature of computer software solutions the fluctuation, and the inherited margin of estimation errors. Estimation is not only vital for fulfilling the software requirements, but also for ensuring the success of the entire project which is defined by the ability of the developer to create a software solution that meets the client's needs while complying with quality standards and remaining within the project-imposed constraints such as time, cost, and other limited resources.

To achieve a better prediction, estimation must be planned, and performed carefully taking into account the project's nature, application domain, complexity, and other factors 1. Traditional model-based and expert-based techniques are usually used for software estimation. The former depends on mathematical models such as COCOMO 2, Function Point Analysis 3, Delphi 4 and PERT 5, while the latter depends on expert judgment and analogy. However, these techniques involve a large margin of error, yet they were initially developed for estimating the development of traditional software systems. This makes them unsuitable for estimating modern software solutions which involve different technologies, paradigms, methodologies, architectures, tools, and programming languages. Use-Case Points (UCP) is one of the modern techniques that is used for estimating effort in object-oriented software solutions 6-8. It depends on the use-case model that represents the system functionality and its interaction with its environment. The UCP method depends on quantifying the number of the uses cases in the system and on qualifying the effort needed for implementing and realizing each use-case in addition to a set of technical and environmental factors.

Machine learning is an artificial intelligence field that involves investigating algorithms and models that mimic the learning process in animals and humans by following fixed and restricted software instructions and code that is set

by humans 9. Learning is one of the most important characteristics of intelligent behavior. Machine learning algorithms in general and Random Forests 10, 11. in particular have achieved excellent estimation accuracy compared to traditional model-based methods such as COCOMO, Function Point Analysis, and to expert-based methods 12-14.

This work investigates and evaluate the use of Random Forests regression as an alternative for predicting effort in object-oriented software projects using a dataset that was collected for 70 object-oriented software engineering projects. The dataset consists of a set of predictors that are related to the requirements of object-oriented software projects such as use-case and actor complexity in addition to technical and environmental factors which are used estimating effort in the UCP method. Applying Random Forests regression aims at enhancing the accuracy of software estimation and would help achieve a better prediction of software development time, cost and required resources which would eventually contribute to the success of software development projects. The application performance of the Random Forests model will be evaluated using R-Squared (R2), Mean Absolute Error (MAE), and the Mean Squared Error (MSE). These metrics will also be calculated for the effort values estimated by the Use-Case Points (UCP) algorithm to provide a valid comparison with the values predicted by the Random Forest Mode. Section 2. provides a critical review of the related works and literature, while Section 3. provides an overview of the dataset and research methodology. Section 4. presents the research findings and discusses its results, while Section 5. summarizes the research conclusion and future work.

RELATED WORK

Random Forests is one of the popular regression techniques in software estimation. Random Forest regression was introduced in several studies as an alternative to the traditional methods in software estimation. However, and despite its popularity as an alternative data mining machine learning to the classical algorithmic and parametric techniques such as COCOMO, Function Point Analysis (FP), Albrecht and other algorithmic methods, Random Forests has been rarely used for predicting effort in object-oriented software projects. This is due to the immaturity of software estimation in modern object-oriented projects and due also to the lack of studies which aimed at optimizing estimation methods for object-oriented development projects such as Use-Case Point Analysis (UCP).

In a study that used the random forest for predicting effort in five ISBSG datasets 15, that covered R8, Tukutuku, COCOMO, Desghrnis and Albrecht datasets, Random forest performance was found superior to regression trees based on NMRE, PRED(0.25) and MdRE metrics. A later study that was performed in 16 Random Forests also outperformed Support Vector Machine (SVM) and COCOMO in predicting software effort which was evaluated using MMRE and Correlation Accuracy. In a study that was designed to investigate regression as an alternative for the UCP method in early project stages, Random Forest achieved the best performance compared to Neural Networks (NN), radial Basis Function networks (RBFN), Long-Linear regression (LLR) and Stochastic Gradient Boosting (SGB) 17. Another study that was conducted in 18 Random Forest also achieved excellent results when applied based on staked ensemble learning compared to deep learning and single med approach. In an earlier study that was conducted by the co-author of this research, Random Forests also achieved the best performance when applied to COCOMO NASA dataset 13.

Furthermore, the validity and the excellent performance of Random Forests were also confirmed by three independent literature review studies which have been reported in 10, 19, 20. In addition, the use of a private dataset in some of the studies reported in the literature limits the reproducibility of these studies and validates the quality of the reported dataset. In addition, it is worth mentioning here that due to the private nature of some of the datasets, the variability of dataset size and the features included in each of the reported studies, it is quite apparent that the comparison of the prediction accuracy across various studies can be invalid even if the same valuation metrics were applied in these studies. In the context of this study, we also found that only a few studies have reported the use of the random forest for predicting effort in object-oriented software engineering projects based on use-case point analysis 17. However, the dataset used in these studies was quite different from the dataset that is used in this research.

METHODOLOGY

The proposed methodology is based on combining machine learning techniques with use-case point analysis. The methodology consists of three major phases and two other supporting activities. The first phase involves preparing data for model construction through data Preprocessing and exploration. The second phase is related to model construction using both use-case point analysis and machine learning. The third phase is dedicated to model testing and evaluation. The first supporting activity involves aggregating the dataset by combining the data related to software requirements, project attributes and other historical data, while the last activity involves evaluation of the results

considering the defined hypothesis, formulated question, and the set objectives as well as the results reported in the related literature. Figure 1. illustrates the research methodology and its core phases and supporting activities.

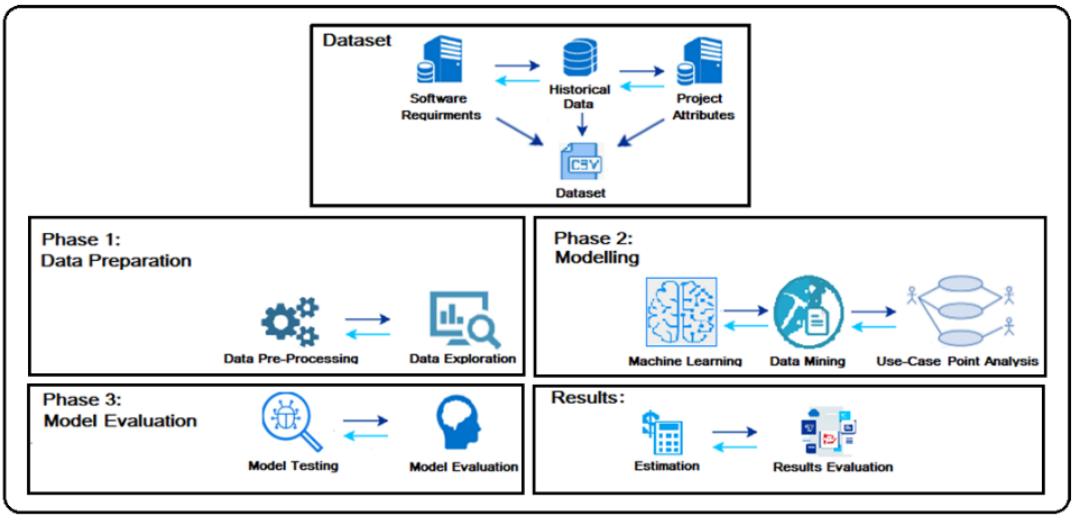

Figure 1. Proposed Research Methodology: consists of three phases that cover data preparation, modelling & model evaluation

Dataset

The dataset involves 70 software projects [21, 22] that span application domains such as communication, banking, manufacturing, insurance, services, government, wholesale, retail, medical and health care sectors. The projects are conducted using contemporary programming languages and involve using modern databases and other software technologies such as Java, C#, C++, CSP, Delphi, and Visual Basic and involve several other modern databases and other software technologies including Oracle, SQL, XML, .Net and power builder. The data set involves applying a wide spectrum of software development process models that include Waterfall, Rapid Application Development (RAD), Incremental, Personal Software Process (PSP) and Rational Unified Process (RUP). The dataset contains a number of UCP factors that are based on historical data which are related to the project's software requirements. Other attributes in the project profile are related to the number of simple, average, and complex use cases and actors in dataset in addition to 13 technical and 8 environmental factors [23].

Dataset Feature	Meaning
Project_No	Project Identification
Simple Actors	Number of actors classified "Simple"
Average Actors	Number of actors classified "Average"
Complex Actors	Number of actors classified "Complex"
UAW	A calculated field that is based on the equation for finding the Unadjusted Actor weight
Simple UC	Number of actors classified "Simple"
Average UC	Number of actors classified "Average"
Complex UC	Number of actors classified "Complex"
UUCW	A calculated field that is based on the equation for finding the Unadjusted Use-Case weight
TCF	A calculated field that is related to projects technical complexity factors
T1-T13	13 Project Technical Complexity Scored Factors
ECF	A calculated field that is related to projects environmental complexity factors
ENV1 - ENV8	8 Project Environmental Complexity Scored Factors
Real_P20	The Real Effort that is spent in each project measured in terms of Person hours. The productivity factor was set to 20.
UCP	A calculated field that is based on UCP equation depending on the values of UAW, UUCW, TCF and ECF.
Estimated_Effort_Person_Hour	The value of effort estimation based on the UCP method.
Real_Effort_Person_Hours	The real development effort spent in each project measures in person-hours.
Sector	The sector of the application domain of the developed software
Language	The programming language in which the code of each project was written.
Methodology	The software engineering methodology that was used in each project.
ApplicationType	The application type of the software project.
DataDonator	The identification of the data donor

Phase 1: Data Preparation

The data preparation phase aims at preparing the dataset for modelling by applying the data pr-processing procedures that are needed to oversee issues that are related to the quality of data and to make it suitable for the applied machine modelling technique which is in this case random forest's regression. This phase may also involve exploring the dataset to gain a better understanding of its samples, features, and implied meanings. It may also involve the visualization of the dataset and applying basic statistics that aim at prospecting its potential trends, patterns, and correlations, in addition to uncovering issues such as data distribution, missing values and outliers 24-27.

Phase 2: Modelling

This phase aims to construct a model that combines machine learning with Use-Case Point Analysis. The first step involves combining the features extracted from use-case point analysis with project attributes and other requirements related features based on a dataset that is combined with historical data that was collected from previous software engineering projects, while the second step will involve applying random forests regression to predict the actual effort that is needed for the software development.

Use-Case Points Analysis

The first step in this phase involves applying Use Case Points Analysis (UCP) 28 based on the use-case descriptive characteristics which cover actors weighting factors, several use-cases and their degree of complexity that is specified based on the number and complexity of the transactions prescribed in the use-case flow of events, analysis classes in addition to its realization scenarios. It also considers technical and environmental factors that also affect the software development estimation. Use case diagrams are used as input, while the UCP is calculated by converting the components of the use case diagram into size metrics that can be then estimated considering a set of complexity and technical variables. 29, 30, 31 and 32. Equation 1 is used for conducting these calculations.

$$Effort = Prouctivity * UCP \qquad \textit{Equation 1}$$

Where the productivity value is usually set based on historical data.

Four components are used to compute the UCP. The first component is produced by solving Equation 2 and computing the Unadjusted Actor Weights (UAW). Actors are divided into three groups (simple actors a_s, average actors a_a, and complex actors). Simple actors represent systems that use a well-defined application programming interface (API), while average actors are those that communicate with the system via a protocol such as TCP/IP, FTP, and HTTP or via data storage systems such as Files and RDBMS. Complex actors refer to graphical user interface (GUI) or web page.

$$UAW = 1 * a_s + 2 * a_a + 3 * a_c \qquad \textit{Equation 2}$$

Where a_a, a_b, and a_c represent the number of simples, average, and complicated use cases, respectively.

The second component is referred to as the Unadjusted Use Cases (UUCW). UUCW require classifying use cases into simple use case, average use case, and complex use case based on the number of transactions that involve each use case. A transaction represents stimuli and responses between actors and the system 33. A simple use case contains up to 3 transactions, while an average use case contains 4-7 transactions. A complex use case contains more than 7 transactions. UUCW can be calculated by Equation 3

$$UUCW = 5 * uc_s + 10 * uc_a + 15 * uc_c \qquad \textit{Equation 3}$$

Where uc_s, uc_a, and uc_c represent the number of simples, average, and complicated use cases, respectively.

The third component used for calculating UCP involves assigning a set of thirteen technical complexity factors (TCF) that might influence and with values that range between 0 and 5. Equation 4 is then used to calculate the TCF.

$$TCF = 0.6 + (0.01 * \sum_{i=1}^{13}(f_i * fw_i)) \qquad \textit{Equation 4}$$

Where f_i denotes the effect value of factor i, fw_i denotes the weight associated with factor i from Table 1.

Table 1 UCP Technical factors.

	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13
Technical factor	Distributed System	Responsive Objective	End User Efficiency	Complex processing	Reusable Code	Easy to Install	Easy to Use	Portable	Easy to change	Concurrent	Security Feature	Access for Third party	Special Training Required
Weight (Fw)	2	1	1	1	1	0.5	0.5	2	1	1	1	1	1

The fourth component is based on assigning eight environmental complexity factors (ECF) that might influence and with values that range between 0 and 5. Equation 5 is then used to calculate the ECF.

$$ECF = 1.4 + (-0.03 * \sum_{i=1}^{8}(E_i * Ew_i)) \qquad \textit{Equation 5}$$

Where E_i denotes the effect value of factor i, Ew_i denotes the weight associated with factor i from Table 2.

Table 2. UCP Environmental Factors

	E1	E2	E3	E4	E5	E6	E7	E8
Environmental factor	Familiarity with the development process used	Application experience	Object-oriented experience of the team	Lead analyst capability	The motivation of the team	Stability of requirements	Part-time staff	Difficult programming language
Weight (Ew)	1.5	0.5	1	0.5	1	2	-1	-1

The results of the UCP calculation are determined by Equation 6., while the final estimated effort in person-hours is calculated by Equation 7.

$$UCP = (UUCW + UAW)\ X\ TCF\ X\ ECF \qquad \textit{Equation 6}$$

$$EE = UCPX\ Hours/UCP \qquad \textit{Equation 7}$$

Where $Hours/UCP$, is an assumed value that denotes the number of hours spent on each use-case.

Random Forests Regression

The random Forests method is an ensemble learning algorithm that was proposed by. It is used for classification and regression. It works by building several decision tree models and then improving the performance of the ensemble model, it combines the results from different decision tree models 11. When used for regression, random forests build several decision trees based on a random vector l with tree predictor h (x, l). The predictor's output is h(x), and the actual value is Y, as in Equation 8.

$$\hat{h}(x) = \bar{y}_k = \frac{1}{n}\Sigma_{i=1}^{n} y_{k_i} \qquad \textit{Equation 8}$$

Phase 3: Model Evaluation

The model evaluation phase involves applying matrices to evaluate the performance of the constructed random forests regression model which covers: R-Squared Error (R2), Mean Absolute Error (MAE), and the Mean Squared Error (MSE) 34. R2 is used to assess the performance of the regression model by measuring the goodness of fit between the observed data points and those that are predicted using the constructed regression model 35, 36. Its values range between 0 and 1. The closest to 1 is the better. The R2 is measured by finding the ratio between the sum of squares regression (SSR) that represents the variation of the predicted values and the sum of squares total (SST) which represents the variation of the actual values which is represented in Equation 9, where ŷ is the predicted value of y.

$$\mathrm{R2} = \frac{SSR}{SST} = \frac{\Sigma(\hat{y}_i - \bar{y})^2}{\Sigma(y_i - \bar{y})^2} \qquad \textit{Equation 9}$$

The Mean Absolute Error (MAE) measures the goodness of regression model fitness by calculating the total absolute difference between the actual and the predicted values divided by the number of values 37. MAE is

represented by Equation 10, while the Mean Squared Error (MSE) is calculated by summing the squared difference between the actual and the predicted values divided by the number of values 35. MSE is represented by Equation 11.

$$MAE = \frac{1}{N}\sum_{i=1}^{N} |y_i - \hat{y}| \qquad \textit{Equation 10}$$

$$MSE = \frac{1}{N}\sum_{i=1}^{N} (y_i - \hat{y})^2 \qquad \textit{Equation 11}$$

The results will also be assessed in terms of the validity of the research hypothesis, their ability to answer the research question and to achieve the research aim. The random forests regression prediction results will also be compared with the real effort reported for each project reported in the historical data and with the effort estimated based on the Use-Case Points method only.

RESULTS & DISCUSSION

To provide a solid ground for comparing the performance of the constructed Random Forests regression model and the performance of the UCP method. The same performance measurements were used for both methods. Six calculated fields were added to the dataset representing the R2 , MAE and MSE in addition to six other features that were used for the intermediate calculations which were computed using excel functions.

The results of applying random forests regression on the dataset that represents profiles of UCP projects have demonstrated excellent improvements when comparing the effort prediction that was achieved using the UCP method only. Figure 2. provides a useful visualization of the Random Forest model using a scatter plot that visualizes the Random Forests model predictions compared to the those calculated by the UCP method and those that represent the actual effort spent on each of the projects. The triangle shape values represent the predictions obtained using the Random Forests regression model, while the square shape values represent the values calculated using the UCP method. The diamond shape values refer to the values of the real effort that was spent in each project. The x-axes represent the numbers that are allocated for identifying the projects, while the y-axes represent the values of the effort measured in person-hours.

The scatter provides insight into the values predicted by the Random Forest model and those calculated by the UCP method. The plot in the diagram shows that the effort values that are predicted by the Random Forest regression model, which were found remarkably close to the real effort recorded for most of the projects. In contrast, the line plot shows that most of the effort values that are estimated based on UCP calculation and which are shown as squares on the plot, are far from the data points which represents the real effort. However, and despite their apparent fluctuation, the line shows that these values still show a trend that goes with the line that represent the values of the real effort. The plot in fact, confirms the applied evaluation metrics which are used to compare the prediction accuracy of the Random Forests and the UCP method.

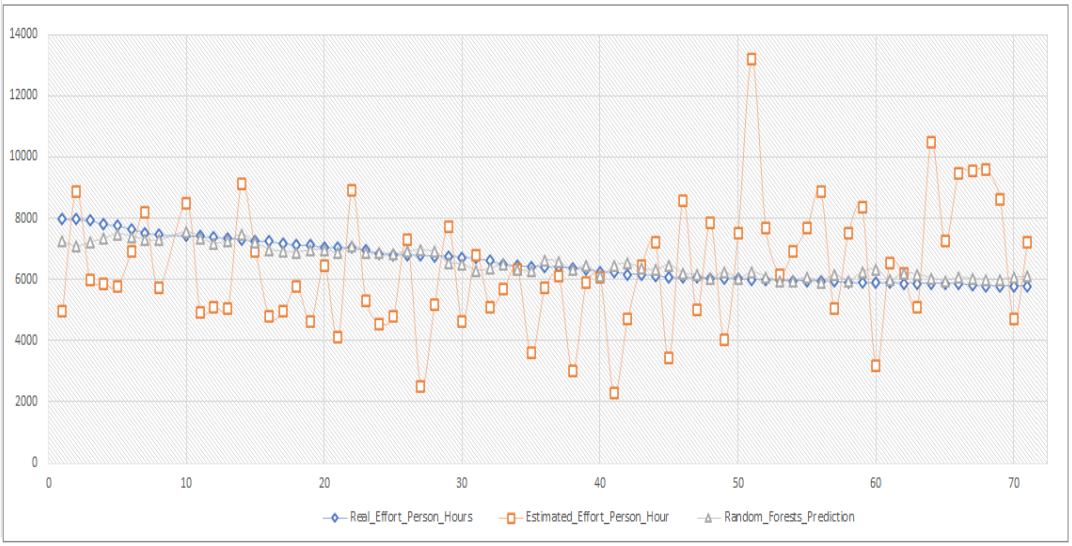

Figure 2. A Visualization of the Prediction of Random Forests compared to Values Calculated by UCP and the Actual Effort

The evaluation results of the constructed random forests regression model have shown excellent estimation effort results which demonstrated an accurate and excellent prediction power which confirms the validity of our proposed methodology. The constructed model scored outstanding performance in R2 Error, MAE and MSE matrices, which are by far better than those achieved when estimating effort using UCP calculations. Table 3. Shows a comparison between the performance of the constructed model compared to the UCP calculation.

Table 3. Performance of the Random Forest Model Compared to UCP Estimation

	R-Squared Error (R2)	MAE	MSE
UCP Estimation	0.03	1819	4886436.6
Random Forests Prediction	0.86	192	64907.6

The R2 error for the Random Forests model was found 0.86 compared to the 0.03 which was calculated for the values estimated using the UCP method. In R2 metric, the closer the values to 1 is the better. In the MAE error metric, the Random Forests model scored 192 compared to the value of 1819 scored by the UCP method. In the MSE metric the Random Forests model scored 4886436.6 compared to 64907.6 for the UCP method. In MAE and MSE metrics, the smaller values are always the better.

The results confirmed the validity of the research hypothesis and were able to provide an affirmative answer to the research question. It also fulfilled the aims of the research which was set to improve UCP estimation by reducing its error rate and increasing its accuracy which was demonstrated in the results illustrated in Figure 2. and Table 3.

CONCLUSION

This paper reported the application of random forest regression as a data mining and machine learning technique that was applied to a dataset that represents profiles of software development historical data that was collected for object-oriented real-life projects. Random forests regression has achieved excellent scores in R2, MAE and MSE metrics. These scores were far better than those achieved by UCP calculations when evaluated based on the same metrics. The prediction results of the constructed model were found excellent when compared to the estimation calculated using UCP equations. This confirms the potential of data mining and machine learning algorithms to be used as alternatives or as complements to improve the accuracy of traditional software estimation techniques.

Future work for extending this study would involve applying other data mining and machine learning algorithms to improve the regression prediction results as well as using supervised classification algorithms to predict software estimation levels. Variable importance algorithms can also be used to rank factors that contributed more to the power of prediction. Such results can be beneficial to software projects managers when planning, managing, and performing software projects.

REFERENCES

1. S. M. Satapathy, Doctorate thesis, National Institute of Technology Rourkela, india (2016).
2. B. Boehm, B. Clark, E. Horowitz, C. Westland, R. Madachy and R. Selby, Annals of software engineering **1** (1), 57-94 (1995).
3. C. R. Symons, IEEE Transactions on Software Engineering **14** (1), 2-11 (1988).
4. C. M. Goodman, J Adv Nurs **12** (6), 729-734 (1987).
5. K. R. MacCrimmon and C. A. Ryavec, Operations Research **12** (1), 16-37 (1964).
6. E. R. Carroll, in *Companion to the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications* (ACM SIGPLAN San Diego, CA, USA, 2005), pp. 257-265.
7. R. K. J. T. J. o. D. S. E. Clemmons, **19** (2), 18-22 (2006).
8. S. Kusumoto, F. Matukawa, K. Inoue, S. Hanabusa and Y. Maegawa, in *10th International Symposium on Software Metrics, 2004. Proceedings.* (IEEE, 2004), pp. 292-299.
9. A. Ławrynowicz and V. Tresp, in *Perspectives on Ontology Learning* (IOS Press, Amsterdam, Netherlands, 2014), pp. 35-50.
10. H. Mustapha and N. Abdelwahed, Procedia computer science **148**, 343-352 (2019).

11. L. Breiman, Mach. Learn. **45** (1), 5-32 (2001).
12. Y. Mahmood, N. Kamaa, Y. Mahmood, N. Kamaa and M. Ali, Software: Practice and experiance; 23 (2), 1-27 (2021).
13. A. BaniMustafa, in *2018 8th International Conference on Computer Science and Information Technology (CSIT)* (IEEE, Amman, Jordan, 2018), pp. 249-256.
14. A. B. Nassif, M. Azzeh, A. Idri and A. Abran, Comput Intell Neurosci **2019**, 8367214 (2019).
15. A. Zakrani, M. Hain and A. Namir, International Journal of Intelligent Engineering and Systems **11** (6), 300-311 (2018).
16. N. A. Zakaria, A. R. Ismail, A. Y. Ali, N. H. M. Khalid, N. Z. J. I. J. o. A. C. S. Abidin and Applications, **12** (6) (2021).
17. S. M. Satapathy, B. P. Acharya and S. K. Rath, IET Software **10** (1), 10-17 (2016).
18. P. V. A G, A. K. K and V. Varadarajan, Electronics **10** (10) (2021).
19. Y. Mahmood, N. Kama, A. Azmi, A. S. Khan and M. Ali, Software: Practice Experience **52** (1), 39-65 (2022).
20. A. Ali and C. Gravino, Journal of software: evolution process **31** (10), e2211 (2019).
21. R. Silhavy, P. Silhavy and Z. Prokopova, Mendeley Data **1**, 2017 (2017).
22. R. Silhavy, P. Silhavy, Z. J. J. o. S. Prokopova and Software, **125**, 1-14 (2017).
23. M. Ochodek, J. Nawrocki and K. Kwarciak, Information Software Technology **53** (3), 200-213 (2011).
24. A. BaniMustafa, The ISC International Journal of Information Security **11** (3), 79-89 (2019).
25. A. Banimustafa and N. Hardy, IEEE Access **8**, 209964-210005 (2020).
26. A. H. BaniMustafa and N. W. Hardy, in *Plant Metabolomics* (Springer, 2011), pp. 317-333.
27. A. M. Bani Mustafa, Aberystwyth University, 2012.
28. G. Karner, Objective Systems SF AB **17**, 1-9 (1993).
29. M. Azzeh and A. Bou Nassif, Journal of Software: Evolution and Process **29** (3) (2017).
30. M. Ochodek, J. R. Nawrocki and K. Kwarciak, Information and Software Technology **53** (3), 200-213 (2011).
31. A. Issa, M. Odeh and D. Coward, in *2nd International Conference on Information Communication Technologies* (IEEE, 2006), Vol. 2, pp. 2766-2771.
32. A. Bou Nassif, L. F. Capretz and D. Ho, in *Companion Proceedings of the 36th International Conference on Software Engineering* (ACM, Hyderabad, India, 2014), pp. 612–613.
33. G. Robiolo, C. Badano and R. O. Orosco, in *Proceedings of the Third International Symposium on Empirical Software Engineering and Measurement, ESEM 2009* (IEEE, Lake Buena Vista, Florida, USA, 2009), pp. 422--425.
34. F. Emmert-Streib and M. Dehmer, Machine learning knowledge extraction **1** (1), 521-551 (2019).
35. D. Harvey, S. Leybourne and P. Newbold, International Journal of forecasting **13** (2), 281-291 (1997).
36. D. Chicco, M. J. Warrens and G. Jurman, Peer J Computer Science **7**, e623 (2021).
37. C. J. Willmott and K. Matsuura, Climate research **30** (1), 79-82 (2005).