Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2024 SCPE. Volume 25, Issues 2, pp. 1276–1285, DOI 10.12694/scpe.v25i2.2213

SOFTWARE EFFORT ESTIMATION USING MACHINE LEARNING ALGORITHMS

KRUTI LAVINGIA [*] RAJ PATEL [†] VIVEK PATEL [‡] AND AMI LAVINGIA [§]

Abstract. Effort estimation is a crucial aspect of software development, as it helps project managers plan, control, and schedule the development of software systems. This research study compares various machine learning techniques for estimating effort in software development, focusing on the most widely used and recent methods. The paper begins by highlighting the significance of effort estimation and its associated difficulties. It then presents a comprehensive overview of the different categories of effort estimation techniques, including algorithmic, model-based, and expert-based methods. The study concludes by comparing methods for a given software development project. Random Forest Regression algorithm performs well on the given dataset tested along with various Regression algorithms, including Support Vector, Linear, and Decision Tree Regression. Additionally, the research identifies areas for future investigation in software effort estimation, including the requirement for more accurate and reliable methods and the need to address the inherent complexity and uncertainty in software development projects. This paper provides a comprehensive examination of the current state-of-the-art in software effort estimation, serving as a resource for researchers in the field of software engineering.

Key words: Software Engineering, Machine Learning, Effort Estimation

1. Introduction.

A large volume of data is produced while software companies develop and generate software [1]. From the requirements phase until maintenance, a unique collection of data is generated at each stage of software development. In software development project management, key factors such as Lines of Code (LOC), historical project data, team skill levels, team size, functional and non-functional requirements, project phases, and development timelines are crucial for success. LOC quantifies code changes, while past project data informs resource allocation. Team skills and size impact productivity, while requirements shape the project's direction [2]. Monitoring project phases and adhering to timelines ensures progress and identifies bottlenecks. These elements collectively enable effective project management, guiding teams to deliver software solutions that align with stakeholder expectations and meet project goals efficiently.

The data produced in software repositories is collected and maintained by software organizations as part of their ongoing efforts to improve the software quality. Several data mining techniques are used to analyze the vast amounts of data kept in software repositories to uncover new patterns and standout data points [3].

Two-thirds of all large projects greatly exceed their original projections and one-third of projects go over budget and are delivered late, as claimed by surveys [4].

The technique of estimating the effort and resources needed to construct a software system is known as software engineering cost estimation. Managers and stakeholders typically use this process to plan and budget software development projects. Cost estimation methods can range from simple rule-of-thumb calculations to more formal methods, such as parametric modelling or expert judgment. Factors that can influence the cost of a software project include the size and complexity of the system, the development methodologies and tools used, and the skill level of the development team [5]. It is essential to note that cost estimation is an iterative process that needs to be repeatedly refined and updated as more information becomes available throughout the project.

In software engineering, effort estimation is the process of predicting the number of human resources, measured in person-hours or person-months, needed to complete any software development project. It is a critical aspect of project management as it helps stakeholders to plan and budget for the project and to make

[*] Nirma University (kruti.lavingia@nirmauni.ac.in)
[†] Institute of technology, Nirma University (20bce218@nirmauni.ac.in)
[‡] Institute of technology, Nirma University(20bce226@nirmauni.ac.in)
[§] Sal College of Engineering (ami.lavingia@sal.edu.in)

informed decisions about resource allocation and project scope [6].

Software engineering has a variety of effort estimating techniques, each having advantages and disadvantages. Among the more popular techniques are:

1.1. Methods for effort estimation.

- **Expert judgment**: This method relies on the experience and expertise of individuals who have previously worked on similar projects. It is often used as a starting point for effort estimation and can provide a rough estimate of the required effort. However, it is subject to bias and can be affected by an individual's experience [7].
- **Analogous estimation**: This method uses the data from previous similar projects to estimate the effort required for the current project. It is a quick and easy method, but it is not always accurate, as the projects may not be entirely similar [8].
- **Three-point estimation**: This method uses the most likely, optimistic, and pessimistic estimates of effort to generate a range of possible values. It helps generate a range of likely effort estimates, but it is a relatively complex method [9].
- **Parametric estimation**: This method uses mathematical models to estimate the effort required for a project. It is based on the project's size, complexity, and other [10].

Estimating is crucial in project management, as inaccuracies in estimation can lead to poor project performance, potentially resulting in project failure. One of the management factors that cause about 65% of lost projects is poor estimation technique [11]. This study uses machine learning to create a model for software cost estimation. As a result, this review aims to test if machine learning is a better technique than using traditional methods to estimate software development effort or vice versa. Support machine learning algorithms such as vector regression, regression algorithms such as simple linear regression, and decision tree-based regression are applied in this study with the assistance of the Python programming language.

The remainder of this paper is broken up into related work that focuses on previous studies that have been done in this particular area. The following section examines machine learning methods and software cost estimation. They utilized machine-learning techniques, data sets, and evaluation standards are thoroughly detailed in the next section. The comparison and in-depth analysis of the experimental results come before the conclusion and section on future work.

However, we do wish to stress the purpose of this paper is to consider how different prediction systems perform under the same conditions and how to evaluate them, not to argue in favor of any particular prediction technique.

2. Related work. Numerous research has suggested various models for calculating the cost of the software. To find alternatives, improve upon, or support existing models, multiple models have been proposed and constructed [12]. Various new models have been developed and constructed to discover alternatives, improve existing models, or assist current models. A well-known method for estimating software costs is the build cost model.

2.1. Without Machine Learning. As introduced by Barry Boehm, the Constructive Cost Model (COCOMO) stands out as the predominant approach within the algorithmic methods category [13]. It relies on a series of equations and parameters derived from past software project experiences for estimation, and its models have garnered widespread practical acceptance. In the context of COCOMO, code size is measured in Thousand Lines Of Code (KLOC), and effort is expressed in person-months. COCOMO is a valuable tool to gauge the quality and effort required for software projects, as exemplified by its application in Manikavelan's study [14], providing approximate estimates within fixed time frames. Moreover, the authors in this particular research extended COCOMO's capabilities by incorporating the Gaussian Membership Function, revealing outstanding performance of the fuzzy-COCOMO model in terms of reducing relative errors.

In the study conducted by Nandan and Deepak [15], a novel approach was employed. They utilized a hybrid BATGSA algorithm to optimize the COCOMO model, drawing data from NASA databases. The study comprehensively compared three distinct techniques implemented using MATLAB. The outcome was a noteworthy decrease in normalized error with the updated COCOMO model.

Notably, the authors introduced an innovative hybrid strategy that amalgamated fuzzy clustering, ABE, and ANN approaches to enhance the accuracy of effort estimation. This novel approach entailed clustering all projects within a newly established framework, effectively mitigating the influence of inconsistent and irrelevant projects on projections. This research resulted in significant improvements, with an average enhancement of 0.25 in the first dataset and remarkable gains of 52 and 94 per cent in the second dataset, as demonstrated by the prediction percentage (PRED) and mean magnitude of relative error (MMRE) performance indicators.

2.2. With Machine Learning. In their study, Shukla et al. leveraged the Desharnais dataset to explore the performance of various machine learning models in estimating software project effort [16]. Notably, their MLPNN model achieved an R2 value of 0.79380, surpassing other models like LR, SVM, and KNN. It effectively explained 79% of the estimated variance, with only marginal differences (6-7%) in R2 values among them.

The research delved into the association between the most correlated elements by Pearson correlation and the effort variable using seven machine learning methods, following an initial correlation analysis of each dataset variable with the effort variable. Performance evaluation was based on error values [17].

Sarro introduced an effort prediction technique combining Confidence Interval Analysis and Mean Absolute Error assessment [18]. This innovative approach demonstrated promise through trials involving over 700 software programs, finding applications in diverse fields like pharmaceutical research, biochemistry, and computer vision. The method selected feature subsets based on optimization techniques and transferred them to classifiers (SVM, ANN, and Decision Tree) for classification and regression tasks involving two optimization algorithms and three classifiers. This process, known as Feature Selection, yielded excellent results across various datasets.

In another study, 93 projects' preprocessed COCOMO NASA benchmark data were employed to make predictions using machine learning techniques like Naive Bayes, Logistic Regression, and Random Forests [19]. Performance evaluation metrics such as Classification Accuracy, Precision, Recall, and AUC were employed following five-fold cross-validation. Each method outperformed the benchmark COCOMO model in production prediction.

V. Anandhiin's investigation focused on regression techniques, notably the M5 algorithm and Linear Regression, for estimating software cost using the Constructive Cost Model dataset [20]. The results indicated that the M5 method exhibited more minor errors, including the mean magnitude of the relative error and Median magnitude of the relative error, compared to Linear Regression in prediction. These clear distinctions highlight where different methods are introduced and provide insights into the authors' approach and findings.

3. Machine Learning. In this section, methods and ML algorithms are discussed; after that information about the dataset is stated along with its structure. Finally, the evaluation standards are the topic of conversation. Below stated section provides a clear and concise summary of some machine learning algorithms that are used to predict the effort for the project.

Machine learning techniques are increasingly thought to be crucial in research. The results of ML approaches are consistently reliable, and they are frequently employed reliable in numerous studies. Using two machine learning methods, Yeha and Deng provided a system to forecast the software product life cycle [21]. The study provided a more accurate and adaptable model for estimating product costs.

To distinguish between different types of breast cancers, Aleriza and Bita, used support vector machines, K-nearest neighbors, and neural network classifiers [22].

Other studies concentrated more on the environment for cost assessment and other relevant elements, such as the software development life cycle relevant to the particular project. For instance, in 2018, research on the impact of organizational factors were published by Rahikkala et al., which looked at how its many components could potentially influence and improve the software cost planning process [23].

3.1. Linear Regression. Based on other attributes' values, linear regression analysis will predict a variable's value [24]. Two types of variables are there in the algorithm, one dependent and another independent. The dependent variable is predicted by an algorithm. The dependent variable is predicted using the independent variable as a basis. Such analysis determines the coefficients of linear mathematical equations using independent variables that may most effectively anticipate the value of the dependent variable. This algorithm fits the output on a straight line to reduce the discrepancy between the actual and anticipated output. The value of A (the dependent variable/attribute) is then estimated from B (the independent attribute/variable).

The technique used is straightforward and comparative. It is less complex than other methods for predictions. The equation for linear regression:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_n X_n + e \tag{3.1}$$

Y is the dependent variable and β_1 ,β_2 ,...,β_n are coefficients and X_1, X_1, ... ,X_n are the independent variables. Here β_0 is the intercept of the line which is generated by the vertical axis. e is an error term; it is the random error used to express some random factors' effect of some random factors on Y (dependent variables).

3.2. Support Vector Regression. Drucker et al. initially presented the Support Vector Regression (SVR) model for regression analysis in 1997 [25].

Convex quadratic programming issues are replaced with convex linear system solutions in the least-squares SVR (LS-SVR) [26] [27], which greatly speeds up training. An extensive empirical investigation has revealed that the LS-generalization SVR's performance is comparable to the SVR's. (Van Gestel et al., 2004) [26].

Among these, Vapnik's SVR, which employs regression analysis via the support vector machine (SVM), has a wide range of applications in the energy-prediction industries. The principle of structural risk minimization underpins SVR. It has clear advantages for small datasets and can maintain excellent generalization ability [28].

In support vector regression, there are two types of hyperparameters, first being the *Kernel* function and second is a variable C which is defined as the penalty parameter of the error term. Kernel parameters affect the separation boundary. Since SVR is a kernel-based method, the performance of SVR heavily depends on the kernel functions. Different kernel equations are there, which can be applied to get the different decision boundaries.

These kernels project the input data into several high-dimensional feature spaces. Because high-dimensional feature spaces perform so well, creating new kernel functions can surpass SVR performance thresholds.

The distribution and shape of the dataset affect how the kernel is used. Here, the objective function for SVR's normal vector's size is $|w|$, which is minimized [26].

$$\mathbf{x_i w + b = 0} \tag{3.2}$$

$$\mathbf{x}_i\mathbf{w} + b \geq +1, y_i = +1 \tag{3.3}$$

$$\mathbf{x}_i\mathbf{w} + b \leq -1, y_i = -1 \tag{3.4}$$

$$y_i(\mathbf{x}_i\mathbf{w} + b) - 1 \geq 0, \quad \forall i \tag{3.5}$$

$$\text{minimize } \tfrac{\|w\|^2}{2} \tag{3.6}$$

$$\text{Maximize } W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{l} \alpha_i\alpha_j y_i y_j (\mathbf{x}_i\mathbf{x}_j) \tag{3.7}$$

3.3. Random Forest Regression. Tin Kam Ho first introduced random forests in 1995. For the creation of a decision tree, the Random forests method is applied, which solves problems of classification and regression.

Later, Breiman extended the technique by combining bagging with features selected at random [29], allowing for the controlled construction of multiple decision trees using variance. Compared to decision trees, the random forest algorithm provides more accurate error rate estimates. More particular, it has been demonstrated mathematically that the error rate always tends to converge as the number of trees rises. In general, because they can quickly adjust to nonlinearities detected in the data, random decision forests tend to predict better than linear regression [30]. Random forest regression produces better results compared to other algorithms, such as support vector machines and Neural Networks, and it is also robust against over lifting [31]. This algorithm can forecast the result by running an unpruned regression on each n-ary tree from the training data and then combining the results of the nary tree forecasts.

3.4. Decision Tree Regression. A tree-based structure called decision tree regression is used to forecast the dependent variable's numerical results. Quinlan's M5 algorithm [32] is implemented using what is known as the M5P algorithm. M5P is a tree-based structure similar to CART (Classification And Regression Trees). M5P-based trees have a multivariate linear model, whereas regression trees had values at the leaves. The trees generated by the classification and regression trees are generally more prominent than M5P-generated model trees.

A tree is built using the usual decision-tree method. This decision tree employs splitting criteria to account for intra-subset variation in the class values of the samples that go down each branch. The formula below can be used to calculate the standard deviation decrease.

$$SDR = sd(T) - \sum_i \frac{|T_i|}{|T|} sd(T_i) \tag{3.8}$$

The abrupt discontinuities that would inevitably arise between nearby linear models at the pruned tree's leaves are corrected by using a method.

After the machine learning algorithms had been applied to the necessary datasets, five key statistical indicators were used as performance and assessment criteria to evaluate the success of the algorithms. Indices [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.

In essence, they are utilizing the ML algorithm to calculate the error between predicted effort and actual effort found in the dataset. Assuming that $\tilde{A}$ is the real effort (dependent variable, to be predicted), $\bar{A}$ is the mean of A, and n is the number of individual data points available. The following formulae can be used to compute the error measures. Equations for the indices are:

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |A_i - \tilde{A}_i| \tag{3.9}$$

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (A_i - \tilde{A}_i)^2} \tag{3.10}$$

$$RAE = \frac{\sum_{i=1}^{n} |A_i - \tilde{A}_i|}{\sum_{n=1}^{n} |A_i - \bar{A}_i|} \tag{3.11}$$

$$RRAE = \sqrt{\frac{\sum_{i=1}^{n} |A_i - \tilde{A}_i|}{\sum_{n=1}^{n} |A_i - \bar{A}_i|}} \tag{3.12}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (A_i - \tilde{A}_i)^2}{\sum_{n=1}^{n} (A_i - \bar{A}_i)^2} \tag{3.13}$$

4. Research Methodology.

4.1. Data Pre Processing. The data utilized in this work is from a NASA software engineering dataset (NASA93) that is publicly available and contains information on several projects that have undergone development at NASA throughout the years. The dataset is structured in ARFF (Attribute Relation File Format) format and comprises 93 rows and 26 columns. The nominal values have been transformed into comparable values for better training. After the descriptive columns that didn't help with training were removed, they were discarded. To make the dataset easier to work with throughout the computation, it is then transformed into the comma-separated values format. Table 4.1 shows a detailed description of the data being used to train the various models

Table 4.1: Description of the data

Attribute	**Symbol**	**Description**	**Datatype**
Project	project name	Name	String
Category of application	cat2	Which field is this project related to	String
System	forg	Flight system or Ground system	character [f , g]
NASA center	center	Which NASA center had worked on the project	Number between 1 to 6
Capability of analyst	acap	Increase these to decrease effort	Positive integer
Capability of programmers	pcap		
Domain experience	aexp		
Current programming techniques	modp		
Software tool usage	tool		
Experience with programming languages	lexp		
Experience with VM	vexp		
Time restriction	sced		
The primary memory restriction	stor	Increase these to decrease effort	Positive integer
Database size	data		
Runtime restriction on CPU	time		
Turnaround time	turn		
Machine volatility	virt		
The difficulty of the process	cplx		
Required Software reliability	rely		
Equivalent physical line of code	equivphyskloc	Kilo lines of code	Positive integer
Development effort	act_effort	The effort in terms of person-month	Positive integer

Table 4.2: Parameters By GridSearchCV for SVR

Variable	**Value**
c	10
Gamma	Auto
Kernel	Linear

4.2. Model Training. "Development effort" is the primary dependent variable that is under study. The dataset's actual value will be compared to the expected value as part of the prediction process carried out by the Machine Learning algorithms. The given dataset predominantly consists of more than 70% numerical data. The data set is then divided at random into a training set and a testing set at a ratio of 70:30.

4.3. Evaluation, Results, and Discussion. In this section, the result of the experiment is discussed and displayed. The default parameters were employed for the LinearRegression and DecisionTreeRegression models. The parameters for SupportVectorRegression are displayed in the table 4.2. In the case of RandomForestRegression, all default parameters are utilized except for the specification that sets *max_depth* to 4. In table 4.2, Best parameters returned by GridSearchCV algorithm for SVR is shown.

Figures 4.1 to 4.3 shows the relationship between various independent variable and dependent variable.

Figure 4.4 provides a clear illustration of the relationship between CPU runtime restrictions and both database size and memory requirements. As the database size and memory requirements expand, the CPU

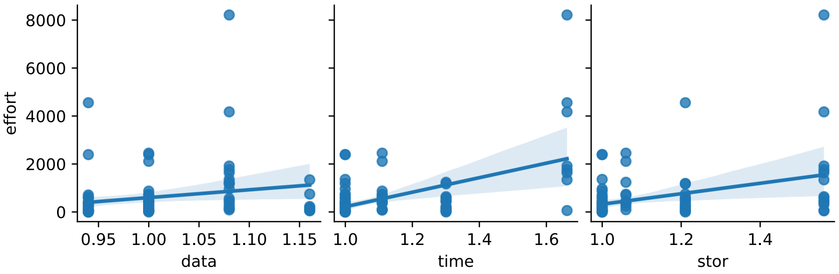

Fig. 4.1: Plotting Dependent Variable(Effort) vs. Independent Variables

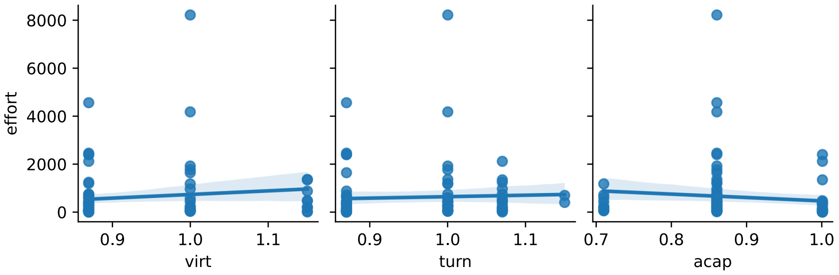

Fig. 4.2: Plotting Dependent Variable(Effort) vs. Independent Variables

experiences increased runtime restrictions due to the necessity for it to handle a greater volume of tasks within the same time frame. Furthermore, an interesting correlation emerges: an escalation in CPU runtime restrictions corresponds to a concurrent increase in the required software reliability. This suggests a dependency between these variables.

Additionally, it is noteworthy that as an analyst's capability improves, there is a parallel enhancement in their domain familiarity. This underscores the positive relationship between analyst competency and domain knowledge acquisition.

The outcome obtained using various error-measuring methods on various models is shown in table 4.3 demonstrates that the Random Forest method's anticipated value and the actual value have a very close connection. Usually R^2 values are between -1 and 1. For highly correlated data R^2 value is closer to 1. Random Forest has the highest correlation coefficient compared to all other models. The other techniques for model comparison include MAE and RAE%, RMSE and RRSE%. RMSE is commonly utilized and is regarded as a preferable all-purpose error measure for numerical forecasts. In general, random decision forests tend to predict better than linear regression because they can quickly adjust to nonlinearities detected in the data. Since the Root Mean Squared Error value for Support Vector Regression is smaller, it suggests that its predictions on the test data are more accurate compared to the other models used in this study.

4.4. Conclusion. To establish the cost required staff, and schedule for software development, it is the job of the project manager to estimate the effort of development. The findings of several studies indicate that early-stage project estimating errors is the primary cause of software project failures. The performance of any estimating approach depends on a number of factors, including project complexity, project duration, personnel skill, development process, and others. The usage of several cost-estimating methodologies is reviewed in this essay. This paper's contribution is the improvement of our understanding of the subject of research provided by the literature review. No approach, namely along the RMSE dimension, estimates software development effort especially well in the absolute sense but comparing relatively, Random Forest Regression yields the best results as it has the lowest R^2 and MAE. Additionally, practical applications of ML-based approaches could

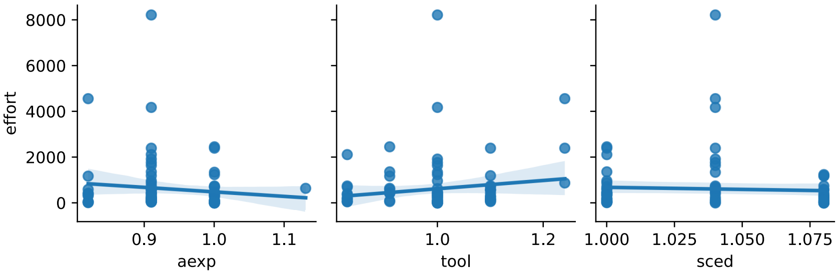

Fig. 4.3: Plotting Dependent Variable(Effort) vs. Independent Variables

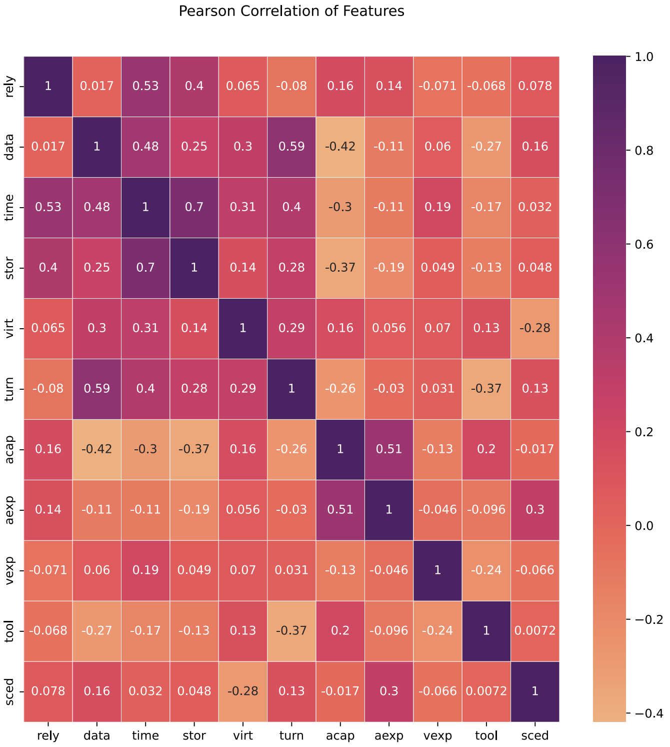

Fig. 4.4: Heat map showing Correlation between different Independent variables

involve project managers and data scientists working collaboratively to gather relevant data, including historical project data, developer expertise, project complexity, and duration. By employing machine learning algorithms on this data, project managers can enhance their ability to make more accurate early-stage project estimates, potentially reducing the risk of project failures and improving overall project management. Future research should focus on selecting optimal metrics for cost assessment, particularly leveraging computational intelligence methods.

REFERENCES

[1] Dhamija, A. & Sikka, S. A Review Paper on Software Engineering Areas Implementing Data Mining Tools & Techniques. *International Journal Of Computational Intelligence Research (IJCIR)*. **13** pp. 559-574 (2017,5)

Table 4.3: Outcomes of ML models using error measuring indices

Method Name	**MAE**	**RMSE**	**RAE**	**RRSE**	$\mathbf{R}^2$
Linear Regression	760.11	1365.5	1.45	1.24	-0.41
Support Vector Regression	178.7	323.61	1.05	1.042	-0.067
Random Forest Regression	642.5	1481.7	0.85	1.070	0.2781
Decision Tree Regression	760.0	1666.1	0.82	1.098	0.0873

[2] Srinivasan, K. & Fisher, D. Machine learning approaches to estimating software development effort. *IEEE Transactions On Software Engineering.* **21**, 126-137 (1995)

[3] Sharma, P. & Singh, J. Systematic Literature Review on Software Effort Estimation Using Machine Learning Approaches. *2017 International Conference On Next Generation Computing And Information Systems (ICNGCIS).* pp. 43-47 (2017)

[4] Moloekken-OEstvold, K., Joergensen, M., Tanilkan, S., Gallis, H., Lien, A. & Hove, S. A survey on software estimation in the Norwegian industry. *10th International Symposium On Software Metrics, 2004. Proceedings..* pp. 208-219 (2004)

[5] Dakwala, A. & Lavingia, K. A Machine learning approach to improve the efficiency of Fake websites detection Techniques.

[6] Walkerden, F. & Jeffery, R. An Empirical Study of Analogy-based Software Effort Estimation. *Empirical Software Engineering.* **4**, 135-158 (1999,6,1), https://doi.org/10.1023/A:1009872202035

[7] Hughes, R. Expert judgement as an estimating method. *Information And Software Technology.* **38**, 67-75 (1996), https://www.sciencedirect.com/science/article/pii/0950584995010459

[8] Shepperd, M. & Schofield, C. Estimating software project effort using analogies. *IEEE Transactions On Software Engineering.* **23**, 736-743 (1997)

[9] Royce, W. Managing the development of large software systems: concepts and techniques. *Proceedings Of The 9th International Conference On Software Engineering.* pp. 328-338 (1987)

[10] Baker, F. & Kim, S. Item response theory: Parameter estimation techniques. (CRC press,2004)

[11] McManus12, J. & Wood-Harper, T. Understanding the sources of information systems project failure. (2007)

[12] Ami, R., Mehta, V. & Lavingia, K. Analyzing the non-linear effects in DWDM optical network using MDRZ modulation format. *International Journal Of Advance Engineering And Research Development (IJAERD) E-ISSN.* pp. 2348-4470

[13] Boehm, B., Abts, C., Brown, A., Chulani, S., Clark, B., Horowitz, E., Madachy, R., Reifer, D. & Steece, B. Software cost estimation with COCOMO II. (Prentice Hall Press,2009)

[14] Manikavelan, D. & Ponnusamy, R. Software quality analysis based on cost and error using fuzzy combined COCOMO model. *Journal Of Ambient Intelligence And Humanized Computing.* pp. 1-11 (2020)

[15] Nandal, D. & Sangwan, O. Software cost estimation by optimizing COCOMO model using hybrid BATGSA algorithm. *International Journal Of Intelligent Engineering And Systems.* **11**, 250-263 (2018)

[16] Shukla, S. & Kumar, S. Applicability of neural network based models for software effort estimation. *2019 IEEE World Congress On Services (SERVICES).* **2642** pp. 339-342 (2019)

[17] Lavingia, K. & Mehta, R. Information retrieval and data analytics in internet of things: current perspective, applications and challenges. *Scalable Computing: Practice And Experience.* **23**, 23-34 (2022)

[18] Sarro, F., Petrozziello, A. & Harman, M. Multi-objective software effort estimation. *2016 IEEE/ACM 38th International Conference On Software Engineering (ICSE).* pp. 619-630 (2016)

[19] BaniMustafa, A. Predicting Software Effort Estimation Using Machine Learning Techniques. *2018 8th International Conference On Computer Science And Information Technology (CSIT).* pp. 249-256 (2018)

[20] Anandhi, V. & Chezian, R. Regression techniques in software effort estimation using cocomo dataset. *2014 International Conference On Intelligent Computing Applications.* pp. 353-357 (2014)

[21] Yeh, T. & Deng, S. Application of machine learning methods to cost estimation of product life cycle. *International Journal Of Computer Integrated Manufacturing.* **25**, 340-352 (2012)

[22] Osareh, A. & Shadgar, B. Machine learning techniques to diagnose breast cancer. *2010 5th International Symposium On Health Informatics And Bioinformatics.* pp. 114-120 (2010)

[23] Rahikkala, J., Hyrynsalmi, S., Leppänen, V. & Porres, I. The role of organisational phenomena in software cost estimation: A case study of supporting and hindering factors. *E-Informatica Software Engineering Journal.* **12**, 167-198 (2018)

[24] Rong, S. & Bao-Wen, Z. The research of regression model in machine learning field. *MATEC Web Of Conferences.* **176** pp. 01033 (2018)

[25] Drucker, H., Burges, C., Kaufman, L., Smola, A. & Vapnik, V. Support vector regression machines. *Advances In Neural Information Processing Systems.* **9** (1996)

[26] Xu, S., An, X., Qiao, X., Zhu, L. & Li, L. Multi-output least-squares support vector regression machines. *Pattern Recognition Letters.* **34**, 1078-1084 (2013), https://www.sciencedirect.com/science/article/pii/S0167865513000196

[27] Joy, R., Lavingia, A. & Lavingia, K. Performance evaluation of transmission distance and bit rates in inter-satellite optical wireless communication system. *Int J Adv Technol Eng Sci (IJATES).* **4**, 36-40 (2021)

[28] Zhong, H., Wang, J., Jia, H., Mu, Y. & Lv, S. Vector field-based support vector regression for building energy consumption prediction. *Applied Energy.* **242** pp. 403-414 (2019), https://www.sciencedirect.com/science/article/pii/S0306261919304878

[29] Breiman, L. Random forests. *Machine Learning.* **45**, 5-32 (2001)

[30] Schonlau, M. & Zou, R. The random forest algorithm for statistical learning. *The Stata Journal.* **20**, 3-29 (2020) (2019)

[31] Ho, T. Random decision forests. *Proceedings Of 3rd International Conference On Document Analysis And Recognition.* **1** pp. 278-282 (1995)

[32] Quinlan, J. & Others Learning with continuous classes. *5th Australian Joint Conference On Artificial Intelligence.* **92** pp. 343-348 (1992)

[33] Al Asheeri, M. & Hammad, M. Machine learning models for software cost estimation. *2019 International Conference On Innovation And Intelligence For Informatics, Computing, And Technologies (3ICT).* pp. 1-6

Edited by: Katarzyna Wasielewska-Michniewska
Regular paper
Received: Apr 1, 2023
Accepted: Jan 8, 2024