Software Development Effort Estimation Techniques: A Survey

Farah B. Ahmad1*, Laheeb M. Ibrahim2

1-2Department of Software Engineering, College of Computer Sciences & Mathematics, University of Mosul, Mosul, Iraq

E-mail: 1*farah.20csp11@student.uomosul.edu.iq, 2laheeb_alzubaidy321966@uomosul.edu.iq

(Received November 24, 2021; Accepted January 17, 2022; Available online March 01, 2022)

DOI: 10.33899/edusj.2021.129868.1156, © 2022, College of Education for Pure Science, University of Mosul. This is an open access article under the CC BY 4.0 license (http://creativecommons.org/licenses/by/4.0/)

ABSTRACT:
Software Effort Estimation (SEE) is used in accurately predicting the effort in terms of (person–hours or person–months). Although there are many models, Software Effort Estimation (SEE) is one of the most difficult tasks for successful software development. Several SEE models have been proposed. However, software effort overestimation or underestimation can lead to failure or cancellation of a project.

Hence, the main target of this research is to find a performance model for estimating the software effort through conduction empirical comparisons using various Machine Learning (ML) algorithms. Various ML techniques have been used with seven datasets used for Effort Estimation. These datasets are China, Albrecht, Maxwell, Desharnais, Kemerer, Cocomo81, Kitchenham, to determine the best performance for Software Development Effort Estimation. Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and R-Squared were the evaluation metrics considered. Results and experiments with various ML algorithms for software effort estimation have shown that the LASSO algorithm with China dataset produced the best performance compared to the other algorithms.

Keywords: Software Effort Estimation (SEE); Machine Learning (ML); Random Forest; Decision Tree; Support Vector Machines (SVM).

تقرئيات تقدير جهود تطوير البرمجيات: دراسة

فرح باسل احمد1، لهيب محمد ابراهيم2

قسم البرمجيات، كلية علوم الحاسوب والرياضيات، جامعة الموصل، الموصل، العراق

المлюص: يتم استخدام تقدير جهد البرمجيات (Software Effort Estimation (SEE)) لتلبية ويشكل دقيق بالجهد من حيث (عدد ساعات العمل أو عدد ساعات العمل)، وعلى الرغم من وجود العديد من النماذج فإن تقدير جهد البرمجيات يعد من أصعب المهام لتطوير البرمجيات الناجحة. حيث تم اقتراح العديد من نماذج تقدير جهد البرمجيات ومع ذلك، فإن الإفراط في تقدير جهد البرمجيات أو النقص في تقدير جهد البرامج يؤدي إلى إلغاء المشروع أو فشل المشروع. إن الهدف الرئيسي لهذا البحث هو العثور على نموذج آداء لتقدير جهد البرمجيات من خلال إجراء دراسة ومقارنات تجريبية

Albrecht، China، Cocomo81، Kemerer، Maxwell Desharnais، Kitchenham
البرمجيات. حيث تم اعتبار الجذر التربيعي لمتوسط الخطأ (RMSE) ومتوسط الخطأ المطلق (MAE) كمقياس للتقييم الذي تم أخذه في الاعتبار. أظهرت النتائج والتجارب مع خوارزميات التعلم الآلي المختلفة لتقدير جهد البرمجيات أن خوارزمية LASSO مع مجموعة بيانات china أنجبت أفضل أداء مقارنة مع خوارزميات التعلم الآلي الأخرى.

الكلمات المفتاحية: تقدير جهد البرمجيات؛ التعلم الآلي؛ الغابة عشوائية؛ شجرة القرار، دعم آلات النواقل.

1. Introduction
Estimating software efforts is an important part of developing software applications because it allows application development teams to finish the process of development on time and within budget [1]. Despite the complexity of estimating the effort, when done correctly, effort estimating serves as a foundation for all following project planning and management stages. The correct progress of the development depends on the correct estimation. As a result, several studies and research have been conducted with the goal of improving the prediction process and obtaining more exact and dependable findings.

Software effort estimation (and other estimations like software size estimation, software cost estimation, and software schedule estimation) create some certainty and commitment and are used for planning and making decisions for the project (planning budget, investment, and pricing). Figure 1 shows the overlap between the types of software estimation process in the life cycle of software development. [2][23]

![Type of software estimation process.][2]

Software estimation consists of the following:

- **Cost Estimation**: Cost estimation plays a significant role in the entire software development cycle, so it is must be completed before the development cycle begins and can continue throughout the software's life cycle [3][24]
- **Effort Estimation**: One of the steps in a software development project that aims to produce high-quality software that can be delivered on time and on budget while meeting the project’s requirements, from which the most realistic value of the effort required is calculated to develop or maintain software that is not yet developed on the basis of inputs. [4] [25]
- **Size Estimation**: Size Estimation is defined as determining the resulting size when developing software. Size Estimation is necessary for software project planning and management.[5] [27]
- **Staffing Estimation**: Staffing Estimation is manipulating a suitable model for software project planning to address the problem of human resource allocation [6] [26]
- **Time estimation**: time estimation is regarded as one of success factors in software development. It is associated with the cost of the project. If the time estimation of the project exceeds its deadline and the budget overruns, then we can consider it as failure of management that fails to deliver the project in the estimated time and cost. [7] [28]

Several models for estimating software effort have been proposed [1]. Initially, estimates of software effort are made using expert judgment [46], the use case point approach [49], user stories [50], function
point [51], and analogy-based estimations [48]. Later, for estimation, multiple Machine learning algorithms such as Linear regression, multiple linear regression, logistic regression, ridge regression, neural networks, lasso regression, decision tree, support vector machine, stepwise regression, Naive bayes, Elasticnet regression, random forest, and so on were used [8] [35][44]. The ensemble methods have gained a lot of attention and also produced more predictions than individual effort prediction algorithms [43]. This paper is a survey of techniques for estimating software development efforts. The following is how this paper is organized. Section 2 is Effort Estimation materials and methods which provides an overview of the Dataset, Evaluation Metrics and Evaluation techniques used in evaluation. Section 3 explains the results and discussion; finally, Section 4 presents conclusions and future work perspectives.

2. Materials and Methods
This section explains Effort Estimation materials and methods which provide an overview of the Dataset, Evaluation Metrics, and Evaluation techniques used in evaluation of Estimation Effort of software.

2.1 Datasets
There are many databases used for estimating effort, which are: China, Kemerer, Cocomo81, Albrecht, Maxwell, Desharnais and Kitchenham. Table 1 displays the repository details of datasets, including the number of Records, attributes, datasets of records, and the output unit of each dataset.

<table>
<thead>
<tr>
<th>Dataset Name</th>
<th>Records</th>
<th>Attributes</th>
<th>Unit of output</th>
</tr>
</thead>
<tbody>
<tr>
<td>China [38]</td>
<td>499</td>
<td>16</td>
<td>Person-Unit of hours</td>
</tr>
<tr>
<td>Kemerer[37]</td>
<td>15</td>
<td>7</td>
<td>Person-Unit of months</td>
</tr>
<tr>
<td>Cocomo81 [39]</td>
<td>63</td>
<td>17</td>
<td>Person-Unit of months</td>
</tr>
<tr>
<td>Albrecht[41]</td>
<td>24</td>
<td>8</td>
<td>Person-Unit of months</td>
</tr>
<tr>
<td>Maxwell[36]</td>
<td>62</td>
<td>27</td>
<td>Person-Unit of hours</td>
</tr>
<tr>
<td>Desharnais[40]</td>
<td>81</td>
<td>12</td>
<td>Person-Unit of hours</td>
</tr>
<tr>
<td>Kitchenham[42]</td>
<td>145</td>
<td>9</td>
<td>Person-Unit of hours</td>
</tr>
</tbody>
</table>

2.2. Evaluation Metrics
The Evaluation Metrics used to Estimate efforts like Percentage of Predictions (Pred), Root Relative Absolute Error (RRAE), Mean-Square Error (MSE), and Correlation Coefficient (CC), selection and of evaluation metrics can affect the success of the search. Therefore, in this paper more general evaluation metrics are used in the majority of papers, like Mean Absolute Error (MAE), Root Mean Squared Error (RMSE) and R-Squared.[29] [33] [45]

1) **Mean Absolute Error (MAE)** [9][32] [34]
The average sum of all absolute errors is referred to as MAE.
Prediction error = (Actual value-Predicted value).
Absolute error = |Prediction error|.

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |Actual value_i - Predicted value_i|$$ (1)

2) **Root Mean Squared Error (RMSE)** [10][47]
The RMSE is a measure of the Standard Deviation (SD) of the Evaluated deviation [18] and it is calculated using eq (2).

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (X_{obs,i} - X_{model,i})^2}$$ (2)

Where, $$X_{obs}$$ - the actual value and $$X_{model}$$ - the expected value[30].
3) **R-Squared** [11] [31]

R-squared is calculated by calculating the Remaining Sum of Squares (RSS) and dividing by the Total Sum of Squares (TSS) and subtracting from 1. It is computed using eq. (3).

\[
R - Squared = 1 - \frac{RSS}{TSS}
\]  

(3)

### 2.3 Software Effort Estimation Evaluation Techniques

Estimating software effort is mostly carried out using numerous methodologies proposed by researchers over the last few decades. The function point-based model, the constructive cost model (COCOMO), and the Putnam program life cycle model (SLIM) were the first three approaches to estimate the efforts of widely used software, and they describe the estimation of effort using historical data in specific formula [21] [22].

Estimating the efforts of software by using machine learning (ML) has greatly been used, by applying these ML Algorithms, experts spend less time estimating the proposed project and more time on other functions of the software system that satisfies the client. In the last ten years, machine learning has been used by many researchers to measure effort.

A. Najm, et al. in 2012 used the Bayes networks method for estimating program effort and the StepWise Regression (SWR) and Naive Bayes Classifier (NBC) models [20].

E. Khatibi, et al. in 2015 developed a model using ABE and selective classification, the proposed model greatly improves performance measures besides improving accuracy [19].

A. Idri, et al. in 2015 conducted a Software Development Effort Estimation (ASEE) research published in (1990-2012), and revised based on an automated search in 4 electronic databases [9].

P. Rijwani, et al. in 2016 used a multilayer forward neural network to estimate effort using a back-propagation algorithm for training. They discovered that the neural network model provided better estimates than the COCOMOII computational model [14].

P. Pospieszny et al. in 2018 presented three effective ML algorithms (General Linear Model (GLM), Support Vector Machines (SVM), and Multi-Layer Perceptron (MLP)) ensemble averaging to estimate effort. The results of the ensemble model show that they are very accurate when compared to other approaches used by other researchers [12].

M. Hammad, et al. in 2018 used four machine learning algorithms: K-star, ANN, SVM, and linear regression to estimation effort. The results demonstrated that a ML approach is better for software effort with a low MAE value. [16]

S. Mensah, et al. in 2018 classify the effort of 14 datasets into three categories (low, moderate, and high). They used the quantiative density function, and then built regression models for these datasets. To look for statistically significant differences in the prediction performances of the models they used Robust statistical tests [13].

Z. Abdelali, et al. in 2019 used the validation method. They compared the constructed RF model to the regression tree model and across 3 datasets Tukutuku, ISBSG, and COCOMO. Evaluation metrics used are Pred (0.25), MdMRE, and MMRE. The results showed that the RF model gives a better result [15].

M. Kumar, et al. in 2020 applied LR, MLP and RF algorithms with 12 attributes and the results showed that linear regression computes higher estimation results than other ML techniques [18].

Table 2 describes software effort estimation survey using different algorithms, and also describes the data sets used in each paper for estimation, evaluation metrics, the results from papers, and Table 3 describes the Advantages and Disadvantages of algorithms used to get best results in each paper for effort estimation.
Table 2. A survey of Software Effort Estimation using different algorithms

<table>
<thead>
<tr>
<th>Papers</th>
<th>datasets</th>
<th>Algorithm</th>
<th>Evaluation measures</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>P. Pospieszny et al. [12]</td>
<td>ISBSG</td>
<td>Ensemble averaging of three ML algorithms - General Linear model. - Support Vector Machines. - Multi-Layer Perceptron.</td>
<td>- RMSE - MAE - MMER - MSE - MMRE - PRED - MBRE</td>
<td>In ensemble were used multi-layer perceptron (MLP), Support Vector Machines (SVM), and general linear model (GLM) combined for effort estimation. The Ensemble model is better than the Single models [12].</td>
</tr>
<tr>
<td>S. Mensah et al. [13]</td>
<td>PROMISE: Desharais China Cocomo Kemerer Cocomonasa1 Cosmic Maxwell Cocomonasa Kitchenham Industry: php_projects Github: Albrecht Miyazaki Telecom</td>
<td>Regression depended Effort Estimation techniques: - elastic net regression, - ordinary least squares regression, - LASSO regression, - ridge regression, - stepwise regression.</td>
<td>- MAE - BMMRE - Adjusted R²</td>
<td>For estimation they considered 14 datasets, and each dataset was grouped into three classes based on the effort attribute, namely low, medium, and high, which is considered the first output, and then six regression models were applied to predict accuracy, which is considered the second output. Elastic net regression outscored the other algorithms in the comparison. [13]</td>
</tr>
<tr>
<td>A. Idri et al. [9]</td>
<td>Maxwell ISBSG Kemerer Desharais Albrecht Abran COCOMO Telecom</td>
<td>-(ASEE) Analogy Based Software Effort Estimation</td>
<td>- MdMRE - Pred(25) - MMRE.</td>
<td>They compared Analogy Based Software Effort Estimation (ASEE) with 8 ML and Non-ML techniques, including expert judgment, the COCOMO model, Function point analysis, regression, artificial neural network, radial, decision trees, and vector regression support base function. ASEE outperformed eight other techniques and provided greater accuracy based on the evaluation measures used, Pred (25), MdMRE, and MMRE, then accuracy was improved when ML and Non-ML were combined with ASEE [9].</td>
</tr>
<tr>
<td>E. Khatibi, et al. [19]</td>
<td>ISBSG</td>
<td>A hybrid estimation model (Analogy Based Estimation) and a weighting system (Outperforms other algorithms) compared with other algorithms like: - Analogy Based Estimation, - Convolutional Neural Network, -- Classification And Regression</td>
<td>- MRE - MMRE - PRED (25)</td>
<td>The findings of the suggested model, which included 448 software projects, were compared to those produced using various estimating approaches, showing that the proposed model greatly improves the performance criteria [19].</td>
</tr>
</tbody>
</table>
Trees, - Multiple Linear Regression.

P. Rijwani et al. [14] COCOMO II. - Multi layered Feed Forward Artificial neural network - MMRE - MSE


Z. abdelali et al. [15] ISBSG R8, Tukutuku, COCOMO - Regression Tree(RT) - Random Forest(RF) - MdMRE - Pred(25) - MMRE

M. Hammad, et al. [16] PROMISE: Usp05-tf - Kstar, - Artificial Neural Network, - Support Vector Machines, - Linear Regression - MAE

M. Kumar, et al. [18] Desharnais dataset(With all12 attributes) Desharnais dataset(With seven selected attributes) - Linear Regression, - Random Forest, - Multi-layer perceptron. - CC - MAE - RMSE - RRAE - RSE

A. Najm, et al. [20] ISBSG - forward Stepwise Regression (SWR), - Naive Bayes Classifier (NBC) - Pred(25)

Table 3. Advantages and Disadvantages of Algorithms used to estimate software efforts

<table>
<thead>
<tr>
<th>Papers</th>
<th>Algorithm</th>
<th>Advantages</th>
<th>Dis–advantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>P. Pospieszny et al. [12]</td>
<td>Ensemble averaging of three ML algorithms - General Linear model. - Support Vector Machines. - Multi-Layer Perceptron.</td>
<td>It reduces prediction error, is stable, and produces more accurate results.</td>
<td>They need more time in Computation and design.</td>
</tr>
<tr>
<td>S. Mensah et al. [13]</td>
<td>Regression dependent Effort Estimation techniques: - elastic net regression, - ordinary least squares regression,</td>
<td>Elastic net regression is a simple method and more preferred than RR or LASSO regression.</td>
<td>These methods cannot solve NON-Linear problems but can only solve Linear problems and the computational cost is high.</td>
</tr>
<tr>
<td>Authors</td>
<td>Methods/Models</td>
<td>Advantages</td>
<td>Disadvantages</td>
</tr>
<tr>
<td>---------</td>
<td>---------------</td>
<td>------------</td>
<td>--------------</td>
</tr>
<tr>
<td>A. Idri et al. [9]</td>
<td>- (ASEE) Analogy Based Software Effort Estimation.</td>
<td>It is similar to human thinking.</td>
<td>It has an inability to properly deal with categorical attributes.</td>
</tr>
<tr>
<td>E. Khatibi, et al. [19]</td>
<td>A hybrid estimation model (Analogy Based Estimation) and a weighting system (Outperforms other algorithms) compared with other algorithms like: - Analogy Based Estimation, -Convolutional Neural Network, --Classification And Regression Trees, - Multiple Linear Regression.</td>
<td>Hybrid methods combine multiple models into a best-assembled model, that improve the performance metrics.</td>
<td>Hybrid approaches are computationally more expensive. The proposed model is not appropriate for data sets with a nonlinear relationship between effort and independent project variables.</td>
</tr>
<tr>
<td>P. Rijwani et al. [14]</td>
<td>- Multi layered Feed Forward Artificial neural network</td>
<td>Suitable for complex data set and ANN learn from previous data, as well as suitable for both Linear and non-Linear functions, so the voltage prediction results are high.</td>
<td>The problems are overfitting and have slow convergence speed.</td>
</tr>
<tr>
<td>B. Marapelli [4]</td>
<td>- K-nearest Neigh, -Linear Regression (LR).</td>
<td>Fast, and it is the simplest way to find relationship between variables, and easy to implement.</td>
<td>This method is able to give a relation between just the linear dependent and independent variables.</td>
</tr>
<tr>
<td>Z. Abdelali et al. [15]</td>
<td>- Regression Tree (RT) - Random Forest (RF)</td>
<td>It is robust against overfitting, easy to use, and can handle huge data sets.</td>
<td>It uses black box method. It is time consuming and complicated.</td>
</tr>
<tr>
<td>M. Hammad, et al. [16]</td>
<td>- K-star, -Artificial Neural Network, -Support Vector Machines, - Linear Regression</td>
<td>It can be applied to unstructured or semi-structured data, and it works even better with a lot of features.</td>
<td>It takes more time to predict in larger datasets.</td>
</tr>
<tr>
<td>M. Kumar, et al. [18]</td>
<td>- Linear Regression, -Random Forest, -Multi-layer perceptron.</td>
<td>It is a simple algorithm to finding the relationship between many variables that will have designed Many applications with less size or effort, which will reduce the complexity of the software.</td>
<td>This algorithm is able to give a linear relationship between the dependent and independent variables.</td>
</tr>
<tr>
<td>A. Najm, et al.[20]</td>
<td>- forward Stepwise Regression (SWR), - Naive Bayes Classifier (NBC)</td>
<td>Naive Bayes is a simple algorithm to implement; it produces better results when the input variables are independent, and the ability to integrate data with expert knowledge.</td>
<td>This method is always based on the assumption that Input variables are independent and this is not always true.</td>
</tr>
</tbody>
</table>
3. Results and Discussion
Table 4 displays the prediction accuracy results of many machine learning techniques used Albrecht dataset with describing the performance metrics MAE, RMSE and R-Squared for ML algorithms. According to the interpretation from Table 4, the Random Forest algorithm produces better results with lower MAE, RMSE, and higher value of R-Squared.

Table 4. Estimated effort performance metrics of machine learning Algorithms using the Albrecht_dataset[11]

<table>
<thead>
<tr>
<th>Algorithms names used</th>
<th>MAE</th>
<th>RMSE</th>
<th>R-squared</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Forest</td>
<td>0.1940703</td>
<td>0.2273109</td>
<td>0.4732279</td>
</tr>
<tr>
<td>Support vector machines</td>
<td>0.271210</td>
<td>0.291869</td>
<td>0.1315232</td>
</tr>
<tr>
<td>Decision Tree</td>
<td>0.2299442</td>
<td>0.3140069</td>
<td>0.0052189</td>
</tr>
<tr>
<td>Neuralnet</td>
<td>0.2208348</td>
<td>0.2676081</td>
<td>0.2699029</td>
</tr>
<tr>
<td>Ridge</td>
<td>0.2495593</td>
<td>0.274339</td>
<td>0.232714</td>
</tr>
<tr>
<td>LASSO</td>
<td>0.2672776</td>
<td>0.2952834</td>
<td>0.1110852</td>
</tr>
<tr>
<td>ElasticNet</td>
<td>0.2522743</td>
<td>0.2771694</td>
<td>0.2168</td>
</tr>
<tr>
<td>Deepnet</td>
<td>0.2702398</td>
<td>0.3222748</td>
<td>-0.0588508</td>
</tr>
</tbody>
</table>

Table 5 shows performance metrics for ML algorithms that used China dataset with describing the performance metrics MAE, RMSE and R-Squared for ML algorithms. According to the interpretation from Table 5, the LASSO algorithm produces better results with lower MAE, RMSE, and higher value of R-Squared.


<table>
<thead>
<tr>
<th>Algorithms names used</th>
<th>MAE</th>
<th>RMSE</th>
<th>R-squared</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Forest</td>
<td>0.03832486</td>
<td>0.0651549</td>
<td>0.8028527</td>
</tr>
<tr>
<td>Support vector Machines</td>
<td>0.04872961</td>
<td>0.109964</td>
<td>0.4384377</td>
</tr>
<tr>
<td>Decision Tree</td>
<td>0.0222997</td>
<td>0.05380968</td>
<td>0.8655325</td>
</tr>
<tr>
<td>Neuralnet</td>
<td>0.01775594</td>
<td>0.0439852</td>
<td>0.9101517</td>
</tr>
<tr>
<td>Ridge</td>
<td>0.01977087</td>
<td>0.03703651</td>
<td>0.9362975</td>
</tr>
<tr>
<td>LASSO</td>
<td>0.01344521</td>
<td>0.02381411</td>
<td>0.9736631</td>
</tr>
<tr>
<td>ElasticNet</td>
<td>0.01406866</td>
<td>0.02462756</td>
<td>0.9718331</td>
</tr>
<tr>
<td>Deepnet</td>
<td>0.09730134</td>
<td>0.1497018</td>
<td>-0.0407606</td>
</tr>
</tbody>
</table>

Table 6 shows performance metrics for ML algorithms used in Desharnais dataset, and representation of MAE, R-Squared, and RMSE for ML algorithms for the Desharnais dataset. According to Table 6, the results differ because the Neuralnet algorithm produces lower MAE, the Ridge algorithm produces lower RMSE, and the LASSO algorithm produces higher value of R-Squared.

Table 6 Estimated effort performance metrics of machine learning Algorithms using the Desharnais_dataset[11]

<table>
<thead>
<tr>
<th>Algorithms names used</th>
<th>MAE</th>
<th>RMSE</th>
<th>R-squared</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Forest</td>
<td>0.1161946</td>
<td>0.1744573</td>
<td>0.38059</td>
</tr>
<tr>
<td>Support Vector machines</td>
<td>0.1052762</td>
<td>0.1993475</td>
<td>0.1912367</td>
</tr>
<tr>
<td>Decision Tree</td>
<td>0.1067227</td>
<td>0.1724028</td>
<td>0.3950932</td>
</tr>
<tr>
<td>Neuralnet</td>
<td>0.08394905</td>
<td>0.1508566</td>
<td>0.3950932</td>
</tr>
<tr>
<td>Ridge</td>
<td>0.08810373</td>
<td>0.1482106</td>
<td>0.5529472</td>
</tr>
<tr>
<td>LASSO</td>
<td>0.08731722</td>
<td>0.1455049</td>
<td>0.5691211</td>
</tr>
<tr>
<td>ElasticNet</td>
<td>0.08874032</td>
<td>0.1463786</td>
<td>0.563931</td>
</tr>
<tr>
<td>Deepnet</td>
<td>0.1913845</td>
<td>0.2396687</td>
<td>-0.169022</td>
</tr>
</tbody>
</table>

Table 7 represents the performance indicators for ML methods when using Kemerer dataset. Table 7 also represents MAE, R-Squared, and RMSE for ML approaches. According to Table 7, the findings are different because the LASSO method produces lower MAE, while the Random Forest technique produces lower RMSE and greater value of R-Squared.

<table>
<thead>
<tr>
<th>Algorithms names used</th>
<th>MAE</th>
<th>RMSE</th>
<th>R-squared</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Forest</td>
<td>0.2076936</td>
<td>0.2357751</td>
<td>0.6032962</td>
</tr>
<tr>
<td>Support vector machines</td>
<td>0.219517</td>
<td>0.2635159</td>
<td>0.5044536</td>
</tr>
<tr>
<td>Decision Tree</td>
<td>0.3709271</td>
<td>0.3954624</td>
<td>-0.1160428</td>
</tr>
<tr>
<td>Neuralnet</td>
<td>0.2179616</td>
<td>0.3219353</td>
<td>0.2603814</td>
</tr>
<tr>
<td>Ridge</td>
<td>0.1971335</td>
<td>0.2810334</td>
<td>0.4363801</td>
</tr>
<tr>
<td>LASSO</td>
<td>0.183967</td>
<td>0.256731</td>
<td>0.5296434</td>
</tr>
<tr>
<td>ElasticNet</td>
<td>0.1890074</td>
<td>0.26093</td>
<td>0.5141316</td>
</tr>
<tr>
<td>Deepnet</td>
<td>0.4011895</td>
<td>0.4142278</td>
<td>-0.2244724</td>
</tr>
</tbody>
</table>

Table 8 describes the performance metrics for ML methods using the Maxwell dataset as an example. For the Maxwell dataset, Table 8 shows MAE, R-Squared, and RMSE vs ML approaches. According to Table 8, the LASSO technique gives better outcomes by producing lower values of MAE, RMSE, and a greater value of R-Squared.

Table 8 Estimated effort performance metrics of machine Learning Algorithms using the Maxwell_dataset[11]

<table>
<thead>
<tr>
<th>Algorithms names used</th>
<th>MAE</th>
<th>RMSE</th>
<th>R-squared</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random forest</td>
<td>0.2330224</td>
<td>0.3098055</td>
<td>0.109088</td>
</tr>
<tr>
<td>Support vector machines</td>
<td>0.311288</td>
<td>0.4006657</td>
<td>-0.4901189</td>
</tr>
<tr>
<td>Decision Tree</td>
<td>0.2896955</td>
<td>0.3897197</td>
<td>-0.4098123</td>
</tr>
<tr>
<td>Neuralnet</td>
<td>0.2245069</td>
<td>0.2914014</td>
<td>0.2117937</td>
</tr>
<tr>
<td>Ridge</td>
<td>0.2162983</td>
<td>0.2901234</td>
<td>0.2186921</td>
</tr>
<tr>
<td>LASSO</td>
<td>0.2106617</td>
<td>0.2859893</td>
<td>0.2407999</td>
</tr>
<tr>
<td>ElasticNet</td>
<td>0.211274</td>
<td>0.2865395</td>
<td>0.2378761</td>
</tr>
<tr>
<td>Deepnet</td>
<td>0.2901784</td>
<td>0.3571974</td>
<td>-0.1843314</td>
</tr>
</tbody>
</table>

Table 9 demonstrates performance metrics for ML algorithms using a Kitchenham dataset. Table 9 displays MAE, R-Squared, and RMSE vs ML approaches. According to the interpretation of Table 9, the LASSO method generates better results because it produces lower MAE, whereas the Neuralnet algorithm produces better results because it produces lower RMSE and greater value of R-Squared.


<table>
<thead>
<tr>
<th>Algorithms names used</th>
<th>MAE</th>
<th>RMSE</th>
<th>R-squared</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random forest</td>
<td>0.1047124</td>
<td>0.1715596</td>
<td>0.3928102</td>
</tr>
<tr>
<td>Support Vector machines</td>
<td>0.1189372</td>
<td>0.4006657</td>
<td>-0.4901189</td>
</tr>
<tr>
<td>Decision Tree</td>
<td>0.1174008</td>
<td>0.2914014</td>
<td>0.2117937</td>
</tr>
<tr>
<td>Neuralnet</td>
<td>0.03921013</td>
<td>0.0739</td>
<td>0.8873365</td>
</tr>
<tr>
<td>Ridge</td>
<td>0.04342892</td>
<td>0.0877456</td>
<td>0.8411641</td>
</tr>
<tr>
<td>LASSO</td>
<td>0.03876806</td>
<td>0.0740225</td>
<td>0.8869626</td>
</tr>
<tr>
<td>ElasticNet</td>
<td>0.03961122</td>
<td>0.07542022</td>
<td>0.8826536</td>
</tr>
<tr>
<td>Deepnet</td>
<td>0.1400501</td>
<td>0.2242306</td>
<td>-0.037252</td>
</tr>
</tbody>
</table>

Table 10 shows performance metrics for ML methods using the Cocomo81dataset as an example. For the Cocomo81 dataset, Table 8 shows a representation of MAE, R-Squared, and RMSE vs ML approaches. According to Table 10, the Neuralnet algorithm offers better results, with lower values of MAE, RMSE, and a greater value of R-Squared.
Table 10: Estimated effort performance metrics of machine learning Algorithms using the Cocomo81 dataset [11]

<table>
<thead>
<tr>
<th>Algorithms names used</th>
<th>Dataset 7-Cocomo81</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAE</td>
</tr>
<tr>
<td>Random forest</td>
<td>0.06047924</td>
</tr>
<tr>
<td>Support vector machines</td>
<td>0.07500071</td>
</tr>
<tr>
<td>Decision Tree</td>
<td>0.08838886</td>
</tr>
<tr>
<td>Neuralnet</td>
<td>0.05387155</td>
</tr>
<tr>
<td>Ridge</td>
<td>0.07912894</td>
</tr>
<tr>
<td>LASSO</td>
<td>0.08097285</td>
</tr>
<tr>
<td>ElasticNet</td>
<td>0.07980254</td>
</tr>
<tr>
<td>Deepnet</td>
<td>0.2548906</td>
</tr>
</tbody>
</table>

Table 11 shows the ML algorithms that get the best results for each dataset’s performance measures. Figure 2 shows a visualization of the best findings for the MAE, RMSE, and R-Squared for each data set, using ML methods. According to Table 11, the LASSO algorithm gives better results in the China data set when compared to other ML algorithms and datasets because it produces lower MAE, RMSE, and a greater value of R-Squared.

Table 11: Estimated effort performance metrics of best machine learning algorithms using different dataset[11]

<table>
<thead>
<tr>
<th>Dataset used</th>
<th>machine learning Algorithms</th>
<th>MAE</th>
<th>RMSE</th>
<th>R-squared</th>
</tr>
</thead>
<tbody>
<tr>
<td>Albrecht</td>
<td>Random Forest</td>
<td>0.1940703</td>
<td>0.2273109</td>
<td>0.4732279</td>
</tr>
<tr>
<td>China</td>
<td>LASSO</td>
<td>0.01344521</td>
<td>0.02381411</td>
<td>0.9736631</td>
</tr>
<tr>
<td>Desharnais</td>
<td>Neuralnet</td>
<td>0.08394905</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Ridge</td>
<td>0.1482106</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>LASSO</td>
<td></td>
<td></td>
<td>0.5691211</td>
</tr>
<tr>
<td>Kemerer</td>
<td>Random Forest</td>
<td>0.2357751</td>
<td>0.6032962</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LASSO</td>
<td>0.183967</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Maxwell</td>
<td>LASSO</td>
<td>0.2106617</td>
<td>0.2859893</td>
<td>0.2407999</td>
</tr>
<tr>
<td>Kitchenham</td>
<td>Neuralnet</td>
<td>0.0739</td>
<td>0.8873365</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LASSO</td>
<td>0.03876806</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cocomo81</td>
<td>Neuralnet</td>
<td>0.05387155</td>
<td>0.09447618</td>
<td>0.8212226</td>
</tr>
</tbody>
</table>

4. Conclusion

Several machine learning algorithms, such as Support Vector Machines, Random Forest, LASSO, Neuralnet, Decision Tree, Ridge, ElasticNet, and Deepnet, are studied and compared in this paper using the Kitchenham, China, Maxwell, Albrecht, Kemerer, Desharnais, and Cocomo81 datasets for Software Effort Estimate (SEE) predicting the amount of time it takes to develop software in terms of (Person _ hour, or Person _ month). Based on a comparison of multiple ML methods, it was discovered that the Random Forest technique outperforms all other measures in the Albrecht dataset, and in
Desharnais, Neuralnet algorithm produce lower values for MAE, where Ridge algorithm provides lower values of RMSE, and LASSO algorithm produces higher value of R-Squared, in Kemerer different as LASSO algorithm produces lower values for MAE, and Random Forest algorithm produces lower values for RMSE, and produces higher R-Squared value. In Maxwell, the case is different as LASSO algorithm produces lower values for MAE, and Random Forest algorithm produces lower values of RMSE, and produces higher R-S, In Kitchenham, the LASSO algorithm provides better results because it produces lower values of MAE, and the Neuralnet algorithm provides better results because it produces lower values of RMSE and produces higher value of R-Squared, and the Neuralnet algorithm provides better results in Cocomo81 because it produces lower values of RMSE, MAE and produces higher value of R-Squared, where the LASSO algorithms in the China dataset outperforms others. Evaluation metrics considered are MAE, RMSE and R-Squared.

References


[23] "Advances in Machine Learning and Computational Intelligence", Springer Science and Business Media LLC, 2021


