Journal of Information Systems Engineering and Management
2024, 9(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Advanced Machine Learning Software Cost Prediction Model using AdaBoost and COCOMO Cost Parameters

Manas Prasad Rout[1*], Sabyasachi Patnaik[2], Umashankar Ghugar[3], Pradeep Kumar Shriwas[4]

[1,2]*Department of Computer Science FM University, Balasore, Odisha mprout@ravenshawuniversity.ac.in spattnaik1965@rediffmail.com*

[3]*Department of Computer Science and Engineering Chandigarh University, Mohali, Punjab, India ughugar@gmail.com*

[4]*Department of Computer Science and Engineering OP Jindal University, Raigarh, India Pradeep.shriwas@opju.ac.in*

ARTICLE INFO

Received: 22 Oct 2024

Revised: 26 Nov 2024

Accepted: 20 Dec 2024

ABSTRACT

Playing a pivotal role in software development, the Constructive Cost Model (COCOMO) offers a systematic and structured approach to cost estimation. It stands as a widely utilized model, aiding project managers in estimating the required effort, time, and cost for software development projects. COCOMO takes into consideration diverse factors, including the project's size, complexity, and the experience of the development team. The utilization of COCOMO empowers software development teams to make informed decisions related to resource allocation, project scheduling, and budgeting. Its application extends to managing expectations, enhancing project planning, and mitigating the risk of cost overruns. The integration of machine learning assumes a critical role in advancing cost estimation within the realm of software development, specifically through COCOMO. Through the utilization of machine learning algorithms, COCOMO gains the capability to analyze and interpret extensive datasets, taking into account numerous complex factors that influence project costs. This work aims to propose a model for software cost estimation using an advanced machine learning technique, i.e. Adaptive boosting, which has improved accuracy, reduced overfitting, effectiveness with imbalanced data, and good generalization capabilities. The proposed work may contribute to the success of software projects, providing a reliable and comprehensive framework for cost estimation.

Keywords: COCOMO, Project cost prediction, Adaptive boosting, Machine learning, Ensemble learning.

INTRODUCTION

Determining the costs of software [1] stands as a crucial aspect within project management, focusing on predicting the resources, time, and financial investment required for software development. This procedure involves activities such as defining project scope, assessing size and complexity, choosing development methodologies, evaluating team expertise, considering the technology stack and external dependencies, and appraising potential risks. Widely employed estimation methods, including expert judgment, analogous estimation, parametric estimation, and three- point estimation, play a significant role. The accuracy of cost estimation holds utmost importance for effective project planning, budgeting, and decision-making. Continuous refinement and adjustments throughout the project lifecycle are essential to adapt to the dynamic nature of software development, ultimately contributing to the realization of successful project outcomes. Software cost estimation serves as a pivotal element in project management, assisting organizations in efficiently allocating resources and making well-informed decisions. Various methods exist to estimate software development costs, each with its strengths and weaknesses. One common approach involves using the knowledge of individuals experienced in software development. These experts examine project requirements, team capabilities, and past data to provide an informed estimate. Though subjective, this method works well when experts have relevant experience and a deep understanding of the project. Another method, called analogous estimation or top-down estimation, involves comparing the ongoing project with past similar ones. By looking at historical data, including cost and effort, teams can predict the outcomes of the new project. This method is quick and straightforward but

assumes the current project is comparable to previous ones. Parametric estimation uses

mathematical models and statistical relationships to estimate costs based on project parameters like size, complexity, and productivity metrics. This data-driven approach is especially handy for large projects with well-defined parameters. COCOMO, a widely used model, categorizes projects by size and complexity, providing equations to estimate effort, duration, and cost. Its flexibility in various development environments allows adjustments based on project-specific features. The Program Evaluation and Review Technique (PERT) estimates three scenarios: “optimistic”, “pessimistic”, and “most likely”, for each task. Using a weighted average, PERT calculates the expected duration or cost, proving useful in handling uncertainties and risks in project estimates. The choice of the most suitable software cost estimation approach depends on the project's characteristics, available data, and organizational preferences. Often, a combination of these methods is used to achieve more accurate and reliable estimates.

Software cost estimation [2] holds great importance in project management and software development. Accurate cost estimation is crucial for effective budget planning, helping organizations allocate resources efficiently and avoid financial constraints. It contributes to project planning by establishing realistic timelines, milestones, and goals. Identifying potential risks through cost estimation aids in risk management, enabling the development of mitigation strategies. Precise cost estimates are essential for informed decision-making, assisting stakeholders in strategic choices regarding project scope and timelines. Additionally, cost estimation is indispensable for managing client expectations, facilitating competitive bidding, optimizing resource utilization, and serving as a reference point for monitoring and controlling project expenses. Ultimately, the accuracy of cost estimates is a key factor in assessing project success, reflecting the organization's proficiency in effective planning and resource management. Estimating the cost of software development comes with many challenges [3] that impact the accuracy and reliability of forecasts in project management. The difficulties arise from the inherent complexity of software development, uncertainties about project scope, the evolving nature of requirements, dynamic technologies, and the diverse expertise within project teams. The complexity increases with the choice of estimation techniques and methodologies, along with the difficulty of measuring intangible factors. External dependencies, like third-party integrations, and unforeseen risks also introduce uncertainties. The iterative development processes, combined with changing client expectations, add to the challenges of cost estimation. To effectively overcome these challenges, a combination of experienced judgment, continuous refinement, and adaptable approaches throughout the project lifecycle is essential. Using advanced algorithms, machine learning transforms software cost estimation [4], aiming to enhance the accuracy and efficiency of projecting project costs. By analyzing historical project data, machine learning algorithms detect patterns and relationships, enabling tailored predictions for the unique characteristics of the current project. This adaptive approach allows the system to improve its estimations over time. To address the complexities of software development, machine learning considers a wide range of variables, including project scope, team expertise, and external dependencies. The integration of machine learning into software cost estimation not only improves prediction precision but also contributes to more informed decision-making, ultimately enhancing project planning and resource allocation in the dynamic landscape of software development.

Several studies have investigated the use of machine learning (ML) in software cost estimation. In article [12], the focus is on applying ML techniques to improve the accuracy and reliability of software project estimation. The authors aim to enhance estimation by using advanced computational models, addressing challenges associated with traditional methods. The study suggests integrating ML algorithms to predict project outcomes more effectively. Similarly, other studies [13-15] also discuss the application of ML in software cost estimation. Ensemble learning offers a significant advantage in software cost estimation by enhancing prediction accuracy and robustness. In the intricate landscape of software development, where projects have diverse and dynamic characteristics, ensemble methods excel by combining the strengths of multiple models. This combination not only reduces the risk of overfitting but also ensures a more stable estimation process, especially when dealing with outliers or unpredictable factors. The ability to include diverse models within an ensemble contributes to improved generalization, allowing for a more nuanced understanding of complex relationships inherent in software projects. Furthermore, ensemble learning is valuable in mitigating risks associated with uncertainties,

Journal of Information Systems Engineering and Management
2024, 9(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

providing project managers with resilient cost estimations. The flexibility in model selection offered by ensemble methods empowers practitioners to tailor their approach to the specific nuances of each software development endeavor, fostering adaptability in an ever-evolving industry. In essence, the advantage of using ensemble learning in software cost estimation lies in its ability to harness collective intelligence, resulting in more accurate and informed decision-making for successful project management.

The remaining sections are organized as follows: Methodology, Result Analysis, and Conclusion in Sections 2, 3, and 4, respectively.

OBJECTIVES

This study utilizes the Adaptive Boosting ensemble learning approach to predict software cost estimation based on historical data related to COCOMO cost attributes. The primary objectives are as follows:

(i) Predicting software cost using various COCOMO software cost attributes such as programmers' capability (pcap), use of software tools (tool), process complexity (cplx), etc.

(ii) Employing a robust machine learning model, Adaptive Boosting, for the prediction of software cost.

2. ADAPTIVE BOOSTING

AdaBoost [5], or Adaptive Boosting (Fig. 1), stands out as a potent ensemble learning technique widely recognized in machine learning. AdaBoost's core strength lies in its ability to boost the accuracy of weak learners, which are simple models performing slightly better than random chances. In tasks like classification and regression, AdaBoost trains a sequence of weak learners on the same dataset, giving more weight to misclassified instances in each round. This step-by-step process allows AdaBoost to focus on previously misclassified data points, effectively improving the model's overall performance. An essential feature of AdaBoost is its adaptability, dynamically adjusting weights for misclassified instances to prioritize the most challenging samples. The final prediction combines the weak learners through a weighted approach, where those with higher accuracy play a more significant role. This adaptability and ensemble strategy make AdaBoost robust against overfitting, enabling it to handle complex relationships within the data effectively. AdaBoost's versatility makes it a valuable asset in machine learning, contributing to enhanced predictive modelling and decision-making across various domains.

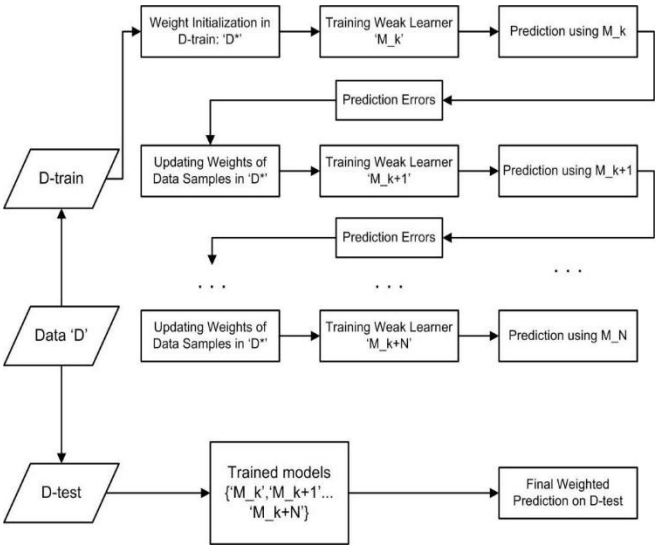

Figure 1. Overview of the working of AdaBoost

3. PROPOSED APPROACH METHODS

The proposed method makes use of an AdaBoost ensemble learning model [5] for the prediction of software cost. Here, the COCOMO81 dataset [7, 8] has been used for model construction, which includes various COCOMO software cost attributes are considered as input attributes and actual cost as output attribute. The used dataset is

Journal of Information Systems Engineering and Management
2024, 9(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

divided into train and test data. Initially, the data samples of the train data are set with same weights. The predefined number of decision trees is constructed out of the data samples from train data. Each decision tree is trained and used for prediction of software cost. Then, the weights of the data samples in training data are adjusted based on the prediction error. Thereafter, another decision tree is constructed from the train data with updated weights. By this way, required number of decision trees are constructed from the train data, and then the entire trained decision trees [6] are used to make prediction on test data. The entire process can be understood from Algorithm 1 and Algorithm 2.

Algorithm 1: AdaBoost for the prediction of software cost by using COCOMO cost parameters
Let C be the COCOMO dataset, which are the collection of data sample with set of values of input attributes (COCOMO cost parameters) and target (actual cost). Split into train set C_{train} and test set C_{test}.
1.Assign equal weights to all training samples in C_{train}. If there are N examples, each weight is set to 1/N initially.
2. Set the number of estimator T
3. For t = 1 to T:
i. Train Weak Decision Tree Regressor (DTRt) (e.g., a decision stump for regression) (Algorithm 2) on the training data C_{train} with the current weights.
ii. Calculate the weighted absolute error of the DTR_t. This is the sum of weights multiplied by the absolute difference between predicted and true values.
iii. Compute the weight of the DTR_t in the final model. It depends on the error, with lower errors receiving higher weight.
iv. Update Weights of data samples in C_{train} by Increasing the weights of examples with higher absolute errors and decrease the weights of examples with lower errors.
4. Make Final Prediction by Combining all the regressors (DTR_t, for t ☐ 1 to T) into a strong regressor by assigning weights based on their individual performance. The final model is a weighted sum of all regressor (DTR_t, for t ☐ 1 to T).

Algorithm 2: Construction of Decision tree regressor (DTR)
1.Select the Best Split by Identify the best feature in C_{train} and corresponding threshold that minimizes the mean squared error (MSE) or another chosen criterion. Here, splitting is done by using information gain.
2. If a stopping criterion is met (e.g., maximum depth reached or minimum samples in a node), create a leaf node with the predicted value for the target variable.
3. Partition the data into two subsets based on the selected feature and threshold.
4. Recursively Repeat: Apply steps 1-3 recursively to each subset until the stopping criteria are met for each branch.
5. Make Predictions by assigning the predicted value to each sample based on the leaf node reached in the tree traversal.
6. Return MSE.

RESULTS

4. EXPERIMENTAL RESULT AND ANALYSIS

The evaluation of the software cost estimation prediction approach utilizes the COCOMO81 dataset [7, 8]. The training data for the model encompasses diverse COCOMO software cost attributes, such as "Acap" (analysts capability), "Cplx" (process complexity), "Rely" (required software reliability), etc., with the target attribute being "Actual" (Actual cost). Subsequently, the trained model is employed to predict software costs. A comparative analysis of the prediction performance involves various machine learning models (refer to Table 1), including

Journal of Information Systems Engineering and Management
2024, 9(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Linear Regression (LR) [9], Stochastic Gradient Descent (SGD) [10], and Support Vector Regressor (SVR) [11]. Table 1 presents the comparison of the studied model in terms of Mean Absolute Error (MAE) (Eq.1) and Mean Squared Error (MSE) (Eq.2).

Table 1. Software cost prediction performance indicators

Prediction Model	**Performance Metrics**	
	MAE	**MSE**
LR	923.8304	4477571.21
SGD	920.8115	2746412.67
SVR	809.5724	6967283.14
Regression tree [12]	943.3075	5193249
Random forest [12]	928.3318	5769025
Bagging	259.98	691718.66
Proposed model (Adaptive boosting)	176.35	42950.90

DISCUSSION

In Eq.1, ac is actual cost, ac is predicted cost, and n is the number of prediction.

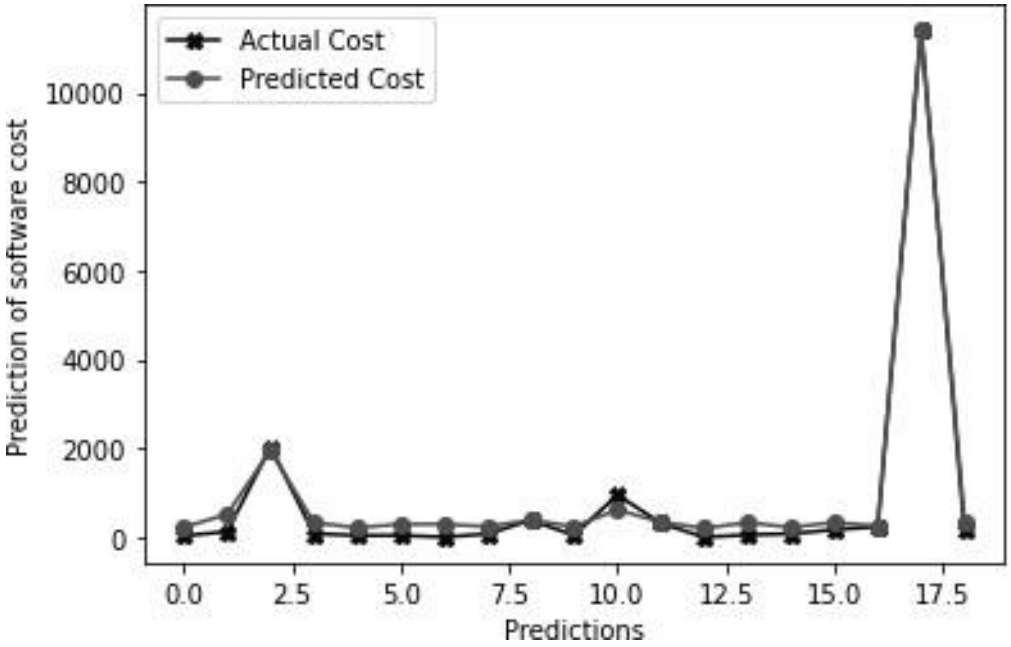

Figure 2. Performance of the proposed approach for the prediction of software cost

In Eq.2, ac is the actual cost, ac is the predicted cost, and n is the number of predictions.

The software cost prediction, as per the proposed approach, is illustrated in Figure 2. The predictive efficacy of the proposed method is assessed against other approaches (Fig. 3) based on MSE, demonstrating its superiority over alternative methods.

Journal of Information Systems Engineering and Management
2024, 9(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

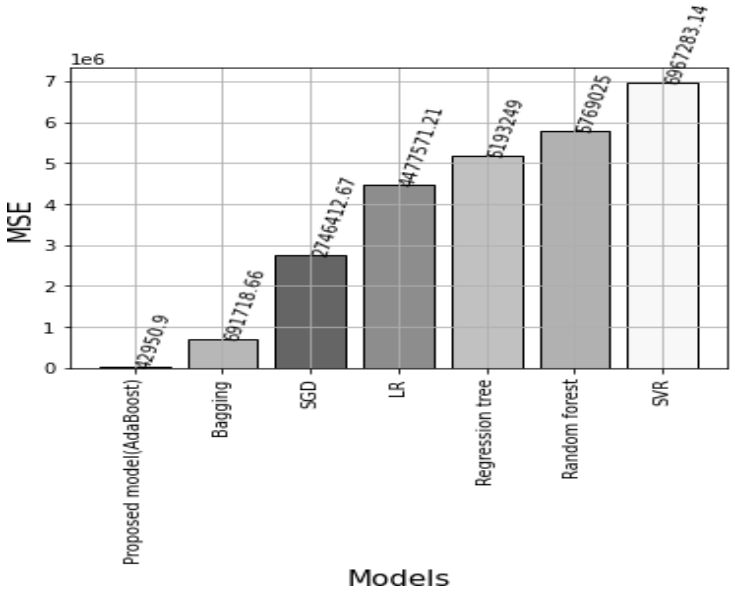

Figure 3. Comparison of all models based on MSE (Bar graph)

5. CONCLUSION

In this work, an efficient ensemble learning model, AdaBoost, has been used for prediction of software cost using COCOMO parameters. The proposed work is compared with other machine learning models and found to be of acceptable performance. However, software cost prediction poses several challenges that organizations must navigate to achieve accurate estimates and effective project management. One significant challenge lies in the inherent complexity of software development projects, which often involve numerous variables and dependencies that are difficult to quantify accurately. Uncertainties in requirements, evolving technology, and changing project scope further complicate the prediction process. Although, the COCOMO is widely used for software cost estimation, it is not without its challenges. One notable difficulty is the sensitivity of COCOMO to accurate input values, such as the size and complexity of the software project. Small errors in these inputs can lead to significant variations in cost predictions. Additionally, COCOMO's reliance on historical data assumes a certain level of similarity between past and current projects, which may not always be the case in rapidly evolving technological landscapes. While COCOMO provides a valuable framework for cost estimation, addressing these challenges requires careful consideration of project specifics, continuous refinement of input parameters, and recognition of the evolving nature of contemporary software development practices. Although machine learning models have shown promise in enhancing accuracy, they are highly dependent on the quality and representativeness of the training data. The availability of comprehensive and diverse datasets for training ML models that align with the specific characteristics of software development projects can be a major hurdle. COCOMO's reliance on historical data assumes certain patterns and trends, but machine learning models may struggle to capture complex relationships in evolving software environments. Addressing these challenges requires careful consideration of data quality, the dynamic nature of software development, and adaptation in the integration of machine learning with COCOMO for software cost prediction.

REFRENCES

[1] Leung, Hareton, and Zhang Fan. "Software cost estimation." Handbook of Software Engineering and Knowledge Engineering: Volume II: Emerging Technologies. 2002. 307-324.
[2] Peixoto, Carlos Eduardo Lima, Jorge Luis Nicolas Audy, and Rafael Prikladnicki. "The importance of the use of an estimation process." Proceedings of the 2010 ICSE Workshop on Software Development Governance. 2010.
[3] Torp, Olav, and Ole Jonny Klakegg. "Challenges in cost estimation under uncertainty—A case study of the decommissioning of Barsebäck Nuclear Power Plant." Administrative sciences 6.4 (2016): 14.

Research Article

[4] Huang, Jianglin, Yan-Fu Li, and Min Xie. "An empirical analysis of data preprocessing for machine learning-based software cost estimation." Information and software Technology 67 (2015): 108-127.
[5] J. Zhu, H. Zou, S. Rosset, T. Hastie, “Multi-class adaboost.” Statistics and its Interface 2.3 (2009): 349-360.
[6] L. Breiman, J. Friedman, R. Olshen, and C. Stone, “Classification and Regression Trees”, Wadsworth, Belmont, CA, 1984.
[7] Boehm, Barry W. Software engineering economics. Springer Berlin Heidelberg, 2002.
[8] http://promise.site.uottawa.ca/SERepository/datasets/cocomo81.arff
[9] Cai, T. Tony, and Peter Hall. "Prediction in functional linear regression." (2006): 2159-2179.
[10] Bottou, Léon. "Large-scale machine learning with stochastic gradient descent." Proceedings of COMPSTAT'2010: 19th International Conference on Computational Statistics Paris France, August 22-27, 2010 Keynote, Invited and Contributed Papers. Physica-Verlag HD, 2010.
[11] Chang, Chih-Chung, and Chih-Jen Lin. "LIBSVM: a library for support vector machines." ACM transactions on intelligent systems and technology (TIST) 2.3 (2011): 1-27.
[12] Zakaria, Noor Azura, et al. "Software project estimation with machine learning." International Journal of Advanced Computer Science and Applications 12.6 (2021).
[13] Baskeles, Bilge, Burak Turhan, and Ayse Bener. "Software effort estimation using machine learning methods." 2007 22nd international symposium on computer and information sciences. IEEE, 2007.
[14] BaniMustafa, Ahmed. "Predicting software effort estimation using machine learning techniques." 2018 8th International Conference on Computer Science and Information Technology (CSIT). IEEE, 2018.
[15] Tayyab, Muhammad Raza, Muhammad Usman, and Waseem Ahmad. "A machine learning based model for software cost estimation." Proceedings of SAI Intelligent Systems Conference (IntelliSys) 2016: Volume 2. Springer International Publishing, 2018.