International Journal of Scientific Research and Modern Technology (IJSRMT) ijsrmt.com
Volume 4, Issue 11, 2025
DOI: https://doi.org/10.38124/ijsrmt.v4i11.949

Enhancing Software Effort Estimation in Agile Projects Through Expert-Guided Feature Selection and Machine Learning

Ali Bashir[1]; Maryam Zakir[2]; Yusra Bajwa[3]

[1]Comsats University Islamabad
[2]National Textile University

Publishing Date: 2025/11/13

Abstract

This study proposes a hybrid methodology that integrates expert judgment and machine learning techniques to improve software effort estimation in Agile projects. A comprehensive survey of 45 software professionals identified key influencing attributes, which were further refined using recursive feature elimination (RFE). The performance of three machine learning models—Linear Regression, Multilayer Perceptron (MLP), and Support Vector Regression (SVR)—was evaluated. Results showed that incorporating expert-informed features significantly improved the accuracy of predictions, with Linear Regression achieving the best results ($R^2 = 0.9319$, MAE = 1726).

I. INTRODUCTION

The precise estimation of development effort serves as the fundamental factor for project success because it affects traditional decisions regarding planning and budgeting and allocating resources [1]. The fundamental nature of iterative development and evolving requirements throughout Agile Software Development (ASD) presents a substantial challenge for obtaining reliable estimates which proves critical [2], [3] due to the mentioned characteristics [4], [5]. The simplicity and natural appeal of standard estimation techniques including expert judgment and story points and analogy-based methods continues as their primary practice because they remain easy to utilize [6]. Project effort assessments based on these methods show irregular results while depending heavily on individual perceptions and contain multiple bias factors that result in inaccurate effort predictions [7].

Research teams use both algorithmic and data-based methods to study software development effort including parametric models such as COCOMO [8] and modern machine learning models including Linear Regression (LR), Support Vector Regression (SVR), and Neural Networks (NNs). These approaches show better effort estimation results through their ability to detect sophisticated non-linear interrelationships between project variables and effort measurement factors [9]. Machine learning models remain underutilized in industry because they present three main implementation barriers which include their black-box nature and weak domain transparency and their dependence on high-quality data. Machine learning systems typically lack the ability to incorporate expert human knowledge about complex project situations especially when tasks involve either new or unclear conditions according to [10], [11].

Agile development makes estimation more challenging through its recurring development cycles as well as fast changing requirements and collaborative estimation approaches [12]. Agile practices incorporate Planning Poker and Team Estimation Game to develop team consensus but they demonstrate inconsistent results alongside insufficient supporting evidence regarding prediction accuracy according to [13], [14]. Many agile environments lack historical performance connection in their estimation processes making time-based validation and calibration of estimates difficult to execute[15], [16]. Expert-based and machine learning estimation approaches share a common flaw in their approach for selecting relevant attributes. Recursively eliminated features through automated processes and feature engineering approaches in ML tend to deselect crucial contextual parameters which experts consider mandatory for their

Bashir, A., Zakir, M., & Bajwa, Y. (2025). Enhancing Software Effort Estimation in Agile Projects Through Expert-Guided Feature Selection and Machine Learning. *International Journal of Scientific Research and Modern Technology*, *4*(11), 28–35. https://doi.org/10.38124/ijsrmt.v4i11.949

work. The rationale behind selecting specific estimation variables by expert-based methods remains undocumented and non-standardized which obstructs the scalability and transportability of their approach.

This study identifies a critical gap in the integration of human judgment into the machine learning pipeline—specifically during the feature selection phase. Previous research has either used expert knowledge in post-prediction calibration or evaluated expert and ML estimates separately, without exploring a combined, synergistic approach. This study addresses this gap by proposing a hybrid effort estimation methodology that systematically incorporates expert-informed feature selection into machine learning regression models.

To achieve this, a mixed-method approach is adopted. First, a structured survey is conducted across a diverse group of agile practitioners to capture their perspectives on which project attributes most significantly influence effort estimation. These expert-informed features are then validated and used as input for three ML regression models—Linear Regression, Multilayer Perceptron, and SVR. The study evaluates model performance using standard metrics such as R-squared, Mean Absolute Error (MAE), and Mean Squared Error (MSE).

The main objective of this research is to enhance the accuracy and practical relevance of effort estimation in Agile development by combining the contextual richness of expert judgment with the predictive power of machine learning. By doing so, the study aims to (i) identify and prioritize effort-influencing factors based on practitioner input, (ii) evaluate the performance improvement in ML models using expert-informed features, and (iii) provide empirical evidence supporting the adoption of hybrid estimation approaches in real-world agile settings. The findings of this study offer valuable insights for software project managers, data scientists, and researchers seeking to improve the reliability and transparency of effort estimation frameworks.

II. LITERATURE REVIEW

Software project management depends on the accurate estimation of project effort because it helps create sound plans along with budget allocations and resource distribution and risk reduction [17]. The development field has introduced an extensive set of methods aimed at improving estimation precision and specifically for Agile Software Development projects since they present sophisticated challenges [2]. The field divides effort estimation models into expert-based methods and algorithmic and hybrid approaches while research focuses on using machine learning to automate estimation methods.

- *Expert-Based Estimation Approaches*

The practice of using expert-based effort estimates proves to be one of the dominant methods in software engineering because Agile environments require minimal formal specifications combined with high flexibility requirements. Domain professionals utilize their knowledge and instinct to estimate the effort requirements for tasks and projects [18]. The most prevalent expert-based techniques for estimation tasks are Planning Poker together with Delphi method as well as T-shirt sizing and Wideband Delphi and Team Estimation Game. These methods prove popular during the early phases of projects because historical information either lacks appropriate relevance or remains scarce from new domain implementation.

Expert-based methods demonstrate a unique capability to handle different project circumstances as their strongest point. Experts effectively integrate unspoken organizational knowledge with business limits as well as team-related aspects into their judgment process which proves challenging to systematize or automate. Estimation models exclude the important considerations practitioners use such as interpersonal elements and stakeholder commitments and regulatory requirements as well as changes to the technology platform.

Expert-based methods consistently face the issue of cognitive bias as their main weakness. The estimation accuracy remains negatively impacted by anchoring together with overconfidence and availability heuristic and groupthink according to research conducted by Jørgensen and Grimstad [19] and Mair and Shepperd [20]. Planning Poker exposes individuals to initial values which serve as anchors throughout the collaborative session resulting in decisions that might not be accurately reflective of various estimates.

Expert-based project estimations fail to provide both repeatability and traceability features that are essential for both regulated projects and large-scale implementations. The study [21] showed expert estimates performed better than automated methods in terms of speed along with flexibility but experts produced data with higher variability while achieving lower accuracy rates on average. Project risks increase due to this unpredictable variability of expert estimations. Expert-based approaches persist in Agile workflows despite their drawbacks because they provide straightforward use as well as minimal configuration needs and collaborative functionality. Agile methodologies like Scrum use estimation sessions to perform both cost projection tasks and to maintain teamwide requirement knowledge along with requirement refinement activities. The secondary function of communication represents a significant factor that organizations continue using expert-based estimation methods.

Modern research indicates the combination of expert opinion with organized framework components helps reduce various estimation challenges. The work of [22] presents decision support tools together with visual analytics to make expert-based estimation both less biased and more calibrated. Research using hybrid methods which combine expert judgment and data-driven approaches like your proposed analysis has emerged as a

solution to manage relevant contextual understanding with robust empirical foundation.

➢ *Algorithmic Estimation Approaches*

Mathematical formulas link project characteristics such as size and complexity and cost drivers in order for algorithms to overcome expert-based method subjectivity [23]. COCOMO II stands as the prominent member of the COCOMO (Constructive Cost Model) family because it determines effort calculation using Lines of Code or Function Points alongside multiple cost-driving factors [24]. The most well-known estimation models include SLIM combined with SEER-SEM supported by Function Point Analysis. Agile project environments find restricted application for these structured yet transparent evaluative models. These models need complete and unchangeable specifications from the beginning of the project lifecycle yet Agile projects typically cannot satisfy such requirements.

The models face difficulties when attempting to deal with non-traditional development patterns such as rapid prototyping and component reuse and changing requirements that are typical in ASD [25].

➢ *Machine Learning-Based Estimation Approaches*

The usage of machine learning (ML) technology in software effort estimation has significantly grown in the last twenty years. ML models extract complicated numerical patterns in historical information while providing more precise predictions and adjustable learning capabilities. The three main models developed for software effort estimation are Linear Regression (LR), Support Vector Regression (SVR), Decision Trees, Random Forests, and Neural Networks according to [9].

Multiple research studies conducted by [26], [27] demonstrate that machine learning models provide better performance than conventional algorithmic methods when they utilize high-quality data sets. The success of these models completely relies on the selection and standard of input features they receive. The common practice of employing Automatic feature selection tactics using RFE and PCA might discard critical domain-specific variables which experts observe in the field. These ML models work as untransparent diagnostic tools that reduce practitioners' trust because of their black box nature.

➢ *Hybrid Approaches*

Hybrid estimation approaches combine the pertinent aspects from both expert-based approaches and data-driven procedures. Two modeling approaches exist that either apply expert-weighted selections for inputs before ML model implementation or unify statistical computations with additional consensus-driven modifications. Expert knowledge has no defined structure within these approaches nor do these approaches undergo empirical real-world Agile assessments [11]. The proposed research solves these testing problems with an integrated approach that uses expert evaluation to establish which input characteristics will support ML regression models. The research enhances prediction accuracy within Agile settings because it combines structured methods to obtain expert knowledge and evaluate framed efforts among real-world Agile settings.

III. METHODOLOGY

The research design combines expert qualitative findings together with machine learning methods to enhance software effort prediction accuracy within Agile methodologies. The study follows a three-step process which begins with the expert insight collection through a structured survey then moves to regression models development using expert-specified along with algorithmically picked features and finally determines the model performance through standard evaluation metrics.

➢ *Expert-Driven Feature Identification*

The initial stage adopts detailed questionnaires distributed among 45 Agile software practitioners to detect effort-influencing factors. Different stages of Agile development received input from developers, quality assurance professionals, project managers, and team leads who came from different organizations to ensure survey participants came from diverse backgrounds. Common estimation variables from the literature served as the focus of the survey which included requirement accuracy together with risk levels and product complexity and security needs and process re-engineering and tool availability. The survey employed Likert scales for respondents to evaluate the extent to which these factors would impact software development effort. The responses provided for frequency analysis led to identify 18 initial attributes that subsequently required model implementation with 14 prominent attributes according to participant input.

➢ *Machine Learning Modeling*

The chosen features became inputs for building machine learning regression models using Linear Regression (LR) and Support Vector Regression (SVR) together with Multilayer Perceptron (MLP) as a third option. The selected models demonstrated effectiveness in previous software engineering studies because they both detected linear and non-linear patterns in data. Two distinct methods of feature selection were used to evaluate expert contributions in the analysis. Recursive Feature Elimination (RFE) served as the first strategy which applied an automated feature selection method based on performance-based rankings. The second evaluation used the 14 features that experts surveyed which provided human-generated insight into what factors most influence software development effort.

➢ *Evaluation Metrics*

The adopted data normalization procedure served as a prerequisite for dividing it into training (80%) and testing (20%) subsets. The evaluation used two types of features with each model trained using RFE-based features and expert-informed features to compare prediction accuracy differences. Three metrics assessed the models' performance by measuring R-squared (R^2) for explaining effort variability and Mean Absolute Error (MAE) for

average error calculation along with Mean Squared Error (MSE) that increases importance of big errors.

All programming activities for Machine Learning model applications and data preparation functions operated through the Python programming language. The data manipulation work was carried out with libraries pandas and NumPy while modeling and feature selection happened through scikit-learn. The SPSS software analyzed survey results statistically while matplotlib and seaborn tools produced the visual outputs. The research method selection included mixed techniques because this methodology enhanced the shortcomings which arise from using expert assessments or artificial intelligence alone. The efficiency of ML models at data pattern recognition becomes irrelevant when experts possess domain-focused insights which are essential for Agile development with its quick-changing requirements. This study implements expert-driven selection of features into the machine learning framework to establish a detailed and understandable Agile software development project effort estimation system.

IV. RESULTS AND DISCUSSION

The authors conducted experiments using Linear Regression (LR) together with Support Vector Regression (SVR) and Multilayer Perceptron (MLP) on feature collections derived both from expert insights and recursive feature elimination (RFE). The standard performance evaluation metrics R-squared (R^2), Mean Squared Error (MSE) and Mean Absolute Error (MAE) assessed the model performance. A multi-stage experiment evaluated the impact of features using a baseline of 8 initial characteristics then adding 10 expert-derived features to reach a total of 18 features and later tested another 4 selected features using RFE.

- *Performance Using Baseline 8 Features*

The first phase training phase used only eight baseline features that included team size and daily/monthly working hours plus effort duration. Table 1 provides the performance results mentioned below.

Table 1 Results with 8 Features

No.	Algorithm	R^2	MSE	MAE
1	Linear Regression	0.901229	27,364,157.53	2415.71
2	Support Vector Regr.	-0.33454	113,053,345.92	6379.14
3	MLP Regression	0.75066	17,601,740.55	2414.18

The proposed model demonstrated better performance than the other models since it produced both the maximum R^2 value and the minimum MAE and MSE values.

- *Performance with 18 Features (Including 10 Expert-Suggested)*

The second stage of the experiment integrated ten more features that domain specialists selected for evaluation. Risk management together with product complexity and technical documentation and teamwork management traits comprised the expanded pool of characteristics. Table 2 shows the obtained results.

Table 2 Results with 18 Features

No.	Algorithm	R^2	MSE	MAE
1	Linear Regression	0.931945	5,276,194.81	1726.01
2	Support Vector Reg.	-0.344582	113,056,405.12	6379.32
3	MLP Regression	0.785794	18,011,040.19	2392.52

The Linear Regression model demonstrated stronger accuracy levels as shown by rising R^2 to 0.9319 while MAE decreased to 1726 which proves that expert-advised features enhance model trustworthiness.

- *Performance with RFE-Selected Features (14+4 = 18)*

The evaluation phase evaluated the effectiveness of incorporating 4 additional features that originated from RFE analysis: organization size, object points, hiring policy and programmer experience. Table 3 displays the obtained results.

Table 3 Results with 4 RFE Features Added

No.	Algorithm	R^2	MSE	MAE
1	Linear Regression	0.932115	5,707,917.17	1718.97
2	Support Vector Regr.	-0.344588	113,056,967.17	6379.35
3	MLP Regression	0.79513	17,225,739.76	2414.86

Linear Regression showed minimal betterment in its performance outcomes after selection of RFE-developed features. The expert evaluation method revealed more significant benefits for accuracy improvement over adding these extra features.

- *Visual Comparison of Models*

The visual review of the predictions helped to evaluate the distance between actual and predicted values.

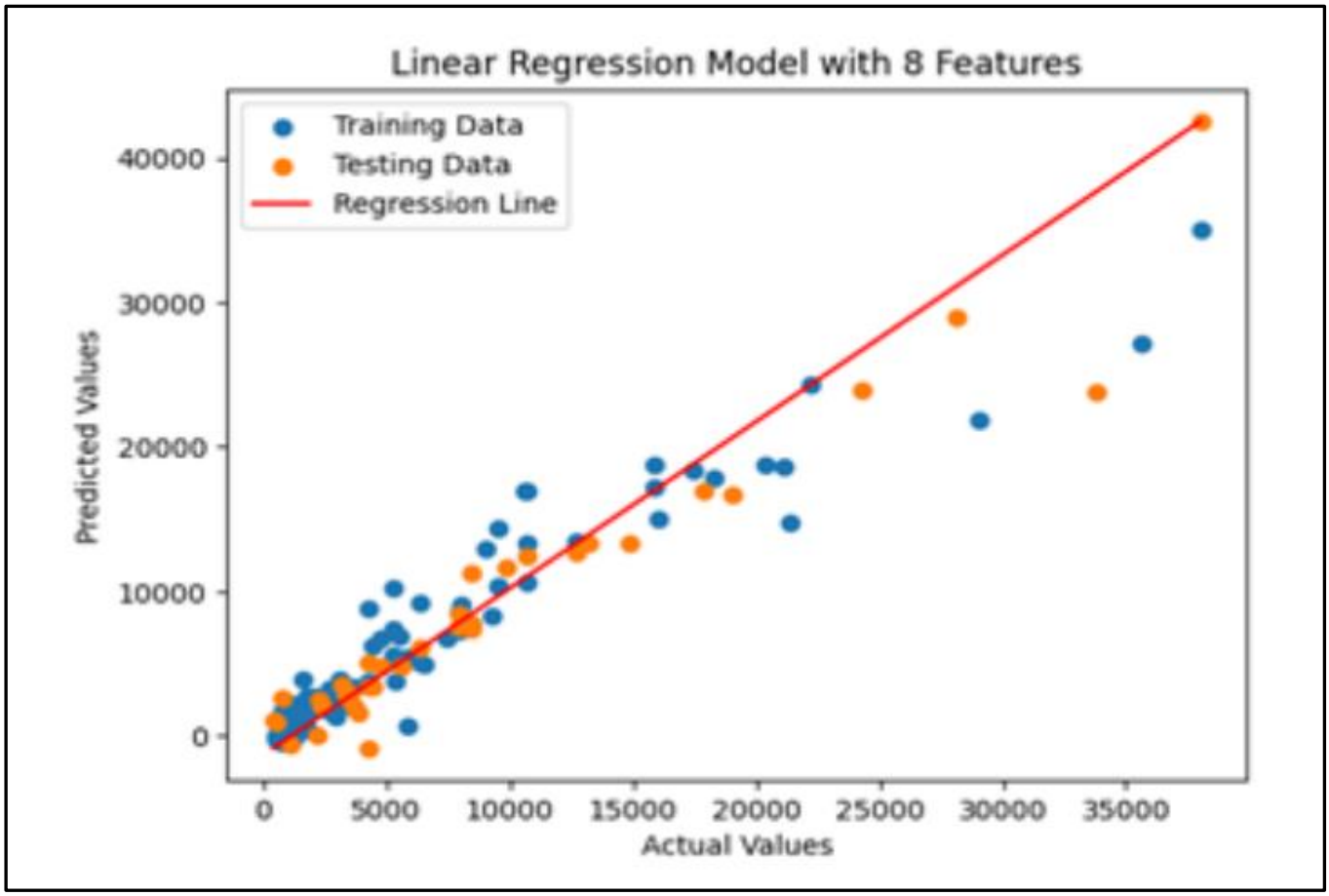

Fig 1 Linear Regression with 8 Features

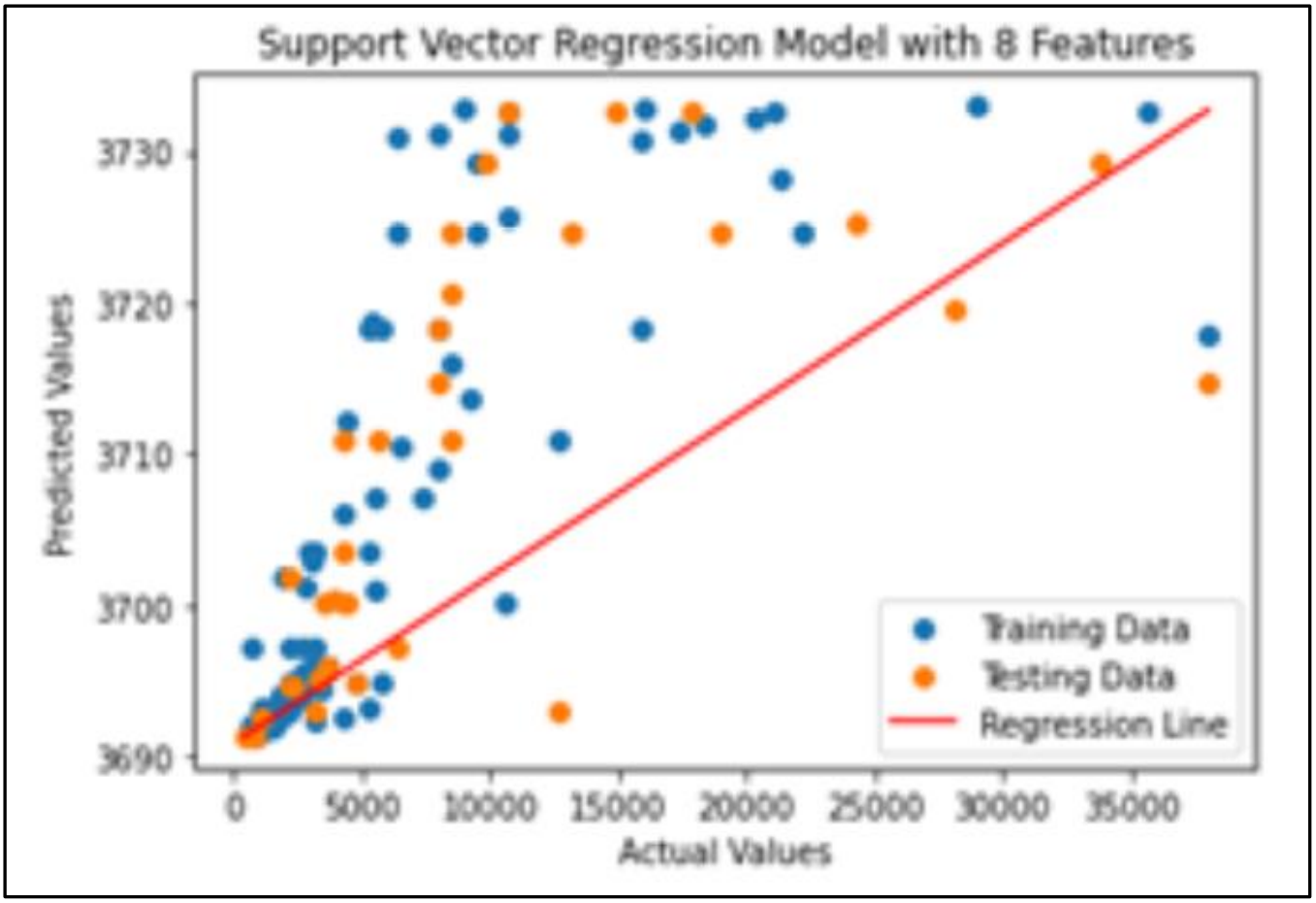

Fig 2 Support Vector Regression with 8 Features

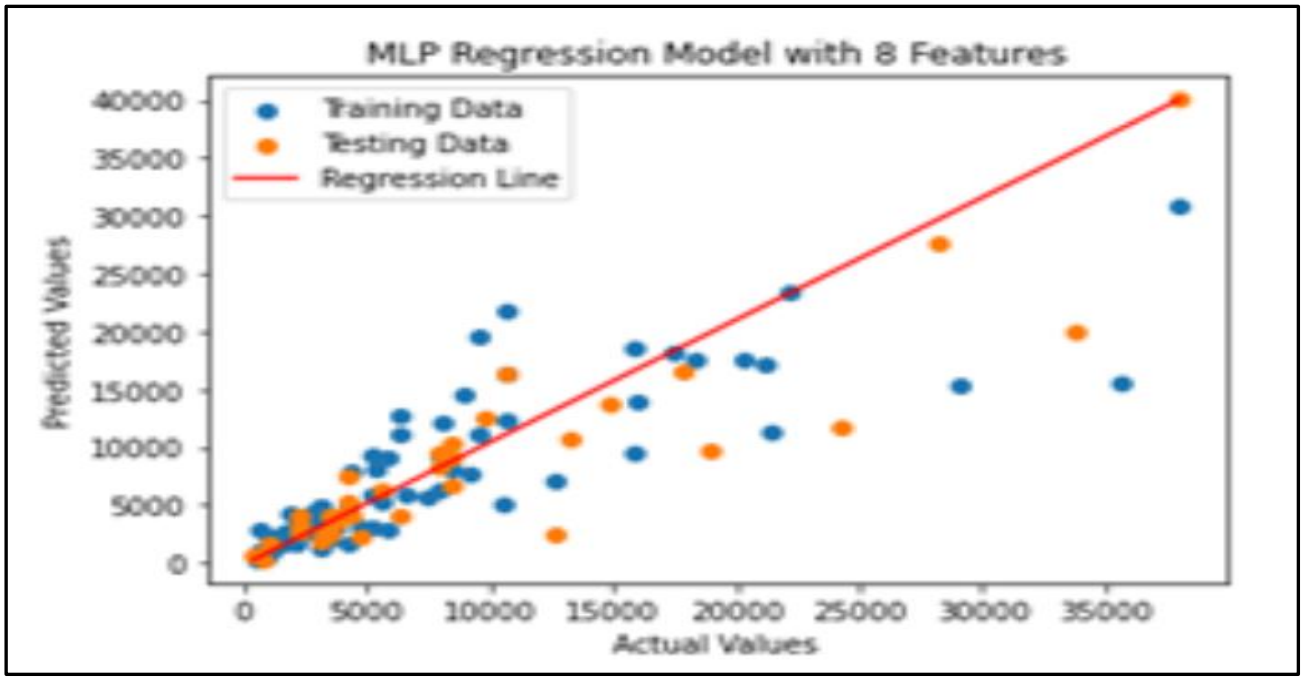

Fig 3 MLP Regression with 8 Features

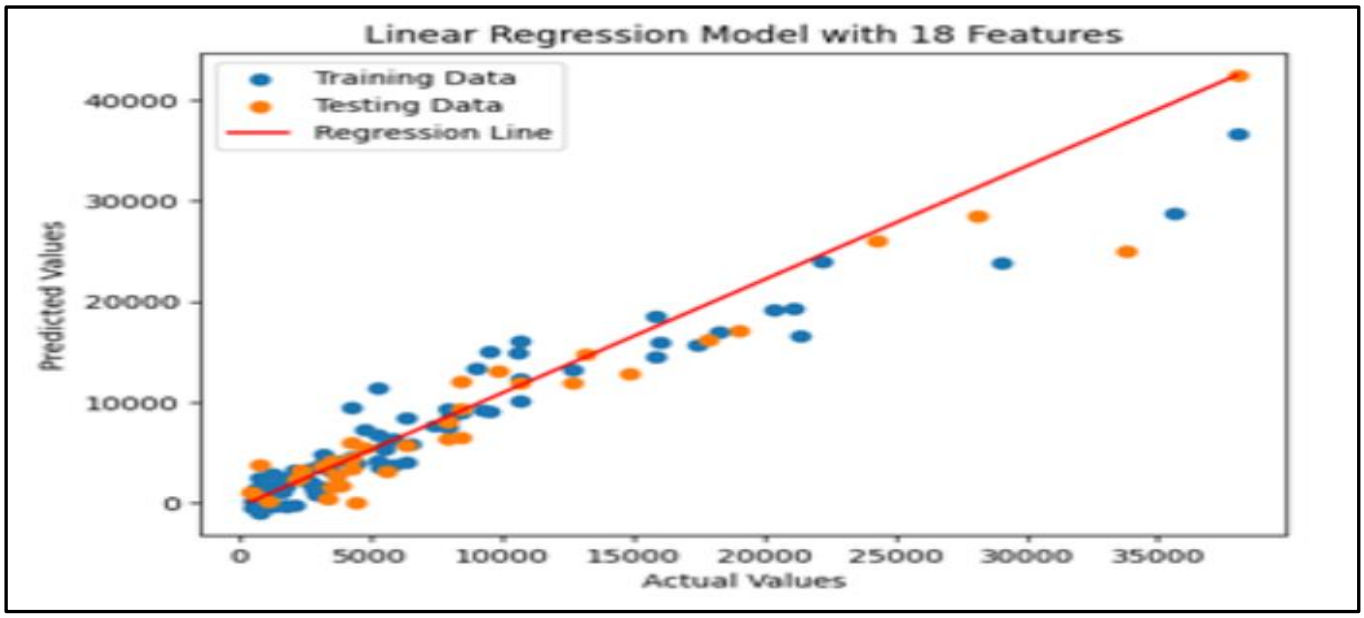

Fig 4 Linear Regression with 18 Features

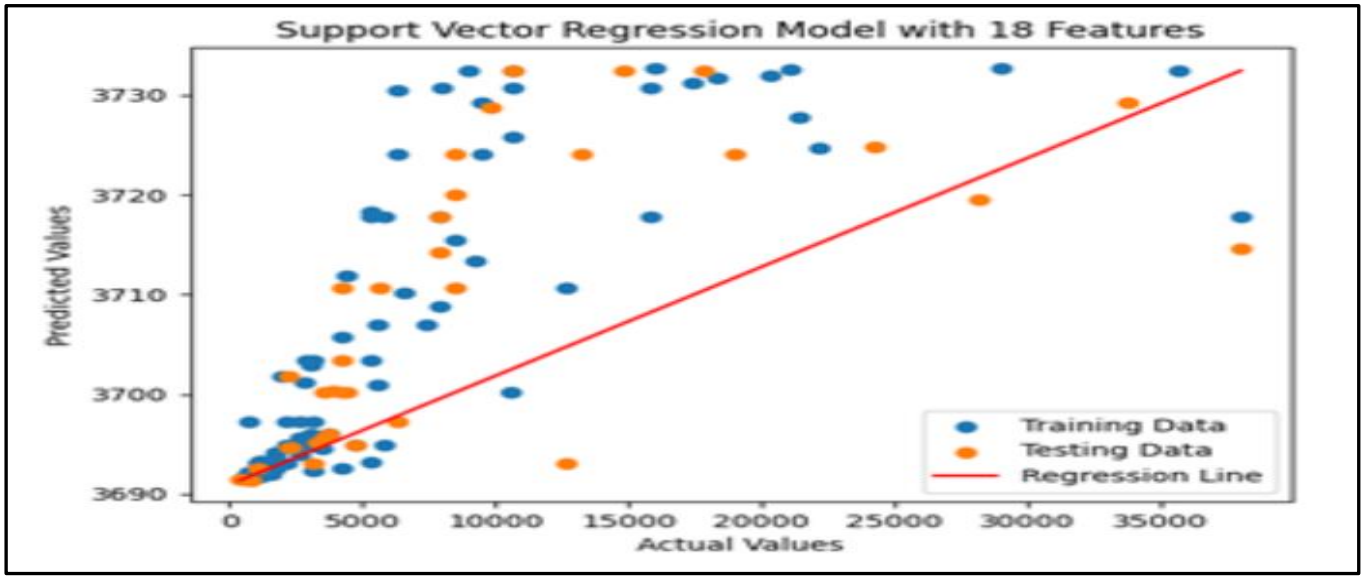

Fig 5 Support Vector Regression with 18 Features

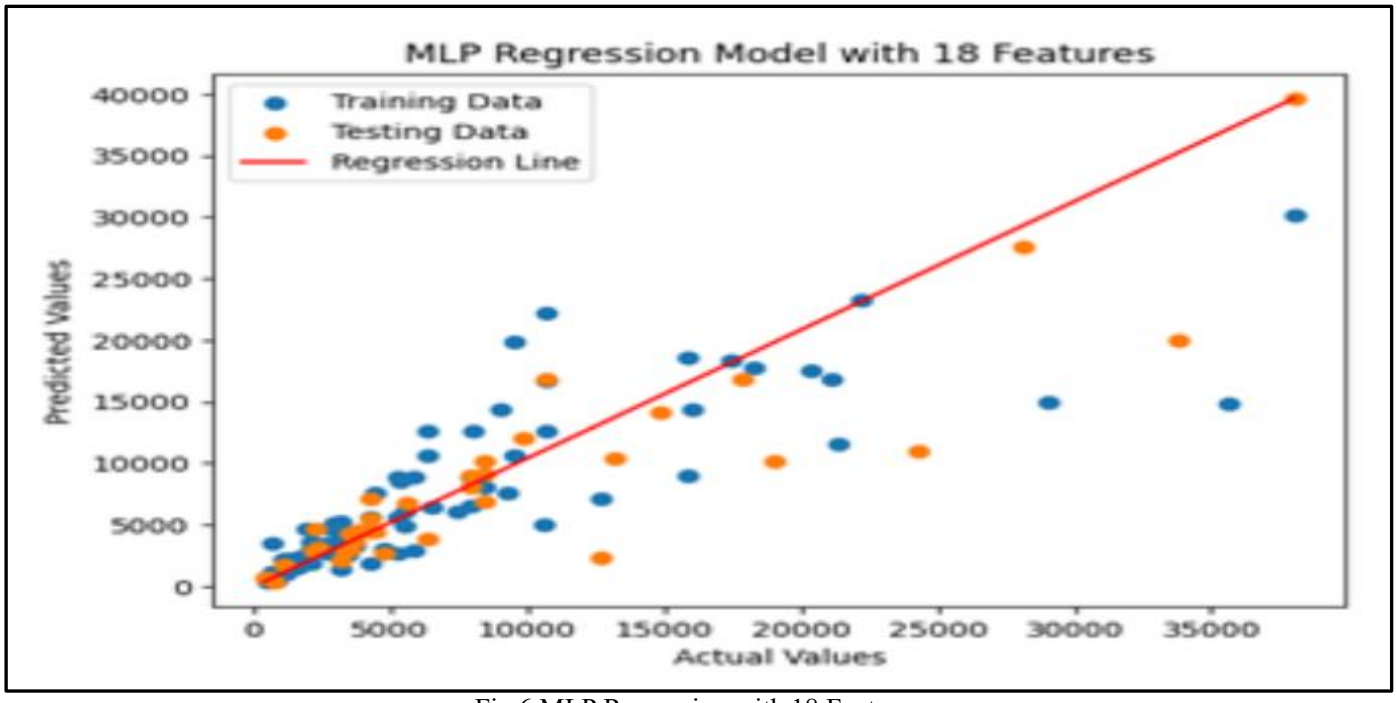

Fig 6 MLP Regression with 18 Features

In the graphical results, the predictions of Linear Regression and MLP appear close to the actual values (hyperplane), indicating good performance, whereas SVR predictions deviated significantly, confirming its poor predictive capacity on this dataset.

V. SUMMARY OF KEY FINDINGS

The results of the study demonstrate that Linear Regression consistently outperformed both Support Vector Regression (SVR) and Multi-Layer Perceptron (MLP) across all feature sets, indicating its superior predictive performance. Notably, the inclusion of expert-informed features led to a significant boost in model accuracy compared to baseline features and those selected through Recursive Feature Elimination (RFE). This suggests that incorporating domain knowledge into the feature selection process can significantly enhance model predictions. On the other hand, SVR struggled with poor generalization, as evidenced by its negative R^2 scores and high error rates, reflecting its inability to accurately predict the effort required in Agile software projects. While MLP showed moderate performance—better than SVR but still inferior to Linear Regression—it highlighted the value of hybrid models that combine domain expertise with data-driven approaches. Overall, the study's findings support the hypothesis that models integrating expert judgment with data-driven techniques can produce more accurate and reliable effort estimates in Agile environments.

VI. CONCLUSION

This study introduced a novel hybrid methodology that integrates expert judgment into the machine learning-based feature selection process for software effort estimation, particularly in Agile development environments. Recognizing the limitations of both traditional expert-based methods and purely data-driven models, the research aimed to develop an approach that leverages the strengths of both human insight and computational precision.

The findings demonstrated that expert-informed features significantly enhance the predictive performance of machine learning models. Among the models evaluated—Linear Regression (LR), Support Vector Regression (SVR), and Multilayer Perceptron (MLP)—Linear Regression consistently achieved the highest accuracy, with an R^2 value of 0.9319 and the lowest Mean Absolute Error (MAE) of 1726 when expert-selected attributes were used. In contrast, SVR underperformed across all feature sets, and MLP provided only moderate results.

These results underscore the importance of incorporating domain expertise into the machine learning pipeline, especially during the feature selection stage. The comparison between expert-informed and RFE-selected attributes further validated that expert-guided features contribute more meaningfully to model accuracy and relevance. The hybrid approach not only improved the estimation outcomes but also increased the interpretability and credibility of the models from a practitioner's standpoint.

While the study offers promising results, it is not without limitations. The dataset used, although representative, was limited in size and scope. Furthermore, the study focused only on regression-based models and did not explore advanced ensemble methods or deep learning techniques. Future research could extend this work by incorporating a larger and more diverse dataset, exploring adaptive and real-time learning models, and applying the framework across different software development methodologies and domains.

In conclusion, this study provides a validated, practical approach to improving effort estimation by

bridging the gap between human expertise and machine learning. It serves as a foundation for future work in building more transparent, context-aware, and accurate estimation tools in software project management.

REFERENCES

[1]. P. Shakya and S. Shakya, "Critical success factor of agile methodology in software industry of nepal," *J. Inf. Technol.*, vol. 2, no. 03, pp. 135–143, 2020.

[2]. M. Fernández-Diego, E. R. Méndez, F. González-Ladrón-De-Guevara, S. Abrahão, and E. Insfran, "An update on effort estimation in agile software development: A systematic literature review," *IEEE Access*, vol. 8, pp. 166768–166800, 2020.

[3]. F. Dumitriu, G. Meşniţă, and L.-D. Radu, "Challenges and Solutions of Applying Large-Scale Agile at Organizational Level.," *Inform. Econ.*, vol. 23, no. 3, 2019.

[4]. S. A. Licorish, M. Galster, G. M. Kapitsaki, and A. Tahir, "Understanding students' software development projects: Effort, performance, satisfaction, skills and their relation to the adequacy of outcomes developed," *J. Syst. Softw.*, vol. 186, p. 111156, 2022.

[5]. K. Kulathunga and S. D. Ratiyala, "Key success factors of scrum software development methodology in Sri Lanka," 2018.

[6]. M. Usman, E. Mendes, and J. Börstler, "Effort estimation in agile software development: a survey on the state of the practice," in *Proceedings of the 19th international conference on Evaluation and Assessment in Software Engineering*, 2015, pp. 1–10.

[7]. M. Rahman, P. P. Roy, M. Ali, T. Gonc, and H. Sarwar, "Software effort estimation using machine learning technique," *Int. J. Adv. Comput. Sci. Appl.*, vol. 14, no. 4, 2023.

[8]. S. Amershi *et al.*, "Software engineering for machine learning: A case study," in *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, IEEE, 2019, pp. 291–300.

[9]. M. Shepperd and S. MacDonell, "Evaluating prediction systems in software project estimation," *Inf. Softw. Technol.*, vol. 54, no. 8, pp. 820–827, 2012.

[10]. M. Rahman, H. Sarwar, M. A. Kader, T. Gonãgalves, and T. T. Tin, "Review and Empirical Analysis of Software Effort Estimation," *IEEE Access*, 2024.

[11]. M. Jørgensen and S. Grimstad, "Avoiding irrelevant and misleading information when estimating development effort," *IEEE Softw.*, vol. 25, no. 3, pp. 78–83, 2008.

[12]. T. Luu Huu, "Study of the scaling of Agile methodology in large organisations: a systematic literature review," 2024, *Universitat Politècnica de Catalunya*.

[13]. Z. Zhang, "The benefits and challenges of planning poker in software development: comparison between theory and practice," *Auckl. Univ. Technol.*, 2017.

[14]. V. Mahnič and T. Hovelja, "On using planning poker for estimating user stories," *J. Syst. Softw.*, vol. 85, no. 9, pp. 2086–2095, 2012.

[15]. B. Alsaadi and K. Saeedi, "Data-driven effort estimation techniques of agile user stories: a systematic literature review," *Artif. Intell. Rev.*, vol. 55, no. 7, pp. 5485–5516, 2022.

[16]. M. Cohn, *Agile estimating and planning*. Pearson Education, 2005.

[17]. K. S. Thant and H. H. K. Tin, "Learning the Efficient Estimation Techniques for Successful Software Project Management," *Int. J. Eng. Technol.*, vol. 11, no. 6, pp. 1–8, 2023.

[18]. Y. Mahmood, N. Kama, and A. Azmi, "A systematic review of studies on use case points and expert-based estimation of software development effort," *J. Softw. Evol. Process*, vol. 32, no. 7, p. e2245, 2020.

[19]. M. Jørgensen, "Unit effects in software project effort estimation: Work-hours gives lower effort estimates than workdays," *J. Syst. Softw.*, vol. 117, pp. 274–281, 2016.

[20]. M. Shepperd, "'Estimating software project effort using analogies': Reflections after 28 years," *IEEE Trans. Softw. Eng.*, 2025.

[21]. B. K. Kumar, S. Bilgaiyan, and B. S. P. Mishra, "Software effort estimation based on ensemble extreme gradient boosting algorithm and modified Jaya optimization algorithm," *Int. J. Comput. Intell. Appl.*, vol. 23, no. 01, p. 2350032, 2024.

[22]. M. Kassab, J. DeFranco, and P. Laplante, "Investigating Bugs in AI-Infused Systems: Analysis and Proposed Taxonomy," in *2022 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, IEEE, 2022, pp. 365–370.

[23]. A. Sultanbekov, "Designing an Effort Estimation Process for Embedded Software Projects," 2024.

[24]. T. P. Boehm, "Tenure choice and expected mobility: a synthesis," *J. Urban Econ.*, vol. 10, no. 3, pp. 375–389, 1981.

[25]. M. Jørgensen, "Forecasting of software development work effort: Evidence on expert judgement and formal models," *Int. J. Forecast.*, vol. 23, no. 3, pp. 449–462, 2007.

[26]. A. Idri, M. Hosni, and A. Abran, "Systematic literature review of ensemble effort estimation," *J. Syst. Softw.*, vol. 118, pp. 151–175, 2016.

[27]. M. Sharma and N. Fotedar, "Software effort estimation with data mining techniques-A review," *Int. J. Eng. Sci. Res. Technol.*, vol. 3, no. 3, 2014.