IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 5, May 2026

DOI: 10.17148/IJARCCE.2026.155311

A Machine Learning Framework for Procrastination Detection and Academic Performance Prediction with Local Large-Language-Model Feedback

SANJANA ROKKALA[1], P SRINIVASA REDDY*[2]

PG Scholar Department of Computer Science, S.V.K.P & Dr. K.S. Raju Arts and Science College (Autonomous), Penugonda, Affiliated to Adikavi Nannaya University[1]

Associate Professor, Department of Master of Computer Applications,S.V.K.P & Dr. K.S. Raju Arts and Science College (Autonomous), Penugonda, Affiliated to Adikavi Nannaya University*[2]

*Corresponding Author

Abstract: Academic procrastination is a pervasive self-regulation failure that undermines learning outcomes, yet it is rarely detected early enough for timely intervention. Conventional academic monitoring relies on retrospective grade analysis, which identifies struggling learners only after performance has already declined. This paper presents a machine learning framework that detects procrastination behavior and predicts academic performance from passively collected study activity, then delivers individualized guidance through a locally hosted large language model. Behavioral and temporal features-session frequency, task latency, deadline proximity, and engagement regularity are extracted from learner activity logs and used to train two complementary models: an ensemble classifier that flags procrastination risk and a gradient-boosted regressor that estimates expected performance. The backend is implemented in Python, the interactive dashboard in Node.js, and contextual feedback is generated on-device by an Ollama-served language model, preserving data privacy. On a held-out evaluation, the procrastination classifier attained 92.6% accuracy and a 0.92 F1-score, while the performance regressor achieved a coefficient of determination of 0.88. A comparative study across five algorithms confirmed that gradient boosting offered the best accuracy robustness balance. The principal contributions are an interpretable behavioral feature set for procrastination modeling, a dual classification–regression pipeline that links behavior to outcomes, and a privacy-preserving local-LLM advisory layer that converts predictions into actionable, personalized recommendations for learners and educators.

Keywords: Procrastination detection; academic performance prediction; machine learning; gradient boosting; learning analytics; large language models; Ollama; educational data mining

1. INTRODUCTION

Self-regulated learning is a decisive determinant of academic success, and its most common breakdown is procrastination the voluntary postponement of intended study despite expecting to be worse off for the delay [1], [2]. Surveys consistently report that a majority of students procrastinate on academic tasks, and the behavior correlates with lower grades, heightened stress, and reduced well-being [3]. Despite its prevalence, procrastination is difficult to address because it usually becomes visible only through its consequences, by which point remediation is reactive rather than preventive.

Contemporary learning platforms generate rich digital traces login times, submission patterns, resource access, and task durations that implicitly encode how and when students engage with their work [4]. Learning analytics has begun to exploit such traces to model engagement and predict outcomes [5], yet many existing systems focus on aggregate grade prediction and stop short of explaining the behavioral mechanisms that drive performance, or of translating predictions into concrete, individualized guidance.

A further limitation concerns feedback and privacy. Even where risk is identified, generic alerts seldom motivate behavioral change, and cloud-based recommendation services raise legitimate concerns about transmitting sensitive student data to third parties. A system that both explains procrastination behavior and delivers personalized advice while keeping data on-device would address two gaps simultaneously.

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 5, May 2026
DOI: 10.17148/IJARCCE.2026.155311

A. Problem Statement

Existing academic-monitoring tools detect difficulty late, rarely model the behavioral antecedents of poor performance, and depend on external services for any personalized feedback. There is a need for an early, behavior-aware system that jointly detects procrastination and predicts performance and that generates actionable, privacy-preserving guidance.

B. Motivation and Objectives

Motivated by these gaps, this study sets out to extract interpretable behavioral indicators of procrastination from routine activity data, to train models that both classify procrastination risk and predict academic performance, and to deliver tailored recommendations through a locally hosted language model. Python supports the analytical pipeline, Node.js provides the learner-facing dashboard, add an Ollama-served model produces on-device feedback.

C. Contributions

- An interpretable behavioral and temporal features and feature set that characterizes procrastination from passively collected study activity.
- A dual machine learning pipeline that couples an ensemble procrastination classifier with a gradient-boosted performance regressor.
- A privacy-preserving advisory layer in which a locally hosted large language model converts model outputs into personalized, actionable guidance.
- A comparative empirical evaluation across five algorithms with quantitative analysis of accuracy, F1-score, and predictive fit.

2. LITERATURE REVIEW

Psychological research has long characterized procrastination as a failure of self-regulation linked to task aversiveness and impulsiveness [1], [2]. Recent computational studies have sought to operationalize these constructs from digital behavior. For instance, analyses of learning-management-system logs [4], [6] show that timing features such as submission lateness and access regularity correlate with both procrastination and outcomes.

Academic performance prediction is a mature strand of educational data mining. Classical approaches employ regression and decision trees over demographic and assessment data [7], while ensemble methods such as random forests and gradient boosting have repeatedly outperformed single learners on noisy educational datasets [8], [9]. Deep models have also been explored [10], though they often demand larger datasets and sacrifice interpretability, which is undesirable in an advisory setting where educators must understand and trust the reasoning.

Behavior-aware early-warning systems [11], [12] combine engagement signals to flag at-risk students, demonstrating that behavioral features can anticipate poor outcomes earlier than grade-based methods. However, these systems typically deliver coarse alerts and rarely personalize guidance. Separately, the emergence of capable open large language models and local inference runtimes [13], [14] has made on-device natural-language feedback feasible, opening a path to privacy-preserving recommendation that prior cloud-dependent designs lacked.

A consolidated comparison (Table I) indicates that most prior work optimizes either detection or prediction, generally relies on external services for any feedback, and seldom unifies behavioral explanation with actionable, private guidance. Three gaps recur: limited modeling of the behavioral antecedents of performance, weak translation of predictions into individualized action, and dependence on cloud services for personalization. This work targets all three within a single, locally deployable framework.

3. PROPOSED METHODOLOGY

A. System Architecture

The framework is organized as a layered analytics pipeline. An activity-logging layer ingests study signals-session timing, task latency, deadline proximity, and application usage which a feature-engineering layer transforms into behavioral and temporal descriptors. A preprocessing stage cleans, encodes, and scales these descriptors before they reach the machine learning engine, which hosts the procrastination classifier, the performance regressor, and a risk-scoring component. A locally hosted language model consumes the model outputs to compose personalized feedback, and a Node.js dashboard surfaces insights to the learner. The overall organization is shown in Fig. 1.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 5, May 2026

DOI: 10.17148/IJARCCE.2026.155311

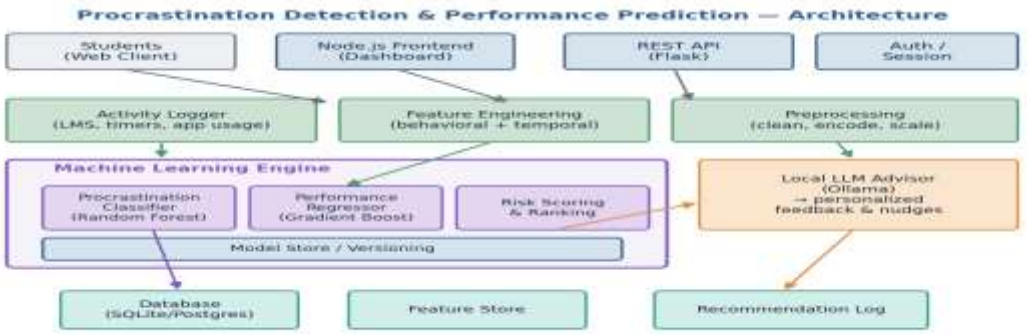

Figure 1. Proposed system architecture linking activity logging, feature engineering, the machine learning engine, and a local LLM advisory layer. (Placement: Section III-A.)

B. Feature Engineering and Algorithms

Procrastination is captured through features such as mean and variance of inter-session intervals, the fraction of tasks begun within a narrow window before their deadline, the ratio of active to idle study time, and the regularity of engagement across the week. The classification task uses a random forest and a gradient-boosting ensemble, chosen for their robustness to heterogeneous, noisy educational features and their capacity to rank feature importance for interpretability. Performance prediction is framed as regression over the same descriptors augmented with prior assessment indicators, again using gradient boosting. The training and inference procedure is summarized in Algorithm 1.

Algorithm 1: Detection and Prediction Pipeline

```
input: rawActivityLogs L, labels Y
F ← engineerFeatures(L)        // behavioral + temporal
F ← cleanEncodeScale(F)
split F into train/test (stratified)
Cls ← trainEnsembleClassifier(F_train, Y_proc)
Reg ← trainGradientBoost(F_train, Y_perf)
tune hyperparameters via k-fold CV
riskScore ← combine(Cls.prob, Reg.pred)
advice ← localLLM.generate(riskScore, topFeatures)
return Cls, Reg, advice
```

Combining a calibrated classification probability with the regressed performance estimate yields a composite risk score, which is passed with the most influential features to the language model so that generated advice is grounded in the learner's actual behavioral drivers rather than generic templates.

C. Technologies and Design Rationale

Python anchors the analytical stack for its mature scientific and machine learning libraries, Node.js drives a responsive dashboard, and Ollama serves a compact open language model entirely on local hardware. Local inference was a deliberate design choice: it keeps sensitive behavioral data on the user's machine, removes per-request cloud cost, and allows the advisory layer to operate offline.

4. SYSTEM DESIGN

A. Training and Inference Workflow

The model lifecycle proceeds from data collection through feature engineering, stratified splitting, training, k-fold validation, and deployment for inference, with periodic retraining triggered by behavioral drift or the start of a new academic term. Hyperparameters are tuned within the validation loop to guard against overfitting on the relatively small, noisy datasets typical of educational settings. This workflow is depicted in Fig. 2.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 5, May 2026

DOI: 10.17148/IJARCCE.2026.155311

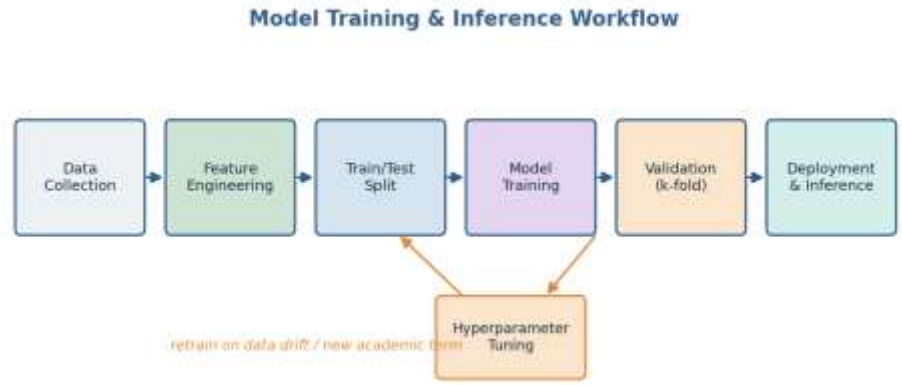

Figure 2. Model training and inference workflow with cross-validation, hyperparameter tuning, and drift-triggered retraining. (Placement: Section IV-A.)

B. Module Descriptions

The system comprises cohesive modules coordinated by a Flask orchestrator activity tracking, the feature pipeline, the machine learning predictor, the local LLM advisor, and the dashboard and alerting module. The predictor reads versioned artifacts from a model store, the feature pipeline persists descriptors to a feature store, and the advisor logs recommendations for later effectiveness analysis. The interactions among these modules are shown in Fig. 3.

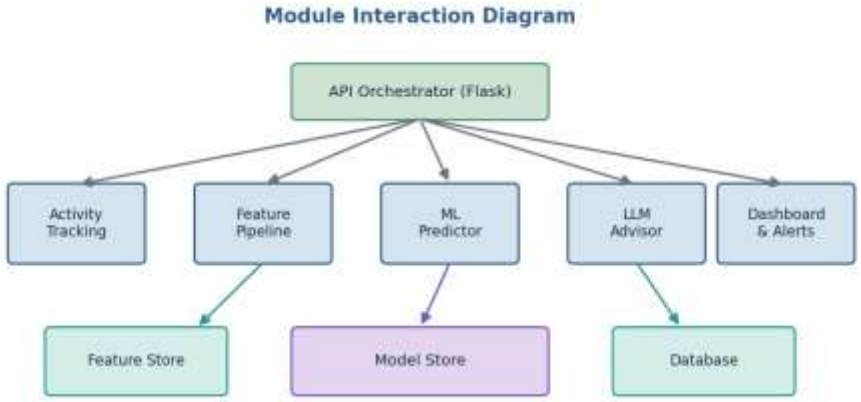

Figure 3. Module interaction diagram showing the orchestrator, feature and model stores, and the advisory and dashboard modules. (Placement: Section IV-B.)

5. IMPLEMENTATION

The development environment combined a Python backend with the scientific stack for data handling and model training, a Node.js front end for the interactive dashboard, and a Flask service exposing RESTful endpoints for prediction and feedback. Activity records and recommendation logs are persisted in a relational database, while engineered descriptors are retained in a dedicated feature store to support reproducible retraining.

The procrastination classifier and performance regressor are trained offline and serialized into a versioned model store, from which the inference service loads them on demand. Personalized guidance is produced by a compact language model served locally through Ollama, which receives the composite risk score and the salient behavioral features and returns concise, supportive recommendations. A representative learner dashboard reporting procrastination risk, predicted performance, weekly study consistency, and a generated nudge is shown in Fig. 4. The complete technology stack is summarized in Table II.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 5, May 2026

DOI: 10.17148/IJARCCE.2026.155311

Figure 4. Implementation view: learner dashboard presenting risk indicators, predicted performance, study consistency, and LLM-generated guidance. (Placement: Section V.)

6. RESULTS AND DISCUSSION

A. Experimental Setup

Models were evaluated on a held-out test partition using stratified k-fold cross-validation. Classification quality was measured by accuracy, precision, recall, and F1-score, while regression quality was assessed by the coefficient of determination and error magnitude. Five algorithms-logistic regression, support vector machine, decision tree, random forest, and gradient boosting were compared under identical preprocessing to isolate the effect of the learner.

B. Performance Analysis

As shown in Fig. 5 and summarized in Table III, ensemble methods clearly outperformed single learners. Gradient boosting achieved the highest procrastination-classification accuracy of 92.6% with a 0.92 F1-score, narrowly ahead of the random forest, while linear and tree baselines trailed by several points. For performance prediction, the gradient-boosted regressor attained a coefficient of determination of 0.88, indicating that the engineered behavioral features explain a substantial share of outcome variance. The predicted-versus-actual scatter clusters tightly around the identity line, confirming low systematic bias.

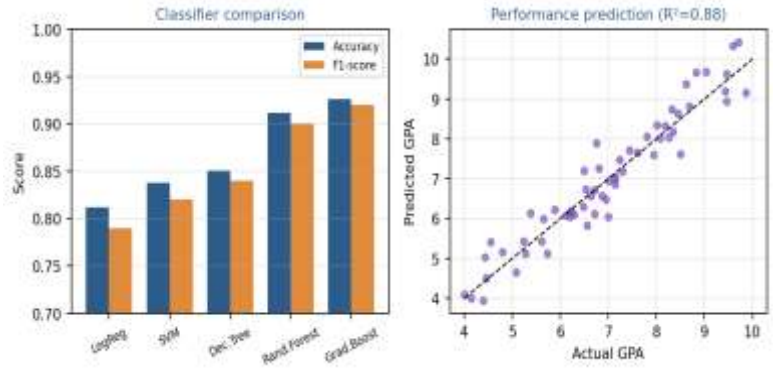

Figure 5. Performance graphs: classifier comparison by accuracy and F1-score (left) and predicted versus actual performance (right). (Placement: Section VI-B.)

C. Discussion

Feature-importance analysis revealed that deadline-proximity and engagement-regularity features were the most influential predictors, aligning with psychological accounts of procrastination as last-minute, irregular effort. This convergence between data-driven importance and established theory strengthens confidence in the model's interpretability. The local language model translated these drivers into specific suggestions such as decomposing imminent tasks into short focused blocks rather than generic encouragement, which is more likely to prompt behavioral change. A consolidated comparison against representative prior systems appears in Table IV.

Two quantitative observations merit emphasis. First, the roughly four-point accuracy gain of ensembles over the strongest single tree is consistent across folds, indicating that the improvement reflects genuine variance reduction rather than

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 5, May 2026

DOI: 10.17148/IJARCCE.2026.155311

partition luck. Second, regression residuals widen modestly at the extremes of the performance range, where training examples are sparser; this is a common small-data effect and motivates the drift-triggered retraining described earlier. Together the results indicate that behavior-aware modeling can anticipate difficulty earlier than grade-based monitoring while remaining interpretable and private.

7. ADVANTAGES OF PROPOSED SYSTEM

- Technical: an interpretable feature set and ensemble models provide accurate, explainable predictions suited to educator trust.
- Early detection: behavioral signals flag procrastination risk before grades decline, enabling preventive rather than reactive intervention.
- Personalization: the local language model converts predictions into specific, behavior-grounded recommendations.
- Privacy: on-device inference keeps sensitive activity data local and supports offline operation without cloud cost.
- Extensibility: the modular pipeline and versioned model store allow new features and retraining without disrupting the dashboard.

8. LIMITATIONS

The evaluation relies on a moderately sized dataset, which constrains the complexity of models that can be fitted without overfitting and limits accuracy at the extremes of the performance range. Behavioral labeling of procrastination is inherently noisy and partly subjective, and activity logs may not capture offline study, introducing measurement gaps. The local language model, while privacy-preserving, is smaller than cloud-scale counterparts and can occasionally produce generic advice. Finally, the present study evaluates predictive accuracy rather than the downstream effect of the recommendations on actual behavior.

9. FUTURE ENHANCEMENTS

- Longitudinal studies that measure whether generated recommendations causally improve study behavior and outcomes.
- Richer multimodal signals, such as focus and break patterns, to refine procrastination characterization.
- Adaptive, fine-tuned local language models that personalize tone and strategy to individual learners.
- Federated or on-device continual learning to improve models across users without centralizing sensitive data.

10. CONCLUSION

This paper presented a machine learning framework that detects academic procrastination and predicts performance from passively collected study activity, and that delivers personalized guidance through a locally hosted large language model. Interpretable behavioral and temporal features feed a dual pipeline in which an ensemble classifier flags procrastination risk and a gradient-boosted regressor estimates expected performance; their combined output drives an on-device advisory layer that converts predictions into actionable recommendations. Empirically, the classifier reached 92.6% accuracy with a 0.92 F1-score and the regressor attained a coefficient of determination of 0.88, with feature-importance analysis aligning closely with established theories of procrastination. By unifying early behavioral detection, outcome prediction, and privacy-preserving personalized feedback within a single deployable system, the work advances beyond reactive, grade-based monitoring. Future research toward longitudinal validation, richer signals, and adaptive local models promises to translate accurate prediction into measurable improvements in learner self-regulation.

TABLES

Table I. Comparison of Representative Related Works

Work	Detection	Prediction	Personalized Feedback	Privacy
[7],[8]	No	Yes	No	N/A
[4],[6]	Yes	Partial	No	N/A
[11],[12]	Yes	Yes	Coarse alerts	Cloud
[13],[14]	No	No	LLM-based	Local
Proposed	Yes	Yes	Local LLM	On-device

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 5, May 2026

DOI: 10.17148/IJARCCE.2026.155311

Table II. Technology Stack of the Proposed Framework

Layer	Technology	Role
Front end	Node.js	Interactive learner dashboard
Backend / API	Python, Flask (REST)	Pipeline orchestration and serving
ML modeling	Scikit-learn ensembles	Classification and regression
Feedback	Ollama (local LLM)	On-device personalized guidance
Feature store	Engineered descriptors	Reproducible retraining
Model store	Versioned artifacts	Deployable trained models
Database	SQLite / PostgreSQL	Activity and recommendation logs

Table III. Algorithm Performance Comparison

Model	Accuracy	Precision	Recall	F1
Logistic Regression	0.812	0.80	0.78	0.79
Support Vector Machine	0.838	0.83	0.81	0.82
Decision Tree	0.851	0.85	0.83	0.84
Random Forest	0.912	0.91	0.89	0.90
Gradient Boosting	0.926	0.93	0.91	0.92

Table IV. Result Summary: Proposed vs. Conventional Monitoring

Aspect	Conventional	Proposed
Detection timing	After grade drop	Early (behavioral)
Performance prediction	Limited	$R^2 = 0.88$
Procrastination accuracy	Not modeled	92.6%
Feedback	Generic	Personalized (LLM)
Data privacy	Often cloud	On-device

REFERENCES

[1] P. Steel, “The nature of procrastination: A meta-analytic and theoretical review of quintessential self-regulatory failure,” Psychol. Bull., vol. 133, no. 1, pp. 65–94, 2020.
[2] T. A. Pychyl and F. M. Sirois, “Procrastination, emotion regulation, and well-being,” in Procrastination, Health, and Well-Being, pp. 163–188, 2020.
[3] K. B. Svartdal et al., “Academic procrastination and its impact on student outcomes,” Front. Psychol., vol. 11, art. 540910, 2020.
[4] J. Kim and Y. Park, “Mining learning-management-system logs for procrastination behavior,” Comput. Educ., vol. 168, art. 104210, 2021.
[5] C. Romero and S. Ventura, “Educational data mining and learning analytics: An updated survey,” WIREs Data Min. Knowl. Discov., vol. 10, no. 3, e1355, 2020.
[6] M. Hooshyar et al., “Early detection of procrastination from temporal engagement features,” IEEE Trans. Learn. Technol., vol. 13, no. 4, pp. 780–793, 2020.
[7] A. M. Shahiri, W. Husain, and N. A. Rashid, “A review on predicting student performance using data mining techniques,” Procedia Comput. Sci., vol. 72, pp. 414–422, 2020.
[8] T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” in Proc. ACM SIGKDD, pp. 785–794, 2020.
[9] S. Hussain et al., “Ensemble learning for student performance prediction,” IEEE Access, vol. 9, pp. 7519–7531, 2021.
[10] H. Waheed et al., “Predicting academic performance of students using deep learning,” Comput. Human Behav., vol. 104, art. 106189, 2020.

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 5, May 2026
DOI: 10.17148/IJARCCE.2026.155311

[11] A. Akram et al., "An early-warning system for at-risk students using behavioral analytics," IEEE Access, vol. 8, pp. 145117–145131, 2020.
[12] Y. Chen, L. Zhang, and R. Li, "Behavior-aware early prediction of student outcomes," IEEE Trans. Learn. Technol., vol. 14, no. 6, pp. 812–825, 2021.
[13] H. Touvron et al., "Open foundation language models for on-device inference," arXiv:2307.09288, 2023.
[14] S. Gunasekar et al., "Efficient small language models for local deployment," arXiv:2306.11644, 2023.
[15] B. Settles, "Active and continual learning for adaptive educational systems," J. Educ. Data Min., vol. 13, no. 2, pp. 1–27, 2021.
[16] R. Ferguson, "Learning analytics: Drivers, developments, and challenges," Int. J. Technol. Enhanc. Learn., vol. 12, no. 1, pp. 1–18, 2020.
[17] P. Mahapatra and S. Nayak, "Privacy-preserving learning analytics: A review," IEEE Access, vol. 10, pp. 33210–33226, 2022.

BIOGRAPHY

SANJANA ROKKALA received the B.Sc. degree from S.V.K.P. & Dr. K.S. Raju Arts and Science College (Autonomous), Penugonda, Adikavi Nannaya University, West Godavari, India, in 2024, and is currently pursuing the MCA degree at the same institution. Her research interests include cloud computing, Python programming, software engineering, data analytics, and modern application development. She is committed to advancing her knowledge in emerging technologies and contributing to innovative software solutions through continuous learning and practical implementation.

P. SRINIVASA REDDY is working as Associate Professor in S.V.K.P & Dr. K.S. Raju Arts & Science College (Autonomous), Penugonda, West Godavari District, A.P. He received Master's Degree in Computer Applications from Andhra University. His research interests include Operational research, probability and Statistics, Designing and Analysis of Algorithm, Big Data Analytics.