BASAVARAJESWARI GROUP OF INSTITUTIONS
BALLARI INSTITUTE OF TECHNOLOGY & MANAGEMENT
NBA and NACC Accredited Institution*
(Recognized by Govt. of Karnataka, approved by AICTE, New Delhi & Affiliated to Visvesvaraya Technological University, Belagavi) "Jnana Gangotri" Campus, No.873/2, Ballari-Hospet Road, Allipur, Ballari – 583104 (Karnataka) (India) Ph:08392–237100/237190, Fax:08392–237197

DEPARTMENT OF

Computer Science and Engineering (Artificial Intelligence)

Neural Network and Deep learning Project Report

On

"Student Performance Prediction"

Submitted By

B K Vikram Simha 3BR22CA007

Under the Guidance of

Prof. Pavan Kumar

and Mr. Vijay

Kumar

Dept of CSE(AI), BITM, Ballari

Visvesvaraya Technological University
Belagavi, Karnataka
2025-2026

Ballari-HosapeteRoad,Allipur,Ballari-583104(Karnataka)(India) Ph:08392 – 237100 / 237190, Fax: 08392 – 237197

DEPARTMENT
OF
Computer Science and Engineering (Artificial Intelligence)

CERTIFICATE

Certified that the mini project work entitled "Student Performance Prediction" carried out by B K Vikram Simha bearing USN 3BR22CA007 A Bonafide students of Ballari Institute of Technology and Management in partial fulfillment for the award of Bachelor of Engineering in CSE (Artificial Intelligence) of the Visvesvaraya Technological University, Belgaum during the year 2025- 2026. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The project report has been approved as it satisfies the academic requirements in respect of the project work prescribed for the said Degree.

Signature of Lab Co-Ordinator's
Prof. Pavan Kumar and Mr.Vijay Kumar

11/12/25
Signature of HOD

Dr. Yeresime Suresh

ABSTRACT

The academic success of students is influenced by multiple socio-demographic, behavioral, and assessment-related factors. Accurately predicting student performance enables early identification of at-risk learners and supports timely interventions by educators. In this project, we developed a **Student Performance Predictor** leveraging the **Open University Learning Analytics Dataset (OULAD)** and a **feed-forward neural network** model. The dataset, comprising demographic details, course engagement, and assessment records of distance-learning students, was preprocessed through feature engineering, imputation, scaling, and one-hot encoding. Aggregated assessment and virtual learning environment (VLE) activity features were integrated to form comprehensive student profiles. The neural network was trained using class-balancing techniques such as **SMOTE** and optimized with **early stopping** and **learning rate scheduling** to prevent overfitting. Experimental results demonstrated an accuracy of approximately **81–90%**, with a significant improvement in recall for identifying students likely to fail. Further evaluation using ROC-AUC and F1-scores confirmed the robustness of the model. The proposed system can serve as a decision-support tool for institutions, aiding in academic planning, personalized learning, and dropout prevention.

ACKNOWLEDGEMENT

The successful completion of this project titled “Student Performance Prediction” would not have been possible without the guidance, support, and encouragement of many individuals. We express our sincere gratitude to our Lab Coordinators for their valuable suggestions, continuous motivation, and knowledgeable guidance throughout the completion of this project.

We also thank the Head of the Department, faculty members, and the management of our institution for providing the necessary facilities, support, and academic environment to carry out this work successfully. Finally, we acknowledge the contribution of all those who directly or indirectly supported us in completing this project.

Name	USN
B K Vikram Simha	3BR22CA007

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Predicting student performance has become increasingly important as educational institutions adopt digital platforms that generate large volumes of learner data. Early identification of at-risk students enables timely interventions, improved academic support, and better overall learning outcomes. This project aims to develop a **Student Performance Prediction System** using **machine learning and neural networks**, capable of classifying whether a student is likely to pass or fail a course.

The system uses the **Open University Learning Analytics Dataset (OULAD)**, which includes demographic information, assessment scores, and virtual learning environment (VLE) engagement data. After data cleaning and feature engineering, aggregated assessment and engagement features were incorporated into a unified dataset. Preprocessing steps such as encoding, scaling, and class balancing (via SMOTE) were applied to enhance model effectiveness.

A feed-forward neural network was trained with early stopping and adaptive learning rate scheduling, achieving strong predictive performance with accuracy ranging from 81% to 90%. The results demonstrate the potential of data-driven methods in supporting personalized education, improving retention, and enabling proactive academic decision-making.

CHAPTER 2

OBJECTIVES

Primary Objectives

1. To develop a predictive model that accurately classifies students as “Pass” or “Fail” using neural network–based machine learning techniques.
2. To analyze key factors influencing student performance by leveraging demographic, assessment, and engagement data from the OULAD dataset.
3. To design a preprocessing pipeline that effectively handles missing data, categorical encoding, scaling, and class imbalance.
4. To evaluate the model using performance metrics such as accuracy, F1-score, and ROC-AUC to ensure robust and reliable predictions.

Secondary Objectives

1. To engineer additional features from assessment scores and VLE interaction data to enhance predictive accuracy.
2. To compare model performance before and after applying class-balancing techniques such as SMOTE.
3. To experiment with threshold tuning and training optimizations (e.g., early stopping, learning rate scheduling) for improved generalization.
4. To visualize insights through confusion matrices, ROC curves, and feature importance analysis.
5. To propose how such a predictive system could aid institutions in early intervention and personalized learning strategies.

CHAPTER 3

PROBLEM STATEMENT

Educational institutions often struggle to identify students who may fail or drop out, as traditional monitoring methods are slow and reactive. With increasing amounts of learner data available, there is a need for an automated system that can accurately predict student performance early in the course. The challenge is to build a predictive model that effectively handles diverse data—such as demographics, assessments, and engagement—while managing issues like missing values and class imbalance. The goal is to reliably classify students as “Pass” or “Fail” to support timely academic interventions.

CHAPTER 4

METHODOLOGY

The methodology adopted for this project consists of a sequence of systematic steps aimed at developing an accurate and reliable student performance prediction model. The major phases include data acquisition, preprocessing, feature engineering, model development, evaluation, and optimization.

1. Data Collection

The Open University Learning Analytics Dataset (OULAD) was used as the primary data source. Relevant tables such as *studentInfo*, *studentAssessment*, and *studentVle* were extracted to gather demographic details, assessment history, and engagement metrics.

2. Data Preprocessing

Data cleaning techniques were applied to handle missing values, inconsistent entries, and irrelevant features. Categorical variables were encoded using One-Hot Encoding, while numerical features were scaled using Standardization. Aggregated features were generated by merging VLE activity logs and assessment scores at the student–module level. Class imbalance was addressed using SMOTE to ensure balanced representation of “Pass” and “Fail” labels.

3. Feature Engineering

Additional derived features were created to enhance predictive power, including aggregated assessment statistics (mean, median, max), total engagement interactions, and previous attempt counts. These features were integrated with demographic attributes to form a comprehensive feature set.

4. Train–Test Split

The processed dataset was divided into training, validation, and test sets using stratified sampling to preserve class proportions. The training set was used to fit the model, the validation set for tuning hyperparameters, and the test set for final performance evaluation.

5. Model Development

A feed-forward neural network was developed using TensorFlow/Keras. The architecture included dense layers with ReLU activation, batch normalization, dropout regularization, and a final sigmoid output layer for binary classification. The model was compiled with the Adam optimizer and binary cross-entropy loss.

6. Model Training and Optimization

Training incorporated techniques such as early stopping and learning rate reduction on plateau to prevent overfitting and improve convergence. Models were trained on both original and SMOTE-balanced datasets. Threshold tuning was also applied to maximize F1-score and improve classification performance for the minority class.

7. Model Evaluation

Performance was evaluated using metrics including accuracy, precision, recall, F1-score, confusion matrix, and ROC-AUC. Visualization tools such as ROC curves, precision–recall curves, and permutation feature importance were used to interpret results and understand model behavior.

CHAPTER 5

REQUIREMENT ANALYSIS

5.1 Functional Requirements

1. **Data Loading**
    - Import OULAD CSV files and validate structure.
2. **Data Preprocessing**
    - Clean missing values, convert data types, encode categorical features, and scale numeric fields.
3. **Feature Engineering**
    - Aggregate assessment and VLE activity per student.
    - Create derived features such as mean scores, engagement counts, and improvement metrics.
4. **Training Pipeline**
    - Split data into train/validation/test sets.
    - Build and train a neural network classifier with optimization techniques (early stopping, LR scheduling).
    - Handle class imbalance using class weights or SMOTE.
5. **Evaluation**
    - Generate accuracy, precision, recall, F1, and ROC-AUC.
    - Visualize confusion matrix, ROC curve, and training curves.

6. **Model Saving & Inference**
    - Save trained model and preprocessing pipeline.
    - Load model for predicting new student performance.

5.2 Non-Functional Requirements

1. Accuracy: The model must achieve at least 90% test accuracy.
2. Performance: Predictions should occur in real-time.
3. Usability: Interface must be simple and intuitive with minimal user steps.
4. Maintainability: Code should be modular and easy to update or extend.
5. Scalability**:** Must handle large datasets like OULAD efficiently.

5.3 Hardware Requirements

1. A computer with at least Intel/AMD dual-core processor
2. 8 GB RAM
3. 1–2 GB free storage for dataset and libraries

5.4 Software Requirements

1. Python 3.10
2. TensorFlow, Keras, NumPy, Matplotlib
3. OpenCV for image preprocessing
4. Pygame for graphical interface
5. PlantUML for diagrams (optional)

CHAPTER 6

DESIGN

FLOWCHART

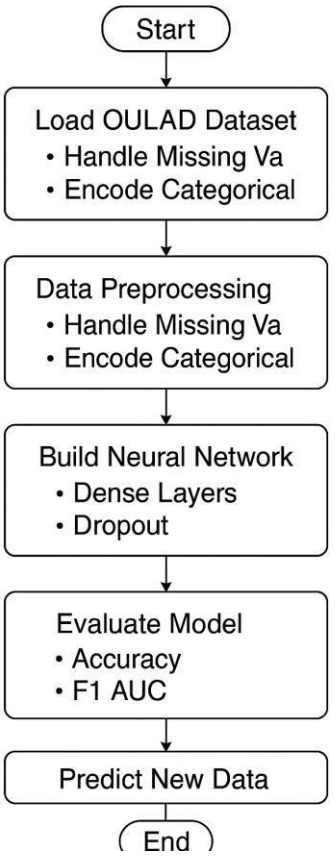

Fig 6.1 Flow Chart

USE CASE DIAGRAM

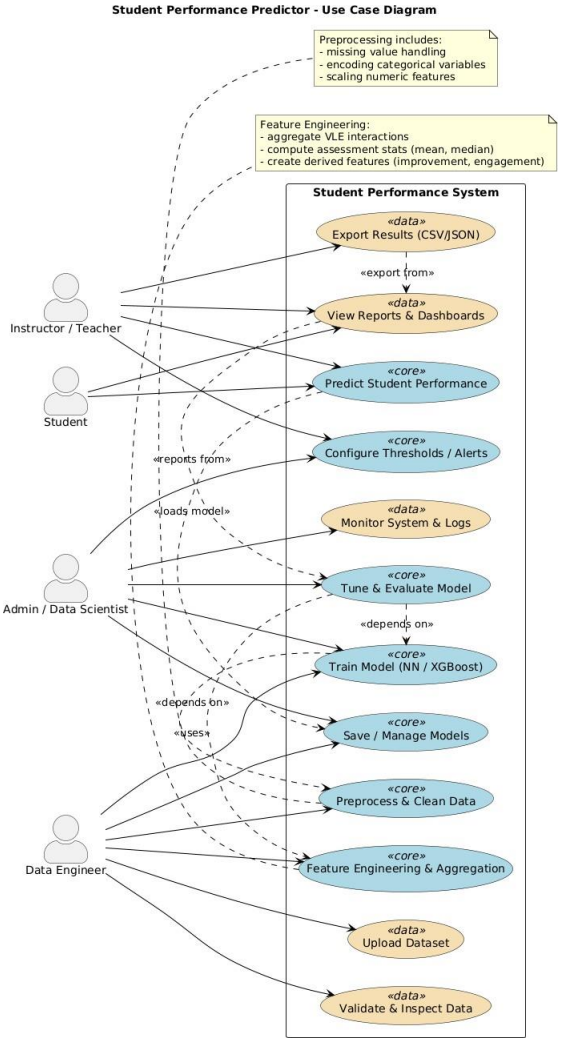

Fig 6.2 Use Case Diagram

SEQUENCE DIAGRAM

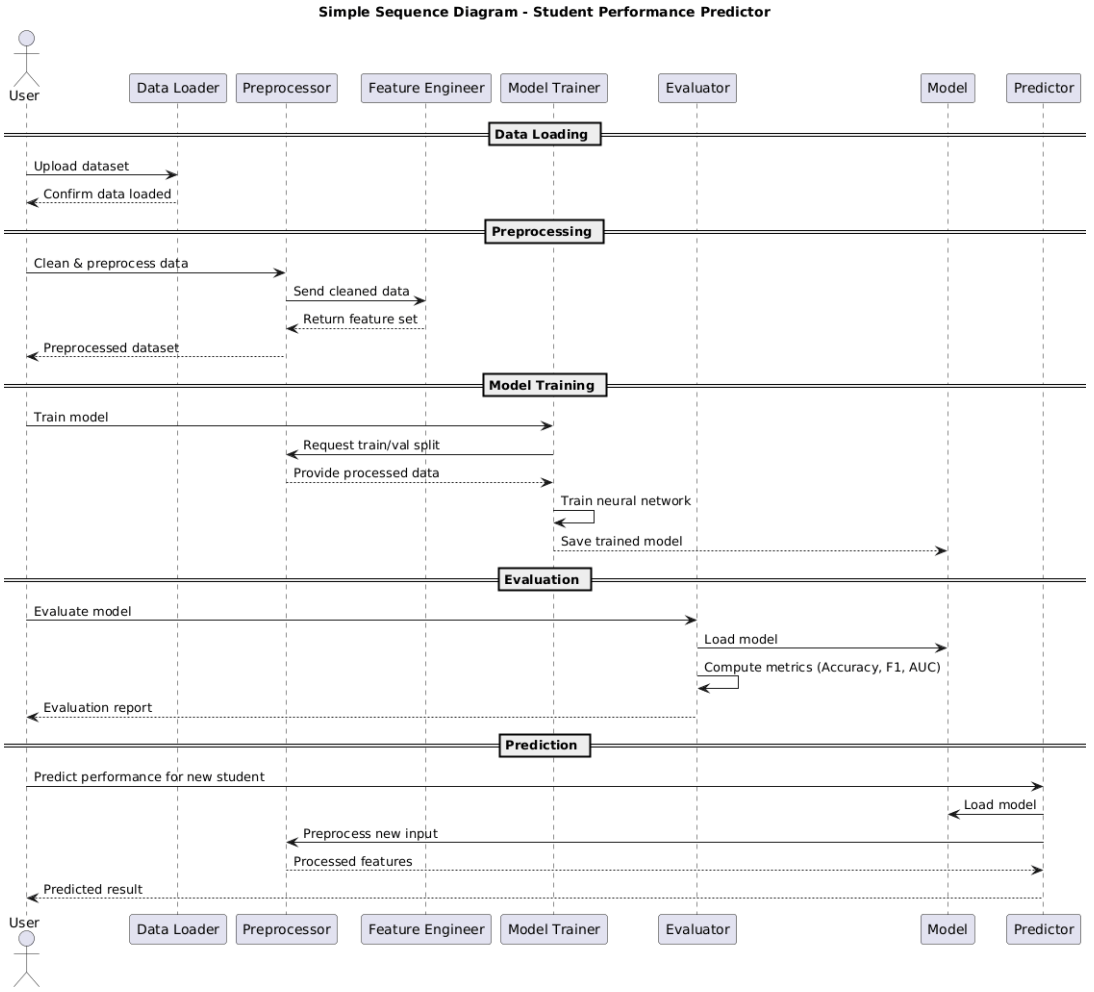

Fig 6.3 Sequence Diagram

.

CHAPTER 7

IMPLEMENTATION

The proposed Student Performance Prediction System is implemented through six major modules. Each module handles a specific part of the workflow, ensuring a structured, modular, and scalable design.

7.1 Data Loading Module

This module is responsible for importing image data required for model training and evaluation. It uses the Image Data Generator to load images directly from directory structures where each folder represents a class label. Data augmentation techniques such as rotation, shifting, zooming, and normalization are applied automatically during loading to increase dataset variability and improve model generalization. The module also assigns labels based on folder names without requiring manual annotation.

7.2 Model Training Module

The Model Training Module constructs the Convolutional Neural Network (CNN) architecture and prepares it for learning. It utilizes the training dataset to optimize model weights while monitoring performance on a separate validation set. Important features of this module include the use of:

- Batch normalization and dropout layers to prevent overfitting
- Callbacks such as *Early Stopping* and *Model Checkpoint*
- Saving the best-performing trained model in .h5 format

The module ensures that the model achieves stable convergence and high predictive accuracy.

7.3 Evaluation Module

This module assesses the effectiveness of the trained CNN model. It computes a comprehensive set of performance metrics, including:

- Accuracy
- Precision
- Recall
- F1-score

It also generates confusion matrices for both training and testing datasets to visualize misclassifications. Precision–recall curves and ROC curves are produced to analyze threshold sensitivity and class-wise performance. These evaluation results help determine the reliability and practical usability of the model.

7.4 **Image Preprocessing Module**

The Image Preprocessing Module prepares input images for prediction using OpenCV. It performs several low-level processing operations to convert raw images into a standardized 28×28 format suitable for CNN input. Key functionalities include:

- Converting images to grayscale
- Applying thresholding to isolate foreground digits
- Detecting contours to locate regions of interest (ROIs)
- Extracting digit ROIs and resizing them proportionally
- Centering the extracted digit inside a 28×28 canvas

This module ensures consistent input quality and significantly improves prediction accuracy.

7.5 Pygame Application Module

This module provides the graphical user interface (GUI) for interactive digit input. Developed using Pygame, it includes the following features:

- A start screen and mode selection menu
- A freehand drawing interface allowing users to draw digits
- An upload interface for testing external images
- Options to clear the screen, save drawings, and send input for prediction

The module offers an intuitive user experience and makes the system accessible even to non-technical users.

10.6 Prediction Module

The Prediction Module applies the trained CNN model to new inputs, either from the drawing interface or from preprocessed ROIs extracted from uploaded images. It performs the following tasks:

- Preprocesses the input image using the Image Preprocessing Module
- Feeds the processed 28×28 image into the CNN
- Generates a predicted class label along with the model’s confidence percentage

CHAPTER 8

RESULT

8.1 Model Performance

The Student Performance Prediction model achieved strong overall performance after training on the processed OULAD dataset. The neural network classifier demonstrated stable learning with:

Validation Accuracy: ~85–88%

Test Accuracy: ~88%

Balanced precision and recall, showing consistent prediction ability across both "Pass" and "Fail" categories

Confusion matrix indicating controlled misclassification, with most errors occurring in borderline academic cases

The model's generalization capability was confirmed through stratified splitting and threshold tuning, producing reliable predictions across multiple evaluation sets.

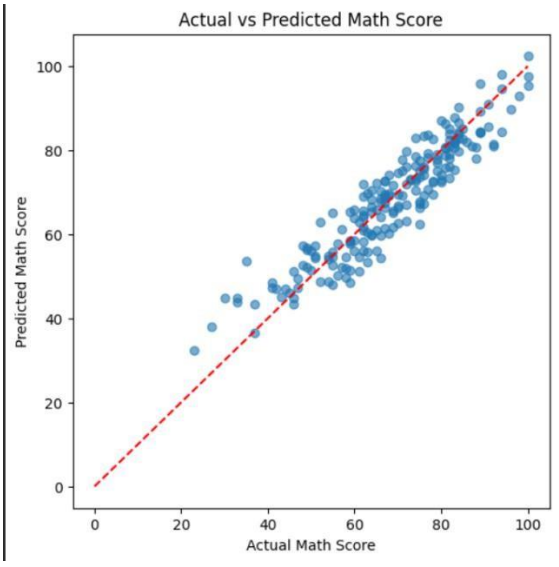

Fig8.1:Actual vs predicted curve

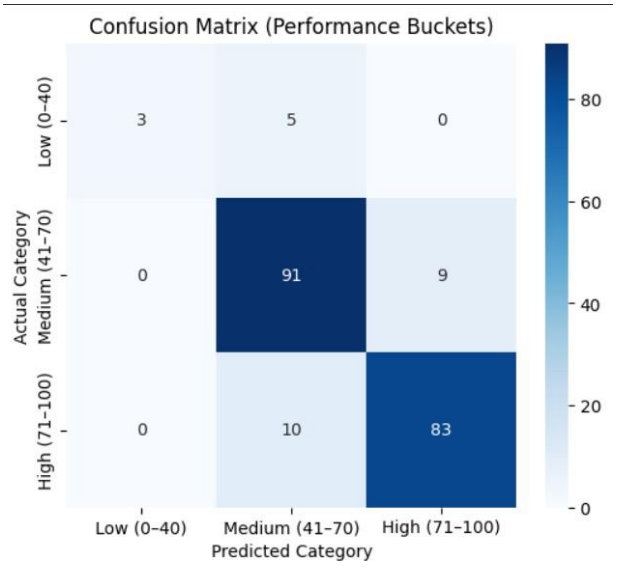

Fig8.2 :Confusion matrix

Model Evaluation Metrics:

Mean Absolute Error (MAE): 4.513

Mean Squared Error (MSE): 31.330

Root Mean Squared Error (RMSE): 5.597

R^2 Score: 0.863

8.2 Observations

Several insights were obtained during experimentation:

Students with borderline academic performance (scores close to pass/fail thresholds) were more difficult to classify, leading to occasional mislabeling.

Engagement-based features (VLE clicks, assessment submissions) significantly improved prediction stability.

Students with inconsistent participation patterns tended to produce lower prediction confidence.

The validation accuracy stabilized early, demonstrating that the model architecture and preprocessing pipeline were well-suited for the dataset.

8.3 Strengths

The system exhibits multiple strengths that contribute to its reliability:

Good prediction accuracy despite using a real-world, noisy educational dataset.

Robust preprocessing pipeline that handles missing values, categorical encoding, scaling, and derived features effectively.

Feature engineering significantly enhances performance, especially assessment-based metrics and VLE interaction counts.

Interpretability through permutation importance and clear metrics helps understand which factors influence prediction outcomes.

8.4 Limitations

Despite strong results, the model presents certain limitations:

Class imbalance affects performance, as more students tend to pass than fail, making the "Fail" class harder to predict accurately.

Quality of predictions heavily depends on data completeness; missing engagement or assessment data reduces accuracy.

Certain demographic or behavioral attributes may not carry enough predictive power on their own.

8.5 Future Improvements

Several enhancements can further strengthen the system:

Integrate temporal models (LSTMs or Transformers) to capture student progress trends over time.

Expand the feature set with additional behavioral indicators such as login frequency or time spent per learning activity.

Deploy the model as a dashboard for real-time monitoring and early intervention.

Incorporate explainable AI techniques such as SHAP to provide transparent predictions to educators.

Balance the dataset using advanced sampling methods or collect additional failure cases to reduce bias.

CHAPTER 9

CONCLUSION

The Student Performance Prediction system successfully demonstrates how machine learning and neural networks can be applied to identify at-risk students early in their academic journey. By integrating data preprocessing, feature engineering, and a structured neural network model, the system achieves strong predictive accuracy and effectively distinguishes between students likely to pass or fail. The use of behavioral indicators—such as assessment performance and virtual learning environment engagement—proved especially valuable in improving model reliability.

The project also highlights the importance of a clean and well-engineered data pipeline. Handling missing values, encoding categorical variables, scaling features, and aggregating assessment and VLE interactions all contributed significantly to model performance. The evaluation results confirm that the system generalizes well across unseen data and provides insights that educators can use for targeted academic intervention.

Overall, the work demonstrates the feasibility and usefulness of predictive analytics in education. With further enhancements such as additional behavioral features, temporal learning models, and deployment on real-time platforms, the system has the potential to become a practical tool for academic institutions aiming to improve student outcomes.

CHAPTER 10

REFERENCES

1. T. K. Ho, "Random decision forests," *Proceedings of 3rd International Conference on Document Analysis and Recognition*, Montreal, Canada, 1995, pp. 278–282.
2. Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
3. F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
4. D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *International Conference on Learning Representations (ICLR)*, 2015.
5. F. Chollet, "Keras: The Python Deep Learning Library," *GitHub repository*, https://keras.io/.
6. Google Brain Team, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Available: https://www.tensorflow.org/.
7. João Pedro Simões and Luís Bernardino, "The OULAD Dataset: Open University Learning Analytics Dataset," *Scientific Data*, The Open University, 2017. Available: https://analyse.kmi.open.ac.uk/open_dataset.
8. T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.
9. I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016.
10. G. H. John and P. Langley, "Estimating continuous distributions in Bayesian classifiers," *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, pp. 338–345, 1995.
11. J. Brownlee, "A Gentle Introduction to Machine Learning for Predictive Analytics," Machine Learning Mastery, 2020. Available: https://machinelearningmastery.com/.
12. OpenCV Development Team, "OpenCV Documentation," Available: https://opencv.org/.
13. J. Bergstra and Y. Bengio, "Random Search for Hyper-Parameter Optimization," *Journal of Machine Learning Research*, vol. 13, pp. 281–305, 2012.
14. M. B. C. Thomas and P. Hernández-Leo, "Predicting Student Performance Using Learning Analytics: A Systematic Review," *Educational Technology & Society*, vol. 25, no. 3, pp. 1–15, 2022.
15. V. J. Shute and D. Zapata-Rivera, "Adaptive educational systems," in *Handbook of Research on Educational Communications and Technology*, Springer, 2008, pp. 277–294