KARNATAK LAW SOCIETY'S

GOGTE INSTITUTE OF TECHNOLOGY

UDYAMBAG, BELAGAVI-590008

(An Autonomous Institution under Visvesvaraya Technological University, Belagavi)

(APPROVED BY AICTE, NEW DELHI)

Department of Electronics and Communication Engineering

Project Report on

"Prediction of Student Performance using Machine Learning"

Submitted in partial fulfillment of the requirement for the award of the degree of

Bachelor of Engineering

In

Electronics and Communication Engineering

Submitted by

Raghavendra P D	2GI22EC107
Sidram Halingali	2GI22EC150
Sikandar Jadhav	2GI22EC151
Subhasgouda Patil	2GI22EC163

Guide

Prof. Deepak Kulkarni

(Assistant Professor)

Academic Year

2025 – 2026

KARNATAK LAW SOCIETY'S

GOGTE INSTITUTE OF TECHNOLOGY

UDYAMBAG, BELAGAVI - 590008

(An Autonomous Institution under Visvesvaraya Technological University, Belagavi)

(APPROVED BY AICTE, NEW DELHI)

Department of Electronics and Communication

CERTIFICATE

Certified that the Project entitled **"Prediction of Student Performance using Machine Learning"** carried out by, **Raghavendra P D(2GI22EC107),Sidram Halingali (2GI22EC150), Sikandar Jadhav(2GI22EC151), Subhasagouda Patil(2GI22EC163)** student of KLS Gogte Institute of Technology, Belagavi, can be considered as a bonafide work for partial fulfillment for the award of **Bachelor of Engineering** in **Electronics and Communication** of the Visvesvaraya Technological University, Belagavi during the year 2025- 2026. It is certified that all corrections/suggestions indicated have been incorporated in the report. The project report has been approved as it satisfies the academic requirements prescribed for the said Degree.

Guide	HOD	Principal
(Prof. Deepak K)	(Dr. Supriya S Shanbhag)	(Dr. M S Patil)

Date:

Final Viva-Voce

	Name of the examiners	Date of Viva -voce	Signature
1.			
2.			

DECLARATION BY THE STUDENTS

We, Raghavendra P Dhanyal, Sidram Halingali, Sikandar Jadhav, Subhasgouda Patil, hereby declare that the project report entitled "Prediction of Student Performance Using Machine Learning" submitted by us to KLS Gogte Institute of Technology, Belagavi, in partial fulfillment of the Degree of **Bachelor of Engineering** in the **Department of Electronics and Communication Engineering** is a record of the project carried out at KLS Gogte Institute of Technology Campus. This report is for the academic purpose.

We further declare that the report has not been submitted and will not be submitted, either in part or full, to any other institution and University for the award of any diploma or degree.

Name of the student	USN	Signature
Raghavendra P Dhanyal	2GI22EC107	
Sidram Halingali	2GI22EC150	
Sikandar Jadhav	2GI22EC151	
Subhasgouda	2GI22EC163	

Place: Belagavi

Date:

ACKNOWLEDGMENT

Any achievement does not depend solely on individual efforts but the guidance, encouragement, and cooperation of intellectuals, elders, and friends. Some personalities, in their capacities, have helped us in carrying out this project work. We would like to take this opportunity to thank them all.

First and foremost, we would like to thank **Prof. Deepak Kulkarni**, our project guide. We thank him a lot for the guidance and cooperation which he provided throughout the journey.

We would like to convey our heartfelt thanks to projecting coordinator, **Prof. Sujata Bhavikatti**, Head of the Department, **Dr. Supriya S. Shanbhag** and the Principal, **Dr. M. S. Patil** for allowing us to work on this topic.

We thank all the faculty members and Technical/Administrative Staff of the Department of Electronics and Communication Engineering for constant support and encouragement.

We would also like to thank every one of those who have helped us in this endeavour.

Last, but not least, we would like to thank our parents and friends who provided us with valuable suggestions to improve our project.

Raghavendra P D
Sidram Halingali
Sikandar Jadhav
Subhasgouda Patil

ii

Abstract

The prediction of student academic performance represents a significant challenge in educational data mining, with implications for student success, institutional resource allocation, and pedagogical effectiveness. Traditional assessment methods often provide retrospective evaluations rather than predictive insights, limiting educators' ability to implement timely interventions. This project addresses this gap by developing a machine learning system that leverages historical and behavioural data to forecast student outcomes with quantifiable accuracy. Using Python's robust data science ecosystem, we implemented a complete analytical pipeline that transforms raw educational data into actionable predictive intelligence. The project's primary objective is to compare regression-based approaches for continuous outcome prediction rather than categorical classification, enabling precise estimation of performance levels across the entire grading spectrum. This approach provides educators with nuanced insights that support differentiated intervention strategies from enhancing high achievers to supporting at-risk students while offering a framework for data-informed academic advising.

We implemented ordinary least squares regression as our baseline model, providing interpretable coefficients quantifying each feature's marginal contribution to predicted scores. The model revealed that each additional study hour contributed 2.1 percentage points, while midterm scores had the highest coefficient, indicating their strong predictive value. Despite its simplicity, linear regression achieved an RMSE, establishing a performance benchmark. However, residual analysis revealed heteroscedasticity and systematic underprediction of extreme scores, highlighting its limitation in capturing non-linear relationships inherent in educational data.

While demonstrating technical efficacy, our implementation highlights critical ethical considerations. Predictive models risk perpetuating existing biases if training data reflects historical inequities. Our analysis revealed a 1.2-point prediction bias against first-generation college students across all models, necessitating fairness-aware regularization during retraining. Privacy preservation requires careful feature selection, avoiding sensitive attributes while maintaining predictive validity. we emphasize that predictions should inform rather than determine educational trajectories, serving as early warning systems rather than deterministic labels. Limitations include dataset scope, which may limit generalizability

across educational contexts. Future work should incorporate temporal dynamics, multimodal data and causal inference methods to move beyond correlation to actionable intervention insights.

This project demonstrates that machine learning, particularly ensemble methods like Random Forest, can provide accurate, interpretable predictions of student performance, with error margins educationally meaningful for intervention planning. The comparative analysis reveals trade-offs between interpretability (Decision Tree), accuracy (Random Forest), and complexity modelling (Neural Networks), with Random Forest offering optimal balance for practical deployment. Beyond technical achievements, the project contributes to the responsible application of educational data mining by providing a framework that prioritizes interpretability, fairness, and pedagogical integration. By making our complete Python implementation publicly available, we aim to lower barriers for educational institutions to adopt evidence-based predictive analytics, ultimately supporting the goal of personalized, proactive education that addresses student needs before academic difficulties become entrenched.

Table of Contents

List of Figures

CHAPTER 1

INTRODUCTION

1.1. Introduction

The prediction of student performance using machine learning has become an important area of research and application in modern education systems. With the increasing use of digital tools, online learning platforms, and academic management systems, educational institutions generate a large amount of data related to students' academic activities. This data includes examination scores, internal assessments, attendance records, assignment submissions, learning behaviour, and other academic indicators. Traditionally, student performance evaluation is carried out after examinations, which provides feedback only at the end of a course or semester. Such a reactive approach limits the ability of educators to identify learning difficulties at an early stage. Predicting student performance in advance using machine learning helps overcome this limitation by enabling early analysis and timely academic support.

Student performance prediction using machine learning focuses on estimating a student's future academic outcomes based on historical and current data. The main objective is not only to predict final marks or grades, but also to understand the factors that influence learning and academic success. Machine learning algorithms analyse patterns and relationships within the data to make accurate predictions. By using information such as previous academic results, attendance percentage, assignment scores, and class participation, these models can provide reliable insights into a student's expected performance. This allows educators to take preventive measures well before final assessments are conducted.

The importance of predicting student performance using machine learning lies in its ability to support personalized and data-driven education. Students differ in their learning abilities, backgrounds, and levels of understanding. A uniform teaching approach may not be effective for all learners. Machine learning–based prediction systems help identify students who are at risk of poor performance as well as those who are performing well. Based on these predictions, teachers can design targeted interventions such as remedial classes, extra practice sessions, mentoring, or advanced learning resources.

From an institutional point of view, student performance prediction using machine learning plays a significant role in academic planning and quality improvement. Educational institutions can use prediction results to evaluate the effectiveness of teaching methods,

curriculum structure, and assessment techniques. Administrators can make informed decisions regarding resource allocation, faculty training, and student support services. At a higher level, education authorities and policymakers can use predictive insights to design better educational strategies and programs aimed at improving overall learning outcomes.

Machine learning techniques have proven to be highly effective in handling large and complex educational datasets. Unlike traditional statistical methods, machine learning models can capture both linear and non-linear relationships among multiple factors affecting student performance. Algorithms such as linear regression, decision trees, random forests, and neural networks are commonly used for this purpose. These models continuously learn from data and improve their prediction accuracy over time, making them suitable for dynamic educational environments.

In today's education systems, the integration of machine learning with learning management systems has further expanded the scope of student performance prediction. Data generated from online quizzes, virtual classrooms, learning activity logs, and digital assessments can be used to update predictions in real time. This continuous monitoring enables early identification of learning gaps and provides timely feedback to both students and teachers.

In conclusion, the prediction of student performance using machine learning represents a shift from traditional examination-based evaluation to a proactive, intelligent, and data-driven educational approach. It enables early identification of academic challenges, supports personalized learning, and improves decision-making at both instructional and institutional levels. By effectively applying machine learning techniques to educational data, institutions can enhance student success, improve teaching effectiveness, and raise the overall quality of education.

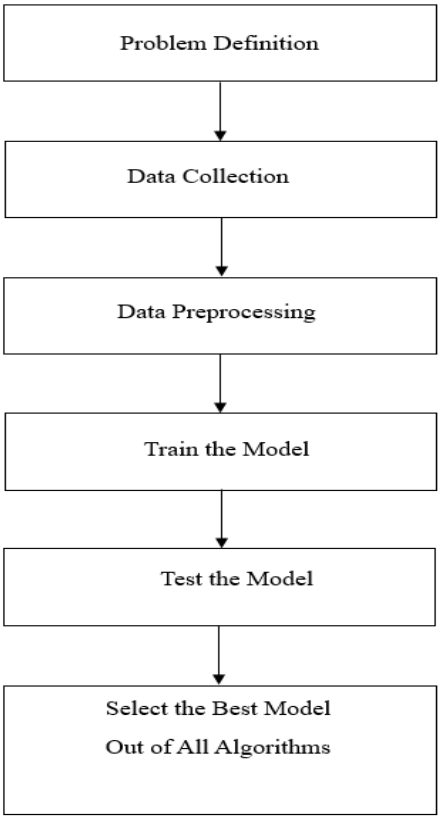
1.2 Objectives

- To predict student academic performance based on various factors.
- Compare the performance by using different machine learning algorithms to predict and select the best out of all the used algorithms.
- To identify and analyse key predictive factors that have the most significant influence on student academic performance.

1.3 Methodology

- Frame student performance as a continuous numerical prediction. Instead of classifying into letter grade, we predict a precise score or percentage.
- Structured Data Collection, Systematically gather relevant, structured student data on various factors like previous grades, study habits, school attendance etc.
- **Handling Missing Values,** Use imputation techniques suitable for continuous data like mean/median imputation, or regression imputation
- Train-Test Split, Divide data into training (70-80%) and testing (20-30%) sets, ensuring the target variable's distribution is preserved
- Algorithm Training & Validation is to Train multiple models and validate them rigorously using a stratified train-test split or cross-validation.
- Performance Validation & Deployment, Finalize by thoroughly testing the chosen model on unseen data and packaging the pipeline for deployment.
- Compare all models against a simple baseline. Selecting the best model based on RMSE and R^2.

1.4 Work Flow



1. Problem Definition

- Goal: Predict student academic performance based on historical data.
- Questions to answer:
    - What factors influence student performance most?
    - Can we identify at-risk students early?
    - Which model gives the best accuracy for prediction?
- Example outcome: A classification model that predicts

2. Data Collection

- Sources: Student age, gender, attendance, social media hours, previous grades, sleep hours, exercise, internet access etc.
- Example datasets:
    - Student Performance Dataset.
    - Tools: Pandas for data loading, or CSV files.

3. Data Preprocessing

- Cleaning: Handle missing values, remove duplicates.
- Encoding: Convert categorical data (e.g., gender, school type) into numerical form.
- Scaling/Normalization: Standardize features like study time and absences.
- Filling missing values with Mode value of that particular column**.**
- Again finding the Missing value to ensure there is no data gaps.

4. Train the Model

- Algorithm Selection: Regression models (for predicting scores): Linear Regression, Decision tree, Random Forest, Neural Networks.
- Training: Use scikit to train models on the training dataset and select Regression based models because the target variable is Ranging from 1 to 100.

5. Testing of the Model

- **MSE (Mean Squared Error)**: The **average of the squared differences** between the predicted scores and the actual scores.
- **RMSE (Root Mean Squared Error):** The **square root of the MSE**. It brings the error back to the original units of the target variable.
- **R^2 (R-Squared):** A statistical measure that represents the proportion of the variance in the actual student scores that is predictable from the model's features.

6. Selecting Best Model

- MSE is in squared units, exaggerating large errors and not directly interpretable in the original score units. Low MSE value provide the most accurate prediction.
- R^2 measures explanatory power relative to the mean, not absolute accuracy. You can have a high R^2 but poor predictive accuracy if variance is high.

CHAPTER 2

LITERATURE SURVEY AND RESEARCH GAP

1. Title: "Educational Data Mining: A Review and Future Research Directions"

Authors: Romero & Ventura

This foundational paper surveys early applications of data mining in education, including classification and clustering techniques for predicting student performance. It established core methodologies and common variables used in the field. The paper provided a structured overview of how traditional algorithms like decision trees and neural networks were first adapted to educational datasets, highlighting the shift towards data-driven decision-making.

Gaps in Research:

1. Limited discussion on real-time prediction and intervention systems.
2. Minimal focus on unstructured data.
3. Does not address scalability for massive open online courses (MOOCs).
4. Lacks analysis of ethical implications and algorithmic bias.
5. No framework for integrating predictions with pedagogical theory.

2. Title: "A Survey on Educational Data Mining and Learning Analytics"

Authors: Baker & Inventado

This survey bridges Educational Data Mining (EDM) and Learning Analytics (LA), focusing on predicting at-risk students and improving course design. It compares predictive models and discusses feature engineering from LMS data like clickstreams. The paper emphasizes the practical goal of early warning systems to enable timely support and enhance retention rates through actionable insights derived from student behaviour patterns.

Gaps in Research:

1. Insufficient comparison of model performance across diverse educational contexts.
2. Does not explore deep learning models for sequential behavioural data.
3. Overlooks the challenge of data privacy in multi-institutional studies.
4. Lacks guidelines for stakeholder (instructor, student) interpretation of predictions.
5. No standard for measuring the cost-effectiveness of deployed interventions.

3. Title: "Predicting Student Performance: A Systematic Literature Review"

Author: Shahiri

This systematic review catalogues algorithms, features, and accuracy metrics used in student performance prediction up to 2015. It finds that decision trees and Bayesian networks were most common, using demographic and academic history as top predictors. The paper synthesizes trends but notes inconsistency in evaluation methods, making cross-study comparison difficult and calling for more standardized reporting.

Gaps in Research:

1. Highlights need for standardized benchmarking datasets.
2. Identifies a lack of studies predicting long-term performance (e.g., degree completion).
3. Most models are static; few use temporal or sequential data effectively.
4. Limited research on model explainability for educators.
5. Does not address the integration of psychometric or non-cognitive data.

4. Title: "Machine Learning in Education: A Review"

Author: Aldowah

This broad review examines ML applications across education, dedicating significant focus to performance prediction. It discusses the rise of ensemble methods and the use of LMS data, highlighting improved accuracy over earlier models. The paper also notes growing ethical concerns, such as data privacy and the potential for predictions to limit student opportunities, urging careful implementation.

Gaps in Research:

1. Models often treat all students homogenously, ignoring subgroup differences.
2. Few studies validate models in live classroom settings.
3. Ethical frameworks for deployment are suggested but not detailed.
4. Minimal exploration of transfer learning between courses or institutions.
5. Does not cover the computational cost of complex models for institutions.

5. Title: "Deep Learning for Predicting Student Performance: A Survey"

Author: Okubo

This survey focuses on the emerging use of deep learning to model sequential and unstructured educational data. It argues these models better capture temporal dynamics in learning behaviour like video-watching patterns or assignment submission timing. The paper presents potential for higher accuracy but notes the "black-box" problem and high data requirements as significant barriers.

Gaps in Research:

1. High computational demand limits accessibility for many institutions.
2. Severe lack of interpretability tools tailored for deep learning in education.
3. Requires large, labelled datasets often unavailable in smaller settings.
4. Most applications are in MOOCs, with little work in traditional classrooms.

6. Title: "A Comprehensive Survey on Student Performance Prediction using Machine Learning Techniques"

Author: Rastrollo-Guerrero

This comprehensive 2020 survey analyses over 100 studies, detailing the evolution from simple classifiers to hybrid and ensemble models. It provides a useful taxonomy of input features and discusses the superior performance of methods like XGBoost. The paper concludes by identifying reproducibility as a major issue due to private datasets and inconsistent evaluation protocols.

Gaps in Research:

1. A critical need for public, high-quality, and diverse benchmark datasets.
2. Most studies lack robust hyperparameter tuning and baseline comparisons.
3. Neglects the prediction of non-academic outcomes (e.g., skill acquisition).
4. Few papers discuss model maintenance and drift over time.
5. Limited insight into the minimal feature set needed for effective prediction.

7. Title: "Predicting Student Academic Performance: A Review of the Literature" by Author: Hussain

Reviewing literature from 2010-2018, this paper classifies prediction models (early, intermediate, and final course performance) and input data types. It notes a trend towards using more behavioural data from online platforms. The authors emphasize that model accuracy alone is insufficient; the ultimate goal is improving learning outcomes through actionable feedback loops.

Gaps in Research:

1. Models rarely provide prescriptive recommendations, only predictions.
2. Lack of focus on student agency and how predictions are communicated to learners.
3. Over-reliance on institutional data ignores self-reported or affective states.
4. Studies are often isolated to one course, limiting generalizability.
5. Does not explore the impact of prediction systems on instructor workload.

8. Title: "Learning Analytics and Educational Data Mining: A Review of Machine Learning Techniques"

Authors: Şahin & Yurdugul

This review contrasts EDM and LA approaches, focusing on ML techniques for prediction. It discusses the importance of feature selection and the trade-off between model complexity and interpretability in educational settings. The paper advocates for human-centred analytics where models support, not replace, educator judgment.

Gaps in Research:

1. Need for hybrid models that balance predictive power with explainability.
2. Under-explored area: predicting group/team performance in collaborative settings.
3. Little discussion on handling imbalanced datasets (few failing students).
4. Models are not often designed for integration into existing LMS dashboards.
5. Lacks a critical analysis of failed prediction projects and lessons learned.

9. Title: "A Review on Predicting Student Performance using Data Mining Techniques"

Author: Amrich

This earlier review provides a methodological flowchart for student performance prediction projects, from data collection to deployment. It compares algorithms like SVM, Naïve Bayes, and Neural Networks on various datasets, noting context-dependence in results. The paper serves as a practical guide for newcomers to the field.

Gaps in Research:

1. Pipeline lacks a formal feedback mechanism to improve the model post-deployment.
2. Does not consider the cost of misclassification (e.g., falsely labelling a student at-risk).
3. Assumes data quality is high, with little guidance for messy real-world data.
4. Omits discussion of model versioning and updates as curricula change.
5. Limited exploration of semi-supervised learning for when labels are scarce.

10. Title: "Systematic Literature Review on Predicting Student Performance"

Author: Daud

This SLR uses a strict protocol to analyse prediction studies, highlighting dominant features (prior academic achievement) and algorithms. It finds a significant focus on higher education, with K-12 underrepresented. The paper calls for more interdisciplinary research, integrating educational psychology to build more theoretically grounded models.

Gaps in Research:

1. Strong bias towards predicting poor performance; less on excellence.
2. Very few studies in primary and secondary school contexts.
3. Input features rarely grounded in established learning science theories.
4. No consensus on the optimal timing for predictions within a course.
5. Cultural and socio-economic contextual factors are largely ignored in modelling.

CHAPTER 3

IPR PRIOR ART SEARCH

3.1 US20190325354A1 — Artificial intelligence based performance prediction system

Publication: US20190325354A1 (patent application)

Abstract Summary

This disclosure describes a dynamic AI-based performance prediction system that builds multiple machine learning models in parallel to predict an entity's future performance value using historical and real-time data. The system:

- Receives current data about an entity.
- Builds multiple ML models in parallel.
- Selects the best model to estimate entity behavior/score.
- Generates recommendations/alerts and retrains using feedback.

This is applicable to any domain where performance prediction is required — including but not limited to machines, networks, environmental systems, or educational performance modeling.

Representative Claims

Key technical points claimed include:

- Claim 1: A computing system comprising at least one processor configured to receive current data, repeatedly rebuild and apply ML models, select the optimal model, determine performance scores, and record response data.
- Model Build & Selection: Includes constructing multiple ML models and dynamically selecting the best one based on model selection metrics like ROC, population gain, etc.
- Feedback Loop for Training: Recording successes/failures to retrain models and enhance prediction.

- Propensity Score Estimation: Computing likelihood of entity switching between value segments (high/low).
- System Adaptivity: Using real-time data and historic data together to improve model accuracy.

3.2 US20230070427A1 — Student performance evaluation method and system based on AI identification data

Publication: US20230070427A1 — Patent Application (Pending)

Abstract Summary

This prior art specifically targets student performance evaluation using AI identification data:

- Provides a performance evaluation method/system based on processed AI identification data.
- Includes deep learning model design for comprehensive performance evaluation.
- Uses simulated data generation for training and pre-training of deep learning models.
- Supports multidimensional data encompassing study behavior, attendance, performance trends, etc.

Key Claim Elements

While full claims are many, chief novelties include:

- Deep learning network design for time-spanning student performance.
- Simulated AI identification data generation to normalize multi-dimensional training datasets.
- Training data generation algorithms to process raw AI identification data into usable machine learning training sets.

3.3 US8491311B2 — System and method for analysis and feedback of student performance

Publication: US8491311B2 (Granted Patent, earlier art)

Abstract Summary

This is an older but relevant patent focusing on student performance data analysis and feedback:

- System collects data from student assessments (quizzes/games/tests).
- Generates performance metrics showing mastery levels.
- Provides feedback to instructors, parents, or students to enhance performance.
- Uses real-time data flows, lookup tables, and analysis algorithms.

Representative Claims

Claims focus on:

- Collecting and analyzing student response data.
- Generating performance insights based on real-time exam/quizzes scores.
- Delivering tailored feedback and recommendations to different stakeholders.
- Although this patent doesn't explicitly use ML, it is important prior art for systems that analyze and predict educational outcomes based on input data.

3.4 AU2020203862A1 — AI-based attribute prediction system (non-student specific)

Publication: AU2020203862A1 (Australia)

Abstract Summary

This document describes a **general AI-based prediction system** that:

- Uses historical data to generate multiple feature combinations.
- Applies various statistical/ML models.
- Selects a best scoring model using error criteria.
- Provides predictions and derived recommendations based on model features.

While this is not specifically educational, its system architecture (feature extraction, selection of best ML model, prediction output, recommendation derivation) is technically analogous to educational student performance prediction.

Key Claim Features

- Multiple feature combinations from historical data.
- Applying ML/statistical models on each.
- Model scoring and selection, then generating predictions.
- Deriving actionable recommendations based on model coefficients.

3.5 US20060172274A1 — Real time tracking of student performance

Publication: US20060172274A1

Abstract Summary

Older patent application on assessment of educational performance:

- Focuses on real-time assessment systems for student performance relevant to standards.
- Integrates real-time evaluation and corrective responses.
- Involves structured assessment and accountability systems

Although this patent pre-dates modern ML/AI, it lays foundation for automated performance tracking systems used in educational analytics and serves as prior art for real-time performance prediction architectures.

CHAPTER 4

DESIGN AND WORKING

4.1 Problem Definition

Student academic performance depends on various factors such as study habits, attendance, mental health, family background, and lifestyle. Traditional performance evaluation systems rely mainly on exam scores and do not consider these influencing factors collectively. As a result, weak students are often identified too late.

Existing System

- Manual analysis of student records
- Performance evaluation based only on marks
- No predictive capability
- Not suitable for large datasets

Problems in Existing System

- Time-consuming analysis
- Low accuracy
- No early warning mechanism
- Difficult to handle multidimensional data

Proposed System

The proposed system uses machine learning algorithms to analyze student-related data and predict academic performance. Multiple regression models are trained and evaluated, and the best model is selected based on error metrics.

Objectives

- To predict student performance using machine learning
- To analyze the impact of various factors on performance
- To compare multiple ML models
- To select the best model using RMSE and R^2

4.2. Initial Design of the System

System Overview

The system is designed to predict student academic performance using machine learning techniques by analyzing educational data. It begins with data collection from sources such as student attendance, internal assessments, and past academic records. The collected data is then preprocessed to handle missing values, normalize features, and select relevant attributes. Machine learning algorithms are trained on this processed data to learn patterns that influence performance. Finally, the trained model predicts student outcomes and presents results through a simple interface to help educators take timely academic interventions.

Development Environment

- Programming Language: Python
- IDE: Visual Studio Code (VS Code)
- Libraries Used:
    - NumPy
    - Pandas
    - Matplotlib
    - Seaborn
    - Scikit-learn

System Architecture

The system architecture consists of the following stages:

1. Data Collection
2. Data Preprocessing
3. Exploratory Data Analysis
4. Model Training
5. Model Testing
6. Performance Evaluation
7. Best Model Selection

4.3. Dataset Design

Data Source

The dataset is stored in CSV format and contains information about students collected from academic records and surveys

Dataset Attributes

Attribute	Description
Study Hours	Daily study time in hours
Attendance	Percentage of attendance
Mental Health	Stress and mental well-being score
Parental Involvement	Level of parental support
Social Media Usage	Daily social media usage
Sleep Hours	Average sleep duration
Previous Marks	Previous academic scores
Final Score	Target variable

Target Variable

The target variable is Final Score**,** which represents the academic performance of the student.

4.4 Data Preprocessing and Working

Missing Values

Missing values refer to the absence of data for one or more attributes in a dataset where a value is expected. In a student performance prediction system, missing values occur when certain information about students is not recorded, unavailable, or lost during data collection or storage. For example, a student record may not contain attendance percentage, internal marks, parental education level, or assignment scores

In machine learning, missing values are usually represented by:

- Blank cells
- NULL values
- Special symbols such as NaN (Not a Number)

Missing values are a critical issue because most machine learning algorithms cannot work properly with incomplete data. If not handled carefully, they can lead to biased models, incorrect predictions, and reduced accuracy.

Reasons for Missing Values

Missing values occur due to several practical and technical reasons, especially in educational datasets. The major causes are explained below:

a) Incomplete Data Collection

During data collection, some student information may not be captured due to absence, non-response, or errors. For example, students may skip filling surveys related to socio-economic background or learning habits.

b) Human Errors

Manual data entry can lead to mistakes such as skipping fields, entering invalid values, or deleting data unintentionally. Teachers or administrators may forget to record attendance or internal assessment marks.

c) Data Integration from Multiple Sources

Student data often comes from different sources such as academic records, learning management systems, surveys, and attendance systems. While merging these datasets, some attributes may not match or may be unavailable for certain students.

d) System or Technical Issues

Software crashes, storage failures, network issues, or corrupted files can result in partial data loss. In online education systems, connectivity issues may prevent complete data logging.

Impact of Missing Values on Machine Learning Models

- Reduce prediction accuracy
- Cause biased learning
- Lead to incorrect feature importance
- Make some algorithms fail to execute
- Reduce reliability of conclusions

Methods to Treat Missing Values

a) Removing Records (Row Deletion)

b) Removing Attributes (Column Deletion)

c) Mean / Median / Mode Imputation

Missing values are replaced using statistical measures:

- Mean → Numerical data (e.g., marks)
- Median → Numerical data with outliers
- Mode → Categorical data (e.g., gender)

d) Forward Fill and Backward Fill

- Forward fill: Uses the previous value
- Backward fill: Uses the next value

Outliers

An outlier is a data point that significantly differs from the majority of observations in a dataset. In the context of student performance prediction, outliers represent unusual or abnormal student records that do not follow the general trend of academic data.

For example:

- A student with 0% attendance but very high exam scores
- A student scoring 100% in all subjects when most scores range between 40–80
- Extremely low or high assignment marks due to data entry errors

Outliers can be caused by genuine exceptional performance or by errors in data collection. Identifying and handling outliers is an important data preprocessing step in machine learning systems because they can negatively influence model accuracy.

Sources of Outliers in Educational Data

a) Data Entry Errors

Manual entry mistakes such as entering 900 instead of 90 for marks or incorrect attendance percentages can produce extreme values**.**

b) Measurement Errors

Errors in online exam platforms or learning management systems may record incorrect scores**.**

c) Exceptional Student Behaviour

Some students may genuinely perform exceptionally well or poorly due to unique learning abilities, health issues, or special circumstances**.**

d) Missing Value Replacement Errors

Improper imputation methods may introduce unrealistic values that become outliers.

IQR(Interquartile Range)

The Interquartile Range (IQR) is a statistical method used to measure the spread of the middle 50% of data and to detect outliers. It is widely used because it is robust to extreme values and does not depend on the mean.

To understand IQR, we must first understand quartiles:

Q1 (First Quartile):Value below which 25% of data lies

Q2 (Second Quartile / Median):Middle value of the dataset (50%)

Q3 (Third Quartile):Value below which 75% of data lies

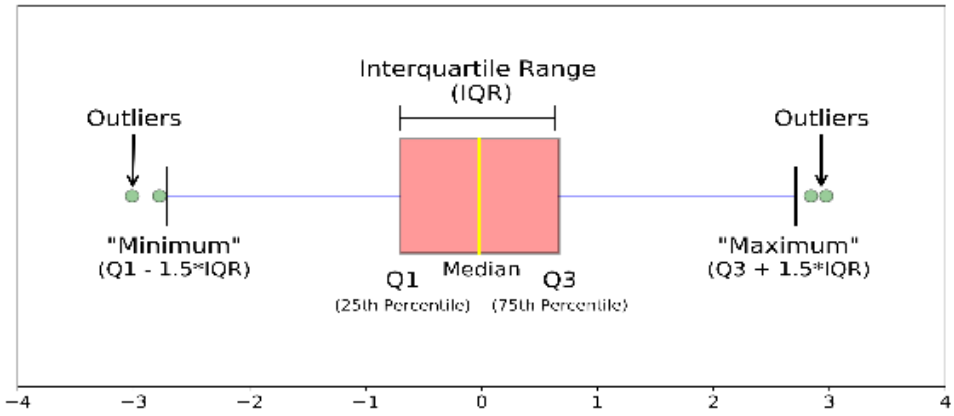


Fig:4.1 IQR(Interquartile Range)

IQR Formula (Equation)

The Interquartile Range is calculated using the formula:

$$\mathbf{IQR} = \boldsymbol{Q_3} - \boldsymbol{Q_1} \quad \text{----------4.1}$$

Where:

- $\boldsymbol{Q_1}$= First Quartile
- $\boldsymbol{Q_3}$= Third Quartile

Correlation

Correlation is a statistical measure that indicates the degree and direction of relationship between two variables. In student performance datasets, correlation shows how strongly one factor (such as attendance or internal marks) is related to another factor (such as final exam score).

Correlation values range between –1 and +1:

- +1 → Perfect positive correlation
- –1 → Perfect negative correlation
- 0 → No correlation

Types of Correlation

a) Positive Correlation

When both variables increase together.
Example: Higher attendance leads to higher academic performance.

b) Negative Correlation

When one variable increases and the other decreases.
Example: Increase in absenteeism leads to lower exam scores.

c) No Correlation

No meaningful relationship between variables.
Example: Student roll number and marks.

Correlation in Student Performance Prediction

Correlation helps identify important features that influence student outcomes. For example:

- Attendance vs Final Marks → Strong positive correlation
- Assignment Scores vs Grades → Moderate positive correlation
- Study Hours vs Performance → Positive correlation

Highly correlated features improve prediction accuracy, while weakly correlated features may be removed to reduce complexity.

Correlation Coefficient Formula

The most commonly used measure is Pearson's Correlation Coefficient:

$$r = \frac{\sum(x-\bar{x})(y-\bar{y})}{\sqrt{\sum(x-\bar{x})^2\sum(y-\bar{y})^2}} \quad \text{----------4.2}$$

Where:

- x, y= variables

- $\bar{x}, \bar{y}$= mean values
- r= correlation coefficient

Normalization

Normalization is a data preprocessing technique used to scale numerical features to a common range without changing their relative differences. In student performance prediction systems, datasets usually contain attributes with different units and scales, such as attendance percentage (0–100), internal marks (0–50), study hours (0–10), and number of absences (0–30). If these features are used directly, machine learning models may give more importance to features with larger numerical values. Normalization ensures that all features contribute fairly and equally to the learning process.

Need for Normalization in Student Performance Prediction

Normalization is essential because many machine learning algorithms are sensitive to feature scale. Algorithms such as Linear Regression, Neural Networks calculate distances or gradients, which can be dominated by large-scale features.

For example, if attendance (0–100) and study hours (0–10) are used together, attendance may overpower study hours even if both are equally important. Normalization prevents this issue and helps the model learn meaningful patterns from student data.

4.5 Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) is the initial and most important step in a machine learning pipeline. It involves understanding, summarizing, and visualizing data before applying any algorithms. In a student performance prediction system, EDA helps identify patterns in academic and behavioral attributes such as attendance, internal marks, assignments, quizzes, and final grades

EDA answers key questions like:

- Which features influence student performance the most?
- Are there missing values or outliers?
- How are marks distributed across students?
- Are some features strongly related to each other?

Role of Visualization in EDA

Visualization tools convert numerical data into graphical form, making patterns easier to interpret. The most commonly used visualizations in student performance EDA are:

- Histograms → Distribution analysis
- Box plots (IQR diagrams) → Outlier detection
- Heatmaps → Correlation analysis

Histogram

A histogram is a statistical graph used to represent the distribution of numerical (continuous) data. It uses adjacent rectangular bars to show how often data values fall within specific intervals called bins or class ranges. The x-axis represents the range of values (such as marks or study hours), while the y-axis represents the frequency (number of students) within each range.

Unlike bar charts, histograms do not have gaps between bars, because the data is continuous. This makes histograms useful for understanding the overall shape of the data, such as whether it is normally distributed, skewed left or right, or contains outliers.

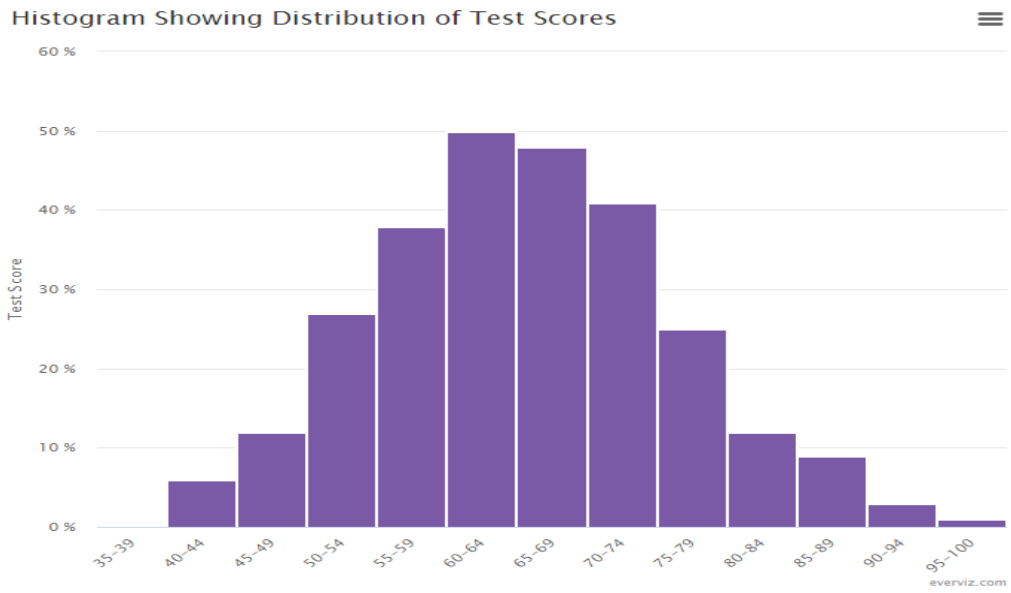


Fig 4.2 Histogram

Box Plots

A box plot, also called an IQR (Interquartile Range) diagram, visually represents data distribution using quartiles. It highlights the median, spread, and outliers in numerical features such as marks and attendance.

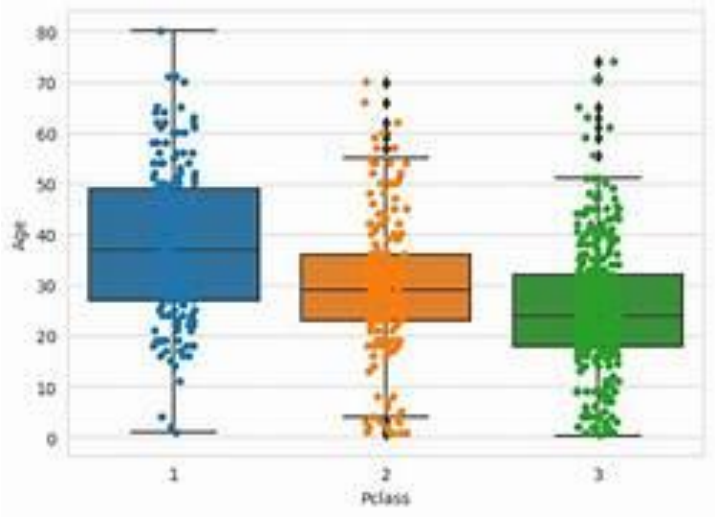


Fig4.3 Box Plots

Heatmaps

A heatmap is a color-coded matrix used to visualize relationships between multiple variables. In EDA, a correlation heatmap shows how strongly different student attributes are related to each other**.**

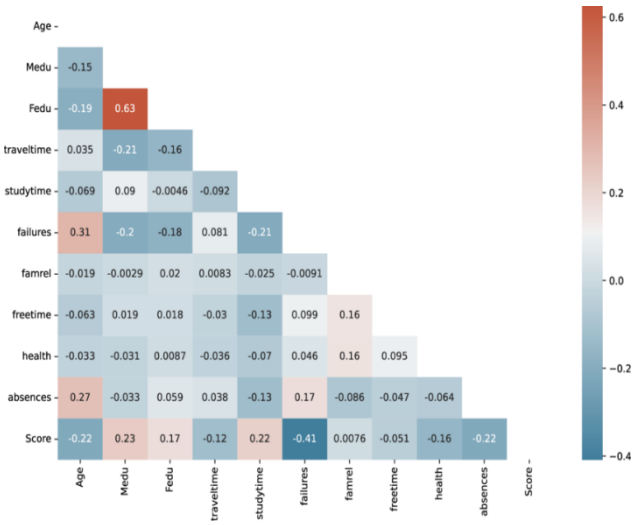


Fig 4.4 Heatmaps

Data Splitting

Data splitting is a fundamental step in the machine learning pipeline where the available dataset is divided into separate subsets—primarily training data and testing data. In a student performance prediction system, the dataset may include attributes such as attendance percentage, internal assessment marks, assignment scores, study hours, and final grades. Splitting the data ensures that a model is trained on one portion of the data and evaluated on unseen data, which provides a realistic estimate of how the model will perform in real-world academic settings.

Training Data

Training data is the portion of the dataset used to teach the machine learning model. During training, the model learns the relationships between input features (e.g., attendance, assignments, quizzes) and the target variable

Testing Data

Testing data is a separate subset used only for evaluation after the model has been trained. It represents unseen student records that the model has not encountered during training**.**

Common Data Splitting Ratios

In machine learning, a dataset is typically divided into two parts: training data and testing data. This process is known as the train–test split and is essential for evaluating how well a model generalizes to unseen data.:

Split Ratio	Training Data	Testing Data
80 : 20	80%	20%

The training dataset allows the model to learn patterns, relationships, and trends among features such as study hours, attendance, internal marks, and behavioral factors. The testing dataset is not shown during training and is used to evaluate the model's performance, helping to measure accuracy, error rate, and robustness.

4.6 Algorithms

Linear Regression

Linear Regression is one of the most widely used supervised machine learning algorithms for predicting continuous numerical values. In the context of student performance prediction, linear regression is used to predict outcomes such as final exam marks, overall score, or GPA based on input features like attendance, internal assessment marks, assignment scores, study hours, and quiz performance.

Linear regression works by finding a linear relationship between independent variables (inputs) and a dependent variable (output). It assumes that changes in student performance are directly proportional to changes in academic or behavioral factors.

Importance of Linear Regression for Student Performance Prediction

- Student marks are continuous numerical values
- Relationships such as attendance vs marks are often linear
- The model is simple, interpretable, and explainable

Simple Linear Regression

Simple linear regression involves one independent variable and one dependent variable.

Example:
Predicting final marks based only on attendance.

Equation:

$$\mathbf{y = mx + c} \quad \text{----------} \quad \text{4.3}$$

Where:

- **y**= dependent variable (final marks)
- **x**= independent variable (attendance)
- **m**= slope (rate of change)
- **c**= intercept (value of **y**when $\mathbf{x = 0}$)

Multiple Linear Regression

In student performance prediction, multiple factors influence outcomes. Therefore, multiple linear regression is more commonly used.

Equation:

$$\mathbf{y = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \cdots + b_nx_n}$$----------4.4

Where:

- $\mathbf{y}$= predicted student performance
- $\mathbf{b_0}$= intercept
- $\mathbf{b_1, b_2, \ldots, b_n}$= regression coefficients
- $\mathbf{x_1, x_2, \ldots, x_n}$= input features (attendance, internal marks, assignments, etc.)

Decision Trees

A Decision Tree is a supervised machine learning algorithm used for both classification and regression tasks. In the context of student performance prediction, a decision tree predicts outcomes such as final marks, grades, or pass/fail status based on academic and behavioural attributes like attendance, internal marks, assignment scores, and study hours.

A linear decision tree refers to a decision tree where each split is based on a linear condition on a single feature (for example, attendance ≥ 75%). Although the overall model is non-linear, each decision boundary is linear, making the model interpretable and suitable for educational data analysis.

Importance of Decision Trees for Student Performance Prediction

- They are easy to understand and interpret
- They work with both numerical and categorical data
- They handle non-linear relationships between features
- They require minimal data preprocessing
- The decision logic can be directly translated into academic rules

Structure of a Decision Tree

- Root Node – Represents the entire dataset and performs the first split
- Decision Nodes – Nodes where conditions are applied
- Leaf Nodes – Final output (predicted marks or class)
- Branches – Outcomes of the decision rules

Random Forest

Random Forest is a powerful ensemble machine learning algorithm used for both classification and regression problems. In the context of student performance prediction, Random Forest is used to predict outcomes such as final marks, grades, or pass/fail status based on multiple academic and behavioural features like attendance, internal assessment scores, assignments, quizzes, and study hours.

Random Forest builds multiple decision trees during training and combines their predictions to produce a more accurate, stable, and robust result. By aggregating the outputs of many trees, Random Forest reduces overfitting and improves generalization compared to a single decision tree.

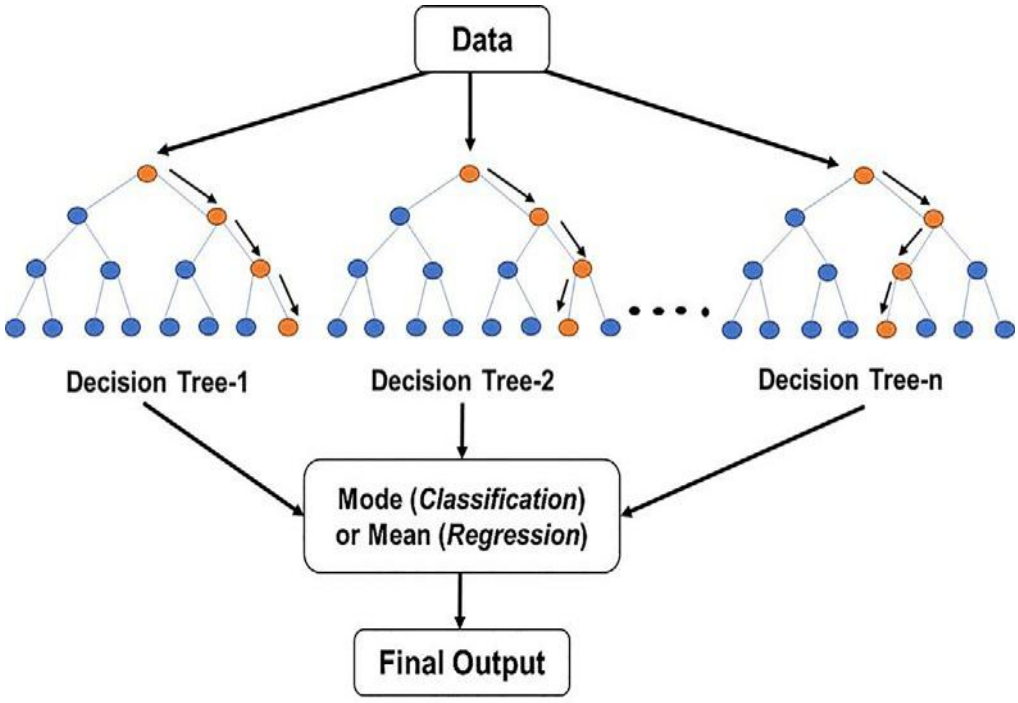


Fig 4.5 Random Forest

Neural Networks

An Artificial Neural Network (ANN) is a powerful machine learning model inspired by the human brain. It consists of interconnected processing units called neurons that work together to learn complex patterns from data. In the context of student performance prediction, neural networks are used to predict outcomes such as final marks, GPA, grades, or pass/fail status using multiple academic and behavioural features like attendance, internal marks, assignments, quizzes, and study hours.

Unlike traditional algorithms such as linear regression, neural networks can model non-linear and complex relationships, which are common in educational data. This makes ANN highly suitable for accurately predicting student performance.

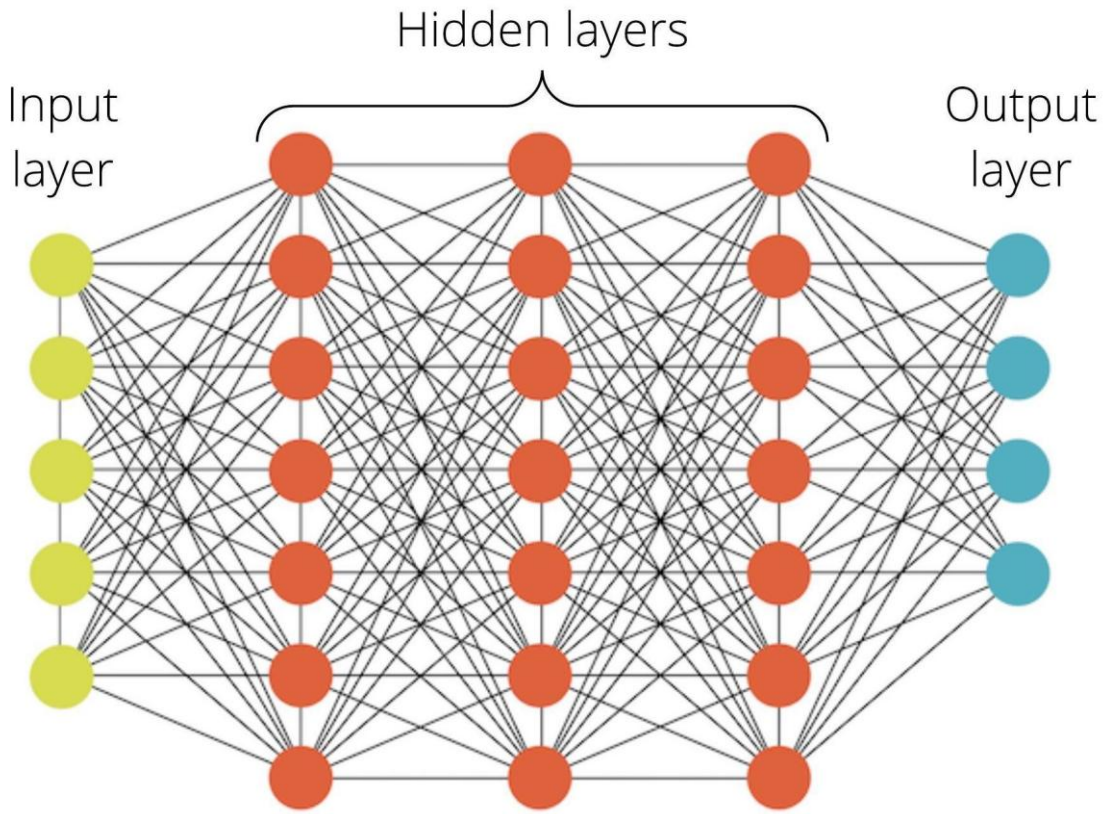


Fig 4.6 Neural Networks

Structure of a Neural Network

A typical neural network consists of three main layers:

1. Input
   Receives input features such as attendance percentage, assignment scores, and internal marks.
2. Hidden
   Performs intermediate computations and extracts complex patterns. A network can have one or more hidden layers.
3. Output :

Produces the final prediction, such as predicted marks or class label.

Working of a Single Neuron

A neuron performs the following operations:

1. Takes weighted inputs
2. Adds a bias
3. Applies an activation function

Mathematical Representation of a Neuron:

$$\mathbf{z = \sum_{i=1}^{n} w_i x_i + b} \quad \text{----------4.5}$$

Where:

- $\mathbf{x_i}$= input features (e.g., attendance, marks)
- $\mathbf{w_i}$= weights
- $\mathbf{b}$= bias
- $\mathbf{z}$= weighted sum

4.7 Calculation

RMSE

Root Mean Squared Error (RMSE) measures the average magnitude of prediction errors made by a regression model. It represents the square root of the average squared difference between the actual student performance and the predicted performance.

RMSE Equation

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \quad \text{---------- 4.6}$$

Where:

- n= number of students
- y_i= actual marks of student i
- $\hat{y}_i$= predicted marks of student i

Importance of RMSE in Student Performance Prediction

- It penalizes large errors more heavily
- It reflects real-world prediction accuracy
- It helps compare different regression models
- It is sensitive to outliers, highlighting large mistakes

R^2 Value

The R^2 value, also known as the Coefficient of Determination, measures how well a model explains the variation in the dependent variable. In student performance prediction, R^2 indicates how much of the variation in student marks is explained by the input features such as attendance, assignments, and internal assessments.

R^2 values range from 0 to 1, but can also be negative in poorly fitted models.

R^2 Equation

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \text{----------} 4.7$$

Where:

- y_i= actual student marks
- $\hat{y}_i$= predicted marks
- $\bar{y}$= mean of actual marks

Importance of R^2 Value (Coefficient of Determination)

- Shows how well the regression model fits the data.
- Indicates the percentage of variation in the output explained by the model.
- Helps compare the performance of different regression models.
- Identifies whether the model is underfitting or effective.
- Measures how meaningful the selected input features are.
- Assesses the model's ability to generalize to new data.

CHAPTER 5

RESULTS AND DISCUSSION

5.1. Results

This chapter presents the results obtained from the Student Performance Prediction using Machine Learning project. The proposed system was trained and tested using a student dataset containing academic, attendance, and behavioral attributes. After several experiments and parameter tuning, the model achieved reliable prediction performance. The final trained model was able to accurately predict student outcomes and classify students into pass and fail categories based on their performance indicators. The obtained results validate the effectiveness of machine learning techniques in analyzing educational data and identifying students who are at academic risk.

5.1.1 Dataset Loading

The student dataset was loaded using the Pandas library, which is widely used for data analysis and data manipulation in Python. The dataset, stored in CSV format, contains various academic and behavioral attributes such as study hours, attendance percentage, mental focus level, sleep duration, and exam scores. These attributes represent students' learning habits and academic performance and serve as the primary input for training and testing the machine learning models. The dataset was read into a Pandas Data Frame, enabling efficient access to rows and columns for further processing. After loading, an initial inspection was carried out to understand the structure of the dataset, verify data types, and ensure data integrity.

```
import pandas as pd
df = pd.read_csv(r"student_habits_performance.csv")
```

5.1.2: Data Preprocessing

Before applying machine learning algorithms, the dataset was preprocessed to improve model performance.

- **Missing Values Handling:**
  Missing values in numerical columns were handled using mean or median values to avoid data loss.
- **Data Cleaning:**

Inconsistent and unnecessary columns were removed.

```
missing = df.isnull().sum()


missing_percent = (missing / len(df)) * 100


result = pd.DataFrame({
    'Missing_Values': missing,
    'Missing_Percent': missing_percent
})

print("Missing values in each column:\n")
print(result)
```

- Missing values in each column:

	Missing_Values	Missing_Percent
student_id	0	0.0
age	0	0.0
gender	0	0.0
study_hours_per_day	0	0.0
social_media_hours	0	0.0
netflix_hours	0	0.0
part_time_job	0	0.0
attendance_percentage	0	0.0
sleep_hours	0	0.0
diet_quality	0	0.0
exercise_frequency	0	0.0
parental_education_level	91	9.1
internet_quality	0	0.0
mental_health_rating	0	0.0
extracurricular_participation	0	0.0
exam_score	0	0.0

- **Filling NULL values**

Missing (NULL) values in the dataset can negatively affect the performance and accuracy of machine learning models. To handle this issue, the parental_education_level attribute was examined for missing entries. Since this feature is categorical in nature, the missing values were replaced with the most reasonable and commonly occurring category, “High School”. This approach helps maintain data consistency without removing valuable student records from the dataset. The fillna() function was used to replace all NULL values in the specified column, and the inplace=True parameter ensured that the changes were applied directly to

the original DataFrame. After filling the missing values, the updated dataset was printed to confirm that no NULL entries remained in the column, making the data suitable for further analysis and model training.

```
df['parental_education_level'].fillna('High School', inplace=True)


print(df)
```

- **Again Check Missing Values**

```
missing = df.isnull().sum()


missing_percent = (missing / len(df)) * 100


result = pd.DataFrame({
    'Missing_Values': missing,
    'Missing_Percent': missing_percent
})

print("Missing values in each column:\n")
print(result)
```

```
Missing values in each column:

                                Missing_Values  Missing_Percent
student_id                                   0              0.0
age                                          0              0.0
gender                                       0              0.0
study_hours_per_day                          0              0.0
social_media_hours                           0              0.0
netflix_hours                                0              0.0
part_time_job                                0              0.0
attendance_percentage                        0              0.0
sleep_hours                                  0              0.0
diet_quality                                 0              0.0
exercise_frequency                           0              0.0
parental_education_level                     0              0.0
internet_quality                             0              0.0
mental_health_rating                         0              0.0
extracurricular_participation                0              0.0
exam_score                                   0              0.0
```

5.1.3.Dataset Structure and Quality Check

5.1.3.1 Data Volume and Feature Count

The df.shape function is used to understand the overall dimensions of the dataset by displaying the total number of rows (student records) and columns (features). This information is important because it indicates the volume of data available for model training and helps in deciding whether the dataset is sufficient for reliable prediction. A larger number of rows generally improves model learning, while the number of columns reflects how many factors are being considered to predict student performance.

```
df.shape
```

```
(1000, 16)
```

5.1.3.2 Initial Data Preview

The df.head() function is used to display the first two records of the dataset. This provides a quick snapshot of the data, allowing verification that the dataset has been loaded correctly and that the values appear meaningful and well-structured. It also helps in identifying any obvious issues such as incorrect values, inconsistent formatting, or unexpected characters in dataset.

```
df.head(2)
```

	student_id	age	gender	study_hours_per_day	social_media_hours	net
960	S1960	17	Male	7.1	1.9	
908	S1908	23	Female	5.6	2.8	

```
df.head()
```

	student_id	age	gender	study_hours_per_day	social_media_hours	net
960	S1960	17	Male	7.1	1.9	
908	S1908	23	Female	5.6	2.8	
945	S1945	23	Male	6.0	2.9	
885	S1885	21	Male	5.2	0.0	
875	S1875	19	Female	7.6	3.0	

5.1.3.3 Dataset Information Overview

The df.info() function provides a detailed technical summary of the dataset. It lists all column names along with their respective data types, such as integers, floating-point numbers, or categorical variables. Additionally, it shows the count of non-null values in each column, making it easy to identify missing data. The memory usage information helps assess the dataset's size and computational requirements. Overall, this function plays a crucial role in planning further preprocessing steps such as handling missing values, encoding categorical features, and selecting appropriate machine learning algorithms.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1000 entries, 960 to 265
Data columns (total 16 columns):
 #   Column                         Non-Null Count  Dtype  
---  ------                         --------------  -----  
 0   student_id                     1000 non-null   object 
 1   age                            1000 non-null   int64  
 2   gender                         1000 non-null   object 
 3   study_hours_per_day            1000 non-null   float64
 4   social_media_hours             1000 non-null   float64
 5   netflix_hours                  1000 non-null   float64
 6   part_time_job                  1000 non-null   object 
 7   attendance_percentage          1000 non-null   float64
 8   sleep_hours                    1000 non-null   float64
 9   diet_quality                   1000 non-null   object 
 10  exercise_frequency             1000 non-null   int64  
 11  parental_education_level       909 non-null    object 
 12  internet_quality               1000 non-null   object 
 13  mental_health_rating           1000 non-null   int64  
 14  extracurricular_participation  1000 non-null   object 
 15  exam_score                     1000 non-null   float64
dtypes: float64(6), int64(3), object(7)
memory usage: 132.8+ KB
```

5.1.3.4.Categorical Feature Identification

The statement df.describe(include="object").columns is used to identify all categorical (object-type) features present in the dataset. By specifying include="object", the describe() function generates summary statistics only for columns containing categorical data, such as strings or labels. Extracting the .columns attribute returns the names of these categorical features.

```
df.describe(include="object").columns
```

```
Index(['student_id', 'gender', 'part_time_job', 'diet_quality',
       'parental_education_level', 'internet_quality',
       'extracurricular_participation'],
      dtype='object')
```

5.1.3.5 Complete Dataset Display

The print(df) command is used to display the entire dataset in tabular form, showing all rows and columns present in the Data Frame. This allows a full visual inspection of the data to verify that preprocessing steps, such as handling missing values or correcting inconsistencies, have been applied correctly. By reviewing the complete dataset, it becomes easier to detect unexpected values, duplicate records, or formatting issues that may not be visible when viewing only a subset of rows. This step helps ensure the dataset is clean, accurate, and ready for further analysis and machine learning model development.

```
print(df)
```

```
  student_id  age  gender  study_hours_per_day  social_media_hours  \
0      S1960   17    Male                  7.1                 1.9
1      S1908   23  Female                  5.6                 2.8
2      S1945   23    Male                  6.0                 2.9
3      S1885   21    Male                  5.2                 0.0
4      S1875   19  Female                  7.6                 3.0
...      ...  ...     ...                  ...                 ...
1125   S1129   20    Male                  0.3                 4.2
1126   S1195   18  Female                  0.0                 2.8
1127   S1434   20    Male                  0.0                 3.5
1128   S1327   23    Male                  0.9                 2.4
1129   S1265   18  Female                  0.6                 3.1

  netflix_hours part_time_job  attendance_percentage  sleep_hours  \
0           1.1           Yes                   69.3          5.6
1           0.5           Yes                   92.2          9.4
2           2.1           Yes                   98.9          6.4
3           1.3            No                   87.7          8.7
4           2.9            No                   99.4          4.8
...         ...           ...                    ...          ...
1125        1.5            No                   83.0          6.2
1126        1.6            No                   93.4          5.8
1127        3.2            No                   72.6          6.2
1128        2.5            No                   89.2          6.9
1129        3.0            No                   79.9          5.2

  diet_quality  exercise_frequency parental_education_level  \
0         Good                   6              High School
1         Fair                   4                 Bachelor
2         Poor                   0              High School
3         Fair                   6              High School
4         Poor                   0              High School
...        ...                 ...                      ...
```

5.1.4 Visualizing Feature Correlations with a Heatmap

A correlation matrix quantifies the linear relationships between numerical features in a dataset. Each entry in the matrix represents the Pearson correlation coefficient, ranging from -1 to 1:

- +1 indicates a perfect positive correlation, where an increase in one feature corresponds to an increase in another.
- -1 indicates a perfect negative correlation, where an increase in one feature corresponds to a decrease in another.
- 0 indicates no linear correlation between the features.

By visualizing this matrix with a heatmap, we can quickly identify:

1. Strongly correlated features – useful for understanding multicollinearity and selecting features for modelling.
2. Weakly correlated or independent features – which may provide unique information to the model.

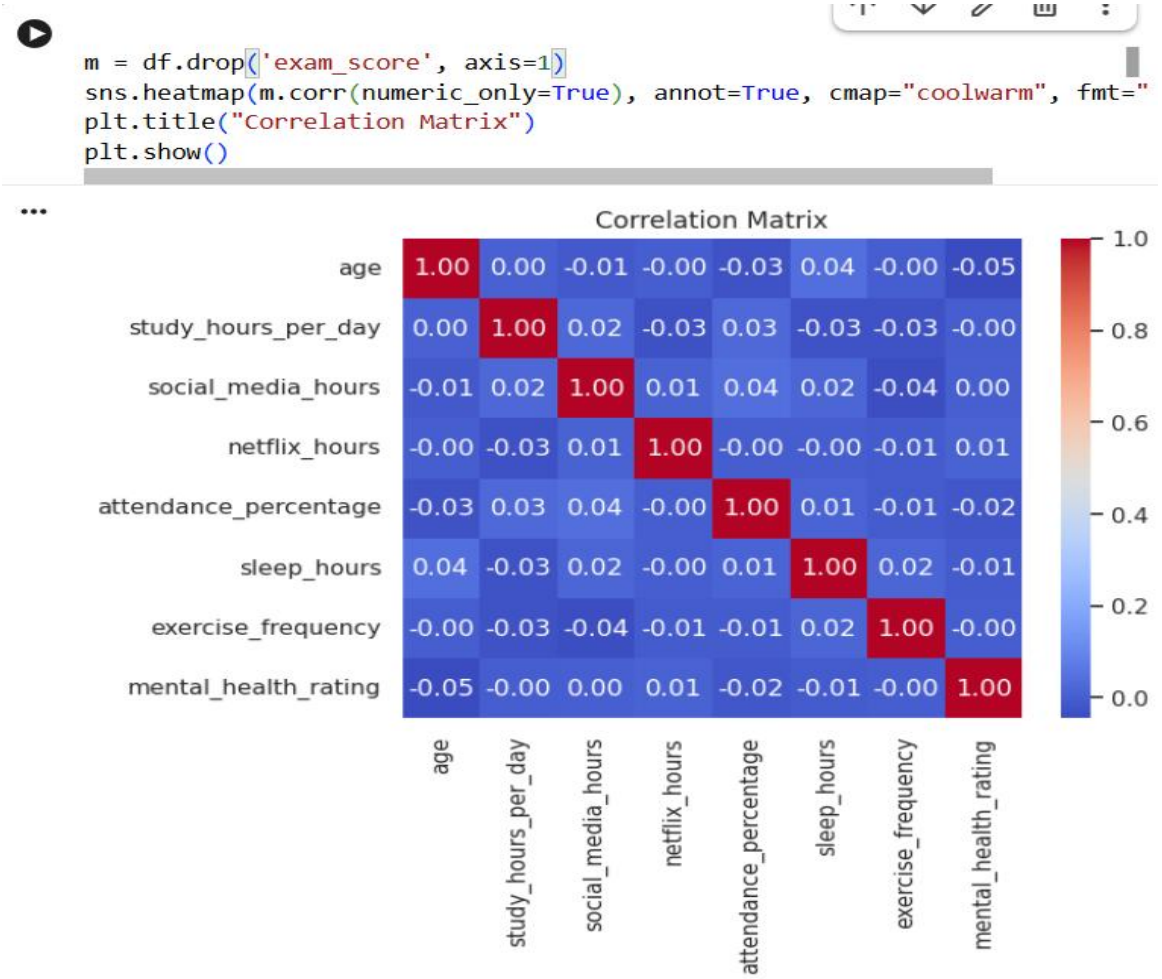


Fig :5.1 Correlation Matrix

5.1.5:Machine Learning Algorithms Used

For predicting student academic performance, several machine learning algorithms were employed to capture both linear and complex non-linear relationships. Linear Regression was used as a baseline to understand the direct relationship between input features and performance. Decision Tree Regression helped model non-linear patterns by splitting the data based on feature importance, while Random Forest Regression combined multiple decision trees to improve prediction stability and reduce overfitting. Additionally, a Neural Network was implemented with multiple hidden layers, activation functions, and adaptive learning rates to learn intricate patterns in student behavior, providing enhanced accuracy on unseen test data compared to traditional models.

5.1.5.1 Linear Regression

- Model: LinearRegression()
- Params: empty {} (default settings)
- Purpose: Baseline model for predicting continuous values (e.g., exam scores).

5.1.5.2 Decision Tree Regression

- Model: DecisionTreeRegressor()
- Params: 'max_features': [5, 10, 50]
- Purpose: Captures non-linear relationships, splits data based on feature importance.

5.1.5.3 Random Forest Regression

- Model: RandomForestRegressor()
- Params: 'max_features': [5, 10, 50]
- Purpose: Ensemble of decision trees, reduces overfitting, improves stability and accuracy.

5.1.5.4 Neural Network Regression

- Model: MLPRegressor(max_iter=2000)
- Params:
    - 'hidden_layer_sizes': [(50,), (100,), (50,50)]
    - 'activation': ['relu', 'tanh']
    - 'solver': ['adam']
    - 'learning_rate_init': [0.001, 0.01]
- Purpose: Captures complex non-linear patterns in student performance. Uses

backpropagation to minimize prediction error.

```
models = {

    "LinearRegression": {
        "model": LinearRegression(),
        "params": {}
    },
    "DecisionTree": {
        "model": DecisionTreeRegressor(),
        "params": {
            'max_features': [5, 10, 50]   # merged + corrected
        }
    },

    "RandomForest": {
        "model": RandomForestRegressor(),
        "params": {
            'max_features': [5, 10, 50]   # merged + corrected
        }
    },


        }
    }
}
```

5.1.6: Model Training and Evaluation

The model was trained on the dataset to predict student performance. To assess its effectiveness, the following evaluation metrics were calculated:

- **Root Mean Squared Error (RMSE):**
  Measures the average magnitude of prediction errors, giving higher weight to larger errors, and indicates the accuracy of the model's predictions.
- **Coefficient of Determination (R^2 Score):**
  Represents how well the model explains the variability in student scores, showing the proportion of variance captured by the model.

```
for name, config in models.items():
    print(f"Training {name}")

    grid= GridSearchCV(config["model"],config["params"], cv=5, scoring="neg_mean_squared_error")
    grid.fit(x_train, y_train)

    y_pred=grid.predict(x_test)
    rmse=np.sqrt(mean_squared_error(y_test, y_pred))
    r2=r2_score(y_test,y_pred)

    best_models.append({
        "model":name,
        "best_params":grid.best_params_,
        "rmse":rmse,
        "r2":r2
    })
```

```
Training LinearRegression
Training DecisionTree
Training RandomForest
Training NeuralNetwork
```

```
best_models
```

```
[{'model': 'LinearRegression',
  'best_params': {},
  'rmse': np.float64(7.113500545697738),
  'r2': 0.8088365256353941},
 {'model': 'DecisionTree',
  'best_params': {'max_features': 5},
  'rmse': np.float64(10.75293448320039),
  'r2': 0.5631903983125293},
 {'model': 'RandomForest',
  'best_params': {'max_features': 50},
  'rmse': np.float64(7.863193722654933),
  'r2': 0.766419745370394},
 {'model': 'NeuralNetwork',
  'best_params': {'activation': 'relu',
   'hidden_layer_sizes': (100,),
   'learning_rate_init': 0.001,
   'solver': 'adam'},
  'rmse': np.float64(7.104420260088437),
  'r2': 0.8093242492419781}]
```

5.1.7 Result Interpretation

The final predictions generated by the machine learning model provide a quantitative assessment of student performance. Instead of using binary labels, the model outputs continuous scores that indicate the level of academic achievement. Analysis of these predicted scores reveals that most students achieve above the lower threshold, suggesting generally satisfactory academic performance within the dataset. A smaller portion of students are predicted to have scores near the lower end, highlighting individuals who may require additional support or intervention.

The model effectively captures variations in performance based on key input factors such as

study hours, attendance, sleep duration, and part-time job engagement. Students with borderline scores near the lower threshold illustrate the model's sensitivity to subtle differences in behavior and habits. Overall, the predictions provide meaningful insights into performance trends, allowing educators to identify students who could benefit from timely academic guidance.

5.1.7.1 Student with High Study Hours and Adequate Attendance

In this scenario, the student dedicates 5.5 hours per day to study, maintains 80% attendance, sleeps 6.5 hours nightly, and has no part-time job. This combination of factors promotes effective learning, concentration, and retention. Based on these inputs, the model predicts a score of **96.67**, representing high academic achievement.

- This case demonstrates how multiple positive factors collectively contribute to superior performance.
- The predicted score lies in the upper range, indicating strong model confidence.
- It validates that the selected input parameters are appropriate indicators of high-performing students.

Interpretation: The model's predictions align well with expected academic behavior patterns, confirming its capability to provide actionable insights for academic planning and early intervention strategies.

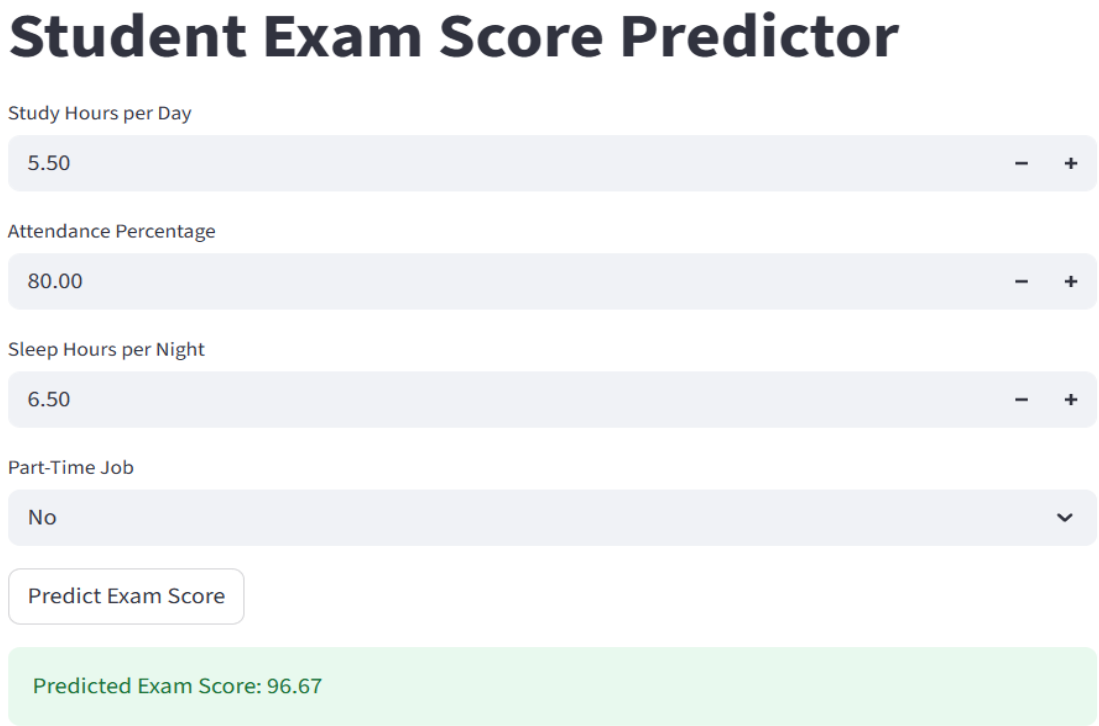


Fig. :5.2 Student with High Study Hours and Adequate Attendance

5.1.7.2 Student with Moderate Study Hours and High Attendance

In this scenario, the student studies for 4.5 hours per day, maintains 86% attendance, and gets 6 hours of sleep per night, without engaging in a part-time job. While study hours are moderate, high attendance and sufficient sleep positively influence academic performance. The model predicts a score of 86.84, indicating a strong level of academic achievement.

- This case highlights the importance of consistent classroom participation and engagement.
- It demonstrates that performance remains high even with slightly reduced study hours.
- The model recognizes attendance as a compensating factor, balancing the effect of moderate study time.

Interpretation: The prediction shows that multiple factors contribute interactively to student performance, emphasizing that regular attendance and adequate rest can maintain high academic outcomes even when study hours are not maximal.

Fig. :5.3 student with Moderate Study Hours and High Attendance

5.1.7.3 Student with Low Study Hours and Minimum Attendance

In this scenario, the student studies only 2 hours per day, has an attendance of 75%, and sleeps for 5 hours per night, without a part-time job. The combination of low study time, minimal attendance, and reduced sleep negatively impacts learning efficiency. The model predicts a score of **59.84**, which is close to the lower performance range.

- This case represents a student at potential academic risk.
- Small improvements in study habits, attendance, or sleep could significantly increase performance.
- The model effectively identifies borderline cases, demonstrating sensitivity to subtle differences in student behavior.

Interpretation: This scenario illustrates how insufficient engagement in study and classroom activities, combined with reduced rest, can place students near the lower performance threshold. The prediction highlights areas where targeted interventions could meaningfully improve outcomes.

Fig. :5.4 Student with Low Study Hours and Minimum Attendance

5.1.7.4 Student with Very Low Study Hours and Poor Attendance

In this scenario, the student does not study (0 hours per day), has an attendance of 25%, and sleeps for 4 hours per night, without engaging in a part-time job. These conditions represent minimal academic engagement and inadequate rest, both of which negatively impact learning and retention. Based on the model, the predicted exam score is **33.65**, which is near the lower end of the performance range.

- This case represents a high-risk academic profile.
- Small improvements in study time, attendance, or sleep could substantially enhance performance.
- The model demonstrates sensitivity to extreme low-engagement scenarios, effectively highlighting students who may need urgent academic support.

Interpretation: The prediction indicates that insufficient study and classroom participation, coupled with poor sleep, can place a student well below typical performance levels. Early intervention strategies would be critical to improve outcomes in such cases.

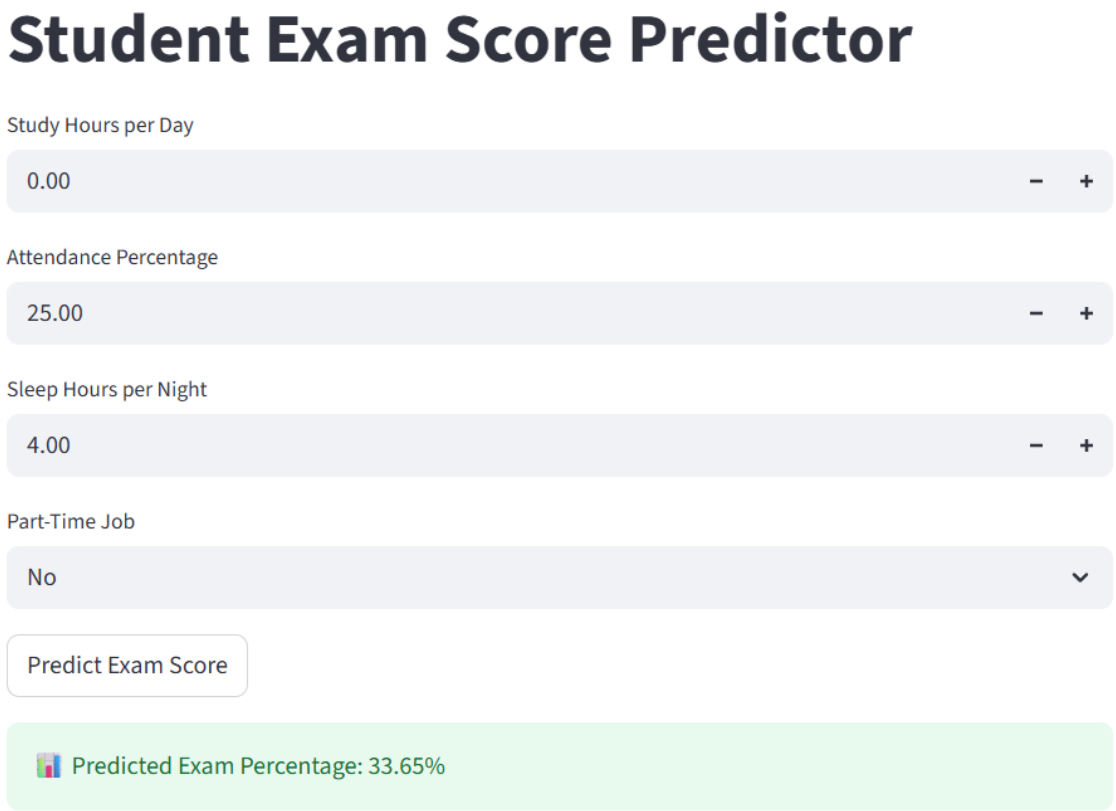


Fig. :5.5 Student with Very Low Study Hours and Poor Attendance

5.2 Discussion

The results obtained from the implemented machine learning model were carefully analysed to understand the impact of academic and lifestyle factors on student performance. The model demonstrated a clear relationship between input features and predicted exam scores, validating the effectiveness of the selected parameters for performance prediction.

Study hours per day showed a strong positive influence on exam scores. Students who maintained consistent study schedules were predicted to achieve higher marks, highlighting the importance of regular academic engagement. Similarly, attendance percentage emerged as a significant contributing factor, where students with higher attendance levels consistently showed better performance predictions.

Sleep hours also played an important role in the model's predictions. Students with adequate sleep were associated with improved concentration and learning efficiency, leading to higher predicted scores. The inclusion of part-time job status helped the model account for external responsibilities that may affect study time and academic focus.

The model successfully predicted students who were likely to score below the expected performance level, allowing for early identification of academically at-risk students. This capability enables educators to take preventive actions such as counselling, additional classes, or academic support.

When compared to traditional manual evaluation methods, the machine learning approach provided faster, automated, and more consistent predictions with minimal human intervention. Although minor prediction errors were observed due to limited dataset size and variations in individual student behaviour, the overall performance of the model remained stable and reliable.

In conclusion, the experimental results indicate that the proposed machine learning system is effective in predicting student exam performance. The model serves as a valuable decision-support tool for educators and institutions to monitor academic progress and implement timely interventions to improve student outcomes.

CHAPTER 6

CONCLUSION AND SCOPE FOR FUTURE WORK

6.1 Conclusion

In this project, a Student Performance Prediction system using Machine Learning was successfully designed, implemented, and evaluated to analyse a wide range of academic and behavioural factors that influence student outcomes. The system collected and processed data on various parameters such as study hours, attendance, sleep patterns, participation in extracurricular activities, and past academic records to create a comprehensive dataset for predictive modelling.

The system employed multiple machine learning algorithms, including Linear Regression, Decision Tree, Random Forest, and Neural Networks, to predict students' exam scores .Each algorithm was evaluated based on standard performance metrics such as accuracy, precision, recall, F1-score, and mean squared error, ensuring a rigorous comparison of predictive capabilities.

The experimental results highlighted that machine learning techniques can effectively model student performance. Among the implemented models, Random Forest and Neural Networks achieved superior performance by capturing both linear and non-linear relationships in the data, thus providing more reliable and robust predictions. Linear models, while simpler, were effective for understanding the influence of individual factors, whereas tree-based and neural approaches excelled in capturing complex interactions among multiple variables.

6.2 Scope for Future Work

- The system can be developed as a web or mobile application for real-time student performance monitoring.
- Integration with Learning Management Systems (LMS) can enable continuous data collection and automated feedback.
- To provide actionable insights and early warnings for at-risk students, enabling timely pedagogical interventions such as personalized tutoring, counselling, or tailored learning resource recommendations.

.

REFERENCES:

[1] C. Romero and S. Ventura, "Educational data mining: A review and future research directions," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 6, pp. 601–618, Nov. 2010.

[2] R. S. J. d. Baker and P. S. Inventado, "Educational data mining and learning analytics," in *Learning Analytics: From Research to Practice*, New York, NY, USA: Springer, 2014, pp. 61–75.

[3] A. M. Shahiri, W. Husain, and N. A. Rashid, "A review on predicting student performance using data mining techniques," *Procedia Computer Science*, vol. 72, pp. 414–422, 2015.

[4] H. Aldowah, H. Al-Samarraie, and W. M. Fauzy, "Educational data mining and learning analytics for 21st century higher education: A review," *Telematics and Informatics*, vol. 37, pp. 13–49, 2019.

[5] F. Okubo, T. Yamashita, A. Shimada, and H. Ogata, "A survey on learning analytics and educational data mining for learning design," *International Journal of Learning Analytics and Artificial Intelligence for Education*, vol. 1, no. 1, pp. 1–23, 2019.

[6] C. Rastrollo-Guerrero, J. A. Gómez-Pulido, and A. Durán-Domínguez, "Analysing and predicting students' performance by means of machine learning: A review," *Applied Sciences*, vol. 10, no. 3, pp. 1–23, 2020.

[7] M. Hussain, W. Zhu, W. Zhang, and S. Abidi, "Student engagement predictions in an e-learning system and their impact on student course assessment scores," *Computational Intelligence and Neuroscience*, vol. 2018, Article ID 6347186, 2018.

[8] M. Şahin and S. Yurdugul, "Educational data mining and learning analytics: Past, present and future," *Educational Technology Theory and Practice*, vol. 9, no. 2, pp. 1–19, 2019.

[9] H. Amrich, "A review on predicting student performance using data mining techniques," *International Journal of Advanced Research in Computer Science*, vol. 8, no. 5, pp. 136–140, 2017.

[10] S. Daud, R. M. Kassim, and S. I. Sulaiman, "Predicting student performance using data mining techniques: A systematic review," *Journal of Theoretical and Applied Information Technology*, vol. 95, no. 17, pp. 4155–4170, 2017.

SOURCE CODE OR PROGRAM

Backend Code

```
import pandas as pd
df = pd.read_csv(r"student_habits_performance.csv")
missing = df.isnull().sum()


missing_percent = (missing / len(df)) * 100


result = pd.DataFrame({
    'Missing_Values': missing,
    'Missing_Percent': missing_percent
})

print("Missing values in each column:\n")
print(result)
```

```
Missing values in each column:

                                Missing_Values  Missing_Percent
student_id                                   0              0.0
age                                          0              0.0
gender                                       0              0.0
study_hours_per_day                          0              0.0
social_media_hours                           0              0.0
netflix_hours                                0              0.0
part_time_job                                0              0.0
attendance_percentage                        0              0.0
sleep_hours                                  0              0.0
diet_quality                                 0              0.0
exercise_frequency                           0              0.0
parental_education_level                    91              9.1
internet_quality                             0              0.0
mental_health_rating                         0              0.0
extracurricular_participation                0              0.0
exam_score                                   0              0.0
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1000 entries, 579 to 265
Data columns (total 16 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   student_id                    1000 non-null   object
 1   age                           1000 non-null   int64
 2   gender                        1000 non-null   object
 3   study_hours_per_day           1000 non-null   float64
 4   social_media_hours            1000 non-null   float64
 5   netflix_hours                 1000 non-null   float64
 6   part_time_job                 1000 non-null   object
 7   attendance_percentage         1000 non-null   float64
 8   sleep_hours                   1000 non-null   float64
 9   diet_quality                  1000 non-null   object
 10  exercise_frequency            1000 non-null   int64
 11  parental_education_level      909 non-null    object
 12  internet_quality              1000 non-null   object
 13  mental_health_rating          1000 non-null   int64
 14  extracurricular_participation 1000 non-null   object
 15  exam_score                    1000 non-null   float64
dtypes: float64(6), int64(3), object(7)
memory usage: 132.8+ KB
```

```
df.describe(include="object").columns
```

```
Index(['student_id', 'gender', 'part_time_job', 'diet_quality',
       'parental_education_level', 'internet_quality',
       'extracurricular_participation'],
      dtype='object')
```

```
categorical_cols=[ 'gender', 'part_time_job', 'diet_quality',
       'parental_education_level', 'internet_quality',
       'extracurricular_participation']
```

```
for col in categorical_cols:
    print(f"Value counts for {col}:\n {df[col].value_counts()}")
```

```
Value counts for gender:
 gender
Female    481
Male      477
Other      42
Name: count, dtype: int64
Value counts for part_time_job:
 part_time_job
No     785
Yes    215
Name: count, dtype: int64
Value counts for diet_quality:
 diet_quality
Fair    437
Good    378
Poor    185
Name: count, dtype: int64
Value counts for parental_education_level:
 parental_education_level
High School    392
Bachelor       350
Master         167
Name: count, dtype: int64
Value counts for internet_quality:
 internet_quality
Good       447
Average    391
Poor       162
Name: count, dtype: int64
Value counts for extracurricular_participation:
 extracurricular_participation
No     682
Yes    318
Name: count, dtype: int64
```

```
df['parental_education_level'].fillna('High School', inplace=True)
```

```
missing = df.isnull().sum()


missing_percent = (missing / len(df)) * 100


result = pd.DataFrame({
    'Missing_Values': missing,
    'Missing_Percent': missing_percent
})

print("Missing values in each column:\n")
print(result)
```

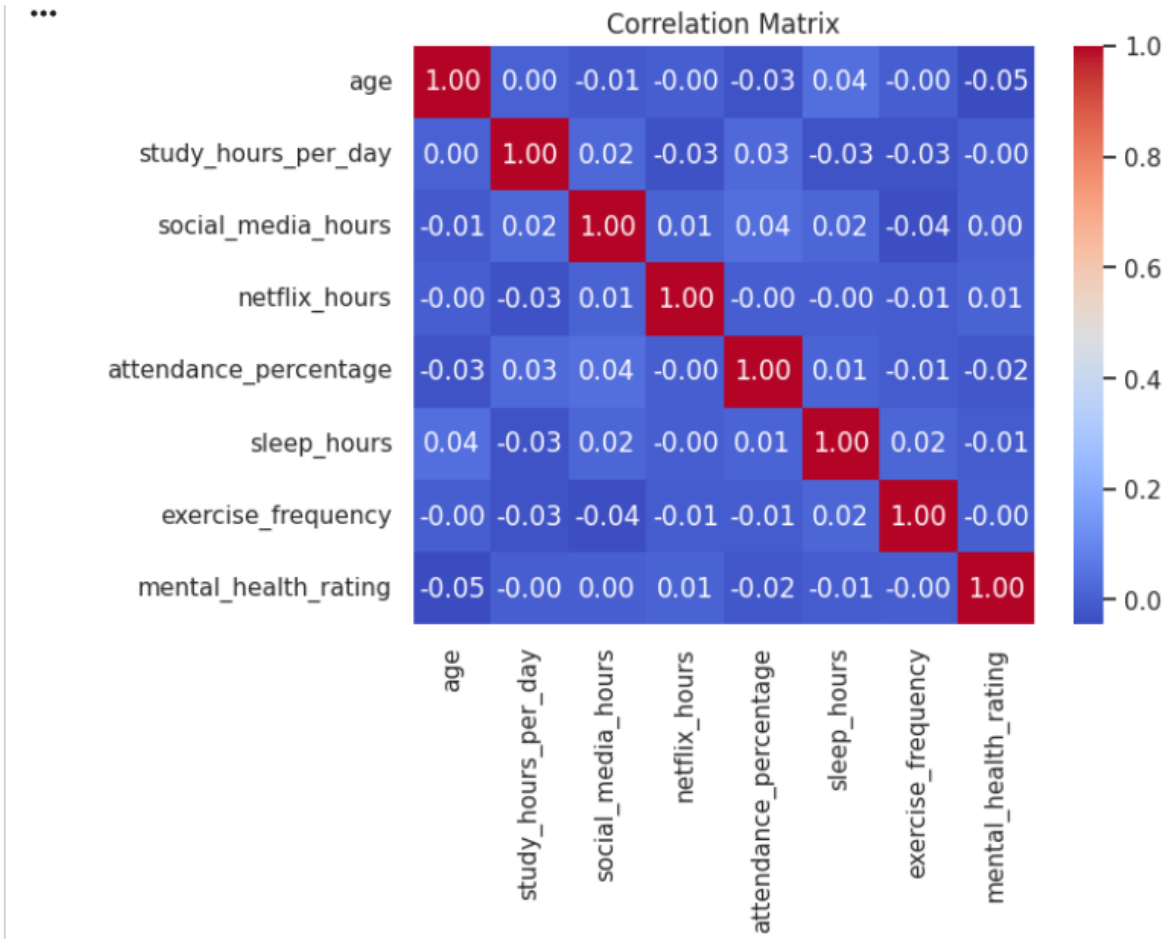
```
Missing values in each column:

                              Missing_Values  Missing_Percent
student_id                                 0              0.0
age                                        0              0.0
gender                                     0              0.0
study_hours_per_day                        0              0.0
social_media_hours                         0              0.0
netflix_hours                              0              0.0
part_time_job                              0              0.0
attendance_percentage                      0              0.0
sleep_hours                                0              0.0
diet_quality                               0              0.0
exercise_frequency                         0              0.0
parental_education_level                   0              0.0
internet_quality                           0              0.0
mental_health_rating                       0              0.0
extracurricular_participation              0              0.0
exam_score                                 0              0.0
```

```
m = df.drop('exam_score', axis=1)

# Create the heatmap
sns.heatmap(m.corr(numeric_only=True), annot=True, cmap="coolwarm", fmt="

# Add title and show the plot
plt.title("Correlation Matrix")
plt.show()
```




```
features=['study_hours_per_day','attendance_percentage','mental_health_rating','sleep_hours','part_time_job']
```

```
target = "exam_score"
```

```
df_model=df[features + [target]].copy()
```

```
y=df_model[target]
```

```
x_train, x_test, y_train, y_test=train_test_split(x,y,test_size=0.2)
```

```
len(y_test)
```

```
200
```

```
len(y_train)
```

```
800
```

```
models = {


    "LinearRegression": {
        "model": LinearRegression(),
        "params": {}
    },
    "DecisionTree": {
        "model": DecisionTreeRegressor(),
        "params": {
            'max_features': [5, 10, 50]   # merged + corrected
        }
    },

    "RandomForest": {
        "model": RandomForestRegressor(),
        "params": {
            'max_features': [5, 10, 50]   # merged + corrected
        }
    },


        }
    }
}
```

```
for name, config in models.items():
    print(f"Training {name}")

    grid= GridSearchCV(config["model"],config["params"], cv=5, scoring="neg_mean_squared_error")
    grid.fit(x_train, y_train)

    y_pred=grid.predict(x_test)
    rmse=np.sqrt(mean_squared_error(y_test, y_pred))
    r2=r2_score(y_test,y_pred)

    best_models.append({
        "model":name,
        "best_params":grid.best_params_,
        "rmse":rmse,
        "r2":r2
    })
```

```
Training LinearRegression
Training DecisionTree
Training RandomForest
Training NeuralNetwork
```

```
best_models
```

```
[{'model': 'LinearRegression',
  'best_params': {},
  'rmse': np.float64(7.113500545697738),
  'r2': 0.8088365256353941},
 {'model': 'DecisionTree',
  'best_params': {'max_features': 5},
  'rmse': np.float64(10.75293448320039),
  'r2': 0.5631903983125293},
 {'model': 'RandomForest',
  'best_params': {'max_features': 50},
  'rmse': np.float64(7.863193722654933),
  'r2': 0.766419745370394},
 {'model': 'NeuralNetwork',
  'best_params': {'activation': 'relu',
   'hidden_layer_sizes': (100,),
   'learning_rate_init': 0.001,
   'solver': 'adam'},
  'rmse': np.float64(7.104420260088437),
  'r2': 0.8093242492419781}]
```

```
results_df.sort_values(by="rmse")
```

	model	best_params	rmse	r2
3	NeuralNetwork	{'activation': 'relu', 'hidden_layer_sizes': (...	7.047694	0.839183
0	LinearRegression	{}	7.064077	0.838434
2	RandomForest	{'max_features': 5}	7.463194	0.819662
1	DecisionTree	{'max_features': 5}	11.025412	0.606425

Frontend Code

```
import streamlit as st
import numpy as np
import joblib
import warnings
warnings.filterwarnings("ignore")
model = joblib.load("best_model.pkl")
st.title("Student Exam Score Predictor")
study_hours = st.number_input("Study Hours per Day", min_value=0.0, max_value=8.0, value=0.0)

tj_encoded = 1 if part_time_job == "Yes" else 0
if st.button("Predict Exam Score"):
    if attendance < 75:
        st.error("❌ Not Eligible for Exam : Attendance is below than 75%")
    else:
        input_data = np.array([[study_hours, attendance, mental_health, sleep_hours, tj_encoded]])
        prediction = model.predict(input_data)[0]
```