

Student Performance Predictor

Android Theory Assignment-3 Report

Student Name: Jazib Asad

Registration Number: 232201056

Class: BSCS-5B

Semester: Fall-2025

Course: Android App Development

University/Institute: Institute of Space Technology, KICSIT

Submitted To: Mr. Uzair Hassan

Submission Date: January 7, 2026

Contents

1 Abstract	2
2 Problem Statement	2
3 Objectives	2
4 System Architecture	2
4.1 Technology Stack	2
4.2 Machine Learning Pipeline	2
5 Implementation Details	3
5.1 Input Parameters	3
5.2 Core Components	3
6 User Interface (UI) Design	4
7 Results and Discussion	4
8 Conclusion	4
9 Future Scope	4

1 Abstract

The **Student Performance Predictor** is an offline Android application designed to estimate a student's Cumulative Grade Point Average (CGPA) based on their academic history and daily habits. By leveraging Machine Learning (TensorFlow Lite) directly on the mobile device, the application provides instant, data-driven predictions without requiring an internet connection. The system also maintains a local history of predictions for user reference.

2 Problem Statement

Students often struggle to gauge how their daily habits and background factors influence their final academic performance. Traditional methods of estimation are often subjective or manual. There is a need for a tool that can scientifically analyze key performance indicators (KPIs) and provide a realistic prediction of academic outcomes to help students plan better.

3 Objectives

- To develop an Android application that uses Machine Learning for regression tasks.
- To implement an offline inference engine using TensorFlow Lite.
- To identify key factors affecting student performance (e.g., study hours, attendance).
- To provide a user-friendly interface using Material Design principles.
- To persist user data locally using a structured database (Room).

4 System Architecture

4.1 Technology Stack

- **Platform:** Android (Min SDK 24, Target SDK 36)
- **Language:** Java
- **Machine Learning:** TensorFlow Lite (TFLite)
- **Database:** Room Persistence Library (SQLite abstraction)
- **UI Framework:** XML Layouts with Material Design 3 Components

4.2 Machine Learning Pipeline

1. **Data Collection:** Dataset (`Students_Performance_data_set.xlsx`) containing student attributes and CGPA.
2. **Preprocessing:**

- Data cleaning (handling ranges, missing values).
- Normalization using `StandardScaler` (Mean and Standard Deviation extraction).

3. Model Training:

- Algorithm: **Linear Regression** (Neural Network with Dense layers).
- Framework: TensorFlow / Keras (Python).

4. Deployment:

- Model converted to `.tflite` format.
- Normalization parameters (Mean/Std Dev) hardcoded in Android app's `PredictionHelper` class.

5 Implementation Details

5.1 Input Parameters

The application requires the following 5 inputs from the user:

1. **Daily Study Hours:** Number of hours spent studying per day.
2. **Attendance (%):** Percentage of classes attended.
3. **Previous SGPA:** Student's Score Grade Point Average from the previous semester.
4. **Credits Completed:** Total academic credits earned so far.
5. **Monthly Family Income:** Economic background indicator.

5.2 Core Components

- **SplashActivity:** Displays the application branding (Graduation Cap Logo) while the app initializes.
- **MainActivity:** Primary interface containing input fields (`TextInputLayout`) and the "Predict" button.
- **PredictionHelper.java:**
 - Loads the `student_performance_model.tflite` file from assets.
 - Normalizes raw user inputs using pre-calculated Mean and Std Dev.
 - Runs inference and returns predicted CGPA.
- **Database Layer (`com.example.assignment3.database`):**
 - `StudentResult`: Entity class for a single prediction record.
 - `StudentDao`: Data Access Object for inserting and querying records.
 - `AppDatabase`: Room database instance.

6 User Interface (UI) Design

The application follows modern Material Design guidelines:

- **Color Palette:** Soft purple theme (#6200EE) with white backgrounds (#F5F7FA) for a clean, professional look.
- **Input Validation:** Ensures all fields are filled with valid numbers before processing.
- **Feedback:** Displays the result in a prominent CardView upon successful prediction.

7 Results and Discussion

The integrated TensorFlow Lite model successfully performs linear regression on the device.

- **Latency:** Predictions are near-instantaneous (< 100ms) as no network call is required.
- **Accuracy:** Normalization ensures inputs of varying scales (small GPA vs. large Income) are handled correctly, yielding mathematically sound predictions.

8 Conclusion

The Student Performance Predictor demonstrates the integration of Machine Learning into mobile application development. It provides an offline, utility-based solution that estimates academic performance accurately. Local storage enhances user experience by retaining past predictions.

9 Future Scope

- **Graphing:** Visualizing progress over time using charts.
- **Cloud Sync:** Optional backup of history to a cloud server.
- **Enhanced Model:** Incorporating more features like "Extracurricular Activities" or "Sleep Hours" if available in future datasets.