Documentation of the project:
Comprehensive Report: **Student Success Prediction**

1. Project Overview

This project aims to predict student success in an online education platform by analyzing data related to student profiles, course engagement, and historical performance. The data is synthetically generated using Python scripts, and a machine learning model (Random Forest Classifier) is used to predict whether students are likely to complete their courses or be at risk of dropping out.

2. Synthetic Data Generation

The synthetic data used in this project was generated using Python scripts, and the output was saved into CSV files. The following datasets were generated:

1. Student Profile Data: Contains demographic and academic information such as age, gender, major, year of study, and region.

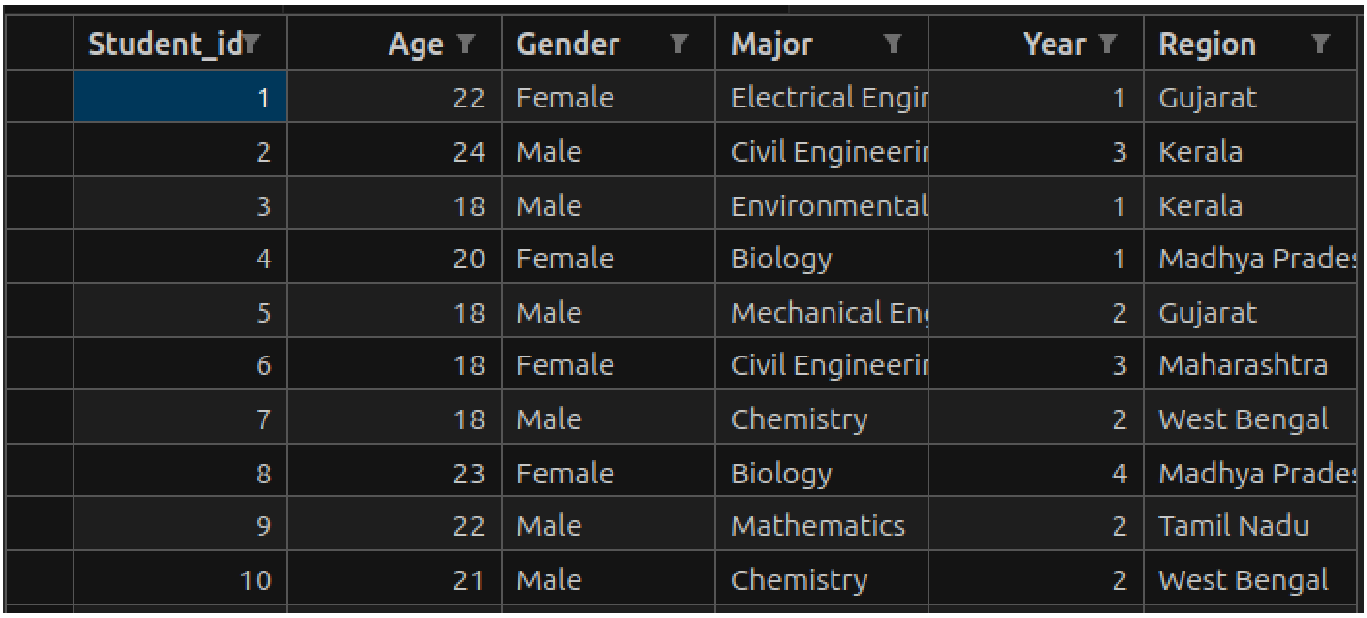
2. Course Engagement Data: Includes metrics related to student activity on the platform, such as logins per week, videos watched, time spent on the platform, and quiz scores.

3. Historical Performance Data: Tracks the number of courses started, courses completed, and average scores across completed courses.

4. Each of these datasets was generated using custom scripts and saved into CSV files, which were later combined for analysis.

CSV Structure Example:

1-Student Profile Data (synthetic_student_data.csv):
Student_id, age,gender, major,year,region
Preview:

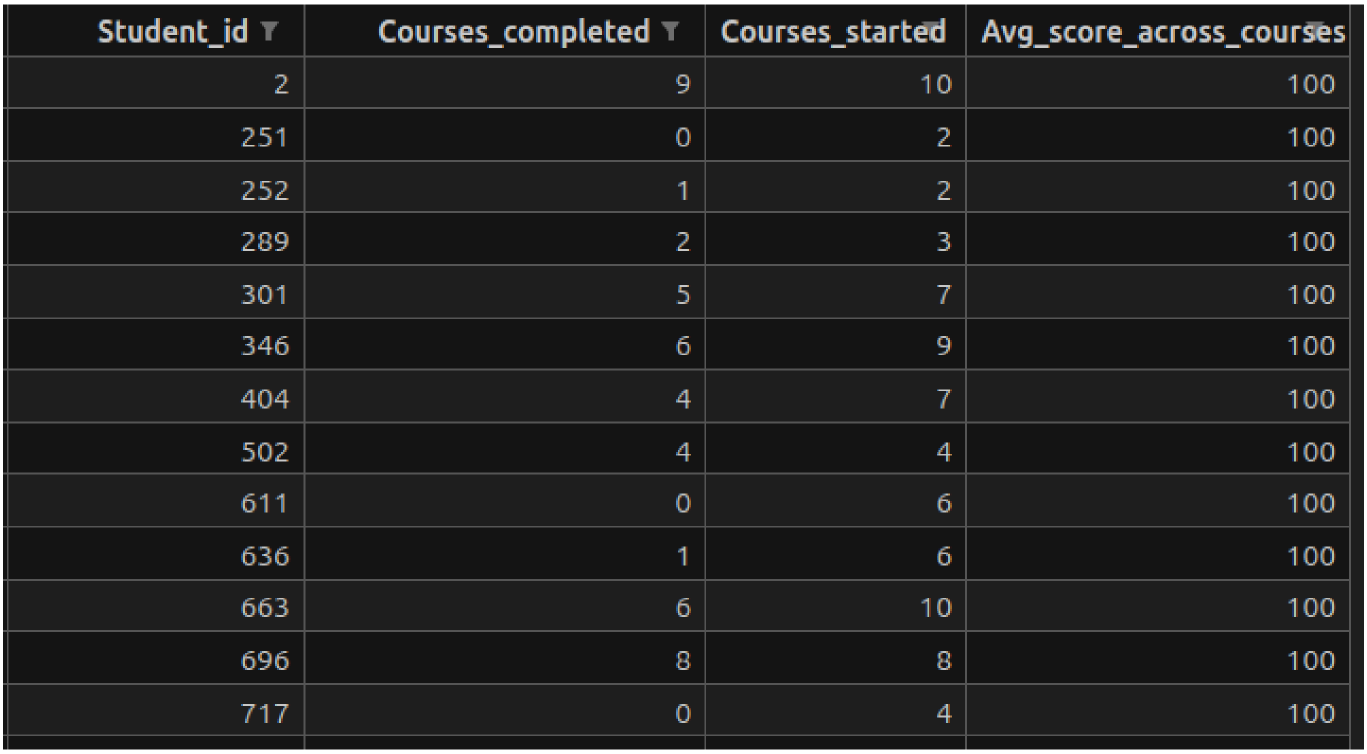


Student_id	Age	Gender	Major	Year	Region
1	22	Female	Electrical Engir	1	Gujarat
2	24	Male	Civil Engineeri	3	Kerala
3	18	Male	Environmental	1	Kerala
4	20	Female	Biology	1	Madhya Prade
5	18	Male	Mechanical En	2	Gujarat
6	18	Female	Civil Engineeri	3	Maharashtra
7	18	Male	Chemistry	2	West Bengal
8	23	Female	Biology	4	Madhya Prade
9	22	Male	Mathematics	2	Tamil Nadu
10	21	Male	Chemistry	2	West Bengal

2-Course Engagement Data (synthetic_course_engagement_data.csv):

- student_id,logins_per_week,videos_watched,time_spent_on_platform,avg_quiz_score

Preview:

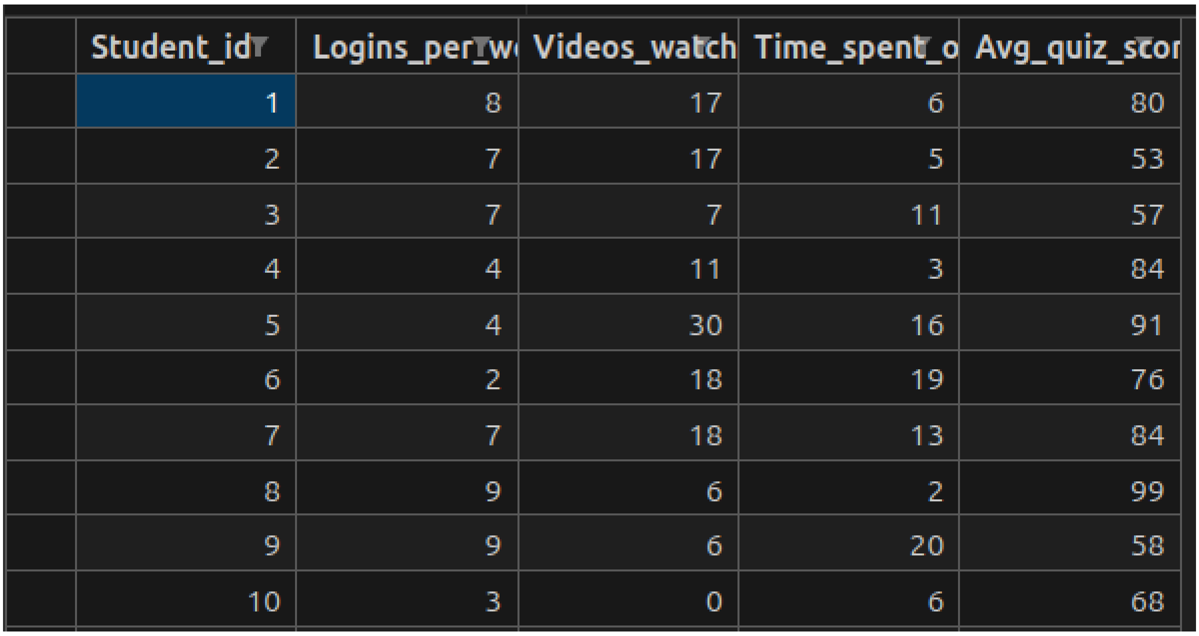


Student_id	Courses_completed	Courses_started	Avg_score_across_courses
2	9	10	100
251	0	2	100
252	1	2	100
289	2	3	100
301	5	7	100
346	6	9	100
404	4	7	100
502	4	4	100
611	0	6	100
636	1	6	100
663	6	10	100
696	8	8	100
717	0	4	100

3-Historical Performance Data (synthetic_historical_data.csv):

- Student_id,courses_started,courses_completed,avg_score_across_courses

Preview:



Student_id	Logins_per_w	Videos_watch	Time_spent_o	Avg_quiz_scor
1	8	17	6	80
2	7	17	5	53
3	7	7	11	57
4	4	11	3	84
5	4	30	16	91
6	2	18	19	76
7	7	18	13	84
8	9	6	2	99
9	9	6	20	58
10	3	0	6	68

Link to access the CSV file:

- https://github.com/Aryamantiwari17/Student_Success_Prediction-/blob/main/data%20science%20project/synthetic_course_engagement_data.csv

- https://github.com/Aryamantiwari17/Student_Success_Prediction-/blob/main/data%20science%20project/synthetic_historical_data.csv

- https://github.com/Aryamantiwari17/Student_Success_Prediction-/blob/main/data%20science%20project/synthetic_student_data.csv

3. Class and Functionality Breakdown
Link of Main Code:
https://github.com/Aryamantiwari17/Student_Success_Prediction-/blob/main/data%20science%20project/main.py

The project is organized into different classes to handle the data processing, model training, evaluation, and visualization tasks. Each class has its specific role, making the code modular and easier to maintain.

Class: DataProcessor
This class is responsible for loading, combining, and preprocessing the data. It takes the three synthetic CSV files and merges them into a single dataset.

Functions:

1. load_data(): Loads the synthetic CSV files into DataFrames.

2. combine_data(): Combines the three datasets on student_id to form a single dataset for analysis.

3. preprocess_data(): Preprocesses categorical data by encoding features like gender, major, and region. It also generates the target variable completed based on the ratio of courses_completed to courses_started.

Class: ModelTrainer
This class handles training the machine learning model and evaluating its performance.

Functions:

1. train_model(): Trains a Random Forest Classifier using the preprocessed data.

2. evaluate_model(): This function evaluates the model on the test data using metrics such as accuracy, precision, recall, and F1 score.

3. get_feature_importance(): Extracts the importance of each feature based on the trained model.

Class: Visualizer
This class is used to visualize the model's performance and feature importance.

Functions:

1. plot_feature_importance(): Plots a bar graph to show the relative importance of each feature.

2. plot_confusion_matrix(): Displays a confusion matrix showing the true and false predictions.

3. plot_evaluation_metrics(): Plots a bar graph showing the accuracy, precision, recall, and F1 score of the model.

Class: StudentSuccessPredictor
This is the main class that ties everything together. It manages the overall workflow, including running the analysis and making predictions for new student data.

Functions:

1. **run_analysis()**: Loads the data, trains the model, evaluates performance, and visualizes the results.

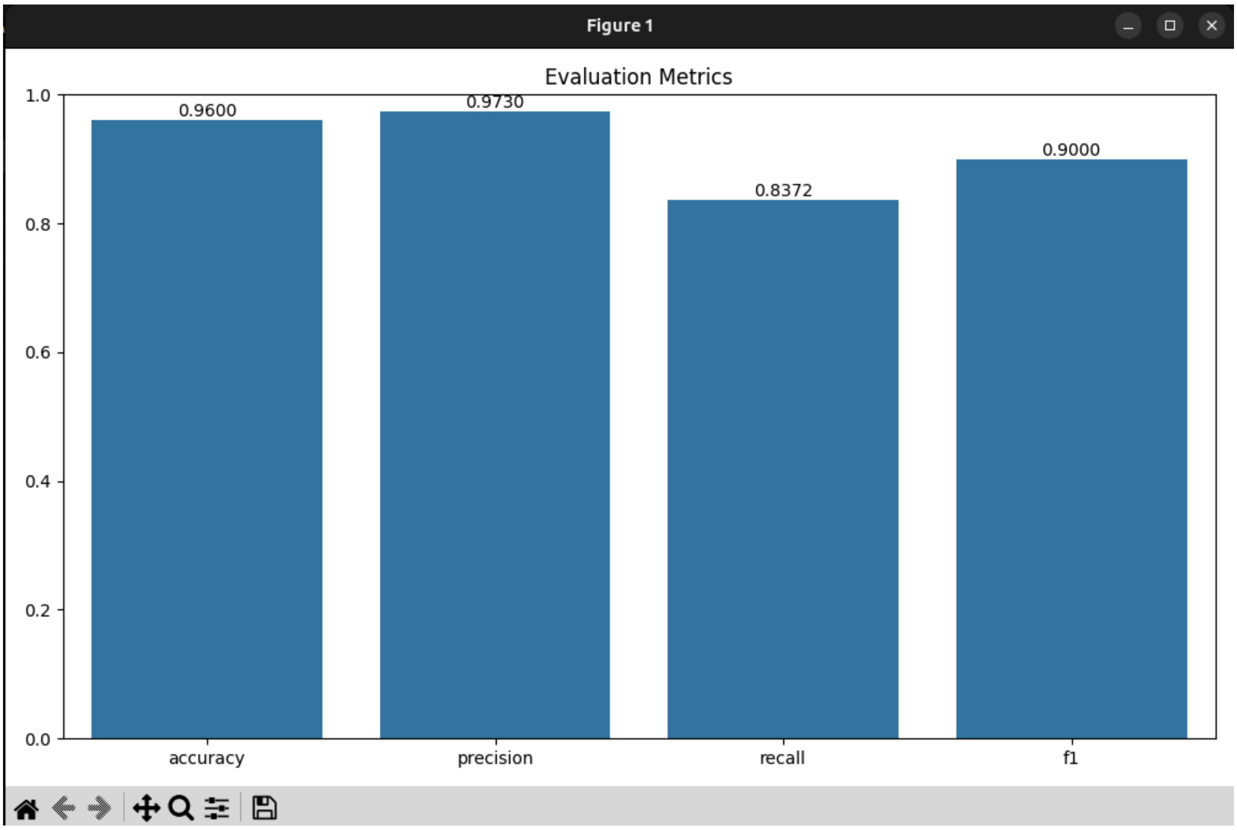
2. **predict_student_success()**: Predicts whether a new student is likely to complete their courses or drop out based on input data.

4. Analysis and Evaluation

Once the data was loaded and combined, the Random Forest Classifier was trained on the synthetic dataset. The evaluation metrics used include:

- Accuracy: 96.00%
- Precision: 97.30%
- Recall: 83.72%
- F1 Score: 90.00%

Preview:



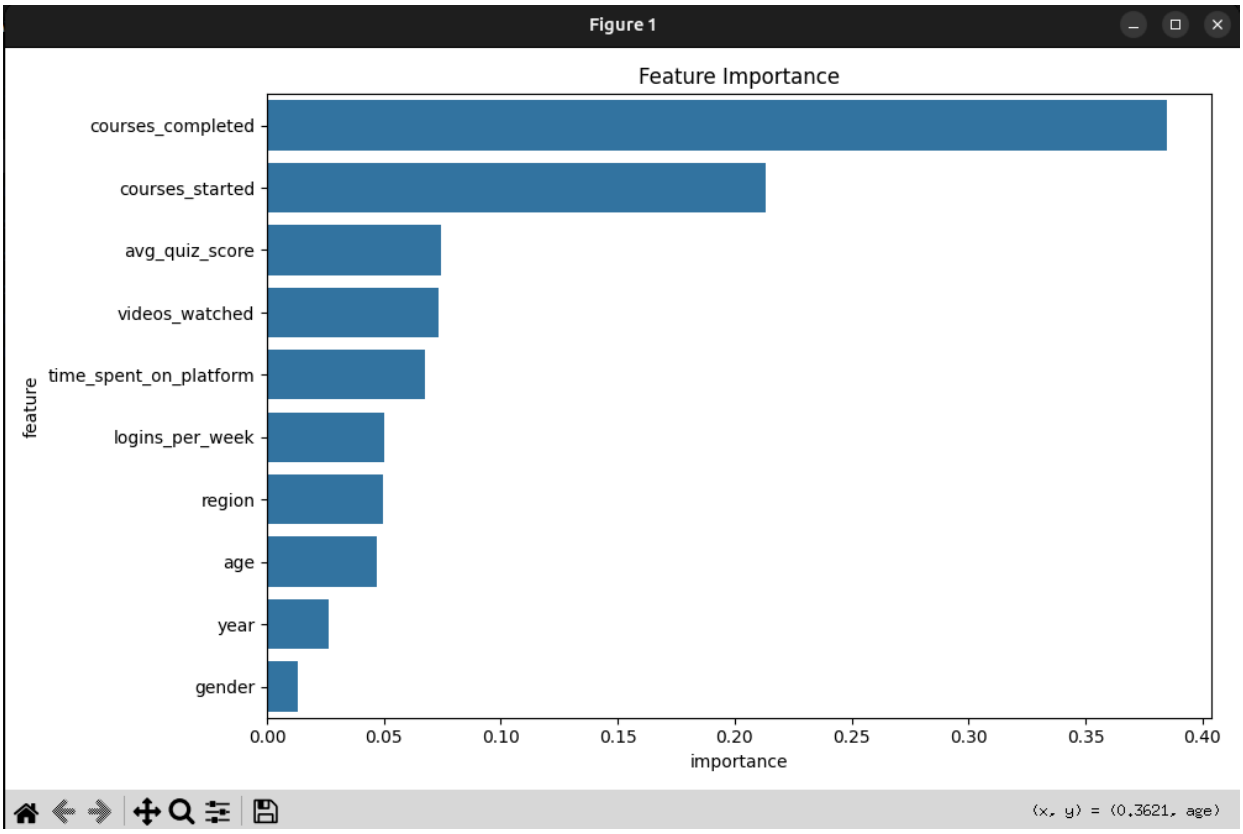
These results show that the model is highly effective at predicting student success, with a high level of precision and overall accuracy.

Feature Importance

The model identified the following features as the most important for predicting student success:

- Courses Completed: 38.46%
- Courses Started: 21.33%
- Average Quiz Score: 7.46%
- Videos Watched: 7.34%
- Time Spent on Platform: 6.75%

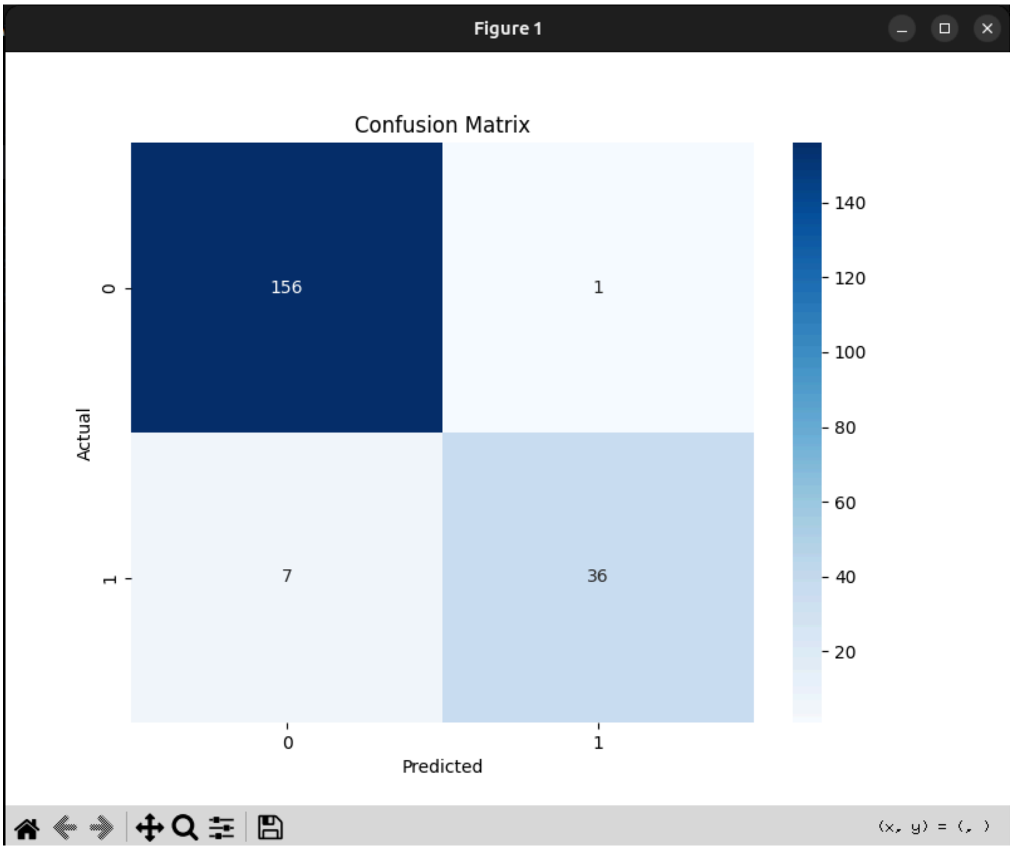
Preview:



Confusion Matrix

The confusion matrix highlights the model's performance in terms of true positives, true negatives, false positives, and false negatives. This matrix helps us understand the types of prediction errors the model makes, such as predicting students as likely to succeed when they are at risk.

Preview:



5. Prediction Output Example

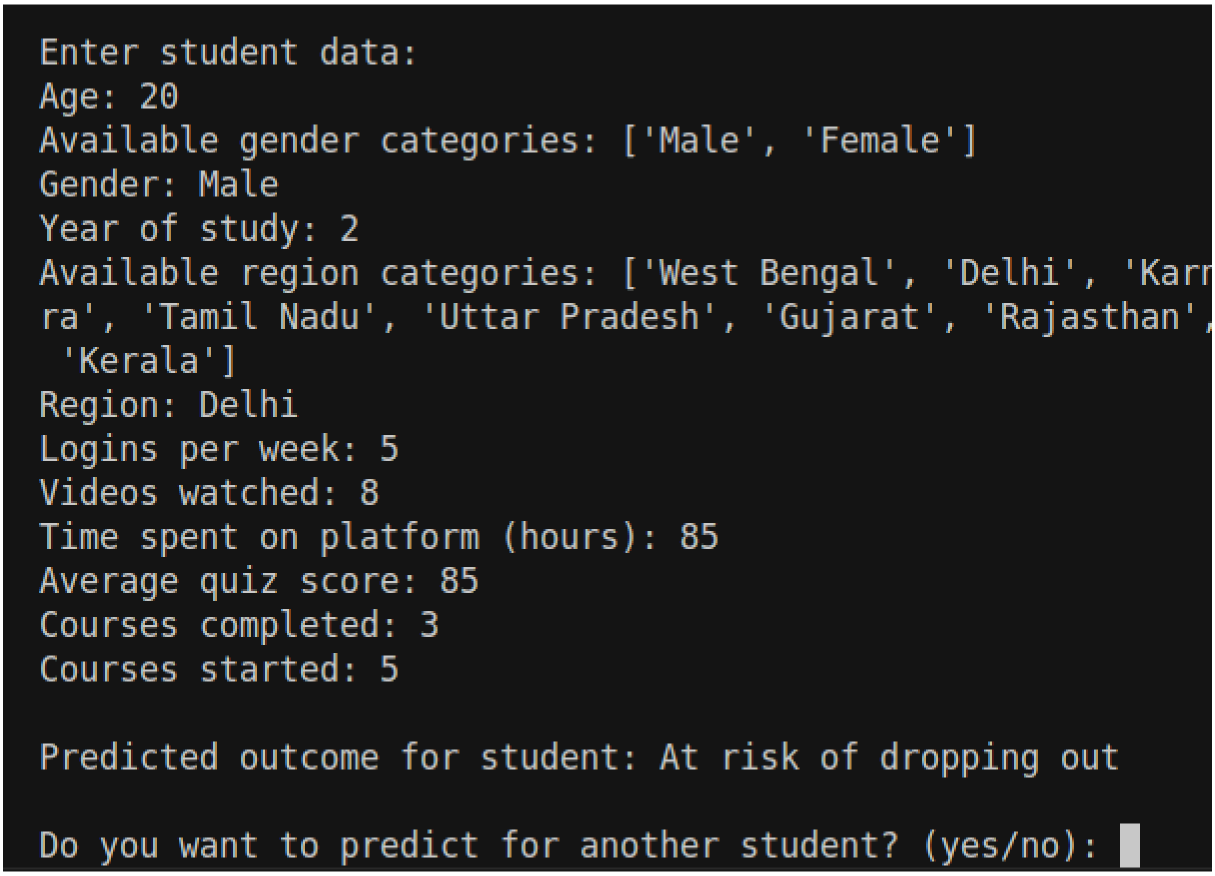
An example was tested using new student input data. The system takes in demographic and engagement information for a student and predicts their outcome:

Input Data:

- Age: 20
- Gender: Male
- Year of study: 2
- Region: Delhi
- Logins per week: 5
- Videos watched: 12
- Time spent on platform: 8 hours
- Average quiz score: 85
- Courses completed: 3
- Courses started: 5

Model Prediction: The model predicted that this student was at risk of dropping out, which could prompt intervention by academic advisors.

Preview:



```
Enter student data:
Age: 20
Available gender categories: ['Male', 'Female']
Gender: Male
Year of study: 2
Available region categories: ['West Bengal', 'Delhi', 'Karn
ra', 'Tamil Nadu', 'Uttar Pradesh', 'Gujarat', 'Rajasthan',
 'Kerala']
Region: Delhi
Logins per week: 5
Videos watched: 8
Time spent on platform (hours): 85
Average quiz score: 85
Courses completed: 3
Courses started: 5

Predicted outcome for student: At risk of dropping out

Do you want to predict for another student? (yes/no):
```

6. Conclusion

- This project successfully demonstrates the use of synthetic data to build a predictive model for student success. By using demographic information, engagement data, and historical performance, the model effectively identifies students who may need additional support to complete their courses.

- This predictive capability can be invaluable for educational institutions looking to improve student retention and success rates.

- Classes in this project ensure modularity and ease of scaling, allowing for future enhancements or changes to specific components, such as data preprocessing or model selection.