International Journal of Scientific Research & Engineering Trends
Volume 12, Issue 2, Mar-Apr-2026, ISSN (Online): 2395-566X

Student Performance Indicator: An End-to-End Machine Learning Pipeline for Predicting Academic Outcomes

Smit Sudani
Dept. of Information Technology
Parul Institute of Engineering and Technology
Parul University, Vadodara, Gujarat, India

Smitsudani555@gmail.com

Abstract: -With all the amount of data that is now available about the students in a school environment, there is no way one could analyze such data manually. The Student Performance Predictor is a web application I designed to help determine the final score that a particular student will get from mathematics class, basing on his demographics and background. The whole machine learning pipeline was implemented by me using the Python language. After experimenting with various models in Jupyter Notebooks and having my kernel crash quite a few times, I managed to find the most accurate one – Random Forest Regressor with an 80% accuracy rate. Next, I embedded this algorithm in my application, which uses the Flask server. User only needs to input some values in three fields to get the prediction instantly.

Keywords- The Student Performance Predictor is a web-based machine learning application that analyzes student data to predict final math scores. It uses Python with a Random Forest Regressor model achieving around 80% accuracy. The system is deployed using a Flask server, allowing users to input basic student details and receive instant predictions. It highlights data-driven analysis, automation, and efficient decision-making in education.

I. INTRODUCTION

Data is everywhere in today's schools. But having data, such as a spreadsheet filled with numbers, doesn't mean that you have information. Most teachers fail to notice subtle signs that would tell them that their students are doing poorly in class. It is crucial to be able to predict performance because it will help us to make decisions about how to intervene and improve the situation. This project was undertaken because we needed some tool that would give us advance warning to let us solve the issues before they arise.

In this report, the development of the Student Performance Indicator through the use of the Python programming language is discussed. Instead of developing a simple program that works in the command prompt, I decided to develop a more advanced system wherein the end-user only interacts with a webpage, which runs the computations on the backend through the power of the algorithm. All of the necessary information is provided in three groups: Student Demographics, Background & Preparation, and Current Academic Scores.

II. LITERATURE REVIEW

Predicting the grades of students has been extensively explored by EDM research community in the past decades. The previous literature has focused on using relatively simple approaches like applying the tools of statistics or regression analysis in order to find some kind of relationship between certain student characteristics and his results on tests. The disadvantage of this technique lies in the fact that the model does not have the ability to capture any type of non-linear relationships which might exist between the variables involved. For example, what would be the effect of the presence of a regular dining time on the student's results in math classes when taking test preparation classes? A simple linear regression could hardly detect it, but nowadays Random Forests proved themselves to be much more effective in this field.

Another limitation which I observed is related to the lack of motivation to deploy and make the developed model workable for real-world applications. Although there are always a number of new models which can effectively address some specific problems proposed, only very little efforts were made towards deploying the model in real-life settings.

III. MATERIALS AND METHODS

This task consisted of two main stages - investigation and modeling on one hand, and development of the website on the other.

A. Tech Stack and Data Sources

Data preprocessing and EDA: data is being processed via Pandas, NumPy, Scikit-Learn, Matplotlib libraries from the Jupyter Notebook environment;

- Back-end technology stack: Python + Flask;
- Front-end technology stack: basic HTML layout with Bootstrap;
- Description of data: data includes demographic data of students, their parent education level, the type of lunch, and previously obtained test scores.

B. Website Creation Process Outline
Preprocessing: dataset is loaded as CSV file; after that, additional preprocessing is done by dividing it into training and test datasets to prevent a potential problem of model memorizing the answers.

Data Transformation: ML algorithms do not work with textual variables, hence, data required some transformations. Specifically, categorical text columns were encoded one hot and numerical ones were standardized. Some Setting With Copy warnings occurred during this step with Pandas.

Model Training: several algorithms were tried in Jupyter Notebook (Linear Regression, Decision Trees, CatBoost, Random Forest); R^2 values and mean absolute errors were calculated to choose models to use.

Pipeline Construction: Since the Random Forest algorithm had proved its superiority, I stored it as a .pkl file. Later, I developed a predict_pipeline.py file in Python. It takes input from the web interface, and after that, scales the dataset exactly as was done while training.

IV. SYSTEM DESIGN AND ARCHITECTURE

To ensure that there is modularity in our code for maintenance purposes, we shall use an appropriate architectural flow when designing the system.

A. User Input Categorization
In order to ensure that the user interface remains simple and logical, we classify our user inputs in three ways:
Student Demographics: This contains information about the student's gender as well as race or ethnicity belonging to any of the five groups A through E.

Background & Preparation: This data contains information relating to the Parental Level of Education, whether or not the student takes a type of Lunch, and whether he or she has taken Test Preparation Courses.

Current Academic Scores: This relates to the grades that the student gets in Reading and Writing subjects, out of 100 marks.

B. The Prediction Flow
When the user clicks the 'Predict' button, the user input form sends a POST request to the flask server. The form data is captured at the Flask server side by our app.py code, converted to pandas dataframe and sent to the PredictPipeline class along with the preprocessor.pkl and model.pkl files.

V. RESULTS AND DISCUSSION

It was not only by luck that I chose Random Forest. In fact, I tried various models in order to see how much the difference can be.

A. Model Performance As far as linear regression gave me a quite good start, the other models definitely won it. After a while, after some tweaking, the performance of Random Forest Regressor came up to a very high 80%. Why is this important? My model has proven to be quite good at identifying relationships among background factors and academic achievement.

B. Discussion The fact that the accuracy is 80% means that there are a lot of impacts from non-academic factors such as diet or parents' educational background on mathematics results. Also, the fact that both of these factors combined with the student's reading and writing skills can determine their mathematical ability.

In addition, because all my data sets are stored separately within distinct modules, loading, cleaning, and training the model would be much simpler in the future. For instance, if I acquire new data next month, I only need to run the training module without making any changes to other parts, including the web server code.

VI. CONCLUSION AND FUTURE WORK

As shown by the Student Performance Indicator, there is indeed worth in transforming school data into valuable information. By merging the random forest algorithm of machine learning with the Flask framework, an amazing prediction accuracy score of 80% was achieved. Moreover, an app was developed to allow people without programming knowledge to use the app to make predictions.

To enhance the app in future, one may consider enlarging the dataset by incorporating other variables such as the students' attendance and participation in other school activities. This will assist in increasing the accuracy of predictions made by the model. Also, it would be awesome if the application can be hosted in the cloud.

Acknowledgment

This is the place where I wish to thank my university guides and professors who have guided me in creating this project. I would also like to thank the open-source community for making available the open source libraries used for this project, such as Scikit-Learn, Pandas, and Flask.

REFERENCES

1. F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825-2830, 2011.
2. A. Pal, "Web application development using Flask," International Journal of Computer Applications, 2020.
3. C. Romero and S. Ventura, "Educational data mining: A review of the state of the art," IEEE Transactions on Systems, Man, and Cybernetics, vol. 40, no. 6, pp. 601-618, 2010.