EduInsight: Report Big Data Technologies Team 8

Samuele Carrubba[1], Alberto Scuderi[2], Julius Schmidt[3]

[1]samuele.carrubba@studenti.unitn.it

[2]alberto.scuderi@studenti.unitn.it

[3]juliusheiko.schmidt@studenti.unitn.it

GitHub: https://github.com/Schmiddl99/BDT24-Team8-edu_insight

***Abstract*— Personalized education refers to a set of systems and methods aimed at offering insights based on someone's educational needs. This project presents EduInsight, a personalized educational platform which predicts a student's university success based on his performances and a questionnaire about his social and familiar circumstances. By analyzing data from multiple sources and platforms and through both machine learning and data analytics, the application generates a concise report on a student's historical records, comparing his performance with that of his peers.**

Keywords *— Student Performance Prediction, Learning Analytics, Data-Driven Education*

i. Introduction

EduInsight is a platform designed to offer personalized insights to a student regarding his university career. Through a data-driven approach and after a brief form, a student can receive key performance indicators as well as a prediction on his ongoing studies, whether he will pass his study program or if he is likely to fail. Different technologies have been used with the idea in mind to experiment with them, obtaining significative overall results.

ii. System Model

a. System architecture

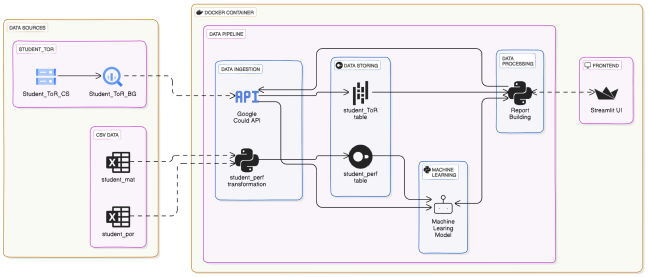


Figure 1: Architecture of EduInsight

The current architecture is made of different parts, as can be seen in Figure 1. The data sources consist of one side of two csv files containing information on the questionnaire answers of students regarding their social and familiar circumstances and their academic performances respectively. These datasets are used to train the machine learning models. The student records are saved in the *Student_ToR* in Google Cloud Storage and Google BigQuery to mimic a real database which we connect to via the Google BigQuery API and a service account. For the actual data pipeline, the Dask DataFrame Python library has been used for data transformation, DuckDB to store the transformed or requested data, and Scikit-Learn to build and evaluate the machine-learning models. Finally, streamlit allowed the realization of the front-end to interact with the user.

Student_perf transformation transforms the CSV files into one cohesive file that uses machine learning-friendly values (integer, float, and boolean). Afterwards, the transformed and concatenated values are stored in a DuckDB relational database.

The *machine learning model* tries to predict if the student will successfully end his studies or if he is on a path to failure. The machine learning model uses different classification algorithms (logistic regression, k-nearest neighbors, multi-layer-perceptron, naive bayes, and support vector machine) to calculate and average the result based on the user input as well as information retained in the student records table (*student_ToR*).

The student records information is obtained through different API calls. Measures like the average grade and the number of lectures being absent are calculated directly through the API calls while the rest of the data is stored in a DataFrame. We opted for a DataFrame for easy in-memory storage and easy data transformation later on.

The *Record Building* retrieves the user's information through an input form and passes it onto the API and the machine learning model. The data from the student records is then used to visualize the comparison between the student and his peers.

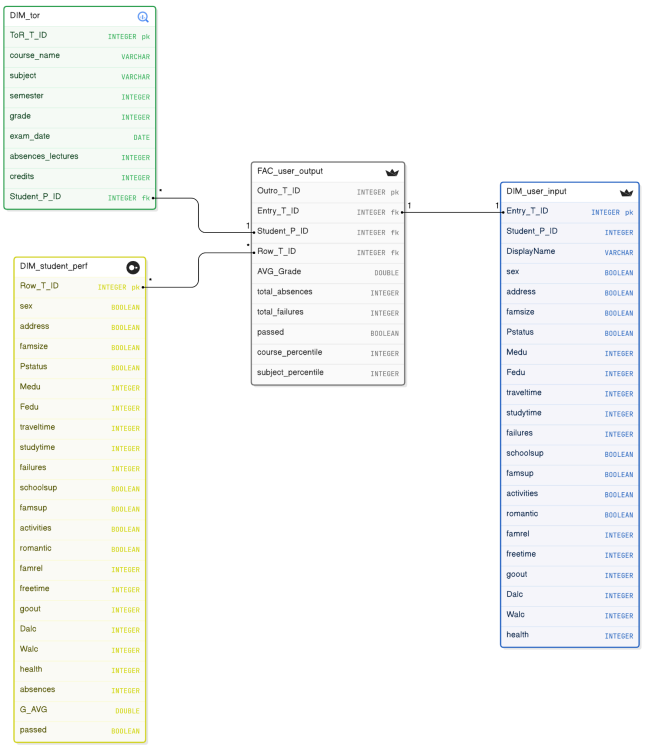


Figure 2: Data Model of EduInsight

b. Technologies

All the technologies involved in the project are meant to smoothly interact with each other. The following is a complete list of them:

1. *Google Cloud*: a comprehensive cloud solution to deploy and scale applications. In this project it has been used To store and query a public dataset. This choice is motivated by a combination of high performance, high levels of security and a friendly user interface. In particular, the components used are:

Google Cloud Storage: to safely store a dataset in a bucket object.

Google BigQuery: to query a dataset in a fast and performant way.

2. *DuckDB*: A database which puts its ease of use as one of its main advantages. It doesn't require any complex server configuration, making it fit easily with the needs of this project.

3. *Docker*: An open platform designed to develop and run applications inside portable containers. It ensures the project is run on different environments, avoiding configuration problems.

4. *Dask*: A python library to scale data processing. Easy to use, it allows to operate with a dataset, no matter its size, and it integrates well with machine learning libraries.

5. *Streamlit*: It's the best option to create a front-end when working with Python. The implementation is fast and it integrates with visualization libraries quite well.

6. *Scikit-learn*: Famous python all in one machine learning library. It offers everything the project needs in terms of preprocessing, ml-model building and evaluation.

iii. Implementation

The first step has been to collect datasets regarding students' performance. Kaggle and Google datasets are the instruments used for it. At the end of the search, two datasets have been collected.

A docker container in which to group all the code has been initialized, with python version set to 3.11, containing a file to list all the libraries and modules necessary to make the project work.

The two datasets have been processed and combined and the resulting one stored in DuckDB.

To simulate access to the student's university database, some processes are followed:

1. At first, a table of student records is built through Python code and its parameters (studentID, course name, subject, semester, exam date, grade, absences, credits) are filled according to realistic criteria. The table is then stored in DuckDB and saved as a CSV.
2. A Google Cloud project is initialized. The table of records is stored in a bucket and from there copied as a table in a BigQuery dataset.

3. A service account with the necessary authorizations is created and Google BigQuery API is enabled. A JSON key associated with the service account is generated. This is fundamental to establish a remote connection to BigQuery API.

Multiple machine learning models have been trained on the dataset. The model development process includes the evaluation of various machine-learning algorithms to predict student performance accurately by using a grid search to find fitting model parameters. It was trained on datasets containing records from multiple academic terms. In the actual prediction later on different models are used and their predictions are collected and converted into a list of prediction results. The majority of the prediction results is part of the outcome displayed to the user. The objective of the prediction is to provide the student with a binary response (yes or no) indicating the probability of success in their studies, based on a set of variables.

Using Streamlit, a dashboard is implemented in which the user adds his student ID and some demographic, familiar and social data.

The student ID is used to retrieve the student's data from the table of records located in the cloud. The table is accessed and queried through the BigQuery API. Different functions return only specific and necessary parts of the entire table as a DataFrame, to ensure the data is handled in memory for quicker calculation times. Subsequently a report is displayed to the user.

iv. Results

The user faces up with a form which has to be filled with relevant information.

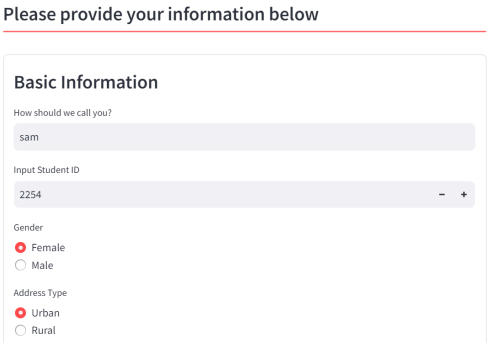
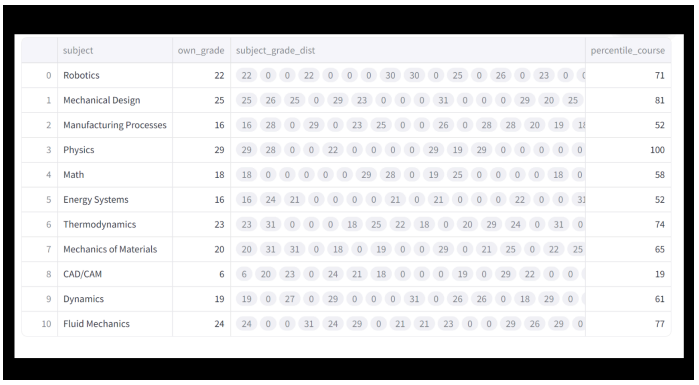


Figure 3: first part of the form

Once the form is submitted, an output is displayed back.



	subject	own_grade	subject_grade_dist	percentile_course
0	Robotics	22	22 0 0 22 0 0 0 30 30 0 25 0 26 0 23 0	71
1	Mechanical Design	25	25 26 25 0 29 23 0 0 0 31 0 0 0 29 20 25	81
2	Manufacturing Processes	16	16 28 0 29 0 23 25 0 0 26 0 28 28 20 19	52
3	Physics	29	29 28 0 0 22 0 0 0 0 29 19 29 0 0 0 0 0	100
4	Math	18	18 0 0 0 0 0 0 29 28 0 19 25 0 0 0 0 18 0	58
5	Energy Systems	16	16 24 21 0 0 0 0 0 21 0 21 0 0 0 22 0 0	52
6	Thermodynamics	23	23 31 0 0 0 18 25 22 18 0 20 29 24 0 31 0	74
7	Mechanics of Materials	20	20 31 31 0 18 0 19 0 0 29 0 21 25 0 22 25	65
8	CAD/CAM	6	6 20 23 0 24 21 18 0 0 0 19 0 29 22 0 0	19
9	Dynamics	19	19 0 27 0 29 0 0 0 31 0 26 26 0 18 29 0	61
10	Fluid Mechanics	24	24 0 0 31 24 29 0 21 21 23 0 0 29 26 29 0	77

Figure 4: part of the output

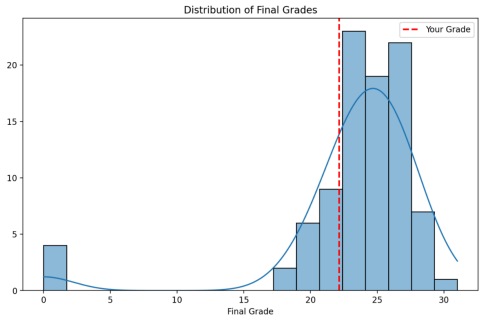


Figure 5: part of the output

The output contains a report of the student's performance and a prediction concerning his future success in the studies. Here the user can also find a comparison between him and his peers in terms of performance.

information on how to run the code are explained in details in the readme file of the github repository.

v. Conclusions

The personalized education platform is designed to enhance student engagement and learning outcomes through the use of big data analytics. The multi-stage recommendation system fulfils the task of predicting student performance and providing personalized insights by jointly exploiting various data sources, machine learning models, and user-friendly interfaces. The system implemented is proven to be suitable for small-scale demonstrations with a limited number of students and courses. It would in principle require adjustments to handle an increased amount of data in a real-world educational setting. Furthermore the solution works on the fundamental prerequisite of being able to query the university database, this can be made possible by either a public access, in the ways addressed in this project, or by being granted the access through permission.

Technical problems regarding dask and streamlit have been dealt with: Dask is slow on small datasets, like the one used, it doesn't support all pandas functions, making some workarounds needed,

furthermore groupby aggregations are slow due to dask scheduling. Streamlit doesn't support real time updates, it's slow when operating with large datasets and the layout options are limited.

Another limitation has been found in the google cloud service account key. Since 16 of June 2024, for security reasons, google cloud automatically disables keys which go public, it doesn't matter the level of authorizations given to the key. For this reason the github repository, public at first, has been set to private. The intention for the immediate future is to find an alternative approach in order to maintain the key hidden while still being able to run the code.

Enhancements regarding the platform capabilities are also planned:

- To implement a document database (MongoDB) to show the student his upcoming exams.

- To move the scikit-learn ml algorithms to Apache Spark.

- To make the prediction more accurate.

Despite the limitations, the results obtained highlight a promising starting solution which can be efficiently enriched by future improvements, potentially revolutionizing personalized education through data-driven insights.

References

[1] Streamlit official documentation. Available at https://docs.streamlit.io/

[2] DuckDB official documentation. Available at https://duckdb.org/docs/index

[3] Google Cloud official documentation. Available at https://cloud.google.com/docs?hl=it

[4] Dask official documentation. Available at https://docs.dask.org/en/stable/

[5] Docker manuals. Available at https://docs.docker.com/manuals/