# Machine Learning End-To-End Project
# LAZY TEACHER

Introduction to Machine Learning | M1 ISD | 2023-2024


Université Paris-Saclay

**Contributors**

Ilan ALIOUCHOUCHE

Ilyes DJERFAF


## Introduction

As part of our Introduction to Machine Learning project in Master 1 ISD at Université Paris-Saclay, we were required to create a model capable of automatically grading essays from a professor's students. The goal was to explore textual data (NLP), model training, and result interpretation.

## Approach

Our approach is unconventional, diverging from standard expectations in data exploration, model training, and evaluation. We crafted a unique pipeline that sets our project apart from typical supervised learning initiatives in NLP. By integrating statistics, AI, and intuition, we developed a comprehensive end-to-end project. Our methodology encompasses four primary components: 1. Exploratory Data Analysis, 2. Statistical Techniques for Unbalanced Data, 3. Advanced Machine Learning Modeling, and finally, 4. API & Application Deployment

## 1. Exploratory Data Analysis (EDA)

In this data cleaning and exploration phase, we attempted to optimize our dataset and extract properties.

- We used the vocabulary of a GloVe model trained on Wikipedia as a reference for Analyzing Vocabulary Coverage.
- Upon analysis, the words not recognized by the model were errors made by the students.
- We were able to confirm with the Mann-Whitney U test that these errors are significant information. (Figure 1)

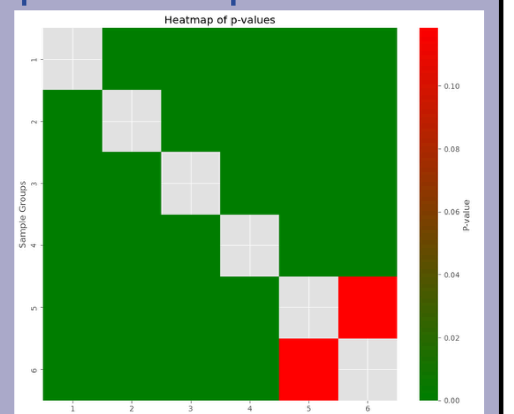


Figure1 Mann-Whitney U test

- Cleaning and correcting these errors allowed us to significantly reduce the size of our vocabulary thanks to the Levenshtein distance.

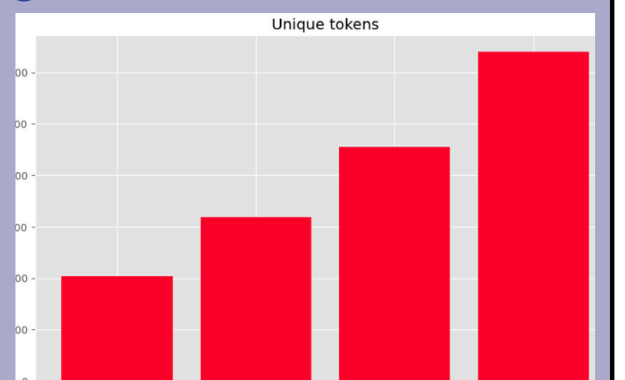


Figure2 Vocab Size Evolution

- With UMAP and embeddings generated by models like BGE, we were able to discern clusters.
- These clusters were analyzed using KMEANS and the LDA algorithm (latent dirichlet allocation) (Figure 3 & Figure 4).

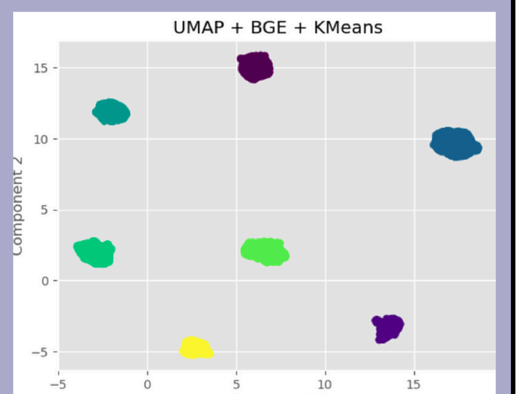


Figure3 Clustering Visu

- Following this exploration, we hypothesize that there are different types of exams and/or different questions in our dataset.

In the following section, we present the statistical study for the problem of unbalanced classes.

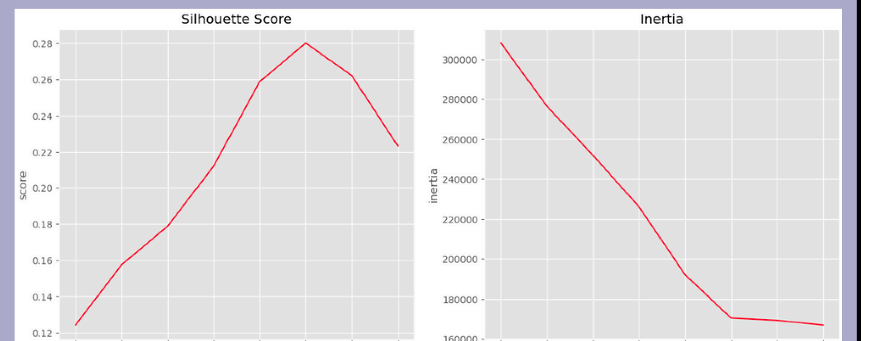


Figure4 Inertia & Silhouette Analysis

## 2. Statistical Techniques for unbalanced data

Handling imbalanced data, as shown in Figure 5, is essential to avoid model bias towards the more frequent classes.

1- A preliminary statistical study was conducted to determine the sample size range from a population with n samples that perfectly represents the population without any loss, based on the margin of error.

This is known as Sensitivity Analysis on Margin Error.

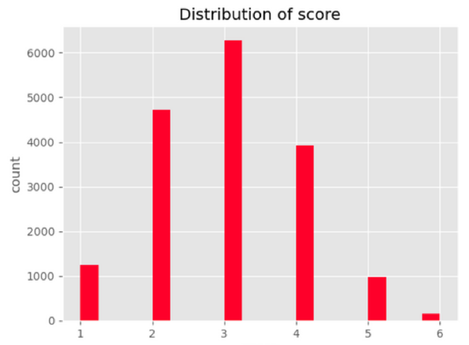


Figure5 Scores Distribution

- The study indicated that from 8,881 rows, we can derive a data sample that accurately represents the base sample. (Figure 6)

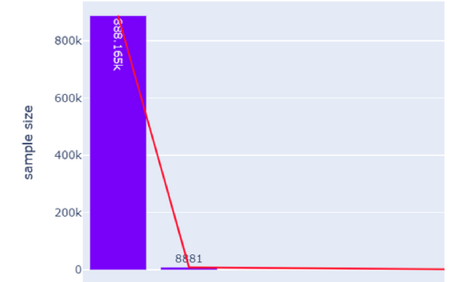

Figure6 Margin Error Analysis

2- To formally establish an undersampling threshold, a relevant approach is to use Hoeffding's Inequality

- Hoeffding's Inequality is commonly used t necessary number of data points based on precision and confidence level.
- Figure 7 displays the various thresholds obtained for certain errors and precisions.
- We will focus on the threholds with an error of 0.01

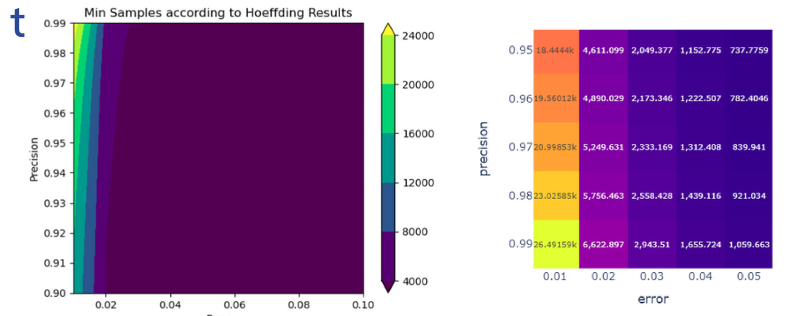


Figure7 Hoeffding's Thresholds

- Figure 8 illustrates the impact of the threshold on the data to determine how many labels are going to be undersampled versus how many will be oversampled.

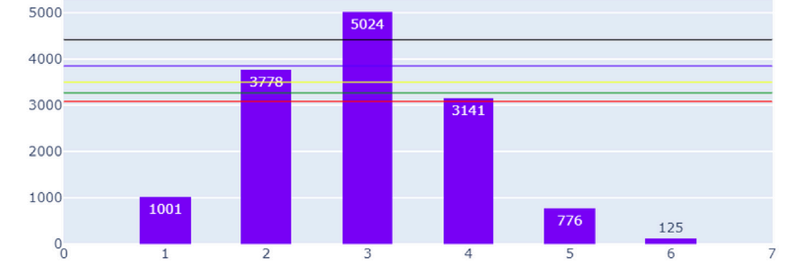


Figure8 Impact of the threshold

- We have concluded that the adequate threshold is 3,084, as it splits the classes in half: 3 classes to oversample and 3 classes to undersample.

3- The next steps were to use some techniques of undersampling and others of text augmentation to balance the corresponding label.

## 3. Advanced Machine Learning Modelisation

- In the modeling phase, we started by using classic machine learning models such as naive Bayes and logistic regression. Given the mediocre performance during training where the models were clearly underfitting, we logically turned to XGBoost, as the following plot shows the F1 macro and weighted scores remain poor (Figure 9)
- We will now move towards pre-trained models to try to optimize performance. As a result, we will use an autoencoder transformer.

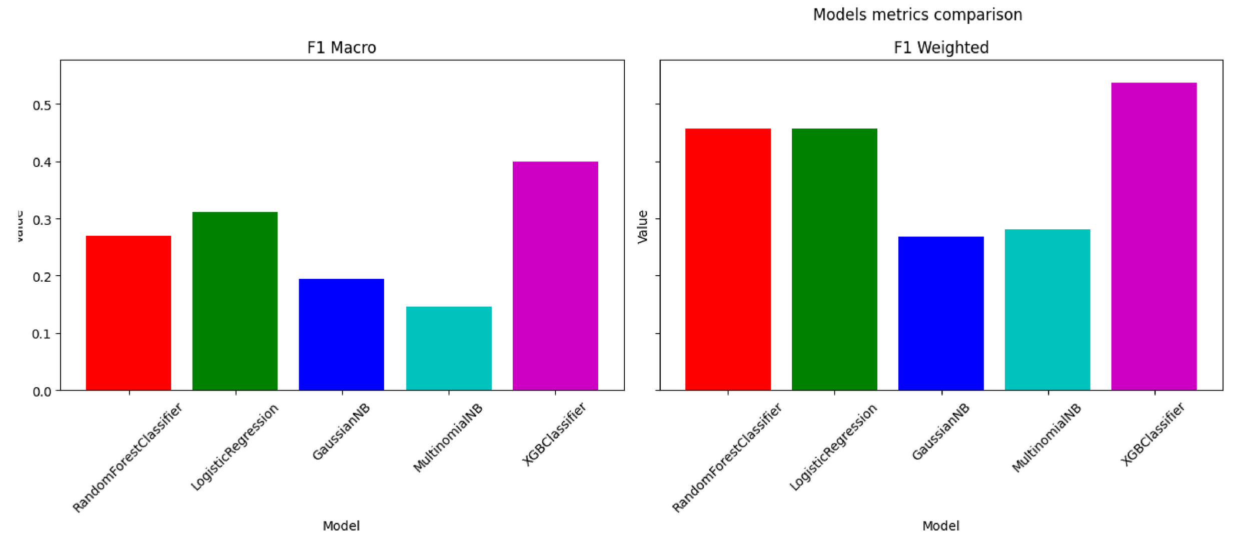


Figure 9 F1 Scores of ML models

- For model selection, we referred to MTEB, which benchmarks embedding models on Natural Language Understanding tasks. We found gte-base-v1.5 available on Hugging Face, which is effective for classification with an input size larger than that of the student essays. The model was trained on the base dataset, oversampled, and with a weighted loss function to manage the imbalance. This was the best performing model (Figure 10) and is availbale at : *https://huggingface.co/ilanaliouchouche/gte-base-lazy-teacher*


Figure 10 Benchmark of GTE fine tuning

- We then analyzed the errors made by this model to thoroughly explore the model's behavior during training (Figure 11). The model understood the concept of label ordinality well. We also attempted to determine if certain types of exams (refer to EDA) are more difficult to predict (Figure 12).

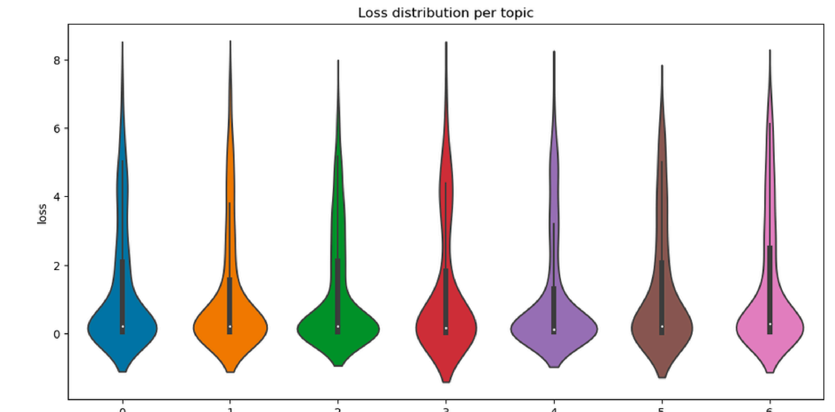


Figure 12 Error by exam type

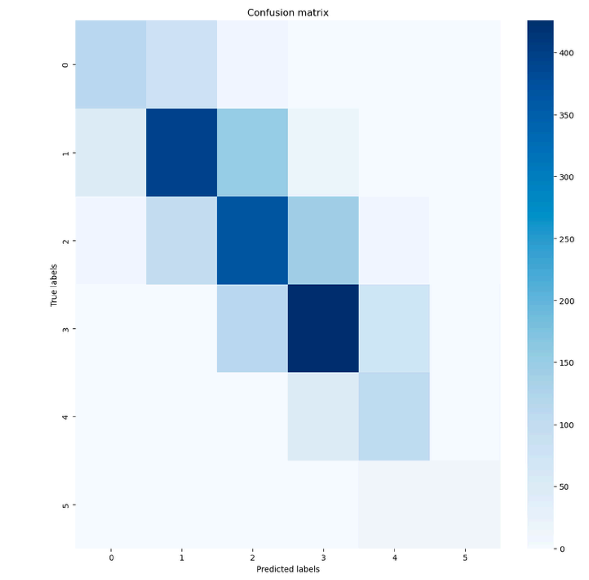


Figure 11 Error by exam type

## 4. API & Application Deployment

Once we have finished the modeling and saved our model, comes the deployment step.

- Using the framework FastAPI, we have built an endpoint allowing us to call the model asynchronously.
- Then, we have developped a User-Friendly Interface (Figure 13) using HTML, CSS and JS.

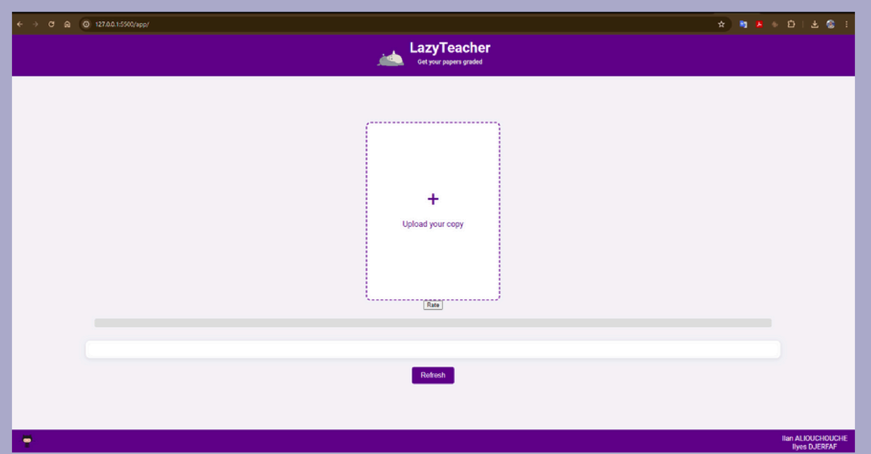


Figure13 Application Front-End

- To run the application, please follow the documentation proposed in this link : *https://github.com/mlengineershub/LazyTeacher/tree/main/app*

## Conclusion

Our project presents an end-to-end solution, methodically progressing from data analysis to the development of a user-friendly application that predicts the grades of student essays.