Course: Python Machine Learning Labs

Project: Develop an end-to-end Machine Learning Pipeline

Instructor: Assan Sanogo

Prepared by: Jana Saleh (GitHub repository of the project)

I. Introduction

A. Project Summary

The goal of this project is to develop an end-to-end pipeline capable of processing essays and providing an output grade that describes the level of English proficiency.

B. Project Goal

By carefully assessing different linguistic elements like grammar, vocabulary, and flow, these forecasts offer important information about people's writing abilities. Their goal is to have a dependable tool that can evaluate how well new students can write in English using the IELTS grading system. This tool would also assist potential students in understanding how much effort they need to put in to reach the next level.

C. Project Background

Writing in English is a key part of language education. Nowadays, with the role played by technological advancement in the education, being able to predict someone's score in English is super important. With the help of advanced technologies and techniques, such as NLP and Machine learning, , we will be able to make this task (grading English essays) easier and faster. Our ambition is to have a reliable tool to assess new students' ability to write in English according to the IELTS grading system.

D. Overview of the ML Pipeline

The developed ML pipeline is designed to predict English essay scores automatically. It involves , briefly, preprocessing the data, selecting features, and training machine learning models. These models are then used to evaluate and score essays efficiently.

II. Data Preparation

A. Data Collection and Cleaning

This project is based on a dataset of 7000+ essays graded by English specialists and grouped into 8 essay sets for which we will want to predict a score that will be referred to later on as **"domain1_score"**. Initially, we conducted a quick examination of our dataset's content, composition, shape, followed by a more in-depth exploration. We then intuitively checked for any

"NaN " values that could potentially disrupt our ML pipeline flow later on. Based on this inspection, we were able to identify a single Nan value for a score to predict. Due to the dataset's size, since it is considered a large one, we decided to remove it to save our dataset's integrity. Additionally, our dataset includes a unique essay_set category (essay_set = 2) that requires special attention. For this type of set, we will have to predict in addition to our **"domain1_score"** mentioned before, a second score named **"domain2_score"**. After verifying the validity of the common score prediction **"domain1_score"**, we checked for NaN values in the second score prediction **"domain2_score"** and validated it as well. Moving forward with our data exploration, we analysed the score distributions with respect to each category. After analysing the distributions of scores for each essay set, we found that for some scores we have an unequal score density distribution which suggests that within the different categories, we have an uneven dispersion of the scores, and some scores are severely under-represented.

Fig. 1. a) *Distribution of domain1_score for essay set 1* ***b)*** *Distribution of domain2_score for essay set 2* ***c)*** *Distribution of domain1_score for essay set 8*

To resolve this issue of dataset unbalancing, several techniques come to mind including: SMOTE, ADASYN, and Random Over Sampling (ROS). After certain reflexion regarding the SMOTE technique and its concepts, we didn't want to go to a normalization procedure that might get in the way of our goal. For essay scores, we definitely have grades outside the normalization interval [0-1]. For our application, if we proceeded with the normalization, we will have to bounce back from this normalization. Thus, we tried to avoid the SMOTE algorithm. Next, we started exploring the ADASYN technique, several problems arised. In fact, for our dataset, for the scores that were severly underrepresented, the ADASYN technique couldn't handle them. It was always generating an error since the ADASYN technique is based on k-neighbourhood density to oversample, for these cases with very few samples, it was unable to apply its k-neighbourhood density concepts and generate new synthetic samples. Finally, we were left with the random oversampling. Based on a bibliographical article [1], ROS presented good results by comparison to the remaining two-techniques, even though it was coupled with other ML concepts. This latter, gave us the motivation to use ROS to balance our dataset and check the behaviour of our in-development model regarding this method of oversampling.

B. Feature Engineering

After preparing our data, we will start now with our feature engineering. In the process of feature engineering, we calculated various features to enhance the predictive power of our model. Our first feature, essay length, provides insights into the composition of each essay, with initial observations revealing outliers in the distribution of essay lengths.

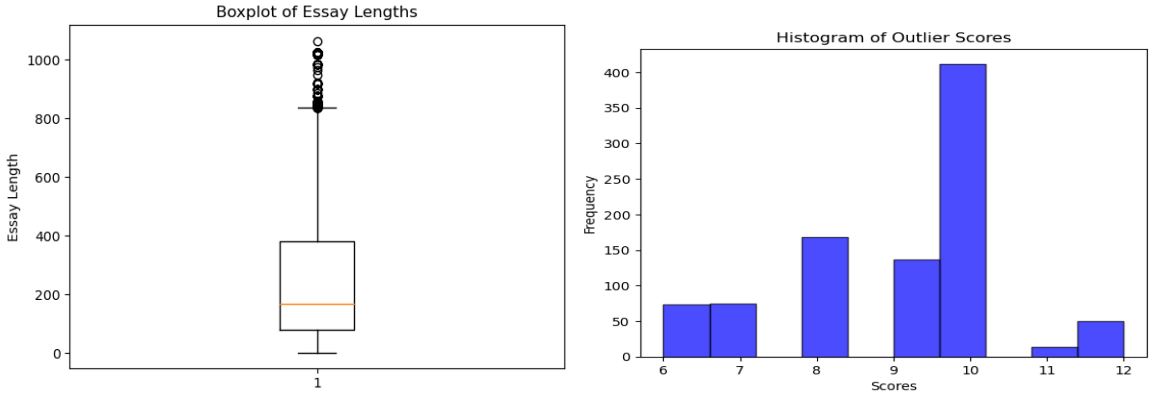
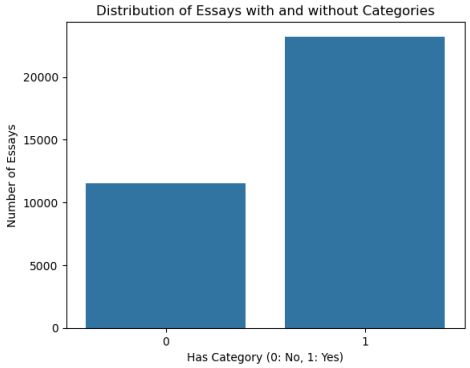


Fig. 2. a) *BoxPlot for essay length* ***b)*** *Distribution of outliers scores*

First, we can notice that the frequency of the presence of outliers is high. Additionally, from the above plots, we can make a hypothesis that says, that our long essays are exhibiting high scores. In fact, since at the end, our goal is to predict a score, these outliers can contain valuable insights regarding our data. It is playing role in the scoring of its respective essays. So, based on that in addition to their amount, we should keep them. Their presence is more valuable than their absence. Additionally, we identified tagged words in the essays (words with special syntax composed of "@" followed by couple of characters), categorizing them to assess their frequency within each category. To ensure the usefulness of this information, we calculated the percentage of tagged words relative to the total essay length, allowing for fair comparison across essays. In other words, Since each essay has a distinct length, the count of tags can't be really representative unless we calculate their percentage of occurrence within each essay (their % with respect to the total essay length). Additionally, we proceeded with checking their frequency in order to know how we will manipulate them. According to the results, we can see that the majority of essays no matter their set, contain at least one tagged word, thus, keeping them might be valuable. Moving forward, we explored structural aspects of the essays, beginning with the sentence-to-paragraph ratio, which offers insights into the level of detail and cohesion within the essays. We then used spaCy for language processing tasks. Following this, we digged more deep into lexical characteristics, computing ratios such as long word token ratio and lemma token ratio to better understand the vocabulary richness and complexity. Finally, we introduced a range of additional features, including those related to sentence structure, readability, and grammatical diversity, to support further our analysis and prediction capabilities. Here's a glimpse of some of the features calculated to every aspect of language assessments, providing a comprehensive set of attributes for analysis and prediction [1].



Essay length	**Short word token ratio**	**Fourth root of the number of word tokens**	**Minor delimiter ratio**	**Interrogative relative determiner ratio (HD)**	**Possessive form ratio (PS)**	**Counting word ratio (RG)**	**Transition words count**	**Adjectives count**
Tagged words count	Lemma token ratio	OVIX (Word Variation Index)	Major delimiter ratio	Participle ratio (PC)	Preposition ratio (PP)	Ordinal counting word ratio (RO)	CTTR (Corrected Type Token Ratio)	Adverbs count

Sentence-to-paragraph ratio	Token sentence ratio	Nominal ratio (NR)	Particle ratio	Paired delimiter ratio (PAD)	Adjective ratio (JJ)	Pronoun ratio (PN)	Number of sentences	Nouns count
Tokenization	Non-initial caps word ratio	Conjunction ratio	Relative adverb ratio	Passive voice ratio	Interrogative relative pronoun ratio (HP)	Pronoun in object form ratio (OBJ)	Flesch Reading Ease score	Verbs count
Part-of-speech tagging	Char sentence ratio	Sub conjunction ratio	Determiner ratio (DT)	Active voice ratio	Foreign word ratio (UO)	Verb ratio (VB)	Gunning Fog index	Word variation index

III. Model Development

A. Model Selection

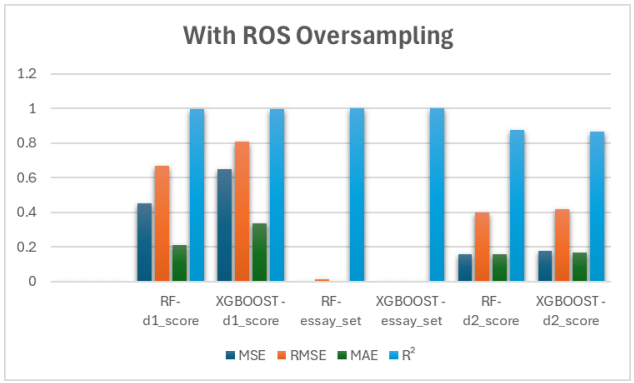
After preparing and organizing our data and engineering its features, we're ready to select and train our machine learning models. We've decided to consider two models based on research from the state-of-art ([2], [3], [4]): Random Forest (RF) and XGBOOST. These 2 models prove being flexible with imbalanced data and normalization (that we tried to avoid at the beginning). In fact, Random Forest for example, handles implicitly the imbalance, since it's based on decision trees constructing mechanism using random features, this randomness will help with lowering the effect of the data imbalance. Which makes it as well robust when it comes to dealing with outliers. On the other hand, when it comes to XGBOOST, it internally incorporates techniques that will play a role in preventing that our model overfit especially in the case of imbalanced dataset where overfitting can become a serious issue.

B. Training Procedure

First, we start the training process with an 80% split of the training data. However, to optimize model performance, we tried a 75% split. During the evaluation, we found that the results were worse than those for the 80% distribution. Recognizing the superior performance of larger training portions, we decided to keep the 80% of the training data split.

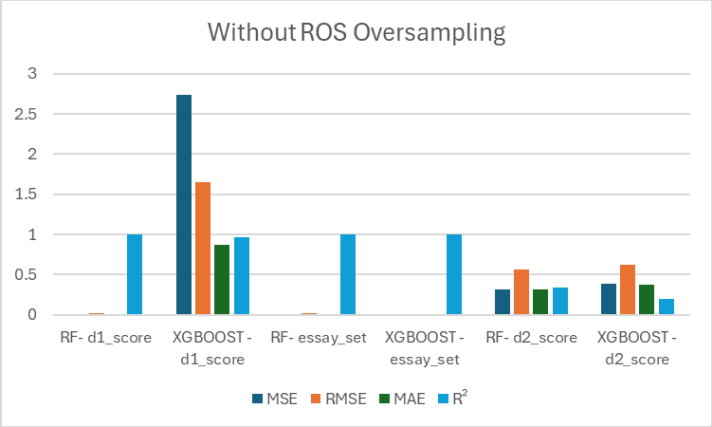
C. Model's Evaluation

For model evaluation, we used RF and XGBOOST. Additionally, we explored the impact of oversampling on the dataset and compared the results with the non-oversampled dataset. The metrics employed for evaluation were mean squared error (MSE), root mean squared error (RMSE), mean absolute error (MAE), and coefficient of determination (R2 score). MSE and RMSE quantify the average squared and square root of errors, respectively, providing insight into the magnitude of prediction errors. MAE calculates the average absolute errors, offering a measure of the model's accuracy. R^2 score tells us about the goodness of fit of the model. Each metric used provides valuable information for understanding the performance and effectiveness of the models in predicting essay scores.



The provided results show how well two different types of models, RF and XGBoost, perform at predicting various aspects of essay scores. they confirm as well what was mentioned in the provided publications and articles. For predicting the scores given to essays (Domain1 and Domain2), the Random Forest model generally performs better than the XGBoost model. This is shown by lower error values (MSE, RMSE, MAE), which indicate how far off the predictions are from the actual scores. Additionally, both models do a great job at explaining the variability in the data, as indicated by high R-squared (R^2) values, with the RF model slightly better than the XGBoost model. However, when it comes to predicting the essay set, both models give us near-perfect accuracy. This means that they are highly successful at correctly identifying which set an essay belongs to. However, these great results can also be an indidcator that our model is presenting an overfitting behavior when it comes to predicting essay_set category.

Now let's check the results for the case of dealing with rawdata and not the processed ones.



The analysis shows that the RF model performs quite well in predicting domain1_score and essay_set. For domain1_score, it has moderate errors, with an average error of about 2.43 and a difference of 1.56. The average absolute error is around 0.81, indicating some level of accuracy. However, when predicting essay_set, RF excels, showing very low errors, with an average error of only 0.00039 and a difference of about 0.02. The absolute error is also very small, around 0.00039. Furthermore, it achieves a near-perfect R^2 value close to 1.0. Conversely, the XGBoost model shows slightly higher errors in predicting domain1_score, with an average error of 2.74 and a difference of approximately 1.66. The absolute error is about 0.87, which is an acceptable accuracy. However, XGBoost performs exceptionally well in predicting essay_set, with no errors and a perfect R^2 value of 1.0. This illustrates its higher accuracy in this prediction compared to RF.

IV. Conclusion and Future recommendations

Comparing the results of models trained with and without oversampling, it's clear that oversampling generally leads to better performance across different evaluation metrics. For example, in the case of domain1_score and essay_set predictions using RF, oversampling significantly reduces the MSE and MAE while increasing the R^2 value, which indicates a better

model fit. Similar behavior is observed in the XGBoost model's performance metrics. In fact, in the oversampled scenario, the MSE for essay_set predictions using XGBoost is reduced to zero, indicating perfect predictions. However, oversampling doesn't always guarantee better results; for example, in the case of domain2_scores, the RF model shows a higher MSE and RMSE with oversampling compared to the non-oversampled model, which means that the oversampled data might have introduced noise or overfitting in our case. Therefore, to address our challenge, we can predict our domain1_score essays and predict the set they belong to using the first model after oversampling the data and after addressing the imbalanced data issue. Furthermore, whenever domain2_score is needed to be predicted, we can use the model trained on raw inital data to get better results.

V. Références

- https://www.scitepress.org/PublishedPapers/2021/103775/103775.pdf
- https://www.researchgate.net/publication/360456237_A_Survey_on_Automatic_Essay_Evaluation_System_using_Machine_Learning
- https://web.archive.org/web/20220809171145id_/http://www.icicel.org/ell/contents/2022/1/el-16-01-11.pdf
- https://link.springer.com/article/10.1007/s10462-021-10068-2#Sec2
- https://www.diva-portal.org/smash/get/diva2:602025/FULLTEXT01.pdf