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

English Quality Prediction

Instructor: Assan Sanogo

I) OVERVIEW

This report outlines the development of a machine learning model aimed at predicting the rating of essays based on their content. The project encompasses the complete lifecycle of machine learning model development, including preprocessing, feature extraction, model selection, training, evaluation, and deployment. The goal is to automate the essay evaluation process, providing a scalable and objective tool for educators and institutions to assess writing quality.

Project hosting and deployment: https://github.com/Dalysko/End-to-end-English-Quality-Prediction

II) Data Analysis

Before starting to work on the database, it is crucial to thoroughly examine and analyze its contents. This analysis will enable us to formulate a comprehensive normalization strategy and gain deeper insights into our data. The data set of the essay rating, scoring and evaluation essays. In the data set, there are 12976 rows and 28 columns.

According to the essay set provided, essay_id is a unique identifier for each essay. The essay_set is likely a representation for essay based on the grade level, topic, or another criterion. The essay column is the beginning of the essay or topic.

The rater1_domain1, rater2_domain1, rater3_domain: Scores were given by different raters according to their evaluation criteria. It is unclear whether rater3_domain1 is used consistently across all the entries since there are missing values.

Domain1_score: Seems to be the addition or final score for the first domain of evaluation (an average score from the first three raters).

Here are some initial observations and considerations for beginning the analysis:

Scoring consistency: Observing to see if the scores given by given raters were consistent by using statistical tools.

Handling of missing data: Some were missing or completely have not much meaning like rater3_domain. Method to do with missing data were determined either by imputation, exclude or a model to hand missing data.

Score Aggregation: The means/methods for combining raters individual score to the final domain1_score is to be determined. It's important to mean is it is average score, weighted score which could affect the outcome.

Score distribution: The distribution of the scoring analysis can provide a significant useful insight into scale's effectiveness. Are most essays concentrated around a specific score, or do they exhibit a broad range of scores?

Essay sets Analysis: The 'essay_set' the column indicates there are different sets of essays. This analysis explores whether a particular or a sets essay receives consistently lower or higher, which might indicate a bias or a difference in difficulty or grading standards.

Rater Bias: It was important to investigate any rater biases. For instance, does 'rater1' constantly give higher or lower scores compared to rater2 and rater3.

Content Analysis: The actual od the essay is very important and crucial. There might be a need to perform textual analysis for keywords, essay length, topic and references correlate with higher or lower scores.

BOATENG Maureen
DIALLO Thierno Ibrahima
SOULEYMAN Mahamadene
DANILINA Elizaveta
SISSOKO Daly

Domain Analysis: There are at least two domains, it will be good to compare the emphasis each rater places on different domains. This could indicate if evaluators share a common understanding of what is significant in an essay.

Investigating the Relationship Between Essay Length and Scores: It would be useful to examine whether a link exists between the extent of the essay content and the assigned scores. Given that the dataset only displays the start of each essay, the availability of the complete text for analysis remains uncertain.

Statistical Exploration: Using descriptive statistics serves as an effective initial step, offering summaries of the scores' central tendency, variability, and distribution shape. Subsequent inferential statistical methods can be employed to examine hypotheses regarding variations in scoring across different groups.

III) Methodology

A) Data Preprocessing

Data preprocessing is one of the critical steps in any machine learning project, encompassing the essential tasks of data cleansing and formatting prior to its utilization in machine learning algorithms.

To train the model effectively, our focus was on preparing the data. While our table contains crucial information within the "essay" column, it won't be directly usable for machine learning training due to its unprocessable data type. Therefore, we have chosen to transform the text into a vectorized format using various functions and libraries in this phase.

Additionally, we possessed two target columns, namely "domain1_score" and "domain2_score." Within this section, we undertake normalization of these columns to streamline model learning. Furthermore, amalgamating these two columns into a single target column is pursued, thereby enhancing model simplicity and facilitating more accurate results.

In the context of Natural Language Processing (NLP), these preprocessing steps encompass a series of tasks, which include:

a) Normalization of domain scores

In this section, we reassessed the ratings provided by evaluators. This involves analyzing the scores relative to the maximum score achievable for each essay type. For essay set 2, we evaluated the relationship between the two scores collectively. These results will be categorized into three tiers: 0 (bad), 1 (good), and 2 (excellent).

The separation is done by dividing the scores by 3 equal parts. For example, essay set 1 can have from 2 to 12 points. We will therefore divide these points into three parts:

- from 2 to 5
- from 6 to 9
- from 10 to 12

Then we take the maximum ratio of each part:

- 5 -> 5/12 = 0.41666 = 0.42
- 9 -> 9/12 = 0.75

Using these values, we will use these intervals to determine the result:

- [0, 0.42] --> 0 (bad)
-]0.42, 0.75] --> 1 (good)
-]0.75, 1] --> 2 (excellent)

After some calculations we determined that the intervals that will fit under all essay sets are:

- [0, 0.42] --> 0 (bad)
-]0.42, 0.75] --> 1 (good)
-]0.75, 1] --> 2 (excellent)

These are the intervals we will use in our function.

Using these calculations, the function produces an end-result.

The function itself at the input takes the whole database. Within the function we establish a 'maxi' n array, which contains the maximum values of each essay set. The output of the function produces a database with the result as a new column named ‘score’.

Following that, we can delete the columns ‘domain1’ and ‘domain2’ replaced them with 'score'.

b) Word Structure

- **Word Count**

  The function Word Count calculates the number of words in the essay text. It employs a lambda function which takes the input text of the essay, splits it into words using the .split() method, focusing on spaces, and finally returns the count of words as the output.

- **Counting the number of unique word and the number of adjectives**

  We used spacy to count unique words and adjectives in a text by utilizing its tokenization and part-of-speech tagging capabilities.

- **Counting the number of references @ in each essay**
  We decided to count the number of references @ in a text. These placeholders are often used to represent specific types such as
  @NUM1numbers or numerical values, @CAPS1: capitalized words, @PERCENT1: percentage values, @MONTH1: months or month names.

c) Readability Metrics

Readability metrics provide an approximation of the complexity of a text, aligning with the U.S. grade level of education, ranging from 0 to 18. These metrics gauge how easily a text can be comprehended, with higher scores indicating greater ease of readability.

- **The Flesch Reading Ease**
  The Flesch Reading Ease scale assigns a score ranging from 1 to 100 to a text, with 100 indicating the highest readability. A score falling between 70 and 80 corresponds to an eighth-grade school level. Therefore, such text should be comfortably understandable for the typical adult reader.
  $206.835 - 1.015\left(\frac{total\ words}{total\ sentences}\right) - 84.6\left(\frac{total\ syllables}{total\ words}\right)$

- **SMOG Index**
  The SMOG Index stands for 'Simple Measure of Gobbledygook,' estimates the years of education required to comprehend a piece of text based on the number of polysyllabic words present.

$$score = 1.0430\sqrt{nombre\ de\ polysyllabes + \frac{30}{nombre\ de\ phrases}} + 3.1291$$

- **The Flesch Kincaid Grade Level**
  The Flesch Kincaid Grade Level serves as a commonly employed tool for measuring the readability of a text, providing an estimate of the reading grade level required to comprehend it. For instance, a Flesch Kincaid level of 8 implies that a reader needs a reading proficiency equivalent to grade 8

or higher to grasp the content. This measurement not only accommodates advanced readers but also suggests that the text is relatively quicker to read.

$$0.39\left(\frac{total\ words}{total\ sentences}\right) + 11.8\left(\frac{total\ syllables}{total\ words}\right) - 15.59$$

- **Dale-Chall Readability**
  The New Dale-Chall readability score assesses a text by comparing it to a set of words typically known by fourth graders. If there are many unfamiliar words, the reading level is higher. Conversely, a lower score indicates better readability for a fourth grader.

$$0.1579\left(\frac{difficult\ words}{words}\ X\ 100\right) + 0.0496\left(\frac{words}{sentences}\right)$$

- **Coleman-Liau Index**
  The Coleman-Liau is a readability formula which shows the reading level of a text. It uses sentences and letters as variables.

$$\mathrm{CLI} = 0.0588\mathrm{L} - 0.296S - 15.8$$

L = average number of letters per 100 words

S = average number of sentences per 100 words

d) Lexical Measures

Lexical measures analyze text based on its vocabulary.

- **Lexical diversity**
  Measures the variety of different words used in a text, indicating how varied or repetitive the vocabulary is.

- **Measure of Textual Lexical Diversity (MTLD)**
  Measure of Textual LD is equal to the mean length of sequential token strings in a text that maintains.
  Algorithm:
  Divide text into segments with TTR value of 0.72
  Dividing the number of words by the number of segments
  The improved version of the algorithm assumes two text passes - in direct and reverse order, and further averaged metric values.
  Our input function accepts text, processes it through the function, and outputs calculated MTLD number.

e) Syntactic Structure Measures

In the syntax index we looked at the text from the syntax side.

Functions accept text at input, divide it into words and give the counted number.

- **Average length of words and sentences**
- **Roots of Sentence Tree**
  A syntax tree represents the syntactic structure of a sentence according to some formal grammar.

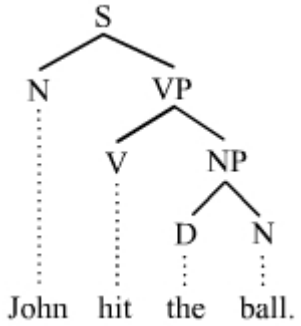

FIG. 1. Parse tree

- **Syntactic complexity**
  Syntactic complexity refers to the range and the degree of sophistication of the forms that appear in language production.

f) Quality Measures

We focused on text quality measures from a grammatical perspective.

- **Misspelling score**

To calculate the number of typos, we will use the "spellchecker" library.
It uses a Levenshtein Distance algorithm to find permutations within an edit distance of 2 from the original word. It then compares all permutations (insertions, deletions, replacements, and transpositions) to known words in a word frequency list. Those words that are found more often in the frequency list are more likely the correct results.
Functions accept text at input, divide it into words and give the counted number.

- **Over-usage of punctuation**

Over-usage of punctuation refers to the excessive or improper use of punctuation marks, such as periods, commas, exclamation marks, and others, within a piece of text.
A higher count indicates a text with more frequent use of punctuation marks, which might suggest more complex or expressive writing.
Conversely, a lower count indicates a text with less frequent use of punctuation marks, which might suggest simpler or more straightforward writing.

g) Emotion Analysis

Analyzing emotions from text involves the task of forecasting the emotional state conveyed within a written passage.

B) Balancing and Modeling

Before starting the training and evaluation of the model, we have re-examined our database after normalization to see if the data has changed.

This step was crucial for ensuring the effective instruction of our model. Additionally, following normalization, the "essay" column is no longer needed as it has already been vectorized.

The graphics indicate variations in the number of essay sets and unbalanced scores within each set, potentially limiting the model's ability to learn effectively.

To address these issues, the model undergoes testing and training. The focus is on balancing the data to ensure the model receives the most informative input.

The data is divided into training, testing, and verification sets. Initially, the data is split randomly into a 0.16 ratio for testing. Then, the training data is further divided into training and verification sets in a 0.2 ratio.

Balancing the database occurs after the separation, ensuring that each set (training, testing, verification) has approximately the same number of instances, while the training data remains substantially larger.

Overall, we aim to optimize the model's learning process by addressing data imbalances and ensuring it receives diverse and informative inputs during training.

a) Training without balancing

Functions shown below are responsible for data separation into training and testing (or validation, depending on the input data) and data storage by different models.

Split Function
- Input: Database to be divided.
- Output: Split data into training and testing (or validation) sets.

Logic, Forest, Birch, Classifier, and Tree Functions:
- Input: Separated data.
- Process: Each function passes the data through a specific model (Logistic Regression, Random Forest Classifier, Birch, MLP Classifier, or Decision Tree Classifier) for learning.
- Evaluation: The trained model is evaluated using validation data, and the accuracy of the model is calculated.
- Output: Percentage of accuracy rounded to a thousandth.
- Note: The model is trained without balancing the data to compare accuracy with and without balancing.

Final Results Table:
- The results from each model and training approach (with and without balancing) are added to the "final_results" table for comparison.

b) Training only with balancing essay sets

We tried to implement the SMOTE (Synthetic Minority Over-sampling Technique) method to balance a dataset specifically for an essay set ID. The method involves running the dataset through a function that takes the database and the column name for which additional data points are needed as inputs. The function outputs a new database with added lines to balance the data. Despite applying the SMOTE method, the results have not changed, indicating the need for further balancing efforts to potentially affect future results.

c) Training with balanced domain score

The focus was on balancing domain scores using the SMOTE (Synthetic Minority Over-sampling Technique) method. The aim is to balance each essay set separately. Despite efforts to balance the data, the analysis indicates that the results have not changed. The suggestion is persisting with data balancing, as it may impact the results in the future.

d) Training with balanced domain score and balanced essay set by oversampling

The past database was merged with the updated values obtained after applying the SMOTE (Synthetic Minority Over-sampling Technique) method to essay set 8. This combination of past data and newly balanced data enriches the dataset for essay set 8.
The process will continue by balancing other essay sets using the SMOTE method, ensuring that the datasets are representative and adequately cover the range of responses across different essay sets.

e) Training with balanced domain score and balanced essay set by undersampling

The process involves balancing a database through under sampling, which might improve accuracy but can lead to data loss. Specifically, the under-sampling technique called Cluster Centroids will be employed to reduce the data from essay sets 1, 5, 6, and 7.

IV) Result

Upon completing all the steps, it becomes apparent that the diverse methods used for data balancing have had varying impacts on different models and their outcomes.

- Logistic Regression: Despite different balancing techniques, there were no significant changes observed in accuracy, which remained at 0.71. However, this model did not yield the best results among all models tested.

- Random Forest Classifier: This model demonstrated the highest accuracy, reaching 0.91, making it the top performer among all models tested. Hence, it was selected for testing on the training set.

- Birch: This model showed the poorest results, with accuracy dropping to 0.021 after the first balancing attempt.

- MLP Classifier: Depending on the balancing technique used, MLP Classifier demonstrated varying accuracy levels. The best result was achieved when only essay sets were balanced, although there were still better-performing models.

- Decision Tree Classifier: This model ranked as the second most effective. While the results varied with different balancing methods, accuracy remained consistently high. It was also chosen for testing on the training set.

Based on the analysis, Random Forest Classifier and Decision Tree Classifier were deemed the most suitable models for the dataset, particularly when balancing only essay sets and oversampling data. Therefore, these models will be used for further testing on the training set.

In the last part of the process, the Forest and Tree functions are duplicated and modified slightly. These modified functions are then applied to the training set. Following this, testing is performed, and the results show that learning by the Random Forest Classifier method yields better outcomes compared to the original methods.

V) Conclusion

In this project, the objective was to build a model for predicting the grades of English essays. The process begins with the inclusion of testing features. Data preprocessing involves normalization of the 'domain scores' column to have a single target column and vectorization of the 'essay' column as we known vectorization is particularly crucial in Natural Language Processing (NLP) as it converts textual data into numerical formats suitable for analysis and modeling purposes. Graphical analysis reveals variations in the number of essay sets and imbalanced scores within each set, potentially impeding the model's learning capabilities. To rectify these issues, the model undergoes testing and training, with a focus on data balancing to provide the most informative input to the model.

Initially, we trained our model without balancing the dataset. Then, we trained it again, this time balancing each essay set using the SMOTE method. However, even after applying SMOTE, the results remained unchanged, suggesting the need for further balancing efforts.
We also trained the model with balanced domain scores. This involved balancing domain scores using the SMOTE method for each essay set separately. Despite these balancing efforts, the results did not improve, indicating that continued efforts to balance the data may be necessary for better outcomes.
Another approach involved training the model with balanced domain scores and balanced essay sets achieved through oversampling. We merged the past database with updated values obtained by applying SMOTE to essay set 8. The intention is to extend this balancing technique to other essay sets using SMOTE to enhance the dataset.

In contrast, we trained the model with balanced domain scores and balanced essay sets through undersampling. This entailed balancing the database by reducing data from essay sets 1, 5, 6, and 7 using Cluster Centroids.

Following this, the Forest and Tree functions undergo duplication and slight modification. The adapted functions are subsequently utilized on the training set. Following this, testing is conducted, demonstrating that the Random Forest Classifier method surpasses the original methods in terms of learning performance.

References

https://www.kdnuggets.com/2017/06/7-techniques-handle-imbalanced-data.html

https://machinelearningmastery.com/xgboost-for-imbalanced-classification/

https://www.linkedin.com/pulse/exploring-most-effective-data-augmentation-techniques-zak-goldwasser

https://www.kaggle.com/c/asap-aes

https://en.wikipedia.org/wiki/Flesch%E2%80%93Kincaid_readability_tests

https://readable.com/readability/flesch-reading-ease-flesch-kincaid-grade-level/

https://fr.wikipedia.org/wiki/SMOG_(indice_de_lisibilit%C3%A9)

https://readable.com/readability/smog-index/

https://readable.com/readability/flesch-reading-ease-flesch-kincaid-grade-level/

https://readable.com/readability/new-dale-chall-readability-formula/

https://fr.wikipedia.org/wiki/Indice_Coleman-Liau

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3813439/#:~:text=speech%2Dlanguage%20pathology.-,The%20Measure%20of%20Textual%20Lexical%20Diversity%20(MTLD%3B%20McCarthy%2C%202005,a%20certain%20TTR%20is%20maintained.

https://en.wikipedia.org/wiki/Parse_tree