Project 3: Movie Review Sentiment Analysis

Fall 2024

Contents

Project Datasets

Movie Review Data

The data set for this project has 50,000 rows, each corresponding to an IMDB movie review, and 1539 columns. The first three columns are

- "**id**": A unique identification number assigned to each review.
- "**sentiment**": A binary label where 0 indicates negative sentiment, and 1 indicates positive sentiment.
- "**review**": The actual text of the movie review.

The remaining 1,536 columns are the 1,536-dimensional embedding generated by OpenAI's `text-embedding-3-large` model. You can learn more about OpenAI embeddings [here].

A portion of this dataset was originally used in a Kaggle competition titled [Bag of Words Meets Bags of Popcorn]. For further insights and sample code, refer to the Kaggle competition discussion.

Training/Test Splits

For this project, you are provided with five different training/test splits. Each split represents a random division of the original dataset into two halves, with 25,000 samples allocated to training and 25,000 to testing. Each split contains three files:

- `train.csv`: The training data with 1,539 columns.
- `test.csv`: The test data with 1,538 columns (excluding the "sentiment" column).
- `test_y.csv`: The test labels file with two columns, "id" and "sentiment".

The zipped dataset is approximately 4 GB. You can access the UIUC Box share link via Coursera/Canvas in the Assignment section for Project 4.

Project Objectives

There are two main objectives for Project 4.

1. Build a Binary Classification Model

The first objective is to construct a binary classification model to predict the sentiment of a movie review.

The evaluation metric for this project is the **Area Under the Curve (AUC)** on the test data. Your goal is to achieve an AUC score of at least 0.986 across all five test data splits.

2. Interpretability Analysis

Using **split 1** and the corresponding trained model, implement an interpretability approach to identify which parts of each review have an impact on the sentiment prediction. Apply your method to **5** randomly selected **positive** reviews and **5** randomly selected **negative** reviews from the **split 1 test** data.

Set a random seed before selecting these 10 reviews (the seed does not need to relate to students' UINs).

Provide visualizations (such as highlighted text) that show the key parts of a review contributing to the sentiment prediction. Discuss the effectiveness and limitations of the interpretability approach you chose.

Submission Guidelines

Please provide the following **two** items on Coursera/Canvas for your project submission:

- **Code**: Your R/Python script should be in a singular file named either mymain.R or mymain.py. This script should:
  - Accept `train.csv` and `test.csv` as inputs.
  - Generate one file named `mysubmission.csv` based on the specified format (described below).
  - Important: Do not try to access `test_y.csv` in your code.
- **R/Python Markdown/Notebook File (in HTML)** or **its link**: This file should contain the following two sections:
  - **Section 1**:
    - Discuss the technical details of the sentiment classification model, including data preprocessing and other key aspects of your model implementation. Your explanation should be detailed enough for your PSL classmates to replicate your results.
    - Report the AUC of your predictions on each of the 5 test datasets (refer to the evaluation metric described above), the execution time of your code, and the specifications of the computer system used (e.g., Macbook Pro, 2.53 GHz, 4GB memory, or AWS t2.large) for each of the 5 splits.
  - **Section 2**: Provide a detailed explanation of your interpretability approach.
    - Include all the necessary code within this file to ensure reproducibility of your results.
    - **Do not** include code to train the classification model on the **split 1** training data. Instead, save the trained model online (e.g., on your GitHub account) and load the model from that source.
    - For your interpretability approach, you may use word or sentence embeddings. Students are **not allowed to use OpenAI embeddings** for this purpose. You may use other free embedding models, such as BERT. Similar to the trained classification model, store these embeddings online so that they can be accessed directly from the provided web address when running your code.

Code Evaluation

Execution:

- For R: We'll run `source(mymain.R)` in RStudio from a clean environment (meaning, no pre-loaded libraries).
- For Python: We'll execute python `mymain.py` from the command line.

We will execute the command in a directory that contains only **three** files:

- `train.csv`
- `test.csv`

After executing your code, we should find a CSV file named `mysubmission.csv` in the same directory. This CSV file should include a header, and its entries should be separated by commas, structured as follows:

```
id,     prob
47604,  0.940001011154441
36450,  0.584891891011812
30088,  0.499236341444505
18416,  0.00687786009135037
```