Mario Hysa
Professor Galletti

Predicting Amazon Movie Review Ratings Using Sentiment Analysis and Logistic Regression

The objective of this project was to develop a predictive model to accurately determine the star ratings (scores ranging from 1 to 5) associated with user reviews from Amazon Movie Reviews. By leveraging the metadata and the textual content of the reviews, the aim was to extract meaningful patterns and features that could enhance the model's predictive capabilities.

To ensure the model could learn from as much data as possible, the entire training dataset was utilized. A random seed was set for reproducibility, and the data was shuffled to prevent any order biases. An initial step involved verifying the presence of the 'Text' column in the training data, which contains the actual review text. It was essential to ensure that all entries in this column were strings to facilitate text processing. Any missing values were filled with empty strings to avoid errors during text analysis.

Understanding the sentiment expressed in the reviews was essential for predicting the star ratings. The VADER (Valence Aware Dictionary and sEntiment Reasoner) sentiment analyzer was employed due to its effectiveness in handling social media texts and its ability to capture nuances like sarcasm and emphasis. VADER provides four sentiment metrics: positive, negative, neutral, and compound scores.

Given the large size of the dataset, processing each review sequentially would be computationally intensive. To address this, the `pandarallel` library was utilized to parallelize the sentiment analysis across multiple CPU cores. This significantly reduced the computation time, making it feasible to process millions of reviews efficiently. Utilizing `pandarallel` for parallel processing allowed for efficient sentiment analysis on a large dataset, enhancing performance without sacrificing accuracy.

To capture various aspects of the reviews that could influence the star ratings, several features were engineered. First, textual features were derived from the review content. The character length of each review was calculated, based on the idea that longer reviews might convey more detailed opinions. Counts of exclamation marks and question marks were included to capture emotional emphasis or uncertainty expressed by the reviewers. Capitalization metrics were also considered, including the ratio of uppercase letters to the total length of the text, indicating the extent of emphasis through capitalization, and the count of fully capitalized words, which could signify shouting or strong emotions. These textual features were based on the hypothesis that reviewers use certain stylistic elements to convey strong feelings, which could correlate with extreme ratings, either high or low.

Sentiment mapping was another crucial feature engineering step. The compound sentiment score from VADER, which ranges from -1 to 1, was linearly transformed to a scale of 1 to 5 to align with the star ratings. This mapping provided a direct comparison between the sentiment and the expected rating. Additionally, the compound sentiment score was discretized into categorical levels from 1 to 5, based on predefined thresholds. This categorization helped in capturing the sentiment intensity in a way that could be more interpretable by the model. The underlying assumption was that by aligning sentiment scores with the rating scale, the model could better learn the relationship between the expressed sentiment and the star rating.

Helpfulness metrics were also incorporated into the feature set. The helpfulness ratio was calculated as the ratio of the number of users who found the review helpful to the total number of users who indicated whether they found the review helpful. This metric reflects the perceived quality or usefulness of the review by other users. Interaction features were created by combining sentiment scores with other metrics. For example, the sentiment-helpfulness interaction was defined as the product of the compound sentiment score and the helpfulness ratio. This feature captures the idea that a review that is both sentimentally strong and considered helpful might have a stronger impact on the rating. Similarly, the sentiment-text length interaction was calculated as the product of the compound sentiment score and the text length, reflecting that longer, sentiment-rich reviews might influence ratings differently than shorter ones. I noticed that reviews deemed helpful by others and exhibiting strong sentiments were likely more influential in predicting the star rating.

Additionally, user and product average scores were computed. The average rating for each product was calculated based on the available training data, accounting for the overall reception of a product. Similarly, the average rating given by each user was computed to capture individual rating tendencies or biases. The assumption was that certain products consistently receive higher or lower ratings due to their quality, and some users may have a propensity to rate products higher or lower on average. Incorporating these averages could help the model adjust predictions accordingly.

To ensure robustness, any missing values in the engineered features were filled with default values that made sense in the context. Numeric features like sentiment scores and counts were filled with neutral or zero values. The average scores for products and users were filled with the overall average score from the training data if specific averages were unavailable. The thought process behind this approach was that filling missing values appropriately prevents the introduction of bias or errors during model training and ensures that all data points are processed consistently.

To capture the semantic content of the reviews beyond what sentiment analysis could provide, the textual data was transformed using the Term Frequency-Inverse Document Frequency (TF-IDF) vectorization method. This technique converts textual data into numerical feature vectors that reflect the importance of words in a document relative to the entire dataset. To reduce computational complexity while retaining significant words, the maximum number of features was limited to 5,000. Common English stop words were excluded to focus on words that carry more meaningful information. This approach optimized the balance between computational efficiency and the complexity of the textual information.

To ensure that all features contributed equally to the model and to improve convergence, numerical features were standardized using z-score normalization. This process scaled the features to have a mean of zero and a standard deviation of one. The scaled numerical features and the TF-IDF vectors were then combined into a single feature matrix, allowing the model to consider both the engineered numerical features and the textual content simultaneously.

A logistic regression model was chosen for its simplicity, interpretability, and effectiveness in multiclass classification problems. The 'newton-cg' solver was selected because it is well-suited for multinomial logistic regression and performs efficiently on large datasets. The regularization parameter (C) was set to 0.5 to balance the trade-off between overfitting and underfitting. Additionally, the maximum number of iterations was increased to 3,000 to ensure that the optimization algorithm had sufficient time to converge. Adjusting hyperparameters based on preliminary results and ensuring convergence was important for achieving optimal model performance.

To evaluate the model's ability to generalize to unseen data, 5-fold cross-validation was performed. In this method, the dataset was split into five subsets, and the model was trained and evaluated five times, each time using a different subset as the validation set and the remaining subsets as the training set. Accuracy was used as the evaluation metric, aligning with the competition's scoring method. From my tests, I saw that consistent cross-validation accuracy scores indicated that the model was not overfitting and had good generalization capabilities.

The test data required the same preprocessing steps as the training data. However, the 'Text' column was not directly available in the test dataset. To address this, the test data was merged with the training data on the 'Id' column to obtain the corresponding 'Text' and other metadata. Any missing 'Text' entries after the merge were filled with empty strings to prevent errors during feature extraction. The assumption was that the 'Id's in the test data corresponded to reviews in the training data. Merging allowed for the reuse of precomputed features where possible.

The same feature engineering functions used for the training data were applied to the test data to ensure consistency. Sentiment analysis was conducted using VADER to calculate the sentiment scores for the test reviews. All the engineered features, including textual features, interaction terms, and average scores, were computed. The TF-IDF vectorizer fitted on the training data was used to transform the test review texts, ensuring that the feature space was identical to that of the training data. Reusing the same vectorizer and feature engineering process maintained consistency between the training and test datasets, which is crucial for accurate predictions.

With the features prepared, the model made predictions on the test data. The numerical features were standardized using the same method as the training data. The scaled numerical features and the TF-IDF vectors were combined into a single feature matrix. The logistic regression model then predicted the star ratings for each review in the test data. The predicted scores were rounded and clipped to ensure they fell within the valid range of 1 to 5. The assumption was that clipping and rounding the predicted scores ensured that the submission met the competition's requirements and reflected realistic star ratings.

Several patterns were noticed during the project. Reviews with strong sentiments and higher helpfulness ratios tended to have more extreme star ratings. Certain textual elements, such as exclamation marks and capitalization, were indicative of the reviewer's emotional intensity and correlated with the rating. Users and products exhibited consistent rating behaviors, which could be leveraged to adjust predictions.

Works Cited

Hutto, C. J., & Gilbert, E. (2014). *VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text.* Proceedings of the Eighth International Conference on Weblogs and Social Media. Available at https://ojs.aaai.org/index.php/ICWSM/article/view/14550. Accessed October 2024.

Python Software Foundation. *NumPy.* https://numpy.org/. Accessed October 2024.

Reber, T. (2018). *pandarallel: Pandas Parallelization Library.* GitHub Repository, https://github.com/nalepae/pandarallel. Accessed October 2024.

The NLTK Project. *Natural Language Toolkit.* https://www.nltk.org/. Accessed October 2024..