Midterm Algorithm Implementation and Performance

Introduction

In this project, I developed a machine learning algorithm to predict review scores based on both textual and numerical data from customer reviews. I used Logistic Regression along with several preprocessing and feature extraction techniques to improve the model's performance. This write-up outlines the steps I took, the reasoning behind my decisions, and some tricks I used to enhance the model's effectiveness.

Data Preprocessing and Feature Engineering

Initial Data Handling

The dataset comprised two main parts: the training and testing sets. My first step was to address missing values and prepare the data for feature extraction. I calculated the Helpfulness score by dividing the HelpfulnessNumerator by HelpfulnessDenominator. If there were missing values, they were replaced with zero to maintain consistency.

Textual Data Cleaning

I performed cleaning on the textual data, particularly focusing on the Text field (which was ultimately used in the final algorithm). Here are the main cleaning steps:

- **Lowercasing:** I converted all text to lowercase to ensure uniformity.
- **Punctuation Removal:** I removed special characters using regular expressions to cut down on noise in the data.
- **Stop Words Removal:** Common stop words were filtered out using NLTK's predefined stop words list. This step allowed for the focus to be on the more informative words in the reviews.

The remove_stop_words function was important to ensure that only relevant words contributed to the model.

Feature Extraction

I extracted several key features:

1. **Helpfulness Ratio:** This numerical feature gives insights into the quality of a review. This feature was the only numerical feature used since the accuracy was most affected by this.
2. **Review (Text) Length:** This numerical feature is the length of words of each Text review. This field was not used in the final model since it did not improve the accuracy.
3. **Short Text:** This is the Text field that was cleaned as noted in the section above.
4. **Short Summary:** This is the Summary field that was cleaned as in the section above. This field was not used in the final model since the combination of ShortText and Short Summary lowered the accuracy.

Latent Semantic Analysis (LSA)

I utilized TfidfVectorizer to convert the cleaned textual data into a numerical format suitable for machine learning. By limiting the maximum features to 5000, I concentrated on the most informative terms while keeping dimensionality in check. To further reduce dimensionality and extract underlying patterns in the data, I employed Truncated Singular Value Decomposition (SVD) as part of LSA. This technique helped the model identify relevant information in the text data, making it more effective.

Numerical Features Standardization
The numerical feature, the helpfulness ratio, was standardized using StandardScaler. This ensures that all features contribute equally to the model.

Model Training and Evaluation

Model Selection
I chose Logistic Regression for its interpretability, efficiency, and effectiveness in handling both binary and multi-class classification tasks. The model was trained using the combined features from LSA and standardized numerical data.

Hyperparameter Tuning
While I set the maximum number of iterations for convergence, I recognize that further hyperparameter tuning (like adjusting regularization strength) could be beneficial in future iterations to optimize performance.

Predictions
After training the model, I used it to predict scores on the test set, which allowed for evaluating performance.

Performance Evaluation

To gauge the model's effectiveness, I compared the predicted scores against the actual scores from the test set. Although I haven't included specific values in this write-up, a thorough evaluation is essential for understanding the model's capabilities.

Special Tricks and Insights

1. **Feature Engineering Insight:** During my initial exploration, I noticed that removing stop words and using LSA significantly improved the model's ability to interpret text. This highlighted the importance of thoughtful feature selection in machine learning.
2. **Dimensionality Reduction:** Utilizing SVD not only reduced computational overhead but also enhanced the model's performance by allowing it to concentrate on the most critical features from the text. I found that increasing the number of components also increased accuracy, but also took a significant amount of time to process.
3. **Combination of Features:** Integrating both textual and numerical features enabled the model to leverage different types of information. This allowed for a more accurate prediction.

4. **Handling Missing Values:** It was important to manage missing values effectively, so I opted to replace them with zeros or empty strings where appropriate, preventing potential issues during training. The other option was to remove the rows with missing values, but that would also remove data in other columns that could be important in the prediction.
5. **Monitoring Overfitting:** By splitting the data into training and test sets and maintaining a reasonable test size, I reduced the risk of overfitting.

Assumptions Made

Throughout this project, I made several assumptions:

- **Data Quality:** I assumed that the data was clean enough and representative of the underlying phenomenon I aimed to model.
- **Independence of Features:** The model assumes that features are independent of each other, which may not always be the case in practice.
- **Relevance of Helpfulness and Text:** I assumed that the helpfulness score and the text review is a valid indicator of star ratings and thus an appropriate target variable.

Conclusion

The final model combines text preprocessing, feature extraction techniques, and a classification algorithm. While the implementation showed promising results, I believe there is still room for improvement through further model tuning and feature engineering. The insights I gained throughout this process highlight the importance of iterative experimentation and refinement in developing effective machine learning models.