CS506 midterm

Oct. 28th 2024

Ruiyang Cao

Introduction:

The goal of this project was to predict review scores using Amazon movie review data. The approach involved a combination of feature engineering, data preprocessing, and model selection. This document describes the final algorithm implemented, the methods used to enhance model performance, and the thought processes behind feature and model choices. The focus was on optimizing the predictive accuracy within computational constraints and leveraging observed patterns to improve predictions.

Data Loading and Initial Exploration

The process started with loading training and testing datasets, containing user reviews and their associated helpfulness metrics. Preliminary exploration of the training data revealed an imbalance in the distribution of review scores, with certain ratings occurring more frequently. This prompted us to consider techniques that would balance the impact of different features to avoid bias toward common ratings.

Feature Engineering

Helpfulness Ratio

A new feature, Helpfulness, was created to quantify user engagement with reviews by dividing HelpfulnessNumerator by HelpfulnessDenominator:
Helpfulness = HelpfulnessNumerator / HelpfulnessDenominator

Assumption: Reviews deemed helpful by other users might correlate with specific ratings, as they may indicate more detailed or useful feedback.

Textual Features

Top keywords were extracted from high-helpfulness reviews using CountVectorizer, which provided binary features indicating the presence of each keyword in the review. We also manually added specific keywords like "love," "recommend," and "great" based on their association with positive ratings.

Assumption: Certain keywords or phrases are likely indicators of sentiment intensity, influencing review scores.

Review Length

Here introduced two features to capture the characteristics of the review text:

- **word_count**: Total word count in the review.
- **avg_word_length**: Average word length in the review text.

Assumption: Longer or more detailed reviews might indicate a stronger opinion, potentially correlating with certain ratings.

Data Preprocessing

To handle missing values and ensure data consistency:

- **Missing Values**: Helpfulness values where the denominator was zero were filled with zero.
- **Data Consistency**: Ensured alignment of all features between the training and testing sets.

Model Creation

Base Models

We used two base models for this project:

- **RandomForestClassifier**: Chosen for its ability to handle non-linear relationships and high-dimensional data.
    - **Advantages**: High interpretability, resilience to overfitting due to ensemble averaging.
- **LogisticRegression**: A linear model included to reduce overfitting and capture linear trends in the data.
    - **Parameters**: Increased max_iter to ensure convergence.

Meta-Model (Stacking with XGBoost)

An XGBoostClassifier was used as the meta-model, with stacking features derived from the mean and max probabilities of each base model. A grid search optimized the hyperparameters (n_estimators, learning_rate, max_depth, and reg_lambda) to find the best-performing combination.

Model Evaluation

Accuracy Score

The model's accuracy was evaluated on the validation set, measuring the proportion of correct predictions over the total predictions made.

Confusion Matrix

To analyze the performance across different classes, we used a confusion matrix:

- **True Positives (TP)**: Correctly predicted scores.
- **False Positives (FP)** and **False Negatives (FN)**: Incorrectly predicted scores that were either over- or under-estimated.

Patterns Noticed and Utilized

1. **Helpfulness Ratio Impact**: Reviews with high helpfulness often corresponded to more extreme ratings, and including this feature improved accuracy for these cases.
2. **Keyword Importance**: Common keywords helped the model capture sentiment, enhancing accuracy for reviews with strong positive or negative opinions.
3. **Temporal Patterns**: By including Time as a feature, we accounted for any evolving trends in review patterns over time.

Assumptions Made

- **Helpfulness Ratio Validity**: Assumed that reviews with HelpfulnessDenominator of zero had meaningful ratios; filled NaNs with zero to address division errors.
- **Keyword Relevance**: Assumed that certain keywords were direct indicators of sentiment, justifying their binary encoding.
- **Feature Independence**: Assumed that selected features contribute independently to predictions, supporting their inclusion without multicollinearity checks.

Conclusion

The final model effectively predicts Amazon review scores by leveraging feature engineering, base and meta-model stacking, and systematic hyperparameter tuning. Key enhancements included custom feature extraction, keyword-based sentiment capture, and thorough validation of model performance. Future work could explore additional text-based features, such as TF-IDF vectorization, to further capture nuanced sentiment in reviews.