CS506 Midterm Fall 2024

1. Introduction

In this project, our task is to predict the user's star rating based on the text and metadata of Amazon movie reviews. Star rating is a direct reflection of user satisfaction with the movie, and accurate prediction can play an important role in recommendation systems and market analysis. However, this task is challenging due to the large data size, class imbalance, and sparsity of text data.

The project requirement is not to use neural networks or deep learning, so I chose classic machine learning methods to complete the task. Our main goal is to achieve good prediction accuracy by optimizing feature engineering and model selection under limited computing resources.

2. Data exploration and analysis

Each review contains the following fields: ProductId (product ID), UserId (user ID), HelpfulnessNumerator (helpful rating), HelpfulnessDenominator (total number of ratings), Score (star rating), Time (time stamp), Summary (review summary)), Text (comment text) and Id (unique identifier).

Through preliminary analysis, I discovered that there is a class imbalance problem in the data. In particular, the number of 5-star and 4-star reviews far outnumbers other ratings. In addition, some reviews lack rating information, and these missing values need to be handled in prediction.

3. Feature Engineering

To extract meaningful features from the review text, I preprocessed the data and used the following feature extraction methods.

Text Preprocessing

I used the NLTK toolkit to perform standard preprocessing of the text, including:

Lowercase: Unify the text format.

Remove HTML tags and URLs: Clean up the noise data in the review.

Remove punctuation and numbers: Reduce the interference of irrelevant content on the model.

Stop word removal: Remove meaningless words such as "the" and "and".

Lemmatization: Reduce the word to its basic form, such as converting "running" to "run".

Feature Extraction

I chose the TF-IDF (Term Frequency-Inverse Document Frequency) vectorization method to extract text features:

TF-IDF vectorization: Convert the text into a numerical feature vector that captures the importance of each word in the review.

n-grams: I used ngram_range=(1,2) to consider words and adjacent word combinations to capture the pattern of common phrases.

Maximum feature number limit: Set max_features=5000 to select the 5000 most representative words to reduce the amount of calculation.

In addition, I also added the length feature of the review as an additional numerical input to further distinguish different star ratings.

4. Model selection and optimization

Choosing the Naive Bayes classifier

In text classification tasks, the Naive Bayes classifier is a classic and efficient algorithm, especially for discrete features (such as word frequency). I chose Multinomial Naive Bayes because it performs well for text classification, has fast training and prediction speeds, and is suitable for large-scale data.

Hyperparameter tuning

To improve model performance, I used GridSearchCV for hyperparameter tuning and optimized the alpha parameter (Laplace smoothing coefficient). Experimental results show that the model performs best when alpha=0.5.

Model evaluation

I evaluated the performance of the model on the validation set, using indicators such as accuracy, precision, recall, and F1 score. The results show that the model performs well in predicting 4-star and 5-star reviews, but the accuracy of 1-star and 2-star reviews is relatively low.

5. Special techniques and improvements

Handling category imbalance

Since 5-star reviews account for a larger proportion, I used class_weight='balanced' when training the model to make the model pay more attention to a few categories. In addition, I reduced the number of 5-star reviews in the training set through downsampling to alleviate the class imbalance problem.

Reduce feature dimension

To optimize computational efficiency, I used the chi-square test for feature selection, retaining the 3000 features most relevant to the target variable. This reduces the calculation time of the model and improves the accuracy of the model.

Model integration

In order to further improve performance, I tried the ensemble learning method. We combine Naive Bayes, Support Vector Machine (SVM) and Random Forest models, and use voting classifiers to integrate the prediction results of multiple models, which significantly improves the stability of the model.

6. Conclusion and outlook

In this project, classic machine learning methods were successfully used to accurately predict the star ratings of Amazon movie reviews. Through in-depth analysis of text and feature engineering, the predictive performance of the model is improved. The final ensemble model outperformed the single model on the test set and achieved higher accuracy.