CS506 Midterm Fall 2024

Yuzhe Wu

1. Overview

The goal of this competition is to predict the star rating associated with user reviews from Amazon Movie Reviews using the available features. Finally, I use LSA and n-gram to extract feature and apply logistic regression model to achieve my highest score.

2. Initial try

After reading the goal of the competition, I found that the amount of 5 star are much higher than 1 - 4 stars. Maybe most reviews will say some good comment. So I think it is a good feature that may contain some key phrase to make the review. With this in mind, I began by analyze the Text and Summary features for each review. I then applied Latent Semantic Analysis (LSA), hoping to capture patterns related to positive sentiment or expressions that could serve as useful features for the model. I expected this could provide valuable insights into the text, especially given the imbalance in star ratings.

3. Model Selection and Initial Testing

Initially, I tested a K-Nearest Neighbors (KNN) model, but the performance was suboptimal. I found that KNN struggled to perform well in the high-dimensional feature space created by the text data. To address this, I switched to a Random Forest model, which is known for its robustness in handling high-dimensional data. Random Forest performed notably better, achieving an accuracy of around 0.53. This provided a good start for me.

In the next stage, I experimented with dimensionality reduction using Singular Value Decomposition (SVD). The goal was to reduce the noise and potentially improve the model's generalizability by reducing the number of features. However, after testing, I observed that dimensionality reduction didn't lead to significant improvement in model accuracy.

I tested K-means++ to create clusters based on text features, intending to see if the clustered groups might represent different rating levels. However, K-means++ is an unsupervised algorithm and isn't designed for label-based prediction tasks. Due to this limitation and the need for distancebased calculations in high-dimensional space, K-means++ did not perform well for this classification task.

Linear Regression: I also tried a Linear Regression model, hypothesizing that it might capture underlying linear relationships between features and ratings. However, Linear Regression struggled with the categorical nature of the ratings, as it assumes a continuous output rather than discrete classes. The resulting predictions were not well-suited for this task, given the complex, non-linear patterns in the data.

4. Feature Expansion

After this, I decided to add several numerical features to provide the model with additional context:

1. HelpfulnessRatio: Defined as HelpfulnessNumerator / (HelpfulnessDenominator + 1), this feature represents the helpfulness rating.
2. ReviewYear: Extracted from the timestamp, this feature indicates when the review was posted.
3. TextLength and SummaryLength: These represent the lengths of the review text and summary, providing insight into the detail level of each review.

Additionally, I included n-gram features to capture word pairings within reviews, expecting that common word pairs (such as “highly recommended” or “not good”) could enhance the model’s predictive power by capturing local context.

5. Experimentation with Alternative Models

After establishing this feature set, I tested various models to improve performance:

1. Linear Regression: I initially considered Linear Regression to model the relationship between features and ratings. However, due to the large number of features, Linear Regression had a high computational cost and struggled to converge. When results were obtained, the accuracy gain was minimal, so I ultimately decided to abandon Linear Regression in favor of more feasible models.

2. Logistic Regression: I then tried Logistic Regression, which managed the high-dimensional data more efficiently and achieved an accuracy of around 0.65. I used grid search for hyperparameter tuning, but unfortunately, this did not yield a significant improvement in performance. However, given its stable performance and manageable computation time, Logistic Regression showed potential as a strong baseline model.

3. XGBoost: Compared to previous models, XGBoost offers a strong performance than previous models, particularly in handling high-dimensional, complex feature spaces, making it a strong candidate for achieving higher accuracy in supervised learning tasksHowever, in this context, XGBoost yielded an accuracy of only 0.50, which was below expectations. After reviewing feature importance and tuning, I concluded that XGBoost was not effectively capturing the patterns in this dataset and decided to revert to Logistic Regression.

6. Final Model and Results

Ultimately, Logistic Regression emerged as the best model for this task. Despite grid search tuning not providing substantial gains, Logistic Regression demonstrated consistent performance with a final accuracy of approximately 0.65. This model effectively balanced computational efficiency and predictive accuracy, making it the optimal choice given the feature space and dataset.