Dylan Mohsen

Professor Galetti

CS 506

28 October 2024

Overview

The objective of this midterm was to create a predictive model to classify review scores using a dataset of customer reviews. I used a Random Forest Classifier, and additional engineered features to improve prediction accuracy and model itself.

Dataset Description

The dataset comprises customer reviews in two CSV files: train.csv and test.csv, containing various review metrics. Each row represents an individual review with columns for helpfulness ratings, review score, and a timestamp.

After loading the dataset, I performed preliminary data visualization to understand the distribution of scores, revealing imbalances in review scores. This prompted adjustments during model training and evaluation. The goal was to have the models predict scores.

Feature Engineering and Selection

To improve model performance, I incorporated engineered features based on observed patterns in the dataset:

- Helpfulness Ratio: Calculated as the ratio of HelpfulnessNumerator to HelpfulnessDenominator. This feature quantifies how helpful a review is, relative to the number of people who found it helpful, providing insight into review credibility. Replaced 0 denominators with 1 to prevent division errors, filling missing values with 0 for seamless integration.
- Review Age: Derived from the timestamp (Time), representing the number of days since the review was posted. This feature was inspired by noticing that older reviews often receive different scores compared to recent ones, potentially capturing biases

in review scores. Subtracted each timestamp from the maximum date in the dataset, standardizing the Review_Age feature across reviews.
- Score Encoding for Training and Submission Sets: To ensure consistency across training and testing, I extracted and merged features on the unique Id identifier. This approach minimized any discrepancies in feature representation across the datasets.

Selected Features

I focused on five primary features: HelpfulnessNumerator, HelpfulnessDenominator, Time, Helpfulness, and Review_Age. These features were selected based on their relevance to review quality and score prediction.

Model Implementation and Training

The model is a Random Forest Classifie**r** with 100 estimators and a max depth of 10. The choice of Random Forest was due to its robustness against overfitting and its ability to handle the complexities of feature interactions without requiring extensive tuning.

For using the Random Forest Classifier, I have used it in previous assignments for other classes so I was already aware of its functionality and how to use it:

Steps Taken

1. Data Splitting: The dataset was split into training and test sets with a 75%-25% split to assess model performance on unseen data.
2. Training the Model: I utilized a Random Forest Classifier with parameters tuned to balance model complexity and computational efficiency.
3. Predictive Modeling: After fitting the model, predictions were generated on the test set, followed by the creation of a submission file based on the final model predictions.

Performance Optimization and Patterns

Key Observations and Adjustments

- Handling Missing Data in Helpfulness: Analyzing the Helpfulness feature revealed that reviews with a HelpfulnessDenominator of 0 skewed the ratio calculation. To

address this, I replaced 0 denominators with 1, filling NaN values to prevent computational errors during training. This adjustment led to a more stable model performance by reducing the noise in the Helpfulness metric.

- Influence of Review Age: The Review_Age feature consistently demonstrated that older reviews had lower scores, potentially due to changing user expectations or updated product versions. Including this feature as a numeric variable led to an increase in accuracy, capturing this underlying trend effectively.
- Model Selection: I tried doing it with KNN and alternative classifiers and found that Random Forest provided the highest accuracy and computational efficiency. It works well because it combines multiple decision trees, which helps it understand different types of patterns in the data.
- Hyperparameter Tuning: The max depth of the trees was restricted to 10 to reduce overfitting. Further tuning was achieved by adjusting the number of estimators and applying cross-validation to validate model stability.

Conclusion

In this project, I developed a model for predicting review scores by focusing on key features and optimizing for both accuracy and efficiency. By engineering meaningful features like Helpfulness and Review_Age and carefully handling missing data, I enhanced the model's ability to capture valuable patterns in the review data. The Random Forest Classifier stood out as the best fit, offering reliable accuracy without requiring complex adjustments. These insights and methods could serve as a strong foundation for further refinement, such as incorporating more nuanced features or exploring additional ensemble methods for even greater accuracy.