CSE 158 Assignment 2: Helpfulness Prediction and Sentiment Analysis for Amazon Fine Foods Reviews

Abstract

This report gives insight into various analyses of Amazon Fine Foods Reviews datasets. There are two notable tasks surrounding our report. The former is predicting the helpfulness of a review given the dataset's other features. The latter is a sentiment analysis of the dataset where we'll section the dataset's given ratings into either positive or negative reviews. Other data explorations will also be noted about any connections to the given tasks.

1 Introduction

Online reviews have become an important quality of life, and consumer choices vary regarding how reviews can shape and narrate an idea or certainly a product. Platforms like Amazon have outputted highly user-generated information that can be leveraged for textural data analysis. From a psychological standpoint or an attempt much so, reviews may serve as valuable experiences that inherently construe as impactful whether as a desire for validation to some extent a bias of what feedback is perceived. Languages used in reviews can be processed into insight into how text can translate into emotions; semantic analysis explores this. Various models and algorithms such as logistic regression to metrics like F1 scores can validate a threshold of useful predictions when handling complex lexicons. Exploring these data is crucial to freely explore insights that may help a business understand its customers or reveal uncanny nuances that could occur.

1.1 Dataset and Exploration

The data used for this study consists of Amazon Fine Foods from Oct 1999 to Oct 2012, ranging from dog treats to candy. The dataset caught our attention due to the longevity span of the data with over a decade of textual data up until 2012 with ~560,000 reviews. The reviews include in this data format a productId, userId, profileName, helpfulness score is the fraction of users who found helpful, rating score from 1-5, the timestamp of the review, the summary, and the extended text for a review. The exact format specifications are:
product/productId, review/userId, review/profileName, review/helpfulness, review/score, review/time, review/summary, and review/text. To add on before the exploring, there were instances where we had to data clean such that there were malformed lines causing issues to parse. Those lines were effectively skipped and the sample size was miniscule. The dataset procured to explore was still over 500,000 reviews.

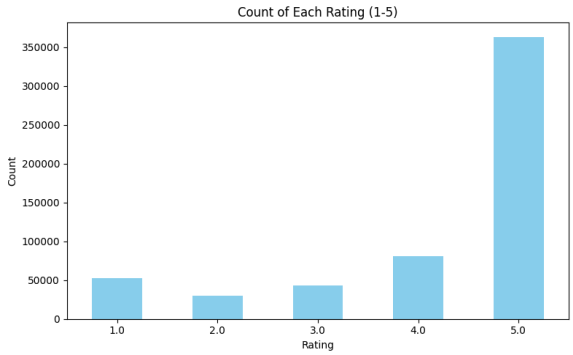

Figure 1.1.1: Rating Distribution

This section introduces the process of analyzing key attributes before training a model. The distribution for the count of each rating from 1-5 shows a strong positive skew. The unimodal distribution indicates a large volume of bias for 5-star ratings leading to inherent biases that can raise concerns and be acknowledged. Some inferences are that users with positive

experiences are likelier to give feedback. A smaller negative review size could implicate a scarcity of early-stage online reviews from this period. Overall, there is a selection bias with positive reviews and a less common practice for negative feedback where other factors like complaints are shared elsewhere instead of a review.

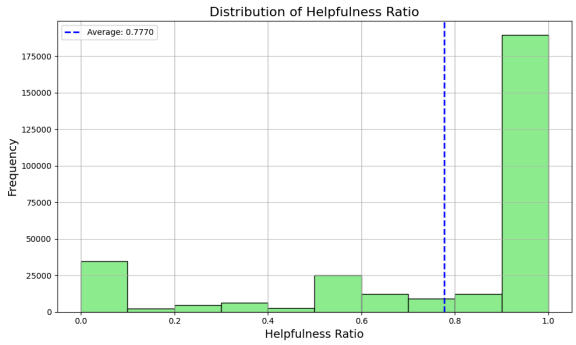

Figure 1.1.2: Helpfulness Ratio Distribution

Leading up to another distribution. A unique trait of this data is the helpfulness measurement. The distribution depicts the ratios to where a fraction of users who found the review helpful or not helpful. A max score is a ratio of 1.0 where all users rated (yes or no for helpful). The skew leads to some similarities to the previous distribution. The average helpfulness ratio follows 0.7770 indicated at the dashed line. Possible interpretations include reviews likely receiving few ratings therefore if there were only a single vote the ratio would default to a ratio of 1.0. Another option is the quality of the information whether from the text or summary feature could be high which appeals to other users to rate them highly where negative reviews are scarce.

2 Helpfulness Prediction

The data shows an interesting choice for a predictive task which we chose to predict whether a review is likely helpful. Noting the skewed distribution and potential sparsity for helpfulness ratings this makes the landscape difficult to navigate with common predictive approaches however, there might be nuances to take an opportunity to still follow up with what to find despite the challenging distribution. Literature-wise, the Fine Foods Reviews dataset was also used for sentiment analysis and recommendation. The data was set as one of many for exploring product recommendations to consumers and also understanding how experience over time affects a review. These are example features to measure when constructing a model for a given prediction task when that dataset was originally applied (latent-factor models for instance).

2.1 Model

To connect back with our task, we employed two key metrics: Mean Squared Error (MSE) and R-squared. MSE provides the difference of the average squares between the actual ratio and the predicted value ratios to gauge assessing how other features could benefit. R^2 helps quantify the proportion of variance to the observed ratios and measures the quality of the regression model. To link up, our starting model employed uses linear regression. Linear regression forms the relationship of the independent variables i.e. the features and a dependent variable or target through a linear equation. The model can be set as a baseline for now. We set the helpfulness ratio as the target variable where we denote the helpful votes divided by the total votes. We then set the two features we believe to have an impact which were the review length of the review text and the term frequency-inverse document frequency (TF-IDF) of measuring how important certain words are and in this case, the top 100 most “significant” terms were added. In terms of text mining and the meaning of significance of a word of this dataset spanning over a decade, we incorporated TF-IDF by normal conventions checking the term frequency in a single document and then measuring the distinctness

across all documents. Adding these for the features brought our results intended on the first-second column below where MSE is around 0.1986 where lower is better and R-squared around 0.0709 where only roughly 7 percent does the model explain the variance in the data.

Model	Linear Regression	K-Nearest Neighbors	LightGBM	Random Forest Regressor
MSE	0.1986	0.0702	0.0495	0.0436
R^2	0.0709	0.6714	0.7686	0.7960

Figure 2.1.1

We can then trial solutions to test a variety of features and modeling strategies to beat this baseline including models pertaining outside the course. The process was conductive and the results following the figure were interesting. We evaluated multiple models including non-linear and linear. We considered drastically improving it during tree-based models involving gradient boosting including XGBoost and LightGBM. The former had a contingency with overconsumption of the ram which crashed the session when running the to operate given the size of the preprocessed dataset. We proceeded with LightGBM for gradient boosting machines which was known for memory efficiency compared to the former. The setup was preprocessing using a TF-IDF vectorizer to convert review texts into numerical features and then limit it to save memory. We then set up the features similar to the baseline and did the train-test split with 80 percent train and 20 percent testing to evaluate performance. The results drastically improved compared to the baseline. Continuing the exhaustion, we considered K-nearest neighbors where the model does not conventionally train a model and makes predictions based on the distance of a data point and how similar it is but the results regressed due to current incompatibility with the size of the dataset and the slow processing. The last model after exhausting other options was a random forest regressor notable for large datasets with many features where non-linear relationships and mixed data types can be captured. The model outputted the best results as stated in Figure 2.1.1. The process of the model follows the same protocols of data cleaning and feature engineering and then unlike linear regression as a reference model, random forest regressor handles these features in an improved manner between its relationships between one feature to another feature. The model outputs multiple decision trees which each makes predictions then the forest is trained on a random size of its trees. The trained dataset uses the provided features in the score, total votes, and TF-IDF implementation to best predict the helpfulness ratio. The model can then be hyper-tuned e.g. grid search but the drawback is the naive time consumption it takes to do so for the testing session.

3 Sentiment Analysis

The next predictive task that will be completed is a sentiment analysis where keywords from a user's text review will be used to predict whether the rating is positive or negative based on what the user wrote in their text review. This model uses a baseline that establishes a lower bounded classification, where all reviews are first classified as positive. The main baseline for comparison that this program uses is the TF-IDF vectorization. The model would take in a user's review and weigh the important words more heavily, then complete logistic regression to classify and compute the review's sentimental merit. For its features and data processing, stopwords and punctuation from the reviewer's text will be removed. Then, tfidfVectorizer was used to transform the text into numerical values for calculation from the TF-IDF vectorization that stored important terms. To verify the validity of this model, we used a couple of test cases to see how the model would run. Using the

product ID, “B00813GRG4,” this item scored one. Using this prediction model, the model yielded correctly. The text review was largely negative as the user wrote that the item came late and the product itself was labeled as Jumbo, yet the product was small and unseasoned. It can also reveal that some user scores do not match up with what they wrote. For example, product item “B000UA0QIQ.” In the review written by the user, the user described how the flavor was good, but gave a rating of two out of five, despite leaving a positive review. Using a review with a rating of five(“B006K2ZZ7K”), the model could state that the rating was hugely positive. Therefore, the model seems to be working properly from these few test cases.

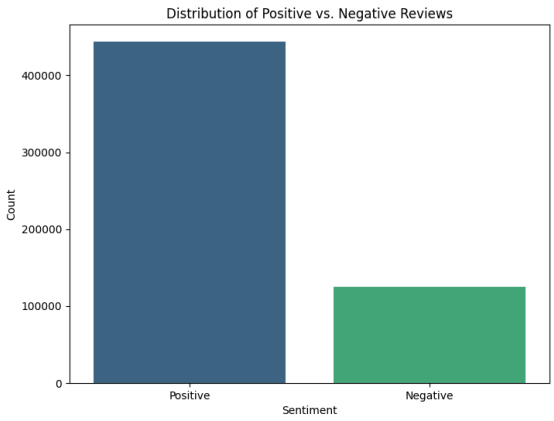

Figure 3.0.1: Review Distribution according to Model

3.1 Model

With this model, logistic regression was mainly used to predict the sentimental value of a review. Because the prediction we are trying to complete is a binary classification, the text data captured was used to weight important words accordingly to determine their impact on the review and their sentimental value. Optimizing this model required setting C to ‘balanced’ to ensure the model did not overfit.

Additionally, parameters for the TF-IDF vectorizer were adjusted to improve feature representation by setting the maximum features to 1000. By weighing the logistic regression model appropriately, class imbalance, which could skew predictions was addressed. Before putting any data into the logistic regression model, the data was processed where punctuation and stop words were removed. All the text was lowercase to ensure consistency between all user-written reviews. Scalability posed a challenge when handling large datasets, as the TF-IDF vectorizer can generate high-dimensional sparse matrices that consume significant memory. To mitigate this, the maximum number of features in TF-IDF was limited. The accuracy of the model using TF-IDF with logistic regression was 0.87. The measurement of accuracy consists of the proportion of correct predictions in classification tasks, or to elaborate, it is the number of correct predictions divided by the total number of predictions made. In addition, the determination of whether or not a prediction is correct is determined by the user’s review rating of the item, in comparison to the prediction done by the model. A one or a two rating would mean the review text written by the user should be negative and a four or a five rating would mean that the user’s review text is positive. Several baseline and alternative models were considered such as the random baseline. This would have predicted that most reviews served as a simple benchmark but performed poorly due to its inability to utilize patterns in the data and the amount of data in the dataset. Bag-of-Words combined with logistic regression was also tested but was less effective than TF-IDF, as it treated all words equally, failing to capture term importance. Using logistic regression with a Bag-of-Words, the model's accuracy became 0.83, lower than the accuracy of the logistical regression model with TF-IDF.

Overall, the logistic regression model with TF-IDF was chosen due to its simplicity and ability to capture significant words and weigh them according to importance.

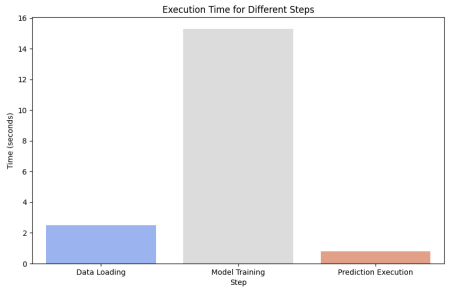

Figure 3.1.1: Time taken at each step

Sentiment analysis of product reviews is a widely researched area in natural language processing (NLP). This project uses web data from Amazon, specifically fine food product reviews. From the publicly available dataset of Amazon reviews, which includes product ID, user ID, review text, and numerical ratings, datasets similar to the one used in this project have been extensively used in prior research to study sentiment analysis, recommender systems, and customer behavior patterns. Due to its structured format and rich textual data, these types of Amazon web data make excellent resources for exploring how reviews reflect customer sentiment. This literature and similar datasets involve using such models like the Bag-of-Words (BoW) and Term Frequency-Inverse Document Frequency (TF-IDF) for feature extraction, combined with traditional machine learning models like logistic regression, support vector machines (SVMs), and random forests for classification tasks. More recently, state-of-the-art methods for sentiment analysis have shifted towards deep learning approaches, such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and transformers like BERT, which can capture the semantic and contextual nuances of text more effectively. Furthermore, in similar datasets, researchers have employed various preprocessing techniques such as tokenization, stemming, lemmatization, and stopword removal to enhance model performance and consistency.

Studies using comparable datasets often benchmark simpler methods, similar to the one used in the project of TF-IDF with logistic regression. Compared to more complex ones, like neural networks, deep learning models tend to outperform traditional methods in terms of accuracy, but often require larger datasets and significant computational resources. With this TF-IDF with logistic regression mode, this project aligns with findings in existing literature, showing that logistic regression with TF-IDF is a robust baseline for sentiment analysis. While state-of-the-art deep learning models like BERT could yield higher accuracy, their complexity and resource requirements make them less practical in a smaller dataset and less computational power.

Model: Logistic Regression	with Bag of Words	with TF-IDF
Accuracy	0.8033	0.8750

Figure 3.1.2: Model Accuracy using the product ID of ‘B006K2ZZ7K’

The results from this logistic regression model with TF-IDF feature extraction achieved reasonable classification in product review sentiment while managing a solid balance between computational efficiency and accuracy. With this model, the model could predict if a review given was a positive or negative review with a high level of accuracy as seen when a user gave a low rating despite leaving a positive review. This model outperforms the random baseline and the Bag-of-Words with logistic regression. However, while the model achieves better performance and delivers results, more complex methods like neural networks or transformer-based models would be a better implementation for sentiment analysis tasks due to improved accuracy and performance. TF-IDF worked well with feature representations as it captured important words in each review while discarding common, insignificant words. This is compared to the Bag-of-words approach, which

treats all words equally, possibly leading to noise in the feature set. In addition, the inclusion of unigrams and bigrams in this model to capture the context of the review text can further increase the performance with such words as "not bad" or "very good," giving better predictions on the review text. Overfitting can occur however when too many rare n-grams are stored. The low frequency of n-grams results in generalized information causing noisy or irrelevant features. Setting a maximum on the TF-IDF vectorizer, helped mitigate this problem by focusing on more meaningful, frequent terms. The model's parameters interpret which words or phrases are positive by assigning a positive coefficient with a positive n-gram and a negative coefficient for negative reviews. Words such as "good," "excellent," and "love" are given positive weights, and words like "poor," "bad," or "disappointing" are given negative weights. The magnitude of the weights given for each word reflects the strength of their sentiment value. This model succeeded because of its simplicity and ability to use TF-IDF effectively to extract features from the review text. By capturing important words and ignoring less informative ones, the model could focus on relevant features. With the help of logistic regression, sentiment was computed from specific words or combinations of words. Moreover, the regularization techniques used to prevent overfitting helped ensure that the model generalized well to new, unseen data. Other models failed or had limitations because of their relationship with the data. Random Forests, for example, are good at capturing non-linear relationships, but struggle with sparse feature space created by TF-IDF. The high-dimensional and sparse nature of text data would make it difficult for random forests to perform efficiently without the help of dimensionality reduction. Neural networks, on the other hand, could also potentially provide better performance by examining more complex relationships in the data but would require a larger dataset and great amounts of computational power for sentiment analysis.

4 Results and Conclusion

Recalling back to the two conclusive models for the notable tasks. Starting with the baseline models from the course and then exhausting iterations to trial improved models in the course or outside so, we found the use of random forest regressor to be extremely viable for helpfulness prediction, and using logistic regression with TF-IDF outputted an optimal solution for sentiment analysis for the ratings. Marginal jumps with accuracy would be suitable as answers for the conclusion question when trialing these models. Logistic regression with TF-IDF worked well due to the moderate size of the dataset and limited computational power as it interprets positive or negative coefficients based on words in the review text. Random forest regressor was a notable and optimal model for prediction the helpfulness can be attributed to its flexibility with pertaining non-linear data, especially between common words and data values of ratios which was ideal with both numeric and text-based features during our trial and error with models that couldn't handle so.

5 References

1. J. McAuley and J. Leskovec. (2013). *Web-FineFoods Dataset*. Stanford University.
2. J. McAuley and J. Leskovec. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. WWW, 2013.
3. Breiman, L. 2001. Random forests. *Machine Learning*, 45, 1, 5-32.