Tahsina Khan, Aleika Chery

Professor Xuan Wang

CSC 46000

5 May 2024

Final Report

Introduction

In the modern age of digital commerce, customer reviews play a pivotal role in shaping consumer decisions and providing valuable feedback to businesses. According to the Local Consumer Review Survey 2024, ninety-eight percent of customers gather insight from online reviews before they begin shopping. Even seemingly insignificant details, like the number of reviews a product accumulates, can tremendously impact a company's credibility as well as their product's legitimacy. The Harvard Business Review reported that in 2021, "online reviews were predicted to affect $3.8 trillion revenues worldwide". It's clear from these statistics that positive and negative feedback from previous customers are now a crucial aspect of our lives, and the way we sell and purchase goods has changed forever.

Although reviews are greatly beneficial and consumer-friendly, they also present several challenges that data science technology can help alleviate. For instance, four percent of all online reviews are fake, which translates to the loss of almost 152 billion dollars (Marciano, 2021). Some other challenges include spam or irrelevant content, unstructured data and biased opinions. Sentiment analysis, a powerful tool in natural language processing, enables us to extract subjective information from textual data, offering organized insights into buyer opinions and emotions. This project aims to perform sentiment analysis on Amazon product reviews, harnessing web scraping techniques to gather data and exploring various methodologies to process and analyze it. Despite encountering challenges in developing a fully functional sentiment analysis model, this report details the comprehensive process undertaken over the course of many weeks, from data collection to model selection, as well as our next steps.

Objectives

The primary objective of this project was to evaluate Amazon reviews in order to gain actionable insights on customer feedback, with the ultimate goal of performing sentiment analysis. In doing so, we aimed to address several challenges inherent in browsing through customer reviews, such as detecting and filtering fake or fraudulent reviews, mitigating biases, and managing the unstructured nature of textual data. Mainly, we sought to tackle the sheer and overwhelming volume of content that users often have to sift through in an attempt to make their purchases. We had a few main objectives: collect our data through web scraping, conduct data exploration and cleaning, provide data visualizations, and finally, develop a comprehensive and accurate machine learning model that implements sentiment analysis.

Data Collection

In late April, we began the process of acquiring the best data for our model to function smoothly. Our initial goal was not only to conduct sentiment analysis, but to categorize reviews based on the main critique or observation being made. For example, a review centered around the visual appeal of the device would be classified as an "appearance" review. The purpose of this objective was so that potential buyers could easily filter through loads of content, and once they specified the aspect of the product they desired reviews on, the sentiment analysis model could classify positive reviews from negative ones. However, due to time limitations, we decided to primarily focus on sentiment analysis.

Keeping this in mind, there were several factors we considered before choosing our data source. We were unsure of what category of products we should target, and how many reviews would be sufficient for an effective model performance. We also considered that the ratio of positive to negative reviews might result in a skewed model that is only accurate on the training set and not the test set. Finally, we decided on sourcing from Amazon reviews due to Amazon's immense reach and popularity: in March of 2023, Amazon attracted 2.4 billion visitors (Similarweb, 2023). Here is a list of the products we scraped from:

- ➔ Samsung Galaxy S24
- ➔ Decker Waffle Maker
- ➔ Apple iPhone 14 Pro
- ➔ Logitech iPad Keyboard Case
- ➔ Apple iPad (10th Generation)

In total, we used five amazon technology products in order to amass a collection of about 530 total reviews. In hindsight, using many different products was not the best choice if we desired model accuracy, but extracting reviews in this way was our solution to a difficult problem we encountered during web scraping. The problem was that Amazon limits the extraction of data from their website by capping the visible reviews to only ten pages. Thus, by implementing selenium and beautifulsoup, we were only able to derive exactly one hundred reviews per product. We did attempt applying an API called Rainforest API that gathered all seven thousand Samsung S24 reviews into a JSON file, but the file had a great deal of unnecessary information that was challenging to reformat.

Regarding the source code, after the Selenium Chrome WebDriver was initialized, we specified the exact Amazon URL reviews page we wanted to scrape from using the get method. An initial HTTP GET request is also made to the URL to verify the connection and retrieve the page content. Then, a loop starts that iterates through the review pages until it no longer detects a specific class in the HTML. In each iteration, the script constructs the URL for the current page and navigates to it using the WebDriver. A delay (time.sleep(5)) is included to ensure the page loads completely before extracting data. The page source is then parsed with BeautifulSoup. The script searches for all review containers using the find_all method with the 'div' tag and the data-hook='review' attribute. For each review, it extracts and appends the reviewer's name, rating, date, title, and review text to the respective lists that were created beforehand. To deal with pagination, the script checks for the presence of the 'Next' button by looking for the 'a-last' class. If the 'Next' button is found, the URL for the next page is constructed and the loop continues. If not, the loop breaks, indicating the end of the review pages. Finally, the URLs are printed to track the pagination process, and the loop continues until all review pages are processed.

Data Cleaning and Preprocessing

Once we created the dataset, we needed to clean and preprocess the data frame so that the model could clearly interpret it. Our data frame consisted of seven features and 530 rows, with 299 positive reviews and 231 negative reviews. When exploring the rating feature, we found that there were 252 five star ratings, 47 four star ratings, 102 three star ratings, 71 two star ratings, and finally, 58 one star ratings. We noticed that our web scraping algorithm had extracted ratings twice: once in the rating feature, and second in the title feature, so our first step was deleting any unnecessary phrases and words that only reduced comprehension of the review. This included removing reviews made in other languages using the langdetect package. We also removed features like rating_date and profile_name because they weren't very relevant to sentiment analysis. Noise such as punctuation, numbers, HTML tags, URLs, and spelling errors were removed using tools and libraries like NLTK, spaCy, and regular expressions.

Additionally, emojis were identified and removed from the text using specialized functions and regex patterns. Emojis are arguably useful for sentiment analysis, but we wanted to ensure that the text data was free from elements that could interfere with the model. Next, case normalization was performed by converting all text to lowercase, which reduced variability and made it easier to compare and match words. Tokenization was employed to break down the text into smaller units, such as words or sentences, using functions like `word_tokenize()` and `sent_tokenize()` from NLTK , which helped isolate meaningful elements of the text for further analysis. Stopwords, common words that add little value to the text, such as articles, prepositions, and conjunctions, were removed using NLTK's stopwords list. Examples include “and”, “but”, “like”, etc. By filtering out stopwords, we were able to focus on the words that convey the sentiment of the reviews. Lemmatization was applied to reduce words to their base or dictionary form, enhancing consistency and reducing complexity in the dataset.

Finally, the cleaned and processed text data was vectorized, converting the words into numerical values so that they could be understood and analyzed by machine learning models. This comprehensive data cleaning approach ensured that the text data was well-prepared for the subsequent stages of analysis.

Data Visualization

For the visualizations we used the Samsung Galaxy S24 product. To create the visualizations, we began by cleaning the data. We parsed the JSON file containing the reviews to extract key fields such as titles, bodies, dates, and ratings of each review. We focused on these specific fields to observe trends independent of the authors, thus minimizing bias. This approach allowed us to maintain the integrity of the data while ensuring that the visualizations reflected genuine patterns and insights. For the visualizations, we used various methods to represent different aspects of the review data. First, we created a bar plot using Matplotlib to show the frequency of reviews across different star ratings. This bar plot revealed that the majority of the reviews were 5-star ratings, with a significantly smaller proportion of 1 and 2-star ratings. Next, we developed a line graph to depict the number of reviews posted over a three-month period. This graph highlighted a notable increase in reviews from early April to the first week of May, likely due to a Mother's Day sale.

Additionally, we generated a word cloud using the WordCloud library, alongside NLTK tools such as stopwords and WordNetLemmatizer. By removing common stopwords and tokenizing and lemmatizing the text, we created a word cloud that effectively represented the most significant terms in the reviews. Lastly, we performed a sentiment polarity distribution analysis using TextBlob to assess the overall emotional tone of the reviews. This analysis classified the reviews into positive, negative, and neutral categories, revealing a predominance of positive reviews, a moderate number of neutral reviews, and fewer negative reviews. The highest peak in the sentiment distribution was observed at a polarity of 0.3, indicating a generally favorable customer sentiment.

Model Evaluation and Challenges

When creating the data visualizations for the Amazon product reviews, we encountered several challenges that required careful consideration and problem-solving. Initially, we had to clean the JSON file, removing unnecessary fields such as author and country while retaining important information like date and review title. This process ensured that the data was streamlined and focused on the relevant aspects of the reviews.

One of the major challenges arose during the creation of the word cloud. The presence of numerous stopwords posed an obstacle, as we had to meticulously add them to the code to eliminate redundant and insignificant words that could skew the visualization's interpretation. Additionally, we had to refine the code to ensure that the word cloud accurately reflected the main themes and sentiments conveyed in the reviews, enhancing its relevance and usefulness.

Performing the sentiment polarity distribution presented its own set of challenges. Determining the optimal number of bins for the graph required thoughtful consideration to effectively represent the distribution of sentiment scores without oversimplifying or overcomplicating the visualization. Moreover, leveraging the TextBlob package for sentiment analysis demanded extensive exploration and experimentation to grasp its functionalities and employ them effectively in the context of the project. Overall, overcoming these challenges demanded patience, adaptability, and a creative approach to data visualization and analysis.

Conclusion and Future Work

In the future, our group aims to enhance the quality and quantity of the data used for sentiment analysis and visualizations of Amazon product reviews. By increasing the dataset size, either through acquiring more Amazon reviews via web scraping or utilizing pre-existing datasets, we can improve model performance and generalization. Additionally, implementing data augmentation techniques, such as synonym replacement or back-translation, will help increase dataset diversity and robustness.

To improve model performance, we plan to experiment with exploring different model architectures, hyperparameters, and optimization techniques to identify the best configuration. We are also interested in exploring advanced neural network architectures like transformer models (e.g., BERT, GPT), which have demonstrated state-of-the-art performance in natural language processing tasks. Furthermore, breaking down reviews to analyze sentiment at a more granular level, such as assessing sentiment about product quality or customer service, will provide deeper insights.

We also envision creating a website that allows users to access real-time sentiment analysis, view visualizations, and read summaries of reviews. This platform would provide an interactive and user-friendly way for consumers and businesses to gain valuable insights from customer feedback, further enhancing the utility and impact of our project.