Amazon Customer Review Sentiment Analysis

ACRSent

**Data Science Capstone Project   
Data Acquisition and Pre-Processing Report**

Date: 05/09/2021

Team Members:

Name: Kevin Fitzpatrick

Name: Tessa Karlovitz

Name: Dustin Luchmee

[The purpose of this report is to describe the data of your project. It includes three major sections: Data Sources, Data-Processing, and Appendix]

**Identifying Data**

**Data Sources:**

[Identify the data sources of your project. It may have more than one data source. Describe each of them and explain why you select the data sources.]

* Amazon Customer Reviews from Amazon.com: books, mobile electronics, non-mobile electronics
  + Amazon freely offers customer review data from 1995-2015 on various products. We decided to pull book, mobile electronics, and non-mobile electronics data for our products since all are commonly bought products from Amazon. They are also products that most men, women, children and adults would buy.
* Amazon 100 Top Ranked Books by Year (Kaggle)
  + We found a dataset on Kaggle.com that contains the top 100 ranked books from Amazon per year. This dataset includes the titles of books so we are able to cross compare the them to see if number of reviews correlates to top ranked books, etc. We would also like to use this as an alternative target for a model (end of this quarter into next quarter).

**Acquisition Process:**

[Describe the data acquisition process. Is the dataset ready for download? How do you download the data? Do you need to write your own code to acquire the data from a public or private source? Describe how you do it. Are there multiple data sources? How do you integrate the data from multiple sources? Any other process involved in the acquisition process?]

We are able pull tsv’s from Amazon directly here: <https://s3.amazonaws.com/amazon-reviews-pds/tsv/index.txt>. It contains a list of downloadable tsv’s for various products. As described above we’ve pulled data for books, mobile electronics and non-mobile electronics. We can read the tsv’s directly from this website into pandas data frames or Python. We do not need to write any additional code as we can utilize read\_csv or open file. In the read\_csv we simply have to specify which tsv we want to load. For example,

url='https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\_reviews\_us\_Mobile\_Electronics\_v1\_00.tsv.gz'

df = pd.read\_csv(url, sep='\t', error\_bad\_lines=False)

There is a second data source on the top 100 ranked Amazon books by year. This comes from Kaggle.com (<https://www.kaggle.com/jiyoungkimpf/amazon-best-sellers-of-20102020-top-100-books>). We also read this data sorce via pandas’s read\_csv.

**Issues:**

[Are there any potential issues in data acquisition that have not be solved yet?]

There are no notable issues to mention with downloading the data.

**Data-Processing**

[Examine the data you have acquired and understand the data properties. Is there any pre-processing you need to do before you can start analyzing the data? For example, missing data, sparsity, noise, veracity, ambiguity, interoperability, etc. Describe each data issue in a sub-section and explain how you clean up the data.]

***Book Data (Tessa)***

The available book review data from Amazon is very large. We downloaded one tsv file which contains reviews between 2012 and 2015. This file began as 10,319,090 reviews and 6.7GB which is why we decided to only pull that one tsv. Our same tools could be applied to other files if desired.

We wrote stand-alone Python code (not Jupyter Notebook) to process the data in three passes.

The first pass removes any lines that contained notable errors. These errors include rows that did not have the expected 15 columns, rows that did not have ‘Books’ as the product category, rows without a valid data, row without a valid star rating, and rows that had empty titles or review bodies. Finally, we removed any non-verified purchases. After the first pass the data became 7,450,412 reviews so we threw away about 3 million reviews for having various data issues. By far the most common issue was we threw away reviews that were not verified purchases. The output file after the first pass became 2.66GB.

The second pass builds a dictionary to store book titles and the number of reviews associated with each unique book. We will use this to remove books with few reviews. It is also useful to understand the nature of the book review data. In fact, most books have few reviews. This produces an output file that contains 1.65 million books. Therefore, there are 1.65 million distinct books represented in the data after the first pass. Most of the books only have a single review. We have begun to perform EDA on this dictionary.

The third pass uses the dictionary from the second pass to remove reviews for books with fewer than a threshold number of reviews. If we use a threshold of 30 reviews, we get a final dataset with 34,194 books in total. There was a total of 2,621,612 reviews.

We ran all three passes on the 6.7GB file in less than 20 minutes so if needed or desired we could certainly process additional book review files.

Our second book dataset is taken from Kaggle.com as mentioned above. We were able to simply download the dataset from Kaggle and load it into pandas with read\_csv. The number of rows in the dataset is 1095. The header file and 1094 rows of ranked books. There are 6 lines missing that were dropped due to NaNs. The years range from 2010 to 2020. Clearly, we will take 2012-2015 to match up with our customer review data from Amazon. The hardest part will be to match on the titles of the books between the book review data and the top 100 list. We will likely need to add in some cleaning to make sure the correct titles match up.

***Electronic Data (Dustin)***

The non-mobile electronic data was acquired from Amazon (<https://s3.amazonaws.com/amazon-reviews-pds/tsv/index.txt>) and read into a pandas data frame within Jupyter notebook. This data sample contained 3,091,024 entries. Initial examination of the data was done by simple python coding (I.e., df.describe(), df[‘column name’].value\_counts()).

Initially, I had examined the number of star ratings for each rating category (1-5). Following that, I ran a histogram to get a feel of the numerical data within the dataset. While limited, what I was able to learn from this preliminary analysis is that many people do not make use of the total votes or helpful votes options. This led me to discarding these data columns from the data frame.

Next, I separated the year from the review date to make a year column. I did this for future analysis to examine what trends may be present in review of electronics. Questions I hope to answer using this column include what changes were present in the review of electronics through the decades, what features did people find most important to comment on in each decade, and what advancements happened during each decade as evidenced through product reviews.

Lastly, a preliminary analysis was done for review headlines to detect if there were any common reviews left. What was found was that most people comment “one star” or “five stars” reviews. What I have observed in examining the data is that products with poor star ratings tend to have longer review text. This is also true if someone is very enamored with a product or a feature of a product, but these lengthy positive reviews appear to be less common.

Next, I removed rows where the product was not an amazon verified purchase and where the marketplace was not US. This was done to ensure that only verified purchased products were included and as to ensure that no extraneous cultural biases existed within the reviews.

In the data frame, I had observed that purchased items were not always non-mobile electronics per se, but rather could be parts needed for devices or material to be used, such as CDs. This data could be problematic, as these reviews may cloud the review analysis of actual devices. Because of this, this data will be filtered out by identifying keywords that could be indicative of “parts” or “content” products. For example, the words “male” and “female” pertain to cables, which would be of interest to remove.

Lastly, because the data frame contained 3,091,024 entries, it was going to be necessary to split the data frame into chunks as the data frame is too large to effectively work with. What was done was I had separated the data frame into 12 equal partitioned frames consisting of 200,000 rows. Each data frame could be called upon and concatenated if desired later on.

***Mobile Electronics Data (Kevin)***

When investigating the mobile electronics dataset, there were 5 total rows that had at least one null value in them. The columns that had a null value varied but included the star\_rating and review\_body which are important variables to use in future analysis. Using pandas, we easily dropped all rows with any null values since there were so few occurrences. Using data will null values would cloud the analytical results so we thought it would be best to just remove them.

To perform analysis where we look at reviews over time, we made sure to change the review date column to a datetime that pandas can easily read and filter by using pandas built-in to\_datetime function.

Having emojis in the review body presented a challenge since we didn’t want to lose the emotion/sentiment behind the emoji, but obviously Python can’t read emojis. To solve the problem, we turned to a Python package called “demoji” which converts an emoji to simple text. For example, a smiling face emoji would be translated to a text string such as “smiling face” so we can read into the emotional intent for sentiment purposes.

When doing some initial EDA, we discovered some characters that needed to be cleaned. These included extra spaces, blanks, html tags, and punctuation. We wanted the spaces, blanks, and tags removed, but decided to keep the punctuation in case it was needed later for sentiment purposes. To clean this up, we used BeautifulSoup to get the text from the review body. This had the effect of removing html tags (mostly line breaks) and cleared up any blanks. Finally, regex was used to remove any instances of 2 or more spaces. For how punctuation was handled, see the Tokenize section below.

The thought was that expanding contractions would be the route to take when doing further analysis so we installed a Python package called “contractions”. This handy package has a function called fix where you can feed it a text string and it will expand all the contractions it finds. For example, with the contractions package, the contraction I’m would be converted to I am. We believe this will help when performing lemmatization and the removal of stop words discussed below.

We want all of the words to be consistent when grouping or counting so using Python’s built-in lower function, we turned all of the words into lowercase for consistency.

Tokenizing text means to split text up into smaller parts, either sentences from a paragraph or words from a sentence. In our case, we wanted a list of words so we went with the Python package NLTK and used the function word\_tokenize to split up the words in the review body. This allowed us to get our list of words, and it also separates out the punctuation in each sentence. We’re not sure if punctuation will be used in our analysis but we wanted to make sure it was there just in case.

When attempting to get keyword extraction or sentiment, we thought it would be best to remove all of the stop words in the review text. Stop words are words that are very commonly used such as “and” and “the” which don’t really provide any sentiment. By removing these stop words, the remaining words should be able to tell us the emotion of the review body. To do this, we used NLTK and imported their stop words. Then, after tokenizing our text, we used a list comprehension to remove the stop words.

Lemmatization is used in NLP to get the words back to an inflected form. For example, the word “talk” could appear in our text as talks, talked, talking, etc. But lemmatization brings the word back to the base form of “talk” to better group and analyze. We used the NLTK WordNetLemmatizer to complete this task on our text.

**Appendix**

[Provide the code or pseudo code, data definition, sample data, and any other information in the appendix here.]

**Electronic Data**

**El –** pandas data frame for electronics data

**Marketplace** – Country where product is purchased

**Customer ID** – unique ID # for each customer

**Review ID** – unique ID # for each review

**Product Parent** – ID # of product family

**Product Title** – Name/description of the product

**Product Category** – Category of product

**Star Rating** – How many stars (average) product is rated

**Verified Purchase** – Whether product purchase was verified through Amazon

**Review Headline –** Headline of review left by customer

**Review Body –** Full text of the review

**Year** – Year review was left

**Mobile Electronics Data Dictionary:**

|  |  |  |
| --- | --- | --- |
| **Field** | **Data Type** | **Description** |
| marketplace | string | 2 letter country code of the marketplace where the review was written. |
| customer\_id | string | Random identifier that can be used to aggregate reviews written by a single author. |
| review\_id | string | The unique ID of the review. |
| product\_id | string | The unique Product ID the review pertains to. In the multilingual dataset the reviews |
| product\_parent | string | Random identifier that can be used to aggregate reviews for the same product. |
| product\_title | string | Title of the product. |
| product\_category | string | Broad product category that can be used to group reviews |
| star\_rating | integer | The 1-5 star rating of the review. |
| helpful\_votes | integer | Number of helpful votes. |
| total\_votes | integer | Number of total votes the review received. |
| vine | string | Review was written as part of the Vine program. |
| verified\_purchase | string | The review is on a verified purchase. |
| review\_headline | string | The title of the review. |
| review\_body | string | The review text. |
| review\_date | datetime | The date the review was written. |

**Book Data Dictionary (excluding columns not used):**

**Amazon:**

**books\_df –** Pandas data frame for book data

**product\_title** – Title of the book

**review\_data** – Date of the review

**review\_body** – body of text of the review

**star\_rating** – star rating given in the review

**verified\_purchase** – yes or no for verified purchase or not

**helpful\_votes** – number of helpful votes for the review

**total\_votes** – total number of votes for the review

**len\_review\_body** – length of the review body for the review

**Kaggle:**

**Year** – year for that top ranked book

**Rank** – rank for that book that year

**Book\_Title** – title of the book

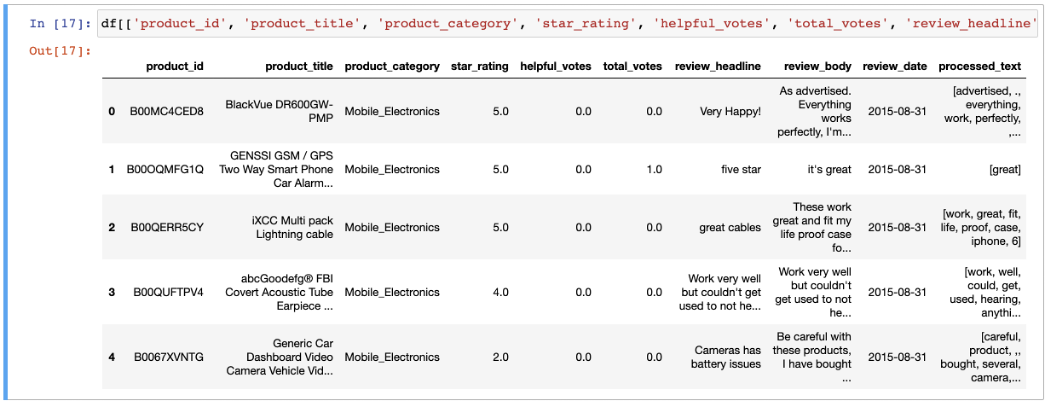
**Author** – Author of the book

**Rating** – Average star rating

**Num\_Customers\_Rated** – Number of customer ratings for the book

**Price** – Price of the book

**Mobile Electronics Sample Data (excluding columns not used):**



**Book Data Sample Data Before 1st Cleaning Pass (excluding columns not used):**

**Table

Description automatically generated**

**Book Data after the 1st Pass:**

Text

Description automatically generated

**Book Data after the 2nd Pass:**

Text

Description automatically generated

**Mobile Electronics Code:**

Link: <https://github.com/TKarlovitzDrexel/DSCI591_ACRSent/blob/main/Mobile%20Electronics%20Pre-Processing.ipynb>

**Book Data Code:**

<https://github.com/TKarlovitzDrexel/DSCI591_ACRSent/blob/main/clean_books_1stpass.py>

<https://github.com/TKarlovitzDrexel/DSCI591_ACRSent/blob/main/clean_books_2ndpass.py>

<https://github.com/TKarlovitzDrexel/DSCI591_ACRSent/blob/main/DSCI591_books_exploratory.ipynb>

Table of Contributions

The table below identifies contributors to various sections of this document.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Section** | **Writing** | **Editing** |
| **1** | **Data Sources** | **All** | **All** |
| **2** | **Data Pre-Processing** | **All** | **All** |
| **3** | **Appendix** | **All** | **All** |

**Grading**

The grade is given on the basis of quality, clarity, presentation, completeness, and writing of each section in the report. This is the grade of the group. Individual grades will be assigned at the end of the term when peer reviews are collected.