Amazon Book Review Analysis with NLP

Melanie Hanna

1. INTRODUCTION

Readers today can judge a book by far more than just its cover. Thanks to Amazon's dominance in the online book marketplace, gaining access to the opinions of thousands of other readers is only a few clicks away. But how can users efficiently evaluate those reviews to find features relevant to their interests?

Amazon currently employs recommender systems based on users' purchases, page views, and product characteristics. This approach works well for products such as electronics or household goods but books have much more nuanced features. In this analysis, we set out to create a recommender system that suggests books based on topics included in favorable reviews by the user through a NLP analysis of review text.

Potential clients of this work include Amazon to offer more targeted recommendations to users but this kind of model can be expanded to any other industry that collects textual review data. For example, an online food delivery company such as Grubhub could apply this model to restaurant review data to suggest new restaurants with similar features to those in the highly rated reviews of each user.

1.1 Data Sets

Amazon book review data was provided by Julian McAuley at the University of California San Diego, who compiled a 5-core data set containing nearly 9 million book reviews (see link below). Columns used from this data are as follows:

- reviewerID: ID of the reviewer
- asin: ID of the product
- reviewerName: Name of the reviewer
- helpful: Helpfulness rating of the review, e.g. 2/3
- reviewText: Text of the review
- overall: rating of the product
- summary: summary of the review provided by reviewer

To provide context to these reviews, I merged in the title of each product by reading through McAuley's metadata file.

http://jmcauley.ucsd.edu/data/amazon/

1.2 Data importation

Both the book reviews and metadata files were provided as JSON files, which could be easily imported into an iPython notebook. However, the book review dataset contained more than 9 GB of uncompressed data and the metadata file 10.5 GB. As I was working on a machine with only 16 GB of RAM, I could not simply import both datasets before merging. The metadata file contained information on *all* Amazon products, not just books, so I first tried to extract only rows with "Books" in the

"Categories" column. Unfortunately, the resulting dataframe of book-only rows was still too large for memory to also accommodate the review dataset.

To conserve memory as much as possible, I used concurrent thread processing to read through the metadata and book reviews file simultaneously and match on the "asin" or product ID. All other metadata was discarded and the "title" key-value pair was added to the reviews dictionary. To cut down on processing time, I limited the number of reviews to 500,000. The reviews are organized by product (i.e. the first five rows are reviews for one book, the next five for another book) so by only using the first half million, we can be sure to include all reviews for only a subset of the books. The final dataset used in this project's analysis contained reviews for more than 12,000 different books from 214,608 unique users.

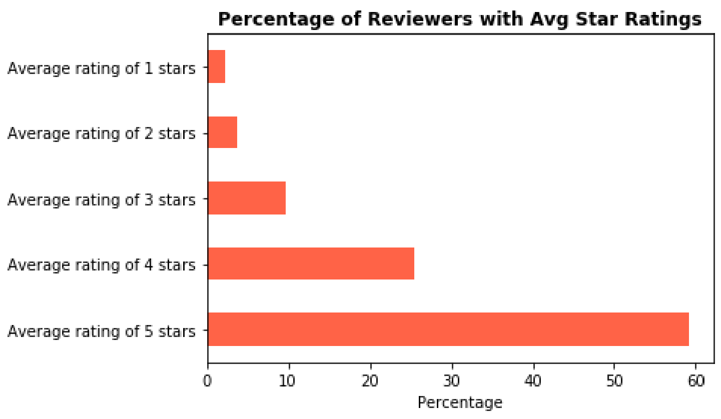
1.3 Data cleaning

Often, the summary of each review offers the most direct view into the reader's feelings about the book so I combined the review text and summary columns into one for the full analysis. All other data was already in the correct type as strings except for the overall score, which was a float. Although the scoring could be thought of as a classification, I wished to keep the score as a numerical entity to preserve the relationship between each rating (i.e. that 3.0 is lower than 4.0).

2. DATA EXPLORATION

As we are evaluating favorability of review text, it can be helpful to understand how many reviews rate the book as 4 stars or higher (the definition of favorable for this analysis). Surprisingly, 78% of all reviews were favorable with only 10% rating the book with 2 stars or fewer. We can surmise that either readers on Amazon are adept at buying books they know they will enjoy or that only satisfied readers take the time to write a review.

Do certain reviewers more reliably rate books favorably or are reviews mainly mixed for most users? After finding the average rating for all users and sorting into 5 bins, I found that almost 58% of users rate books 5 stars on average. In stark contrast, only around 8% rate books 1 or 2 stars on average. A little less than 10% of reviewers rate 3 stars on average—these are our users who either review some books negatively and others positively or find them all rather so-so.



2.1 N-gram analysis

Using scikit-learn’s CountVectorizer to convert each review into a bag-of-words vector, we can determine which words are used most frequently across all reviews. First, I “stemmed” the review text to ensure such words as “like” and “liked” were counted as the same by applying nltk’s word_tokenize method in conjunction with a PorterStemmer. The vectorizer’s maximum features were set at 100, and the default stop words were applied to filter out common words like “the” and “a”. The top 20 most frequent single words used across our subset of reviews are shown below.

1. **Book**
2. Thi
3. ‘s
4. Wa
5. **Read**
6. N’t
7. Hi
8. **Stori**
9. ’’
10. **Like**
11. **Love**
12. ‘ ‘
13. Ha
14. **Charact**
15. Just
16. **Time**
17. **Veri**
18. **Good**
19. Did
20. **Great**

We can see that the stemmer made some interesting choices—such as “very” changed to “veri” and “thi” to “this”. Words with more meaning are bolded in the list below and represent what we’d typically expect from a book review: “love”, “like”, “good”, “great” to indicate feelings and approval and “charact”, “book”, “read”, “stori” to show the main topic of each review.

We can also see how this analysis compares when we sort the review’s text into bi-grams.

1. Thi book
2. Read thi
3. Book wa
4. Thi wa
5. Read book
6. Thi stori
7. Love thi
8. Thi seri
9. Thi novel
10. Main charact
11. Enjoy thi
12. Like thi
13. Recommend thi
14. Book read
15. Wa veri

Only 15 bi-grams were returned in the top 100 features that did not contain punctuation. We can see that several words from the previous list make a repeat appearance (“love”, “book”, “stori”), but we’ve also discovered that “main character” is an important feature.

REFERENCES

Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering
R. He, J. McAuley
WWW, 2016