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.

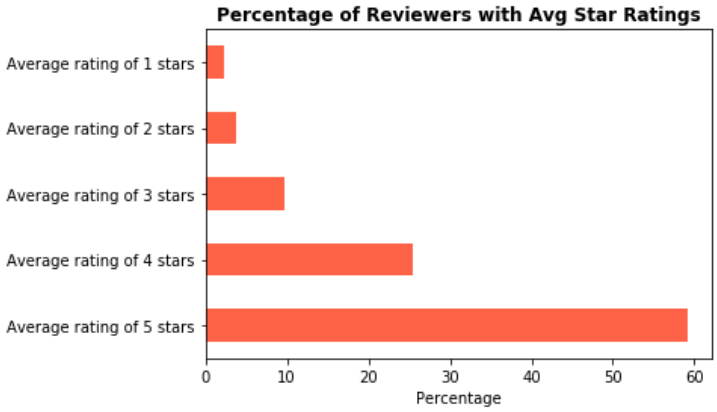
1.3 Data cleaning

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 preferred 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 mediocre.



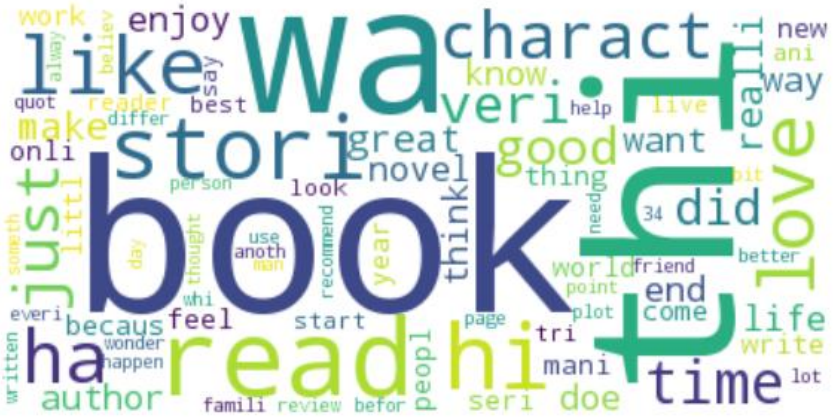
2.1 N-gram analysis

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.

These word frequencies can also be visualized as a word cloud, shown below.



We can 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 that did not contain punctuation were returned in the top 100 features. Several words from the previous list make a repeat appearance ("love", "book", "stori"), but by grouping into bi-grams, we've also discovered that "main character" is an important feature.

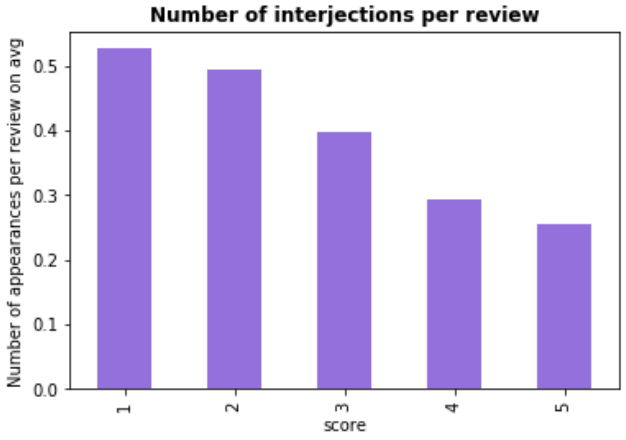
2.2 Parts of speech

The spaCy package for NLP in Python is much more powerful and robust than NLTK, which was used in the previous n-gram analysis. Using spaCy, we can classify all tokens by their parts of speech (nouns, adjectives, pronouns, etc.) and investigate any possible patterns. For example, do 5-star ratings tend to use more adjectives than 1- or 2-star ratings?

By iterating through all reviews in the dataframe, I counted the different parts of speech present in each review for a given star-rating and added to a collective dictionary. I then "normalized" the parts-of-speech count by dividing by the number of reviews for that star-rating, producing a dataframe that contained the average number of each part of speech per review for each star-rating.

No. of stars	**1**	**2**	**3**	**4**	**5**
Adjective	19.18	21.48	22.28	21.74	16.70
Adverb	15.72	18.30	17.61	15.14	11.26
Interjection	0.53	0.49	0.40	0.29	0.26
Noun	35.45	38.57	39.51	38.88	30.10
Number	2.00	1.99	1.95	1.84	1.50
Pronoun	12.57	14.18	13.88	12.51	10.02
Verb	36.66	40.03	39.57	36.93	28.61

The data does not support the hypothesis that 5-star reviewers use adjectives with greater frequency than 1-star reviewers. However, positive reviews use interjectives (ex. "Wow!", "Gee whiz!") at twice the rate of negative ones.



3. NAÏVE BAYES MODELING

We can train a Naïve Bayes model to predict the star rating of a review by "reading" the review's text. From Bayes' Theorem,

$$P(c|f) = \frac{P(c \cap f)}{P(f)}$$

Where c represents the classification of each star rating and f represents a feature vector of words. Therefore, we are finding the probability of each rating given a certain bag-of-words representation of a textual book review.

3.1 Model creation

All reviews were vectorized into a bag-of-words representation, omitting stop-words and any terms occurring in more than 90% of the documents. Their respective scores were segregated into a separate series. The bag-of-words matrix and the score series were divided into a training set (70% of the total) and a test set (the remaining 30%). A multinomial Naïve Bayes classifier was then fit to the training data.

3.2 Model evaluation

The default Naïve Bayes classifier returned a 67% accuracy on the training set and a 60% accuracy on the test set, indicating this model is not overfit. The accuracy here is not exceptional but the model does have five different classes to choose from and random guessing would produce an accuracy of just 20%.

Grid search optimization was used to try and improve the accuracy from the default parameter of alpha = 1.0. After searching with five alphas between 0.1 and 50, 1.0 was still returned as the most accurate model but with 0.1 as the second-most accurate. To ensure the true maximum was found, I then searched again with five alphas between 0.6 and 4. An alpha of 0.8 returned the most accurate model in this case but with only a 0.008% increase in accuracy over alpha = 1.0.

I then fed the model some new reviews pulled from Amazon for *Hillbilly Elegy*, which was published in 2016 and not included in the dataset. The first review was a 5-star rating:

> *"This Harvard Law grad finally has a Yale man he can respect. I grew up without running water in Boone County, WV, and wound up with a degree from Harvard Law School. JD Vance's story brought me to tears and cheers, for he has told the story of my people."*

The model predicted with 90% certainty that this review was rated as 5 stars, even though the review does not contain any giveaway words like "love" or "enjoy".

Next, I fed a negative review (1 star).

> *"Insulting Trash. Classicism to the tenth degree. No actual analytical thought in how circumstances out of the individuals control contributed to their plight. Also, don't*

degrade a whole society based on your own trashy family. Also, the author seems to think he's special. He's not, he had decent grandparents so he had some good influences that contributed to his success. He should focus more on that instead of his mommy daddy issues."

The model again predicted the star rating correctly, though with only 50% certainty this time. Potentially the more difficult task would be to rate a 3-star or "so-so" review:

"Good, but no real surprises. I enjoyed reading this memoir because of the fact it was a memoir and, if well written, I am generally intrigued by the lives of others. However, one of the big hypes of the book was it supposedly contained the answers to why so many voted for Trump. I didn't find any fascinating revelations on that issue and am still stupefied by it. There were some interesting statistics though and, overall, it was a good read. I'm sure there will be good discussions at book group tonight!"

The model incorrectly classified the above review as 4 stars with 72% certainty but 3 stars was the next most probable answer. We can see that although the Naïve Bayes classifier has learned sentiment analysis and can correctly classify reviews as positive or negative, it struggles with identifying the shades of grey in between.

4. TOPIC MODELING WITH LATENT DIRICHLET ALLOCATION

We can infer greater meaning into these book reviews by determining their underlying themes or topics. Latent Dirichlet allocation (LDA) represents documents as a mixture of topics, which are in turn represented by a mixture of terms. The number of topics is pre-defined. LDA makes the prior assumption that these mixtures follow Dirichlet probability distributions.

4.1 Data cleaning with spaCy

Before we can identify themes within each review, the text must be pre-processed. First, I parsed each review through spaCy and lemmatized each token. SpaCy's Phraser function was then applied once to group bigrams such as "nursing home" and "unabridged audio". Finally, stop-words and pronouns were removed. An example of the transformation is below:

Original:

"Flawless. Anything I've read by Gibran is, in my mind, flawless. This, the most famous of his works, is no exception. It is simple, yet deep; honest and profound; moving and inspirational. Gibran's work is one of a kind, and can be far more encouraging and moving than any self-help program or therapy or anything like that. The poetic style, the aphorisms, the parables, the almost biblical feel, are all just what over-worked, over-stressed, modern and spiritually starved worldly people need."

Transformed:

"flawless read gibran mind flawless most_famous work no_exception simple deep honest profound inspirational gibran work kind far encouraging self help program therapy like poetic style aphorism parable biblical feel work stress modern spiritually starve worldly people ne"

4.2 Topic model creation

I used the gensim library to first learn the dictionary of the corpora by iterating through all 5-star reviews and filtering out very rare or very common tokens. The reviews were then converted to a bag-of-words representation and saved as a matrix, which was then fed as training data to the LdaMulticore algorithm. The number of topics was set at 50.

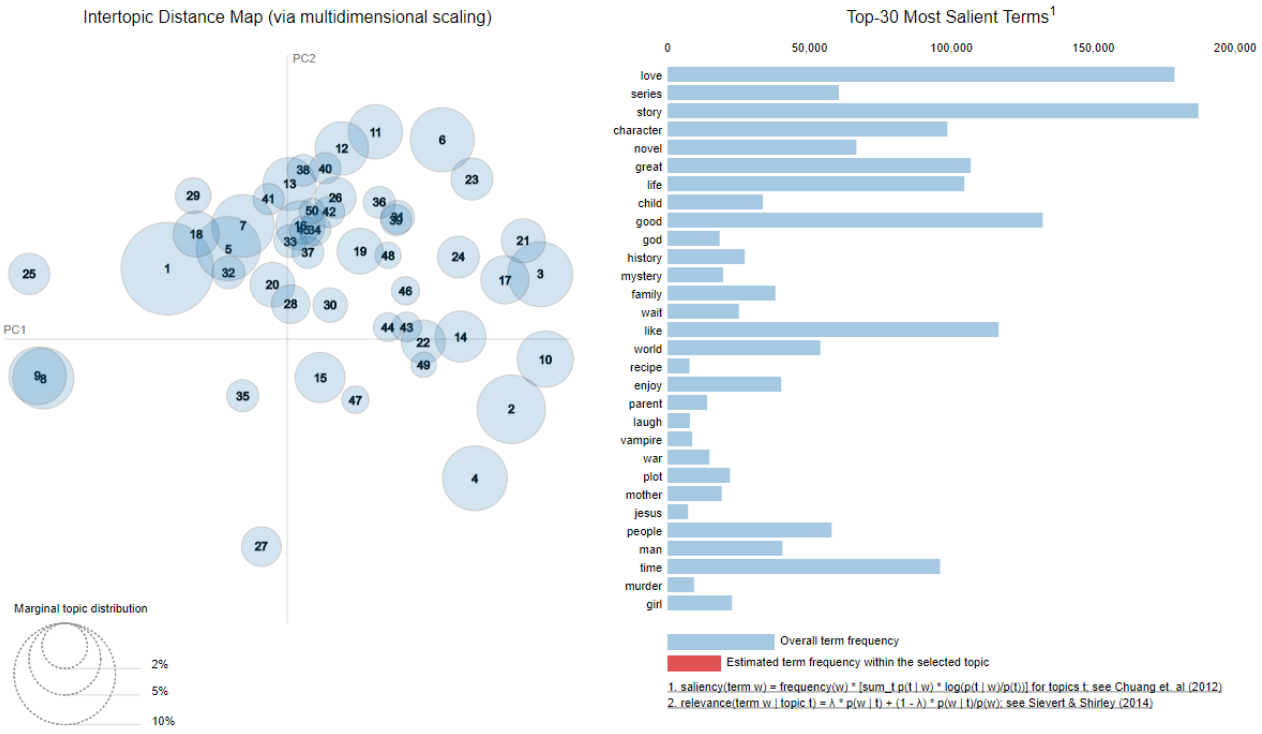
Each topic was then manually inspected to try and discern the overall theme:

Topic No.	Theme
0	Rome
1	sports, monsters
2	Christianity
3	general
4	character
5	sci-fi
6	humankind
7	contractions
8	general
9	general
10	women
11	good, great
12	world, experience
13	life, people
14	general
15	love, like
16	family relationships
17	story, family
18	physical book
19	plot
20	male names
21	angel
22	general
23	paranormal
24	history

Topic No.	Theme
25	story
26	America
27	humor
28	fantasy
29	general
30	good, great
31	Jewish history
32	American military
33	survival
34	fairy tale
35	like, love
36	self-help
37	story
38	family
39	cookbook
40	general
41	series
42	murder mystery
43	technical
44	general
45	novel
46	Dallas
47	children
48	general
49	female names

Certain topics were easy to classify, like #27, which contained terms like "laugh", "funny", "hilarious" and was defined as "humor". Some were less definite. Topic #18 included terms like "word", "art", "edition", "illustration" and was classified as "physical book" to indicate features pertaining to the book itself. Others were simply classified as "general", such as #48. I could not find an overarching theme to the dominant tokens of "juliette", "like", "know" in this topic.

We can visualize these topics in two-dimensional space using pyLDAvis (see below). Unfortunately, the numbers assigned to the topics in the visualization do not match the numbers given by gensim (above).



We see that most topics are clustered in the top righthand corner of the plot but a few are in the opposite quadrant. These outliers (9, 8, 25, 1) to the left appear to be our "like, love" topics and deal more with descriptions. Near the bottom of the plot, topics 27, 35, 4, 2, 15 are versions of the "cookbook", "self-help" and "physical book" themes. Therefore, we can surmise that the y-axis separates themes by abstraction, with more concrete themes on the bottom and more abstract themes (family, adventure) on the top. The x-axis could then be descriptive on the left and object-based (vampire, Jesus) on the right.

4.3 Topic model evaluation

We can apply the LDA model directly to review text to extract the underlying themes. The text first must undergo the same pre-processing as above—parsing, lemmatizing, finding bi-grams, and omitting stop-words. The processed text is then converted to a bag-of-words vector and fed to the model.

For example, the following review returned "physical book" (0.493), "life, people" (0.271), and "good, great" (0.184) from the LDA model.

> *"Words to Live By and Learn From. A must in everyone's library for graceful poetry, inpirational reading and lessons to live by and to even to let go. A new printing would be a welcome as the "used" ones in good condition are quite expensive."*

We can surmise that the model extracted the "physical book" topic from the last sentence about printing. The other two topics refer to the favorability of the review ("good, great") and that the review mentions life lessons ("life, people").

4.4 Recommender system

By extracting the topics in each review, we can build a recommender system to match users with titles that the model identifies as containing the same themes as books the user has reviewed favorably (5-star rating). After taking a reviewer ID as input, the recommender system first iterates through all favorable reviews by the reviewer and extracts the three most common themes with frequency above 0.05. Then the system runs through all five-star reviews in the database and finds the dominant topics in each from the LDA model. If a match exists between user-preferred topics and a dominant topic in the review, the book's title is added to a list of recommendations.

This recommender system was used to find suggested titles for a reviewer going by the name of Aistis Zidanavicius (ID: A010997525FU27TAPMJCG), who rated highly the following titles:

The Monk Who Sold His Ferrari
The Zahir: A Novel of Obsession
Brave New World
The Alchemist
The Greatness Guide: 101 Lessons for Making What's Good at Word and in Life Even Better
Start Where You Are: Life Lessons in Getting from Where You Are to Where You Want to Be

The LDA model identified the "life, people", "like, love" and "self-help" topics as the most dominant in Aistis' reviews of these titles, and the recommender system returned more than 10,500 titles as matches for these themes. A sample of those titles included *Big Fish, Shadowmagic: Prince of Hazel and Oak,* and *The Hair Color Mix Book: More than 150 Recipes for Salon-Perfect Color at Home*.

As the user's first two preferred topics are some of the broadest on our list, the system returned a wide variety of books. The first two titles in the sample could potentially be a match for the mysticism and fantasy that are themes in *Brave New World* and *The Alchemist*, and we can only conclude that the system identified *The Hair Color Mix Book* as a self-help guide.

5. CONCLUSIONS AND FUTURE WORK

As is clearly shown above, recommending titles by topic will need further refinement. One suggestion would be to create more numerous and therefore more specific topics in the LDA model. Additionally, we could search for topics that perhaps are not the most common in the user's reviews but are similar (as shown on the 2-D plot above) to other less common topics in his/her reviews. For example, although perhaps the "American military" and "survival" topics each appear only once as preferred themes for a user, the model would consider the user's preference for action-oriented books and weight those topics more heavily.

This recommender system is not a stand-alone solution for book suggestions. Data on the users themselves should be incorporated to narrow down the large number of recommendations. For

example, the system would probably not have suggested *The Hair Color Mix Book* if it had known Aistis was a man.

Additionally, the LDA model should be trained on a larger dataset. Due to memory constraints, not every review from each user was captured in this analysis, and the model can better assess a reviewer's preferences with more data.

Recommendations for the client based on this analysis:

- After further refinement as outlined above, implement the recommender system to suggest books based on a user's preferred themes.
- Identify new titles to stock that correspond to the terms the Naïve Bayes model finds most accurate in predicting favorability of a review.

REFERENCES

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

Modern NLP in Python (2016)
Patrick Harrison
GitHub, https://github.com/skipgram/modern-nlp-in-python/blob/master/executable/Modern_NLP_in_Python.ipynb