Book Data Analysis from GoodReads and #Books Twitter Analysis

Laura L. Lamoureux

Syracuse University

Purpose

The purpose of this project was to use Python to read in data collected from Goodreads and perform tasks on the data to see what the data had to tell us. For example, was there a correlation between attributes like page count and average rating, or page count and number of ratings. A secondary purpose was to read in Twitter data around the #books to see what could be learned from that specific hashtag.

Dataset Details

Originally, this project started dataset from Kaggle seen here: https://www.kaggle.com/jealousleopard/goodreadsbooks. However, after attempting to pre-process the data, it became apparent that the dataset was not going to be able to address the questions put forth in the proposal.

After some additional research, another Goodreads dataset was found on GitHub; found here: https://github.com/BahramJannesar/GoodreadsBookDataset/tree/master/dataset. Here are several JSON files that were curated a month ago. The first JSON file "book1-100k.json" was selected.

The dataset downloads as a JSON file. The dataset contains 58,292 rows with 18 columns.

Field Name	Data Type	Data Description	Sample
Id	Integer	A unique Identification number for each book.	1
Name	Object(String)	Book title.	Harry Potter and the Half-Blood Prince
RatingDist1	Object(String)	Count of the number of 1 'stars' the book has received	1:9896
pagesNumber	Integer	Number of pages the book contains.	652
RatingDist4	Object(String)	Count of the number of 4 'stars' the book has received	4:556485
RatingDistTotal	Object(String)	Total Rating Count	total:2298124
PublishMonth	Integer	Month book was published	16
PublishDay	Integer	Day book was published	9
Publisher	Object(String)	Name of Publisher	Scholastic Inc.
CountsOfReview	Integer	Count of text reviews	28062
PublishYear	Integer	Year book was published	2006
Language	Object(String)	Language book edition is published in	eng
Authors	Object(String)	Author of book	J.K. Rowling
Rating	Float	Average Rating for book	4.57
RatingDist2	Object(String)	Count of the number of 2 'stars' the book has received	2:25317
RatingDist5	Object(String)	Count of the number of 5 'stars' the book has received	5:1546466
ISBN	Object(String)	The International Standard Book Number (ISBN) is a numeric commercial book identifier which is intended to be unique	None
RatingDist3	Object(String)	Count of the number of 3 'stars' the book has received	3:159960

Loading and Preprocessing

The first step was import necessary libraries. Then the file was read in.

```
In [6]:
# Read in JSON file from Github: https://github.com/BahramJannesar/GoodreadsBookDataset/tree/master/dataset
# This dataset was collected last month
bookDF = pd.read_json('book1-100k.json')

# Display number of rows and columns
print("This dataframe has {} rows".format(bookDF.shape[0]))
print("This dataframe has {} columns".format(bookDF.shape[1]))

This dataframe has 58292 rows
This dataframe has 18 columns
```

Based on the large number of rows, it was decided to narrow down the data to books that had a language code of 'eng'. As a vast majority of the books curated, had that label it was felt that it would give a wide enough sample to be able to see some interesting data. This left a sample of the original data at 15,988 rows and the 18 columns.

```
In [9]:
# Limiting books to those labeled as english
is_lang_eng = bookDF['Language']=='eng'

eng_bookDF = bookDF[is_lang_eng]

# Display number of rows and columns
print("This dataframe has {} rows".format(eng_bookDF.shape[0]))
print("This dataframe has {} columns".format(eng_bookDF.shape[1]))

This dataframe has 15988 rows
This dataframe has 18 columns
```

The preprocessing of the data was next. As seen in the data details, the count for ratings was listed as #:#### - which implies the 'star' number before the colon and the total number (count) of that particular stars. For example: 1:9896 – meaning this particular book received 9,896 "one star" ratings. In order to be able to use those columns/that data – this needed to be fixed. The column "RatingDistTotal" was listed in much the same way: total:2298124. These six attributes were split into two additional columns for each original attribute. Then the column with the "counts" were added back into our sample data frame (eng_bookDF) as 'new' columns (with updated column names to reflect the data that they held). These attributes were then converted to numeric so that data analysis could be conducted on those attributes.

It was also noted during this preprocessing phase that the "PublishMonth" column was in fact the "day"; and the column labeled "PublishDay" was actually the month. This was discovered when looking at the data dictionary (we don't have 16 months).

PublishMonth	Integer	Month book was published	16

A book was looked up manually in Goodreads to confirm the theory that the "PublishDay" was actually held the "PublishMonth" information. The solution to this issue was to drop the currently labeled "PublishMonth" and rename "PublishDay" to "PublishMonth"

Other columns were also removed at this time as they were determined unnecessary for the data analysis to come. It should be noted that the original columns for "RatingDist1", etc. were eliminated as the count columns created in a previous preprocessing step replaced the need to keep those columns. This preprocessing left us with 15 columns/attributes to work with.

```
# Split RatingDist5 column into just the count column
countRating5 = eng_bookDF['RatingDist5'].str.split(":", n=1, expand = True)

# Creating a column for the counts for Rating 5
eng_bookDF['Rating5Count'] = countRating5[1]

# Split RatingDistTotal column into just the count column
countRatingTotal = eng_bookDF['RatingDistTotal'].str.split(":", n=1, expand = True)

# Creating a column for the counts for Rating Total
eng_bookDF['RatingTotal'] = countRatingTotal[1]

# Remove columns that are not going to be used
eng_bookDF = eng_bookDF.drop(columns=['PublishMonth', 'ISBN', 'Publisher','RatingDist1', 'RatingDist2',
                                      'RatingDist3','RatingDist4','RatingDist5','RatingDistTotal'], axis = 1)

# Rename "PublishDay" to "PublishMonth"
eng_bookDF = eng_bookDF.rename(columns={'PublishDay': 'PublishMonth'})

# Convert some attributes to numeric
eng_bookDF['Rating1Count'] = pd.to_numeric(eng_bookDF['Rating1Count'])
eng_bookDF['Rating2Count'] = pd.to_numeric(eng_bookDF['Rating2Count'])
eng_bookDF['Rating3Count'] = pd.to_numeric(eng_bookDF['Rating3Count'])
eng_bookDF['Rating4Count'] = pd.to_numeric(eng_bookDF['Rating4Count'])
eng_bookDF['Rating5Count'] = pd.to_numeric(eng_bookDF['Rating5Count'])
eng_bookDF['RatingTotal'] = pd.to_numeric(eng_bookDF['RatingTotal'])

# Display column names and datatypes for those columns
eng_bookDF.info()
```

```
Data columns (total 15 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Id              15988 non-null  int64
 1   Name            15988 non-null  object
 2   pagesNumber     15988 non-null  int64
 3   PublishMonth    15988 non-null  int64
 4   CountsOfReview  15988 non-null  int64
 5   PublishYear     15988 non-null  int64
 6   Language        15988 non-null  object
 7   Authors         15988 non-null  object
 8   Rating          15988 non-null  float64
 9   Rating1Count    15988 non-null  int64
 10  Rating2Count    15988 non-null  int64
 11  Rating3Count    15988 non-null  int64
 12  Rating4Count    15988 non-null  int64
 13  Rating5Count    15988 non-null  int64
 14  RatingTotal     15988 non-null  int64
dtypes: float64(1), int64(11), object(3)
```

Research Questions: **it should be noted that the questions below were adjusted from the project proposal with the change in datasets*

1. Rating counts:
    a. For each "star" rating, what is the percentage of total ratings?
    b. What does the rating distribution look like?
    c. What are the top 20 most rated books?
    d. Does the page count affect rating counts?
2. Average rating:
    a. What are the top 20 most rated authors?
    b. Does page count affect average rating?
    c. Does the number of text reviews affect the average rating?
3. Twitter Data
    a. # Books

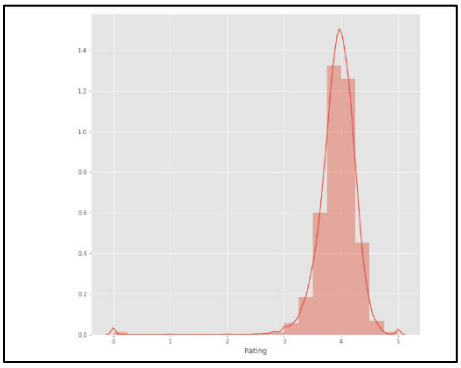
Data Analysis

Ratings Count Data

On Goodreads books are rated (usually after they are read) with 1 to 5 stars (5 being the best). The first question to be answered was which percentage due each of the star values make up the total count of ratings. It was a surprise that the percentages came out the way they did. It appears that 4 and 5 star ratings happen more often than 1, 2 and 3 star ratings combined. It was expected that 3 maybe 4 star ratings made up the bulk of the total. Perhaps that expectation comes from writer's own Goodreads experience.

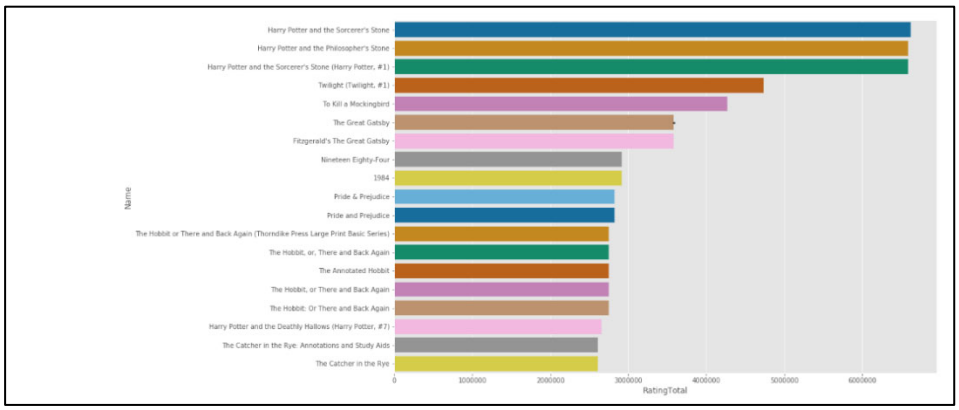
```
In [67]:
# Calculate the percentage of star ratings to total ratings
percentage1 = round(eng_bookDF['Rating1Count'].sum() / eng_bookDF['RatingTotal'].sum(), 2)
print('Percent of 1 Star Ratings to Total Rating Counts: '+str(percentage1)+'%')

percentage2 = round(eng_bookDF['Rating2Count'].sum() / eng_bookDF['RatingTotal'].sum(), 2)
print('Percent of 2 Star Ratings to Total Rating Counts: '+str(percentage2)+'%')

percentage3 = round(eng_bookDF['Rating3Count'].sum() / eng_bookDF['RatingTotal'].sum(), 2)
print('Percent of 3 Star Ratings to Total Rating Counts: '+str(percentage3)+'%')

percentage4 = round(eng_bookDF['Rating4Count'].sum() / eng_bookDF['RatingTotal'].sum(), 2)
print('Percent of 4 Star Ratings to Total Rating Counts: '+str(percentage4)+'%')

percentage5 = round(eng_bookDF['Rating5Count'].sum() / eng_bookDF['RatingTotal'].sum(), 2)
print('Percent of 5 Star Ratings to Total Rating Counts: '+str(percentage5)+'%')

Percent of 1 Star Ratings to Total Rating Counts: 0.02%
Percent of 2 Star Ratings to Total Rating Counts: 0.05%
Percent of 3 Star Ratings to Total Rating Counts: 0.18%
Percent of 4 Star Ratings to Total Rating Counts: 0.33%
Percent of 5 Star Ratings to Total Rating Counts: 0.42%
```

This data was also verified with the distribution plot. This plot heavily skews to the 3-5 star rating range. It looks like readers are more likely to give a book a middle to high rating. It's possible that people find it easier to discern their own labels for 3, 4 and 5; with 3 being average and 5 being amazing. Or people don't bother rating what they deem a not great book. There's also an assumption that these are books people have read. So if a reader doesn't finish a 'bad' book – it may not get rated as such.

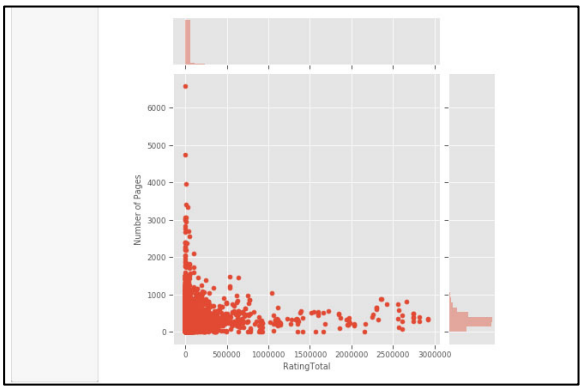


It came as no surprise that some of the top 20 books were from the Harry Potter series given how many of the books have been sold worldwide (500 million+). It was also interesting to see how many classic books were listed here: To Kill a Mockingbird, The Great Gatsby, 1984, Pride & Prejudice to name a few. It should be noted that there are duplicates here. This is due to the many varied editions of these classic books. Humans know that "The Great Gatsby" and "Fitzgerald's The Great Gatsby" are the same book. However, unless the software is built to discern all of the variability in all of the editions of the books (and it's not clear how one could do that at least to this writer); it would be difficult to manage.

Especially since each edition comes with it's own cover, publication date, possibly a slight change in the title ("Fitzgerald's The Great Gatsby); and quite possibly a different publisher.

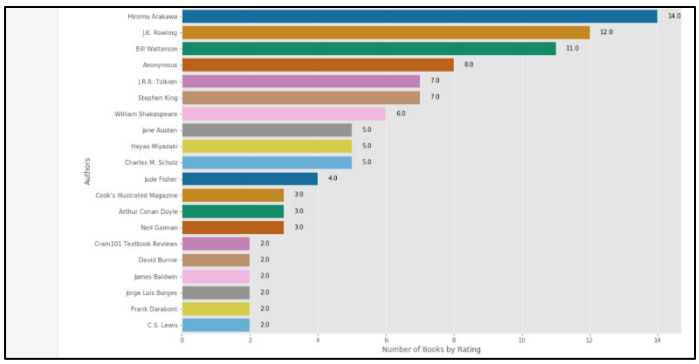


Page count affect on Rating Totals was difficult to discern; perhaps with more skill and/or time, this could be an interesting topic to spend more time. Especially since "large books" are always popular – for the simple reason that they "take too long to read".

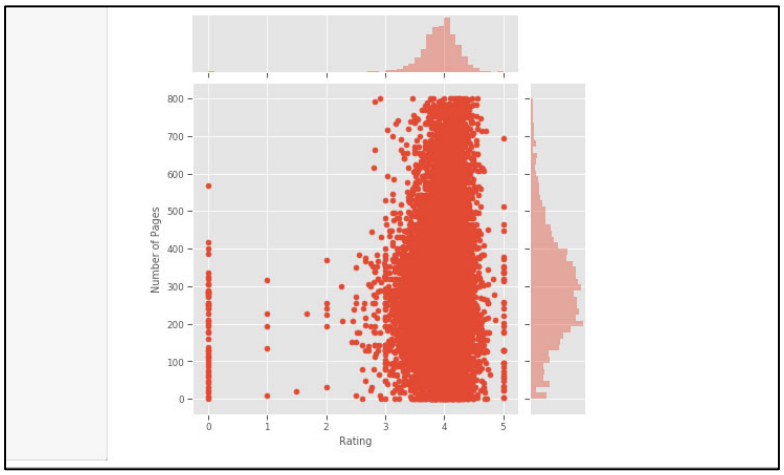


Average Rating Data

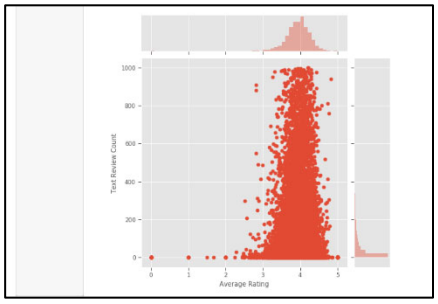
The top 20 authors list does not align perfectly with the top-rated books. It is interesting to note the breadth of authors that are represented. It could be determined that the authors ratings are skewed by the many different editions of their books.



Attempting to look at page count data and whether that impacts average ratings. Tried to limit the outliers so kept the page number to 800 (which is why the graph looks a bit truncated). This data is in line with the other pieces of data that we've looked at. That an average rating between 3 – 5 is where most books are rated, page count aside. It is interesting to note here that there seems to be a quite a bit of smaller (>100 pages) that are given a 5-star rating.

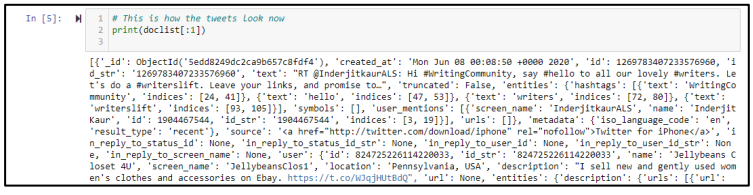


The next question that needed to be answered was if there was a link between the number of text reviews and ratings. After starting out with no limitations on the number of Reviews, kept changing the parameter to be less and less until finally stopping at 1,000 reviews. It does show that there are more books with 0-200 reviews that are still being rated between 3-5 stars.



Twitter Data based on #books

4,000 tweets related to the #books were captured from Twitter. The tweets were stored in a database named books_project in a collection called book_tweets.



```
In [5]:
# This is how the tweets look now
print(doclist[:1])

[{'_id': ObjectId('5edd8249dc2ca9b657c8fdf4'), 'created_at': 'Mon Jun 08 00:08:50 +0000 2020', 'id': 1269783407233576960, 'id_str': '1269783407233576960', 'text': "RT @InderjitkaurALS: Hi #WritingCommunity, say #hello to all our lovely #writers. Let's do a #writerslift. Leave your links, and promise to…", 'truncated': False, 'entities': {'hashtags': [{'text': 'WritingCommunity', 'indices': [24, 41]}, {'text': 'hello', 'indices': [47, 53]}, {'text': 'writers', 'indices': [72, 80]}, {'text': 'writerslift', 'indices': [93, 105]}], 'symbols': [], 'user_mentions': [{'screen_name': 'InderjitkaurALS', 'name': 'Inderjit Kaur', 'id': 1904467544, 'id_str': '1904467544', 'indices': [3, 19]}], 'urls': []}, 'metadata': {'iso_language_code': 'en', 'result_type': 'recent'}, 'source': '<a href="http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone</a>', 'in_reply_to_status_id': None, 'in_reply_to_status_id_str': None, 'in_reply_to_user_id': None, 'in_reply_to_user_id_str': None, 'in_reply_to_screen_name': None, 'user': {'id': 824725226114220033, 'id_str': '824725226114220033', 'name': 'Jellybeans Closet 4U', 'screen_name': 'JellybeansClos1', 'location': 'Pennsylvania, USA', 'description': "I sell new and gently used women's clothes and accessories on Ebay. https://t.co/WJqjHUtBdQ", 'url': None, 'entities': {'description': {'urls': [{'url':
```

Started by looking at how many of the tweets were retweeted. 86% of the tweets collected were retweeted 50 times or less. Only 311 tweets had been retweeted more than 100 times.

```
In [15]:
# Looking at retweet counts; displaying the information and comparing that information
def FilterByGreaterThanRetweet(tweets, nNumberOfRetweets ):
    lTweets = []
    for tweet in tweets:
        if tweet['retweet_count'] > nNumberOfRetweets:
            lTweets.append(tweet)
    return lTweets
def FilterByLessThanRetweet(tweets, nNumberOfRetweets ):
    lTweets = []
    for tweet in tweets:
        if tweet['retweet_count'] < nNumberOfRetweets:
            lTweets.append(tweet)
    return lTweets
lTweetsLessThanTen        = FilterByLessThanRetweet(doclist, 50)
lTweetsGreaterThanHundred = FilterByGreaterThanRetweet(doclist, 100)

print("Number of tweets that have less than 50  retweets:",len(lTweetsLessThanTen))
print("Number of tweets that have more than 100 retweets:",len(lTweetsGreaterThanHundred))

Number of tweets that have less than 50  retweets: 3445
Number of tweets that have more than 100 retweets: 311
```

Looking at some of the tweets that were not retweeted and some of the tweets that were tweeted more than 100 times, it's hard to determine what makes a tweet "retweet-able". It could be the number of followers; commercial tweets from well-known authors, book-stores or publishers could garner more retweets than the average Twitter user.

```
In [17]:
# Display some of the tweets with info about date, user, message, and number of retweets.
def print_tweet_data(tweets):
    for tweet in tweets:
        print('\nDate:', tweet['created_at'])
        print('From:', tweet['user']['name'])
        print('Message', tweet['text'])
        print('Number of Retweets:',tweet['retweet_count'])

# Display some of the tweets with info about date, user, message, and number of retweets.
print_tweet_data(lTweetsLessThanTen[:10])

Date: Mon Jun 08 00:08:50 +0000 2020
From: Jellybeans Closet 4U
Message RT @InderjitkaurALS: Hi #WritingCommunity, say #hello to all our lovely #writers. Let's do a #writerslift. Leave you
r links, and promise to…
Number of Retweets: 8

Date: Mon Jun 08 00:08:49 +0000 2020
From: Edgar Alvarado13 e.v.
Message RT @novelicious: We live for #books.
― Umberto Eco
#writing #amwriting #reading #bookslover #love #books
#Art Zirine https://t.co/ng2kkkwey…
Number of Retweets: 19
```

```
In [18]:
# Display some of the tweets with info about date, user, message, and number of retweets.
print_tweet_data(lTweetsGreaterThanHundred[:10])

Date: Mon Jun 08 00:08:11 +0000 2020
From: Jen
Message RT @StevenLenton: Celebrating over a million book sales (how did that happen?!) with a lovely giveaway! Simply follo
w, retweet and be in th…
Number of Retweets: 1006

Date: Mon Jun 08 00:04:35 +0000 2020
From: Valery Sobolev
Message RT @judehaste_write: "An Adrenalin Rush"
#StayPositive_StaySafe
Let the books do the walking #escapism
🌈USA &amp;amp; UK 🌈 https://t.co/78iawh5K3…
Number of Retweets: 536

Date: Mon Jun 08 00:04:24 +0000 2020
From: Valery Sobolev
Message RT @judehaste_write: #worldbookday2020 thanks to all the #authors #artists who support and share our works. https://
t.co/7ewHDOHE2P
🌟 #Re…
Number of Retweets: 399
```

Tweets can sometimes be hard to cull through even with a unifying hashtag such as #books. There are many times that a hashtag will get used because it is popular; even if the tweet doesn't really relate to the hashtag. For example look at the list of hashtags below that were often included with the original

```
# print out the top number of words with frequencies
# go through the first 20 tweets and find the entities
print("Top", limit, "Frequency Hashtags")
for (word, frequency) in hashtags_sorted[:limit]:
        print (word, frequency)

usage: python twitter_hashtags.py <DBname> <DBcollection> <number>
Top 20 Frequency Hashtags
books 955
Books 433
DataScience 286
BigData 280
Analytics 260
IoT 225
IIoT 224
Python 186
Kindle 171
IARTG 157
PyTorch 129
RStats 124
Amazon 120
reading 117
book 109
WritingCommunity 100
writerslift 91
amreading 80
NLProc 80
ebook 72
```

#books. As the frequency of hashtags with “DataScience”, “BigData”, “Analytics” are listed here, their inclusion with the #books hashtag appears to be an odd combination for the frequency.

Conclusion

Goodreads is a great source of data around books, users, authors, publishers, etc. The drawback with the data is that there are multiple editions in a variety of languages, publications, page numbers etc. Determining what data you want to work with is important; even more important is determining which data questions you’d like to ask before doing any of the work.

The data itself was more clean than other datasets might have been. It’s important with this Goodreads data to never forget about the edition issue. There may be a resolution out there that I’m not aware of. The other ‘main’ issue is how this dataset originally defined the month a book was publisher – when it was clearly the day it was publisher not the month.

It was interesting to see how many readers rate books in the 3-star to 5-star range. All of the data manipulation around rating counts; and average ratings all points to a conclusion that is readers don’t seem like giving books a low rating.

In today’s world, where anonymity behind a screen seems to have fostered a pattern of behavior that allows people to say/do whatever they feel like behind that veil of anonymity. It does not appear to seem that way with the ratings on Goodreads. Quite frankly, it was a real surprise to see such a low number of 1-star, and 2-star ratings. It would be interesting to narrow down the sample and take a closer look at the demographics of the data. Perhaps cull out some review text to analyze to see if the text reviews support the star ratings.

Twitter book data can be overwhelming, especially with how often book-related hashtags are used. With the brief look here at the Twitter data, it would seem hard to use these tweets to determine book popularity, unless you knew the specific book title. The hashtag data while interesting doesn’t tell us as much as we weren’t thinking it might. Book recommendations might also be hard to determine by using hashtag data. More time/skill with working with Twitter data might reveal more interesting information that isn’t visible at this time.