What is a Goodread?

Anurag Gupta
New York University
ag7662@nyu.edu

Arnab Arup Gupta
New York University
ag7654@nyu.edu

Jaimin Dineshbhai Khanderia
New York University
jk7178@nyu.edu

Jaya Mundra
New York University
jm8834@nyu.edu

Yang Tang
New York University
ty@cs.nyu.edu

***Abstract*—Recommendation systems are used by pretty much every major company to enhance the quality of their services. A book recommendation system can be used by any bibliophile to discover books or by organizations to advertise similar books for a user to buy. It is observed that users tend to read/buy books similar to the ones they have already read. We have designed a system that would recommend books in both a content-based filtering manner as well as a collaborative filtering manner by taking into account a user's own taste, similar users' tastes and the sentiment from users' reviews. We have worked with Spark on Zeppelin and produced some Top - k analytics, incorporated sentiment analysis and used these as building blocks in our recommendation systems.**

***Index Terms*—big data, goodreads, spark, recommendation system**

I. Introduction

Recommendation systems exploit user preferences and traits to prioritize and recommend items that the users would like. Recommendations systems are a huge value-addition for large companies like Google, Amazon, Facebook, Netflix as they drive significant customer engagement and revenue. Recommendation systems not only take advantage of users by tempting them to buy more products and services customized to their tastes but also keep them engaged for a longer time to show them more ads and get more clients. A book recommendation system is a type of recommendation system where we have to recommend similar books to the reader based on his interest. The book recommendation system is being used by online websites which provide e-books like google play books, open library, Goodreads.

We aim to build a simple book recommendation system using the reviews of users. We have a large dataset that was collected in late 2017 from goodreads.com which we can process using Spark and store in HDFS. We first discuss the methods in Section IV that we are using to build a robust recommendation system. We discuss the design for our data flow and the results of data cleaning and profiling. In section VI, we illustrate the algorithms and results we achieved in detail.

We analyze the datasets to find the top K analytics, sentiment analysis of user reviews and lastly book recommendations. We have generated book recommendations in four different ways which are explained in detail in Section IV-B. Our work incorporated Sentiment Analysis that when applied to reviews in our data, generates a score representative of the sentiment of the comment, to aid and increase the veracity of our results. We believe that this cross-domain analytic borrowing models from Natural Language Processing brings out the humanness of our data and make our models more accurate to providing, semantically, more accurate suggestions.

II. Motivation

The data collected from Goodreads provides a lot of information about authors, books and their reviews by users, and user's books list. This information can be used by multiple stakeholders for their diverse uses. As the dataset contains information about users' interests in books and their reviews on various genres, book retailers can use the information generated by this project to advertise books to appropriate customers. They can also maintain appropriate stockpiles by finding the popular books among users, so their business isn't affected by the popularity of certain books. Additionally, retailers can use this information to advertise and recommend books belonging to new genres to increase their sales.

Apart from book retailers, the data also greatly benefits the user. Before choosing any book, the user can take the advantage of this large dataset to get the views of the community on the book. They can get the ratings of the book, most positive and negative reviews by sentiment and upvote/downvote count. Using this information, the user can easily pick up a new book or explore books in a new genre. The user can also take advantage of the four different recommendation systems built by us to discover new books, which they have a higher chance of liking. The user can get recommendations based on the books they liked, based on users similar to themselves, based on authors they like or based on the reviews written by them. Apart from all the benefits mentioned, the user also gains the power of a large reading community, which can facilitate the user to get the currently popular genre or books.

Lastly, our analytics will also benefit authors. They can review the ongoing trend among the readers and retailers. The authors can find the popular genre and top books amongst users and try to write new books according to the current ongoing trend. Therefore, our analytics benefits various stakeholders in a multitude of ways as described above.

III. Experimental Setup

A. Platform

All the work has been done in Scala using Spark RDDs and DataFrames through the Zeppelin notebook interface.

B. Data Sources

The datasets we used for recommendations were collected in late 2017 from Goodreads.com [1], where users' public shelves were scraped and the User IDs and review IDs were anonymized. We have used these datasets for academic use only.

The books, authors, and reviews datasets were in JSON format and users' public shelves dataset was in CSV format, totaling the overall dataset size to 70GB. We cleaned and profiled all these datasets individually using Spark through Zeppelin as mentioned in the previous sub-section III-A.

C. Data Cleaning and Profiling

The given section talks about the cleaning and profiling that was performed on all the datasets. We carried out two different forms of data profiling for all the datasets that are explained below:

- Count the number of records per category: In this profiling, we counted the number of records for each value in each column, i.e., we formed a dictionary with the key being a specific value in the column and the value being the number of times it appears in that column.
- Count the average for a particular field: We counted the average number of records for a particular column in this profiling, i.e. we produce the average statistic for a field that will give us a measure of central tendency for that column in the books dataset.

1) ***Books Dataset***:

Description - The dataset contains the metadata for the Goodreads books. The book dataset comprises data of 939194 books (8.6 GB). The metadata consists of keys like Book ID, Language code, Author ID, Title, etc.

Data cleaning - During the data cleaning phase, the following fields: *isbn, series, asin, kindle_asin, format, link, publication_day, isbn13, publication_month, url, image_url, text_reviews_count, work_id, edition_information, publication_year* were dropped as they were not required for the analysis. The fields *authors, popular_shelves and similar_books* were nested substructures and so we exploded those columns to extract the normalized version of those fields. The dataset was then subjected to null value analysis and the default values were set. The following block lists down the schema of the cleaned books dataset:

```
root
 |-- book_id: string
 |-- title: string
 |-- title_without_series: string
 |-- average_rating: double
 |-- description: string
 |-- num_pages: integer
 |-- ratings_count: integer
 |-- author_ids: array
 |    |-- element: string
 |-- publisher: string
 |-- popular_shelves_genre: array
 |    |-- element: string
 |-- is_ebook: boolean
 |-- similar_books: array
 |    |-- element: string
 |-- country_code: string
 |-- language_code: string
 |-- genres: array
 |    |-- element: string
```

Data profiling - The following are the some of the results from the data profiling that was carried out on the cleaned books dataset:

- Total number of books = $\sim$ 9M
- Total number of ebooks out of all books = $\sim$ 678K
- Average length of description of book = $\sim$ 695 letters
- Average number of pages in a book = $\sim$ 178
- Average number of books that were similar to a book = 12

2) ***User-Book Interactions***:

Description - The dataset comprises of $\sim$229M interactions ($\sim$ 4.1GB) containing information related to the interaction between a user and a book. This includes whether the user has read the book on Goodreads or not, the rating given by the user to the book, the dates the ratings had been given and last updated as well the dates on which the user read the book.

Data cleaning - The dataset includes three files - goodreads_interactions.csv, book_id_map.csv and user_id_map.csv. For the data cleaning phase, the data from the files were joined and merged. The fields *user_id, book_id* were dropped from goodreads_interactions.csv and user_id was replaced with user_id from user_id_map.csv and book_id was replaced from book_id in book_id_map.csv. The records with null value in user_id, review_id and book_id were dropped because they do not add value to our analysis. The cleaned data sample for the interactions dataset looks as shown below:

```
root
 |-- user_id: string
 |-- book_id: integer
 |-- is_read: integer
 |-- rating: integer
 |-- is_reviewed: integer
```

Data profiling - Following were the results obtained from profiling the cleaned data:

- Computed total number of reviews/ratings given by a user using user_id and ratings fields
- Calculated total number of reviews/ratings for a book using book_id and ratings fields
- Computed average rating of a given book with book_id and rating

- Calculated total number of books read by a user using user_id and is_read
- Counted the number of books in each range of the ratings

*3) **Reviews**:*

Description - The reviews dataset has complete multilingual review text but without spoiler tags. This dataset is relatively large and contains more than 15.7M reviews of ~2M books.

Data cleaning - For the data cleaning phase, we dropped the columns that were not required for the prescriptive analytics. The fields *user_id*, *book_id*, or *review_id* were checked for null values, and records with missing values were dropped. The default values for the other columns that need to be kept were set. The schema for review data after the cleaning looks as shown here.

```
root
 |-- book_id: string
 |-- user_id: string
 |-- review_id: string
 |-- review_text: string
 |-- rating: integer
 |-- n_votes: long
 |-- n_comments: long
 |-- read_at: timestamp
 |-- started_at: timestamp
 |-- date_modified: timestamp
```

Data profiling - Some of the results obtained from profiling the clean data are as follows:

- The average length of review text = 703 characters
- Total count of reviews = ~ 15.7M
- Total count of books with reviews = ~ 2M.
- Calculated the average rating for each book using the rating and book_id.
- Calculated average rating given by each user
- Count the total number of reviews for each book
- Calculated the count of reviews given by each user
- Counted the books for each rating

*4) **Authors**:*

Description - The authors data set contains the details about the authors of the books and contains details of their average rating, number of ratings and count of text reviews. The size of the data set is around 102MB.

Data cleaning - For the data cleaning phase, no columns were dropped from the Authors data set. Additionally, by joining this data set with the one containing information about books, we have added a few attributes that represent certain important characteristics of an author. This includes the author's book count and the first and last years they published books in. The cleaned and augmented data sample for the authors data set looks as shown below:

```
root
 |-- author_id: string
 |-- average_rating: double
 |-- name: string
 |-- ratings_count: long
 |-- text_reviews_count: long
 |-- books_count: long
 |-- first_published_in: integer
 |-- last_published_in: integer
```

Data profiling - Some profiling was carried out to compute the following results:

- Total count of authors = ~829K
- Average rating distribution: 4 star $\sim 70\%$, 3 star $\sim 15\%$, 5 star $\sim 10\%$ and the rest are 1 and 2 stars.
- First published in: most authors in our data set first published after 1990 with an exponential rise since 1958.
- Book count distribution: The majority of authors have written 1 or 2 books with very few with number of books authored greater than 5. Some authors have authored as many books as 134, which is probably multiple listings of the same original work in different distributions.

These results were included in the data ingestion report and are left out of this paper for the sake of brevity.

D. Data Ingestion

The data ingestion step, as shown in design diagram 1, is explained in this section. All of the cleansed datasets were stored in the Parquet format on HDFS during this phase for the following advantages:

- Good for storing big data of any kind (structured data tables, images, videos, documents).
- Saves on cloud storage space by using highly efficient column-wise compression, and flexible encoding schemes for columns with different data types.
- Increased data throughput and performance using techniques like data skipping, whereby queries that fetch specific column values need not read the entire row of data.

IV. Methodology

A. Data design

Figure 1 shows the overall design of our implementation of the project "What is a goodread?". The following is the sequence of phases that occur in our design:

1) *Data Collection*: We first collect the data for the Goodreads books and reviews which is discussed in detail in section III-B.
2) *Data Cleaning*: The data is cleaned through various operations like dropping unnecessary fields, and null values analysis which is described in detail in the section III-C.
3) The cleaned dataset was then used in two different phases which are described as follows:
    a) *Data Profiling*: All the cleaned datasets were then profiled to gather some useful statistics related to the dataset which is described in more detail in the section III-C.

b) *Data Ingestion*: The cleaned datasets were then stored on HDFS in Parquet format which is discussed in the section III-D.

4) *Analytics*: The final application then generates different analytics based on the user id given as input. The details about the different analytics and the results of those analytics are discussed in the section IV-B and VI respectively.

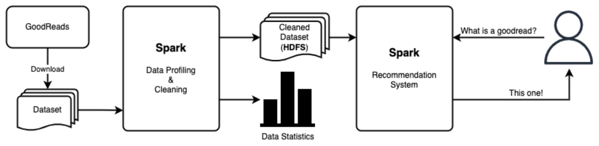


Fig. 1. Data design

B. Models

This section describes the four models of our recommendation system and argues the goodness of their ideologies.

User's Taste: This model captures a user's likes and tries to model their taste, as well as the similarity of books, provided by the data set itself, in recommending them new books to read. Given a user (user_id) X, we first compute a list (L_X) of books they gave four or five stars to from the reviews data set. We then use the books data set to lookup these books and extract books similar to each of these books (S_X). Lastly, we filter these books by their rating count and then sort them by their average rating.

We believe this is a good estimation of their taste since we look at a user's own ratings of books, and Goodreads' similarity measure is well-reputed and can be trusted. We also filter the number of reviews given to the books to make sure we do not include lesser-known books. Additionally, we apply sorting by average rating to show the best of these similar books to the user.

Similar Users' Tastes: This model, along with a user's taste, also captures other user's tastes and presents recommendations based on a similarity function between our user and others based on their interactions with a common set of books. Like before, we get a list of books (L_X) a user X rated highly. This time around we find other users who also gave a high rating to these books, say (U_X). These users can be said to have similar tastes to X. Now, we refer to the reviews data set and find a list of books that were given high ratings by users U_X. Finally, this list is filtered on the number of reviews given to books and sorted in descending order by their average rating.

The goodness of this analysis follows from the authenticity of ratings provided by users and the number of users that find these books to be interesting since we filter on the count of reviews. The sort function provides a robust ordering in exploring this set of book recommendations.

Favorite Authors: For this model, we compute L_X as before, and then compute a list of authors of these books (A_X). Using this list, we filter the books data set to find works written by these authors liked by the user, call it B_{A_X}. We then perform filtering on the rating count and sorting by the average rating of books in B_{A_X} as before to make the quality of results more robust.

This model accounts for the characteristics of an author, assuming that an author's works are characterized by their way of writing, their way of unfolding the plot, the references they use, their quality of English and these are some of the many factors that draw readers to admire authors for their works. While each book is unique, books by the same author more often than not are similar in a manner that cannot be presented by any metric. This model is included to account for those similarities.

User's Sentiments: This model is similar to the first in that it looks at a user's own opinions, their taste from how they felt about certain books. The major difference is that, while accounting for their sentiments we look at the review text they wrote, allowing us to account for natural language information. A user would not care too much about the difference between a 4 and a 5 star rating but their feedback written in the natural language would be a well-thought statement they make about the book, and so, in many ways, provide a higher quality representation of their taste.

For this, we refer to review text and conduct sentiment analysis to compute a sentiment score for those reviews. We then sort the reviews by their sentiment score in descending order and take the first ten books with the highest sentiment scores. Following this step, we lookup these books in the books data set and make a list of books similar to them. Lastly, as always, we filter these books by their review count and sort by their average rating.

C. Sentiment Analysis

In any application that works with extensive sets of user-generated data, one contemplates the assistance of Natural Language Processing to generate metrics understandable by machines to enable our models to incorporate the true essence of the user-generated data. We have ventured so far as to make use of a fairly intuitive approach towards **Sentiment Analysis** on the text data found in the reviews data set. We have referred to the AFINN lexicon [2], a list of English words compiled and manually rated by Finn Årup Nielsen with an integer score in the range [-5, 5]. This is a well-established popular data set of words generally used in most cases.

We compute the sentiment score (χ) by utilizing the following formula:

$$\chi(s) = \frac{\sum_i^N \eta(w_i)}{N}$$

where w_i is a word in the sentence s and N is its length in terms of the number of words, and η is a map between a word and its sentiment score in the AFINN dataset [2].

The summation step takes care of all the words in the sentence and the normalization (dividing by the sentence length) takes care of the overflow (underflow) of the score in case someone just repeats the same word multiple times without adding true meaning to the actual comment. While this formula has certain shortcomings which can be overcome as mentioned in the Future Scope of our work, it has a good balance between complexity and veracity of results and is quite appropriate for our purposes of sentiment analysis that deals with short groups of sentences.

V. Coding Challenges

The following were the code challenges that were faced by the team during their course of work:

- Sentiment analysis dataset: We calculated the average sentiment score for the review text written by a user for a certain book using a different dataset that comprised the sentiment score for regularly occurring English words. We ran into a problem when trying to use and access the dataset on other Spark executors because it was stored on the Spark driver. As a result, we had to overcome this problem by using DataFrame joins, which made it easier to retrieve afterwards.
- Nested substructure in dataset: We came across a lot of nested substructures in the datasets, thus we had to extract the appropriate subfields from those fields to build new normalized fields for analytics. We used the 'explode' function to normalize the data to query and join multiple tables for our analytics.

VI. Results and Observations

This section of our paper has been dedicated to discussing the observations that we made and their goodness. The observations have been categorized into three sections as given below.

A. *Top - k Analytics*

Most popular Genres

We first start by using the Books dataset to find the most popular genres. The books dataset contains for each book a list of genres that the book belongs to. We use this list to calculate the number of books that are present in each genre. The top 10 popular genres are represented in Figure 2 along with the percentage of books in each genre.

As we can see from the figure 2, almost 25% of books belong to the fiction genre followed by history and romance genre. Additionally, it is also possible for a book to belong to more than one genre. This distribution provided us with a general description of how our books dataset is divided and helped us in other analytics.

The goodness of the results stems from the methods used to calculate the top genres. Our method increments the count for each genre that a book belongs to. This makes sure that the book is included in all the genres that it can belong. So, the list of books in each genre is not disjoint. The top genres derived by this method are also compared with the genre data of Goodreads and our results hold up.

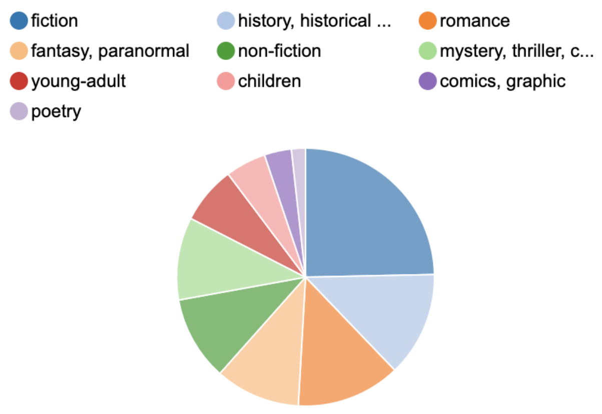


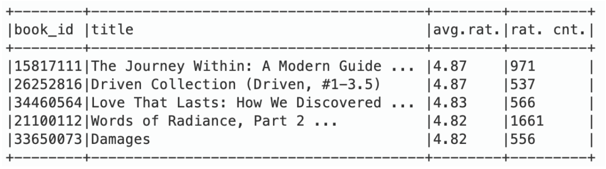
Fig. 2. Most popular genres along with their corresponding percentage

Top Books

We then look at the top books present in the Books dataset. Looking at the most popular books helps both the users and book retailers. The users can explore popular books to look at the current ongoing trends and can explore new books. The retailers can also look at popular books to build their stockpiles and advertise currently trending books to customers.

As we can see from the Fig 3, the table contains the title of the five highest-rated book along with the rating count. The rating count signifies the number of ratings received by the book.

The goodness of the results is ensured by taking into account the number of ratings received by the book along with the average rating. Our method first filters the books dataset, so that the dataset contains only books having at least a given number of ratings. On this filtered dataset of books, we take the top 5 books according to average ratings. This ensures that we take books with the highest average rating which have received enough ratings. The results derived by this method are also compared with the data on Goodreads and it holds up.



```
+--------+---------------------------------------+--------+---------+
|book_id |title                                  |avg.rat.|rat. cnt.|
+--------+---------------------------------------+--------+---------+
|15817111|The Journey Within: A Modern Guide ... |4.87    |971      |
|26252816|Driven Collection (Driven, #1-3.5)     |4.87    |537      |
|34460564|Love That Lasts: How We Discovered ... |4.83    |566      |
|21100112|Words of Radiance, Part 2 ...          |4.82    |1661     |
|33650073|Damages                                |4.82    |556      |
+--------+---------------------------------------+--------+---------+
```

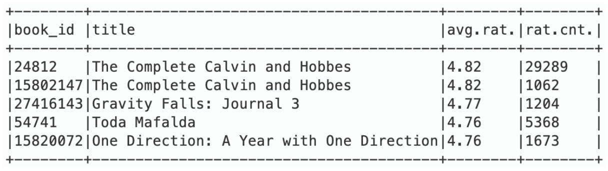
Fig. 3. Top five books

Top Books by Genre

Next, we use the top genres found in the previous section to get the top 5 books in each genre. Both bibliophiles (users) and book retailers will benefit from this analysis. The users will be able to find and explore the highest rated books for the genre they are interested in and our recommendation system will be able to present them with the latest trending books for that genre. Book retailers can use the same data to promote and advertise popular books to particular users in order to boost their interaction.

As we can see from the Fig 4, the table contains the top five books in the 'Graphics-Comics' genre.

The goodness of the results is ensured by taking into account the number of ratings received by the book as well as the average rating of the book. Our method first filters the books dataset to include only books that belong to a particular genre and have at least a given number of ratings. We take the top 5 books based on average ratings from this filtered collection of books. This guarantees that we select books with the highest average rating that have received sufficient ratings. The results derived from this method were compared with the data available on Goodreads and were found to be similar, hence validating the results.



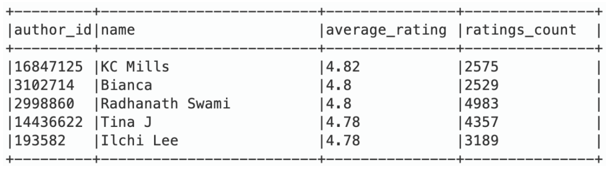
```
+--------+----------------------------------------+--------+--------+
|book_id |title                                   |avg.rat.|rat.cnt.|
+--------+----------------------------------------+--------+--------+
|24812   |The Complete Calvin and Hobbes          |4.82    |29289   |
|15802147|The Complete Calvin and Hobbes          |4.82    |1062    |
|27416143|Gravity Falls: Journal 3                |4.77    |1204    |
|54741   |Toda Mafalda                            |4.76    |5368    |
|15820072|One Direction: A Year with One Direction|4.76    |1673    |
+--------+----------------------------------------+--------+--------+
```

Fig. 4. Top five books in Comics, Graphics Genre

Top Authors

We find the top k-authors which can be used to explore new books by popular authors. To find the top authors, we use the average rating and rating count in the authors dataset. The authors list is sorted based on average rating and filtered with a ratings count greater than a threshold rating count. The 5 shows the results obtained for the top five authors.

The goodness of the results is ensured by taking into account the count of ratings along with the average rating. Our method increments the count for each author to whom a book belongs. This makes sure that the book is included for all the authors as there could be multiple authors for a book. The top authors derived by this method are also compared with the author data of Goodreads and our results hold up.



```
+---------+-------------------------+---------------+---------------+
|author_id|name                     |average_rating |ratings_count  |
+---------+-------------------------+---------------+---------------+
|16847125 |KC Mills                 |4.82           |2575           |
|3102714  |Bianca                   |4.8            |2529           |
|2998860  |Radhanath Swami          |4.8            |4983           |
|14436622 |Tina J                   |4.78           |4357           |
|193582   |Ilchi Lee                |4.78           |3189           |
+---------+-------------------------+---------------+---------------+
```

Fig. 5. Top five authors

Popular genres for a user

The analytics can be used to develop personalized ads based on a user's favorite genre. We use the reviews dataset along with genre data from the books dataset to find the popular genre for a user. We search for all the highly-rated books by a user and count the genres to which the books belong.

The goodness of the results comes from the methods used to calculate the top genres. Our method increases the count for each genre that a book belongs to ensuring that the book is included in all the genres. So, the list of books in each genre is not disjoint. We also ensure to only consider the highly-rated books by the user and not every book that has been rated.

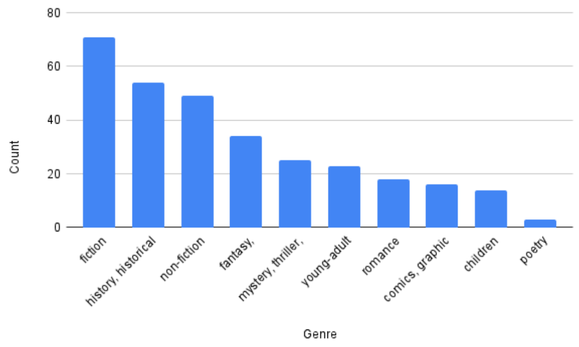


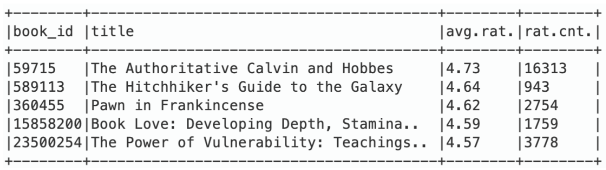
Fig. 6. Most read genres by a user

B. Recommendation Systems

In our project, we have explored four different paradigms of recommending books to a reader. They have some degree of similarity within themselves, as explained before, (Section IV-B). but present different results as shown in the tables (mention table numbers). These results are explained in appropriate detail in this section.

Model: User's Taste

The results are presented in Fig. 7. Referring to the user's top categories (Fig. 6), one can see that the recommendation following this model falls under the user's top categories. The top 2 recommendations fall under 'Fiction' which is the user's favorite genre. The next (3rd) recommendation falls under both fiction and historical fiction, and the last 2 recommendations fall under non-fiction, all of which are among the top 3 genres of this particular user. This is as expected because a list of the user's top categories is an aggregate of their favorite books, and, by extension, their taste.

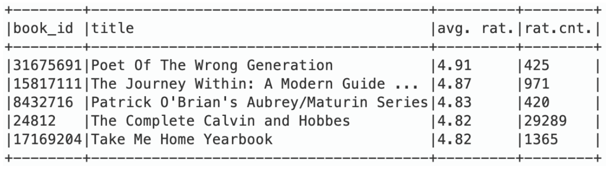


```
+--------+----------------------------------------+--------+--------+
|book_id |title                                   |avg.rat.|rat.cnt.|
+--------+----------------------------------------+--------+--------+
|59715   |The Authoritative Calvin and Hobbes     |4.73    |16313   |
|589113  |The Hitchhiker's Guide to the Galaxy    |4.64    |943     |
|360455  |Pawn in Frankincense                    |4.62    |2754    |
|15858200|Book Love: Developing Depth, Stamina..  |4.59    |1759    |
|23500254|The Power of Vulnerability: Teachings.. |4.57    |3778    |
+--------+----------------------------------------+--------+--------+
```

Fig. 7. Recommendation on user's own taste

Model: Similar Users' Tastes

The recommendations from this model are presented in Fig. 8. In this case, however, we find some variety as far as genres are concerned. One can see that the first recommendation falls under 'Romance' which is only the 7th most common genre for this user. While some others may fall under the user's top 3 categories such as 'Patrick O'Brian's Aubrey/Maturin series' which is a work of historical fiction, these may be attributed to the large size of these genres owing to their generality. In essence, the degree of variance in this recommendation model is higher than the first one because it has one extra edge in a hypothetical graph of connectivity, where the more the edges, the farther the two entities in their similarity.

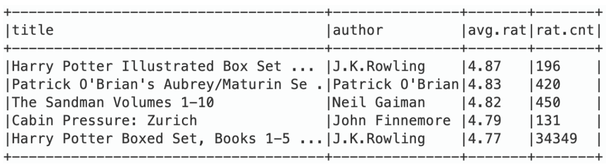


```
+---------+-------------------------------------+---------+--------+
|book_id  |title                                |avg. rat.|rat.cnt.|
+---------+-------------------------------------+---------+--------+
|31675691|Poet Of The Wrong Generation          |4.91     |425     |
|15817111|The Journey Within: A Modern Guide ...|4.87     |971     |
|8432716 |Patrick O'Brian's Aubrey/Maturin Series|4.83    |420     |
|24812   |The Complete Calvin and Hobbes        |4.82     |29289   |
|17169204|Take Me Home Yearbook                 |4.82     |1365    |
+---------+-------------------------------------+---------+--------+
```

Fig. 8. Recommendation on similar users' shelves

Model: Favorite Authors

These results are presented in Fig. 9 where one can find the book title and the author name alongside. From a quick perusal of this graph, one can conclude that our user is a fan of J.K. Rowling's work which is why two of the top 5 recommendations feature works by this author. This model captures author characteristics as explained before, and would most likely feature popular rather than new and upcoming authors. While this model may hinder the discovery of new authors, it features a good reliable metric that includes authors with well-renowned and illustrious works to their name.

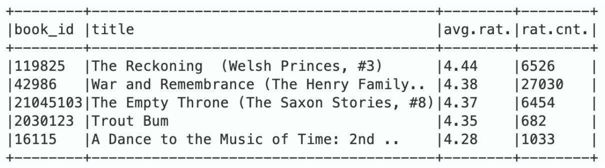


```
+----------------------------------+---------------+-------+-------+
|title                             |author         |avg.rat|rat.cnt|
+----------------------------------+---------------+-------+-------+
|Harry Potter Illustrated Box Set ...|J.K.Rowling    |4.87   |196    |
|Patrick O'Brian's Aubrey/Maturin Se .|Patrick O'Brian|4.83   |420    |
|The Sandman Volumes 1-10          |Neil Gaiman    |4.82   |450    |
|Cabin Pressure: Zurich            |John Finnemore |4.79   |131    |
|Harry Potter Boxed Set, Books 1-5 ...|J.K.Rowling    |4.77   |34349  |
+----------------------------------+---------------+-------+-------+
```

Fig. 9. Recommendation on user's favorite author

Model: User's Sentiments

The results from this recommendation system are presented in Fig. 10. An observation one can make while comparing these recommendations to the others is that the average rating in general for these books are lower in comparison to those provided by other models. This may be attributed to the fact that this list is generated from books which the author wrote excellent reviews for. Now there is a good chance that the author gave a 5 star rating to them, but that is not being captured by the model in providing recommendations. As discussed before, this captures natural language information in the form of the reader's (user's) sentiments, which is usually not garbage (after filtering) and, therefore, has a higher chance of providing more accurate recommendations.



```
+---------+---------------------------------------+---------+--------+
|book_id  |title                                  |avg.rat. |rat.cnt.|
+---------+---------------------------------------+---------+--------+
|119825   |The Reckoning  (Welsh Princes, #3)     |4.44     |6526    |
|42986    |War and Remembrance (The Henry Family..|4.38     |27030   |
|21045103|The Empty Throne (The Saxon Stories, #8)|4.37     |6454    |
|2030123  |Trout Bum                              |4.35     |682     |
|16115    |A Dance to the Music of Time: 2nd ..   |4.28     |1033    |
+---------+---------------------------------------+---------+--------+
```

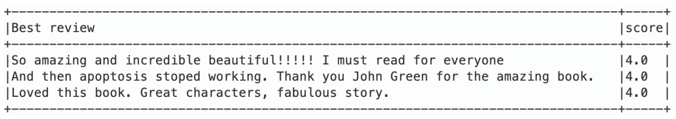
Fig. 10. Recommendation on user's review sentiments

C. Sentiment Analysis

We are partitioning these results into two sections: one which presents the strongest opinions and another which represents the most-common opinions.

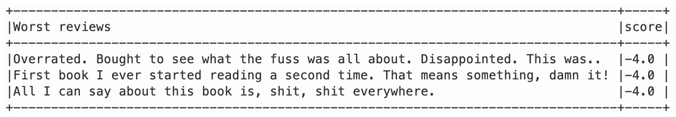
Based on sentiment

These results are shown in Figures 11 and 12 that contain the best and worst reviews for 'Fault in our Stars' by John Green respectively. This is computed by first calculating the sentiment score for each review of the book and then sorting the reviews by their sentiment score: ascending for worst reviews and descending for best reviews. As one can see from reading these review texts in the table, the readers who wrote them to feel very strongly about the book. This result can help readers decide whether they want to read a book or not, since some readers were passionate enough about the book to write such detailed and emotional reviews about them. They also provide an overview of the best and worst parts of the book, some controversial thoughts and what stood out from the rest.



```
+------------------------------------------------------------------------+-----+
|Best review                                                             |score|
+------------------------------------------------------------------------+-----+
|So amazing and incredible beautiful!!!!! I must read for everyone       |4.0  |
|And then apoptosis stoped working. Thank you John Green for the amazing book.  |4.0  |
|Loved this book. Great characters, fabulous story.                      |4.0  |
+------------------------------------------------------------------------+-----+
```

Fig. 11. Best reviews based on review sentiment

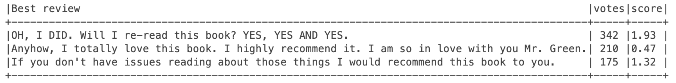


```
+---------------------------------------------------------------------+-----+
|Worst reviews                                                        |score|
+---------------------------------------------------------------------+-----+
|Overrated. Bought to see what the fuss was all about. Disappointed. This was..  |-4.0 |
|First book I ever started reading a second time. That means something, damn it! |-4.0 |
|All I can say about this book is, shit, shit everywhere.             |-4.0 |
+---------------------------------------------------------------------+-----+
```

Fig. 12. Worst reviews based on review sentiment

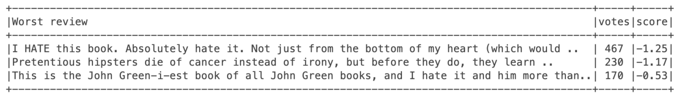
Based on the number of votes

These results are presented in figures 13 and 14 containing the most popular positive and negative reviews respectively for 'Fault in our Stars' by John Green. This is computed by first calculating a sentiment score for each review. Then, a filter is applied to only select the scores above zero for positive review (and below zero for negative review respectively) and they are sorted in descending order by the number of votes they have received from other users. These votes are an indication of how many people second the review author's opinion in what they have written. This metric is highly reliable and can be used by other readers to know how the reading community at large feel about a particular book. To some extent, even authors may refer to these analytical results to as to understand the crowd sentiment regarding their work, which can help them gather useful feedback to help improve their future work. While these opinions are neither very positive or negative, as can be seen in the scores alongside them, they are what many people agree with and helps users know what makes the book popular among readers.



```
+--------------------------------------------------------------------------------------------+-----+-----+
|Best review                                                                                  |votes|score|
+--------------------------------------------------------------------------------------------+-----+-----+
|OH, I DID. Will I re-read this book? YES, YES AND YES.                                       | 342 |1.93 |
|Anyhow, I totally love this book. I highly recommend it. I am so in love with you Mr. Green.| 210 |0.47 |
|If you don't have issues reading about those things I would recommend this book to you.     | 175 |1.32 |
+--------------------------------------------------------------------------------------------+-----+-----+
```

Fig. 13. Score of best reviews



```
+--------------------------------------------------------------------------------------------+-----+-----+
|Worst review                                                                                |votes|score|
+--------------------------------------------------------------------------------------------+-----+-----+
|I HATE this book. Absolutely hate it. Not just from the bottom of my heart (which would ..  | 467 |-1.25|
|Pretentious hipsters die of cancer instead of irony, but before they do, they learn ..      | 230 |-1.17|
|This is the John Green-i-est book of all John Green books, and I hate it and him more than..| 170 |-0.53|
+--------------------------------------------------------------------------------------------+-----+-----+
```

Fig. 14. Score of worst reviews

VII. Conclusion

We have described how "What is a Goodread?" generates recommendations and provides user's sentiments of the reviews. The application processes and analyzes over 70GB of data using Spark RDDs and DataFrames. We remove unnecessary fields from the data and check for invalid values for columns. We store the cleaned data in Parquet format on HDFS for further profiling and analysis.

The system has multiple use cases for bibliophiles, book retailers and the community. It is beneficial to find out highly-rated books, the most popular books in each genre, top-rated authors, top-rated genres, and user's favorite genres for newsletters. It can also be used to explore new books based on user's ratings.

The application can also be used for personalized advertisement in the form of book recommendations. The system makes recommendations based on four criteria - User's taste, similar users' taste, favorite authors and user's sentiments. The implementation also generates review sentiment analysis to give top useful reviews to users for a book. Information like genre, similar books, ratings and number of ratings, review text, authors from the datasets were employed to find a user's taste and perform sentiment analysis.

While working on this project we have learned the following. Firstly, the importance of population is emphasized in big data analytical results, where the number of people and their opinions determine the results and one has to always look at the overall picture rather than focusing on intricacies and outliers in the data. Secondly, there are many different ways of calculating/presenting / reporting/computing a result, and all of them have certain pros and cons which can only be understood if one looks at the algorithm used. To determine what is best for one's purposes, it is important to look at what parts of the data set are being used. Thirdly, the best way to approach a big data project is to start with profiling and working on the entirety of the data and then splitting it into parts for specific analytical results and applying filters. After these steps, we were able to come up with our 4 different recommendation systems. Lastly, one has to admit any amount of work on a particular big data set is never enough to capture a hundred percent of the results and there's always scope for future work.

In our case, while the application generated recommendations as expected, the queries could be enhanced for improved recommendations. The recommendations can be made more robust by combining the above-mentioned several criteria considered for book recommendations.

Acknowledgements

We thank Professor Yang Tang for his support and guidance during various stages of the project and for the course material that helped us learn various design principles and technologies. We also acknowledge Goodreads and UCSD Book graph, for providing the dataset for our analysis. The NYU HPC was prompt in helping us to share the datasets with the team and solve technical difficulties.

References

[1] DataSet - https://sites.google.com/eng.ucsd.edu/ucsdbookgraph/home

[2] Finn Årup Nielsen "A new ANEW: Evaluation of a word list for sentiment analysis in microblogs". Proceedings of the ESWC2011 Workshop on 'Making Sense of Microposts': Big things come in small packages 718 in CEUR Workshop Proceedings 93-98. 2011 May.

[3] Mengting Wan, Rishabh Misra, Ndapa Nakashole, and Julian McAuley. 2019. Fine-Grained Spoiler Detection from Large-Scale Review Corpora. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 2605–2610, Florence, Italy. Association for Computational Linguistics.

[4] Wan, Mengting, and Julian McAuley. "Item recommendation on monotonic behavior chains." In Proceedings of the 12th ACM conference on recommender systems, pp. 86-94. 2018.

[5] Zaharia, Matei, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, and Ion Stoica. "Spark: Cluster computing with working sets." In 2nd USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 10). 2010.