CSC111 Project Proposal: Book Recommendation System

Jayden Chiola-Nakai, Maria Ma, Kaiwen Zheng, Shaqeel Hazmi Bin Radzifuddin

March 7, 2023

Problem Description and Research Question

- **Overview of Background Knowledge**
  Goodreads is an American social cataloging website founded by Otis and Elizabeth Chandler in 2006 and launched in 2007, aiming to provide a guide for consumers to find books they want to read in the era of the digital age. It is the world's largest website for users to not only discover books but also socialize with other users by establishing a network amongst all registered readers through group activities[1]. In particular, users of Goodreads can browse books that, for example, are newly released or on the featured lists, add books to their personal bookshelves, rate and review books, and get recommendations from Goodreads based on the previous reviews given by the users. In addition, users can engage in discussion boards on various topics in groups. They also have the option of adding other users as "friends," getting access to books on their bookshelves, as well as sharing reviews, posts, and book recommendations with each other[2].

- **Motivation and Goal**
  While Goodreads is a convenient platform that has provided helpful guidance to numerous users to explore books in variety since its launch in 2007, it also has several disadvantages with respect to its recommendation system, which may affect user experience to a certain degree depending on the specific user's preference. To begin with, in order to obtain suggestions for future readings from Goodreads, the user needs to provide ratings to at least 20 books first so that the system learns the user's preference; otherwise, the user must manually explore books by narrowing down the genres in which they are interested. For newcomers to this website or reading in general, this feature can create inconvenience since it is both difficult and time-consuming to narrow the search amongst hundreds and thousands of books even in one genre. Therefore, **our goal is to build a book recommendation system for young adult book readers that can automatically make recommendations based on the preferences data the user provides as well as the books they have already saved, all using the repository of young adult books extracted from Goodreads**. Specifically, we propose a recommendation system that recommends books to new users and those who have already read books or saved interesting books in the system. For the latter, the program will recommend books that, in terms of genre, are similar to those that already appear on their bookshelves.

Computational Plan

- **Data Structures Used**
  The main data structures that serve as the driving force for our recommendation system are the decision tree and graph. Specifically, the recommendation system is separated into the following two subsystems based on the user data:

  - Decision-tree-based recommendation subsystem: this subsystem is used to make recommendations to the users with no account registered and those with an existing account who would like to explore books whose genres are not presented in the user's past reading activities. We model this subsystem with a decision tree, where each path down the decision tree corresponds to the responses provided by the user to a sequence of questions (e.g., the rating system the user prefers*, the genres the user wants to explore, the range of the number of pages the user prefers a book to have, whether the user accepts the E-book version, etc.) that is used to narrow down the potential candidates of recommendation.
  - Graph-based recommendation subsystem: this subsystem is employed when the user has an existing account and wants to get further recommendations based on the user's books stored on their bookshelf.

We model this subsystem using a graph, wherein every vertex represents a book, and every edge connecting two vertices represents that these two books are similar in terms of the genres to which they belong.

- **Description of Datasets** We will use the datasets provided on UCSD Book Graph[5,6] to simulate the behaviour of our recommendation system. Specifically, we will use the following datasets for our simulation purpose:

  1. goodreads_books_young_adult.json.gz
  2. goodreads_interactions_young_adult.json.gz
  3. goodreads_reviews_young_adult.json.gz
  4. goodreads_book_authors.json.gz
  5. goodreads_book_genres_initial.json.gz

  Note that we have chosen young adult books among all genres because the original dataset, with a size of 2GB, describing the complete book graph, contains metadata of approximately 2.36M books. Also, the structure of the datasets we chose is the same as that of the complete dataset, which serves our simulation well.

  1. goodreads_books_young_adult.json.gz
     This file contains detailed meta-data about 93,398 books with the genre of "young adult". The attributes included in this file are as follows:
     - `isbn`: the ISBN of this book
     - `text_reviews_count`: the number of text reviews for this book
     - `series`: the series this book belongs to
     - `country_code`: the country code of this book
     - `language_code`: the language code of this book
     - `popular_shelves`: the popular user shelves on which this book appears
     - `asin`: the ASIN number (Amazon Standard Identification Number) of this book
     - `is_ebook`: whether this book is an E-book
     - `average_rating`: the average rating of this book
     - `kindle_asin`: the Kindle ASIN of this book
     - `similar_books`: a list of IDs of books that users who like the current book also like
     - `description`: the description of this book
     - `format`: the format of this book
     - `link`: the link of this book on Goodreads
     - `authors`: the authors of this book
     - `publisher`: the publisher of this book
     - `num_pages: the number of pages this book has`
     - `publication_day`: the day on which this book was published
     - `isbn13`: the ISBN13 of this book
     - `publication_month`: the month in which this book was published
     - `edition_information`: the edition of this book
     - `publication_year`: the year in which this book was published
     - `url`: the URL of this book on Goodreads
     - `image_url`: the URL of the cover image of this book
     - `book_id`: the ID of this book
     - `ratings_count`: the number of ratings given to this book
     - `work_id`: the ID of the work (the abstract version of this book regardless of editions)
     - `title`: the title of this book
     - `title_without_series`: the title of this book without including its series

     The attributes we will include in our system are {`country_code, language_code, is_ebook, average_rating, similar_books, description, format, link, authors, publisher, num_pages, edition_information, publication_year, url, image_url, book_id, ratings_count, title`}. Every other attribute will be removed for the purpose of our system.

A sample row in this JSON file with the attributes mentioned above removed is as follows:

```
{
"country_code": "US",
"language_code": "",
"is_ebook": "true",
"average_rating": "4.04",
"similar_books": ["519546", "1295074", "21407416"],
"description": "This is the final tale in the bestselling author L.J. Smith's romantic
horrortrilogy. Now, Kaitlyn Fairchild and her friends must put the powerful crystalto the
test--to stop an experiment that has turned one of them into a psychicvampire. Kaitlyn must
choose at last--Rob or Gabriel.",
"format": "",
"link": "https://www.goodreads.com/book/show/12182387-the-passion",
"authors": [{"author_id": "50873", "role": ""}, {"author_id": "232533", "role": "Ubersetzer"}]
"publisher": "",
"num_pages": "",
"edition_information": "",
"publication_year": "",
"url": "https://www.goodreads.com/book/show/12182387-the-passion",
"image_url":
"https://s.gr-assets.com/assets/nophoto/book/111x148-bcc042a9c91a29c1d680899eff700a03.png",
"book_id": "12182387",
"ratings_count": "4",
"title": "The Passion (Dark Visions, #3)"
}
```

2. goodreads_interactions_young_adult.json.gz

   This file contains 34,919,254 user-book interactions where the users review books belonging to the "young adult" genre with the corresponding timestamps.

   The attributes included in this file are as follows:

   - user_id: the ID of the user that reviews this book
   - book_id: the ID of the book reviewed by the user above
   - review_id: the ID of the review given by the user above to this book
   - is_read: whether the user has read this book
   - rating: the rating given by the user above to this book, 0 indicating "not provided"
   - review_text_incomplete: incomplete review text, where an empty string means "not provided"
   - date_added: the date the user reviewed this book
   - date_updated: the date the user updated the review for this book
   - read_at: the date the user read this book
   - started_at: the date the user started reading this book

   The attributes we are going to include in our system are the following: {user_id, book_id, review_id, rating, date_added, date_updated, read_at, started_at}. Every other attribute will be removed for the purpose of our system.

   A sample row in this JSON file with the attributes mentioned above removed is as follows:

```
{
"user_id": "8842281e1d1347389f2ab93d60773d4d",
"book_id": "18667753",
"review_id": "be53fe83a6fc83474052f84692f6e90a",
"rating": 0,
"date_added": "Wed Mar 29 00:12:52 -0700 2017",
"date_updated": "Wed Mar 29 00:12:52 -0700 2017",
"read_at": "",
"started_at": ""
}
```

3. goodreads_reviews_young_adult.json.gz

   This file contains 2,389,900 detailed reviews given by readers of the books with the genre ”young adult.”

   The attributes included in this file are as follows:

   - `user_id`: the ID of the user reviewing this book
   - `book_id`: the ID of this book
   - `review_id`: the ID of the review provided by this user
   - `rating`: the rating given by this user
   - `review_text`: the review text written by this user
   - `date_added`: the date the review was added
   - `date_updated`: the date the review was updated
   - `read_at: the date the user read this book`
   - `started_at: the date the user started this book`
   - `n_votes: the number of votes for this review`
   - `n_comments: the number of comments for this review`

   The attributes we are going to include in our system are the following: {`user_id, book_id, review_id, rating, review_text, date_added, date_updated`}. Every other attribute will be removed for the purpose of our system.

   A sample row in this JSON file with the attributes mentioned above removed is as follows:

```
{ "user_id": "8842281e1d1347389f2ab93d60773d4d",
"book_id": "2767052",
"review_id": "248c011811e945eca861b5c31a549291",
"rating": 5,
"review_text":
"I cracked and finally picked this up. Very enjoyable quick read - couldn't put it down
- it was like crack. \n I'm a bit bothered by the lack of backstory of how Panem and the
Hunger Games come about. It is just kind of explained away in a few paragraphs and we are
left to accept this very strange world where teenagers are pitted into an arena each year
to kill each other? I was expecting it because I've seen Battle Royale, but I would have
appreciated knowing more of the backstory of how the world could have come into such a odd
state. \n I suppose what makes a book like this interesting is thinking about the strategy
of it all. The players are going to be statistically encouraged to band together because
they will last longer that way, but by definition of course any partnership will be broken,
and the drama of how that unfolds is always interesting and full of friendships broken and
betrayal. Each character approached the game in their own way. Some banded together in
larger coalitions, some were loners initially and banded together later. And some were just
loners, like Foxface. A lot depended on your survival skill: could you find food and water
on your own? Self-dependence is highly valued - and of course our hero was strong there.
\n All in all, a fun read, but I feel kind of dirty for having read it.",
"date_added": "Wed Jan 13 13:38:25 -0800 2010",
"date_updated": "Wed Mar 22 11:46:36 -0700 2017"
}
```

4. goodreads_book_authors.json.gz

   This file contains the authors in goodreads_books_young_adult.json.gz.

   The attributes included in this file are as follows:

   - `average_rating`: the average rating of this author
   - `author_id`: the ID of this author
   - `text_reviews_count`: the number of reviews given by this author to other books
   - `name`: the name of this author
   - `ratings_count`: the ratings assigned by this author to other books

   The attributes we are going to include in our system are the following: {`average_rating, author_id, name`}. Every other attribute will be removed for the purpose of our system.

A sample row in this JSON file with the attributes mentioned above removed is as follows:

```
{
"average_rating": "3.98",
"author_id": "604031", "name": "Ronald J. Fields"
}
```

5. goodreads_book_genres_initial.json.gz

   This file contains a fuzzy version of book genres extracted from users' popular shelves with a simple keyword-matching process.

   The attributes included in this file are as follows:

   - `book_id`: the ID of this book
   - `genres`: the genres this book belongs to

   The attributes we are going to include in our system are the following: {book_id, genres}.

   A sample row in this JSON file is as follows:

```
{
'book_id': '7327624',
'genres':
{'fantasy, paranormal': 31, 'fiction': 8, 'mystery, thriller, crime': 1, 'poetry': 1}
}
```

- **Modelling**

  With the datasets and main data structures mentioned above, we first initialize our system where:

  - In the decision-tree-based recommendation subsystem, we initialize this subsystem by generating a complete tree based on a sequence of criteria, then store books characterized by specific features as the leaf of respective paths. Specifically, we categorize each book based on a sequence of criteria, given by $S = [c_1, c_2, \ldots, c_n, b]$, where c_i is a particular criterion that is used to characterize a book and b is the book characterized by criteria $[c_1, c_2, \ldots, c_n]$. We then insert the sequence S into the decision tree, where every entry is the parent node of the next entry in S and b is the book characterized by criteria $[c_1, c_2, \ldots, c_n]$. Since there may be more than one book that has the same sequence of characteristics, we store this collection of books as the leaf of c_n, where b is one of them.
  - In the graph-based recommendation system, we base on the information of similar books in goodreads_books_young_adult.json.gz and construct a graph where books that are similar to each other are connected. Specifically, for each book b with a collection of IDs of similar books C, we construct a graph in which b is adjacent to every book in C.

  To make recommendations based on the decision-tree-based subsystem, we first ask the user a series of questions and trace a particular path down to the corresponding leaf of the decision tree based on the user's responses. Once one of the leaves of the decision tree is reached, a collection of recommended books stored in that leaf will be presented to the user.

  For recommendations made via the graph-based subsystem, we start with a given book and search for those not read by the user and connected to that book. It is possible that a book has multiple layers of connection with other similar books. As we further "expand" away from the books the user reads, we are reaching those less relevant to the user's taste than those that stay "closer" to the user. Because of this, we need to restrain our search for books that are indirectly connected to the user within certain levels. Hence, we define the *relevant factor (RF)* as a pre-set positive integer that is equal to the maximum number of "layers" of books we will explore. Once the relevance factor is reached, we terminate our outward search.

- **User Interaction**

  This program will make use of a GUI (graphical user interface) to allow the user to interact with the book recommendation system. `PyQt6` will be the Python library used to implement the GUI. It will consist of two main parts: the first part, for first-time users, will recommend the user books based on the preferences they enter (done using the decision-tree-based subsystem); the second part, for existing users, will allow the user to browse those books as well as other similar books (retrieved using the graph-based subsystem), save and unsave them in the system, and provide them detailed book information.

- **Libraries Used**

  In our recommendation system, we will use several Python libraries that have yet to be covered in this course so far. In particular, we will use the following libraries:

  - `pandas`: `pandas` is a Python library that provides high-performance data analysis tools for Python programming language[3]. Since our recommendation system is heavily based on building up a repository of information about books and their interactions with the users using the datasets we obtained, improving the efficiency of data manipulations plays a central role in boosting the overall efficiency of our system. To obtain well-organized tabular data repositories, we will use the `pandas.core.frame.DataFrame` data type to accomplish this goal. In particular, we will use `pandas.read_json` or `pandas.read_csv` by passing the file path of the relevant JSON/CSV file as a string into these methods to obtain a `pandas.core.frame.DataFrame` object that organizes the file into a tabular data sheet, wherein the attributes in the original dataset correspond to the columns of the resulting data sheet. Also, since there are attributes removed from the original datasets, we will use the method `pandas.core.frame.DataFrame` to remove the columns that correspond to the attributes to be removed.
  - `PyQt6`: `PyQT6` is a set of Python bindings for Qt v6—which is a set of cross-platform C++ libraries—that allow the creation of a GUI (graphical user interface). It will be used to let the user interact with the book recommendation system, using widgets such as `QLineEdit` (for text entry), `QPushButton` (for button clicks), and `QListWidget` (to display list items) as well as functions such as connect (to add widget functionality), `setFont`/`setFixedSize`/`setStyleSheet` (to add graphical design), and `addWidget`/`setLayout` (to organize the arrangement of the widgets). PyQt6 is useful because it makes the program not only limited to the CLI (command line interface) and increases UX (user experience).
  - `json`, `gzip`, `numpy`: these Python libraries will be used to extract (in the `.json.gz` format) and analyze (e.g. using `numpy.nan` which represents an invalid entry) data.

References

1. “About Goodreads.” Goodreads, Goodreads, https://www.goodreads.com/about/us.
2. “Goodreads.” Wikipedia, Wikimedia Foundation, 24 Feb. 2023, https://en.wikipedia.org/wiki/Goodreads.
3. Mengting Wan, Julian McAuley, ”Item Recommendation on Monotonic Behavior Chains”, in *RecSys’18*.
4. “PyQt6 6.4.2.” PyPI, https://pypi.org/project/PyQt6/.
5. Mengting Wan, Rishabh Misra, Ndapa Nakashole, Julian McAuley, ”Fine-Grained Spoiler Detection from Large-Scale Review Corpora”, in *ACL’19*.
6. ”Pandas Documentation#.” Pandas Documentation - Pandas 1.5.3 Documentation, https://pandas.pydata.org/docs/.