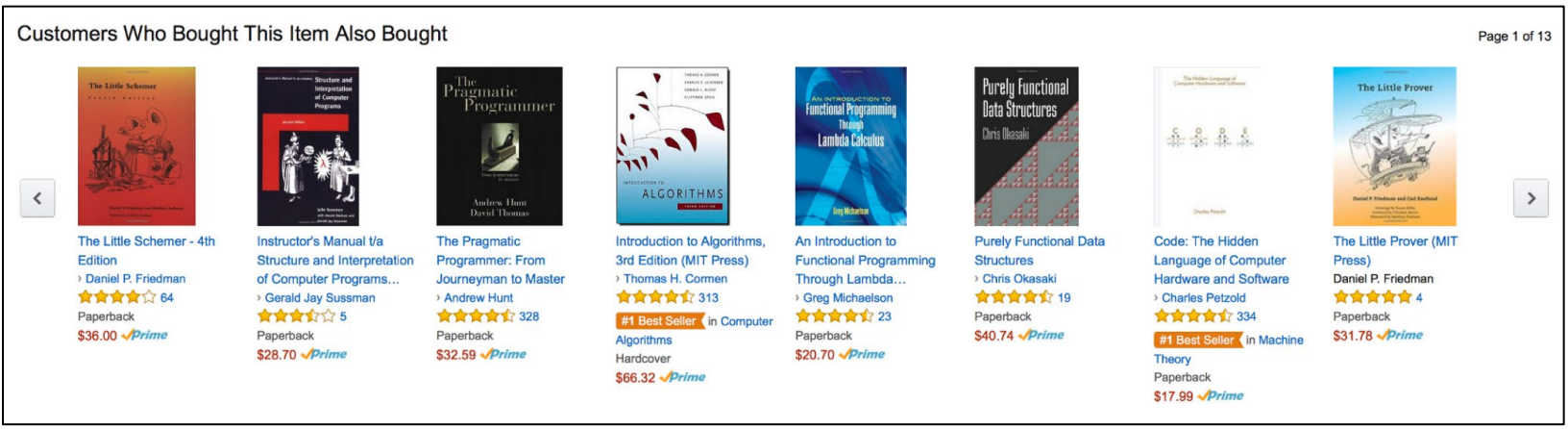
Network Science Project Report

User Rating Based Book Recommendation System

Kushal Hebbar, Nitin Srinath

School of Computing, Department of Industrial Engineering

Clemson University



❖ Abstract

Recommendation systems are a form of information filtering that is based on preferences of the user. The focus of a recommendation system to improve user experience and interaction with the system to increase traffic and in turn increase company revenue. Recommendation systems are used in a range of areas such as music, movies, books, news, restaurants, search engines. User preferences are tracked through user inputs, previous interactions with the system or by accessing web cookies, based on these preferences recommendations are generated to keep the user engaged with the platform.

The goal of our project is to build a recommendation system using a custom algorithm and making use of graph properties to achieve expected outcomes.

The dataset that we are using for experimentation is the BookCrossing dataset.

❖ Introduction

"Customers who bought this item also bought this item", "You may also like…". These are some of the sentences that suggests that a website is using a recommender system to improve your experience and interaction. The dataset that we have used for experimentation is the BookCrossing community dataset that consists of 278858 users and 271360 books. The dataset also consists of 1.1 million ratings that connescts the users to the books. It forms a bipartite graph with users as a column of nodes, books as the other column of nodes which are connected based on the user ratings which acts as the weight of the edge. A higher rating from a user on a book denotes that he/she is more likely to recommend that book to others. The ideology behind our algorithm is "If a person is currently reading a book, and others have read similar books with higher ratings then recommend those books". Books are evaluated based on how similar they are to each other which in turn means that there is a common edge between 2 books in the network with the edge weight 2 denoting that two users have read and recommend that book and hence they have similar tastes.

For further accurate recommendations we make use of the users personal reading list and find similar books based on the average ratings. Our algorithm creates a subgraph where the edge weights are the number of users who have common books on their personal reading lists and the node weight is the average rating of the book. We then implement two

methods called “common neighbors method” and “common path method” which will be explained in detail in the Algorithm section of the report.

❖ Dataset and Preprocessing

The dataset that we are using is the BookCrossing user rating dataset from the BookCrossing community. It is a bipartite network that consists of 278858 users, 271360 books and 1.1 million ratings.

```
books.head()
```

	ISBN	bookTitle	bookAuthor	yop	publisher	count
0	0195153448	Classical Mythology	Mark P. O. Morford	2002	Oxford University Press	278859
1	0002005018	Clara Callan	Richard Bruce Wright	2001	HarperFlamingo Canada	278860
2	0060973129	Decision in Normandy	Carlo D'Este	1991	HarperPerennial	278861
3	0374157065	Flu: The Story of the Great Influenza Pandemic...	Gina Bari Kolata	1999	Farrar Straus Giroux	278862
4	0393045218	The Mummies of Urumchi	E. J. W. Barber	1999	W. W. Norton & Company	278863

```
users.head()
```

	userID	Location	Age	count
0	1	nyc, new york, usa	34	1
1	2	stockton, california, usa	18	2
2	3	moscow, yukon territory, russia	34	3
3	4	porto, v.n.gaia, portugal	17	4
4	5	farnborough, hants, united kingdom	34	5

```
final.head()
```

	count	userID	bookRating
0	278860	8	5
1	278860	11676	8
2	278860	67544	8
3	278860	116866	9
4	278860	123629	9

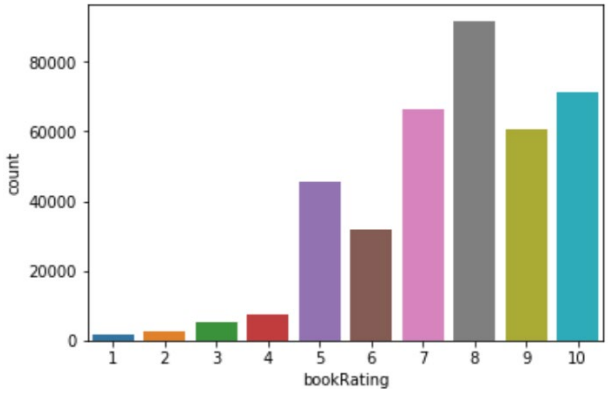
There are 3 tables: users, books and final (ratings table). The above images describe the columns in each table along with the type of data. During the data cleaning phase, we realized several issues and a few findings which will be discussed here.

- Year of Publication column consisted of publisher names
- Year of Publication column also consisted of years that were clear outliers which were fixed by imputing them with the mean value of years
- Age of users had similar issues as the year of publication. Age values ranged from 0 to 244 with missing data as well. Age column was cleaned by using a range of 8 to 90 years.

- The dataset consisted of users in the BookCrossing community who had never rated any books and were hence given a value of 0 in the ratings column. These users were not of interest and hence were removed from the dataset and only the users with explicit ratings (1-10) were included for our analysis.
- Each of the tables (users, books and ratings) were given a column called count for analysis as the ISBN values consisted of characters and special characters and the removal of these rows would be removal of key data. The count in the user table starts from 1 to number of users, the count in the books table starts from number of users+1 to number of books.

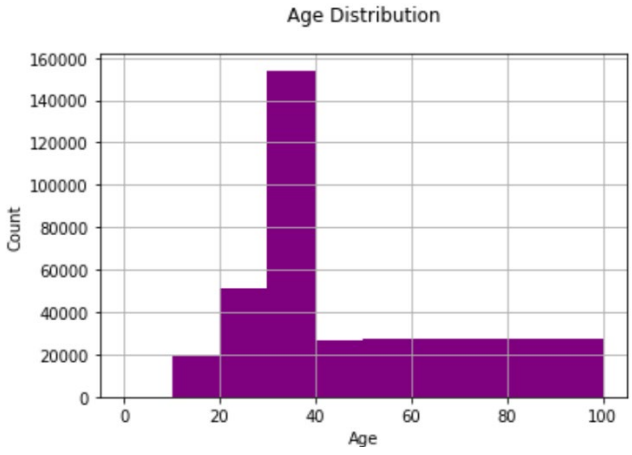
❖ Exploratory Data Analysis

- During EDA we found the following rating distribution:



From the above graph we see that users seem to be generous with their ratings as ratings 7-10 have the highest count.

- We then looked at the age distribution to check what age bracket the users who rated the books belonged to and the following was the result:



Users from age 20-40 have given the highest number of ratings, this was a surprising factor for us as we expected age 10-20 to be a lot higher than what it is due to teenagers who read a lot of books and feel the need to voice their opinions towards a book.

- We moved on from there to find the top 10 most rated books and the average rating of each of these books and the following was the resulting table:

ISBN	bookRating	rating_count
0316666343	8.185290	707
0971880107	4.390706	581
0385504209	8.435318	487
0312195516	8.182768	383
0060928336	7.887500	320
059035342X	8.939297	313
0142001740	8.452769	307
0446672211	8.142373	295
044023722X	7.338078	281
0452282152	7.982014	278

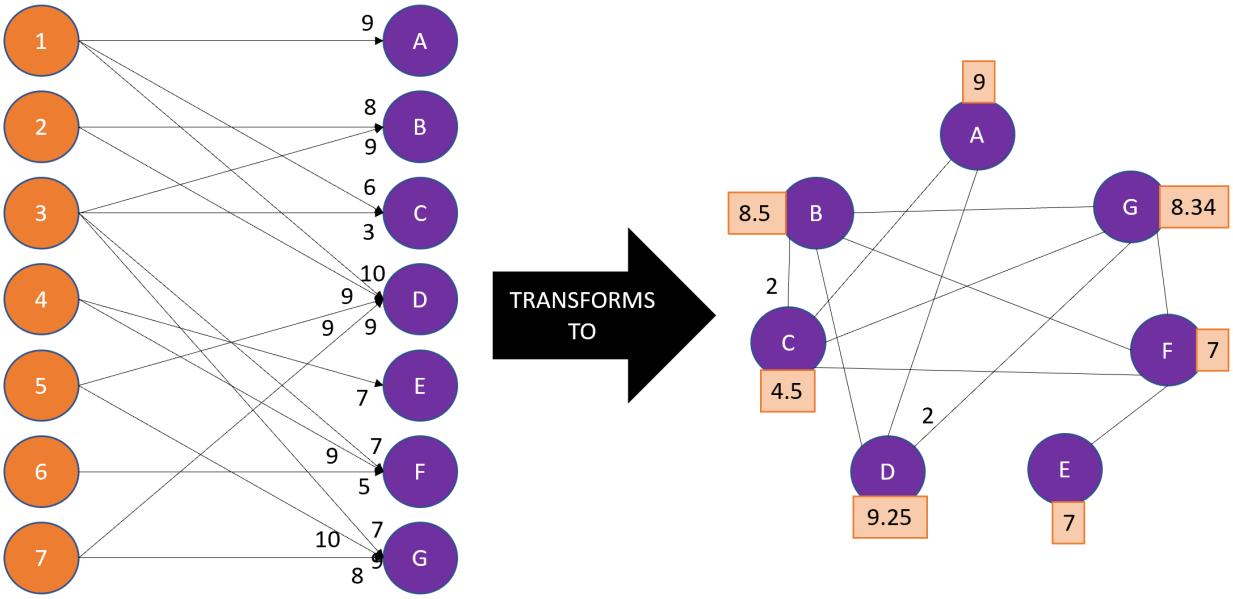
We can observe that book with ISBN= '0971880107' has a total rating count of 581 but we cannot infer that the rating count makes it a good book as we can see that the average rating of that book is 4.39 putting the book below average and hence should not be recommended to any user to read. This observation caused us to make use of the average rating of the books as the key aspect of the graph algorithm.

❖ Algorithm

The dataset is a bipartite graph of one set consisting of users and another consisting of books. The edges represent the rating that a user has given a book. The subgraph generation algorithm uses the following set of steps to form a subgraph of only the books such that we retain the information needed in a smaller graph. This subgraph is then used for the recommendation algorithms that follow.

Subgraph generation pseudocode

1. Create a new graph with the nodes representing only the books from the initial graph
2. If two books have been rated by 1 person, create an edge between them.
3. If two books have N people rating them together, give an edge weight of N between them.
4. Assign the books their average ratings as their node weights.



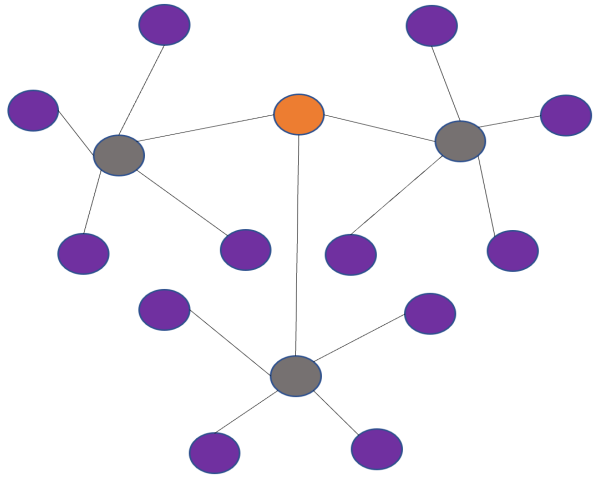
The above described example has 7 users (in orange) and 7 books (in purple). That graph then transforms to the graph shown on the right. In the graph on the right, observe that BC has an edge weight of 2 which means that 2 users have read and rated both books B and C. Their average ratings are shown on the nodes in squares. They represent the node weights.

This algorithm ensures that we capture the information about the ratings of the books as well as create links between books that are read by people together. It also creates stronger links when many people read them together.

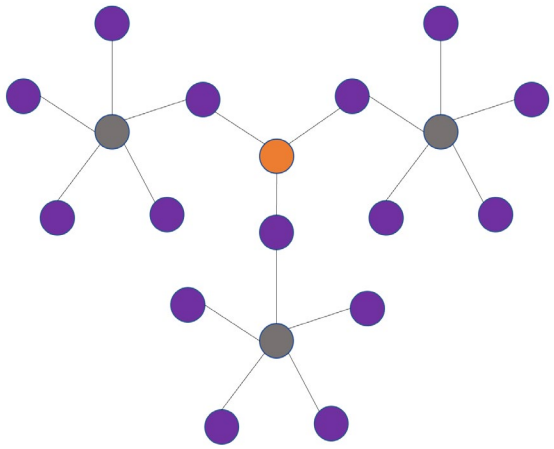
Next, we use this subgraph generated as input to the recommendation algorithms described below.

The first algorithm identifies common neighbors among all the books in the reading list of the user (or the reading list that is given to it) and gives them as the recommendation because these books are very close to the books in the reading list. Refer to the below graph where the grey

nodes represent the user's reading list, and the orange node represents the recommendation made.



The second algorithm identifies shortest paths between each pair of nodes in the reading list and then finds which nodes lie on the common path between each pair. The books corresponding to these common nodes are given as a recommendation. For instance, below is a subgraph generated and the reading list corresponds to the nodes in grey and the orange node is the recommendation made by the algorithm.



If a user is viewing a single book, this book is weighed against all its neighbors and the recommendation list works as a ranking of the parameter *reco_rank=edgeweight*average rating,* and thus captures both the popularity among the books as well as their average ratings.

In summary, we give the suggestion as: “You are currently viewing this book. But the following books are read by many users who have read the current book and their ratings are very good.”

If we make recommendations on the reading list, we use the common neighbors or the common paths algorithms to say that: “You have read these books. Here are the books which are similar based on popularity.”

To generate a subgraph for the entire dataset is computationally expensive (O(n*m*m) where n is number of users and m is number of books). But it is not necessary to compute the entire subgraph to make recommendations based on the algorithms proposed. It is enough to capture the books on the paths between the books and the neighbors of the books in the reading list given. This method is used to obtain the relevant subgraphs and generate recommendations.

Also, if a node in a graph is highly unconnected (very few readers in case of books and very few rated books in case of users), the code generates a random set of recommendations. This solves the problem of cold start.

User based recommendations

The algorithm proposed starts from a reading list as input. However, we have developed a way to obtain reading lists for a user. For a user, the average rating of all the books that have been rated by him/her is found and the books that have ratings greater than the average are sorted and used as the user’s reading list. This list is then used by the algorithm to generate recommendations for the user. Again, if a user is new, he/she will have a choice of creating a starting reading list or given a set of random suggestions.

Results

Sample results for recommendations based on subgraph generation for a single book that is currently being viewed

```
You are currently viewing:
52172    Best of Sherlock Holmes
Name: bookTitle, dtype: object
We recommend that you read:
162014    Warlock
Name: bookTitle, dtype: object
54684    Monstrous Regiment (Pratchett, Terry)
Name: bookTitle, dtype: object
144034    The People Next Door
Name: bookTitle, dtype: object
55575    The Piano Man's Daughter
Name: bookTitle, dtype: object
116249    Courage My Love: A Novel
Name: bookTitle, dtype: object
246514    Longshot Edition Uk
Name: bookTitle, dtype: object
111358    The Stainless Steel Rat Joins The Circus (Stai...
Name: bookTitle, dtype: object
64072    Code to Zero
Name: bookTitle, dtype: object
6424    Great Expectations (Wordsworth Classics)
Name: bookTitle, dtype: object
28263    Deadlock (V.I. Warshawski Novels (Paperback))
Name: bookTitle, dtype: object
```

```
You are currently viewing:
123834    Confederacy of the Dead
Name: bookTitle, dtype: object
We recommend that you read:
123951    Adulthood Rites: Xenogenesis (Butler, Octavia/...
Name: bookTitle, dtype: object
123963    The Dragon and the Fair Maid of Kent
Name: bookTitle, dtype: object
123959    Widow's Weeds (Clan Novel: Tremere Trilogy, Bo...
Name: bookTitle, dtype: object
123883    Dark Tyrants (Vampire - the Dark Ages)
Name: bookTitle, dtype: object
123958    Widow's Walk (Clan Novel: Tremere Trilogy, Boo...
Name: bookTitle, dtype: object
123884    Sea Horse (Saddle Club (Paperback))
Name: bookTitle, dtype: object
123899    Tunes for Bears to Dance to
Name: bookTitle, dtype: object
123891    X,Y
Name: bookTitle, dtype: object
123962    The Nihilesthete
Name: bookTitle, dtype: object
123961    Marooned (Star Trek Voyager, No 14)
Name: bookTitle, dtype: object
```

```
You are currently viewing:
52210    Little Women
Name: bookTitle, dtype: object
We recommend that you read:
52439    Murder on the Ballarat Train
Name: bookTitle, dtype: object
52424    Sarum: The Novel of England
Name: bookTitle, dtype: object
52411    Dear Mom Thank You For Everything
Name: bookTitle, dtype: object
52410    The Moon in the Water
Name: bookTitle, dtype: object
52408    The Chains of Fate
Name: bookTitle, dtype: object
52397    The Makeover Murders
Name: bookTitle, dtype: object
52396    Women as mothers
Name: bookTitle, dtype: object
80851    A Hundred and One Uses of a Dead Cat
Name: bookTitle, dtype: object
52421    Winter Solstice
Name: bookTitle, dtype: object
52387    Fresh Milk: The Secret Life of Breasts
Name: bookTitle, dtype: object
```

Sample results for recommendations given based on a reading list input

```
Your reading list is:
213701    Harry Potter and the Order of the Phoenix (Boo...
Name: bookTitle, dtype: object
3459    Harry Potter and the Chamber of Secrets (Book 2)
Name: bookTitle, dtype: object
3839    Harry Potter and the Prisoner of Azkaban (Book 3)
Name: bookTitle, dtype: object
From observations we made on your reading list, we recommend you read:
5431    Harry Potter and the Goblet of Fire (Book 4)
Name: bookTitle, dtype: object
5432    Harry Potter and the Chamber of Secrets (Book 2)
Name: bookTitle, dtype: object
2809    Harry Potter and the Sorcerer's Stone (Book 1)
Name: bookTitle, dtype: object
77242    Velvet Song
Name: bookTitle, dtype: object
93596    George &amp;amp; the Virgin
Name: bookTitle, dtype: object
2143    Harry Potter and the Sorcerer's Stone (Harry P...
Name: bookTitle, dtype: object
41893    J. K. Rowling: The Wizard Behind Harry Potter
Name: bookTitle, dtype: object
81736    The Fowlers of Sweet Valley (Sweet Valley Saga)
Name: bookTitle, dtype: object
3476    Jane Eyre (Bantam Classics)
Name: bookTitle, dtype: object
```

```
Your reading list is:
117529    The Lost World (Ladybird Children's Classics)
Name: bookTitle, dtype: object
122743    Payback
Name: bookTitle, dtype: object
115891    The Hound of the Baskervilles: Another Adventu...
Name: bookTitle, dtype: object
From observations we made on your reading list, we recommend you read:
26461    Adventures of Huckleberry Finn
Name: bookTitle, dtype: object
117372    Dinosaurs
Name: bookTitle, dtype: object
```

Sample results for a user based on his/her obtained reading list

User number 11676:

```
Your reading list is:
248971    Garzanti - Gli Elefanti: Dolori Del Giovane We...
Name: bookTitle, dtype: object
248969    El Lobo de Mar
Name: bookTitle, dtype: object
248949    Felix und das liebe Geld
Name: bookTitle, dtype: object
248944    Un peu plus loin sur la droite
Name: bookTitle, dtype: object
248926    Tom Strong - Book Two of the Heroic New Series
Name: bookTitle, dtype: object
248913    Tom Green - Udder Insanity
Name: bookTitle, dtype: object
248908    Hold Your Horses
Name: bookTitle, dtype: object
248907    My Brother's a World-Class Pain: A Sibling's G...
Name: bookTitle, dtype: object
248887    Pizza
Name: bookTitle, dtype: object
248880    Autumn Angel
Name: bookTitle, dtype: object
From observations we made on your reading list, we recommend you read:
32440    Tending Roses
Name: bookTitle, dtype: object
81596    The life of Sir Arthur Conan Doyle
Name: bookTitle, dtype: object
32454    Blue Diary
Name: bookTitle, dtype: object
16079    Lullaby: A Novel
Name: bookTitle, dtype: object
32466    Breathing Water
Name: bookTitle, dtype: object
16083    The Hot Zone
Name: bookTitle, dtype: object
212699    Sins
Name: bookTitle, dtype: object
48867    The Watcher
Name: bookTitle, dtype: object
16105    The Pilgrimage
Name: bookTitle, dtype: object
179948    Silence on Monte Sole
Name: bookTitle, dtype: object
```

User #98391

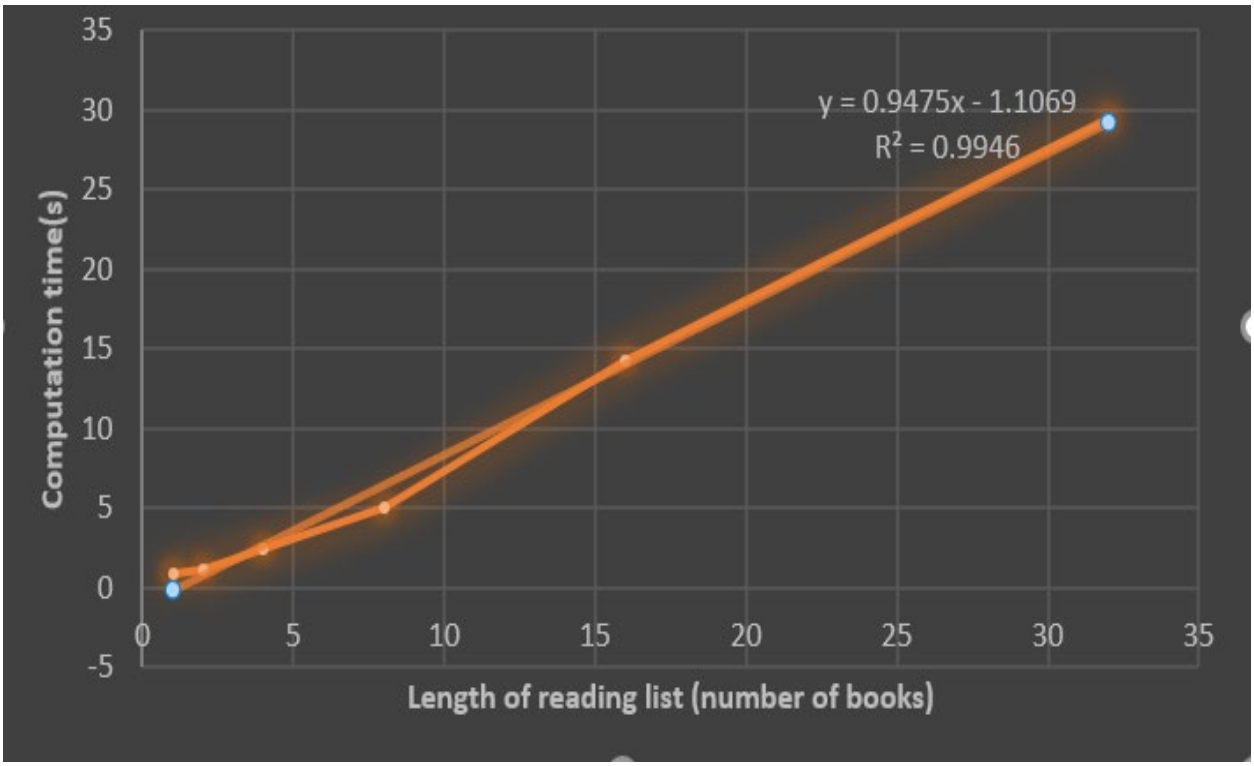
```
Your reading list is:
250688    Legacy Of The Black Dragon
Name: bookTitle, dtype: object
250687    World-walker
Name: bookTitle, dtype: object
250685    The Deadly Garden Tour (Five Star First Editio...
Name: bookTitle, dtype: object
250684    Heist and Seek
Name: bookTitle, dtype: object
250683    Eye Of Newt (Five Star First Edition Mystery S...
Name: bookTitle, dtype: object
250682    Speak Now
Name: bookTitle, dtype: object
250681    A Test of Faith
Name: bookTitle, dtype: object
250678    The Mongol Reply (Five Star First Edition Myst...
Name: bookTitle, dtype: object
250676    Persuasive Evidence
Name: bookTitle, dtype: object
250674    Pure Dynamite
Name: bookTitle, dtype: object
From observations we made on your reading list, we recommend you read:
14222    The Legacy
Name: bookTitle, dtype: object
32454    Blue Diary
Name: bookTitle, dtype: object
32477    Cold Truth
Name: bookTitle, dtype: object
65246    Tell Me Why
Name: bookTitle, dtype: object
32479    In the Blood
Name: bookTitle, dtype: object
32501    Threshold: A Novel of Deep Time
Name: bookTitle, dtype: object
48902    The Cross-Legged Knight
Name: bookTitle, dtype: object
48926    The Destiny
Name: bookTitle, dtype: object
32548    Kentucky Sunrise
Name: bookTitle, dtype: object
16168    Shadows and Light
Name: bookTitle, dtype: object
```

From the above results, it can be observed that even though the algorithm does not work towards finding similar books based on the context or genre or any other factor but popularity among users and ratings, the recommendations are relevant based on mere observation. For example, when Harry Potter books were given in the reading list, the recommendations suggested were other Harry Potter books.

Computation time analysis

To analyze computation time, we first observed the changing parameters in the algorithm. The only parameter observed to change was the length of the reading list. Note that we have not included the time taken to read the graph from the dataset for analysis. This time was a constant and would not matter once the dataset has been read.

The following figure shows the computation time when varied against the length of the reading list given.



Now, these times are subject to change based on the system specifications but we can clearly conclude that a linear relationship exists between the computation time and the length of the reading list.

This is also logical considering that the time varies as O(n*m*q) where n is the number of users, m is the number of books and q is the length of the reading list. If we fix n and m based on the dataset, the only changing parameter here is q and the variation of computational effort (and hence time) with q is linear.

Analysis of performance of the recommendation system/ quality of the recommendations

To analyze the quality of the results, we first selected a random sample set of users. We obtained a part of their reading lists (training lists) based on the method described in the previous section. Then, we ran the algorithm on every sample user and compared the recommendations made against all the books rated by that user. In essence, we made recommendations on the training reading list and saw if the recommendations given by the algorithm were also read by the user (but not in the training list). If the recommended set of books contained the books that were read by the user, we considered that a true positive. If the recommended set of books did not contain any book read by the user, we considered that a false positive.

For the set of experiments conducted, we fixed the length of the reading list for every user sampled to be equal to 10 and smaller only if the user rated less books. To eliminate the problem of cold start, we randomly sampled users who have rated 5 or more books only (for the first set of tests) and 9 or more books (for the second set of tests). The table below details the results found.

Time	Sample_users	Min_rated	TP	FP	Precision		Time	Sample_users	Min_rated	TP	FP	Precision
2.12	20	9	622	3	99.52%		1.56	20	5	279	96	74.40%
4.28	40	9	707	148	82.69%		3.13	40	5	840	492	63.06%
6.11	60	9	1493	28	73.87%		4.23	60	5	806	1295	38.36%
8.23	80	9	1756	26	67.53%		6.39	80	5	3101	15635	16.55%

The results show that the average precision (true_posititive/(true_positive+false_positive)) is 80.90% when sample user had read at least 9 books and 48.09% when the sample user had read a at least 5 books.

We can observe that when we randomly sampled users who have given fewer ratings, we found that the number of false positives was high, and the more such users we sampled, the higher the false positives. This is logical, because they have read very few books which are almost always in the training reading list. If recommendations are made, there are hardly any more books that the users have read that are already rated by them. This also adds credibility to the algorithm because it is now suggesting new books previously not read by the user.

When we increased the average number of books read by the users in the sample set, our results drastically changed. Since the user has now read many books, the training list will now contain a small portion of the books that the user has read, and hence the recommendations will include a large set of books that the user has already read, but are not in the training list.

This adds credibility to the algorithm.

How to run our code:

- Python script 'NS_Data_Pre-processing.ipynb' can be run to preprocess the data which will produce data files 'new_books.csv' and 'final.csv' which will be used in the algorithm
- Files 'final.csv' and 'new_books.csv' can already be found in the folder provided
- 'NS_algo_clean.ipynb' file is the python script that consists of our algorithm
- When the above python script is executed, you will have the following options:
  - Input one book, example: 279444
  - Input a reading list, example: 279444 290442 331996 278860 278864
  - Input userID, example: 11676 -> This will produce User based recommendations
- Select one of the above options by following the instructions provided

- The python script ‘NS_algo_eval.ipynb’ is the file needed to run our algorithm evaluation, here you have to enter the number of sample users, example: 20