Can TF-IDF analysis of book titles be used to construct an accurate item-based book recommender-system?

Group W06G7

Miles Sandles
1461917
msandles@student.unimelb.edu.au

Alexander Broadley
1513634
broadleya@student.unimelb.edu.au

Quoc Khang Do
1375531
quockhangd@student.unimelb.edu.au

1. Executive Summary

In this report we aimed to construct an item based recommender system, which could predict user ratings for new books, based on the ratings they had given similar books. Specifically, we wanted to see if implementing TF-IDF on book titles, then using cosine similarity as a similarity metric would allow us to accurately predict user ratings of new books. If the system were accurate, it would provide commercial benefit to book sellers, as they would be able to preferentially stock new books which were predicted to be more popular, and tailor book recommendations to individual users, potentially generating more custom from successful recommendations. Although we found that the recommendation system did not accurately predict the ratings users gave new books, we provided suggestions for future improvements to this system – chiefly using book summaries rather than book titles when producing TF-IDF vectors, and driving customers to provide more ratings on books they have read. Implementing such changes would require more computing power and commercial resources than were available to us, but this would make the recommendation system more accurate, and therefore more commercially valuable.

2. Introduction

In this report we aimed to create an item-based recommender system which predicts users' ratings of new books based on ratings they have given previous books. The methodology behind this recommender system is that for each book, we find the TF-IDF score for its title, and use this as a similarity metric to find its most similar books. We chose to use this metric as it would allow us to compute a similarity score between already rated books and those which have not. This would, in some way, mitigate 'cold start' issues that arise when recommending a novel book to a database of existing users (Lika, Kolomvatsos and Hadjiefthymiades, 2014). We also chose an item-based similarity metric as item similarity is more stable than user similarity, which could change over time.

After these predictions are made, we were then able to recommend which new books bookstores should stock by recommending books that we predict to be highly rated amongst many users. If our predicted scores are reliable, this framework would provide a basis for which bookstores, who track their users' reading habits (or at least have access to their ratings of various books), could better allocate their spending by purchasing new books that they can be confident their reader base will enjoy. They would also be able to identify books likely to be popular amongst only a few readers, and recommend/stock these books specifically for these customers, saving money on over-stocking 'niche' books.

There were 6 datasets used in this report. 'BX-Books', 'BX-Ratings, 'BX-Users', 'BX-NewBooks', 'BX-NewBooksUsers' and 'BX-NewBooksRatings'. In BX-Books and BX-NewBooks the title, author, publisher, year of publication and ISBN for a variety of books was stored. In BX-Users and BX-NewBooksUsers the age, country, state and user ID of each user of the system was stored. Finally, in BX-Ratings and BX-NewBooksRatings the ISBN number, user ID and score associated with each rating was stored. The goal was to use the information in these datasets (except the ratings in 'BX-NewBooksRatings') to predict the rating that users in 'BX-NewBooksUsers' would give to the books in 'BX-NewBooks'. These could then be compared to the actual ratings they awarded these books in 'BX-NewBooksRatings'.

3. Methodology

3.1. Data Preprocessing

Users datasets ('BX-Users.csv', 'BX-NewBooksUsers.csv)

User-city, User-state and User-country:

As the data in these columns was categorical, the goal of cleaning the columns was to correct formatting errors/differences preventing unnecessary separation of categories (i.e. 'USA' and 'United States' should be one category).

First, all trailing whitespaces were removed, all letters made lower case, and any non white-space or alphabetical characters removed.

Missing values were identified in the 'User-City', 'User-State' and 'User-Country' columns, but as these values were not going to be the focus of our analysis, we did not remove the data concerning these users to ensure as many users were present in the dataset as possible.

The 'country_clean' library from 'dataprep.clean' was then used to convert different country name aliases to one standard format. For example 'u s', 'u s a' and 'america" all became 'united states'

User-age:

As we wanted the values in this column to be purely numerical, we first checked for any instances where the age was stored as a word rather than a number (i.e. 'thirteen' rather than '13'). As there were no instances of this, we then removed any non digit characters. Range errors were then corrected by only allowing users to be in the age range of 13-100.

As ~40% of records in the dataset were missing a value for user age, but this value would not be critical for later analysis, we imputed the missing values by sampling for the distribution of known ages.

Books datasets ('BX-Books', 'BX-NewBooks'):

ISBN number:

We first checked the records for missing or duplicated ISBN values. No records were missing any values, but one ISBN number (039592720x) in 'BX-Books' was duplicated. One of these records lacked a title, but was otherwise identical, and so was removed from the dataset.

We then checked that all ISBN numbers were 10 or 13 digits long, with the last digit (the check digit) potentially being an 'X' (representing the number 10) as according to the standard ISBN format (What is an ISBN? | International ISBN Agency, no date). This revealed that one record in 'BX-NewBooks' had used an ASIN number (B00009EF82) rather than ISBN. This was corrected to its 10-digit ISBN number (038529929X).

Year-of-publication:

We then checked the year of publication column for missing values, non-integer values, 0 values, impossible/future values (i.e. year of publication > 2024) and exceptionally low values (year of publication < 1800). No missing or non-integer values were found, but in 'BX-Books' there were 314 zero values and 3 future values and in 'BX-NewBooks' there were 185 zero values and 2 future values.

The ISBN database API was used to replace these values with their correct publication years where possible. When a request was not able to retrieve a corrected publication date, an error code was introduced into the dataset to allow easy detection of these cases.

After using the ISBN db API, only 6 books did not have correct publication dates in the 'BX-Books' dataset, and so these were corrected manually. In 'BX-NewBooks' correct publication dates could not be retrieved for 164 of the error values, but as this field is not crucial for our later analysis these records were left in the dataset in order to maximise the number of books that we had information for.

Book-Author:

We wanted the data in this column to be categorical, to potentially facilitate integration of Book-Author into our recommender system. As such, the goal of cleaning these values was to get as many of the alternative formats for each name to match as possible post cleaning.

The book author column was first checked for missing or non-string values and none were found. All author names were first made lower case (to prevent any mismatches due to alternative capitalisation). Any text between brackets, which usually contained uninformative information for the purpose of standardising author names [i.e. Wayland Drew (adapter)] was then removed. All punctuation was removed, as well as common titles (dr, mrs, mr, ms etc.) and any stop words. Leading and ending white spaces were also removed, alongside any double spaces within the author names.

There were many cases where only one word of the author's name was recorded, often the surname. This alone was often insufficient to identify the author with certainty, especially when the surname was common and so shared by many authors (i.e. 'Smith'). The ISBN database API was therefore again used to search for the complete author's name in these cases. If this successfully retrieved more of the authors name (i.e. more than one word of it) these values were cleaned by the same processes as above before being put into the data frame.

Even after these steps author names could often appear multiple times in different formats, for example J. R. R. Tolkien could appear as:

john ronald reuel tolkien , j r r tolkien , jrr tolkien , j tolkien , tolkien jrr , tolkien j r r , tolkien , tolkien john ronald reuel , tolkien

Therefore, to standardise author names into one format, each was split into its individual words, and these words sorted in descending alphabetical order (A-Z) by their first letter. The first letter was used to prevent abbreviations of name(s) affecting the order of the sort (i.e. 'j r r tolkien' and' tolkien john ronald reuel' would both sort into first name, middle names, last name). If a name consisted of more than one word, the first letter of the first word was then taken alongside all of the last to represent the author's name. In this example, all these various formats (except the one-word 'tolkien') would standardise to 'j tolkien'.

Finally, to handle any remaining one-word names, each was used to search the data frame for another author name containing the word and replaced with the standardised format name version (i.e. 'tolkien' would be replaced with 'j tolkien').

Book-publisher:

We once again wanted the values in these columns to be categorical, to potentially integrate the book publisher into our recommender system. The goal of cleaning these values was therefore to get as many publisher names to match post processing, despite formatting differences.

Again, we checked for missing and non-string values, finding none in 'BX-NewBooks' but 37 in 'BX-Books'. The ISBN db API was again used to fetch the book publishers for the records with NA values, and successfully retrieved a publisher for each of these records. We then made all names lower

case, removed text in brackets, removed stop words, removed punctuation, and removed double spaces for the same reasons as when cleaning author names.

If the last letter of any word was an ‘s’ it was removed, as some publisher names were occasionally pluralised. We then removed various words and abbreviations associated with publishing companies in general such as book, corporation, llc, publisher, as these were again only occasionally present the company name. Finally, as there were still many cases of less common miscellaneous words separating publisher names (i.e. ‘warner’ and ‘warner vision’), we calculated how many publisher names each word appeared in and represented each publisher by its most commonly appearing word. For example, ‘penguin usa’ became ‘penguin’.

Book-title:

Unlike book author and publisher, we wanted to use the book titles to calculate a metric of book similarity, rather than have it function as a category. We used TF-IDF to represent each book-title in vector space, and the distance between them in this space as a measure of similarity.

We first checked for records missing book titles, finding 31 such records in ‘BX-Books’ and 9 in ‘BX-NewBooks’. Given how crucial these values were to our analysis, we used the ISBN db API to fetch the book titles for any records missing one. This successfully replaced all missing values with a book title. We then made all book titles lowercase and removed any text between brackets as this often contained information unrelated to the content of the book (i.e. ‘(book 1 of 5)’). We removed any punctuation and converted any digits into word format using the ‘num2words’ library (i.e. ‘1000’ became one thousand. Stop words were then removed from the titles, as they do not convey significant information regarding book theme (Rajaraman and Ullman, 2011). The remaining words in the book titles were then stemmed and lemmatised to allow different inflections of the words to be recognised as one, a standard practice in language processing(What Are Stemming and Lemmatization? | IBM, 2024). This process is demonstrated below on a demo book title below, changes that will occur at each step are highlighted in red.

‘The Lord of the Rings : the fellowship of the ring (book 1)’

Made lower case, text in brackets removed, punctuation removed, double spaces removed

‘the lord of the rings the fellowship of the ring’

Stop words removed

‘lord rings fellowship ring’

Lemmatize and stem words

‘lord ring fellowship ring’

Ratings datasets (‘BX-Ratings.csv’, ‘BX-NewBooksRatings.csv):

First we checked the datasets for any empty cells/missing values, none were found in either.

ISBN number:

We then checked that all ISBN numbers were correctly formatted, as in the books datasets (above). In the BX-ratings datasets, the ISBN number ‘039592720x’ was incorrectly capitalised in 4 records, so this was corrected manually. In the NewBooksRatings dataset the ASIN number ‘B00009EF82’ appeared 3 times, so this was manually corrected to the book's ISBN number '038529929X'.

User-ID:
Although user ID is stored as a string, we wanted to make sure each ID was a valid number, and so we checked that each could be successfully converted to an integer. All IDs in both dataset were valid numbers, and so no further cleaning was necessary.

Book-Rating:
We checked that all book ratings were integers, and in the range 0-10 (the range of valid rating). All ratings passed these checks, so no further cleaning was necessary.

3.2. Constructing the recommender system

After cleaning all the data, the ISBN number column was used to merge the ratings for the books, users and rating datasets, producing one data frame for the newBooks datasets and one for the Books datasets. Due to lack of computational power, a subsample of each of these data frames was taken (1000 new books, 5000 'original' books), and the two datasets merged into one larger dataframe.

The *scikit-learn* library was used, on each book, to calculate the TF-IDF vector for its cleaned, stemmed and lemmatised book title, essentially assigning a vector to each book. Then, for each pair of books (vectors), we calculated their similarity by finding the cosine similarity between them. This information was stored in a matrix assigning a similarity score to each pair of books – a higher similarity score means the pair of books are more similar.

Now, to predict a user's ratings of a new book in 'NewBooksRatings', we have to find the top three most similar books that they have rated before (using our similarity matrix) and find the weighted average rating of those books, where the weights correspond to the similarity scores.

There are 3 cases:

1. If there are no similar books that the users have rated before (i.e similarity score = 0), their predicted rating is set to NaN signifying that there was insufficient data for the recommender system to make a valid prediction

2. If there was a book that they have read in the past which had the same exact title (i.e similarity score = 0), the predicted rating for the new book would be the same as the rating they gave for the old book. Checking this was necessary as the same book can be published multiple times, and therefore have multiple ISBN numbers.

3. Otherwise, find the weighted average rating of the top three most similar books they have rated before. If there are less than three, find the weighted average of those.

4. Data Exploration and Analysis/Results

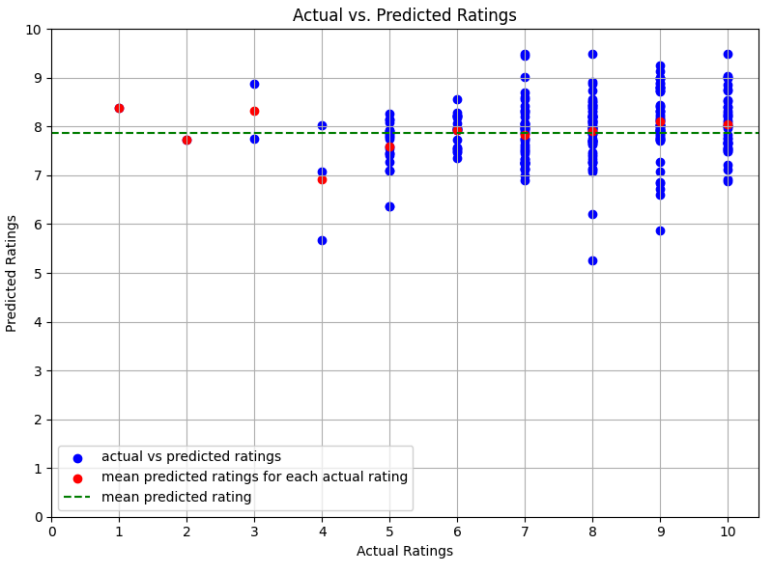

Figure 1: Scatter plot showing actual rating given to books in the 'BX-NewBooks' dataset against predicted raings. Red dots show the mean predicted rating for each 'actual' rating, and the green line shows the overall mean predicted rating.

Figure 1 shows the results of our predictive system on the 6000 book subset of the data (5000 from BX-Books- and 1000 from BX-NewBooks). As can be seen in the figure the mean predicted rating for each actual rating is consistently between 6.9 and 8.5, and our predictions therefore show no correlation with the actual rating users gave the book. It is also clear that there is relatively little variation in the predicted ratings, with most predicted ratings falling between 7 and 9 (overall mean predicted rating of 7.88). The mean absolute error (1.29) for this recommender system was, however, relatively low.

Figure 2 and 3 show the distribution of the number of books users have rated in both the entire 'BX-Books' dataset (figure 2), and the subset used for analysis (figure 3). In the subset of 'BX-Books' you can see that the vast majority of the users have only rated one book, with very few users having rated more than 2 books. In the entire dataset, a similar pattern is observed, however more users exist in each category. Furthermore, there is a significant number of users who have rated an extreme number of books, as seen by the long 'tail' in the plot.

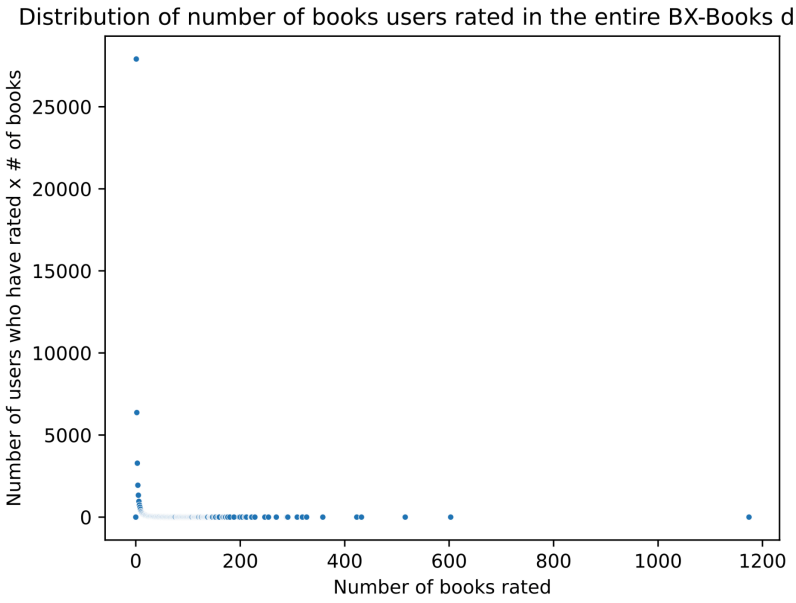

Figure 2: Distribution of the number of books users rated in the entire 'BX-Books' dataset

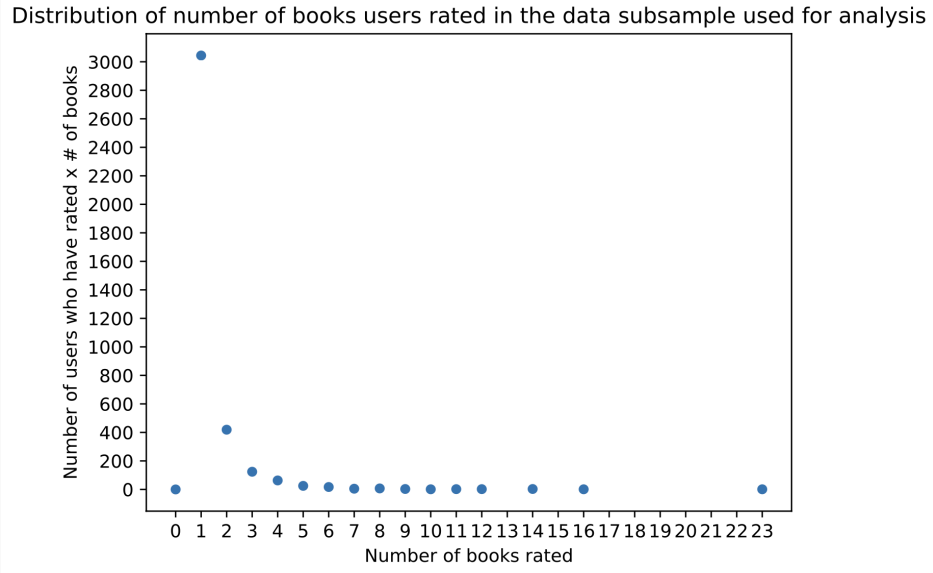

Figure 3: Distribution of the number of books users rated in the subsample of the merged 'BX-Books' datasets

Figure 4 and 5 show the average rating of each book, against the number of ratings, for books in both the 'BX-Books' (figure 4) and 'BX-NewBooks'(figure 5). The orange line in these figures shows the mean book rating. As can be seen in these figures, the mean book rating in 'BX-Books' was 7.76, which is very similar to the mean rating predicted by this recommendation system (figure 1). This is also similar to the mean of the 'actual' ratings that we were attempting to predict, which was 7.57 (figure 5).

Figure 4: Scatter plot showing the mean rating of a book (x axis) against the number of ratings it received (y axis), the vertical yellow line shows the mean book rating of 7.76.

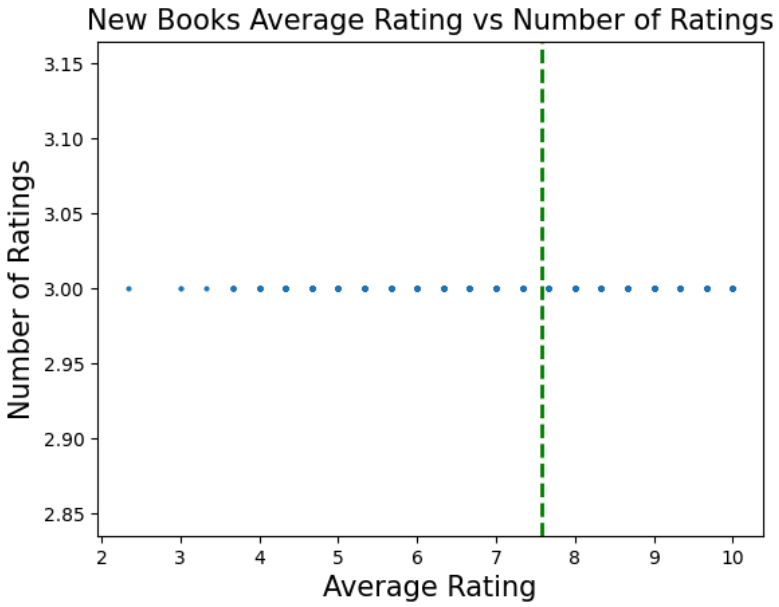

Figure 5: Scatter plot showing the mean of the actual rating of every book in 'BX-NewBooks' (x axis) against the number of ratings it received (y axis), the vertical yellow line shows the mean book rating of 7.57.

5. Discussion and Interpretation

Our recommender system appeared relatively accurate when evaluated using the mean absolute error metric (mean difference between predicted and actual rating), with an MAE of only 1.29. However, further analysis of our predicted rating revealed that this was not the case. In figure 1 we can clearly see that there is actually no correlation between the predicted values and the actual ratings users gave these books, and instead for every 'actual' the mean predicted rating was relatively constant, with the majority of predicted ratings being greater than 7, and overall mean predicted rating being 7.88.

When taking into account the data in figures 4 and 5, which shows that the mean actual book rating was approximately 7.7 in both newBooks and Books, one explanation for our relatively low MAE is that we tend to predict the average book rating for all books, regardless of what users actually rated them. As such, our predictions remain largely unreliable and therefore uninformative, despite a promising MAE metric. As to why our predictions were so unreliable, there are a variety of possible explanations.

Firstly, the fact that most users had rated a low number of books massively impairs our ability to recommend new books based on item similarity. As figure 2 shows, in the subset of data we had to use to run our recommender system due to computational constraints, most users had only rated one book in the 'BX-Books' subsample. If the book(s) they had read and rated previously were not similar to the one we were trying to predict the rating of, it was effectively impossible to assign a predicted rating. Furthermore, if only 1 similar book was read, then noise is likely to massively affect the rating we predict, again impairing the reliability of our prediction.

Another issue is whether TF-IDF vectors of the book titles accurately reflect books of similar themes/content. Although in some cases it is easy to see how titles close in TF-IDF vector space would be similar in theme, for example the sci-fi books 'Dune' and ' Dune Messiah', this is not the case for all similar books, such as the fantasy novels 'The Hobbit' and 'The Lord of the Rings'. One obvious improvement would therefore be to use book summaries, rather than book titles, which are also available through the ISBN db API, as these are arguably more likely to reflect the theme and genre of the book than the title alone. This idea is discussed further in 'Limitations and Improvement Opportunities'.

Although our recommendation system was not found to be particularly accurate, we can still demonstrate the commercial benefits of such a system if it was accurate. Supposing our predictions were accurate, we could recommend that book stores stock more highly of those books we predicted to be highly rated. Stocking such books will generate more money for stores, as not only will they be able to generate more money through profits by selling popular books, they will waste less money stocking books that customers would not like, and would therefore not sell well. For example, 318 of the books we predicted the rating of had an average rating greater than 8. If these ratings were reliable, these books would be good candidates to prioritise stocking as they would likely sell well.

It would also allow bookstores to tailor recommendations to individual customers based on their previous reading history. This would allow bookstores to market potentially niche books to just the small number of users that would like them. This would maximise revenue, as books not popular with the general audience can be stocked to meet the demand just of the subset of customers likely to enjoy them. Furthermore, if a bookstore is able to consistently recommend new books to customers that they enjoyed, they are likely to retain the custom of those customers, rather than have them purchase books elsewhere, further increasing profits.

Lastly, despite not being drawn from term frequency comparisons, results for data drawn on ratings and number of ratings (see Figures 4 and 5) for books would be useful when considering books to buy that would sell simply from their popularity: books with more ratings would likely be sold at a high volume but can't be guaranteed to be enjoyed by everyone who buys them, on the other hand books rater *higher* would likely result in better reactions from customers but may not sell as many copies. We also found that 351 books had predicted ratings 8 and above which could be provided to owners to recommend books to buy that are predicted to be most enjoyed.

6. Limitations and Improvement Opportunities

One of the primary limitations of our project is computing power. As discussed above, we had to take a subset of both the 'BX-Books' and 'BX-NewBooks' datasets when implementing our recommender system. As such, the number of books each user we were trying to predict ratings had rated was very low (figure 2). This undoubtedly impaired the accuracy of our results, as even when customers had read a similar book, one rating of a book of a similar theme is unlikely to accurately reflect how a user feels about books of this theme in general. For example, liking 'The Hobbit' does not necessarily mean you would enjoy all fantasy books, but if you liked 'The Hobbit', 'The Lord of the rings' (1, 2 and 3) and 'Game of Thrones' you are much more likely to enjoy fantasy books as a general genre.

Access to greater computing power would minimise this limitation to some extent, as we would not have to subset the data at all. As such, we would have access to all previous ratings of every user when trying to predict their ratings for new books. However, the extent to which this would improve the scores is likely minimal, as even in the full dataset the vast majority of users rated on1y or 2 books (figure 3). Therefore, if this system were to be implemented commercially, users could be much more actively encouraged to rate books they have read, to increase the relatively poor amount of rating data available for each user even in the full dataset (figure 3). The more ratings that could be recorded for each user, the more accurate the prediction system would get for them, as the chances they had read a book similar to one we wanted to predict their rating for would increase.

Furthermore, access to greater computer power would allow for more content with which to use term frequency analysis on. We lacked the storage capacity and data reading speed with which we would have been able to use book summaries, rather than being limited to book titles. An even greater extension of this could be to incorporate text from within the books themselves. With more time we could also incorporate other book metrics into our similarity score, for example whether or not two books were written by the same author.

7. Conclusion

The analysis of term frequency and inverse document frequency (tf-idf) in book titles to compare and construct a recommendation system for users and book store owners, is based on the idea that authors invest lots of energy into creating titles that simplify the overall theme or material content of their books, both for fiction and nonfiction. If this was always the case, then the distance between book titles in tf-idf vector would allow us to compute accurate similarity metrics between books based purely on their titles, rather than users rating of them. One benefit of such a system, if it were accurate, is that we could predict the ratings users would give books no users in the system had rated, which is a problem facing many recommendation systems.

Unfortunately our implementation of such a system lacked accuracy for a variety of potential reasons. One major reason for this inaccuracy was that most users only tended to rate one or two books (see Figure 2), thus making it hard to predict their rating for new books. However, while our results would be considered consistently inaccurate in finding the actual values users would rate a book after reading, its performance would seem impressive based on the mean average error metric alone, highlighting the danger of using these metrics in standalone when assessing these systems (see Figure 1).

To improve the accuracy of such a system for implementation in a real life setting, where an accurate recommender could have significant commercial benefits, we recommend using book summaries, rather than book titles, when performing tf-idf analysis and encouraging users to rate more of the books they have read, as both would likely improve the accuracy of the system.

Overall, the tf-idf based system we have implemented using book titles would be ineffective for recommending books to users and bookstore owners. It provides predictions that are too inaccurate to ensure customer satisfaction and prolonged sales for a business. Without considering other factors, the information provided by tf-idf analysis between book titles cannot consistently make correct predictions of how a user would enjoy a book.

8. References:

Lika, B., Kolomvatsos, K. and Hadjiefthymiades, S. (2014). 'Facing the cold start problem in recommender systems'. *Expert Systems with Applications*, 41 (4, Part 2), pp. 2065–2073. doi: 10.1016/j.eswa.2013.09.005.

Rajaraman, A. and Ullman, J. D. (eds). (2011). 'Data Mining'. in *Mining of Massive Datasets*. Cambridge: Cambridge University Press, pp. 1–17. doi: 10.1017/CBO9781139058452.002.

'The Impact of Online Recommendations and Consumer Feedback on Sales' by Pei-Yu Chen, Shin-yi Wu et al. (no date). Available at: https://aisel.aisnet.org/icis2004/58/ (Accessed: 10 May 2024).

What Are Stemming and Lemmatization? | IBM. (2024). Available at: https://www.ibm.com/topics/stemming-lemmatization (Accessed: 9 May 2024).

What is an ISBN? | International ISBN Agency. (no date). Available at: https://www.isbn-international.org/content/what-isbn/10 (Accessed: 7 May 2024).