COMP20008 Assignment 2 Report
Group W10G9

Do Yeong Kim	**Kunal Dewani**	**ShuYuan Zhang**
1081064	1378903	1339044
doyeongk@student.unimelb.edu.au	kdewani@student.unimelb.edu.au	shuyzhang2@student.unimelb.edu.au

Executive Summary

This report provides an analysis and evaluation of data processing performed on a collection of datasets from an online bookstore containing information about books, users, and the ratings given to books by users. The data processing was conducted with the purpose of improving the end user experience by delivering value through insights gained from the data.

Methods used during data preprocessing include techniques such as cleaning, stem extraction, and lemmatization. Machine learning techniques were then used to gain insights from this data, namely k-means clustering to group similar books, item-item based collaborative filtering to build a recommendation system, and fuzzy string matching using Levenshtein distance to improve our searching feature.

These techniques, along with general data analysis techniques, were used to create an interactive program that can return useful information about books and authors such as highest rated and most popular books, and recommendations of books that users may enjoy based on other books they liked.

Recommendations to improve the program are to incorporate more detailed information about books such as genre to make better predictions, and to separate the program by its intended audience of managers and users.

Introduction

Online bookstores continuously seek ways to improve their inventory turnover along with improving their customer experience. Bookstores can improve their inventory turnover by extracting insights from the bookstores dataset that help managers in decision as to what kind of books should be stored in inventory, while recommending buyers as to which books to buy to improve customer experience.

This report focuses on the datasets provided and how the concepts of data wrangling, data preprocessing and machine learning can be utilised to achieve the aforementioned objectives. The research question being explored through the team's research and discussion in this technical report is as follows:

How can we use data to deliver value to the bookstore by improving the online user experience?

The datasets utilised are the extracted from the *csv* files titled *BX-Books.csv*, *BX-Ratings.csv* and *BX-Users.csv*. The data wrangling and preprocessing was done using python and in particular, the pandas library, along with the Scipy library for implementing the machine learning techniques of k-means clustering, item-item based collaborative filtering and fuzzy string matching.

Item-item based collaborative filtering was performed to produce a similarity score/ rating between books, thereby allowing the program to recommend a user books similar to the ones they liked. K-means clustering was then performed as an unsupervised machine learning technique to classify the various book titles into different clusters/categories, again allowing for improved recommendation. Finally, fuzzy string matching was implemented in the creation of a robust search function. If the particular book title or author is not found, then the most similar books or authors are provided based on a similarity score. This report

also highlights the power of Python and its data manipulation and data analysis operations, and their utility in real-world applications.

Methodology

The initial approach to interpreting the data was to understand the layout and the type of information included in each file. An entity relationship diagram (Figure 1) was constructed for the raw data to help visualise and understand the relationship between the data sources.

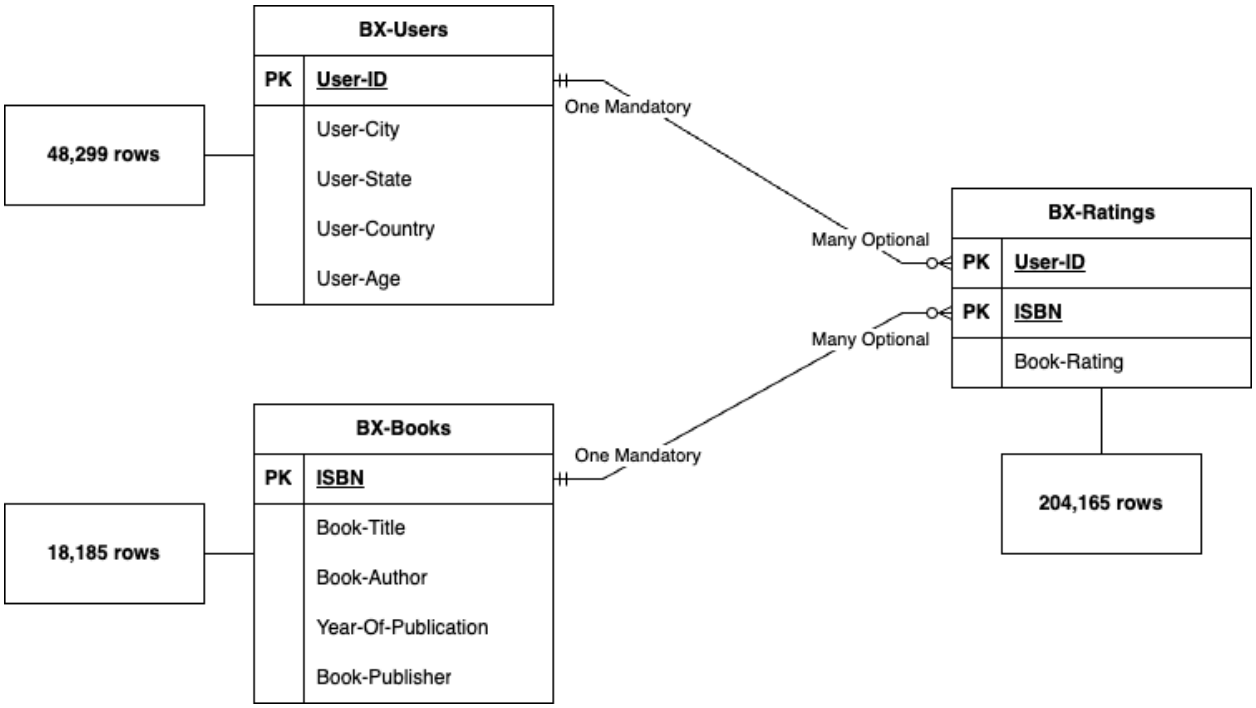

Figure 1: Entity relationship diagram showing the relationship between the raw data files. Notably, the BX-Ratings file has a Composite Foreign Primary Key consisting of the User-ID from the BX-Users file and the ISBN from the BX-Books file.

Before beginning data analysis, there were a number of opportunities to clean the raw data. Case standardisation to title case was applied to each column of the BX-Books.csv and BX-Users.csv containing string values; the User-City, User-State, and User-Country columns in the BX-Users.csv file contained all lower-case values, and the Book-Title and Book-Author columns in the BX-Books.csv file contained some capitalised strings. All missing data, such as empty column values and "N/A", was imputed with a None value as per Python conventions. Additionally, all values in the User-Country column had four erroneous double-quotes which were removed during preparation of the data. These cleaning steps were applied to the raw data by using the df.map() function to apply our clean_value() helper function to the relevant columns during the initial ingestion of the raw csv files as dataframes. The cleaned data was written to the output directory as csv files and also returned as a Pandas DataFrame for use within the program.

Basic general information was then gathered about users, books, authors, and locations by analysing the raw book, user, and rating data (refer to Table 1). For users, the average book rating, the total number of ratings, the highest rated book and rating value, and favourite author (author with the highest average rating) were calculated and written to the UserInfo.csv file in the output directory. For books, the average

rating, highest rating, lowest rating, and number of ratings were collected in the BookInfo.csv file. The total number of books, the average rating, and the total number of ratings of each author were gathered in the AuthorInfo.csv file. Finally, location information was written to the LocationInfo.csv file, containing the highest rated book and rating value, and the most popular book and the number of ratings per city.

Table 1: Extract from AuthorInfo.csv file, containing values calculated from the raw data set.

Book-Author	Book-Count	AverageRating	TotalRatings
Stephen King	195	7.879557751117380	4251
Nora Roberts	127	7.630633931843170	2729
Danielle Steel	100	7.239309827456860	1333

Then, several machine learning techniques were applied to gain deeper insights into the data.

Firstly, **item-item based collaborative filtering** was implemented to understand which books to recommend to a user based on the similarity rating of other books compared to either their favourite or a selected book. The SciPy library was used to construct the initial user-item matrix where each cell (i, j) contains the rating given to each book j by user i. The matrix was then converted to a compressed sparse row matrix where only the non-zero values were stored, significantly reducing memory usage. The compressed user-item matrix was transposed to switch the items and users, and the cosine_similarity function from the scikit-learn library was used to create the final item-item similarity matrix.
Then, the accuracy of the item-item based collaborative filtering was evaluated by calculating the Root Mean Squared Error (RMSE) of its predictions. Since the intended test data (BX-NewBooks and BX-NewBooksRatings) contains books that don't exist in the original dataset, we split the original BX-Ratings dataset into a training set (90%, around 180,000 rows) and a test set (10%, around 20,000 rows) to calculate the RMSE. This 90/10 split was chosen to ensure the model had enough training data to avoid underfitting, while retaining enough test data to avoid overfitting and strengthen the confidence in the performance evaluation. The calculated RMSE of 0.2008 shown in Figure 2 suggests that the model is accurate in its predictions and generalises well to unseen data.

Figure 2: Root Mean Squared Error calculation from the program output.

Additionally, an **unsupervised machine learning technique, clustering**, was performed to group books with similar titles to one another. Data preprocessing is performed on titles here, utilising different libraries such as NLTK and pandas. We first normalise each title by converting it to lowercase and removing punctuation marks to minimise text noise. We then tokenize the titles, breaking them into individual words. Stop words were subsequently removed to exclude commonly used but uninformative words. Finally, each word is stemmed and lemmatized using PorterStemmer and WordNetLemmatizer respectively. This process reduces words to their basic forms, enhancing the clarity and relevance of the data for subsequent machine learning tasks.
After preprocessing, we apply unsupervised clustering to classify the relevance of book titles. We use TF-IDF vectorization to convert textual data into numeric format. TF-IDF vectorization was chosen because it effectively weighs the importance of each word in the corpus, highlighting words that are

important in a specific document but uncommon overall. This is crucial for distinguishing thematic content within different book titles. This data transformation enables us to perform complex analysis of text.

We used the K-means algorithm through the Scikit-learn library to classify titles into 900 clusters based on their TF-IDF features. This number was chosen to balance detail and computational efficiency. The K-means clustering algorithm was adopted because of its ability to efficiently handle large data sets and its ability to effectively form distinct, interpretable groups, which makes it well suited to be applied to our data set. To ensure that each cluster was of a suitable size for meaningful analysis, we refined the clusters by merging smaller clusters and dividing larger clusters, using the Euclidean distance between cluster centres as a guide . The clustering results are logged back to the original file, which contains the clusters for each title. Finally, through these methods, we can achieve association and recommendation based on the user's favourite books.

Fuzzy string matching using Levenshtein distance was also implemented to improve our book and author search feature by suggesting the best matches when a user searches for a value missing from the data. Initially, user-provided book titles are standardised to title case using a helper function, title_case, which capitalises the first letter of each word. Furthermore, if the exact title isn't found in our dataset, the system employs the fuzzywuzzy library's process.extract method to find the closest matches. This function compares the user's input against all book titles in our dataset and returns the top five matches based on their similarity scores. This approach ensures that users are provided with viable alternatives, significantly enhancing the likelihood of discovering their desired book, even if the initial input was slightly incorrect.

Finally, an interactive terminal program was constructed to create a simple interface that allows the user to explore, analyse, and interpret the data by querying the datasets for some of the various information collected.

Data Exploration, Analysis, Discussion and Interpretation

Data exploration was conducted by constructing an interactive terminal program that takes user input and returns insights gained through data analysis. The terminal program represents an interface with some of the functionality expected of the online bookstore, such as a recommendation algorithm, basic user profiling, a list of the most popular books, and the ability to search the inventory of books and authors.

Interactive terminal program

The main program loop consists of three branches (Figure 3). The first branch, an implementation of the book recommendation system, takes a User ID as input and returns recommendations as a list of the top 5 most similar books to the user’s favourite book through item-item collaborative filtering. Branches two and three give the user the option to obtain some general information gathered about books and authors such as the highest rated and most popular book and author, and the ability to search for specific books and authors. Upon initialisation, the program presents the Main Menu with the above options and awaits user input to navigate the program.

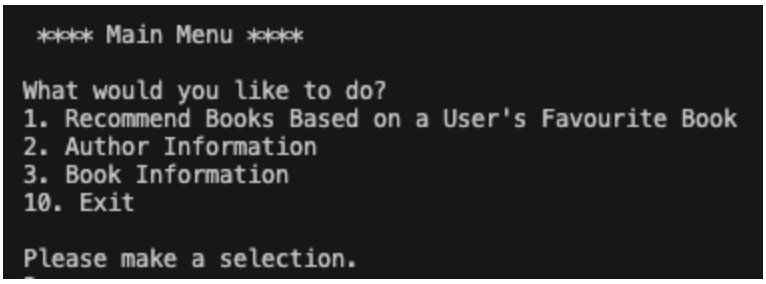

Figure 3: The Main Menu of the program with the three branches for the users to choose from.

Book Recommendation System

The item-item based recommendation system is able to recommend books by searching for a specific User ID, or by searching for a book by ISBN or title.

The first option of searching by User ID is intended to simulate part of a user management system by creating a user profile and offering personalised recommendations. The program will return the top five books with the highest calculated similarity ratings based on the user's favourite book. The program also returns various user information such as the user's favourite book and the rating they gave that book, favourite author (by highest average rating), the user's average rating, and the total count of ratings by that user. Figure 4 shows an example where User 4809, whose favourite book is *Harry Potter and the Chamber of Secrets,* is recommended several books from the *Harry Potter* series by the model. This reflects the expected output, where the user is recommended other *Harry Potter* books which are likely to be given similar ratings by users who also liked *Harry Potter and the Chamber of Secrets*.

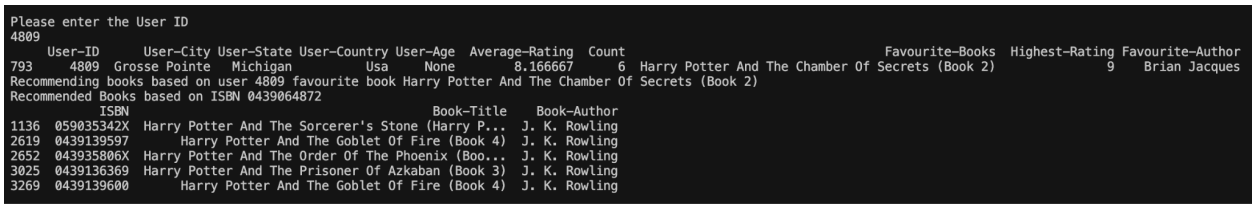

Figure 4: Option 1 from the Main Menu. User information and book recommendations are generated by searching for User 4809. The user's favourite book is *Harry Potter and the Chamber of Secrets* and is expected to also enjoy several other *Harry Potter* books.

The other option is to search for a specific book by ISBN or book title to get book recommendations. This might be implemented by the online bookstore as a recommendation algorithm that provides links to books similar to the currently viewed book on a web page. Again, the results shown in Figure 5 are intuitively correct; when searching for *Meineid: Roman* by ISBN, a German book, the program recommends five other books also written in German.

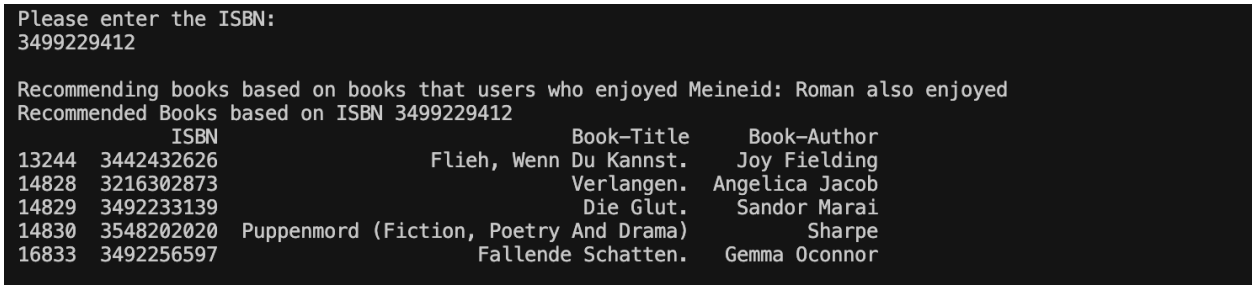

Figure 5: When searching for a German book *Meineid: Roman* by ISBN, the program recommends other German books.

Next, the accuracy of the item-item based collaborative filtering recommendation system was evaluated by calculating the Root Mean Squared Error (RMSE) for predictions made on a test and training dataset. From a 90/10 split on the original Ratings dataset to create the test and training data, the RMSE was calculated to be 0.2008 (Figure 2). Figure 6 shows a visual representation of the actual ratings given by a user against the predictions made by the model. Although some outliers exist, the predicted ratings can be seen to group closely around the trendline of perfect predictions.

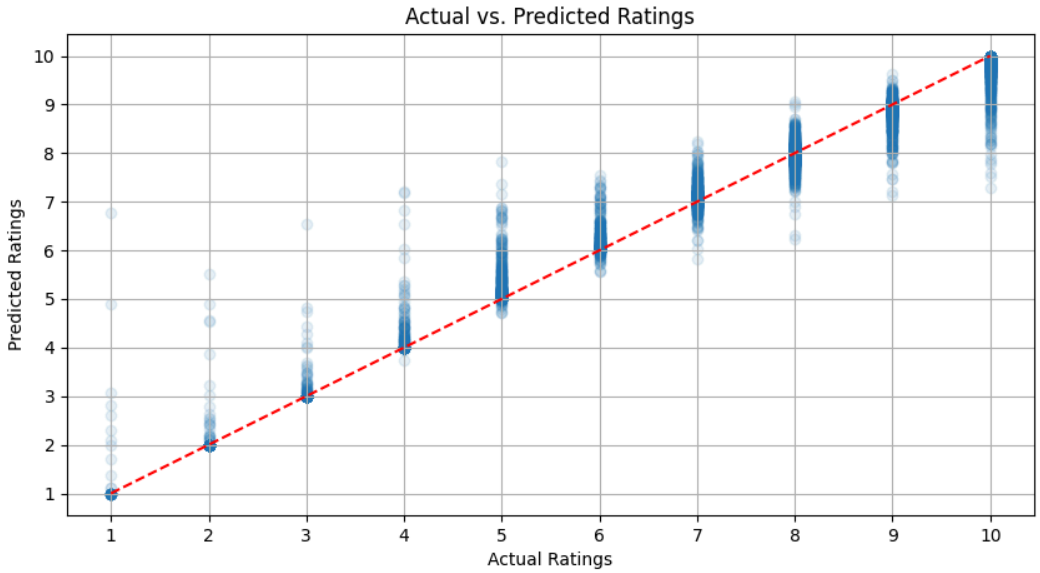

Figure 6: Scatter plot of the actual ratings given to books by users against the predicted ratings assigned by the model. The red line represents perfect predictions where the model predicted the exact rating given by the user.

This analysis validates the intuition that the predictions made in Figure 4 and 5 are good recommendations, suggesting the model is very accurate in its predictions and can be used as a reliable recommendation system for the bookstore.

General information about books and authors

Options 2 and 3 from the Main Menu (Figure 2) can return information about books and authors, including the top ten highest rated and the most popular books/authors. These might represent the selection of books that would appear on the homepage of the online bookstore to get the most user retention.

The highest rated books and authors were calculated by sorting the average rating calculated in the BookInfo and AuthorInfo files generated during initial data processing (refer to Table 1). After some deliberation, the data was limited to only books/authors with more than 30 ratings to avoid outliers where the average rating was skewed by a very small number of reviews.

Figure 7 shows the adjusted output for the top ten highest rated books, which includes a diverse range of highly rated books such as extremely popular novels from the *Harry Potter* and *Lord of the Rings* series, a Pulitzer Prize winner in *Lonesome Dove*, and *A Tree Grows in Brooklyn,* a bestselling American classic autobiography originally published in 1943.

Top Ten Highest Rated Books (Minimum 30 Ratings)

	ISBN	Book-Title	Book-Author	Year-Of-Publication	Book-Publisher	Average Rating
2075	0345339738	The Return Of The King (The Lord Of The Rings,...	J.R.R. Tolkien	1986	Del Rey	9.397260
1877	039480001X	The Cat In The Hat	Dr. Seuss	1957	Random House Books for Young Readers	9.312500
2619	0439139597	Harry Potter And The Goblet Of Fire (Book 4)	J. K. Rowling	2000	Scholastic	9.311111
4148	043936213X	Harry Potter And The Sorcerer's Stone (Book 1)	J. K. Rowling	2001	Scholastic	9.235294
693	067168390X	Lonesome Dove	Larry Mcmurtry	1988	Pocket	9.212121
428	0345339711	The Two Towers (The Lord Of The Rings, Part 2)	J.R.R. Tolkien	1986	Del Rey	9.135802
1916	0064400557	Charlotte's Web (Trophy Newbery)	E. B. White	1974	HarperTrophy	9.123077
737	006092988X	A Tree Grows In Brooklyn	Betty Smith	1998	Perennial	9.096774
1430	0618002219	The Hobbit: Or There And Back Again	J.R.R. Tolkien	1999	Houghton Mifflin Company	9.078947
5469	0618002227	The Fellowship Of The Ring (The Lord Of The Ri...	J. R. R. Tolkien	1999	Houghton Mifflin Company	9.071429

Figure 7: Top ten highest rated books with a minimum of 30 ratings

The most popular books and authors were found by simply sorting the BookInfo and AuthorInfo by the highest number of user ratings. Figure 6 shows the top ten most popular authors by number of ratings.

Top Ten Most Popular Authors (By Number of Ratings)

	Book-Author	Book-Count	AverageRating	TotalRatings
0	Stephen King	195	7.879558	4251
15	John Grisham	54	7.557486	3453
1	Nora Roberts	127	7.630634	2729
25	James Patterson	45	7.709879	2237
89	J. K. Rowling	25	8.984185	1644
47	Michael Crichton	33	7.538267	1581
13	Anne Rice	54	7.500000	1504
8	Mary Higgins Clark	65	7.574766	1498
59	Janet Evanovich	30	7.952449	1409
11	Dean R. Koontz	56	7.552764	1393

Figure 8: Top ten most popular authors by number of ratings. Other useful information such as total book count and average rating is also shown.

The results from Figures 7 and 8 might be used by the online bookstore to populate the items shown on the front page of their website to unknown or new users to display books that appeal to the widest audience possible. Additionally, the average rating and the total number of ratings can be used to provide additional context to the user when shown alongside the book.

Searching for specific books and authors

Options 2 and 3 from the main menu also give the user the ability to search for a specific book or author, representing the search feature of the online bookstore. Searching for a book or author returns basic information about the item.

An important feature of our search function is its ability to effectively handle incorrect user input. This allows the user of the online bookstore to more consistently find the results they want. Firstly, case sensitivity was handled in the case of searching for a specific book title or author name by matching the user input to the title case applied during data preprocessing. When a user searches for a book title, ISBN, or author name that does not exist in the data set, the closest matching results are suggested. Figure 9 shows an example where if a user mistypes the author name in their search, the system utilises fuzzy string matching algorithms to find and suggest the closest matches in the database. This ensures that user can still find the results they are looking for despite minor errors in their input.

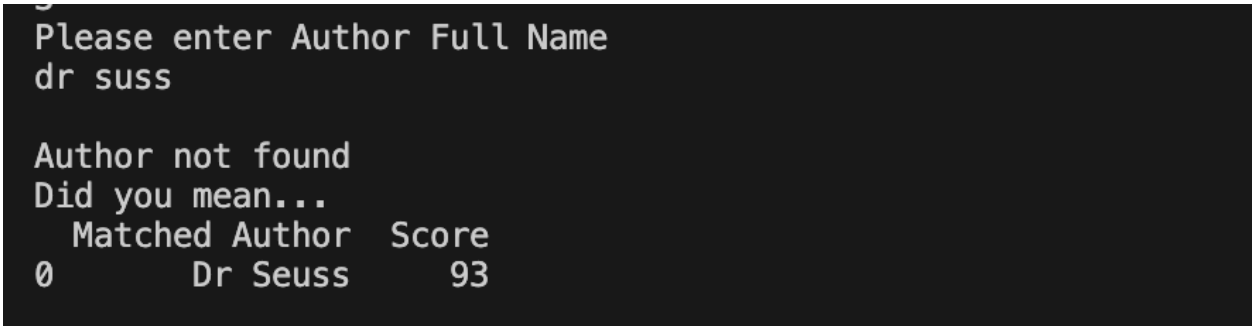

Figure 9: Example of using fuzzy string matching to guess the user's intended search result.

Additionally, the program is also able to suggest books from the same cluster as the book they searched to widen the range of their search results in a calculated manner. Functionally, this would be used in the search feature implemented by the online bookstore to offer a variety of related search results. In practice, the usefulness of clustering algorithms is demonstrated by its ability to recommend books from the same cluster when a user shows interest in a particular title. For example, when a user searches for a book title, the program retrieves other books from the same cluster that may be of interest to the user. Figure 10 shows that when a user enters the title *Old Man and the Sea*, its cluster is identified and up to five other books in the same cluster are listed that contain similar keywords to the original search.

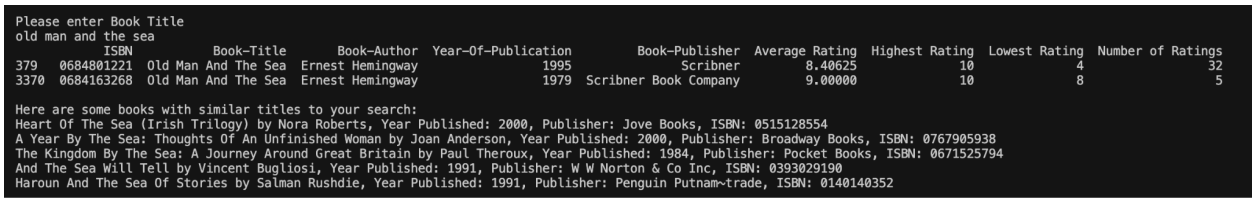

Figure 10: Clustering used to provide recommendations based on similarity to the book title. Searching for *Old Man and the Sea* offers recommendations for other books with the keyword "sea" in the title.

These recommendations are based on book title categories only. Clustered book title data analysis and its application to our bookstore system clearly demonstrates how unsupervised machine learning can improve user experience and efficiency, as well as improving the variety of search results that the bookstore can offer.

The cumulative distribution plot of the number of clustered books illustrates the distribution of book titles among clusters (Figure 11).

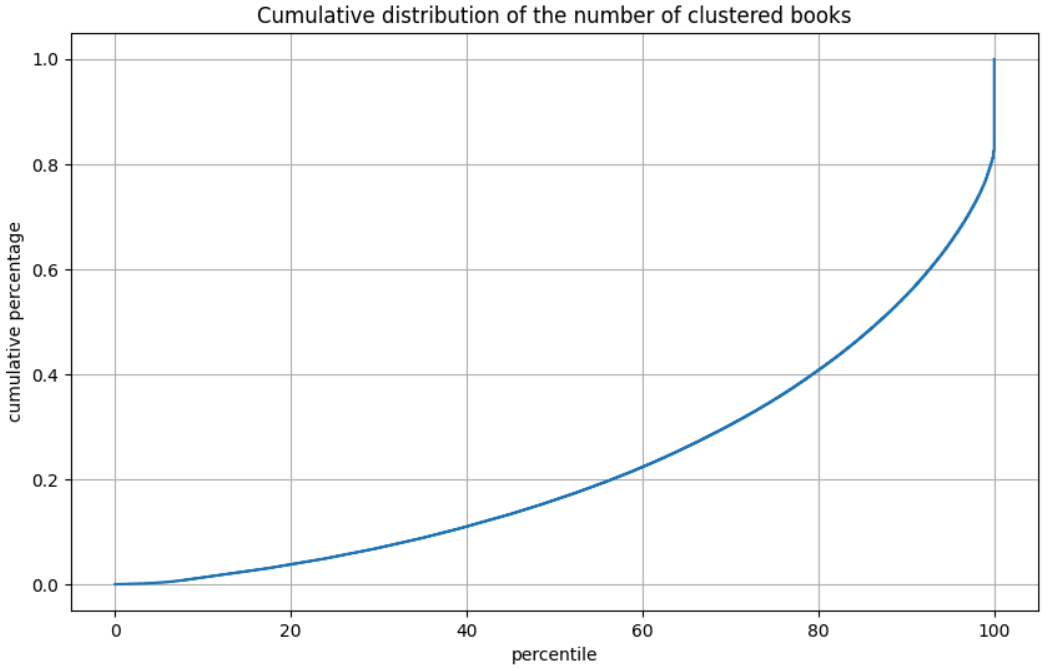

Figure 11: This function plots the cumulative distribution of clustered books by calculating the cumulative sum of books per cluster and displaying it against their percentile ranking.

As shown, the cumulative percentage of books in each cluster gradually increases until it rises sharply around the 80th percentile. This shows that while most clusters are relatively small and contain few titles, a few clusters contain a disproportionately large number of titles. This skew in the distribution highlights the challenge of cluster balancing and demonstrates the need for adjustments during clustering to ensure each cluster is meaningful and manageable.

Results

We were able to provide value to the online bookstore in a number of ways by applying a variety of data analysis and machine learning techniques to their existing data. The results of these techniques could be implemented by the bookstore to significantly improve the user experience.

The accuracy of the item-item based recommendation system (Figure 2, Figure 6) shows the potential of this approach in the context of implementing a book recommendation system. We were able to make precise predictions for a user's rating of a book by analysing the ratings given by other users. The bookstore can use this model to provide better, more personalised recommendations to their customers, improving the customer experience, user retention, and increasing book sales. However, the item-item recommendation system suffers from the cold-start problem, potentially requiring further analysis and incorporation of user and location data in the model to help make predictions for new books.

The general information provided about books and authors (Figure 7, Figure 8) reveal broad insights into the books and authors which can again be utilised to improve the customer experience. Metrics such as the highest rated author and the average rating for a book can provide valuable context to customers. Furthermore, the managers of the bookstore can use these insights to help make informed decisions when

managing their inventory. Further analysis of the existing data, such as user location information, could be used to yield even greater insights.

The clustering and fuzzy search algorithms implemented are valuable supplementations to the search feature. By providing the closest match to the user's search input we can increase the probability that the user can find the book or author they were searching for, even if an exact match does not exist. Additionally, alongside the recommended books provided by the recommendation system, we can offer alternative yet still relevant book options to the user through the clustering algorithm, providing variety in the sort of related books the bookstore can recommend.

We were thus able to create value through the use and implementation of the above data processing techniques on the existing data by creating a book recommendation system, a robust search feature, and providing useful information about books and authors.

Limitations and Improvement Opportunities

There were a number of limitations and improvement opportunities identified during this project.

The program itself can be optimised to reduce load times. Currently, the data cleaning and pre-processing occurs every time the main program is executed. This pre-processing can be completed prior to program execution to eliminate loading times.

The underlying data also still has a number of issues:

- Location information is inconsistent: e.g. "Usa" and "America" are the same country but are treated as distinct locations by the program.
- The same authors are recorded with slightly different names: e.g. J.K. Rowling & J. K. Rowling, J R R Tolkien & J.R.R. Tolkien

The item-item based recommendation system does not easily extend to new books. Because the method relies on existing user ratings to make predictions, it suffers from the cold-start problem where it is difficult to make predictions about new books due to the lack of available information about the user interactions with the new book. Currently, user and book information is not considered when making recommendations, only the similarity between items is evaluated. However, this data could be incorporated into the recommendation system approach to partially remedy this issue.

It is observed from the cumulative distribution of clustered books that there is a significant bias in the size of the clusters, with a few clusters containing a large number of books. This situation may affect the effectiveness of clustering for recommendation purposes, as smaller clusters may not provide enough options, and larger clusters may lack specificity. The following options could improve the results from clustering:

- Static clustering method: Choose to use a fixed number of clusters (900) and assume that subject content is evenly distributed among book titles, which may not be true. This approach can lead to misgrouping, where books on different topics are forced into the same cluster due to similar word usage patterns.
- Dynamic clustering: Implementing a more dynamic clustering approach can allow the number of clusters to be determined based on the inherent structure of the data, rather than being set in advance. This may lead to more meaningful and organically formed clusters.
- Post-clustering optimization: Introducing a post-clustering optimization stage to re-evaluate and adjust clusters based on additional indicators such as intra-cluster similarity and inter-cluster differences can improve clustering accuracy. Additionally, leveraging user feedback to dynamically adjust a cluster can also improve its accuracy and relevance over time.

Conclusion

In this report, we have outlined the various aspects in which we have created opportunities to provide value to the users and managers of the online bookstore. The item-item based recommendation system, clustering to group similar book titles, the general information obtained from the data, and the search feature improved by fuzzy string matching, can all be utilised to achieve the main objective of aiding the bookstore in improving customer satisfaction and experience.

Item-item based collaborative filtering proved to be an effective way to recommend books; the calculated RMSE of 0.2008 suggests the model makes reliable and accurate predictions. General data analysis techniques yielded valuable insights into the underlying data, such as the highest rated and most popular books and ratings, as well as their average ratings. These insights can be incorporated into the information displayed alongside a book or author to provide context to the user about how other users perceive the item, and can also be used by managers to help make informed decisions about their inventory. Finally, clustering and fuzzy string matching significantly increased the functionality of the search feature by offering the user alternative results based on their search input, greatly increasing the likelihood that they are able to find their desired result.

The item-item based collaborative filtering recommendation system, insights gained from analysis of the existing data, and the clustering and fuzzy search implementation in our searching algorithm combine to provide a valuable set of insights for the bookstore to continue to delight their customers.