READ RADAR:

BOOK RECOMMENDATION SYSTEM

IE 6400 PROJECT REPORT

STUDENT NAME: RIDA KHAN

STUDENT ID: 002804008

PROFESSOR: ANDY CHEN

# TABLE OF CONTENTS

[**TABLE OF CONTENTS 2**](#_51tqxawj1dkc)

[**BUSINESS UNDERSTANDING 3**](#_2776htaw6gxh)

[Introduction 3](#_anmpq2slzkyo)

[Background 5](#_br1t5kmavr2p)

[Business and Data Mining Goals 5](#_78nq3ahq9mnj)

[**DATA UNDERSTANDING & PREPARATION 6**](#_o2gez7bgjyh2)

[Titles Dataframe: 9](#_c8kav9hu0yf2)

[Genres Dataframe: 9](#_qwtjgiardyeu)

[Reviews Dataframe: 9](#_qe0gjqh95vsy)

[**MODELING 10**](#_afmln5zh8uta)

[Logistic regression to predict if a book is high rated 11](#_79jlghliomqn)

[Text analysis for search query 11](#_561yn59q9lep)

[Collaborative filtering for recommender 12](#_mjvzeyo5w5n4)

[**RESULTS & EVALUATION 12**](#_ff43s7v7k8yk)

[**RECOMMENDATIONS 13**](#_twwvz3620nb)

[**REFERENCES 15**](#_5m9uowhjngjt)

[**APPENDIX 16**](#_y3n264gxxbnn)

[Exploratory Data Analysis 16](#_xr9jn3i9t73q)

[Results 20](#_jhkxmtxa89ql)

[FrontEnd 22](#_c5frqxxglnhr)

# 

# 

# BUSINESS UNDERSTANDING

### **Introduction**

In the present fast-paced era, the rapid and instantly gratifying lifestyle has emerged as a result of technological advancements and the AI revolution. People around the world simply require to convey their concern and receive thousands of solutions in mere seconds. Especially with the advent of artificial intelligence tools and large language models, any individual can do a wide range of tasks ranging from generating art, to streamlining business flows. While the ability to acquire information about virtually anything is certainly quite advantageous, the ability to filter through trillions of bytes of data and being able to access only the most relevant information is a lot more efficient. In lieu of this, a very prominent feature in most applications or websites that we see is the ability to receive personalised content in the form of recommendations. For the purpose of this report, we have chosen to take into consideration only the book community and design a system that essentially provides book recommendations.

Recommendation systems are ubiquitous in nature and have a role in nearly every industry tool imaginable, ranging from e-commerce (Amazon) and entertainment (Netflix) to job searching (Indeed) and even reading (Goodreads). Since these systems are so prevalent, in terms of implementation, they can be classified into three broad types, namely, content-based, collaborative filtering based and hybrid recommender systems. While these can be implemented using deep learning concepts, such as Restricted Boltzman Machines and Auto Encoders as well, we will mainly focus on the Machine learning aspect here.

**Content-Based Recommender Systems:**

This type of recommendation system involves reviewing the history of a user’s previously liked choices and using them to generate new recommendations based on the features of that item that the user reacted to positively. In the case of our datasets, the use case could potentially involve a preference of a certain type of genre or combination of genres per user. One could like or show interest in majorly fiction and fantasy books in order for the recommender system to suggest more fiction and fantasy books that they haven’t read, another feature that could be selected is average rating or even publishers. However, in order to achieve a more personalised recommendation, the genres alone would suffice.

**Collaborative Filtering Based Recommender Systems:**

This is the type of recommender system that does not rely on the item’s features directly and instead uses relations and similarities between different user’s liked lists in order to generate recommendations. In the case of our book recommender system, this would work by taking in our input list of liked books, finding other users with the same liked books then recommending books to us based on their lists. This implies that this system considers other similar users’s preferences while recommending a book.

Additionally, there are various other techniques that can be used in order to create a recommender system, however, a very common method is the hybrid method where we can utilise multiple types of models in order to generate our recommendations and make them more similar and accurate.

### **Background**

The business problem that this project addresses is one that the reading community faces, namely, the creation of a personalised book recommendation system. This project is potentially a great business idea for an application that could be used by readers, libraries, publishers and authors that will allow each of these actors to determine what the most potentially profitable, readable books are. While, in a more comprehensive sense, this project can be scaled to have specific features per stakeholder, due to time constraints, the scope has been contained to address readers.

Our book recommendation system aims to enhance the reader experience and potentially increase customer engagement, retention and possible revenue generation and a successful phased implementation of the same can position this platform as a leader, fostering a strong and loyal user community. This project specifically leverages data from Goodreads including book titles, genres, publication information, ratings and reviews from goodreads users.

The book recommender project will predict what books a user would prefer reading based on their previously selected liked books, other users who have liked and highly rated these books and their liked books as well. Moreover, we will also implement a model to predict the possible average ratings for a book. Most importantly, each implemented machine learning model to predict the best books for recommendation will be evaluated using metrics in order to determine the most effective one.

### **Business and Data Mining Goals**

In order to better understand the complexities and intricacies of this system it is essential to determine what our business and data mining goals are so that we have a certain standard against which we can evaluate if these have been achieved at the end of the project. The primary business goal for this system is essentially to increase user engagement for all sectors (readers, writers and publishers) through personalised recommendations.

The rationale behind this is that personalised recommendations can initiate customer growth and retention through providing a tailored reading journey for readers, generating insights on popular content, genres and tropes for writers and publishers as well. In order to achieve this business goal a few strategies that could be employed are - user profiling, dynamic recommendation algorithms, user feedback integration, enhanced user interfaces, and gamification elements as well.

Additionally, apart from the business goals, the fundamental data mining goal would be to predict reader preferences based on historical interactions data, demographic data, and various book features to determine the best fit for recommendations.

The general data sources for this purpose remain the same as mentioned above in the business goal section however the approach here to achieve our goal involves the use of modelling techniques such as content based filtering, collaborative filtering and hybrid models.

Most importantly, as these goals are being achieved we should strive for continuous improvement and innovation in order to keep this system updated and working to the best of its ability.

# DATA UNDERSTANDING & PREPARATION

To begin with, the datasets in their unedited form were acquired through this website, in the datasets section: <https://mengtingwan.github.io/data/goodreads.html>

For the sake of this project only the following datasets were utilised, and all this data was scraped from goodreads:

1. goodreads\_books.json.gz: Contains detailed information on about 2M books
2. goodreads\_book\_genres\_initial.json.gz: Extracted fuzzy book genres (genre tags are extracted from users' popular shelves by a simple keyword matching process)
3. goodreads\_interactions.csv: Complete user-book interactions
4. book\_id\_map.csv: Contains user id and book id map

In order to load the datasets, we used a different approach from the typical pd.read\_csv() command that we are well acquainted with. The reason for this was because we had a mix of both json and csv files and because we had very heavy data.

In this case, we needed to use a more scalable and memory efficient technique instead of loading the entire dataset, and that happened to be streaming the data line by line.

Moreover, it proved better to parse through the lines of data and select only the parts that we want to use for our dataset, and in order to implement this effectively, we used a parsing function.

**Parsing Function for titles dataset:**

def parse\_books(line):

data = json.loads(line)

return{

"book\_id": data["book\_id"],

"title": data["title\_without\_series"],

"ratings": data["ratings\_count"],

"average\_rating": data["average\_rating"],

"url": data["url"],

"cover\_image": data["image\_url"],

"num\_pages": data ["num\_pages"],

"publication\_year":data["publication\_year"],

"publisher" : data["publisher"],

}

**Parsing Function for reviews dataset:**

def parse\_reviews(line):

data = json.loads(line)

return{

"user\_id": data["user\_id"],

"book\_id": data["book\_id"],

"review\_id": data["review\_id"],

"rating": data["rating"],

"review\_text": data["review\_text"]

}

Upon having the data loaded on to the jupyter notebook, each of the dataframes went through a preprocessing stage to make them usable in order to generate insights for the exploratory data analysis and prepare them for the data modelling phase.

The general steps to handle this section involved running a .info() command to determine basic information about each loaded dataframe and then determining the changes that had to be made followed by repeating the command to verify everything.

### **Titles Dataframe:**

The titles dataframe contained a total of 8 columns that were all string types and had no missing values, however, it was evident that because we converted the json files into a dictionary and then to a dataframe that there may have been hidden missing values in the form of whitespaces. Changes to titles dataframe:

- Ensure 'ratings' column is numeric

- Create a separate modified title for ease to search

- Ensure 'average\_rating' is a float number

- Ensure 'num\_pages' is an int number

- Ensure 'publication\_year' is a date-time year

- Fill in the hidden missing values appropriately

### **Genres Dataframe:**

The genres dataframe contained a total of 2 columns that were all string types and had no missing values, however, it was evident because we converted the json files into a dictionary and then to a dataframe that there may have been hidden missing values in the form of whitespaces.

The primary change was to extract the list of comma separated genres and turn them into boolean features for each book\_id, so that this dataset could be merged with the titles dataset in order to keep only the necessary values.

### **Reviews Dataframe:**

The reviews dataframe had 11 columns in the dataset, out of which we needed only the first 5 - user\_id, book\_id, review\_id, rating and review\_text. Moreover, we needed to ensure that the dataset contained only the book\_id from our main titles dataset. Finally, we needed to create a new feature in order to better predict the high rating using logistic regression, that would be the count of review texts per book, in the titles dataset.

Upon completion of the preprocessing stage, we had two ready datasets, namely titles and cleaned\_genres which could be merged on book\_id as the key column.

That led us to a stage where we could begin asking some questions related to the business context of our data. Since we were working with book related data, the primary stakeholders involved would be, publishers, authors and the reading community. The exploratory data analysis results can be found in the appendix, as linked.

# MODELING

This section of our document contains in depth information regarding the data modelling that was conducted to implement the book recommendation system. Here we have documented the fundamental processing of our data frames and evaluated the best fitted models for our system. After numerous trials and errors, we determined that using linear regression would be rendered ineffective due to the lack of numerical features within the dataset, however, we did settle for a logistic regression model that used the genres along with other features to determine if a book was highly rated or not.

Alongside this classification type of model, we went along with the implementation of text analysis using a term frequency - inverse document frequency matrix (TF-IDF) in order to build the search query functionality.

Finally, we implemented a collaborative filtering model using scipy’s coo matrix import in order to generate the book recommendations from a liked and rated book list of a particular user. Moreover, in order to determine how well these models performed, we chose the following metrics:

**1. Logistic Regression**

- Accuracy, Precision, Confusion Matrix, Summary Statistics

**2. Text analysis using TF-IDF**

- Similarity Score, Domain Knowledge

**3. Collaborative filtering for recommendations**

- Similarity Score, Matrix operations, Domain Knowledge

The code in order to implement this section will be attached in the supporting jupyter notebook, but in order to get a better understanding of each model, let's delve deeper:

### **Logistic regression to predict if a book is high rated**

This model firstly takes into account a threshold value for a high rating, which was chosen to be 3.5 and above. In order to implement this classification, the main features that were chosen were namely, ['average\_rating','text\_review\_count','fantasy', 'paranormal', 'fiction', 'mystery', 'thriller', 'crime', 'poetry', 'romance', 'non-fiction', 'history', 'historical fiction’, 'biography', 'children', 'young-adult', 'comics', 'graphic'] and the independent feature that was to be determined was [‘high\_rating’]. The model was implemented using the statsmodels library import and was coupled with a train-test split. The result metrics for this model are attached in the appendix.

### **Text analysis for search query**

The TF-IDF matrix is a key component that powers this search functionality, which enables users to enter search queries and obtain the most pertinent results from the titles dataframe. The Term Frequency-Inverse Document Frequency technique is utilised by the system to precisely determine the relevance of terms in titles and their overall importance in the dataset. In order to provide a more precise and customised search experience, the search function analyses user queries, computes cosine similarity with the TF-IDF matrix, and finds the highest rated titles. This improves textual matching accuracy while also taking user ratings into account, providing a diverse range of titles for users to explore and interact with.

### **Collaborative filtering for recommender**

Finally, to improve its capacity to make tailored book recommendations, the recommender function integrates collaborative filtering approaches. By utilising user interactions and preferences, collaborative filtering makes use of patterns and similarities to make predictions about a user's preferences based on the activities of readers who are similar to them. Collaborative filtering provides customised and varied suggestions by extending the recommendation system's reach beyond individual user profiles through the analysis of user interactions within the dataset. This cooperative strategy increases the recommendation system's overall efficacy and guarantees a more thorough and personalised reading experience for each user.

# RESULTS & EVALUATION

This section of the document covers the end result of this project alongside its effectiveness. For the purpose of better judgement and display of each feature, a front end for this project was built using streamlit in python.

In terms of individual model evaluation results and metrics, the performance of the logistic regression resulted in 99% accuracy, the precision for both classes was 1 and that indicates that the model almost always predicted correctly, if a book was highly rated or not.

Furthermore, for the TF-IDF text vectorizer model implemented for the search query, the main evaluating metric was the similarity score based on which the best possible fit from the dataframe was returned.

Finally, the collaborative filtering model that was implemented in the recommender function, worked by implementing matrix operations on a user and book matrix to determine the next best read based on the similarity score and top high ratings. The results of this will be attached in the appendix.

The project's tangible accomplishments are summarised in the results and evaluation section, which also emphasises the significance of the system's individual characteristics and overall performance. Through direct interaction and evaluation of the project's results, stakeholders can engage with the streamlit front end as a dynamic showcase.

# RECOMMENDATIONS

It is evident after reviewing the last section of this report that there were particular challenges this project had to overcome. These difficulties resulted in a thoughtful compilation of recommendations and further thinking, each with the intention of enhancing the business and functional aspects of our customised book recommendation system.

The difficulties posed by the "User Cold Start" scenario were especially notable in the context of user engagement. This happens when a new user signs up for the system but we don't know enough about their preferences or usage patterns. Due to little data, our recommendation system's cornerstone, collaborative filtering, struggles to identify related users or make meaningful recommendations for these newcomers.

In the same way, the "Item Cold Start" problem is brought on by the addition of new items to the system. When a novel item is introduced without user interaction data, it can be difficult to find consumers who have the same interests and provide accurate suggestions for such a unique item.

We suggest the following solutions to these problems in order to guarantee our recommendation system's ongoing development. First of all, we support the development of a methodical procedure for "Continuous Model Refinement." Our system will be able to dynamically adjust to changing user preferences and new reading patterns through regular updates to the recommendation models that incorporate fresh user interactions and book data.

Concurrently, it is important to establish a "Feedback Loop" in order to collect user input regarding books that are suggested. Our recommendation algorithms will be refined and improved thanks in large part to this iterative feedback mechanism, which will promote a user-centric approach to system enhancement.

In addition to these user-centred approaches, we propose a deliberate "Collaboration with Publishers" to enhance our dataset with more metadata. This joint endeavour attempts to enhance the calibre of our suggestions by adding more thorough details on books, genres, and writers.

Finally, in order to tackle issues concerning user ratings, we suggest a "User Education on Rating System." In addition to increasing the quality of user input, providing users with instructional resources regarding the significance and implications of their evaluations will result in more precise forecasts and tailored suggestions.

All together, these linked suggestions provide a thorough approach to problem-solving that guarantees our customised book recommendation system will always be improved.

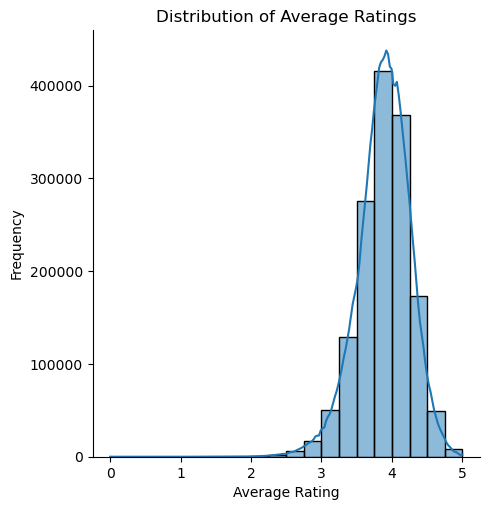
# REFERENCES

* Roy, D., Dutta, M. A systematic review and research perspective on recommender systems. *J Big Data* 9, 59 (2022). <https://doi.org/10.1186/s40537-022-00592-5>
* Mengting Wan, Julian McAuley, "[Item Recommendation on Monotonic Behavior Chains](https://mengtingwan.github.io/paper/recsys18_mwan.pdf)", in RecSys'18. [[bibtex](https://dblp.uni-trier.de/rec/conf/recsys/WanM18.html?view=bibtex)]
* Mengting Wan, Rishabh Misra, Ndapa Nakashole, Julian McAuley, "[Fine-Grained Spoiler Detection from Large-Scale Review Corpora](https://mengtingwan.github.io/paper/acl19_mwan.pdf)", in ACL'19. [[bibtex](https://dblp.uni-trier.de/rec/conf/acl/WanMNM19.html?view=bibtex)]

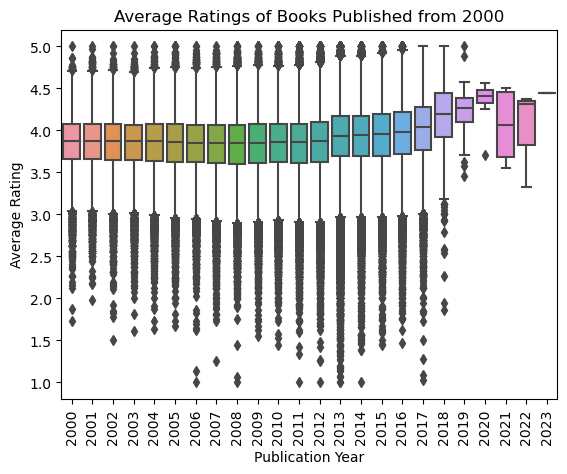
# APPENDIX

### **Exploratory Data Analysis**

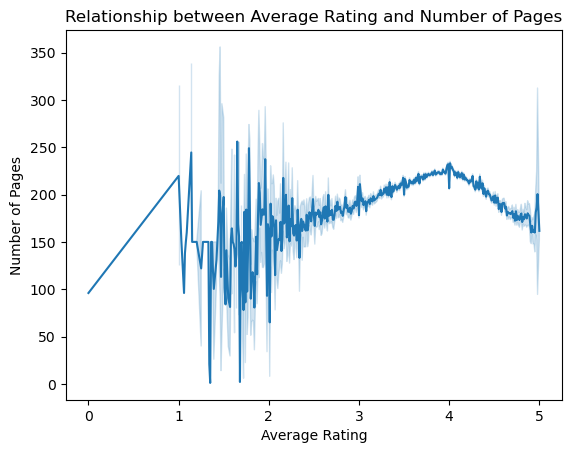
Question 1: What is the distribution of average ratings for books?



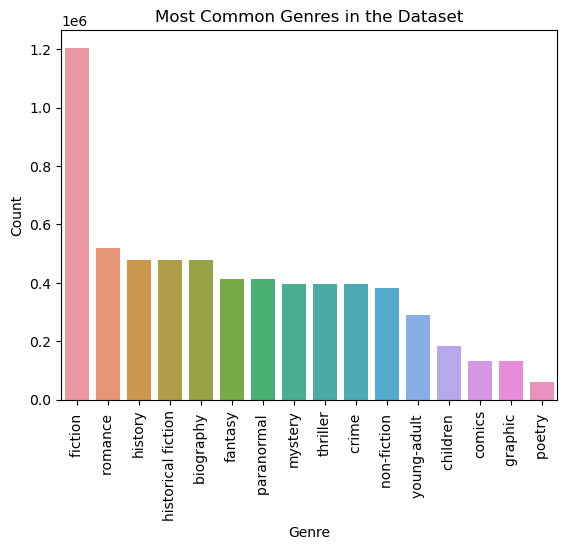
Question 2: How are the average ratings of books distributed by publication year 2000 onwards?



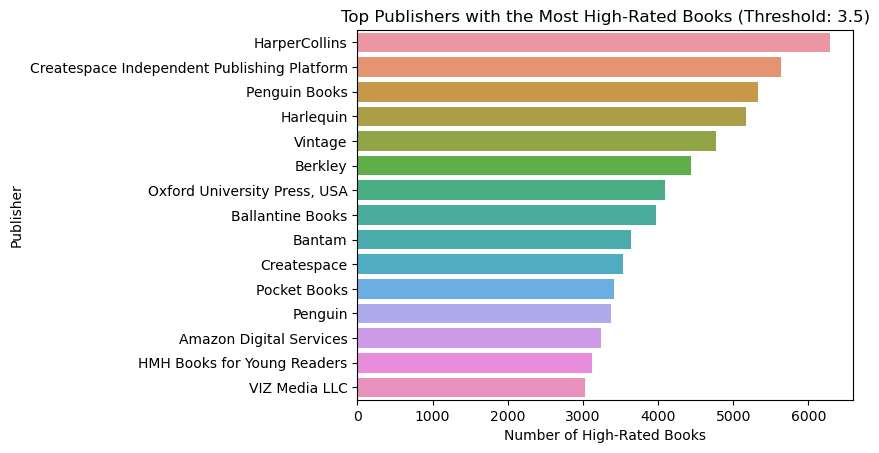
Question 3: Is there a correlation between the number of pages and average ratings of books?



Question 4: What are the most common genres in the dataset, and how do they correlate with user ratings?



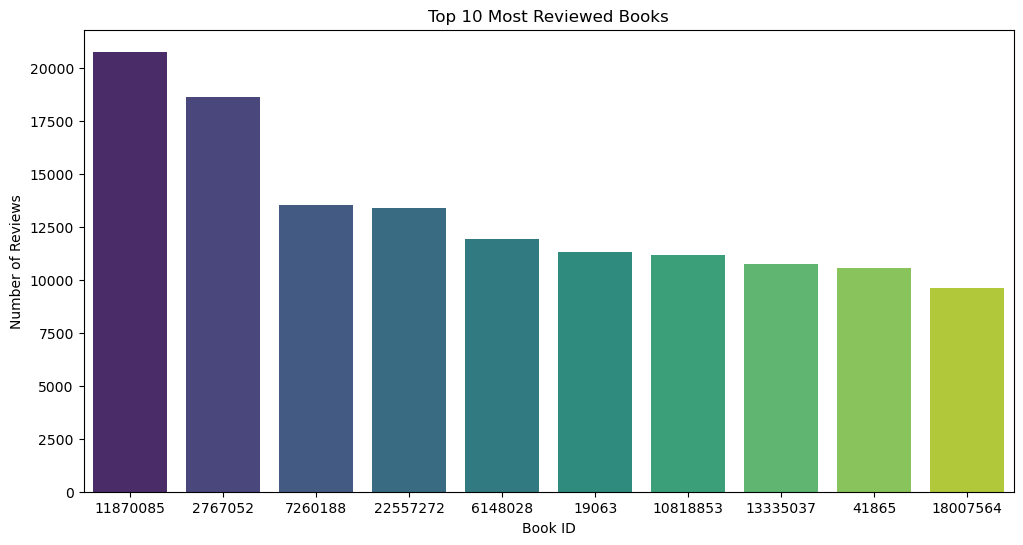
Question 5: Which publishers have the most highly-rated books in the dataset?



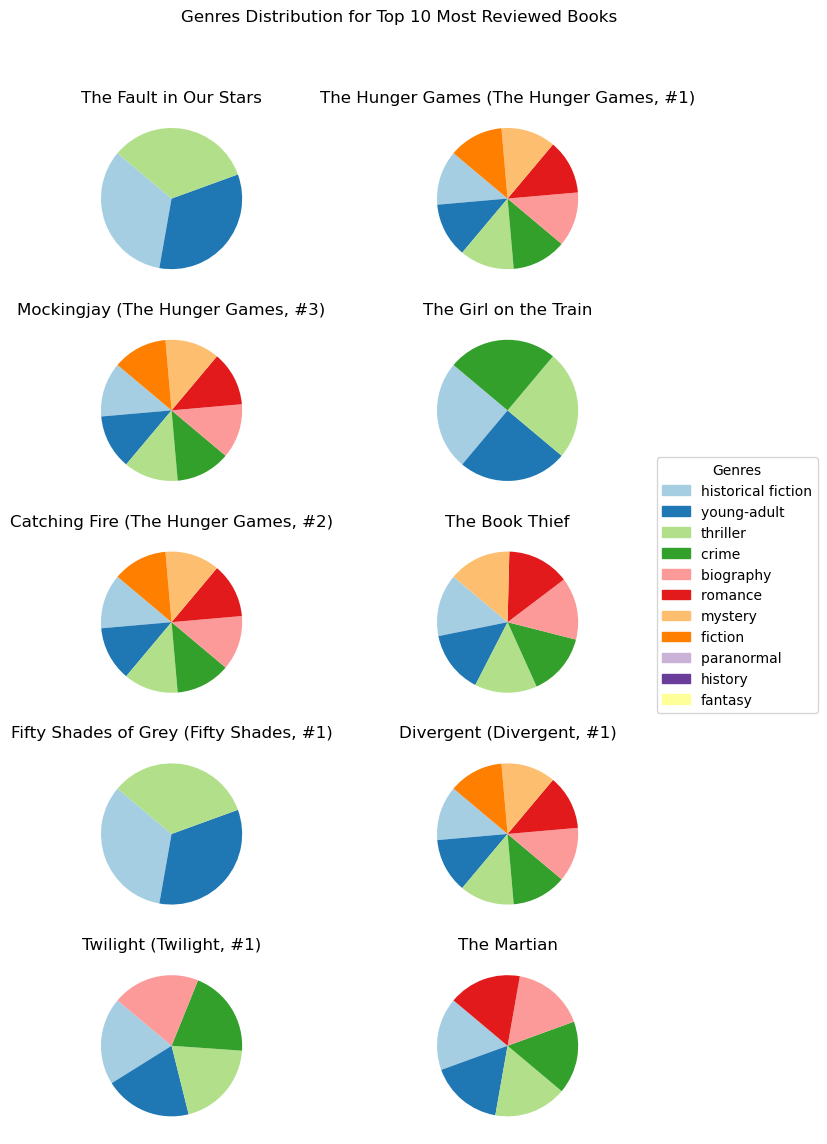
Question 6: How many books have unknown publisher information?

Number of Books with Unknown Publisher: 403628

Question 7: What are the top 10 highest reviewed books?

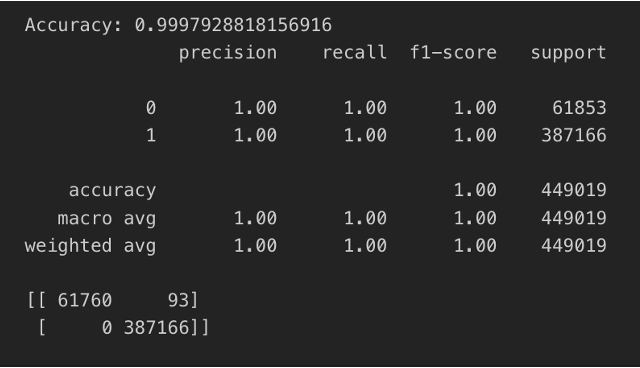
- -

Question 8: What is the genre distribution for the most highly reviewed books?

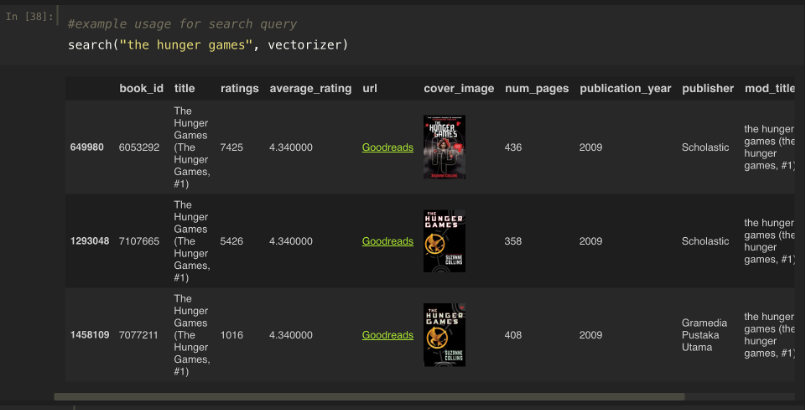


### **Results**

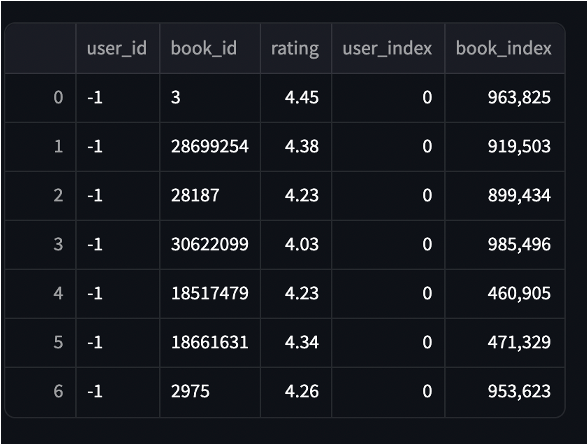
Logistic Regression



Text Analysis



Collaborative Filtering





### **FrontEnd**

