IRON
HACK

Data Analytics

NLP and Machine Learning Based Model for Genre Classification of Books

Berkay Ersoy

March 2023

Table of contents

Introduction

The project aims to analyze metadata of books and develop a machine learning model using NLP techniques to predict the genre of a book based on its description.

Always having the interest to analyze text data brought me to the idea to focus on books this time for my final project. Combining my love for technology and useful tools with my recent habit of reading books, I decided to explore books metadata from Goodreads.com. Metadata are the words and phrases used to describe a book like a title, description, rating, and genre. They are not only limited to attributes of the book but can also be author, author id, author nationality, and more.

Knowing that one of the biggest companies in the world started from selling books and considering numerous businesses that revolve around books, I thought I could analyze and build a model for businesses. My model can be utilized by a variety of entities, from major online corporations to local libraries and bookstores seeking to update their databases with accurate book genres.

Focusing on book descriptions and genres, I constructed my model around natural language processing (NLP) and machine learning, allowing it to predict a book's genre based on its description.

Having explained the end goal, my project is not only limited to the machine learning model but also involves finding and cleaning data, conducting EDA and visualization, creating a database entity relationship diagram, setting up the database, and querying data from it to progress through my end goal.

Plan of project

1. Obtain data from the appropriate sources
2. Load CSV into Python
3. Web scrape additional data
4. Combine the CSV and web-scraped data in Python
5. Clean the data
6. Creating the database in MySQL
7. Export clean data to MySQL for processing
8. Querying the data in MySQL
9. Doing explanatory data analysis in Python
10. Natural language processing
11. Creating the machine learning model

Data and data sources

I was searching for datasets to use for my final project on the internet about books datasets and possibly from Goodreads.com.

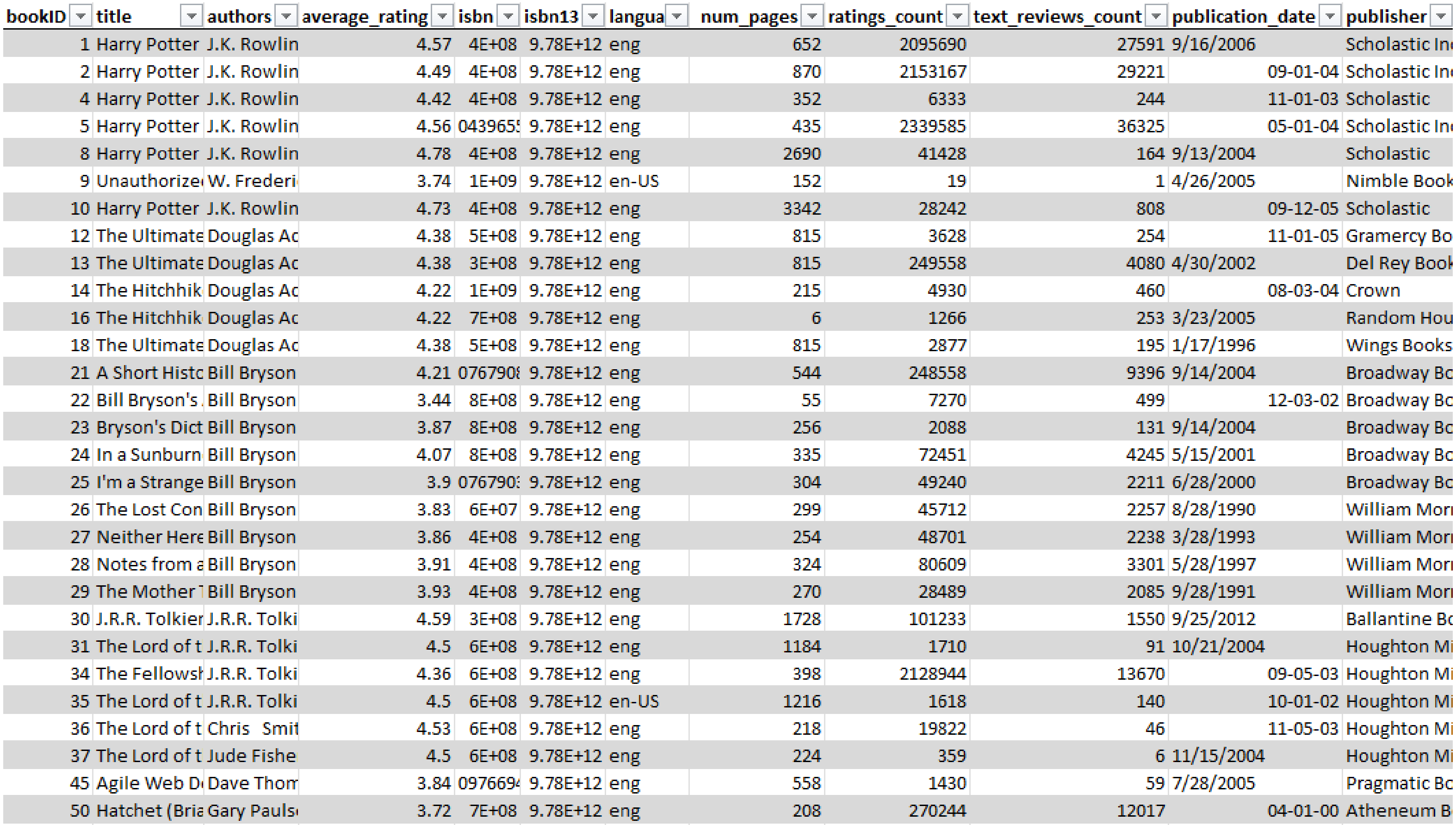
Goodreads is a social networking website that is focused on books. It allows users to create virtual bookshelves, rate and review books they have read, see what their friends are reading, and get personalized book recommendations based on their reading history. Goodreads also allows users to see the average rating of a book based on all the ratings and reviews submitted by its members. Users can also read book reviews to get a sense of what others thought of a particular book before deciding whether or not to read it themselves. The website has over 2.5 billion books added to virtual bookshelves by its members.

I found a dataset on Kaggle consisting of 11127 rows and 12 columns called “Goodreads books”. The dataset fields are:

Field Name	Description
bookID	Unique ID given by Goodreads
title	Title of the book
authors	Authors of the book
average_rating	Average rating of the book (0 to 5)
isbn	The international standard book number (10 digits)
isbn13	The international standard book number (13 digits)
language_code	Languages of the book
num_pages	Number of pages
ratings_count	Number of ratings the book has on the website
text_reviews_count	Number of text reviews the book has
publication_date	Publication date

publisher	Publisher name

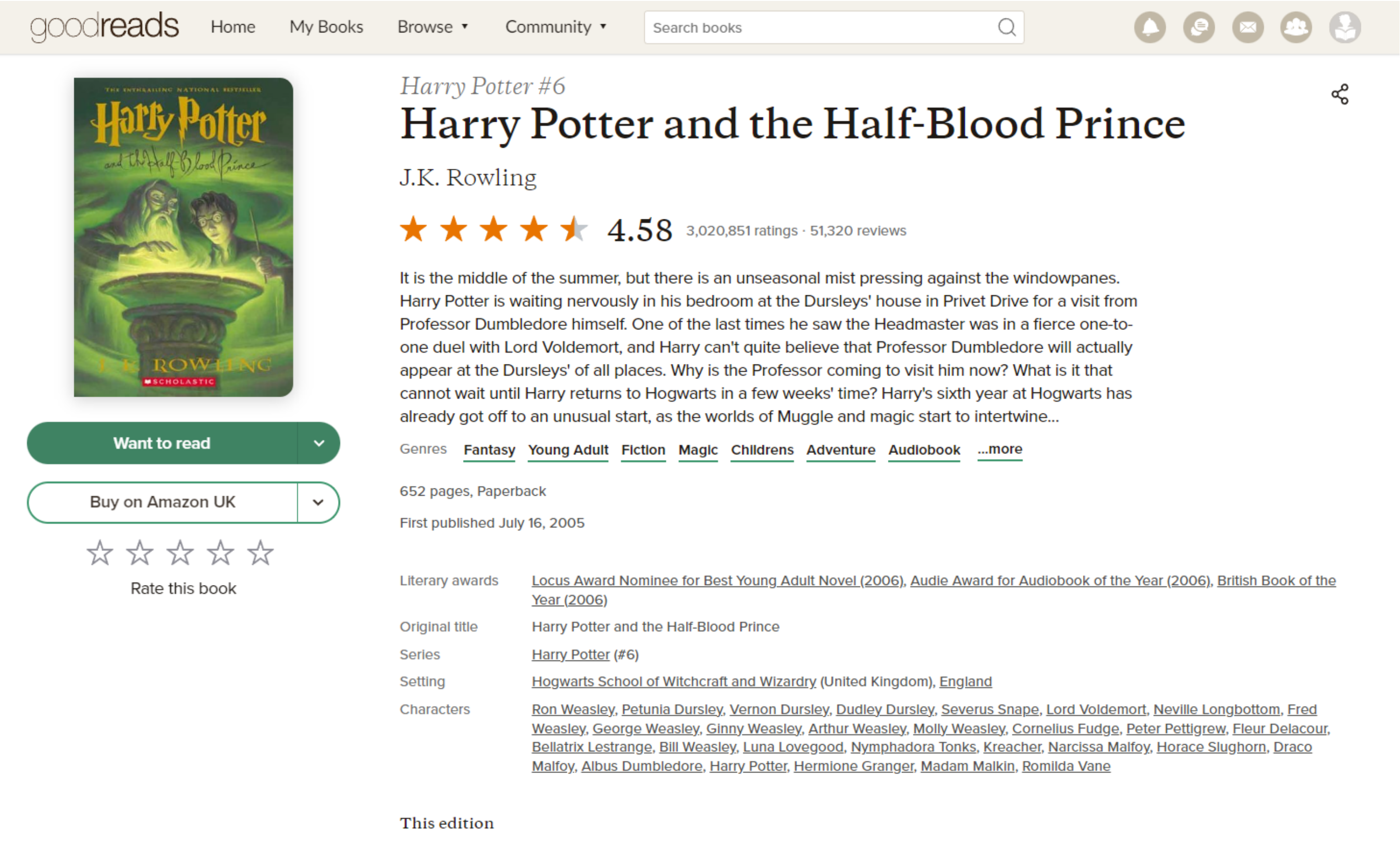
Look of the CSV file:



bookID	title	authors	average_rating	isbn	isbn13	langua	num_pages	ratings_count	text_reviews_count	publication_date	publisher
1	Harry Potter	J.K. Rowlin	4.57	4E+08	9.78E+12	eng	652	2095690	27591	9/16/2006	Scholastic Inc
2	Harry Potter	J.K. Rowlin	4.49	4E+08	9.78E+12	eng	870	2153167	29221	09-01-04	Scholastic Inc
4	Harry Potter	J.K. Rowlin	4.42	4E+08	9.78E+12	eng	352	6333	244	11-01-03	Scholastic
5	Harry Potter	J.K. Rowlin	4.56	0439655	9.78E+12	eng	435	2339585	36325	05-01-04	Scholastic Inc
8	Harry Potter	J.K. Rowlin	4.78	4E+08	9.78E+12	eng	2690	41428	164	9/13/2004	Scholastic
9	Unauthorize	W. Frederi	3.74	1E+09	9.78E+12	en-US	152	19	1	4/26/2005	Nimble Book
10	Harry Potter	J.K. Rowlin	4.73	4E+08	9.78E+12	eng	3342	28242	808	09-12-05	Scholastic
12	The Ultimate	Douglas Ac	4.38	5E+08	9.78E+12	eng	815	3628	254	11-01-05	Gramercy Bo
13	The Ultimate	Douglas Ac	4.38	3E+08	9.78E+12	eng	815	249558	4080	4/30/2002	Del Rey Book
14	The Hitchhik	Douglas Ac	4.22	1E+09	9.78E+12	eng	215	4930	460	08-03-04	Crown
16	The Hitchhik	Douglas Ac	4.22	7E+08	9.78E+12	eng	6	1266	253	3/23/2005	Random Hou
18	The Ultimate	Douglas Ac	4.38	5E+08	9.78E+12	eng	815	2877	195	1/17/1996	Wings Books
21	A Short Histc	Bill Bryson	4.21	0767908	9.78E+12	eng	544	248558	9396	9/14/2004	Broadway Bo
22	Bill Bryson's .	Bill Bryson	3.44	8E+08	9.78E+12	eng	55	7270	499	12-03-02	Broadway Bo
23	Bryson's Dict	Bill Bryson	3.87	8E+08	9.78E+12	eng	256	2088	131	9/14/2004	Broadway Bo
24	In a Sunburn	Bill Bryson	4.07	8E+08	9.78E+12	eng	335	72451	4245	5/15/2001	Broadway Bo
25	I'm a Strange	Bill Bryson	3.9	0767903	9.78E+12	eng	304	49240	2211	6/28/2000	Broadway Bo
26	The Lost Con	Bill Bryson	3.83	6E+07	9.78E+12	eng	299	45712	2257	8/28/1990	William Mor
27	Neither Here	Bill Bryson	3.86	4E+08	9.78E+12	eng	254	48701	2238	3/28/1993	William Mor
28	Notes from a	Bill Bryson	3.91	4E+08	9.78E+12	eng	324	80609	3301	5/28/1997	William Mor
29	The Mother T	Bill Bryson	3.93	4E+08	9.78E+12	eng	270	28489	2085	9/28/1991	William Mor
30	J.R.R. Tolkier	J.R.R. Tolki	4.59	3E+08	9.78E+12	eng	1728	101233	1550	9/25/2012	Ballantine Bo
31	The Lord of t	J.R.R. Tolki	4.5	6E+08	9.78E+12	eng	1184	1710	91	10/21/2004	Houghton Mi
34	The Fellowsh	J.R.R. Tolki	4.36	6E+08	9.78E+12	eng	398	2128944	13670	09-05-03	Houghton Mi
35	The Lord of t	J.R.R. Tolki	4.5	6E+08	9.78E+12	en-US	1216	1618	140	10-01-02	Houghton Mi
36	The Lord of t	Chris Smit	4.53	6E+08	9.78E+12	eng	218	19822	46	11-05-03	Houghton Mi
37	The Lord of t	Jude Fishe	4.5	6E+08	9.78E+12	eng	224	359	6	11/15/2004	Houghton Mi
45	Agile Web D	Dave Thon	3.84	0976694	9.78E+12	eng	558	1430	59	7/28/2005	Pragmatic Bo
50	Hatchet (Bria	Gary Pauls	3.72	7E+08	9.78E+12	eng	208	270244	12017	04-01-00	Atheneum B

As my target variable was the book genre, I required book descriptions to train my model. However, my dataset was incomplete, lacking both book descriptions and genres. I decided to use the dataset that I found on Kaggle about Goodreads books and collect the missing information from Goodreads myself. Then I started to think about the ways to collect the data.

Example web page of a book from Goodreads:



Harry Potter #6

Harry Potter and the Half-Blood Prince

J.K. Rowling

★★★★☆ 4.58 3,020,851 ratings · 51,320 reviews

It is the middle of the summer, but there is an unseasonal mist pressing against the windowpanes. Harry Potter is waiting nervously in his bedroom at the Dursleys' house in Privet Drive for a visit from Professor Dumbledore himself. One of the last times he saw the Headmaster was in a fierce one-to-one duel with Lord Voldemort, and Harry can't quite believe that Professor Dumbledore will actually appear at the Dursleys' of all places. Why is the Professor coming to visit him now? What is it that cannot wait until Harry returns to Hogwarts in a few weeks' time? Harry's sixth year at Hogwarts has already got off to an unusual start, as the worlds of Muggle and magic start to intertwine...

Genres Fantasy Young Adult Fiction Magic Childrens Adventure Audiobook ...more

652 pages, Paperback

First published July 16, 2005

Literary awards	Locus Award Nominee for Best Young Adult Novel (2006), Audie Award for Audiobook of the Year (2006), British Book of the Year (2006)
Original title	Harry Potter and the Half-Blood Prince
Series	Harry Potter (#6)
Setting	Hogwarts School of Witchcraft and Wizardry (United Kingdom), England
Characters	Ron Weasley, Petunia Dursley, Vernon Dursley, Dudley Dursley, Severus Snape, Lord Voldemort, Neville Longbottom, Fred Weasley, George Weasley, Ginny Weasley, Arthur Weasley, Molly Weasley, Cornelius Fudge, Peter Pettigrew, Fleur Delacour, Bellatrix Lestrange, Bill Weasley, Luna Lovegood, Nymphadora Tonks, Kreacher, Narcissa Malfoy, Horace Slughorn, Draco Malfoy, Albus Dumbledore, Harry Potter, Hermione Granger, Madam Malkin, Romilda Vane

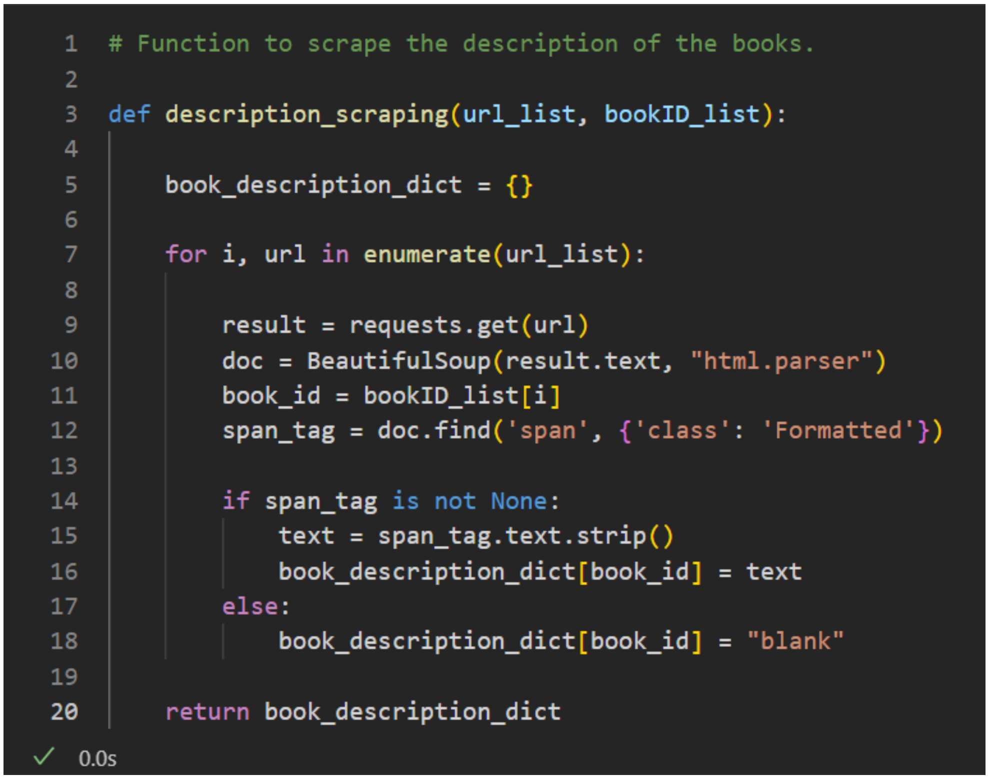
This edition

(source: https://www.goodreads.com/book/show/1)

Data collection

The data previously presented was obtained by downloading it from Kaggle.

Initially, I intended to use the Goodreads API to gather the book descriptions and genres for the books within my dataset. However, due to Goodreads no longer providing new developer keys for their public developer API as of December 8th, 2020, I resorted to web scraping. I utilized the Requests and BeautifulSoup Python libraries to scrape the additional data. Below is the Python function that I wrote to scrape the book descriptions:



```python
# Function to scrape the description of the books.

def description_scraping(url_list, bookID_list):

    book_description_dict = {}

    for i, url in enumerate(url_list):

        result = requests.get(url)
        doc = BeautifulSoup(result.text, "html.parser")
        book_id = bookID_list[i]
        span_tag = doc.find('span', {'class': 'Formatted'})

        if span_tag is not None:
            text = span_tag.text.strip()
            book_description_dict[book_id] = text
        else:
            book_description_dict[book_id] = "blank"

    return book_description_dict
```

✓ 0.0s

Since I had the book id's of the books in the dataset I could create the url's of the books by adding the "goodreads.com/book/show/" prefix. This function takes two lists, url and bookID lists. Then loops on the url list and retrieves the HTML of each url with the requests library. It then uses the BeautifulSoup library to parse the HTML and extract the book description. If it exists, the text content of the tag is extracted and stored in the "book_description_dict" with the corresponding book ID. If the tag does not exist, the book ID is still stored in the dictionary with a value of "blank". Finally, the function returns the "book_description_dict" with all the book descriptions and their corresponding IDs.

Scraping the book descriptions was a simpler task compared to scraping the genres in my project, as the HTML for the descriptions was relatively more consistent.

However, the genre scraping presented several challenges, including validating the accuracy of the scraped data, encountering errors due to inconsistencies in the HTML pages of the books, and risking getting blocked by the website if the scraping was not performed in batches. To overcome these obstacles, I divided the genres scraping into multiple steps, used delay in between each request with “time.sleep” and VPN to avoid being blocked by the website. Initially, I planned to scrape three genres per book for later analysis but eventually decided that the first genre of each book was sufficient for my target variable. In the end, I successfully scraped the genre of each book of my dataset from the website.

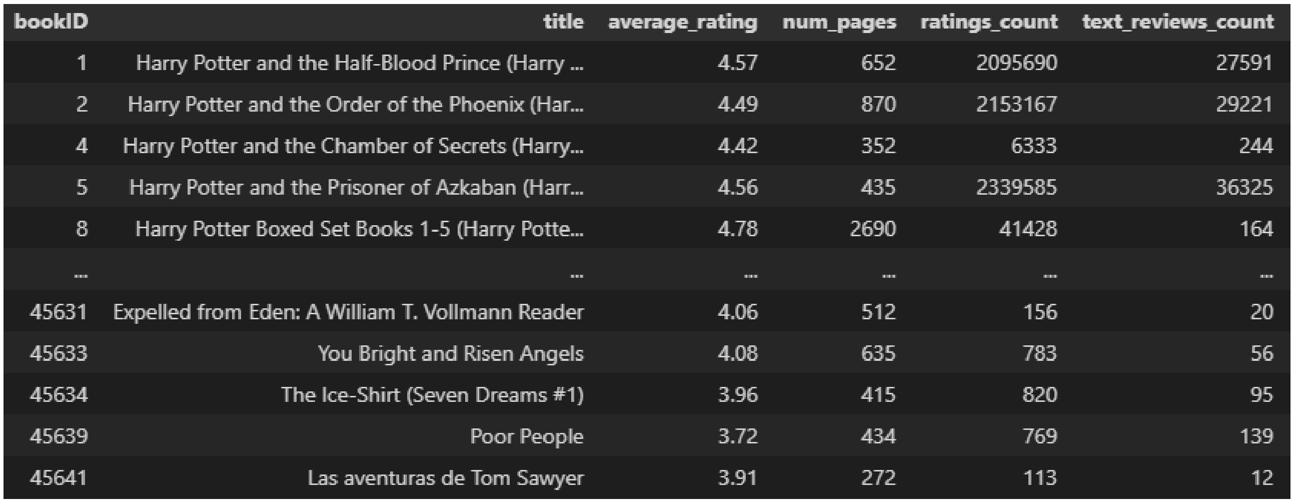
You can locate the function for scraping book genres in the Python file named "books_web_scraping.ipynb", under the name "extract_book_genres()". This function resembles the "description_scraping()" function but utilizes different HTML tags to retrieve the genres. Additionally, it incorporates error handling and sleep time to avoid getting errors.

Data cleaning and Exploratory data analysis

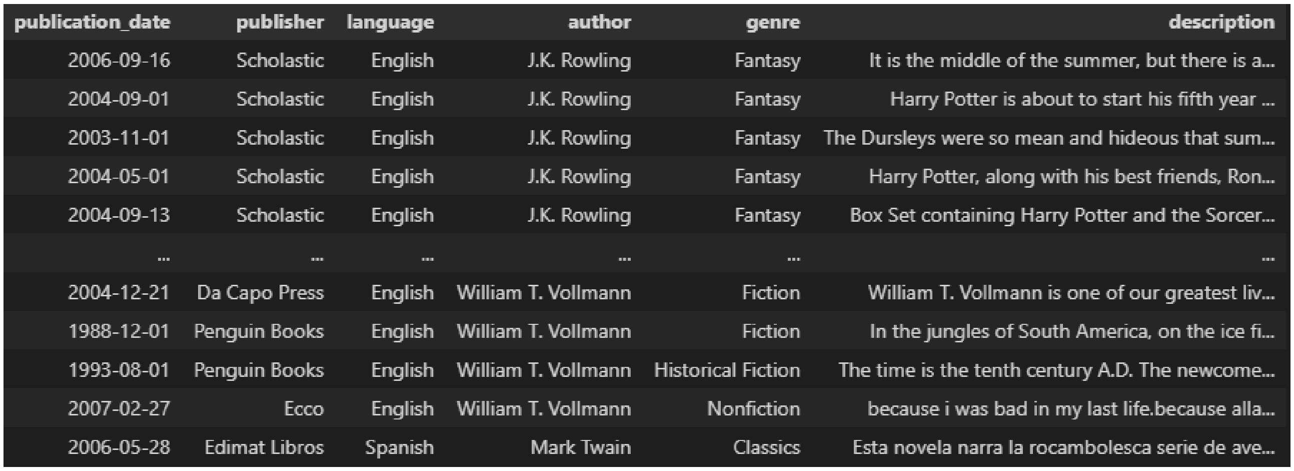
Firstly I investigated the dataset for wrong data types, NaN values, duplicates, corrupted rows, and whitespaces before and after the titles, as well as wrong dates, and fixed them. These steps can be found in my main Python file “books_project.ipynb”. I also made sure that the dates were correct and converted the "language_code" column values to language names to improve clarity. Additionally, I cleaned up the author names to only include the primary author and removed any narrator or translator names. To ensure that the data was accurate, I validated it and corrected any errors, such as future dates or unknown encoding characters. The "publishers" column had some issues with incorrect names, so I used a string matching function from the "fuzzywuzzy" library to search for similar names and correct them.

Fortunately, I could keep most of my data and ended up removing only a few rows after the cleaning. Then, I added the book genres and descriptions that I scraped in another file to my dataframe. I used the “pickle” module and pickle files to export/import and save my dictionaries that I scraped for reliability and speed even though my files were not excessively large. This method not only helped to optimize my current workflow but also futureproofed my project, making it easier to scale up if the dataset grows in the future.

The dataframe after cleaning in two images:



bookID	title	average_rating	num_pages	ratings_count	text_reviews_count
1	Harry Potter and the Half-Blood Prince (Harry ...	4.57	652	2095690	27591
2	Harry Potter and the Order of the Phoenix (Har...	4.49	870	2153167	29221
4	Harry Potter and the Chamber of Secrets (Harry...	4.42	352	6333	244
5	Harry Potter and the Prisoner of Azkaban (Harr...	4.56	435	2339585	36325
8	Harry Potter Boxed Set Books 1-5 (Harry Potte...	4.78	2690	41428	164
...	...	...	...	...	...
45631	Expelled from Eden: A William T. Vollmann Reader	4.06	512	156	20
45633	You Bright and Risen Angels	4.08	635	783	56
45634	The Ice-Shirt (Seven Dreams #1)	3.96	415	820	95
45639	Poor People	3.72	434	769	139
45641	Las aventuras de Tom Sawyer	3.91	272	113	12



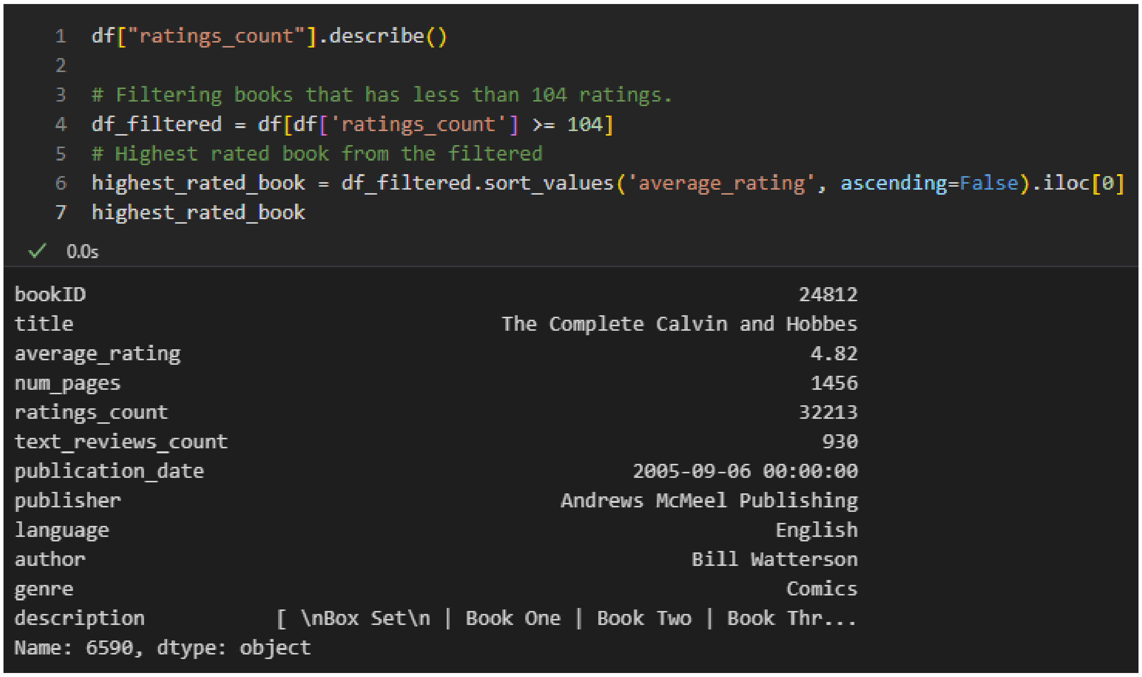
publication_date	publisher	language	author	genre	description
2006-09-16	Scholastic	English	J.K. Rowling	Fantasy	It is the middle of the summer, but there is a...
2004-09-01	Scholastic	English	J.K. Rowling	Fantasy	Harry Potter is about to start his fifth year ...
2003-11-01	Scholastic	English	J.K. Rowling	Fantasy	The Dursleys were so mean and hideous that sum...
2004-05-01	Scholastic	English	J.K. Rowling	Fantasy	Harry Potter, along with his best friends, Ron...
2004-09-13	Scholastic	English	J.K. Rowling	Fantasy	Box Set containing Harry Potter and the Sorcer...
...	...	...	...	...	...
2004-12-21	Da Capo Press	English	William T. Vollmann	Fiction	William T. Vollmann is one of our greatest liv...
1988-12-01	Penguin Books	English	William T. Vollmann	Fiction	In the jungles of South America, on the ice fi...
1993-08-01	Penguin Books	English	William T. Vollmann	Historical Fiction	The time is the tenth century A.D. The newcome...
2007-02-27	Ecco	English	William T. Vollmann	Nonfiction	because i was bad in my last life.because alla...
2006-05-28	Edimat Libros	Spanish	Mark Twain	Classics	Esta novela narra la rocambolesca serie de ave...

After data cleaning, I started with my explanatory data analysis.

Highest-rated book

First, I checked for the book that had the highest rating after filtering out the books with less than 104 ratings. 104 number was chosen because %25 of the books had less than 104 ratings in the data set. The highest-rated book is called “The Complete Calvin and Hobbes”, with an average rating of 4.82 out of 5 as of this dataset was collected. Turns out it is an american daily comic strip.

The code:



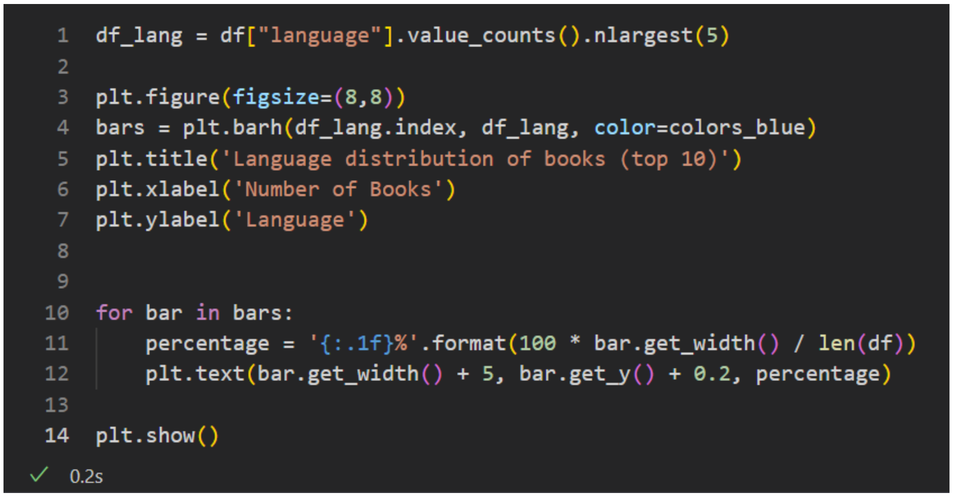
```python
df["ratings_count"].describe()

# Filtering books that has less than 104 ratings.
df_filtered = df[df['ratings_count'] >= 104]
# Highest rated book from the filtered
highest_rated_book = df_filtered.sort_values('average_rating', ascending=False).iloc[0]
highest_rated_book
```

```
✓ 0.0s
```

```
bookID                                                     24812
title                                 The Complete Calvin and Hobbes
average_rating                                              4.82
num_pages                                                   1456
ratings_count                                              32213
text_reviews_count                                           930
publication_date                                 2005-09-06 00:00:00
publisher                                Andrews McMeel Publishing
language                                                 English
author                                              Bill Watterson
genre                                                     Comics
description            [ \nBox Set\n | Book One | Book Two | Book Thr...
Name: 6590, dtype: object
```

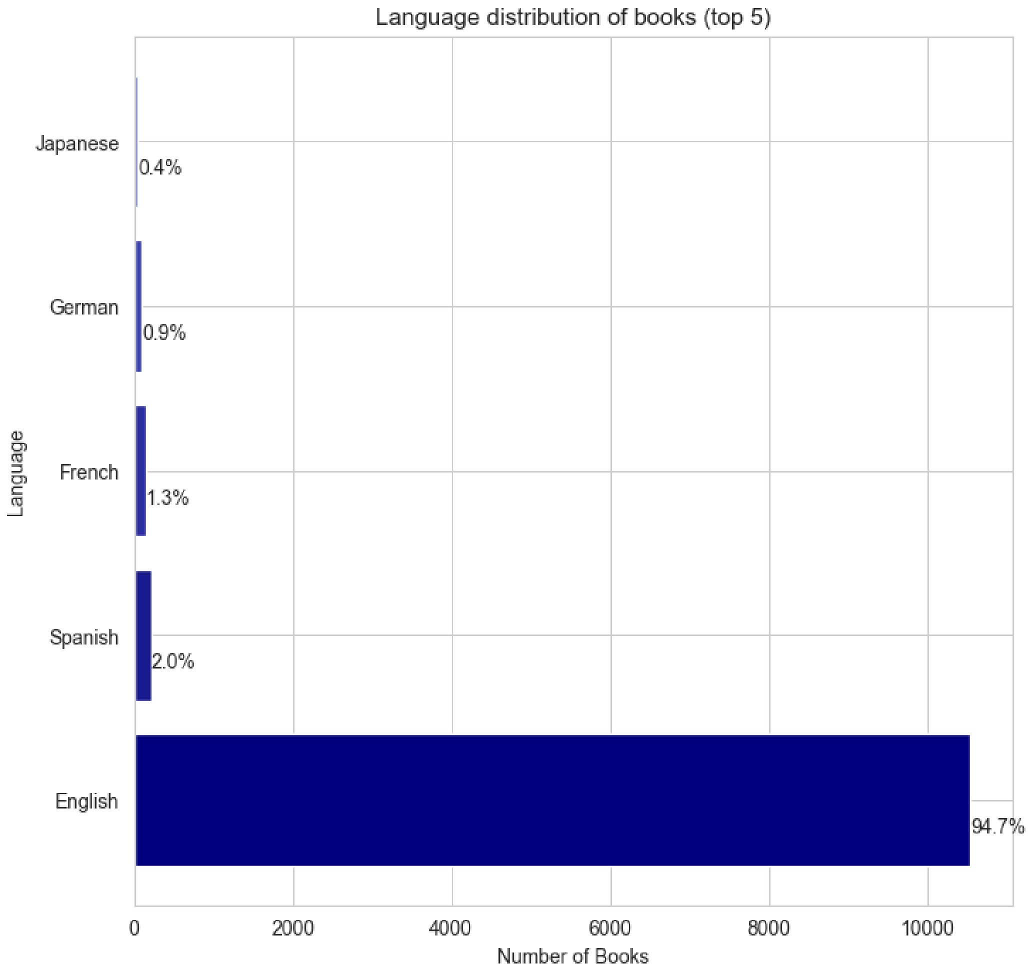
Language distribution of books

We can see that %94.7 of the books in my dataset are in the English language. This aligns with the top 5 most popular languages on Goodreads, as reported in September 2021, which are English, Spanish, French, German, and Italian. This correlation is also reflected in the small sample dataset that I possess, as shown in the following chart.



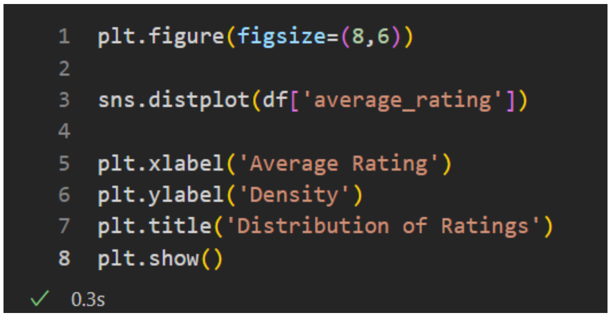
```python
df_lang = df["language"].value_counts().nlargest(5)

plt.figure(figsize=(8,8))
bars = plt.barh(df_lang.index, df_lang, color=colors_blue)
plt.title('Language distribution of books (top 10)')
plt.xlabel('Number of Books')
plt.ylabel('Language')


for bar in bars:
    percentage = '{:.1f}%'.format(100 * bar.get_width() / len(df))
    plt.text(bar.get_width() + 5, bar.get_y() + 0.2, percentage)

plt.show()
```

✓ 0.2s

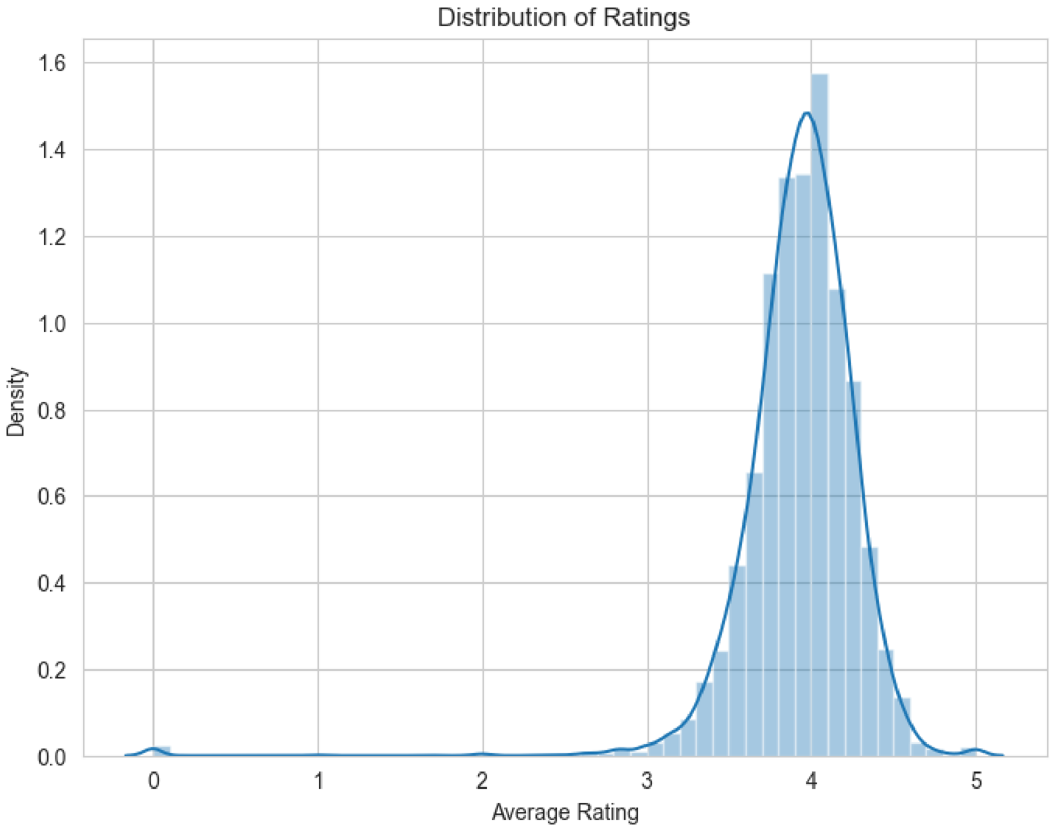


Distribution of ratings

For the average rating of books, I plotted the data on a distribution plot which shows a minimal left-skewed chart with most of the ratings being between 3 and 5 stars, and the average seems to be around 4 stars.

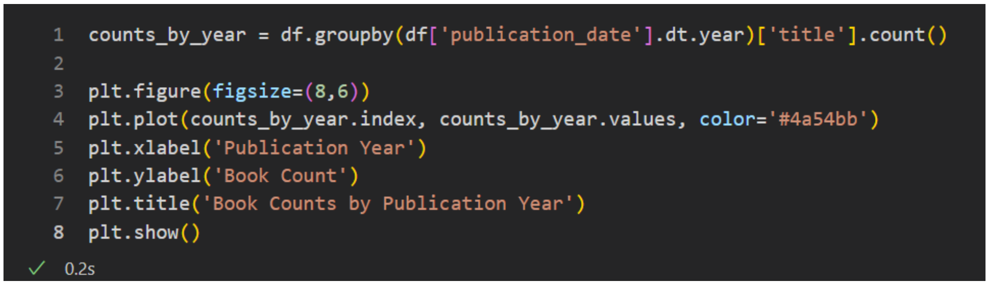


```
plt.figure(figsize=(8,6))

sns.distplot(df['average_rating'])

plt.xlabel('Average Rating')
plt.ylabel('Density')
plt.title('Distribution of Ratings')
plt.show()
```



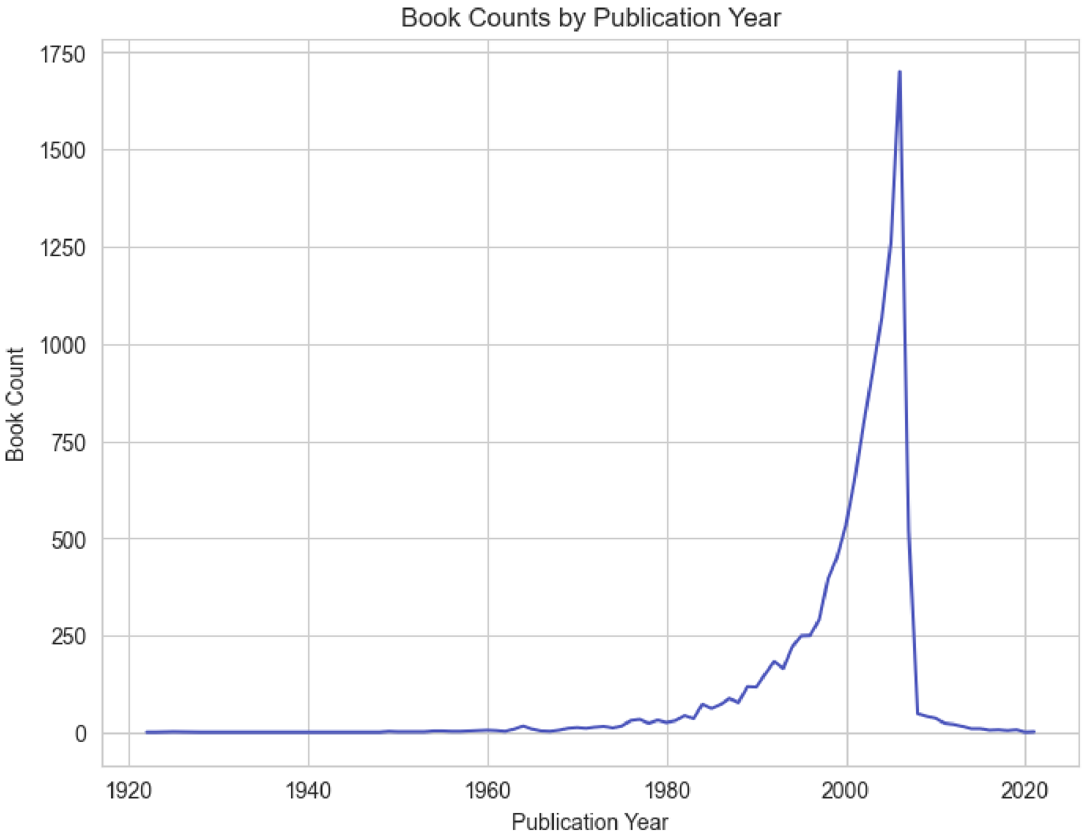
Book Counts by Publication Year

When we count the book counts by publication year, we can see that there are more books in the dataset between the years 2000 and 2010. The chart shows a left-skewed appearance showing fewer books before 1980 and after 2010 in the dataset.



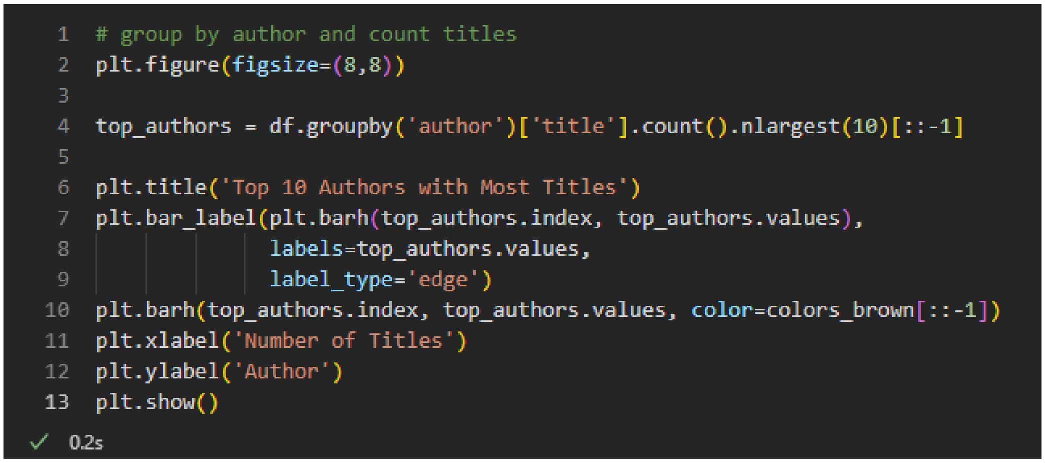
```
counts_by_year = df.groupby(df['publication_date'].dt.year)['title'].count()

plt.figure(figsize=(8,6))
plt.plot(counts_by_year.index, counts_by_year.values, color='#4a54bb')
plt.xlabel('Publication Year')
plt.ylabel('Book Count')
plt.title('Book Counts by Publication Year')
plt.show()
```

✓ 0.2s

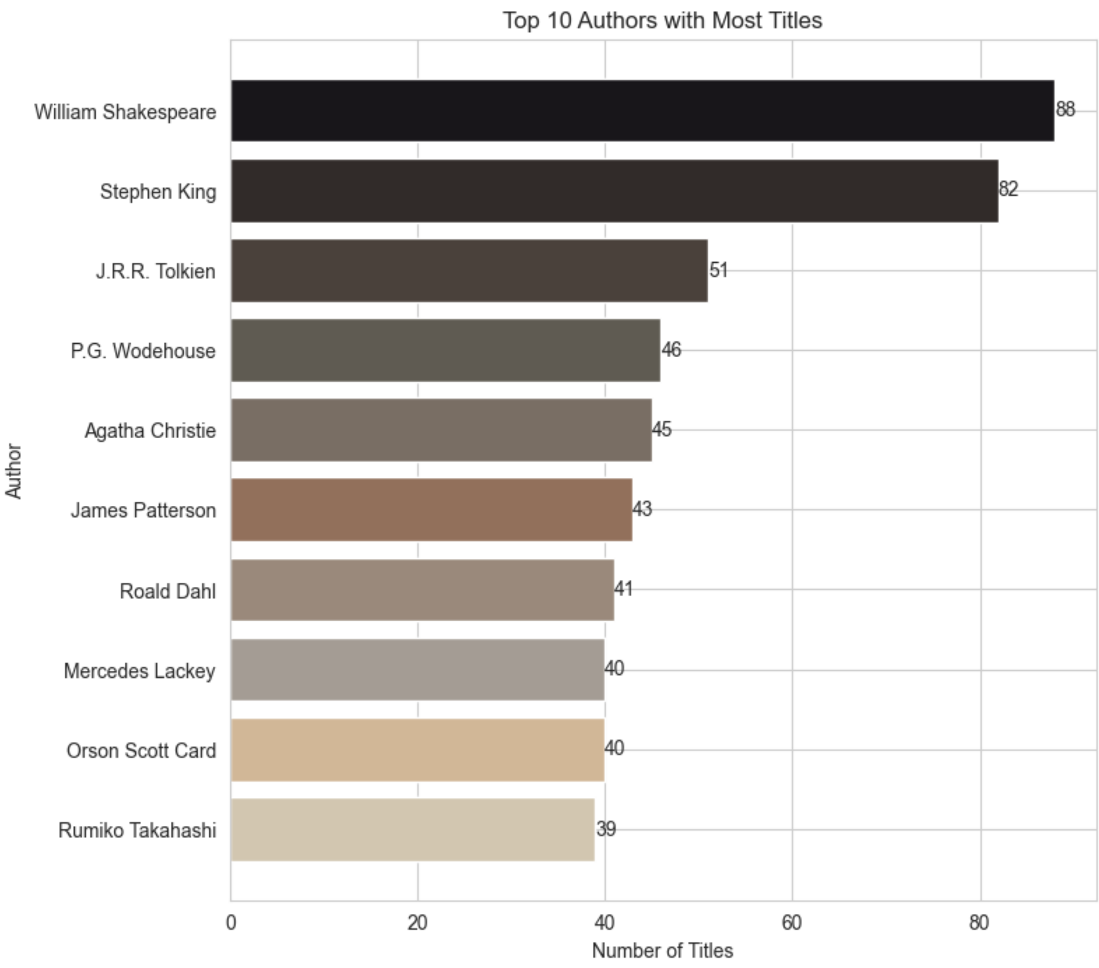


Top 10 Authors with most titles

Below is a chart showing the top 10 authors with the most titles. William Shakespeare is at the top with 88 works in this dataset. However, upon searching on the internet, it appears that William Shakespeare has about 37 plays in total. This suggests that he either may have written works in other genres, or there may be different editions of the same book published by different publishers since he is a widely renowned author. Indeed there is the same book from different publishers in the dataset.

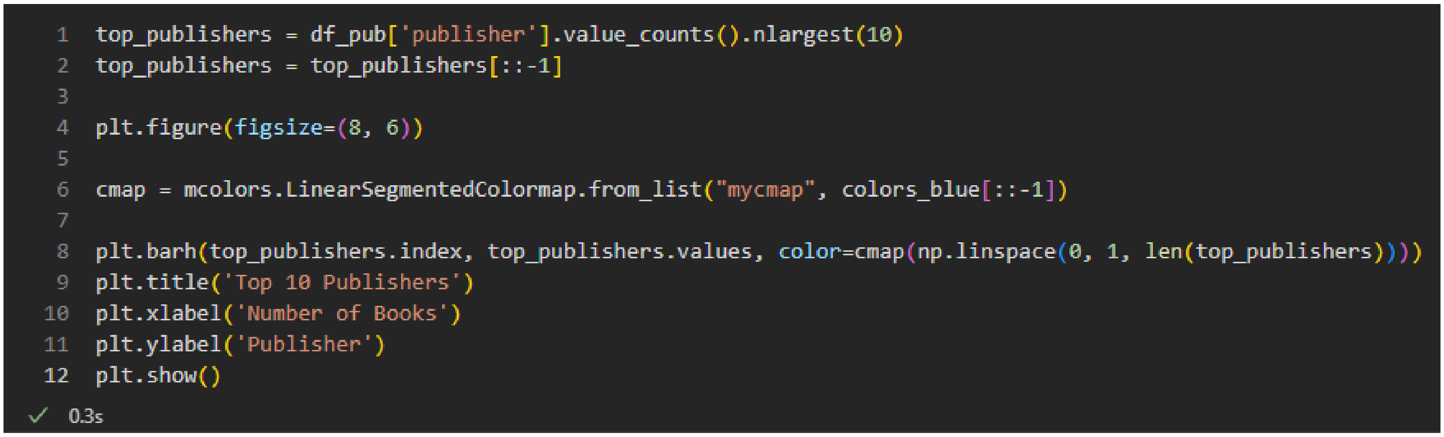


```python
# group by author and count titles
plt.figure(figsize=(8,8))

top_authors = df.groupby('author')['title'].count().nlargest(10)[::-1]

plt.title('Top 10 Authors with Most Titles')
plt.bar_label(plt.barh(top_authors.index, top_authors.values),
              labels=top_authors.values,
              label_type='edge')
plt.barh(top_authors.index, top_authors.values, color=colors_brown[::-1])
plt.xlabel('Number of Titles')
plt.ylabel('Author')
plt.show()
```

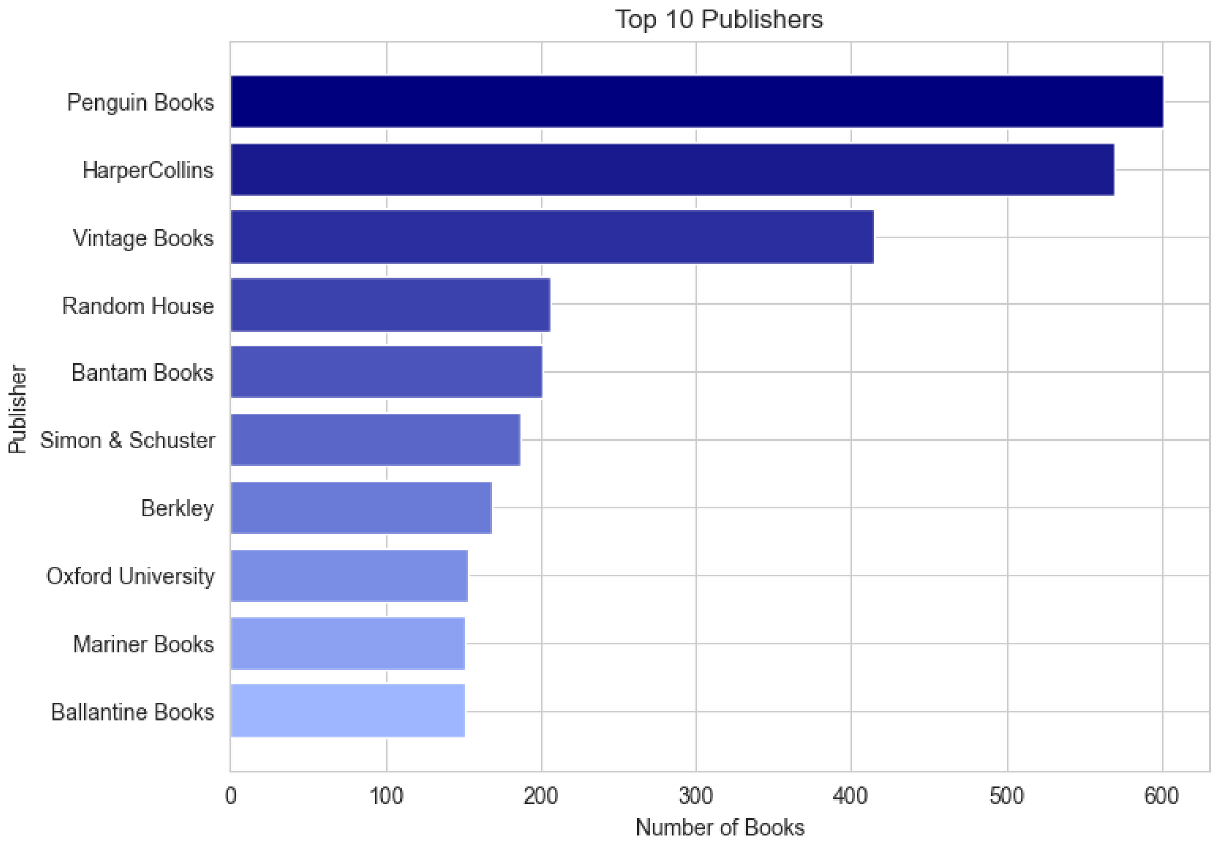


Top 10 Publishers

Among publishers with a significant book count, Penguin Books, HarperCollins, and Vintage Books lead the way. However, as the dataset lacks sales information, we cannot gain any insights into related metrics such as sales performance.

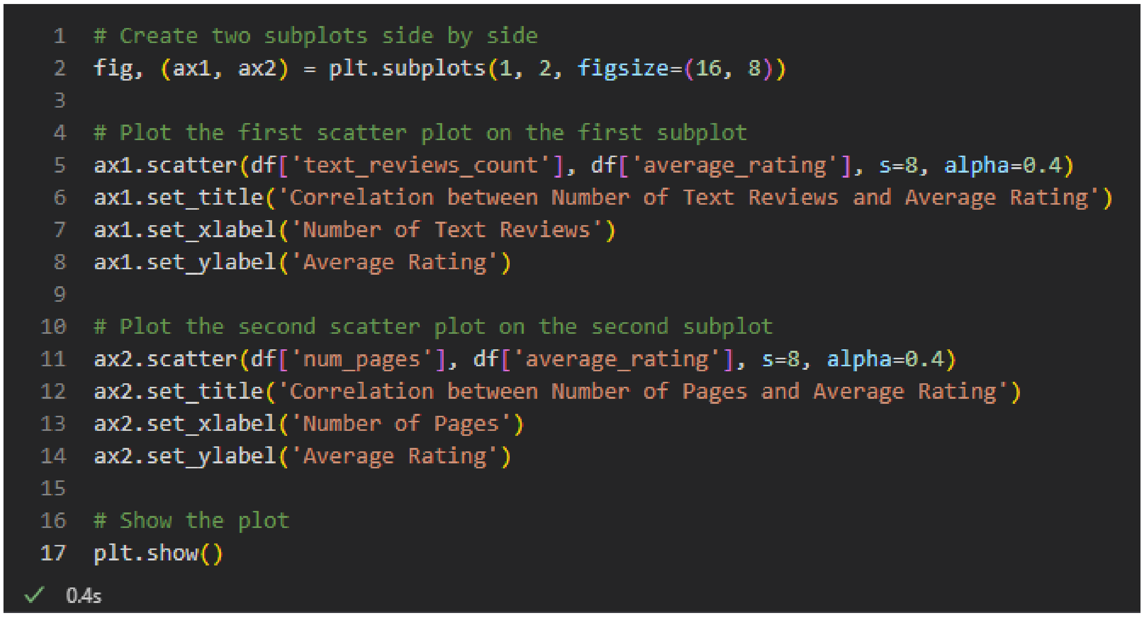


```python
top_publishers = df_pub['publisher'].value_counts().nlargest(10)
top_publishers = top_publishers[::-1]

plt.figure(figsize=(8, 6))

cmap = mcolors.LinearSegmentedColormap.from_list("mycmap", colors_blue[::-1])

plt.barh(top_publishers.index, top_publishers.values, color=cmap(np.linspace(0, 1, len(top_publishers))))
plt.title('Top 10 Publishers')
plt.xlabel('Number of Books')
plt.ylabel('Publisher')
plt.show()
```

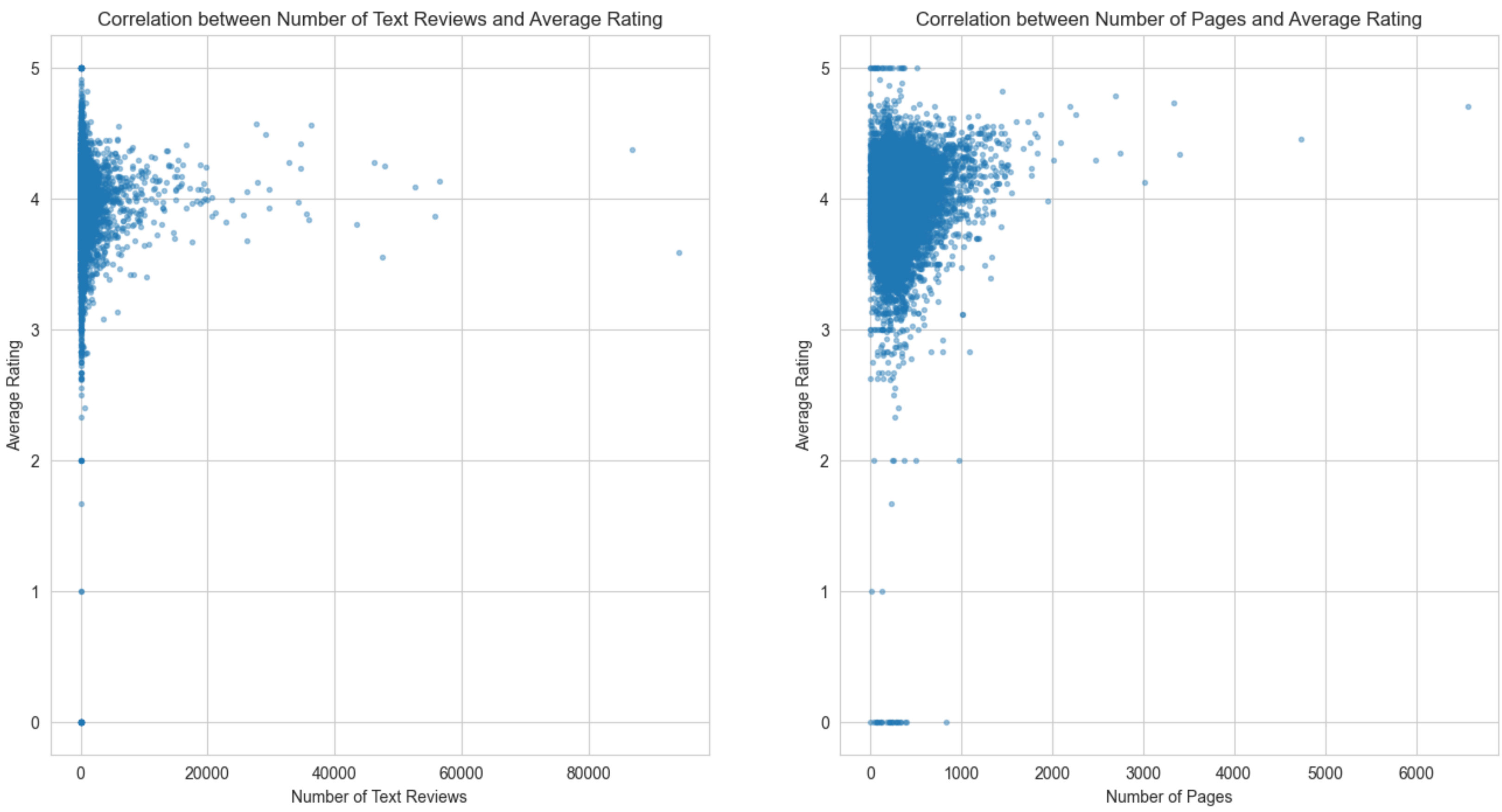


Correlation between Number of Text Reviews and Number of Pages vs Average Rating

In the left scatter plot, there appears to be no discernible correlation between the number of text reviews and the average rating, and it is notable that the majority of books have fewer than 5000 text reviews. In contrast, when examining the relationship between the number of pages and the average rating, it is evident that books with more than 2000 pages receive consistently high ratings above 4. Upon closer inspection of the dataset, it becomes apparent that books with over 2000 pages are often collections of popular book series which explains the high rating amongst them.

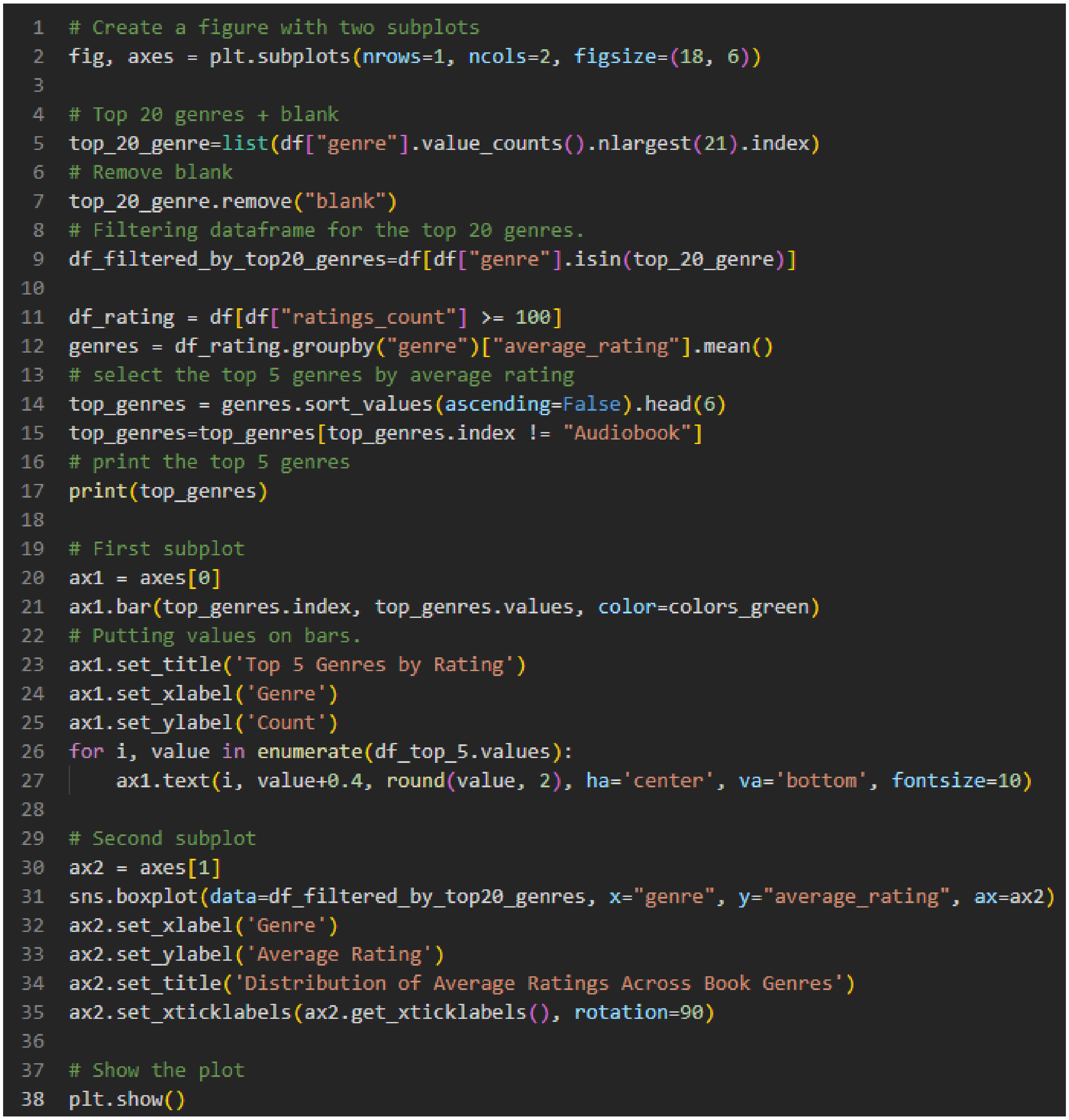


```python
# Create two subplots side by side
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(16, 8))

# Plot the first scatter plot on the first subplot
ax1.scatter(df['text_reviews_count'], df['average_rating'], s=8, alpha=0.4)
ax1.set_title('Correlation between Number of Text Reviews and Average Rating')
ax1.set_xlabel('Number of Text Reviews')
ax1.set_ylabel('Average Rating')

# Plot the second scatter plot on the second subplot
ax2.scatter(df['num_pages'], df['average_rating'], s=8, alpha=0.4)
ax2.set_title('Correlation between Number of Pages and Average Rating')
ax2.set_xlabel('Number of Pages')
ax2.set_ylabel('Average Rating')

# Show the plot
plt.show()
```

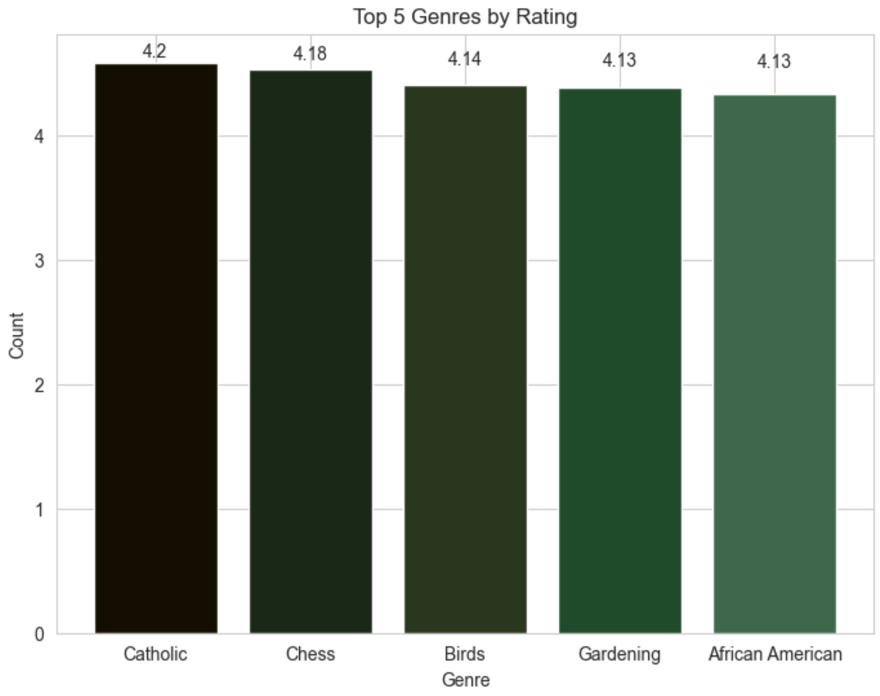


Genre Analysis, Top 5 Genres by Average Rating and Distribution of Average Ratings

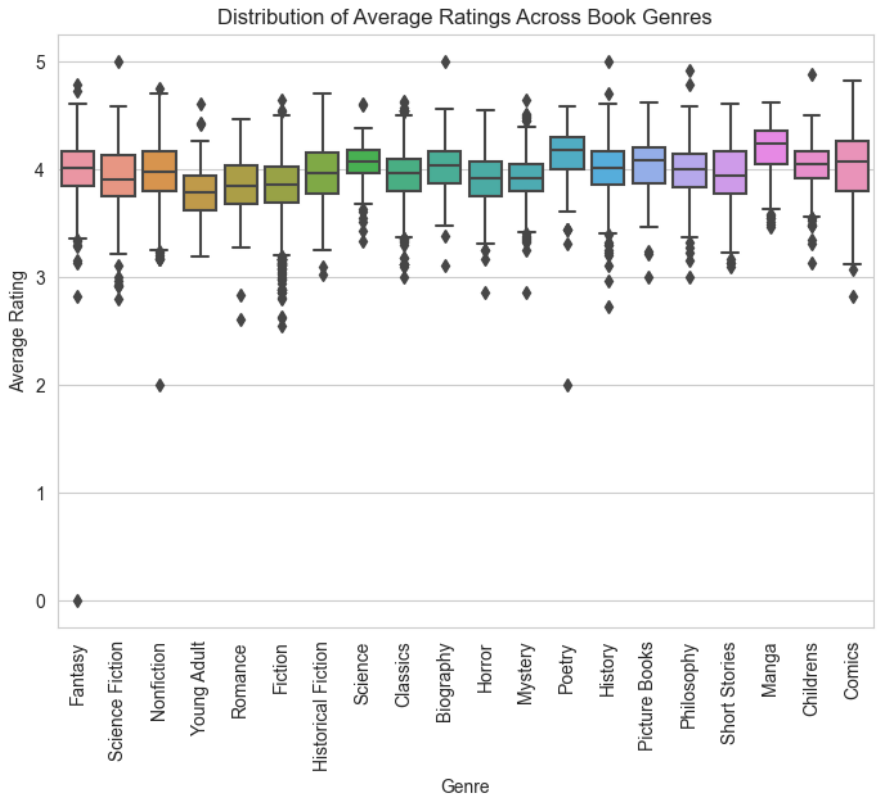
Top 5 genres are plotted after filtering out the books that has less than 100 ratings. We can see niche genres such as Catholic, Chess, Birds has the highest average rating in this dataset.



```python
# Create a figure with two subplots
fig, axes = plt.subplots(nrows=1, ncols=2, figsize=(18, 6))

# Top 20 genres + blank
top_20_genre=list(df["genre"].value_counts().nlargest(21).index)
# Remove blank
top_20_genre.remove("blank")
# Filtering dataframe for the top 20 genres.
df_filtered_by_top20_genres=df[df["genre"].isin(top_20_genre)]

df_rating = df[df["ratings_count"] >= 100]
genres = df_rating.groupby("genre")["average_rating"].mean()
# select the top 5 genres by average rating
top_genres = genres.sort_values(ascending=False).head(6)
top_genres=top_genres[top_genres.index != "Audiobook"]
# print the top 5 genres
print(top_genres)

# First subplot
ax1 = axes[0]
ax1.bar(top_genres.index, top_genres.values, color=colors_green)
# Putting values on bars.
ax1.set_title('Top 5 Genres by Rating')
ax1.set_xlabel('Genre')
ax1.set_ylabel('Count')
for i, value in enumerate(df_top_5.values):
    ax1.text(i, value+0.4, round(value, 2), ha='center', va='bottom', fontsize=10)

# Second subplot
ax2 = axes[1]
sns.boxplot(data=df_filtered_by_top20_genres, x="genre", y="average_rating", ax=ax2)
ax2.set_xlabel('Genre')
ax2.set_ylabel('Average Rating')
ax2.set_title('Distribution of Average Ratings Across Book Genres')
ax2.set_xticklabels(ax2.get_xticklabels(), rotation=90)

# Show the plot
plt.show()
```



The distribution seems to be similar for the top 20 genres that has the most books but we can still spot out niche genres such as Poetry having the interquartile range (the body) of the boxplot higher than 4 stars.



SQL or NoSQL

SQL and NoSQL are two different types of database management systems with their own set of characteristics and advantages. Here are some major differences between the two:

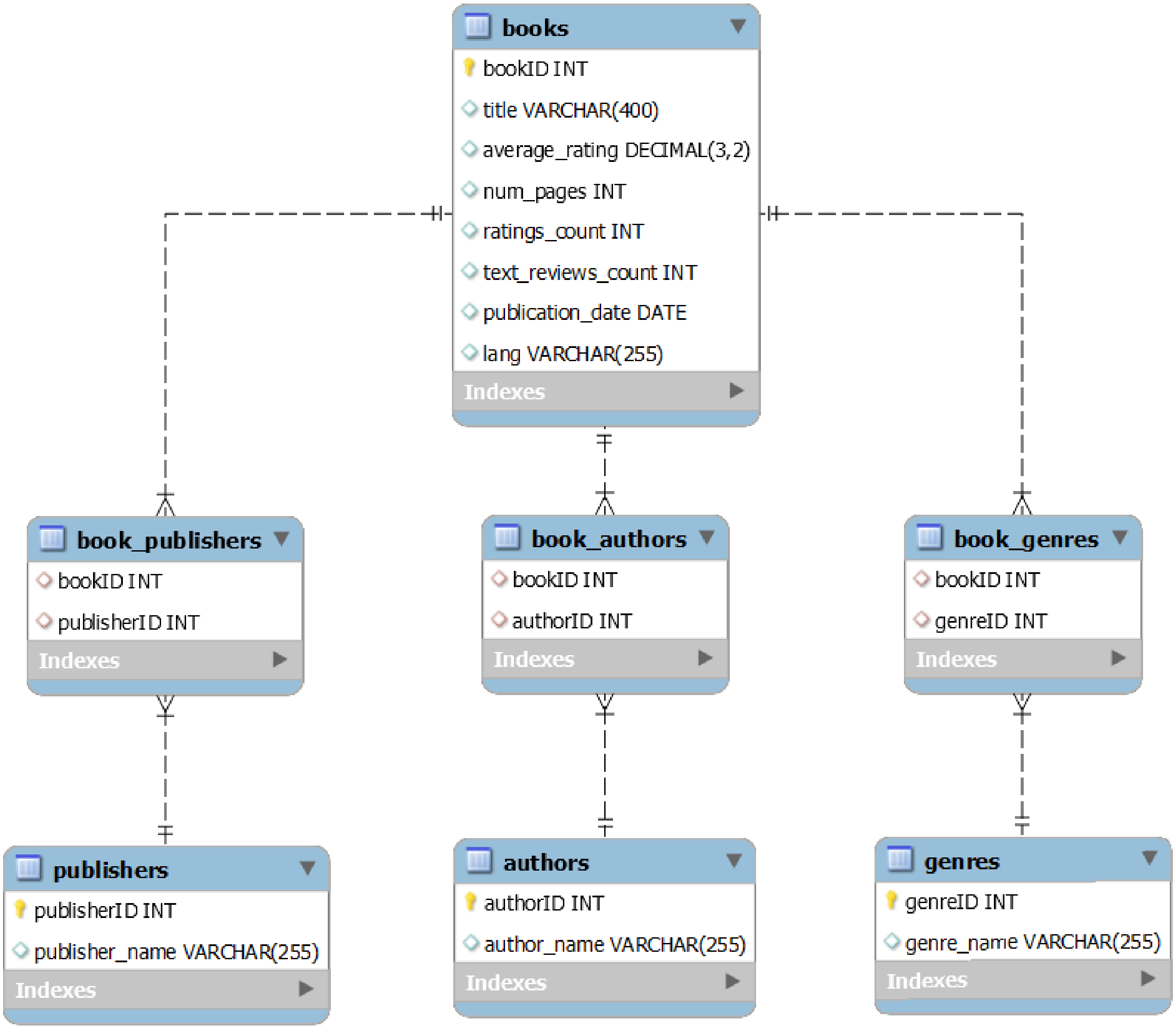
1. Data Model: SQL databases use a tabular data model where data is stored in tables consisting of rows and columns. NoSQL databases, on the other hand, use various data models such as document-oriented, key-value pairs, graph, and column-family.
2. Schema: SQL databases have a rigid schema that needs to be defined before data can be stored. In contrast, NoSQL databases have a flexible schema that allows data to be stored without a predefined structure.
3. Scalability: SQL databases are vertically scalable, which means they can handle an increasing amount of data by increasing the resources of a single server. NoSQL databases are horizontally scalable, which means they can handle an increasing amount of data by adding more servers to a distributed system.
4. Transactions: SQL databases have built-in support for transactions that ensure data consistency and integrity. NoSQL databases may or may not support transactions, depending on the specific database technology used.
5. Querying: SQL databases support complex querying using SQL (Structured Query Language). NoSQL databases have their own query languages that are often simpler and more focused on specific data models.
6. Cost: SQL databases tend to be more expensive due to their need for specialized hardware and software. NoSQL databases are often more affordable because they can run on commodity hardware and open-source software.

It seems appropriate since I have structured tables with a predefined schema to use SQL. Furthermore, if I expect to add more data or other sources, SQL would be optimal due to vertical scalability in my case.

ERD

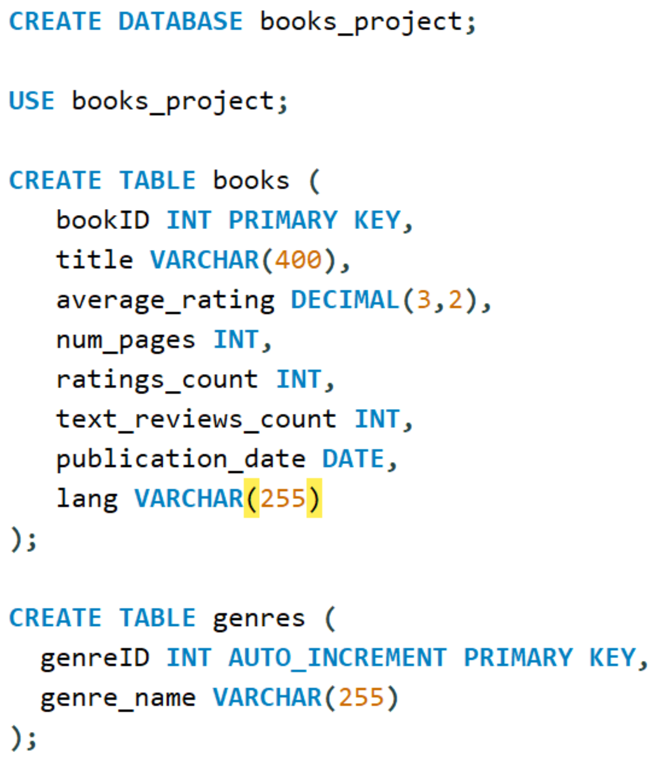
The entity relationship diagram for my database can be seen below:

Many to many relationships have been broken with intermediary tables.



MySQL Queries

Firstly, I created the database and the tables according to the entity relationship diagram. Examples of table creations:



```
CREATE DATABASE books_project;

USE books_project;

CREATE TABLE books (
   bookID INT PRIMARY KEY,
   title VARCHAR(400),
   average_rating DECIMAL(3,2),
   num_pages INT,
   ratings_count INT,
   text_reviews_count INT,
   publication_date DATE,
   lang VARCHAR(255)
);

CREATE TABLE genres (
  genreID INT AUTO_INCREMENT PRIMARY KEY,
  genre_name VARCHAR(255)
);
```

Once the data has been cleaned, we can proceed with inserting the tables into MySQL using SQL Alchemy. SQL Alchemy is a popular Python library used for SQL database management. It provides a set of high-level API for communicating with SQL databases, allowing us to abstract away the low-level details of SQL queries and database connections.

By using SQL Alchemy, we can easily and efficiently interact with MySQL databases from within our Python code, simplifying the process of inserting our cleaned data into the database.

Query 1) Top 10 books with the most pages

```
SELECT bookID, title, num_pages FROM books
ORDER BY num_pages DESC
LIMIT 5;
```

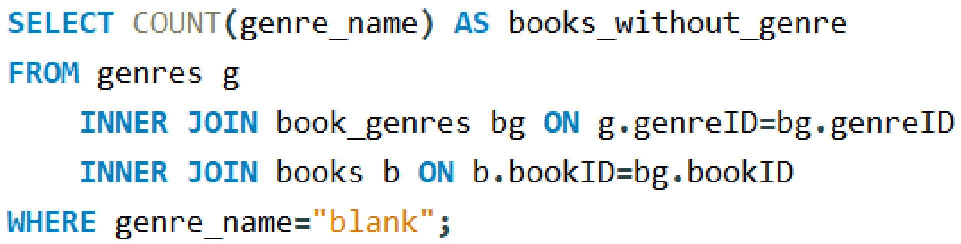
bookID	title	pages
24520	The Complete Aubrey/Maturin Novels (5 Volumes)	6576
25587	The Second World War	4736
44613	Remembrance of Things Past (Boxed Set)	3400
10	Harry Potter Collection (Harry Potter #1-6)	3342
25709	Summa Theologica 5 Vols	3020

Query 2) Who wrote the most books?

```
SELECT COUNT(title) AS book_count, author_name
FROM books b
    INNER JOIN book_authors ba ON b.bookID=ba.bookID
    INNER JOIN authors a ON ba.authorID=a.authorID
GROUP BY a.author_name
ORDER BY book_count DESC;
```

book_count	author_name
88	William Shakespeare
82	Stephen King
51	J.R.R. Tolkien
46	P.G. Wodehouse
45	Agatha Christie

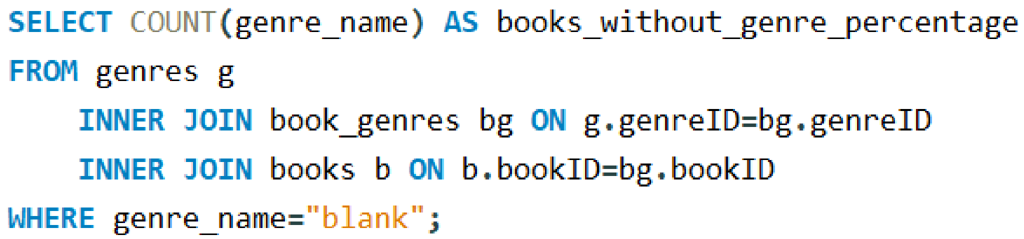
Query 3) How many books are there without a genre?



```
SELECT COUNT(genre_name) AS books_without_genre
FROM genres g
    INNER JOIN book_genres bg ON g.genreID=bg.genreID
    INNER JOIN books b ON b.bookID=bg.bookID
WHERE genre_name="blank";
```

	books_without_genre
>	374

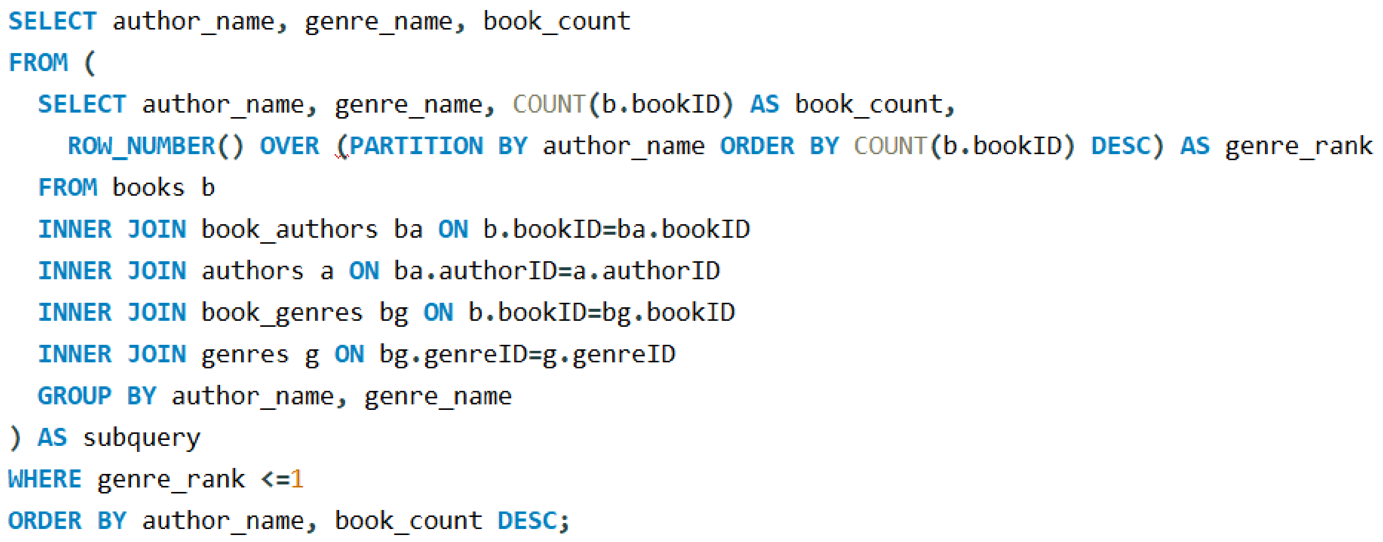
Query 4) Percentage of books without genre in the dataset?



```
SELECT COUNT(genre_name) AS books_without_genre_percentage
FROM genres g
    INNER JOIN book_genres bg ON g.genreID=bg.genreID
    INNER JOIN books b ON b.bookID=bg.bookID
WHERE genre_name="blank";
```

	books_without_genre_percentage
>	3.36

Query 5) Favorite genre of authors (random section taken from the query result)



```
SELECT author_name, genre_name, book_count
FROM (
  SELECT author_name, genre_name, COUNT(b.bookID) AS book_count,
    ROW_NUMBER() OVER (PARTITION BY author_name ORDER BY COUNT(b.bookID) DESC) AS genre_rank
  FROM books b
  INNER JOIN book_authors ba ON b.bookID=ba.bookID
  INNER JOIN authors a ON ba.authorID=a.authorID
  INNER JOIN book_genres bg ON b.bookID=bg.bookID
  INNER JOIN genres g ON bg.genreID=g.genreID
  GROUP BY author_name, genre_name
) AS subquery
WHERE genre_rank <=1
ORDER BY author_name, book_count DESC;
```

author_name	genre_name	book_count
Adam Rex	Classics	1
Adam Smith	Economics	2
Adam Swift	Philosophy	2
Adam Woog	History	1
Adolfo Bioy Casares	Poetry	1
Adrian Guelke	Grad School	1
Adrian McKinty	Comics	2
Aeschylus	Classics	14
Aesop	Horror	1
Agatha Christie	Mystery	44
Ai Morinaga	Manga	1
Adam Rex	Classics	1

Conclusion

After conducting the previous EDA, we can derive some key findings. Firstly, the majority of books in the dataset are published in English. Secondly, niche genres generally have a higher rating than others, but most genres still have an average rating of around 4 stars. Thirdly, most books are less than 500 pages, and there is no clear correlation between the number of pages and the average rating.

Regarding the queries, we can further investigate each author's favorite genre by counting the number of books they have written in each genre. As expected, most authors tend to specialize in a particular genre.

We found that 3 percent of the books in the dataset do not have a genre. Upon further investigation, we discovered that these books are actually short brochures rather than full-length books. This insight is particularly relevant to our model, as it means we cannot use these books in our predictions as the test set, as they do not have a genre rather than the genre being missing on the website. Using these books would negatively impact the accuracy of our model.

To proceed, I will perform natural language processing to enable the computer to comprehend textual data. After that, I will identify the most suitable machine learning model and train a supervised classification model to be able to guess the genres of the books.