Management Access Use Big Data - Final Project

Name: Riya Shetty

UID: 2000858722

1. Introduction

Goodreads is one of the world's largest websites for book lovers and for developers who wish to develop a recommender system using Goodreads data. My goal for this project is to evaluate the book's information in order to determine the user's reading habits and what variables contribute to the rising popularity of specific books. This is one of my personal projects that I created and worked on during my undergraduate studies. The recommendation system and analysis developed in this project were carried out using Python tools such as matplotlib and seaborn to make plots for analysis. I've always wanted to construct a website out of it and host it on a cloud server so that it's available to other book lovers like myself who are eager to explore new books and genres. Hence, for this purpose, I have made use of GCP.

For the course of this project, I have made use of the Goodreads data available on Kaggle. This data is the cleanest and large data which is taken directly from the Goodreads website and hence inspired me to use this data for my recommendation system. This data was entirely scraped via the Goodreads API, so kudos to them for providing such a simple interface to scrape their database.

Data Description: This dataset has 12 columns and 11,131 rows. There are approximately 6.6K authors in 31 unique languages, and the number of titles under which the book was published was approximately 10.4K different titles. The column description of the dataset looks like this:

1. **bookID** - A unique Identification number for each book.
2. **title** - The name under which the book was published.
3. **authors** - Names of the authors of the book. Multiple authors are delimited with -.
4. **average_rating** - The average rating of the book received in total.
5. **ISBN** - Another unique number to identify the book, is the International Standard Book Number.
6. **isbn13** - A 13-digit ISBN to identify the book, instead of the standard 11-digit ISBN.
7. **language_code** - Helps understand what is the primary language of the book. For instance, eng is standard for English.
8. **num_pages** - Number of pages the book contains.
9. **ratings_count** - Total number of ratings the book received.
10. **text_reviews_count** - Total number of written text reviews the book received.
11. **publication_date** - Date when the book was first published.
12. **publisher** - The name of the publisher.

2. Background

For the course of this project, I have made use of various technologies and algorithms. I would like to produce a brief description of each one of them so that it would easier to understand the practical implementation of each of them that I would be describing in the methodology:

1. **Python Libraries**: Python is a general-purpose, high-level programming language. Its design concept prioritizes code readability by employing heavy indentation. Python is garbage-collected and dynamically typed. It is compatible with a variety of programming paradigms, including structured, object-oriented, and functional programming.
2. **PySpark**: PySpark is considered a Python interface for Apache Spark. Python APIs may be used to develop apps in PySpark. This interface also allows you to utilize PySpark Shell to interactively examine
3. **Google Cloud Platform**: GCP allows one to choose between computing, storage, big data, machine learning, and application services for your web, mobile, analytics, and, back-end solutions.
4. **K-means Algorithm:** K-means clustering is a basic and widely used unsupervised machine learning technique. In data mining, the K-means method begins with a set of randomly chosen centroids that serve as the starting points for each cluster and then performs iterative (repetitive) computations to maximize the locations of the centroids.
5. **Recommendation System:** A recommender system, also known as a recommendation system, is a type of information filtering system that provides recommendations for items that are most relevant to a certain user.

The project primarily focuses on answering these major questions:

1. What is the average rating of any user per author?
2. Does the number of pages make an impact on ratings and popularity?
3. What does the average distribution of the books per author look like?
4. Which language is highly used for the publication of the books?
5. Can ratings be used to recommend books?

3. Methodology

I made use of the Google Cloud Platform to create a recommendation system since a cloud platform is highly useful in constructing efficient and scalable apps.

I created this recommendation system in PySpark, a distributed computing framework that does fast dynamic, large-scale data processing by using parallel computing. I used Spark API calls to leverage Python with the PySpark framework. It can do both memory and disk computations for faster data processing.

There are certain steps that need to be done to create storage on the Google cloud. These steps are linked to each other and hence need to be followed in an orderly manner. The steps followed for this include:

Step 1: Create a project on the Google Cloud Platform.

Step 2: Create a bucket on the Google Cloud Platform and then upload the books dataset onto the bucket. Primarily, buckets in GCP are used for storage purposes.

Step 3: In the next step, I created a Virtual Private Cloud (VPC) network.

Step 4: Since the VPC network is now created I focused on creating clusters using dataproc available on GCP.

Step 5: Now that the clusters have been made, the next step would include the process of loading the data, and performing the quality assurance techniques using Pyspark.

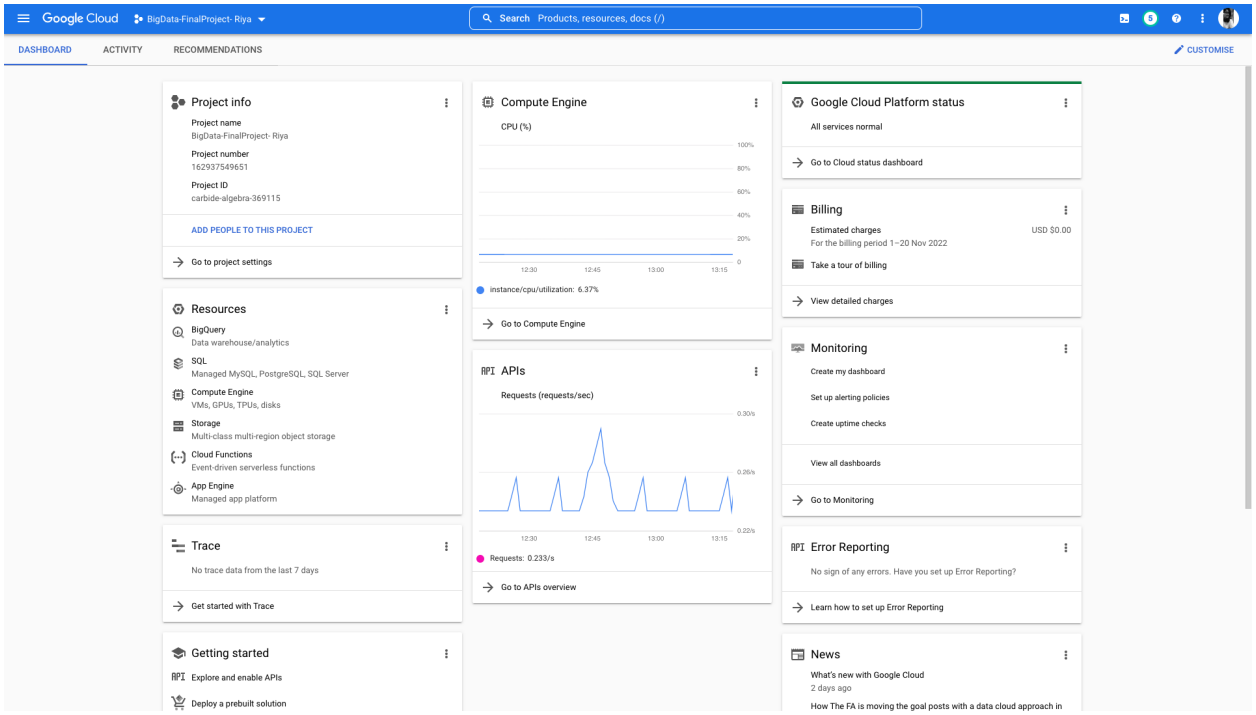
Step 6: I imported the dataset from Pyspark to Python to perform the data visualizations and do the required analysis.

Step 7: Data Visualisations and the recommendation system using the K-Means algorithm.

I would like to present a brief overlook of each of the steps mentioned above:

1. **<u>Project creation on GCP</u>**

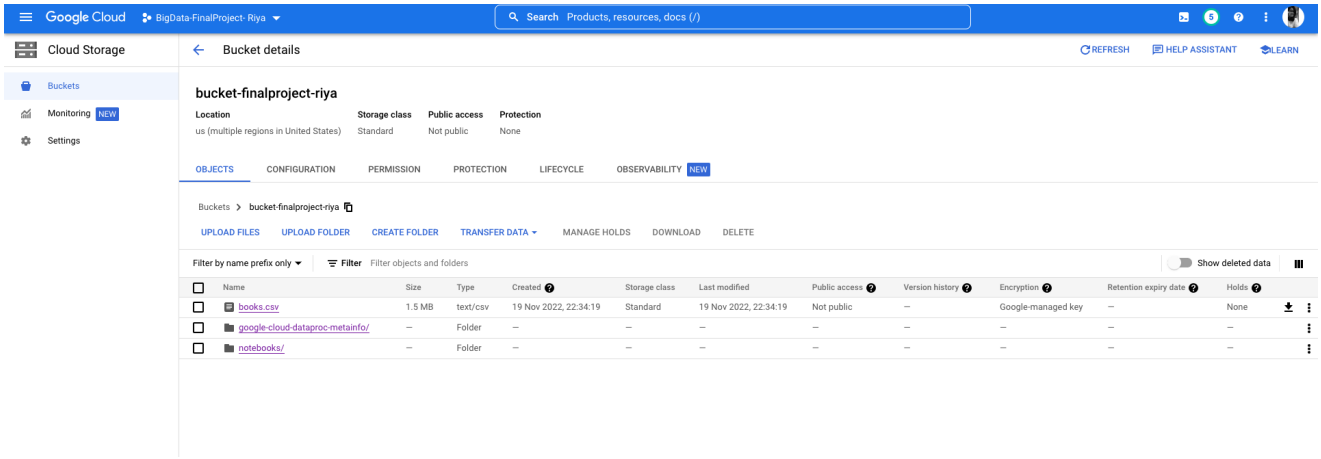
   For this, I created a project with the name, BigData-FinalProject- Riya. The follwoing is the configuration of the projection created.

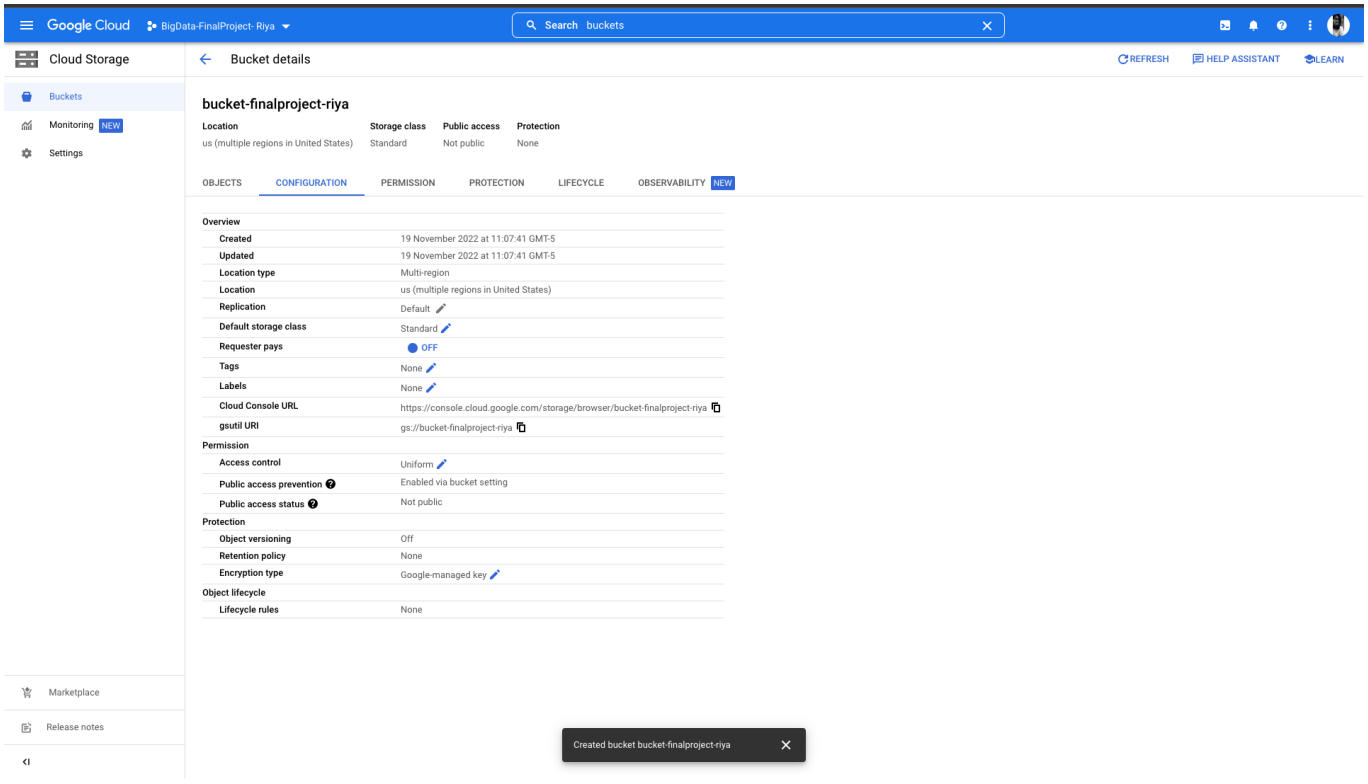


2. **<u>Bucket projection on GCP</u>**

   The downloaded dataset from Kaggle needs to be used for the visualisations and analysis. Hence, for this purpose I need to create a storage on the GCP and for this purpose I created a bucket namely,

bucket-finalproject-riya with the necessary configurations shown below. Here, I loaded the CSV file of the books dataset which I would be using for analysis.





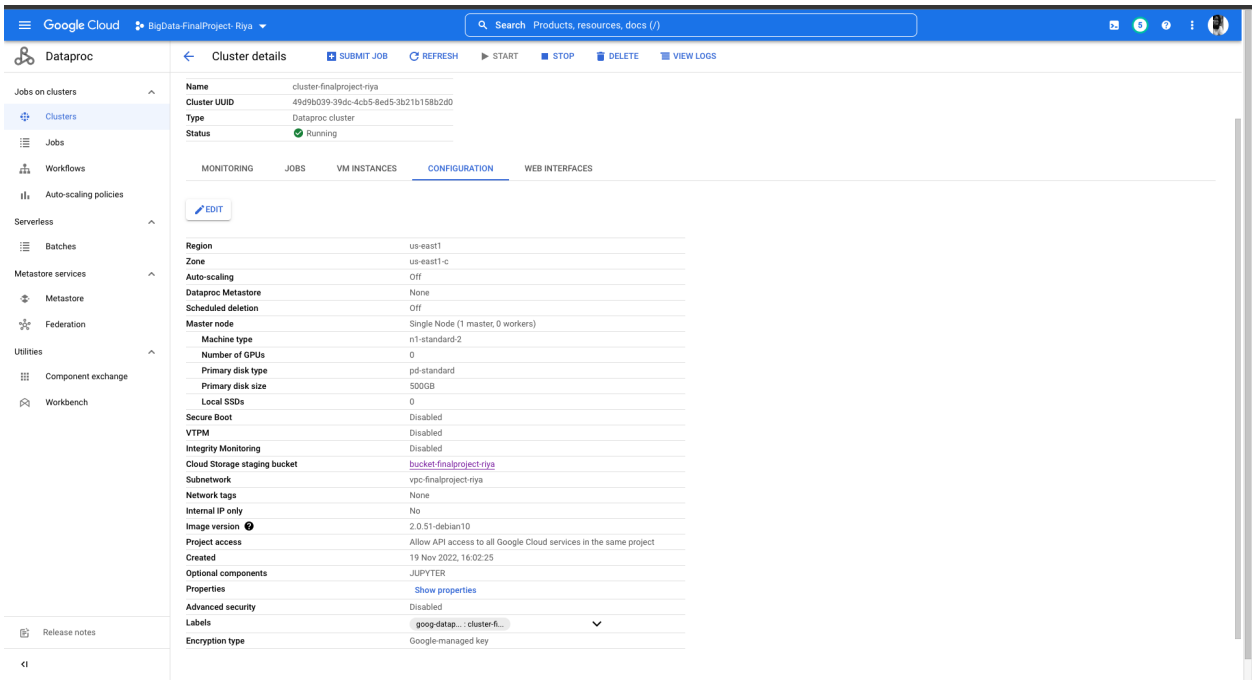
3. Creation of VPC network

Virtual Private Cloud (VPC) is nothing but a virtualized version of a physical network that connects to on-premises networks and distributes traffic from the Google cloud. For this I created a virtual cloud namely, vpc-finalproject-riya. Search for VPC network in the search bar → Create a VPC network by

naming your project → Select the Dynamic Routing mode as Regional and MAximum transmission unit as 1460 and then click on create.

4. **Creation of Dataproc clusters**

Now that the VPC network is created, I would construct a cluster cluster-finalproject-riya by selecting Single node with 1 master. I enabled component gateway and added Jupyter Notebook as an option. I set up my cluster with the following specifications: n1-standard-2, 2 vCPU, and 7.5 GB memory. I also customized my cluster by selecting the primary network as the VPC network I built in the previous part and the storage stating bucket as the bucket I created in the previous section.



5. Data Loading and quality assurance using Pyspark

Once the cluster has loaded up and running → Choose the web-interface on the dataproc clusters age → Open the Jupyter notebook option → Create a new folder in the local disk folder → Create a new PySpark Notebook → Do all the necessary imports → Perform the data preprocessing in the pyspark and python to check and remove any null values and duplicate values.

In [1]:
```
# Necessary Imports
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn import neighbors
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from scipy.cluster.vq import kmeans, vq
import seaborn as sns
from matplotlib.lines import Line2D
import sys
from collections import Counter
from pyspark.sql.functions  import *
from pyspark.sql.types import *
from pyspark import SparkConf, SparkContext
from pyspark.ml.recommendation import ALS
from pyspark.sql import SparkSession
from pyspark.sql.types import IntegerType
from pyspark.sql.functions import col
from pyspark.sql.types import DateType
from pyspark.sql.functions import col,isnan, when, count
```

In [2]:
```
# Necessary Configurations
/gateway/default/node/conf?host&port = SparkConf().setMaster("local[*]").setAppName("Books")
/gateway/default/node/conf?host&port.set("spark.executor.memory", "6G")
/gateway/default/node/conf?host&port.set("spark.driver.memory", "2G")
/gateway/default/node/conf?host&port.set("spark.executor.cores", "4")
/gateway/default/node/conf?host&port.set("spark.serializer", "org.apache.spark.serializer.KryoSerializer")
/gateway/default/node/conf?host&port.set("spark.default.parallelism", "4")
spark._jsc.hadoopConfiguration().set('fs.gs.impl', 'com.google.cloud.hadoop.fs.gcs.GoogleHadoopFileSystem')
```

- Create the spark session and load the dataset into spark

```
In [3]: spark = SparkSession.builder.config(/gateway/default/node/conf?host&port = /gateway/default/node/conf?host&port).appName("spark session").getOrCreate()
```

Load the data into Pyspark

```
In [4]: books = spark.read.load("gs://bucket-finalproject-riya/books.csv",
                               format = 'csv',
                               sep = ',',
                               header = 'true',
                               inferSchema = 'true').cache()
```

- Check for the null values and duplicate values using pyspark commands

Data Preprocessing

Further I will be making use of the pyspark and python libraries and functions to understand and analyze the data

```
In [*]: # Lets check the count of the books in the dataframe
        print(books.count())
```

```
[Stage 4:>                                                          (0 + 1) / 1]
11123
```

I will also check for the duplicate values in the data. For this I will count the number of distinct books in the dataset. If the count of the distinct books is equal to the len of the books calculated above then there are no duplicate values in the data.

```
In [*]: # Lets check for the duplicate values.
        (books.dropDuplicates()).count() == (books.count())
```

```
Out[6]: True
```

```
In [7]: books.printSchema()
```

```
root
 |-- bookID: integer (nullable = true)
 |-- title: string (nullable = true)
 |-- authors: string (nullable = true)
 |-- average_rating: double (nullable = true)
 |-- isbn: string (nullable = true)
 |-- isbn13: double (nullable = true)
 |-- language_code: string (nullable = true)
 |-- num_pages: integer (nullable = true)
 |-- ratings_count: integer (nullable = true)
 |-- text_reviews_count: integer (nullable = true)
 |-- publication_date: string (nullable = true)
 |-- publisher: string (nullable = true)
```

Checking for null values

```
In [8]: books.select([count(when(isnan(c) | col(c).isNull(), c)).alias(c) for c in books.columns]).show()
```

```
[Stage 6:>                                                          (0 + 1) / 1]
+------+-----+-------+--------------+----+------+-------------+---------+-------------+------------------+-----------
-----+---------+
|bookID|title|authors|average_rating|isbn|isbn13|language_code|num_pages|ratings_count|text_reviews_count|publication
_date|publisher|
+------+-----+-------+--------------+----+------+-------------+---------+-------------+------------------+-----------
-----+---------+
|     0|    0|      0|             0|   0|     0|            0|        0|            0|                 0|
0|        0|
+------+-----+-------+--------------+----+------+-------------+---------+-------------+------------------+-----------
-----+---------+
```

6. Conversion from Pyspark to Python

In the next step I'm converting the dataset into a dataframe so as to perform the analysis and visualizations. And I have generated the generated the summary statistics for the dataset as well.

In [7]:
```
bk = books.toPandas()
df = pd.DataFrame(bk)
df.head()
```

```
ERROR:root:Exception while sending command.
Traceback (most recent call last):
  File "/usr/lib/spark/python/lib/py4j-0.10.9-src.zip/py4j/java_gateway.py", line 1200, in send_command
    answer = smart_decode(self.stream.readline()[:-1])
  File "/opt/conda/miniconda3/lib/python3.8/socket.py", line 669, in readinto
    return self._sock.recv_into(b)
ConnectionResetError: [Errno 104] Connection reset by peer

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "/usr/lib/spark/python/lib/py4j-0.10.9-src.zip/py4j/java_gateway.py", line 1033, in send_command
    response = connection.send_command(command)
  File "/usr/lib/spark/python/lib/py4j-0.10.9-src.zip/py4j/java_gateway.py", line 1211, in send_command
    raise Py4JNetworkError(
py4j.protocol.Py4JNetworkError: Error while receiving
```

Out[7]:

	bookID	title	authors	average_rating	isbn	isbn13	language_code	num_pages	ratings_count	text_reviews_count	publication_date	publisher
0	1	Harry Potter and the Half-Blood Prince (Harry ...	J.K. Rowling/Mary GrandPré	4.57	439785960	9780439785969	eng	652	2095690	27591	9/16/2006	Scholastic Inc.
1	2	Harry Potter and the Order of the Phoenix (Har...	J.K. Rowling/Mary GrandPré	4.49	439358078	9780439358071	eng	870	2153167	29221	9/1/2004	Scholastic Inc.
2	4	Harry Potter and the Chamber of Secrets (Harry...	J.K. Rowling	4.42	439554896	9780439554893	eng	352	6333	244	11/1/2003	Scholastic
3	5	Harry Potter and the Prisoner of Azkaban (Harr...	J.K. Rowling/Mary GrandPré	4.56	043965548X	9780439655484	eng	435	2339585	36325	5/1/2004	Scholastic Inc.
4	8	Harry Potter Boxed Set Books 1-5 (Harry Potte...	J.K. Rowling/Mary GrandPré	4.78	439682584	9780439682589	eng	2690	41428	164	9/13/2004	Scholastic

In [8]:
```
# Dataset summary statistics
df.describe()
```

Out[8]:

	bookID	average_rating	isbn13	num_pages	ratings_count	text_reviews_count
count	11123.000000	11123.000000	1.112300e+04	11123.000000	1.112300e+04	11123.000000
mean	21310.856963	3.934075	9.759880e+12	336.405556	1.794285e+04	542.048099
std	13094.727252	0.350485	4.429758e+11	241.152626	1.124992e+05	2576.619589
min	1.000000	0.000000	8.987060e+09	0.000000	0.000000e+00	0.000000
25%	10277.500000	3.770000	9.780345e+12	192.000000	1.040000e+02	9.000000
50%	20287.000000	3.960000	9.780582e+12	299.000000	7.450000e+02	47.000000
75%	32104.500000	4.140000	9.780872e+12	416.000000	5.000500e+03	238.000000
max	45641.000000	5.000000	9.790008e+12	6576.000000	4.597666e+06	94265.000000

7. **Data Visualization and Recommendation System**

I have Data Visualisation using the python libraries and have mentioned the plots and the analysis on them in the plots below. Also, I have spoken about the K-menas clustering algorithm and how it has turned to be an efficient method in recommending books to any user.

4. Results

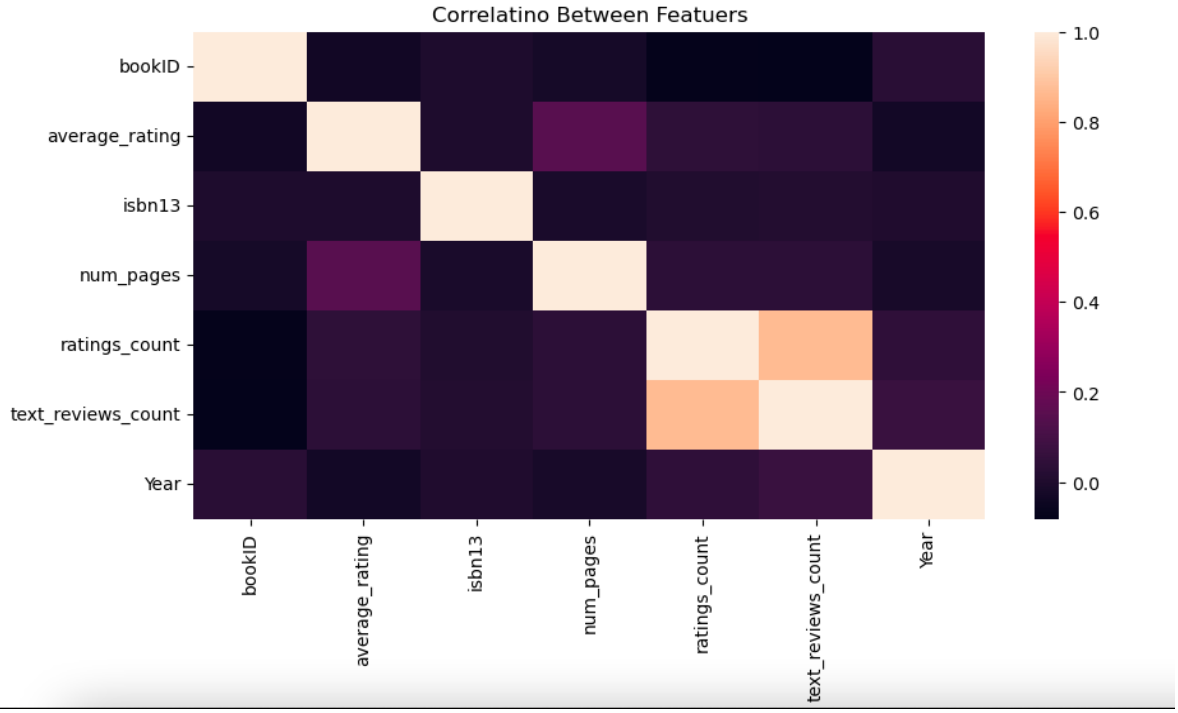
In this section I would be describing about all the plots that I have generated using various python libraries.

a. Correlation plot:

Correlation is a statistical term that describes how closely two variables are connected linearly (meaning they change together at a constant rate). It's a typical method for explaining simple interactions without stating a cause and effect link.

The best practice to understand the plot relation between different variables in the dataset is to use Correlation plot

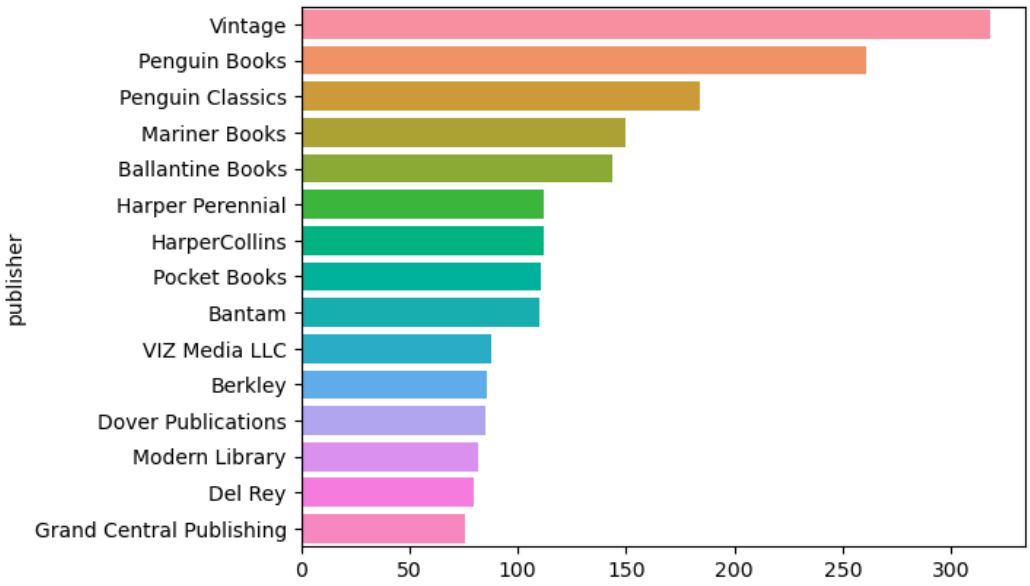
In [11]:
```
plt.figure(figsize=(10,5))
sns.heatmap(df.corr());
plt.title("Correlatino Between Featuers");
```



b. The top 15 publishers present in the dataset:

In [12]:
```
publisher = df.value_counts('publisher').sort_values(ascending=False).head(15)
sns.barplot(y=publisher.index,x = publisher)
```

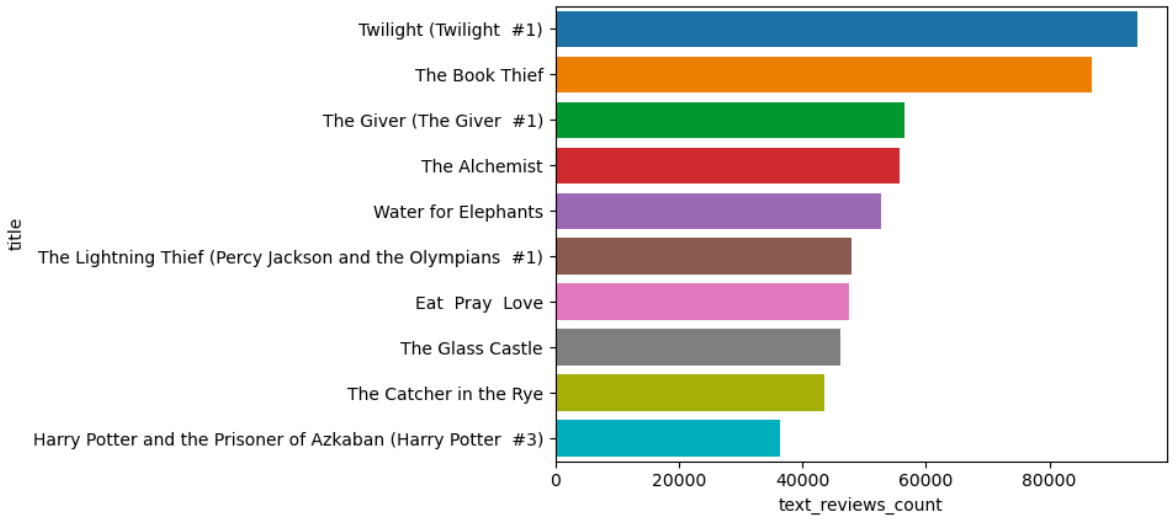
Out[12]: <AxesSubplot:ylabel='publisher'>



c. Highest text review count:

```
In [13]: x = df.loc[:,['title', 'text_reviews_count']].sort_values(by=['text_reviews_count'],ascending = False).head(10)
         sns.barplot(y = 'title',x='text_reviews_count',data = x)
```

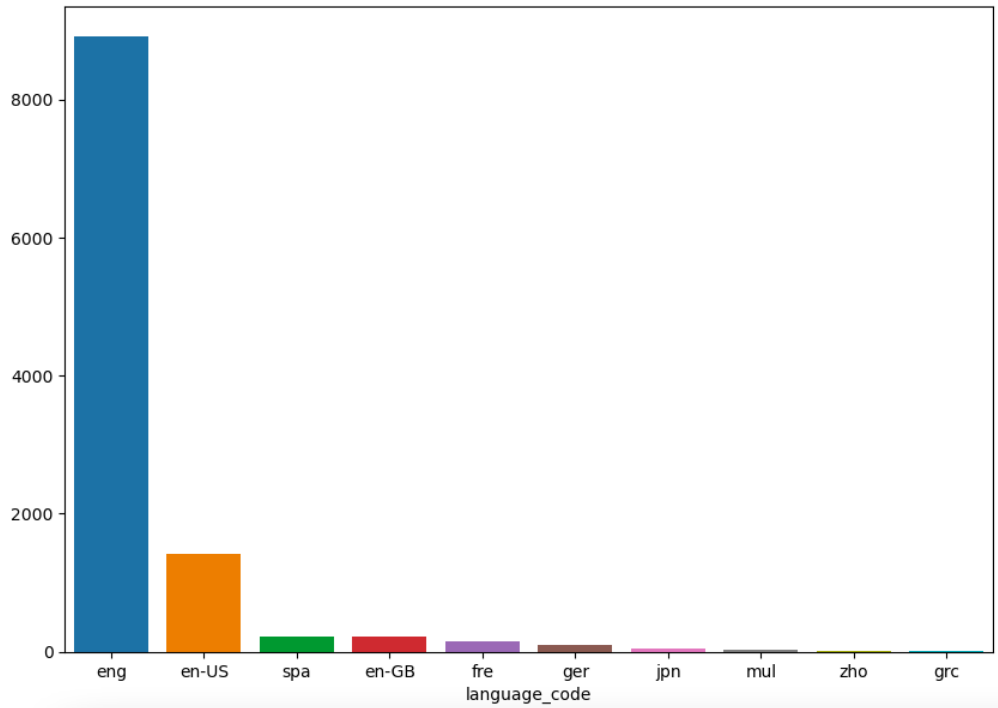
```
Out[13]: <AxesSubplot:xlabel='text_reviews_count', ylabel='title'>
```



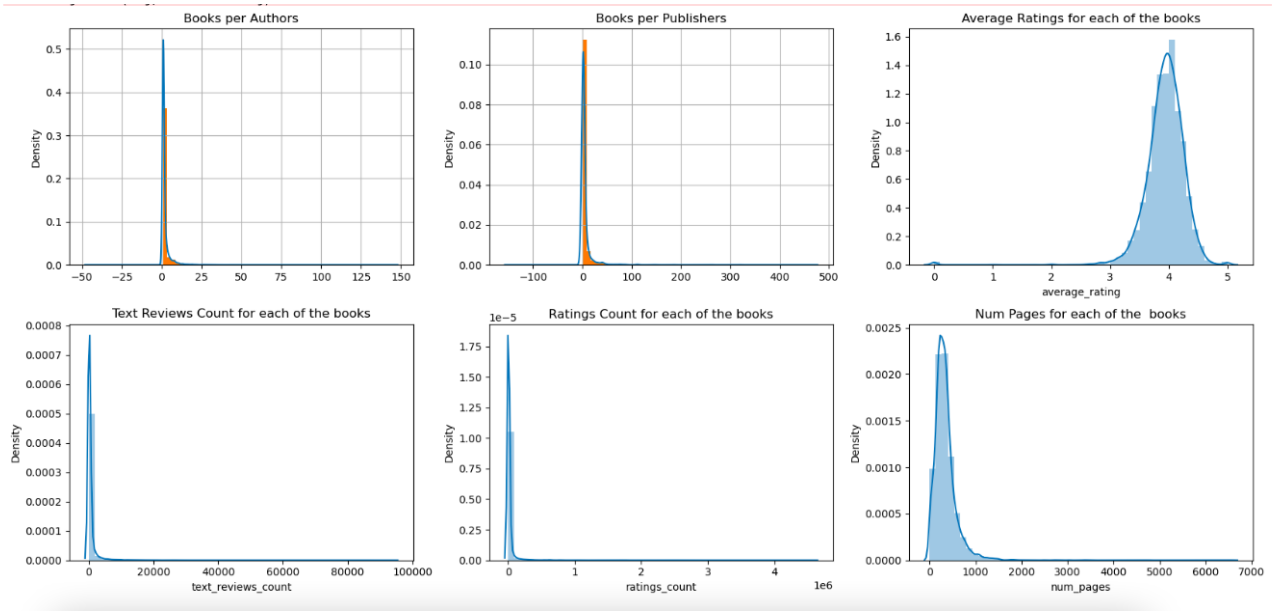
d. The 10 most common languages in the dataset:

```
language_code = df.value_counts('language_code').sort_values(ascending=False)
sns.barplot(x = language_code.index[:10],y = language_code[:10])
```

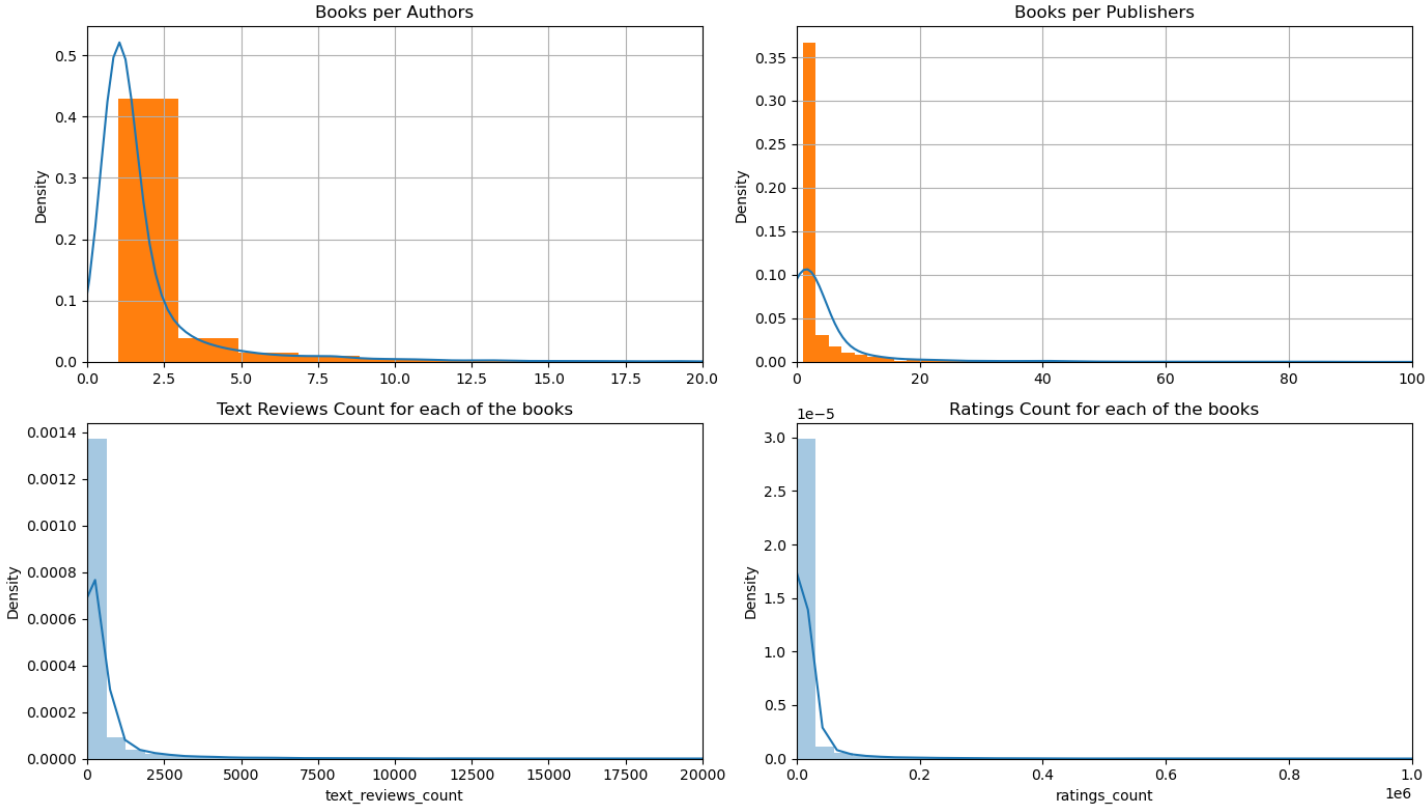
```
Out[14]: <AxesSubplot:xlabel='language_code'>
```



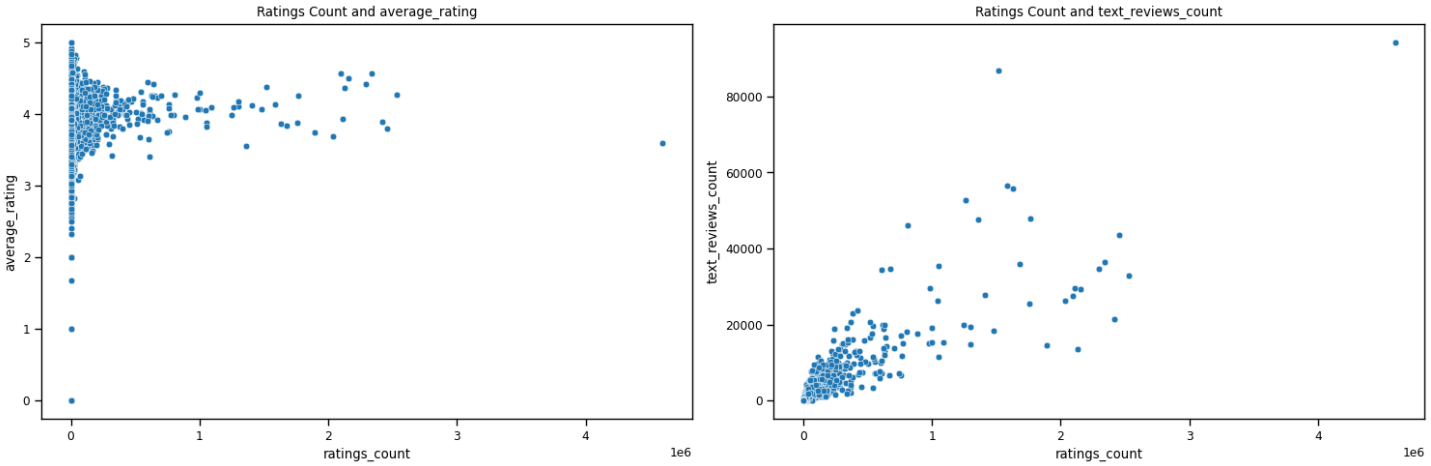
e. These particular graphs show up the probability density distribution among the variables in the dataset. But since we are not able to see the probability distribution for certain graphs like Books per Author, Books per Publisher etc. Hence, we would be plotting another graph for visualising thes graphs.



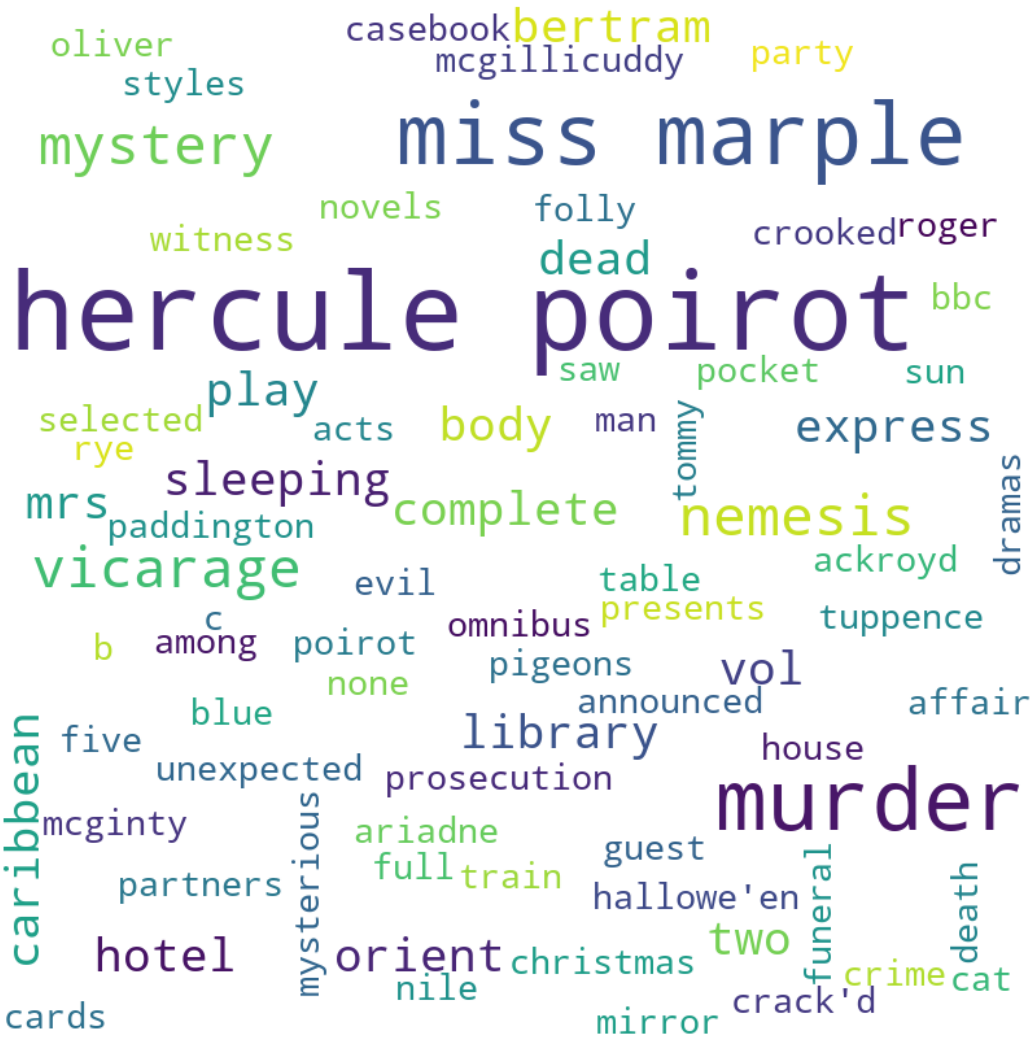
f. This is the plot to visulaise various graphs that we couldnt observe in the previous plotting



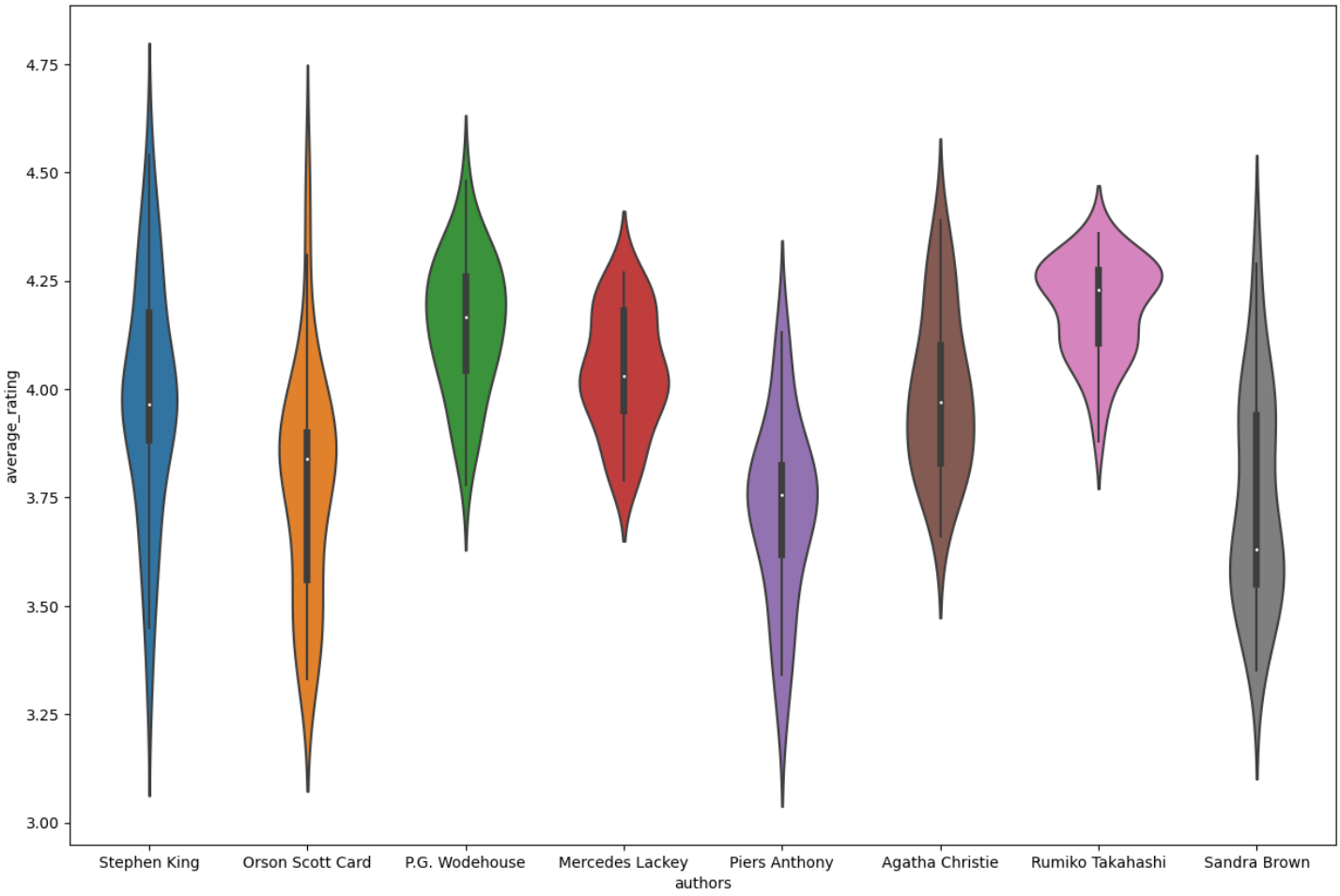
g. Relation between numerical variables:



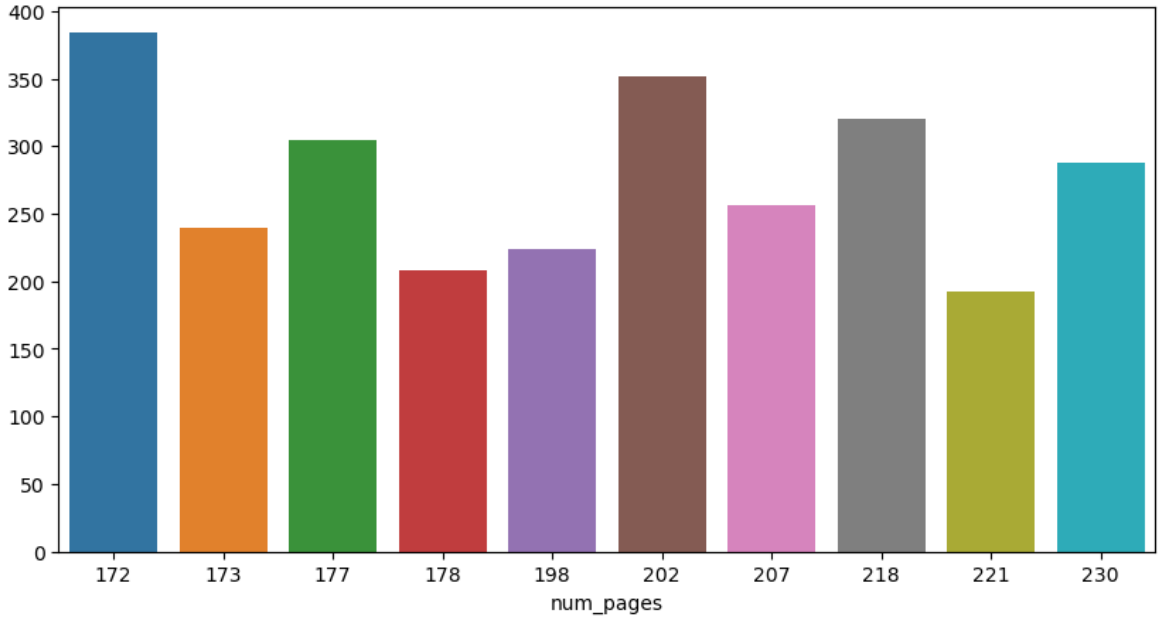
h. A wordcloud demonstration of the most common words used in the Agatha Christie books:



i. Average rating distribution among the top 8 authors:



j. The distribution of number of pages in the dataset:



- In this section, I'm intending to describe the K-Means algorithm, through which I have implemented the Recommendation System. In K-Means I first locate the k closest cases to it based on the distance with the neighboringpoints, and then predict the label using either a majority voting scheme or a weighted majority voting scheme.
- So for the scope of this project I have taken 2 variables into consideration which are average_rating and ratings_count so that the algorithm can understand datapoints within these 2 groups. Its also an observation that when the count increased, the rating would end up near the clusters. The green squares indicate the centroids of the clusters. The average rating looks to get sparser as the number of ratings falls, with increased volatility and less accuracy. I have set up the value of the number of clusters as 4. Due to this 4 clusters will be developed for the average_rating and ratings_count values.

Recommendation System

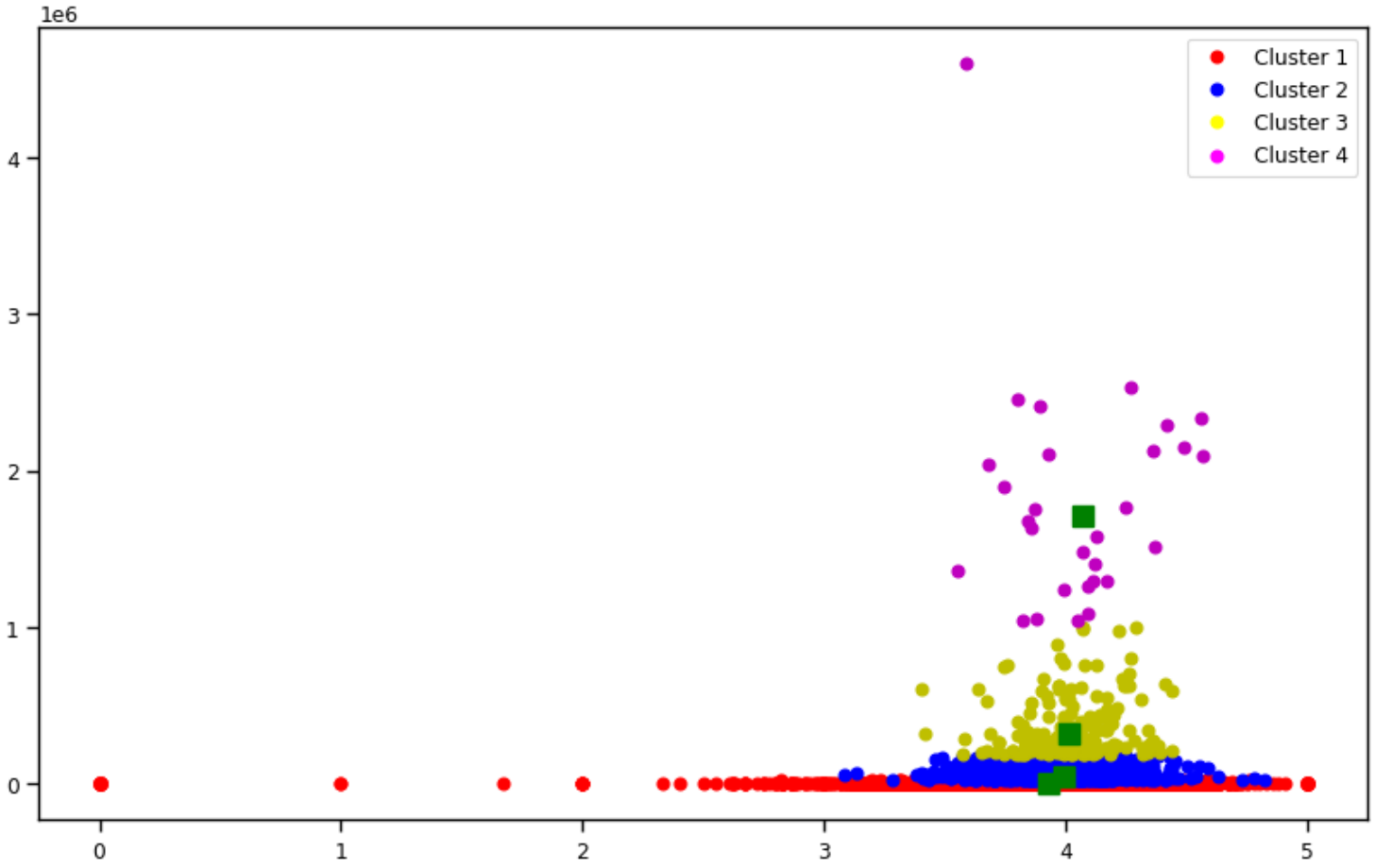
In [61]:
```
def idfromname(name):
    return df[df["title"]==name].index.tolist()[0]

all_books_names = list(df.title.values)

def recommendedbooks(query=None,id=None):
    if id:
        for id in indices[id][1:]:
            print(df.iloc[id]["title"])
    if query:
        found_id = idfromname(query)
        for id in indices[found_id][1:]:
            print(df.iloc[id]["title"])
```

In [62]:
```
recommendedbooks("Angels & Demons (Robert Langdon  #1)")
```

```
The Catcher in the Rye
Animal Farm
The Hobbit  or There and Back Again
Lord of the Flies
Harry Potter and the Chamber of Secrets (Harry Potter  #2)
```

In [63]:
```
idfromname("Animal Farm")
```

Out[63]: 2114



5. Discussion

This dataset exploration has been one of the most wonderful experience since, I got learn a lot about pyspark and GCP in depth. I learnt various steps such as

1. Creation of the buckets for storage which I had performed in one of the previous assignments but through this assignment I understood how to load the dataset into the bucket and how to load this dataset into Pyspark using the GCP link which looks like *gs://bucket-finalproject-riya/books.csv*.
2. Using dataproc in the GCP to create clusters helped me understand how I can multiprocess in the GCP. One can make use of Jupyter notebook but at the same time make use of the Hiveserver, in short parallel processing becomes much easier.
3. I intend to extend this project by making a website and hence making this dynamic and for this I will be needing the buckets, clusters in the future. I have always explored the ML and other analytics in the GCP and I understand how it can be thoroughly used to make the whole project dynamic.
4. I believe cloud computing is an ever growing field and will always be in need, since one can make use of ML (which are included in GCP), analytics and various other fields on a cloud server without worrying for the loss of data.

Following to this visualizations I would like to share various interpretations about the dataset.

1. It can be seen from the correlation that most of the variables in the dataset are not closely related to each other except for the fact that the text review count of each user and the rating count depend on each other. It can be said that the users who gave a rating also gave a text review to the book
2. The top 5 highly rated authors are Stephen King, P.G. Wodehouse, Rumiko Takahashi, Orson Scott Card, Agatha Christie.
3. From the density distribution graph, it can be seen that the as the rating count increases the rating of the book is nearing 4, whereas the rating counts decrease the rating becomes more sparse.
4. Finally, with the implementation of K-Means Clustering algorithm I was able to recommend books to the user by implementing certain functions like idfromname and recommendedbooks.

Problems encountered during the implementation of the project:

1. While loading the pyspark data, I was getting a java server error. Hence to resolve this issue I referred to the assignments and online resources like stackoverflow to understand where I was going wrong and hence had to certain configurations for the data loading process.
2. While creating the clusters I was not able to implement it. This was because I had not setup my customised VPC as the default network. Since the VPC network was based on the bucket created previously. Once this was done, the issue was resolved.

6. Conclusion

With the use of the assignments, I was able to create a pyspark object in which I loaded my dataset from the GCP bucket and then check for the null values, duplicate values in the dataset. This project helped strengthen my knowledge and skills in various domains such as virtual networks, parallel processing. It helped me deepen my knowledge in the Google cloud platform, which would help me in the near future. With the help of various online resources as well I was able to successfully implement a recommendation system in which I made use of a cloud platform for storage and parallel processing and pyspark to load and convert the data in python for analysis. I further intend to develop this project and add more dependencies via GCP for my project.

7. References

Dataset: https://www.kaggle.com/datasets/jealousleopard/goodreadsbooks

1. https://www.jmp.com/en_us/statistics-knowledge-portal/what-is-correlation.html
2. https://www.kaggle.com/code/greeshmagirish/goodreads-eda-with-matplotlib-and-seaborn/notebook
3. https://www.kaggle.com/code/rowancurry/eda-on-goodreads-dataset
4. https://www.kaggle.com/code/doreenl/goodreads-books-recommendation-a-complete-eda
5. https://kontext.tech/article/689/pyspark-read-file-in-google-cloud-storage
6. https://towardsdatascience.com/how-did-we-build-book-recommender-systems-in-an-hour-part-2-k-nearest-neighbors-and-matrix-c04b3c2ef55c