Book Recommendation System

By Mayank Prajapati

Introduction

Welcome to our journey of building a Book Recommendation System in Python, right here in Jupyter Notebook. In this project, we'll walk you through the exciting process of creating a personalized book recommendation system, step by step. By the end of this project, you'll have a robust system that suggests books tailored to individual tastes and preferences.

Key Steps in our Project:

1. **Data Collection**: We'll start by gathering a diverse dataset of books. This dataset will include book titles, authors, genres, user ratings, and more.
2. **Feature Engineering**: We'll explore and manipulate the dataset to create meaningful features for our recommendation system. This might include text-based features from book descriptions or user-related features.
3. **Exploratory Data Analysis (EDA)**: Before diving into model building, we'll perform EDA to gain insights into the dataset. We'll visualize trends, patterns, and relationships within the data.
4. **Data Preprocessing**: Data quality is crucial. We'll clean, handle missing values, and transform data to ensure it's ready for modeling.
5. **Recommendation Model Building**: This is where the magic happens! We'll implement various recommendation algorithms, such as collaborative filtering, content-based filtering, or hybrid models, to generate book recommendations.

Now, let's embark on this exciting journey of creating a book recommendation system, opening the doors to a world of personalized literary exploration!

Importing necessary libraries

In [1]:
```
import pandas as pd
import numpy as np

# for data visualisation
import matplotlib.pyplot as plt
import seaborn as sns

# for interactive plots
import ipywidgets
from ipywidgets import interact
from ipywidgets import interact_manual
```

Importing Dataset

In [2]:
```
df = pd.read_csv(r"C:\Users\dell\Downloads\books.csv", error_bad_lines=False)
```

```
C:\Users\dell\AppData\Local\Temp\ipykernel_15932\1584259486.py:1: FutureWarning: The error_bad_lines argument has
been deprecated and will be removed in a future version. Use on_bad_lines in the future.


  df = pd.read_csv(r"C:\Users\dell\Downloads\books.csv", error_bad_lines=False)
Skipping line 3350: expected 12 fields, saw 13
Skipping line 4704: expected 12 fields, saw 13
Skipping line 5879: expected 12 fields, saw 13
Skipping line 8981: expected 12 fields, saw 13
```

Exploring Data

In [3]:
```
df.head(10)
```

Out[3]:

	bookID	title	authors	average_rating	isbn	isbn13	language_code	num_pages	ratings_count	text_reviews_cou
0	1	Harry Potter and the Half-Blood Prince (Harry ...	J.K. Rowling/Mary GrandPré	4.57	0439785960	9780439785969	eng	652	2095690	275
1	2	Harry Potter and the Order of the Phoenix (Har...	J.K. Rowling/Mary GrandPré	4.49	0439358078	9780439358071	eng	870	2153167	292
2	4	Harry Potter and the Chamber of Secrets (Harry...	J.K. Rowling	4.42	0439554896	9780439554893	eng	352	6333	2
3	5	Harry Potter and the Prisoner of Azkaban (Harr...	J.K. Rowling/Mary GrandPré	4.56	043965548X	9780439655484	eng	435	2339585	363
4	8	Harry Potter Boxed Set Books 1-5 (Harry Potte...	J.K. Rowling/Mary GrandPré	4.78	0439682584	9780439682589	eng	2690	41428	1
5	9	Unauthorized Harry Potter Book Seven News: "Ha...	W. Frederick Zimmerman	3.74	0976540606	9780976540601	en-US	152	19	
6	10	Harry Potter Collection (Harry Potter #1-6)	J.K. Rowling	4.73	0439827604	9780439827607	eng	3342	28242	8
7	12	The Ultimate Hitchhiker's Guide: Five Complete...	Douglas Adams	4.38	0517226952	9780517226957	eng	815	3628	2
8	13	The Ultimate Hitchhiker's Guide to the Galaxy ...	Douglas Adams	4.38	0345453743	9780345453747	eng	815	249558	40
9	14	The Hitchhiker's Guide to the Galaxy (Hitchhik...	Douglas Adams	4.22	1400052920	9781400052929	eng	215	4930	4

In [4]:
```
df.tail(10)
```

Out[4]:

	bookID	title	authors	average_rating	isbn	isbn13	language_code	num_pages	ratings_count	text_revie
11113	45617	O Cavalo e o Seu Rapaz (As Crónicas de Nárnia ...	C.S. Lewis/Pauline Baynes/Ana Falcão Bastos	3.92	9722330551	9789722330558	por	160	207	
11114	45623	O Sobrinho do Mágico (As Crónicas de Nárnia #1)	C.S. Lewis/Pauline Baynes/Ana Falcão Bastos	4.04	9722329987	9789722329989	por	147	396	
11115	45625	A Viagem do Caminheiro da Alvorada (As Crónica...	C.S. Lewis/Pauline Baynes/Ana Falcão Bastos	4.09	9722331329	9789722331326	por	176	161	
11116	45626	O Príncipe Caspian (As Crónicas de Nárnia #4)	C.S. Lewis/Pauline Baynes/Ana Falcão Bastos	3.97	9722330977	9789722330978	por	160	215	
11117	45630	Whores for Gloria	William T. Vollmann	3.69	0140231579	9780140231571	en-US	160	932	
11118	45631	Expelled from Eden: A William T. Vollmann Reader	William T. Vollmann/Larry McCaffery/Michael He...	4.06	1560254416	9781560254416	eng	512	156	
11119	45633	You Bright and Risen Angels	William T. Vollmann	4.08	0140110879	9780140110876	eng	635	783	
11120	45634	The Ice-Shirt (Seven Dreams #1)	William T. Vollmann	3.96	0140131965	9780140131963	eng	415	820	
11121	45639	Poor People	William T. Vollmann	3.72	0060878827	9780060878825	eng	434	769	
11122	45641	Las aventuras de Tom Sawyer	Mark Twain	3.91	8497646983	9788497646987	spa	272	113	

In [5]:
```
df.shape
```

Out[5]:
```
(11123, 12)
```

In [6]:
```
df.columns
```

Out[6]:
```
Index(['bookID', 'title', 'authors', 'average_rating', 'isbn', 'isbn13',
       'language_code', '  num_pages', 'ratings_count', 'text_reviews_count',
       'publication_date', 'publisher'],
      dtype='object')
```

In [7]:
```
df.columns = df.columns.str.strip()
```

In [8]:
```
df.columns
```

Out[8]:
```
Index(['bookID', 'title', 'authors', 'average_rating', 'isbn', 'isbn13',
       'language_code', 'num_pages', 'ratings_count', 'text_reviews_count',
       'publication_date', 'publisher'],
      dtype='object')
```

In [9]:
```
df.dtypes
```

Out[9]:
```
bookID                int64
title                object
authors              object
average_rating      float64
isbn                 object
isbn13                int64
language_code        object
num_pages             int64
ratings_count         int64
text_reviews_count    int64
publication_date     object
publisher            object
dtype: object
```

In [10]:
```
df.describe()
```

Out[10]:

	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

In [11]:
```
df.describe(include = 'object')
```

Out[11]:

	title	authors	isbn	language_code	publication_date	publisher
count	11123	11123	11123	11123	11123	11123
unique	10348	6639	11123	27	3679	2290
top	The Iliad	Stephen King	0439785960	eng	10/1/2005	Vintage
freq	9	40	1	8908	56	318

In [12]:
```
df.isnull().sum()
```

Out[12]:
```
bookID                0
title                 0
authors               0
average_rating        0
isbn                  0
isbn13                0
language_code         0
num_pages             0
ratings_count         0
text_reviews_count    0
publication_date      0
publisher             0
dtype: int64
```

In [13]:
```
df.duplicated().any()
```

Out[13]:
```
False
```

In [14]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11123 entries, 0 to 11122
Data columns (total 12 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   bookID              11123 non-null  int64  
 1   title               11123 non-null  object 
 2   authors             11123 non-null  object 
 3   average_rating      11123 non-null  float64
 4   isbn                11123 non-null  object 
 5   isbn13              11123 non-null  int64  
 6   language_code       11123 non-null  object 
 7   num_pages           11123 non-null  int64  
 8   ratings_count       11123 non-null  int64  
 9   text_reviews_count  11123 non-null  int64  
 10  publication_date    11123 non-null  object 
 11  publisher           11123 non-null  object 
dtypes: float64(1), int64(5), object(6)
memory usage: 1.0+ MB
```

Feature Engineering

Extract Important Features
Reducing the size of Features
Creating new features from the existing ones

In [15]:
```
df.columns
```

Out[15]:
```
Index(['bookID', 'title', 'authors', 'average_rating', 'isbn', 'isbn13',
       'language_code', 'num_pages', 'ratings_count', 'text_reviews_count',
       'publication_date', 'publisher'],
      dtype='object')
```

In [16]:
```
df.isbn.nunique()
```

Out[16]:
```
11123
```

In [17]:
```
df.isbn13.nunique()
```

Out[17]:
```
11123
```

In [18]:
```
df.drop(['bookID', 'isbn', 'isbn13'], axis = 1, inplace = True)
```

In [19]:
```
df.columns
```

Out[19]:
```
Index(['title', 'authors', 'average_rating', 'language_code', 'num_pages',
       'ratings_count', 'text_reviews_count', 'publication_date', 'publisher'],
      dtype='object')
```

In [20]:
```
df.publication_date
```

Out[20]:
```
0         9/16/2006
1          9/1/2004
2         11/1/2003
3          5/1/2004
4         9/13/2004
            ...    
11118    12/21/2004
11119     12/1/1988
11120      8/1/1993
11121     2/27/2007
11122     5/28/2006
Name: publication_date, Length: 11123, dtype: object
```

In [21]:
```
df['year'] = df['publication_date'].str.split('/')
df['year'] = df['year'].apply(lambda x: x[2])
```

In [22]:
```
df.head(5)
```

Out[22]:

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

In [23]:
```
df.dtypes
```

Out[23]:
```
title                 object
authors               object
average_rating       float64
language_code         object
num_pages              int64
ratings_count          int64
text_reviews_count     int64
publication_date      object
publisher             object
year                  object
dtype: object
```

In [24]:
```
df['year'] = df['year'].astype('int')
```

In [25]:
```
df.dtypes
```

Out[25]:
```
title                 object
authors               object
average_rating       float64
language_code         object
num_pages              int64
ratings_count          int64
text_reviews_count     int64
publication_date      object
publisher             object
year                   int32
dtype: object
```

In [26]:
```
df.columns
```

Out[26]:
```
Index(['title', 'authors', 'average_rating', 'language_code', 'num_pages',
       'ratings_count', 'text_reviews_count', 'publication_date', 'publisher',
       'year'],
      dtype='object')
```

In [27]:
```
df['year'].min()
```

Out[27]:
```
1900
```

In [28]:
```
df['year'].max()
```

Out[28]:
```
2020
```

In [29]:
```
df.columns
```

Out[29]:
```
Index(['title', 'authors', 'average_rating', 'language_code', 'num_pages',
       'ratings_count', 'text_reviews_count', 'publication_date', 'publisher',
       'year'],
      dtype='object')
```

Exploratory Data Analysis

In [30]:
```
df[df['year'] == 2020][['title', 'authors','average_rating','language_code','publisher' ]]
```

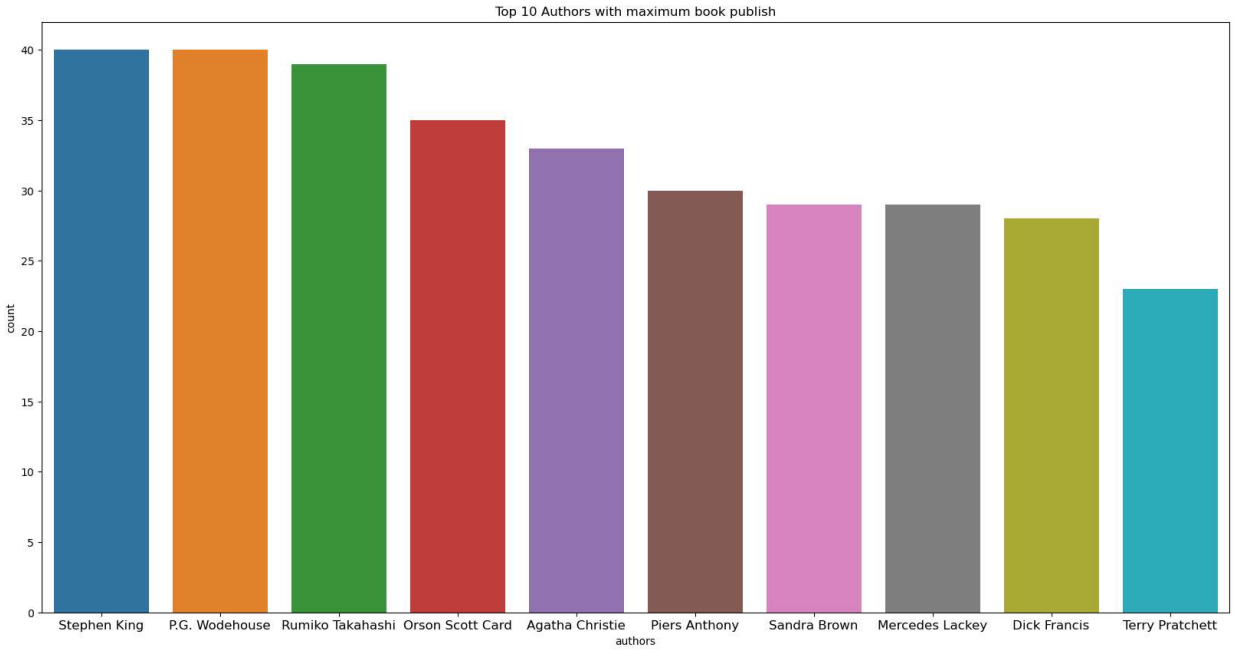
Out[30]:

	title	authors	average_rating	language_code	publisher
9664	A Quick Bite (Argeneau #1)	Lynsay Sands	3.91	eng	Avon

In [31]:
```
df.groupby(['year'])['title'].agg('count').sort_values(ascending = False).head(20)
```

Out[31]:
```
year
2006    1700
2005    1260
2004    1069
2003     931
2002     798
2001     656
2000     534
2007     518
1999     450
1998     396
1997     290
1996     250
1995     249
1994     220
1992     183
1993     165
1991     151
1989     118
1990     117
1987      88
Name: title, dtype: int64
```

In [32]:
```
plt.figure(figsize = (20,10))
sns.countplot(x = 'authors', data = df,
              order = df['authors'].value_counts().iloc[:10].index)
plt.title("Top 10 Authors with maximum book publish")
plt.xticks(fontsize = 12)
plt.show()
```



In [33]:
```
df.columns
```

Out[33]:
```
Index(['title', 'authors', 'average_rating', 'language_code', 'num_pages',
       'ratings_count', 'text_reviews_count', 'publication_date', 'publisher',
       'year'],
      dtype='object')
```

In [34]:
```
df.language_code.value_counts()
```

Out[34]:
```
eng      8908
en-US    1408
spa       218
en-GB     214
fre       144
ger        99
jpn        46
mul        19
zho        14
grc        11
por        10
en-CA       7
ita         5
enm         3
lat         3
swe         2
rus         2
srp         1
nl          1
msa         1
glg         1
wel         1
ara         1
nor         1
tur         1
gla         1
ale         1
Name: language_code, dtype: int64
```

In [35]:
```
df.groupby(['language_code'])[['average_rating',
                               'ratings_count',
                               'text_reviews_count']].agg('mean').style.background_gradient(cmap = 'Wistia')
```

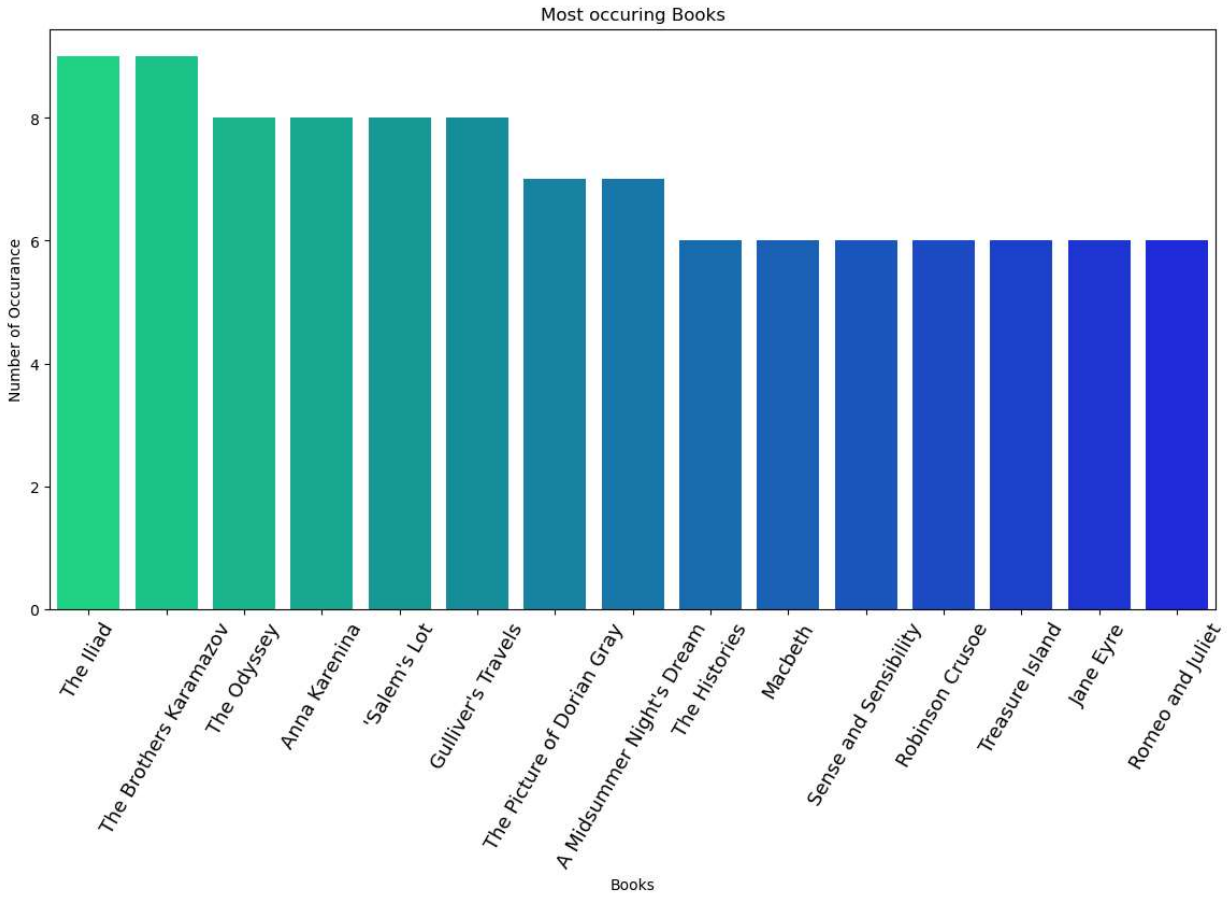
Out[35]:

language_code	average_rating	ratings_count	text_reviews_count
ale	4.360000	102.000000	16.000000
ara	3.550000	122.000000	12.000000
en-CA	4.025714	4086.714286	324.428571
en-GB	3.923411	2463.691589	104.060748
en-US	3.914659	3773.906960	160.357244
eng	3.934062	21570.272564	645.156601
enm	3.873333	3233.666667	84.000000
fre	3.971528	3277.319444	64.513889
ger	3.950101	234.727273	8.232323
gla	4.470000	11.000000	0.000000
glg	3.360000	36.000000	2.000000
grc	3.707273	52.454545	2.454545
ita	4.078000	3234.400000	55.800000
jpn	4.268696	68.304348	3.152174
lat	4.353333	114.666667	12.333333
msa	4.110000	28.000000	6.000000
mul	4.126316	386.631579	19.263158
nl	4.180000	67.000000	9.000000
nor	3.600000	86.000000	8.000000
por	3.945000	165.100000	13.500000
rus	4.255000	4477.000000	98.500000
spa	3.929312	4636.114679	91.123853
srp	0.000000	0.000000	0.000000
swe	3.455000	2671.000000	157.000000
tur	4.420000	1000.000000	41.000000
wel	5.000000	1.000000	0.000000
zho	4.456429	20.428571	0.500000

In [36]:
```
book = df['title'].value_counts()[:20]
book
```

Out[36]:
```
The Iliad                     9
The Brothers Karamazov        9
The Odyssey                   8
Anna Karenina                 8
'Salem's Lot                  8
Gulliver's Travels            8
The Picture of Dorian Gray    7
A Midsummer Night's Dream     7
The Histories                 6
Macbeth                       6
Sense and Sensibility         6
Robinson Crusoe               6
Treasure Island               6
Jane Eyre                     6
Romeo and Juliet              6
Collected Stories             6
The Secret Garden             6
The Scarlet Letter            6
The Great Gatsby              6
Frankenstein                  5
Name: title, dtype: int64
```

In [37]:
```
# to find most occuring book in our data
plt.figure(figsize = (14, 7))
book = df['title'].value_counts()[:15]
sns.barplot(x = book.index, y = book,
            palette = 'winter_r')
plt.title("Most occuring Books")
plt.ylabel("Number of Occurance")
plt.xlabel("Books")
plt.xticks(rotation=60, fontsize = 13)
plt.show()
```



In [38]:
```
sns.distplot(df['average_rating'])
plt.show()
```

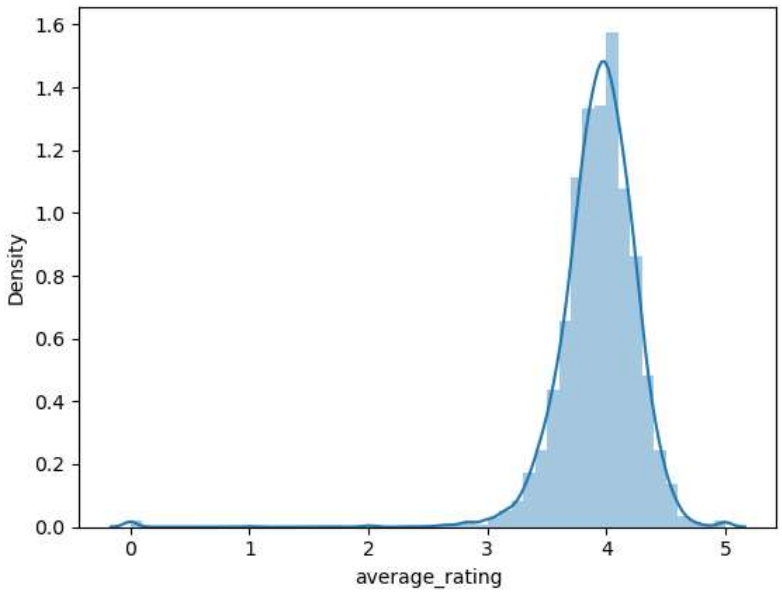
```
C:\Users\dell\AppData\Local\Temp\ipykernel_15932\3493288629.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751 (https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751)

  sns.distplot(df['average_rating'])
```



In [39]:
```
df[df.average_rating == df.average_rating.max()][['title','authors','language_code','publisher']]
```

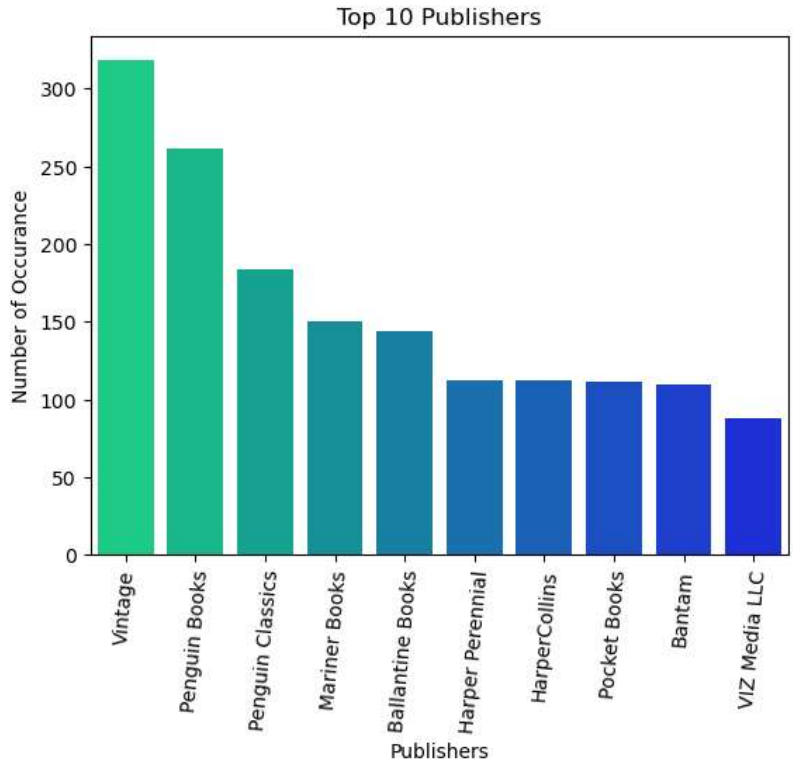
Out[39]:

	title	authors	language_code	publisher
624	Comoediae 1: Acharenses/Equites/Nubes/Vespae/P...	Aristophanes/F.W. Hall/W.M. Geldart	grc	Oxford University Press USA
786	Willem de Kooning: Late Paintings	Julie Sylvester/David Sylvester	eng	Schirmer Mosel
855	Literature Circle Guide: Bridge to Terabithia:...	Tara MacCarthy	eng	Teaching Resources
1243	Middlesex Borough (Images of America: New Jersey)	Middlesex Borough Heritage Committee	eng	Arcadia Publishing
4125	Zone of the Enders: The 2nd Runner Official St...	Tim Bogenn	eng	BradyGames
4788	The Diamond Color Meditation: Color Pathway to...	John Diamond	eng	Square One Publishers
4933	Bulgakov's the Master and Margarita: The Text ...	Elena N. Mahlow	eng	Vantage Press
5023	The Complete Theory Fun Factory: Music Theory ...	Ian Martin/Katie Elliott	eng	Boosey & Hawkes Inc
5474	The Goon Show Volume 4: My Knees Have Fallen ...	NOT A BOOK	eng	BBC Physical Audio
5476	The Goon Show Volume 11: He's Fallen in the W...	NOT A BOOK	eng	BBC Physical Audio
5647	Winchester Shotguns	Dennis Adler/R.L. Wilson	eng	Chartwell Books
5648	Colossians and Philemon: A Critical and Exeget...	R. McL. Wilson	eng	T&T Clark Int'l
6184	Taxation of Mineral Rents	Ross Garnaut	eng	Oxford University Press USA
6247	The New Big Book of America	Todd Davis/Marc Frey	eng	Courage Books
6775	Delwau Duon: Peintiadau Nicholas Evans = Symph...	Nicholas Evans/Rhonda Evans	wel	Y Lolfa
8544	Fanning the Flame: Bible Cross and Mission	Chris Green/Chris Wright/Paul Douglas Gardner	eng	Zondervan
9282	Oliver Wendell Holmes in Paris: Medicine Theo...	William C. Dowling	eng	University Press of New England
9324	Tyrannosaurus Wrecks (Stanley #1)	Laura Driscoll/Alisa Klayman-Grodsky/Eric ...	eng	Disney Press
9720	The Irish Anatomist: A Study of Flann O'Brien	Keith Donohue	eng	Academica Press
9847	The American Campaign: U.S. Presidential Campa...	James E. Campbell	eng	Texas A&M University Press
9893	His Princess Devotional: A Royal Encounter Wit...	Sheri Rose Shepherd	eng	Multnomah
10262	Bill Gates: Computer Legend (Famous Lives)	Sara Barton-Wood	eng	Raintree

In [40]:
```
publisher = df['publisher'].value_counts()[:20]
publisher
```

Out[40]:
```
Vintage                          318
Penguin Books                    261
Penguin Classics                 184
Mariner Books                    150
Ballantine Books                 144
Harper Perennial                 112
HarperCollins                    112
Pocket Books                     111
Bantam                           110
VIZ Media LLC                     88
Berkley                           86
Dover Publications                85
Modern Library                    82
Del Rey                           80
Tor Books                         76
Grand Central Publishing          76
Oxford University Press  USA      75
Oxford University Press           73
Scribner                          73
W. W. Norton  Company             68
Name: publisher, dtype: int64
```

In [41]:
```
publisher = df['publisher'].value_counts()[:10]
sns.barplot(x = publisher.index, y = publisher, palette = 'winter_r')
plt.title("Top 10 Publishers")
plt.ylabel("Number of Occurance")
plt.xlabel("Publishers")
plt.xticks(rotation = 85, fontsize = 10)
plt.show()
```



In [42]:
```
df.publisher.value_counts()
```

Out[42]:
```
Vintage                          318
Penguin Books                    261
Penguin Classics                 184
Mariner Books                    150
Ballantine Books                 144
                                ...
University of Calgary Press        1
Marlowe & Company                  1
University Press of America        1
Abstract Studio                    1
VeloPress                          1
Name: publisher, Length: 2290, dtype: int64
```

In [43]:
```
df.columns
```

Out[43]:
```
Index(['title', 'authors', 'average_rating', 'language_code', 'num_pages',
       'ratings_count', 'text_reviews_count', 'publication_date', 'publisher',
       'year'],
      dtype='object')
```

In [44]:
```
def recomd_books_publishers(x):
    a = df[df['publisher'] == x][['title', 'average_rating']]
    a = a.sort_values(by = 'average_rating', ascending = False)
    return a.head(10)
```

In [45]:
```
recomd_books_publishers('Vintage')
```

Out[45]:

	title	average_rating
7371	Remembrance of Things Past: Volume II - The Gu...	4.53
335	The Power Broker: Robert Moses and the Fall of...	4.51
10838	The Civil War Vol. 1: Fort Sumter to Perryville	4.42
1775	The Son Avenger (The Master of Hestviken #4)	4.40
1505	A Fine Balance	4.36
9626	Nobody Knows My Name	4.35
2267	The Stories of Vladimir Nabokov	4.30
3112	All of Us: The Collected Poems	4.30
8787	Selected Stories	4.28
4019	Selected Stories	4.28

In [46]:
```
recomd_books_publishers('Penguin Books')
```

Out[46]:

	title	average_rating
4244	The Complete Maus	4.55
5564	The Penguin Companion to European Literature	4.50
1381	Before The Mayflower A History of Black America	4.44
4602	Selected Non-Fictions	4.43
3011	The Read-Aloud Handbook	4.41
4551	Life With Jeeves (Jeeves #6 2 & 4)	4.39
1275	East of Eden	4.37
3304	Ludwig Wittgenstein: The Duty of Genius	4.36
4980	Life at Blandings	4.35
10867	The Portable Dorothy Parker	4.34

Recommending Books based on Publishers

In [47]:
```
@interact
def recomd_books_publishers(publisher_name = list(df['publisher'].value_counts().index)):
    a = df[df['publisher'] == publisher_name][['title', 'average_rating']]
    a = a.sort_values(by = 'average_rating', ascending = False)
    return a.head(10)
```

publisher_... Vintage

	title	average_rating
7371	Remembrance of Things Past: Volume II - The Gu...	4.53
335	The Power Broker: Robert Moses and the Fall of...	4.51
10838	The Civil War Vol. 1: Fort Sumter to Perryville	4.42
1775	The Son Avenger (The Master of Hestviken #4)	4.40
1505	A Fine Balance	4.36
9626	Nobody Knows My Name	4.35
2267	The Stories of Vladimir Nabokov	4.30
3112	All of Us: The Collected Poems	4.30
8787	Selected Stories	4.28
4019	Selected Stories	4.28

Recommending Books based on Authors

In [48]:
```
@interact
def recomd_books_authors(Authors_name = list(df['authors'].value_counts().index)):
    a = df[df['authors'] == Authors_name][['title', 'average_rating']]
    a = a.sort_values(by = 'average_rating', ascending = False)
    return a.head(10)
```

Authors_na… Robert A. Heinlein

	title	average_rating
4643	The Moon is a Harsh Mistress	4.17
151	The Door Into Summer	4.01
4787	Starship Troopers	4.01
4786	Starship Troopers	4.01
4785	Starship Troopers	4.01
4642	Citizen of the Galaxy	3.99
155	Time for the Stars (Heinlein's Juveniles #10)	3.97
4639	Tunnel in the Sky (Heinlein's Juveniles #9)	3.94
4036	Stranger In A Strange Land	3.92
152	Stranger in a Strange Land	3.92

In [49]:
```
df.columns
```

Out[49]:
```
Index(['title', 'authors', 'average_rating', 'language_code', 'num_pages',
       'ratings_count', 'text_reviews_count', 'publication_date', 'publisher',
       'year'],
      dtype='object')
```

Recommending Books based on Language

In [50]:
```
@interact
def recomd_books_lang(language = list(df['language_code'].value_counts().index)):
    a = df[df['language_code'] == language][['title', 'average_rating']]
    a = a.sort_values(by = 'average_rating', ascending = False)
    return a.head(10)
```

language eng

	title	average_rating
4933	Bulgakov's the Master and Margarita: The Text ...	5.0
8544	Fanning the Flame: Bible Cross and Mission	5.0
9324	Tyrannosaurus Wrecks (Stanley #1)	5.0
6247	The New Big Book of America	5.0
6184	Taxation of Mineral Rents	5.0
5648	Colossians and Philemon: A Critical and Exeget...	5.0
5647	Winchester Shotguns	5.0
9720	The Irish Anatomist: A Study of Flann O'Brien	5.0
5476	The Goon Show Volume 11: He's Fallen in the W...	5.0
5474	The Goon Show Volume 4: My Knees Have Fallen ...	5.0

Data Preprocessing

In [51]:
```
df.head(5)
```

Out[51]:

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

In [52]:
```
def num_to_obj(x):
    if x >0 and x <=1:
        return "between 0 and 1"
    if x > 1 and x <= 2:
        return "between 1 and 2"
    if x > 2 and x <=3:
        return "between 2 and 3"
    if x >3 and x<=4:
        return "between 3 and 4"
    if x >4 and x<=5:
        return "between 4 and 5"
df['rating_obj'] = df['average_rating'].apply(num_to_obj)
```

In [53]:
```
df['rating_obj'].value_counts()
```

Out[53]:
```
between 3 and 4    6285
between 4 and 5    4735
between 2 and 3      69
between 1 and 2       7
between 0 and 1       2
Name: rating_obj, dtype: int64
```

In [54]:
```
rating_df = pd.get_dummies(df['rating_obj'])
rating_df.head()
```

Out[54]:

	between 0 and 1	between 1 and 2	between 2 and 3	between 3 and 4	between 4 and 5
0	0	0	0	0	1
1	0	0	0	0	1
2	0	0	0	0	1
3	0	0	0	0	1
4	0	0	0	0	1

In [55]:
```
df.columns
```

Out[55]:
```
Index(['title', 'authors', 'average_rating', 'language_code', 'num_pages',
       'ratings_count', 'text_reviews_count', 'publication_date', 'publisher',
       'year', 'rating_obj'],
      dtype='object')
```

In [56]:
```
language_df = pd.get_dummies(df['language_code'])
language_df.head()
```

Out[56]:

	ale	ara	en-CA	en-GB	en-US	eng	enm	fre	ger	gla	...	nl	nor	por	rus	spa	srp	swe	tur	wel	zho
0	0	0	0	0	0	1	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	1	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	1	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	1	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	1	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

5 rows × 27 columns

In [57]:
```
features = pd.concat([rating_df,language_df, df['average_rating'],
                      df['ratings_count'], df['title']], axis = 1)
features.set_index('title', inplace= True)
features.head()
```

Out[57]:

title	between 0 and 1	between 1 and 2	between 2 and 3	between 3 and 4	between 4 and 5	ale	ara	en-CA	en-GB	en-US	...	por	rus	spa	srp	swe	tur	wel	zho	average_rating
Harry Potter and the Half-Blood Prince (Harry Potter #6)	0	0	0	0	1	0	0	0	0	0	...	0	0	0	0	0	0	0	0	4.57
Harry Potter and the Order of the Phoenix (Harry Potter #5)	0	0	0	0	1	0	0	0	0	0	...	0	0	0	0	0	0	0	0	4.49
Harry Potter and the Chamber of Secrets (Harry Potter #2)	0	0	0	0	1	0	0	0	0	0	...	0	0	0	0	0	0	0	0	4.42
Harry Potter and the Prisoner of Azkaban (Harry Potter #3)	0	0	0	0	1	0	0	0	0	0	...	0	0	0	0	0	0	0	0	4.56
Harry Potter Boxed Set Books 1-5 (Harry Potter #1-5)	0	0	0	0	1	0	0	0	0	0	...	0	0	0	0	0	0	0	0	4.78

5 rows × 34 columns

In [58]:
```
from sklearn.preprocessing import MinMaxScaler
```

In [59]:
```
scaler = MinMaxScaler()
features_scaled = scaler.fit_transform(features)
```

In [60]:
```
features_scaled
```

Out[60]:
```
array([[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
        0.00000000e+00, 9.14000000e-01, 4.55816060e-01],
       [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
        0.00000000e+00, 8.98000000e-01, 4.68317403e-01],
       [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
        0.00000000e+00, 8.84000000e-01, 1.37743803e-03],
       ...,
       [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
        0.00000000e+00, 7.92000000e-01, 1.78351363e-04],
       [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
        0.00000000e+00, 7.44000000e-01, 1.67258779e-04],
       [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
        0.00000000e+00, 7.82000000e-01, 2.45776879e-05]])
```

Model Building

In [61]:
```
from sklearn import neighbors
```

In [62]:
```
model = neighbors.NearestNeighbors(n_neighbors=5, algorithm = 'ball_tree',
                                   metric = 'euclidean')
model.fit(features_scaled)
dist, idlist = model.kneighbors(features_scaled)
```

In [63]:
```
df['title'].value_counts()
```

Out[63]:
```
The Iliad                                               9
The Brothers Karamazov                                  9
The Odyssey                                             8
Anna Karenina                                           8
'Salem's Lot                                            8
                                                       ..
The Noonday Demon: An Atlas of Depression               1
The Noonday Demon: An Anatomy of Depression             1
My Secret: A PostSecret Book                            1
The Secret Lives of Men and Women: A PostSecret Book    1
Las aventuras de Tom Sawyer                             1
Name: title, Length: 10348, dtype: int64
```

In [64]:
```
@interact
def BookRecomender(book_name = list(df['title'].value_counts().index)):
    book_list_name = []
    book_id = df[df['title'] == book_name].index
    book_id = book_id[0]
    for newid in idlist[book_id]:
        book_list_name.append(df.iloc[newid].title)
    return book_list_name
```

book_name: The Iliad

```
['The Iliad',
 'The Call of the Wild',
 "She's Come Undone",
 'The Fountainhead',
 'Beloved']
```

Now we have come to the end of this project