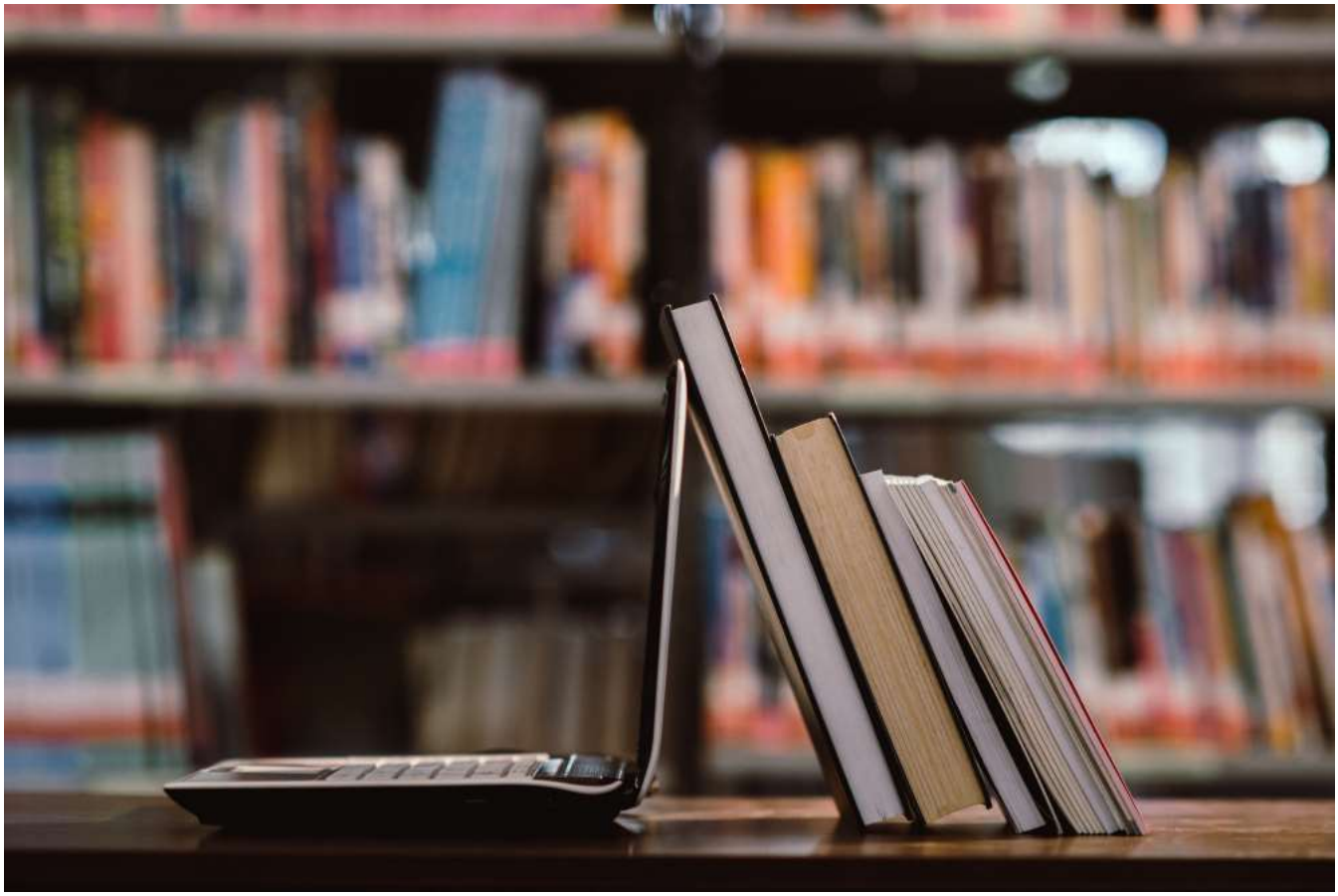
BOOK RECOMMENDATION SYSTEM USING K-NEAREST NEIGHBOUR

AUTHOR : SYED SHAAHID ¶



In [ ]:

```
dataset link ="https://www.kaggle.com/datasets/jealousleopard/goodreadsbooks"
Githublink = "https://github.com/syedshaahid02/syed"
```

In [ ]:

```
# Importing libraries
```

In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
## for interactive plots
import ipywidgets
from ipywidgets import interact
from ipywidgets import interact_manual
## For Ignoring Warning ErrorMessage
from warnings import filterwarnings
filterwarnings('ignore')
```

In [2]:

```
# Importing Dataset
```

In [3]:

```
df =pd.read_csv("books.csv",error_bad_lines = False)
```

```
b'Skipping line 3350: expected 12 fields, saw 13\nSkipping line 4704: expected 12 fields, saw 13\nSkipping line 5879: expec
ted 12 fields, saw 13\nSkipping line 8981: expected 12 fields, saw 13\n'
```

In [4]:

```
df.head()
```

Out[4]:

	bookID	title	authors	average_rating	isbn	isbn13	language_code	num_pages	ratings_count	text_reviews_count	publication
0	1	Harry Potter and the Half-Blood Prince (Harry ...	J.K. Rowling/Mary GrandPré	4.57	0439785960	9780439785969	eng	652	2095690	27591	9/16
1	2	Harry Potter and the Order of the Phoenix (Har...	J.K. Rowling/Mary GrandPré	4.49	0439358078	9780439358071	eng	870	2153167	29221	9/1
2	4	Harry Potter and the Chamber of Secrets (Harry...	J.K. Rowling	4.42	0439554896	9780439554893	eng	352	6333	244	11/1
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
4	8	Harry Potter Boxed Set Books 1-5 (Harry Potte...	J.K. Rowling/Mary GrandPré	4.78	0439682584	9780439682589	eng	2690	41428	164	9/13

In [5]:

```
df.columns
```

Out[5]:

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

In [6]:

```
df.shape
```

Out[6]:

```
(11123, 12)
```

In [7]:

```
df.columns=df.columns.str.strip()
```

In [8]:

```
df.dtypes
```

Out[8]:

```
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 [9]:

```
df.describe()
```

Out[9]:

	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 [10]:

```
# checking statistical summary of all categorical columns
```

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	000617695X	eng	10/1/2005	Vintage
freq	9	40	1	8908	56	318

In [12]:

```
# checking sum of all null values
```

In [13]:

```
df.isnull().sum()
```

Out[13]:

```
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 [14]:

```
# check if any duplicate row is present
```

In [15]:

```
df.duplicated().any()
```

Out[15]:

```
False
```

In [16]:

```
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 exsisting onesw

In [17]:

```
#droping unnecessary column present in database
```

In [18]:

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

In [19]:

```
df.publication_date
```

Out[19]:

```
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 [20]:

```
## Creating new year columns
```

In [21]:

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

In [22]:

```
df.head()
```

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]:

```
## Changing datatype of year columns from object to integer
```

In [25]:

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

In [26]:

```
df.dtypes
```

Out[26]:

```
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 [27]:

```
df.columns
```

Out[27]:

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

In [28]:

```
# Exploratory Data Analysis
```

In [ ]:

```
## Filter year== 2022 and get required output based on input
```

In [29]:

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

Out[29]:

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

In [ ]:

```
##Filter Year == 2018 and get required output based on input.
```

In [30]:

```
df[df['year'] == 2018][['title', 'authors','average_rating','language_code','publisher' ]]
```

Out[30]:

	title	authors	average_rating	language_code	publisher
3171	Ariel: The Restored Edition	Sylvia Plath/Frieda Hughes	4.27	eng	Harper Perennial Modern Classics
4080	El Perfume: Historia De Un Asesino	Patrick Süskind	4.02	spa	Planeta Publishing
4082	The Perfume Factory	Alex Austin	4.18	eng	Kindle
8068	El diablo de la botella	Robert Louis Stevenson/Diana Castellanos/Eleon...	3.74	spa	Grupo Editorial Norma S.A.
11085	El alquimista: una fábula para seguir tus sueños	Paulo Coelho/Juan Godó Costa	3.86	eng	Rayo

In [31]:

```
##Creating Groupby function based on Year and Title column
```

In [32]:

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

Out[32]:

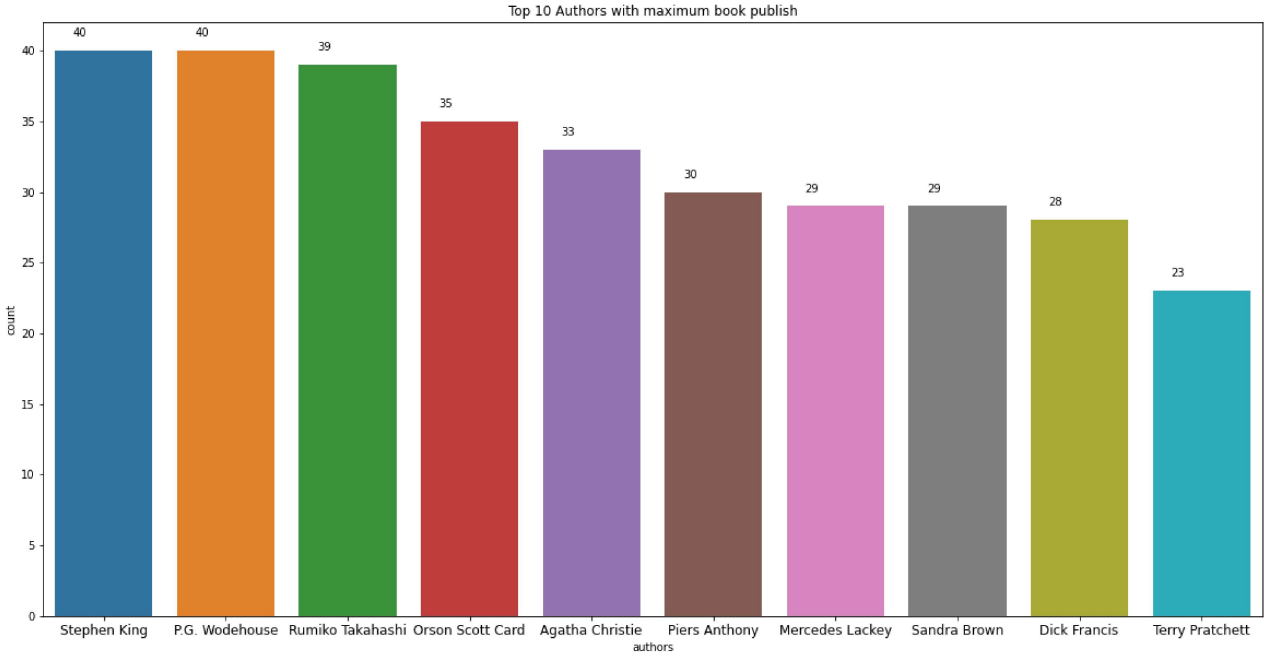
```
year
2006    1700
2005    1260
2004    1069
2003     931
2002     798
2001     656
2000     534
2007     518
1999     450
1998     396
1997     290
1996     250
1995     249
1994     220
1992     183
1993     165
1991     151
1989     118
1990     117
1987      88
Name: title, dtype: int64
```

In [ ]:

```
## Ploting Counplot graph for "Top 10 Authors with Maximum book Publish"
```

In [33]:

```
plt.figure(figsize = (20, 10))
ax = 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)
for p in ax.patches:
 ax.annotate(format(p.get_height()), (p.get_x()+0.15, p.get_height()+1))
plt.show()
```



In [ ]:

```
# Sort All Value Count of language_code.
```

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
lat         3
enm         3
rus         2
swe         2
nor         1
tur         1
ale         1
srp         1
nl          1
msa         1
glg         1
ara         1
gla         1
wel         1
Name: language_code, dtype: int64
```

In [ ]:

```
##Creating Groupby Function base on language_code Column and getting Required Output.
```

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]:

```
##Checking Top 20 Value Count of Title Column.
```

In [37]:

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

Out[37]:

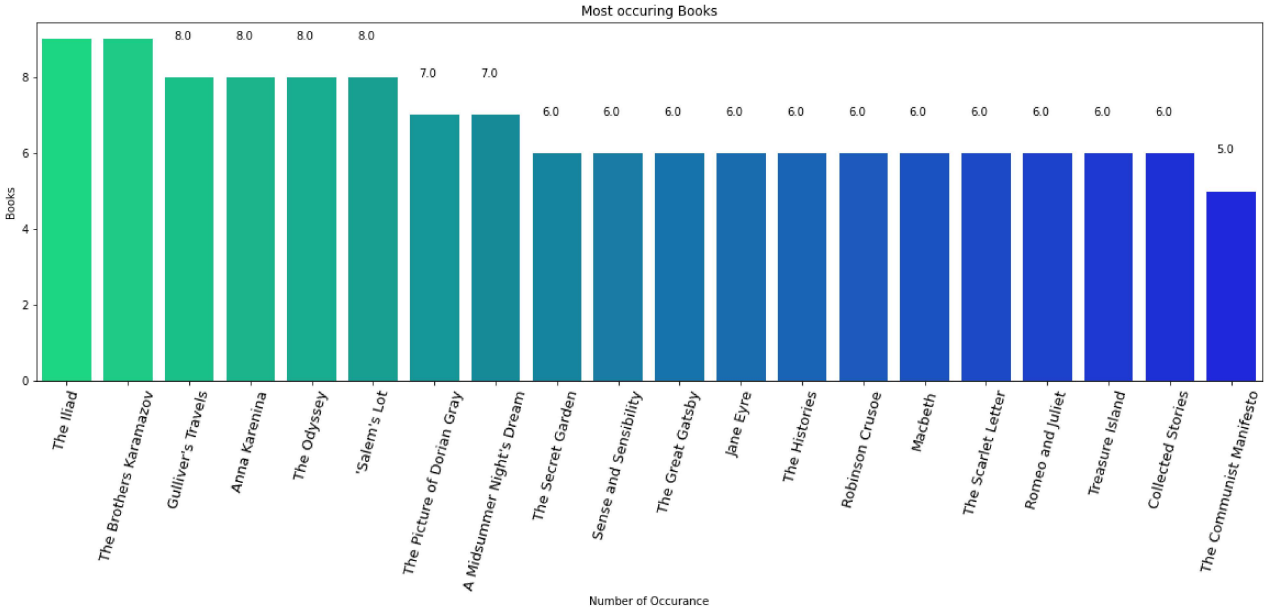
```
The Iliad                      9
The Brothers Karamazov         9
Gulliver's Travels             8
Anna Karenina                  8
The Odyssey                    8
'Salem's Lot                   8
The Picture of Dorian Gray     7
A Midsummer Night's Dream      7
The Secret Garden              6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

In [ ]:

```
#Plotting Barplot to find most occuring book in our data.
```

In [38]:

```
### Plotting BarPLot to find most occuring book in our data
plt.figure(figsize = (20, 6))
book = df['title'].value_counts()[:20]
ax = sns.barplot(x = book.index, y = book,
 palette = 'winter_r')
plt.title("Most occuring Books")
plt.xlabel("Number of Occurance")
plt.ylabel("Books")
plt.xticks(rotation = 75, fontsize = 13)
for p in ax.patches:
 ax.annotate(format(p.get_height()), (p.get_x()+0.15, p.get_height()+1))
plt.show()
```

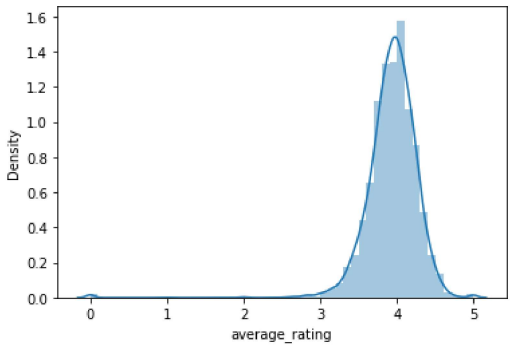


In [ ]:

```
#Ploting Distribution Graph on Average_Rating.m
```

In [39]:

```
### Ploting Distribution Graph on Average_Rating.
sns.distplot(df['average_rating'])
plt.show()
```



In [40]:

```
#Sorting Dataset related with maximum Average_Rating Column
```

In [41]:

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

Out[41]:

	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 [ ]:

```
#Checking Top 20 Publisher in Dataset
```

In [42]:

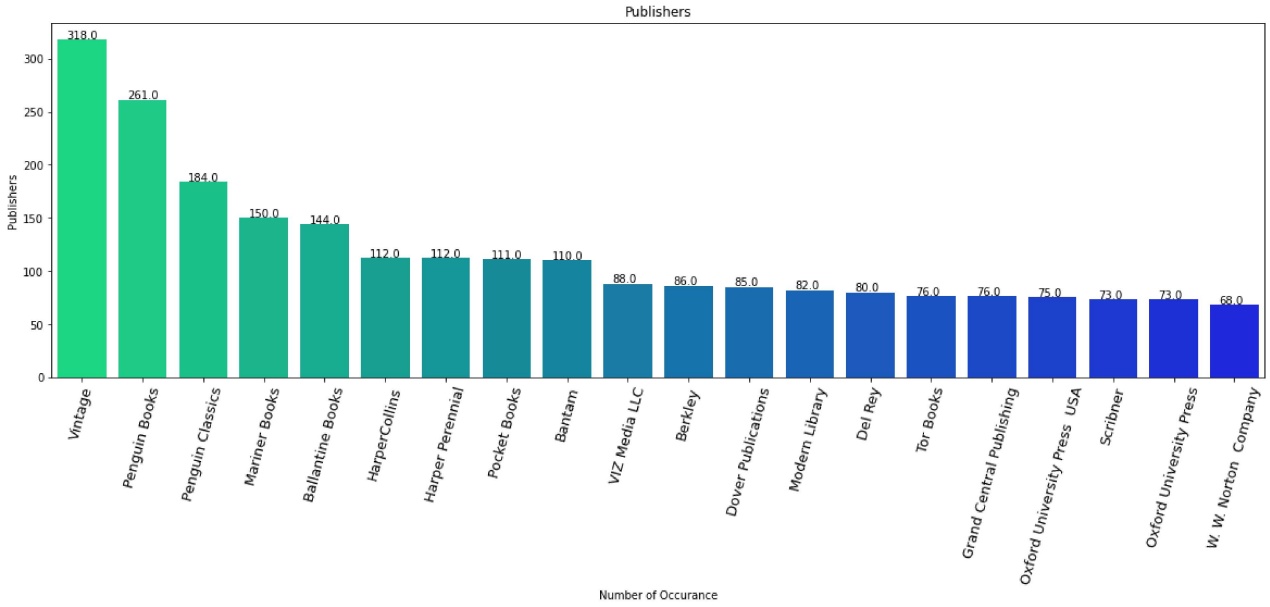
```
publisher = df['publisher'].value_counts()[:20]
publisher
```

Out[42]:

```
Vintage                        318
Penguin Books                  261
Penguin Classics               184
Mariner Books                  150
Ballantine Books               144
HarperCollins                  112
Harper Perennial               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
Scribner                        73
Oxford University Press         73
W. W. Norton  Company           68
Name: publisher, dtype: int64
```

In [43]:

```
## Plotting Barplot for Top 20 Publishers
plt.figure(figsize = (20, 6))
publisher = df['publisher'].value_counts()[:20]
ax = sns.barplot(x = publisher.index, y = publisher, palette = 'winter_r')
plt.title("Publishers")
plt.xlabel("Number of Occurance")
plt.ylabel("Publishers")
plt.xticks(rotation = 75, fontsize = 13)
for p in ax.patches:
 ax.annotate(format(p.get_height()), (p.get_x()+0.15, p.get_height()+1))
plt.show()
```



In [ ]:

```
Now Book Recommendation System Implementation
Recommending Books based on Publishers
Recommending Books based on Authors
Recommending Books based on Language
```

In [ ]:

```
#Recommending Books based on Publishers
```

In [44]:

```
## Checking Total Value count of all Top Publishers
df.publisher.value_counts()
```

Out[44]:

```
Vintage                      318
Penguin Books                261
Penguin Classics             184
Mariner Books                150
Ballantine Books             144
                            ...
Pearson Longman                1
Pocket Books/Star Trek         1
Random House Puzzles & Games   1
Quinteto                       1
DAW Fantasy                    1
Name: publisher, Length: 2290, dtype: int64
```

In [45]:

```
#Defining Function for Recommending Books based on Publishers
```

In [46]:

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

In [47]:

```
### Checking Function of Recommending Books based on Publishers name "Vintage"
recomd_books_publisheres('Vintage')
```

Out[47]:

	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 [48]:

```
### Checking Function of Recommending Books based on Publishers name "Penguin Books"
recomd_books_publisheres('Penguin Books')
```

Out[48]:

	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

In [ ]:

```
##Creating Interactive Plotting using ipywidgets
```

In [49]:

```
@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

In [50]:

```
df.columns
```

Out[50]:

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

In [ ]:

```
#Recommending Books based on Authors
```

In [51]:

```
### Creating Interactive Plotting using ipywidgets for Recommending Books based on Authors
@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… Stephen King

	title	average_rating
1575	Carrie / 'Salem's Lot / The Shining	4.54
3150	The Green Mile	4.44
9863	Different Seasons	4.35
8650	Different Seasons	4.35
3155	On Writing: A Memoir	4.33
6016	The Dark Tower (The Dark Tower #7)	4.28
1459	The Drawing of the Three (The Dark Tower #2)	4.23
3488	The Shining	4.22
3486	The Shining	4.22
3156	The Shining	4.22

In [ ]:

```
#Recommending Books based on Language
```

In [52]:

```
### Creating Interactive Plotting using ipywidgets for Recommending Books based on Language
@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 [53]:

```
### Checking Top 2 Rows
df.head(2)
```

Out[53]:

	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

In [ ]:

```
#Creating Function for Converting Number to Object on Average_Rating Column.
```

In [56]:

```
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 [57]:

```
### Now Checking Total Value for each Converted Objects.
df['rating_obj'].value_counts()
```

Out[57]:

```
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 [58]:

```
## Creating One-Hot Encoding on Rating columns
rating_df = pd.get_dummies(df['rating_obj'])
rating_df.head()
```

Out[58]:

	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 [59]:

```
## Creating One-Hot Encoding on language_code columns
language_df = pd.get_dummies(df['language_code'])
language_df.head()
```

Out[59]:

	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 [60]:

```
### Performing Concatenation Function for Joining two Dataframe
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[60]:

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	ratings_cou
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	20956
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	21531
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	63
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	23395
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	414

5 rows × 34 columns

In [ ]:

```
## Feature Scaling
```

In [61]:

```
### Importing MinMax SCaler on dataset
from sklearn.preprocessing import MinMaxScaler
```

In [62]:

```
scaler = MinMaxScaler()
features_scaled = scaler.fit_transform(features)
```

In [63]:

```
features_scaled
```

Out[63]:

```
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 [64]:

```
### IMporting Neighbors Library.
from sklearn import neighbors
```

In [65]:

```
### Model Fitting
model = neighbors.NearestNeighbors(n_neighbors=5, algorithm = 'ball_tree',
 metric = 'euclidean')
model.fit(features_scaled)
dist, idlist = model.kneighbors(features_scaled)
```

In [66]:

```
df['title'].value_counts()
```

Out[66]:

```
The Iliad                                          9
The Brothers Karamazov                             9
Gulliver's Travels                                 8
Anna Karenina                                      8
The Odyssey                                        8
                                                  ..
Time To Be In Earnest: A Fragment Of Autobiography 1
Tempest (Star Wars: Legacy of the Force  #3)       1
The Ravenscar Dynasty (Ravenscar  #1)              1
Autobiography of Thomas Jefferson                  1
Secret Identity (Lost  #2)                         1
Name: title, Length: 10348, dtype: int64
```

In [68]:

```
### Creating Book Recommendation System while using Book Title through Interactive Plotting Library.
@interact
def BookRecomender(book_name = list(df['title'].value_counts().index)):
    book_list_name = []
    book_id = df[df['title'] == book_name]
    book_id = book_id.index[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']
```