This pdf was made using docker

Youssef Salah Mostafa

22010442

Intelligent Systems

Department

Level 2

Cloud

Computing-Assignment 2

This assignment was a data analytics assignment, our task was to analyze the "Harry Potter Series" in the "Popular Books" dataset, these steps include : Data Cleaning, Data Preprocessing and Data Analysis.
First, we have to load the data, to do so we downloaded the dataset from kaggle and used pandas to import it.

```
df = pd.read_csv('./books.csv')
df.head()
```

✓ 0.1s Python

	book_id	goodreads_book_id	best_book_id	work_id	books_count	isbn	isbn13	authors	original_publication_year	original_title	...	ratings_count	work_ratings_count	work_text_reviev
0	1	2767052	2767052	2792775	272	439023483	9.780439e+12	Suzanne Collins	2008.0	The Hunger Games	...	4780653	4942365	
1	2	3	3	4640799	491	439554934	9.780440e+12	J.K. Rowling, Mary GrandPré	1997.0	Harry Potter and the Philosopher's Stone	...	4602479	4800065	
2	3	41865	41865	3212258	226	316015849	9.780316e+12	Stephenie Meyer	2005.0	Twilight	...	3866839	3916824	
3	6	11870085	11870085	16827462	226	525478817	9.780525e+12	John Green	2012.0	The Fault in Our Stars	...	2346404	2478609	
4	12	13335037	13335037	13155899	210	62024035	9.780062e+12	Veronica Roth	2011.0	Divergent	...	1903563	2216814	

5 rows × 23 columns

Then we check the column information, for non-null counts and data types,.

```
df.info()
```

✓ 0.0s Python

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1354 entries, 0 to 1353
Data columns (total 23 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   book_id                    1354 non-null   int64
 1   goodreads_book_id          1354 non-null   int64
 2   best_book_id               1354 non-null   int64
 3   work_id                    1354 non-null   int64
 4   books_count                1354 non-null   int64
 5   isbn                       1302 non-null   object
 6   isbn13                     1310 non-null   float64
 7   authors                    1354 non-null   object
 8   original_publication_year  1351 non-null   float64
 9   original_title             1302 non-null   object
 10  title                      1354 non-null   object
 11  language_code              1245 non-null   object
 12  average_rating             1354 non-null   float64
 13  ratings_count              1354 non-null   int64
 14  work_ratings_count         1354 non-null   int64
 15  work_text_reviews_count    1354 non-null   int64
 16  ratings_1                  1354 non-null   int64
 17  ratings_2                  1354 non-null   int64
 18  ratings_3                  1354 non-null   int64
 19  ratings_4                  1354 non-null   int64
...
 21  image_url                  1354 non-null   object
 22  small_image_url            1354 non-null   object
dtypes: float64(3), int64(13), object(7)
memory usage: 243.4+ KB
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

Then we check the null count in each column.

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

✓ 0.0s — Python

```
book_id                       0
goodreads_book_id             0
best_book_id                  0
work_id                       0
books_count                   0
isbn                         52
isbn13                       44
authors                       0
original_publication_year     3
original_title               52
title                         0
language_code               109
average_rating                0
ratings_count                 0
work_ratings_count            0
work_text_reviews_count       0
ratings_1                     0
ratings_2                     0
ratings_3                     0
ratings_4                     0
ratings_5                     0
image_url                     0
small_image_url               0
dtype: int64
```

Then we drop the nulls, and check the duplicates count, which is zero.

```
df.dropna(inplace=True)
df.duplicated().sum()
```

✓ 0.0s — Python

```
0
```

Then we reduce the dataframe into a new one, where the reduction condition is that the book title contains "Harry Potter".

```
condition = df['original_title'].str.contains('Harry Potter')
rdf = df[condition]
rdf.head()
```

✓ 0.0s — Python

	book_id	goodreads_book_id	best_book_id	work_id	books_count	isbn	isbn13	authors	original_publication_year	original_title	...	ratings_count	work_ratings_count	work_text_revie
1	2	3	3	4640799	491	439554934	9.780440e+12	J.K. Rowling, Mary GrandPré	1997.0	Harry Potter and the Philosopher's Stone	...	4602479	4800065	
6	18	5	5	2402163	376	043965548X	9.780440e+12	J.K. Rowling, Mary GrandPré, Rufus Beck	1999.0	Harry Potter and the Prisoner of Azkaban	...	1832823	1969375	
8	21	2	2	2809203	307	439358078	9.780439e+12	J.K. Rowling, Mary GrandPré	2003.0	Harry Potter and the Order of the Phoenix	...	1735368	1840548	
9	23	15881	15881	6231171	398	439064864	9.780439e+12	J.K. Rowling, Mary GrandPré	1998.0	Harry Potter and the Chamber of Secrets	...	1779331	1906199	
10	24	6	6	3046572	332	439139600	9.780439e+12	J.K. Rowling, Mary GrandPré	2000.0	Harry Potter and the Goblet of Fire	...	1753043	1868642	

5 rows × 23 columns

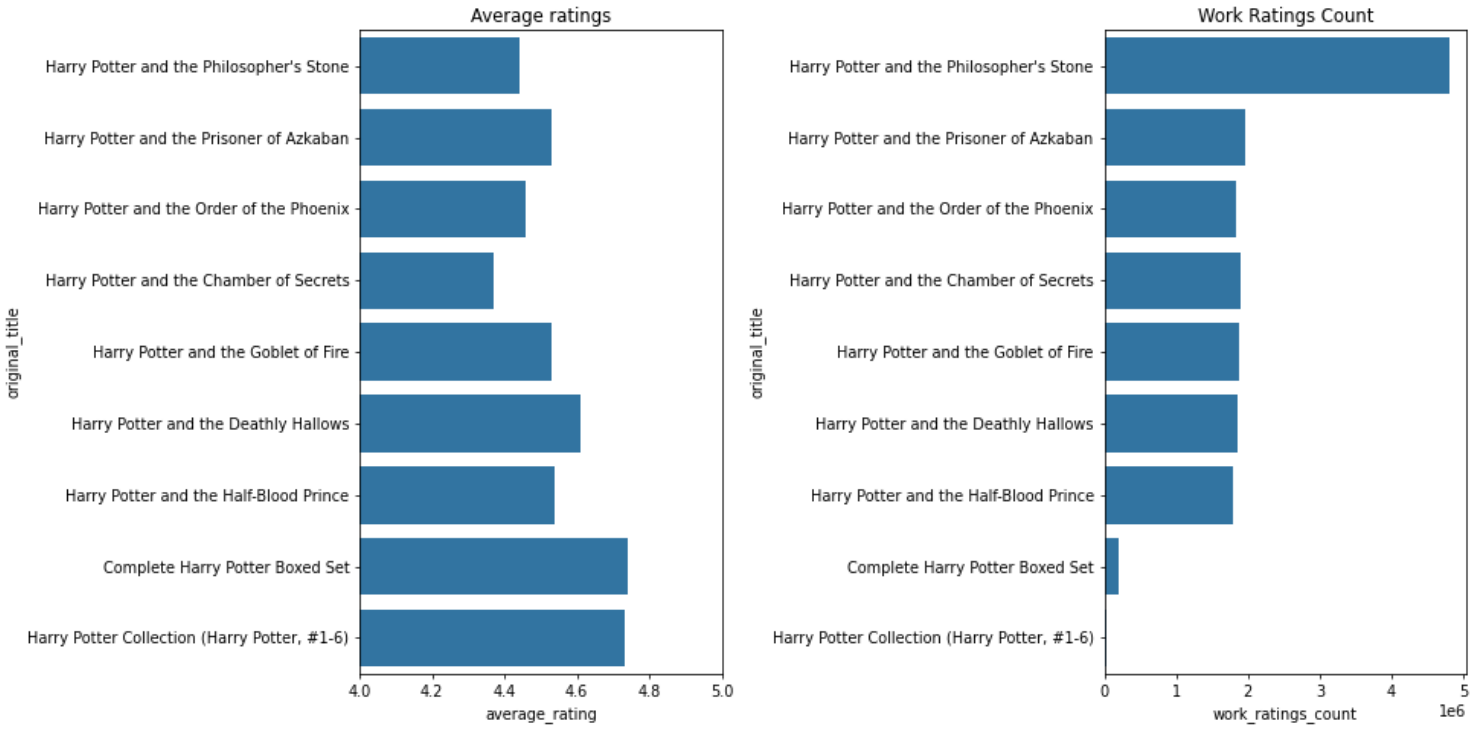
Then we drop the columns that contain categorical data, leaving behind the numeric data, and the book title.

```
rdf = rdf.drop([
    'book_id','best_book_id', 'goodreads_book_id','work_id','isbn','isbn13','authors',
    'title','language_code','image_url','small_image_url'
    ],axis=1).drop(1036)
rdf.head()
```

✓ 0.0s — Python

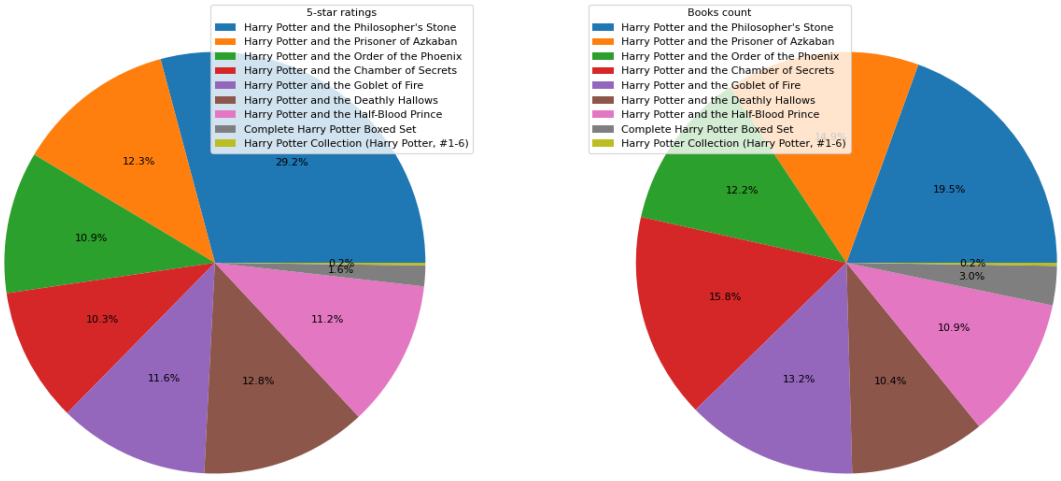
	books_count	original_publication_year	original_title	average_rating	ratings_count	work_ratings_count	work_text_reviews_count	ratings_1	ratings_2	ratings_3	ratings_4	ratings_5
1	491	1997.0	Harry Potter and the Philosopher's Stone	4.44	4602479	4800065	75867	75504	101676	455024	1156318	3011543
6	376	1999.0	Harry Potter and the Prisoner of Azkaban	4.53	1832823	1969375	36099	6716	20413	166129	509447	1266670
8	307	2003.0	Harry Potter and the Order of the Phoenix	4.46	1735368	1840548	28685	9528	31577	180210	494427	1124806
9	398	1998.0	Harry Potter and the Chamber of Secrets	4.37	1779331	1906199	34172	8253	42251	242345	548266	1065084
10	332	2000.0	Harry Potter and the Goblet of Fire	4.53	1753043	1868642	31084	6676	20210	151785	494926	1195045

Then we plot the average rating for each book and the review count for each book.



We can see that the highest rated book is "Complete Harry Potter Boxed Set", and the one with the most reviews is "Harry Potter and the Philosopher's Stone".

Then we plot the total 5 star reviews for all books and the number of copies for each book, the book with the most copies would be the best seller.



We can see that the one with the most 5 star reviews is "Harry Potter and the Philosopher's Stone", it is also the one with the most copies made.

Now we compute the descriptive statisitcs for the data and save them into a new dataframe.

```
sdf = rdf.describe()
sdf
```

0.1s Python

	books_count	original_publication_year	average_rating	ratings_count	work_ratings_count	work_text_reviews_count	ratings_1	ratings_2	ratings_3	ratings_4	ratings_5
count	9.000000	9.000000	9.0000	9.000000e+00	9.000000e+00	9.000000	9.000000	9.000000	9.000000	9.000000e+00	9.000000e+00
mean	280.444444	2001.333333	4.5500	1.704790e+06	1.805367e+06	32528.777778	13850.666667	29039.888889	161493.111111	4.534314e+05	1.147551e+06
std	153.337138	3.708099	0.1253	1.301040e+06	1.357471e+06	22335.707549	23359.121891	30238.686638	134964.423859	3.345266e+05	8.488109e+05
min	6.000000	1997.000000	4.3700	2.461800e+04	2.627400e+04	882.000000	203.000000	186.000000	946.000000	3.891000e+03	2.104800e+04
25%	263.000000	1998.000000	4.4600	1.678823e+06	1.785676e+06	27520.000000	6676.000000	20210.000000	113646.000000	3.839140e+05	1.065084e+06
50%	307.000000	2000.000000	4.5300	1.746574e+06	1.847395e+06	31084.000000	7308.000000	21516.000000	151785.000000	4.944270e+05	1.161491e+06
75%	376.000000	2005.000000	4.6100	1.779331e+06	1.906199e+06	36099.000000	9363.000000	31577.000000	180210.000000	5.094470e+05	1.266670e+06
max	491.000000	2007.000000	4.7400	4.602479e+06	4.800065e+06	75867.000000	75504.000000	101676.000000	455024.000000	1.156318e+06	3.011543e+06

We can see that the average rating for the series is 4.55.

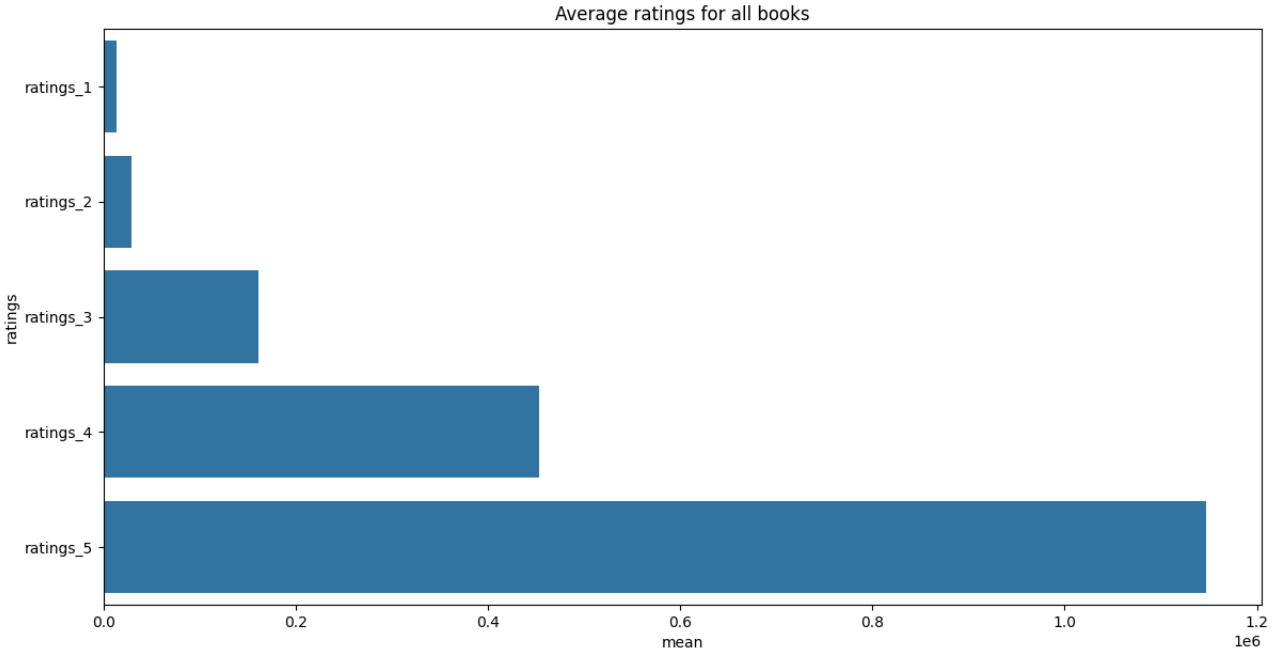
Next we reduce the statistics to just the mean and the number star reviews, through which we would determine the average star reviews for the series.

```
rsdf = sdf.drop(['original_publication_year','average_rating','ratings_count','work_ratings_count','work_text_reviews_count','books_count'],axis=1).transpose()
ratings = list(rsdf.index)
mean = list(rsdf['mean'])
pltdf = pd.DataFrame({'mean':mean,'ratings':ratings})
pltdf
```

0.0s Python

	mean	ratings
0	1.385067e+04	ratings_1
1	2.903989e+04	ratings_2
2	1.614931e+05	ratings_3
3	4.534314e+05	ratings_4
4	1.147551e+06	ratings_5

Now we plot the reduce statistics into a bar plot.



We can see that by far, the most reviews gained were the 5-star reviews, followed second by the 4-star reviews.