Dokerfile and Data Analysis with Popular Books Dataset

Cloud Computing – Assignment 2

Amany Fathy Mohamed Yaseen

2022565968

AI

Code Snapshots : Results of Analysis

- After cleaning and preprocessing the data (handle missing values and check duplicates), I have filtered the data so I can extract the data related to Harry Potter to apply the analysis.

Checking Missing Values

```
In [5]: # to calculate the total null values for each column
        total_null_b = books.isna().sum().sort_values(ascending=False)
        total_null_b.head(10)

Out[5]: language_code          109
        isbn                    52
        original_title          52
        isbn13                  44
        original_publication_year    3
        title                   0
        goodreads_book_id       0
        best_book_id            0
        work_id                 0
        books_count             0
        dtype: int64
```

Checking duplicates

```
In [6]: #check duplicates
        books.duplicated().sum()

Out[6]: 0
```

Total Number of Missing Values

```
In [7]: # total number of null values
        before_handle = books.isna().sum().sum()
        before_handle

Out[7]: 260
```

Handle Missing Values

```
In [8]: #determine the columns to handle and fill with the most frequent value
        #mode().iloc[0] -> mode is to determine the most frequent value,
        #iloc[0] to select the first mode in case there are multiple modes

        handle_columns = ['language_code', 'isbn', 'original_title', 'isbn13', 'original_publication_year']
        books[handle_columns] = books[handle_columns].fillna(books[handle_columns].mode().iloc[0])

        total_null_af = books.isna().sum().sort_values(ascending=False)
        total_null_af.head(10)

Out[8]: small_image_url          0
        title                    0
        goodreads_book_id        0
        best_book_id             0
        work_id                  0
        books_count              0
        isbn                     0
        isbn13                   0
        authors                  0
        original_publication_year    0
        dtype: int64
```

```
In [9]: # check the total number of null values after handling them
        after_handle = books.isna().sum().sum()
        after_handle

Out[9]: 0
```

Check Number of Missing Values after handling them

Since our dataset is small and the missing values in the columns are of small proportion, So I decided to fill the missing values with the most frequent value in each column (mode) to avoid losing data or information.

Harry Potter Data

Since the 'original_title' column was containing null values, I chose to select the rows of Harry Potter based on the 'title' column.

```
In [10]: #To extract the data of Harry Potter books
         hp_books = books['title'].str.contains('Harry Potter', case=False) # Check that each title contains the substring 'Harry Potter'
         harry_potter = books.loc[hp_books].reset_index() # Use the output boolean series (hp_books) to select the rows of condition True

         display(harry_potter)
         print(harry_potter.shape)
```

	index	book_id	goodreads_book_id	best_book_id	work_id	books_count	isbn	isbn13	authors	original_publication_year	...	ratings_coun
0	1	2	3	3	4640799	491	439554934	9.780440e+12	J.K. Rowling, Mary GrandPré	1997.0	...	460247

Shape of the data : 11 rows × 24 columns

hp_books is a boolean series, where True is for the title containing the substring 'Harry Potter'.

Case = False, to not be case sensitive.

Check the consistency of 'original_title' column

Since the 'original title' column contained missing values before, which I filled with the most frequent value, I will check the column to see if it has a title unrelated to Harry Potter.

If there is a title unrelated to Harry Potter, I would rather refill it with (Harry Potter).

I observed an unrelated title at index 10, where a boolean False indicates that it doesn't satisfy the condition.

```
In [19]: # I observed that at index 10, the output boolean is False which indicates that the title at this index
         # is unrelated to Harry Potter
         # So I will fill it with "Harry Potter"

         harry_potter['original_title'].str.contains('Harry Potter', case=False)

Out[19]: 0      True
         1      True
         2      True
         3      True
         4      True
         5      True
         6      True
         7      True
         8      True
         9      True
         10    False
         Name: original_title, dtype: bool
```

Print the unrelated title :

```
In [20]: # Pass the output boolean series to print the title which is unrelated to Harry Potter
         harry_potter.iloc[10]['original_title']

Out[20]: 'Dark Reunion'
```

Handle the unrelated title :

I have iterated over the cells of the Harry Potter dataframe and applied the check criteria. If the title doesn't contain the substring 'Harry Potter', I will refill the values at this cell with the substring.

(.at) function is used to set a value at a specific cell in the dataframe, where 'i' is the row and 'original_title' is the column name.

In [21]:
```
# I will iterate over the harry potter dataframe and check if 'Harry Potter' string is a member in title.
# If it is not, I will fill this cell with 'Harry Potter' using (.at) function.
# Else, I will continue to check another title at another location.

for i,title in enumerate(harry_potter['original_title']):
    if 'Harry Potter' not in title:
        harry_potter.at[i,'original_title'] = "Harry Potter"
    else:
        continue
display(harry_potter['original_title'])
```

```
0              Harry Potter and the Philosopher's Stone
1              Harry Potter and the Prisoner of Azkaban
2             Harry Potter and the Order of the Phoenix
3               Harry Potter and the Chamber of Secrets
4                   Harry Potter and the Goblet of Fire
5                 Harry Potter and the Deathly Hallows
6                Harry Potter and the Half-Blood Prince
7                       Complete Harry Potter Boxed Set
8          Harry Potter Collection (Harry Potter, #1-6)
9     The Magical Worlds of Harry Potter: A Treasury...
10                                        Harry Potter
Name: original_title, dtype: object
```

Most Selling Books

I chose to calculate the books count which indicates how many editions of the book are released and sold.

In [22]:
```
# Group by the book title and ID and calculate the book count for each title
# where the book count indicates how many editions of the book were released and sold
# Sort values descendingly to know the books of the highest count

most_selling = harry_potter.groupby(['book_id', 'title'])['books_count'].sum().sort_values(ascending=False).reset_index()
print('The Top most selling books are : ')
most_selling.head(5) # To print the top 5
```

```
The Top most selling books are :
```

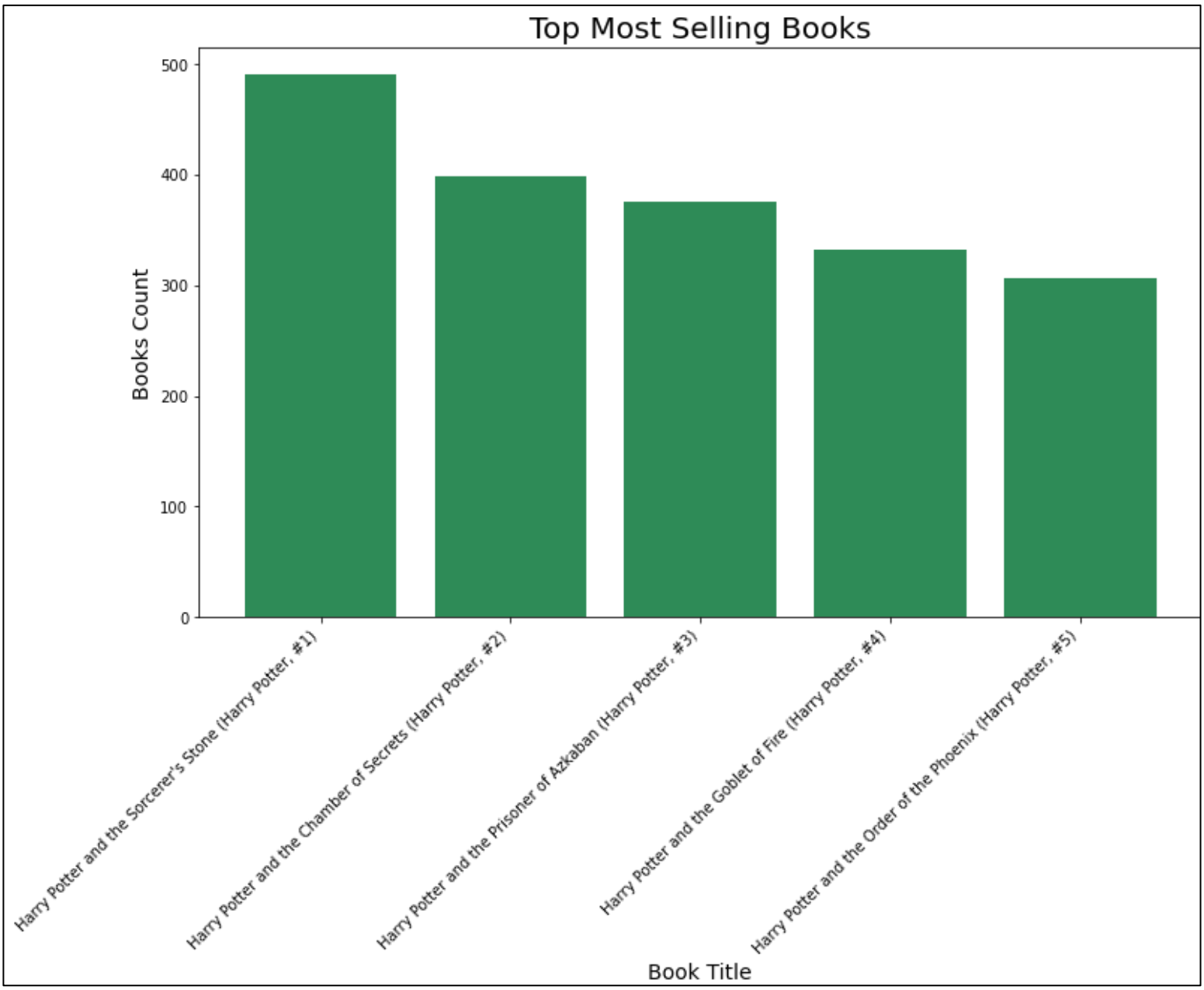
Out[22]:

	book_id	title	books_count
0	2	Harry Potter and the Sorcerer's Stone (Harry P...	491
1	23	Harry Potter and the Chamber of Secrets (Harry...	398
2	18	Harry Potter and the Prisoner of Azkaban (Harr...	376
3	24	Harry Potter and the Goblet of Fire (Harry Pot...	332
4	21	Harry Potter and the Order of the Phoenix (Har...	307

Barplot of the Top Sellling Books

In [23]:
```
# A bar chart plot shows the top selling books of harry potter.
plt.figure(figsize=(12, 7))
plt.bar(most_selling['title'].head(5), most_selling['books_count'].head(5), color='seagreen')
plt.title('Top Most Selling Books', size=20)
plt.xlabel('Book Title',size=14)
plt.ylabel('Books Count',size=14)
plt.xticks(rotation=45, ha='right')
plt.show()
```

The Barplot :



Average Rating of Harry Potter Books

- First approach, I calculated the mean of the 'average_rating' column to calculate the average rating of Harry Potter books regardless of the number of ratings for each book as all have equal weights.

```
In [25]: # Calculate the average rating of Harry Potter books regardless of the number of ratings for each book as all have equal weights
         avg_rate = harry_potter['average_rating'].mean()
         print(f'The average rating of Harry Potter books = {avg_rate}')

         The average rating of Harry Potter books = 4.482727272727273
```

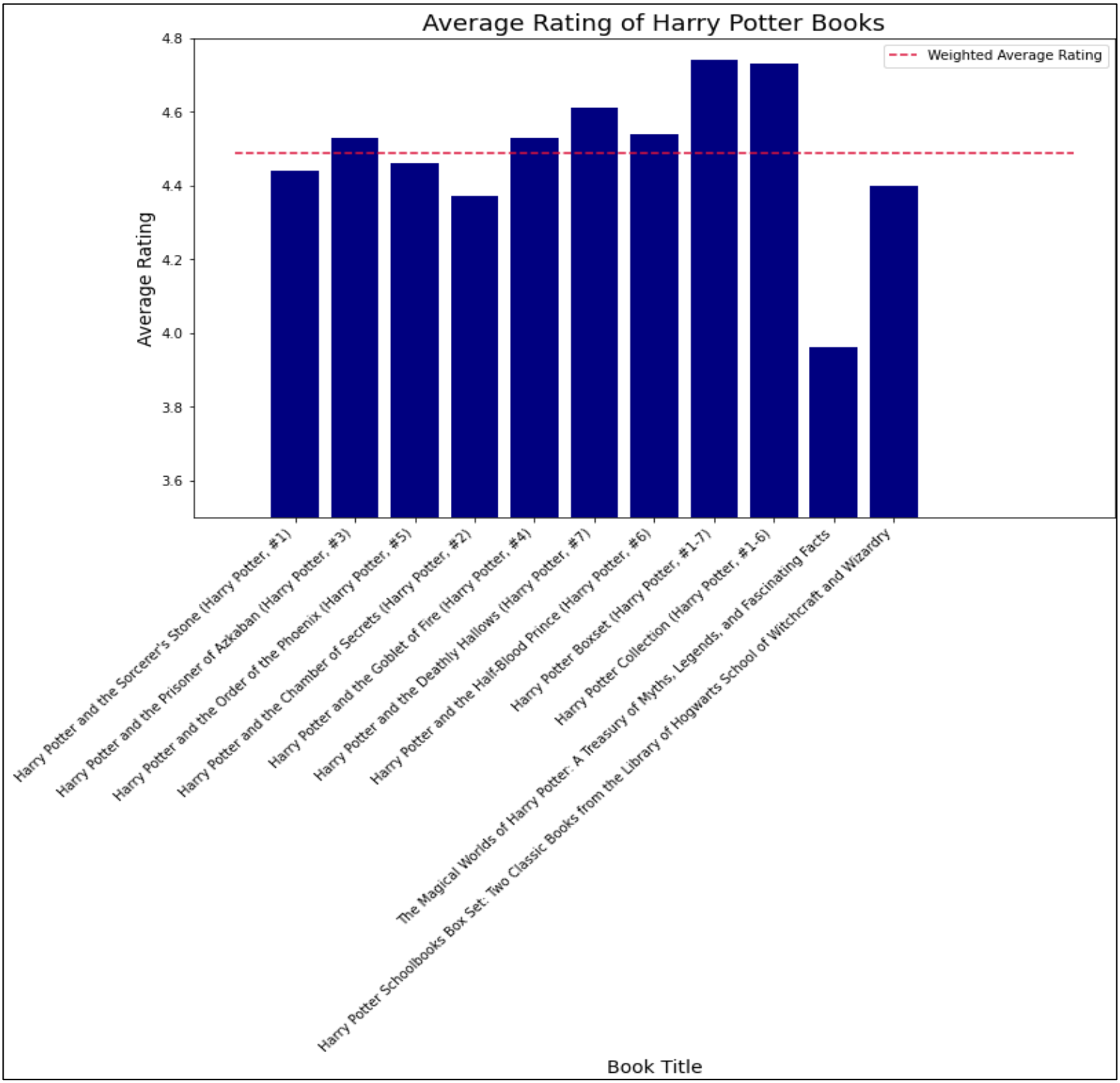
- Second approcah, I calculated the weighted average of the Harry Potter books where I took the ratings count for each book into account. So, the books that have the largest ratings count will have a large weight indicating their contribution.

```
In [26]: # Here, calculating the weighted average took the ratings count for each book into account.
         weighted_avgerage = (harry_potter['average_rating'] * harry_potter['ratings_count']).sum() / harry_potter['ratings_count'].sum()
         print(f'The weighted average rating of Harry Potter books = {weighted_avgerage}')

         The weighted average rating of Harry Potter books = 4.489114370335377
```

Barplot of Average Rating

```
In [27]: # A bar chart plot shows the average rating of the harry potter books.
         # The line represents the weighted average
         # hlines -> to add a horizontal line at a specific y value from xmin to xmax, where the y value I want to indicate
         # is the weighted average

         plt.figure(figsize=(12, 7))
         plt.bar(harry_potter['title'], harry_potter['average_rating'], color='navy')
         plt.hlines(weighted_avgerage, xmin=-1, xmax=13, color='crimson', linestyles='dashed',label='Weighted Average Rating')
         plt.ylim((3.5, 4.8)) # As I have observed the the minimum average rating value is 3.96, and maximum is 4.74
         plt.title('Average Rating of Harry Potter Books', size=18)
         plt.xlabel('Book Title',size=14)
         plt.ylabel('Average Rating',size=14)
         plt.xticks(rotation=45, ha='right') # To avoid the overlapping of titles
         plt.legend()
         plt.show()
```

The Barplot :



Observe minimum and maximum average ratings for plotting

```
In [24]: # Print the average rating of the books,to observe the minimum and maximum for plotting
         harry_potter['average_rating']

Out[24]: 0     4.44
         1     4.53
         2     4.46
         3     4.37
         4     4.53
         5     4.61
         6     4.54
         7     4.74
         8     4.73
         9     3.96
         10    4.40
         Name: average_rating, dtype: float64
```

The minimum average rating value is 3.96, and maximum is 4.74.

hlines function is used to add a horizontal line at a specific y value from xmin to xmax, where the y value I want to indicate is the weighted average.

I observed from the plot that the books Harry Potter and the Sorcerer's Stone, Harry Potter and the Order of the Phoenix, Harry Potter and the Chamber of Secrets, The Magical Worlds of Harry Potter, and the Harry Potter Schoolbooks Box Set have an average rating less than the weighted average rating of all books, which implies that these books have received lower ratings, considering the weight assigned to each book.