DATE: 04/24/23

STREAMING PLATFORM ANALYSIS BASED ON IMDB

by

Ruthwik Nadam

TABLE OF CONTENTS

1. OVERVIEW

A total of four OTT platforms datasets namely Netflix, Prime, Disney+ and Hulu was considered in this project. A new column named 'platform' in each of the Netflix, Prime, Disney+ and Hulu in each of the respective data set and then merged all of them.

After merging the 4 datasets and performing data cleaning and wrangling operations to create a merged and clean data set of all the movies and TV shows listed in 4 OTT platforms, the IMDB ratings, votes and other attributes were planned to retrieve from the IMDB python library for each of the movie or TV show.

IMDBpy library is a python package which can retrieve and manage data from the Internet movie Database (IMDB). We can search for movies, TV series, actors, actresses, directors etc. We can access various data such as movie titles, directors, actors/actresses, release year, runtime, ratings, etc. This will create a completely unique dataset which will have ratings for all the movies and TV shows listed in the 4 OTT platforms.

We have planned to perform the Exploratory Data Analysis (EDA) on this unique dataset. EDA was performed to find out whether there is a relationship between the runtime and IMDB rating of a movie, do movies with a higher release year tend to have higher IMDB ratings, Is there a correlation between the number of votes and IMDB rating of a movie, Are there any countries that produce more movies than others, Is there a relationship between the certificate rating and the IMDB rating of a movie, Are there any outliers in the dataset, such as movies with very low ratings despite having a high number of votes, Are there any patterns in when movies were added to the platform, such as a higher number of movies being added during certain months.

By analyzing the IMDB ratings, this EDA can help identify which movies are most popular among different users. This information can help streaming services personalize their content recommendations and improve user engagement. By analyzing which movies are most popular on OTT platforms, content providers can make more informed decisions about which movies to acquire and license for their own streaming services. Due to limited resources, we have faced call time issues because of which we have considered only Netflix movies for performing our EDA.

2. Dataset Preparation

The raw data was downloaded from the .csv files available in the dataset Netflix Movies and TV Shows from Kaggle [1]. The rest of the data sets were also downloaded from Kaggle [2][3][4].

The dataset was created by importing necessary libraries, reading data from multiple CSV files, merging them, performing data cleaning and data wrangling operations, and exporting the final data to a CSV file.

Initially, the required libraries such as Pandas and Numpy were imported using the 'import' method. Pandas is a popular data manipulation library for Python, and NumPy is a numerical computing library that supports multi-dimensional arrays and matrices. Then, all the raw csv dataset file were read into 4 different data frames with their respective names as Netflix, Prime, Disney and Hulu. A new column named *'platform'* was created in each of the data frames for reference.

The *'concat'* method from the panda's library was used to concatenate all the 4 data frames into a single data frame name *'combined_df'* [5]. The *'ignore_index=True'* argument was used to reset the index of the concatenated data frame. The merged data frame was exported into a .csv file with the filename *'combined_dataset.csv'* which is supposed to the raw data using the *'to*_csv' method [6].

The first objective in the data cleaning and data wrangling operation was performed to drop all the Nan values (missing) in the combined data frame using the *'drop.na()'* method [7]. Also, there were a few columns which seemed unnecessary for the EDA so, those columns *"show_id, cast"* and *"description"* were dropped from the data frame using the *'drop()'* method along with the *'axis=1'* argument which performs the operation on the columns instead of the indexes [8].

To avoid any errors in the search query, the leading or trailing whitespace from the *'title'* column were removed using the *'str.strip()'* method [9]. The cleaned dataset with all non-null columns was exported into a .csv file as '*combined_cleaned.csv*'. This cleaned dataset contained 6151 indexes and 10 columns.

The *IMDbPY* module is used to extract movie data from the IMDb database [10]. First, the *IMDbPY* module was imported and a *IMDbPY* object named '*ia'* was created. It is used to retrieve movie data from the IMDb database.

To check if the code was working, a test run with a statis condition on one movie title 'Jeans' was assigned to the variable *'title'*. This variable will be used to search for a movie by its title. The IMDB database was searched for movies with a title that matches the string stored in the 'title' variable using the *'search_movie'* method. Then the search results in *'title_search'* variable, which is a list of movie objects was stored.

Then retrieved the first movie object from the *'title_search'* list and stored it inside the *'title1'* variable. For the title obtained from the *'search_movie'* method, the IMDB ID was stored inside the variable *'ID_imdb'* using the *'getID()'* method. Using the *'get__movie'* method from the

IMDB python package, all the attributes such as its title, rating, genres, runtime, director etc., can be obtained. All the movie attributes of title *'Jeans'* was stored inside a variable *'movie_data'*.

To confirm which attributes can we get, all the keys inside the variable *'movie_data'* was printed. This confirmed that the variable in question can provide the IMDB rating for the searched movie. To test run it in on data frame, on a dynamic condition, a function *'IMDB_info'* was created. The function would take input a str *'title'* and a variable *'info_type'*, where the variable *'info_type'* would be any of the attributes of that movie such as *'ratings', 'votes', 'director'*, etc. The function would then search for the movie with the input title in the IMDB database and return the value which is requested as the second attribute in the function.

Due to limited resources, having faced run-time issues on the read time operations because of which we have considered a smaller data set, which is a Netflix movies dataset and the movies released in the year 2018 to 2021.

This time only the Netflix dataset from Kaggle was considered. The raw csv file *"netflix_titles.csv"* was read using the pandas *'read_csv()'* function and stored it in a pandas Data Frame named *'Netflix'* and printed the information about the *'netflix'* Data Frame, including the number of rows, number of columns, column names, data types, and non-null values in each column to figure out how many non-null values are present in the raw data.

Then dropped any rows in the *'Netflix'* Data Frame that contain missing values (Nan) and modified the Data Frame in place using the *'inplace'* parameter and checked if the columns *'show_id', 'cast'* and *'description'* are present in the *'Netflix'* Data Frame. If the condition is true, we dropped the columns 'show_id', 'cast', and 'description' from the 'Netflix' Data Frame along the column's (axis=1) and modifies the Data Frame in place using the *'inplace'* parameter.

A new Data Frame named *'netflix_filtered'* was created by selecting only the rows from the *'Netflix'* Data Frame where the *'release_year'* column is between 2018 and 2021 (inclusive). A *'copy()'* method is used to create a new copy of the Data Frame, rather than just a reference to the original Data Frame [11].

Then created a new Data Frame named *'netflix_movies'* by selecting only the rows from the *'netflix_filtered'* Data Frame where the 'type' column is equal to 'Movie'. Here also a *'copy()'* method is used to create a new copy of the Data Frame, rather than just a reference to the original Data Frame.

Then extracted the numeric value from each string in the 'duration' column of the *'netflix_movies'* Data Frame using the regular expression pattern '(\d+)' and converted it to an integer using the *'astype()'* method [12][13]. The str attribute is used to access the string methods of the 'duration' column and any leading or trailing whitespace from the strings in the 'title' column of the *'netflix_movies'* Data Frame are removed using the *'str.strip()'* method.

Lastly, sorted the rows of the *'netflix_movies'* Data Frame in ascending order based on the values in the *'title'* column using the *'sort_values()'* method and renamed the columns of the Data Frame *'netflix_movies'* using a dictionary that maps the old column names to the new column names. The '*inplace=True'* parameter is used to modify the Data Frame in place and printed the information about the Data Frame *'netflix_movies'*. The index of the Data Frame *'netflix_movies'* is reset to be sequential and drop the old index, using the *'reset_index()'* method.

Stored the data from Data Frame *'netflix_movies'* to an Excel file called *"Netflix.xlsx"* using the *'to_excel()'* method. The parameter *'index=False'* was used to indicate that the row index should not be included in the output and displayed the first five rows of the Data Frame *'netflix_movies'* using the *'head()'* method. This dataset was resulted in 8 columns and 1682 rows.

Due to the high wait time of certain search query, the code was crashing continuously giving the read time operation time out error. So, instead of working on the *'netflix_movies'* dataset, a list of data frames was created under which each data frame would have a maximum of 300 rows. These data frames would be run inside a for loop calling a function *'func'* which again calls a new function *'IMDB_info'* by using the *'apply()'* method to request an API call to search for the movie in the IMDB database and return the value of the second parameter asked in the apply method.

Inside the 'IMDB_info' a new global variable *'cached_results'* was defined as a type of dictionary to store the search values from the IMDB database only if the title is not present in the dictionary. This would decrease the computational time of the code when its run again from the beginning, as the dictionary *'cached_results'* would already consist of the title if it was searched before the code crashed due to the read time operation error.

Finally, from the IMDB database the IMDB rating, and votes were retrieved for each of the movie listed in all the smaller data frames inside the list. Each of these updated smaller data frames are stored inside excel sheets and all those excel files are later concatenated into the original data frame of all the movies listed in Netflix and are released in the year 2018 to 2021 along with their IMDB ratings and votes [14]. This updated data frame consisted of few Null values which were dropped resulting in a clean non-null dataset with 1631 rows and 10 columns. This data set is saved inside a .csv file *'Netlix_movies.csv'* dropping the index and *'Type'* column as they were not required. This .csv file can be later called to perform EDA and visualize the data.

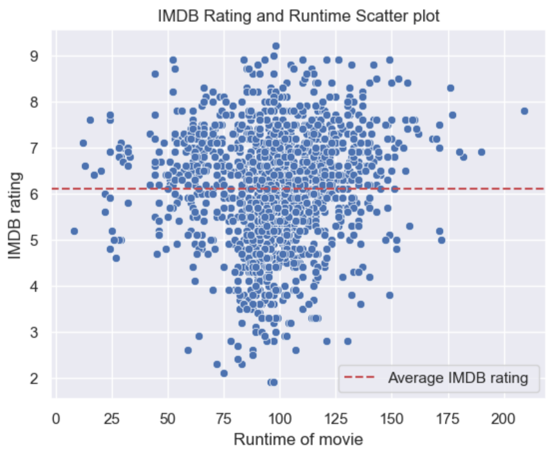
3. Exploratory Data Analysis (EDA)

1. Load the Dataset: We have loaded the Netflix dataset into a data frame using a suitable data analysis library, such as Pandas and Numpy in Python. The dataset contains 1631 entries with 10 columns. The columns are: *"Title", "Director", "Countries", "Date Added", "Release Year", "Certificate", "Runtime", "Genres", "IMDB_rating",* and *"Votes".* The data types of the columns are object (6 columns), int64 (2 columns), and float64 (2 columns).
2. Explore the Data: We also used various data exploration techniques to understand the structure and contents of the dataset. This may include checking the dimensions of the data frame, examining the first few rows, checking data types, identifying missing values, and understanding the distribution of values in different columns.
3. Clean the Data: We have performed data cleaning operations, such as handling missing values, correcting data types, handling duplicates, and addressing any inconsistencies or errors in the data.
4. Analyze the Data: We used descriptive statistics, data visualization, and other exploratory techniques to gain insights into the data using the 'matplotlib' and 'seaborn' libraries in python. This may involve calculating summary statistics, creating histograms, box plots, and scatter plots, and performing data aggregations and grouping operations to understand patterns and trends in the data.
5. Feature Engineering: Also extracted meaningful features from the data that can be used for further analysis or modeling. This may involve creating new variables, transforming variables, and deriving insights from the data.
6. Identify Patterns and Relationships: Used data visualization and statistical analysis techniques to identify patterns, relationships, and correlations between variables in the dataset. This can help uncover interesting insights and generate hypotheses for further analysis.
7. Draw Conclusions: Summarized our findings and drawn conclusions based on the results of our analysis. Communicated our findings effectively through visualizations, reports, and presentations.
8. Validate and Interpret Results: Validated our findings using appropriate statistical techniques and interpreted the results in the context of the problem or question we are trying to find an answer.

EDA is an iterative process, and we may need to perform multiple rounds of data exploration, cleaning, and analysis to gain a comprehensive understanding of the dataset.

The following hypothesis were considered on the *'netflix_movies'* dataset and EDA was performed to confirm if those hypotheses are true or not by visualizing the data.

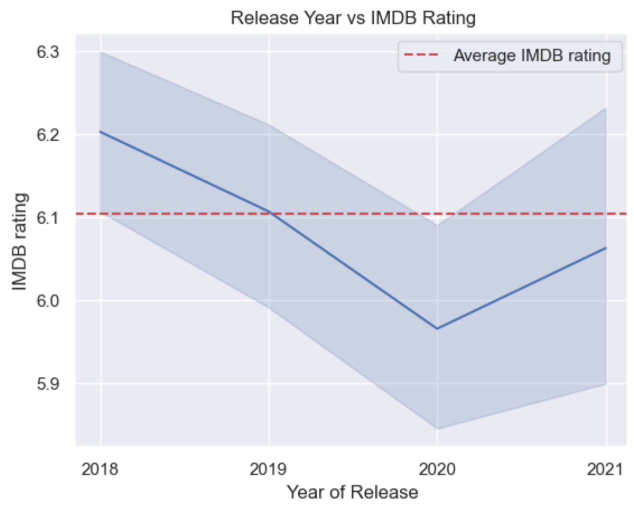
1. Is there a relationship between the runtime and IMDB rating of a movie?

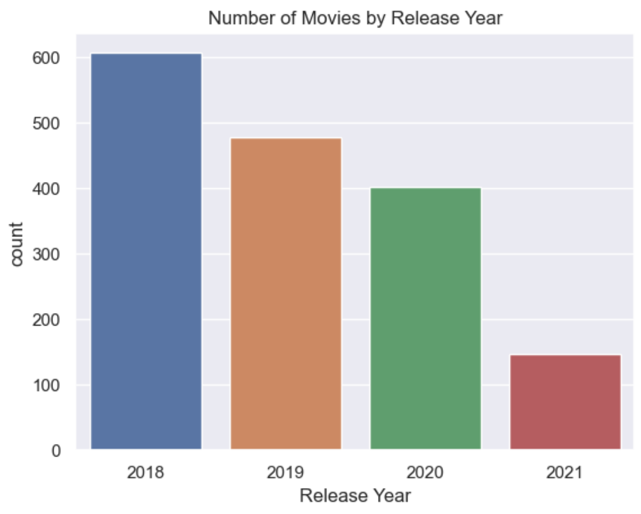


Although we are unable to verify this, there appears to be a positive correlation between the IMDB rating and the length of the films. This suggests that if the length of the films increases, so does the IMDB rating. For all Netflix-listed films released between 2018 and 2021, the average IMDB rating is around 6.1.

Therefore, we can confirm that the IMDB rating will not be affected by the length of the movie. The runtime of the film isn't critical with the IMDB rating.

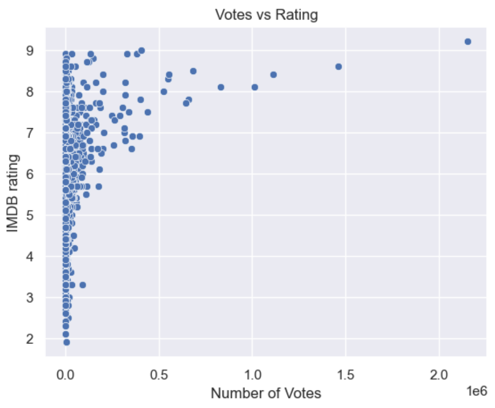
2. Do movies with a higher release year tend to have higher IMDB ratings?

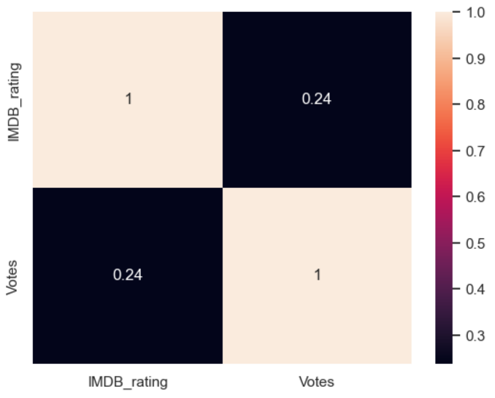




According to the plot, the IMDB ratings for movies decreased from 2018 to 2020, but increased from 2020 to 2021, but did not reach the average of 6.1, indicating that movies with good content began to be released in 2020. However, the count plot reveals that more than 600 films were released in 2018, and that the number of releases decreased each year until 2021. Even though there are fewer movies released in 2021, the average IMDB ratings are higher than in 2020.

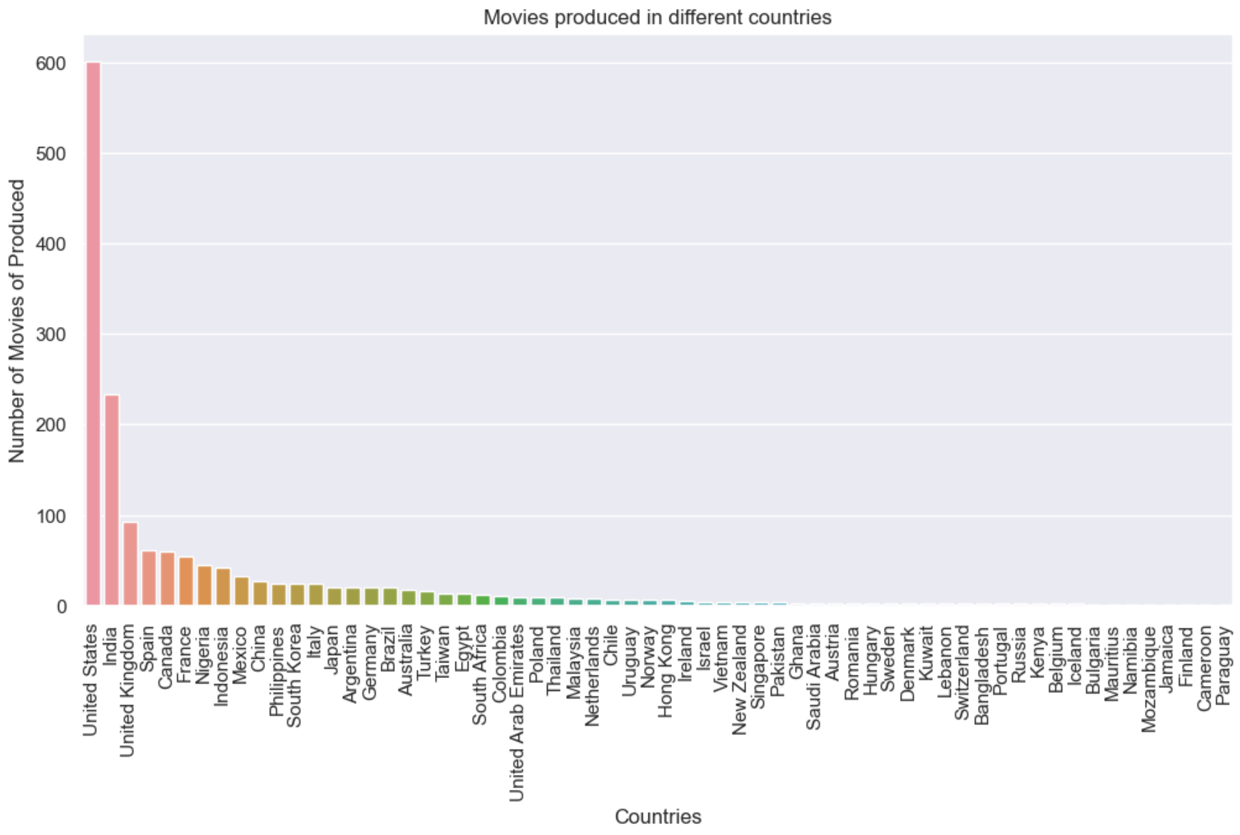
3. Is there a correlation between the number of votes and IMDB rating of a movie?





The correlation between votes and IMDB rating is not particularly strong. Given that the correlation factor is only 0.24, we can conclude that high votes do not correspond to high IMDB ratings. In any case, from the disperse plot among votes and IMDB rating we can see an exception which derive that regardless of having no connection amongst votes and evaluations, there are not many motion pictures which got largest number of most elevated and furthermore has high appraisals.

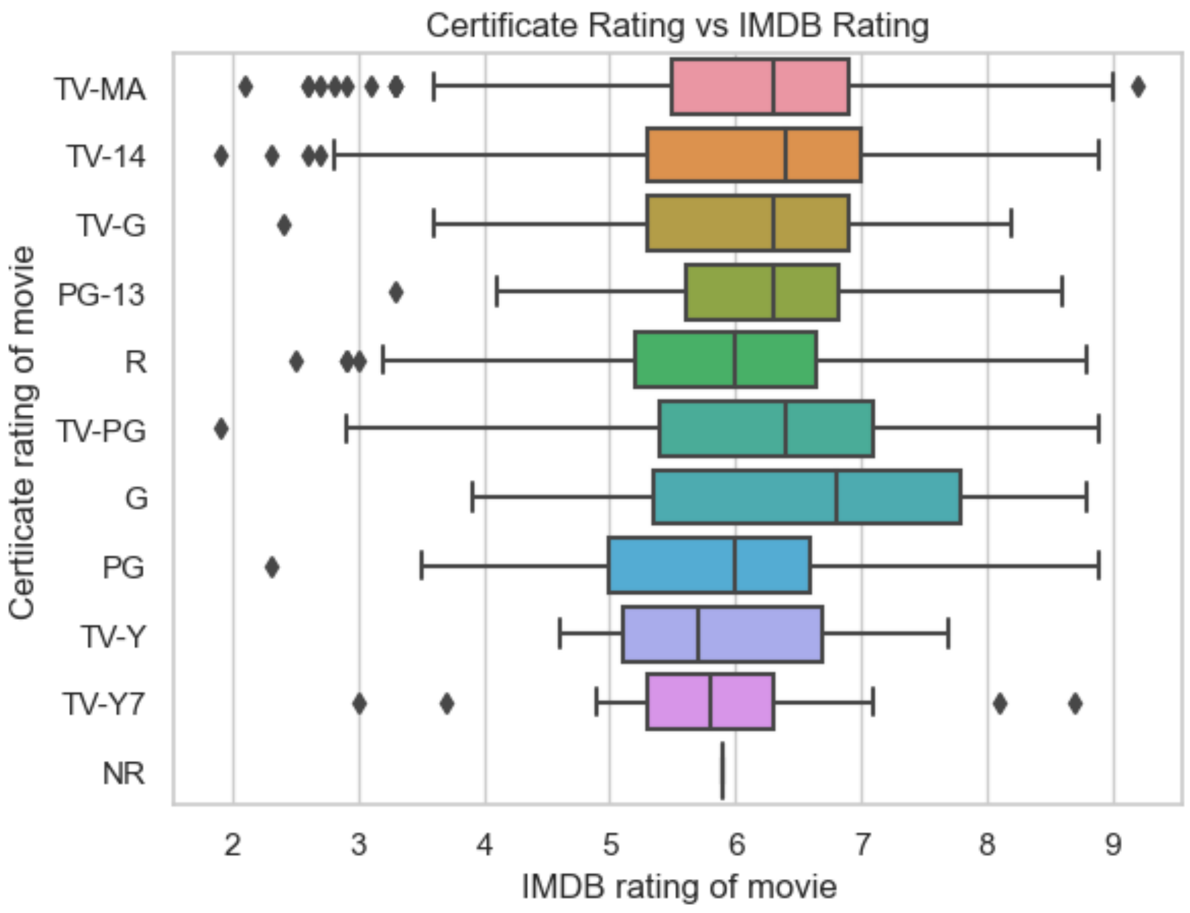
4. Are there any countries that produce more movies than others?



We can deduce that the United States has produced the most Netflix movies that were released between 2018 and 2021, assuming that only the first country is considered in the list of countries where the movie was released. To verify which nation produces the most movies, we are making this assumption here.

In this way, US has delivered more than 600 films which are released somewhere in the year of 2018 to 2021 and are listed in Netflix.

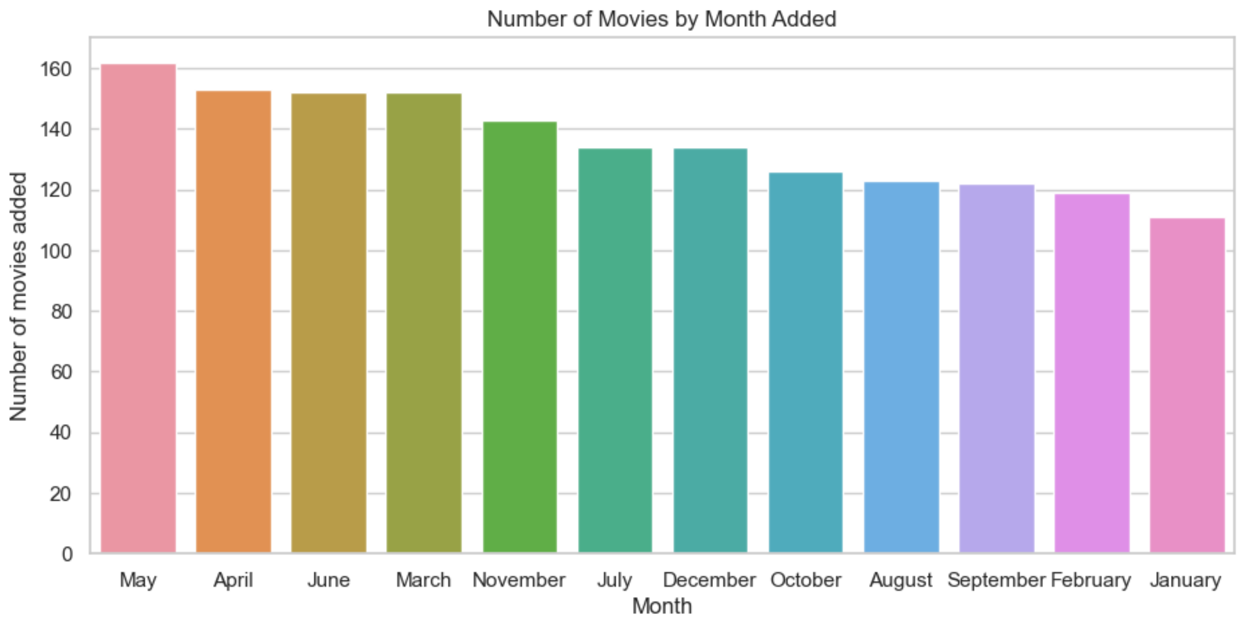
5. Is there a relationship between the certificate rating and the IMDB rating of a movie?



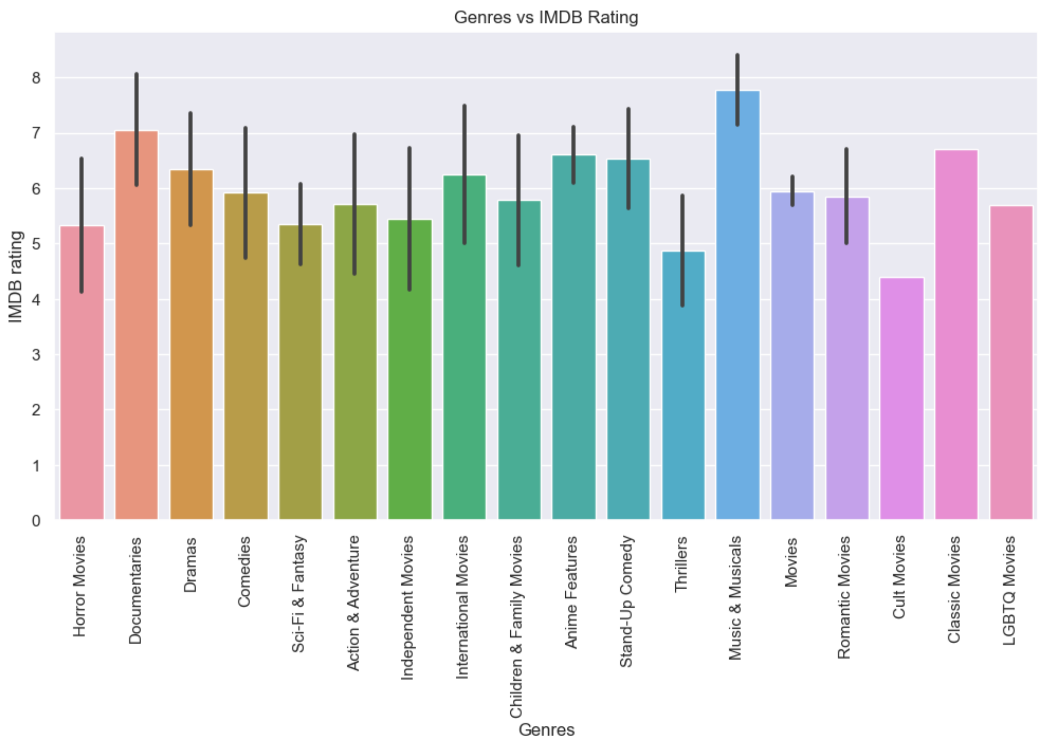
The box plot reveals that there are a few outliers in the data, indicating that some movies have ratings that are lower than the typical IMDB ratings for that certificate rating. We can likewise affirm from here, that the information is slanted right inferring that the IMDB rating isn't typically dispersed. The most elevated evaluations for a film are given for a ‘TV-MA' declaration rating however the most noteworthy normal rating is for a 'G' testament rating film, which is more than the normal of ‘TV-MA’ film.

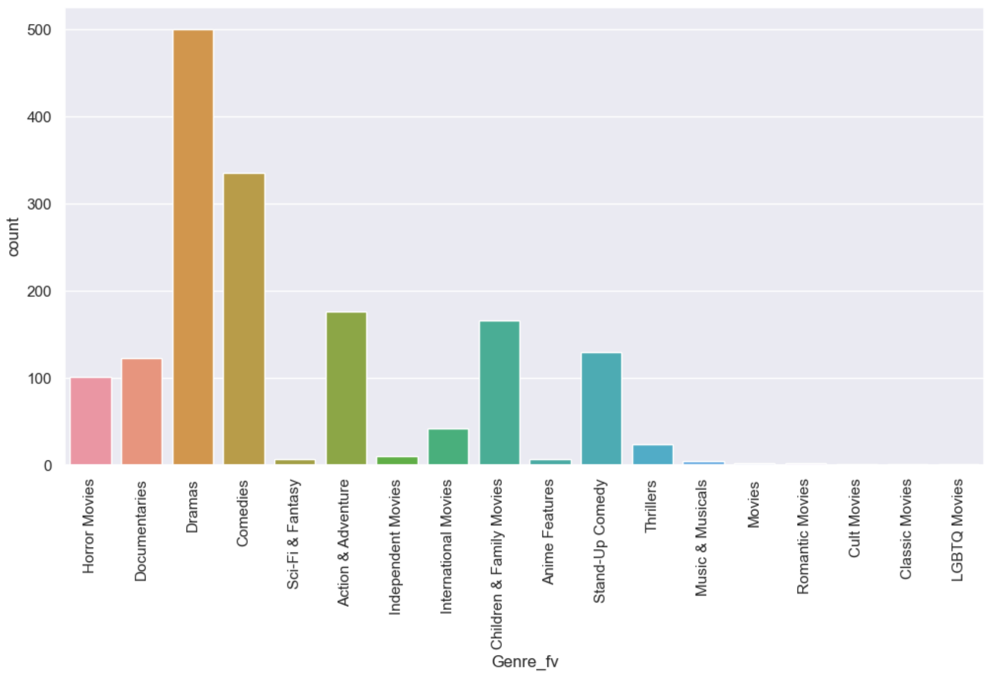
6. In which month are the movies added to the platform more?

Surprisingly, movies are most frequently added to Netflix during the summer months—April, May, and June—rather than spring. In May, Netflix added more than 160 movies that were released between 2018 and 2021.



7. Do certain genres tend to have higher IMDB ratings than others?

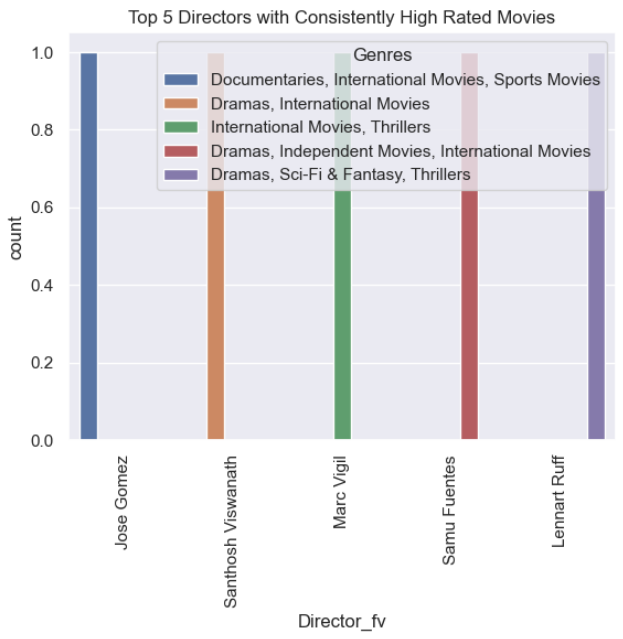




We can deduct from the preceding plot that movies in the "Music & Musicals" genre have the highest average IMDB ratings of any other genre. "Cult Movies" is listed as the lowest. Obviously, motion pictures with classes like 'Dramas' and 'Comedies' which have big number of film count don't have the most noteworthy typical rating, which perhaps due to few motion pictures getting low appraisals in that kind which cut down the normal.

However, we can confirm that, in contrast to films in the "Music & Musicals" genre, films in the "Dramas" genre will have a significant impact on their IMDB ratings.

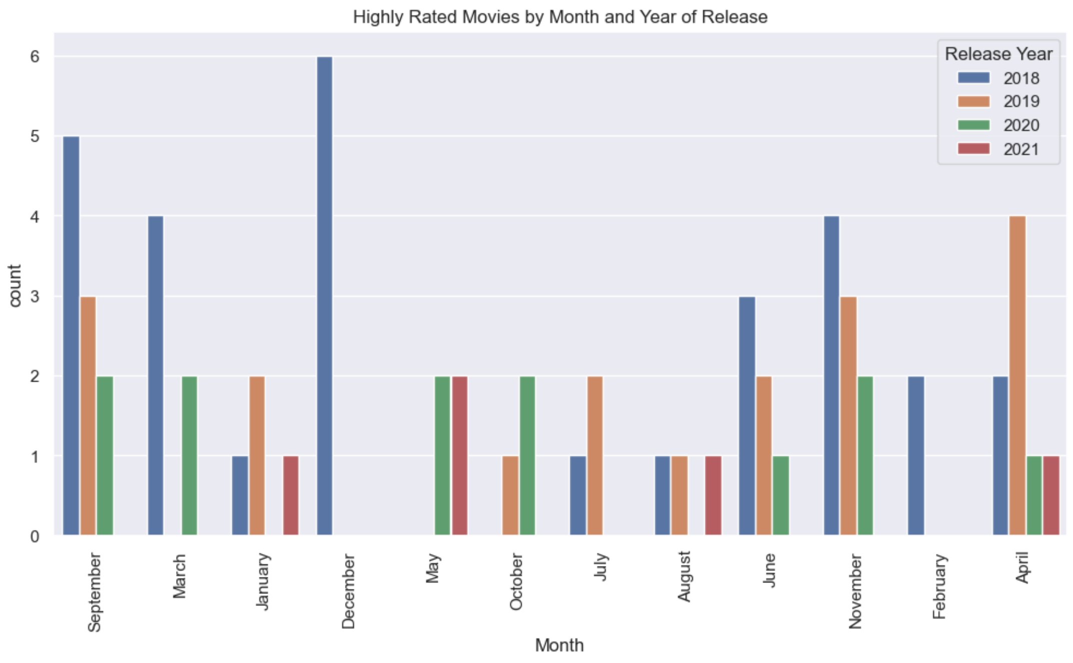
8. Are there any directors who consistently produce highly rated movies across different genres?



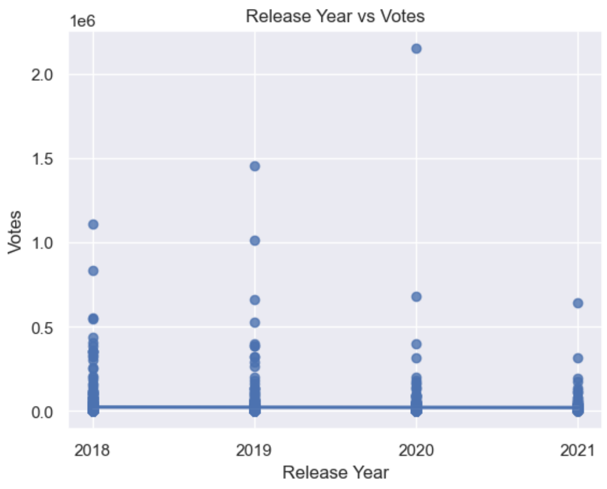
We can conclude from the above count plot that these five directors have maintained moral behavior in their films. These films, which were released between 2018 and 2021 and are also available on Netflix, have the highest average IMDB rating. Additionally, the fact that movies in the "Dramas" subgenre have maintained a favorable average rating lends credence to the preceding hypothesis.

9. Are there any patterns in the release dates of highly rated movies?

It is evident from this that movies that were released in 2018 are only listed in December, whereas movies were released in April each year. However, September saw a greater average number of films released.

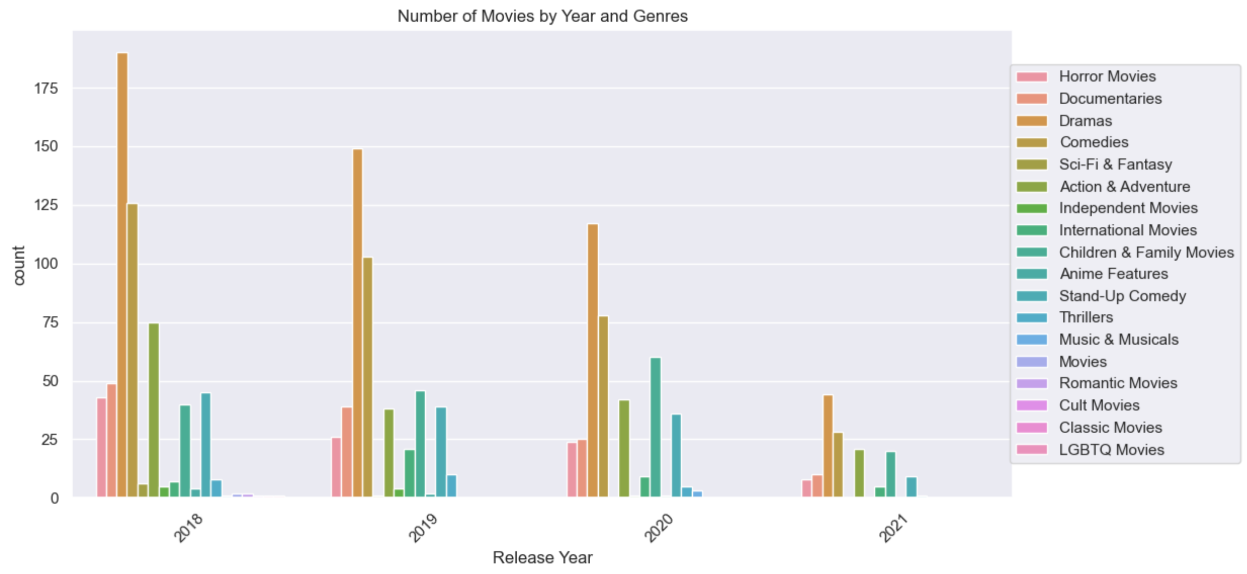


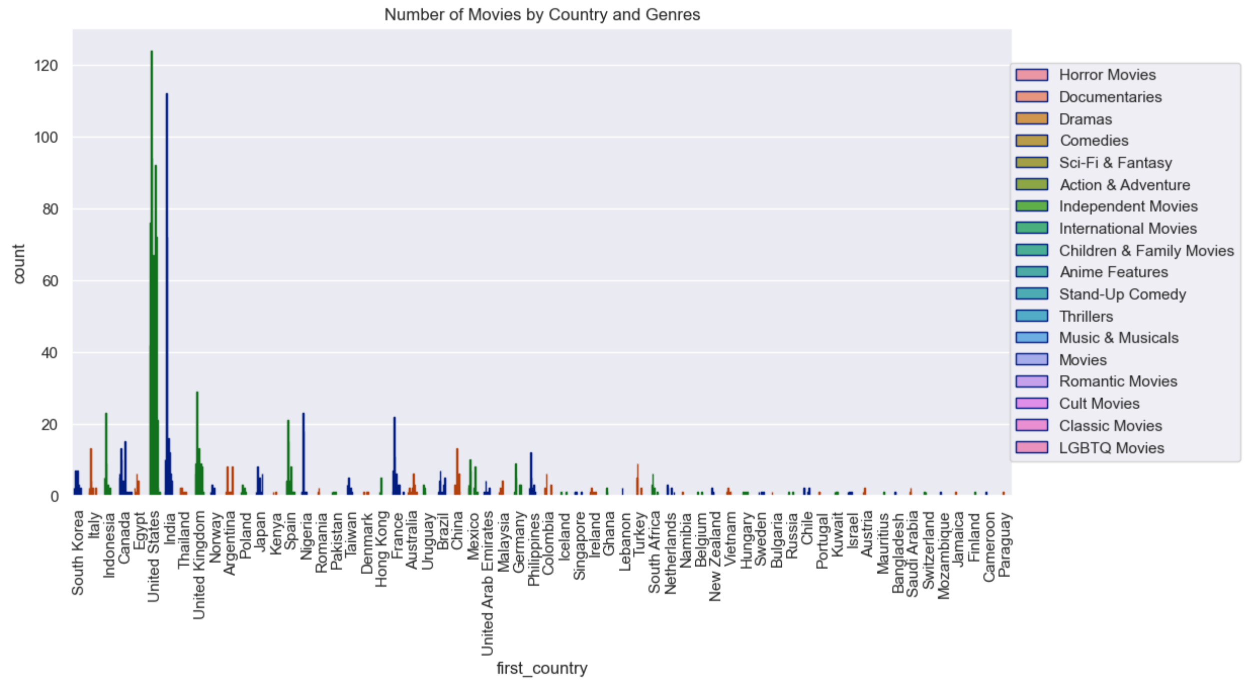
10. Is there a relationship between the number of votes received by a movie and its release year?



The year of release has had a significant impact on the number of votes cast for the films. In 2020, there appears to be an anomaly with abnormally high votes in comparison to other candidates. Based on the conclusion drawn from the previous hypothesis, the films that were released in the year 2019 received the most votes, despite having lower IMDB ratings.

11. Are there any genres that have become popular over time, and in which countries?

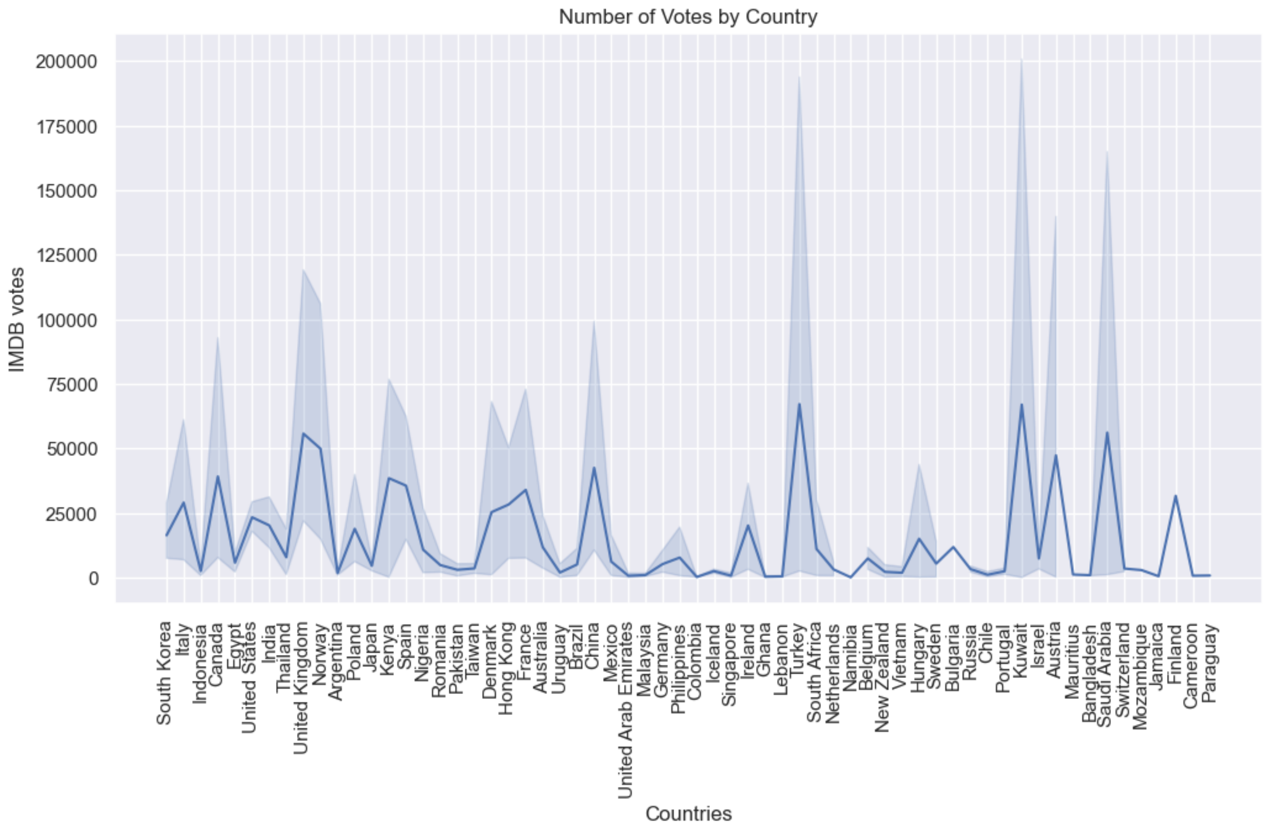




In view of the above plots, we can affirm that 2018 has more deliveries and US appears to offer more to that. Also, we can back up the previous hypothesis that movies in the drama and comedy genres are making more movies and getting better reviews on average.

12. Are there any patterns in the number of votes received by movies from different countries?

Based on the votes, it doesn't seem like there is a particular pattern, but it seems like most of the votes came from Turkey, even though more movies are released in the United States. Therefore, we can conclude that there is no country-specific voting pattern. The votes are listed in the Netflix OTT and have no bearing on the country in which the films were released.



4. **Conclusion**:

To add IMDb ratings and votes to Netflix, we would follow a multi-step process. First, we would scrape or extract data from the IMDb website using web scraping techniques to collect information on ratings and votes for movies or TV shows available on Netflix. IMDb is a popular online database that provides ratings and votes from millions of users for various movies and TV shows. Next, we would clean and process the extracted data to ensure accuracy and consistency. This might involve handling missing values, correcting data types, and removing any duplicates or outliers. Once the data is cleaned and processed, we would integrate it into the Netflix database to enrich the content information with IMDb ratings and votes. This can be achieved by matching the movie or TV show titles from Netflix with the corresponding titles on IMDb and then merging the IMDb data with the Netflix data based on a unique identifier, such as the movie or TV show title or IMDb ID. After merging the data, we would conduct additional analysis and visualization to explore the relationship between IMDb ratings, votes, and other variables in the Netflix dataset, such as release year, genres, and runtime. This analysis would help Netflix gain insights into the quality and popularity of its content based on user ratings and votes from IMDb. The IMDb ratings and votes can also be used to improve content recommendation algorithms, personalize user experiences, and make data-driven decisions for content acquisition and production. However, it's important to note that using IMDb data may have limitations, such as potential biases in user ratings, sample size, and representativeness of the IMDb user base. Therefore, we would exercise careful consideration and validation of the IMDb data to ensure its reliability and suitability for integration into the Netflix database.

5. References:

[1]. Netflix Movies and TV Shows, https://www.kaggle.com/datasets/shivamb/netflix-shows

[2]. Amazon Prime Movies and TV Shows, https://www.kaggle.com/datasets/shivamb/amazon-prime-movies-and-tv-shows

[3]. Disney+ Movies and TV Shows, https://www.kaggle.com/datasets/shivamb/disney-movies-and-tv-shows

[4]. Hulu Movies and TV Shows, https://www.kaggle.com/datasets/shivamb/hulu-movies-and-tv-shows

[5]. Merge, join, concatenate and compare, https://pandas.pydata.org/docs/user_guide/merging.html

[6]. Saving a Pandas Dataframe as a CSV, https://www.geeksforgeeks.org/saving-a-pandas-dataframe-as-a-csv/

[7]. Pandas DataFrame dropna() Method, https://www.w3schools.com/python/pandas/ref_df_dropna.asp

[8]. How to drop one or multiple columns in Pandas Dataframe, https://www.geeksforgeeks.org/how-to-drop-one-or-multiple-columns-in-pandas-dataframe/

[9]. Python String strip() Method, https://www.w3schools.com/python/ref_string_strip.asp

[10]. Python IMDbPY – Searching a movie, https://www.geeksforgeeks.org/python-imdbpy-searching-a-movie/

[11]. Pandas DataFrame copy() Method, https://www.w3schools.com/python/pandas/ref_df_copy.asp

[12]. Regular Expression in Python with Examples | Set 1, https://www.geeksforgeeks.org/regular-expression-python-examples-set-1/

[13]. Python | Pandas DataFrame.astype(), https://www.geeksforgeeks.org/python-pandas-dataframe-astype/

[14]. DataFrame.to_excel() method in Pandas, https://www.geeksforgeeks.org/dataframe-to_excel-method-in-pandas/