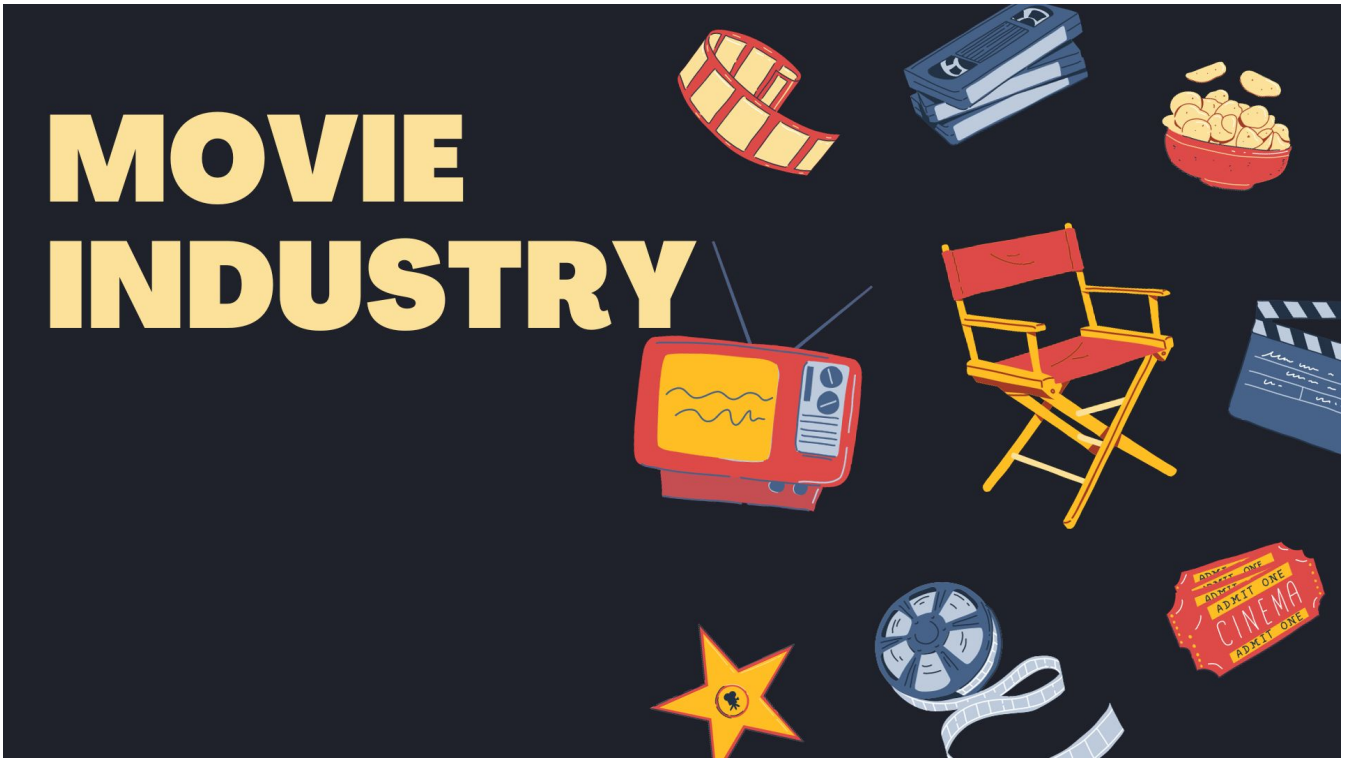
EXPLORATORY DATA ANALYSIS IN THE MOVIE INDUSTRY



OVERVIEW

Filmmaking or film production is the process by which a motion picture is produced. Filmmaking involves a number of complex and discrete stages, beginning with an initial story, idea, or commission.

The film production process can be divided into countless steps to take a film from concept to a finished piece. There are three key stages that take place in the production of any film: pre-production (planning), production (filming), and post-production (editing, color-grading, and visual effects). However, a few of the common factors that contribute to a successful film include: a compelling storyline; a well written script; great actors who have a reach to the audience; a visionary director alongside a director of photography and editor. among other factors.

In this project, I'll be using exploratory data analysis to generate insights for a business stakeholder (Microsoft) about what type of films are currently doing the best at the box office.

Problem Statement

My project's aim is to assist the head of Microsoft's new movie studio with insights that would help them decide what type of films to create based on the currently trending movies, movie ratings and genres. Microsoft may be able to know the type of movies to create based on the above factors, calculating the cost of production, as well as understanding their target audience.

Data Understanding

In this project I'll be using movie datasets from IMDB and Box Office Mojo which are online databases of information related to films, television series, podcasts, etc. The data files provide the start year, ratings, movie titles as well as other features like the genres and running time of the movies over the last decades.

In [1]:
```
# Importing necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
# Loading the datasets
basics = pd.read_csv('title.basics.csv')
ratings = pd.read_csv('title.ratings.csv')
```

Basics Dataset

The basics dataset includes records from 2010 to 2022. It contains information about the primary tiltle, orignal title, start year, tuntime and genres of the movies.

In [3]:
```
# Displaying the top of the dataset
basics.head()
```

Out[3]:

	tconst	primary_title	original_title	start_year	runtime_minutes	genres
0	tt0063540	Sunghursh	Sunghursh	2013	175.0	Action,Crime,Drama
1	tt0066787	One Day Before the Rainy Season	Ashad Ka Ek Din	2019	114.0	Biography,Drama
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drama
3	tt0069204	Sabse Bada Sukh	Sabse Bada Sukh	2018	NaN	Comedy,Drama
4	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy

In [4]:
```
# Displaying the bottom of the dataset
basics.tail()
```

Out[4]:

	tconst	primary_title	original_title	start_year	runtime_minutes	genres
146139	tt9916538	Kuambil Lagi Hatiku	Kuambil Lagi Hatiku	2019	123.0	Drama
146140	tt9916622	Rodolpho Teóphilo - O Legado de um Pioneiro	Rodolpho Teóphilo - O Legado de um Pioneiro	2015	NaN	Documentary
146141	tt9916706	Dankyavar Danka	Dankyavar Danka	2013	NaN	Comedy
146142	tt9916730	6 Gunn	6 Gunn	2017	116.0	NaN
146143	tt9916754	Chico Albuquerque - Revelações	Chico Albuquerque - Revelações	2013	NaN	Documentary

In [5]:
```
# Checking the info of the dataset
basics.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 146144 entries, 0 to 146143
Data columns (total 6 columns):
 #   Column           Non-Null Count   Dtype  
---  ------           --------------   -----  
 0   tconst           146144 non-null  object 
 1   primary_title    146144 non-null  object 
 2   original_title   146123 non-null  object 
 3   start_year       146144 non-null  int64  
 4   runtime_minutes  114405 non-null  float64
 5   genres           140736 non-null  object 
dtypes: float64(1), int64(1), object(4)
memory usage: 6.7+ MB
```

In [6]:
```
# Checking the columns
list(basics)
```

Out[6]:
```
['tconst',
 'primary_title',
 'original_title',
 'start_year',
 'runtime_minutes',
 'genres']
```

In [7]:
```
# Counting number of unique elements in primary_title
basics['primary_title'].value_counts()
```

Out[7]:
```
Home                              24
The Return                        20
Broken                            20
Homecoming                        16
Alone                             16
                                  ..
Viktor                             1
Hooked to the Silver Screen        1
Anaamika                           1
Blood for Blood                    1
Chico Albuquerque - Revelações     1
Name: primary_title, Length: 136071, dtype: int64
```

In [8]:
```
# Counting number of unique elements in original_title
basics['original_title'].value_counts()
```

Out[8]:
```
Broken                                19
Home                                  18
The Return                            17
Alone                                 13
Freedom                               13
                                      ..
With Real Stars Above My Head          1
Alle anderen sind nicht gleich anders  1
ACP Ranveer                            1
Longtails                              1
Chico Albuquerque - Revelações         1
Name: original_title, Length: 137773, dtype: int64
```

In [9]:
```
# Counting number of unique elements in start_year
basics['start_year'].value_counts()
```

Out[9]:
```
2017    17504
2016    17272
2018    16849
2015    16243
2014    15589
2013    14709
2012    13787
2011    12900
2010    11849
2019     8379
2020      937
2021       83
2022       32
2023        5
2024        2
2026        1
2025        1
2115        1
2027        1
Name: start_year, dtype: int64
```

In [10]:
```
# Counting number of unique elements in genres
basics['genres'].value_counts()
```

Out[10]:
```
Documentary                 32185
Drama                       21486
Comedy                       9177
Horror                       4372
Comedy,Drama                 3519
                            ...
Adventure,Music,Mystery         1
Documentary,Horror,Romance      1
Sport,Thriller                  1
Comedy,Sport,Western            1
Adventure,History,War           1
Name: genres, Length: 1085, dtype: int64
```

Ratings Dataset

The ratings dataset contain records for average rating and number of votes per movie title.

In [11]:
```
# Displaying the top of the dataset
ratings.head()
```

Out[11]:

	tconst	averagerating	numvotes
0	tt10356526	8.3	31
1	tt10384606	8.9	559
2	tt1042974	6.4	20
3	tt1043726	4.2	50352
4	tt1060240	6.5	21

In [12]:
```
# Displaying the bottom of the dataset
ratings.tail()
```

Out[12]:

	tconst	averagerating	numvotes
73851	tt9805820	8.1	25
73852	tt9844256	7.5	24
73853	tt9851050	4.7	14
73854	tt9886934	7.0	5
73855	tt9894098	6.3	128

In [13]:
```
# Checking the info of the dataset
ratings.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 73856 entries, 0 to 73855
Data columns (total 3 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   tconst         73856 non-null  object 
 1   averagerating  73856 non-null  float64
 2   numvotes       73856 non-null  int64  
dtypes: float64(1), int64(1), object(1)
memory usage: 1.7+ MB
```

In [14]:
```
# Checking the columns
list(ratings)
```

Out[14]:
```
['tconst', 'averagerating', 'numvotes']
```

In [15]:
```
# Counting number of unique elements in averagerating
ratings['averagerating'].value_counts()
```

Out[15]:
```
7.0     2262
6.6     2251
7.2     2249
6.8     2239
6.5     2221
        ...
9.6       18
10.0      16
9.8       15
9.7       12
9.9        5
Name: averagerating, Length: 91, dtype: int64
```

In [16]:
```
# Counting number of unique elements in numvotes
ratings['numvotes'].value_counts()
```

Out[16]:
```
6         2875
5         2699
7         2476
8         2167
9         1929
          ...
11307        1
4361         1
365110       1
2288         1
4057         1
Name: numvotes, Length: 7349, dtype: int64
```

Data Preparation

Data Cleaning

Checking for duplicates

In [17]:
```
# Checking if we have duplicates in the basics dataset
basics.duplicated().sum()
```

Out[17]:
```
0
```

There are zero duplicated values in the basics dataset

In [18]:
```
# Checking if we have duplicates in the ratings dataset
ratings.duplicated().sum()
```

Out[18]:
```
0
```

There are zero duplicated values in the ratings dataset as well.

Checking for missing values

In [19]:
```
# Checking the total numbers of missing values
print(basics.isnull().sum())
```

```
tconst                 0
primary_title          0
original_title        21
start_year             0
runtime_minutes    31739
genres              5408
dtype: int64
```

This dataset has a total of 21 missing values in the originatl_title column, 31739 in the runtime_minutes column and 5408 in rhe genres column.

In [20]:
```
print(ratings.isnull().sum())
```

```
tconst           0
averagerating    0
numvotes         0
dtype: int64
```

This dataset does not contain any missing values in any of the columns.

Removing missing value

In [21]:
```
# Drop the missing values from the basics dataset and assign the new dataset the variable basics.
basics1 = basics.dropna()
basics1
```

Out[21]:

	tconst	primary_title	original_title	start_year	runtime_minutes	genres
0	tt0063540	Sunghursh	Sunghursh	2013	175.0	Action,Crime,Drama
1	tt0066787	One Day Before the Rainy Season	Ashad Ka Ek Din	2019	114.0	Biography,Drama
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drama
4	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy
5	tt0111414	A Thin Life	A Thin Life	2018	75.0	Comedy
...	...	...	...	...	...	...
146134	tt9916160	Drømmeland	Drømmeland	2019	72.0	Documentary
146135	tt9916170	The Rehearsal	O Ensaio	2019	51.0	Drama
146136	tt9916186	Illenau - die Geschichte einer ehemaligen Heil...	Illenau - die Geschichte einer ehemaligen Heil...	2017	84.0	Documentary
146137	tt9916190	Safeguard	Safeguard	2019	90.0	Drama,Thriller
146139	tt9916538	Kuambil Lagi Hatiku	Kuambil Lagi Hatiku	2019	123.0	Drama

112232 rows × 6 columns

The new dataframe contains 112232 rows and 6 columns.

In [22]:
```
# Checking if there are any missing values in the new dataframe
basics1.isnull().sum()
```

Out[22]:
```
tconst             0
primary_title      0
original_title     0
start_year         0
runtime_minutes    0
genres             0
dtype: int64
```

All the nissing values have been dropped.

Meging Datasets

In [23]:
```
data = pd.merge(basics1,ratings,on="tconst",how="left")
```

In [24]:
```
# Listing the top 10 movies with the highest voter count
data = pd.merge(basics1,ratings,on="tconst",how="left").sort_values(by="numvotes",ascending=Fals
data[['primary_title', 'start_year', 'genres', 'averagerating', 'numvotes']].head(10)
```

Out[24]:

	primary_title	start_year	genres	averagerating	numvotes
4106	Inception	2010	Action,Adventure,Sci-Fi	8.8	1841066.0
3961	The Dark Knight Rises	2012	Action,Thriller	8.4	1387769.0
264	Interstellar	2014	Adventure,Drama,Sci-Fi	8.6	1299334.0
15896	Django Unchained	2012	Drama,Western	8.4	1211405.0
303	The Avengers	2012	Action,Adventure,Sci-Fi	8.1	1183655.0
468	The Wolf of Wall Street	2013	Biography,Crime,Drama	8.2	1035358.0
2836	Shutter Island	2010	Mystery,Thriller	8.1	1005960.0
20296	Guardians of the Galaxy	2014	Action,Adventure,Comedy	8.1	948394.0
4533	Deadpool	2016	Action,Adventure,Comedy	8.0	820847.0
4233	The Hunger Games	2012	Action,Adventure,Sci-Fi	7.2	795227.0

In [25]:
```
# List the top 10 high rated films.
# We'll consider films which received ratings from atleast 20,000 users.

data[data.numvotes > 20000].sort_values(by="averagerating",ascending=False)
data[['primary_title', 'start_year', 'genres', 'averagerating', 'numvotes']].head(10)
```

Out[25]:

	primary_title	start_year	genres	averagerating	numvotes
4106	Inception	2010	Action,Adventure,Sci-Fi	8.8	1841066.0
3961	The Dark Knight Rises	2012	Action,Thriller	8.4	1387769.0
264	Interstellar	2014	Adventure,Drama,Sci-Fi	8.6	1299334.0
15896	Django Unchained	2012	Drama,Western	8.4	1211405.0
303	The Avengers	2012	Action,Adventure,Sci-Fi	8.1	1183655.0
468	The Wolf of Wall Street	2013	Biography,Crime,Drama	8.2	1035358.0
2836	Shutter Island	2010	Mystery,Thriller	8.1	1005960.0
20296	Guardians of the Galaxy	2014	Action,Adventure,Comedy	8.1	948394.0
4533	Deadpool	2016	Action,Adventure,Comedy	8.0	820847.0
4233	The Hunger Games	2012	Action,Adventure,Sci-Fi	7.2	795227.0

Inception (2010) has the highest average rating followed by Interstellar (2014) and The Dark Knight Rises (2012). This shows that Action, Drama, Adventure and Sci-Fi movies receive a higher average rating.

In [26]:
```
# Checking the info of the merged dataset
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 112232 entries, 4106 to 112231
Data columns (total 8 columns):
 #   Column           Non-Null Count   Dtype  
---  ------           --------------   -----  
 0   tconst           112232 non-null  object 
 1   primary_title    112232 non-null  object 
 2   original_title   112232 non-null  object 
 3   start_year       112232 non-null  int64  
 4   runtime_minutes  112232 non-null  float64
 5   genres           112232 non-null  object 
 6   averagerating    65720 non-null   float64
 7   numvotes         65720 non-null   float64
dtypes: float64(3), int64(1), object(4)
memory usage: 7.7+ MB
```

In [27]:
```
# Checking the first five elements of the new dataset
data.head()
```

Out[27]:

	tconst	primary_title	original_title	start_year	runtime_minutes	genres	averagerating	nu
4106	tt1375666	Inception	Inception	2010	148.0	Action,Adventure,Sci-Fi	8.8	18
3961	tt1345836	The Dark Knight Rises	The Dark Knight Rises	2012	164.0	Action,Thriller	8.4	13
264	tt0816692	Interstellar	Interstellar	2014	169.0	Adventure,Drama,Sci-Fi	8.6	12
15896	tt1853728	Django Unchained	Django Unchained	2012	165.0	Drama,Western	8.4	12
303	tt0848228	The Avengers	The Avengers	2012	143.0	Action,Adventure,Sci-Fi	8.1	11

In [28]:
```
# Create a file called “data.csv” and save the final table in .csv format
data.to_csv("data.csv")
```

EXPLORATORY DATA ANALYSIS

Univariate Analysis

In [29]:
```
# Descriptive statistics of our dataset
data.describe()
```

Out[29]:

	start_year	runtime_minutes	averagerating	numvotes
count	112232.000000	112232.000000	65720.000000	6.572000e+04
mean	2014.402078	86.261556	6.320902	3.954674e+03
std	2.639042	167.896646	1.458878	3.208823e+04
min	2010.000000	1.000000	1.000000	5.000000e+00
25%	2012.000000	70.000000	5.500000	1.600000e+01
50%	2014.000000	87.000000	6.500000	6.200000e+01
75%	2017.000000	99.000000	7.300000	3.520000e+02
max	2022.000000	51420.000000	10.000000	1.841066e+06

The average rating for movies is 6.5 for all genres while the average runtime is 87 minutes.

In [30]:
```
data['averagerating'].describe()
```

Out[30]:
```
count    65720.000000
mean         6.320902
std          1.458878
min          1.000000
25%          5.500000
50%          6.500000
75%          7.300000
max         10.000000
Name: averagerating, dtype: float64
```

In [31]:
```
data['genres'].describe()
```

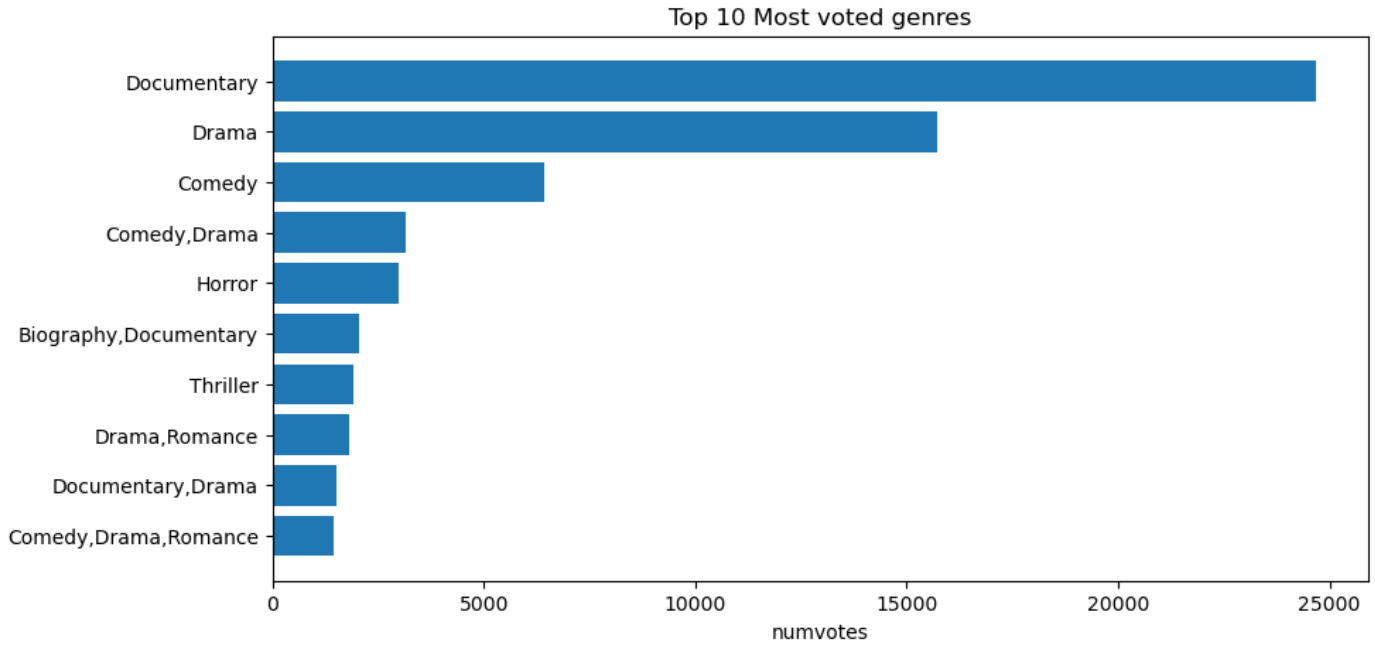
Out[31]:
```
count          112232
unique           1040
top       Documentary
freq            24672
Name: genres, dtype: object
```

In [32]:
```
# Checking the frequency distribution of the genres in our dataset
data.genres.value_counts()
```

Out[32]:
```
Documentary                    24672
Drama                          15725
Comedy                          6413
Comedy,Drama                    3163
Horror                          2975
                               ...
Family,Musical,Sport               1
Comedy,Documentary,Thriller        1
Action,Adult,Comedy                1
Documentary,News,Reality-TV        1
Family,War                         1
Name: genres, Length: 1040, dtype: int64
```

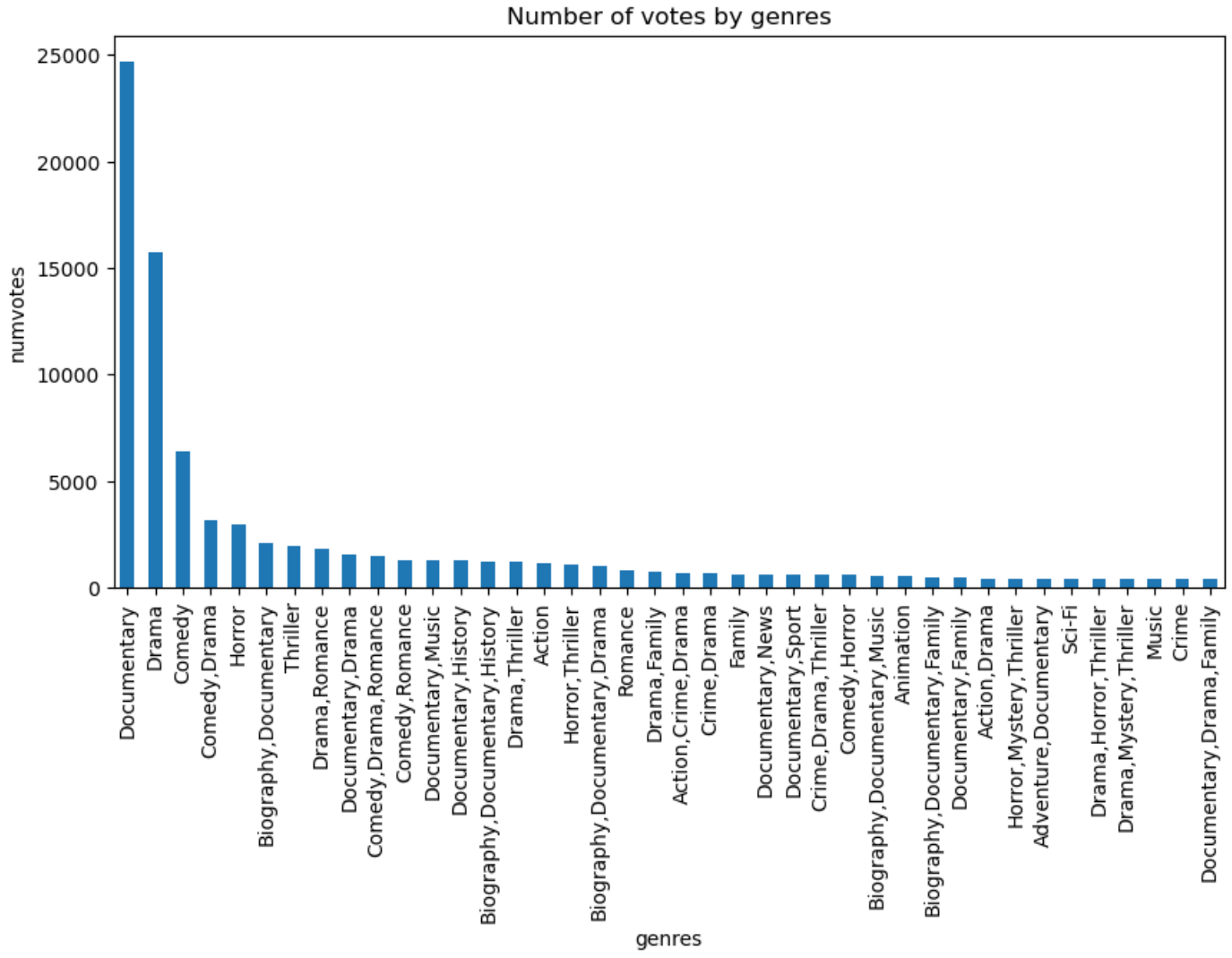
The highest rated genres are Documentary films followed by drama then comedy. 'Family, War' is one of the least rated genre.

In [33]:
```
# Checking top 10 most voted genres and plot a bar graph showing the same

best_gen=pd.DataFrame(data['genres'].value_counts().head(10)).sort_values(by='genres')

plt.figure(figsize = (10,5))
plt.barh(best_gen.index, best_gen.genres)
plt.xlabel('numvotes')
plt.title('Top 10 Most voted genres')
plt.show()
```



In [34]:
```
# Create a histogram to symbolize the relationship between genres and numvotes
data.genres.value_counts().nlargest(40).plot(kind='bar', figsize=(10,5))
# Add labels and title
plt.title("Number of votes by genres")
plt.ylabel('numvotes')
plt.xlabel('genres')
```

Out[34]:
```
Text(0.5, 0, 'genres')
```



The above diagram indicate that Documentary films have the highest number of votes.

Average runtime for films

The average runtime for films is 87 minutes.

In [35]:
```
# Descriptive statistics for runtime_minutes
data.runtime_minutes.describe()
```

Out[35]:
```
count    112232.000000
mean         86.261556
std         167.896646
min           1.000000
25%          70.000000
50%          87.000000
75%          99.000000
max       51420.000000
Name: runtime_minutes, dtype: float64
```

In [36]:
```
use = data[data.runtime_minutes.notnull()]
use["runtime_minutes"] = use.runtime_minutes.astype(int)
use[use.runtime_minutes>50000]
```

Out[36]:

	tconst	primary_title	original_title	start_year	runtime_minutes	genres	averagerating	numvotes
103414	tt8273150	Logistics	Logistics	2012	51420	Documentary	5.0	17.0

The above data shows that Logistics (2012) has the highest number of runtime in minutes. This means that it is the longest movie in our dataset.

In [37]:
```
use.sort_values(by="runtime_minutes",ascending=False).head()
```

Out[37]:

	tconst	primary_title	original_title	start_year	runtime_minutes	genres	averagerating
103414	tt8273150	Logistics	Logistics	2012	51420	Documentary	5.0
36745	tt2659636	Modern Times Forever	Modern Times Forever	2011	14400	Documentary	6.2
97438	tt7492094	Nari	Nari	2017	6017	Documentary	NaN
71074	tt5068890	Hunger!	Hunger!	2015	6000	Documentary,Drama	NaN
72306	tt5136218	London EC1	London EC1	2015	5460	Comedy,Drama,Mystery	NaN

This data shows that most of the films with the highest number of runtime are documentaries. Logistics is the longest followed by Modern Times Forever.

I grouped the data by primary_title and summed the numvotes of each title, resulting in a new dataframe that shows the primary titles with the highest number of votes, sorted in descending order and printed the top 10 most popular titles (primary_title) by the number of votes (numvotes).

In [38]:
```
# group by primary_title and sum the numvotes
title_popularity = data.groupby('primary_title')['numvotes'].sum().sort_values(ascending=False)
```

```
# print the top 10 most popular title
print(title_popularity.head(10))
```

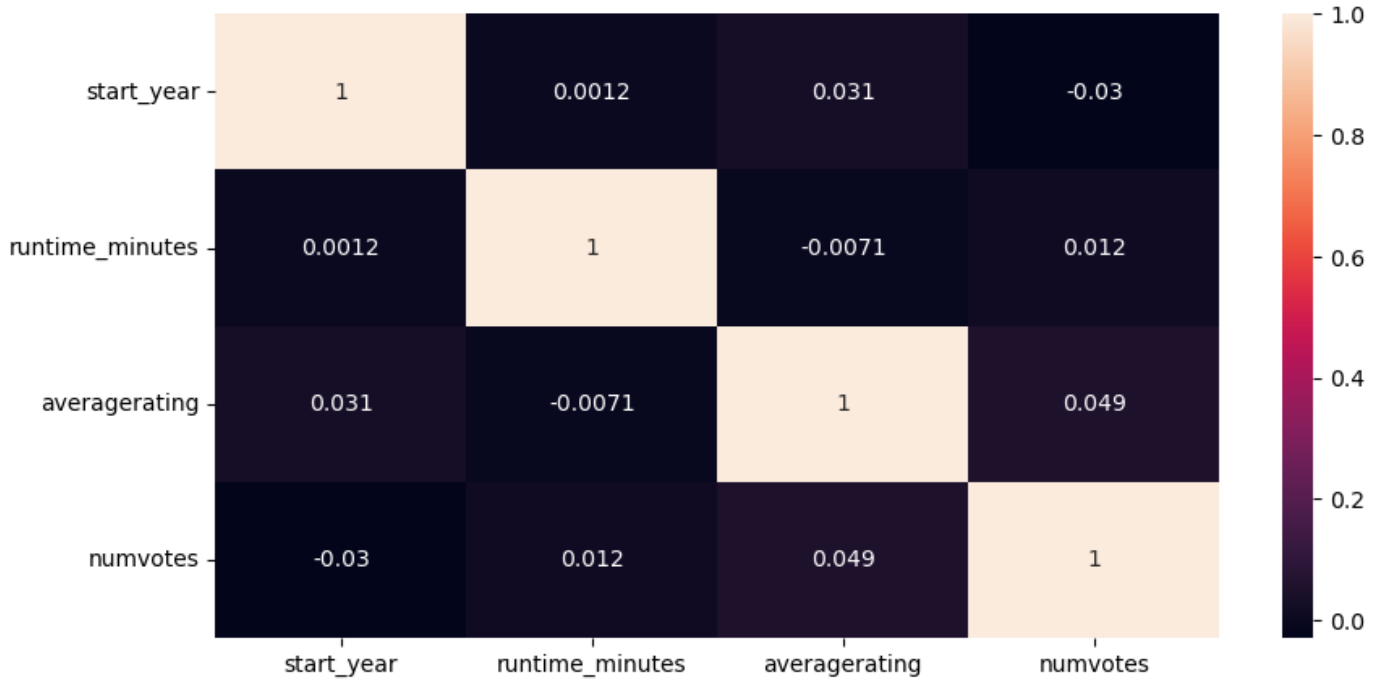
```
primary_title
Inception                  1841066.0
The Dark Knight Rises      1387769.0
Interstellar               1299334.0
Django Unchained           1211405.0
The Avengers               1183655.0
The Wolf of Wall Street    1035358.0
Shutter Island             1005960.0
Guardians of the Galaxy     948394.0
Deadpool                    820847.0
The Hunger Games            795227.0
Name: numvotes, dtype: float64
```

Inception has the highest number of votes, followed by The Dark Knight Rises.

BIVARIATE ANALYSIS

Bivariate Analysis helps to understand how variables are related to each other and the relationship between dependent and independent variables present in the dataset.

In [39]:
```
# Heatmap for visualizing the strength and direction of the relationship between
# different variables in the dataset.

plt.figure(figsize=(10,5))
corr = data.corr()
sns.heatmap(corr, xticklabels=corr.columns, yticklabels=corr.columns, annot=True)
plt.show()
```

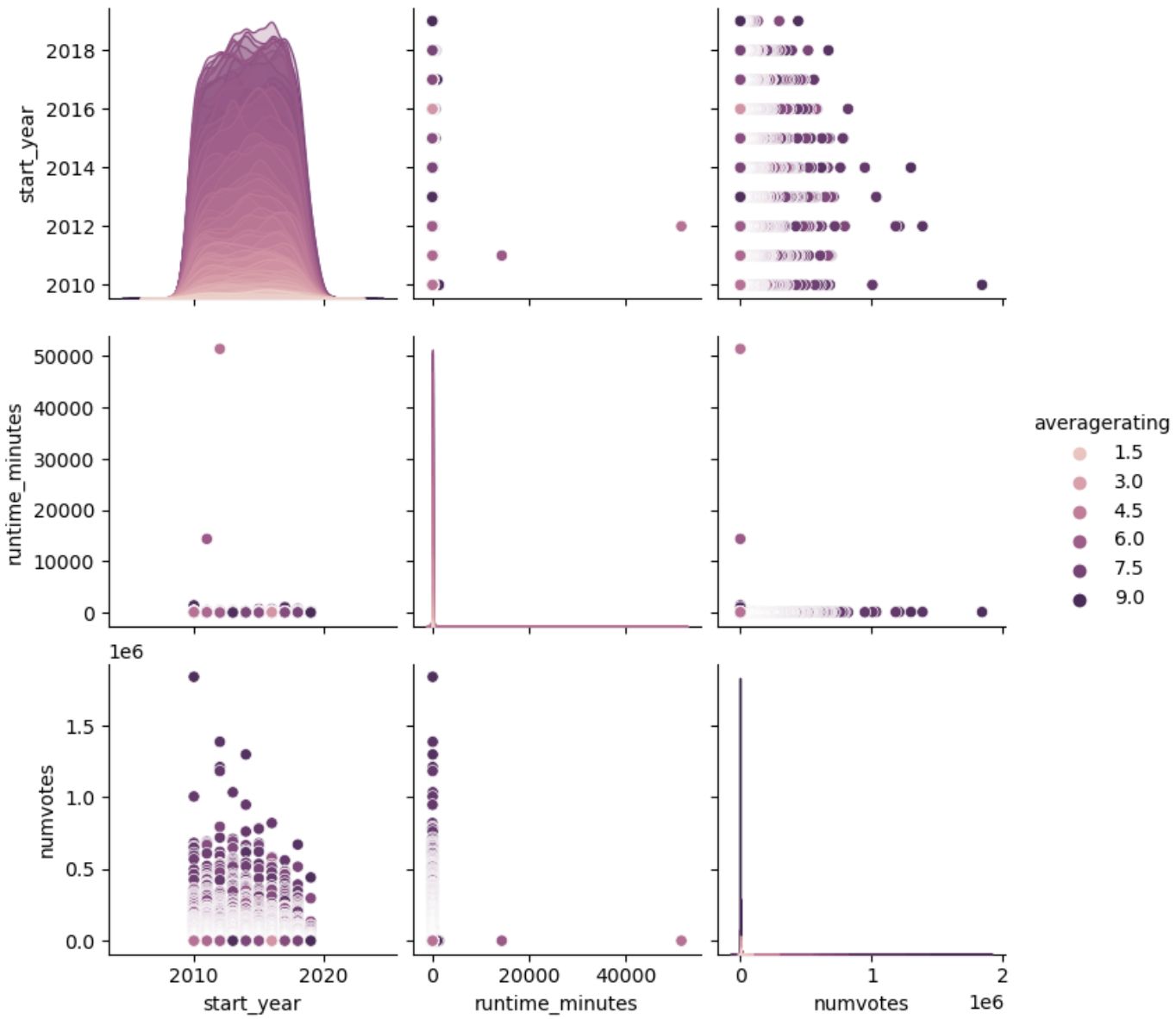


There's a weak positive correlation between average rating (averagerating) and number of votes (numvotes). This correlation (0.044) indicates that as the number of votes per movie increases, the average rating of the movie also increases, although the relationship is very weak.

Multivariate Analysis

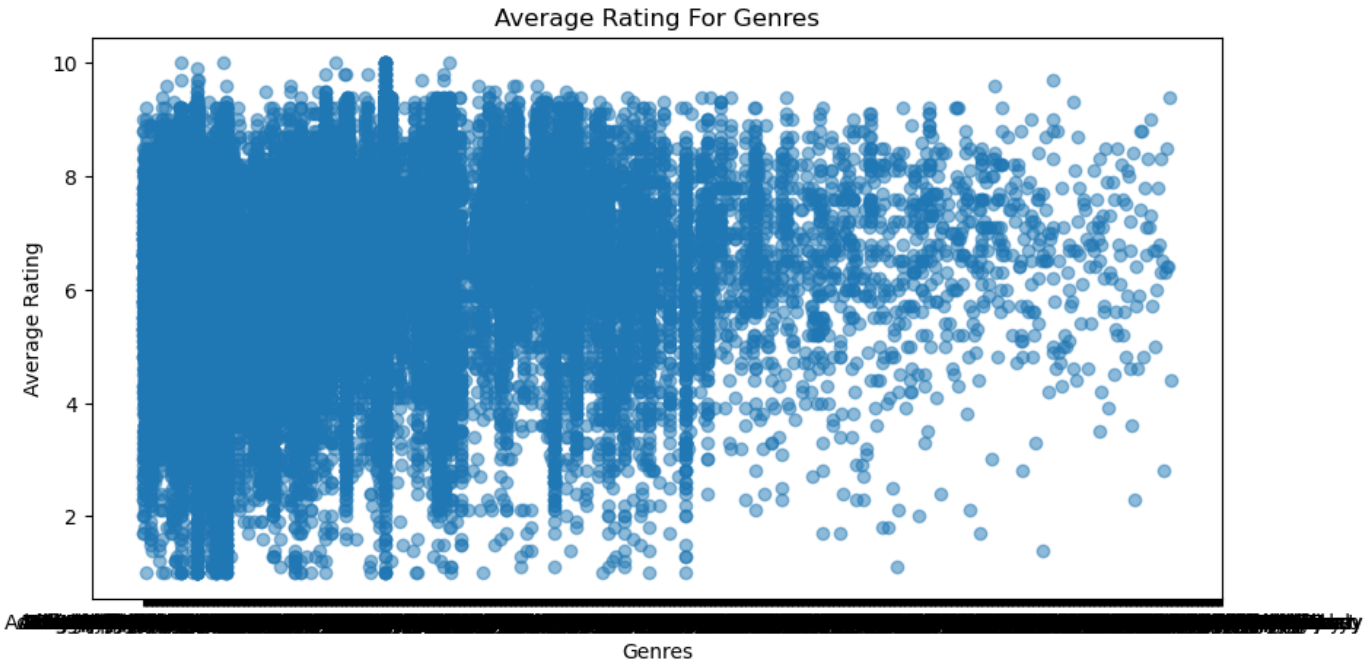
Multivariate analysis helps to understand the relationship between more than two variables in the dataset.

In [40]:
```
# Pairplot to visualize the distribution of the data and relationships between variables
sns.pairplot(data, hue='averagerating')
plt.show()
```



The pairplot shows that movies that started in 2010 have a higher average rating than movies that started in 2020. Additionally, one could observe that the average rating for number of votes is spread out more than the rating for runtime.

In [41]:
```
# Scatterplot to visualize the relationship between average rating and genres
plt.figure(figsize=(10,5))
plt.scatter(x=data['genres'], y=data['averagerating'], alpha=0.5)
plt.title('Average Rating For Genres')
plt.xlabel('Genres')
plt.ylabel('Average Rating')
plt.show()
```



From the visualisation, we can see that there's no correlation between the genres and the average rating.

Conclusions

This analysis yields to three recommendations that would help Microsoft to decide what type of films to create.

- **Invest in making movie genres that reach a bigger audience**

Investing in films/movies like Drama, Comedy, Action and Documentary will ensure high number of audience as they are the ones with the highest number of votes and ratings from our data. Identify target audiences and demographics to tailor content and marketing efforts. Also, evaluate market trends, including IMDB performance, streaming services like NETFLIX and viewer preferences.

- **Average Runtime**

The average runtime for the movies is 87 minutes. Top rated movies are longer around 140 - 160 minutes. I'ld recommend aiming for about a 2hours movies as that appears to be the right duration for even movies in cinemas.

- **Sustainable Practices**

Embrace sustainable filmmaking practices to reduce environmental impact and save on production costs. Examine revenue and profit margins for various types of movies and assess the cost-effectiveness of marketing and distribution strategies.

Next Steps

Further analyses could yield additional insights to further help the head of Microsoft's new movie studio in making decision on the type of movies to produce.

- **Audience engagement through social media**

Utilize social media analytics to understand audience sentiment and engagement. Investigate the use of audience data for personalized recommendations.

- **Content production**

Evaluate the performance of directors, actors, and creative teams. Analyze the relationship between production budgets and box office success.

- **Assessing risks and challenges**

Analyze potential risks and challenges, such as pandemic impacts, legal disputes, and content controversies and develop risk mitigation strategies.

In [ ]: