

Final Project Submission

Student Information

- Student name: Silah Rugut
- Student pace: Part Time
- Scheduled project review date/time: 5th Nov 2023
- Instructor name: Samwel G. Mwangi
- Blog post URL: N/A

Business Problem

Microsoft is considering venturing into the movie industry as all the big companies are already creating original video content. It intends to create a successful movie studio and it's major problem is the lack of knowledge in this field. It wants to understand the current trends in the film industry and make informed decisions on the types of movies to create for maximum success.

To help Microsoft solve this problem, i will consider:

1. Which are the years with the highest number of movies produced?
2. What is the relationship between the production budget and the worldwide profits over time?
3. What genre was highly produced?
4. Which genres are the most profitable?
5. What is the general trend of the average profits over the years?
6. What are the highly rated movie genres?
7. What is the relationship between the production budget and profits?
8. What is the relationship between movie ratings and profits?
9. What were the top 10 highly rated movie titles by revenue?

Table of Contents:

- Introduction
- Data Understanding
- Data Sources
- Data Wrangling
- Exploratory Data Analysis
- Results/Findings
- Conclusion & Recommendations
- The Next Steps

Introduction

This report is an analysis of IMDB & Box Office Mojo data set containing information about 3017 movies produced between years 2010 and 2019.

Business Understanding

The stated business problem presented by Microsoft is establishing their own movie studio to compete within the movie market, and needing to know what kind of movies will be the most successful.

This analysis aims at solving the stated business problem by determining what kind of movies have been most successful in terms of - average rating and profits from the year 2000 to 2018. In utilizing three large datasets from movie giants IMDb, The movie database and Box Office Mojo.

Data Sources

In this project, I will analyse movie data from the below sites

- Box Office Mojo
- IMDB
- Rotten Tomatoes
- TheMovieDB.org

The Specific files for analysis are:

- imdb.title.basics.csv.gz
- imdb.title.ratings.csv.gz
- tn_movie_budgets.csv.gz

Data Wrangling

In this section, I merge the above files to come up with a single dataframe that I can now use to perform exploratory data analysis.

```
In [435]: ▶ # Importing the necessary libraries  
import pandas as pd  
import numpy as np  
import matplotlib.pyplot as plt  
%matplotlib inline  
import seaborn as sns
```

```
In [436]: ▶ imdb_title_basics_df = pd.read_csv('./zippedData/imdb.title.basics.csv')
imdb_title_basics_df.head()
```

Out[436]:

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

```
In [437]: ▶ imdb_title_basics_df.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 [438]: ▶ imdb_title_ratings_df = pd.read_csv('./zippedData/imdb.title.ratings.csv')
imdb_title_ratings_df.head()
```

Out[438]:

	tconst	average_rating	num_votes
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 [439]: `imdb_title_ratings_df.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 [440]: `# Merging the title_ratings and title basics dfs`
`new_title_ratings_df = pd.merge(imdb_title_basics_df, imdb_title_ratings_df, on='tconst')`
`new_title_ratings_df.head()`

Out[440]:

	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 [441]: `new_title_ratings_df.info()`

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 73856 entries, 0 to 73855
Data columns (total 8 columns):
#   Column          Non-Null Count  Dtype
---  -
0   tconst           73856 non-null  object
1   primary_title    73856 non-null  object
2   original_title   73856 non-null  object
3   start_year       73856 non-null  int64
4   runtime_minutes  66236 non-null  float64
5   genres           73052 non-null  object
6   averagerating    73856 non-null  float64
7   numvotes         73856 non-null  int64
dtypes: float64(2), int64(2), object(4)
memory usage: 5.1+ MB
```

```
In [443]: # Dropping the columns that are not relevant in answering my problem statement
new_title_ratings_df.drop(columns=['original_title', 'runtime_minutes'], inplace=True)
new_title_ratings_df.head()
```

Out[443]:

	tconst	title	start_year	genres	averagerating
0	tt0063540	Sunghursh	2013	Action, Crime, Drama	7.0
1	tt0066787	One Day Before the Rainy Season	2019	Biography, Drama	7.2
2	tt0069049	The Other Side of the Wind	2018	Drama	6.9
3	tt0069204	Sabse Bada Sukh	2018	Comedy, Drama	6.1
4	tt0100275	The Wandering Soap Opera	2017	Comedy, Drama, Fantasy	6.5

```
In [444]: # Renaming the primary_title column to title since the primary title is not relevant
new_title_ratings_df.rename(columns={'primary_title': 'title'}, inplace=True)
new_title_ratings_df.head()
```

Out[444]:

	tconst	title	start_year	genres	averagerating
0	tt0063540	Sunghursh	2013	Action, Crime, Drama	7.0
1	tt0066787	One Day Before the Rainy Season	2019	Biography, Drama	7.2
2	tt0069049	The Other Side of the Wind	2018	Drama	6.9
3	tt0069204	Sabse Bada Sukh	2018	Comedy, Drama	6.1
4	tt0100275	The Wandering Soap Opera	2017	Comedy, Drama, Fantasy	6.5

```
In [445]: new_title_ratings_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 73856 entries, 0 to 73855
Data columns (total 5 columns):
#   Column          Non-Null Count  Dtype
---  -
0   tconst          73856 non-null  object
1   title           73856 non-null  object
2   start_year      73856 non-null  int64
3   genres          73052 non-null  object
4   averagerating   73856 non-null  float64
dtypes: float64(1), int64(1), object(3)
memory usage: 3.4+ MB
```

```
In [446]: # Introducing the movie budgets file to provide us with the production l
tn_movie_budgets_df = pd.read_csv('./zippedData/tn.movie_budgets.csv.gz')
tn_movie_budgets_df.head()
```

Out[446]:

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross
0	1	Dec 18, 2009	Avatar	\$425,000,000	\$760,507,625	\$2,776,345,279
1	2	May 20, 2011	Pirates of the Caribbean: On Stranger Tides	\$410,600,000	\$241,063,875	\$1,045,663,875
2	3	Jun 7, 2019	Dark Phoenix	\$350,000,000	\$42,762,350	\$149,762,350
3	4	May 1, 2015	Avengers: Age of Ultron	\$330,600,000	\$459,005,868	\$1,403,013,963
4	5	Dec 15, 2017	Star Wars Ep. VIII: The Last Jedi	\$317,000,000	\$620,181,382	\$1,316,721,747

```
In [447]: # Renaming the movie column to title to facilitate inner merging
tn_movie_budgets_df.rename(columns={'movie': 'title'}, inplace=True)
tn_movie_budgets_df.head()
```

Out[447]:

	id	release_date	title	production_budget	domestic_gross	worldwide_gross
0	1	Dec 18, 2009	Avatar	\$425,000,000	\$760,507,625	\$2,776,345,279
1	2	May 20, 2011	Pirates of the Caribbean: On Stranger Tides	\$410,600,000	\$241,063,875	\$1,045,663,875
2	3	Jun 7, 2019	Dark Phoenix	\$350,000,000	\$42,762,350	\$149,762,350
3	4	May 1, 2015	Avengers: Age of Ultron	\$330,600,000	\$459,005,868	\$1,403,013,963
4	5	Dec 15, 2017	Star Wars Ep. VIII: The Last Jedi	\$317,000,000	\$620,181,382	\$1,316,721,747

In [448]: `tn_movie_budgets_df.info()`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5782 entries, 0 to 5781
Data columns (total 6 columns):
#   Column                Non-Null Count  Dtype
---  -
0   id                    5782 non-null   int64
1   release_date          5782 non-null   object
2   title                 5782 non-null   object
3   production_budget     5782 non-null   object
4   domestic_gross        5782 non-null   object
5   worldwide_gross       5782 non-null   object
dtypes: int64(1), object(5)
memory usage: 271.2+ KB
```

In [449]: `# Merging the new_title_ratings with the tn_movie_budgets_df to improve data = pd.merge(new_title_ratings_df, tn_movie_budgets_df, how='inner', data.head())`

Out[449]:

	tconst	title	start_year	genres	averagerating	id	release_
0	tt0249516	Foodfight!	2012	Action,Animation,Comedy	1.9	26	Dec 31, :
1	tt0326592	The Overnight	2010	NaN	7.5	21	Jun 19, :
2	tt3844362	The Overnight	2015	Comedy,Mystery	6.1	21	Jun 19, :
3	tt0337692	On the Road	2012	Adventure,Drama,Romance	6.1	17	Mar 22, :
4	tt4339118	On the Road	2014	Drama	6.0	17	Mar 22, :

In [450]: `data.info()`

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2875 entries, 0 to 2874
Data columns (total 10 columns):
#   Column                Non-Null Count  Dtype
---  -
0   tconst                2875 non-null   object
1   title                 2875 non-null   object
2   start_year            2875 non-null   int64
3   genres                2867 non-null   object
4   averagerating         2875 non-null   float64
5   id                    2875 non-null   int64
6   release_date          2875 non-null   object
7   production_budget     2875 non-null   object
8   domestic_gross        2875 non-null   object
9   worldwide_gross       2875 non-null   object
dtypes: float64(1), int64(2), object(7)
memory usage: 247.1+ KB
```

Data Cleaning

```
In [451]: # Dropping the id & tconstruct columns as they are not useful in our analysis
data.drop('id', axis=1, inplace=True)
data.drop('tconst', axis=1, inplace=True)
```

```
In [452]: # Creating a new column using the release date called release year
data['release_date'] = pd.to_datetime(data['release_date'])
data['release_year'] = data['release_date'].dt.year
data.head()
```

Out[452]:

	title	start_year	genres	averagerating	release_date	production_budget
0	Foodfight!	2012	Action,Animation,Comedy	1.9	2012-12-31	\$4
1	The Overnight	2010	NaN	7.5	2015-06-19	
2	The Overnight	2015	Comedy,Mystery	6.1	2015-06-19	
3	On the Road	2012	Adventure,Drama,Romance	6.1	2013-03-22	\$2
4	On the Road	2014	Drama	6.0	2013-03-22	\$2

```
In [453]: # Converting production_budget, domestic_gross , worldwide_gross to numeric
# Remove '$' and convert to numeric for 'production_budget'
data['production_budget'] = pd.to_numeric(data['production_budget'].replace('$', ''))

# Remove '$' and convert to numeric for 'domestic_gross' and 'worldwide_gross'
data['domestic_gross'] = pd.to_numeric(data['domestic_gross'].replace('$', ''))
data['worldwide_gross'] = pd.to_numeric(data['worldwide_gross'].replace('$', ''))
```

```
In [454]: # Creating new columns domestic profits and worldwide profits
data['domestic_profits'] = data['domestic_gross'] - data['production_budget']

# Calculating the worldwide Profits
data['worldwide_profits'] = data['worldwide_gross'] - data['production_budget']
```


In [455]: `data.head()`

Out[455]:

	title	start_year	genres	averagerating	release_date	productio
0	Foodfight!	2012	Action,Animation,Comedy	1.9	2012-12-31	
1	The Overnight	2010	NaN	7.5	2015-06-19	
2	The Overnight	2015	Comedy,Mystery	6.1	2015-06-19	
3	On the Road	2012	Adventure,Drama,Romance	6.1	2013-03-22	
4	On the Road	2014	Drama	6.0	2013-03-22	

In [456]:

```
# Since the values for profits and budget are huge, we divide all by 1m
data['domestic_gross_in_mill'] = data['domestic_gross'] / 10**6
data['worldwide_gross_in_mill'] = data['worldwide_gross'] / 10**6
data['production_budget_in_mill'] = data['production_budget'] / 10**6
data['domestic_profits_in_mill'] = data['domestic_profits'] / 10**6
data['worldwide_profits_in_mill'] = data['worldwide_profits'] / 10**6
data.head()
```

Out[456]:

	title	start_year	genres	averagerating	release_date	productio
0	Foodfight!	2012	Action,Animation,Comedy	1.9	2012-12-31	
1	The Overnight	2010	NaN	7.5	2015-06-19	
2	The Overnight	2015	Comedy,Mystery	6.1	2015-06-19	
3	On the Road	2012	Adventure,Drama,Romance	6.1	2013-03-22	
4	On the Road	2014	Drama	6.0	2013-03-22	

```
In [457]: # Removing the columns already reproduced.
data.drop(columns=['production_budget', 'domestic_gross', 'worldwide_gross'], inplace=True)
data.head()
```

Out[457]:

	title	start_year	genres	averagerating	release_date	release_y
0	Foodfight!	2012	Action,Animation,Comedy	1.9	2012-12-31	2
1	The Overnight	2010	NaN	7.5	2015-06-19	2
2	The Overnight	2015	Comedy,Mystery	6.1	2015-06-19	2
3	On the Road	2012	Adventure,Drama,Romance	6.1	2013-03-22	2
4	On the Road	2014	Drama	6.0	2013-03-22	2

```
In [458]: # Checking for duplicates
duplicates = data.duplicated()
duplicates
```

Out[458]:

```
0      False
1      False
2      False
3      False
4      False
...
2870    False
2871    False
2872    False
2873    False
2874    False
Length: 2875, dtype: bool
```

```
In [459]: # To verify whether there are any duplicates in the dataframe
data[data.duplicated(keep=False)]
```

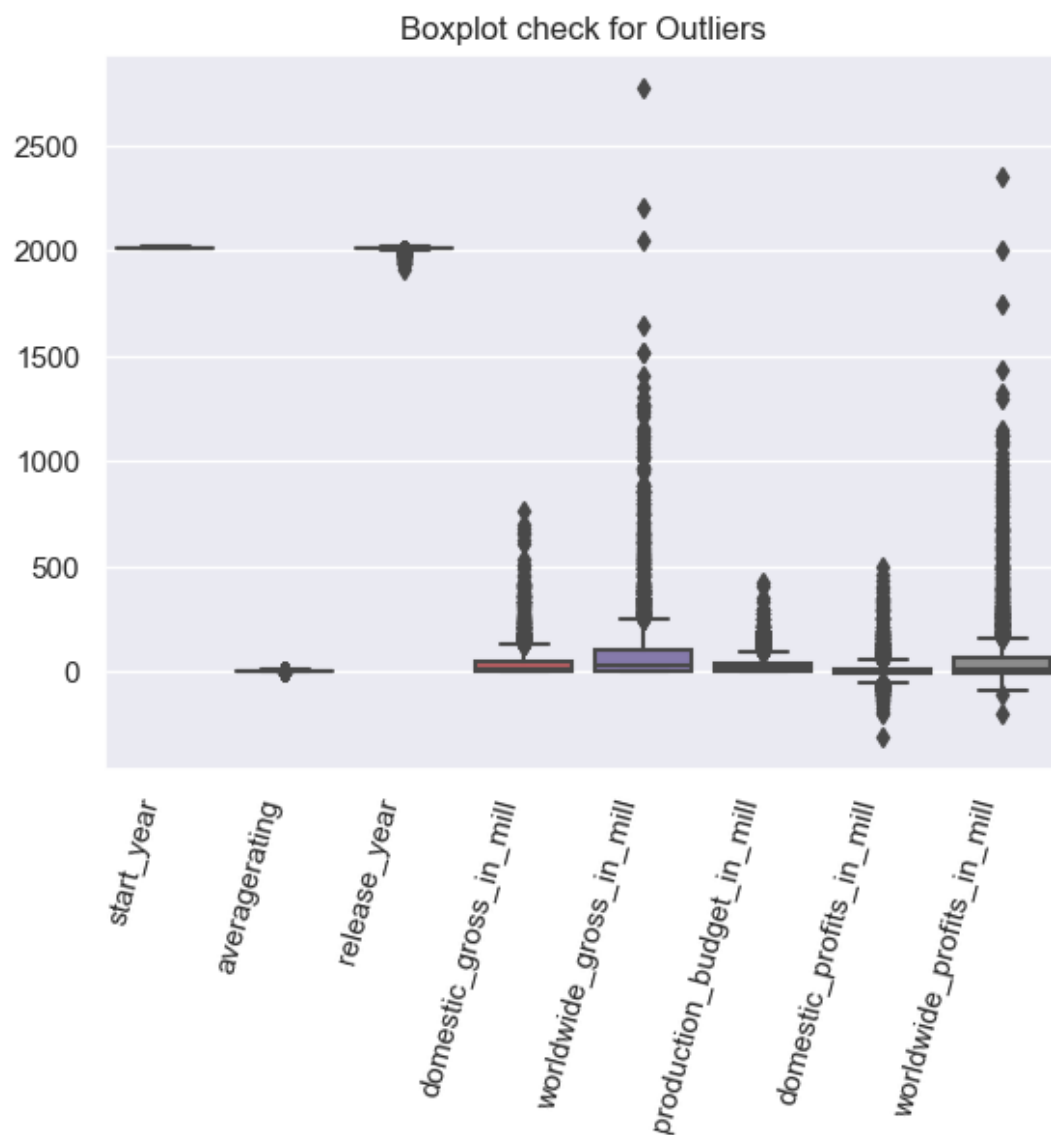
Out[459]:

	title	start_year	genres	averagerating	release_date	release_year	domestic_gross_in_
--	-------	------------	--------	---------------	--------------	--------------	--------------------

```
In [460]: # Checking for null values in each column
data.isna().sum()
```

```
Out[460]: title                                0
start_year                                   0
genres                                       8
averagerating                              0
release_date                               0
release_year                               0
domestic_gross_in_mill                     0
worldwide_gross_in_mill                   0
production_budget_in_mill                 0
domestic_profits_in_mill                  0
worldwide_profits_in_mill                 0
dtype: int64
```

```
In [461]: # Checking for outliers
sns.boxplot(data=data)
sns.set(style="darkgrid")
plt.xticks(rotation=75, ha='right')
plt.title('Boxplot check for Outliers')
plt.show()
```



```
In [462]: ▶ # Checking the release years of our dataset in ascending order
sorted_release_years = data['release_year'].sort_values(ascending=True)
print(sorted_release_years)
```

2551	1915
1890	1927
1111	1940
2416	1940
1107	1940
	...
1998	2019
1346	2019
2489	2019
2025	2019
2611	2019

Name: release_year, Length: 2875, dtype: int64

```
In [463]: ▶ # Limiting our data to movies released between 2000 and before 1st Jan 2019
start_date = '2000-01-01'
end_date = '2019-01-01'
df = data[(data['release_date'] >= start_date) & (data['release_date'] < end_date)]
```

```
In [464]: ▶ # Double checking to ensure that the data has been sliced between years
sorted_release_years = df['release_year'].sort_values(ascending=True)
print(sorted_release_years)
```

2874	2000
548	2000
549	2000
1542	2000
2543	2000
	...
2693	2018
426	2018
1823	2018
959	2018
1899	2018

Name: release_year, Length: 2553, dtype: int64

```
In [ ]: ▶ # Converting release_year to string datatype
df['release_year'] = df['release_year'].astype(str)
```

In [468]: `df.info()`

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2553 entries, 0 to 2874
Data columns (total 11 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   title                                2553 non-null   object
1   start_year                           2553 non-null   int64
2   genres                               2545 non-null   object
3   averagerating                        2553 non-null   float64
4   release_date                         2553 non-null   datetime64[ns]
5   release_year                         2553 non-null   object
6   domestic_gross_in_mill               2553 non-null   float64
7   worldwide_gross_in_mill              2553 non-null   float64
8   production_budget_in_mill            2553 non-null   float64
9   domestic_profits_in_mill             2553 non-null   float64
10  worldwide_profits_in_mill            2553 non-null   float64
dtypes: datetime64[ns](1), float64(6), int64(1), object(3)
memory usage: 239.3+ KB
```

Exploratory Data Analysis

In [470]: `# Getting a summary to understand the data`
`df.describe()`

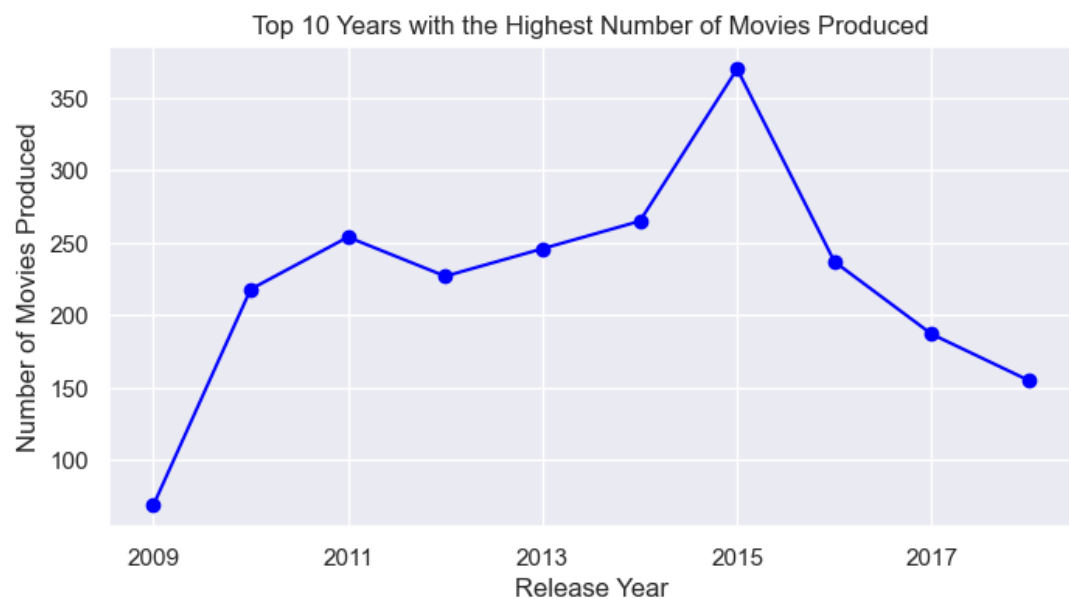
Out[470]:

	start_year	averagerating	domestic_gross_in_mill	worldwide_gross_in_mill	proc
count	2553.000000	2553.000000	2553.000000	2553.000000	
mean	2013.809244	6.254524	42.849598	105.165461	
std	2.491215	1.175536	75.455290	207.079844	
min	2010.000000	1.600000	0.000000	0.000000	
25%	2012.000000	5.600000	0.307631	1.642939	
50%	2014.000000	6.400000	14.677674	30.063805	
75%	2016.000000	7.100000	51.100486	101.379287	
max	2019.000000	9.300000	760.507625	2776.345279	

```
In [471]: # 1. Which are the years with the highest number of movies produced?
# The top 10 years with the highest number of movies produced
top_10_years = df['release_year'].value_counts().nlargest(10)
print(top_10_years)
```

```
2015    370
2014    265
2011    254
2013    246
2016    237
2012    227
2010    218
2017    187
2018    155
2009     69
Name: release_year, dtype: int64
```

```
In [472]: plt.figure(figsize=(8, 4))
top_10_years.sort_index().plot(kind='line', marker='o', color='blue')
plt.title('Top 10 Years with the Highest Number of Movies Produced')
plt.xlabel('Release Year')
plt.ylabel('Number of Movies Produced')
plt.show()
```



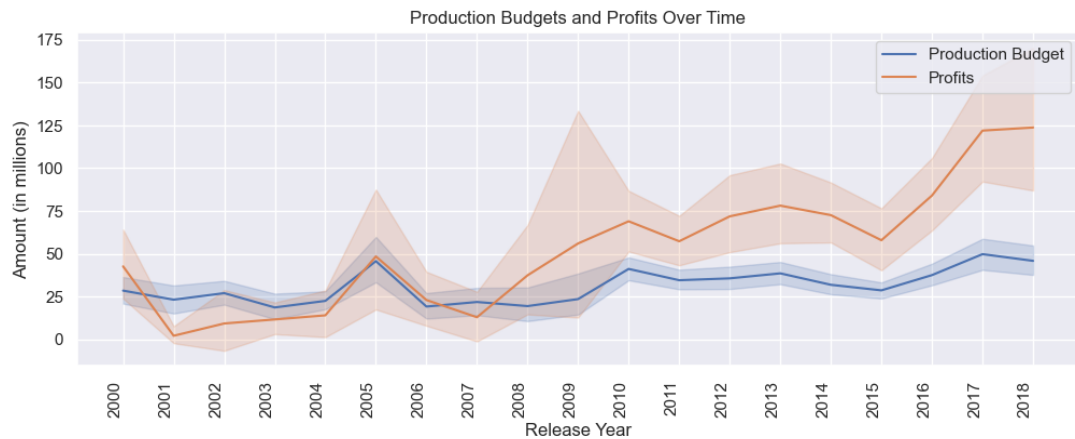
```
In [473]: df.columns
```

```
Out[473]: Index(['title', 'start_year', 'genres', 'averagerating', 'release_date',
               'release_year', 'domestic_gross_in_mill', 'worldwide_gross_in_mill',
               'production_budget_in_mill', 'domestic_profits_in_mill',
               'worldwide_profits_in_mill'],
              dtype='object')
```

In [474]: **▶** *# 2. What is the relationship between the production budget and the wor*

```
df_sorted = df.sort_values('release_year')

plt.figure(figsize=(12, 4))
sns.lineplot(x='release_year', y='production_budget_in_mill', data=df_s
sns.lineplot(x='release_year', y='worldwide_profits_in_mill', data=df_s
plt.title('Production Budgets and Profits Over Time')
plt.xlabel('Release Year')
plt.ylabel('Amount (in millions)')
plt.xticks(rotation=90, ha='right')
plt.legend()
plt.show()
```



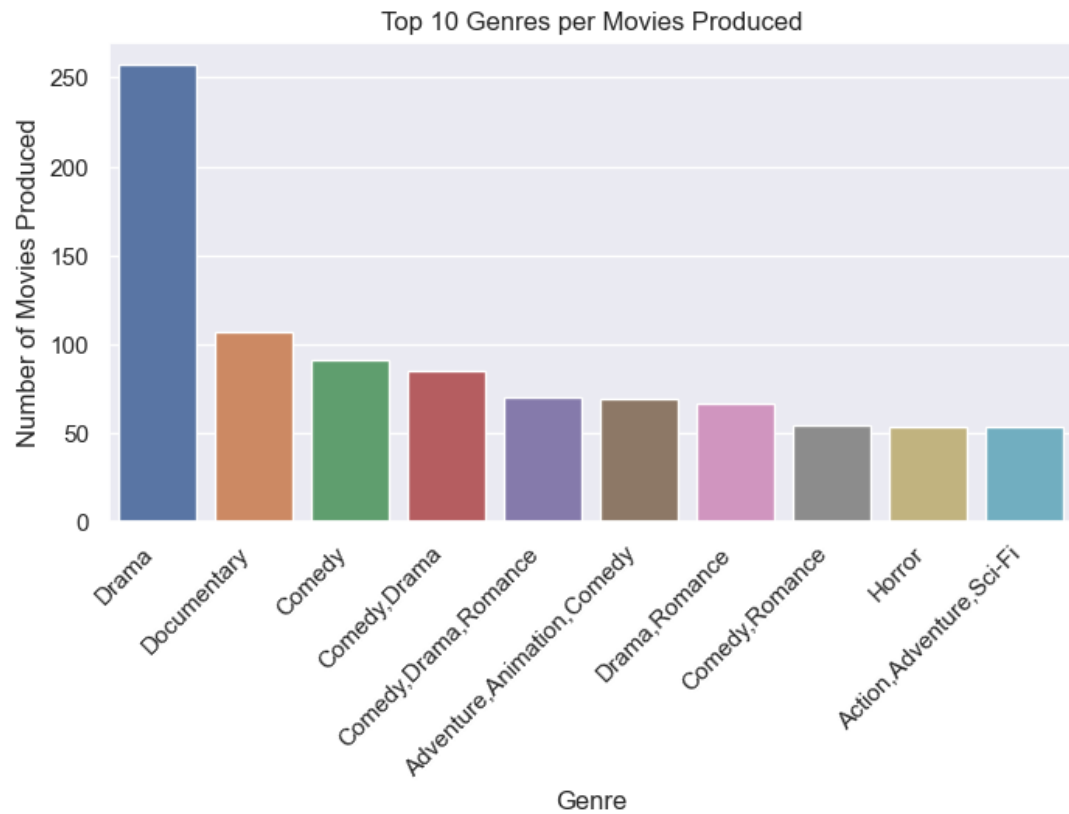
In [475]: **▶** *# 3.What genre was highly produced.*

```
genre_counts = df['genres'].value_counts().sort_values(ascending=False)
genre_counts.head(10)
```

```
Out[475]: Drama                257
Documentary                 107
Comedy                      91
Comedy,Drama                85
Comedy,Drama,Romance        70
Adventure,Animation,Comedy  69
Drama,Romance                66
Comedy,Romance               54
Action,Adventure,Sci-Fi      53
Horror                       53
Name: genres, dtype: int64
```

```
In [476]: top_genres = df['genres'].value_counts().nlargest(10)
```

```
plt.figure(figsize=(8, 4))
sns.barplot(x=top_genres.index, y=top_genres.values,)
plt.title('Top 10 Genres per Movies Produced')
plt.xlabel('Genre')
plt.ylabel('Number of Movies Produced')
plt.xticks(rotation=45, ha='right')
plt.show()
```



```
In [477]: df.columns
```

```
Out[477]: Index(['title', 'start_year', 'genres', 'averagerating', 'release_date',
               'release_year', 'domestic_gross_in_mill', 'worldwide_gross_in_mill',
               'production_budget_in_mill', 'domestic_profits_in_mill',
               'worldwide_profits_in_mill'],
              dtype='object')
```

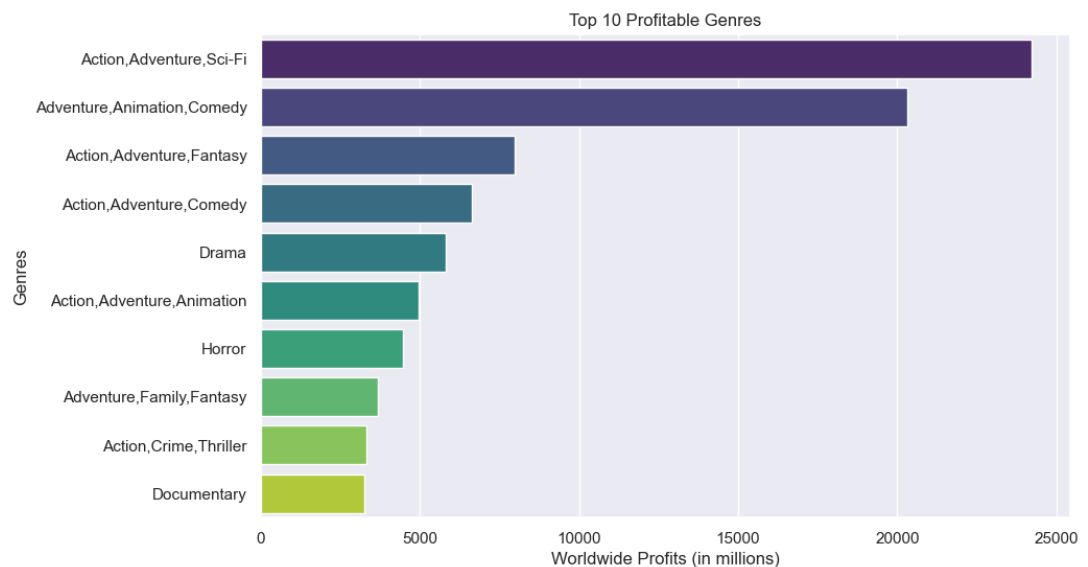


```
In [478]: # 4. Which genres are the most profitable
top_10_profitable_genres = df.groupby('genres')['worldwide_profits_in_mill'].max()
top_10_profitable_genres
```

```
Out[478]: genres
Action,Adventure,Sci-Fi      24213.766663
Adventure,Animation,Comedy    20321.014565
Action,Adventure,Fantasy      7986.155783
Action,Adventure,Comedy       6635.558760
Drama                        5818.192862
Action,Adventure,Animation    4973.873272
Horror                        4477.882968
Adventure,Family,Fantasy      3666.994251
Action,Crime,Thriller         3334.094820
Documentary                   3268.306807
Name: worldwide_profits_in_mill, dtype: float64
```

```
In [479]: plt.figure(figsize=(10, 6))
sns.barplot(x=top_10_profitable_genres.values, y=top_10_profitable_genres.index)
plt.title('Top 10 Profitable Genres')
plt.xlabel('Worldwide Profits (in millions)')
plt.ylabel('Genres')

plt.show()
```



```
In [480]: df.columns
```

```
Out[480]: Index(['title', 'start_year', 'genres', 'averagerating', 'release_date',
               'release_year', 'domestic_gross_in_mill', 'worldwide_gross_in_mill',
               'production_budget_in_mill', 'domestic_profits_in_mill',
               'worldwide_profits_in_mill'],
              dtype='object')
```

```
In [400]: # 5. What is the general trend of the average profits over the years
average_gross_by_year = df.groupby('release_year')['worldwide_profits_in_mill'].mean()
average_gross_by_year
```

```
Out[400]: release_year
2000      42.592318
2001       2.030446
2002       9.213339
2003      11.572078
2004      13.972990
2005      48.396859
2006      22.922621
2007      12.854817
2008      37.357367
2009      55.980328
2010      68.903208
2011      57.263991
2012      71.822146
2013      78.083481
2014      72.534761
2015      57.790808
2016      84.268362
2017     121.941726
2018     124.621112
```

```
In [481]: plt.figure(figsize=(12, 4))
sns.lineplot(x=average_gross_by_year.index, y=average_gross_by_year.values)
plt.title('Average Worldwide Profits Over the Years')
plt.xlabel('Release Year')
plt.ylabel('worldwide_profits_in_mill')
plt.show()
```

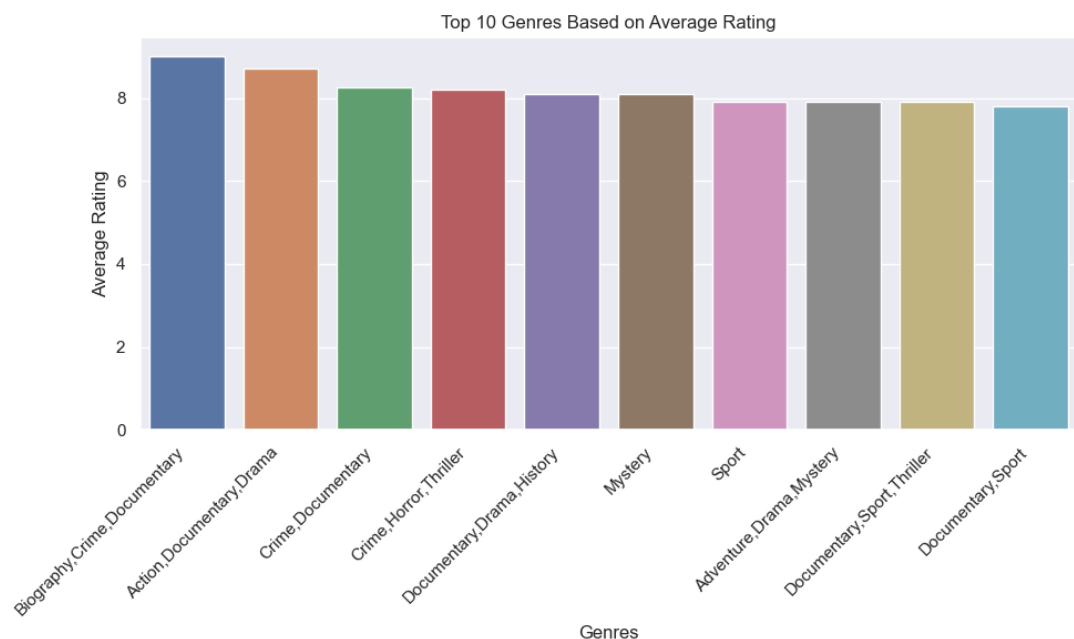


```
In [482]: # 6. What are the highly rated movie genres?
genre_ratings = df.groupby('genres')['averagerating'].mean()
sorted_genres = genre_ratings.sort_values(ascending=False)
top_10_genres = sorted_genres.head(10)
top_10_genres
```

```
Out[482]: genres
Biography,Crime,Documentary    9.00
Action,Documentary,Drama      8.70
Crime,Documentary             8.25
Crime,Horror,Thriller         8.20
Documentary,Drama,History     8.10
Mystery                      8.10
Sport                        7.90
Adventure,Drama,Mystery       7.90
Documentary,Sport,Thriller     7.90
Documentary,Sport             7.80
Name: averagerating, dtype: float64
```

```
In [483]: df_top_10_genres = top_10_genres.reset_index(name='average_rating')

plt.figure(figsize=(10, 6))
sns.barplot(x='genres', y='average_rating', data=df_top_10_genres)
plt.title('Top 10 Genres Based on Average Rating')
plt.xlabel('Genres')
plt.ylabel('Average Rating')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



```
In [484]: df.columns
```

```
Out[484]: Index(['title', 'start_year', 'genres', 'averagerating', 'release_date',
               'release_year', 'domestic_gross_in_mill', 'worldwide_gross_in_mill',
               'production_budget_in_mill', 'domestic_profits_in_mill',
               'worldwide_profits_in_mill'],
              dtype='object')
```

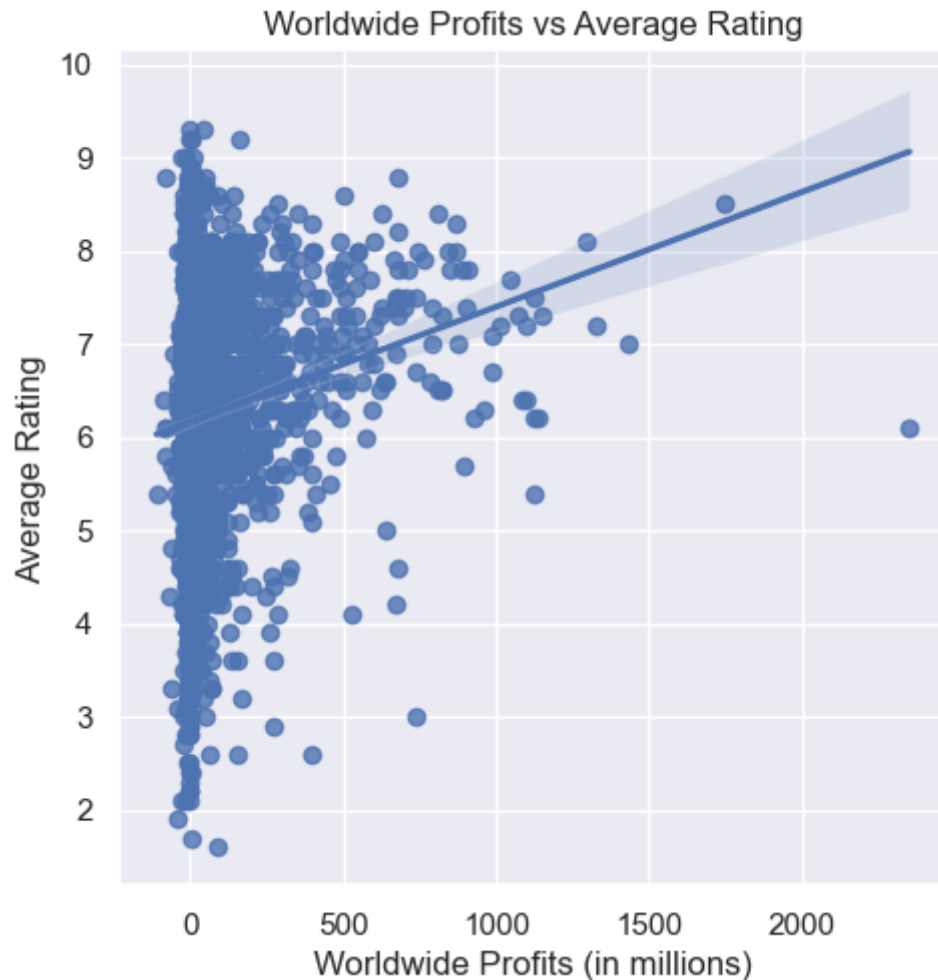
In [485]: **▶** *# 7. What is the relationship between the production budget and profits*

```
sns.lmplot(x='production_budget_in_mill', y='worldwide_profits_in_mill')
plt.title('Production Budget vs Worldwide Profits')
plt.xlabel('Production Budget (in millions)')
plt.ylabel('Worldwide Profits (in millions)')
plt.show()
```



In [486]: `# 8. What is the relationship between movie ratings and profits`

```
sns.lmplot(x='worldwide_profits_in_mill', y='averagerating', data=df)
plt.title('Worldwide Profits vs Average Rating')
plt.xlabel('Worldwide Profits (in millions)')
plt.ylabel('Average Rating')
plt.show()
```



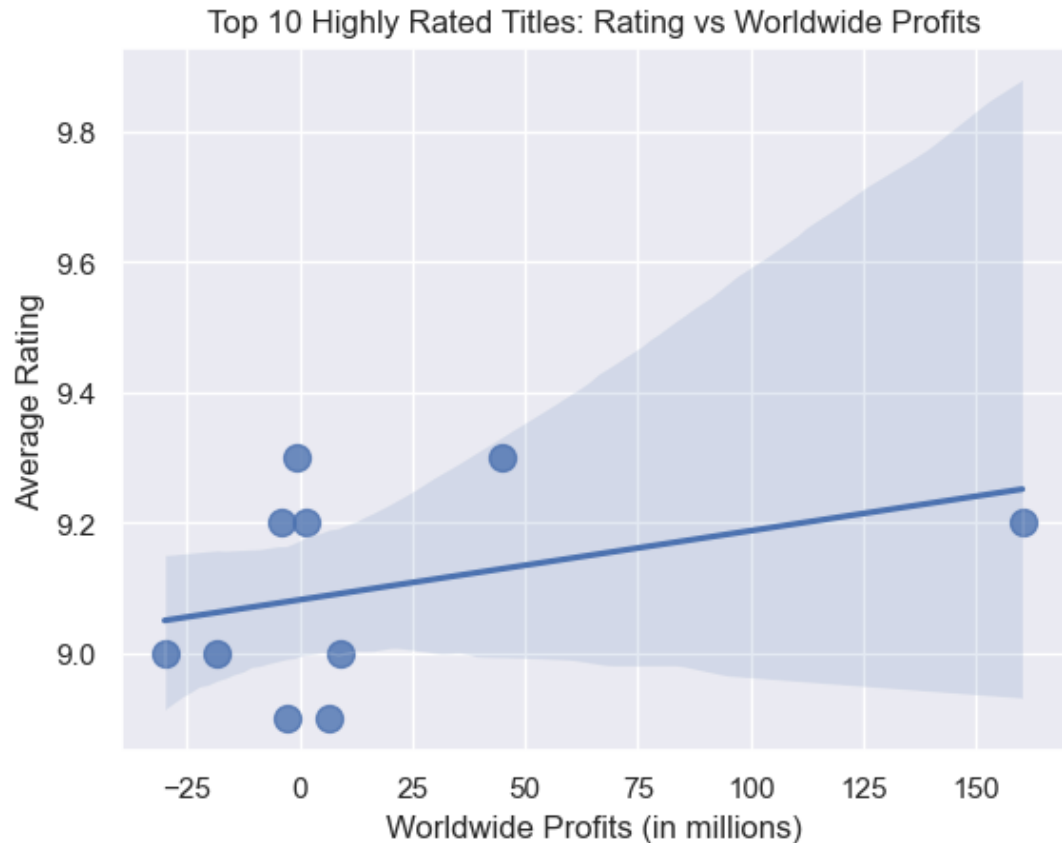
In [487]: `# 9. What were the top 10 highly rated movie titles by revenue?`
`top_10_highly_rated = df.nlargest(10, 'averagerating')[['title', 'worldwide_profits_in_mill']`
`print("Top 10 Highly Rated Titles by Average Rating and Worldwide Profits")`
`print(top_10_highly_rated)`

Top 10 Highly Rated Titles by Average Rating and Worldwide Profits:

	title	worldwide_profits_in_mill
1393	Crossroads	45.000000
1394	Crossroads	-0.500000
193	The Runaways	-4.221368
684	The Wall	1.495262
1343	Traffic	160.300000
1804	Survivor	-18.296719
1921	Frailty	8.947280
2065	Dragonfly	-29.936195
1505	Dark Blue	-2.737935
2614	Bobby	6.597806

```
In [488]: top_10_highly_rated = df.nlargest(10, 'averagerating')

# Regression plot
sns.regplot(x='worldwide_profits_in_mill', y='averagerating', data=top_10_highly_rated)
plt.title('Top 10 Highly Rated Titles: Rating vs Worldwide Profits')
plt.xlabel('Worldwide Profits (in millions)')
plt.ylabel('Average Rating')
plt.show()
```



Results

The stated business problem presented by Microsoft is establishing their own movie studio to compete within the movie market, and needing to know what kind of movies will be the most successful.

This analysis aims at solving the stated business problem by determining what kind of movies have been most successful in terms of - average rating and profits from the year 2000 to 2018. In utilizing three large datasets from movie giants IMDb, The movie database and Box Office Mojo, the data is credible.

The analysis of movies from the year 2000 to 2018 shows the following:

1. Number of Movies produced have been declining over time.
Movie production had been on an upward trajectory in terms of films produced annually until it peaked in the year 2015. However, there has been a steep reduction in the number of movies released from 2015 onwards.
2. The correlation between production budget and profits have been positive. There was a negative correlation between production budget and profits between the year 2000 and 2007. Since then, there has been a positive correlation

3. The highly produced genre is Drama (257) followed by Documentary (107), Comedy and Drama.
4. The most profitable genres on average are Action, Adventure, Sci-fi. Adventure, Animation, Comedy all averaging more than 20 billion dollars.
5. The average profits generated from the movie business has been on a steady growth from 2007 to 2018.
6. The highly audience rated movie genres are Biography, Crime, Documentary followed by Action, Documentary, Drama. Crime, Horror, Thriller also rank in the top 5.
7. There is a high positive correlation between movie ratings and profits.
8. There is a low positive correlation between movie ratings and profits when looking at the highly rated movie title.

Conclusions & Recommendations

This analysis leads to three recommendations of what movies to produce for Microsoft's new Movie studio.

1. To start by producing Action, Adventure, Sci-Fi and Adventure, Animation and Comedy movies to assured sustained profits across the globe. These genres have demonstrated success over 18 years.
2. The profits generated by movie producing companies over the 18 years have been on a steady growth. This means that the movie producing business is a business that is worth the investment
3. The number of movies produced have been declining over time. This means that the movie producing companies have started diversifying to movie streaming sites and therefore Microsoft will have to include a movie streaming website in their budget.

The Next Steps

Further analysis could yield additional insights that would better inform Microsoft in their decision making:

1. Analyze streaming data - The dataset used in this analysis was from the tradition theatre released movies and therefore there is need to analyze data on profits and ratings from streaming websites to better inform the investment decision
2. Movies like Black Panther hit the top charts in the year 2018 as it appealed to a certain demographic. So there is need to study which demographic in terms of gender, race, and language a certain genre appeal to. So Microsoft can tailor its studio to production of movies appealing to diverse groups of people
3. Analyze the most recent data from the year 2019 to 2023. This can bring alot of insight on the trend of movie production revenue given that there was Covid-19 pandemic that likely influenced production of movies and also the consumer behavior.