Type *Markdown* and LaTeX: α^2

Final Project Submission

Please fill out:

- Student name: Wesley Ochiel
- Student pace: Full time
- Scheduled project review date/time:
- Instructor name: 28th May 2023
- Blog post URL:

PROJECT OVERVIEW

This project was initiated by Microsoft Studio, which is a new movie production studio. They requested us to conduct a research, using other databases from other experienced movie production companies such as Box Office Mojo, IMDB, Rotten Tomatoes, The MovieDB and The Number, with the aim of guiding them on producing good movies which will succeed in the future. The fact that this is a new movie studio, they don’t have any knowledge on producing good movies that will succeed in future. So we’ll have to use the imported data, from other production studios, with the aim of determining the movies that they will use to produce other good movies.

The main objective of this project is to analyze the current trends in the film industry by determining what types of films or movies that are currently succeeding at the box office. The findings will be used to give insights in making judgements regarding the type of movies that the Microsoft Company, which is a new studio, will produce. Project goals include the following: -

To investigate the current movie trends, in accordance to the Box Office Productions

To examine the necessary statistics that will be used to determine the movies that are succeeding at the Box Office.

To translate the findings into actionable insights that will be used by Microsoft Studio to make informed decisions to produce good movies.

To provide recommendations on the types of films that are likely to be successful in the future.

BUSINESS UNDERSTANDING

Microsoft Studio has made a decision to launch a new movie studio and enter into the film business industry. They don't have the skills or the expertise to make successful movies that will help them to earn more profits like other production companies at the Box Office. They must comprehend the current trends and tastes of movies, that will be liked by their targeted consumers, in order to make wise selections regarding the kinds of movies to produce.

This project's goal and objective is to investigate the kinds of movies that are currently doing well at the Box Office and offer useful information that Microsoft can use to decide what kinds of movies to make. In order to determine the most popular genres, actors, directors, and other elements that influence a film's industry success, the project will involve evaluating movie data.

The findings that will be made from the analysis will be applied by Microsoft’s new movie studio with the aim of producing good movies and films that have a potential to be successful in the market.

DATA UNDERSTANDING

Data Sources

In this project, we used data from other movie and film production companies, such as Box Office Mojo, IMDB, Rotten Tomatoes and The MovieDB. These data samples were imported into our databases of which we used them to determine the current movies and the genres that are currently doing well in the market. The data contains movie title, the studio production, domestic gross, foreign gross, the year that they were produced and the number of votes and reviews that were reviewed by their targeted consumers after watching them.

Data Quality

During the importation of our data into our database, we encountered some issues like missing data in the datasets while other datasets had duplicates of which I had to first clean the data by removing duplicates and taking care of missing data and this helped me to improve the quality of the data from our sources.

BUSINESS RECOMMENDATIONS

I would like to recommend Microsoft to produce the movie Beautiful Boy[2018] since it has earned the other production companies with over $8.7 million of which this will earn, the new Microsoft studio, more profits as compared to other movies.

I would like to recommend Microsoft studio to produce more of Drama movies and series since they have been performing well over the years and this will attract more audience from all over the world.

I would also like to recommend Microsoft studio to produce other movies like Kid with a bike, Desierto and Once Upon A time in Anatolia since they are likely to do well in the future after having a lot of experiences over the years like other movie production companies.

DATA CLEANING

IMPORTING DATA USING PANDAS

In [4]:

```
import pandas as pd
```

In [5]:

```
df = pd.read_csv("./bom.movie_gross.csv")
df
```

Out[5]:

	title	studio	domestic_gross	foreign_gross	year
0	Toy Story 3	BV	415000000.0	652000000	2010
1	Alice in Wonderland (2010)	BV	334200000.0	691300000	2010
2	Harry Potter and the Deathly Hallows Part 1	WB	296000000.0	664300000	2010
3	Inception	WB	292600000.0	535700000	2010
4	Shrek Forever After	P/DW	238700000.0	513900000	2010
...	...	...	...	...	...
3382	The Quake	Magn.	6200.0	NaN	2018
3383	Edward II (2018 re-release)	FM	4800.0	NaN	2018
3384	El Pacto	Sony	2500.0	NaN	2018
3385	The Swan	Synergetic	2400.0	NaN	2018
3386	An Actor Prepares	Grav.	1700.0	NaN	2018

3387 rows × 5 columns

In [6]:

```
df.shape
```

Out[6]:

```
(3387, 5)
```

GROUPING OF IMPORTED DATA

In [7]:

```
df.groupby('domestic_gross')['year'].mean().head()
```

Out[7]:

```
domestic_gross
100.0    2013.000000
300.0    2015.666667
400.0    2014.500000
500.0    2017.000000
600.0    2012.000000
Name: year, dtype: float64
```

CHECKING FOR DUPLICATE DATA

In [8]:

```
df.duplicated().value_counts()
```

Out[8]:

```
False    3387
Name: count, dtype: int64
```

IDENTIFYING OF NANs

In [9]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3387 entries, 0 to 3386
Data columns (total 5 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   title           3387 non-null   object 
 1   studio          3382 non-null   object 
 2   domestic_gross  3359 non-null   float64
 3   foreign_gross   2037 non-null   object 
 4   year            3387 non-null   int64  
dtypes: float64(1), int64(1), object(3)
memory usage: 132.4+ KB
```

In [10]:

```
df.isna()
```

Out[10]:

	title	studio	domestic_gross	foreign_gross	year
0	False	False	False	False	False
1	False	False	False	False	False
2	False	False	False	False	False
3	False	False	False	False	False
4	False	False	False	False	False
...	...	...	...	...	...
3382	False	False	False	True	False
3383	False	False	False	True	False
3384	False	False	False	True	False
3385	False	False	False	True	False
3386	False	False	False	True	False

3387 rows × 5 columns

In [11]:

```
df .isna().sum()
```

Out[11]:

```
title                0
studio               5
domestic_gross      28
foreign_gross     1350
year                 0
dtype: int64
```

In [12]:

```
for col in df.columns:
    print(col, '\n', df[col].value_counts(normalize=True).head(), '\n\n')
```

```
title
 title
Bluebeard                0.000590
Before We Go             0.000295
Knock Knock              0.000295
Kindergarten Teacher     0.000295
Welcome to Leith         0.000295
Name: proportion, dtype: float64


studio
 studio
IFC      0.049083
Uni.     0.043465
WB       0.041396
Fox      0.040213
Magn.    0.040213
Name: proportion, dtype: float64


domestic_gross
 domestic_gross
1100000.0    0.009527
1000000.0    0.008931
1300000.0    0.008931
1200000.0    0.007443
1400000.0    0.006847
Name: proportion, dtype: float64


foreign_gross
 foreign_gross
1200000    0.011291
1100000    0.006873
4200000    0.005891
1900000    0.005891
1300000    0.005400
Name: proportion, dtype: float64


year
 year
2015    0.132861
2016    0.128727
2012    0.118099
2011    0.117803
2014    0.116622
Name: proportion, dtype: float64
```

In [ ]:

```
Dealing With  Missing  Data
```

In [ ]:

```
df['studio'].value_counts()
```

Out[16]:

```
studio
IFC           166
Uni.          147
WB            140
Fox           136
Magn.         136
             ...
E1              1
PI              1
ELS             1
PalT            1
Synergetic      1
Name: count, Length: 257, dtype: int64
```

In [ ]:

```
# dropping the studio column
df['studio'].fillna(value='IFC', inplace=True)
```

In [ ]:

```
# rechecking the studio column on missing values
df.isna().sum()
```

Out[18]:

```
title                0
studio               0
domestic_gross      28
foreign_gross     1350
year                 0
dtype: int64
```

In [ ]:

```
# replacing the domestic_gross column with mean values
df['domestic_gross'].fillna(df['domestic_gross'].median(), inplace=True)
```

In [ ]:

```
df.isna().sum()
```

Out[20]:

```
title                0
studio               0
domestic_gross       0
foreign_gross     1350
year                 0
dtype: int64
```

In [ ]:

```
# replacing the domestic_gross column with mean values
df.dropna(inplace=True)
```

In [ ]:

```
# rechecking the domestic_gross column on missing values
df.isna().sum()
```

Out[22]:

```
title             0
studio            0
domestic_gross    0
foreign_gross     0
year              0
dtype: int64
```

In [ ]:

```
df
```

Out[28]:

	title	studio	domestic_gross	foreign_gross	year
0	Toy Story 3	BV	415000000.0	652000000	2010
1	Alice in Wonderland (2010)	BV	334200000.0	691300000	2010
2	Harry Potter and the Deathly Hallows Part 1	WB	296000000.0	664300000	2010
3	Inception	WB	292600000.0	535700000	2010
4	Shrek Forever After	P/DW	238700000.0	513900000	2010
...	...	...	...	...	...
3382	The Quake	Magn.	6200.0	NaN	2018
3383	Edward II (2018 re-release)	FM	4800.0	NaN	2018
3384	El Pacto	Sony	2500.0	NaN	2018
3385	The Swan	Synergetic	2400.0	NaN	2018
3386	An Actor Prepares	Grav.	1700.0	NaN	2018

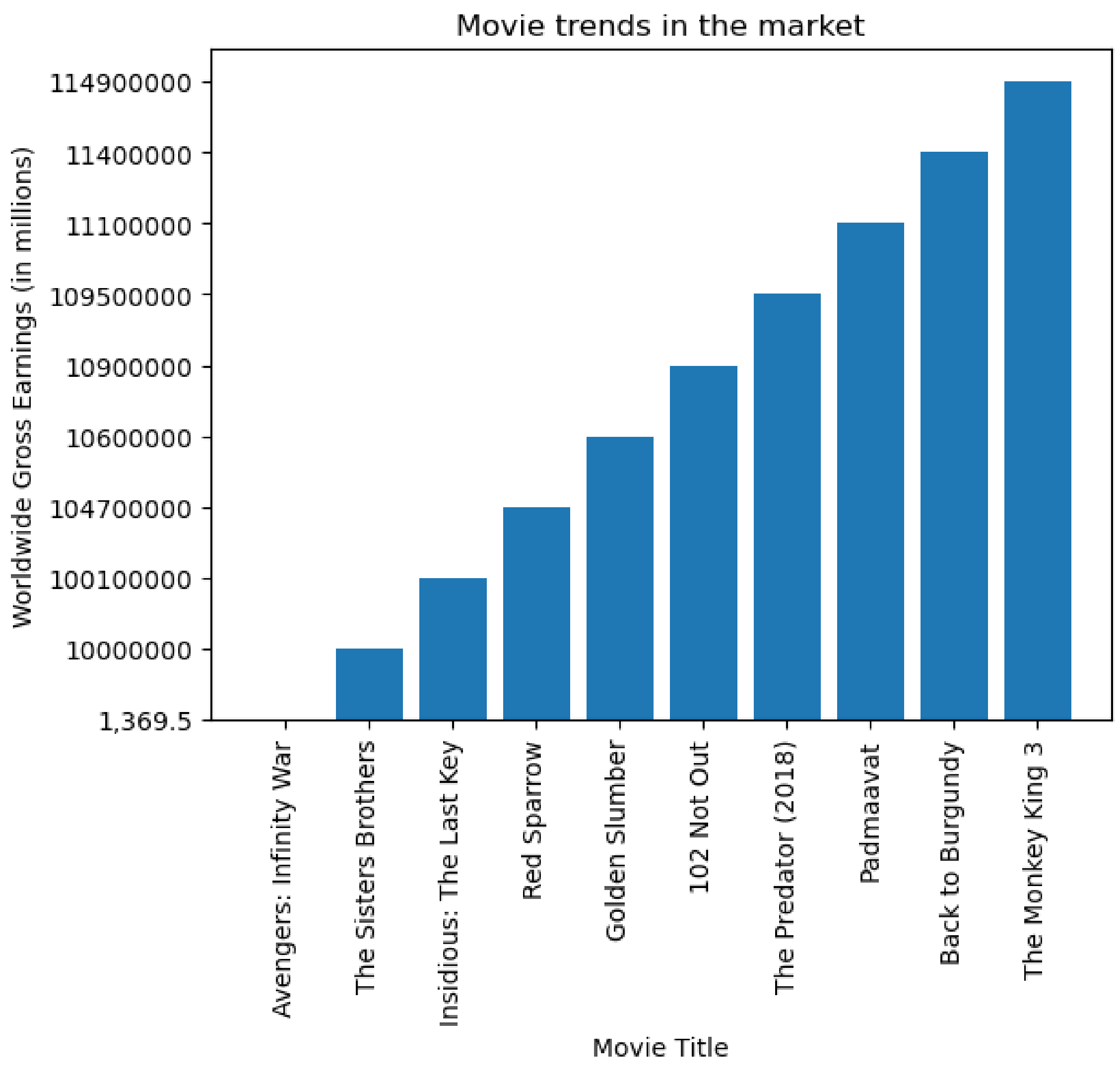
3387 rows × 5 columns

Data Visualization

Current movie trends that are doing well in the market

In [ ]:

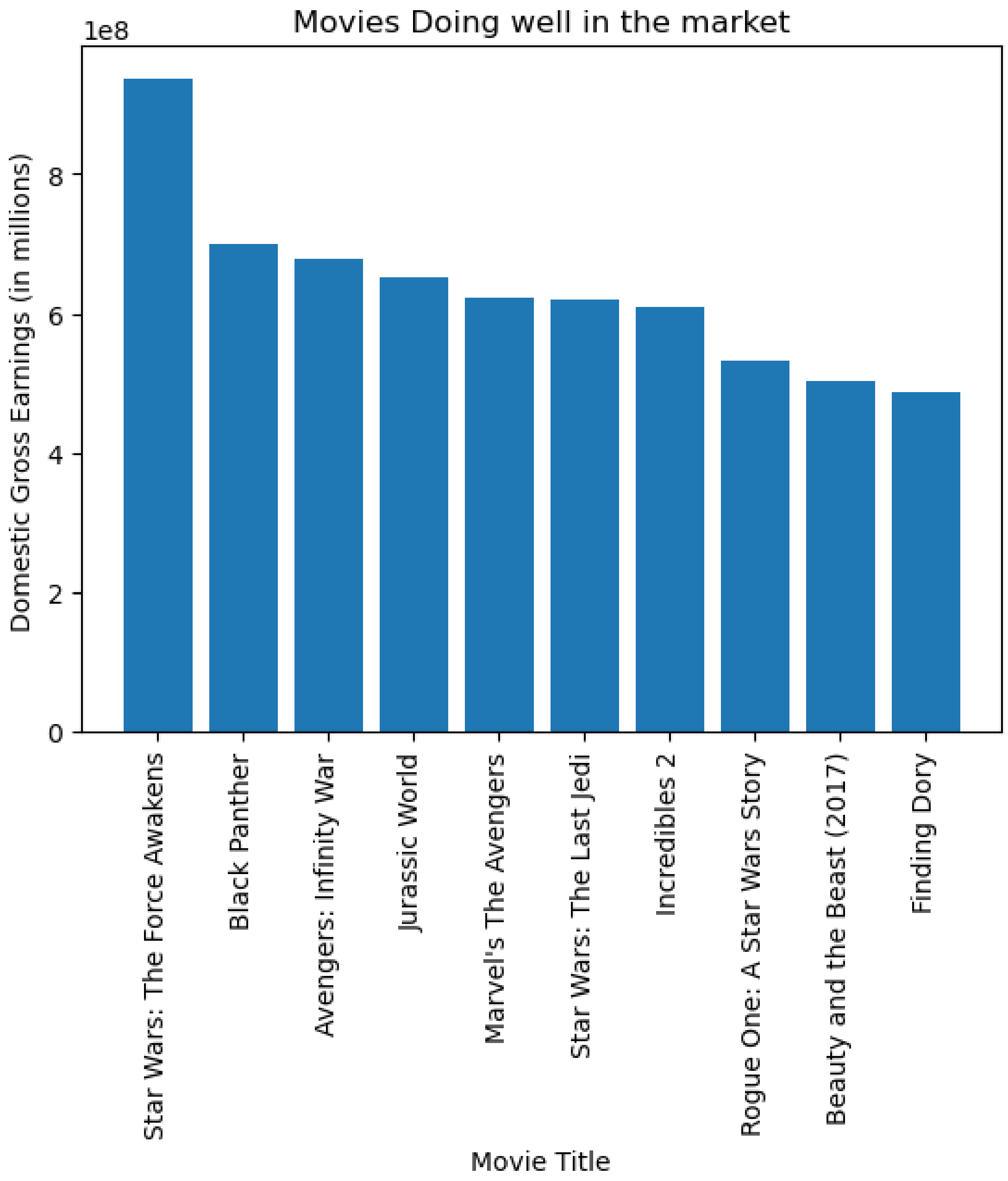
```
import pandas as pd

# Read the movie data into a DataFrame
df = pd.read_csv('./bom.movie_gross.csv')

# Filter to include movies released in 2018
movies_2018 = df[df['year'] == 2018]

# Sort the movies by worldwide gross earnings
movies_2018_sorted = movies_2018.sort_values('foreign_gross', ascending=True)
# Select the top 10 movies
top_10_movies_2018 = movies_2018_sorted.head(10)

# Create a bar graph of the top 10 movies
plt.bar(top_10_movies_2018['title'], top_10_movies_2018['foreign_gross'])

# set the title and axis labels
plt.title('Movie trends in the market')
plt.xlabel('Movie Title')
plt.ylabel('Worldwide Gross Earnings (in millions)')

# Rotate the x-axis labels for better readability
plt.xticks(rotation=90)

# Display the graph
plt.show()
```



Determining the movies that are doing well in the market

In [ ]:

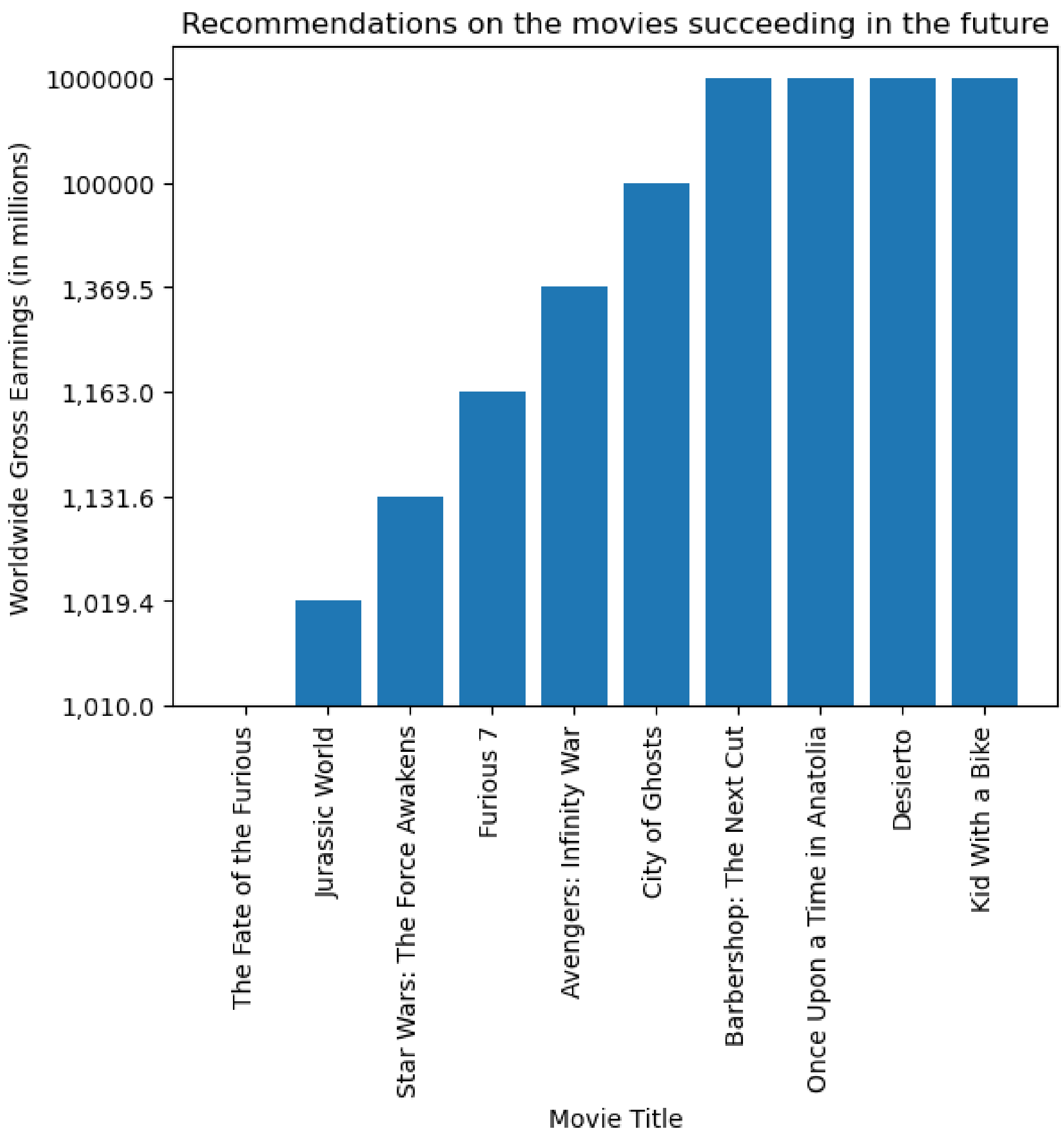
```
import pandas as pd

# Read the movie data into a DataFrame
df = pd.read_csv('./bom.movie_gross.csv')

# Filter to include only movies with non-null domestic gross earnings
movies_domestic = df[df['domestic_gross'].notnull()]

# Sort the movies by domestic gross earnings
movies_domestic_sorted = movies_domestic.sort_values('domestic_gross', ascending=False)
# Select the top 10 movies
top_10_movies_domestic = movies_domestic_sorted.head(10)

# Create a bar graph of the top 10 movies
plt.bar(top_10_movies_domestic['title'], top_10_movies_domestic['domestic_gross'])

# Set the title and axis labels
plt.title('Movies Doing well in the market')
plt.xlabel('Movie Title')
plt.ylabel('Domestic Gross Earnings (in millions)')

# Rotate the x-axis labels for better readability
plt.xticks(rotation=90)

# Display the graph
plt.show()
```



Recommendations on the types of films that are likely to be successful in the future.

In [ ]:

```
import pandas as pd

# Read the movie data into a DataFrame
df = pd.read_csv('./bom.movie_gross.csv')

# Sort the movies by worldwide gross earnings
movies_sorted = df.sort_values('foreign_gross', ascending=True)

# Select the top 10 movies
top_10_movies = movies_sorted.head(10)
# Create a bar graph of the top 10 movies
plt.bar(top_10_movies['title'], top_10_movies['foreign_gross'])

# Set the title and axis labels
plt.title('Recommendations on the movies succeeding in the future')
plt.xlabel('Movie Title')
plt.ylabel('Worldwide Gross Earnings (in millions)')

# Rotate the x-axis labels for better readability
plt.xticks(rotation=90)

# Display the graph
plt.show()
```



DATA CLEANING im.db Data

In [ ]:

```
pip install pandasql
```

```
Collecting pandasql
  Downloading pandasql-0.7.3.tar.gz (26 kB)
  Preparing metadata (setup.py): started
  Preparing metadata (setup.py): finished with status 'done'
Requirement already satisfied: numpy in c:\users\nicholas owino\anaconda3
\envs\learn-env\lib\site-packages (from pandasql) (1.23.5)
Requirement already satisfied: pandas in c:\users\nicholas owino\anaconda3
\envs\learn-env\lib\site-packages (from pandasql) (2.0.1)
Collecting sqlalchemy
  Downloading SQLAlchemy-2.0.15-cp310-cp310-win_amd64.whl (2.0 MB)
     ---------------------------------------- 2.0/2.0 MB 49.3 kB/s eta 0:0
0:00
Requirement already satisfied: tzdata>=2022.1 in c:\users\nicholas owino\a
naconda3\envs\learn-env\lib\site-packages (from pandas->pandasql) (2023.3)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\nicholas
owino\anaconda3\envs\learn-env\lib\site-packages (from pandas->pandasql)
(2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\nicholas owino\ana
conda3\envs\learn-env\lib\site-packages (from pandas->pandasql) (2023.3)
Collecting greenlet!=0.4.17
  Downloading greenlet-2.0.2-cp310-cp310-win_amd64.whl (192 kB)
     ------------------------------------- 192.2/192.2 kB 44.4 kB/s eta 0:
00:00
Requirement already satisfied: typing-extensions>=4.2.0 in c:\users\nichol
as owino\anaconda3\envs\learn-env\lib\site-packages (from sqlalchemy->pand
asql) (4.5.0)
Requirement already satisfied: six>=1.5 in c:\users\nicholas owino\anacond
a3\envs\learn-env\lib\site-packages (from python-dateutil>=2.8.2->pandas->
pandasql) (1.16.0)
Building wheels for collected packages: pandasql
  Building wheel for pandasql (setup.py): started
  Building wheel for pandasql (setup.py): finished with status 'done'
  Created wheel for pandasql: filename=pandasql-0.7.3-py3-none-any.whl siz
e=26800 sha256=e2e60c935898f4f361d99732919fe48574c67eb831d96ab3a0386466b3c
94709
  Stored in directory: c:\users\nicholas owino\appdata\local\pip\cache\whe
els\e9\bc\3a\8434bdcccf5779e72894a9b24fecbdcaf97940607eaf4bcdf9
Successfully built pandasql
Installing collected packages: greenlet, sqlalchemy, pandasql
Successfully installed greenlet-2.0.2 pandasql-0.7.3 sqlalchemy-2.0.15
Note: you may need to restart the kernel to use updated packages.
```

Import Data Packages

In [15]:

```
import pandas as pd
```

In [16]:

```
import sqlite3
```

In [17]:

```
conn = sqlite3.connect("im.db")
```

In [18]:

```
pd.read_sql("""SELECT name FROM sqlite_master  WHERE type = 'table';""", conn)
```

Out[18]:

	name
0	movie_basics
1	directors
2	known_for
3	movie_akas
4	movie_ratings
5	persons
6	principals
7	writers

In [19]:

```
query = "SELECT name FROM sqlite_master WHERE type='table';"
table_names = pd.read_sql(query, conn)
```

In [20]:

```
print(table_names)
```

```
            name
0   movie_basics
1      directors
2      known_for
3     movie_akas
4  movie_ratings
5        persons
6     principals
7        writers
```

Converting the tables into dataframes

In [21]:

```
pd.read_sql("""SELECT * FROM movie_basics;""", conn)
```

Out[21]:

	movie_id	primary_title	original_title	start_year	runtime_minutes	ger
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,Fan
...	...	...	...	...	...	
146139	tt9916538	Kuambil Lagi Hatiku	Kuambil Lagi Hatiku	2019	123.0	Dra
146140	tt9916622	Rodolpho Teóphilo - O Legado de um Pioneiro	Rodolpho Teóphilo - O Legado de um Pioneiro	2015	NaN	Documen
146141	tt9916706	Dankyavar Danka	Dankyavar Danka	2013	NaN	Com
146142	tt9916730	6 Gunn	6 Gunn	2017	116.0	N
146143	tt9916754	Chico Albuquerque - Revelações	Chico Albuquerque - Revelações	2013	NaN	Documen

146144 rows × 6 columns

In [22]:

```
pd.read_sql("""SELECT * FROM movie_ratings;""", conn)
```

Out[22]:

	movie_id	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
...	...	...	...
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

73856 rows × 3 columns

Joining the Two Tables

In [23]:

```
df = pd.read_sql("""SELECT * FROM movie_basics
    JOIN movie_ratings
    USING (movie_id);""", conn)
df
```

Out[23]:

	movie_id	primary_title	original_title	start_year	runtime_minutes	genr
0	tt0063540	Sunghursh	Sunghursh	2013	175.0	Action,Crime,Drai
1	tt0066787	One Day Before the Rainy Season	Ashad Ka Ek Din	2019	114.0	Biography,Drai
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drai
3	tt0069204	Sabse Bada Sukh	Sabse Bada Sukh	2018	NaN	Comedy,Drai
4	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fanta
...	...	...	...	...	...	
73851	tt9913084	Diabolik sono io	Diabolik sono io	2019	75.0	Documenta
73852	tt9914286	Sokagin Çocuklari	Sokagin Çocuklari	2019	98.0	Drama,Fan
73853	tt9914642	Albatross	Albatross	2017	NaN	Documenta
73854	tt9914942	La vida sense la Sara Amat	La vida sense la Sara Amat	2019	NaN	Nc
73855	tt9916160	Drømmeland	Drømmeland	2019	72.0	Documenta

73856 rows × 8 columns

In [24]:

```
df.isna().sum()
```

Out[24]:

```
movie_id              0
primary_title         0
original_title        0
start_year            0
runtime_minutes    7620
genres              804
averagerating         0
numvotes              0
dtype: int64
```

In [25]:

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

In [26]:

```
df.isna().sum()
```

Out[26]:

```
movie_id           0
primary_title      0
original_title     0
start_year         0
runtime_minutes    0
genres             0
averagerating      0
numvotes           0
dtype: int64
```

In [27]:

```
df
```

Out[27]:

	movie_id	primary_title	original_title	start_year	runtime_minutes	
0	tt0063540	Sunghursh	Sunghursh	2013	175.0	Actio
1	tt0066787	One Day Before the Rainy Season	Ashad Ka Ek Din	2019	114.0	B
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	
4	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy
6	tt0137204	Joe Finds Grace	Joe Finds Grace	2017	83.0	Adventure,An
...	...	...	...	...	...	
73849	tt9911774	Padmavyuhathile Abhimanyu	Padmavyuhathile Abhimanyu	2019	130.0	
73850	tt9913056	Swarm Season	Swarm Season	2019	86.0	
73851	tt9913084	Diabolik sono io	Diabolik sono io	2019	75.0	
73852	tt9914286	Sokagin Çocuklari	Sokagin Çocuklari	2019	98.0	
73855	tt9916160	Drømmeland	Drømmeland	2019	72.0	

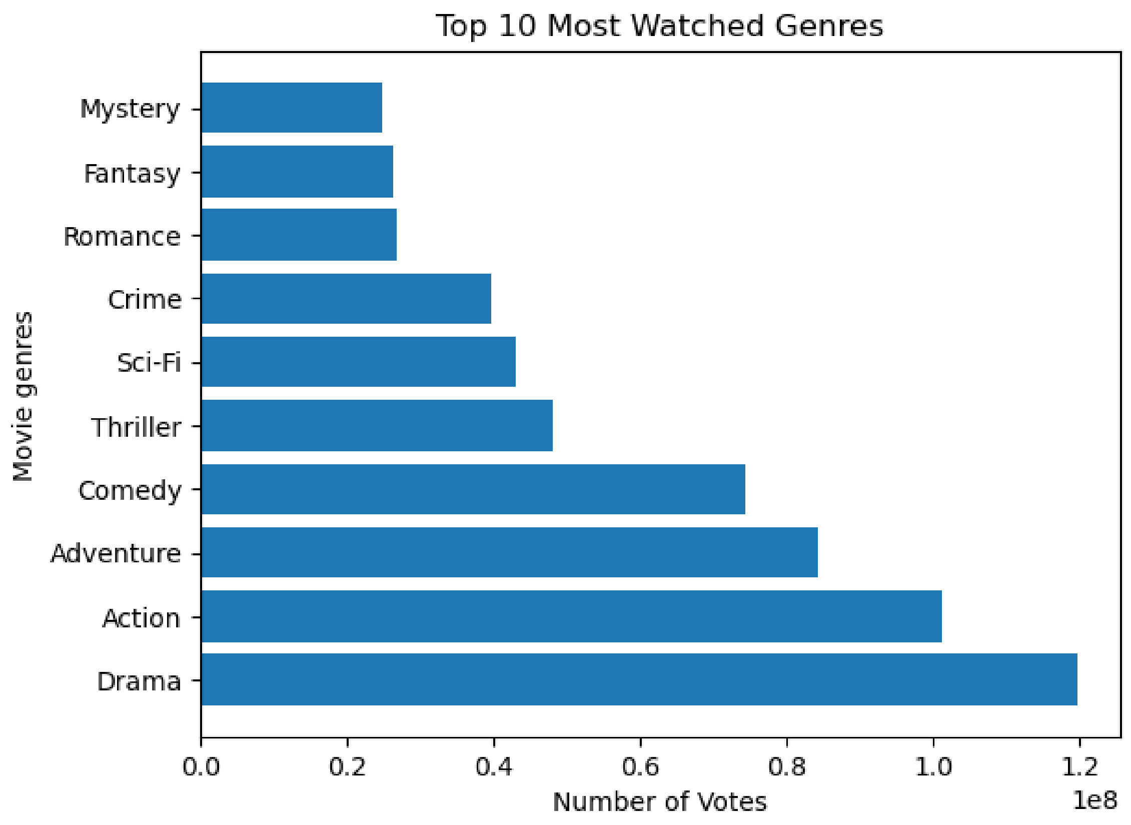
65720 rows × 8 columns

Data Visualization

Current movie trends in the market by genres

In [28]:

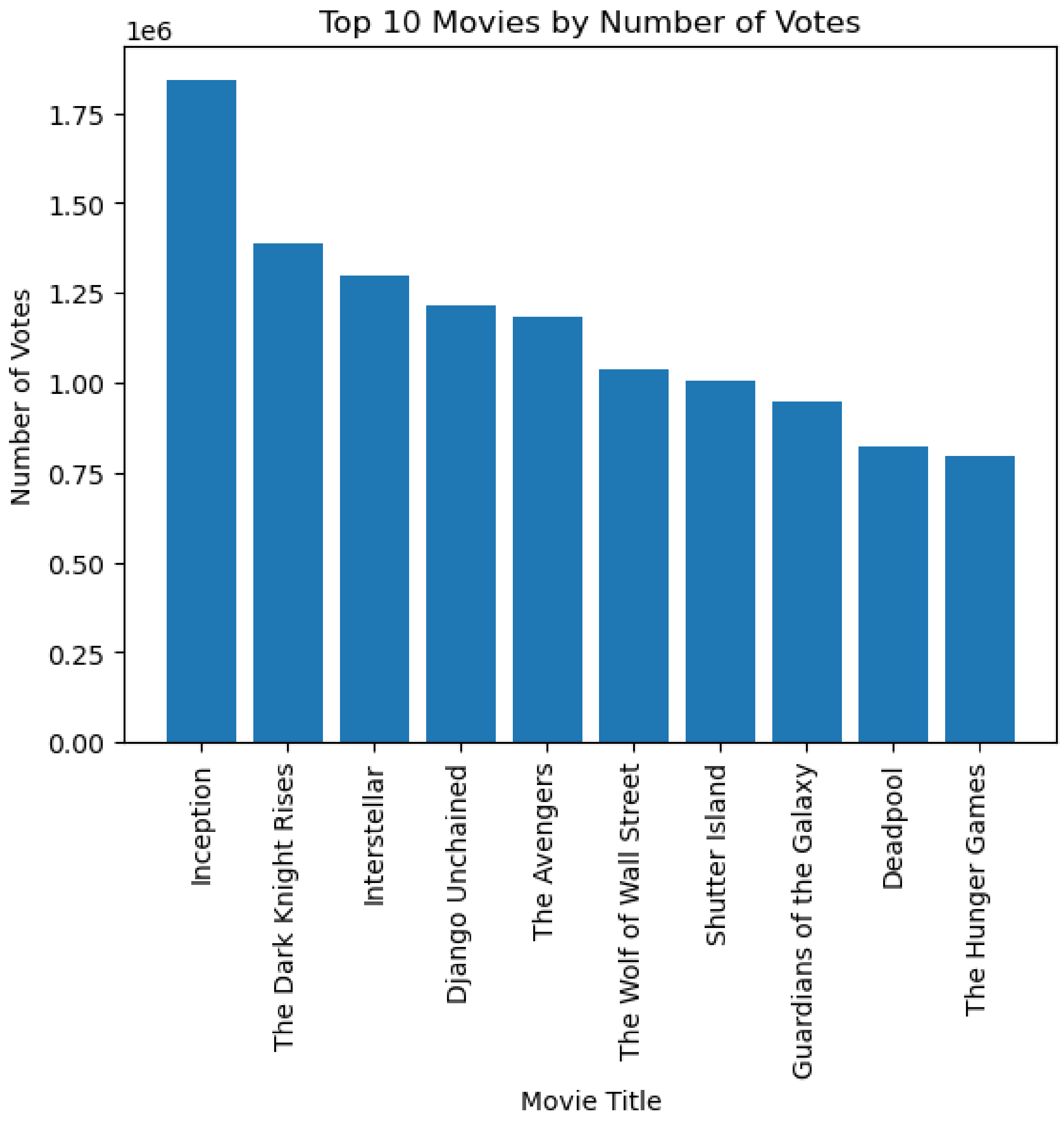
```
# Import the necessary packages
import pandas as pd
import sqlite3
import matplotlib.pyplot as plt

# Load the dataset into a pandas dataframe
df = pd.read_sql("""SELECT * FROM movie_basics
    JOIN movie_ratings
    USING (movie_id);""", conn)
df

# Split the genres column into separate values
df['genres'] = df['genres'].str.split(',')

# Explode the genres column to create one row per genre per movie
df = df.explode('genres')

# Group by genre and sum the number of votes
genre_votes = df.groupby('genres')['numvotes'].sum()

# Sort by number of votes and take the top 10
top_genres = genre_votes.sort_values(ascending=False)[:10]

# Plot a horizontal bar chart
plt.barh(top_genres.index, top_genres.values)
plt.xlabel('Number of Votes')
plt.ylabel('Movie genres')
plt.title('Top 10 Most Watched Genres')
plt.show()
```



Determining the movies that are doing well in the market

In [29]:

```
# Importing the necessary packages
import pandas as pd
import sqlite3
import matplotlib.pyplot as plt

# Load the movie data into a Pandas DataFrame
df = pd.read_sql("""SELECT * FROM movie_basics
    JOIN movie_ratings
    USING (movie_id);""", conn)
df

# Sort the movies by number of votes in descending order
df = df.sort_values('numvotes', ascending=False)

# Select the top 10 movies
top_10 = df.head(10)

# Create a bar graph of the top 10 movies
plt.bar(top_10['primary_title'], top_10['numvotes'])
plt.xticks(rotation=90)
plt.xlabel('Movie Title')
plt.ylabel('Number of Votes')
plt.title('Top 10 Movies by Number of Votes')
plt.show()
```


Top 10 Movies by Number of Votes
1e6
Number of Votes
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
Inception
The Dark Knight Rises
Interstellar
Django Unchained
The Avengers
The Wolf of Wall Street
Shutter Island
Guardians of the Galaxy
Deadpool
The Hunger Games
Movie Title