Final Project Submission

Please fill out:

- Student name: LISA MWIKALI
- Student pace: part time
- Scheduled project review date/time:
- Instructor name: WILLIAM OKOMBA , SAMUEL G MWANGI, NOAH KANDIE
- Blog post URL:

PROJECT OVERVIEW

The project aimed to guide Microsoft's entry into the entertainment industry by conducting a comprehensive analysis of current box office trends. The business problem was to provide actionable insights for the decision-making process regarding the type of films Microsoft's new studio should create. The data utilized for this analysis consisted of movie title, genres,domestic gross and foreign gross for movies made, year of release, studios, run time in minutes, average rating for different movie titles, number of votes and other relevant factors sourced from imdb database .

BUSINESS PROBLEM

The business problem revolves around Microsoft's entry into the entertainment industry with the establishment of a new movie studio The main pain points include the need to make informed decisions about the type of films the studio should create to maximize success?I picked the data analysis questions by thinking about what knowledge Microsoft might need to have, being that they are new to the movie game. The data questions aim to address key aspects crucial for strategic decision-making in this contet.t

DATA UNDERSTANDING

The sample includes a diverse set of movies, spanning various genres, gross , release dates and ratings. It represents a cross-section of the industry to provide insights into broader trends and patterns.The primary target variables are "domestic_gross" and "averagerating" which will serve as the measure of a movie's success.

LOADING THE DATA

```
#Importing the libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

#getting and reading each csv I intend to use
bom_movie_gross = pd.read_csv("C:\\Users\\Lisa\\Desktop\\PHASE1
PROJECT\\dsc-phase-1-project\\zippedData\\bom.movie_gross.csv")
```

```
imdb_title_ratings = pd.read_csv('C:\\Users\\Lisa\\Desktop\\PHASE1
PROJECT\\dsc-phase-1-project\\zippedData\\imdb.title.ratings.csv')
imdb_title_basics = pd.read_csv('C:\\Users\\Lisa\\Desktop\\PHASE1
PROJECT\\dsc-phase-1-project\\zippedData\\imdb.title.basics.csv')
```

DATA INSPECTION

```
# Previewing the first 5 rows of bom movie gross to get a sense of its
structure.
bom_movie_gross.head()
```

```
                                          title studio  domestic_gross
\
0                                Toy Story 3     BV     415000000.0

1                  Alice in Wonderland (2010)     BV     334200000.0

2  Harry Potter and the Deathly Hallows Part 1     WB     296000000.0

3                                   Inception     WB     292600000.0

4                           Shrek Forever After   P/DW     238700000.0


  foreign_gross  year
0     652000000  2010
1     691300000  2010
2     664300000  2010
3     535700000  2010
4     513900000  2010
```

```
# Previewing the last 5 rows of bom movie gross to get a sense of its
structure.
bom_movie_gross.tail()
```

```
                             title      studio  domestic_gross
foreign_gross  \
3382                   The Quake       Magn.          6200.0
NaN
3383  Edward II (2018 re-release)          FM          4800.0
NaN
3384                    El Pacto        Sony          2500.0
NaN
3385                    The Swan  Synergetic          2400.0
NaN
3386            An Actor Prepares       Grav.          1700.0
NaN

      year
3382  2018
```

```
3383  2018
3384  2018
3385  2018
3386  2018
```

```
# Previewing the first 5 rows of imdb title ratings to get a sense of
its structure.
imdb_title_ratings.head()
```

```
       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
```

```
# Previewing the last 5 rows of imdb title ratings to get a sense of
its structure.
imdb_title_ratings.tail()
```

```
          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
```

```
# Previewing the first 5 rows of imdb title basics to get a sense of
its structure.
imdb_title_basics.head()
```

```
      tconst                             title
original_title  \
0  tt0063540                         Sunghursh
Sunghursh
1  tt0066787  One Day Before the Rainy Season             Ashad Ka Ek
Din
2  tt0069049       The Other Side of the Wind  The Other Side of the
Wind
3  tt0069204                  Sabse Bada Sukh             Sabse Bada
Sukh
4  tt0100275         The Wandering Soap Opera       La Telenovela
Errante

   start_year  runtime_minutes                genres
0        2013            175.0    Action,Crime,Drama
1        2019            114.0       Biography,Drama
2        2018            122.0                 Drama
3        2018              NaN          Comedy,Drama
4        2017             80.0  Comedy,Drama,Fantasy
```

```
# Previewing the last 5 rows of imdb title basics to get a sense of
its structure.
imdb_title_basics.tail()
```

```
           tconst                                              title  \
146139  tt9916538                           Kuambil Lagi Hatiku
146140  tt9916622  Rodolpho Teóphilo - O Legado de um Pioneiro
146141  tt9916706                                Dankyavar Danka
146142  tt9916730                                        6 Gunn
146143  tt9916754              Chico Albuquerque - Revelações

                                    original_title  start_year  \
146139                           Kuambil Lagi Hatiku        2019
146140  Rodolpho Teóphilo - O Legado de um Pioneiro        2015
146141                               Dankyavar Danka        2013
146142                                       6 Gunn        2017
146143               Chico Albuquerque - Revelações        2013

        runtime_minutes       genres
146139            123.0        Drama
146140              NaN  Documentary
146141              NaN       Comedy
146142            116.0          NaN
146143              NaN  Documentary
```

MERGING THE FILES

```
# Merging imbdb title basics  and imdb title ratings based on tconst
which is common to  both of the tables
imdb_title_basics_and_ratings = pd.merge(imdb_title_basics,
imdb_title_ratings, on='tconst', how='inner')

imdb_title_basics_and_ratings.head()
```

```
      tconst                             title
original_title  \
0  tt0063540                          Sunghursh
Sunghursh
1  tt0066787  One Day Before the Rainy Season             Ashad Ka Ek
Din
2  tt0069049       The Other Side of the Wind  The Other Side of the
Wind
3  tt0069204                  Sabse Bada Sukh             Sabse Bada
Sukh
4  tt0100275         The Wandering Soap Opera       La Telenovela
Errante

   start_year  runtime_minutes                genres  averagerating
numvotes
0        2013            175.0    Action,Crime,Drama            7.0
```

```
77
1        2019            114.0       Biography,Drama            7.2
43
2        2018            122.0                 Drama            6.9
4517
3        2018              NaN          Comedy,Drama            6.1
13
4        2017             80.0  Comedy,Drama,Fantasy            6.5
119
```

```
# Merging bom Movie Gross with imdb title basics and ratings using
'title' as the common column between file1 and merged_file2_file3
merged = pd.merge(bom_movie_gross, imdb_title_basics_and_ratings,
left_on='title', right_on='title', how='inner')

merged.head()
```

```
                        title studio  domestic_gross foreign_gross
year  \
0                 Toy Story 3     BV     415000000.0     652000000
2010
1                   Inception     WB     292600000.0     535700000
2010
2         Shrek Forever After   P/DW     238700000.0     513900000
2010
3  The Twilight Saga: Eclipse   Sum.     300500000.0     398000000
2010
4                  Iron Man 2   Par.     312400000.0     311500000
2010

      tconst              original_title  start_year  runtime_minutes
\
0  tt0435761                Toy Story 3        2010            103.0

1  tt1375666                  Inception        2010            148.0

2  tt0892791         Shrek Forever After        2010             93.0

3  tt1325004  The Twilight Saga: Eclipse        2010            124.0

4  tt1228705                  Iron Man 2        2010            124.0


                       genres  averagerating  numvotes
0  Adventure,Animation,Comedy            8.3    682218
1     Action,Adventure,Sci-Fi            8.8   1841066
2  Adventure,Animation,Comedy            6.3    167532
3     Adventure,Drama,Fantasy            5.0    211733
4     Action,Adventure,Sci-Fi            7.0    657690
```

```
#checking the shape of the dataset
merged.shape
```

```
(3028, 12)
```

```
#checking for data types
merged.dtypes
```

```
title                object
studio               object
domestic_gross      float64
foreign_gross        object
year                  int64
tconst               object
original_title       object
start_year            int64
runtime_minutes     float64
genres               object
averagerating       float64
numvotes              int64
dtype: object
```

```
#checking for missing values
merged.isnull().sum()
```

```
title                   0
studio                  3
domestic_gross         22
foreign_gross        1195
year                    0
tconst                  0
original_title          0
start_year              0
runtime_minutes        47
genres                  7
averagerating           0
numvotes                0
dtype: int64
```

DATA CLEANING

Dropping Variables with Null Values and removing outliers

Variables Dropped: studio, domestic_gross, foreign_gross, run_time in minutes, genres I chose to drop this variables because they are crucial for the analysis and they contained null values. Removing rows with missing values in these key variables ensures the integrity of the analysis, as imputing critical financial and content-related information may introduce inaccuracies especially considering I plan to use the columns for my visualizations. I then used the IQR method to address outliers in the variable "domestic_gross" .My reason was because the

outliers might distort the assessment of the movie's rating, and addressing them ensures a more accurate representation of the central tendency.

```
#dropping the missing values
merged = merged.dropna()
merged
```

```
                                        title      studio
domestic_gross  \
0                                Toy Story 3          BV
415000000.0
1                                  Inception          WB
292600000.0
2                      Shrek Forever After        P/DW
238700000.0
3               The Twilight Saga: Eclipse        Sum.
300500000.0
4                                Iron Man 2        Par.
312400000.0
...                                     ...         ...
...
2928            Bilal: A New Breed of Hero          VE
491000.0
2931                      I Still See You         LGF
1400.0
2941                  The Catcher Was a Spy         IFC
725000.0
2960                              Time Freak  Grindstone
10000.0
3002  Antonio Lopez 1970: Sex Fashion & Disco          FM
43200.0

     foreign_gross  year     tconst
original_title  \
0        652000000  2010  tt0435761                              Toy
Story 3
1        535700000  2010  tt1375666
Inception
2        513900000  2010  tt0892791                    Shrek Forever
After
3        398000000  2010  tt1325004              The Twilight Saga:
Eclipse
4        311500000  2010  tt1228705                               Iron
Man 2
...            ...   ...        ...
...
2928       1700000  2018  tt3576728              Bilal: A New Breed
of Hero
2931       1500000  2018  tt2160105                        I Still
See You
```

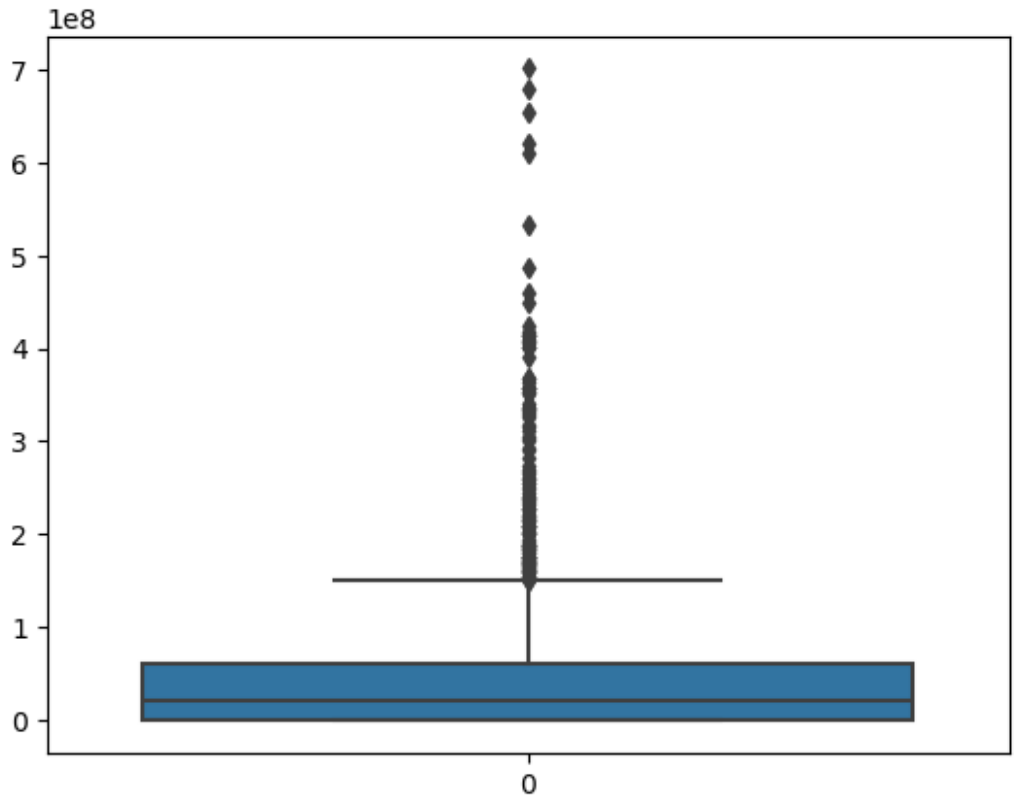
```
2941        229000  2018  tt4602066                      The Catcher Was
a Spy
2960        256000  2018  tt6769280                                Time
Freak
3002         30000  2018  tt5792490  Antonio Lopez 1970: Sex Fashion &
Disco

      start_year  runtime_minutes                       genres
averagerating  \
0           2010            103.0  Adventure,Animation,Comedy
8.3
1           2010            148.0     Action,Adventure,Sci-Fi
8.8
2           2010             93.0  Adventure,Animation,Comedy
6.3
3           2010            124.0     Adventure,Drama,Fantasy
5.0
4           2010            124.0     Action,Adventure,Sci-Fi
7.0
...          ...              ...                          ...
...
2928        2015            105.0  Action,Adventure,Animation
8.0
2931        2018             98.0            Fantasy,Thriller
5.7
2941        2018             98.0         Biography,Drama,War
6.2
2960        2018            104.0        Comedy,Drama,Romance
5.7
3002        2017             95.0       Biography,Documentary
6.5

      numvotes
0       682218
1      1841066
2       167532
3       211733
4       657690
...        ...
2928     16854
2931      5010
2941      4653
2960      3455
3002       102

[1768 rows x 12 columns]
```

```
#checking for duplicated values
merged.duplicated()
```

```
0       False
1       False
2       False
3       False
4       False
        ...
2928    False
2931    False
2941    False
2960    False
3002    False
Length: 1768, dtype: bool
```

```
#checking for outliers
sns.boxplot(merged["domestic_gross"])
```

```
<Axes: >
```




```
#removing outliers using interquartile range method
q1 = merged["domestic_gross"].quantile(0.25)
q3 = merged["domestic_gross"].quantile(0.75)
iqr = q3 - q1
q1, q3, iqr
```

```
(1300000.0, 61100000.0, 59800000.0)
```

```
#finding the upper limit and lower limit
upper_limit = q3 + (1.5 * iqr)
lower_limit = q1 - (1.5 * iqr)
upper_limit, lower_limit
```

```
(150800000.0, -88400000.0)
```

```
#checking for outliers
merged.loc[(merged["domestic_gross"] > upper_limit) |
(merged["domestic_gross"] < lower_limit)]
```

```
                          title studio  domestic_gross foreign_gross
year  \
0                   Toy Story 3     BV     415000000.0     652000000
2010
1                     Inception     WB     292600000.0     535700000
2010
2            Shrek Forever After   P/DW     238700000.0     513900000
2010
3     The Twilight Saga: Eclipse   Sum.     300500000.0     398000000
2010
4                     Iron Man 2   Par.     312400000.0     311500000
2010
...                          ...    ...             ...           ...
...
2770     Solo: A Star Wars Story     BV     213800000.0     179200000
2018
2776        Mary Poppins Returns     BV     172000000.0     177600000
2018
2777               A Quiet Place   Par.     188000000.0     152900000
2018
2778               A Quiet Place   Par.     188000000.0     152900000
2018
2782           Crazy Rich Asians     WB     174500000.0      64000000
2018

         tconst              original_title  start_year
runtime_minutes  \
0     tt0435761                  Toy Story 3        2010
103.0
1     tt1375666                    Inception        2010
148.0
2     tt0892791          Shrek Forever After        2010
93.0
3     tt1325004  The Twilight Saga: Eclipse        2010
124.0
4     tt1228705                   Iron Man 2        2010
124.0
```

```
...         ...                          ...         ...               .
..
2770  tt3778644     Solo: A Star Wars Story        2018
135.0
2776  tt5028340        Mary Poppins Returns        2018
130.0
2777  tt6347308               A Quiet Place        2016
80.0
2778  tt6644200               A Quiet Place        2018
90.0
2782  tt3104988           Crazy Rich Asians        2018
120.0

                          genres  averagerating  numvotes
0     Adventure,Animation,Comedy            8.3    682218
1        Action,Adventure,Sci-Fi            8.8   1841066
2     Adventure,Animation,Comedy            6.3    167532
3        Adventure,Drama,Fantasy            5.0    211733
4        Action,Adventure,Sci-Fi            7.0    657690
...                          ...            ...       ...
2770    Action,Adventure,Fantasy            7.0    226243
2776       Comedy,Family,Fantasy            6.9     52103
2777                 Documentary            6.6        18
2778         Drama,Horror,Sci-Fi            7.6    305031
2782              Comedy,Romance            7.0     96617

[146 rows x 12 columns]
```

```
#handling the outliers - trimming the data
merged = merged.loc[(merged["domestic_gross"] <= upper_limit) &
(merged["domestic_gross"] >=lower_limit)]
#print the data after removing the outliers
merged
```

```
                                                 title      studio  \
8     The Chronicles of Narnia: The Voyage of the Da...         Fox
9                                 The King's Speech       Wein.
11                Prince of Persia: The Sands of Time          BV
12                                        Black Swan        FoxS
13                                          Megamind        P/DW
...                                              ...         ...
2928                        Bilal: A New Breed of Hero          VE
2931                                  I Still See You         LGF
2941                       The Catcher Was a Spy          IFC
2960                                    Time Freak  Grindstone
3002           Antonio Lopez 1970: Sex Fashion & Disco          FM

      domestic_gross foreign_gross  year     tconst  \
8        104400000.0     311300000  2010  tt0980970
9        135500000.0     275400000  2010  tt1504320
```

```
11        90800000.0     245600000  2010  tt0473075
12       107000000.0     222400000  2010  tt0947798
13       148400000.0     173500000  2010  tt1001526
...              ...           ...   ...        ...
2928        491000.0       1700000  2018  tt3576728
2931          1400.0       1500000  2018  tt2160105
2941        725000.0        229000  2018  tt4602066
2960         10000.0        256000  2018  tt6769280
3002         43200.0         30000  2018  tt5792490

                                        original_title  start_year  \
8     The Chronicles of Narnia: The Voyage of the Da...        2010
9                                     The King's Speech        2010
11                Prince of Persia: The Sands of Time        2010
12                                         Black Swan        2010
13                                           Megamind        2010
...                                               ...         ...
2928                         Bilal: A New Breed of Hero        2015
2931                                   I Still See You        2018
2941                            The Catcher Was a Spy        2018
2960                                       Time Freak        2018
3002            Antonio Lopez 1970: Sex Fashion & Disco        2017

      runtime_minutes                      genres  averagerating
numvotes
8               113.0    Adventure,Family,Fantasy            6.3
129663
9               118.0     Biography,Drama,History            8.0
593629
11              116.0    Action,Adventure,Fantasy            6.6
254975
12              108.0              Drama,Thriller            8.0
648854
13               95.0     Action,Animation,Comedy            7.3
207488
...               ...                         ...            ...
...
2928            105.0  Action,Adventure,Animation            8.0
16854
2931             98.0            Fantasy,Thriller            5.7
5010
2941             98.0         Biography,Drama,War            6.2
4653
2960            104.0        Comedy,Drama,Romance            5.7
3455
3002             95.0       Biography,Documentary            6.5
102

[1622 rows x 12 columns]
```

DATA MODELING

FEATURE ENGINEERING

```
# Define the bins and labels for the rating
bins = [0, 2, 4, 6, 8, 10]
labels = ['Poor', 'Fair', 'Average', 'Good', 'Excellent']

# Create a new column 'Rating' by grouping values into the specified
bins
merged['Rating'] = pd.cut(merged["averagerating"], bins=bins,
labels=labels, right=False)

# Display the updated DataFrame
merged
```

```
C:\Users\Lisa\AppData\Local\Temp\ipykernel_1084\2317632734.py:6:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  merged['Rating'] = pd.cut(merged["averagerating"], bins=bins,
labels=labels, right=False)
```

	title	studio	\
8	The Chronicles of Narnia: The Voyage of the Da...	Fox	
9	The King's Speech	Wein.	
11	Prince of Persia: The Sands of Time	BV	
12	Black Swan	FoxS	
13	Megamind	P/DW	
...	...	...	
2928	Bilal: A New Breed of Hero	VE	
2931	I Still See You	LGF	
2941	The Catcher Was a Spy	IFC	
2960	Time Freak	Grindstone	
3002	Antonio Lopez 1970: Sex Fashion & Disco	FM	

	domestic_gross	foreign_gross	year	tconst	\
8	104400000.0	311300000	2010	tt0980970	
9	135500000.0	275400000	2010	tt1504320	
11	90800000.0	245600000	2010	tt0473075	
12	107000000.0	222400000	2010	tt0947798	
13	148400000.0	173500000	2010	tt1001526	
...	...	...	...	...	
2928	491000.0	1700000	2018	tt3576728	
2931	1400.0	1500000	2018	tt2160105	
2941	725000.0	229000	2018	tt4602066	
2960	10000.0	256000	2018	tt6769280	

```
3002         43200.0         30000  2018  tt5792490

                                          original_title  start_year  \
8     The Chronicles of Narnia: The Voyage of the Da...        2010
9                                       The King's Speech        2010
11                  Prince of Persia: The Sands of Time        2010
12                                          Black Swan        2010
13                                            Megamind        2010
...                                                 ...         ...
2928                          Bilal: A New Breed of Hero        2015
2931                                    I Still See You        2018
2941                               The Catcher Was a Spy        2018
2960                                          Time Freak        2018
3002            Antonio Lopez 1970: Sex Fashion & Disco        2017

      runtime_minutes                      genres  averagerating
numvotes  \
8               113.0    Adventure,Family,Fantasy            6.3
129663
9               118.0     Biography,Drama,History            8.0
593629
11              116.0    Action,Adventure,Fantasy            6.6
254975
12              108.0              Drama,Thriller            8.0
648854
13               95.0     Action,Animation,Comedy            7.3
207488
...               ...                         ...            ...
...
2928            105.0  Action,Adventure,Animation            8.0
16854
2931             98.0            Fantasy,Thriller            5.7
5010
2941             98.0         Biography,Drama,War            6.2
4653
2960            104.0        Comedy,Drama,Romance            5.7
3455
3002             95.0       Biography,Documentary            6.5
102

         Rating
8          Good
9     Excellent
11         Good
12    Excellent
13         Good
...         ...
2928  Excellent
2931    Average
2941       Good
```

```
2960    Average
3002       Good

[1622 rows x 13 columns]
```

EDA

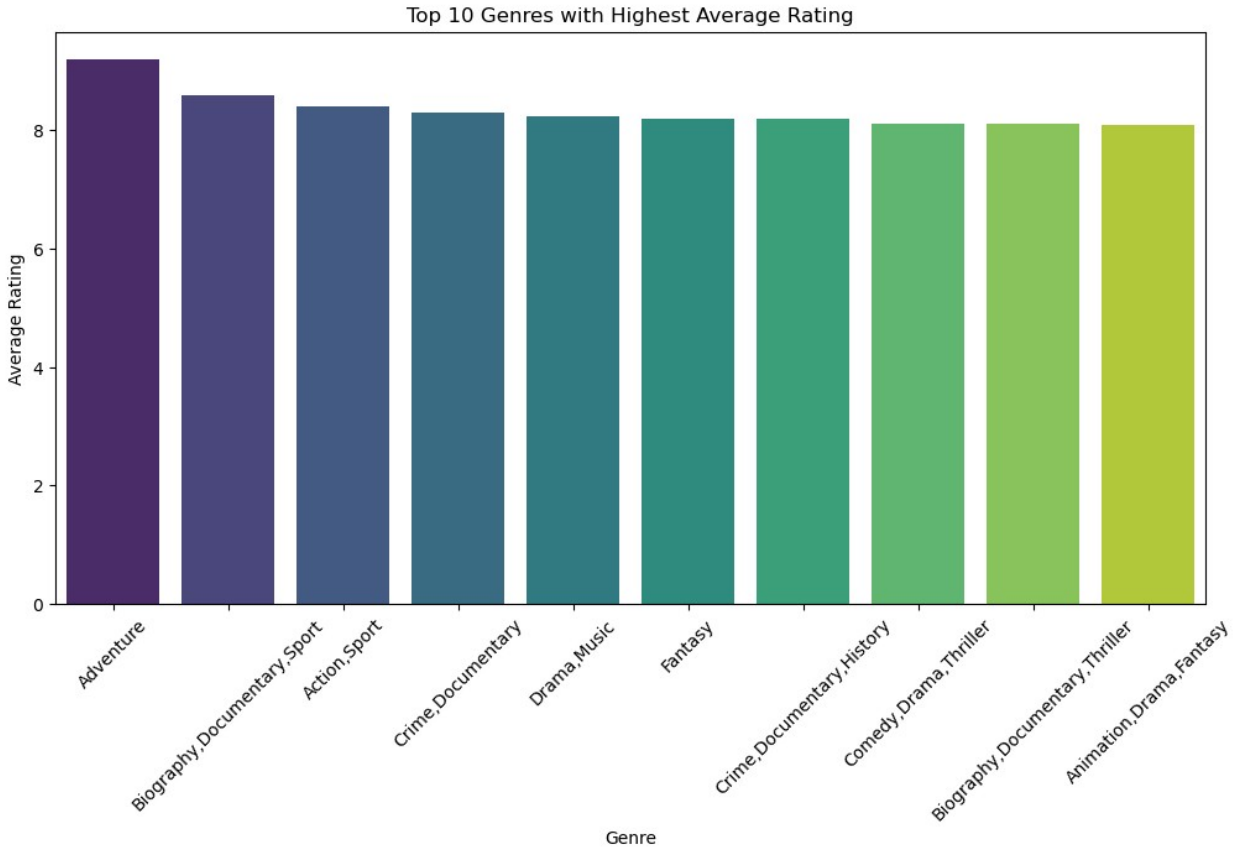
```
# Calculating the weighted average rating and sum of domestic gross
for each genre
weighted_avg_rating = merged.groupby('genres').agg({'averagerating':
lambda x: (x * merged.loc[x.index, 'numvotes']).sum() /
merged.loc[x.index, 'numvotes'].sum(),
                                                    'domestic_gross': 'sum',
                                                    'numvotes':
'sum'}).reset_index()

# Sort by weighted average rating in descending order
sorted_merged_rating =
weighted_avg_rating.sort_values(by='averagerating', ascending=False)
top_10_average_rating = sorted_merged_rating.head(10)

# Sort by domestic gross in descending order
sorted_merged_domestic =
weighted_avg_rating.sort_values(by='domestic_gross', ascending=False)
top_10_domestic_gross = sorted_merged_domestic.head(10)

# Display the sorted weighted average ratings and domestic gross
print("Top Genres based on Weighted Average Rating:")
print(sorted_merged_rating.head(10))

print("\nTop Genres based on Domestic Gross:")
print(sorted_merged_domestic.head(10))
```

```
Top Genres based on Weighted Average Rating:
                            genres  averagerating  domestic_gross
numvotes
60                         Adventure       9.200000       3600000.0
47
114     Biography,Documentary,Sport       8.593394       1776000.0
55511
58                      Action,Sport       8.400000       4200000.0
8
167                Crime,Documentary       8.300000       4300000.0
65304
207                      Drama,Music       8.235441      61900000.0
774367
228                          Fantasy       8.200000        146000.0
12
168       Crime,Documentary,History       8.200000        708000.0
15
```

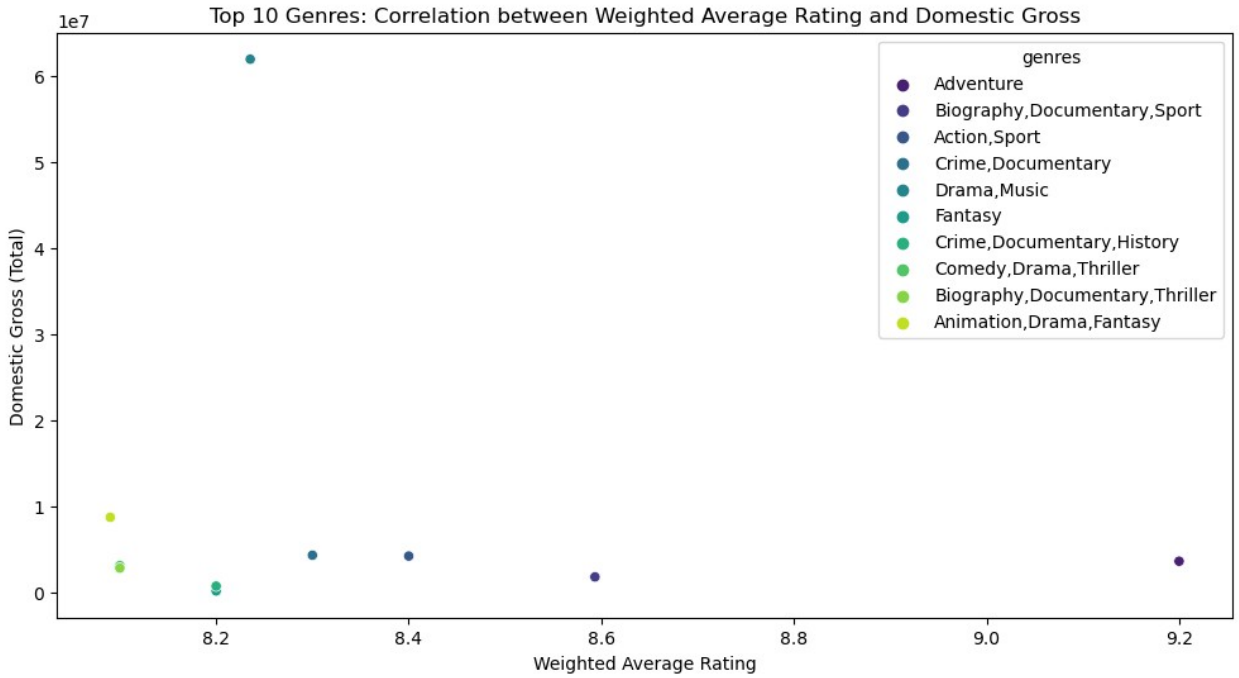
```
143           Comedy,Drama,Thriller       8.100000       3100000.0
151123
115  Biography,Documentary,Thriller       8.100000       2800000.0
47994
102         Animation,Drama,Fantasy       8.090101       8700000.0
204500

Top Genres based on Domestic Gross:
                         genres  averagerating  domestic_gross
numvotes
61   Adventure,Animation,Comedy       6.406943    3.105838e+09
2035366
189                       Drama       7.349237    2.241894e+09
2720080
124                      Comedy       5.970694    2.159740e+09
3426092
10      Action,Adventure,Sci-Fi       6.554069    1.653900e+09
4451412
133               Comedy,Drama       6.826813    1.640057e+09
2545366
160              Comedy,Romance       6.086460    1.487929e+09
2700097
141        Comedy,Drama,Romance       6.968311    1.445831e+09
4418196
19          Action,Comedy,Crime       6.545402    1.429857e+09
2889974
180                 Documentary       7.204841    1.346385e+09
91862
237     Horror,Mystery,Thriller       6.173324    1.307266e+09
2256877
```

DATA VISUALIZATION

```
# Plotting top 10 genres with the highest average rating
plt.figure(figsize=(12, 6))
sns.barplot(x='genres', y='averagerating', data=top_10_average_rating,
palette='viridis')
plt.xlabel('Genre')
plt.ylabel('Average Rating')
plt.title('Top 10 Genres with Highest Average Rating')
plt.xticks(rotation=45)  # Rotate x-axis labels for better visibility
plt.show()
```




```
# Plotting correlation
plt.figure(figsize=(12, 6))
sns.scatterplot(x='averagerating', y='domestic_gross',
data=top_10_average_rating, hue='genres', palette='viridis')
plt.xlabel('Weighted Average Rating')
plt.ylabel('Domestic Gross (Total)')
plt.title('Top 10 Genres: Correlation between Weighted Average Rating
and Domestic Gross')
plt.show()
```



Evaluation Top 10 Genres by Average Rating: The bar plot displaying the top 10 genres with the highest average ratings provides valuable insights into the audience's preferences. Itwill helps Microsoft understand which genres tend to receive the highest ratings, aiding in decision-making on film types.a: The model, myhis case, is a representation of the data analysis approach rather than a predictive model. The fit is determined by how well the code accurately calculates and visualizes the top genres based on average ratings.

Correlation Between Weighted Average Rating and Domestic Grtion: The scatter plot illustrating the correlation between the weighted average rating and domestic gross allows Microsoft to explore potential relationships between audience ratings and financial success. This insight is crucial for making strategic decisions on film produo Data: The fit of the scatter plot is determined by how well it represents the correlation between weighted average ratings and domestic gross based on the available data.

Generalization and BusinesTtaset. If the dataset is a good representation of potential future scenarios, the insights gained from the analysis may be applicable to similar siFor tess Impact: The poten business impacttbecause thelysis. If thare in alignment withet trends and audience preferences, Microsoft can make informed decisions on fi and potentially, potente on ly maximizitined insights.

Benefits to Business: The model, in this context, is a tool for exploraa predictive model. Its benefit to the business lies in providing actionable insights and informing strategic decision-making for Microsoft's entry into the entertmaking process.

se?

Conclusions

Provide your conclusions about the work you've done, including any limitations or next steps. Top 10 Genres by Average Rating: The bar plot successfully identifies the top genres with the

highest average ratings, providing insights into audience preferences. This information can guide Microsoft in making strategic decisions about the types of films to prioritiz

e. Correlation Between Weighted Average Rating and Domestic Gros:

The scatter plot explores the relationship between weighted average ratings and domestic gross, aiding in understanding the potential financial success of f Limitations:nss:

The analysis is limited by the representativeness of the dataset. If the dataset does not adequately capture the diversity of audience preferences, the recommendations may be skewed.rocess.

What would you recommend the business do as a result of this work?

Utilize insights from the top genres by average rating to strategically select film genres that align with audience preferences.

Consider the correlation between weighted average rating and domestic gross to strike a balance between creating artistically valuable films and ensuring commercial success.

To enhance confidence in the results, continuous validation, refinement, and consideration of external factors influencing the entertainment industry are essential.

What are some reasons why your analysis might not fully solve the business problem There is a limited data scope. The analysiI have done s is based on available data, which may not cover all relevant factors influencing film success. A more extensive dataset may provide a more comprehensivanalysisng Audience ratings are subjective and may not fully capture the nuanced reasons behind film success. Factors like marketing, competition, and timing are not explicitly considered in this analysis.s. t?