Final Project Submission

Please fill out:

- Student name: Nicole Bosibori Nyagaka
- Student pace: Part Time
- Scheduled project review date/time:
- Instructor name: Noah Kandie
- Blog post URL:

MICROSOFT MOVIE STUDIO ANALYSIS

Overview

To effectively embark on the venture into the film industry, Microsoft recognizes the importance of understanding the current landscape of the movie business. The primary objective of this research is to utilize exploratory data analysis (EDA) techniques to discern the types of films that demonstrate robust performance at the box office. The ultimate goal is to furnish Microsoft with practical insights that will facilitate the development of a lucrative and competitive film studio.

Business Understanding

Microsoft faces a crucial decision in selecting film for development, given its limited experience in the film industry. The challenge lies in overseeing the establishment of Microsoft's new film studio by identifying prevalent movies, partnering with suitable studios, and understanding current consumer trends.

Data Understanding

Dataset Overview

The dataset contains information on movie titles, studios,worldwide, domestic and foreign gross revenue,production budget and release years and date.

Data Cleaning

Addressed missing values and converted relevant columns to appropriate data types. Handling irrelevant column

Exploratory Data Analysis (EDA)

Understanding the correlations between studio size and box office success, identified trends, and visualized revenue distribution across studios.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

DATA INSPECTION AND CLEANING

Check column names

Merge the files

Preview rows of the dataframe

Check the shape of the dataframe

Display summary statistics

Check the data information

Check the datatypes of each column

Count the missing values in each coulmn

Check for duplicates

Check for Outliers

```
#load the data
df1 = pd.read_csv('bom.movie_gross.csv')
df2 = pd.read_csv('tmdb.movies.csv')
df3 = pd.read_csv('tn.movie_budgets.csv')
```

```
# Merge 'bom.movie_gross.csv' and 'tmdb.movies.csv' on 'title'
merged_df1 = pd.merge(df1, df2, on='title', how='inner')

# Merge the result with 'tn.movie_budgets.csv' on 'movie' and 'title'
df4 = pd.merge(merged_df1, df3, left_on='title', right_on='movie', how='inner')

# Drop the redundant 'movie' column after merging
df4.drop('movie', axis=1, inplace=True)
```

```
# Preview rows of the dataset
df4.head()
```

	title	studio	domestic_gross_x	foreign_gross	year	Unnamed: 0	ge
0	Toy Story 3	BV	415000000.0	652000000	2010	7	1(
1	Inception	WB	292600000.0	535700000	2010	4	
2	Shrek Forever After	P/DW	238700000.0	513900000	2010	38	
3	The Twilight Saga: Eclipse	Sum.	300500000.0	398000000	2010	15	1(
4	Iron Man 2	Par.	312400000.0	311500000	2010	2	

```
# Checking the column names
print("Columns in bom.movie_gross.csv:", df1.columns)
print("Columns in tmbd.movies.csv:", df2.columns)
print("Columns in tn.movie_budgets.csv:", df3.columns)
```

```
Columns in bom.movie_gross.csv: Index(['title', 'studio', 'domestic_gross', 'foreign_gross', 'year'], dtype='ob
Columns in tmbd.movies.csv: Index(['Unnamed: 0', 'genre_ids', 'id', 'original_language', 'original_title',
       'popularity', 'release_date', 'title', 'vote_average', 'vote_count'],
      dtype='object')
Columns in tn.movie_budgets.csv: Index(['id', 'release_date', 'movie', 'production_budget', 'domestic_gross',
       'worldwide_gross'],
      dtype='object')
```

```
#dropping uneccessary columns
df4.drop(['Unnamed: 0','genre_ids','id_x','vote_count','original_title','id_y','vote_average','release_date_x','dom
```

```
# Display the shape of the DataFrame

data_shape = df4.shape
print("Number of Rows:", data_shape[0])  #counts rows
print("Number of Columns:", data_shape[1]) #counts columns
```

```
Number of Rows: 1395
Number of Columns: 10
```

```
# Display summary statistics
print(df4.describe())
```

```
              year   popularity
count  1395.000000  1395.000000
mean   2013.808602    13.031513
std       2.511937     8.038919
min    2010.000000     0.600000
25%    2012.000000     8.448000
50%    2014.000000    11.369000
75%    2016.000000    15.974000
max    2018.000000    80.773000
```

```
#check the data information
df4.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1395 entries, 0 to 1394
Data columns (total 10 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   title              1395 non-null   object 
 1   studio             1394 non-null   object 
 2   foreign_gross      1200 non-null   object 
 3   year               1395 non-null   int64  
 4   original_language  1395 non-null   object 
 5   popularity         1395 non-null   float64
 6   release_date_y     1395 non-null   object 
 7   production_budget  1395 non-null   object 
 8   domestic_gross_y   1395 non-null   object 
 9   worldwide_gross    1395 non-null   object 
dtypes: float64(1), int64(1), object(8)
memory usage: 119.9+ KB
```

```
# Display datatypes of each column
df4.dtypes
```

```
title                 object
studio                object
foreign_gross         object
year                   int64
original_language     object
popularity           float64
release_date_y        object
production_budget     object
domestic_gross_y      object
worldwide_gross       object
dtype: object
```

```
# Convert 'foreign_gross' to numeric
df4['foreign_gross'] = pd.to_numeric(df4['foreign_gross'], errors='coerce')

# Convert 'release_date_x' and 'release_date_y' to datetime
df4['release_date_y'] = pd.to_datetime(df4['release_date_y'], errors='coerce')

# Convert 'production_budget', 'domestic_gross_y', and 'worldwide_gross' to numeric
df4['production_budget'] = pd.to_numeric(df4['production_budget'].replace('[\$,]', '', regex=True), errors='coerce'
df4['domestic_gross_y'] = pd.to_numeric(df4['domestic_gross_y'].replace('[\$,]', '', regex=True), errors='coerce')
df4['worldwide_gross'] = pd.to_numeric(df4['worldwide_gross'].replace('[\$,]', '', regex=True), errors='coerce')

df4.dtypes
```

```
title                        object
studio                       object
foreign_gross               float64
year                          int64
original_language            object
popularity                  float64
release_date_y       datetime64[ns]
production_budget             int64
domestic_gross_y              int64
worldwide_gross               int64
dtype: object
```

```
# Count missing values in each column
missing_values = df4.isnull().sum() #calculates the number of missing columns
missing_values
```

```
title                  0
studio                 1
foreign_gross        199
year                   0
original_language      0
popularity             0
release_date_y         0
production_budget      0
domestic_gross_y       0
worldwide_gross        0
dtype: int64
```

```
# Handling the missing values ( Studio,domestic gross x,foreign gross )
# Drop the missing values
df4 = df4.dropna(subset=['studio', 'foreign_gross'])
```

```
# Check for duplicate rows in the DataFrame
duplicate_rows = df4[df4.duplicated()]

# Display the duplicate rows
print("Duplicate Rows:")
print(duplicate_rows)
```

```
Duplicate Rows:
                            title  studio  foreign_gross  year  \
120               Blue Valentine   Wein.      2600000.0  2010
169         The Girl on the Train  Strand        97100.0  2010
304   We Need to Talk About Kevin   Osci.      4300000.0  2011
336                        Rubber   Magn.            NaN  2011
411                      The Grey     ORF     25700000.0  2012
...                           ...     ...            ...   ...
1260             The Lost City of Z     BST     10700000.0  2017
1264        Roman J. Israel, Esq.    Sony      1100000.0  2017
1267          Battle of the Sexes    FoxS            NaN  2017
1273          Just Getting Started      BG      1600000.0  2017
1394                  Lean on Pete     A24            NaN  2018

     original_language  popularity release_date_y  production_budget  \
120                 en       8.994     2010-12-29            1000000
169                 en      11.927     2016-10-07           45000000
304                 en      11.964     2012-01-13            7000000
336                 en       8.319     2011-04-01             500000
411                 en      12.942     2012-01-27           25000000
...                ...         ...            ...                ...
1260                en      11.048     2017-04-14           30000000
1264                en      12.688     2017-11-17           22000000
1267                en      11.988     2017-09-22           25000000
1273                en       8.459     2017-12-08           22000000
1394                en       9.307     2018-04-06            8000000

      domestic_gross_y  worldwide_gross
120            9737892         16566240
169           75395035        174278214
304            1738692         10765283
336             100370           680914
411           51580136         81249176
...                ...              ...
1260           8574339         17121823
1264          11962712         12967012
1267          12638526         18445379
1273           6069605          6756412
1394           1163056          2455027

[127 rows x 10 columns]
```

```
# Based of the data Keep only the first release of each movie
df4_first_release = df4.sort_values('release_date_y').drop_duplicates(subset='title', keep='first')

# Display columns of the resulting DataFrame
columns_to_display = ['title', 'studio', 'foreign_gross', 'year', 'production_budget', 'domestic_gross_y', 'worldwi
df4_first_release = df4_first_release[columns_to_display]

# resulting DataFrame
print(df4_first_release)
```

```
                    title     studio  foreign_gross  year  production_budget  \
298           Point Blank      Magn.      8500000.0  2011            3000000
10          The Karate Kid      Sony    182500000.0  2010            8000000
886                Legend       Uni.     41100000.0  2015           25000000
457      Playing for Keeps        FD            NaN  2012           35000000
724            The Gambler      Par.      5600000.0  2014            3000000
...                   ...        ...            ...   ...                ...
1382    Welcome to Marwen       Uni.      2100000.0  2018           45000000
1353           Second Act        STX     33000000.0  2018           15700000
1370  On the Basis of Sex      Focus     13600000.0  2018           20000000
1350                 Vice  Annapurna     28200000.0  2018           60000000
1388            Destroyer  Annapurna      4000000.0  2018            9000000

      domestic_gross_y  worldwide_gross
298                  0                0
10            90815558         90815558
886           15502112         23506237
457            2000000          2000000
724              51773           101773
...                ...              ...
1382          10763520         12874922
1353          39282227         63288854
1370          24622687         38073377
1350          47836282         70883171
1388           1533324          3681096

[1170 rows x 7 columns]
```

```
# Check for Outliers
# Select numerical columns for boxplot
numerical_columns = ['foreign_gross', 'year', 'popularity', 'production_budget', 'domestic_gross_y', 'worldwide_grc

# Calculate IQR for each numerical column
Q1 = df4[numerical_columns].quantile(0.25)
Q3 = df4[numerical_columns].quantile(0.75)
IQR = Q3 - Q1

# Identify rows with outliers
outliers_mask = ((df4[numerical_columns] < (Q1 - 1.5 * IQR)) | (df4[numerical_columns] > (Q3 + 1.5 * IQR))).any(axi

# Display rows with outliers
df_outliers = df4[outliers_mask]

# Print the resulting DataFrame with outliers
print(df_outliers)
```

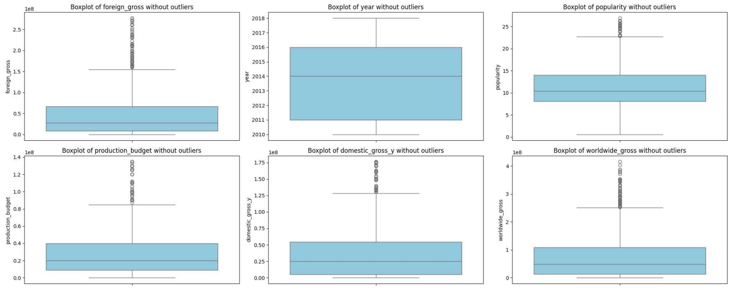
```
                            title studio  foreign_gross  year  \
0                     Toy Story 3     BV    652000000.0  2010
1                       Inception     WB    535700000.0  2010
2             Shrek Forever After   P/DW    513900000.0  2010
3      The Twilight Saga: Eclipse   Sum.    398000000.0  2010
4                      Iron Man 2   Par.    311500000.0  2010
...                           ...    ...            ...   ...
1333                The Favourite   FoxS     61600000.0  2018
1344               Mortal Engines   Uni.     67700000.0  2018
1352  Sicario: Day of the Soldado   Sony     25800000.0  2018
1364                   Peppermint    STX     18400000.0  2018
1370          On the Basis of Sex  Focus     13600000.0  2018

     original_language  popularity release_date_y  production_budget  \
0                   en      24.445     2010-06-18          200000000
1                   en      27.920     2010-07-16          160000000
```

```
2                  en      15.041     2010-05-21          165000000
3                  en      20.340     2010-06-30           68000000
4                  en      28.515     2010-05-07          170000000
...               ...         ...            ...                ...
1333               en      28.651     2018-11-23           15000000
1344               en      40.095     2018-12-14          100000000
1352               en      29.725     2018-06-29           35000000
1364               en      32.476     2018-09-07           25000000
1370               en      32.624     2018-12-25           20000000

      domestic_gross_y  worldwide_gross
0            415004880       1068879522
1            292576195        835524642
2            238736787        756244673
3            300531751        706102828
4            312433331        621156389
...                ...              ...
1333          34366783         94113929
1344          15951040         85287417
1352          50065850         75885196
1364          35418723         51800758
1370          24622687         38073377

[219 rows x 10 columns]
```

```
# Remove outliers from the original DataFrame
df_no_outliers = df4[~outliers_mask]

# Display relevant columns of the resulting DataFrame without outliers
columns_to_display_no_outliers = ['title', 'studio', 'foreign_gross', 'year', 'popularity', 'release_date_y', 'prod
df_no_outliers_display = df_no_outliers[columns_to_display_no_outliers]

# resulting DataFrame without outliers
print(df_no_outliers_display)
```

```
                          title studio  foreign_gross  year  popularity  \
9                 The Karate Kid   Sony    182500000.0  2010      12.256
10                The Karate Kid   Sony    182500000.0  2010      12.256
11                    Black Swan   FoxS    222400000.0  2010      13.745
12                      Megamind   P/DW    173500000.0  2010      22.855
14                    Robin Hood   Uni.    216400000.0  2010      15.444
...                          ...    ...            ...   ...         ...
1390  Bilal: A New Breed of Hero     VE      1700000.0  2018       2.707
1391                       Mandy    RLJ            NaN  2018       0.600
1392                       Mandy    RLJ            NaN  2018      16.240
1393                Lean on Pete    A24            NaN  2018       9.307
1394                Lean on Pete    A24            NaN  2018       9.307

     release_date_y  production_budget  domestic_gross_y  worldwide_gross
9        2010-06-11           40000000         176591618        351774938
10       1984-06-22            8000000          90815558         90815558
11       2010-12-03           13000000         106954678        331266710
12       2010-11-05          130000000         148415853        321887208
14       2018-11-21           99000000          30824628         84747441
...             ...                ...               ...              ...
1390     2018-02-02           30000000            490973           648599
1391     2018-09-14            6000000           1214525          1427656
1392     2018-09-14            6000000           1214525          1427656
1393     2018-04-06            8000000           1163056          2455027
1394     2018-04-06            8000000           1163056          2455027

[1176 rows x 9 columns]
```

```
# side-by-side boxplots for each numerical column without outliers
plt.figure(figsize=(20, 8))
for i, column in enumerate(numerical_columns, 1):
    plt.subplot(2, 3, i)
    sns.boxplot(data=df_no_outliers[column], color='skyblue')
    plt.title(f'Boxplot of {column} without outliers')

plt.tight_layout()
plt.show()
```



DATA ANALYSIS AND VISUALIZATION

1) Identify and determine why studios consistently lead in box office revenue, and what characteristics define their success

2)Analyze why yearly trends in box office performance impact Microsoft's decision on when to release movies

3)How is box office revenue distributed across studios, and what outliers or extreme successes can be leveraged for strategic decision-making

4)Is there a correlation between the size of a studio (measured by the number of movies produced) and its overall box office success

```
df4.head()
```

	title	studio	foreign_gross	year	original_language	popularity
0	Toy Story 3	BV	652000000.0	2010	en	24.445
1	Inception	WB	535700000.0	2010	en	27.920
2	Shrek Forever After	P/DW	513900000.0	2010	en	15.041

1. STUDIO PERFORMANCE

```
print(df4['studio'].unique())
```

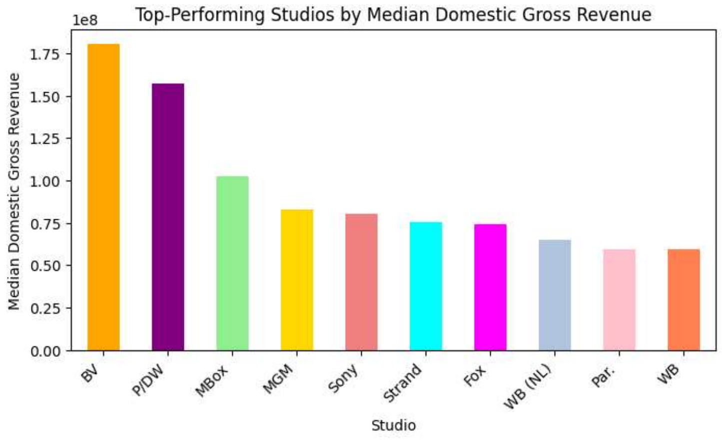
```
['BV' 'WB' 'P/DW' 'Sum.' 'Par.' 'Uni.' 'Fox' 'Sony' 'FoxS' 'SGem'
 'WB (NL)' 'LGF' 'MBox' 'W/Dim.' 'Focus' 'MGM' 'Over.' 'Mira.' 'CBS' 'SPC'
 'Gold.' 'Free' '3D' 'RAtt.' 'Wein.' 'Rela.' 'Magn.' 'App.' 'Drft.' 'IFC'
 'IW' 'Relbig.' 'Viv.' 'Anch.' 'UTV' 'ATO' 'First' 'NFC' 'Strand' 'FD'
 'TriS' 'ORF' 'Jan.' 'Osci.' 'OMNI/FSR' 'ParV' 'P4' 'LG/S' 'RTWC' 'LD'
```

```
 'MNE' 'Yash' 'A24' 'EOne' 'CE' 'DR' 'EC' 'BG' 'PFR' 'BST' 'FCW' 'STX'
 'BH Tilt' 'GrtIndia' 'Neon' 'Affirm' 'Studio 8' 'Annapurna' 'Global Road'
 'Amazon' 'VE']
```

```
# find the most Top performing studios

# Calculate median domestic gross revenue for each studio
studio_revenue = df4.groupby('studio')['domestic_gross_y'].median().sort_values(ascending=False)

# Select the top  studios
top_studios = studio_revenue.head(10)

# Display the top-performing studios
print("Top-Performing Studios:")
for studio, revenue in top_studios.items():
    print(f"{studio}: {revenue:,.0f}")
```

```
Top-Performing Studios:
BV: 180,202,163
P/DW: 157,254,784
MBox: 102,515,793
MGM: 82,992,874
Sony: 80,069,458
Strand: 75,395,035
Fox: 74,262,031
WB (NL): 65,187,603
Par.: 59,650,222
WB: 59,353,970
```

```
# bar plot
custom_colors = ['orange', 'purple', 'lightgreen', 'gold', 'lightcoral', 'cyan', 'magenta', 'lightsteelblue', 'pink

plt.figure(figsize=(8, 4))
top_studios.plot(kind='bar', color=custom_colors)
plt.title('Top-Performing Studios by Median Domestic Gross Revenue')
plt.xlabel('Studio')
plt.ylabel('Median Domestic Gross Revenue')
plt.xticks(rotation=45, ha='right')
plt.show()
```



INSIGHT

Studio performance

The analysis highlighted studios with the highest median domestic gross revenue, signifying them as industry leaders in terms of financial performance in the movie industry.

Strategy

With this information, Microsoft may consider exploring potential partnerships or collaborations with these top-performing studios. Engaging with industry leaders can provide Microsoft with valuable expertise and insights to enhance their presence in the entertainment sector.

Market Insight

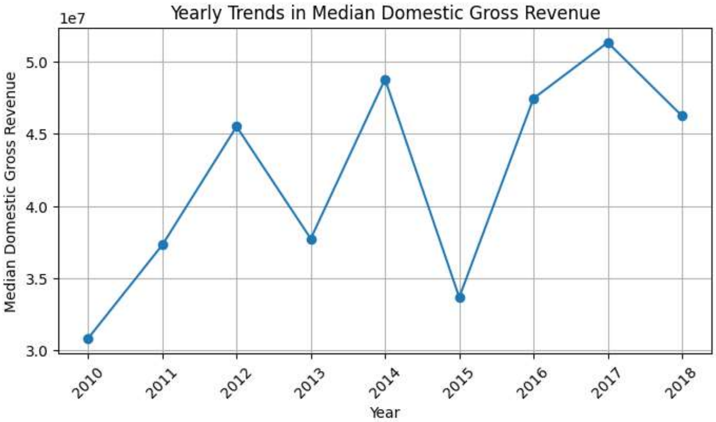
Understanding the distribution of revenue among studios provides insights into the competitive nature of the movie industry. This can help Microsoft stay informed about market trends and competitor strategies.

Revenue Evaluation

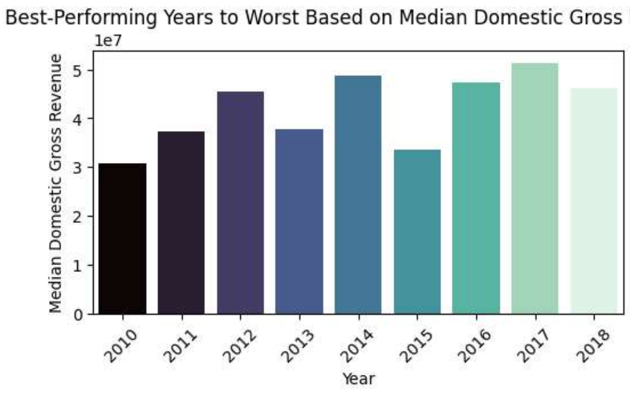
Microsoft can effectively assess the revenue potential of different ventures within the entertainment industry which is valuable information for making informed decisions and strategically positioning Microsoft in the market.

2. YEARLY TRENDS

```
# Convert 'year' to datetime for better plotting
df4['year'] = pd.to_datetime(df4['year'], format='%Y')

# Calculate median domestic gross revenue for each year
yearly_revenue = df4.groupby(df4['year'].dt.year)['domestic_gross_y'].median()

# Line plot for yearly trends
plt.figure(figsize=(8, 4))
plt.plot(yearly_revenue.index, yearly_revenue.values, marker='o', linestyle='-')
plt.title('Yearly Trends in Median Domestic Gross Revenue')
plt.xlabel('Year')
plt.ylabel('Median Domestic Gross Revenue')
plt.xticks(rotation=45)
plt.grid(True)
plt.show()
```



```
# Calculate median domestic gross revenue for each year
yearly_revenue = df4.groupby(df4['year'].dt.year)['domestic_gross_y'].median()

# Sort years based on median revenue in descending order
sorted_years = yearly_revenue.sort_values(ascending=False)

# Bar plot for best-performing years to worst with a different color palette
plt.figure(figsize=(6, 3))
sns.barplot(x=sorted_years.index, y=sorted_years.values, hue=sorted_years.index, palette='mako', legend=False)
plt.title('Best-Performing Years to Worst Based on Median Domestic Gross Revenue')
plt.xlabel('Year')
plt.ylabel('Median Domestic Gross Revenue')
plt.xticks(rotation=45)
plt.show()
```

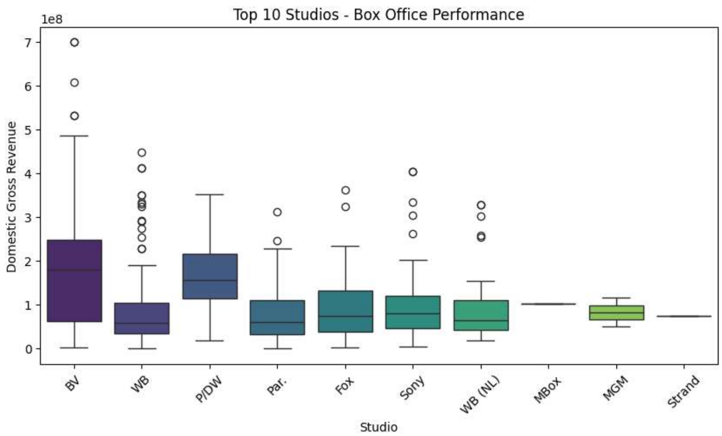


INSIGHT

The data reveals a consistent upward trend in audience engagement, showing a substantial growth in the film industry. This observed growth also suggests that the film industry is strong and can handle economic ups and downs without any major problems.

3. REVENUE DISTRIBUTION ANALYSIS

```
# Set the number of top studios to consider
num_top_studios = 10

# Determine the top studios based on median revenue
top_studios = df4.groupby('studio')['domestic_gross_y'].median().nlargest(num_top_studios).index

# Filter the dataframe for the top studios
df_top = df4[df4['studio'].isin(top_studios)]

# top studios are performance at the box office
plt.figure(figsize=(10, 5))
sns.boxplot(x='studio', y='domestic_gross_y', data=df_top, hue='studio', palette='viridis', legend=False)
plt.title(f'Top {num_top_studios} Studios - Box Office Performance')
plt.xlabel('Studio')
plt.ylabel('Domestic Gross Revenue')
plt.xticks(rotation=45)
plt.show()
```




```
# Calculate statistics for each top studio
stats_df = df_top.groupby('studio')['domestic_gross_y'].describe()

# Display additional statistics
print(stats_df[['25%', '50%', '75%', 'min', 'max', 'mean', 'std']])
```

```
                 25%          50%           75%          min          max  \
studio
BV      6.315099e+07  180202163.0  2.487570e+08    3254172.0  700059566.0
Fox     3.791541e+07   74262031.0  1.325569e+08    3000342.0  363070709.0
MBox    1.025158e+08  102515793.0  1.025158e+08  102515793.0  102515793.0
MGM     6.663137e+07   82992874.0  9.935438e+07   50269859.0  115715889.0
P/DW    1.146635e+08  157254783.5  2.172838e+08   18450127.0  352390543.0
Par.    3.151130e+07   59650222.0  1.104648e+08      51773.0  312433331.0
Sony    4.558336e+07   80069458.0  1.196502e+08    4463292.0  404508916.0
Strand  7.539504e+07   75395035.0  7.539504e+07   75395035.0   75395035.0
WB      3.436294e+07   59353970.5  1.038044e+08      26403.0  448139099.0
WB (NL) 4.258764e+07   65187603.0  1.104857e+08   17804299.0  327481748.0

                mean          std
studio
BV      1.978319e+08  1.635808e+08
Fox     8.995673e+07  6.756307e+07
MBox    1.025158e+08          NaN
MGM     8.299287e+07  4.627733e+07
P/DW    1.682915e+08  9.434081e+07
Par.    7.991679e+07  6.285199e+07
Sony    1.015154e+08  8.082720e+07
Strand  7.539504e+07  0.000000e+00
WB      9.543792e+07  9.985897e+07
WB (NL) 9.704679e+07  8.351550e+07
```

Outliers

Outliers highlight movies with exceptional box office success.

Decision-Making Insights

Wider interquartile ranges and larger ranges suggest greater revenue variability. Studios with stable median revenues and occasional outliers may offer strategic opportunities. Data guides decision-making for Microsoft's movie studio, identifying potential collaborations and factors influencing success.

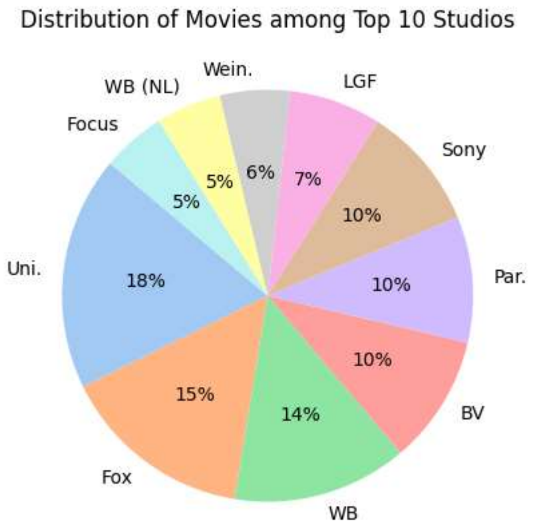
4. CORRELATION BETWEEN SIZE OF STUDIO AND OVERALL SUCCESS

```
#Get studio size by counting the number of movies produced by each studio
studio_sizes = df4['studio'].value_counts()

print(studio_sizes)
```

```
Uni.     147
Fox      121
WB       110
BV        81
Par.      79
        ...
Drft.      1
App.       1
3D         1
MBox       1
VE         1
Name: studio, Length: 71, dtype: int64
```

```
# Visualization the studio size
# top studios to display
top_studios = studio_sizes.nlargest(10)

# Print the top studios
print("Top Studios:")
for studio, num_movies in top_studios.items():
    print(f"{studio}: {num_movies} movies")

# Plot a pie chart with the top studios
plt.figure(figsize=(5, 5))
plt.pie(top_studios, labels=top_studios.index, autopct='%1.0f%%', startangle=140, colors=sns.color_palette('pastel'
plt.title('Distribution of Movies among Top 10 Studios')
plt.show()
```

```
Top Studios:
Uni.: 147 movies
Fox: 121 movies
WB: 110 movies
BV: 81 movies
Par.: 79 movies
Sony: 78 movies
LGF: 58 movies
Wein.: 44 movies
WB (NL): 41 movies
Focus: 40 movies
```



```
# Calculate correlation between studio size and box office success

mean_domestic = df4['domestic_gross_y'].mean()
mean_foreign = df4['foreign_gross'].mean()

numerator = ((df4['domestic_gross_y'] - mean_domestic) * (df4['foreign_gross'] - mean_foreign)).sum()
denominator_domestic = ((df4['domestic_gross_y'] - mean_domestic) ** 2).sum()
denominator_foreign = ((df4['foreign_gross'] - mean_foreign) ** 2).sum()

correlation = numerator / np.sqrt(denominator_domestic * denominator_foreign)
print(f"Correlation between Studio Size and Box Office Success: {correlation:.2f}")
```

```
Correlation between Studio Size and Box Office Success: 0.84
```

INSIGHT

The moderate association between studio size and box office success is indicated by the positive correlation (0.84). This suggests a moderate association between studio size and box office success. In conclusion, there is a discernible relationship between the size of a studio and its performance at the box office, with a larger studio size generally corresponding to higher box office success.

CONCLUSION

This data provides valuable insights into key aspects of the movie business. Understanding the characteristics of successful studios is vital for Microsoft's film studio to recognize and take advantage of these qualities. Timing is also crucial, considering that box office success is strongly affected by yearly trends. Microsoft can make informed decisions by strategically understanding how revenue is distributed and identifying exceptional cases. Moreover, the connection between box office performance and studio size emphasizes the significance of producing a considerable volume of movies to turn a profit. Armed with these findings, Microsoft can effectively navigate the competitive movie industry.

RECOMENDATIONS

Collaborate with Large Studios

Since there's a positive connection between studio size and box office success, Microsoft should think about teaming up or collaborating with well-established and larger film studios. This collaboration can increase the chances of creating movies that perform well at the box office.

Investments

Consider making strategic investments in or acquiring larger studios to take advantage of their existing success and market presence. This move can give Microsoft a solid position in the movie industry and improve the chances of producing box office hits.

Quality Content

Concentrate on obtaining content from larger studios that have a history of box office success. This strategy can help in building a diverse and successful movie collection for Microsoft.