Final Project Submission

Please fill out:

- Student name: `William Muthama`
- Student pace: `part time hybrid`
- Scheduled project review date/time: `23rd July to 30th July`
- Instructor name: `Anthonny Muiko`
- Blog post URL: `https://github.com/WILLY-GUSH/dsc-phase-2-project-v3`

Overview

This analysis aims to guide Trupress's entry into film production by examining data from the film industry to identify the optimal director, release month, and genres for a high Return on Investment film.

Business Problem

Trupress plans to start a movie studio to create original content. Using data from IMDb and The Numbers, I will analyze various films to determine the best directors, release months, and genres for achieving the highest Return on Investment.

Data Understanding

The data sources include:

-`IMDB`

-`The Numbers`

These datasets provide information on film titles, release dates, genres, gross profits, and production budgets. Combining this data will help identify the most profitable options for Trupress's new movie studio.

```
# Import libraries
import pandas as pd
import sqlite3
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

# Load IMDb data
conn = sqlite3.connect("im.db")
query = """
SELECT
    mb.primary_title AS movie_title,
    mb.genres,
    p.primary_name AS director_name
```

```
FROM movie_basics AS mb
JOIN directors AS d ON mb.movie_id = d.movie_id
JOIN persons AS p ON d.person_id = p.person_id
GROUP BY mb.primary_title
HAVING primary_profession LIKE '%director%'
"""
imdb = pd.read_sql(query, conn)

# Load The Numbers data
tn_mb = pd.read_csv('bom.movie_gross.csv')

# Display data information
imdb.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 121179 entries, 0 to 121178
Data columns (total 3 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   movie_title    121179 non-null  object
 1   genres         118367 non-null  object
 2   director_name  121179 non-null  object
dtypes: object(3)
memory usage: 2.8+ MB
```

```
# Display data information
tn_mb.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
```

IMDB Data Overview:

The IMDB dataset, which is the primary source for this project, includes records from the movie_basics and persons tables. It features over 120,000 film titles `(movie_title)`, various genres `(genres)`, and directors' names `(director_name)`.

```
# Display the first five rows
imdb.head()
```

```
             movie_title                    genres
director_name
0  !Women Art Revolution               Documentary  Lynn Hershman-
Leeson
1       #1 Serial Killer                    Horror          Stanley
Yung
2                    #5  Biography,Comedy,Fantasy           Ricky
Bardy
3             #50Fathers                    Comedy   Joddy Eric
Matthews
4                   #66                     Action          Asun
Mawardi
```

```python
# Extract genres and make calculations of each unique genre
imdb['genres'].value_counts()
```

```
Documentary                  28141
Drama                        17947
Comedy                        7812
Horror                        3455
Comedy,Drama                  2949
                             ...
Family,History,Mystery           1
Action,Animation,Music           1
Animation,Crime                  1
Animation,History,Horror         1
Crime,History,Mystery            1
Name: genres, Length: 1035, dtype: int64
```

```python
# first 20 entries
imdb['director_name'].value_counts()[:20]
```

```
Omer Pasha                     62
Stephan Düfel                  48
Rajiv Chilaka                  47
Larry Rosen                    45
Graeme Duane                   44
Gérard Courant                 44
Claudio Costa                  42
Nayato Fio Nuala               40
Eckhart Schmidt                36
Tetsuya Takehora               33
Charlie Minn                   29
Yoshikazu Katô                 27
Paul T.T. Easter               27
David DeCoteau                 26
Philip Gardiner                26
Narinderpal Singh Chandok      26
Ram Gopal Varma                25
Kazuyoshi Sekine               25
```

```
Mototsugu Watanabe          25
Manny Velazquez             25
Name: director_name, dtype: int64
```

Data Cleaning

IDBM Data Cleaning

For The Numbers dataset, I will rename the columns, extract the release month, remove unnecessary columns, convert financial columns to floats, and reformat the foreign gross to a more readable number.

Additionally, I'll remove records without domestic or foreign gross profit.

```
# Rename the movie column
tn_mb.rename(columns={'title': 'movie_title'}, inplace=True)

# Display the columns
tn_mb.columns
```

```
Index(['movie_title', 'studio', 'domestic_gross', 'foreign_gross',
'year'], dtype='object')
```

```
# Extract the release month from the release date
tn_mb['year'] = tn_mb['year'].astype(str)

# Convert financial columns to float
tn_mb['domestic_gross'] = tn_mb['domestic_gross'].replace('[\$,]', '',
regex=True).astype(float)
tn_mb['foreign_gross'] = tn_mb['foreign_gross'].replace('[\$,]', '',
regex=True).astype(float)

# Remove records with both domestic and worldwide gross equal to 0
tn_mb = tn_mb[(tn_mb['domestic_gross'] != 0) & (tn_mb['foreign_gross']
!= 0)]
```

Merging Datasets

Combining the data from The Numbers and IMDB allows for a unified dataset for feature engineering and analysis. I'll exclude unmatched records to avoid missing values.

```
tn_mb.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3387 entries, 0 to 3386
Data columns (total 5 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   movie_title     3387 non-null   object
```

```
 1   studio          3382 non-null   object
 2   domestic_gross  3359 non-null   float64
 3   foreign_gross   2037 non-null   float64
 4   year            3387 non-null   object
dtypes: float64(2), object(3)
memory usage: 158.8+ KB
```

```
tn_mb.head()
```

```
                                   movie_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    652000000.0  2010
1    691300000.0  2010
2    664300000.0  2010
3    535700000.0  2010
4    513900000.0  2010
```

```
# Merge datasets on the 'movie_title' column
movie_data = pd.merge(tn_mb, imdb, on='movie_title', how='inner')

# Create ROI column
movie_data['roi'] = movie_data['foreign_gross'] -
movie_data['domestic_gross']

# Reorder and drop unnecessary columns
movie_data = movie_data[['movie_title', 'year', 'genres',
'director_name', 'roi']]

movie_data.head(5)
```

```
                  movie_title  year                       genres  \
0                   Inception  2010     Action,Adventure,Sci-Fi
1         Shrek Forever After  2010  Adventure,Animation,Comedy
2  The Twilight Saga: Eclipse  2010     Adventure,Drama,Fantasy
3                     Tangled  2010  Adventure,Animation,Comedy
4             Despicable Me  2010     Animation,Comedy,Family

       director_name          roi
0  Christopher Nolan  243100000.0
```

```
1      Mike Mitchell  275200000.0
2        David Slade   97500000.0
3       Byron Howard  190200000.0
4       Chris Renaud   40100000.0
```

Analysis

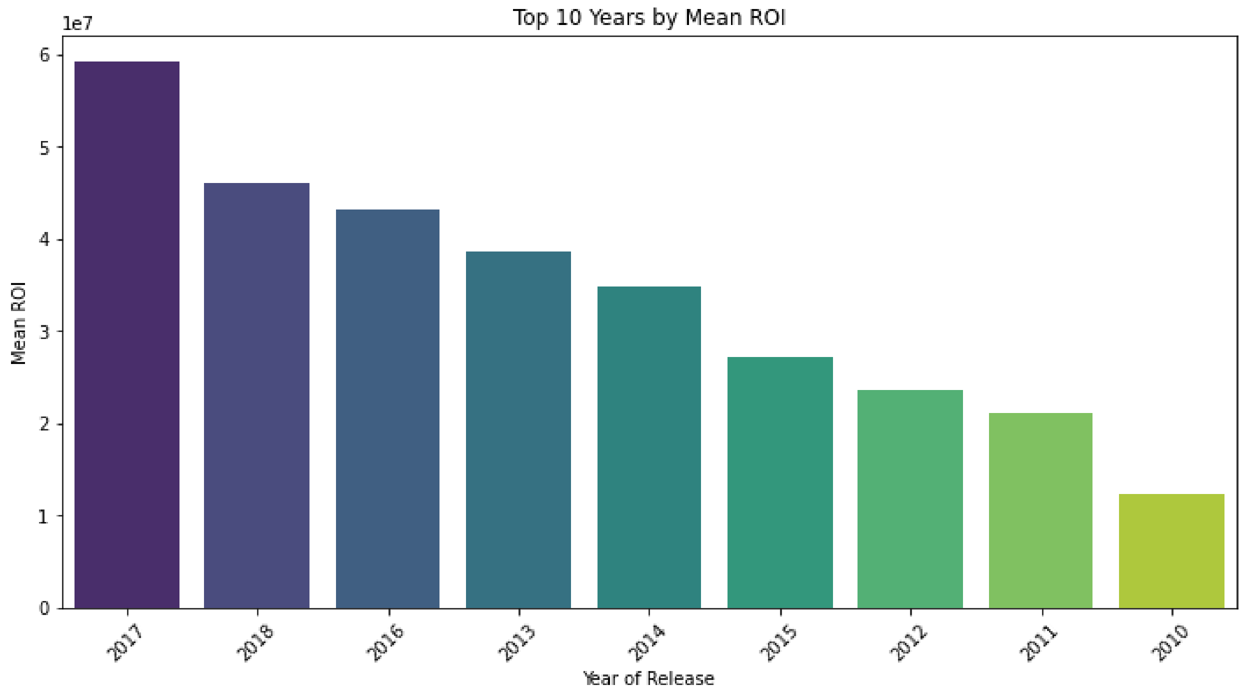
Most Profitable Year of Release

Films released in 2017, 2018, 2016 offer the highest mean Return on Investment, with November as a secondary option if delays occur.

```
# Group data by release year and calculate count, mean, and median of ROI
profit_years = movie_data.groupby('year')['roi'].agg(['count', 'mean', 'median'])

profit_years_mean = profit_years.sort_values(by='mean', ascending=False).head(10)
profit_years_mean
```

```
      count          mean      median
year
2017    128  5.913238e+07  10850000.0
2018    123  4.601018e+07   2900000.0
2016    144  4.306255e+07   6293250.0
2013    151  3.858987e+07   6353000.0
2014    161  3.476179e+07   3526000.0
2015    140  2.713382e+07   2531000.0
2012    178  2.348356e+07   1564000.0
2011    218  2.110560e+07   2800000.0
2010    175  1.228090e+07     16900.0
```

```
# Reset the index to have 'year' as a column
profit_years_mean = profit_years_mean.reset_index()

# Plot the bar chart
plt.figure(figsize=(12, 6))
sns.barplot(x='year', y='mean', data=profit_years_mean, palette='viridis')

# Add labels and title
plt.xlabel('Year of Release')
plt.ylabel('Mean ROI')
plt.title('Top 10 Years by Mean ROI')
plt.xticks(rotation=45)
plt.show()
```



Director Most Likely to Create a Film with a High Return on Investment

Based on data, the top directors likely to provide high Return on Investment are:

`Steven Spielberg`

`Ridley Scott`

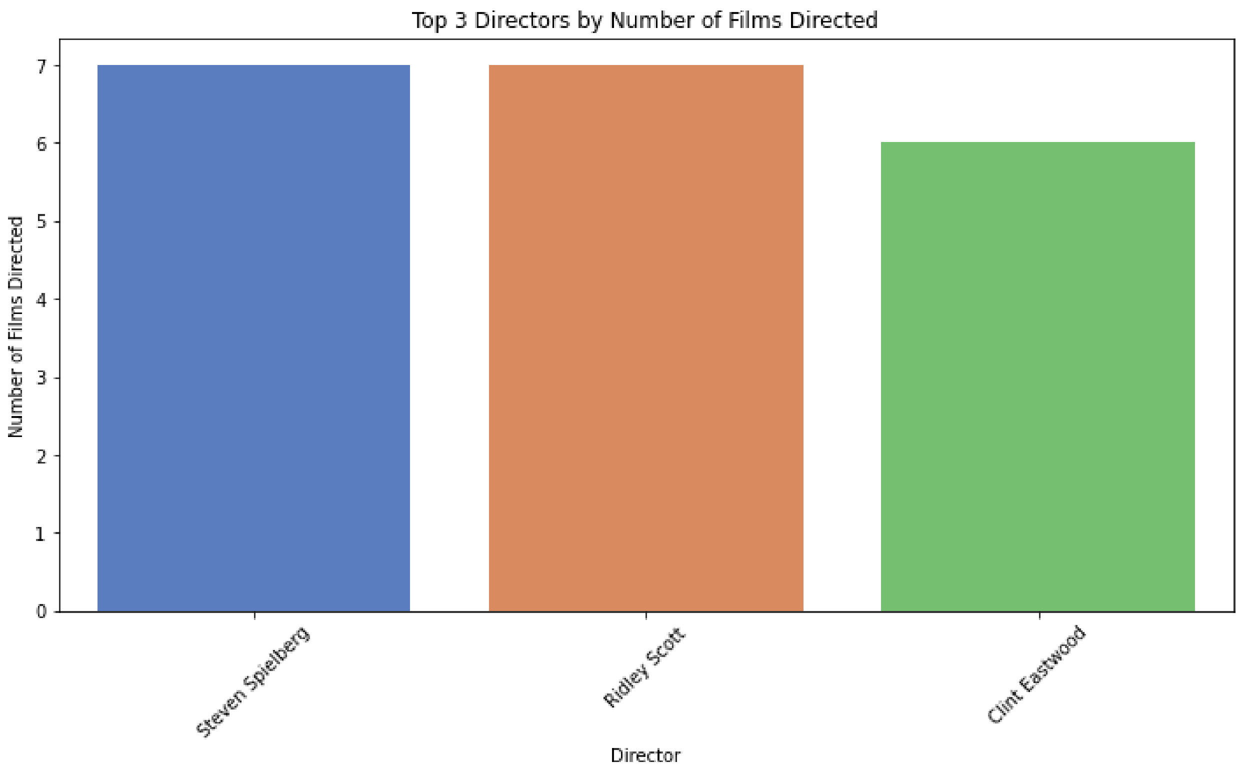
`Clint Eastwood`

```
# Group data by director and calculate count, mean, and median of ROI
profit_directors_avg = movie_data.groupby('director_name')
['roi'].agg(['count', 'mean', 'median'])

# Sort by the number of films directed and display top 5 directors
top_directors = profit_directors_avg.sort_values(by='count',
ascending=False).head(3)
top_directors
```

	count	mean	median
director_name			
Steven Spielberg	7	8.061429e+07	20900000.0
Ridley Scott	7	1.013000e+08	111100000.0
Clint Eastwood	6	-2.566667e+07	-12450000.0

```
# Reset the index to have 'director_name' as a column
top_directors = top_directors.reset_index()

# Plot the number of films directed by top directors
plt.figure(figsize=(12, 6))
```

```
sns.barplot(x='director_name', y='count', data=top_directors,
palette='muted')
plt.xlabel('Director')
plt.ylabel('Number of Films Directed')
plt.title('Top 3 Directors by Number of Films Directed')
plt.xticks(rotation=45)
plt.show()
```



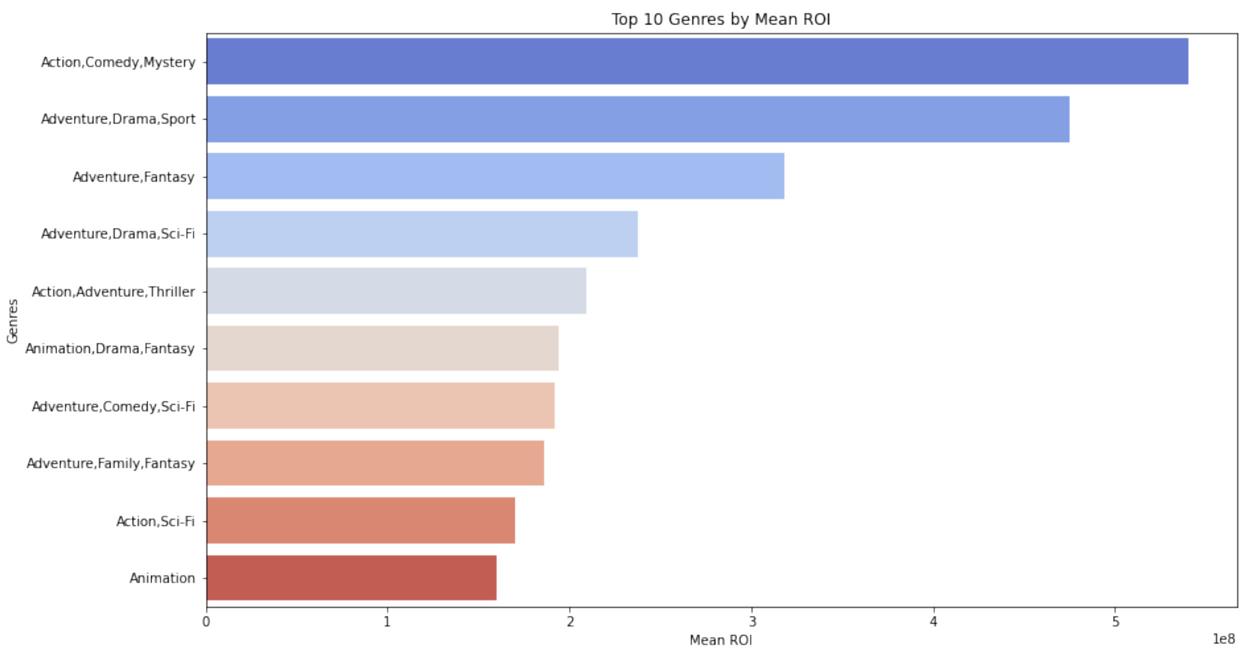
Return on Investment Based on Genre

Films with the combination of genres such as Action, Adventure, and Sci-Fi are most likely to provide a high Return on Investment.

```
# Group data by genres and calculate count, mean, and median of Return
on Investment
profit_genre_avg = movie_data.groupby('genres')
['roi'].agg(['count','mean', 'median'])

# Sort by mean Return on Investment and display top 10 genres
top_genres_mean = profit_genre_avg.sort_values(by='mean',
ascending=False).head(10)
top_genres_mean
```

	count	mean	median
genres			
Action,Comedy,Mystery	1	5.401000e+08	540100000.0

```
Adventure,Drama,Sport          1  4.750000e+08  475000000.0
Adventure,Fantasy              3  3.182333e+08  441600001.0
Adventure,Drama,Sci-Fi         2  2.373500e+08  237350000.0
Action,Adventure,Thriller     13  2.095154e+08  144900000.0
Animation,Drama,Fantasy        2  1.939500e+08  193950000.0
Adventure,Comedy,Sci-Fi        2  1.917470e+08  191747000.0
Adventure,Family,Fantasy       7  1.858857e+08  145400000.0
Action,Sci-Fi                  1  1.701000e+08  170100000.0
Animation                      1  1.602000e+08  160199999.0

# Reset the index to have 'genres' as a column
top_genres_mean = top_genres_mean.reset_index()

# Plot the bar chart
plt.figure(figsize=(14, 8))
sns.barplot(x='mean', y='genres', data=top_genres_mean,
palette='coolwarm')

# Add labels and title
plt.xlabel('Mean ROI')
plt.ylabel('Genres')
plt.title('Top 10 Genres by Mean ROI')
plt.show()
```



Conclusions and Recommendations

1Release Timing: Aim for film releases in `2017`, `2018`, `2016` offer.

2Director Selection: Focus on directors like Steven `Spielberg`, `Ridley Scott`, and `Clint Eastwood`.

`3`Genre Selection: Prioritize films with genres such as `Action, Comedy, Mystery` for the best Return on Investment.