Final Project Submission

Please fill out:

- Student name: Andrew Reusche
- Student pace: self paced
- Scheduled project review date/time: December 6, 2024 at 4pm
- Instructor name: Mark Barbor

Movie Data Analysis

Authors: Andrew Reusche

Overview

For over 100 years, countless movies have been produced and released, making them not only a staple of the American pastime but also a proven business venture that can return great profits if executed properly. However, what may have made a movie a successful investment years ago, may not be in line with the modern-day market. To help solve for that, this project uses Exploratory Data Analysis (E.D.A.)to highlight some attributes that modern successful films have had, in an effort to help guide those new to movie production.

Business Problem

My company recognizes that more large organizations are branching into the movie production business with mixed results in terms of investment success. Now our company too would like a share in the market, and they plan to achieve this through opening their own movie production studio, but currently do not know anything about creating movies, or what makes them successful in the modern market. To help guide the head of our company's new movie studio, I will use E.D.A. on existing movie data to generate insights into attributes that are commonly present in modern successful movies. The studio head can then use these insights to help decide factors that could make this new movie venture successful.

Data Understanding

To compile relevant information for this analysis, I drew and combined data from two of the largest and most well-known reputable movie information hubs: IMDB.com, and The-Numbers.com. These combined databases contain information on 139,457 movies from 1915 up to 2020, with each record including information on many factors such as their earnings, staff, and ratings.

Due to these datasets not including a complete list of all the details and factors that went into making and releasing these movies, we can only outline some factors that were commonly present in modern profitable movies instead of claiming the outright cause of success.

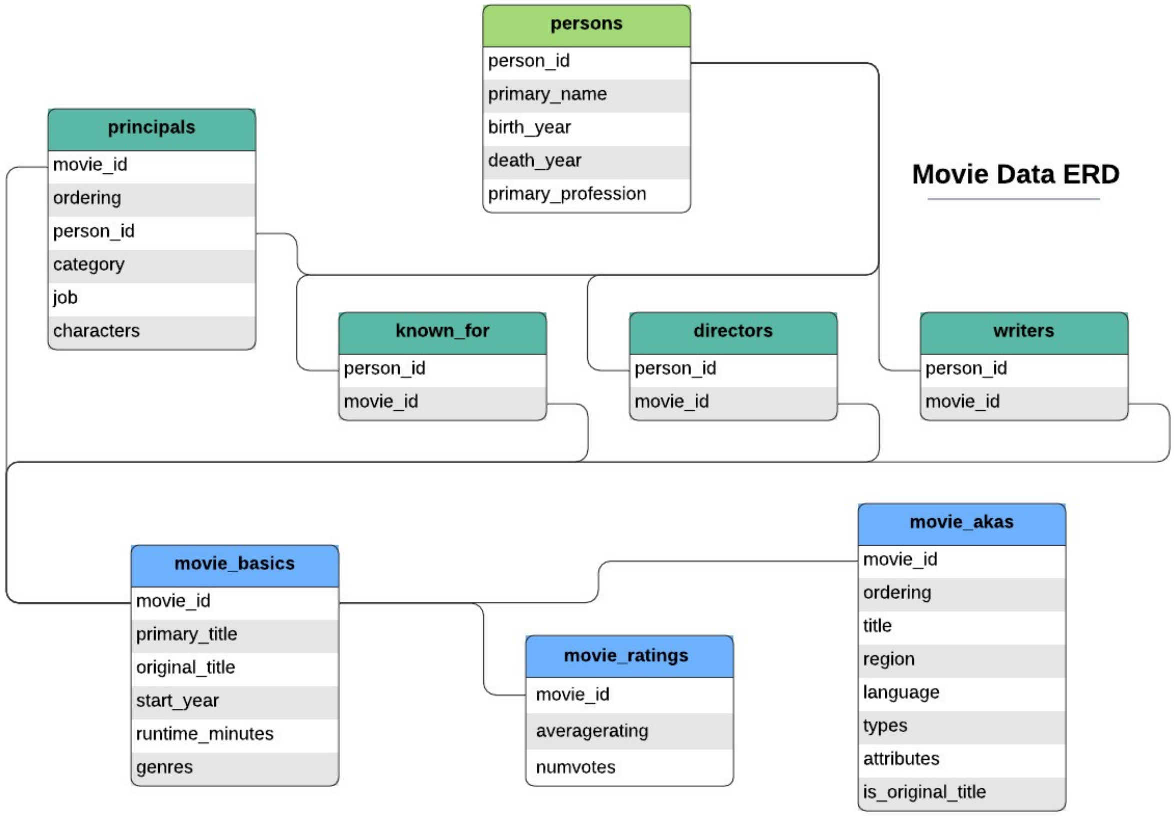
```
#Import the relevant libraries to help us view and manipulate the
data.

import numpy as np
import pandas as pd
import sqlite3

#read in the csv file from The-Numbers.com and save it as a Dataframe
TN = pd.read_csv('../zippedData/tn.movie_budgets.csv', index_col= 0)
TN.head()
```

```
    release_date                                        movie  \
id
1   Dec 18, 2009                                       Avatar
2   May 20, 2011  Pirates of the Caribbean: On Stranger Tides
3    Jun 7, 2019                                 Dark Phoenix
4    May 1, 2015                      Avengers: Age of Ultron
5   Dec 15, 2017            Star Wars Ep. VIII: The Last Jedi

   production_budget domestic_gross worldwide_gross
id
1       $425,000,000   $760,507,625  $2,776,345,279
2       $410,600,000   $241,063,875  $1,045,663,875
3       $350,000,000    $42,762,350    $149,762,350
4       $330,600,000   $459,005,868  $1,403,013,963
5       $317,000,000   $620,181,382  $1,316,721,747
```

```
#information on the TN dataset
TN.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 5782 entries, 1 to 82
Data columns (total 5 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   release_date       5782 non-null   object
 1   movie              5782 non-null   object
 2   production_budget  5782 non-null   object
 3   domestic_gross     5782 non-null   object
 4   worldwide_gross    5782 non-null   object
dtypes: object(5)
memory usage: 271.0+ KB
```

Here is an Entity Relatiohship Diagram (E.R.D.) for the SQL Datastructure I am about to bring in.




```
#read in the SQL file from IMDB.com and save it as a Dataframe

#connects to the SQLite database
conn = sqlite3.connect('../zippedData/im.db')

#a query to dictate what data is retrieved from the file
query= """
        SELECT mov.genres, mov.primary_title
        FROM movie_basics AS mov;
        """

#reads the data into a database
IMDB = pd.read_sql(query, conn)

#the database currently has multiple genres for each movie
#this changes so there is one instance (row) for each movie's genre
IMDB['genres'] = IMDB['genres'].str.split(',')
IMDB = IMDB.explode('genres', ignore_index=True)

#previews the first 10 rows of the dataset
IMDB.head(10)
```

```
      genres                     primary_title
0     Action                         Sunghursh
1      Crime                         Sunghursh
2      Drama                         Sunghursh
3  Biography  One Day Before the Rainy Season
4      Drama  One Day Before the Rainy Season
5      Drama       The Other Side of the Wind
6     Comedy                   Sabse Bada Sukh
7      Drama                   Sabse Bada Sukh
8     Comedy          The Wandering Soap Opera
9      Drama          The Wandering Soap Opera
```

```
#Info on the IMDB dataset
IMDB.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 234958 entries, 0 to 234957
Data columns (total 2 columns):
 #   Column         Non-Null Count   Dtype 
---  ------         --------------   ----- 
 0   genres         229550 non-null  object
 1   primary_title  234958 non-null  object
dtypes: object(2)
memory usage: 3.6+ MB
```

Movie Data

```
#rename column in IMDB so it can merge with other dataframes on
"movie"
IMDB.rename(columns={'primary_title': 'movie'}, inplace=True)

#merge the TN dataframe with the IMDB dataframe on the "movie" column
movies_df =TN.merge(IMDB, how='outer', on=['movie'])

#preview of the merged dataset
movies_df.head()
```

```
   release_date                                        movie  \
0  Dec 18, 2009                                       Avatar   
1  May 20, 2011  Pirates of the Caribbean: On Stranger Tides   
2  May 20, 2011  Pirates of the Caribbean: On Stranger Tides   
3  May 20, 2011  Pirates of the Caribbean: On Stranger Tides   
4   Jun 7, 2019                                 Dark Phoenix   

  production_budget domestic_gross worldwide_gross     genres
0      $425,000,000   $760,507,625  $2,776,345,279     Horror
1      $410,600,000   $241,063,875  $1,045,663,875     Action
2      $410,600,000   $241,063,875  $1,045,663,875  Adventure
3      $410,600,000   $241,063,875  $1,045,663,875    Fantasy
4      $350,000,000    $42,762,350    $149,762,350     Action
```

Data Preperation

Data Cleaning and Trimming

In this E.D.A., we will specifically be focussing a subset of this merged data, detailing movie production budget level, genre, and release date in relation to the domestic profitability of the 938 different movies released between 2015 and 2020. All three of these attributes can have highly variable options but are also factors that any new studio should easily be able to control when trying to make their first movie.

Narrowing the scope of movies we analyze down to only ones that were released in 2015 and later helps us focus more on what the current market trends are saying. Without this scope narrowing, the trend results may be skewed by what was popular 20 or 50+ years ago. Additionally, this scope could be altered to show movies released during any time period, to help represent the trends for that era.

As for why we will be focussing on domestic gross profits over worldwide gross profits, this movie studio is a new venture for our company, and as such it is important to keep goals manageable and obtainable. A solid marker that this new venture is moving in the right direction would be to first have local success, and then if we want to scale the operation up we can shift to the global market afterward.

To make the Dataframe easier to manipulate for analysis, I will drop irrelevant columns, and where needed, instead of estimating and imputing new values, I will drop all rows that are missing relevant data. This is done to avoid creating an over-representation bias in the dataset. Certain column values will also be modified to make them more manipulateable and filterable.

After these measures have been taken, our Dataframe will be ready for feature engineering, which can be used to extract more useful meaning from the dataset.

```
#gets rid of the 'worldwide_gross' column
movies_df= movies_df.drop('worldwide_gross', axis=1)

#gets rid of any duplicate rows in the dataset
movies_df= movies_df.drop_duplicates(keep='first')

#gets rid of any rows that NaN values in this listed subset of columns
movies_df= movies_df.dropna(subset=['movie',
'release_date','production_budget',
                                    'domestic_gross'])

#creates a row that extracts the year out of the release date
movies_df['year']= movies_df['release_date'].str[-4:].astype(int)

#filters the dataframe to only contain movies that were released in
2015 or later
movies_df= movies_df[movies_df['year']>= 2015]

#converts the strings in the monetary value columns to integers while
stripping unneeded characters
```

```
x=['production_budget','domestic_gross']

for col in x:
    movies_df[col] = movies_df[col].replace({'\$': '', ',': ''}, 
regex=True).\
                        astype(int)

#preview of the cleaned and filtered dataframe
movies_df.head()
```

```
  release_date                    movie  production_budget  
domestic_gross  \
4  Jun 7, 2019             Dark Phoenix          350000000   
42762350   
5  Jun 7, 2019             Dark Phoenix          350000000   
42762350   
6  Jun 7, 2019             Dark Phoenix          350000000   
42762350   
7  May 1, 2015  Avengers: Age of Ultron          330600000   
459005868   
8  May 1, 2015  Avengers: Age of Ultron          330600000   
459005868   

      genres  year  
4     Action  2019  
5  Adventure  2019  
6     Sci-Fi  2019  
7     Action  2015  
8  Adventure  2015  
```

```
#description of the data in this new dataset
movies_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 2497 entries, 4 to 11272
Data columns (total 6 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   release_date       2497 non-null   object
 1   movie              2497 non-null   object
 2   production_budget  2497 non-null   int64 
 3   domestic_gross     2497 non-null   int64 
 4   genres             2326 non-null   object
 5   year               2497 non-null   int64 
dtypes: int64(3), object(3)
memory usage: 136.6+ KB
```

Feature Engineering

Here I use feature engineering to create new columns that will extact more meaningful information from the dataset.

The dataframe is altered to create a new column that simply explains how profitable a movie is. If a movie spends more on its budget than was earned back that movie will be associated with an "Investment Loss". If a movie made more back than was spent on its budget, but not enough to double its investment amount, it will be associated with a "Profit Made". If a movie earns at least twice the amount back that was invested in the budget it will be associated with "Investment Doubled", putting it into the most successful of the movie earnings options.

```
#creates a column to show how much money was made by a movie,
#in relation to how much was spent on the budget
movies_df['profit_multiplier']=
movies_df['domestic_gross']/movies_df['production_budget']

#creates a function to bin all of the profits / losses into one of
three categories
def profitability(num):
    if 0 < num < 1:
        x='Investment Loss'
    elif 1 < num < 2:
        x='Profit Made'
    else:
        x='Investment Doubled'
    return x

#creates a new column by applying the binning function to the previous
profit column
movies_df['profitability']=
movies_df['profit_multiplier'].apply(lambda x: \

profitability(x))
```

The dataframe is altered to creat a new column that tells us what season of the year that a movie was released during. If people are more prone to going to see movies during certain times of the year, this could help highlight that trend.

```
#creates a new column that extracts the month from the release date
column
movies_df['Release_Month']= movies_df['release_date'].str[:3]

#creates a function that bins each month into 1 of 4 seasons
def month_to_season(month):
    season_dict= { 'Mar': 'Spring', 'Apr': 'Spring', 'May': 'Spring',
                   'Jun': 'Summer', 'Jul': 'Summer', 'Aug': 'Summer',
                   'Sep': 'Fall', 'Oct': 'Fall', 'Nov': 'Fall',
                   'Dec': 'Winter', 'Jan': 'Winter', 'Feb': 'Winter'}
    return season_dict.get(month)
```

```
#creates a new season column by applying the season function to the
month column
movies_df['Season']= movies_df['Release_Month'].apply( lambda x :
month_to_season(x))
```

The dataframe is altered again to create a new column that tells us the general amount of money that was spent on a movie production budget. These production budget level increments were found on the Production Budget blog of StudioBinder.com.

```
Link: https://www.studiobinder.com/blog/production-budget/
```

Here, if a movie's production budget is under 5,000,000 dollars we categorize it as "Low Budget". If the movie's budget is greater than or equal to 5,000,000 dollars but less than 50,000,000 dollars we categorize it as "Mid Level Budget", and if the movie's budget is over 50,000,000 dollars we categorize it as a "High End Budget".

If certain budgets tend to have more success than others this could help highlight that.

```
#creates a function that bins all movie budgets into 1 of 3 budget
levels
def budget_leveler(budget):
    if 0 < budget < 5000000:
        x='Low Budget'
    elif 5000000 <= budget <= 50000000:
        x='Mid Level Budget'
    else:
        x='High End Budget'
    return x

#creates a new budget-level column by applying the budget function to
the budget column
movies_df['level']= movies_df['production_budget'].apply(lambda x: \

budget_leveler(x))
```

Let's take a look at the updated dataset now that all the new columns have been engineered.

```
#previews the dataset
movies_df.head()

  release_date                   movie  production_budget
domestic_gross  \
4  Jun 7, 2019            Dark Phoenix          350000000
42762350
5  Jun 7, 2019            Dark Phoenix          350000000
42762350
6  Jun 7, 2019            Dark Phoenix          350000000
42762350
```

```
7  May 1, 2015  Avengers: Age of Ultron          330600000
459005868
8  May 1, 2015  Avengers: Age of Ultron          330600000
459005868

      genres  year  profit_multiplier    profitability Release_Month
Season  \
4     Action  2019           0.122178  Investment Loss           Jun
Summer
5  Adventure  2019           0.122178  Investment Loss           Jun
Summer
6     Sci-Fi  2019           0.122178  Investment Loss           Jun
Summer
7     Action  2015           1.388403      Profit Made           May
Spring
8  Adventure  2015           1.388403      Profit Made           May
Spring

             level
4  High End Budget
5  High End Budget
6  High End Budget
7  High End Budget
8  High End Budget

#dataset info
movies_df.info()

<class 'pandas.core.frame.DataFrame'>
Index: 2497 entries, 4 to 11272
Data columns (total 11 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   release_date       2497 non-null   object
 1   movie              2497 non-null   object
 2   production_budget  2497 non-null   int64
 3   domestic_gross     2497 non-null   int64
 4   genres             2326 non-null   object
 5   year               2497 non-null   int64
 6   profit_multiplier  2497 non-null   float64
 7   profitability      2497 non-null   object
 8   Release_Month      2497 non-null   object
 9   Season             2497 non-null   object
 10  level              2497 non-null   object
dtypes: float64(1), int64(3), object(7)
memory usage: 234.1+ KB
```

Analysis

We will use this dataframe to conduct three analysises that the new department head can use to help steer the direction the studio is going.

```
#import libraries used to graph this data
import matplotlib.pyplot as plt
%matplotlib inline
```

Budget Level Analysis

When making a movie, budget is often one of the first items that is brought up, and can directly affect everthing from your casting options, to scale of production. On the flip side, the more money you invest the more money you stand to loose if things don't work out. Movie budget levels can be broken down into three commonly recognized ranges: Low Level Budget (under 5,000,000 dollars), Mid Level Budget (between 5,000,000 dollars and 50,000,000 dollars), and High End Budget (over 50,000,000 dollars). This analysis displays how often (in the 2015-2020 market) these production budget levels result in a domestic investment net loss, investment profit, and even a profit that doubles the investment cost.

Here I prep the dataframe for this specific production budget analysis.

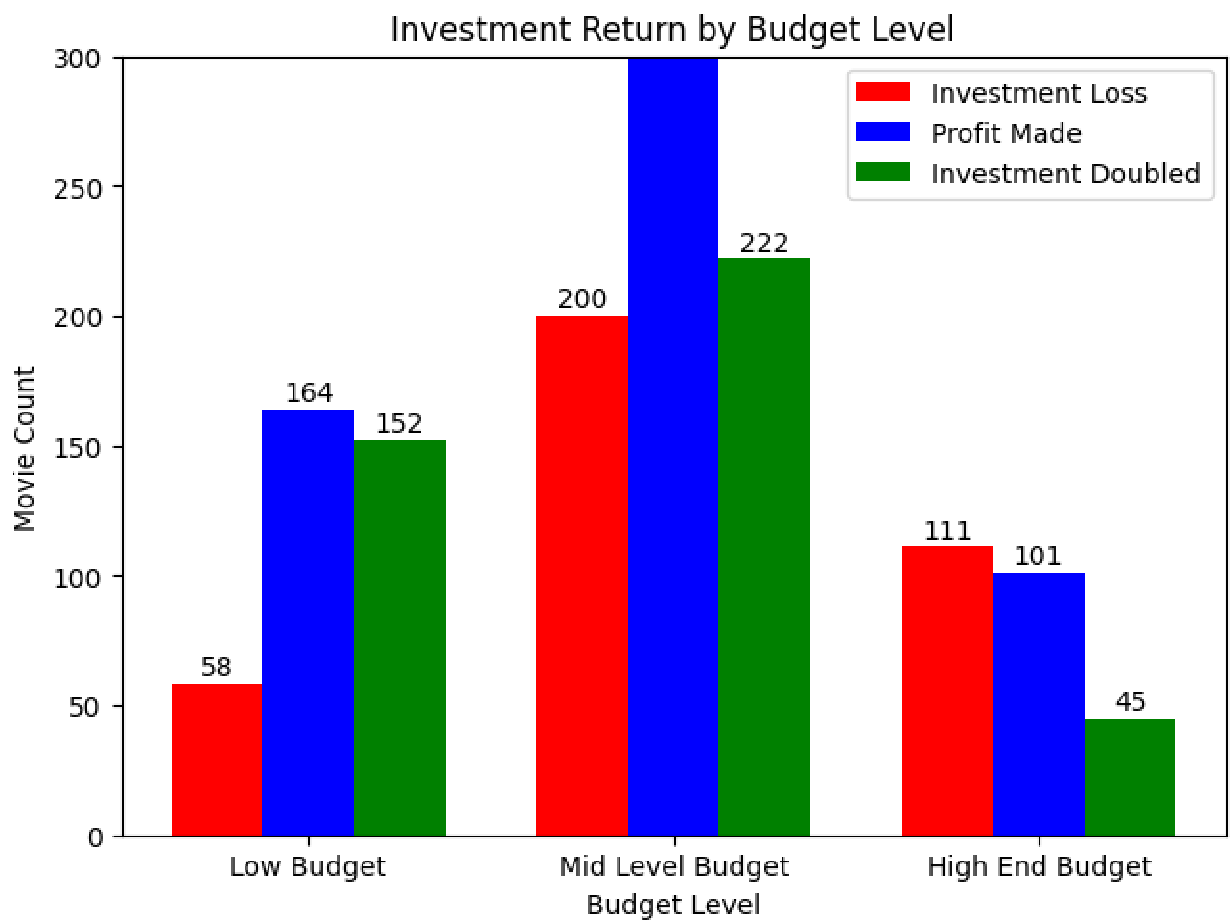
```
#creates a new dataframe from a subset of the movies dataframe
Budget_analysis=movies_df[['movie','level','profitability']]

#drops all duplicate rows in this dataframe
Budget_analysis=Budget_analysis.drop_duplicates(subset=['movie'],
keep='first')

#gets rid of the now unnecessary movie column
Budget_analysis= Budget_analysis.drop('movie', axis= 1)

#create a new dataframe grouped by budget-level that counts each
instance of profitability
budget_counts = Budget_analysis.groupby('level')['profitability']\
                .value_counts().unstack(fill_value=0)
budget_counts= budget_counts.reset_index()

#create a new dataframe that rearranges the columns and rows of the
above dataframe,
#to be in a sensible order
budget_counts2 = budget_counts[[budget_counts.columns[0],
budget_counts.columns[2],
                                budget_counts.columns[3],
budget_counts.columns[1]]] #rearanges the columns
budget_counts2 = pd.concat([budget_counts2.iloc[1:],
budget_counts2.iloc[:1]],
                           ignore_index=True) #rearanges the rows
budget_counts2= budget_counts2.set_index('level')
```

```
#updates the dataframe so that the "Profit Made" column now also
includes all the instances
# of movies that at least doubled their investment amount. This is a
better representation
#of how many movies actually made profit over the previous version.
budget_counts2['Profit Made']= budget_counts2['Profit Made']+\
                                budget_counts2['Investment Doubled']
budget_counts2= budget_counts2.reset_index()

#view the updated dataset we will use for analysis
budget_counts2

profitability             level  Investment Loss  Profit Made  \
0                    Low Budget               58          164
1              Mid Level Budget              200          304
2               High End Budget              111          101

profitability  Investment Doubled
0                             152
1                             222
2                              45
```

Now that the dataframe has been shaped for Production Budget Analysis we graph it to analize the results.

```
#extract values from the budget_counts2 dataframe
budgets = budget_counts2['level']
losses= budget_counts2['Investment Loss'].to_list()
profits= budget_counts2['Profit Made'].to_list()
doubles= budget_counts2['Investment Doubled'].to_list()

#organizes some of the extracted data into a callable dictionary
budget_counts = {
    'Investment Loss': losses,
    'Profit Made': profits,
    'Investment Doubled': doubles
    }

#prepares the X axis we will graph
x = np.arange(len(budgets)) #gives the number of different budget
levels
width = 0.25  #width of the bars to be displayed
multiplier = 0 #ticker variable to offset the bar positions in the
graph

colors = ['red', 'blue', 'green'] #the colors we will assign to the
profit levels

fig, ax = plt.subplots(layout='constrained') #creates figure and axis
```

```
for plotting

#a for loop to iterate through the dataset values, graphing them, and
grouping
#them by their budget level and associated profit result
for (attribute, measurement), color in zip(budget_counts.items(),
colors):
    offset = width * multiplier
    rects = ax.bar(x + offset, measurement, width, label=attribute,
color= color)
    ax.bar_label(rects, padding=1)
    multiplier += 1


ax.set_ylabel('Movie Count') #sets Y axis title
ax.set_xlabel('Budget Level') #sets X axis title
ax.set_title('Investment Return by Budget Level') #sets plot title
label
ax.set_xticks(x + width, budgets) #sets the X axis labels and position
ax.legend(loc='upper right', ncols=1) #dictates position and layout of
legend
ax.set_ylim(0, 300) #sets Y axis tick limit for viewing ease

#save the plot as a PNG file
plt.savefig("../images/Investment_Return_by_Budget_Level.png",
dpi=150)

#shows the plot
plt.show()
```



From the graph we can see that even though the Mid Level Budget is the most frequently used level from 2015-2020, the movies that used a Low Budget production budget (under 5,000,000 dollars) expierienced the lowest ratio of investment losses and the highest ratio of the production investment returning at least double it's amount in the domestic market.

Movie Genre Analysis

So many different types of movies exist. Some make us laugh, some make us cry, and others thrill or educate us. With so many options out there, it begs the question are certain movie genres more profiable than others? This analysis aims to show how profitable different movie genres are in the domestic 2015-2020 market.

Here I prep the dataframe for this specific movie genre analysis.

```
#creates a new dataframe from a subset of the movies dataframe
Genre_analysis=movies_df[['movie','genres','profitability']]

#drops all duplicate rows in this dataframe
Genre_analysis=Genre_analysis.drop_duplicates(keep='first')

#drops all rows of the 'genres' column that have NaN as the genre
Genre_analysis= Genre_analysis.dropna(subset=['genres'])
```

```
#gets rid of the now unnecessary movie column
Genre_analysis= Genre_analysis.drop('movie', axis= 1)

#create a new dataframe grouped by genres that counts each instance of
profitability
genre_counts = Genre_analysis.groupby('genres')
['profitability'].value_counts()\
                .unstack(fill_value=0)
#sorts the outputted rows by highest count of "Investment Doubled"
Instances
genre_counts= genre_counts.sort_values(by=['Investment Doubled'],
ascending=[False])
genre_counts= genre_counts.reset_index()

#creates a new dataframe from the top 10 most frequent "Investment
Doubled" genres
top_10_genre_counts= genre_counts.head(10)

#create a new dataframe reordering the dataframe columns into a
sensible order
top_10_genre_counts2 =
top_10_genre_counts[[top_10_genre_counts.columns[0],

top_10_genre_counts.columns[2],

top_10_genre_counts.columns[3],

top_10_genre_counts.columns[1]]]

#updates the dataframe so that the "Profit Made" column now also
includes all the instances
# of movies that at least doubled their investment amount. This is a
better representation
#of how many movies actually made profit over the previous version.
top_10_genre_counts2['Profit Made']= top_10_genre_counts2['Profit
Made']+ \
                                    top_10_genre_counts2['Investment
Doubled']

#view the updated dataset we will use for analysis
top_10_genre_counts2
```

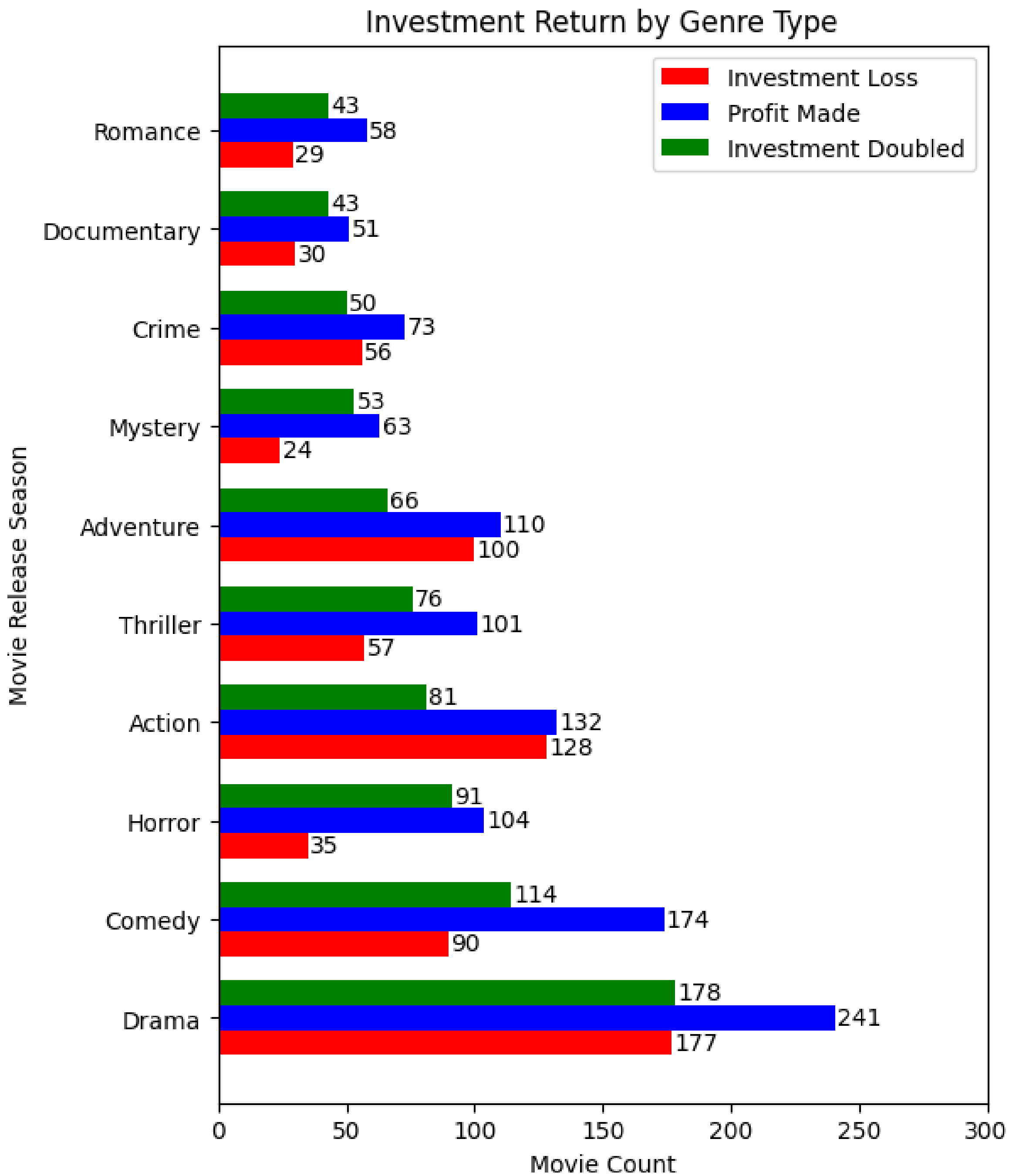
profitability	genres	Investment Loss	Profit Made	Investment Doubled
0	Drama	177	241	178
1	Comedy	90	174	114
2	Horror	35	104	

```
91
3                    Action              128          132
81
4                  Thriller               57          101
76
5                 Adventure              100          110
66
6                   Mystery               24           63
53
7                     Crime               56           73
50
8               Documentary               30           51
43
9                   Romance               29           58
43
```

Now that the dataframe has been shaped for the Movie Genre Analysis we graph it to analize the results.

```
#extract values from the top_10_genre_counts2 dataframe
genres = top_10_genre_counts2['genres']
losses = top_10_genre_counts2['Investment Loss'].to_list()
profits = top_10_genre_counts2['Profit Made'].to_list()
doubles = top_10_genre_counts2['Investment Doubled'].to_list()

#organizes some of the extracted data into a callable dictionary
genre_counts = {
    'Investment Loss': losses,
    'Profit Made': profits,
    'Investment Doubled': doubles
}

#prepares the y axis we will graph
y = np.arange(len(genres)) #gives the number of different genres
height = 0.25  #height of the bars to be displayed
multiplier = 0  #ticker variable to offset the bar positions in the
graph

#the colors we will assign to the profit levels
colors = ['red', 'blue', 'green']

#creates figure and axis for plotting
fig, ax = plt.subplots(layout='constrained', figsize=(6,7))

#a for loop to interate through the dateset values, graphing them, and
grouping
#them by their genre and associated profit result
for (attribute, measurement), color in zip(genre_counts.items(),
colors):
```

```
    offset = height * multiplier
    rects = ax.barh(y+ offset, measurement, height, label=attribute,
color=color)
    ax.bar_label(rects, padding=1)
    multiplier += 1


ax.set_xlabel('Movie Count') #sets X axis title
ax.set_ylabel('Movie Release Season') #sets Y axis title
ax.set_title('Investment Return by Genre Type') #sets graph title

ax.set_yticks(y+ height, genres) #sets the Y axis labels and position

ax.legend(loc='upper right', ncols=1) #dictates position and layout of
legend

ax.set_xlim(0, 300) #sets X axis tick limit for viewing ease

#save the plot as a PNG file
plt.savefig("../images/Investment_Return_by_Genre_Type.png", dpi=150)

#show the plot
plt.show()
```



Here we can see that Drama movies tend to be the most common kind of movie released in the 2015-2020 market, however, the amount of times Drama movies experience an investment loss in the Domestic market is just about even with the amount of times their profits double their investments. Alternatively, Horror movies seem to have a significantly lower ratio of movies that resulted in an investment loss, and a lot higher ratio of movies that resulted in earnings at least double what the production budget investment was.

Movie Release Season Analysis

Movies are constantly released throughout the year. People see them after work, on vacations, and when they get together with friends. With all the different potential times of the year that a movie can be released, is there a season that tends to result in higher profitability? Here I run this analysis for the 2015-2020 Domestic market.

Here I prep the dataframe for this specific Seasional Movie Release Date analysis.

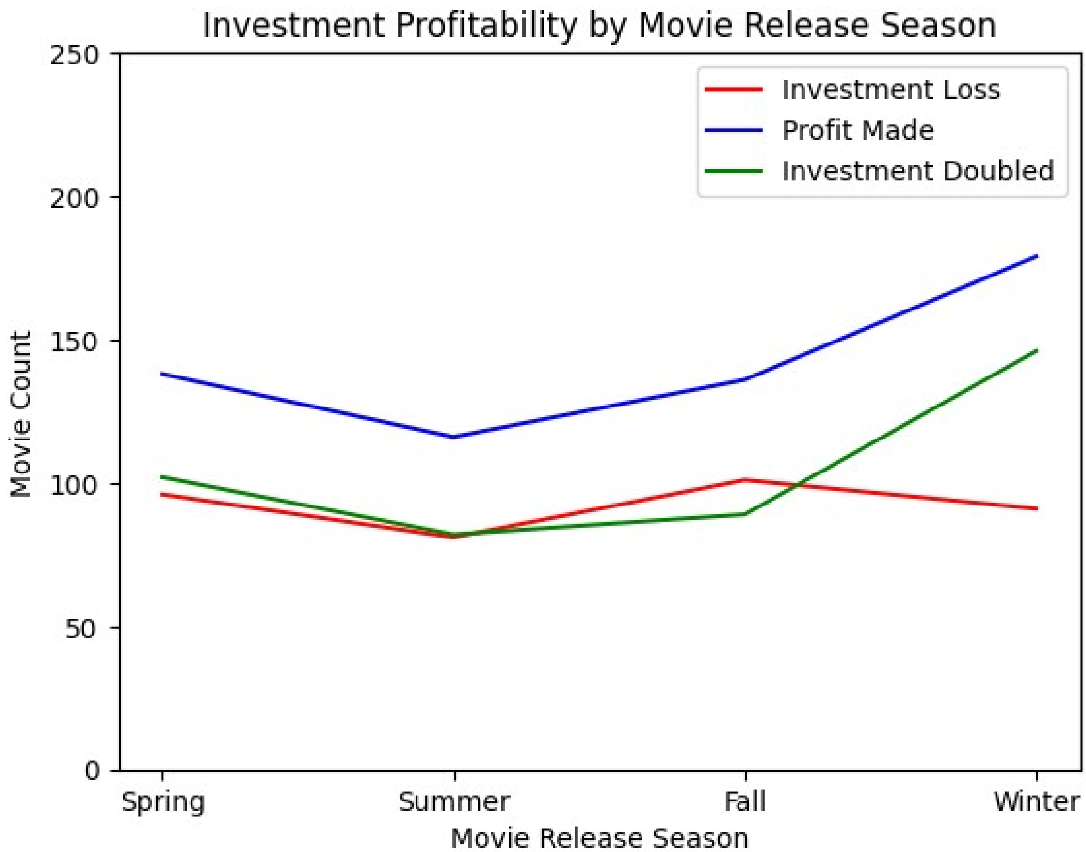
```
#creates a new dataframe from a subset of movies dataframe
Season_analysis= movies_df[['movie','Season', 'profitability']]

#drops all duplicate rows in this dataframe
Season_analysis= Season_analysis.drop_duplicates(subset=['movie'],
keep='first')

#gets rid of the now unnecessary movie column
Season_analysis= Season_analysis.drop('movie', axis= 1)

#create a new dataframe grouped by season of release that counts each
instance of profitability
seasonal_profits= Season_analysis.groupby('Season')
['profitability'].value_counts().\
                    unstack(fill_value=0)
seasonal_profits= seasonal_profits.reset_index()

#create a new dataframe reordering the dataframe columns into a
sensible order
seasonal_profits2 = seasonal_profits[[seasonal_profits.columns[0],
                                      seasonal_profits.columns[2],
                                      seasonal_profits.columns[3],
                                      seasonal_profits.columns[1]]]

#reorders the dataframe rows into a sensible order
seasonal_profits2 = pd.concat([seasonal_profits2.iloc[1:2],
                               seasonal_profits2.iloc[2:3],
                               seasonal_profits2.iloc[:1],
                               seasonal_profits2.iloc[3:]],
                              ignore_index=True)

#updates the dataframe so that the "Profit Made" column now also
includes all the instances
# of movies that at least doubled their investment amount. This is a
better representation
#of how many movies actually made profit over the previous version.
seasonal_profits2['Profit Made']= seasonal_profits2['Profit Made']+ \
                                    seasonal_profits2['Investment
Doubled']
```

```
#view the updated dataset we will use for analysis
seasonal_profits2
```

profitability	Season	Investment Loss	Profit Made	Investment Doubled
0	Spring	96	138	102
1	Summer	81	116	82
2	Fall	101	136	89
3	Winter	91	179	146

Now that the dataframe has been shaped for the Seasional Movie Release Date analysis we graph it to analize the results.

```
#extract values from the seasonal_profits2 dataframe
loss = seasonal_profits2['Investment Loss']
profit = seasonal_profits2['Profit Made']
doubled = seasonal_profits2['Investment Doubled']
Season = seasonal_profits2['Season'].tolist()

#creates figure and axis for plotting
fig, ax = plt.subplots()

#plots a color coded line chart for each profit level across the
seasons
ax.plot(Season, loss, label= "Investment Loss", color= 'red')
ax.plot(Season, profit, label= "Profit Made", color= 'blue')
ax.plot(Season, doubled, label= "Investment Doubled", color= 'green')

ax.legend() #shows the legend

ax.set_ylim(0,250) #sets Y axis limit for viewing ease


ax.set_ylabel('Movie Count') #sets Y axis title
ax.set_xlabel('Movie Release Season') #sets X axis title
ax.set_title('Investment Profitability by Movie Release Season') #sets
graph title

#Save the plot as a PNG file
plt.savefig("../images/Investment_Return_by_Release_Season.png",
dpi=150)

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



Here we see that the amount of unsuccessful movies (or movies that resulted in an production budget investment loss) are somewhat similar throughout the entire year. However, only the Winter Season (from December - February) has both a significantly higher amount of movies that resulted in a domestic profit and a domestic profit that doubled the production budget that the amount of movies that experienced a loss.

Conclusions

This analysis resulted in 3 recommendations that our new movie studio director can use to help guide this new business venture.

Implementing these recommendations into the criteria for producing and launching a new movie in the modern domestic market could help minimize the risk of undesirable results such as an investment loss.

#1 This is a new movie production, studio so starting out with a lower-level movie production budget under $5,000,000 could be the safest and most sensible option. From 2015 - 2020, compared to higher budget levels, the movies released at this budget level experienced the lowest percentage of investment losses and the highest percentage of at least doubling their investment. Additionally, if the movie is not a success the amount of money lost would be relatively limited compared to what would be at stake if a higher budget was used.

#2 In the same spirit of minimizing the risk of a project failure and maximizing the chance of success, I would suggest making a Horror movie. From 2015 - 2020, movies in the Horror genre experienced both the lowest amount of investment losses and the highest amounts of investments doubling in returns when compared to the overall amount of movies released for a genre. Horror movies also can pair nicely with lower production budget levels.

#3 Based on this analysis I would suggest launching the movie to the public during one of the winter months (December, January, February). In the graph, we can see that from 2015 - 2020, even though the number of movies launched resulting in an investment loss was relatively even throughout the year, there was a significant spike in movies where profits were made and especially a significant upward deviation from the general level of investments doubled throughout the year during these winter months.

Following these suggestions could allow us to break into the market with a minimized risk, more easily controlled losses, and entry at a lower price point. Additionally, once we have a solid footing in the market and have gained experience and recognition we could expand out to global markets, more lavish productions, and other genres.

Next steps

Additional analysis could help further minimize investment risk, and highlight what is currently popular and profitable in the movie market. Three future analyses could be:

#1 Actor Analysis: Given more data about different actors, this analysis could help narrow down who the current popular actors are, or the actors that cause the biggest spike in ticket sales when put in a movie.

#2 Marketing Technique Analysis: Given more data about different marketing techniques, this could shed light on what the most effective current strategies are, and what it would take to roll them out.

#3 Global Market Analysis: Given more data about the global market this could help predict what types of movies and attributes of movies would be successful in other countries around the world, increasing our market share.