How to Make the Most Money$ with one Movie

We are representatives for a small indie film company. Today we will share with you the insights that we have gained in our search for determining the greatest return for your investment.

Chuck Bui
Christopher Turner
Jack Jeffries
Sarah Brittle

Project 1 – Mo Money$ Movie$

Scope and Criteria

Our assumption was to find a genre with the highest return on investment, the optimal budget range with the best rate of return and determine how of a high rating to aim for to entice movie goers.

Answer the questions:

Does budget affect overall profit?

Do ratings need to be high to be profitable?

Which genres stand out as big money makers?

We looked at only theater released movies within the year range of 2005-2015 with a 'measurable' budget and revenue: minimum $5000 for each

Key Metrics: Movie budgets, profits, ratings and genre

Resources: Kaggle data set with 45,000 movies, OMDB API

QUICK TAKE OF THE DATA ACQUISITION

- Data set, "movies_metadata.csv" from Kaggle with over 3500 upvotes which has a Usability rating of 8.24 and 45,000 movies. This was needed for its budget and revenue data.
- Based on our criteria, we filtered that down to 2,316 movies for our dataset.
- Created an API call for OMDB to cross reference with the cleaned data set by utilizing IMDB ID's, which is a universal standard for movie data. This was needed for ratings and number of votes data.

IMPORTING THE DATA AND BEGIN CLEANING

Project1_MoMovies - Mo Money Movies

```
# dependencies
import pandas as pd
from pathlib import Path
```

```
#import csv
tmdb_data_csv = Path("Resources/movies_metadata.csv")
tmdb_data = pd.read_csv(tmdb_data_csv,low_memory=False)
```

```
# check columns
print(tmdb_data.columns)
```

```
Index(['adult', 'belongs_to_collection', 'budget', 'genres', 'homepage', 'id',
       'imdb_id', 'original_language', 'original_title', 'overview',
       'popularity', 'poster_path', 'production_companies',
       'production_countries', 'release_date', 'revenue', 'runtime',
       'spoken_languages', 'status', 'tagline', 'title', 'video',
       'vote_average', 'vote_count'],
      dtype='object')
```

```
#drop unneeded columns
tmdb_data= tmdb_data[["imdb_id", "original_title", "budget", "revenue", "release_date"]]
print(tmdb_data)
```

```
# create year from release date
tmdb_data["Year"] = tmdb_data["release_date"].str[-4:]
print(tmdb_data)
```

```
         imdb_id                original_title     budget      revenue  \
0      tt0114709                     Toy Story   30000000  373554033.0
1      tt0113497                       Jumanji   65000000  262797249.0
2      tt0113228              Grumpier Old Men          0          0.0
3      tt0114885             Waiting to Exhale   16000000   81452156.0
4      tt0113041   Father of the Bride Part II          0   76578911.0
...          ...                           ...        ...          ...
45461  tt6209470                          0.0          0      رگ خواب
45462  tt2028550           Siglo ng Pagluluwal          0          0.0
45463  tt0303758                      Betrayal          0          0.0
45464  tt0008536           Satana likuyushchiy          0          0.0
45465  tt6980792                      Queerama          0          0.0

      release_date  Year
0       10/30/1995  1995
1       12/15/1995  1995
2       12/22/1995  1995
3       12/22/1995  1995
4        2/10/1995  1995
...            ...   ...
45461          NaN   NaN
45462   11/17/2011  2011
45463     8/1/2003  2003
45464   10/21/1917  1917
45465     6/9/2017  2017

[45466 rows x 6 columns]
```

```
# delete unneeded col
tmdb_data = tmdb_data.drop(columns=["release_date"])
```

```
#rename columns
tmdb_data = tmdb_data.rename(columns={"imdb_id": "IMDB ID", "original_title": "Title", "budget": "Budget", "revenue": "Revenue"})
tmdb_data.head()
```

	IMDB ID	Title	Budget	Revenue	Year
0	tt0114709	Toy Story	30000000	373554033.0	1995
1	tt0113497	Jumanji	65000000	262797249.0	1995
2	tt0113228	Grumpier Old Men	0	0.0	1995
3	tt0114885	Waiting to Exhale	16000000	81452156.0	1995
4	tt0113041	Father of the Bride Part II	0	76578911.0	1995

CLEANING BUDGET AND YEAR | CREATING THE FILTER FOR CRITERIA

```
tmdb_data.dtypes
```

```
IMDB ID     object
Title       object
Budget      object
Revenue    float64
Year        object
dtype: object
```

```
# convert to float and force non numerics to nan. errors='coerce' ensures that values that can't be converted to a
# number is replaced with NaN. check link below for to_numeric/coerce explanations
# https://stackoverflow.com/questions/33961028/remove-non-numeric-rows-in-one-column-with-pandas
tmdb_data['Year'] = pd.to_numeric(tmdb_data['Year'], errors='coerce')

#drop nans and convert to int from float
tmdb_data = tmdb_data.dropna(subset=['Year'])
tmdb_data['Year'] = tmdb_data['Year'].astype(int)
tmdb_data
```

```
# filter for movies with measureable budget and revenue within our range
filtered_movies = tmdb_data[(tmdb_data['Budget'] >= 5000) & (tmdb_data['Revenue'] >= 5000) & (tmdb_data['Year'] >= 2005) & (tmdb_data['Year'] <= 2015)]
filtered_movies
```

	IMDB ID	Title	Budget	Revenue	Year
4356	tt2018086	Camille Claudel 1915	3512454	115860.0	2013
9352	tt0318081	A Sound of Thunder	80000000	5989640.0	2005
9441	tt0366627	The Jacket	29000000	21126225.0	2005
9460	tt0373926	The Interpreter	80000000	162944923.0	2005
9475	tt0377109	The Ring Two	50000000	161451538.0	2005
...	...	...	...	...	...
44970	tt0453365	FC Venus	2196531	2411594.0	2005
45250	tt0479751	சிவாஜி	12000000	19000000.0	2007
45409	tt0933361	Dikari	800000	1328612.0	2006
45412	tt1718881	Про любоff	2000000	1268793.0	2010
45422	tt1110037	Антидурь	5000000	1413000.0	2007

2329 rows × 5 columns

API BY IMDB ID FROM FILTERED DATA SET | MERGING AND RENAMING

[32]:
```
# OMDB url and api prep for imdb id
omdb_url = "http://www.omdbapi.com/?apikey=" + omdb_key + "&i="
```

[34]:
```
# declare imdb id's for searching api - if we divide the request. or I might pay the $1 to simplify
imdb_ids = data_df["IMDB ID"]
```

[36]:
```
# Init empty movie data list
movies_data = []

for id in imdb_ids:
    omdb_response = requests.get(omdb_url + id)
    omdb_data = omdb_response.json()

    if "imdbID" in omdb_data:
            movie_info = {
                "Title": omdb_data.get("Title"),
                "Year": omdb_data.get("Year"),
                "IMDB Rating": omdb_data.get("imdbRating"),
                "IMDB Votes": omdb_data.get("imdbVotes"),
                "Genre": omdb_data.get("Genre"),
                "Box Office": omdb_data.get("BoxOffice"),
                "IMDB ID": omdb_data.get("imdbID")
            }
            movies_data.append(movie_info)
```

[ ]:
```
# Merge datasets
mo_movies_df = pd.merge(movies_df, data_df, on="IMDB ID", how="inner")
mo_movies_df
```

[ ]:

	Title_x	Year_x	IMDB Rating	IMDB Votes	Genre	Box Office	IMDB ID	Title_y	Budget	Revenue	Year_y
0	Camille Claudel 1915	2013	6.5	3,889	Biography, Drama	$35,296	tt2018086	Camille Claudel 1915	3512454	115860.0	2013
1	Camille Claudel 1915	2013	6.5	3,889	Biography, Drama	$35,296	tt2018086	Camille Claudel 1915	3512454	115860.0	2013
2	Camille Claudel 1915	2013	6.5	3,889	Biography, Drama	$35,296	tt2018086	Camille Claudel 1915	3512454	115860.0	2013
3	Camille Claudel 1915	2013	6.5	3,889	Biography, Drama	$35,296	tt2018086	Camille Claudel 1915	3512454	115860.0	2013
4	A Sound of Thunder	2005	4.2	20,549	Action, Adventure, Horror	$1,900,451	tt0318081	A Sound of Thunder	80000000	5989640.0	2005
...	...	...	...	...	...	...	...	...	...	...	...
2327	FC Venus	2005	5.5	2,325	Comedy, Romance, Sport	N/A	tt0453365	FC Venus	2196531	2411594.0	2005
2328	Sivaji	2007	7.5	21,484	Action, Crime, Drama	N/A	tt0479751	சிவாஜி	12000000	19000000.0	2007
2329	Dikari	2006	6.5	693	Comedy	N/A	tt0933361	Dikari	800000	1328612.0	2006
2330	Pro lyuboff	2010	5.8	297	Drama	N/A	tt1718881	Про любоff	2000000	1268793.0	2010
2331	Antidur	2007	3.4	232	Action, Comedy, Crime	N/A	tt1110037	Антидурь	5000000	1413000.0	2007

2332 rows × 11 columns

[ ]:
```
# clean up title
mo_movies_df = mo_movies_df.rename(columns={"Title_x": "Title"})
mo_movies_df = mo_movies_df.drop(columns=["Title_y"])
```

DROPPING A FEW DUPED MOVIES AND CREATING OUR DATASET

```
# clean up year
mo_movies_df = mo_movies_df.rename(columns={"Year_x": "Year"})
mo_movies_df = mo_movies_df.drop(columns=["Year_y"])
```

```
# are there duplicates?
duplicates = mo_movies_df.loc[mo_movies_df.duplicated(subset=["Title"], keep=False),"Title"].unique()
```

```
duplicates
```

```
array(['Camille Claudel 1915', 'Fantastic Four', 'Day of the Falcon',
       'The Illusionist', 'Unknown', 'The Host', 'The Signal', 'The Wave',
       'The Girl with the Dragon Tattoo', 'Stolen', 'The Other Woman',
       'Captive'], dtype=object)
```

```
# drop those dupes
cleaned_df = mo_movies_df.drop_duplicates(subset="Title", keep='first', inplace=False, ignore_index=False)
```

```
cleaned_df
```

	Title	Year	IMDB Rating	IMDB Votes	Genre	Box Office	IMDB ID	Budget	Revenue
0	Camille Claudel 1915	2013	6.5	3,889	Biography, Drama	$35,296	tt2018086	3512454	115860.0
4	A Sound of Thunder	2005	4.2	20,549	Action, Adventure, Horror	$1,900,451	tt0318081	80000000	5989640.0
5	The Jacket	2005	7.1	119,641	Drama, Fantasy, Mystery	$6,303,762	tt0366627	29000000	21126225.0
6	The Interpreter	2005	6.4	111,280	Crime, Mystery, Thriller	$72,708,161	tt0373926	80000000	162944923.0
7	The Ring Two	2005	5.4	101,457	Horror, Mystery	$76,231,249	tt0377109	50000000	161451538.0
...	...	...	...	...	...	...	...	...	...
2327	FC Venus	2005	5.5	2,325	Comedy, Romance, Sport	N/A	tt0453365	2196531	2411594.0
2328	Sivaji	2007	7.5	21,484	Action, Crime, Drama	N/A	tt0479751	12000000	19000000.0
2329	Dikari	2006	6.5	693	Comedy	N/A	tt0933361	800000	1328612.0
2330	Pro lyuboff	2010	5.8	297	Drama	N/A	tt1718881	2000000	1268793.0
2331	Antidur	2007	3.4	232	Action, Comedy, Crime	N/A	tt1110037	5000000	1413000.0

2316 rows × 9 columns

```
cleaned_df.to_parquet("Resources/mo_movies_data.parquet", index=False)
```

BUDGET VS REVENUE

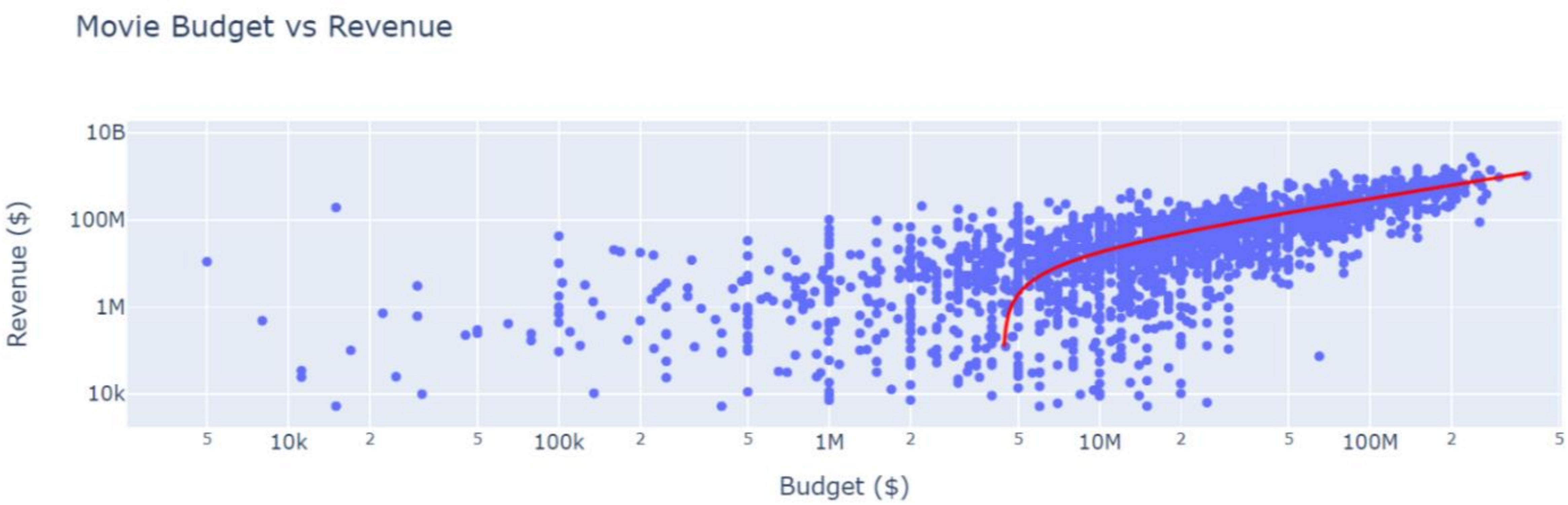
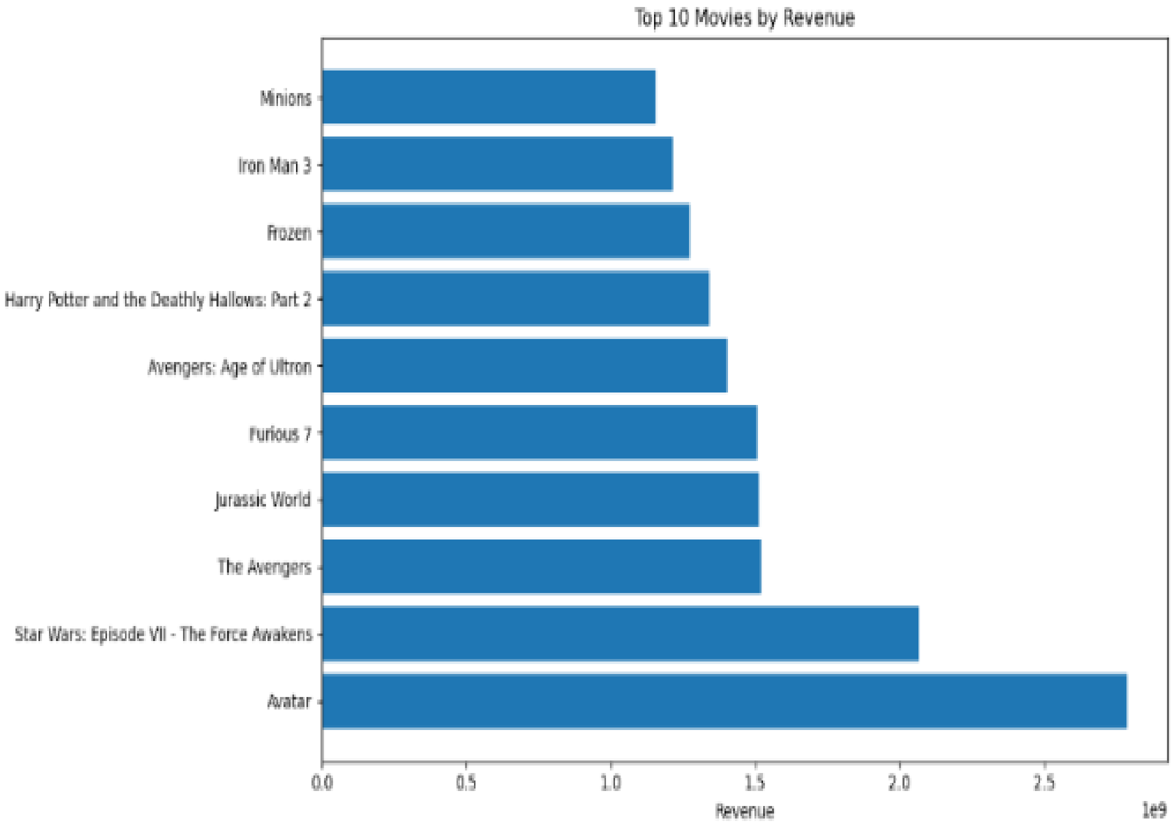


Figure 1 Scatter plot showing the relationship between money spent and money earned

- The graph shows a relative correspondence between amount spent and amount earned
- There is a moderate r value of .6 between these two

BUDGET VS REVENUE CONT'D



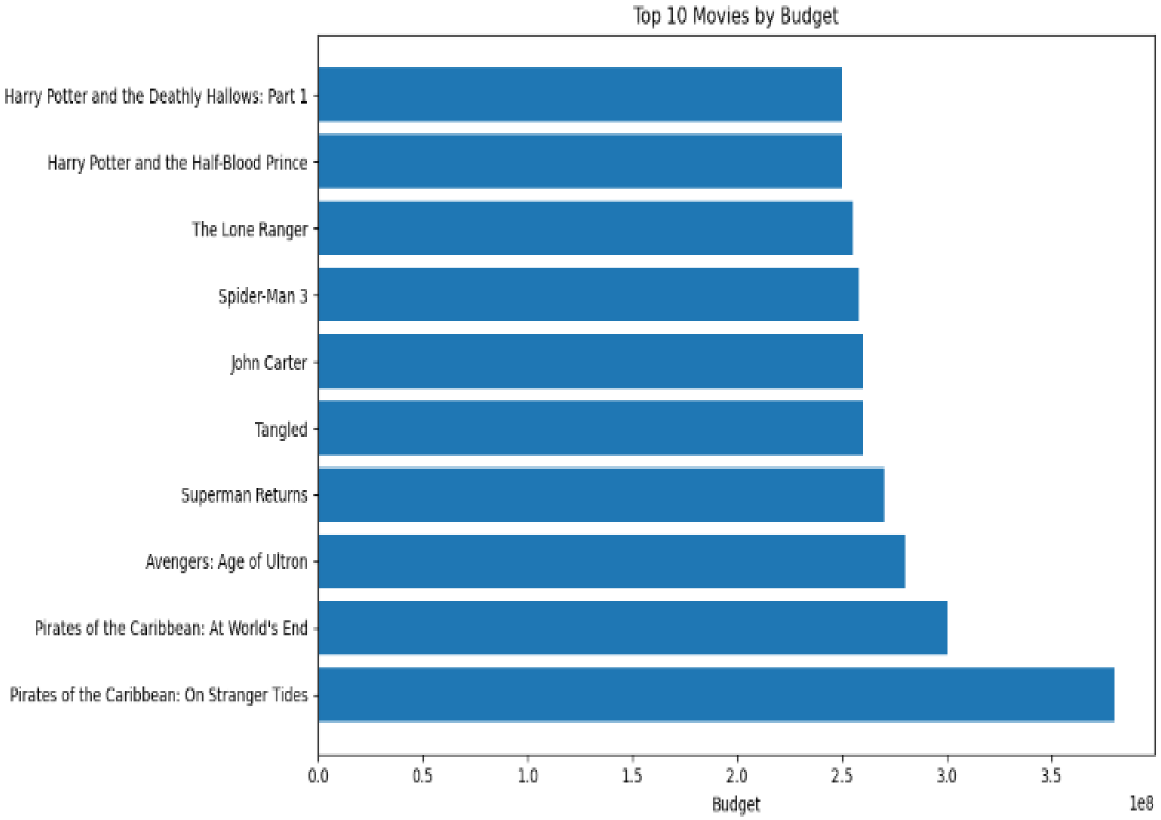


Figure 2 Top ten movies by budget and by revenue. Only one movie exists in both of these graphs.

- While there is a moderate correlation, there are some outliers. For example, Avengers: Age of Ultron is the only movie from top 10 by budget that is also in top 10 by revenue.
- A high budget does not always equate to ROI, as movies that make more are also the movies that spend more.

RETURN ON INVESTMENT

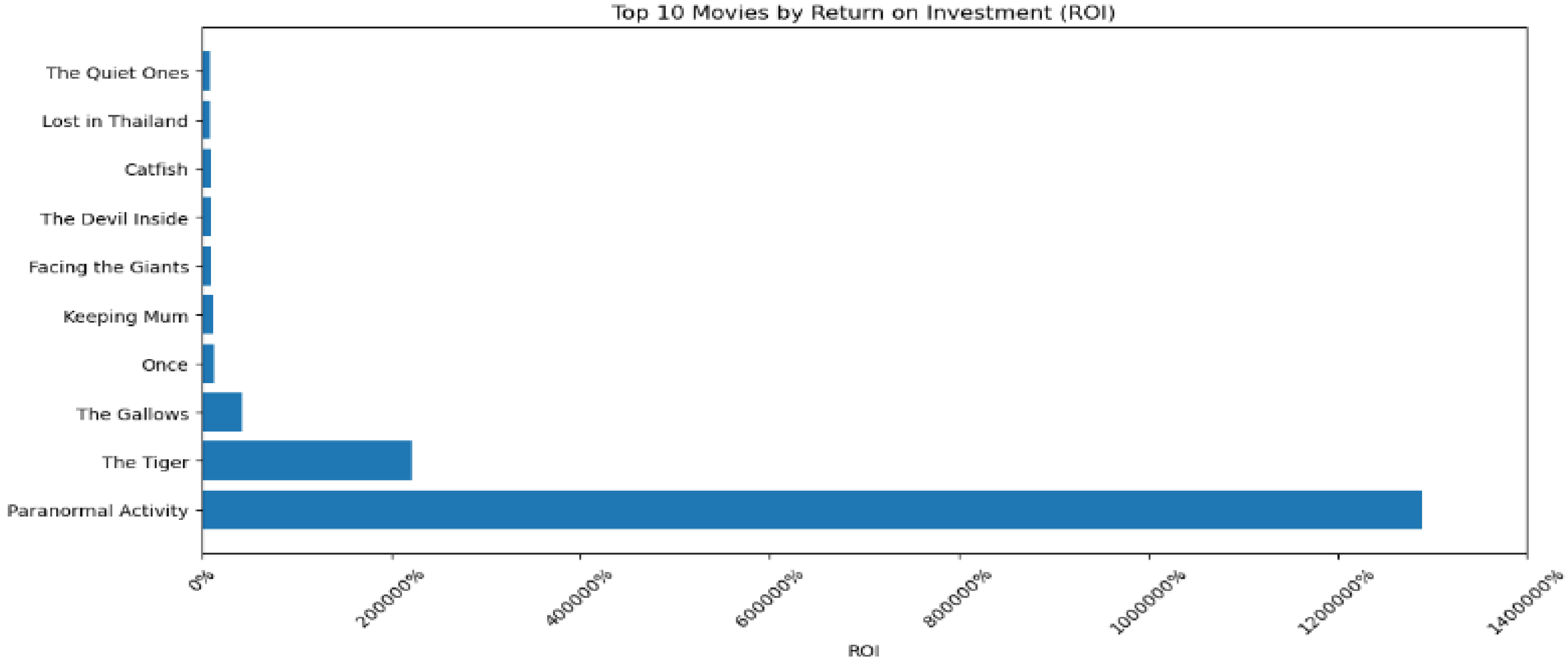
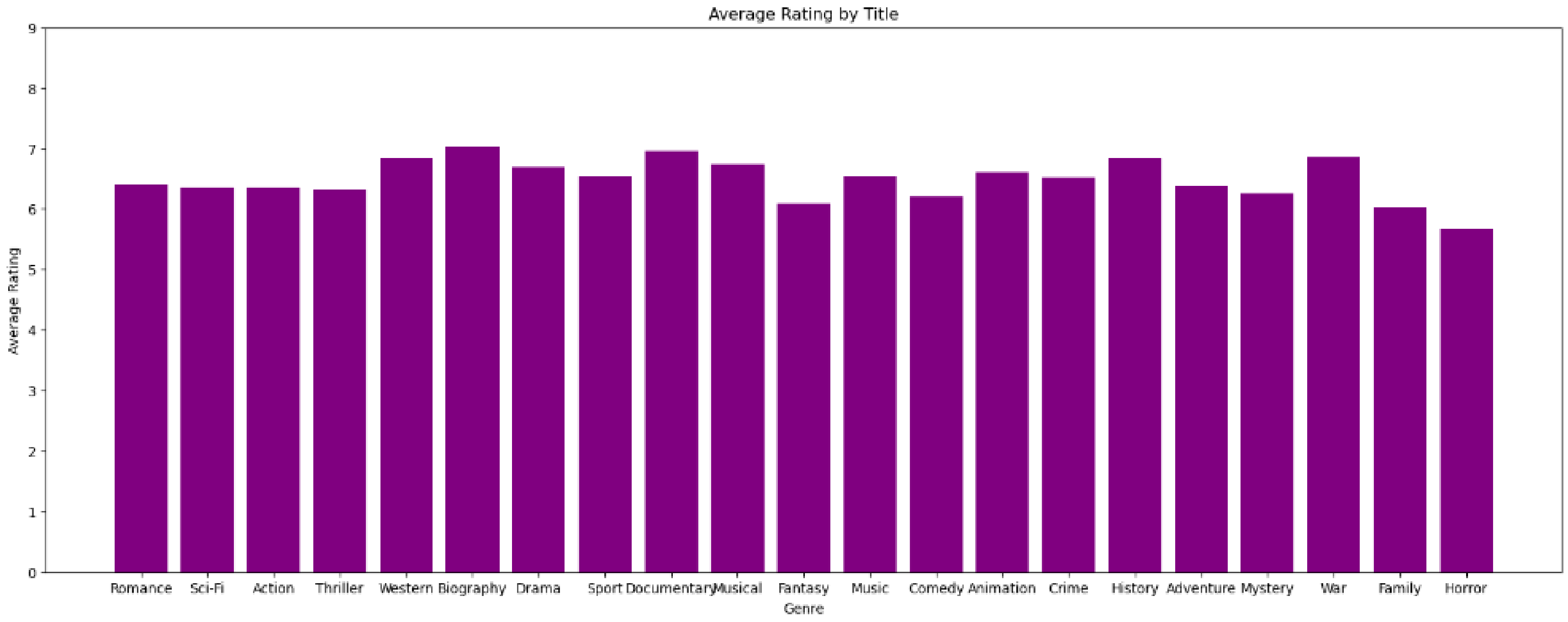


Figure 3 The highest return on investment. "Paranormal Activity" stands out as a clear outlier.

- While there is a moderate correlation, there are some outliers. For example, Avengers: Age of Ultron is the only movie from top 10 by budget that is also in top 10 by revenue.

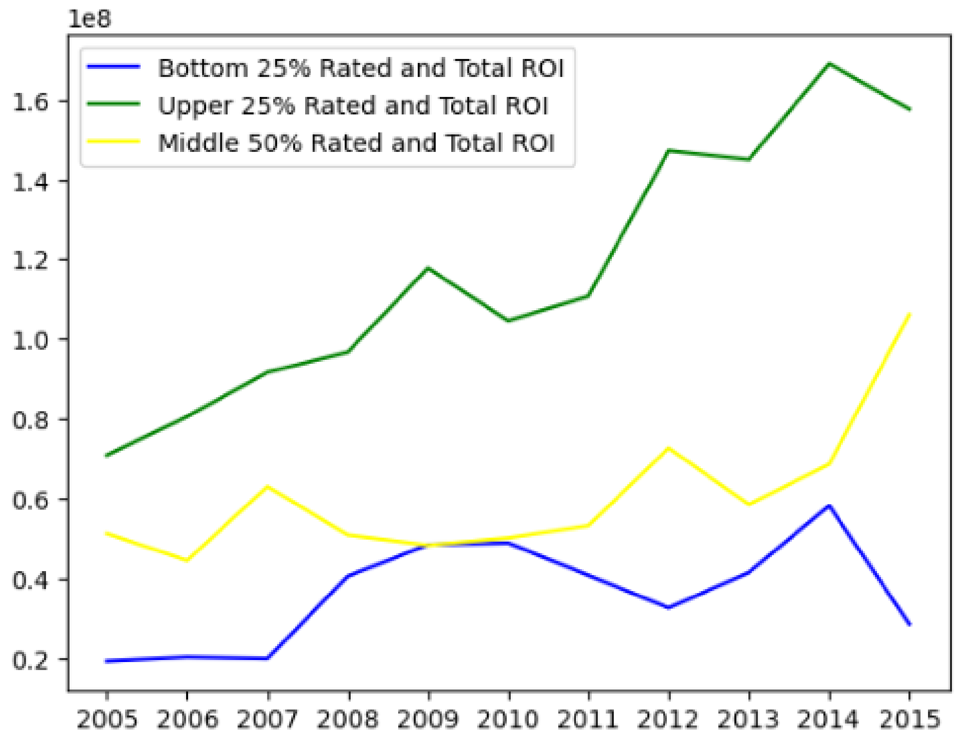
DO RATINGS MATTER FOR PROFIT?

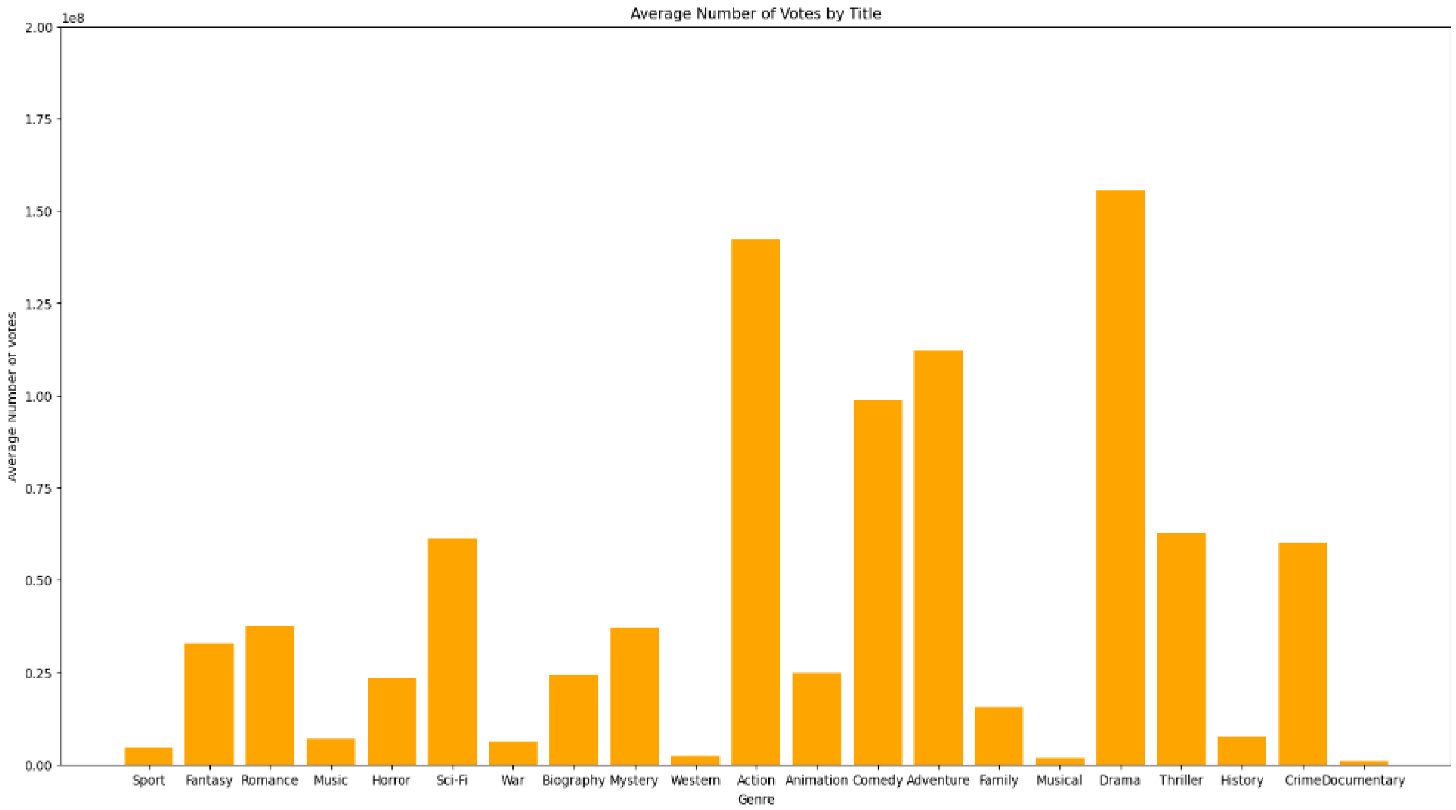
- Overall, ratings remain the same across all years and genre.
- The decade average variable is just .2



	IMDB Rating				
	mean	median	var	std	sem
Year					
2005	6.354450	6.50	1.025020	1.012432	0.073257
2006	6.386321	6.50	1.058817	1.028988	0.070671
2007	6.506771	6.60	1.033933	1.016825	0.073383
2008	6.301932	6.40	1.204948	1.097701	0.076296
2009	6.417972	6.50	0.891296	0.944085	0.064089
2010	6.368996	6.40	0.887061	0.941839	0.062238
2011	6.327039	6.40	0.999654	0.999827	0.065501
2012	6.402913	6.45	1.011894	1.005929	0.070086
2013	6.512500	6.50	0.876345	0.936133	0.062548
2014	6.375962	6.40	1.111690	1.054367	0.073107
2015	6.491878	6.60	0.842179	0.917703	0.065384

DO RATINGS MATTER FOR PROFIT? CONT'D

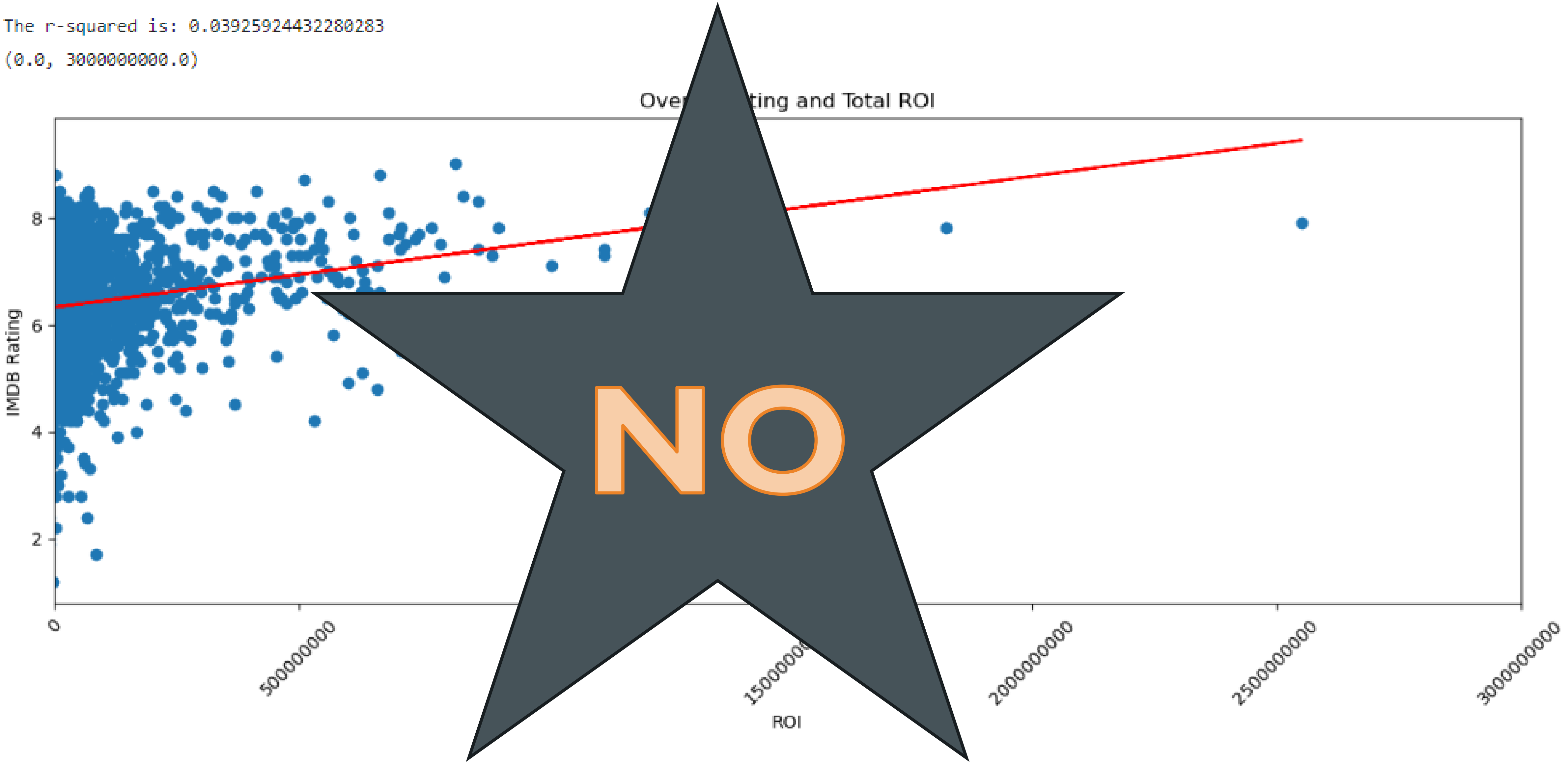




- Average number of votes show that engagement may be higher for certain genres, but the average ratings for these genres still remain comfortably within the range of all genres.

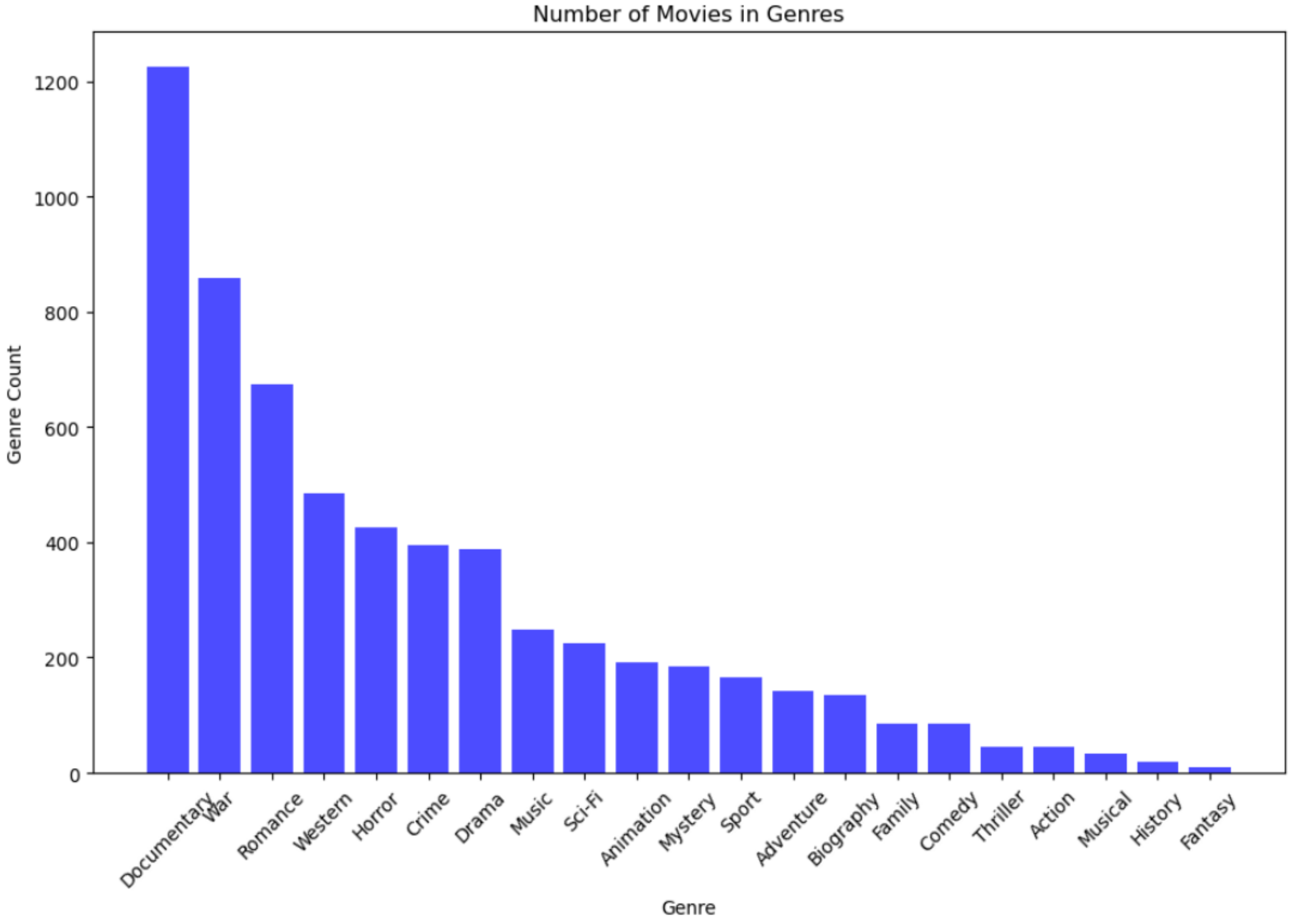
DO RATINGS MATTER FOR PROFIT? CONT'D

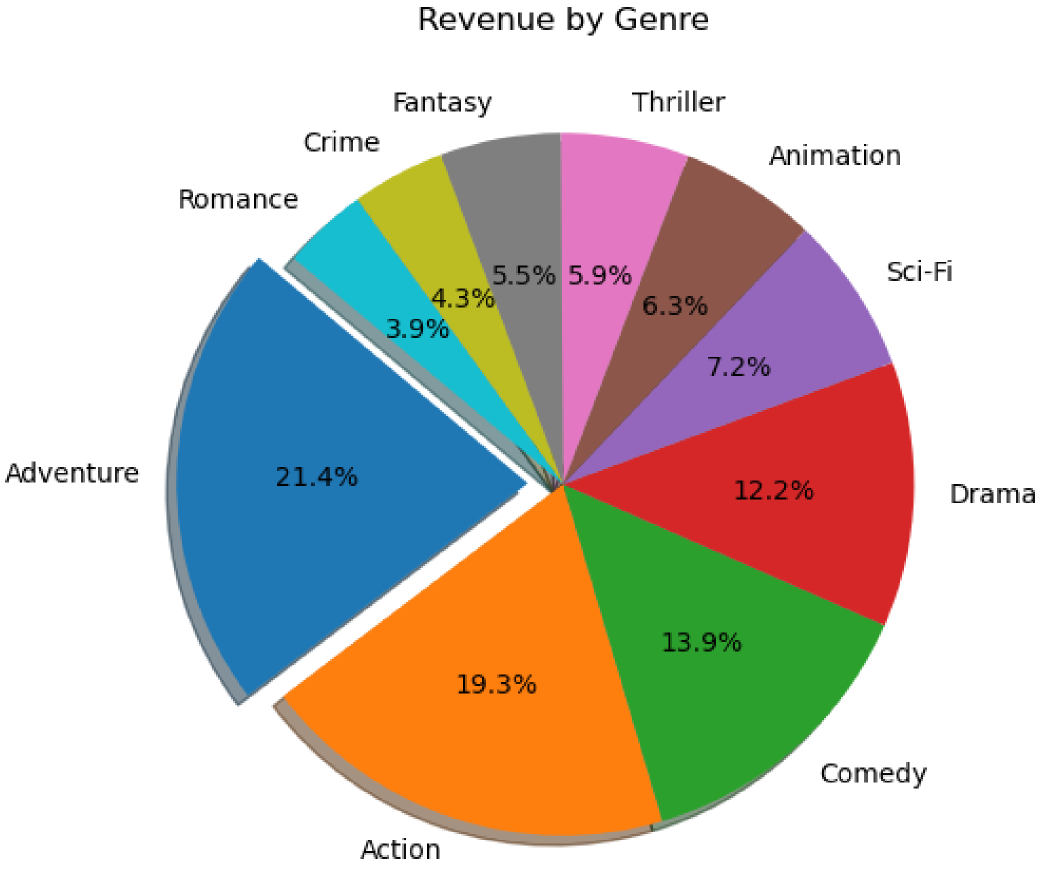
```
The r-squared is: 0.03925924432280283
(0.0, 3000000000.0)
```



GENRE VS RATING AND REVENUE

- Of the 2,316 movies, Drama is the highest reported
- Adventure only takes up 8% of the movies by volume.

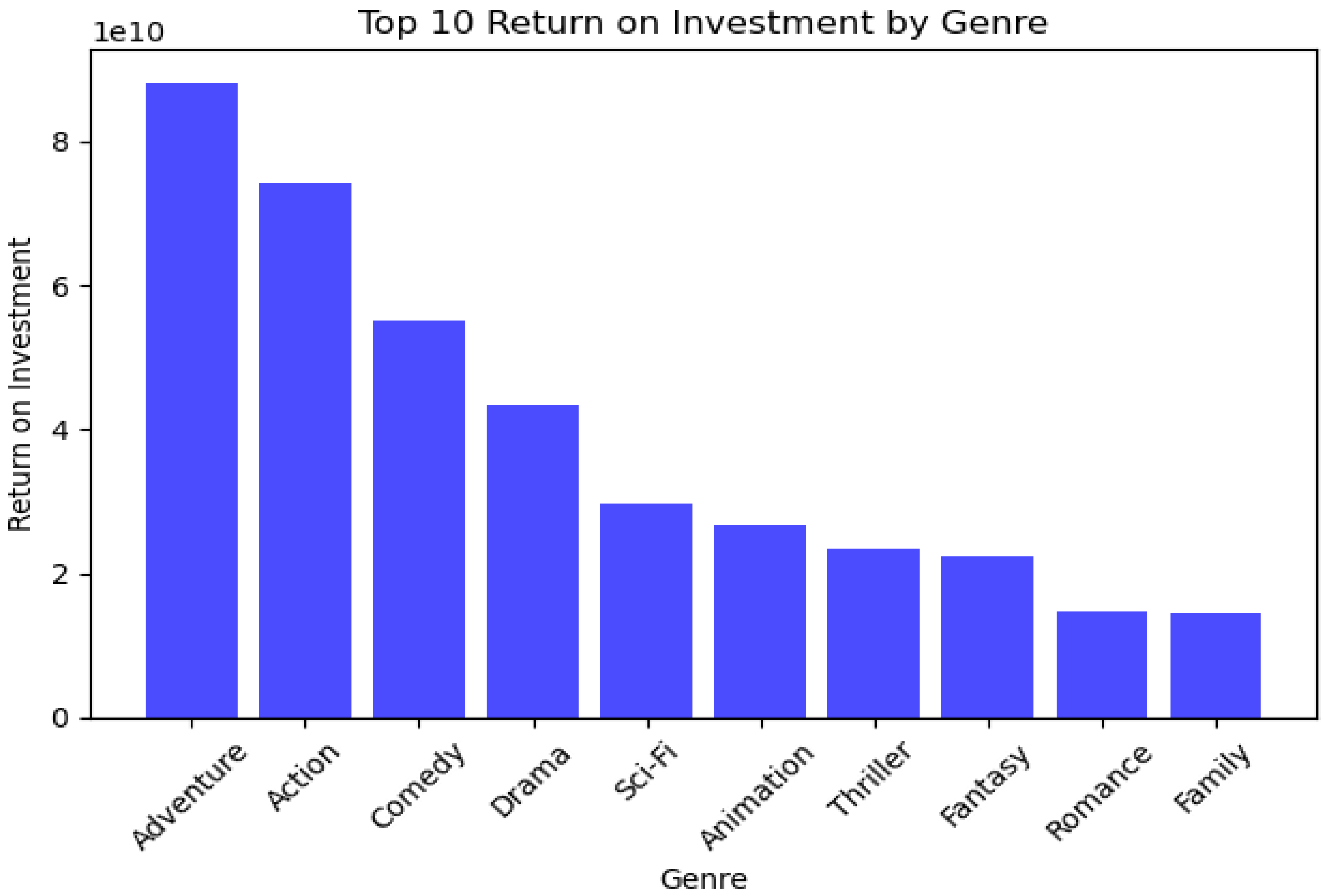




- Adventure has the highest revenue, followed closely by Action

GENRE VS RETURN ON INVESTMENT

- Shown previously, Adventure has a low market saturation.
- But you can clearly see that it's ROI is well above most other genres.
- This makes the adventure genre a great choice for a new movie.



Adventure and action movies deliver the highest return on investment.

CONCLUSION

- Being highly rated is not an important factor in making profit. An average rating of 6-7 out of 10 is sufficient.
- Our genre should be an action-adventure movie to utilize our top two highest earning genres.
- The top ten of genres suggests that a higher budget does not guarantee a higher ROI. We suggest a medium budget of $37,441,333 to maximize profits.
- We believe our data justifies our conclusion to aim for creating an average rated action-adventure movie, with a medium budget to maximize our total profit.