Movies Final Report

1. Overview

This Project is divided into two parts:

1. The Story of Film: This section aims at narrating the history, trivia and facts behind the world of cinema through the lens of data. Extensive Exploratory Data Analysis is performed on Movie Metadata about Movie Revenues, Casts, Crews, Budgets, etc. through the years. Two predictive models are built to predict movie revenues and movie success. Through these models, we also aim at discovering what features have the most significant impact in determining revenue and success.

2. Movie Recommender Systems: This part is focused around building various kinds of recommendation engines; namely the Simple Generic Recommender, the Content Based Filter and the User Based Collaborative Filter. The performance of the systems are evaluated in both a qualitative and quantitative manner.

2. Data Wrangling

2.1 Data Collection

The MovieLens Full Dataset was readily available at the GroupLens Website (https://grouplens.org/datasets/movielens/). This dataset contained 26 million ratings from 270,000 users on 60,000 movies. One file in this dataset, links.csv, contained the TMDB and IMDB IDs for all the movies.

I signed up for an API Key with TMDB. This gave me access to data at 3 endpoints. Each endpoint gave me details about the movie, its cast and crew information and plot keywords. I wrote 3 separate scrapers to hit each endpoint and collect this data for all 60,000 movies. Since TMDB has a restriction of 40 requests every 10 seconds, this task took a day to execute.

All the data collected was in the form of stringified JSON which demanded more processing. The data obtained from scraping was in the form of stringified JSON. This had to be converted into CSV Files to enable easier parsing

2.2 Getting to Know Data

The data used in this project has been obtained from two sources: The Movie Database (TMDB) and MovieLens.

MovieLens has a publicly available dataset that contains 26 million ratings and 750,000 tag applications applied to 45,000 movies by 270,000 users. It also includes tag genome data with 12 million relevance scores across 1,100 tags. A small subset of this dataset, containing 10,000 ratings for 9000 movies from 700 users is also available.

One of the files contains the TMDB ID of every movie listed in the MovieLens dataset. Using this ID, the metadata, credits and keywords of all 60,000 movies were obtained by running a script that requested and parsed data from TMDB Open API. The data collected was initially in the JSON format but was converted into CSV files using Python's Pandas Library. The following files were used in the project:

1. movies_metadata.csv: The file containing metadata collected from TMDB for over 60,000 movies. Data includesbudget, revenue, date released, genres, etc.
2. credits.csv: Complete information on credits for a particular movie. Data includes Director, Producer, Actors, Characters, etc.
3. keywords.csv: Contains plot keywords associated with a movie.

4. links_small.csv: Contains the list of movies that are included in the small subset of the Full MovieLens Dataset.
5. Ratings_small.csv: The MovieLens Dataset containing 100,000 ratings on 9,000 movies from 700 users. The main dataset used for building the Collaborative Filter.

2.3 Cleaning

The dataset had a lot of features which had 0s for values it did not possess. These values were converted to NaN. Some features were still in the form of a Stringified JSON Object. They were converted into Python Dictionaries using Python's ast library. These were further reduced into lists since we did not have a need for ID, timestamp and other attributes. The dataframe was exploded wherever the analysis demanded it (for instance, genres and production countries).

Finally, most of the features were converted into a Python basic type (integer, string, float) by removing all the unclean values. The date string was converted into a Pandas Datetime and from it, we extracted the month, year and day of release of every movie.

3. Exploratory Data Analysis and Visualization

In this section, the various insights produced through descriptive statistics and data visualization is presented.

This forms the crux of the first section of my Capstone Project.

3.1 Production Countries

1. The Movies in the dataset are overwhelmingly in the English Language and shot in the United States of America.
2. Europe is also an extremely popular location with the UK, France, Germany and Italy in the top five.
3. Japan and India are the most popular Asian countries when it comes to movie production

3.2 Franchise Movies

The **Avenger Collection** Franchise is the most successful movie franchise raking in more than 7.769 billion dollars from 4 movies. The **Star Wars Movies** come in a close second with a 7.71 billion dollars from 8 movies. **Harry Potter collection** is third.

3.3 Production Companies

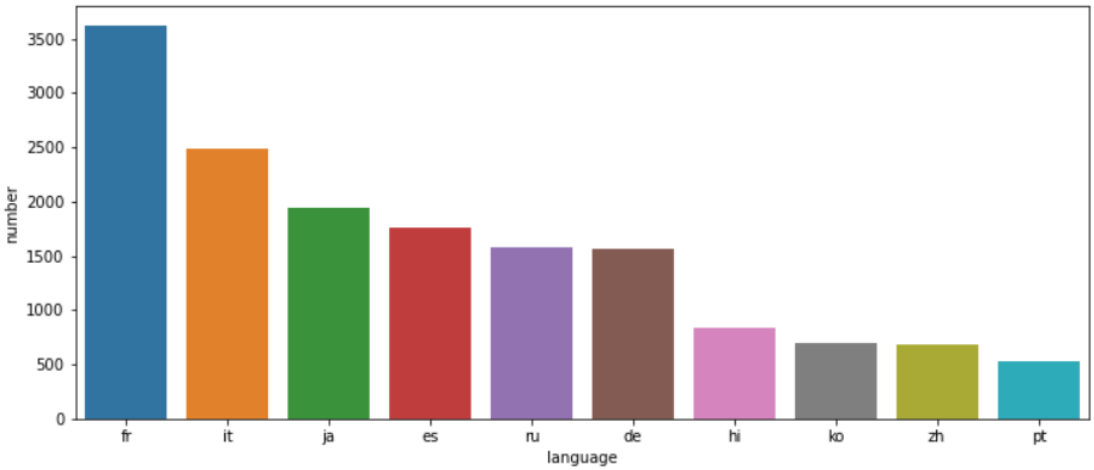
1. **Warner Bros** is the highest earning production company of all time earning a staggering 78.7 billion dollars from close to 600 movies. **Universal Pictures** and **20th Century Fox** are the second and the third highest earning companies with 68 billion dollars and 62 billion dollars in revenue respectively.
2. **Warner Bros. Pictures** produced the most number of movies. That is not surprising as it is also the highest earning production company. Fallowed by Universal Pictures and Paramount.

3.4 Movie Title word cloud

The word **Love** is the most commonly used word in movie titles. **Girl** and **Man** are also among the most commonly occurring words. I think this encapsulates the idea of the ubiquitous presence of romance in movies pretty well.

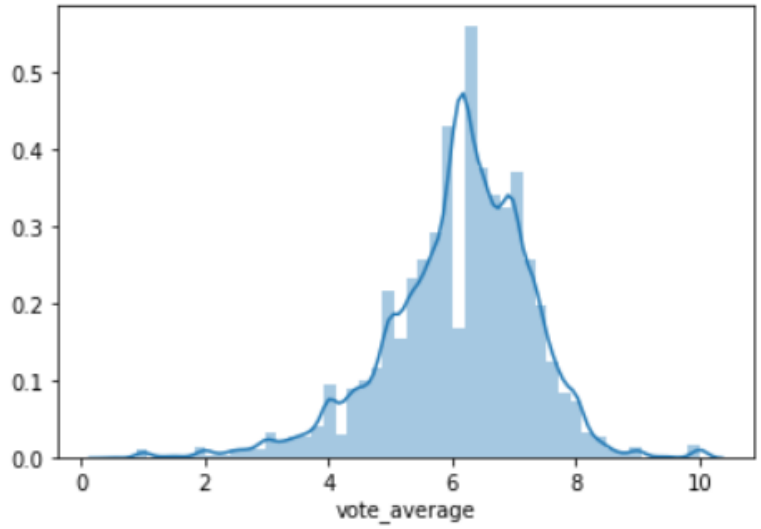
3.5 Original Languages

There are over 111 languages represented in our dataset. As we had expected, English language films form the overwhelmingly majority. French and Italian movies come at a very distant second and third respectively.

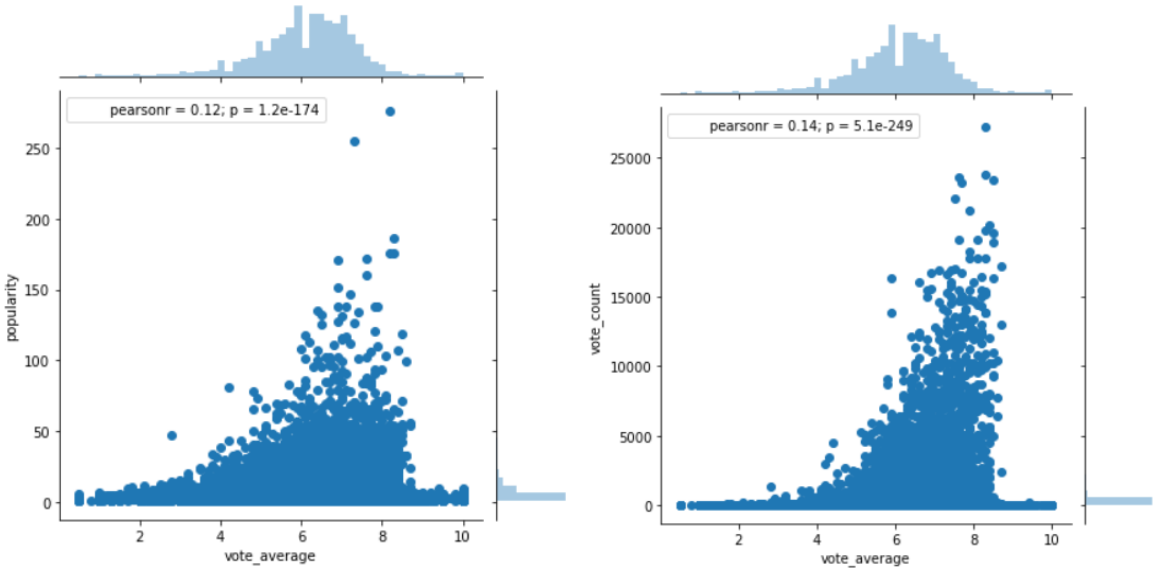


As mentioned earlier, French and Italian are the most commonly occurring languages after English. Japanese and Hindi form the majority as far as Asian Languages are concerned.

3.6 Popularity, Vote Average and Vote count

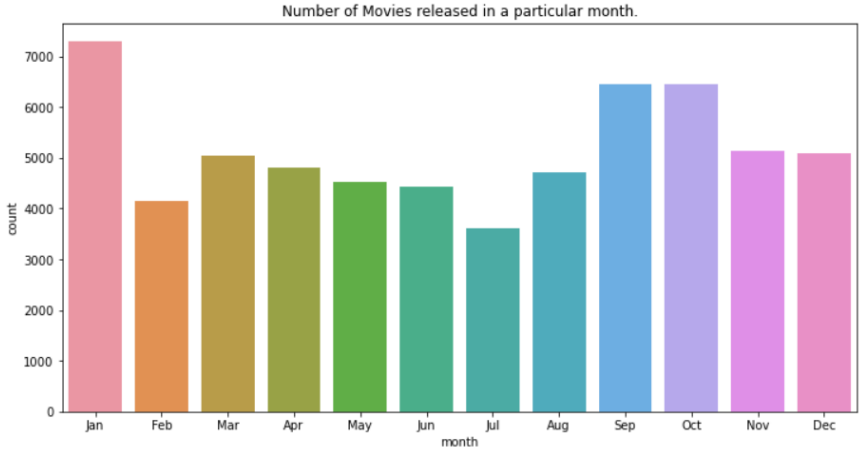


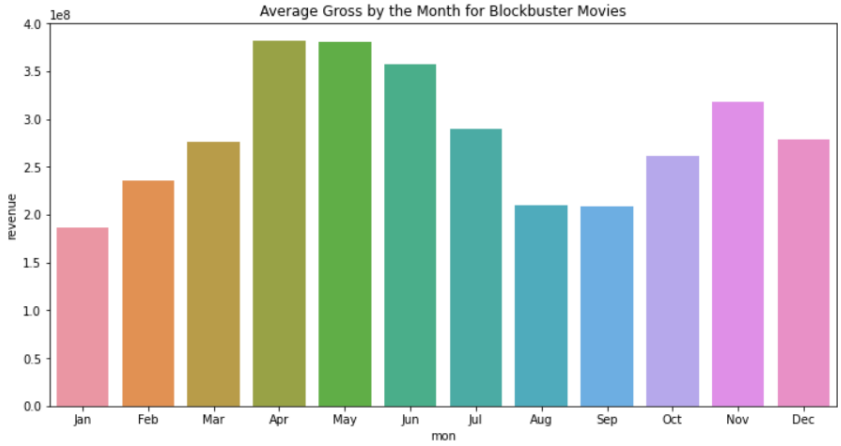
1. **Joker**. is the most popular movie by the TMDB Popularity Score. Frozen II and Avengers: Infinity War movies come in second and third respectively.
2. Inception and The Dark Knight, two critically acclaimed and commercially successful Christopher Nolan movies figure at the top of The Most Voted On Movies Chart.
3. The Shawshank Redemption and The Godfather are the two most critically acclaimed movies in the TMDB Database. Interestingly, they are the top 2 movies in IMDB's Top 250 Movies list too. They have a rating of over 9 on IMDB as compared to their 8.5 TMDB Scores.



4. Surprisingly, the Pearson Coefficient of the two aforementioned quantities is a measly 0.12 which suggests that there is no tangible correlation. In other words, popularity and vote average and independent quantities. Which means a large number of votes on a particular movie does not necessarily imply that movie is popular
5. There is a very small correlation between Vote Count and Vote Average. A large number of votes on a particular movie does not necessarily imply that the movie is good.

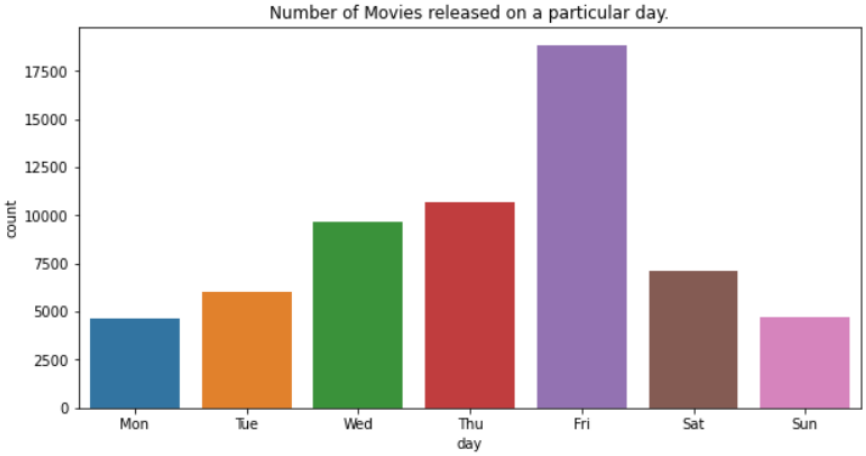
3.7 Movie Release Dates





It appears that January is the most popular month when it comes to movie releases. In Hollywood circles, this is also known as the dump month when subpar movies are released by the dozen.

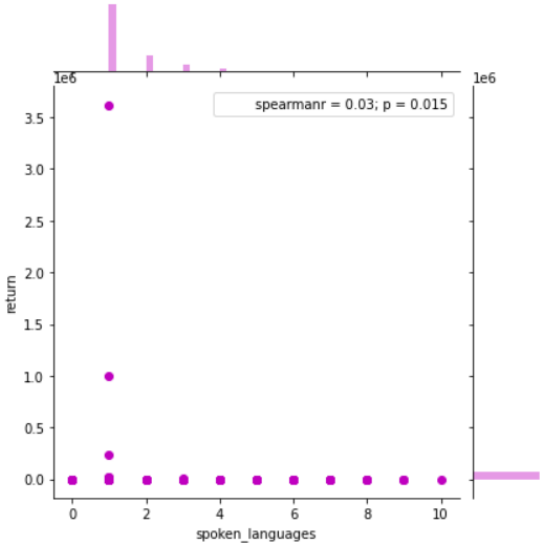
We see that the months of April, May and June have the highest average gross among high grossing movies. This can be attributed to the fact that blockbuster movies are usually released in the summer when the kids are out of school and the parents are on vacation and therefore, the audience is more likely to spend their disposable income on entertainment.



Friday is clearly the most popular day for movie releases. This is understandable considering the fact that it usually denotes the beginning of the weekend. Sunday and Monday are the least popular days and this can be attributed to the same aforementioned reason.

3.8 Spoken Languages

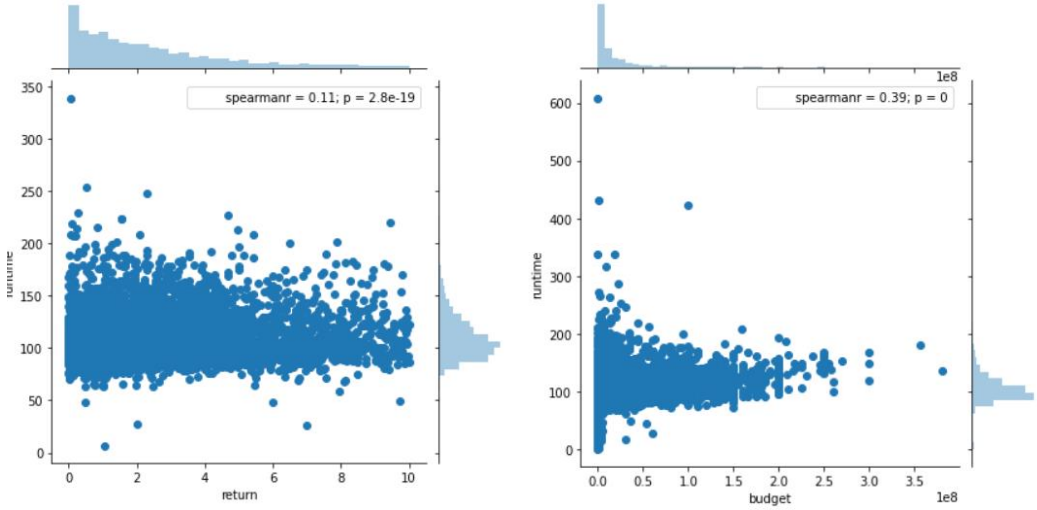
The movie with the most number of languages, Train Station is a movie about the character he/she misses a train. From there, the directors take Brown on a fantastical journey and throughout it illustrate the infinite possibilities that exist when a single event interrupts a person's timeline. Each decision s/he makes leads to a different scenario, each one told by a different director. This explains the sheer diversity of the movie in terms of language.



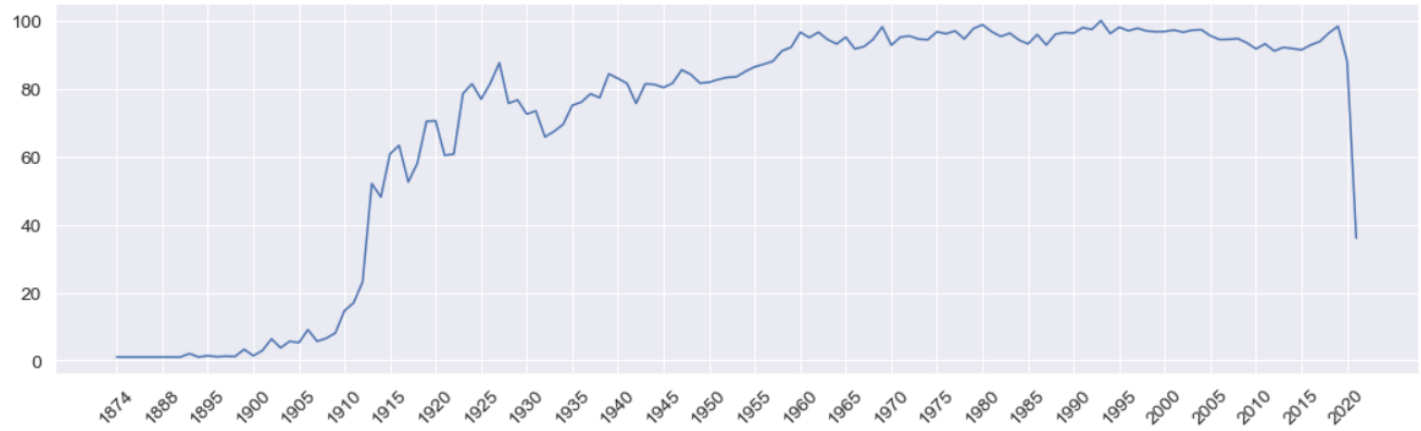
There is no correlation between the number of languages and returns of a movie.

3.9 Runtime

The average length of a movie is about 1 hour and 30 minutes. The longest movie on record in this dataset is a staggering 902 minutes (or 15 hours) long.

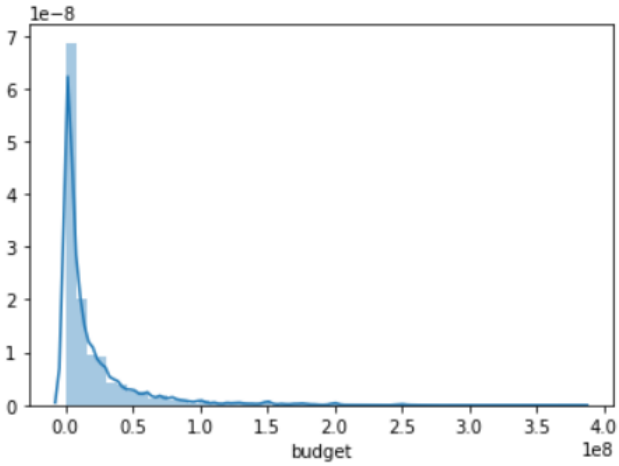


- There seems to be no relationship between runtime and return. The success of a movie is independent of its duration.
- The two quantities, runtime and budget have a much weaker correlation than I had expected. In retrospect, the genre of the movie tends to have a much greater impact on budget. A 3 hour art film will cost significantly lesser than a 90 minute Sci-Fi movie.



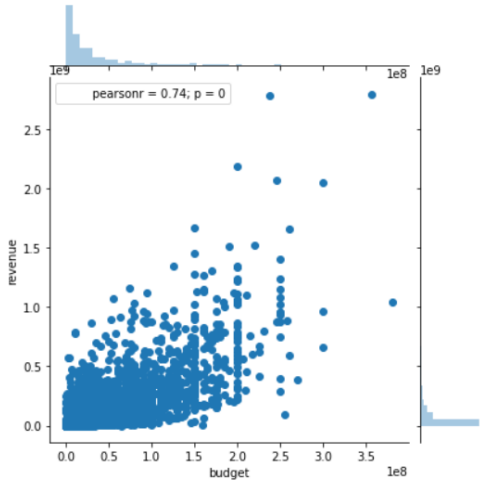
We notice that films started hitting the 60 minute mark as early as 1914. Starting 1924, films started having the traditional 90 minute duration and has remained more or less constant ever since.

3.10 Budget



The distribution of movie budgets shows an exponential decay. More than 75% of the movies have a budget smaller than 25 million dollars.

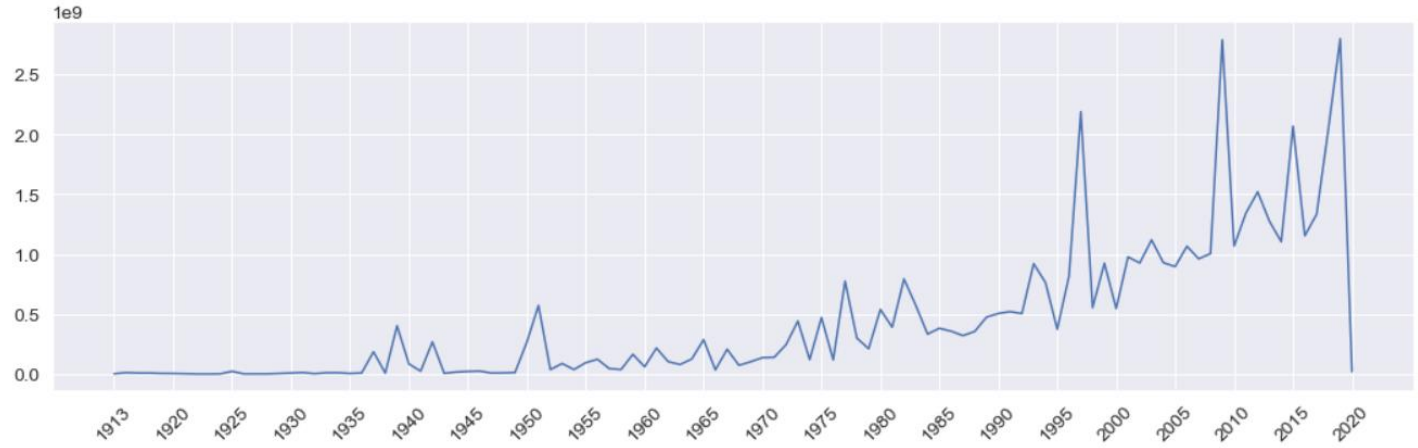
Two Pirates of the Carribean films occupy the top spot in this list and 5th spot with a staggering budget of over 300 million dollars. In between superhero movies collections Avengers: Endgame, Avengers: infinity war, Justice League were present.



The Pearson r value of 0.73 between the two quantities indicates a very strong correlation.

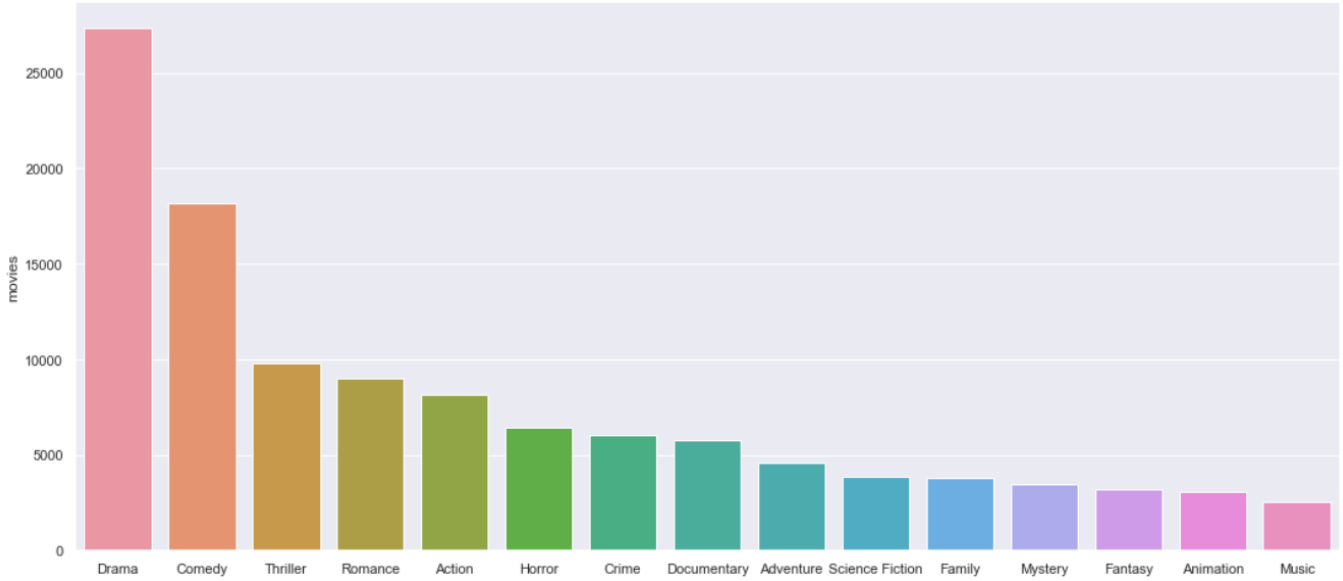
3.11 Revenue

The mean gross of a movie is **63.1 million dollars** whereas the median gross is much lower at **12.9 million dollars**, suggesting the skewed nature of revenue. The lowest revenue generated by a movie is just 1 dollar whereas the highest grossing movie of all time has raked in an astonishing **2.78 billion dollars**.

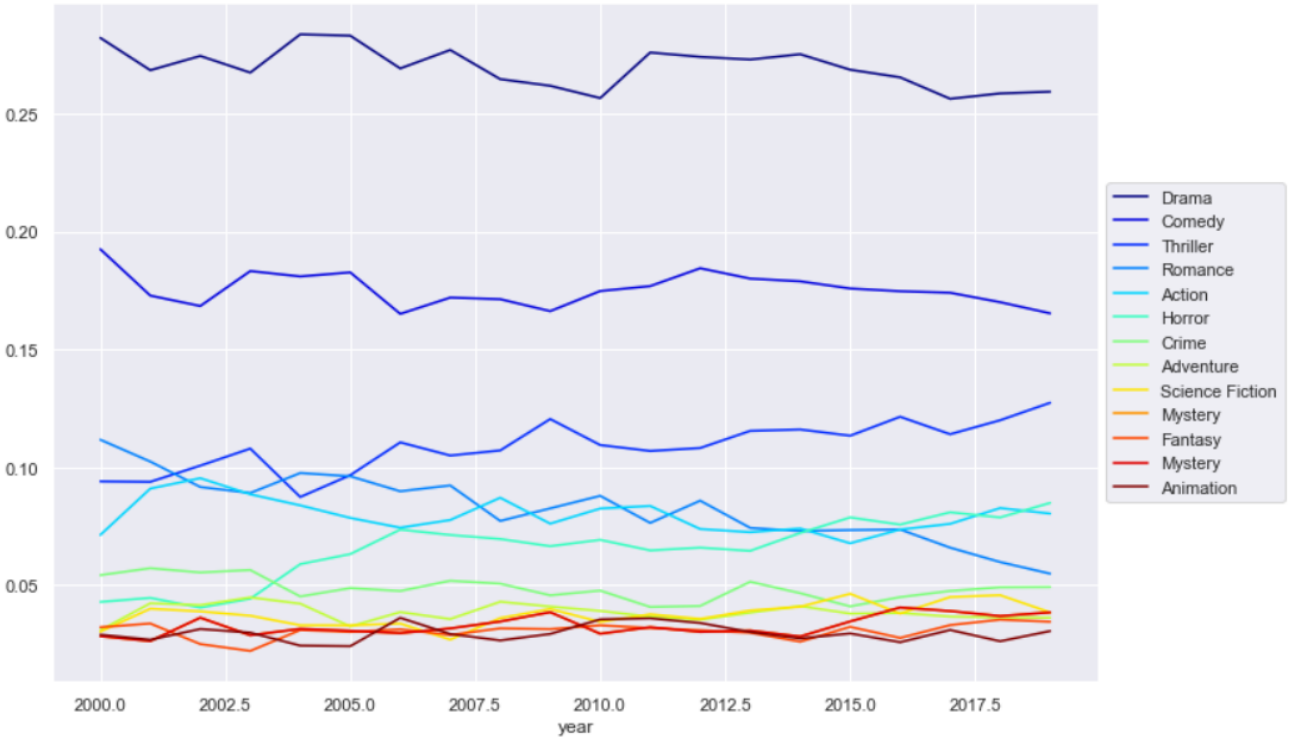


As can be seen from the figure, the maximum gross has steadily risen over the years. The world of movies broke the 1 billion dollar mark in 1997 with the release of Titanic. It took another 12 years to break the 2 billion dollar mark with Avatar. Both these movies were directed by James Cameron.

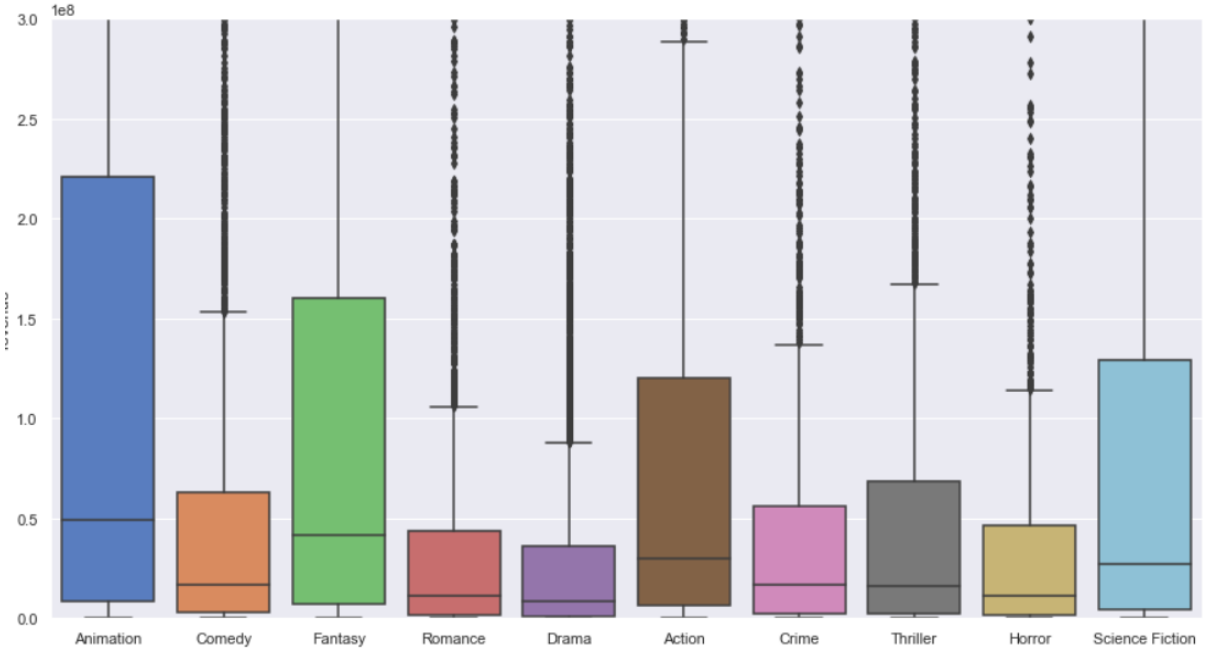
3.12 Genres



Drama is the most commonly occurring genre with almost half the movies identifying itself as a drama film. Comedy comes in at a distant second with 25% of the movies having adequate doses of humor. Other major genres represented in the top 10 are Action, Horror, Crime, Mystery, Science Fiction, Animation and Fantasy.

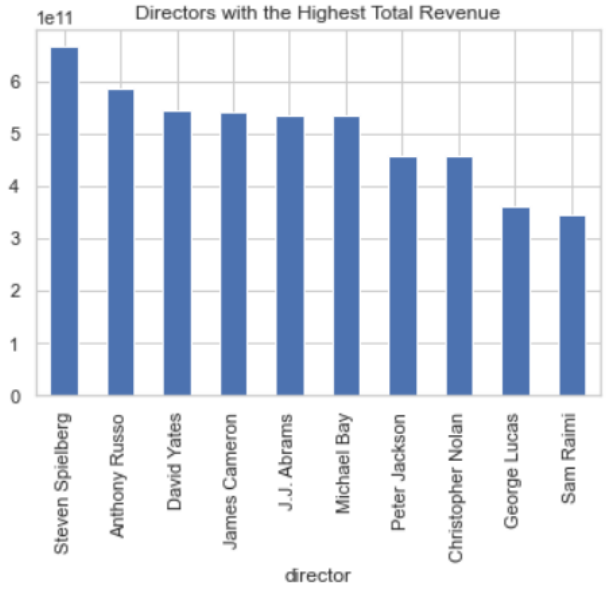


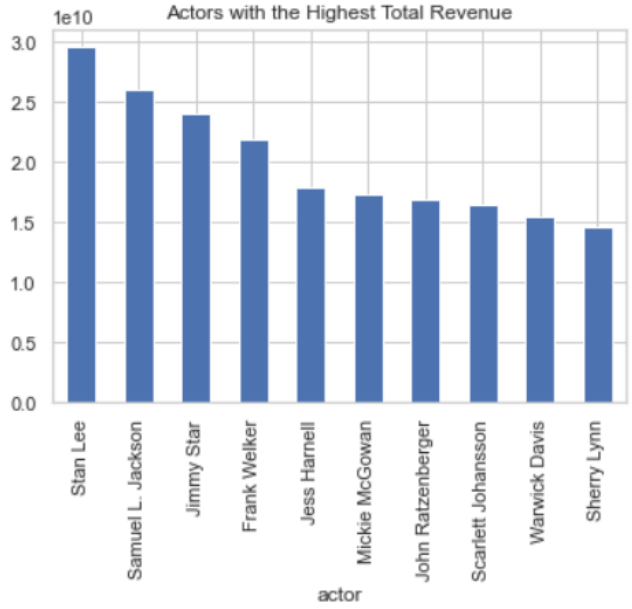
The proportion of movies of each genre has remained fairly constant since the beginning of this century except for Drama. The proportion of drama films has fallen by over 5%. Thriller movies have enjoyed a slight increase in their share.



Animation movies has the largest 25-75 range as well as the median revenue among all the genres plotted. **Fantasy and Science Fiction** have the second and third highest median revenue respectively.

3.13 Cast & Crew





4. Regression: Predicting Movie Revenue

Predicting Movie Revenues is an extremely popular problem in Machine Learning We will be using TMDB's Popularity Score and Vote Average as our features in our model to assign a numerical value to popularity. However, it must be kept in mind that these metrics will not be available when predicting movie revenues in the real world, when the movie has not been released yet.

4.1 Feature Engineering

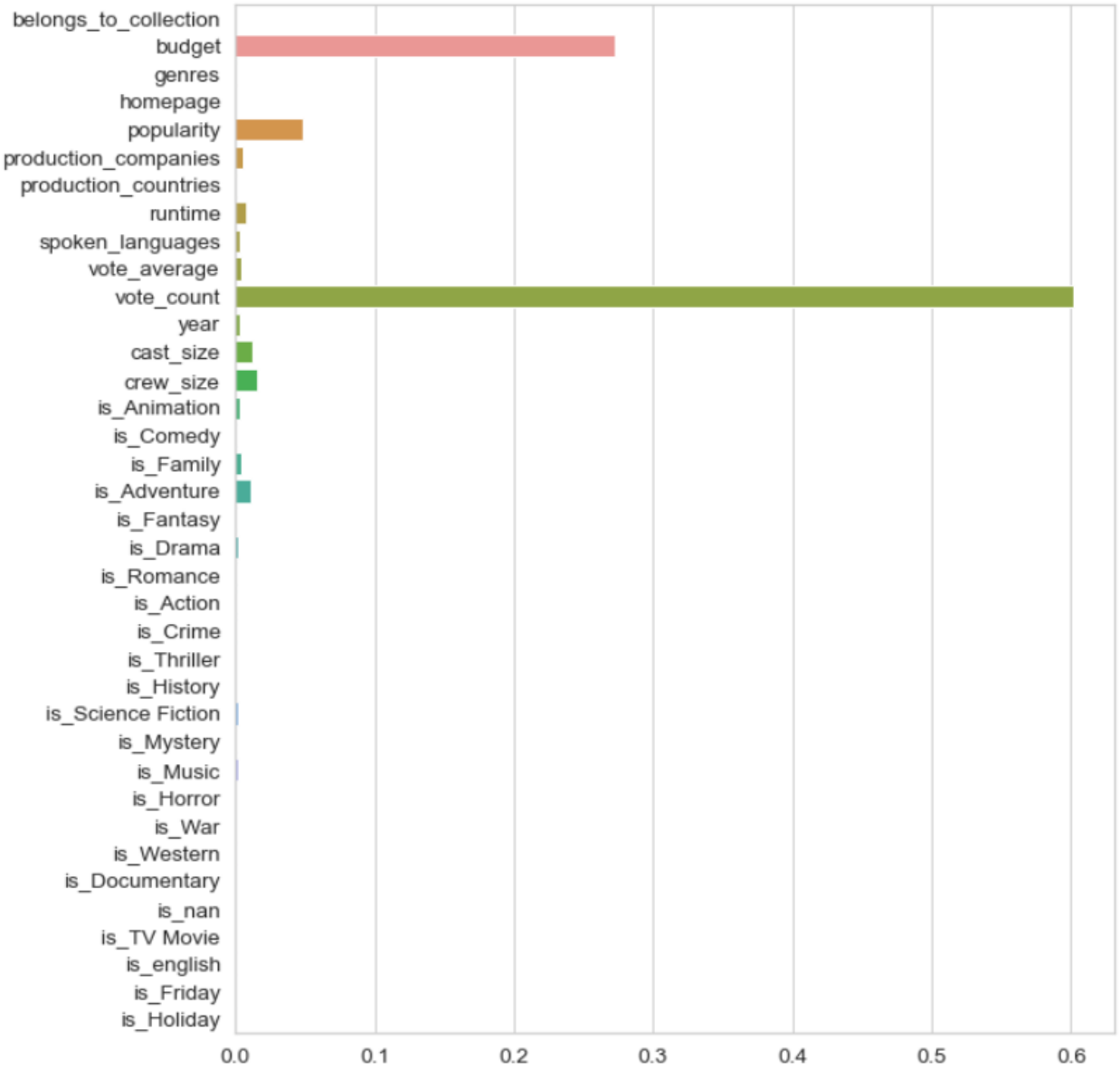
We will perform the following feature engineering tasks:

1. **belongs_to_collection** will be turned into a Boolean variable. 1 indicates a movie is a part of collection whereas 0 indicates it is not.
2. **genres** will be converted into number of genres.
3. **homepage** will be converted into a Boolean variable that will indicate if a movie has a homepage or not.
4. **original_language** will be replaced by a feature called **is_foreign** to denote if a particular film is in English or a Foreign Language.
5. **Production_companies** will be replaced with just the number of production companies collaborating to make the movie.
6. **Production_countries** will be replaced with the number of countries the film was shot in.
7. **day** will be converted into a binary feature to indicate if the film was released on a Friday.
8. **month** will be converted into a variable that indicates if the month was a holiday season.

4.2 Model

The model that I choose for regression is the Gradient Boosting Regression. The Coefficient of Determination Score obtained by the regressor was 0.79.

4.3 Feature Importance



We notice that **vote_count**, is the most important feature to our Gradient Boosting Model. This goes on to show the importance of popularity metrics in determining the revenue of a movie. **Budget** was the second most important feature followed by **Popularity**.

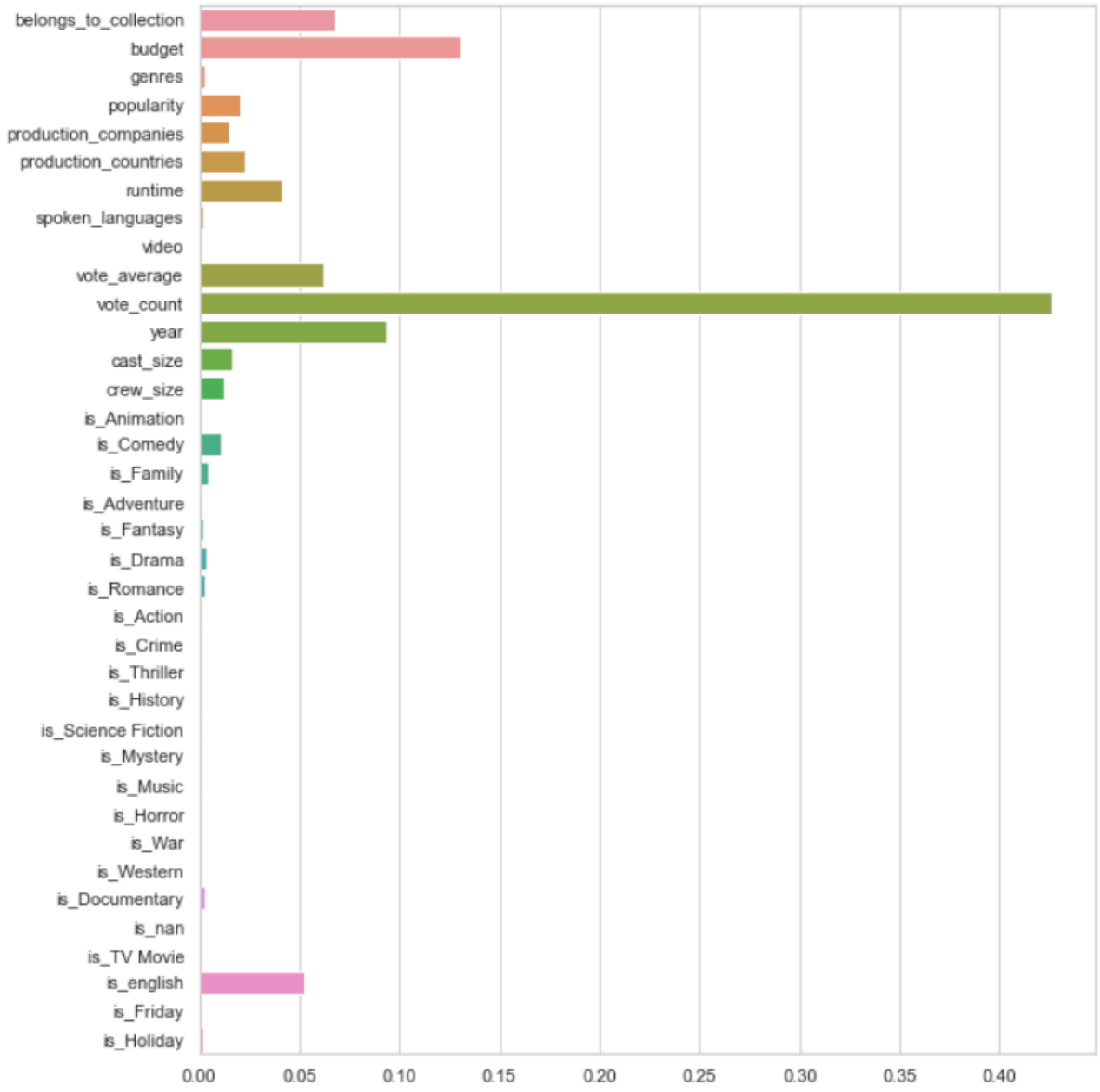
5. Classification: Predicting Movie Success

The Classification model uses the same Feature Engineering steps as those followed by the Regression Model built in the previous section.

5.1 Model

The model that I choose for classification is the Gradient Boosting Classifier. The model showcased an accuracy of 77% with unseen test cases

5.2 Feature Importance



We see that **Vote Count** is once again the most significant feature identified by our Classifier. Other important features include **Budget**, **belongs_to_collection** and **Year**. With this, we will conclude our discussion on the classification model and move on to the main part of the project.

6. Recommendation Systems

A **Recommender System** refers to a **system** that is capable of predicting the future preference of a set of items for a user, and recommend the top items.

6.1 Simple Recommender

The Simple Recommender offers generalized recommendations to every user based on movie popularity and (sometimes) genre. The basic idea behind this recommender is that movies that are more popular and more critically acclaimed will have a higher probability of being liked by the average audience. This model does not give personalized recommendations based on the user.
I used the TMDB Ratings to come up with our Top Movies Chart. I used IMDB's weighted rating formula to construct my chart.

The next step was to determine an appropriate value for m, the minimum votes required to be listed in the chart. I used 95th percentile as the cutoff. In other words, for a movie to feature in the charts, it must have more votes than at least 95% of the movies in the list.

	title	year	vote_count	vote_average	popularity	genres	wr
14909	Inception	2010	27268	8	41.479	[Action, Science Fiction, Adventure]	7.926664
21059	Interstellar	2014	23833	8	65.838	[Adventure, Drama, Science Fiction]	7.916448
12227	The Dark Knight	2008	23385	8	71.553	[Drama, Action, Crime, Thriller]	7.914902
2908	Fight Club	1999	20162	8	46.674	[Drama]	7.901832
24872	Avengers: Infinity War	2018	19819	8	186.028	[Adventure, Action, Science Fiction]	7.900201
352	Pulp Fiction	1994	19623	8	54.985	[Thriller, Crime]	7.899245
18932	Django Unchained	2012	19162	8	46.684	[Drama, Western]	7.896921
411	Forrest Gump	1994	18956	8	42.077	[Comedy, Drama, Romance]	7.895847
2523	The Matrix	1999	17814	8	49.458	[Action, Science Fiction]	7.889465
4925	The Lord of the Rings: The Fellowship of the Ring	2001	17795	8	51.270	[Adventure, Fantasy, Action]	7.889352

These are the top 10 movie recommendation from the Simple Recommender which can be used for a new user.

6.2 Content Based Recommender

My approach to building the recommender was extremely hacky. What I did was create a metadata dump for every movie which consisted of genres, director, main actors and keywords. I then used a Countvectorizer to create a count matrix. I then calculated the cosine similarities and returned movies that are most similar.

I also added a mechanism to remove bad movies and return movies which are popular and have had a good critical response

I took the top 25 movies based on similarity scores and calculate the vote of the 60th percentile movie. Then, using this as the value of m, I calculated the weighted rating of each movie using IMDB's formula like I did with the Simple Recommender.

In [81]:
```
improved_recommendations('The Dark Knight')
```

Out[81]:

	title	vote_count	vote_average	year	wr
6384	The Prestige	10590	8	2006	7.819507
7731	The Dark Knight Rises	16707	7	2012	6.929488
5977	Batman Begins	15004	7	2005	6.921901
7304	Kick-Ass	8850	7	2010	6.872220
7177	Law Abiding Citizen	3258	7	2009	6.697171
8123	Kick-Ass 2	4542	6	2013	5.923193
1127	Batman Returns	4191	6	1992	5.917817
4320	Daredevil	3515	5	2003	5.094919
8799	Batman v Superman: Dawn of Justice	13884	5	2016	5.028002
1240	Batman & Robin	3413	4	1997	4.291681

6.3 Collaborative Filtering

The content based engine suffers from some severe limitations. It is only capable of suggesting movies that are close to a certain movie. That is, it is not capable of capturing tastes and providing recommendations across genres.

Also, the engine that I built is not really personal in that it doesn't capture the personal tastes and biases of a user. Anyone querying our engine for recommendations based on a movie will receive the same recommendations for that movie, regardless of who s/he is.

Therefore, I used a technique called Collaborative Filtering to make recommendations to Movie Watchers. Collaborative Filtering is based on the idea that users similar to a me can be used to predict how much I will like a particular product or service those users have used/experienced but I have not.

I did not implement Collaborative Filtering from scratch. Instead, I used the Surprise library that provides extremely powerful algorithms like Singular Value Decomposition (SVD) to minimize RMSE (Root Mean Square Error) and give great recommendations.

```
Evaluating RMSE, MAE of algorithm SVD on 5 split(s).

                  Fold 1  Fold 2  Fold 3  Fold 4  Fold 5  Mean    Std
RMSE (testset)    0.8752  0.8756  0.8738  0.8734  0.8804  0.8757  0.0025
MAE (testset)     0.6716  0.6696  0.6751  0.6722  0.6741  0.6725  0.0019
Fit time          15.07   14.96   14.14   15.07   14.88   14.82   0.35
Test time         0.44    0.74    0.48    0.48    0.38    0.51    0.12
```

6.4 Hybrid Recommender

The Hybrid Recommender brought together techniques from both Content Based and Collaborative Filtering Based engines to provide personalized Similar Movie Recommendations to Users based on their taste.

```
hybrid(1, 'Avatar')
```

	title	vote_count	vote_average	year	id	est
563	Terminator 2: Judgment Day	8533	8.0	1991	280	4.961209
1006	The Terminator	8698	7.6	1984	218	4.877801
8524	Star Wars: The Force Awakens	14989	7.4	2015	140607	4.743601
8281	X-Men: Days of Future Past	11772	7.5	2014	127585	4.739453
969	Aliens	6340	7.9	1986	679	4.729334
8533	Black Panther	16086	7.4	2018	284054	4.684835
7412	Alice in Wonderland	21	6.1	1933	25694	4.465827
8753	Star Trek Beyond	4984	6.7	2016	188927	4.459703
7138	Green Lantern: First Flight	224	6.6	2009	17445	4.415512
8059	Star Trek Into Darkness	7097	7.3	2013	54138	4.339972

```
hybrid(500, 'Avatar')
```

	title	vote_count	vote_average	year	id	est
969	Aliens	6340	7.9	1986	679	4.397864
563	Terminator 2: Judgment Day	8533	8.0	1991	280	4.384852
8524	Star Wars: The Force Awakens	14989	7.4	2015	140607	4.110229
8281	X-Men: Days of Future Past	11772	7.5	2014	127585	3.977752
1006	The Terminator	8698	7.6	1984	218	3.904374
7412	Alice in Wonderland	21	6.1	1933	25694	3.860915
8059	Star Trek Into Darkness	7097	7.3	2013	54138	3.698867
8533	Black Panther	16086	7.4	2018	284054	3.637653
7138	Green Lantern: First Flight	224	6.6	2009	17445	3.587362
4394	Ghosts of the Abyss	87	7.0	2003	24982	3.574494

We see that for our hybrid recommender, we get different recommendations for different users although the movie is the same. Hence, our recommendations are more personalized and tailored towards particular users.

7. Conclusion

This report highlighted the processes of data wrangling, inferential statistics, data visualization, feature engineering and predictive modelling performed on the Movies Dataset. All the results and insights gained as part of these processes were also highlighted. With these insights, a Gradient Boosting Regressor and Classifier were built to predict Movie Revenue and Success respectively with a Score of 0.78 and 0.8 respectively.

In addition, four recommendation engines were built based on different ideas and algorithms:

1. **Simple Recommender**: This system used overall TMDB Vote Count and Vote Averages to build Top Movies Charts, in general and for a specific genre. The IMDB Weighted Rating System was used to calculate ratings on which the sorting was finally performed.
2. **Content Based Recommender**: We built two content based engines; one that took movie overview and taglines as input and the other which took metadata such as cast, crew, genre and keywords to come up with predictions. We also devised a simple filter to give greater preference to movies with more votes and higher ratings.
3. **Collaborative Filtering**: We used the powerful Surprise Library to build a collaborative filter based on singular value decomposition. The RMSE obtained was less than 1 and the engine gave estimated ratings for a given user and movie.
4. **Hybrid Engine**: We brought together ideas from content and collaborative filtering to build an engine that gave movie suggestions to a particular user based on the estimated ratings that it had internally calculated for that user.