Project Luther: Film Production Budget ROI

Analysis: Torin Rettig

1. Proposal

Filmmaking is an expensive, high-stakes business and a tremendous amount of time, effort and resources are devoted to tracking a film's performance. While filmmakers want all of their films to be a breakout hit that has tremendous ROI, the first hurdle for revenues to clear is the cost to make the film.

But what are the factors that contribute to a film's financial success? The budget of the film? The popularity of the actors? The number of articles written about it? In this analysis we will create a linear regression model that predicts what proportion of a film's budget will be recouped in revenue, and in optimizing the model we hope to discover the factors that most influence a film's financial fate.

Clearly identifying these factors would be very useful for film productions that are always looking to maximize their financial success while pursuing their art.

Assumptions

- We will assume that the major factors that most affect ROI prediction are the revenues for opening weekend and specific intervals soon after release, both domestic and international.
- We will also look at qualitative data such as the articles written about the film, MOVIEmeter rating, the ratings breakdown of the film, both in terms of the rating itself and the number of votes determining that rating.

Methodology

- Through existing IMDb TSV files and web scraping, gather financial, cast, director, ratings and other details of films with a recorded budget and performance numbers that have ended their theatrical run. I decided on an interval of 2012 - 2017, which provided a total of 1319 films.
- Gather actor popularity details available through IMDb and other sites for the principals of these films.
- Build a linear regression to determine the factors that most greatly affect ROI on the film's budget.

Data

- Target - Proportion of budget recouped through gross revenue.
- Features - Film financial performance numbers.
- Features - Film ratings breakdown and MOVIEmeter rating at opening.
- Features - STARmeter rating and other details of principal cast and directors.
- Data Sources: IMDb, IMDbPro

Collected Data

1. TARGET = Gross Revenue Proportion of Budget (Gross Revenue/Budget), at end of theatrical run
2. Production budget (IMDb) - (Integer)
3. Opening weekend gross (IMDB) - (Integer)
4. US gross of film (IMDb) - (Integer)
5. Foreign gross of film (IMDB) - (Integer)
6. Worldwide gross of film (IMDb) - (Integer)
7. MOVIEmeter rating at launch (Integer)
8. Ratings Breakdown of film (IMDB) - (Float)
9. Ratings Breakdown number of votes (IMDb) - (Integer)
10. Articles written about film (IMDb) - (Integer)

Prediction

ROI on production budget

2. Tools

- Python
- Jupyter Notebooks
- Selenium
- Pandas
- Numpy
- Scikit-learn
- StatsModels

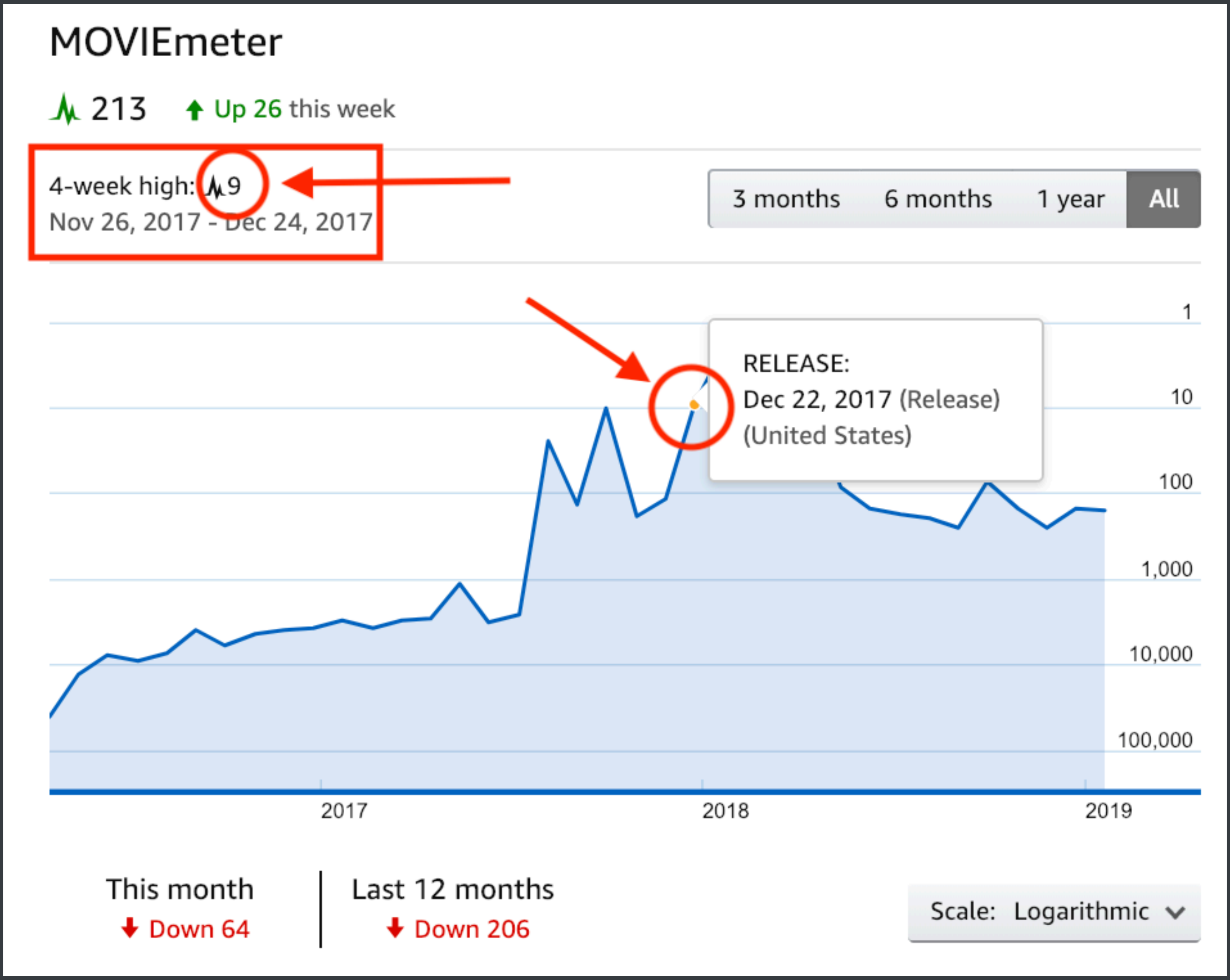
3. Data Collection

Source File: /project_luther_TR/web_scraping_functions.ipynb

Data collection involved getting both precompiled spreadsheets from IMBb and web scraping of film pages from IMDbPro. The spreadsheets contain ID information for films as well as useful attributes. The ID information provides a unique ID for each film that can be used to loop through the URLs of each film and scrape pages with each loop. They also contain genre and cast details that would make it unnecessary to scrape them directly from the pages.

On each film page I used Selenium and the Chrome web driver to find each element of interest and extract it. I used the XPath identifier to extract almost all of the needed elements.

One element that required a little more work was getting a past MOVIEmeter rating, the one at the time of film release, not the current one. IMDbPro provides an indirect way of doing this. There is a history of the MOVIEmeter rating as a chart at the bottom of each film page, and it identifies the point at which the film was released. However, the only way to get to it is through an interactive JavaScript element that only responds when moused over. I got around this by triggering the JavaScript object with the `driver.execute_script('return graphData')` command that reveals every single possible interval for the chart, with a unique string tag ('NON_TV_RELEASE') for the release date. Getting the values associated with this tag made it possible to get the MOVIEmeter rating for release.



I was also able to get daily box office numbers by taking the additional step of activating the box office tab on each page and scraping 60 days worth of daily gross revenue numbers, theater numbers and evolving MOVIEmeter ratings. Unfortunately the scraping was inconsistent with this step, resulting in a great deal of NaN values to the point where it was unusable. Most likely the issue has something to do with the amount of time between URL requests. I was able to improve the scraping of box office numbers by increasing the delay between requests via `time.sleep()`, but it still produced too many NaNs to be usable. I recommend continuing to adjust wait times and or investigating Selenium timeout options to address this if you want to get box office information.

Another note. This scraping technique takes a very long time and will need to be monitored or requires retry logic when a page cannot be reached. Each page currently requires about 20 seconds and with anywhere between 10 to a few hundred page requests there will be a server 500 error or timeout issue. Introducing retry logic can potentially get around these issues, but even barring any issues the scraping process will still take many hours.

- Data Sources
  - IMDb: TSV files provided freely by the IMDb site which are updated daily
    - Datasets: https://datasets.imdbws.com/
    - Descriptions of datasets : https://www.imdb.com/interfaces/
    - IMDb Pro: Film search pages to determine film set and film details pages to collect film details (attributes/features)
      - Search Page: https://pro.imdb.com/find?q=&ref_=hm_nv_search#keyspace=TITLE&type=movie&status=RELEASED&sort=ranking&year=2012-2017&budget=0.01-&gross=0.01-
      - Film details page example: https://pro.imdb.com/title/tt5580390/?ref_=search_title_result_5

4. Data Cleaning

Source File: /project_luther_TR/feature_engineering_and_modeling.ipynb

The data cleaning process was fairly straightforward. There were comparatively few NaN values in the final features and in examining the values the observations they were in could be safely removed from the data set without adversely affecting the overall distribution. There were a few rogue strings that were in the budget column that escaped the string to int conversion process that occurred during the web scraping and importing process, but there were only two and they were easily removed.

Additional preparation of the dataframe took the form of dropping unnecessary columns and converting the entire data frame to float values so for the regression process.

5. Feature Engineering

Source File: /project_luther_TR/feature_engineering_and_modeling.ipynb

Feature engineering was rather light. The one solid candidate for creating a new feature based on existing features was creating a new feature called OpeningProp, which is the proportion of the budget earned in the opening weekend of a film's release. This seemed like a good choice since much attention in Hollywood is put on the performance of a film in its opening weekend. That performance often turns out to be indicative of the overall box office performance of a film. This turned out to seemingly have the strongest correlations with the Target value.

6. Feature Selection and Modeling

Source File: /project_luther_TR/feature_engineering_and_modeling.ipynb

The initial approach to seeing relationships between features and the target was pair plotting and scatter/regression plots. The scales of the numerical values of the features were fairly differentiated so I ended up using the log transformed version of the data for most explorations.

The log transformed pair plot revealed some promising relationships. The strongest was between Target and OpeningProp, as mentioned above. There are others, as can be seen in the notebook, that showed potential as well, but in looking at them individually it wasn't totally clear which ones should stay and which ones should go from the model.

To get a better idea I moved into actual modeling. The first thing I tried was the Ordinary Least Square regression with StatsModels. This yielded an R2 of 0.646, with some promising coefficients with Budget, Ratings, RatingsVotes, Articles, and the aforementioned OpeningProp.

I then moved on to the regular 60/20/20, Train/Split/Validate approach. This produced an R2 score of 0.6959, a bit better than simple OLS, but it also for some reason zeroed out all coefficients except for OpeningProp (4.22) and Ratings (0.72). These are not surprising numbers in terms of their importance to film evaluation, but zeroing out everything else seemed a bit extreme.

Next I moved on to the regularization method of LASSO. I first tried 60/20/20 LASSO, after plotting the Mean Absolute Error to discover the optimum Alpha value. This LASSO approach had a 0.7956 R2 and put just about all of the features back into play, as all had coefficients.

I then tried automated regularization with both LASSO and Ridge cross validation. The cross validation method of obtaining the alpha is more rigorous so I applied that to both Ridge CV and LASSO CV. Both of them produced very similar scores with Ridge at 0.6961 and LASSO at 0.6951, and both kept all of the features in play with coefficient numbers. OpeningProp, Ratings and RatingsVotes were not surprisingly the strongest, but all seemed to contribute to the model.

Since the cross validation method is much recommended over simpler methods the choice was between Ridge CV and LASSO CV, and because LASSO CV is supposed to handle smaller feature sets better, LASSO CV is the model I would go with at this point.

7. Future Work

Overall I didn't get nearly as far in the feature engineering and modeling process as I wanted. I also didn't get to include many of the potential features I was aiming for. In the future I'd like to continue working on this model, adding a number of additional features such as genre, Metacritic, principal cast details (STARmeter rating, Instagram/Twitter followers, Age, etc), daily revenue over various periods up to 60 days post-release, and derived features related to International and North American revenue over time. I think they all have the potential to add predictive power.

Additionally I'd like to dive deeper into the modeling process to improve my skills. There are many approaches I simply didn't have time to explore due to time constraints. I definitely look forward to continuing to work with this data and project idea in the future.