Movie Review Bombing: Rotten Tomatoes Critic vs. Audience Score

An Introductory Predictive Analysis

Rahul Rajeev

Bellevue University

DSC550-T301 Data Mining

Dr. Brett Werner

May 27, 2023

Introduction

Rotten Tomatoes stands out as the premier platform for critic reviews, and for that reason, most people tend to believe the critic scores over watching and judging the movies for themselves. I am sometimes one of those gullible people although for only movies that I am not very invested in. That being said, I believe critics can "review bomb" because they have viewing parties and review the movie before the official release. Review bombing a movie on Rotten Tomatoes can dissuade people from watching the movie for themselves and resulting in a worse box office result.

It’s important to solve this issue for two reasons. The first being to experience films as people have in the past, without any prior knowledge of the film except for the title and a clear mind. Since films were criticized by single film critics and without the assistance of presentation through newspapers and the internet, most people did not know what to expect in the film, which made the experience all the more enjoyable. The second reason is to help smaller production studios. Other than the larger, more internationally known movie studios such as Universal, MGM, Disney, there are smaller studios who make their movies as a passion project rather than for profit. Unfortunately, critics and other audience reviewers often have varying opinions about writers, directors, and other factors of a movie per studio. This opinion can lead to reviews that are biased, hurtful, and can take away from the success of a particular movie.

If I were to pitch this problem to stakeholders, I would preface it in a number of ways. To movie studios, I would preface this analysis as a way for them to profit. They can engineer the perfect movie, with the highest rated directors, writers, genre, sound mix, etc. To Rotten Tomatoes and other review sources, I could demonstrate the issue of having too many opinions allowed to generate a review for a movie. Showing how accurately the model can predict the rating of a movie, it may be wise to try and limit its usage to those who are less opinionated and more passionate about enjoying the movie for what it truly is, regardless of the studio or those who created it.

I obtained the data from a kaggle dataset which shared both the review and movie data as two separate CSV files. I then manipulated and joined the datasets into one using a Jupyter notebook and the appropriate pandas libraries. The documentation of the data is clear and each column is thoroughly explained. Since it is a kaggle post, the data isn't the most recently updated, and it only goes up to a Thor movie that was released in 2022. The link can be found in the sources section

Summary of Project Progress

Milestone 1 - Early Data Analysis and Visualizations

I decided to start with figuring out the distribution of reviews by genre and directors.

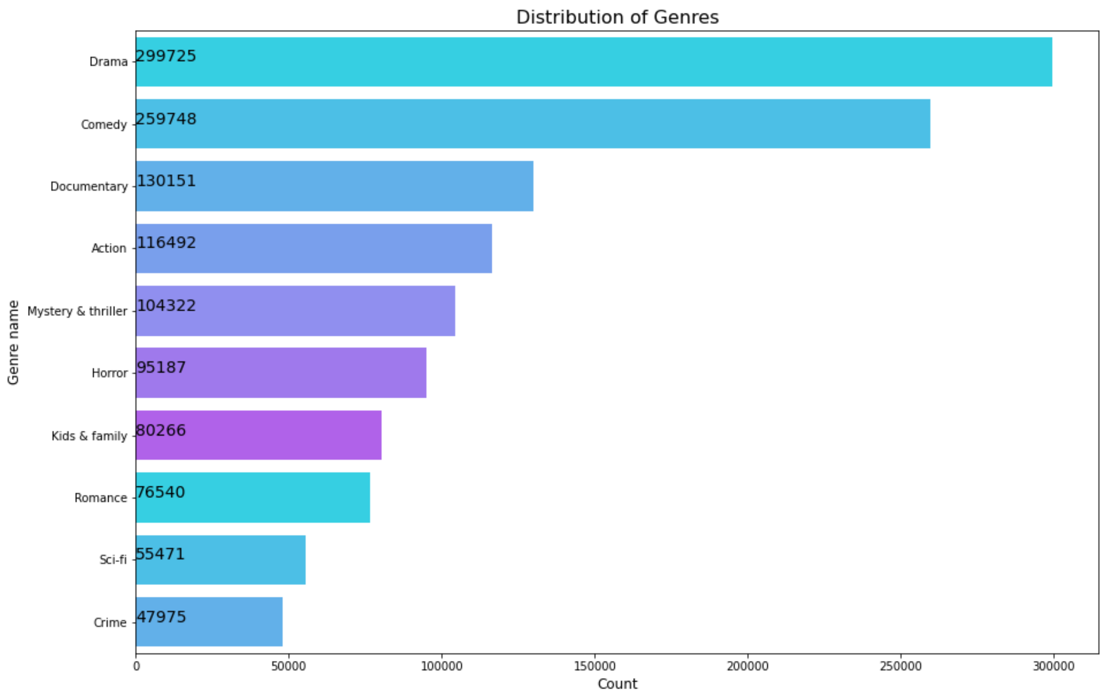

Figure 1 - Genre Popularity in Movies

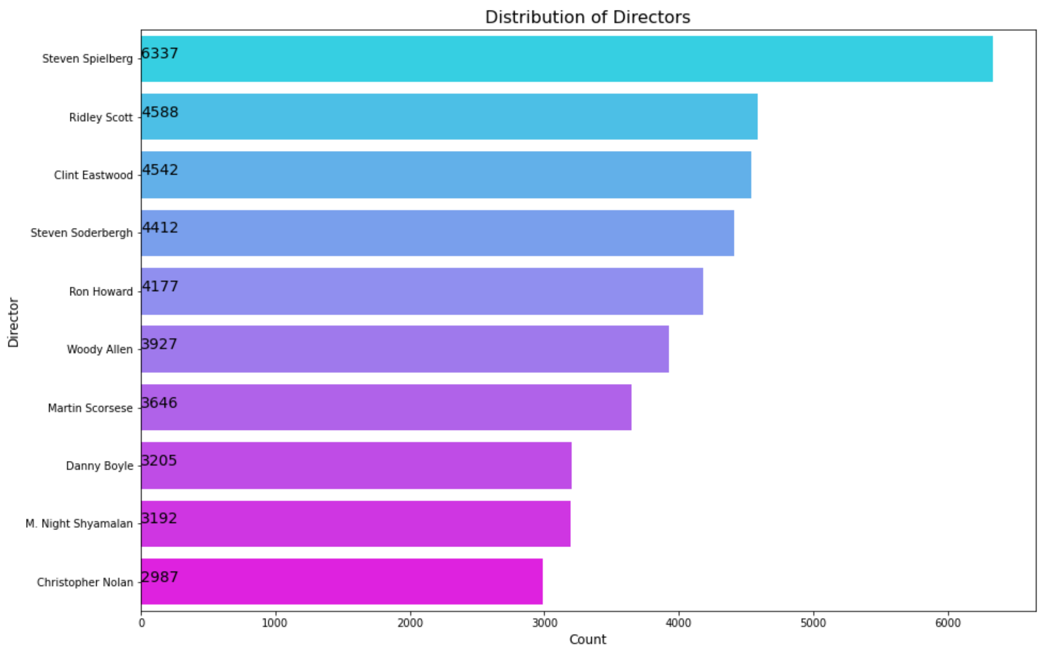

Figure 2 - Directors Popularity in Movies

The top genres are documentary, comedies, mysteries and thrillers, drama, action, crime, romance, sci-fi, horror, and kids and family. The other genres are most likely mixes of multiple genres, which is harder to categorize. Therefore, I cut down the genre column and selected only the first genre for each movie. The bar graph of top directors are also shown, and top directors usually have critics divisive on new projects. So looking at top directors I have an idea of who they are. Steven Spielberg is definitely known for his directing so movies from him will most likely have high scores from both the audience and critics.Ridley Scott comes in a close second, then Clint Eastwood. Both of them are noteworthy directors which also tends to gather critics to review their movies.

Next I decided to look at the distribution of audience rating and tomato meter rating.

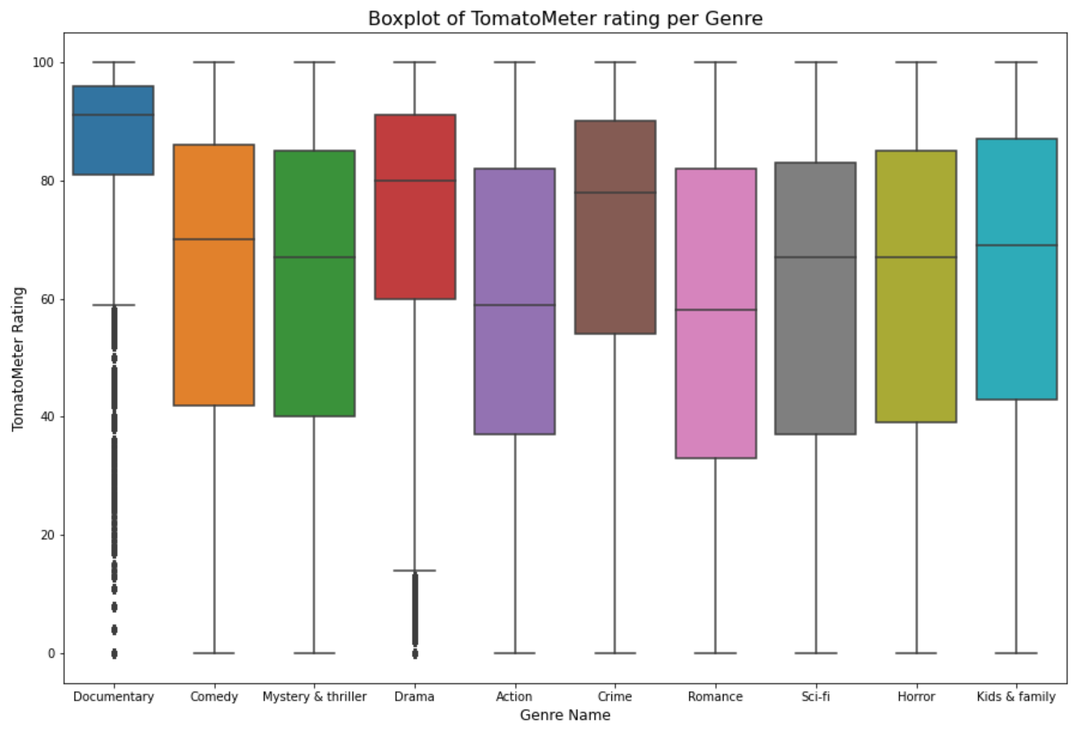

Figure 3 - Tomato Meter Rating Distribution

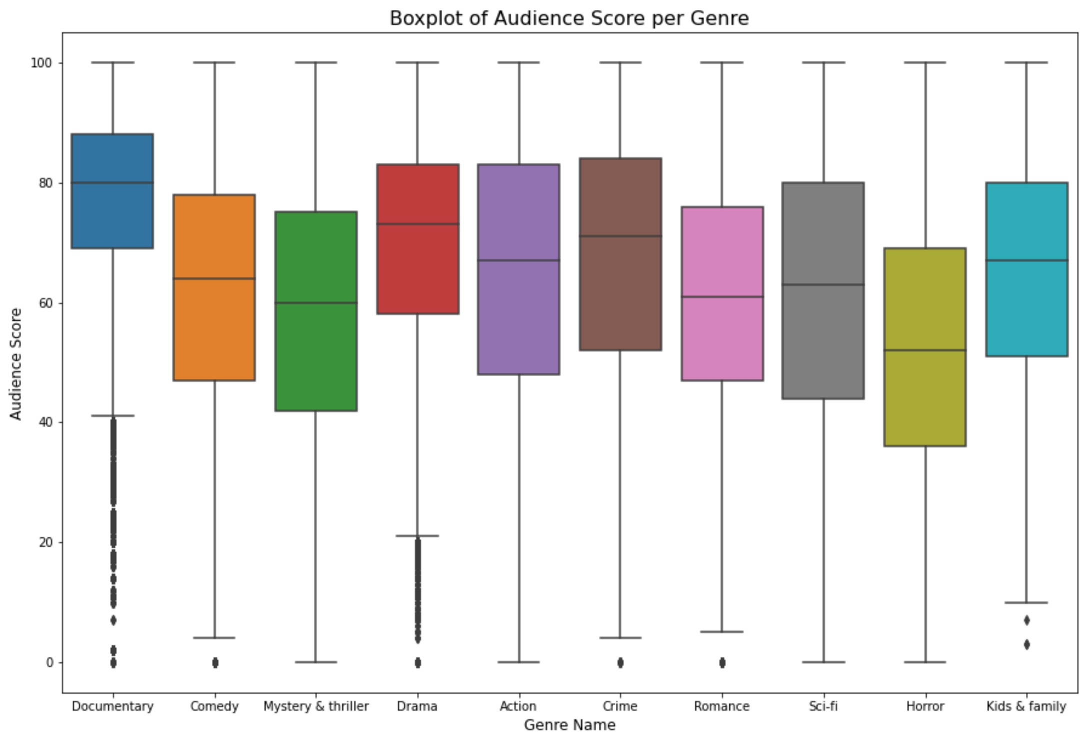

Figure 4 - Audience Rating Distribution

The next two boxplots show the distribution of scores from audience and critics from each genre. It is a bit apparent that audience scores are more tightly packed compared to critic scores as critics are more knowledgeable about each genre including information about directors and writers. They are aware of the strengths and weaknesses of the people that make the movies and therefore deliver the truth more harshly. Audiences are more interested in the movie itself, and therefore seem to clump around each other.

Milestone 2 - Data Preparation

I started data preparation by dropping columns that won't have an affect on the model as well as the original genre column that had all the genres combined in a comma separated list. I also dropped the critic score column as it had too many errors in entry and strings of fractions. Then I dropped rows that had missing values for tomato rating and audience rating because that is my target value. Since the dataset also included worded reviews from each critic, I wanted to pass them in as a value into the model. So I decided to use my knowledge of sentiment, and using TextBlob and PorterStemmer, I replaced the worded reviews with sentiment scores between -1 to 1 where closer to -1 signifies a negative review and closer to positive 1 signifies a positive review.

Going into milestone 3, I found myself running into an issue of dimensionality. Since my categorical columns had too many unique values, assigning them to dummy variables would create a very large amount of columns which would exceed the memory of my computer. For example, if I were to use dummy variables for the writers column, I would have about 30473 columns. It took me a while to decide on exactly what to do, but I decided on a method that I believed would be a fair way to assign the unique values. I created buckets of values specific to each column based on the value counts and a relative fairness. Then I used these buckets to assign specific frequencies of values to a category from 1 to 5. That way it was easier to assign dummy variables to a smaller set of columns.

For example, starting with the publication name, the reviews are all pretty close together, so it's hard to determine which ones will be the most frequent. I replaced the publication names with categories where the publication has more than 2500 reviews were assigned to category 0, more than 2000 reviews were assigned to category 1, more than 1500 reviews were assigned to category 2, more than 1000 were assigned to category 3, and then more than 500 reviews were assigned to category 4. The rest should be set to “other” or category 5. Each categorical column I decided to approach with a unique set of buckets and the SoundMix column had its very own approach as well. It was a temporary fix to save memory while performing computations for the predictive model. But if I had no limit on computational power, this bucketing wouldn’t be required.

Once I replaced all the frequencies with categories, it was easier to convert to dummy variables The final transformation I worked on was replacing the tomato rating and audience rating numerical values with binary discrete values. I used the buckets provided by Rotten Tomatoes for the most accurate representation and used a similar method for replacing the categorical columns with the ranges in a dictionary. Replacing the column of continuous values to discrete values made it easier to pass into a classification model rather than trying to use a regression model.

Milestone 3 - Model Building

I will train the model based on parameters the audience may know as public knowledge to predict audience scores, and for critic scores, I will include additional parameters because the other criteria would matter. For example, the director of a movie might place bias on a critic's review, but not for an audience review. For an audience member, the runtimeMinutes may matter but for critics it absolutely matters. Genre is a parameter known to both the audience and the critic. Creating models and training it on different parameters based on audience and critic knowledge may lead to an understanding of where critics give a score opposite to the expected one. Here's what the rundowns for each one:

1. To predict audience score, I will use the rating, runtime, first_genre.

2. To predict critic score, I will use all of the previous features and add on istopcritic, reviewstate, director, writer, distributor, soundmix, and polarity to keep track of the reviews.

Finally,. I tried using three of the binary classification models to predict in each of the situations to see the benefits of each model compared to each other, and additionally, passed the model through a grid search to get the optimal hyperparameters.

Results of Audience Rating

Without much need to transform data, I converted the necessary columns to dummy columns and split the test/train data effectively. Then I ran grid searches on decision tree, KNN, and logistic regression in order to get the best model for classification. For the audience rating, since there wasn't a large dimensionality, KNN had decent results, and decision trees/logistic regression had similar results. Luckily the KNN model wasn't overtrained for this particular chunk of the dataset, so I could pick an optimal K for clustering.

Results of Tomato Rating

Data transformation was required for this particular task because the categories I wanted to pass in had high variability in terms of categories. For example, there were many unique directors, writers, and review publications. In order to reduce this variability and simplify the data, I bucketed the values by the frequency they appeared in, each bucketing system catered thoughtfully for each column. Then I converted the columns to dummy columns, and ran the models. Since there was a larger dimensionality for this dataset, still 500000 rows but 61 columns instead of 25 columns. Because of this large difference, KNN started to fail, and the initial number of neighbors were very overfit. Unfortunately, I moved on to using Logistic Regression and Decision Tree which both ended up having a similar result in accuracy. If I were to move this model to live, I would highly suggest using one of those two. They don't seem to be overfit.

Conclusion

The model building told me that reviews from both audience and critics are dependent on the factors of the movie. However, how accurately the rating can be predicted is still dependent on the methods used.

The model isn’t ready to be deployed because there are definitely some additional components I can work on for the model. For example, the bucketing of categories for each column may not be the best way to approach the problem. In addition to that, the confusion matrices revealed that the models for audience rating and tomato rating have similar amounts of false-false results and true-true results. I think it would be better to have more true-true results before deploying the models. Additionally, since I have tried multiple binary classification models, I’m not sure which of them would work the best with validation sets or live data as of yet. For the audience rating, it was clear that the KNN classifier worked the best, but the tomato rating had similar scores with the decision tree and logistic regression. The KNN classifier model seemed to be overfit, so I decided to move away from that.

Another reason why the model isn't ready to be deployed is that I accidentally scaled the dummy columns in addition to the continuous columns, which can lead to false results. I could push this as an additional task to complete when working on this model for a live environment. I did try scaling only the continuous columns and saw that there definitely was a difference, but I wanted to save it for a later time so I could work on current projects.

My recommendations for the models to be passed to live would be to test it in an environment where memory is not limited by the notebook or coding interface. The main issue with passing the categorical columns was that it would create an excess of dummy columns, almost 30,000 for the director column. Once I have this 'unlimited' environment, I feel that the model will be able to predict any movie without the use of bucketing.

Additional opportunities that still need to be explored are trying other classification models such as SVM or SMOTE to fit the data. I tried SVM on one of the datasets, but it seemed to take up too much computational power and didn't actually finish processing. I could also try passing in the movies that were missing columns and impute the missing values with the median/mode of the specific categorical column. I could also try to use linear, nonlinear regression to try and estimate the audience or tomato score