Predicting Box Office Revenue for Movies

Matt Vitelli mvitelli@stanford.edu

Abstract—This project analyzes the use of features extracted from network representations of the Internet Movie Database (IMDB). I show that through the use of these features it is possible to build more powerful prediction models compared to common baseline methods.

I. INTRODUCTION

Box office revenue prediction is an important problem in the film industry that governs financial decisions made by producers and investors. Generally, these predictions are made using some basic statistical techniques as described in [1]. While these approaches are common practice, they often only provide a coarse estimate of revenue prediction before a film has been released. The goal of this project is to develop a computational model for predicting box office revenues based on public data for movies extracted from popular online movie databases.

II RELATED WORK

Movie revenue prediction has been studied in a variety of contexts ranging from economics and business to statistics and forecasting. Fundamentally, revenue prediction is a regression task in which we seek to estimate a single number representing the gross revenue based on a variety of factors.

In much of the movie revenue prediction literature, there are several features that are present across nearly all of the models, as discussed in [1], [2], and [3]. Namely, these features are MPAA rating, release date, and movie genre. Release dates are generally a powerful feature, as high-grossing films are typically released during the winter or summer seasons. MPAA ratings are usually

correlated with movie revenues, as R or NC-17 ratings tend to limit the total number of viewers.

In [2], the authors make use of a NLP-based prediction model that combines n-gram, part-of-speech features, and dependency relations with features extracted from film metadata. Their dataset consisted of 1,718 films and NLP data extracted from popular news sites. Additionally, they model revenue prediction as a continuous estimation task and seek to minimize the mean average error (MAN) between their observations and their predictions.

[3] utilized a two-layer neural network to categorize movie revenue into ten buckets. In this sense, revenue is modeled as a discrete quantity rather than a continuous one. The authors' dataset consisted of 834 movies from ShowBiz Data, a private movie database website. Their model is unique, as it was the only neural network model I could find related to the task of revenue prediction.

III. PROBLEM STATEMENT

In this paper, I describe revenue prediction as a discrete estimation task using supervised learning. Formally, we have a set of training examples $(\Theta^{(1)}, Y^{(1)}, \Theta^{(2)}, Y^{(2)}, ..., \Theta^{(n)}, Y^{(n)})$ where

each
$$\Theta^{(i)} = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_k \end{bmatrix}$$
 corresponds to a vector of input

features for a particular movie and each $Y^{(i)} \in R^{10}$ is a one-hot vector with an active index corresponding to one of ten possible revenue categories.

We can model the relationship between Θ_i and Y_i in a variety of ways. In this paper, I use two different types of models. The first represents the relationship to Θ_i and

 Y_i as a linear classifier coupled with a softmax activation function. More formally, we can describe the model as follows:

Let
$$\widehat{Y^{(i)}} = g(W\Theta^{(i)}), \widehat{Y^{(i)}} \in R^m, W \in R^{mxk},$$

$$g_i(x) = \frac{\exp(x_i)}{\sum_{i=1}^m \exp(x_i)}$$

We would like our predictions to best align with our ground truth observations Y_i . In order to accomplish this, we define a loss function $l(\Theta, Y)$ as follows:

$$l(\Theta, Y) = -\frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{m} Y_j^{(i)} \log(\widehat{Y^{(i)}})$$

Here $l(\Theta, Y)$ represents the cross-entropy loss function.

The second model is conceptually similar to the first model, only we represent the mapping between Θ_i and Y_i via an additional nonlinearity as follows:

$$\widehat{Y^{(t)}} = g(Uf(W\Theta_i)), \widehat{Y^{(t)}} \in R^m, W \in R^{hxk}, U \in R^{mxh},$$

$$g_i(x) = \frac{\exp(x_i)}{\sum_{j=1}^m \exp(x_j)},$$

$$f(z) = \tanh(z)$$

The model above is commonly referred to as a twolayer neural network with a tanh activation function. It is well known that adding additional layers increases the model's representational power. This model serves as a useful comparison for [3] and also allows us to analyze the performance between the simple linear classifier.

IV. FEATURES & GRAPH MODELING

I used features corresponding to [3] in conjunction with features I extracted from actor-actor, actor-movie, and move-movie relationships. More specifically, my base set of features consisted of the following:

1. Indicator features for each movie genre. Movies spanning multiple genres have multiple active indicator features.

- 2. Indicator features for each MPAA rating.
- 3. Competition, as defined by [3], which is high during June and November, medium during July and December, and low during all other months.

This base set of features constitutes a subset of the features used in [3]. Unfortunately, the dataset in [3] was not made publically available and several of the features used in [3] were not applicable to my dataset. In addition to these base features, I created a set of features meant to encapsulate certain graphical properties extracted from actor-actor, actor-movie, and movie-movie relationships.

First, an actor-actor graph is created, where nodes correspond to actors and edges exist between actors if they have worked on at least one movie together. The actor-actor features are described as follows:

- 1. Histograms of degree centrality measures. This feature is meant to encode the distribution of degrees present among actors in a particular movie.
- 2. Histograms of betweenness centrality measures. This feature is meant to encode the distribution of betweenness centralities, which I suspected would be closely associated with independent films based on inspecting the actors with the highest betweenness scores. Because these films' box office success generally falls within a certain range, I speculated that this would be a useful feature.
- 3. Histograms of closeness centrality measures. This feature is meant to encode the distribution of closeness centralities, which I figured would likely be a good indicator for whether or not a movie had a large proportion of popular actors, and hence had a higher likelihood of box office success.

I also create an actor-movie graph where one set of nodes corresponds to actors and a second set of nodes corresponds to movies. An edge exists between an actor and a movie if the actor was present in the movie.

The actor-movie features are described as follows:

1. Histogram of average aggregate ratings for actors. An actor's average aggregate rating is calculated as

the mean of each of their associated movies' IMDB ratings, MetaScore ratings, and Rotten Tomatoes ratings. The intuition behind this feature is that actors associated with highly rated films are likely to attract consumers and hence will be associated with greater box office success.

2. Histogram of actor degree distributions. This feature encodes the distribution of the number of films each actor has been associated with. It is used primarily as a method for measuring how many stars are in a particular movie, with the assumption that stars are likely in a larger number of movies than non-stars.

Finally, a movie-movie graph is created, where nodes correspond to movies and edges are created between two films if they were released within a similar timeframe. I measure a particular movie's competition value as the movie's degree. The intuition behind this feature is that movies with high degrees were likely competing more for consumer attention.

v. Dataset Acquisition and Technical Approach

My dataset was aggregated primarily from the Internet Movie Database (IMDB) and the Open Movie Database (OMDB). First, I downloaded the public lists actors. directors, and films from Unfortunately, the film public lists did not contain much metadata about each film and did not contain any information regarding box office revenues. The dataset consisted of roughly 1.2 million films. In order to get metadata and box office revenues, I scraped OMDB for each of the 1.2 million films and aggregated the metadata for each film into a single database. The entire process took roughly forty hours to complete. Once the metadata was extracted for each film. I filtered the list of films to films that had valid box office revenue data. Of the 1.2 million films, only 3,593 of them had valid box office revenue data.

Of the 1.2 million films extracted from IMDB around 230,000 of them had metadata available from OMDB. I used these films as a means of computing different graphical features for actor-movie, actor-actor,

and movie-movie relationships. Unfortunately, my actoractor graph ended up being extremely dense (around 1 billion edges), making it impractical for computing betweenness and closeness centrality measures. To compensate for this, I used an iterative procedure in which nodes were selectively removed from the actoractor graph based on their degree until the total number of edges in the graph was within some computationally tractable threshold. This reduced actor-actor graph was then used to compute betweenness centrality and closeness centrality measures for each of the remaining actors. As such, the centrality measures used in my models are approximations.

I implemented and tested several models using the features outlined above. All of the programming for this project was done in Python. I utilized OMDB.py, a Python library for communicating and parsing data from OMDB, as a means of extracting metadata from the set of movies I was able to assemble from IMDB. In addition, I used Keras, a popular machine learning library, as a means of training and validating my models. All of the models were trained using Stochastic Gradient Descent (SGD) until convergence (around 50 epochs). For the two-layer neural network model, I use a hidden layer size equal to to the input feature vector size. I experimented with larger and smaller hidden layer sizes, but found that due to the limited amount of training data I tended to get the best results if I kept it proportional to the input size. The ten buckets for movie revenue are estimated by fitting ten clusters to the revenue data points using K-means. I used seven bins for each of the histogram features described above.

VI. EXPERIMENTS AND EVALUATION

I evaluated the baseline features with both the single-layer softmax model and two-layer neural network in order to get a rough sense of the performance characteristics of these features alone. I also tested each set of features individually. As my metric for performance, I used the traditional F1 score. All of the models listed in the table below were trained independently fifty times and the highest performing

models from each category were used for the metrics below. The table below shows the F1 scores evaluated on a test set consisting of 360 films.

	1 Layer Model	2 Layer Model
Baseline Features	54.06%	47.05%
Baseline +	53.76%	47.63%
Movie-Movie		
Competition		
Features		
Baseline +	57.1%	55.98%
Aggregate Rating		
Histogram		
Features		
Baseline + Actor	54.3%	56.54%
Movie Degree		
Histogram		
Features		
Baseline +	56.26%	43.17%
Closeness		
Centrality		
Histogram		
Features		
Baseline +	14.48%	57.66%
Betweenness		
Centrality		
Histogram		
Features		
Baseline +	56.54%	45.12%
Degree Centrality		
Histogram		
Features		

After evaluating the performance of the features above, I ran another set of experiments on high performing combinations of features. A summary of the highest performing feature combinations is shown in the table below:

	1 Layer Model	2 Layer Model
Baseline +	57.66%	55.71%
Aggregate		
Rating		
Histogram +		
Actor Movie		
Degree		
Histogram		

Features		
Baseline +	58.21%	57.38%
Aggregate		
Rating		
Histogram +		
Closeness		
Centrality		
Histogram		
Features		
Baseline +	59.33%	55.43%
Movie-Movie		
Competition +		
Aggregate		
Rating		
Histogram		
Features		
Baseline +	61.28%	54.87%
Aggregate		
Rating		
Histogram +		
Degree		
Centrality		
Histogram		
Features		

There are a couple of interesting trends that are observable from the data above. First, nearly all of the two-layer neural network models seem to underperform the single-layer models with the exception of the baseline features and betweenness centrality histogram features, which significantly outperforms the singlelayer model. I believe this is due two possible explainations: 1) betweenness centrality histograms may be reasonably similar across all movies in my dataset and hence not a good feature or 2) the betweenness centrality histograms are very sparse, as they were computed from a greatly reduced graph and as a result, some of the actors in my movie database do not have betweenness scores, leading to inaccurate histograms. I suspect the drop in performance between the one-layer models and the two-layer neural networks is due to the fact that my dataset is fairly small relative to the number of parameters in the neural network.

The second interesting observation is that aggregate movie histogram features seem to be consistently correlated with the highest F1 scores. To

some extent, this makes sense as histograms that are mostly distributed with low aggregate actor ratings are probably unsuccessful, while histograms distributed with mostly high aggregate actor ratings are likely more successful.

Interestingly enough, movie-movie competition turned out to be a poor feature choice compared to the baseline features. I believe this is because competition is likely already modeled sufficiently well using the high, medium, and low month indicators from [3].

VII. FUTURE WORK

There are a variety of extensions that could be made to the existing model proposed in this paper. One alternative would be to model movie revenues as a continuous quantity rather than a discrete quantity. Another extension would be to introduce a temporal model for how movie genre and actor popularity change over time, which one might suspect would lead to more accurate revenue predictions. Finally, perhaps the biggest improvement that could be made would be to acquire more box office revenue data.

VIII. CONCLUSION

I developed a computational model for movie revenue prediction using a combination of features extracted from movie database metadata, actor-actor relationship graphs, actor-movie relationship graphs, and movie-movie relationship graphs. I demonstrated that by using features extracted from these actor-actor and actor-movie relationship graphs, we are able to create a more accurate model than using metadata features alone.

IX. REFERENCES

- [1] Simonoff, J. S. and Sparrow, I. R. Predicting movie grosses: Winners and losers, blockbusters and sleepers. In *Chance*, 2000.
- Joshi, M., Das, D., Gimpel, K., and Smith, N. A. Movie Reviews and Revenues: An Experiment in Text Regression. In Proceedings of the North American Chapter of the Association for Computational Linguistics Human Language Technologies Conference, 2010.
- [3] Sharda, R. and Delen, D. Predicting box-office success of motion pictures with neural networks. In *Expert Systems with Applications*, 2006.