Predicting Movie Box Office Gross

Jason van der Merwe
jasonvdm@stanford.edu

Bridge Eimon
beimon@stanford.edu

CS 229, CS 221, Stanford University
December 8, 2013

Abstract

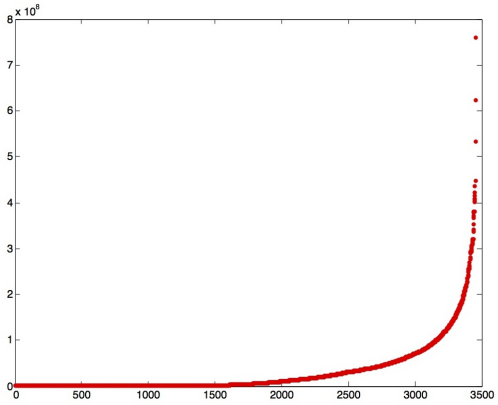
Predicting movie box office gross is an application of machine learning that has been tackled by very few people. Our experience with the project revealed several key insights into how well movies make money in the box office. We applied two different algorithms, linear regression and logistic regression generalized for multiple labels, and saw that logistic regression was the more accurate approach. To increase our accuracy, we applied k-means clustering on titles and complimentary features. We ran diagnostics to determine the fail points of our algorithms and where we should seek to improve them. Finally, we analyzed our results to draw insight into future work.

Introduction

Famous blockbuster movies of the past ten years have included epic films such as The Avengers, Lord of the Rings, Avatar and the Dark Knight series. All these movies made over $300 million in the box office. However, these films had massive budgets to match their returns, which required investors with deep pockets. Our project aims to assist investors in valuing the future return of a movie.

Data

To begin, we acquired a text file representation of every title in IMDBs database. Then, we queried the RottenTomatoes API with these titles (limiting our search to movies from 2000 and later due to a of lack information available for earlier movies). Each data entry included a movie id, the director, title, genres, rating, runtime, release date, actors, studio and RottenTomatoes url. Our final dataset contained 3456 data points, ranging from movies which grossed $181 to $761 million. After plotting the data points box office gross, we noticed a nearly exponential scale (figure 1). So, we decided to split our data labels into 5 logarithmic scale buckets to linearize the data.



Methodologies

Feature Extraction

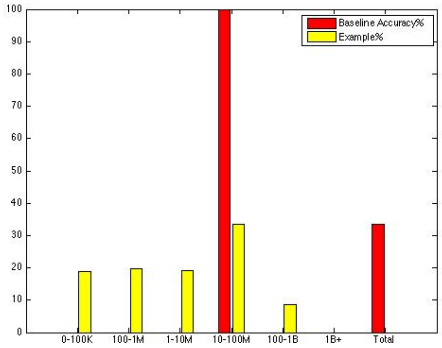
We are representing the feature vector as a sparse vector, which allows us to have 0 as the value for many features. The feature vector looks like this:

$$\begin{bmatrix} ACTOR : BradPitt = 1 \\ DIRECTOR : JamesCameron = 1 \\ RELEASEMONTH : 12 = 1 \\ ACTOR : OwenWilson = 0 \\ ACTOR + GENRE : \\ OwenWilson + Romance = 1 \\ . \\ . \\ . \end{bmatrix}$$

Where an element is 1 if that actor, director, release date etc. is present in the example and 0 if it is not. The weight vector allowed us to score individual elements of a feature vector for a proposed movie. We used single features, such as actor, director, release month, length of the movie and rating (PG-13 etc). We also implemented complementary features, such as director and actor, actor and genre. We also including a k-means score for titles, which is explained further in a later section.

Baseline System

Our baseline system split our data into 80% training data, and 20% testing data. We categorized box office income into buckets, by sections on a logarithmic scale (0-100k, 100k-1m, 1m-10m, 10m-100m, 100m-1b, 1b+). To classify, our baseline system returned the label that appeared most often in the training data, 10m-100m. Running this baseline system, we achieved 33.45% accuracy on the test data. Not only was our accuracy low, but obviously, our model didn't generalize or gain insight into the factors that affect movie box office income.



Linear Regression

Our fist learning algorithm was linear regression. We used least mean squares, specifically stochastic gradient descent, to learn the weight vectors. Our update rule was as follows:

$$\theta_j^i := \theta_j^i + \alpha(y^i - h_\theta(x))x_j^i$$

$$h_\theta(x) = \theta^T x$$

We ran LMS until convergence, meaning, until the difference in the weight vector between two iterations was very small. This took over 168,000 iterations because the feature space was so large (thousands of unique actors and directors). Unfortunately, our results with linear regression were poor. Only 30% of our predictions on our test data (data split 80/20 again) were within a predetermined margin of error (100%) from the actual result. After examining the predictions, we realized that the range of box office numbers is too high. For instance, the lowest grossing movie made $181, while Avatar made $761 million. This range made it extremely difficult to implement linear regression, so we decided to explore other options.

Logistic Regression Generalized for Multiple Labels

Since linear regression did not work as well as

desired, we decided to use the bucket system we identified in the baseline system. Each bucket represents an earnings range based on a logarithmic scale. Our classification process involved assigning a score to each label classification for a given movie. The label that has the highest score will be predicted to be the best match.

The logistic regression function is defined as:

$$i_{prediction} = \text{argmax}_i g(\theta_i^T x)$$

Where θ_i is the weight vector learned for the ith label, which we predict. $i_{prediction}$ is the score given to that label. To do this, we used a binary classifier for each one versus all label. In simple terms, we would calculate the score of a movie that, for example, was either in the bucket of \$1m-\$10m or not in that bucket. We repeated this for every bucket and return the highest score.

K-Means Clustering Score for Titles

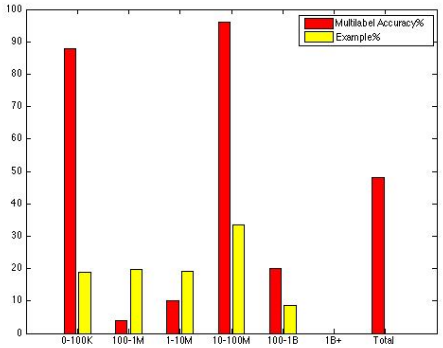
In order to include the movie title in the feature vector, we needed to give a movie title a score. This is one of the issues we ran into as we weren't sure how to assign a score to a title for a movie, but we wanted to include the title since the title of a movie does have an effect on the movie's success. To accomplish this, we have decided to utilize K-Means clustering. K-Means is a powerful unsupervized learning algorithm which clusters similar data based on inputted feature vectors. We are including in our feature vector the following features: length of the title, number of words, occurence of numbers, and individual substrings. K-Means outputs a cluster number which is the cluster that the title is assigned to. The perceptron will then learn which clusters do better, or if there is any significance at all. The K-Means algorithim is defined in the functions below:
Initialize Centroids $\mu_1, \mu_2, ..., \mu_k \in \mathbb{R}^n$

For each i set $c^i := argmin_j \parallel x^i - \mu_j \parallel^2$

For each j set $\mu_j := \frac{\sum_{i=1}^m \mathbb{I}\{c^i = j\} x^i}{\sum_{i=1}^m \mathbb{I}\{c^i = j\}}$

Results

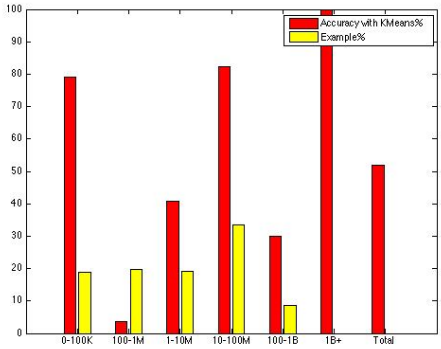
Logistic regression was the most effective algorithm. Ultimately, we achieved 52% accuracy. The percentage accuracy for each bucket can be seen in the figure including K-Means. Compared to random choosing of a bucket (6 buckets, so the percentage chance you guess right is 16.67%), our model is over three times more accurate. Our first iteration of linear regression achieved only 33% accuracy. The addition of complementary features increased accuracy to around 45%. Additionally, the elimination of single actors and directors inceased our accuracy by a couple points to 50% accuracy.



When we implemented K-Means clustering on the titles, we increased this accuracy to 52%.

We also implemented LOOCV (leave one out cross validation) and applied this technique to a smaller dataset of 500 samples. We ran LOOCV without K-Means and achieved 48% accuracy, which was impressive on a small data set. However, LOOCV took so long on

the small sample size that it was not feasible to run LOOCV on the full data set containing 3456 entries.



Analysis

Feature Extraction

The one issue we ran into was that our feature space was too large. Because of the abundance of actors and directors, our feature space had tens of thousands of entries. Not only did this hurt our accuracy but it also slowed our algorithms considerably. To account for this, we eliminated any actors or directors that only appeared in one movie. This made sense because these actors and directors, presumably, wouldn't be valuable in any other films because they had yet to prove their worth. This sped up our learning process considerably. This increased the speed of the algorithms dramatically, as well as increasing the accuracy. The increase in accuracy confirmed our suspicions that these directors and actors were weak indicators of movie box office success.

We decided to implement complementary features because it is common for actors to specialize in a genre, or for directors to always pick the same actors. These features had a dramatic impact on our accuracy.

Feature Weights Analysis

We wanted to see which features were weighted the heaviest, so we ran a script to sort the weight vectors for each label. What we found was very interesting. Below are the top five features for the label $100m-1b:

('STUDIO:Sony Pictures', 2459.0)
('ACTOR:Anne Hathaway', 2307.0)
('STUDIO:Warner Bros. Pictures/
Village Roadshow', 2237.0)
('DIRECTOR:Peter Jackson', 2029.0)
('STUDIO:Lions Gate', 1954.0)

Overwhelmingly, the studio made the biggest difference in every single label category. Another observation found was that all Bollywood movies were classified in the $1-10m range. This was evident by the fact that in the $1-10m range, the most weighted features were an Indian Bollywood production company and Indian actors. This was very intriguing. We also found that the horror film genre had a large impact in the $10-100m range. After some research, this seems to be because horror films have a cult following and are cheap to make, so they are made often.

The figure above shows the prediction accuracy for each bucket. Our model is very accurate for the $0-100k and $10-100m ranges. We believe this is the case for the $0-100k because the actors, directors and studios producing the indie films won't make much money because they are indie films. Plus, since they have low budgets, they'll very rarely include big name actors, so the model easily predicts correctly. Many box office movies fall into the $10-100m range. They may include one or two big name actors, but

never very many. This is the typical range for the average movie, so our model predicts this range easily. We have around a 45% accuracy rate with the $1-10m range, probably because of the presence of Bollywood films.

K-Means

The figures above also show the difference between the accuracy of the model with and without kmeans. K-Means increased the overall accuracy by 2%, but it also evened out the accuracies for individual labels. The more accurate labels became slightly less accurate, however, labels which were very inaccurate became more accurate. We believe that the presence of kmeans gave more weight to movies which were sequels (one of the kmeans features was the number of occurences of numbers in the title). Length was also a feature, which gave more weight to movies that may have been in a series, like Lord of the Rings.

Analysis of Incorrect Predictions

Yuvvraaj was a movie that we predicted incorrectly. Our model predicted the $1-10m label, presumably because this was a Bollywood film. The director was Subhash Ghai and the two main actors were Anil Kapoor and Zayed Khan. The movie actually made $0.6m and underperformed most Bollywood films. So although the prediction was incorrect, it makes sense that the prediction was for $1-10m since the film was from Bollywood.

XXX was another movie we predicted incorrectly. Our model predicted $10-100m, but the actual was $141m. While the movie included famous actors such as Samuel L. Jackson and Vin Diesel, Jackson was weighted heavier for movies in the $10-100m range, but Vin Diesel and Rob Cohen (the director) were rated higher for the blockbuster range ($100m-1b). However, the biggest difference was the studio, Columbia Pictures. They have a weight of 4053 for the $10-100m range, but only a 457 weight for the blockbuster range. Again, this supports our hypothesis that the studio makes the biggest difference in the monetary success of the film.

We also predicted one film, The Other Side Of Heaven, incorrectly by a huge margin. In the box office, it only made $4.4m, but we predicted that it would make $100-1B. At first, we were very confused by this result, however, Anne Hathaway starred in the film. Her weight feature for movies in the $1-10m range is negative (-2101), because she is very rarely in these films. But her weight for movies in the blockbuster range is extremely high (2307). The movie was also released in April, 2001, a couple months before the Princess Diaries, Hathaway's first breakout role. So at that point in her career, she wasn't as valuable. None of the other actors were in any other films, so they were not in the feature space. But since Anne Hathaway carries such weight (see the feature weights analysis above), the prediction was for the blockbuster range.

Future Work

We plan on continuing to iterate on this project. We recently learned of an easy way to get budget data and plan on including the budget of a film as an additional feature. We believe that this will increase the accuracy dramatically. The project was enjoyable and many of our friends enjoyed hearing about different predictions, so we plan on turning this into a website where people can input their dream films and see how much they would make. We believe that this project shares valuable insights into the factors which determine the box office success.