Prediction of Movie Box Office Performance

Qian Wang, Shuo Liu, TianRun Sun

Georgetown University

Abstract

This paper looks at quantifiable and easily obtainable and recognized factors that affect a movie's success in box office performance. The goal of this analysis is to test the relationship between multiple variables: budget, revenue, genre, cast, crew, runtime, language, etc. And we apply traditional statistical analysis methods and machine learning models to generate interesting patterns for predicting box office performance of movies using data collected from TMDB movie database and YouTube. We compare different models and find a good prediction model to predict Box Office performance. The prediction is based on decision factors derived from a historical movie database and amount of trailer review of YouTube. We label the prediction in three class, Hit, Neutral and Flop, based on ROI(revenue/budget). The performance of the final results can be validated by follow-up movies released in late 2017.

Keywords: Data mining; Statistical analysis; Machine learning; YouTube; TMDB

I. INTRODUCTION

Movies are an integral part of modern culture, generating over billions and billions of dollars each year and delivering rich, intricately crafted stories to a worldwide audience. The film industry not only serves as a deeply expressive artistic medium from approximately 38 billion USD in 2016 to about 50 billion

USD in 2020(Statista, 2016). Moreover, the US is one of the most highly ranked countries in making the most movies per year. Therefore, it is an interesting question that which type of movies are more likely to succeed in box office performance. It is essential to understand what features of movies affect moviegoers' pattern so that movie marketers are in a better position to align and adjust their strategies to capture the interest of potential moviegoers, for better box office performance of their films.

The topic of movies is of considerable interest in many industries. Research has been done to generate models for predicting revenue of movies. Back in 2013, Google have published a research paper on quantifying movie magic with Google search [1]. Basically it dived into data analysis on Google search volume, amount of paid clicks, the movie trailer engagement and come to the conclusion that the times of trailers searches on both Google and YouTube are leading indicators of Box Office success.

In our research, we focus on analyzing how different factors affect a movie's Box Office success including budget, revenue, runtime, genre, cast, crew, etc. Together, we have used 13 attributes to represent the assessment of movies derived from TMDB Movie Database. In social media aspect, some of these factors are either hard to quantify or difficult to obtain reliably, for example, Google search volume

of some words of a movie does not mean the actual trend of the moviegoers' interest in it. Instead, we obtain consistently and essential factors like YouTube trailer reviews since trailers of movies is one of the most important and effective advertising ways. All of these attribute values are applied to generate movie classes: Hit, Neutral, or Flop. We use a multiple linear regression model to analyze all the features and determine which features are more related to Box Office performance. On the other hand, we apply predictive models like Decision trees, kNN, SVM other methods to compare the accuracy of different of models and find a good prediction model.

In the following section, we collected movie data from 2005 to 2017, and the related YouTube trailers reviews of them. We will explain the data collection and the ways we clean in details. Section III presents the experiments performed. Results and discussion are given in Section IV.

II.DATA COLLECTION

Data sources

We gather data from basic features of movies from 2005 to 2017 on the US market.

We collect movie information from TMDb (*i.e.* " The Movie Database"), a community built movie database opens for use with an API key. TMDb is a good movie database but due to its non-profit and community-contributed feature, the information of it may contain some errors. The information we need on this database mainly contain two part: basic information of movies, and credits of movies, which are stored separately. By using the discover module, we can easily find out all movies in the US released between 2005 and 2017. In the data collection part, we use script movieInfo.py to collect the information, and store our results in db.json. There are 52973 items in db.json.

We try to get data from social media in order to find relationship between social media and movies' assessments. Youtube is a good social media. By finding the number of views of each movies' trailers can assess the movie. The following steps shows how we getting data from youtube. Step one is to get the list of movie's name. It is easily to find by traversing the json file from TMDB. The second step is to clean the name of each movies. Many movie names are not in English and it cannot be read by python. I change the unicode of those name who is not in English so that the DNS can understand the url. Step three. After combined the URL, a request is sent to the youtube server. Using regular expression to find the first 20 videos related to the certain movies' trailer. Comparing all of them and find the trailer with the largest views. The return number is set as the number of views. The last step is to loop the previous function, find all trailers' views and save the data into csv file. The Fig 1 shows the distribution of views for all movies.

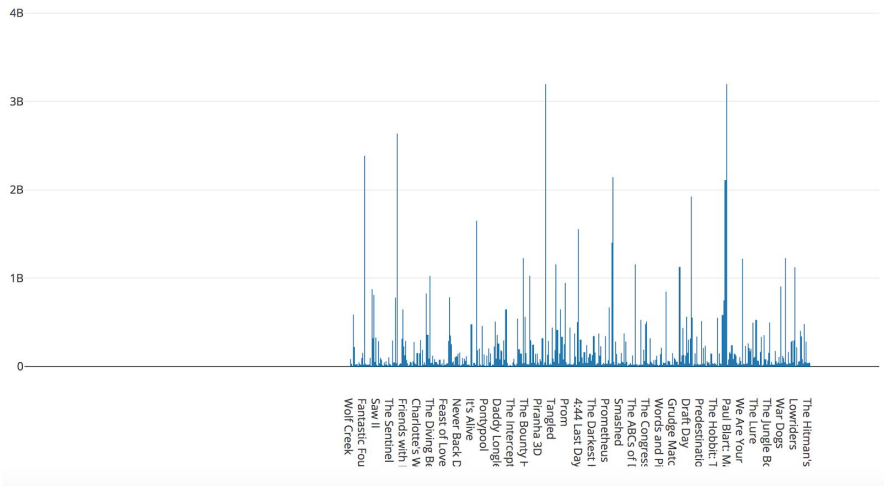


Fig 1

Data cleaning

In cleanliness part (cleanliness.py), we make some standards for movie attributes specified in the following table:

Table I. Movie Data

Attributes	Standard	Counter of bad items
budget	Should be a positive number, but zero (for no information) is acceptable	errBudget
genres	Should have at least one genre	emptyGenres

homepage	Should have one as string, or set to None for no information	errHomepage
production_companies	Should have at least one	emptyCompany
production_countries	Should have at least one	emptyCountry
revenue	Should be a positive number	errRevenue
runtime	Should be a number more than 30 Should not be blank	tooShort emptyRuntime
spoken_language	Should have at least one	emptySLang
title	Should not be blank	emptyTitle
cast	Should be no less than average amount of cast information divided by 4[1]	fewCast
crew	Should be no less than average amount of cast information divided by 8	fewCrew

If the counter is large, we consider the attribute as bad. We firstly removed all items that have a revenue equal to 0, for the revenue is a key information we need and we cannot give it a value arbitrarily. Then we test the remaining 3580 items with these counters mentioned above and our results are:

Table II. Error Statistics

[1]

Counters	Value (smaller is better)
errBudget	0
errHomepage	1573
errRevenue	1
emptyCountry	103
emptyCompnay	259
emptyGenres	27
emptyRuntime	1
tooShort	41
emptySLang	67
emptyTitle	0
tooFewCast	318
TooFewCrew	537

We think the quality of the database (by our definition) is not too bad, for only about one-six of the items have bad attributes and need to be fixed.

In cleaning part (cleaning.py), we follow the standards in cleanliness part. We mark empty homepage as None instead of ("empty string"), remove items with minus revenue, with no runtime or shorter than 30 mins, with no title and with too few credits information, and consider the original language as one of the

spoken language of a movie if its spoken language is missing. By doing these, we remove some "irreparable" items and reduce the number of items to 2917. We store them in movieDbClean.json for future use. We also created a movieList.csv with movie names and id's in it.

Data description

We define revenue/budget as ROI, representing the box office performance of a movie. Also, for this part of our project, we quantify cast and crew as the number of casts and crews relatively, to show the size of a movie. We choose the first genre/country/company id from the list of every movie, which represent the primary genre/country/company of a movie.

The attribute YouTube View is the maximum number of views of each movie's most-clicked trailer. This is a new attribute we just got permission from the professor and collected.

Table III. Cleaned Data Description

Attributes	Means	Median	Standard deviation
budget	42626950.0	25000000	5.011864e+07
revenue	100085900.0	27206120	1.941830e+08
ROI(revenue/budget)	612.51	1.97	2.244447e+04
runtime	107.87	105	1.892636e+01
cast	25.7	19	2.148605e+01
crew	32.1	21	3.193185e+01
YouTube View	41848100	5995642	1.788084e+08

III. RESEARCH METHODOLOGY

In the last Section, we have collected the basic movie information from TMDb and cleaned it in specific rules. Also we collected the YouTube Views related to the cleaned movies. In this section, we will first apply multiple regression models to dive into the correlations between different features and ROI to find the most essential features. What's more, we will use machine learning models to generate interesting patterns to predict Box Office performance. And we compare different models to find a good prediction model.

Correlation analysis:

In order to find the correlations between various features and ROI In this part, we analyze the correlation method to those features. The reason we plan to use correlation method is that this is a simple way to find a roughly relationship between two features. Since we have over 16 features in the data set and some of them is helpful to the next analyzing but there are some data that has no effect on the following project. Using correlations, we may find weather or not this feature is helpful to the project. And this analysis will make an influence on feature selection in the next parts.

To make correlations meaningly, we plan to find a feature that has a positive influence on the result. When using other features to correlate this features, we can make a prediction for whether those features are useful. In many papers of analyze movies, there is a parameter called ROI, calculated as "revenue/budget". Using ROI to make correlation with other features, we can get a scatter plot and a regression line. The result shows like Fig 2.

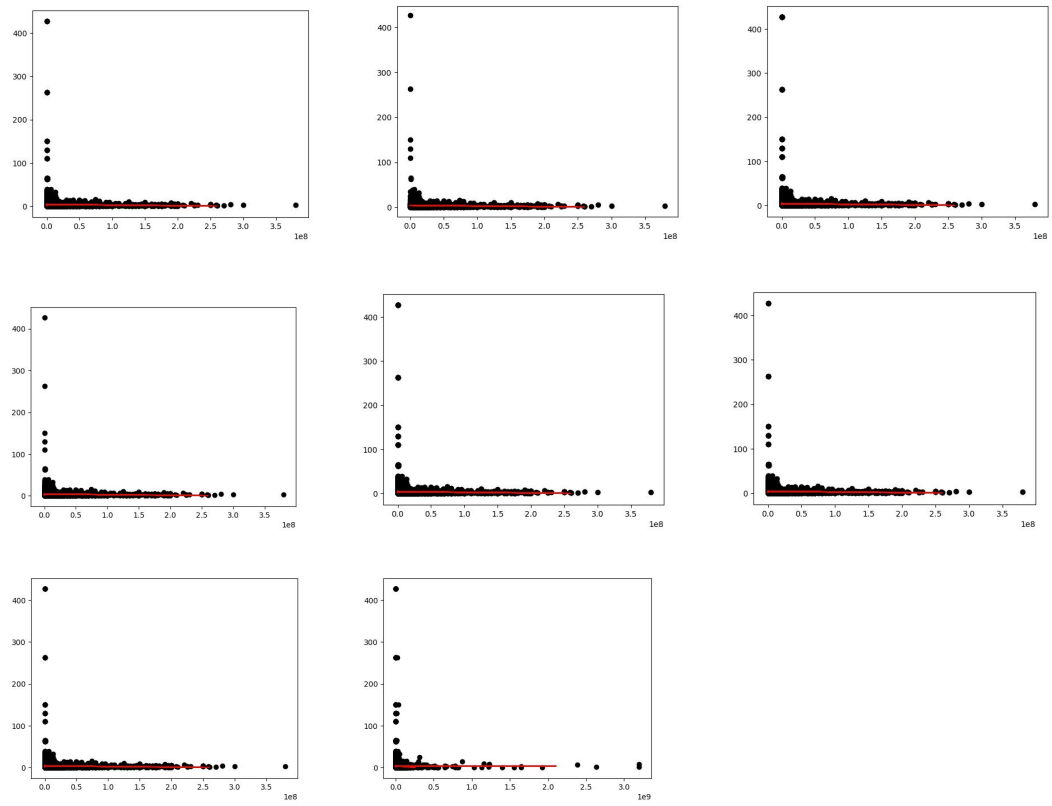


Fig 2

The features are budget, cast_number, company, country, crew, genre, runtime and youtube views.

From the graph, there is only a regression lines with a very small slope. We cannot get much more information from this chart. In order to get a better a result, a linear regression is used to analysis those features.

Regression methods

In this part we try to learn the ROI score of movies with regression methods. We used different regressor from *sklearn* package on our datasets: k-nearest neighbor (*KNN*), decision tree regressor (*CART*), linear support vector machine (*LinearSVR*), three linear regressor: without penalization (*LR*), with l1 penalization (*Lasso*), and with l2 penalization (*Ridge*), two random regressor: random forest (*RF*) and extra tree (*ExtraTree*), and two boosting methods AdaBoost and gradient boosting (*GB*). We used 10-fold cross validation for estimating general performance of methods.

We use r-square to evaluate performance of those methods. Results are shown in table 1. We see that extra tree and CART performs better than the other methods.

Table 1: Average r-square scores on different methods

Methods	KNN	CART	ExtraTree	LR	Lasso	Ridge
r2	-0.642943	**-4.587444**	**-5.920193**	-0.006885	-0.011470	-0.006802
Methods	RF	SVM	AdaBoost	GB		
r2	-0.738448	-0.053004	-2.657924	-1.299513		

Classification methods

In this part, we divided ROI into three groups: [0,3], (3,5], and (5,+inf), corresponding to *bad*, *good*, and *hit* movies. The reason of choosing that division is the revenue comes from box. Beside budget, the movie company also need to pay money to the cinema, advertisement and performing activities. Those expenses are not included in the budget. Commonly speaking, if the movie company wants to make a profit to a movie, the revenue should be 3 times larger than budget, that is to say, ROI = 3 is the judgement of whether or not a movie makes a profit or loss money. If revenue/budget larger than 3, the company can make a profit. Besides, if ROI > 5, which means the movie has a better profits, and that means this movie attracts many people and have a good assessment. So, if revenue/budget larger than 5, this movie should be a hit movie.

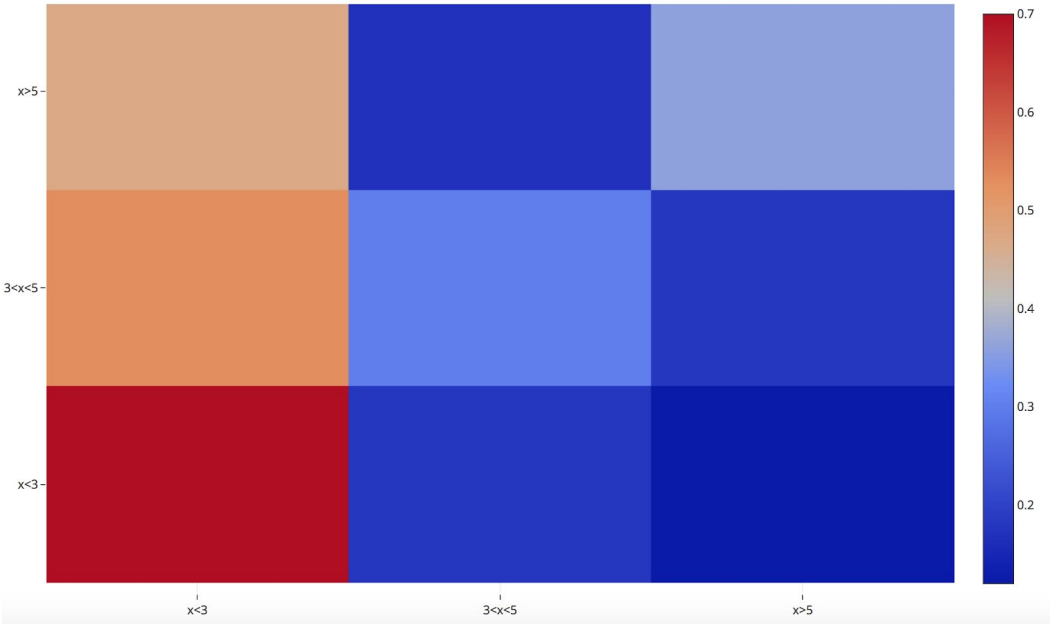
We tried several machine learning methods on the datasets we collected using *sklearn* classifiers: k-nearest neighbor (*KNN*), decision tree classifier (*CART*), linear support vector machine (*LinearSVC*), two random classifiers: random forest (*RF*) and extra tree (*ExtraTree*), and two boosting methods: AdaBoost and gradient boosting (*GB*). These classifiers are run on our movie dataset. We used 10-fold cross validation for estimating general performance of methods.

We use F1-score to evaluate performance of our methods. The results are shown in Table 2. We also generated a confusion matrix for CART, which reaches the highest macro F1-score among all tests.

Table 2: Average F1 scores on different methods

Methods	KNN	CART	ExtraTree	RF	SVM	AdaBoost	GB
F1(micro)	0.640012	0.510068	0.520928	0.647627	0.517694	0.655224	**0.669916**
F1(macro)	0.348896	0.375062	0.372000	0.378529	0.272562	0.382565	**0.402035**

Fig 3



From Table 2 we can see boosting methods outperform other classifiers. But after we took a closer look on the details of their F1-scores, we found that CART gave better F1-scores on all three classes, while GB and AdaBoost gave such high F1-score on one class that even macro F1-score was pulled up by that single score.

The figure of confusion matrix also shows that there is a tendency that our classifier would like to underestimate the performance of a movie (see the darker cells on upper-left corner). Our guess is that, aside from the unbalanced training set, movies that turned out to have bad box office performance might have more common characteristics than those whose performance were good. Also, from investor's point

of view, underestimating a movie would not lead to greater loss of profit than overestimating one, so the tendency toward upper-left corner would be better than a lower-right tendency in practice.

Importance of features

Among the methods we talked above, we choose Lasso regression and CART to interpret contributions or levels of importance of different features in predictions. We used our whole dataset to train the regressor and classifier.

In the Lasso regressor, '*budget*', '*crew*' and '*youtubeView*' (in this order) contribute the most, while '*runtime*' and '*cast*' contribute very less.

Table 3: Absolute value of Lasso score of each numeric features

Feature	*runtime*	*cast* *i.e.* # of cast	*crew* *i.e.* # of crew	*youtubeView*	*budget*
Lasso score	~0.000	~0.000	1.9649	0.6139	38.2498

In the decision tree classifier (*CART*), we see that the root node is 'youtubeView<=7408751.5', and two nodes on the second level are 'budget<=15200000' (for True on root node) and 'cast<=42.5' (for False), Which indicates that these three features contributes a lot when the CART model classifies new entries.

IV. CONCLUSION

We try to find correlation between different features and ROI and a good prediction model to predict Box Office performance. In our experiment, we collected data from TMDb and YouTube. And the result turns out the correlations are not obvious since it shows a regression line with small slope. Then we apply different regression methods like KNN, CART, ExtraTree, LR, Lasso and Ridge. And the r-squares show that Extra Tree and CART performs better than other methods. Finally, we apply different machine learning methods on the dataset with different classifiers and generate a confusion matrix for CART since it has highest F1-macro score.

We come to a conclusion that CART is the better prediction model for the dataset. Moreover, the confusion matrix shows that all classifiers tend to underestimate the Box Office performance of a movie possibly because bad movies have more common characteristics and our training data is unbalanced.

Overall in our research, we have some limitations. We have noisy dataset and during the cleaning process, we reduce too much data. On the other hand, we don't have many quantifying and effective features and our accuracy does not perform very well in our machine learning methods implement.