Predicting movie earnings

(Coursera Capstone Project)

Tomasz Porzycki
24/10/2019

Summary
The main question of this project is: **Can the earnings of a movie be predicted?**
Cinematography is the art of motion-picture photography. It is an artistic work used for entertainment or communication of important ideas. On the other hand, we often use the term film industry. And the term "industry" is more related with business and money making. As a result, big film studios are more interested in the second aspect – making money. Can they predict, if a movie will be successful and make profit in advance of releasing it in cinemas?

1. Introduction

1.1. Problem

Film producers invest huge amounts of money in film making with the purpose to make profit. Now good reception of the film depends on audience and its taste. However the film studios try to secure the maximum profit by hiring popular actors, well-known directors, selecting interesting scripts. But can they predict the film earnings in advance?

1.2. Interest

The profit of a film is not only in the interest of the producer.
Actors negotiate their salary in percentage of the film total earning, so naturally they want to ensure and maximize their profit.
Banks or other companies lend money for making the film, so they want to guarantee money back.

2. Approach

2.1. Approach

- Create a database of 100 most successful movies of recent 10 years (2010- Sep 2019) that can be used for model training
- Identify available features which can potentially impact success of the title
- Use opening earnings as the target feature to be predicted
- Train the model based on the above data
- Predict earning for upcoming titles (Oct-Dec 2019)

2.2. **Data requirements**

What information about a film is available before its premiere that can be used to predict its success?

- Director
- Cast (actors)
- Genre
- Plot
- Duration
- Release date
- Producer
- Budget

These facts are well know in advance therefore can be used for prediction.

2.3. **Data source**

- **imdb.com** - is a popular portal of information related to films – including cast, production crew and personal biographies, plot summaries, trivia, fan and critical reviews, and ratings

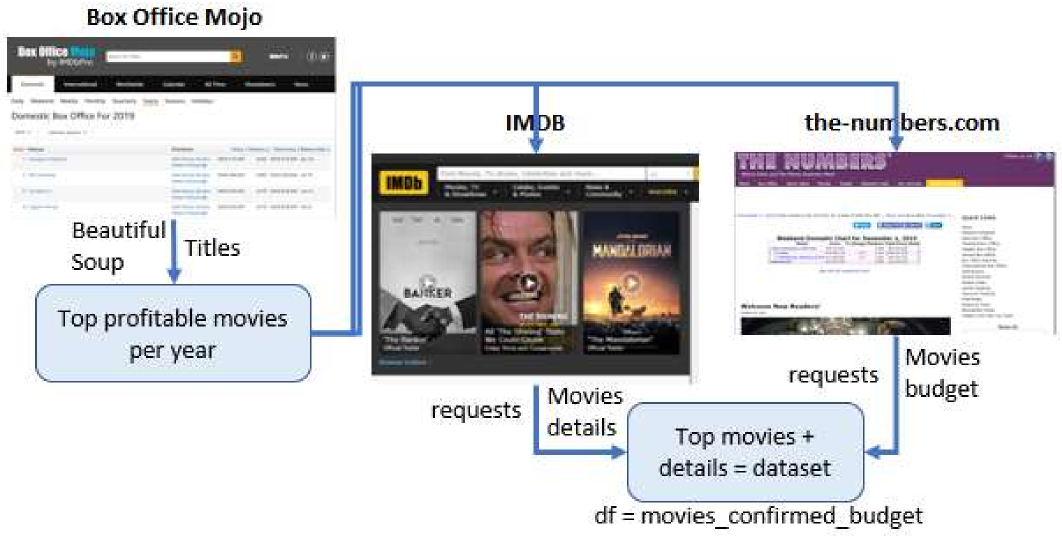
Use omd

- **boxofficemojo.com** – list most profitable titles in each year
- **the-numbers.com** – contains information about the movie budgets and earnings

2.4. Data collection

Using web scrapping (BeautifulSoup) obtain list of most successful film per year -> 900 titles

Download features using OMDB API and store in a dataframe.



2.5. Data understanding / exploratory analysis

The result of data collection is a dataframe containing below raw information.

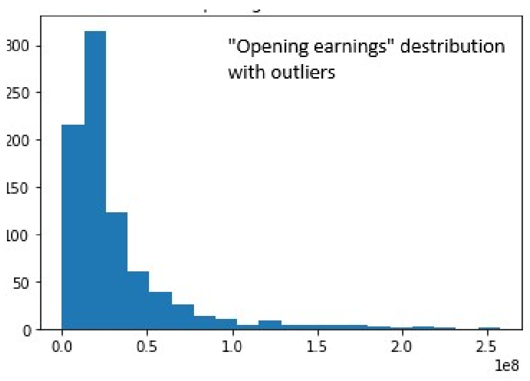
- There are mainly categorical features
- The target value (Opening) is highly skewed.

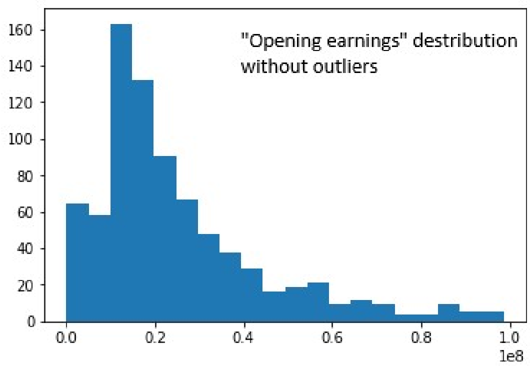
In numbers:
900 – total rows
58 – rows have missing data. Remove.

Data	Format	Note	Use for modelling
Title	String	Film title used only for reference in data collection	No – not relevant
Rank	String	Film box office position in particular year	No – available only after release
Studio	String	Film company which produced the picture	No - duplicate with Production column
Total	String -> convert to float	Total earnings	No – use opening earnings for quicker results
Total _theathres	Number	Number of theatres the film was screened	No - available only after release
Opening	String -> convert to float	Earnings in the opening period	Yes – use as target
Opening _theathres	Number	Number of theatres the film was screened	No - available only after release
Actors	String – need to separate	Names of starring actors	Yes
Awards	String	Gained awards	No - available only af ter release
Country	String - need to separate	Country in which film was produced	Yes
Director	String - need to separate	Name of the director(s)	Yes
Genre	String	Category	Yes
Language	String	Language	Yes
Plot	String	Story	Yes
Production	String	Film studio	Yes
Rated	String	Age category	Yes
Ratings	String	Rating by audience	No - available only after release
Released	String – dd-mmm-yy	Date of the premiere	Yes – keep month and year
Runtime	String – transform to number	Film length in minutes	Yes
Writer	String	Author of the script	Yes
Budget	String – transform to number	Film budget in USD	Yes

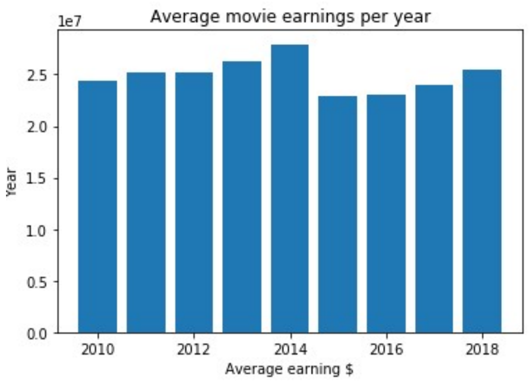
2.6. Data Preparation

Opening – target
Transformation: remove $ sign and convert from string to number
The target value data is highly skewed; therefore, the outliers are removed.

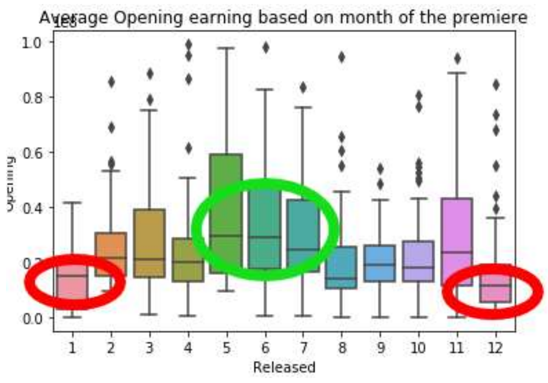




Year
Transformation: none
Logical expectation was the movie earning grow each year, however average earning per year graph does not confirm this, therefore remove this feature.

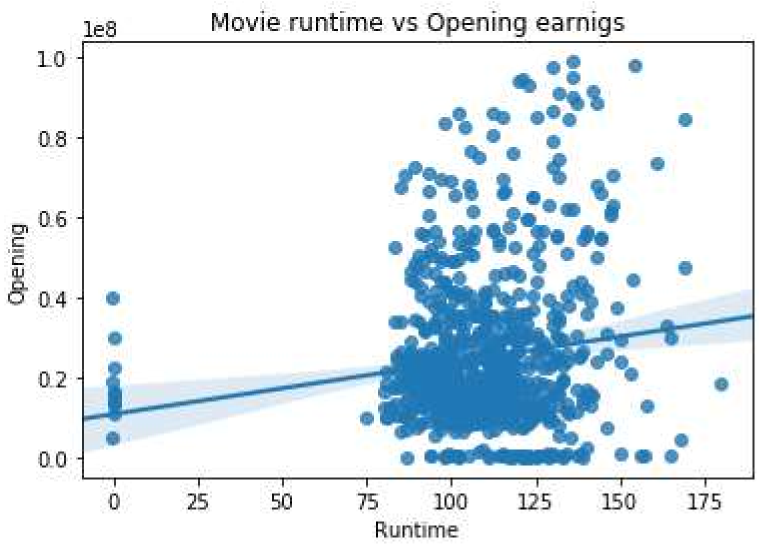


Released
Transformation: from date in string format dd-mm-yy extract only the month
Represents the month of the premiere. The average earnings are highest for months of May-July, while January, December report lowest profits. Keep the feature.



Runtime
Transformation: remove min and convert from string to number
There is no correlation between the movie duration and earnings – remove from

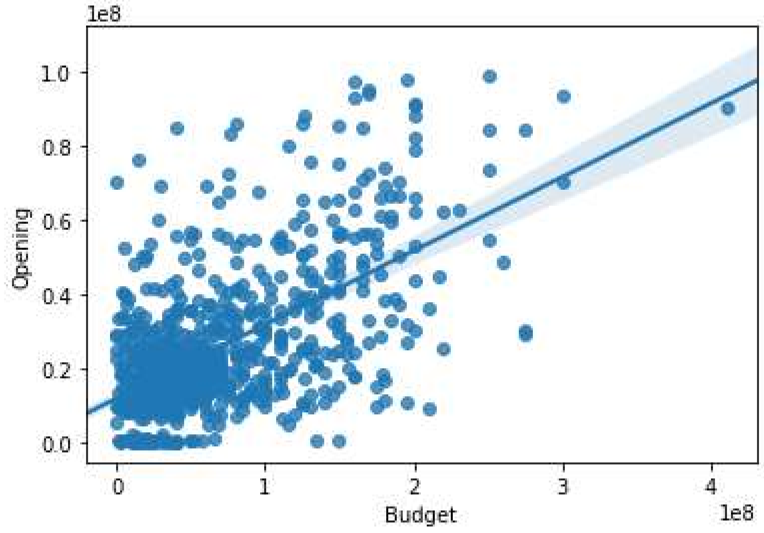
model.



Budget

Transformation: remove $ sign and convert from string to number

The correlation between movie budget and earnings is not strong. However, keep this feature.

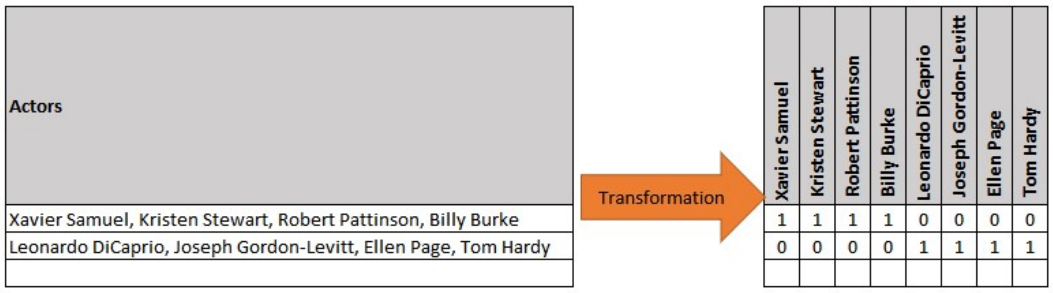


Categorical features: Actors, Country, Director, Genre, Language, Rated, Writer, Production

Transformation:

- clean data, remove comments in brackets, align nomenclature

- split the string into separate items, example:



Actors
Xavier Samuel, Kristen Stewart, Robert Pattinson, Billy Burke
Leonardo DiCaprio, Joseph Gordon-Levitt, Ellen Page, Tom Hardy

Transformation

Xavier Samuel	Kristen Stewart	Robert Pattinson	Billy Burke	Leonardo DiCaprio	Joseph Gordon-Levitt	Ellen Page	Tom Hardy
1	1	1	1	0	0	0	0
0	0	0	0	1	1	1	1

The result are high dimensional sparse matrixes for each categorical feature.

Categorical feature	**Number of items / new features**
Actors	1778
Country	43
Director	539
Genre	22
Language	71
Writer	1478
Rated	5

This is the biggest challenge of this project. The dimensionality of the data increases significantly and the final matrix is very sparse, which can impact the accuracy of the machine learning model. It also makes it difficult to evaluate feature importance to the model. Anyway, this is a method to encode categorical data.

Plot
Transformation: translate the short story of the movie into a bag of words using nltk library – tokenize
The result is a high dimensionality sparse matrix similar to the one for categorical features. Total number of columns = 9814 (which is number of unique words in all movie plots).
To reduce the dimensionality – only the words, which frequency is > =15 is selected, which results in matrix of 2733 columns.

3. Modelling

3.1. Preparation

All the data is combined in one matrix of:
- 6443 columns – features
- 802 rows -movies

Data set is split in training and testing set and scaled using MinMaxScaler

3.2. Training the model

Several machine learning algorithms are selected to test best fitting model as I could not predict the best model for high dimensionality sparse matrix.
The models are evaluated based on R squared coefficient .

Summary table:

Algorithm	**R^2 train data**	**R^2 test data**	**Comment**
Linear regression	0.820	-1.840	Linear model is not appropriate for the dataset
Decision Tree	0.238	0.185	The result is very poor

Neural Network	0.997	0.217	Overfitting
Gradient Boosting	0.671	0.435	Overfitting, however train/test result is closer
Support Vector Machine	1.0	0.0	SVM does not provide any significant result

3.3 Evaluation

Overfitting

None of the algorithms performed well. The possible reason maybe the dataset in form of sparse matrix.
Only 567 actors (out of 1778 total) played in more than 1 movie.
Only 229 actors played in more than 2 movies.
Only 205 directors (out of 539) made more than 1 movie.
Only 455 writers (out of 1478) made more than 1 movie.
As a result some algorithms (Linear, Neural Network, Gradient Boosting) perform well on train data, but poorly on test data.

Tuning

Best performing algorithms are Gradient Boosting, Neural Network. Possibly with some parameter tuning the result could still be better. However, I consider tuning of those parameters outside the scope of this project.

2nd loop

Considering the input data is sparse matrix I eliminate some categorical features, which do not provide much information and re-run the best performing algorithm – Gradient Boosting.
Features to be removed: Actors, Country, Director, Writer, Language

To dataset is reduced from 6443 to 2761 columns.

The result is

Algorithm	**R^2 train data**	**R^2 test data**	**Comment**
Gradient Boosting 1st loop	0.671	0.435	
Gradient Boosting 2nd loop	0.691	0.465	The result is just slightly better

4. Final conclusion

The result is disappointing. For every algorithm the model was overfitting and the

evaluation on test data was not good enough to deploy the model for business use. Definitely there is room for improvement:
- create bigger training data set
- find a way to evaluate importance of each feature
- create hyper feature to better describe the data
- test other, more sophisticated algorithms
- fine tune model parameters

Just the give an idea of the accuracy of the model (Gradient Boosting), the table below show predicted opening earnings vs prediction for three movies.

Movie	**Actual earnings**	**Predicted earnings**	**Delta**
1	24’830’443	15’734’224	-36.6%
2	21’052’227	27’278’085	29.6%
3	10’609’795	24’447’134	130.4%