# Progress Report – Movie Box Office Revenue Prediction

Yunbo (Robert) Chen [ID: 1871493]

University of Washington Bothell, Bothell WA, USA

**Abstract.** The project is trying to predict the revenue of a movie before it is in the theater and compare the importance of various features of a movie. The project uses over 4000 movies throughout years and combined many methods proposed by previous researchers. We used history gross values to measure actors and directors' "star values" during data pre-processing. Both statistical machine learning methods, such as support vector machine, random forest, and logistic regressions, and neural network approach are implemented for result comparison. After the experiment on the neural network approach, I was able to achieve an accuracy almost as high as previous researchers who used not as many, or as recent, movie dataset as I did. There are more ways improve the accuracy of the prediction, such as using first-week box office revenue as labels instead of long-term revenue.

**Keywords:** Machine Learning, Classification, Movie, Box Office, Artificial Neural Network, Predict.

## 1 Introduction

### 1.1 Backgrounds

More and more people are going into the theatre and watch movies more often. The film industry has never been more flourishing. Therefore, it is beneficial for a professional movie producer to have a sense on the box office revenue of a coming-soon movie and know what would help their movies become more popular in theaters.

Many possible factors that could affect the overall box office revenue of a movie. Many people are going to see a movie with actors that they like, or a director that they believe could bring a good movie and make their time worth in the theater. Some people will look at the production companies that produced the movie and determine the budget and the production quality of the movie. This project is trying to predict the revenue of a movie before it is in the theater. All the factors that described above will be considered as possible factors in this project.

Previously without much shared information on the Internet, the most difficult part for researchers would be to gather enough movie information for data processing. Nowadays many companies are willing to share a fraction of their data on data analysis competition websites in order to invite competent people to solve their problems. The dataset I used was also posted on such websites. It is called "The Movie Dataset" on Kaggle.com [1] and contains information and features, such as actors, release years, keywords, IMDB links, et cetera, of over 40,000 movies. It is a very comprehensive dataset that contains not only basic information on the movies, but also reviews, poster links, and other links to multimedia materials that could be potentially used for sentiment analysis or image classification as well. For the use of this project, I only picked ones that I think are mostly related to the prediction of movie revenue before the movie is on.

Movie box office forecast is never an easy problem, because it contains many non-linear, even unpredictable factors such as society environment, news on the actors/directors/producers, economy and historical events, and other movies in the same time slots.

### 1.2 Related Works

Many researchers tackled the problem before. Dr. Sharda and Delen [3] invented a 9-class classification method to classify the revenue number and directed the problem into a classification problem. They implemented a backpropagation neural network approach to solve the problem [3] and achieved a better accuracy (36.9%) than other machine learning methods. The accuracy is not very high but reasonable given the nature of the problem. There is also a one-away accuracy, which calculates the accuracy of prediction that predicted label is within 1 class of the ground truth label. Also, they got a one-away accuracy of 75.2%. Note that they used only 834 movies dating from 1998 to 2002.

Pertaining to Dr. Sharda and Delen's research [3], I have also found another data analyst using the same labeling to predict movie box office revenue on Kaggle. Lin [4] used 3154 movies from TMDB database. He implemented multiple statistical machine learning algorithms to solve the problem, and the best accuracy he got is 32.96% using Kernel SVM and one-away accuracy of 70.18%. It is a good result since the number of movies is much higher than that of Dr. Sharda and Delen's.

Last but not least, Dr. Zhang et al. [5] used a 6-class labeling method to predict movie box office revenue class. Their dataset contains 241 Chinese movies between 2005 and 2006, which is not large but concentrated. Zhang et al. used many preset weight values in their data-preprocessing. For example, they have a different weight set for different genres of movies, different months, weeks, and festivals. Their prudence, rigor, and efforts in putting in and retrieving those weights are incredible. They also implemented a backpropagation neural network for the approach. Eventually, with 6-class labeling, they achieve an accuracy of 47.9% and one-away accuracy of 82.9%.

## 2 Method and Approach

### 2.1 Refining the Problem

Before actually processing the data, I have done literature review on the previous works related to movie revenue prediction. I found that the accuracy of the regression is very low comparing to other regression problems. In Nikhil Apte's paper [2], their average testing error is around 110%-130%, which means the prediction is pretty far away from the ground truth most of the time. The highest accuracy they got for testing error is 20%, in which they transform the problem into a clustering problem. In this case, predicting with regression will not yield any practical information on the result, because the difference in box office revenue is huge between different movies, a 10% difference in a 1-billion-dollar movie will be a 50% error of a 20-million-dollar movie. The result will not be helpful for movie producers' discretion.

Therefore, I decided to change the problem into a classification problem. After reading through Dr. Ramesh Sharda's paper [3], I decided to use his labeling on the box office revenue.

**Table 1.** Classification on box office revenue

| Class | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| Range (millon) | <1 | 1-10 | 10-20 | 20-40 | 40-65 | 65-100 | 100-150 | 150-200 | 200< |

Then I made a histogram on my dataset and checked the distribution of movies in each class. It looks fair since more movies should be in the preceding classes (from 1 million to 10 million is a big cross in film industry). There are many movies falling into class 1, 2, because many movies with small productions, or independent movies, are not preferred, or acknowledged by the public, therefore they cannot hit the public market or get higher row-piece rate. Moreover, since the industry is booming, there are more blockbuster movies hitting class 9, with gross revenue over 200 million dollars.

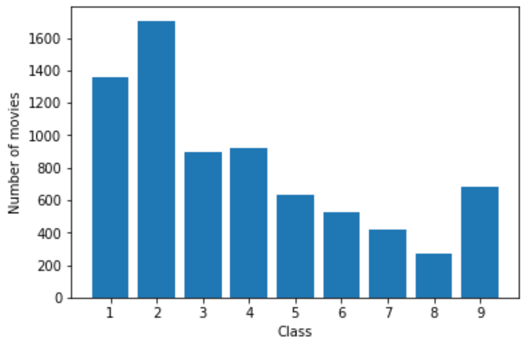

**Fig. 1.** Number of movies in each class in The Movie Database | Kaggle

## 2.2 Pre-processing Data

### *2.2.1 Data Cleaning*

In pre-processing my dataset, I first found out that the dataset consists of many old movies with no revenue or budget information. Thus, I removed the rows with 0 revenue or 0 budget, and the number of rows decreased dramatically to 5381. Then to make the prediction more reflective to current industry and avoid getting "dragged down" by old movies, I filtered out the movies before 1990. I also filtered out the movies not in English, as we want the dataset to be as concentrated as possible. Then the number of entries goes down to 4321. In the remaining 4321 movies, there are some entries with invalid production companies, actors, or director names. At the end of the data cleaning, there are 3661 valid, recent movie entries. Though over 90 percent of the movies entries are removed, the amount of movies pending for analysis is still higher than other research. I'm holding 20% of the data as testing data and using 5-fold cross-validation technique.

Since there are many features that are unrelated to my problem, such as movie reviews, keywords, poster links. I need to decide on which features I keep. Since I defined my problem as pre-release box office revenue prediction, I removed the IMDB rating and popularity columns, as they are calculated after release.

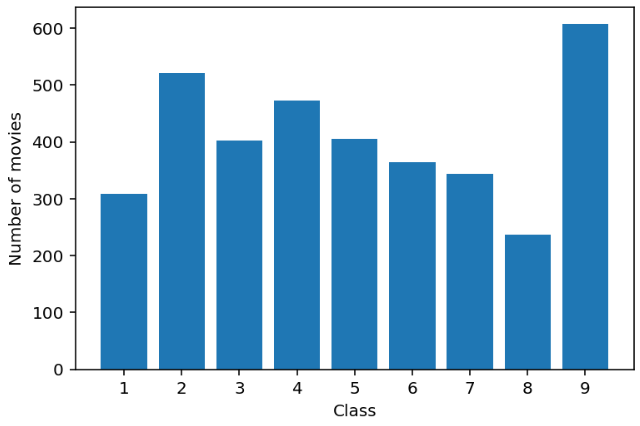

**Fig. 2.** Label class distribution after data cleaning.

### *2.2.2 Categorical Values*

While pre-processing the data, the problem of representing actor and director features remains unsolved. Since each movie contains a director and a list of actors, we need to find a way to let the model comprehend the names and learn which actor would weigh more in the result. Simply one-hot encoding all the actors/directors names is not a good approach since it wastes a lot of computational space and did not represent the "star values" well.

While looking online for solutions, I found a way to translate the actor nominal values into quantitative representations in Jiayong Lin's paper [4]. Lin uses 6 new features to represent actors and director values: the gross revenue, IMDB score and the number of movies the actor or director participated before the movie. In my problem, I'm calculating only the history gross revenue in my data, since I just need to find a way to represent the actor/director's influence before the movie. If the actor did not cast a movie before this movie, the value will be 1.0. For the representation of production companies, I also used history gross revenue, but the average of it, to avoid the fact that more large production companies would make the number too large compared to other movies with small production companies.

**Table 2.** New Features for Star Values

| Feature Name | Description |
|---|---|
| DirectorHistoryGross | Gross revenue of the director prior to this movie |
| ActorHistoryAverage_Gross | Gross revenue of actors in the movie before this movie |
| ProductionCompanyAverage_Gross | Average gross revenue of the production companies in the movie before this movie |

Another categorical feature to be translated is the genre feature. To better represent the genre feature, I used the common method similar to one-hot encoding: adding binary features into the dataset, with each of them representing whether the movie is in that genre or not. For example, a movie with genre IDs of [16, 35, 10751] will have genre_15, genre_35 and genre_10751 to be 1, and other genre ID features to be 0. In my dataset, there are in total 21 unique genres, therefore 21 binary genre features added into the dataset.

After all the data pre-processing, here are all the attributes that will be sent to the machine learning model:

- **adult**: whether the movie is adult-only or not
- **budget**: budget of the movie
- **release_year**: year of release
- **runtime**: runtime of the movie
- **director_history_gross**: history gross revenue of the director before the movie
- **actor_history_gross**: history gross revenue of the actor before the movie
- **production_companies_gross**: average history gross revenue of the production companies before the movie
- **genre_id** x 21: 21 binary features representing the genre of the movie
- **label**: label of the movie according to its revenue, ground truth.

The binary features and the other features are normalized separately and then concatenated together because they need different ways of normalization. Binary features needed to be normalized with row axis, while the history gross values needed gaussian normalization to decrease the drastic difference in gross revenue numbers.

## 2.3 Models

For the statistical machine learning approaches, it is relatively easy to set up the model. I basically used scikit-learn library [6] to set up the model and tune the parameters.

For the setup of artificial neural network. I am initially inspired by the backpropagation network designed by Dr. Zhang et al. [5]. They used two hidden layers, with 30 units in the first layer and 10 units in the second layer. Since they have 6 classes (outputs), and I have 9, plus the huge difference in dataset size, I redesigned the network with 60 neurons in the first layer and 30 neurons in the second layer. I then added two dropout layers in between the layers because during the experiment I noticed an overfitting issue, where the training loss goes to 0 and validation loss goes up. For the activation functions, I chose ReLU in both layers and softmax for the output layer.

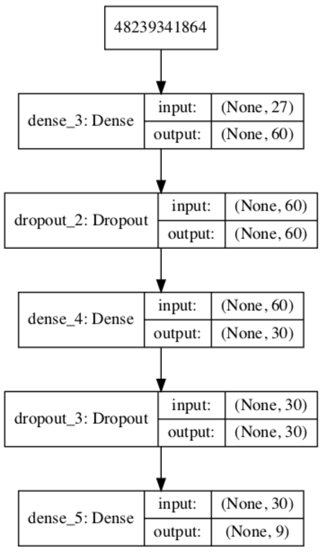

**Fig. 3.** Artificial neural network model structure.

## 3 Experiment Details and Result

### 3.1 Results of Statistical Machine Learning Approaches

**Table 3.** Accuracies of Statistical Machine Learning Approaches

| | **SVM** | **Logistic Regression** | **Random Forest** |
|---|---|---|---|
| **Bingo** | 0.342 | 0.332 | 0.343 |
| **One-Away** | 0.549 | 0.585 | 0.601 |

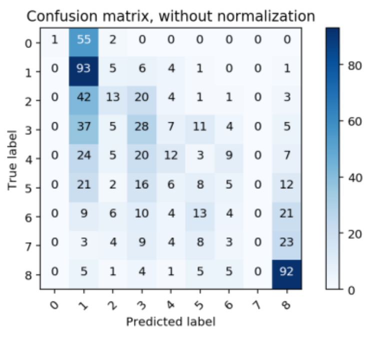

**Fig. 4.** Confusion Matrix of using Support Vector Machine (C=0.8, kernel=’rbf’, gamma=’scale’)

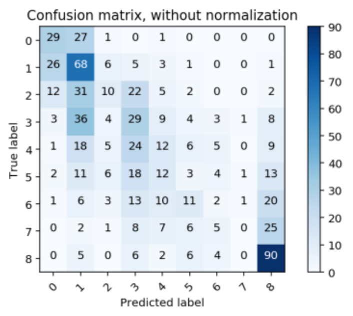

**Fig. 5.** Confusion Matrix of using Logistic Regression (C=1.8, solver=’lbfgs’, multinomial, max_iter=200)

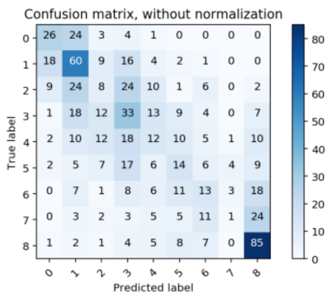

**Fig. 6.** Confusion Matrix of using Random Forest (n_estimators=1000, criterion=’gini’)

We can see that both SVM and Random Forest models are getting a similar accuracy of over 0.34, whereas logistic regression model performs a little worse, with an accuracy over 0.33. In terms of one-away accuracy, random forest model did best in getting as close revenue range as possible, while SVM gets the worst accuracy in one-away prediction.

### 3.2 Results of Artificial Neural Network

| | **Neural Network** |
|---|---|
| **Bingo** | 0.357 |
| **One-Away** | 0.604 |

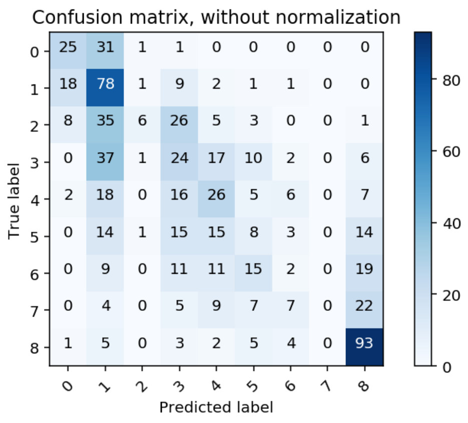

**Fig. 7.** Confusion Matrix of using Artificial Neural Network

Due to the nature of the dataset, the use of neural network did not improve much upon the statistical machine learning result. After using grid search cross validation [10], I found that stochastic gradient descent [7] generally performs better than other optimizers, and with a learning rate of 0.01, after 300 epochs, the model could achieve a test accuracy of 0.357 and one-away accuracy of 0.604, which are all higher than that of using statistical machine learning approaches. The library I used is TensorFlow [8] with Keras [9] as the high-level library.

## 3.3 Feature Importance

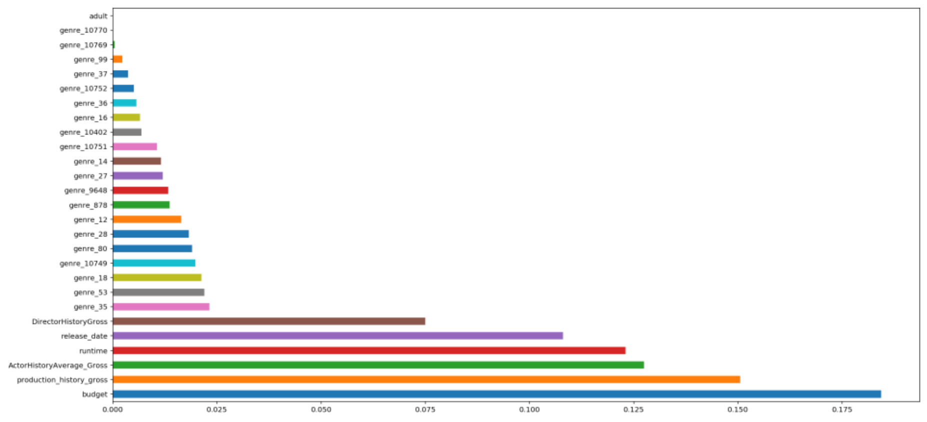

**Fig. 8.** Feature Importance via Random Forest Classifier

**Table 4.** Feature Importance Values via Random Forest Classifier

| | |
|---|---|
| *budget* | 0.184396 |
| *production_history_gross* | 0.150550 |
| *ActorHistoryAverage_Gross* | 0.127442 |
| *runtime* | 0.123007 |
| *release_date* | 0.108049 |
| *DirectorHistoryGross* | 0.074995 |
| *genre_35* | 0.023213 |
| *genre_53* | 0.021960 |
| *genre_18* | 0.021263 |
| *genre_10749* | 0.019065 |
| *genre_28* | 0.018769 |
| *genre_80* | 0.018048 |
| *genre_12* | 0.016768 |
| *genre_878* | 0.012839 |
| *genre_27* | 0.011432 |
| *genre_9648* | 0.011106 |
| *genre_10751* | 0.010948 |
| *genre_14* | 0.010770 |
| *genre_16* | 0.006826 |
| *genre_10402* | 0.005871 |
| *genre_36* | 0.004974 |
| *genre_99* | 0.001404 |
| *genre_10769* | 0.000375 |
| *genre_10770* | 0.000135 |
| *adult* | 0.000000 |

I used random forest classifier to extract the feature importance, because it is relatively difficult to extract feature importance from other machine learning algorithm as it requires permutating on each of the features and testing its sensitivity to the result. Since the result of random forest classifier is close to the highest accuracy we have got, it is reasonable to use random forest to look at feature importance.

## 4 Conclusion and Discussion

Based on the result from both statistical machine learning methods and artificial neural network methods, I was able to predict the movie box office revenue with an accuracy of 35.7% and one-away accuracy of 61%. The result is better than Jiayong Lin's result [4] using the same class labeling with smaller size of dataset. Though my accuracy is 3.3% lower than Dr. Sharda and Delen's backpropagation model [3], they only used over 834 movies, which is not even 20% of my dataset size. However, the one-away accuracies on all my models are not lower than Dr. Sharda and Lin's. Dr. Zhang et al. [5] used 6 class labeling as their baseline model and only 214 movies. Therefore it is almost certain that they would reach a higher accuracy than mine. There is no baseline to compare between their model and mine at this moment.

From the bar graph depicting feature importance via random forest classifier, we can see that *budget* is the most important factor determining the box office revenue. It is reasonable because budget is related to many other features: hiring good director and actors needs decent payroll; making high-quality after effect and building epic stages/scenes increases expense as well. Following is the *average history gross revenue of the production companies* because only well-reputed production companies could afford such high expense. Then interestingly, we have *runtime* and *actor history gross* as next important features. Since the model does not have definitive high accuracy, I believe the runtime feature here should not be assigned a lot of weights. Nonetheless, actors "star values" should be addressed important. Then we have *release date* as fourth important feature, because in our database the year spans almost 30 years and the model need the release year to have a baseline on the average revenue of movies in the same year. *Director history gross* is ranked second-last in the non-genre features but is still much more important than the genre features. *Adult-only* is not important at all as it received 0 feature importance.

However, there are a lot of improvements and explorations that can be done. I am planning on resetting my labeling into 6 class labeling and run my neural network model in order to compare my accuracy with Dr. Zhang et al.'s. [5]. To further improve the accuracy on my model, I also intend to divide my dataset by years and train a model for movies each year. Since the time difference is dramatically reduced, I'm expecting an improvement in accuracy for a year-by-year model. Reflecting on the feature importance graph, some of the features that could potentially be interfering the prediction, such as adult, which has 0 importance in the prediction, and runtime, could be removed.

Lastly, the revenue in the dataset is overall revenue which is accumulated over time. People will be affected by features such as ratings, social media reviews, etc.. Therefore, the revenue itself needs to be modified to a more instant value such as first-weekend revenue, or 2-week revenue, which could better reflect the market without much interference from after-release factors. I think it will be very promising to have a higher accuracy with the improvements mentioned above.

## 5 References

1. Rounak Banik, “The Movie Dataset | Kaggle”, https://www.kaggle.com/rounakbanik/the-movies-dataset, last accessed 2018/11/5.
2. Nikhil Apte, Mats Forssell, Anahita Sidhwa: “Predicting Movie Revenue”. http://cs229.stanford.edu/proj2011/ApteForssellSidhwa-PredictingMovieRevenue.pdf, last accessed 2016/12/16.
3. Sharda, Ramesh, and Dursun Delen. "Predicting box-office success of motion pictures with neural networks." Expert Systems with Applications 30.2 (2006): 243-254.
4. Jiayong Lin, “Movie Box Office Prediction System”, https://www.kaggle.com/tmdb/tmdb-movie-metadata/discussion/28576, last accessed 2018/11/5
5. Zhang, L., Luo, J., & Yang, S. (2009). Forecasting box office revenue of movies with BP neural network. Expert Systems with Applications, 36(3 PART 2), 6580–6587. http://doi.org/10.1016/j.eswa.2008.07.064
6. Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.
7. Bottou, Léon. "Large-scale machine learning with stochastic gradient descent." Proceedings of COMPSTAT'2010. Physica-Verlag HD, 2010. 177-186.
8. Abadi, Martín, et al. "Tensorflow: a system for large-scale machine learning." OSDI. Vol. 16. 2016.
9. Chollet, François. "Keras: Deep learning library for theano and tensorflow.(2015)." There is no corresponding record for this reference (2015).
10. Hsu, Chih-Wei, Chih-Chung Chang, and Chih-Jen Lin. "A practical guide to support vector classification." (2003): 1-16.