

Movie Box Office Prediction

Quan Yuan

School of Computing Science
Simon Fraser University
qya23@sfu.ca

Yuheng Liu

School of Computing Science
Simon Fraser University
yuheng_liu@sfu.ca

Wenlong Wu

School of Computing Science
Simon Fraser University
wwa95@sfu.ca

Motivation and Background

Since the rapid growth in movie production, movie has already become an important part of daily life. Every day there are various new movies released. Thus, more money flows into the movie market. Investing a movie usually costs a huge amount of money. Normally the revenue is higher with the higher investment; however, once the investment fails, the loss will also be a huge amount of money, which is not the investors and production company expect. To help them reduce the risk, we decided to build such a system to predict the movie box office, so that the production company and investors can evaluate the movie to be produced in more details.

In addition to this, the system can also help audiences make decisions when they are choosing movies. Normally people will refer to the rates and comments of the movie to decide whether the movie is worth watching or not. However, the rates and comments sometimes are not accurate as they could be affected by some artificial factors. Thus, with our box office prediction system, the audiences can know whether the movie to be released will be popular or not, which could be a good indicator when they are choosing movies.

Problem Statement

1. How to scrape the data from the website?

We intended to scrape data from IMDB, Twitter, Rotted Potatoes, Douban. However, some of them have a security protection system that declines our large number of HTTP requests. Luckily, IMDB seems more friendly for people to access their data. In terms of this issue, we decided to scrape data only from IMDB. For the scraping tool, we chose BeautifulSoup for parsing HTML and XML documents. To achieve a high-efficiency scraping, the webpage “top-50 movie” was selected to be scraped, and we found there was a tag “release_date” in the URL of “top-50 movie”, which was the key to access data from different years. Based on this, we build several lists to store the movie’s info. After iterating a number of loops, we realized only top-50 was not enough for us to analyze. Therefore, we found there was another tag in the URL “start” allowing us to change the starting ranking number for scraping up to 10,000 movies per year. At the end of the scraper program, the lists are converted to Pandas dataframe for further process. The figure below shows movies is retrieved accordingly by the tags in URL on the IMDB website.

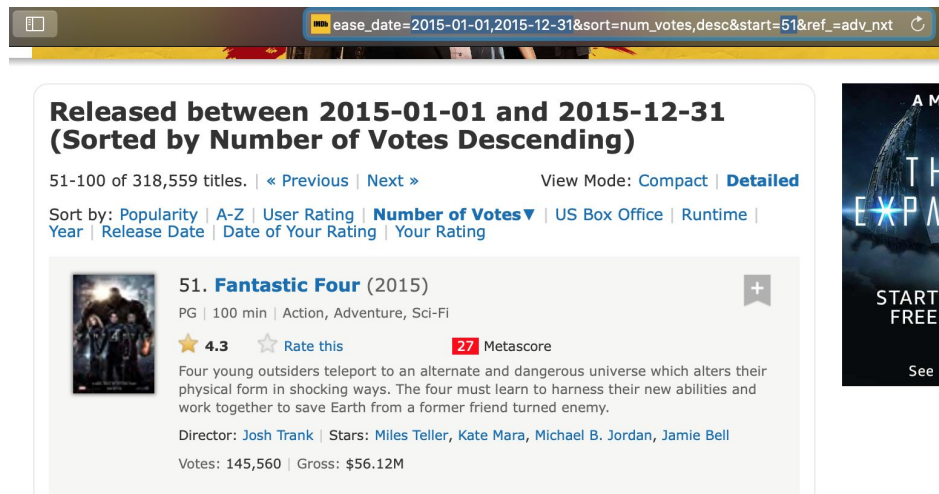


Figure 1 - IMDb "Top-50" webpage

2. How to cleanse the raw data?

As all the data are scraped from the website, there exists some irrelevant, redundant, or empty data that needs to be clean. For example, as we scrape 10,000 movies from IMDB every year, some tv-series are scraped, which are not considered for this project study, thus we just simply remove them. For some movies that have empty values on certain columns, the measure we take is to fill up the empty fields with mean values.

3. How to choose the appropriate algorithm to build the model?

In order to build a better model and let the model better reflects the actual situation of the measured data, we tried many different feature combinations. The models we have compared are linear regression, random forest, neural network, and xgboost, and we found xgboost generated the best result.

Data Science Pipeline

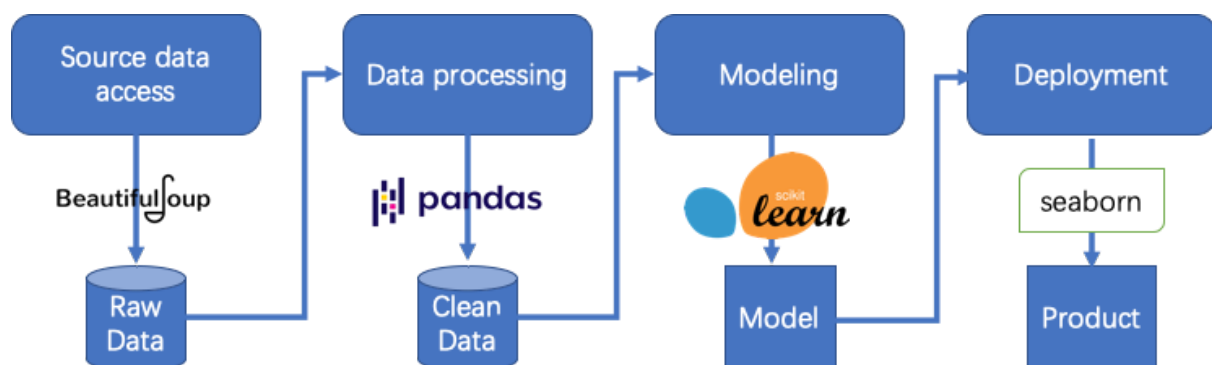


Figure 2 - Pipeline

For the data source, the online movie database - IMDB is selected to scrape movies' info, such as movie name, release year, content rating, runtime, and so on. Besides IMDB, we also chose Twitter

hashtag trends as another feature of “Popularity”, which reflects the popularity of the particular movie during a certain period. After obtaining the raw data, the data processing started. This includes converting some non-numeric features into numeric to facilitate training models. For example, to deal with the movie’s genre features, our solution is to use one-hot to represent the different types of genre with numerics. Also, the average box office of each actor is calculated to demonstrate the influence made by the actor. According to the star cast, the average box office of these stars in the movie is summed to represent the new feature - “stars_value”, which shows the total average box office made by the cast.

After processing the data, we performed model selection and finally chose XGBoost. Then we made a selection of features among many possible combinations. For the purpose of maximizing a model’s performance without overfitting or creating a high variance, we performed model tuning, which are grid search, cross-validation, and r2 score approaches.

Finally, we combined the model results and clean data, and used seaborn to generate different graphs for clients to make the decision.

Methodology

1. Feature Engineering

1.1 Feature Introduction

	movie	year	release_date	content_rating	runtime	genre	director	stars	imdb	gross	metascore	vote	country	budget	worldwide_gross	production_company	avg_all	avg_30	max_30	near_holiday
0	The Dark Knight	2008.0	18-Jul-08	PG-13	152 min	Action, Crime, Drama	Christopher Nolan	['Christian Bale', 'Heath Ledger', 'Aaron Eckh...	9.0	534858444	84.0	2187347	USA	\$185,000,000	1004934033	Warner Bros.	0.251533742	0	0	False
1	WALL·E	2008.0	27-Jun-08	G	98 min	Animation, Adventure, Family	Andrew Stanton	['Ben Burtt', 'Elissa Knight', 'Jeff Garlin', ...	8.4	223808164	95.0	951079	USA	\$180,000,000	521311860	FortyFour Studios	2.705521472	0	0	False
2	Iron Man	2008.0	2-May-08	PG-13	126 min	Action, Adventure, Sci-Fi	Jon Favreau	['Robert Downey Jr.', 'Gwyneth Paltrow', 'Terr...	7.9	318412101	79.0	902365	USA	\$140,000,000	585366247	Paramount Pictures	7.012269939	0	0	False
3	Slumdog Millionaire	2008.0	25-Dec-08	R	120 min	Drama, Romance	Danny Boyle	['Loveleen Tandan', 'Dev Patel', 'Freida Pinto...	8.0	141319928	85.0	771043	UK	\$15,000,000	378021385	Celador Films	0.024539877	0	0	True
4	Gran Torino	2008.0	9-Jan-09	R	116 min	Drama	Clint Eastwood	['Clint Eastwood', 'Bee Vang', 'Christopher Ca...	8.1	148095302	72.0	693297	Germany	\$33,000,000	269958228	Matten Productions	0.024539877	0	0	False

Figure 3 - Raw Dataset

The most important task of feature engineering is to convert those categorical variables into numerical variables, as we need to build the model for the system and the model only accepts numerical inputs. Thus, by using Pandas library, the raw data can be cleansed and integrated to the data we want. Then, we can build a model and analyze it based on the integrated data. Actually, both Pandas and Spark are able to accomplish the task of cleansing and integrating data. We decided to use Pandas based on the following reasons:

- During the development, we frequently used IPython Notebook. Pandas offers a more tabular view of dataframe than Spark does.

- Pandas dataframes are in-memory and single server, and Spark dataframes are distributed on the clusters. As for the project, the data size is not large, all the processes are executed on the local machine. Thus, using Pandas is relatively more efficient.
- Compared to Spark, Pandas library is relatively more convenient and powerful, which saves us much time on cleansing and integrate data.
- Every time running the Spark, it always takes much time on setting up and starting Spark. However, with Pandas, the time can be saved.

1.2 Feature expansion

It is not enough to only focus on the meta information of the movie. Some features not included in the dataset scraped from IMDB and some external factors have potential influence on the final box office. For example, the release date which is near an official holiday may lead more families to go to the cinema. Hence, we integrated some other information sources to create some new features beneficial to our model. 'Near_holiday' column was generated according to whether the release date is near to an official holiday.

Also, the popularity of an upcoming movie plays an important role in the final box office and rating. How to quantify popularity is a critical problem in this project. Trendsmap.com provides the trend of a specific topic and how many people discussed on this topic on Twitter.

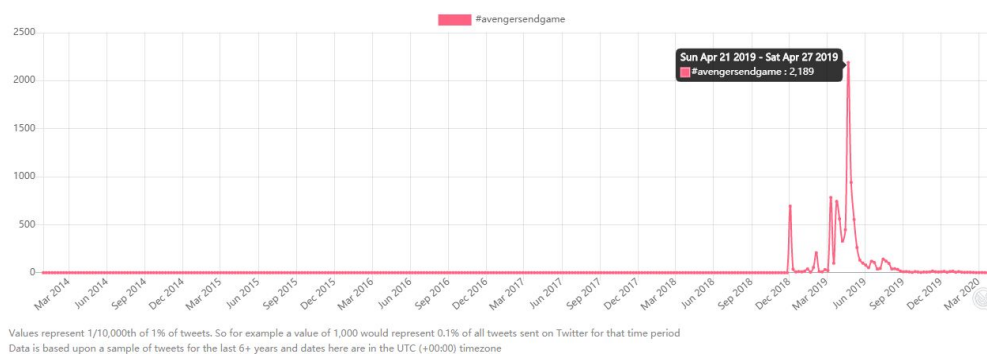


Figure 4 - Topic popularity on Twitter

It is straightforward to show the popularity of a certain movie on the social media. To obtain the convincing data from this website, we extracted the keywords from the title of the movie and regarded those keywords as a topic on Twitter to search and scrape the data from this website. Three features were generated by this step. The average popularity, the maximal popularity around the release date and the average popularity among the whole timeline. These three different features can give us different information to our model.

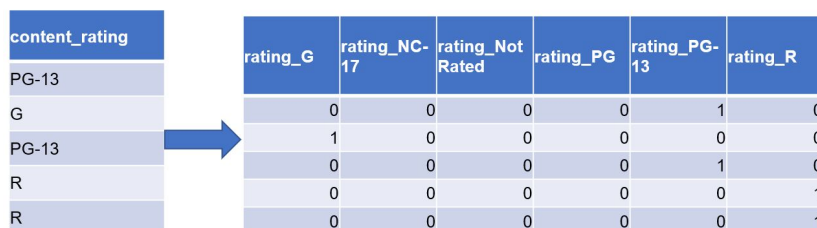
movie	avg_all	avg_30	max_30
Joker	6.871166	119.75	224
Avengers: Endgame	28.89877	1032.5	2189
Once Upon a Time... in Hollywood	1.03681	16.75	36

Figure 5 - popularity column

1.3 Categorical variables processing

As we mentioned before, the main task was to convert categorical variables to numeric variables. For different columns, we used three methods to do the transformation. That is one-hot encoding, mean/sum encoding and only focusing on does it have and how many it has.

1.3.1 One-hot Encoding



content_rating	rating_G	rating_NC-17	rating_Not Rated	rating_PG	rating_PG-13	rating_R
PG-13	0	0	0	0	1	0
G	1	0	0	0	0	0
PG-13	0	0	0	0	1	0
R	0	0	0	0	0	1
R	0	0	0	0	0	1

Figure 6 - One-hot encoding

This is a basic way to convert categorical variables to a vector. Thus, we used it on content rating column. However, it is not proper to transfer 'stars' column to a vector because there would be too many columns after transformation, which may slow down the training process and reduce the accuracy of the model. Therefore, it comes to our second method.

1.3.2 Mean/Sum encoding



stars
['Christian Bale', 'Heath Ledger', 'Aaron Eckhart', 'Michael Caine']
['Ben Burt', 'Elissa Knight', 'Jeff Garlin', 'Fred Willard']
['Robert Downey Jr.', 'Gwyneth Paltrow', 'Terrence Howard', 'Jeff Bridges']
['Loveleen Tandan', 'Dev Patel', 'Freida Pinto', 'Saurabh Shukla', 'Anil Kapoor']
['Clint Eastwood', 'Bee Vang', 'Christopher Carley', 'Ahney Her']

stars	
Robert Downey Jr.	418513851
Quayne Johnson	208295712
Chris Hemsworth	2771961838
Mark Ruffalo	2712137655
Chris Evans	2683433425
Joe Russo	1945272831
Bradley Cooper	1862669919
Chris Pratt	1854656331
Zoe Saldana	1846924448
Vin Diesel	1774376282

stars_value
1082431638
3399544
2716331629
3923375
529069413

Figure 7 - mean/sum encoding

To avoid generating too many columns, we could replace the original column by a related number. The total or average box office of an actor is a good alternative to represent the value that an actor brings to the movie. However, does this method cause over-fitting? Because we derived this new column by the target column, the influence of this method may cause our model to fully depend on this column. In the following experiments, we learned more about this method and we will talk about it later.

1.4 Has famous actors? And how many?

Sometimes, people decide to watch a movie just because a famous actor plays in this movie. That means we can only focus on does any famous actor play in this movie and how many famous actors are in this movie. We selected the top K actors and replaced the original column by the number of famous actors in this movie. For some other features, such as director, production_company, this is a

good way to reduce the amount of generated columns and meanwhile remain the information. For ‘actor’ column, who is in this movie also matters. Hence, we generated a vector of only K columns to record who plays in this movie.

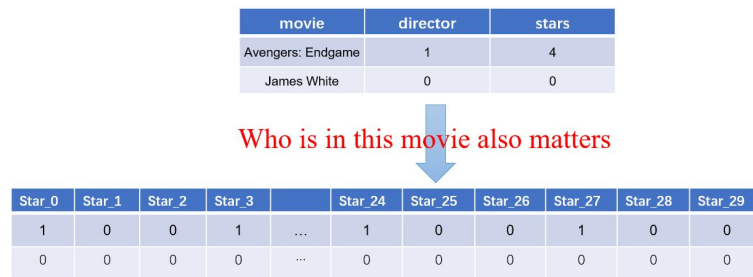


Figure 8 - ‘stars’ column processing

2. Modeling

After the feature engineering, the next step is to build a model. The workflow of modeling is shown below.

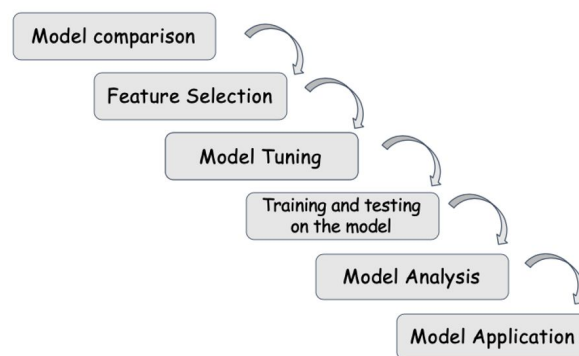


Figure 9 - Modeling workflow

2.1 Model Comparison

In order to get the best effect, model comparison is the first step that we need to do. By using different algorithms to build the model and running a basic prediction, it was found that tree based algorithms produced better results than linear regression. Eventually, XGBoost algorithm was selected.

2.2 Feature Selection

As there were lots of different combination of features, it was hard to tell which kind of methods of feature engineering was the best one. Several rounds of testings were conducted and we calculated the feature importance to decide which feature needs to be removed or kept.

The following figure shows the feature importances when using mean encoding to convert categorical variables to numeric data. The top 3 columns were generated by mean encoding according to the gross column (the target column). Because those columns were calculated by the target column, it made their feature importances very high. It seems like we were using the target value as a feature to predict the

target value. Hence, we need to penalize those features or use the third method mentioned before to replace.

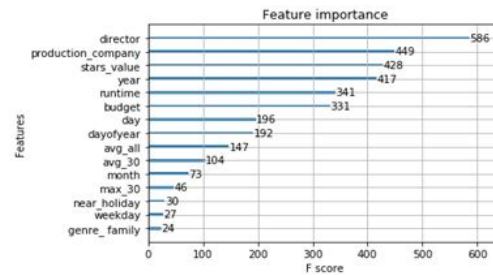


Figure 10 - Feature importance-I

The second figure occurred when we added vote column as a feature. Before a movie is released, it is impossible to get the vote column. Thus, it is not corresponded with the fact to include this column for modeling.

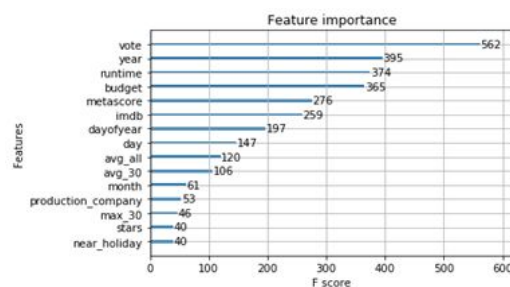


Figure 11 - Feature importance-II

2.3 Model Tuning

To make the model have a better performance, grid search was used for choosing the hyperparameters. Through the grid search and cross validation, we avoided overfitting and increased the accuracy of the model.

2.4 Training and Testing

The dataset was divided into training set and testing set with the ratio 7 : 3. Due to the large fluctuation of the target value, we tried to use log operation to resize the target value in order to make it more stable. But we eventually abandoned it because it didn't bring any benefit to the result. After training with the model we got from the previous step, we calculated the r2 score to evaluate the performance of our model on the testing set. The final score represents that our model has the ability to predict a possible box office but it is not very precise.

2.5 Model Analysis and Application

The purpose of this project is to provide an quantitative advice for audiences about how to choose a good movie and also help the producer figure out what kind of factors that they can control to increase the revenue of the movie. For now, the model can reflect whether a movie could be popular according

to its features. Some visualizations mentioned below also represents the relationship between those features and the box office. By controlling important features, we can do a prediction about the possible change on the revenue and help producers to make their plans.

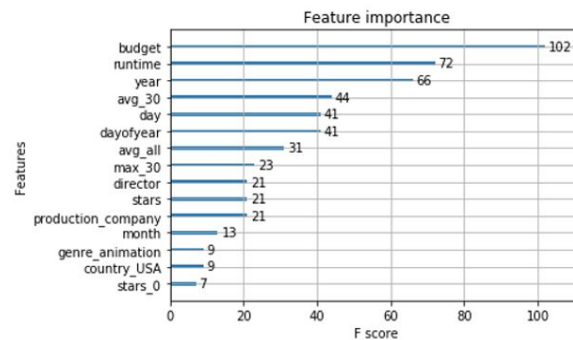


Figure 12 - Feature importance-III

3. Evaluation

After feature selection and model tuning, we had done all the preparation of building the final model. The metrics used for evaluation is r2 score. To tell if the model is overfitting, we used cross validation and calculated the score of validation dataset. Early stopping was an another method used for avoiding overfitting. Through observing the learning curve, we could easily figure out when the overfitting may happen and then stop training in this iteration.

The score of testing phase represents the model is able to reflect the possible revenue to a certain extent. However, the predicted value still has a difference with the actual value. The reason of the result may be that the scale of some features and the target value is too large or some important features are lost in the dataset. So, our next move is to eliminate the influence of the scale of numeric variables.

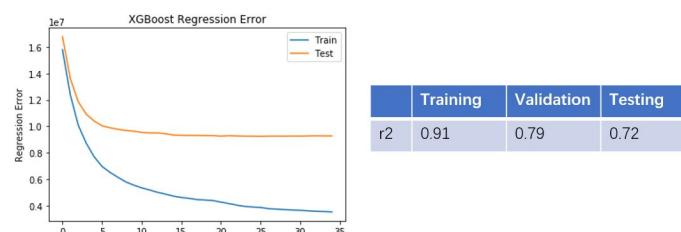


Figure 13 - Result

Data Product

1. Visualization

With necessary data cleaning and integration, the raw data is converted to the readable integrated data. Based on the integrated data, data analysis on the relationship between revenue and other factors can be made. Thus, we built graphs below to demonstrate the relationships, which effectively show what would be the indicators to the high revenue. And this could also be referred by the audience when they are choosing movie or the production company and investors when they intend to invest a movie.

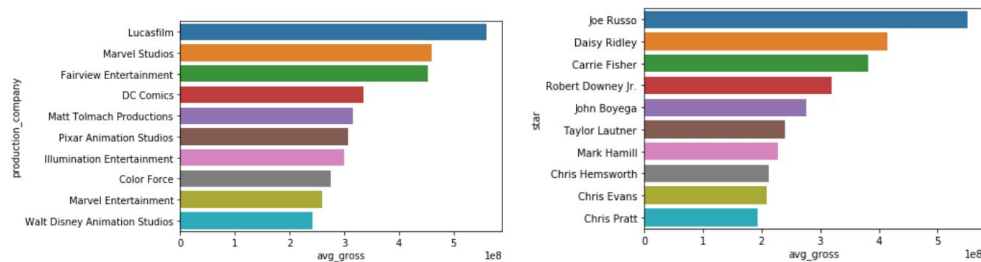


Figure 14 - Visualization

2. Model

By applying feature engineering to the integrated data, those important features are successfully extracted, and converted into the format that is acceptable to the model. The processed features are stored as a data file. Thus, the model is constructed and trained with the data file. According to the various comparisons, XGBoost algorithm is applied to the model. The model is able to predict the movie box office given the related information of the movie. Also, more studies could be continued on the data file.

Lessons Learnt

1. Data Collection

From this project, we learned how to collect data from websites using BeautifulSoup. Based on the APIs, we successfully retrieved the related information of the movie. Apart from assignment 1, we have achieved scraping information on different pages rather than scraping information on a single page.

2. Data Cleaning and Integration

As some columns contain NaN values, we need to fill them up with proper value. Also, some columns hide some potential information so that we need to find it out by ourselves. When we were implementing data integration, we tried to combined every aspects that might influence the revenue of the movie. Not only use the meta information of the movie, but focus on some external factors. From this project, we learned that every step in data science domain is not independent. The method used in data cleaning and integration must serve for the modeling phase. It is important to combine every steps to solve the problem.

3. Feature Engineering

The main task of feature engineering in this project is to convert categorical variables into numeric data. It is not as simple as using one-hot encoding to do the transformation. Different encoding methods may cause different influence to the model. From this part, we learned multiple ways to replace the original categorical variable by a related number. In the meanwhile, this method would keep the model conservative to the testing set.

Summary

In this project, we propose a machine learning approach to predict the movie box office. First, we collected the movie's basic information by using BeautifulSoup python package as the scraping tool. The scraped dataset contained invalid data and noisy data which might influence the accuracy of our model result. To avoid this kind of issue, python pandas and other techniques(one-hot encoding, mean/sum encoding) were applied to process the raw data. After processing, we performed a model selection for better reflects the actual situation of the measured data and the XGBoost model stood out with higher accuracy. In the end, we combined the model prediction results and seaborn to provide visualization for clients. Based on the model, the movie box office can be predicted.

Reference

1. Z. Di, J. Xiu, J. Lin and Y. Qian, "Research on movie-box prediction model and algorithm based on neural network," *2016 4th International Conference on Cloud Computing and Intelligence Systems (CCIS)*, Beijing, 2016, pp. 224-228.
2. J. Ahmad, P. Duraisamy, A. Yousef and B. Buckles, "Movie success prediction using data mining," *2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, Delhi, 2017, pp. 1-4.
3. K. R. Apala, M. Jose, S. Motnam, C. -. Chan, K. J. Liszka and F. de Gregorio, "Prediction of movies box office performance using social media," *2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2013)*, Niagara Falls, ON, 2013, pp. 1209-1214.
4. N. Quader, M. O. Gani, D. Chaki and M. H. Ali, "A machine learning approach to predict movie box-office success," *2017 20th International Conference of Computer and Information Technology (ICCIT)*, Dhaka, 2017, pp. 1-7.
5. J. Brownlee, "Avoid Overfitting By Early Stopping With XGBoost In Python", 2019.
Retrieved from
<https://machinelearningmastery.com/avoid-overfitting-by-early-stopping-with-xgboost-in-python/>