

Movie Revenue Prediction Using Machine Learning Models

Divyanshu Dubey

BS DataScience IIT-Madras

23f2001494@ds.study.iitm.ac.in

Abstract—In the contemporary film industry, accurately predicting a movie's earnings is paramount for maximizing profitability. This project aims to develop a machine learning model for predicting movie earnings based on input features like the movie name, the MPAA rating of the movie, the genre of the movie, the year of release of the movie, the IMDb Rating, the votes by the watchers, the director, the writer and the leading cast, the country of production of the movie, the budget of the movie, the production company and the runtime of the movie. Through a structured methodology involving data collection, preprocessing, analysis, model selection, evaluation, and improvement, a robust predictive model is constructed. Linear Regression, Decision Trees, Random Forest Regression, Bagging, XGBoosting and Gradient Boosting have been trained and tested. Model improvement strategies include hyperparameter tuning and cross-validation. The resulting model offers promising accuracy and generalization, facilitating informed decision-making in the film industry to maximize profits.

I. INTRODUCTION

A. Motivation

Imagine you are a filmmaker or head of a movie production house and you have a big question: what makes a movie a blockbuster hit or a flop?

You might think it depends on the star power of the actors, the vision of the director, the budget of the production, or the genre of the story.

Or you might think it is simply the quality of the storytelling that captivates the audience and earns high ratings. But the answer is not straightforward or easy.

There are many factors that influence the earnings of a movie, and the true combination of these factors has not been mastered yet. That's why we have developed a machine learning model that reveals the most important factors for box office success by analyzing real data from a wide variety of movies produced around the world. With our model, filmmakers can make more informed decisions and optimize their movie production for maximum profit and popularity.

B. Rationale

We hypothesize that certain parameters hold more significance in predicting movie revenue than others. Specifically, we conjecture that the director's track record and the genre of the film carry substantial weight in this prediction model.

Our observations suggest that despite lower IMDb ratings, action-oriented films often demonstrate strong performance

at the box office. Conversely, genres such as comedy or emotional dramas, despite potentially higher IMDb ratings, may not achieve comparable revenue outcomes to their action counterparts.

These insights underscore the complex interplay between film attributes and audience preferences, prompting us to assign greater importance to factors like directorial history and

genre classification within our predictive framework.

C. Overview

In this project, we follow a structured methodology to build and evaluate our predictive model. We first collect a large dataset of movies and their features from various sources and

custom tailor the datasets to suit our needs.

We then pre-process the data to handle missing values, outliers, and categorical variables. We perform data analysis to explore the data and understand its characteristics and relationships. We use descriptive statistics, inferential statistics, and data visualization techniques to gain insights into the data like using a graph to compare the accuracy of our model's performance of training and test data.

We then select several machine learning algorithms that are suitable for regression tasks, such as decision trees and random forests. We train and test our models using cross-validation and compare their performance using metrics such as R-squared mean error and Mean Squared Logarithmic Error. We also apply model improvement strategies such as hyper-parameter tuning, feature selection and regularization to enhance the accuracy and generalization of our models. The resulting model offers promising results and can be used to predict the revenue of any movie based on its features.

II. LITERATURE REVIEW

In this section, we will see the definitions of Principal Component Analysis (PCA), Label Encoder, Standard Scaler,

SelectKBest features, GridSearchCV, Train Test Split and provide some literature content on the models which we are

going to use. We will also explain the evaluation metrics 'R² score' and 'MSLE'.

Principal Component Analysis (PCA)

PCA is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly

correlated variables into a set of values of linearly uncorrelated variables called principal components. This technique is used to emphasize variation and bring out strong patterns in a dataset.

Label Encoder

Label Encoder is a utility method to convert categorical data into numerical data. It assigns each unique category in the data to an integer value, making the data more suitable for algorithmic processing.

Standard Scaler

StandardScaler is a preprocessing technique used to standardize features by removing the mean and scaling to unit variance. It transforms the data such that its distribution will have a mean value of 0 and a standard deviation of 1. This standardization is particularly useful for machine learning algorithms that assume all features are centered around zero and have variance in the same order.

SelectKBest Features

SelectKBest is a feature selection method in Scikit-Learn. It selects features according to the k highest scores of a specified scoring function. It's a way to select the 'k' best features in your dataset, where 'k' is a parameter you choose.

GridSearchCV

GridSearchCV is a library function that is a member of sklearn's model selection package. It helps to loop through predefined hyperparameters and fit your estimator (model) on your training set. In addition to that, you can specify the number of times for the cross-validation for each set of hyperparameters.

Train Test Split

The 'train test split' function from the 'sklearn.model_selection' module is a utility that divides a dataset into randomized training and testing subsets. Each subset is distinct, meaning that no data point can be present in both subsets. This allows for the model to be trained on one subset of the data, and then validated on an entirely separate subset. In our case, we applied an 80/20 split on our dataset for training and testing our models.

Models

A. Linear Regression

Linear Regression is a statistical approach for modelling the relationship between a dependent variable and one or more independent variables.

B. Decision Trees

A Decision Tree is a decision support tool that uses a tree-like model of decisions and their possible consequences. It is one way to display an algorithm that only contains conditional control statements.

C. Gradient Boosting

Gradient Boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees.

D. Bagging

Bootstrap Aggregating, often abbreviated as Bagging, is a meta-algorithm designed to improve the stability and accuracy of machine learning algorithms used in statistical classification and regression. It also reduces variance and helps to avoid overfitting.

E. Random Forests

Random Forests is a learning method that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.

F. XGBoosting

XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework. The XGBoost library offers several advantages over traditional machine learning algorithms, including:

- Parallelization: XGBoost supports parallelization, which allows it to train models efficiently on large datasets by leveraging multiple CPU cores or GPUs.
- Regularization: XGBoost includes built-in regularization techniques, such as L1 and L2 regularization, to prevent overfitting and improve model generalization.
- Handling Missing Data: XGBoost can automatically handle missing data without the need for imputation, making it more robust and efficient.

Evaluation Metrics

R² Score

The R² score, also known as the coefficient of determination, is a statistical measure that shows the proportion of the variance for a dependent variable that's explained by an independent variable or variables in a regression model. It provides an indication of goodness of fit and therefore a measure of how well unseen samples are likely to be predicted by the model.

Mean Squared Logarithmic Error (MSLE)

The Mean Squared Logarithmic Error (MSLE) is a statistical measure used to evaluate the accuracy of regression models, particularly when the target variable exhibits exponential growth or when the dataset contains outliers. It is calculated as the average of the squared differences between the logarithm of the predicted values and the logarithm of the actual values. MSLE provides an understanding of the relative error between predictions and actual values, giving more weight to underestimates than overestimates. A lower

MSLE value indicates a better fit of the model to the data. For example, if your MSLE is 0.1, it suggests that your model's predictions deviate from the actual values by approximately 10% on average when considering the log-transformed scale.

III. DATASET SELECTION

A. Modification of Dataset Strategy

In our initial proposal, we constructed a bespoke dataset by integrating four distinct datasets: The **Movies Industry Dataset**, **IMDb 5000 Movies Multiple Genres Dataset**, **IMDb 5000 Movies Dataset**, and **Top 500 Movies Budget**.

This integration was performed sequentially, resulting in a **comprehensive dataset encompassing 7118 movies** with input variables such as budget, director, genre, leading crew, and IMDb ratings, and the output variable being gross revenue. (Refer to the **README** for a detailed explanation of the dataset construction process).

However, during the progression of our project, we identified the need to diverge from our originally proposed datasets. We opted for a refined dataset derived solely from **The Movies Industry Dataset**, achieved by excluding entries with missing values. This pivotal decision was informed by a multitude of considerations, which are detailed in the following subsections.

B. Rationale Behind the Dataset Transition

The primary challenge encountered with our initially finalized dataset was its suboptimal performance with our predictive models, reflected in lower accuracy rates. This was attributed to our initial approach of selecting input variables without a thorough analysis of the available data, leading us to force-fit the datasets into a preconceived framework of revenue determinants, thus compromising the integrity of the final dataset. Additionally, the potential for confusion due to movies sharing similar titles necessitated a cautious approach to dataset merging, which presented its own set of complexities.

Acknowledging these issues, we paused to re-evaluate our methodology. We delved into a deeper examination of the data surrounding movie revenue prediction and gained insights into structuring the dataset to train a model that is both comprehensive and robust in forecasting movie earnings.

C. Benefits of the Optimized Dataset

Our continued exploration for an ideal dataset that encapsulates the core of our research led us to "The Movies Dataset," which aligned perfectly with our criteria. We ensured the dataset's reliability by eliminating entries with null values before subjecting it to our predictive models. A significant advantage of our current dataset is its ability to address the limitations of our prior data. It includes additional input variables such as Year, Production Company, and Votes (*alongside the IMDb score*), which collectively enhance the accuracy of our machine learning model in predicting movie revenues. Moreover, sourcing the dataset from a single origin has eliminated the extraneous noise that previously arose from dataset amalgamation.

IV. DATA ANALYSIS

TABLE I: Description of Movie Dataset Features

Feature Name	Description
Name	The title of the movie
Rating	The MPAA rating of the movie
Genre	The genre of the movie
Year	The year the movie was released
Released	The release date of the movie
Score	The IMDb rating of the movie
Votes	The number of votes the movie received
Director	The person who directed the movie
Writer	The person who wrote the movie script
Star	The main actor or actress in the movie
Country	The country where the movie was produced
Budget	The budget of the movie
Company	The production company of the movie
Runtime	The duration of the movie

In the data analysis phase, we thoroughly examined the collected movie dataset to gain insights into its structure, distribution, and relationships between features and the target variable (*movie earnings*). We utilized descriptive statistics, inferential statistics, and data visualization techniques to explore the data.

A. Descriptive Statistics

We computed summary statistics such as mean, median, standard deviation, minimum, maximum, and quartiles for numerical features like budget and ratings. This gave us a general understanding of the central tendency and spread of the data.

B. Inferential Statistics

We conducted statistical tests to analyze relationships between different features and the target variable. For example, we used correlation analysis to examine the strength and direction of relationships between numerical features and earnings.

V. PREPROCESSING

A. Dataset Parameters

We have 14 parameters, including both non-numerical and numerical data types.

- 1) name
- 2) rating
- 3) genre
- 4) year
- 5) released
- 6) score
- 7) votes
- 8) director
- 9) writer

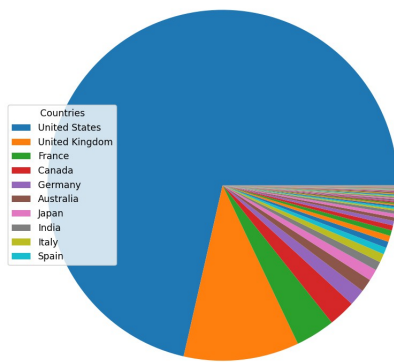


Fig. 1: Distribution of Movies by Country

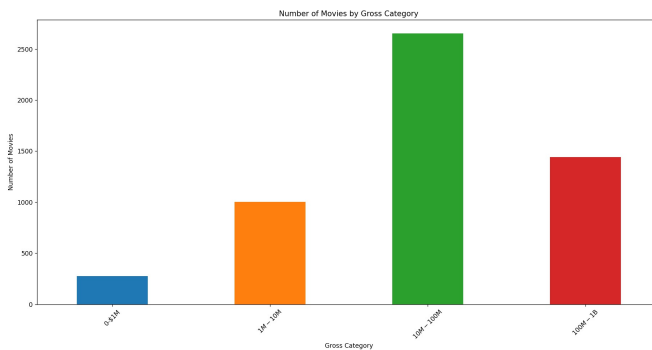


Fig. 2: Histogram of Gross Categories

- 10) star
- 11) country
- 12) budget
- 13) company
- 14) runtime

We have 9 non-numerical data types

- 1) name
- 2) rating
- 3) genre
- 4) released
- 5) director
- 6) writer
- 7) star
- 8) country
- 9) company

To prepare the data for regression models, we need to encode the non-numerical features into numerical labels. We will utilize the **LabelEncoder** from scikit-learn for this purpose.

B. Handling Null values

In the **Movie Industry Dataset**, there are 2,247 null values across the 11 parameters totalling 7669. Since budget and gross are our main parameter and output, we dropped those datasets and we were left with 5422 datasets.

```

name      0
rating    77
genre     0
year      0
released  2
score     3
votes     3
director  0
writer    3
star      1
country   3
budget    2171
gross     189
company   17
runtime   4
dtype: int64

```

Fig. 3: Null Values

C. PCA

We tried to use Principal Component Analysis but we were not able to achieve significant increase in the accuracy. We believe the reason for this is that PCA works effectively when there are more parameters and datasets. It will be more useful in image or sound regression/classification as the dimensions of the image/sound will come into play. Thus, we have decided to drop PCA in our preprocessing.

D. Best Features

We wanted to know which features contribute more to the revenue of the movie.

So we had used **SelectKBest** and then categorized and predicted all the 8668 features which can be viewed in **Features Scores**.

Since there were a lot of features for us to analyse, we then printed only those features with scores greater than 100 which can be viewed in **Best Features Scores**.

```

Feature      Score
rating_PG-13 194.402250
rating_R     337.959444
genre_Action 239.210623
genre_Animation 276.657989
genre_Comedy 116.786195
director_Anthony Russo 238.515051
director_James Cameron 127.387274
writer_Christopher Markus 199.157551
writer_James Cameron 108.337247
star_Chris Pratt 105.232686
star_Daisy Ridley 120.605232
star_Daniel Radcliffe 102.613119
star_Robert Downey Jr. 151.485500
company_Lucasfilm 110.297606
company_Marvel Studios 496.764152
company_Pixar Animation Studios 107.263914
company_Walt Disney Pictures 440.975532
year         440.975532
score        282.397725
votes        3292.085413
budget       6569.008340
runtime      446.121279

```

Fig. 4: K Best Features

As per our intuition, the budget played was the most deciding feature, with a score of about 6569. However, to our surprise, votes played the next most important role in predicting the revenue.

Additionally, runtime, year, and score also played a decent role in predicting the revenue.

Also, there were a few individual companies and individuals in the field of direction, cast and production who had an impact on the way the movies performed at the box office.

Also, there is a general trend that can be noticed where PG-13 and R-rated movies tend to do well.

Action, Animation, and Comedy are well-performing genres which also tend to increase the revenue of the movie.

E. Feature Scaling

Enhanced feature preprocessing is implemented in `feature_scaling.py`:

- Log transformation for skewed numerical features, particularly budget and revenue
- Standard Scaler applied to normalize numerical features
- Label Encoder used to encode categorical values, converting text-based categories into numerical format

F. Feature Engineering

New features are created in our models:

- vote score ratio
- budget year ratio
- vote year ratio
- score runtime ratio
- budget per minute
- votes per year
- is recent
- is high budget
- is high votes
- is high score

These engineered features capture complex relationships

and trends in the data, enhancing our model's ability to discover patterns. The combination of ratio-based and binary features provides a richer representation of the movie attributes, leading to improved predictive performance across our various models.

G. Impact on Model Accuracy

The implementation of these preprocessing techniques and feature engineering strategies has significantly enhanced our model's accuracy. By addressing data imbalances and capturing nuanced relationships, we've achieved substantial reductions in relative error metrics, resulting in more reliable and precise revenue predictions for stakeholders in the film industry.

VI. MODEL SELECTION

For predicting movie earnings, we have explored several regression techniques and ensemble methods like Linear Regression with and without PCA Preprocessing, Decision Trees, Gradient Boosting, Bagging, Random Forests and XGBoosting. Ensemble methods like Random Forests and Gradient Boosting are capable of handling complex interactions and non-linearities in the data.

1) *Linear Regression*: Initially used to explore the dataset's linear relationships and determine if a straight-line model could adequately represent the data.

2) *Gradient Boosting*: Employed as a predictive technique to create an ensemble model consisting of weaker prediction models (typically decision trees). Each subsequent learner in the ensemble aims to correct errors made by its predecessors. GB is known for its robustness against overfitting, making it a valuable algorithm for predictive tasks.

3) *Random Forest*: Utilized as an ensemble learning method for classification. RF constructs multiple decision trees and makes predictions based on the majority vote of these trees. Despite efforts to reduce dataset dimensionality, the RF model exhibited overfitting on the training set and did not outperform other models significantly.

4) *Decision Tree*: Employed as a hierarchical decision support model. DTs use a tree-like structure to represent decisions and their consequences, including chance events, resource costs, and utility. They provide a clear visualization of the algorithm's decision-making process.

5) *Bagging*: Used to improve model performance by training multiple models on random subsets of the original data. The results from these models are combined through a voting mechanism to make predictions, resulting in more accurate and robust predictions.

6) *XGBoost*: Applied as a popular machine learning algorithm under ensemble learning. XGBoost is effective for supervised learning tasks like regression and classification. It iteratively builds a predictive model by combining predictions from multiple individual models, often decision trees, to enhance prediction accuracy and model performance.

VII. MODEL IMPROVEMENT

In our pursuit of enhancing the model's performance, we employed several strategies. Let's break down the key components:

A. Hyperparameter Tuning using `GridSearchCV`

`GridSearchCV` is a powerful tool that performs an exhaustive search over specified parameter values for an estimator. In our case, we're using a *Gradient Boosting Regressor* with a squared error loss function. Here's how it works:

- We define a dictionary called `param_grid` containing various hyperparameters (e.g., number of estimators, maximum depth, and learning rate).
- `GridSearchCV` systematically tests different combinations of these hyperparameters using cross-validation (in our case, 5-fold cross-validation).
- The best combination of parameters, as determined by the highest R-squared score, is extracted using the `best_params` attribute.
- Armed with these optimal parameters, we initialize the final model.
- The model is then trained on the training data and evaluated on both the training and test datasets to validate its accuracy.

By following this process, we aim to create a robust and high-performing model.

B. Implementing `XGBRegressor` using `xgboost` library

XGBoost (Extreme Gradient Boosting) is a powerful and efficient implementation of the gradient boosting algorithm, which is widely used for various machine learning tasks, including regression and classification. In this project, we employed the `XGBRegressor` from the `xgboost` library to train and test a regression model for predicting movie revenues based on various features such as release date, genre, director, and more.

To train and evaluate the XGBoost regression model, we followed these steps:

- 1) Load the movie dataset and preprocess the data by encoding categorical features using `LabelEncoder` from `scikit-learn`.
- 2) Split the dataset into training and testing sets using `train_test_split` from `scikit-learn`.
- 3) Define a custom callback class `TrackR2Score` that inherits from `xgb.callback.TrainingCallback`. This class overrides the `after_iteration` method to calculate and store the R-squared score on the training set after each iteration during the training process.
- 4) Perform hyperparameter tuning using `GridSearchCV` from `scikit-learn`. We passed the `TrackR2Score` callback to the `XGBRegressor` instance used as the estimator in `GridSearchCV`, allowing us to track the R-squared score during the grid search process.
- 5) Train the final `XGBRegressor` model using the best hyperparameters found by `GridSearchCV`.
- 6) Evaluate the model's performance on both the training and testing sets using various metrics, such as R-squared score and Mean Absolute Percentage Error (MAPE).
- 7) Visualize the actual vs. predicted values for both the training and testing sets.
- 8) Plot the training R-squared score curve to observe the gradual improvement of the model during the training process.

The following plot shows the training R^2 score curve, which illustrates the model's gradual improvement during the training process: By leveraging the powerful XGBoost library and

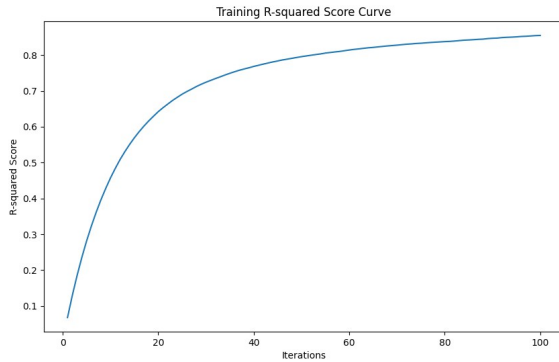


Fig. 5: Training R^2 Score Curve

implementing a custom callback function, we were able to

effectively train and evaluate a regression model for predicting movie revenues while monitoring the model's performance during the training process.

VIII. MODEL EVALUATION

We will evaluate the performance of our models using appropriate metrics such as Mean Squared Logarithmic Error ($MSLE$) and Coefficient of Determination (R^2). Taken together, these two metrics provide a comprehensive view of your model's performance. The R^2 score tells you how well your model is capturing the patterns in the data, while the $MSLE$ gives you a sense of the relative error in your model's predictions, with the advantage of being less sensitive to outliers and more appropriate for data with exponential growth trends (like budget and revenue in our case). Additionally, we will compare the performance of our model with baseline models to assess its effectiveness.

Model	R^2	$MSLE$
Linear Regression		
Training Set	0.6181	0.0053
Testing Set	0.6520	0.0051
Decision Tree		
Training Set	0.8310	0.0024
Testing Set	0.5994	0.0059
Bagging		
Training Set	0.8380	0.0023
Testing Set	0.7105	0.0042
Gradient Boost		
Training Set	0.8750	0.0016
Testing Set	0.7350	0.0040
XG Boost		
Training Set	0.8633	0.0018
Testing Set	0.7402	0.0041
Random Forest		
Training Set	0.8475	0.0022
Testing Set	0.7235	0.0041

TABLE II: Model Evaluation Result

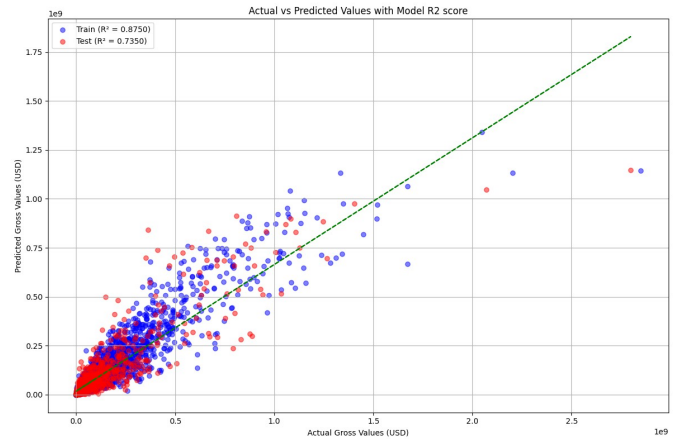


Fig. 6: Gradient Boosting Test Results

IX. USER INTERFACES FOR MOVIE REVENUE PREDICTION

We have developed two user-friendly interfaces for producers to access our movie revenue prediction model: a Command Line Interface (CLI) and a Streamlit Web Application. Both interfaces allow users to input the 14 key parameters on which our XGBoost model, known for its accuracy and speed, has been trained.

A. Command Line Interface (CLI)

The CLI can be accessed via `main.py`. Producers can run `python main.py` and enter the required parameters to predict the revenue for their upcoming movie, as envisioned in our Problem Statement.

B. Streamlit Web Application

For those preferring a graphical interface, we have created a beautiful and easy-to-use Streamlit web application. This application can be accessed at **Movie Revenue Prediction**

Web App. It offers the same functionality as the CLI but with an intuitive, interactive interface that makes the prediction process even more straightforward

C. Revenue Prediction Categories

Both interfaces provide revenue estimates (in USD) within the following ranges:

- Low Revenue: 10M or less
- Medium-Low Revenue: 10M - 40M
- Medium Revenue: 40M - 70M
- Medium-High Revenue: 70M - 120M
- High Revenue: 120M - 200M
- Ultra High Revenue: 200M or more

This categorization allows producers to quickly gauge the potential financial performance of their movie based on the input parameters, whether they're using the CLI or the web application.

X. CONCLUSION

In our comprehensive analysis of movie revenue prediction models, Gradient Boosting and XGBoost emerged as the top

Squared Logarithmic Error (MSLE). These results underscore the robustness and accuracy of our approach.

We have successfully addressed our initial project goal: providing stakeholders with accessible software for accurate movie revenue prediction. Our model consistently places predictions within the correct revenue range bracket with reasonably high confidence, and within neighboring brackets with absolute confidence. Our model can serve as a valuable tool for filmmakers, offering helpful insights to support their decision-making process in the complex landscape of movie production and financing.

Looking ahead, we are committed to further improving our models' performance and generalizability. We plan to expand our training dataset, incorporating more diverse and extensive data. This approach will enhance the robustness of our models, enabling them to generalize even better across a wider range of scenarios in the dynamic film industry landscape.