FITE3010 X KAGGLE

TMDB Box Office Prediction

Created By:
Masood Ahmed (3035812127)
Muhammad Mubeen (3035831783)
Alexia Chan (3035930187)

SCORE & RANKING

Playground Prediction Competition

TMDB Box Office Prediction

Can you predict a movie's worldwide box office revenue?

Kaggle · 1,395 teams · 4 years ago

Overview Data Code Models Discussion Leaderboard Rules Team

Submissions

You selected 0 of 2 submissions to be evaluated for your final leaderboard score. Since you select
Kaggle auto-selected up to 2 submissions from among your public best-scoring unselected submi
evaluated submission with the best Private Score is used for your final score.

Submissions evaluated for final score

All Successful Selected Errors

Submission and Description

submission.csv
Error (after deadline) · 5d ago

submission.csv
Error (after deadline) · 5d ago

submission.csv
Complete (after deadline) · 5d ago

submission.csv
Complete (after deadline) · 5d ago

submission.csv
Complete (after deadline) · 5d ago

Score: 1.46101
Private score: 1.46101

UPLOADED FILES

submission.csv (100 KiB)

DESCRIPTION

Enter a description

0 / 500

SELECT FOR FINAL SCORE

Select this submission to be scored for your final leaderboard score

Score: 1.46101
Private score: 1.46101

Search

Overview Data Code Models Discussion Leaderboard Rules Late Submission ...

The private leaderboard is calculated over the same rows as the public leaderboard in this competition.
This competition has completed. This leaderboard reflects the final standings.

#	Team	Members	Score	Entries	Last	Solution
1	Arkadiy		0.68770	83	4y	
2	IIR		0.82986	62	4y	
3	rizaisik		1.00406	41	4y	
4	BOE_IOT_AIBD		1.13942	37	4y	
			1.16173	38	5y	
			1.21933	97	5y	
			1.36288	123	4y	
8	Valeska K		1.38916	24	4y	
9	Qiwei Qian		1.47674	23	5y	
10	Dolly Zhang		1.48070	39	4y	
11	Kamal Chhirang		1.51056	79	4y	
12	ORAX		1.59725	28	5y	

us:

Score: 1.46101

Private score: 1.46101

INTRODUCTION

IMPORTING LIBRARIES

Category	Libraries
Data Manipulation	pandas, numpy
Date and Time	datetime, time, calendar
Data Visualization	matplotlib.pyplot, seaborn

Category	Libraries
String Manipulation	re
Statistical Analysis	scipy.stats, math, statistics
Machine Learning	sklearn, xgboost, lightgbm
Model Interpretability	shap

DATA INSPECTION

No. of rows and columns

- 23 Columns

Non-null Content Count

Data Type

- e.g. object, int64

Missing Values

- To be handled

EXPLORATORY DATA ANALYSIS & FEATURE ENGINEERING

ANALYZING 23 VARIABLES

(Correspond to the 23 columns including revenue)

Analysis Flow

1. **Pre-processing**
   - Missing Values, Messy Structure
2. **Univariate Analysis**
   - Mean, Median, Frequency
3. **Bivariate Analysis** to revenue
   - Correlation -> store positively correlated features

Analysis Results

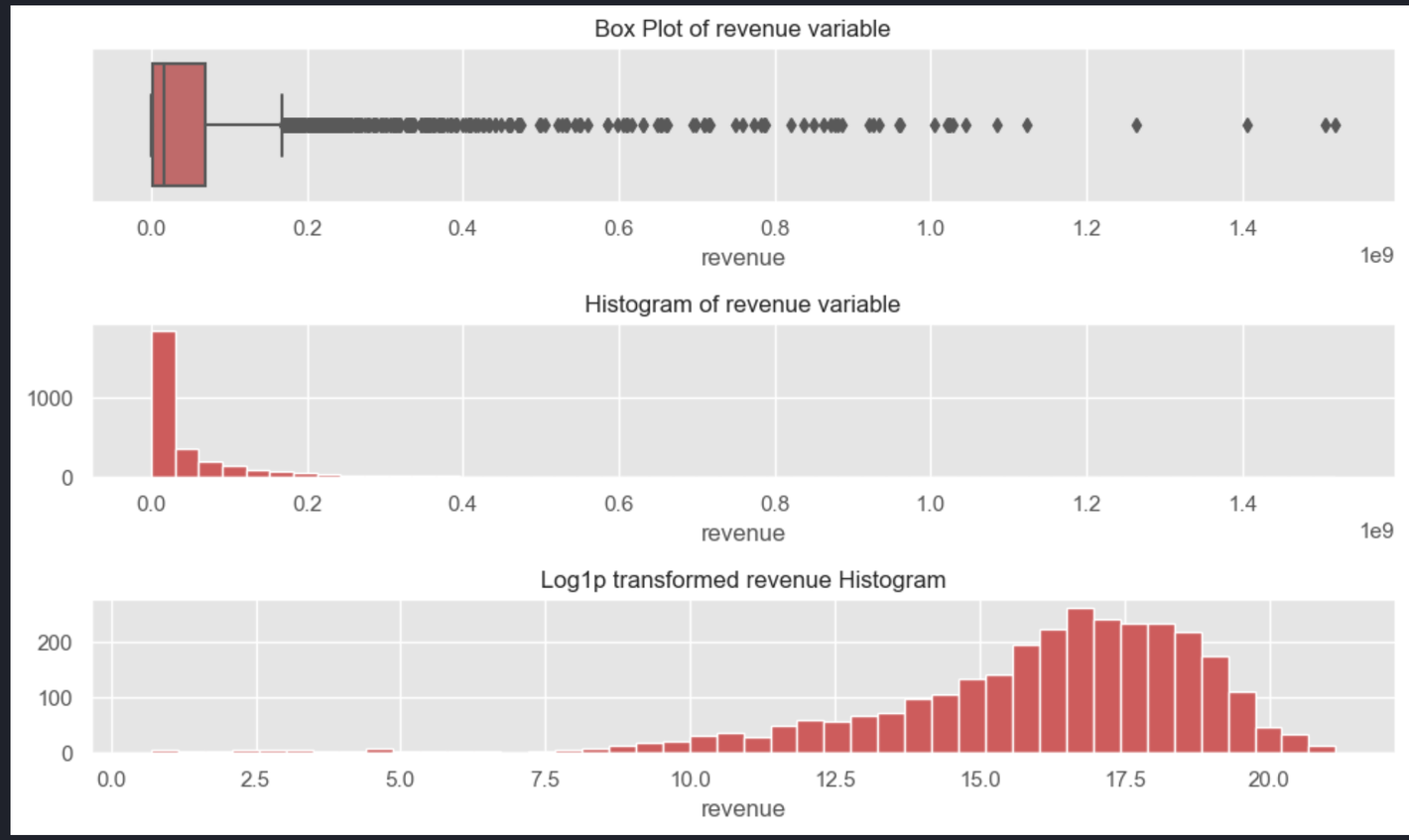
1. **train_features**
   - features we will use
2. **log_features**
   - features to be log-transformed
3. **cols_to_drop**
   - columns to be dropped

TRAIN_FEATURES

- Features to be used for modeling
- **Result:**
  - ['belongs_to_collection', 'has_homepage', 'originally_english', 'topStudio', 'usa_produced', '1960s', '1970s', ..., 'topLeadEditor']
- **Notable features:**
  - **extra features added:**
    - IMDb rating

LOG_FEATURES

- Some features are more manageable after a log-transformed
  - e.g. skewed data
- **Result:**
  - ['revenue', 'budget_processed', 'genre_rank', 'num_genres', ... , 'runtime_to_year_ratio']



COLS_TO_DROP

- No valuable insights -> Drop
- **Results:**
    - ['imdb_id', 'original_title', 'overview', 'popularity', 'poster_path', 'status']

ADDITIONAL FEATURE ENGINEERING

BUDGET & REVENUE DATA OPTIMIZATION

- **Issue:**
  - Excessive garbage data
- **Earlier Approach:**
  - Bad values replaced with median budget.
- **New Approach:**
  - Average budget and revenue of involved studios calculated and used.
- **Outcome:**
  - Distributions improved, yet still contain lower values.
- **Next Steps:**
  - Stick to provided data, despite potential inaccuracies. Desire to retrieve true revenues and budgets.

ADJUSTING FOR TIME AND INFLATION

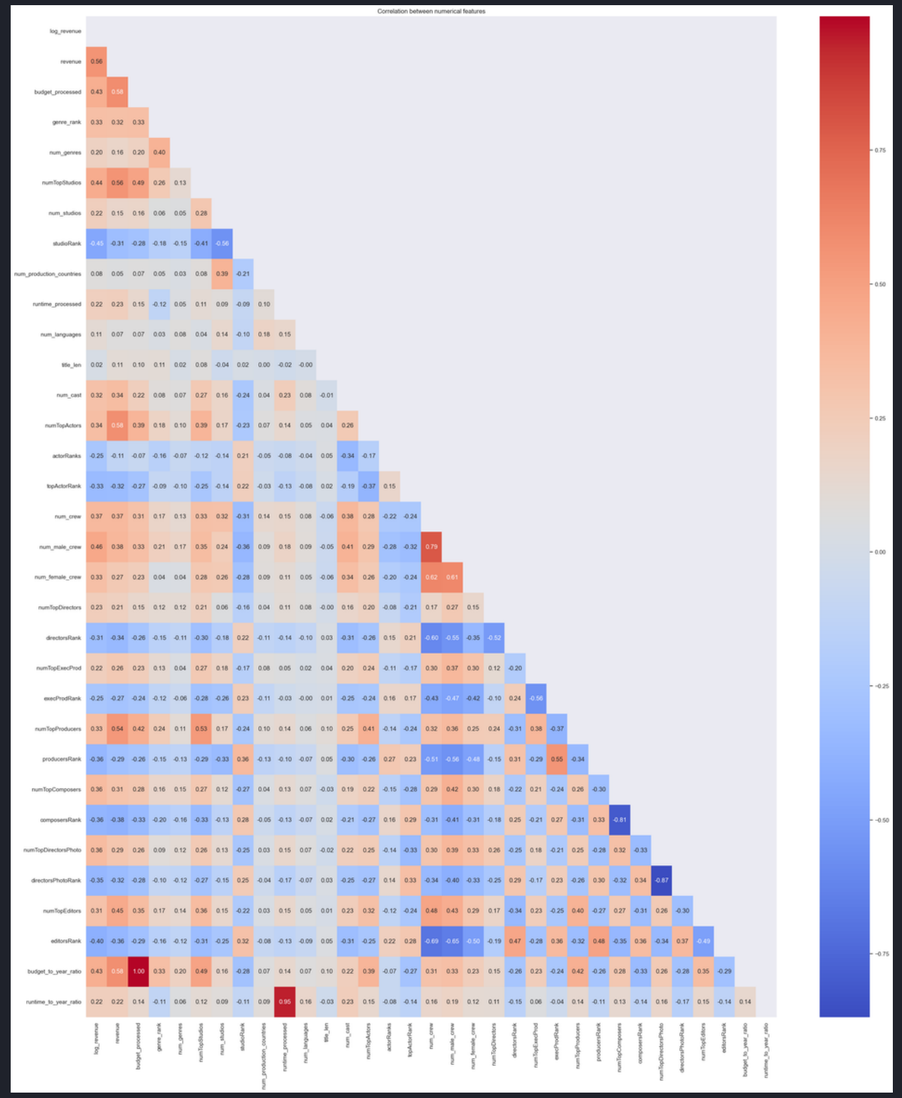
- **Issue:**
  - Dataset spans from the 1960s
    - needs time and inflation adjustment.
- **Solution:**
  - Introduce two features
    - budget/year ratio
    - runtime/year ratio.
  - Outcome
    - Capture significant industry changes over decades.
- **Goal:** Enhance model accuracy.

FEATURE SELECTION

- **Action:**
  - Dropped 5 non-contributing columns
- **Data Preparation:**
  - Applied log transformation to all numerical features in the training set -> normalize the distribution.
- **Result:**
  - Final feature set consists of 63 distinct variables.
- **Next Step:**
  - Examine the correlation matrix of all numerical features to further understand their relationships.

CORRELATION

- **Finding:**
  - Budget is highly correlated with revenue.
- **Issue:**
  - Significant multicollinearity observed in data.
- **Adjustment:**
  - Shifting from linear regression due to data complexity.
- **Next Step:**
  - Select an algorithm resilient to multicollinearity.



MODELING

SET-UP

- **Data Preparation:**
  - Split data into **80% training** and **20% testing** sets.
- **Tools:**
  - Developed algorithm_pipeline, evaluate, and metrics **helper functions** for **model tuning and evaluation**
- **Next Step:**
  - Apply these functions in model building and assessment.

#1 Random Forest

Predicts based on the average across a number of individual decision trees

#2 Extra Trees

Predicts using random values for splits in decision trees

#3 XGBoost

A gradient boosting framework known for speed and efficiency

#4 LightBGM

A gradient boosting framework known for speed, efficiency, and low memory usage

RANDOM FOREST

- **Baseline:**
- **Grid Search Hyperparameter Tuning**
- **Feature Importance:**
  - Most Significant: Budget and budget-to-year ratio
  - Other key features: actor ranks, top studio, number of male crew, and title length.
- **SHAP Values:**
  - Utilized to interpret the model. High budget values and presence of a top studio are strong revenue indicators.
- **Outcome:**
  - Encouraging results providing good accuracy.

```
    print('Base Model:')
    base_accuracy = evaluate(rfr_base_model, X_test, y_test.values)
    print()
    print('Model after Tuning:')
    rfr_best_model = rfr.best_estimator_
    best_accuracy = evaluate(rfr_best_model, X_test, y_test.values)

    print('Improvement of {:0.2f}%.'.format( 100 * (best_accuracy - base_accuracy) / base_accuracy))
```

2]

```
Base Model:
Average Error: 1.0958
Accuracy = 91.601%

Model after Tuning:
Average Error: 1.0705
Accuracy = 91.738%
Improvement of 0.15%.
```

```
    y_pred = rfr_best_model.predict(X_test)
    print('Random Forest Model (After Tuning) Metrics:')
    metrics(y_pred, y_test.values)
```

3]

```
Random Forest Model (After Tuning) Metrics:
Mean Squared Error:  2.394
Root Mean Squared Error:  1.5473
Mean Absolute Error:  1.0705
Test Set Accuracy (from Mean Absolute Percentage Error):91.738%
```

EXTRA TREES

- **Baseline**
- **Grid Search Hyperparameter Tuning**
- **Feature Importance:**
  - Similar to Random Forest
  - Most Significant: top studio
  - Notable addition: the number of top directors of photography.
- **SHAP Values:**
  - Movies with a top studio -> much higher revenue.
- **Outcome:**
  - Slightly less effective than Random Forest but still yields similar results.

```
    base_accuracy = evaluate(et_base_model, X_test, y_test.values)

    et_best_model = et_model.best_estimator_
    best_accuracy = evaluate(et_best_model, X_test, y_test.values)


    print('Improvement of {:0.2f}%.'.format( 100 * (best_accuracy - base_accuracy) / base_accuracy))
```

```
Average Error: 1.1155
Accuracy = 91.422%
Average Error: 1.0977
Accuracy = 91.506%
Improvement of 0.09%.
```

```
    y_pred = et_best_model.predict(X_test)
    print('Extra Trees Model (After Tuning) Metrics:')
    metrics(y_pred, y_test.values)
```

```
Extra Trees Model (After Tuning) Metrics:
Mean Squared Error:  2.5047
Root Mean Squared Error:  1.5826
Mean Absolute Error:  1.0977
Test Set Accuracy (from Mean Absolute Percentage Error):91.506%
```

XGBOOST

- **Baseline**
- **Grid Search Hyperparameter Tuning:**
  - Start: worst baseline performance
  - After tuning: achieved accuracy on par with the first two models
- **Feature Importance:**
  - Similar to previous models
  - Most Significant: number of directors of photography.
- **SHAP Analysis:**
  - Key predictor: budget-to-year ratio
  - No. of top directors of photography is less important than initially suggested by the model.

```python
print('Base Model:')
base_accuracy = evaluate(xgb_base_model, X_test, y_test.values)
print()
print('Model after Tuning:')
xgb_best_model = xgb_model.best_estimator_
best_accuracy = evaluate(xgb_best_model, X_test, y_test.values)


print('Improvement of {:0.3f}%.'.format( 100 * (best_accuracy - base_accuracy) / base_accuracy))
```

```
Base Model:
Average Error: 1.1935
Accuracy = 91.002%

Model after Tuning:
Average Error: 1.0851
Accuracy = 91.588%
Improvement of 0.644%.
```

```python
y_pred = xgb_best_model.predict(X_test)
print('XGBoost Model (After Tuning) Metrics:')
metrics(y_pred, y_test.values)
```

```
XGBoost Model (After Tuning) Metrics:
Mean Squared Error:  2.4787
Root Mean Squared Error:  1.5744
Mean Absolute Error:  1.0851
Test Set Accuracy (from Mean Absolute Percentage Error):91.588%
```

LIGHTGBM

- **Baseline**
- **Grid Search Hyperparameter Tuning:**
  - Highest accuracy achieved
  - Minor improvements post-tuning
- **Feature Importance:**
  - Budget features remain critical
  - New high-ranking features:
    - e.g. genre rank, actor rank, and studio rank.
- **SHAP Analysis:**
  - Confirms the importance of custom rankings
  - SHAP values provide valuable insights into model predictions

```
    lgbm_base_model = LGBMRegressor()
    lgbm_base_model.fit(X_train, y_train.values.ravel())
    base_accuracy = evaluate(lgbm_base_model, X_test, y_test.values)

    lgbm_best_model = lgbm.best_estimator_
    best_accuracy = evaluate(lgbm_best_model, X_test, y_test.values)


    print('Improvement of {:0.2f}%.'.format( 100 * (best_accuracy - base_accuracy) / base_accuracy))
```

```
[LightGBM] [Warning] Auto-choosing row-wise multi-threading, the overhead of testing was 0.001750 seconds.
You can set `force_row_wise=true` to remove the overhead.
And if memory is not enough, you can set `force_col_wise=true`.
[LightGBM] [Info] Total Bins 2898
[LightGBM] [Info] Number of data points in the train set: 2399, number of used features: 62
[LightGBM] [Info] Start training from score 16.199708
Average Error: 1.0836
Accuracy = 91.720%
Average Error: 1.0602
Accuracy = 91.792%
Improvement of 0.08%.
```

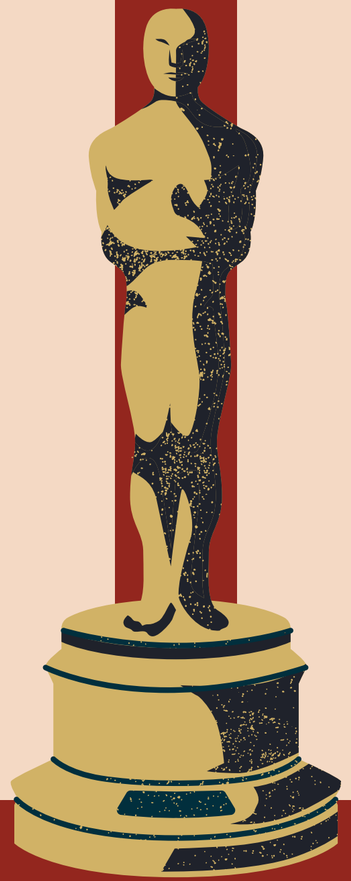
```
    y_pred = lgbm_best_model.predict(X_test)
    print('LightGBM Model (After Tuning) Metrics:')
    metrics(y_pred, y_test.values)
```

```
LightGBM Model (After Tuning) Metrics:
Mean Squared Error:  2.3365
Root Mean Squared Error:  1.5286
Mean Absolute Error:  1.0602
Test Set Accuracy (from Mean Absolute Percentage Error):91.792%
```

CONCLUSION

BEST-MODEL

- **LightGBM:**
  - RMSE of **1.5286**
  - accuracy of **91.79%.**



PERFORMANCE NOTE

- **All four** models achieved an **accuracy greater than 90%**
- a satisfactory outcome.

LIMITATIONS & IMPROVEMENTS

DATA

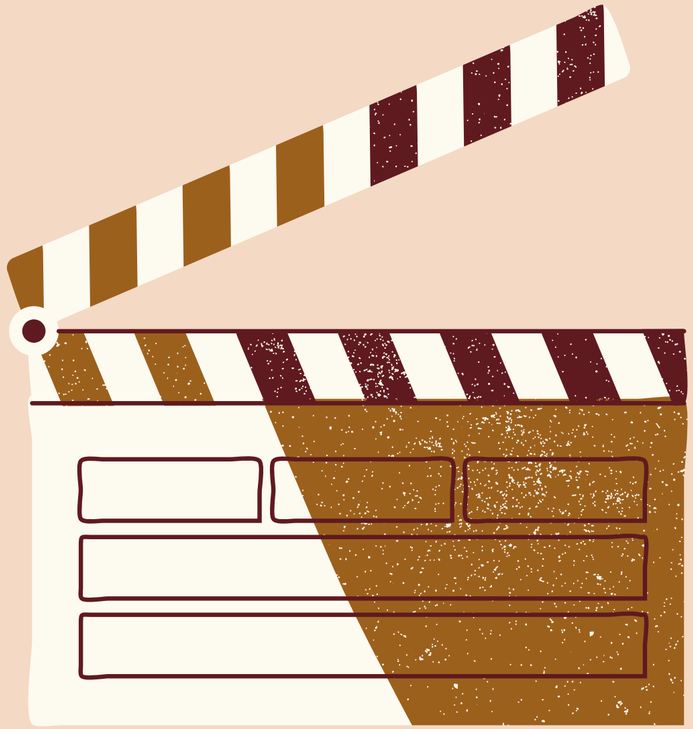
- **Data Limitations:**
  - Limited to 3000 data points from a completed Kaggle competition
    - resulting in a smaller training set
    - Numerous inconsistencies and missing values, particularly in budget and revenue columns.
- **Improvement:**
  - Access to more comprehensive and accurate data -> significantly improve model performance.

FEATURE ANALYSIS AND CREATION

- **Current Approach:**
  - Filled missing budget/revenue values using average of other films from the same studios.
- **Not Utilized:**
  - Several columns like title, overview, keywords, poster path were not fully utilized.
- **Improvement:**
  - Consider regression models to predict missing budget values, sentiment analysis on text-based columns, or neural networks for complex features.

MODEL SELECTION AND TUNNING

- **Current Approach:**
  - Chose outlier-resistant models due to the presence of many outliers and messy data.
- **Improvement:**
  - With more experience, could discover better modeling options and tuning approaches.



THANK YOU!