**Kaggle Project 7: TMDB Box Office Prediction**

**Prepared by: Dwaipayan Mukherjee (2211569, AIA AIML)**

**About the Project**:

In this dataset, we are provided with 7398 movies and a variety of metadata obtained from The Movie Database (TMDB). Movies are labelled with **id**. Data points include cast, crew, plot keywords, budget, posters, release dates, languages, production companies, and countries.

We are predicting the worldwide revenue for 4398 movies in the test file.

**Steps I have followed through the project:**

1. **Imports**:

I have imported the necessary libraries and the train, test and sample\_submission datasets as Pandas Dataframes.

1. **Understanding the datasets**:

I have first investigated the info of the columns in the train and test data. Then I have explored the summary statistics of the numerical columns of the datasets.

1. **Dealing with missing values**:

I have first explored the columns which have missing values. I found that there are two sets of columns having missing values: one with majority of its values missing and the other with only a small fraction missing. The columns **belongs\_to\_collection** and **homepage** have majority of their values missing, so I have dropped them from the train and test datasets. I observed that the column **runtime** had only two missing values in the train set and only four missing values in the test set. So, I have filled them manually from the internet. Similarly, the column **release\_date** had only one missing value in the test set and I have filled it manually from the internet as well. For the rest of nominal columns except **overview**, **poster\_path**, **tagline** and **Keywords** (which I have not used in revenue prediction), I have put in **“none”** in place of the missing values.

1. **Formatting the dates**:

I have formatted the dates in the datetime format of month-day-year, and created three new variables named **release\_year**, **release\_month** and **release\_day**. Now the maximum value of **release\_year** was found to be 2068, which is impossible. So I have created a function to keep **release\_year** not exceeding 2019 (the release of this competition).

1. **Exploratory Data Analysis**:

I have first explored the univariate distributions of **revenue**, **log revenue**, **budget** and **popularity**, **runtime** and **release\_year**. Then I have investigated the trends of **revenue**, **budget**, **popularity** and **runtime** over the years. Lastly I have explored the bivariate distributions of these variables with each other.

1. **Feature Engineering**:

Many features are in JSON format. First, I convert the features in JSON format to nominal format. Then, I am counting the occurrences in those features which I plan to use in the model, unless they are not redundant. Then I converted nominal data to numerical.

Budget has zero values for many movies including some high budget movies. Additionally, it does not make sense to have movies with 0 runtimes .So, I am imputing those zero values with mean.

1. **Preparing the data for fitting the models**:

I have trained the models on a selected list of features, namely, **release\_year**, **release\_day**, **release\_month**, **status**, **original\_language**, **budget**, **popularity**, **genres\_count**, **production\_companies**, **production\_countries**, **spoken\_language\_count**, **cast\_count**, **crew\_count** and **runtime**. I have separated the features and the target columns.

1. **Models**:

For each model, I have used a 10-fold cross validation and taken the ten-fold average **mean\_absolute\_error** as the ultimate metric for the models.

**[8.1] Random Forest Regressor**:

First I have used a Random Forest Regressor and the average MAE came out as **1.4328**. Upon investigation, the most important features came as **budget**, **popularity**, **release\_year**, **runtime**, **production\_companies**, **cast\_count**, **crew\_count**, **production\_countries**, **release\_month** and **release\_day**.

**[8.2] XG Boost Regressor**:

Upon using an XGBoost Regressor, I found the average MAE to be **2.8215** and the most important features to be **budget**, **popularity**, **cast\_count**, **production\_companies,** **crew\_count**, **spoken\_languages\_count**, **runtime**, **release\_year**, **original\_language** and **production\_countries**.

**[8.3] LightGBM Regressor**:

Upon using LGBMRegressor, I found the average MAE to be **1.4347**.

1. **Final Prediction on test set**:

Since, among the model I have tried, the Random Forest Regressor gave the lowest average MAE score (and hence performed best), I trained the RF model on the entire train dataset and used it to make predictions on the test set, saved as **submissions.csv** file.