MOVIE RECOMMENDATION SYSTEM

LETTERBOXD

Developed and Presented by Varshini Chellapilla for Springboard

01 PROJECT OVERVIEW

LETTERBOXD:
Letterboxd is a social media platform that provides film enthusiasts with a space to review films, create lists of movies, and find other fans to connect with. Other features include keeping track of one’s personal history of movies seen, a watchlist of movies that one wants to see in the future that can be constantly updated, and the chance to see what friends, critics and fellow fans are watching and reviewing. However, the platform currently does not have an official recommendation system that analyzes a user’s movie history and watchlist to provide recommendations of movies to watch and review.

PROJECT AIMS:
The goal of this third capstone project is to create a recommendation system based on collaborative filtering using data from Letterboxd and TMDB (The Movie Database) to provide Letterboxd users with movie recommendations based on past ratings and other calculated trends.

DATA COLLECTION:
Unfortunately, the Letterboxd API is, as of the date of this project documentation, in private beta. In lieu of this, Sam Learner uploaded a dataset with scraped, publicly-accessible Letterboxd ratings data of the top 4000 users on the platform in any given month on Kaggle. There are three datasets available: Users, Ratings, Movies.

The dataset can be found on Kaggle at this link.

02 DATA WRANGLING

ORIGINAL DATASET:
Ratings — Contains information on movie ratings from top 4,000 Letterboxd users. The ratings are based on a scale of 0 to 10.

Users — Contains information on users represented in the *Ratings* dataset. Contained 8,000 unique users.

Movies — Contains information on every movie represented in the *Ratings* dataset, enriched with data from TMDB obtained by Mr. Learner on Kaggle. This is a dataset of ~285,000 rows.

FACETS OF DATA WRANGLING:
First, the three datasets were merged to form one large dataframe of over 11 million rows. The merging was performed keeping the user IDs and movie IDs as the connection point respectively.

Next, in order to properly carry out the various stages of data wrangling, the merged dataset was observed from a bird's eye perspective.

Rows that contained missing values were either dropped or filled in with statistically inferred solutions. If the cells containing missing values make up for 5% of the data in the column or less, those cells -- and their adjoining rows -- were dropped. Categorical columns with missing values were filled in using terms like "Unavailable."

Additionally, columns were cleaned, data types were ensured to be correct and so were any rating scales.

0 3 EXPLORATORY DATA ANALYSIS

AIM:
The goal of exploratory data analysis was to determine the factors that affect user ratings and to observe other relationships between the features of the dataset.

In order to do so, the following goals were developed at the very beginning of the process:

- Determine the distribution of ratings & number of reviews among movies, users, years, and genres
- Note the most and least popular movies as well as genres
- Observe changes in popularity of genres and movies over time

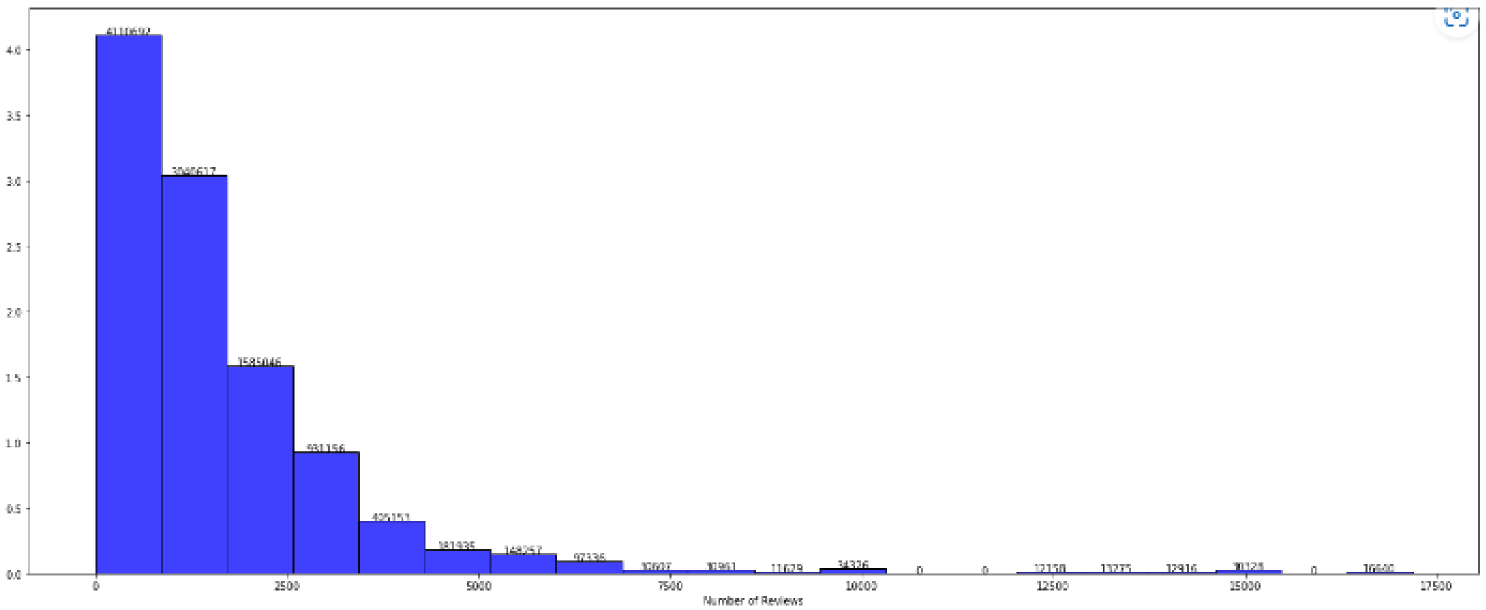
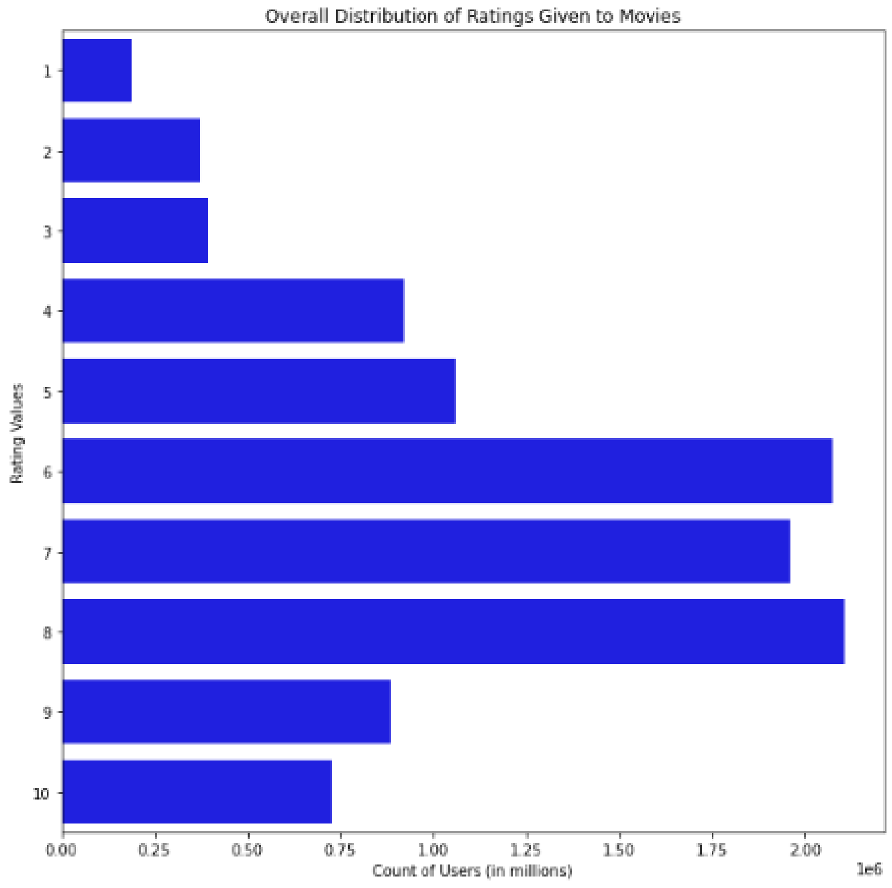


Figure 1: A histogram showing the distribution of the number of total reviews given by users in the dataset.

04

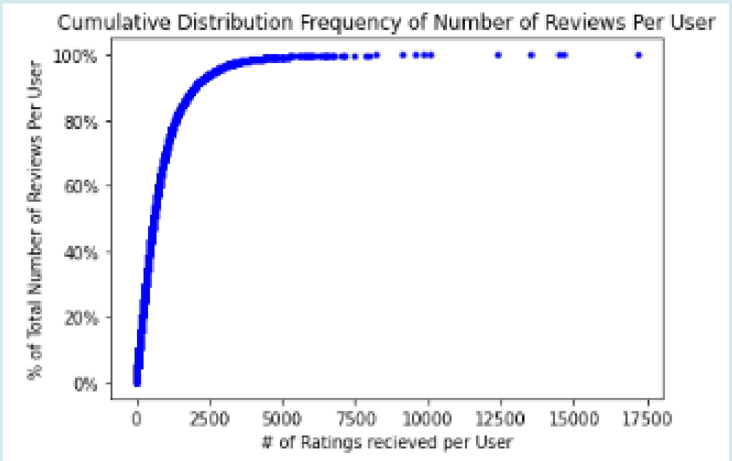
Figure 2: A bar plot of the overall distribution of ratings given to various movies by the users in the dataset.



As seen in Figure 2, the most common rating value given to a movie is **8**. Exactly 2,108,551 movies were given an 8. This is immediately followed by **6** with 2,077,362 movies. 725,248 movies have been given **10** and 185,622 movies have been given a **1**.

In terms of the number of reviews left by the users to all movies in the dataset, the data is right skewed, implying that most users don't leave relatively large numbers of reviews. Over 8 million users have left less than 2,500 reviews in total.

Figure 3: A probability plot showcasing the empirical cumulative distribution frequency of the number of reviews per user by the ratings the user gives out for each movie.



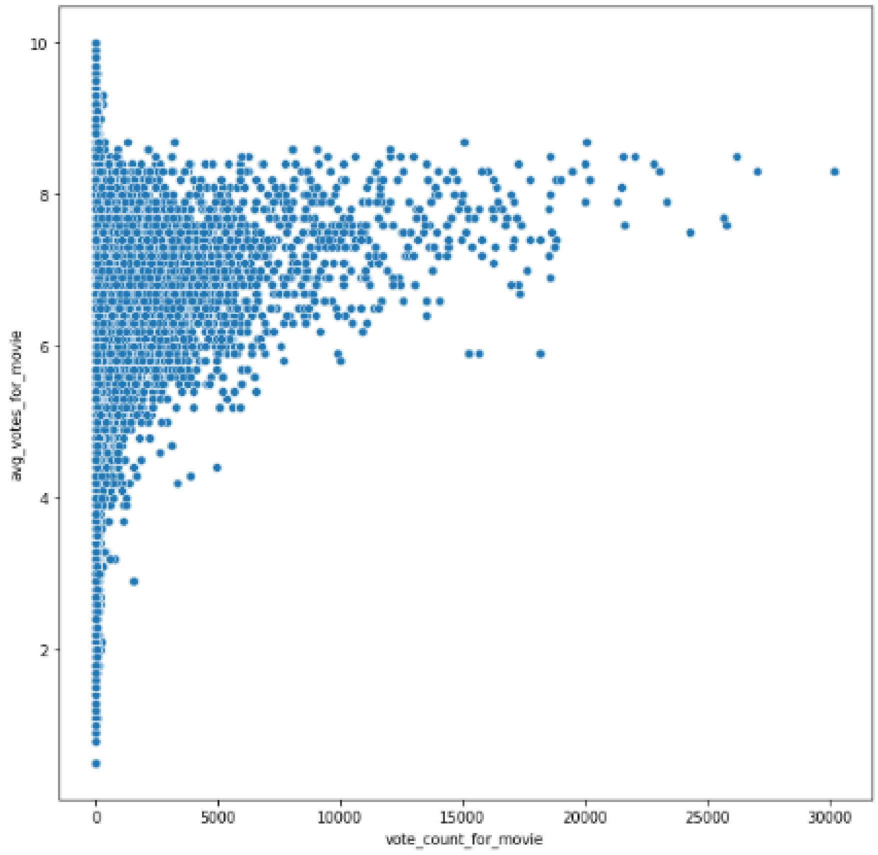


Figure 4: A scatter plot highlighting the distribution of average vote (on a 0-10 scale) received by each movie in the dataset and the total number of votes each was given. This plot is based on data from TMDB.

It was also found that many movies that received higher average vote from unique users on TMDB were given more number of votes. This supports a behavioral observation that showcases that audience members tend to only vote for movies they liked. Additional information noted was that of the distribution of the average rating received by movies based on the date they were released. Interestingly, movies in the '30s, and between the '40s and the '70s received the highest average rating by users.

The highest average user rating given to a movie was 7.42, and the lowest is 1.0 which is for a movie from the future (post 2022). Thus, we do not count it in our analysis. In fact, it was considered better for the algorithm to remove any movies that had a release date beyond 2022.

In regards to the countries of production for the movies in the dataset, the United States ranked the highest with 68,143 unique movies. It was followed by the United Kingdom, France, Germany, Canada and India.

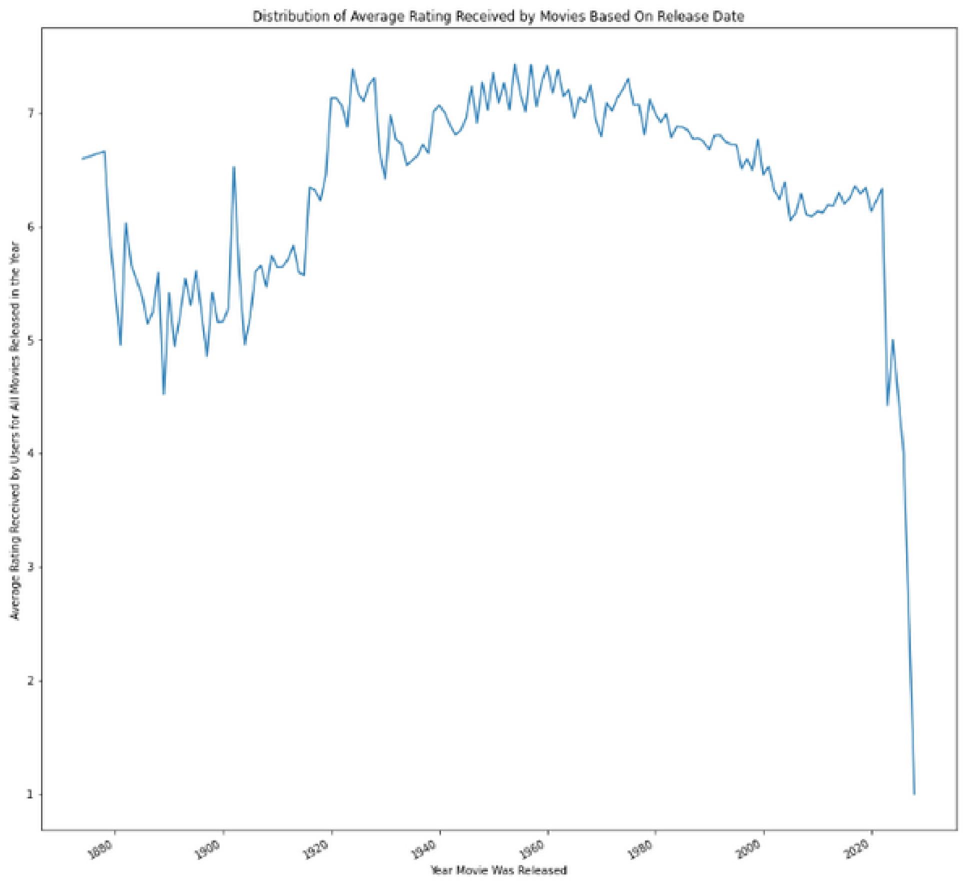


Figure 5: A line plot showing the distribution of average ratings received by movies based on their date of release.

Lastly, the top ten most popular movies in the dataset can be seen below in Figure 6.

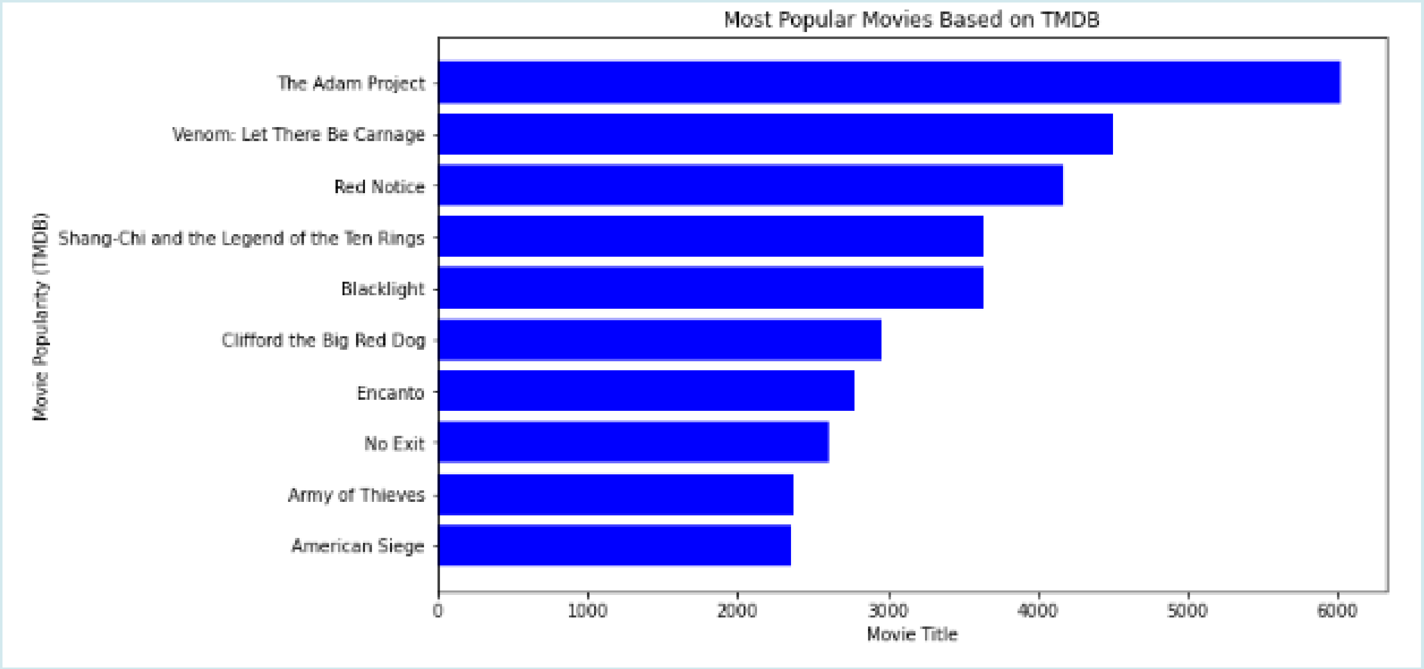


Figure 6: A horizontal bar graph with the top 10 most popular movies int eh dataset (according to data from TMDB).

05 DATA MODELING

Before modeling, the dataset was preprocessed to resolve categorical columns and remove any unnecessary rows that could burden our large dataset. The dataset was then spliced to create a dataset with only numeric data on the users, movies, and user ratings to each movie.

We have chosen to design a user-based collaborative filtering model.

Collaborative filtering is a type of recommendation system that filters information by learning the preferences, in this case ratings, of other users. The assumption taken by the model is that people who like similar movies will continue to like the same movies. If two people have similar preferences on some things, chances are they have similar preferences on other things as well.

Our system is also model-based which means that it employs a machine learning algorithm such as SVD or NMF. Additionally, the system being user-based means that it identifies other users similar to the inputted user and then calculates the desired rating.

Our dataset is extremely big! It contains 10 million rows. Prior to our EDA phase, the project was moved from Jupyter Notebooks to Google Colaboratory Notebooks to take advantage of the higher RAM afforded by cloud computing. However, this was still not enough. Processing was taking a very long time (over an hour at some parts). Thus, random sampling was conducted to select rows to use in our modeling dataset. The maximum number of rows accomodated was 10,000.

05

Going forward, we used the Surprise package. It was built specifically to create and analyze recommendation systems for rating data.

First, the dataset was split into training and testing sets in order to avoid overfitting on the same dataset when we later train and test the dataset.

The following models were attempted before choosing the model with the best performance and accuracy: Single Value Decomposition (SVD), SVD++, K-NEarest Neighbors (KNN) Basic, KNN With Means, Non-negative Matrix Factorization, and CoClustering.

Cross validaiton was used and the RMSE (Root Mean Squared Error) scores were calculated for each one to determine which model would be the best one. SVD++ had the lowest RMSE scores and so it was selected.

The next step is hyperparameter tuning. Using GridSearchCV the best parameters for the model were predicted:

- n_factors = 20
- n_epochs = 50
- lr_all = 0.005
- reg_all = 0.4

Following this, a custom function was created to take a the predicted and tuned model as input and return a total of five movies from our sampled dataset.

07 FUTURE DIRECTION

In the future, I believe the model can be improved upon by combining the collaborative filtering system created here with a content-based filtering system that takes into consideration the genres, the actors, the overviews, and such as it makes recommendations. This hybrid recommendation system, weighted, would be more accurate and serve during more scenarios.

Additionally, if possible, I believe a method should be found to employ all 10 million rows of the initial dataset to create a wider range of availabilities, as well as a diverse collection of movies and users to learn from.