Capstone 3 Final Report: Amazon Movie Recommendation System

Problem Statement

Nearly every online service we use these days makes use of recommendation systems. Streaming services like Netflix recommend what to watch next, social media sites recommend people we may know, and online shopping sites recommend products we may want to buy. These online services are growing more and more popular, and as they do, it becomes ever more important for those services to develop effective recommendation engines to increase sales and customer satisfaction. The purpose of this project was to learn how to create recommendation systems.

By using data from over 4800 users and 206 movies, I created a model to estimate the rating a user would give to a movie they had not seen yet. I decided to use the surprise python library with different algorithms to determine which worked best.

Data Wrangling

This dataset required almost no wrangling. It was already cleaned with only whole numbers from 1-5 for ratings and NaN for movies a user had not rated yet. Every movie had been rated at least once and every user had rated at least one movie. I did not need to drop any rows or columns. The only change I made to the dataframe was to prepare it to be a surprise dataset by melting it so that user_id, Movie, and rating were the only 3 columns instead of user_id being rows and movies being columns. I did this after my exploratory data analysis as I was not sure when I initially loaded the data what method I would use for my model.

Exploratory Data Analysis

I began my analysis by generating a heatmap to determine the sparsity of the dataframe. The dataframe appears to be very sparse and depending on the results of the model I chose, might have needed to use a summary statistic as a fallback rating for movies that do not have enough ratings.

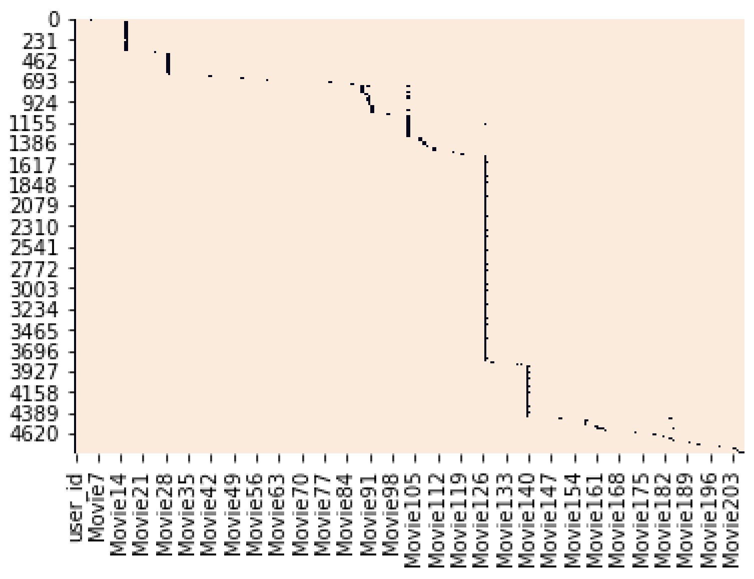

Figure 1: Heatmap to show sparsity of data

I plotted histograms of the mean, median, and mode of each movie's ratings. For the mode I needed to calculate the mean of each movie's mode rating as many have more than one result and it did not plot well. Due to this I decided to eliminate mode as a fallback option.

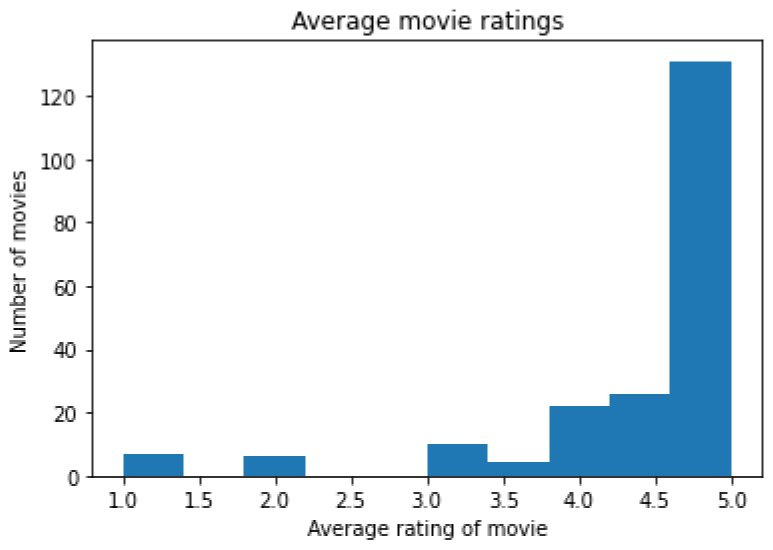

Figure 2: Histogram of average movie ratings

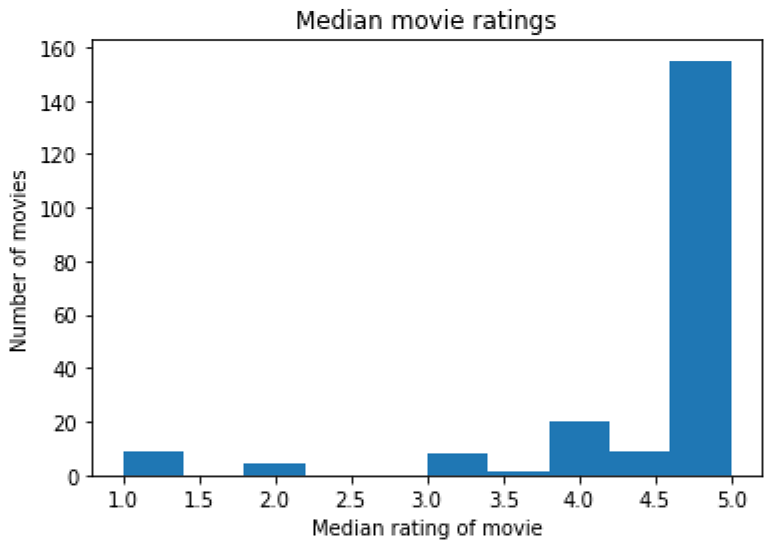

Figure 3: Histogram of median movie ratings

It appears that the vast majority of ratings left by users is 5 stars. I also plotted histograms of the mean and median ratings left by each user.

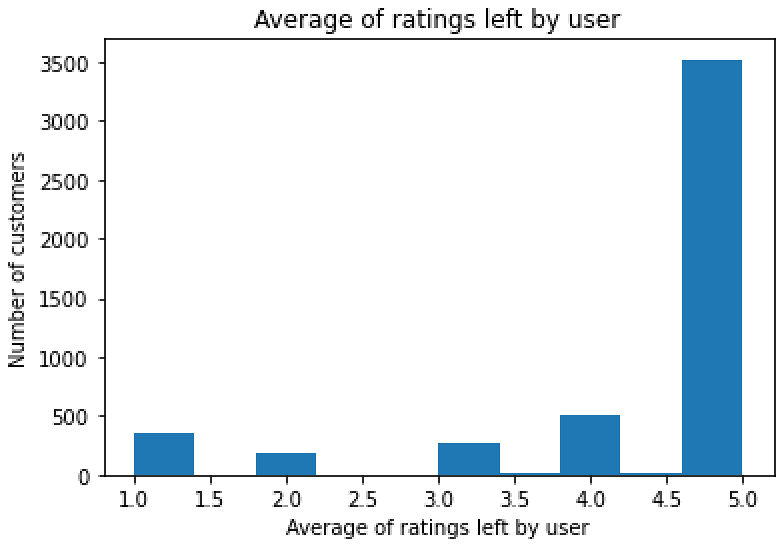

Figure 4: Histogram of average ratings left by user

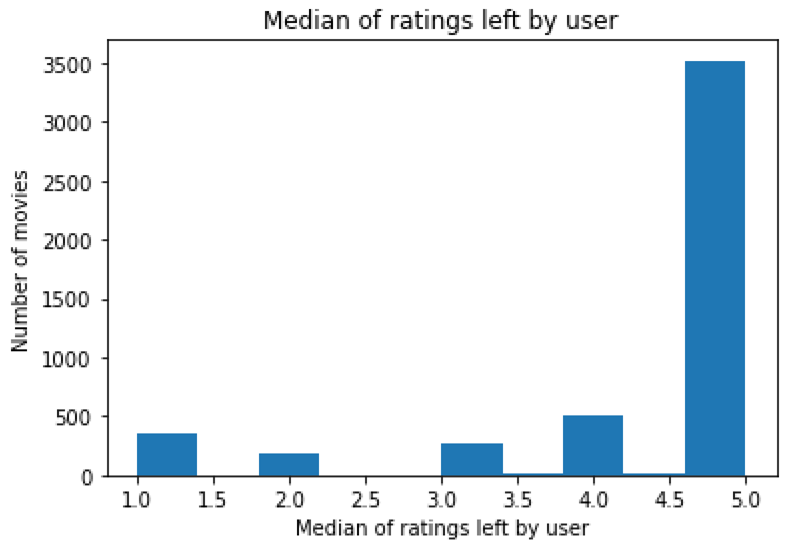

Figure 5: Histogram of median of ratings left by user

Preprocessing and Modelling

As I decided to use the surprise library to build my model, I imported the modules necessary to build an SVD model as that is the one I found mentioned most often for this kind of model. This is when I melted the dataframe into one that can be read by surprise. I split the data into 75/25 train/test sets, and instantiated, fit, and predicted an SVD model using 3-fold cross-validation.

Now that I knew how the library worked, I decided to test other algorithms. I used a for loop to run many different algorithms and output the metrics to a dataframe for easier reading. The SVD model performed best, although all of the algorithms had nearly the same RMSE.

I then remade the SVD algorithm and defined functions to gather the number of users who rated a movie and the number of movies rated by a user and output this with the predictions generated by the model to a dataframe so I could plot the results.

Unfortunately the model overfits the data. Nearly all of the movies are not rated, and thus the model predicts almost all of them as having the lowest possible rating. A solution to this would be to gather metadata about movies and users to build a hybrid model. Failing this, Amazon can wait until more users leave ratings which would improve the results over time.

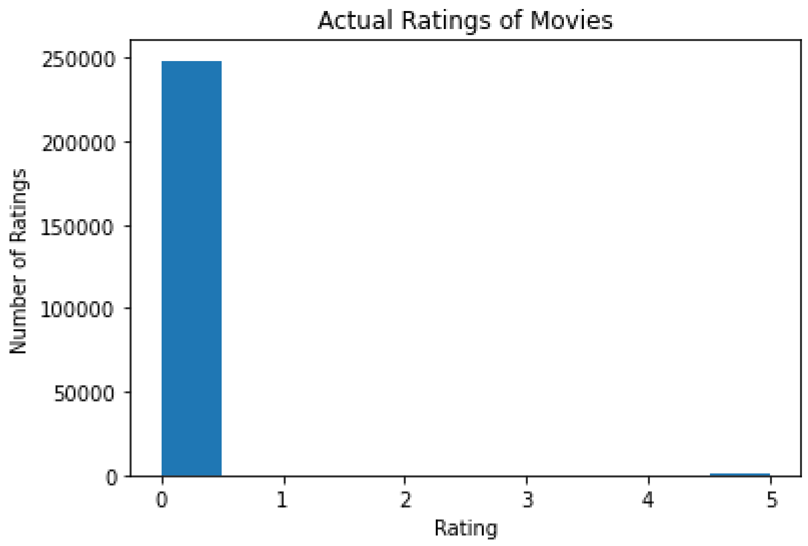

Figure 6: Actual Ratings of Movies

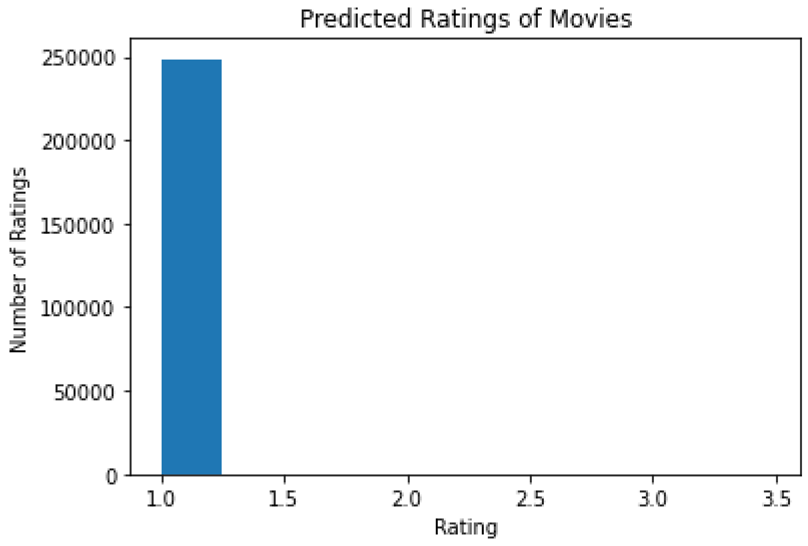

Figure 7: Predicted Ratings of Movies