MovieLens Recommendation Model

Jose Burgos

2026-02-18

Introduction

Machine learning models are widely used in data science to predict outcomes based off of existing data. Large datasets can provide a plethora of information that could either be directly used or wrangled in such a way to derive predictors. Statistical methods can then be used with those variables to build simple or complex models for real-world predictions and applications. One such application is in recommendation systems utilized by companies like Netflix to take information from their large databases of user metadata and predict what movies or shows a user may be interested in based on their viewing habits and preferences.

The first capstone project as part of the HarvardX Professional Certificate in Data Science is to utilize R and the publicly available MovieLens movie rating datasets to build a recommendation model that predicts a user's rating of a given movie. There are several datasets available ranging from 100,000 to 32,000,000 ratings that have been standards for machine learning research and projects. The goal of this capstone project is to build a recommendation system using one of the MovieLens datasets with 10,000,000 ratings. The dataset includes three files:

1. "movies.dat" with movie IDs, titles, and genres
2. "ratings.dat" with user IDs, movie IDs, ratings, and timestamps
3. "tags.dat" with user IDs, movie IDs, tags related to the rating, and timestamps

For the purposes of this project, only "movies.dat" and "ratings.dat" were joined into a dataset that was then split into a training set ("edx") and a final holdout test set with a ratio of 90/10 per the project prompt and provided code. The goal is to develop a model that uses the "edx" training set to achieve a root mean square error (RMSE) below 0.86490 when applied to the actual ratings in the final holdout test set.

This report outlines the data cleaning, exploration, and visualizations used to gain insight into variables that could be used to develop a linear regression model. The "edx" training set is then further divided into a training set and a developmental test set in a 90/10 ratio separate from the final holdout test set. Promising variables from exploration are then incorporated one variable at a time with monitoring and discussion of RMSE performance to minimize errors while not overfitting with too many variables. The final model is then applied to the final holdout test set to check if the project goal has been achieved before discussing limitations and future work.

Methods and Analysis

The “edx” and final holdout test datasets provided by the course each contain six columns:

1. userId: integers unique to each anonymous rater. Each rater has rated one or move movies.
2. movieId: integers unique to each movie. Each movie has been rated one or more times.
3. rating: a number on a 5-point scale with half-point increments. This is the rating from a unique user rating a unique movie at a unique time.
4. timestamp: an integer that represents the number seconds passed since 12:00am January 1, 1970 on Coordinated Universal Time (UTC).
5. title: a character string that has the movie title with the release year in parentheses appended.
6. genres: a character string that contains a combination of any of 19 genres that describe the movie. Genres include Action, Adventure, Animation, Children’s, Comedy, Crime, Documentary, Drama, Fantasy, Film-Noir, Horror, IMAX, Musical, Mystery, Romance, Sci-Fi, Thriller, War, and Western. In cases where a film spans multiple genres, each genre is sorted alphabetically and separated by “|”.

These variables can either be directly used or wrangled to extract more information than an integer or string without additional joining of outside datasets. For the purposes of exploration, the “edx” training set is copied into an exploration set that includes new mutated columns that extract multiple pieces of information from existing columns. The new columns for exploration include:

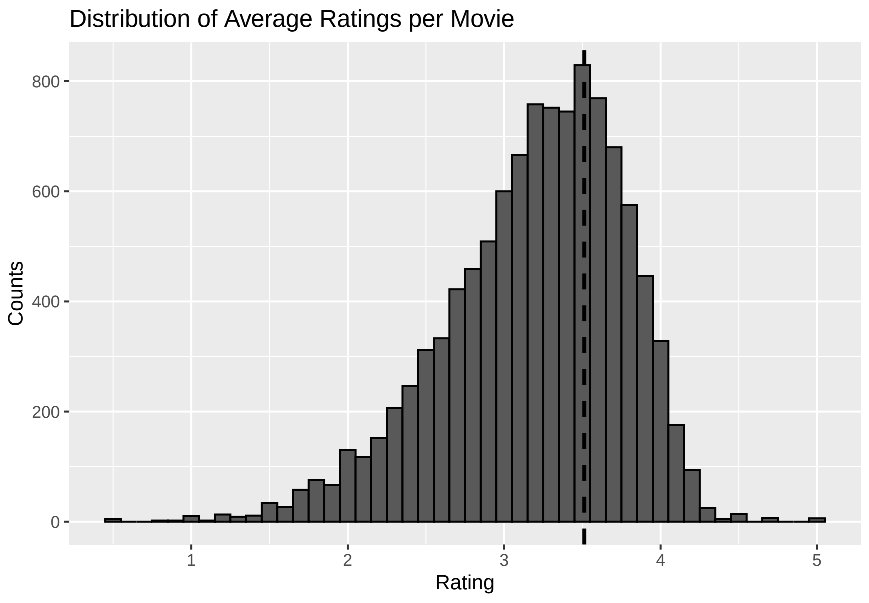
1. rating_week: a version of the datetime rounded to the nearest week.
2. rating_year: extraction of the year from the rating timestamp.
3. movie_year: extraction of the release year from the movie title.
4. years_since_release: integer representing the number of years between movie release and a rating.
5. Action: binary True or False if “Action” is part of the genres string.
6. Adventure: binary True or False if “Adventure” is part of the genres string.
7. Animation: binary True or False if “Animation” is part of the genres string.
8. Children: binary True or False if “Children” is part of the genres string.
9. Comedy: binary True or False if “Comedy” is part of the genres string.
10. Crime: binary True or False if “Crime” is part of the genres string.
11. Documentary: binary True or False if “Documentary” is part of the genres string.
12. Drama: binary True or False if “Drama” is part of the genres string.
13. Fantasy: binary True or False if “Fantasy” is part of the genres string.
14. Film_Noir: binary True or False if “Film-Noir” is part of the genres string.
15. Horror: binary True or False if “Horror” is part of the genres string.
16. IMAX: binary True or False if “IMAX” is part of the genres string.
17. Musical: binary True or False if “Musical” is part of the genres string.
18. Mystery: binary True or False if “Mystery” is part of the genres string.
19. Romance: binary True or False if “Romance” is part of the genres string.
20. Sci_Fi: binary True or False if “Sci-Fi” is part of the genres string.
21. Thriller: binary True or False if “Thriller” is part of the genres string.
22. War: binary True or False if “War” is part of the genres string.
23. Western: binary True or False if “Western” is part of the genres string.

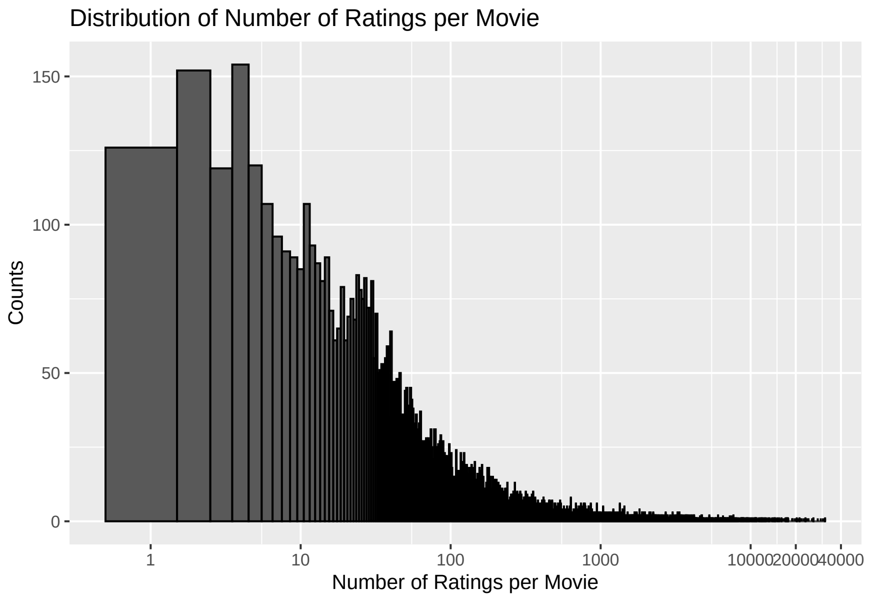
The general approach to this project is to use linear regression with the effects of factors added one at a time to minimize the root mean square error (RMSE). RMSE is calculated by taking the difference between predicted and actual ratings for each rating, squaring each residual, taking the average of all the squared values, and taking the square root of that value. In R, this error metric can be defined as:

```r
RMSE <- function(true_ratings, predicted_ratings){sqrt(mean((true_ratings - predicted_ratings)^2))}
```

This approach requires a starting model to build upon. A simple guess is to use the mean of a training set to predict all ratings in a test set. This can be used as a benchmark to see which factors might have enough of a bias from the mean to warrant the addition of a bias term to the initial guess of the training mean.

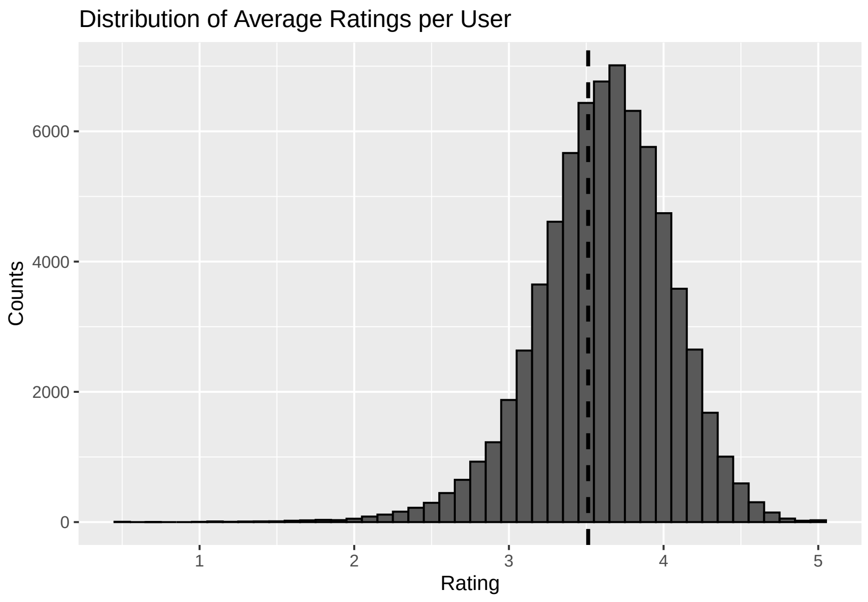
The first variable of potential bias is the movies themselves - how well does any given rating of a movie compare to the average rating of that same movie across the entire training set? One way to visualize this is to make a histogram of the average ratings of each movie and compare them to the "edx" set mean, as well as a histogram of how frequent movies with low numbers of ratings compare to those with high numbers of ratings:

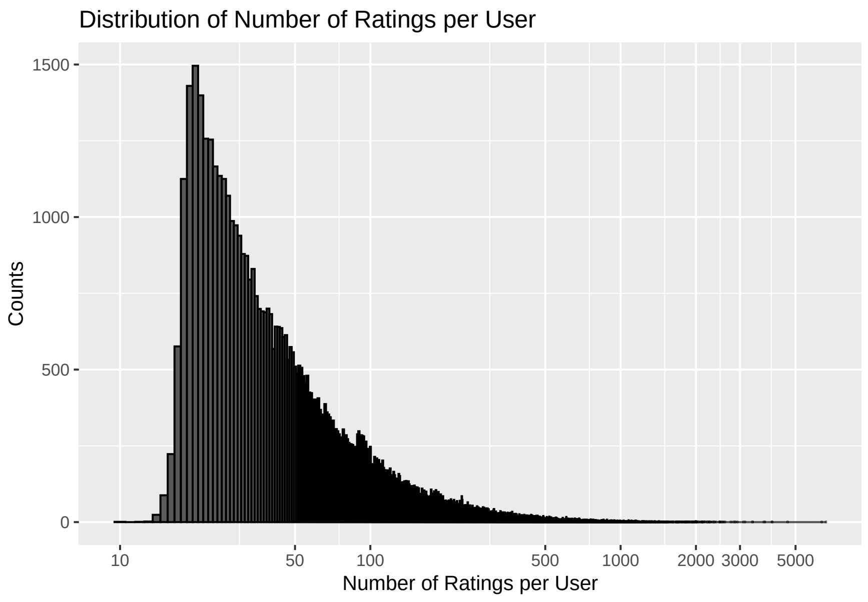




Average movie ratings are non-normal but smooth in distribution and skewed negative from the “edx” mean. This implies a potential effect from movieId that could be corrected with a bias term. When numbers of ratings per movie are plotted on a log scale, it is clear that a fraction of movies have thousands of ratings while a substantial number of movies have less than 100 ratings. Rating frequency of a movie may be related to how critically acclaimed it is. This discrepancy should be taken into account in the model.

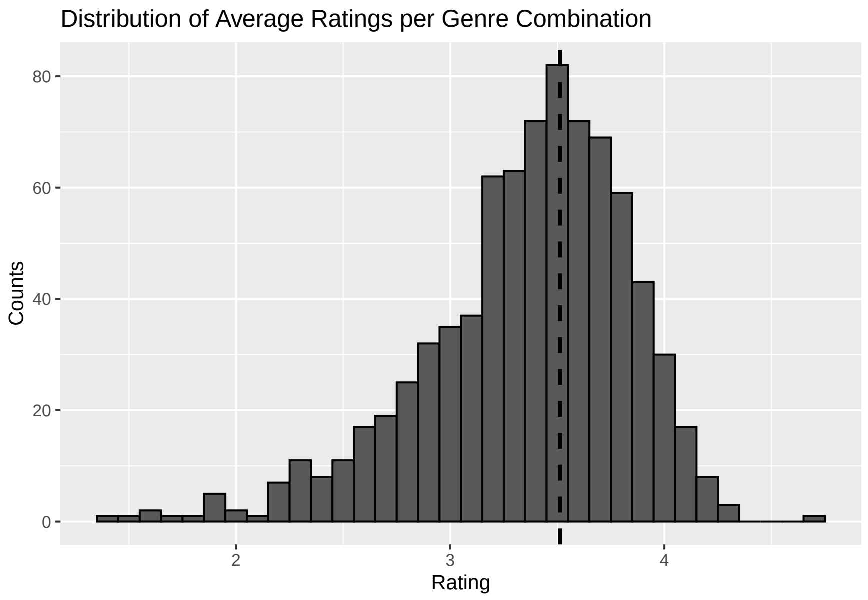
Another variable of interest is the identity of the user - how well does a user’s given rating of a movie compare to the average movie rating of the user of all of their rated movies across the entire training set? It is possible that some raters are more generous and some are more critical with ratings. This bias can be visualized with a histogram of the average ratings of each user compared to the “edx” mean, as well as a histogram of how frequent users with low numbers of ratings compare to those with high numbers of ratings:





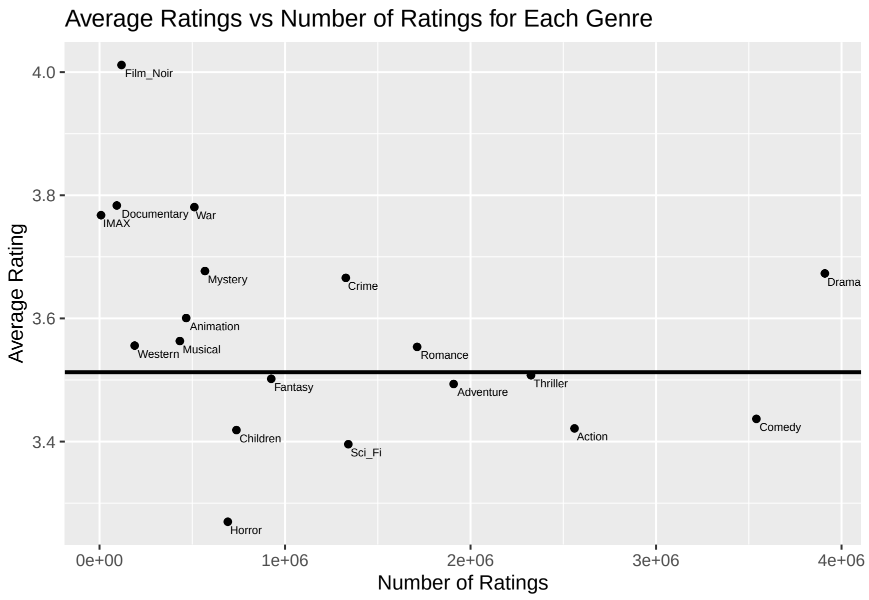
Average user ratings are visually normal but shifted positive from the “edx” mean. This implies a potential effect from userId that could be corrected with another bias term. When numbers of ratings per user are plotted on a log scale, it is clear that a subset of users have thousands of ratings while many users have less than 100 ratings. This discrepancy should also be taken into account in the model.

Another raw categorical variable from the provided datasets is the genre column that show the combinations of genres covered by a given film. Some movies have no genres listed, some have one genre, and some have multiple. Each unique combination has its own average rating that could be influenced by what kind of movies individual users prefer. This bias can be visualized with a histogram of the average ratings of each genre combination compared to the “edx” mean:



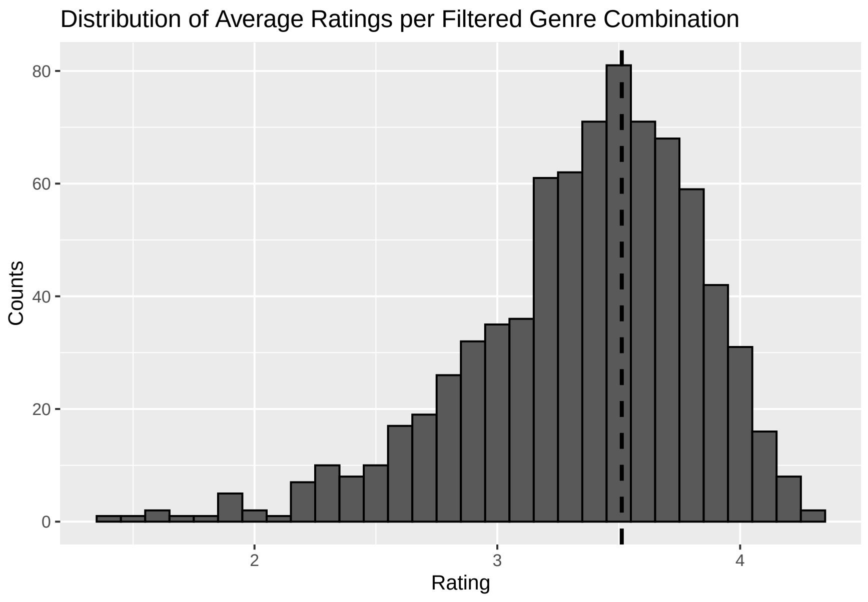
There is non-normal but relatively smooth distribution skewed negative from the “edx” mean. This implies a potential effect from genres that could be corrected with another bias term.

Theoretically, there are 524,288 possible combinations that could arise from the 19 individual genres. Most will be extremely niche and either associated with very few movies or none at all. Other combinations with one or two genres represented will be encountered far more frequently. However, it is possible that there are extra combinations in the training set created by the inclusion of less frequent genres. For the individual components of genres, the counts and average ratings for each can be visualized in a scatter plot:



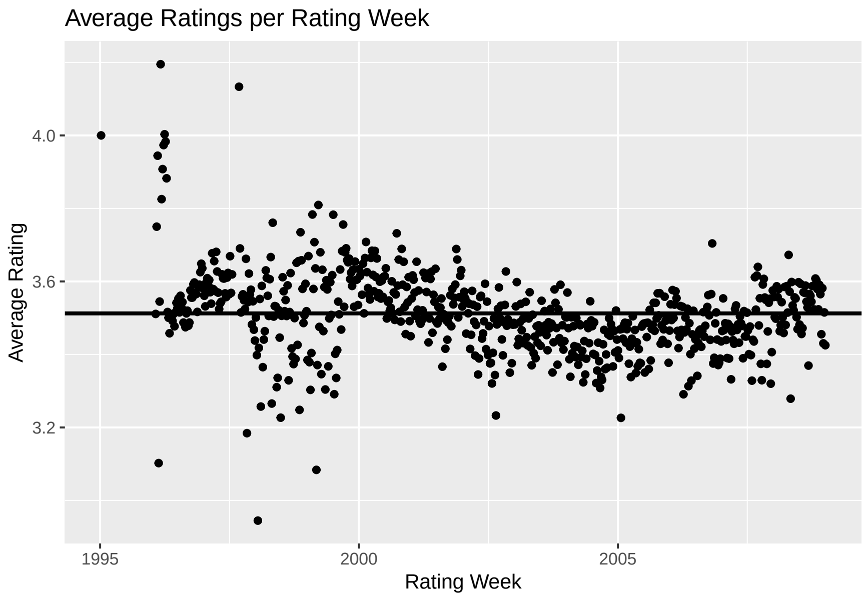
Some genres like IMAX (8181 ratings), Documentary (93066 ratings), and Film-Noir (118571 ratings) are relatively infrequent and have averages that deviate slightly more from the "edx" mean than genres with hundreds of thousands of ratings like Drama, Comedy, and Action. Infrequent genres can create niche genre combinations that could potentially overfit a model with unnecessary categories. This can be investigated by creating a filtered genres column that removes infrequent genres from the provided genre strings. This would reduce the number of unique genre combinations without sacrificing too much predictive data. One pitfall to filtering too many genres is that it is possible that movies that are very different from one another get lumped into a no-genre category that gets its own average rating.

Filtering with as few genres as possible to avoid this, IMAX is chosen as it is the most infrequent genre and is more of a movie format than an actual genre. This can be visualized with a histogram of the average ratings of each filtered genre combination and compare them to the "edx" mean and the raw genres counterpart from before:

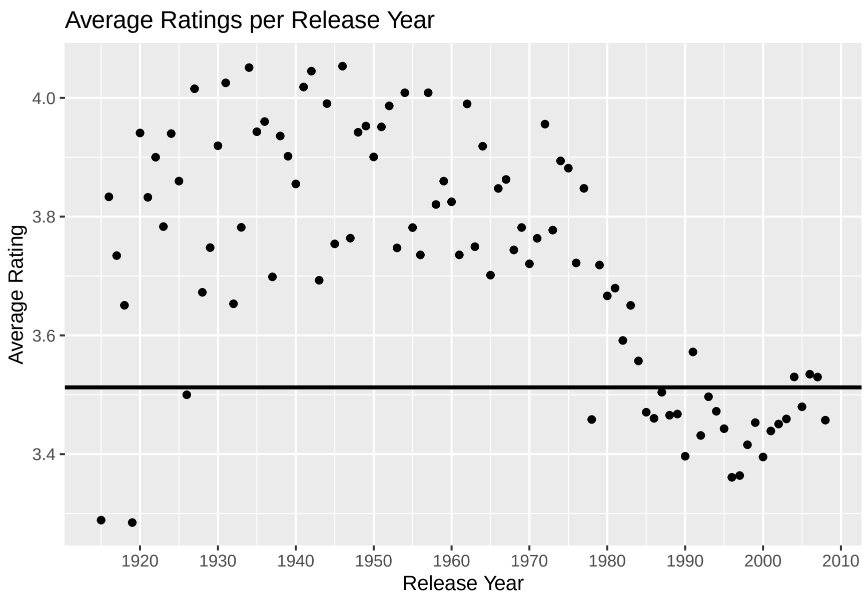


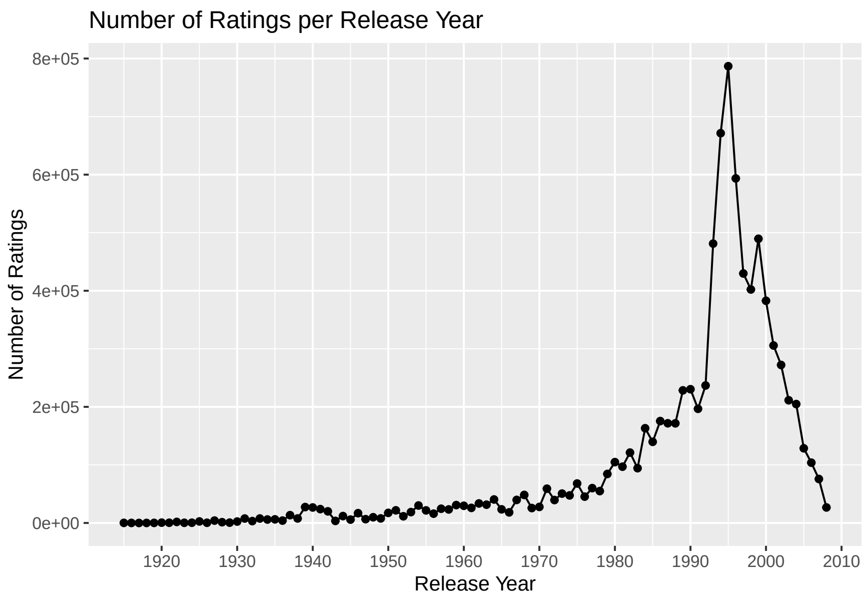
Filtering out IMAX alone eliminates a few of the outliers greater than the mean to make the distribution closer to normal, despite still skewing negative from the "edx" mean. It could also imply that IMAX as a movie format leads to better reviews. Development of the model will be done with only IMAX filtered out to investigate proof of concept.

Lastly, time dependence can be included in the model based off of the available information. The provided datasets include two sources of time variables: the timestamp of the rating and the release year of the movie. Plotting individual ratings as a function of time would not be effective due to ratings being in discrete half-point steps. To more effectively extract a rating time trend, it would be best to round each timestamp to the nearest week and average each week for time trend evaluation:

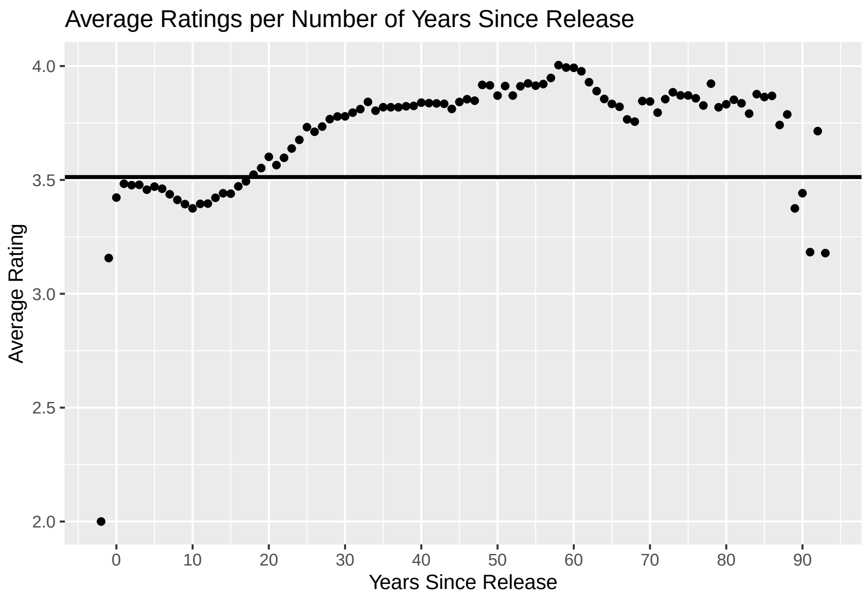


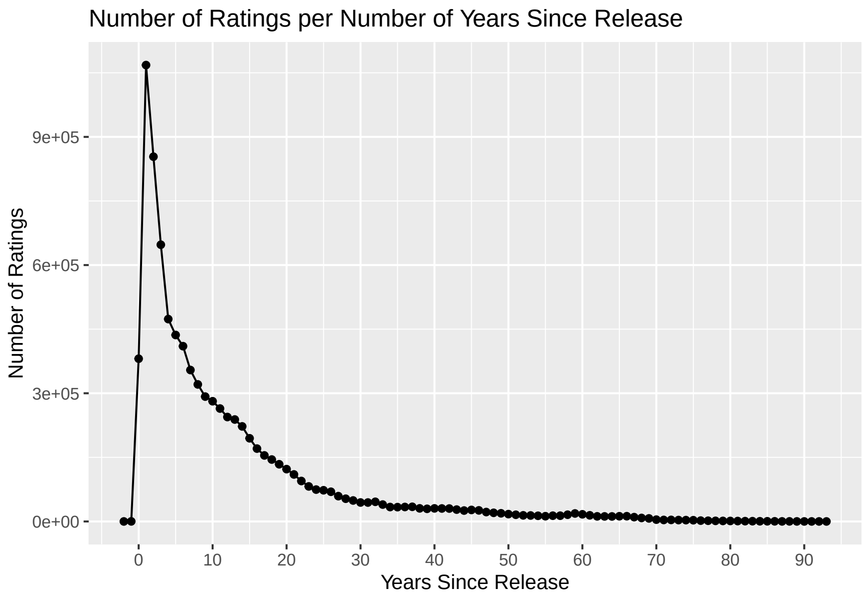
Most weeks have averages that are close to the “edx” average, but the first few years of ratings had a greater variability in average ratings. Movie release years can be extracted from the title strings to similarly plot average rating dependence on release year:





The release years show a clear non-linear trend where older movies tend to be higher rated than newer movies, though there are fewer movies from before 1980 than there are after. Another potentially useful measure of a time effect is to take the difference between a rating year and the release date:





This also shows a clear non-linear trend that combines the potential effects of the two available measures of time. However, there are a few instances where the rating timestamp came before the release year indicating some inconsistencies in release years and extreme outliers coming from reviews that were done prior to film release. In order to avoid overfitting with variables that depend on each other, it will be best to ignore the number of years since release and only use the rating timestamp and release year as time-based predictors for this model.

Based on the exploration thus far, it makes sense to build a model around a training mean and iteratively adding biases from the most promising predictors. Movie ID, user ID, genre combination (possibly filtered), movie release year, and rating timestamp have been identified as variables of most interest.

Before building a model, the original “edx” training set should be split into a training and validation test set to be used before the final application to the final holdout test set. It is important to note that the way the original MovieLens datasets are constructed and split by the course guarantees that every userId has at least one rating in both sets, and that every movieId had at least one rating in both sets. All non-duplicates between sets are added back to the “edx” training set solely for model development. The same methodology is used to split the “edx” set where 90% of the data remains in the training set and 10% of the data will become the validation test set, with non-duplicates between sets then being added back to the training set.

Results

A simple starting point for a model is to assume that the average rating of the entire training set is the predicted rating for the test sets and calculate its RMSE as a baseline model.

Model	RMSE
Course Goal RMSE	0.86490
Average	1.06005

This RMSE is far from the goal of <0.86490, but errors can be added to account for variation. Now that the baseline model of the training set has been established by assuming the average training set rating is the prediction for every rating, the first factor to add to the model is the movieId. Every movie in the validation and final holdout test sets has at least one rating in the training set, and the average rating for each movie can be used to evaluate how strongly each movie deviates from the training set average model. A grouped summary table is created with movieId's as an index, and each movie's average difference between actual rating and the prior model's prediction (training set mean) is used to calculate residuals to minimize and correct. The average rating summary table is then joined to the validation test set to incorporate the movie effect corrections to the prediction.

Model	RMSE
Course Goal RMSE	0.86490
Average	1.06005
Average w/ Movie Effects	0.94296

Incorporation of movie effects is effective in reducing the RMSE closer to the target. Like every movieId, every user in the validation and final holdout test sets has at least one rating in the training set. Its effect can be compounded onto the previous model by making a new grouped summary table with userId's as an index, and each movie's average difference between actual rating and the prior model's prediction (training average plus movie effect bias) to calculate residuals to minimize and correct. This second table is then joined to the validation test set to incorporate the user effect corrections to the prediction.

Model	RMSE
Course Goal RMSE	0.86490
Average	1.06005
Average w/ Movie Effects	0.94296
Average w/ Movie + User Effects	0.86468

The addition of user effects to the model is effective in reducing the RMSE closer to the target than the training average and movie effects on their own. Depending on how the "edx" set is randomly split into training and test sets, the RMSE already achieves the goal during testing. However, it is not guaranteed that the RMSE from testing will be the same as with the final holdout test set. The current model may not be robust enough with new data. It is worth exploring the effects of adding a few other predictors to the model, though too many more could lead to overfitting.

Since every movieId and title in the validation and final holdout test sets has at least one rating in the training set, each genre combination from the test sets will also have at least one match in the training set. The effects of genre combinations, whether as-is or filtered, can be incorporated into the model in the same way as with movieId's and userId's. Its effect can be compounded onto the previous model by making a new grouped summary table with genres as an index, and each genre combination's average difference between actual rating and the prior model's prediction (training average plus movie and user effects bias) to calculate residuals to minimize and correct. This third table is then joined to the validation test set (and eventually the final holdout test set) to incorporate the genres effect corrections to the prediction.

Model	RMSE
Course Goal RMSE	0.86490
Average	1.06005
Average w/ Movie Effects	0.94296
Average w/ Movie + User Effects	0.86468
Average w/ Movie + User + Genres Effects	0.86432

Incorporation of the genres effects is modestly effective in reducing the RMSE compared to the previous model. With 19 genres being combined into over 700 combinations, some combinations will be very niche with only a few representations, as opposed to the most common combinations with hundreds of thousands of representations. As discussed during exploration, it might make sense to trim off references to IMAX as it is infrequent and more of a format than a genre. In order to evaluate if filtering is a good idea, the "edx" training and test sets have a new filtered genre column added that removes IMAX from the character strings. Its effect can be compounded onto the previous model by making a new grouped summary table with filtered genres as an index, and each filtered genre combination's average difference between actual rating and the prior model's prediction (training average plus movie and user effects bias) to calculate residuals to minimize and correct. A filtered model removing just one infrequent genre can be applied to see if it improves RMSE more than using the genres as-is.

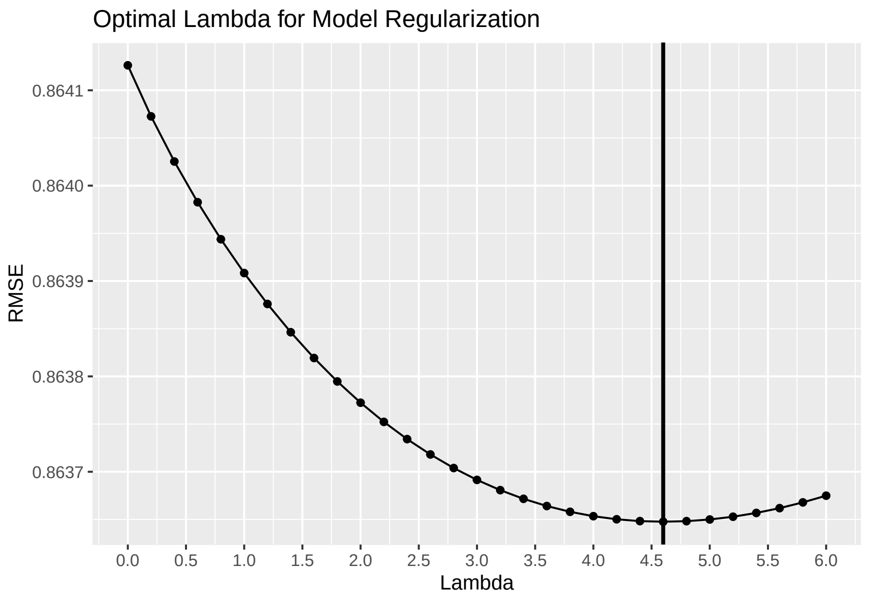
Model	RMSE
Course Goal RMSE	0.86490
Average	1.06005
Average w/ Movie Effects	0.94296
Average w/ Movie + User Effects	0.86468
Average w/ Movie + User + Genres Effects	0.86432
Average w/ Movie + User + Filtered Genres Effects	0.86433

Filtering genres for even one uncommon genre actually increases the RMSE in the validation test set. To avoid overcomplicating the model, the raw genres will be used as-is with references to IMAX. The addition of genre effects to the mean + user effects + movie effects model does make a small improvement, so it is kept in the model.

One more factor to consider is the time effects of when a movie was released and/or rated. Every user and every movie in the validation and final holdout test sets has at least one rating in the training set. This also means that every discrete release year as derived from the movie title has at least one match in the training set. The approach that has been used of joining tables of average bias per grouped movie, user, or genre combination can also be applied. However, the same guarantee does not necessarily hold for rating timestamps or time between rating and release. The same approach of creating a table grouped by movie years to join onto the model can be tried with the release year to see if there is further improvement.

Model	RMSE
Course Goal RMSE	0.86490
Average	1.06005
Average w/ Movie Effects	0.94296
Average w/ Movie + User Effects	0.86468
Average w/ Movie + User + Genres Effects	0.86432
Average w/ Movie + User + Genres + Release Year Effects	0.86413

The addition of the release year effects is effective in further reducing the RMSE. An additional technique to try is regularization of the model, which adds a penalty to large estimates of effect biases when the sample sizes used to calculate the estimates are small compared to other estimates. This should be effective as the prior exploration shows that movie, users, genres, and release years all have averages from samplings that range from a few counts to hundreds of thousands of counts. The average with movie + user + genres + release year effects model can be regularized all at once with cross validation of penalty lambda values.



Model	RMSE
Course Goal RMSE	0.86490
Average	1.06005
Average w/ Movie Effects	0.94296
Average w/ Movie + User Effects	0.86468
Average w/ Movie + User + Genres Effects	0.86432
Average w/ Movie + User + Genres + Release Year Effects	0.86413
Average w/ Regularized Movie + User + Genres + Release Year Effects	0.86365

Regularization using the optimal least squares penalty lambda identified though cross-validation is effective in further reducing the RMSE. The last readily available predictor to investigate is the rating timestamp, which is a continuous variable. Since timestamps or rounded timestamp weeks do not always have matches between the training and test sets, the approach of constructing tables to join onto test sets will not work. An alternative way to incorporate the rating timestamp into the model is to calculate residuals between actual ratings and predicted ratings from the regularized model and fit a line as a function of the rating timestamp. This fitted line represents a formula that can be applied to any given timestamp for generating a residual correction to add to the other bias terms in the model. Once a slope and intercept from the fit have been calculated and used to derive a formula column in the test sets, a final RMSE can be evaluated before choosing a final model for the final holdout test set.

Model	RMSE
Course Goal RMSE	0.86490
Average	1.06005
Average w/ Movie Effects	0.94296
Average w/ Movie + User Effects	0.86468
Average w/ Movie + User + Genres Effects	0.86432
Average w/ Movie + User + Genres + Release Year Effects	0.86413
Average w/ Regularized Movie + User + Genres + Release Year Effects	0.86365
Average w/ Regularized Movie + User + Genres + Release Year Effects & Timestamp Residual Fit	0.86356

This model has shown to have the lowest RMSE. When applied to the final holdout test set after deriving a release year column from the movie titles for joining the release year bias, the model is also successful in delivering an RMSE below the course goal of 0.86490.

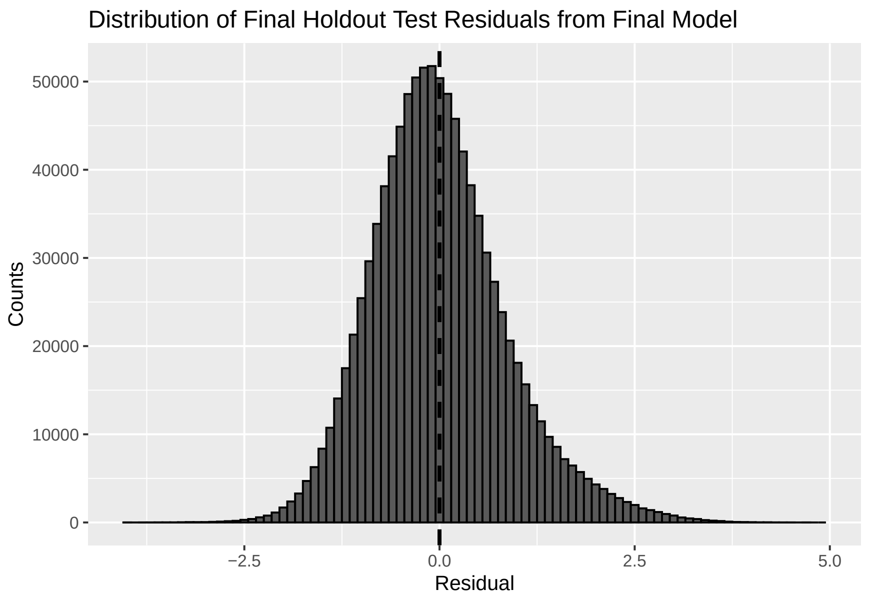
Model	RMSE
Course Goal RMSE	0.86490
Best Model Training RMSE	0.86356
Best Model Final Holdout Set RMSE	0.86458

Conclusion

The goal of this capstone project to build a recommendation model that predicts a user's rating of a given movie in a final holdout test set with an RMSE of 0.86490 using several linear regression techniques. Initial exploration of the provided data shows that movieId, userId, genres, movie release years as derived from movie titles, and rating timestamps could be the most promising predictors for a linear regression model.

The method of dividing the "edx" training set into 90% training and 10% developmental test sets (with users or movies not found in both tables being put only into the training set) is the same method as was done to split the "edx" training set from the final holdout test set. This ensures that all movieIds, userIds, genres, titles, and release years found in the test sets are found in the training set as well. The way these sets are split also enables a modeling approach of using the training mean as the prediction and adding predictor bias effects one at a time. For most identified predictors, a grouped summary table of average deviations between actual rating and the prior model's prediction is created and joined onto test sets as an additional bias term. Cross-validated regularization is applied to the model that uses the training average with movie, user, genres, and release year effects accounted for to suppress outlier effects from movies not often rated, users who rate infrequently, genre combinations that are too unique, and release years with few movies rated. The last refinement to the model is the incorporation of the rating timestamp effects. This involves fitting a line to the remaining residuals in the training set as a function of timestamp and using the line to formulate a predicted offset from rating timestamp effects that can be applied to any timestamp in the test sets. This final model is the end product of the project.

A histogram of the remaining residuals in the final holdout test set can help illustrate if further refinements could improve the RMSE.



The peak of the residual distribution is very close to the ideal of zero but shifted slightly negative, and the shape is slightly skewed positive. Future work would involve finding ways to tighten the distribution, make the curve more normal, and bring the peak closer to zero. The provided data sets are limited in information that could be used as predictors, but it is possible to wrangle other predictive information that may be effective without overfitting. Filtering genres does not seem to work when trialing removal of "IMAX", but it is possible that optimizing the right genre(s) to filter out or finding some other ways to transform the genre combinations into effective predictors could improve the model.

Due to the way the model is constructed, it is limited only to predicting ratings for users and movies that are both found in the training set. Any new users, movies, genres, and release years would not be able to have the model applied as-is, and the new ratings would need to be included in a retraining of the model. The rating timestamp effects component of the model offers the flexibility of interpolation for ratings within the timespan that the model is trained on, but it is unclear how a linear approximation would extrapolate to newer review timestamp years after the last of the fitted timestamps. Future work could also be done on making this predictor more robust than a linear fit on what may be a non-linear best fit. The overall linear regression approach is effective for the size of the datasets used in this project on a home computer, but there are many more sophisticated machine learning algorithms left to explore that could result in better predictions.