MovieLens Project

Morgan Houston

11 June 2021

1 Introduction

The goal of this project is to apply the lessons learned through the Harvard edX Data Science program, and to build upon those lessons. We will do this by utilizing the movieLens dataset to build a model which predicts movie ratings.

The movieLens dataset is provided by Grouplens Research. GroupLens collected rating datasets from the MovieLens web site. For this analysis, we are using the MovieLens 10M Dataset which consists of 10 million ratings of 10,000 movies by 72,000 users (GroupLens, 2021).

In order to develop our model, we separate the edx data set into a training set and a test set. Throughout our model development, we use the training set as the data we know, which our model will learn from, and the test set as the data we do not know, which we will use to compare our predictions to. We use an 80%/20% split for our training and test data sets.

After we have developed our model, we will test it against a validation dataset, which our model has never seen, as a final evaluation of our model's performance and a measure of how we would expect it to perform in a production environment.

2 Methods/Analysis

2.1 Data Analysis

First we will explore the structure of the edx dataset by examining the first few rows of the edx dataset.

userId	movieId	rating	timestamp	title	genres
1	122	5	838985046	Boomerang (1992)	Comedy|Romance
1	185	5	838983525	Net, The (1995)	Action|Crime|Thriller
1	292	5	838983421	Outbreak (1995)	Action|Drama|Sci-Fi|Thriller
1	316	5	838983392	Stargate (1994)	Action|Adventure|Sci-Fi
1	329	5	838983392	Star Trek: Generations (1994)	Action|Adventure|Drama|Sci-Fi
1	355	5	838984474	Flintstones, The (1994)	Children|Comedy|Fantasy

We see that we have six columns in the edx dataset: userId, movieId, rating, timestamp, title, and genres. Each row in the dataset represents a single user's rating for a single movie.

According to the README provided with the MovieLens 10M dataset, the userId is a unique, anonymized, identifer for a user. The movieId is the real movieLens id. Movie titles are expected to match those in IMDB, and include the year the movie was released. Ratings are on a 5-star scale, in half-star increments. Genres are a pipe-delimited list, from a pre-selected list of genres (GroupLens, 2021).

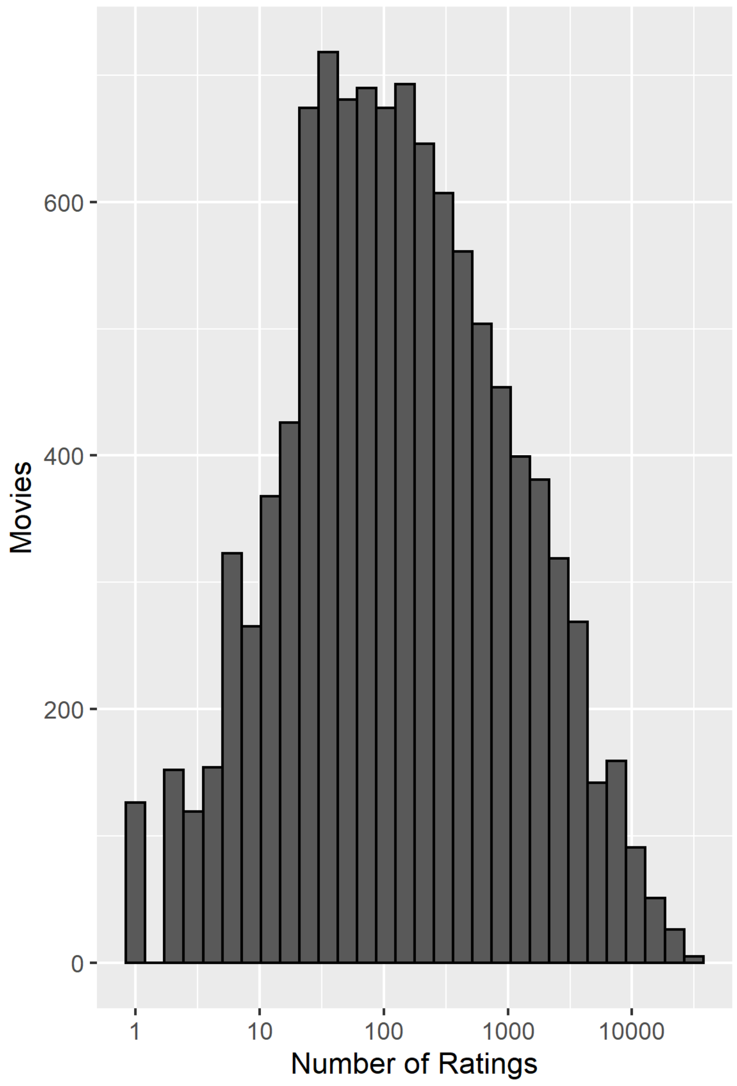
We can use the summary() function to see the the summary statistics for each column of the dataset, and confirm that we do not have any null values.

	userId	movieId	rating	timestamp	title	genres
	Min. : 1	Min. : 1	Min. :0.500	Min. :7.897e+08	Length:9000055	Length:9000055
	1st Qu.:18124	1st Qu.: 648	1st Qu.:3.000	1st Qu.:9.468e+08	Class :character	Class :character
	Median :35738	Median : 1834	Median :4.000	Median :1.035e+09	Mode :character	Mode :character
	Mean :35870	Mean : 4122	Mean :3.512	Mean :1.033e+09	NA	NA
	3rd Qu.:53607	3rd Qu.: 3626	3rd Qu.:4.000	3rd Qu.:1.127e+09	NA	NA
	Max. :71567	Max. :65133	Max. :5.000	Max. :1.231e+09	NA	NA

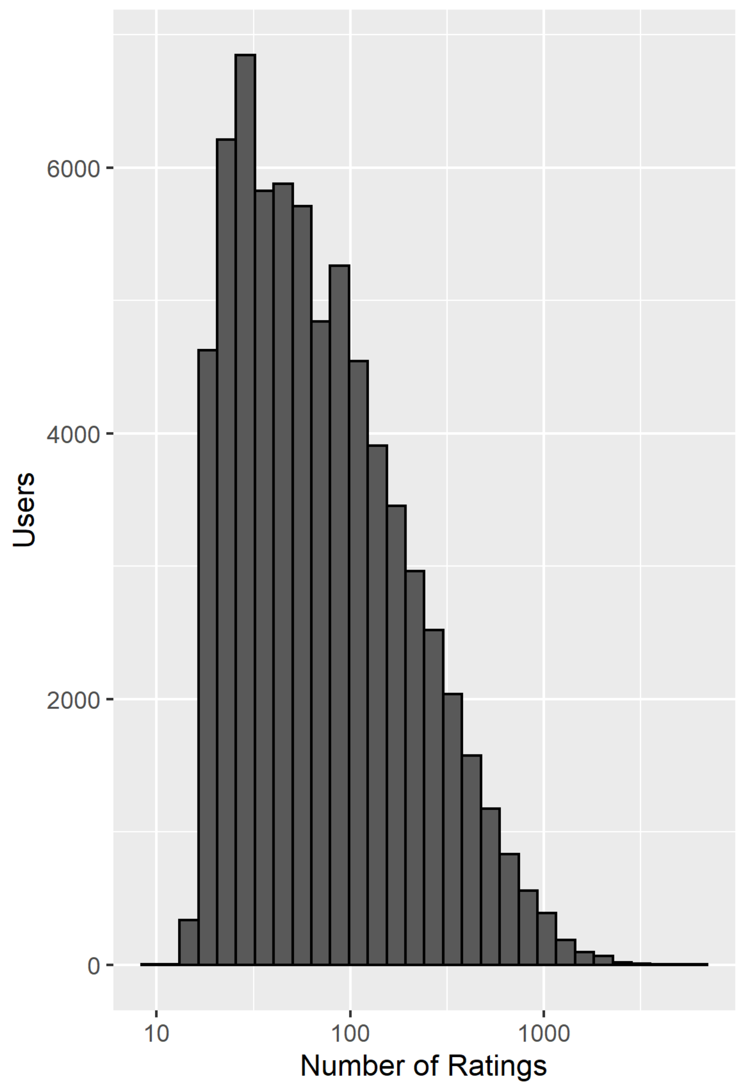
We can see the number of distinct movies and users in the edx dataset:

users	movies
69878	10677

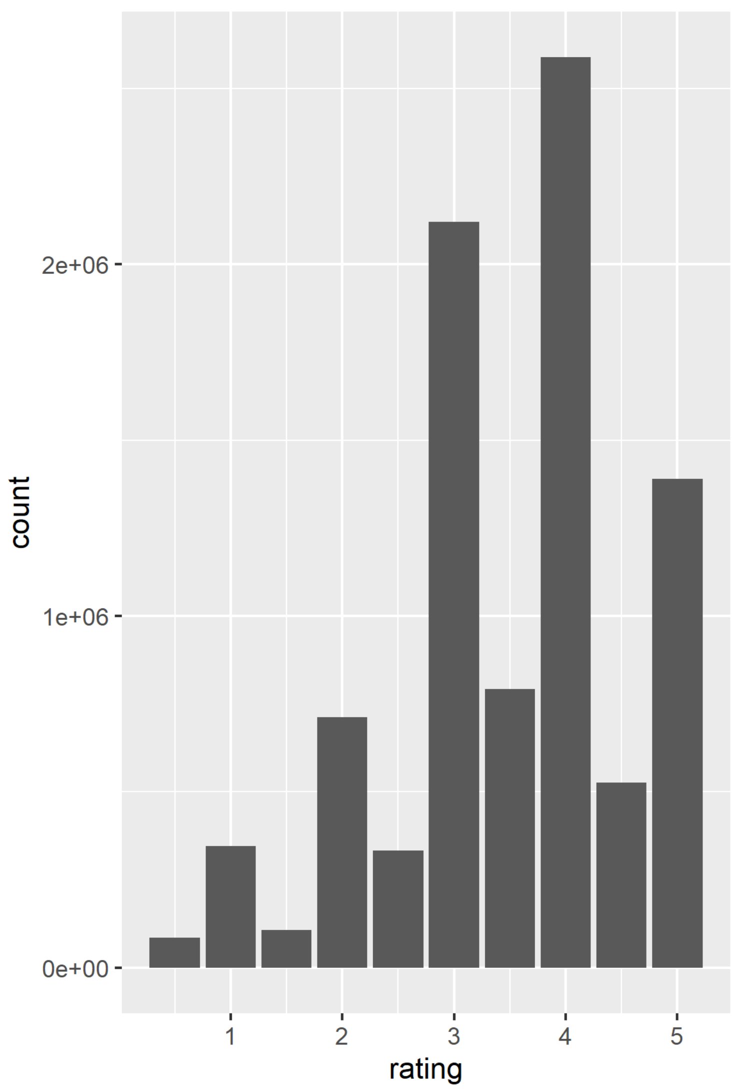
Since the number of rows in the dataset is much lower than the number of distinct users times the number of distinct movies, we know that not every user has rated every movie. We can look at the distribution of ratings per movie to illustrate this, and see that the distribution looks roughly normal. Some movies have very few ratings, while other movies are widely rated by users.



We can also examine the activity of the users by looking at the distribution of ratings each user has provided. Here we see a left-skewed distribution, which indicates that it is more common for users to only rate a few movies, than to rate very many movies.



We can also examine the distribution of the movie ratings in the dataset, and observe that in general, half star ratings are less frequent than whole star ratings.



We also look at the distribution of genres in the dataset. Given that the genres variable is a pipe-separated list of film genres for the movie, we will use separate_rows() to seee how many instances of each of the 19 base genres are present.

genres	count
Drama	3910127
Comedy	3540930
Action	2560545
Thriller	2325899
Adventure	1908892
Romance	1712100
Sci-Fi	1341183
Crime	1327715
Fantasy	925637
Children	737994
Horror	691485
Mystery	568332
War	511147
Animation	467168
Musical	433080
Western	189394
Film-Noir	118541
Documentary	93066
IMAX	8181
(no genres listed)	7

2.2 Model Development

2.2.1 Simple Model

As we think about how we can tackle this problem, we start with the simplest approach and predict the same rating for all movies, regardless of user. We can write this as:

$Y_{u,i} = \mu + \epsilon_{u,i}$

where the $\epsilon_{u,i}$ are independent errors and μ is the true rating for all movies. The estimate that minimizes the RMSE of this model is the least squares estimate of μ, which is the average of all ratings.

We first calculate the average of all the ratings in the train set:

```
## [1] 3.512458
```

We then calculate the RMSE using this simple model and comparing to the actual ratings in the test set, and create a table to store the RMSE of each of our models in.

model_number	method	RMSE
1	Just the average	1.059938

2.2.2 Movie Effect

Intuitively, we know that some movies are generally rated higher than others. We can see this in the data if we look at the lowest rated movies.

movieId	title	count	avg_rating
5805	Besotted (2001)	2	0.5000000
8394	Hi-Line, The (1999)	1	0.5000000
61768	Accused (Anklaget) (2005)	1	0.5000000
63828	Confessions of a Superhero (2007)	1	0.5000000
64999	War of the Worlds 2: The Next Wave (2008)	2	0.5000000
8859	SuperBabies: Baby Geniuses 2 (2004)	56	0.7946429

We note that the lowest rated movies are those which only have one or two ratings and are fairly obscure movies.

We can also look at the top rated movies in the dataset and see familiar films that we would expect to be widely highly rated.

movieId	title	count	avg_rating
296	Pulp Fiction (1994)	31362	4.154789
356	Forrest Gump (1994)	31079	4.012822
593	Silence of the Lambs, The (1991)	30382	4.204101
480	Jurassic Park (1993)	29360	3.663522
318	Shawshank Redemption, The (1994)	28015	4.455131
110	Braveheart (1995)	26212	4.081852

We adjust our initial model to account for this movie effect by adding a new term, b_i, which represents the average ranking for movie i:

$Y_{u,i} = \mu + b_i + \epsilon_{u,i}$

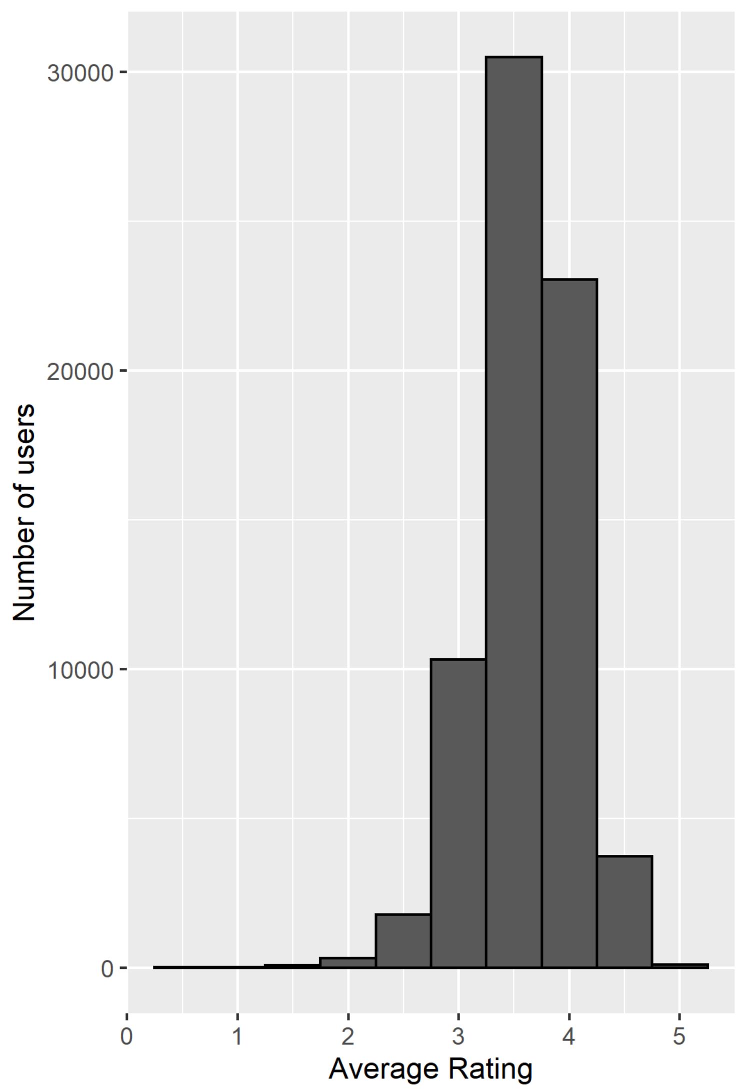
We obtain this movie effect by finding the average rating for each movie in the training data set. We then apply this model to the test set to find our predicted ratings, and calculate the RMSE of our predictions and the actual ratings in the test set. We see an improvement in the RMSE by adding the movie effect to our model.

model_number	method	RMSE
1	Just the average	1.0599383
2	With movie effect	0.9431317

2.2.3 User Effect

As we saw in our data exploration, some users are significantly more active than others.

From experience, we also know that some users will generally rank movies higher than others, and other users will only rate those movies at the extreme ends of their opinions (either loved the movie or hated the movie). This is supported by looking at the average rating by user, where we see that different users have different average ratings.



This leads us to explore the addition of a user effect. We again adjust our model and add a new term b_u, which represents the average ranking for user u.

$Y_{u,i} = \mu + b_i + b_u + \epsilon_{u,i}$

We again see an improvement in our RMSE, and that we have increased the accuracy of our model's predictions.

model_number	method	RMSE
1	Just the average	1.0599383
2	With movie effect	0.9431317
3	With movie + user effect	0.8653576

2.2.4 Genre Effect

We also expect that some genres will be more popular than others.We take the naive approach and look at capturing the genre effect at the category level first i.e. Action|Drama|Romance, instead of the individual genres.

We add a term, b_g to our model to capture this genre effect:

$Y_{u,i} = \mu + b_i + b_u + b_g + \epsilon_{u,i}$

Here, we see a negligble improvement in our RMSE as it is less than half a percent.

model_number	method	RMSE
1	Just the average	1.0599383
2	With movie effect	0.9431317
3	With movie + user effect	0.8653576
4	With movie + user +genre effect	0.8650217

2.2.5 Regularization

Next, we explore where our model made mistakes, and look at our top 10 largest residuals.

movieId	title	rating	residual
193	Showgirls (1995)	5.0	4.807277
527	Schindler's List (1993)	1.0	-4.741277
904	Rear Window (1954)	0.5	-4.947239
922	Sunset Blvd. (a.k.a. Sunset Boulevard) (1950)	0.5	-4.963833
1260	M (1931)	0.5	-4.952137
3435	Double Indemnity (1944)	0.5	-4.967198
4973	Amelie (Fabuleux destin d'AmÃ©lie Poulain, Le) (2001)	0.5	-5.034430
4993	Lord of the Rings: The Fellowship of the Ring, The (2001)	0.5	-4.889455
4993	Lord of the Rings: The Fellowship of the Ring, The (2001)	0.5	-4.770576
5952	Lord of the Rings: The Two Towers, The (2002)	0.5	-5.047782

We note that we have a mix of well-known movies and obscure movies here. We look at how frequently these movies are rated in the dataset.

movieId	count
193	2934
527	18563
904	6365
922	2351
1260	1551
3435	1707
4973	6978
4993	23138
5952	10440

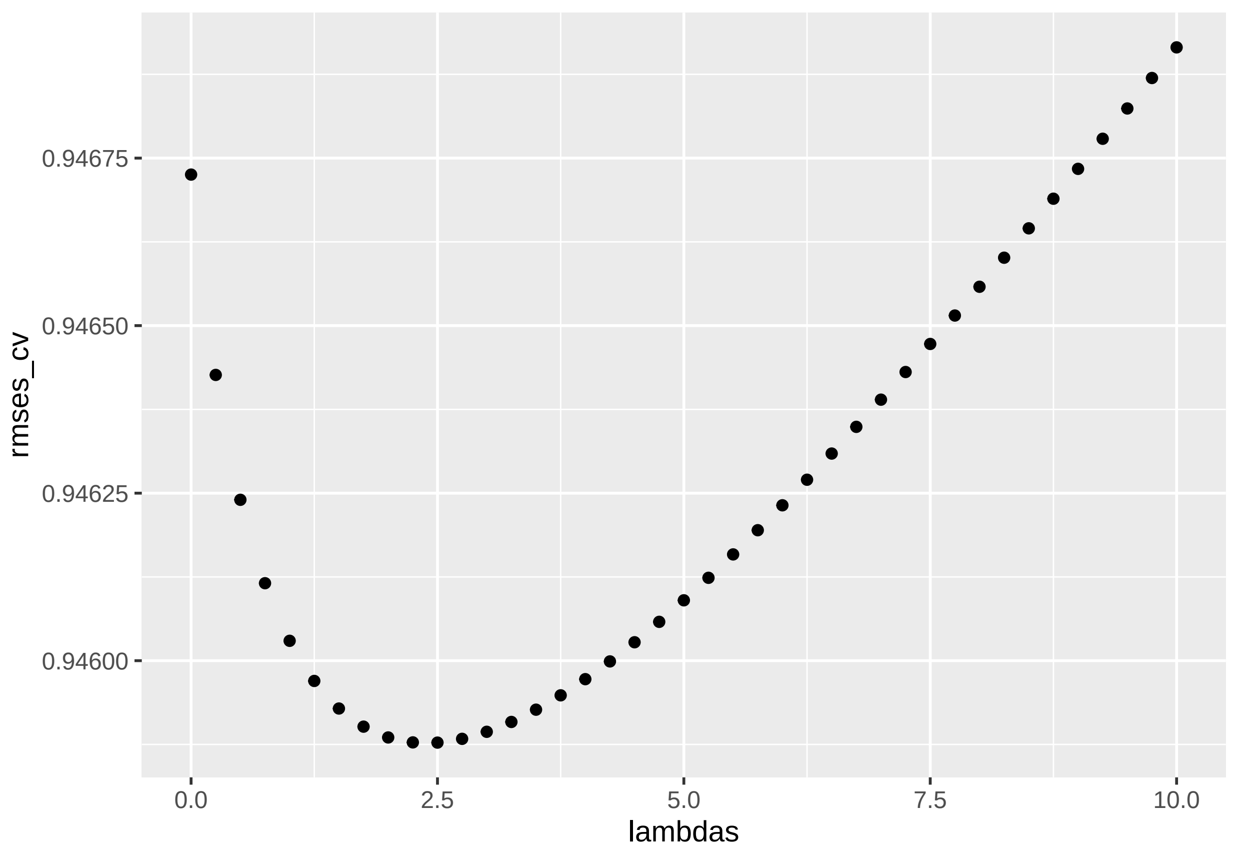
Some of the movies have been rated many times, and we have users which have rated them much lower than others, while some of these movies have much fewer ratings. We can penalize those movies with very few ratings, which are subject to more noise in the estimates.

2.2.6 Movie Effect Regularization

We use regularization on our movie effect model by introducing a new parameter, lambda, into our model, and instead of minimizing the least squares equation, we minimize a new equation with a penalty term:

$\sum_{u,i}(y_{u,i} - \mu - b_i)^2 + \lambda \sum_i b_i^2$

We use 5-fold cross-validation to select the value of lambda. We then plot the values of lambda against the average RMSE calculated to see which value of lambda minimizes the RMSEs.



We choose the value of lambda which minimizes the average RMSE and apply it to our model. We calculate the RMSE of the regularized model to see if it improves.

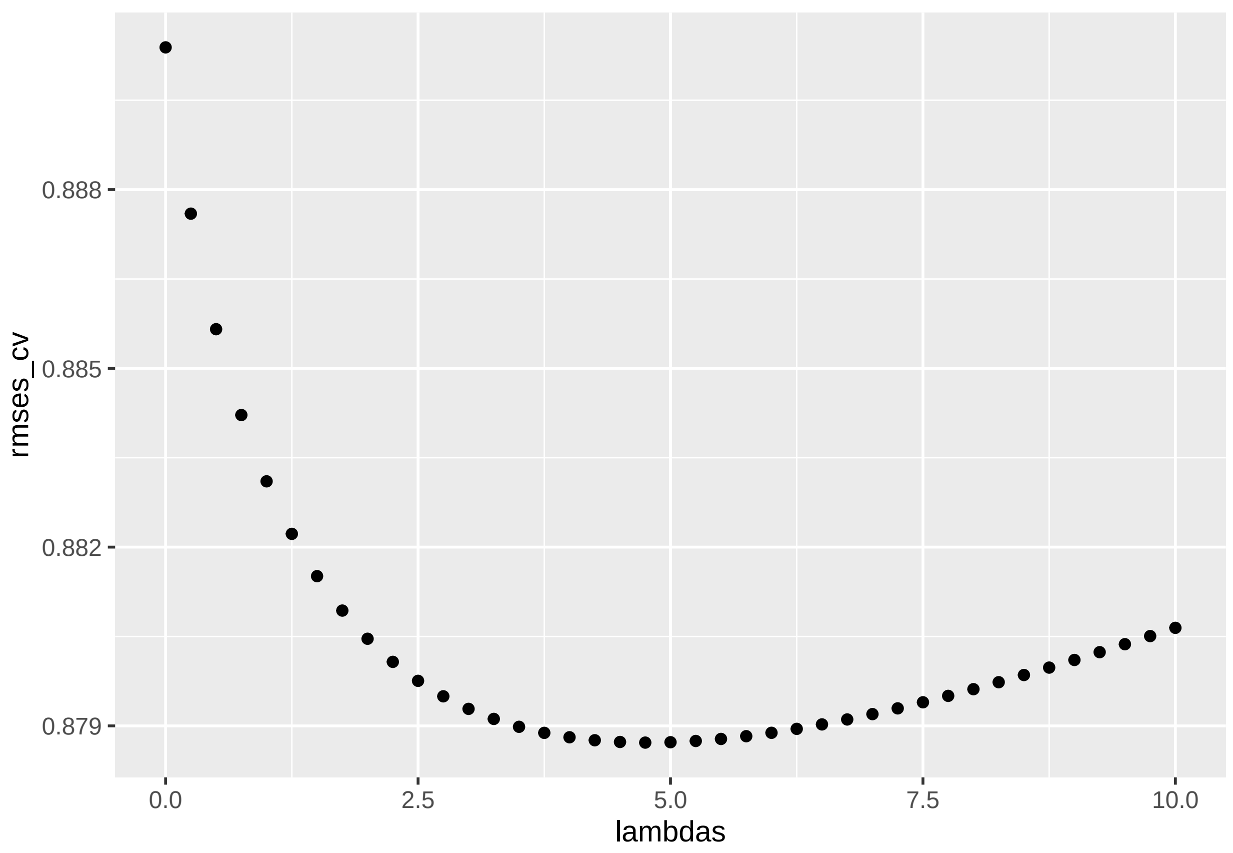
model_number	method	RMSE
1	Just the average	1.0599043
2	With movie effect	0.9437429
3	With movie + user effect	0.8659319
4	With movie + user +genre effect	0.8655941
5	Regularized movie effect	0.9436745

2.2.7 Movie + User Effect Regularization

Next we look at tuning the model with both the movie effect and user effect. We modify our model:

$\sum_{u,i}(y_{u,i} - \mu - b_i - b_u)^2 + \lambda(\sum_i b_i^2 + \sum_i b_u^2)$

We again use 5-fold cross validation to choose lambda and plot the values of lambda against the average RMSE calculated to see which value of lambda minimizes the RMSEs.



We choose the value of lambda which minimizes the average RMSE and apply it to our model. We calculate the RMSE of the regularized model to see if it improves.

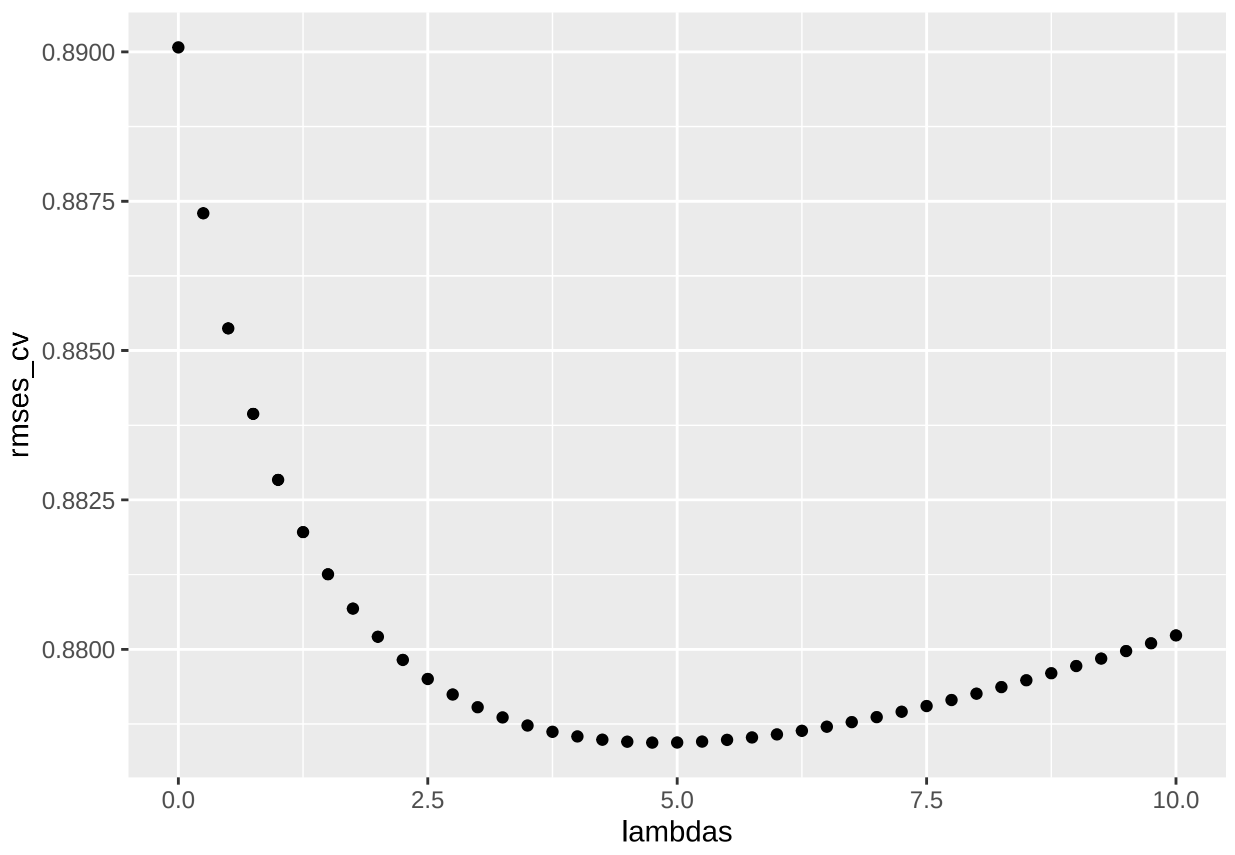
model_number	method	RMSE
1	Just the average	1.0599043
2	With movie effect	0.9437429
3	With movie + user effect	0.8659319
4	With movie + user +genre effect	0.8655941
5	Regularized movie effect	0.9436745
6	Regularized movie + user effect	0.8652421

2.2.8 Movie + User + Genre Effect Regularization

Next we look at look at tuning the model with the movie effect, user effect, and genre effect. We again modify our model:

$\sum_{u,i}(y_{u,i} - \mu - b_i - b_u - b_g)^2 + \lambda(\sum_i b_i^2 + \sum_i b_i^2 + \sum_u b_g^2)$

We again use 5-fold cross validation to choose lambda and plot the values of lambda against the average RMSE calculated to see which value of lambda minimizes the RMSEs.



We choose the value of lambda which minimizes the average RMSE and apply it to our model. We calculate the RMSE of the regularized model to see if it improves.

model_number	method	RMSE
1	Just the average	1.0599043
2	With movie effect	0.9437429
3	With movie + user effect	0.8659319
4	With movie + user +genre effect	0.8655941
5	Regularized movie effect	0.9436745
6	Regularized movie + user effect	0.8652421
7	Regularized movie + user + genre effect	0.8652421

2.2.9 Matrix Factorization

Finally we consider another technique to see if we can further improve the RMSE. If we consider an m x n matrix of ratings, with n columns for each movie in the dataset, and m rows for each user in the dataset, the value in row m and column n is the rating provided by that user for that movie.

As described by r-bloggers.com, the idea of matrix factorization is to approximate this m x n rating matrix as the product of two matrices of lower dimensions, $P_{k\times m}$ and $Q_{k\times n}$. Then, if p_u is the u-th column of P, and q_v is the v-th column of Q, we will predict the rating given by user u on item v as $p'_u q_v$.

We implement this using the recosystem package. We tune and train the model using the default values, and calculate the RMSE of the model.

model_number	method	RMSE
1	Just the average	1.0599043
2	With movie effect	0.9437429
3	With movie + user effect	0.8659319
4	With movie + user +genre effect	0.8655941
5	Regularized movie effect	0.9436745
6	Regularized movie + user effect	0.8652421
7	Regularized movie + user + genre effect	0.8652421
8	Matrix Factorization	0.7906868

3 Results

The matrix factorization model has the best RMSE on the test set, so this is the one we select as our final model. We apply it to the final validation set and calculate the RMSE.

```
## [1] 0.7819903
```

4 Recommendations for Further Improvements

As demonstrated in this project, the genre categories as presented in the dataset did little to improve the RMSE. However, this project could be furthered by investigating the impact of each individual genre, and their interactions as a category as a multiple linear regression model:

$Y_{u,i} = \mu + b_i + b_u + \sum_{k=1}^{K} x^k \beta_k + \epsilon_{u,i}$ with $x^k_{u,i} = 1$ if $g_{u,i}$ is genre k.

However, due to system limitations, we were unable to perform this calculation on a dataset of this size.

Additionally, we could consider the year of release of a film as compared to the year of a review - we might expect to see very old movies rated highly (i.e. the classics), as well as an effect for brand new movies rated highly i.e. summer blockbusters.

We would also like to further investigate the usage of matrix factorization, and explore the impact the additional features like genres, or movies in a series (i.e. Godfather 1, Godfather 2, Godfather 3).

5 Conclusion

This project aimed to build on the material presented through the Harvard edX machine learning course and chapter 33 of *Introduction to Data Science,Data Analysis and Prediction Algorithms with R*, and to demonstrate the knowledge gained through the program as a whole through the development of a movie rating prediction model. This project explored multiple aspects of the MovieLens dataset and the effects of the movie, user, and genre on the rating, as well as the matrix factorization technique.

6 References

1. Irizarry, Rafael A (May 24,2021). *Introduction to Data Science,Data Analysis and Prediction Algorithms with R.* https://rafalab.github.io/dsbook/
2. GroupLens, 2021. *MovieLens.* https://grouplens.org/datasets/movielens/
3. Yixuan's Blog - R (July 14,2016). *recosystem: Recommender System Using Parallel Matrix Factorization* https://www.r-bloggers.com/2016/07/recosystem-recommender-system-using-parallel-matrix-factorization/