MovieLens Recommendation System

HarvardX Capstone Project

K. H. Hogan

8 December 2020

Introduction

Recommendation systems are machine learning algorithms that make predictions from ratings users have given relevant items. Netflix has, perhaps, the most well-known movie recommendation system. The HarvardX Data Science Capstone project is to create a movie recommendation system from the MovieLens 10M dataset. The assignment perform data analysis, data wrangling, the use discovered predictors to train a model to predict ratings. The MovieLens 10M dataset was collected by and obtained from the GroupLens research lab at the University of Minnesota. The data can be found at: https://grouplens.org/datasets/movielens/

Prior to training an algorithm, the data must be split into separate training and validation sets. Here the training set will contain 90% of the data and used for analysis and exploration. The validation set contains the remaining 10% will is used only to validate the final model results.

After visually exploring the training set, key predictors showing variability in ratings will be chosen to train the model. The training set will be split 90/10 again for model building and evaluation. According to class instructions, I will assume a Bayesian approach and begin the model based on the simplest prediction: the average rating of the training set. The recommendation algorithm will be built by calculating the effects of chosen predictors on the benchmark average rating. Once all effects are applied, a set of predicted ratings will be calculated and tested against the validation set ratings. Model performance will be evaluated by root mean square error.

Data Analysis

Data Exploration

The MovieLens 10M dataset contains 10 million observations and is a sparse ratings matrix meaning that not all users have rated all movies. Totals and average movie rating in the training set:

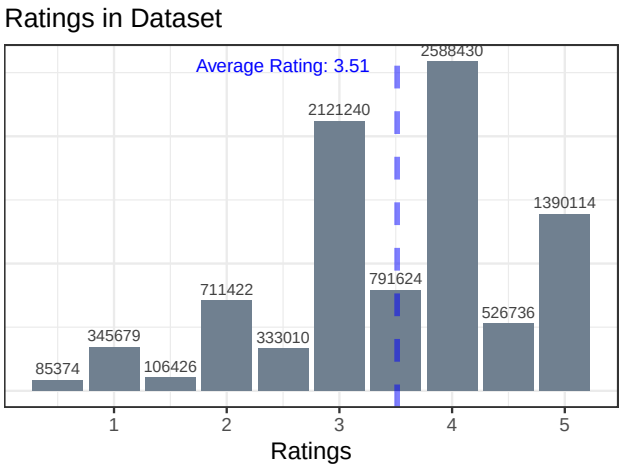
Users	Movies	Ratings	Average_Rating
69878	10677	9000055	3.512465

Variables provided include: userId, movieId, rating, timestamp of rating, movie title with year released, and genres.

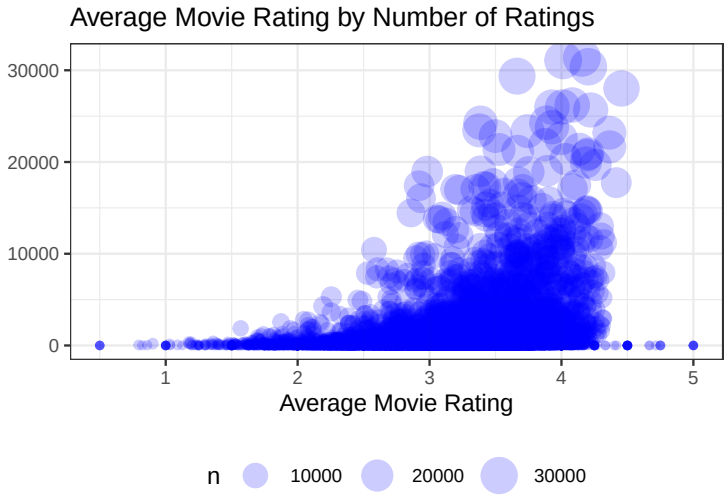
Table 1: First lines of data set:

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

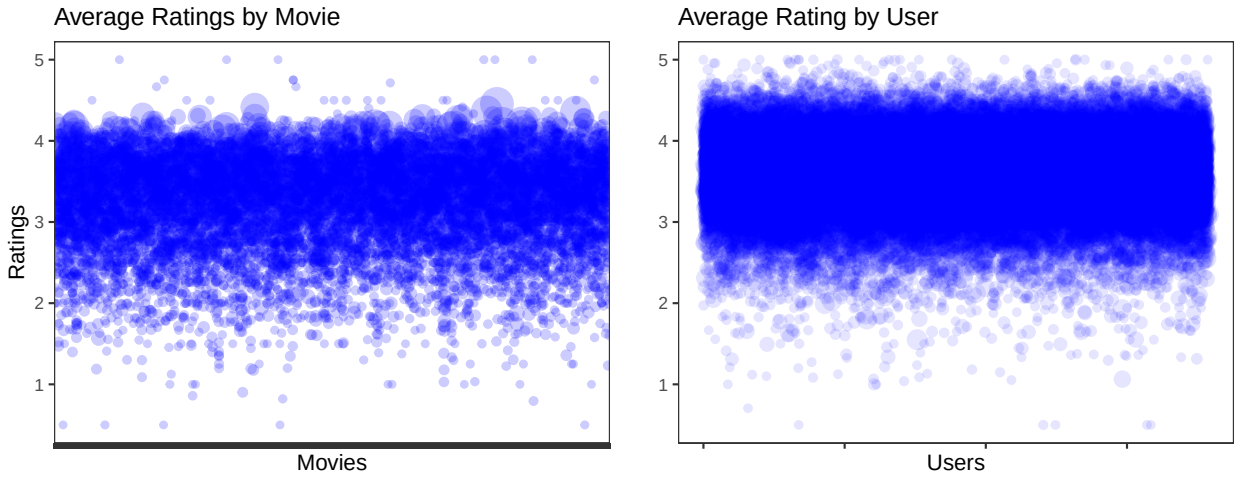
Since the model is predicting ratings, it is important to see the distribution of ratings in the data.



The majority of user ratings are 3 or higher. Looking at ratings by movie, shows that movies with more ratings tend to have higher ratings, and the majority movies have less than 5000 ratings.



Visualizing average ratings by movie and by user show a lot of variability centering between 3 and 4 indicating movie and user specific effects.



Users' ratings appear to average slightly higher than movie ratings.

Movie Average: 3.192 User Average: 3.614

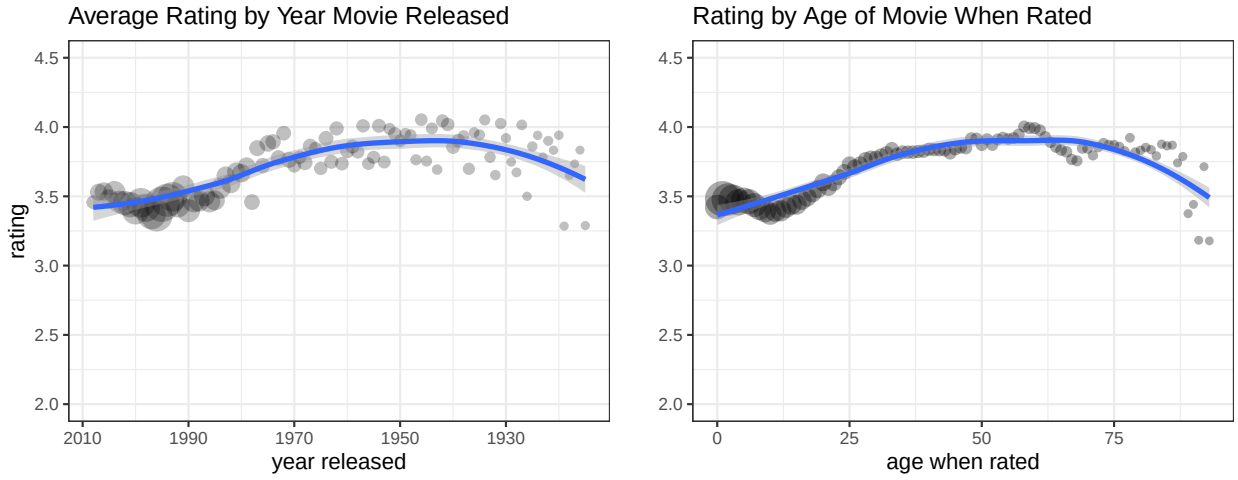
Next I want to explore date effects, but the the data must be transformed first.

Data Cleaning

Initial data cleaning involves creating new variables by separating the year released from the title and converting the timestamp into a readable date rated. I will also create an age of the movie when rated column.

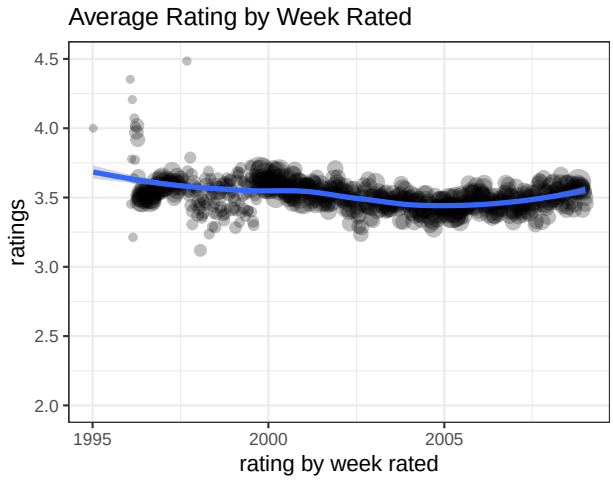
Data Analysis

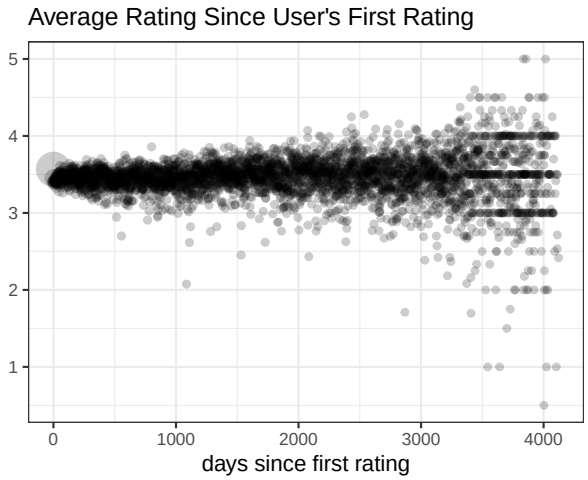
It is now possible to visual average ratings by year movie was released and date rated to look for variability.



There is a pattern for higher ratings for older movies both for year movie was released and age of movie when rated. The pattern is very similar, but the variability is different. I will explore both as predictors.

Now I will look for effects of time-based, user-specific effects of date movie was rated and if their ratings change over time.

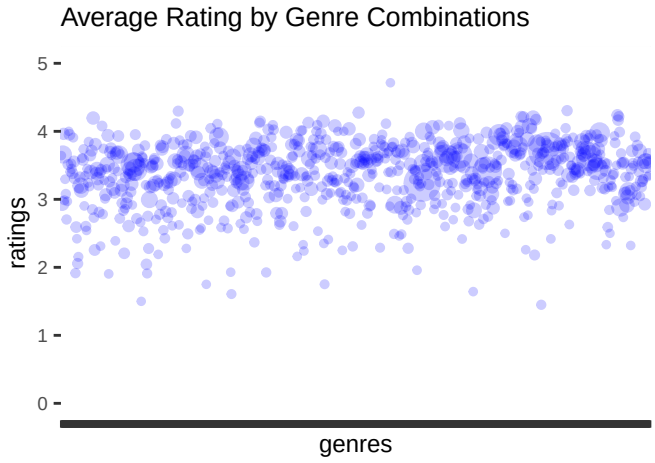




Both the year movie was released and age of movie when rated show some variability. The time of year of when rated and how long the user has been rating movies show variability.

Since ratings by days become more sparse as time increased, I will round to weeks rated to prevent NAs in the data. I must perform additional pre-processing to extract how many weeks since the user first started rating movies. To discover the true first date rated will mean comparing the minimum date rated by user in both the training and test tests, choosing the date and then adding it back to both sets.

I am also interested in the power of genres as predictors.



Genres also show a lot of variability. There are 797 unique combinations of genres.

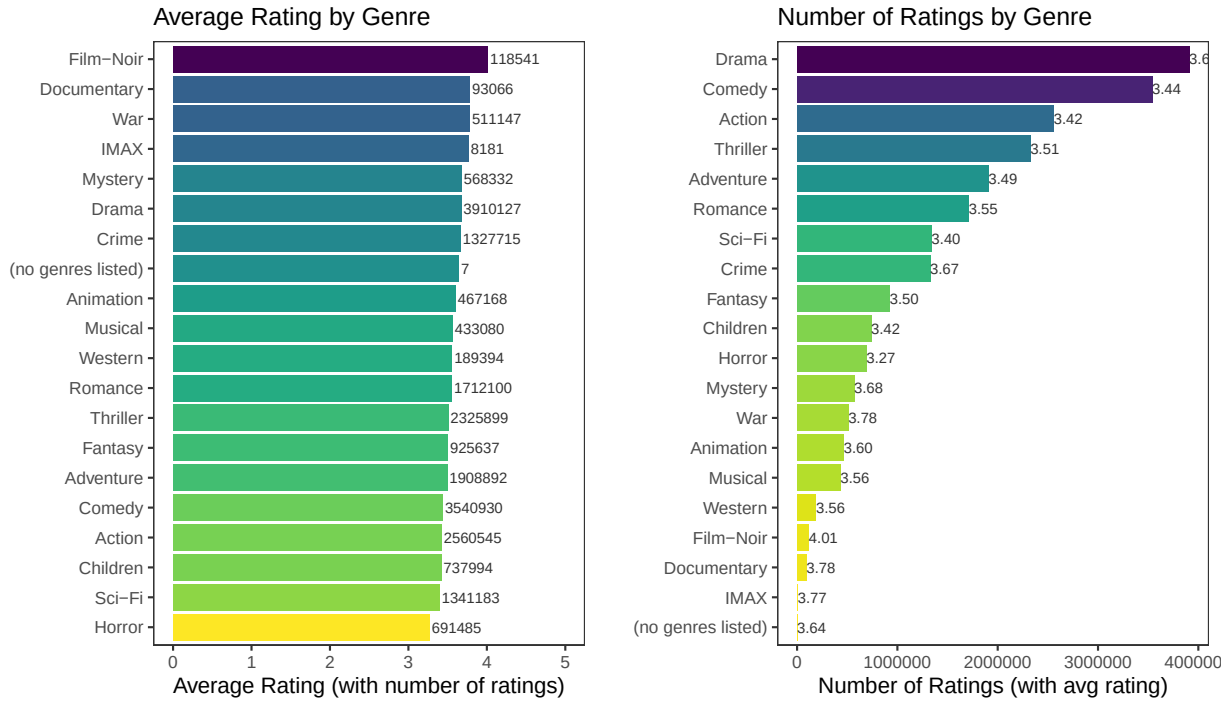
Table 2: Top 10 rated genres

genres	avg	movies	n
Animation|IMAX|Sci-Fi	4.71	1	7
Drama|Film-Noir|Romance	4.30	3	2989
Action|Crime|Drama|IMAX	4.30	1	2353
Animation|Children|Comedy|Crime	4.28	1	7167
Film-Noir|Mystery	4.24	2	5988
Crime|Film-Noir|Mystery	4.22	2	4029
Film-Noir|Romance|Thriller	4.22	1	2453
Crime|Film-Noir|Thriller	4.21	5	4844
Crime|Mystery|Thriller	4.20	15	26892
Action|Adventure|Comedy|Fantasy|Romance	4.20	1	14809

In the top 10 rated genres, all but one have three or more genres, and half have only 1 movie.

Exploring further, I find the number of movies per combination varies a lot with a range of 1, 1815. The number of ratings varies even more: 2, 733296. Unfortunately, 594 genre combinations have less than five movies, and 113 have less than 50 ratings. With so many movies having a unique genre combination, the format of this variable reduces it's predictive capability and cannot capture the interaction of genres on ratings.

The genres variable needs to be split out to explore genres individually.



The number of ratings per genre vs average rating per genre does not follow the trend observed earlier where movies with more ratings average higher. The modeling for this project and limits the ability to model on the interaction of the genres. It would be best to split the genres out to separate variables, but that would increase the size of the dataset or make the used in the project extremely long. Instead, I will include both this combination of genres as a predictor and will perform matrix factorization of the residuals of the model.

Considering the data further, I can see there are also "no genres listed" and IMAX genres. Should IMAX be considered a genre?

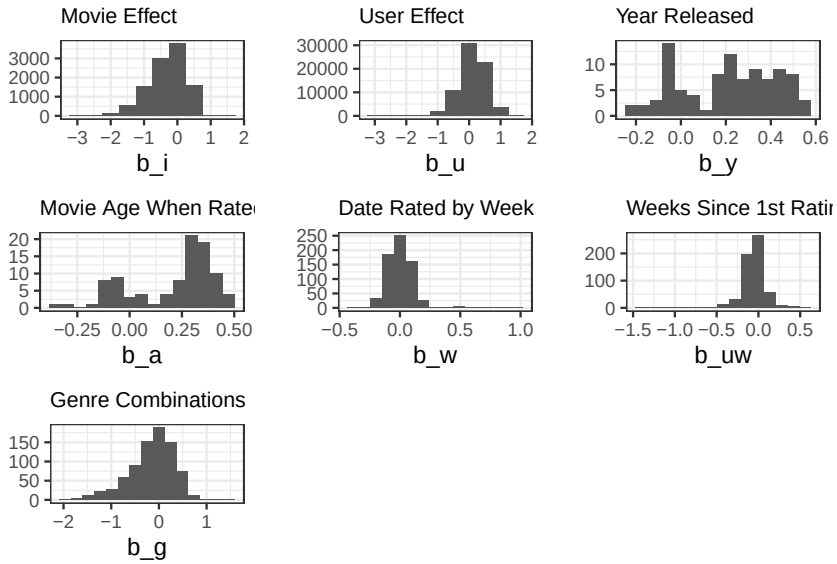
There is only one movie with no genre: Pull My Daisy which was a short drama according to Wikipedia. I will and update the movie's genre. There are only 29 IMAX movies and 12 IMAX genres.

Table 3: Top 10 IMAX Movies

movieId	title	genres	avg	n
58559	Dark Knight, The	Action|Crime|Drama|IMAX	4.30	2353
3159	Fantasia 2000	Animation|Children|IMAX|Musical	3.54	1934
54001	Harry Potter and the Order of the Phoenix	Adventure|Drama|Fantasy|IMAX	3.66	1385
1797	Everest	Documentary|IMAX	3.92	1039
8965	Polar Express, The	Adventure|Animation|Children|Fantasy|IMAX	3.20	550
4453	Michael Jordan to the Max	Documentary|IMAX	3.00	89
4458	Africa: The Serengeti	Documentary|IMAX	3.98	89
6289	Ghosts of the Abyss	Documentary|IMAX	3.37	75
62999	Madagascar: Escape 2 Africa	Action|Adventure|Animation|Children|Comedy|IMAX	3.30	66

Five of the top rated IMAX movies are blockbuster films which were additionally released in IMAX. Therefore the IMAX genre included could lead to overtraining. I will remove IMAX from these films since most viewers would not have been watching the movie in IMAX.

Now I will inspect the additional modeling ability the my chosen predictors:



Each chosen predictor shows predictive power varying between a range of .75 to 3 points. This appears promising, but it also shows the possibility of calculating predictions outside of the ratings range of 0.5 to 5. The prediction range will need to be monitored. Also, while genre combinations has the one of the largest distributions, many of the combinations have few movies, I hypothesize that modeling with the genre combinations as grouped in the dataset will limit its predictive power.

Modeling Methods & Results

Model Evaluation

To evaluate the model, predictions will be compared against test set data to obtain RMSE. The goal of the project is to achieve a Root Mean Squared Error (RMSE) < 0.8649 on the validation set.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i,u}(\hat{y}_{i,u} - y_{i,u})^2}$$

For comparison the team who won NetFlix's 2009 challenge to improve their recommendation system by 10% had to achieve an RMSE of about 0.857. (Incidentally, that team won a $1 million prize!)

Model Building

Due to the size of the dataset and small processor capability of laptops, modeling approaches are limited. Per project instructions, I will build on the recommendation system model presented in HarvardX's Machine Learning course. The model will be built using the mean calculated at 3.512 then adding on predictor effects, referred to as biases, discovered during data exploration to the base model. I will run each predictor as a model in order to measure decrease in RMSE. From my prior analysis chosen effects include:

- b_{i}: movie-specific effects
- b_{u}: user-specific effects
- b_{y}: year the movie was released
- b_{a}: age of the movie when it was rated
- b_{d}: date when the movie was rated (by week)
- b_{uw}: number of weeks since the user's first rating
- b_{g}: genre effect (with genre combinations)

Base Model: The base model is just the average of all ratings in the dataset. It assumes any predicted rating (Y) is the average rating (μ) for all movies (i) and users (u) where differences are explained as independent errors (ε). $Y_{i,u} = \mu + \varepsilon_{i,u}$

Model	RMSE
Naive Bayes	1.060129

Model 1: Adds movie-specific effect (b_i) to the base model. $Y_{i,u} = \mu + b_i + \varepsilon_{i,u}$

Model	RMSE
Movie Effect	0.943064

Model 2: Adds user-specific (b_u) effects. $Y_{i,u} = \mu + b_i + b_u + \varepsilon_{i,u}$

Model	RMSE
Movie + User Effects	0.8646348

As I add on effects, I need to check the range of my predictions. Model 2 prediction range: -0.543, 5.934 The calculated predictions are already outside of the possible ratings range of 0.5 to 5. I will simply update any predictions under 0.5 and over 5 to those ratings.

Model 2b: Applies upper and lower limits. $Y_{i,u} = lim_{.5 \to 5}(\mu + b_i + b_u + \varepsilon_{i,u})$

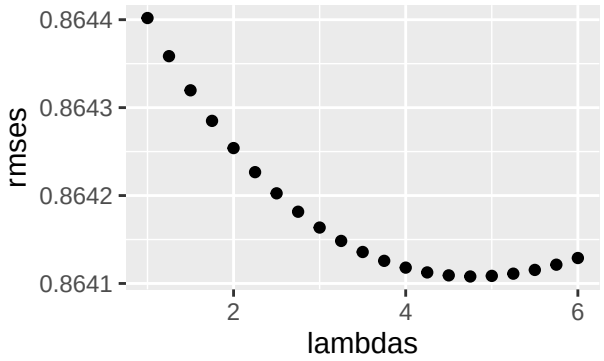
Model	RMSE
Movie + User Effects with limits	0.8644179

Correcting out of range predictions does slightly improve RMSE. I will add this step after each prediction.

Model 3: Applies regularization to movie and effects. In model 1 and 2, all movies and users are weighted equally. Data exploration showed that some movies have very few ratings while other have thousands. Therefore, weighing the effects of movies and users with one rating the same as those with a 1000 ratings can lead to poor results. Introducing regularization to the model will apply a penalty term the the number of ratings to minimize effects of those with few observations.

The movie effect penalty term will be calculated as: $b_i(\lambda) = \frac{1}{n_i+\lambda}\sum_{u=1}^{n_i}(Y_{i,u} - \hat{\mu})$

We now calculate with penalized least squares. $\sum_{i,u}(y_{i,u} - \mu - b_i - b_u) + \lambda(\sum_i b_i^2 + \sum_u b_u^2)$



The Lambda which minimizes RMSE is 4.75.

$$Y_{i,u} = lim_{.5\to5}(\mu + b_i(\lambda) + b_u(\lambda) + \varepsilon_{i,u})$$

Model	RMSE
Movie & User Regularized Effects	0.8639802

Introducing regularization and updating out of limit predictions, reduces the modeling RMSE below the project goal (at least for the test set). Results will continue to improve by continuing to add on effects from above analysis.

Model 4: Adds year the movie was released (b_y) effect. $Y_{i,u} = lim_{.5\to5}(\mu + b_i(\lambda) + b_u(\lambda) + b_y + \varepsilon_{i,u})$

Model	RMSE
Movie&User Regularized + Year Released Effects	0.8636894

Model 5: Adds age of the movie (b_a) when it was rated.

$$Y_{i,u} = lim_{.5\to5}(\mu + b_i(\lambda) + b_u(\lambda) + b_y + b_a + \varepsilon_{i,u})$$

Model	RMSE
Movie&User Reg + Year + Age When Rated	0.8633508

Model 6: Adds the date the movie was rated rounded to the week (b_d).

$$Y_{i,u} = lim_{.5\to5}(\mu + b_i(\lambda) + b_u(\lambda) + b_y + b_a + b_d + \varepsilon_{i,u})$$

Model	RMSE
Movie&User Reg + Year + Age&Week When Rated	0.8632033

Model 7: Adds how many weeks since the user's first rating (b_{uw}).

$$Y_{i,u} = lim_{.5 \to 5}(\mu + b_i(\lambda) + b_u(\lambda) + b_y + b_a + b_d + b_{uw} + \varepsilon_{i,u})$$

Model	RMSE
Movie/User Reg + Year + Age/Week Rated + User Rating Weeks	0.8630963

Model 8: Adds the effect of genres as they are combined in the data (b_g).

$$Y_{i,u} = lim_{.5 \to 5}(\mu + b_i(\lambda) + b_u(\lambda) + b_y + b_a + b_d + b_{uw} + b_g + \varepsilon_{i,u})$$

Model	RMSE
Movie&User Reg + Year + Age&Week Rated + UserWeeks + GenreCombo	0.8627457

The model including genre combinations did not improve RMSE as much as I expected. Possibly, this was be due to modeling the 791 unique genre combinations instead of factoring them out separately to model their interactions. However there is a lot of information not included in the MovieLens data that impacts ratings including: actors, directors, and user demographics like gender and age.

In order to improve on the model, I will user matrix factorization on the residuals from Model 8. "Matrix factorization is a class of collaborative filtering algorithms used in recommender systems. Matrix factorization algorithms work by decomposing the user-item interaction matrix into the product of two lower dimensionality rectangular matrices."[1]

Model 9: Matrix factorization of model's residuals. Constructing a matrix with the vector of residuals from model 8 (r) with user vector (p) and movie vector (q). $r_{u,i} \approx p_u q_i$

$$Y_{i,u} = lim_{.5 \to 5}(\mu + b_i(\lambda) + b_u(\lambda) + b_y + b_a + b_d + b_{uw} + b_g + p_u q_i + \varepsilon_{i,u})$$

I used the Recosystem package. First, I removed all large elements from my environment that were not be used in the matrix factorization. I calculated the residuals for training and test sets from model 8, and transposed them into matrix as: userId, movieId, residuals. I constructed a Reco() object (r) the tuned parameters with on the training set based on CRAN.R Project literature. The model was then trained with minimized opts and predictions were generated for residuals which were then added to predicted ratings from model 8.

Model	RMSE
Movie&User Reg + Year + Age&WeekRated + UserWeeks + Genres + Matrix Factorization	0.7892525

Matrix factorization has improved the model considerably.

[1] https://en.wikipedia.org/wiki/Matrix_factorization_(recommender_systems)

Final Model Validation

For final validation, the model is tested with the original held-out validation set. I am looking to achieve a RMSE < .86490.

Final Validation Results:

Model	RMSE
Final: Movie&User Reg + Year + Age&Week Rated + User Rating Weeks	0.8638498
Final: Movie&User Reg + Year + Age&WeekRated + UserWeeks + Genres + MatrixFactorzn	0.7898906

Conclusion

I have used the MovieLens 10M dataset to analyze movie ratings based on movie, user and time effects. The goal of the project was to build a model based on these discoveries to predict ratings achieving an RMSE of less than 0.8649.

In the model above, RMSE decreases as we add predictors. Movie and User-specific effects along with regularization from the class model decreased RMSE the most at 11%, 8%, and 6% respectively. The year the movie was released, age when rated and genre combination effects decreased RMSE by 4% each. The user date-rated effects were smaller at 2% each. Modeling for genres was limited as there were 791 unique combinations although many had only one movie.

After modeling for effects found in the data, I used matrix factorization to discover latent effects. Recosystem matrix factorization was used to process the residuals after movie, user and time effects were applied. Matrix factorization decreased RMSE to 0.7898906 a 25.5% improvement from guessing the average and well below the project goal.

RMSE measures the model accuracy, but looking at actual model outcomes can be more understandable. A comparison of the most rated movies from the validation set shows predictions within .04 of the actual rating.

title	count	avg_rating	prediction
Pulp Fiction	3502	4.18	4.15
Forrest Gump	3378	4.02	4.04
Silence of the Lambs, The	3286	4.21	4.19
Jurassic Park	3271	3.64	3.67
Shawshank Redemption, The	3111	4.48	4.47
Terminator 2: Judgment Day	2964	3.93	3.94
Fugitive, The	2954	3.99	4.01
Braveheart	2942	4.09	4.10
Star Wars: Episode IV - A New Hope (a.k.a. Star Wars)	2894	4.21	4.22
Batman	2763	3.38	3.39

But what about movies with less than average rating and fewer ratings? As the number of ratings declines, the accuracy naturally declines as well. The model performs relatively well as long as there are at least 100 ratings, but even with less only 50 to 100 ratings the predictions are still relatively close for the highest rated movies.

Top 10 Movies Average Rating < 3 (100-1000 Ratings)				Top 10 Movies (only 50-100 Ratings)			
title	n	avg	prediction	title	count	avg_rating	prediction
Deep Impact	657	3.00	2.99	To Live	79	4.27	4.18
Pet Sematary	194	2.99	2.96	Animal Crackers	51	4.20	4.13
Romeo Must Die	214	2.99	2.98	Children of Paradise	59	4.19	4.35
EDtv	246	2.99	3.05	Wild Strawberries	59	4.18	4.14
Beavis and Butt-Head Do America	444	2.99	3.07	Into the Wild	86	4.17	3.95
Wolf	336	2.99	3.06	Best Years of Our Lives, The	88	4.17	4.21
Sudden Death	240	2.99	3.02	Nights of Cabiria	59	4.17	4.24
Shadow, The	228	2.99	2.93	Trust	52	4.16	4.15
Casper	689	2.99	3.01	Night at the Opera, A	75	4.16	4.19
Man Who Knew Too Little, The	105	2.99	3.16	Grand Illusion	88	4.16	4.25

There are many modeling alternatives. I did not try linear regression. If genres were split apart and pivoted wide then formatted as factors and with movieId and userId also converted to factors, a regression equation with the movieId + userId + year + age rated + date rated + genre1 * genre2 * . . . * genre20 where genre interaction could be incorporated which should provide a better results than model 8. K-nearest-neighbors would also be powerful since like movies, users and genres are rated similarly. In fact Netflix uses 1300 neighborhoods in their very successful recommender system. Joining this data with the IMDB data contain more movie-specific data such as actors or awards received would also be a good method although it would considerably increase the set size.

Overall, incorporating multiple effects even those with with small predicting capability combined with using matrix factorization of the residuals produced a relatively successful model.