Creating Movie Recommendation Systems using Subgroup Analysis

Will Powers

12/28/2020

Overview

Motivation

In 2006 Netflix created a challenge entitled the "Netflix Challenge." The goal was to encourage developers to create a data model trained from a database of 100 million user ratings that would more accurately predict a given user's rating on a movie than Netflix's software at the time.

Project Goals

The goal of this paper is to attempt the objectives of that competition to create an efficient prediction model, using a smaller version of that database, both developed by "GroupLens".

The Dataset

The Dataset used for this paper is a modified version of the original data set containing only ~10 million user ratings. The dataset is developed by the creators of the original data set, GroupLens, a research lab at the University of Minnesota. A brief preview of the data set is below:

```
##    userId movieId rating timestamp                         title
## 1:      1     122      5 838985046              Boomerang (1992)
## 2:      1     185      5 838983525               Net, The (1995)
## 3:      1     292      5 838983421               Outbreak (1995)
## 4:      1     316      5 838983392               Stargate (1994)
## 5:      1     329      5 838983392 Star Trek: Generations (1994)
## 6:      1     355      5 838984474       Flintstones, The (1994)
##                            genres isDrama isComedy isAction
## 1:                Comedy|Romance   FALSE     TRUE    FALSE
## 2:         Action|Crime|Thriller   FALSE    FALSE     TRUE
## 3:  Action|Drama|Sci-Fi|Thriller    TRUE    FALSE     TRUE
## 4:       Action|Adventure|Sci-Fi   FALSE    FALSE     TRUE
## 5: Action|Adventure|Drama|Sci-Fi    TRUE    FALSE     TRUE
## 6:       Children|Comedy|Fantasy   FALSE     TRUE    FALSE
```

Key Steps

The ideal method for predicting would be linear regression. However, due to the size of the data set and my hardware limitations, a simpler approach will be needed. The prediction method that will be used for this is a modeling approach where we first assume the most simplistic model, that every predicted rating is the

average of all movie ratings. Then we will then look at subgroups of the data, split by certain characteristics, and see how the expected value of these subgroups differs from the mean of all ratings. Then a better model for a given predicted rating would be to take the average rating and add the difference of the rating's subgroup's average to the total average. Then to continually make the model better, we will create more partitions and compare the subgroup means to our model's predictions in order to generate more accurate models in a similar way to the previous step. To see this mathematically, we will define the prediction for $rating_i$ as Y_i and after each step of splitting up the data, we will have $partition_j$. The rating we wish to predict will be in a certain subgroup of the data, which we can denote as $subgroup_{ij}$ and we would like to find the average of that subgroup, $subgroup\_average_{ij}$. Then would like to find the difference of that subgroup average from the prediction provided by the previous terms of the model, denoted $subgroup\_error_{ij}$ or $SGE_{i,j}$ for short. Therefore for N partitions of the data, we can mathematically describe our prediction for $rating_i$ as:

$$Y_i = mean \ + \ SGE_{1,j} \ + \ SGE_{2,j} \ + ... + \ SGE_{n,j}$$

After we train the model in this way we will try to use regularization strategies to try and discount the smaller subgroups of data that may create outliers in our data, such as movies that receive very few ratings. Finally, we will test the accuracy of our model by predicting the rating on a previously unused data set that was partitioned for validation purposes and then we will find the Root Mean Square Error (RMSE), the same method for determining the accuracy in the original Netflix Competition.

Methods & Analysis

Process

We will employ various techniques to create the best possible prediction. Those techniques will be outlined briefly in this section. First, we must clean and prepare our data set for modeling. Next we will explore the data and visualize it so we can best understand it. Then we will build our model using a train set and continually test it against the test set. Finally, we will test the performance of our model.

Techniques

Data Cleaning

The data has already been thoroughly cleaned and organized by the GroupLens Team.

Data Preparation

To prepare for data modeling we will separate our data into a validation set (90% of the data) and the training set (10% of the data). Further, in order to test out different variations of models, we will separate our training set roughly equally into 2 cross-validation sets. Also, from the brief examination of the data above, we can see that genres are not given as single variables, but as sets of variables. In other words, we are only given a string of all the genres to which a movie applies. If we are to categorize genres into separate sets for our analysis, then all movies that contain one genre, for example 'Comedy' will not be grouped together, only movies that apply to a certain mix of genres will be grouped together. If we split up these groupings into individual variables, we can examine the frequencies that certain genres are attached to different ratings, seen below.

```
##     Drama    Comedy    Action  Thriller
##   3910127   3540930   2560545   2325899
## Adventure   Romance    Sci-Fi     Crime
##   1908892   1712100   1341183   1327715
```

```
##            Fantasy           Children             Horror            Mystery
##             925637             737994             691485             568332
##                War          Animation            Musical            Western
##             511147             467168             433080             189394
##          Film-Noir        Documentary              IMAX (no genres listed)
##             118541              93066               8181                  7
```

We can see that the most frequent genres are Comedy, Drama and Action. In preparing the data, we will add new boolean variables to each rating to signal whether they are associated with these genres: isComedy, isDrama and isAction. We will then integrate these new categorizations and see how they affect the RMSE.

Data Exploration of Training Set

Correlation of Rating Index with Rating Value

```
## [1] 0.00233064
```

Mean of All Ratings

```
## 3.51246520160155
```

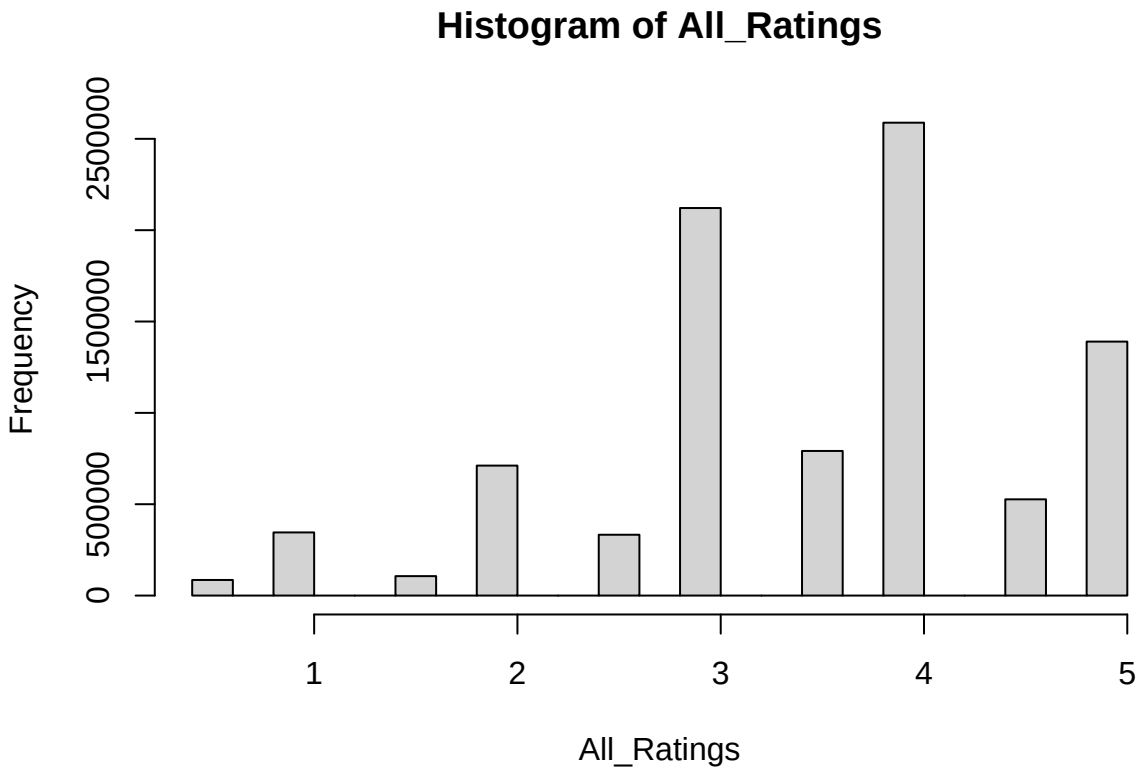
Median of All Ratings

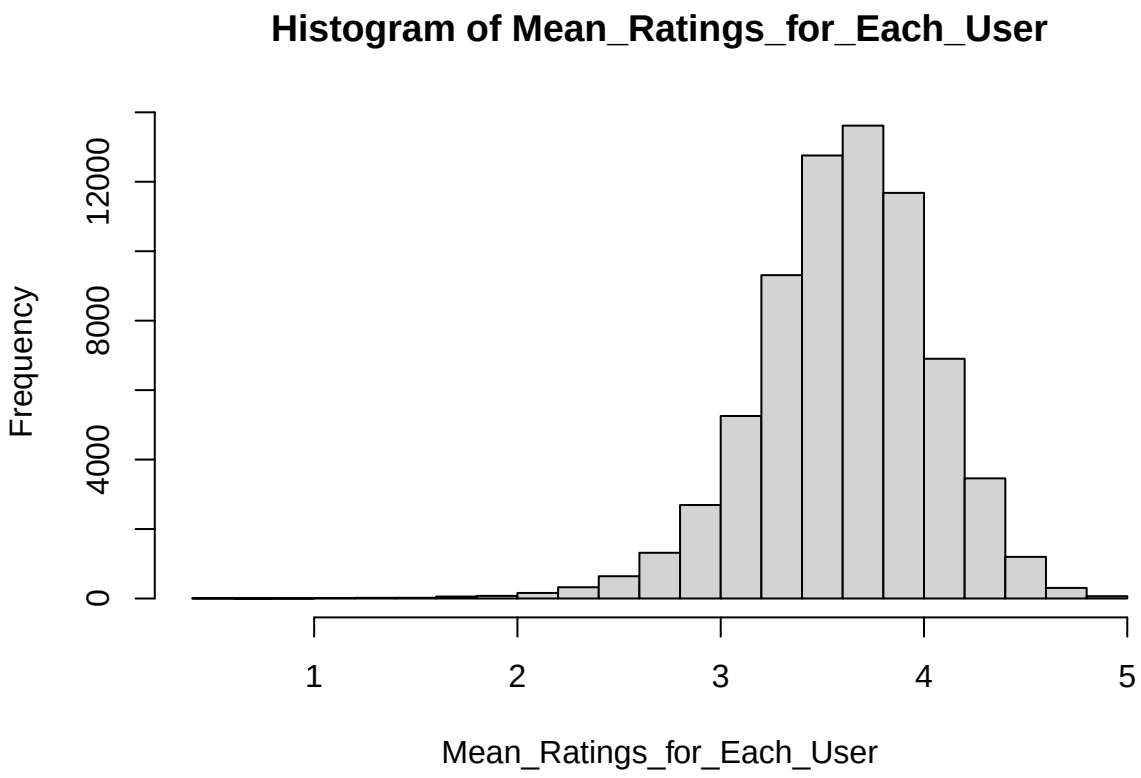
```
## 4
```

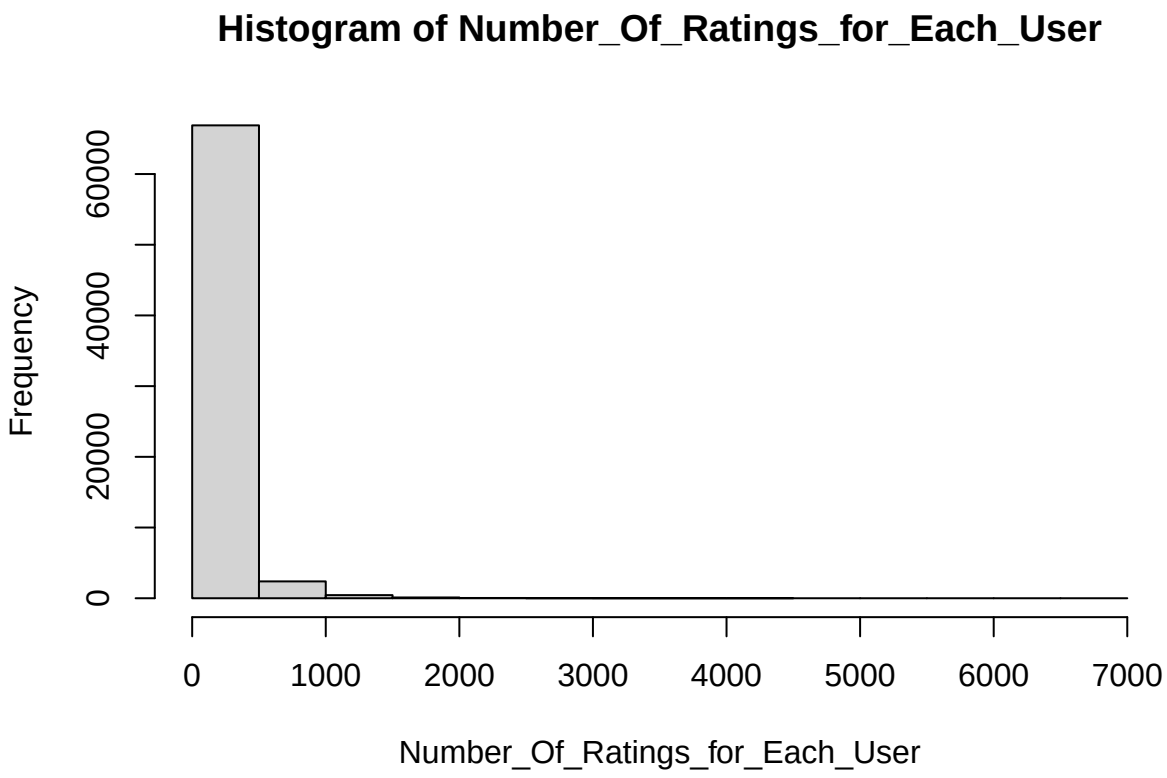
Mode of All Ratings

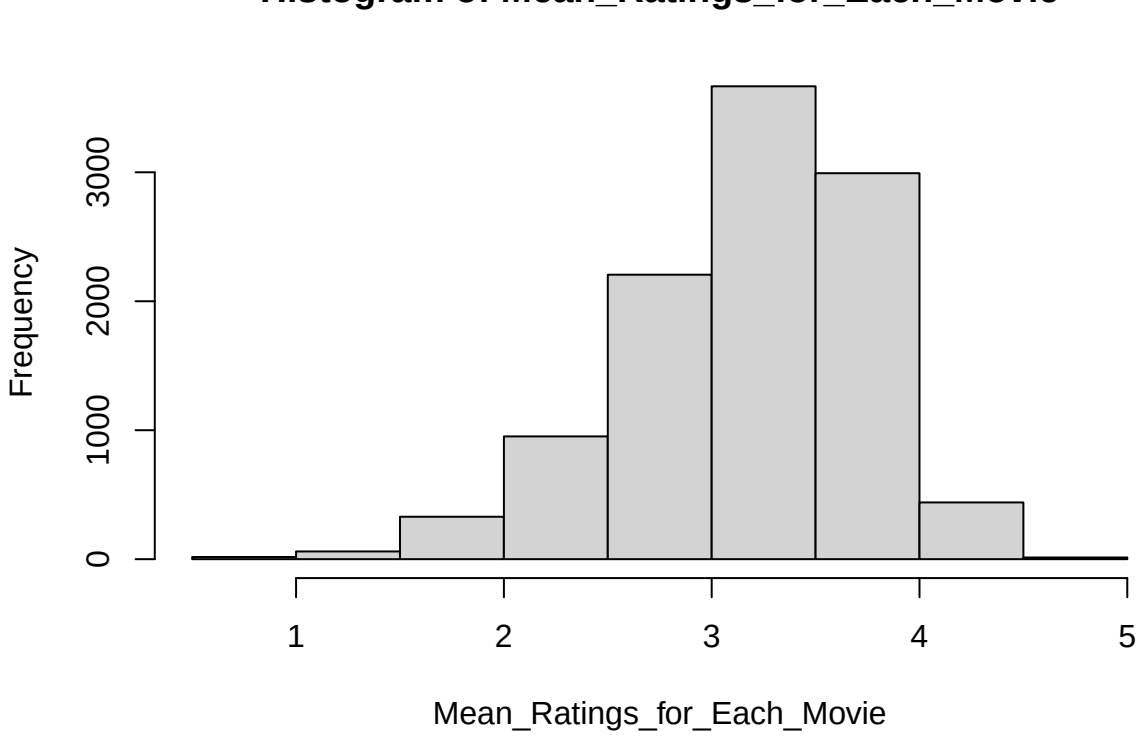
```
## 4
```

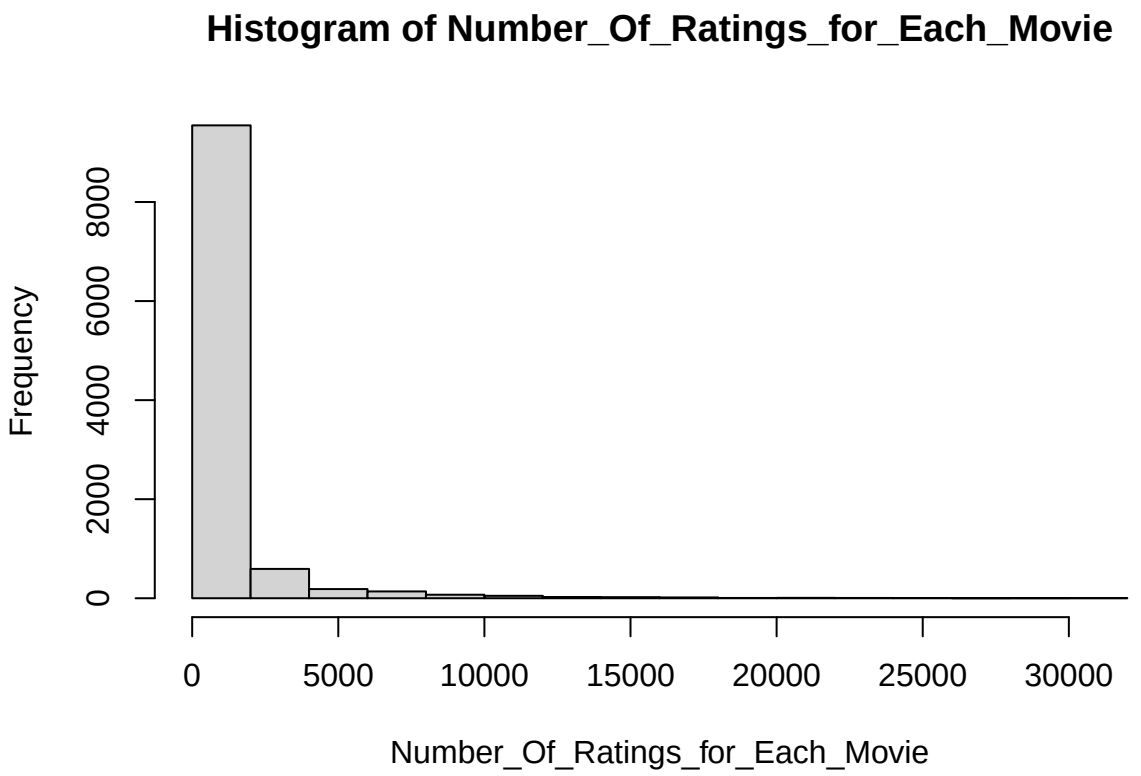
Data Visualization

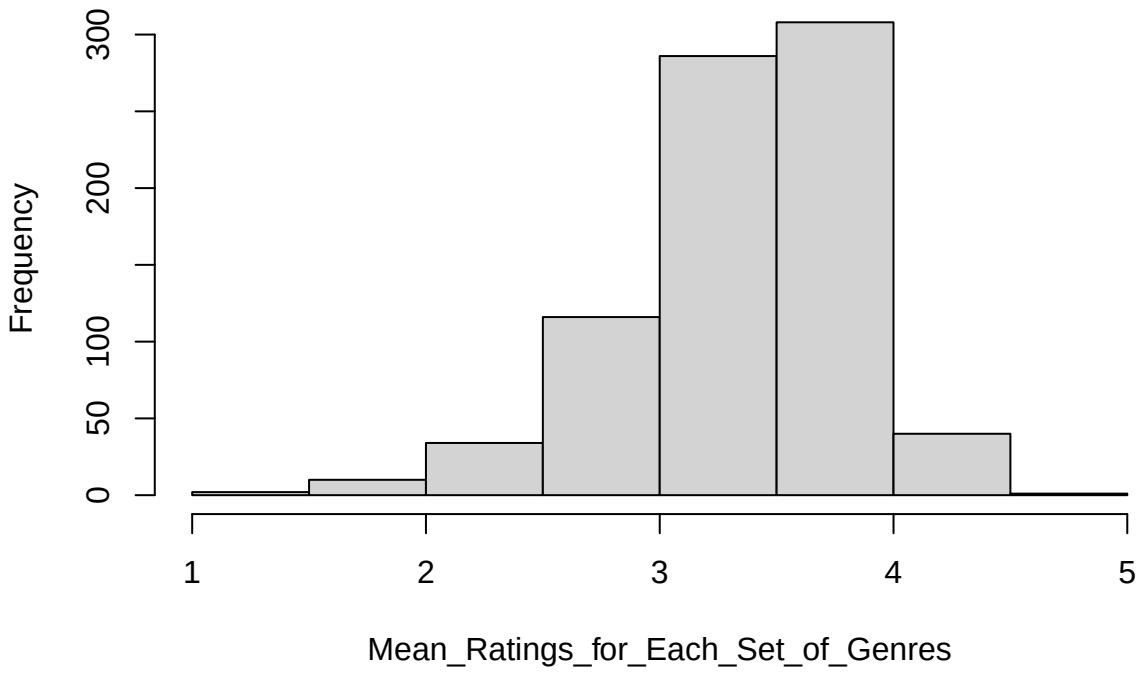


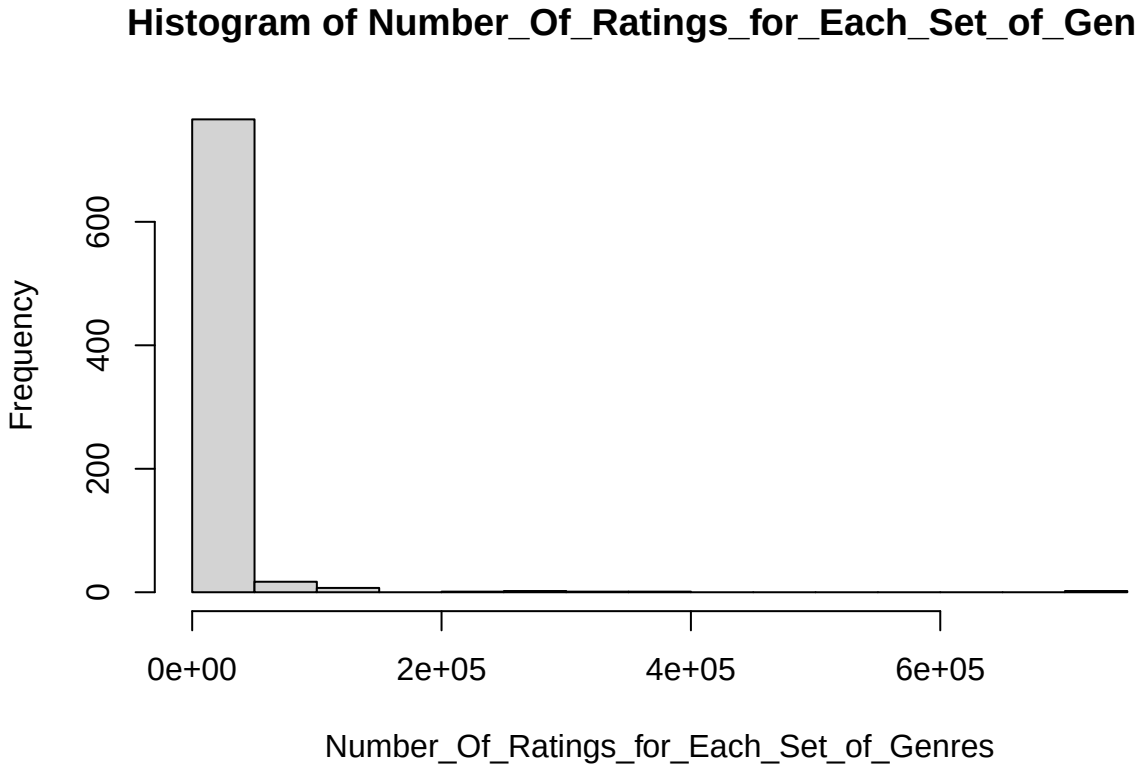



Histogram of Number_Of_Ratings_for_Each_User
Frequency
0
20000
40000
60000
0
1000
2000
3000
4000
5000
6000
7000
Number_Of_Ratings_for_Each_User


Histogram of Mean_Ratings_for_Each_Movie
Frequency
0
1000
2000
3000
1
2
3
4
5
Mean_Ratings_for_Each_Movie


Histogram of Number_Of_Ratings_for_Each_Movie
Frequency
0
2000
4000
6000
8000
0
5000
10000
15000
20000
25000
30000
Number_Of_Ratings_for_Each_Movie


Histogram of Mean_Ratings_for_Each_Set_of_Genres
Frequency
0
50
100
200
300
1
2
3
4
5
Mean_Ratings_for_Each_Set_of_Genres



Insights

Looking at the distribution of means within different categorizations (by user, movie and genre), we can see that they are somewhat normally distributed with a left skew. Because the raw values of the ratings and differences between ratings among these categorizations will be useful to our model we will not use mean normalization or scaling to try to normalize these distributions. Likewise, normalizing the data along one dimension may inhibit the normalization of this data along another axis.

Above we have validated that there is no correlation between row indexes and ratings in the training set, using Pearson's correlation coefficient ($r < .01$). This proves that the rows are sufficiently randomized to partition into randomized subsets that still sufficiently represent the total population.

We can also see in the histograms that the number of ratings per user, the number of ratings per movie, and the number of ratings per genre set are all right-skewed distribution, which shows that there are many ratings, users and genre sets with very little ratings. This makes sense to our assumptions that there are many more niche ratings and genres that only appeal and are rated by a small set of users. The data also validates the idea that there are many more disengaged users than engaged users.

Since we will create a model based on these categorizations, it would make sense for us to use regularization to minimize the value our model places on users, ratings and genre sets with low ratings. This is because there is likely not enough data for us to say how a user would rate a movie if they have not rated many movies nor if that movie or genre set does not have enough data to say what the average person would think of it.

Modeling Approach 1

First, we will create a model incrementally using the first cross-validation set. We will now look at the RMSE of a model that predicts a rating based on returning the mean rating of all ratings.

RMSE using Mean of All Ratings

```
## 1.06039011310668
```

Now we will make our model more complex by partitioning the data into different categories. For each category, we will find that category's average rating. We will see how that rating differs from our previous model with a hypothetical variable *effect* where *effect* is the *average for the category the rating is in* minus the *previous model's prediction for a rating*. We will then update our prediction model by adding each rating's *effect* to the *previous prediction* by its *effect*. This will get us to a closer prediction of the actual rating.

First we will apply this method to categorization based on the user ID associated with a rating. Then we will use other categorizations to see how updating the model based on those improves our model's RMSE.

RMSE after Updating model based on *userId*

```
## 0.966085665177983
```

RMSE after Updating model based on *genre*

```
## 0.931110667319105
```

RMSE after Updating model based on *movie*

```
## 0.869553495984921
```

RMSE after Updating model based on *isComedy*

```
## [1] 0.8695535
```

RMSE after Updating model based on *isDrama*

```
## [1] 0.8695535
```

RMSE after Updating model based on *isAction*

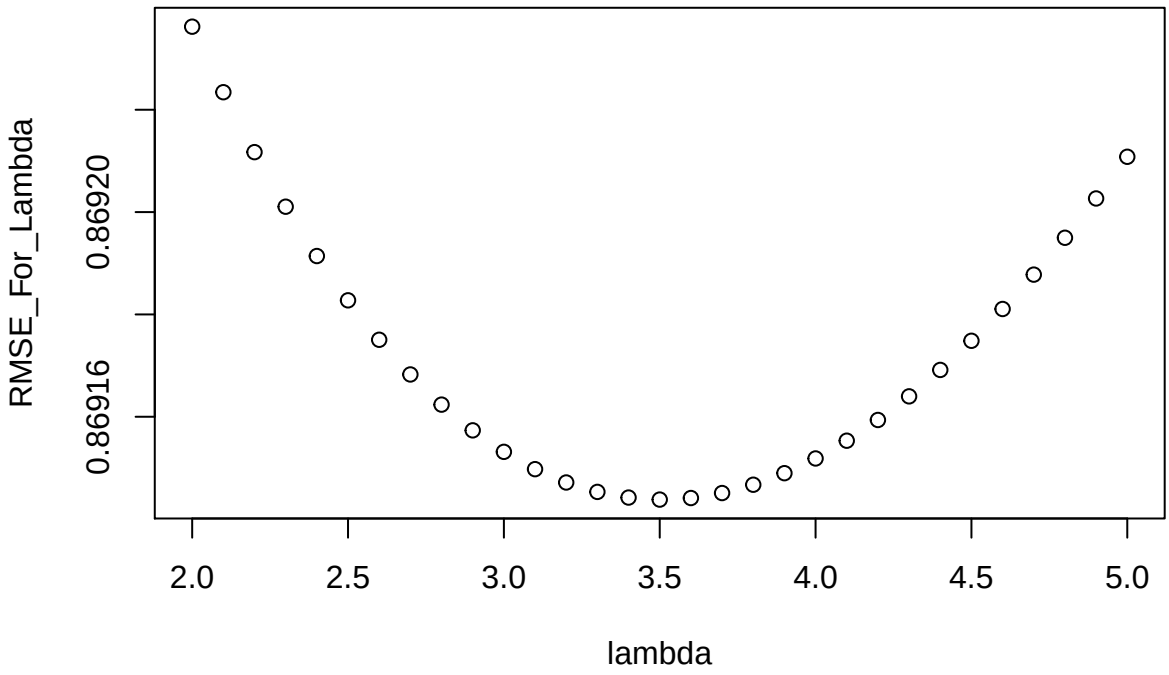
```
## [1] 0.8695535
```

Modeling Approach 2

We can see that much of our updating has decreased the RMSE. However, our new variables isComedy, isDrama, isAction did not decrease the RMSE significantly. Therefore we will leave those categorizations out in our next model. Our next model will repeat the steps of the previous one, except we will explore using regularization to mitigate the effect that categories with small amounts of ratings have on the final model. In other words, if a category has a small amount of ratings, in the model updating, we will minimize the *effect* it has on deviating updated predictions for ratings within that category from the previous prediction. We will do this by updating our equation for *effect* by modifying the term for the *average for the category the rating is in*. The mean of such a vector of ratings could be re-written in code as *mean(vector)* or alternatively *sum(vector)/length(vector)*. In order to minimize that term towards zero, more so with small values of *length(vector)*, we will add a constant value *lambda* to the denominator of that term. Also, since we do not know the optimal value of *lambda* to reduce the RMSE of our model, we will re-try the modeling technique with 31 different values of *lambda* from 2 to 5 at .1 increments. Since we are creating a new model, we will also now use our second cross-validation set to train this model.

RMSE of Models Varying by Lambda, followed by plot of values

```
##       lambda RMSE_For_Lambda
##  [1,]    2.0       0.8692362
##  [2,]    2.1       0.8692234
##  [3,]    2.2       0.8692117
##  [4,]    2.3       0.8692011
##  [5,]    2.4       0.8691914
##  [6,]    2.5       0.8691828
##  [7,]    2.6       0.8691750
##  [8,]    2.7       0.8691683
##  [9,]    2.8       0.8691624
## [10,]    2.9       0.8691573
## [11,]    3.0       0.8691531
## [12,]    3.1       0.8691498
## [13,]    3.2       0.8691472
## [14,]    3.3       0.8691453
## [15,]    3.4       0.8691442
## [16,]    3.5       0.8691438
## [17,]    3.6       0.8691441
## [18,]    3.7       0.8691451
## [19,]    3.8       0.8691467
## [20,]    3.9       0.8691490
## [21,]    4.0       0.8691519
## [22,]    4.1       0.8691553
## [23,]    4.2       0.8691594
## [24,]    4.3       0.8691640
## [25,]    4.4       0.8691692
## [26,]    4.5       0.8691748
## [27,]    4.6       0.8691811
## [28,]    4.7       0.8691878
## [29,]    4.8       0.8691950
## [30,]    4.9       0.8692027
## [31,]    5.0       0.8692108
```



Lambda Corresponding To The Minimum RMSE

```
##       lambda RMSE_For_Lambda
##  [1,]    2.0       0.8692362
##  [2,]    2.1       0.8692234
##  [3,]    2.2       0.8692117
##  [4,]    2.3       0.8692011
##  [5,]    2.4       0.8691914
##  [6,]    2.5       0.8691828
##  [7,]    2.6       0.8691750
##  [8,]    2.7       0.8691683
##  [9,]    2.8       0.8691624
## [10,]    2.9       0.8691573
## [11,]    3.0       0.8691531
## [12,]    3.1       0.8691498
## [13,]    3.2       0.8691472
## [14,]    3.3       0.8691453
## [15,]    3.4       0.8691442
## [16,]    3.5       0.8691438
## [17,]    3.6       0.8691441
## [18,]    3.7       0.8691451
## [19,]    3.8       0.8691467
## [20,]    3.9       0.8691490
## [21,]    4.0       0.8691519
## [22,]    4.1       0.8691553
## [23,]    4.2       0.8691594
## [24,]    4.3       0.8691640
## [25,]    4.4       0.8691692
## [26,]    4.5       0.8691748
## [27,]    4.6       0.8691811
## [28,]    4.7       0.8691878
## [29,]    4.8       0.8691950
```

```
## [30,]    4.9       0.8692027
## [31,]    5.0       0.8692108
```

Final Model

Process

Looking at the second model, we have seen that the optimal value of *lambda* to minimize the RMSE of that model is *3.5*. We will use that value with the 2nd modeling approach using regularization on the validation set to test the model for final results and get an RMSE.

Accuracy

Final RMSE

```
## 0.850136220110739
```

Performance

To assess the speed at which the final model took, we will at the number of seconds that it took to run on a specific computer.

Computer Specifications MacBook Pro (Retina, 13-inch, Early 2015) Processor: 3.1 GHz Dual-Core Intel Core i7 Memory: 16 GB 1867 MHz DDR3

Time in Seconds

```
## 2.12900000000002
```

Conclusion

Summary

We see now that the modeling approach outlined multiple times above was somewhat successful and regularization was effective in reducing the RMSE. During the final modeling, the RMSE was recorded of the simplified model using only the mean of all ratings, which we will call the "Baseline" RMSE. We will now see how that compares to the final RMSE and use that as a baseline to see how effective our optimization techniques were.

RMSE of "Baseline" RMSE

```
## 1.06120172208118
```

Absolute Difference between "Baseline"

```
## 0.211065501970441
```

Limitations

The limitations of this work discussed in this paper are a limitation on data size, computing power, time and money. While there is a fuller set of data for me to perform this analysis on, the reduction to a smaller subset was ultimately done for performance considerations due to lack of access to more powerful processors.

Future work

In the future, I hope to continue this work, using a larger data set and a higher-performing machine in which to do that analysis, to hopefully create more rigorous models for predicting movie ratings. Deep-learning and Bayesian models are an interesting area of research that I would like to explore more.