Movielens Project Submission

Johan Wikström

03 april 2019

Contents

1 Introduction

Trying to estimate the rating a specific individual will give a specific movie is a important but difficult task. The purpose of the project described in this report is to attack this task; using rating data provided by Netflix, and methods learned in the Data Science Professional Certificate Program held by HarvardX on edx.org.

The data provided by Netflix consists of ten million rows, where each row represents a rating made by a specific user for a specific movie. The rows consists of: the user who gave the rating, the movie rated, rating given, movie title, genres of the movie, and the date the rating was given. A small sample of the data can be seen in the table below.

userId	movieId	rating	title	genres	date
1	122	5.00	Boomerang (1992)	Comedy|Romance	1996-08-02
1	185	5.00	Net, The (1995)	Action|Crime|Thriller	1996-08-02
1	292	5.00	Outbreak (1995)	Action|Drama|Sci-Fi|Thriller	1996-08-02
1	316	5.00	Stargate (1994)	Action|Adventure|Sci-Fi	1996-08-02
1	329	5.00	Star Trek: Generations (1994)	Action|Adventure|Drama|Sci-Fi	1996-08-02
1	355	5.00	Flintstones, The (1994)	Children|Comedy|Fantasy	1996-08-02

The first step taken to approach the problem was to create a simple model. Then, analyse the errors made by the model to find trends, patterns, and model flaws. Methods to leverage these trends and patterns was then incrementally integrated into the model, evolving it, making it more complete and flaws could be minimized. Examples of methods used to minimize flaws and leverage trends and patterns are K-fold cross validation to fit model parameters, Regularization to constrain total variability, and Matrix Factorization to model features.

To measure the performance of the model, the random mean square error, further denoted as RMSE; Was chosen by the course staff. The final model reached an RMSE of 0.79, and consisted of a mean constant, a regularized user bias, a regularized movie bias, and a set of forty features.

2 Analysis and Methods

This chapter provides a brief statistical analysis of the data, followed by more in depth analysis to explore and find possibilities to improve the model. With each finding, the method used to address that finding is also presented.

The data set provided by Netflix consists of ratings given a movie by a user. The total data set consisted of 10 000 054 ratings given to 10 677 unique movies by 69 878 unique users.

The ratings can be represented as a matrix Y where each row is a user, each column is a movie, and the cell values are the ratings.

$$Y = \begin{bmatrix} y_{1,1} & y_{1,2} & \cdots & y_{1,I} \\ y_{2,1} & y_{2,2} & \cdots & y_{2,I} \\ \vdots & \vdots & \ddots & \vdots \\ y_{J,1} & y_{J,2} & \cdots & y_{J,I} \end{bmatrix}, \tag{1}$$

where $y_{j,i}$ is the rating given for movie i by user j. In this representation, the matrix Y is very sparse, with only about 1.3% of the matrix being known.

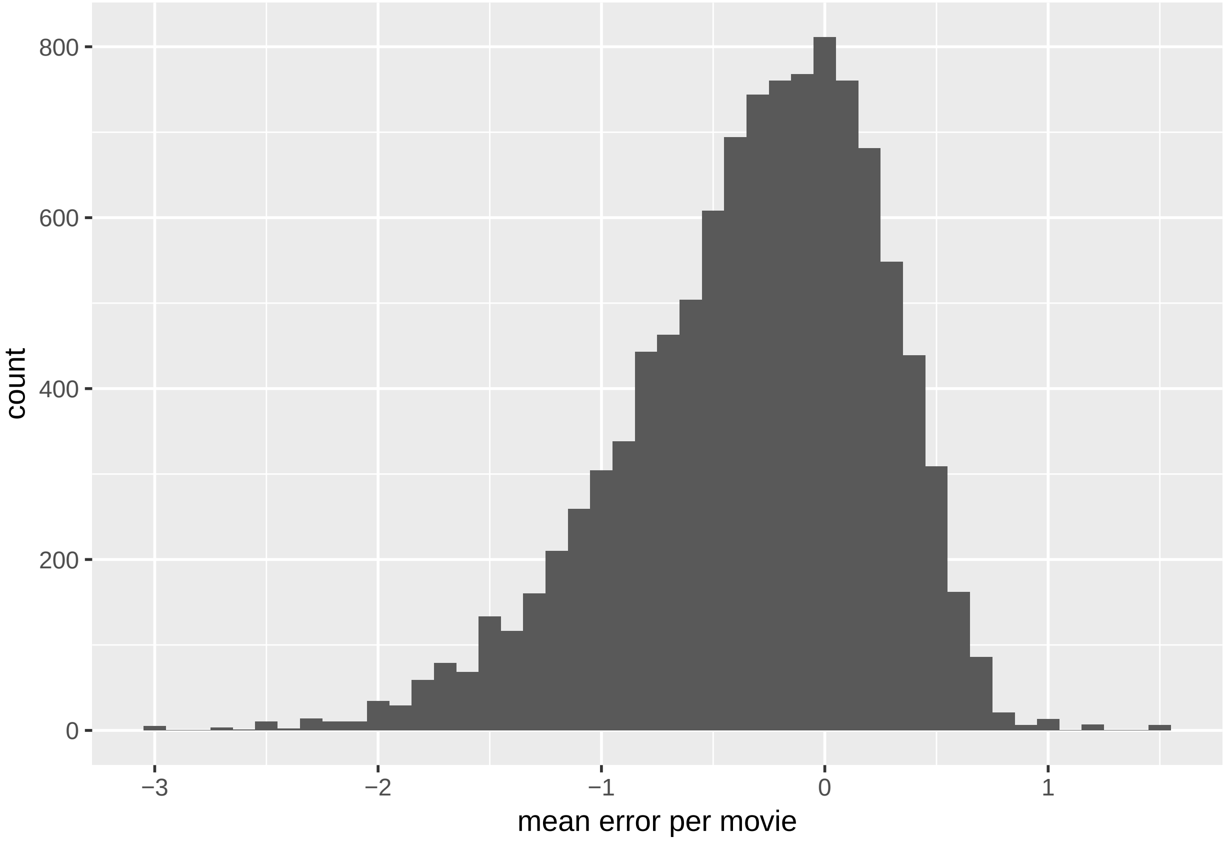
The goal is then trying to estimate unknown ratings $\hat{y}_{j,i}$ that minimizes the RMSE

$$\sqrt{\frac{1}{N}\sum_{J,I}(y_{j,i}-\hat{y}_{j,i})^2}. \tag{2}$$

A very simple model, and a good place to start, is to estimate an unknown rating to be the mean rating for every movie and user pair.

$$\hat{y}_{j,i} = \hat{\mu}, \tag{3}$$

where $\hat{\mu}$ is the mean of all ratings. If we examine the mean errors made for each movie, $\hat{\mu} - y_{j,i}$, shown in the histogram below. We can see that some movies tends to be given higher ratings than the average and some tends to be given lower ratings.



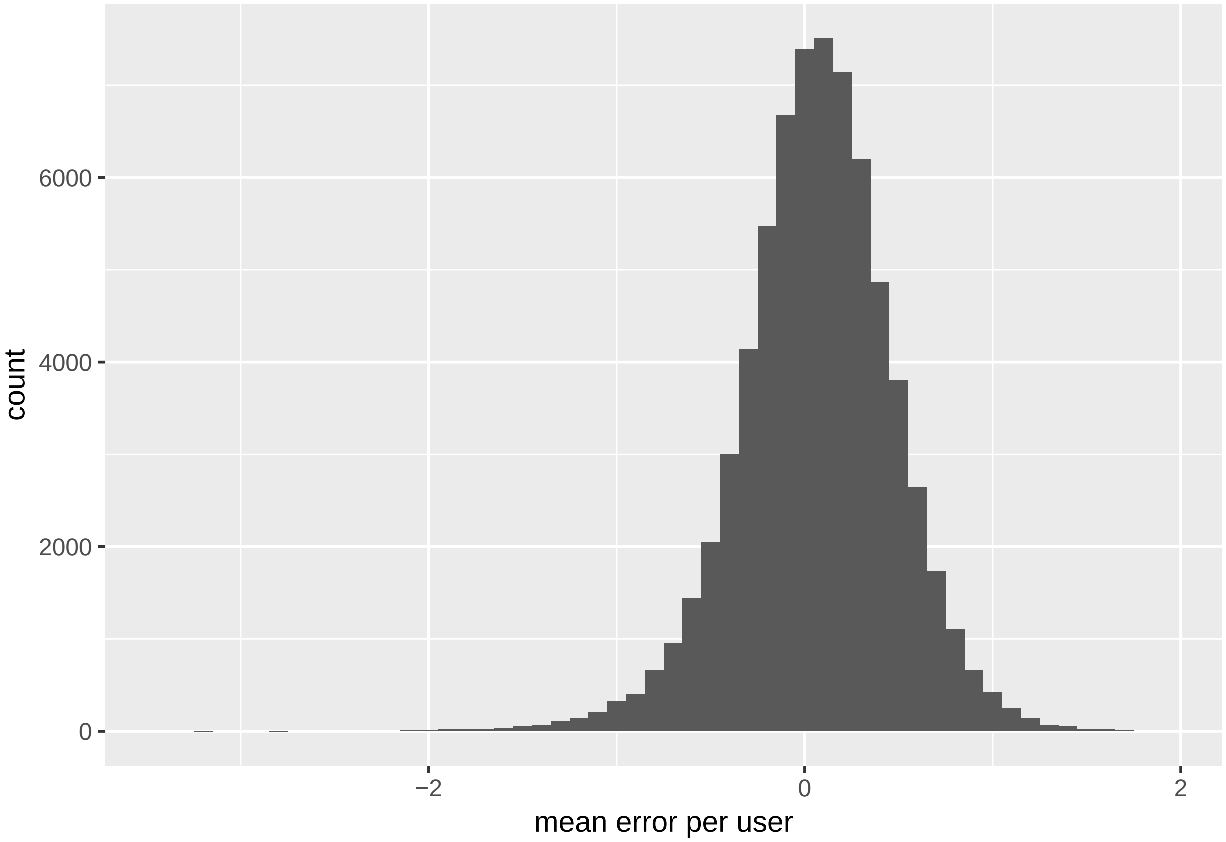
This pattern is quite intuitive, some movies are good and some are bad. Leading to good movies being rated higher compared to bad movies on average. To take this pattern into account, a movie bias term is added to the simple model

$$\hat{y}_{j,i} = \hat{\mu} + \hat{b}_i, \tag{4}$$

where $\hat{b}_i$ is the movie bias estimate for movie i. The movie bias can be estimated using least squares, in this case however it can be calculated as the mean error made by the simple model (3) for each movie for each movie i:

$$\hat{b}_i = \frac{1}{n_i}\sum_{j=1}^{n_i} y_{j,i} - \hat{\mu}, \tag{5}$$

where n_i is the number of ratings for movie i. If we then further examine the errors made by our model with movie biases (4), we see a similar pattern for users which is visualized in the histogram below.



It seems that some users tend to give movies higher ratings then the average, while some tend to give lower ratings. To take this user bias into account, we add a user bias term, $\hat{b}_j$ to the model (4).

$$\hat{y}_{j,i} = \hat{\mu} + \hat{b}_i + \hat{b}_j. \tag{6}$$

The user bias can be estimated using least squares minimization, which will result in the mean error for every user with the movie bias model (4).

$$\hat{b}_j = \frac{1}{n_j}\sum_{i=1}^{n_j} y_{j,i} - \hat{\mu} - \hat{b}_i, \tag{7}$$

where n_j is number of ratings made by user j.

By estimating b_i and b_j using least squares, the mean for each movie i and user j respectively, the bias estimates will become susceptible to noisy estimates due to a low number of observations for some movies and user.

For example, if a movie has been rated few times, then each of those ratings will have a large influence on the estimates made for that movie. Then, when we estimate the rating for the same movie but from another user, that estimate will be highly influenced by just a few ratings and probably be far from the true bias due to the noise introduced. The noise is very apparent if a movie has only been rated once, or a user only has rated once. Then, the bias estimate will be error made by that single observation, for example if movie i has only been rated by user 250 then the bias will be estimated as $b_i = y_{i,j} - \hat{\mu}$.

This effect can be seen if we look at the largest estimated movie biases, which can be seen in the table 1. We see that both the largest, positive and negative, biases belong to movies with few ratings.

movieId	b_i_hat	number_of_ratings
5805	-3.01	2
8394	-3.01	1
61768	-3.01	1
51209	1.49	1
53355	1.49	1
64275	1.49	1

Table 1: Largest, negative and postive, movie biases

The same noisy estimates can be seen for users where users with few rated movies have the largest biases, shown in the table 2 below.

userId	b_j_hat	number_of_ratings
13496	-3.39	17
48146	-3.25	25
49862	-3.14	17
46484	1.79	24
18591	1.86	19
13524	1.89	20

Table 2: Largest, negative and postive, user biases

The method used for handling these noisy estimates is regularization. Regularization aims at controlling the total variability of the movie biases $\sum_{j=1}^{n_i} \hat{b}_j^2$ by minimizing an equation that adds a penalty for large estimates instead of minimizing the least squares as we did in (7) and (**??**). The function that will be minimized is very similar to the least squares:

$$\frac{1}{N}\sum_{j,i}(y_{j,i} - \hat{\mu} - \hat{b}_j - \hat{b}_i) + \lambda_j \sum_j \hat{b}_j^2 + \lambda_i \sum_i \hat{b}_i^2\Big), \tag{8}$$

where the first term is the regular least squares equation, and the second and third terms are penalties that gets larger when many bias estimates are large. λ_i and λ_j are constants for scaling the penalties for movies and users respectively.

Using calculus, we can derive that the movie biases and user biases that minimizes function (8) are:

$$\hat{b}_i = \frac{1}{\lambda_I + n_j}\sum_{u=1}^{n_j} y_{u,i} - \hat{\mu}, \tag{9}$$

and

$$\hat{b}_j = \frac{1}{\lambda_J + n_i} \sum_{u=1}^{n_i} y_{j,u} - \hat{\mu} - \hat{b}_i. \tag{10}$$

Now, the next problem is to choose suitable values for λ_i and λ_j. These constants will affect the resulting estimates, and the estimate can therefore be seen as a function of these constants, $\hat{y}_{j,i}(\lambda_i, \lambda_j)$. Our goal is then to determine which constants that minimize the estimated RMSE (2). To minimize the RMSE, a simple grid method was used where a grid of different constant values, $\lambda_i = [3, 4.., 7]$ and $\lambda_j = [4, 5..7]$, was constructed, and the RMSE estimate for each combination calculated. It is important to note that the RMSE we calculate and will attempt to minimize is an estimation of the true RMSE. To achieve a fair estimation of the true RMSE, K-fold cross validation, with five folds, was performed. That means that the training set was split into five equally large, non overlapping, random sets. With these sets, the RMSE estimations for the grid was calculated five times, each time with a different training and test set. Then, the mean RMSE for each combination of constants was calculated from the five optimizations, and the constants resulting in the lowest estimated mean RMSE was chosen. Which was $[\lambda_i = 3, \lambda_j = 5]$.

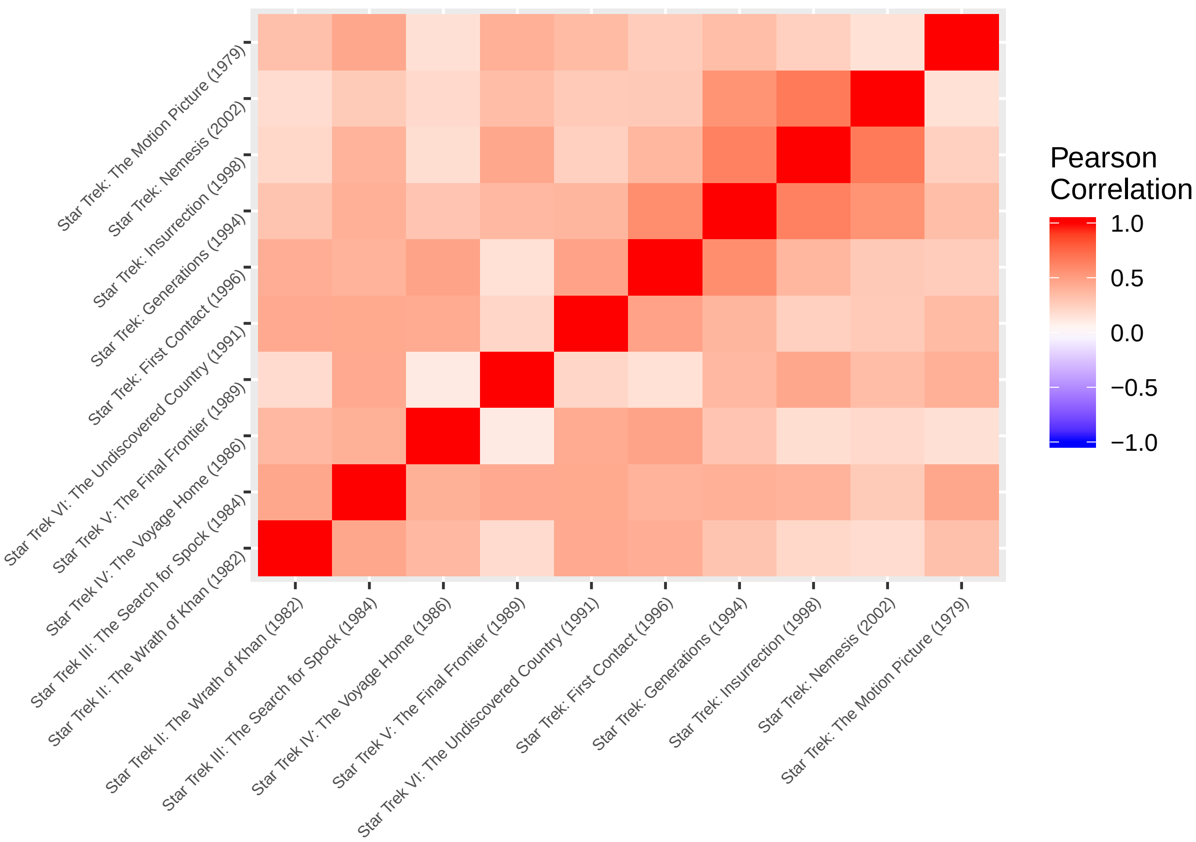
Now, we have created a model (4), which takes into account the difference in movies and user. The next step would be to incorporate knowledge gained from the ratings given by a user, and the ratings received for a movie. Specifically, a users movie preference. For example, some users might like science-fiction movies, while some user might dislike science-fiction movies. In the same sense, users probably have preferences for a certain movie series, actors, or other more difficult attributes such as sceneries, soundtracks, or colors.

To start, we define the true rating $y_{j,i}$ as the predicted rating $\hat{y}_{j,i}$ with an additional error, $r_{j,i}$

$$y_{j,i} = \hat{y}_{j,i} + r_{j,i}. \tag{11}$$

We can then analyse the patterns in the errors made $r_{j,i}$. One way of confirming that there are unexplained patterns, is to examine the correlation between errors. For example, if we look just at the correlation between errors made for the ten Star Trek Movies and the same users. We should see a pattern if there exists a preference resulting in some users enjoying the Star Trek Movies more and some users dislike them.

In the heat map below, the Pearson correlation has been calculated between the ratings given to the Stark Trek Movies, only user that has rated five or movies was included. A high positive correlation can be seen across all movies, which supports the argument above regarding the existence of preferences.



To find these preferences, a method called Funk SVD was used, which uses matrix factorization to estimate the errors $r_{j,i}$. First, we represent the errors as a matrix R

$$R = \begin{bmatrix} r_{1,1} & r_{1,2} & \dots & r_{1,I} \\ r_{2,1} & r_{2,2} & \dots & r_{2,I} \\ \vdots & \vdots & \ddots & \vdots \\ r_{J,1} & r_{J,2} & \dots & r_{J,I} \end{bmatrix} \in \mathbb{R}^{J \times I}, \text{ where } r_{j,i} = y_{j,i} - \hat{\mu} - \hat{b}_i - \hat{b}_j. \tag{12}$$

The rows represents each user and the columns each movie, each cell then contains the error made or missing values for movies not yet rated by certain users. The Funk SVD method tries to estimate the complete matrix, $\hat{R}$ using two lower dimensional matrices, U and V:

$$\hat{R} = U * V^t, \tag{13}$$

where $U \in \mathbb{R}^{J \times N}$, $V \in \mathbb{R}^{I \times N}$, and N is the number of preferences or so called features calculated. Each user has N number of user latent factors, which will be estimated, defining that user's preferences. Similarly each movie has N number of movie latent factor, defining that movie's features. The two matrices can be calculated using gradient decent. When U and V are calculated, we can estimate the errors $\hat{r}_{j,i}$ by multiplying theses two matrices:

$$\hat{r}_{j,i} = \sum_{n=1}^{N} u_{j,n} * v_{n,i}, \tag{14}$$

which results in the error estimation matrix $\hat{R}$. Due to the large size of R, a couple of GBs, the calculation of U and V was very memory and time consuming. The calculation was run on an AWS server with 32 GB of ram and took around 36 hours to complete.

Note that R is incomplete with missing cell values while $\hat{R}$ is complete. Giving us the opportunity to leverage the knowledge about a user's preferences and a movie's features to more accurately estimate the rating a user will give each movie.

Applying this knowledge, we define the final model as:

$$\hat{y}_{j,i} = \hat{\mu} + \hat{b}_i + \hat{b}_j + \sum_{n=1}^{N} u_{j,n} * v_{n,i}, \tag{15}$$

which uses the estimated mean, movies bias, user bias, user preferences, and movie features.

3 Results

Applying the final model (15), on the validation set resulted in an RMSE of 0.79. Summary statistics of the model parameters are shown below. In table 3 is the value of $\hat{\mu}$.

$\hat{\mu}$	3.51

Table 3: $\hat{\mu}$

Summary statistics for the movie and user biases, $\hat{B}_i$ and $\hat{B}_j$, in table 4.The user och movie biases have very similar statistics, their mins and maxes are similar, the same for the percentiles. However, the movies biases seem to be distributed slightly to the negative numbers while the user biases are close to zero, slightly positive.

	mean	median	25th percentile	75th percentile	max	min
$\hat{B}_i$	-0.32	-0.24	-0.67	0.10	1.49	-3.01
$\hat{B}_j$	0.06	0.07	-0.18	0.32	1.89	-3.39

Table 4: Statistics of the final user movie and user biases

Statistics of the movie features in matrix V in table 5. Overall, there seems to be no statistics that stands out. The first set of features seems to negative ones. The last set seem to be slightly positive.

Feature	mean	median	25th percentile	75th percentile	max	min
Feature 1	-0.44	-0.41	-0.62	-0.23	0.23	-1.77
Feature 2	0.25	0.21	-0.01	0.50	1.45	-1.44
Feature 3	-0.27	-0.23	-0.44	-0.07	0.63	-1.19
Feature 4	0.21	0.23	0.05	0.41	1.25	-1.47
Feature 5	-0.08	-0.05	-0.26	0.13	1.21	-1.85
Feature 6	-0.02	-0.02	-0.23	0.20	1.13	-1.24
Feature 7	-0.04	-0.06	-0.24	0.14	1.32	-1.07
Feature 8	-0.01	-0.02	-0.19	0.16	1.37	-1.02
Feature 9	0.05	0.06	-0.11	0.23	1.22	-1.14
Feature 10	0.09	0.10	-0.06	0.26	1.28	-1.05
Feature 11	0.11	0.10	-0.04	0.24	1.23	-1.04
Feature 12	0.04	0.04	-0.11	0.19	1.18	-0.98
Feature 13	0.07	0.07	-0.06	0.20	1.04	-1.08

Feature 14	0.06	0.05	-0.07	0.19	1.49	-0.81
Feature 15	0.04	0.04	-0.08	0.16	1.02	-1.38
Feature 16	0.06	0.06	-0.07	0.19	1.03	-1.24
Feature 17	0.06	0.06	-0.07	0.20	1.01	-0.99
Feature 18	0.05	0.04	-0.07	0.16	1.38	-0.93
Feature 19	0.08	0.08	-0.04	0.21	1.09	-1.58
Feature 20	0.08	0.08	-0.04	0.20	1.00	-0.86
Feature 21	0.07	0.07	-0.05	0.19	0.85	-0.98
Feature 22	0.08	0.07	-0.03	0.19	1.01	-0.83
Feature 23	0.06	0.06	-0.05	0.17	1.13	-0.78
Feature 24	0.12	0.12	0.01	0.24	0.91	-0.80
Feature 25	0.04	0.04	-0.06	0.15	0.92	-0.81
Feature 26	0.10	0.10	-0.01	0.22	1.05	-0.75
Feature 27	0.07	0.08	-0.03	0.18	1.03	-0.88
Feature 28	0.09	0.09	-0.01	0.19	1.08	-0.75
Feature 29	0.09	0.09	0.01	0.18	0.89	-0.88
Feature 30	0.10	0.09	-0.01	0.20	0.93	-0.81
Feature 31	0.08	0.08	-0.01	0.17	0.94	-0.82
Feature 32	0.09	0.09	-0.00	0.19	0.90	-0.81
Feature 33	0.10	0.10	0.03	0.17	0.77	-0.64
Feature 34	0.09	0.09	-0.00	0.18	0.87	-0.93
Feature 35	0.11	0.10	0.09	0.12	0.43	-0.13
Feature 36	0.11	0.10	0.09	0.12	0.25	-0.03
Feature 37	0.10	0.10	0.09	0.10	0.18	0.01
Feature 38	0.10	0.10	0.09	0.11	0.18	0.04
Feature 39	0.10	0.10	0.09	0.10	0.16	0.04
Feature 40	0.10	0.10	0.09	0.11	0.15	0.05

Table 5: Statistics of the movie features

And finally, statistics of the user preferences in matrix U in table 6. All preferences seem to be slightly positive, most of them centered around 0. An interesting statistic that stands out are the maxes. For the first set of preferences, the max is very large. This can mean that some user have a very strong preferences in regards to the first set of features.

Preference	mean	median	25th percentile	75th percentile	max	min
Preference 1	0.28	0.26	0.19	0.35	9.08	-0.63
Preference 2	0.06	0.07	-0.15	0.28	4.47	-1.45
Preference 3	0.28	0.27	0.16	0.38	2.73	-6.61
Preference 4	0.08	0.08	-0.11	0.27	2.09	-2.18
Preference 5	0.07	0.07	-0.12	0.26	1.84	-2.14
Preference 6	0.03	0.03	-0.15	0.20	2.02	-1.61
Preference 7	0.04	0.05	-0.12	0.21	1.64	-1.65
Preference 8	0.05	0.06	-0.11	0.21	2.08	-1.70
Preference 9	0.05	0.05	-0.09	0.19	2.28	-1.50
Preference 10	0.05	0.06	-0.09	0.20	1.77	-1.70
Preference 11	0.05	0.06	-0.07	0.18	1.61	-1.49
Preference 12	0.03	0.04	-0.10	0.17	1.19	-1.56
Preference 13	0.05	0.05	-0.07	0.17	1.46	-1.52
Preference 14	0.06	0.07	-0.05	0.18	1.18	-1.63
Preference 15	0.06	0.06	-0.05	0.17	1.34	-1.34
Preference 16	0.07	0.07	-0.04	0.18	1.03	-1.23
Preference 17	0.06	0.07	-0.05	0.18	0.98	-1.65
Preference 18	0.08	0.09	-0.01	0.18	1.25	-1.02

Preference 19	0.08	0.08	-0.03	0.18	1.19	-1.10
Preference 20	0.07	0.08	-0.03	0.18	1.17	-1.25
Preference 21	0.08	0.08	-0.02	0.18	1.18	-0.99
Preference 22	0.09	0.09	-0.00	0.18	0.91	-0.97
Preference 23	0.08	0.09	-0.02	0.18	1.18	-1.13
Preference 24	0.09	0.09	-0.00	0.18	1.47	-0.97
Preference 25	0.08	0.09	-0.01	0.18	1.48	-0.91
Preference 26	0.09	0.09	0.00	0.18	0.88	-1.17
Preference 27	0.09	0.09	-0.00	0.18	1.17	-0.91
Preference 28	0.09	0.09	0.00	0.18	1.11	-1.03
Preference 29	0.10	0.10	0.02	0.17	0.98	-0.75
Preference 30	0.09	0.09	0.01	0.17	1.01	-0.94
Preference 31	0.09	0.09	0.01	0.17	1.02	-0.98
Preference 32	0.09	0.09	0.01	0.17	1.13	-1.01
Preference 33	0.10	0.10	0.04	0.17	0.94	-0.66
Preference 34	0.09	0.09	0.02	0.17	0.82	-0.73
Preference 35	0.11	0.10	0.09	0.12	0.70	-0.08
Preference 36	0.10	0.10	0.09	0.11	0.23	-0.54
Preference 37	0.10	0.10	0.10	0.10	0.67	0.04
Preference 38	0.10	0.10	0.10	0.10	0.20	-0.38
Preference 39	0.10	0.10	0.10	0.10	0.55	0.04
Preference 40	0.10	0.10	0.10	0.10	0.17	-0.25

Table 6: Statistics of the user preferences

4 Conclusion and Discussion

In regards of the resulting RMSE, I am very pleased. The data is complex to create model against due to it's seemingly non-informative nature. And I think the method used in this project have managed to capture, in a somewhat good way, the information that can be extracted from the data, namely the individual users preference and each movies features.

I could however been more efficient in some of the methodology. For example I believe the use of cross validation to estimate λ_j and λ_i was a bit overkill since the sheer amount of data might have been sufficient to get fairly good estimates. In that sense, another optimization method could have been applied to fine tune the value of these parameters. Also, more time could have been spent on investigating the results och matrix factorization. For example, it would have been interesting to see the effect different number of latent factors would have made to the results. And an attempt to explain the preference and feature the latent factors represented.

In regards of scalability, I think this method is applicable since the re-calibration of the matrix factorization, which required most time and computing power, is easier than the initial calculation. This since users and movies can be bound into the existing vectors U and V.