Netflix Recomendation System - HarvardX Capstone Report

Thierry Morvany

09/09/2021

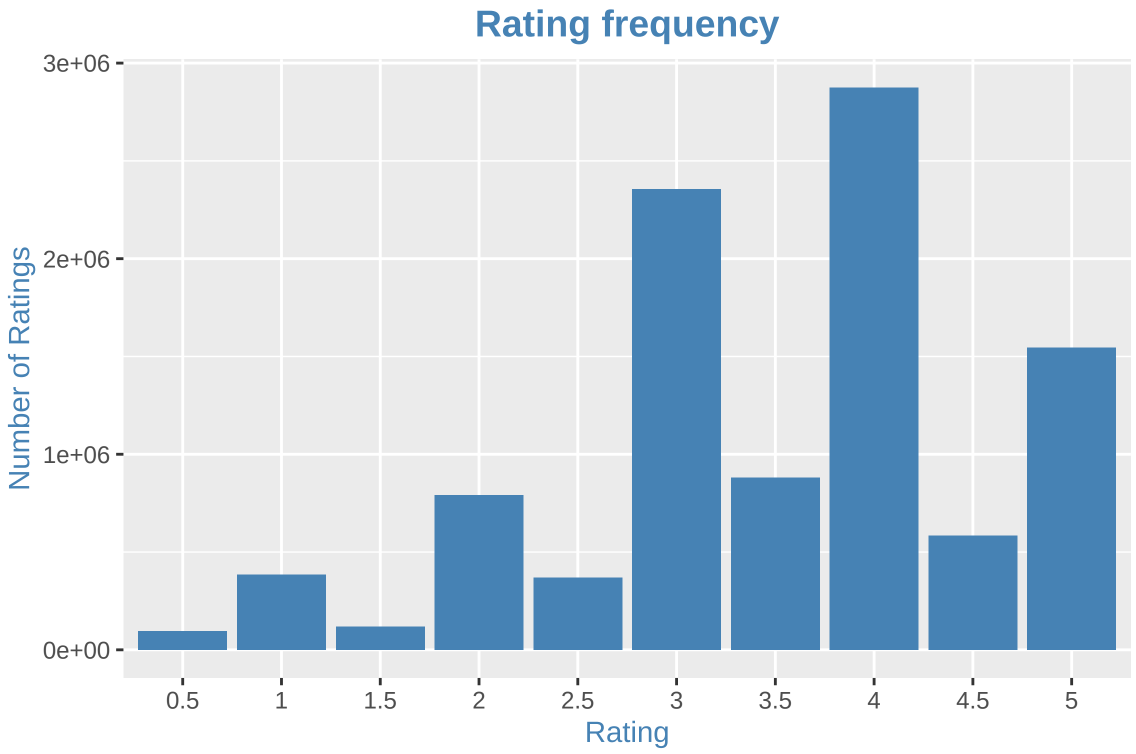
Contents

I- Introduction

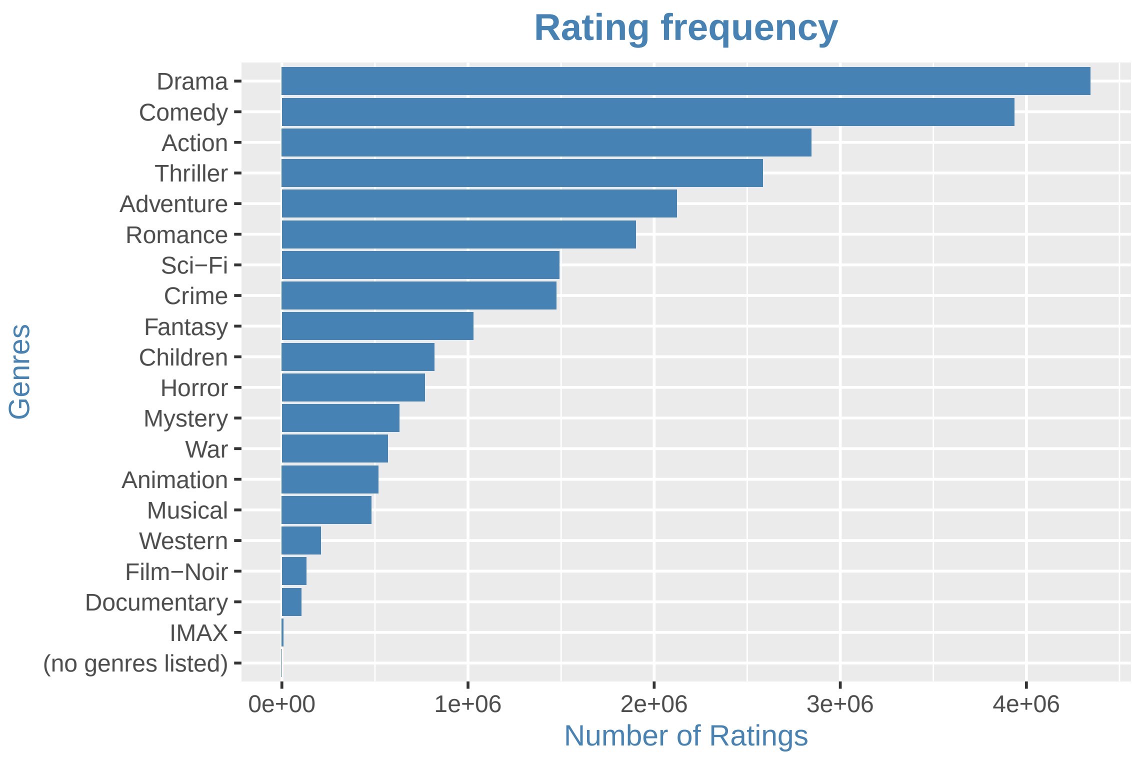
The Movielens project is part of the final course in the **HarvardX Professional Certificate in Data Science**. And this report is part of the Movielens project. The challenge is to create and evaluate prediction models that will predict the ratings some Netflix users would assign to some movies. The Netflix data is not publicly available so I used the dataset MovieLens proposed by HarvardX and provided by *GroupLens research Lab*. The dataset consists of 69878 users who rated 10677 movies grouped in 797 combinations of 20 distinct genres. In this report, I start with a shallow analysis of the dataset then, I describe the different models evaluated to develop a recommendation system to be applied to Netflix movies ratings.
Please, note that my 16GB-ram laptop was not powerfull enough all along. I then had to create an instance -of type r5.4xlarge on an Amazon Web Services (AWS) account to run the code built with R version 4.0.2. The creation procedure is described *here*.

II- Data Analysis

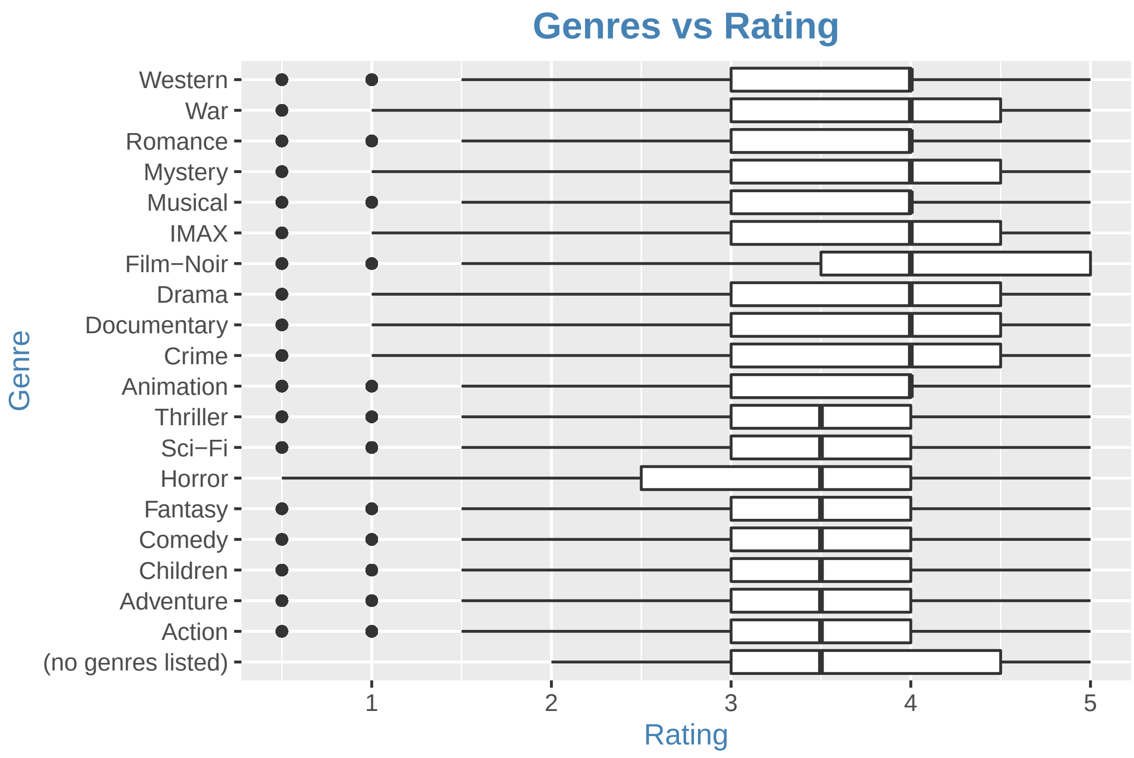
I start with a quick overview of the ratings distributed across the genres:



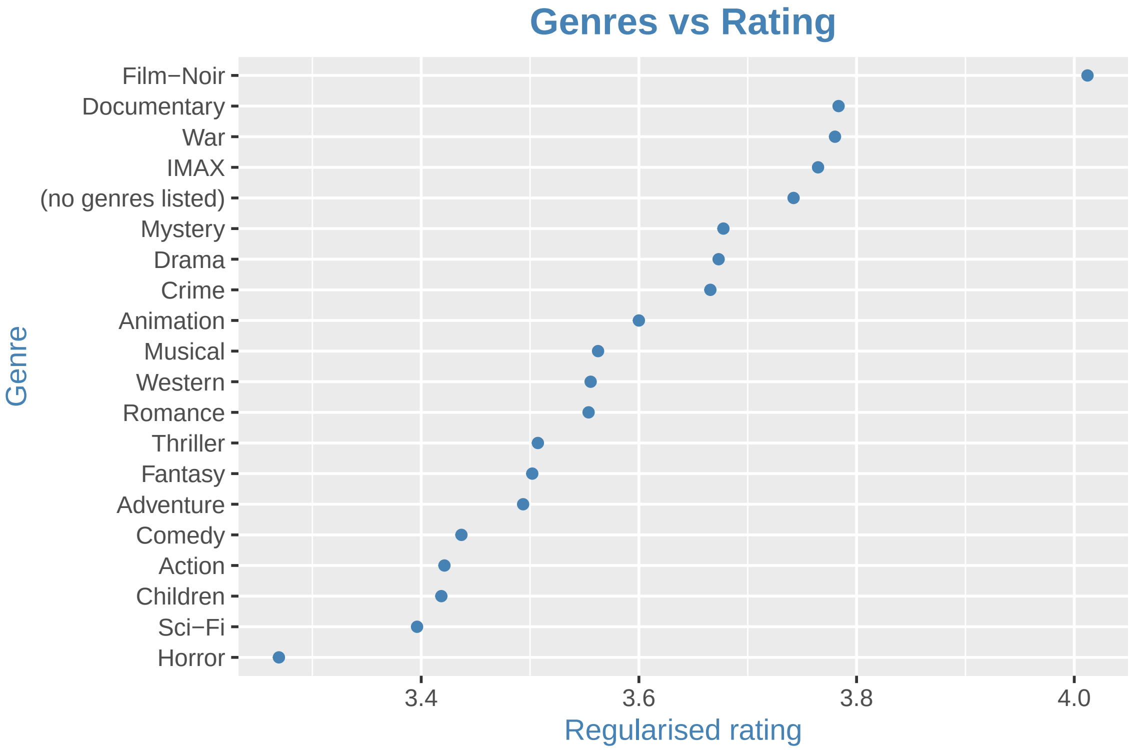
We can clearly see that a large majority (82.42 %) of the marks are above 2.5, and even 50.06 % of the marks are a 4 or over. I expected a more balanced distributions across the ratings. Nevertheless, it makes sense as people often watch movies pitched and recommended by a friend… or a good recommendation system!
We observe below that the genres of blockbusters movies like drama, comedy or action movies are often rated. On the other hand, film-noir, documentaries or IMAX movies are not often rated, probably due to their scarcity in the movie market.



At genre level, let's see how users mark the movies they have watched. I try to boxplot the distribution of rating by genre:



It is not possible to analyze such a monolithic graph! Then, I decided to use regularisation to balance number of rating effect. To achieve this, I use the values $\mu_v = 3.9$ and $\lambda_v = 4.4$. This values will be calculated in the machine learning analysis in the chapter *III-4 MODEL 4: Model 3 + Factorisation*



The genres film noir, documentary and IMAX are clearly the best marked genres but also the less often marked/watched. We may think these genres are probably a connoisseur' privilege. On the other side, the genre action get the highest rating frequency and one of the worst regularised rating. It is more the genre of blockbusters that drains many viewers who are sometimes disappointed. I could analyze the data set in many other axis, but let's go on with the development of a recommendation system.

III- Methods

The purpose of a machine learning algorithm is to train an algorithm using a train dataset for which we do know the outcome. Then, the algorithm computes some **predicted ratings** for a validation set for which we also know the outcome. Finally, how close the **predicted ratings** are from the **True ratings** of the validation set defined the quality of the model. The loss function used to measure the quality of the prediction algorithm is RMSE calculated as below:

$$RMSE \ = \ \sqrt{\frac{1}{N}\sum_{Validation\ Set} \left(Predicted\ ratings - True\ ratings\right)^2}$$

The first step I took is to use a naive prediction model in which I predicted all unknown ratings $r_{u,i}$ with the the overall average μ_0:

$$r_{u,i} \ = \ \frac{1}{N}\sum_{train\ set} True\ ratings \ = \ \mu_0$$

Obviously, any other algorithm should give a better result to worth being considered.
From this naive model, I went on with a baseline prediction for an unknown rating $r_{u,i}$. This baseline model accounts for the user and item effects, b_u and b_i which are the observed deviations of user u and item i respectively, from the average. In this configuration the baseline predicted rating is formulated as below:

$$r_{u,i} = \mu_0 + b_u + b_i$$

In this approach, the residual is noted $\epsilon_{u,i}$ when true ratings $Y_{u,i}$ are defined as below:

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

III-1 MODEL 1: Movies and users effects

In this model, I took into account only the user and the movie effects (b_u and b_i) to evaluate the predicted ratings $r_{u,i} = \mu_0 + b_u + b_i$, then calculated RMSE:

Model	Description	RMSE
MODEL 0	Naive	1.061202
MODEL 1	Movie & user effects	0.865349

Quite a satisfying inprovement of 18.46 % from the naive model!

The models -naive and basic studied till now try to predict a continuous variable since the models calculate as predicted ratings any decimal values. But, the true ratings can be seen as a categorical variable since the possible values are limited to every half from 0.5 to 5. Therefore, an idea is to round the predicting ratings to these limited values. At the same time, this will suppress the very few negative and over 5 values.

```
# Calculate "categorical" iu_effect rmse
predicted_ratings[predicted_ratings < 0] <- 0
predicted_ratings[predicted_ratings == 0] <- 0.5
predicted_ratings[predicted_ratings > 5] <-
predicted_ratings <- round(predicted_ratings/0.5)*0.5

cat_iu_effect_rmse <- RMSE(predicted_ratings, validation$rating)

# Calculate "categorical" naive rmse
predicted_ratings <- validation %>%
   mutate(pred = mu) %>%
   .$pred
predicted_ratings[predicted_ratings < 0] <- 0
predicted_ratings[predicted_ratings == 0] <- 0.5
predicted_ratings[predicted_ratings > 5] <- 5
predicted_ratings <- round(predicted_ratings/0.5)*0.5

cat_naive_rmse <- RMSE(predicted_ratings, validation$rating)
```

With this adjustment, it seems that rounding the outcomes does not improve the RMSE.

We will check this in chapter *IV- Results*

Model	Description	RMSE	RMSE_cat
MODEL 0	Naive	1.061202	1.06127
MODEL 1	Movie & user effects	0.865349	0.87770

III-2 MODEL 2a: Duo-regularisation

Then I evaluate the penalty parameters λ_i and λ_u that constrain the total variability of the effect sizes. First, for each item i we set:

$$b_i = \frac{\sum_{u \epsilon R(i)} (r_{u,i} - \mu)}{\lambda_i + |R(i)|}$$

That is coded as below:

```
b_i <- train_edx %>%
  group_by(movieId) %>%
  summarize(b_i = sum(rating - mu) / (n() + l1))
```

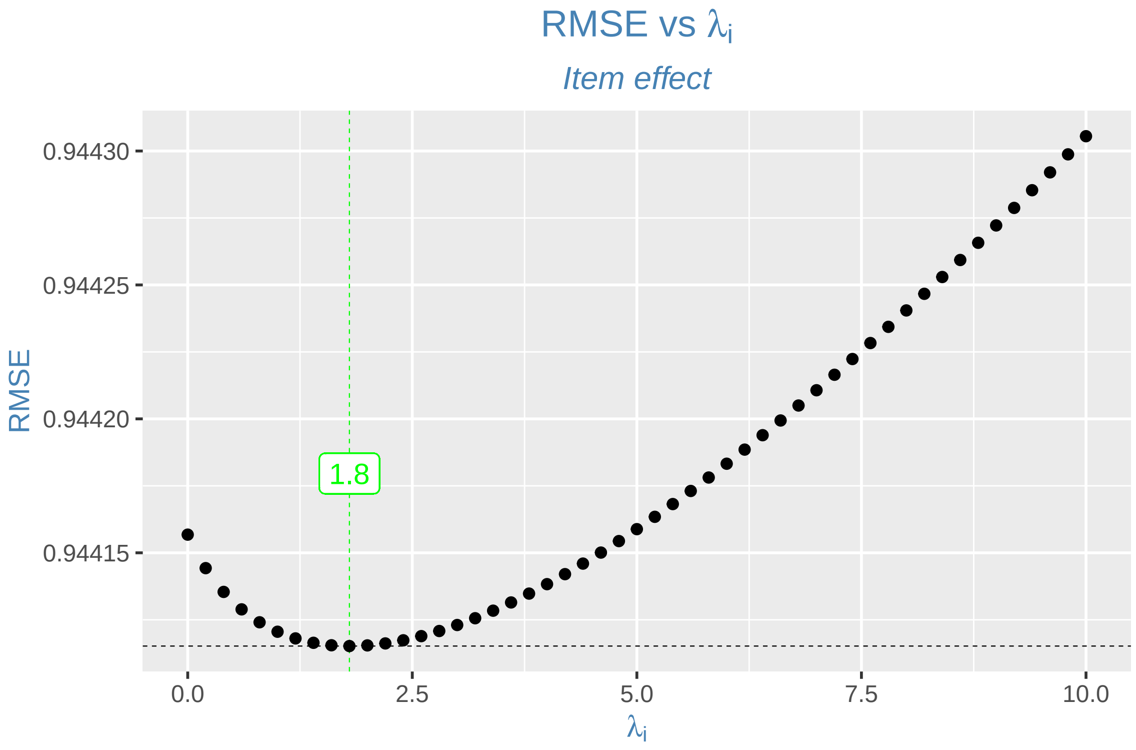
Then, for each user u we set:

$$b_u = \frac{\sum_{u \epsilon R(u)} (r_{u,i} - \mu - b_i)}{\lambda_u + |R(u)|}$$

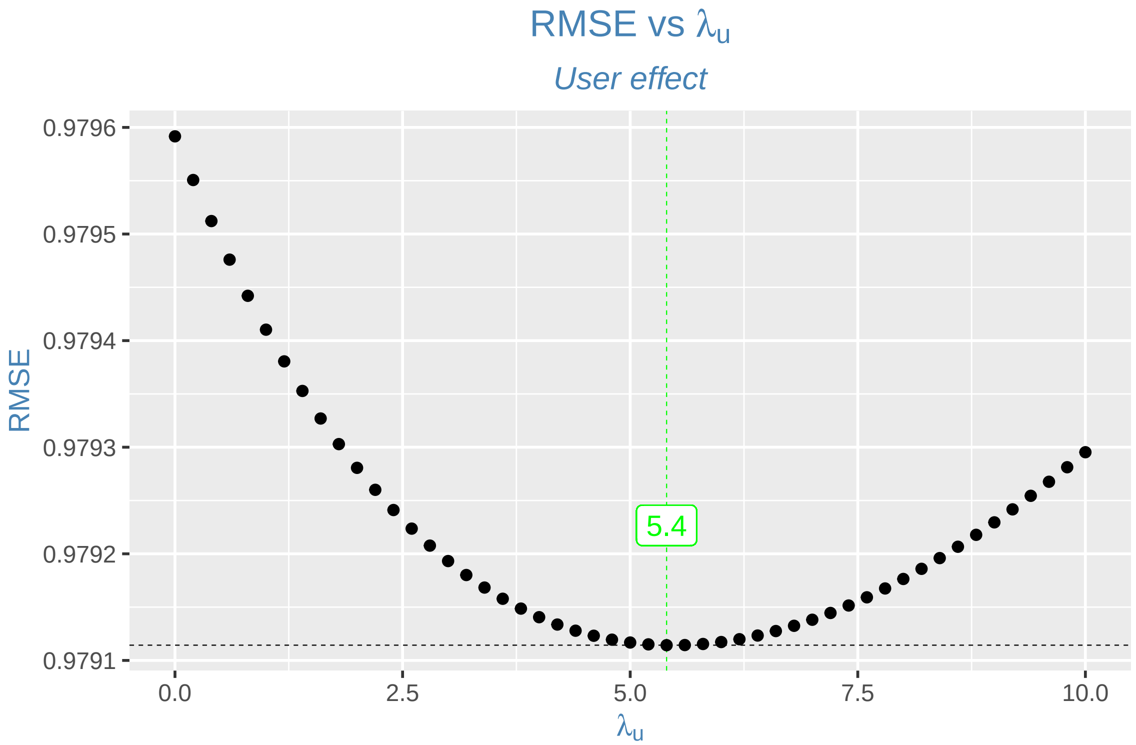
That is coded as below:

```
b_u <- train_edx %>%
  group_by(userId) %>%
  summarize(b_u = sum(rating - mu)/ (n() + l2))
```

And the results are shown below:



The computation gives the penalty due to the item effect $\lambda_i = 1.8$



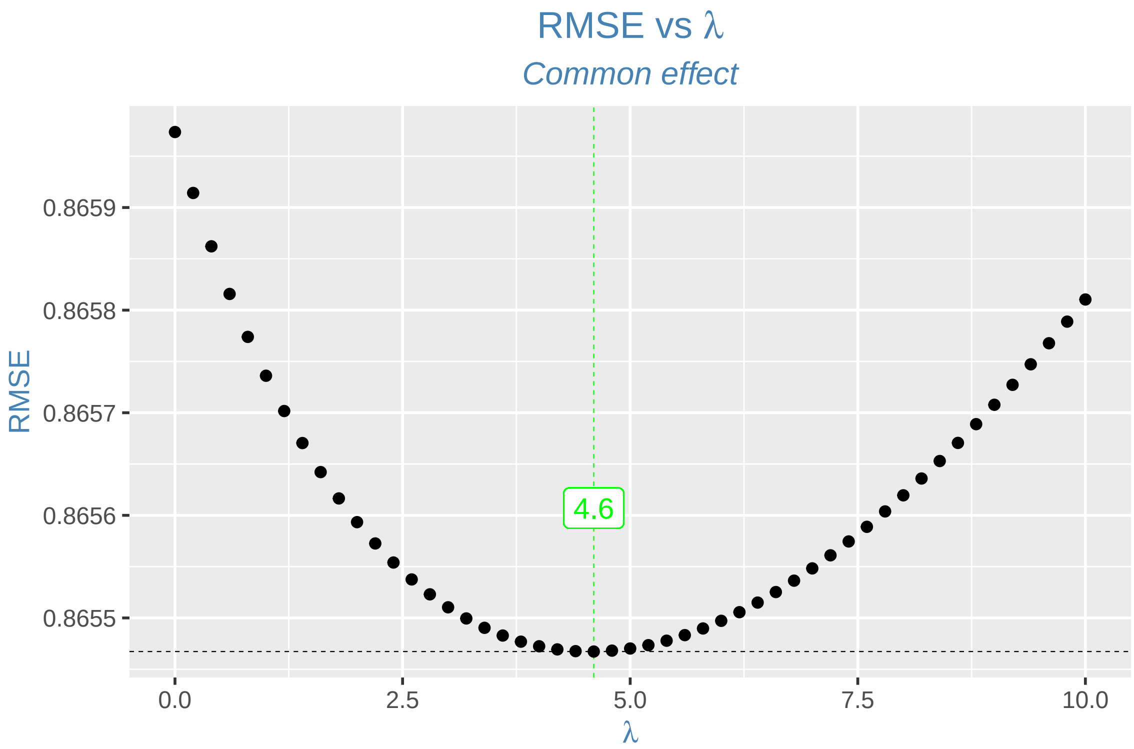
The calculation gives the penalty attached to the user effect: $\lambda_u = 5.4$. The loss function RMSE improves.

Model	Description	RMSE
MODEL 0	Naive	1.061202
MODEL 1	Movie & user effects	0.865349
MODEL 2a	MODEL 1 + Duo-regularisation	0.864861

This first adjustment that deals with two penalty parameters offers an improvement of 0.056 %.

III-3 MODEL 2b: Common regularisation

I wanted to try a single penalty parameter that would apply both to item and user. This model that I labeled "MODEL 2b + Uni-regularisation" outputs the unique λ equal to 4.6



Model	Description	RMSE
MODEL 0	Naive	1.061202
MODEL 1	Movie & user effects	0.865349
MODEL 2a	MODEL 1 + Duo-regularisation	0.864861
MODEL 2b	MODEL 1 + Uni-regularisation	0.864822

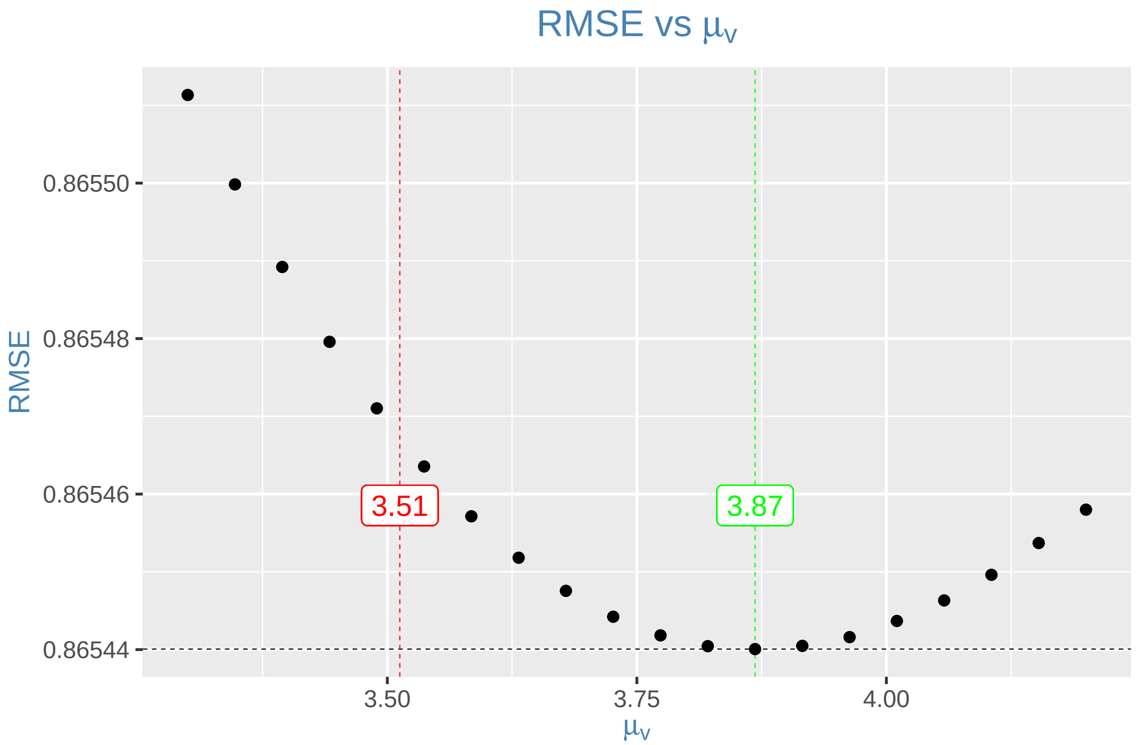
But did not improve RMSE that much… Just 0.005 %!

There is an assumption in the previous models. I use μ_0 (~ 3.51) the average of all ratings of all movies based on the only rated movies. The assumption was to assign the same mark μ_0 to any movie that is not rated by a user. In next model, I try to determine if another average μ_v could give a best result.

III-4 MODEL 3: Adjusted average

We make the average vary around $\mu_0 = 3.51$ and adjust the penalty parameter λ for each iteration:

```
#-------------------------------------------------------------------
#--- MODEL 3 : Model 2b + Average adjusted
#-------------------------------------------------------------------

var_mu <- seq(3.3, 4.2, length.out=20)
params <- lapply(var_mu, function(avg){

  lambdas <- seq(2, 7, 0.2)
  rmses_l <- sapply(lambdas, function(l){

    b_i <- train_edx %>%
      group_by(movieId) %>%
      summarize(b_i = sum(rating - avg) / (n() + l))

    b_u <- train_edx %>%
      left_join(b_i, by="movieId") %>%
      group_by(userId) %>%
      summarize(b_u = sum(rating - b_i - avg)/(n() + l))

    predicted_ratings <-
      test_edx %>%
      left_join(b_i, by = "movieId") %>%
      left_join(b_u, by = "userId") %>%
      mutate(pred = avg + b_i + b_u) %>%
      .$pred

    RMSE(predicted_ratings, test_edx$rating)

  })
   list(penalty = lambdas[which.min(rmses_l)],
        rmse = min(rmses_l))
})
```



Model	Description	RMSE
MODEL 0	Naive	1.061202
MODEL 1	Movie & user effects	0.865349
MODEL 2a	MODEL 1 + Duo-regularisation	0.864861
MODEL 2b	MODEL 1 + Uni-regularisation	0.864822
MODEL 3	MODEL 2b + Adjusted average	0.864790

My satisfaction upholds as this algorithm still improves RMSE by 0.004 %, when μ_v is set to 3.87 and λ adjusted to $\lambda_v = 4.4$. We will use the value $\mu_v = 3.87$ of the average in next models.

III-5 MODEL 4: Model 3 + Factorisation

The previous model outputs the rating residuals $\epsilon_{u,i}$:

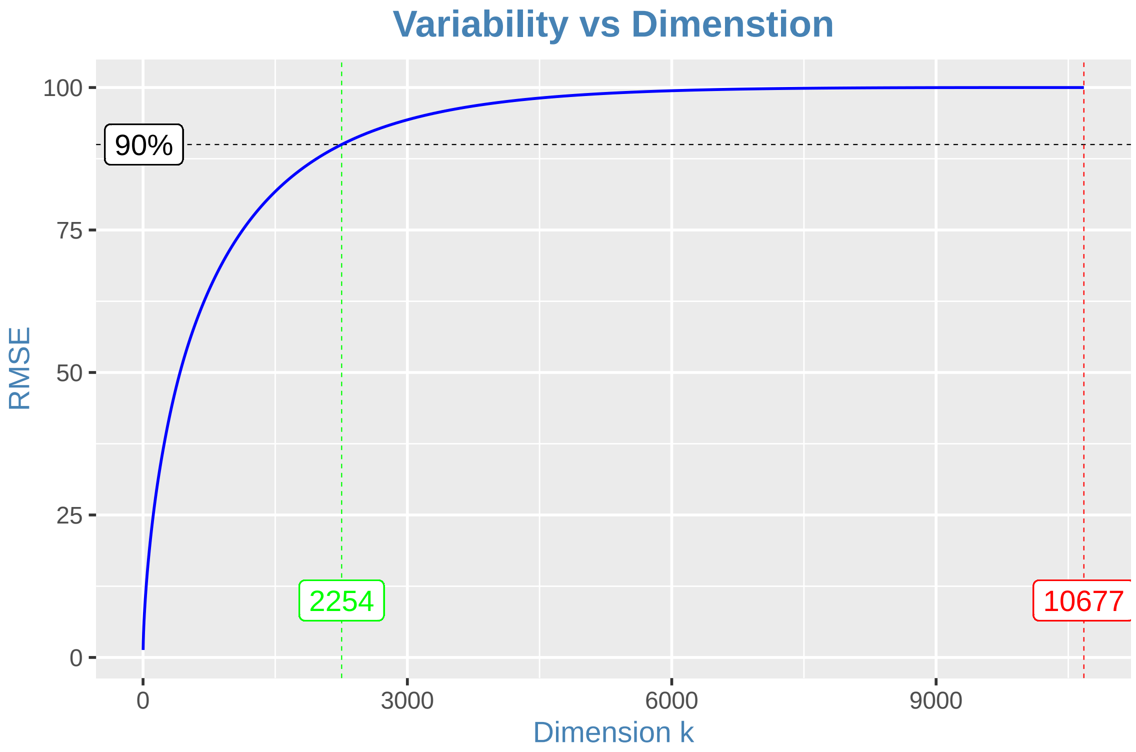
$$\epsilon_{u,i} = Y_{u,i} - \mu_v - b_i - b_u$$

I used the Singular Value Decomposition algorithm to reduce the sparse matrix to the rank k that explain 90% of variability of the residuals. In this configuration residuals is evaluated as:

$$\epsilon_{u,i} = \sum_{n=1}^{k} p_{u,n} q_{n,i}$$

```r
#-----------------------------------------------------------------
#--- MODEL 4: Model 3 + Factorisation
#-----------------------------------------------------------------
# Set average value
mu <- best_mu

# Transform residuals to a matrix...
rating_residuals <- edx %>%
  left_join(b_i, by='movieId') %>%
  left_join(b_u, by='userId') %>%
  mutate(residu= rating - mu - b_i - b_u) %>%
  select(userId, movieId, residu) %>%
  spread(movieId, residu)  %>%
  as.matrix()
rating_residuals[is.na(rating_residuals)] <- 0
rownames(rating_residuals) <- rating_residuals[,1]
rating_residuals <- rating_residuals[,-1]

# ... to reduce using SVD
svd <- svd(rating_residuals)
d <- as.matrix(svd$d)
expl_rate <- cumsum(d^2)/sum(d^2) * 100
max_k <- dim(d)[1]

# Get k
k <- min(which(expl_rate> 90))  #k explains 90% Of the variability
# Get reduction rate
reduc <- ((max_k-k)/max_k) *100
```

A 79



Improvements in the loss function RMSE is shown:

Model	Description	RMSE
MODEL 0	Naive	1.061202
MODEL 1	Movie & user effects	0.865349
MODEL 2a	MODEL 1 + Duo-regularisation	0.864861
MODEL 2b	MODEL 1 + Uni-regularisation	0.864822
MODEL 3	MODEL 2b + Adjusted average	0.864790
MODEL 4	MODEL 3 + Residuals factorisation (SVD)	0.862139

IV- Results

The final model where the Single Value Decomposition evaluates the residuals, offers an interesting 18.76 % improvement from the naive model. Even from the basic model -Movie and user effects the improvement is still of 0.37 % to reach a RMSE of 0.862.

Model	Description	RMSE
MODEL 0	Naive	1.061202
MODEL 1	Movie & user effects	0.865349
MODEL 2a	MODEL 1 + Duo-regularisation	0.864861
MODEL 2b	MODEL 1 + Uni-regularisation	0.864822
MODEL 3	MODEL 2b + Adjusted average	0.864790
MODEL 4	MODEL 3 + Residuals factorisation (SVD)	0.862139

The rounding of the predicting ratings to the closest 0.5 never improved the loss function:

Model	Description	RMSE	RMSE_cat	Improvement
MODEL 0	Naive	1.061202	1.061270	-0.00642
MODEL 1	Movie & user effects	0.865349	0.877700	-1.42725
MODEL 2a	MODEL 1 + Duo-regularisation	0.864861	0.876718	-1.37088
MODEL 2b	MODEL 1 + Uni-regularisation	0.864822	0.876687	-1.37188
MODEL 3	MODEL 2b + Adjusted average	0.864790	0.876670	-1.37374
MODEL 4	MODEL 3 + Residuals factorisation (SVD)	0.862139	0.874066	-1.38337

V- Conclusion

A shallow data analysis shows that the less rated genres are often high rated. Probably because these genres of movies like film noir, IMAX, and documentary are watched by a few connoisseurs. On the other hand, action movies that drains many viewers are more likely disappointing.

The first models based on the correction of the item and the user effects generates the RMSE equals to 0.865. The Single Value Decomposition algorithm applied to the rating residuals lets predict users' ratings to movies with a satisfying RMSE equal to 0.862. Inpsired by Bell and Koren in their paper describing their final solution to the Netflix Prize:

While the literature mostly concentrates on the more sophisticated algorithmic aspects, we have learned that an accurate treatment of main effects is probably at least as significant as coming up with modeling breakthroughs.

Some further explorations should be evaluated:

- hilight and normalise of the genre effect
- hilight and normalise of the temporal effect for example.

I tried to use the package *recommenderlab* with no real success. That should be tempted again. It would be also interesting to test ensemble models.

Finally, I want to thank the staff for their very prompt replies to my questions but also HarvardX and the course instructor Rafael Irizarry for the quality of the course.