MovieLens Project

by Erem Ugras

15 February 2021

```
load("workspace.RData")
```

```
# SECTION 1 - INTRODUCTION


# The purpose of this project is to develop a recommendation system which is similar to
# the system that was awarded the grand prize by Netflix in a contest in 2009.
# The recommendation system of interest basically predicts movies that
# a user might like based on the ratings the user has provided on different movies.
# The contest, called The Netflix Prize, was an open competition
# with a grand prize of $1,000,000.
# The winner recommendation system was based on loss function
# which produced the lowest Residual Mean Squared Error (RMSE)
# and more than 10% improvement of the RMSE that the Netflix' algorithm, Cinematch,
# generated at that time.
# The RMSE that won the grand prize was 0.8567 as opposed to Cinematch's RMSE 0.9525.
# It has been claimed that a 1% improvement of the RMSE can make a big positive difference
# to identify the "top-10" most recommended movies for a user.
# Reference: (YehudaKoren (2007-12-18). "How useful is a lower RMSE?". Netflix Prize Forum.
# Reference cont'd: Archived from the original on 2012-03-03.)
# The contestants were provided a dataset of 100M users with movie ratings for the contest.
# In this project, a smaller size dataset of 10M users will be used.
# I will explain the steps taken throughout my script to obtain a desired RMSE.


# Table of Content:
# Section 1 - Introduction
# Section 2 - Analysis
#   Section 2.1 - Data Wrangling
#   Section 2.2 - Splitting Edx to Train and Test Sets
#   Section 2.3 - Loss Function
#   Section 2.4 - Modeling Different Recommendation Systems and Computing RMSEs Accordingly
#   Section 2.5 - Penalized Least Squares (Regularization)
#   Section 2.6 - Choosing the Penalty Terms
#   Section 2.7 - Computing the Final RMSE by Using the Validation Set (Final Hold-Out Test)
# Section 3 - Results
# Section 4 - Conclusion
```

```
# SECTION 2 - ANALYSIS
```

```
# SECTION 2.1 - DATA WRANGLING

# My script starts with the chunk of code to download and tidy the MovieLens 10M dataset
# from grouplens.org.
# It is the chunk of code that is provided in the course to obtain the edx set and
# validation set (the final hold-out test set) out of the MovieLens 10M Dataset.
# The validation set will only be used to evaluate the RMSE of my final algorithm.
# The edx set will be split into separete training and test sets in the next section
# to design and test my algorithm.
```

```
# SECTION 2.2 - SPLITTING EDX TO TRAINING AND TEST SETS

# edx and validation datasets were created in the previous section.
# In this section, edx dataset will be split into "train_set_edx" and "test_set_edx"
# which will be used to train and test my algorithm.

# 80% of the edx dataset will constitute the train_set_edx and
# the rest will constitute the test_set_edx
```

```
# SECTION 2.3 - LOSS FUNCTION

# The typical error loss is selected as the preferred recommendation system
# in this project which was also the method of the winner algorithm
# in the Netflix Prize in 2009.
```

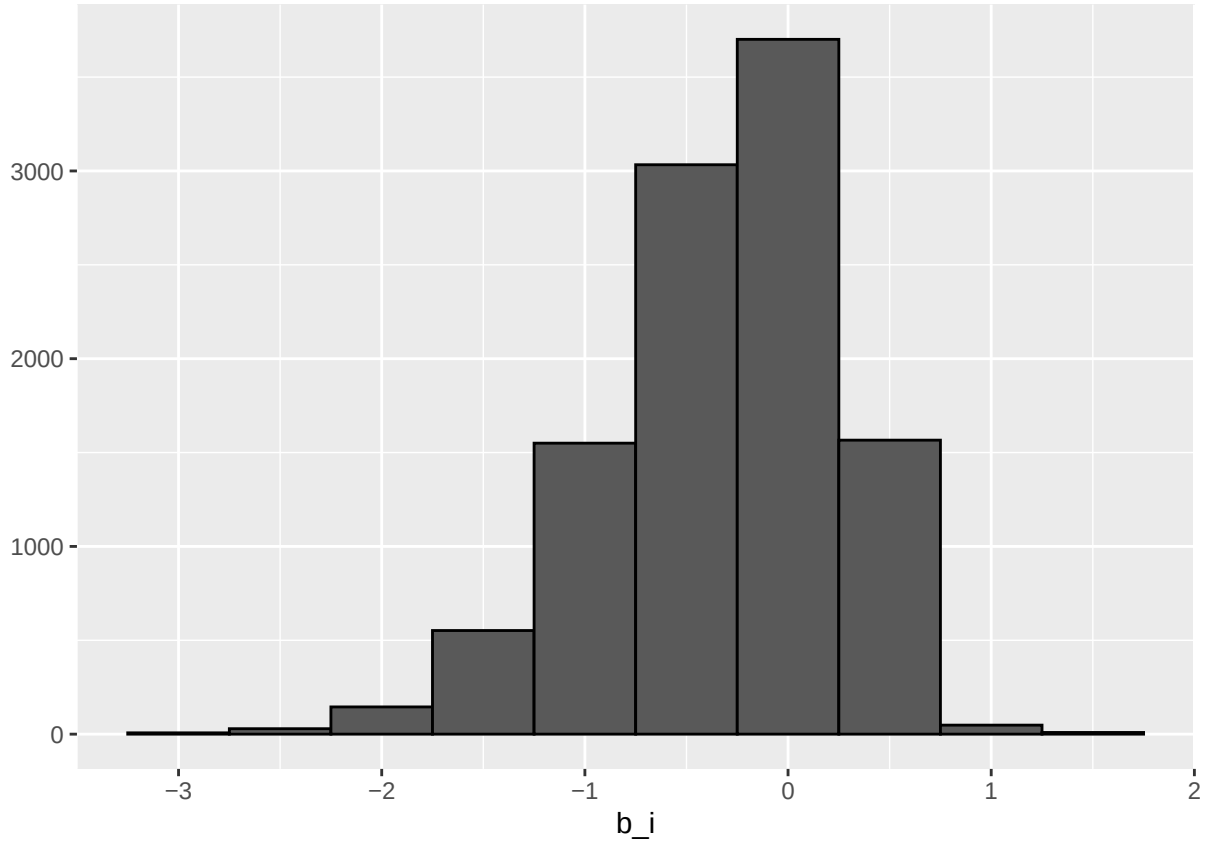
```
# SECTION 2.4 - MODELING DIFFERENT RECOMMENDATION SYSTEMS AND
# COMPUTING RMSEs ACCORDINGLY

# To start off, the simplest possible recommendation system is built which
# predicts the same rating for all movies regardless of user.
# The value for the "same rating" is chosen to be the average of all rating
# which is represented by mu here:
mu
```

```
## [1] 3.512462
```

```
# RMSE based on the average rating of all ratings is calculated:
just_the_average_result
```

```
## [1] 1.060368
```

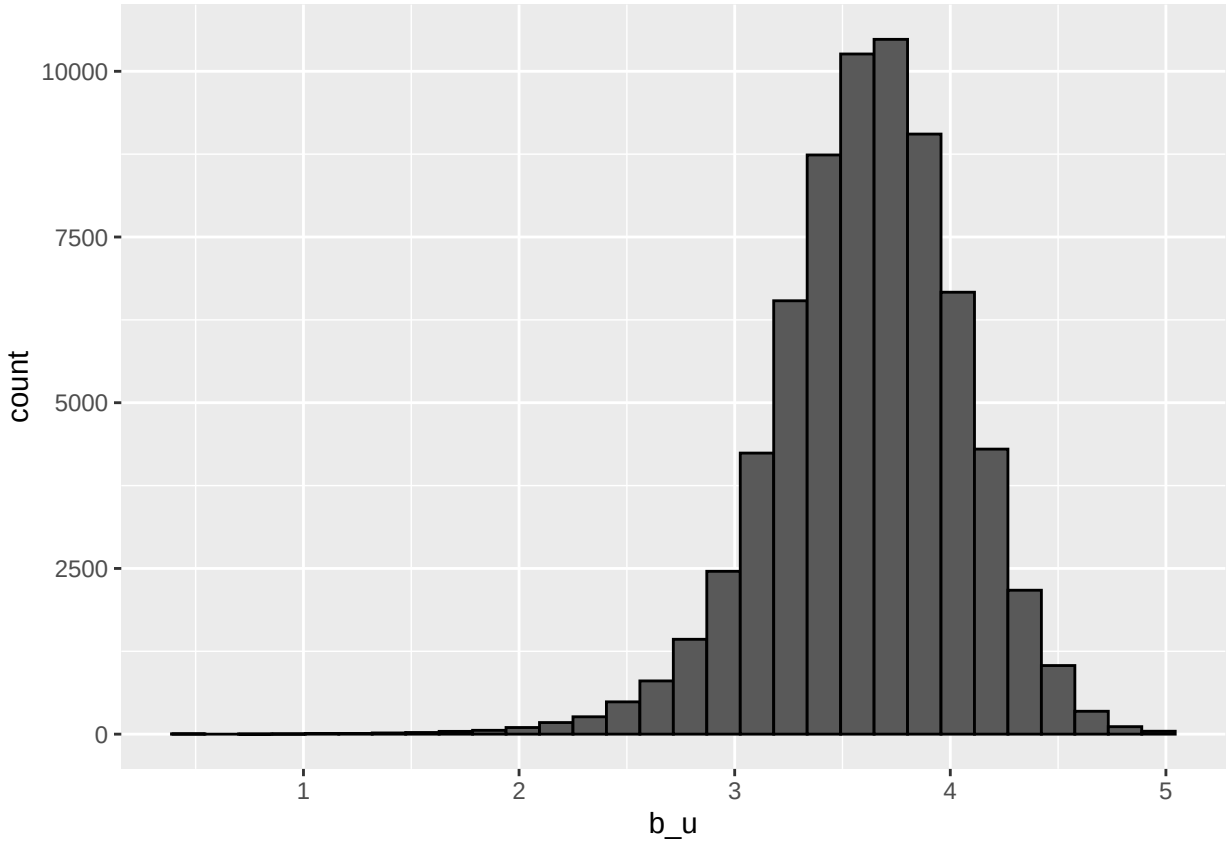
```
# It is confirmed by data that some movies are rated higher than others.
# This bias is called the movie-specific effect.
# As shown on this plot, the effect of movie bias is noteworthy:
plot_effect_of_movie
```




```
# When this bias is added to the simplest recommendation system
# that was introduced previously, a small improvement is observed in RMSE:
movie_effect_model_result
```

```
## [1] 0.9431022
```

```
# Some users are likely to give higher ratings to movies in general and some are not.
# This effect is called user-specific effect.
# This plot shows the user-specific effect for the users who have rated over 100 movies:
plot_effect_of_user
```

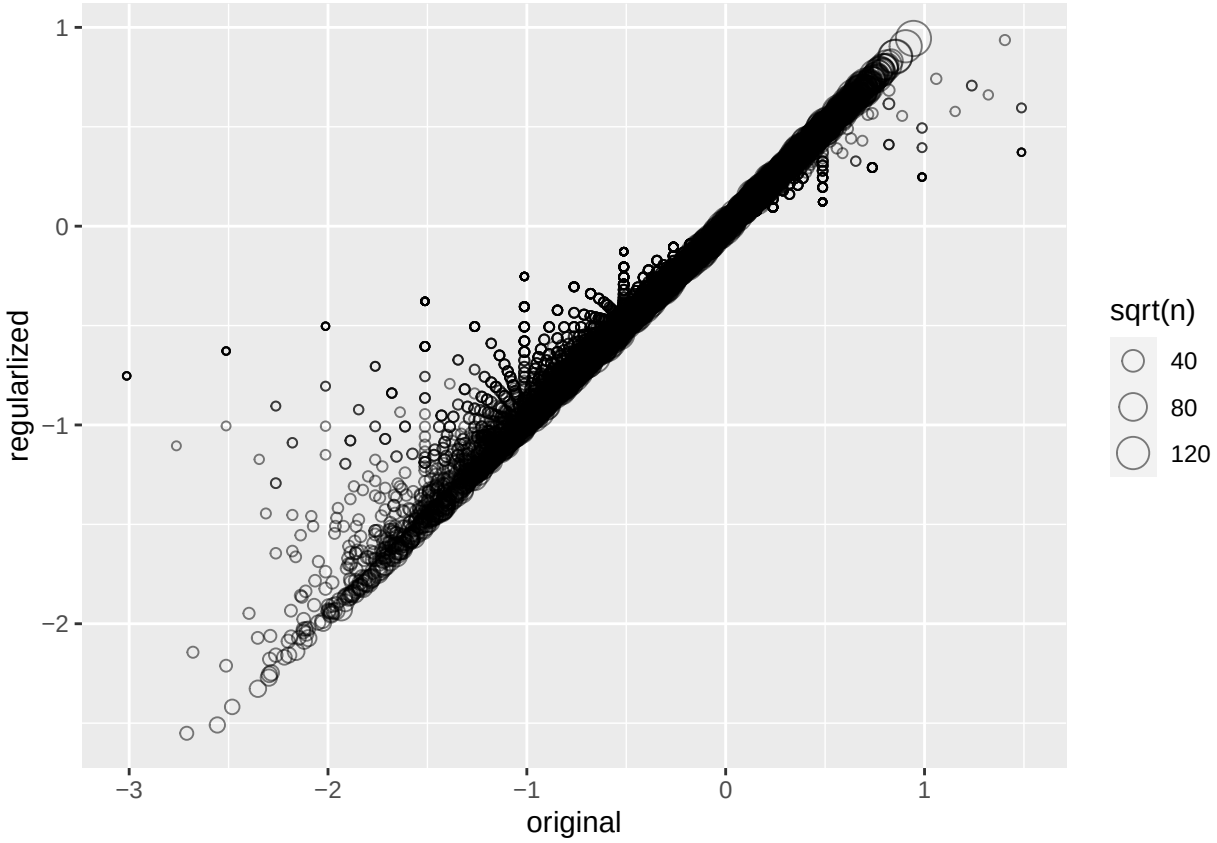



```
# The user-specific effect is added to the the previous model
# with the movie-specific effect.
# When an overcritical user (negative b_u) rates a good movie (positive b_i),
# both effects counter each other and a better prediction could be obtained:
movie_and_user_effects_model_result
```

```
## [1] 0.8654546
```

```
# SECTION 2.5 - PENALIZED LEAST SQUARES (REGULARIZATION)

# It is not surprising that some high ranked or low ranked movies were rated
# by very few users which causes larger estimates of b_i.
# Large estimates of b_i can increase the RMSE.
# Therefore, a technique which will penalize large estimates is needed.
# Penalized Least Squares (Regularization) is the technique that
# will be used to penalize the large estimates in my analysis.
# Another term, lambda, is introduced as the driver of penalty.
# When the number of rating for a movie is high, a case which
# will give us a stable estimate, lambda is effectively ignored.
# On the other hand, when the number of rating for a movie is small,
# the regularized least squared estimate of b_i is shrunken towards 0.
# The larger lambda, the more the regularized estimate of b_i shrinks.
# A range of values for lambda will be used to obtain the best RMSE.

# To see how the estimates shrink, let's make a plot of the regularized
```

```
# estimates versus the least squares estimates.
plot_estimates_shrink
```




```
# Now, let's look at the top 10 best movies based on
# the penalized least squared estimates of b_i:
top_10_movies
```

title	b_i	n
Shawshank Redemption, The (1994)	0.9442327	22430
More (1998)	0.9361363	6
Godfather, The (1972)	0.9049127	14169
Usual Suspects, The (1995)	0.8544729	17274
Schindler's List (1993)	0.8510517	18625
Sunset Blvd. (a.k.a. Sunset Boulevard) (1950)	0.8313438	2340
Double Indemnity (1944)	0.8132371	1695
Casablanca (1942)	0.8105483	9051
Seven Samurai (Shichinin no samurai) (1954)	0.8012368	4146
Rear Window (1954)	0.7955437	6372

```
# These movies show that penalized estimates method provides
# a stable estimates of movies with high number of ratings.

# RMSE that is computed based on penalized estimates which only
```

```
# covers the movie effect in this case:
regularized_movie_effect_model
```

```
## [1] 0.9430591
```

```
# SECTION 2.6 - CHOOSING THE PENALTY TERMS

# Lambda is a tuning parameter so we can use cross-validation to choose it.

# This time RMSE is computed based on penalized estimates with movie
# and user effects together:
regularized_movie_and_user_effect_model
```

```
## [1] 0.864777
```

```
# RMSE based on penalized estimates with movie and user effects together
# is the lowest that have been obtained so far.
```

```
# SECTION 2.7 - COMPUTING THE FINAL RMSE BY USING THE VALIDATION SET
# (FINAL HOLD-OUT SET)

# In this section, the RMSE is computed by using the validation set
# to test the final algorithm which is based on penalized least
# squares estimates with movie and user effects.
final_rmse_validation_set_result
```

```
## [1] 0.8649857
```

```
# SECTION 3 - RESULTS


# The results that were obtained in each section are presented and
# discussed in this section. Firstly, let us tabulate the results in the table below:

#  Method                                               RMSE
# Just the Average (Section 2.4)                      1.060368
# Movie Effect Model (Section 2.4)                    0.943102
# Movie + User Effects Model (Section 2.4)            0.865454
# Regularized Movie Effect Model (Section 2.5)        0.943059
# Regularized Movie + User Effect Model (Section 2.6) 0.864777
# Final RMSE Validation Set (Section 2.7)             0.864985

# As seen the results in the table above, the lowest RMSE was obtained
# in Regularized Movie + User Effect Model in Section 2.6
# compared to the results obtained in other models.
# Therefore, that model was selected to compute the Final RMSE with the Validation Set
# in Section 2.7.
```

```
# SECTION 4 - CONCLUSION

# The final RMSE that was computed with the Validation Set in Section 2.7
```

```
# is way lower than the Cinematch's RMSE 0.9525.
# However, it is still greater than the RMSE 0.8567 that
# won the Netflix Prize in 2009 as explained in Section 1.
# Obviously, there is still work to be performed to improve the final RMSE.
# The final model that was used in this analysis does not account the fact that
# groups of movies and groups of users have similar rating patterns.
# For example, users who have liked romantic movies tend to like
# other romantic movies more then expected. This effect needs to be added to the model.
# Singular value decomposition (SVD) and principal component analysis (PCA) are
# proper approaches to follow for further RMSE improvement.
# SVD and PCA can be added to the model by using the Recommenderlab package.
```