MovieLens Project

2022-07-06

EXECUTIVE SUMMARY

The goal is to create a movie recommendation system using the MovieLens Dataset: https://grouplens.org/datasets/movielens/10m/.
the train and validation sets are provided within the course in www.edx.org.

We develop our algorithm using the edx set.
For a final test of our final algorithm, we predict movie ratings in the validation set (the final hold-out test set) as if they were unknown.
RMSE will be used to evaluate how close our predictions are to the true values in the validation set (the final hold-out test set).

Few basic observations:

Table 1: First 5 instances of edx 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

The output we want to predict/estimate is the *rating* variable.

The edx set and the validation set have the following dimensions:

```
## [1] "Instances of the edx set: 9000055  "
```

```
## [1] "Instances of the validation set: 999999  "
```

```
## [1] "Variables of the edx and validation set: 6  "
```

Table 2: Original Variable names

Data_variables
userId

Data_variables
movieId
rating
timestamp
title
genres

Key steps to create the movie recommendation system:

First step: Data cleaning, exploration and visualization
Second step: Building the model with the edx set using the user, movie, time and genres effects
Third step: Tuning regularization with a partition of the edx set
Fourth step: Applying the models and regularization to the validation set and checking the RMSE results

METHOD AND ANALYSIS

PRELIMINARY

The initial script comes from EDX course, and it creates edx and validation sets.

In order to avoid modifying the original data, the files are renamed:
edx –> train_set
validation –> test_set

DATA CLEANING, EXPLORATION AND VISUALIZATION

We identify 3 main *predictors*: **MovieId**, **userId** and **genres**.
We will see a bit later how we can transform the **timestamp** as a *predictor*.

See below the number of the main 3 unique *predictors* in the data set :

Table 3: Number of unique predictors

variable	unique_numbers
movies	10677
users	69878
genres	797

The unique number of each predictor is high, so training a model may freeze and fail Rstudio.

Below, for information, we have calculated the 5 most rated movies of the data set :

Table 4: 5 most rated movies

movieId	title	count
296	Pulp Fiction (1994)	31362
356	Forrest Gump (1994)	31079
593	Silence of the Lambs, The (1991)	30382
480	Jurassic Park (1993)	29360
318	Shawshank Redemption, The (1994)	28015

. . . and the 5 least rated movies of the data set :

Table 5: 5 least rated movies

movieId	title	count
64976	Hexed (1993)	1
65006	Impulse (2008)	1
65011	Zona Zamfirova (2002)	1
65025	Double Dynamite (1951)	1

movieId	title	count
65027	Death Kiss, The (1933)	1

The data base include a **timestamp**, which corresponds to the date of rating.
The **timestamp** can be converted into rating date's year, with a new column called **ratingDate** :

Table 6: Creation of ratingDate column

timestamp	title	ratingDate
838985046	Boomerang (1992)	1996
838983525	Net, The (1995)	1996
838983421	Outbreak (1995)	1996
838983392	Stargate (1994)	1996
838983392	Star Trek: Generations (1994)	1996

In the title of the movie, we can see the year of release of the movie.
We can extract this year, and create a new column called **movieDate**.

Table 7: Creation of movieDate column

timestamp	title	ratingDate	movieDate
838985046	Boomerang (1992)	1996	1992
838983525	Net, The (1995)	1996	1995
838983421	Outbreak (1995)	1996	1995
838983392	Stargate (1994)	1996	1994
838983392	Star Trek: Generations (1994)	1996	1994

It can be interesting to have a *predictor* based on time difference between the year of the rating and the year of release of the movie.
We can estimate that if a movie was released long time ago compare to the date of watching (and rating), it might have less impact on the user, compare to a fresh new movie.
We call this new column **ratingPeriod**, and we will check the effect later.

In very few cases compare to the total number of instances, the **ratingPeriod** is less than 0.

Table 8: Negative ratingPeriod

ratingPeriod	n
-2	3
-1	172
0	380915

In order to avoid this situation, we modify the **ratingPeriod** to 0 when it appears negative initially. it makes more sense.

After this modification, see below the number of movies with **ratingPeriod** less or equal to 5:

Table 9: 5 lowest ratingPeriod

ratingPeriod	n
0	381090
1	1068070
2	853680
3	647650
4	473660

See below the number of movies with the 5 highest **ratingPeriod**:

Table 10: 5 highest ratingPeriod

ratingPeriod	n
89	108
90	60
91	41
92	28
93	14

The train_set and test_set are entirely modified to add the **ratingPeriod** column.

METHOD

Due to the high unique number of movies and users, training the data set is not feasible.
Then we will build the model, according to the following formula:

$$Y_{u,i,t,g} = \mu + b_i(\lambda) + b_u(\lambda) + b_t(\lambda) + b_g(\lambda) + \varepsilon_{u,i,t,g}(\lambda)$$

Where:
$Y_{u,i,t,g}$ represents the estimated value.
μ represents the mean
b_i represents the movie effect
b_u represents the user effect
b_t represents the time effect
b_g represents the gender effect
λ represents the regularization tuning parameter

We will tune regularization parameter λ by creating a partition of the train_set.

Once the model with regularization is built, we calculate the RMSE of the validation set (test_set).

MODELING

THE NAIVE MODEL

We build a naive model where all movies are rated the average μ.

$$Y = \mu + \varepsilon$$

We calculate the rating average of the train_set.

```
## [1] "The average of the ratings in the train_set is: 3.51246520160155  "
```

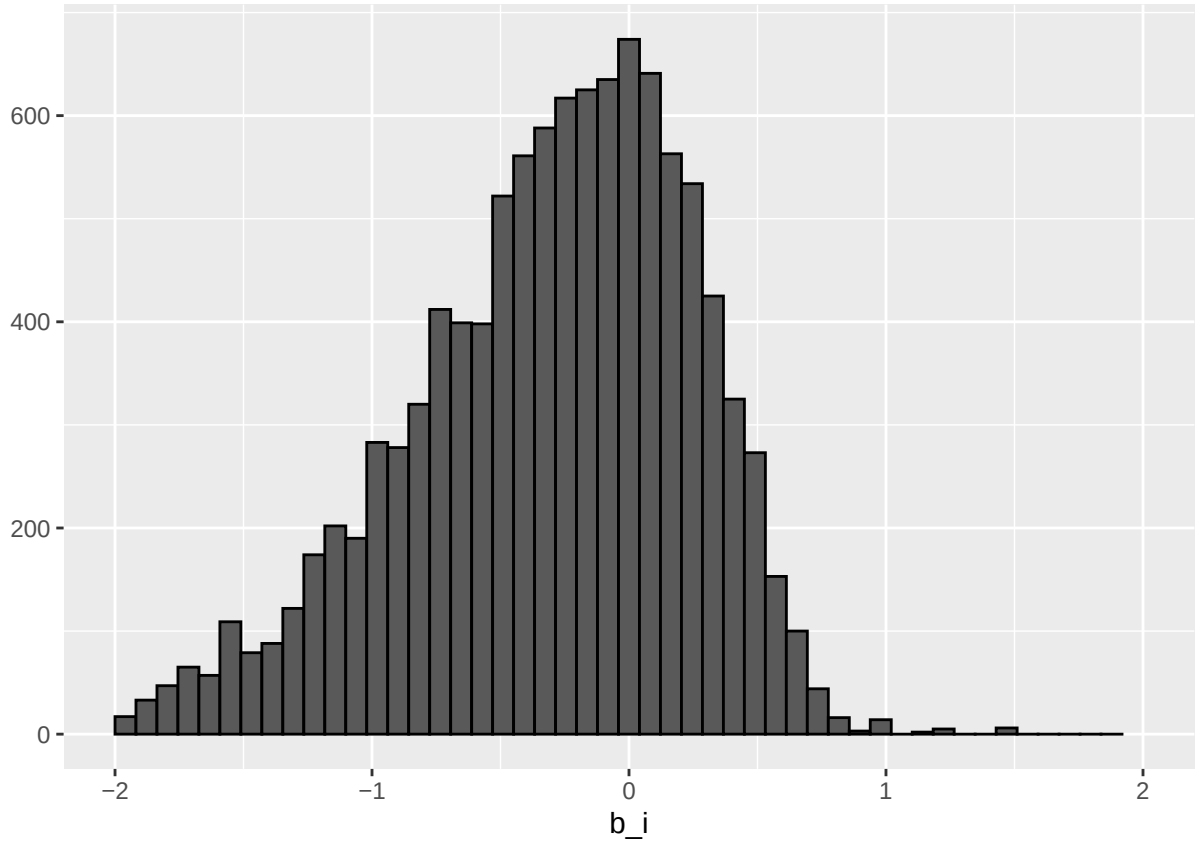
The next step is to include the movie effect.

THE FIRST MODEL

The 1st model calculate the movie effect b_i from the training set.

$$Y_i = \mu + b_i + \varepsilon_i$$

We can visualize the distribution of the movie effect on the drawing below:



The effect is important.

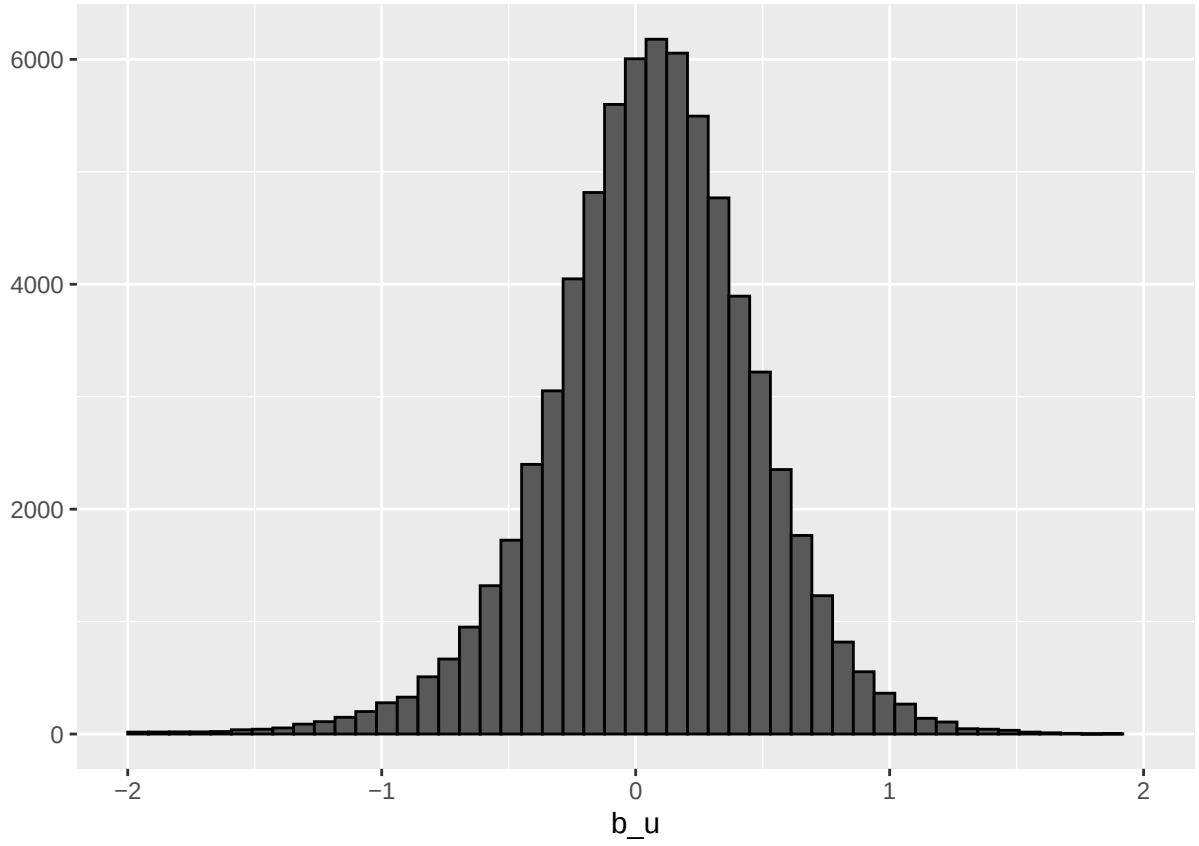
The next step is to include the user effect.

THE SECOND MODEL

The 2nd model calculate the user effect b_u from the training set.

$$Y_{u,i} = \mu + b_i + b_u + \varepsilon_{u,i}$$

We can visualize the distribution of the user effect on the drawing below:



The effect is important.

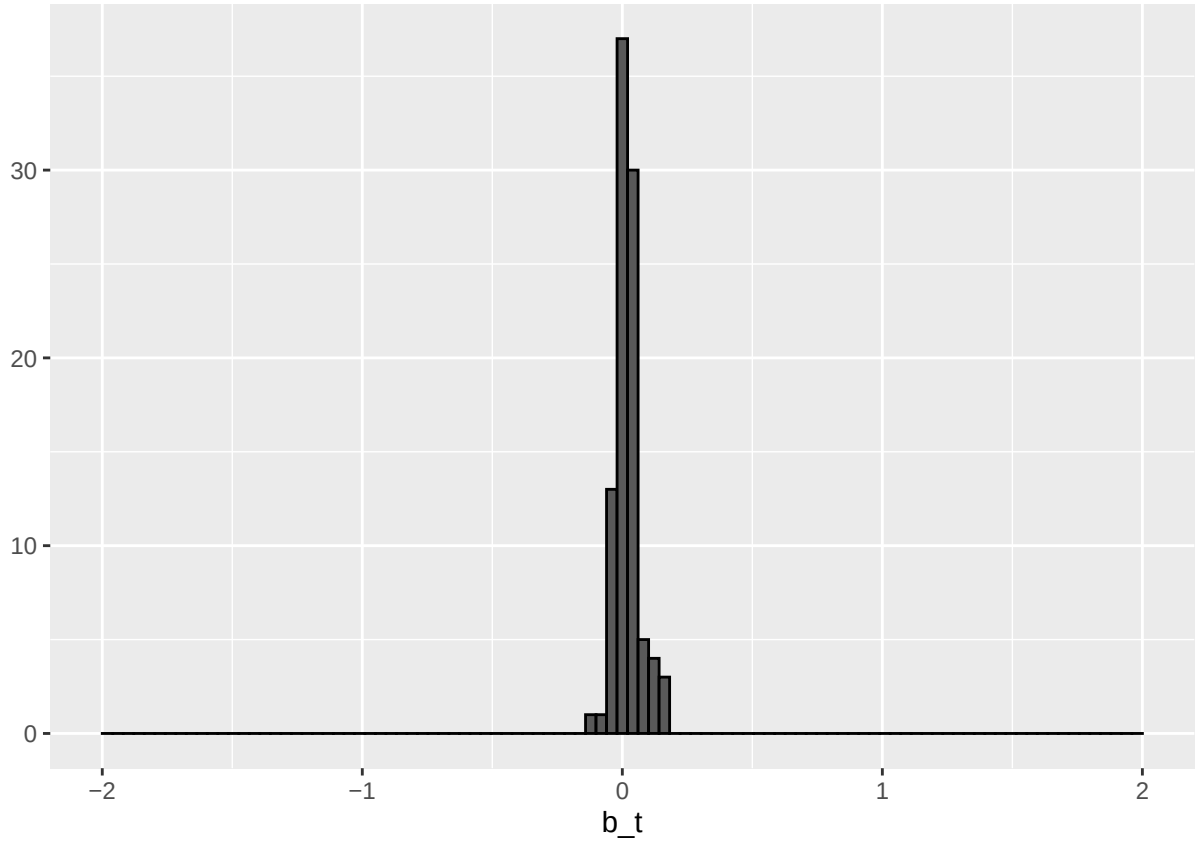
The next step is to include the time effect.

THE THIRD MODEL

The 3nd model calculate the time effect b_t from the training set.

$$Y_{u,i,t} = \mu + b_i + b_u + b_t + \varepsilon_{u,i,t}$$

We can visualize the distribution of the time effect on the drawing below:



The effect is not very important, but not negligible.

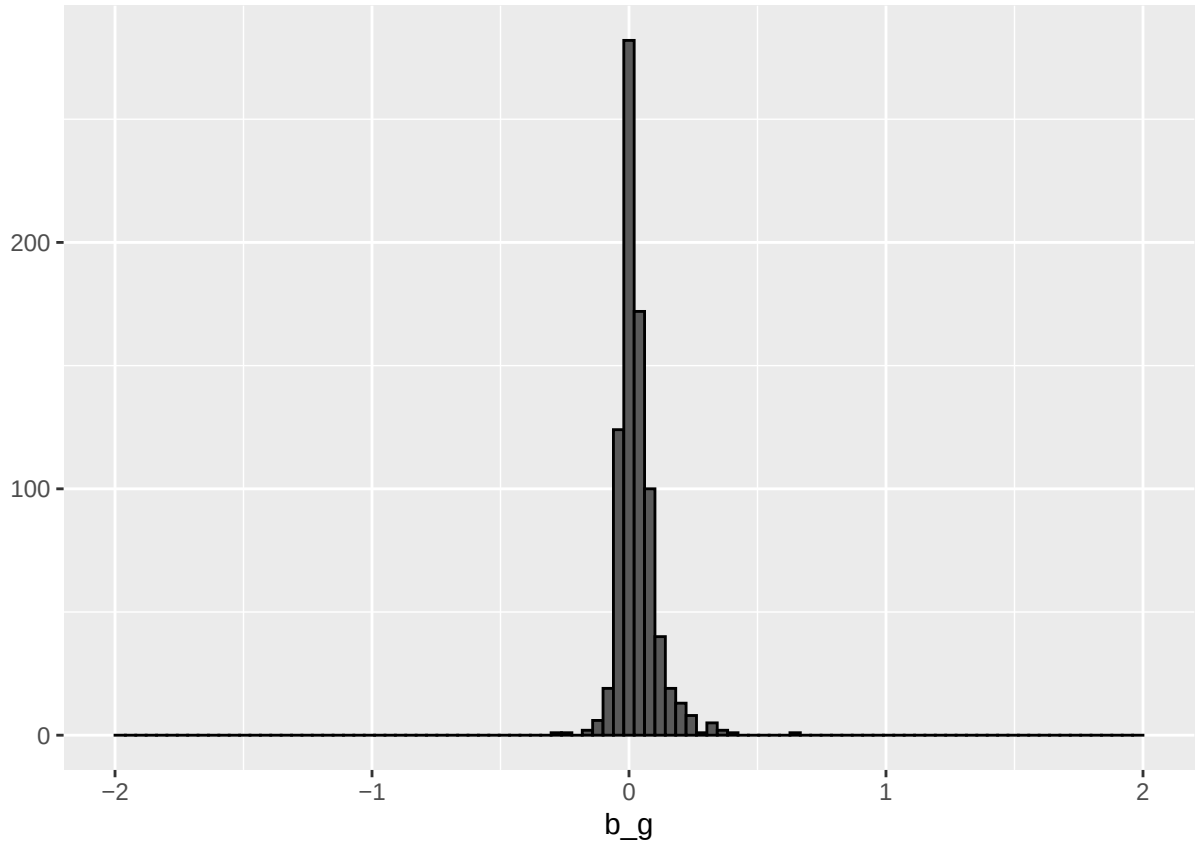
The next step is to include the genres effect.

THE FOURTH MODEL

The 4th model calculate the genres effect b_g from the training set.

$$Y_{u,i,t,g} = \mu + b_i + b_u + b_t + b_g + \varepsilon_{u,i,t,g}$$

We can visualize the distribution of the genres effect on the drawing below:



The effect is not very important, but not negligible.

We continue optimizing the model with regularization.

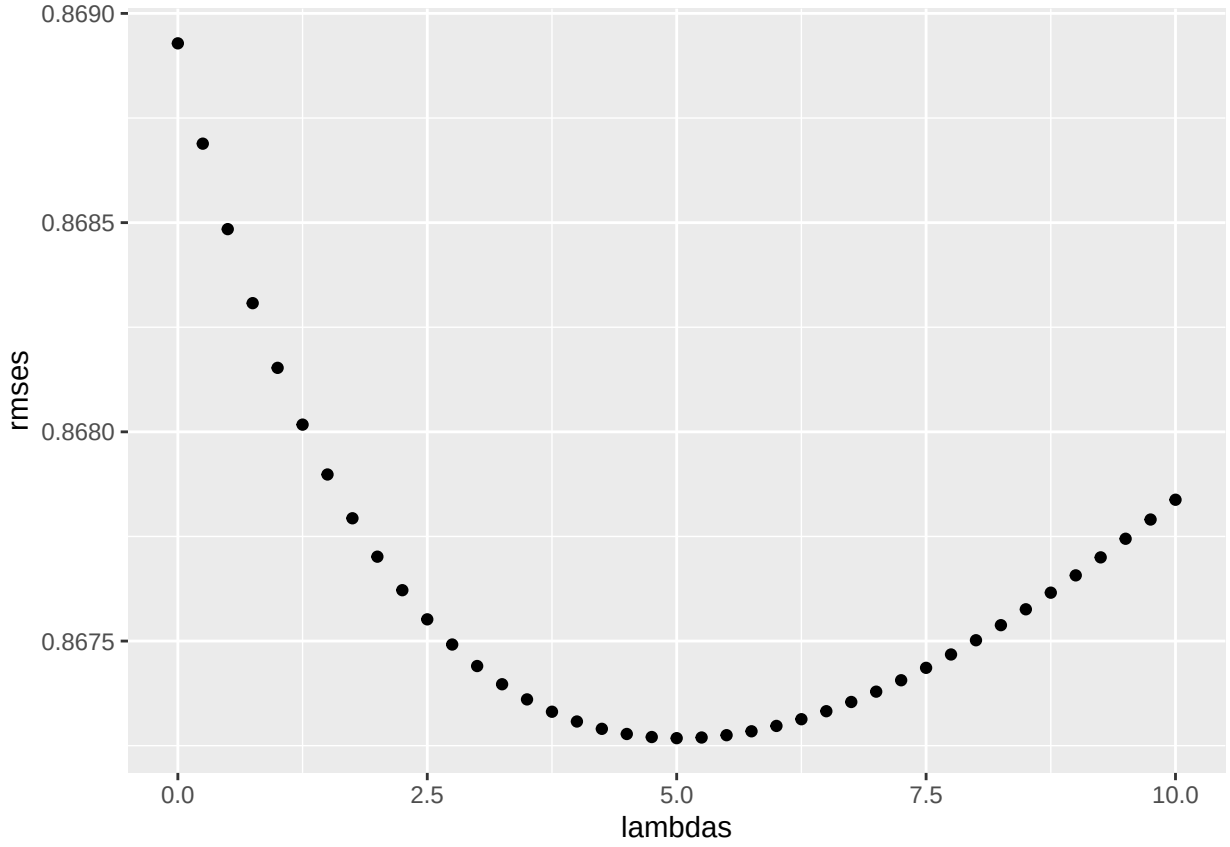
REGULARIZATION

In order to avoid tuning with the test set (validation set), we create a partition on the train set (random cut by half).

We tune lambda λ from 0 to 10 with steps each 0.25 and perform tuning with only the train set.

$$Y_{u,i,t,g} = \mu + b_i(\lambda) + b_u(\lambda) + b_t(\lambda) + b_g(\lambda) + \varepsilon_{u,i,t,g}(\lambda)$$

We obtain the following tuning graph and lambda value:




```
## [1] "the best tune of lambda is 5  "
```

We now obtain the final model for our recommendation movie system.

RESULTS

We now calculate the RMSE of the validation set considering the model with regularization we just built.

All the values of the predictors (userId, movieId, ratingPeriod, Genres) are in the train and test data sets (they have been built in such way), so we just need to apply *leftjoin* of the calculated effects from the training set to the test set with the best tune for the regularization. Then we can calculate the prediction and RMSE.

we get the following table:

Table 11: RMSE results

method	RMSE	target_RMSE
Just the Average Model	1.0612018	< 0.86490
+ Movie Effect Model	0.9439087	< 0.86490
+ User Effect Model	0.8653488	< 0.86490
+ Time Effect Model	0.8649043	< 0.86490
+ Genre Effect Model	0.8645065	< 0.86490
+ Regularization Effect Model	0.8639796	< 0.86490

The results after adding genre effect and regularization is better that the target RMSE required by the project.

CONCLUSION

Due to the huge number of movies and users, our movie recommendation system is not training the data set (it is not feasible).
Instead, we build the model, according to the following formula:

$$Y_{u,i,t,g} = \mu + b_i(\lambda) + b_u(\lambda) + b_t(\lambda) + b_g(\lambda) + \varepsilon_{u,i,t,g}(\lambda)$$

We calculate the mean and we apply the following effects : movies, users, time and genres.

Then, we tune regularization on a partition of the train set to find the best tune of λ .

Finally we apply the models and regularization to the test_set.

We can see the results in the table below:

Table 12: final RMSE

method	RMSE	target_RMSE
Just the Average Model	1.0612018	< 0.86490
+ Movie Effect Model	0.9439087	< 0.86490
+ User Effect Model	0.8653488	< 0.86490
+ Time Effect Model	0.8649043	< 0.86490
+ Genre Effect Model	0.8645065	< 0.86490
+ Regularization Effect Model	0.8639796	< 0.86490

The RMSE obtained by our recommendation movie system with all models + regularization is better than the performance required for the project.