Recommendation system project

Paolo Gabriele

22 April 2021

Contents

INTRODUCTION

The goal of this project is creating from the MovieLens dataset a recommender system, based on the rating, the type of movie and user information.

Several versions of the Movie Lens dataset are available. We will use the MovieLens 10M dataset. This dataset contains approximately 10 Milions of movies ratings, divided in 9 Milions for training and one Milion for validation. Into the training dataset there are approximately **70.000 users** and **11.000 different movies** divided in 20 genres such as Action, Adventure, Horror, Drama, Thriller and more.

After an initial data exploration, we'll build five types of recommender systems. Each of them will be evaluated with the RMSE - Root Mean Squared Error that should be at least lower than **0.86490**.

RMSE = $\sqrt{\frac{1}{n}\sum_{t=1}^{n} e_t^2}$

EXPLORATORY ANALYSIS

The 10 Millions dataset is divided into two dataset: `edx` for training purpose and `validation` for the validation phase.

The `edx` dataset contains approximately 9 Millions of rows with 70.000 different users and 11.000 movies with rating score between 0.5 and 5. There is no missing value (0 or NA) and the column rating has only numbers.

```
## Precisely 9000055 rows
```

```
##  [1] "0.5" "1"   "1.5" "2"   "2.5" "3"   "3.5" "4"   "4.5" "5"
```

```
## [1] "No missing value"
```

```
## [1] "Only numbers in rating"
```

```
## There are 6 variables:
```

```
## Classes 'data.table' and 'data.frame':   9000055 obs. of  6 variables:
##  $ userId   : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ movieId  : num  122 185 292 316 329 355 356 362 364 370 ...
##  $ rating   : num  5 5 5 5 5 5 5 5 5 5 ...
##  $ timestamp: int  838985046 838983525 838983421 838983392 838983392 838984474 838983653 838984885 8
##  $ title    : chr  "Boomerang (1992)" "Net, The (1995)" "Outbreak (1995)" "Stargate (1994)" ...
##  $ genres   : chr  "Comedy|Romance" "Action|Crime|Thriller" "Action|Drama|Sci-Fi|Thriller" "Action|A
##  - attr(*, ".internal.selfref")=<externalptr>
```

1. userId** `<integer>` that contains the unique identification number for each user.

2. movieId** `<numeric>` that contains the unique identification number for each movie.

3. rating** `<numeric>` that contains the rating of one movie by one user. Ratings are made on a 5-Star scale with half-star increments.

4. timestamp `<integer>` that contains the timestamp for one specific rating provided by one user.

5. title** `<character>` that contains the title of each movie including the year of the release.

6. genres** `<character>` that contains a list of pipe-separated of genre of each movie.

Here there is an example of the first rows:

```
## Warning: 'as.tibble()' is deprecated as of tibble 2.0.0.
## Please use 'as_tibble()' instead.
## The signature and semantics have changed, see '?as_tibble'.
## This warning is displayed once every 8 hours.
## Call 'lifecycle::last_warnings()' to see where this warning was generated.

## # A tibble: 9,000,055 x 6
##    userId movieId rating timestamp title                  genres
##     <int>   <dbl>  <dbl>     <int> <chr>                  <chr>
##  1      1     122      5 838985046 Boomerang (1992)       Comedy|Romance
##  2      1     185      5 838983525 Net, The (1995)        Action|Crime|Thriller
##  3      1     292      5 838983421 Outbreak (1995)        Action|Drama|Sci-Fi|T~
##  4      1     316      5 838983392 Stargate (1994)        Action|Adventure|Sci-~
##  5      1     329      5 838983392 Star Trek: Generation~ Action|Adventure|Dram~
##  6      1     355      5 838984474 Flintstones, The (199~ Children|Comedy|Fanta~
##  7      1     356      5 838983653 Forrest Gump (1994)    Comedy|Drama|Romance|~
##  8      1     362      5 838984885 Jungle Book, The (199~ Adventure|Children|Ro~
##  9      1     364      5 838983707 Lion King, The (1994)  Adventure|Animation|C~
## 10      1     370      5 838984596 Naked Gun 33 1/3: The~ Action|Comedy
## # ... with 9,000,045 more rows
```

### Let's explore each variable

a.Genres

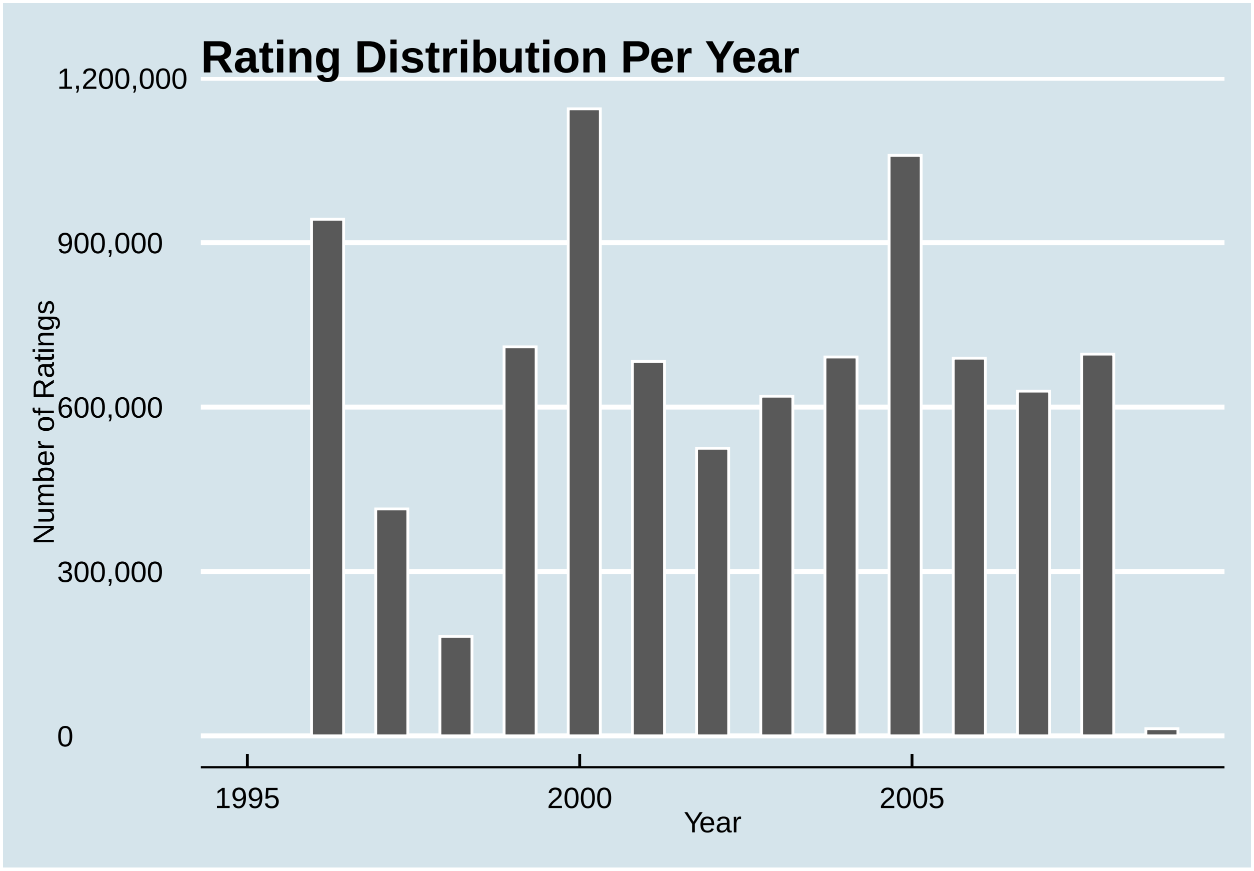
As we can see several movies are classified in more than one genre. The table below lists the first 20 genres sorted in descend order by frequency

```
## # A tibble: 20 x 2
##    genres                             n
##    <chr>                          <int>
##  1 Drama                         733296
##  2 Comedy                        700889
##  3 Comedy|Romance                365468
##  4 Comedy|Drama                  323637
##  5 Comedy|Drama|Romance          261425
##  6 Drama|Romance                 259355
##  7 Action|Adventure|Sci-Fi       219938
##  8 Action|Adventure|Thriller     149091
##  9 Drama|Thriller                145373
## 10 Crime|Drama                   137387
## 11 Drama|War                     111029
## 12 Crime|Drama|Thriller          106101
## 13 Action|Adventure|Sci-Fi|Thriller 105144
## 14 Action|Crime|Thriller         102259
## 15 Action|Drama|War               99183
## 16 Action|Thriller                96535
## 17 Action|Sci-Fi|Thriller         95280
## 18 Thriller                       94662
## 19 Horror|Thriller                75000
## 20 Comedy|Crime                   73286
```

b.Date

The rating period was collected over almost 14 years

```
## # A tibble: 1 x 3
##   `Initial Date` `Final Date` Period
##   <date>         <date>       <Duration>
## 1 1995-01-09     2009-01-05   441479727s (~13.99 years)
```



The following table lists the days with more ratings:

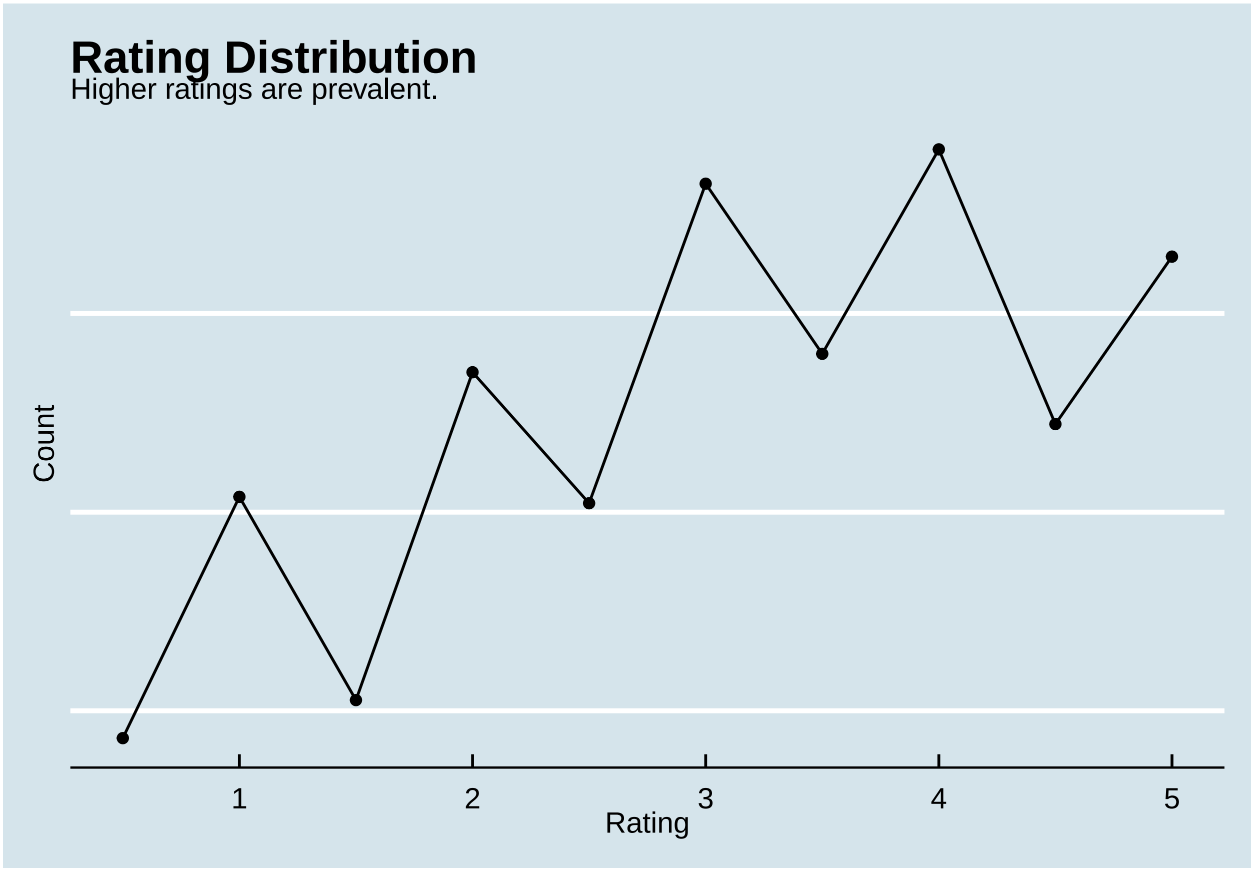
```
## # A tibble: 10 x 3
## # Groups:   date [4]
##    date       title                                                       count
##    <date>     <chr>                                                       <int>
##  1 1998-05-22 Chasing Amy (1997)                                            322
##  2 2000-11-20 American Beauty (1999)                                        277
##  3 1999-12-11 Star Wars: Episode IV - A New Hope (a.k.a. Star Wars) (1977)   254
##  4 1999-12-11 Star Wars: Episode V - The Empire Strikes Back (1980)          251
##  5 1999-12-11 Star Wars: Episode VI - Return of the Jedi (1983)              241
##  6 2005-03-22 Lord of the Rings: The Two Towers, The (2002)                  239
##  7 2005-03-22 Lord of the Rings: The Fellowship of the Ring, The (2001)      227
##  8 2000-11-20 Terminator 2: Judgment Day (1991)                              221
##  9 1999-12-11 Matrix, The (1999)                                             210
## 10 2000-11-20 Jurassic Park (1993)                                           201
```

c.Ratings

Users have the option to choose a rating value from 0.5 to 5.0, totaling 10 possible values.

```
## # A tibble: 10 x 2
##    rating   users
##     <dbl>   <int>
##  1    0.5   85374
##  2    1    345679
##  3    1.5  106426
##  4    2    711422
##  5    2.5  333010
##  6    3   2121240
##  7    3.5  791624
##  8    4   2588430
##  9    4.5  526736
## 10    5   1390114
```

most of the users have rated 3 or more:

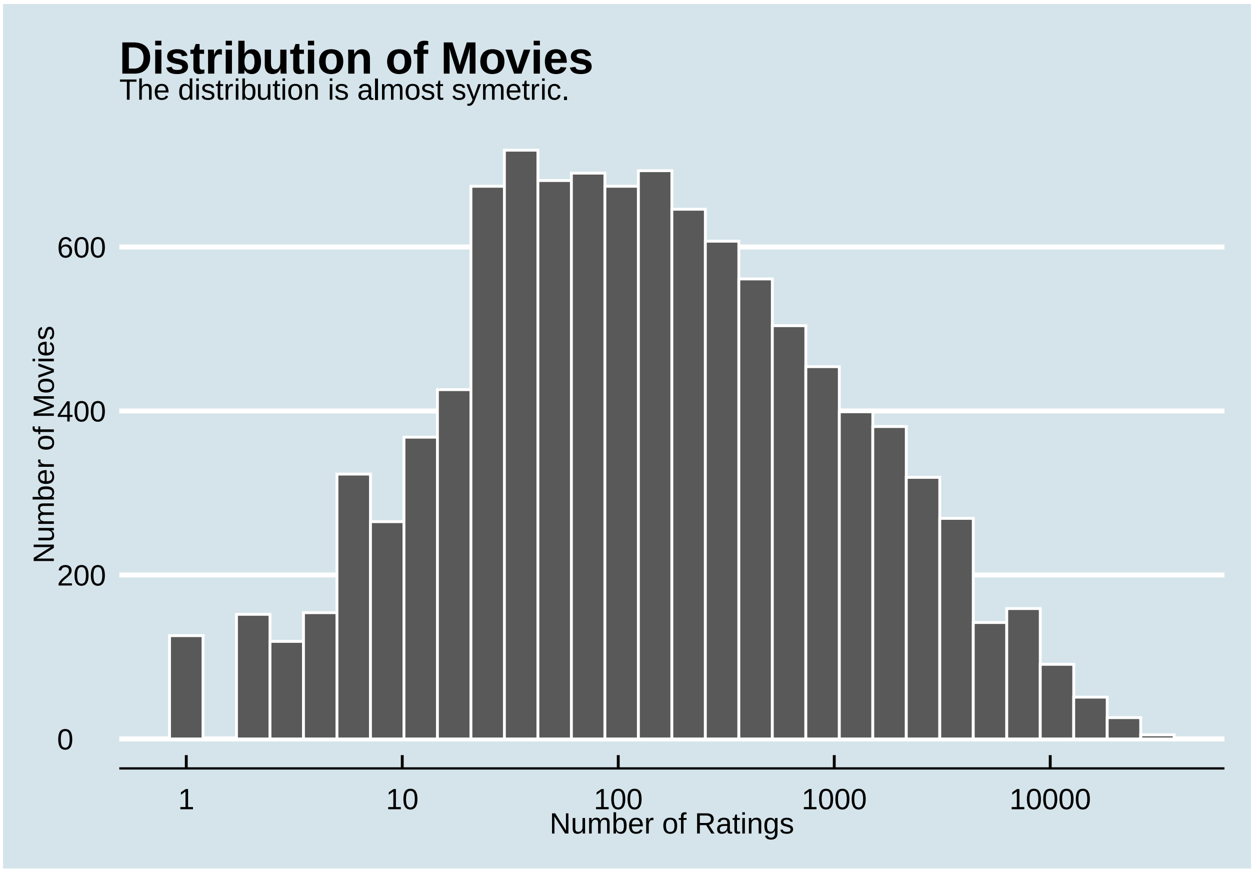


d.Movies

In the edx set there are:

```
## 10677 different movies
```

Some of them are rated more than others:



Roughly 10% of movies have less than 10 ratings

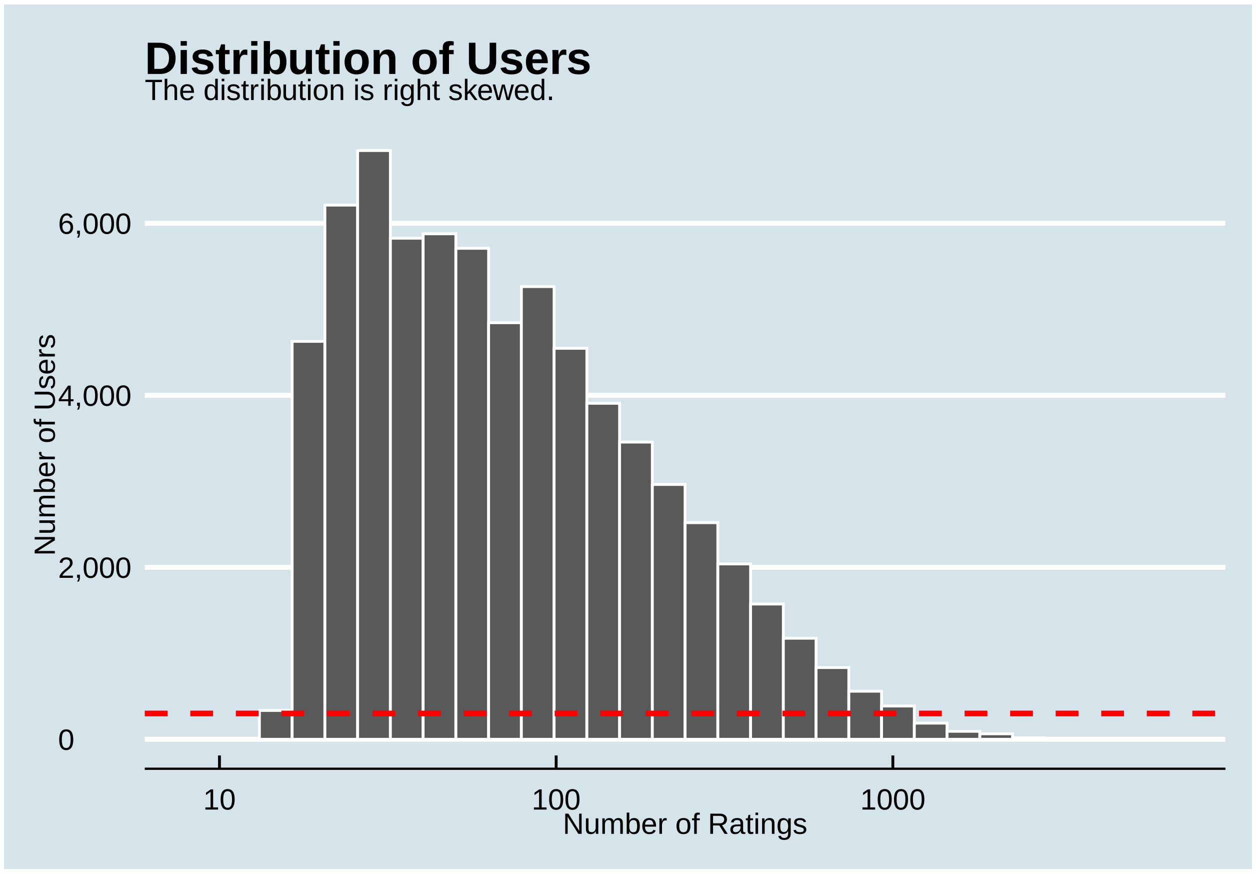
```
## Precisely 9.871687 %
```

e.Users

In the edx set there are

```
## 69878 different users
```

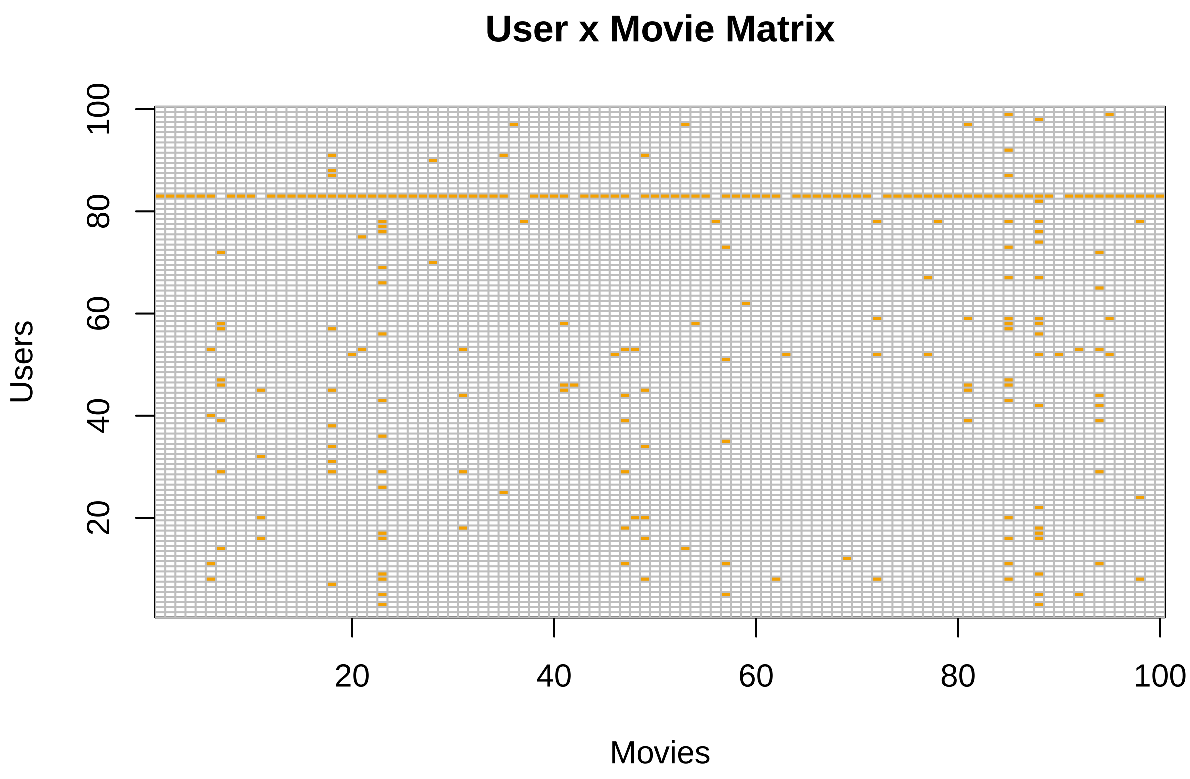
The majority of users rate few movies, while a few users rate more than a thousand movies. That's why the user distribution is right skewed



Roughly 5% of users rated less than 20 movies.

```
## Precisely 4.97 %
```

We can show it better in the sample of edx set with the below heatmap of users x movies Notice that four movies have more ratings, and one or two users are more active



DATA CLEANING

Several features can be used to predict the rating for a given user. However, many predictors increases the model complexity and requires more computer resources, so in this research the estimated rating uses only movie and user information

—-> MODELING

Let's build five models and choose the one which has the best loss function. Our goal is ***RMSE<=0.8649***

———FIRST MODEL- RANDOM PREDICTION

The first model randomly predicts the ratings using the observed probabilities in the training set. Since the training set is a sample of the entire population and we don't know the real distribution of ratings, the Monte Carlo simulation with replacement provides a good approximation of the rating distribution

```
##              Method     RMSE
## 1      Project Goal 0.864900
## 2 Random prediction 1.501867
```

The RMSE of random prediction is **very high** *1.501867*

———SECOND MODEL- MODEL PREDICTION WITH JUST THE AVERAGE

In the second model we predict the same rating for all movies regardless of user.We can use a model that assumes the same rating for all movies and users with all the differences explained by random variation would look like this

$Y_{\mu,\epsilon} = \mu + \epsilon$

where ϵ is the independent error sampled from the same distribution centered at 0 and μ is the "true" rating for all movies, that is in our case the average of all ratings

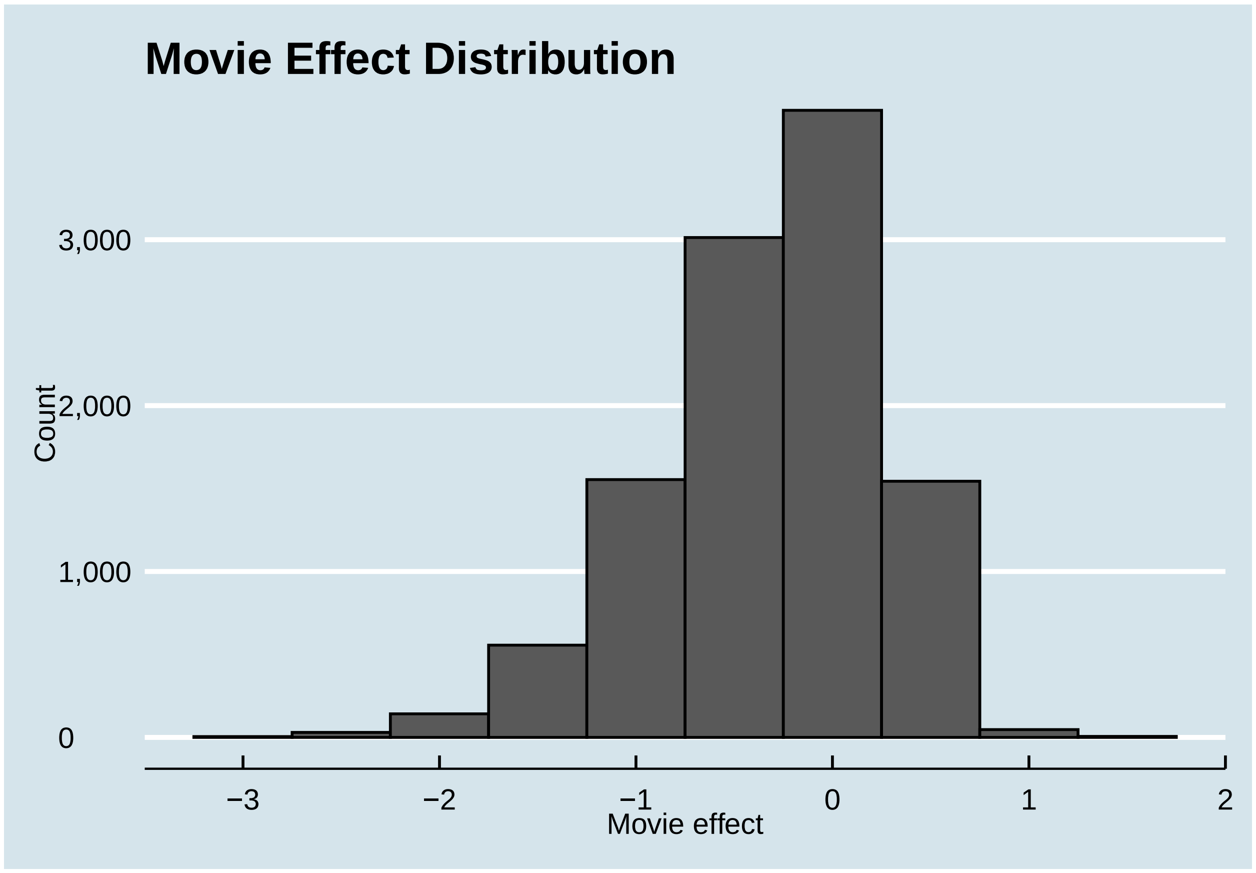
```
##             Method     RMSE
## 1     Project Goal 0.864900
## 2 Random prediction 1.501867
## 3  Just the average 1.061202
```

We have obtained a light improvement *1.061202*

———THIRD MODEL- MODEL PREDICTION WITH JUST THE AVERAGE+ MOVIE EFFECT

In the third model we take into account that some movies are rated higher than others. We can augment our previous model by adding the term b_i to represent average ranking for movie i. In our case b_i is just the average of $Y - \mu$ for each movie i

```
## # A tibble: 6 x 2
##   movieId    b_i
##     <dbl>  <dbl>
## 1       1  0.415
## 2       2 -0.307
## 3       3 -0.365
## 4       4 -0.648
## 5       5 -0.444
## 6       6  0.303
```



```
##                  Method      RMSE
## 1          Project Goal 0.8649000
## 2     Random prediction 1.5018674
## 3      Just the average 1.0612018
## 4 just the average + bi 0.9439087
```

We have still obtained a little improvement *0.9439615*

———FORTH MODEL- MODEL PREDICTION WITH JUST THE AVERAGE+MOVIE EFFECT+USER EFFECT

But we can do better considering the user effect or in other words the effect depending on the type of user, as some are very selective and others rate all. So we can improve our model this way:

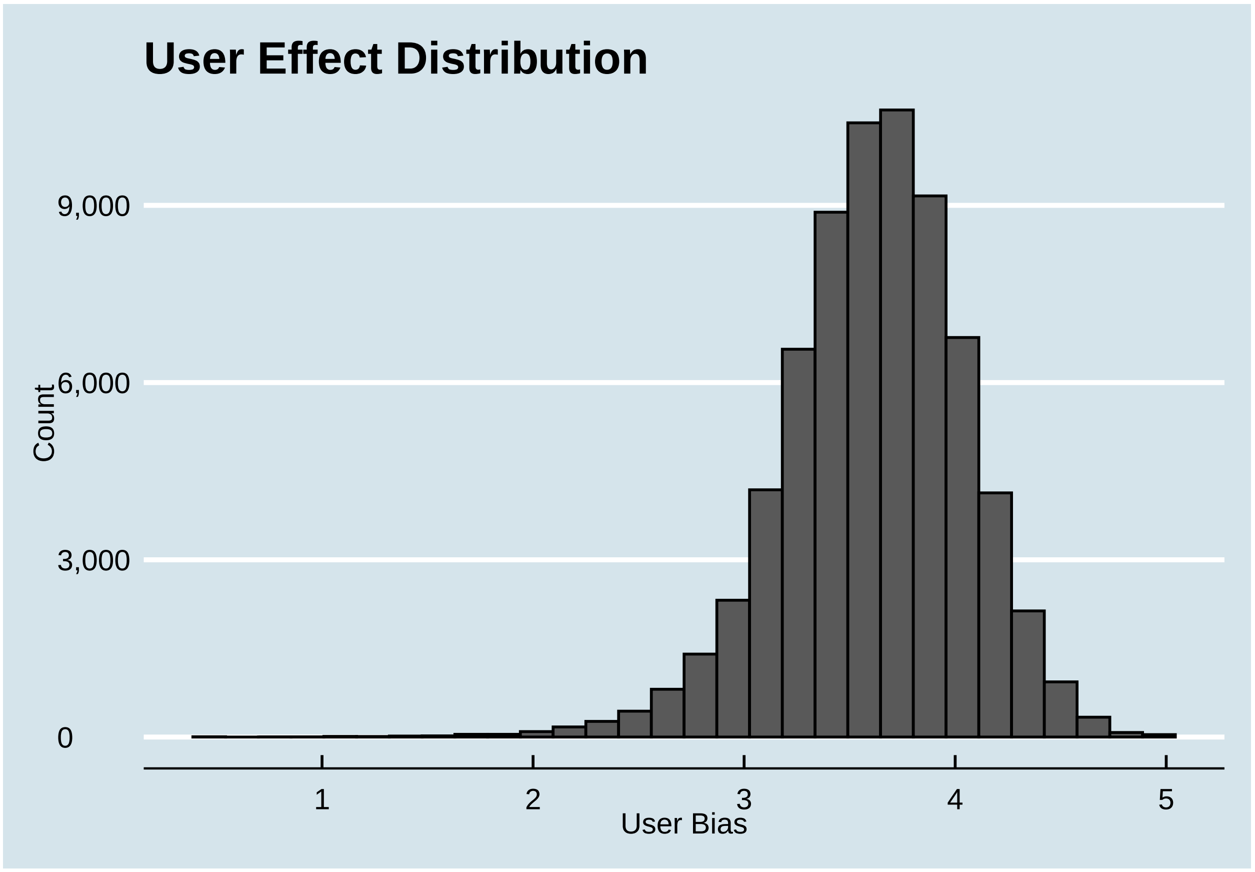
$Y_{u,i} = \mu + b_i + b_u + \varepsilon_{u,i}$

where b_u is a user-specific effect. Now if a cranky user (negative b_u) rates a great movie (positive b_i), the effects counter each other and we may be able to correctly predict that this user gave this great movie a 3 rather than a 5.

```
##                       Method      RMSE
## 1               Project Goal 0.8649000
## 2          Random prediction 1.5018674
## 3           Just the average 1.0612018
## 4      just the average + bi 0.9439087
## 5 just the average + bi + bu 0.8653488
```

As expected we still have an improvement *0.8646843*

The user effect is normally distributed



The final RMSE in not as good as we want yet. To do better we have to consider the frequency of rating. Let's see the top 10 best movies (ranked by bi). These seem unknown movies

```
## # A tibble: 10 x 1
##    title
##    <chr>
##  1 Hellhounds on My Trail (1999)
##  2 Satan's Tango (SÃ¡tÃ¡ntangÃ³) (1994)
##  3 Shadows of Forgotten Ancestors (1964)
##  4 Fighting Elegy (Kenka erejii) (1966)
##  5 Sun Alley (Sonnenallee) (1999)
##  6 Blue Light, The (Das Blaue Licht) (1932)
##  7 Who's Singin' Over There? (a.k.a. Who Sings Over There) (Ko to tamo peva) (1~
##  8 Human Condition II, The (Ningen no joken II) (1959)
##  9 Human Condition III, The (Ningen no joken III) (1961)
## 10 Constantine's Sword (2007)
```

Let's look at how often they are rated

```
##  [1] 1 2 1 1 1 1 4 4 4 2
```

Let's look now at the top 10 best movies (ranked by bi). These seem unknown movies too

```
## # A tibble: 10 x 1
##    title
##    <chr>
##  1 Besotted (2001)
##  2 Hi-Line, The (1999)
##  3 Accused (Anklaget) (2005)
##  4 Confessions of a Superhero (2007)
##  5 War of the Worlds 2: The Next Wave (2008)
##  6 SuperBabies: Baby Geniuses 2 (2004)
##  7 Hip Hop Witch, Da (2000)
##  8 Disaster Movie (2008)
##  9 From Justin to Kelly (2003)
## 10 Criminals (1996)
```

Let's look at how often they are rated

```
##  [1]   2   1   1   1   2  56  14  32 199   2
```

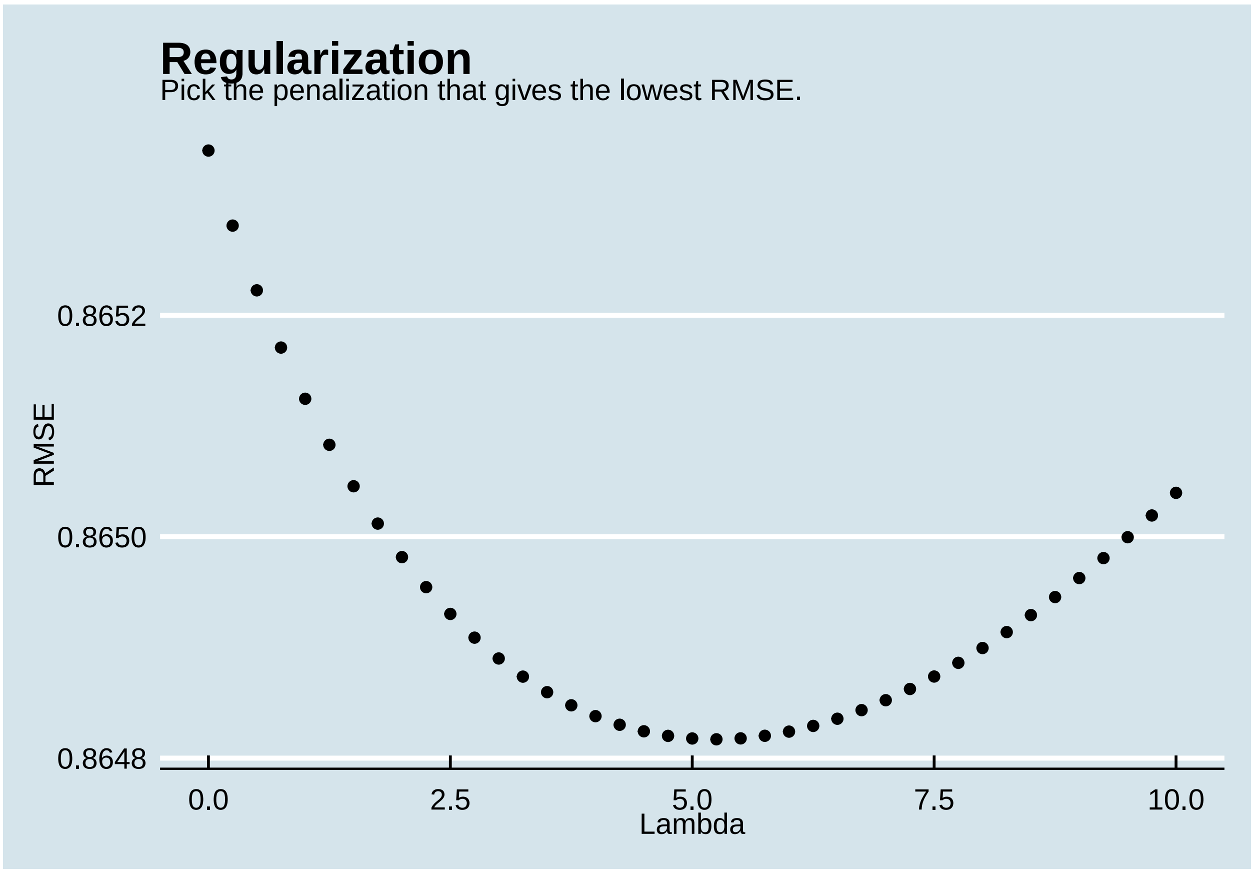
The supposed* "best"* and *"worst"* movies were rated in general by very few users, which could give more uncertainty to the result

———FIFTH MODEL- REGULARIZATION

To correct the uncertainty we regularize the user and movie effects adding a penalty factor lambda, that penalizes small sample sizes and has little or no impact otherwise. Thus, estimated movie and user effects can be calculated with these formulas which is a parameter. We define a number of values for lambda and use the regularizatION function to pick the best value that minimizes the RMSE

We can go further defining a set of lambdas to tune

Let's plot the lambda vs RMSE



Finally, we apply the best lambda to the linear model. We pick the lambda that returns the lowest RMSE and calculate the predicted rating using the best parameters

Here is the result table

```
##                         Method      RMSE
## 1                 Project Goal 0.8649000
## 2            Random prediction 1.5018674
## 3             Just the average 1.0612018
## 4         just the average + bi 0.9439087
## 5 just the average + bi + bu 0.8653488
## 6        Regularized bi and bu 0.8648170
```

Regularization made the expected final improvement in RMSE to hit our mark

—->FINAL VALIDATION

As we can see from the result table, regularization achieved the target RMSE. Now we calculate the RMSE in the validation set. The project goal is achieved if the RMSE stays below the target ***0.86490***

```
##                                      Method      RMSE
## 1                              Project Goal 0.8649000
## 2                         Random prediction 1.5018674
## 3                          Just the average 1.0612018
## 4                     just the average + bi 0.9439087
## 5                just the average + bi + bu 0.8653488
## 6                     Regularized bi and bu 0.8648170
## 7 Final Regularization (edx vs validation) 0.8648170
```

As expected, the RMSE calculated on the validation set (0.8648177) is lower than the target of 0.8649. Let's see what are the top 10 best movies with our recommendation system:

```
## # A tibble: 10 x 1
## # Groups:   title [7]
##    title
##    <chr>
##  1 Usual Suspects, The (1995)
##  2 Shawshank Redemption, The (1994)
##  3 Shawshank Redemption, The (1994)
##  4 Shawshank Redemption, The (1994)
##  5 Eternal Sunshine of the Spotless Mind (2004)
##  6 Star Wars: Episode IV - A New Hope (a.k.a. Star Wars) (1977)
##  7 Schindler's List (1993)
##  8 Donnie Darko (2001)
##  9 Star Wars: Episode VI - Return of the Jedi (1983)
## 10 Schindler's List (1993)
```

and the yop 10 worst movies

```
## # A tibble: 10 x 1
## # Groups:   title [9]
##    title
##    <chr>
##  1 Battlefield Earth (2000)
##  2 Police Academy 4: Citizens on Patrol (1987)
##  3 Karate Kid Part III, The (1989)
##  4 PokÃ©mon Heroes (2003)
##  5 Turbo: A Power Rangers Movie (1997)
##  6 Kazaam (1996)
##  7 Free Willy 3: The Rescue (1997)
##  8 PokÃ©mon Heroes (2003)
##  9 Shanghai Surprise (1986)
## 10 Steel (1997)
```

CONCLUSION

The goal of this project is creating from the **MovieLens dataset** a recommender system, based on the rating, the type of movie and user information. We have used the 10M version of the MovieLens dataset to make the computation a little easier. The dataset has been divided in a training set, called edx and in a test set, called validation. After exploring the edx dataset variables, we have reduced the model complexity keeping only movie and user informations to estimate the ratings. In the training set we have tested five models: the first model randomly predicted the ratings using the observed probabilities, the second one predicted the same rating for all movies regardless of user, the third took into account that some movies were rated higher than others, the forth considered the user effect, the fifth regularized the user and movie effects adding a penalty factor lambda. The last one has been chosen for the best RMSE (<0.864), The best model with regularization has been applied to the validation set and, as expected, we have obtained the same good result for the RMSE, namely 0.86490.