Movielens

annalee0107

2022/2/18

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this

Packages used :

tidyverse,caret, data.table, lubridate, ggplot2

Movielens Dataset:

MovieLens 10M dataset: https://grouplens.org/datasets/movielens/10m/

MovieLens 10M movie ratings. Stable benchmark dataset. 10 million ratings and 100,000 tag applications applied to 10,000 movies by 72,000 users. Released 1/2009.

Edx dataset:

Edx dataset is ceated as training set. Validation dataset is created as testing set

Number of rows and columns in edx dataset :

```
## [1] 9000055       6
```

First 5 rows overiew in edx dataset :

```
##    userId movieId rating timestamp                         title
## 1:      1     122      5 838985046              Boomerang (1992)
## 2:      1     185      5 838983525               Net, The (1995)
## 3:      1     292      5 838983421               Outbreak (1995)
## 4:      1     316      5 838983392               Stargate (1994)
## 5:      1     329      5 838983392 Star Trek: Generations (1994)
##                            genres
## 1:                Comedy|Romance
## 2:         Action|Crime|Thriller
## 3:  Action|Drama|Sci-Fi|Thriller
## 4:       Action|Adventure|Sci-Fi
## 5: Action|Adventure|Drama|Sci-Fi
```

How many zeros, threes were given as ratings in the edx dataset:

```
##   n()
## 1   0
```

```
##       n()
## 1 2121240
```

How many movie ratings are in each of the following genres in the edx dataset?

```
## [1] "Drama"
```

```
## [1] 3910127
```

```
## [1] "Comedy"
```

```
## [1] 3540930
```

```
## [1] "Thriller"
```

```
## [1] 2325899
```

```
## [1] "Romance"
```

```
## [1] 1712100
```

Number of userId and movieId in edx dataset :

```
##   n_users n_movies
## 1   69878    10677
```

How many ratings by movie title in the edx dataset:

```
## [1] "by title:"
```

```
## # A tibble: 10,676 x 2
##    title                                                     n
##    <chr>                                                 <int>
##  1 Pulp Fiction (1994)                                   31362
##  2 Forrest Gump (1994)                                   31079
##  3 Silence of the Lambs, The (1991)                      30382
##  4 Jurassic Park (1993)                                  29360
##  5 Shawshank Redemption, The (1994)                      28015
##  6 Braveheart (1995)                                     26212
##  7 Fugitive, The (1993)                                  25998
##  8 Terminator 2: Judgment Day (1991)                     25984
##  9 Star Wars: Episode IV - A New Hope (a.k.a. Star Wars) (1~ 25672
## 10 Apollo 13 (1995)                                      24284
## # ... with 10,666 more rows
```

```
## [1] "by rating:"
```

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

Convert timestamp into date in edx dataset , Extract movie year from title and calculate yearlapsed (between movie year and rating year) in edx dataset :

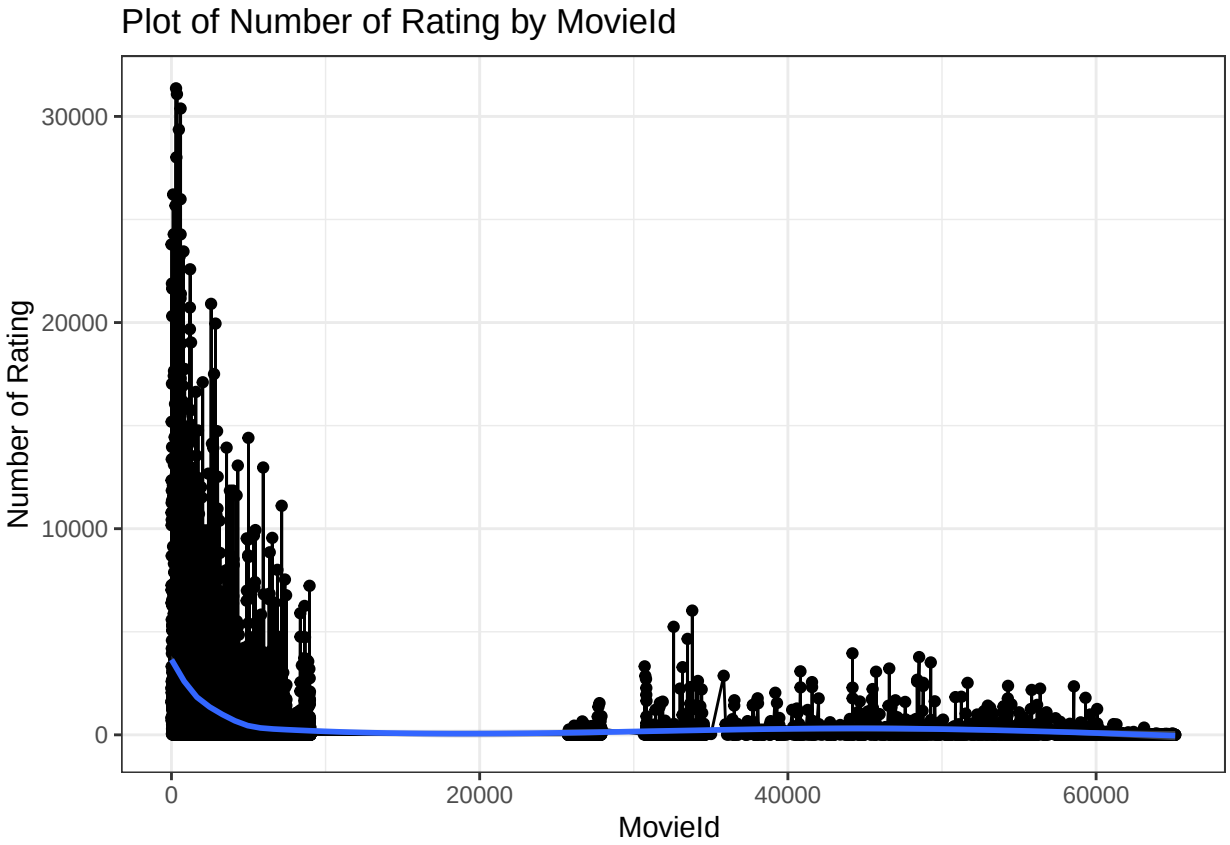
```
##    userId movieId rating myear                       genres
## 1:      1     122      5  1992               Comedy|Romance
## 2:      1     185      5  1995         Action|Crime|Thriller
## 3:      1     292      5  1995  Action|Drama|Sci-Fi|Thriller
## 4:      1     316      5  1994       Action|Adventure|Sci-Fi
## 5:      1     329      5  1994 Action|Adventure|Drama|Sci-Fi
## 6:      1     355      5  1994       Children|Comedy|Fantasy
##                    date       week      month ryear yearlapsed
## 1: 1996-08-02 11:24:06 1996-08-04 1996-08-01  1996          4
## 2: 1996-08-02 10:58:45 1996-08-04 1996-08-01  1996          1
## 3: 1996-08-02 10:57:01 1996-08-04 1996-08-01  1996          1
## 4: 1996-08-02 10:56:32 1996-08-04 1996-08-01  1996          2
## 5: 1996-08-02 10:56:32 1996-08-04 1996-08-01  1996          2
## 6: 1996-08-02 11:14:34 1996-08-04 1996-08-01  1996          2
```

Number of ratings and mean rating plots for edx dataset:

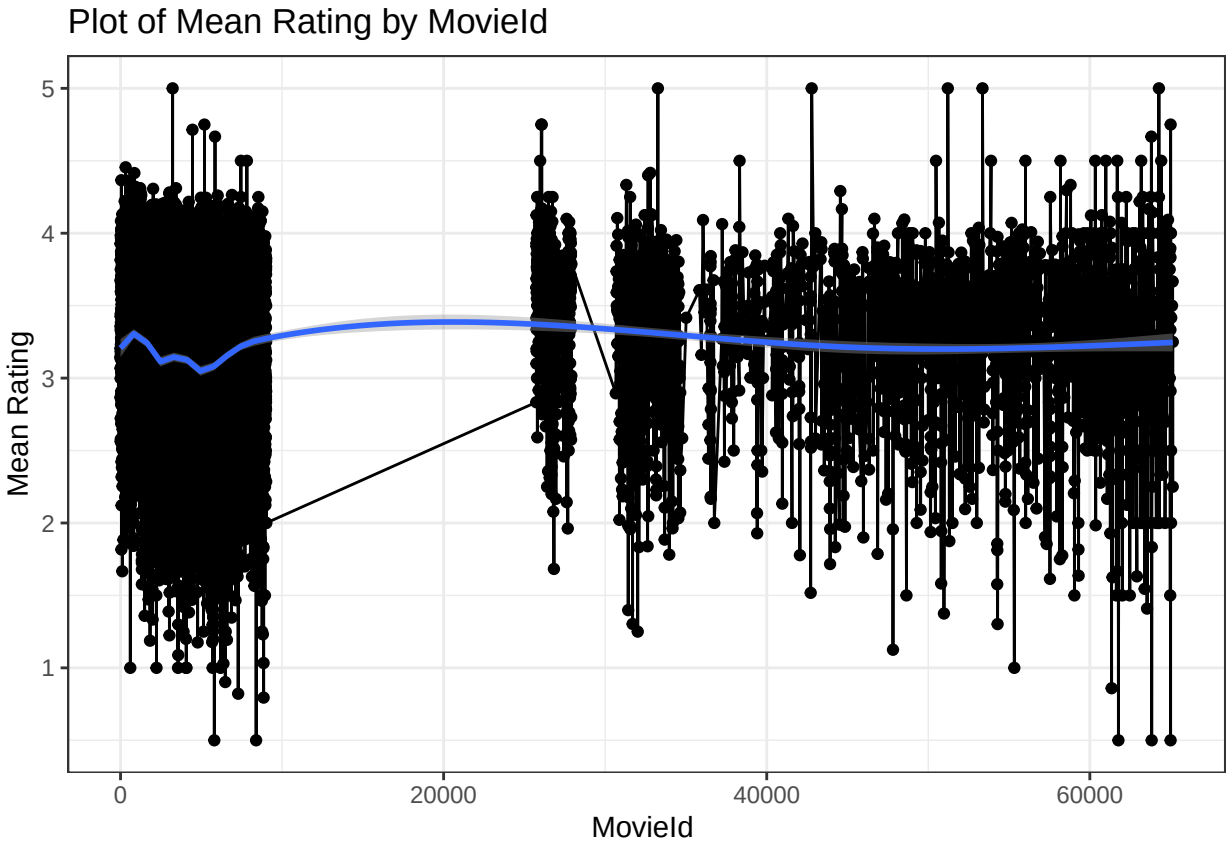
Number of ratings and mean rating by movieId:

```
## # A tibble: 10,677 x 3
##    movieId     n avg_rating
##      <dbl> <int>      <dbl>
##  1     296 31362       4.15
##  2     356 31079       4.01
##  3     593 30382       4.20
##  4     480 29360       3.66
##  5     318 28015       4.46
##  6     110 26212       4.08
##  7     457 25998       4.01
##  8     589 25984       3.93
##  9     260 25672       4.22
## 10     150 24284       3.89
## # ... with 10,667 more rows
```

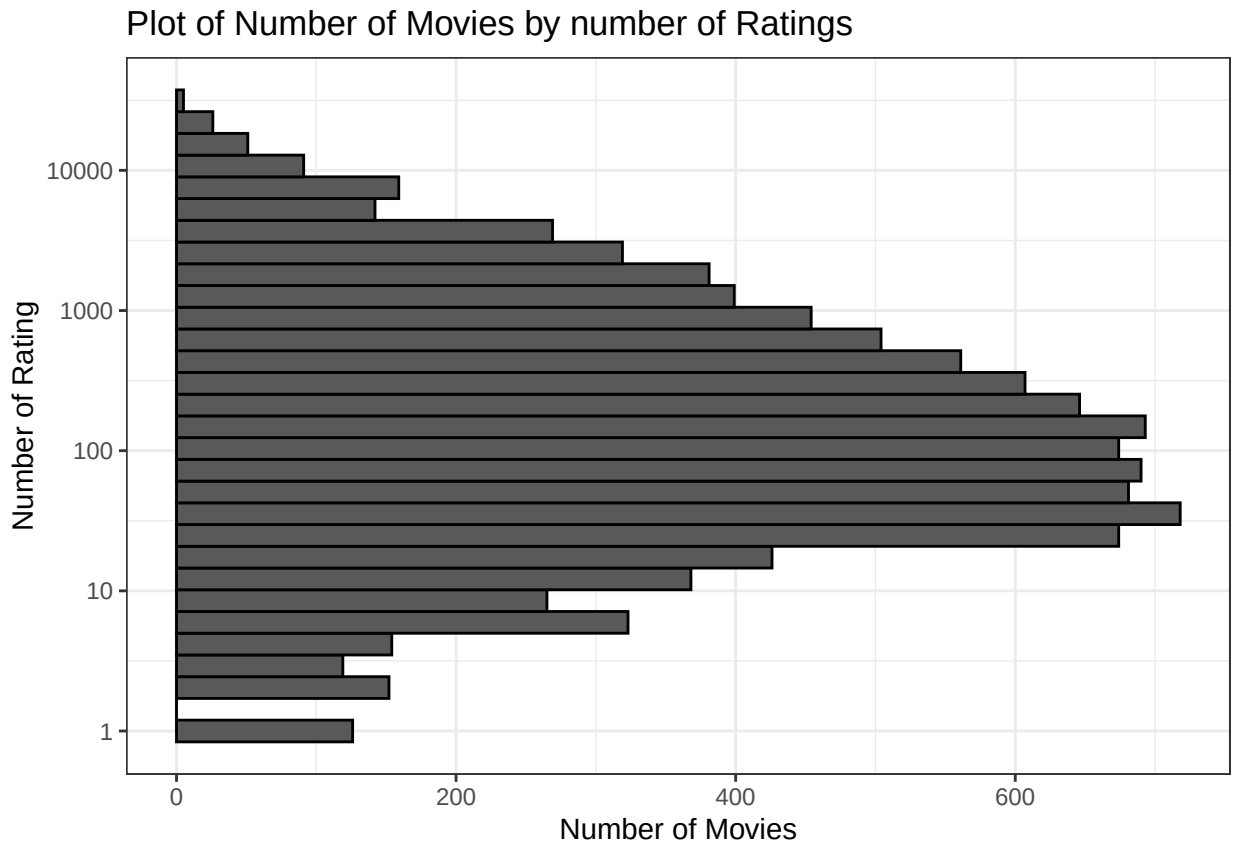
```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```




```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```


Plot of Mean Rating by MovieId
Mean Rating
5
4
3
2
1
0
20000
40000
60000
MovieId

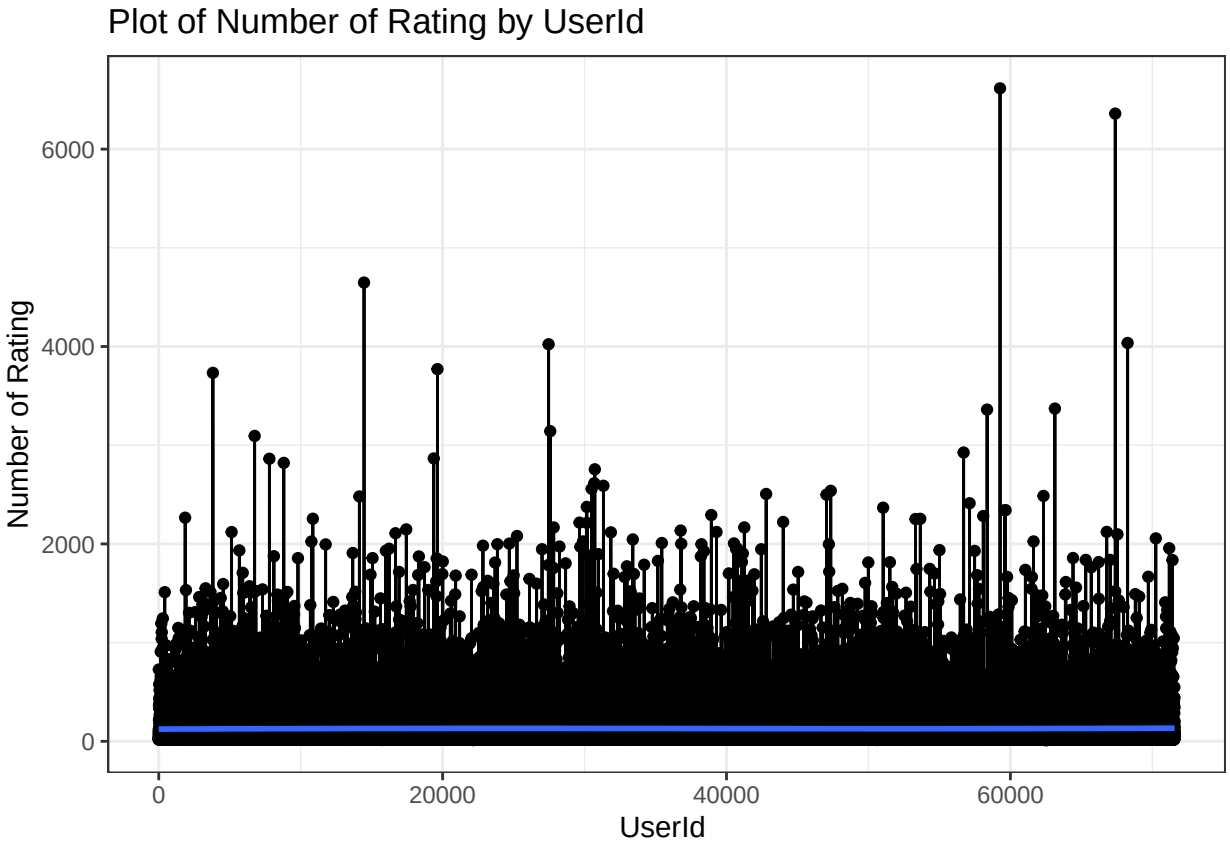
Number of movies by number of ratings :



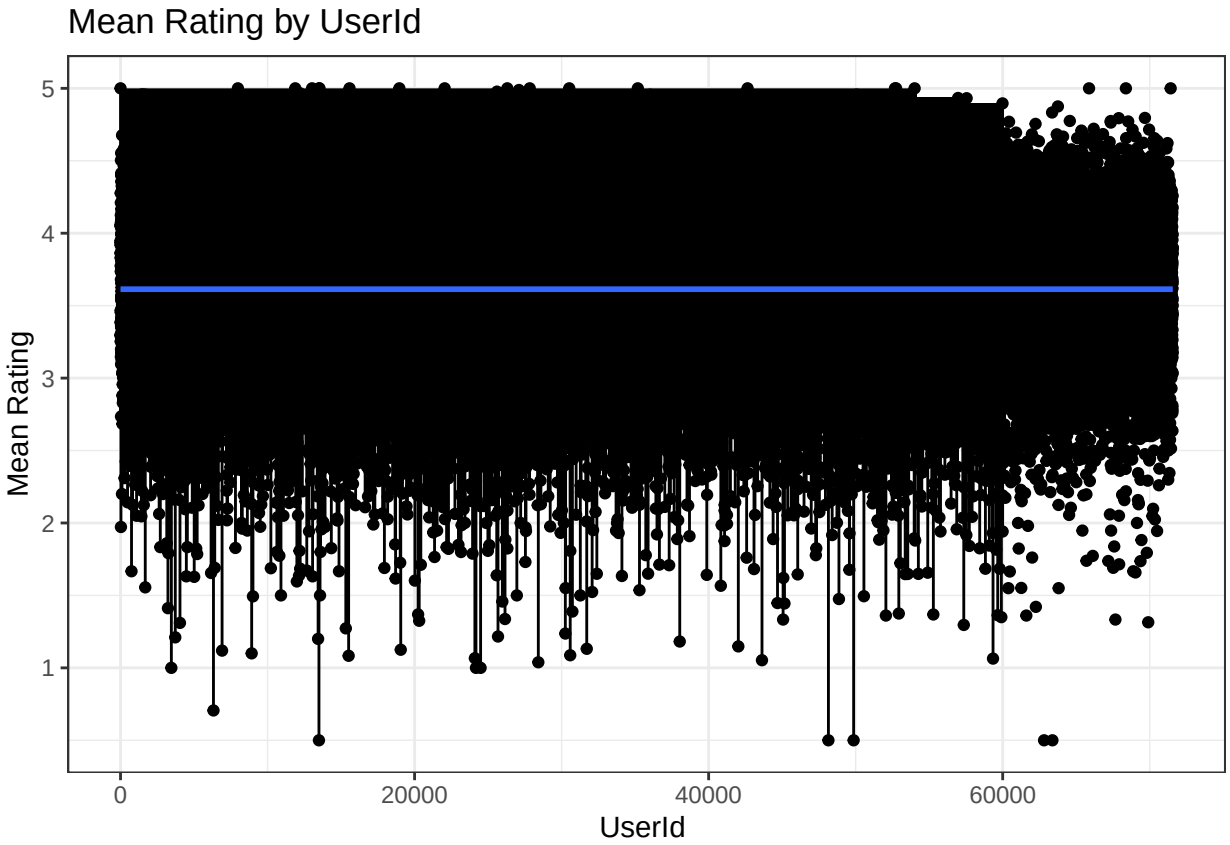
Number of ratings and mean rating by userId:

```
## # A tibble: 69,878 x 3
##    userId     n avg_rating
##     <int> <int>      <dbl>
##  1  59269  6616       3.26
##  2  67385  6360       3.20
##  3  14463  4648       2.40
##  4  68259  4036       3.58
##  5  27468  4023       3.83
##  6  19635  3771       3.50
##  7   3817  3733       3.11
##  8  63134  3371       3.27
##  9  58357  3361       3.00
## 10  27584  3142       3.00
## # ... with 69,868 more rows
```

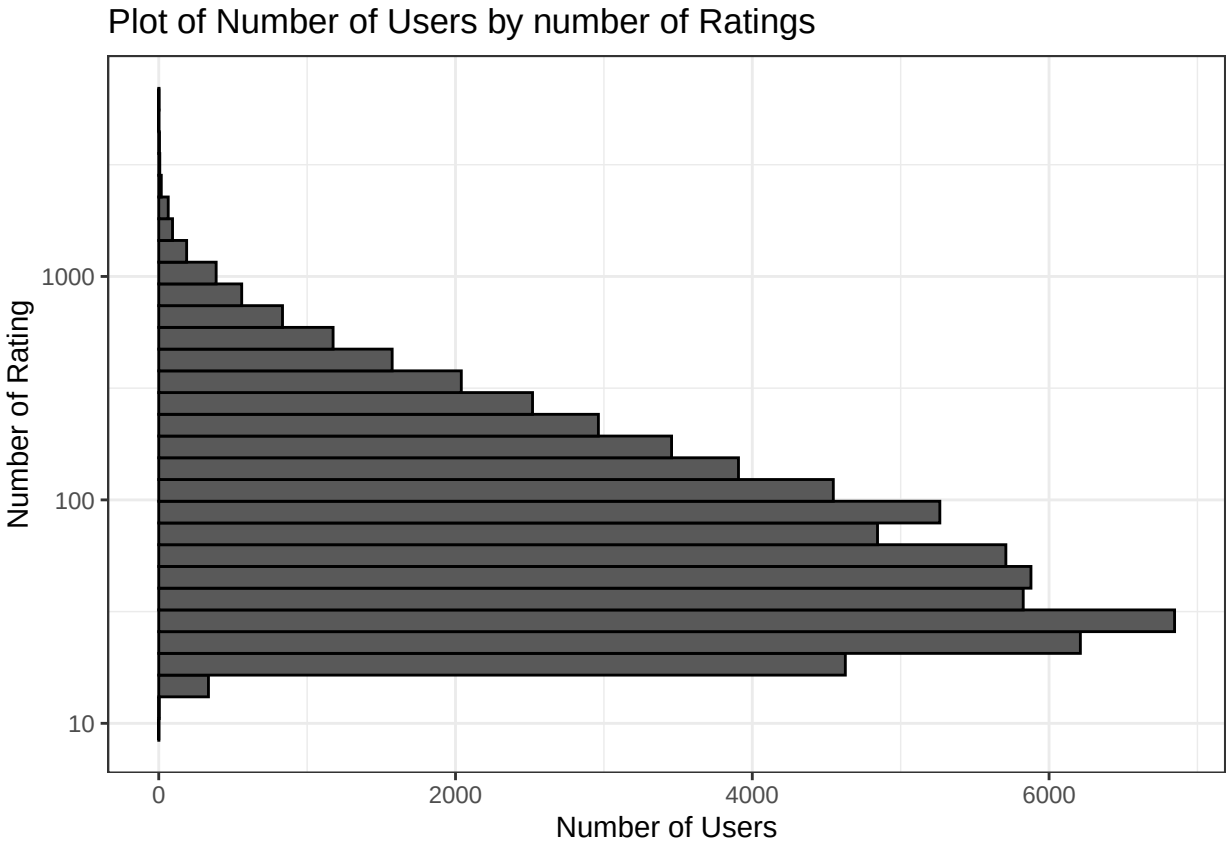
```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```




```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



Number of users by number of ratings :

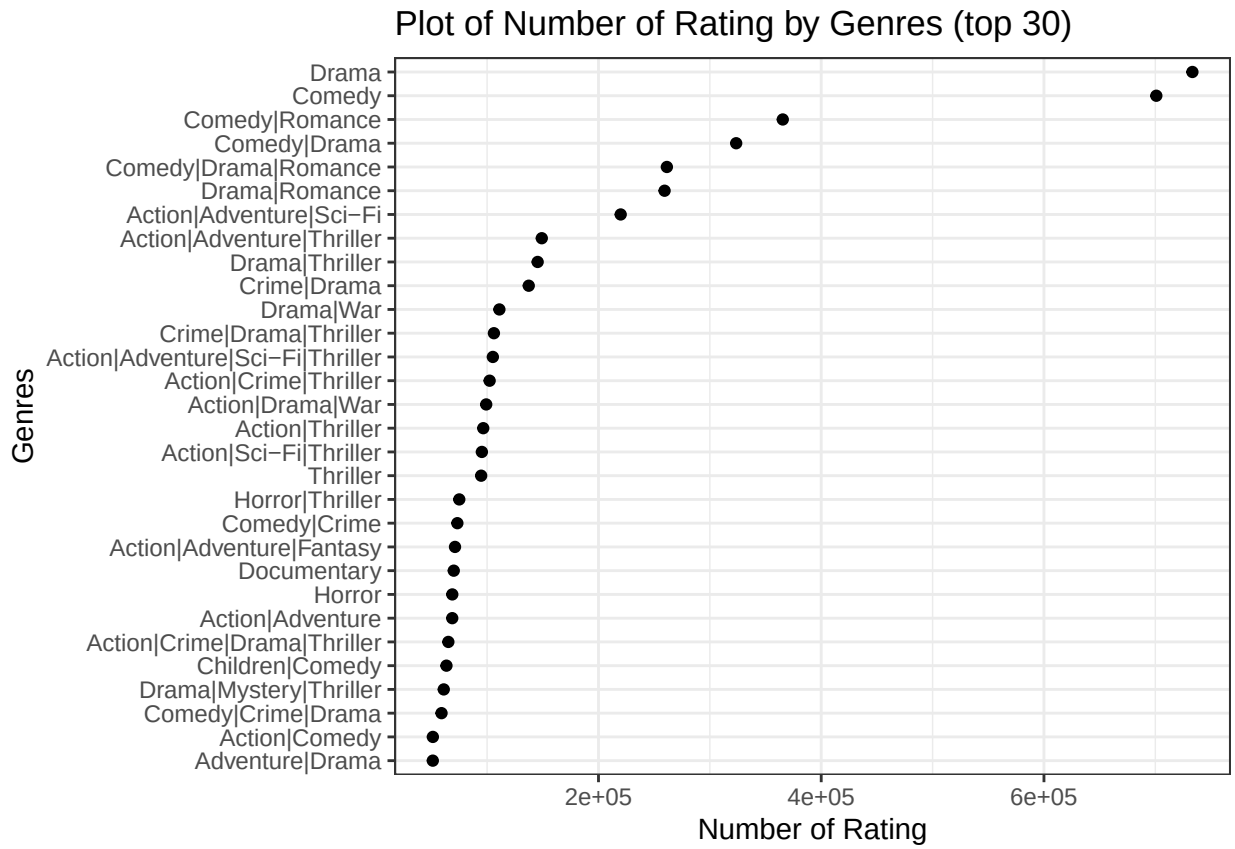


Number of ratings and mean rating by genres:

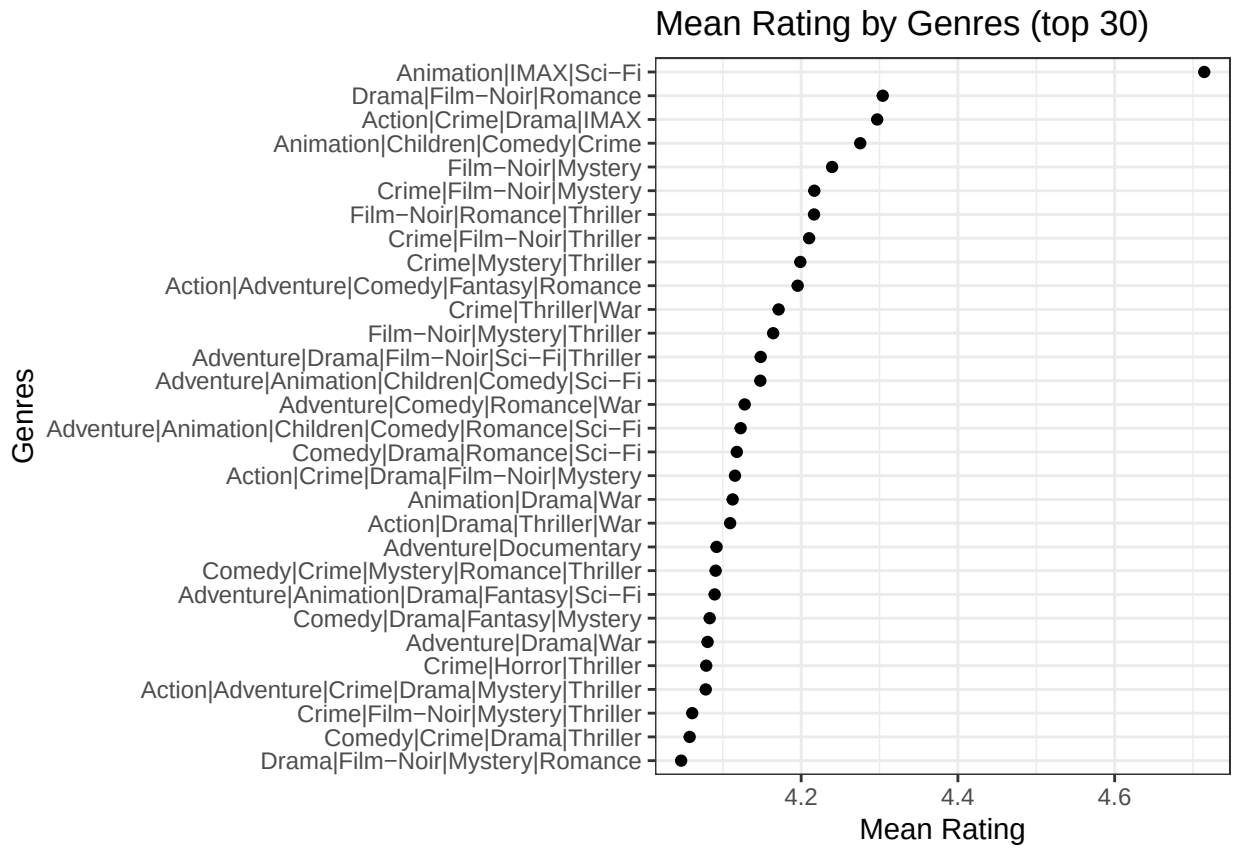
```
## # A tibble: 797 x 3
##    genres                          n avg_rating
##    <chr>                       <int>      <dbl>
##  1 Drama                      733296       3.71
##  2 Comedy                     700889       3.24
##  3 Comedy|Romance             365468       3.41
##  4 Comedy|Drama               323637       3.60
##  5 Comedy|Drama|Romance       261425       3.65
##  6 Drama|Romance              259355       3.61
##  7 Action|Adventure|Sci-Fi    219938       3.51
##  8 Action|Adventure|Thriller 149091       3.43
##  9 Drama|Thriller             145373       3.45
## 10 Crime|Drama                137387       3.95
## # ... with 787 more rows
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

```
## geom_path: Each group consists of only one observation. Do you
## need to adjust the group aesthetic?
```




```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
## geom_path: Each group consists of only one observation. Do you
## need to adjust the group aesthetic?
```

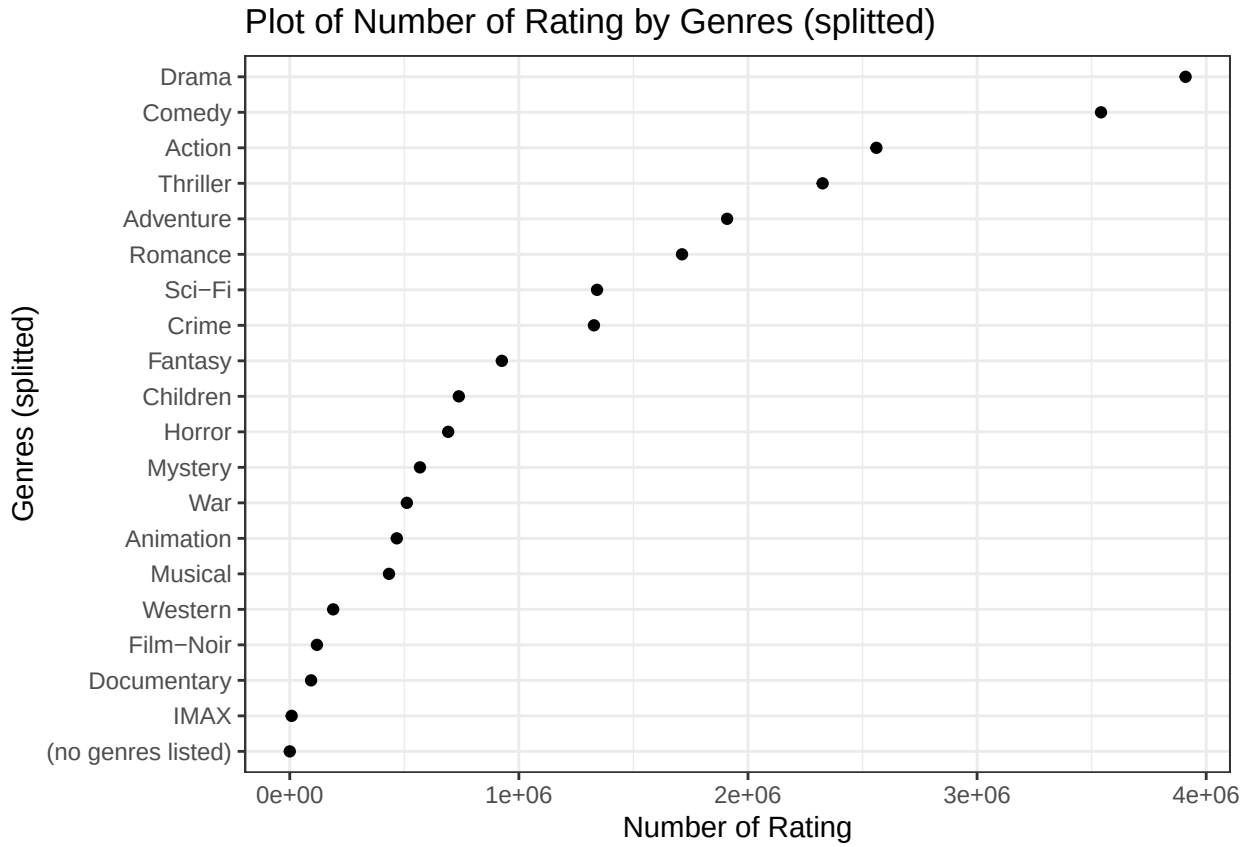


Number of ratings and mean rating by splitted genres:

```
## # A tibble: 20 x 3
##    genres2                 n avg_rating
##    <chr>               <int>      <dbl>
##  1 Drama             3910127       3.67
##  2 Comedy            3540930       3.44
##  3 Action            2560545       3.42
##  4 Thriller          2325899       3.51
##  5 Adventure         1908892       3.49
##  6 Romance           1712100       3.55
##  7 Sci-Fi            1341183       3.40
##  8 Crime             1327715       3.67
##  9 Fantasy            925637       3.50
## 10 Children           737994       3.42
## 11 Horror             691485       3.27
## 12 Mystery            568332       3.68
## 13 War                511147       3.78
## 14 Animation          467168       3.60
## 15 Musical            433080       3.56
## 16 Western            189394       3.56
## 17 Film-Noir          118541       4.01
## 18 Documentary         93066       3.78
## 19 IMAX                 8181       3.77
## 20 (no genres listed)      7       3.64
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

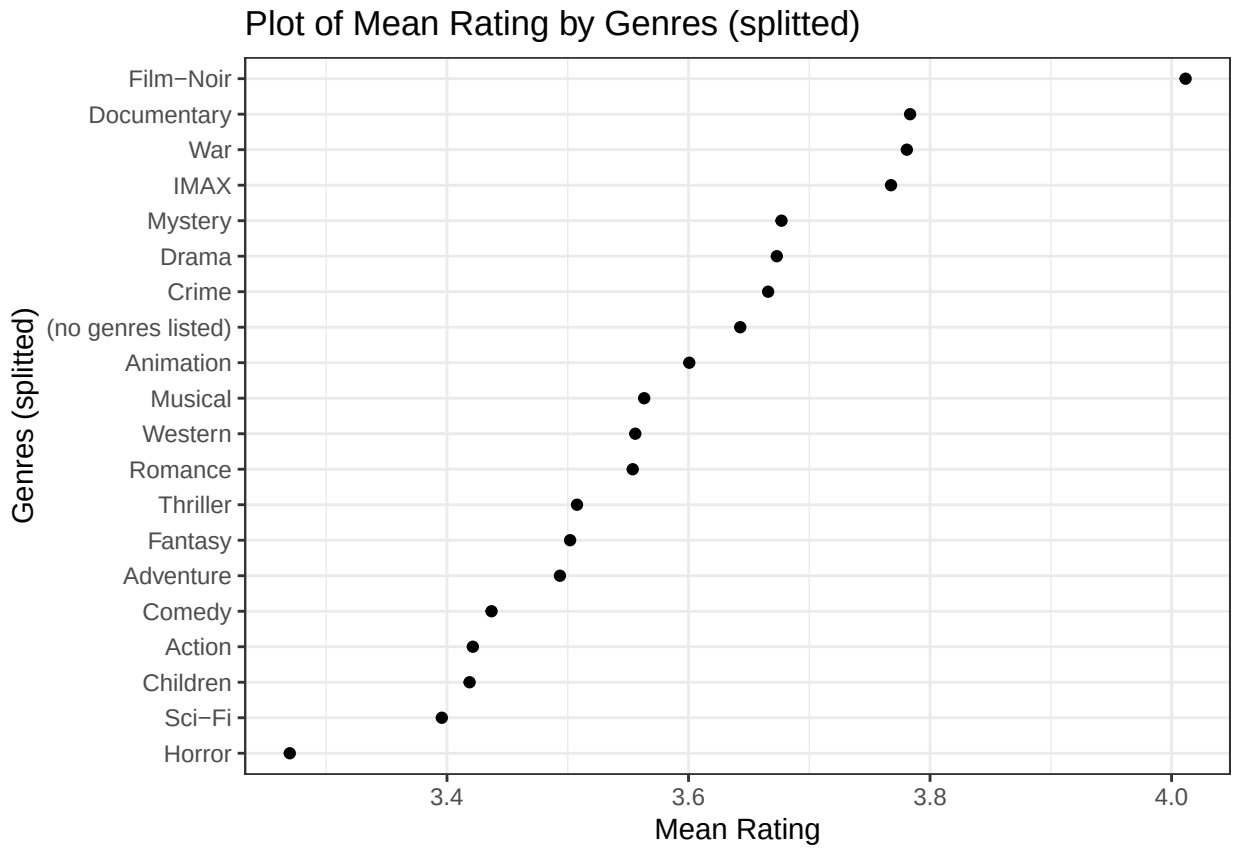
```
## geom_path: Each group consists of only one observation. Do you
## need to adjust the group aesthetic?
```




```
## # A tibble: 20 x 3
##    genres2                n avg_rating
##    <chr>              <int>      <dbl>
##  1 Film-Noir          118541       4.01
##  2 Documentary         93066       3.78
##  3 War                511147       3.78
##  4 IMAX                 8181       3.77
##  5 Mystery            568332       3.68
##  6 Drama             3910127       3.67
##  7 Crime             1327715       3.67
##  8 (no genres listed)       7       3.64
##  9 Animation          467168       3.60
## 10 Musical            433080       3.56
## 11 Western            189394       3.56
## 12 Romance           1712100       3.55
## 13 Thriller          2325899       3.51
## 14 Fantasy            925637       3.50
## 15 Adventure         1908892       3.49
## 16 Comedy            3540930       3.44
## 17 Action            2560545       3.42
## 18 Children           737994       3.42
## 19 Sci-Fi            1341183       3.40
```

```
## 20 Horror              691485       3.27
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
## geom_path: Each group consists of only one observation. Do you
## need to adjust the group aesthetic?
```

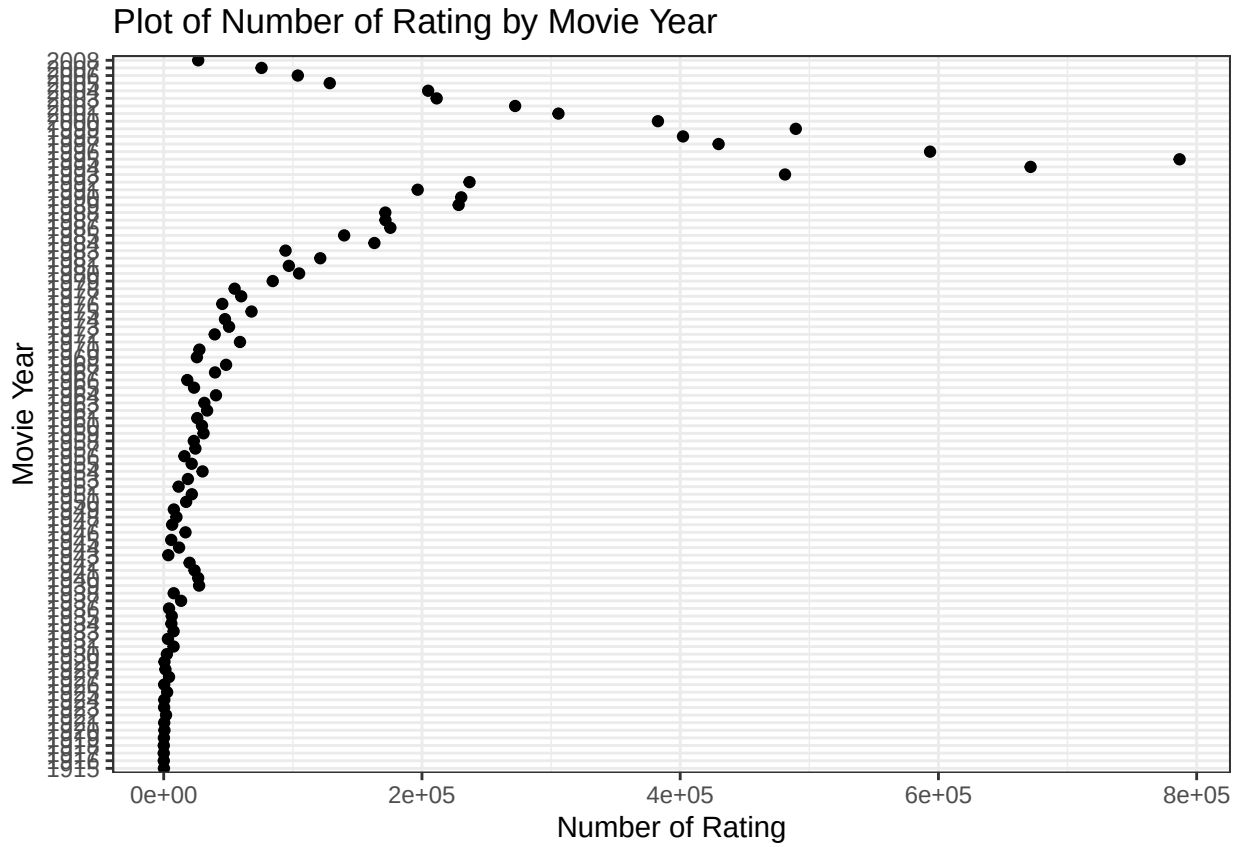


Number of ratings and mean ratings by movie year:

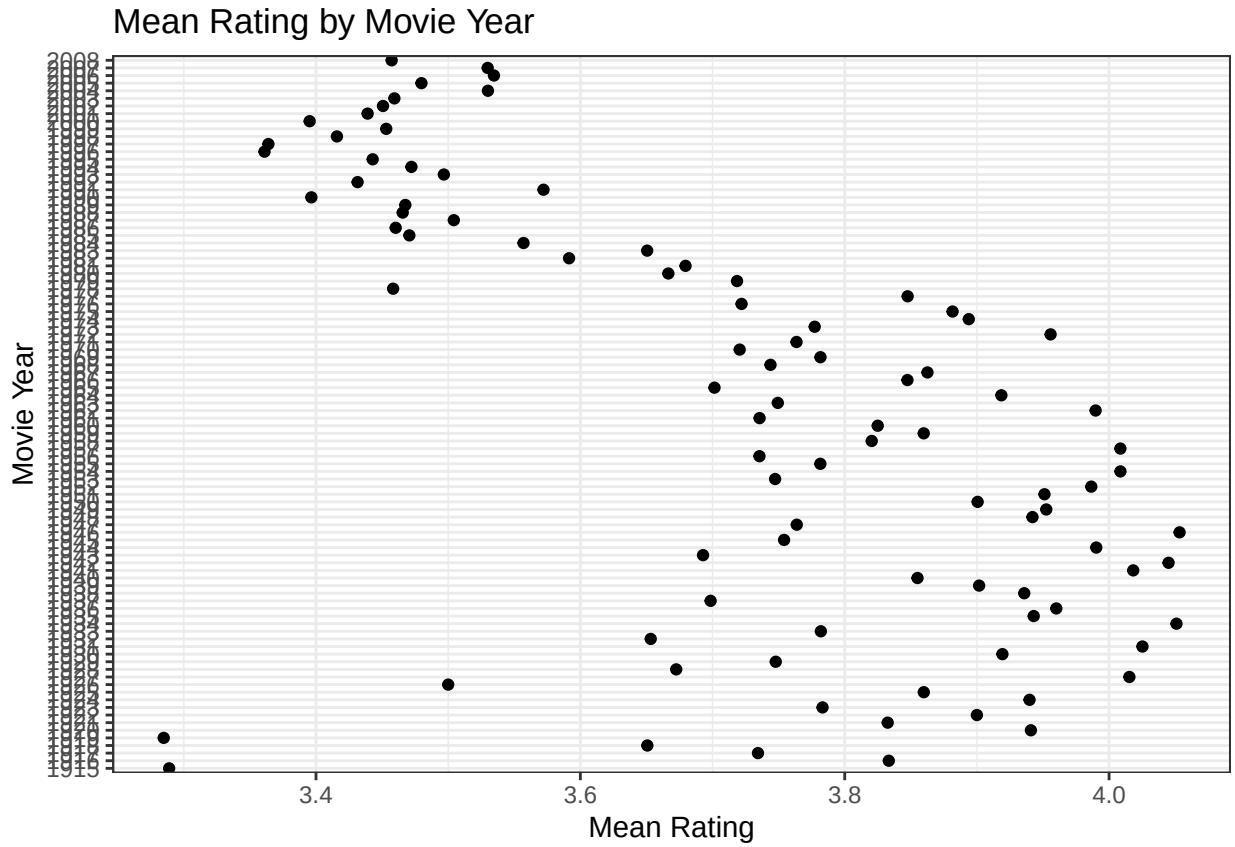
```
## # A tibble: 94 x 3
##    myear     n avg_rating
##  * <chr> <int>      <dbl>
##  1 1915    180       3.29
##  2 1916     84       3.83
##  3 1917     32       3.73
##  4 1918     73       3.65
##  5 1919    158       3.28
##  6 1920    575       3.94
##  7 1921    406       3.83
##  8 1922   1825       3.9
##  9 1923    316       3.78
## 10 1924    457       3.94
## # ... with 84 more rows
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

```
## geom_path: Each group consists of only one observation. Do you
## need to adjust the group aesthetic?
```




```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

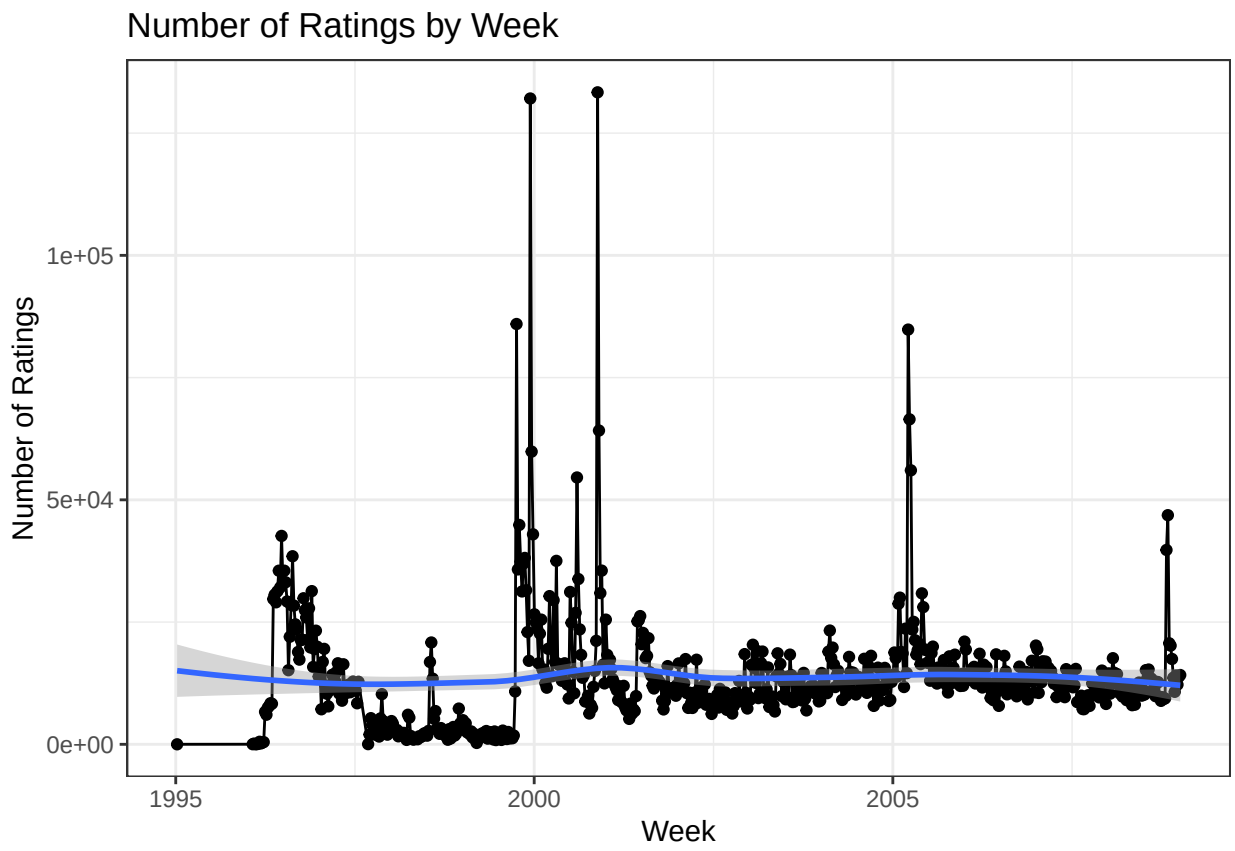


Number of ratings and mean rating over time:

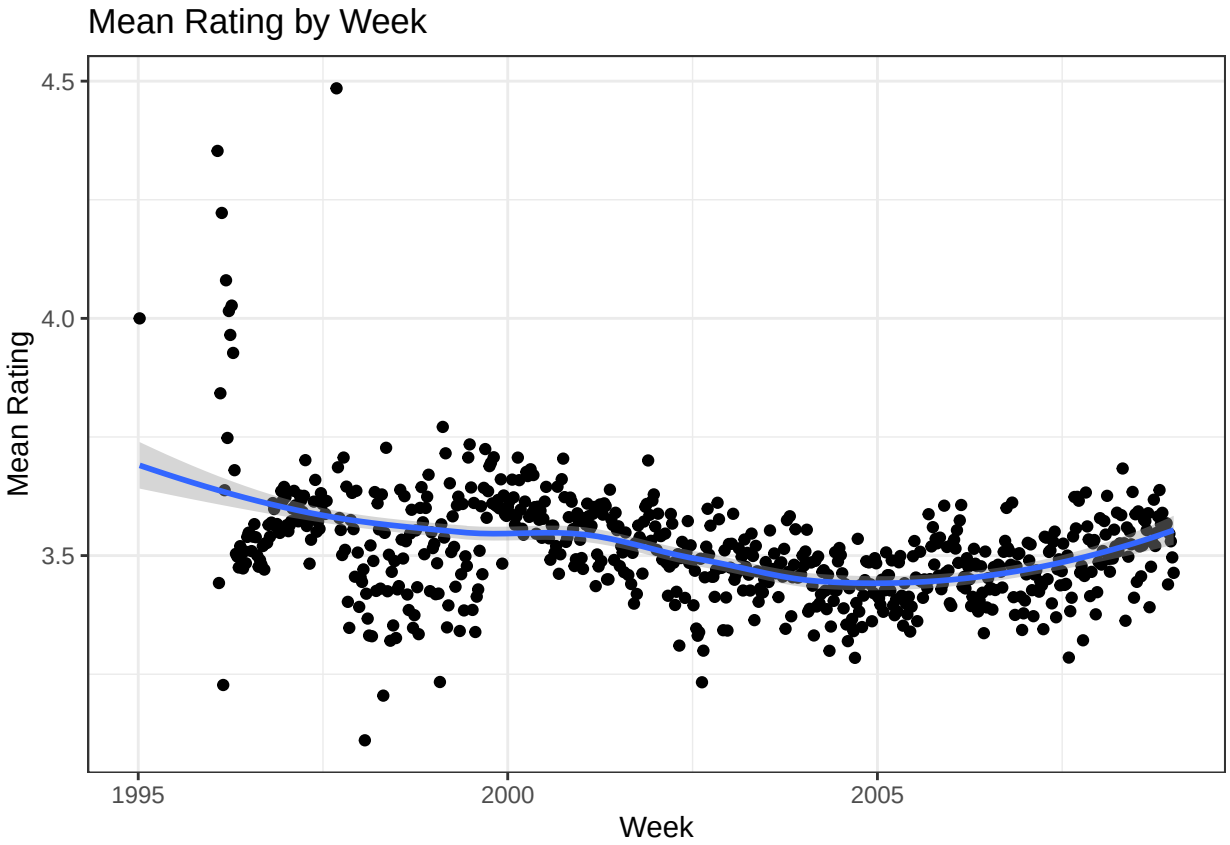
```
## [1] "by week"
```

```
## # A tibble: 671 x 3
##    week                    n avg_rating
##    <dttm>              <int>      <dbl>
##  1 2000-11-19 00:00:00 133343       3.56
##  2 1999-12-12 00:00:00 132084       3.63
##  3 1999-10-03 00:00:00  85961       3.69
##  4 2005-03-20 00:00:00  84823       3.40
##  5 2005-03-27 00:00:00  66471       3.37
##  6 2000-11-26 00:00:00  64175       3.48
##  7 1999-12-19 00:00:00  59862       3.59
##  8 2005-04-03 00:00:00  56038       3.41
##  9 2000-08-06 00:00:00  54575       3.56
## 10 2008-11-02 00:00:00  46863       3.57
## # ... with 661 more rows
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



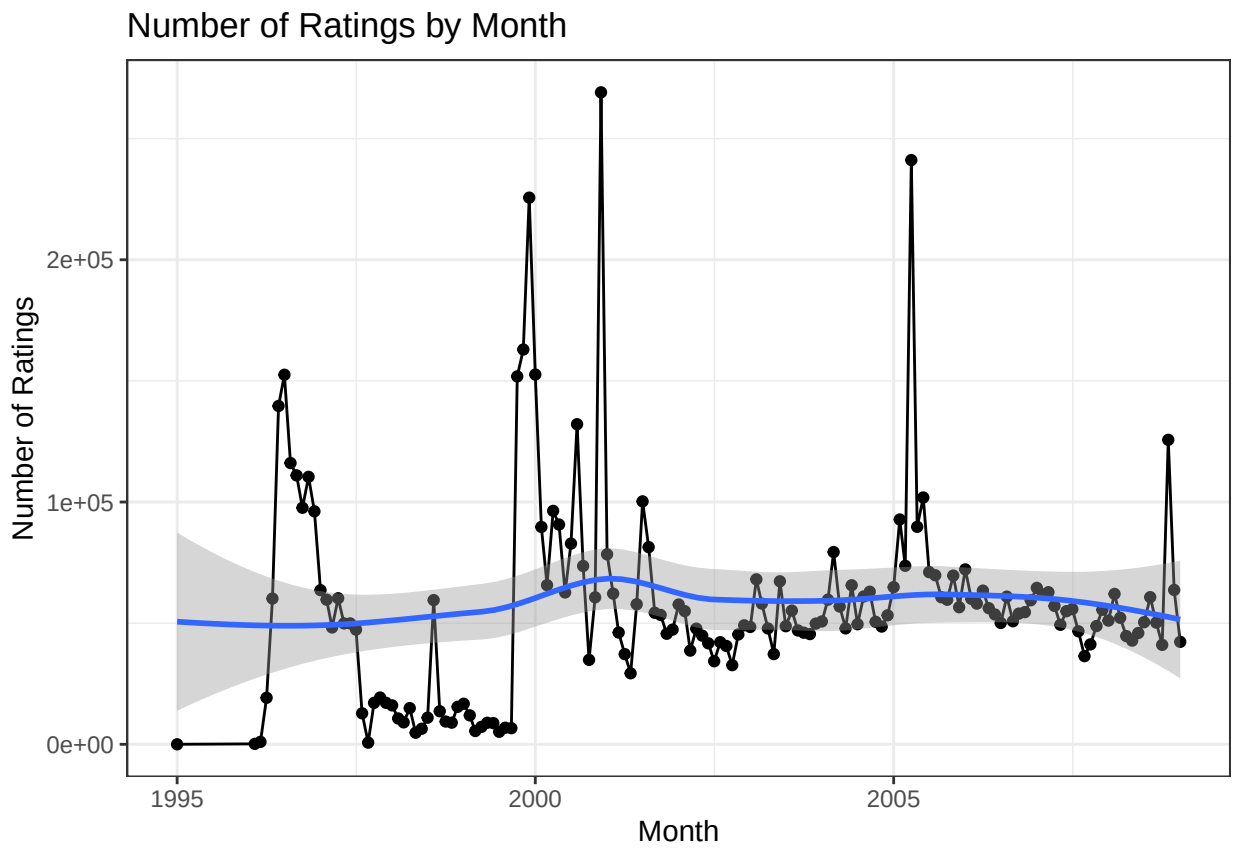

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



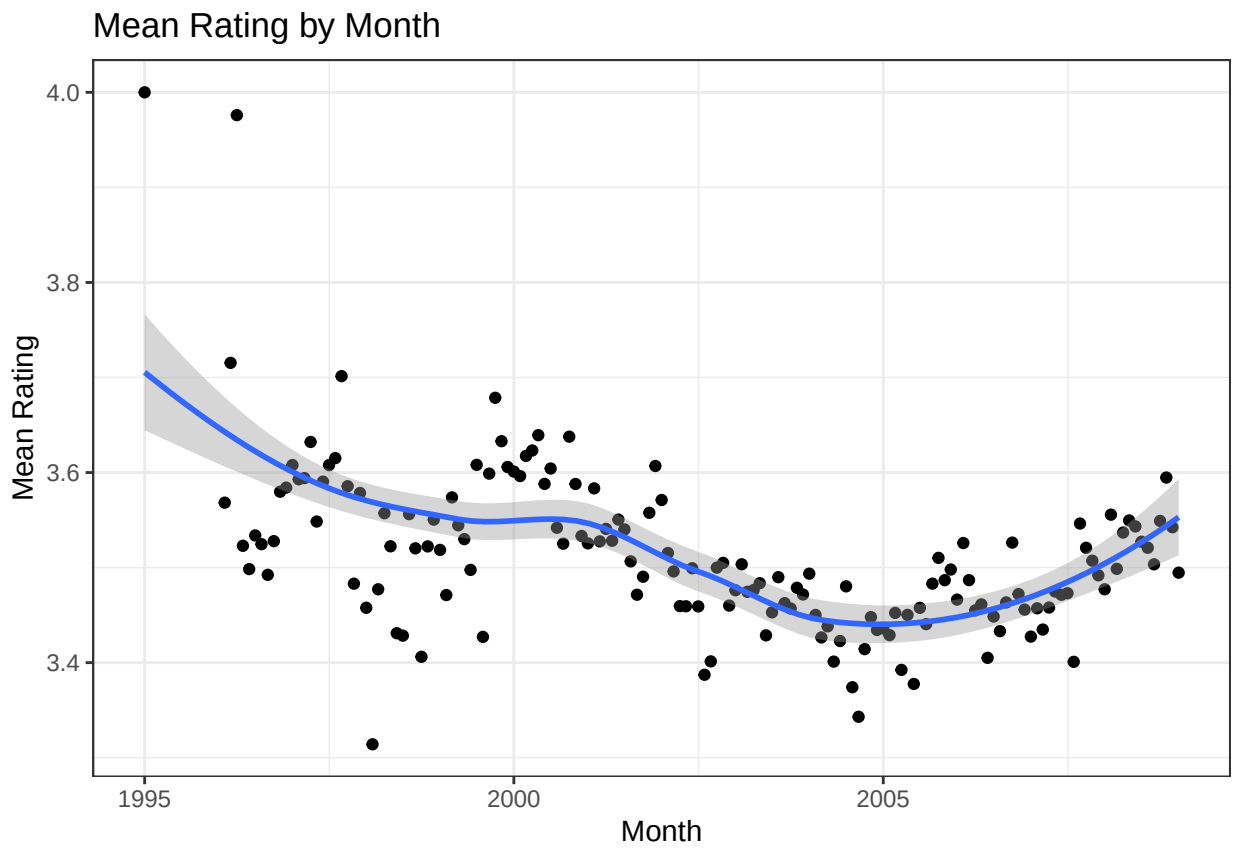

```
## [1] "by month"
```

```
## # A tibble: 157 x 3
##    month                   n avg_rating
##    <dttm>              <int>      <dbl>
##  1 2000-12-01 00:00:00 269102       3.53
##  2 2005-04-01 00:00:00 241132       3.39
##  3 1999-12-01 00:00:00 225639       3.61
##  4 1999-11-01 00:00:00 162954       3.63
##  5 2000-01-01 00:00:00 152618       3.60
##  6 1996-07-01 00:00:00 152549       3.53
##  7 1999-10-01 00:00:00 151851       3.68
##  8 1996-06-01 00:00:00 139644       3.50
##  9 2000-08-01 00:00:00 132110       3.54
## 10 2008-11-01 00:00:00 125670       3.59
## # ... with 147 more rows
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



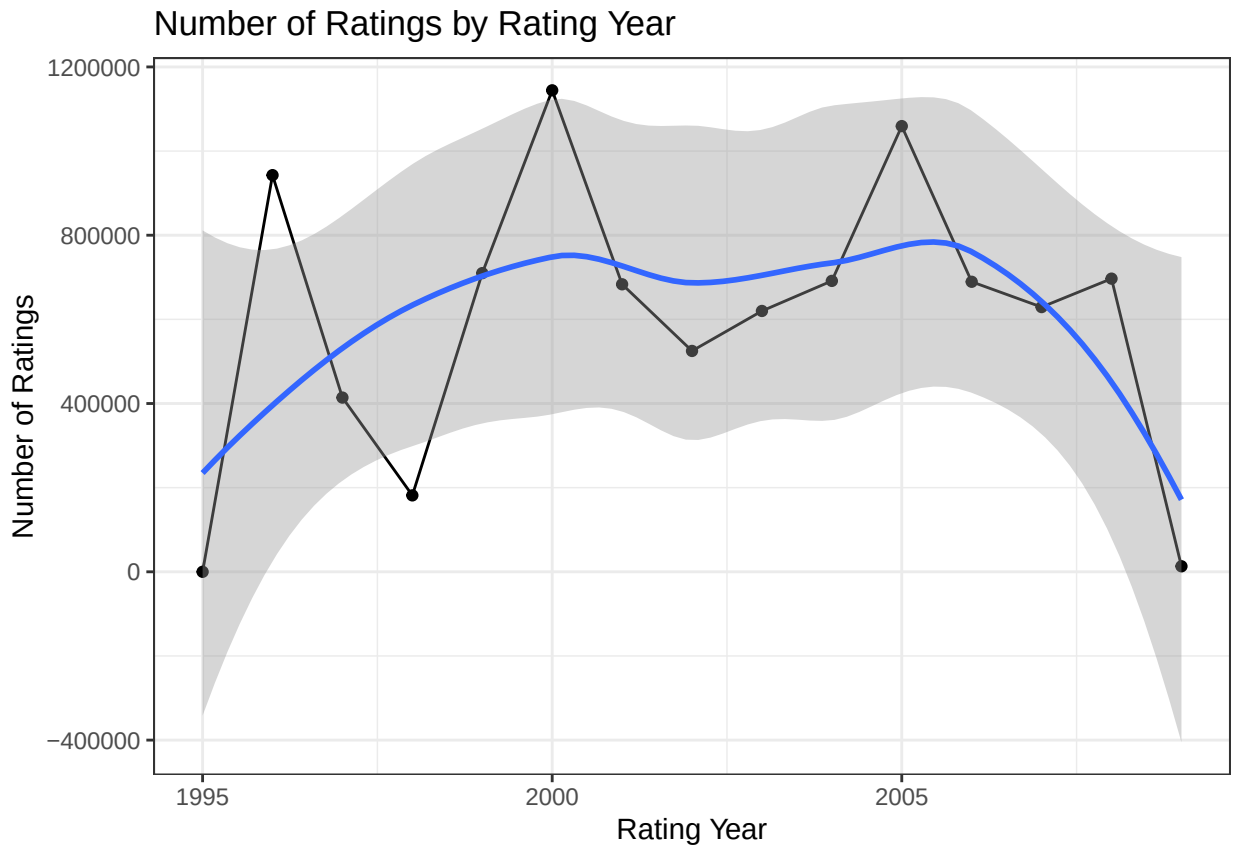

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



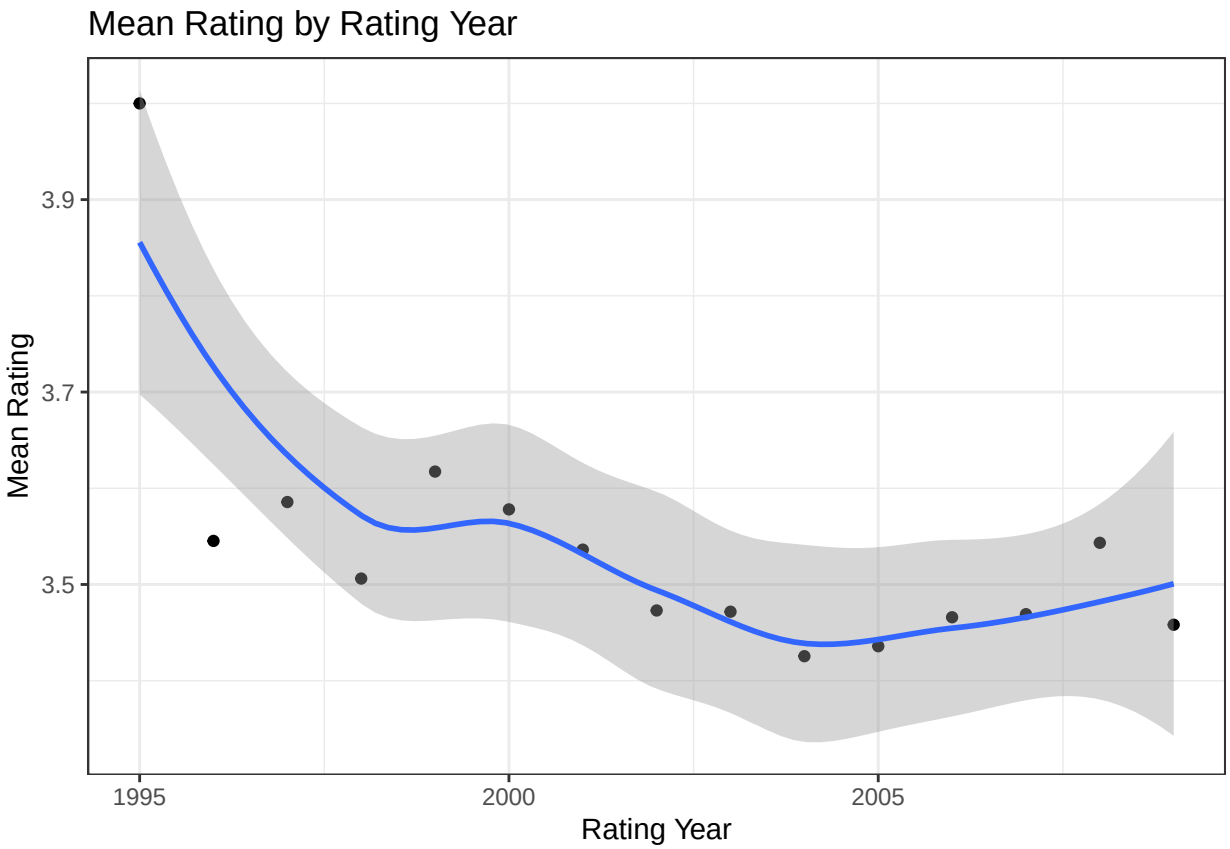

```
## [1] "by Year"
```

```
## # A tibble: 15 x 3
##    ryear       n avg_rating
##    <dbl>   <int>      <dbl>
##  1  2000 1144349       3.58
##  2  2005 1059277       3.44
##  3  1996  942772       3.55
##  4  1999  709893       3.62
##  5  2008  696740       3.54
##  6  2004  691429       3.43
##  7  2006  689315       3.47
##  8  2001  683355       3.54
##  9  2007  629168       3.47
## 10  2003  619938       3.47
## 11  2002  524959       3.47
## 12  1997  414101       3.59
## 13  1998  181634       3.51
## 14  2009   13123       3.46
## 15  1995       2       4
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

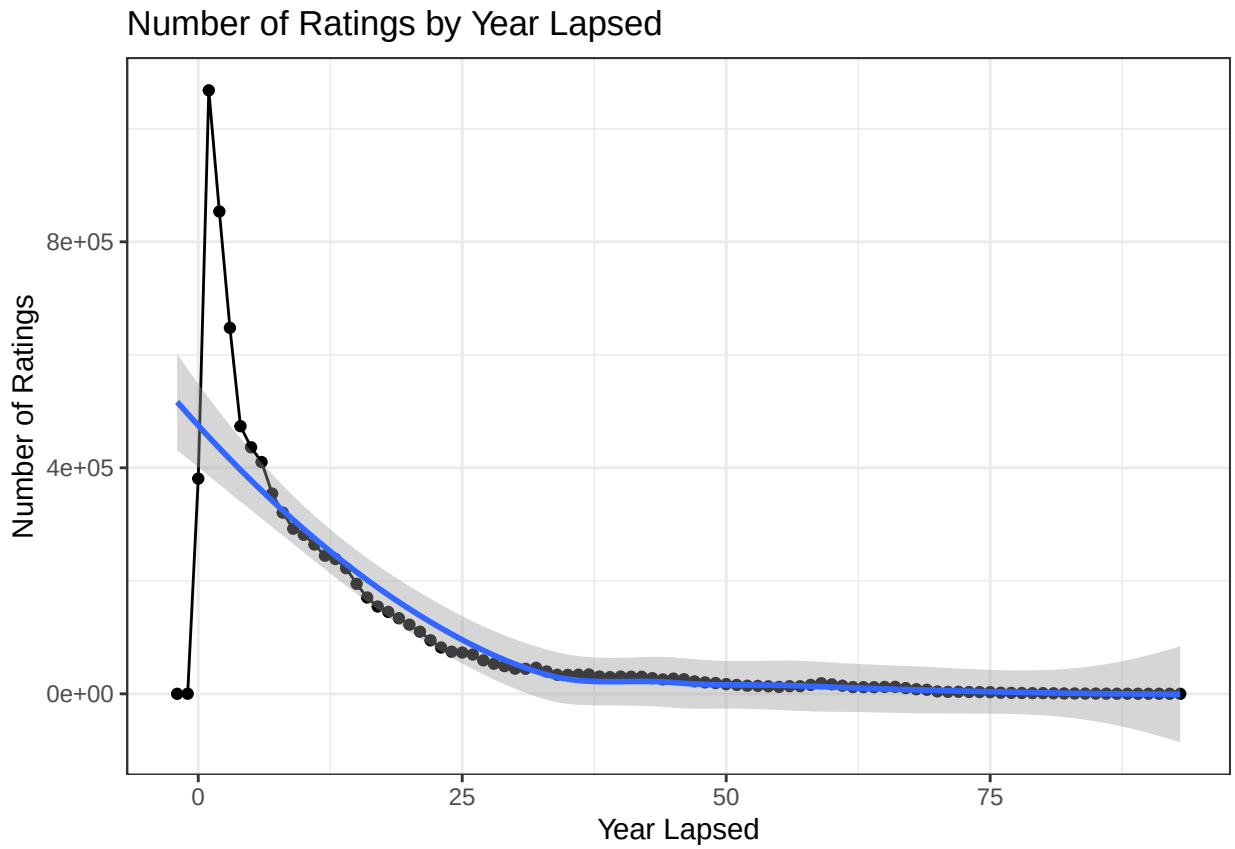


Number of ratings and mean rating by year lapsed:

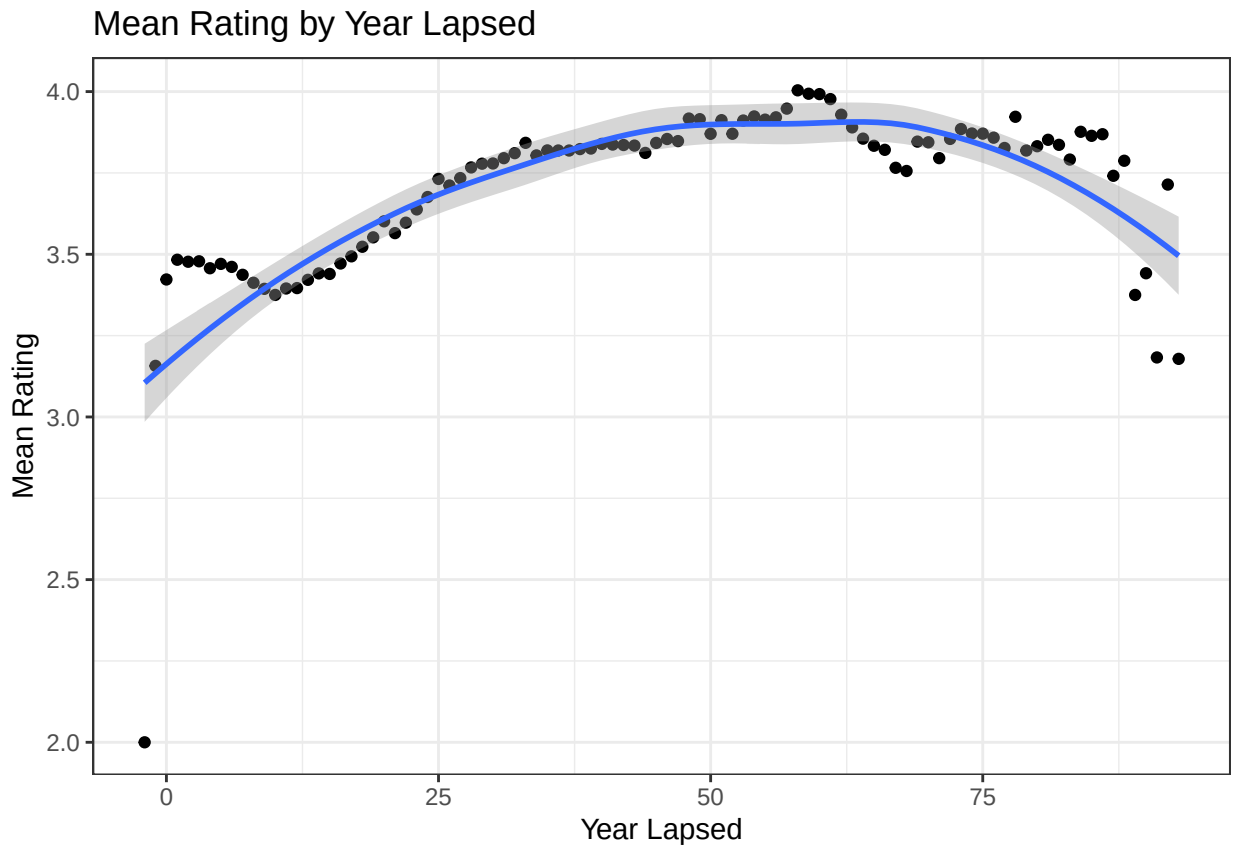
```
## [1] "by yearlapsed"
```

```
## # A tibble: 96 x 3
##    yearlapsed       n avg_rating
##         <dbl>   <int>      <dbl>
##  1          1 1068070       3.48
##  2          2  853680       3.48
##  3          3  647650       3.48
##  4          4  473660       3.46
##  5          5  436378       3.47
##  6          6  410159       3.46
##  7          0  380915       3.42
##  8          7  354368       3.44
##  9          8  320713       3.41
## 10          9  292203       3.39
## # ... with 86 more rows
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



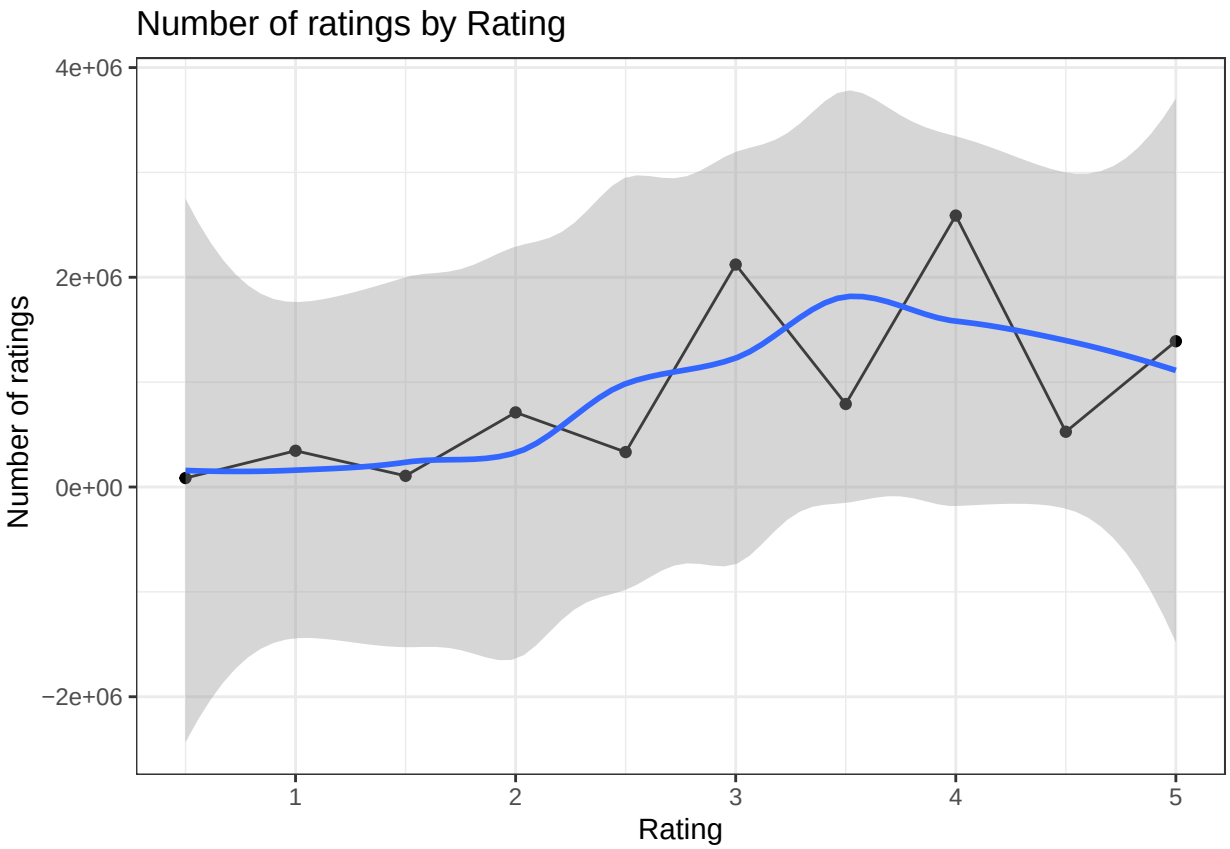

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



Number of ratings by rating:

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

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



RMSE using different models

Root Mean Square Error (RMSE) is the standard deviation of the residuals (prediction errors). Residuals are a measure of how far from the regression line data points are; RMSE is a measure of how spread out these residuals are. In other words, it tells you how concentrated the data is around the line of best fit.

Mean rating of Edx dataset :

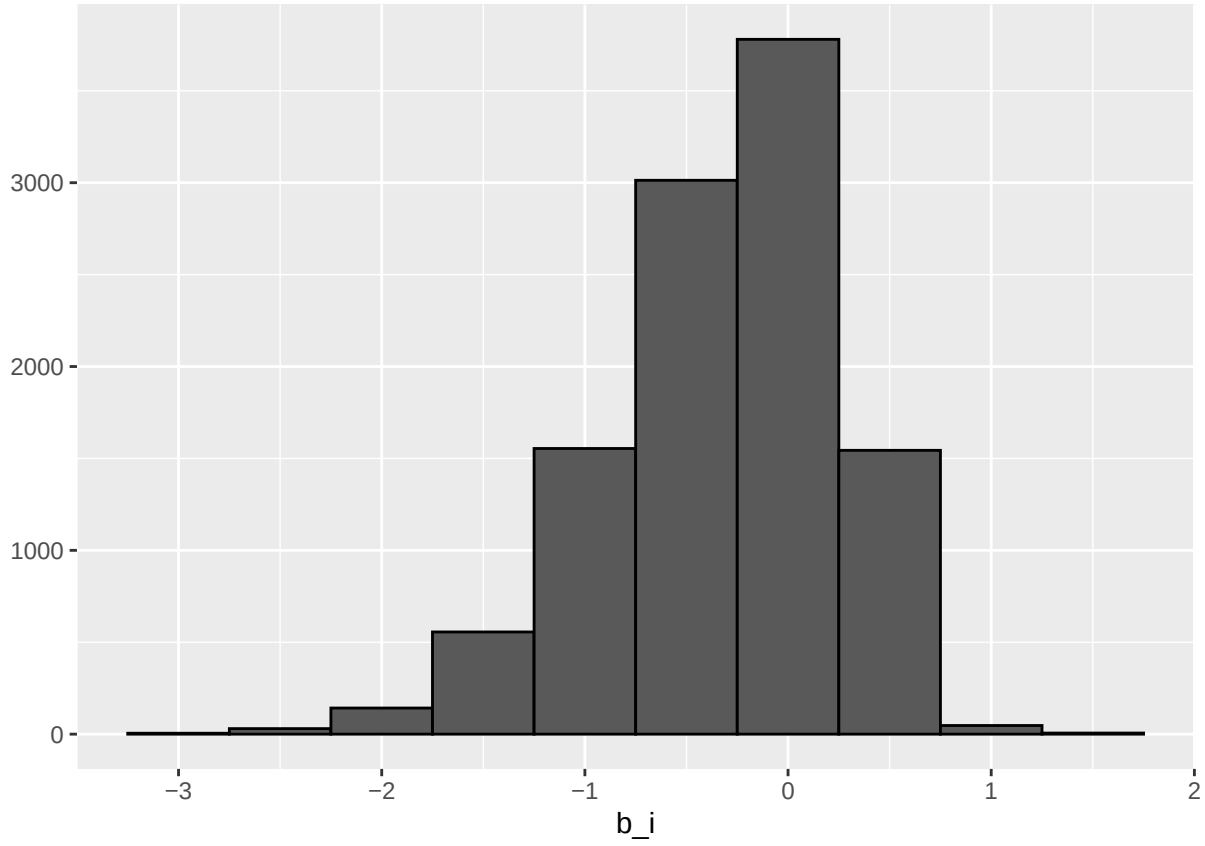
```
## [1] 3.512465
```

##Predict using :

1. Mean rating

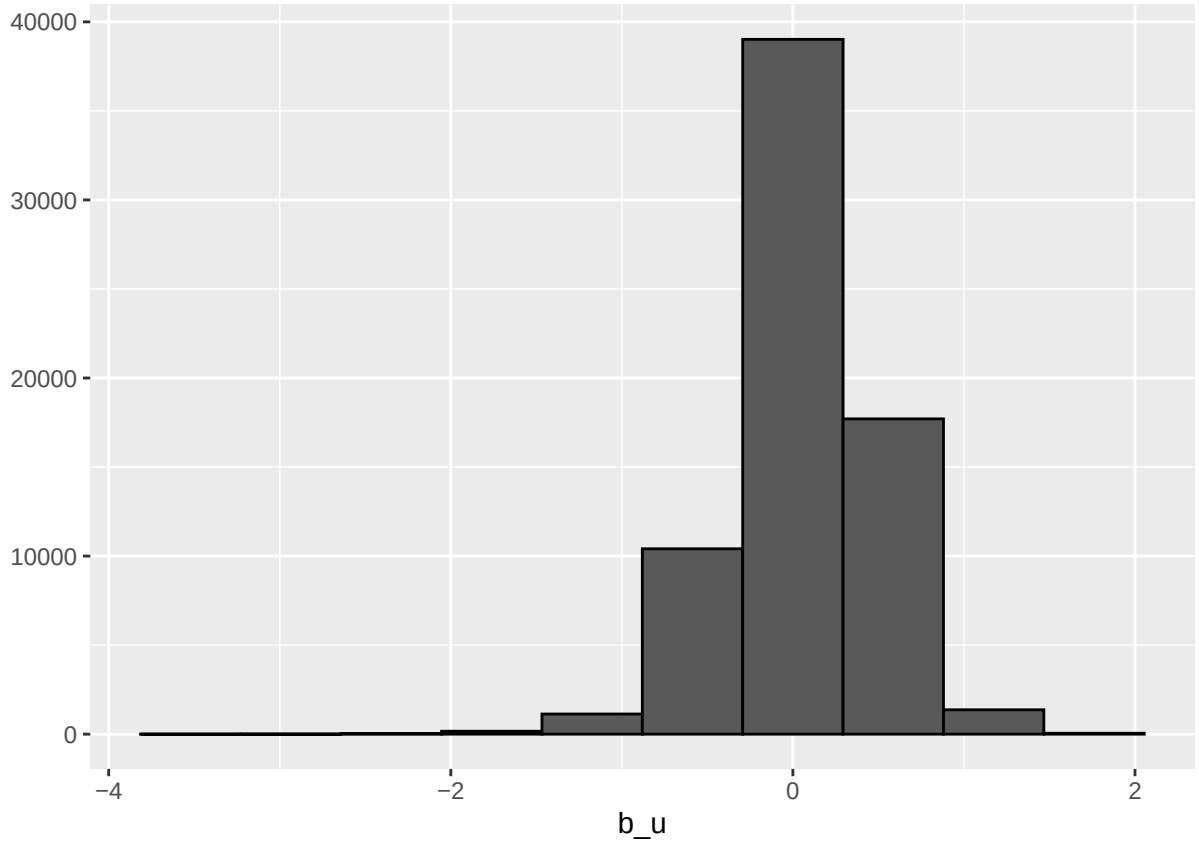
method	RMSE
Using Mean Rating	1.061202

2. Mean rating and movie effect :



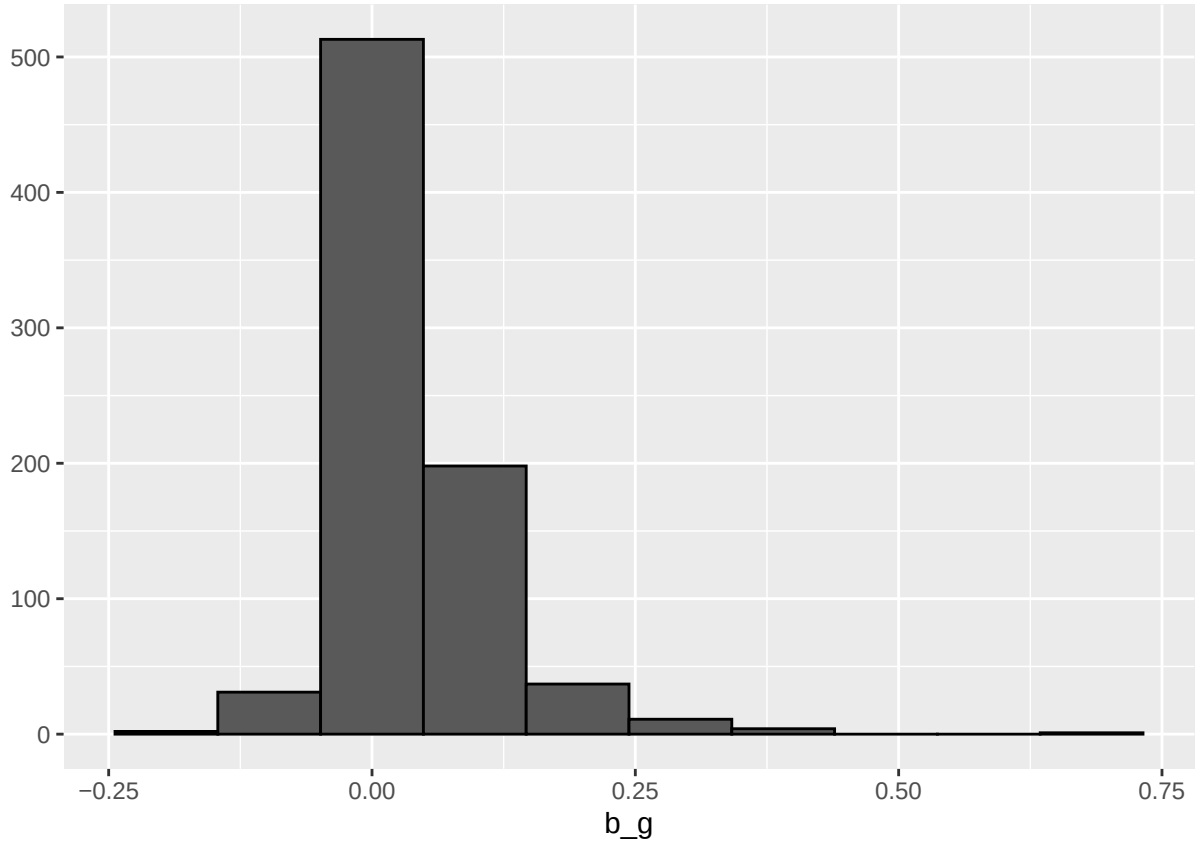
method	RMSE
Using Mean Rating	1.0612018
Movie Effect Model	0.9439087

3. User effect :



method	RMSE
Using Mean Rating	1.0612018
Movie Effect Model	0.9439087
Movie + User Effects Model	0.8653488

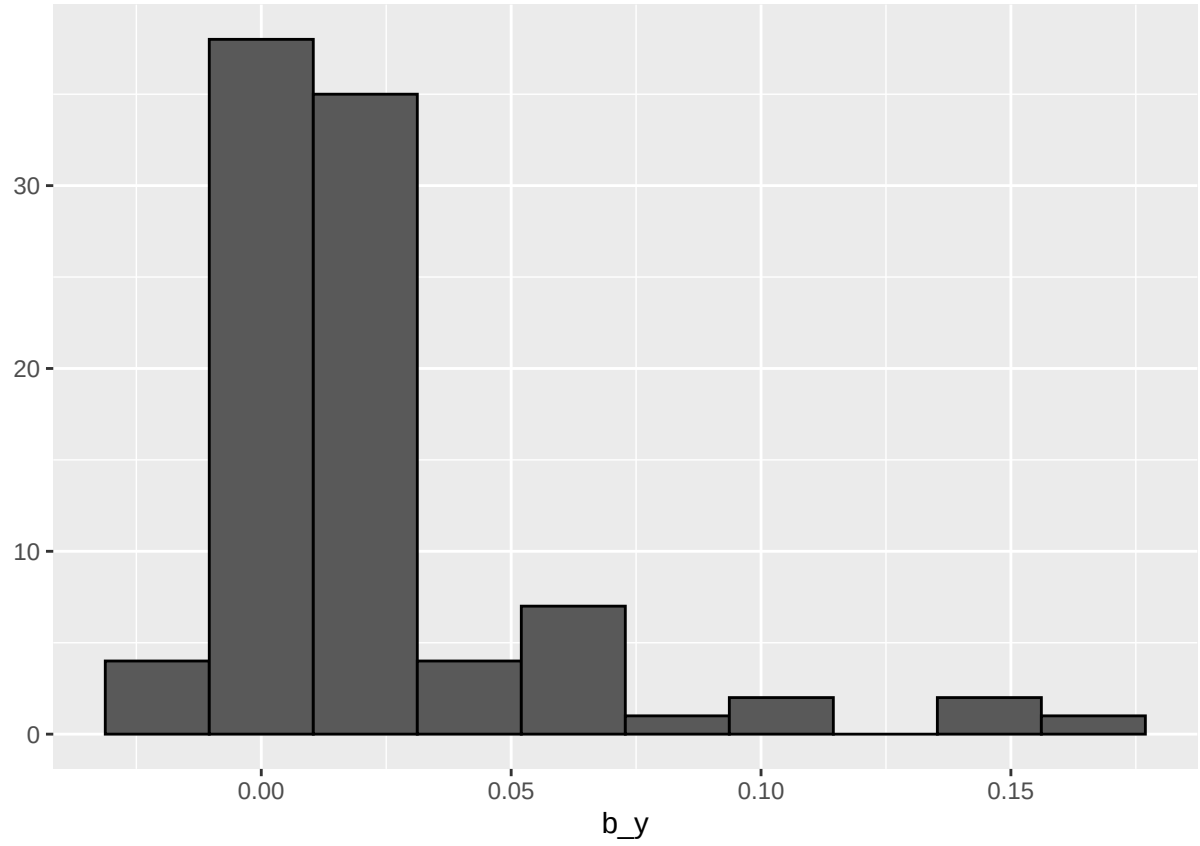
4. Genre effect



method	RMSE
Using Mean Rating	1.0612018
Movie Effect Model	0.9439087
Movie + User Effects Model	0.8653488
Movie + User + Genre Effects Model	0.8649469

5. movie year

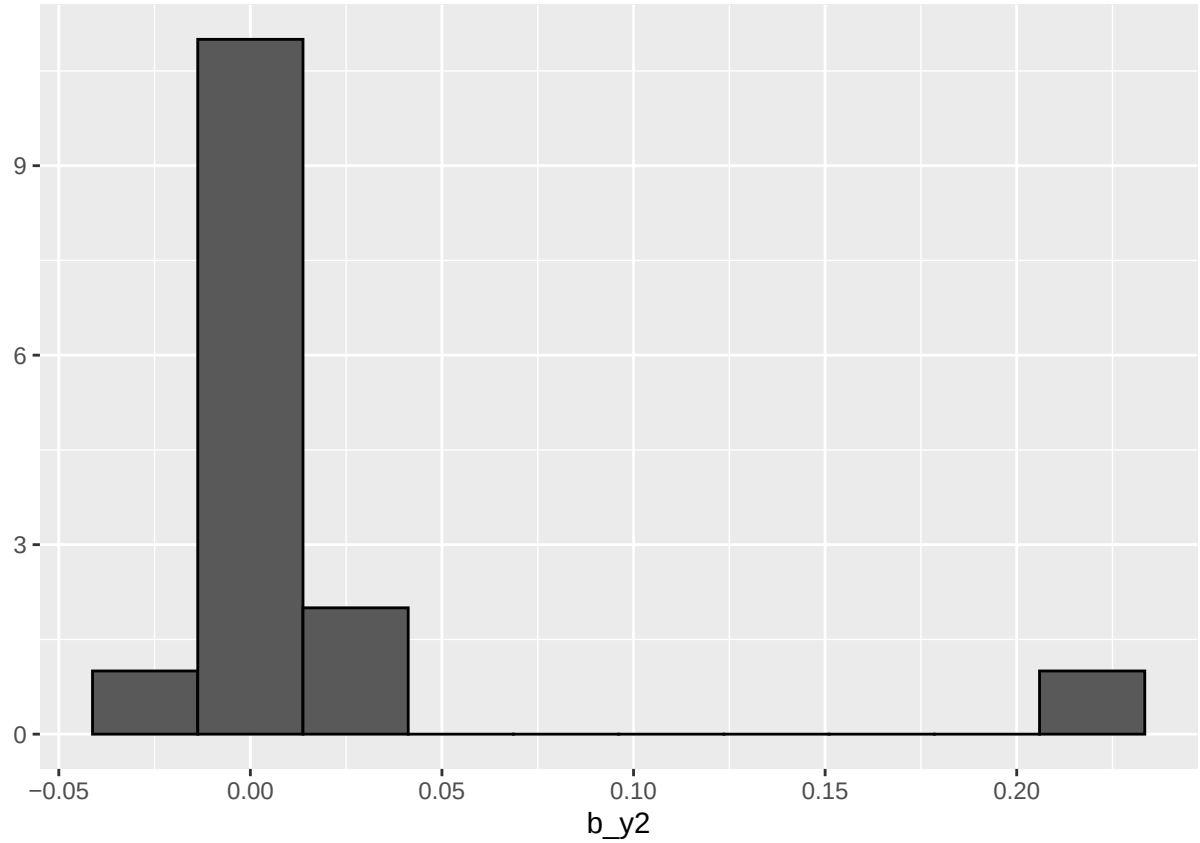
```
## [1] "movie year effect"
```



method	RMSE
Using Mean Rating	1.0612018
Movie Effect Model	0.9439087
Movie + User Effects Model	0.8653488
Movie + User + Genre Effects Model	0.8649469
Movie + User + Genre + Movie year Effects Model	0.8647606

6. rating year rating year average rating

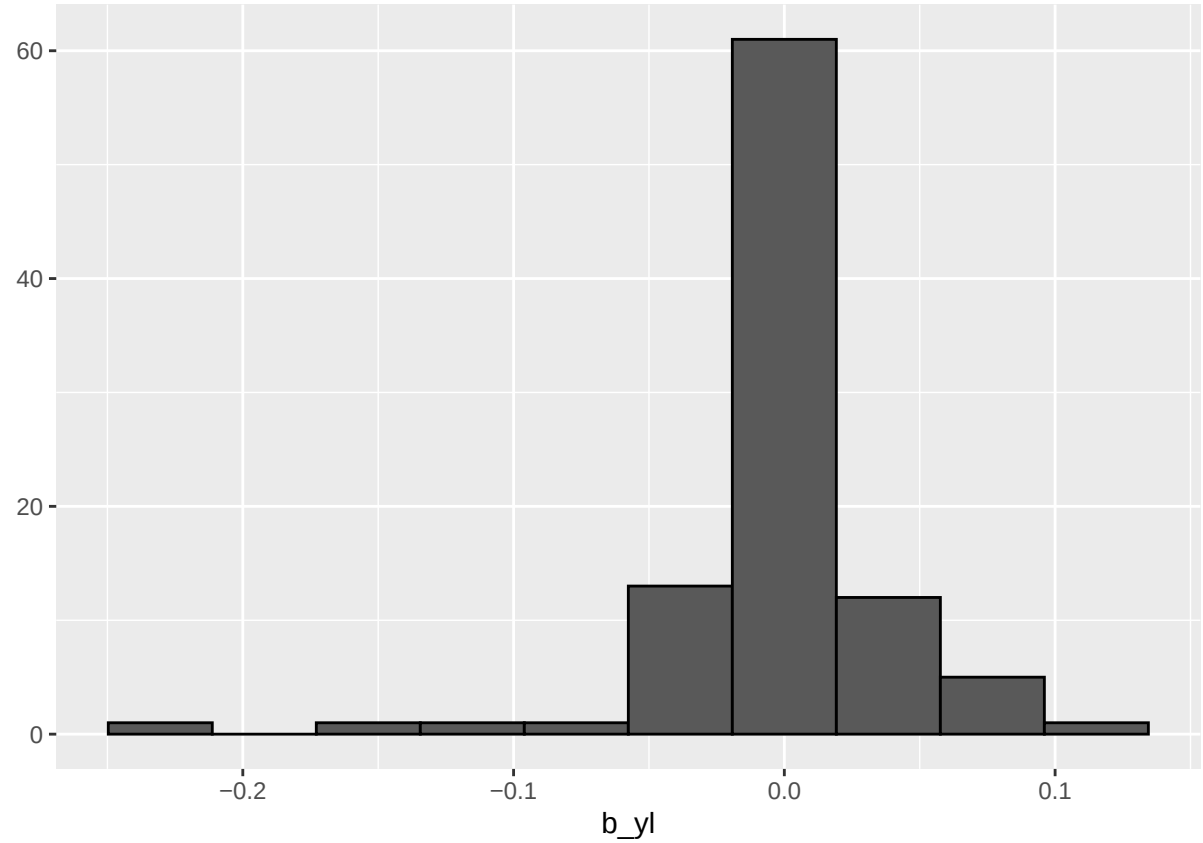
```
## [1] "rating year effect"
```



method	RMSE
Using Mean Rating	1.0612018
Movie Effect Model	0.9439087
Movie + User Effects Model	0.8653488
Movie + User + Genre Effects Model	0.8649469
Movie + User + Genre + Movie year Effects Model	0.8647606
Movie + User + Genre + Movie year + Rating year Effects Model	0.8646655

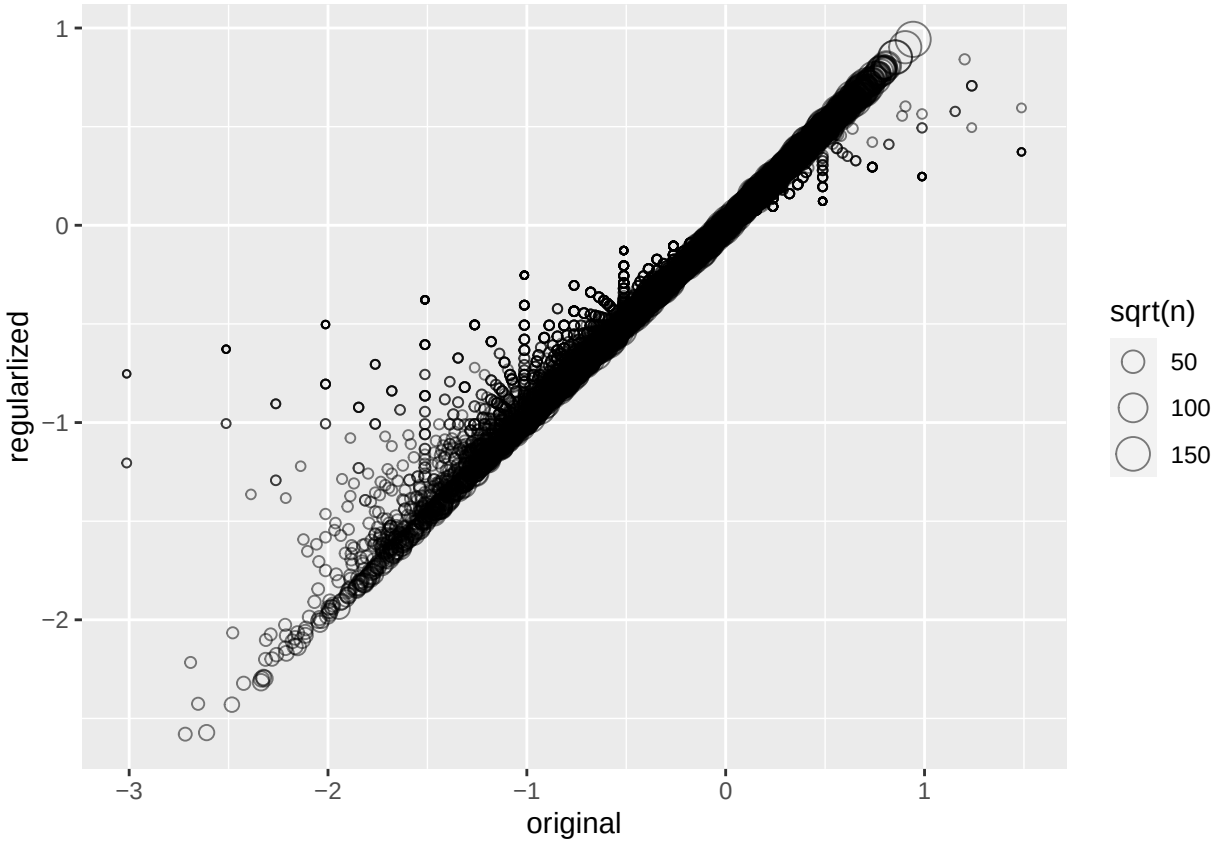
7. Yearlapsed effect

```
## [1] "Yearlapsed  effect"
```



method	RMSE
Using Mean Rating	1.0612018
Movie Effect Model	0.9439087
Movie + User Effects Model	0.8653488
Movie + User + Genre Effects Model	0.8649469
Movie + User + Genre + Movie year Effects Model	0.8647606
Movie + User + Genre + Movie year + Rating year Effects Model	0.8646655
Movie + User + Genres + Movie year + Rating year + Year lapsed Effects Model	0.8644061

8. Regularized Movie Effect




```
## Joining, by = "movieId"
```

b_i	n
0.9425650	28015
0.9027482	17747
0.8532702	21648
0.8509180	23193
0.8412744	7
0.8077428	11232
0.8058817	7935
0.8025903	2922
0.7981535	2967
0.7972415	2154

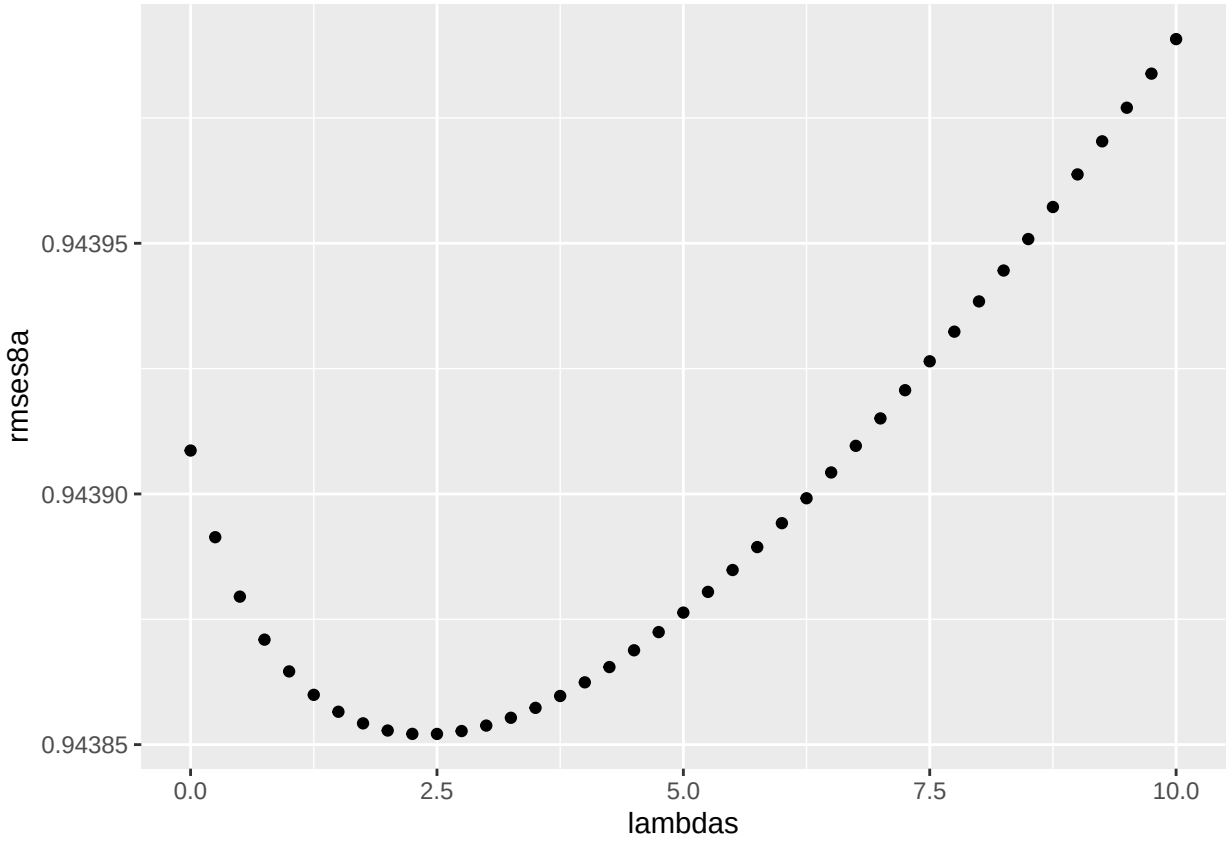
```
## Joining, by = "movieId"
```

b_i	n
-2.579628	5
-2.571686	17
-2.430055	19
-2.425683	8
-2.321798	11
-2.316449	41
-2.300088	31
-2.297157	39

b_i	n
-2.291909	26
-2.200377	6

method	RMSE
Using Mean Rating	1.0612018
Regularized Movie Effect Model	0.9438538

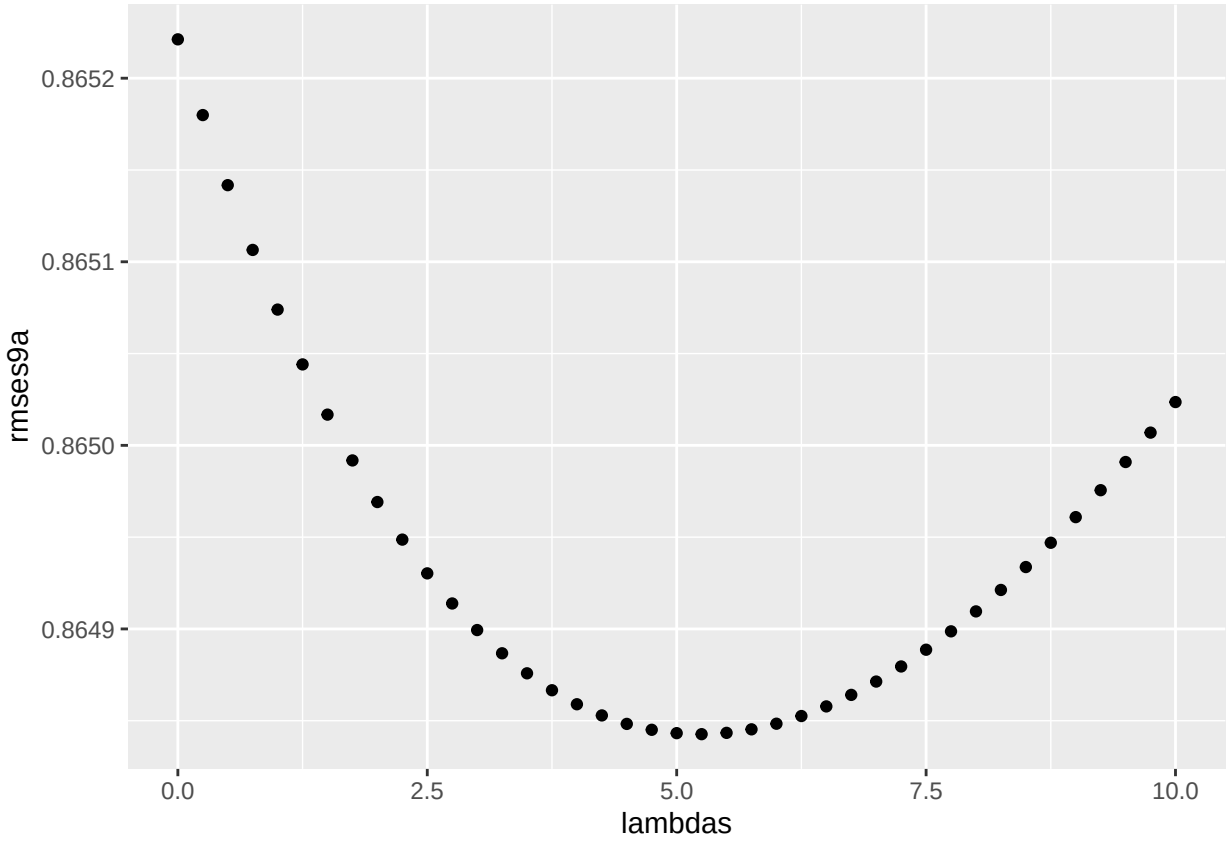
9. optimise lamdas for movie effect




```
## [1] 2.5
```

method	RMSE
Using Mean Rating	1.0612018
Regularized Movie Effect Model	0.9438521

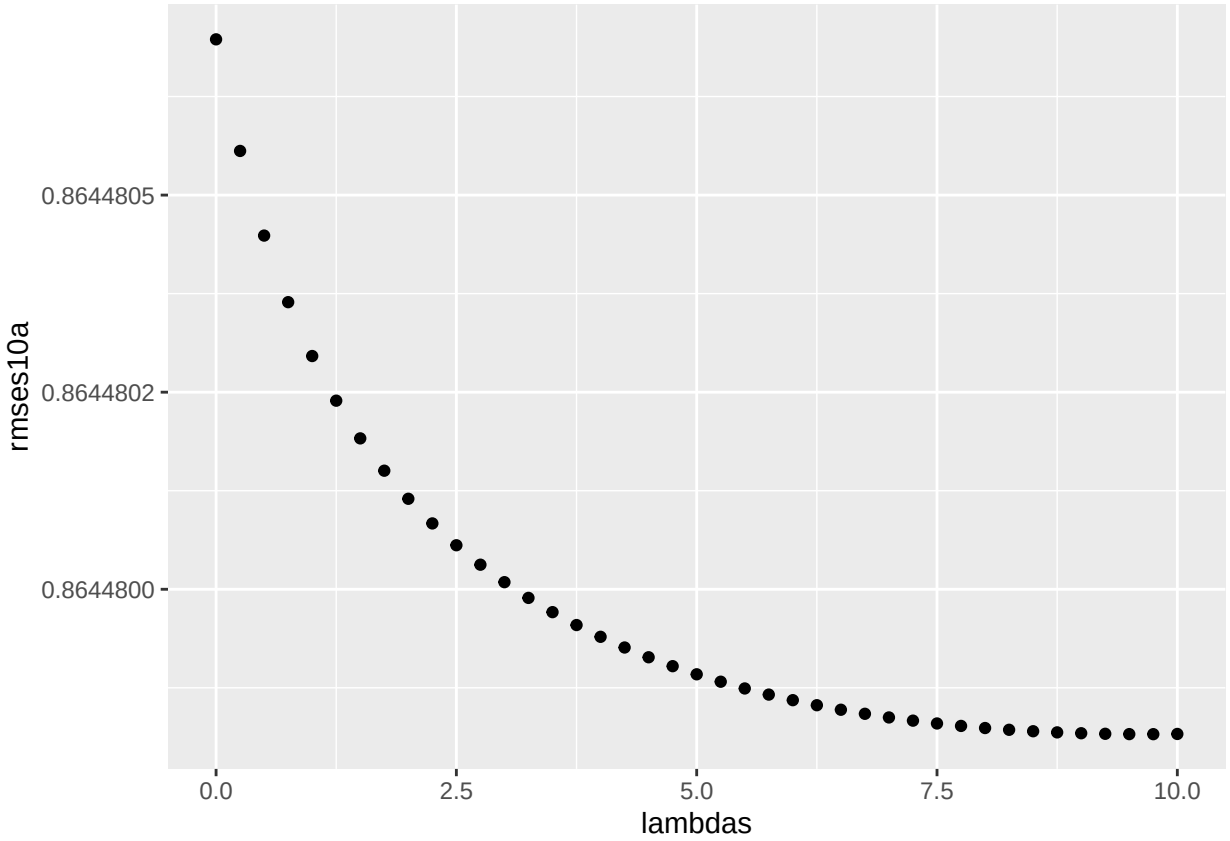
10. optimise lambdas for user effect




```
## [1] 5.25
```

method	RMSE
Using Mean Rating	1.0612018
Regularized Movie Effect Model	0.9438521
Regularized User Effect Model	0.8648427

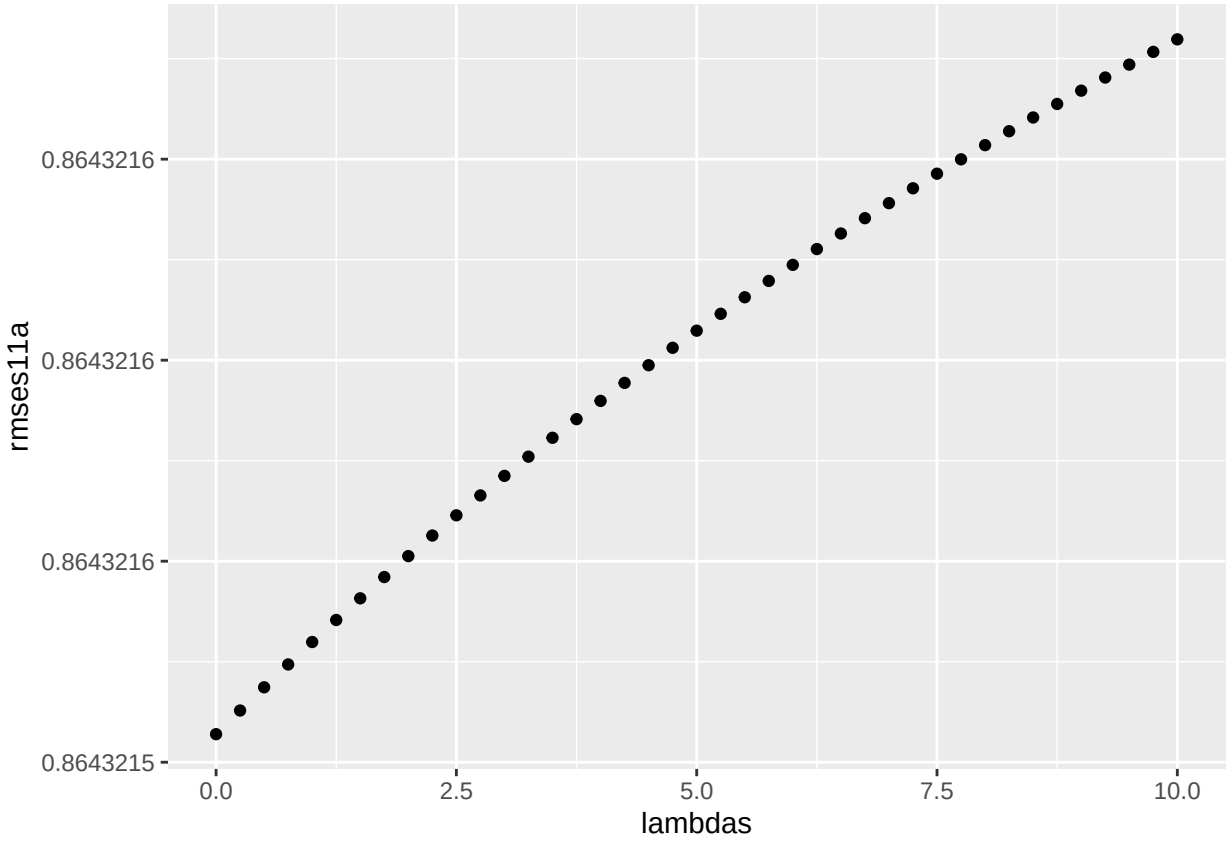
11. optimise lambdas for genres effect




```
## [1] 9.75
```

method	RMSE
Using Mean Rating	1.0612018
Regularized Movie Effect Model	0.9438521
Regularized User Effect Model	0.8648427
Regularized Genres Effect Model	0.8644798

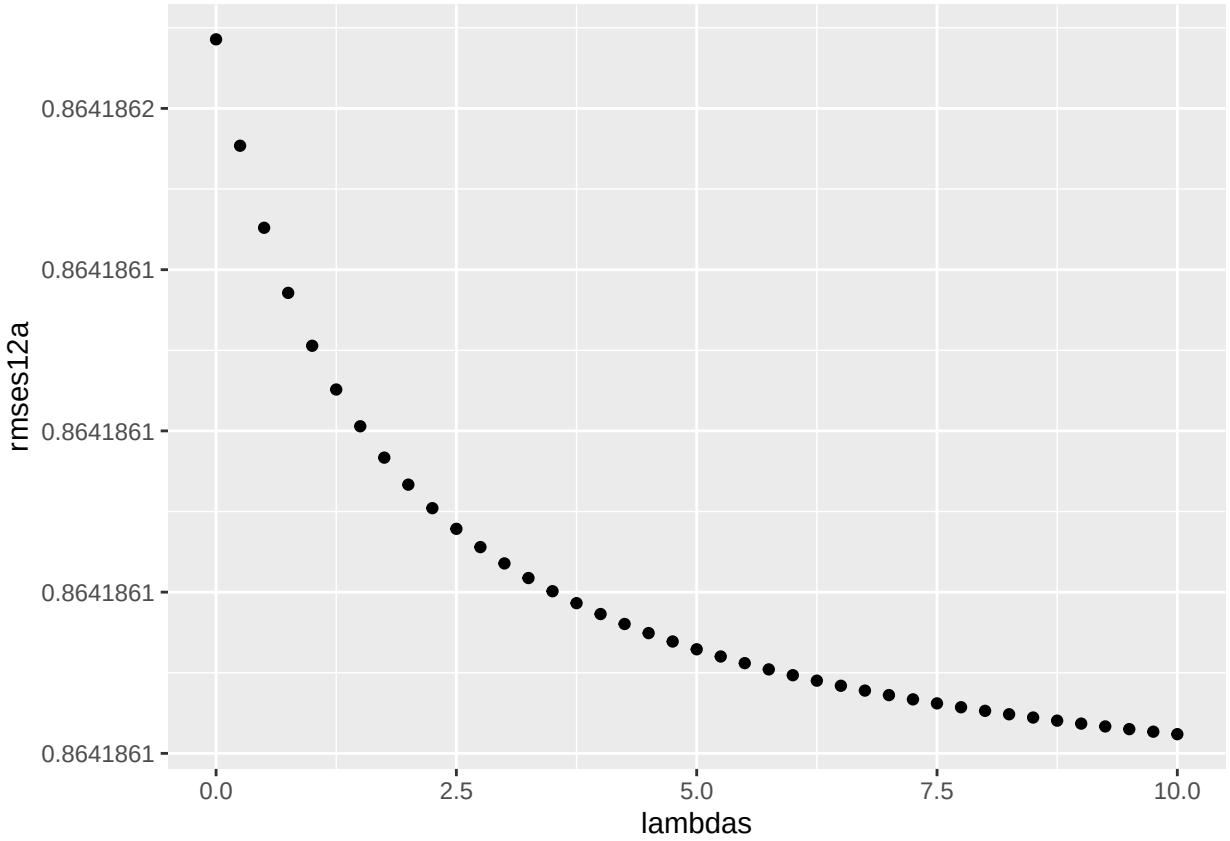
12. optimise lamdas for movie year effect




```
## [1] 0
```

method	RMSE
Using Mean Rating	1.0612018
Regularized Movie Effect Model	0.9438521
Regularized User Effect Model	0.8648427
Regularized Genres Effect Model	0.8644798
Regularized Movie Year Effect Model	0.8643215

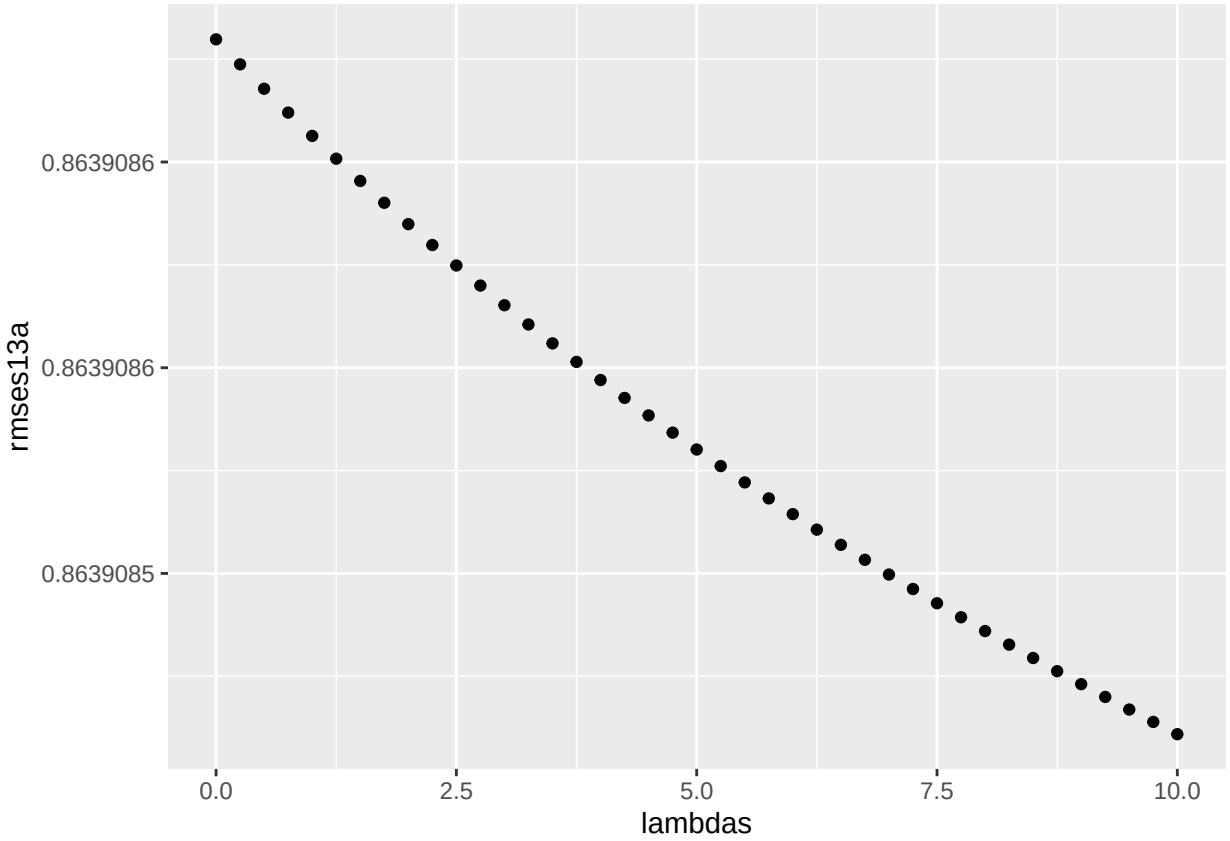
13. optimise lamdas for rating year effect




```
## [1] 10
```

method	RMSE
Using Mean Rating	1.0612018
Regularized Movie Effect Model	0.9438521
Regularized User Effect Model	0.8648427
Regularized Genres Effect Model	0.8644798
Regularized Movie Year Effect Model	0.8643215
Regularized Rating Year Effect Model	0.8641861

14. optimise lamdas for year lapsed effect




```
## [1] 10
```

method	RMSE
Using Mean Rating	1.0612018
Regularized Movie Effect Model	0.9438521
Regularized User Effect Model	0.8648427
Regularized Genres Effect Model	0.8644798
Regularized Movie Year Effect Model	0.8643215
Regularized Rating Year Effect Model	0.8641861
Regularized Year lapsd Effect Model	0.8639085

End of report