Movielens

Michele Pesce

5 Maggio 2019

Overview

The goal of this project is prediction ratings of a recommender dataset whose name is **Movielens**. This is a regression problem since the *rating* label is a numeric continuos variable, so that the target is to minimize RMSE. The dataset consistS of ten million records, so it's very hard for desktop hardware systems to execute ML algorithms without waiting too long or crashing, then only a particular modeling approach has been possible using the whole data. This huge dataset would be better computed by a very powerful system or parallelized in clusters. The dataframe **Movielens** has been split in **edx** as train set and **validation** as validation set. This document reports the following sections:

- **Data cleaning** In this section, the data integrity is explored, and possibly executed all the operations to make the dataset assessable for further analisys
- **Data exploration** Variables in the dataset are explored and checked in order to choose the relevant factors that could be used as predictors for the target labels.
- **The modeling approach** ML algorithms that best fit the kind of problem are chosen to predict the outcome, on the basis of the predictors assesed on the data exploration section. Usually, more than an ML is checked and tuned and chosen the one that minimize rmse or maximize accuracy.
- **Results** The ML outcomes are compared, and evaluated pros and cons of the different approaches.
- **Conclusions** To summary the operations, and take in consideration further different approaches that could improve the project and the results

Data Cleaning

The first operation on data is to check the presence of inconsistent values that could affect the correctness of analysis. Generally, if we don't take in count incoherent values, mean and sd could be affected by bias, all the further analisys would be affected by some kind of error. Using the R summary tools we can see if there are NA's on the movielens dataframe (10000054 records, 6 variables).

This is the summary:

```
##      userId         movieId          rating        timestamp
##  Min.   :    1   Min.   :    1   Min.   :0.500   Min.   :7.897e+08
##  1st Qu.:18123   1st Qu.:  648   1st Qu.:3.000   1st Qu.:9.468e+08
##  Median :35741   Median : 1834   Median :4.000   Median :1.035e+09
##  Mean   :35870   Mean   : 4120   Mean   :3.512   Mean   :1.033e+09
##  3rd Qu.:53608   3rd Qu.: 3624   3rd Qu.:4.000   3rd Qu.:1.127e+09
##  Max.   :71567   Max.   :65133   Max.   :5.000   Max.   :1.231e+09
##
##                                 title                          genres
##  Pulp Fiction (1994)              :  34864   Drama                : 815084
##  Forrest Gump (1994)              :  34457   Comedy               : 778596
##  Silence of the Lambs, The (1991):  33668   Comedy|Romance       : 406061
##  Jurassic Park (1993)             :  32631   Comedy|Drama         : 359494
##  Shawshank Redemption, The (1994):  31126   Comedy|Drama|Romance: 290231
##  Braveheart (1995)                :  29154   Drama|Romance        : 288539
##  (Other)                          :9804154   (Other)              :7062049
```

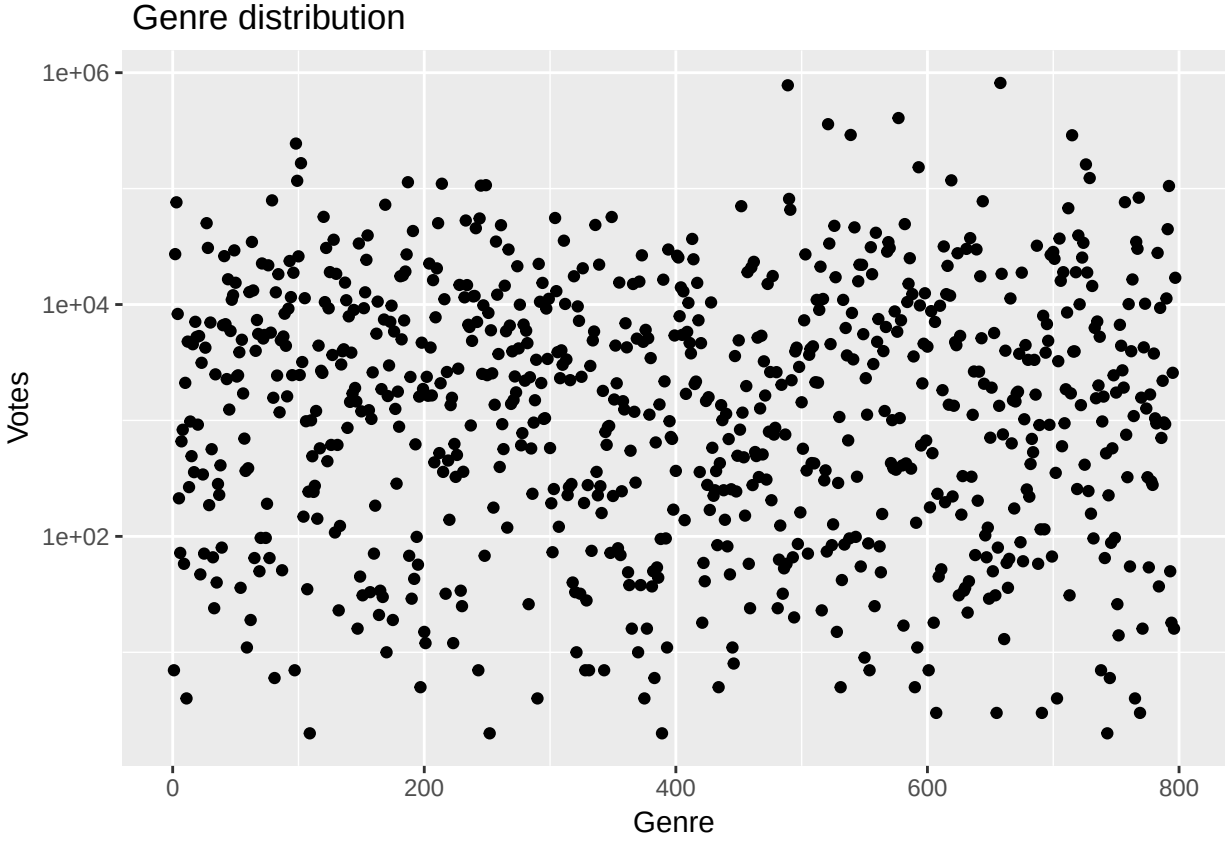
As shown in the table, there are not NA'S or empty values, so all the metrics are calculated on true data.

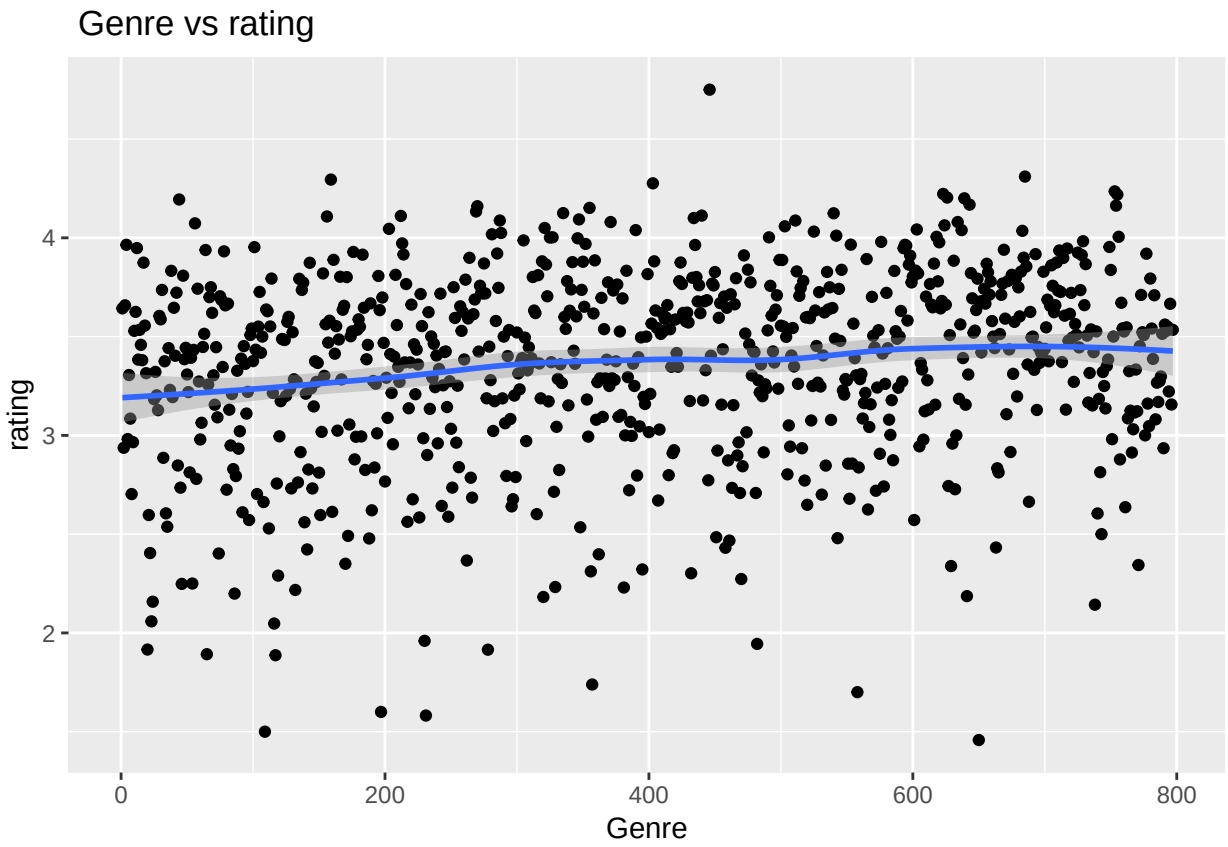
Data Exploration

In this section the variables are evaluated and explored using some visualization tecniques. First of all, the variability is took in consideration, but It's useful to ckeck the relationship with the "rating" label. Finally the correlation review.

- **The genres**

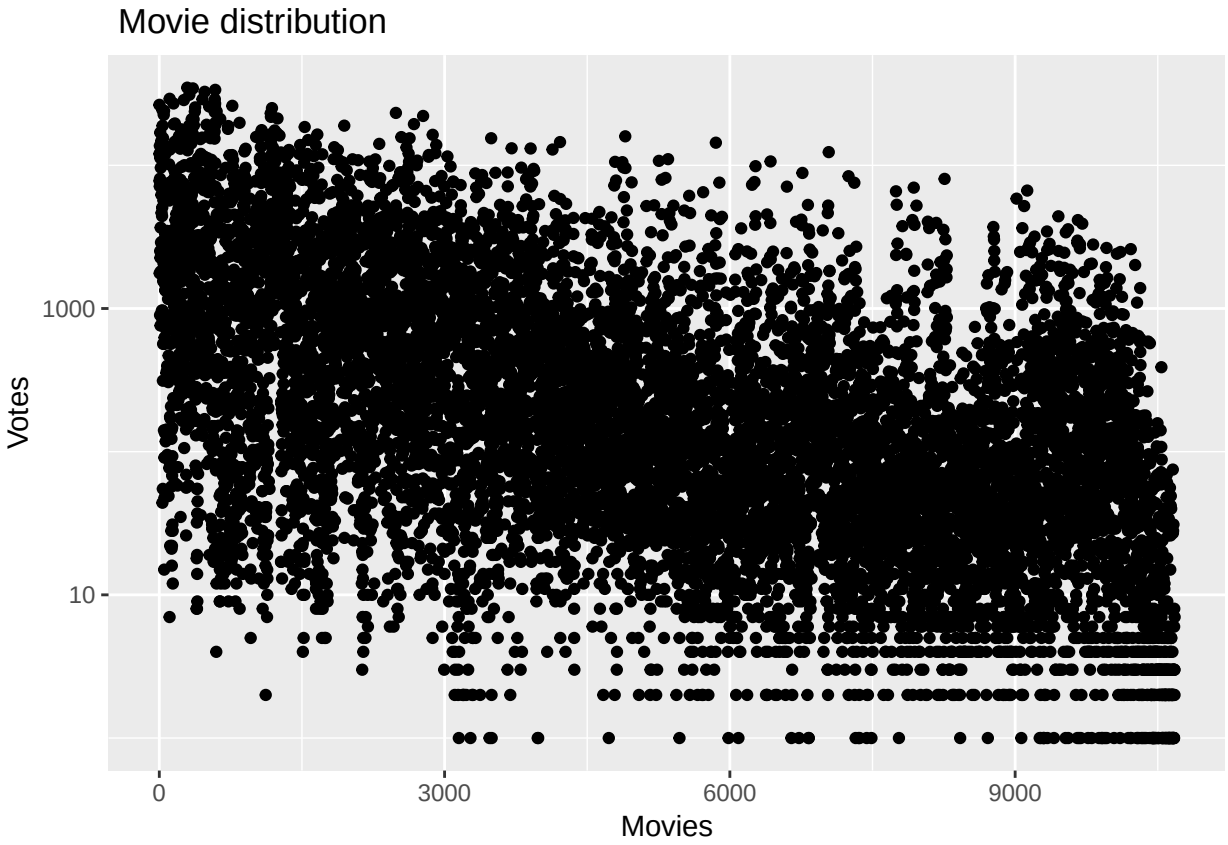
  There are 797 genres, but in the most of cases, they are combination of different genres. As shown in the first plot, there is great variability across genres, in the sense that blockbusters have much more votes than cult movies. Generally, with this kind of variability, regularization should be considered: items with few votes have lower weight compared to that most voted. The second plot suggests that the average rating has some variability across genres and this fact could be useful to improve prediction.

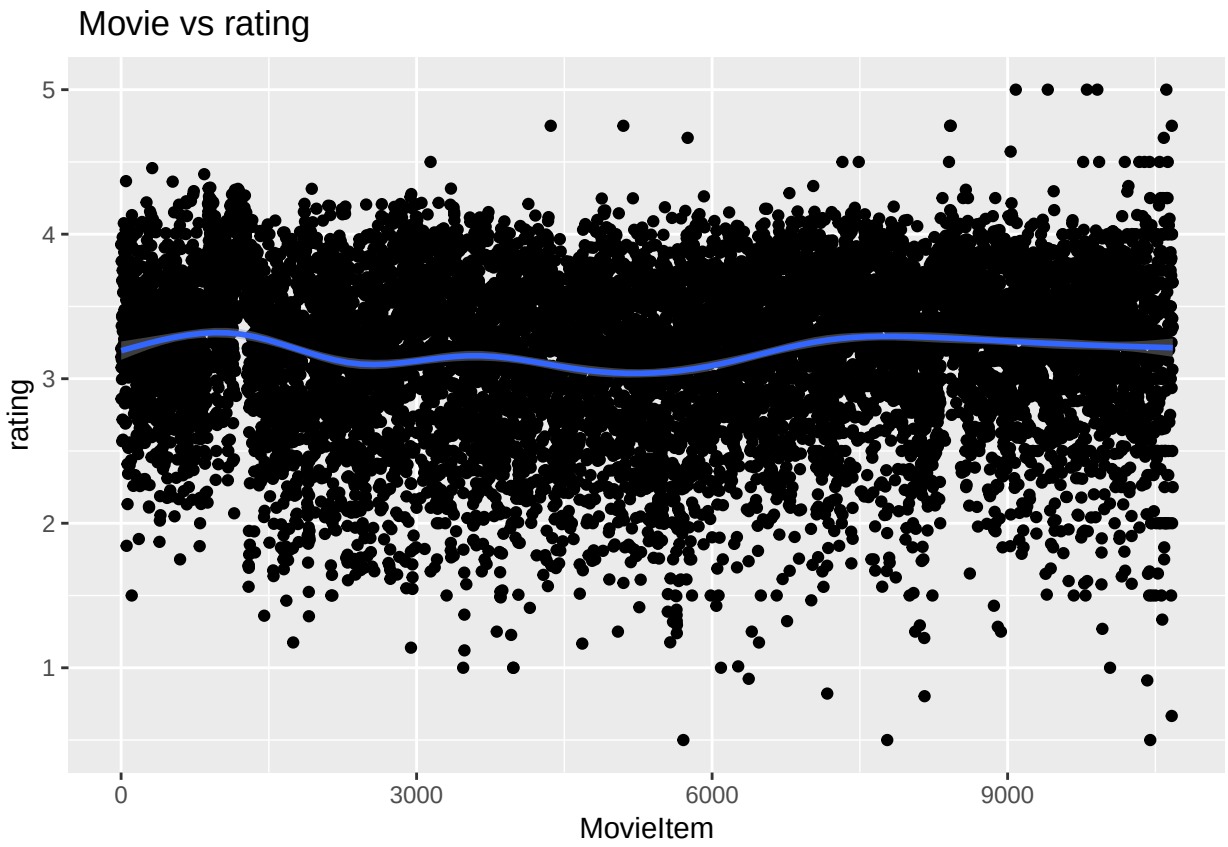




- **The movieId**

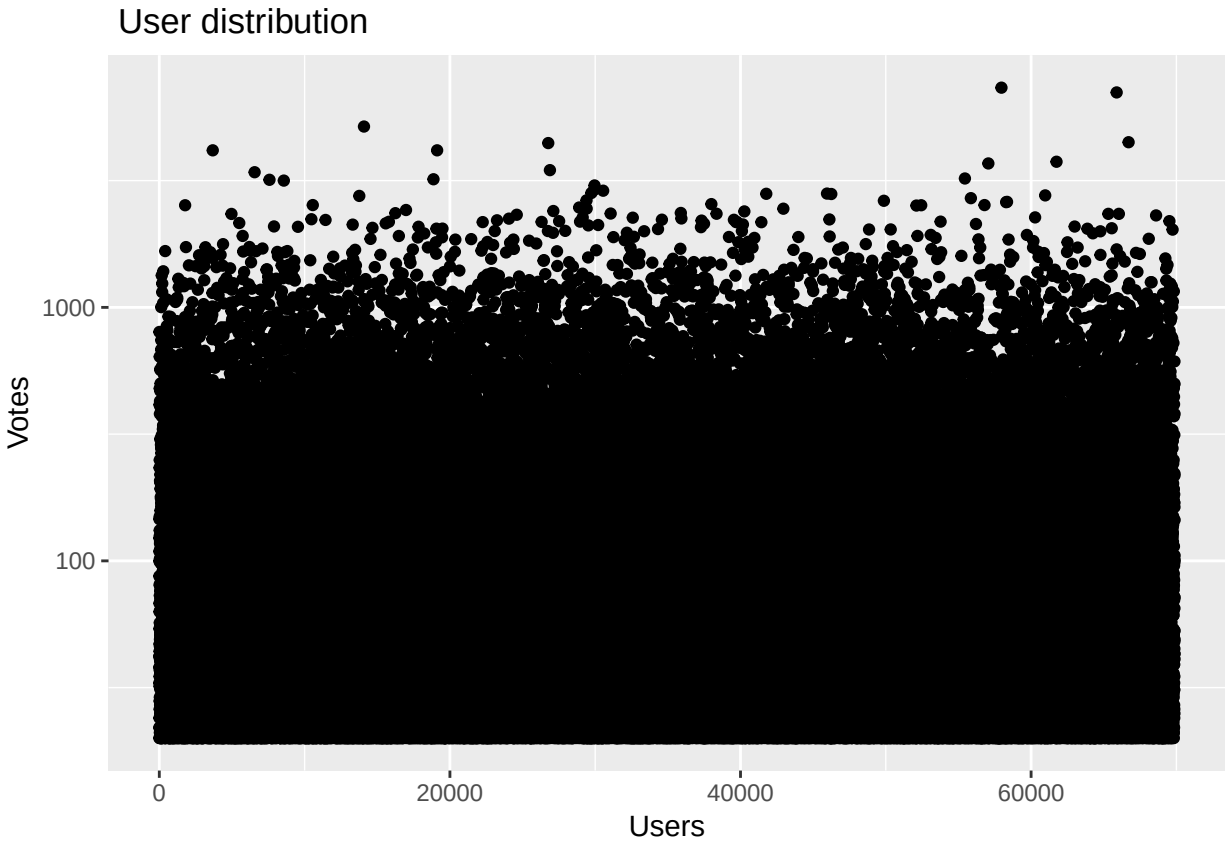
  As shown in the next 2 plots, and as intuition suggests, the movieId is a central feature that must be used as predictor. On the y axis of the first plot the scale is logarithmic to make the graphics more readable. There is great variability, among movies and votes. The second plot shows movie versus average rating, the smoothing gives fair evidence of variation, but also some apparent outliers that should be treated with regularization approach.

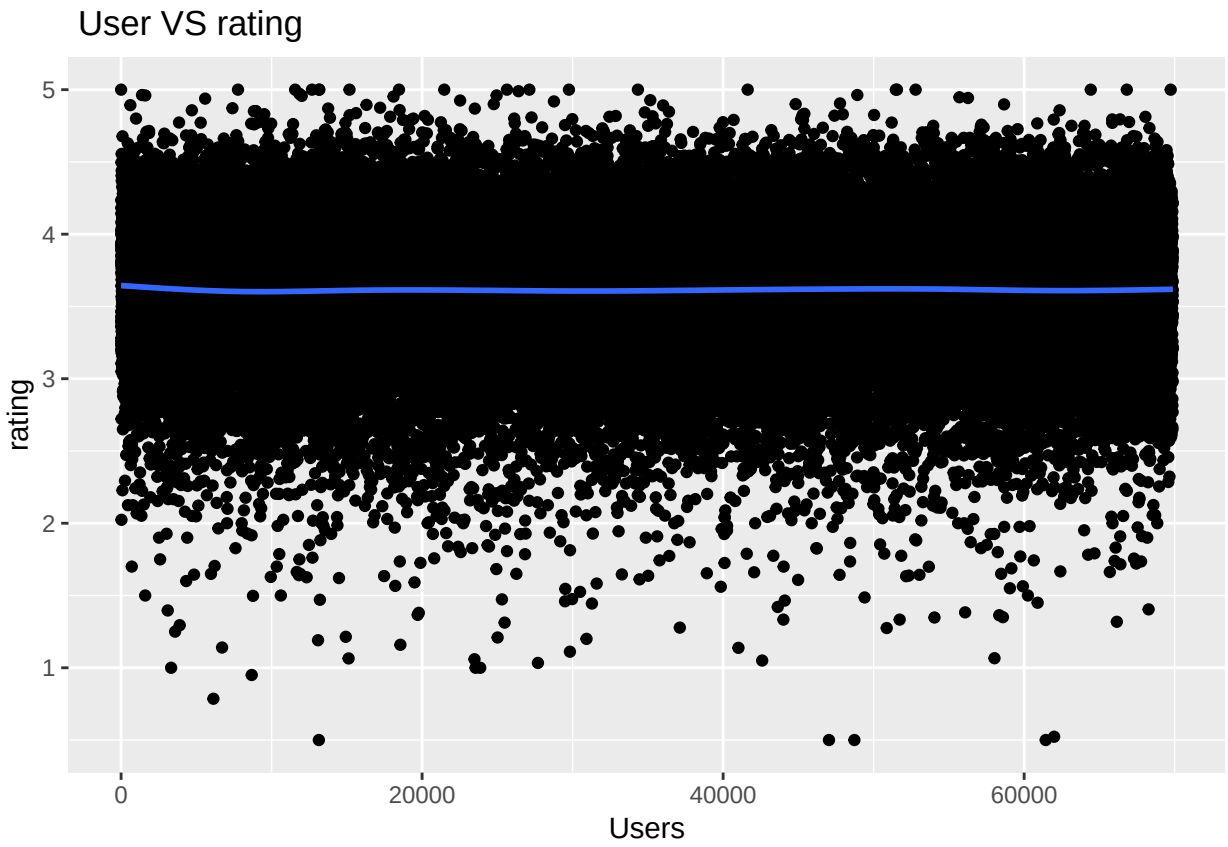

Movie distribution
Votes
1000
10
0
3000
6000
9000
Movies



- **The userId**

  The following two plots show users vs votes and ratings.

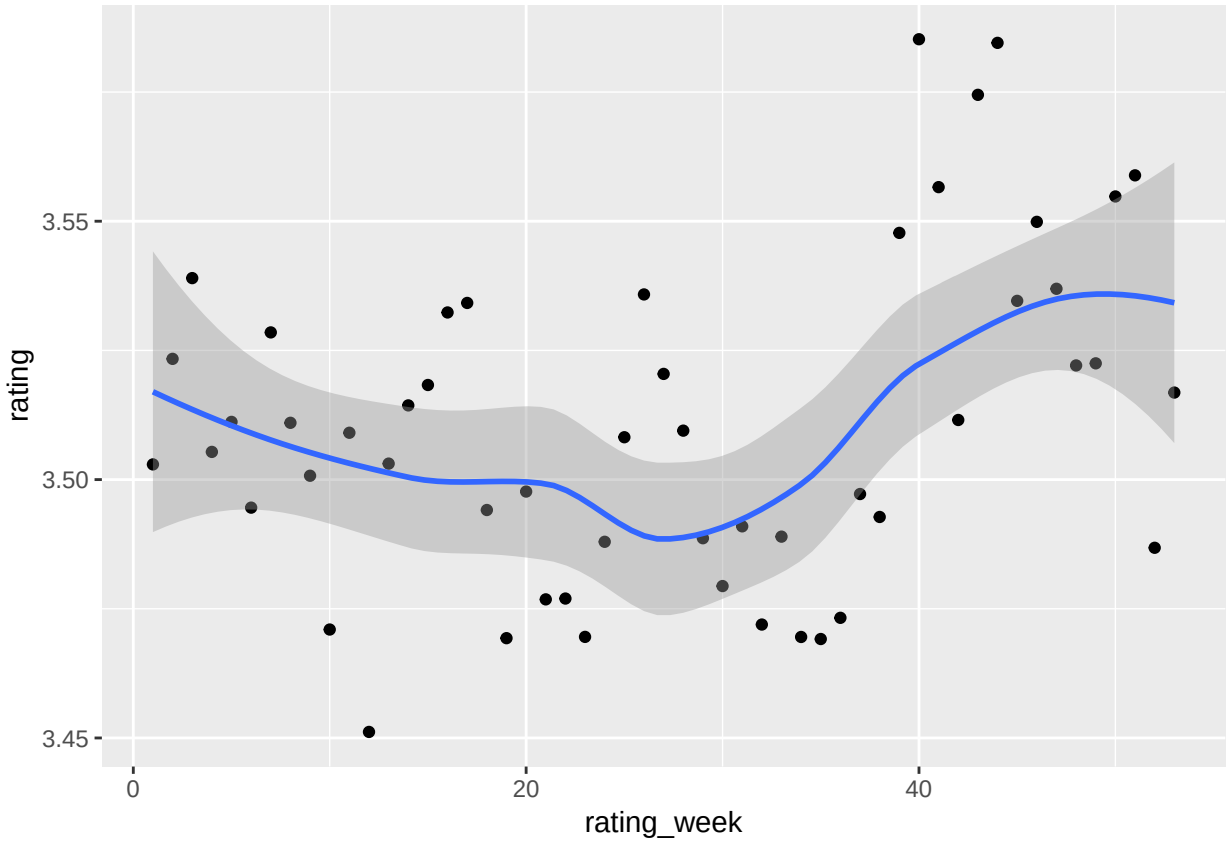

User distribution
1000
100
Votes
0
20000
40000
60000
Users

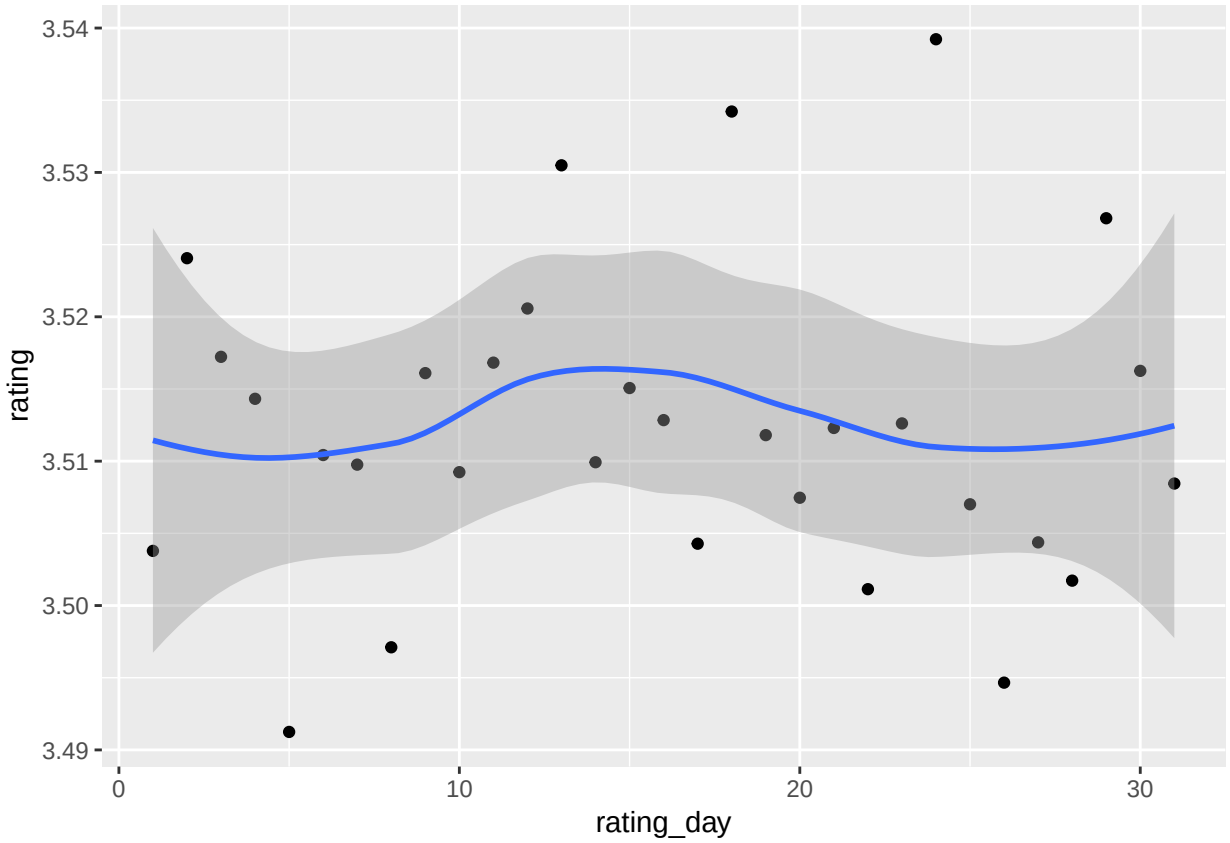


The first one shows in y axis - in the log mode scale – the amount of preferences among the users, the plot looks quite uniform, but there is a wide range of preferences, between the most active giving 7359 ratings to the laziest who gives 20.
The second plot shows Users vs rating, the smoothing function does not give a very useful information, but such a wide range of votes should be taken in account when valuing RMSE.

- **Timestamp**

  The next graphics shows the relationship between the two variables. The time base chosen for the analysis is the week; with the support of the smoothing function, it's quite clear that the timestamp has some effect on rating, but not so strong to be considered as a predictor factor.


rating
3.55
3.50
3.45
0
20
40
rating_week

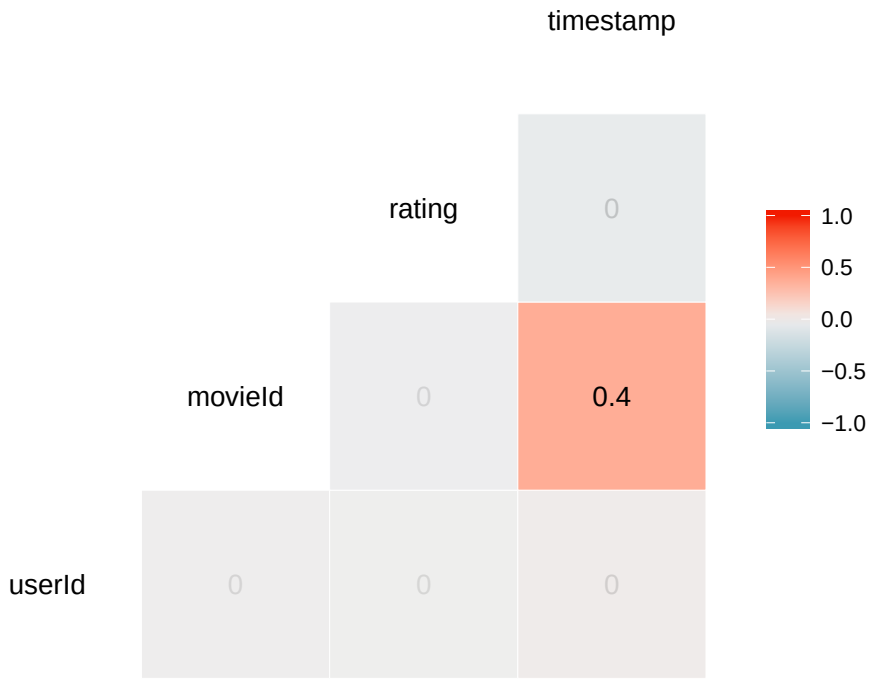


- **Title**

  The title has the same information of movieId (the difference is only the data type) and could be adopted only for some visualization.

- **Correlation**

  At the end of data exploration, it's important to inspect the relationship between variables, if two or more of them are correlated, they would give quite same information about prediction, so we will choose only the ones with no relationship among them. Using the R function ggcorr() from the GGally library, we can see at one glance the correlation between the numeric movielens variables, apart from not numeric variables that we early decided to keep out. Furthermore, because of the low prediction effect, rating_week factor will not be considered. The genreId is related to the string variable genres which have been converted into numeric to be evaluated in the correlation computation.



The modeling approach

To apply the machine learning concepts the data frame must be split in two data set:

- The training set which will be used to train the ML algorithm
- The test set on which the ML algorithm trained on training set will be implemented to make the predictions

In this project the **movielens** df is split in the **edx** data frame as training set and the **validation** data frame as test set. The *rating* is the variable to predict, it's a numeric real type so the model to be used is "regression" and the metric for the evaluation is the RMSE

- **Regularization and the user+movie effect approach**

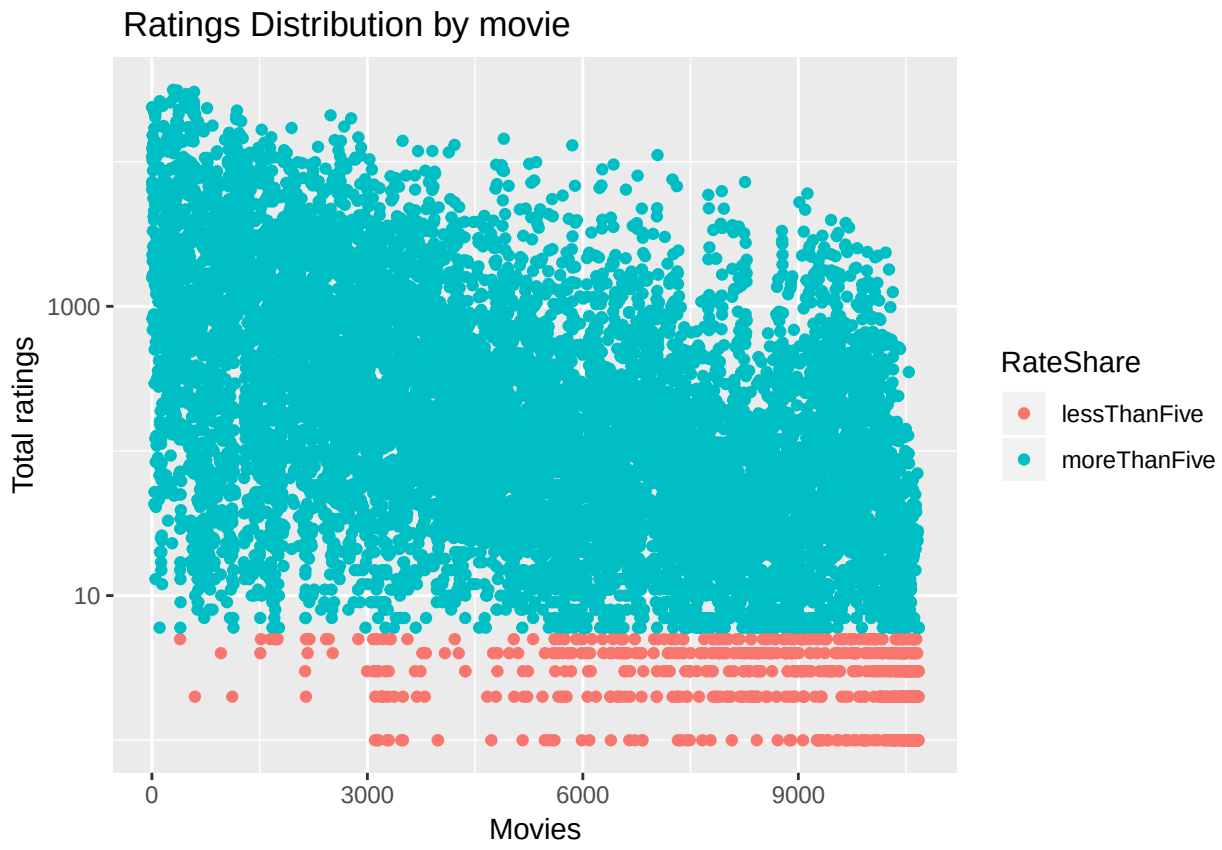
  This approach is based on the processing of the following formula: $Y_{u,i} = \mu + b_i + b_u + \varepsilon_{u,i}$

  Where:

  - $Y_{u,i}$ = The rating based on user+movie effect
  - μ= The average rating
  - b_i= movie effect
  - b_u= user effect
  - $\varepsilon_{u,i}$= error

  Although this is a valid approach, the rating distribution through movies, as hinted in the above paragraph, tells that some movies are rated more than others. Next graphics shows movies versus ratings. The graphic is split in two parts: the upper in blue referring to the movies with more than five ratings, the lower in red to that movies with five or lower number of ratings. The red section, just as

an example, shows that movies with few users ratings is a minor but a significant part of data. This are noisy data, that should be processed by regularization.



To overcome the problem of this kind of variability, it must be considered a mathematical model that minimizes the effect of low number of ratings and emphasize the weight of high rated movies. The central element of this model is a parameter λ which must be chosen after tuning operations, the formulas are:

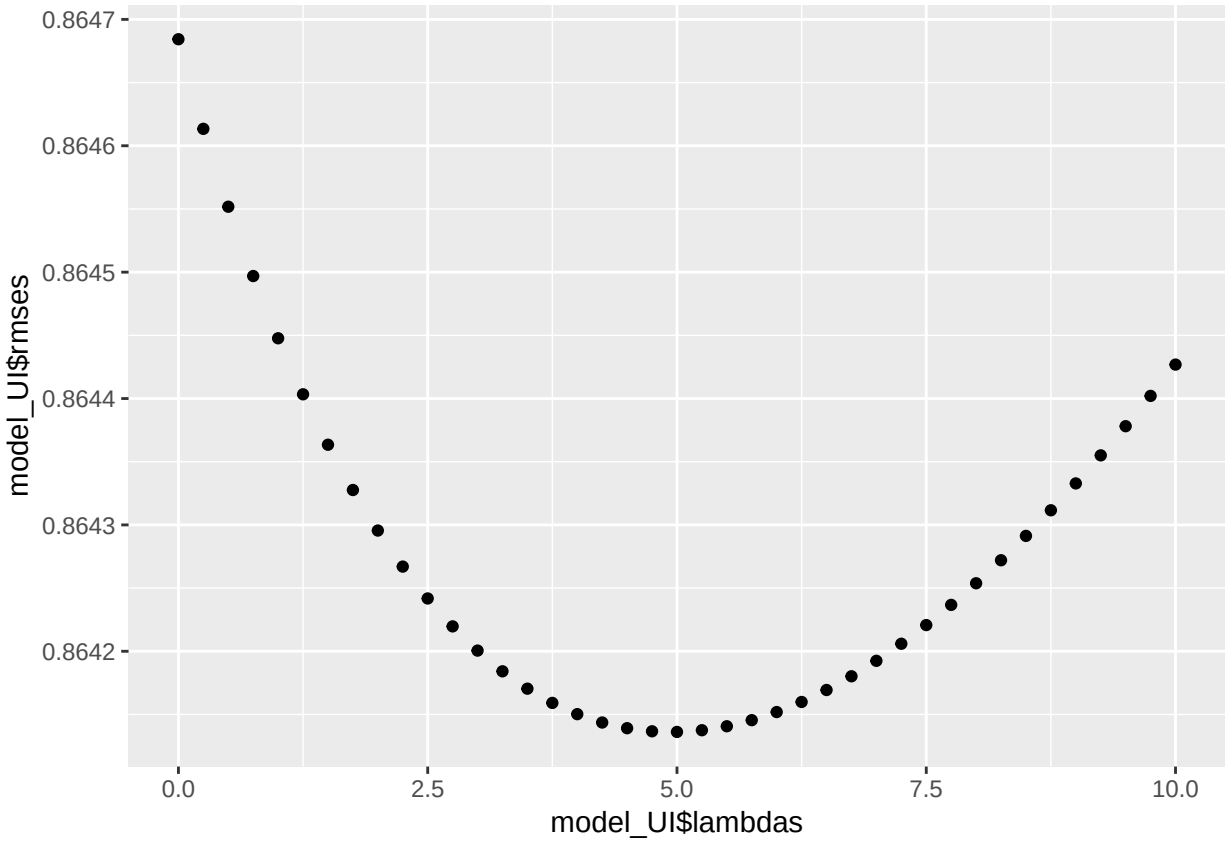
$\hat{b}_i(\lambda) = \frac{1}{\lambda + n_i} \sum_{u=1}^{n_i} (Y_{u,i} - \hat{\mu})$

$\hat{b}_u(\lambda) = \frac{1}{\lambda + n_i} \sum_{u=1}^{n_i} \left(Y_{u,i} - \hat{\mu} - \hat{b_i}\right)$

The tuning process consists of a CROSS VALIDATION procedure, which shoul be done only in training set, so we must further split the **edx** dataframe in two data frames:

- The *edx_train* that is used to calculate $\hat{b}_i(\lambda)$ and $\hat{b}_u(\lambda)$
- The *edx_test* as part of train data seta is used to compute RMSEs.

Then, we can plot the CROSS VALIDATION results.

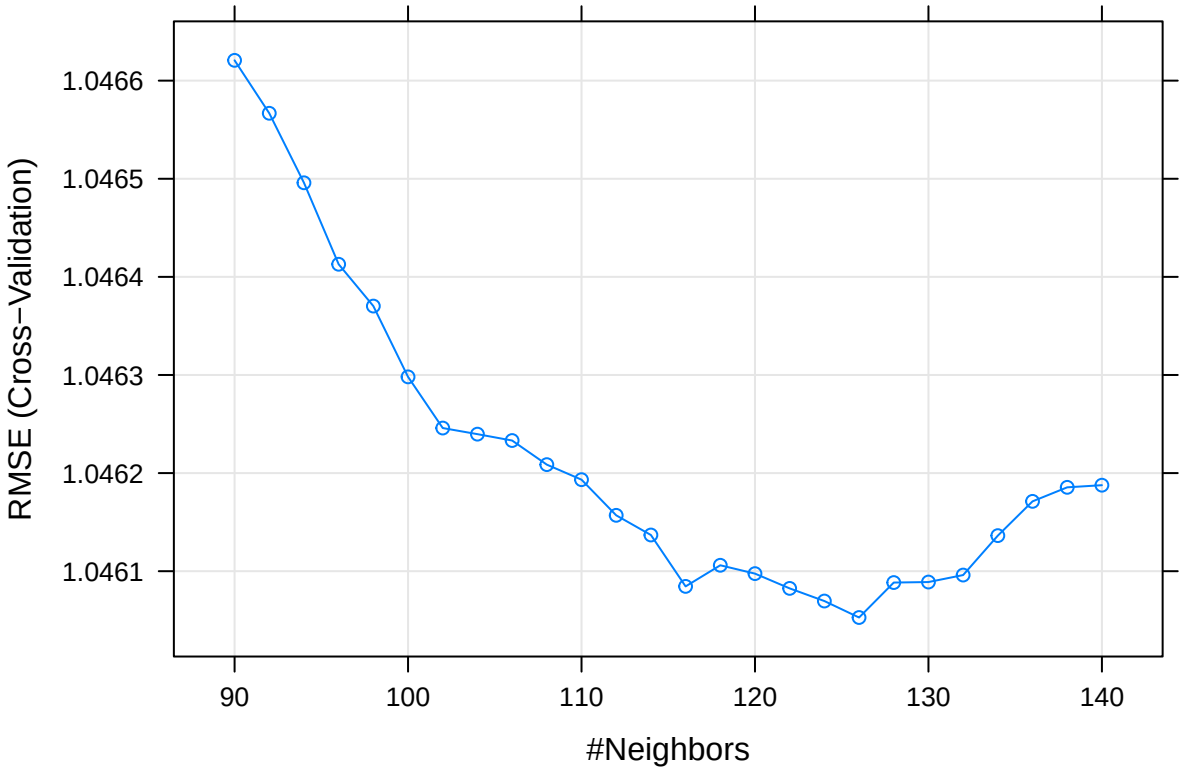


Next, let's apply the *best_lambda* on the target edx and validation sets and finally calculate predicted ratings

The RMSE is: 0.8648177

- **KNN approach**

The KNN ML algorithm has been used only on a sample of the whole edx dataframe. The reason is that it takes too long to compute 10M records dataset. Although the sample is the 6% of total records, the available hardware tooks about 12 hour computation time



The model_knn object has been saved on a file for safety. The computed best K is 126 on a range [90,130] step by 2, this is the plot

The RMSE is: 1.0658169

Results

The results of the two modeling approach are the following

method	rmse
knn	1.0658169
uireg	0.8648177

As described in the above section the Knn approach takes in count only the 6% of the entire dataset, this is only the final model. Many attepts have been done before this final result: different samples have been tested (3%, 5%) and different tuning parameters have been chosed from lower to higher k, any of this took long before the outcomes, even the 100% of data, but in this case the system crashed!. Probably different ML approaches or different tuning parameters would give better results. In any case the RMSE has been always above 1,00. The regularized UI approach has been faster, more effective and gave a good RMSE.

Conclusions

The results of regularized userId/movieId approcach are satisfactory, While KNN not. The genre factor has not been taken in consideration because of hardaware and computing systems restrictions. If computer

clusters and parallel system were available, either al least a more powerful computer, It would be possible more advanced approaches, one of these would use matrix factorization. Matrix factorization is very much related to factor analysis, singular value decomposition (SVD) and principal component analysis (PCA). The model userId+moveId leaves out an important source of variation related to the fact that groups of movies have similar rating patterns and groups of users have similar rating patterns as well, so the approach would be greatly improved by studying the *residuals*. The concept is based on the fact that through residuals is possible to figure out the relationship between movie genres and rating, this wouold be a further step to improve RMSE.