PH125.9x Capstone Examination - HarvardX

Report on MovieLens Project

Philippe Lambot

March 14, 2019

****

MOVIELENS PROJECT

****

****

****

MOVIELENS PROJECT

****

Important Foreword - Requirements

Dear Readers,

In this project, there are 6 files, which you can find in my GitHub repository: https://github.com/Dev-P-L/Recommendation-System . The 6 files are: Recommendation_Script.R, Recommendation_Report.Rmd, Recommendation_Report.pdf, movie_user_genres_time_results.csv, time_genres_movie_user_results.csv and README.md.

Recommendation_Script.R has been run in RStudio Version 1.1.456 - © 2009-2018 RStudio, Inc.

Recommendation_Report.Rmd has been knitted to HTML in RStudio Version 1.1.456 - © 2009-2018 RStudio, Inc. as well.

The version of R that I use on my PC is

- R version 3.5.1 (2018-07-02) – “Feather Spray”
- Copyright (C) 2018 The R Foundation for Statistical Computing
- Platform: x86_64-w64-mingw32/x64 (64-bit)

The operating system on my PC is Windows 10.

I cannot guarantee Recommendation_Script.R running on other versions. Neither can I guarantee Recommendation_Report.Rmd being knitted to HTML on other versions.

Before running Recommendation_Script.R or knitting Recommendation_Report.Rmd to HTML, **please adapt the working directory and choose a working directory on your machine. It has to be done on line 73 in Recommendation_Script.R and on line 220 in the file Recommendation_Report.Rmd. This is essential.**

When you have indicated in the code a working directory on your machine, then, and only then, you can run Recommendation_Script.R and knit Recommendation_Report.Rmd to HTML. **The file Recommendation_Script.R must absolutely be run before the file Recommendation_Report.Rmd can be knitted to HTML.** Indeed, data from MovieLens (edx and the validation set) are downloaded ONLY by Recommendation_Script.R, are saved in the chosen working directory and are available there for Recommendation_Report.Rmd to retrieve them and use them.

Recommendation_Script.R takes approximately 3 quarters of an hour to run on my laptop and it takes more or less 35 minutes to knit Recommendation_Report.Rmd (knit to HTML) on my laptop. But I need to close all other applications before launching these programs in order to avoid blue screens!

The code in both programs contains instructions to download the following packages if they are not available on your machine: tidyverse, caret, lubridate, broom and kableExtra. Data files will also be downloaded.

Good luck!

I. Introductory Section

This project takes place in the framework of the PH.125.9x Capstone Examination organized by HarvardX in Data Science. It is a recommendation system challenge aiming at achieving an RMSE lower than 0.8775 on a validation set sampled from the MovieLens 10M dataset.

In this introductory section, I will develop five subsections: the first one about my personal motivation, the second one about objectives and constraints of this project, the third one about datasets, the fourth one about the key steps from this project and the fifth one about documents management.

A. Personal Motivation

My personal motivation is **professional**. After years of data management, functional analysis, data analysis, I have decided to *touch base* again with formal education and to take a comprehensive programme in data science encompassing exploratory analysis and visualization, statistics, wrangling, machine learning and productivity tools. I have chosen HarvardX’s Professional Certificate in Data Science and this is the last step of it.

B. Objectives and Constraints of this Project

Of course, **my first objective is about performance: reaching an RMSE lower than 0.8775** when predicting ratings on the validation set. But I have decided to go a little bit further.

My second objective is analytical: getting some additional insight into rating variation by splitting rating variation into more types of effects, adding time and genres effects to the movie and user effects already developed in the course, even if this does not bring any further performance improvement.

My third objective is pedagogical, is about learning and practicing. I wish to practice visualization, statistics, time series analysis, wrangling, clustering, analysis, modeling and to get some additional experience in R and R Markdown. For instance, my interest in time series comes from many data analysis challenges having a time pattern, such as in finance, economics, sociology or linguistics. This is not the reproducible time from Galileo's throwing an object from the Pisa Tower but real time.

Aiming at these objectives had to be geared within the limits of two **constraints**: first, limitations in **time resources**; second, limitations in **computer power**, especially in RAM. **RAM management** is ubiquitous in my code.

C. Datasets

The HarvardX PH.125 Team has clearly shown the way to accessing and structuring data from the MovieLens 10M dataset accessible from:

https://grouplens.org/datasets/movielens/10m/

http://files.grouplens.org/datasets/movielens/ml-10m.zip .

The HarvardX PH.125 Team has designed an edx data frame of approximately 9 million rows and a validation set of roughly one million rows.

When conducting exploratory analysis and looking for an effective global model, I need a training set smaller than edx to limit the risks of slow processing, blue screens and sessions aborted. I also need a test set to test models without using the validation set. I have decided to **split edx in halves**, one sample being a training set and the second one a test set - well an "internal" test set, different from the validation set. I call these sets **edx_train** and **edx_test** in the rest of this report.

After designing an efficacious global model, I will run it, as a final model, on the whole of edx and validate it on the validation set. The validation set will in no way be used before the very final model. Ratings from the validation set will never be used to build up predicted ratings; they will only be used to measure the very final RMSE on the validation set.

D. Key Steps of the Project

The key steps of the project have already been adumbrated in the **table of contents** on the first page of **Recommendation_Report.pdf**.

After retrieving and structuring datasets, I will produce exploratory analysis including visualization, statistics, time series analysis, wrangling, clustering and effect per effect pre-analysis. I will analyze the dependent variable, i.e. ratings, and four independent variables, i.e. timestamp, genres, movieId and userId. I will especially focus on checking relevancy of time and genres effects as candidates for predicting ratings.

The next section is about global analysis and modeling. It will deal with designing global models, training them on edx_train and testing them on edx_test.

In the next section, about the final model, I will run the final model on edx and test it ("validate" it) on the validation set.

Results from the final model then come in a separate section, before reaching the conclusion section.

An appendix deals with RAM and layout management.

E. Documents Management

The **Recommendation_Script.R file contains code for all steps**, from downloading data to visualizing, wrangling, analyzing, modeling, testing and getting results.

The code from Recommendation_Script.R is largely **commented upon from a technical point of view in the Recommendation_Script.R file**. **Analytical comments** about data management are available **in Recommendation_Report.Rmd and Recommendation_Report.pdf**.

The biggest part of the code from Recommendation_Script.R has been copied to Recommendation_Report.Rmd, with **one notable exception**, i.e. the first part of the code from Recommendation_Script.R that downloads the MovieLens dataset and splits it into edx and the validation set. Indeed, when this part of the code is integrated into Recommendation_Report.Rmd, knitting Recommendation_Report.Rmd triggers blue screens, at least on my laptop. Consequently, I have run Recommendation_Script.R and created and saved edx and the validation set in my working directory. The code from Recommendation_Report.Rmd gets started with retrieving edx from the working directory.

Readers willing to run Recommendation_Script.R and JUST AFTER knit Recommendation_Report.Rmd (please knit to HTML) should be read hereinabove the "Important Foreword - Requirements".

For readers' convenience, the Recommendation_Script.R code leaves in the workspace, at the end, **three objects that contain predicted ratings on the validation set and the final RMSE on the validation set**. Consequently, readers have the opportunity to have a look at predicted ratings and not only at summarizing tables. Of course, there are also summarizing tables in Recommendation_Report.pdf.

As already adumbrated in the first chunk of code, I have opted in the YAML from Recommendation_Report.Rmd for an **html_document output** and I have issued it in **PDF format** as Recommendation_Report.pdf. In the numerous intermediate versions of this project, out of Recommendation_Report.Rmd I produced texts in html instead of pdf: they proved to be **very useful working documents** to cope with the equivalent of, altogether, more than 60 pages full of text, code, graphs (21) and short or long tables (26).

Among their advantages, let me first mention the table of contents containing dynamic links towards body text, which was most convenient.

Second, html presentation offers visual continuity, contrary to a PDF format which plays "whack_a_mole" with graphs and often leaves graphs and comments separated on different pages (see e.g. the excellent "R Markdown: The Definitive Guide" https://bookdown.org/yihui/rmarkdown/r-code.html#figures , where the expression "whack-a-mole" comes from).

Third, the html version allows to introduce in Recommendation_Report.Rmd the useful code "code_folding: hide", which permits, in the rendered html working document, to "press" a button, section by section, and visualize the code which remains otherwise hidden (see https://bookdown.org/yihui/rmarkdown/html-document.html#code-folding); this option is most useful since it combines two advantages: fluency in reading text and availability of code when desired!

I could not use a fourth advantage from html and kableExtra combined together, i.e. highlighting important results from tables with colored backgrounds and font, because these options did not show in the final document in PDF.

The documents are **accessible in a repository in GitHub**. Here is the link again: https://github.com/Dev-P-L/Recommendation-System .

This repository contains the files Recommendation_Script.R, Recommendation_Report.Rmd and Recommendation_Report.pdf. It also contains the files movie_user_genres_time_results.csv, time_genres_movie_user_results.csv. Both files come from similar programs that have not been included to not overburden this project. These files play absolutely no role in the code that leads to the final results.

After this introductory section, let's move to the first operational section.

II. Retrieving edx and Splitting edx into edx_train and edx_test

Let's retrieve edx and split it in halves into edx_train and edx_test. Both will be saved in the working directory so that they can be removed from workspace when they are not needed in code, to spare RAM.

Splitting edx in halves

- permits edx_train and edx_test to be both representative of edx from a **statistical point of view**, each containing approximately 4,500,000 observations
- and permits **RAM management**, which has been ubiquitous in my project, as already adumbrated hereinabove.

Let's now explore edx and its components.

III. Exploratory Analysis including Visualization, Statistics, Time Series Analysis, Wrangling, Clustering and Effect per Effect Pre-analysis

A. Observations Number and Variables in edx

Let's check the number of observations in edx.

Number_of_rows_in_edx
9000055

Here is the list of variables from edx.

Variables_in_edx	Status
rating	Dependent_variable
userId	Independent_variable
movieId	Independent_variable
timestamp	Independent_variable
title	Independent_variable

Variables_in_edx	Status
genres	Independent_variable

There are five independent variable candidates (predictor candidates).

Relevancy of movie and user effects has already been demonstrated in the PH125.8x course; it needs confirming and illustrating here. Information about users comes from the variable userId. Information about movies originates in two variables: title and movieId. movieId has been used as the identifier of movies, the variable title providing here some complementary information.

Relevancy of time and genres has to be fundamentally checked here. Information about time is available in two variables: the timing of the ratings is in timestamp; the timing of the movies (release dates) is included in the variable title. I will concentrate on timestamp because it gives the timing of the dependent variable. Investigating release dates falls beyond the time I can allocate to exploratory analysis.

Consequently, I will keep four predictors, timestamp, genres, movieId and userId, the variable title only providing some complementary information.

Let's explore variables from edx.

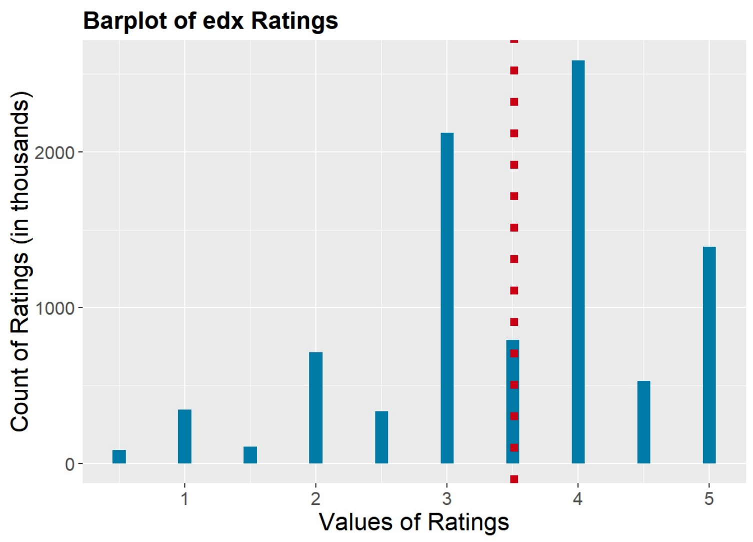
B. Distribution of the Dependent Variable in edx, i.e. rating

Let's have a look at some characteristics from the distribution of the dependent variable, i.e. rating. First the mean and median of edx ratings.

Mean_of_Ratings_in_edx	Median_of_Ratings_in_edx
3.512465	4

In the table above, we see that the average and the median of ratings are different.

Let's create a barplot of ratings. I have preferred a barplot rather than a histogram because the variable rating is a discrete variable and a histogram might create a fallacious impression of continuity.



In the graph above, we see that the mean (represented in crimson) is different from the mode. We also note that stars have more occurrences than the nearest half-stars (e.g. more 4 and 3 ratings than 3.5 ratings).

Information about medians (by subcategories), mode and prevalence of stars was **explicitely** dealt with when I was busy with the accuracy version of this project. In the RMSE version, variation will be taken care of by using means from subcategories, i.e. time periods, genres, movies and users.

C. Is Time Relevant as an Independent Variable, as a Predictor?

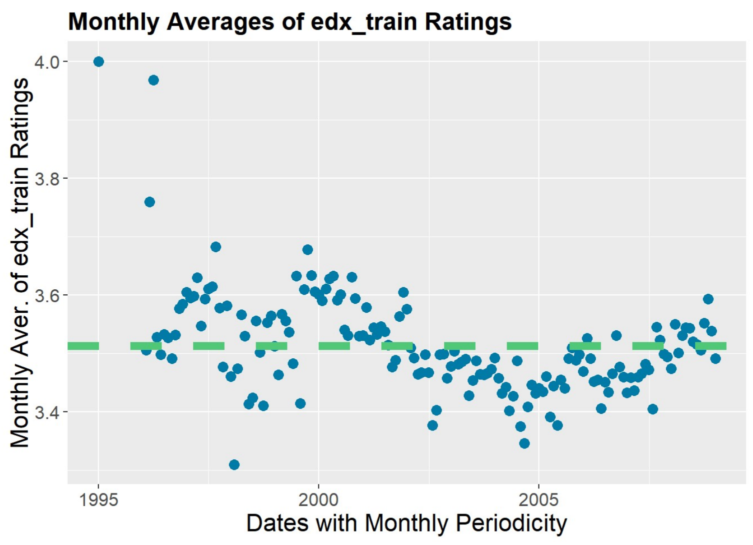
Let's get started with assembling ratings per month, per week, per day and per hour. I prefer to use that broad a spectrum of periodicities in order to organize a kind of cross-validation: which periodicity works best?

I also used irregular time periods, i.e. time periods determined by the levels of ratings: as long as successive ratings remained around a specific level, it was one period. This did not improve results in terms of RMSE and I do not report on them in order to not overburden this report that is already rather long.

1. Visual Evidence of Time Effect

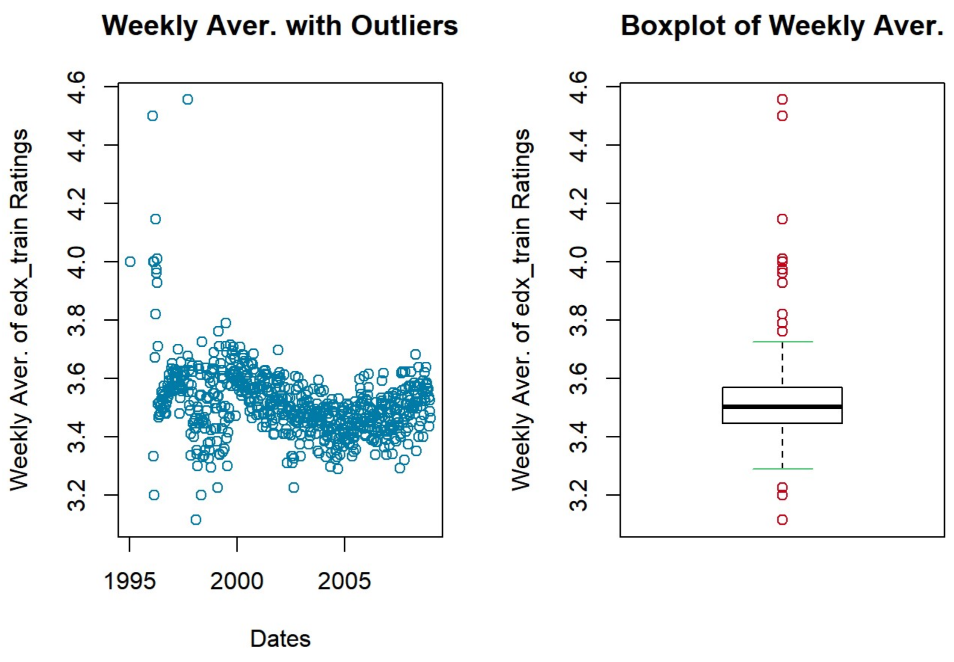
We'll work on edx_train instead of edx for 2 reasons: managing RAM and leaving an opportunity to test on edx_test.

Let's first give a graphical representation of monthly averages of edx_train ratings. The general average of edx_train ratings is represented by a horizontal line in green.

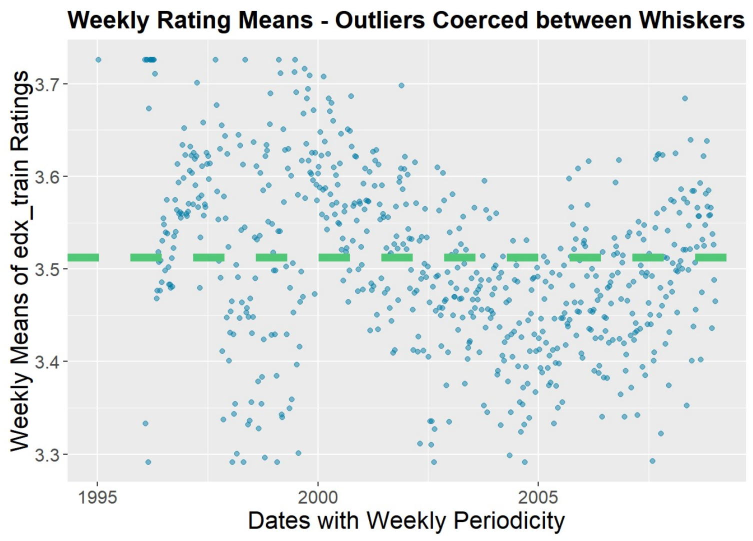


Visually, there seems to be a **slight time effect**.

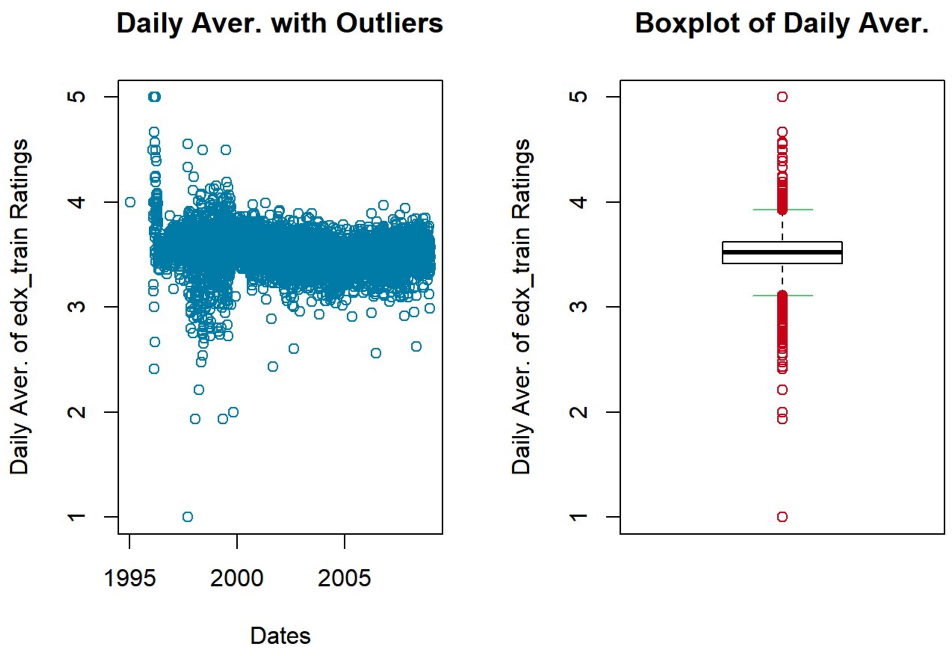
Let's now move to **weekly averages** of edx_train ratings. A dual graphical presentation will be useful.



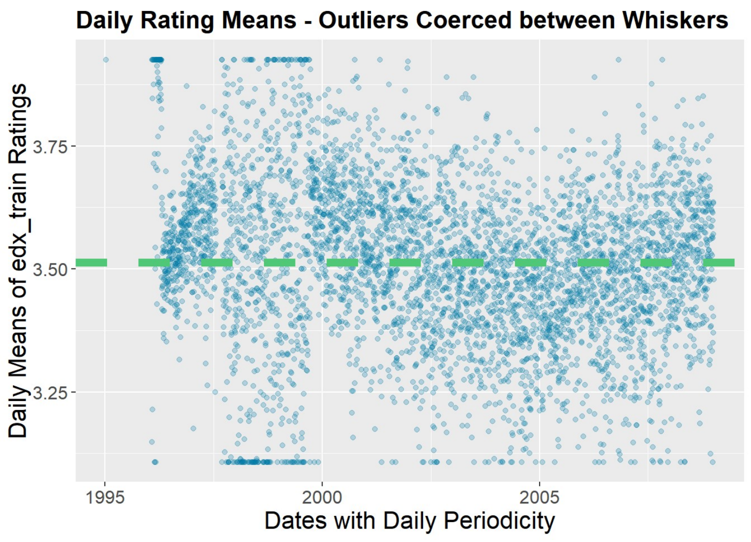
For visualization purposes, let's widen the main band of points by **"pulling" outliers towards the center**, stopping at the level of the extremes of the boxplot whiskers. Let's see the new graph.



The visual impression of time effect remains. Let’s move to **daily averages**.

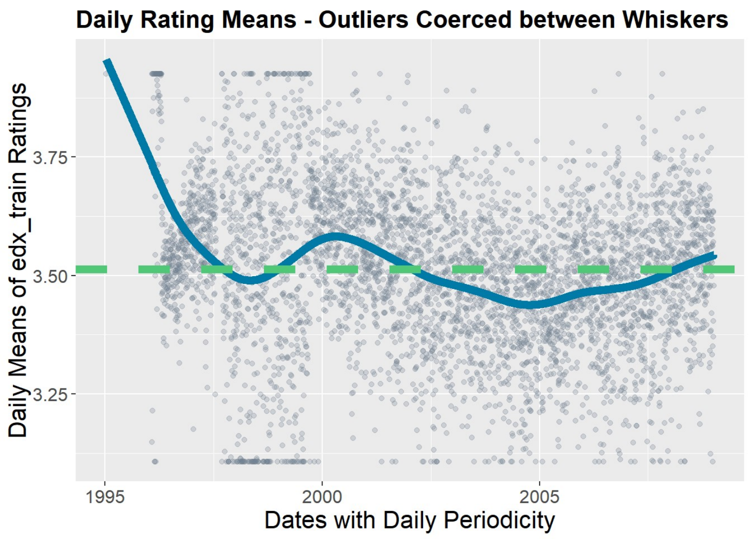


Let’s draw a graph with daily averages of edx_train ratings coerced between whiskers.

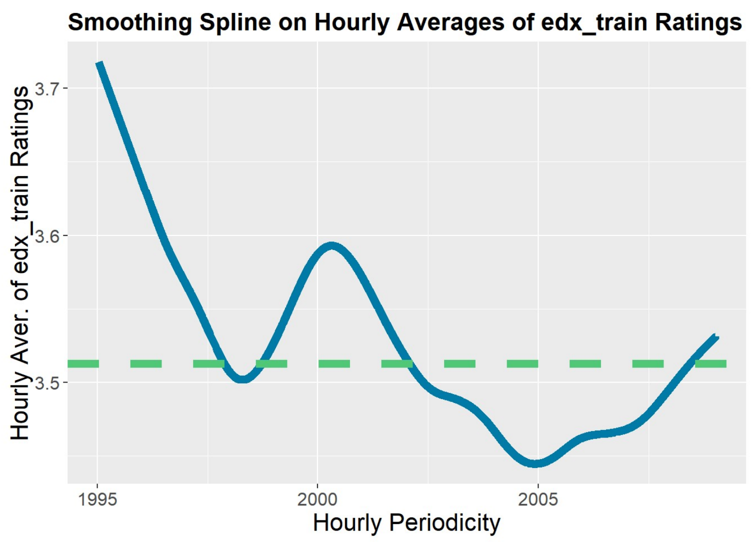


Visual evidence can be highlighted by smoothing daily averages of edx_train ratings, e.g. with a **smoothing spline**. With the option df = 10 in the code from the smooth.spline() function, I have determined the equivalent number of degrees of freedom of the smoothing spline and regulated the smoothness degree (see e.g. https://www.rdocumentation.org/packages/stats/versions/3.5.2/topics/smooth.spline). Fitted values from the smoothing spline are represented as a blue line on the next graph.

For illustrative purposes, points representing daily averages of edx_train ratings have been coerced between whiskers (to reduce the y-axis range); this has not been done to run the smoothing spline, which has been applied to uncoerced daily averages of edx_train ratings (as can be seen in the code).



Representing **hourly averages** on the same kind of graph would lead to some visual fuzziness. Let’s only represent visual results from a smoothing spline (in blue) with the general average of ratings (in green). There is a temporal pattern, even if variation range remains limited!



2. Statistical Evidence of Time Effect

After visual evidence having been provided, let's look for another kind of evidence. Is there a statistically significant time trend in ratings? Let's start checking with a **first linear regression of ratings on weekly dates**, which is an intermediate periodicity…

```
##
## Call:
## lm(formula = rating ~ date_week, data = temp)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.0084 -0.5414  0.4229  0.5297  1.5504
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.840e+00  4.475e-03  858.10   <2e-16 ***
## date_week   -3.169e-10  4.306e-12  -73.59   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.059 on 4500265 degrees of freedom
## Multiple R-squared:  0.001202,	Adjusted R-squared:  0.001202
## F-statistic:  5415 on 1 and 4500265 DF,  p-value: < 2.2e-16
```

The coefficient of the time trend is **statistically significant according to the p-value**. But **the R-squared (R2) is minute**.

With respect to the statistical significance of the coefficient, we know that the validity of the T-tests is **conditional on some statistical assumptions being fulfilled**. I could check e.g. the normality of residuals, the lack of heteroscedasticity (variance changing over time) and of autocorrelation.

In case of e.g. heteroscedasticity, I could change the model, taking e.g. a curved time trend (as I will do it anyway in some of the following regressions). Among innumerable contributions to that field, I would just refer to an illustrative one: https://datascienceplus.com/how-to-detect-heteroscedasticity-and-rectify-it/.

I could also use a variant of T-tests that is robust against heteroscedasticity and/or autocorrelation of residuals, as explained in e.g. http://eclr.humanities.manchester.ac.uk/index.php/R_robust_se#Autocorrelation_and_heteroskedasticity_robust_standard_errors .

Investigations about normality, heteroscedasticity and autocorrelation are beyond the scope of this project and of resources.

Anyway, we already have **some visual evidence of time effect**, we also have **a first indication of statistical significance**. What we need now is **moving towards predictive power** as required in this project. I am not going to mention any more p-values, which are anyway inferior to 0.001 in all regressions in this project.

I will try and improve the R2 in the following linear regressions by changing the time trend. In the first linear regression the time trend was simply a series of dates with weekly periodicity. Let's try a time trend with more flexibility in a second regression: splined weekly averages of edx_train ratings.

3. Performance Evidence from Different Time Trends

Let’s remember that R2 is simply 1 - (SSE / SST), where SSE is the sum of squared errors (residuals) and SST is the total sum of squares (here sum of the squared differences between ratings and the mean of ratings). SST is simply the sum of residuals of the baseline model, which consists of using the mean of edx_train ratings to predict edx_train ratings. Since SST is the same for all time trends, **we can increase R2 only by decreasing the SSE on edx_train**.

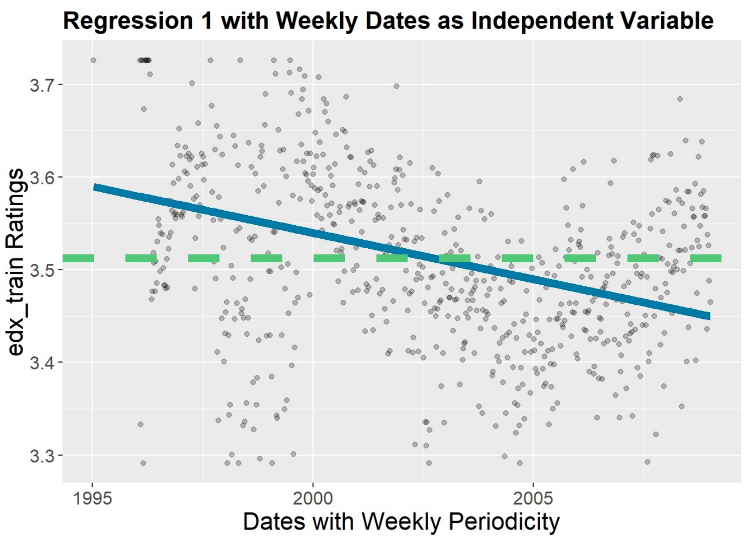
That will also automatically decrease the RMSE on edx_train, which is simply the square root of the division of SSE by the number of observations. When we have found the best time trend on edx_train, we can test it on edx_test, where we will use RMSE instead of R2 since it is the official criterion in this project.

Let’s launch the second regression: edx_train ratings will be regressed on splined weekly averages of edx_train ratings. This time trend is curved and is no longer a straight line of dates as in the first regression. Let’s **compare results from regression 1 (with dates as x data) and regression 2 (with splined weekly averages of edx_train ratings as x data)**. We’ll use the broom package and the glance() function to extract R2.

Regression	x_data	R2	SSE	SST	Checking_R2	RMSE_on_edx_train
Regression 1	Dates with Weekly Periodicity	0.0012019	5050777	5056854	0.0012019	1.059400
Regression 2	Splined Weekly Averages of Ratings	0.0023834	5044802	5056854	0.0023834	1.058773

R2 from the second regression is **slightly higher** and, automatically, RMSE is slightly lower.

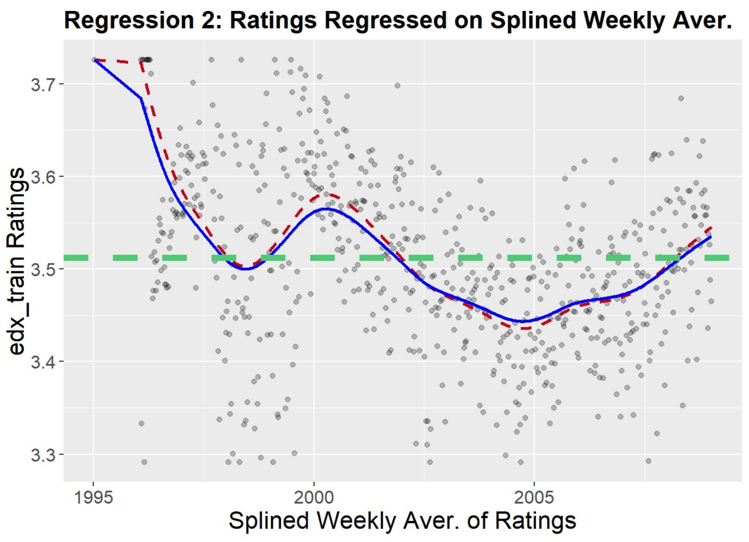
Why have we obtained slightly better results? Let’s visualize on graphs the results from the two regressions. First a graph with results from the first regression of edx_train ratings on weekly dates; fitted values are represented as a blue line; for illustrative purposes, (coerced) weekly averages have been added as points.



We see that, as a **straight line**, the time trend can capture the downward trend but not the curved patterns from edx_train ratings (here represented by weekly averages).

Let’s have a look at a graph with the second regression: edx_train ratings regressed on splined weekly averages of edx_train ratings. As a blue line, the fitted values from the second regression. In crimson, the x values from the regression, i.e. the splined weekly averages. There is very little difference between the two series. In each series, one value has been decreased at the very beginning of the period in order to improve visual comparability with the previous graph.

For illustrative purposes, weekly averages of edx_train ratings (coerced between whiskers) have been added as points.



The blue line is curved and is in a better position to capture rating movements than the straight line from the previous graph.

Before moving to more time trends, let’s slightly digress into a small methodological point. In the second regression, we regressed ratings on splined weekly averages of ratings. Fitted values from the regression were very close to input values, as could be seen on the last graph. Knowing that, **why stick to the linear regression as an evaluation and comparison tool?**

Using linear regressions offers two advantages in this context.

On the one hand, linear regression has provided the analytical comfort of statistical significance for all time trends, within the limits circumscribed above.

On the other hand, linear regression offers the methodological comfort of accepting two very dissimilar kinds of input data here: linear regression accepts, as input data, series of dates (weekly dates in the first regression) as well as a transformation of ratings (splined weekly averages of ratings in the second regression).

Let’s go back to our comparison.

With regression 2, R2 is already slightly higher but it still remains minute. Let’s investigate more broadly **in search of other possible time representations**. We already have 11 possible time trends:

- 4 lists of time units (monthly, weekly, daily or hourly periodicity),
- 4 lists of averages of ratings (with each of the four periodicities),
- 3 smoothing splines (smoothing spline, with df = 10, on weekly, daily or hourly averages).

Let’s add a few ones, varying fitting degree. First we can add a smoothing spline on monthly averages (with df = 6 to preserve smoothness of monthly periodicity).

Then we can add a smoothing spline with built-in cross-validation to each of the
4 periodocities. I have opted for cv = TRUE, which triggers a **leave-one-out cross-validation** process (see e.g. https://www.rdocumentation.org/packages/stats/versions/3.5.2/topics/smooth.spline).

By the way, I also tried the option cv = FALSE, which triggers a “generalized” cross-validation process; results in terms of R2 on edx_train and RMSE on edx_test were very close to the results obtained with the option cv = TRUE; consequently, I am not going to add the results with cv = FALSE to this report to not overburden it.

Now we have **16 time trend candidates**: 4 series of dates, 4 series of averages, 4 series of splined averages with option df = 6 or 10 and 4 series of splined averages with leave-one-out cross-validation.

Let’s organize a results table, **run a for loop with linear regressions** of edx_train ratings on the 16 time trends and insert outputs into the results table. Let’s have a look at that table.

Time_Trend_on_edx_train	**R2**
Hourly av. of ratings	0.1154066
Daily av. of ratings	0.0190400
Weekly av. of ratings	0.0070829
Monthly av. of ratings	0.0048071
Daily av. splined with leave-one-out CV	0.0044471
Hourly av. splined with leave-one-out CV	0.0041705
Monthly av. splined with leave-one-out CV	0.0033157

Time_Trend_on_edx_train	R2
Weekly av. splined with leave-one-out CV	0.0029473
Hourly av. splined with 10 DF	0.0027632
Weekly av. splined with 10 DF	0.0023834
Daily av. splined with 10 DF	0.0022906
Monthly av. splined with 6 DF	0.0022433
Weekly dates	0.0012019
Monthly dates	0.0012016
Hourly dates	0.0012014
Daily dates	0.0012013

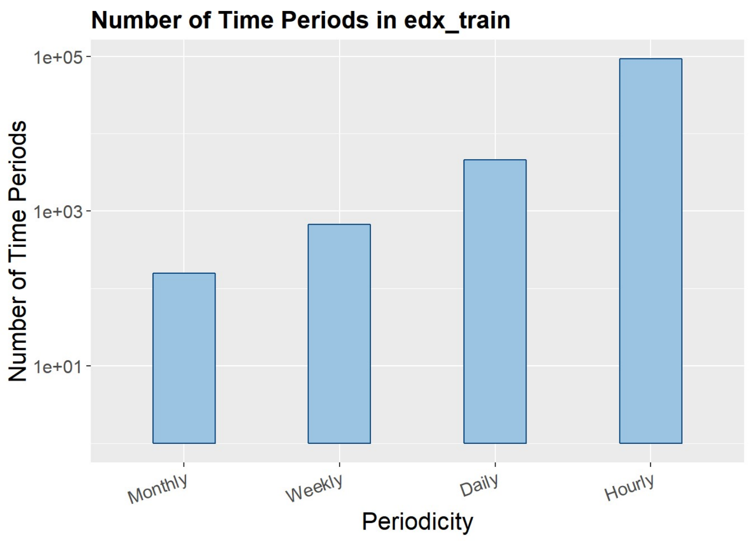
We get the **highest R2 for hourly averages of ratings, with daily averages as a second best**.

I would like to comment upon these results from two points of view.

On the one hand, periodicity strongly impacts the number of time periods, as will be shown by the next graph and the next table. Shorter periodicities (hourly or daily) have many more time periods, especially so for hourly periodicity! **More time periods can allow greater leeway in getting more flexible x data in the regression.**

As shown by the last table above, hourly and daily periodicities are on top in terms of R2 for averages. They remain on top (in reverse order) for averages splined with leave-one-out cross-validation, but R2 becomes tiny. For averages splined with predetermined number of degrees of freedom, hourly periodicity is on top and monthly at the end. For series of dates, order is completely different but here the number of time periods is not necessarily an asset since x data in the regression remain anyway a “flat” line of dates!

Here is a graph that illustrates the wide spectrum of time period counts per periodicity.



Here is a table with the number of time periods per periodicity.

Periodicity	Number_of_Time_Periods
Monthly	157
Weekly	671
Daily	4630
Hourly	92924

On the other hand, methodology also impacts R2 performance, combining with periodicity. We have seen that averages of ratings come first. Averages splined with leave-one-out cross-validation are below, which is normal: spline means less fitting! Averages splined with a predetermined number of degrees of freedom is still a little bit lower: normal since I have deliberately smoothed more with the number of degrees of freedom than the cross-validation process did! Series of dates come last, which is also normal since series of dates are simply a time line without inflection, producing a straight line in the regression!

Let’s check up on this **double interpretation related to flexibility**: let’s add a column with the standard deviations of the time trends, used as a rough estimate of the fitting degree of the corresponding time trend (correlation could also be used but would look rather tautological with respect to R2, wouldn’t it?). We have recourse to the sapply() function.

For illustrative purposes, standard deviations will be zeroed for dates, since dates series are just monotonously increasing functions of time without inflection.

Time_Trend_on_edx_train	R2	Fitting_Degree
Hourly av. of ratings	0.1154066	0.3601110
Daily av. of ratings	0.0190400	0.1462697
Weekly av. of ratings	0.0070829	0.0892125
Monthly av. of ratings	0.0048071	0.0734956
Daily av. splined with leave-one-out CV	0.0044471	0.0638053
Hourly av. splined with leave-one-out CV	0.0041705	0.0648504
Monthly av. splined with leave-one-out CV	0.0033157	0.0604158
Weekly av. splined with leave-one-out CV	0.0029473	0.0620303
Hourly av. splined with 10 DF	0.0027632	0.0536499
Weekly av. splined with 10 DF	0.0023834	0.0617195
Daily av. splined with 10 DF	0.0022906	0.0603577
Monthly av. splined with 6 DF	0.0022433	0.0534855
Weekly dates	0.0012019	0.0000000
Monthly dates	0.0012016	0.0000000
Hourly dates	0.0012014	0.0000000
Daily dates	0.0012013	0.0000000

Visual statement: for averages (lines 2 to 5 from the table above, headers included) higher levels of R2 correspond to higher levels of the estimated fitting degree. On lines 6 to 17, that relation remains but as a general tendency.

The main question now is: **will the hourly averages of ratings not show overfitting when run on edx_test?**

Among the 16 time trends, two groups can be distinguished: 4 lists of dates and 12 transformations of edx_train ratings. The 4 lists of dates cannot be applied as such to edx_test. We will stick to the 12 transformations of edx_train ratings and run them on edx_test in a for loop. The next table shows RMSEs on edx_test sorted in ascending order.

Time_Trend_on_edx_train	R2_on_edx_train	Fitting_Degree_on_edx_train	RMSE_on_edx_test
Hourly av. of ratings	0.1154066	0.3601110	1.017180
Daily av. of ratings	0.0190400	0.1462697	1.051406
Weekly av. of ratings	0.0070829	0.0892125	1.056904
Monthly av. of ratings	0.0048071	0.0734956	1.057997
Daily av. splined with leave-one-out CV	0.0044471	0.0638053	1.058228
Hourly av. splined with leave-one-out CV	0.0041705	0.0648504	1.058361
Monthly av. splined with leave-one-out CV	0.0033157	0.0604158	1.058781
Weekly av. splined with leave-one-out CV	0.0029473	0.0620303	1.058999
Hourly av. splined with 10 DF	0.0027632	0.0536499	1.059108
Weekly av. splined with 10 DF	0.0023834	0.0617195	1.059341
Monthly av. splined with 6 DF	0.0022433	0.0534855	1.059375
Daily av. splined with 10 DF	0.0022906	0.0603577	1.059404

There is an inverse relationship between R2 on edx_train and RMSE on edx_test.

Hourly averages remain on top when their performance if evaluated **in terms of RMSE on edx_test**. There is no apparent overfitting on edx_train, at least not within the scope of this exploratory exercise where time effect is evaluated in isolation, without interaction with other effects in a global model.

The second best is daily averages.

For the sake of comparison, let's compute the RMSE on edx_test for the baseline model, i.e. the model using the average of edx_train ratings to predict edx_test ratings.

Method_for_RMSE_on_edx_test	RMSE_on_edx_test
Baseline Model (mean of edx_train ratings applied to edx_test)	1.060625

We see that there is a non negligible difference in performance between the baseline model and the time effect model with hourly averages, even if the RMSE remains above 1 with the hourly averages. There is also a difference between the performance of the baseline model and the performance of the daily averages, although the difference is sensibly smaller.

4. Closing Subsection on Time Effect: Interpretation, Avenues of Research and Conclusion

On the tables above, we can see that hourly averages of edx_train ratings outperform other time trends in terms of R2 on edx_train. The second best is daily averages of edx_train ratings.

We can also state that there is no apparent overfitting with hourly averages since they perform quantitatively well not only on edx_train but also on edx_test. When evaluated on edx_test, the second best remains also unchanged, i.e. daily averages of edx_train ratings.

Nevertheless, it should be clear that this test has been done -ceteris paribus-, i.e. without taking into account interaction with other effects. But movie effect and time effect could offset each other, or user effect and time effect. More generally, hourly averages performance should be evaluated in the framework of a global model, of a multi-effect model, where time effect could engage with the other effects. It shall be done in the global analysis and modeling section.

In addition, interpretation could also take another direction. The very concept of time effect has been used in a massive way, in a monolithic way. But **what hides actually behind the very notion of "time effect"?**

Could the visual evidence about time effect be partially due to some users quitting the rating system and new users entering the system with different rating patterns (on an average)? In such a case, visual evidence of time effect would also be part of user effect or it could be considered as cross time-user effect.

The same holds with new movies being released, which can e.g. change the breakdown into genres over time.

Another subcomponent of time effect could be "experience" effect: people's rating patterns can mutate over time because they "become a harsher critic over time" (see e.g. http://blog.echen.me/2011/10/24/winning-the-netflix-prize-a-summary/). Once again that would be a mix of effects, i.e. a mix of time effect and user effect.

Rating behavioral patterns could also be linked to age irrespective of "experience": irrespective of time elapsed since their first rating, users' behavioral patterns could depend on age. Psychological and sociological research could help in that direction.

Ratings of one movie can also follow its own time pattern because e.g. users who are eager to rate a movie soon after it is released could be more enthusiastic than the "average rater". In that respect, it has been suggested, in the reference mentioned two paragraphs hereinabove, to use "the number of days since the movie's first rating by anyone". But one can immediately think of a caveat: cannot we have "old" movies entering the rating system "long" after their release date, especially at the beginning of the rating system but even also later? In such cases, newness of a movie in the recommendation system might have to be distinguished in some cases from newness of a movie based on its release date (release dates are included in the variable "title" in edx).

Ratings could also be partially linked to short cycles in users' mood: Saturday night's ratings might tend to be different, on an average, from Sunday evening's ratings…

These are just a few possible subcomponents of time effect. Many more possible subcomponents can be thought of.

As a conclusion, **hourly averages outperform other time trends in a one-effect contest, followed by daily averages, in terms of R2 on edx_train or RMSE on edx_test**. Numerous avenues of research remain open and might be promising but they are beyond both the scope of this project and available resources. Further treatment will be concentrated, in the global analysis and modeling section, on the two "best performing" time trend candidates: in that section, we will **evaluate how hourly averages and daily averages engage with other types of effects in a global model**.

D. Is the Variable Genres Relevant as an Independent Variable, as a Predictor?

1. The Variable Genres as It Stands in edx_train

How many genres do we have in edx_train?

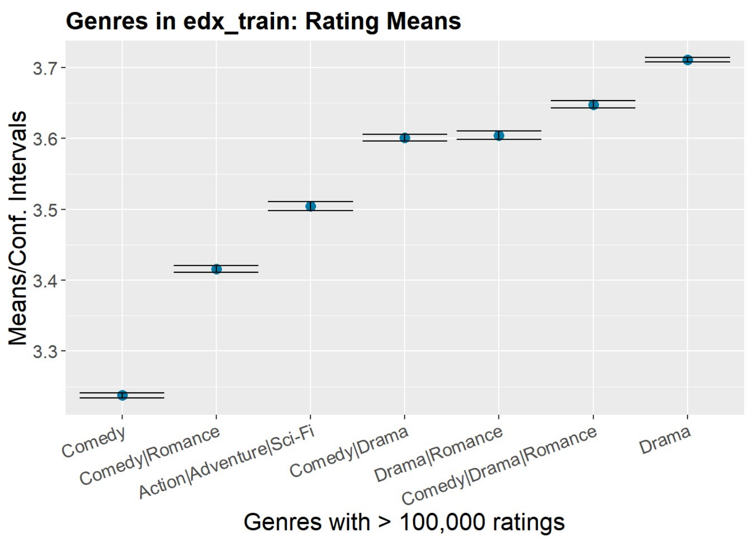
Number_of_Genres_in_edx_train
797

a. Visual Evidence of Genres Effect

Let’s visualize averages of ratings per genre to answer one question: is the variable genres a potential candidate as a predictor (independent variable)? For visual clarity, let’s limit the graph to genres with more than 100,000 ratings.

By the way, in “genres effect”, “genres” is used in the plural because there are often two or more genres corresponding to one movie, such as in the category “Comedy|Romance” (see hereunder).

Means per genre will be represented with estimates of confidence intervals. Confidence intervals have been estimated by means plus or minus standard errors multiplied by 2, standard errors being the standard deviations divided by the number of observations. Hypothesis made: means are normally distributed, which is probably true except for genres with very few observations.



There is some variation among genres in terms of rating means. Estimates of confidence intervals being rather narrow, at least for the most populated genres, is also encouraging. Let’s now compute RMSE on edx_test.

b. Performance Evidence of the Genres Effect on edx_test in Terms of RMSE

Let’s compute the RMSE with genres effect on edx_test, combining neither with time effect, nor with movie effect nor with user effect. Let’s compare the RMSE from this simple genres effect model with the RMSE from the baseline model.

Model_Trained_on_edx_train_and_Tested_on_edx_test	RMSE_on_edx_test
Baseline Model (no independent variable, just mean(edx_train$rating)	1.060625
Genres Effect Model	1.018413

From the table above, we learn that **the genres effect model improves the RMSE on edx_test in the same way, quantitatively, as the time effect represented by hourly averages of ratings**. We still have to **check**, in the global analysis and modeling section, whether the genres effect combines with other effects in a **globally beneficial way** in terms of RMSE on edx_test.

Furthermore, in the variable genres, we often have composite genres (“Comedy|Romance”, etc.) and there are many of them. Consequently, let’s explore another concept, i.e. basic genres (“Comedy”, “Romance”).

2. Basic Genres

a. Building up Basic Genres and Visualizing Distribution of Ratings per Basic Genre

Let’s split composite genres into basic genres and represent the distribution of ratings in edx_train per basic genre, keeping in mind a caveat: the same rating comes back several times if the original genre is a composite genre (“Comedy|Romance”, etc.)! Consequently, there is partial overrepresentation!

basic_genres	Number_of_Ratings_in_edx_train
Drama	1955517
Comedy	1770968
Action	1279986
Thriller	1163244
Adventure	954134
Romance	854968
Sci-Fi	670050
Crime	664115
Fantasy	462753
Children	368310
Horror	346371
Mystery	284029
War	255822
Animation	233405
Musical	216629
Western	94776
Film-Noir	59069
Documentary	46590
IMAX	4088
(no genres listed)	3

b. Some Cleaning

There is a genre defined as “(no genres listed)”, with 3 occurrences. Let’s have a look.

movieId	rating	title	basic_genres
8606	5.0	Pull My Daisy (1958)	(no genres listed)
8606	2.0	Pull My Daisy (1958)	(no genres listed)
8606	3.5	Pull My Daisy (1958)	(no genres listed)

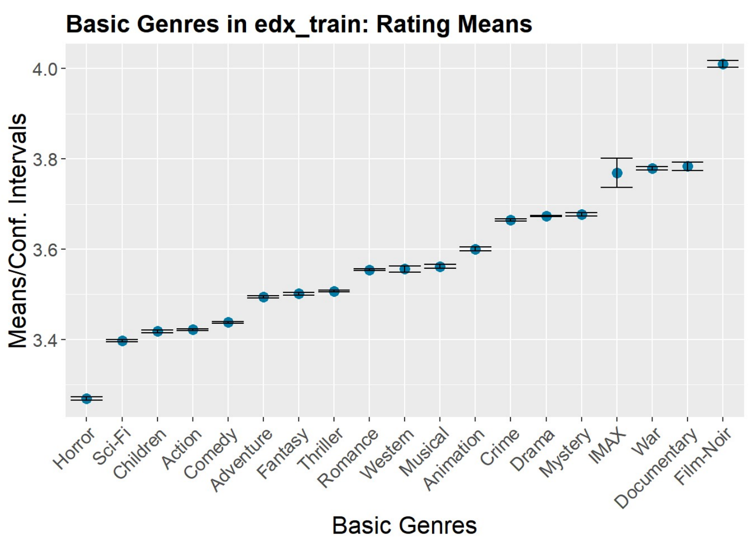
Let’s do some data cleaning. There is only one movie in that basic genre and I would not like that peculiarity to impact some groupings of basic genres later on. We cannot simply drop that movie because it might be present in the validation set and we don’t know it. After some research, I have added that movie to “Documentary” (for motivation of cinematographic choice, see e.g. http://www.allocine.fr/film/fichefilm_gen_cfilm=26892.html). Consequently, **the “(no genres listed)” disappears from the basic genres**.

Should we do the same with the last but one basic genre, IMAX? Actually, it is in a different situation with more than 4,000 ratings as indicated in the last but one table and more movies as mentioned in the next table. Let’s maintain this basic genre.

Number_of_Movies_in_IMAX
29

c. Are Basic Genres Relevant as an Independent Variable, as a Predictor?

Let's first visualize rating variation per basic genre. The next graph gives rating means per basic genre, ranked according to mean magnitude, and estimates of confidence intervals.



There is some variation in means of edx_train ratings, which could possibly contribute to lower RMSE on edx_test. Narrowness of estimated confidence intervals is also supportive of that idea.

Because of some overlapping, we can't use basic genres in a direct way as we did with the variable genres. We could try regressions with numerous independent variables but this is beyond resources. We will just try and combine the basic genres into a smaller number of clusters.

3. Clusters of Basic Genres

a. Some Wrangling

Let's do some wrangling. First, let's establish a link between each movie and the basic genres it belongs to. We only keep one row for each combination of one movieId with one basic genre. Readers interested in that exercise are welcome to read the code whose output is shortly summarized here.

Then let's rearrange data with two technical purposes:

- having each movie only on one row,
- having each basic genre as a separate column.

Now, we have one row per movieId. We have 20 columns, one for movieId and 19 for the 19 basic genres. In the 19 columns related to basic genres, we have NA each time a movie does not belong to a basic genre! Let's change the NAs into 0.

For one movieId, on one row, we have as many non-zero digits as the number of the basic genres the movie belongs to. Actually, we are only interested in having 1s instead of a rating simply to indicate the basic genres the movie belongs to. Let's change all non-zero digits into 1.

Now we have an additional taxonomy for movies: each movie is linked to all basic genres it belongs to.

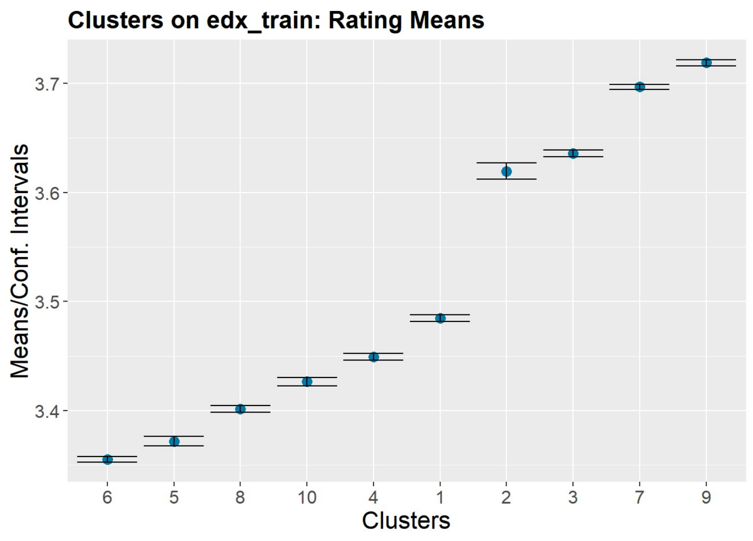
b. Building up Clusters of Basic Genres

Let's build up a limited number of **clusters (10) on the basis of basic genres**. We make results reproducible with set.seed(1). We run 30 iterations because of the variability in results from the kmeans() function.

Analyzing the composition of clusters is beyond resources.

c. Visual Evidence of Clusters Being a Potential Candidate for Predicting Ratings

Let's check for some evidence about cluster effect being a potential candidate for predicting ratings on edx_test. Let's produce a graph with rating means per cluster.



On the graph above, there is **some variation among means of edx_train ratings per cluster**, with rather narrow estimated confidence intervals and estimated confidence intervals that do not intersect each other.

The graph might be a visual indication of the cluster effect being a possible candidate for contributing to lower RMSE on edx_test.

d. Performance Evidence of Clusters Being a Potential Candidate for Predicting Ratings

Let's now calculate the cluster effect on edx_train and test it on edx_test by computing the RMSE on edx_test.

Model_Trained_on_edx_train_and_Tested_on_edx_test	RMSE_on_edx_test
Baseline Model (no independent variable, just mean(edx_train$rating))	1.060625
Genres Effect Model	1.018413
Cluster Effect Model	1.051093

The RMSE on edx_test also indicates a cluster effect, but it is much smaller than the effect of genres as they stand in the variable "genres" from edx_train.

4. Conclusion for Genres, Basic Genres and Clusters of Basic Genres

There is a cluster effect, but the cluster effect appears quantitatively as a second best in terms of RMSE on edx_test in comparison with the genres effect.

The basic genres effect cannot be used directly under the form of rating means since there would be some overlapping between basic genres. The 19 columns/vectors related to basic genres could be used e.g. in multiple regression, be it linear or tree, but this is beyond resources.

Consequently, we'll opt for the genres effect in the global model from the analysis and modeling section … if the genres effect engages well with the time, movie and user effects. If not, we could fall back on the cluster effect.

E. Exploratory Analysis of Movie Effect

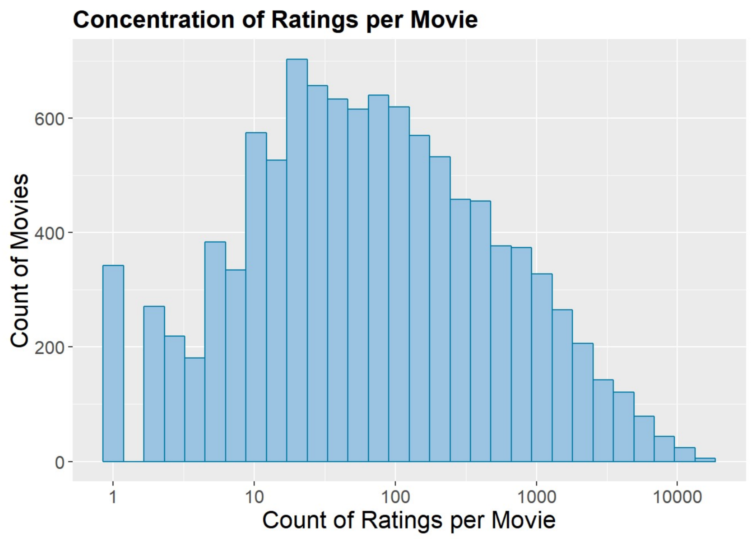
1. Number of Movies in edx

The number of movies is the same in edx_train as in edx (by construction):

Number_of_Movies_in_edx_train
10677

2. Distribution of Ratings per Movie

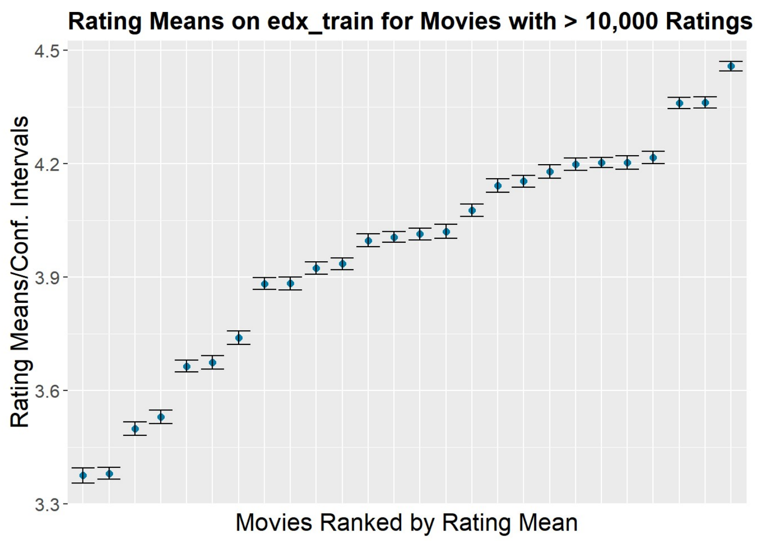
Let’s visualize the distribution of ratings per movie: some movies get many more ratings than others.



Let’s see the titles of the movies with more than 10,000 ratings. We insert these names into a table because they are too long to be indicated on the next graph near the x-axis ticks.

Title	Number_of_Ratings_in_edx_train
Pulp Fiction (1994)	15739
Forrest Gump (1994)	15476
Silence of the Lambs, The (1991)	15261
Jurassic Park (1993)	14596
Shawshank Redemption, The (1994)	13888
Braveheart (1995)	13143
Terminator 2: Judgment Day (1991)	12976
Fugitive, The (1993)	12928
Star Wars: Episode IV - A New Hope (a.k.a. Star Wars) (1977)	12802
Batman (1989)	12178
Apollo 13 (1995)	12161
Toy Story (1995)	12099
Independence Day (a.k.a. ID4) (1996)	11823
Dances with Wolves (1990)	11685
Schindler’s List (1993)	11544
True Lies (1994)	11406
Star Wars: Episode VI - Return of the Jedi (1983)	11257
12 Monkeys (Twelve Monkeys) (1995)	10993
Usual Suspects, The (1995)	10898
Fargo (1996)	10738
Speed (1994)	10624
Aladdin (1992)	10535
Star Wars: Episode V - The Empire Strikes Back (1980)	10419
Matrix, The (1999)	10369
Seven (a.k.a. Se7en) (1995)	10107
American Beauty (1999)	10052

Let’s have a look at a graph with means of edx_train ratings and estimates of confidence intervals for movies with more than 10,000 ratings so that we can get some insight into variation per movie.



The range between the maximum mean and the minimum mean is **relatively broad** and estimates of confidence intervals are rather narrow. Several estimates of confidence intervals intersect each other.

This graph cannot be easily compared with the previous graph dedicated to the genres effect, because they are both partial in terms of ratings covered.

A comparison could be conducted with respect to the cluster effect graph since this last graph covers all edx_train ratings. The range between maximum and minimum rating means is much broader with the movie effect.

3. Conclusion of the Exploratory Analysis of Movie Effect

Movie effect seems a **promising candidate for RMSE on edx_test** on the basis of averages variation and narrowness of estimated confidence intervals (at least for the movies with more than 10,000 ratings).

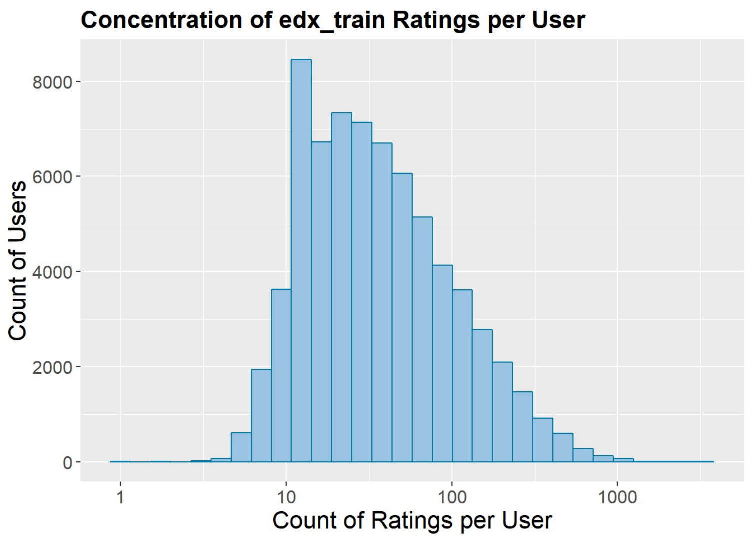
F. Exploratory Analysis of User Effect

1. Number of Users

Number_of_Users
69878

2. Distribution of Ratings per User

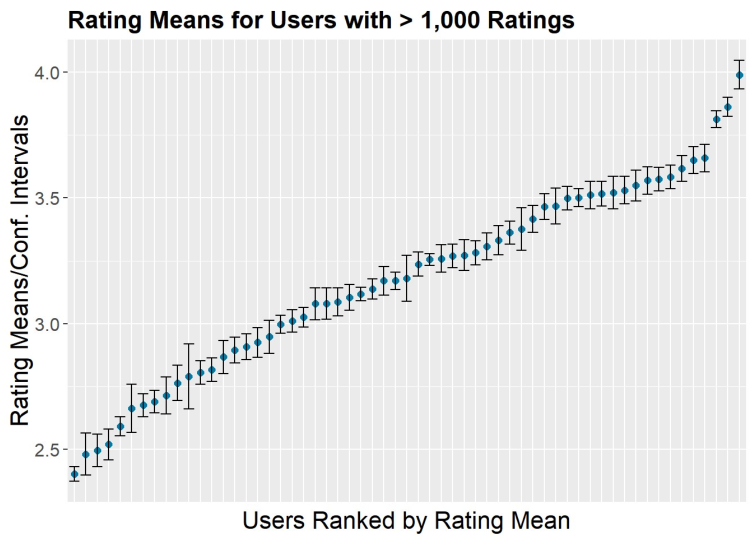
The following histogram of ratings per user shows that some users have issued many more ratings than others.



Let’s have a look at a table of users with more than 1,500 ratings.

userId	Number_of_Ratings_per_User_in_edx_train
59269	3309
67385	3225
14463	2318
68259	2009
27468	2006
19635	1905
3817	1863
58357	1677
63134	1641
27584	1605
6757	1579

Let’s draw a graph of rating means and estimates of confidence intervals for users with more than 1,000 ratings.



The range between the maximum mean and the minimum mean is **rather broad**. Most estimates of confidence intervals intersect other estimates of confidence intervals.

3. Conclusion of the Exploratory Analysis of User Effect

Visually, as indicated by the last graph, user effect seems a **promising candidate for RMSE on edx_test** on the basis of averages variation and narrowness of estimated confidence intervals (at least for users with more than 1,000 ratings, which is an exception).

Let's move to the global analysis and modeling section.

IV. Global Analysis and Modeling Section on edx_train and edx_test

In this section, I need to bundle the different effects into a global model, run it on edx_train and test it on edx_test. There are **four effect candidates** for contributing to lower RMSE on edx_test: hourly averages of ratings, averages per genres (as they stand in the variable genres in edx_train), averages per movieId and averages per userId.

I spontaneously turned to **means per subcategories** (as I had turned to medians in the accuracy version) to avoid working on linear regressions with numerous subcategories such as movieIds (as factors) or userIds on so many observations.

As very clearly explicated in course PH125.8x, regressing ratings on e.g. "as.factor(movieId)" with **the lm() function would be methodologically correct but "computerly" very slow**. The more so with a standard laptop.

In the course, the model with two effects is presented (with other symbols) as

```
RATINGS = M + MOVIE EFFECT + USER EFFECT + ERROR TERM
```

where M is "the 'true' rating for all movies" and the error term represents "independent errors sampled from the same distribution centered at 0".

Least squares have been applied in a way different from the lm() function. An approximation has been calculated in the following way:

- M has been estimated by the general rating mean,
- the movie effects have been estimated by the rating means per movieId minus M
- and the user effects have been estimated by the rating means per userId minus M and minus the movie effects.

I am going to do the same, *mutatis mutandis*, but with four effects: time effect, genres effect, movie effect and user effect. Means are calculated on edx_train and then the model is tested on edx_test.

By the way, in a linear regression, the **order of the independent variables** would not matter. Here, it **sensibly impacts results**, as we will see hereunder. That is why this method makes me rather think of a **cascade** of linear regressions: a first regression being applied to the first independent variable only, then the residuals being regressed on the second variable, etc. This kind of cascade, or hierarchy, of linear regressions has been discussed at length (see e.g. https://besjournals.onlinelibrary.wiley.com/doi/full/10.1046/j.1365-2656.2002.00618.x). Results from such a cascade as well as interpretability are different from those of multiple regression.

More generally, coexistence of different ways of thinking and of different cultures in data science has been largely conducted as explicated in e.g. the excellent paper "Statistical Modeling: The Two Cultures (with comments and a rejoinder by the author)" by Leo Breiman et alia: https://projecteuclid.org/euclid.ss/1009213726 . But these most interesting discussions fall completely outside the scope of this project.

On trying to fix a 4-effect model, I had to answer two questions:

- which kind of averages should I take to represent time effect,
- which order should I apply to independent variables?

Let's examine the 3 global models that I have tested and that bring different answers to these two questions. Let's get started with two global models that did not match expectations for different reasons.

A. Global Models Unsuccessful in Attaining both First and Second Objectives on edx_train and edx_test

The first and second objectives are at stake here: obtaining an RMSE inferior to 0.8775 on edx_test and getting more insight into effects.

The third objective is no relevant criterion in this section about global analysis and designing models. Indeed, the third objective is a pedagogical one for me: benefiting from this project as a broad learning and practicing experience, which is already in a stage of quasi-fulfillment from my point of view.

1. Hourly Averages in First Position on edx_train and edx_test

My first try was with the following combination of answers to the two questions above: time effect represented by hourly averages and independent variables ranked in the order time-genres-movie-user.

The final RMSE on edx_test is well above the target of 0.8775. Consequently, I have dropped this model. I am not going to pour down the corresponding code which would overburden this project and look like sheer redundancy with respect to the code that will eventually fit edx_test.

REJECTED_Global_Model	**RMSE_on_edx_test**
Baseline Model	1,06062549225372
Time Effect Model with HOURLY AVERAGES OF RATINGS	**1,01717997680373**
Time and Genres Effects Model	0,982468834263581
Time Genres and Movie Effects Model	0,92308721450055
Time Genres Movie and User Effects Model	**0,887461464673567**

Let's analyze results from this global model.

First, time effect is represented here by hourly averages of edx_train ratings. Its impact on edx_test RMSE can be quantified as the difference in RMSE between the Time Effect Model and the Baseline Model. That difference is non negligible here. It is the same as in the stand-alone exercise of exploratory analysis, which is normal: time effect being in first position, it automatically reproduces its previous performance level.

Second, genres effect. Genres effect has been added to the Time Effect Model, transforming it into a Time and Genres Effects Model. The quantitative influence of the genres effect is measured here by the difference between the RMSE from the Time and Genres Effects Model and the RMSE from the Time Effect Model: this quantitative influence is slightly reduced in comparison with the exploratory analysis where the genres effect was tested on its own. Indeed, in such a cascade system (or hierarchical system), the quantitative influence of a variable strongly depends on its position in the order of the cascade, as we will see it much more clearly in the second global model, where time and genres effects positioned at the end of the cascade will have their quantitative influence shrunk to almost nothing.

Third, movie effect exercises the largest quantitative impact.

Fourth, user effect has a quantitative influence almost the same size as the genres effect: this does not match visual evidence collected in the exploratory analysis, which was, it must be said, partial. I am willing to see how much the user effect can contribute to lower RMSE in other models.

Fifth, **the main objective is not reached**: final RMSE (on the last line from the table above) is **far above the target of 0.8775!** The regularization procedure, which can bring a little help as shown hereunder, is not included here.

I would broach the following interpretation.

Placed in first position, time effect represented by hourly averages seems to develop a ravenous appetite for rating variation, taking a rather big chunk from rating variation, just as in exploratory analysis.

This is not so surprising since hourly averages of ratings are very flexible, with not far from 100,000 hourly averages of ratings, i.e. sensibly more than user averages and many more than movie averages.

Moreover, as pinpointed in the closing subsection from the exploratory analysis of time effect, the very composition of what has been quantified as "time effect" seems heterogeneous, possibly encompassing parts of movie, user and genres effects.

Furthermore, time effect, not only takes away rather much rating variation, but it also seems to do so in a inefficacious way, possibly disrupting movie and user effects and apparently reducing the whole efficacy of the model. It is not just a transfer from e.g. movie and user effects towards time effect, there also seems to be disruption. This view could be partially supported by looking at the second global model in the next subsection:

the fourth line (headers included) from the results table of the second model shows that the movie effect and the user effect in first and second position respectively can produce a much better RMSE result than in the first model where they are preceded by the time and genres effects. In the second model, the movie and user effects alone produce an RMSE much lower than 0.8775.

Actually, genres effect could be “incriminated” as well since e.g. there is some intersection between genres effect and movie effect. But we will see hereunder that the third global model performs rather well with the same genres effect in second position once time effect has been “downgraded” from hourly to daily averages…

Since I do not have any expedient solution to the problem and since this global model does not attain the first objective, I feel obliged to discard this model.

2. Hourly Averages in Last Position on edx_train and edx_test

Then I ran hourly averages with the other effects in the order movie-user-genres-time. Here are the results in terms of RMSE on edx_test.

REJECTED_Global_Model	**RMSE_on_edx_test**
Baseline Model	1,06062549225372
Movie Effect Model	**0,944387933959757**
Movie and User Effects Model	0,869418019952897
Movie User and Genres Effects Model	0,869066740408623
Movie User Genres and Time (HOURLY AVERAGES OF RATINGS) Effects Model	**0,86758614613947**

Let me pinpoint five aspects from the results above.

First, movie effect and user effect have been boosted and play a role that is quantitatively in line with expectations raised by the exploratory analysis.

Second, genres effect and time effect have shrunk. Movie effect and user effect have taken away a lot of rating variation, rather efficiently, and the quantitative impact from genres effect and time effect does not match prospects from exploratory analysis, especially for genres!

Third, the final RMSE (on the last line from the table above) is well below 0.8775.

Fourth, by the way, the table above shows that it is possible to slightly beat the 2-effect model with only movie and user effects (fourth line from the table above, headers included) with a 4-effect model integrating also genres and time effects (last line). Regularization is not taken into account in this comparison.

Fifth, **the main objective is reached**, i.e. an RMSE lower than 0.8775, but **the second objective is not**, i.e. bringing some additional insight into effects by letting genres and time effects display their potential without, of course, endangering our reaching the first objective. Consequently, I am not going to content myself with this model.

Just as with the first rejected model, I am not going to publish the corresponding code to not overburden Recommendation_Report.Rmd (neither did I in Recommendation_Script.R for the same reasons).

3. Way Forward on edx_train and edx_test

I wish to develop another global model fulfilling the first and second objectives. To do that, I will **move back to the second best for time effect, i.e. daily averages**.

I am going to maintain the genres effect, not replacing it with the cluster effect if it is not necessary.

B. Global Model Successfully Accomplishing Objectives One and Two on edx_train and edx_test

1. Time (daily averages), Genres, Movie and User Effects on edx_train and edx_test

Let’s run daily averages of ratings, genres effect, movie effect and user effect and have a look at the new results.

Method_Run_on_edx_train_and_Tested_on_edx_test	**RMSE_on_edx_test**
Baseline Model (no independent variable, just mean(edx_train$rating))	1.0606255
Time Effect Model (daily averages)	1.0514060

Method_Run_on_edx_train_and_Tested_on_edx_test	**RMSE_on_edx_test**
Time and Genres Effects Model	1.0101202
Time Genres and Movie Effects Model	0.9382468
Time Genres Movie and User Effects Model	**0.8717513**

What are the results from the third global model?

First, time effect is slight. On the basis of the rather long analysis of time effects, I would consider this quantification of time effect as a reasonable mix between two preoccupations:

- on the one hand, acknowledging some visual, statistical and performance evidence of time effect;
- on the other hand, limiting the impact of that effect, because of a probably large intersection with movie, user and maybe genres effects, as already adumbrated hereinabove at the end of the exploratory analysis of time effect, and because of a possible disruptive effect.

Further investigation is beyond the scope of this project and beyond resources.

Second, in parallel with time effect contributing less to lower RMSE, genres, movie and user effects contribute more.

Third, genres effect, in particular, delivers a non negligible contribution to lower RMSE, in line with exploratory analysis.

Fourth, movie effect is substantial, in line with exploratory analysis as well.

Fifth, user effect has contributed the largest increase to lower RMSE. It is in line with visual evidence from exploratory analysis.

Sixth, this model is slightly less effective than the second model in terms of RMSE but it **meets the first and the second objectives**: an RMSE well below 0.8775 and more insight into rating variation with four effects in lieu of two.

Consequently, I am going to stick to this model: time effect with daily averages, genres effect with the genres as they stand in the variable genres in edx_train, movie effect and user effect.

I will just try and boost performance a little bit by following the PH125.8x course in applying a regularization, or penalization, procedure.

2. Regularization on edx_train and edx_test

Variability tends to be higher in smaller groups and this can be detrimental to predicting performance. **Penalizing higher variability in smaller subcategories (e.g. users with very few ratings)** can be achieved by adding a factor lambda to the denominator of the means. This factor is proportionately much more impactful for smaller groups than for larger groups.

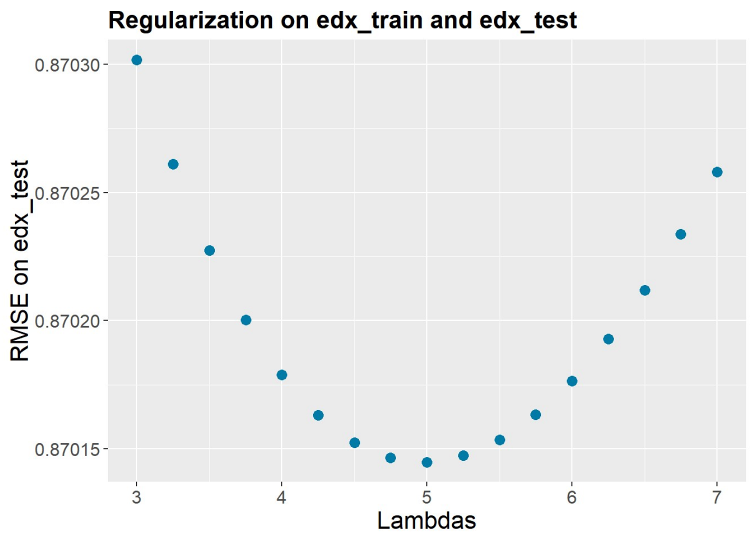
The regularization procedure will be **applied to the 4-effect model** at once. The **same lambda will be used for all averages**, be it for time averages, genres averages, movie averages or user averages of edx_train ratings.

A **cross-validation** process will be launched on edx_train and edx_test. Regularized time, genres, movie and user effects will be calculated on edx_train using different lambda values from a predetermined sequence; with these regularized effects, RMSEs will be computed on edx_test and the best RMSE will determine the optimal lambda and the optimal vector of predicted ratings.

In the following section about the final model, the **same optimal lambda will be used on edx and the validation set**.

The cross-validation process was first launched on a sequence of lambda values ranging from 0 to 20 with increments of 1. Around the provisional optimum, the cross-validation procedure has been refined on a narrower range with increments of 0.25.

The next graph shows the optimal value for lambda. The next table gives the RMSE on edx_test after regularization.



On the graph above, we see that the optimal value of lambda is 5.

The regularization process has delivered a very **slight decrease in the RMSE on edx_test**, as the comparative table hereunder shows it.

Accepted_Model_on_edx_train_and_edx_test	RMSE_on_edx_test
Baseline Model (no independent variable, just mean(edx_train$rating))	1.0606255
Time Effect Model (daily averages)	1.0514060
Time and Genres Effects Model	1.0101202
Time Genres and Movie Effects Model	0.9382468
Time Genres Movie and User Effects Model	0.8717513
Time Genres Movie and User Effects + Regularization Model	**0.8701448**

Without any quantitative difference on the final RMSE, regularization could have been applied stepwise: first on the Time Effect Model alone (third line from the table above, headers included) with one specific lambda, then on the Time and Genres Effects Model (4th line) with another lambda (maybe with identical value), then on the Time Genres and Movie Effects Model (5th line) with still another lambda and eventually on the Time Genres Movie and User Effects Model (6th line) with the lambda we already know to obtain the same 7th line.

This stepwise approach has not been followed for two reasons: it would have overburdened the code and the final RMSE would be strictly the same.

Let’s switch to a little refinement.

3. Coercion of Out-of-range Predicted Ratings on edx_test

I have added a small correction. There is no rating outside the 0.5-5 range in edx. Consequently predicted ratings have been coerced to 0.5 or 5 according to their being respectively below or above the 0.5-5 range.

Here is the result.

Accepted_Model_on_edx_train_and_edx_test	RMSE_on_edx_test
Baseline Model (no independent variable, just mean(edx_train$rating))	1.0606255
Time Effect Model (daily averages)	1.0514060
Time and Genres Effects Model	1.0101202
Time Genres and Movie Effects Model	0.9382468
Time Genres Movie and User Effects Model	0.8717513
Time Genres Movie and User Effects + Regularization Model	0.8701448

Accepted_Model_on_edx_train_and_edx_test	**RMSE_on_edx_test**
Time Genres Movie User Effects + Regularization + Coercion Model	**0.8700332**

The coercion process has a **minute quantitative impact**.

4. Conclusion about the Accepted Model on edx_train and edx_test

It is possible to run **four effects (time, genres, movie and user)** on edx_train and to obtain on edx_test an **RMSE lower than 0.8775 and a reasonably balanced breakdown between four effects**.

Time effect has been downgraded from hourly averages to **daily averages**. It exercises a quantitative role that is small but in line with the long analysis of that effect.

Genres effect makes a non negligible impact on RMSE, compatible with exploratory analysis; it seems natural that ratings are partially related to genres.

Movie effect and user effect exercise more important quantitative roles, in line with anticipations.

Regularization and coercion impacts are quantitatively marginal.

We can now run this model on edx and validate it on the validation set, with the lambda value issued from the cross-validation process on edx_train and edx_test, i.e. 5.

V. Final Model Run on edx and Validated on the Validation Set

I am going to run the chosen global model on edx exactly as I did on edx_train. edx has been chosen to use all available information. The four independent variables (predictors) are

- the averages of ratings with a daily periodicity,
- the averages of ratings per genre,
- the averages of ratings per movie
- and the averages of ratings per user.

Lambda will have the value already determined on edx_train and edx_test, i.e. 5.

From the validation set, the only piece of information used to compute predicted ratings will consist of the subcategories from timestamp, genres, movieId and userId, i.e. daily dates originating from timestamp, the genres, the movieIds and the userIds. These subcategories will be the identifiers permitting the transfer to the validation set of time, genres, movie and user effects calculated on edx.

Ratings from the validation set will only be used to calculate RMSE on the validation set. **Ratings from the validation set shall never be used to calculate predicted ratings on edx_test**.

Results in terms of RMSE will be given in the next section.

VI. Final Results on edx and the Validation Set

Here are the final results.

Final_Model_on_edx_and_the_Validation_Set	**RMSE_on_Validation_Set**
Baseline Model (no independent variable, just mean(edx$rating)	1.0612018
Time Effect (daily averages) Model	1.0518462
Time and Genres Effects Model	1.0101402
Time Genres and Movie Effects Model	0.9377932
Time Genres Movie and User Effects Model	0.8676056
Time Genres Movie and User Effects + Regularization Model	0.8670962
Time Genres Movie and User Effects + Regularization + Coercion Model	**0.8669750**

THE FINAL RMSE ON THE VALIDATION SET IS 0.8669750.

VII. Conclusion

As a conclusion, the three objectives have been fully met.

First, the final RMSE on the validation set is 0.8669750, i.e. well below 0.8775.

Second, the final 4-effect model has shed more insight into variation of ratings. There is a small time effect, quantified in a balanced way. Genres effect is non negligible and in line with exploratory analysis. Movie effect and user effect are more substantial and have fulfilled their prospects from exploratory analysis as well.

Third, from a pedagogical point of view, I have appreciated and enjoyed benefiting from innumerable opportunities to learn and practice exploratory analysis, visualization, statistics, time series analysis, wrangling, clustering, analysis, modeling, R and R Markdown.

Many avenues of research have been contemplated in methodology, data management and data analysis … and might be explored in other data science projects.

Appendix

A. RAM Management

RAM management has been a constant challenge during this project. After reading a lot about it, I managed to use a few practical tricks towards RAM management. Among them, let me mention:

- cleaning workspace,
- cleaning the console pane,
- clearing all plots,
- removing objects such as data frames and graphs as soon as they were no longer useful,
- removing columns from data frames.

This last “trick”, for instance, has proved decisive to avoid blue screens when dealing with edx as a whole on my PC.

Furthermore I managed to speed up processes e.g. by using functions from the tidyverse instead of base R, such as the functions from the join family instead of the merge() function.

B. Layout Management

Professional communication needs among others readability and clarity. I have tried to cater for both aspects by working among others on layout, without trying to rival book publishers. Let me indicate a few points:

- table of contents inserted in R Markdown,
- title levels in R Markdown,
- body text in R Markdown (double justification and font size),
- graphs with R (line and point size and shape, colors (see hereunder)) and R Markdown (dimensions, position, font size differentiated by titles, axis labels and axis tick labels),
- tables with kableExtra (column width, line height, bordered, font size, bold),
- maximal capitalization of text titles, graph titles, axis labels and axis tick labels (of course in R).

I have picked up a set of colors from https://www.colorhexa.com/007ba7 , trying to combine them harmoniously and professionally. Almost all colored parts are colored with one of these colors:

- Cerulean blue #007ba7,
- Pale cerulean #9bc4e2,
- Paris green #50c878,
- Harvard crimson #c90016.

****