Movielens - v5

Francois PLACE

2020/09/09

Introduction

GroupLens Research has collected and proposes several sizes of datasets relating to movies and ratings. The main target is to study and develop a movie recommender system, to facilitate choices. “Movielens”, from GroupLens, is a web site which helps people to find movies to watch.

The Movielens 10M dataset, used in our project, contains ten millions ratings, for about 10000 movies and 72000 users. The dataset is divided into training and test set. The training set holds 90% of the total dataset.

In the training set, we have 10677 movies, for 69878 users.

The goal of our project is to develop a model, based on datas in the training set, which is able to predict a rating, for a specific film and identified user.

In the end of the project, the model is used with the test set, and performance is evaluate with RMSE (residual mean squared error).

There are five main parts in the following, which represents the principal tasks performed for the project :

- Data cleaning
- Data exploration (general information, graphical representations, analysis of stand alone effects)
- First results
- Models evaluation
- Results section

R code is below.

```
# Libraries used and training set preparation
#
# LIBRARIES
library(dplyr)
library(caret)
library(lubridate) #date and time manipulation
library(ggplot2)
library(stringr)

# LOADING DATASET
load('edx.Rda') #previously saved with function save, because re run is longer

#edx_base, a copy of edx, is used in the following
edx_base <- edx
```

Analysis Section

Data cleaning

Firstly, we need to know if some datas are missing in edx dataset. We focus on cells in the data frame, which could be empty or filled with 'NA'. In a second time, we convert timestamp in readable date and time format.

We use this code :

```
# DATA CLEANING
##seeking for 'NA' in edx dataset
test1 <- ifelse(edx == 'NA', 1, 0)
##seeking for blank cells in edx dataset
test2 <- ifelse(edx == "" | edx == " ", 1, 0)

sum(test1)
```

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

```
sum(test2) #zero in both cases indicates no "NA" and no blank cells
```

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

```
##add date and time in readable format in edx_base
dt <- as_datetime(edx_base$timestamp)
edx_base <- edx_base %>% mutate(dt)
```

There are no 'NA' and no empty cells in the original training dataset (edx).

Data exploration

```
## GENERAL FIGURES

###number of rows in edx_base
edx_n <- edx_base %>% nrow()

###number of movies rated in edx_base
edx_m <- edx_base %>% group_by(movieId) %>%
  summarize(n()) %>% nrow()

###number of users in edx_base
edx_u <- edx_base %>% group_by(userId) %>%
  summarize(n()) %>% nrow()

###average rating
edx_ra <- mean(edx_base$rating)

###period of time covered by edx_base
oldest <- min(edx_base$timestamp)
oldest <- as_datetime(oldest)
```

```
mrecent <- max(edx_base$timestamp)
mrecent <- as_datetime(mrecent)

duration <- interval(oldest, mrecent)

###genders : number of classifications (multi-genders items) in the dataset
edx_g <- edx_base %>% group_by(genres) %>%
  summarize(n()) %>% nrow()
```

```
###printing results
#number of rows
edx_n
```

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

```
#number of movies
edx_m
```

```
## [1] 10677
```

```
#number of users
edx_u
```

```
## [1] 69878
```

```
#average rating
edx_ra
```

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

```
#period of time
duration
```

```
## [1] 1995-01-09 11:46:49 UTC--2009-01-05 05:02:16 UTC
```

```
#multi-gender items
edx_g
```

```
## [1] 797
```

Data exploration - graphical elements

(the goal is to examine the distribution of variables, or to detect a trend)

- rating distribution
- users vs genders
- new movies every year
- time effect on rating for a particular movie

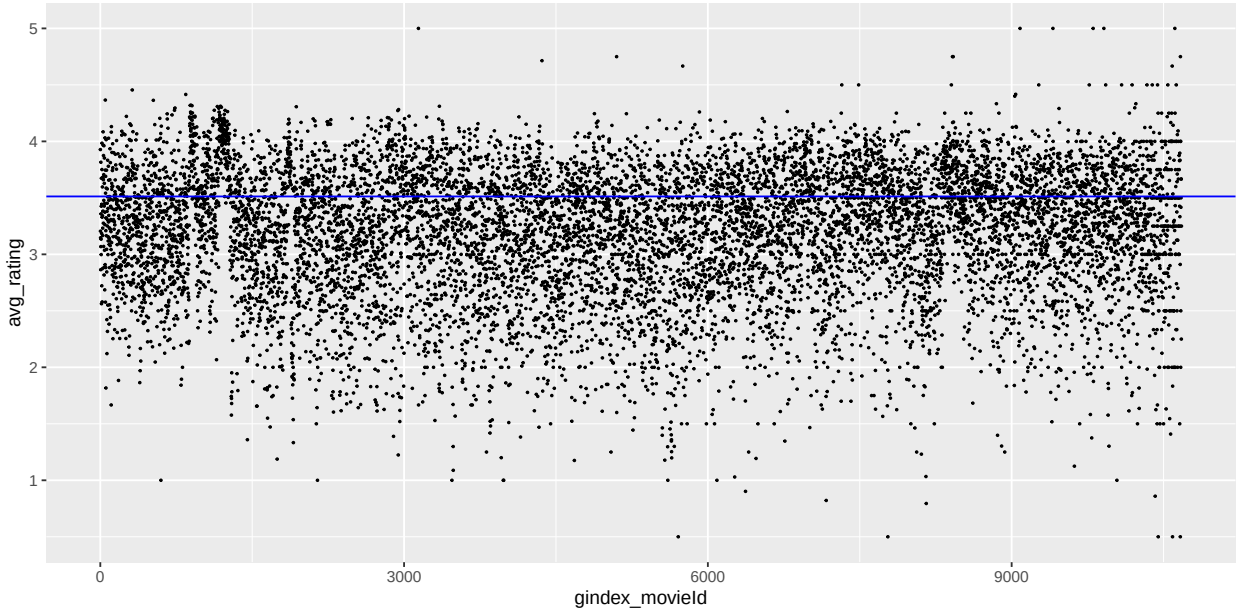
Graph1

```r
###movies vs rating : the idea is to examine the rating distribution, by movie
gr1 <- edx_base %>% group_by(movieId) %>%
  summarize(avg_rating = mean(rating)) %>% mutate()

gindex_movieId <- seq(1,edx_m,1)         #creates a new index because movieId
gr1 <- data.frame(gindex_movieId, gr1)   #is not continuous (edx_m = total number of movies )

grt <- gr1 %>%
  ggplot(aes(x=gindex_movieId, y=avg_rating)) +
  geom_point(size=0.3) +
  geom_hline(yintercept=edx_ra, colour = "blue") #draw a blue line for average rating
```

```r
grt #draw graph
```



We can notice that the range of ratings is large. Our first idea is trying to quantify the relation movie <-> rating.

Graph2

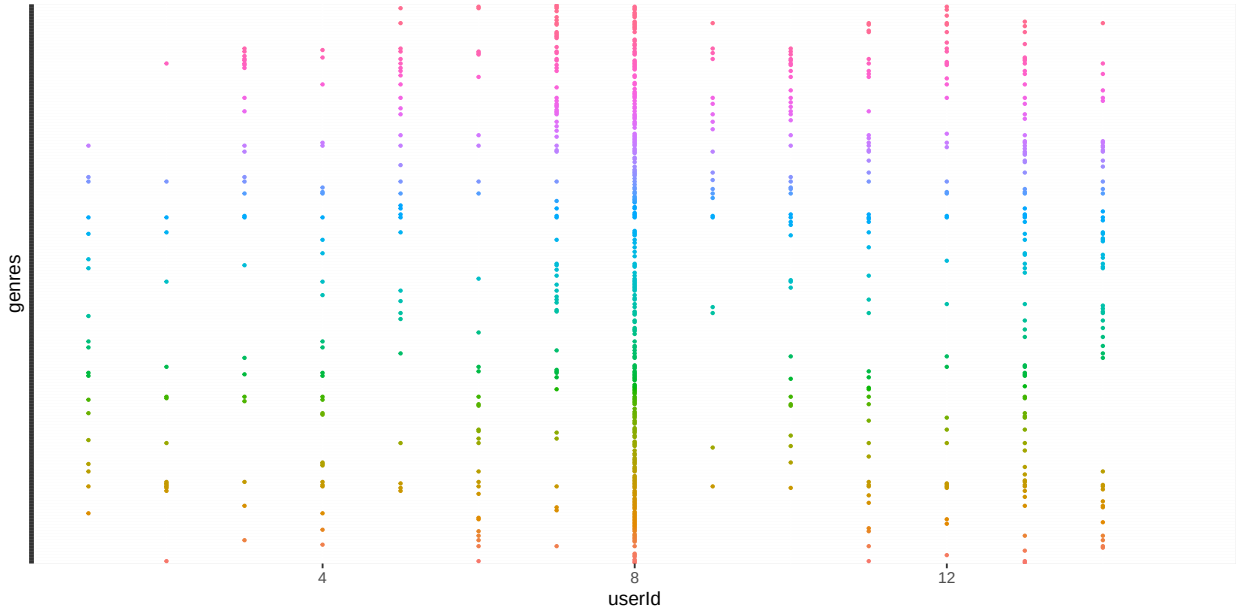
```r
###users vs gender : the idea is to see if users have particular
#preferences concerning movie genders
gr2 <- edx_base %>% group_by(userId, genres) %>% #group by user and by gender(s)
  summarize() %>% mutate()

gr2s <- gr2[1:1500,]  #test only on 1500 first rows in the dataset

grt <- gr2s %>%
  ggplot(aes(x=userId, y=genres, color=genres)) + #every color represents a gender category
  scale_x_continuous(limits = c(1,15)) +          #print only the first 15 users
  theme(legend.position='none') +                 #delete legend
  theme(axis.text.y = element_blank()) +          #delete description of every gender
  geom_point(size=0.5)
```

```
grt #draw graph
```

```
## Warning: Removed 773 rows containing missing values (geom_point).
```



It seems that gender(s) preferred by users are scattered. Maybe gender could be an interesting predictor for our model.

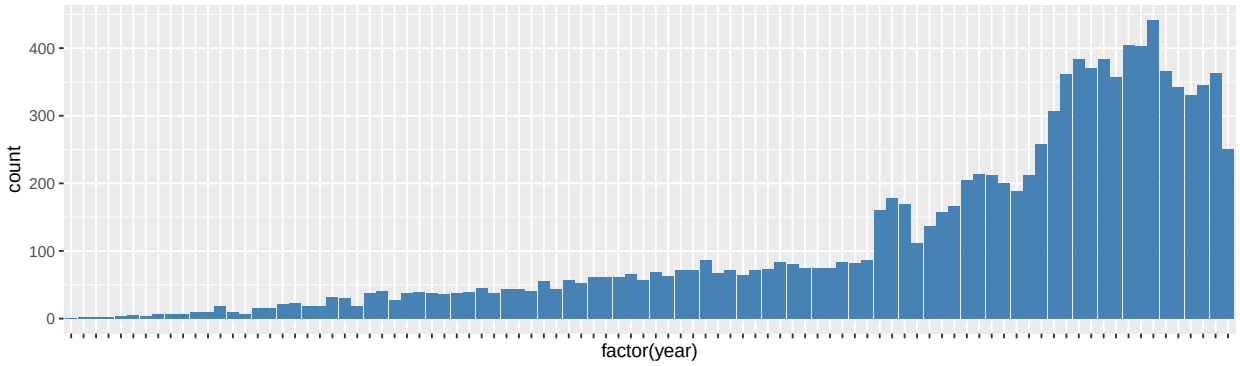
Graph3

```
###4.2.3/new movies by years
spl <- str_extract(edx_base$title, "\\(\\d{4}\\)")     #year is extracted with brackets
year <- str_extract(spl, "\\d{4}")                     #brackets are deleted

gr3 <- edx_base %>% mutate(year) %>% select(title, year)   #year is added to gr3
gr3b <- gr3 %>% group_by(year, title) %>%
  summarize() %>% mutate()                          #the final dataframe used to plot is gr3b

grt <- gr3b %>% ggplot(aes(x = factor(year))) +
  theme(axis.text.x = element_blank()) +
#geom_bar is useful to plot categories (here : every year)
  geom_bar(fill="steelblue")
```

```
grt #draw graph
```



We plot this graph to have an overview of new products by year.

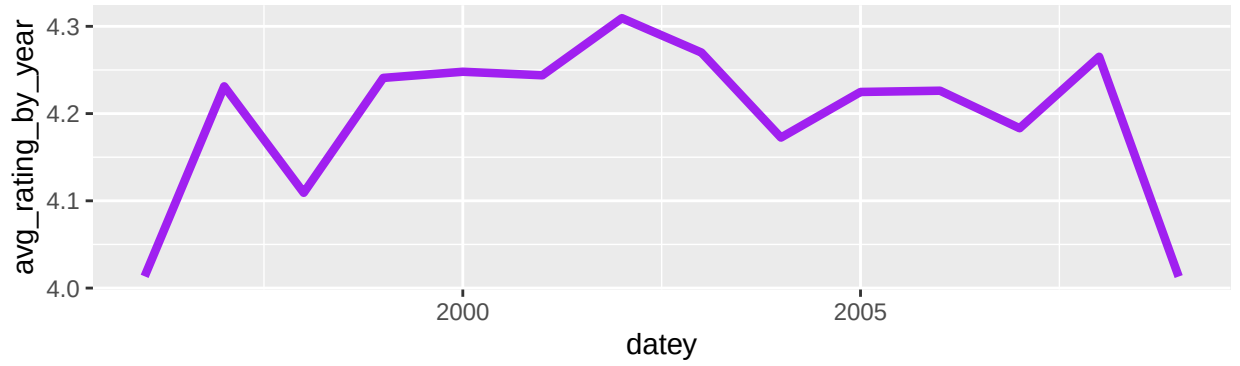
Graph4

```
###time effect on rating, for a specific movie - test with movieId 296
datey <- year(edx_base$dt) #extract year of ratings
test <- edx_base %>% mutate(datey) %>% filter (movieId==296)

gr4 <- test %>% group_by(datey) %>%
  summarize(avg_rating_by_year = mean(rating)) %>% mutate()

grt <- gr4 %>%
  ggplot(aes(x=datey, y=avg_rating_by_year)) +
  geom_line(size=1.5, color="purple")
```

```
grt #draw graph
```



We have previously selected one movie, and arranged datas by years : variation of ratings (averaged) by year is significant. It seems there is a time effect on rating. We try to determine in the following if time effect has to be taken in account.

Data exploration - individual effects analysis

- movie effect version 1
- movie effect version 2
- user effect

- gender effect
- gender preferred by user effect
- time effect
- test : time effect on rating for the most rated movie, by user

```
#create training and test set
test_index <- createDataPartition(edx_base$rating, times=1, p=0.5, list=FALSE)
trainset <- edx_base[-test_index,]
testset <- edx_base[test_index,]
testset <- testset[1:4500027,] #with this, testset has exactly the same number
                               #of rows than trainset

train1 <- trainset
train2 <- trainset
test1 <- testset
test2 <- testset

mu <- mean(trainset$rating)
train1 <- train1 %>% mutate(mu)     #add 'mu' to train1
train2 <- train2 %>% mutate(mu)     #add 'mu' to train2
trainset <- trainset %>% mutate(mu) #add 'mu' to trainset
```

(description is in the code, summary of results in the end of the section)

```
###MOVIE EFFECT (version 1) (fil)
#
feffect <- train1 %>% group_by(movieId) %>%
  summarize(fil = mean(rating-mu)) %>%
  mutate()
#'fil' is the movie effect, calculated as
#the average difference between rating and mu
#RMSE
test1 <- test1 %>% mutate(mu) #add a column 'mu' to test1

ftest1 <- test1 %>%
  left_join(feffect, by='movieId')

ftest1[is.na(ftest1)] <- 0 #replace NA by 0 in dataframe ftest1

preds <- ftest1$mu + ftest1$fil
f1 <- RMSE(preds, ftest1$rating)
```

```
###MOVIE EFFECT (version 2) (fil2)
#
#I apply a regularization linked to the relative weight of each movie
#in the total number of ratings
#Hypothesis : if the movie has few ratings, the 'movie effect' is less significant

#create a dataframe with number of ratings per movie
table <- train2 %>% group_by(movieId) %>%
  summarize(nbr = n()) %>% mutate()

med <- median(table$nbr) #median
```

```r
#hypothesis  : if the number of ratings for one movie is less than the median,
#we apply a 'penalty' of 10% (x 0.9) on movie effect
table <- table %>% mutate(med)
penalty <- ifelse(table$nbr < table$med, 0.9, 1)
table <- table %>% mutate(penalty)

table <- table %>% left_join(feffect, by='movieId') #feffect is the same as in
fil2 <- table$penalty * table$fil              #MOVIE EFFECT V1 (train1 = train2)
table <- table %>% mutate(fil2)                #fil2 is the new movie effect, with penalty

f2effect <- table %>% select(movieId, fil2)
#f2effect is a dataframe with the movieId and the new movie effect

#RMSE
test2 <- test2 %>% mutate(mu) #add a column 'mu' to test2

ftest2 <- test2 %>%
  left_join(f2effect, by='movieId')

ftest2[is.na(ftest2)] <- 0 #replace NA by 0 in the dataframe ftest2

preds <- ftest2$mu + ftest2$fil2
f2 <- RMSE(preds, ftest2$rating)
```

```r
###USER EFFECT (usr)
#
ueffect <- trainset %>% group_by(userId) %>%
  summarize(usr = mean(rating-mu)) %>%
  mutate()
#'usr' is the user effect, calculated as the average difference between rating and mu
#RMSE
testset <- testset %>% mutate(mu) #add a column 'mu' to testset

ftestset <- testset %>%
  left_join(ueffect, by='userId')

ftestset[is.na(ftestset)] <- 0 #replace NA by 0 in the dataframe ftestset

preds <- ftestset$mu + ftestset$usr
u <- RMSE(preds, ftestset$rating)
```

```r
###(MULTI)GENDER EFFECT (gen)
#
geffect <- trainset %>% group_by(genres) %>%
  summarize(gen = mean(rating-mu)) %>%
  mutate()
#'gen' is the gender effect, calculated as the average difference between rating and mu
#RMSE
ftestset <- testset %>%
  left_join(geffect, by='genres')

ftestset[is.na(ftestset)] <- 0 #replace NA by 0 in the dataframe ftestset
```

```r
preds <- ftestset$mu + ftestset$gen
g <- RMSE(preds, ftestset$rating)
```

```r
###"(MULTI)GENDER PREFERRED BY USER" EFFECT
#
gueffect <- trainset %>% group_by(genres, userId) %>%
  summarize(gu = mean(rating-mu)) %>% #'gu' is the gender preferred by user effect
  mutate()
#RMSE
ftestset <- testset %>%
  left_join(gueffect, by=c('genres', 'userId'))

ftestset[is.na(ftestset)] <- 0 #replace NA by 0 in the dataframe ftestset

preds <- ftestset$mu + ftestset$gu #our prediction
gu <- RMSE(preds, ftestset$rating)
```

```r
###IMPACT OF TIME ON RATING
#
fit <- lm(rating ~ timestamp, trainset) #fit a linear regression
yhat <- predict(fit, testset) #yhat is the prediction

t <- RMSE(yhat, testset$rating) #RMSE
```

```r
###TEST : Time effect on rating, for the most rated film, by user
#
plus <- trainset %>% group_by(movieId) %>% summarize(nb=n()) %>% mutate()
plus <- data.frame(plus)
plus <- plus[order(-plus$nb),]
maxi <- plus[1,] # maxi gives the movieId for the movie which has
                 #the most important number of ratings

edx296 <- trainset %>% filter (movieId ==296)
#fit a linear regression model (timestamp and userId are predictors)
fit <- lm(rating ~ timestamp + userId, trainset)
yhat <- predict(fit, testset)

tf <- RMSE(yhat, testset$rating) #RMSE
```

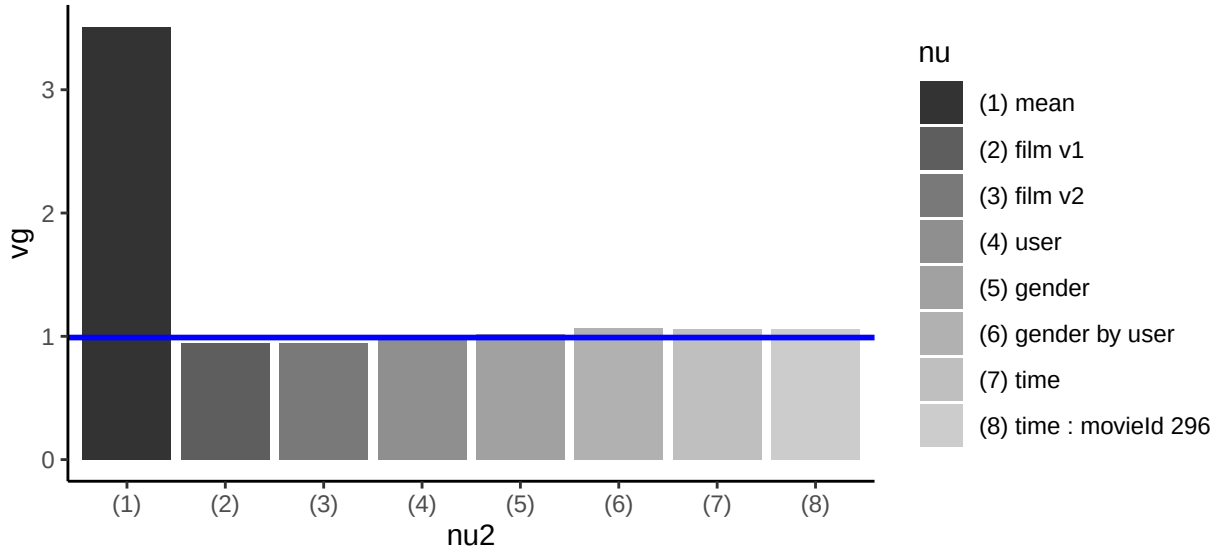
```r
###RESULTS (vgg)
# x axis : effects studied above
# y axis : RMSE (vg)

vg <- c(mu, f1, f2, u, g, gu, t, tf)
nu <- c("(1) mean", "(2) film v1", "(3) film v2", "(4) user", "(5) gender", "(6) gender by user", "(7)
nu2 <- c("(1)", "(2)", "(3)", "(4)", "(5)", "(6)", "(7)", "(8)")
vgg <- data.frame(nu,nu2,vg)
vgg
```

```
##                       nu nu2        vg
## 1              (1) mean (1) 3.5122836
## 2           (2) film v1 (2) 0.9440299
```

```
## 3            (3) film v2 (3) 0.9439644
## 4               (4) user (4) 0.9820291
## 5             (5) gender (5) 1.0179954
## 6     (6) gender by user (6) 1.0704029
## 7               (7) time (7) 1.0594780
## 8 (8) time : movieId 296 (8) 1.0594737
```

```
vgg %>% ggplot(aes(x = nu2 , y = vg , fill=nu)) +
geom_col() +
scale_fill_grey() +
theme_classic() +
#add a line to estimate if the considered effect has to be taken into account for calculus
geom_hline(yintercept=0.99, colour="blue", size=1)
```



We take the hypothesis that a predictor is relevant if the precision of the prediction is less than one star. We try several models with different combinations.

Models Evaluation

We compute a comparison between four models :

- movie effect version 1 + user effect
- movie effect version 2 + user effect
- movie effect version 2 + user effect + gender preferred by user effect
- movie effect version 2 + user effect + gender effect

(description is in the code, summary of results in the end of the section)

```
###MODEL 1 (movie effect version 1 + user effect)

#movie effect is already calculated in 4.3.1 : feffect / fil
ftrainset <- trainset %>% left_join(feffect, by='movieId')   #movie effect is added
```

```
#usr
ueffect <- ftrainset %>% group_by(userId) %>%                #ratings are grouped by userId
  summarize(usr = mean(rating-mu-fil)) %>%
  mutate()
#'usr' is calculated as the average by user of rating-mu-fil
ftrainset <- ftrainset %>% left_join(ueffect, by='userId')   #user effect is added

#RMSE
ftestset <- testset %>%
  left_join(feffect, by='movieId') %>%
  left_join(ueffect, by='userId')

ftestset[is.na(ftestset)] <- 0                               #replace NA by 0 in dataframe ftestset

preds <- ftestset$mu + ftestset$fil + ftestset$usr
model1 <- RMSE(preds, ftestset$rating)                       #calculate RSME for model1
```

```
###MODEL2 (movie effect version 2 + user effect)

#movie effect is already caculated in 4.3.2 : f2effect / fil2
f2trainset <- trainset %>% left_join(f2effect, by='movieId') #movie effect is added

#usr
ueffect <- f2trainset %>% group_by(userId) %>%               #ratings are grouped by userId
  summarize(usr = mean(rating-mu-fil2)) %>%
  mutate()
f2trainset <- f2trainset %>% left_join(ueffect, by='userId') #user effect is added

#RMSE
ftestset <- testset %>%
  left_join(f2effect, by='movieId') %>%
  left_join(ueffect, by='userId')

ftestset[is.na(ftestset)] <- 0                               #replace NA by 0 in dataframe ftestset

preds <- ftestset$mu + ftestset$fil2 + ftestset$usr
model2 <- RMSE(preds, ftestset$rating)                       #calculate RSME for model2
```

```
###MODEL3 (movie effect version 2 + user effect + gender preferred by user effect)

#movie effect is already caculated in 4.3.2 : f2effect / fil2
f3trainset <- trainset %>% left_join(f2effect, by='movieId') #movie effect is added

#usr
ueffect <- f3trainset %>% group_by(userId) %>%               #ratings are grouped by userId
  summarize(usr = mean(rating-mu-fil2)) %>%
  mutate()
f3trainset <- f3trainset %>% left_join(ueffect, by='userId') #user effect is added

#gu (gender preferred by user)
gueffect <- f3trainset %>% group_by(genres, userId) %>%
#ratings are grouped by genres and by userId
  summarize(gu = mean(rating-mu-fil2-usr)/10) %>%
```

```
  mutate()
#as user is already taken in account (ueffect), we consider gu only for 10% of its value
f3trainset <- f3trainset %>% left_join(gueffect, by=c('genres', 'userId'))
#gu effect is added

#RMSE
ftestset <- testset %>%
  left_join(f2effect, by='movieId') %>%
  left_join(ueffect, by='userId') %>%
  left_join(gueffect, by=c('genres', 'userId'))

ftestset[is.na(ftestset)] <- 0                          #replace NA by 0 in dataframe ftestset

preds <- ftestset$mu + ftestset$fil2 + ftestset$usr + ftestset$gu
model3 <- RMSE(preds, ftestset$rating)                  #calculate RSME for model3
```

```
###MODEL4 (movie effect version 2 + user effect + gender effect)

#movie effect is already caculated in 4.3.2 : f2effect / fil2
f4trainset <- trainset %>% left_join(f2effect, by='movieId')  #movie effect is added

#usr
ueffect <- f4trainset %>% group_by(userId) %>%                #ratings are grouped by userId
  summarize(usr = mean(rating-mu-fil2)) %>%
  mutate()
f4trainset <- f4trainset %>% left_join(ueffect, by='userId')  #user effect is added

#g (gender)
geffect <- f4trainset %>% group_by(genres) %>%                #ratings are grouped by genres
  summarize(g = mean(rating-mu-fil2-usr)) %>%
  mutate()
f4trainset <- f4trainset %>% left_join(geffect, by='genres')  #g effect is added

#RMSE
ftestset <- testset %>%
  left_join(f2effect, by='movieId') %>%
  left_join(ueffect, by='userId') %>%
  left_join(geffect, by='genres')

ftestset[is.na(ftestset)] <- 0                          #replace NA by 0 in dataframe ftestset

preds <- ftestset$mu + ftestset$fil2 + ftestset$usr + ftestset$g
model4 <- RMSE(preds, ftestset$rating)                  #calculate RSME for model4
```

```
#RESULTS

model1
```

```
## [1] 0.8696689
```

```
model2
```

```
## [1] 0.8695269
```

```
model3
```

```
## [1] 0.866201
```

```
model4
```

```
## [1] 0.8691807
```

Result Section

The third model (Model 3) gives the best performance (predictors : film v2 + user + gender preferred by user).

The model has to be calculated again, with the whole training dataset : edx

```
mu <- mean(edx$rating)

#calculate the movie effect
feffect <- edx %>% group_by(movieId) %>% summarize(fil = mean(rating-mu)) %>% mutate()

#create a dataframe with number of ratings per movie
table <- edx %>% group_by(movieId) %>%
  summarize(nbr = n()) %>% mutate()

med <- median(table$nbr) #median

#if the number of ratings for one movie is less than the median,
#we apply a 'penalty' of 10% (x 0.9) on movie effect
table <- table %>% mutate(med)
penalty <- ifelse(table$nbr < table$med, 0.9, 1)
table <- table %>% mutate(penalty)

table <- table %>% left_join(feffect, by='movieId')           #fil is added to calculate fil2
fil2 <- table$penalty * table$fil
table <- table %>% mutate(fil2)

f2effect <- table %>% select(movieId, fil2)
edxfinal <- edx %>% left_join(f2effect, by='movieId')         #movie effect is added

#usr
ueffect <- edxfinal %>% group_by(userId) %>%
  summarize(usr = mean(rating-mu-fil2)) %>% mutate()
edxfinal <- edxfinal %>% left_join(ueffect, by='userId')      #user effect is added

#gu (gender preferred by user)
gueffect <- edxfinal %>% group_by(genres, userId) %>%
  summarize(gu = mean(rating-mu-fil2-usr)/10) %>% mutate()
#as the user effect is already taken in account in the model,
#we consider gu only for 10% of its value

#gender by user effect is added
edxfinal <- edxfinal %>% left_join(gueffect, by=c('genres', 'userId'))
```

Our model is ready to use, we apply it on testset.

```
# rating average and the tree effects (movie effect v2, user effect,
# gender preferred by user effect) calculated with the training set (edx) are linked
# with relevant rows of the testset (validate)

# Because some couples (multi)gender items/user are not present in the test set,
# the effect gu (gender preferred by user) is sometimes NA. We replace NA by zeros.

load('validation.Rda')

val <- validation %>%
  left_join(f2effect, by='movieId') %>%
  left_join(ueffect, by='userId') %>%
  left_join(gueffect, by=c('genres', 'userId'))

val[is.na(val)] <- 0

res_final <- mu + val$fil2 + val$usr + val$gu
```

```
RMSE(res_final, validation$rating) #calculate RMSE on test set
```

```
## [1] 0.8608042
```

The final model selected is model 3 (predictors : movie v2 + user + gender preferred by user). The residual mean squared error with the test set is 0.8608 . It's less than one, and as described in the book "Introduction to Data Science", it is similar to standard deviation : our model is able to predict with a precision better than one star.

In terms of performance, I think it's also possible to capitalize on time effects, but I don't know for the moment how to improve results with time, and to integrate it in the model.

Concerning "gender by user effect", the performance of the predictor taken alone is not good. But in association with others, and minimized, it improves the final result. I think it is possible to calculate the best level for it (x %), maybe with an iterative approach.

For the "penalty" on movie effect, it's the same : is the median the best figure to separate datas ? And a penalty of 10% maybe is not the best choice. A more rigorous approach is probably possible.

Conclusion

After data cleaning and a first study, the process was to take available predictors one by one, and try to estimate if they could contribute with others in the model, to get finally a good prediction with the test set.

In a second time, to evaluate the quality of the prediction with several predictors, I built four models with different combinations. Then I selected the best one.

In terms of limitations, I think I could write a code to repeat the same process (build a model with several predictors), but with all combinations of all predictors.

Another possibility concerning genders is perhaps to separate them for every movie, and to build a system with individual marks, to have a better view of the impact on rating.

Regularization can also be applied to other predictors, such as users.