MovieLens Project

Victor Lopez

16/2/2021

INTRODUCTION

On this project our goal is to predict on most accurately way the rate a user will give to a movie, the data contain: UserId, MovieId, Movie genres, timestamp. On this project we start from data proportioned by the script on Capstone section, a sample data was taken from movielens , and shaped to have a data frame on tidy format, stored on edx data frame, also a portion of data was taken to validate the model, stored on validation data frame. exploration and analysis will be using only edx data, due validation data.frame is for final validation purpose only, to test the model edx data is splitted on edx_train and edx_test The data can give us an idea of a movie most common rate, also with this history we can figure out the user behavior to have a better estimate of movie rates.

METHODS/ANALYSIS

To analyze data we will take a look of variables and number of observations

```r
str(edx)
```

```
## Classes 'data.table' and 'data.frame':   9000055 obs. of  6 variables:
##  $ userId   : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ movieId  : num  122 185 292 316 329 355 356 362 364 370 ...
##  $ rating   : num  5 5 5 5 5 5 5 5 5 5 ...
##  $ timestamp: int  838985046 838983525 838983421 838983392 838983392 838984474 838983653 838984885 83
##  $ title    : chr  "Boomerang (1992)" "Net, The (1995)" "Outbreak (1995)" "Stargate (1994)" ...
##  $ genres   : chr  "Comedy|Romance" "Action|Crime|Thriller" "Action|Drama|Sci-Fi|Thriller" "Action|Ac
##  - attr(*, ".internal.selfref")=<externalptr>
```

We can have more information about variables using summary command

```r
summary(edx)
```

```
##      userId         movieId          rating        timestamp
##  Min.   :    1   Min.   :    1   Min.   :0.500   Min.   :7.897e+08
##  1st Qu.:18124   1st Qu.:  648   1st Qu.:3.000   1st Qu.:9.468e+08
##  Median :35738   Median : 1834   Median :4.000   Median :1.035e+09
##  Mean   :35870   Mean   : 4122   Mean   :3.512   Mean   :1.033e+09
##  3rd Qu.:53607   3rd Qu.: 3626   3rd Qu.:4.000   3rd Qu.:1.127e+09
##  Max.   :71567   Max.   :65133   Max.   :5.000   Max.   :1.231e+09
##     title              genres
##  Length:9000055     Length:9000055
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##
```

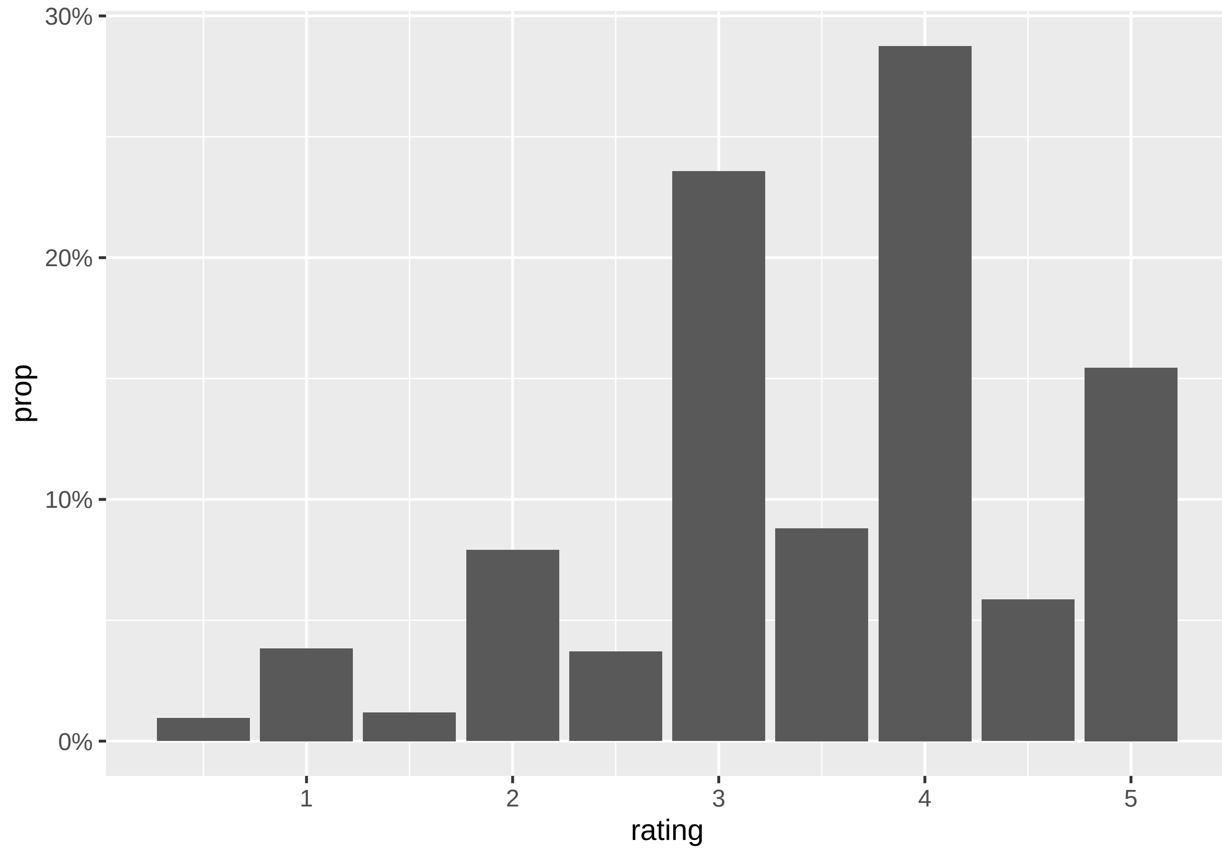
First five Observation of data

```r
head(edx,5)
```

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

exploring Rating behavior,

```r
edx%>%ggplot(aes(x=rating))+geom_bar(aes(y=..prop..,group=1))+
   scale_y_continuous(labels = scales::percent_format())
```



We notice 1. There is only whole or half rate. 2. Half rate are less common than whole rate 2. 3 and 4 rate are the most common

To creat a predicion model the first task is to split edx data on two, one for training and other for testing, we split edx into edx_train and edx_test 15% of the data will store on edx_test and 85% of data will be used on training

```r
set.seed(1, sample.kind="Rounding")
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

```r
test_index <- createDataPartition(y = edx$rating, times = 1, p = 0.15, list = FALSE)
edx_train <- edx[-test_index,]
temp1 <- edx[test_index,]
# Make sure userId and movieId in edx_train set are also in edx_test set
edx_test <- temp1 %>%
      semi_join(edx_train, by = "movieId") %>%
      semi_join(edx_train, by = "userId")
# Add rows removed from edx_test set back into edx_train
removed <- anti_join(temp1, edx_test)
```

```
## Joining, by = c("userId", "movieId", "rating", "timestamp", "title", "genres")
```

```r
edx_train <- rbind(edx_train, removed)
```

Also On our analysis testing we will need a function to calculate the RMSE :

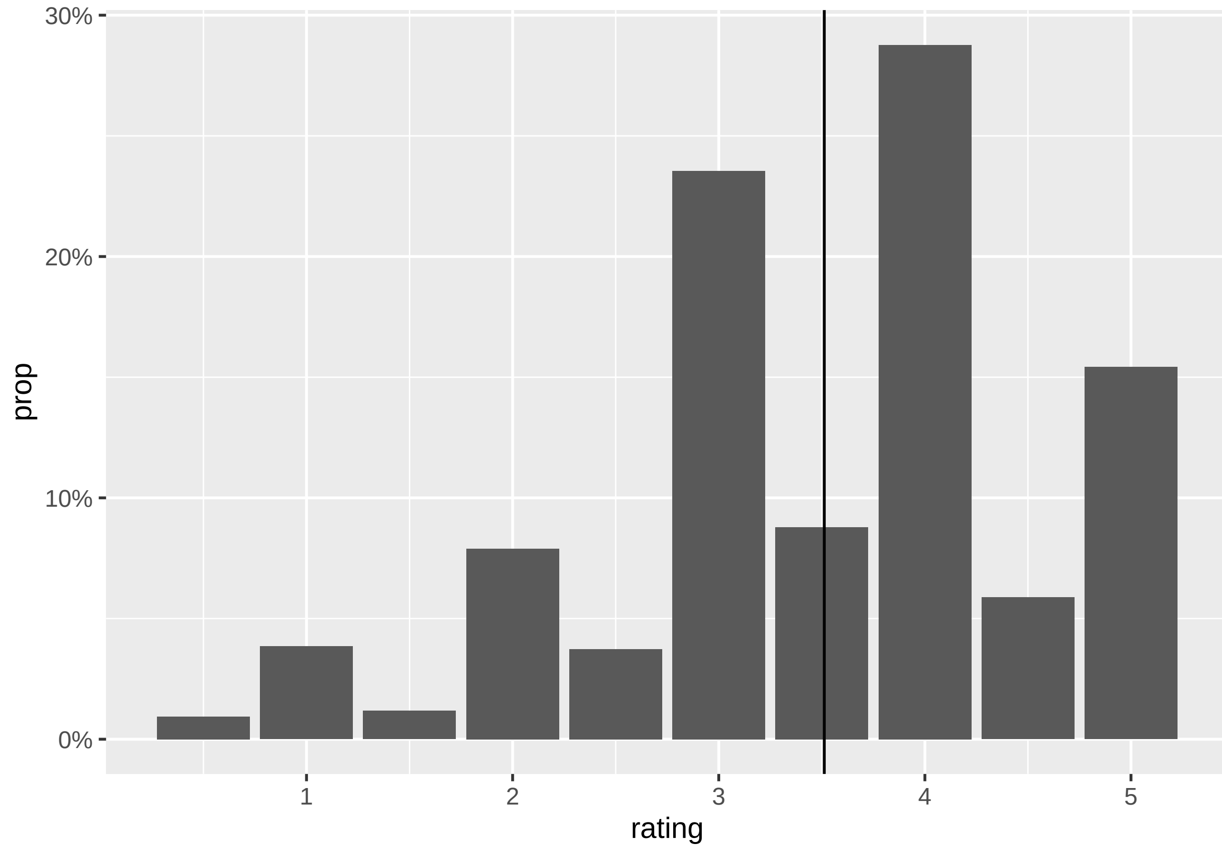
```r
RMSE<-function(predicted,real){
sqrt(mean((predicted-real)^2))}  #function to predict RMSE
```

To apreciate Predicion on a more graphic way, we will graphic error predicion on each rate group using this function

```r
ERRORF<-function(predicted,real){
   dataf1<-data.frame(predicted,real)
   dataf1<-dataf1%>%mutate(e=dataf1[,1]-dataf1[,2])
   index<-createDataPartition(y=dataf1$e,times=1,p=0.001,list=FALSE)
   #only a portion of total error
   dataf<-dataf1[index,]
dataf%>%ggplot(aes(x=dataf[,2],y=dataf[,3],colour=dataf[,3]))+
   geom_point(alpha=1/100000,size = 0.001)+geom_jitter()+
  scale_colour_gradient2()+ labs(title = "Sample of rates errors", x= "Rates 1 to 5", y="Predicted erro
}
```

The rating average will be a good value to start prediction model, the rating average is 3.512465 on edx_train , and we got an 1.06 RMSE on edx_test

```r
u<- mean(edx_train$rating)
edx_test%>%ggplot(aes(x=rating))+geom_bar(aes(y=..prop..,group=1))+
   geom_vline(xintercept =3.512)+scale_y_continuous(labels = scales::percent_format())
```



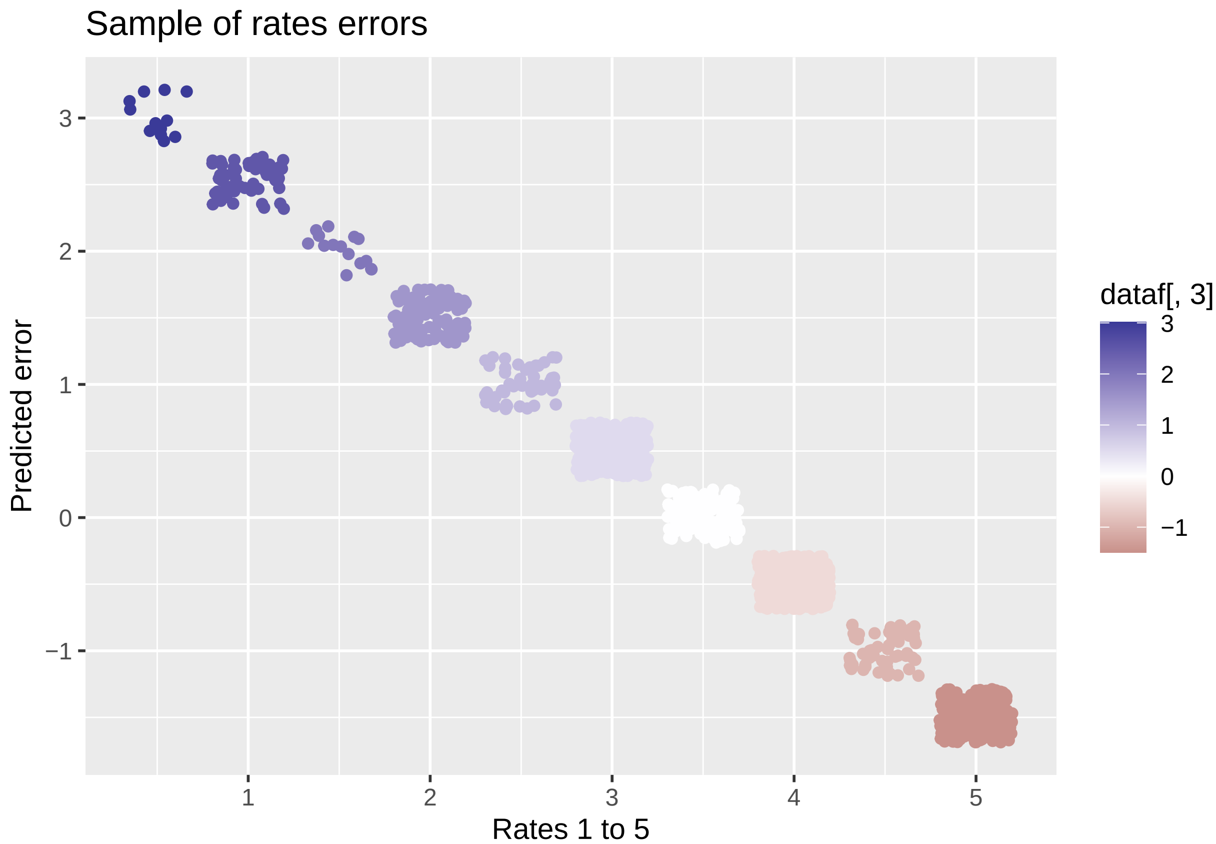

```r
edx_test<-edx_test%>%mutate(p1=mean(edx_train$rating))
edx_train<-edx_train%>%mutate(p1=mean(edx_train$rating))
```

RMSE:

```r
RM<-RMSE(edx_test$p1,edx_test$rating)
RM
```

```
## [1] 1.060077
```

```r
RMSEplot<-c(RM)
RMSEplot2<-c("Average")
ERRORF(edx_test$p1,edx_test$rating)
```



As we notice , on movie rate 3.5 we have 0 error, but We still have high difference on other rates ,above the average and others below, we have a large amount of error on movies rated , the model need another variable, the movie effect,

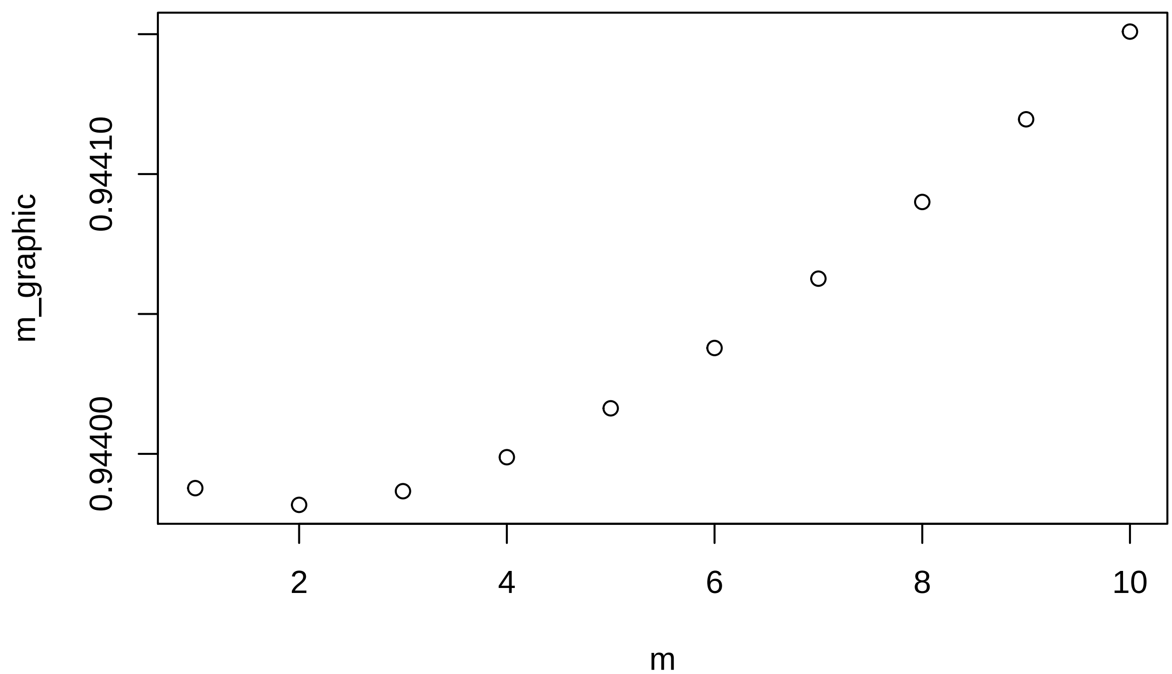
IMPROVING WITH MOVIE EFFECT: to take this into account the movies rating error will be grouped by movie. but a factor should be taken into account too, what would happen if a movie has only one or two rated, the average of error could be slanted, based only on few rated we can't have a good estimate of how good a movie is for other users, an "m" variable is used to normalize average and reduce slanted. movie_b=(rating-u)/m on next steps are the calculation of this m value to minimize RMSE

```
m<-c(1:10) # Vector of m values to test
# Function to calculate RMSE with each new m value
best_m<- function(train,test,m){
train<-train%>%mutate(u=mean(train$rating))
test<-test%>%mutate(u=mean(train$rating))
data1<-suppressWarnings(train%>%mutate(b1=rating-u)%>%group_by(movieId)%>%summarize(movie_b=sum(b1)/(n(
test<-test%>%left_join(data1,by="movieId")
test<-test%>%mutate(pred=u+movie_b)
RMSE(test$rating,test$pred)
}
m_graphic<- sapply(m,best_m,train=edx_train,test=edx_test) #to see the m behavior on RMSE
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
plot(m,m_graphic)
```




```
which.min(m_graphic) #best m value to movie effect
```

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

the movie effect will be on data frame data_movie and stored on edx_train on column movie_b

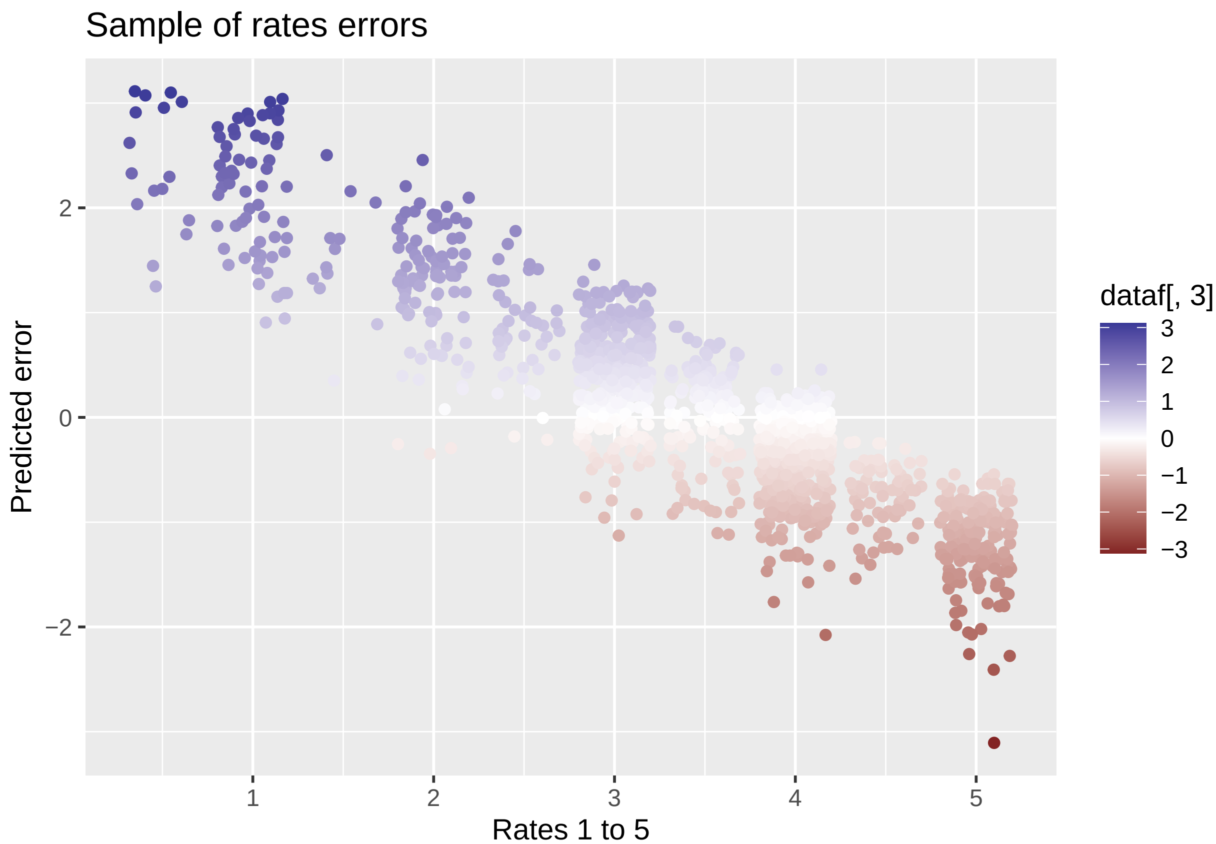
```
data_movie <-edx_train%>%mutate(b1=rating-u)%>%group_by(movieId)%>%summarise(movie_b=sum(b1)/(n()+2))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
edx_train<-edx_train%>%left_join(data_movie,by="movieId")#adding movie effect on edx_train
edx_test<-edx_test%>%left_join(data_movie,by="movieId")#adding movie effect on edx_test
edx_test<-edx_test%>%mutate(p2=u+movie_b)
RM<-RMSE(edx_test$p2,edx_test$rating)
RM
```

```
## [1] 0.9439818
```

```
RMSEplot<-append(RMSEplot,RM)
RMSEplot2<-append(RMSEplot2,"Movie Effect")
ERRORF(edx_test$p2,edx_test$rating)
```



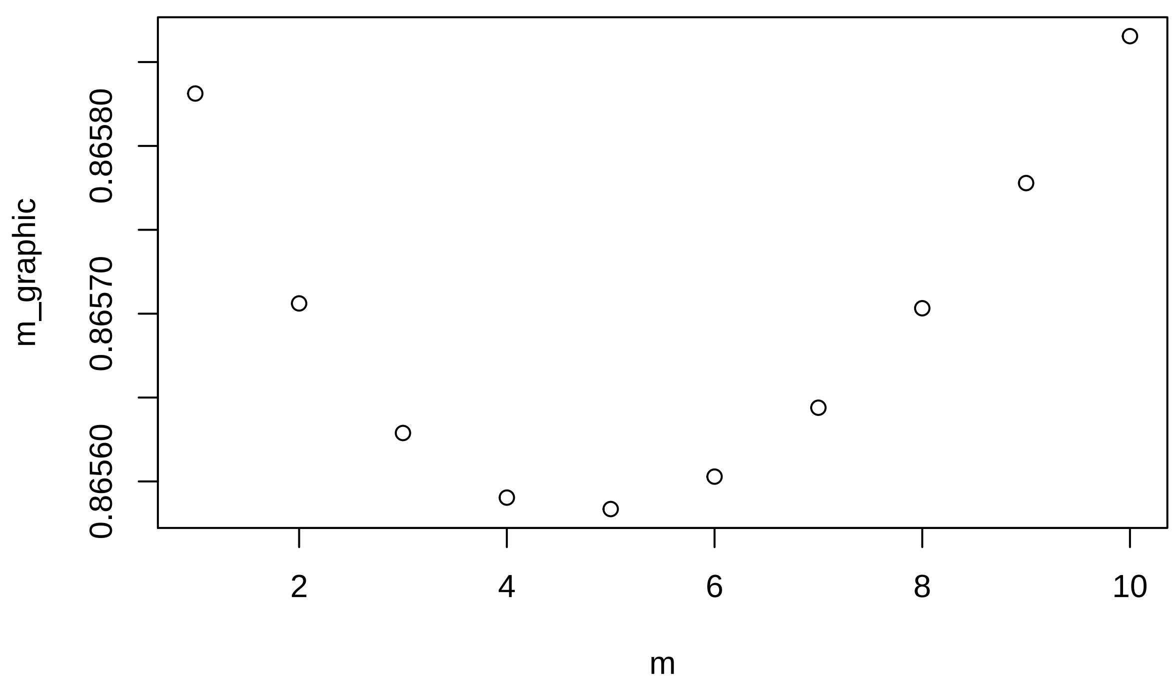
IMPROVING WITH USER EFFECT: As same as Movie effect model can be improve, adding the user effect, on next scrip user effect will be find and named user_b, as same as movie effect a factor "m" will be used to normalize an minimize the effect that a few user has rated a movie.

```
m<-c(1:10) # Vector of m values to test
# Function to calculate RMSE with each new m value
best_m2<- function(train,test,m){
data1<-suppressWarnings(train%>%mutate(b2=rating-(u+movie_b))%>%group_by(userId)%>%summarize(user_b=sum
test1<-test%>%left_join(data1,by="userId")
test1<-test1%>%mutate(pred=u+movie_b+user_b)
RMSE(test$rating,test1$pred)
}
m_graphic<- sapply(m,best_m2,train=edx_train,test=edx_test)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
plot(m,m_graphic)
```




```
which.min(m_graphic)  #best m value to user effect
```

```
## [1] 5
```

The user effect will be on the column user_b and store on data frame data_user

```
data_user<-edx_train%>%mutate(b2=rating-(u+movie_b))%>%group_by(userId)%>%summarize(user_b=sum(b2)/(n()
```

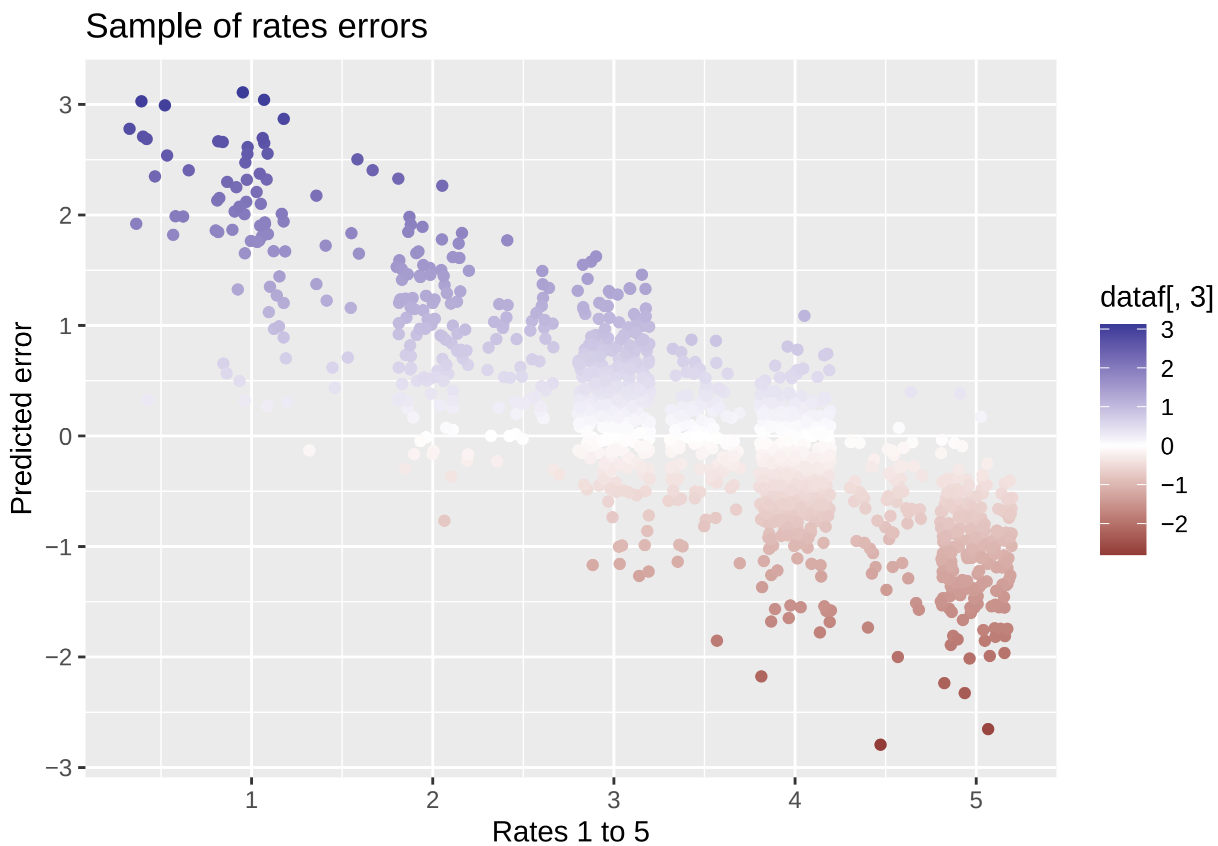
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

Calculating RMSE on edx_test with mean, movie effect and user effect, using u+movie_b+userb

```
edx_train<-edx_train%>%left_join(data_user,by="userId")
edx_test<-edx_test%>%left_join(data_user,by="userId")
edx_test<-edx_test%>%mutate(pred=u+movie_b+user_b)
RM<-RMSE(edx_test$pred,edx_test$rating)
RM
```

```
## [1] 0.8655835
```

```
RMSEplot<-append(RMSEplot,RM)
RMSEplot2<-append(RMSEplot2,"User effect")
ERRORF(edx_test$pred,edx_test$rating)
```



We can add another variable to the model, the genre, if a user like a specific genre for example a user like Drama movies, any movie could has a chance to get higher rate if is a Drama movie when is rated by this user. Genre likeliness is subjective and depend from user, the model also has to take this into account, also a movie could have more than one genre. Take into account all the genres and analyze the genres by each user could add a lot of complexity to the model, so a exploration to the data is made to choose the most representative genres

```
genres<-as.data.frame(str_split_fixed(edx_train$genres,"\\|",4))
genres<-gather(genres)
genres%>%group_by(value)%>%summarise(genre_number=n())%>%arrange(desc(genre_number))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 101 x 2
##    value       genre_number
##    <chr>              <int>
##  1 ""              11235235
##  2 "Drama"          3268472
##  3 "Comedy"         2900189
```

```
##  4 "Action"         2176966
##  5 "Thriller"       1796417
##  6 "Adventure"      1621952
##  7 "Romance"        1325979
##  8 "Crime"          1110463
##  9 "Sci-Fi"         1037526
## 10 "Fantasy"         659432
## # ... with 91 more rows
```

Most common genres on movies are: “Drama”,”Comedy”,”Action”,Thriller” to include to the model analysis the column Drama,Comedy,Action,Thriller are adding on edx_train, to know if a movie contain one of these genres, if so, userId also will be associated on the genre column

```
edx_train<-edx_train%>%
   mutate(Drama=ifelse(str_detect(genres,"Drama"),str_c(userId,"-Drama"),"x"))%>%
   mutate(Comedy=ifelse(str_detect(genres,"Comedy"),str_c(userId,"-Comedy"),"x"))%>%
   mutate(Action=ifelse(str_detect(genres,"Action"),str_c(userId,"-Action"),"x"))%>%
   mutate(Thriller=ifelse(str_detect(genres,"Thriller"),str_c(userId,"-Thriller"),"x"))
```

As same as movie effect and user effect are stored on movie_data and user_data respectively a data frame will be created for each of the four representative genres to store the genre effect

```
edx_train<-edx_train%>% mutate(pred=u+movie_b+user_b)%>%mutate(err=rating-pred)  #this error will be gr
data_drama<-edx_train%>% group_by(Drama)%>%summarise(Drama_b=mean(err))%>%
   mutate(Drama_b=ifelse(Drama=="x",0,Drama_b))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
data_comedy<-edx_train%>% group_by(Comedy)%>%summarise(Comedy_b=mean(err))%>%
   mutate(Comedy_b=ifelse(Comedy=="x",0,Comedy_b))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
data_action<-edx_train%>% group_by(Action)%>%summarise(Action_b=mean(err))%>%
   mutate(Action_b=ifelse(Action=="x",0,Action_b))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
data_thriller<-edx_train%>% group_by(Thriller)%>%summarise(Thriller_b=mean(err)) %>%mutate(Thriller_b=i
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

In order to test the RMSE improvement on edx_test , the next steps should be followed 1.Add 4 columns one for each representative genre, and this column will identify if the movie contains the genre and will associate the user

```
edx_test<-edx_test%>%mutate(Drama=ifelse(str_detect(genres,"Drama"),str_c(userId,"-Drama"),"x"))%>% muta
```

2.Adding the genres effect from genres data frames data_drama, data_comedy, data_action, data_thriller

```
edx_test<-edx_test%>%left_join(data_drama,by="Drama")%>% left_join(data_comedy,by="Comedy")%>% left_join
```

3.Calculate prediction using u+movie_b+user_b+Drama_b+Comedy_b+Action_b+Thriller_b

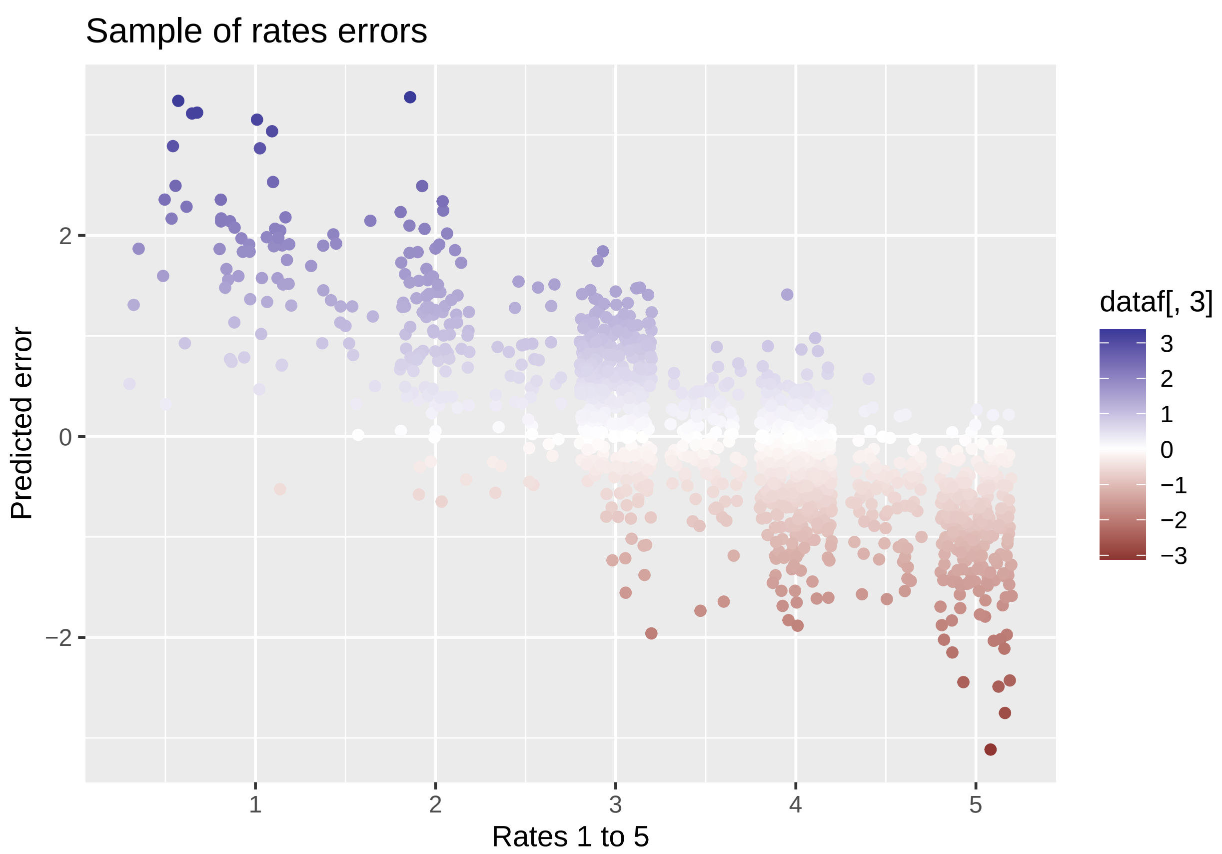
```
edx_test<-edx_test%>%mutate(pred=u+movie_b+user_b+Drama_b+Comedy_b+Action_b+Thriller_b)
RMSE(edx_test$pred,edx_test$rating)
```

```
## [1] 0.864205
```

```
RM<-RMSE(edx_test$pred,edx_test$rating)
RM
```

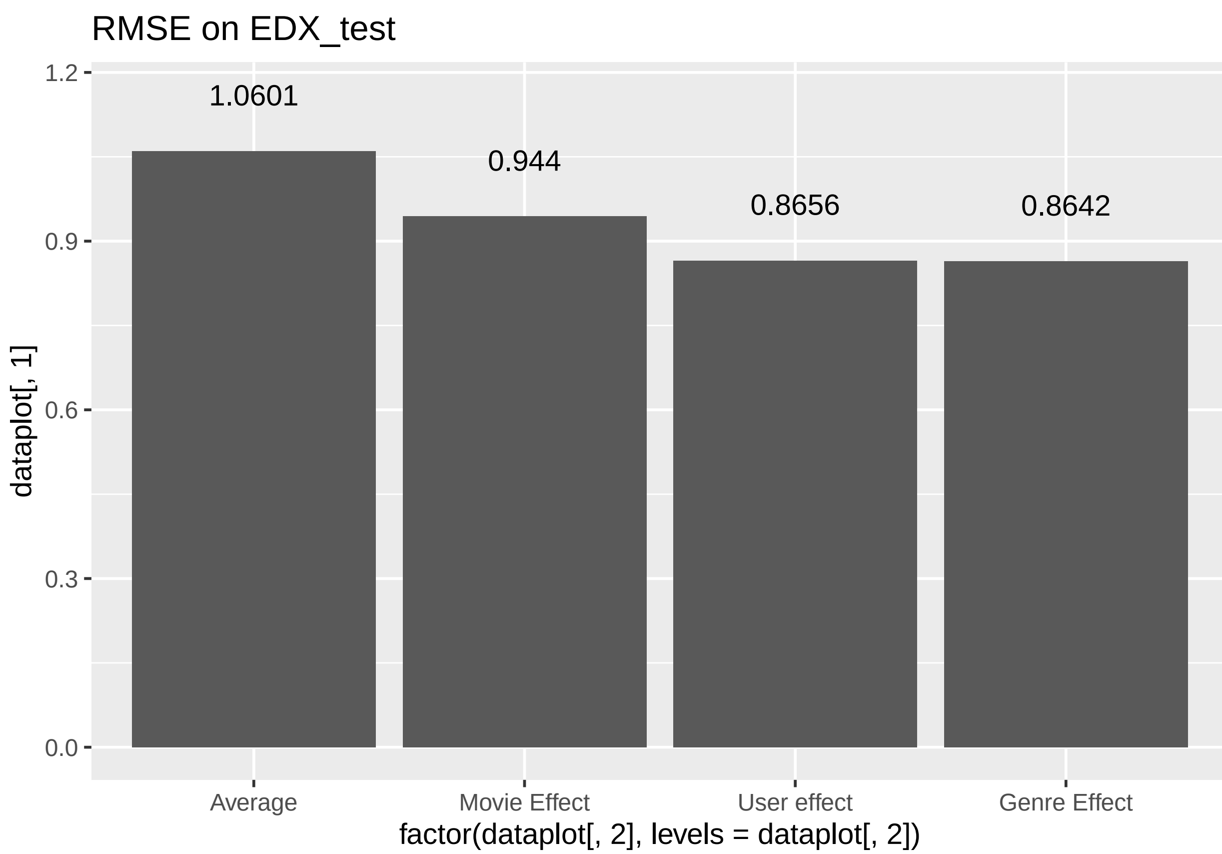
```
## [1] 0.864205
```

```
RMSEplot<-append(RMSEplot,RM)
RMSEplot2<-append(RMSEplot2,"Genre Effect")
ERRORF(edx_test$pred,edx_test$rating)
```



Final Model: The RMSE has been improve adding variables to the model, as shown on next graphic

```
dataplot<-data.frame(RMSEplot,RMSEplot2)
dataplot%>%ggplot(aes(x=factor(dataplot[,2],levels = dataplot[,2]),y=dataplot[,1],label=round(dataplot[
```



RESULTS At final, testing the model on validation data In order to calculate prediction 1 . we have to add the data from data frames: data_movie, user_movie, to add the movie an user effect 2. we add de columns who permit us join the data from data frames: data_drama, data_comedy, data_action, data_thriller 3. in some case a user could rate a genre not evaluated on genres data frames for that reason all na are replaced with 0 4.finaly prediction is calculated with the formula pred=u+movie_b+user_b+Drama_b+Comedy_b+Action_b+Thriller_b

```
validation1<-validation%>%mutate(u=u)%>%
   left_join(data_movie,by="movieId")%>%left_join(data_user,by="userId")%>%
   mutate(Drama=ifelse(str_detect(genres,"Drama"),str_c(userId,"-Drama"),"x"))%>%
   mutate(Comedy=ifelse(str_detect(genres,"Comedy"),str_c(userId,"-Comedy"),"x"))%>%
   mutate(Action=ifelse(str_detect(genres,"Action"),str_c(userId,"-Action"),"x"))%>%
   mutate(Thriller=ifelse(str_detect(genres,"Thriller"),str_c(userId,"-Thriller"),"x"))%>%
   left_join(data_drama,by="Drama")%>% left_join(data_comedy,by="Comedy")%>%
   left_join(data_action,by="Action")%>%
   left_join(data_thriller,by="Thriller")%>%
   replace_na(list(Drama_b=0,Comedy_b=0,Action_b=0,Thriller_b=0))%>%
   mutate(pred=u+movie_b+user_b+Drama_b+Comedy_b+Action_b+Thriller_b)
```

RMSE is calculated:

```
RMSE(validation1$pred,validation1$rating)
```

```
## [1] 0.863613
```