PROJECT I: MovieLens

FH

September 30, 2020

Inroduction

The analysis performed within this project is using analysis strategies of a recommendation system developed by the winners of the *Netflix challenge*. Since Netflix data is not publicly available the data used to develop a recommendation system is based on the **MovieLens** data. The original "MovieLens"(20M) data set was generated by the GroupLens research lab and can be found here: - MovieLens for 20M dataset https://grouplens.org/datasets/movielens/20m/ - MovieLens for 10M dataset https://grouplens.org/data sets/movielens/10m/

The recommendations were developed for the **edx** data which is a subset of the **MovieLens** data. For the evaluation of the recommendation algorithm a **validation** data set was generated. The **validation** data set was only used in the final step to test the final algorithm and contained only 10% of **edx** data. The final model with the smallest RMSE was chosen to be applied on **edx** data to calculate the parameters of the model. In last step this model was evaluated by calculating RMSE(residual mean squared error) on the **validation** set. Following libraries were loaded:

```
library(dslabs)
library(tidyverse)
library(caret)
library(dplyr)
library(lubridate)
library(ggplot2)
library(gridExtra)
library(data.table)
```

```
#  Code to generate edx data set
ratings <- fread(text = gsub("::", "\t", readLines("ml-10M100K/ratings.dat")),
                 col.names = c("userId", "movieId", "rating", "timestamp"))
movies <- str_split_fixed(readLines("ml-10M100K/movies.dat"), "\\::", 3)
colnames(movies) <- c("movieId", "title", "genres")
movies <- as.data.frame(movies, stringsAsFactors=TRUE) %>%
  mutate(movieId = as.numeric(levels(movieId))[movieId],
         title = as.character(title),
         genres = as.character(genres))
movielens <- left_join(ratings, movies, by = "movieId")
```

```
#  Code to generate the validation set
set.seed(675)
test_index <- createDataPartition(y = movielens$rating, times = 1, p = 0.1, list = FALSE)
edx <- movielens[-test_index,]
temp <- movielens[test_index,]

validation <- temp %>%
```

```
  semi_join(edx, by = "movieId") %>%
  semi_join(edx, by = "userId")

removed <- anti_join(temp, validation)
edx <- rbind(edx, removed)

rm(ratings, movies, test_index, temp, movielens, removed)
```

Data exploration and data processing

The **edx** data set contains 9,000,055 observations and 6 variables represented in 6 total columns. Each row represents one user giving one rating to one specific movie.

```
#  Number of variables
ncol(edx)
```

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

```
#  Number of observarions
nrow(edx)
```

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

The generated data set **edx** contains no missing data and consists of following variables:

- *movieId* is a numerical variable denotes id's for each movie
- *title* is a string variable describing the title of a movie
- *year* numerical variable with the year of movie release from 1902 to 2016
- *genres* is a categorical variable that represent 19 different genres
- *userId* a numerical variable to identify unique users
- *rating* a numerical variable from 0 to 5
- *timestamp* represents time when rating was given in seconds since January 1, 1970

```
summary(edx)
```

```
##      userId         movieId          rating        timestamp
##  Min.   :    1   Min.   :    1   Min.   :0.500   Min.   :7.897e+08
##  1st Qu.:18114   1st Qu.:  648   1st Qu.:3.000   1st Qu.:9.468e+08
##  Median :35732   Median : 1834   Median :4.000   Median :1.035e+09
##  Mean   :35867   Mean   : 4119   Mean   :3.512   Mean   :1.033e+09
##  3rd Qu.:53601   3rd Qu.: 3624   3rd Qu.:4.000   3rd Qu.:1.127e+09
##  Max.   :71567   Max.   :65133   Max.   :5.000   Max.   :1.231e+09
##     title              genres
##  Length:9000055     Length:9000055
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##
```

Before using the data for visualization and analysis several steps of data transformation were proceeded. The variable *title*" was split in 2 variables *title* and *release_year*. The variable *timestamp* was transformed to a year format and called *rating_year*. *age_years* was created as a difference between *release_year* and *rating_year*. The original *genres* variable represents a combination of several genres,for the purpose of data exploration this variable was split into 19 distinct genres and was only used to create plots.

```
-{r echo = FALSE} #  Split genres into single columns per genre, rename in edx_genre
edx_genre<-separate_rows(edx, genres, sep = "\\|") -
```

```
#  Check missing values
sum(is.na(edx))
```

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

```
head(edx,10)
```

```
##     userId movieId rating timestamp                                title
##  1:      1     122      5 838985046                             Boomerang
##  2:      1     185      5 838983525                              Net, The
##  3:      1     231      5 838983392                         Dumb & Dumber
##  4:      1     292      5 838983421                              Outbreak
##  5:      1     316      5 838983392                              Stargate
##  6:      1     329      5 838983392                 Star Trek: Generations
##  7:      1     355      5 838984474                       Flintstones, The
##  8:      1     356      5 838983653                           Forrest Gump
##  9:      1     362      5 838984885                       Jungle Book, The
## 10:      1     370      5 838984596 Naked Gun 33 1/3: The Final Insult
##     release_year                          genres rating_year age_years
##  1:         1992                  Comedy|Romance        1996         4
##  2:         1995           Action|Crime|Thriller        1996         1
##  3:         1994                          Comedy        1996         2
##  4:         1995    Action|Drama|Sci-Fi|Thriller        1996         1
##  5:         1994         Action|Adventure|Sci-Fi        1996         2
##  6:         1994   Action|Adventure|Drama|Sci-Fi        1996         2
##  7:         1994         Children|Comedy|Fantasy        1996         2
##  8:         1994        Comedy|Drama|Romance|War        1996         2
##  9:         1994      Adventure|Children|Romance        1996         2
## 10:         1994                    Action|Comedy        1996         2
```

Users and Movies

There are 69878 unique users, 10677 movies and 10407 movie titles in the **edx** data set.

```
#  Number of unique users, movies and movie titles in edx
edx %>%summarize(n_users  = n_distinct(userId),
                 n_movies = n_distinct(movieId),
                 n_title  = n_distinct(title))
```

```
##   n_users n_movies n_title
## 1   69878    10677   10407
```

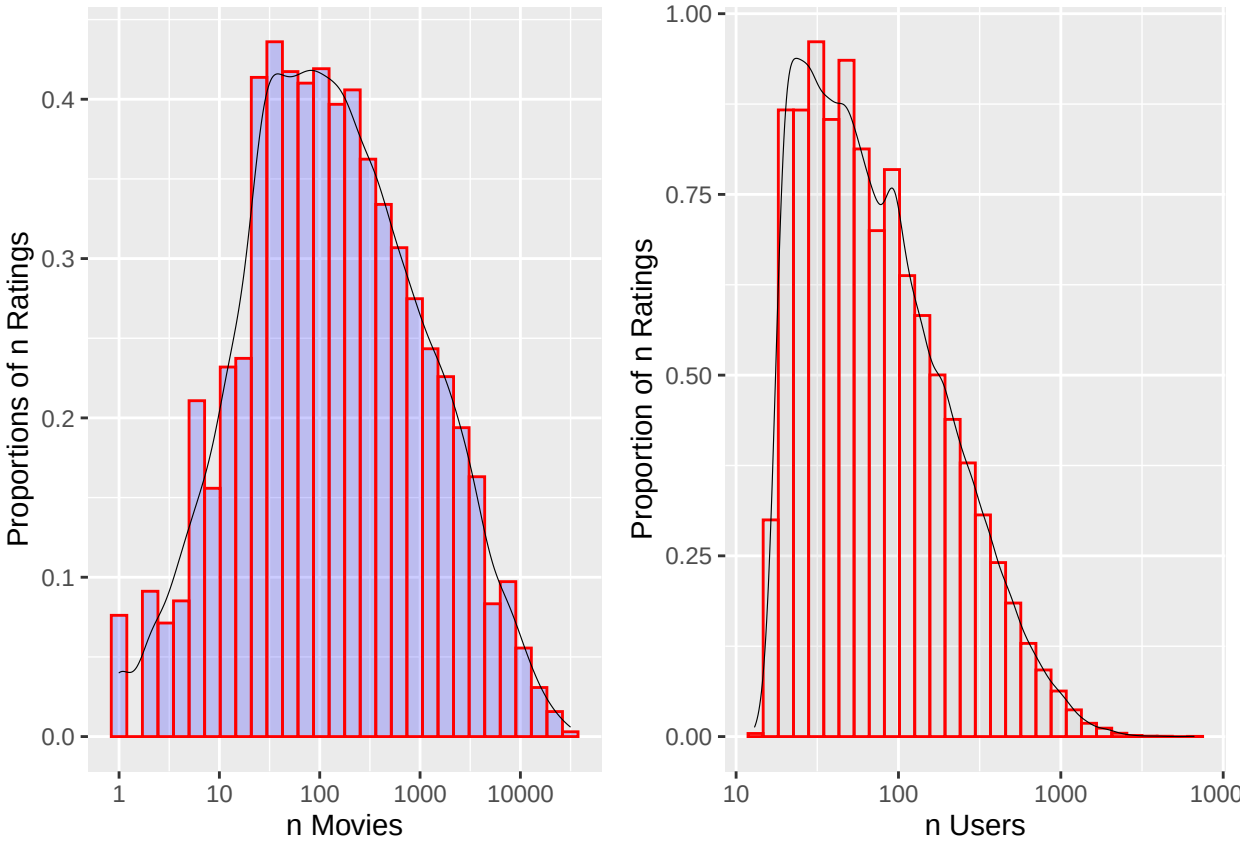
```
#  Top 10 of movies with greatest number of ratings
edx%>%group_by(movieId, title)%>%
  summarise(count = n())%>%arrange(desc(count))%>%
  top_n(10, count)
```

```
## `summarise()` regrouping output by 'movieId' (override with `.groups` argument)
```

```
## # A tibble: 10,677 x 3
## # Groups:   movieId [10,677]
##    movieId title                                                 count
```

```
##      <dbl> <chr>                                                      <int>
##  1     296 "Pulp Fiction "                                            31408
##  2     356 "Forrest Gump "                                            31095
##  3     593 "Silence of the Lambs, The "                               30265
##  4     480 "Jurassic Park "                                           29428
##  5     318 "Shawshank Redemption, The "                               28003
##  6     110 "Braveheart "                                              26270
##  7     457 "Fugitive, The "                                           26023
##  8     589 "Terminator 2: Judgment Day "                              25955
##  9     260 "Star Wars: Episode IV - A New Hope (a.k.a. Star Wars) "   25705
## 10     592 "Batman "                                                  24279
## # ... with 10,667 more rows
```

As we can see the distribution of rating count among the number of movies and number of users is skewed. Not all users were equally active in giving rating and some movies received more ratings than other. For this reason movie and user effect were taken into modeling the prediction of rating.



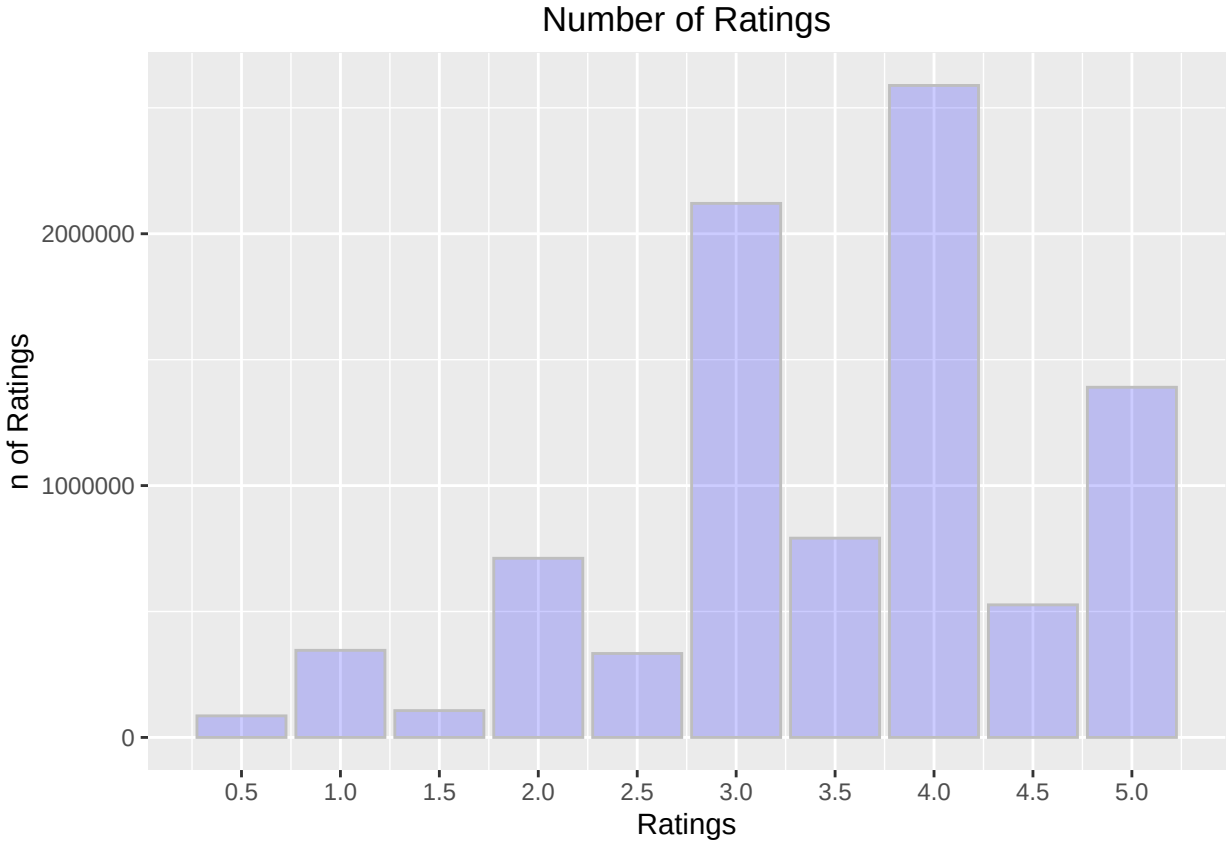
Distribution of Ratings

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

```
## # A tibble: 10 x 2
##    rating   count
##     <dbl>   <int>
##  1    4   2588714
```

```
##  2    3   2120440
##  3    5   1390384
##  4    3.5  791304
##  5    2    711339
##  6    4.5  526429
##  7    1    345861
##  8    2.5  333269
##  9    1.5  106502
## 10    0.5   85626
```

The most given ratings were 4.0 and 3.0. Different numbers of ratings among different ratings indicate that users were more likely to give a rating then they liked a movie. Full ratings outnumbered the half ratings.



Genre

In order to demonstrate the effect of genre the *genres* variable was transformed into a single column per genre.

There were 19 unique genres and one category with no listed genres.

```
#  Number of different genres
edx_genre%>%summarize(genre = n_distinct(genres))
```

```
## # A tibble: 1 x 1
##   genre
##   <int>
## 1    20
```

```r
#  Number of movies in different genres
edx_genre%>%group_by(genres) %>%
            summarize(count = n())%>%
            top_n(10)%>%arrange(desc(count))
```

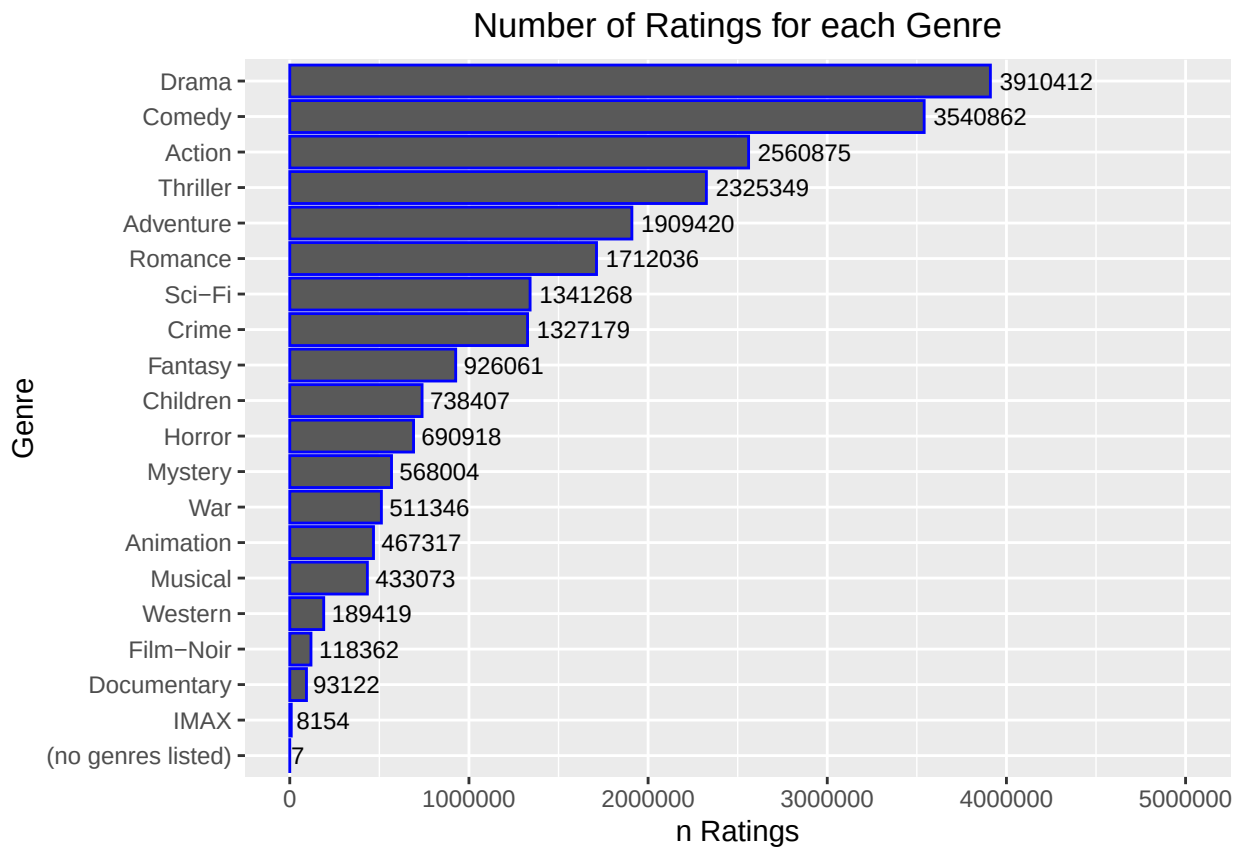
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## Selecting by count
```

```
## # A tibble: 10 x 2
##    genres      count
##    <chr>       <int>
##  1 Drama     3910412
##  2 Comedy    3540862
##  3 Action    2560875
##  4 Thriller  2325349
##  5 Adventure 1909420
##  6 Romance   1712036
##  7 Sci-Fi    1341268
##  8 Crime     1327179
##  9 Fantasy    926061
## 10 Children   738407
```

Number of ratings for each genre shows that "Drama", "Comedy" and "Action" were top 3 genres that received the most ratings. "Film-Noir", "Documentaries" and "IMAX" received the least number of ratings.

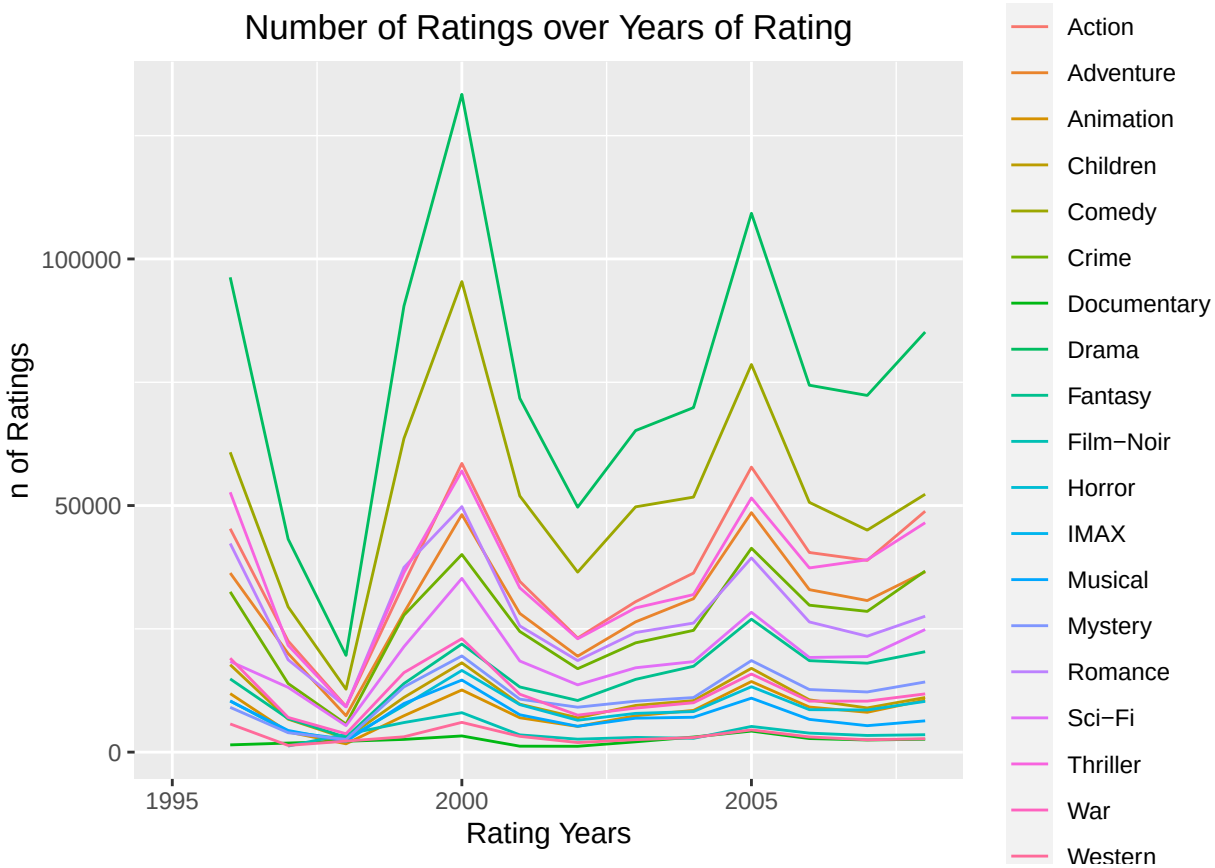
```
## `summarise()` ungrouping output (override with `.groups` argument)
```



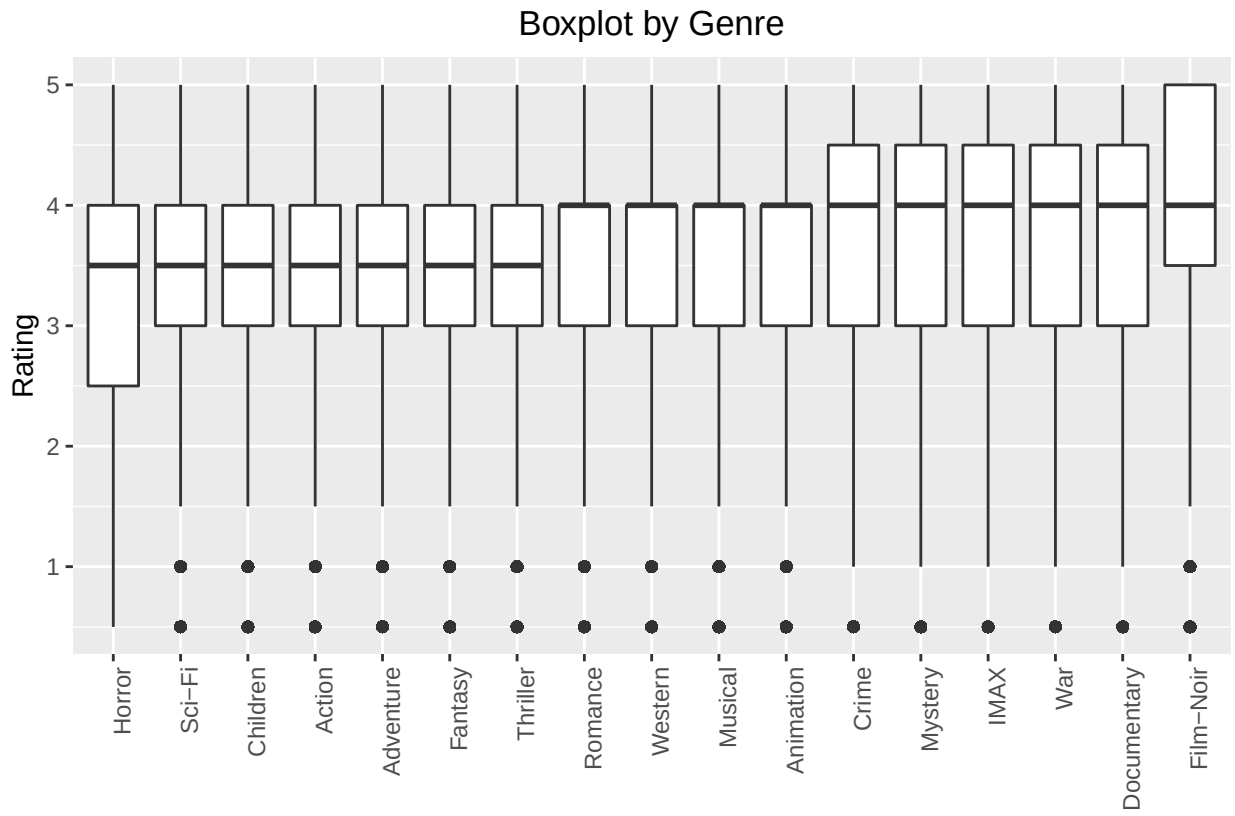
Number of ratings over years when rating was given for each genre. "Drama" remained a popular choice over years by receiving the most number of positive ratings from 4.0 and above. As it is shown in the plot below, "Drama" and "Comedy" received the highest number of rating 4.0 and above in year 2000, while "Documentary" remained flat over years.

```
## `summarise()` regrouping output by 'genres' (override with `.groups` argument)
```

```
## Warning: Removed 1 row(s) containing missing values (geom_path).
```



Following Box plots show how the ratings for each genre are spread in **edx**. For example "Film-Noir" shows the highest percentage on positive ratings although this genre received only few ratings compared to other genres. Similar to "Film-Noir", "Documentary" is a highly rated genre with not many ratings.



Method Section

It was important NOT to use the **validation** set to train the algorithm. The **edx** data was split into a **train_set** and **test_set**, where the test_set was set to 20% of the **edx data**. The **train_set** was used to train all model, while **test_set** was used to estimate predictions of ratings and to calculate the RMSE in order to evaluate the model.

```
set.seed(110)
test_index <- createDataPartition(y=edx$rating, times = 1, p = 0.2, list = FALSE)
test_set <- edx[test_index, ]
train_set <- edx[-test_index, ]
```

semi_join function removes entries for users and movies in test_set that dont's appear in th train_set.

```
test_set <- test_set %>%
  semi_join(train_set, by = "movieId") %>%
  semi_join(train_set, by = "userId")
```

There were 7 possible predictors of rating: *genres*, *release_year*, *rating_year*, *movieId*, *userId*, *title*, *age_years*. Models based on one and a combination of several predictors were tested on the **train_set**. The **test_set** was used to calculate RMSE of each model as it was done in the *Netflix challenge*. RMSE's of multiple models were compared with each other. After the comparison model that yielded the smallest RMSE was used for the final evaluation on the **validation** set to test the final algorithm. **RMSE < 0.86490** was considered acceptable. The RMSE (residual mean squared error) is defined as:

$RMSE = \sqrt{\frac{1}{N}\sum_{u,m}(\hat{y}_{u,m} - y_{u,m})^2}$ Where $y_{u,i}$ is rating by user u for movie m and $\hat{y}_{u,m}$ is the prediction

of the rating. N is a number of movie and users.

Following code for RMSE was used

```
RMSE <- function(true_ratings, predicted_ratings){
  sqrt(mean((true_ratings-predicted_ratings)^2, na.rm = T))
}
```

Important: The validation data (the final hold-out test set) should NOT be used for training your algorithm and should ONLY be used for evaluating the RMSE of your final algorithm. You should split the edx data into separate training and test sets to design and test your algorithm.

Explain the method and techniques used, Visualization with insights, data exploration. Use at least two Methods You will use the following code to generate your data sets. Develop your algorithm using the edx set. For a final test of your algorithm, predict movie ratings in the validation set (the final hold-out test set) as if they were unknown. RMSE will be used to evaluate how close your predictions are to the true values in the validation set (the final hold-out test set).

Prediction Performance RMSE in validation data set >90?

Results Section

Model performance.

For a final test of the algorithm, predict movie ratings were predicted in the validation set as if they were unknown. RMSE was used to evaluate how close predictions of rating were to the true values in the validation set.

Conclusion Section

Brief summary of the report. It's limitation and future work.