Movie Data Analysis

Nachiket Kore

January 2, 2019

R Markdown

Movie data analysis

```
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(tidyr)
library(stringr)
library(ggplot2)

setwd('C:/Users/nachi/OneDrive/Desktop/Tredence')


#Read csv file
links = read.csv('links.csv', header = TRUE, fill = TRUE)
movies = read.csv('movies.csv', header = TRUE, fill = TRUE)
ratings = read.csv('ratings.csv', header = TRUE, fill = TRUE)
tags = read.csv('tags.csv', header = TRUE, fill = TRUE)

# Making genre value change that will be required later
movies$genres <- as.character(movies$genres)
movies$genres[movies$genres == '(no genres listed)'] <- '(no_genres_listed)'

# Merge all data
d1 <- merge(movies, links, by.x = "movieId", by.y = "movieId", all.x = TRUE)
d2 <- merge(d1, ratings, by.x = "movieId", by.y = "movieId", all.x = TRUE)
# d3 <- merge(d2, tags, by.x = c("movieId","userId"), by.y = c("movieId","userId"), all.x
= TRUE)


### Levels of data for the datasets above
# --> links is at Movie level
# --> movies is at Movie level
# --> ratings is at Movie,user level
# --> tags is at Movie,user,tag level
# --> d1 is at Movie level
```

```
# --> d2 is at Movie,user level


# Typecasting
d2$title <- as.character(d2$title)
d2$genres <- as.character(d2$genres)
# d2$tag <- as.character(d2$tag)
d2$release_dt <- as.integer(substr(d2$title, nchar(d2$title)-4, nchar(d2$title)-1))
```

NOTE:

Some of the movies do not have the release year mentioned against their name. Since there are very few of them, they have been ignored in the following analysis.

Data wrangling

1. Average rating for each movie released in or after 1996

```
m1996 <- d2[d2$release_dt >= 1996,]
# Keeping necessary columns
m1996 <- m1996[c("title", "rating")]        # Filtering by name in this situation. Mostly done by index
m1996_ratings <- aggregate(m1996, by = list(m1996$title), FUN = mean, na.rm = TRUE)

# Clearing memory space allocated to temporary variables
m1996_ratings$title <- NULL
m1996 <- NULL
```

2. Top 5 most reviewed movies every year after 1994

```
r1994 <- d2[(d2$release_dt > 1994 & !is.na(d2$rating)), ]
# Removing NAs that are intrduced because some movies do not have missing dates
r1994 <- r1994[!is.na(r1994$release_dt),]
# Keeping necessary columns (not really necessary to do this)
r1994 <- r1994[c("title", "movieId", "release_dt")]
r1994_review <- aggregate(movieId ~ title + release_dt, data = r1994, FUN = length)
names(r1994_review)[3] <- "num_ratings"

# Getting top 5
ans <- r1994_review %>% group_by(release_dt) %>% top_n(n = 5, wt = num_ratings)
# Imp NOTE: The no. of rows in the answer is not a multiple of 5 as there are movies with the same no. of ratings
ans <- ans[order(ans$release_dt, -ans$num_ratings),]
ans <- ans %>%                                                  # Creating a rank required for plotting later
          group_by(release_dt) %>%
          mutate(rank_top5 = order(order(num_ratings, decreasing=TRUE)))


# Clearing memory space allocated to temporary variables
# r1994_review <- NULL
r1994 <- NULL
```

3. Average rating for “Drama”, “Romance” and “Drama and Romance” movies. This has been done using tags not genres - Can be done just as easily by genres

```
# Function to get the list of movies that match the tag(s)
mlist <- function(x)
{
  tags_temp <- tags[tags$tag %in% x ,]
  tags_temp$userId <- NULL

  temp <- spread(tags_temp, key = tag, value = timestamp)
  movie_list <- unlist(temp[complete.cases(temp),]$movieId)

  if(length(movie_list)==0){
    print("There are no movies that match all the following genres")
    print(x)
    return(NULL)
  } else {
    return(movie_list)
  }
}

result <- function(x)
{
  if(is.null(x)){
    print("There are no such movies")
    print(x)
    return()
  } else {
    genre_data <- d2[d2$movieId %in% tag_movie_list,]
    genre_data <- genre_data[c("title", "rating")]
    genre_rating <- aggregate(rating ~ title, genre_data, FUN = mean, na.rm = TRUE) #
Movie level ratings
    return(mean(genre_data$rating)) # Genre level rating
  }
}

# Drama
tag_list <- c("drama")
tag_movie_list <- mlist(tag_list)
result(tag_movie_list)

## [1] 4.150209

# Romance
tag_list <- c("romance")
tag_movie_list <- mlist(tag_list)
result(tag_movie_list)

## [1] 3.542184

# Drama and Romance
tag_list <- c("drama", "romance")
tag_movie_list <- mlist(tag_list)

## [1] "There are no movies that match all the following genres"
## [1] "drama"   "romance"
```

```
result(tag_movie_list)
```

```
## [1] "There are no such movies"
## NULL

## NULL
```

VERY IMP

NOTE: Tag matching has been done in terms of equality, not as a 'contains' eg: when the tag input is 'drama', movies with 'courtroom drama' have not been selected. But, the function can be easily modified for the 'contains' scenario as well.

NOTE: Both the functions can be integrated into one but the idea remains the same

4. Number of customers who rated a movie tagged as "horror" by year

```
# tags_in_question <- c("horror", "drama")
tags_in_question <- c("horror")
movie_list <- unlist(unique(tags[tags$tag %in% tags_in_question,]$movieId))

temp_df <- d2[d2$movieId %in% movie_list,]
yearly_users <- aggregate(userId ~ release_dt, temp_df, FUN = length)
print(paste("The yearly number of users that rated a movie with
the",tags_in_question,"tag(s) is/are"))
```

```
## [1] "The yearly number of users that rated a movie with the horror tag(s) is/are"
```

```
print(yearly_users)
```

```
##   release_dt userId
## 1       1986    126
## 2       1990     44
## 3       1991     25
## 4       2010     42
```

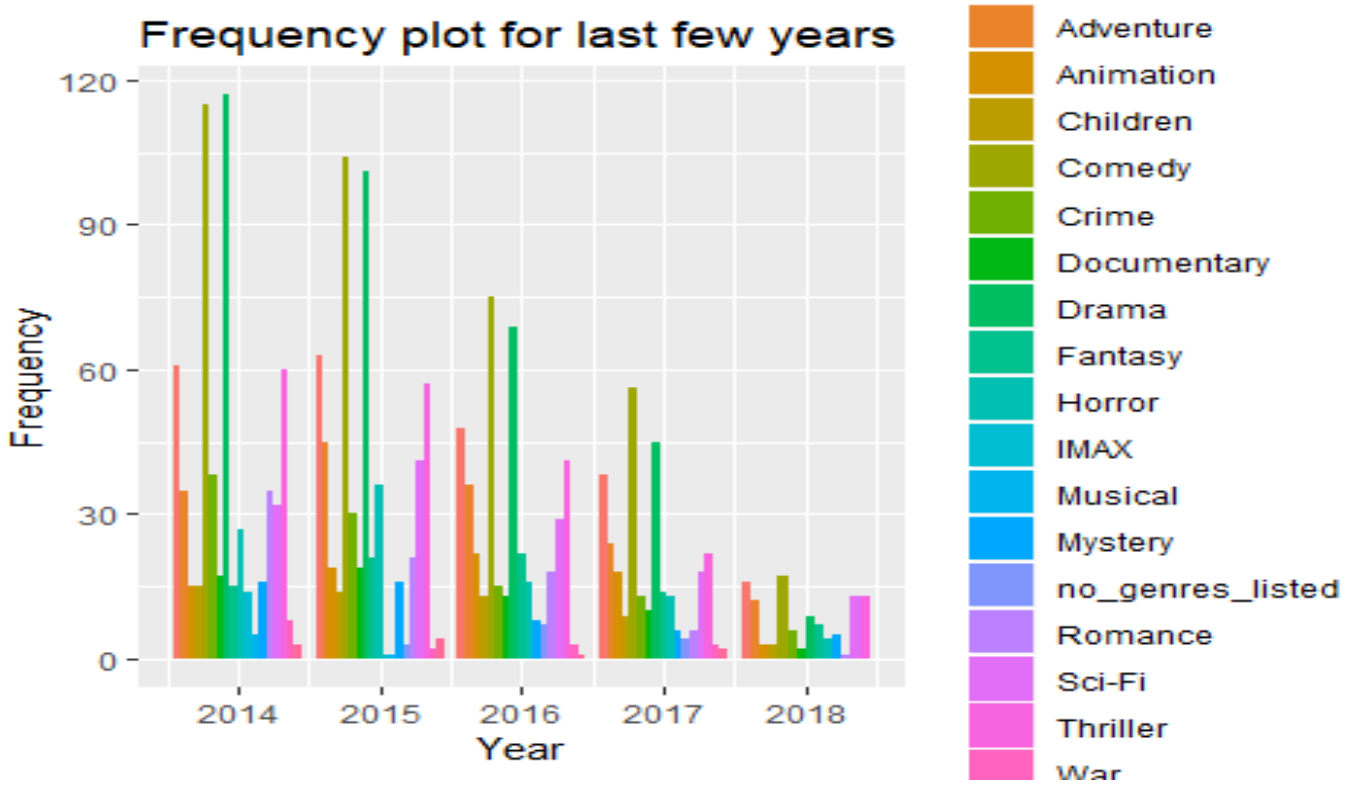
```
temp_df <- NULL
```

Data plotting

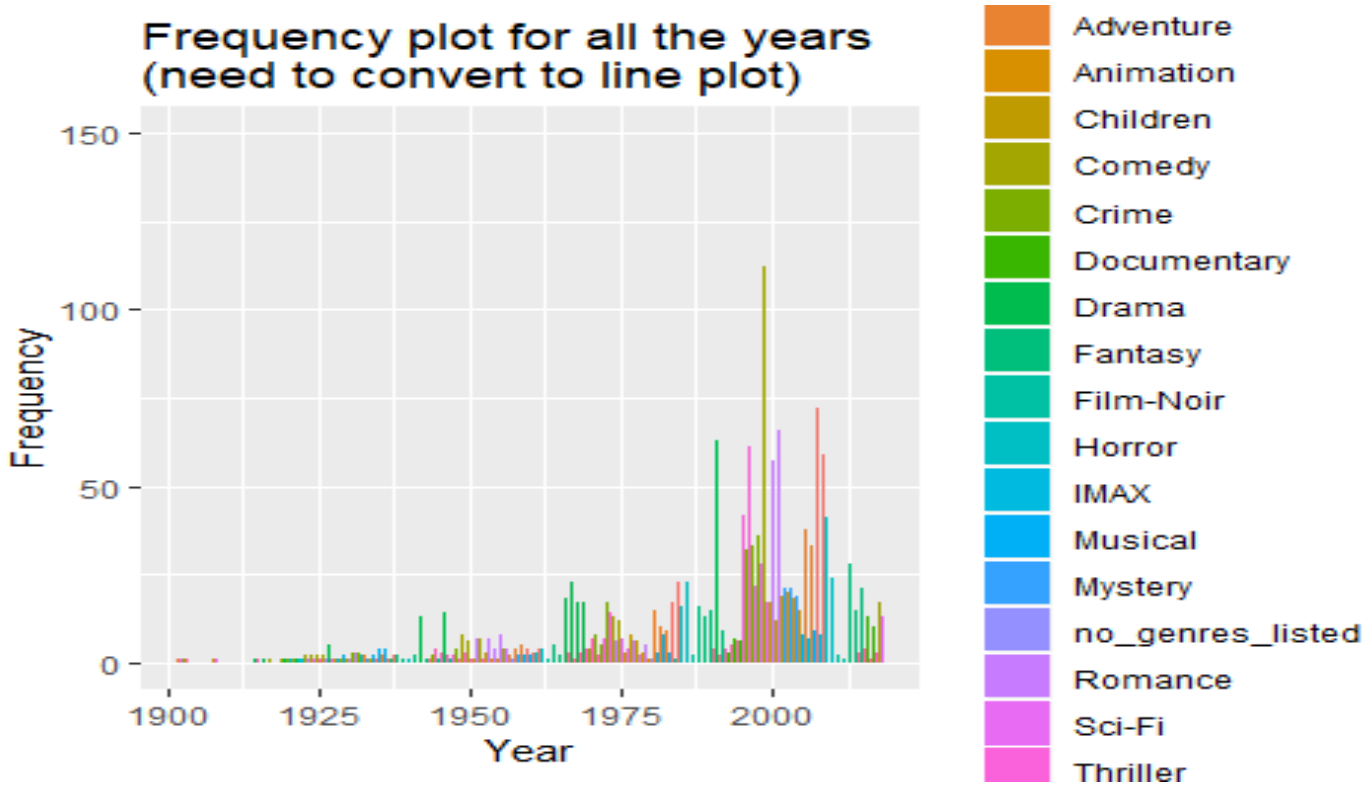
1. Trend of movie genres by the release years i.e. frequency of different genres of movies released each year. If a movie is across multiple genres then count them in all

```
# Using the splitstackshape package
# library(splitstackshape)

# transpose_fn <- function(x)
# {
#
#   temp <- x[c("movieId","genres")]
#   temp_wide <- cSplit(temp, "genres", sep="|")
#   temp_long <- melt(temp_wide, id.vars = c("movieId"))
#   temp_long <- temp_long[!is.na(temp_long$value),]
#   temp_long$variable <- NULL
#
#   # Add back movie title info
#   x <- merge(x, temp_long, by="movieId", all.x = TRUE)
#   x$genres <- as.character(x$genres)
#   x$title <- as.character(x$title)
#   x$genres <- NULL
#   names(x)[3] <- "genres"
#   return(x)
#
# }

# Using the stringr package
library(reshape2)

##
## Attaching package: 'reshape2'

## The following object is masked from 'package:tidyr':
##
##     smiths

transpose_fn <- function(x)
{
  temp <- movies[c("movieId","genres")]
  temp$genres <- as.character(temp$genres)
  abc <- strsplit(temp$genres,"|", fixed = TRUE)
  abc <- as.data.frame(str_split_fixed(abc,' ', 20))
  abc <- abc[, colSums(abc != "") != 0]
  abc <- as.data.frame(lapply(abc, gsub, pattern="c\\(", replacement=''))
  abc <- as.data.frame(lapply(abc, gsub, pattern='[() ",]', replacement=''))
  abc <- as.data.frame(apply(abc, 2, function(x) gsub("^$|^ $", NA, x)))
  abc[] <- lapply(abc, function(x) if(is.factor(x)) as.character(x) else x)
  temp_wide <- cbind(temp,abc)
  temp_wide$genres <- NULL
  temp_long <- melt(temp_wide, id.vars = c("movieId"))
  temp_long <- temp_long[!is.na(temp_long$value),]
  temp_long$variable <- NULL

  # Add back movie title info
  x <- merge(x, temp_long, by="movieId", all.x = TRUE)
```

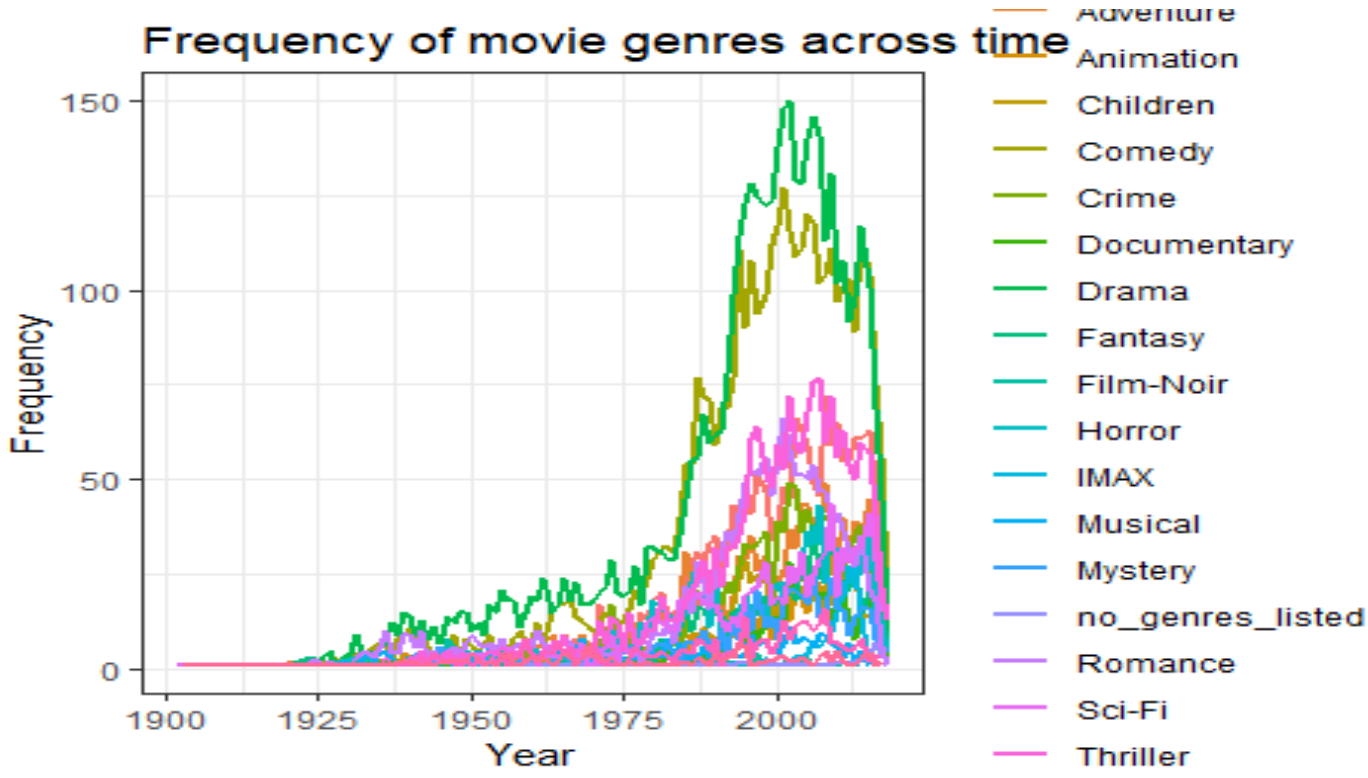
```
  x$genres <- as.character(x$genres)
  x$title <- as.character(x$title)
  x$genres <- NULL
  names(x)[3] <- "genres"
  return(x)

}



# Create data for plotting
df <- transpose_fn(movies)
df$year <- as.integer(substr(df$title, nchar(df$title)-4, nchar(df$title)-1))
df_plot <- aggregate(movieId ~ genres + year, data = df, FUN = length)
df_plot <- df_plot[order(-df_plot$year, df_plot$genres, -df_plot$movieId),]

## Plotting
library(ggplot2)

# Filtering the data for a few years to just visualize the plot
zzz <- df_plot[df_plot$year>2013,]
ggplot(zzz,aes(x = year,y = movieId, fill=genres)) +
  geom_bar(stat="identity", position = "dodge") +
  xlab("Year")+ylab("Frequency") + ggtitle("Frequency plot for last few years")
```




```
# Plot for the entire data
ggplot(df_plot,aes(x = year,y = movieId, fill=genres)) +
  geom_bar(stat="identity", position = "dodge") +
  xlab("Year")+ylab("Frequency") + ggtitle("Frequency plot for all the years \n(need to
convert to line plot)")
```



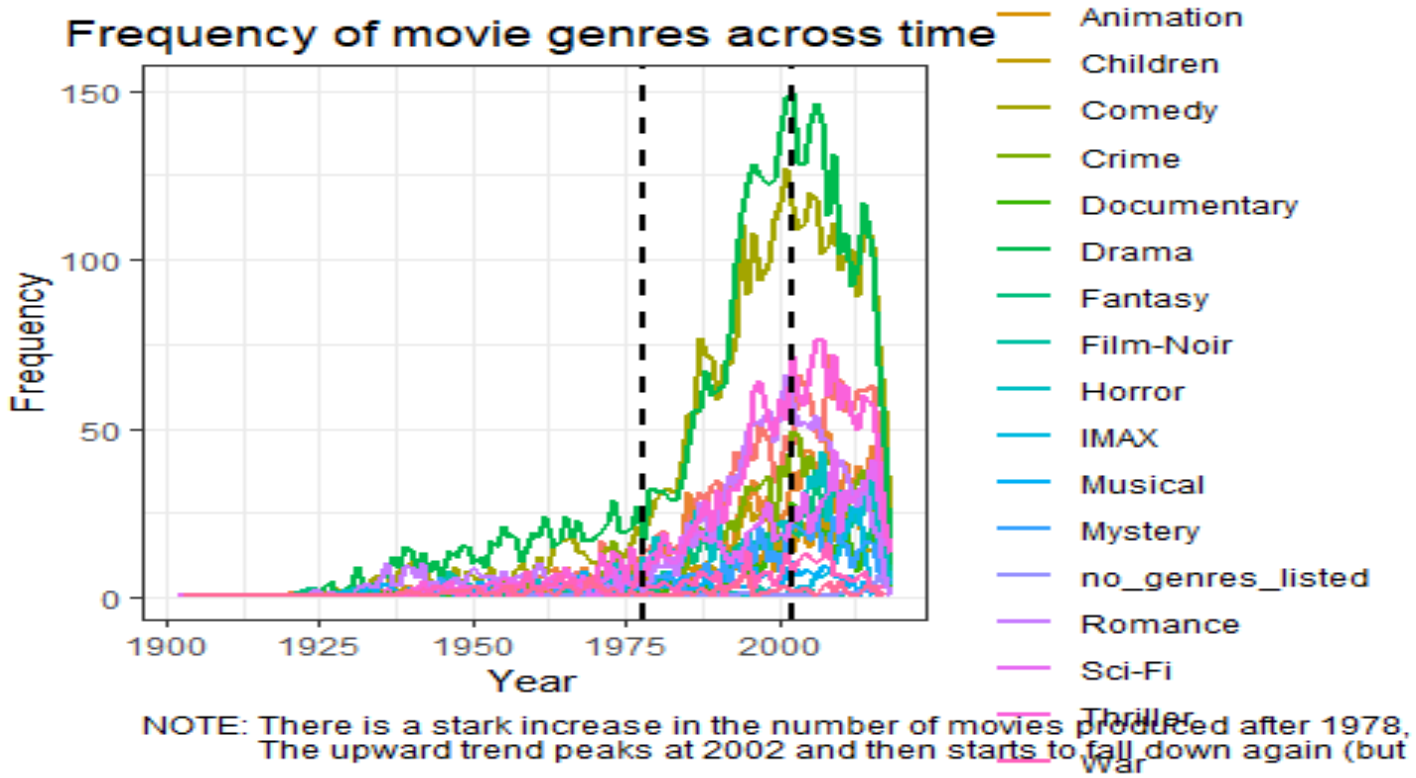

```
## --> A bar graph is not the best way to visualize the trend. Using a line graph below
ggplot(df_plot, aes(year, movieId, group = genres, colour = genres)) +
  geom_line(size = 1) + theme_bw() +
  xlab("Year")+ylab("Frequency") + ggtitle("Frequency of movie genres across time")
```



2. Is there a shift in trend?

```
## --> Yes there is
ggplot(df_plot, aes(year, movieId, group = genres, colour = genres)) +
  geom_line(size = 1) + theme_bw() +
```

```
  xlab("Year")+ylab("Frequency") + ggtitle("Frequency of movie genres across time") +
  geom_vline(xintercept=1978, linetype="dashed", size = 0.75) +
  geom_vline(xintercept=2002, linetype="dashed", size = 0.75) +
  theme(plot.title = element_text(hjust = 0.5),
        plot.caption = element_text(hjust = 0)) +
  labs(caption = "NOTE: There is a stark increase in the number of movies produced after
1978, with the 'Drama' and 'Comedy' genres appearing the most.
             The upward trend peaks at 2002 and then starts to fall down again (but it
might be due to incomplete data)."
       , face="bold", size=5)
```



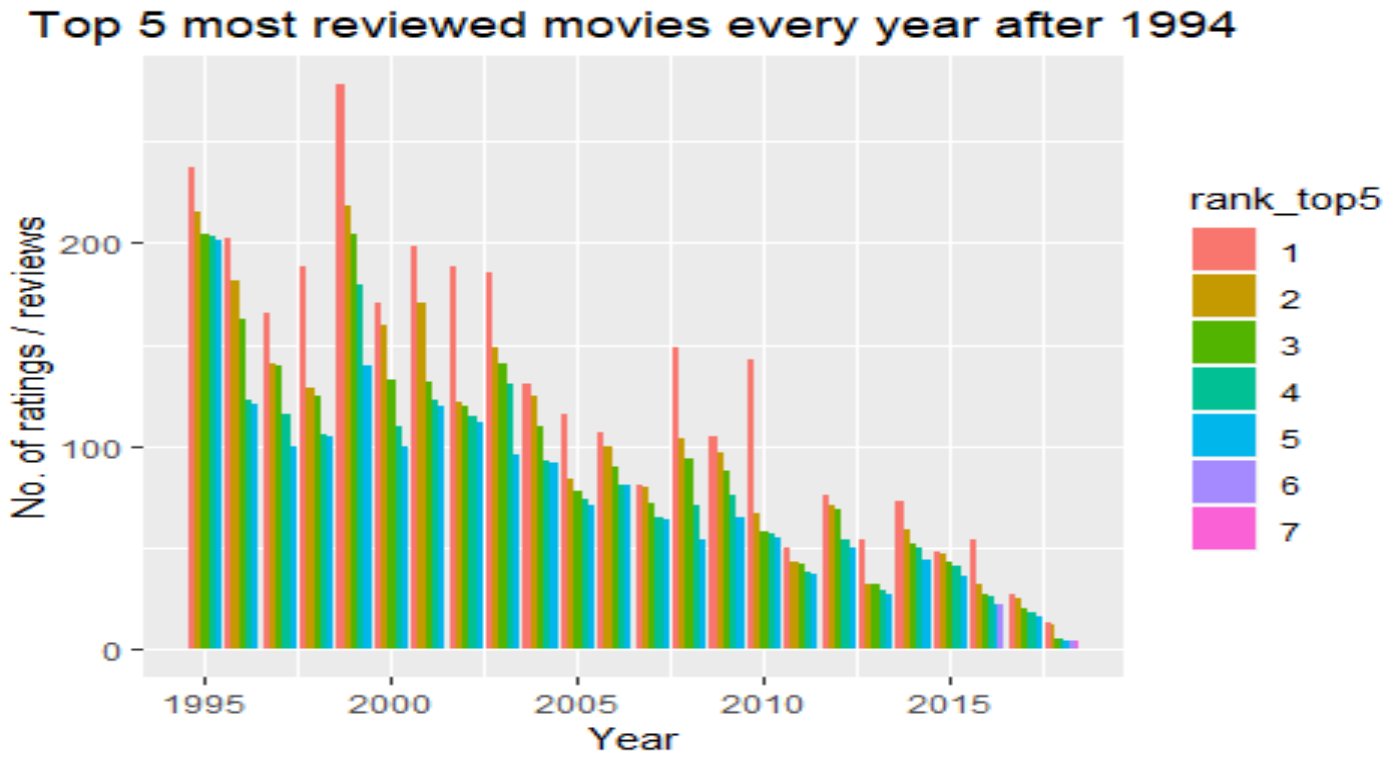
3. Top 5 most reviewed movies every year after 1994 - would like to see all the years plotted at one go

```
head(ans)
```

```
## # A tibble: 6 x 4
## # Groups:   release_dt [2]
##   title                                release_dt num_ratings rank_top5
##   <chr>                                     <int>       <int>     <int>
## 1 Braveheart (1995)                          1995         237         1
## 2 Toy Story (1995)                           1995         215         2
## 3 Usual Suspects, The (1995)                 1995         204         3
## 4 Seven (a.k.a. Se7en) (1995)                1995         203         4
## 5 Apollo 13 (1995)                           1995         201         5
## 6 Independence Day (a.k.a. ID4) (1996)       1996         202         1
```

```
ans_temp <- ans
ans_temp$rank_top5 <- as.character(ans_temp$rank_top5)

ggplot(ans_temp,aes(x = release_dt,y = num_ratings, fill=rank_top5)) +
  geom_bar(stat="identity", position = "dodge") +
  theme(plot.title = element_text(hjust = 0.5)) +
  # geom_text(aes(label=title), position=position_dodge(width=1), vjust=-0.25, angle=-90)
```

```
+
  xlab("Year")+ylab("No. of ratings / reviews") + ggtitle("Top 5 most reviewed movies
every year after 1994")
```



VERY IMP

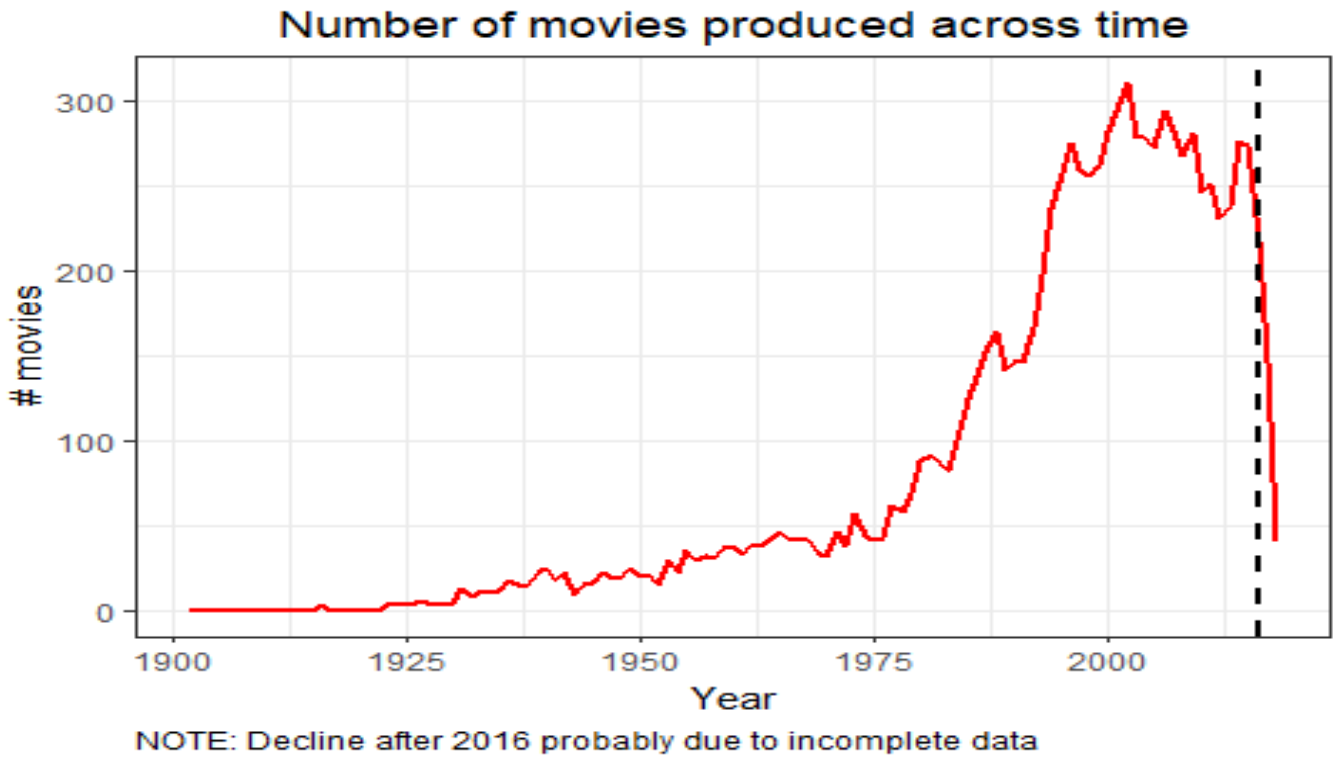
NOTE: 1) The rankings go from 1 to 7 (not 1 to 5) because year 2016 and 2018 have more than 1 movie with the same number of reviews

2) This chart does not show the movie names. They can be sbown by uncommenting the geom_text() line in the above code. This is not recommended as the plot becomes too cluttered and unreadable. This issue can be resolved by displaying the movie name when you hover over a bar using the mouse. This cannot be done in ggplot, but can be done using the plotly library

What else can you do with the data?

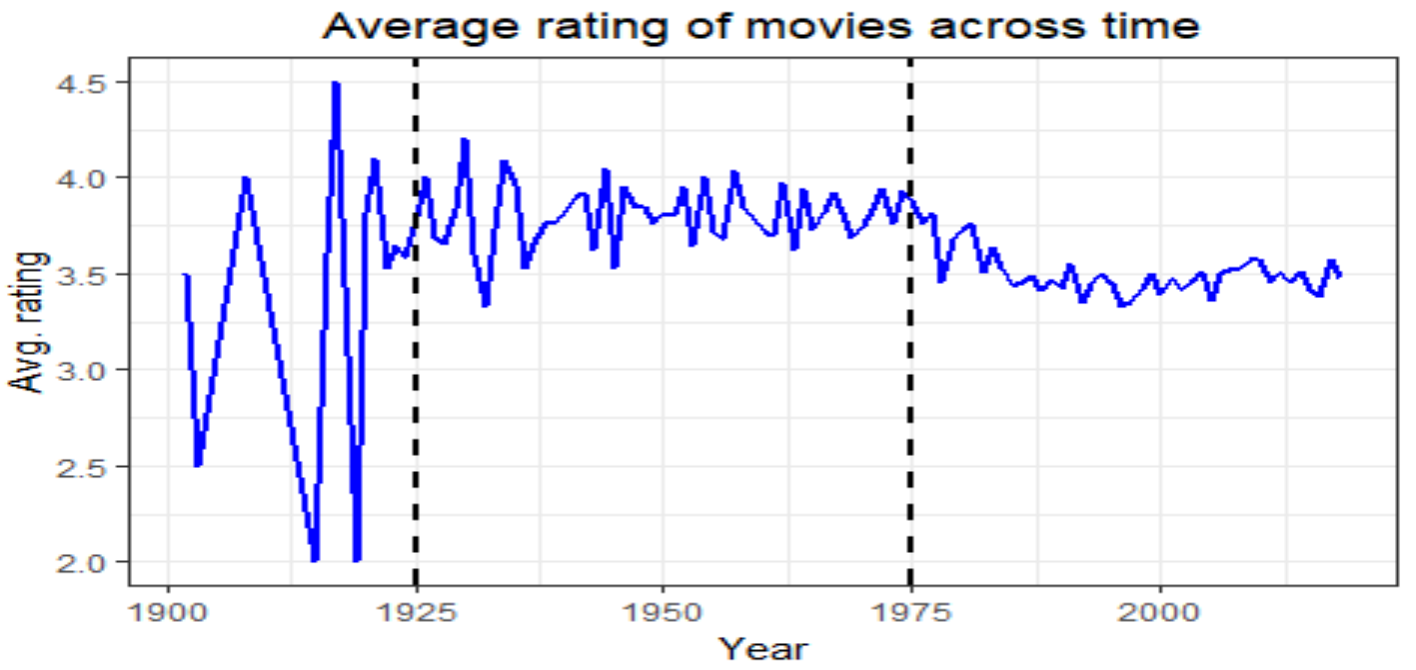
4. What was the golden age of cinema? - i.e. Comparing movie trends (number of movies) across years and Average rating of movies across time

```
library(gtable)
library(grid)

temp_movies <- movies
temp_movies$title <- as.character(temp_movies$title)
temp_movies$year <- as.integer(substr(temp_movies$title, nchar(temp_movies$title)-4,
nchar(temp_movies$title)-1))
temp_movies_agg <- aggregate(title ~ year, data = temp_movies, FUN = length)

df_temp <- d2[(!is.na(d2$rating) & !is.na(d2$release_dt)), ]
df_temp <- aggregate(rating ~ release_dt, data = df_temp, FUN = mean, na.rm = TRUE)

# Individual plots
#1 No. of movies
ggplot(temp_movies_agg, aes(year, title)) +
  geom_line(size = 1, colour = "red") + theme_bw() +
  xlab("Year")+ylab("# movies") + ggtitle("Number of movies produced across time") +
  geom_vline(xintercept=2016, linetype="dashed", size = 0.75) +
  theme(plot.title = element_text(hjust = 0.5),
        plot.caption = element_text(hjust = 0)) +
  labs(caption = "NOTE: Decline after 2016 probably due to incomplete data"
       , face="bold", size=5)
```

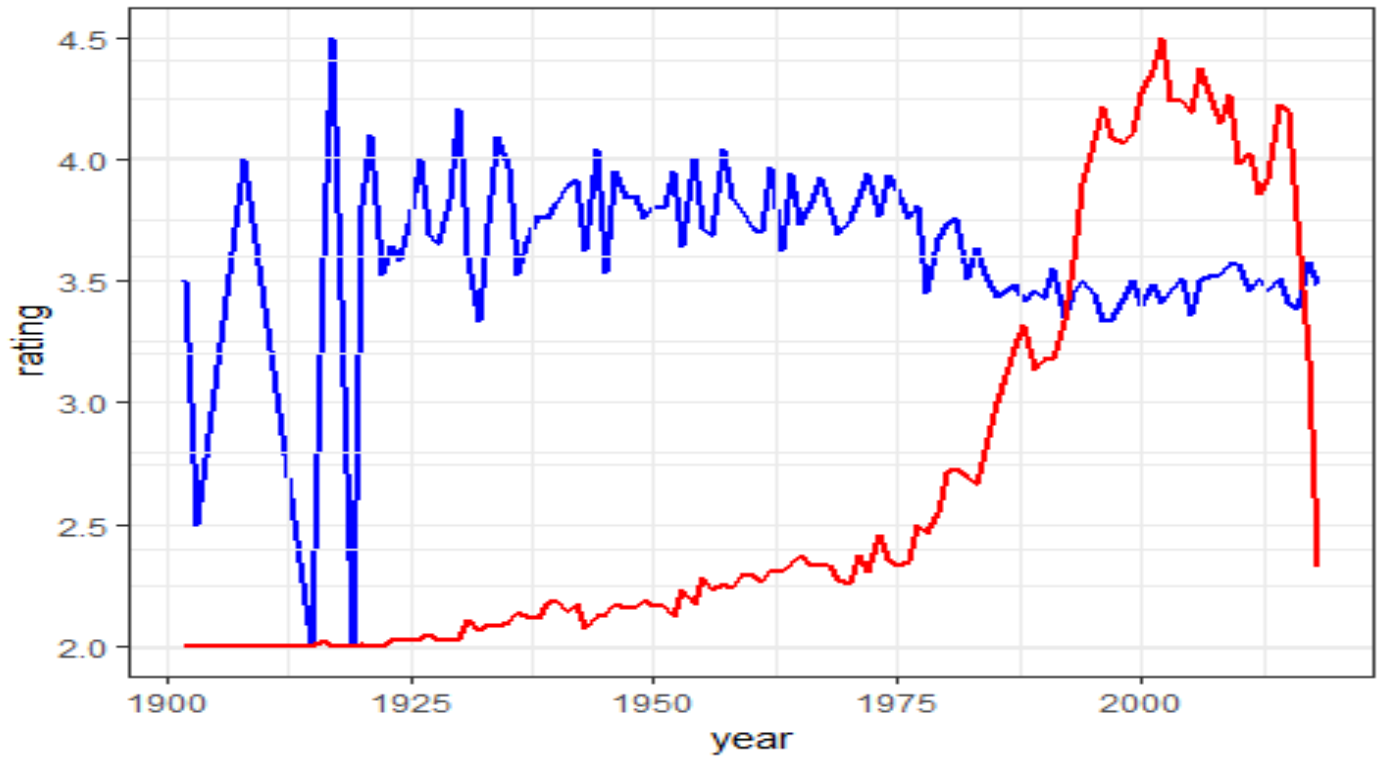



```
#2 Average rating
ggplot(df_temp, aes(release_dt, rating)) +
  geom_line(size = 1, colour = "blue") + theme_bw() +
  xlab("Year")+ylab("Avg. rating") + ggtitle("Average rating of movies across time") +
  geom_vline(xintercept=1925, linetype="dashed", size = 0.75) +
```

```
  geom_vline(xintercept=1975, linetype="dashed", size = 0.75) +
  theme(plot.title = element_text(hjust = 0.5),
        plot.caption = element_text(hjust = 0)) +
  labs(caption = "1) The plot shows extreme fluctuations before 1925 because there are
very few movies produced in those years
                  \n2) There seems to be a small drop in average rating after 1975"
       , face="bold", size=5)
```




```
# Combining into one plot
df_temp <- cbind(df_temp, temp_movies_agg)

grid.newpage()
# Two plots
p1 <- ggplot(df_temp, aes(year, rating)) + geom_line(size = 1, colour = "blue") +
theme_bw()
p2 <- ggplot(df_temp, aes(year, title)) + geom_line(size = 1, colour = "red") +
theme_bw() %+replace%
  theme(panel.background = element_rect(fill = NA))

# Extract gtable
g1 <- ggplot_gtable(ggplot_build(p1))
g2 <- ggplot_gtable(ggplot_build(p2))

# Overlapping the panel of 2nd plot on that of the 1st plot
pp <- c(subset(g1$layout, name == "panel", se = t:r))
g <- gtable_add_grob(g1, g2$grobs[[which(g2$layout$name == "panel")]], pp$t, pp$l, pp$b,
pp$l)

# Drawing
grid.draw(g)
```

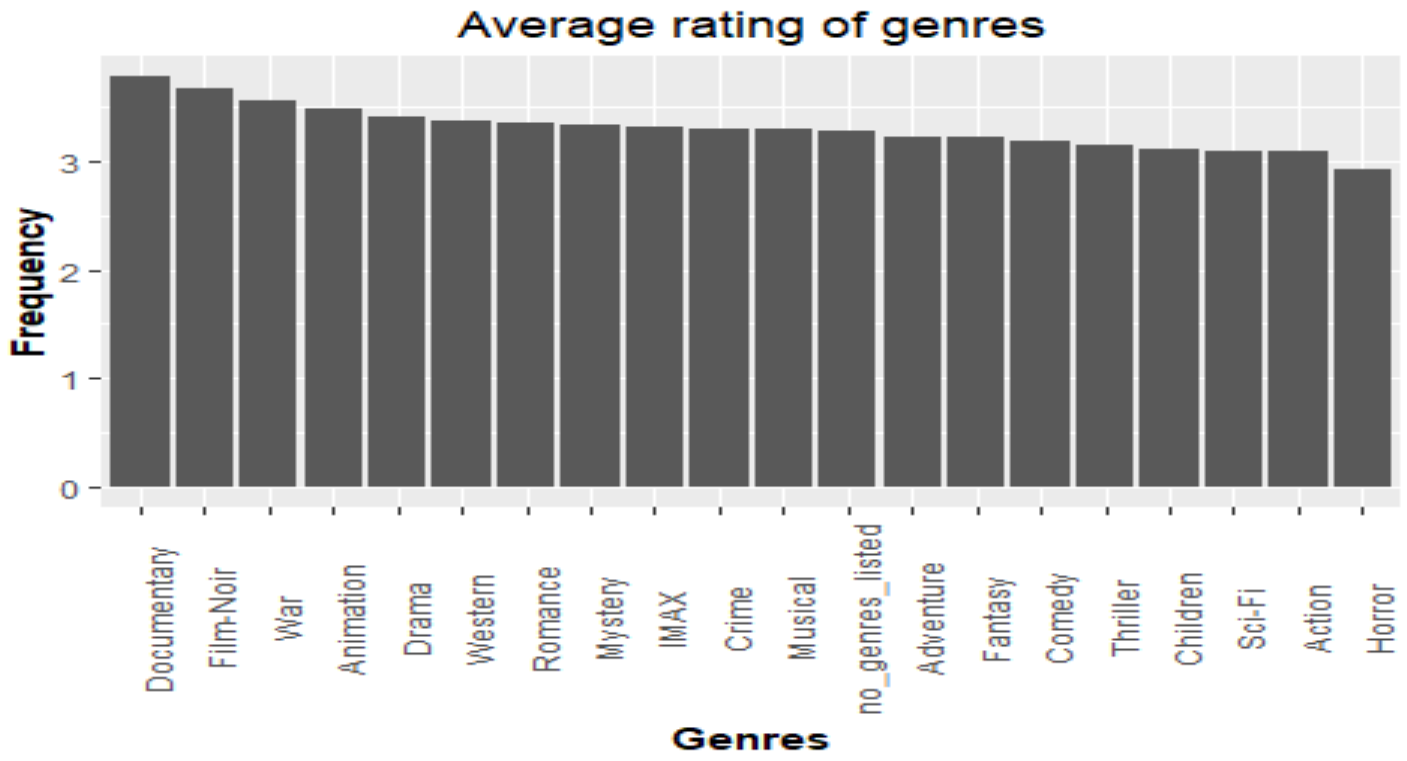



```
# <Just an extra line>
```

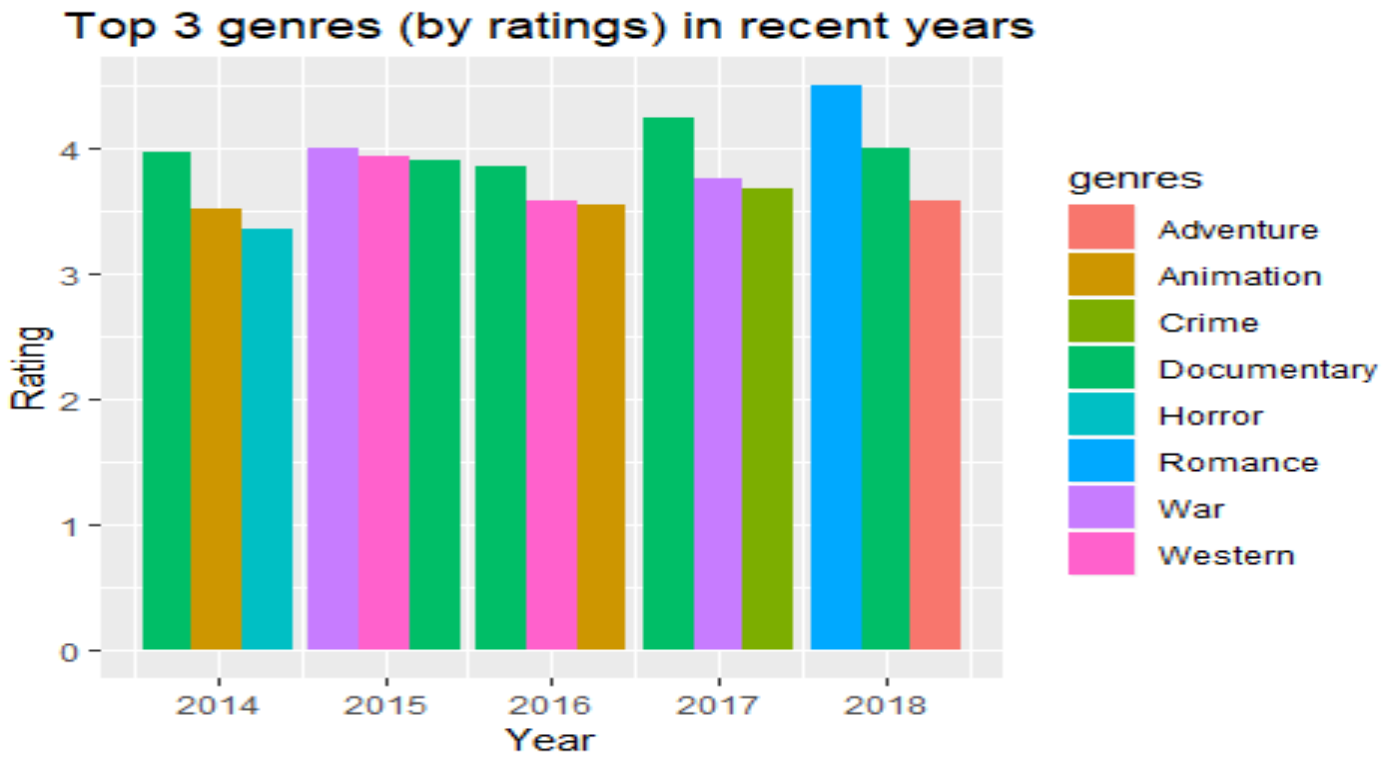
NOTE: The convoluted method of combining graphs has to be done because the scales of both the metrics are not comparable. If they are comparable, then you can just use two geom_line() commands in the same plot

5. Which genres are the best (in terms of ratings)?

```
mov_ratings <- aggregate(rating ~ title, data = d2, FUN = mean, na.rm = TRUE)
df_temp <- merge(df, mov_ratings, by.x = "title", by.y = "title", all.x = TRUE)
df_temp <- df_temp[(!is.na(df_temp$rating) & !is.na(df_temp$year)), ]

# Overall
df_temp1 <- aggregate(rating ~ genres, data = df_temp, FUN = mean, na.rm = TRUE)
df_temp1 <- df_temp1[order(-df_temp1$rating),]
ggplot(df_temp1,aes(x = reorder(genres, -rating),y = rating), fill = genres) +
  geom_bar(stat="identity", position = "dodge") +
  xlab("Genres")+ylab("Frequency") + ggtitle("Average rating of genres") +
  theme(plot.title = element_text(hjust = 0.5),
        axis.title.x = element_text(face="bold"),
        axis.text.x = element_text(angle = 90),
        axis.title.y = element_text(face="bold"))
```



```r
# Popular genres in the recent years (in terms of ratings) - User prefrences
df_temp2 <- aggregate(rating ~ genres + year, data = df_temp, FUN = mean, na.rm = TRUE)
df_temp2 <- df_temp2[order(-df_temp2$rating),]
df_temp2 <- df_temp2 %>% group_by(year) %>% top_n(n = 3, wt = rating)
temp <- df_temp2[df_temp2$year>2013,]
temp <- temp[order(temp$year, -temp$rating),]
temp$Id=rep(c(1:3),5)     # This si created to sort the individual groups of genres in
each year

ggplot(temp,aes(x = year,y = rating, group = Id, fill=genres)) +
  geom_bar(stat="identity", position = "dodge") +
  xlab("Year")+ylab("Rating") + ggtitle("Top 3 genres (by ratings) in recent years") +
  theme(plot.title = element_text(hjust = 0.5),
        plot.caption = element_text(hjust = 0))
```



6. Most Frequently occuring tags

```
tag_freq <- aggregate(userId ~ tag, data = tags, FUN = length)
names(tag_freq)[2] <- "Tag_Freq"
temp <- tag_freq %>% top_n(n = 10, wt = Tag_Freq)

ggplot(temp,aes(x = reorder(tag, -Tag_Freq),y = Tag_Freq)) +
  geom_bar(stat="identity", position = "dodge") +
  xlab("Tag")+ylab("Frequency") + ggtitle("Frequently occuring tags") +
  theme(plot.title = element_text(hjust = 0.5),
        axis.title.x = element_text(face="bold"),
        axis.text.x = element_text(angle = 90),
        axis.title.y = element_text(face="bold"))
```