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

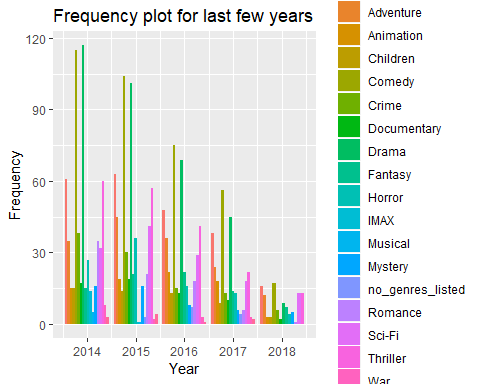
#### 1. Trend of movie genres by the release years i.e. frequency of different genres of movies released each year. If a movie is across multiple genres then count them in all

# 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)

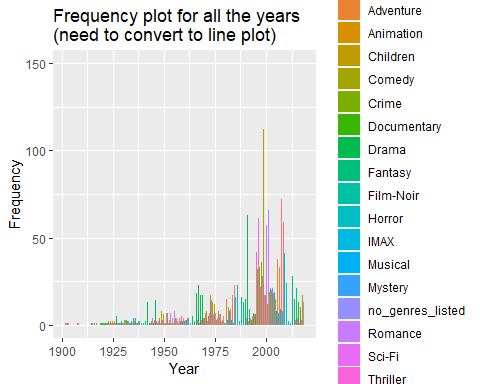
##   
## Attaching package: 'reshape2'

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

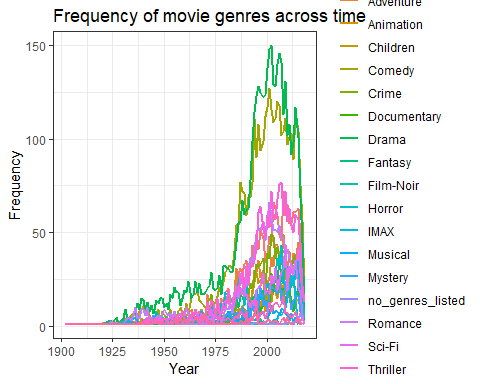
transpose\_fn <- function(x)  
{  
 temp <- movies[c("movieId","genres")]  
 temp$genres <- as.character(temp$genres)  
 abc <- strsplit(temp$genres,"|", fixed = TRUE)  
 abc <- as.data.frame(str\_split\_fixed(abc,' ', 20))  
 abc <- abc[, colSums(abc != "") != 0]  
 abc <- as.data.frame(lapply(abc, gsub, pattern="c\\(", replacement=''))  
 abc <- as.data.frame(lapply(abc, gsub, pattern='[() ",]', replacement=''))  
 abc <- as.data.frame(apply(abc, 2, function(x) gsub("^$|^ $", NA, x)))  
 abc[] <- lapply(abc, function(x) if(is.factor(x)) as.character(x) else x)  
 temp\_wide <- cbind(temp,abc)  
 temp\_wide$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)")

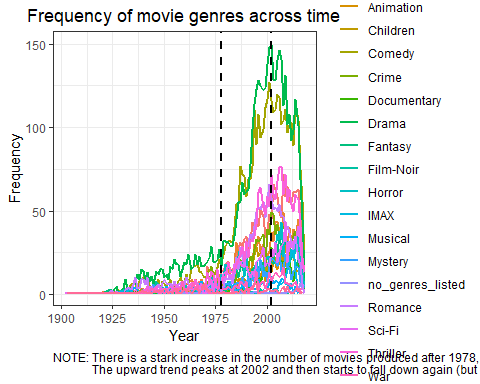


## --> A bar graph is not the best way to visualize the trend. Using a line graph below  
ggplot(df\_plot, aes(year, movieId, group = genres, colour = genres)) +   
 geom\_line(size = 1) + theme\_bw() +   
 xlab("Year")+ylab("Frequency") + ggtitle("Frequency of movie genres across time")



#### 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)

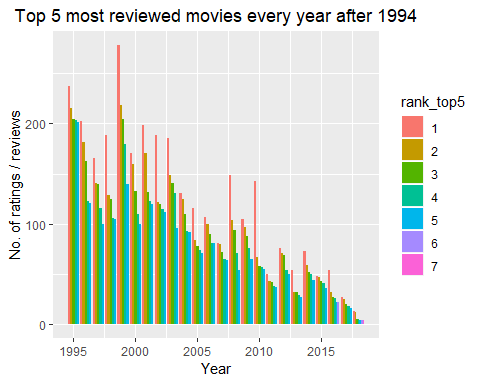


#### 3. Top 5 most reviewed movies every year after 1994 - would like to see all the years plotted at one go

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

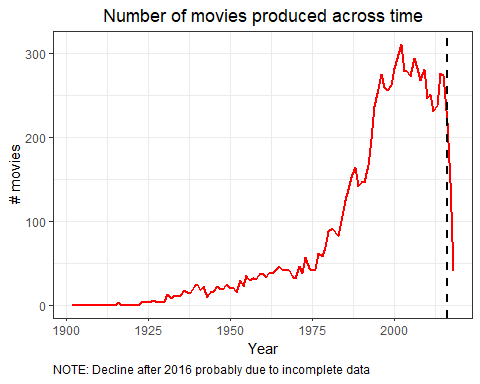
##### NOTE: 1) The rankings go from 1 to 7 (not 1 to 5) because year 2016 and 2018 have more than 1 movie with the same number of reviews

##### 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

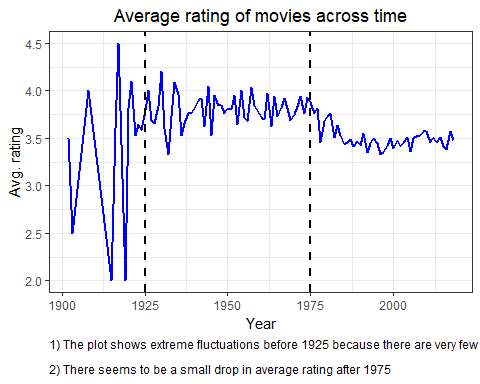
### What else can you do with the data?

#### 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

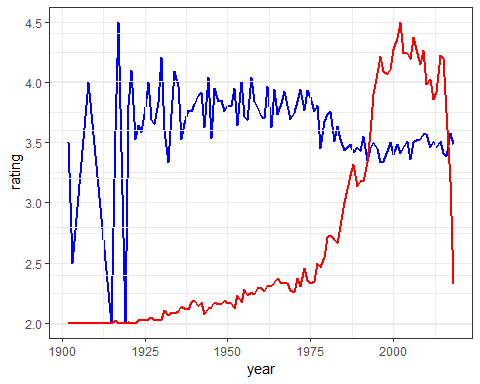
library(gtable)  
library(grid)  
  
temp\_movies <- movies  
temp\_movies$title <- as.character(temp\_movies$title)  
temp\_movies$year <- as.integer(substr(temp\_movies$title, nchar(temp\_movies$title)-4, nchar(temp\_movies$title)-1))  
temp\_movies\_agg <- aggregate(title ~ year, data = temp\_movies, 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)



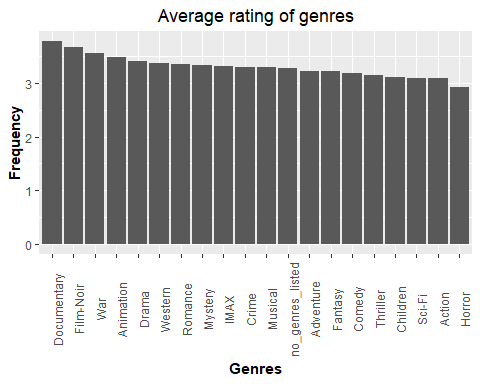
# <Just an extra line>

##### 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

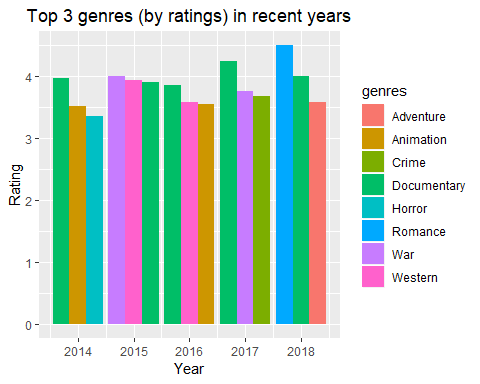
#### 

#### 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"))



# 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"))

