Proyecto Netflix Movies MADM

Laura Moreno, Josep Roman, Paul Ramírez

11/28/2020

Contenidos

1 Objetivo

2 Data Wrangle

2.1 Importación de datos

2.1.1 Importación datos puntuaciones películas

Info de los archivos "combined_data_.txt" The first line of each file contains the movie id followed by a colon. Each subsequent line in the file corresponds to a rating from a customer and its date in the following format:

UserID,Rating,Date

- MovieIDs range from 1 to 17770 sequentially.
- UserIDs range from 1 to 2649429, with gaps. There are 480189 users.
- Ratings are on a five star (integral) scale from 1 to 5.
- Dates have the format YYYY-MM-DD.

Selección de 250 películas de manera aleatoria Utilizamos el código de Ricardo para seleccionar nuestras 250 peliculas con las siguientes modificaciones:

```
filas_ID_combined_all = read.csv(here("Data","filas_ID_combined_all.txt"))
set.seed(081034)
n_filas = nrow(filas_ID_combined_all)
muestra_grupo = sample(1:n_filas, 250, replace=F)
pelis <- filas_ID_combined_all[as.vector(muestra_grupo),]
```

Cargamos los 4 archivos originales con las puntuaciones:

```
attach(pelis)

data1 = read_tsv(here("Raw data","combined_data_1.txt"),col_names = FALSE)
data2 = read_tsv(here("Raw data","combined_data_2.txt"),col_names = FALSE)
data3 = read_tsv(here("Raw data","combined_data_3.txt"),col_names = FALSE)
data4 = read_tsv(here("Raw data","combined_data_4.txt"),col_names = FALSE)
```

Generamos un tibble vacío, y en función del archivo en el que se encuentre la pelicula, vamos añadiendo en `scores` las filas correspondientes a nuestras películas:

```
scores = tibble()
for(i in 1:nrow(pelis)){
  if (data[i]==1){
    scores = rbind(scores,data1[fila[i]:fila_final[i],])
  }
  else if (data[i]==2){
    scores = rbind(scores,data2[fila[i]:fila_final[i],])
  }
  else if (data[i]==3){
    scores = rbind(scores,data3[fila[i]:fila_final[i],])
  }
  else {
    scores = rbind(scores,data4[fila[i]:fila_final[i],])
  }
}
```

Guardamos un csv con solo nuestras 250 películas en el formato original

```
write_csv(scores, here("Data", "nuestras_pelis_raw.csv"))
```

Carga archivo puntuaciones de nuestras 250 películas Cargamos el csv generado en el paso anterior:

```
aux = read_csv(here("Data", "nuestras_pelis_raw.csv"), col_names = T)
```

2.1.2 Importación datos información sobre las películas

Carga archivo titulos películas

```
#rm(titles)
#algunas peliculas tienen una coma en su nombre, así que cargamos primero todo como una única columma, para luego dividirlo en 3,
titles = read_table(here("Data",'movie_titles_raw.csv'), col_names=F) %>%
  separate(col = 1, into = c("MovieID", "Release_Year", "Title"), sep = ",", extra = "merge")
```

2.2 Limpieza de los datos

2.2.1 Limpieza datos puntuaciones películas

Aplicamos el código de Ricardo para limpiar el dataframe `aux` y pasar al dataframe `scores` con una fila para cada valoración de usuario

Reorganizamos variables y asignamos tipos de variable:

```
#Reorganización
scores %<>% relocate(MovieID, UserID, Date, Score)
#Asignación del tipo de dato
scores %<>% mutate(across(c(MovieID:UserID, Score), as.integer))
scores %<>% mutate(Date = as.Date(Date))
```

2.2.2 Limpieza datos títulos películas

```
head(titles)
```

```
## # A tibble: 6 x 3
##   MovieID Release_Year Title
##   <chr>   <chr>        <chr>
## 1 1       2003         Dinosaur Planet
## 2 2       2004         Isle of Man TT 2004 Review
## 3 3       1997         Character
## 4 4       1994         Paula Abdul's Get Up & Dance
## 5 5       2004         The Rise and Fall of ECW
## 6 6       1997         Sick
```

```
titles  %<>% mutate(across(c(MovieID:Release_Year), as.integer))
```

2.2.3 Join de ´scores´ con ´titles´

Hacemos un **left join con** de *scores* con *titles* para añadir a la primera los títulos de cada película y el año en que se publicaron

- El `left_join` se queda con todas las observaciones que aparecen en el primer *dataset*, es decir, solo tendrá en cuenta las películas observadas en *scores*.

- El `join` entre tablas lo hemos hecho con la columna `MovieID`, presente en ambas tablas. Tal y como vemos en la tabla `movies_titles.csv`, cada película tiene un `MovieID` único, lo que se conoce como *clave primaria*. No obstante, en la tabla `scores` cada MovieID puede ser puntuada por varios `UserID`, en este caso, la *clave primaria* se constituye a partir de la combinación de ambas variables.

```
scores %<>% left_join(titles, by = 'MovieID')
summary(scores)
kable(head(scores))
```

2.2.4 Exportación datos limpios

Exportamos el archivo csv limpio para trabajar con el a partir de ahora

```
write_csv(scores,here("Data", "nuestras_pelis.csv"))
```

2.2.5 Importación datos limpios para analizar en la sección Estadística Descriptiva

```
scores = read_csv(here("Data","nuestras_pelis.csv"))

# Cambiamos los tipos de variable necesarios
scores %<>% mutate(across(c(MovieID,UserID,Score,Release_Year), as.integer))
```

3 Estadística Descriptiva

Vemos que tenemos información de la peliculas 1 a la 15, y las puntuaciones se hicieron entre el 2000 y el 2005 (mayoritariamente en 2005). Distribución de los meses y dias en que se puntuo es uniforme.

Veamos más informacion sobre los datos:

```
length(unique(scores$UserID)) #20537 usuarios distintos
```

```
## [1] 327577
```

```
table(scores$Score) # frecuencia puntuaciones
```

```
##
##      1      2      3      4      5
##  76876 167088 455242 515741 293945
```

```
table(head(scores$MovieID)) # frecuencia title
```

```
##
## 515
##   6
```

3.1 Pregunta 1

Justifica para cada una de las variables de la tabla anterior el tipo de dato que mejor se ajusta a cada una de ellas: numérico, ordinal, categórico. . . .

```
glimpse(scores)
```

```
## Rows: 1,508,892
## Columns: 6
## $ MovieID      <int> 515, 515, 515, 515, 515, 515, 515, 515, 515, 515, 515,...
## $ UserID       <int> 2295232, 1560318, 2550394, 1502043, 1507284, 771626, 1...
## $ Date         <date> 2005-08-16, 2005-10-04, 2005-11-01, 2005-08-15, 2005-...
## $ Score        <int> 1, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, ...
## $ Release_Year <int> 2005, 2005, 2005, 2005, 2005, 2005, 2005, 2005, 2005, ...
## $ Title        <chr> "Avia Vampire Hunter", "Avia Vampire Hunter", "Avia Va...
```

Variables tipo *int*: MovieID, CustomerID, Score, Release_Year - *UserID*: Contiene un número entero, estos son objetos que contienen un único campo, un identificado ID para cada cliente, no queremos duplicados. - *MovieID*: Contiene un número entero, estos son objetos que contienen un único campo, un identificado ID para cada película, no queremos duplicados. Un integer es inmutable. No obstante, cuando creemos gráficas los vamos a transformar a *chr* para evitar que los ejes escalen los números. - *Release_Year*: No existen años con decimales, por lo tanto utilizar variables para datos enteros seria suficiente. - *Score*: Las puntuaciones son números enteros del 1 al 5. Las películas no aceptan decimales como puntuación.

Variables tipo *date*: Date - *Date* : esta variable incluye datos de tipo fecha (YY/MM/DD) por ello lo más adecuado es tratarlo como una variable de este tipo. Gracias a esto, podemos aplicar paquetes como *lubridate* para manipular fechas.

Variables tipo *chr*: Title - *Title*: Utilizamos el tipo carácter porque nos interesan objetos que representan un conjunto de letras.

3.2 Pregunta 2

Estudia la distribución del numero de películas estrenadas por año. Realiza un gráfico que muestre esta distribución haciendo los ajustes necesarios (agrupaciones, cambios de escala, transformaciones. . .)

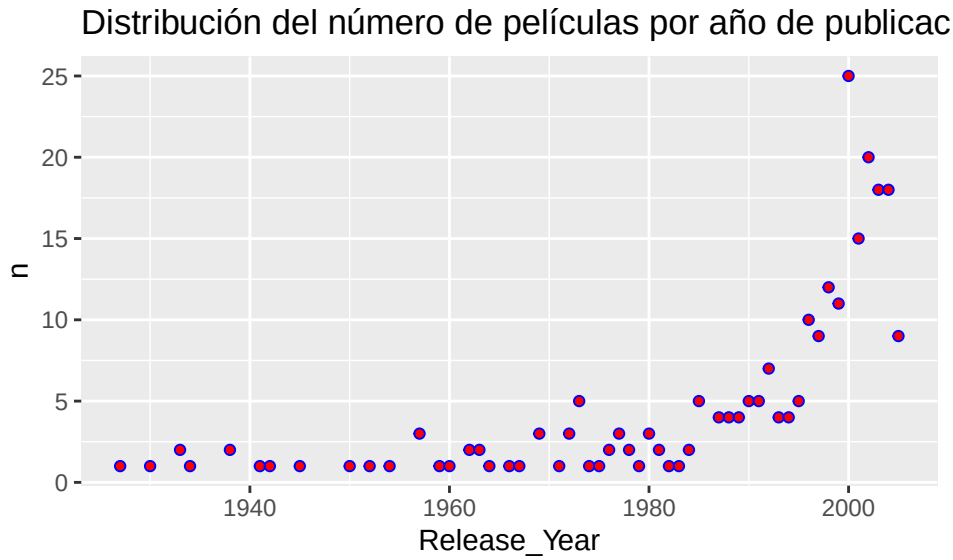
```
#id_year <- group_by(scores, MovieID) %>%
  #summarise(Release_Year = unique(Release_Year))
#movies_per_year <- id_year %>% group_by(Release_Year) %>%
  #summarise(Count_Movies = n_distinct(MovieID))

nuestros_movie_ids <- tibble(MovieID = unique(scores$MovieID))
nuestras_titles <- titles %>%
  right_join(nuestros_movie_ids, by = "MovieID")

movies_per_year <- nuestras_titles %>%
  group_by(Release_Year) %>%
  summarise(n = n())

summary(movies_per_year)
```

```
##   Release_Year        n
##  Min.   :1927   Min.   : 1.00
##  1st Qu.:1960   1st Qu.: 1.00
##  Median :1978   Median : 2.00
##  Mean   :1974   Mean   : 4.63
##  3rd Qu.:1992   3rd Qu.: 5.00
##  Max.   :2005   Max.   :25.00
```

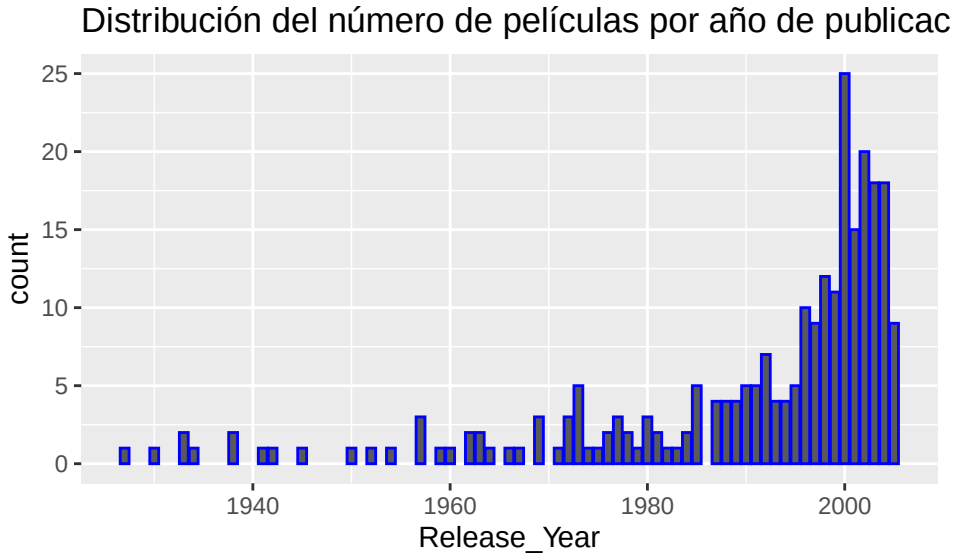
```
max_n_table <- movies_per_year %>%
  filter(n == max(n))
max_n_year <- max_n_table$Release_Year
max_n <- max_n_table$n
```

El año que se estrenaron más películas fue el 2000 y se estrenaron 25 y en un 50% de los años se estrenaron como mucho 2 pelis. Tener en cuenta que esto es sobre una muestra de un 1.4068655% del total de las películas de netflix.

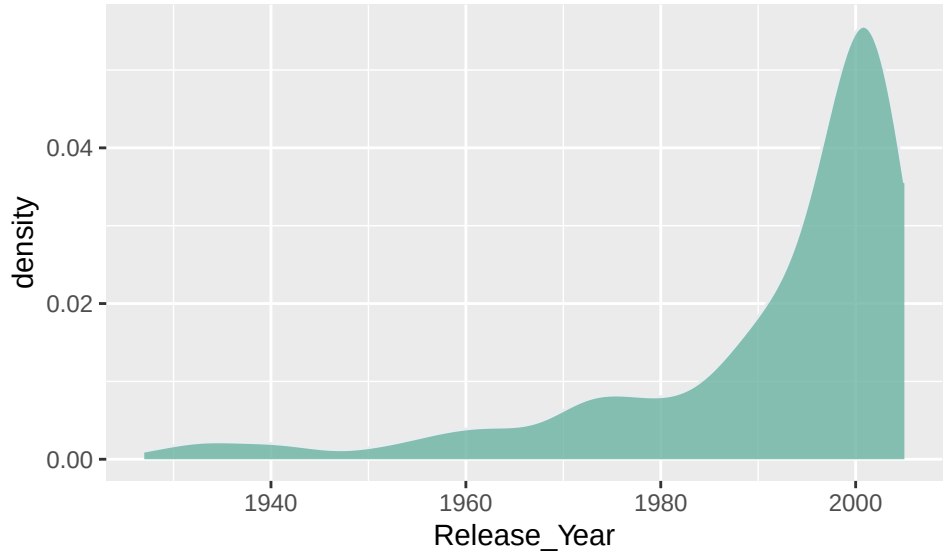
```
ggplot(data = movies_per_year) + #sistema de coordenadas al que añadir puntos(creates an empty graph)
  geom_point(mapping=aes(Release_Year, n), color='blue', fill='red', shape=21) +
  ggtitle('Distribución del número de películas por año de publicación')
```




```
ggplot(data = nuestras_titles) +
  geom_bar(mapping=aes(x=Release_Year), stat='count', color='blue') +
  ggtitle('Distribución del número de películas por año de publicación')
```




```
ggplot(data = nuestras_titles, aes(x=Release_Year)) +
  geom_density(fill="#69b3a2", color="#e9ecef", alpha=0.8)
```




```
release_year_score_avg <- scores %>%
  group_by(Release_Year) %>%
  summarise(Mean_Score = mean(Score), n = n()) %>%
  arrange(desc(Mean_Score))

kable(head(release_year_score_avg))
```

Release_Year	Mean_Score	n
1952	4.101009	29225
1934	4.025687	11095
1971	3.880000	375
1981	3.853545	56502
1945	3.840952	9494
1941	3.771467	2958

3.3 Pregunta 3

Investiga la librería lubridate (o la que consideréis para manipulación de datos) y utilízala para transformar la columna de la fecha de la valoración en varias columnas por ejemplo year, month, week, day_of_week.

Usamos la librería *lubridate* para generar variables separadas para año, número de mes, número de semana del año, número de día del mes, número de día de la semana, y una variable binaria que specifica si el día es fin de semana o entre semana.

```
scores %<>% mutate(
  Year = year(Date),
  n_month = month(Date),
  Week = week(Date),
  Day = day(Date),
  n_day_week = wday(Date,
                    week_start = getOption("lubridate.week.start", 1)
                    ),
  Is_Weekend = if_else( isWeekend(Date) == TRUE, "Weekend", "Weekday" )
  )
```

A partir de las variables de número de mes y número de día de la semana, creamos sendos factores ordenados para el mes y el día de la semana.

```
scores %<>% mutate(
  Month = ordered(n_month, levels = seq(1, 12, 1),labels = month.abb),
  Day_Week = ordered(n_day_week, levels = seq(1, 7, 1), labels = day.abb)
)

scores_dates <- scores %>%
  select(MovieID, UserID, Score, Date, Year, Month, Day, Day_Week, Is_Weekend)

kable(head(scores_dates, 3))
```

MovieID	UserID	Score	Date	Year	Month	Day	Day_Week	Is_Weekend
515	2295232	1	2005-08-16	2005	Aug	16	Tue	Weekday
515	1560318	3	2005-10-04	2005	Oct	4	Tue	Weekday
515	2550394	1	2005-11-01	2005	Nov	1	Tue	Weekday

3.4 Pregunta 4

Genera un tabla que para cada película nos dé el número total de valoraciones, la suma de las valoraciones, la media las valoraciones, y otras estadísticos de interés (desviación típica, moda , mediana).

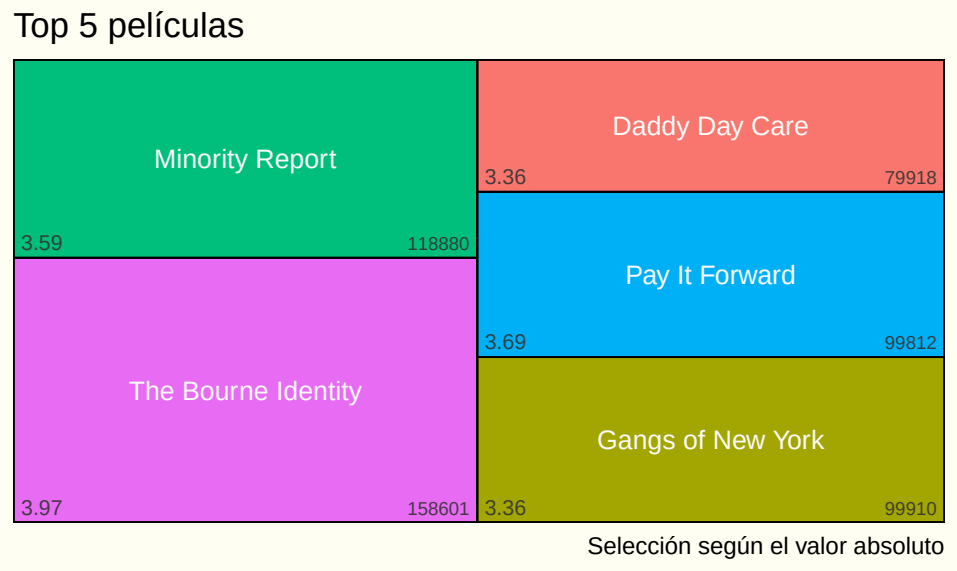
```
movie_scores <- scores %>%
  group_by(MovieID) %>%
  summarise(Sum_Score = sum(Score), Mean_Score = mean(Score), SD_Score = sd(Score), Mode_Score = mlv(Score), Median_Score = medi
  left_join(titles, by = 'MovieID')

movie_scores_table <- movie_scores %>%
  ungroup() %>%
  select(-MovieID, -Release_Year) %>%
  relocate(Title, n, Sum_Score, Mean_Score, SD_Score, Mode_Score, Mean_Score)

kable(head(movie_scores_table %>% arrange(desc(Mean_Score))))
```

Title	n	Sum_Score	Mean_Score	SD_Score	Mode_Score	Median_Score
Curb Your Enthusiasm: Season 3	12148	52674	4.336022	0.9997265	5	5
Prime Suspect 3	2637	11222	4.255593	0.8992768	5	4
Singin' in the Rain	29225	119852	4.101009	0.9473879	5	4
VeggieTales: Dave and the Giant Pickle	2476	10102	4.079968	1.1253132	5	4
The Thin Man	11095	44665	4.025687	0.9605373	5	4
Felicity: Season 3	2820	11320	4.014184	1.2366969	5	4

3.5 Pregunta 5

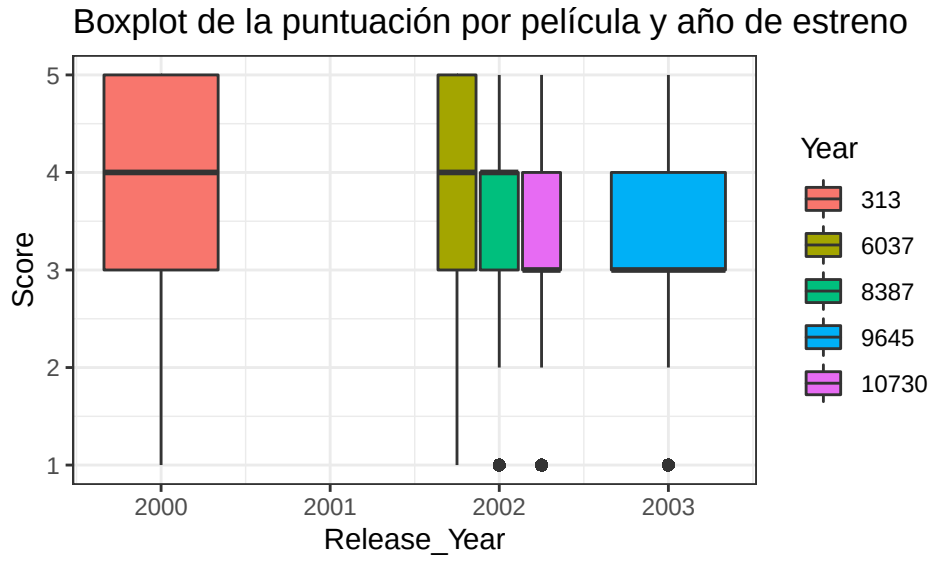
De las cinco películas con más número total de valoraciones, compara sus estadísticos y distribuciones (histogramas, boxplot, violin plot,...)

A continuación, representamos el top 5 películas en un `treemap`:

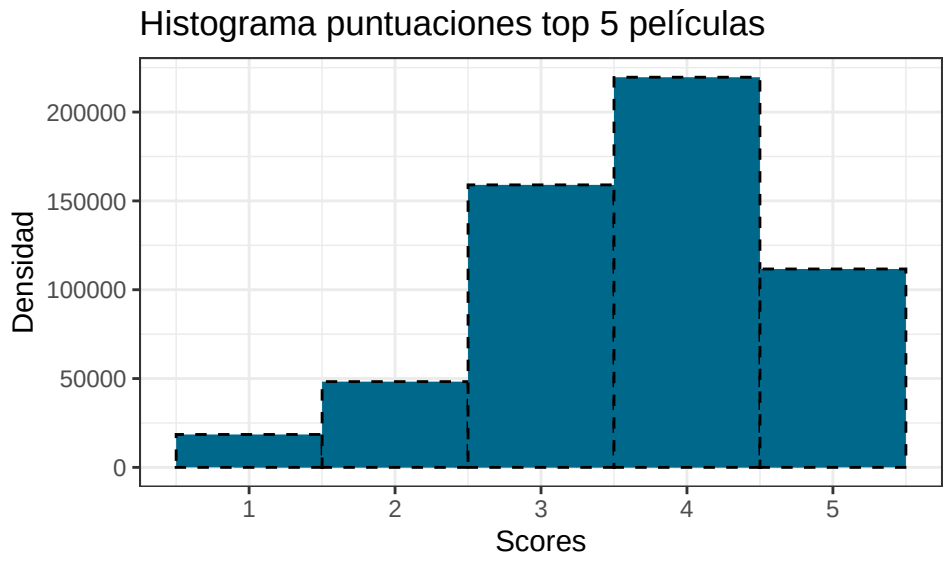


Primero, vamos a comparar los estadísticos de estas cinco películas mediante un *boxplot* y un *histograma*:

```
par(bg="grey98", mar=c(3,3,3,3), mfcol=c(1,2))
top5 = scores %>%
  filter(MovieID %in% c('6037', '8387' , '10730' , '313', '9645'))

pl <- ggplot(top5, aes(x = Score, y = Release_Year, fill = factor(MovieID)))
pl + geom_boxplot() + theme_bw() + coord_flip() + labs(fill = "Year", title = 'Boxplot de la puntuación por película y año de es
  scale_y_continuous(breaks = seq(1999, 2005, 1))
```




```
ggplot(top5, aes(x=top5$Score))+
  geom_histogram(aes(y=..density..), position="identity", alpha=0.5) +
  geom_histogram(bins=5, color="black", fill="deepskyblue4",
                 linetype="dashed") +
  labs(title="Histograma puntuaciones top 5 películas",x="Scores", y = "Densidad")+
  theme_bw()
```

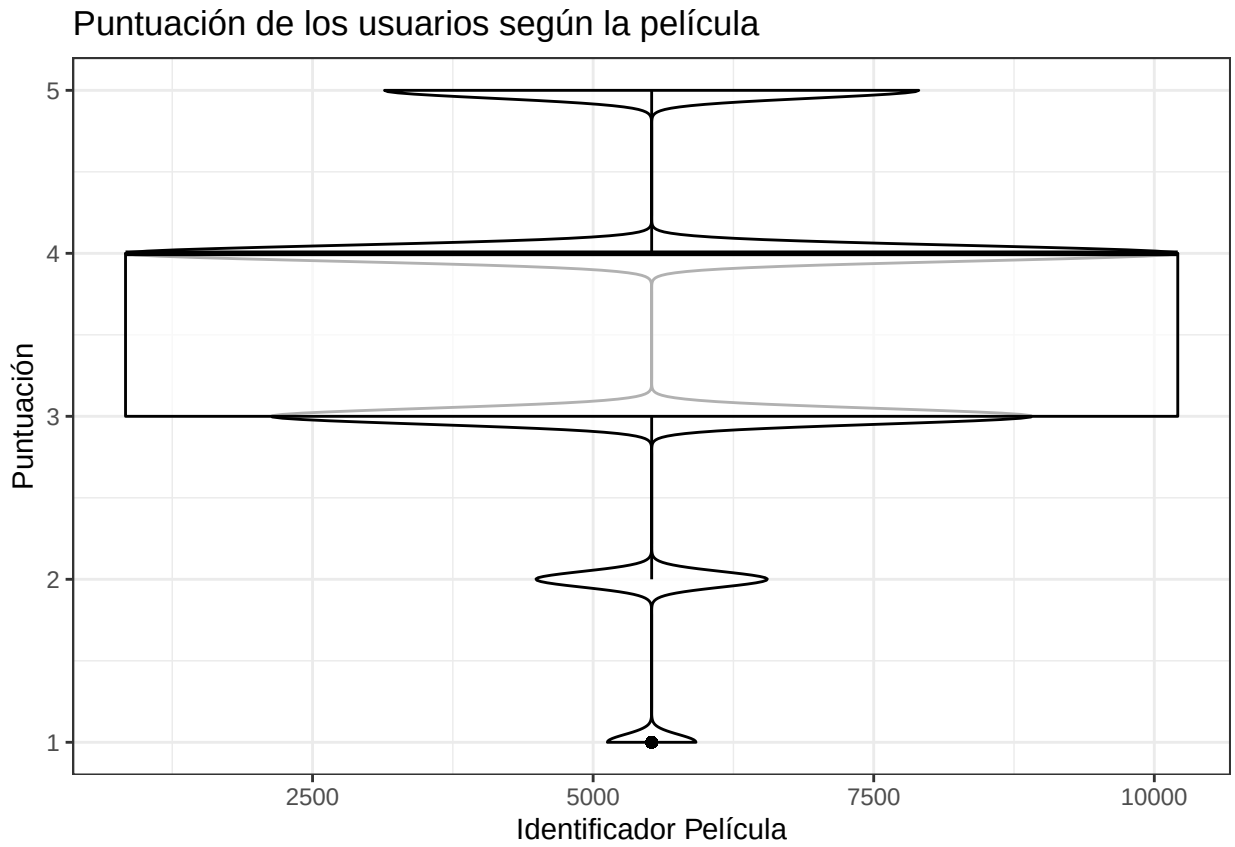


En el *boxplot* se representan las cinco películas por año de estreno. Como muestra el gráfico, las películas se estrenaron entre el 2000 y el 2003 (excluyendo el 2001, donde no se produjo ningún estreno). De estas, la mayoria tienen su mediana en la puntuación 4, seria el caso de las películas con ID 313, 6037 y 8387 respectivamente. Luego, el resto tienen como mediana la puntuación 3.

El histograma nos sirve para reforzar la afirmación anterior, vemos que la frecuencia de puntuaciones 3 y 4 son las más abundantes.

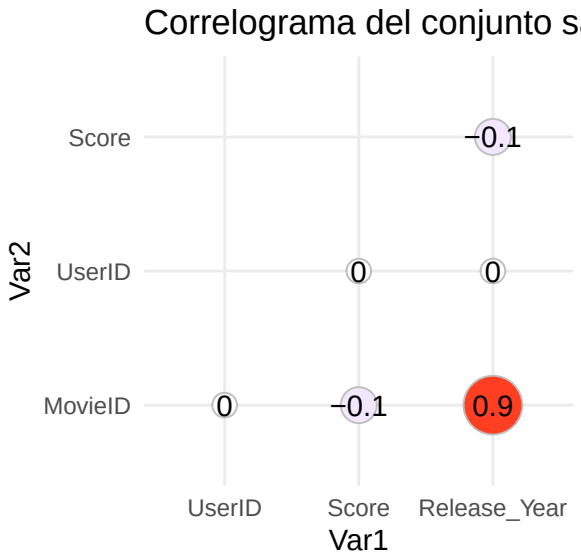
Como alternativa al *boxplot*, se puede utilizar el +violin plot*. En este caso vamos a representar la puntuación otorgada por los usuarios a cada película:

```
plot2
```



Por último, estudiamos si existe relación entre la puntuación media y el año de estreno. Como podemos observar, no existe relación entre

Luego, vamos a observar si existe correlación entre estas variables. Debido al tamaño de la muestra, no se puede establecer si existe o no relación.



3.6 Pregunta 6

Investiga la distribución de valoraciones por día de la semana y por mes.¿Qué meses y días de la semana se valoran más películas en netflix?

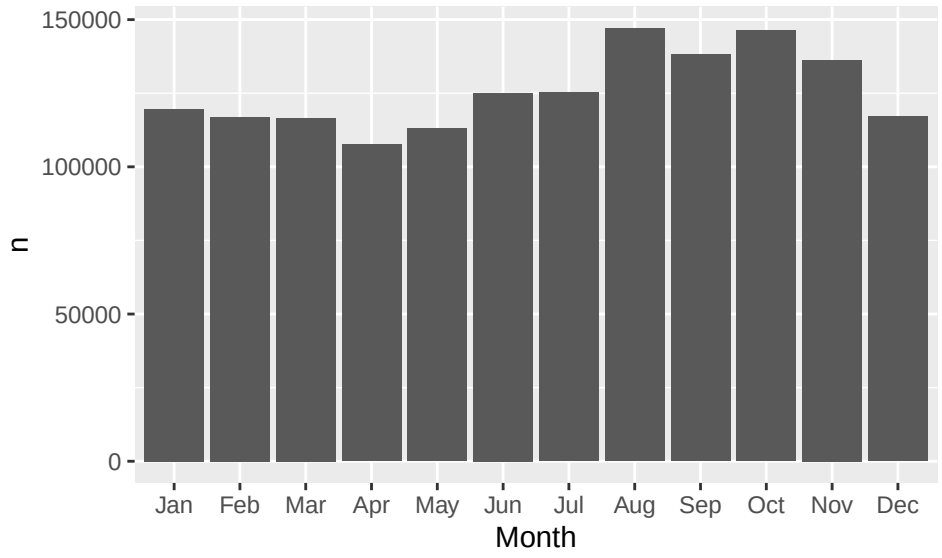
```
month_scores <- scores %>%
  group_by(Month) %>%
  summarise(Mean_Score = mean(Score), n = n())

day_week_scores <- scores %>%
  group_by(Day_Week) %>%
  summarise(Mean_Score = mean(Score), n = n())

kable(month_scores %>% arrange(desc(n)))
```

Month	Mean_Score	n
Aug	3.454642	147065
Oct	3.558666	146252
Sep	3.522692	138154
Nov	3.503592	136295
Jul	3.570329	125319
Jun	3.570904	125042
Jan	3.491908	119567
Dec	3.516152	117014
Feb	3.495188	116994
Mar	3.485262	116570
May	3.541652	113008
Apr	3.516569	107612

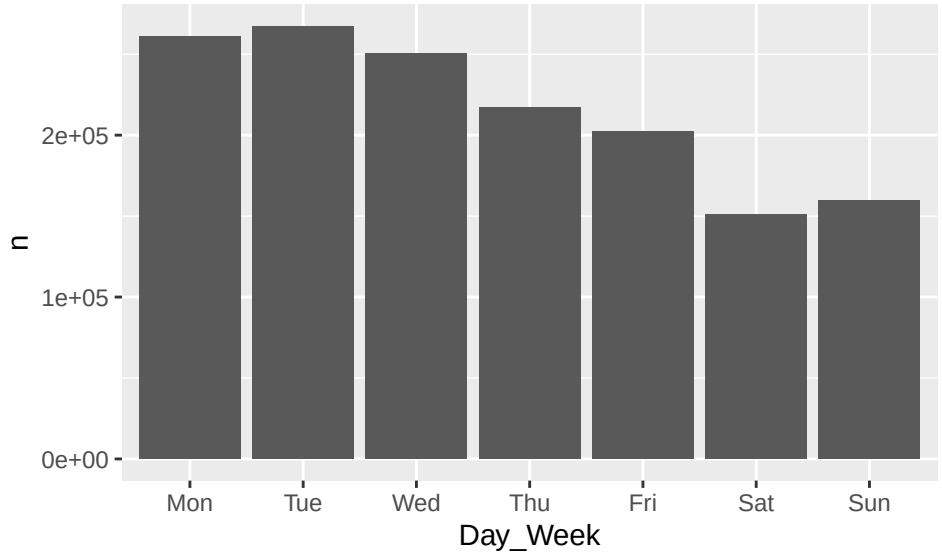
```
kable(day_week_scores %>% arrange(desc(n)))
```

Day_Week	Mean_Score	n
Tue	3.507362	267460
Mon	3.508918	260777
Wed	3.526315	250692
Thu	3.523394	217364
Fri	3.522742	202050
Sun	3.522538	159776
Sat	3.527674	150773

```
ggplot(data = month_scores, aes(x = Month, y = n)) +
  geom_bar(stat = "identity") +
  scale_x_discrete(drop = FALSE, breaks = levels(scores$Month))
```




```
ggplot(data = day_week_scores, aes(x = Day_Week, y = n)) +
  geom_bar(stat = "identity") +
  scale_x_discrete(drop = FALSE, breaks = levels(scores$Day_Week))
```




```
weekend_weekday_scores <- scores %>%
  group_by(Is_Weekend) %>%
  summarise(Mean_Score = mean(Score), n = n())

kable(weekend_weekday_scores)
```

Is_Weekend	Mean_Score	n
Weekday	3.517167	1198343
Weekend	3.525031	310549

```
n_scores_weekend = weekend_weekday_scores  %>% filter(Is_Weekend == TRUE) %>% select(n)
n_scores = sum(weekend_weekday_scores$n)
n_scores_weekend_weekday_ratio = n_scores_weekend / n_scores #el 18% de las valoraciones son en fin de semana, que es menos que
```

3.7 Pregunta 7

Genera una tabla agrupada por película y año del número de valoraciones. Representa la tabla gráficamente para las 10 películas con mayor número de valoraciones.

```
table7.1 <- group_by(scores, MovieID, Year=year(Date)) %>%
  summarise(votes = n_distinct((UserID)))                    #agrupamos por pelicula y año en que fue valorada

table7.2 <- group_by(scores, MovieID) %>%
  summarise(votes = n_distinct((UserID)))                    #agrupamos por pelicula

top10 <- head(arrange(table7.2, desc(votes)), 10)            #las 10 más votadas

movies_onfire <- filter(table7.1, MovieID %in% top10$MovieID) #onfire porque es un heatmap y son las pelis mas votadas
for (i in 1:10) {
  movie <- top10$MovieID[i]
  indexes <- which(movies_onfire$MovieID == movie)
  movies_onfire$MovieID <- replace(movies_onfire$MovieID, indexes, i)
  count=i
}                                                            #Ordenamos las pelis según el top10
kable(top10)
```

MovieID	votes
6037	158601
8387	118880
10730	99910
313	99812
9645	79918
6329	79630
9828	56004
8159	50665
2953	47811
2395	42843

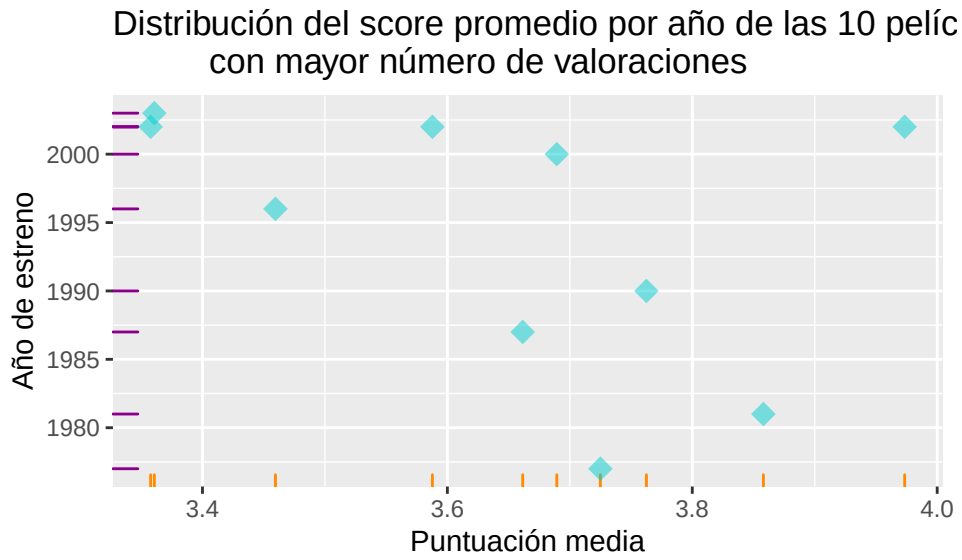
Para visualizar la distribución de votaciones por año que obtubieron las 10 peliculas más votadas de *Netflix*, creamos un Heatmap

```
# 1r creamos una secuencia significativa de intervalos
# barplot(height = movies_onfire$votes, ylim =c(1,80000)) #Cambiando los limites de y que intervalos son significativos

secuencia <- cut(movies_onfire$votes,
                 breaks = c(min(movies_onfire$votes), 1000, 2000, 3000, 5000, 7000, 10000, 15000,
                            20000, 25000, 30000,35000,40000,45000,50000,max(movies_onfire$votes)),
                 labels=c('0<','1k-2k', '2k-3k', '3k-5k', '5k-7k', '7k-10k', '10k-15k', '15k-20k',
                          '20k-25k', '25k-30k', '30k-35k', '35k-40k', '40k-45k', '45k-50k','>85k'),
                 include.lowest = T) #15 values
# 2n creamos una paleta de 15 colores
library(RColorBrewer)
nb.cols <- 15
mycolors <- colorRampPalette(brewer.pal(9, "YlOrRd"))(nb.cols)

# 3r creamos el Heatmap
ggplot(movies_onfire, aes(text = paste('Votes:', votes), y = MovieID, x = Year )) +  #text para visualizar votos en el interacti
  geom_tile(aes(fill = secuencia)) +
  scale_y_continuous(breaks=1:10) +
  scale_x_continuous(breaks=1999:2005) +
  scale_fill_manual(values = mycolors) +               #secuencia de colores
  labs(fill = 'Votes') -> hm                            #Legend name
ggplotly(hm, tooltip = c('text', 'MovieID', 'Year'))
```

3.8 Pregunta 8

Distribución del score promedio por año de las 10 películas con mayor número de valoraciones.

```
ggplot() +
  #geom_point(color = 'red', fill = 'red', size = 4, shape = 18, alpha = 0.5) +
  geom_rug(data=ej_10, mapping=aes(x=Mean_Score), color="darkorange") +
  geom_rug(data=ej_10, mapping=aes(y=Release_Year), color="darkmagenta") +
  geom_point(data=ej_10, mapping=aes(x=Mean_Score, y=Release_Year), color = 'darkturquoise', fill = 'darkmagenta', size = 4, sha
  ggtitle("Distribución del score promedio por año de las 10 películas
          con mayor número de valoraciones") +
  #theme(plot.title = element_text(size = 10, face = "bold", family = 'Century')) +
  xlab('Puntuación media') +
  ylab('Año de estreno')
```




```
#+
 # theme_bw() +
  #theme(text=element_text(family="Broadway", face="bold", size=10))
```

3.9 Pregunta 9

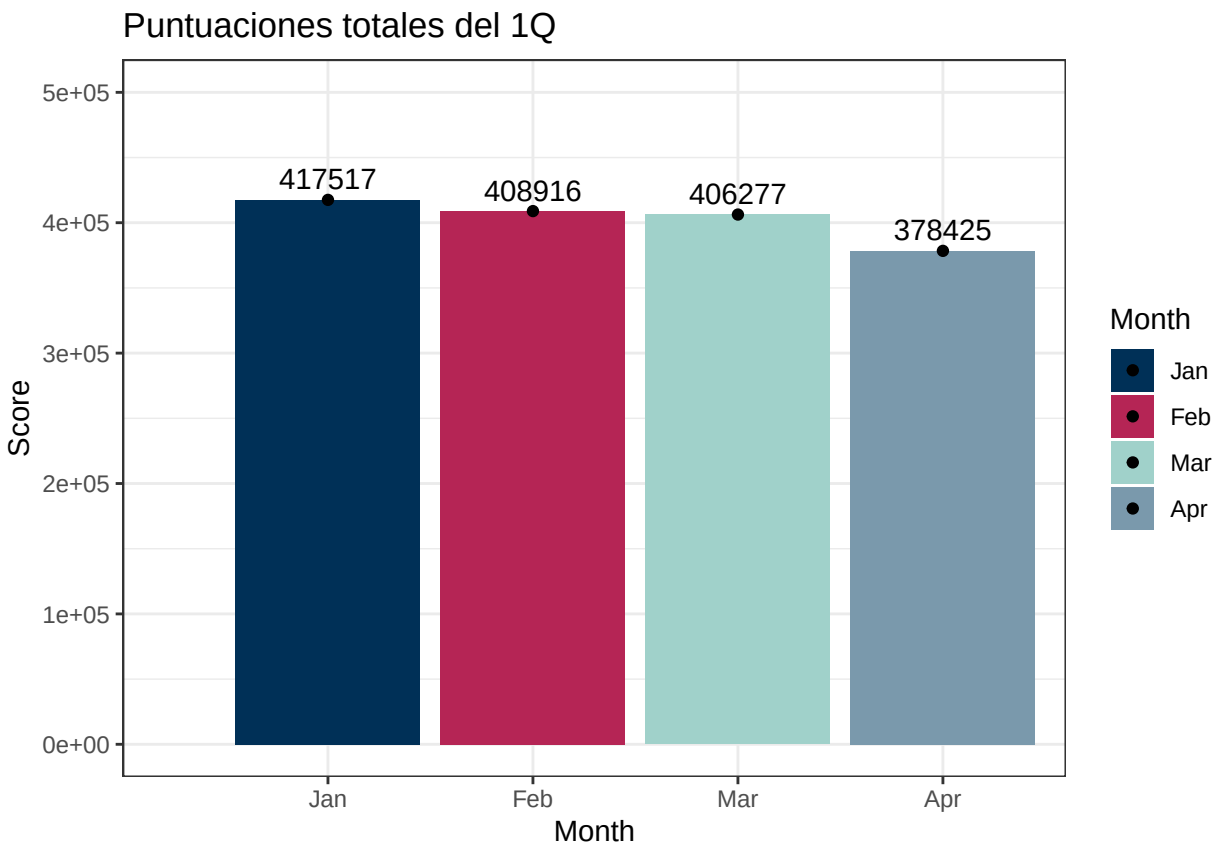
Realiza algún gráfico o estudió de estadísticos adicional que consideres informativo en base al análisis exploratorio anterior.

1. Puntuaciones por _quarter_

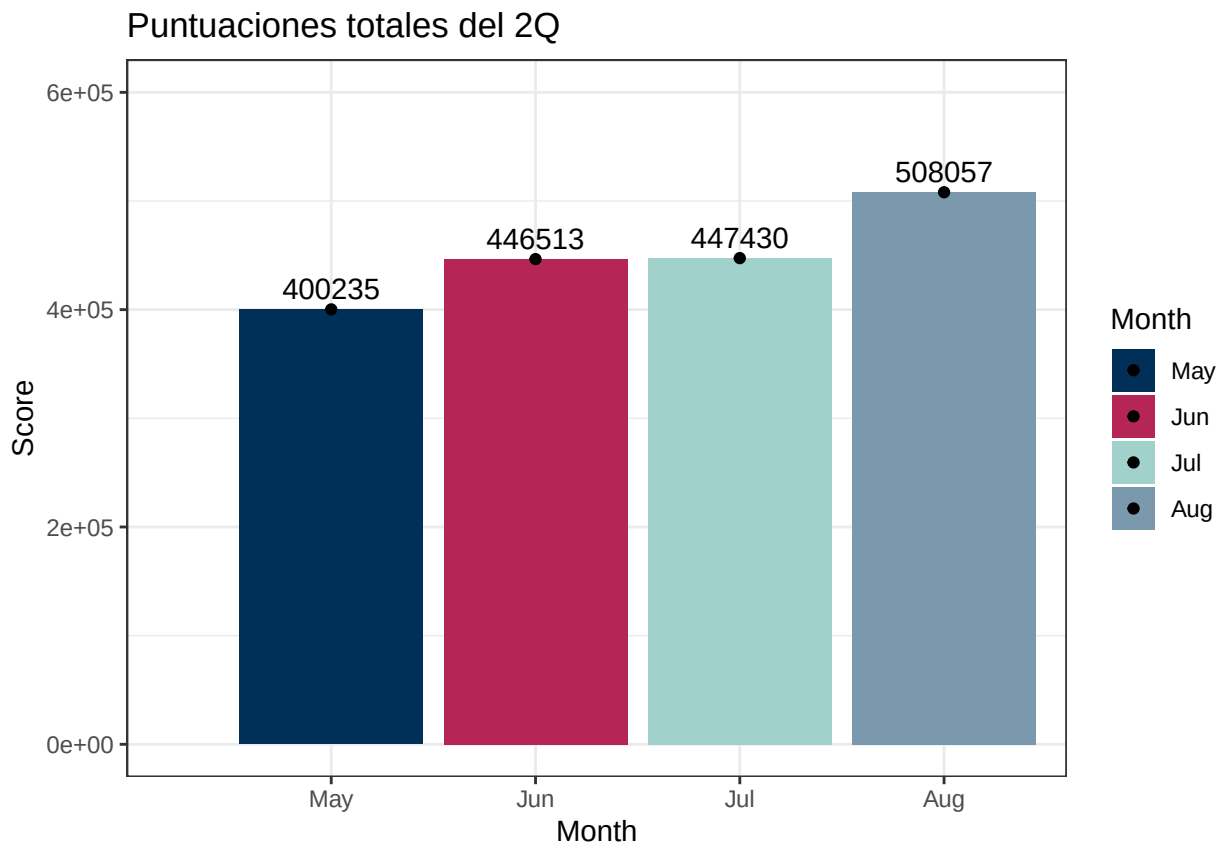
En este apartado vamos a representar los meses que recogen mayores puntuaciones. Para observar mejor los resultados, hemos decidido dividir el estudio en tres escenarios diferentes. En primer lugar, el Q1 que comprende de enero a abril (ambos incluidos), y lo mismo con Q2 de mayo a agosto y, finalmente, Q3 de septiembre a diciembre.

Este ha sido el resultado:

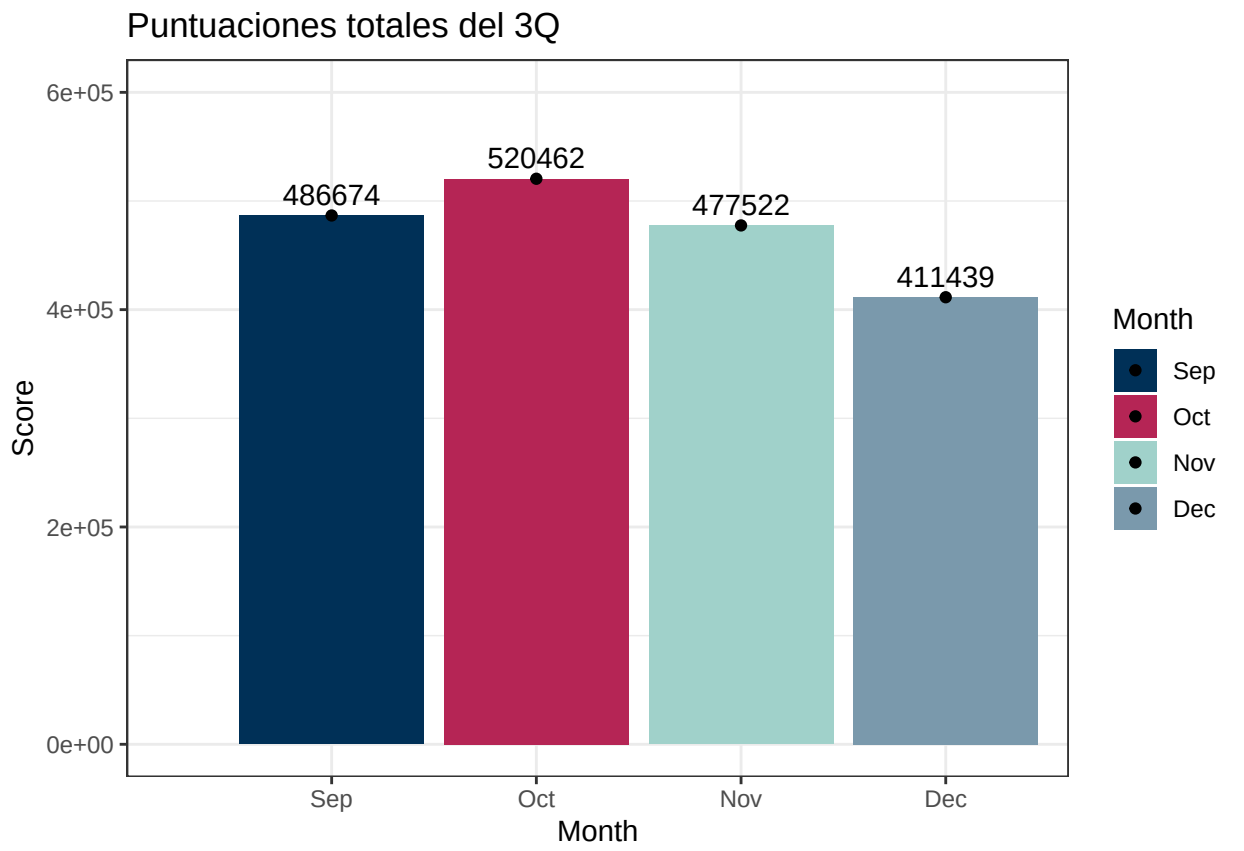
```
par(mfrow = c(3,1))
gg1
```


Puntuaciones totales del 1Q
5e+05
4e+05
3e+05
2e+05
1e+05
0e+00
Score
417517
408916
406277
378425
Jan
Feb
Mar
Apr
Month
Month
Jan
Feb
Mar
Apr

```
gg2
```


Puntuaciones totales del 2Q
6e+05
4e+05
2e+05
0e+00
Score
400235
446513
447430
508057
May
Jun
Jul
Aug
Month
Month
May
Jun
Jul
Aug

```
gg3
```




```
par(mfrow = c(1,1))
```

Observando estos resultados, podemos saber qué mes recoge las puntuaciones más altas. En el primer escenario (1Q) es enero, en el segundo escenario (2Q) es agosto y en el tercer escenario (3Q) es octubre.

Buscar los top 5 usuarios que más películas han puntuado. Luego, comparar con el top 1 usuario qué películas han dejado de evaluar el resto.

Primero vamos a buscar el número de total de películas que han sido evaluadas por usuario:

```
#número de veces que ha votado cada usuario
num_votos_por_usuario = aggregate(scores$UserID, by = list(Usuario=scores$UserID), length)
```

En segundo lugar, seleccionaremos el *top 5 usuarios* que más películas han puntuado,

```
df <- scores %>% group_by(UserID) %>% count()
df <- scores %>% group_by(UserID) %>% summarise(NN = n())
df <- scores %>% group_by(UserID) %>%
  summarise(NN = n(), percent = n()/nrow(.) ) #Añadir a la tabla el % que representa cada país en el Total
df <- scores %>% group_by(UserID) %>%
  summarise (NN = n()) %>%
  mutate(percent= NN / sum(NN))
top_5_users <- head(df[order(df$NN, decreasing = TRUE),],5)
knitr::kable(top_5_users, digits = 5, align = "c", caption = "Top 5 Usuarios")
```

Table 8: Top 5 Usuarios

UserID	NN	percent
305344	249	0.00017
387418	247	0.00016
2439493	232	0.00015
1664010	223	0.00015
2118461	208	0.00014

En tercer lugar, buscaremos qué películas han sido evaluadas por estos usuarios. Seguidamente, compararemos el total de películas evaluadas por el usuario `top_1` con el resto:

```
##top_pelis_1
top_pelis_1 = scores %>%
  filter(UserID %in% c('305344'))

##top_pelis_2
top_pelis_2 = scores %>%
  filter(UserID %in% c('387418'))

##top_pelis_3
top_pelis_3 = scores %>%
  filter(UserID %in% c('2439493'))

##top_pelis_4
top_pelis_4 = scores %>%
  filter(UserID %in% c('1664010'))

##top_pelis_5
top_pelis_5 = scores %>%
  filter(UserID %in% c('2118461'))
```

El usuario que más películas ha puntuado es el *305344*, entonces vamos a comparar el resto de usuarios con este:

```
top_pelis_1$comp2 <- as.integer(top_pelis_1$MovieID %in% top_pelis_2$MovieID)
top_pelis_1$comp3 <- as.integer(top_pelis_1$MovieID %in% top_pelis_3$MovieID)
top_pelis_1$comp4 <- as.integer(top_pelis_1$MovieID %in% top_pelis_4$MovieID)
top_pelis_1$comp5 <- as.integer(top_pelis_1$MovieID %in% top_pelis_5$MovieID)
Dif_1_2= top_pelis_1 %>%
           filter(comp2 %in% c('0'))
Dif_1_3= top_pelis_1 %>%
  filter(comp3 %in% c('0')) # obtener las que no aparecen
Dif_1_4= top_pelis_1 %>%
  filter(comp4 %in% c('0')) # obtener las que no aparecen
Dif_1_5= top_pelis_1 %>%
  filter(comp5 %in% c('0')) # obtener las que no aparecen
```

Películas que el `top_2` no ha evaluado pero el `top_1` si lo ha hecho:

```
films_2 = Dif_1_2$Title
```

Películas que el `top_3` no ha evaluado pero el `top_1` si lo ha hecho:

```
films_3 = Dif_1_3$Title
```

Películas que el `top_4` no ha evaluado pero el `top_1` si lo ha hecho:

```
films_4 = Dif_1_4$Title
```

Películas que el `top_5` no ha evaluado pero el `top_1` si lo ha hecho:

```
films_5 = Dif_1_5$Title
```

```
2. Puntuaciones por película
3. Puntuaciones por usuario
4. Número de puntuaciones por película, usuario y año lanzamiento
5. Distribucion de los scores (boxplot,barplot)
6. Series temporales de puntuaciones
7. Distribución de cuantos usuarios evaluan cuantas pelis totales y diferentes
```

4 Sistema de Recomendación / Similaridad (opcional)