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

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 títulos películas

Carga archivo titulos películas

```
#rm(titles)
#algunas peliculas tienen una coma en su nombre, así que cargamos primero todo como una única columna, 
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:

```r
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:

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

2.2.5 Importación datos limpios para analizar en la sección Estadística Descriptiva

```r
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

3.1 Resumen

- 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.

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:

3.2 Tipo de variables

```r
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, 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.

3.3 Distribución películas estrenadas por año

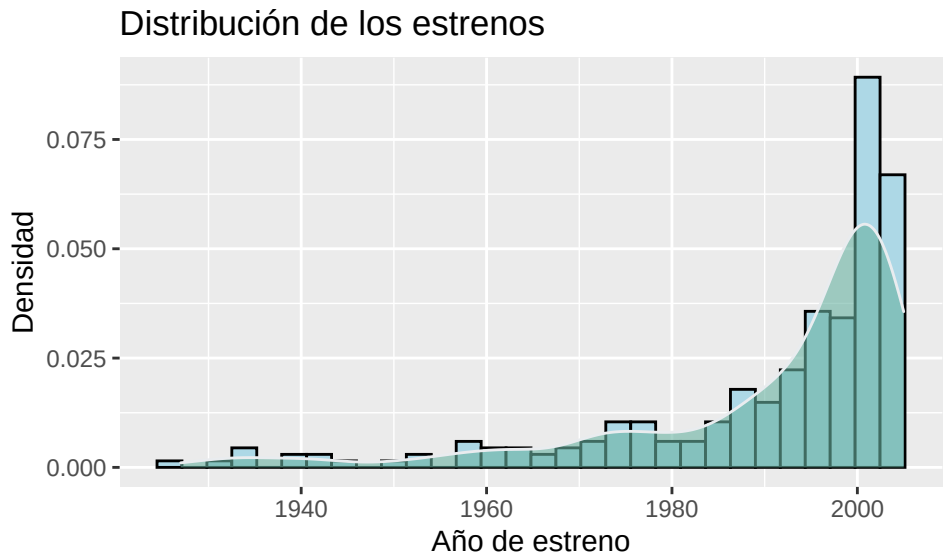
```
nuestros_movie_ids <- tibble(MovieID = unique(scores$MovieID))
nuestros_titles <- titles %>%
  right_join(nuestros_movie_ids, by = "MovieID")

movies_per_year <- nuestros_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 1.41% del total de las películas de netflix.

```
ggplot(data = nuestros_titles,aes(x=Release_Year, y=..density..)) +
  geom_histogram(colour="black", fill="lightblue") +
  geom_density(fill="#69b3a2", color="#e9ecef", alpha=0.6) +
  labs(x='Año de estreno', y='Densidad', title='Distribución de los estrenos')
```



3.4 Transformación variable fecha valoración

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

kable(head(scores_dates_table, 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

Table 1: Top 5 Usuarios

Title	n	Sum_Score	Mean_Score	SD_Score	Mode_Score	Median_Score
Curb Your Enthusiasm: Season 3	12148	52674	4.336	1.000	5	5
Prime Suspect 3	2637	11222	4.256	0.899	5	4
Singin' in the Rain	29225	119852	4.101	0.947	5	4
VeggieTales: Dave and the Giant Pickle	2476	10102	4.080	1.125	5	4
The Thin Man	11095	44665	4.026	0.961	5	4
Felicity: Season 3	2820	11320	4.014	1.237	5	4

3.5 Estadísticos dataframe puntuaciones

```
movie_scores <- scores %>%
  group_by(MovieID) %>%
  summarise(Sum_Score = sum(Score), Mean_Score = mean(Score), SD_Score = sd(Score), Mode_Score = mlv(Sc
  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))), digits = 3, align = "c", caption = "Top 5
kable_styling(latex_options = c("striped"),
font_size = 8) %>%
column_spec(1,width = "4cm")
```

3.6 Comparación top 5 películas con más valoraciones

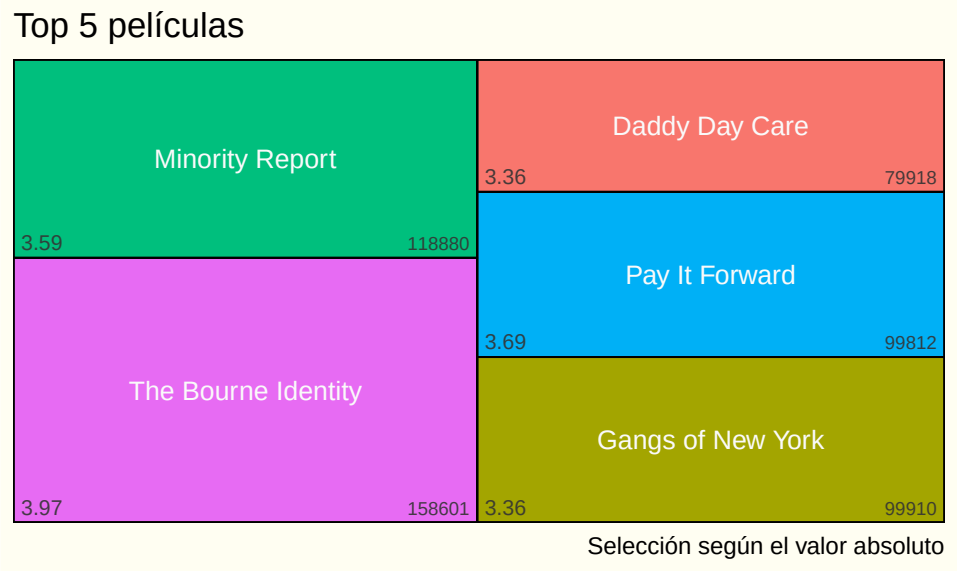
A continuación, representamos el top 5 películas en un `treemap`, este incluye las siguientes variables:

1) El título de las cinco películas que recogen más valoraciones.

2) El tamaño de cada recuadro es proporcional al total de las valoraciones obtenidas.

2.1) Las valoraciones obtenidas se reflejan, además, en la esquina inferior derecha.

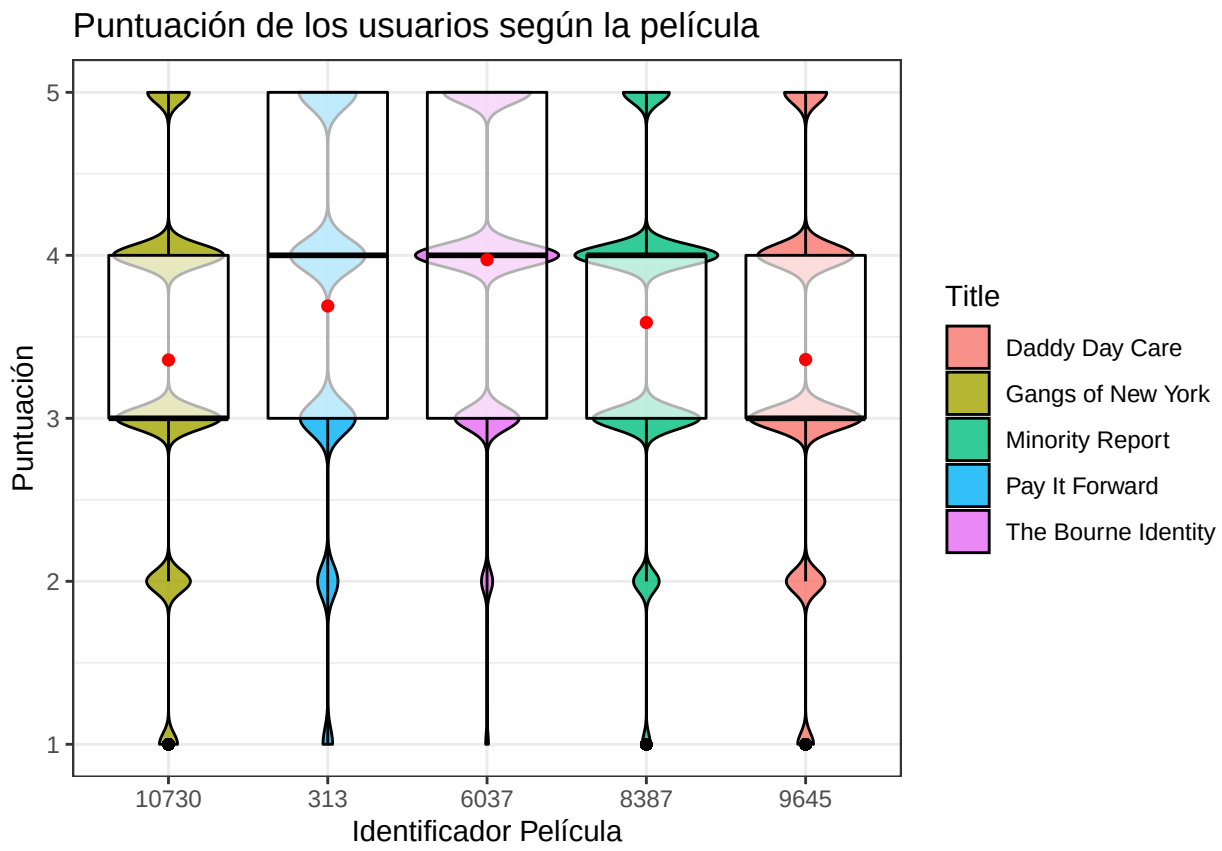
3) Finalmente, la esquina inferior izquierda es la puntuación media de cada película.



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

Utilizaremos el *diagrama de violin* para visualizar la distribución de los datos y su densidad de probabilidad. En este caso vamos a representar la puntuación otorgada por los usuarios a cada película. A continuación, veremos que este gráfico es una combinación de un diagrama de cajas y bigotes y un diagrama de densidad. El punto rojo es la media de cada película, mientras que la línea negra que atraviesa el diagrama de cajas es la moda.

```
plot2
```

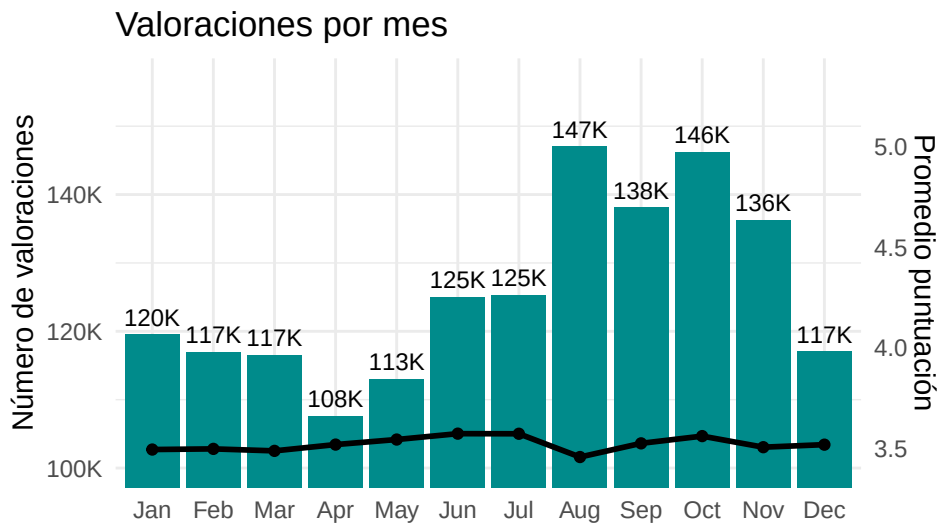


3.7 Análisis del número de valoraciones por més y día de la semana

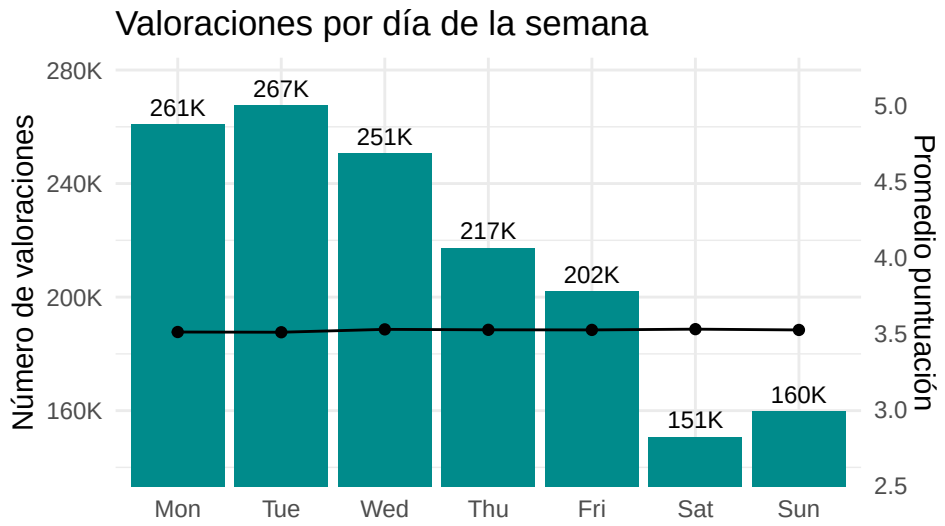
```
ks <- function (x) { number_format(accuracy = 1,
                                   scale = 1/1000,
                                   suffix = "K",
                                   big.mark = ",")(x) }
```

```
coeff <- max(scores$Score) / max(month_scores$n)
n_max_limit <- max(month_scores$n) + 10^4
n_min_limit <- plyr::round_any((n_max_limit - min(month_scores$n))*2, 10^4)

gg1 <- ggplot(data = month_scores, aes(x = Month))


gg2 <- gg1 +
  geom_bar(aes(y = n), fill = "darkcyan", stat = "identity") +
  coord_cartesian(ylim = c(n_min_limit, n_max_limit)) +
  geom_point(aes(y = Mean_Score/coeff)) +
  geom_line(aes(y = Mean_Score/coeff), size = 1, group = 1) +
  scale_y_continuous(
    name = "Número de valoraciones",
    labels = ks,
    sec.axis = sec_axis(~.*coeff, name = "Promedio puntuación")
```

```
  ) +
  labs(title = "Valoraciones por mes", x = "") +
  geom_text(aes(y = n, label = ks(n)), angle = 0, vjust = -0.5, size = 3) +
  theme_minimal()

gg2
```




```
coeff <- max(scores$Score) / max(day_week_scores$n)
n_max_limit <- max(day_week_scores$n) + 10^4
n_min_limit <- plyr::round_any(min(day_week_scores$n) - 10^4, 10^4, f = floor)

gg1 <- ggplot(data = day_week_scores, aes(x = Day_Week))


gg2 <- gg1 +
  geom_bar(aes(y = n), fill = "darkcyan", stat = "identity") +
  coord_cartesian(ylim = c(n_min_limit, n_max_limit)) +
  geom_point(aes(y = Mean_Score/coeff)) +
  geom_line(aes(y = Mean_Score/coeff), group = 1) +
  scale_y_continuous(
    name = "Número de valoraciones",
    labels = ks,
    sec.axis = sec_axis(~.*coeff, name = "Promedio puntuación")
  ) +
  labs(title = "Valoraciones por día de la semana", x = "") +
  geom_text(aes(y = n, label = ks(n)), angle = 0, vjust = -0.5, size = 3) +
  theme_minimal()


gg2
```




```
weekend_weekday_scores <- scores_dates %>%
  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
```

3.8 Análisis top 10 películas con más valoraciones por año de valoración

```
table7.1 <- group_by(scores, MovieID, Year=year(Date)) %>%
  summarise(Votes = n_distinct((UserID)), Mean = round(mean(Score),3))  #agrupamos por pelicula y año e

top10_votada <- head(arrange(movie_scores[,c('MovieID','n','Mean_Score','Title')], desc(n)), 10)  #las

movies_onfire <- filter(table7.1, MovieID %in% top10_votada$MovieID) #onfire porque es un heatmap y son

movies_onfire$Ranking <- movies_onfire$MovieID
for (i in 1:10) {
  movie <- top10_votada$MovieID[i]
  indexes <- which(movies_onfire$MovieID == movie)
  movies_onfire$Ranking <- replace(movies_onfire$Ranking, indexes, i)
  count=i
}                                  #Ordenamos las pelis según el top10
top10_votada$Ranking=1:10
kable(top10_votada[,c('Ranking','Title', 'n')], align = "c", caption = "Top 5 Usuarios") %>%
```

Table 2: Top 5 Usuarios

Ranking	Title	n
1	The Bourne Identity	158601
2	Minority Report	118880
3	Gangs of New York	99910
4	Pay It Forward	99812
5	Daddy Day Care	79918
6	Edward Scissorhands	79630
7	Stripes	56004
8	Annie Hall	50665
9	Moonstruck	47811
10	Scream	42843

```
kable_styling(latex_options = c("striped"),
font_size = 8) %>%
column_spec(1,width = "4cm")
```

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

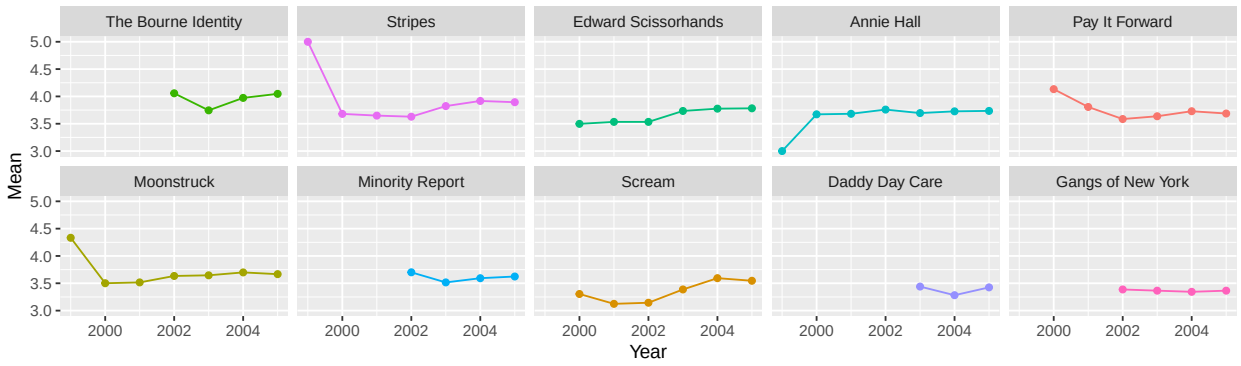
```
# 1r creamos una secuencia significativa de intervalos
# barplot(height = movies_onfire$votes, ylim =c(1,80000)) #Cambiando los limites de y que intervalos so

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
# movies_onfire <- arrange(movies_onfire, desc(Ranking))

ggplot(movies_onfire, aes(text = paste('Votes:', Votes), ID = MovieID, y = Ranking, x = Year )) +  #tex
  geom_tile(aes(fill = secuencia)) +
  scale_y_continuous(breaks=10:1) +
  scale_x_continuous(breaks=1999:2005) +
  scale_fill_manual(values = mycolors) +              #secuencia de colores
  labs(fill = 'Votes') -> hm                           #Legend name
```

También puedes visualizar la versión interactiva de este gráfico.

3.9 Evolución del score promedio de las 10 películas con más valoraciones

```
movies_onfire %<>% left_join(titles[,-2], by = 'MovieID')
orden_titulos <- arrange(top10_votada[,c('Title', 'Mean_Score')], desc(Mean_Score))
```

```
movies_onfire %<>% transform(Title=factor(Title, levels=as.vector(orden_titulos$Title)))

ggplot(movies_onfire, aes(Year, Mean, group=MovieID, colour=factor(MovieID))) +
  geom_point() +
  geom_line() +
  facet_wrap(~Title, nrow = 2, scale='fixed')+
  theme(legend.position="none")
```



Las peliculas están ordenadas por orden descendiente de su puntuación media.

3.10 Estudios adicionales

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 usuario en el
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") %>%
kable_styling(latex_options = c("striped"),
font_size = 8) %>%
column_spec(1,width = "4cm")
```

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:

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

Table 3: 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

```
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:

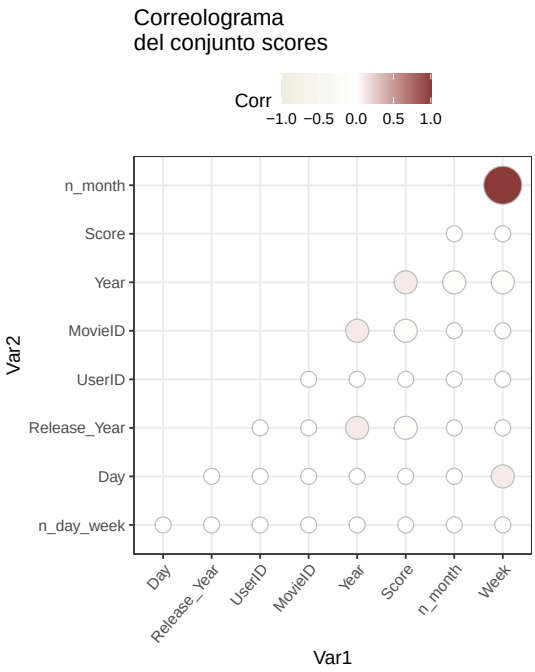
```
films_4 = Dif_1_4$Title
```

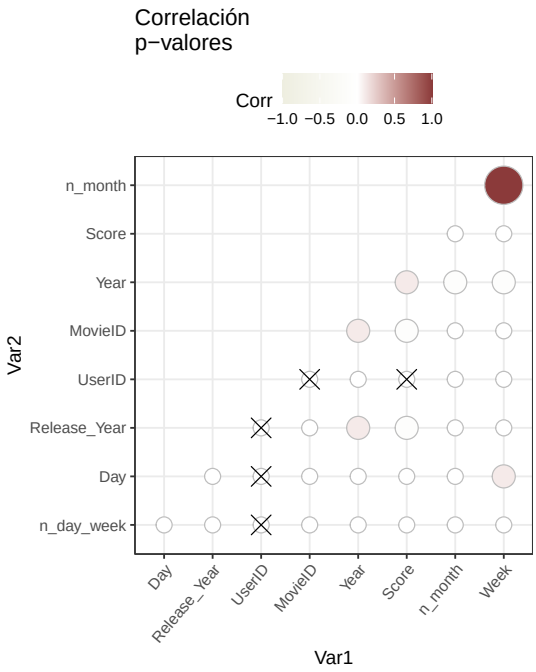
Películas que el `top_5` no ha evaluado pero el `top_1` si lo ha hecho:

```
films_5 = Dif_1_5$Title
```

Para terminar, realizamos un correolograma con el dataset. Hemos creado un correolograma de la tabla `scores_dates`, solamente con los valores numéricos. Luego, se ha incluido también un correolograma con los p-valores de las dimensiones anteriores, para saber si estas son o no son significantes.

Las variables que presentan correlación son las sombreadas en color. Después, la tabla de al lado señala aquellas variables que no son significantes en nuestro análisis.


Correolograma
del conjunto scores
Corr
−1.0 −0.5 0.0 0.5 1.0
n_month
Score
Year
MovieID
UserID
Release_Year
Day
n_day_week
Var2
Day
Release_Year
UserID
MovieID
Year
Score
n_month
Week
Var1


Correlación
p−valores
Corr
−1.0 −0.5 0.0 0.5 1.0
n_month
Score
Year
MovieID
UserID
Release_Year
Day
n_day_week
Var2
Day
Release_Year
UserID
MovieID
Year
Score
n_month
Week
Var1