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 columna, 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

```
scores = aux %>% mutate(fila=row_number())
filas=grep(":",scores$X1)
filas_ID= scores %>%
  filter( fila %in% filas ) %>%
  mutate(ID=as.integer(gsub(":","",X1)))
reps=diff(c(filas_ID$fila,max(scores$fila)+1))

scores %<>%
  mutate(ID1=rep(filas_ID$X1,times=reps)) %>%
  filter(!(fila %in% filas)) %>%
  select(-fila) %>%
  separate(X1,into=c("UserID","Score","Date"),sep=",") %>%
  mutate(Score=as.integer(Score)) %>%
  separate(col = ID1,into=c("MovieID","borrar")) %>%
  select(-borrar) %>% mutate(MovieID=as.numeric(MovieID))
```

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))
```

```
## Warning: Problem with `mutate()` input `..1`.
## i NAs introduced by coercion
## i Input `..1` is `across(c(MovieID:Release_Year), as.integer)`.
```

```
## Warning in fn(col, ...): NAs introduced by coercion
```

```
## Warning: Problem with `mutate()` input `..1`.
## i NAs introduced by coercion
## i Input `..1` is `across(c(MovieID:Release_Year), as.integer)`.
```

```
## Warning in fn(col, ...): NAs introduced by coercion
```

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))
glimpse(scores) #para ver el tipo de dato
```

```
## 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...
```

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
```

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

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

3.1 Pregunta 1

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. . . .

```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. - *Release_Year*: No existen años con decimales, por lo tanto utilizar variables para datos enteros seria suficiente. Movie_title: chr. Utilizamos el tipo carácter porque nos interesan objetos que representan un conjunto de letras. - *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

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

Valoración media por 'Release_Year', de mayor a menor:

```r
id_year <- group_by(scores, MovieID) %>%
  summarise(Release_Year = unique(Release_Year))
```

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

```r
movies_per_year <- id_year %>% group_by(Release_Year) %>%
  summarise(Count_Movies = n_distinct(MovieID))
```

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

```r
which(movies_per_year$Release_Year==2000)
```

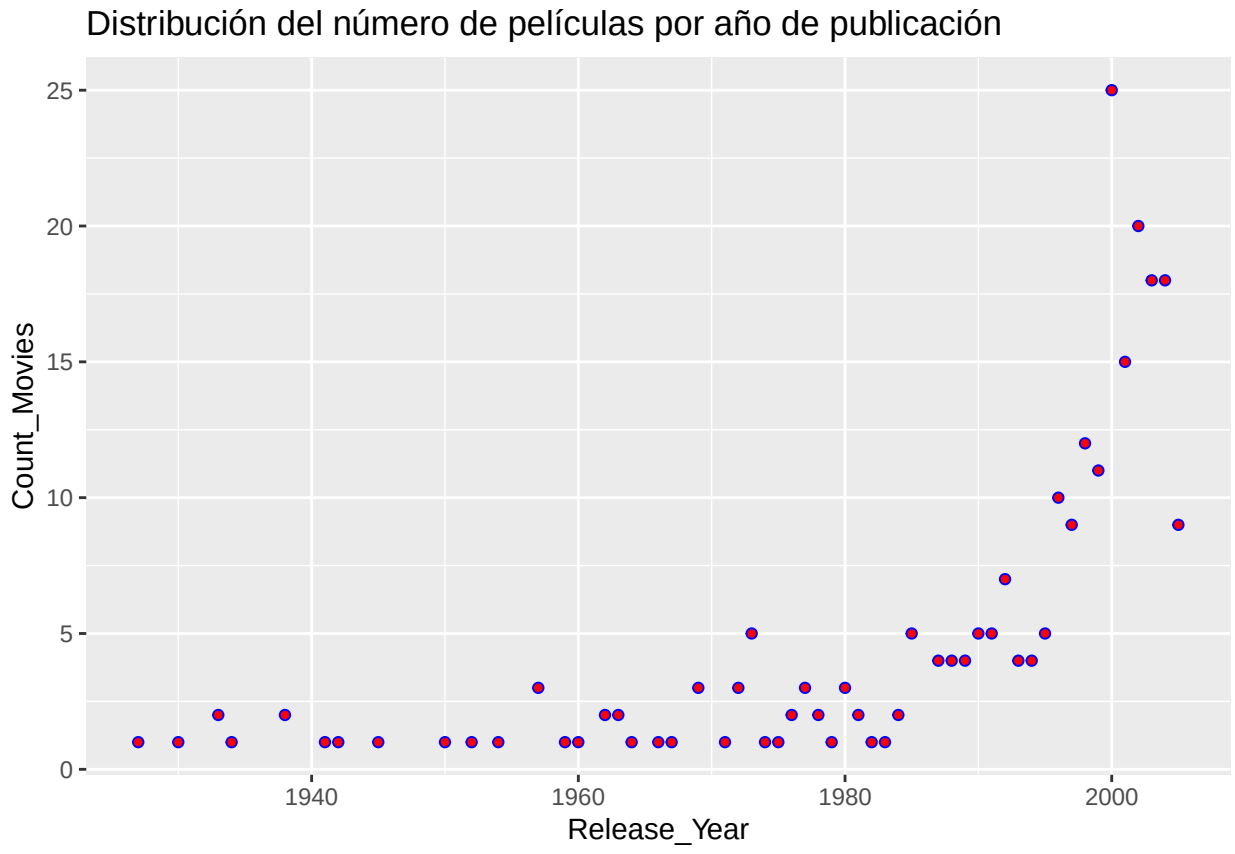
```
## [1] 49
```

```r
summary(movies_per_year)
```

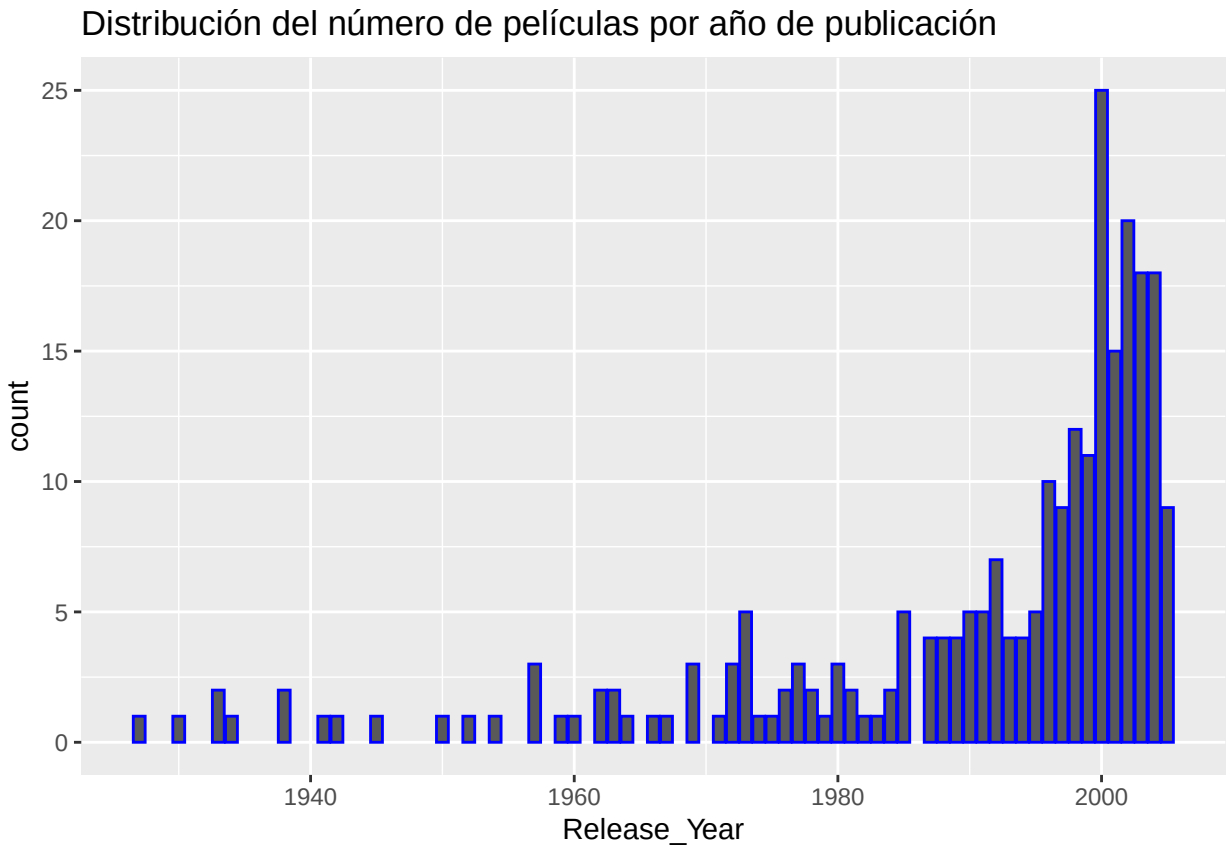
```
##   Release_Year   Count_Movies
##  Min.   :1927   Min.   : 1.00
##  1st Qu.:1960   1st Qu.: 1.00
##  Median :1978   Median : 2.00
##  Mean   :1974   Mean   : 4.63
##  3rd Qu.:1992   3rd Qu.: 5.00
##  Max.   :2005   Max.   :25.00
```

El año que se estrenaron más películas se estrenaron 25 y en un 50% de los años se estrenaron como mucho 2 pelis.(tener en cuanta que esto es siempre sobre nuestra 250)

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




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




```
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

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.

Valoración media por día de la semana, de mayor a menor:

```
scores %<>% mutate(
  Year = year(Date),
  Month = month(Date),
  Week = week(Date),
  Day = day(Date),
  Day_Week = wday(Date,
                  label = TRUE,
                  abbr = TRUE,
                  week_start = getOption("lubridate.week.start", 7)
                  ),
  Is_Weekend = isWeekend(Date)
  )
```

Valoración media entre semana / fin de semana:

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

kable(weekend_weekday_scores)
```

Is_Weekend	Mean_Score	n
FALSE	3.517167	1198343
TRUE	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
```

3.4 Pregunta 4

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).

Valoración media por película, de mayor a menor:

```
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')

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

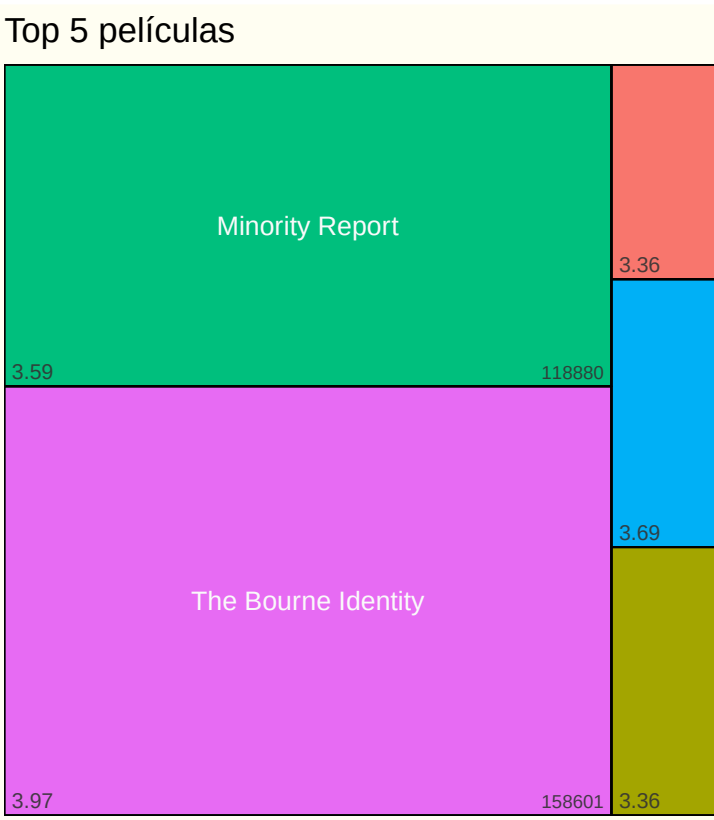
MovieID	Sum_Score	Mean_Score	SD_Score	Mode_Score	Median_Score	n	Release_Year	Title
4353	52674	4.336022	0.9997265	5	5	12148	2002	Curb Your Enthusiasm: Season 3
7393	11222	4.255593	0.8992768	5	4	2637	1993	Prime Suspect 3
2360	119852	4.101009	0.9473879	5	4	29225	1952	Singin' in the Rain
2144	10102	4.079968	1.1253132	5	4	2476	2004	VeggieTales: Dave and the Giant Pickle
8382	44665	4.025687	0.9605373	5	4	11095	1934	The Thin Man
8940	11320	4.014184	1.2366969	5	4	2820	2000	Felicity: Season 3

```
kable(head(movie_scores %>% arrange(desc(n))))
```

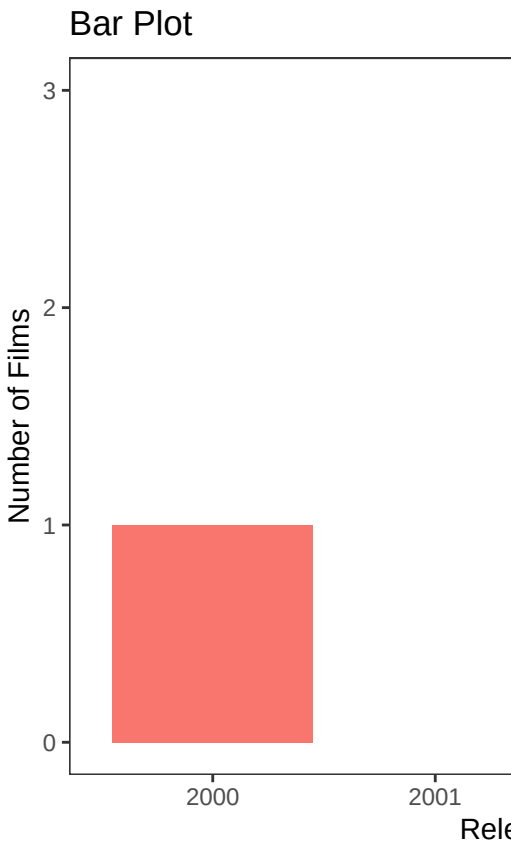
MovieID	Sum_Score	Mean_Score	SD_Score	Mode_Score	Median_Score	n	Release_Year	Title
6037	630194	3.973455	0.8450219	4	4	158601	2002	The Bourne Identity
8387	426515	3.587778	0.9070202	4	4	118880	2002	Minority Report
10730	335467	3.357692	1.0526205	4	3	99910	2002	Gangs of New York
313	368241	3.689346	1.0925298	4	4	99812	2000	Pay It Forward
9645	268580	3.360695	1.0365314	3	3	79918	2003	Daddy Day Care
6329	299611	3.762539	0.9627352	4	4	79630	1990	Edward Scissorhands

3.5 Pregunta 5

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



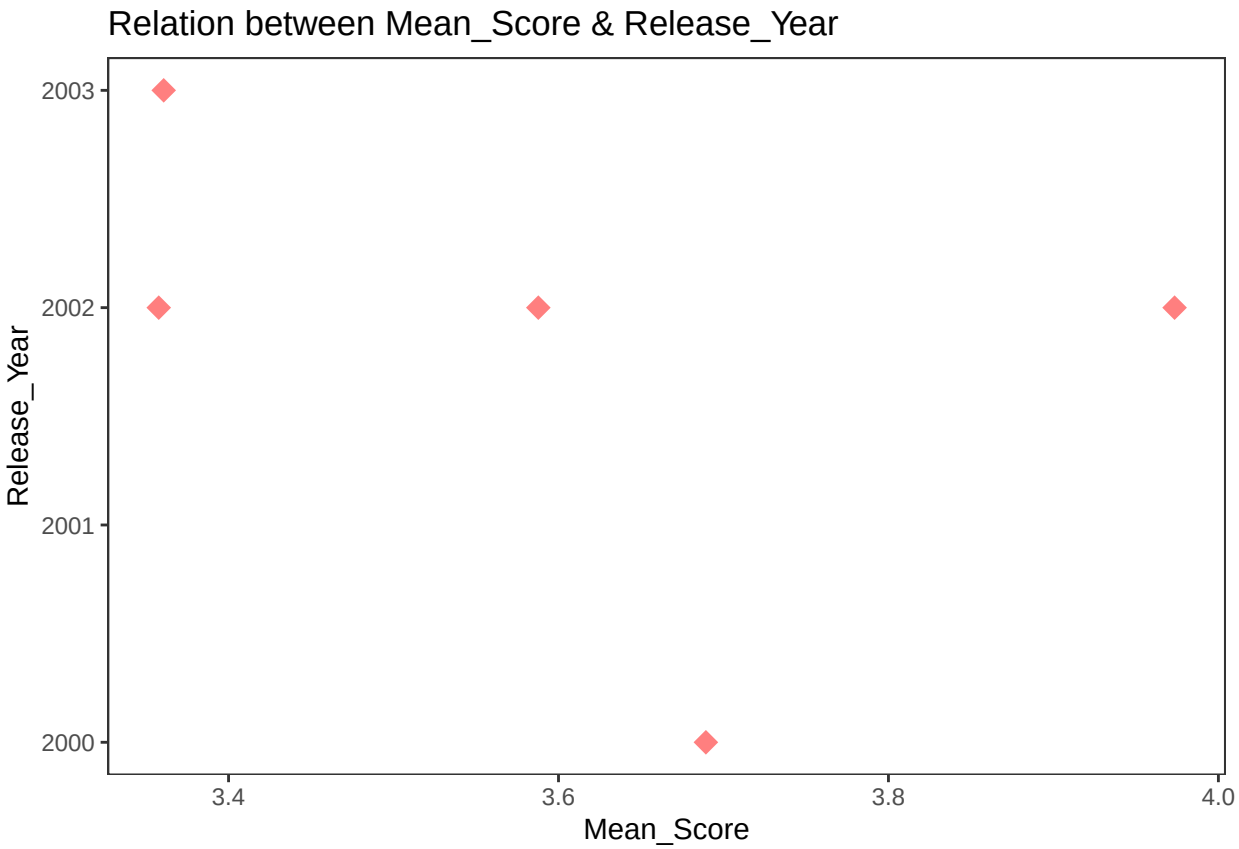
A continuación, representamos el top 5 películas en un `treemap`:



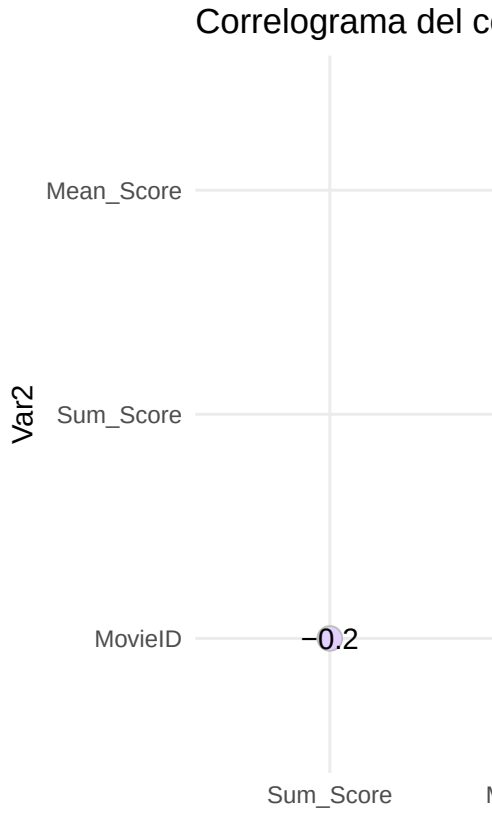
Mediante un `barplot` podemos ver el año de estreno de las películas seleccionadas:

Para terminar con los gráficos de dispersión, estudiamos si existe relación entre, la puntuación media y el año de estreno:

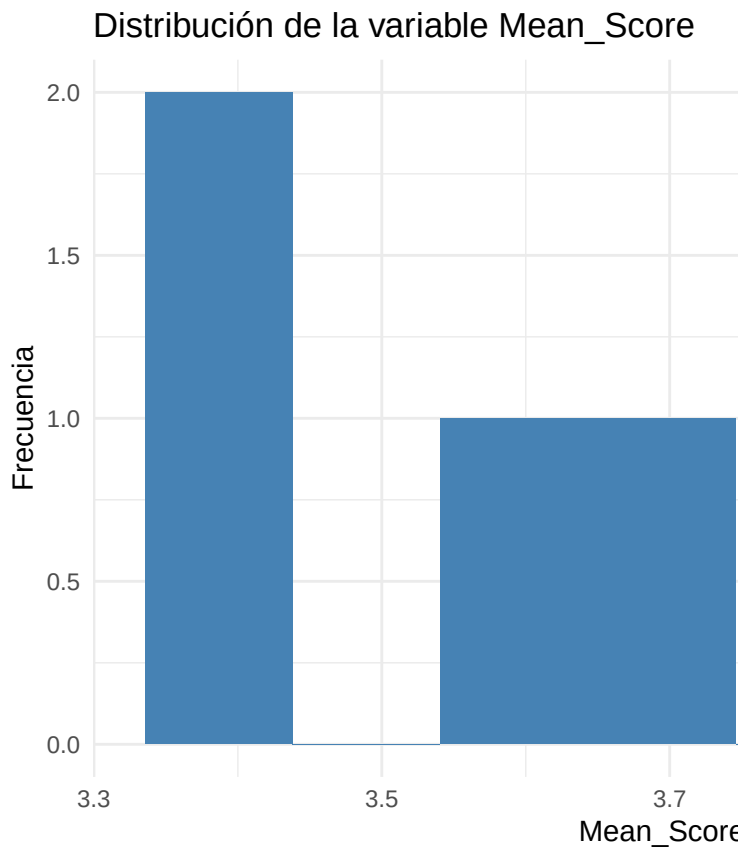
```
##Gráfico de Dispersión (Scatterplot)
ggplot(data = sample, aes(x = Mean_Score, y = Release_Year)) +
  geom_point(color = 'red', fill = 'red', size = 4, shape = 18, alpha = 0.5) +
  #geom_smooth(color = 'red') + #para poner línea de tendencia
  xlab('Mean_Score') +
  ylab('Release_Year') +
  ggtitle('Relation between Mean_Score & Release_Year') +
  theme_test()
```



Luego, vamos a observar si existe correlación entre estas variables: Observamos que las variables `n` (valoración



absoluta) esta muy correlacionada con la `Mean_Score`, lo cual es lógico:

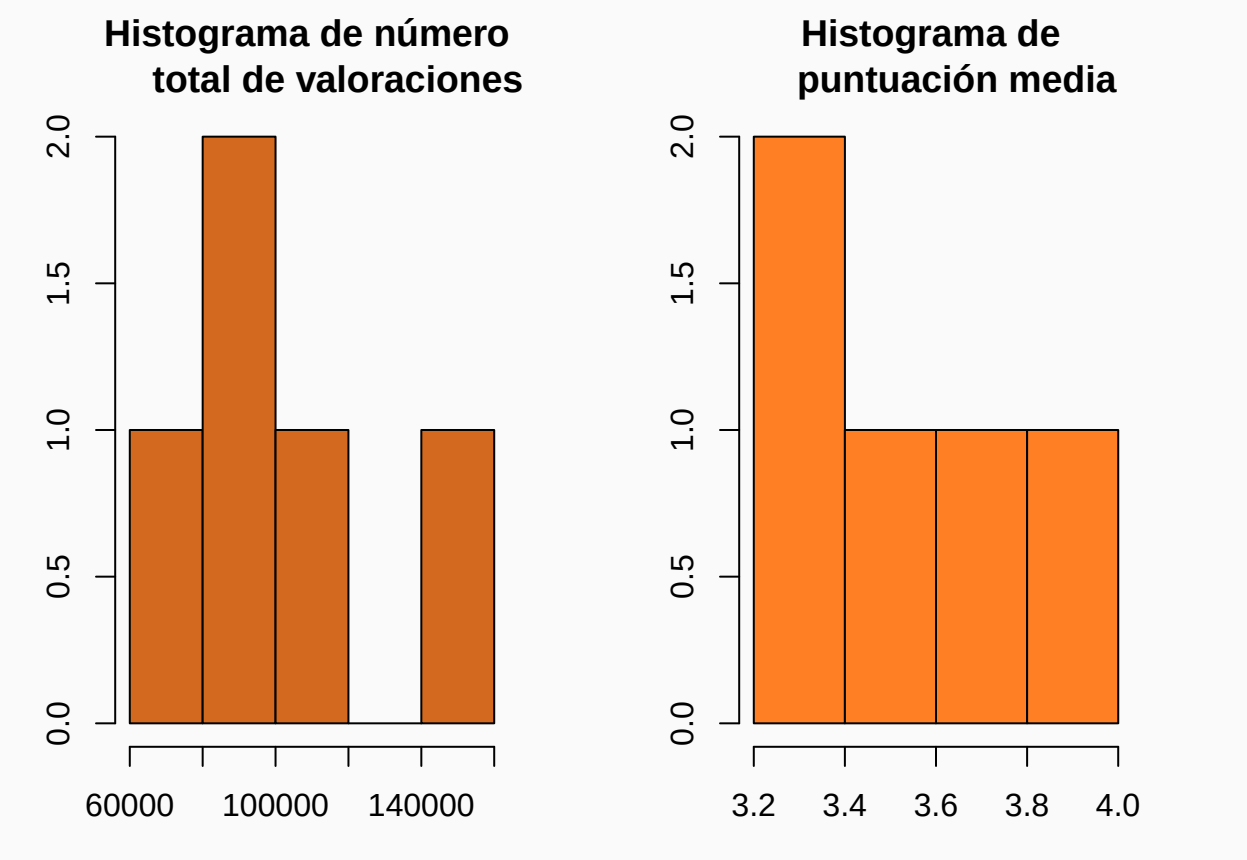


Después, estudiamos la distribución de la variable `Mean_Score`:

Histograma

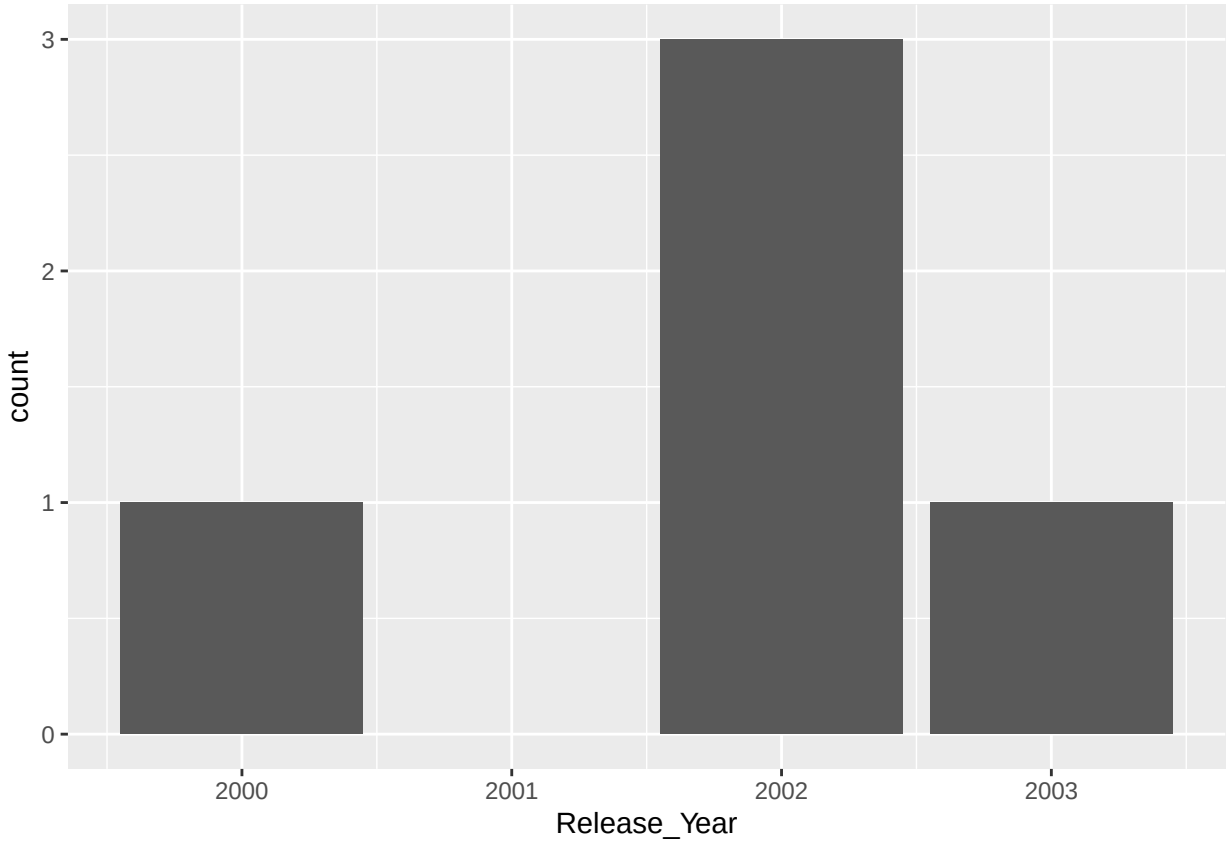
```
library(ggplot2)
```

```
par(bg="grey98", mar=c(3,3,3,3), mfcol=c(1,2))
#Primer gráfico
hist(x = sample$n, main = "Histograma de número
     total de valoraciones",
     xlab = "Valoraciones", ylab = "Frecuencia", col = 'chocolate')
#Segundo gráfico
hist(x = sample$Mean_Score, main = "Histograma de
    puntuación media",
     xlab = "Valoraciones", ylab = "Frecuencia", col = 'chocolate1')
```



ggplot

```
ggplot(data = sample, aes(x = Release_Year), main = Años) + geom_bar()
```



3.6 Pregunta 6

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())
```

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

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

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

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

Month	Mean_Score	n
8	3.454642	147065
10	3.558666	146252
9	3.522692	138154
11	3.503592	136295

Month	Mean_Score	n
7	3.570329	125319
6	3.570904	125042
1	3.491908	119567
12	3.516152	117014
2	3.495188	116994
3	3.485262	116570
5	3.541652	113008
4	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

3.7 Pregunta 7

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

3.8 Pregunta 8

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

3.9 Pregunta 9

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

    1. Puntuaciones por fecha
    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)