

Proyecto Netflix Movies MADM

Laura Moreno, Josep Roman, Paul Ramírez

11/28/2020

Contenidos

1	Objetivo	1
2	Data Wrangle	1
2.1	Importación de datos	1
2.2	Limpieza de los datos	2
3	Estadística Descriptiva	4
4	Sistema de Recomendación / Similaridad (opcional)	9

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 películas 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 película, 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[filas[i]:filas_final[i],])
  }
  else if (data[i]==2){
    scores = rbind(scores,data2[filas[i]:filas_final[i],])
  }
  else if (data[i]==3){
    scores = rbind(scores,data3[filas[i]:filas_final[i],])
  }
  else {
    scores = rbind(scores,data4[filas[i]:filas_final[i],])
  }
}
```

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

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

scores %<%
  mutate(ID1=rep(filas_ID$X1, times=reps)) %>%
  filter(!(filas %in% filas)) %>%
  select(-filas) %>%
  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))
#para separar Date en año mes día
scores %<>% mutate(Date_copy = Date) %>% separate(Date_copy, into = c("Year", "Month", "Day"), sep = "-")
```

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))
```

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

```
setwd("~/Developer/GitHub/proyecto-netflix-movies-madm/Data")
write_csv(scores, "nuestras_pelis.csv")
```

```
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, 2005, ...
## $ Title        <chr> "Avia Vampire Hunter", "Avia Vampire Hunter", "Avia Va..."
```

3 Estadística Descriptiva

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

Veamos más información 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
```

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, 2, 1, ...
## $ Release_Year <int> 2005, 2005, 2005, 2005, 2005, 2005, 2005, 2005, 2005, 2005, ...
## $ Title        <chr> "Avia Vampire Hunter", "Avia Vampire Hunter", "Avia Va..."
```

3.0.1 Variables tipo *int*: MovieID, CustomerID, Score, Release_Year

- *CustomerID*: 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 sería 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.

3.0.2 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.

3.0.3 Variables tipo *chr*: Title

- *Title*: Utilizamos el tipo carácter porque nos interesan objetos que representan un conjunto de letras.
2. Estudia la distribución del número de películas estrenadas por año. Realiza un gráfico de muestra esta distribución haciendo los ajustes necesarios (agrupaciones, cambios de escala, transformaciones. . .)

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

```
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. 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_day_week <- scores %>% mutate(Day_Week = weekdays(Date))
scores_day_week %<>% mutate(Is_Weekend = isWeekend(Date))

day_week_score_avg <- scores_day_week %>%
  group_by(Day_Week) %>%
  summarise(Mean_Score = mean(Score), n = n()) %>%
  arrange(desc(Mean_Score))

kable(day_week_score_avg)
```

Day_Week	Mean_Score	n
Saturday	3.527674	150773
Wednesday	3.526315	250692
Thursday	3.523394	217364
Friday	3.522742	202050
Sunday	3.522538	159776
Monday	3.508918	260777
Tuesday	3.507362	267460

Valoración media entre semana / fin de semana:

```
weekend_weekday_score_avg <- scores_day_week %>%
  group_by(Is_Weekend) %>%
  summarise(Mean_Score = mean(Score), n = n())
kable(weekend_weekday_score_avg)
```

Is_Weekend	Mean_Score	n
FALSE	3.517167	1198343
TRUE	3.525031	310549

```
n_scores_weekend = weekend_weekday_score_avg %>% filter(Is_Weekend == TRUE) %>% select(n)
n_scores = sum(weekend_weekday_score_avg$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
```

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_score_avg <- scores %>%
  group_by(MovieID) %>%
  summarise(Mean_Score = mean(Score), n = n()) %>%
  left_join(titles, by = 'MovieID') %>%
  arrange(desc(Mean_Score))
kable(head(movie_score_avg))
```

MovieID	Mean_Score	n	Release_Year	Title
4353	4.336022	12148	2002	Curb Your Enthusiasm: Season 3
7393	4.255593	2637	1993	Prime Suspect 3
2360	4.101009	29225	1952	Singin' in the Rain
2144	4.079968	2476	2004	VeggieTales: Dave and the Giant Pickle
8382	4.025687	11095	1934	The Thin Man
8940	4.014184	2820	2000	Felicity: Season 3

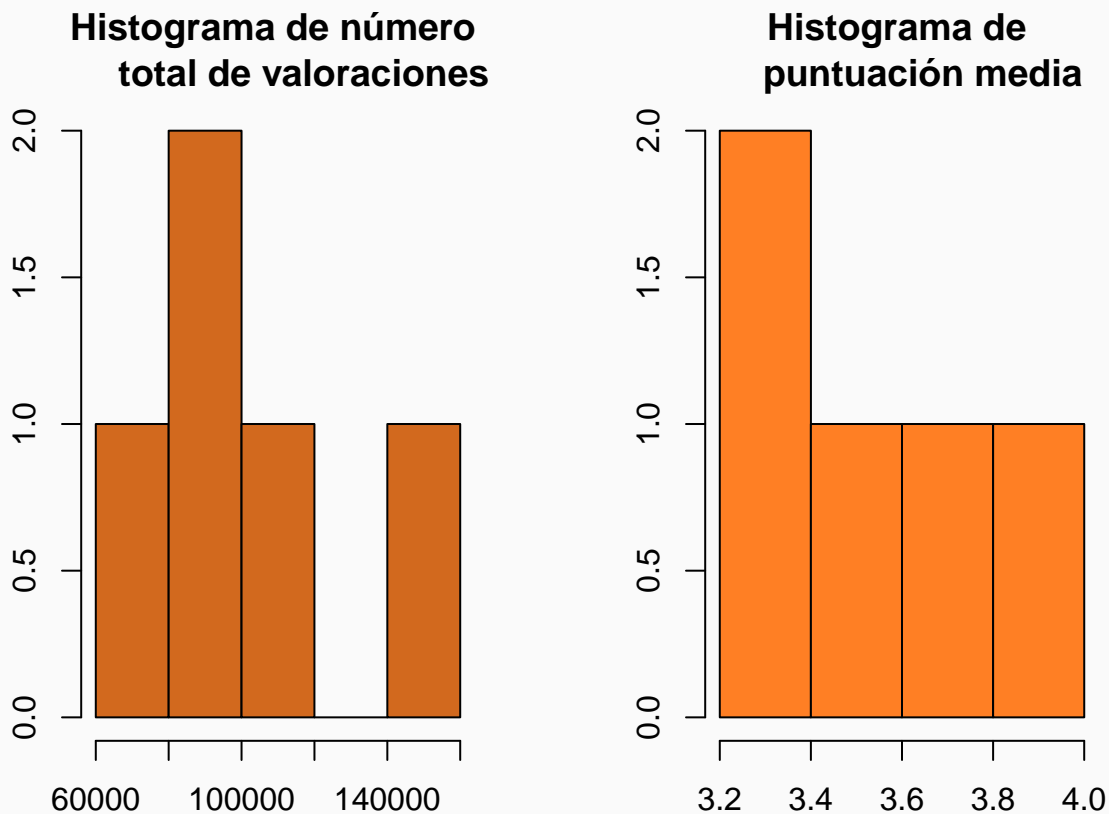
5. De las cinco películas con más número total de valoraciones, compara sus estadísticos y distribuciones (histogramas, boxplot, violin plot, . . .)

```
sample = head(movie_score_avg[order(movie_score_avg$n, decreasing = TRUE)],5)
sample
```

```
## # A tibble: 5 x 5
##   MovieID Mean_Score      n Release_Year Title
##   <int>     <dbl> <int>    <int> <chr>
## 1   6037       3.97 158601    2002 The Bourne Identity
## 2   8387       3.59 118880    2002 Minority Report
## 3  10730       3.36  99910    2002 Gangs of New York
## 4    313       3.69  99812    2000 Pay It Forward
## 5   9645       3.36  79918    2003 Daddy Day Care
```

###Histograma

```
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')
```



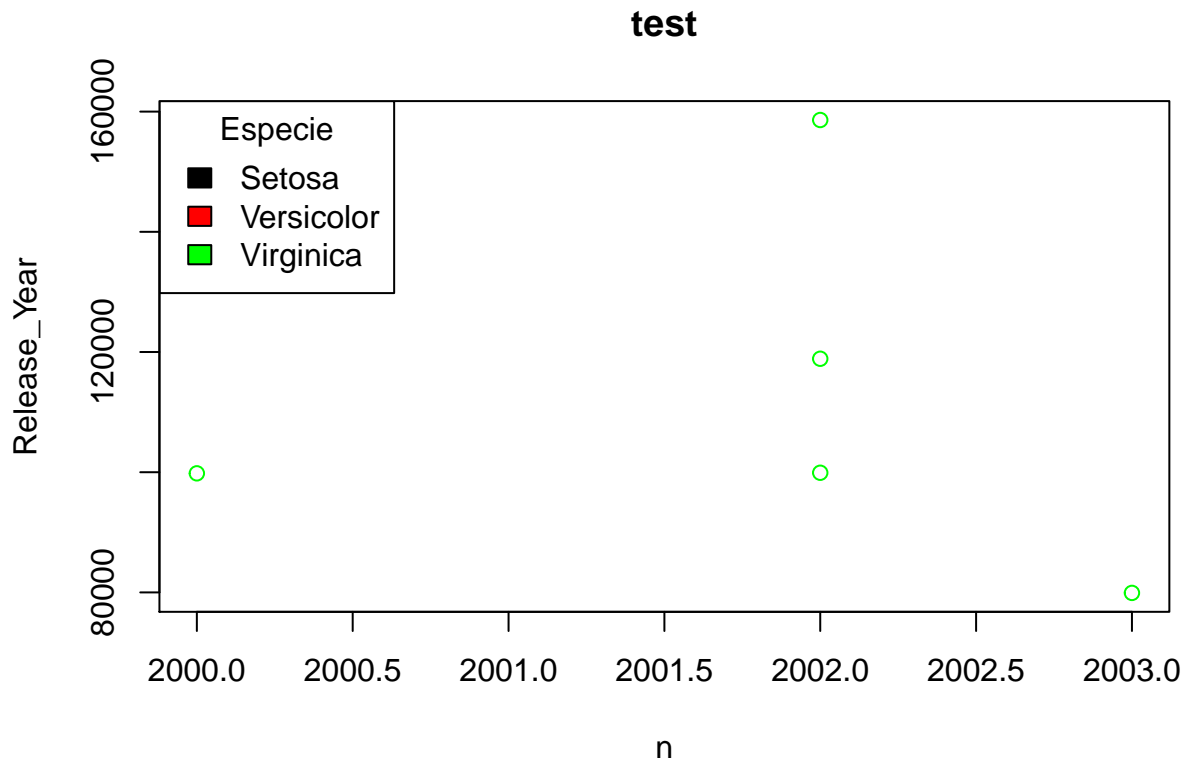
###Plot

```
#Si damos como argumentos la variable Release_Year y la variable lMean_Score, obtenemos una tabla de contingencia, que asignaremos
tab_sample <- table(sample$Release_Year, sample$Mean_Score)
prop.table(tab_sample, margin = 1)
```

```
##
##      3.35769192273046 3.36069471207988 3.58777759084791 3.6893459704244
## 2000      0.0000000      0.0000000      0.0000000      1.0000000
## 2002      0.3333333      0.0000000      0.3333333      0.0000000
## 2003      0.0000000      1.0000000      0.0000000      0.0000000
##
```

```
##      3.97345540065952
## 2000      0.0000000
## 2002      0.3333333
## 2003      0.0000000
```

```
plot(x = sample$Release_Year, y = sample$n, col = 'green',
     main = "test", xlab = "n", ylab = "Release_Year")
legend(x = "topleft", legend = c("Setosa", "Versicolor", "Virginica"),
      fill = c("black", "red", "green"), title = "Especie")
```



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?
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 .
8. Distribución del score promedio por año de las 10 películas con mayor número de valoraciones.
9. Realiza algún gráfico o estudio 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)