TEXAS McCombs
The University of Texas at Austin
McCombs School of Business

Great Learning
POWER AHEAD

Ciencia de Datos y Analítica de Negocios

Python para Ciencia de Datos: NumPy y Pandas - Semana 2

Caso de estudio de MovieLens

Agenda de la sesión

- Introducción a Numpy y Pandas.

- Visión general de los conjuntos de datos.
- Familiarizarse con las funciones de Pandas.
- Extraer información útil de los datos.

Contexto

El sistema de rating de películas que los cinéfilos conocen hoy en día, ha existido desde hace más de 50 años. A lo largo de los años, los estándares culturales y normas han cambiado, al igual que las calificaciones de las películas. Sin embargo, incluso hoy en día, el proceso de calificación de una película sigue siendo un secreto de la industria muy bien guardado.

Objetivo

MovieLens es una empresa en el dominio de Internet y entretenimiento que tiene una base de datos en línea, de información relacionada con películas, series de televisión, contenido de transmisión en línea, incluido el elenco, el equipo de producción, curiosidades, calificaciones y reseñas críticas y de fanáticos. Usted, ha sido contratado como científico de datos por la empresa. Se le han proporcionado los siguientes tres conjuntos de datos, se le ha pedido que lleve a cabo un análisis detallado de los datos y que presente algunas ideas significativas, que ayudarán a la empresa a dirigirse a sus usuarios de una mejor manera.

- **movie.csv**: El archivo contiene información relacionada con las películas y sus géneros. Columns: movie id: identificación de la película, movie title: título de la película, release date: fecha de lanzamiento, Action: Acción, Adventure: Aventura, Animation: Animación, Children's: Infantil, Comedy: Comedia, Crime: Crimen, Documentary: Documental, Drama: Drama, Fantasy: Fantasía, Film-Noir: Cine negro, Horror: Terror, Musical: Musical, Mystery: Misterio, Romance: Romance, Sci-Fi: Ciencia ficción, Thriller: Suspense, War: Guerra, Western: Western
- **user.csv**: Contiene información sobre los usuarios que han calificado las películas. Columnas: user id: identificación del usuario, age: edad, gender: género, occupation: ocupación, zip code: código postal
- **ratings.csv**: Contiene información de las valoraciones dadas por los usuarios a una determinada película. Columnas: user id: identificación del usuario, movie id: identificación de la película, rating: calificación, timestamp: marca de tiempo

Uno de los primeros pasos para realizar cualquier análisis es importar las librerías necesarias que nos ayudarán a realizar diversas operaciones sobre los datos.

NumPy, Pandas son las librerías de Python más utilizadas en la Ciencia de Datos. Proporciona estructuras y herramientas de análisis de datos de alto rendimiento y fáciles de usar.

Entonces, primero importemos **NumPy y Pandas** para que podamos utilizar las funciones disponibles en estas librerías para analizar mejor nuestros datos.**

Importando los paquetes necesarios

```
import pandas as pd
import numpy as np
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mour
```

Ahora hemos importado pandas como pd y numpy como np. Aquí la palabra 'as' se usa como un alias.

Carguemos los tres conjuntos de datos usando la función Pandas **read**, para que con ello podamos usarlos y comenzar con nuestro análisis.

Leyendo los conjuntos de datos

```
# Leer conjuntos de datos usando 'read_csv' del paquete de pandas
movie = pd.read_csv("/content/movie.csv")
user = pd.read_csv("/content/user.csv")
ratings = pd.read_csv("/content/ratings.csv")
```

Resumen de datos

Los pasos iniciales para comprender cualquier conjunto de datos son:

- Observe las primeras filas del conjunto de datos, para verificar si el conjunto de datos se ha cargado correctamente o no.
- Obtener información sobre el número de filas y columnas en el conjunto de datos.
- Averigüe los tipos de datos de las columnas para garantizar que los datos se almacenen en el formato preferido y que el valor de cada propiedad sea el esperado.

- Verifique el resumen estadístico del conjunto de datos para obtener una descripción general de las columnas numéricas de los datos.

Profundicemos ahora en cada uno de los conjuntos de datos individualmente y comprendamos bien estos aspectos.

Conjunto de datos de usuario

Mostrando las primeras 5 filas del conjunto de datos del usuario

La función **head()** nos ayudará a echar un vistazo a las primeras 5 filas del conjunto de datos

```
user.head()
```

	user id	age	gender	occupation	zip code
0	1	24	M	technician	85711
1	2	53	F	other	94043
2	3	23	M	writer	32067
3	4	24	M	technician	43537
4	5	33	F	other	15213

```
user.tail
```

```
pandas.core.generic.NDFrame.tail
def tail(n: int=5) -> NDFrameT
```

```
5     parrot
6      shark
7      whale
8      zebra

Viewing the last 5 lines
```

El conjunto de datos del usuario se ha cargado correctamente.

- *El conjunto de datos del usuario contiene información demográfica sobre los usuarios que clasificaron las películas*.
- *La columna `gender` tiene valores como M y F, donde M representa usuarios masculinos y F representa usuarios femeninos*

- *También podemos observar que los usuarios tienen diferentes antecedentes ocupacionales como 'technician', 'writer', etc.*
- *La columna `zip code` representa el área donde reside el usuario*

Comprender la forma del conjunto de datos del usuario

Ahora verifiquemos las dimensiones del conjunto de datos usando el atributo ***shape***.

El atributo ***shape*** nos ayudará a obtener el número de filas y columnas del conjunto de datos.
Conocer las dimensiones del database nos ayudará a comprender la cantidad de puntos de datos con los que estamos trabajando y demuestra ser beneficioso, especialmente cuando el conjunto de datos es grande (que consta de una gran cantidad de filas y columnas).

```
user.shape
```

```
(943, 5)
```

- *Hay 943 filas y 5 columnas en el conjunto de datos del usuario.*

Comprobando los tipos de datos de las columnas para el conjunto de datos del usuario

Ahora somos conscientes de la forma del dataframe. Ahora echemos un vistazo a los tipos de datos con los que estamos trabajando.

La función ***info()*** nos ayudará a comprender los tipos de datos de las columnas.
Comprender los tipos de datos garantiza que los datos se almacenen en el formato preferido y que el valor de cada propiedad sea el esperado.

```
user.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 943 entries, 0 to 942
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   user id     943 non-null    int64 
 1   age         943 non-null    int64 
 2   gender      943 non-null    object
 3   occupation  943 non-null    object
 4   zip code    943 non-null    object
dtypes: int64(2), object(3)
memory usage: 37.0+ KB
```

- *Podemos ver que las columnas `gender`, `occupation` y `zip code` son de tipo 'objeto' mientras que las otras columnas son de tipo 'entero', ya que contienen números en ellas.*
- *El tipo de datos de la columna `zip code` es un 'objeto' que parece ser incorrecto y necesita más investigación ya que esta columna parece contener solo valores 'enteros', como se ve en las primeras 5 filas del dataframe.*

Obtención del resumen estadístico para el conjunto de datos del usuario

Ahora podemos echar un vistazo al resumen estadístico del conjunto de datos.

La función **describe()** nos ayudará a encontrar el resumen estadístico.

El resumen estadístico nos dará una visión general de las columnas numéricas de los datos y mostrará varias características como el mínimo, el máximo, la media, la desviación estándar, etc. de las columnas. Esto nos ayudará a comprender la distribución de los valores presentes en las columnas numéricas.

```
user.describe()
```

	user id	age
count	943.000000	943.000000
mean	472.000000	34.051962
std	272.364951	12.192740
min	1.000000	7.000000
25%	236.500000	25.000000
50%	472.000000	31.000000
75%	707.500000	43.000000
max	943.000000	73.000000

```
user.describe(include='all')
```

	user id	age	gender	occupation	zip code
count	943.000000	943.000000	943	943	943
unique	NaN	NaN	2	21	795
top	NaN	NaN	M	student	55414
freq	NaN	NaN	670	196	9
mean	472.000000	34.051962	NaN	NaN	NaN
std	272.364951	12.192740	NaN	NaN	NaN
min	1.000000	7.000000	NaN	NaN	NaN
25%	236.500000	25.000000	NaN	NaN	NaN
50%	472.000000	31.000000	NaN	NaN	NaN
75%	707.500000	43.000000	NaN	NaN	NaN
max	943.000000	73.000000	NaN	NaN	NaN

- *Podemos ver en la columna `age` que la edad promedio de los usuarios que clasificaron las películas es de 34 años*.
- *La edad de los usuarios va desde los 7 años (mínimo) hasta los 73 años (máximo)*.
- *Dado que el ID de usuario es un identificador único de los usuarios que clasificaron las películas, la interpretación de sus estadísticas resumidas no brindará información significativa.*

Ahora tenemos una idea básica del conjunto de datos del usuario. Realicemos ahora operaciones similares en los otros dos conjuntos de datos e intentemos encontrar información relevante de ellos también.

Conjunto de datos de la película

Mostrando las primeras 5 filas del conjunto de datos de la película

```
movie.head()
```

	movie id	movie title	release date	Action	Adventure	Animation	Childrens	Comedy	Crime	Do
0	1	Toy Story	1-Jan-95	0	0	1	1	1	0	
1	2	GoldenEye	1-Jan-95	1	1	0	0	0	0	
2	3	Four Rooms	1-Jan-95	0	0	0	0	0	0	
3	4	Get Shorty	1-Jan-95	1	0	0	0	1	0	
4	5	Copycat	1-Jan-95	0	0	0	0	0	1	

5 rows × 21 columns

- *Como podemos ver, el conjunto de datos de 'movie' contiene información relacionada con las películas y sus géneros. Mientras que `movie id`, `movie title` y `released date` son información relacionada con las películas, el resto de las columnas son específicamente los géneros de las películas.*
- *Las columnas asociadas a los géneros de las películas tienen valores de 0 y 1. El valor '1' significa que una película en particular pertenece a ese género, mientras que el valor '0' representa que la película no es parte de ese género.*
- *También podemos ver que una película puede tener varios géneros, ya que hay más de una columna que tiene el valor '1' para una película en particular en las columnas asociadas con los géneros.*
- *Por ejemplo, la película llamada 'Toy Story' tiene géneros como 'Animation', 'Childrens' y 'Comedy'.*

Comprender la forma del conjunto de datos de la película

```
movie.shape
```

```
(1680, 21)
```

- *Hay 1680 filas y 21 columnas en el conjunto de datos de la película.*

Comprobación de los tipos de datos de las columnas para el conjunto de datos de la película

```
movie.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1680 entries, 0 to 1679
Data columns (total 21 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   movie id      1680 non-null   int64 
 1   movie title   1680 non-null   object
 2   release date  1680 non-null   object
 3   Action        1680 non-null   int64 
 4   Adventure     1680 non-null   int64 
 5   Animation     1680 non-null   int64 
 6   Childrens     1680 non-null   int64 
 7   Comedy        1680 non-null   int64 
 8   Crime         1680 non-null   int64 
 9   Documentary   1680 non-null   int64 
 10  Drama         1680 non-null   int64 
 11  Fantasy       1680 non-null   int64 
 12  Film-Noir     1680 non-null   int64 
 13  Horror        1680 non-null   int64 
 14  Musical       1680 non-null   int64 
 15  Mystery       1680 non-null   int64 
 16  Romance       1680 non-null   int64 
 17  Sci-Fi        1680 non-null   int64 
 18  Thriller      1680 non-null   int64 
 19  War           1680 non-null   int64 
 20  Western       1680 non-null   int64 
dtypes: int64(19), object(2)
memory usage: 275.8+ KB
```

- *Podemos ver que las columnas `movie title` y `release date` son de tipo objeto ya que estas columnas tienen algo de texto.*
- *Todas las demás columnas son de tipo entero ya que contienen números en ellas.*

Obtención del resumen estadístico del conjunto de datos de la película

```
movie.describe()
```

	movie id	Action	Adventure	Animation	Childrens	Comedy	
count	1680.000000	1680.000000	1680.000000	1680.000000	1680.000000	1680.000000	1680.0
mean	841.525595	0.149405	0.080357	0.025000	0.072619	0.300595	0.0
std	485.609591	0.356593	0.271926	0.156171	0.259587	0.458653	0.2
min	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
25%	421.750000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
50%	841.500000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
75%	1261.250000	0.000000	0.000000	0.000000	0.000000	1.000000	0.0
max	1682.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.0

```
movie.describe().T
```

	count	mean	std	min	25%	50%	75%	max
movie id	1680.0	841.525595	485.609591	1.0	421.75	841.5	1261.25	1682.0
Action	1680.0	0.149405	0.356593	0.0	0.00	0.0	0.00	1.0
Adventure	1680.0	0.080357	0.271926	0.0	0.00	0.0	0.00	1.0
Animation	1680.0	0.025000	0.156171	0.0	0.00	0.0	0.00	1.0
Childrens	1680.0	0.072619	0.259587	0.0	0.00	0.0	0.00	1.0
Comedy	1680.0	0.300595	0.458653	0.0	0.00	0.0	1.00	1.0
Crime	1680.0	0.064881	0.246389	0.0	0.00	0.0	0.00	1.0
Documentary	1680.0	0.029762	0.169980	0.0	0.00	0.0	0.00	1.0
Drama	1680.0	0.431548	0.495440	0.0	0.00	0.0	1.00	1.0
Fantasy	1680.0	0.013095	0.113717	0.0	0.00	0.0	0.00	1.0
Film-Noir	1680.0	0.014286	0.118701	0.0	0.00	0.0	0.00	1.0
Horror	1680.0	0.054762	0.227583	0.0	0.00	0.0	0.00	1.0
Musical	1680.0	0.033333	0.179559	0.0	0.00	0.0	0.00	1.0
Mystery	1680.0	0.036310	0.187115	0.0	0.00	0.0	0.00	1.0
Romance	1680.0	0.147024	0.354235	0.0	0.00	0.0	0.00	1.0
Sci-Fi	1680.0	0.060119	0.237778	0.0	0.00	0.0	0.00	1.0
Thriller	1680.0	0.149405	0.356593	0.0	0.00	0.0	0.00	1.0
War	1680.0	0.042262	0.201246	0.0	0.00	0.0	0.00	1.0
Western	1680.0	0.016071	0.125788	0.0	0.00	0.0	0.00	1.0

- *El resumen estadístico de la columna `movie_id` no proporcionará información significativa, ya que es un identificador único de cada una de las películas. El resumen estadístico de las columnas relacionadas con los géneros tampoco es de mucha ayuda, ya que estas columnas solo contienen valores de 0 y 1.*

Ahora también hemos explorado el conjunto de datos de la película. Veamos ahora finalmente el dataframe de calificaciones.

Calificaciones (Rating)

Mostrando las primeras 5 filas del conjunto de datos de calificaciones

```
ratings.head()
```

	user id	movie id	rating	timestamp
0	196	242	3	881250949
1	186	302	3	891717742
2	22	377	1	878887116
3	244	51	2	880606923
4	166	346	1	886397596

El conjunto de datos de calificaciones también se ha importado correctamente sin ningún error.

- *El dataframe de calificaciones contiene información sobre las calificaciones otorgadas por los usuarios a una película en particular.*

Comprender la forma del conjunto de datos de calificaciones

```
ratings.shape
```

```
(100000, 4)
```

- *Hay 100000 filas y 4 columnas en el conjunto de datos de calificaciones. Este es un conjunto de datos bastante grande en comparación con los dos dataframes anteriores.*

Comprobación de los tipos de datos de las columnas para el conjunto de datos de calificaciones

```
ratings.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100000 entries, 0 to 99999
Data columns (total 4 columns):
 #   Column     Non-Null Count   Dtype
---  ------     --------------   -----
 0   user id    100000 non-null  int64
 1   movie id   100000 non-null  int64
 2   rating     100000 non-null  int64
 3   timestamp  100000 non-null  int64
dtypes: int64(4)
memory usage: 3.1 MB
```

- *Aquí todas las columnas son del tipo de datos enteros ya que contienen números en ellas.*

Obtención del resumen estadístico para el conjunto de datos de clasificaciones

```
ratings.describe()
```

	user id	movie id	rating	timestamp
count	100000.00000	100000.000000	100000.000000	1.000000e+05
mean	462.48475	425.530130	3.529860	8.835289e+08
std	266.61442	330.798356	1.125674	5.343856e+06
min	1.00000	1.000000	1.000000	8.747247e+08
25%	254.00000	175.000000	3.000000	8.794487e+08
50%	447.00000	322.000000	4.000000	8.828269e+08
75%	682.00000	631.000000	4.000000	8.882600e+08
max	943.00000	1682.000000	5.000000	8.932866e+08

- *Los números de la columna `timestamp` no transmiten ninguna información significativa. Sin embargo, en la columna "rating", podemos ver que la calificación promedio de todas las películas es de alrededor de 3.53, mientras que las calificaciones oscilan entre 1 y 5.*

Ahora tenemos una idea considerable sobre los conjuntos de datos con los que estamos trabajando. Es hora de profundizar y encontrar respuestas a algunas preguntas importantes que nos ayudarán a tener una mejor idea sobre la industria.

Extraigamos algunas ideas significativas de los datos respondiendo algunas preguntas orientadas a los negocios

¿Cuántas películas pertenecen a un género en particular?

Habíamos visto anteriormente que una película en particular puede tener múltiples géneros. Así que investiguemos más sobre esto.

¿Pero recuerdas qué conjunto de datos contiene información sobre géneros de películas?

Sí, el conjunto de datos de la película contiene esta información. Así que tratemos de mirar eso.

```
movie.head()
```

	movie id	movie title	release date	Action	Adventure	Animation	Childrens	Comedy	Crime	Do
0	1	Toy Story	1-Jan-95	0	0	1	1	1	0	
1	2	GoldenEye	1-Jan-95	1	1	0	0	0	0	
2	3	Four Rooms	1-Jan-95	0	0	0	0	0	0	
3	4	Get Shorty	1-Jan-95	1	0	0	0	1	0	
4	5	Copycat	1-Jan-95	0	0	0	0	0	1	

5 rows × 21 columns

Analicemos este conjunto de datos usando algunas funciones de Pandas

Funciones de Pandas

Obteniendo todas las columnas del conjunto de datos de la película

Imprimamos todas las columnas del conjunto de datos de la película para que podamos tener una mejor idea de qué géneros hay.

```
movie.columns
```

```
Index(['movie id', 'movie title', 'release date', 'Action', 'Adventure',
       'Animation', 'Childrens', 'Comedy', 'Crime', 'Documentary', 'Drama',
       'Fantasy', 'Film-Noir', 'Horror', 'Musical', 'Mystery', 'Romance',
       'Sci-Fi', 'Thriller', 'War', 'Western'],
      dtype='object')
```

- *Podemos ver que todas las columnas que van desde 'Acción' hasta 'Western' son columnas que están relacionadas con los géneros de las películas.*

Dado que hay muchos géneros presentes en este conjunto de datos, creemos una variable para almacenar todos estos valores como una lista para que podamos usarla para un análisis posterior.

```
genres = ['Action', 'Adventure',
       'Animation', 'Childrens', 'Comedy', 'Crime', 'Documentary', 'Drama',
       'Fantasy', 'Film-Noir', 'Horror', 'Musical', 'Mystery', 'Romance',
       'Sci-Fi', 'Thriller', 'War', 'Western']


# El uso de la función 'len()' para calcular el número de géneros
print('The number of genres present in the movie dataset are ', len(genres))
```

```
The number of genres present in the movie dataset are  18
```

- ***Hay 18 géneros en total.***

Una de las formas de encontrar la respuesta es tomar todas las columnas de género y encontrar la suma de cada columna. Aquí, en lugar de escribir las columnas de género nuevamente, usaremos la lista de 'géneros'.

```
# Imprimir la lista de los generos
genres
```

```
['Action',
 'Adventure',
 'Animation',
 'Childrens',
 'Comedy',
 'Crime',
 'Documentary',
 'Drama',
 'Fantasy',
 'Film-Noir',
 'Horror',
 'Musical',
 'Mystery',
 'Romance',
 'Sci-Fi',
 'Thriller',
 'War',
 'Western']
```

```
movie[genres].sum()
```

```
Action         251
Adventure      135
Animation       42
Childrens      122
Comedy         505
```

```
Crime          109
Documentary     50
Drama          725
Fantasy         22
Film-Noir       24
Horror          92
Musical         56
Mystery         61
Romance        247
Sci-Fi         101
Thriller       251
War             71
Western         27
dtype: int64
```

```
movie[genres].sum(axis=0)
```

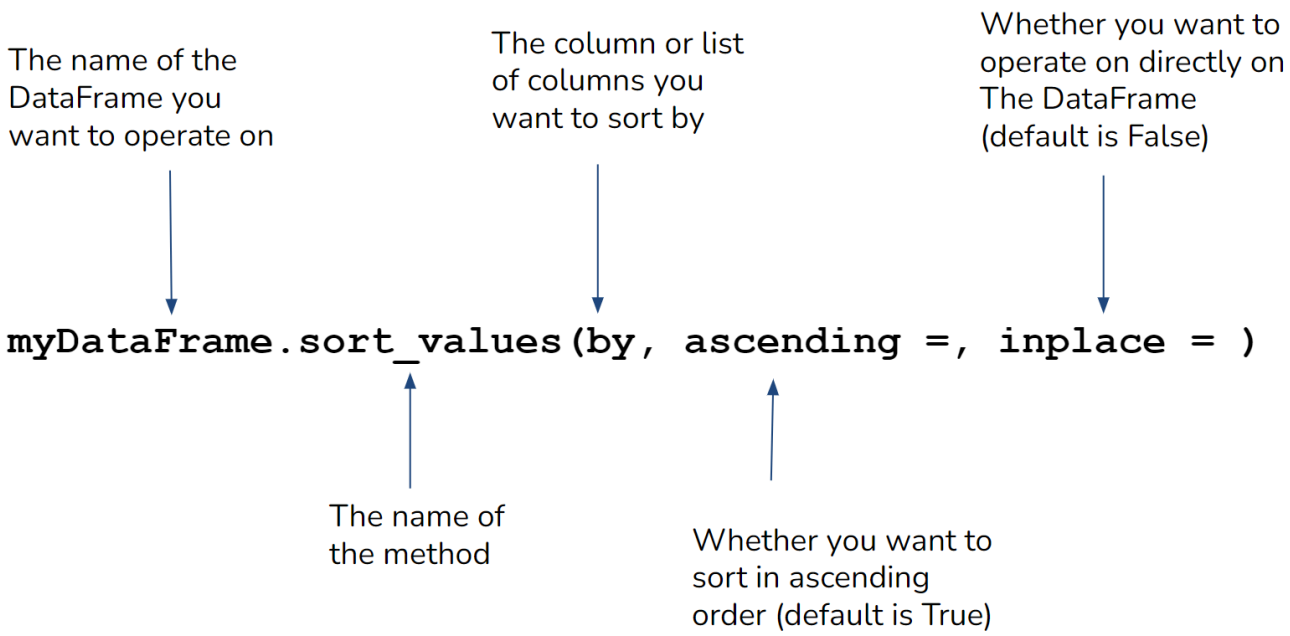
```
Action         251
Adventure      135
Animation       42
Childrens      122
Comedy         505
Crime          109
Documentary     50
Drama          725
Fantasy         22
Film-Noir       24
Horror          92
Musical         56
Mystery         61
Romance        247
Sci-Fi         101
Thriller       251
War             71
Western         27
dtype: int64
```

- *Aquí, podemos ver que el género 'Action' tiene 251 películas, el género 'Aventure' tiene 135 películas y así sucesivamente. Sin embargo, hubiera sido mucho mejor si tuviéramos una serie en la que los números estuvieran dispuestos en orden ascendente o descendente para que pudiéramos encontrar fácilmente los géneros con el mayor y menor número de películas.*

Esto se puede lograr simplemente usando la función **sort_values()** para organizar los números en orden ascendente o descendente.

función sort_values()

La función sort_values() de Pandas ordena un dataframe en orden ascendente o descendente de la columna pasada. Es diferente a la función ordenada de Python, ya que no puede ordenar un dataframe y no se puede seleccionar una columna en particular.




```
movie[genres].sum().sort_values(ascending = False) #ascending= False ordenará los valores er
```

```
Drama          725
Comedy         505
Action         251
Thriller       251
Romance        247
Adventure      135
Childrens      122
Crime          109
Sci-Fi         101
Horror          92
War             71
Mystery         61
Musical         56
Documentary     50
Animation       42
Western         27
Film-Noir       24
Fantasy         22
dtype: int64
```

Perspectivas

- *Las películas se distribuyen en 18 géneros en total.*
- *Podemos observar que el género 'Drama' tiene el mayor número de películas con 725 en total seguido de 'Comedy' con 505.*

- *Los cinco géneros principales con el mayor número de películas son 'Drama', 'Comedy', 'Action', 'Thriller' y 'Romance'.*
- *El género 'Fantasy' tiene el menor número de películas (22 en total).*

¿Podemos encontrar las películas que tienen más de un género?

Como se discutió anteriormente, una película en particular puede tener más de un género, por lo que para responder a la pregunta, podemos usar los pasos a continuación:

- El enfoque básico para encontrar esta información será agregar una nueva columna al dataframe de la película que contendrá la suma de 1 de las columnas de género a lo largo de la fila, para una película en particular.
- Después de esto, podemos averiguar la cantidad de películas que tienen más de un género simplemente seleccionando las filas donde la nueva columna agregada tiene un valor de más de 1.

Entonces, creemos una nueva columna llamada "Number of Genres" y agréguela al dataframe.

```
movie["Number of Genres"] = movie.loc[:, genres].sum(axis=1)
```

```
movie.loc[:, genres].sum(axis=1)
```

```
0       3
1       3
2       1
3       3
4       3
       ..
1675    1
1676    2
1677    2
1678    1
1679    1
Length: 1680, dtype: int64
```

- *El código anterior agrega una nueva columna llamada `Number of Genres` al dataframe. Esta nueva columna consiste en la suma de 1 de las columnas de género a lo largo de la fila para una película en particular.*

- *movie.loc[:, genders] - Esto recoge todas las filas y columnas que están presentes en la lista de géneros*.
- *El parámetro eje = 1 se usa para obtener la suma de los valores a lo largo de la fila, mientras que el eje = 0 se usa para obtener la suma de los valores a lo largo de la columna*.

Ahora imprimamos el encabezado de la columna de la película y veamos si la nueva columna se ha agregado o no.

```
movie.head()
```

	movie id	movie title	release date	Action	Adventure	Animation	Childrens	Comedy	Crime	Do
0	1	Toy Story	1-Jan-95	0	0	1	1	1	0	
1	2	GoldenEye	1-Jan-95	1	1	0	0	0	0	
2	3	Four Rooms	1-Jan-95	0	0	0	0	0	0	
3	4	Get Shorty	1-Jan-95	1	0	0	0	1	0	
4	5	Copycat	1-Jan-95	0	0	0	0	0	1	

5 rows × 22 columns

- *Como podemos ver, se ha agregado una nueva columna llamada `Número de géneros` al final del marco de datos*

Ahora recojamos las filas donde `Number of Genres` > 1 para que podamos encontrar la cantidad de películas con más de 1 género.

```
movie[movie['Number of Genres']>1] #esto creará un subconjunto de los datos según la condici
```

	movie id	movie title	release date	Action	Adventure	Animation	Childrens	Comedy	Crime
0	1	Toy Story	1-Jan-95	0	0	1	1	1	0
1	2	GoldenEye	1-Jan-95	1	1	0	0	0	0
3	4	Get Shorty	1-Jan-95	1	0	0	0	1	0
4	5	Copycat	1-Jan-95	0	0	0	0	0	1
6	7	Twelve Monkeys	1-Jan-95	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...
1666	1669	MURDER and murder	20-Jun-97	0	0	0	0	0	1
1667	1670	Tainted	1-Feb-98	0	0	0	0	1	0
1670	1673	Mirage	1-Jan-95	1	0	0	0	0	0
1676	1679	B. Monkey	6-Feb-98	0	0	0	0	0	0
1677	1680	Sliding Doors	1-Jan-98	0	0	0	0	0	0

849 rows × 22 columns

Imprimamos la forma de este marco de datos para que sepamos la cantidad de películas que tienen más de un género.

```
movie[movie['Number of Genres']>1].shape
```

```
(849, 22)
```

- *Hay 849 películas que tienen más de un género.*

```
movie.shape
```

```
(1680, 22)
```

- *El conjunto de datos de la película original contenía 1680 películas. Entonces, de 1680 películas, 849, es decir, alrededor del 50 % de las películas tienen más de un género.*

Perspectivas

- *De 1680 películas, 849 películas que comprenden alrededor del 50% tienen más de un género y 831 películas tienen un solo género*.
- *Esto muestra que tenemos una proporción casi igual (50:50) de películas que tienen más de un género y que tienen un solo género*.

¿Podemos encontrar los géneros que más gustan a los usuarios?

Como ya hemos visto, la información sobre las calificaciones está presente en el conjunto de datos de calificaciones, sin embargo, las columnas de género están presentes en el conjunto de datos de la película. La única columna que conecta estos dos conjuntos de datos es la columna de identificación de la película. Podemos 'merge' ambos conjuntos de datos sobre la base de una variable común que es 'movie_id'.

Merge/Join / Concatenation
Los pandas brindan varias facilidades para combinar fácilmente Series o DataFrame con varios tipos de lógica de conjuntos para los índices y la funcionalidad de álgebra relacional en el caso de operaciones de Join/Merge.

Aquí están los diferentes tipos de Join (uniones):

Inner Join: Devuelve registros que tienen valores coincidentes en ambas tablas.

Inner Join

Index	Id	Name	Age
0	1	Alex	25
1	2	Amy	23
2	3	Allen	22
3	4	Alice	21
4	5	Ayoung	24

Left Table

Index	Id	Subject
0	1	sub2
1	2	sub4
2	3	sub3
3	4	sub6
4	5	sub5

Right Table

Index	Id	Name	Age	Subject
0	1	Alex	25	sub2
1	2	Amy	23	sub4
2	3	Allen	22	sub3
3	4	Alice	21	sub6
4	5	Ayoung	24	sub5

After Merging datasets on Id

Syntax : `merged = pd.merge(left, right, on = 'id')`

Left Join: Devuelve todos los registros de la tabla izquierda y los registros coincidentes de la tabla derecha.

Left Join

Index	Customer_id	Product
0	1	Oven
1	2	Oven
2	3	Oven
3	4	Television
4	5	Television
5	6	Television

Left Table

Index	Customer_id	State
0	2	California
1	4	California
2	6	Texas

Right Table

Index	Customer_id	Product	State
0	1	Oven	nan
1	2	Oven	California
2	3	Oven	nan
3	4	Television	California
4	5	Television	nan
5	6	Television	Texas

After Left Join datasets on Customer_id

Syntax : `merged = pd.merge(left, right, on = 'Customer_id', how = 'left')`

Right Join: Devuelve todos los registros de la tabla derecha y los registros coincidentes de la tabla izquierda.

Right Join

Index	Customer_id	Product
0	1	Oven
1	2	Oven
2	3	Oven
3	4	Television
4	5	Television
5	6	Television

Left Table

Index	Customer_id	State
0	2	California
1	4	California
2	6	Texas

Right Table

Index	Customer_id	State	State
0	2	Oven	California
1	4	Television	California
2	6	Television	Texas

After right Join datasets on Customer_id

Syntax : `merged = pd.merge(left, right, on = 'Customer_id', how = 'right')`

Outer/Full Join: Devuelve todos los registros cuando hay una coincidencia en la tabla izquierda o derecha.

Outer/Full Join

Index	Customer_id	Product
0	1	Oven
1	2	Oven
2	3	Oven
3	4	Television
4	5	Television
5	6	Television

Left Table

Index	Customer_id	State
0	2	California
1	4	California
2	6	Texas

Right Table

Index	Customer_id	Product	State
0	1	Oven	nan
1	2	Oven	California
2	3	Oven	nan
3	4	Television	California
4	5	Television	nan
5	6	Television	Texas

After Outer Join datasets on Customer_id

Syntax : `merged = pd.merge(left, right, on = 'Customer_id', how = 'outer')`

```
df_merge = movie.merge(ratings, on = 'movie id', how = 'inner')
df_merge.head()
```

	movie id	movie title	release date	Action	Adventure	Animation	Childrens	Comedy	Crime	Documer
0	1	Toy Story	1-Jan-95	0	0	1	1	1	0	
1	1	Toy Story	1-Jan-95	0	0	1	1	1	0	
2	1	Toy Story	1-Jan-95	0	0	1	1	1	0	
3	1	Toy Story	1-Jan-95	0	0	1	1	1	0	
4	1	Toy Story	1-Jan-95	0	0	1	1	1	0	

5 rows × 25 columns

- *Como podemos ver, las columnas `user id`, `rating`, `timestamp` se agregan al final del marco de datos*

Para encontrar los géneros que más les gustan a los usuarios, podemos encontrar las calificaciones promedio de cada uno de los géneros y luego compararlos.
Para esto podemos usar la lista de `géneros`.

```
genres
```

```
['Action',
 'Adventure',
```

```
 'Animation',
 'Childrens',
 'Comedy',
 'Crime',
 'Documentary',
 'Drama',
 'Fantasy',
 'Film-Noir',
 'Horror',
 'Musical',
 'Mystery',
 'Romance',
 'Sci-Fi',
 'Thriller',
 'War',
 'Western']
```

Ejecutaremos un bucle en el dataframe fusionado para las columnas que están presentes en la lista de géneros. Siempre que se encuentre que un género en particular tiene el valor '1' en su columna, se extraerá la calificación correspondiente a ese valor '1' y para ello calcularemos el valor promedio de todas estas calificaciones extraídas para un género en particular.

```
for i in genres:
    print(i,':' , df_merge[df_merge[i]==1].rating.mean())
```

```
Action : 3.480245417953027
Adventure : 3.503526503308369
Animation : 3.5766990291262135
Childrens : 3.3532442216652742
Comedy : 3.3940734781442745
Crime : 3.6322780881440098
Documentary : 3.6728232189973613
Drama : 3.6873793708484772
Fantasy : 3.2152366863905324
Film-Noir : 3.9215233698788228
Horror : 3.2903893172841827
Musical : 3.521396851029471
Mystery : 3.63813155386082
Romance : 3.621704948358255
Sci-Fi : 3.5607227022780834
Thriller : 3.5090069495245064
War : 3.815811874866993
Western : 3.6132686084142396
```

Perspectivas

```
df_merge[df_merge[i]==1].rating
```

```
6985     2
6986     2
6987     2
6988     3
6989     5
        ..
99353    3
99354    4
99355    2
99356    3
99804    1
Name: rating, Length: 1854, dtype: int64
```

- *El género 'Film-Noir' tiene las calificaciones promedio más altas con una calificación promedio de 3.92 seguido del género 'War' con una calificación de 3.81*.
- *El género 'Fantasy' tiene las calificaciones promedio más bajas con una calificación promedio de 3.21*.
- *Entre 18 géneros, solo 5 géneros han recibido una calificación inferior a 3.5, lo que significa que alrededor del 72 % de los géneros tienen una calificación superior al promedio*.

Hasta ahora, hemos realizado análisis sobre los géneros y las películas y generado algunas ideas.

¿Podemos encontrar qué películas han sido las más preferidas por los usuarios?

Comencemos revisando las películas mejor calificadas por los usuarios. Como ya sabemos, los conjuntos de datos `movie` y `ratings` tienen la información relacionada con las calificaciones de las películas.
Por lo tanto, podemos *combinar* ambos conjuntos de datos sobre la base de una variable común que es `movie_id` y obtener las calificaciones promedio de todas las películas. Como ya hemos fusionado estos conjuntos de datos antes, podemos llamarlo nuevamente para obtener los valores

```
df_merge.head()
```

	movie id	movie title	release date	Action	Adventure	Animation	Childrens	Comedy	Crime	Documer
0	1	Toy Story	1-Jan-95	0	0	1	1	1	0	
1	1	Toy Story	1-Jan-95	0	0	1	1	1	0	
2	1	Toy Story	1-Jan-95	0	0	1	1	1	0	
3	1	Toy Story	1-Jan-95	0	0	1	1	1	0	
4	1	Toy Story	1-Jan-95	0	0	1	1	1	0	

5 rows × 25 columns

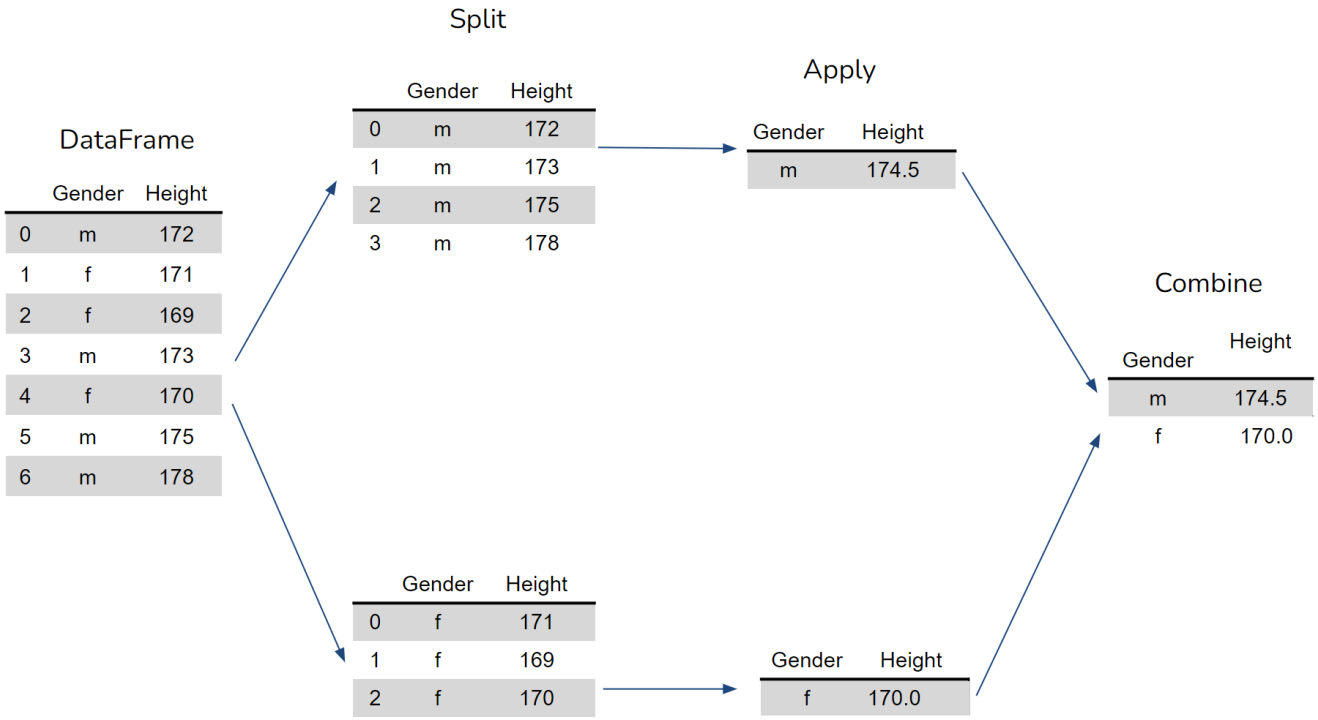
- *Los conjuntos de datos se fusionaron correctamente y las columnas se agregaron al final del conjunto de datos*

Podemos notar que se ha dado calificación múltiple a una sola película de diferentes usuarios.

Para descubrir las películas preferidas por los usuarios, primero debemos obtener las calificaciones promedio de cada película

Podemos realizar esto usando la función *group_by* de Pandas

Función agrupada
GroupBy de Pandas es una función poderosa y versátil en Python. Le permite dividir sus datos en grupos separados para realizar cálculos para un mejor análisis. *GroupBy* nos permite agrupar nuestros datos en función de diferentes características y obtener una idea más precisa sobre sus datos.



Ahora apliquemos la función groupby a nuestros conjuntos de datos:

```
avg_rating = df_merge.groupby(['movie title']).mean()[['rating']].reset_index() # 'reset_inc
avg_rating
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-90-b3c2a025bfda> in <cell line: 1>()
----> 1 avg_rating = df_merge.groupby(['movie title']).rating.mean()
[['rating']].reset_index() # 'reset_index' restablecerá el índice del dataframe a la
indexación predeterminada (0 al número de filas menos 1)
      2 avg_rating

                                 7 frames
/usr/local/lib/python3.10/dist-packages/pandas/core/indexes/base.py in
_raise_if_missing(self, key, indexer, axis_name)
   5936                 if use_interval_msg:
   5937                     key = list(key)
-> 5938                 raise KeyError(f"None of [{key}] are in the [{axis_name}]")
   5939
   5940             not_found = list(ensure_index(key)[missing_mask.nonzero()
[0]].unique())
```

```
# Agrupar por 'movie title' y calcular la media de las calificaciones ('rating')
avg_rating = df_merge.groupby(['movie title'])['rating'].mean()

# Mostrar el resultado
avg_rating
```

```
movie title
'Til There Was You                  2.333333
1-900                               2.600000
101 Dalmatians                      2.908257
12 Angry Men                        4.344000
187                                 3.024390
                                      ...
Young Guns                          3.207921
Young Guns II                       2.772727
Young Poisoner's Handbook, The      3.341463
Zeus and Roxanne                    2.166667
Á köldum klaka (Cold Fever)         3.000000
Name: rating, Length: 1657, dtype: float64
```

Esto muestra la calificación promedio de todas las películas.

Cambiemos el nombre de la columna `rating` a `Avg_rating` para una mejor comprensión.

```
avg_rating.rename(columns={'rating':'Avg_rating'},inplace=True)
avg_rating.head()
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-92-54f07c291ef8> in <cell line: 1>()
----> 1 avg_rating.rename(columns={'rating':'Avg_rating'},inplace=True)
      2 avg_rating.head()

TypeError: Series.rename() got an unexpected keyword argument 'columns'
```

También podemos ordenarlos en orden descendente o ascendente usando la función *sort_values*

Veamos las películas con las calificaciones promedio más altas

```
avg_rating.sort_values(ascending=False, by= 'Avg_rating')
```

```
avg_rating[avg_rating['Avg_rating']==5] #subdividirá los datos según la condición especifica
```

Perspectivas

- *Podemos observar películas como **Great Day in Harlem, A, Prefontaine**etc, son las películas mejor calificadas*.
- *Películas como**Shadow of Angels (Schatten der Engel), Power 98**, etc. son las menos valoradas*
- *Hay 10 películas que han sido calificadas como 5.0*.

Pero sabemos que hay películas que son calificadas más de una vez por diferentes usuarios y, por lo tanto, esto puede afectar la calificación general de la película.

Entonces, ¿podemos encontrar qué películas se clasifican la mayoría de las veces?

Vamos a crear una variable que contenga las películas con su conteo de calificaciones.

```
movie_count = df_merge.groupby(['movie title'])['rating'].count().reset_index()
movie_count
```

Podemos volver a cambiar el nombre de la columna `rating` a `Rating_counts` y ordenarlos en orden descendente para mostrar el conteo de calificaciones de mayor a menor.

```
movie_count.rename(columns={'rating':'Rating_counts'},inplace=True)
movie_count.head()
```

```
movie_count.sort_values(ascending=False, by= 'Rating_counts')
```

- *Podemos observar que **Star Wars** ha sido calificada la mayor cantidad de veces (**583 veces**) y hay pocas películas que tienen menos calificaciones como**Paris Was a Woman ,Á köldum klaka (Fever cold)**.*

Hay otras películas que en su mayoría son calificadas por los usuarios, así que revisemos las películas que tienen más de 100 calificaciones.

```
movie_100 = movie_count[movie_count['Rating_counts']>100] #this will subset the dataset 'mov
movie_100
```

Hay 334 películas calificadas más de 100 veces por los usuarios.

Entre estas películas, averigüemos **qué películas han sido mejor calificadas por los clientes** para que podamos saber qué tipo de películas son las preferidas por los usuarios en su mayoría.

```
avg_rating.head()
```

```
movie_100.head()
```

Podemos fusionar ambos conjuntos de datos y tener una vista combinada sobre la base de la calificación promedio y los conteos de calificación.

```
df_top= avg_rating.merge(movie_100, on = 'movie title', how = 'inner')
df_top
```

```
df_top.sort_values(ascending=False, by='Avg_rating').head(25) #mostrará las 25 mejores pelíc
```

Del mismo modo, también podemos ordenarlos según los conteos de calificaciones.

```
df_top.sort_values(ascending=False, by='Rating_counts').head(25) #mostrará las 25 mejores pe
```

Perspectivas

- *Hay 334 películas en total que han recibido calificaciones más de 100 veces*.
- *Según la calificación promedio,**Close Shave, A**tiene la calificación promedio máxima con un conteo de calificación de 112*.
- *Según el conteo de calificación,**Star Wars**ha recibido el número máximo de conteos con una calificación promedio de 4.35*.
- *Hay pocas películas que tienen menos calificación como Paris Was a Woman, Á köldum klaka (Cold Fever)*

Dado que hemos extraído las películas mejor calificadas, con la mayor cantidad de calificaciones otorgadas por los usuarios.

¿Podemos extraer alguna relación entre los detalles demográficos de los usuarios y las calificaciones de las películas?

Todos los conjuntos de datos se pueden utilizar para identificar la demografía de los usuarios. Por lo tanto, fusionaremos todos los conjuntos de datos y realizaremos el análisis.

<*><>

Dado que ya hemos fusionado los conjuntos de datos `rating` y `movie` en una variable `df_merge`, podemos fusionar el conjunto de datos `user` con este conjunto de datos.

```
# Merging el conjunto de datos de usuario, con el conjunto de datos de movie y ratings (alre
df_merge_all = df_merge.merge(user, on = 'user id', how = 'inner')
```

```
df_merge_all.head()
```

Veamos varios atributos de los usuarios para conocer las preferencias de las películas y sus