laSalle ENG
Universitat Ramon Llull

Máster en Big Data

Asignatura: Casos de analítica
Fecha: 07/05/2024

Apellidos: Ripoll Batlló Nombre: Albert Login: a.ripoll Curso: **2023/24**

Caso 3: Book analysis

Este tercer caso gira entorno al mundo de la lectura. Específicamente, se quiere hacer un análisis de un dataset de libros para, en última instancia, crear un modelo que recomiende nuevas lecturas. Para llevarlo a cabo, partiremos de dos conjuntos de datos:

1- *Book-Crossing dataset*, que puede descargarse del siguiente enlace: https://www.kaggle.com/datasets/somnambwl/bookcrossing-dataset.
2- *LSGoodreads dataset:* que puede descargarse del eStudy.

Con estos datos, se os propone que apliquéis técnicas de estadística, analítica, minería de datos y visualización para responder a las siguientes preguntas. No hay restricciones acerca de las técnicas ni tecnologías a utilizar siempre y cuando los resultados sean reproducibles y estén debidamente justificados. Explicitad y detallad todos los pasos hechos para responder a cada pregunta y las conclusiones que podáis derivar de ellas.

PRIMERA PARTE: ANÁLISIS CUANTITATIVO.

1.1 Primer examen preliminar del de los datos.

• ¿En qué formato está el dataset? El Book-Crossing dataset de Kaggle descargado consta de 3 archivos CSV (Coma Separated Values) que son: Books.csv , Ratings.csv , Users.csv

Al abrir en Excel los archivos vemos que los formatos del los archivos. Para el caso de:

- Books.csv es de 271380×5 siendo la primera fila el encabezado de los 271380 libros listados.
- Ratings.csv es de 1048576×3.
- Users.csv es de 278860×2.

• ¿Cómo podemos leerlo correctamente?

1- Subiendo los archivos en Drive
2- Montando Google Collab en Drive

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

3- Definiendo la ruta base para cada compañero de trabajo comentándolo para facilitar correr el código en distintos disposivos.

```
ruta_base = '/content/drive/MyDrive/BigData/Casos de analítica/Caso 3. Book Analysis/'
```

4- Usar read_csv de Pandas separando los elementos por ; y asociando cada archivo a un dataframe distinto.

```
import pandas as pd
df_books = pd.read_csv(ruta_base + 'Books.csv', delimiter=';')
df_ratings = pd.read_csv(ruta_base + 'Ratings.csv', delimiter=';')
df_users = pd.read_csv(ruta_base + 'Users.csv', delimiter=';')
```

• ¿Qué campos hay en cada fichero del *dataset*?

- En Books.csv son ISBN, Title, Author, Year, Publisher.

- En Ratings.csv son User-ID, ISBN, Rating.
- En Users.csv son User-ID, Age

```
print(df_books.columns)
print(df_ratings.columns)
print(df_users.columns)

Index(['ISBN;Title;Author;Year;Publisher'], dtype='object')
Index(['User-ID;ISBN;Rating'], dtype='object')
Index(['User-ID;Age'], dtype='object')
```

• ¿Cuál es su significado?

Campos	Significado
ISBN	Es el International Standard Book Number, un identificador único para cada libro.
Title	Es el título del libro. Hay 242.154 distintos. `df_books['Title'].nunique()`
Author	Es el autor del libro. Hay 102.029 distintos. `df_books['Author'].nunique()`
Year	Es el año de publicación del libro.
Publisher	Es la editorial publicadora. Hay 16.910 distintas. `df_books['Publisher'].nunique()`
User-ID	Es el identificador de usuario. Parece que hay 278.859 usuarios distintos. `df_users['User-ID'].nunique()`
Age	Son las edades de los usuarios.
Rating	Calificación de los libros por parte de los usuarios que va de 0 a 10. `df_ratings['Rating'].value_counts()`

• ¿Existen valores aparentemente incorrectos? Sí.

○ En Year.

No hay valores NaN.

No hay valores no enteros.

Hay 4619 valores que tienen 0 como año de publicación. Serían valores nulos.

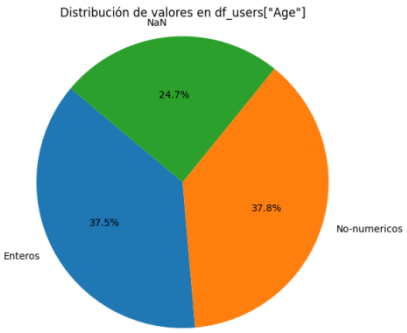
Hay 2 valores que están entre el año 1 y el año 1455 (año en el cual se publicó el primer libro impreso, la Biblia)

Hay 7308 valores que están por debajo del año 1970 (año en el cual se empezó a listar los libros con ISBN)

Hay 12 valores que están por encima del año 2024.

Debido a ese análisis se considera que hay 4.633 valores de edades de libros incorrectas (4621 valores de year menor de 1455 más los 12 valores de year mayor a 2024). Esto representa un 1,7% de los datos (4.633 valores de 271.378 de que se disponen). Estos datos no se eliminan puesto que el hecho que la edad de publicación del libro no se sepa o sea incorrecta no implica que el libro no exista, pero si que implica que no podemos considerar este dato para el análisis de este 1,7% de los libros.

○ En Age.



Hay 110.232 valores vacíos 'NaN'.

Hay 168.627 valores no que no son ni 'NaN' ni son numéricos.

En concreto hay 167.151 valores que son tipo string.

En resumen, solo hay 167.151 valores numéricos.

Todos los valores numéricos en Age son enteros.

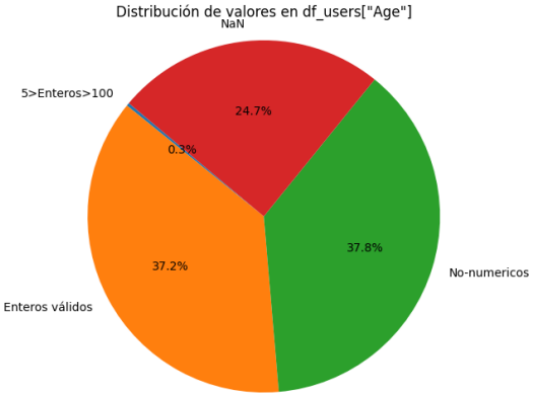
Age no solo contiene la edad de los individuos sino que también es un volcado de países de orígenes de los usuarios en formato string.

Hay 364 individuos con una edad superior a 100 años.

Hay 95 individuos con una edad superior a 111 años (edad máxima de un humano según Guiness World Records)

Hay 412 individuos con una edad igual a 0 años.

Hay 458 individuos con una edad entre 0 y 5 años, extremos excluidos (edad a partir de la cual comúnmente se aprende a leer).



Debido a ese análisis se considera no tener en cuenta los datos de Age para el análisis ya que hay muy pocos datos que puedan realmente representar la edad como enteros. En el diagrama de tarta podemos ver claramente que solo un 37% de los datos contienen edades útiles.

○ <u>En User-ID</u>.

No hay valores NaN en users ni en rating sobre el User-ID.

Vemos que hay 6230 valores de User-ID que están presentes en el dataframe de ratings a los que no les corresponde ningún usuario del dataframe de users. Esto podría ser debido a que el dataframe de users no esté actualizado o que sean usuarios erróneos. Dado que del dataframe de users tampoco sacamos muchas características de esos ya que hemos descartado usar 'Age', consideraremos a esos 6230 votaciones como votaciones de usuarios válidos para nuestro análisis.

○ <u>En ISBN</u>.

No hay valores NaN.

Un ISBN es una expresión de 10 o 13 dígitos. En un ISBN de 10 dígitos, hay 9 dígitos y una terminación con X. En un ISBN de 13 dígitos no hay dígito de verificación. Así que podemos ver si la expresión regular a partir de su inicio (^) tiene 9 dígitos (\d{9}) y sigue con una x minúscula o mayúscula ([\dXx]) antes de terminar ($) o (|) si por el contrario a partir de su inicio (^) tiene 13 dígitos (\d{13}) antes de terminar ($). O sea: `isbn_pattern = r'^\d{9}[\dXx]$|^\d{13}$'`

Con este análisis se detectan 117 libros que tienen el ISBN incorrecto. No son datos que se descarten, pero si que tener en cuenta que hay algún valor erróneo en los datos.

○ <u>En rating</u>.

No se detectan valores incorrectos ya que no hay NaN y todos los valores son 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 o 10.

1.2 Utiliza técnicas de webscrapping para enriquecer el dataset de *Goodreads,* ya sea con libros nuevos o añadiendo características.

• ¿Cuántos libros nuevos o características has podido añadir?

- Se han añadido 1478 libros nuevos al dataset.
- El dataset original tenía las siguiente características: ISBN, Title, Author, Year, Publisher + User-ID, ISBN, Rating + Age
- Los datos añadidos tienen las siguientes características interesantes, así que se han añadido bastantes: book_title, book_series, book_language, published (año de publicación), author, num_pages, num_ratings, average_rating, rating_distribution. En concreto todas las características añadidas son: ['book_id_title', 'book_id', 'cover_image_uri', 'book_title', 'book_series', 'book_settings', 'book_characters', 'book_language','year_first_published', 'authorlink', 'author', 'num_pages', 'format', 'genres', 'people_curr_read', 'peop_want_to_read', 'num_ratings', 'num_reviews', 'average_rating', 'rating_distribution', 'awards']

• ¿Qué criterio has seguido para escoger los libros que querías añadir al dataset? Detalla todo el proceso y las elecciones de diseño.

Elegimos los libros de la librería "Hottest Dirty Talk" con el criterio de elegir y añadir libros que sean de temática erótica. Los cambios hechos son los siguientes:

1) Del código `#GET BOOKS FROM LIST` cambiamos la línea 46 por el nombre de la web adecuado. Cambiamos la terminación "`201106.Best_books_of_May_2024`" por la terminación "`42779.Hottest_Dirty_Talk_`" que es la correspondiente al url de esta lista de libros de la cual cogerá los libros .
2) Del código `#GET BOOKS` cambiamos la línea 500 el nombre del .txt a generar. De "`my_list_of_books.txt`" por "`good_list_books.txt`". Aquí genera un .txt que contiene las terminaciones de todos los libros que hay en la librería de la lista de libros.
3) Y cambiamos la siguiente línea de "`classic_book_metadata`" a "`eroticaBookList`" para el nombre del directorio que se generará.

El proceso del programa es el siguiente:

1) Genera un .txt "good_list_books.txt" que contiene las terminaciones de los libros de la librería pedida "Hottest_Dirty_Talk_"
2) Concatena el enlace base de la página con las terminaciones de los libros que hay en el .txt.
3) Hace el scraping en estos sitios. Abre y cierra las páginas listadas y va cogiendo los datos de cada página (libro). La información se guarda en un json. Todo se guarda en el directorio que le hemos dicho, el "eroticaBookList"
4) El programa guarda también all_books.json que lo borramos y all_books.csv que es el que usamos para ver todos los datos que hay.

Finalmente unimos el dataset de libros originales con el webscrapeado. De esta forma podemos seguir nuestro análisis con 1478 libros nuevos añadidos a nuestra base de datos. Base de datos nueva que se usa en el punto 4.2. Para esa unión lo que se hace es comparar las columnas de un data frame y de otro para ver las faltantes.

```python
columns_books = set(df_books.columns)
columns_newbooks = set(df_newbooks.columns)

# Comprobar si todas las columnas de df_books están en df_newbooks
if columns_books.issubset(columns_newbooks):
    print("Todas las columnas de df_books están en df_newbooks.")
else:
    print("No todas las columnas de df_books están en df_newbooks.")
    missing_columns = columns_books - columns_newbooks
    print("Columnas ausentes en df_newbooks:")
    for column in missing_columns:
        print(column)
```

```
No todas las columnas de df_books están en df_newbooks.
Columnas ausentes en df_newbooks:
Year
ISBN
Author
Publisher
Title
```

Se modifican de los libros nuevos.

```
df_newbooks.rename(columns={'year_first_published': 'Year', 'book_title': 'Title',
'author': 'Author', 'book_id': 'ISBN'}, inplace=True))
```

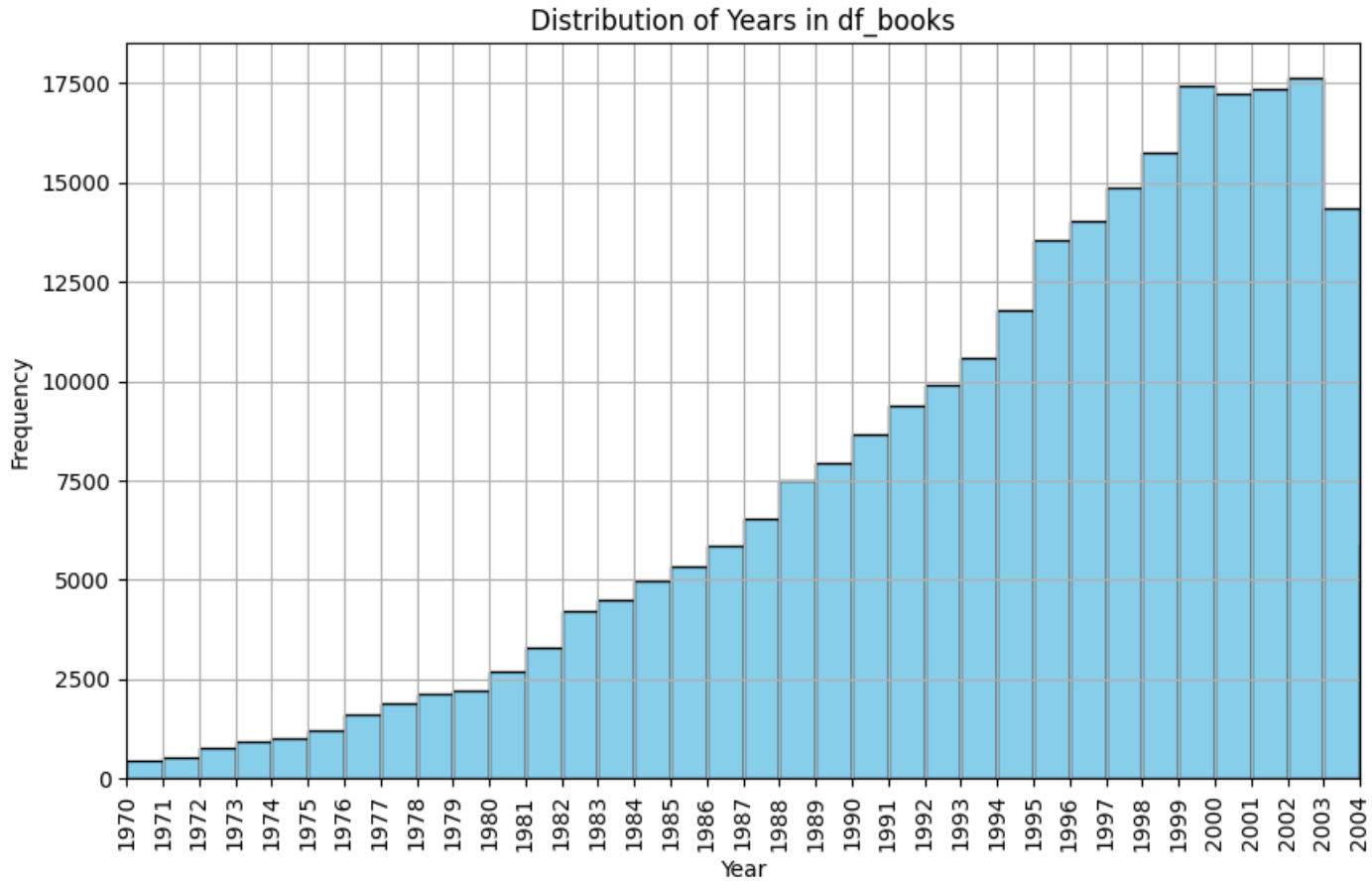
Y se contatenan los data frames.

```
df_allbooks = pd.concat([df_books, df_newbooks], ignore_index=True)
df_allbooks.drop(columns=set(df_newbooks.columns) - set(df_books.columns),
inplace=True)
```

SEGUNDA PARTE: ANÁLISIS CUALITATIVO.

2.1

- ¿Cuál fue el año en el que se publicaron más libros? Muéstralo en un gráfico. En 2002



Para responder a esta pregunta simplemente se ha usado `df_books['Year'].mode()`.

Para encontrar el gráfico más representativo primero se ha graficado un histograma filtrando los años entre 1970 y 2024 ya que esta es la edad a partir de la cual se empezó a registrar libros con ISBN y 2024 es la edad actual. El tamaño de la caja, se ha determinado de 30 para ver una forma inicial → `plt.hist(df_books[(df_books['Year'] > 1970) & (df_books['Year'] < 2024)]['Year'], bins=30)`.

Al ver la forma inicial del gráfico se ha observado que después del 2004 las reseñas son muy escasas y no aportan valor visual. Entonces se ha acotado más el gráfico hasta 2004 con tamaño de bin de 1 →
`plt.hist(df_books[(df_books['Year'] >= 1970) & (df_books['Year'] <= 2004)]['Year'], bins=range(1970, 2006, 1))`

- ¿Y el autor más prolífico? Agatha Christie.

Para responder a esta pregunta simplemente hemos tenido que encontrar la moda de la columna de autor `df_books['Author'].mode()`.

- ¿Cuántos libros suyos hay en el dataset? 632

Para responder a esta pregunta simplemente hemos contado los valores donde el autor es Agatha Cristie `(df_books['Author']=='Agatha Christie').value_counts()`.

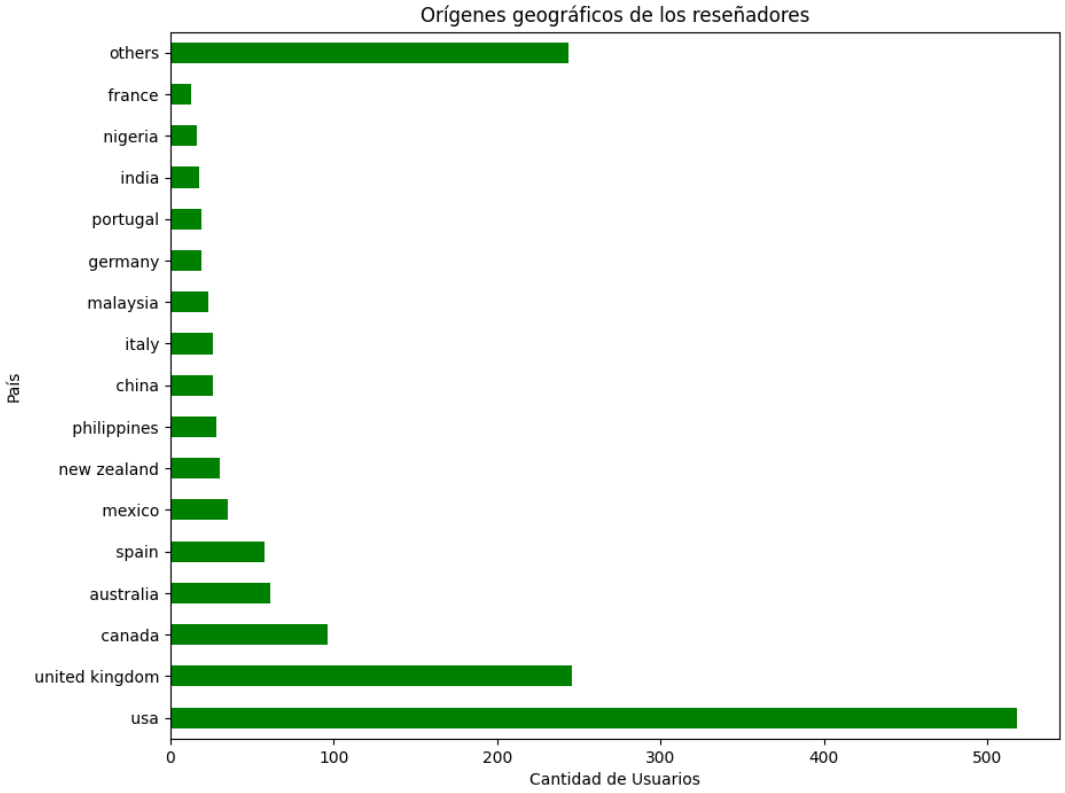
```
Author
False    270747
True        632
Name: count, dtype: int64
```

Queríamos representar esta información gráficamente pero el gráfico tardaba mucho en cargar. Hemos investigado porqué y la razón parece ser en que hay muchos autores diferentes! Hay 102.029 autores distintos.

```
1 df_books['Author'].nunique()
```

```
102029
```

2.2 Analiza los orígenes geográficos y la edad de los reseñadores.



Para hacer este gráfico primero hemos filtrado los valores tipo string de la columna Age del dataframe users ya que esta parte es la que contiene los orígenes geográficos de los reseñadores. Hemos tenido que hace un poco de ingenio en programar esta parte: 1) Intentar convertir la columna Age a numérica y en caso que no pueda que añada NaN. 2) Filtrar los usuarios donde el anterior comando ha dado NaN. 3) Obtener dataframe llamado geography con los Users y su

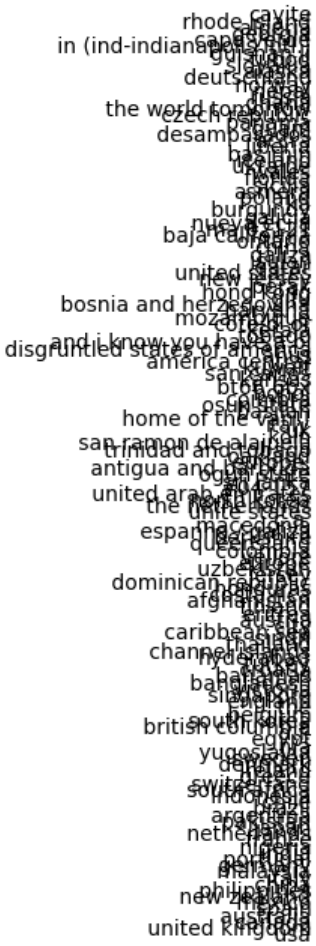
Country. 4) Eliminar los valores NaN que quedaron. 5) Contar los países que hay.

```
# 1) Intentar convertir la columna 'Age' a números y obtener los valores NaN

age_numeric = pd.to_numeric(df_users['Age'], errors='coerce')
# 2) Filtrar los orígenes geográficos de los reseñadores (los datos que son strings)
geography = df_users[age_numeric.isna()]
# 3) Cambiar el nombre de la columna 'Age' a 'Country'
geography.rename(columns={'Age': 'Country'}, inplace=True)
 # 4)
geography['Country'].isna().sum()
```

110232

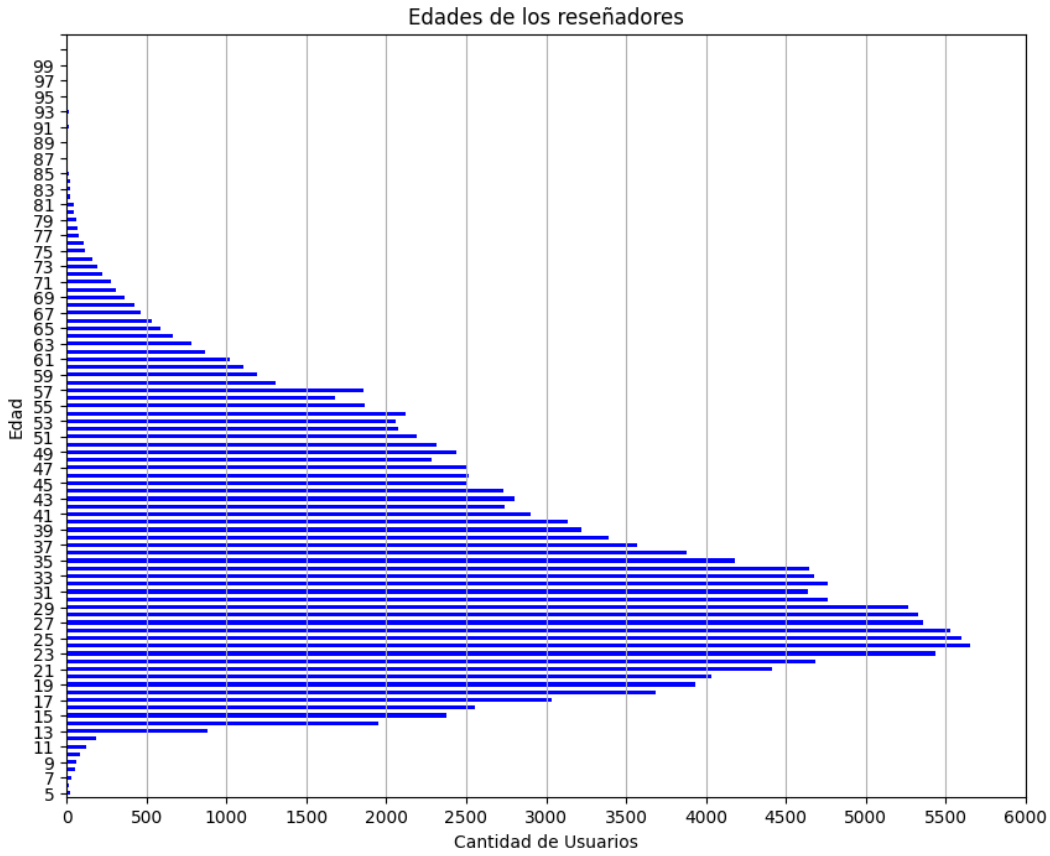
```
geography.dropna(subset=['Country'], inplace=True)
# 5) Contar los valores únicos en la columna 'Country'
geography['Country'].value_counts()
```



En el dataframe geography hay 147 países distintos. Si se representan todos los datos, la lectura se hace imposible. Además esta columna parece estar recopilada manualmente y contiene datos incorrectos o distintos como se puede ver en la figura del margen derecho de esta página. Por ejemplo aparecen datos incorrectos como “home of the vann” , “and i know you have” “the world tomorrow” o aparecen datos que se refieren a países distintos escritos de formas distintas “usa”, “united states”, “united states of america”, “new york”. Es por eso que se opta por representar los países con un mínimo de 10 usuarios y añadir una barra de “others”

```
country_counts = geography['Country'].value_counts() # País - Cantidad de usuarios de ese país
filtered_countries = country_counts[country_counts > 10] # Filtramos los países con como mínimo más de 10 usuarios
other_count = country_counts[country_counts < 10].sum() # Sumamos los países que tienen menos de 10 usuarios
final_data = pd.concat([filtered_countries, pd.Series({'others': other_count})]) # Unimos los datos de Others

final_data.plot(kind='barh', figsize=(10, 8), color='green')
plt.title('Orígenes geográficos de los reseñadores')
plt.xlabel('Cantidad de Usuarios')
plt.ylabel('País')
plt.show()
```



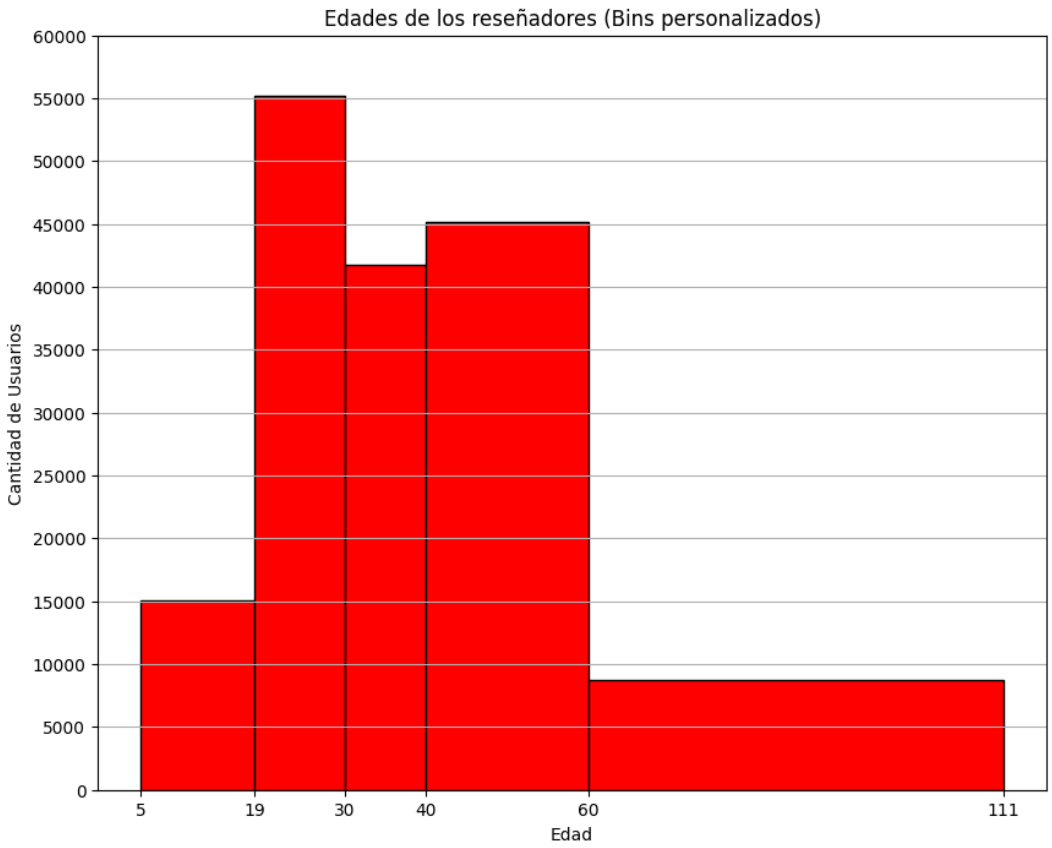
Para analizar las edades de los reseñadores, primero hemos tenido que filtrar del dataframe de usuarios las columnas numéricas de “Age” ya que también hay valores string de localizaciones volcadas en esa columna. En segundo lugar hemos filtrado la edad de los reseñadores entre 5 y 100 ya que recordemos que habían bastantes (412) valores de edad 0 y lo hemos pasado a formato entero.

```
# Convertir la columna 'Age' a números
df_users['Age'] = pd.to_numeric(df_users['Age'], errors='coerce')

# Filtrar la edad de los reseñadores (enteros entre 5 y 100)
age = df_users[(df_users['Age'].notnull()) & (df_users['Age'] >= 5) &
(df_users['Age'] <= 100)]
age['Age'] = age['Age'].astype(int)
age['Age'].value_counts()
```

Hemos comprobado que no haya ningún valor NaN y hemos graficado las edades con la máxima precisión, edad por edad, sin hacer agrupaciones por histograma. Los ticks del eje vertical se han fijado a 2 ya que con 1 quedaban demasiado juntos y los del eje horizontal se han puesto de 500 usuarios en 500 marcando las grids verticales para mejorar la lectura del gráfico.

A parte, se ha repetido el gráfico pero con cajas personalizadas para menores de edad, jóvenes (18-30), trentañeros (30-40), cuarentones+cinquentones (40-60) y mayores de 60 (60-111) para proporcionar una lectura más buena a simple vista aunque menos precisa.



TERCERA PARTE: ANÁLISIS PREDICTIVO.

3.1 Una editorial nos ha contactado para ver qué parámetros debería tener un libro para que fuera existoso. A partir del dataset y su análisis, orienta a la editorial sobre qué parámetros deben seguir a la hora de publicar un nuevo libro.

A partir de los datos que tenemos, se entiende que un libro es exitoso como mayor *rating* tenga ese libro. Los parámetros que se pueden considerar que influyen en que el *rating* sea más alto o bajo son:

• Edad → Edad a la que va dirigida el libro. Publicar libros dirigidos para los grupos de edades que puntúan mejor el libro.
• Localización → País al que va dirigido el libro. Publicar libros en los países donde se puntúa mejor el libro.
• Autor → Fomentar la publicación de libros de determinados autores que tienen puntuaciones más altas.

Para realizar el análisis tenemos que crear un dataset con las tablas books, ratings y users y que tenga en cuenta los valores incorrectos de determinados campos vistos en apartados anteriores. Sobre todo de la columna Age.

Primero uniremos la tabla Ratings con la tabla Users por User-ID. Debido a los análisis del punto 2.2 (código en .ipynb) ya tenemos como dataframes a age y geography con los nombres de los usuarios y su edad y país.

```
1 age.tail(2)
```

	User-ID	Age
278853	278853	17
278855	278855	50

```
1 geography.tail(2)
```

	User-ID	Country
278215	278215	new zealand
278809	278809	united kingdom

• **Para Geography:** De la columna ‘Age’ se extraen los valores tipo string. Se obtienen 1 476 usuarios con su país. Se unen los datos de usuario con las votaciones (ratings). Sobre los valores únicos de los países realizamos un mapping para pasar a países que realmente son países con nombre único y filtramos aquellos con más de 2 votaciones.

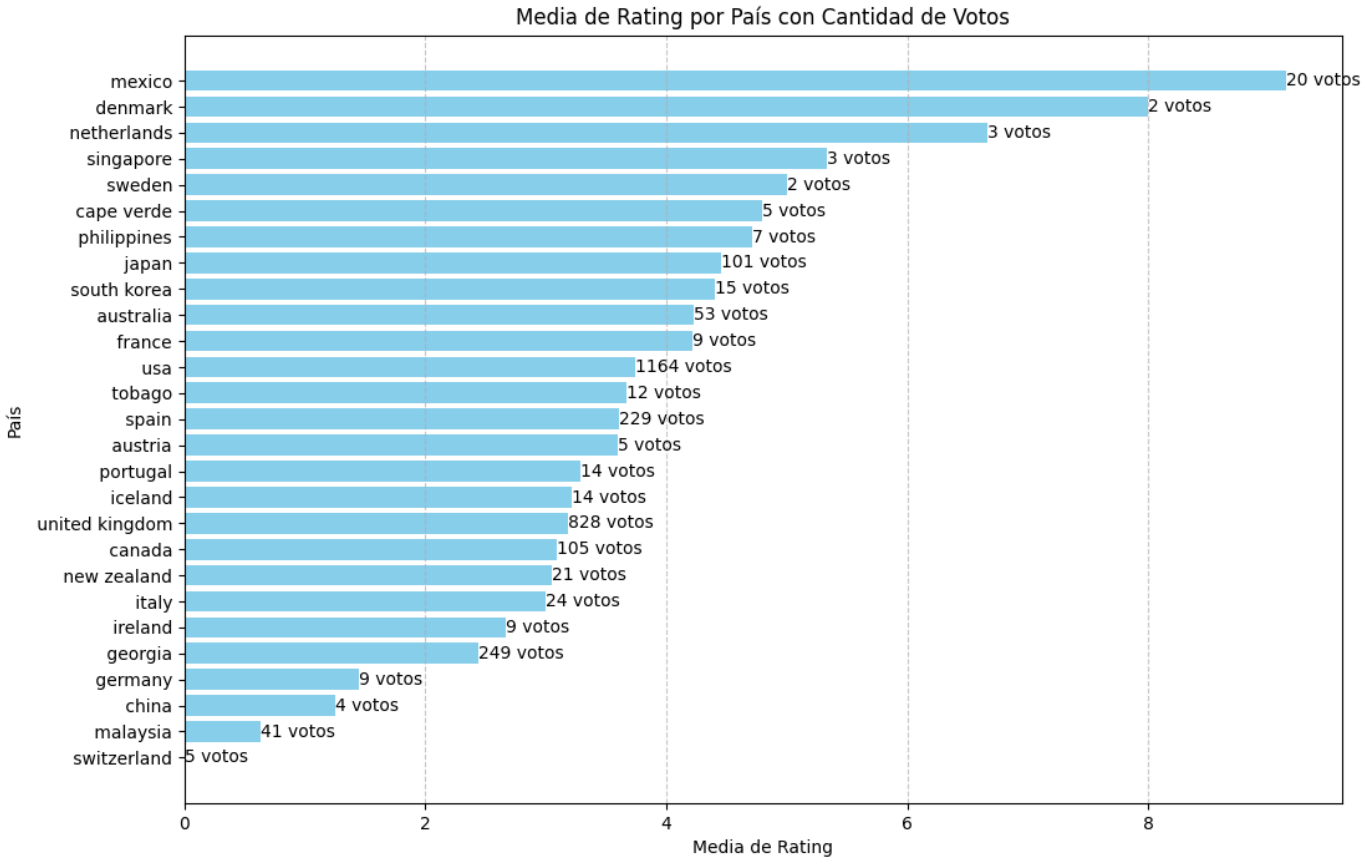
Country	
usa	1139
united kingdom	804
georgia	249
spain	224
japan	99
canada	89
australia	53
malaysia	41
italy	24
kansas	21
new zealand	21
mexico	20
channel islands	18
south korea	15
british columbia	15
portugal	14
iceland	14
tobago	12
france	9
ireland	9
germany	8
philippines	7
switzerland	5
cape verde	5
austria	5
england	5

china	4
espanha / galiza	3
singapore	3
gifu	2
denmark	2
sweden	2
netherlands	2
galiza	2
rhode island	1
finland	1
disgruntled states of america	1
afghanistan	1
deutschland	1
greece	1
india	1
jersey	1
nigeria	1
united states	1
the netherlands	1
c.a.	1
south africa	1
honduras	1
chile	1
guam	1
the	1
the world tomorrow	1
guishan li	1
ontario	1
d.f.	1

```
mapping = {
    'british columbia': 'canada',
    'channel islands': 'united kingdom',
    'jersey': 'united kingdom',
    'england': 'united kingdom',
    'united states': 'usa',
    'kansas': 'usa',
    'espanha / galiza': 'spain',
    'galiza': 'spain',
    'rhode island': 'usa',
    'disgruntled states of america': 'usa',
    'deutschland': 'germany',
    'the netherlands': 'netherlands',
    'ontario': 'canada',
    'guam': 'usa',
    'gifu': 'japan'}
```

Country	
usa	1164
united kingdom	828
georgia	249
spain	229
canada	105
japan	101
australia	53
malaysia	41
italy	24
new zealand	21
mexico	20
south korea	15
iceland	14
portugal	14
tobago	12
germany	9
ireland	9
france	9
philippines	7
cape verde	5
switzerland	5
austria	5
china	4
singapore	3
netherlands	3
denmark	2
sweden	2

Seguidamente representamos los datos para entender qué países valoran mejor los libros.



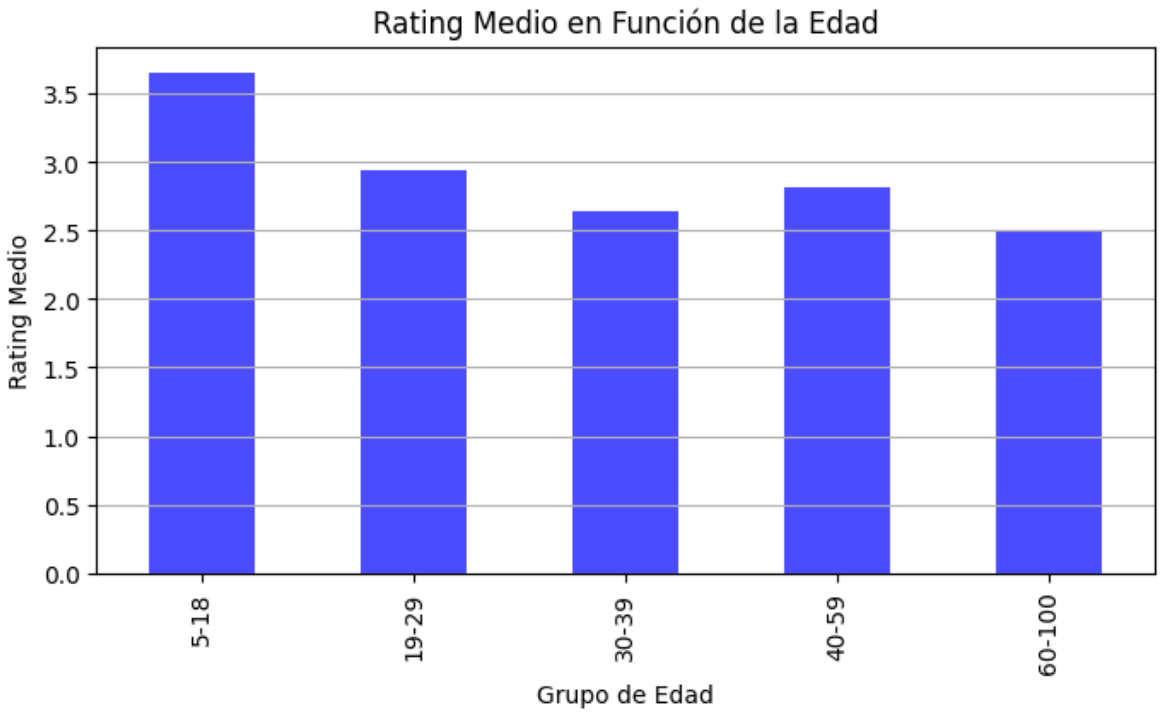
Observamos que países como México, Dinamarca y Holanda tienden a valorar mejor de media los libros por lo que podría pensarse focalizar más ventas en estos países. Sin embargo, la cantidad de datos que se tienen por países es muy poca ya que de pocos usuarios se tiene la información de su país. Si se quiere sacar conclusiones más fiables se debería recopilar ese dato más frecuentemente y con consistencia en los nombres de los países.

• **Para Age** se convierte la columna a número entero, se filtran entre 5 y 100 años y se quitan los NaN (valores no convertibles a enteros). Se consiguen 278 855 usuarios con edad.

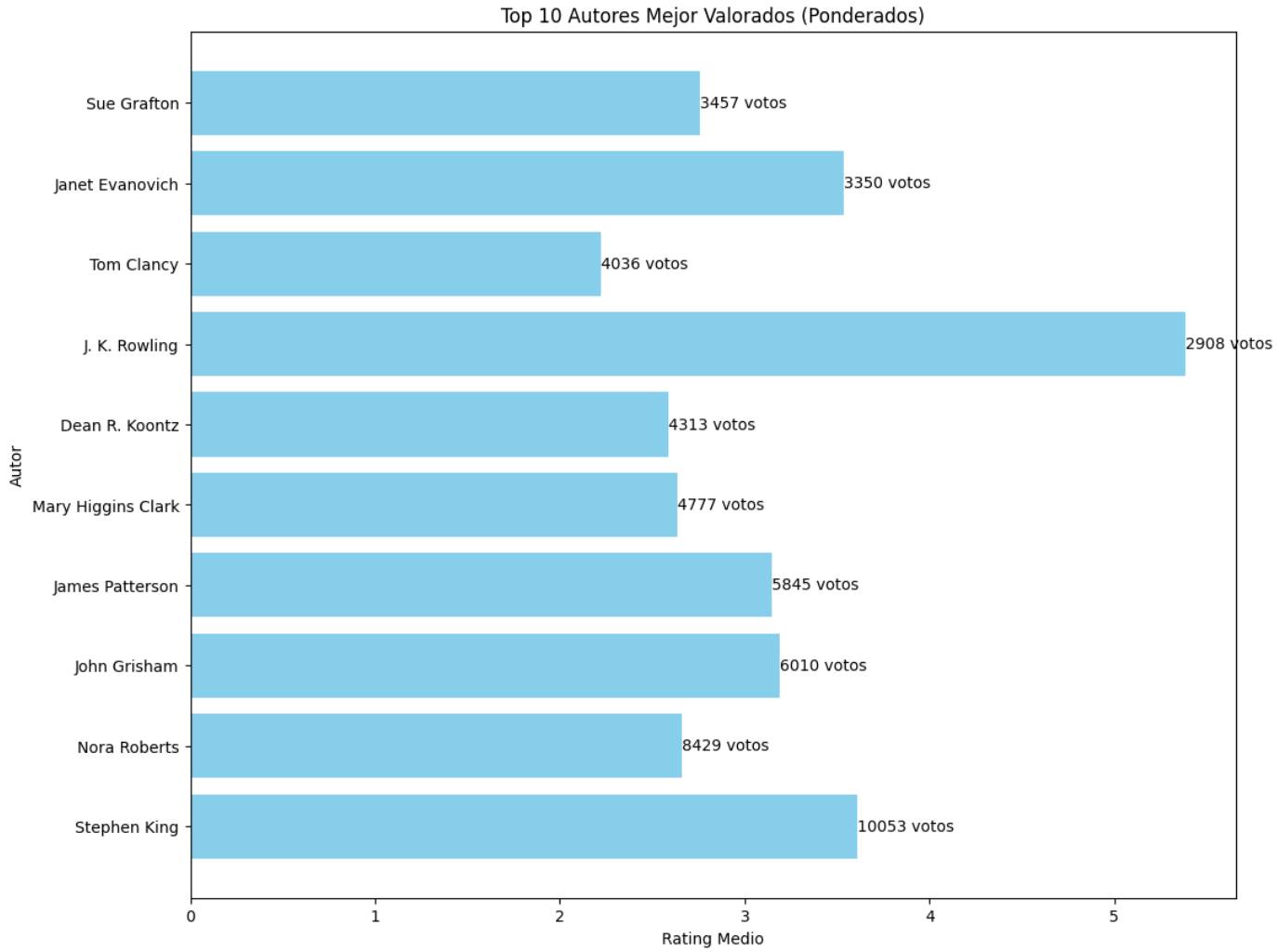
Entonces se hace un left join con los ratings (los datos de Age se añaden al dataframe de Ratings). Esa unión genera 364 147 valores nulos y 785 633 valores no nulos. Se descarta ese tercio nulo.

Con los datos restantes se mira la dependencia entre el rating y la edad. Se encuentra que no hay correlación con R = -0,01 y un diagrama de dispersión que claramente muestra independencia entre las variables.

Al analizar el rating medio según el rango de edad, se observa que los jóvenes menores de edad puntúan una media de 0,5 - 1 punto por encima de los otros grupos de edad. Así que quizás es recomendable publicar para ese grupo de edad.



• **Para Autor:** Se unen los dataframes de libros y de ratings por código ISBN. Se consideran los autores mejor valorados según su rating y la cantidad de puntuaciones que tienen sus libros. Usando una ponderación para esa puntuación. Se ha provado diferentes ponderaciones hasta dar con la óptima `weight_rating = 0.9979; weight_count = 0.0021` ya que una décima más ponderando el rating y nos aparece un autor con 13 valoraciones que sería no significativo. También esos valores están en relación a que count es del orden de 2000 y rating el orden de 5. Por lo que 2000/5 = 0,0025 similar a la ponderación dada `weight_rating/weight_count =` 0,021/0,9979.



Debido a este análisis la editorial podría considerar fomentar la publicación de libros de estos autores.

3.2 Diseña un modelo que, a partir de un libro de entrada, te recomiende una nueva lectura. Puedes utilizar o bien el dataset proporcionado o bien enriquecerlo (por ejemplo, utilizando técnicas de *web scrapping* 😉 o añadiendo más atributos a los libros actuales). Respecto a este sistema, a modo de ejemplo, explica las recomendaciones que proporcionaría el modelo si entráramos los siguientes libros:

- *A Court of Thornes and Roses de Sarah J. Maas*
- *Hamlet de William Shakespeare*
- *La apología de Sócrates de Platón*

Se usa el dataset `df_newbooks` que contiene los libros del web scrapping hecho de los libros de erótica (unos 1400 libros) con el dataset `df_goodreads` descargado de la práctica. Se unen para tener 7134 libros, se cambian nombres de algunas columnas, se extrae las cantidades de 5,4,3,2,1 estrellas dentro del diccionario de `rating_distribution` y se saca las cantidades de premios `num_awards` a partir de la lista de premios `awards`.

A su vez se hace todo un trabajo de cambio de formato ya que hay números que aparecen como 42.7k cuando quieren simbolizar 42700 o 1m cuando simbolizan 1000000 y hay que tener cuidado con las comas y puntos así como los distintos formateos de los datos y los NaN. Por eso se aplica la función `standard_df` que asegura el correcto formato de los datos. Este paso es muy importante antes de crear los transformadores y hacer un modelo de recomendación KNN.

Seguidamente se definen las características de los libros que servirán para predecir `['book_series', 'book_language', 'Year', 'Author', 'num_pages', 'format', 'genres', 'people_curr_read', 'peop_want_to_read', 'num_ratings', 'num_reviews', 'average_rating', '1 Star', '2 Stars', '3 Stars', '4 Stars', '5 Stars', 'num_awards']` que se clasifican en numéricas y categóricas. Entonces la función `recommend_books` ordena el listado de libros según la distancia del KNN.

Para sacar las recomendaciones a partir de una query se hace lo siguiente:

• Paso 1: Con el nombre del libro en la query, se va a la página de goodreads y se busca el libro. Al encontrarlo, de la URL se saca el ID y se guarda en un .txt en el path output que se le determine:

```
query = "A Court of Thornes and Roses"

html_content = search_book(query)
print(html_content)
base_path = "/content/drive/MyDrive/BigData/Casos de analítica/Caso 3. Book Analysis/"
with open(base_path + 'book.txt', 'w') as f:
    f.write(html_content)

args = ["python", base_path + "get_books.py", "--book_ids_path", base_path + "book.txt", "--output_directory_path", base_path + "feat_book"]
```

• Paso 2: Después esa tener el .txt, en el mismo código de Python, se usa un nuevo archivo de Python. El “get_books”. Se ejecuta como si fuese en local pero lo hace el mismo código de Python. El “get_books” se ejecuta diciéndole que lea el mismo archivo book.txt que pusimos previamente como nombre al .txt que tiene el ID del libro encontrado en el paso 1. Con ese ID se hace el scrapping con los códigos proporcionados por la profesora y se genera el json de ese nuevo libro.

```
subprocess.run(args)
file_path = base_path + "feat_book/all_books.json"
delete_files(file_path)
```

• Paso 3: Se coge el json de las características del libro. Se convierten esas características del libro query a un formato Data Frame con el mismo formato que se usó para entrenar. Ese Data Frame se usa como entrada para el modelo recomendador, y basado en las características de entrada es como se recomienda el siguiente libro.

```
book_features = get_book_info()
book_index = 0
recommendations = recommend_books(pipeline, book_features, book_index)

print(recommendations[0])
```

• A Court of Thornes and Roses de Sarah J. Maas → Recomienda The Seven Husbands of Evelyn Hugo

• Hamlet de William Shakespeare→ Recomienda Macbeth

• La apología de Sócrates de Platón →Recomienda Gorgias

Consideraciones

Deben contestarse todas las preguntas para considerar el trabajo como entregado. Podéis realizarlo en grupos de máximo 2 personas.

Entrega

Para que el trabajo conste como entregado, debéis subir al eStudy ambos miembros del grupo tanto un informe del trabajo como el código utilizado. Los datasets que se hayan modificado también deberán ser entregados. La fecha de entrega es el 14 de junio de 2024.

Anexo

Para hacer este caso, es necesario hacer webscrapping de Goodreads. Para ello, se os ha facilitado un seguido de scripts de Python que os pueden ayudar con la tarea. Se recomienda utilizar el software PyCharm para correr los códigos, aunque luego en análisis del dataset se haga en Google Colab.

Los scripts más interesantes son los siguientes:

- Script **get_books_from_list.py**: en este script, podéis pasarle un enlace que contenga una lista de libros de Goodreads (por ejemplo, esta: https://www.goodreads.com/list/show/8599.Magical_and_Mystical_Creatures_Titles_ONLY) y se descargará el título y el identificador de Goodreads para guardarlo en una lista en formato .txt. Esta lista nos servirá para hacer obtener la información de los libros utilizando el siguiente script.

- Script **get_books.py**: la función de este script es descargarse la información individual de cada libro gracias a una lista obtenida por el script get_books_from_list.py. De este modo, abrirá la página individual de cada libro (por ejemplo, esta: https://www.goodreads.com/book/show/5907.The_Hobbit) y se guardará información en referencia al título, el autor, las valoraciones, etc. Para cada libro analizado, se guardará un fichero .json con la información encontrada. Además, cuando se hayan terminado de leer todos los libros de la lista, se creará un fichero .CSV con la información de todos los libros.