```
1 import pandas as pd
2 import seaborn as sns
3 import matplotlib.pyplot as plt
4 df=pd.read_csv("netflix_titles.csv")
5 df.head()
```

	show_id	type	title	director	cast	country	date_added	release_
0	81145628	Movie	Norm of the North: King Sized Adventure	Richard Finn, Tim Maltby	Alan Marriott, Andrew Toth, Brian Dobson, Cole...	United States, India, South Korea, China	September 9, 2019	
1	80117401	Movie	Jandino: Whatever it Takes	NaN	Jandino Asporaat	United Kingdom	September 9, 2016	
2	70234439	TV Show	Transformers Prime	NaN	Peter Cullen, Sumalee Montano, Frank Welker, J...	United States	September 8, 2018	
3	80058654	TV Show	Transformers: Robots in Disguise	NaN	Will Friedle, Darren Criss, Constance Zimmer, ...	United States	September 8, 2018	
4	80125979	Movie	#realityhigh	Fernando Lebrija	Nesta Cooper, Kate Walsh, John Michael Higgins...	United States	September 8, 2017	

Próximos pasos: Generar código con df | Ver gráficos recomendados | New interactive sheet

```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 6234 entries, 0 to 6233
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   show_id       6234 non-null   int64 
 1   type          6234 non-null   object
 2   title         6234 non-null   object
 3   director      4265 non-null   object
 4   cast          5664 non-null   object
 5   country       5758 non-null   object
 6   date_added    6223 non-null   object
 7   release_year  6234 non-null   int64 
 8   rating        6224 non-null   object
 9   duration      6234 non-null   object
 10  listed_in     6234 non-null   object
 11  description   6234 non-null   object
dtypes: int64(2), object(10)
memory usage: 584.6+ KB
```

```
print(df.isnull().sum())
```

```
show_id            0
type               0
title              0
director        1969
cast             570
country          476
date_added        11
release_year       0
rating            10
duration           0
listed_in          0
description        0
dtype: int64
```

```
print(df.duplicated().sum())
```

```
0
```

```
df['director'].fillna("Unknown",inplace=True)
df["cast"].fillna("Unknown",inplace=True)
df["country"].fillna("Unknown",inplace=True)
df.dropna(subset=["date_added"],inplace=True)
mas_comun=df["rating"].mode()[0]
```

```
<ipython-input-31-5197ac8abaab>:1: FutureWarning: A value is trying to be set
The behavior will change in pandas 3.0. This inplace method will never work b

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.
```

```
df['director'].fillna("Unknown",inplace=True)
```

```
<ipython-input-31-5197ac8abaab>:2: FutureWarning: A value is trying to be set
The behavior will change in pandas 3.0. This inplace method will never work b

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.
```

```
  df["cast"].fillna("Unknown",inplace=True)
```

```
<ipython-input-31-5197ac8abaab>:3: FutureWarning: A value is trying to be set
The behavior will change in pandas 3.0. This inplace method will never work b

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.
```

```
  df["country"].fillna("Unknown",inplace=True)
```

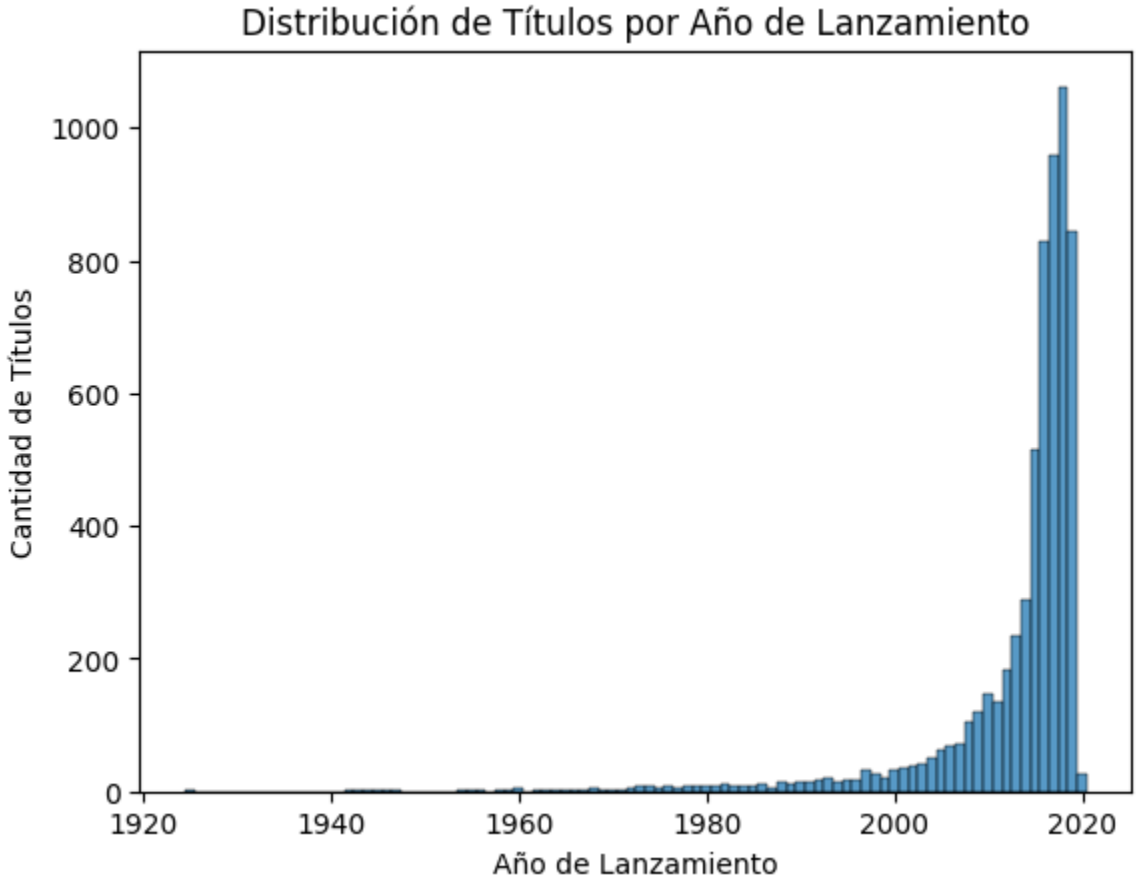
```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 6223 entries, 0 to 6222
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   show_id       6223 non-null   int64 
 1   type          6223 non-null   object
 2   title         6223 non-null   object
 3   director      6223 non-null   object
 4   cast          6223 non-null   object
 5   country       6223 non-null   object
 6   date_added    6223 non-null   object
 7   release_year  6223 non-null   int64 
 8   rating        6214 non-null   object
 9   duration      6223 non-null   object
 10  listed_in     6223 non-null   object
 11  description   6223 non-null   object
dtypes: int64(2), object(10)
memory usage: 632.0+ KB
```

```
1 #¿Cuántos títulos se estrenaron por año?
2 print(df.groupby("release_year")["title"].count())
```

```
release_year
1925       1
1942       2
1943       3
1944       3
1945       3
         ...
2016     828
2017     959
2018    1062
2019     843
2020      25
Name: title, Length: 72, dtype: int64
```

```
import matplotlib.pyplot as plt
sns.histplot(x="release_year", data=df, discrete=True) # discrete=True para a
plt.title("Distribución de Títulos por Año de Lanzamiento")
plt.xlabel("Año de Lanzamiento")
plt.ylabel("Cantidad de Títulos")
plt.show()
```



Se puede observar que en los años más recientes se han lanzado más peliculas, es una tendencia muy alta

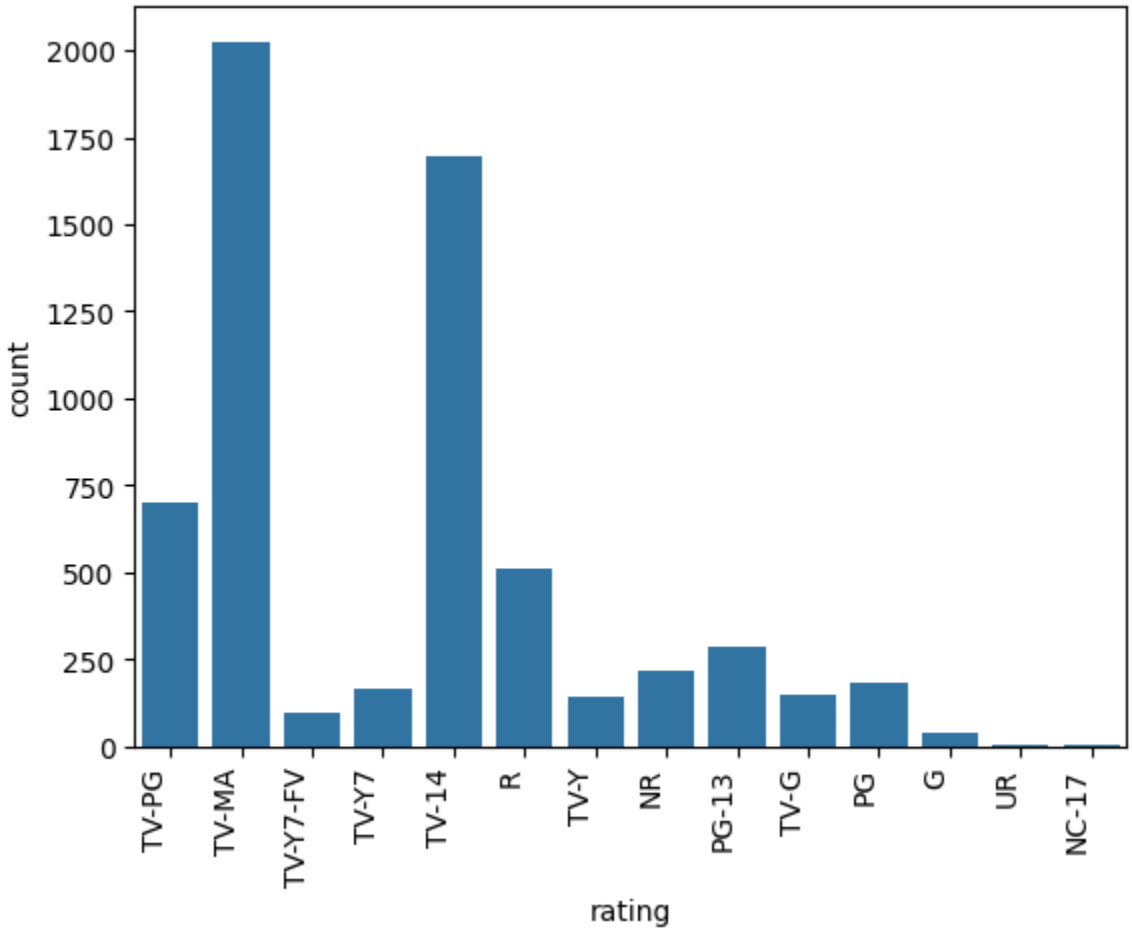
```
#¿Cuál es el título con mayor duración?
index_max_duration = df["duration"].idxmax()
titulo_max_duracion = df.loc[index_max_duration, "title"]
max_duration = df.loc[index_max_duration, "duration"]

print(f"El titulo con mayor duracion es {titulo_max_duracion} (duración: {max
```

```
El titulo con mayor duracion es #realityhigh (duración: 99 min)
```

```
#¿Qué clasificaciones (Rating) son más comunes?
print(df["rating"].value_counts())
```

```
rating
TV-MA       2025
TV-14       1695
TV-PG        699
R            508
PG-13        286
NR           217
PG           184
TV-Y7        168
TV-G         149
TV-Y         142
TV-Y7-FV      95
G             37
UR             7
NC-17          2
Name: count, dtype: int64
```

```
sns.countplot(x="rating", data=df)
plt.xticks(rotation=90, ha='right')
plt.show()
```



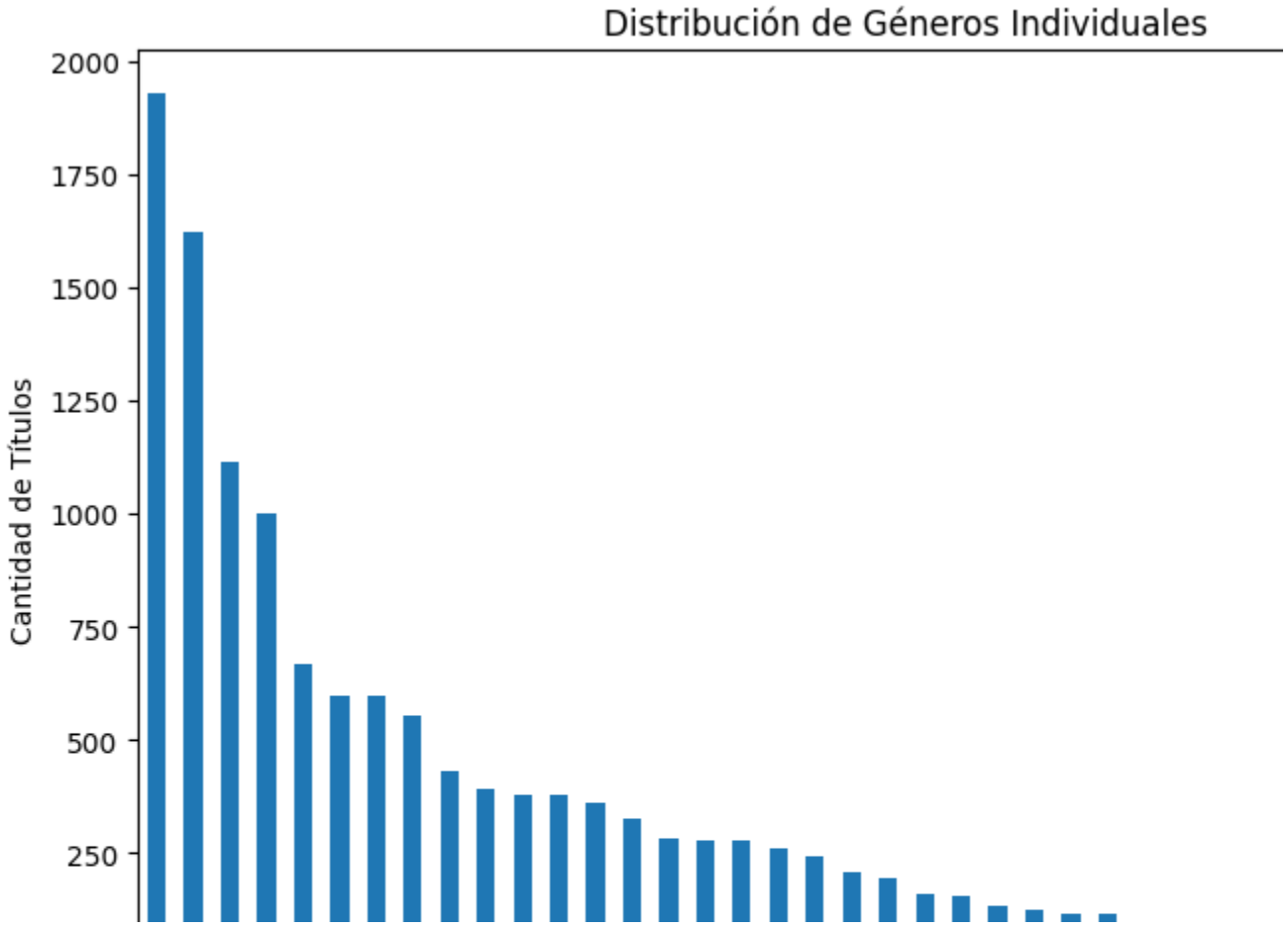
TV-MA y TV-14 son notablemente los rating en mayor tendencia

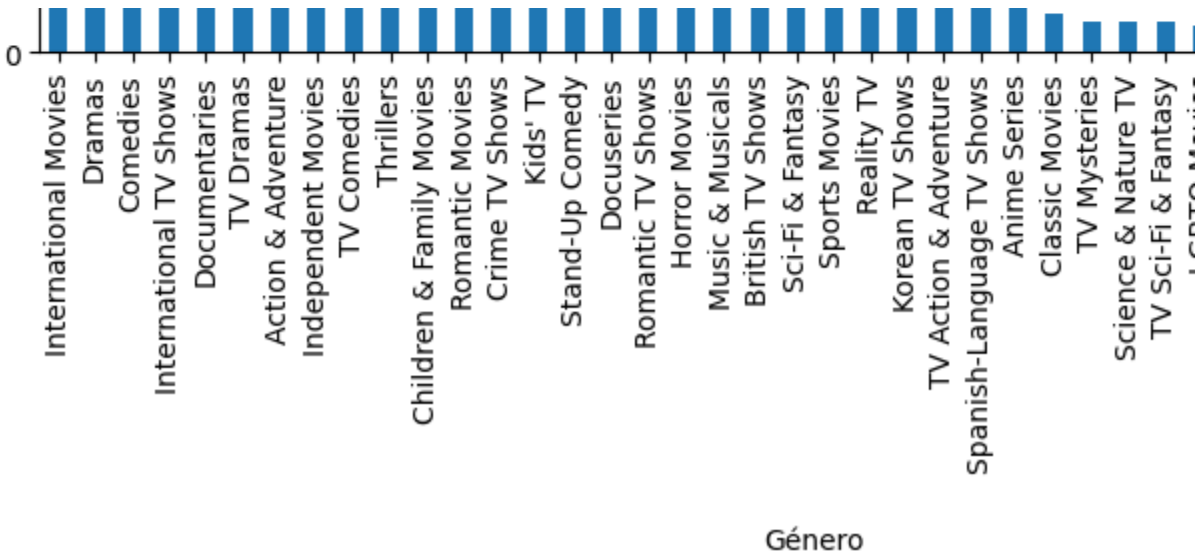
```
#¿Cómo se distribuyen los géneros (listed_in) más populares?
```

```
2 print(df["listed_in"].value_counts())
```

```
listed_in
Documentaries                                         299
Stand-Up Comedy                                       273
Dramas, International Movies                          248
Dramas, Independent Movies, International Movies      186
Comedies, Dramas, International Movies                174
                                                     ...
Classic & Cult TV, Kids' TV, TV Comedies                1
British TV Shows, TV Comedies, TV Dramas                1
Romantic TV Shows, TV Action & Adventure, TV Dramas     1
TV Comedies, TV Dramas, TV Horror                       1
Reality TV, Spanish-Language TV Shows                   1
Name: count, Length: 461, dtype: int64
```

```
1 genres = df["listed_in"].str.split(", ")
2
3 all_genres = pd.Series([genre for sublist in genres for genre in sublist])
4
5 all_genres.value_counts().plot(kind='bar', figsize=(10, 6))
6 plt.title("Distribución de Géneros Individuales")
7 plt.xlabel("Género")
8 plt.ylabel("Cantidad de Títulos")
9 plt.show()
```





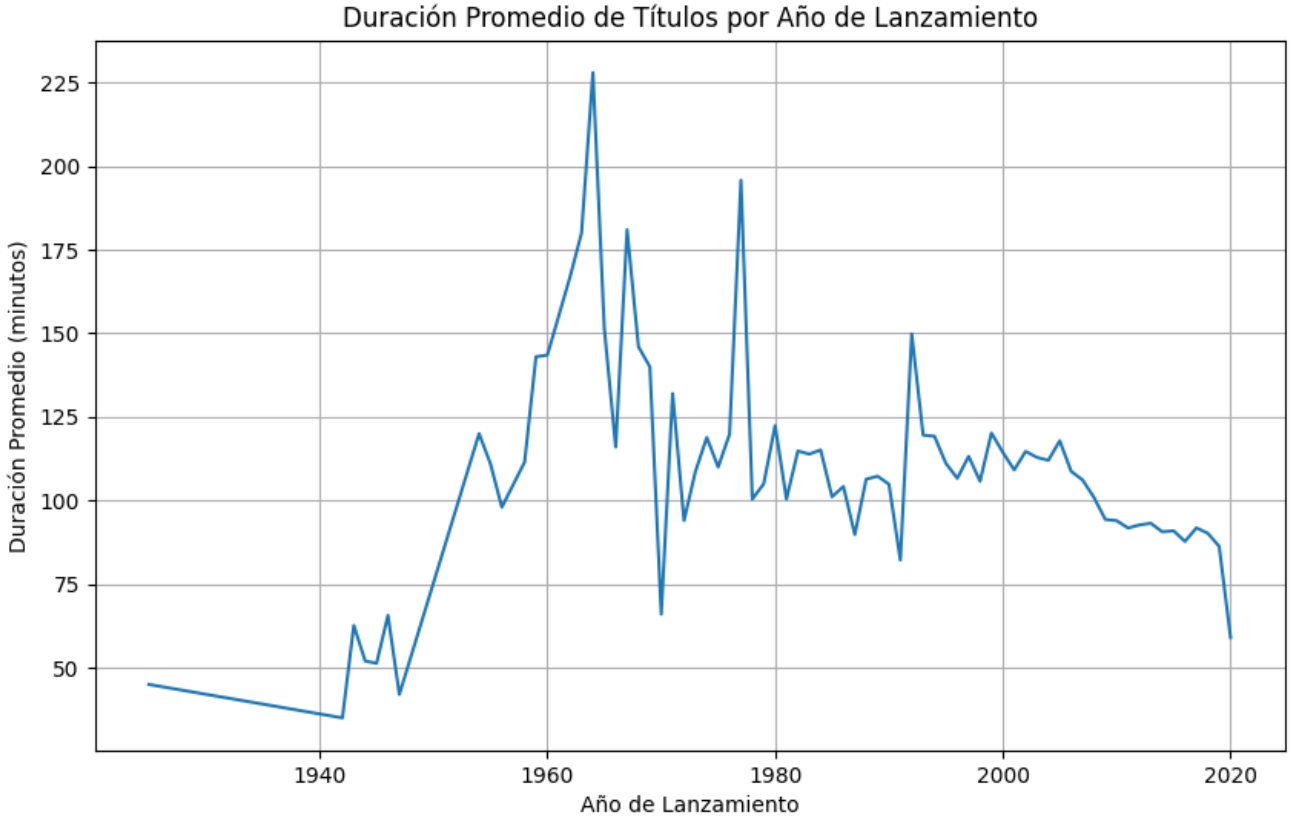
International Movies es el genero más visto y TV shows es el menos visto

```
#¿La duración promedio ha cambiado con los años?
import pandas as pd

def convert_duration_to_minutes(duration_str):
    if 'Season' in duration_str:
        # Asumimos una duración promedio por temporada (esto es una estimació
        # Puedes ajustarla si tienes información más precisa
        return int(duration_str.split(' ')[0]) * 45  # Ejemplo: 45 minutos po
    elif 'min' in duration_str:
        return int(duration_str.split(' ')[0])
    else:
        return None  # Manejar otros formatos si existen

df['duration_minutes'] = df['duration'].apply(convert_duration_to_minutes)

print(df.groupby("release_year")["duration_minutes"].mean())
```

```
release_year
1925    45.000000
1942    35.000000
1943    62.666667
1944    52.000000
1945    51.333333
          ...
2016    87.768116
2017    91.827946
2018    90.208098
2019    86.294187
2020    59.040000
Name: duration_minutes, Length: 72, dtype: float64
```

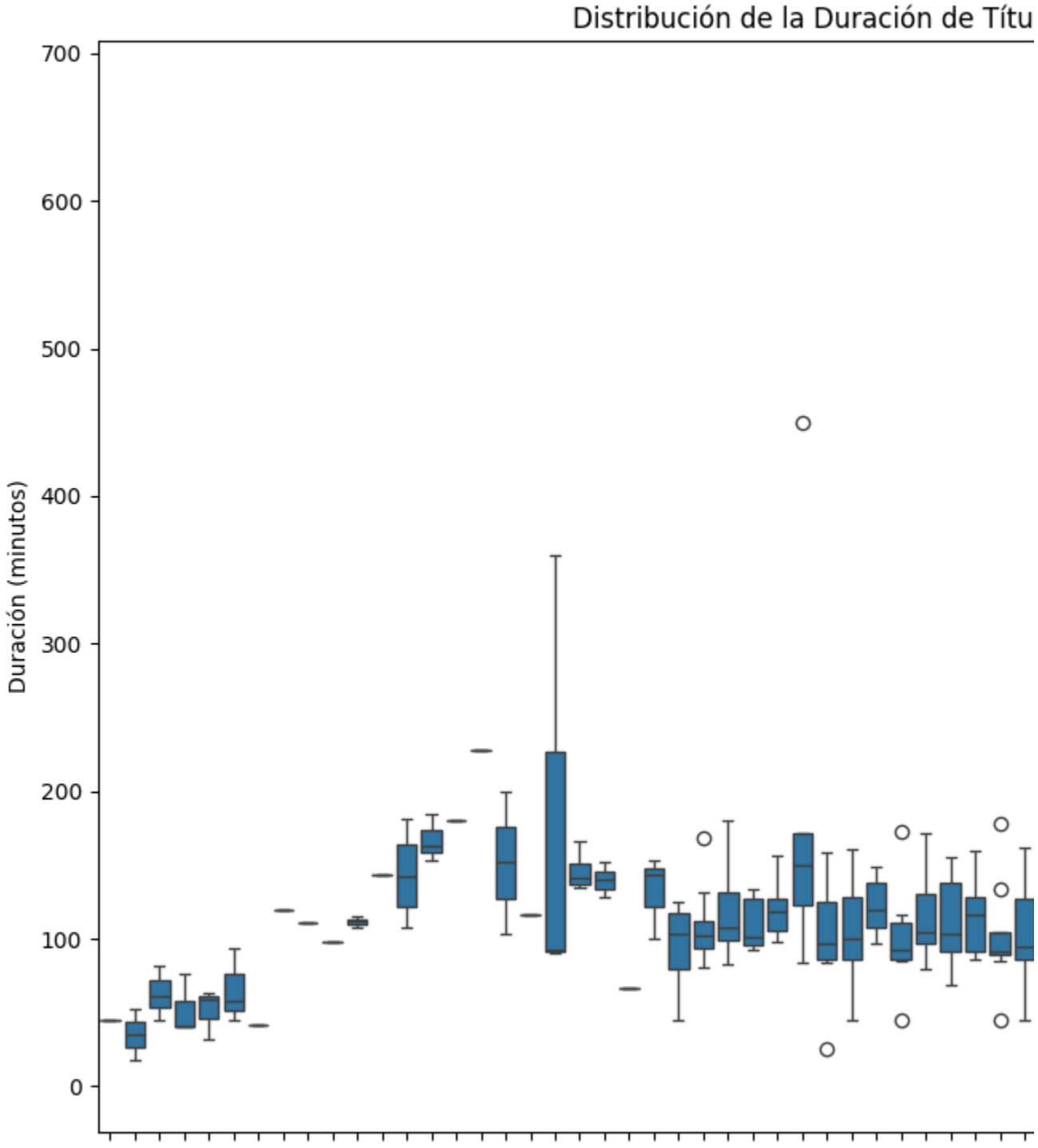
```
import seaborn as sns
```

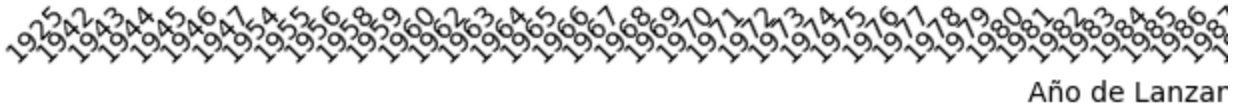
```
 1 import seaborn as sns
 2 import matplotlib.pyplot as plt
 3
 4 average_duration_by_year = df.groupby("release_year")["duration_minutes"].mea
 5
 6 plt.figure(figsize=(10, 6))
 7 sns.lineplot(x=average_duration_by_year.index, y=average_duration_by_year.val
 8 plt.title("Duración Promedio de Títulos por Año de Lanzamiento")
 9 plt.xlabel("Año de Lanzamiento")
10 plt.ylabel("Duración Promedio (minutos)")
11 plt.grid(True)
12 plt.show()
```



Es muy irregular aquì ya que en los recientes años del df se puede observar que la media de la duración de las peliculas era muy corta poy lo cual esta se fue elevando mucho hasya alcanzar en aproximadamente 225 minutos de media en 1962 aproximadamente, despues empezo a bajar y tener algunos picos (no igual de altos), en la actualidad las peliculas son de corta duración (no como la de los primeros años del df pero si de aproximadamente de 65 minutos)

```
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(12, 8))
sns.boxplot(x="release_year", y="duration_minutes", data=df)
plt.title("Distribución de la Duración de Títulos por Año de Lanzamiento")
plt.xlabel("Año de Lanzamiento")
plt.ylabel("Duración (minutos)")
plt.xticks(rotation=45, ha="right") # Rotar etiquetas si hay muchos años
plt.tight_layout()
plt.show()
```





En este análisis exploratorio de datos (EDA) se logró obtener una comprensión clara del comportamiento de los usuarios respecto a la aplicación, analizando tanto las valoraciones (ratings) como los comentarios. Se identificaron patrones relevantes, como la relación entre la puntuación otorgada y la polaridad de los comentarios, así como posibles áreas de mejora basadas en la frecuencia de palabras negativas.

Además, el análisis permitió detectar si existen problemas recurrentes reportados por los usuarios, lo cual puede ser clave para priorizar actualizaciones futuras de la app. Este proceso también evidenció la importancia del análisis textual para entender mejor la experiencia del usuario más allá de las métricas cuantitativas.

Aunque gran parte del trabajo fue automatizado con ayuda externa, este ejercicio sentó las bases para desarrollar habilidades futuras en limpieza, visualización y análisis de datos. Para próximas sesiones, se recomienda enfocarse en realizar estos pasos de forma manual para reforzar el aprendizaje y fomentar la independencia analítica.