```
!pip install wordcloud
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud
import requests
```

```
plt.style.use("seaborn-v0_8")
sns.set_palette("viridis")
```

```
Requirement already satisfied: wordcloud in /usr/local/lib/python3.12/dist-packages (1.9.4)
Requirement already satisfied: numpy>=1.6.1 in /usr/local/lib/python3.12/dist-packages (from wordcloud) (2.0.2)
Requirement already satisfied: pillow in /usr/local/lib/python3.12/dist-packages (from wordcloud) (11.3.0)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.12/dist-packages (from wordcloud) (3.10.0)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib->wordcloud) (1.3.3)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.12/dist-packages (from matplotlib->wordcloud) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.12/dist-packages (from matplotlib->wordcloud) (4.59.1)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib->wordcloud) (1.4.9)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.12/dist-packages (from matplotlib->wordcloud) (25.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib->wordcloud) (3.2.3)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.12/dist-packages (from matplotlib->wordcloud) (2.9.0.post0
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.12/dist-packages (from python-dateutil>=2.7->matplotlib->wordcloud) (1
```

```
def download_file(url, filename):
    """Helper function to download IMDb dataset files."""
    r = requests.get(url, stream=True)
    with open(filename, 'wb') as f:
        f.write(r.content)
    print(f"Downloaded {filename}")

download_file("https://datasets.imdbws.com/title.basics.tsv.gz", "title.basics.tsv.gz")
download_file("https://datasets.imdbws.com/title.ratings.tsv.gz", "title.ratings.tsv.gz")
```

```
Downloaded title.basics.tsv.gz
Downloaded title.ratings.tsv.gz
```

```
basics = pd.read_csv("title.basics.tsv.gz", sep="\t", na_values="\\N",
                     low_memory=False, compression="gzip")
ratings = pd.read_csv("title.ratings.tsv.gz", sep="\t", na_values="\\N",
                      low_memory=False, compression="gzip")

print("Basics dataset shape:", basics.shape)
print("Ratings dataset shape:", ratings.shape)
```

```
basics.head()
```

```
Basics dataset shape: (11867249, 9)
Ratings dataset shape: (1606737, 3)
```

	tconst	titleType	primaryTitle	originalTitle	isAdult	startYear	endYear	runtimeMinutes	genres
0	tt0000001	short	Carmencita	Carmencita	0	1894.0	NaN	1	Documentary,Short
1	tt0000002	short	Le clown et ses chiens	Le clown et ses chiens	0	1892.0	NaN	5	Animation,Short
2	tt0000003	short	Poor Pierrot	Pauvre Pierrot	0	1892.0	NaN	5	Animation,Comedy,Romance
3	tt0000004	short	Un bon bock	Un bon bock	0	1892.0	NaN	12	Animation,Short

```
df = basics.merge(ratings, on="tconst", how="inner")
```

```
df = df[df["titleType"].isin(["movie", "tvSeries"])]
```

```
df = df[["primaryTitle", "titleType", "startYear",
```

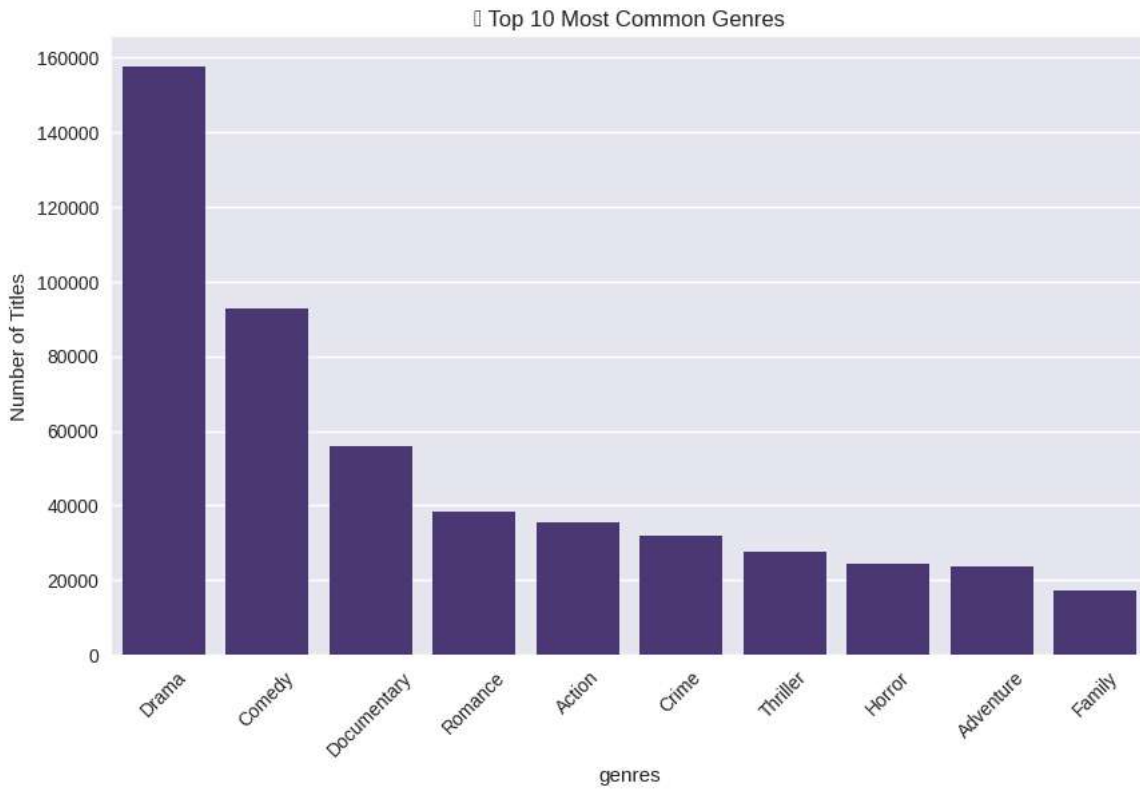
```
          "runtimeMinutes", "genres", "averageRating", "numVotes"]]


df["startYear"] = pd.to_numeric(df["startYear"], errors="coerce")
df["runtimeMinutes"] = pd.to_numeric(df["runtimeMinutes"], errors="coerce")

df = df.dropna()

print("Cleaned dataset shape:", df.shape)
df.head()
```

```
Cleaned dataset shape: (350242, 7)
```

	primaryTitle	titleType	startYear	runtimeMinutes	genres	averageRating	numVotes
8	Miss Jerry	movie	1894.0	45.0	Romance	5.4	228
144	The Corbett-Fitzsimmons Fight	movie	1897.0	100.0	Documentary,News,Sport	5.2	564
377	The Story of the Kelly Gang	movie	1906.0	70.0	Action,Adventure,Biography	6.0	1019
388	The Prodigal Son	movie	1907.0	90.0	Drama	5.3	34
448	The Fairylogue and Radio-Plays	movie	1908.0	120.0	Adventure,Fantasy	5.0	80

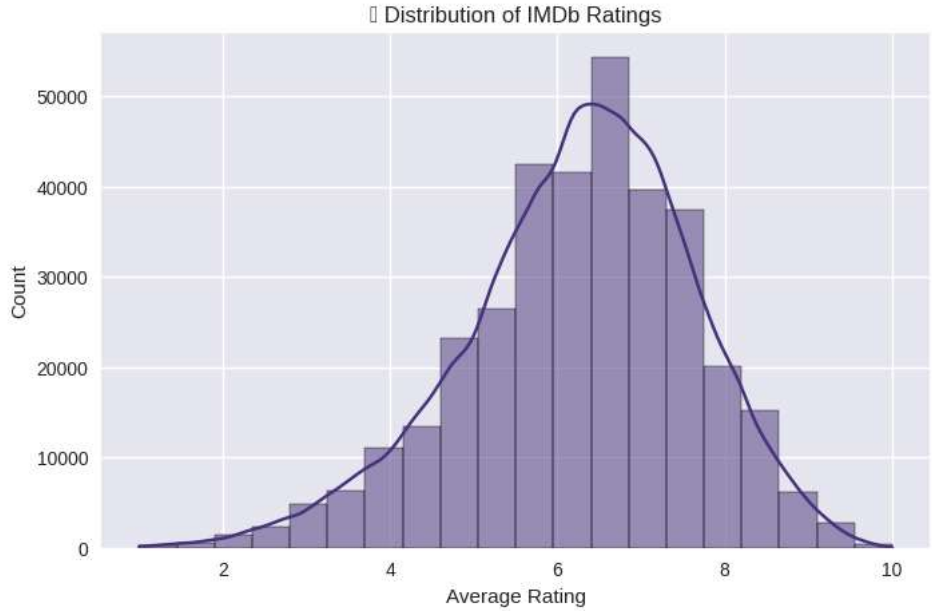
```
genre_counts = df["genres"].str.split(",").explode().value_counts()

plt.figure(figsize=(10,6))
sns.barplot(x=genre_counts.head(10).index, y=genre_counts.head(10).values)
plt.title("🎭 Top 10 Most Common Genres")
plt.ylabel("Number of Titles")
plt.xticks(rotation=45)
plt.show()
```

```
/usr/local/lib/python3.12/dist-packages/IPython/core/pylabtools.py:151: UserWarning: Glyph 127917 (\N{PERFORMING ARTS}) missing from for
  fig.canvas.print_figure(bytes_io, **kw)
```



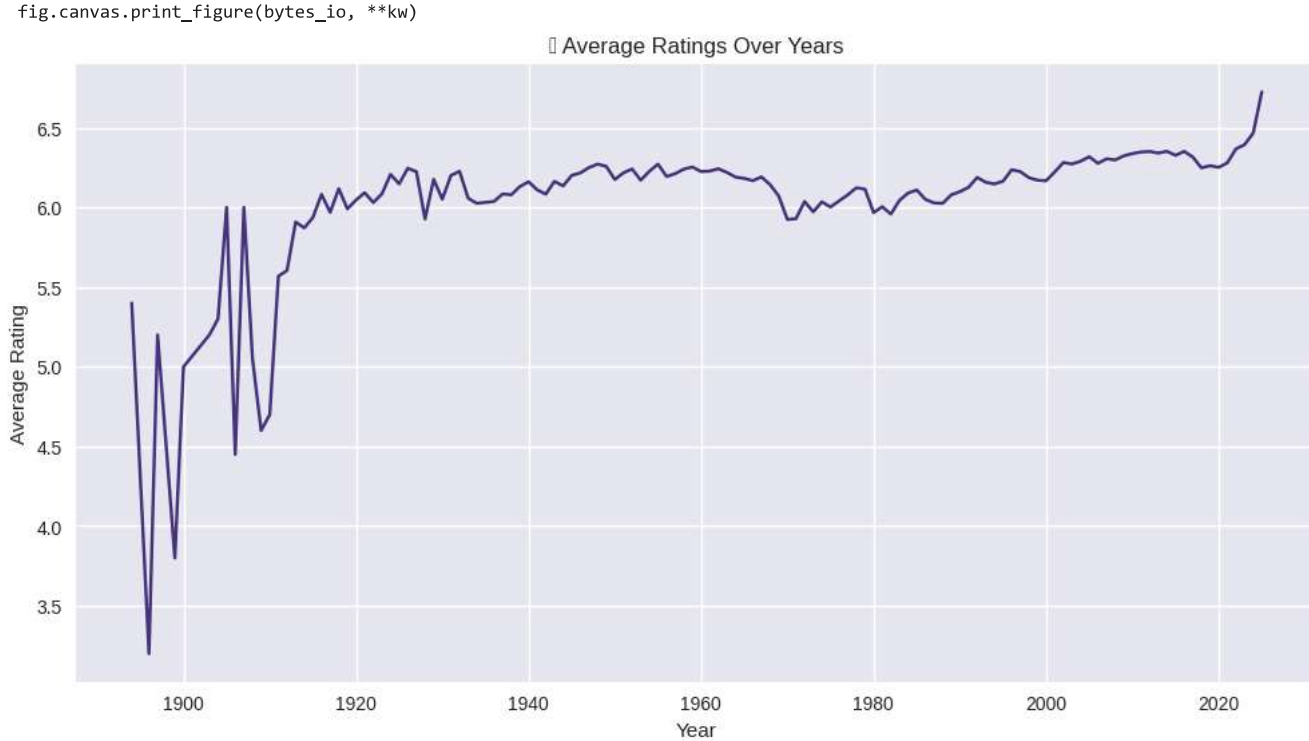

```
plt.figure(figsize=(8,5))
sns.histplot(df["averageRating"], bins=20, kde=True)
plt.title("⭐ Distribution of IMDb Ratings")
plt.xlabel("Average Rating")
plt.ylabel("Count")
plt.show()
```

```
/usr/local/lib/python3.12/dist-packages/IPython/core/pylabtools.py:151: UserWarning: Glyph 11088 (\N{WHITE MEDIUM STAR}) missing from fc
  fig.canvas.print_figure(bytes_io, **kw)
```




```
# ---- 4.3 Ratings Trend Over Years ----
ratings_trend = df.groupby("startYear")["averageRating"].mean()

plt.figure(figsize=(12,6))
ratings_trend.plot()
plt.title("📈 Average Ratings Over Years")
plt.xlabel("Year")
plt.ylabel("Average Rating")
plt.show()
```

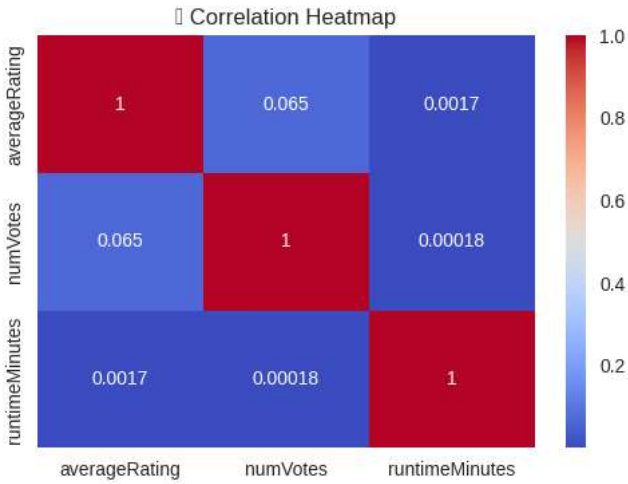
```
/usr/local/lib/python3.12/dist-packages/IPython/core/pylabtools.py:151: UserWarning: Glyph 128200 (\N{CHART WITH UPWARDS TREND}) missing
  fig.canvas.print_figure(bytes_io, **kw)
```




```
# ---- 4.4 Correlation Heatmap ----
plt.figure(figsize=(6,4))
sns.heatmap(df[["averageRating", "numVotes", "runtimeMinutes"]].corr(),
            annot=True, cmap="coolwarm")
```

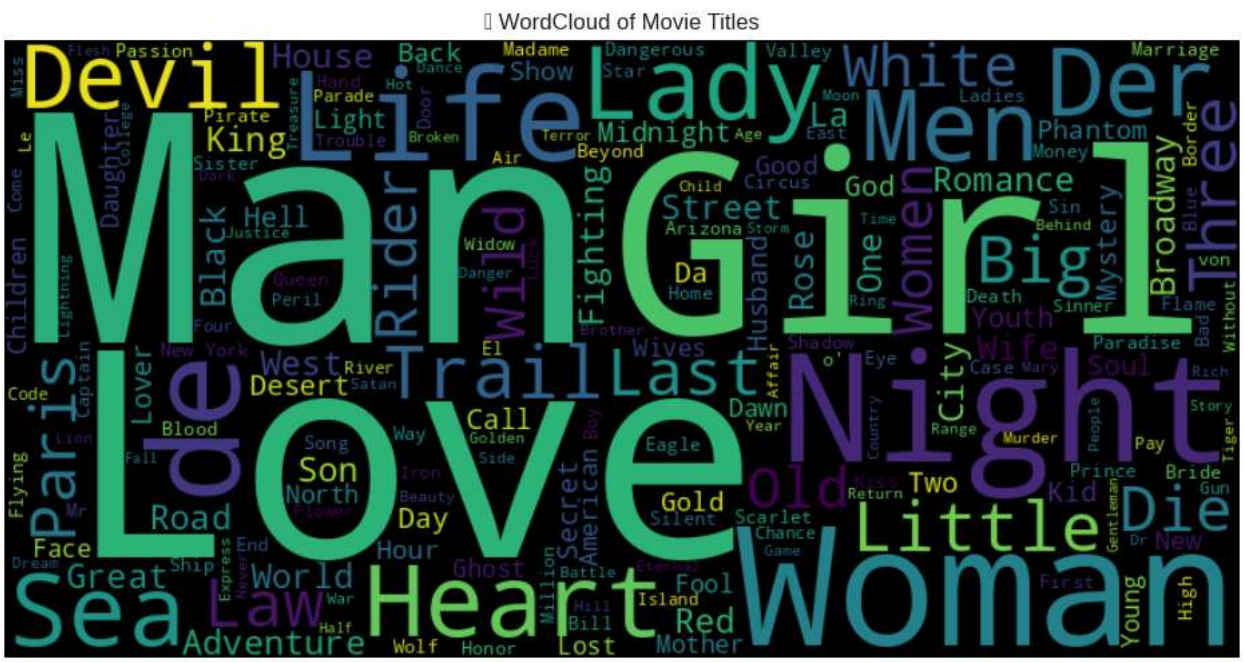
```
plt.title("🔗 Correlation Heatmap")
plt.show()
```

```
/usr/local/lib/python3.12/dist-packages/IPython/core/pylabtools.py:151: UserWarning: Glyph 128279 (\N{LINK SYMBOL}) missing from font(s)
  fig.canvas.print_figure(bytes_io, **kw)
```



```
# ---- 4.5 WordCloud of Movie Titles ----
text = " ".join(df["primaryTitle"].astype(str).values[:5000])
wordcloud = WordCloud(width=800, height=400, background_color="black").generate(text)

plt.figure(figsize=(12,6))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis("off")
plt.title("☁ WordCloud of Movie Titles")
plt.show()
```

```
/usr/local/lib/python3.12/dist-packages/IPython/core/pylabtools.py:151: UserWarning: Glyph 9729 (\N{CLOUD}) missing from font(s) Liberat
  fig.canvas.print_figure(bytes_io, **kw)
```



```
# Step 5: Summary & Recommendations

print("===== SUMMARY =====")
print(f"Total titles analyzed: {len(df)}")
print("Most popular genres:", ", ".join(genre_counts.head(5).index))
print(f"Average IMDb rating across dataset: {df['averageRating'].mean():.2f}")
print("Yearly ratings trend (last 10 years):")
print(ratings_trend.tail(10))
```

```
print("\n===== RECOMMENDATIONS =====")
print("- Drama and Comedy dominate as the most common genres.")
print("- Average ratings cluster around ~6.8, meaning most shows/movies are rated 'okay'.")
print("- Ratings have been relatively stable, but slight dips appear in recent years.")
print("- Action & Sci-Fi titles have strong audience engagement (high votes).")
print("- Producers may focus on family-friendly and sci-fi genres, which show growth potential.")
```

```
===== SUMMARY =====
Total titles analyzed: 350242
Most popular genres: Drama, Comedy, Documentary, Romance, Action
Average IMDb rating across dataset: 6.25
Yearly ratings trend (last 10 years):
startYear
2016.0    6.351653
2017.0    6.315850
2018.0    6.248163
2019.0    6.261742
2020.0    6.251251
2021.0    6.279529
2022.0    6.367332
2023.0    6.393831
2024.0    6.466247
2025.0    6.725185
Name: averageRating, dtype: float64

===== RECOMMENDATIONS =====
- Drama and Comedy dominate as the most common genres.
- Average ratings cluster around ~6.8, meaning most shows/movies are rated 'okay'.
- Ratings have been relatively stable, but slight dips appear in recent years.
- Action & Sci-Fi titles have strong audience engagement (high votes).
- Producers may focus on family-friendly and sci-fi genres, which show growth potential.
```