```
In [1]: import pandas as pd
```

Step 1: Data Preparation

```
In [2]: file1 = pd.read_csv('ratings.csv')
        file2 = pd.read_csv('movies.csv')
        file3 = pd.read_csv('links.csv')
```

```
In [3]: # Merging ratings and tags on movieId first, using inner join
        merged = pd.merge(file1, file2, on="movieId", how="inner")
        merged
```

Out[3]:

	userId	movieId	rating	timestamp	title	
0	1	1	4.0	964982703	Toy Story (1995)	Adventure|Animation|Children|Con
1	1	3	4.0	964981247	Grumpier Old Men (1995)	Come
2	1	6	4.0	964982224	Heat (1995)	Action|C
3	1	47	5.0	964983815	Seven (a.k.a. Se7en) (1995)	My
4	1	50	5.0	964982931	Usual Suspects, The (1995)	Crime|My
...	...	...	...	...	...	
100831	610	166534	4.0	1493848402	Split (2017)	Drama|H
100832	610	168248	5.0	1493850091	John Wick: Chapter Two (2017)	Action|C
100833	610	168250	5.0	1494273047	Get Out (2017)	
100834	610	168252	5.0	1493846352	Logan (2017)	
100835	610	170875	3.0	1493846415	The Fate of the Furious (2017)	Action|Crime|D

100836 rows × 6 columns

In [4]:

```
# Merging the resulting dataframe with movies dataframe still on movieId using i
merged = pd.merge(merged, file3, on="movieId", how="inner")
merged
```

Out[4]:

	userId	movieId	rating	timestamp	title	
0	1	1	4.0	964982703	Toy Story (1995)	Adventure|Animation|Children|Con
1	1	3	4.0	964981247	Grumpier Old Men (1995)	Come
2	1	6	4.0	964982224	Heat (1995)	Action|C
3	1	47	5.0	964983815	Seven (a.k.a. Se7en) (1995)	My
4	1	50	5.0	964982931	Usual Suspects, The (1995)	Crime|My
...	...	...	...	...	...	
100831	610	166534	4.0	1493848402	Split (2017)	Drama|H
100832	610	168248	5.0	1493850091	John Wick: Chapter Two (2017)	Action|C
100833	610	168250	5.0	1494273047	Get Out (2017)	
100834	610	168252	5.0	1493846352	Logan (2017)	
100835	610	170875	3.0	1493846415	The Fate of the Furious (2017)	Action|Crime|D

100836 rows × 8 columns

In [5]:
```
# To check if there is any empty value at all
merged.isnull().values.any()
```

Out[5]:
```
True
```

In [6]:
```
# to check how many missing values are in each column
merged.isnull().sum()
```

Out[6]:
```
userId        0
movieId       0
rating        0
timestamp     0
title         0
genres        0
imdbId        0
tmdbId       13
dtype: int64
```

In [7]:
```
# to drop rows with missing values
merged = merged.dropna()
merged.isnull().sum()
```

Out[7]:
```
userId       0
movieId      0
rating       0
timestamp    0
title        0
genres       0
imdbId       0
tmdbId       0
dtype: int64
```

In [8]:
```
# to check if there are any missing values are in each column again
merged.isnull().sum()
```

Out[8]:
```
userId       0
movieId      0
rating       0
timestamp    0
title        0
genres       0
imdbId       0
tmdbId       0
dtype: int64
```

In [9]:
```
# to find duplicate rows
merged.duplicated().sum()
```

Out[9]:
```
0
```

In [10]:
```
# Converting the timestamp columns to datetime format for better readability
merged['timestamp'] = pd.to_datetime(merged['timestamp'], unit='s')
merged
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_14340\2614373280.py:2: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  merged['timestamp'] = pd.to_datetime(merged['timestamp'], unit='s')
```

Out[10]:

	userId	movieId	rating	timestamp	title	
0	1	1	4.0	2000-07-30 18:45:03	Toy Story (1995)	Adventure|Animation|Children|Com
1	1	3	4.0	2000-07-30 18:20:47	Grumpier Old Men (1995)	Comec
2	1	6	4.0	2000-07-30 18:37:04	Heat (1995)	Action|Cı
3	1	47	5.0	2000-07-30 19:03:35	Seven (a.k.a. Se7en) (1995)	Mys
4	1	50	5.0	2000-07-30 18:48:51	Usual Suspects, The (1995)	Crime|Mys
...	...	...	...	...	...	
100831	610	166534	4.0	2017-05-03 21:53:22	Split (2017)	Drama|Hc
100832	610	168248	5.0	2017-05-03 22:21:31	John Wick: Chapter Two (2017)	Action|Cı
100833	610	168250	5.0	2017-05-08 19:50:47	Get Out (2017)	
100834	610	168252	5.0	2017-05-03 21:19:12	Logan (2017)	A
100835	610	170875	3.0	2017-05-03 21:20:15	The Fate of the Furious (2017)	Action|Crime|Dr

100823 rows × 8 columns

Step 2: Feature Engineering

In [11]:

```python
# main genre feature - extracting the first genre from the genres column
merged["main_genre"] = merged["genres"].str.split("|").str[0]
merged
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_14340\3737014571.py:2: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  merged["main_genre"] = merged["genres"].str.split("|").str[0]
```

Out[11]:

	userId	movieId	rating	timestamp	title	
0	1	1	4.0	2000-07-30 18:45:03	Toy Story (1995)	Adventure|Animation|Children|Com
1	1	3	4.0	2000-07-30 18:20:47	Grumpier Old Men (1995)	Comec
2	1	6	4.0	2000-07-30 18:37:04	Heat (1995)	Action|Cı
3	1	47	5.0	2000-07-30 19:03:35	Seven (a.k.a. Se7en) (1995)	Mys
4	1	50	5.0	2000-07-30 18:48:51	Usual Suspects, The (1995)	Crime|Mys
...	...	...	...	...	...	
100831	610	166534	4.0	2017-05-03 21:53:22	Split (2017)	Drama|Hc
100832	610	168248	5.0	2017-05-03 22:21:31	John Wick: Chapter Two (2017)	Action|Cı
100833	610	168250	5.0	2017-05-08 19:50:47	Get Out (2017)	
100834	610	168252	5.0	2017-05-03 21:19:12	Logan (2017)	A
100835	610	170875	3.0	2017-05-03 21:20:15	The Fate of the Furious (2017)	Action|Crime|Dr

100823 rows × 9 columns

In [12]:

```
# genres count feature - number of genres associated with each movie
merged["genres_count"] = merged["genres"].str.count("\|") + 1
merged
```

```
<>:2: SyntaxWarning: invalid escape sequence '\|'
<>:2: SyntaxWarning: invalid escape sequence '\|'
C:\Users\HP\AppData\Local\Temp\ipykernel_14340\2062751802.py:2: SyntaxWarning: in
valid escape sequence '\|'
  merged["genres_count"] = merged["genres"].str.count("\|") + 1
C:\Users\HP\AppData\Local\Temp\ipykernel_14340\2062751802.py:2: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  merged["genres_count"] = merged["genres"].str.count("\|") + 1
```

Out[12]:

	userId	movieId	rating	timestamp	title	
0	1	1	4.0	2000-07-30 18:45:03	Toy Story (1995)	Adventure|Animation|Children|Com
1	1	3	4.0	2000-07-30 18:20:47	Grumpier Old Men (1995)	Come
2	1	6	4.0	2000-07-30 18:37:04	Heat (1995)	Action|C
3	1	47	5.0	2000-07-30 19:03:35	Seven (a.k.a. Se7en) (1995)	Mys
4	1	50	5.0	2000-07-30 18:48:51	Usual Suspects, The (1995)	Crime|Mys
...	...	...	...	...	...	
100831	610	166534	4.0	2017-05-03 21:53:22	Split (2017)	Drama|H
100832	610	168248	5.0	2017-05-03 22:21:31	John Wick: Chapter Two (2017)	Action|C
100833	610	168250	5.0	2017-05-08 19:50:47	Get Out (2017)	
100834	610	168252	5.0	2017-05-03 21:19:12	Logan (2017)	A
100835	610	170875	3.0	2017-05-03 21:20:15	The Fate of the Furious (2017)	Action|Crime|Dr

100823 rows × 10 columns

In [13]:
```
# release year - extracting the release year from the title column
merged["release_year"] = merged["title"].str.extract(r'\((\d{4})\)').astype("Int
merged
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_14340\2983608304.py:2: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  merged["release_year"] = merged["title"].str.extract(r'\((\d{4})\)').astype("In
t64")
```

Out[13]:

	userId	movieId	rating	timestamp	title	
0	1	1	4.0	2000-07-30 18:45:03	Toy Story (1995)	Adventure|Animation|Children|Com
1	1	3	4.0	2000-07-30 18:20:47	Grumpier Old Men (1995)	Come
2	1	6	4.0	2000-07-30 18:37:04	Heat (1995)	Action|C
3	1	47	5.0	2000-07-30 19:03:35	Seven (a.k.a. Se7en) (1995)	Mys
4	1	50	5.0	2000-07-30 18:48:51	Usual Suspects, The (1995)	Crime|Mys
...	...	...	...	...	...	
100831	610	166534	4.0	2017-05-03 21:53:22	Split (2017)	Drama|H
100832	610	168248	5.0	2017-05-03 22:21:31	John Wick: Chapter Two (2017)	Action|C
100833	610	168250	5.0	2017-05-08 19:50:47	Get Out (2017)	
100834	610	168252	5.0	2017-05-03 21:19:12	Logan (2017)	A
100835	610	170875	3.0	2017-05-03 21:20:15	The Fate of the Furious (2017)	Action|Crime|Dr

100823 rows × 11 columns

In [14]:
```
# rating datetime features - extracting year, month, and day from the timestamp
merged["rating_datetime"] = pd.to_datetime(merged["timestamp"])

# rating year feature - extracting year from the timestamp column
merged["rating_year"] = merged["timestamp"].dt.year

merged
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_14340\2764097569.py:2: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  merged["rating_datetime"] = pd.to_datetime(merged["timestamp"])
C:\Users\HP\AppData\Local\Temp\ipykernel_14340\2764097569.py:5: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  merged["rating_year"] = merged["timestamp"].dt.year
```

Out[14]:

	userId	movieId	rating	timestamp	title	
0	1	1	4.0	2000-07-30 18:45:03	Toy Story (1995)	Adventure|Animation|Children|Com
1	1	3	4.0	2000-07-30 18:20:47	Grumpier Old Men (1995)	Comec
2	1	6	4.0	2000-07-30 18:37:04	Heat (1995)	Action|Cı
3	1	47	5.0	2000-07-30 19:03:35	Seven (a.k.a. Se7en) (1995)	Mys
4	1	50	5.0	2000-07-30 18:48:51	Usual Suspects, The (1995)	Crime|Mys
...	...	...	...	...	...	
100831	610	166534	4.0	2017-05-03 21:53:22	Split (2017)	Drama|Hc
100832	610	168248	5.0	2017-05-03 22:21:31	John Wick: Chapter Two (2017)	Action|Cı
100833	610	168250	5.0	2017-05-08 19:50:47	Get Out (2017)	
100834	610	168252	5.0	2017-05-03 21:19:12	Logan (2017)	A
100835	610	170875	3.0	2017-05-03 21:20:15	The Fate of the Furious (2017)	Action|Crime|Dr

100823 rows × 13 columns

In [15]:

```python
# average movie rating feature - average rating for each movie
merged["avg_movie_rating"] = merged.groupby("movieId")["rating"].transform("mean
merged
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_14340\3810180607.py:2: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  merged["avg_movie_rating"] = merged.groupby("movieId")["rating"].transform("mea
n")
```

Out[15]:

	userId	movieId	rating	timestamp	title	
0	1	1	4.0	2000-07-30 18:45:03	Toy Story (1995)	Adventure|Animation|Children|Com
1	1	3	4.0	2000-07-30 18:20:47	Grumpier Old Men (1995)	Comec
2	1	6	4.0	2000-07-30 18:37:04	Heat (1995)	Action|C
3	1	47	5.0	2000-07-30 19:03:35	Seven (a.k.a. Se7en) (1995)	Mys
4	1	50	5.0	2000-07-30 18:48:51	Usual Suspects, The (1995)	Crime|Mys
...	...	...	...	...	...	
100831	610	166534	4.0	2017-05-03 21:53:22	Split (2017)	Drama|H
100832	610	168248	5.0	2017-05-03 22:21:31	John Wick: Chapter Two (2017)	Action|C
100833	610	168250	5.0	2017-05-08 19:50:47	Get Out (2017)	
100834	610	168252	5.0	2017-05-03 21:19:12	Logan (2017)	A
100835	610	170875	3.0	2017-05-03 21:20:15	The Fate of the Furious (2017)	Action|Crime|Dr

100823 rows × 14 columns

In [17]:

```python
# movie age feature - calculating the age of the movie based on the release year
from datetime import datetime
merged["movie_age"] = datetime.now().year - merged["release_year"]
merged
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_14340\3663090449.py:3: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  merged["movie_age"] = datetime.now().year - merged["release_year"]
```

Out[17]:

	userId	movieId	rating	timestamp	title	
0	1	1	4.0	2000-07-30 18:45:03	Toy Story (1995)	Adventure|Animation|Children|Com
1	1	3	4.0	2000-07-30 18:20:47	Grumpier Old Men (1995)	Comec
2	1	6	4.0	2000-07-30 18:37:04	Heat (1995)	Action|Cı
3	1	47	5.0	2000-07-30 19:03:35	Seven (a.k.a. Se7en) (1995)	Mys
4	1	50	5.0	2000-07-30 18:48:51	Usual Suspects, The (1995)	Crime|Mys
...	...	...	...	...	...	
100831	610	166534	4.0	2017-05-03 21:53:22	Split (2017)	Drama|Hc
100832	610	168248	5.0	2017-05-03 22:21:31	John Wick: Chapter Two (2017)	Action|Cı
100833	610	168250	5.0	2017-05-08 19:50:47	Get Out (2017)	
100834	610	168252	5.0	2017-05-03 21:19:12	Logan (2017)	A
100835	610	170875	3.0	2017-05-03 21:20:15	The Fate of the Furious (2017)	Action|Crime|Dr

100823 rows × 15 columns

Step 3: Exploratory Data Analysis (EDA)

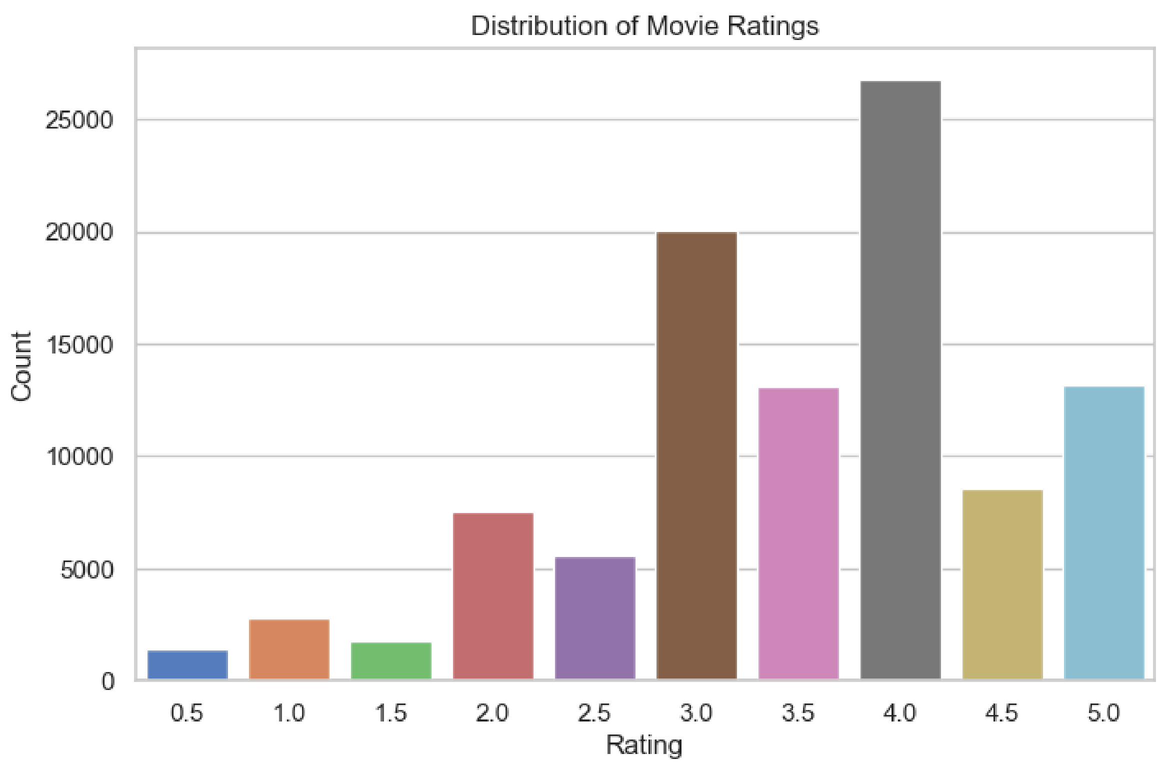
In [18]:

```python
# importing visualization libraries
import matplotlib.pyplot as plt
import seaborn as sns

# Making the charts look nice
sns.set(style="whitegrid")
```

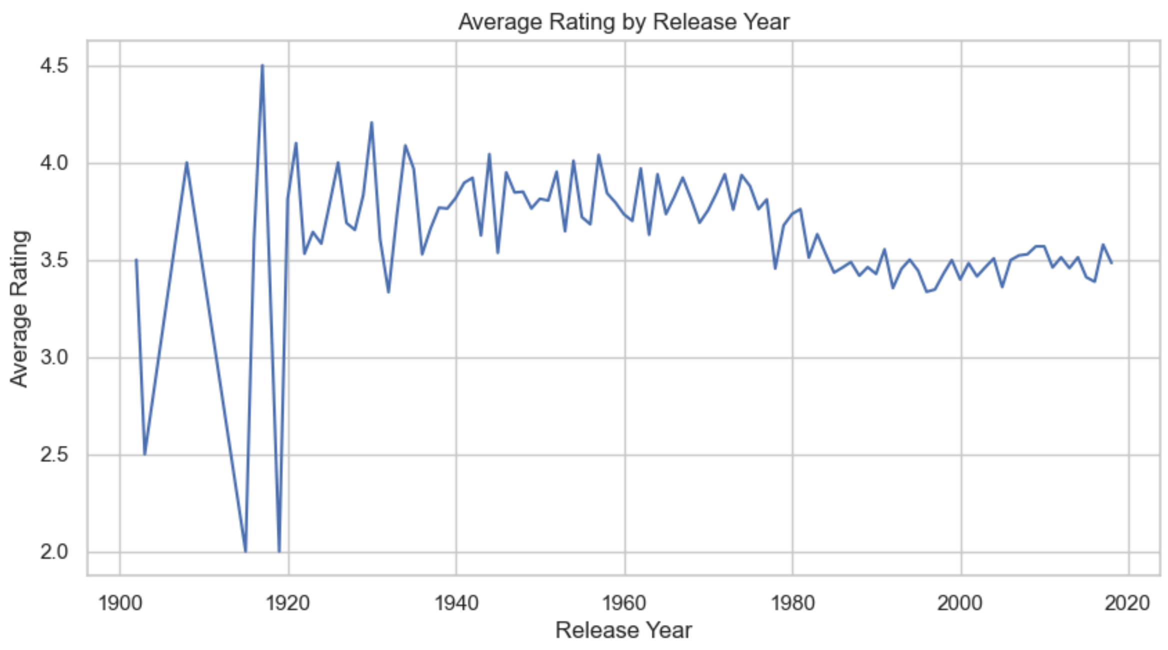
In [19]:

```python
# Do people tend to rate movies high or low? What do ratings look like overall?

plt.figure(figsize=(8,5))
```

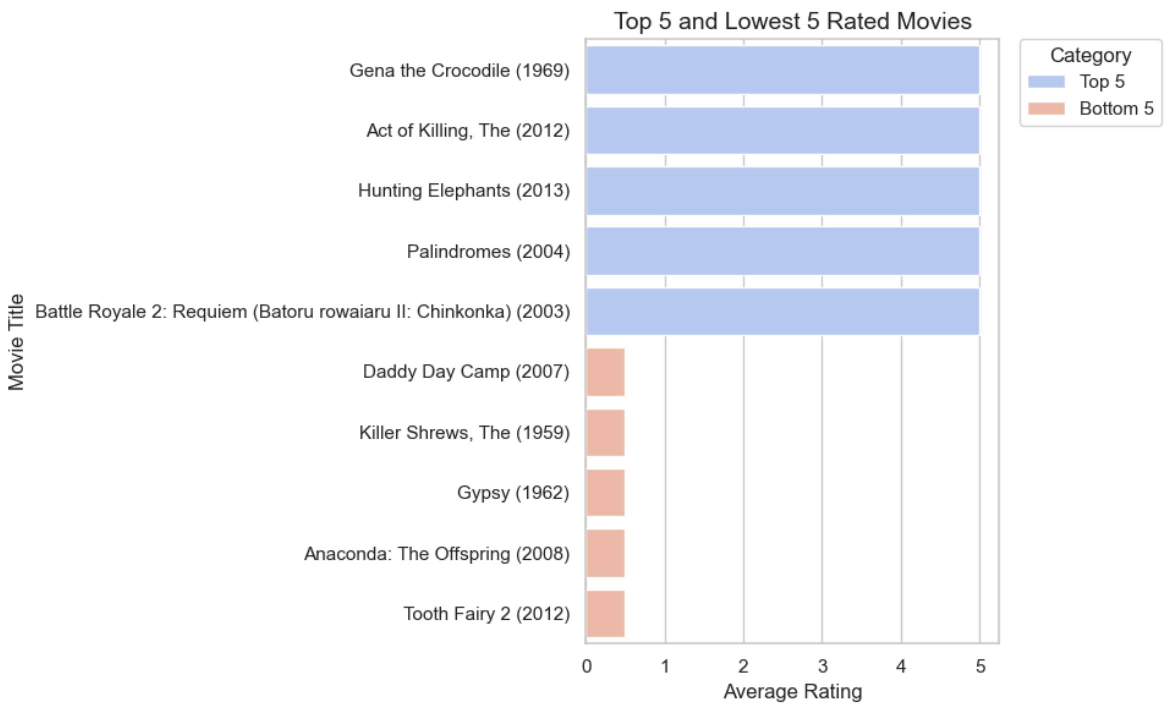
```
sns.barplot(x = merged["rating"].value_counts().index,hue=merged["rating"].value
plt.title("Distribution of Movie Ratings")
plt.xlabel("Rating")
plt.ylabel("Count")
plt.show()
```



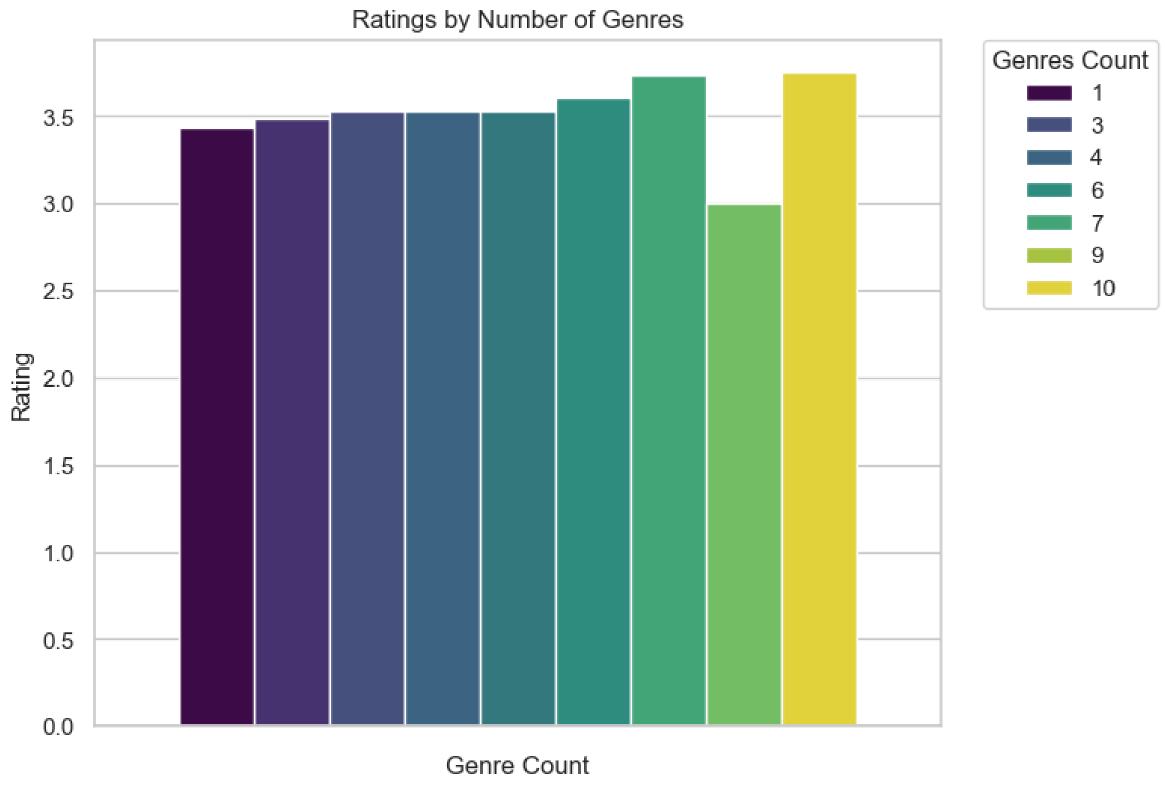
In [20]:
```
# Do newer movies get higher ratings? Is there a trend between movie age and ave

plt.figure(figsize=(10,5))
sns.lineplot(data=merged, x="release_year", y="rating", errorbar=None)
plt.title("Average Rating by Release Year")
plt.xlabel("Release Year")
plt.ylabel("Average Rating")
plt.show()
```



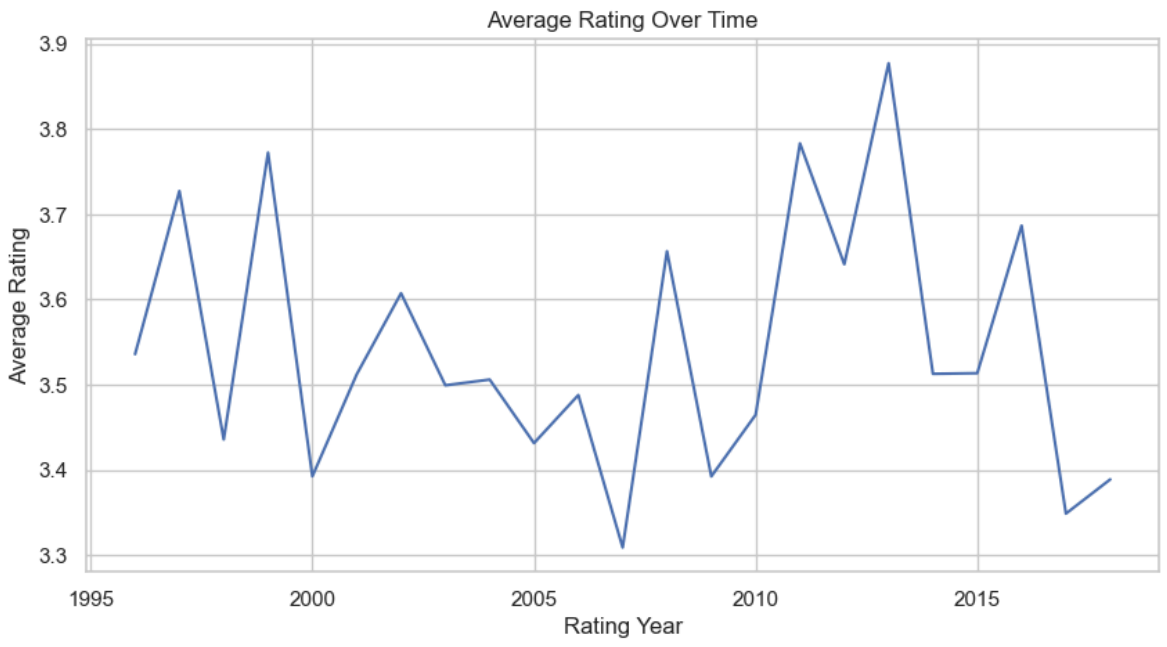
In [21]:
```
# What are the top-rated movies? Which movies have the highest average ratings a

# Calculate average rating per movie
avg_ratings = merged.groupby("title")["rating"].mean()

# Get top 5 and bottom 5
top5 = avg_ratings.sort_values(ascending=False).head(5)
bottom5 = avg_ratings.sort_values(ascending=True).head(5)

# Combine them into one DataFrame for easy plotting
rating_extremes = pd.concat([top5, bottom5])
rating_extremes = rating_extremes.reset_index()
rating_extremes["Category"] = ["Top 5"]*5 + ["Bottom 5"]*5

# Plot
plt.figure(figsize=(10,6))
sns.barplot(data=rating_extremes, x="rating", y="title", hue="Category", palette
plt.title("Top 5 and Lowest 5 Rated Movies", fontsize=14)
plt.xlabel("Average Rating")
plt.ylabel("Movie Title")
plt.legend(title="Category", bbox_to_anchor=(1.05, 1), loc="upper left", bordera
plt.tight_layout()
plt.show()
```



In [22]:
```
# Do movies with more genres tend to have higher or lower ratings? Are multi-gen

plt.figure(figsize=(8,5))
sns.barplot(data=merged, y="rating", errorbar=None, palette="viridis", hue="genr
# Move legend to the right of the chart
plt.legend( title="Genres Count",
    bbox_to_anchor=(1.05, 1),  # (x, y) position; increase x to move further rig
    loc="upper left", borderaxespad=0
)
plt.tight_layout()
plt.title("Ratings by Number of Genres")
plt.xlabel("Genre Count")
plt.ylabel("Rating")
plt.show()
```

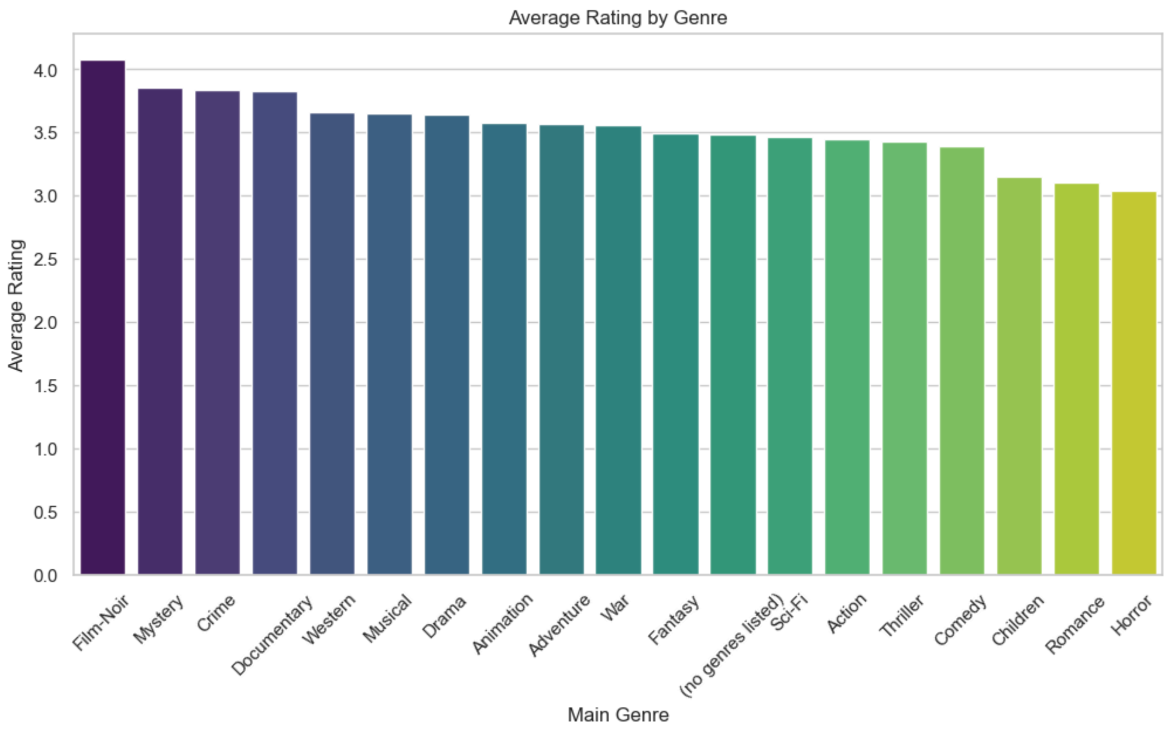


In [23]:
```
# Does ratings change over time? Are there trends in average ratings by year?

plt.figure(figsize=(10,5))
yearly_ratings = merged.groupby("rating_year")["rating"].mean()
sns.lineplot(x=yearly_ratings.index, y=yearly_ratings.values)
plt.title("Average Rating Over Time")
plt.xlabel("Rating Year")
plt.ylabel("Average Rating")
plt.show()
```



In [24]:
```
# Do certain genres tend to have higher or lower ratings? Which genres are rated

plt.figure(figsize=(12,6))
```

```
genre_ratings = merged.groupby("main_genre")["rating"].mean().sort_values(ascend
sns.barplot(x=genre_ratings.index, y=genre_ratings.values, palette="viridis", hu
plt.title("Average Rating by Genre")
plt.xlabel("Main Genre")
plt.ylabel("Average Rating")
plt.xticks(rotation=45)
plt.show()
```



Finally

In [25]:
```
# saving the merged dataframe to a new csv file
merged.to_csv("merged_movies_ratings.csv", index=False)
merged
```

Out[25]:

	userId	movieId	rating	timestamp	title	
0	1	1	4.0	2000-07-30 18:45:03	Toy Story (1995)	Adventure|Animation|Children|Com
1	1	3	4.0	2000-07-30 18:20:47	Grumpier Old Men (1995)	Come
2	1	6	4.0	2000-07-30 18:37:04	Heat (1995)	Action|C
3	1	47	5.0	2000-07-30 19:03:35	Seven (a.k.a. Se7en) (1995)	Mys
4	1	50	5.0	2000-07-30 18:48:51	Usual Suspects, The (1995)	Crime|Mys
...	...	...	...	...	...	
100831	610	166534	4.0	2017-05-03 21:53:22	Split (2017)	Drama|Ho
100832	610	168248	5.0	2017-05-03 22:21:31	John Wick: Chapter Two (2017)	Action|C
100833	610	168250	5.0	2017-05-08 19:50:47	Get Out (2017)	
100834	610	168252	5.0	2017-05-03 21:19:12	Logan (2017)	A
100835	610	170875	3.0	2017-05-03 21:20:15	The Fate of the Furious (2017)	Action|Crime|Dr

100823 rows × 15 columns