## Introduction

### What is Netflix?

Netflix, Inc. is a prominent American subscription-based streaming service and production company. It boasts an extensive library of films and television series, comprising both licensed content and original productions known as Netflix Originals. As of March 31, 2023, Netflix has garnered approximately 232.5 million paid memberships across over 190 countries, making it the world's most-subscribed video on demand streaming service.

Founded by Reed Hastings and Marc Randolph in Scotts Valley, California, Netflix began as a DVD sales and rental service. Within a year, it transitioned to focus solely on DVD rentals. In 2007, the company revolutionized its business model by introducing streaming media and video on demand services.

**Objective**

The aim is to analyze the available data to generate insights that will aid Netflix in determining which types of shows and movies to produce and how to expand its business in various countries.

**About the Data**

The dataset, as of mid-2021, includes information on approximately 8,807 movies and TV shows available on Netflix. It encompasses various details such as cast, directors, ratings, release year, duration, and more, all compiled into a single CSV file.

**Features of the Dataset**

Show ID: Unique identifier for each show.

Type Identifier: Specifies whether the entry is a movie or a TV show. Title: The title of the movie or TV show.

Director: The director of the movie or TV show.

Cast: The actors involved in the movie or show.

Country: The country where the movie or show was produced.

Date Added: The date the movie or show was added to Netflix.

Release Year: The year the movie or show was originally released.

Rating: The TV rating of the movie or show.

Duration: Total duration in minutes for movies or the number of seasons for TV shows.

Listed In: The genres the movie or show falls under.

Description: A brief summary of the movie or show.

## Importing Python Libraries - Numpy, Pandas, Matplotlib, Seaborn

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

## Importing the Dataset

```
netflix_data = pd.read_csv('netflix.csv')
```

## Exploring the Data

```
netflix_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   show_id       8807 non-null   object
 1   type          8807 non-null   object
 2   title         8807 non-null   object
 3   director      6173 non-null   object
 4   cast          7982 non-null   object
 5   country       7976 non-null   object
 6   date_added    8797 non-null   object
 7   release_year  8807 non-null   int64
 8   rating        8803 non-null   object
 9   duration      8804 non-null   object
 10  listed_in     8807 non-null   object
 11  description   8807 non-null   object
dtypes: int64(1), object(11)
memory usage: 825.8+ KB
```

```
netflix_data.dtypes
```

```
show_id         object
type            object
title           object
director        object
cast            object
country         object
date_added      object
release_year     int64
rating          object
duration        object
listed_in       object
description     object
dtype: object
```

```
netflix_data.describe(include="all")
```

| | show_id | type | title | director | cast | country | date_added | release_year | rating | duration | listed_in | description |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **count** | 8807 | 8807 | 8807 | 6173 | 7982 | 7976 | 8797 | 8807.000000 | 8803 | 8804 | 8807 | 8807 |
| **unique** | 8807 | 2 | 8807 | 4528 | 7692 | 748 | 1767 | NaN | 17 | 220 | 514 | 8775 |
| **top** | s1 | Movie | Dick Johnson Is Dead | Rajiv Chilaka | David Attenborough | United States | January 1, 2020 | NaN | TV-MA | 1 Season | Dramas, International Movies | Paranormal activity at a lush, abandoned prope... |
| **freq** | 1 | 6131 | 1 | 19 | 19 | 2818 | 109 | NaN | 3207 | 1793 | 362 | 4 |
| **mean** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | 2014.180198 | NaN | NaN | NaN | NaN |
| **std** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | 8.819312 | NaN | NaN | NaN | NaN |
| **min** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | 1925.000000 | NaN | NaN | NaN | NaN |
| **25%** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | 2013.000000 | NaN | NaN | NaN | NaN |
| **50%** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | 2017.000000 | NaN | NaN | NaN | NaN |
| **75%** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | 2019.000000 | NaN | NaN | NaN | NaN |
| **max** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | 2021.000000 | NaN | NaN | NaN | NaN |

```
netflix_data.head() #shows 1st 5 data from dataset
```

Show hidden output

## Data Cleaning and Analysis

### Un-nesting the columns

```
def unnest(df, column):
    return (df.drop(column, axis=1).join(df[column].str.split(',', expand=True).stack()
    .reset_index(level=1, drop=True).rename(column)))

# Un-nesting the 'cast' column
un_cast = unnest(netflix_data, 'cast')

# Un-nesting the 'title' column
un_title = unnest(netflix_data, 'title')

# Un-nesting the 'country' column
unn_country = unnest(netflix_data, 'country')

# Un-nesting the 'listed_in' (genre) column
unn_listed_in = unnest(netflix_data, 'listed_in')

# Un-nesting the 'director' column
unn_director = unnest(netflix_data, 'director')
```

## Checking and Handling Null values

```
netflix_data.isnull().sum()
```

```
show_id            0
type               0
title              0
director        2634
cast             825
country          831
date_added        10
release_year       0
rating             4
duration           3
listed_in          0
description        0
dtype: int64
```

## For Categorical columns with null values, replace it with Unknown_Columnname and for Numerical columns, replacing with 0.

```
catcol = ['director', 'cast', 'country', 'listed_in', 'rating', 'date_added', 'release_year']
for i in catcol:
  netflix_data[i].fillna(f'Unknown {i.capitalize()}', inplace=True)

netflix_data.head()
```

| | show_id | type | title | director | cast | country | date_added | release_year | rating | duration | listed_in | description |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | s1 | Movie | Dick Johnson Is Dead | Kirsten Johnson | Unknown Cast | United States | September 25, 2021 | 2020 | PG-13 | 90 min | Documentaries | As her father nears the end of his life, filmm... |
| 1 | s2 | TV Show | Blood & Water | Unknown Director | Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban... | South Africa | September 24, 2021 | 2021 | TV-MA | 2 Seasons | International TV Shows, TV Dramas, TV Mysteries | After crossing paths at a party, a Cape Town t... |
| 2 | s3 | TV Show | Ganglands | Julien Leclercq | Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi... | Unknown Country | September 24, 2021 | 2021 | TV-MA | 1 Season | Crime TV Shows, International TV Shows, TV Act... | To protect his family from a powerful drug lor... |
| 3 | s4 | TV Show | Jailbirds New Orleans | Unknown Director | Unknown Cast | Unknown Country | September 24, 2021 | 2021 | TV-MA | 1 Season | Docuseries, Reality TV | Feuds, flirtations and toilet talk go down amo... |
| 4 | s5 | TV Show | Kota Factory | Unknown Director | Mayur More, Jitendra Kumar, Ranjan Raj, Alam K... | India | September 24, 2021 | 2021 | TV-MA | 2 Seasons | International TV Shows, Romantic TV Shows, TV ... | In a city of coaching centers known to train I... |

```
numcol = [ 'duration' ]
for i in numcol:
  netflix_data[i].fillna(0, inplace = True)

netflix_data.head()
```

| | show_id | type | title | director | cast | country | date_added | release_year | rating | duration | listed_in | description |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | s1 | Movie | Dick Johnson Is Dead | Kirsten Johnson | Unknown Cast | United States | September 25, 2021 | 2020 | PG-13 | 90 min | Documentaries | As her father nears the end of his life, filmm... |
| 1 | s2 | TV Show | Blood & Water | Unknown Director | Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban... | South Africa | September 24, 2021 | 2021 | TV-MA | 2 Seasons | International TV Shows, TV Dramas, TV Mysteries | After crossing paths at a party, a Cape Town t... |
| 2 | s3 | TV Show | Ganglands | Julien Leclercq | Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi... | Unknown Country | September 24, 2021 | 2021 | TV-MA | 1 Season | Crime TV Shows, International TV Shows, TV Act... | To protect his family from a powerful drug lor... |
| 3 | s4 | TV Show | Jailbirds New Orleans | Unknown Director | Unknown Cast | Unknown Country | September 24, 2021 | 2021 | TV-MA | 1 Season | Docuseries, Reality TV | Feuds, flirtations and toilet talk go down amo... |
| 4 | s5 | TV Show | Kota Factory | Unknown Director | Mayur More, Jitendra Kumar, Ranjan Raj, Alam K... | India | September 24, 2021 | 2021 | TV-MA | 2 Seasons | International TV Shows, Romantic TV Shows, TV ... | In a city of coaching centers known to train I... |

## We can also see that There is presense of 3 unusual values in rating column, replacing it with appropriate values:

```
netflix_data['rating'].replace({'74 min':np.nan ,'84 min' : np.nan, '66 min' : np.nan},inplace = True)
```

```
ratings = {"13TH: A Conversation with Oprah Winfrey & Ava DuVernay" : 'TV-PG',
         "Gargantia on the Verdurous Planet" : 'TV-PG',
          "Little Lunch" : 'TV-Y',
          "My Honor Was Loyalty" : 'PG-13',
          "Louis C.K. 2017" : 'TV-MA',
          "Louis C.K.: Hilarious" : 'TV-MA',
          "Louis C.K.: Live at the Comedy Store":'TV-MA'}

for i in ratings:
    netflix_data.loc[netflix_data['title'] == i,'rating'] = ratings[i]
```

```
netflix_data['rating'].unique()
```

```
array(['PG-13', 'TV-MA', 'PG', 'TV-14', 'TV-PG', 'TV-Y', 'TV-Y7', 'R',
       'TV-G', 'G', 'NC-17', 'NR', 'TV-Y7-FV', 'UR'], dtype=object)
```

```
#Checking for Null values
netflix_data.isnull().sum()
```

```
show_id         0
type            0
title           0
director        0
cast            0
country         0
date_added      0
release_year    0
rating          0
duration        0
listed_in       0
description     0
dtype: int64
```

## Checking Unique and No. of unique values of each column

```
#Checking Unique Values
for i in ['type','release_year','rating']:
    print('Unique Values in',i,'column are :-')
    print(netflix_data[i].unique())
    print('-'*70)
```

```
Unique Values in type column are :-
['Movie' 'TV Show']
----------------------------------------------------------------------
Unique Values in release_year column are :-
[2020 2021 1993 2018 1996 1998 1997 2010 2013 2017 1975 1978 1983 1987
 2012 2001 2014 2002 2003 2004 2011 2008 2009 2007 2005 2006 1994 2015
 2019 2016 1982 1989 1990 1991 1999 1986 1992 1984 1980 1961 2000 1995
 1985 1976 1959 1988 1981 1972 1964 1945 1954 1979 1958 1956 1963 1970
 1973 1925 1974 1960 1966 1971 1962 1969 1977 1967 1968 1965 1946 1942
 1955 1944 1947 1943]
----------------------------------------------------------------------
Unique Values in rating column are :-
['PG-13' 'TV-MA' 'PG' 'TV-14' 'TV-PG' 'TV-Y' 'TV-Y7' 'R' 'TV-G' 'G'
 'NC-17' 'NR' 'TV-Y7-FV' 'UR']
----------------------------------------------------------------------
```

```
#Checking value_counts()
for i in ['type','release_year','rating']:
    print('Value count in',i,'column are :-')
    print(netflix_data[i].value_counts())
    print('-'*70)
```

```
Value count in type column are :-
type
Movie      6131
TV Show    2676
Name: count, dtype: int64
----------------------------------------------------------------------
Value count in release_year column are :-
release_year
2018    1147
2017    1032
2019    1030
2020     953
2016     902
        ...
1959       1
1925       1
1961       1
1947       1
```

```
1966       1
Name: count, Length: 74, dtype: int64
----------------------------------------------------------------------
Value count in rating column are :-
rating
TV-MA       3210
TV-14       2160
TV-PG        865
R            799
PG-13        491
TV-Y7        334
TV-Y         308
PG           287
TV-G         220
NR            80
G             41
TV-Y7-FV       6
NC-17          3
UR             3
Name: count, dtype: int64
----------------------------------------------------------------------
```

## Adding 3 columns - year_added,month_added,week_added to facilitate further data analysis

```
netflix_data['date_added'] = pd.to_datetime(netflix_data['date_added'], errors='coerce')
netflix_data = netflix_data.dropna(subset=['date_added'])

#adding new columns
netflix_data.loc[:, 'year_added'] = netflix_data['date_added'].dt.year
netflix_data.loc[:, 'month_added'] = netflix_data['date_added'].dt.month_name()
netflix_data.loc[:, 'week_added'] = netflix_data['date_added'].dt.isocalendar().week

netflix_data.head()
```

Show hidden output

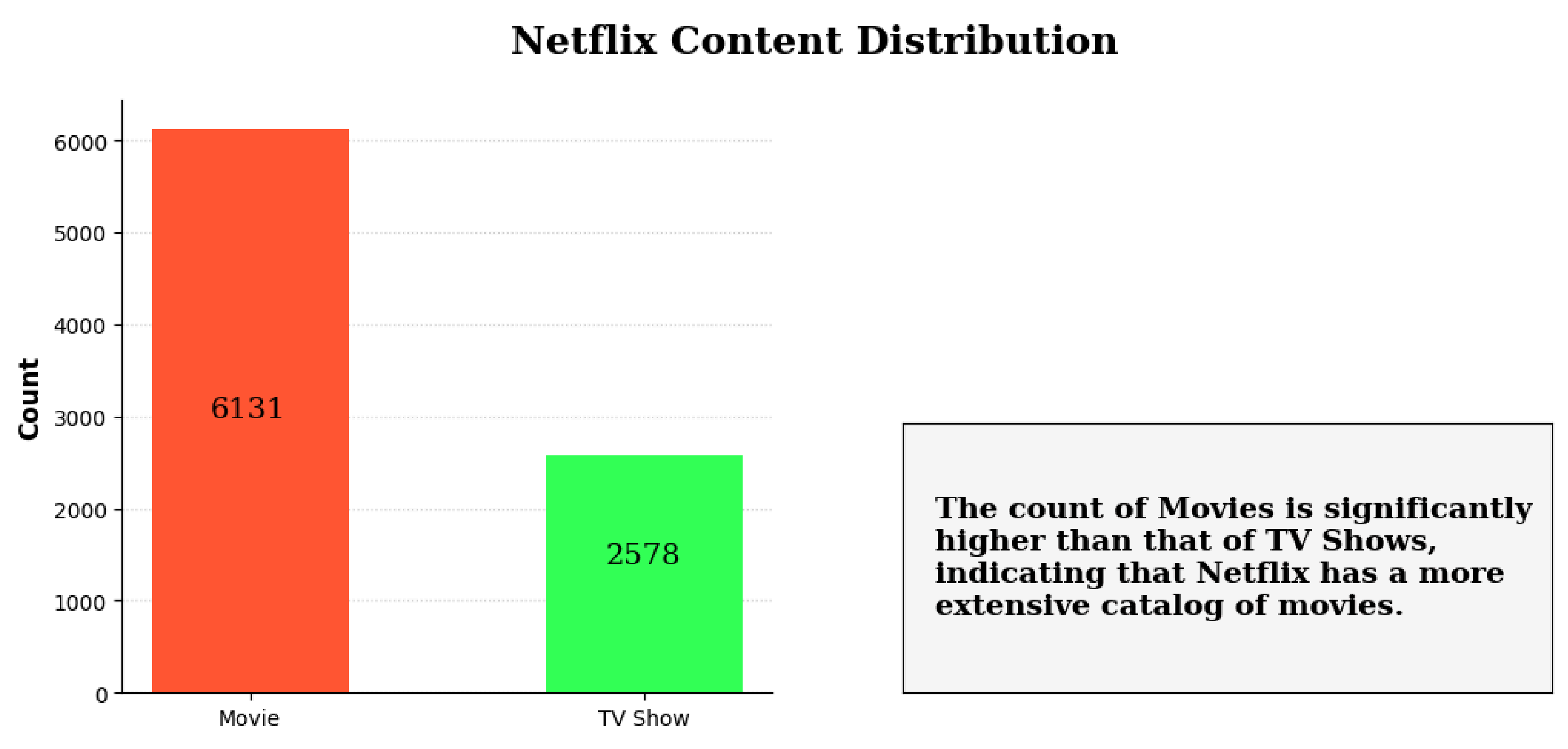
## Data Visualization

### 1. Content Distribution

```
x = netflix_data['type'].value_counts()
x
```

```
type
Movie      6131
TV Show    2578
Name: count, dtype: int64
```

```
fig = plt.figure(figsize=(12, 5))
gs = fig.add_gridspec(2, 2)

# creating graph for count of movies
ax0 = fig.add_subplot(gs[:, 0])
ax0.bar(x.index, x.values, color=['#FF5733', '#33FF57'], zorder=2, width=0.5)
ax0.set_ylabel('Count', fontsize=12, fontweight='bold')

# adding value_count label
ax0.text(-0.1, 3000, x.values[0], fontsize=14, fontweight='light', fontfamily='serif', color='black')
ax0.text(0.9, 1400, x.values[1], fontsize=14, fontweight='light', fontfamily='serif', color='black')
ax0.grid(color='lightgray', linestyle=':', axis='y', zorder=0)

# removing the axis lines
for s in ['top','right']:
    ax0.spines[s].set_visible(False)

# adding text insight
ax1 = fig.add_subplot(gs[1, 1])
ax1.set_facecolor('#f6f5f5')
ax1.set_xticks([])
ax1.set_yticks([])

ax1.text(0.05, 0.5, 'The count of Movies is significantly \nhigher than that of TV Shows, \nindicating that Netflix has a more \nextensive catalog of movies.',
         va='center', ha='left', fontsize=14, fontweight='bold', fontfamily='serif', color='black')

# adding title to the visual
fig.suptitle('Netflix Content Distribution', fontproperties={'family': 'serif', 'size': 18, 'weight': 'bold'})

plt.show()
```



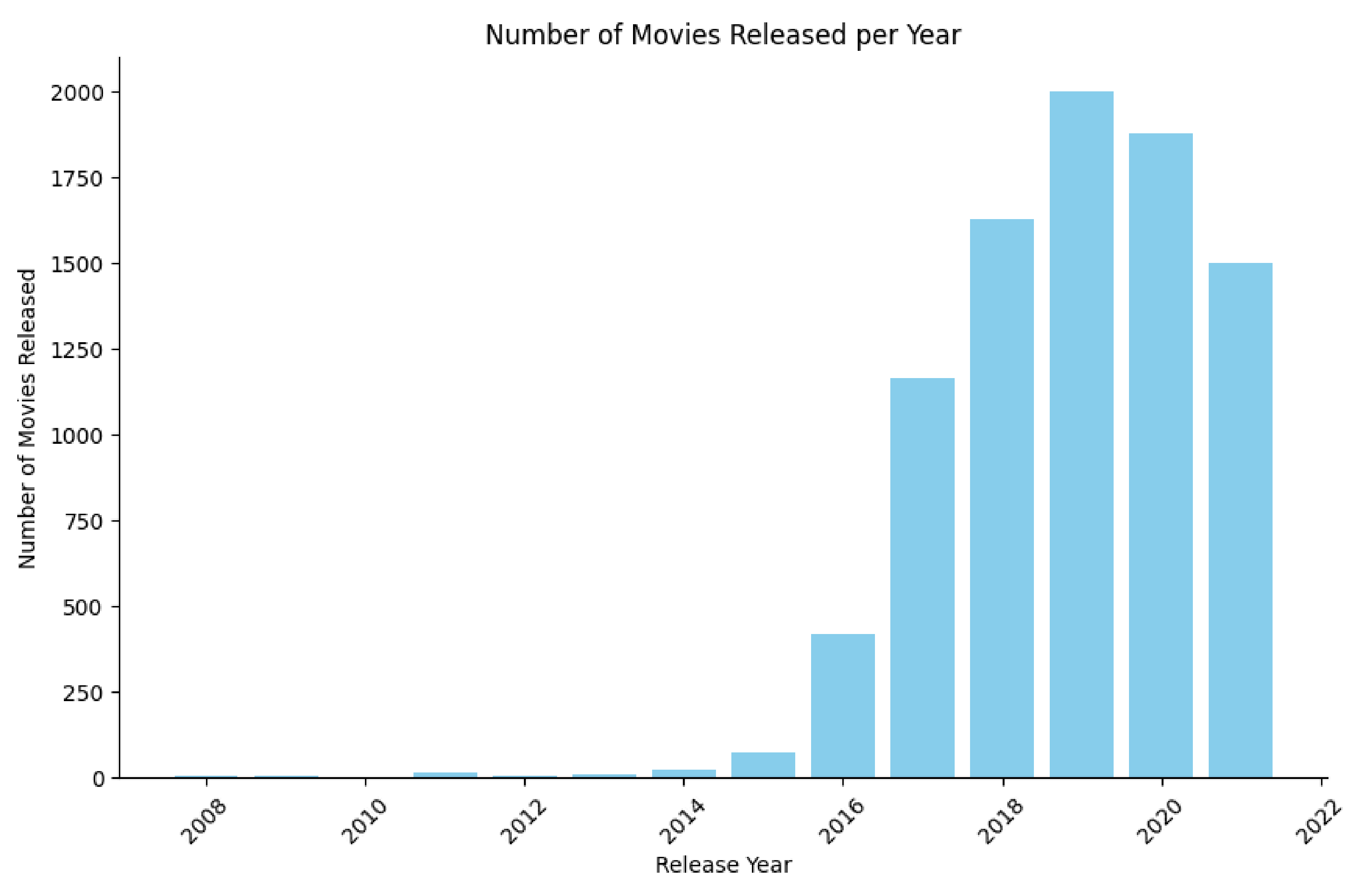
The count of Movies is significantly higher than that of TV Shows, indicating that Netflix has a more extensive catalog of movies.

### 2. How has the number of movies released per year changed over the last 20-30 years?

```
current_year = pd.Timestamp.now().year
start_year = current_year - 30
movies_last_30_years = netflix_data[netflix_data['year_added'] >= start_year]

# Group by release year and count the number of movies released each year
movies_per_year = movies_last_30_years.groupby('year_added').size()

# Print the data
print("Number of Movies Released per Year over the Last 30 Years:")
print(movies_per_year)

# Calculate statistical measures
total_movies = movies_per_year.sum()
average_movies_per_year = movies_per_year.mean()
max_movies_year = movies_per_year.idxmax()
max_movies_count = movies_per_year.max()

# Print statistical measures
print("\nStatistical Measures:")
print(f"Total number of movies released: {total_movies}")
print(f"Average number of movies released per year: {average_movies_per_year:.2f}")
print(f"Year with the most movies released: {max_movies_year} with {max_movies_count} movies")
```

```
Number of Movies Released per Year over the Last 30 Years:
year_added
2008       2
2009       2
2010       1
2011      13
2012       3
2013      10
2014      23
2015      73
2016     418
2017    1164
2018    1625
2019    1999
2020    1878
2021    1498
dtype: int64

Statistical Measures:
Total number of movies released: 8709
Average number of movies released per year: 622.07
Year with the most movies released: 2019 with 1999 movies
```

```
fig, ax = plt.subplots(figsize=(10, 6))

# Removing the axis lines
for s in ['top', 'right']:
    ax.spines[s].set_visible(False)

# Plotting the trend of movie releases over the last 30 years
ax.bar(movies_per_year.index, movies_per_year.values, color='skyblue')

# Setting title and labels
ax.set_title('Number of Movies Released per Year')
ax.set_xlabel('Release Year')
ax.set_ylabel('Number of Movies Released')

# Adding gridlines
#ax.grid(axis='y')

plt.xticks(rotation=45)
plt.show()
```



## Insight

**Early days of Netflix Originals:** In the early 2010s, Netflix was just starting to produce its own original content. They may have been focusing on building their library of licensed content and figuring out the best way to produce their own movies.

**Increased investment in originals:** As Netflix became more successful, they may have decided to invest more money in producing original content, including movies. This could explain the rise in releases in the later 2014s.

**Competition:** Competition from other streaming services like Hulu and Amazon Prime Video may have also spurred Netflix to produce more original content, including movies.

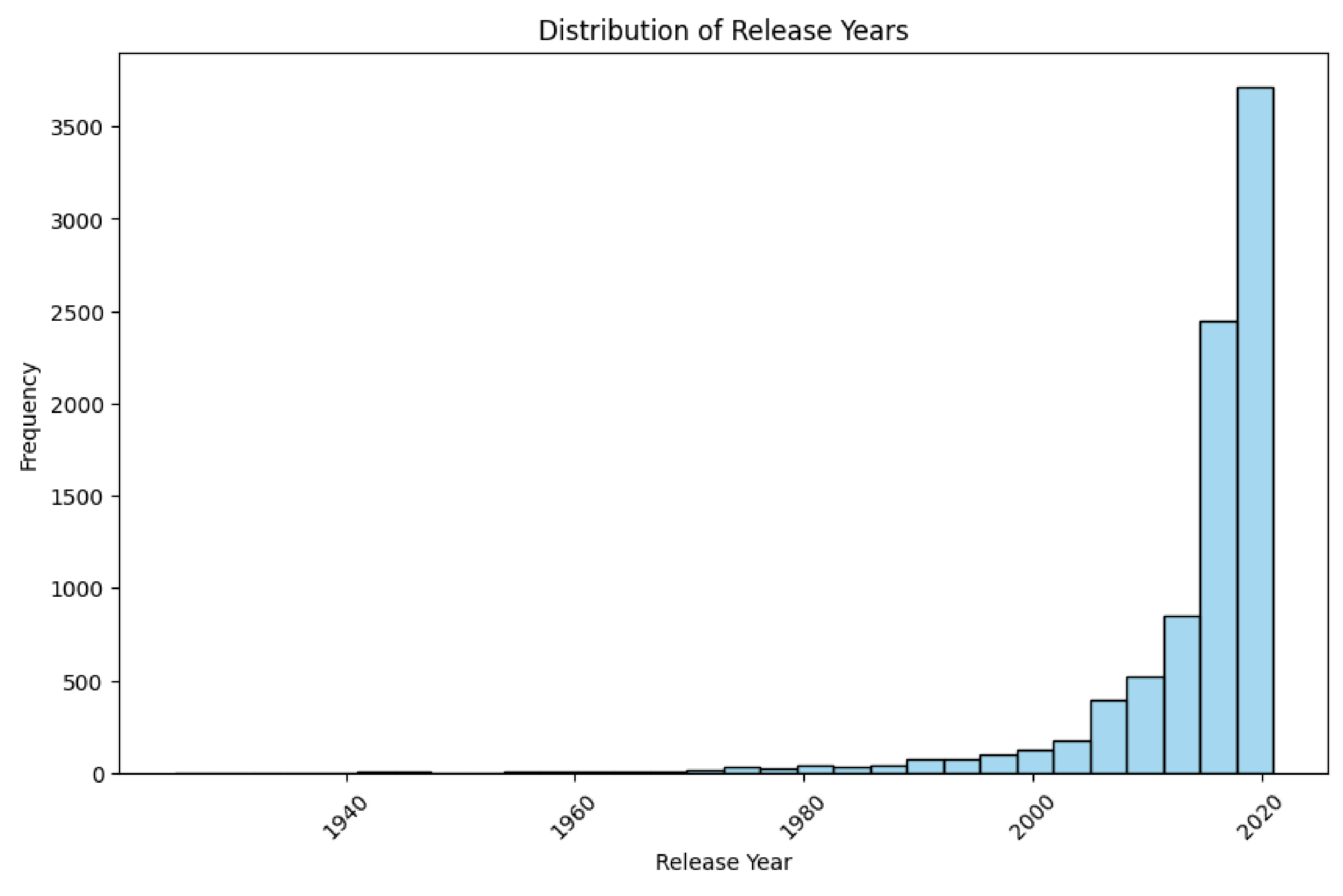
**The pandemic:** The COVID-19 pandemic may have caused some disruption to Netflix's movie production schedule in 2019 to 2021. But the increase in time spent at home likely contributed to the rise in Netflix movie releases. With more people looking for entertainment options at home, Netflix may have seen an opportunity to cater to this demand by releasing more content which has already finished post production work. By doing this they targeted to retain current users and increase new user subscription.

## Number of Titles released per year

```
titles_per_year = netflix_data['release_year'].value_counts().sort_index()

# Print the result
print("Number of Titles Released per Year:")
print(titles_per_year)
```

```
Number of Titles Released per Year:
release_year
1925       1
1942       2
1943       3
1944       3
1945       4
        ... 
2017    1016
2018    1140
2019    1030
2020     953
2021     592
Name: count, Length: 74, dtype: int64
```

```
fig, ax = plt.subplots(figsize=(10, 6))

# Plotting the histogram
sns.histplot(netflix_data['release_year'], bins=30, color='skyblue', ax=ax, edgecolor='black')

# Setting title and labels
ax.set_title('Distribution of Release Years')
ax.set_xlabel('Release Year')
ax.set_ylabel('Frequency')

# Rotate x-axis labels for better readability
plt.xticks(rotation=45)

# Show plot
plt.show()
```



## Insight

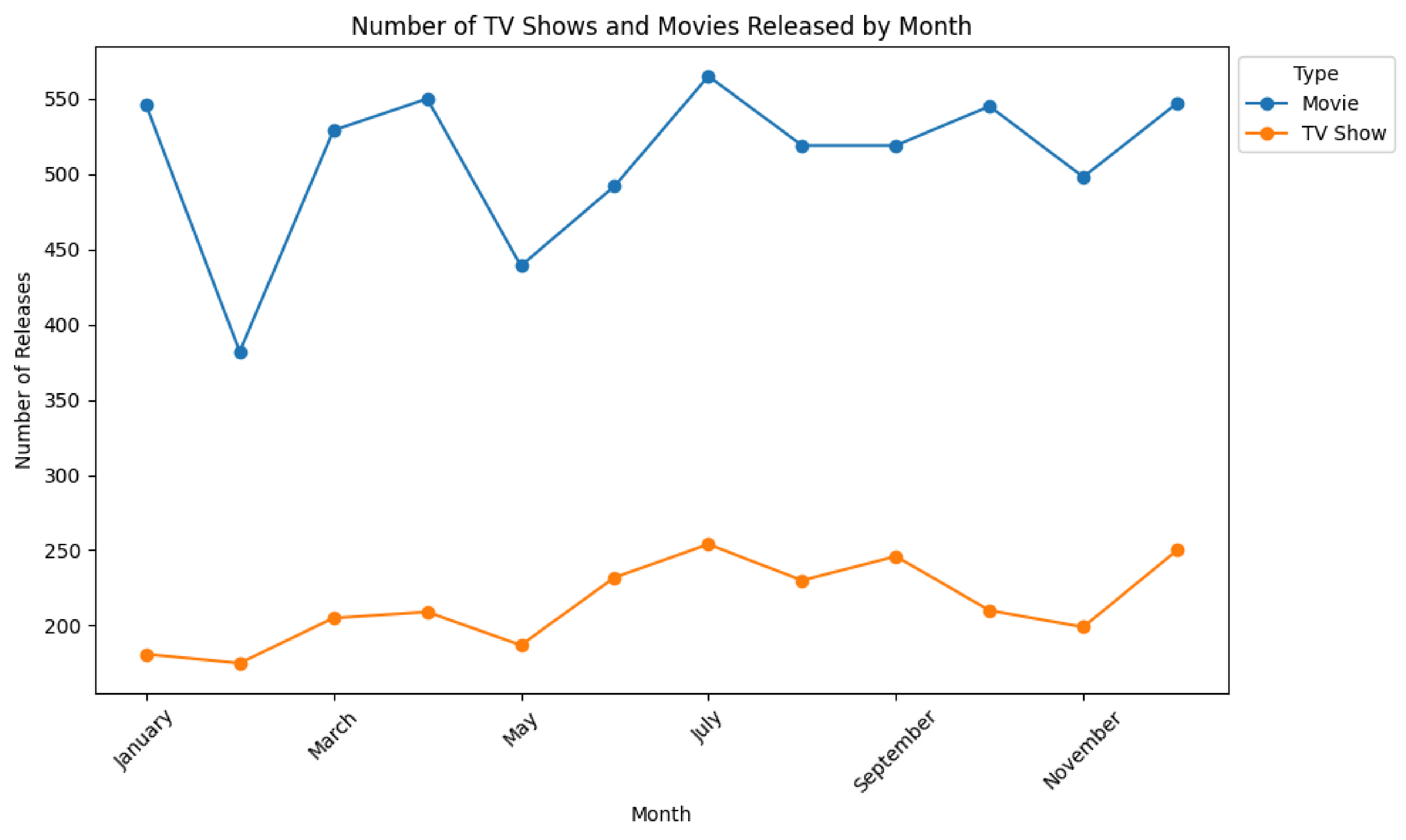
The distribution of release years on Netflix leans heavily towards the right, indicating a right skew. This means that most of the content available is relatively new, with a significant portion having been released in (2010-2022).

## 3. Best month to launch TV Show/Movie.

```
netflix_data['date_added'] = pd.to_datetime(netflix_data['date_added'])

# Extract the month from the 'date_added' column
netflix_data['month_added'] = netflix_data['date_added'].dt.month_name()

# Group by month and count the number of TV shows and movies added in each month
monthly_counts = netflix_data.groupby(['month_added', 'type']).size().unstack(fill_value=0)

# Reorder months
months_order = ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August', 'September', 'October', 'November', 'December']
monthly_counts = monthly_counts.reindex(months_order)

# Calculate the total number of TV shows and movies released in each month
monthly_total = monthly_counts.sum(axis=1)

# Find the month with the highest number of releases
best_month = monthly_total.idxmax()
max_releases = monthly_total.max()

# Print the result
print("Best Month to Launch a TV Show/Movie:")
print(f"{best_month} with {max_releases} releases")
```

```
Best Month to Launch a TV Show/Movie:
July with 819 releases
```

```
# Plotting the distribution of releases across different months
fig, ax = plt.subplots(figsize=(10, 6))

monthly_counts.plot(kind='line', ax=ax, marker='o')
ax.set_xlabel('Month')
ax.set_ylabel('Number of Releases')
ax.set_title('Number of TV Shows and Movies Released by Month')
ax.legend(title='Type', labels=['Movie', 'TV Show'], loc='upper left', bbox_to_anchor=(1, 1))

plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



## Insight

**Content Seasons:** There seems to be a pattern to when Netflix releases new movies and shows. They tend to add a lot of content around the holidays, especially in December and January. It's like a big present for viewers! You might also notice more content in summer (July and August) and around fall holidays (October).

**Content Flow:** Just like the weather, Netflix releases can vary throughout the year. Sometimes, there's a burst of new content, and other times, there might be a bit less. This might be because Netflix knows what kind of shows people like to watch at different times of year.

## Recommendations

**Targeted Releases:** Considering seasonal trends, plan strategic releases of high-demand genres during peak months (e.g., holiday comedies in December).

**Content Scheduling Optimization:** Schedule lower-demand releases during slower months (e.g., February) to balance viewership throughout the year.

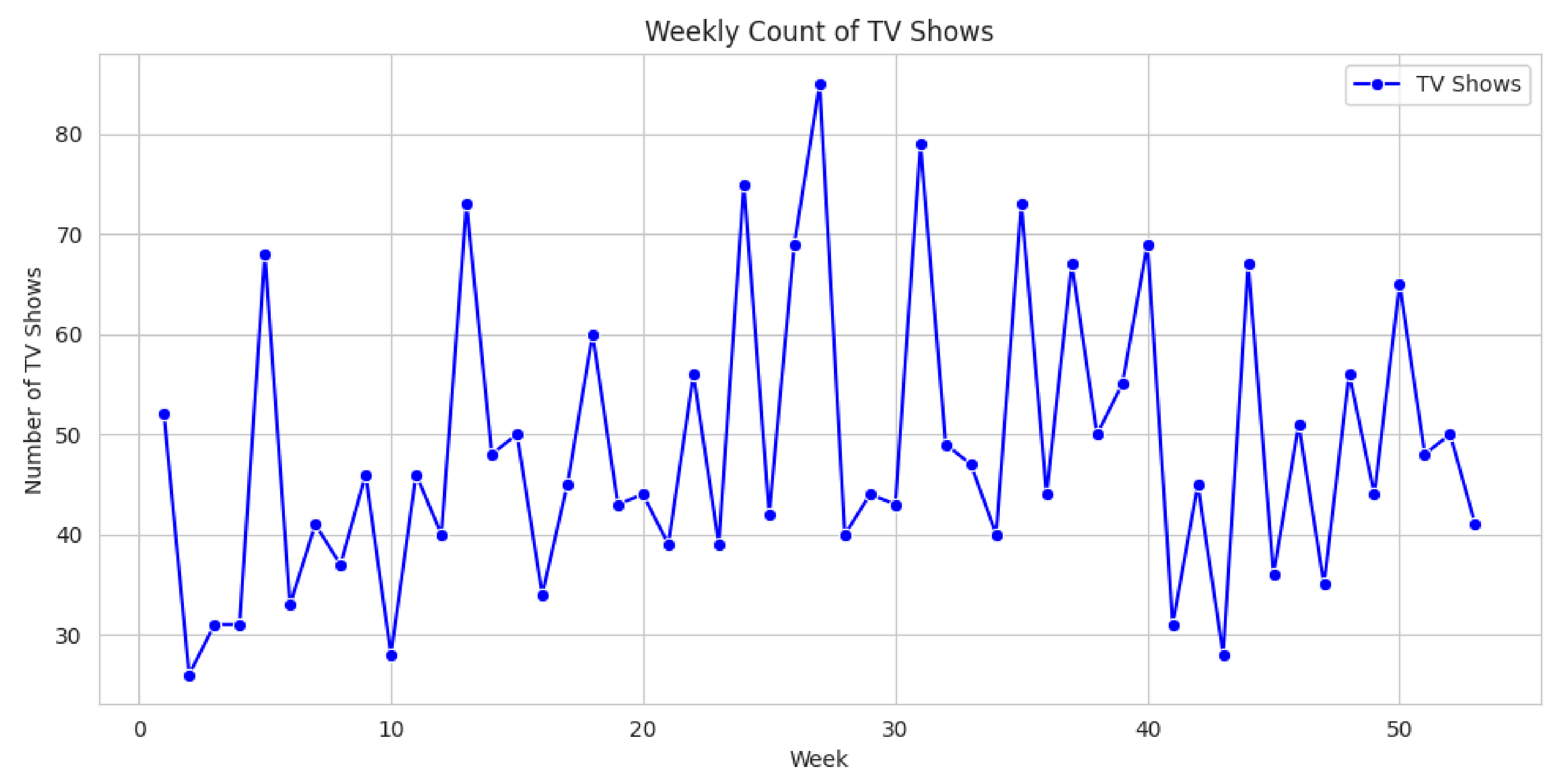
**Platform-Specific Strategies:** Tailor content releases and promotions to user behavior on different platforms. Global Content Strategy: Plan international content releases to capitalize on regional holidays and viewing preferences

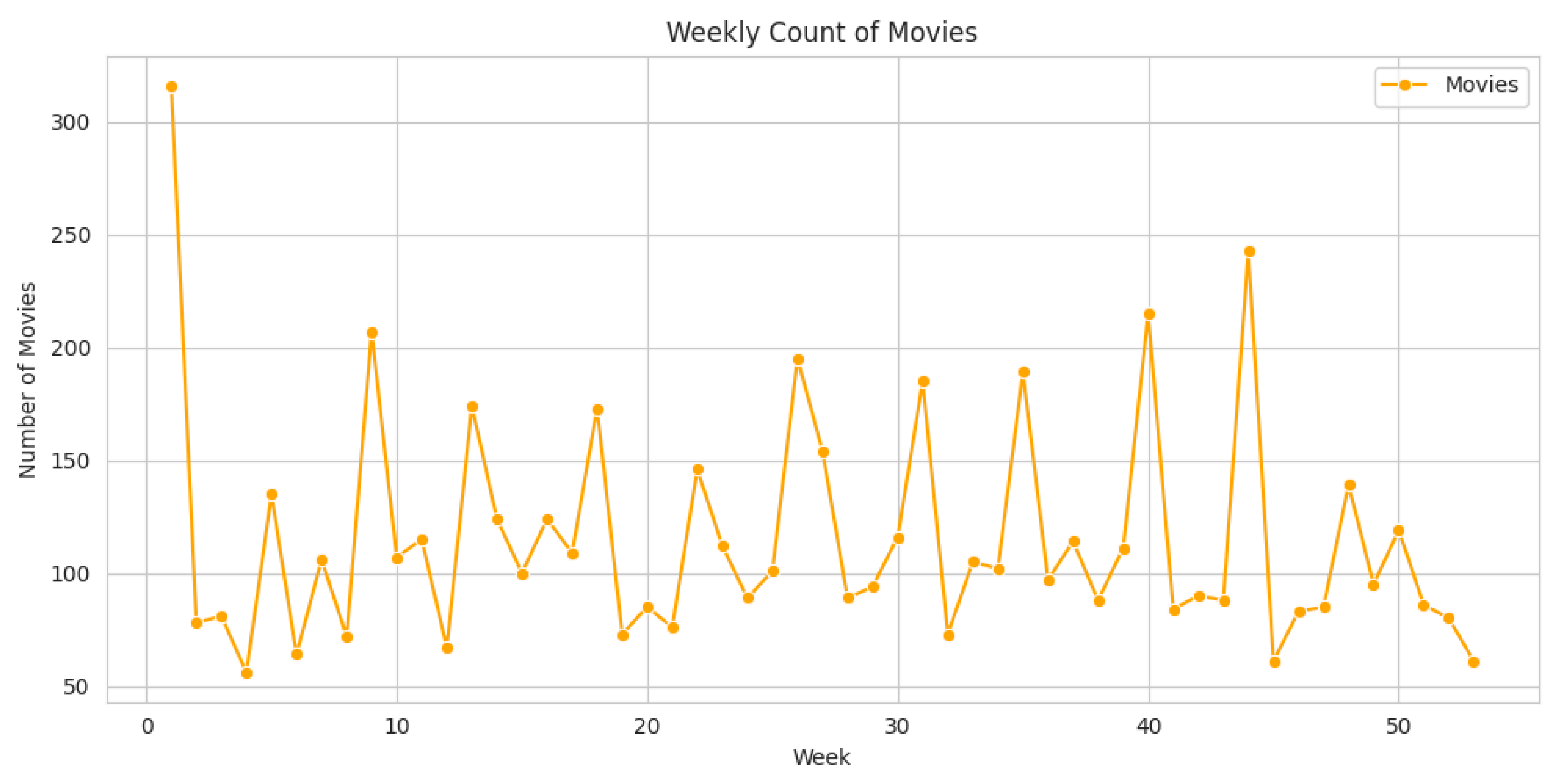
## Best week to launch TV Show/Movie.

```
netflix_data = netflix_data[netflix_data['date_added'] != 'Unknown Date_added']
netflix_data['date_added'] = pd.to_datetime(netflix_data['date_added'])

# Extract the week from the 'date_added' column
netflix_data['Week'] = netflix_data['date_added'].dt.isocalendar().week

# Filteration for TV shows and movies
tv_shows = netflix_data.query('type == "TV Show"')
movies = netflix_data.query('type == "Movie"')

# Counting the number of titles per week
tv_shows_weekly = tv_shows.groupby('Week')['title'].count()
movies_weekly = movies.groupby('Week')['title'].count()

# Finding the week with the highest count
best_tv_shows_week = tv_shows_weekly.idxmax()
best_tv_shows_count = tv_shows_weekly.max()
best_movies_week = movies_weekly.idxmax()
best_movies_count = movies_weekly.max()

# Print the result
print("Best Week to Launch a TV Show:")
print(f"Week {best_tv_shows_week} with {best_tv_shows_count} TV shows released")
print("\nBest Week to Launch a Movie:")
print(f"Week {best_movies_week} with {best_movies_count} movies released")
```

```
Best Week to Launch a TV Show:
Week 27 with 85 TV shows released

Best Week to Launch a Movie:
Week 1 with 316 movies released
```

```
sns.set_style('whitegrid')

# Creating subplots
fig, ax = plt.subplots(2, 1, figsize=(10, 10))

# Plotting TV shows
sns.lineplot(x=tv_shows_weekly.index, y=tv_shows_weekly.values, ax=ax[0], color='blue', marker='o', label='TV Shows')
ax[0].set_title('Weekly Count of TV Shows')
ax[0].set_xlabel('Week')
ax[0].set_ylabel('Number of TV Shows')

# Plotting movies
sns.lineplot(x=movies_weekly.index, y=movies_weekly.values, ax=ax[1], color='orange', marker='o', label='Movies')
ax[1].set_title('Weekly Count of Movies')
ax[1].set_xlabel('Week')
ax[1].set_ylabel('Number of Movies')

# Adding legend
ax[0].legend()
ax[1].legend()

plt.tight_layout()
plt.show()
```





### Insights

**Content Release Bursts:** Netflix seems to follow a pattern of releasing a bunch of new movies and shows all at once, followed by a few weeks with fewer releases. It's like they like to give us a big surprise every few weeks.

**TV Show Summer Surge:** New TV shows seem to get released more often around the end of June, perhaps to keep us entertained through the summer months.

### Recommendations

**Content Drops & Campaigns:** Creating "content drop" campaigns around these peak weeks, generating excitement with trailers, exclusive clips, and social media contests to boost viewership for the new releases.

**Optimization:** Recommending ways to personalize the Netflix recommendation engine based on user behavior during these peak content weeks. For example, if a user binges a new show during a release burst, the engine could prioritize surfacing similar content in the following weeks.

## 4. Analyse the Actors and Directors in TV Shows/Movies

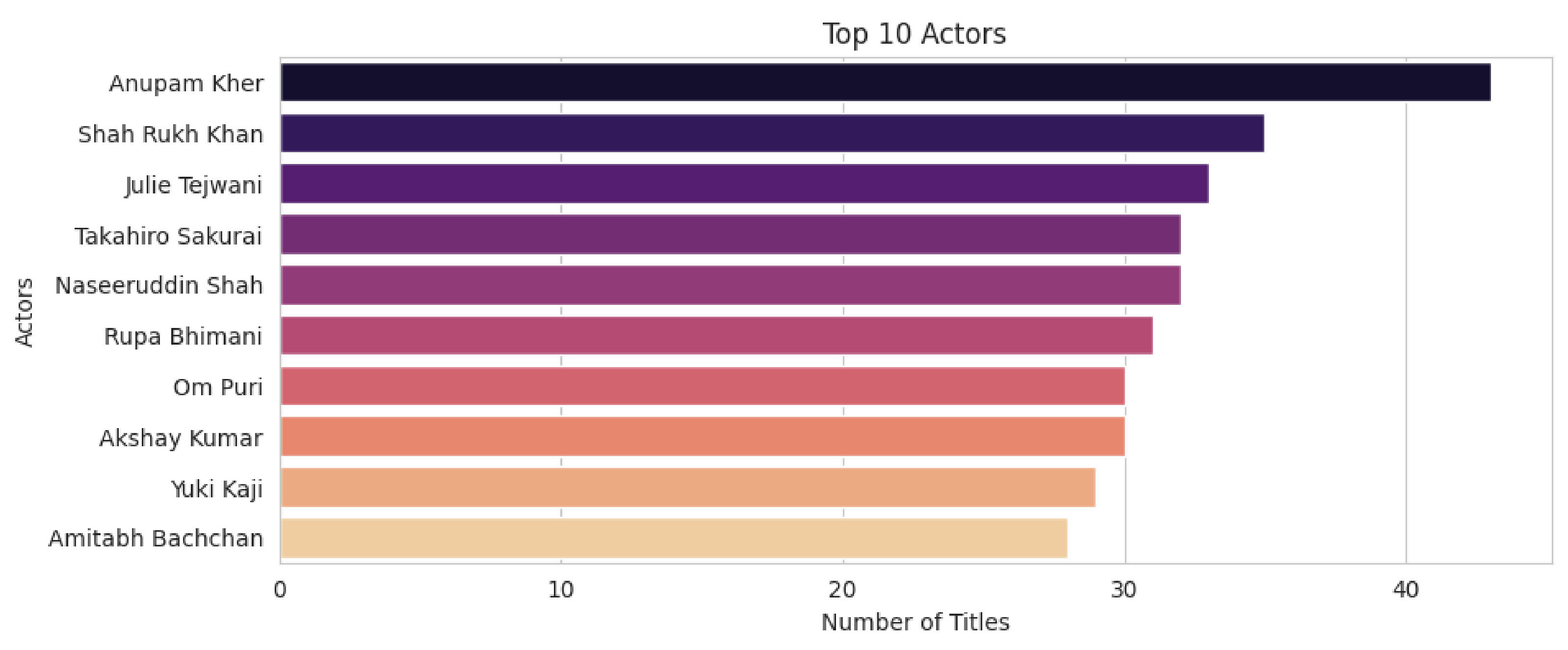
#### Top 10 Actor's who have appeared in most movies or TV shows.

```
un_cast['cast'] = un_cast['cast'].str.strip()
unique_cast_titles_count = un_cast.groupby('cast')['show_id'].nunique().sort_values(ascending=False).head(10)
```

```
top_10_actors = unique_cast_titles_count.head(10)

# Print the result
print("Top 10 Actors with Most Titles:")
print(top_10_actors)
```

```
Top 10 Actors with Most Titles:
cast
Anupam Kher         43
Shah Rukh Khan      35
Julie Tejwani       33
Takahiro Sakurai    32
Naseeruddin Shah    32
Rupa Bhimani        31
Om Puri             30
Akshay Kumar        30
Yuki Kaji           29
Amitabh Bachchan    28
Name: show_id, dtype: int64
```

```
fig, ax = plt.subplots(figsize=(10, 4))

# Plotting the bar plot
sns.barplot(x=unique_cast_titles_count.values, y=unique_cast_titles_count.index, hue=unique_cast_titles_count.index, palette='magma', ax=ax, dodge=False)

# Setting title and labels
ax.set_title('Top 10 Actors')
ax.set_xlabel('Number of Titles')
ax.set_ylabel('Actors')

plt.show()
```



## Insight

**Star Power:** Actors like Anupam Kher and Shah Rukh Khan have a large and loyal following, as shown by their frequent appearances in various projects.

**Global Appeal:** The presence of actors from diverse regions like Japan indicates that audiences enjoy content featuring international talent.

### Recommendations:

**Strategic Collaborations:** Partnering with popular actors like Anupam Kher and Shah Rukh Khan can leverage their fanbase and boost viewership.

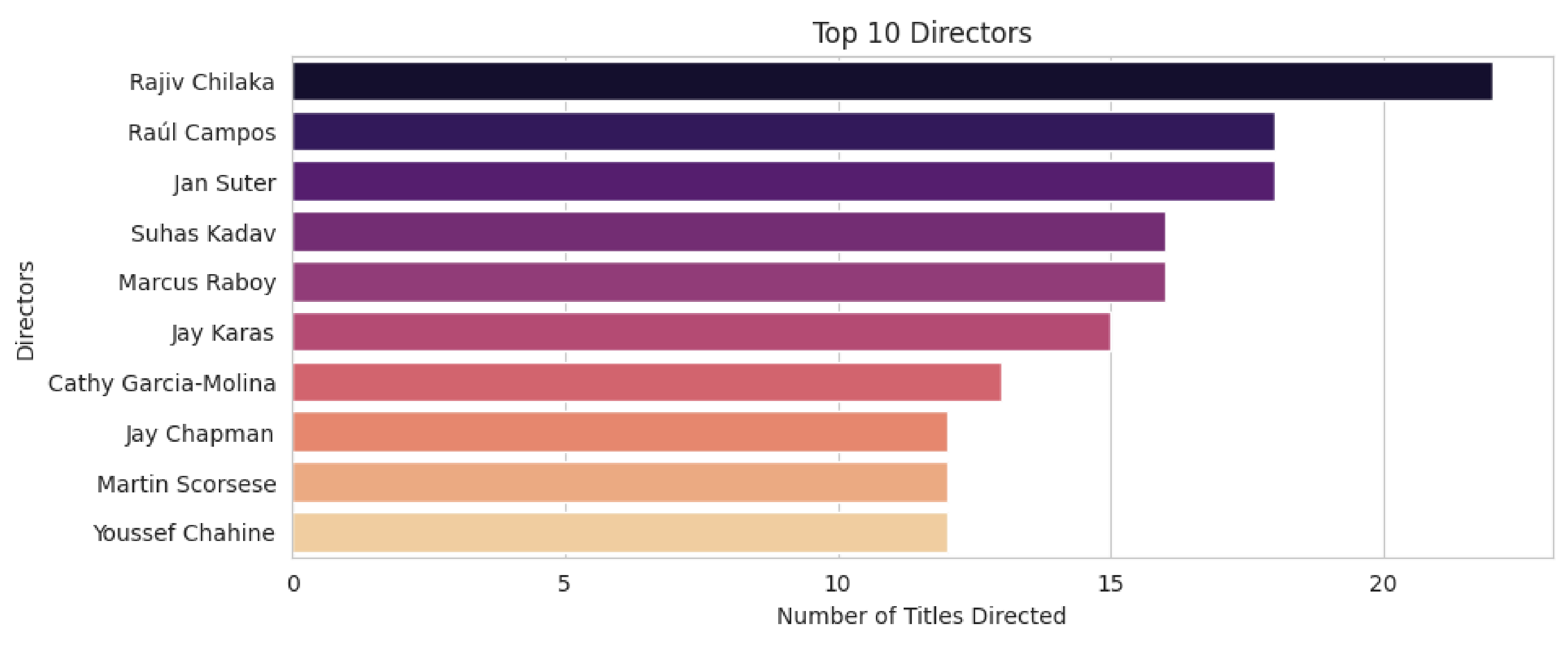
**Expand Your World:** Explore creating content with international actors and stories to broaden Netflix's global reach and appeal.

#### Top 10 Director's who have appeared in most movies or TV shows.

```
directors = unn_director.groupby('director')['title'].nunique().sort_values(ascending=False).head(10)
directors
```

```
director
Rajiv Chilaka          22
Raúl Campos            18
 Jan Suter             18
Suhas Kadav            16
Marcus Raboy           16
Jay Karas              15
Cathy Garcia-Molina    13
Jay Chapman            12
Martin Scorsese        12
Youssef Chahine        12
Name: title, dtype: int64
```

```
fig, ax = plt.subplots(figsize=(10, 4))

# Plotting the bar plot
sns.barplot(x=directors.values, y=directors.index, hue=directors.index, palette='magma', ax=ax, dodge=False)

# Setting title and labels
ax.set_title('Top 10 Directors')
ax.set_xlabel('Number of Titles Directed')
ax.set_ylabel('Directors')

plt.show()
```



## Insight

**Director Powerhouse:** Rajiv Chilaka, Raúl Campos, and Jan Suter are the top directors, consistently creating content for Netflix. This highlights their talent and the platform's trust in their vision.

**Global Vision:** Directors from various backgrounds and regions contribute to Netflix, showcasing their commitment to diverse storytelling.

## Recommendations:

**Fresh Voices:** Supporting directors like Suhas Kadav and Marcus Raboy indicates Netflix's openness to fresh talent. This fosters new perspectives within the platform's content.

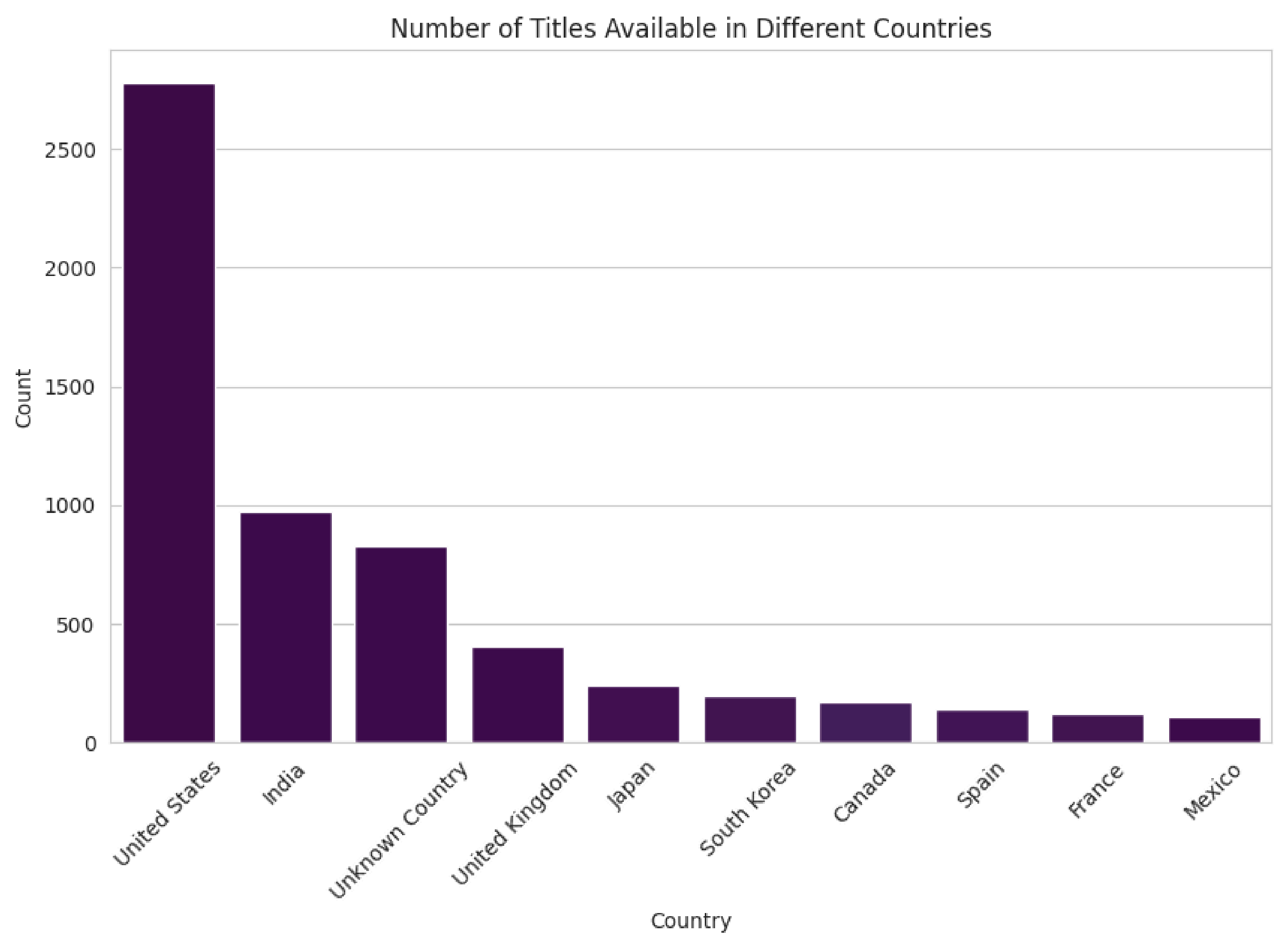
**Quality First:** Leverage the expertise of experienced directors like Martin Scorsese to deliver critically acclaimed content that attracts a broad audience.

## 5. Distribution of Audience For TV Shows/Movies in Netflix

```
country_counts = netflix_data['country'].value_counts().head(10)

print("Number of Titles Available in Different Countries (Top 10):")
print(country_counts)
```

```
Number of Titles Available in Different Countries (Top 10):
country
United States      2778
India               971
Unknown Country     827
United Kingdom      403
Japan               241
South Korea         195
Canada              173
Spain               141
France              122
Mexico              110
Name: count, dtype: int64
```

```
fig, ax = plt.subplots(figsize=(10, 6))

# Create a count plot with Seaborn using the specified axes
# Assign the `x` variable to `hue` and set `legend=False`
sns.countplot(data=netflix_data, x='country',
              order=netflix_data['country'].value_counts().head(10).index,
              palette='viridis', ax=ax, hue='country', legend=False)

# Customize the plot using the axes object
ax.set_title('Number of Titles Available in Different Countries')
ax.set_xlabel('Country')
ax.set_ylabel('Count')

# Rotate the x-tick labels
ax.tick_params(axis='x', rotation=45)

# Show the plot
plt.show()
```



## Insight

**Dominance of the United States:**The United States leads significantly with 2778 titles, indicating a strong focus on content from this country. This suggests that Netflix's library is heavily skewed towards American content, which may reflect both production and demand factors.

**Significant Presence of India:**India is the second highest with 971 titles. This indicates a substantial demand and supply of Indian content, which could be attributed to the large and diverse Indian audience both domestically and internationally.

## Recommendations

**Content Diversification:** While the US has a significant lead, increasing the diversity of content from other countries can attract a more global audience. Focus on acquiring more content from countries like Japan, South Korea, Spain, France, and Mexico.

**Localized Recommendations:** Enhance the algorithm to offer more localized recommendations based on a user's location and viewing history. This can be particularly beneficial in countries with diverse content preferences.

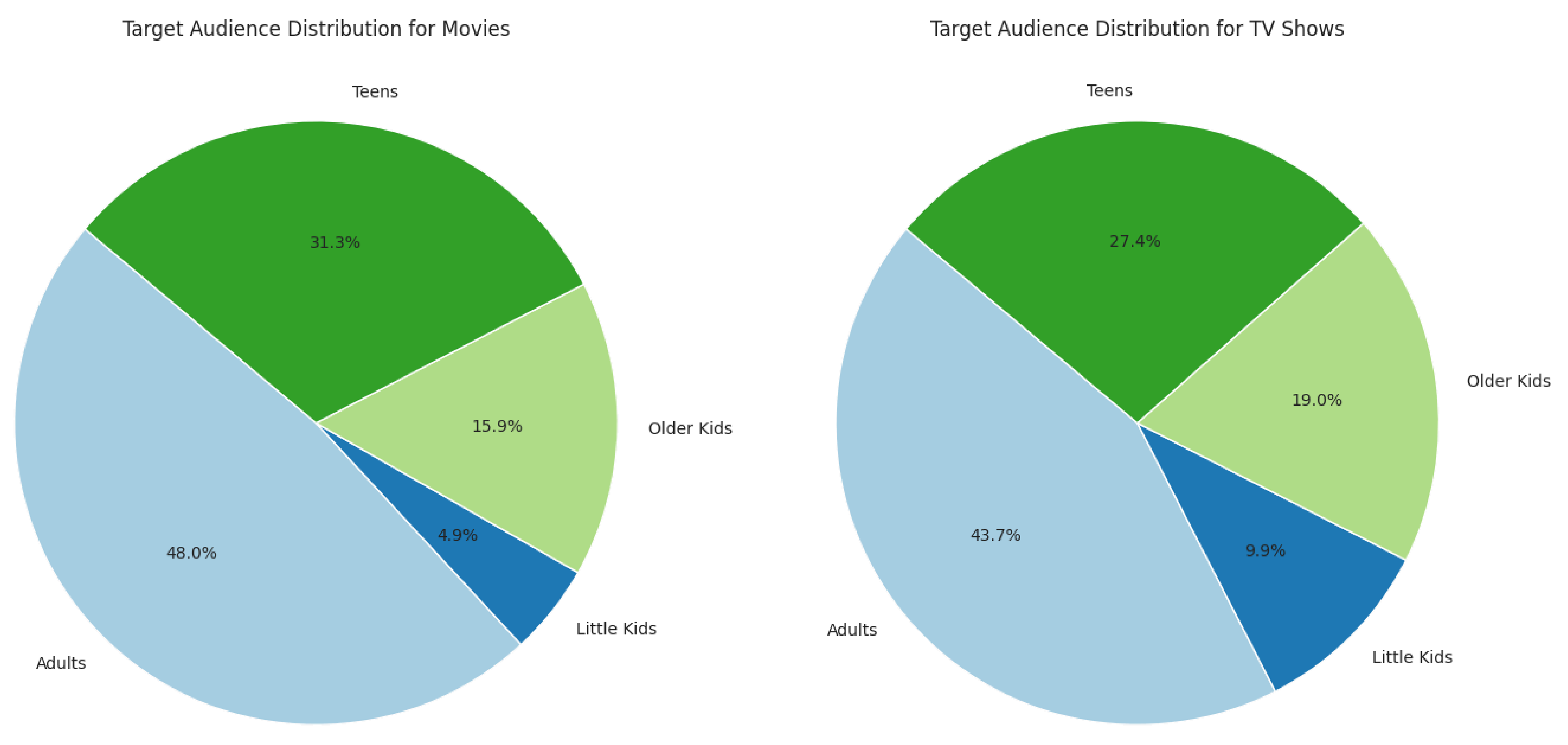
**Language Options:** Increase language options for subtitles and dubbing to make content more accessible to non-English speaking audiences.

## 6. Target Audience Distrubtion for TV Shows/ Movies

```
age_mapping = {
    'TV-Y': 'Little Kids',
    'TV-G': 'Little Kids',
    'G': 'Little Kids',
    'TV-Y7': 'Older Kids',
    'TV-Y7-FV': 'Older Kids',
    'PG': 'Older Kids',
    'TV-PG': 'Older Kids',
    'PG-13': 'Teens',
    'TV-14': 'Teens',
    'R': 'Adults',
    'TV-MA': 'Adults',
    'NC-17': 'Adults',
    'NR': 'Adults',
    'UR': 'Adults'
}

# Apply the mapping to create a new column for age group
netflix_data['age_group'] = netflix_data['rating'].map(age_mapping)

# Group by age group and type to count the number of titles
age_distribution = netflix_data.groupby(['age_group', 'type']).size().unstack(fill_value=0)

# Print the non-graphical analysis
print("Target Audience Distribution:")
print(age_distribution)
```

```
Target Audience Distribution:
type         Movie  TV Show
age_group
Adults        2943     1126
Little Kids    298      256
Older Kids     972      490
Teens         1918      706
```

```
fig, ax = plt.subplots(1, 2, figsize=(14, 7))

# Pie chart for Movies
ax[0].pie(age_distribution['Movie'], labels=age_distribution.index, autopct='%1.1f%%', startangle=140, colors=plt.cm.Paired(range(len(age_distribution))))
ax[0].set_title('Target Audience Distribution for Movies')

# Pie chart for TV Shows
ax[1].pie(age_distribution['TV Show'], labels=age_distribution.index, autopct='%1.1f%%', startangle=140, colors=plt.cm.Paired(range(len(age_distribution))))
ax[1].set_title('Target Audience Distribution for TV Shows')

plt.tight_layout()
plt.show()
```



### Insight

**Balanced Content Library:** Netflix offers a balanced content library with roughly half of its movies and TV shows targeting adult audiences.

**Strong Focus on Teens and Kids:** The platform caters significantly to younger demographics with 30% of content aimed at teenagers (movies) and children (TV shows) respectively. This is further emphasized by the presence of anime in the TV show category.

### Recommendations:

**Content Diversification:** Consider diversifying content offerings within adult and teen categories to cater to various tastes and preferences. For example, explore subgenres within these categories like documentaries, foreign films, or coming-of-age stories for teens.

**Anime Popularity:** The presence of anime highlights its popularity among viewers. Explore expanding the anime library strategically, possibly by region or genre, to keep existing fans engaged and attract new ones.

## 7. Popular Genres by Rating Categories

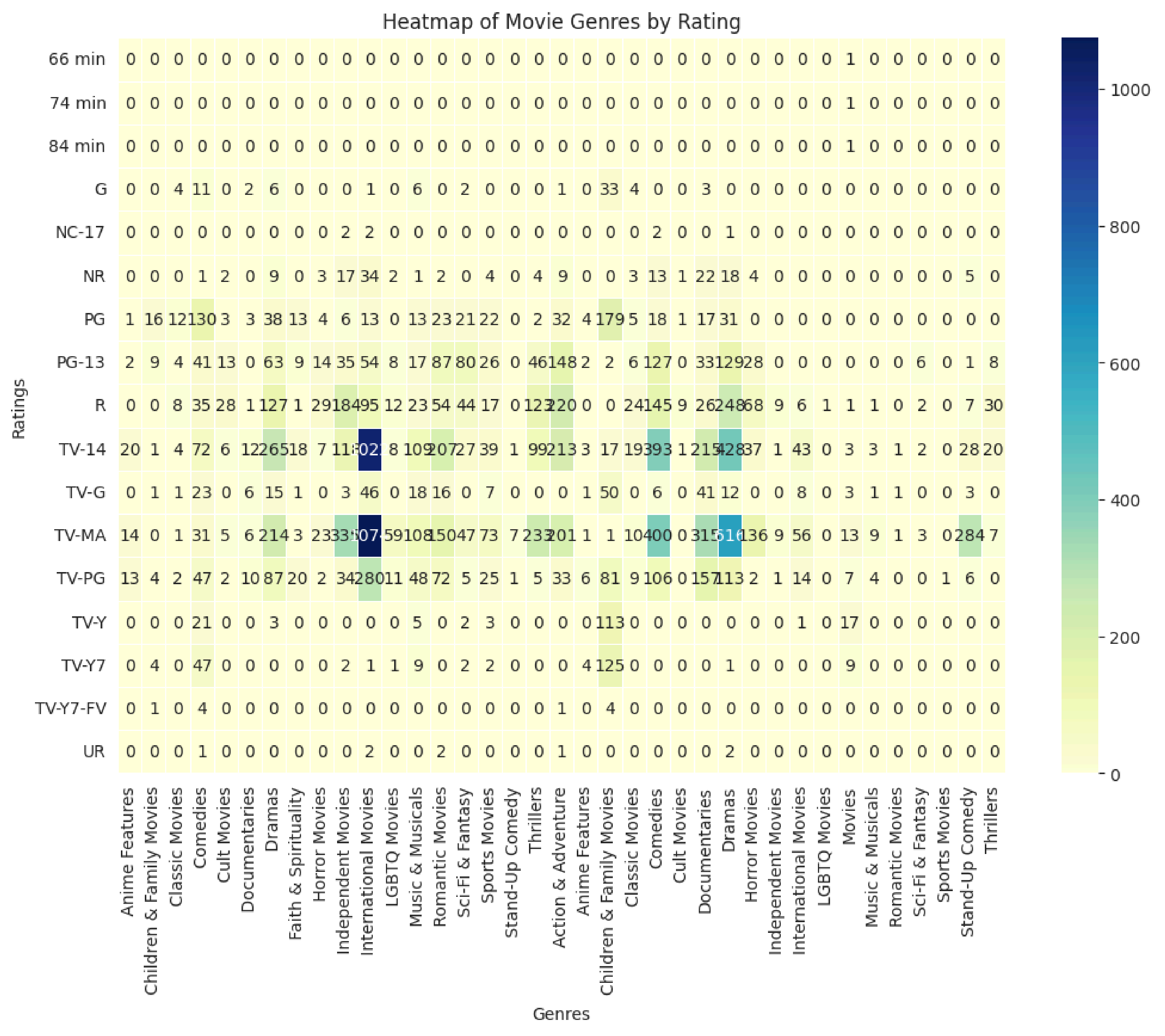
### Understanding which genres are popular within specific rating categories

```
movies_df = netflix_data[netflix_data['type'] == 'Movie']

movies_df = movies_df.copy()
movies_df['duration_numeric'] = movies_df['duration'].str.extract('(\d+)').astype(float)

# Unnest the 'listed_in' column to explore the relationship between movie ratings and genres
movies_genre_rating = unn_listed_in[unn_listed_in['type'] == 'Movie'].groupby(['rating', 'listed_in']).size().unstack(fill_value=0)

# Calculate the average duration for each rating category
average_duration_by_rating = movies_df.groupby('rating')['duration_numeric'].mean()

movies_genre_rating, average_duration_by_rating
```

```
(listed_in   Anime Features   Children & Family Movies   Classic Movies  \
 rating
 66 min                   0                          0                0
 74 min                   0                          0                0
 84 min                   0                          0                0
 G                        0                          0                4
 NC-17                    0                          0                0
 NR                       0                          0                0
 PG                       1                         16               12
 PG-13                    2                          9                4
 R                        0                          0                8
 TV-14                   20                          1                4
 TV-G                     0                          1                1
 TV-MA                   14                          0                1
 TV-PG                   13                          4                2
 TV-Y                     0                          0                0
 TV-Y7                    0                          4                0
 TV-Y7-FV                 0                          1                0
 UR                       0                          0                0

 listed_in   Comedies   Cult Movies   Documentaries   Dramas  \
 rating
 66 min             0             0               0        0
 74 min             0             0               0        0
 84 min             0             0               0        0
 G                 11             0               2        6
 NC-17              0             0               0        0
 NR                 1             2               0        9
 PG               130             3               3       38
 PG-13             41            13               0       63
 R                 35            28               1      127
 TV-14             72             6              12      265
 TV-G              23             0               6       15
 TV-MA             31             5               6      214
 TV-PG             47             2              10       87
 TV-Y              21             0               0        3
 TV-Y7             47             0               0        0
 TV-Y7-FV           4             0               0        0
```

```
UR                  1             0                0        0

listed_in   Faith & Spirituality   Horror Movies   Independent Movies  ...  \
rating                                                                 ...
66 min                         0               0                    0  ...
74 min                         0               0                    0  ...
84 min                         0               0                    0  ...
G                              0               0                    0  ...
NC-17                          0               0                    2  ...
NR                             0               3                   17  ...
PG                            13               4                    6  ...
PG-13                          9              14                   35  ...
R                              1              29                  184  ...
TV-14                         18               7                  118  ...
TV-G                           1               0                    3  ...
TV-MA                          3              23                  335  ...
TV-PG                         20               2                   34  ...
TV-Y                           0               0                    0  ...
TV-Y7                          0               0                    2  ...
```

```
fig, ax = plt.subplots(figsize=(12, 8))
sns.heatmap(movies_genre_rating, cmap='YlGnBu', annot=True, fmt='d', linewidths=.5, ax=ax)

# Setting title and labels
ax.set_title('Heatmap of Movie Genres by Rating')
ax.set_xlabel('Genres')
ax.set_ylabel('Ratings')

# Display the plot
plt.show()
```



## Insight

**Popular Genres:** Thrillers (R) and Stand-Up Comedy (TV-MA) have the highest number of movies across different ratings. This indicates that these genres are popular among audiences.Sci-Fi & Fantasy (PG-13) also have a significant number of movies, suggesting a strong interest in this genre among viewers.

**Average Duration:** The average duration varies across different ratings. For instance, NC-17 and R-rated movies have longer durations compared to other ratings, suggesting potentially more complex or mature content.

## Recommendations:

**Target Audience Preferences:** Tailor content duration and themes based on the target audience's preferences and age suitability. For example, if targeting younger audiences, shorter durations and content with TV-Y or TV-G ratings might be more appropriate.

**Content Development:** Explore opportunities to create more content in popular genres like Thrillers and Sci-Fi & Fantasy, especially considering their high demand across various ratings.

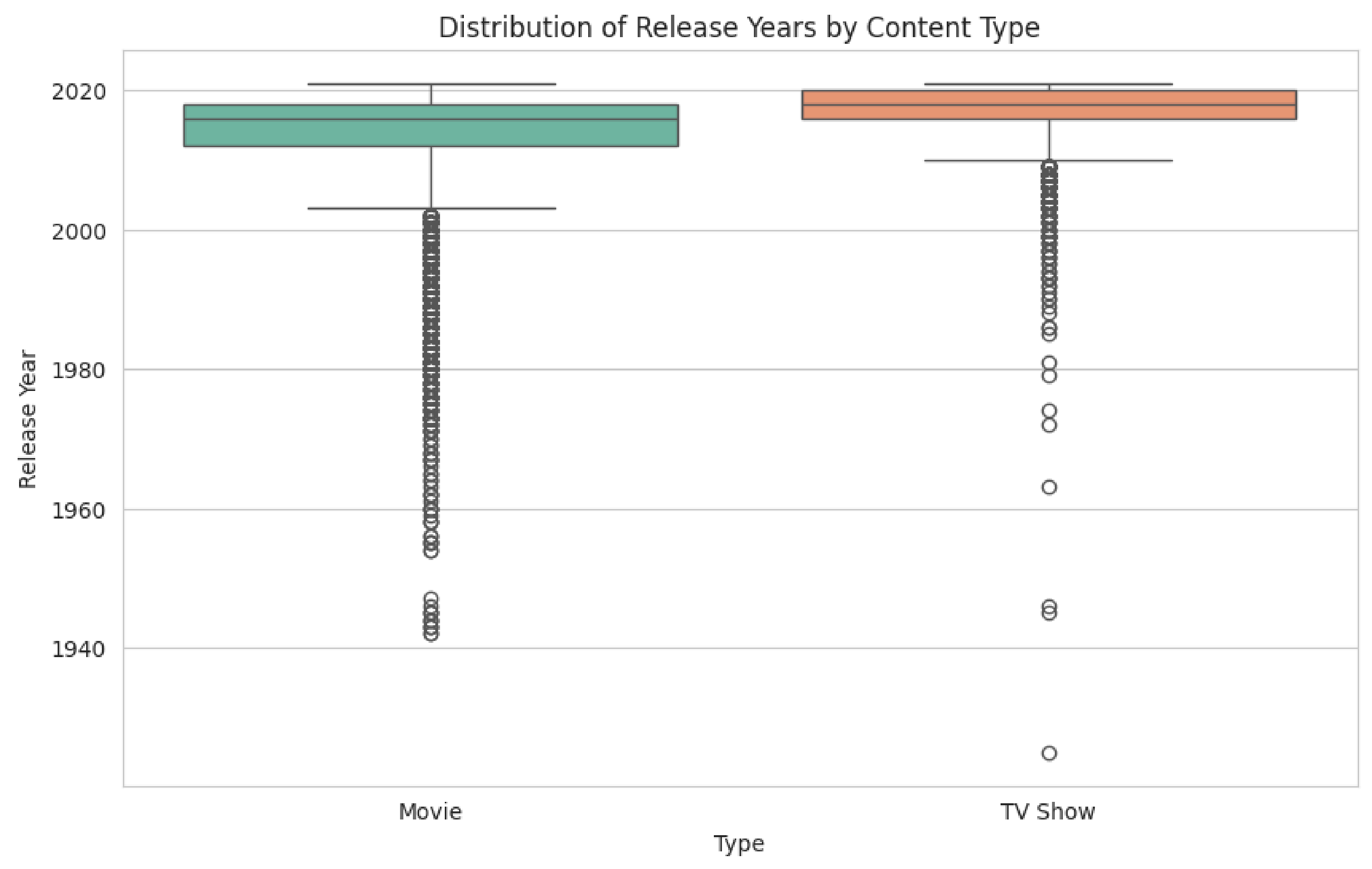
## 8. Distribution of release years for Movies and TV shows

```
release_year_stats_by_type = netflix_data.groupby('type')['release_year'].describe()

print(release_year_stats_by_type)
```

```
          count         mean       std     min     25%     50%     75%     max
type
Movie    6131.0  2013.121514  9.678169  1942.0  2012.0  2016.0  2018.0  2021.0
TV Show  2578.0  2016.754849  5.579880  1925.0  2016.0  2018.0  2020.0  2021.0
```

```
fig, ax = plt.subplots(figsize=(10, 6))

# Plot the boxplot
sns.boxplot(x='type', y='release_year', data=netflix_data, palette='Set2', ax=ax)

# Set the title and labels
ax.set_title('Distribution of Release Years by Content Type')
ax.set_xlabel('Type')
ax.set_ylabel('Release Year')

plt.show()
```

```
<ipython-input-40-a18644927891>:4: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.boxplot(x='type', y='release_year', data=netflix_data, palette='Set2', ax=ax)
```



## Insight

**Mean Release Year:**

Movies: The average release year for movies is 2013.12, indicating that the majority of movies in the dataset are relatively recent, with an emphasis on the past decade.

TV Shows: The average release year for TV shows is 2016.75, suggesting that TV shows are, on average, newer than movies.

**Median Release Year:**

Movies: The median release year is 2016, indicating that half of the movies were released before this year and half after.

TV Shows: The median release year is 2018, indicating a more recent focus in the TV shows category.

**Spread and Variability:**

Movies: The standard deviation is 9.68, showing a wider spread in the release years of movies. The minimum release year is 1942, indicating a significant historical range.

TV Shows: The standard deviation is 5.58, indicating less variability in the release years of TV shows compared to movies. The minimum release year is 1925, but the first quartile (25%) starts from 2016, showing a more recent collection.

**Quartiles:**

Movies: The interquartile range (IQR) for movies is from 2012 to 2018, indicating that the central 50% of movie release years are within this range.

TV Shows: The IQR for TV shows is from 2016 to 2020, showing that the central 50% of TV show release years are within a more recent range compared to movies.

**Maximum Release Year:**

Both movies and TV shows have a maximum release year of 2021, indicating the dataset includes very recent content up to the year 2021.