Introduction

About NETFLIX: Netflix is one of the most popular media and video streaming platforms. They have over 10000 movies or tv shows available on their platform, as of mid-2021, they have over 222M Subscribers globally. This tabular dataset consists of listings of all the movies and tv shows available on Netflix, along with details such as - cast, directors, ratings, release year, duration, etc.

Netflix Data Analysis with Python

Data Preparation

In [42]:
```
# Importing necessary libraries for data analysis and visualization
import pandas as pd  # pandas for data manipulation and analysis
import numpy as np  # numpy for numerical operations
import matplotlib.pyplot as plt  # matplotlib for data visualization
import seaborn as sns  # seaborn for enhanced data visualization
```

In [43]:
```
# Loading the dataset from a CSV file
df = pd.read_csv('https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/000/940/original/netflix.csv')

# Displaying the first few rows of the dataset
df.head()
```

Out[43]:

	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	NaN	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	NaN	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...	NaN	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	NaN	NaN	NaN	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	NaN	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...

In [44]:
```
# Computing descriptive statistics for the dataset
df.describe()
```

Out[44]:

	release_year
count	8807.000000
mean	2014.180198
std	8.819312
min	1925.000000
25%	2013.000000
50%	2017.000000
75%	2019.000000
max	2021.000000

In [45]:
```
# Obtaining information about the dataset
df.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
```

In [88]:
```
# Checking for column names in the dataset
df.columns
```

Out[88]:
```
Index(['show_id', 'type', 'title', 'director', 'cast', 'country', 'date_added',
       'release_year', 'rating', 'duration', 'listed_in', 'description',
       'month_added', 'month_name_added', 'week_added', 'year_added'],
      dtype='object')
```

In [46]:
```
# Checking for missing values in the dataset
df.isnull().sum()
```

Out[46]:
```
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
```

In [47]:
```
# Checking for duplicate rows in the dataset
df.duplicated().sum()
```

Out[47]:
```
0
```

In [48]:
```
# Checking for shape of the dataset
df.shape
```

Out[48]:
```
(8807, 12)
```

1. Handling null values

a. For categorical variables with null values, updating those rows as unknown_column_name.

b. Replacing with 0 for continuous variables having null values.

In [49]:
```
# Replacing missing values in the 'director' column with 'No Data'
df['director'].replace(np.nan, 'Unknown Director', inplace=True)
```

```
# Replacing missing values in the 'cast' column with 'No Data'
df['cast'].replace(np.nan, 'Unknown Cast', inplace=True)
```

In [50]:
```
# Filling missing values in the 'country' column with the mode value
df['country'] = df['country'].fillna(df['country'].mode()[0])
```

In [51]:
```
# Finding the mode rating for movies and TV shows
movie_rating = df.loc[df['type'] == 'Movie', 'rating'].mode()[0]
tv_rating = df.loc[df['type'] == 'TV Show', 'rating'].mode()[0]

# Filling missing rating values based on the type of content
df['rating'] = df.apply(lambda x: movie_rating if x['type'] == 'Movie' and pd.isna(x['rating'])
                        else tv_rating if x['type'] == 'TV Show' and pd.isna(x['rating'])
                        else x['rating'], axis=1)
```

In [52]:
```
# Finding the mode duration for movies and TV shows
movie_duration_mode = df.loc[df['type'] == 'Movie', 'duration'].mode()[0]
tv_duration_mode = df.loc[df['type'] == 'TV Show', 'duration'].mode()[0]

# Filling missing duration values based on the type of content
df['duration'] = df.apply(lambda x: movie_duration_mode if x['type'] == 'Movie'
                          and pd.isna(x['duration'])
                          else tv_duration_mode if x['type'] == 'TV Show'
                          and pd.isna(x['duration'])
                          else x['duration'], axis=1)
```

In [53]:
```
# Dropping rows with missing values
df.dropna(inplace=True)
```

In [54]:
```
# Converting the 'date_added' column to datetime format
df["date_added"] = pd.to_datetime(df['date_added'])
```

In [57]:
```
# Extracting month, month name,week and year from the 'date_added' column
df['month_added'] = df['date_added'].dt.month
df['month_name_added'] = df['date_added'].dt.month_name()
df['week_added'] = df['date_added'].dt.isocalendar().week
df['year_added'] = df['date_added'].dt.year
```

2. Un-nesting the columns

a. Un-nesting the columns those have cells with multiple comma separated values by creating multiple rows

In [58]:
```
# Splitting and expanding the 'cast' column
df_cast = df['cast'].str.split(',', expand=True).stack()
df_cast = df_cast.reset_index(level=1, drop=True).to_frame('cast')
df_cast['show_id'] = df['show_id']

# Splitting and expanding the 'country' column
df_country = df['country'].str.split(',', expand=True).stack()
df_country = df_country.reset_index(level=1, drop=True).to_frame('country')
df_country['show_id'] = df['show_id']

# Splitting and expanding the 'listed_in' column
df_listed_in = df['listed_in'].str.split(',', expand=True).stack()
df_listed_in = df_listed_in.reset_index(level=1, drop=True).to_frame('listed_in')
df_listed_in['show_id'] = df['show_id']

# Splitting and expanding the 'director' column
df_director = df['director'].str.split(',', expand=True).stack()
df_director = df_director.reset_index(level=1, drop=True).to_frame('director')
df_director['show_id'] = df['show_id']
```

b. Non-Graphical Analysis: Value counts and unique attributes

In [59]:
```
df.nunique()
```

Out[59]:
```
show_id             8797
type                   2
title               8797
director            4529
cast                7683
country              748
date_added          1714
release_year          74
rating                17
duration             220
listed_in            513
description         8765
month_added           12
month_name_added      12
week_added            53
year_added            14
dtype: int64
```

3. Find the counts of each categorical variable both using graphical and non-graphical analysis.

a. For Non-graphical Analysis:

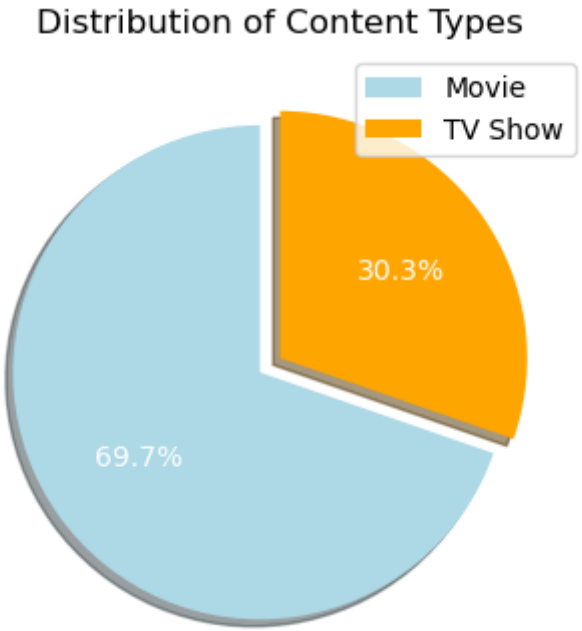
b. For graphical analysis:

In [60]:
```
round(df["type"].value_counts()/len(df)*100,2)
```

Out[60]:
```
Movie      69.69
TV Show    30.31
Name: type, dtype: float64
```

In [61]:
```
# Calculate the percentage distribution of content types
x = df.groupby(['type'])['type'].count()
y = len(df)
r = ((x/y) * 100).round(2)

# Create a DataFrame to store the percentage distribution
mf_ratio = pd.DataFrame(r)
mf_ratio.rename({'type': '%'}, axis=1, inplace=True)
```

```
# Plot the 3D-effect pie chart
plt.figure(figsize=(8, 4))
colors = ['lightblue', 'orange']
explode = (0.1, 0)
plt.pie(mf_ratio['%'], labels=mf_ratio.index, autopct='%1.1f%%',
colors=colors, explode=explode, shadow=True, startangle=90,
textprops={'color': 'white'})

plt.legend(loc='upper right')
plt.title('Distribution of Content Types')
plt.show()
```



4. Comparison of tv shows vs. movies.

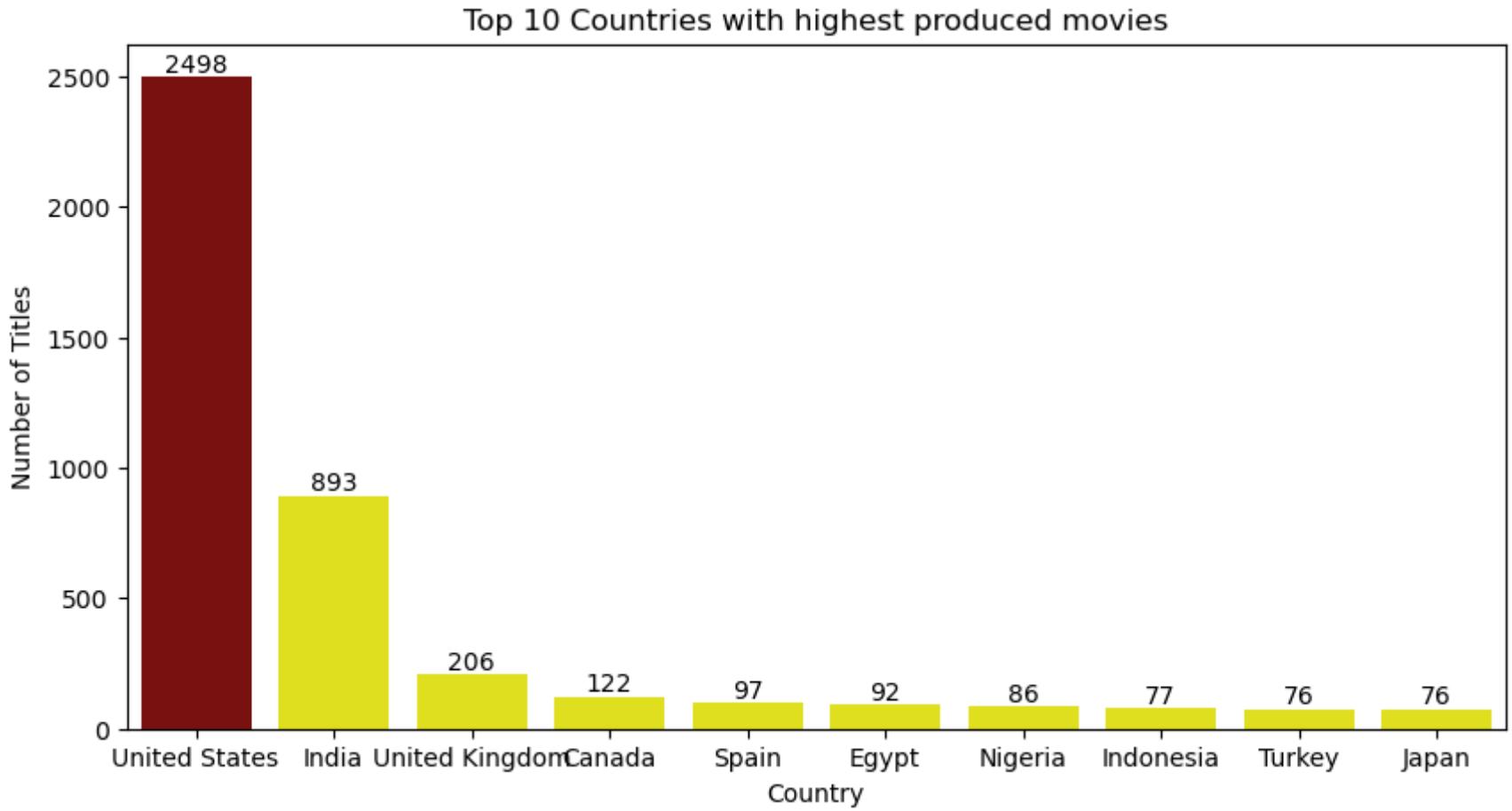
a. Find the number of movies produced in each country and pick the top 10 countries.

b. Find the number of Tv-Shows produced in each country and pick the top 10 countries.

In [62]:
```
df1=df.loc[(df["type"]=="Movie")]
df1.groupby("country")["title"].count().sort_values(ascending=False).head(10)
```

Out[62]:
```
country
United States     2498
India              893
United Kingdom     206
Canada             122
Spain               97
Egypt               92
Nigeria             86
Indonesia           77
Turkey              76
Japan               76
Name: title, dtype: int64
```

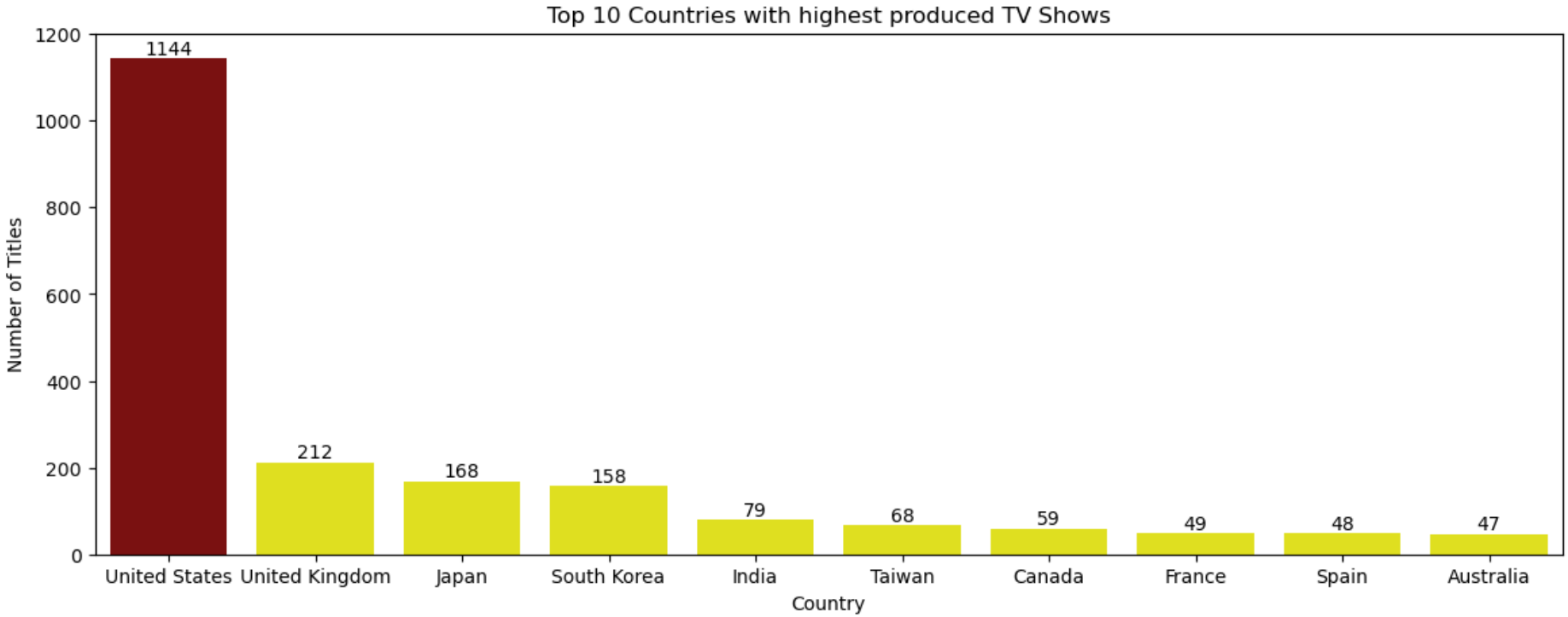
In [63]:
```
df1=df.loc[(df["type"]=="Movie")]
top_10_countries=df1.groupby("country")["title"].count().sort_values(ascending=False).head(10)

# Plot the top 10 countries
plt.figure(figsize=(10, 5))
colors = ['darkred'] + ['yellow'] * (len(top_10_countries) - 1)
bar_plot = sns.barplot(x=top_10_countries.index, y=top_10_countries.values, palette=colors)

plt.xlabel('Country')
plt.ylabel('Number of Titles')
plt.title('Top 10 Countries with highest produced movies')

# Add count values on top of each bar
for index, value in enumerate(top_10_countries.values):
    bar_plot.text(index, value, str(value), ha='center', va='bottom')

plt.show()
```



In [64]:
```
df2=df.loc[(df["type"]=="TV Show")]
df2.groupby("country")["title"].count().sort_values(ascending=False).head(10)
```

Out[64]:
```
country
United States     1144
United Kingdom     212
Japan              168
South Korea        158
India               79
Taiwan              68
Canada              59
France              49
Spain               48
Australia           47
Name: title, dtype: int64
```

In [93]:
```
df2=df.loc[(df["type"]=="TV Show")]
top_10_countries=df2.groupby("country")["title"].count().sort_values(ascending=False).head(10)

# Plot the top 10 countries
plt.figure(figsize=(14, 5))
colors = ['darkred'] + ['yellow'] * (len(top_10_countries) - 1)
bar_plot = sns.barplot(x=top_10_countries.index, y=top_10_countries.values, palette=colors)

plt.xlabel('Country')
plt.ylabel('Number of Titles')
plt.title('Top 10 Countries with highest produced TV Shows')

# Add count values on top of each bar
for index, value in enumerate(top_10_countries.values):
    bar_plot.text(index, value, str(value), ha='center', va='bottom')

plt.show()
```



5. What is the best time to launch a TV show?

a. Find which is the best week to release the Tv-show or the movie. Do the analysis separately for Tv-shows and Movies

b. Find which is the best month to release the Tv-show or the movie. Do the analysis separately for Tv-shows and Movies

In [66]:
```
df
```

Out[66]:

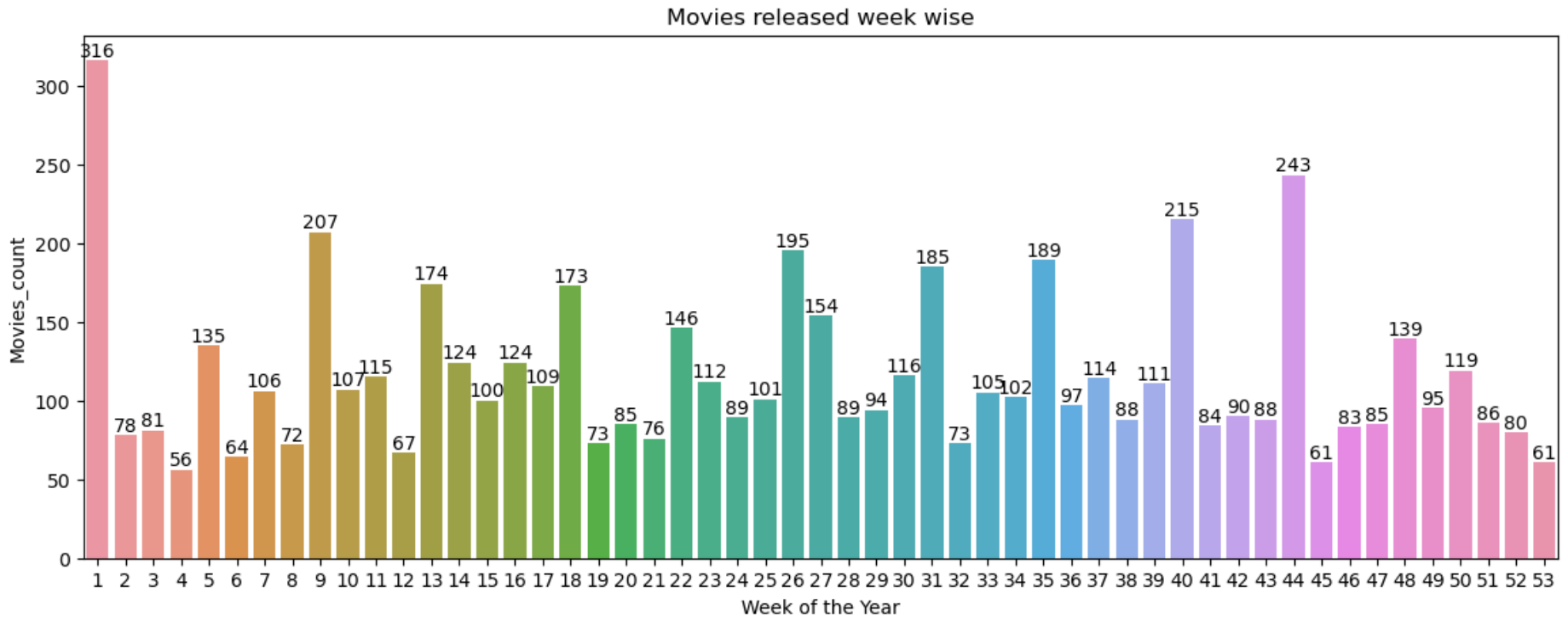
	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	month_added
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	Unknown Cast	United States	2021-09-25	2020	PG-13	90 min	Documentaries	As her father nears the end of his life, filmm...	9
1	s2	TV Show	Blood & Water	Unknown Director	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	2021-09-24	2021	TV-MA	2 Seasons	International TV Shows, TV Dramas, TV Mysteries	After crossing paths at a party, a Cape Town t...	9
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	United States	2021-09-24	2021	TV-MA	1 Season	Crime TV Shows, International TV Shows, TV Act...	To protect his family from a powerful drug lor...	9
3	s4	TV Show	Jailbirds New Orleans	Unknown Director	Unknown Cast	United States	2021-09-24	2021	TV-MA	1 Season	Docuseries, Reality TV	Feuds, flirtations and toilet talk go down amo...	9
4	s5	TV Show	Kota Factory	Unknown Director	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	2021-09-24	2021	TV-MA	2 Seasons	International TV Shows, Romantic TV Shows, TV ...	In a city of coaching centers known to train I...	9
...	...	...	...	...	...	...	...	...	...	...	...	...	...
8802	s8803	Movie	Zodiac	David Fincher	Mark Ruffalo, Jake Gyllenhaal, Robert	United States	2019-11-20	2007	R	158 min	Cult Movies, Dramas, Thrillers	A political cartoonist, a crime reporter and a...	11

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	month_added
					Downey J...								
8803	s8804	TV Show	Zombie Dumb	Unknown Director	Unknown Cast	United States	2019-07-01	2018	TV-Y7	2 Seasons	Kids' TV, Korean TV Shows, TV Comedies	While living alone in a spooky town, a young g...	7
8804	s8805	Movie	Zombieland	Ruben Fleischer	Jesse Eisenberg, Woody Harrelson, Emma Stone, ...	United States	2019-11-01	2009	R	88 min	Comedies, Horror Movies	Looking to survive in a world taken over by zo...	11
8805	s8806	Movie	Zoom	Peter Hewitt	Tim Allen, Courteney Cox, Chevy Chase, Kate Ma...	United States	2020-01-11	2006	PG	88 min	Children & Family Movies, Comedies	Dragged from civilian life, a former superhero...	1
8806	s8807	Movie	Zubaan	Mozez Singh	Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanan...	India	2019-03-02	2015	TV-14	111 min	Dramas, International Movies, Music & Musicals	A scrappy but poor boy worms his way into a ty...	3

8797 rows × 16 columns

In [68]:
```
df_movies = df[df['type'] == 'Movie']
movies_count = df_movies['week_added'].value_counts().sort_index()
# movies_count = movies_count.sort_values(ascending=False)

plt.figure(figsize=(14, 5))
bar_plot = sns.barplot(x=movies_count.index, y=movies_count.values)

plt.xlabel('Week of the Year')
plt.ylabel('Movies_count')
plt.title('Movies released week wise')

for index, value in enumerate(movies_count.values):
```

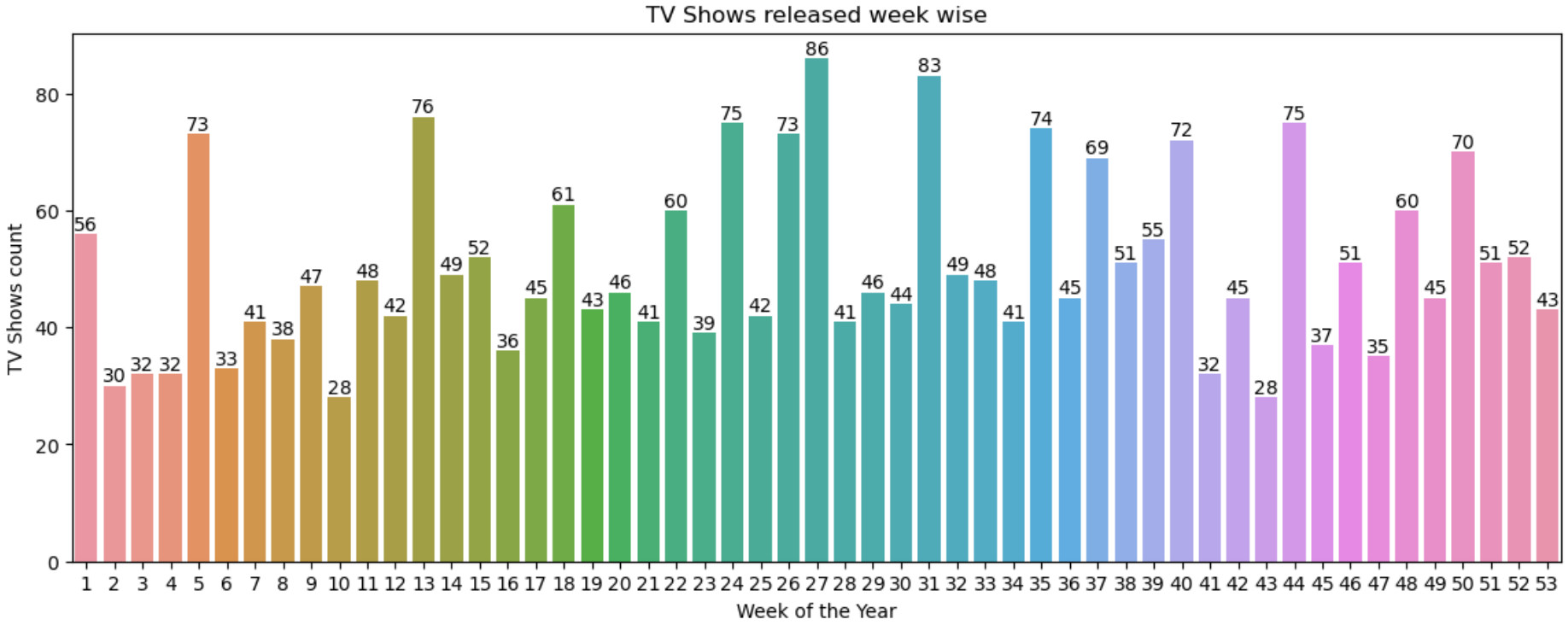
```
    bar_plot.text(index, value, str(value), ha='center', va='bottom')

plt.show()
```



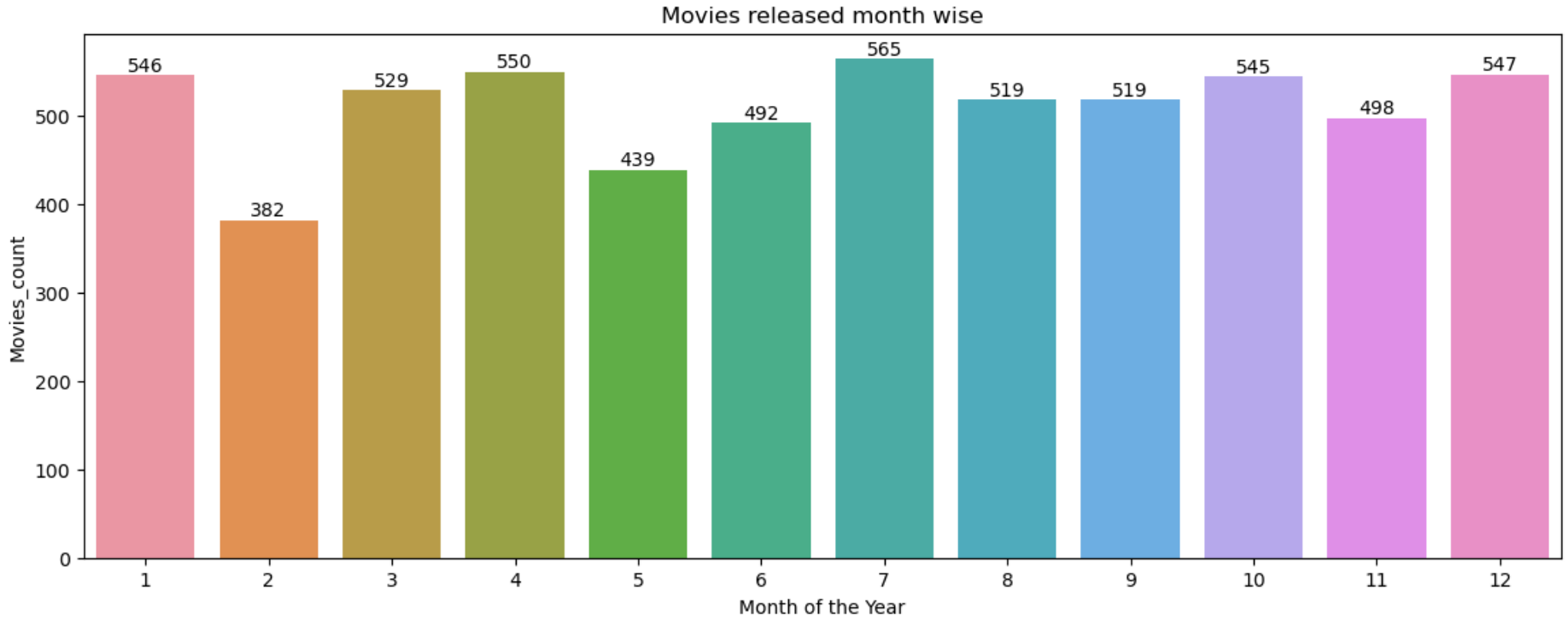
In [ ]:

In [69]:
```
df_tv_shows = df[df['type'] == 'TV Show']
tv_shows_count = df_tv_shows['week_added'].value_counts().sort_index()
# movies_count = movies_count.sort_values(ascending=False)

plt.figure(figsize=(14, 5))
bar_plot = sns.barplot(x=tv_shows_count.index, y=tv_shows_count.values)

plt.xlabel('Week of the Year')
plt.ylabel('TV Shows count')
plt.title('TV Shows released week wise')

for index, value in enumerate(tv_shows_count.values):
    bar_plot.text(index, value, str(value), ha='center', va='bottom')
```

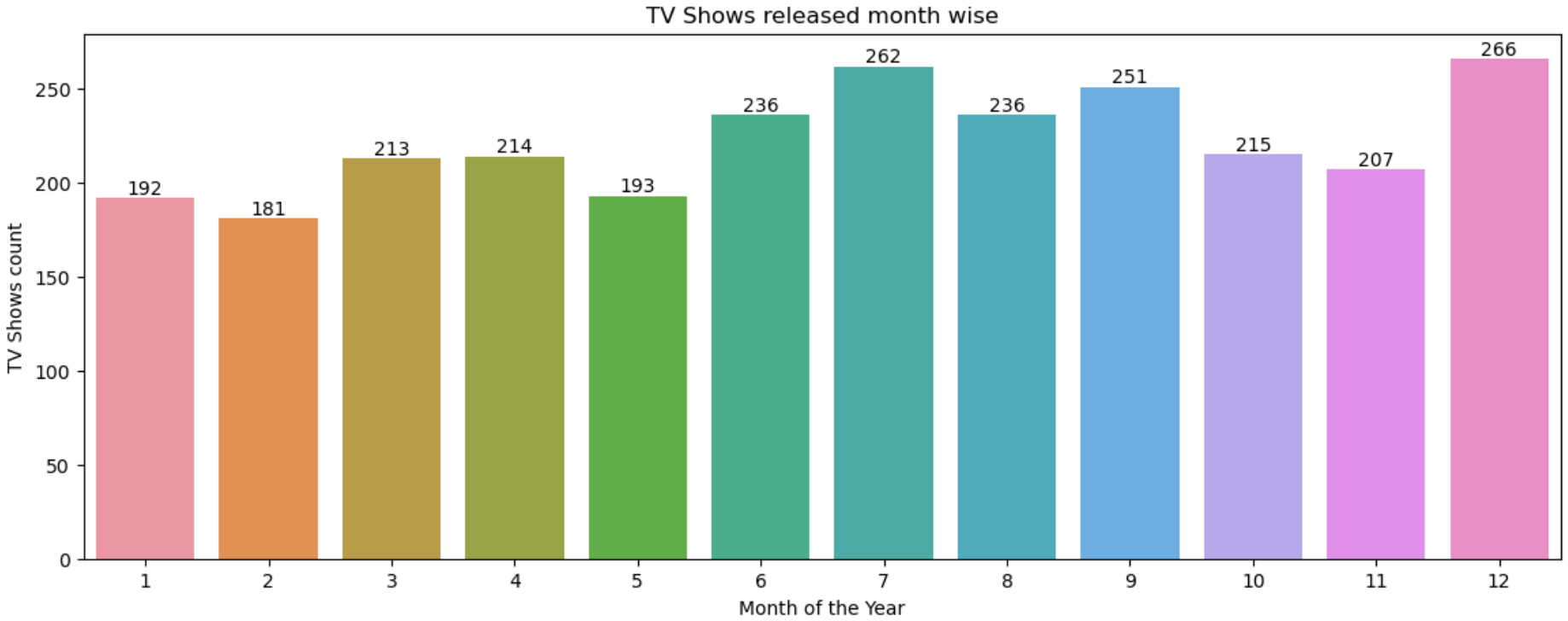
```
plt.show()
```



In [70]:
```
df_movies = df[df['type'] == 'Movie']
movies_count = df_movies['month_added'].value_counts().sort_index()
# movies_count = movies_count.sort_values(ascending=False)

plt.figure(figsize=(14, 5))
bar_plot = sns.barplot(x=movies_count.index, y=movies_count.values)

plt.xlabel('Month of the Year')
plt.ylabel('Movies_count')
plt.title('Movies released month wise')

for index, value in enumerate(movies_count.values):
    bar_plot.text(index, value, str(value), ha='center', va='bottom')

plt.show()
```



In [71]:
```
df_tv_shows = df[df['type'] == 'TV Show']
tv_shows_count = df_tv_shows['month_added'].value_counts().sort_index()
# movies_count = movies_count.sort_values(ascending=False)

plt.figure(figsize=(14, 5))
bar_plot = sns.barplot(x=tv_shows_count.index, y=tv_shows_count.values)

plt.xlabel('Month of the Year')
plt.ylabel('TV Shows count')
plt.title('TV Shows released month wise')

for index, value in enumerate(tv_shows_count.values):
    bar_plot.text(index, value, str(value), ha='center', va='bottom')

plt.show()
```



In [ ]:

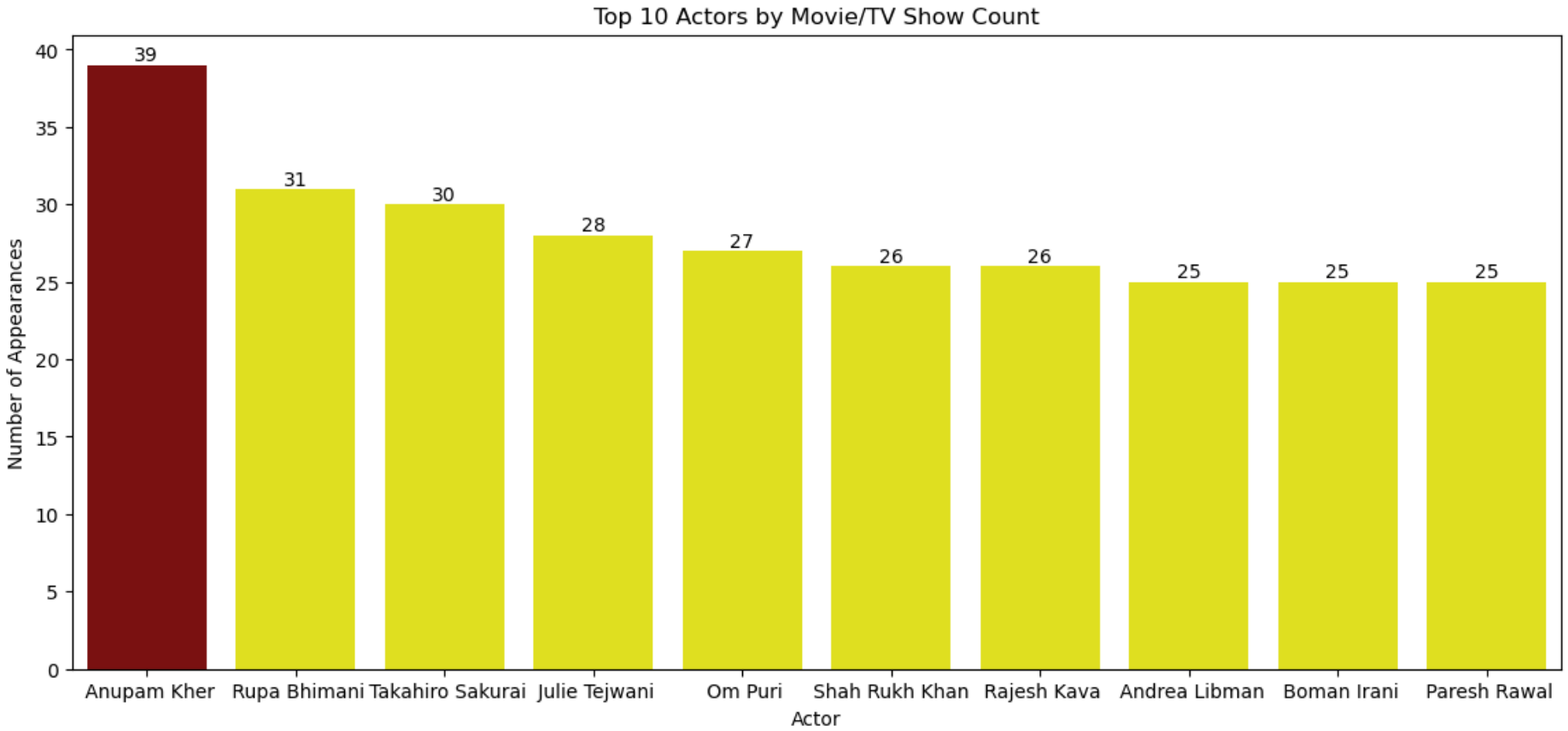
6. Analysis of actors/directors of different types of shows/movies.

a. Identify the top 10 actors who have appeared in most movies or TV shows.

b. Identify the top 10 directors who have appeared in most movies or TV shows.

In [74]:
```
# Count the occurrences of each actor
cast_counts = df_cast['cast'].value_counts()[1:]

# Select the top 10 actors
top_10_cast = cast_counts.head(10)

plt.figure(figsize=(14, 6))
colors = ['darkred'] + ['yellow'] * (len(top_10_cast) - 1)
```

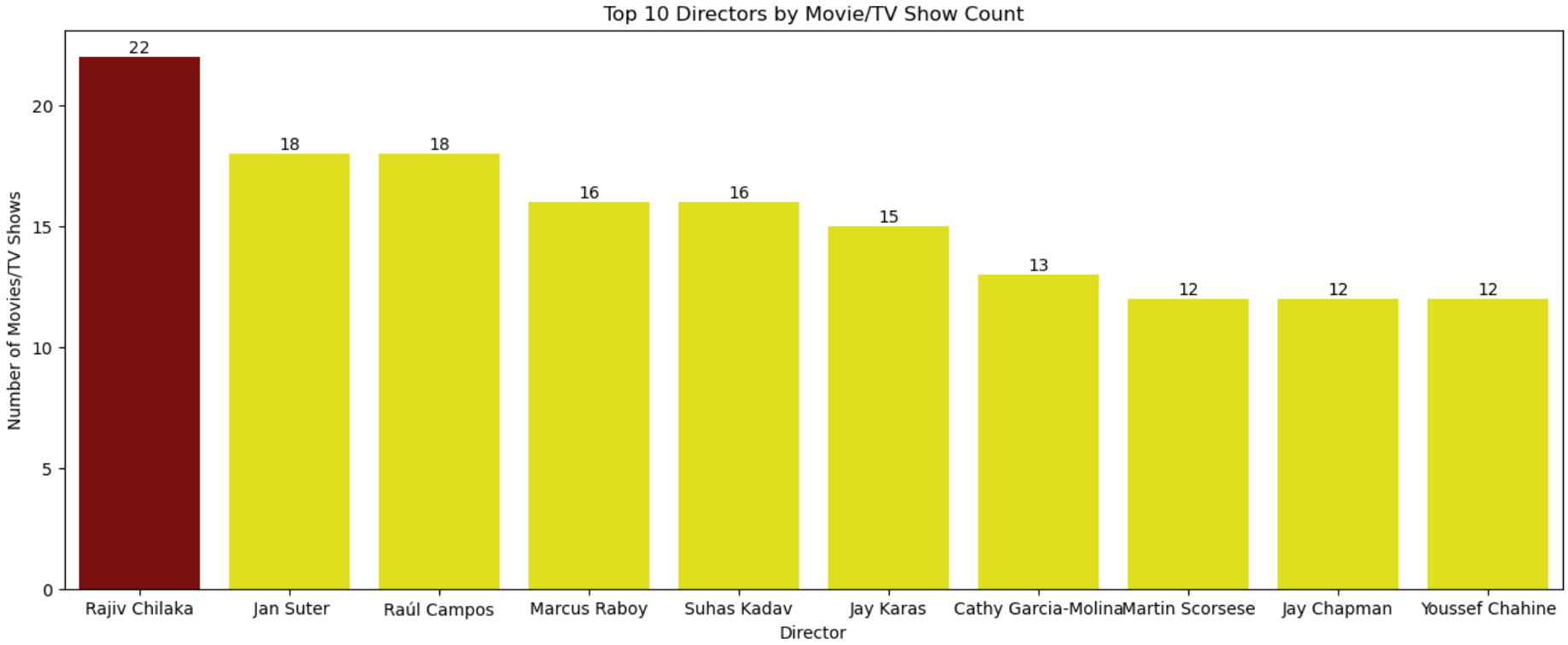
```
bar_plot = sns.barplot(x=top_10_cast.index, y=top_10_cast.values, palette=colors)

plt.xlabel('Actor')
plt.ylabel('Number of Appearances')
plt.title('Top 10 Actors by Movie/TV Show Count')

# Add count values on top of each bar
for index, value in enumerate(top_10_cast.values):
    bar_plot.text(index, value, str(value), ha='center', va='bottom')

plt.show()
```



From the above graph it is derived that the top 10 directors are: 1.Anupam Kher 2.Rupa Bhimani 3.Takahiro Sakurai 4.Julie Tejwani 5.Om Puri 6.Shah Rukh Khan 7.Rajesh Kava 8.Andrea Libman 9.Boman Irani 10.Paresh Rawal

In [78]:
```
# Count the occurrences of each actor
director_counts = df_director['director'].value_counts()[1:]

# Select the top 10 actors
top_10_directors = director_counts.head(10)

plt.figure(figsize=(16, 6))
colors = ['darkred'] + ['yellow'] * (len(top_10_directors) - 1)
bar_plot = sns.barplot(x=top_10_directors.index, y=top_10_directors.values, palette=colors)

plt.xlabel('Director')
plt.ylabel('Number of Movies/TV Shows')
plt.title('Top 10 Directors by Movie/TV Show Count')

# Add count values on top of each bar
for index, value in enumerate(top_10_directors.values):
    bar_plot.text(index, value, str(value), ha='center', va='bottom')

plt.show
```

Out[78]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```

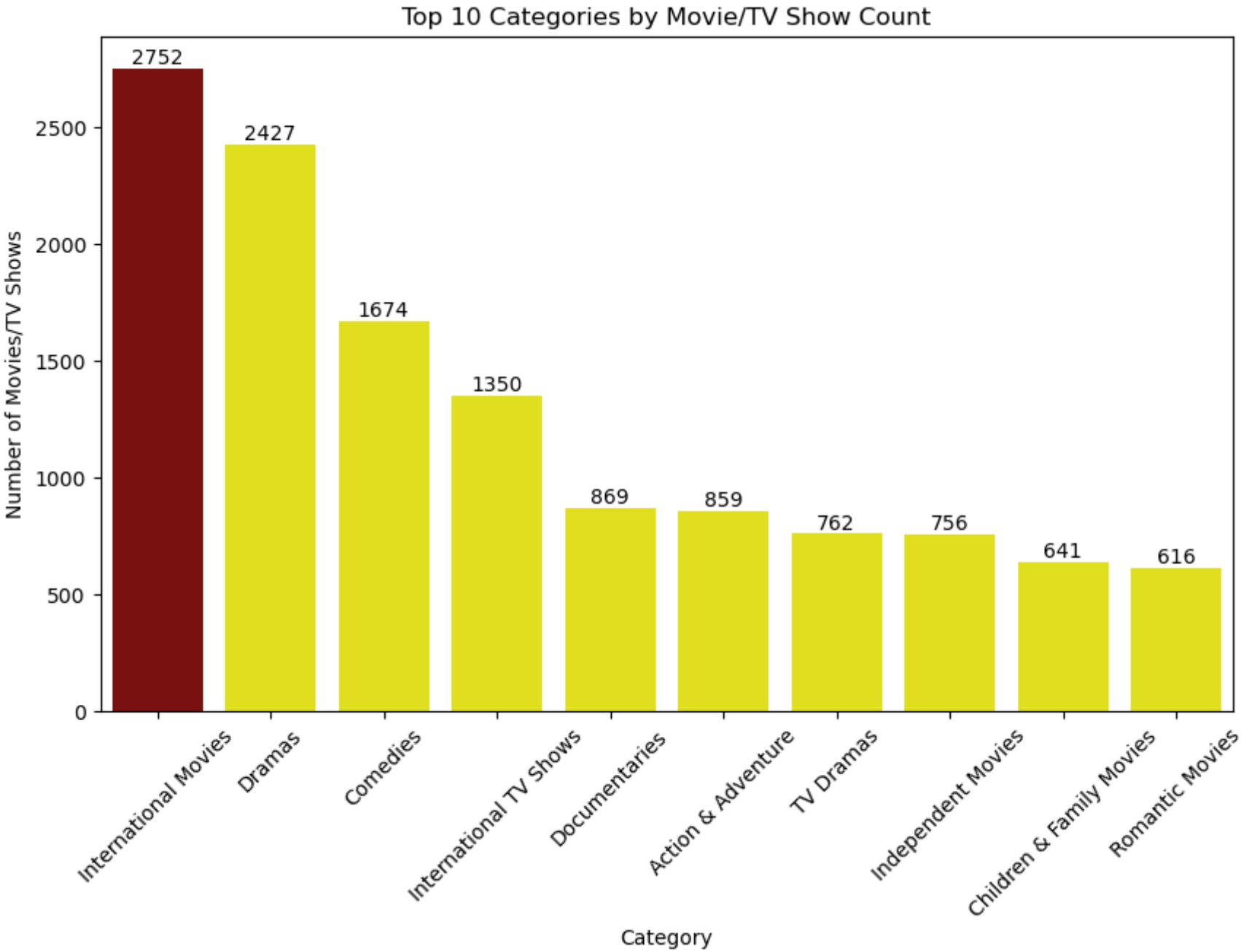


From the above graph it is derived that the top 10 directors are: 1.Rajiv Chilaka 2.Jan Suter 3.Raul Campos 4.Marcus Raboy 5.Suhas Kadav 6.Jay Karas 7.Cathy Garcia 8.Martin Scorsese 9.Jay Chapman 10.Youssef Chahine

7. Which genre movies are more popular or produced more

In [79]:
```
df_listed_in['listed_in'] = df_listed_in['listed_in'].str.strip()

# Count the occurrences of each actor
listed_in_counts = df_listed_in['listed_in'].value_counts()

# Select the top 10 actors
top_10_listed_in = listed_in_counts.head(10)

plt.figure(figsize=(10, 6))
colors = ['darkred'] + ['yellow'] * (len(top_10_directors) - 1)
bar_plot = sns.barplot(x=top_10_listed_in.index, y=top_10_listed_in.values, palette=colors)
```

```
# Customize the plot
plt.xlabel('Category')
plt.ylabel('Number of Movies/TV Shows')
plt.title('Top 10 Categories by Movie/TV Show Count')
plt.xticks(rotation=45)

# Add count values on top of each bar
for index, value in enumerate(top_10_listed_in.values):
    bar_plot.text(index, value, str(value), ha='center', va='bottom')

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

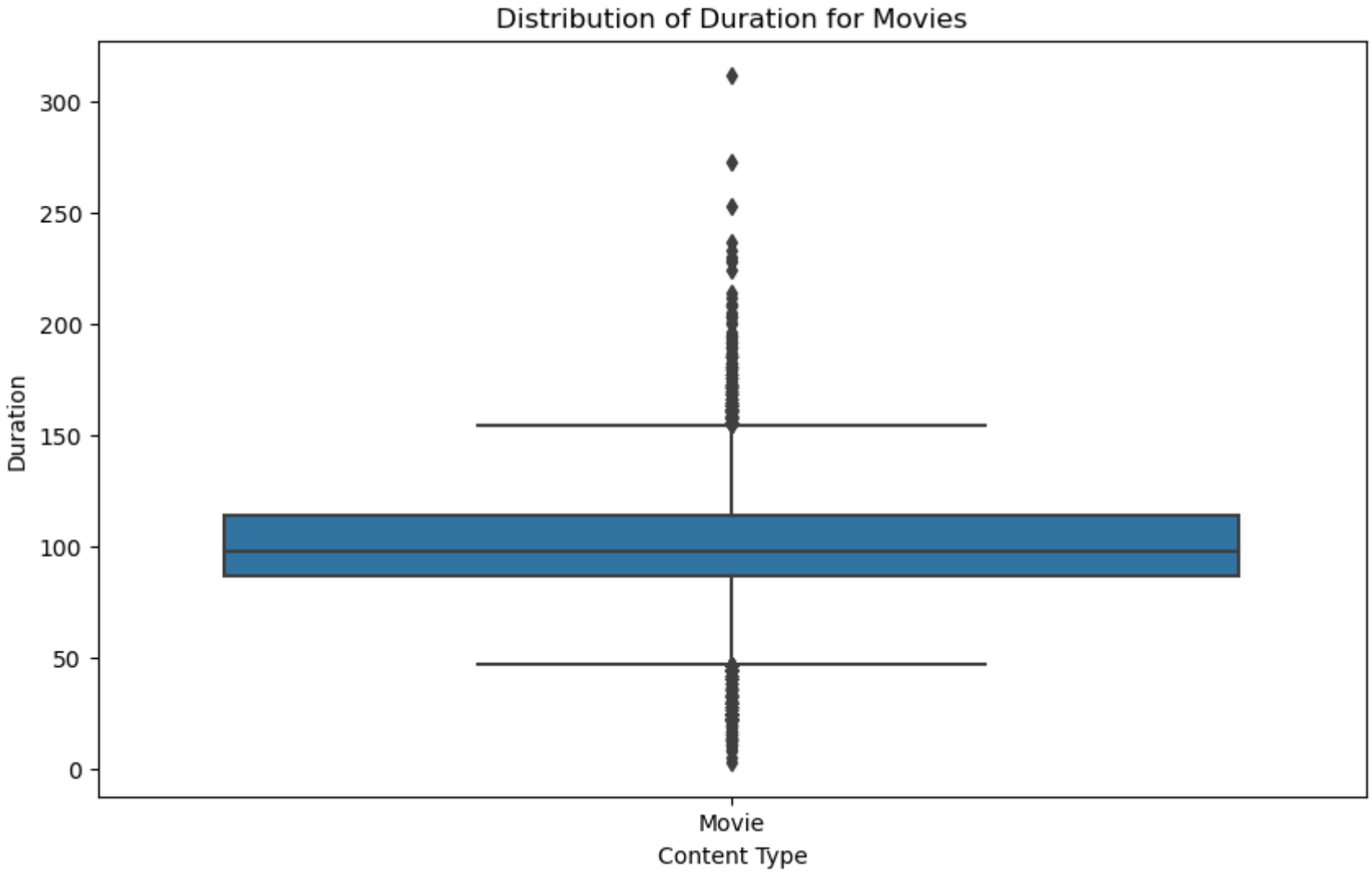

Top 10 Categories by Movie/TV Show Count
Number of Movies/TV Shows
Category
0
500
1000
1500
2000
2500
2752
2427
1674
1350
869
859
762
756
641
616
International Movies
Dramas
Comedies
International TV Shows
Documentaries
Action & Adventure
TV Dramas
Independent Movies
Children & Family Movies
Romantic Movies

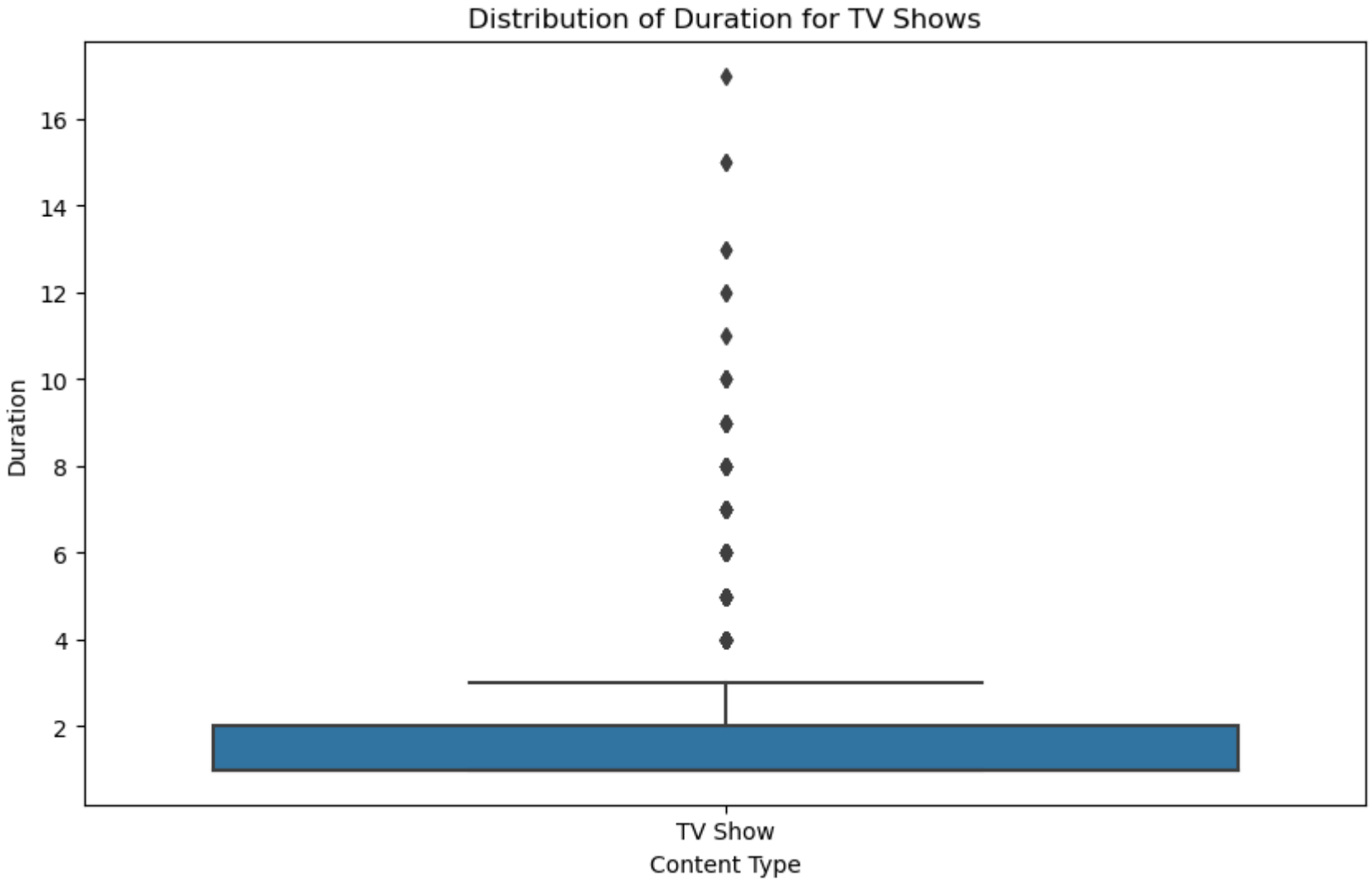
From the above graph it is derived that the top 10 categories are: 1.International Movies 2.Dramas 3.Comedies 4.International TV Shows 5.Documentaries 6.Action and Adventure 7.TV Dramas 8.Independent Movies 9.Children and Family Movies 10.Romantic Movies

8. Duration distribution for movies and TV shows

In [104…

```
# Creating a boxplot for movie duration
plt.figure(figsize=(10, 6))
sns.boxplot(data=df_movies, x='type', y='duration')
plt.xlabel('Content Type')
plt.ylabel('Duration')
plt.title('Distribution of Duration for Movies')
plt.show()

# Creating a boxplot for TV show duration
plt.figure(figsize=(10, 6))
sns.boxplot(data=df_tv_shows, x='type', y='duration')
plt.xlabel('Content Type')
plt.ylabel('Duration')
plt.title('Distribution of Duration for TV Shows')
plt.show()
```

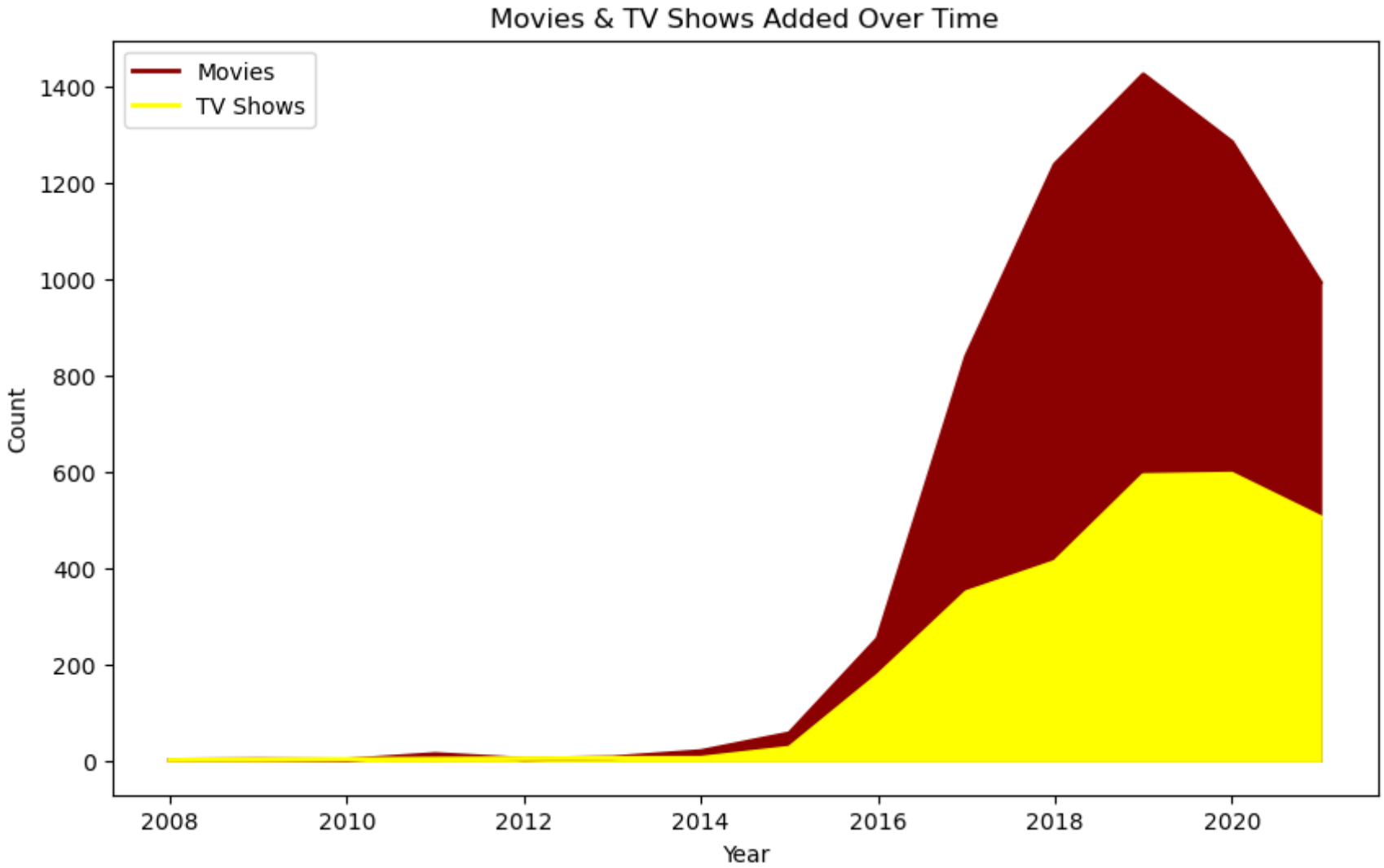

Distribution of Duration for Movies
Duration
300
250
200
150
100
50
0
Movie
Content Type



9. Movies and TV Shows added over time

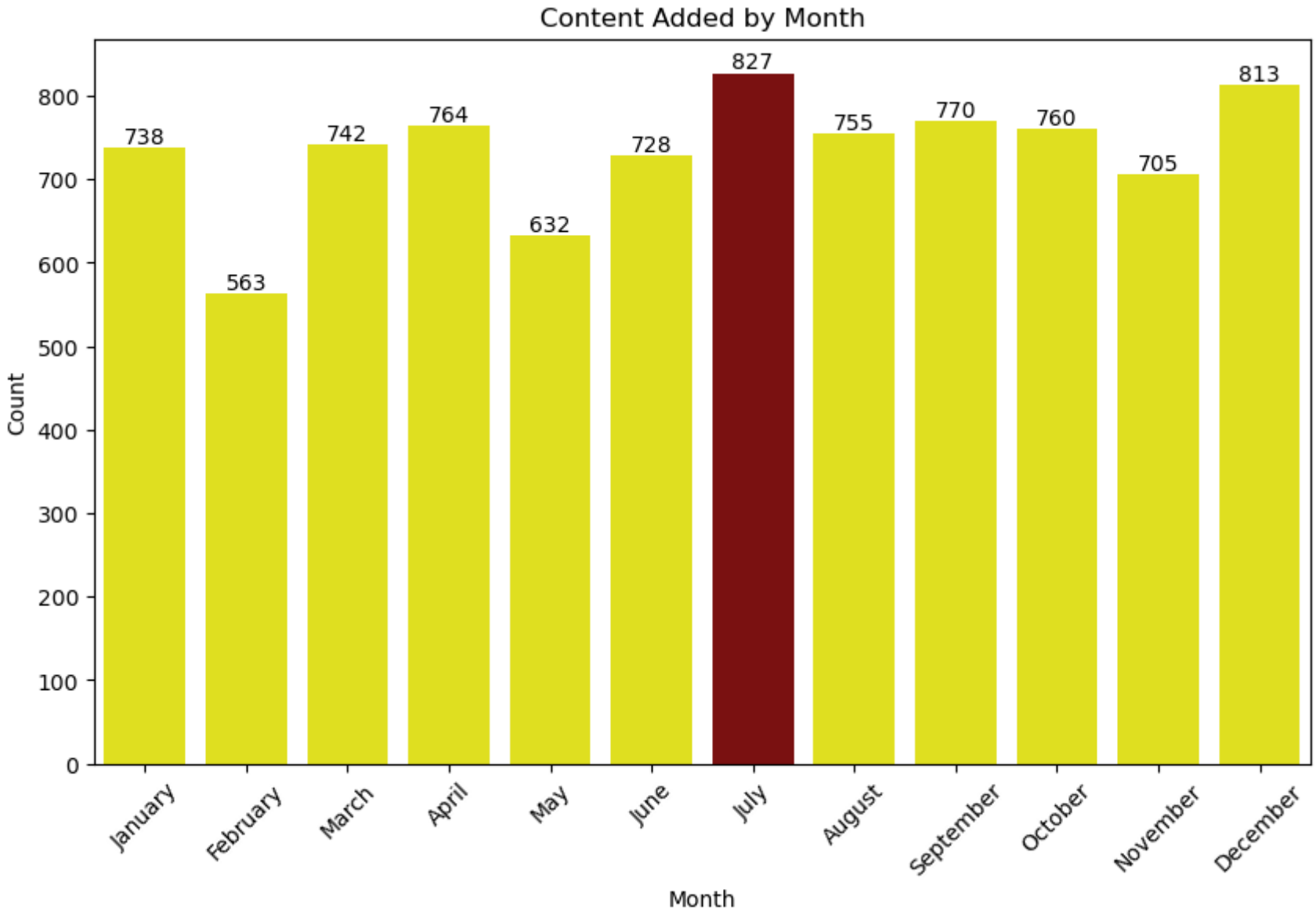
In [80]:
```
# Filter the DataFrame to include only Movies and TV Shows
df_movies = df[df['type'] == 'Movie']
df_tv_shows = df[df['type'] == 'TV Show']

# Group the data by year and count the number of Movies and TV Shows
```

```
# added in each year
movies_count = df_movies['year_added'].value_counts().sort_index()
tv_shows_count = df_tv_shows['year_added'].value_counts().sort_index()

# Create a line chart to visualize the trends over time
plt.figure(figsize=(10, 6))
plt.plot(movies_count.index, movies_count.values, color='darkred',
label='Movies', linewidth=2)
plt.plot(tv_shows_count.index, tv_shows_count.values, color='yellow',
label='TV Shows', linewidth=2)

# Fill the area under the line charts
plt.fill_between(movies_count.index, movies_count.values, color='darkred')
plt.fill_between(tv_shows_count.index, tv_shows_count.values, color='yellow')

# Customize the plot
plt.xlabel('Year')
plt.ylabel('Count')
plt.title('Movies & TV Shows Added Over Time')
plt.legend()

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



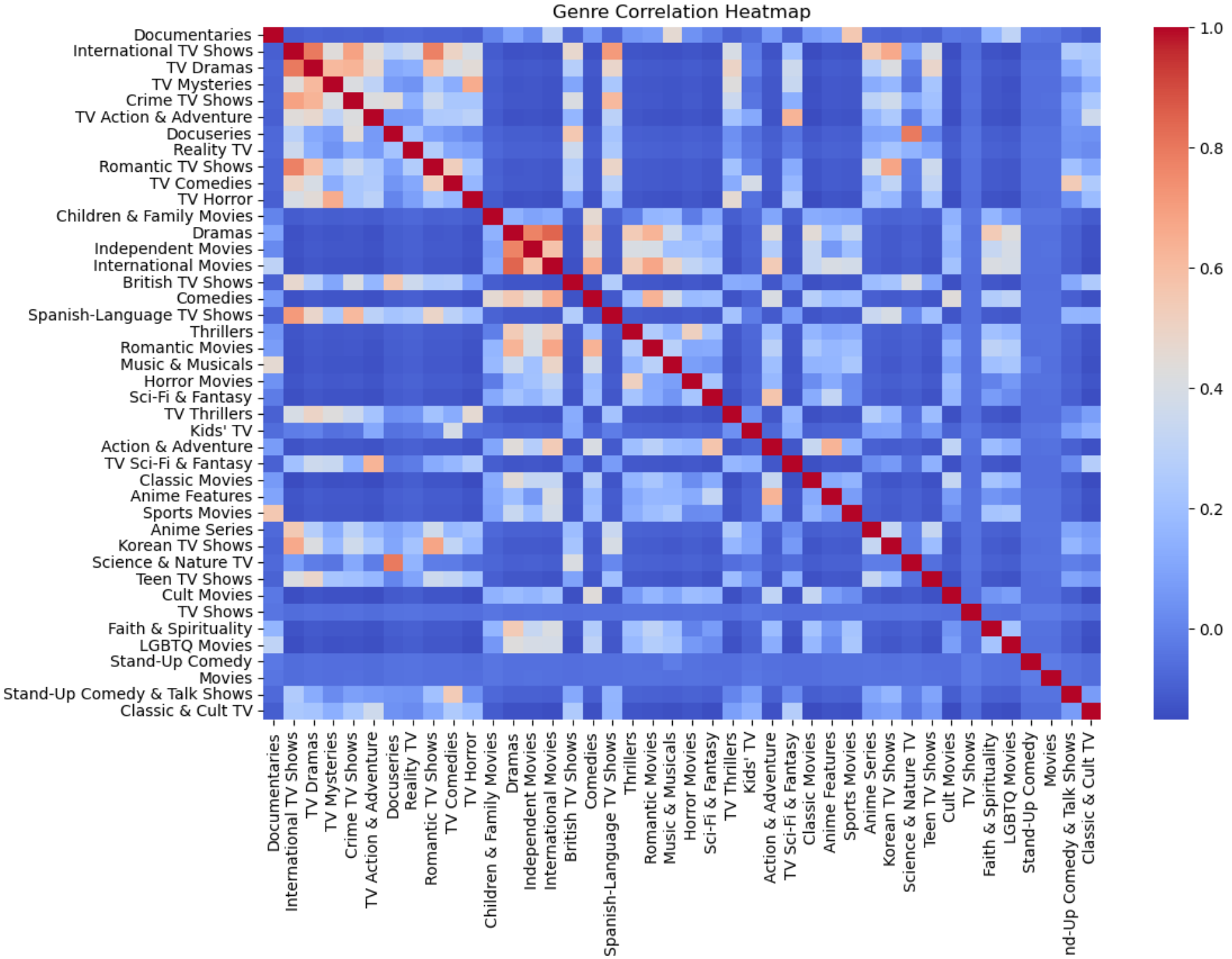
In [81]:
```
# Extract the month from the 'date_added' column
df['month_added'] = pd.to_datetime(df['date_added']).dt.month_name()

# Define the order of the months
month_order = ['January', 'February', 'March', 'April', 'May', 'June', 'July',
               'August', 'September', 'October', 'November', 'December']

# Count the number of shows added in each month
monthly_counts = df['month_added'].value_counts().loc[month_order]
```

```
# Determine the maximum count
max_count = monthly_counts.max()

# Set the color for the highest bar and the rest of the bars
colors = ['darkred' if count == max_count else 'yellow' for count in monthly_counts]

# Create the bar chart
plt.figure(figsize=(10, 6))
bar_plot = sns.barplot(x=monthly_counts.index, y=monthly_counts.values, palette=colors)

# Customize the plot
plt.xlabel('Month')
plt.ylabel('Count')
plt.title('Content Added by Month')

# Add count values on top of each bar
for index, value in enumerate(monthly_counts.values):
    bar_plot.text(index, value, str(value), ha='center', va='bottom')

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

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



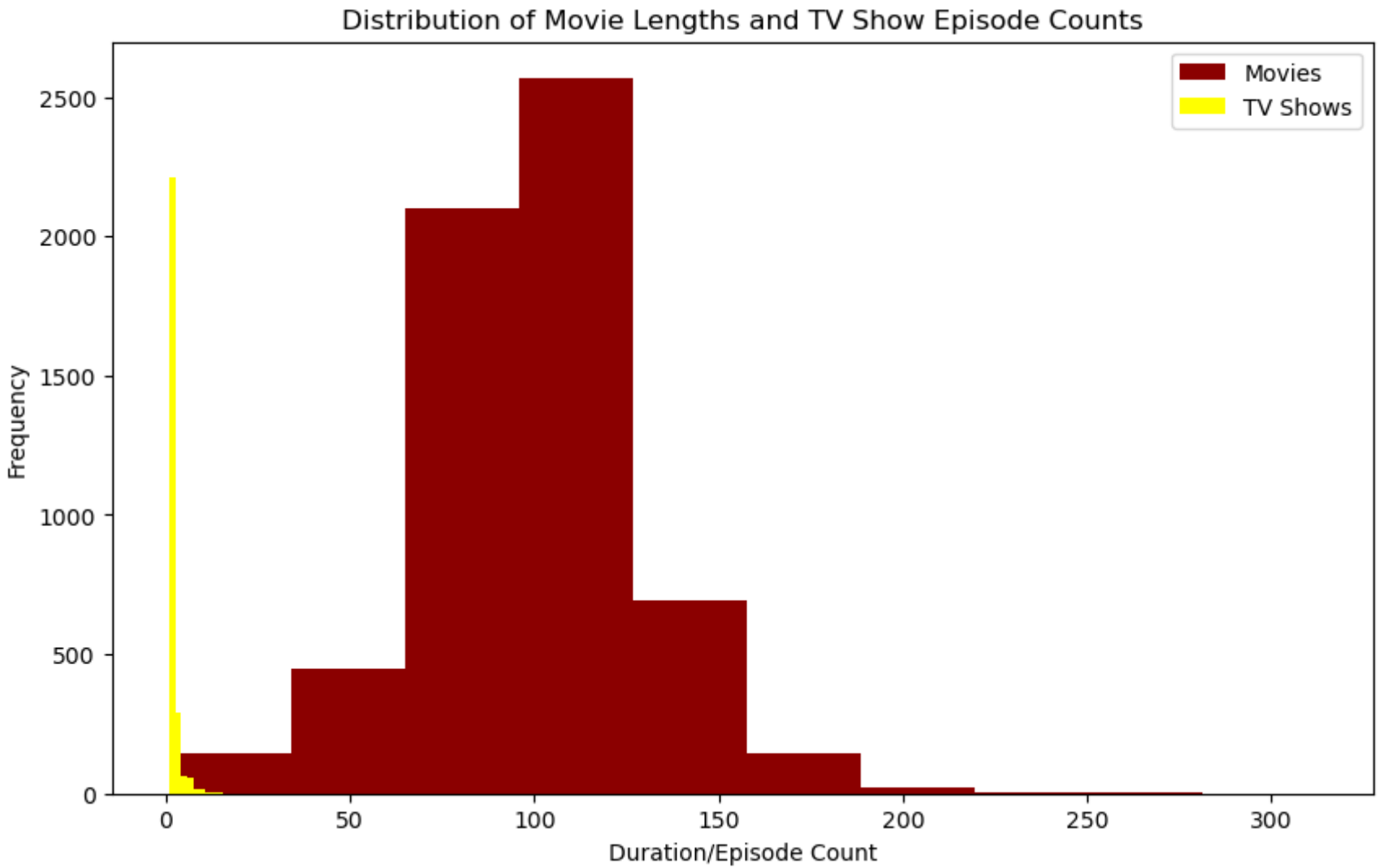
In [86]:
```
# Extracting unique genres from the 'listed_in' column
genres = df['listed_in'].str.split(', ', expand=True).stack().unique()

# Create a new DataFrame to store the genre data
genre_data = pd.DataFrame(index=genres, columns=genres, dtype=float)
```

```
# Fill the genre data DataFrame with zeros
genre_data.fillna(0, inplace=True)

# Iterate over each row in the original DataFrame and update the genre data DataFrame
for _, row in df.iterrows():
    listed_in = row['listed_in'].split(', ')
    for genre1 in listed_in:
        for genre2 in listed_in:
            genre_data.at[genre1, genre2] += 1

# Create a correlation matrix using the genre data
correlation_matrix = genre_data.corr()

# Create the heatmap
plt.figure(figsize=(12, 8))
sns.heatmap(correlation_matrix, annot=False, cmap='coolwarm')

# Customize the plot
plt.title('Genre Correlation Heatmap')
plt.xticks(rotation=90)
plt.yticks(rotation=0)

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


Genre Correlation Heatmap
Documentaries
International TV Shows
TV Dramas
TV Mysteries
Crime TV Shows
TV Action & Adventure
Docuseries
Reality TV
Romantic TV Shows
TV Comedies
TV Horror
Children & Family Movies
Dramas
Independent Movies
International Movies
British TV Shows
Comedies
Spanish-Language TV Shows
Thrillers
Romantic Movies
Music & Musicals
Horror Movies
Sci-Fi & Fantasy
TV Thrillers
Kids' TV
Action & Adventure
TV Sci-Fi & Fantasy
Classic Movies
Anime Features
Sports Movies
Anime Series
Korean TV Shows
Science & Nature TV
Teen TV Shows
Cult Movies
TV Shows
Faith & Spirituality
LGBTQ Movies
Stand-Up Comedy
Movies
Stand-Up Comedy & Talk Shows
Classic & Cult TV
1.0
0.8
0.6
0.4
0.2
0.0

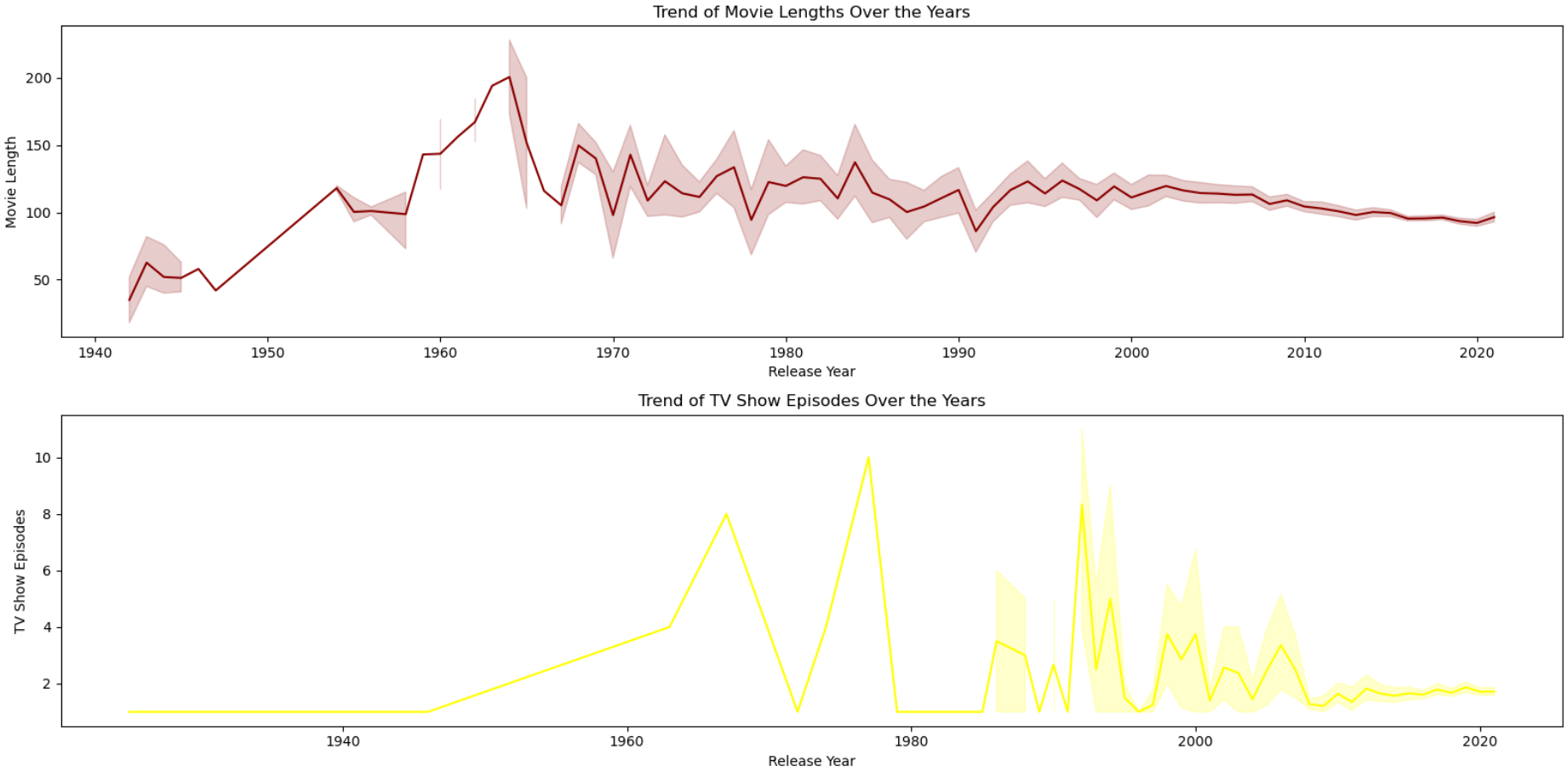
Sta

In [92]:
```
# Extract the movie lengths and TV show episode counts
movie_lengths = df_movies['duration'].str.extract('(\d+)', expand=False).astype(int)
tv_show_episodes = df_tv_shows['duration'].str.extract('(\d+)', expand=False).astype(int)

# Plot the histogram
plt.figure(figsize=(10, 6))
plt.hist(movie_lengths, bins=10, color='DarkRed', label='Movies')
plt.hist(tv_show_episodes, bins=10, color='Yellow', label='TV Shows')

# Customize the plot
plt.xlabel('Duration/Episode Count')
plt.ylabel('Frequency')
plt.title('Distribution of Movie Lengths and TV Show Episode Counts')
plt.legend()

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



In [85]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Extract the movie lengths and TV show episodes from the 'duration' column
movie_lengths = df_movies['duration'].str.extract('(\d+)', expand=False).astype(int)
tv_show_episodes = df_tv_shows['duration'].str.extract('(\d+)', expand=False).astype(int)

# Create line plots for movie lengths and TV show episodes
plt.figure(figsize=(16, 8))
```

```
plt.subplot(2, 1, 1)
sns.lineplot(data=df_movies, x='release_year', y=movie_lengths, color=colors[0])
plt.xlabel('Release Year')
plt.ylabel('Movie Length')
plt.title('Trend of Movie Lengths Over the Years')

plt.subplot(2, 1, 2)
sns.lineplot(data=df_tv_shows, x='release_year', y=tv_show_episodes,color=colors[1])
plt.xlabel('Release Year')
plt.ylabel('TV Show Episodes')
plt.title('Trend of TV Show Episodes Over the Years')

# Adjust the layout and spacing
plt.tight_layout()

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



Insights

1. Our analysis revealed that Netflix had added more movies than TV shows, aligning with the expectation that movies dominate their content library.
2. Content Addition: July emerged as the month when Netflix adds the most content, closely followed by December, indicating a strategic approach to content release.
3. Genre Correlation: Strong positive associations were observed between various genres, such as TV dramas and international TV shows, romantic and international TV shows, and independent movies and dramas. These correlations provide insights into viewer preferences and content interconnections.

4. Netflix saw its real growth starting from the year 2015, & we can see it added more Movies than TV Shows over the years.Also, it is interesting that the content addition dropped in 2020. This could be due to the pandemic situation.
5. Movie Lengths: The analysis of movie durations indicated a peak around the 1960s, followed by a stabilization around 100 minutes, highlighting a trend in movie lengths over time.
6. 2019 is the year Netflix added more movies.
7. TV Show Episodes: Most TV shows on Netflix have one season, suggesting a preference for shorter series among viewers.
8. Our data analysis journey showcased the power of data in unraveling the mysteries of Netflix's content landscape, providing valuable insights for viewers and content creators.
9. As the streaming industry evolves, understanding these patterns and trends becomes increasingly essential for navigating the dynamic landscape of Netflix and its vast library.

Recommendation

1. As International Movies , Dramas , Comedies and International TV shows contribute about 50 % . Netflix should focus more on adding contect related to these as they are most popular and watched more.
2. Audience is watching movies which has around 100-120 min duration , so should add more content with 100-120 min duration.
3. Audience is watching TV Shows which has 1-2 seasons , so should add more content with 1-2 seasons.
4. The pie chart visualization shows that approximately 70% of the content on Netflix consists of film, while the remaining 30% are TV shows. Netflix should try to balance both the categories as both are popular.

In [ ]: