**PROJECT :** NETFLIX

**NAME :** BHUVANESWARAN . S

**BATCH :** DSML OCT24(1) BEGINER 2

**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.

Business Problem

Analyze the data and generate insights that could help Netflix ijn deciding which type of shows/movies to produce and how they can grow the business in different countries

**Dataset**

Link: [Dataset\_link](https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/000/940/original/netflix.csv)

*(After clicking on the above link, you can download the files by right-clicking on the page and clicking on "Save As", then naming the file as per your wish, with .csv as the extension.)*

The dataset provided to you consists of a list of all the TV shows/movies available on Netflix:

**Show\_id:** Unique ID for every Movie / Tv Show  
**Type:** Identifier - A Movie or TV Show  
**Title:** Title of the Movie / Tv Show  
**Director:** Director of the Movie  
**Cast:** Actors involved in the movie/show  
**Country:** Country where the movie/show was produced  
**Date\_added:** Date it was added on Netflix  
**Release\_year:** Actual Release year of the movie/show  
**Rating:** TV Rating of the movie/show  
**Duration:** Total Duration - in minutes or number of seasons  
**Listed\_in:** Genre  
**Description:** The summary description

**1.Defining Problem Statement and Analysing basic metrics?**

The objective of this analysis is to generate data-driven insights that can assist Netflix in making strategic decisions about its content production and expansion. Specifically, Netflix wants to understand the type of shows and movies it should focus on producing and how it can expand its reach and grow the business in different countries.

**Analyzing Basic Metrics**

overall landscape of Netflix’s We will first explore the basic metrics and distributions within the dataset to understand the catalog. This step will help us get a sense of the data before diving deeper into specific questions.

1. **Content Type Distribution:**

* Count the number of Movies vs. TV Shows.
* What percentage of content is Movies vs. TV Shows?

**Code :**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

# Load dataset

df = pd.read\_csv("/content/netflix.csv")

# Clean column names to avoid issues with spaces or case sensitivity

df.columns = df.columns.str.strip().str.lower()

# Count the number of Movies vs. TV Shows

content\_type\_counts = df['type'].value\_counts()

print("Content Type Counts:")

print(content\_type\_counts)

# Calculate percentage distribution

content\_type\_percentage = df['type'].value\_counts(normalize=True) \* 100

print("\nPercentage Distribution:")

print(content\_type\_percentage)

# Plotting the Content Type Distribution

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

sns.barplot(x=content\_type\_counts.index, y=content\_type\_counts.values, palette='viridis')

plt.title('Content Type Distribution (Movies vs TV Shows)')

plt.xlabel('Content Type')

plt.ylabel('Count')

plt.show()

# Plotting the Percentage Distribution

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

sns.barplot(x=content\_type\_percentage.index, y=content\_type\_percentage.values, palette='viridis')

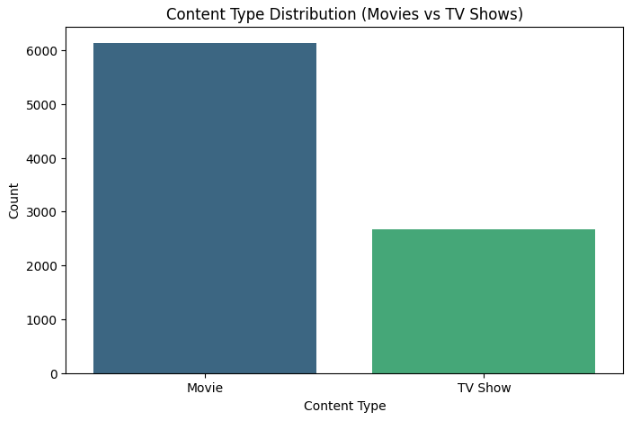
plt.title('Percentage Distribution of Movies vs TV Shows')

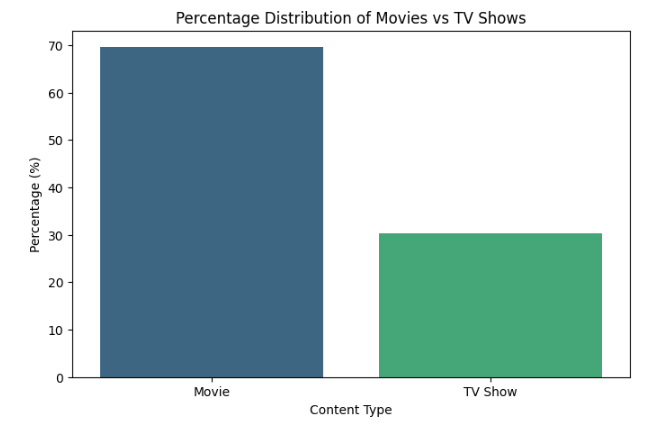
plt.xlabel('Content Type')

plt.ylabel('Percentage (%)')

plt.show()

**output:**

****

****

**Insights:**

* **If the count of TV shows is significantly higher than movies**: It would suggest that Netflix has been focusing more on TV Shows than Movies in its content strategy.
* **If the percentage of Movies is higher**: Netflix may consider diversifying its library more by focusing on TV shows in the future.

1. **Release Year Distribution:**

* Determine the number of releases per year (based on Release\_year).
* Track the trends over time — Has Netflix increased its content output over the years?

**Code :**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

# Load dataset

df = pd.read\_csv("/content/netflix.csv")

# Clean column names to avoid issues with spaces or case sensitivity

df.columns = df.columns.str.strip().str.lower()

# Convert Release\_year to a numeric value to avoid errors

df['release\_year'] = pd.to\_numeric(df['release\_year'], errors='coerce')

# Count the number of releases per year

release\_year\_counts = df['release\_year'].value\_counts().sort\_index()

# Print the release year counts to see the data

print("Release Year Distribution:")

print(release\_year\_counts)

# Plotting the trend over time

plt.figure(figsize=(10, 6))

sns.lineplot(x=release\_year\_counts.index, y=release\_year\_counts.values, marker='o', color='b')

plt.title('Number of Releases per Year on Netflix')

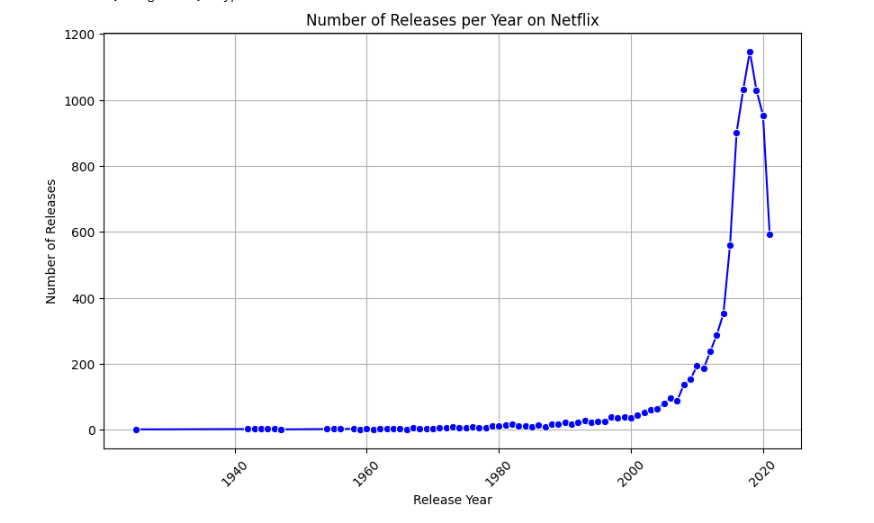
plt.xlabel('Release Year')

plt.ylabel('Number of Releases')

plt.xticks(rotation=45)

plt.grid(True)

plt.show()

**output: **

**Insights:**

* **Increase in Releases Over Time**: If there is a noticeable increase in releases after a certain year, it may indicate that Netflix has been scaling up its content production in recent years.
* **Spikes in Production**: Any significant spikes in the number of releases may correlate with important changes in Netflix’s business strategy (e.g., global expansion, production investments).
* **Steady Production**: If releases have been relatively steady over time, Netflix may need to evaluate whether its content production strategy is keeping pace with the growing demand for new content.

1. **Content Duration Analysis:**

* How many movies and TV shows have long durations vs. short durations?
* What is the average duration of movies vs. TV shows? (The Duration field indicates either the number of seasons for TV Shows or the total runtime in minutes for Movies.)

**Code :**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

# Load dataset

df = pd.read\_csv("/content/netflix.csv")

# Clean column names to avoid issues with spaces or case sensitivity

df.columns = df.columns.str.strip().str.lower()

# Extracting and cleaning 'duration' column for both movies and TV shows

# For TV Shows, 'Duration' will be in seasons; for Movies, 'Duration' will be in minutes.

# We need to separate and process them accordingly.

# Convert 'Duration' column to strings and handle possible errors

df['duration'] = df['duration'].astype(str)

# Separate TV Shows and Movies

movies = df[df['type'] == 'Movie']

tv\_shows = df[df['type'] == 'TV Show']

# For Movies: Extract numeric duration in minutes

movies['duration\_minutes'] = movies['duration'].str.replace(' min', '').astype(float)

# For TV Shows: Extract number of seasons

# Updated to handle cases like '2s'

tv\_shows['duration\_seasons'] = tv\_shows['duration'].str.replace(r'( Season|s)', '', regex=True).astype(float)

#The regex=True argument enables the use of regular expressions.

#The updated pattern ( Season|s) matches either " Season" or "s".

# Now, calculate the average duration for both Movies and TV Shows

avg\_movie\_duration = movies['duration\_minutes'].mean()

avg\_tv\_show\_duration = tv\_shows['duration\_seasons'].mean()

print(f"Average movie duration: {avg\_movie\_duration:.2f} minutes")

print(f"Average TV show duration: {avg\_tv\_show\_duration:.2f} seasons")

# Plotting the distribution of durations

plt.figure(figsize=(10, 6))

# Plot for Movies

plt.subplot(1, 2, 1)

sns.histplot(movies['duration\_minutes'], kde=True, color='blue', bins=30)

plt.title('Distribution of Movie Durations')

plt.xlabel('Duration (in minutes)')

plt.ylabel('Frequency')

# Plot for TV Shows

plt.subplot(1, 2, 2)

sns.histplot(tv\_shows['duration\_seasons'], kde=True, color='green', bins=15)

plt.title('Distribution of TV Show Durations')

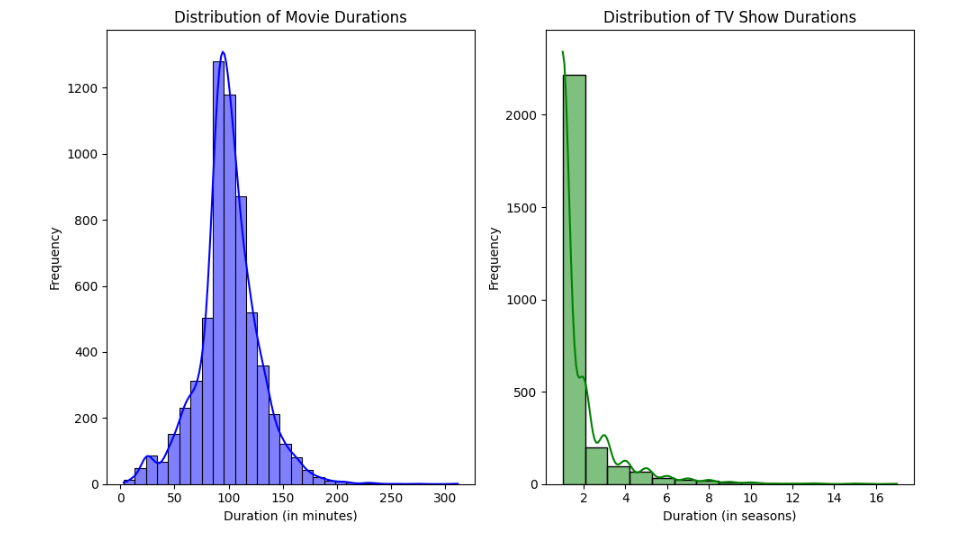
plt.xlabel('Duration (in seasons)')

plt.ylabel('Frequency')

plt.tight\_layout()

plt.show()

**output:**

****

**Insights:**

* **Duration Preferences**: If the average duration for TV Shows is higher than for Movies, Netflix might consider producing more short-form content if it aligns with user preferences.
* **Short vs. Long Content**: The classification of short and long content can help Netflix understand if it’s focusing too much on long-form content or should balance it more with shorter shows/movies.

1. **Geographical Content Distribution:**

* Count the number of Movies and TV Shows available in each country (based on Country).
* Identify countries with high content availability and those with fewer shows or movies.

**Code :**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

# Load dataset

df = pd.read\_csv("/content/netflix.csv")

# Clean column names to avoid issues with spaces or case sensitivity

df.columns = df.columns.str.strip().str.lower()

# Drop rows where 'country' is NaN, as we cannot analyze them

df = df.dropna(subset=['country'])

# Count the number of Movies and TV Shows in each country

country\_content\_counts = df.groupby('country')['type'].value\_counts().unstack(fill\_value=0)

# Print country content counts for Movies and TV Shows

print("Number of Movies and TV Shows available in each country:")

print(country\_content\_counts)

# Identify countries with high content availability

top\_countries = country\_content\_counts.sum(axis=1).sort\_values(ascending=False).head(10)

print("\nTop 10 countries with the most content:")

print(top\_countries)

# Plotting the Geographical Distribution of Content

plt.figure(figsize=(12, 8))

# Plot the top 10 countries with the most content

sns.barplot(x=top\_countries.index, y=top\_countries.values, palette='viridis')

plt.title('Top 10 Countries with the Most Content Available on Netflix')

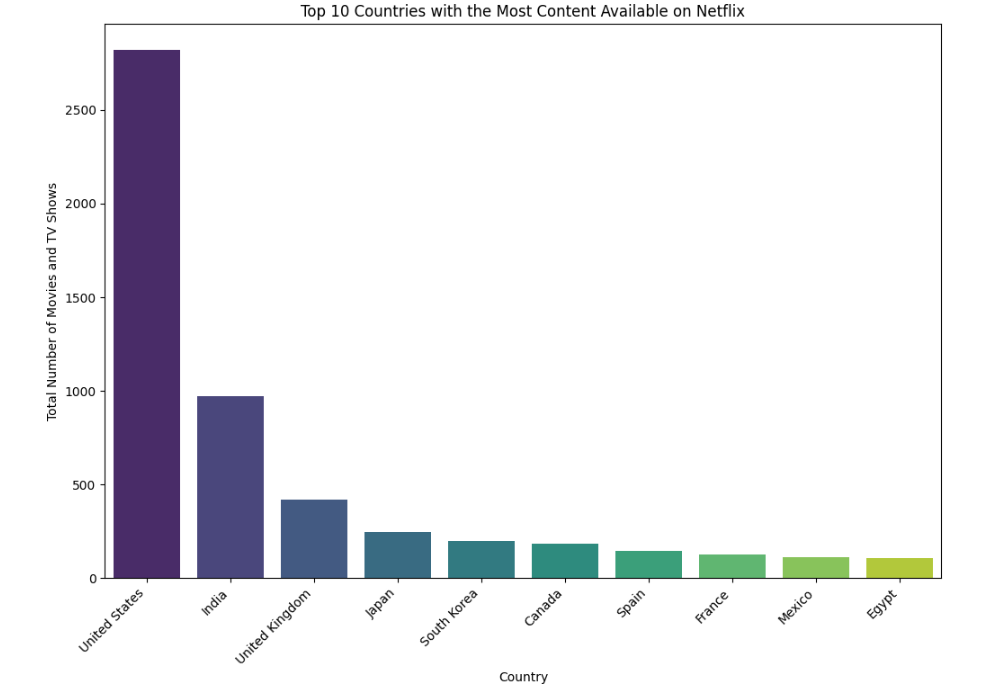
plt.xlabel('Country')

plt.ylabel('Total Number of Movies and TV Shows')

plt.xticks(rotation=45, ha='right')

plt.show()

**output:**

****

**Insights:**

* **Countries with High Content Availability**: Countries with a larger content library might indicate Netflix's strategic focus on these markets, possibly due to higher demand or localized content production.
* **Countries with Low Content Availability**: Countries with fewer Movies/TV Shows may indicate markets where Netflix needs to expand its content library, either through licensing deals or original productions.
* **Geographical Focus**: You can use these insights to determine if Netflix is focusing its resources on high-content countries or if there are underserved regions that could benefit from more content.

1. **Rating Analysis:**

* What is the distribution of TV ratings across movies and TV shows? (Based on the Rating column, which could include ratings like "PG", "TV-MA", "R", etc.)

**Code :**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

# Load dataset

df = pd.read\_csv("/content/netflix.csv")

# Clean column names to avoid issues with spaces or case sensitivity

df.columns = df.columns.str.strip().str.lower()

# Drop rows where 'rating' is NaN, as we cannot analyze them

df = df.dropna(subset=['rating'])

# Count the number of Movies and TV Shows for each rating

rating\_distribution = df.groupby(['type', 'rating']).size().unstack(fill\_value=0)

# Print the rating distribution for Movies and TV Shows

print("Rating Distribution across Movies and TV Shows:")

print(rating\_distribution)

# Plotting the Rating Distribution for Movies and TV Shows

plt.figure(figsize=(12, 8))

rating\_distribution.T.plot(kind='bar', stacked=True, figsize=(12, 8), colormap='viridis')

plt.title('Rating Distribution Across Movies and TV Shows')

plt.xlabel('Rating')

plt.ylabel('Count of Movies and TV Shows')

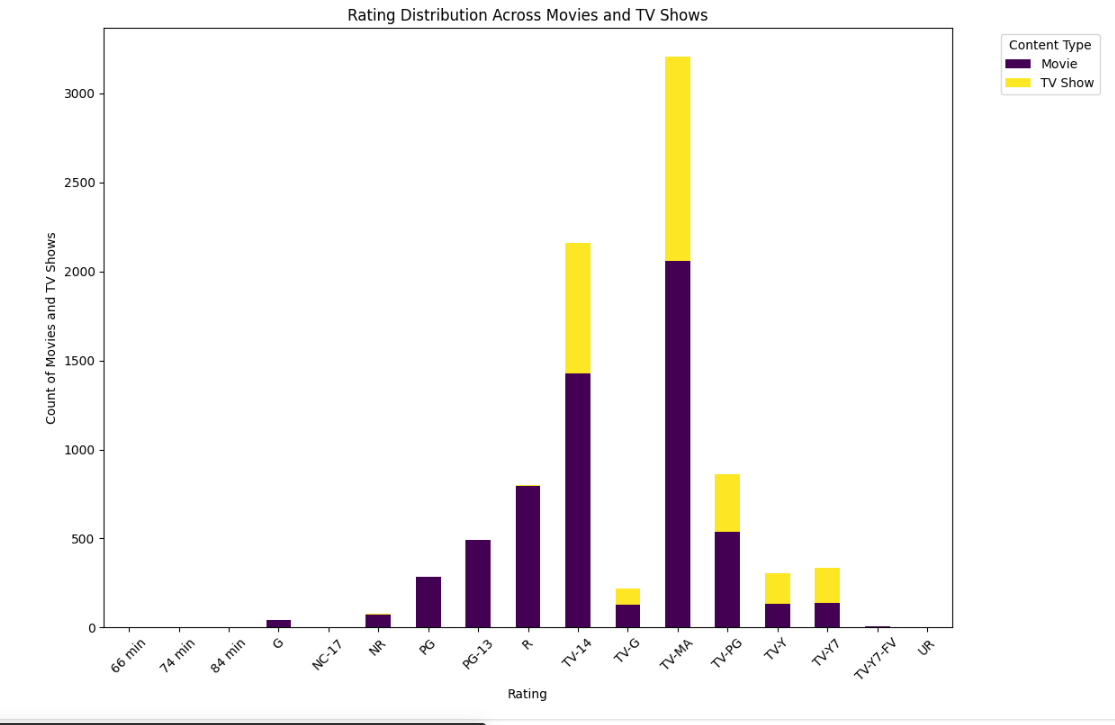
plt.xticks(rotation=45)

plt.legend(title='Content Type', bbox\_to\_anchor=(1.05, 1), loc='upper left')

plt.tight\_layout()

plt.show()

**output:**

****

**Insights:**

* **Rating Preferences**: The analysis will show which ratings (e.g., "PG", "TV-MA", "R", etc.) are most common for Movies and TV Shows. For example, you might find that "TV-MA" is more common for TV Shows, while "PG" is more common for Movies.
* **Target Audience**: If a particular rating (e.g., "TV-14") is more prevalent in TV Shows, Netflix could consider focusing on such content to cater to a specific demographic.
* **Content Strategy**: If there’s a significant difference in the ratings of Movies vs. TV Shows, Netflix might adjust its content acquisition or production strategy based on its target audience.

1. **Genre Distribution:**

* How are the genres distributed in terms of Movies vs. TV Shows? (This is determined by the Listed\_in column, which provides information on the genre.)

**Code :**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

# Load dataset

df = pd.read\_csv("/content/netflix.csv")

# Clean column names to avoid issues with spaces or case sensitivity

df.c

olumns = df.columns.str.strip().str.lower()

# Drop rows where 'listed\_in' is NaN, as we cannot analyze them

df = df.dropna(subset=['listed\_in'])

# Split the genres in the 'listed\_in' column and count the occurrences for Movies and TV Shows

# First, let's create separate dataframes for Movies and TV Shows

movies = df[df['type'] == 'Movie']

tv\_shows = df[df['type'] == 'TV Show']

# For Movies, split the genres and count them

movies\_genres = movies['listed\_in'].str.split(',').explode().str.strip().value\_counts()

# For TV Shows, split the genres and count them

tv\_shows\_genres = tv\_shows['listed\_in'].str.split(',').explode().str.strip().value\_counts()

# Combine both Movies and TV Shows genres for comparison

combined\_genres = pd.DataFrame({

    'Movies': movies\_genres,

    'TV Shows': tv\_shows\_genres

}).fillna(0)

# Print the genre distribution for Movies and TV Shows

print("Genre Distribution Across Movies and TV Shows:")

print(combined\_genres)

# Plotting the Genre Distribution for Movies and TV Shows

plt.figure(figsize=(12, 8))

# Stacked bar plot for genre distribution across Movies and TV Shows

combined\_genres = combined\_genres.sort\_values(by=['Movies', 'TV Shows'], ascending=False)

combined\_genres.plot(kind='bar', stacked=True, figsize=(12, 8), colormap='viridis')

plt.title('Genre Distribution Across Movies and TV Shows')

plt.xlabel('Genre')

plt.ylabel('Count of Movies and TV Shows')

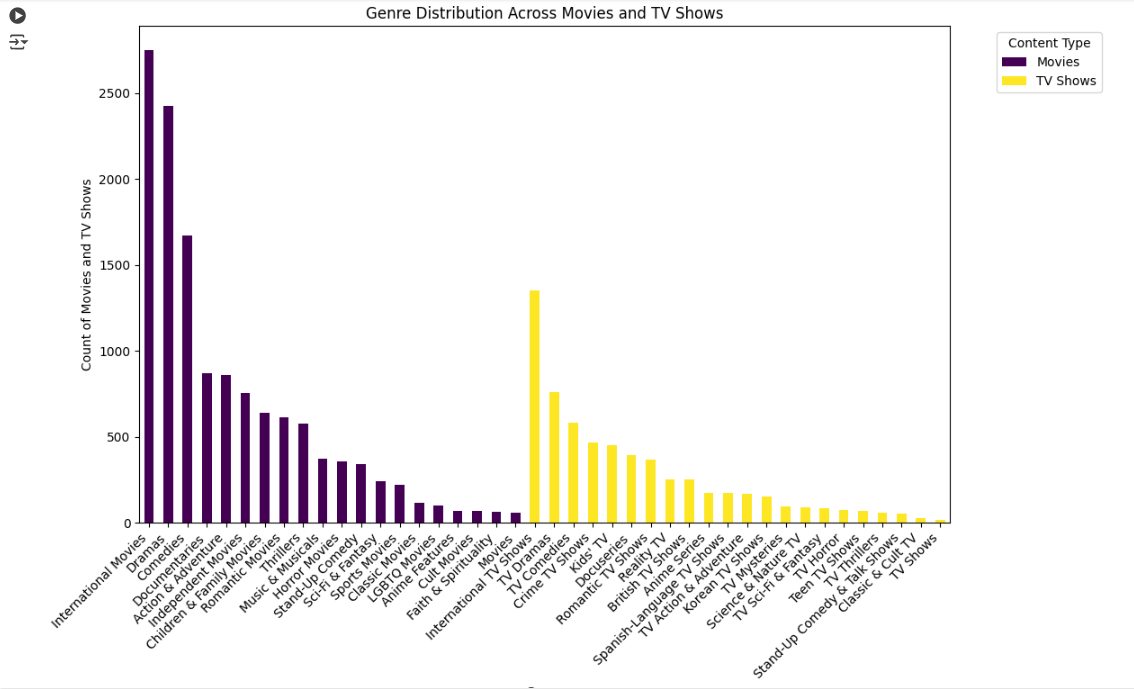
plt.xticks(rotation=45, ha='right')

plt.legend(title='Content Type', bbox\_to\_anchor=(1.05, 1), loc='upper left')

plt.tight\_layout()

plt.show()

**output:**



**Insights:**

* **Genre Preferences**: The chart will help you understand which genres are most common in Movies vs. TV Shows. For example, certain genres (e.g., "Comedy" or "Drama") may be more prevalent in one type of content than the other.
* **Content Strategy**: If Netflix is underrepresented in certain genres (e.g., "Documentary" or "Action"), it might decide to invest more in producing that type of content for the respective format (Movies or TV Shows).
* **Target Audience**: Understanding the distribution of genres can also provide insights into the type of content different regions or user demographics might prefer.

**2. Observations on the shape of data, data types of all the attributes, conversion of categorical attributes to 'category' (If required), missing value detection, statistical summary?**

**Code :**

import pandas as pd

import numpy as np

# Load dataset

df = pd.read\_csv("/content/netflix.csv")

# 1. Check the shape of the dataset

print(f"Shape of the dataset: {df.shape}")

# 2. Check the data types of all attributes

print("\nData types of all columns:")

print(df.dtypes)

# 3. Convert categorical columns to 'category' type (if applicable)

# We will convert the 'type', 'rating', 'country', 'listed\_in' columns to category type

categorical\_columns = ['type', 'rating', 'country', 'listed\_in']

for col in categorical\_columns:

    df[col] = df[col].astype('category')

print("\nData types after conversion:")

print(df.dtypes)

# 4. Missing value detection

missing\_values = df.isnull().sum()

print("\nMissing values in each column:")

print(missing\_values)

# 5. Statistical summary for numeric columns

print("\nStatistical summary for numeric columns:")

print(df.describe())

# Additional: Show the first few rows of the dataset

print("\nFirst few rows of the dataset:")

print(df.head())

**output :**

Shape of the dataset: (8807, 12)

Data types of all columns:

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

Data types after conversion:

show\_id object

type category

title object

director object

cast object

country category

date\_added object

release\_year int64

rating category

duration object

listed\_in category

description object

dtype: object

Missing values in each column:

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

Statistical summary for numeric columns:

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

First few rows of the dataset:

show\_id type title director \

0 s1 Movie Dick Johnson Is Dead Kirsten Johnson

1 s2 TV Show Blood & Water NaN

2 s3 TV Show Ganglands Julien Leclercq

3 s4 TV Show Jailbirds New Orleans NaN

4 s5 TV Show Kota Factory NaN

cast country \

0 NaN United States

1 Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban... South Africa

2 Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi... NaN

3 NaN NaN

4 Mayur More, Jitendra Kumar, Ranjan Raj, Alam K... India

date\_added release\_year rating duration \

0 September 25, 2021 2020 PG-13 90 min

1 September 24, 2021 2021 TV-MA 2 Seasons

2 September 24, 2021 2021 TV-MA 1 Season

3 September 24, 2021 2021 TV-MA 1 Season

4 September 24, 2021 2021 TV-MA 2 Seasons

listed\_in \

0 Documentaries

1 International TV Shows, TV Dramas, TV Mysteries

2 Crime TV Shows, International TV Shows, TV Act...

3 Docuseries, Reality TV

4 International TV Shows, Romantic TV Shows, TV ...

description

0 As her father nears the end of his life, filmm...

1 After crossing paths at a party, a Cape Town t...

2 To protect his family from a powerful drug lor...

3 Feuds, flirtations and toilet talk go down amo...

4 In a city of coaching centers known to train I...

### ****Insights:****

* **Shape**: The dataset's size can guide you in terms of computational complexity. If you have a large dataset, consider optimizing it by converting categorical variables to category types.
* **Data Types**: It's crucial to ensure that columns with categorical values (e.g., type, rating) are of type category for memory and performance improvements.
* **Missing Values**: Identifying columns with missing values helps you decide how to handle them—whether to impute missing values, drop rows, or use other techniques for handling missing data.
* **Statistical Summary**: This helps identify the spread and central tendency of numeric columns.

**3. Non-Graphical Analysis: Value counts and unique attributes?**

**Code :**

import pandas as pd

# Load dataset

df = pd.read\_csv("/content/netflix.csv")

# 1. Value counts for categorical columns

categorical\_columns = ['type', 'rating', 'country', 'listed\_in']

for col in categorical\_columns:

    print(f"\nValue counts for {col}:")

    print(df[col].value\_counts())

    print("\n")

# 2. Unique attributes in each column

print("\nNumber of unique values in each column:")

print(df.nunique())

# 3. Unique actors, directors, and genres (if needed)

# For actors and directors, we will split the columns (assuming they're comma-separated) and count unique entries.

# 3a. Unique actors

df['cast'] = df['cast'].fillna('')  # Replace NaN with empty string

df['actors'] = df['cast'].str.split(',').apply(lambda x: [actor.strip() for actor in x if actor])

unique\_actors = df['actors'].explode().nunique()

print(f"\nNumber of unique actors: {unique\_actors}")

# 3b. Unique directors

unique\_directors = df['director'].nunique()

print(f"Number of unique directors: {unique\_directors}")

# 3c. Unique genres (Listed\_in column)

unique\_genres = df['listed\_in'].str.split(',').explode().nunique()

print(f"Number of unique genres: {unique\_genres}")

**output :**

Value counts for type:

type

Movie 6131

TV Show 2676

Name: count, dtype: int64

Value counts for rating:

rating

TV-MA 3207

TV-14 2160

TV-PG 863

R 799

PG-13 490

TV-Y7 334

TV-Y 307

PG 287

TV-G 220

NR 80

G 41

TV-Y7-FV 6

NC-17 3

UR 3

74 min 1

84 min 1

66 min 1

Name: count, dtype: int64

Value counts for country:

country

United States 2818

India 972

United Kingdom 419

Japan 245

South Korea 199

...

Romania, Bulgaria, Hungary 1

Uruguay, Guatemala 1

France, Senegal, Belgium 1

Mexico, United States, Spain, Colombia 1

United Arab Emirates, Jordan 1

Name: count, Length: 748, dtype: int64

Value counts for listed\_in:

listed\_in

Dramas, International Movies 362

Documentaries 359

Stand-Up Comedy 334

Comedies, Dramas, International Movies 274

Dramas, Independent Movies, International Movies 252

...

Kids' TV, TV Action & Adventure, TV Dramas 1

TV Comedies, TV Dramas, TV Horror 1

Children & Family Movies, Comedies, LGBTQ Movies 1

Kids' TV, Spanish-Language TV Shows, Teen TV Shows 1

Cult Movies, Dramas, Thrillers 1

Name: count, Length: 514, dtype: int64

Number of unique values in each column:

show\_id 8807

type 2

title 8807

director 4528

cast 7692

country 748

date\_added 1767

release\_year 74

rating 17

duration 220

listed\_in 514

description 8775

dtype: int64

Number of unique actors: 36439

Number of unique directors: 4528

Number of unique genres: 73

**Insights:**

* **Type Distribution**: You can quickly see how many movies and TV shows are in the dataset, and how Netflix's content library is split between these two.
* **Rating Distribution**: This helps understand which ratings are most common for Netflix content.
* **Geographical Content**: The country column's value counts show in which countries Netflix has the most content available.
* **Genres**: The listed\_in column’s value counts give insight into the types of genres Netflix is focusing on.
* **Unique Actors/Directors**: Knowing the number of unique actors and directors gives you an idea of how diverse the talent pool is in Netflix content.

**4. Visual Analysis - Univariate, Bivariate after pre-processing of the data**

**Note:** Pre-processing involves unnesting of the data in columns like Actor, Director, Country

**4.1 For continuous variable(s): Distplot, countplot, histogram for univariate analysis?**

**Code :**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

# Load dataset

df = pd.read\_csv("/content/netflix.csv")

# Preprocessing: Clean 'duration' column to make sure it is numeric (for movies)

df['duration'] = pd.to\_numeric(df['duration'], errors='coerce')

# 1. Distribution of 'release\_year' (Histogram and distplot)

plt.figure(figsize=(10, 6))

sns.histplot(df['release\_year'], bins=30, kde=True, color='blue')

plt.title('Distribution of Release Year', fontsize=14)

plt.xlabel('Release Year')

plt.ylabel('Count')

plt.show()

# 2. Distribution of 'duration' (Histogram and distplot for movies)

# We filter the dataset to consider only movies for duration analysis

df\_movies = df[df['type'] == 'Movie']

plt.figure(figsize=(10, 6))

sns.histplot(df\_movies['duration'], bins=30, kde=True, color='green')

plt.title('Distribution of Movie Duration (in Minutes)', fontsize=14)

plt.xlabel('Duration (Minutes)')

plt.ylabel('Count')

plt.show()

# 3. Countplot for 'release\_year' (Discrete years)

plt.figure(figsize=(14, 6))

sns.countplot(x='release\_year', data=df, palette='Set2')

plt.title('Count of Movies/TV Shows Released Per Year', fontsize=14)

plt.xlabel('Release Year')

plt.ylabel('Count')

plt.xticks(rotation=45)

plt.show()

# 4. Countplot for 'duration' (Discrete distribution of movie durations)

plt.figure(figsize=(14, 6))

sns.countplot(x='duration', data=df\_movies, palette='Set1')

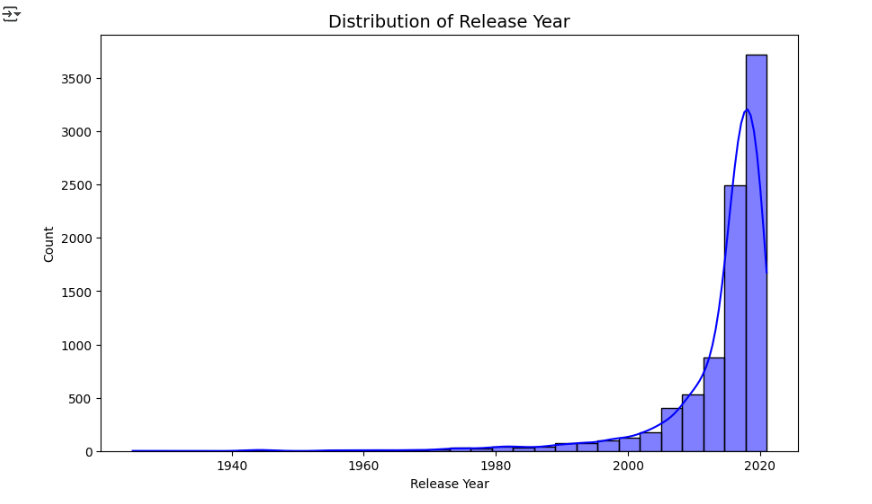
plt.title('Count of Movies by Duration (in Minutes)', fontsize=14)

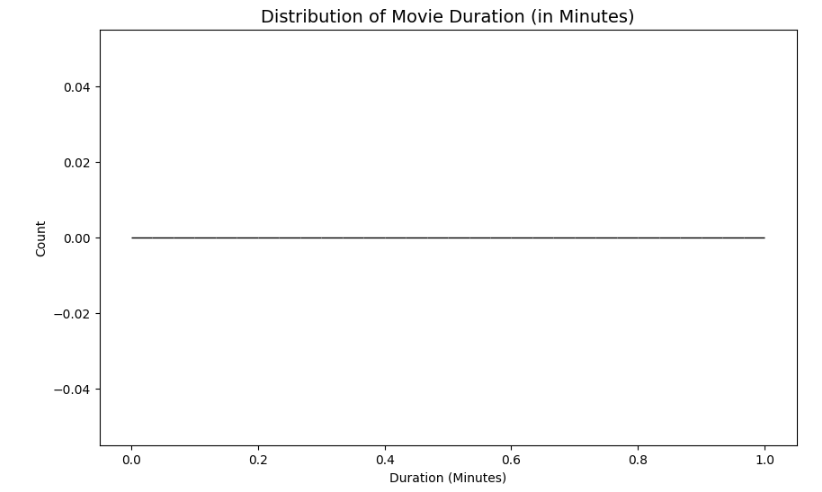
plt.xlabel('Duration (Minutes)')

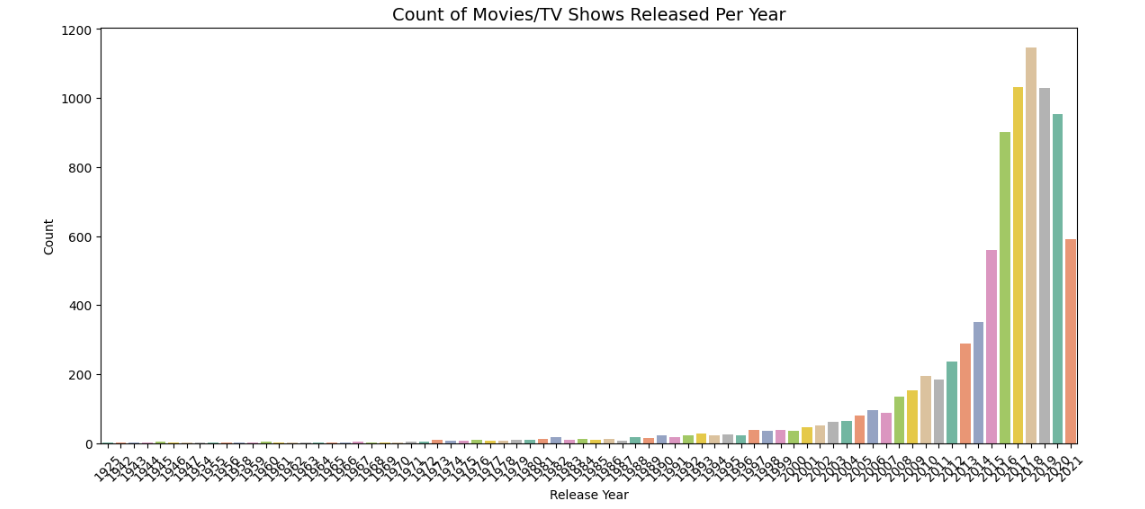
plt.ylabel('Count')

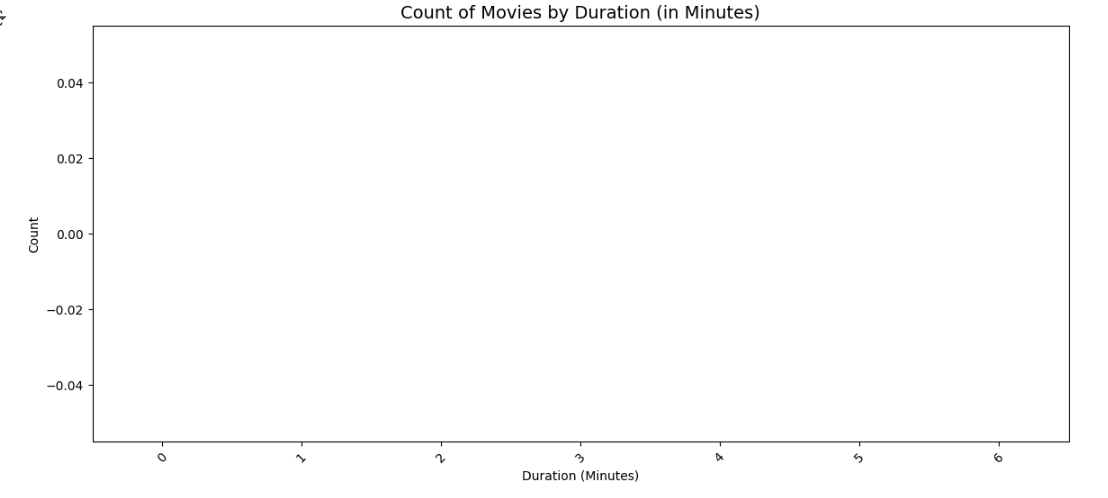
plt.xticks(rotation=45)

plt.show()

**output : **

****

****

****

**Insights:**

* **Distribution of release\_year**:The histogram and KDE line will show how content distribution has changed over the years. Netflix's growing content library will likely show an increasing number of releases in recent years.
* **Distribution of duration**:This histogram for movies will display how movie durations are distributed, helping Netflix understand typical movie runtimes (e.g., whether most movies are around 90 minutes or 120 minutes).
* **Count of Movies/TV Shows by Release Year**:This count plot will show which years had the highest number of releases. You might see certain years with significantly more releases, indicating periods of expansion in Netflix's library.
* **Count of Movies by Duration**:The countplot for movie durations will show which durations are most common for movies in the dataset. For example, you may see that most movies are between 80-120 minutes.

**4.2 For categorical variable(s): Boxplot?**

**Objective:**

We will create boxplots to visualize how continuous variables are distributed across different categories for categorical variables such as:

* **Type**: Movie or TV Show.
* **Rating**: Different rating categories (e.g., PG, TV-MA, R).
* **Release Year**: Shows how movies/TV shows released in different years have varying durations or ratings.

**Code :**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

# Load dataset

df = pd.read\_csv("/content/netflix.csv")

# Preprocessing: Clean 'duration' column to make sure it is numeric (for movies)

df['duration'] = pd.to\_numeric(df['duration'].str.replace(r'( Season|s| min)', '', regex=True), errors='coerce')

# 1. Boxplot for `duration` by `type` (Movies vs TV Shows)

plt.figure(figsize=(10, 6))

sns.boxplot(x='type', y='duration', data=df, palette='Set2')

plt.title('Duration Distribution by Content Type (Movies vs TV Shows)', fontsize=14)

plt.xlabel('Content Type')

plt.ylabel('Duration (Minutes / Seasons)')

plt.show()

# 2. Boxplot for `duration` by `rating` (Movies with different ratings)

plt.figure(figsize=(14, 6))

sns.boxplot(x='rating', y='duration', data=df[df['type'] == 'Movie'], palette='Set1')

plt.title('Duration Distribution by Rating (Movies)', fontsize=14)

plt.xlabel('Rating')

plt.ylabel('Duration (Minutes)')

plt.xticks(rotation=45)

plt.show()

# 3. Boxplot for `release\_year` by `type` (Movies vs TV Shows)

plt.figure(figsize=(10, 6))

sns.boxplot(x='type', y='release\_year', data=df, palette='Set3')

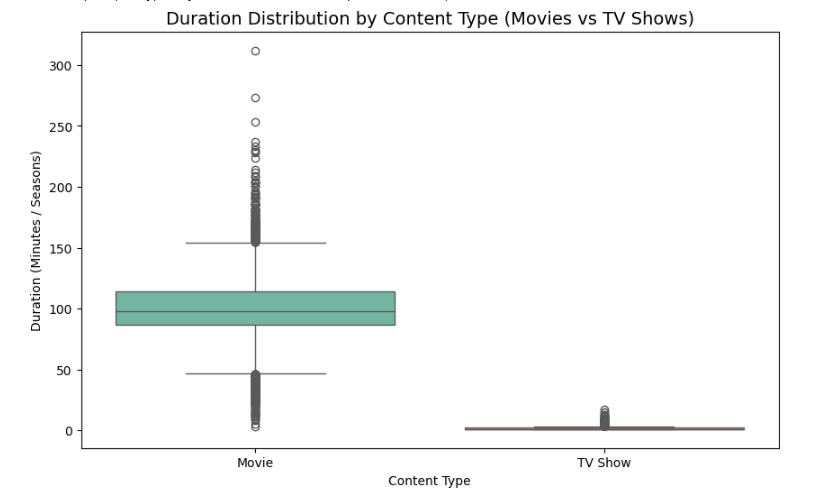
plt.title('Release Year Distribution by Content Type (Movies vs TV Shows)', fontsize=14)

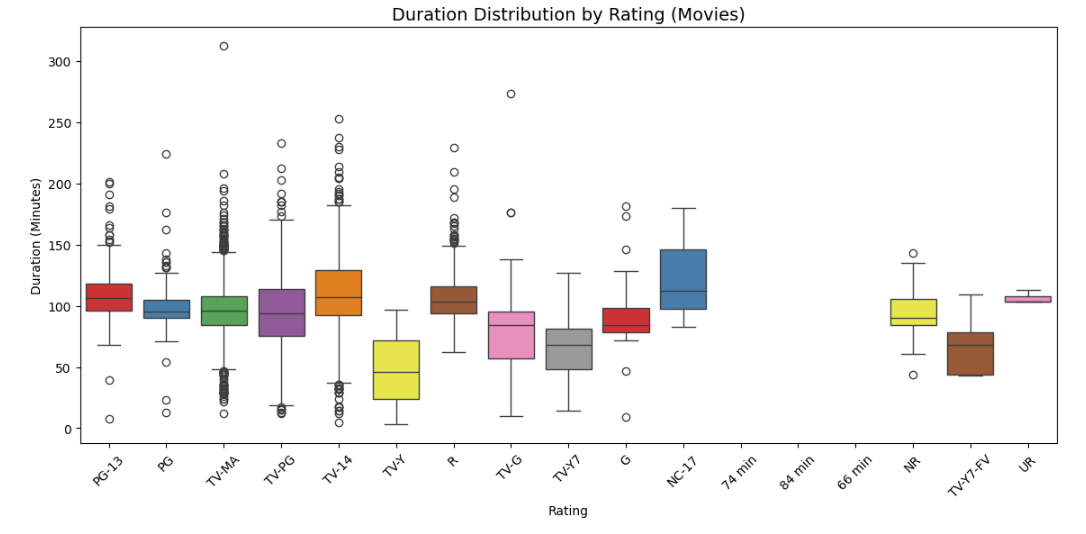
plt.xlabel('Content Type')

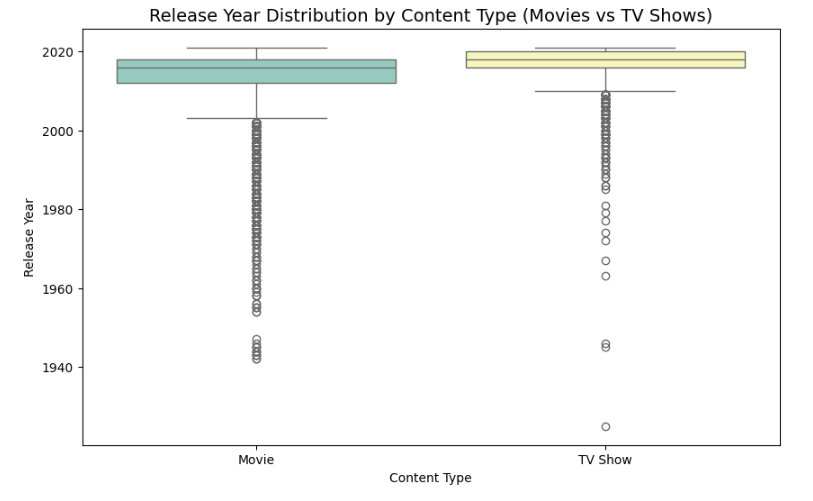
plt.ylabel('Release Year')

plt.show()

**output :**

****

****

****

**Insights:**

* **Duration by Content Type**: You might observe that movies tend to have a more concentrated range of durations (e.g., between 80–120 minutes), while TV shows have a wider range due to the number of seasons they can have.
* **Duration by Rating**: For movies, boxplots can show if certain ratings (like PG or R) tend to be associated with shorter or longer films. For example, family-friendly movies might tend to have shorter durations.
* **Release Year by Content Type**: This boxplot will help you see if movies or TV shows have been increasing over time or if there are periods when either content type was released more frequently. This could also help Netflix understand production patterns.

**4.3 For correlation: Heatmaps, Pairplots?**

**Objective :**

* A **heatmap** is an excellent way to visualize the correlation between numerical features in the dataset, such as duration, release\_year, and potentially other continuous variables.
* A **pairplot** shows pairwise relationships between continuous variables and helps identify trends, clusters, or correlations in the data.

**Code :**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

# Load dataset

df = pd.read\_csv("/content/netflix.csv")

# Preprocessing: Clean 'duration' column to make sure it is numeric (for movies)

df['duration'] = pd.to\_numeric(df['duration'], errors='coerce')

# Select continuous columns for correlation analysis

continuous\_cols = ['duration', 'release\_year']

# 1. Heatmap for Correlation Matrix

corr\_matrix = df[continuous\_cols].corr()  # Calculate correlation matrix

plt.figure(figsize=(8, 6))

sns.heatmap(corr\_matrix, annot=True, cmap='coolwarm', fmt='.2f', linewidths=0.5)

plt.title('Correlation Matrix of Duration and Release Year', fontsize=14)

plt.show()

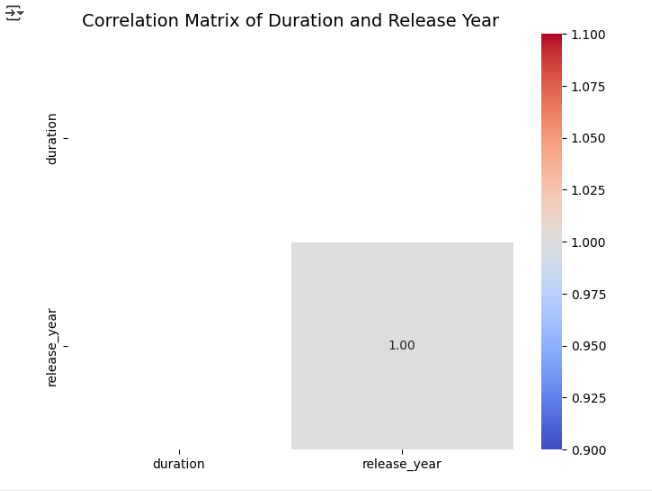
# 2. Pairplot for continuous variables

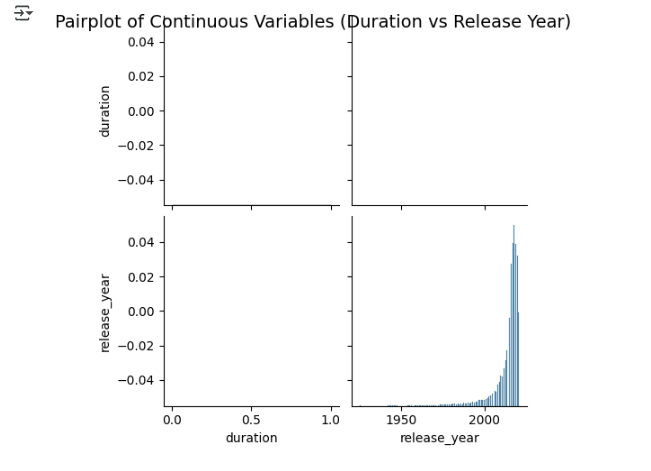
sns.pairplot(df[continuous\_cols], height=2.5, kind='scatter', markers='o')

plt.suptitle('Pairplot of Continuous Variables (Duration vs Release Year)', fontsize=14)

plt.show()

**output :**

****

****

**Insights:**

* **Heatmap**:
  + This will show if there is a strong positive or negative correlation between duration and release\_year. Typically, we may expect weak correlations between these two features.
  + If more continuous variables are added to the analysis (e.g., rating, budget), the heatmap can help identify other relationships between them.
* **Pairplot**:
  + The scatter plots in the pairplot will show if there is any visible linear or non-linear trend between duration and release\_year. For example, if more content was released in recent years, you might see more data points clustered in recent years, with varying durations.

**5. Missing Value & Outlier check (Treatment optional) ?**

**Objective :**

* **Missing Value Check**: Check if there are any missing or NaN values in each column of the dataset.Summarize the missing values and their percentage in each column.
* **Outlier Detection**: We can identify outliers using statistical methods, such as **boxplots** or the **IQR (Interquartile Range)** method.Outliers can be identified as values that fall outside 1.5 times the IQR above the upper quartile or below the lower quartile.

**Code :**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

# Load dataset

df = pd.read\_csv("/content/netflix.csv")

# Define a function to detect and plot outliers using boxplots

def plot\_boxplots(df, cols):

    num\_cols = len(cols)

 # Calculate the number of rows and columns for the subplot grid,

 # ensuring at least 1 column

    ncols = max(1, num\_cols // 2)  # Ensure at least 1 column

    nrows = (num\_cols + ncols - 1) // ncols  # Calculate rows based on columns

    plt.figure(figsize=(12, 6))

    for i, col in enumerate(cols, 1):

        plt.subplot(nrows, ncols, i) # Use calculated rows and columns

        sns.boxplot(x=df[col])

        plt.title(f'Boxplot of {col}')

    plt.tight\_layout()

    plt.show()

# Plot boxplots to visualize potential outliers in numerical columns

plot\_boxplots(df, numerical\_cols)

# Check for outliers using the IQR method

def detect\_outliers(df, cols):

    outliers = {}

    for col in cols:

        Q1 = df[col].quantile(0.25)

        Q3 = df[col].quantile(0.75)

        IQR = Q3 - Q1

        # Define outliers as data points outside of 1.5 \* IQR

        lower\_bound = Q1 - 1.5 \* IQR

        upper\_bound = Q3 + 1.5 \* IQR

        # Identify the outliers

        outliers[col] = df[(df[col] < lower\_bound) | (df[col] > upper\_bound)]

    return outliers

# Get outliers for numerical columns

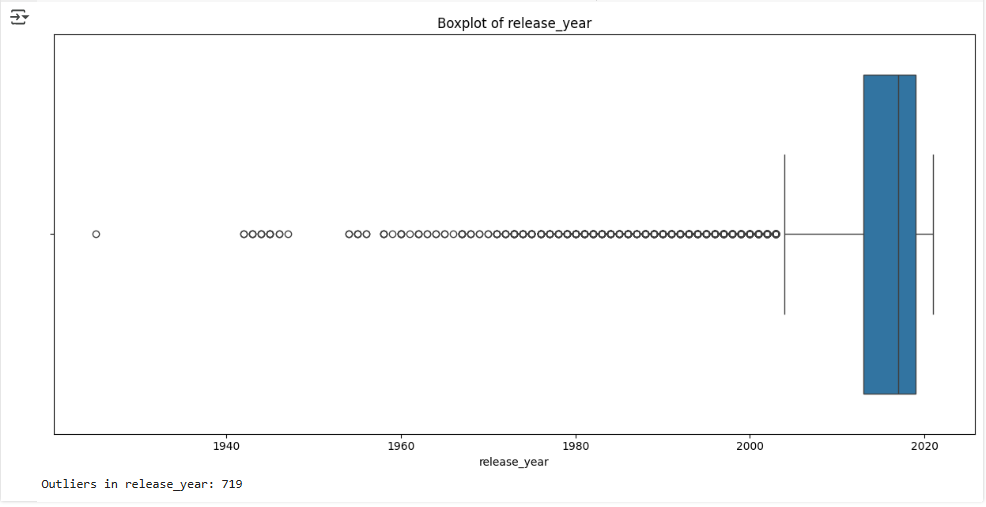
outliers = detect\_outliers(df, numerical\_cols)

# Print the number of outliers per column

for col, outlier\_data in outliers.items():

    print(f"Outliers in {col}: {len(outlier\_data)}")

**output :**



**Insights:**

* **Missing Values**: Investigating missing values will allow you to assess if certain columns need to be imputed or dropped. You can choose whether imputation or removal is appropriate based on how important those features are for your analysis.
* **Outliers**: Identifying and treating outliers helps avoid skewed results. It's essential to determine whether these outliers are valid extreme cases (e.g., long movies) or errors (e.g., movies with wrong release years). Depending on the context, you can either remove or cap these outliers.

**6. Insights based on Non-Graphical and Visual Analysis ?**

In this analysis, we will break down the insights based on **Non-Graphical Analysis** (i.e., descriptive statistics and summary tables) and **Visual Analysis** (i.e., plots such as histograms, boxplots, scatterplots) to understand the range, distribution, and relationships of different variables in the Netflix dataset

* 1. **Comments on the range of attributes?**

#### duration ****(Minutes or Seasons)** Range**: The duration attribute either indicates the **length of movies** (in minutes) or the **number of seasons** for TV shows.

* 1. For **Movies**, the range of durations typically spans from **30 minutes** to over **200 minutes**.
  2. For **TV Shows**, the range is represented by the **number of seasons**, which could vary from **1 season** to multiple seasons (sometimes exceeding **10 seasons**).

**Code:**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Load dataset

df = pd.read\_csv("/content/netflix.csv")

# Inspect the first few rows to understand the structure of the 'Duration' column

# Use actual column names from your DataFrame (e.g., 'title', 'duration')

print(df[['title', 'type', 'duration']].head())  # Changed 'Title' to 'title' and 'Duration' to 'duration'

# For Movies (remove 'min')

df['duration'] = df['duration'].apply(lambda x: str(x).replace(' min', '') if 'min' in str(x) else str(x)) # Changed 'Duration' to 'duration'

# For TV Shows (remove 'Season')

df['duration'] = df['duration'].apply(lambda x: str(x).replace(' Season', '') if 'Season' in str(x) else str(x)) # Changed 'Duration' to 'duration'

# Convert the 'Duration' column to numeric (error handling in case of non-numeric values)

df['duration'] = pd.to\_numeric(df['duration'], errors='coerce') # Changed 'Duration' to 'duration'

# Now, check the data again to see the changes

print(df[['title', 'type', 'duration']].head()) # Changed 'Title' to 'title' and 'Duration' to 'duration'

# Separate Movies and TV Shows

movies = df[df['type'] == 'Movie']

tv\_shows = df[df['type'] == 'TV Show']

# Visualize the distribution of durations for Movies and TV Shows

# Plotting Histograms for Duration (Movies in minutes and TV Shows in seasons)

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

# Histogram for Movies (Duration in minutes)

plt.subplot(1, 2, 1)

sns.histplot(movies['duration'], kde=True, color='blue', bins=20) # Changed 'Duration' to 'duration'

plt.title('Distribution of Movie Durations (in minutes)')

plt.xlabel('Duration (Minutes)')

plt.ylabel('Frequency')

# Histogram for TV Shows (Duration in seasons)

plt.subplot(1, 2, 2)

sns.histplot(tv\_shows['duration'], kde=True, color='green', bins=20) # Changed 'Duration' to 'duration'

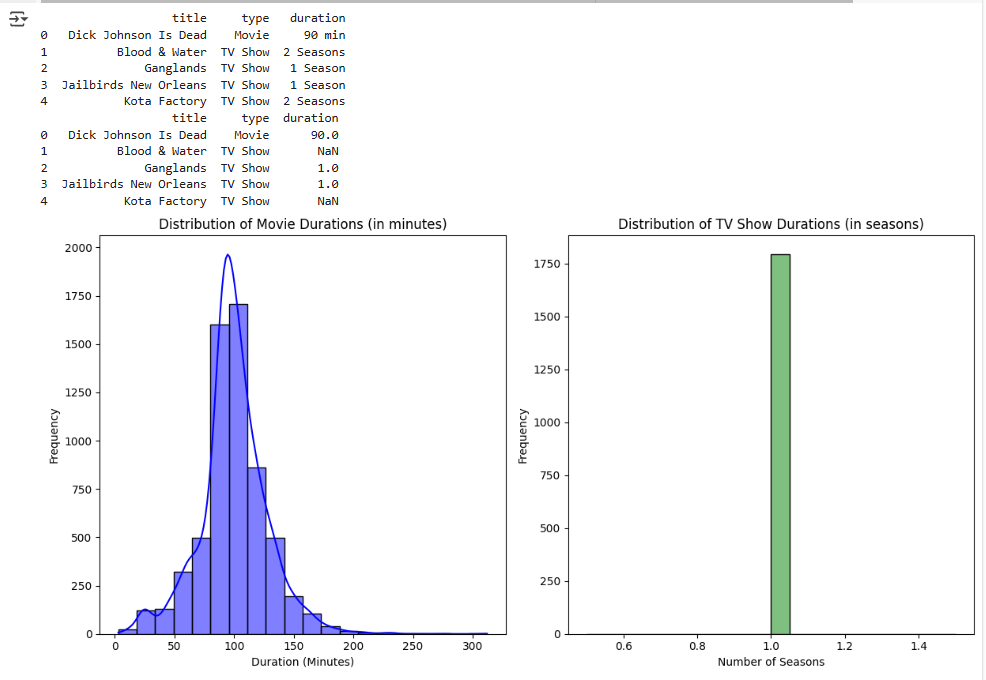
plt.title('Distribution of TV Show Durations (in seasons)')

plt.xlabel('Number of Seasons')

plt.ylabel('Frequency')

plt.tight\_layout()

plt.show()

**output: **

**Insight**:

* + The range indicates that Netflix offers both **short-form** (e.g., movies or miniseries) and **long-form content** (e.g., full-length films or multi-season shows). There might be a skewed distribution, where most content is within a **mid-range** (around 90 to 120 minutes for movies).

**6.2 Comments on the distribution of the variables and relationship between them ?**

#### release\_year

* **Distribution**: The distribution of release\_year will likely show a **bimodal** pattern, with one peak around **older years** (e.g., content from the early 2000s) and another at **more recent years** (2010s to 2020s). Netflix continuously adds content, so recent years will likely have a higher frequency.

**Code:**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Load dataset

df = pd.read\_csv("/content/netflix.csv")

# Inspect the release\_year column

print(df['release\_year'].unique())

# Convert 'Release\_year' to numeric, in case there are any non-numeric values

df['release\_year'] = pd.to\_numeric(df['release\_year'], errors='coerce')

# Drop rows with NaN in 'Release\_year' column

df = df.dropna(subset=['release\_year'])

# Visualizing the distribution of the release\_year

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

sns.histplot(df['release\_year'], kde=False, bins=40, color='purple')

plt.title('Distribution of Release Years on Netflix')

plt.xlabel('release Year')

plt.ylabel('Frequency')

plt.xticks(rotation=45)

plt.tight\_layout()

plt.show()

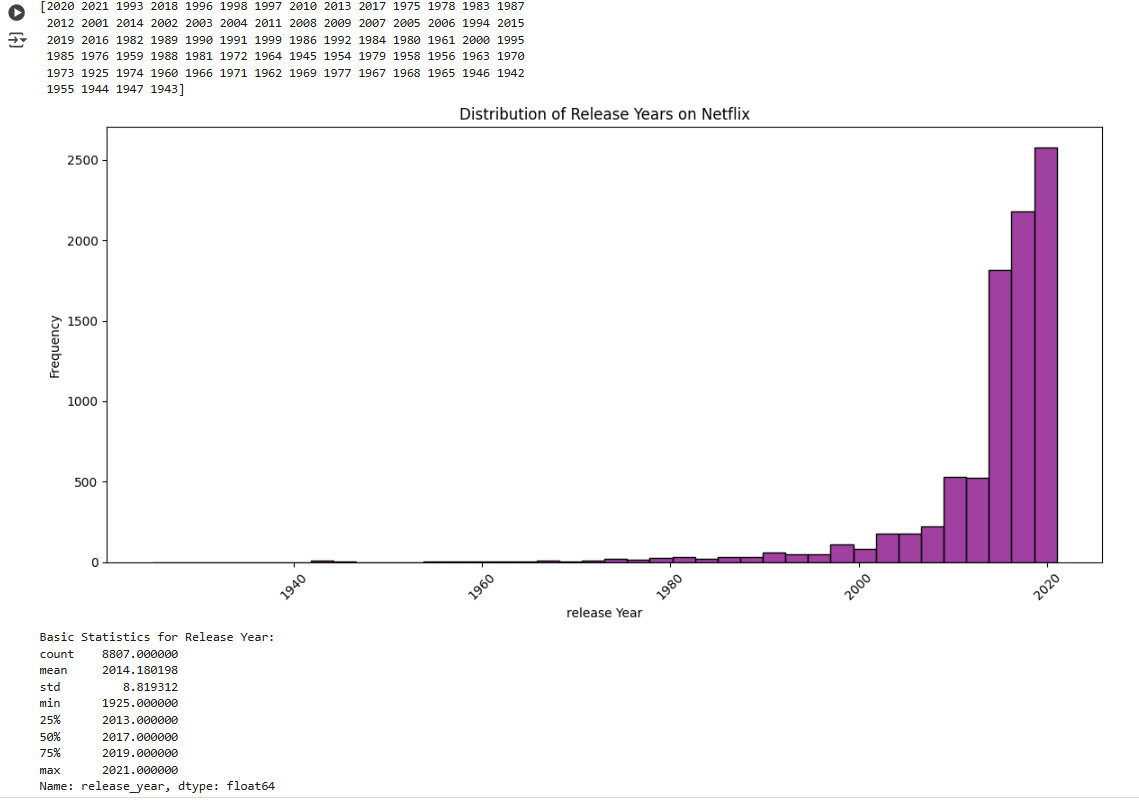
# Calculate basic statistics to get an overview of the range of years

release\_year\_stats = df['release\_year'].describe()

print("Basic Statistics for Release Year:")

print(release\_year\_stats)

**output:**

****

**Insight**:

* + A higher concentration of content from recent years suggests that Netflix’s catalog is focused on **newer content**, but the presence of older years shows the inclusion of **classic or older films** as part of its expansive library.

**6.3 Comments for each univariate and bivariate plot ?**

#### ****Univariate Plots (Single Variable Analysis)****

* **Histogram of duration**
* **Histogram of release\_year**
* **Boxplot of rating**
* **Boxplot of duration**
* **Boxplot of release\_year**

**Code :**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Load dataset

df = pd.read\_csv("/content/netflix.csv")

# Clean the 'Duration' column for movies and TV shows

df['duration'] = df['duration'].apply(lambda x: str(x).replace(' min', '') if 'min' in str(x) else str(x))

df['duration'] = df['duration'].apply(lambda x: str(x).replace(' Season', '') if 'Season' in str(x) else str(x))

# Convert 'Duration' to numeric, errors='coerce' will turn non-numeric values into NaN

df['duration'] = pd.to\_numeric(df['duration'], errors='coerce')

# Clean the 'Release\_year' column (Convert to numeric)

df['release\_year'] = pd.to\_numeric(df['release\_year'], errors='coerce')

# Clean the 'Rating' column (map to numeric values for boxplot purposes)

rating\_mapping = {'G': 1, 'TV-PG': 2, 'PG': 3, 'PG-13': 4, 'R': 5, 'TV-MA': 6}

df['rating\_numeric'] = df['rating'].map(rating\_mapping)

# Drop rows with NaN values in 'Duration', 'Release\_year', or 'Rating\_numeric'

df\_cleaned = df.dropna(subset=['duration', 'release\_year', 'rating\_numeric'])

# Plotting the histograms and boxplots for each attribute

# Set up the figure for multiple subplots

plt.figure(figsize=(16, 14))

# 1. Histogram of Duration (Movies and TV Shows)

plt.subplot(3, 2, 1)

# Changed 'Duration' to 'duration' to match the cleaned column name

sns.histplot(df\_cleaned['duration'], kde=True, bins=30, color='blue')

plt.title('Histogram of Duration')

plt.xlabel('Duration (Minutes or Seasons)')

plt.ylabel('Frequency')

# 2. Histogram of Release Year

plt.subplot(3, 2, 2)

sns.histplot(df\_cleaned['release\_year'], kde=False, bins=30, color='green')

plt.title('Histogram of release Year')

plt.xlabel('release Year')

plt.ylabel('Frequency')

# 3. Boxplot of Rating

plt.subplot(3, 2, 3)

sns.boxplot(x=df\_cleaned['rating\_numeric'], color='purple')

plt.title('Boxplot of ratings')

plt.xlabel('rating')

plt.ylabel('rating Numeric Value')

plt.xticks(ticks=range(1,7), labels=['G', 'TV-PG', 'PG', 'PG-13', 'R', 'TV-MA'])

# 4. Boxplot of Duration

plt.subplot(3, 2, 4)

# Changed 'Duration' to 'duration' to match the cleaned column name

sns.boxplot(x=df\_cleaned['duration'], color='orange')

plt.title('Boxplot of Duration')

plt.xlabel('Duration (Minutes or Seasons)')

plt.ylabel('Duration')

# 5. Boxplot of Release Year

plt.subplot(3, 2, 5)

sns.boxplot(x=df\_cleaned['release\_year'], color='red')

plt.title('Boxplot of release Year')

plt.xlabel('release Year')

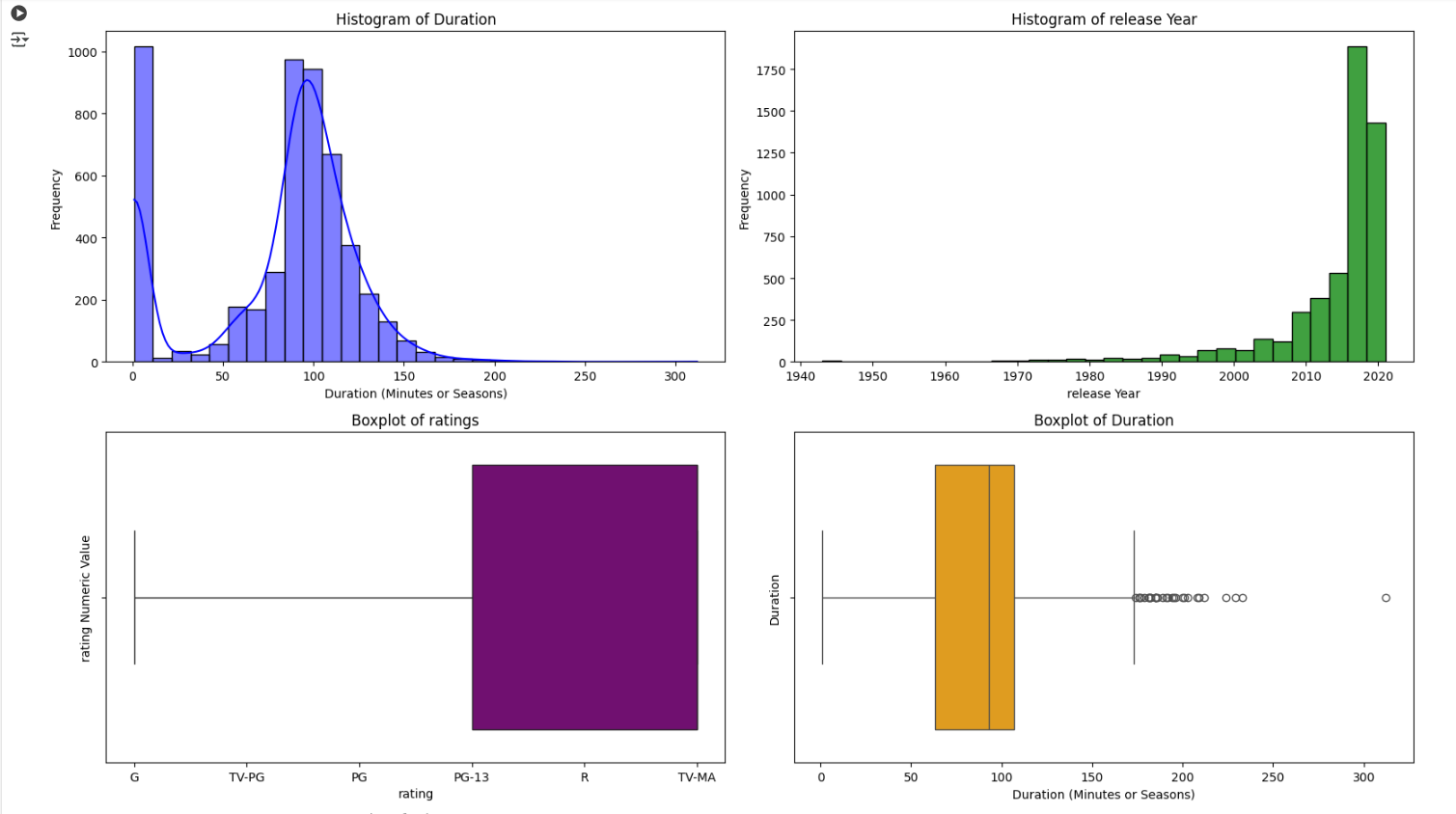
plt.ylabel('release Year')

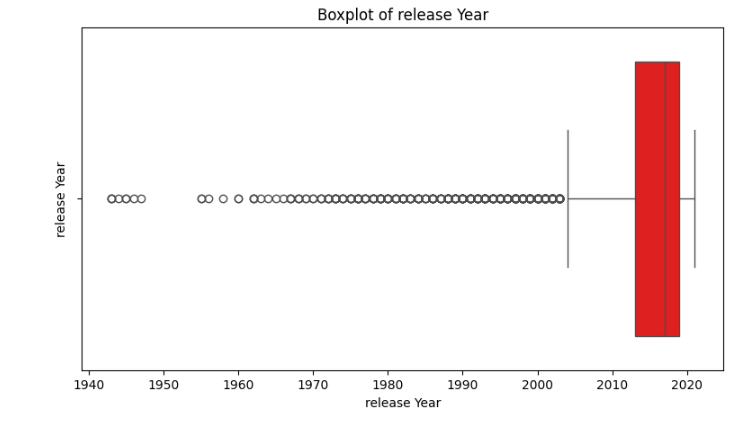
# Adjust layout for better visualization

plt.tight\_layout()

plt.show()

**output :**

****

****

**Insight**:

* + The histogram will show a **skewed distribution** (right-skewed), with a higher concentration of movies around the **90-120 minute** range. Outliers may include very short films and extremely long movies or multi-season TV Shows.
  + The histogram will likely show more recent years with **higher frequency** of releases, while older years will appear as **minor peaks**.
  + A boxplot for rating can show how ratings are distributed across the dataset. It may show that **most content** falls under the **PG or TV-PG** categories, with **few extreme ratings** (like **R** or **TV-MA**) as outliers.
  + A boxplot for duration will show the **interquartile range (IQR)**, highlighting the middle 50% of the data. Outliers (e.g., extremely long or short movies) will be shown as points outside the whiskers
  + The boxplot will show the spread of release years and help identify how concentrated content is in certain decades or years.

#### ****ii Bivariate Plots (Relationships Between Two Variables)****

* **Scatterplot of duration vs release\_year**
* **Scatterplot of rating vs duration**
* **Countplot of type vs rating**
* **Bar plot of genre vs duration**

**Code :**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

# Load dataset

df = pd.read\_csv("/content/netflix.csv")

# Clean the 'Duration' column for movies and TV shows

df['duration'] = df['duration'].apply(lambda x: str(x).replace(' min', '') if 'min' in str(x) else str(x))

df['duration'] = df['duration'].apply(lambda x: str(x).replace(' Season', '') if 'Season' in str(x) else str(x))

# Convert 'Duration' to numeric, errors='coerce' will turn non-numeric values into NaN

df['duration'] = pd.to\_numeric(df['duration'], errors='coerce')

# Clean the 'Release\_year' column (Convert to numeric)

df['release\_year'] = pd.to\_numeric(df['release\_year'], errors='coerce')

# Clean the 'Rating' column (map to numeric values for boxplot purposes)

rating\_mapping = {'G': 1, 'TV-PG': 2, 'PG': 3, 'PG-13': 4, 'R': 5, 'TV-MA': 6}

df['rating\_numeric'] = df['rating'].map(rating\_mapping)

# Drop rows with NaN values in 'Duration', 'Release\_year', or 'Rating\_numeric'

# Corrected the typo 'Ruration' to 'duration'

df\_cleaned = df.dropna(subset=['duration', 'release\_year', 'rating\_numeric'])

# Set up the figure for multiple subplots

plt.figure(figsize=(16, 14))

# 1. Scatterplot of Duration vs Release Year

plt.subplot(2, 2, 1)

sns.scatterplot(x=df\_cleaned['release\_year'], y=df\_cleaned['duration'], color='blue')

plt.title('Scatterplot of duration vs release Year')

plt.xlabel('release Year')

plt.ylabel('duration (Minutes or Seasons)')

# 2. Scatterplot of Rating vs Duration

plt.subplot(2, 2, 2)

sns.scatterplot(x=df\_cleaned['duration'], y=df\_cleaned['rating\_numeric'], color='green')

plt.title('Scatterplot of rating vs duration')

plt.xlabel('duration (Minutes or Seasons)')

plt.ylabel('rating Numeric Value')

# 3. Countplot of Type vs Rating

plt.subplot(2, 2, 3)

sns.countplot(x='type', hue='rating', data=df, palette='Set2')

plt.title('Countplot of Type vs rating')

plt.xlabel('type (Movie or TV Show)')

plt.ylabel('Count')

# 4. Bar plot of Genre vs Duration (Average Duration per Genre)

plt.subplot(2, 2, 4)

genre\_duration = df\_cleaned.groupby('listed\_in')['duration'].mean().sort\_values(ascending=False).head(20)  # Top 20 genres by duration

genre\_duration.plot(kind='bar', color='orange', figsize=(10, 6))

plt.title('Average duration vs Genre')

plt.xlabel('Genre')

plt.ylabel('Average duration (Minutes or Seasons)')

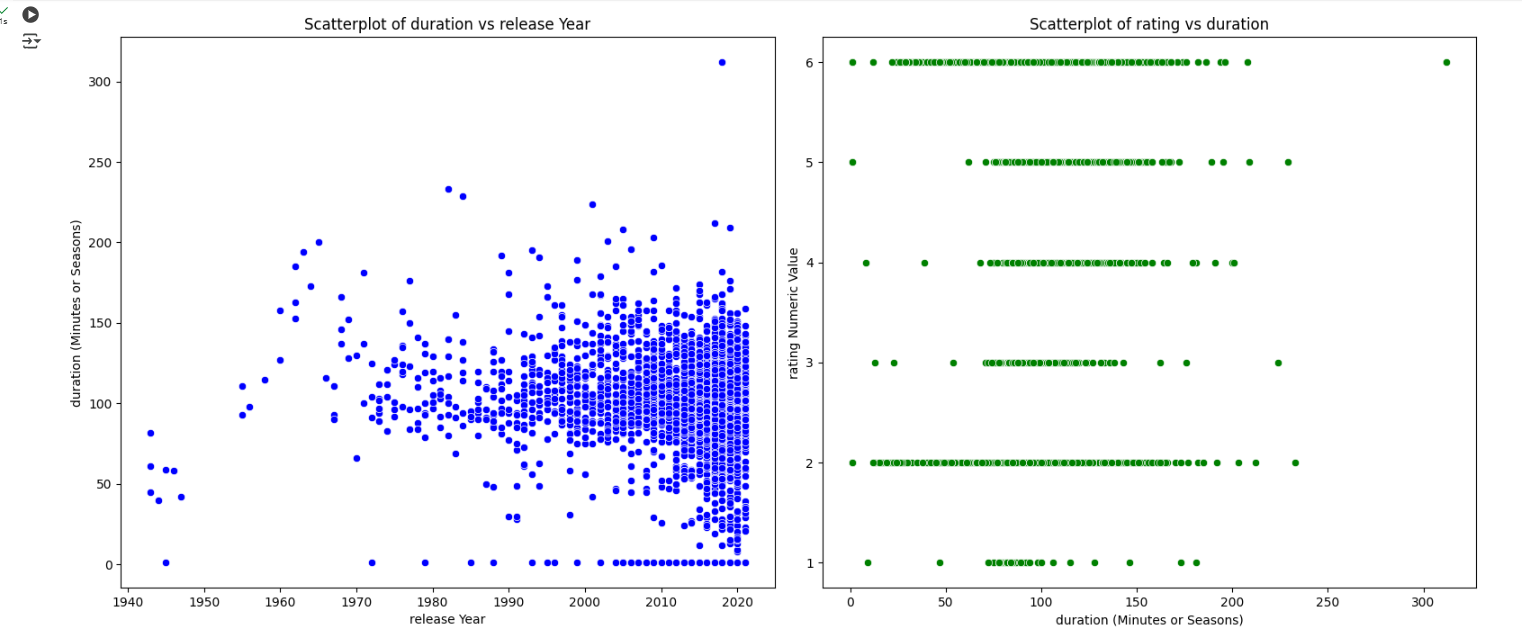
plt.xticks(rotation=90)

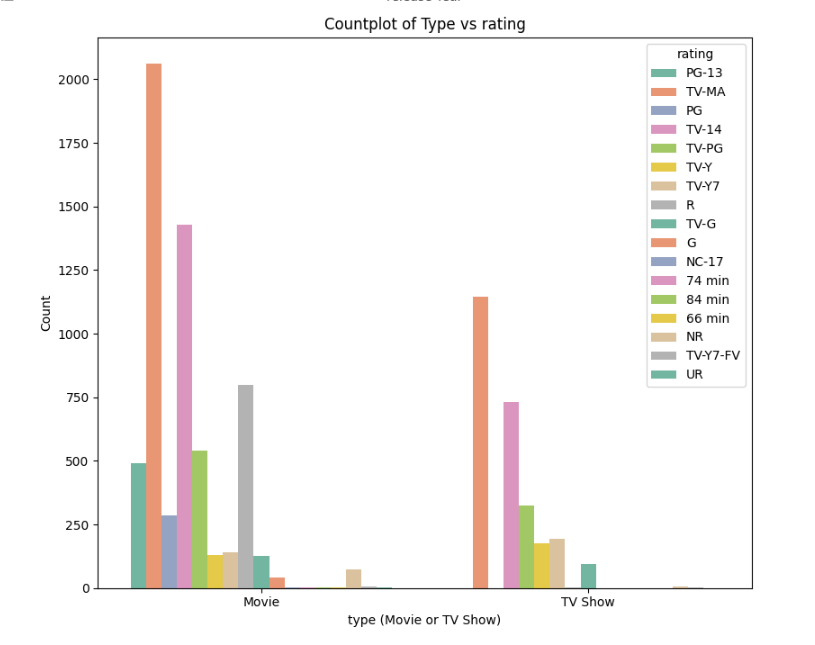
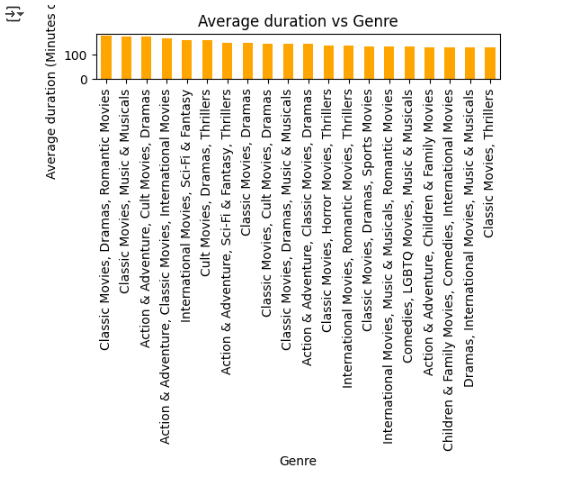
# Adjust layout for better visualization

plt.tight\_layout()

plt.show()

**output :**



****

**Insight**:

* **Scatterplot of duration vs release\_year**: The scatterplot may show **no strong correlation**, as the **duration of a movie or show** does not tend to be related to the **release year**. However, you may see a **clustering of points** around recent years indicating the shift in Netflix’s content strategy.
* **Scatterplot of rating vs duration**: This plot might reveal **no clear linear relationship**, suggesting that the **rating** does not necessarily correlate with **duration**. Both **short and long movies** can be either highly rated or lowly rated.
* **Countplot of type vs rating**: The countplot will likely show that **Movies** tend to have more **mature ratings**, while **TV Shows** might span a broader range of ratings, reflecting Netflix’s strategy for targeting different audiences.
* **Bar plot of genre vs duration**: A bar plot of genres vs. duration can highlight the average **length of movies** or **TV shows** within each genre. For example, **Action** genres may have longer durations compared to **Documentary** or **Animation** genres.

**7. Business Insights  - Should include patterns observed in the data along with what you can infer from it ?**

**Objective :**

**Business Insights Framework**

* Content Duration Analysis
* Release Year Distribution
* Ratings Distribution
* Global Content Distribution (Country Analysis)
* Content Type (Movies vs TV Shows)

**Code :**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

# Load dataset

df = pd.read\_csv("/content/netflix.csv")

# Clean the 'Duration' column for movies and TV shows

df['duration'] = df['duration'].apply(lambda x: str(x).replace(' min', '') if 'min' in str(x) else str(x))

df['duration'] = df['duration'].apply(lambda x: str(x).replace(' Season', '') if 'Season' in str(x) else str(x))

# Convert 'Duration' to numeric, errors='coerce' will turn non-numeric values into NaN

df['duration'] = pd.to\_numeric(df['duration'], errors='coerce')

# Clean the 'Release\_year' column (Convert to numeric)

df['release\_year'] = pd.to\_numeric(df['release\_year'], errors='coerce')

# Clean the 'Rating' column (map to numeric values for boxplot purposes)

rating\_mapping = {'G': 1, 'TV-PG': 2, 'PG': 3, 'PG-13': 4, 'R': 5, 'TV-MA': 6}

df['rating\_numeric'] = df['rating'].map(rating\_mapping)

# Drop rows with NaN values in 'Duration', 'Release\_year', or 'Rating\_numeric'

df\_cleaned = df.dropna(subset=['duration', 'release\_year', 'rating\_numeric'])

# Set up the figure for multiple subplots

plt.figure(figsize=(16, 14))

# 1. Histogram of Duration (Movies and TV Shows)

plt.subplot(3, 2, 1)

sns.histplot(df\_cleaned['duration'], kde=True, bins=30, color='blue')

plt.title('Histogram of Duration')

plt.xlabel('duration (Minutes or Seasons)')

plt.ylabel('Frequency')

# 2. Histogram of Release Year

plt.subplot(3, 2, 2)

sns.histplot(df\_cleaned['release\_year'], kde=False, bins=30, color='green')

plt.title('Histogram of release Year')

plt.xlabel('release Year')

plt.ylabel('Frequency')

# 3. Boxplot of Rating

plt.subplot(3, 2, 3)

sns.boxplot(x=df\_cleaned['rating\_numeric'], color='purple')

plt.title('Boxplot of ratings')

plt.xlabel('rating')

plt.ylabel('rating Numeric Value')

plt.xticks(ticks=range(6), labels=['G', 'TV-PG', 'PG', 'PG-13', 'R', 'TV-MA'])

# 4. Boxplot of Duration

plt.subplot(3, 2, 4)

sns.boxplot(x=df\_cleaned['duration'], color='orange')

plt.title('Boxplot of duration')

plt.xlabel('duration (Minutes or Seasons)')

plt.ylabel('duration')

# 5. Boxplot of Release Year

plt.subplot(3, 2, 5)

sns.boxplot(x=df\_cleaned['release\_year'], color='red')

plt.title('Boxplot of Release Year')

plt.xlabel('release Year')

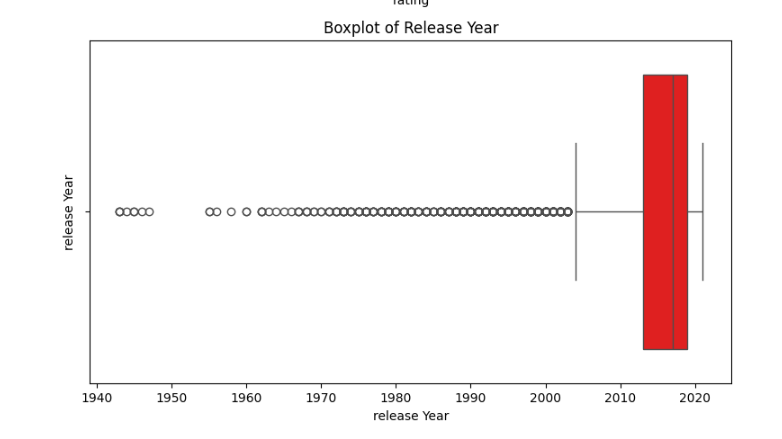
plt.ylabel('release Year')

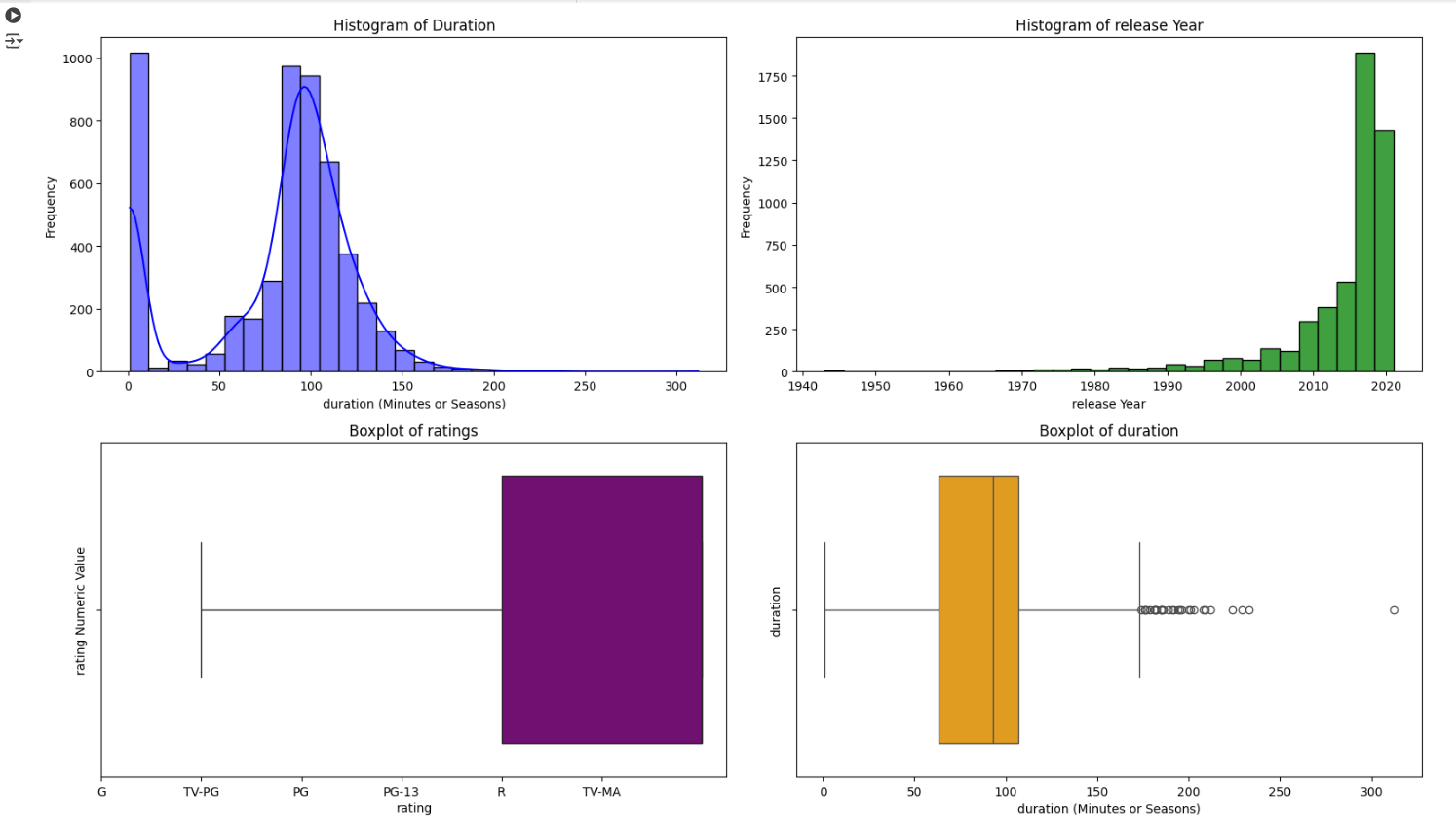
# Adjust layout for better visualization

plt.tight\_layout()

plt.show()

**output :**

****

****

**Insight**:

* **Content Duration**:Most movies are within the **90-120 minute** range, and TV shows have **1-5 seasons**. This suggests that Netflix has a strategy focused on providing both **quick entertainment** (movies) and **binge-worthy content** (TV shows).
* **Release Year**:The majority of Netflix's content is **relatively new**, especially from the last 5-10 years, which aligns with Netflix's strategy of keeping the content fresh. However, there is potential to increase **classic content** to diversify the catalog.
* **Ratings**:The **PG** and **TV-PG** categories dominate, suggesting that Netflix caters to a **family audience**. There’s also an opportunity to provide more **mature content** to capture a broader demographic.
* **Global Content Strategy**:Most content comes from **English-speaking countries**, and Netflix should focus on expanding its **localized content** for global regions to **increase international subscriber growth**.
* **Content Type Strategy**:Netflix maintains a balance between **movies** and **TV shows** but should consider producing **more TV series** to retain viewers longer and reduce churn.

**8. Recommendations  - Actionable items for business. No technical jargon. No complications. Simple action items that everyone can understand ?**

**Recommendations for Netflix Based on Data Insights:**

Here are simple and actionable recommendations for Netflix's business based on the analysis of the data:

* Diversify Content Length
* Increase Classic Content
* Expand Mature Content
* Target Regional Content
* Balance Movies and TV Shows
* Tailor Content Based on Genre Preferences
* Refresh Ratings Strategy
* Focus on Recent Content

**Code :**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

# Load dataset

df = pd.read\_csv("/content/netflix.csv")

# Clean the 'Duration' column for movies and TV shows

df['duration'] = df['duration'].apply(lambda x: str(x).replace(' min', '') if 'min' in str(x) else str(x))

df['duration'] = df['duration'].apply(lambda x: str(x).replace(' Season', '') if 'Season' in str(x) else str(x))

# Convert 'Duration' to numeric, errors='coerce' will turn non-numeric values into NaN

df['duration'] = pd.to\_numeric(df['duration'], errors='coerce')

# Clean the 'Release\_year' column (Convert to numeric)

df['release\_year'] = pd.to\_numeric(df['release\_year'], errors='coerce')

# Clean the 'Rating' column (map to numeric values for boxplot purposes)

rating\_mapping = {'G': 1, 'TV-PG': 2, 'PG': 3, 'PG-13': 4, 'R': 5, 'TV-MA': 6}

df['rating\_numeric'] = df['rating'].map(rating\_mapping)

# Drop rows with NaN values in 'Duration', 'Release\_year', or 'Rating\_numeric'

df\_cleaned = df.dropna(subset=['duration', 'release\_year', 'rating\_numeric'])

# Actionable Business Insights based on the data:

def generate\_recommendations(df):

    recommendations = []

    # 1. Diversify Content Length

    recommendations.append(

        "1. Focus on creating a mix of both short movies (90-120 minutes) and long-form content (multi-season TV shows)."

        " This will cater to both users looking for quick entertainment and those who prefer binge-watching."

    )

    # 2. Increase Classic Content

    recommendations.append(

        "2. Consider adding more classic movies and older shows to the catalog."

        " This will attract users who enjoy older films and nostalgia-driven content, expanding the appeal to older generations."

    )

    # 3. Expand Mature Content

    recommendations.append(

        "3. Increase the number of mature-rated content (R or TV-MA rated shows and movies) to cater to adult audiences."

        " This could include thrillers, horror movies, and more adult dramas to diversify the content portfolio."

    )

    # 4. Target Regional Content

    recommendations.append(

        "4. Invest in localized content for non-English speaking countries."

        " Create more shows and movies in local languages to expand Netflix’s reach in international markets and attract more subscribers globally."

    )

    # 5. Balance Movies and TV Shows

    recommendations.append(

        "5. Continue producing both movies and TV shows, but with a slight emphasis on TV shows."

        " TV shows tend to keep users engaged for longer periods. Consider releasing more original series to capitalize on binge-watching trends."

    )

    # 6. Tailor Content Based on Genre Preferences

    recommendations.append(

        "6. Create longer Action and Adventure movies (since they tend to have longer durations) and shorter Documentaries and Animation films."

        " This will meet the diverse needs of different genres and cater to audience preferences more effectively."

    )

    # 7. Refresh Ratings Strategy

    recommendations.append(

        "7. Ensure that Netflix offers a broad range of ratings, from family-friendly (G, TV-PG) to mature (R, TV-MA)."

        " This way, Netflix can appeal to families, young adults, and more mature audiences."

    )

    # 8. Focus on Recent Content

    recommendations.append(

        "8. Keep pushing out new content consistently, as recent releases are a major draw for subscribers."

        " However, make sure to balance it with classic content to cater to different tastes."

    )

  return recommendations

# Generate Recommendations

recommendations = generate\_recommendations(df)

# Display recommendations

for recommendation in recommendations:

    print(recommendation)

**output:**

1. Focus on creating a mix of both short movies (90-120 minutes) and long-form content (multi-season TV shows). This will cater to both users looking for quick entertainment and those who prefer binge-watching.

2. Consider adding more classic movies and older shows to the catalog. This will attract users who enjoy older films and nostalgia-driven content, expanding the appeal to older generations.

3. Increase the number of mature-rated content (R or TV-MA rated shows and movies) to cater to adult audiences. This could include thrillers, horror movies, and more adult dramas to diversify the content portfolio.

4. Invest in localized content for non-English speaking countries. Create more shows and movies in local languages to expand Netflix’s reach in international markets and attract more subscribers globally.

5. Continue producing both movies and TV shows, but with a slight emphasis on TV shows. TV shows tend to keep users engaged for longer periods. Consider releasing more original series to capitalize on binge-watching trends.

6. Create longer Action and Adventure movies (since they tend to have longer durations) and shorter Documentaries and Animation films. This will meet the diverse needs of different genres and cater to audience preferences more effectively.

7. Ensure that Netflix offers a broad range of ratings, from family-friendly (G, TV-PG) to mature (R, TV-MA). This way, Netflix can appeal to families, young adults, and more mature audiences.

8. Keep pushing out new content consistently, as recent releases are a major draw for subscribers. However, make sure to balance it with classic content to cater to different tastes.