netflix-business-case-study-dav

March 3, 2024

[1]:
```python
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import gdown
```

[2]:
```
!wget https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/000/940/
 ↪original/netflix.csv -O netflixdata.csv
```

```
--2024-03-03 12:14:09--  https://d2beiqkhq929f0.cloudfront.net/public_assets/ass
ets/000/000/940/original/netflix.csv
Resolving d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)…
108.157.172.176, 108.157.172.10, 108.157.172.183, …
Connecting to d2beiqkhq929f0.cloudfront.net
(d2beiqkhq929f0.cloudfront.net)|108.157.172.176|:443… connected.
HTTP request sent, awaiting response… 200 OK
Length: 3399671 (3.2M) [text/plain]
Saving to: 'netflixdata.csv'

netflixdata.csv     100%[===================>]   3.24M  --.-KB/s    in 0.1s

2024-03-03 12:14:10 (24.5 MB/s) - 'netflixdata.csv' saved [3399671/3399671]
```

[3]:
```python
netflix=pd.read_csv('netflixdata.csv')
```

[4]:
```python
netflix.head()
```

[4]:
```
  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…
```

[5]:
```
netflix.tail()
```

[5]:
```
     show_id     type        title         director  \
8802   s8803    Movie       Zodiac    David Fincher
8803   s8804  TV Show  Zombie Dumb              NaN
8804   s8805    Movie   Zombieland  Ruben Fleischer
8805   s8806    Movie         Zoom     Peter Hewitt
8806   s8807    Movie       Zubaan      Mozez Singh

                                                   cast        country  \
8802  Mark Ruffalo, Jake Gyllenhaal, Robert Downey J…  United States
8803                                                NaN            NaN
8804  Jesse Eisenberg, Woody Harrelson, Emma Stone, …  United States
8805  Tim Allen, Courteney Cox, Chevy Chase, Kate Ma…  United States
8806  Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanan…          India

             date_added  release_year rating   duration  \
8802  November 20, 2019          2007      R    158 min
8803       July 1, 2019          2018  TV-Y7  2 Seasons
8804   November 1, 2019          2009      R     88 min
8805   January 11, 2020          2006     PG     88 min
8806      March 2, 2019          2015  TV-14    111 min
```

```
                                             listed_in  \
8802                     Cult Movies, Dramas, Thrillers
8803             Kids' TV, Korean TV Shows, TV Comedies
8804                            Comedies, Horror Movies
8805                 Children & Family Movies, Comedies
8806   Dramas, International Movies, Music & Musicals

                                              description
8802   A political cartoonist, a crime reporter and a…
8803   While living alone in a spooky town, a young g…
8804   Looking to survive in a world taken over by zo…
8805   Dragged from civilian life, a former superhero…
8806   A scrappy but poor boy worms his way into a ty…
```

[6]:
```
# Checking total rows and columns
print(f"Count of rows: {netflix.shape[0]}  columns: {netflix.shape[1]}")
```

```
Count of rows: 8807  columns: 12
```

The dataset consists of 8,807 entries with 12 attributes:

- 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: The 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

###Concise Summary To get a concise summary of the dataset, we use the df.info() function. It provides information about the number of non-null values and the data types of each column. This summary helps identify missing values and potential issues with data types.

[7]:
```
#Information about the Netflix data given:
netflix.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
```

[8]:
```
#Describing the Netflix data columns:
netflix.describe()
```

[8]:
```
       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
```

[9]:
```
netflix.nunique()
```

[9]:
```
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
```

##Netflix's Global Reach

With its incredible expansion, Netflix has grown to become a major player in the streaming market. Here are some significant figures that highlight its worldwide influence:

- Netflix is one of the most popular media and video streaming platforms. They have over 8000 movies or tv shows available on their platform, as of mid-2021, they have over 200M

Subscribers globally.

- International Expansion: Netflix has effectively created a global footprint, being accessible in over 190 countries. The company has worked hard to ensure that its material is accessible to a wide range of viewers by providing dubbing and subtitles in multiple languages.

[10]:
```
#Missing Value Detection:
netflix.isnull().any()
```

[10]:
```
show_id         False
type            False
title           False
director         True
cast             True
country          True
date_added       True
release_year    False
rating           True
duration         True
listed_in       False
description     False
dtype: bool
```

[11]:
```
#Finding out the number of null values in each column:
netflix.T.apply(lambda x: x.isnull().sum(), axis = 1)
```

[11]:
```
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
```

[12]:
```
#Total number of null values in the data:
netflix.isnull().sum().sum()
```

[12]:
```
4307
```

0.1 Basic Analysis

1. Handling null values

a. For categorical variables with null values, update those rows as unknown_column_name. Example : Replace missing value with Unknown Actor for missing value in Actors column.

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

[13]:
```
netflix[netflix['duration'].isna()]
#3 Unknown duration values are found in duration column , and it is also found␣
↪that by mistake those data got entered in rating column
```

[13]:
```
     show_id   type                                     title    director  \
5541   s5542  Movie                           Louis C.K. 2017  Louis C.K.
5794   s5795  Movie                     Louis C.K.: Hilarious  Louis C.K.
5813   s5814  Movie  Louis C.K.: Live at the Comedy Store  Louis C.K.

            cast        country          date_added  release_year  rating  \
5541  Louis C.K.  United States       April 4, 2017          2017  74 min
5794  Louis C.K.  United States  September 16, 2016          2010  84 min
5813  Louis C.K.  United States     August 15, 2016          2015  66 min

     duration listed_in                                         description
5541      NaN    Movies  Louis C.K. muses on religion, eternal love, gi…
5794      NaN    Movies  Emmy-winning comedy writer Louis C.K. brings h…
5813      NaN    Movies  The comic puts his trademark hilarious/thought…
```

[14]:
```
temp = netflix[netflix['duration'].isna()].index
netflix.loc[temp] = netflix.loc[temp].fillna(method = 'ffill' , axis = 1)
```

[15]:
```
# replaced the wrong entries done in the rating column
netflix.loc[temp ,'rating'] = 'Unknown rating'
```

[16]:
```
netflix.loc[temp]
```

[16]:
```
     show_id   type                                     title    director  \
5541   s5542  Movie                           Louis C.K. 2017  Louis C.K.
5794   s5795  Movie                     Louis C.K.: Hilarious  Louis C.K.
5813   s5814  Movie  Louis C.K.: Live at the Comedy Store  Louis C.K.

            cast        country          date_added release_year  \
5541  Louis C.K.  United States       April 4, 2017         2017
5794  Louis C.K.  United States  September 16, 2016         2010
5813  Louis C.K.  United States     August 15, 2016         2015

               rating duration listed_in  \
5541  Unknown rating   74 min    Movies
5794  Unknown rating   84 min    Movies
5813  Unknown rating   66 min    Movies
```

```
                                            description
5541  Louis C.K. muses on religion, eternal love, gi…
5794  Emmy-winning comedy writer Louis C.K. brings h…
5813  The comic puts his trademark hilarious/thought…
```

[17]:
```
netflix.director.fillna('Unknown director', inplace=True)
netflix.cast.fillna('Unknown cast', inplace=True)
netflix.country.fillna('Unknown country', inplace=True)
netflix.rating.fillna('Unknown rating', inplace=True)
netflix.duration.fillna('Unknown duration', inplace=True)
```

[18]:
```
# Converting the 'date_added' column to datetime format
netflix["date_added"] = pd.to_datetime(netflix['date_added'])
```

[19]:
```
# Extracting month, month name, and year from the 'date_added' column
netflix['month_added'] = netflix['date_added'].dt.month
netflix['month_name_added'] = netflix['date_added'].dt.month_name()
netflix['year_added'] = netflix['date_added'].dt.year
```

###Descriptive Statistics It is essential to use descriptive statistics to comprehend the general features of the dataset. We can learn more about the count, mean, standard deviation, minimum, maximum, and quartiles, among other numerical qualities.

[20]:
```
netflix.describe()
```

[20]:
```
       month_added   year_added
count  8797.000000  8797.000000
mean      6.654996  2018.871888
std       3.436554     1.574243
min       1.000000  2008.000000
25%       4.000000  2018.000000
50%       7.000000  2019.000000
75%      10.000000  2020.000000
max      12.000000  2021.000000
```

2. Un-nesting the columns

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

   Unnesting below columns

- cast
- director
- country
- listed_in

[21]:
```
#Unnesting and exploring cast column:
netflix_cast = netflix[['show_id' , 'type' ,'cast']]
```

```
netflix_cast.dropna(inplace = True)
netflix_cast['cast'] = netflix_cast['cast'].apply(lambda x : x.split(','))
netflix_cast = netflix_cast.explode('cast')
netflix_cast['cast'] = netflix_cast['cast'].str.strip()
netflix_cast
```

```
<ipython-input-21-a8ae1e1e26b9>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  netflix_cast.dropna(inplace = True)
<ipython-input-21-a8ae1e1e26b9>:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  netflix_cast['cast'] = netflix_cast['cast'].apply(lambda x : x.split(','))
```

[21]:
```
     show_id     type                  cast
0         s1    Movie          Unknown cast
1         s2  TV Show            Ama Qamata
1         s2  TV Show           Khosi Ngema
1         s2  TV Show         Gail Mabalane
1         s2  TV Show        Thabang Molaba
…        …       …                     …
8806   s8807    Movie      Manish Chaudhary
8806   s8807    Movie          Meghna Malik
8806   s8807    Movie         Malkeet Rauni
8806   s8807    Movie        Anita Shabdish
8806   s8807    Movie  Chittaranjan Tripathy

[64951 rows x 3 columns]
```

[22]:
```
#Unnesting and exploring director column:
netflix_dir = netflix[['show_id' , 'type' ,'director']]
netflix_dir.dropna(inplace = True)
netflix_dir['director'] = netflix_dir['director'].apply(lambda x : x.split(','))
netflix_dir = netflix_dir.explode('director')
netflix_dir['director'] = netflix_dir['director'].str.strip()
netflix_dir
```

```
<ipython-input-22-0ac95d03ceb3>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

```
  netflix_dir.dropna(inplace = True)
<ipython-input-22-0ac95d03ceb3>:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  netflix_dir['director'] = netflix_dir['director'].apply(lambda x :
x.split(','))
```

[22]:
```
     show_id     type          director
0         s1    Movie   Kirsten Johnson
1         s2  TV Show  Unknown director
2         s3  TV Show   Julien Leclercq
3         s4  TV Show  Unknown director
4         s5  TV Show  Unknown director
…        …        …                 …
8802   s8803    Movie     David Fincher
8803   s8804  TV Show  Unknown director
8804   s8805    Movie   Ruben Fleischer
8805   s8806    Movie      Peter Hewitt
8806   s8807    Movie       Mozez Singh

[9612 rows x 3 columns]
```

[23]:
```python
#Unnesting and exploring country column:
netflix_country = netflix[['show_id' , 'type' ,'country']]
netflix_country.dropna(inplace = True)
netflix_country['country'] = netflix_country['country'].apply(lambda x : x.
 ↪split(','))
netflix_country = netflix_country.explode('country')
netflix_country['country'] = netflix_country['country'].str.strip()
netflix_country
```

```
<ipython-input-23-19f176bf2445>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  netflix_country.dropna(inplace = True)
<ipython-input-23-19f176bf2445>:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  netflix_country['country'] = netflix_country['country'].apply(lambda x :
x.split(','))
```

```
[23]:      show_id     type          country
      0         s1    Movie    United States
      1         s2  TV Show     South Africa
      2         s3  TV Show  Unknown country
      3         s4  TV Show  Unknown country
      4         s5  TV Show            India
      …        …        …               …
      8802   s8803    Movie    United States
      8803   s8804  TV Show  Unknown country
      8804   s8805    Movie    United States
      8805   s8806    Movie    United States
      8806   s8807    Movie            India

      [10850 rows x 3 columns]
```

```
[24]: #Unnesting and exploring listed_in column:
      netflix_list = netflix[['show_id' , 'type' ,'listed_in']]
      netflix_list.dropna(inplace = True)
      netflix_list['listed_in'] = netflix_list['listed_in'].apply(lambda x : x.
       ↪split(','))
      netflix_list = netflix_list.explode('listed_in')
      netflix_list['listed_in'] = netflix_list['listed_in'].str.strip()
      netflix_list
```

```
<ipython-input-24-3cbb550e74b5>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  netflix_list.dropna(inplace = True)
<ipython-input-24-3cbb550e74b5>:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  netflix_list['listed_in'] = netflix_list['listed_in'].apply(lambda x :
x.split(','))
```

```
[24]:      show_id     type                 listed_in
      0         s1    Movie             Documentaries
      1         s2  TV Show    International TV Shows
      1         s2  TV Show                 TV Dramas
      1         s2  TV Show              TV Mysteries
      2         s3  TV Show            Crime TV Shows
      …        …        …                       …
      8805   s8806    Movie  Children & Family Movies
```

```
8805    s8806     Movie                      Comedies
8806    s8807     Movie                        Dramas
8806    s8807     Movie      International Movies
8806    s8807     Movie            Music & Musicals

[19323 rows x 3 columns]
```

0.2 What does 'good' look like?

1. Find the counts of each categorical variable both using graphical and nongraphical analysis.
   a. For Non-graphical Analysis:

   Hint : We want you to find the values counts of each category for the given column

[25]:
```
# 2 types of content present in dataset - either Movie or TV Show
netflix['type'].unique()
```

[25]:
```
array(['Movie', 'TV Show'], dtype=object)
```

[26]:
```
netflix.describe()
```

[26]:
```
       month_added   year_added
count  8797.000000  8797.000000
mean      6.654996  2018.871888
std       3.436554     1.574243
min       1.000000  2008.000000
25%       4.000000  2018.000000
50%       7.000000  2019.000000
75%      10.000000  2020.000000
max      12.000000  2021.000000
```

[27]:
```
netflix[['listed_in','type', 'country', 'rating','director','duration']].
 ↪describe(include=['object'])
```

[27]:
```
                           listed_in   type        country rating  \
count                           8807   8807           8807   8807
unique                           514      2            749     15
top     Dramas, International Movies  Movie  United States  TV-MA
freq                             362   6131           2818   3207

                director  duration
count               8807      8807
unique              4529       220
top     Unknown director  1 Season
freq                2634      1793
```

[28]:
```
netflix.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
Data columns (total 15 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          8807 non-null   object
 4   cast              8807 non-null   object
 5   country           8807 non-null   object
 6   date_added        8797 non-null   datetime64[ns]
 7   release_year      8807 non-null   object
 8   rating            8807 non-null   object
 9   duration          8807 non-null   object
 10  listed_in         8807 non-null   object
 11  description       8807 non-null   object
 12  month_added       8797 non-null   float64
 13  month_name_added  8797 non-null   object
 14  year_added        8797 non-null   float64
dtypes: datetime64[ns](1), float64(2), object(12)
memory usage: 1.0+ MB
```

[29]:
```
#value counts for key categorical column data:
value_counts_type = netflix['type'].value_counts()
value_counts_country = netflix['country'].value_counts()
value_counts_rating = netflix['rating'].value_counts()
value_counts_director = netflix['director'].value_counts()
value_counts_duration = netflix['duration'].value_counts()
value_counts_release_year = netflix['release_year'].value_counts()


value_counts_type, value_counts_country, value_counts_rating,␣
 ↪value_counts_director, value_counts_duration, value_counts_release_year
```

[29]:
```
(Movie      6131
 TV Show    2676
 Name: type, dtype: int64,
 United States                                       2818
 India                                                972
 Unknown country                                      831
 United Kingdom                                       419
 Japan                                                245
                                                     …
 Romania, Bulgaria, Hungary                             1
 Uruguay, Guatemala                                     1
 France, Senegal, Belgium                               1
```

```
Mexico, United States, Spain, Colombia       1
United Arab Emirates, Jordan                 1
Name: country, Length: 749, dtype: int64,
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
Unknown rating       7
TV-Y7-FV             6
NC-17                3
UR                   3
Name: rating, dtype: int64,
Unknown director                 2634
Rajiv Chilaka                      19
Raúl Campos, Jan Suter             18
Suhas Kadav                        16
Marcus Raboy                       16
                                 …
Raymie Muzquiz, Stu Livingston      1
Joe Menendez                        1
Eric Bross                          1
Will Eisenberg                      1
Mozez Singh                         1
Name: director, Length: 4529, dtype: int64,
1 Season     1793
2 Seasons     425
3 Seasons     199
90 min        152
94 min        146
             …
16 min          1
186 min         1
193 min         1
189 min         1
191 min         1
Name: duration, Length: 220, dtype: int64,
2018    1147
2017    1032
2019    1030
2020     953
```

```
 2016     902
         …
 1959       1
 1925       1
 1961       1
 1947       1
 1966       1
 Name: release_year, Length: 74, dtype: int64)
```

##Observations - The average release_year being around 2014, along with a median of 2017, suggests that Netflix has a lot of content from the recent decade. - The high frequency of Movies compared to TV Shows indicates a stronger focus on movie content. - The United States appears to be the most common country for content production, followed by a wide range of other countries, indicating a diverse content catalog. - The rating "TV-MA" is the most frequent, suggesting a focus on mature audiences.

[30]:
```
merged_data = netflix.merge(netflix_dir, on="show_id", how="inner")
merged_data = merged_data.merge(netflix_cast, on="show_id", how="inner")
merged_data = merged_data.merge(netflix_country, on="show_id", how="inner")
merged_data = merged_data.merge(netflix_list, on="show_id", how="inner")
```

```
<ipython-input-30-8618985d9d96>:3: FutureWarning: Passing 'suffixes' which cause
duplicate columns {'type_x'} in the result is deprecated and will raise a
MergeError in a future version.
  merged_data = merged_data.merge(netflix_country, on="show_id", how="inner")
```

[31]:
```
merged_data.head()
```

[31]:
```
  show_id   type_x                 title        director_x  \
0      s1    Movie  Dick Johnson Is Dead  Kirsten Johnson
1      s2  TV Show         Blood & Water  Unknown director
2      s2  TV Show         Blood & Water  Unknown director
3      s2  TV Show         Blood & Water  Unknown director
4      s2  TV Show         Blood & Water  Unknown director

                                             cast_x      country_x  \
0                                      Unknown cast  United States
1  Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban…   South Africa
2  Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban…   South Africa
3  Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban…   South Africa
4  Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban…   South Africa

  date_added release_year rating   duration  … month_name_added year_added  \
0 2021-09-25         2020  PG-13     90 min  …        September     2021.0
1 2021-09-24         2021  TV-MA  2 Seasons  …        September     2021.0
2 2021-09-24         2021  TV-MA  2 Seasons  …        September     2021.0
3 2021-09-24         2021  TV-MA  2 Seasons  …        September     2021.0
4 2021-09-24         2021  TV-MA  2 Seasons  …        September     2021.0
```

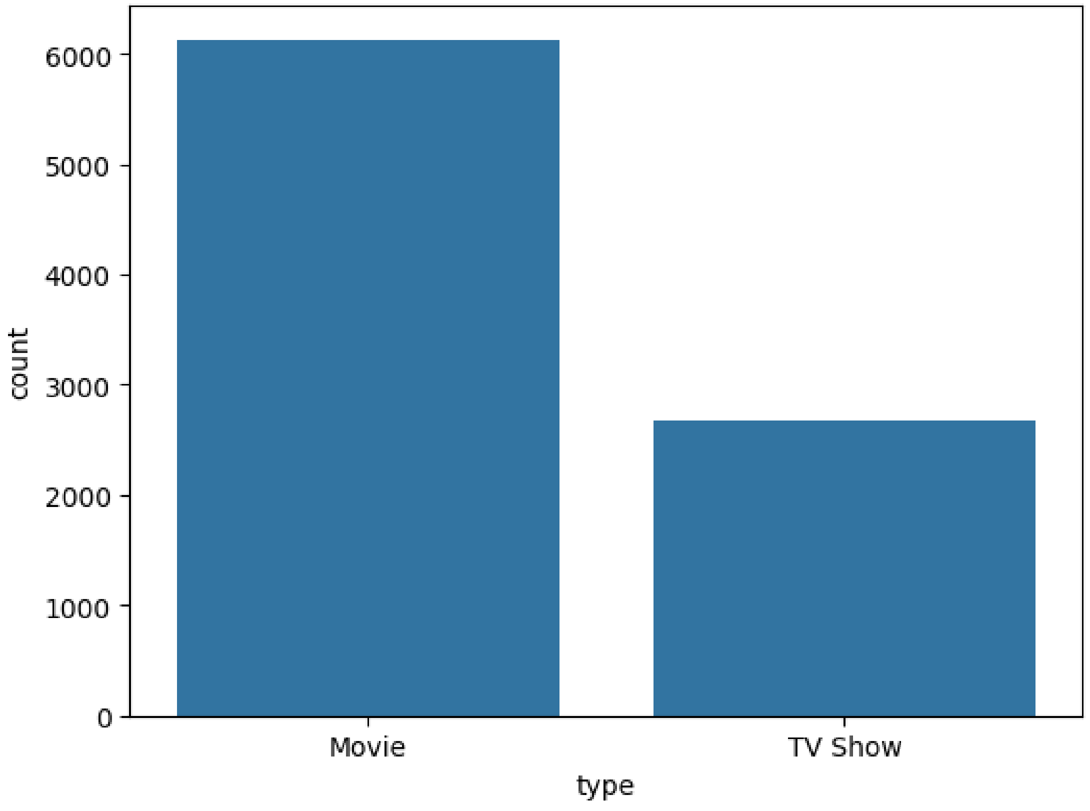
```
    type_y         director_y   type_x        cast_y   type_y       country_y  \
0    Movie    Kirsten Johnson    Movie  Unknown cast    Movie   United States
1  TV Show   Unknown director  TV Show    Ama Qamata  TV Show    South Africa
2  TV Show   Unknown director  TV Show    Ama Qamata  TV Show    South Africa
3  TV Show   Unknown director  TV Show    Ama Qamata  TV Show    South Africa
4  TV Show   Unknown director  TV Show   Khosi Ngema  TV Show    South Africa

      type              listed_in_y
0    Movie            Documentaries
1  TV Show  International TV Shows
2  TV Show                TV Dramas
3  TV Show              TV Mysteries
4  TV Show  International TV Shows

[5 rows x 23 columns]
```

### b. For graphical analysis:

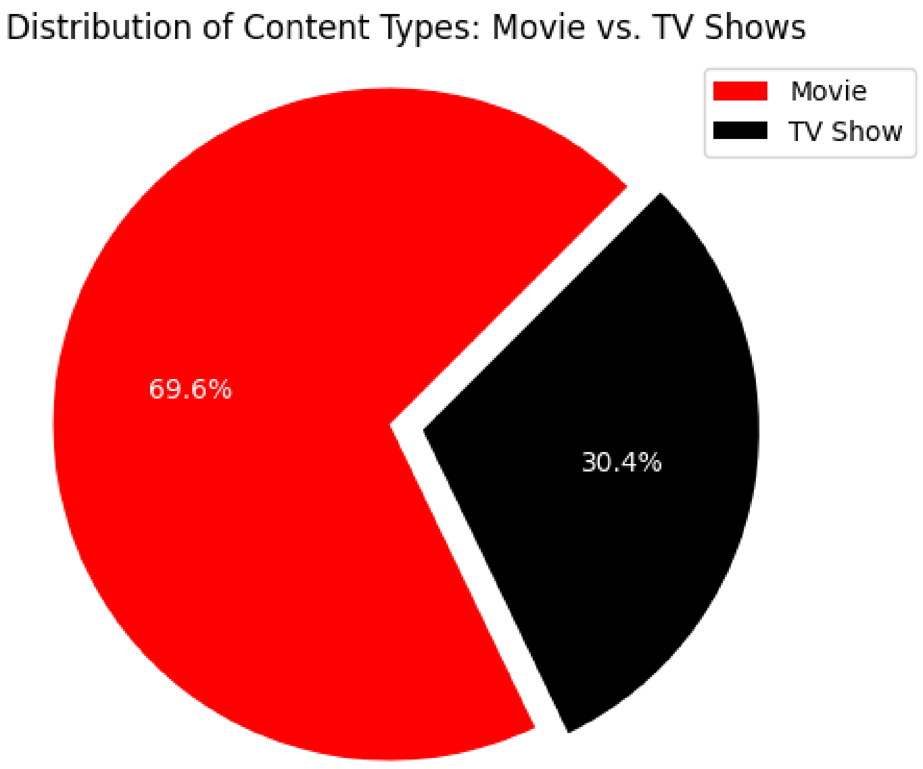
Hint : We can use a count plot to get the counts of each category

[32]:
```
sns.countplot(data=netflix,x='type')
```

[32]:
```
<Axes: xlabel='type', ylabel='count'>
```



[33]:
```
# Univariate Example with Pie Chart for 'Type' (Movie/TV Show)
type_counts = netflix['type'].value_counts()
labels=type_counts.index
sizes=type_counts.values

plt.figure(figsize=(7, 5))
plt.pie(sizes, labels=labels, autopct='%1.1f%%',colors = ['red' ,␣
 ↪'black'],textprops={'color':'white'},startangle=45,explode=(0.1,0))
plt.title('Distribution of Content Types: Movie vs. TV Shows')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.legend(loc='upper right')
plt.show()
```

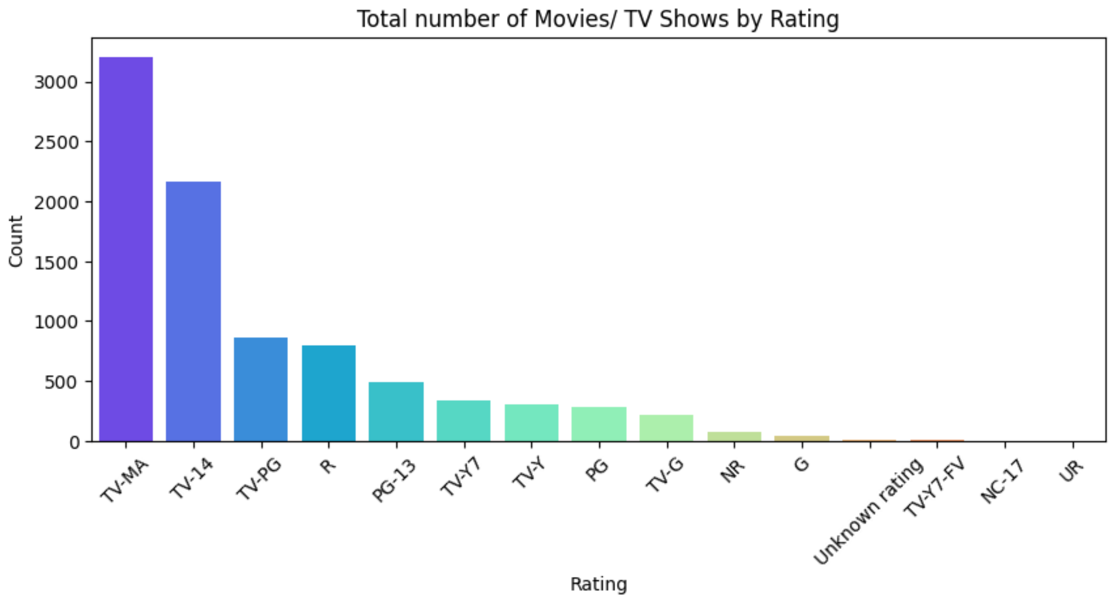


Analysis: The pie chart visualization shows that 69.6% of the content on Netflix consists of film, while the remaining 30.4% are TV shows.

[75]:
```
# Countplot for Rating
plt.figure(figsize=(10, 4))
```

```
sns.countplot(x='rating', data=netflix, order=netflix['rating'].value_counts().
  ↪index, palette='rainbow')
plt.title('Total number of Movies/ TV Shows by Rating')
plt.xlabel('Rating')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```

```
<ipython-input-75-4e6e19cd62f7>:3: FutureWarning:

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

  sns.countplot(x='rating', data=netflix,
order=netflix['rating'].value_counts().index, palette='rainbow')
```

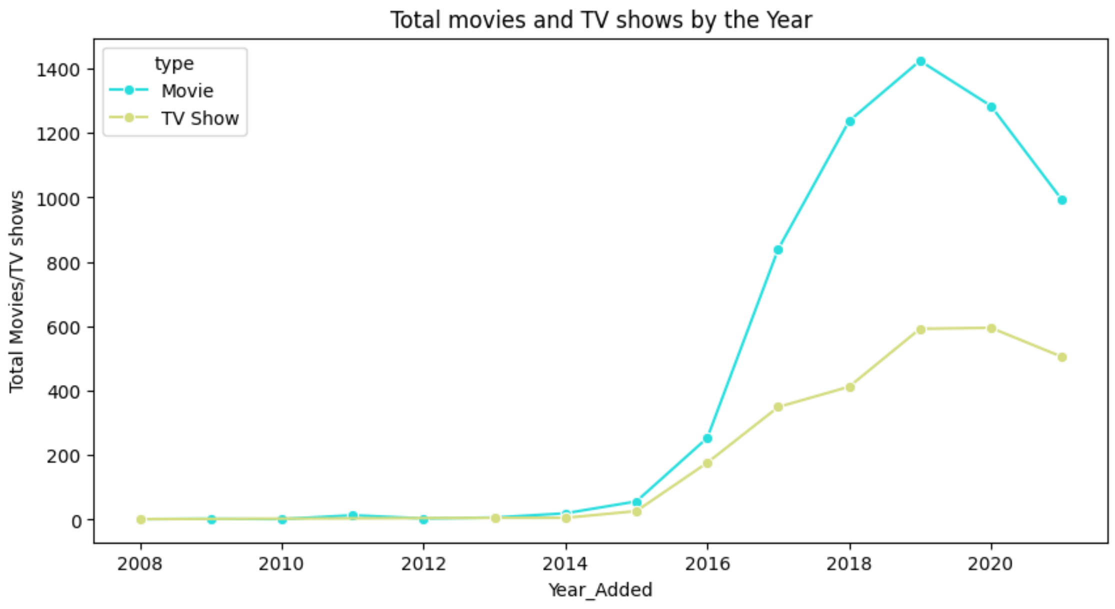


Analysis: Most fo the TV shows and Movies are rated TV-MA, which means the content is appropriate for Mature Audiences. This is followed by rating where parents are strongly cautioned that they need to be cautious to not show the content for age under 14.

How has the number of movies/TV shows added on Netflix per year changed over the time?

```
[35]: date_df = netflix.groupby(['year_added' ,'type' ])['show_id'].count().
        ↪reset_index()
      date_df.rename({'show_id' : 'total movies/TV shows'}, axis = 1 , inplace = True)
```

[36]:
```
plt.figure(figsize = (10,5))
sns.lineplot(data = date_df , x = 'year_added' , y = 'total movies/TV shows' ,␣
 ↪hue = 'type', marker = 'o'  , ms = 6,palette = 'rainbow')
plt.xlabel('Year_Added' , fontsize = 10)
plt.ylabel('Total Movies/TV shows' , fontsize = 10)
plt.title('Total movies and TV shows by the Year' , fontsize = 12)
plt.show()
```



Analysis: The line chart illustrates the number of movies and TV shows added to Netflix over time. It visually represents the growth and trends in content additions, with separate lines for films and TV shows.

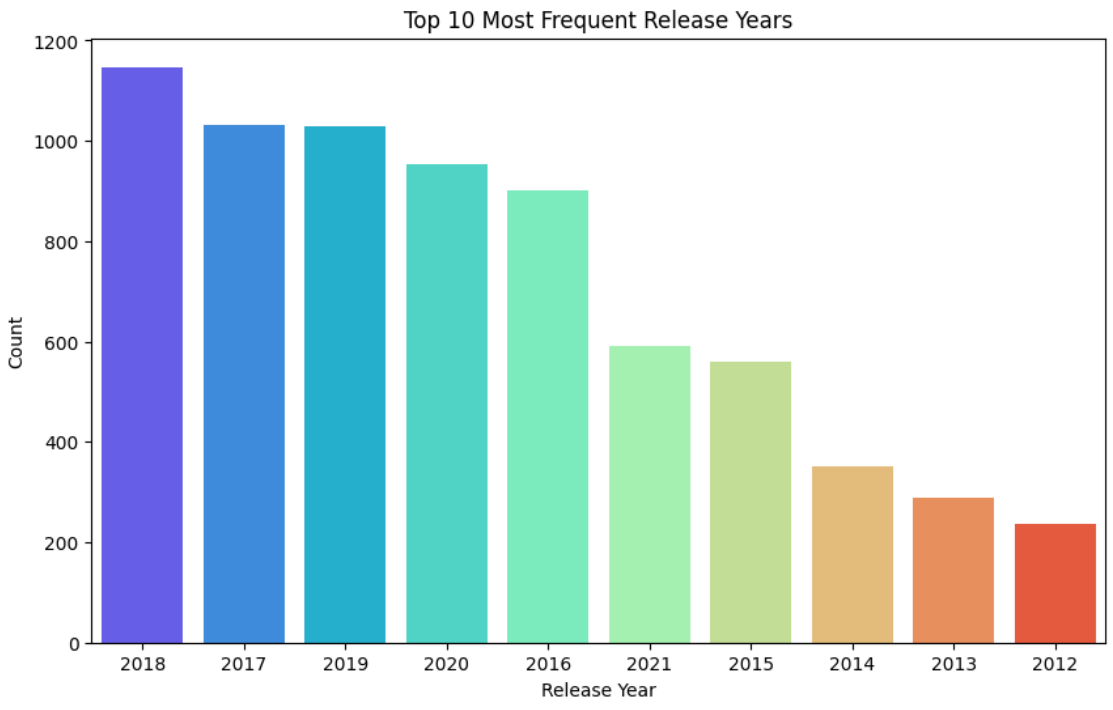
Netflix saw its real growth starting from the year 2015, & we can see it added more Movies than TV Shows over the years.

Also, it is interesting that the content addition dropped in 2020. This could be due to the pandemic situation, and there were very less movies or TV shows being produced or continued production. However, it might increase in future since the world has adjusted to cautious culture.

##Top 10 Release years

[109]:
```
# Countplot for top 10 release years
plt.figure(figsize=(10, 6))
sns.countplot(data=netflix, x='release_year', order=netflix['release_year'].
 ↪value_counts().iloc[:10].index,palette='rainbow')
plt.title('Top 10 Most Frequent Release Years')
plt.xlabel('Release Year')
plt.ylabel('Count')
plt.show()
```

```
<ipython-input-109-b71d9b3129c2>:3: FutureWarning:

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

  sns.countplot(data=netflix, x='release_year',
order=netflix['release_year'].value_counts().iloc[:10].index,palette='rainbow')
```



Analysis: This bar chart shows that the content on Netflix is increasing year by year, since the movie/ TV show production is also increasing on OTT platforms year by year.

Recently, people have started buying subscriptions for OTT platforms such as Netflix to view their favorite shows and movies. More the releases, more the content on OTTs, Netflix being one of them.

The top 10 most frequent release years are all from the recent past, with the year 2018 having the most content.

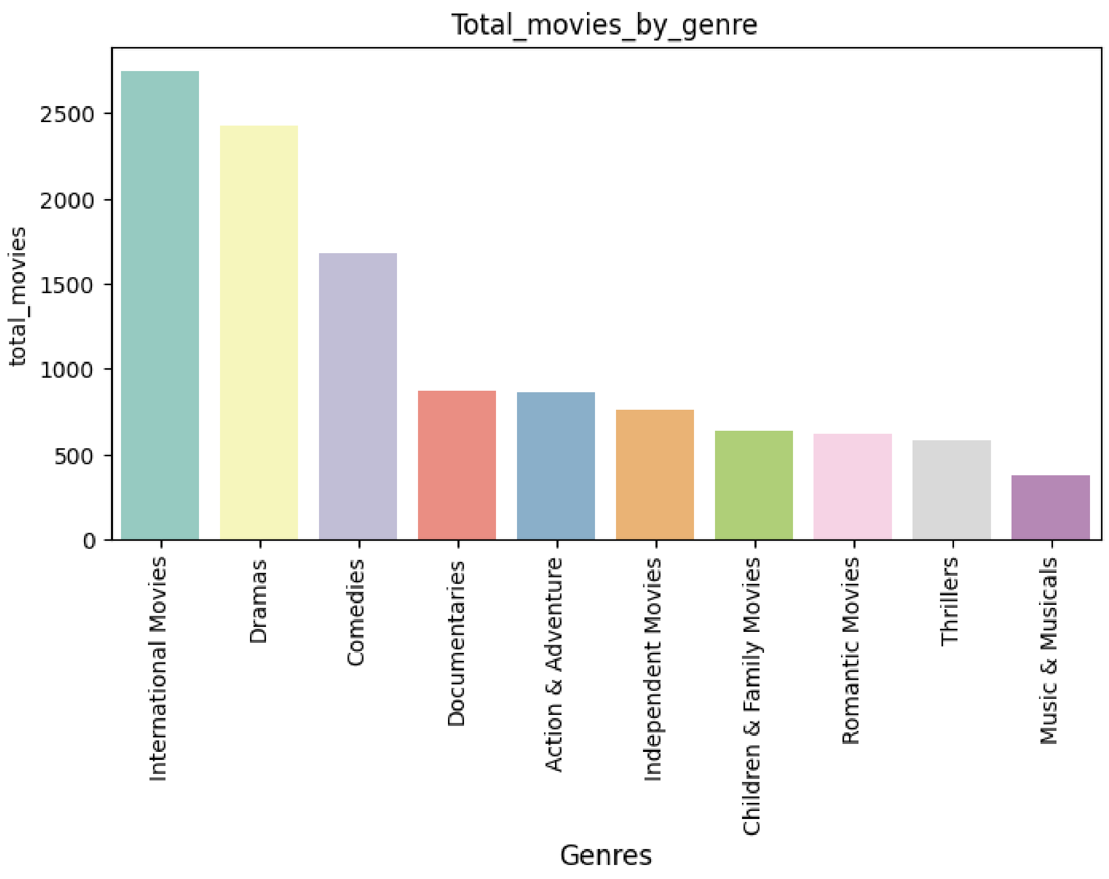
Total movies/ TV shows by genre

[37]:
```python
top_10_movie_genres = netflix_list[netflix_list['type'] == 'Movie'].listed_in.
 ↪value_counts().head(10).index
df_movie = netflix_list.loc[netflix_list['listed_in'].isin(top_10_movie_genres)]
```

```
top_10_TV_genres = netflix_list[netflix_list['type'] == 'TV Show'].listed_in.
 ↪value_counts().head(10).index
df_tv = netflix_list.loc[netflix_list['listed_in'].isin(top_10_TV_genres)]
```

[38]:
```
plt.figure(figsize= (8,4))
sns.countplot(data = df_movie , x = 'listed_in' , order =␣
 ↪top_10_movie_genres,palette='Set3')
plt.xticks(rotation = 90 , fontsize = 10)
plt.ylabel('total_movies' , fontsize = 10)
plt.xlabel('Genres' , fontsize = 12)
plt.title('Total_movies_by_genre')
plt.show()
```

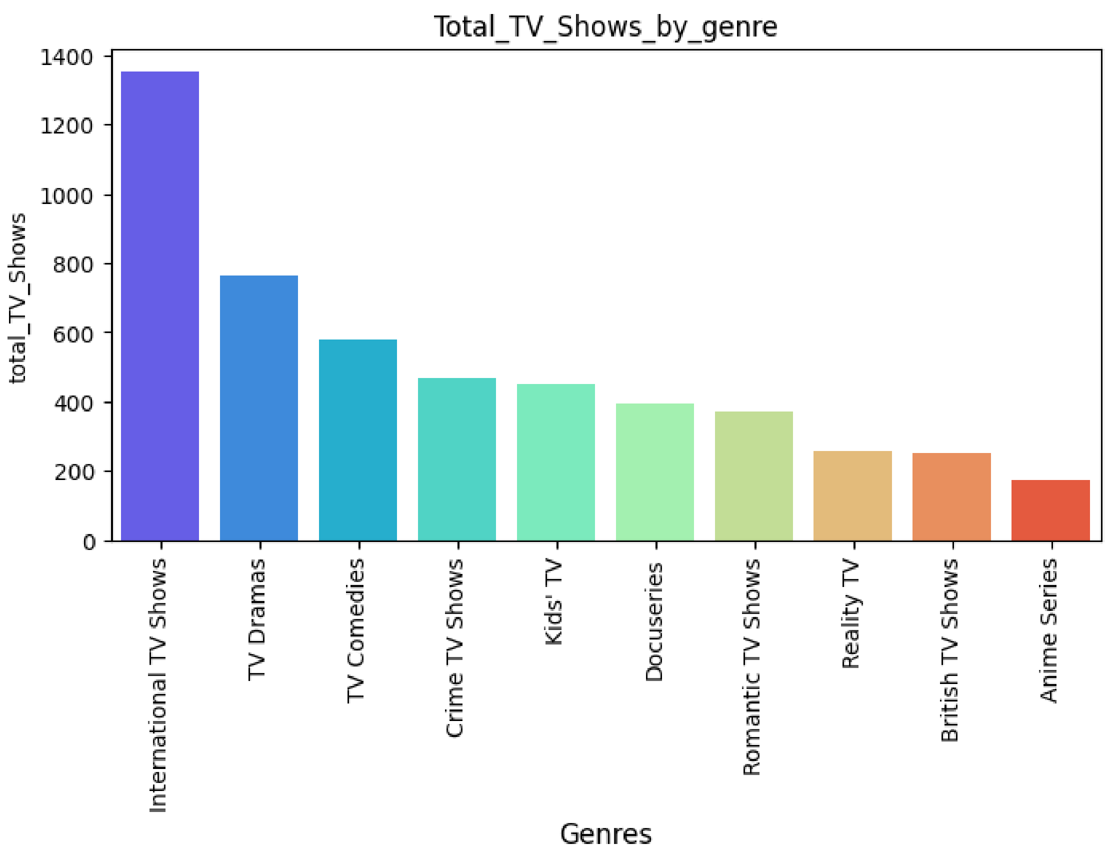
```
<ipython-input-38-b709973ec7fa>:2: FutureWarning:

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

  sns.countplot(data = df_movie , x = 'listed_in' , order =
top_10_movie_genres,palette='Set3')
```



Analysis: The above bar plot shows that the top genre is International movies, which shows that Netflix is adding the content from all over the globe, also showing the diversification of users, content on Netflix platform. Second beong the Drama genre shows that users prefer to watch dramas and comedies over serious content.

[39]:
```
plt.figure(figsize= (8,4))
sns.countplot(data = df_tv , x = 'listed_in' , order =␣
 ↪top_10_TV_genres,palette='rainbow')
plt.xticks(rotation = 90 , fontsize = 10)
plt.ylabel('total_TV_Shows' , fontsize = 10)
plt.xlabel('Genres' , fontsize = 12)
plt.title('Total_TV_Shows_by_genre')
plt.show()
```

```
<ipython-input-39-0b87ae3ba895>:2: FutureWarning:

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

  sns.countplot(data = df_tv , x = 'listed_in' , order =
top_10_TV_genres,palette='rainbow')
```



```
[71]: netflix.release_year.min() , netflix.release_year.max()
```

```
[71]: (1925, 2021)
```

Total movies/TV shows distribution by duration of the content

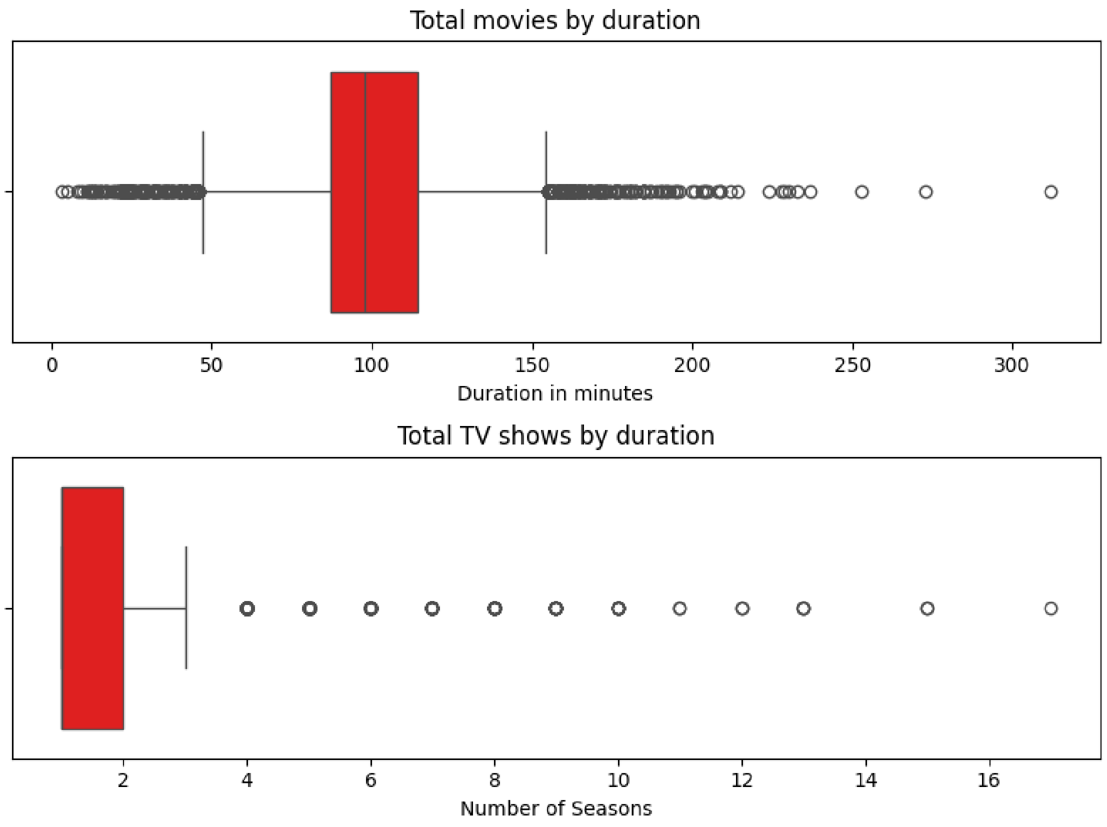
```
[72]: movies  = netflix.loc[netflix['type'] == 'Movie']
      tv_shows = netflix.loc[netflix['type'] == 'TV Show']

      movies['duration'] = movies['duration'].str[:-3]
      movies['duration'] = movies['duration'].astype('float')

      tv_shows['duration'] = tv_shows.duration.str[:-7].apply(lambda x : x.strip())
      tv_shows['duration'] = tv_shows['duration'].astype('float')

      movies.rename({'duration': 'duration_in_minutes'} ,axis = 1 , inplace = True)
      tv_shows.rename({'duration': 'duration_in_seasons'} ,axis = 1 , inplace = True)
```

```
<ipython-input-72-68da1b3863cb>:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  movies['duration'] = movies['duration'].str[:-3]
<ipython-input-72-68da1b3863cb>:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  movies['duration'] = movies['duration'].astype('float')
<ipython-input-72-68da1b3863cb>:7: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  tv_shows['duration'] = tv_shows.duration.str[:-7].apply(lambda x : x.strip())
<ipython-input-72-68da1b3863cb>:8: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  tv_shows['duration'] = tv_shows['duration'].astype('float')
<ipython-input-72-68da1b3863cb>:10: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  movies.rename({'duration': 'duration_in_minutes'} ,axis = 1 , inplace = True)
<ipython-input-72-68da1b3863cb>:11: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  tv_shows.rename({'duration': 'duration_in_seasons'} ,axis = 1 , inplace =
True)
```

[74]:
```
fig, ax = plt.subplots(2,1, figsize=(8,6))

sns.boxplot (data = movies , x = 'duration_in_minutes' ,ax =ax[0],color = 'red')
ax[0].set_xlabel('Duration in minutes' ,  fontsize = 10)
ax[0].set_title('Total movies by duration')

sns.boxplot (data = tv_shows , x = 'duration_in_seasons' , ax = ax[1],color =␣
 ↪'red')
```

```
ax[1].set_xlabel('Number of Seasons' ,  fontsize = 10)
ax[1].set_title('Total TV shows by duration')

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

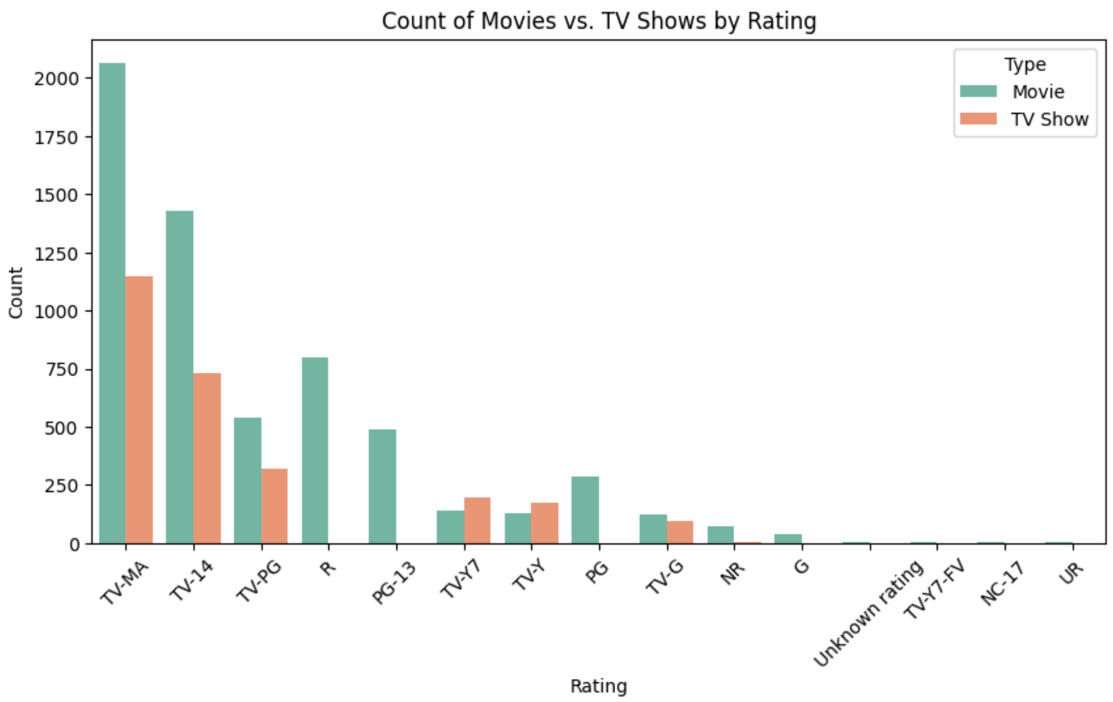


###Analysis Movie Duration: 50 mins - 150 mins is the general range excluding potential outliers (values lying outside the whiskers of boxplot)

TV Show Duration: 1-3 seasons is the general range for TV shows excluding potential outliers

###Bivariate Analysis Relationship Between Type and Rating

[189]:
```
# Countplot for Type vs Rating
plt.figure(figsize=(10, 5))
sns.countplot(x='rating', hue='type', data=netflix, order=netflix['rating'].
  ↪value_counts().index, palette='Set2')
plt.title('Count of Movies vs. TV Shows by Rating')
plt.xlabel('Rating')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.legend(title='Type')
plt.show()
```



Observation:

-Both Movies and TV Shows predominantly fall under the “TV-MA” and “TV-14” ratings.

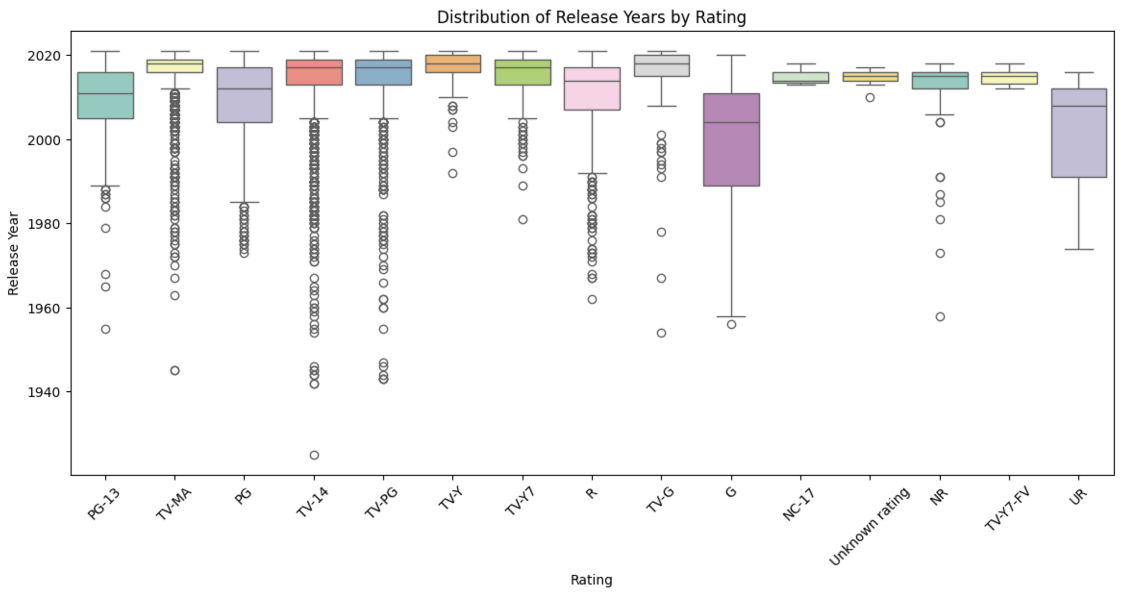
-The distribution of ratings between Movies and TV Shows is somewhat similar, though Movies have a higher count in most rating categories.

####Relationship Between Rating and Release Year

[191]:
```
# Boxplot for rating vs. release_year
plt.figure(figsize=(14, 6))
sns.boxplot(x='rating', y='release_year', data=netflix, palette='Set3')
plt.title('Distribution of Release Years by Rating')
plt.xlabel('Rating')
plt.ylabel('Release Year')
plt.xticks(rotation=45)
plt.show()
```

```
<ipython-input-191-123fd0fba695>:3: FutureWarning:

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

```
  sns.boxplot(x='rating', y='release_year', data=netflix, palette='Set3')
```



Observations:

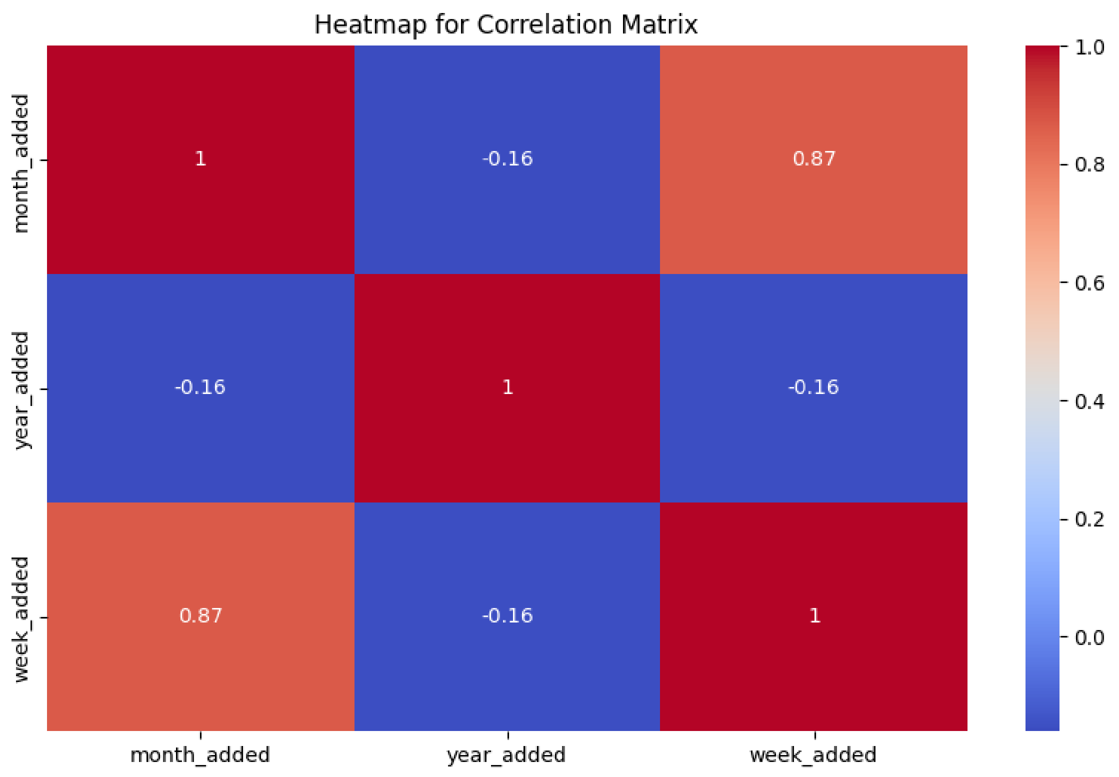
-The boxplot shows that the median release year for most ratings is relatively recent.

-Content with ratings "TV-Y" and "TV-Y7" tends to be older compared to other ratings.

###Correlation Analysis: Heatmaps Heatmap for Correlation Matrix

[193]:
```python
# Heatmap for correlation matrix
correlation_matrix = netflix.corr()

plt.figure(figsize=(10, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title('Heatmap for Correlation Matrix')
plt.show()
```

```
<ipython-input-193-1c00dd88e80a>:2: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it will
default to False. Select only valid columns or specify the value of numeric_only
to silence this warning.
  correlation_matrix = netflix.corr()
```

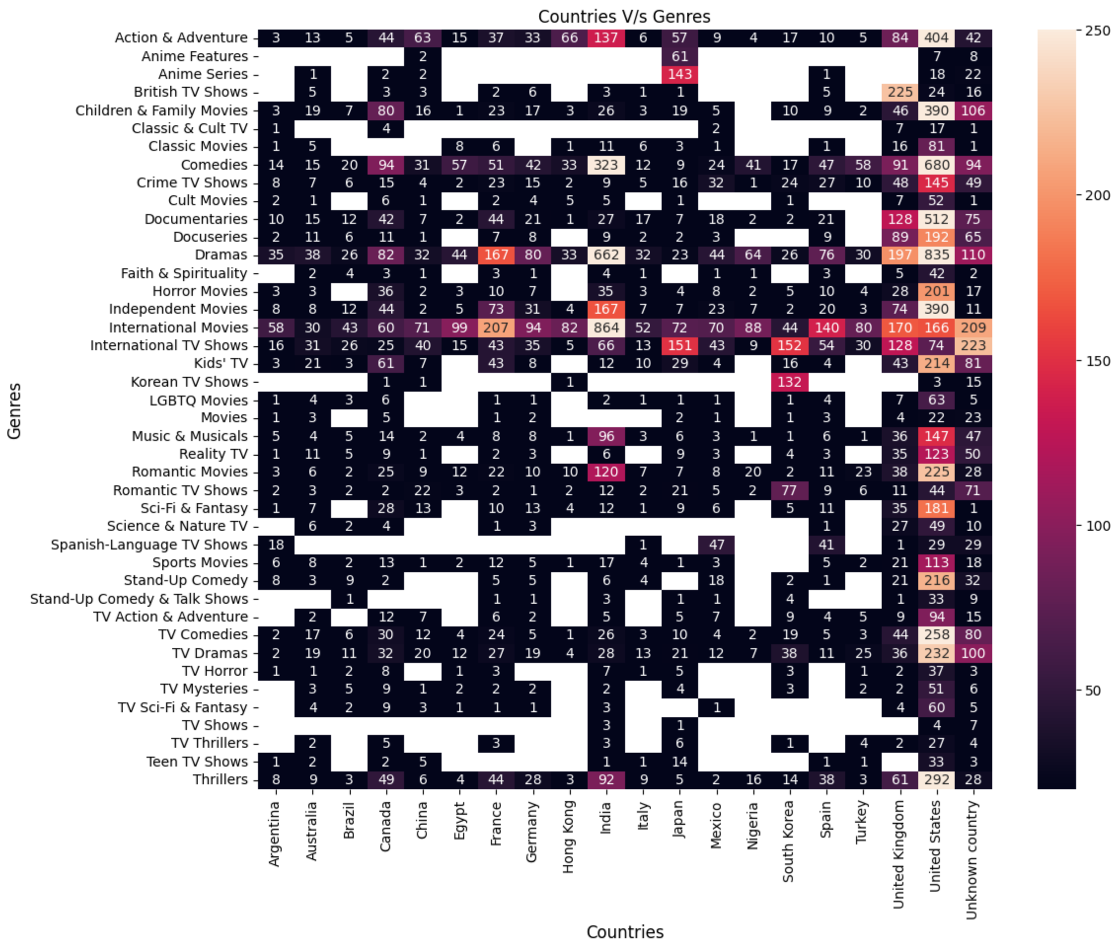


Lets now check popular genres in top 20 countries

```
[198]: top_20_country = netflix_country.country.value_counts().head(20).index
       top_20_country = netflix_country.loc[netflix_country['country'].
        ↪isin(top_20_country)]
       random_x = top_20_country.merge(netflix_list , on = 'show_id').drop_duplicates()
       country_genre = random_x.groupby([ 'country' , 'listed_in'])['show_id'].count().
        ↪sort_values(ascending = False).reset_index()
       country_genre = country_genre.pivot(index = 'listed_in' , columns = 'country' ,␣
        ↪values = 'show_id')
```

```
[199]: plt.figure(figsize = (12,10))
       sns.heatmap(data = country_genre , annot = True , fmt=".0f" , vmin = 20 , vmax␣
        ↪= 250 )
       plt.xlabel('Countries' , fontsize = 12)
       plt.ylabel('Genres' , fontsize = 12)
       plt.title('Countries V/s Genres' , fontsize = 12)
```

```
[199]: Text(0.5, 1.0, 'Countries V/s Genres')
```



Analysis: Popular genres across countries: Action & Adventure, Children & Family Movies, Comedies, Dramas, International Movies & TV Shows, TV Dramas, Thrillers

Country-specific genres: Korean TV shows (Korea), British TV Shows (UK), Anime features and Anime series (Japan), Spanish TV Shows (Argentina, Mexico and Spain)

United States and United Kingdom have a good mix of almost all genres.

Maximum International movies are produced in India.

2. Comparison of tv shows vs. movies.

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

   Hint : We want you to apply group by each country and find the count of unique titles of movies

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

   Hint : We want you to apply group by each country and find the count of unique titles of Tv-shows

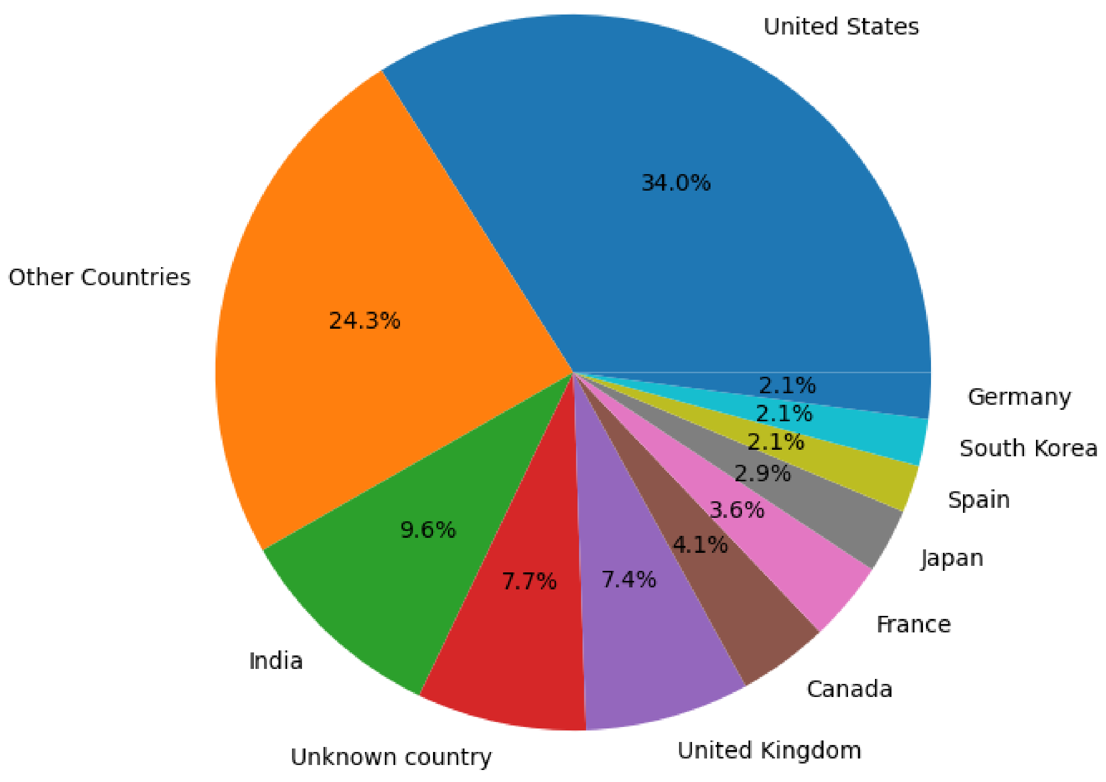
```
[201]: netflix.country.value_counts()
```

```
[201]: United States                             2818
       India                                      972
       Unknown country                            831
       United Kingdom                             419
       Japan                                      245
                                                 …
       Romania, Bulgaria, Hungary                   1
       Uruguay, Guatemala                           1
       France, Senegal, Belgium                     1
       Mexico, United States, Spain, Colombia       1
       United Arab Emirates, Jordan                 1
       Name: country, Length: 749, dtype: int64
```

```
[184]: # Countplot for country
       df = netflix_country.country.value_counts().head(10).index
       top_10 = netflix_country.loc[netflix_country['country'].isin(df)]
       netflix_country['cat'] = netflix_country['country'].apply(lambda x : x if x in␣
        ↪df else 'Other Countries' )
```

```
[42]: x = netflix_country.cat.value_counts()

      plt.figure(figsize = (7,7))
      plt.pie(x , labels = x.index, autopct='%1.1f%%')
      plt.title('Total Content produced in each country' , fontsize = 12)
      plt.show()
```

Total Content produced in each country



Analysis: The pie chart visualization reveals that the United States is the top country where Netflix is popular.

[43]:
```
netflix.country[netflix['type']=='Movie'].value_counts()
```

[43]:
```
United States                         2058
India                                  893
Unknown country                        440
United Kingdom                         206
Canada                                 122
                                      …
United Kingdom, Russia, United States    1
Paraguay, Argentina                      1
United Kingdom, Malawi                   1
Austria, Iraq, United States             1
United Arab Emirates, Jordan             1
Name: country, Length: 652, dtype: int64
```

```
[44]: netflix.country[netflix['type']=='TV Show'].value_counts()
```

```
[44]: United States                                  760
      Unknown country                                391
      United Kingdom                                 213
      Japan                                          169
      South Korea                                    158
                                                    …
      Belarus                                          1
      United Kingdom, Australia                        1
      France, Australia, Germany                       1
      Australia, New Zealand, United States            1
      United States, France, South Korea, Indonesia    1
      Name: country, Length: 197, dtype: int64
```

```
[45]: y = netflix.groupby(['country' , 'type'])['show_id'].count().reset_index()
      y.pivot(index = 'country' , columns = 'type' , values = 'show_id').
       ↪sort_values('Movie',ascending = False)
```

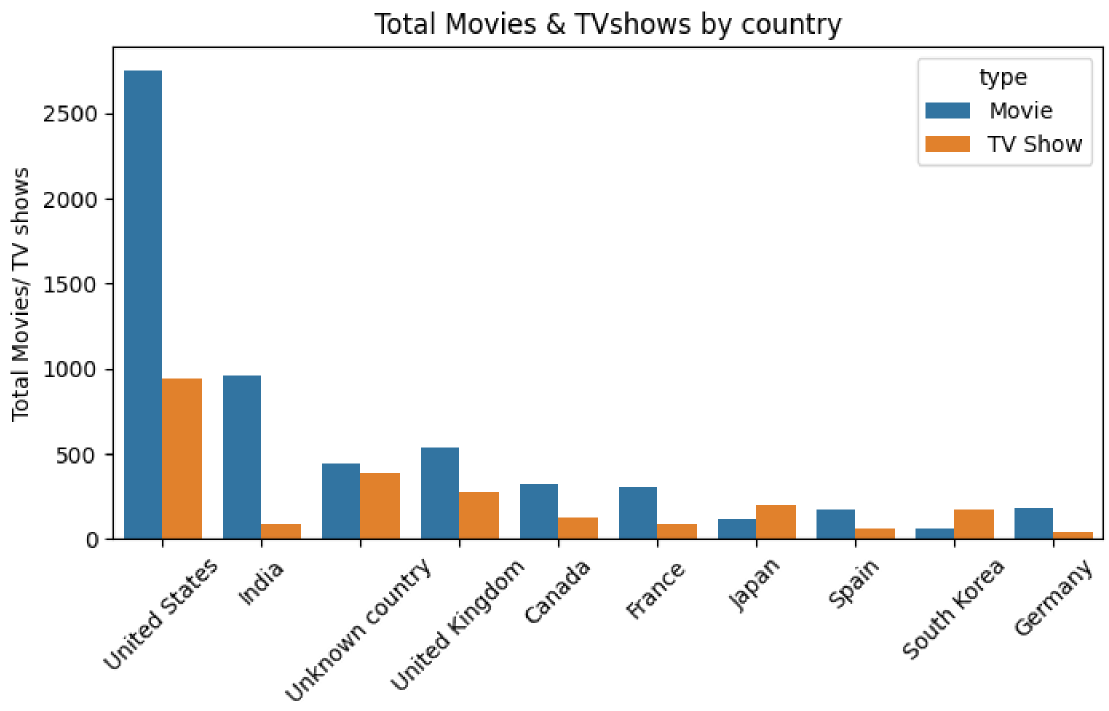
```
[45]: type                                Movie  TV Show
      country
      United States                      2058.0    760.0
      India                               893.0     79.0
      Unknown country                     440.0    391.0
      United Kingdom                      206.0    213.0
      Canada                              122.0     59.0
      …                                     …      …
      United States, New Zealand, Japan     NaN      1.0
      United States, Poland                 NaN      1.0
      United States, Singapore              NaN      1.0
      United States, South Korea, China     NaN      2.0
      Uruguay, Germany                      NaN      1.0

      [749 rows x 2 columns]
```

```
[188]: x = top_10.groupby(['country' , 'type'])['show_id'].count().reset_index()
       x.pivot(index = 'country' , columns = 'type' , values = 'show_id').
        ↪sort_values('Movie',ascending = False)
```

```
[188]: type              Movie  TV Show
       country
       United States      2752      938
       India               962       84
       United Kingdom      534      272
       Unknown country     440      391
       Canada              319      126
       France              303       90
```

```
Germany           182      44
Spain             171      61
Japan             119     199
South Korea        61     170
```

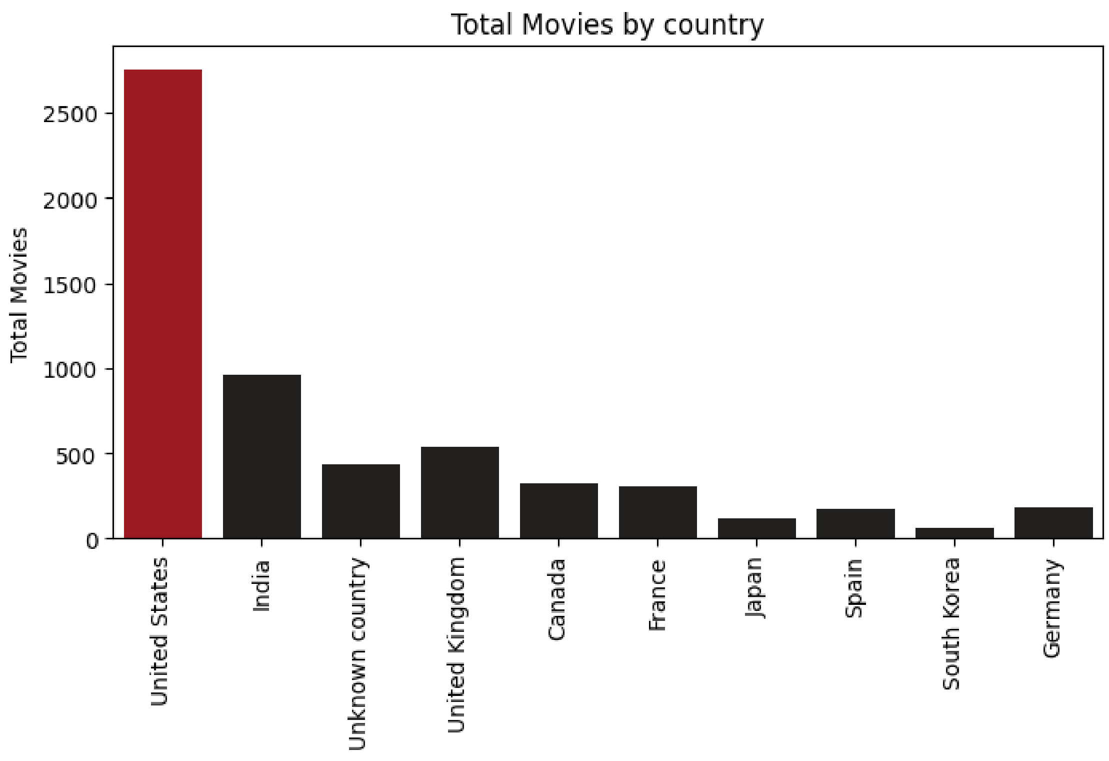
[187]:
```
plt.figure(figsize= (8,4))
sns.countplot(data = top_10 , x = 'country' , order = df , hue = 'type')
plt.xticks(rotation = 45 , fontsize = 10)
plt.ylabel('Total Movies/ TV shows' , fontsize = 10)
plt.xlabel('')
plt.title('Total Movies & TVshows by country')
plt.show()
```



Analysis: The bar chart visualization reveals that the United States is the top country where Netflix is popular.

[48]:
```
plt.figure(figsize= (8,4))
colors = ['#b20710'] + ['#221f1f'] * (len(top_10) - 1)
cp=sns.countplot(data = top_10[top_10['type']=='Movie'] , x = 'country' , order␣
 ↪= df,palette=colors)
plt.xticks(rotation = 90 , fontsize = 10)
plt.ylabel('Total Movies' , fontsize = 10)
plt.xlabel('')
plt.title('Total Movies by country')
plt.show()
```

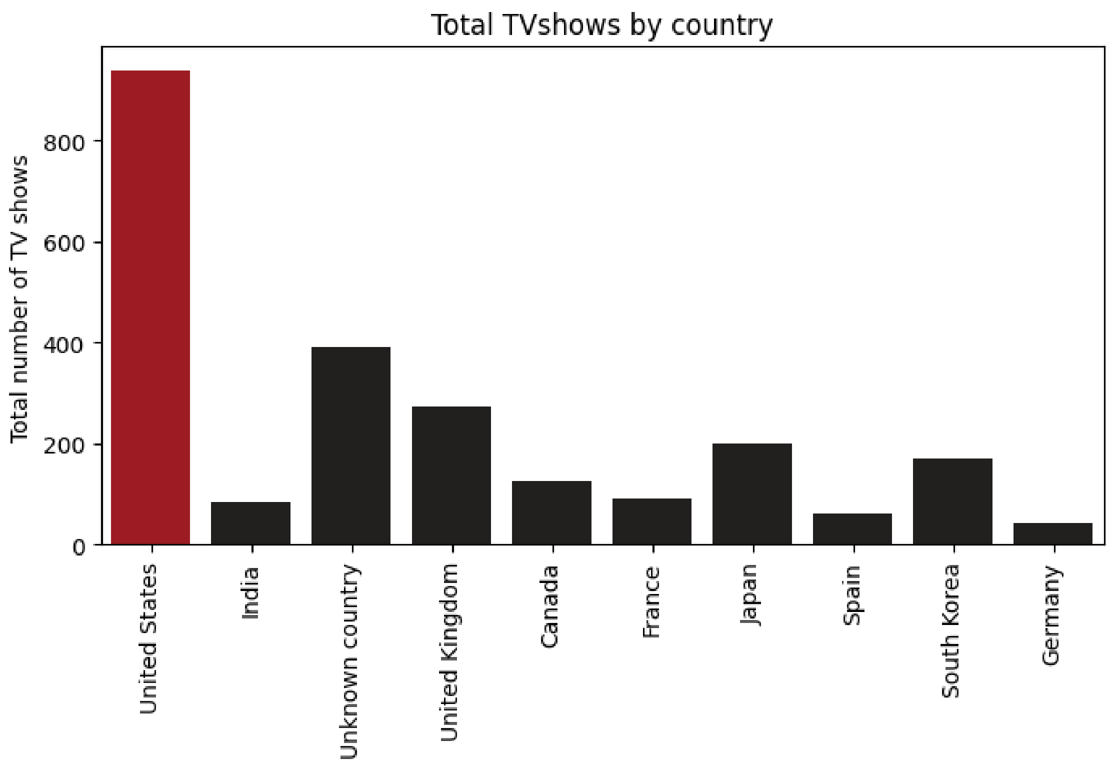
```
<ipython-input-48-83251209fa39>:3: FutureWarning:

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

  cp=sns.countplot(data = top_10[top_10['type']=='Movie'] , x = 'country' ,
order = df,palette=colors)
<ipython-input-48-83251209fa39>:3: UserWarning: The palette list has more values
(8218) than needed (10), which may not be intended.
  cp=sns.countplot(data = top_10[top_10['type']=='Movie'] , x = 'country' ,
order = df,palette=colors)
```



Analysis : US as a country is the highest and top country where Netflix is prevalent.

```
[49]: plt.figure(figsize= (8,4))
colors = ['#b20710'] + ['#221f1f'] * (len(top_10) - 1)
cp=sns.countplot(data = top_10[top_10['type']=='TV Show'] , x = 'country' ,␣
  ↪order = df,palette=colors)
plt.xticks(rotation = 90 , fontsize = 10)
plt.ylabel('Total number of TV shows' , fontsize = 10)
plt.xlabel('')

plt.title('Total TVshows by country')
plt.show()
```

```
<ipython-input-49-5ac4225f67ca>:3: FutureWarning:

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

  cp=sns.countplot(data = top_10[top_10['type']=='TV Show'] , x = 'country' ,
order = df,palette=colors)
<ipython-input-49-5ac4225f67ca>:3: UserWarning: The palette list has more values
(8218) than needed (10), which may not be intended.
  cp=sns.countplot(data = top_10[top_10['type']=='TV Show'] , x = 'country' ,
order = df,palette=colors)
```

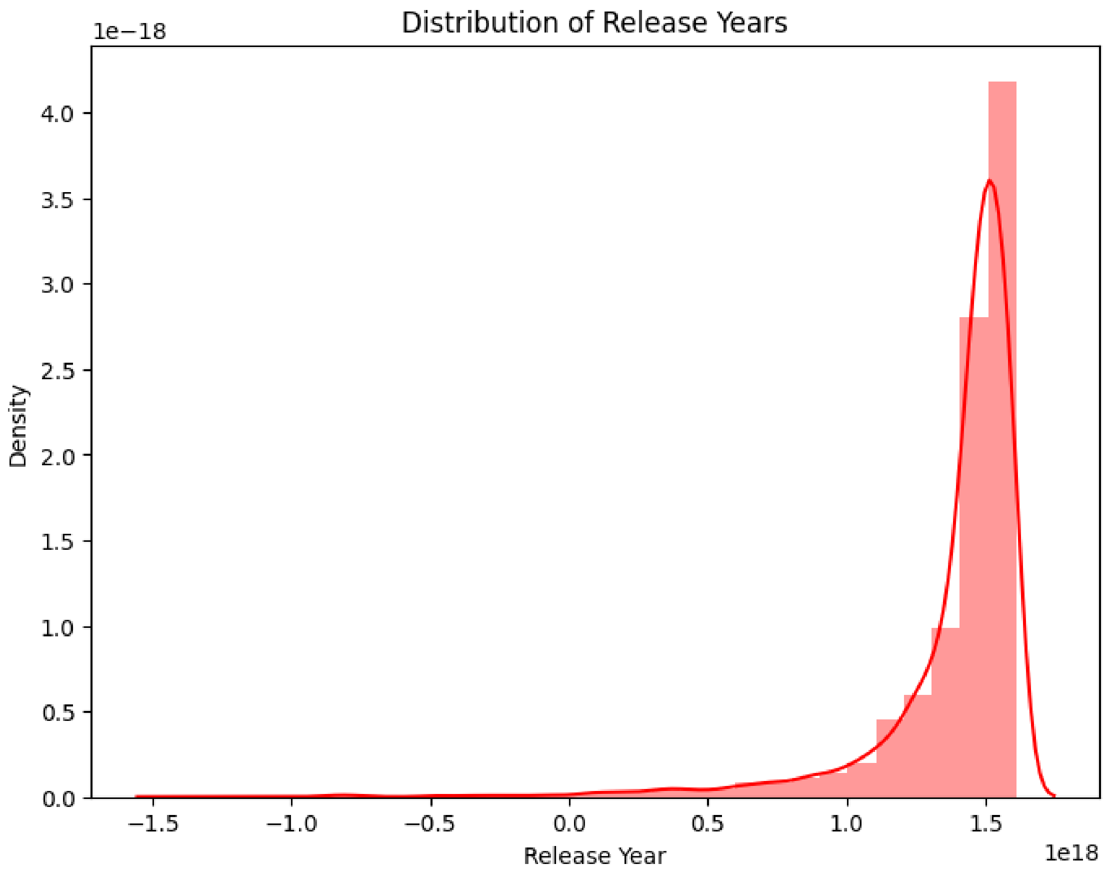


Analysis : US as a country is the highest and top country where Netflix is prevalent. Both in terms of Movies and TV shows, United States tops the chart on Netflix.

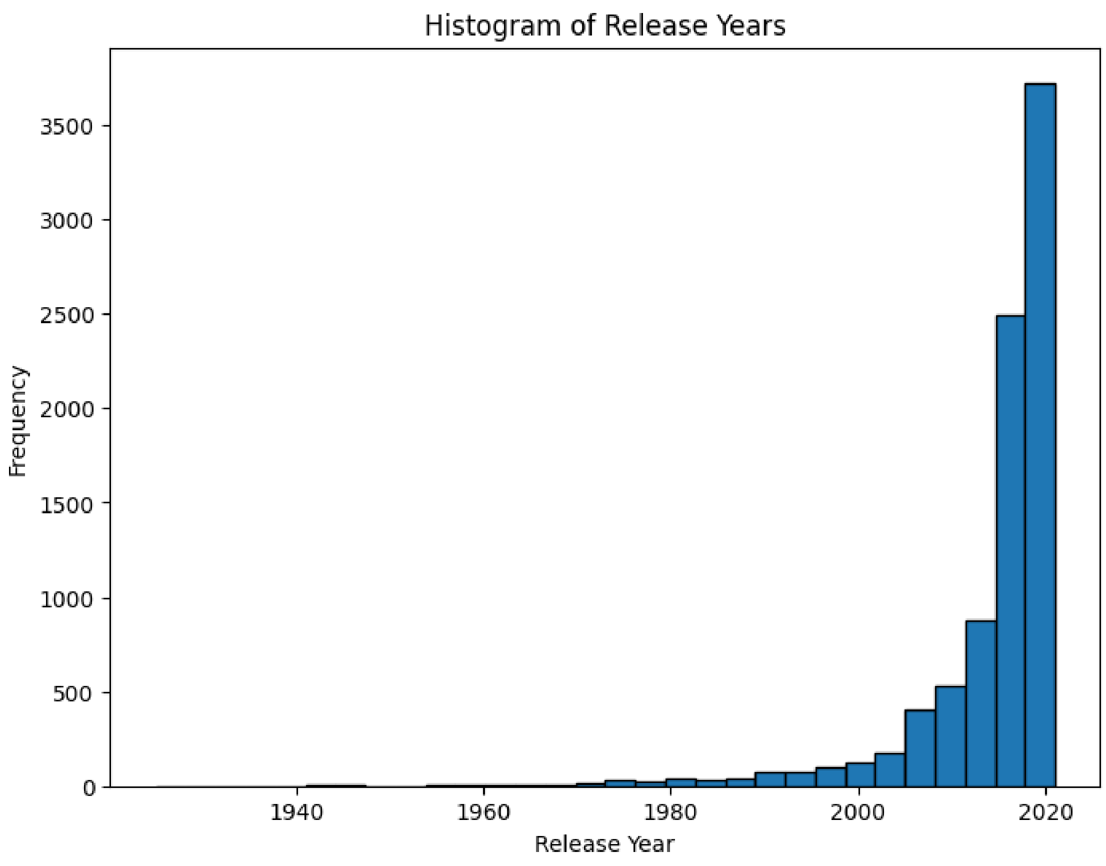
Distplot & Histogram for Release Years

[182]:
```
# Distplot for release_year
plt.figure(figsize=(8, 6))
sns.distplot(netflix['release_year'], kde=True, bins=30,color='red')
plt.title('Distribution of Release Years')
plt.xlabel('Release Year')
plt.ylabel('Density')
plt.show()
```

```
<ipython-input-182-9bef8bb3ed9b>:3: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(netflix['release_year'], kde=True, bins=30,color='red')
```



[183]:
```
# Histogram for release_year
plt.figure(figsize=(8, 6))
plt.hist(netflix['release_year'], bins=30, edgecolor='black')
plt.title('Histogram of Release Years')
plt.xlabel('Release Year')
plt.ylabel('Frequency')
plt.show()
```



Analysis: The right-skewed distribution of release years suggests that a large portion of the content available on Netflix is quite recent, having been released within the last ten years.

3. 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

   Hint : We expect you to create a new column and group by each week and count the total number of movies/ tv shows.

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

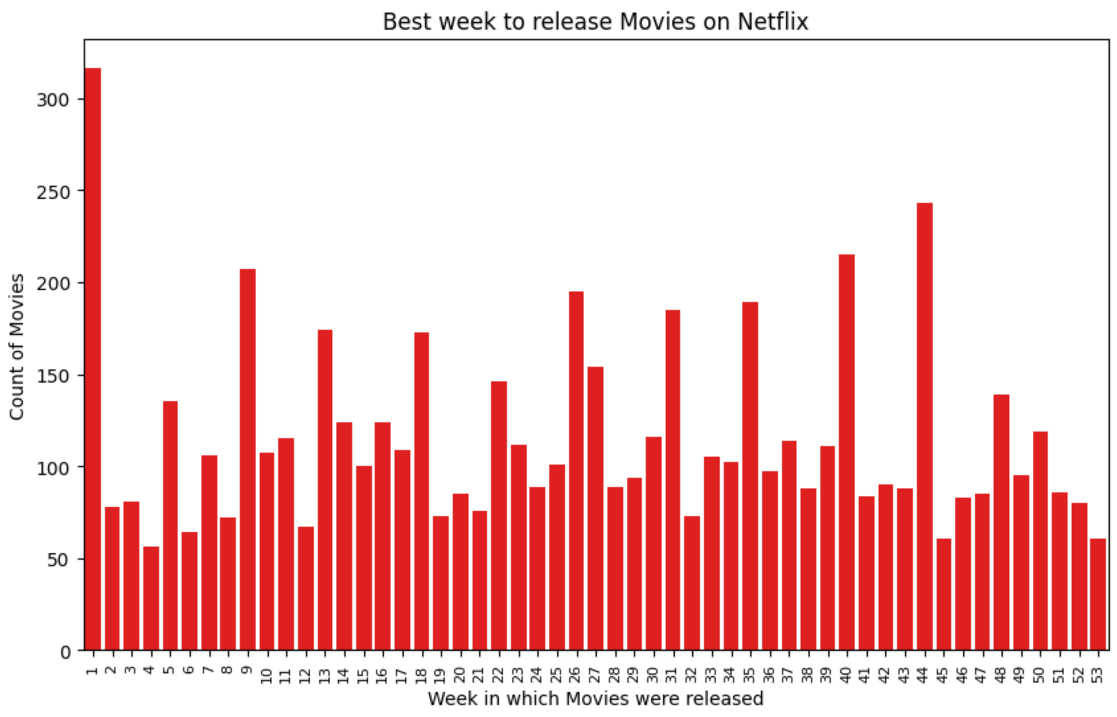
   Hint : We expect you to create a new column and group by each month and count the total number of movies/ tv shows.

[157]:
```
movies['week_added']= movies['date_added'].dt.week
tv_shows['week_added']= tv_shows['date_added'].dt.week
```

```
<ipython-input-157-202ff701684c>:1: FutureWarning: Series.dt.weekofyear and
Series.dt.week have been deprecated. Please use Series.dt.isocalendar().week
instead.
  movies['week_added']= movies['date_added'].dt.week
<ipython-input-157-202ff701684c>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  movies['week_added']= movies['date_added'].dt.week
<ipython-input-157-202ff701684c>:2: FutureWarning: Series.dt.weekofyear and
Series.dt.week have been deprecated. Please use Series.dt.isocalendar().week
instead.
  tv_shows['week_added']= tv_shows['date_added'].dt.week
<ipython-input-157-202ff701684c>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  tv_shows['week_added']= tv_shows['date_added'].dt.week
```

[158]:
```
week_preferred_movies = movies.groupby(['week_added'])['show_id'].count().
  ↪reset_index()
sorted_week_preferred_movies = week_preferred_movies.
  ↪sort_values(by='week_added')

week_preferred_tvshows = tv_shows.groupby(['week_added'])['show_id'].count().
  ↪reset_index()
sorted_week_preferred_tvshows = week_preferred_tvshows.
  ↪sort_values(by='week_added')
```

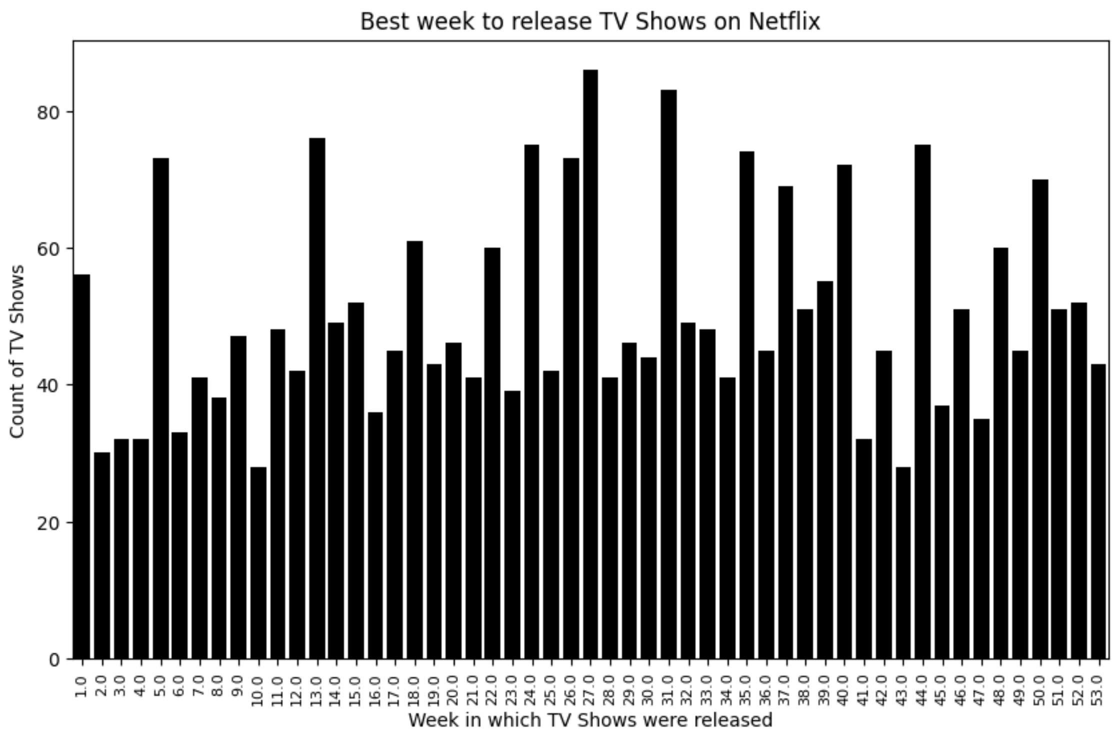
[169]:
```
plt.figure(figsize = (10,6))
sns.barplot(data=sorted_week_preferred_movies, x = 'week_added', y =␣
  ↪'show_id',color='red')
plt.title('Best week to release Movies on Netflix')
plt.xlabel('Week in which Movies were released')
plt.ylabel('Count of Movies')
plt.xticks(rotation=90,fontsize=8)
plt.show()
```



Analysis: The above bar plot shows that the most preferred week to upload content is January and November, which shows that the holiday season is the best season to upload and people are eager to watch them during their free time.

January marks the new year plus christmas vacation time, and the release during those times gives lot of views on Netflix, November being the holiday season for most of the people in international countries also helps netflix gain more views and hence popularity.

[197]:
```
plt.figure(figsize = (10,6))
sns.barplot(data=sorted_week_preferred_tvshows, x = 'week_added', y =␣
↪'show_id',color='black')
plt.title('Best week to release TV Shows on Netflix')
plt.xlabel('Week in which TV Shows were released')
plt.ylabel('Count of TV Shows')
plt.xticks(rotation=90,fontsize=8)
plt.show()
```



Analysis: The above bar plot shows that the most preferred week to upload content is May, June, end of the year which shows that the school holidays or college holidays are the most preferred time to upload TV show content. This shows that the main audience for TV show is from teenage group, either studying or going to colleges.

[118]:
```
month_preferred_movies = movies.groupby(['month_name_added' ,␣
 ↪'month_added'])['show_id'].count().reset_index()
sorted_month_preferred_movies = month_preferred_movies.
 ↪sort_values(by='month_added')

month_preferred_tvshows = tv_shows.groupby(['month_name_added' ,␣
 ↪'month_added'])['show_id'].count().reset_index()
sorted_month_preferred_tvshows = month_preferred_tvshows.
 ↪sort_values(by='month_added')
```

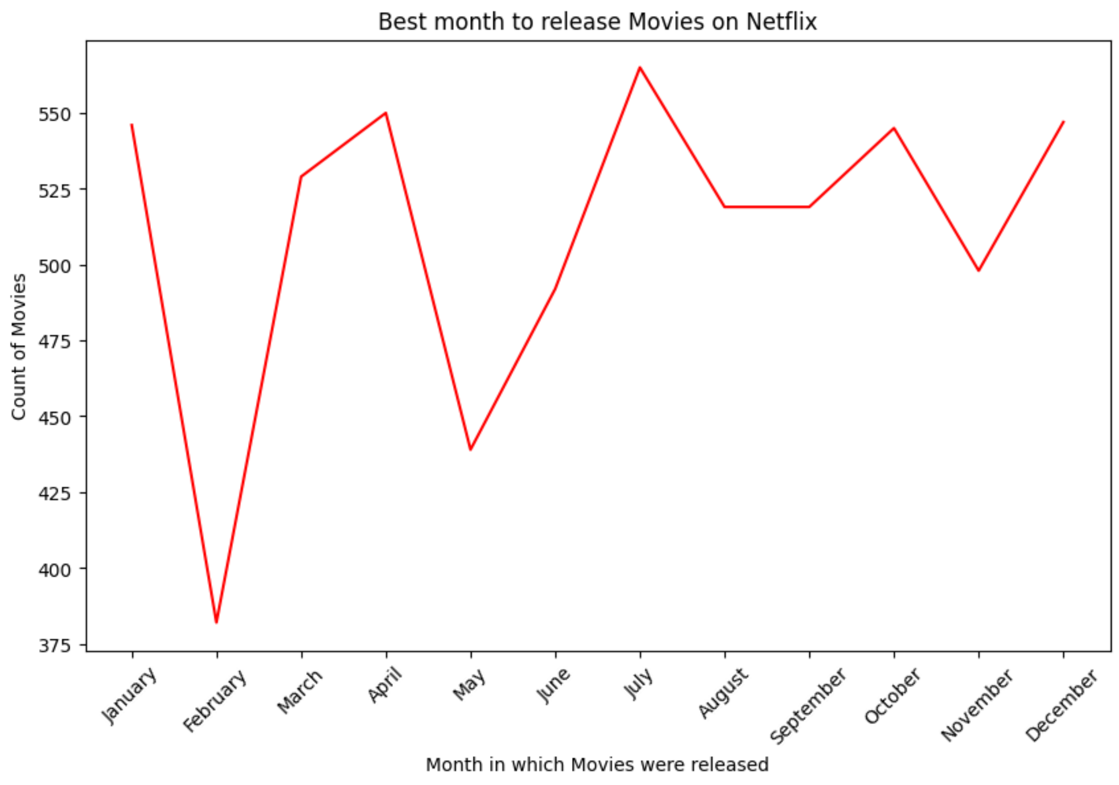
[122]:
```
sorted_month_preferred_movies
```

[122]:
```
  month_name_added  month_added  show_id
4          January          1.0      546
3         February          2.0      382
7            March          3.0      529
0            April          4.0      550
8              May          5.0      439
```

```
6            June        6.0     492
5            July        7.0     565
1          August        8.0     519
11      September        9.0     519
10        October       10.0     545
9        November       11.0     498
2        December       12.0     547
```

[139]:
```
plt.figure(figsize = (10,6))
sns.lineplot(data=sorted_month_preferred_movies, x = 'month_name_added', y =␣
 ↪'show_id',color='red')
plt.title('Best month to release Movies on Netflix')
plt.xlabel('Month in which Movies were released')
plt.ylabel('Count of Movies')
plt.xticks(rotation=45)
```

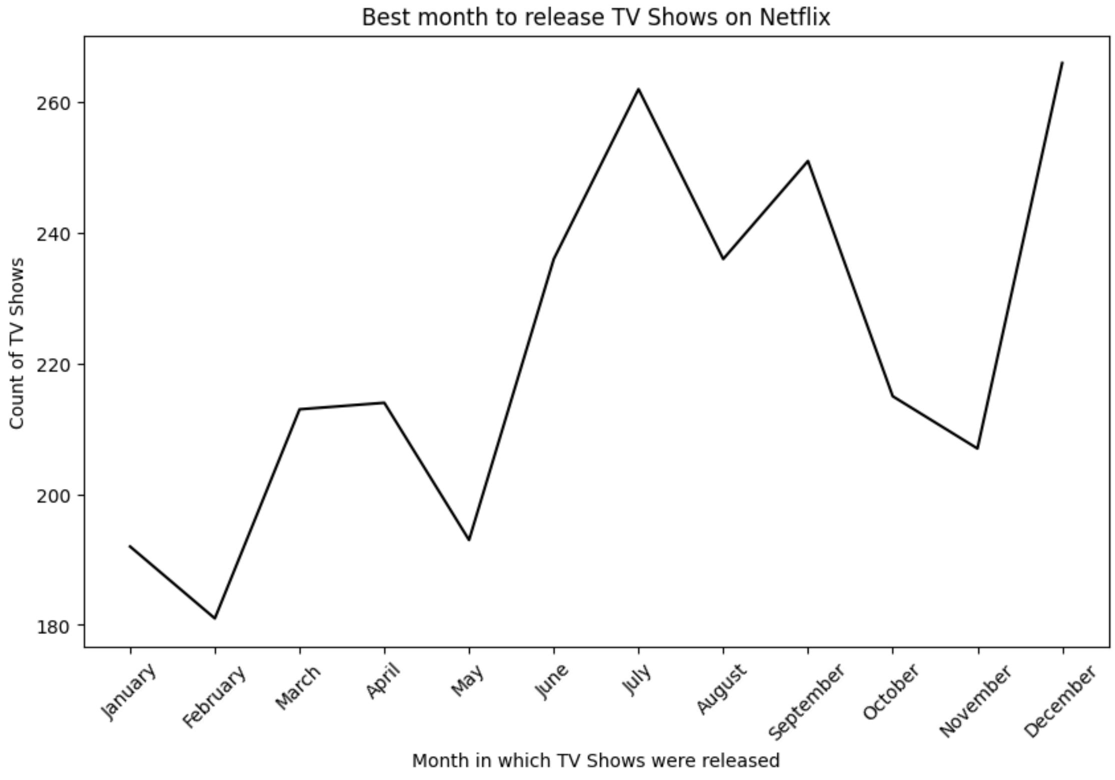
[139]:
```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11],
 [Text(0, 0, 'January'),
  Text(1, 0, 'February'),
  Text(2, 0, 'March'),
  Text(3, 0, 'April'),
  Text(4, 0, 'May'),
  Text(5, 0, 'June'),
  Text(6, 0, 'July'),
  Text(7, 0, 'August'),
  Text(8, 0, 'September'),
  Text(9, 0, 'October'),
  Text(10, 0, 'November'),
  Text(11, 0, 'December')])
```



Analysis: The line graph shows that July, December,January are the months when Netflix adds the most movie content to its library. This information can be valuable for viewers who want to anticipate new releases during these months.

[124]:
```python
plt.figure(figsize = (10,6))
sns.lineplot(data=sorted_month_preferred_tvshows, x = 'month_name_added', y =␣
 ↪'show_id',color='black')
plt.title('Best month to release TV Shows on Netflix')
plt.xlabel('Month in which TV Shows were released')
plt.ylabel('Count of TV Shows')
plt.xticks(rotation=45)
```

[124]:
```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11],
 [Text(0, 0, 'January'),
  Text(1, 0, 'February'),
  Text(2, 0, 'March'),
  Text(3, 0, 'April'),
  Text(4, 0, 'May'),
  Text(5, 0, 'June'),
  Text(6, 0, 'July'),
  Text(7, 0, 'August'),
  Text(8, 0, 'September'),
  Text(9, 0, 'October'),
  Text(10, 0, 'November'),
  Text(11, 0, 'December')])
```



Analysis: The line graph shows that July and December are the months when Netflix adds the most TV show content to its library. This information can be valuable for viewers who want to anticipate new releases during these months.

This could indicate that Netflix aims to capitalize on holiday free time and the new year period when viewers are more likely to engage with content. Launching new seasons during these months could potentially result in higher viewership and engagement rates.

4. 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.

   Hint : We want you to group by each actor and find the count of unique titles of Tv-shows/movies

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

   Hint : We want you to group by each director and find the count of unique titles of Tv-shows/movies

```
[50]: tmp_df = netflix.groupby(["director"])[["title"]].count()
      tmp_df
```

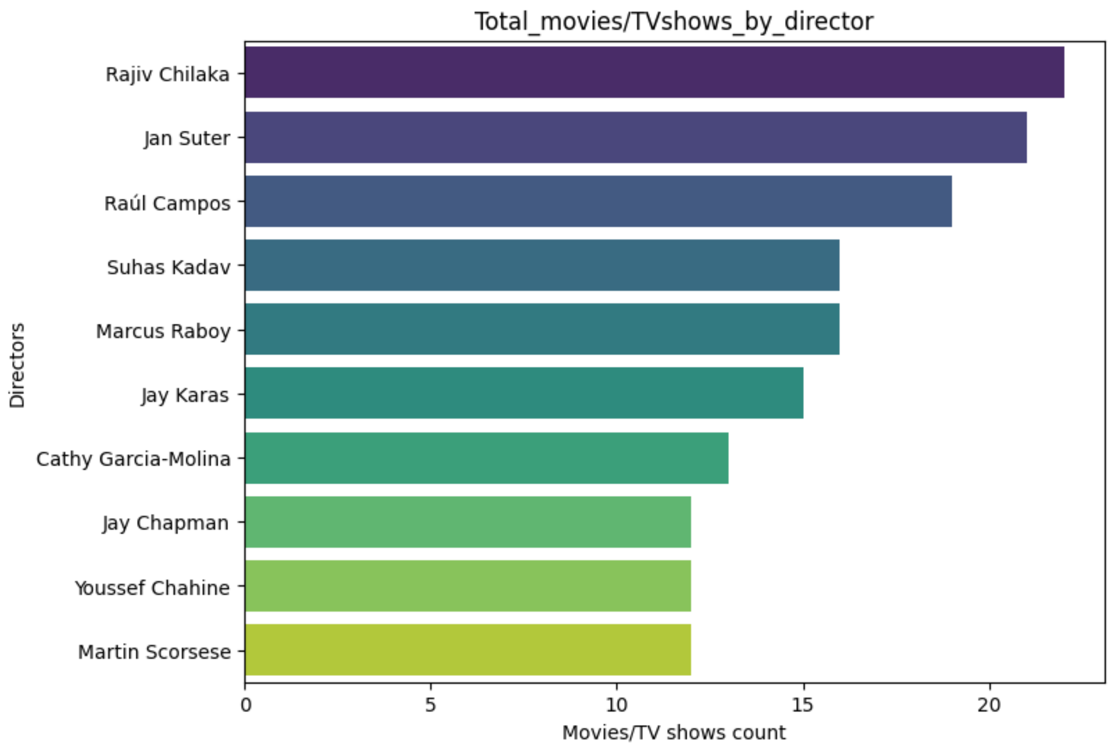
```
[50]:                      title
      director
      A. L. Vijay              2
      A. Raajdheep             1
      A. Salaam                1
      A.R. Murugadoss          2
      Aadish Keluskar          1
      …                      …
      Çagan Irmak              1
      Ísold Uggadóttir         1
      Óskar Thór Axelsson      1
      Ömer Faruk Sorak         2
      Şenol Sönmez             2

      [4529 rows x 1 columns]
```

```
[62]: # total Movies directed by top 10 known directors
      netflix_dir_known = netflix_dir[netflix_dir['director']!='Unknown director']
      top_10_dir = netflix_dir_known.director.value_counts().head(10).index
      df_dir_10 = netflix_dir.loc[netflix_dir['director'].isin(top_10_dir)]
      df_dir_10
```

```
[62]:      show_id   type              director
      406     s407  Movie         Rajiv Chilaka
      407     s408  Movie         Rajiv Chilaka
      408     s409  Movie         Rajiv Chilaka
      409     s410  Movie         Rajiv Chilaka
      410     s411  Movie         Rajiv Chilaka
      …        …     …                    …
      7513   s7514  Movie           Suhas Kadav
      7820   s7821  Movie      Martin Scorsese
      8272   s8273  Movie      Martin Scorsese
      8735   s8736  Movie      Martin Scorsese
      8789   s8790  Movie  Cathy Garcia-Molina

      [158 rows x 3 columns]
```

```
[64]: plt.figure(figsize= (8 , 6))
      sns.countplot(data = df_dir_10 , y = 'director' , order = top_10_dir , orient =␣
       ↪'v',palette='viridis')
      plt.xlabel('Total_movies/TV shows' , fontsize = 10)
      plt.xlabel('Movies/TV shows count')
      plt.ylabel('Directors' , fontsize = 10)
      plt.title('Total_movies/TVshows_by_director')
plt.show()
```

```
<ipython-input-64-78ee58f69347>:2: FutureWarning:

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

  sns.countplot(data = df_dir_10 , y = 'director' , order = top_10_dir , orient
= 'v',palette='viridis')
```



Analysis : The bar chart displays the top 10 directors with the most movies or TV shows. Rajiv Chilaka seems to have directed the most content in the Netflix library.

[65]:
```
x = netflix_dir_known.director.value_counts()
x
```

[65]:
```
Rajiv Chilaka      22
Jan Suter          21
Raúl Campos        19
Suhas Kadav        16
Marcus Raboy       16
                   ..
```

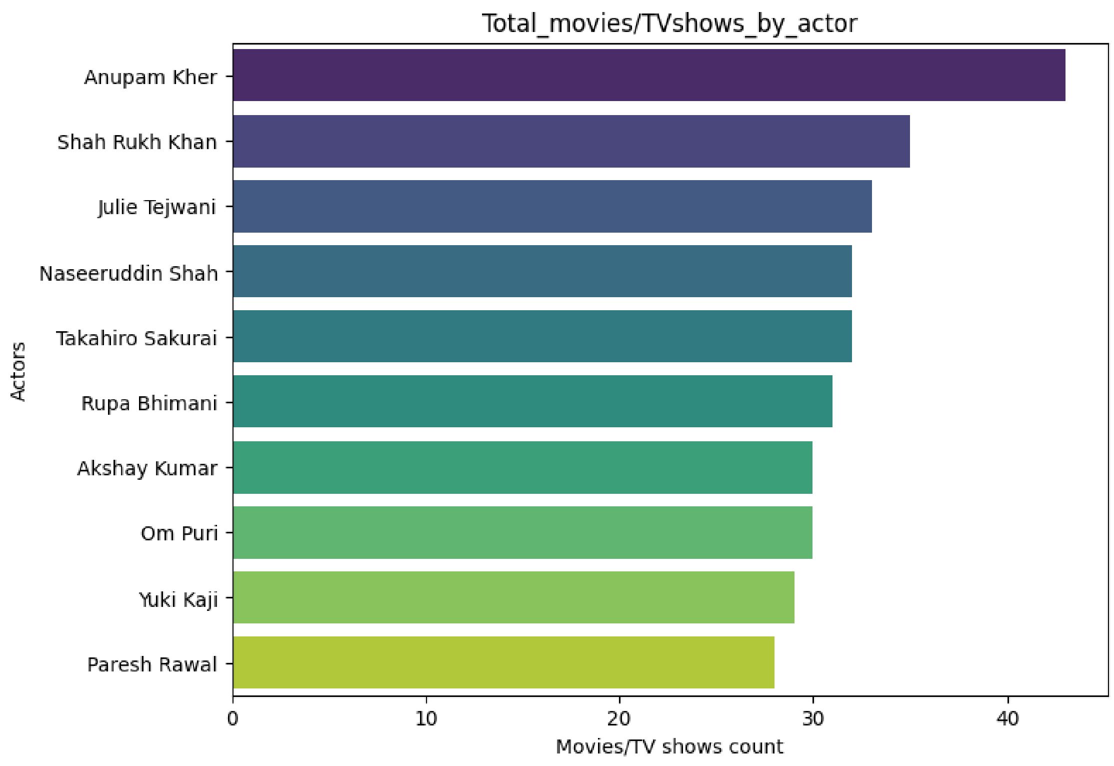
```
Raymie Muzquiz      1
Stu Livingston      1
Joe Menendez        1
Eric Bross          1
Mozez Singh         1
Name: director, Length: 4993, dtype: int64
```

[67]:
```
# total Movies and TV shows - top 10 known actors
netflix_cast_known = netflix_cast[netflix_cast['cast']!='Unknown cast']
top_10_actor = netflix_cast_known['cast'].value_counts().head(10).index
df_actor_10 = netflix_cast.loc[netflix_cast['cast'].isin(top_10_actor)]
df_actor_10
```

[67]:
```
     show_id     type              cast
39       s40  TV Show     Julie Tejwani
39       s40  TV Show      Rupa Bhimani
89       s90  TV Show     Julie Tejwani
89       s90  TV Show      Rupa Bhimani
114     s115    Movie    Shah Rukh Khan
…         …       …                …
8674   s8675    Movie           Om Puri
8687   s8688    Movie           Om Puri
8688   s8689    Movie  Naseeruddin Shah
8769   s8770    Movie       Anupam Kher
8772   s8773    Movie       Anupam Kher

[323 rows x 3 columns]
```

[68]:
```
plt.figure(figsize= (8 , 6))
sns.countplot(data = df_actor_10 , y = 'cast' , order = top_10_actor , orient =␣
 ↪'v',palette='viridis')
plt.xlabel('Total_movies/TV shows' , fontsize = 10)
plt.xlabel('Movies/TV shows count')
plt.ylabel('Actors' , fontsize = 10)
plt.title('Total_movies/TVshows_by_actor')
plt.show()
```

```
<ipython-input-68-8b584bb97f63>:2: FutureWarning:

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

  sns.countplot(data = df_actor_10 , y = 'cast' , order = top_10_actor , orient
= 'v',palette='viridis')
```



Analysis : The bar chart shows that Anupam Kher has the highest appearances in movies and TV shows.

[69]:
```
y = netflix_cast_known.cast.value_counts()
y
```

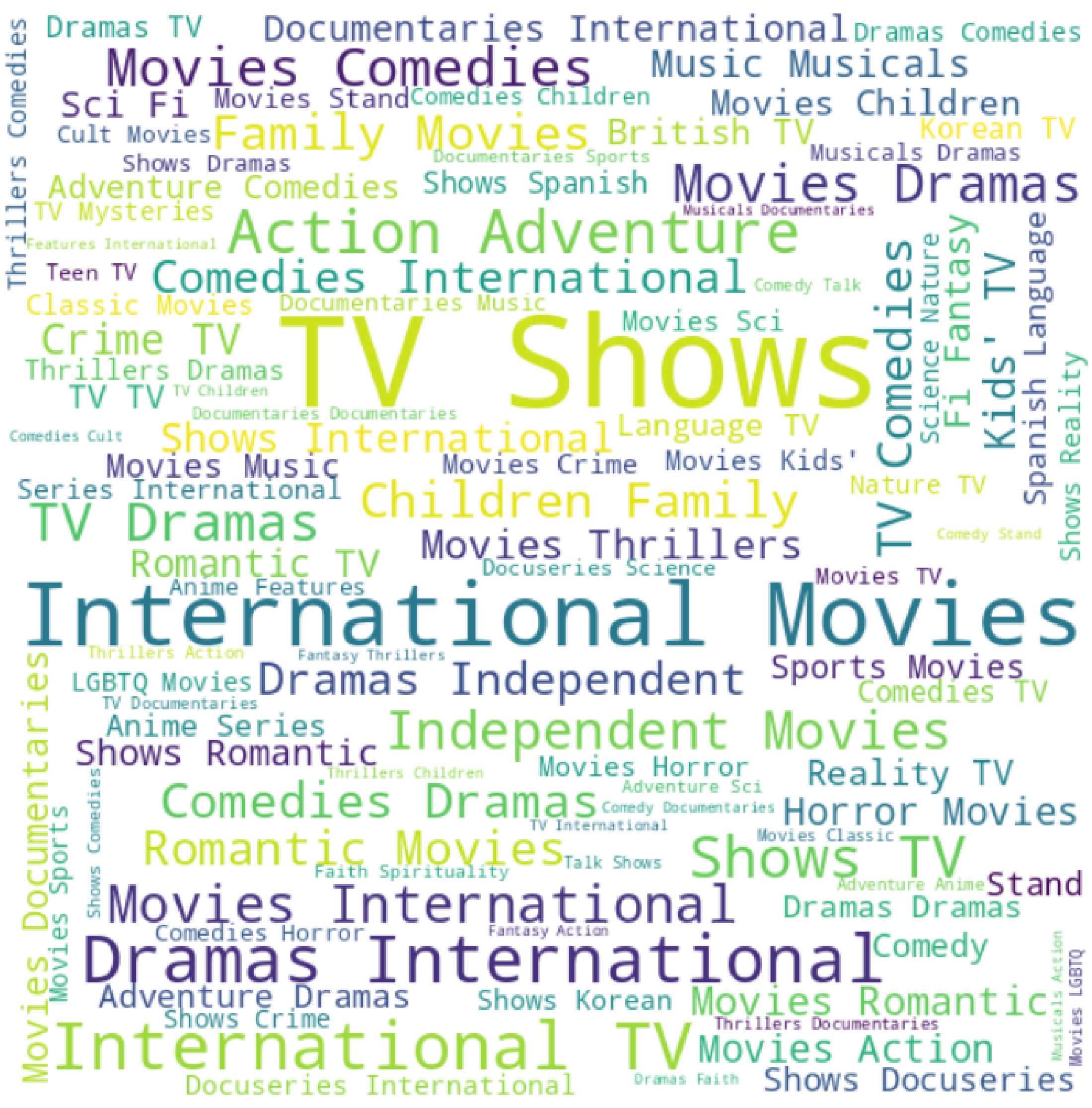
[69]:
```
Anupam Kher                43
Shah Rukh Khan             35
Julie Tejwani              33
Naseeruddin Shah           32
Takahiro Sakurai           32
                           ..
Maryam Zaree                1
Melanie Straub              1
Gabriela Maria Schmeide     1
Helena Zengel               1
Chittaranjan Tripathy       1
Name: cast, Length: 36439, dtype: int64
```

5. Which genre movies are more popular or produced more

   Hint : We want you to apply the word cloud on the genre columns to know which kind of genre is produced

[79]:
```
from wordcloud import WordCloud , STOPWORDS, ImageColorGenerator
```

[86]:
```
# Concatenate all the genres into a single string
text = ' '.join(netflix_list['listed_in'])

wordcloud = WordCloud(width = 600, height = 600,
                background_color ='white',
                min_font_size = 8).generate(text)

# plot the WordCloud image
plt.figure(figsize = (7, 7), facecolor = None)
plt.imshow(wordcloud,interpolation='Bilinear')
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()
```

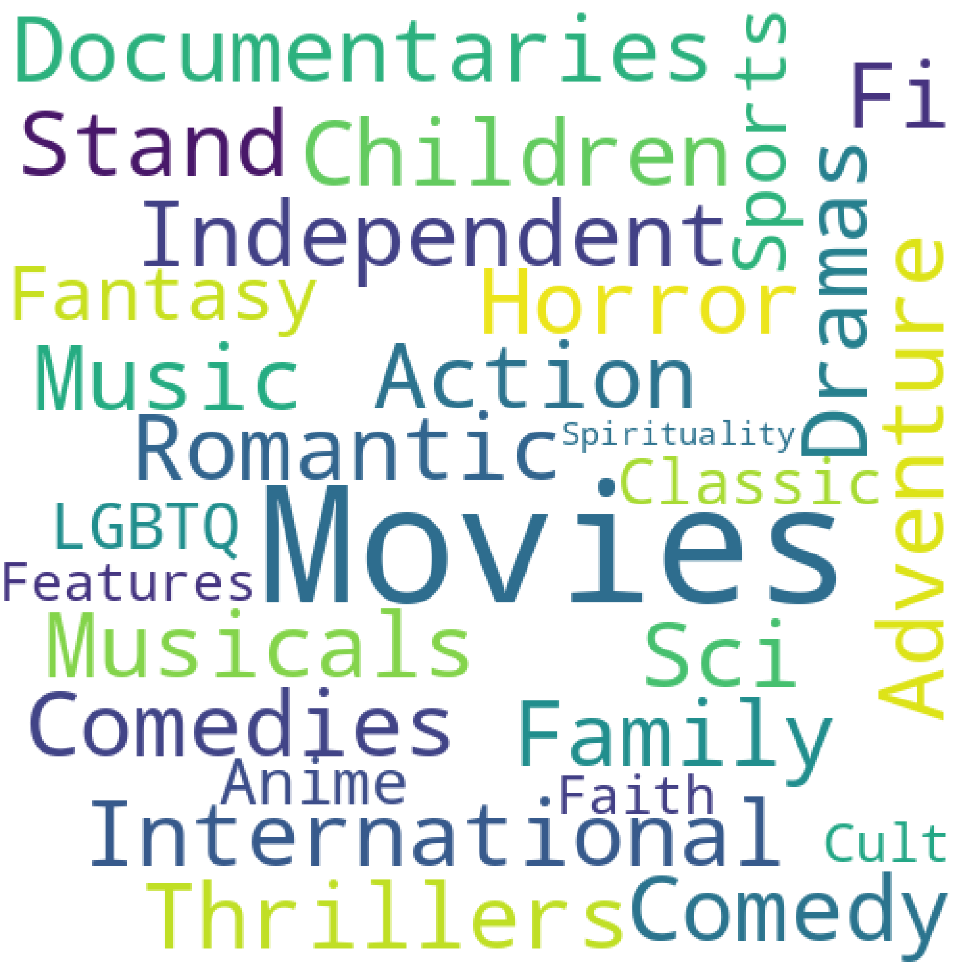


Analysis: We can see from the above wordplot that the most shows are of the genre International movies, are TV shows, Drama, comedies.

[87]:
```
#movie genres - popular
popular_genres = netflix_list[netflix_list['type'] == 'Movie'].listed_in.
  ↪value_counts().index
popular_genres.values
```

[87]:
```
array(['International Movies', 'Dramas', 'Comedies', 'Documentaries',
       'Action & Adventure', 'Independent Movies',
       'Children & Family Movies', 'Romantic Movies', 'Thrillers',
       'Music & Musicals', 'Horror Movies', 'Stand-Up Comedy',
       'Sci-Fi & Fantasy', 'Sports Movies', 'Classic Movies',
        'LGBTQ Movies', 'Anime Features', 'Cult Movies',
        'Faith & Spirituality', 'Movies'], dtype=object)
```

[90]:
```
#MOVIE GENRES:

# Concatenate all the genres into a single string
text = ' '.join(popular_genres.values)

wordcloud = WordCloud(width = 500, height = 500,
                background_color ='white',
                min_font_size = 3).generate(text)

# plot the WordCloud image
plt.figure(figsize = (5, 5), facecolor = None)
plt.imshow(wordcloud,interpolation='Bilinear')
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()
```



[92]:
```
#TV Shows genres - popular
popular_genres_tv = netflix_list[netflix_list['type'] == 'TV Show'].listed_in.
 ↪value_counts().index
popular_genres_tv.values
```

[92]:
```
array(['International TV Shows', 'TV Dramas', 'TV Comedies',
       'Crime TV Shows', "Kids' TV", 'Docuseries', 'Romantic TV Shows',
       'Reality TV', 'British TV Shows', 'Anime Series',
       'Spanish-Language TV Shows', 'TV Action & Adventure',
       'Korean TV Shows', 'TV Mysteries', 'Science & Nature TV',
       'TV Sci-Fi & Fantasy', 'TV Horror', 'Teen TV Shows',
       'TV Thrillers', 'Stand-Up Comedy & Talk Shows',
       'Classic & Cult TV', 'TV Shows'], dtype=object)
```

[100]:
```
#TV SHOWS GENRES:

# Concatenate all the genres into a single string
text = ' '.join(popular_genres_tv.values)

wordcloud = WordCloud(width = 400, height = 400,
                background_color ='white',
                min_font_size = 3).generate(text)

# plot the WordCloud image
plt.figure(figsize = (4, 4), facecolor = None)
plt.imshow(wordcloud,interpolation='Bilinear')
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()
```

6. Find After how many days the movie will be added to Netflix after the release of the movie (you can consider the recent past data)

   Hint : We want you to get the difference between the columns having date added information and release year information and get the mode of difference. This will give an insight into what will be the better time to add in Netflix

```
[151]: netflix['release_year'] = pd.to_datetime(netflix['release_year'],format='%Y')
```

```
[177]: recent_data = netflix[netflix['release_year']>'2000-01-01']
       days_to_add = (recent_data['date_added']-recent_data['release_year']).mode()
       print("The mode or the days_to_add released movies or shows on netflix after␣
        ↪the release date is:", days_to_add[0])
```

```
The mode or the days_to_add released movies or shows on netflix after the
release date is: 334 days 00:00:00
```

##Insights based on Non-Graphical and Visual Analysis:

- Around 69.9% content on Netflix is Movies and around 30.4% content is TV shows. -The movies and TV shows uploading on the Netflix started from the year 2008, It had very lesser content till 2014. Post 2015, there was a drastic change, the content upload increased and peaked in 2019. -Year 2020 and 2021 has seen the drop in content added on Netflix, possibly because of Pandemic. But still , TV shows content have not dropped as drastic as movies. -Since 2018, there is still a small rise in TV shows, unlike the continued decrease in movies. Being in continuous uptrend , TV shows surpassed the movies count in mid 2020. It shows the rise in popularity of tv shows in recent years. -Netflix has movies from variety of directors. Around 4993 known directors have their movies or tv shows on Netflix, plus many from unknown directors as well. -United States is the highset contributor. -The release year for shows is concentrated in the range 2005-2021. 50 mins - 150 mins is the range of movie durations, excluding potential outliers. -1-3 seasons is the range for TV shows seasons, excluding potential outliers. various ratings of content is avaialble on netfilx, for the various viewers categories like kids, adults , families. Highest number of movies and TV shows are rated TV-MA (for mature audiences). -International Movies and TV Shows , Dramas , and Comedies are the top 3 genres on Netflix for both Movies and TV shows.

##Business Insights

- Netflix have majority of content which is released after the year 2000. It is observed that the content older than year 2000 is very scarce on Netflix.
- Senior Citizen could be the target audience for such content, which is almost missing currently. The current content is majorly for Teenagers and yound adult population around the globe.
- Maximum content (more than 80%) is
  - TV-MA - Content intended for mature audiences aged 17 and above.
  - TV-14 - Content suitable for viewers aged 14 and above.
  - TV-PG - Parental guidance suggested (similar ratings - PG-13 , PG)
  - R - Restricted Content, that may not be suitable for viewers under age 17.

  These ratings' movies target Matured and Adult audience. Rest 20 % of the content is for kids aged below 13. It shows that Netflix is currently serving mostly Mature audiences or Children with parental guidance.
- Most popular genres on Netflix are International Movies and TV Shows , Dramas , Comedies, Action & Adventure, Children & Family Movies, Thrillers.
- Maximum content of Netflix which is around 75% , is coming from the top 10 countries. Rest of the world only contributes 25% of the content. More countries can be focussed in future to grow the business.
- Liking towards the shorter duration content is on the rise. (duration 75 to 150 minutes and seasons 1 to 3) This can be considered while production of new content on Netflix.

##Recommendations - In most countries, except the US, very limited genres are targeted. The current available genres seem to be best suited to the US and a few countries, but there is a need for more genres that are popular in the region. for example: Indian Mythological content is highly popular. Netflix can create such more country specific genres and it might also be liked acorss the world just like Japanese Anime and Korean TV shows. - Country specific insights - The content need to be targetting the demographic of any country. Netflix can produce higher number of content in the perticular rating as per demographic of the country. For example: Country like India , which is highly populous , has maximum content available only in three rating TV-MA, TV-14 , TV-PG. It is unlikely to serve below 14 age and above 35 year age group. This demographic barrier can be broken if Netflix adds more and more content suitable for majority of the age groups.

###Data-Backed Recommendations

1. Expand Older TV Show Portfolio

Quantifiable Insight: Compared to movies, the median release date for television series is a little older. By the year 2000, only a small percentage of the available television programmes had been released. Recommendation: To attract a wider age group, including older adults who may have fond memories of old series, Netflix could consider adding more classic TV shows to its catalogue in the light of this focus on new television programmes.

2. Regional Customization

Quantifiable Insight: Nearly 50% of the entire Netflix catalogue is made up of content originating in the US, India and UK. Recommendation: With content from 748 different country combinations available, Netflix has the opportunity to further customize its offering on the basis of regional popularity. This could lead to an increase in local subscriptions and customer satisfaction.

3. Explore Underrepresented Genres and Ratings

Quantifiable Insight: Ratings 'TV-MA' and 'TV-14' account for 61.2% of all content. There are fewer genres in the catalogue, such as documentary films and children's movies. Recommendation: To attract a more diverse audience, Netflix could broaden its portfolio by examining lesser known genres and television ratings.

4. Seasonal Releases

Quantifiable Insight: There is a noticeable spike in the number of TV shows added during July, December and January, suggesting these are peak months for new releases. Recommendation: Given this seasonal trend, Netflix could focus on releasing highly anticipated new seasons or exclusive content during these months to capitalize on increased viewership.

```
[ ]:
```