

Business_Case_Netflix

December 20, 2023

```
[ ]: !gdown https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/000/940/
      ↪original/netflix.csv
```

Downloading...

From: https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/000/940/original/netflix.csv

To: /content/netflix.csv

100% 3.40M/3.40M [00:00<00:00, 33.3MB/s]

```
[ ]:
```

0.0.1 1. Defining Problem Statement and Analysing basic metrics:

PROBLEM STATEMENT:

The goal is to analyze the Netflix dataset to derive meaningful insights that can inform decision-making. Specifically, aims to understand the distribution of content, identify trends in release years, explore the popularity of genres, and gain insights into the production landscape by examining countries and directors

ANALYSING BASIC METRICS:

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

      df = pd.read_csv('netflix.csv')
      df.head()
```

```
[ ]: 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
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...

```
[ ]: df.info()
```

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

```
[ ]:
```

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

```
[ ]:
```

```
# Shape of the data  
  
df.shape  
print('No. of Rows =',df.shape[0])  
print('No. of Cols =',df.shape[1])
```

```
No. of Rows = 8807
```

```
No. of Cols = 12
```

```
[ ]:
```

```
# data types of all the attributes  
  
data_types = df.dtypes  
data_types
```

```
[ ]:
```

```
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
```

```
[ ]:
```

```
# Conversion of categorical attributes to 'category'  
  
Categories = ['type','listed_in','title','country','description']  
Categories = df[Categories].astype('category')  
Categories.dtypes
```

```
[ ]:
```

```
type         category  
listed_in    category  
title        category  
country      category  
description   category  
dtype: object
```

```
[ ]: # missing value detection

data_missing_values = df.isna().sum()
data_missing_values
```

```
[ ]: 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

Stat_summary = df.describe(include='all')
Stat_summary
```

```
[ ]:      show_id  type      title      director \
count      8807   8807      8807      6173
unique      8807     2      8807      4528
top         s1  Movie  Dick Johnson Is Dead  Rajiv Chilaka
freq         1   6131         1         19
mean        NaN    NaN         NaN         NaN
std         NaN    NaN         NaN         NaN
min         NaN    NaN         NaN         NaN
25%         NaN    NaN         NaN         NaN
50%         NaN    NaN         NaN         NaN
75%         NaN    NaN         NaN         NaN
max         NaN    NaN         NaN         NaN

      cast      country      date_added  release_year \
count      7982      7976      8797      8807.000000
unique      7692      748      1767         NaN
top  David Attenborough  United States  January 1, 2020         NaN
freq         19      2818         109         NaN
mean         NaN         NaN         NaN      2014.180198
std         NaN         NaN         NaN         8.819312
min         NaN         NaN         NaN      1925.000000
25%         NaN         NaN         NaN      2013.000000
50%         NaN         NaN         NaN      2017.000000
```

75%	NaN	NaN	NaN	2019.000000
max	NaN	NaN	NaN	2021.000000

	rating	duration	listed_in \
count	8803	8804	8807
unique	17	220	514
top	TV-MA	1 Season	Dramas, International Movies
freq	3207	1793	362
mean	NaN	NaN	NaN
std	NaN	NaN	NaN
min	NaN	NaN	NaN
25%	NaN	NaN	NaN
50%	NaN	NaN	NaN
75%	NaN	NaN	NaN
max	NaN	NaN	NaN

	description
count	8807
unique	8775
top	Paranormal activity at a lush, abandoned prope...
freq	4
mean	NaN
std	NaN
min	NaN
25%	NaN
50%	NaN
75%	NaN
max	NaN

```
[ ]:
```

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

```
[ ]: # Type : Movies & TV shows

type_count = df['type'].value_counts()
type_nunique = df['type'].nunique()

print('Type count:\n\n',type_count)
print('\nUnique type:',type_nunique)
```

Type count:

```
Movie      6131
TV Show    2676
Name: type, dtype: int64
```

Unique type: 2

```
[ ]: # Title of the Movies & TV shows

title_count = df['title'].value_counts()
title_nunique = df['title'].nunique()

print('Title count:\n\n',title_count)
print('\nUnique title:',title_nunique)
```

Title count:

Dick Johnson Is Dead	1
Ip Man 2	1
Hannibal Buress: Comedy Camisado	1
Turbo FAST	1
Masha's Tales	1
..	
Love for Sale 2	1
ROAD TO ROMA	1
Good Time	1
Captain Underpants Epic Choice-o-Rama	1
Zubaan	1

Name: title, Length: 8807, dtype: int64

Unique title: 8807

```
[ ]: # Unique Directors Count:

director_count = df['director'].value_counts()
director_nunique = df['director'].nunique()

print('Directors count:\n\n',director_count)
print('\nTotal Unique directors:',director_nunique)
```

Directors count:

Rajiv Chilaka	19
Raúl Campos, Jan Suter	18
Marcus Raboy	16
Suhas Kadav	16
Jay Karas	14
..	
Raymie Muzquiz, Stu Livingston	1
Joe Menendez	1
Eric Bross	1
Will Eisenberg	1
Mozes Singh	1

Name: director, Length: 4528, dtype: int64

Total Unique directors: 4528

```
[ ]: # Cast Count:

Cast_count = df['cast'].value_counts()
Cast_nunique = df['cast'].nunique()

print('Casts in the Movies & TV shows:\n\n',Cast_count)
print('\nTotal Cast count:',Cast_nunique)
```

Casts in the Movies & TV shows:

```
David Attenborough
19
Vatsal Dubey, Julie Tejawani, Rupa Bhimani, Jigna Bhardwaj, Rajesh Kava, Mousam,
Swapnil
14
Samuel West
10
Jeff Dunham
7
David Spade, London Hughes, Fortune Feimster
6
..
Michael Peña, Diego Luna, Tenoch Huerta, Joaquin Cosio, José María Yazpik, Matt
Letscher, Alyssa Diaz
1
Nick Lachey, Vanessa Lachey
1
Takeru Sato, Kasumi Arimura, Haru, Kentaro Sakaguchi, Takayuki Yamada, Kendo
Kobayashi, Ken Yasuda, Arata Furuta, Suzuki Matsuo, Koichi Yamadera, Arata Iura,
Chikako Kaku, Kotaro Yoshida      1
Toyin Abraham, Sambasa Nzeribe, Chioma Chukwuka Akpotha, Chioma Omeruah,
Chiwetalu Agu, Dele Odule, Femi Adebayo, Bayray McNwizu, Biodun Stephen
1
Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanana, Manish Chaudhary, Meghna Malik,
Malkeet Rauni, Anita Shabdish, Chittaranjan Tripathy
1
Name: cast, Length: 7692, dtype: int64
```

Total Cast count: 7692

```
[ ]: # Country Count:

Country_count = df['country'].value_counts()
Country_nunique = df['country'].nunique()
```

```
print('Country wise count:\n\n',Country_count)
print('\nTotal unique countries:',Country_nunique)
```

Country wise count:

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: country, Length: 748, dtype: int64

Total unique countries: 748

```
[ ]: # date_added Count:

date_added_count = df['date_added'].value_counts()
date_added_nunique = df['date_added'].nunique()

print('date_added count:\n\n',date_added_count)
print('\nTotal unique date_added:',date_added_nunique)
```

date_added count:

January 1, 2020	109
November 1, 2019	89
March 1, 2018	75
December 31, 2019	74
October 1, 2018	71
...	
December 4, 2016	1
November 21, 2016	1
November 19, 2016	1
November 17, 2016	1
January 11, 2020	1

Name: date_added, Length: 1767, dtype: int64

Total unique date_added: 1767

```
[ ]: # release_year Count:
```



```

release_year_count = df['release_year'].value_counts()
release_year_nunique = df['release_year'].nunique()

print('release_year count:\n\n',release_year_count)
print('\nTotal unique release_year count:',release_year_nunique)

```

release_year count:

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

Total unique release_year count: 74

[]: *# rating Count:*

```

rating_count = df['rating'].value_counts()
rating_nunique = df['rating'].nunique()

print('rating count:\n\n',rating_count)
print('\nTotal unique rating:',rating_nunique)

```

rating count:

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: rating, dtype: int64
```

Total unique rating: 17

```
[ ]: # Duration Count:

duration_count = df['duration'].value_counts()
duration_nunique = df['duration'].nunique()

print('duration count:\n\n',duration_count)
print('\nTotal unique duration:',duration_nunique)
```

duration count:

```
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
```

Total unique duration: 220

```
[ ]: # listed_in Count:

listed_in_count = df['listed_in'].value_counts()
listed_in_nunique = df['listed_in'].nunique()

print('listed_in count:\n\n',listed_in_count)
print('\nTotal unique listed_in count:',listed_in_nunique)
```

listed_in count:

```
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: listed_in, Length: 514, dtype: int64
```

Total unique listed_in count: 514

```
[ ]: # description Count:
```

```
description_count = df['description'].value_counts()
description_nunique = df['description'].nunique()

print('description count:\n\n',description_count)
print('\nTotal unique description count:',description_nunique)
```

description count:

Paranormal activity at a lush, abandoned property alarms a group eager to redevelop the site, but the eerie events may not be as unearthly as they think.
4

Challenged to compose 100 songs before he can marry the girl he loves, a tortured but passionate singer-songwriter embarks on a poignant musical journey.
3

A surly septuagenarian gets another chance at her 20s after having her photo snapped at a studio that magically takes 50 years off her life.
3

Multiple women report their husbands as missing but when it appears they are looking for the same man, a police officer traces their cryptic connection.
3

Secrets bubble to the surface after a sensual encounter and an unforeseen crime entangle two friends and a woman caught between them. 2

..

Sent away to evade an arranged marriage, a 14-year-old begins a harrowing journey of sex work and poverty in the slums of Accra.
1

When his partner in crime goes missing, a small-time crook's life is transformed as he dedicates himself to raising the daughter his friend left behind. 1

During 1962's Cuban missile crisis, a troubled math genius finds himself drafted to play in a U.S.-Soviet chess match - and a deadly game of espionage. 1

A teen's discovery of a vintage Polaroid camera develops into a darker tale when she finds that whoever takes their photo with it dies soon afterward. 1

A scrappy but poor boy worms his way into a tycoon's dysfunctional family, while facing his fear of music and the truth about his past. 1

Name: description, Length: 8775, dtype: int64

Total unique description count: 8775

```
[ ]:
```

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

4.2 For categorical variable(s): Boxplot

4.3 For correlation: Heatmaps, Pairplots

```
[ ]: # Pre-processing involves unnesting of the data in columns like Actor, Director, Country

# Unnesting the 'cast' column
df['cast'] = df['cast'].str.split(',')
df_cast = df.explode('cast')

# Unnesting the 'listed_in' column
df['listed_in'] = df['listed_in'].str.split(',')
df_listed_in = df.explode('listed_in')

# Unnesting the 'country' column
df['country'] = df['country'].str.split(',')
df_country = df.explode('country')

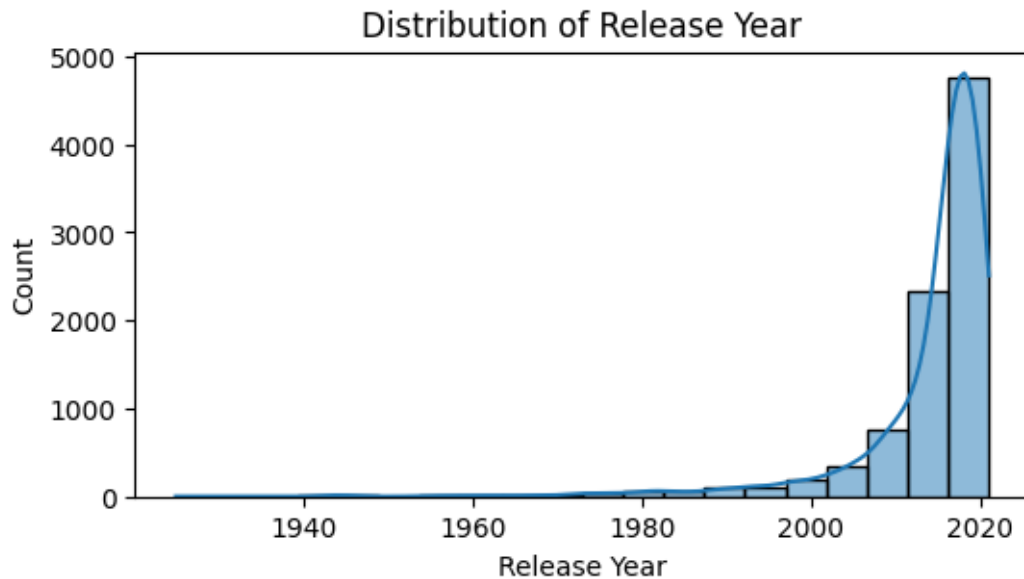
# Unnesting the 'director' column
df_director = df.explode('director')
```

```
[ ]:
```

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

```
[ ]: # Continuous variable : Release_year

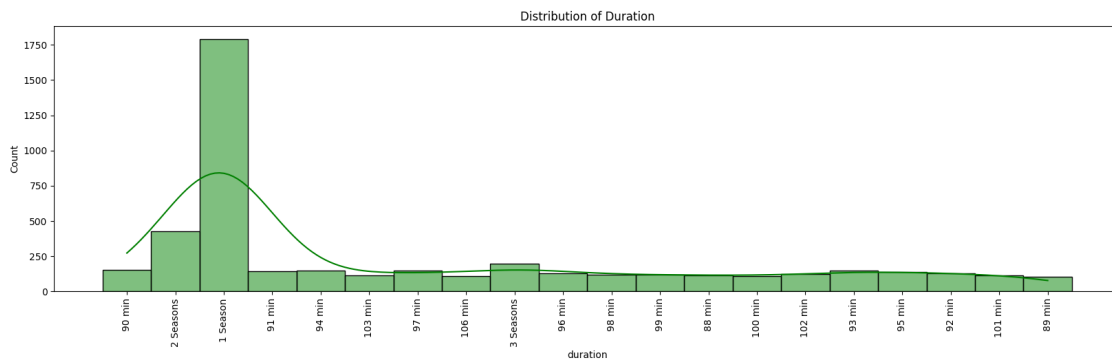
plt.figure(figsize=(6,3))
sns.histplot(data = df, x='release_year', bins = 20, kde = True)
plt.title('Distribution of Release Year')
plt.xlabel('Release Year')
plt.ylabel('Count')
plt.show()
```



```
[ ]: # Continuous variable : Duration
```

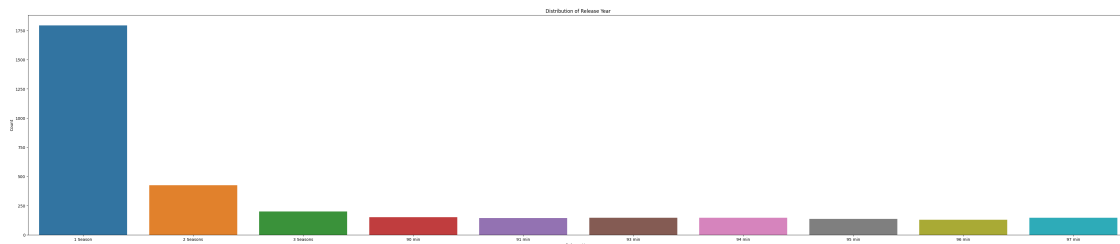
```
duration = df['duration'].value_counts().index[:20]
df_duration = df[df['duration'].isin(duration)]

plt.figure(figsize=(20,5))
sns.histplot(data = df_duration ,x='duration',bins = 20,kde = True,color='green')
plt.title('Distribution of Duration')
plt.xticks(rotation = 90)
plt.xlabel('duration')
plt.ylabel('Count')
plt.show()
```



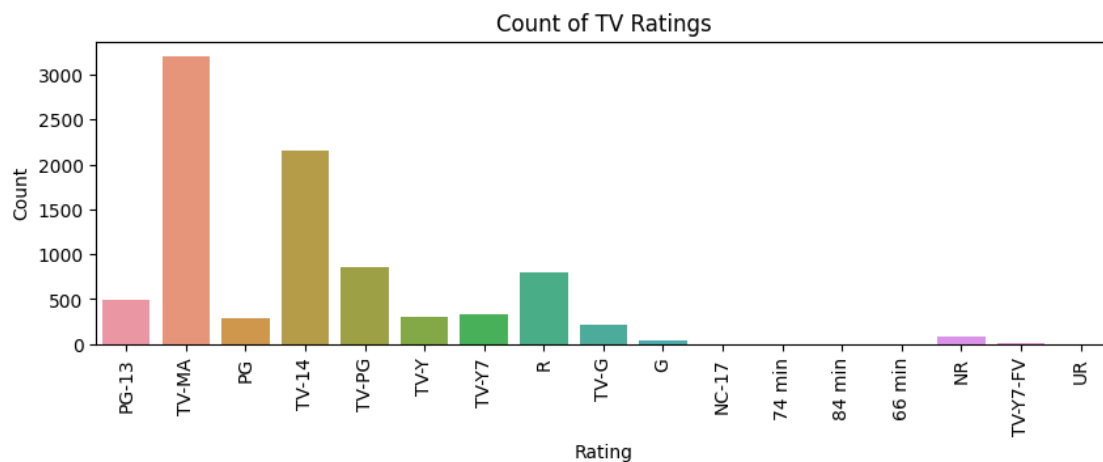
```
[ ]: # Continuous variable : Release_year(with value_counts().index)
duration = df['duration'].value_counts().index[:10]
df_duration = df[df['duration'].isin(duration)]

plt.figure(figsize=(50,10))
sns.countplot(data = df_duration,x='duration',order = df_duration['duration'].
    ↪sort_values().unique())
plt.title('Distribution of Release Year')
plt.xlabel('Release Year')
plt.ylabel('Count')
plt.show()
```



```
[ ]: # Univariate : Ratings counts

plt.figure(figsize=(10, 3))
sns.countplot(x='rating', data=df)
plt.title('Count of TV Ratings')
plt.xlabel('Rating')
plt.ylabel('Count')
plt.xticks(rotation=90)
plt.show()
```



```
[ ]:
```

4.2 For categorical variable(s): Boxplot

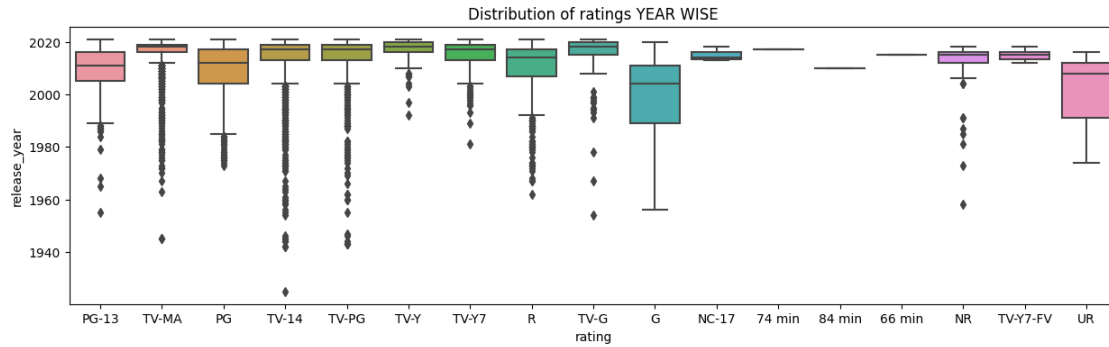
```
[ ]: # categorical variable : Year wise distribution of movies and TV shows
```

```
sns.boxplot(data = df, y = 'type', x = 'release_year')
plt.xlabel('release_year')
plt.ylabel('Types')
plt.title('Year wise distribution of movies and TV shows')
plt.show()
```



```
[ ]: # categorical variable: Distribution of ratings YEAR WISE
```

```
plt.figure(figsize=(15,4))
sns.boxplot(data = df, y = 'release_year', x = 'rating')
plt.xlabel('rating')
plt.ylabel('release_year')
plt.title('Distribution of ratings YEAR WISE')
plt.show()
```



[]:

4.3 For correlation: Heatmaps, Pairplots

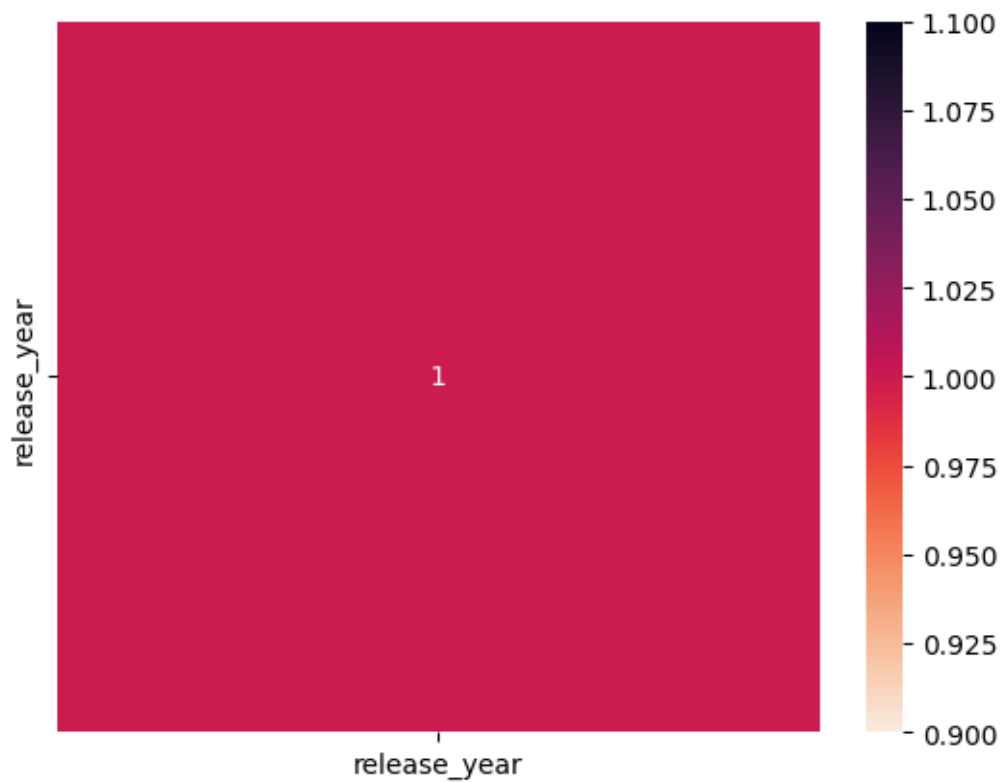
[]:

```
# Heatmap for correlation matrix

a = df[['release_year', 'duration']]
a = a.dropna()
correlation = a.corr()
sns.heatmap(correlation, annot= True, cmap = 'rocket_r')
plt.show()
```

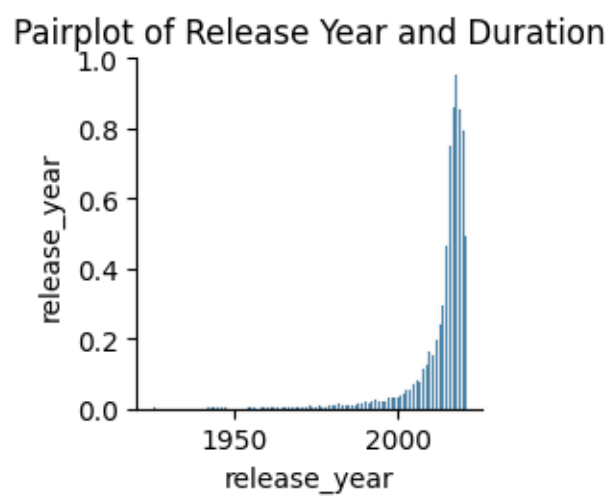
<ipython-input-23-72a80a8a1c77>:5: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.

```
correlation = a.corr()
```

```
[ ]: # Pairplot for Release_year and Duration

sns.pairplot(df[['release_year', 'duration']])
plt.title('Pairplot of Release Year and Duration')
plt.show()
```



```
[ ]:
```

5. Missing Value & Outlier check

```
[ ]: # Missing Value :
```

```
missing_value = df.isna().sum()
missing_value
```

```
[ ]: 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
```

```
[ ]: remove_missing_value = df.dropna().head()
      remove_missing_value
```

```
[ ]: show_id      type      title      director \
7      s8      Movie      Sankofa      Haile Gerima
8      s9      TV Show  The Great British Baking Show      Andy Devonshire
9      s10     Movie      The Starling      Theodore Melfi
12     s13     Movie      Je Suis Karl      Christian Schwochow
24     s25     Movie      Jeans      S. Shankar
```

```
cast \
7      [Kofi Ghanaba, Oyafunmike Ogunlano, Alexandr...
8      [Mel Giedroyc, Sue Perkins, Mary Berry, Pau...
9      [Melissa McCarthy, Chris O'Dowd, Kevin Kline...
12     [Luna Wedler, Jannis Niewöhner, Milan Pesche...
24     [Prashanth, Aishwarya Rai Bachchan, Sri Laks...
```

```
country      date_added \
7      [United States, Ghana, Burkina Faso, United... September 24, 2021
8      [United Kingdom] September 24, 2021
9      [United States] September 24, 2021
12     [Germany, Czech Republic] September 23, 2021
24     [India] September 21, 2021
```

	release_year	rating	duration	\
7	1993	TV-MA	125 min	
8	2021	TV-14	9 Seasons	
9	2021	PG-13	104 min	
12	2021	TV-MA	127 min	
24	1998	TV-14	166 min	

	listed_in	\
7	[Dramas, Independent Movies, International M...	
8	[British TV Shows, Reality TV]	
9	[Comedies, Dramas]	
12	[Dramas, International Movies]	
24	[Comedies, International Movies, Romantic Mo...	

	description
7	On a photo shoot in Ghana, an American model s...
8	A talented batch of amateur bakers face off in...
9	A woman adjusting to life after a loss contend...
12	After most of her family is murdered in a terr...
24	When the father of the man she loves insists t...

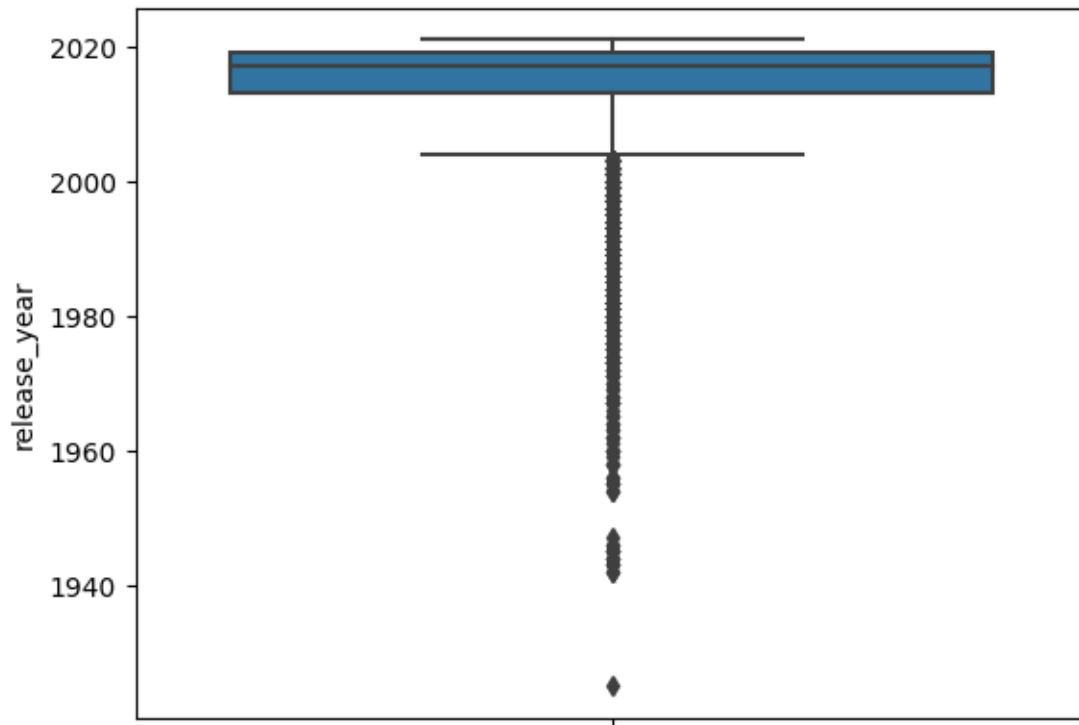
```
[ ]: fill_missing_value = df.fillna(0)
fill_missing_value.isna().sum()
```

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

```
[ ]: # Box plot for outlier visualization

numerical_columns = ['release_year', 'duration']

for column in numerical_columns:
    sns.boxplot(y=df[column])
    plt.show()
```



```

-----
TypeError                                Traceback (most recent call last)
<ipython-input-12-124d9e81a77f> in <cell line: 5>()
      4
      5 for column in numerical_columns:
----> 6     sns.boxplot(y=df[column])
      7     plt.show()

/usr/local/lib/python3.10/dist-packages/seaborn/categorical.py in boxplot(data,
    ↪x, y, hue, order, hue_order, orient, color, palette, saturation, width, dodge,
    ↪fliersize, linewidth, whis, ax, **kwargs)
    2229 ):
    2230
-> 2231     plotter = _BoxPlotter(x, y, hue, data, order, hue_order,
    2232                             orient, color, palette, saturation,
    2233                             width, dodge, fliersize, linewidth)

/usr/local/lib/python3.10/dist-packages/seaborn/categorical.py in __init__(self,
    ↪x, y, hue, data, order, hue_order, orient, color, palette, saturation, width,
    ↪dodge, fliersize, linewidth)
    783         width, dodge, fliersize, linewidth):
    784

```

```

--> 785         self.establish_variables(x, y, hue, data, orient, order, hue_order)
    786         self.establish_colors(color, palette, saturation)
    787

/usr/local/lib/python3.10/dist-packages/seaborn/categorical.py in establish_variables(self, x, y, hue, data, orient, order, hue_order, units)
    542
    543         # Figure out the plotting orientation
--> 544         orient = infer_orient(
    545             x, y, orient, require_numeric=self.require_numeric
    546         )

/usr/local/lib/python3.10/dist-packages/seaborn/_oldcore.py in infer_orient(x, y, orient, require_numeric)
    1592         warnings.warn(single_var_warning.format("Horizontal", "y"))
    1593         if require_numeric and y_type != "numeric":
-> 1594             raise TypeError(nonnumeric_dv_error.format("Vertical", "y"))
    1595         return "v"
    1596

TypeError: Vertical orientation requires numeric `y` variable.

```

```
[ ]:
```

6. Insights based on Non-Graphical and Visual Analysis (10 Points)

6.1 Comments on the range of attributes

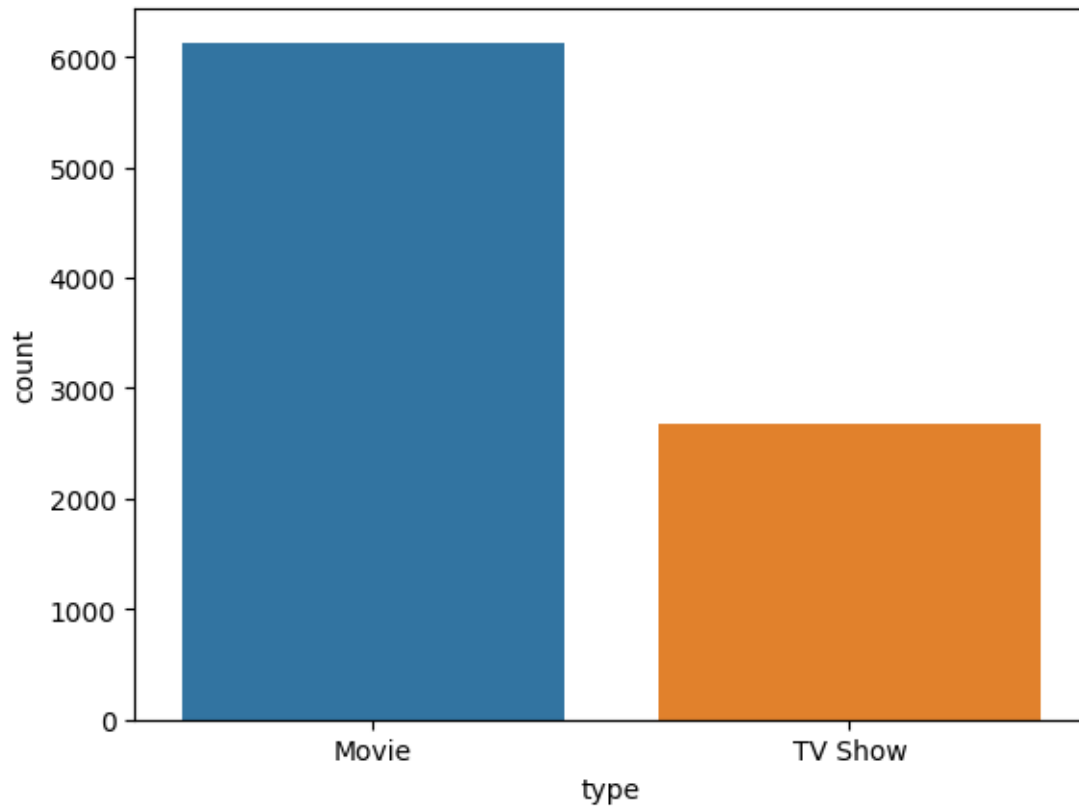
6.2 Comments on the distribution of the variables and relationship between them

6.3 Comments for each univariate and bivariate plot

```
[ ]: # 6.1 Comments on the range of attributes
```

#Show ID: Observation and Insight: it is a unique ID. even if any rows or values are with the other columns, we can find them out by using this unique ID.

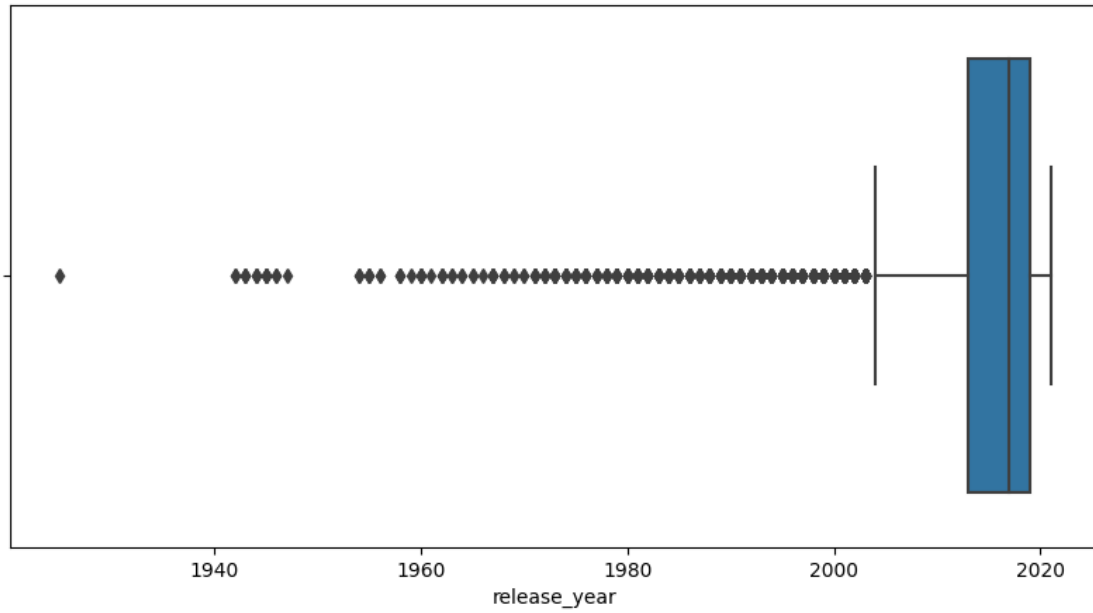
```
[ ]: sns.countplot(data = df, x = 'type')
plt.show()
```



1 Type:

Observation and Insight: The Netflix data has 50%+ higher movie counts when compared to the TV shows. it seems adding more TV shows helps to bring the balance between the both.

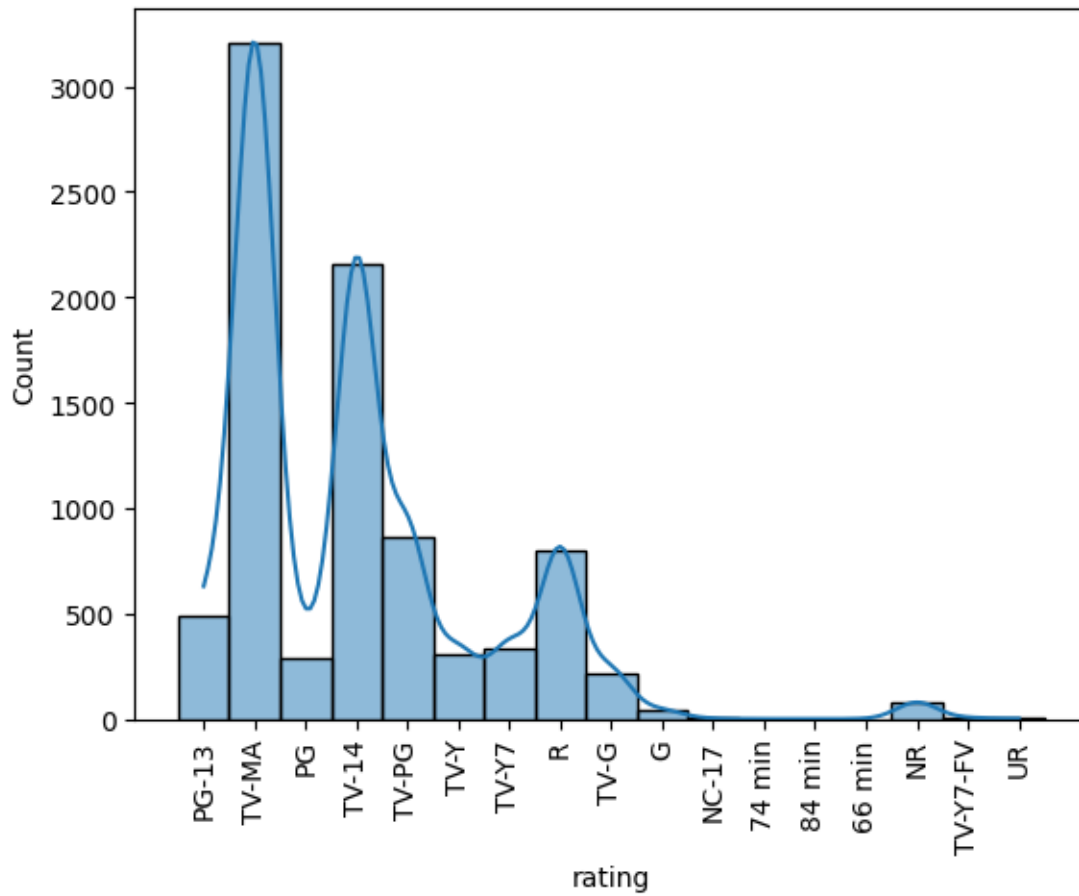
```
[ ]: plt.figure(figsize= (10,5))  
sns.boxplot(data = df, x = 'release_year')  
plt.show()
```



2 Release Year :

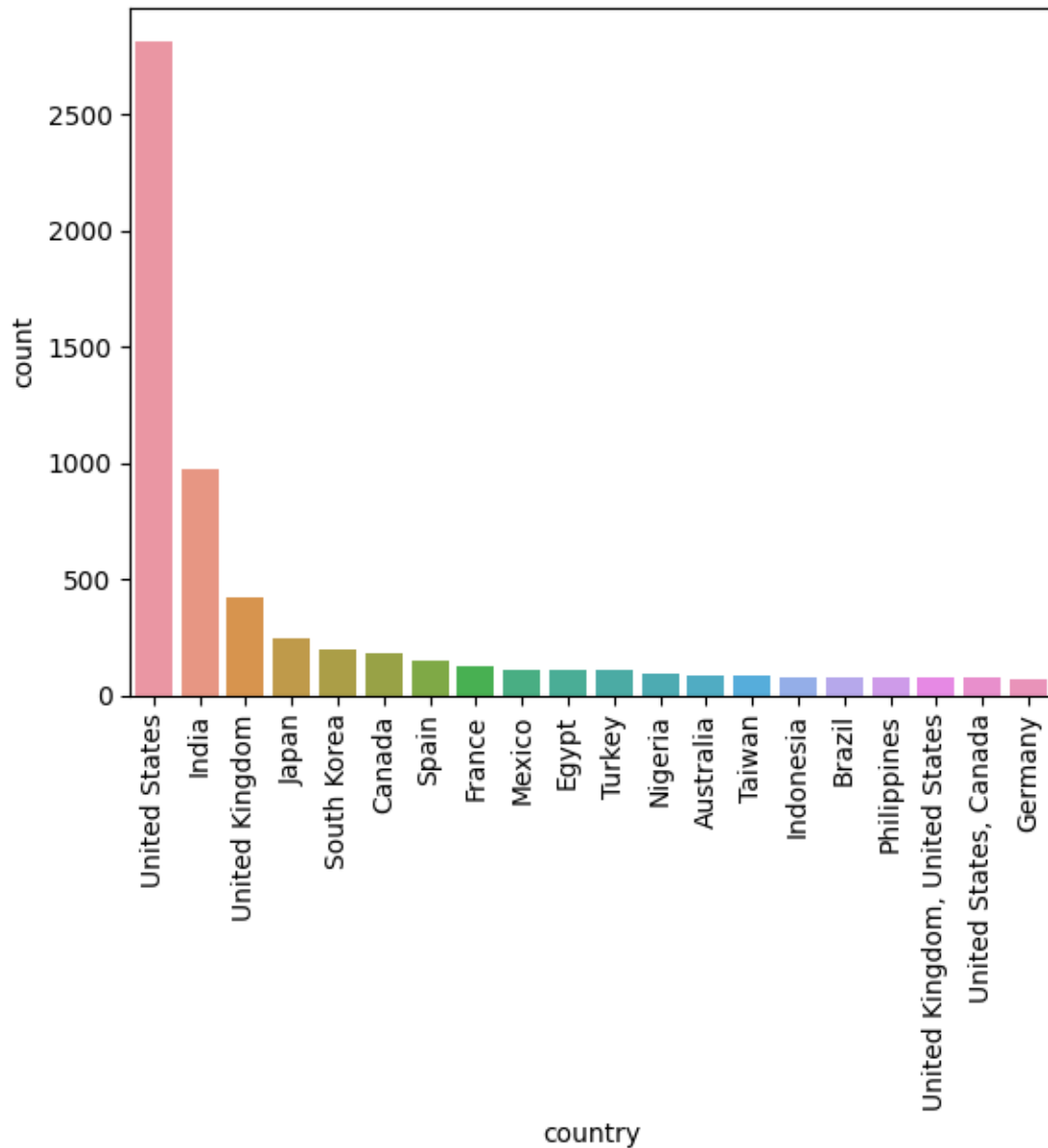
Observation and Insight: The given data clearly shows that the average of the movies and TV shows were released after the year 2015. The consistency in releasing movies and Tv shows started in the year 2000

```
[ ]: sns.histplot(data = df,x = 'rating',kde = True)
plt.xticks(rotation = 90)
plt.show()
```



#Ratings : Observation and Insight: From the given data , the top 3 ratings are ‘TV-MA’, ‘TV-14’ and ‘TV-PG’.

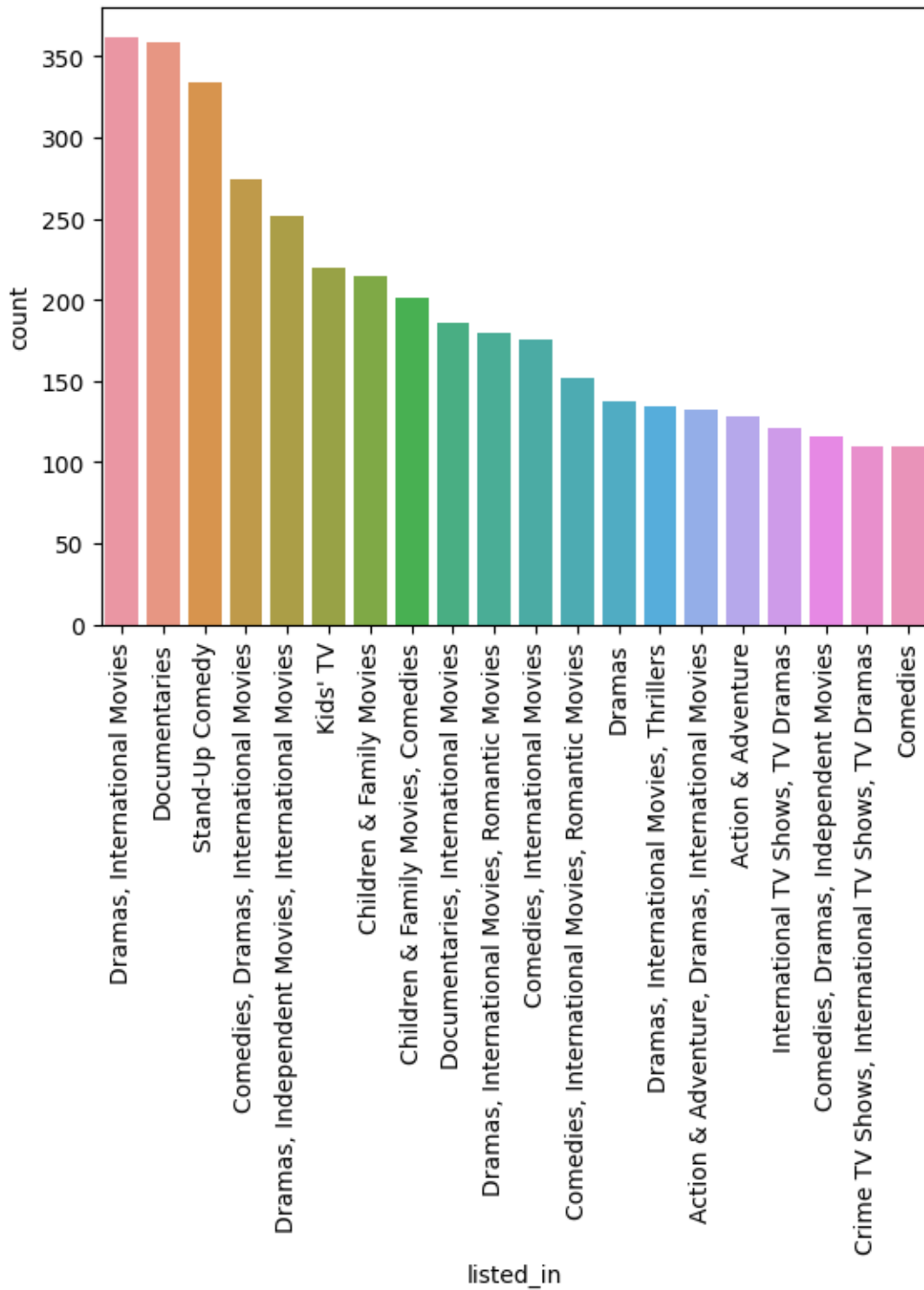
```
[ ]: sns.countplot(data = df,x = 'country',order = df['country'].value_counts().
      ↪index[:20])
plt.xticks(rotation = 90)
plt.show()
```

3 Country :

Observation and Insight: From the given data, the most released TV shows and movies are from the United States and the 2nd highest is the INDIA.

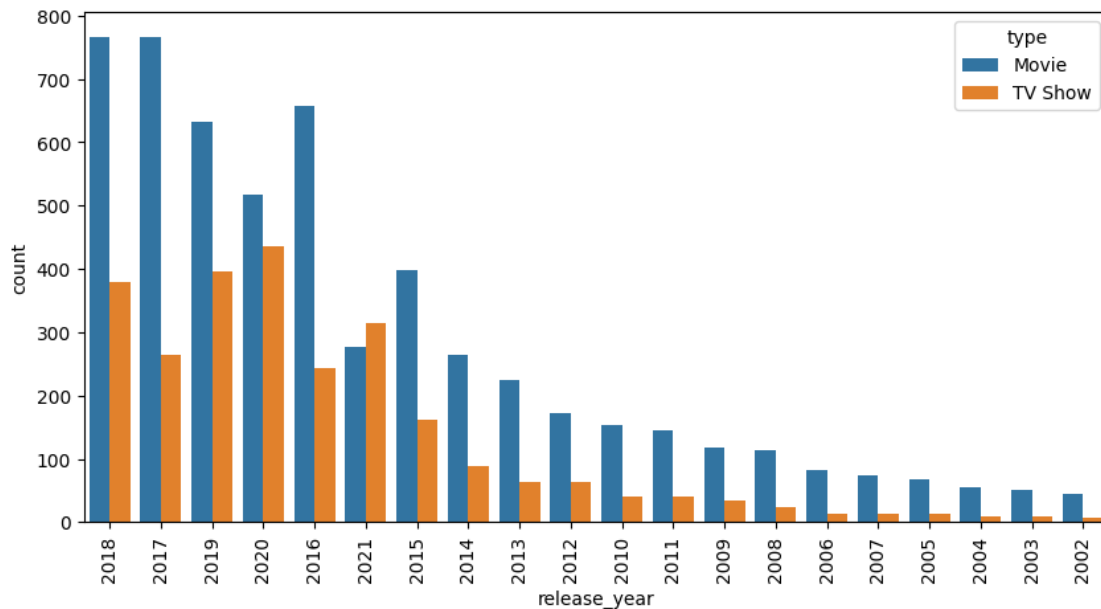
```
[ ]: sns.countplot(data = df,x = 'listed_in',order = df['listed_in'].value_counts().
      ↪index[:20])
plt.xticks(rotation = 90)
plt.show()
```



#Listed_in : Observation and Insight: From the given data, the top 2 categories of genres from the movies and TV shows are 'Dramas, international movies', 'Documentaries' and 'Standup comedies'

```
[ ]: # 6.2 Comments on the distribution of the variables and relationship between
      ↪ them
```

```
[ ]: # Release Year and Type:
plt.figure(figsize=(10, 5))
sns.countplot(x='release_year', hue='type', data=df, order = df['release_year'].
      ↪ value_counts().index[:20])
plt.xticks(rotation = 90)
plt.show()
```

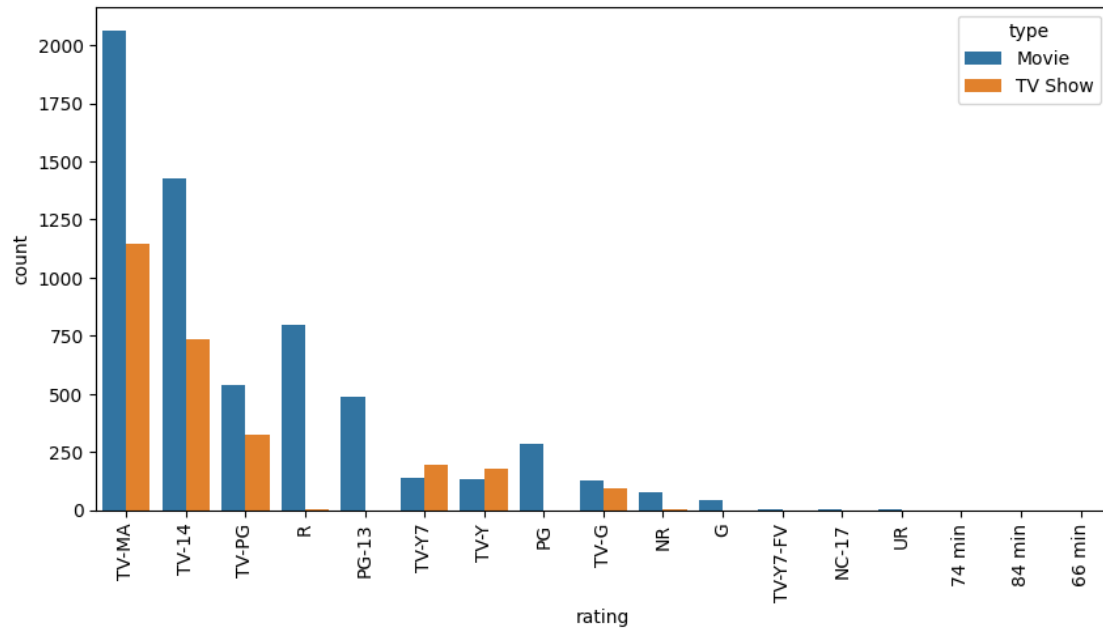


4 Release Year and Type:

Insights:

- Movies and TV shows started to increase in the count from the year 2002. Till 2014 TV shows were not popular when compared to the movies.
- In the year 2018, the total number of movies and TV shows released was very high when compared to the other years. Especially the movies from the year 2017 and 2018 are comparatively high when compared to other years.
- The No. of movies and TV shows gradually started to decrease from the year 2017 to 2021.

```
[ ]: # Rating and Type:
plt.figure(figsize=(10, 5))
sns.countplot(x='rating', hue='type', data=df, order = df['rating'].
      ↪ value_counts().index[:20])
plt.xticks(rotation = 90)
plt.show()
```

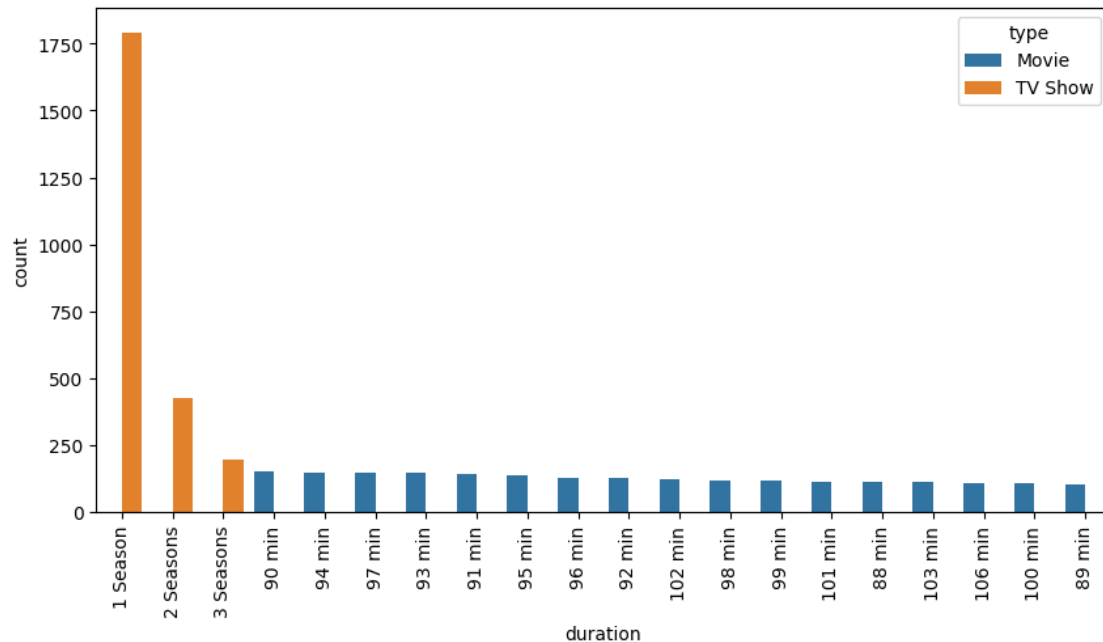


5 Rating and Type:

Insights:

- The top 3 ratings for the Movies and TV shows are TV-MA, TV-14 and TV-PG.
- Ratings for the Movies are comparatively high when compared to the TV shows.

```
[ ]: # Duration and Type:
plt.figure(figsize=(10, 5))
sns.countplot(x='duration', hue='type', data=df, order = df['duration'].
    ↳value_counts().index[:20])
plt.xticks(rotation = 90)
plt.show()
```

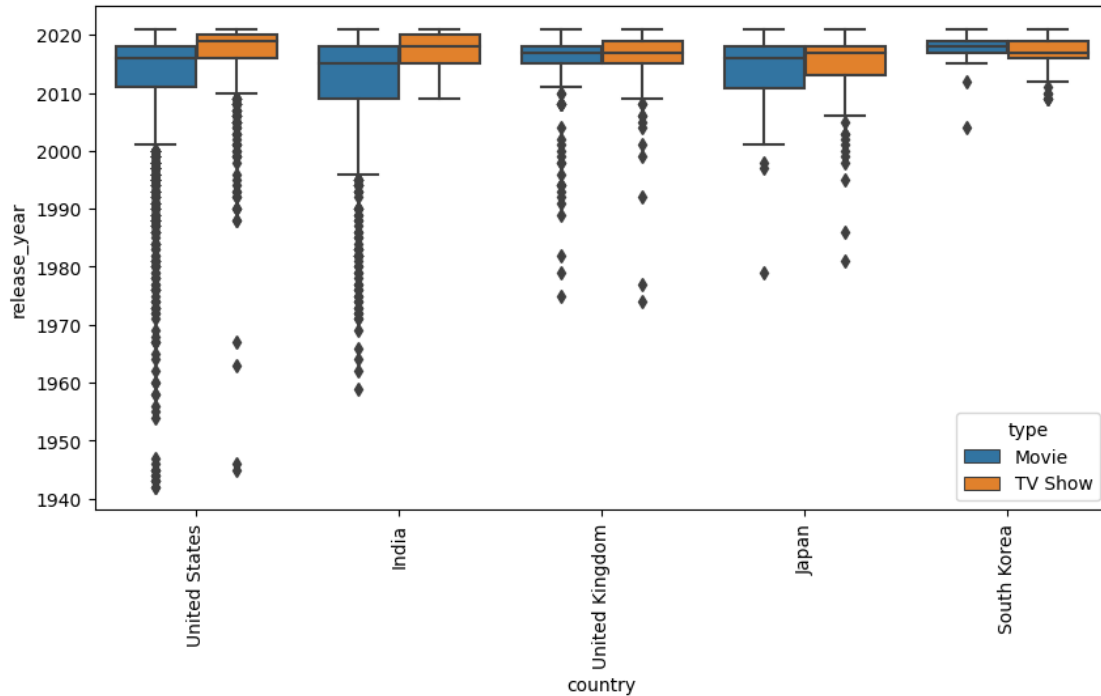


6 Duration and Type:

Insights:

- The duration of 1 season is comparatively high when compared to the other durations.
- TV shows are dominated in this plot, and the Top 3 count rankings are TV shows (1 season, 2 season and 3 season)

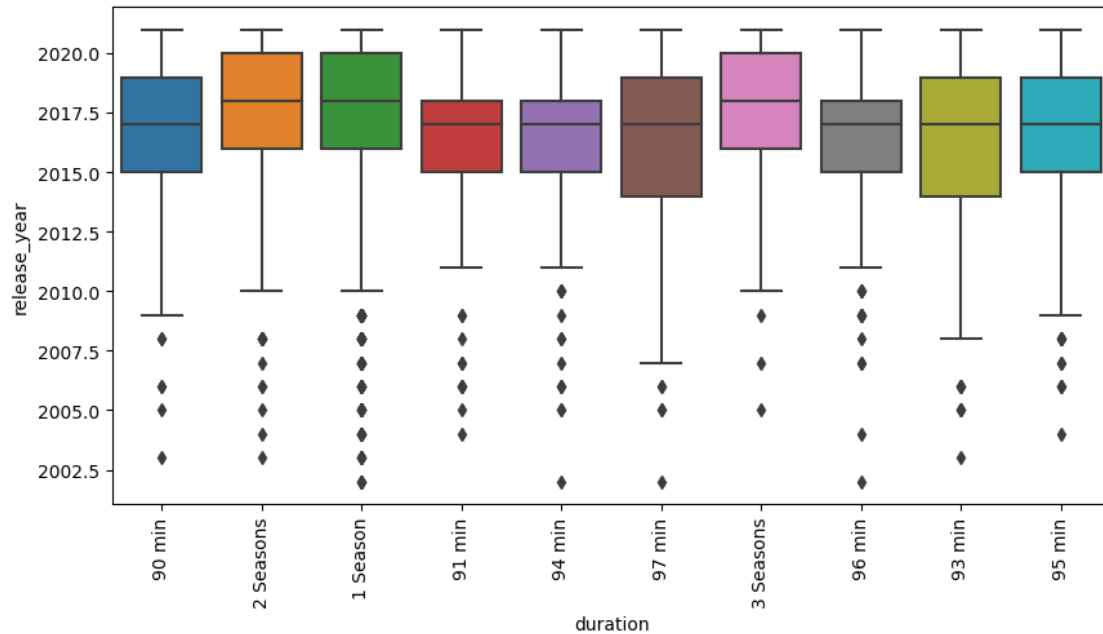
```
[ ]: # Country and Year:
plt.figure(figsize =(10,5))
sns.boxplot(data = df, x ='country',y = 'release_year',hue= 'type',order =_
↳df['country'].value_counts().index[:5])
plt.xticks(rotation = 90)
plt.show()
```



7 Country and Year:

Insights: - The United States is very popular and started movie and TV show production in the year 1940. - India is the second-largest in production. - Comparatively, movies started releasing before TV shows. - TV shows start to become famous after 2017 in both the US and India.

```
[ ]: # Duration and Releasing Year :
duration_counts = df['duration'].value_counts().index[:10]
release_year_counts = df['release_year'].value_counts().index[:20]
filtered_df = df[df['duration'].isin(duration_counts)& df['release_year'].
    ↳isin(release_year_counts)]
plt.figure(figsize=(10, 5))
sns.boxplot(data=filtered_df, x='duration', y='release_year')
plt.xticks(rotation=90)
plt.show()
```



8 Duration and Releasing Year :

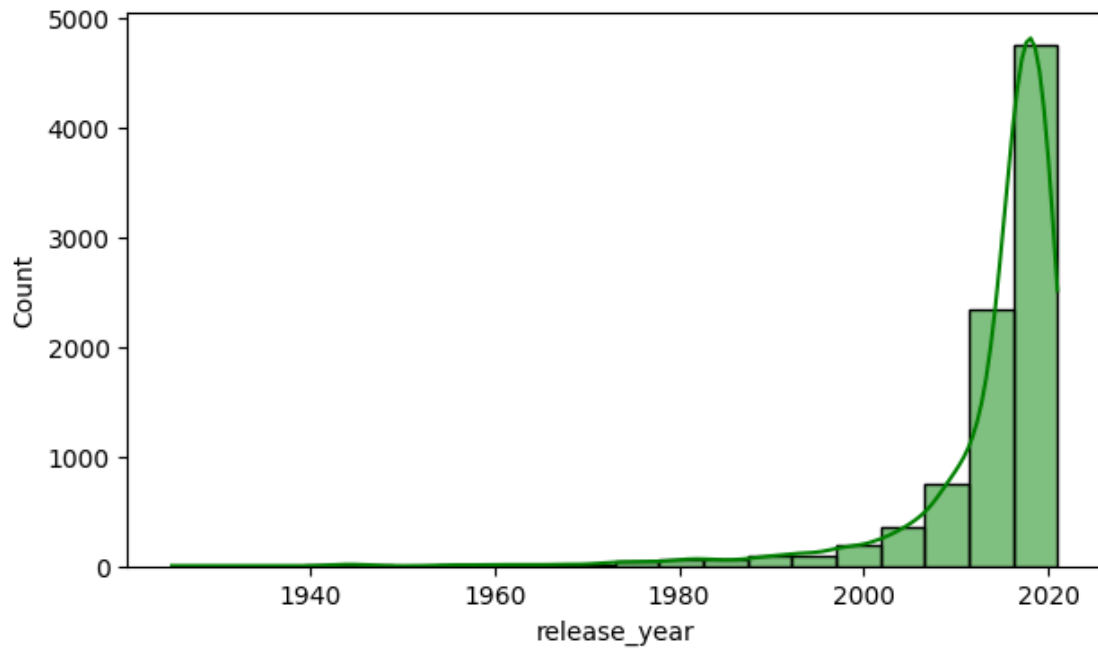
Insights:

- 1 season, 2 seasons, and 3 seasons are comparatively high when compared to all other durations.
- The minimum and maximum for 1 season are 2011 and 2021. The average is 2018.
- The average range for all the durations is released between the years 2016 and 2018.

```
[ ]: # 6.3 Comments for each univariate and bivariate plot
```

```
[ ]: # Univariate :
```

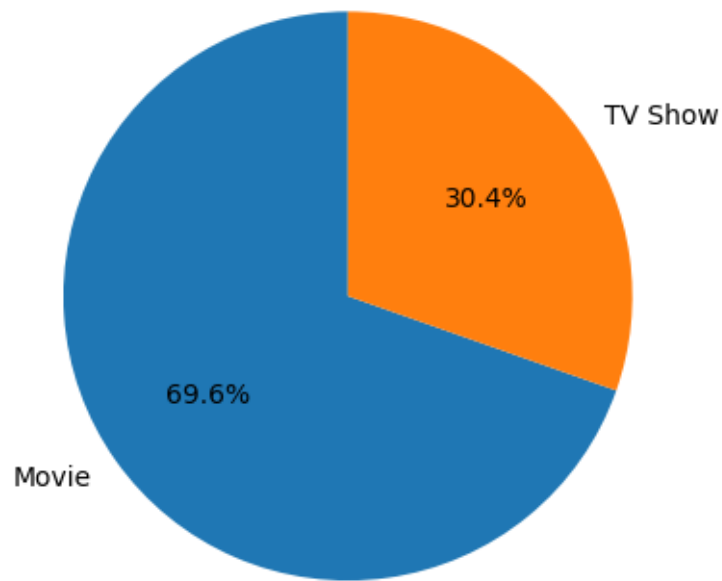
```
# Histogram of Release_year:
plt.figure(figsize = (7,4))
sns.histplot(data = df,x = 'release_year',bins =20,color = 'green',kde = True)
plt.show()
```



Comments: - Most entries are concentrated in recent years (between 2000 to 2020). - The maximum count of entries happened in the year 2018.

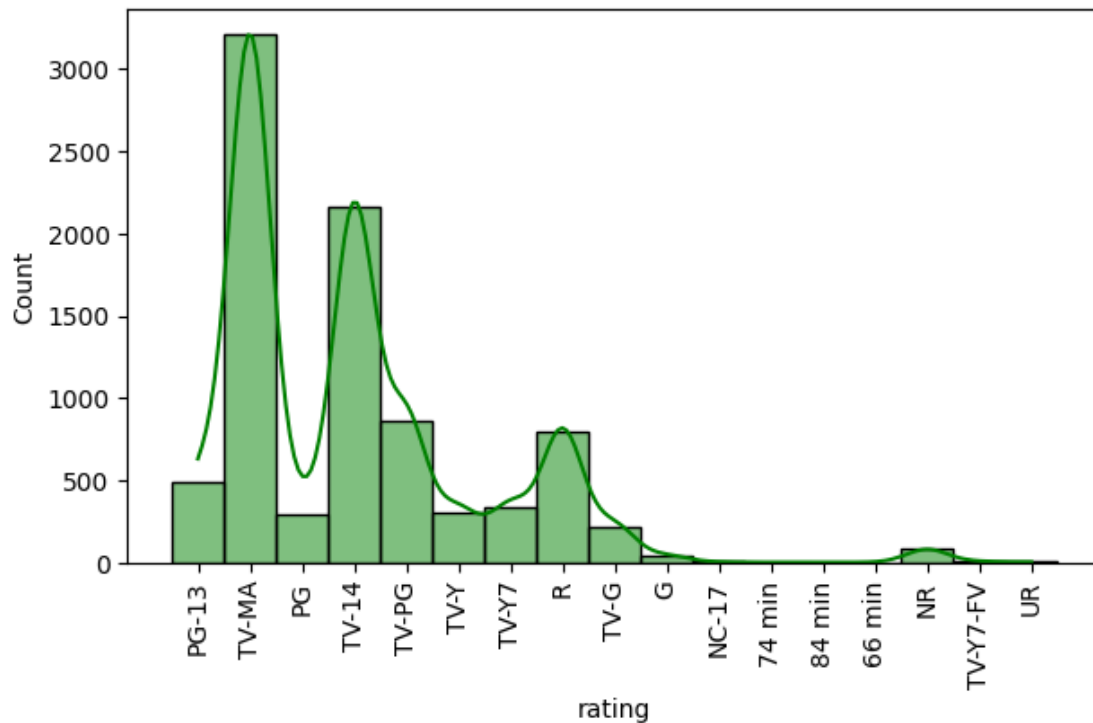
```
[ ]: # Pie Chart of Type:
type_counts = df['type'].value_counts()
plt.pie(type_counts, labels=type_counts.index, autopct='%1.1f%%', startangle=90)
plt.title('Distribution of Types')
plt.show()
```


Distribution of Types



Comments - In the distribution of types, 70% of types are Movies, and the balance 30% are TV shows - This shows movies are very popular from the past trends to the recent trends.

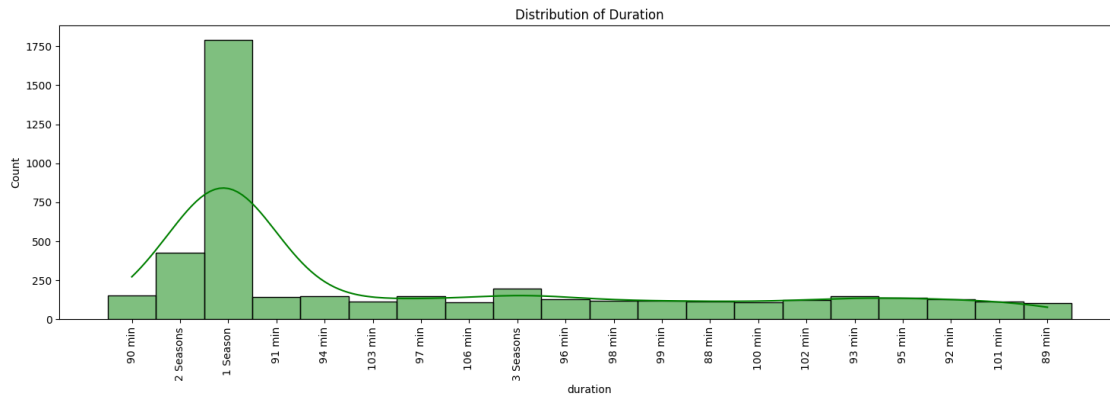
```
[ ]: # Bar Plot of Rating
plt.figure(figsize = (7,4))
sns.histplot(data = df,x = 'rating',bins =20,color = 'green',kde =True)
plt.xticks(rotation = 90)
plt.show()
```



Comments: - The top 3 ratings for the Movies and TV shows are TV-MA, TV-14 and TV-PG. - TV-MA seems to be the more common rating on Netflix.

```
[ ]: # Hist Plot of Duration:
duration = df['duration'].value_counts().index[:20]
df_duration = df[df['duration'].isin(duration)]

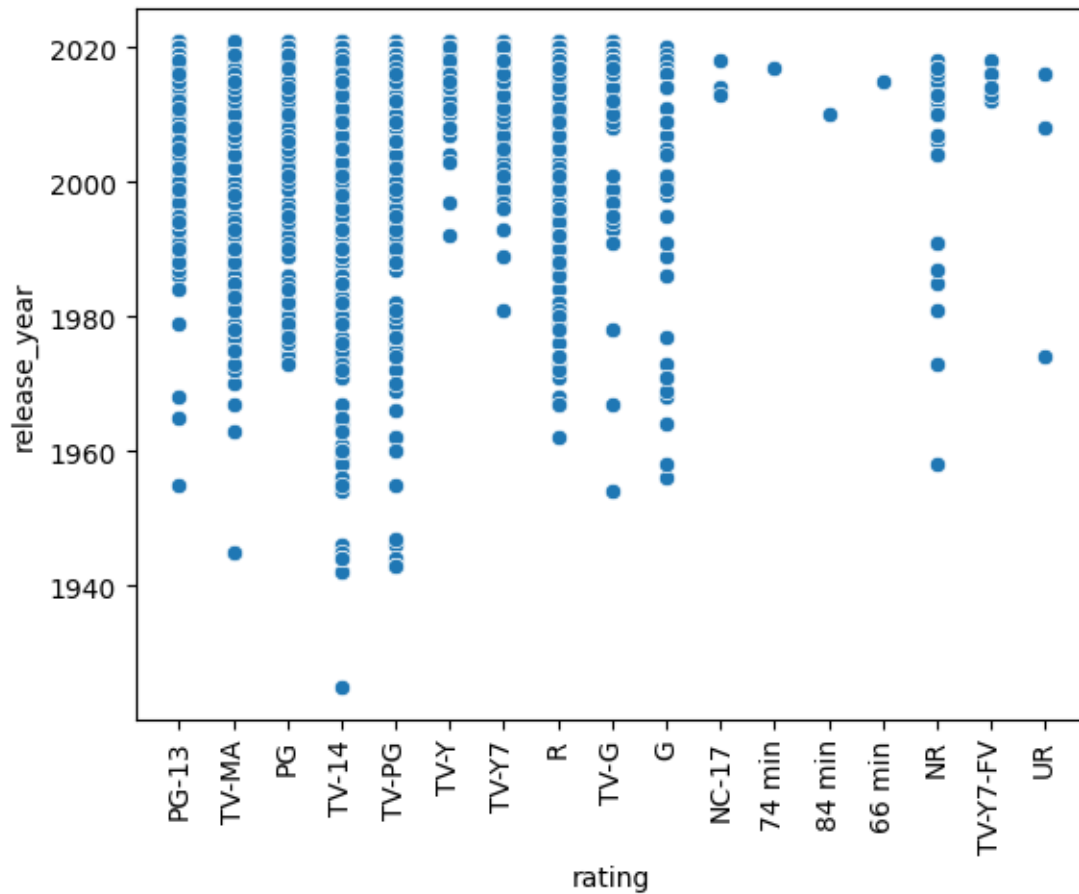
plt.figure(figsize =(18,5))
sns.histplot(data = df_duration ,x='duration',bins = 20,kde = True,color='green')
plt.title('Distribution of Duration')
plt.xticks(rotation = 90)
plt.show()
```



Comments: - 1 season seems to be very popular on Netflix and following that 2 seasons and 3 seasons come under the popular list.

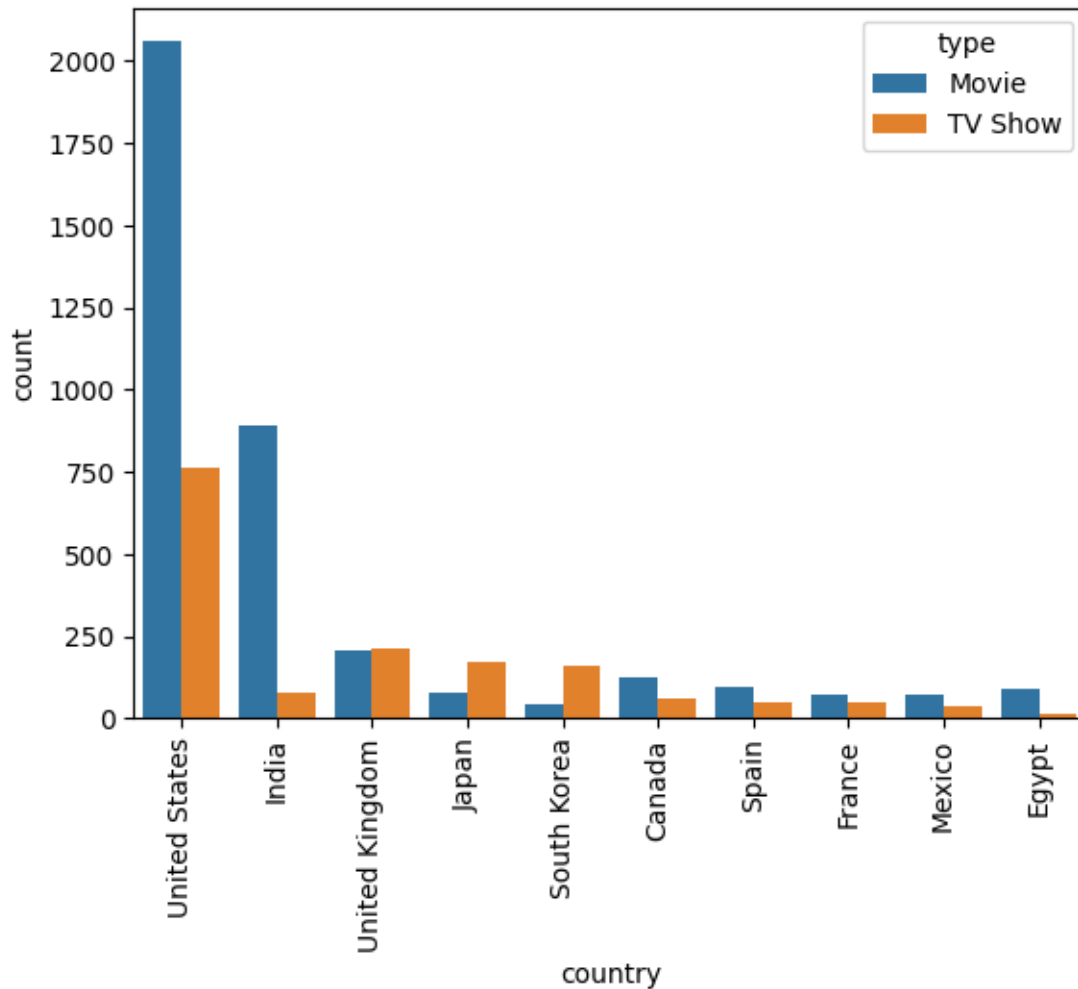
```
[ ]: # bivariate
```

```
[ ]: # Scatter Plot of Release_year vs Rating:
sns.scatterplot(data = df,x = 'rating',y='release_year')
plt.xticks(rotation = 90)
plt.show()
```



Comments: - The top 3 ratings for the Movies and TV shows are TV-MA, TV-14 and TV-PG. - TV-MA seems to be the more common rating on Netflix. - From 2000 onwards, the ratings seem to be more concentrated and consistencies are maintained by PG-13, TV-MA, PG, TV-14, TV-PG, TV-Y7 and R.

```
[ ]: # Count Plot of Type vs Country:
sns.countplot(data = df,x = 'country',hue='type',order = df['country'].
    ↳value_counts().index[:10])
plt.xticks(rotation = 90)
plt.show()
```

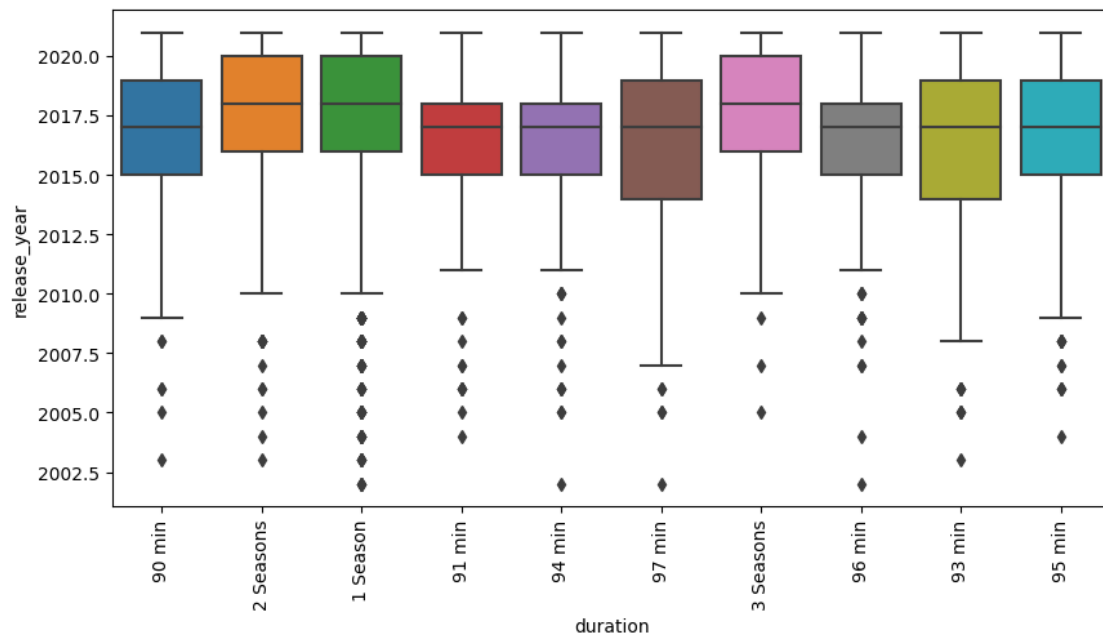


Comments:

- From the given data, the most released TV shows and movies are from the United States, and the second highest in INDIA.
- India and the US, provide the most priority to the movies when compared to TV shows.
- Japan and South Korea seem to be more popular for TV shows than movies.
- The UK has almost the same weightage for both movies and TV shows

```
[ ]: # Box plot of Duration Vs Releasing Year :
duration_counts = df['duration'].value_counts().index[:10]
release_year_counts = df['release_year'].value_counts().index[:20]
filtered_df = df[df['duration'].isin(duration_counts)& df['release_year'].
    ↳isin(release_year_counts)]
plt.figure(figsize=(10, 5))
sns.boxplot(data=filtered_df, x='duration', y='release_year')
plt.xticks(rotation=90)
```

```
plt.show()
```

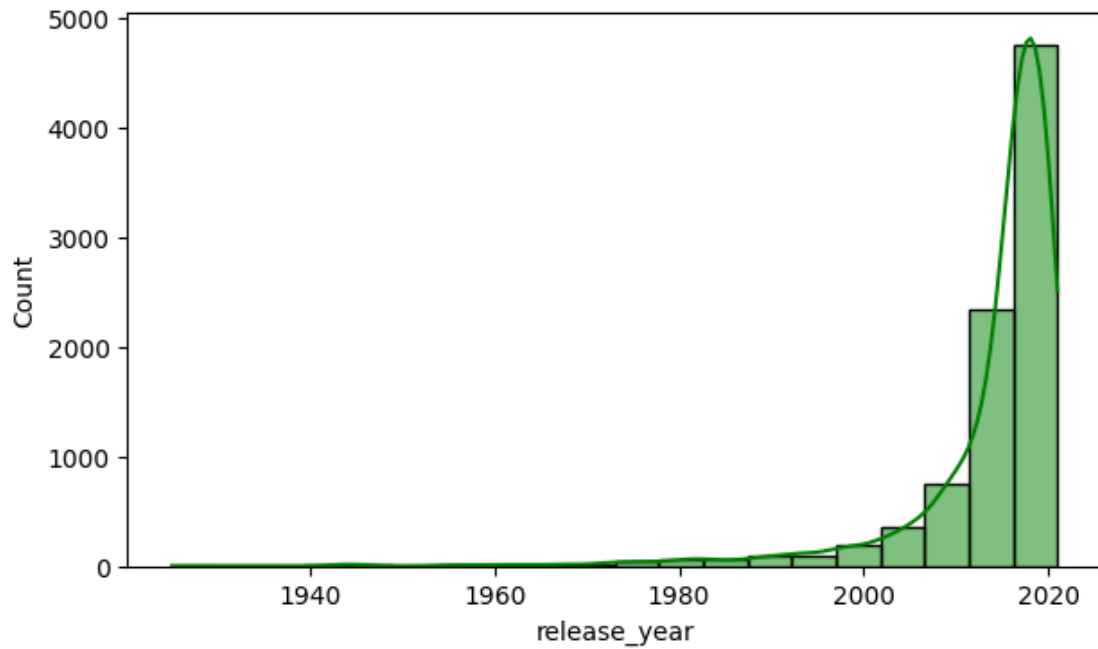


Comments: - 1 season, 2 seasons, and 3 seasons are comparatively high when compared to all other durations. - The minimum and maximum for 1 season are 2011 and 2021. The average is 2018. - The average range for all the durations is released between the years 2016 and 2018.

```
[ ]:
```

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

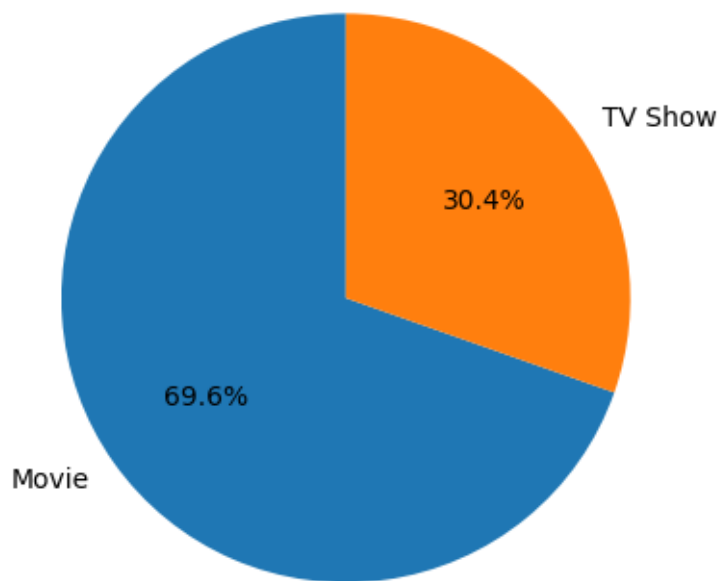
```
[ ]: # Histogram of Release_year:
plt.figure(figsize = (7,4))
sns.histplot(data = df,x = 'release_year',bins =20,color = 'green',kde = True)
plt.show()
```



Inference: - increasing the production and release of TV shows and movies is directly proportional to the increase in the business of Netflix.

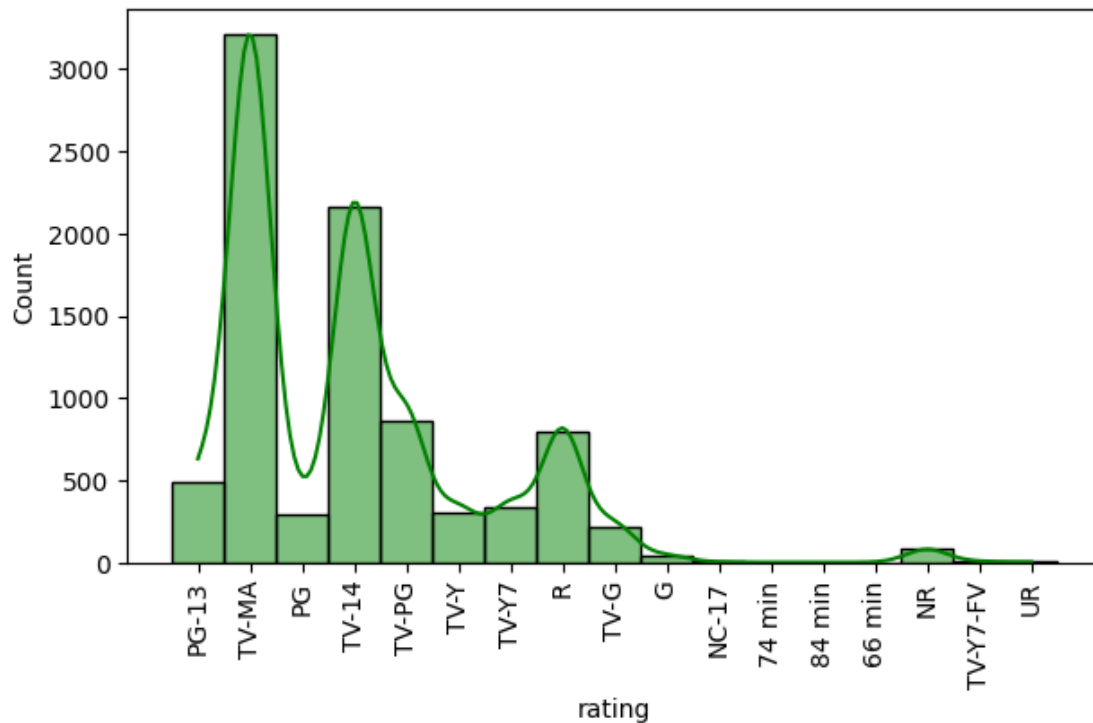
```
[ ]: # Pie Chart of Type:
type_counts = df['type'].value_counts()
plt.pie(type_counts, labels=type_counts.index, autopct='%1.1f%%', startangle=90)
plt.title('Distribution of Types')
plt.show()
```

Distribution of Types



Inference: - It seems like the business is mostly reliant on movies, with almost 70% of the content being distributed in that format. - Consistent production of movies is likely critical to keeping the business afloat. Additionally, there seems to be some use of a pie chart to identify and target a specific audience for their content.

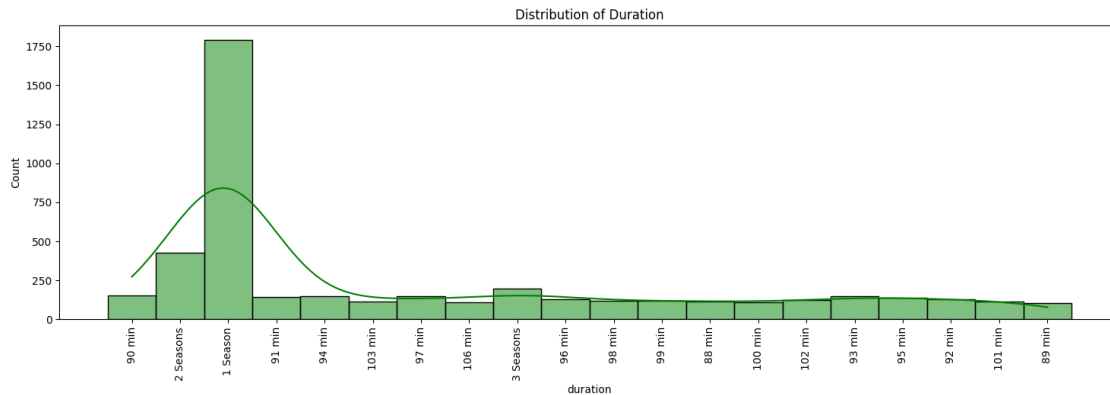
```
[ ]: # Bar Plot of Rating
plt.figure(figsize = (7,4))
sns.histplot(data = df,x = 'rating',bins =20,color = 'green',kde =True)
plt.xticks(rotation = 90)
plt.show()
```

Inference: - TV-MA is the most popular rating when compared to all others. - The content and types that come under the TV-Ma ratings help to identify the selective audience and increase the production of it helps to increase the business for the Netflix.

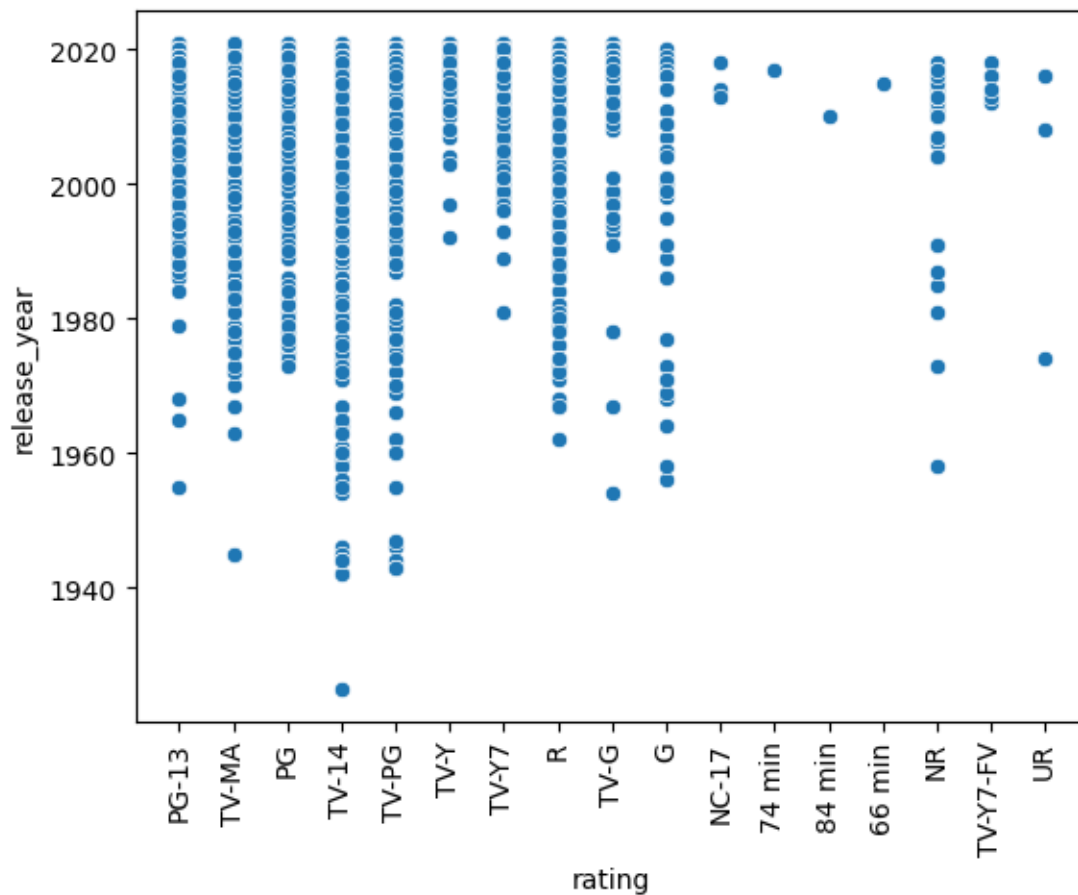
```
[ ]: # Hist Plot of Duration:
duration = df['duration'].value_counts().index[:20]
df_duration = df[df['duration'].isin(duration)]

plt.figure(figsize =(18,5))
sns.histplot(data = df_duration ,x='duration',bins = 20,kde = True,color='green')
plt.title('Distribution of Duration')
plt.xticks(rotation = 90)
plt.show()
```



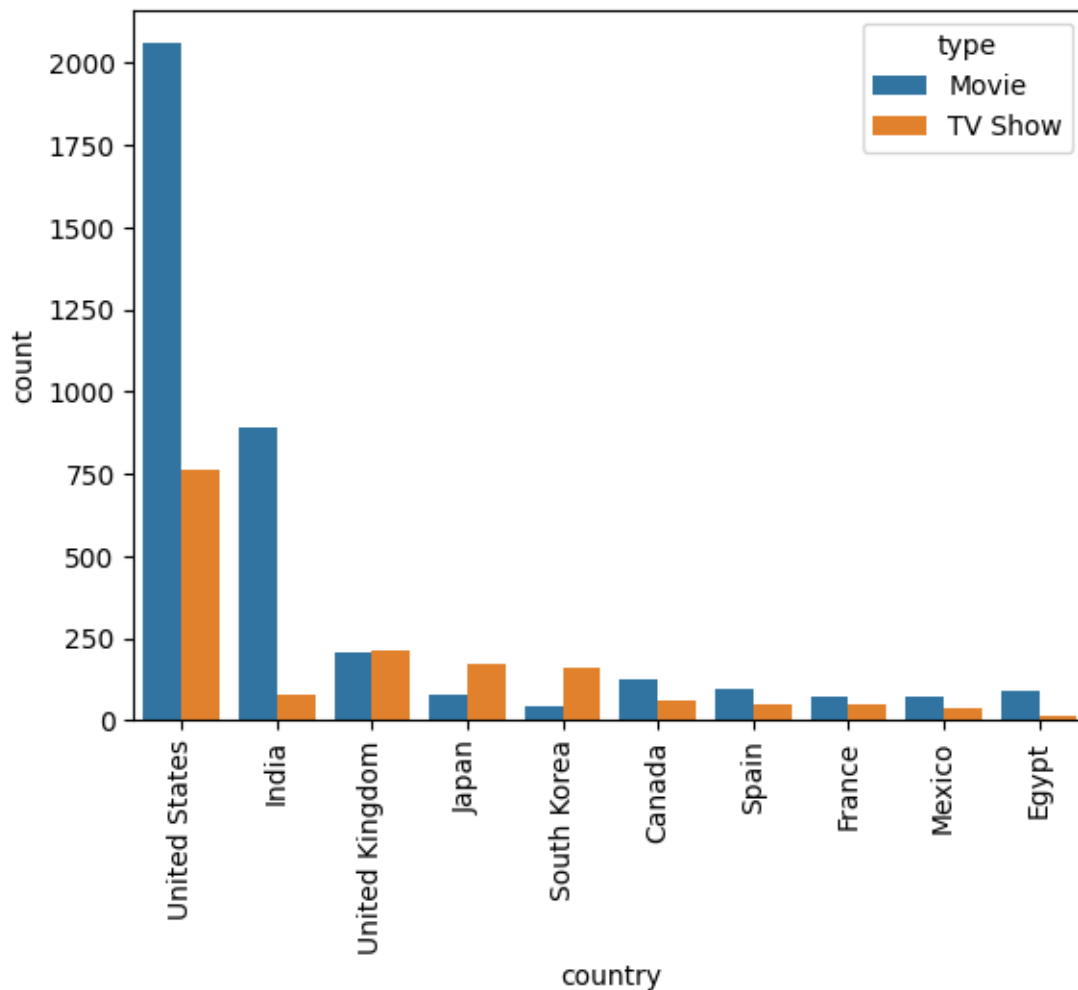
Inference: - 1 season TV shows are the most preferred and highly related to the counts. - Users are happily satisfied with the 1-season TV shows when compared to the Movies and other TV shows.

```
[ ]: # Scatter Plot of Release_year vs Rating:
sns.scatterplot(data = df,x='rating',y='release_year')
plt.xticks(rotation = 90)
plt.show()
```



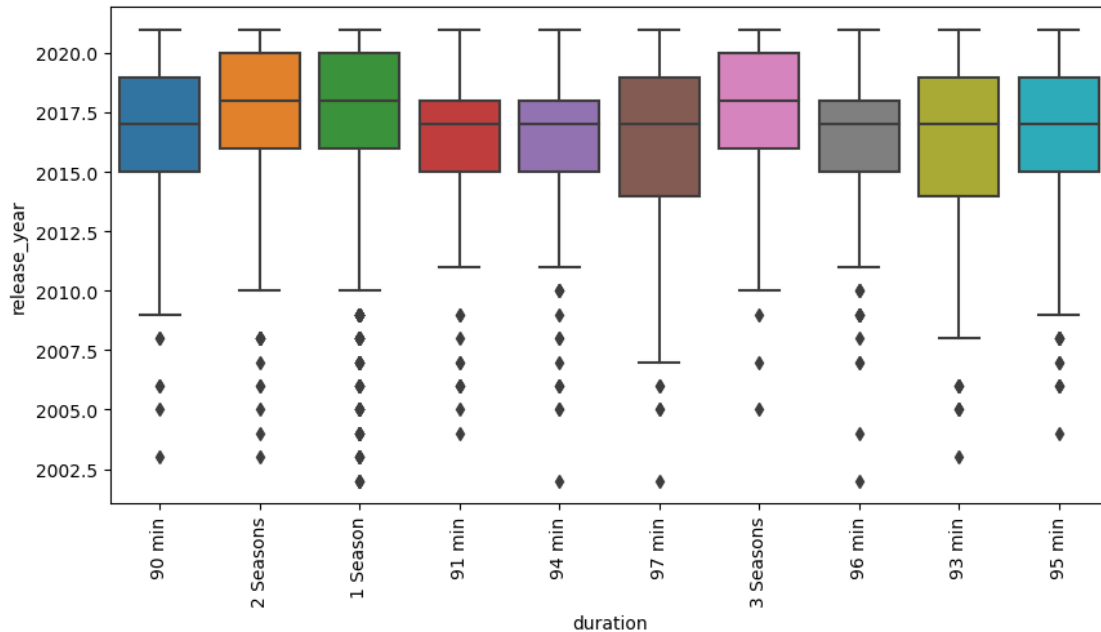
Inference: - Ratings seem to be more concentrated year on year. - Increasing the ratings helps to make the user to be more engaged. - The genres that come under TV-MA have to increase in production since it has very good popularity and ratings.

```
[ ]: # Count Plot of Type vs Country:
sns.countplot(data = df,x ='country',hue='type',order = df['country'].
    ↪value_counts().index[:10])
plt.xticks(rotation = 90)
plt.show()
```



Inference: - Comparatively, movies have the highest count in both the US and INDIA. - Increasing the production of movies in all other countries helps to increase the new users that are directly going to increase the business of Netflix

```
[ ]: # Box plot of Duration Vs Releasing Year :
duration_counts = df['duration'].value_counts().index[:10]
release_year_counts = df['release_year'].value_counts().index[:20]
filtered_df = df[df['duration'].isin(duration_counts)& df['release_year'].
    ↳isin(release_year_counts)]
plt.figure(figsize=(10, 5))
sns.boxplot(data=filtered_df, x='duration', y='release_year')
plt.xticks(rotation=90)
plt.show()
```



Inference: - High concentration of 1 season clearly shows that the users are proactively watching it and the business profit value is higher comparatively with the other duration over year on year.

```
[ ]:
```

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

- Ensure a balanced strategy between movies and TV shows. currently movies seems to be higher and the balance is not there.
- Emphasize more content acquisition and production in genres 'Dramas, international movies'. this might help to get higher audience engagement.
- Increase investment in content from countries like US and INDIA with high production contributions. This might incrate the profit margin rate
- Stay updated on release year trends like the TV shows with 1 season seems to be most popular in the current trend and adjust the content acquisition accordingly.
- Promote content featuring successful directors and cast members

- Craft engaging and descriptive content summaries.
- Use user preferences for content duration to refine recommendation algorithms.
- Analyze the best times to add content based on historical data and user activity patterns.
- Establish a feedback loop for user preferences and incorporate it into content decisions.
- Explore collaborations with successful directors, actors, and production houses.

Implementing these recommendations can enhance user satisfaction, broaden audience reach, and contribute to the continued success of the Netflix.