Mindset

Evaluation will be kept lenient, so make sure you attempt this case study. It is understandable that you might struggle with getting started on this. Just brainstorm, discuss with peers, or get help from TAs. There is no right or wrong answer. We have to get used to dealing with uncertainty in business. This is exactly the skill we want to develop.

About NETFLIX

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

Business Problem

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

Dataset

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

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

In [1]:

```python
# Importing libs
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans, AffinityPropagation
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")
import plotly as py
import plotly.graph_objs as go
import os
py.offline.init_notebook_mode(connected = True)
#print(os.listdir("../input"))
import datetime as dt
import missingno as msno
plt.rcParams['figure.dpi'] = 140

#df = pd.read_csv("/content/sample_data/netflix.csv")
```

```
---------------------------------------------------------------------------
-
ModuleNotFoundError                       Traceback (most recent call las
t)
Cell In[1], line 2
      1 # Importing libs
----> 2 import pandas as pd
      3 import numpy as np
      4 from sklearn.preprocessing import StandardScaler

ModuleNotFoundError: No module named 'pandas'
```

1. Defining Problem Statement and Analysing basic metrics (10 Points)

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

In [2]:

```
df = pd.read_csv("/content/sample_data/netflix.csv")
df.head(3)
```

```
---------------------------------------------------------------------------
-
NameError                                 Traceback (most recent call las
t)
Cell In[2], line 1
----> 1 df = pd.read_csv("/content/sample_data/netflix.csv")
      2 df.head(3)

NameError: name 'pd' is not defined
```

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 (10 Points)

Data Types, descriptive analysis for the features, missing value analysis and more is covered in the next cells

In [3]:

```
df.isnull().sum()
```

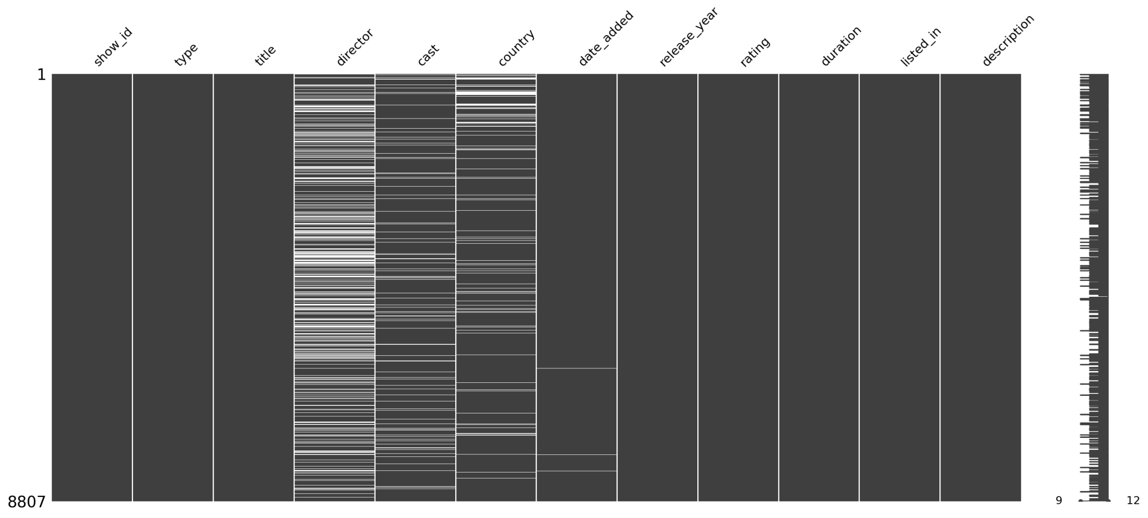
Out[3]:

```
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
```

Missing Values Graphical

In [4]:

```
import missingno as miss
import matplotlib.pyplot as plt

%matplotlib inline
miss.matrix(df)
plt.show()
```



In [5]:

```
print("***Dataframe structure and the data types of each feature***")
print('\n' * 2)
print(df.dtypes)
print('\n' * 5)
print("***Descriptive Analysis for all columns of the dataframe***")
print('\n' * 2)
print(df.describe(include='all'))
```

```
***Dataframe structure and the data types of each feature***



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





***Descriptive Analysis for all columns of the dataframe***



       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
```

Observations:

There are 8807 entries in the dataset.

There are 2 integer types and rest of them are string objects.

show_id and release_year are mentioned in integers. Later, we will create year_added and month_added for better analysis

date_added field is of type object here and need to be converted to Date-Time appropriate field

release_year is integer field and for now can be used as it is for the analysis

In [6]:

```python
# Analyzing Missing/Null/NaN data

for i in df.columns:
    null_rate = df[i].isna().sum() / len(df) * 100
    if null_rate > 0 :
        print("{} null rate: {}%".format(i,round(null_rate,2)))
```

```
director null rate: 29.91%
cast null rate: 9.37%
country null rate: 9.44%
date_added null rate: 0.11%
rating null rate: 0.05%
duration null rate: 0.03%
```

Dealing with the missing data This is always scenario dependant and here we would:

1. Country: replace blank countries with the mode (most common) country for simplicity, though other approaches are available
2. Director: Analysis around the directors would be interesting hence even with such High NULL rate lets retain this feature
3. Cast: I want to keep cast as it could be interesting to look at a certain cast's films

In [7]:

```
#  pre-processing  NULL valuees

df['country'] = df['country'].fillna(df['country'].mode()[0])  # Replacing the missing c
df['cast'].replace(np.nan, 'Unknown',inplace  = True)
df['director'].replace(np.nan, 'Unknown',inplace  = True)
print("Before dropping NA:")
print(df.shape)
# Drops
df.dropna(inplace=True)
print("After dropping NA:" )
print(df.shape)
# Drop Duplicates
df.drop_duplicates(inplace= True)
print("After removing Duplicates:" )
print(df.shape)

#df_o['date_added'] = df_o['date_added'].fillna(df_o['date_added'].mode()[0])
df['rating'].replace(np.nan, 'Not Available',inplace  = True)
df['duration'].replace(np.nan, '0',inplace  = True)
```

```
Before dropping NA:
(8807, 12)
After dropping NA:
(8790, 12)
After removing Duplicates:
(8790, 12)
```

In [8]:

```
# Validating that no Null values in any feature
df.isnull().sum()
```

Out[8]:

```
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
```

In [9]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 8790 entries, 0 to 8806
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   show_id       8790 non-null   object
 1   type          8790 non-null   object
 2   title         8790 non-null   object
 3   director      8790 non-null   object
 4   cast          8790 non-null   object
 5   country       8790 non-null   object
 6   date_added    8790 non-null   object
 7   release_year  8790 non-null   int64 
 8   rating        8790 non-null   object
 9   duration      8790 non-null   object
 10  listed_in     8790 non-null   object
 11  description   8790 non-null   object
dtypes: int64(1), object(11)
memory usage: 892.7+ KB
```

The date_added feature datatype needs to be modified For further analysis we are segregating the part of the dates and adding them as a additional feature in columns

In [10]:

```
df["date_added"] = pd.to_datetime(df['date_added'])
df['month_added']=df['date_added'].dt.month
df['month_name_added']=df['date_added'].dt.month_name()
df['year_added'] = df['date_added'].dt.year

df.head(3)
```

Out[10]:

	show_id	type	title	director	cast	country	date_added	release_year	ratin
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	Unknown	United States	2021-09-25	2020	PG 1
1	s2	TV Show	Blood & Water	Unknown	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	2021-09-24	2021	TV M
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	United States	2021-09-24	2021	TV M

3. Non-Graphical Analysis: Value counts and unique attributes (10 Points)

In [11]:

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

Out[11]:

	show_id	type	title	director	cast	country	date_added	release_year	rat
count	8790	8790	8790	8790	8790	8790	8790	8790.000000	8
unique	8790	2	8790	4527	7679	748	1713	NaN	
top	s1	Movie	Dick Johnson Is Dead	Unknown	Unknown	United States	2020-01-01 00:00:00	NaN	
freq	1	6126	1	2621	825	3638	110	NaN	3
first	NaN	NaN	NaN	NaN	NaN	NaN	2008-01-01 00:00:00	NaN	N
last	NaN	NaN	NaN	NaN	NaN	NaN	2021-09-25 00:00:00	NaN	N
mean	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2014.183163	N
std	NaN	NaN	NaN	NaN	NaN	NaN	NaN	8.825466	N
min	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1925.000000	N
25%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2013.000000	N
50%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2017.000000	N
75%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2019.000000	N
max	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2021.000000	N

Total Unique contents available in the dataset

In [12]:

```
df['title'].drop_duplicates(keep='last').value_counts().value_counts()[1]
```

Out[12]:

```
8790
```

In [13]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 8790 entries, 0 to 8806
Data columns (total 15 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   show_id           8790 non-null   object
 1   type              8790 non-null   object
 2   title             8790 non-null   object
 3   director          8790 non-null   object
 4   cast              8790 non-null   object
 5   country           8790 non-null   object
 6   date_added        8790 non-null   datetime64[ns]
 7   release_year      8790 non-null   int64
 8   rating            8790 non-null   object
 9   duration          8790 non-null   object
 10  listed_in         8790 non-null   object
 11  description       8790 non-null   object
 12  month_added       8790 non-null   int64
 13  month_name_added  8790 non-null   object
 14  year_added        8790 non-null   int64
dtypes: datetime64[ns](1), int64(3), object(11)
memory usage: 1.1+ MB
```

In [14]:

```
df.isna().sum()
```

Out[14]:

```
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
month_added         0
month_name_added    0
year_added          0
dtype: int64
```

In [15]:

```
column_list=df.columns
for col in column_list:
  print("The value count for " , col , "is "  )
  print()
  print(df[col].value_counts())
  print("****"*7)
  print('\n'*2)
# another way of obtaining the same output
#print(car_df['Gear'].value_counts()
#cy_count = car_df['Cylinder'].value_counts()
```

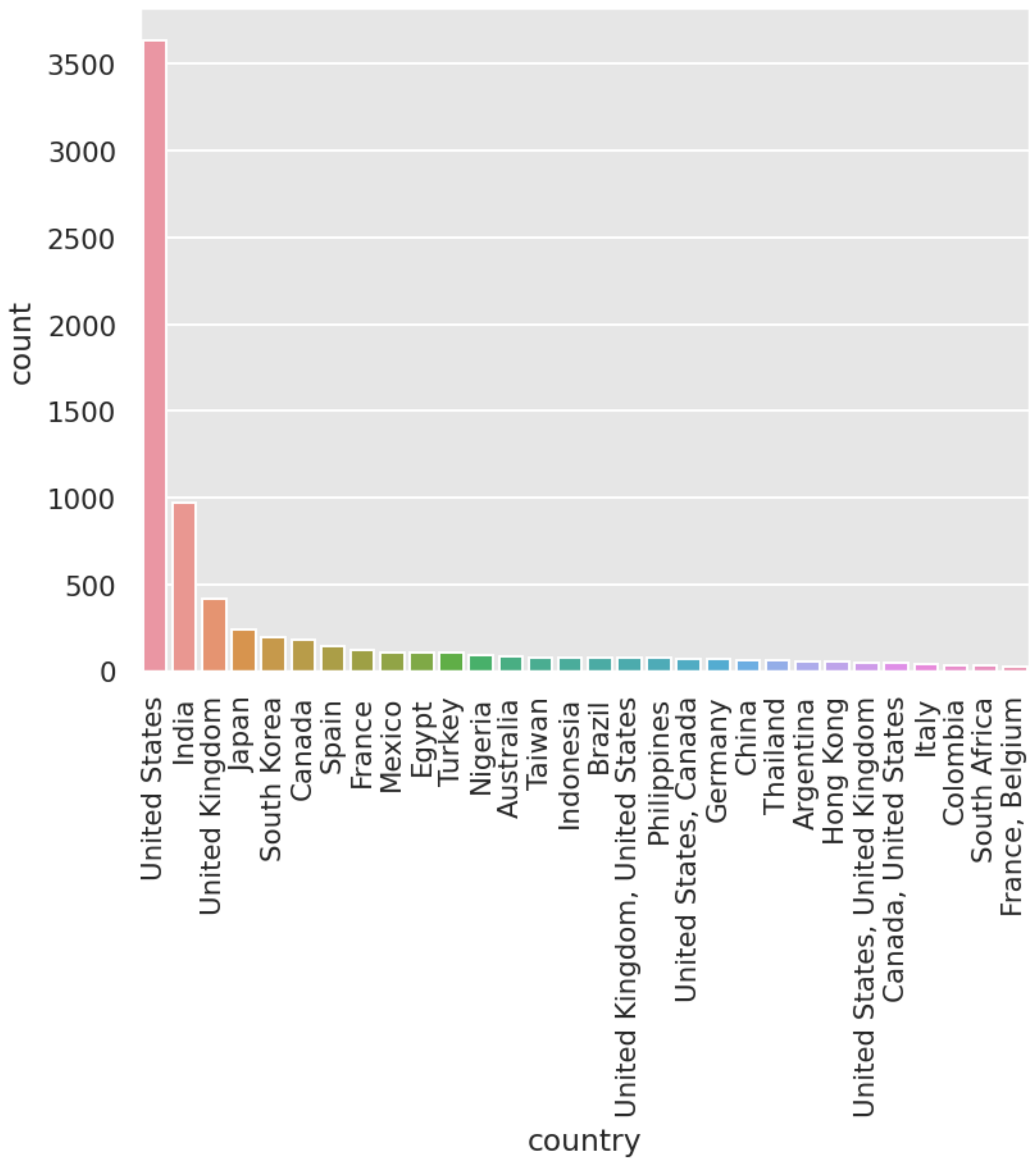
```
The value count for  show_id is 

s1       1
s5867    1
s5861    1
s5862    1
s5863    1
        ..
s2924    1
s2923    1
s2922    1
s2921    1
s8807    1
Name: show_id, Length: 8790, dtype: int64
****************************




The value count for  type is 
```

In [16]:

```
## Country wise distribution of the content available on the platform

group_country_movies = df.groupby('country')['show_id'].count().sort_values(ascending =

countries_list = []
count_list = []
for index, value in group_country_movies.items():
    countries_list.append(index)
    count_list.append(value)

cars = {
    'country': countries_list,
    'count': count_list
}

df4 = pd.DataFrame(cars, columns = ['country', 'count'])


sns.set_style("darkgrid", {"axes.facecolor": ".9"})
# possible styles: whitegrid, dark, white

sns.set_context("notebook")


ax = sns.barplot(x = "country", y = "count", data = df4)
ax.set_xticklabels(ax.get_xticklabels(), rotation = 90)
```

Out[16]:

```
[Text(0, 0, 'United States'),
 Text(1, 0, 'India'),
 Text(2, 0, 'United Kingdom'),
 Text(3, 0, 'Japan'),
 Text(4, 0, 'South Korea'),
 Text(5, 0, 'Canada'),
 Text(6, 0, 'Spain'),
 Text(7, 0, 'France'),
 Text(8, 0, 'Mexico'),
 Text(9, 0, 'Egypt'),
 Text(10, 0, 'Turkey'),
 Text(11, 0, 'Nigeria'),
 Text(12, 0, 'Australia'),
 Text(13, 0, 'Taiwan'),
 Text(14, 0, 'Indonesia'),
 Text(15, 0, 'Brazil'),
 Text(16, 0, 'United Kingdom, United States'),
 Text(17, 0, 'Philippines'),
 Text(18, 0, 'United States, Canada'),
 Text(19, 0, 'Germany'),
 Text(20, 0, 'China'),
 Text(21, 0, 'Thailand'),
 Text(22, 0, 'Argentina'),
 Text(23, 0, 'Hong Kong'),
 Text(24, 0, 'United States, United Kingdom'),
 Text(25, 0, 'Canada, United States'),
 Text(26, 0, 'Italy'),
 Text(27, 0, 'Colombia'),
 Text(28, 0, 'South Africa'),
 Text(29, 0, 'France, Belgium')]
```


count
3500
3000
2500
2000
1500
1000
500
0
United States
India
United Kingdom
Japan
South Korea
Canada
Spain
France
Mexico
Egypt
Turkey
Nigeria
Australia
Taiwan
Indonesia
Brazil
United Kingdom, United States
Philippines
United States, Canada
Germany
China
Thailand
Argentina
Hong Kong
United States, United Kingdom
Canada, United States
Italy
Colombia
South Africa
France, Belgium
country

In [17]:

```
group_country_movies.head(20)
```

Out[17]:

```
country
United States                    3638
India                             972
United Kingdom                    418
Japan                             243
South Korea                       199
Canada                            181
Spain                             145
France                            124
Mexico                            110
Egypt                             106
Turkey                            105
Nigeria                            95
Australia                          85
Taiwan                             81
Indonesia                          79
Brazil                             77
United Kingdom, United States      75
Philippines                        75
United States, Canada              73
Germany                            67
Name: show_id, dtype: int64
```

Observation:

1. United States tops the list when it comes to the overall content added from Netflix. More than 3500 contents added in the United States followed by India.
2. In India, Netflix added almost 1000 titles.
3. UK and Japan takes third and fourth place in Netflix content published over time till date.

Questions for future analysis:

1. How many titles added in India?
2. Which year is the best for each country in terms of content growth?

Actionable Items:

1. Pre-process the country, Actor, Director etc columns to un-nest the values
2. Also the countries can be transformed in the regions/continents/sectors to have lesser catagories and to have regional analysis

Keeping only Numeric part of the Duration column

We split the duration column, for splitting string value

In [17]:

```

```

In [18]:

```
#Splitting duration and adding only numbers in new column

df['new_duration']=df['duration'].str.split(' ').str[0]
```

Casting nested datas to un-nested data's

As there are nested date's in cast,country,genre,director, we need to unnest and merge to single dataframe
We will be using the explode() function for the same

In [19]:

```
# Un-nesting the Country Field

## books = books.assign(tags=books.tags.str.split(","))
df_1 = df.assign(country=df.country.str.split(", "))
#print("Row count before Exploding Country Field")
#print(df_1['title'].value_counts().value_counts())
df_1=df_1.explode("country")
#print("Row count After Exploding Country Field")
#print(df_1['title'].value_counts().value_counts())

df_1.head(3)
```

Out[19]:

	show_id	type	title	director	cast	country	date_added	release_year	ratin
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	Unknown	United States	2021-09-25	2020	PG 1
1	s2	TV Show	Blood & Water	Unknown	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	2021-09-24	2021	TV M
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	United States	2021-09-24	2021	TV M

In [20]:

```
df_1['country'].head(20)
```

Out[20]:

```
0        United States
1         South Africa
2        United States
3        United States
4                India
5        United States
6        United States
7        United States
7                Ghana
7         Burkina Faso
7       United Kingdom
7              Germany
7             Ethiopia
8       United Kingdom
9        United States
10       United States
11       United States
12             Germany
12      Czech Republic
13       United States
Name: country, dtype: object
```

In [21]:

```
# Un-nesting the Country Field

## books = books.assign(tags=books.tags.str.split(","))
df_1 = df.assign(country=df.country.str.split(", "))
#print("Row count before Exploding Country Field")
#print(df_1['title'].value_counts().value_counts())
df_1=df_1.explode("country")
#print("Row count After Exploding Country Field")
#print(df_1['title'].value_counts().value_counts())

df_1.head(5)
```

Out[21]:

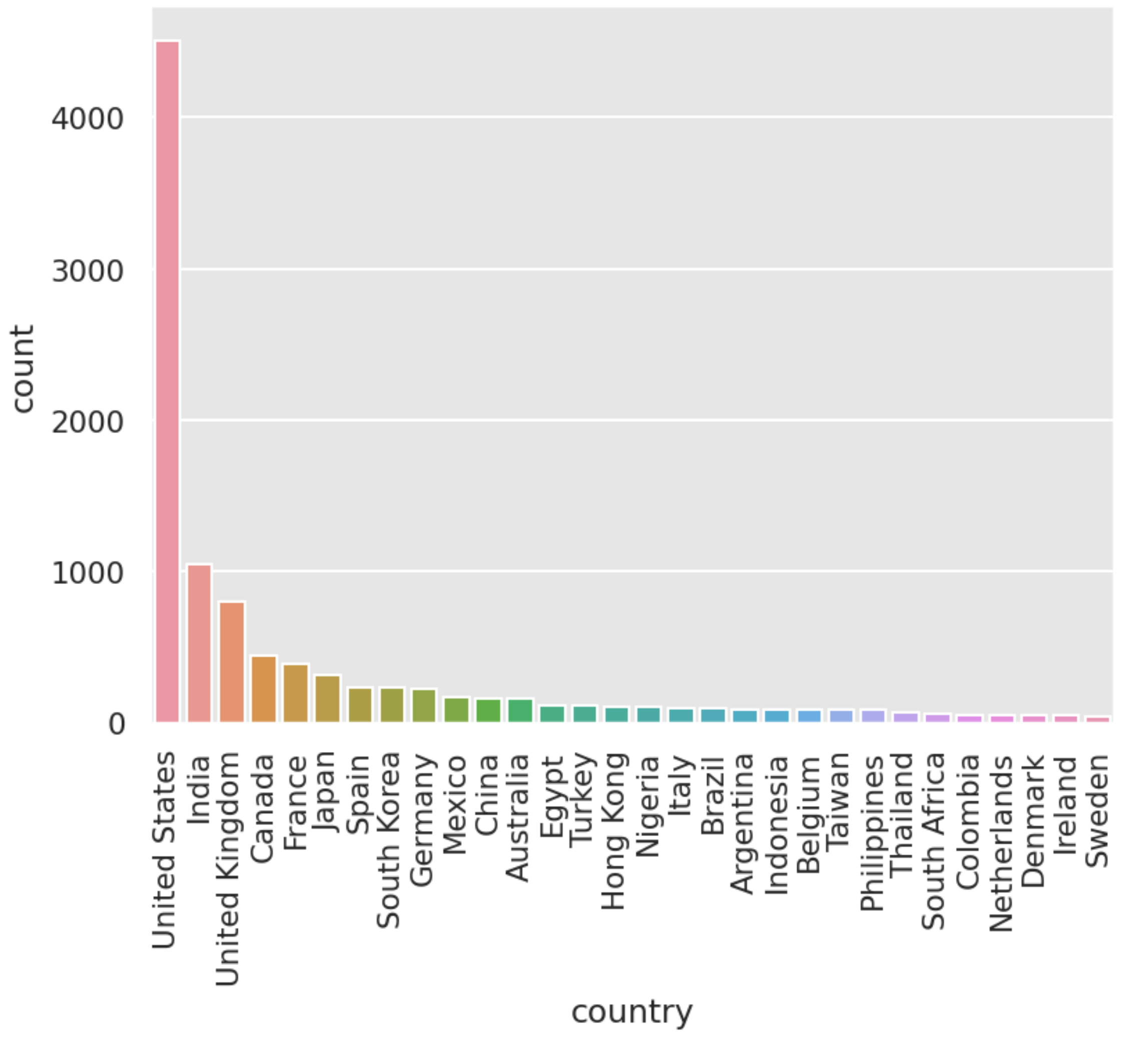
	show_id	type	title	director	cast	country	date_added	release_year	ratin
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	Unknown	United States	2021-09-25	2020	PG 1
1	s2	TV Show	Blood & Water	Unknown	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	2021-09-24	2021	TV M
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	United States	2021-09-24	2021	TV M
3	s4	TV Show	Jailbirds New Orleans	Unknown	Unknown	United States	2021-09-24	2021	TV M
4	s5	TV Show	Kota Factory	Unknown	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	2021-09-24	2021	TV M

In [22]:

```
## Country wise distribution of the content available on the platform  -- To validate th

group_country_movies = df_1.groupby('country')['show_id'].count().sort_values(ascending

countries_list = []
count_list = []
for index, value in group_country_movies.items():
    countries_list.append(index)
    count_list.append(value)

cars = {
    'country': countries_list,
    'count': count_list
}

df4 = pd.DataFrame(cars, columns = ['country', 'count'])


sns.set_style("darkgrid", {"axes.facecolor": ".9"})
# possible styles: whitegrid, dark, white

sns.set_context("notebook")


ax = sns.barplot(x = "country", y = "count", data = df4)
ax.set_xticklabels(ax.get_xticklabels(), rotation = 90)
```

Out[22]:

```
[Text(0, 0, 'United States'),
 Text(1, 0, 'India'),
 Text(2, 0, 'United Kingdom'),
 Text(3, 0, 'Canada'),
 Text(4, 0, 'France'),
 Text(5, 0, 'Japan'),
 Text(6, 0, 'Spain'),
 Text(7, 0, 'South Korea'),
 Text(8, 0, 'Germany'),
 Text(9, 0, 'Mexico'),
 Text(10, 0, 'China'),
 Text(11, 0, 'Australia'),
 Text(12, 0, 'Egypt'),
 Text(13, 0, 'Turkey'),
 Text(14, 0, 'Hong Kong'),
 Text(15, 0, 'Nigeria'),
 Text(16, 0, 'Italy'),
 Text(17, 0, 'Brazil'),
 Text(18, 0, 'Argentina'),
 Text(19, 0, 'Indonesia'),
 Text(20, 0, 'Belgium'),
 Text(21, 0, 'Taiwan'),
 Text(22, 0, 'Philippines'),
 Text(23, 0, 'Thailand'),
 Text(24, 0, 'South Africa'),
 Text(25, 0, 'Colombia'),
 Text(26, 0, 'Netherlands'),
 Text(27, 0, 'Denmark'),
 Text(28, 0, 'Ireland'),
 Text(29, 0, 'Sweden')]
```



In [23]:

```
group_country_movies.head(20)
```

Out[23]:

```
country
United States     4509
India             1046
United Kingdom     803
Canada             445
France             393
Japan              316
Spain              232
South Korea        231
Germany            226
Mexico             169
China              162
Australia          158
Egypt              117
Turkey             113
Hong Kong          105
Nigeria            103
Italy               99
Brazil              97
Argentina           91
Indonesia           90
Name: show_id, dtype: int64
```

In [24]:

```
df_1.shape
```

Out[24]:

```
(10828, 16)
```

In [25]:

```
# Un-nesting the Director Field

## books = books.assign(tags=books.tags.str.split(","))
df_2 = df_1.assign(director=df.director.str.split(", "))
#print("Row count before Exploding director Field")
#print(df_2['title'].value_counts().value_counts())
df_2=df_2.explode("director")
#print("Row count After Exploding director Field")
#print(df_2['title'].value_counts().value_counts())

df_2.head(5)
df_2.shape
```

Out[25]:

```
(11895, 16)
```

In [26]:

```
group_country_movies.head(20)
```

Out[26]:

```
country
United States     4509
India             1046
United Kingdom     803
Canada             445
France             393
Japan              316
Spain              232
South Korea        231
Germany            226
Mexico             169
China              162
Australia          158
Egypt              117
Turkey             113
Hong Kong          105
Nigeria            103
Italy               99
Brazil              97
Argentina           91
Indonesia           90
Name: show_id, dtype: int64
```

In [27]:

```
# Un-nesting the Cast Field

## books = books.assign(tags=books.tags.str.split(","))
df_3 = df_2.assign(cast=df.cast.str.split(", "))
#print("Row count before Exploding director Field")
#print(df_3['title'].value_counts().value_counts())
df_3=df_3.explode("cast")
#print("Row count After Exploding director Field")
#print(df_3['title'].value_counts().value_counts())

df_3.head(5)
df_3.shape
```

Out[27]:

```
(89272, 16)
```

In [28]:

```
df_1.shape
```

Out[28]:

```
(10828, 16)
```

In [29]:

```
df_3.head()
```

Out[29]:

	show_id	type	title	director	cast	country	date_added	release_year	rating
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	Unknown	United States	2021-09-25	2020	PG-13
1	s2	TV Show	Blood & Water	Unknown	Ama Qamata	South Africa	2021-09-24	2021	TV-MA
1	s2	TV Show	Blood & Water	Unknown	Khosi Ngema	South Africa	2021-09-24	2021	TV-MA
1	s2	TV Show	Blood & Water	Unknown	Gail Mabalane	South Africa	2021-09-24	2021	TV-MA
1	s2	TV Show	Blood & Water	Unknown	Thabang Molaba	South Africa	2021-09-24	2021	TV-MA

In [30]:

```
# Un-nesting the listed_in Field

## books = books.assign(tags=books.tags.str.split(","))
df_4 = df_3.assign(listed_in=df.listed_in.str.split(", "))
print("Row count before Exploding listed_in Field")
print(df_4['title'].value_counts().value_counts())
df_4=df_4.explode("listed_in")
print("Row count After Exploding listed_in Field")
print(df_4['title'].value_counts().value_counts())

df_4.head(5)
df_4.shape
```

```
Row count before Exploding listed_in Field
1      1362
10     1120
8       986
6       560
9       541
       ...
78        1
75        1
46        1
29        1
468       1
Name: title, Length: 81, dtype: int64
Row count After Exploding listed_in Field
24     680
1      678
30     624
2      608
18     503
      ...
141      1
153      1
156      1
171      1
700      1
Name: title, Length: 117, dtype: int64
```

Out[30]:

```
(201763, 16)
```

In [31]:

```
df.shape
```

Out[31]:

```
(8790, 16)
```

In [32]:

```
## validation for data consistency after un-nesting the data
group_country_movies = df_1.groupby('country')['show_id'].count().sort_values(ascending
group_country_movies_unique = df_4.groupby(['country'])['show_id'].nunique().sort_values

print(group_country_movies.head(10))
print(group_country_movies_unique.head(10))
```

```
country
United States     4509
India             1046
United Kingdom     803
Canada             445
France             393
Japan              316
Spain              232
South Korea        231
Germany            226
Mexico             169
Name: show_id, dtype: int64
country
United States     4509
India             1046
United Kingdom     803
Canada             445
France             393
Japan              316
Spain              232
South Korea        231
Germany            226
Mexico             169
Name: show_id, dtype: int64
```

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

In [33]:

```
#importing Seaborn library
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go
from plotly.subplots import make_subplots
import matplotlib.pyplot as plt
```

In [34]:

```
df_4.columns
```

Out[34]:

```
Index(['show_id', 'type', 'title', 'director', 'cast', 'country', 'date_ad
ded',
       'release_year', 'rating', 'duration', 'listed_in', 'description',
       'month_added', 'month_name_added', 'year_added', 'new_duration'],
      dtype='object')
```

In [34]:

```

```

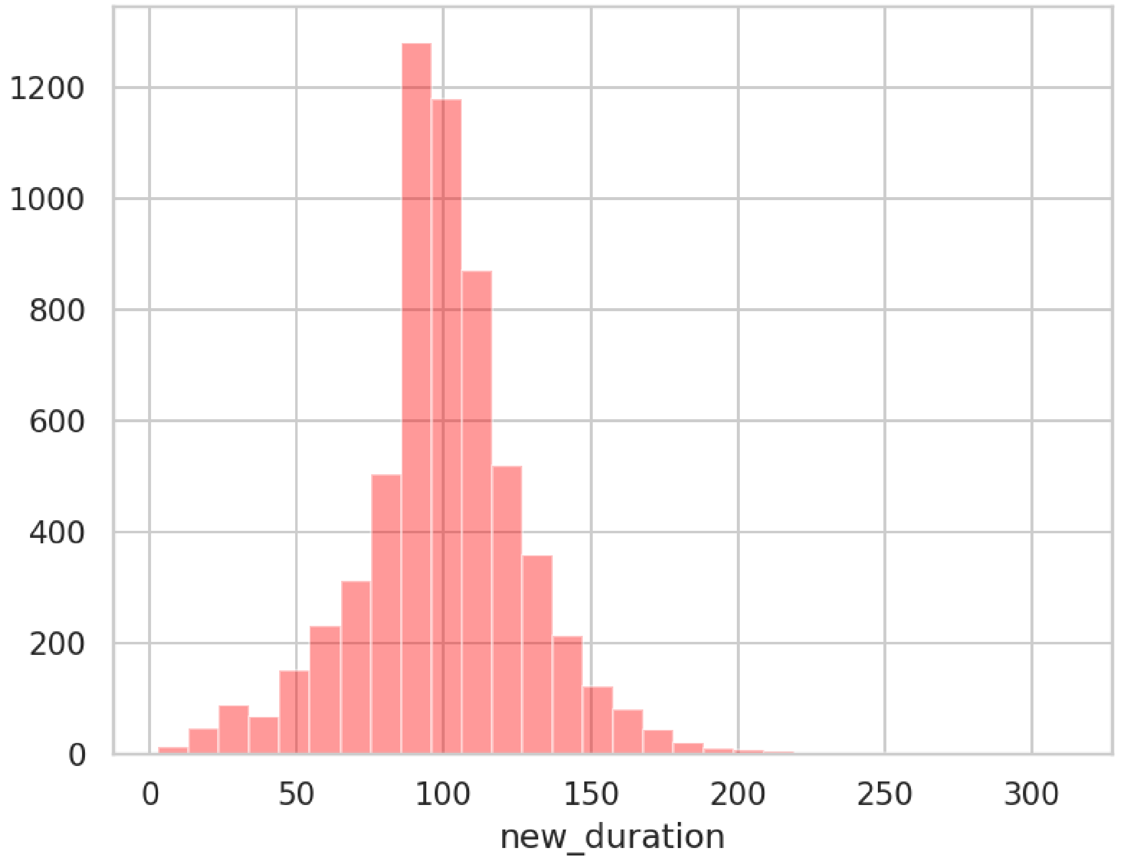
4.1 For continuous variable(s): Distplot, countplot, histogram for univariate analysis (10 Points)

In [35]:

```
## distribution plot for the duration of the content of type 'Movie'
sns.set_style('whitegrid')
sns.distplot(df[df['type']=='Movie']['new_duration'], kde = False, color ='red', bins =
```

Out[35]:

```
<Axes: xlabel='new_duration'>
```



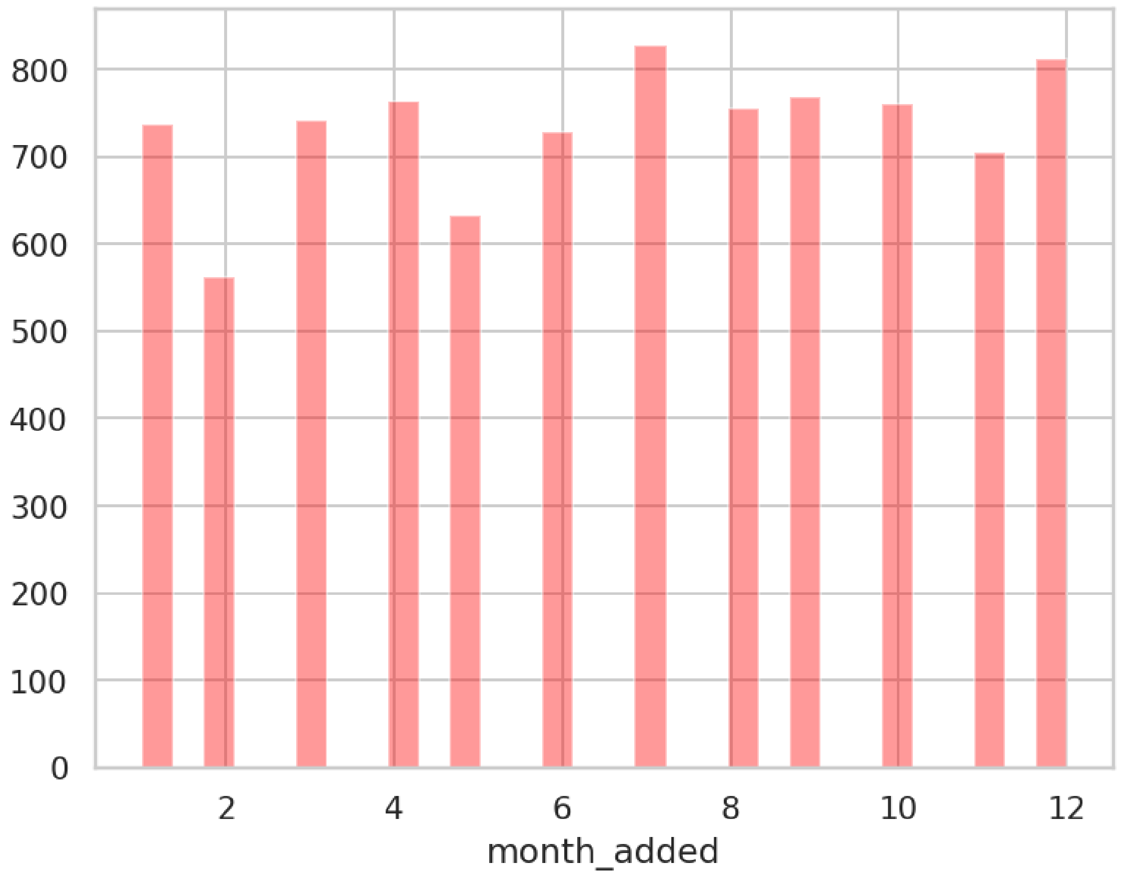
Above graph indicates that the movies on the netflix platform are having runtime length of approx 100~120 minutes which is the normal movie length time. The content available in the short-movies seems very less and the short-films content can be made available in more quantity subject to the viewership(TRP)analysis.

In [36]:

```
## distribution plot for the month of the content
sns.set_style('whitegrid')
sns.distplot(df['month_added'], kde = False, color ='red', bins = 30)
#sns.distplot(target_0[['sepal length (cm)']], hist=False, rug=True)
#sns.distplot(target_1[['sepal length (cm)']], hist=False, rug=True)
```

Out[36]:

```
<Axes: xlabel='month_added'>
```

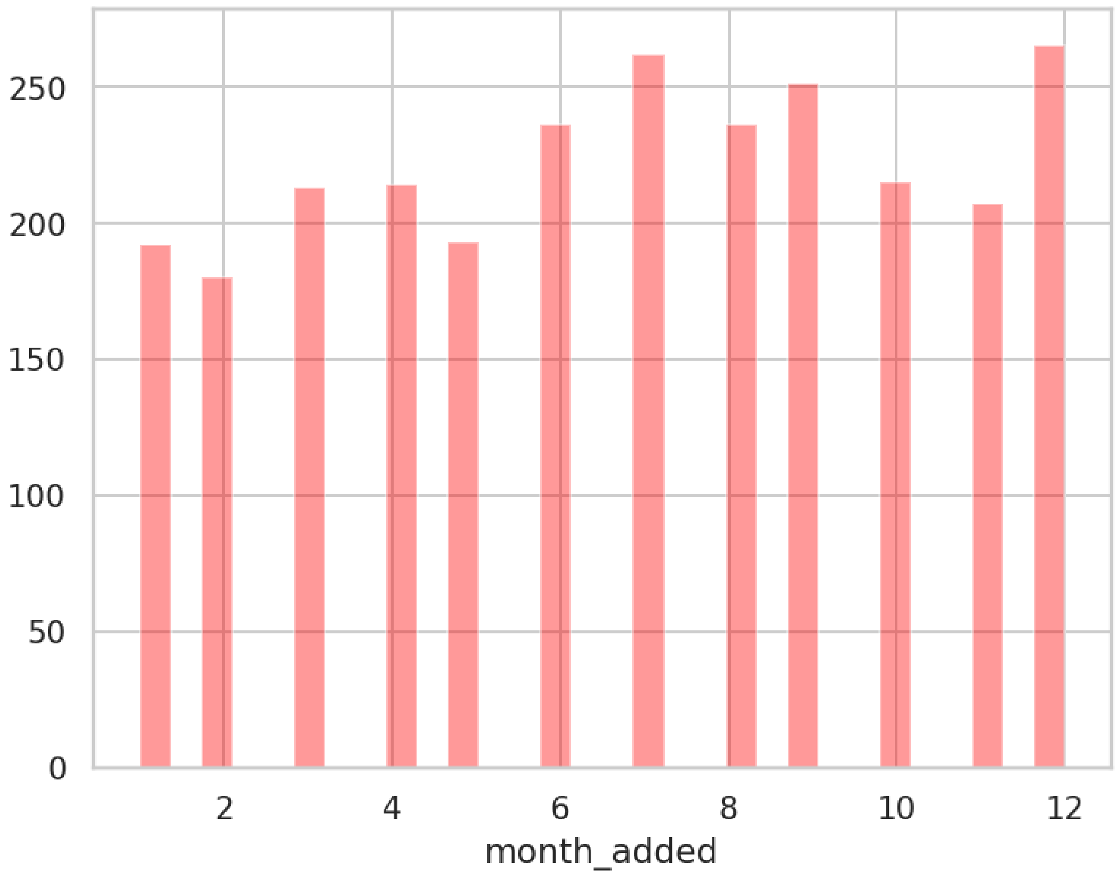


In [37]:

```
## distribution plot for the month of the content
sns.set_style('whitegrid')
sns.distplot(df[df['type']!='Movie']['month_added'], kde = False, color ='red', bins = 3
#sns.distplot(target_0[['sepal length (cm)']], hist=False, rug=True)
#sns.distplot(target_1[['sepal length (cm)']], hist=False, rug=True)
```

Out[37]:

```
<Axes: xlabel='month_added'>
```



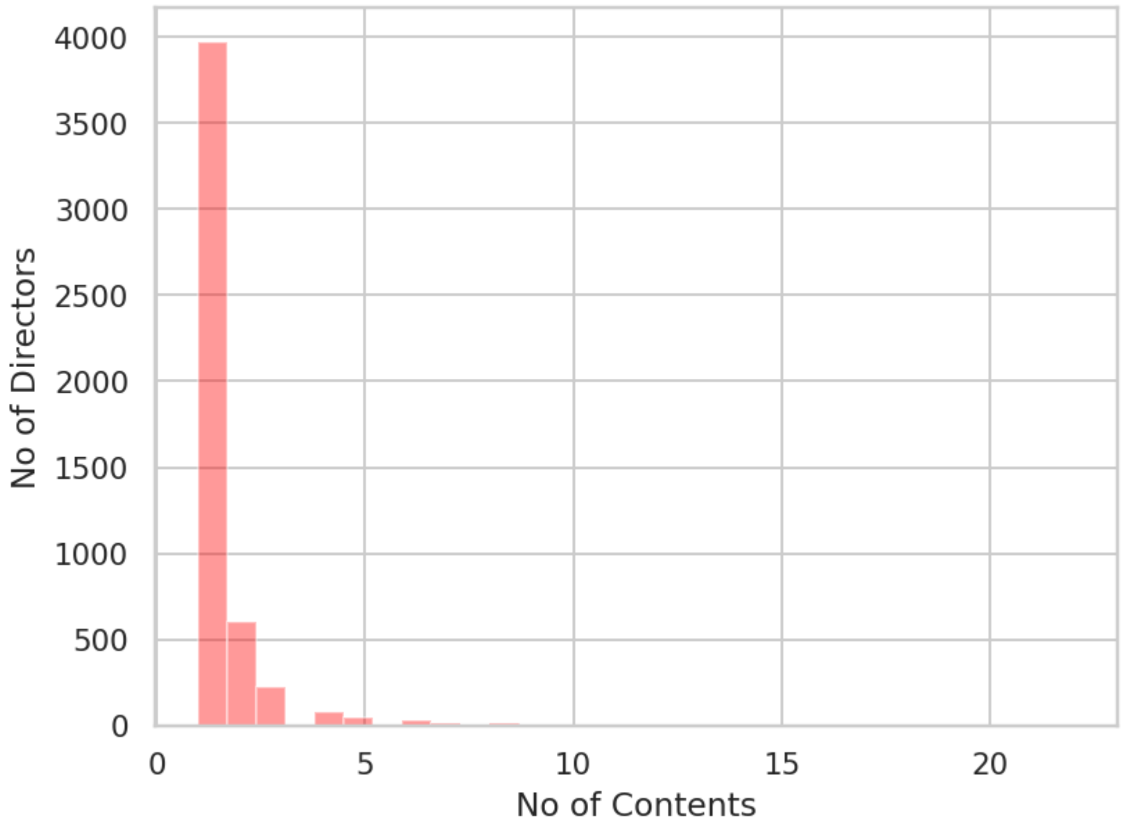
The distribution of the content based on the month of uploading on the platform, indicates that there is no fixed pattern as such and is mostly dependent on the release of the contents. This could be the indicator of the fact that the content publishing is subject to the competiotion and availability of the content.

In [38]:

```
## distribution plot for the month of the content
sns.set_style('whitegrid')
fig = sns.distplot(df_4[df_4['director']!='Unknown'].groupby(['director'])['show_id'].nu
#fig.set_title("Month-wise Content Distr")
fig.set_xlabel("No of Contents")
fig.set_ylabel("No of Directors")
```

Out[38]:

```
Text(0, 0.5, 'No of Directors')
```



Above plot indicates that there are very few contents delivered by same directors. Based on the fame and the rating of the directors strategy can be established to upload content from the directors having more viewership on the platform.

In [39]:

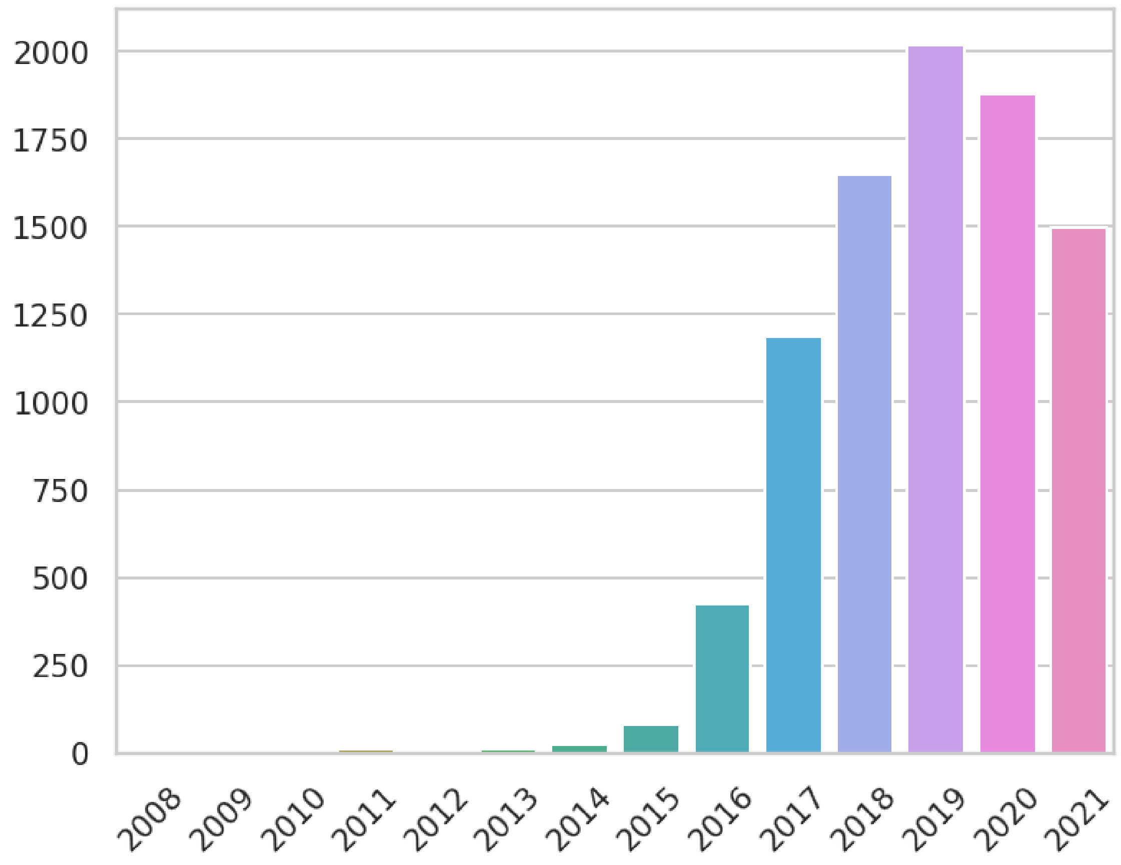
```
#df_4.groupby(['director'])['show_id'].nunique().sort_values(ascending = False).head(50)
#df_4[df_4['director']!='Unknown'].groupby(['director'])['show_id'].nunique().sort_value

df_4[df_4['type']=='Movie']['rating'].unique()
```

Out[39]:

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

In [40]:

```
## Year wise content distribution based on releasing it on the platform
import seaborn as sns
import matplotlib.pyplot as plt

ax = sns.barplot(
    x = df['year_added'].value_counts().keys(),
    y = df['year_added'].value_counts().values
)
ax.set_xticklabels(ax.get_xticklabels(), rotation = 45)
plt.show()
```



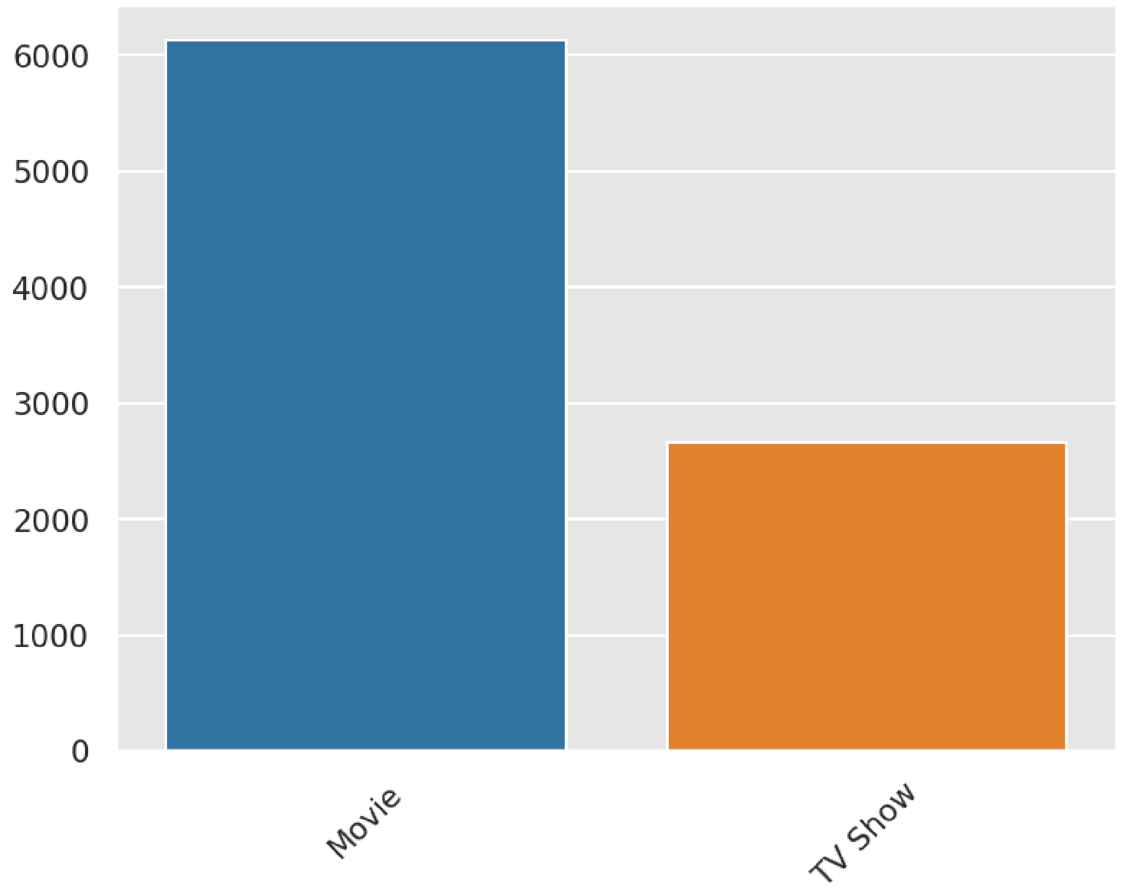
From the above graph and the non-graphical statistical details below can be concluded:

1. most of the content is from 2013-2019 releases
2. most of the content was added between 2018-2020
3. 2019 being the year with highest content, there has been slight drop in the content published.
4. 2019 also was the year where maximum runtime of video content was uploaded

4.2 For categorical variable(s): Boxplot (10 Points)

In [43]:

```
## Year wise content distribution based on releasing it on the platform
import seaborn as sns
import matplotlib.pyplot as plt

ax = sns.barplot(
    x = df['type'].value_counts().keys(),
    y = df['type'].value_counts().values
)
ax.set_xticklabels(ax.get_xticklabels(), rotation = 45)
plt.show()
```

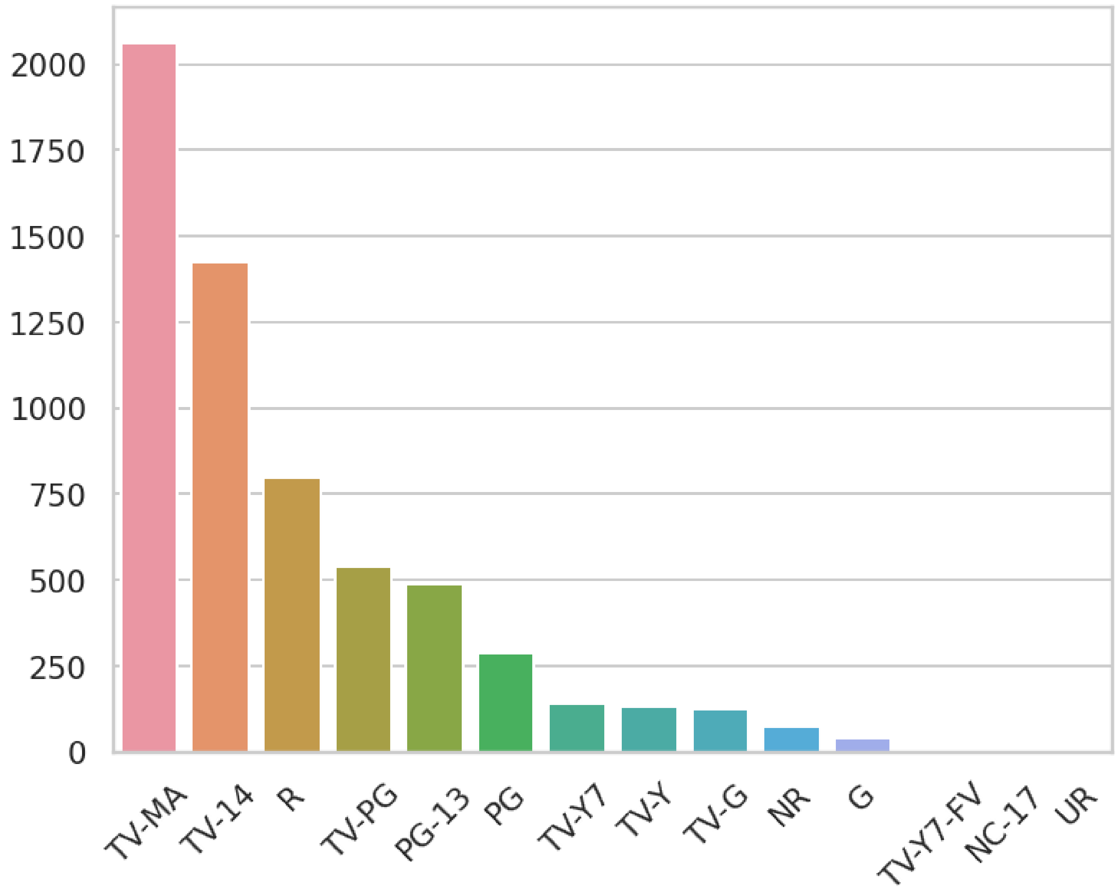


Netflix has more Movie content published than the TV shows

1. Viewrship and rating analysis can be done to decide on expanding the TV show footprint

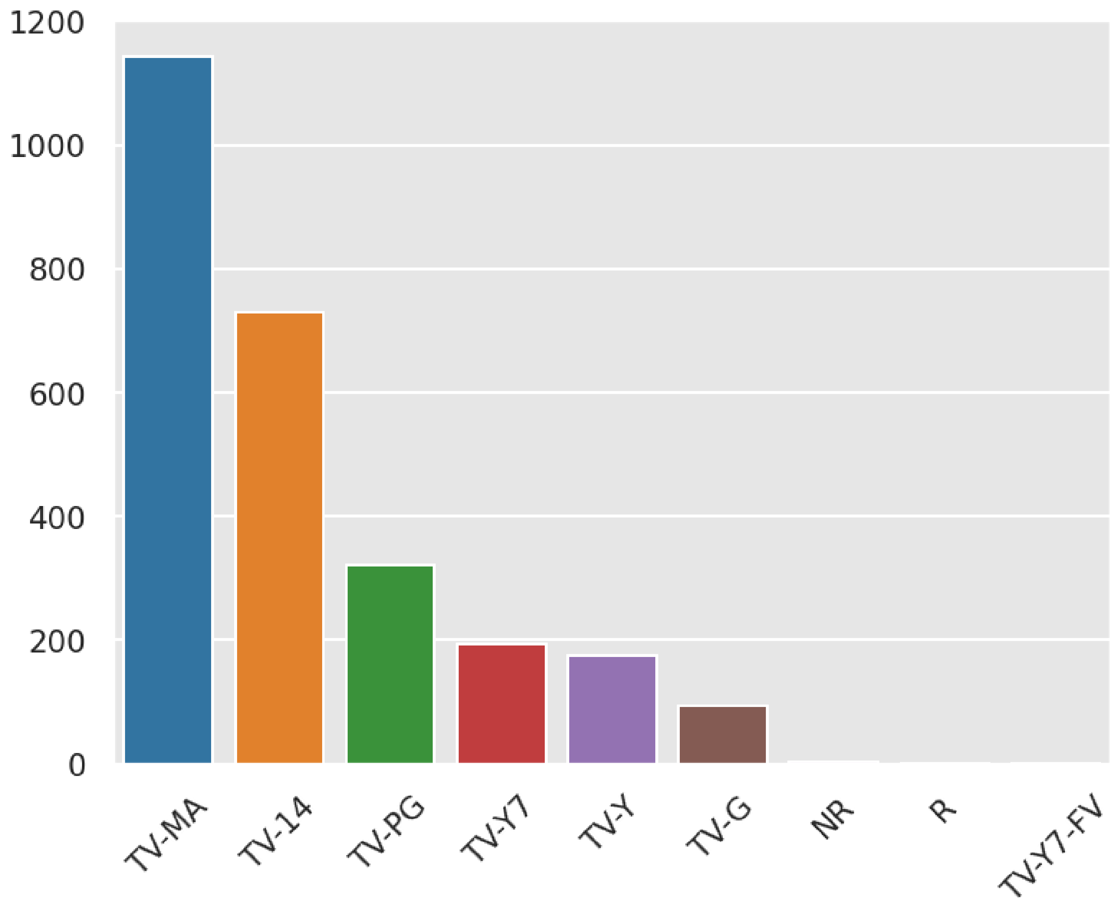
In [51]:

```
## rating plot for the movie content
import seaborn as sns
import matplotlib.pyplot as plt

ax = sns.barplot(
    x = df[df['type']=='Movie']['rating'].value_counts().keys(),
    y = df[df['type']=='Movie']['rating'].value_counts().values
)
ax.set_xticklabels(ax.get_xticklabels(), rotation = 45)
plt.show()
```



In [45]:

```
## rating plot for the 'TV Show'' content
import seaborn as sns
import matplotlib.pyplot as plt

ax = sns.barplot(
    x = df[df['type']!='Movie']['rating'].value_counts().keys(),
    y = df[df['type']!='Movie']['rating'].value_counts().values
)
ax.set_xticklabels(ax.get_xticklabels(), rotation = 45)
plt.show()
```



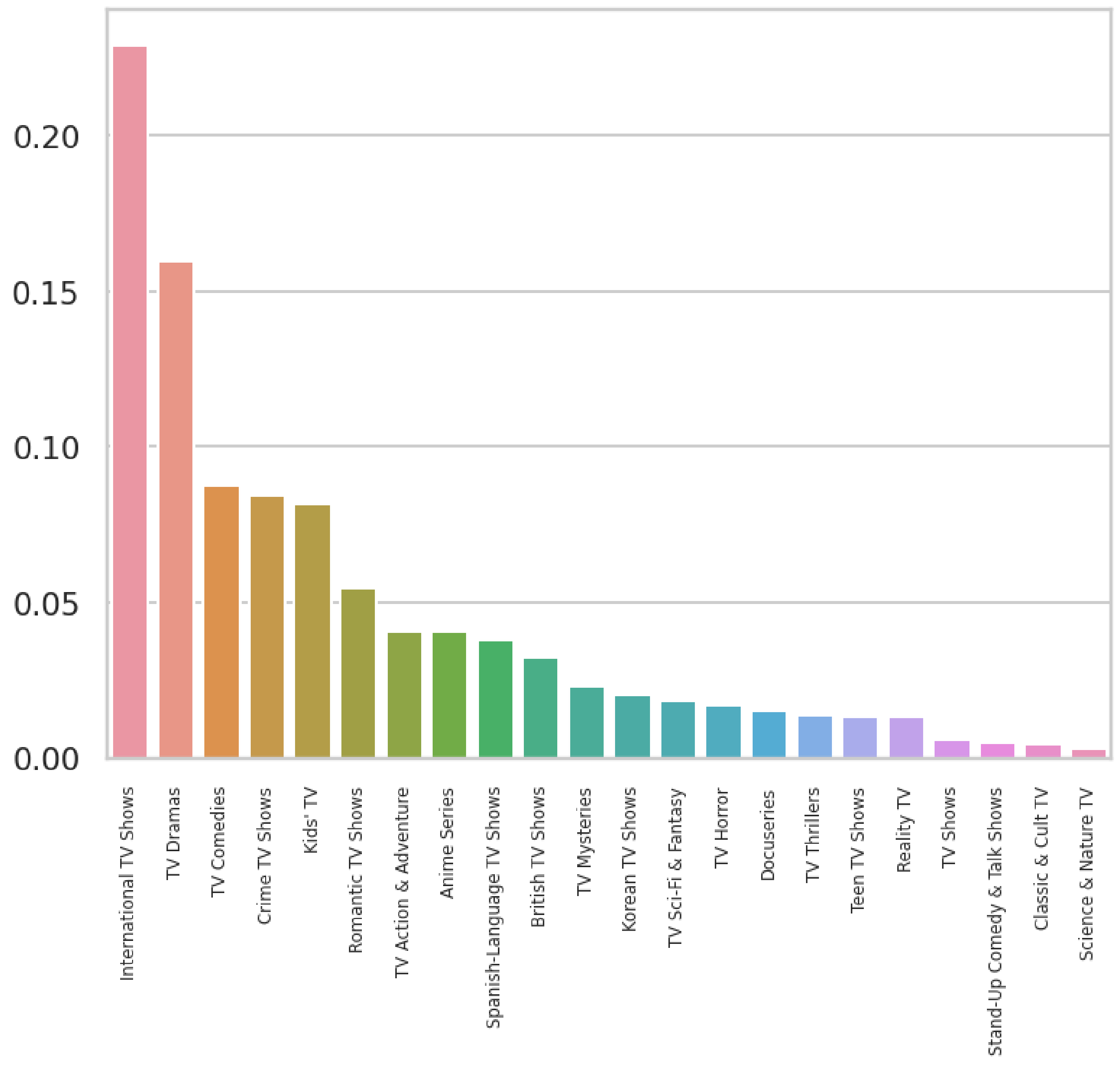
Majority of the TV show content is catered for the 17+ years age group.

The Y7 group contents can be merged together as TV-Y7-FV is similar to TV-Y7, which can be helpful to ease the analysis.

Future Work: The country wise ditribution of the rated contents can be studied further to provide more insights.

In [52]:

```
## Genre plot for the 'TV Show'' content
import seaborn as sns
import matplotlib.pyplot as plt

ax = sns.barplot(
    x = df_4[df['type']!='Movie']['listed_in'].value_counts(normalize=True).keys(),
    y = df_4[df['type']!='Movie']['listed_in'].value_counts(normalize=True).values
)
ax.set_xticklabels(ax.get_xticklabels(),fontsize=6, rotation = 90)
plt.show()
```



1. International Shows, Drams and Comedy are the highly available genre on Netflix.
2. Documentries and Science/Nature Shows contribution can be increased subject to the viewership

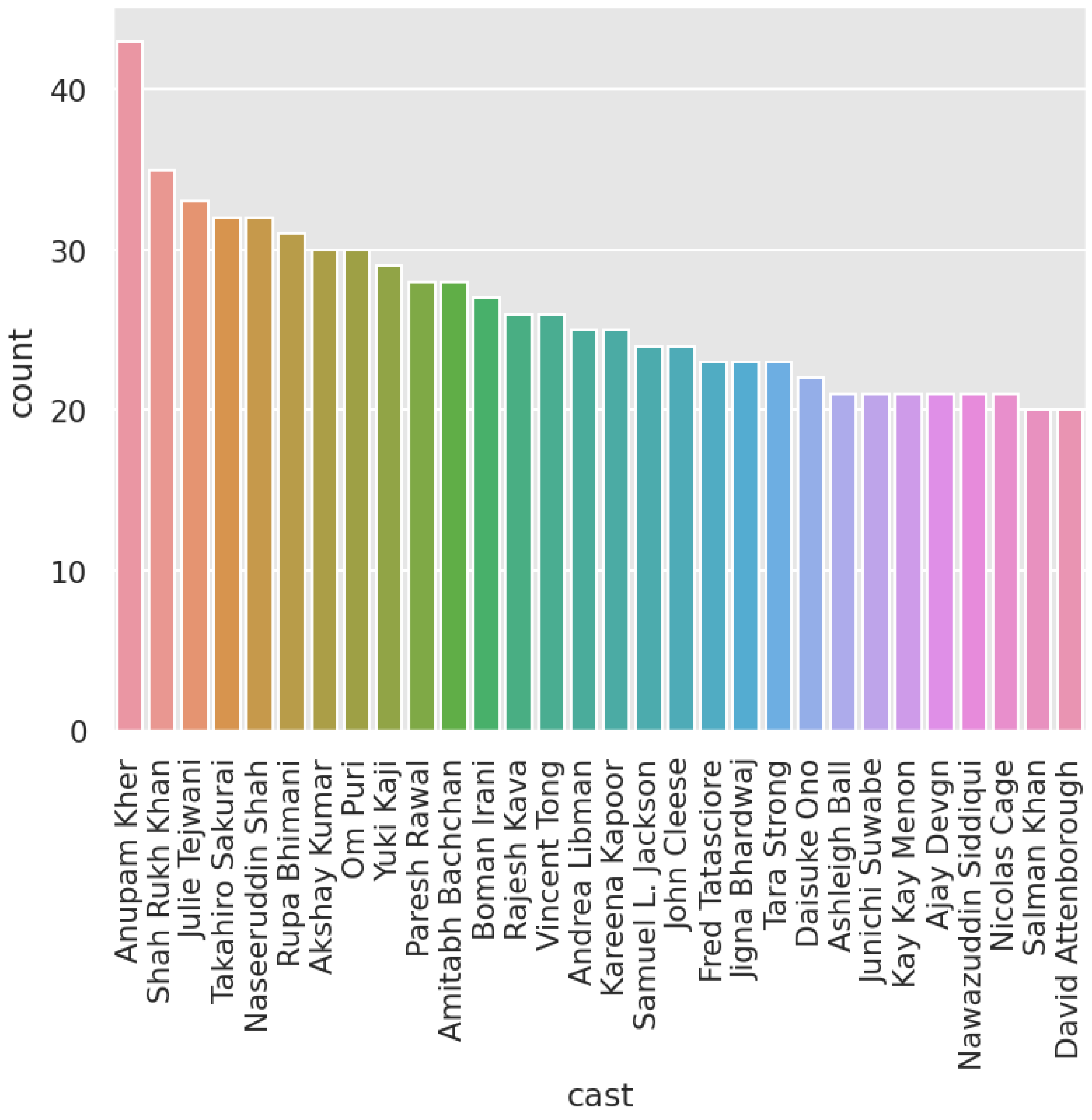
Future Scope: . Growth of the Korean series contents in other countries can be studied to give more insights. Each genre can further be analysed from the regional scope to provide insights.

In [47]:

```
## Top 30 actors in terms of no of contents available over netflix - excluding the Unkno

group_cast_movies = df_4[df_4['cast']!='Unknown'].groupby('cast')['show_id'].nunique().s

cast_list = []
count_list = []
for index, value in group_cast_movies.items():
    cast_list.append(index)
    count_list.append(value)

cars = {
    'cast': cast_list,
    'count': count_list
}

df4 = pd.DataFrame(cars, columns = ['cast', 'count'])


sns.set_style("darkgrid", {"axes.facecolor": ".9"})
# possible styles: whitegrid, dark, white

sns.set_context("notebook")


ax = sns.barplot(x = "cast", y = "count", data = df4)
ax.set_xticklabels(ax.get_xticklabels(), rotation = 90)
```

Out[47]:

```
[Text(0, 0, 'Anupam Kher'),
 Text(1, 0, 'Shah Rukh Khan'),
 Text(2, 0, 'Julie Tejwani'),
 Text(3, 0, 'Takahiro Sakurai'),
 Text(4, 0, 'Naseeruddin Shah'),
 Text(5, 0, 'Rupa Bhimani'),
 Text(6, 0, 'Akshay Kumar'),
 Text(7, 0, 'Om Puri'),
 Text(8, 0, 'Yuki Kaji'),
 Text(9, 0, 'Paresh Rawal'),
 Text(10, 0, 'Amitabh Bachchan'),
 Text(11, 0, 'Boman Irani'),
 Text(12, 0, 'Rajesh Kava'),
 Text(13, 0, 'Vincent Tong'),
 Text(14, 0, 'Andrea Libman'),
 Text(15, 0, 'Kareena Kapoor'),
 Text(16, 0, 'Samuel L. Jackson'),
 Text(17, 0, 'John Cleese'),
 Text(18, 0, 'Fred Tatasciore'),
 Text(19, 0, 'Jigna Bhardwaj'),
 Text(20, 0, 'Tara Strong'),
 Text(21, 0, 'Daisuke Ono'),
 Text(22, 0, 'Ashleigh Ball'),
 Text(23, 0, 'Junichi Suwabe'),
 Text(24, 0, 'Kay Kay Menon'),
 Text(25, 0, 'Ajay Devgn'),
 Text(26, 0, 'Nawazuddin Siddiqui'),
 Text(27, 0, 'Nicolas Cage'),
 Text(28, 0, 'Salman Khan'),
 Text(29, 0, 'David Attenborough')]
```



Top 30 actors in terms of no of contents available over netflix

In terms of the volume of the contents available Indian actors are on top Slaman Khan has less presence on the Netflix, subject to the availability of the audience the content can be increased for the similar actors.

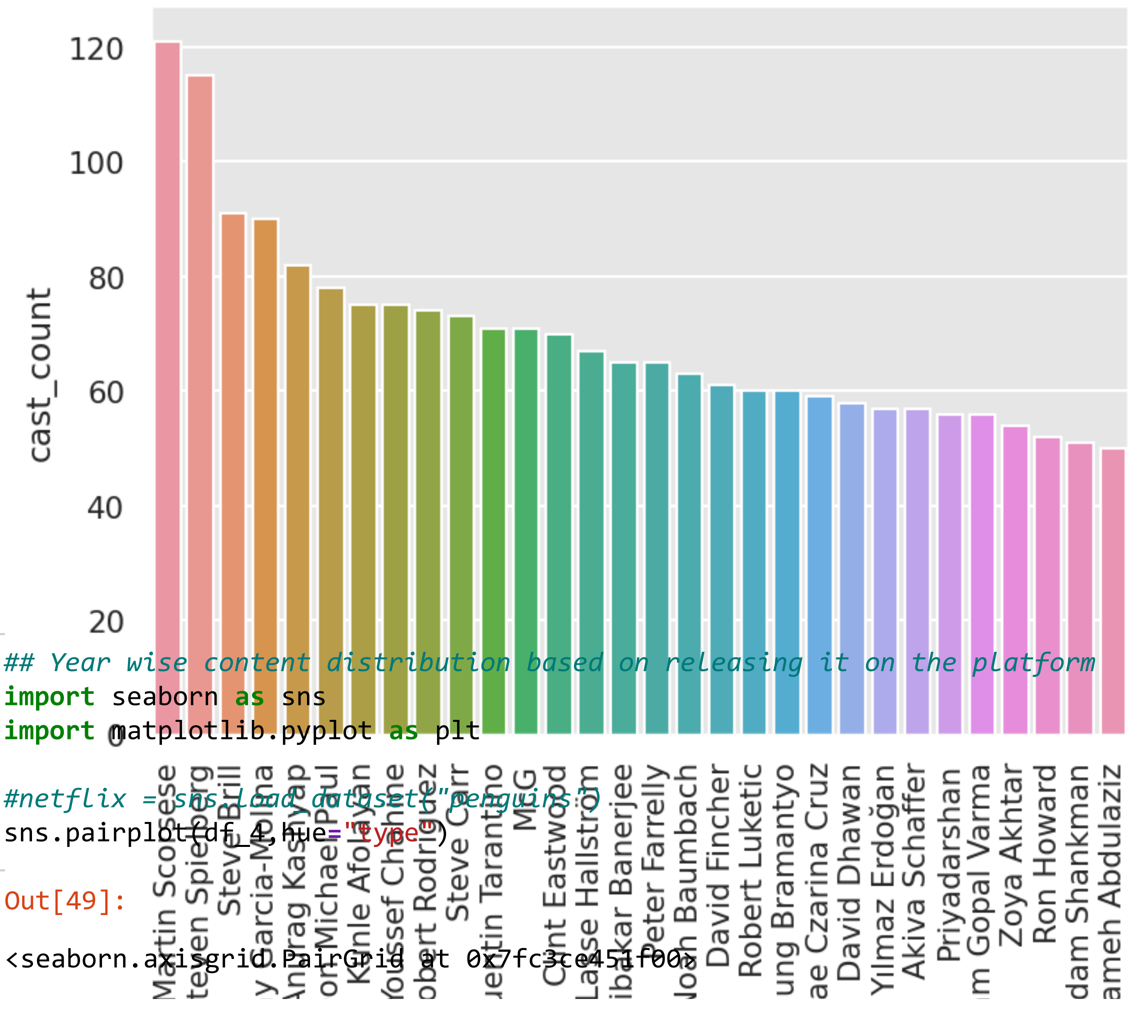
Future Scope: The analysis based on the presence of the actor in the titles can be done to appropriate the representation.

In [48]:

```
## Top 30 directors who repeatedly works with the most number of distinct actors wrt no 
# excluding the Unknown director member contents

group_dir_cast = df_4[df_4['director']!='Unknown'].groupby('director')['cast'].nunique()

dir_list = []
count_list = []
for index, value in group_dir_cast.items():
    dir_list.append(index)
    count_list.append(value)

cars = {
    'dir': dir_list,
    'cast_count': count_list
}

df4 = pd.DataFrame(cars, columns = ['dir', 'cast_count'])


sns.set_style("darkgrid", {"axes.facecolor": ".9"})
# possible styles: whitegrid, dark, white

sns.set_context("notebook")


ax = sns.barplot(x = "dir", y = "cast_count", data = df4)
ax.set_xticklabels(ax.get_xticklabels(), rotation = 90)
ax.set_yticklabels(ax.get_yticklabels())
```

Out[48]:

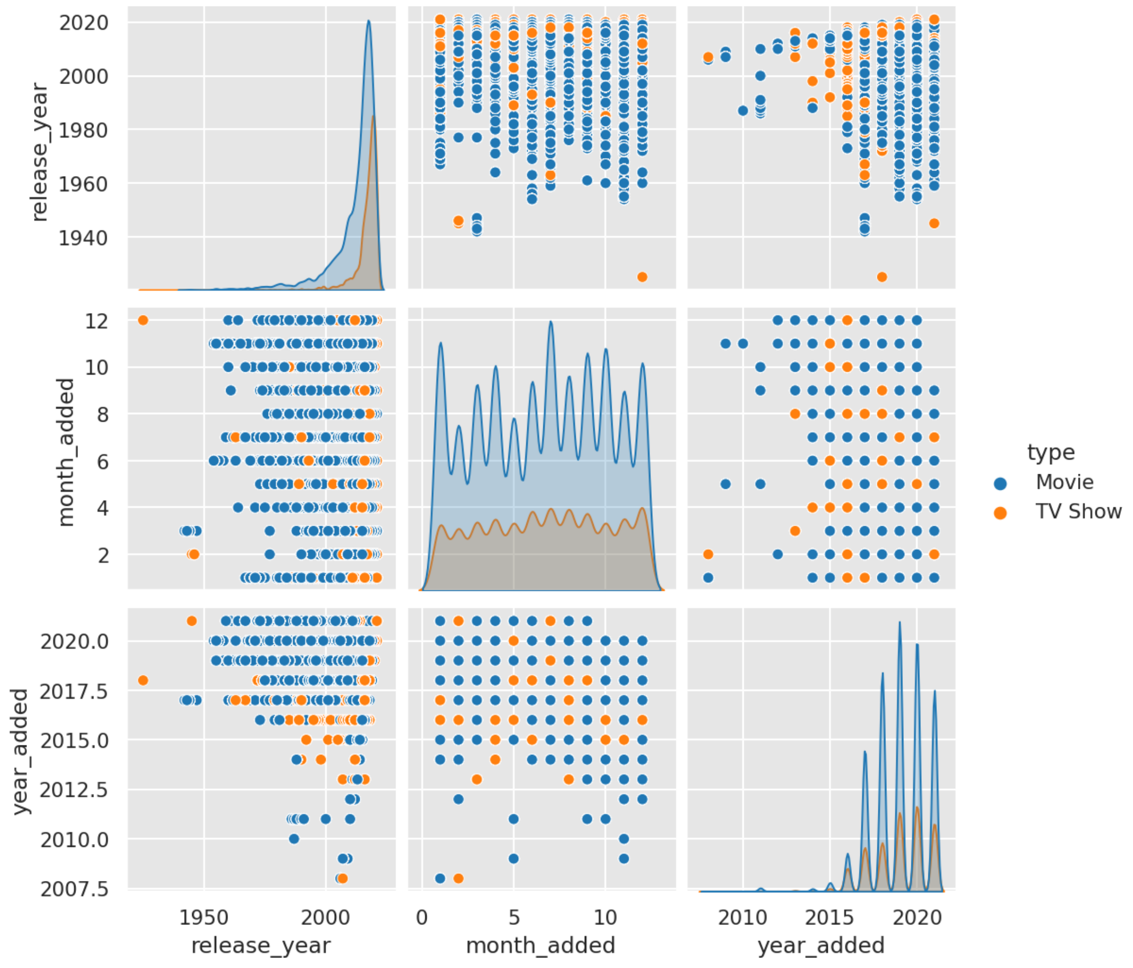
```
[Text(0, 0.0, '0'),
 Text(0, 20.0, '20'),
 Text(0, 40.0, '40'),
 Text(0, 60.0, '60'),
 Text(0, 80.0, '80'),
 Text(0, 100.0, '100'),
 Text(0, 120.0, '120'),
 Text(0, 140.0, '140')]
```



```
## Year wise content distribution based on releasing it on the platform
import seaborn as sns
import matplotlib.pyplot as plt

#netflix = sns.load_dataset("penguins")
sns.pairplot(df_4,hue="type")
```

Out[49]:

```
<seaborn.axisgrid.PairGrid at 0x7fc3ce451f00>
```



1. The Pair plot for the Numerical continuous variables have been plotted successfully

2. The TV Show and Movie content have been uploaded in consistent ratio in terms of the volume
3. The density of bubbles indicates that the majority of content released post 2013 has been uploaded, and the upload were higher in 2018-20 period
4. In recent years the TV show contents have been uploaded in short times which could be an indicator of upload of TV series with multiple seasons being uploaded

In [51]:

```
df_4.corr()
```

Out[51]:

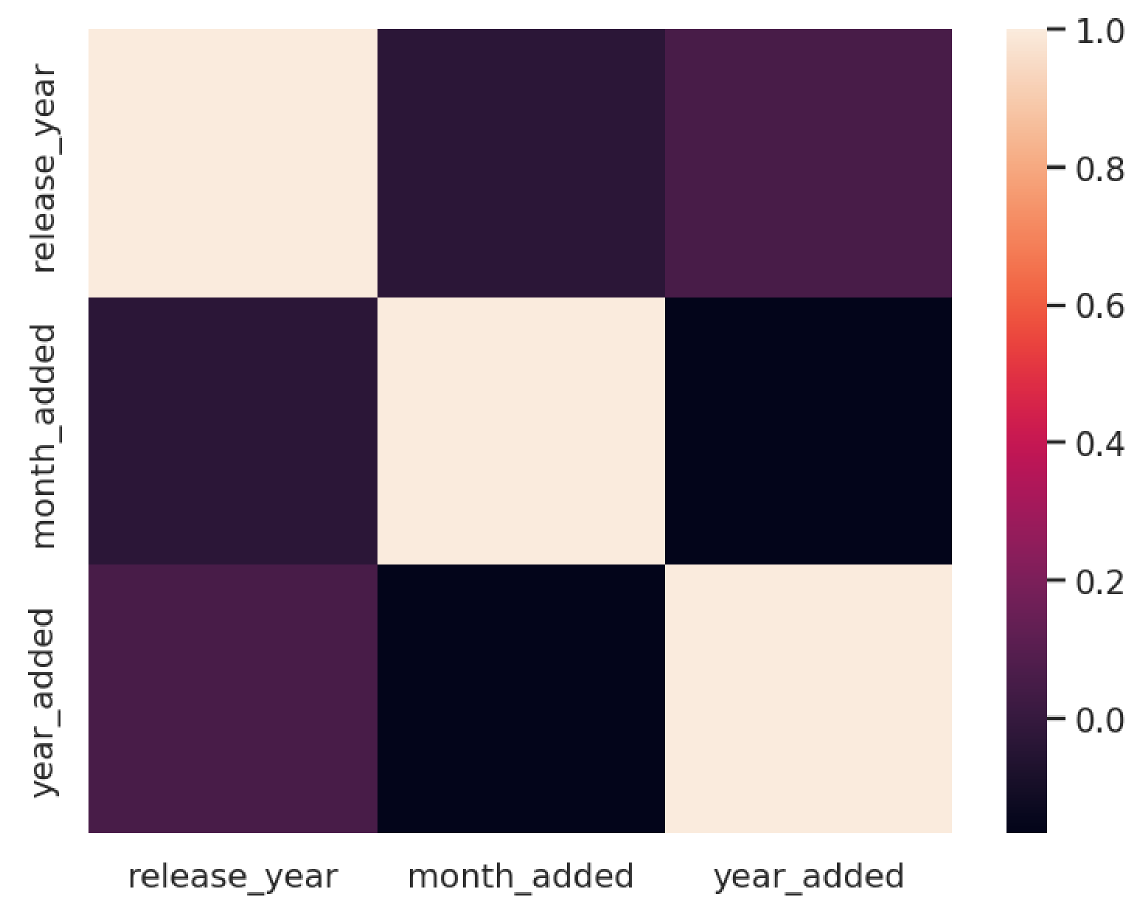
	release_year	month_added	year_added
release_year	1.000000	-0.031691	0.053031
month_added	-0.031691	1.000000	-0.166516
year_added	0.053031	-0.166516	1.000000

In [52]:

```
## Year wise content distribution based on releasing it on the platform
import seaborn as sns
import matplotlib.pyplot as plt

#netflix = sns.load_dataset("penguins")
#df_heatmap = df.pivot('type','country','year_added' )
sns.heatmap(df_4.corr())
```

Out[52]:

```
<Axes: >
```



Release year and year_added have little correlation between them which can be explained by the fact that the majority of the contents from the recent years have been published recently on Netflix

As the older released content is published the correlation index reduces

In [53]:

```
'''
value_counts = df['course_difficulty'].value_counts()
## df.groupby('your_column_1')['your_column_2'].value_counts()
# converting to df and assigning new names to the columns
df_value_counts = pd.DataFrame(value_counts)
df_value_counts = df_value_counts.reset_index()
df_value_counts.columns = ['unique_values', 'counts for course_difficulty'] # change col
df_value_counts
'''
''' Top countries list
country
United States     4509
India             1046
United Kingdom     803
Canada             445

country_list = ['United States', 'India','United Kingdom','Canada','France','Japan','Spa
```

Out[53]:

```
" Top countries list\ncountry\nUnited States     4509\nIndia             1
046\nUnited Kingdom     803\nCanada             445\n\ncountry_list = ['Un
ited States', 'India','United Kingdom','Canada','France','Japan','Spai
n','South Korea','Germany', 'Mexico']"
```

In [53]:

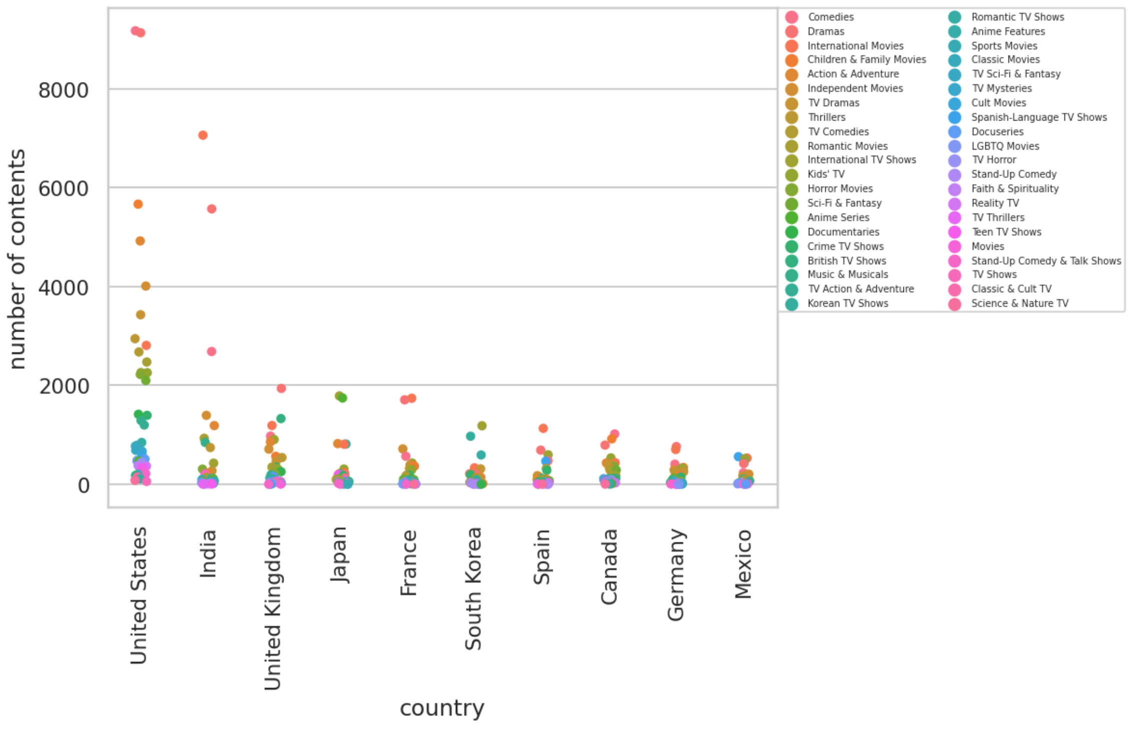
```
country_list = ['United States', 'India','United Kingdom','Canada','France','Japan','Spa
#.sort_values(ascending=False)
value_counts=df_4[df_4['country'].isin(country_list)].groupby('country')['listed_in'].va
df_value_counts = pd.DataFrame(value_counts)

#df.index.rename()
df_value_counts=df_value_counts.rename_axis(['Country','Genre'])
#
df_value_counts = df_value_counts.reset_index()
df_value_counts.columns = ['country' ,'Listed_in', 'number of contents'] # change column

df_value_counts.head(5)
#sns.heatmap(df_value_counts.corr())
```

Out[53]:

	country	Listed_in	number of contents
0	United States	Comedies	9171
1	United States	Dramas	9131
2	India	International Movies	7059
3	United States	Children & Family Movies	5665
4	India	Dramas	5569

In [54]:

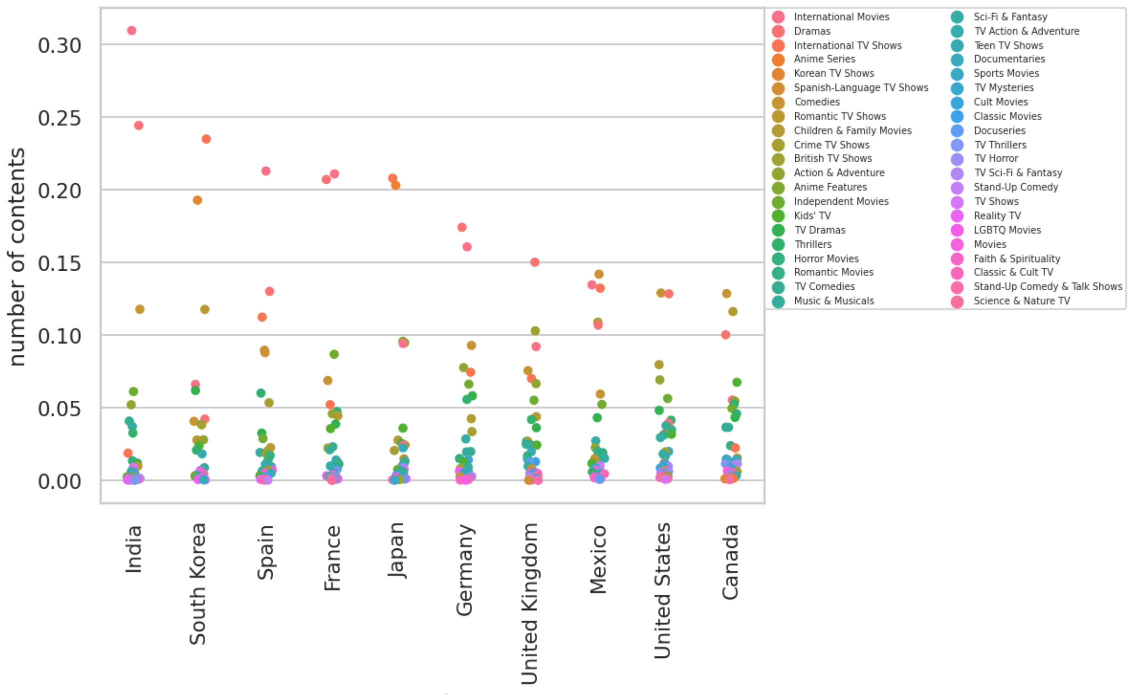
```
# Content Genre based analysis for top 10 countries
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set_style("whitegrid")

'''fig=sns.catplot(data=df_value_counts, x="country", y="number of contents",hue="Listed_
plt.setp(fig.get_legend().get_texts(), fontsize='10')
#fig.tick_params(axis='x', rotation=90)
plt.setp(fig, fontsize='4')
'''
ax = sns.stripplot(x="country", y="number of contents", hue="Listed_in", data=df_value_c
ax.tick_params(axis='x', rotation=90)
plt.setp(ax.get_legend().get_texts(), fontsize='2') # for legend text
plt.legend()
plt.legend(bbox_to_anchor=(1, 1), loc=2,ncol=2,fontsize='5', borderaxespad=0.)
plt.show()
```



In [56]:

```
country_list = ['United States', 'India','United Kingdom','Canada','France','Japan','Spa
#.sort_values(ascending=False)
value_counts=df_4[df_4['country'].isin(country_list)].groupby('country')['listed_in'].va
df_value_counts = pd.DataFrame(value_counts)

#df.index.rename()
df_value_counts=df_value_counts.rename_axis(['Country','Genre'])
#
df_value_counts = df_value_counts.reset_index()
df_value_counts.columns = ['country' ,'Listed_in', 'number of contents'] # change column

print(df_value_counts.head(10))
#sns.heatmap(df_value_counts.corr())

# Content Genre based analysis for top 10 countries
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set_style("whitegrid")

ax = sns.stripplot(x="country", y="number of contents", hue="Listed_in", data=df_value_c
ax.tick_params(axis='x', rotation=90)
plt.setp(ax.get_legend().get_texts(), fontsize='2') # for legend text
plt.legend()
plt.legend(bbox_to_anchor=(1, 1), loc=2,ncol=2,fontsize='5', borderaxespad=0.)
plt.show()
```

```
       country                Listed_in  number of contents
0        India     International Movies            0.309415
1        India                   Dramas            0.244104
2  South Korea  International TV Shows            0.234781
3        Spain     International Movies            0.212794
4       France     International Movies            0.210807
5        Japan  International TV Shows            0.207815
6       France                   Dramas            0.206930
7        Japan            Anime Series            0.202931
8  South Korea          Korean TV Shows            0.192742
9      Germany                   Dramas            0.174082
```



le on the

ıt in Indian

ı and other

ıntent in Indian

ı

commend for
inclusion/exclusion of the contents

In [ ]:

```
value_counts=df_4[df_4['country'].isin(country_list)].groupby('country')['listed_in'].va
value_counts.keys()
```

5. Missing Value & Outlier check (Treatment optional) (10 Points)

Using boxplots to identify outliers

In [ ]:

```
# Box Plot
import seaborn as sns
sns.boxplot(df_4['new_duration'])
```

Black Mirror: Bandersnatch Movie was found to be having the runtime recorded as >300Min which is incorrect. With the outlier analysis it was detected, and can either be corrected by changing the value to 90 Min (actual runtime) or otherwise.

If this is due to error in data collection then no immediate actions required, If the data is misrepresented on the portal as well then immediate action required for correction.

In [ ]:

```
# Box Plot
import seaborn as sns
sns.boxplot(df_4['year_added'])
```

In [ ]:

```
# Box Plot
import seaborn as sns
sns.boxplot(df_4['release_year'])
```

In [ ]:

```
# Box Plot
import seaborn as sns
value_counts=df_1['rating'].value_counts().sort_values(ascending=False)
value_counts.head(15)
orig_df = pd.read_csv("/content/sample_data/netflix.csv")
print(orig_df['rating'].unique())
print(df_1['rating'].unique())
```

In [ ]:

```
# Validating that no Null values in any feature
df.isnull().sum()
```

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

--> Relevant comments are available with the Graphs

7. Business Insights (10 Points) -

USA and India are content volume wise the top most countries for Netflix business. The International content has huge presence in Indian market. Netflix hax very less presence in the third world contires and potential markets can be studied for the growth. The Anime Genre is to be evaluated for the growth in the non east-asian markets.

8. Recommendations (10 Points) -

1. As India has relatively very less Anime content available on the portal, keeping the increasing demand in the young crowd in india the Anime content should be made available in larger proportion
2. The third world contries can be studied to anlyze the potential to ensure earlier market penetration.
3. The 17+ content is available in more quantity. As the parental control and safety features are added the content catering to yonger audience shall be made avaialble in larger quantity
4. The TV Show content volume is smaller compared to the Movies. As the continuous audience captivation is to be achieved its recommended that the TV Shows and similar content to be made avaialble in larger quantity as new and refreshing content will be helpful in captivating more audience.

In [ ]:

```

```