1. Defining Problem Statement and Analysing basic metrics

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

In [363…

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

In [364…

```
url='https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/000/940/original/netflix.csv'
df=pd.read_csv(url,na_values='NaN')
```

In [365…

```
# Basic metrics
df.head()
```

Out[365]:

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

In [366…

```
df.tail()
```

Out[366]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
8802	s8803	Movie	Zodiac	David Fincher	Mark Ruffalo, Jake Gyllenhaal, Robert Downey J...	United States	November 20, 2019	2007	R	158 min	Cult Movies, Dramas, Thrillers	A political cartoonist, a crime reporter and a...
8803	s8804	TV Show	Zombie Dumb	NaN	NaN	NaN	July 1, 2019	2018	TV-Y7	2 Seasons	Kids' TV, Korean TV Shows, TV Comedies	While living alone in a spooky town, a young g...
8804	s8805	Movie	Zombieland	Ruben Fleischer	Jesse Eisenberg, Woody Harrelson, Emma Stone, ...	United States	November 1, 2019	2009	R	88 min	Comedies, Horror Movies	Looking to survive in a world taken over by zo...
8805	s8806	Movie	Zoom	Peter Hewitt	Tim Allen, Courteney Cox, Chevy Chase, Kate Ma...	United States	January 11, 2020	2006	PG	88 min	Children & Family Movies, Comedies	Dragged from civilian life, a former superhero...
8806	s8807	Movie	Zubaan	Mozez Singh	Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanan...	India	March 2, 2019	2015	TV-14	111 min	Dramas, International Movies, Music & Musicals	A scrappy but poor boy worms his way into a ty...

In [367…

```
# Column names
df.columns
```

Out[367]:

```
Index(['show_id', 'type', 'title', 'director', 'cast', 'country', 'date_added',
       'release_year', 'rating', 'duration', 'listed_in', 'description'],
      dtype='object')
```

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.

In [368…

```
# Shape of the data
df.shape
```

Out[368]:

```
(8807, 12)
```

In [369…

```
# Datatypes of all the attributes with null value detection
df.info()
```

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

In [370…

```
# Statistical Summary of Data
df.describe(include='int')
```

Out[370]:

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

In [371…

```
# Statistical Summary of Data
df.describe(include='object')
```

Out[371]:

	show_id	type	title	director	cast	country	date_added	rating	duration	listed_in	description
count	8807	8807	8807	6173	7982	7976	8797	8803	8804	8807	8807
unique	8807	2	8807	4528	7692	748	1767	17	220	514	8775
top	s1	Movie	Dick Johnson Is Dead	Rajiv Chilaka	David Attenborough	United States	January 1, 2020	TV-MA	1 Season	Dramas, International Movies	Paranormal activity at a lush, abandoned prope...
freq	1	6131	1	19	19	2818	109	3207	1793	362	4

In [372…

```
# Column wise null values
df.isna().sum()
```

Out[372]:

```
show_id            0
type               0
title              0
director        2634
cast             825
country          831
date_added        10
release_year       0
rating             4
duration           3
listed_in          0
description        0
dtype: int64
```

In [373…

```
# Null values of duration column
df.loc[df['duration'].isna()]
```

Out[373]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
5541	s5542	Movie	Louis C.K. 2017	Louis C.K.	Louis C.K.	United States	April 4, 2017	2017	74 min	NaN	Movies	Louis C.K. muses on religion, eternal love, gi...
5794	s5795	Movie	Louis C.K.: Hilarious	Louis C.K.	Louis C.K.	United States	September 16, 2016	2010	84 min	NaN	Movies	Emmy-winning comedy writer Louis C.K. brings h...
5813	s5814	Movie	Louis C.K.: Live at the Comedy Store	Louis C.K.	Louis C.K.	United States	August 15, 2016	2015	66 min	NaN	Movies	The comic puts his trademark hilarious/thought...

As we observe above three duration column values have been interchanged with rating column values, rectified the same correction in the below code.

In [374...

```
df['duration'][df['duration'].isna()] = df['rating'][df['duration'].isna()]
df['duration']=df['duration'].apply(lambda x:str(x).split()[0])
df['duration']=df['duration'].astype(int)
df
```

```
C:\Users\Sadiq\AppData\Local\Temp\ipykernel_11692\3956682623.py:1: 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
  df['duration'][df['duration'].isna()] = df['rating'][df['duration'].isna()]
```

Out[374]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG-13	90	Documentaries	As her father nears the end of his life, filmm...
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	2021	TV-MA	2	International TV Shows, TV Dramas, TV Mysteries	After crossing paths at a party, a Cape Town t...
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	September 24, 2021	2021	TV-MA	1	Crime TV Shows, International TV Shows, TV Act...	To protect his family from a powerful drug lor...
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	NaN	September 24, 2021	2021	TV-MA	1	Docuseries, Reality TV	Feuds, flirtations and toilet talk go down amo...
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	September 24, 2021	2021	TV-MA	2	International TV Shows, Romantic TV Shows, TV ...	In a city of coaching centers known to train l...
...	...	...	...	...	...	...	...	...	...	...	...	...
8802	s8803	Movie	Zodiac	David Fincher	Mark Ruffalo, Jake Gyllenhaal, Robert Downey J...	United States	November 20, 2019	2007	R	158	Cult Movies, Dramas, Thrillers	A political cartoonist, a crime reporter and a...
8803	s8804	TV Show	Zombie Dumb	NaN	NaN	NaN	July 1, 2019	2018	TV-Y7	2	Kids' TV, Korean TV Shows, TV Comedies	While living alone in a spooky town, a young g...
8804	s8805	Movie	Zombieland	Ruben Fleischer	Jesse Eisenberg, Woody Harrelson, Emma Stone, ...	United States	November 1, 2019	2009	R	88	Comedies, Horror Movies	Looking to survive in a world taken over by zo...
8805	s8806	Movie	Zoom	Peter Hewitt	Tim Allen, Courteney Cox, Chevy Chase, Kate Ma...	United States	January 11, 2020	2006	PG	88	Children & Family Movies, Comedies	Dragged from civilian life, a former superhero...
8806	s8807	Movie	Zubaan	Mozez Singh	Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanan...	India	March 2, 2019	2015	TV-14	111	Dramas, International Movies, Music & Musicals	A scrappy but poor boy worms his way into a ty...

8807 rows × 12 columns

In [375…

```
df.isna().sum() #As a result, duration column has no null values.
```

Out[375]:

```
show_id            0
type               0
title              0
director        2634
cast             825
country          831
date_added        10
release_year       0
rating             4
duration           0
listed_in          0
description        0
dtype: int64
```

In [545…

```
#conversion of categorical attributes to 'category'
df['date_added'] = pd.to_datetime(df["date_added"])
```

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

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

6. Insights based on Non-Graphical and Visual Analysis with comments.

In [376…

```
# column wise unique count
df.nunique()
```

Out[376]:

```
show_id         8807
type               2
title           8807
director        4528
cast            7692
country          748
date_added      1767
release_year      74
rating            17
duration         210
listed_in        514
description     8775
dtype: int64
```

In [377…

```
df['type'].nunique()
```

Out[377]:

```
2
```

In [378…

```
df['type'].value_counts()
```

Out[378]:

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

In [379…

```
df[df['type']=='Movie']['duration'].describe()
```

Out[379]:

```
count    6131.000000
mean       99.564998
std        28.289504
min         3.000000
25%        87.000000
50%        98.000000
75%       114.000000
max       312.000000
Name: duration, dtype: float64
```

Unnesting of cast column

In [380…

```
cast_data=df[["title","cast"]]
cast_data["cast"]=cast_data["cast"].str.split(", ")
df_cast=cast_data.explode("cast")
df_cast.replace("nan",
          np.nan, inplace=True)
df_cast.dropna(inplace=True)
df_cast
```

```
C:\Users\Sadiq\AppData\Local\Temp\ipykernel_11692\768506992.py: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
  cast_data["cast"]=cast_data["cast"].str.split(", ")
```

Out[380]:

	title	cast
1	Blood & Water	Ama Qamata
1	Blood & Water	Khosi Ngema
1	Blood & Water	Gail Mabalane
1	Blood & Water	Thabang Molaba
1	Blood & Water	Dillon Windvogel
...	...	...
8806	Zubaan	Manish Chaudhary
8806	Zubaan	Meghna Malik
8806	Zubaan	Malkeet Rauni
8806	Zubaan	Anita Shabdish
8806	Zubaan	Chittaranjan Tripathy

64126 rows × 2 columns

In [381…

```
# Value counts of cast column
df_cast['cast'].value_counts()
```

Out[381]:

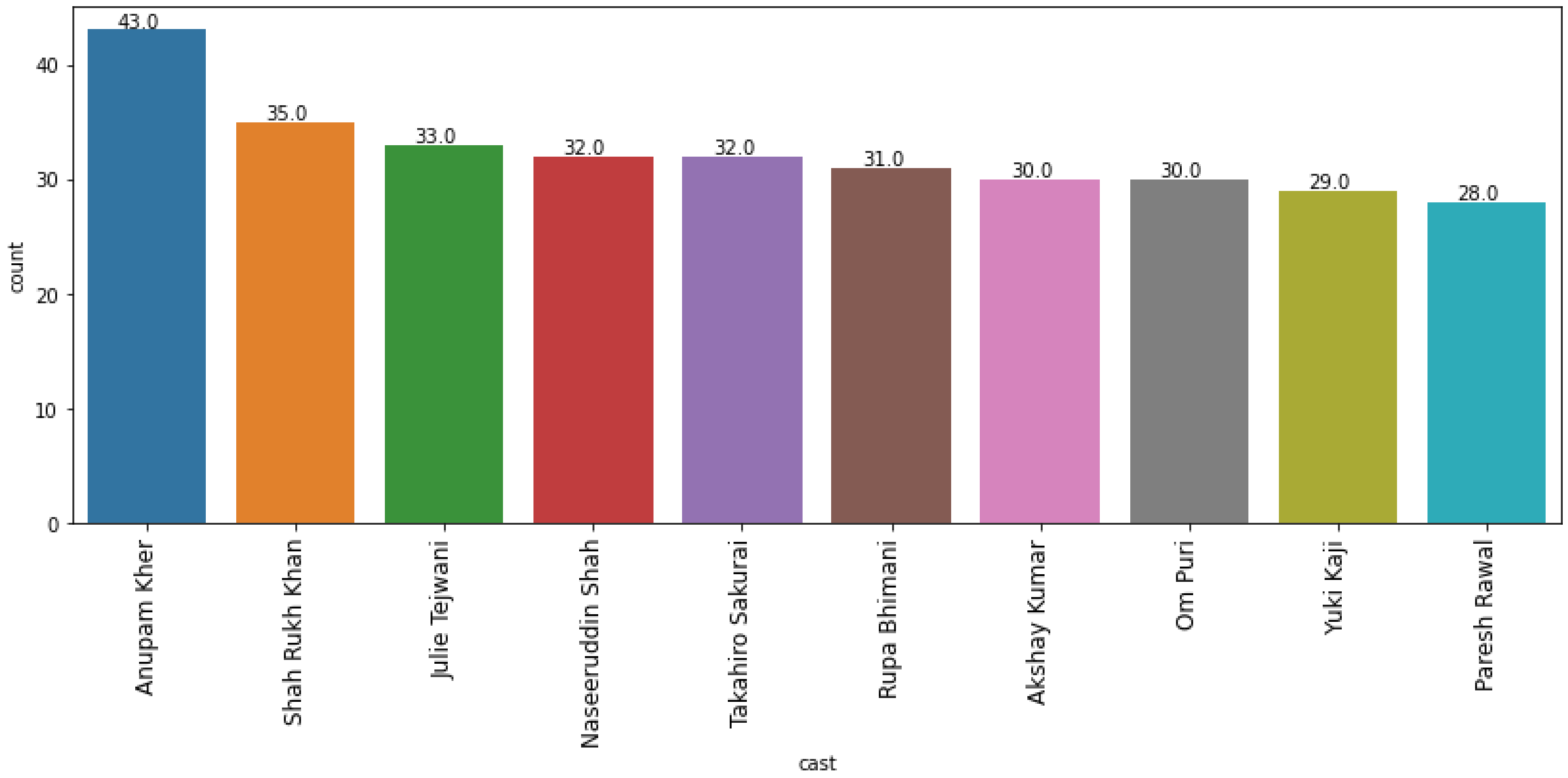
```
Anupam Kher                  43
Shah Rukh Khan               35
Julie Tejwani                33
Naseeruddin Shah             32
Takahiro Sakurai             32
                             ..
Maryam Zaree                  1
Melanie Straub                1
Gabriela Maria Schmeide       1
Helena Zengel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

Uni-variate Analysis

In [382…

```
fig=plt.figure(figsize=(14,5))

a=sns.countplot(x=df_cast["cast"],
 order=df_cast["cast"].value_counts().index[0:10])
plt.xticks(rotation=90,fontsize=12)
for p in a.patches:
   a.annotate('{:.1f}'.format(p.get_height()), (p.get_x()+0.20, p.get_height()+0.25))
plt.show()
```



Comment: Anupam Kher has acted in highest number of Netflix(Movies/TV shows)

In [383…

```
# Unique count of cast
df_cast.nunique()
```

Out[383]:

```
title     7982
cast     36439
dtype: int64
```

In [384…

```
# Total count
df_cast.value_counts().sum()
```

Out[384]:

```
64126
```

Unnesting of director column

In [385…

```
dir_data=df[["title","director"]]
dir_data["director"]=dir_data["director"].str.split(", ")
df_dir=dir_data.explode("director")
df_dir.replace("nan",
          np.nan, inplace=True)
df_dir.dropna(inplace=True)
df_dir
```

```
C:\Users\Sadiq\AppData\Local\Temp\ipykernel_11692\2311497627.py: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
  dir_data["director"]=dir_data["director"].str.split(", ")
```

Out[385]:

	title	director
0	Dick Johnson Is Dead	Kirsten Johnson
2	Ganglands	Julien Leclercq
5	Midnight Mass	Mike Flanagan
6	My Little Pony: A New Generation	Robert Cullen
6	My Little Pony: A New Generation	José Luis Ucha
...	...	...
8801	Zinzana	Majid Al Ansari
8802	Zodiac	David Fincher
8804	Zombieland	Ruben Fleischer
8805	Zoom	Peter Hewitt
8806	Zubaan	Mozez Singh

6978 rows × 2 columns

In [386…

```
# Value counts of director
df_dir['director'].value_counts()
```

Out[386]:

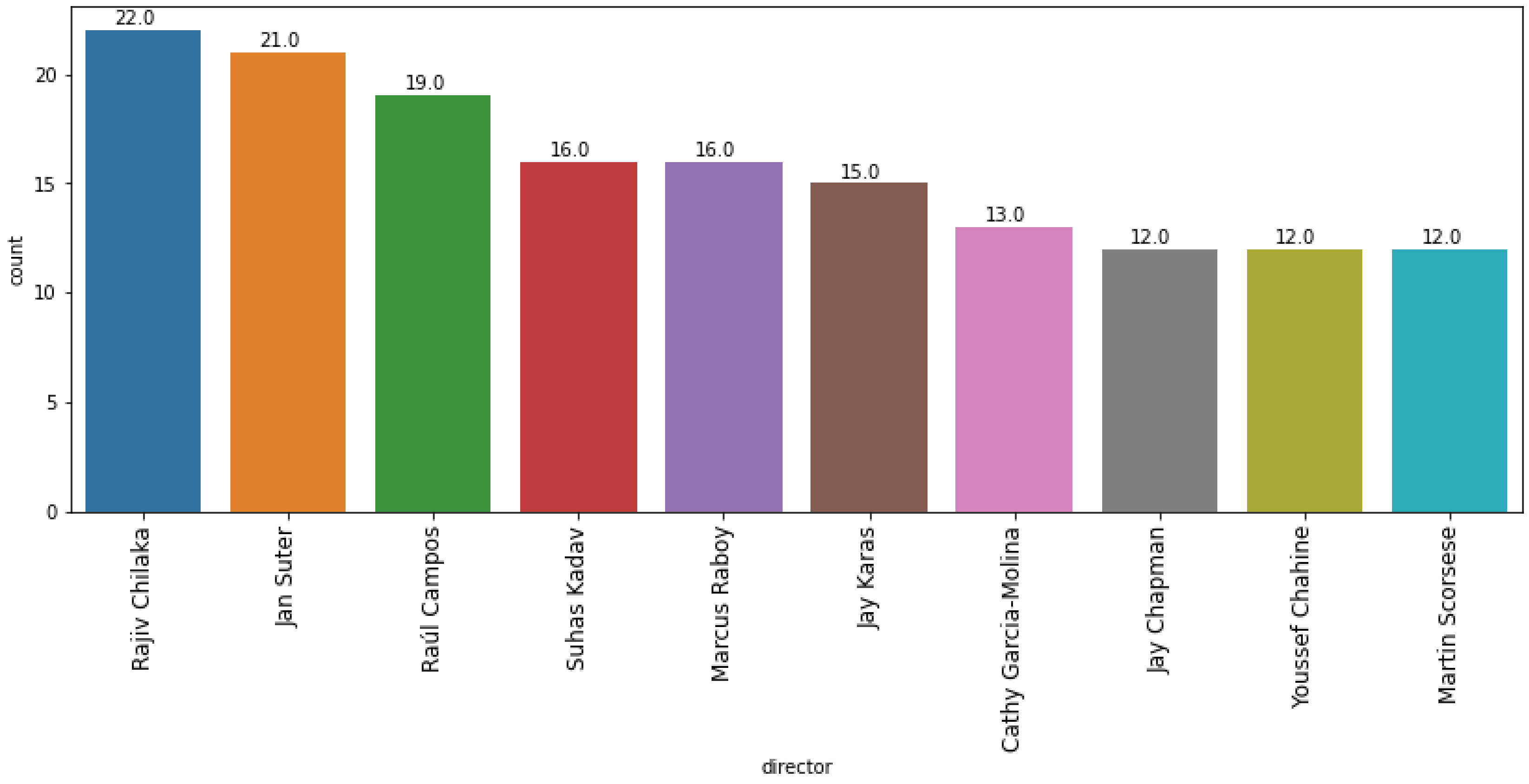
```
Rajiv Chilaka     22
Jan Suter         21
Raúl Campos       19
Suhas Kadav       16
Marcus Raboy      16
                  ..
Raymie Muzquiz     1
Stu Livingston     1
Joe Menendez       1
Eric Bross         1
Mozez Singh        1
Name: director, Length: 4993, dtype: int64
```

Uni-variate Analysis

In [387…

```
fig=plt.figure(figsize=(14,5))

a=sns.countplot(x=df_dir["director"],
 order=df_dir["director"].value_counts().index[0:10])
plt.xticks(rotation=90,fontsize=12)
for p in a.patches:
   a.annotate('{:.1f}'.format(p.get_height()), (p.get_x()+0.20, p.get_height()+0.25))
plt.show()
```



Comment- Rajiv Chilaka has been directed highest number of Netflix(Movies/ TV Shows).

In [388…

```
# Unique count of director
df_dir.nunique()
```

Out[388]:

```
title       6173
director    4993
dtype: int64
```

In [389…

```
# Total count
df_dir.value_counts().sum()
```

Out[389]:

```
6978
```

Unnesting of country column

In [390…

```python
cy_data=df[["title","country"]]
cy_data["country"]=cy_data["country"].str.split(", ")
df_cy=cy_data.explode("country")
df_cy.replace("nan",
          np.nan, inplace=True)
df_cy.dropna(inplace=True)
df_cy
```

```
C:\Users\Sadiq\AppData\Local\Temp\ipykernel_11692\3787550946.py: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
  cy_data["country"]=cy_data["country"].str.split(", ")
```

Out[390]:

	title	country
0	Dick Johnson Is Dead	United States
1	Blood & Water	South Africa
4	Kota Factory	India
7	Sankofa	United States
7	Sankofa	Ghana
...	...	...
8801	Zinzana	Jordan
8802	Zodiac	United States
8804	Zombieland	United States
8805	Zoom	United States
8806	Zubaan	India

10014 rows × 2 columns

In [391…

```python
# Value counts of country
df_cy['country'].value_counts()
```

Out[391]:

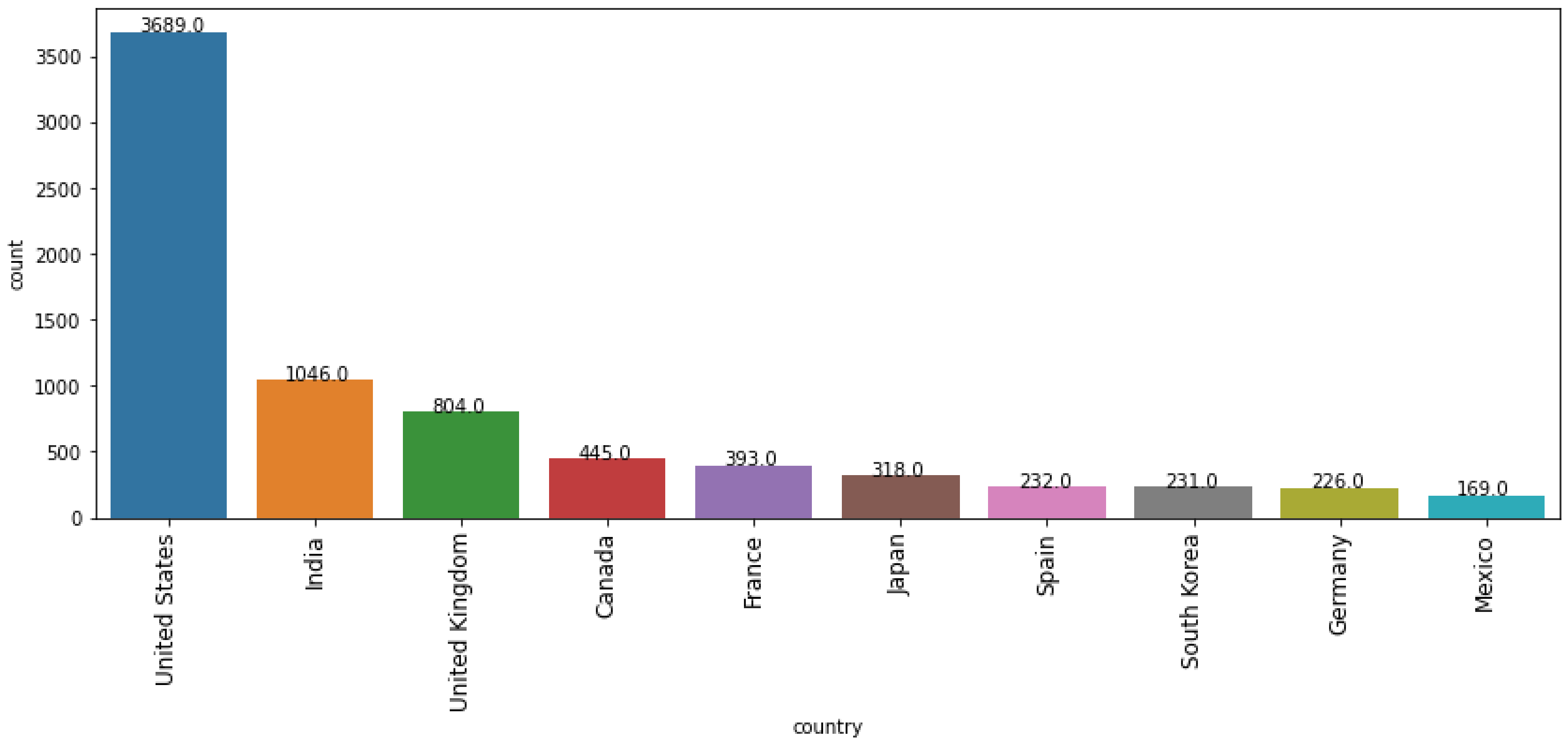
```
United States     3689
India             1046
United Kingdom     804
Canada             445
France             393
                  ...
Bermuda              1
Ecuador              1
Armenia              1
Mongolia             1
Montenegro           1
Name: country, Length: 127, dtype: int64
```

Uni-variate Analysis

In [392…

```python
fig=plt.figure(figsize=(14,5))

a=sns.countplot(x=df_cy["country"],
 order=df_cy["country"].value_counts().index[0:10])
plt.xticks(rotation=90,fontsize=12)
for p in a.patches:
   a.annotate('{:.1f}'.format(p.get_height()), (p.get_x()+0.20, p.get_height()+0.45))
plt.show()
```



Comment- United States has highest number of Netflix(Movies/ TV Shows).

In [393…

```
df_cy['country'].value_counts()
```

Out[393]:

```
United States     3689
India             1046
United Kingdom     804
Canada             445
France             393
                  ... 
Bermuda              1
Ecuador              1
Armenia              1
Mongolia             1
Montenegro           1
Name: country, Length: 127, dtype: int64
```

In [394…

```
# Unique values of country
df_cy['country'].unique()
```

Out[394]:

```
array(['United States', 'South Africa', 'India', 'Ghana', 'Burkina Faso',
       'United Kingdom', 'Germany', 'Ethiopia', 'Czech Republic',
       'Mexico', 'Turkey', 'Australia', 'France', 'Finland', 'China',
       'Canada', 'Japan', 'Nigeria', 'Spain', 'Belgium', 'South Korea',
       'Singapore', 'Italy', 'Romania', 'Argentina', 'Venezuela',
       'Hong Kong', 'Russia', '', 'Ireland', 'Nepal', 'New Zealand',
       'Brazil', 'Greece', 'Jordan', 'Colombia', 'Switzerland', 'Israel',
       'Taiwan', 'Bulgaria', 'Algeria', 'Poland', 'Saudi Arabia',
       'Thailand', 'Indonesia', 'Egypt', 'Denmark', 'Kuwait',
       'Netherlands', 'Malaysia', 'Vietnam', 'Hungary', 'Sweden',
       'Lebanon', 'Syria', 'Philippines', 'Iceland',
       'United Arab Emirates', 'Norway', 'Qatar', 'Mauritius', 'Austria',
       'Cameroon', 'Palestine', 'Uruguay', 'United Kingdom,', 'Kenya',
       'Chile', 'Luxembourg', 'Cambodia', 'Bangladesh', 'Portugal',
       'Cayman Islands', 'Senegal', 'Serbia', 'Malta', 'Namibia',
       'Angola', 'Peru', 'Mozambique', 'Cambodia,', 'Belarus', 'Zimbabwe',
       'Puerto Rico', 'Pakistan', 'Cyprus', 'Guatemala', 'Iraq', 'Malawi',
       'Paraguay', 'Croatia', 'Iran', 'West Germany', 'United States,',
       'Albania', 'Georgia', 'Soviet Union', 'Morocco', 'Slovakia',
       'Ukraine', 'Bermuda', 'Ecuador', 'Armenia', 'Mongolia', 'Bahamas',
       'Sri Lanka', 'Latvia', 'Liechtenstein', 'Cuba', 'Nicaragua',
       'Poland,', 'Slovenia', 'Dominican Republic', 'Samoa', 'Azerbaijan',
       'Botswana', 'Vatican City', 'Jamaica', 'Kazakhstan', 'Lithuania',
       'Afghanistan', 'Somalia', 'Sudan', 'Panama', 'Uganda',
       'East Germany', 'Montenegro'], dtype=object)
```

In [395…

```
# Unique count of country
df_cy.nunique()
```

Out[395]:

```
title      7976
country     127
dtype: int64
```

Unnesting of Listed_in column

In [396…

```
li_data=df[["title","listed_in"]]
li_data["listed_in"]=li_data["listed_in"].str.split(", ")
df_li=li_data.explode("listed_in")
df_li.replace("nan",
          np.nan, inplace=True)
df_li.dropna(inplace=True)
df_li
```

```
C:\Users\Sadiq\AppData\Local\Temp\ipykernel_11692\2075411862.py: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
  li_data["listed_in"]=li_data["listed_in"].str.split(", ")
```

Out[396]:

	title	listed_in
0	Dick Johnson Is Dead	Documentaries
1	Blood & Water	International TV Shows
1	Blood & Water	TV Dramas
1	Blood & Water	TV Mysteries
2	Ganglands	Crime TV Shows
...	...	...
8805	Zoom	Children & Family Movies
8805	Zoom	Comedies
8806	Zubaan	Dramas
8806	Zubaan	International Movies
8806	Zubaan	Music & Musicals

19323 rows × 2 columns

In [397…

```
# Value counts of listed_in
df_li['listed_in'].value_counts()
```

Out[397]:

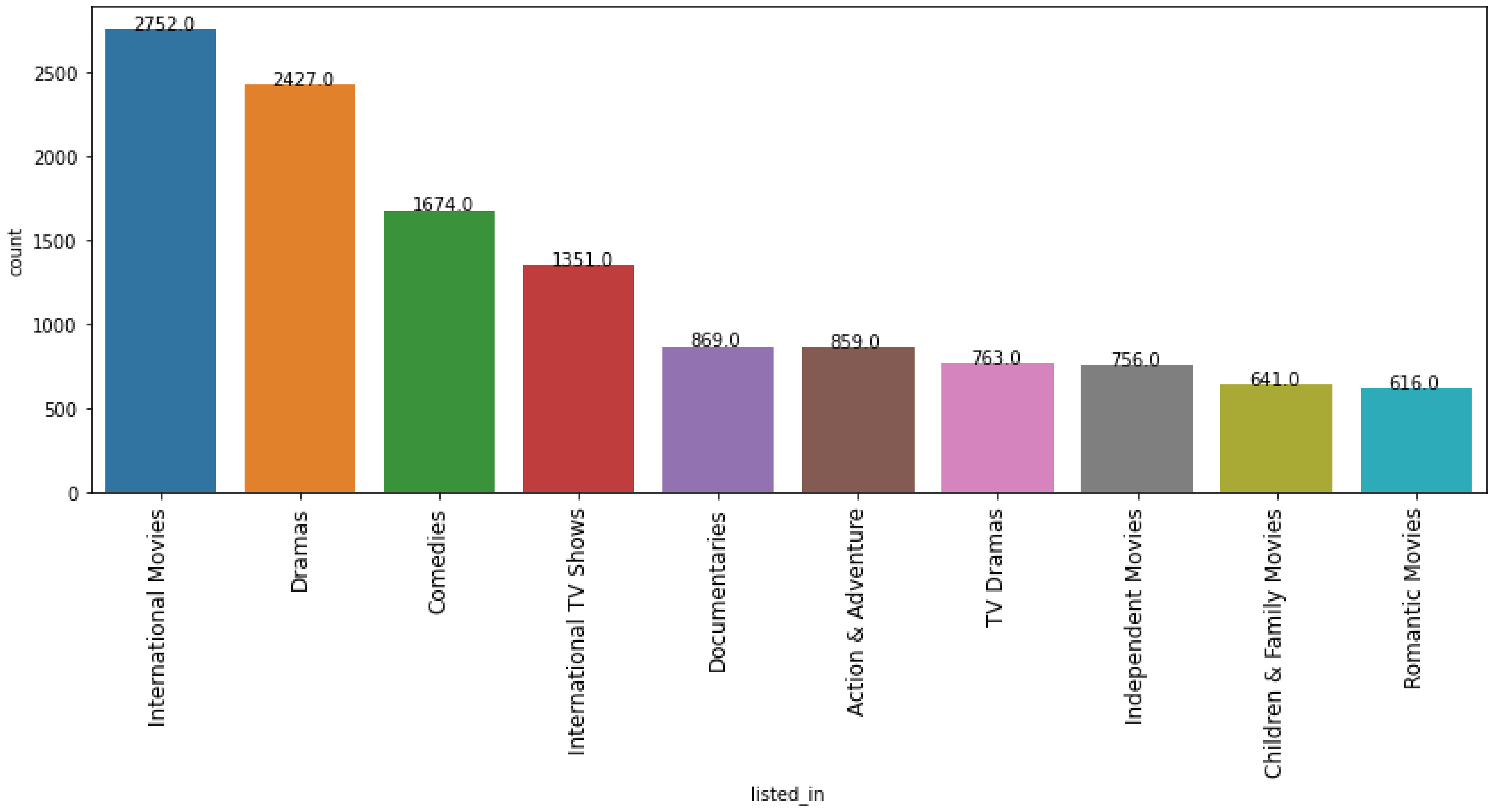
```
International Movies            2752
Dramas                          2427
Comedies                        1674
International TV Shows          1351
Documentaries                    869
Action & Adventure               859
TV Dramas                        763
Independent Movies               756
Children & Family Movies         641
Romantic Movies                  616
TV Comedies                      581
Thrillers                        577
Crime TV Shows                   470
Kids' TV                         451
Docuseries                       395
Music & Musicals                 375
Romantic TV Shows                370
Horror Movies                    357
Stand-Up Comedy                  343
Reality TV                       255
British TV Shows                 253
Sci-Fi & Fantasy                 243
Sports Movies                    219
Anime Series                     176
Spanish-Language TV Shows        174
TV Action & Adventure            168
Korean TV Shows                  151
Classic Movies                   116
LGBTQ Movies                     102
TV Mysteries                      98
Science & Nature TV               92
TV Sci-Fi & Fantasy               84
TV Horror                         75
Anime Features                    71
Cult Movies                       71
Teen TV Shows                     69
Faith & Spirituality              65
TV Thrillers                      57
Movies                            57
Stand-Up Comedy & Talk Shows      56
Classic & Cult TV                 28
TV Shows                          16
Name: listed_in, dtype: int64
```

Uni-variate Analysis

In [398…

```
fig=plt.figure(figsize=(14,5))

a=sns.countplot(x=df_li["listed_in"],
 order=df_li["listed_in"].value_counts().index[0:10])
plt.xticks(rotation=90,fontsize=12)
for p in a.patches:
   a.annotate('{:.1f}'.format(p.get_height()), (p.get_x()+0.20, p.get_height()+0.75))
plt.show()
```



Comment- The highest number of Netflix[Movies/TV shows] is listed in International Movies category.

In [399…

```
# Unique values of listed_in
df_li['listed_in'].unique()
```

Out[399]:

```
array(['Documentaries', 'International TV Shows', 'TV Dramas',
       'TV Mysteries', 'Crime TV Shows', 'TV Action & Adventure',
       'Docuseries', 'Reality TV', 'Romantic TV Shows', 'TV Comedies',
       'TV Horror', 'Children & Family Movies', 'Dramas',
       'Independent Movies', 'International Movies', 'British TV Shows',
       'Comedies', 'Spanish-Language TV Shows', 'Thrillers',
       'Romantic Movies', 'Music & Musicals', 'Horror Movies',
       'Sci-Fi & Fantasy', 'TV Thrillers', "Kids' TV",
       'Action & Adventure', 'TV Sci-Fi & Fantasy', 'Classic Movies',
       'Anime Features', 'Sports Movies', 'Anime Series',
       'Korean TV Shows', 'Science & Nature TV', 'Teen TV Shows',
       'Cult Movies', 'TV Shows', 'Faith & Spirituality', 'LGBTQ Movies',
       'Stand-Up Comedy', 'Movies', 'Stand-Up Comedy & Talk Shows',
       'Classic & Cult TV'], dtype=object)
```

In [400…

```
# Unique count of listed in column
df_li.nunique()
```

Out[400]:

```
title        8807
listed_in      42
dtype: int64
```

In [401…

```
#Total count
df_li.value_counts().sum()
```

Out[401]:

```
19323
```

In [402…

```
# Value counts of Rating
df['rating'].value_counts()
```

Out[402]:

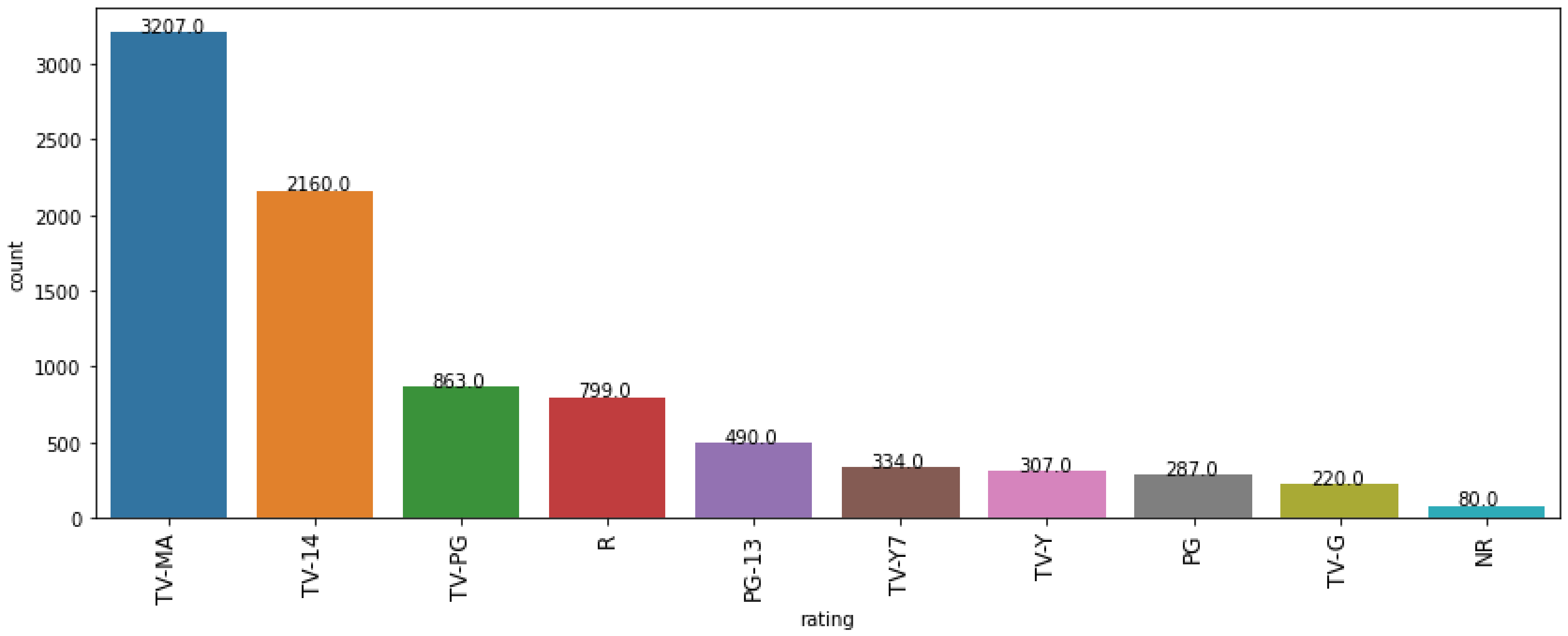
```
TV-MA       3207
TV-14       2160
TV-PG        863
R            799
PG-13        490
TV-Y7        334
TV-Y         307
PG           287
TV-G         220
NR            80
G             41
TV-Y7-FV       6
NC-17          3
UR             3
74 min         1
84 min         1
66 min         1
Name: rating, dtype: int64
```

Uni-Variate Analysis

In [403…

```
fig=plt.figure(figsize=(14,5))

a=sns.countplot(x=df["rating"],
 order=df["rating"].value_counts().index[0:10])
plt.xticks(rotation=90,fontsize=12)
for p in a.patches:
   a.annotate('{:.1f}'.format(p.get_height()), (p.get_x()+0.20, p.get_height()+2.95))
plt.show()
```



Comment- The highest rating of Netflix[Movies/TV Shows] is 'TV-MA'.

In [404…

```
# Unique values of Rating
df['rating'].unique()
```

Out[404]:

```
array(['PG-13', 'TV-MA', 'PG', 'TV-14', 'TV-PG', 'TV-Y', 'TV-Y7', 'R',
       'TV-G', 'G', 'NC-17', '74 min', '84 min', '66 min', 'NR', nan,
       'TV-Y7-FV', 'UR'], dtype=object)
```

In [405…

```
# Unique count of Rating
df['rating'].nunique()
```

Out[405]:

```
17
```

In [406…

```
# Value counts of release year
df['release_year'].value_counts()
```

Out[406]:

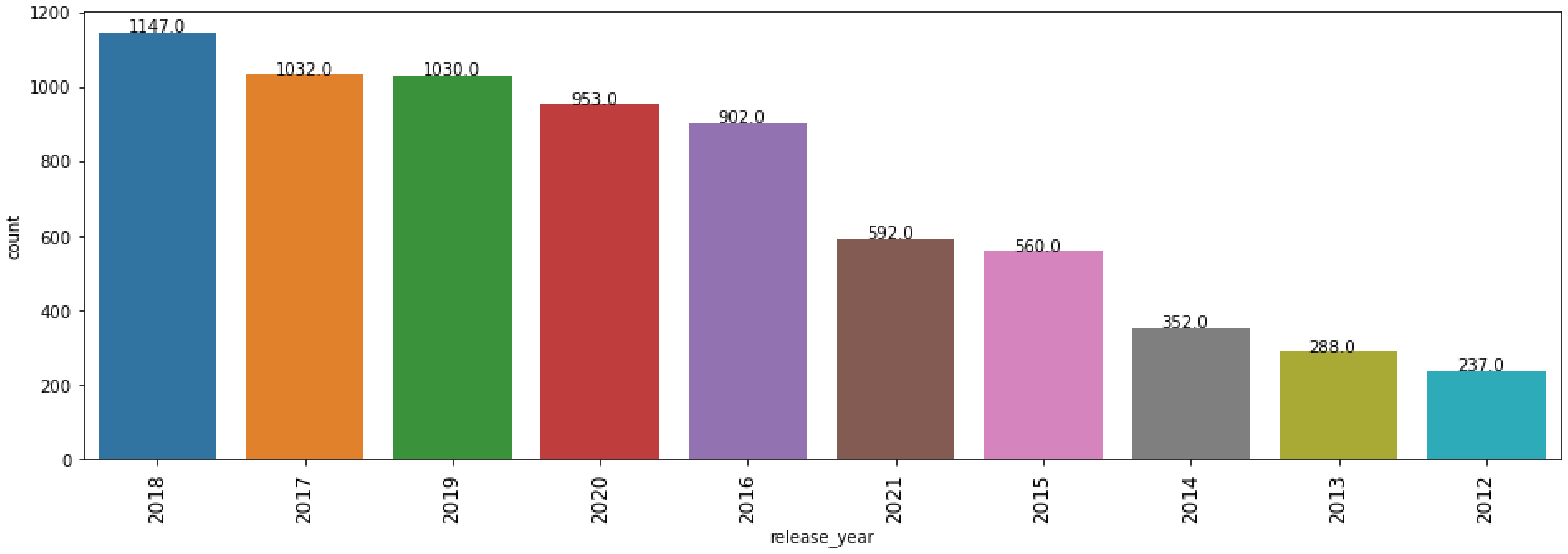
```
2018    1147
2017    1032
2019    1030
2020     953
2016     902
        ... 
1959       1
1925       1
1961       1
1947       1
1966       1
Name: release_year, Length: 74, dtype: int64
```

Uni-Variate Analysis

In [407…

```
fig=plt.figure(figsize=(16,5))

a=sns.countplot(x=df["release_year"],
 order=df["release_year"].value_counts().index[0:10])
plt.xticks(rotation=90,fontsize=12)
for p in a.patches:
   a.annotate('{:.1f}'.format(p.get_height()), (p.get_x()+0.20, p.get_height()+0.45))
plt.show()
```



Comment- The Highest number of Netflix[Movies/TV Shows] have been released on 2018 year.

In [408…

```
# Unique values of release_year
df['release_year'].unique()
```

Out[408]:

```
array([2020, 2021, 1993, 2018, 1996, 1998, 1997, 2010, 2013, 2017, 1975,
       1978, 1983, 1987, 2012, 2001, 2014, 2002, 2003, 2004, 2011, 2008,
       2009, 2007, 2005, 2006, 1994, 2015, 2019, 2016, 1982, 1989, 1990,
       1991, 1999, 1986, 1992, 1984, 1980, 1961, 2000, 1995, 1985, 1976,
       1959, 1988, 1981, 1972, 1964, 1945, 1954, 1979, 1958, 1956, 1963,
       1970, 1973, 1925, 1974, 1960, 1966, 1971, 1962, 1969, 1977, 1967,
       1968, 1965, 1946, 1942, 1955, 1944, 1947, 1943], dtype=int64)
```

In [409…

```python
# Unique count of release_year
df['release_year'].nunique()
```

Out[409]:

```
74
```

Unnesting of Title, Cast and Type

In [410…

```python
tc1=df[["title","cast","type"]]
tc1["cast"]=tc1["cast"].str.split(", ")
tc=tc1.explode("cast")
tc.replace("nan",
          np.nan, inplace=True)
tc.dropna(inplace=True)
tc
```

```
C:\Users\Sadiq\AppData\Local\Temp\ipykernel_11692\1882680494.py: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
  tc1["cast"]=tc1["cast"].str.split(", ")
```

Out[410]:

	title	cast	type
1	Blood & Water	Ama Qamata	TV Show
1	Blood & Water	Khosi Ngema	TV Show
1	Blood & Water	Gail Mabalane	TV Show
1	Blood & Water	Thabang Molaba	TV Show
1	Blood & Water	Dillon Windvogel	TV Show
...	...	...	...
8806	Zubaan	Manish Chaudhary	Movie
8806	Zubaan	Meghna Malik	Movie
8806	Zubaan	Malkeet Rauni	Movie
8806	Zubaan	Anita Shabdish	Movie
8806	Zubaan	Chittaranjan Tripathy	Movie

64126 rows × 3 columns

In [411…

```python
tc[tc['type']=='Movie']['cast'].value_counts()
```

Out[411]:

```
Anupam Kher                42
Shah Rukh Khan             35
Naseeruddin Shah           32
Akshay Kumar               30
Om Puri                    30
                           ..
Sushma Bakshi               1
Yusuf Hussain               1
Amarjeet Amle               1
Priya                       1
Chittaranjan Tripathy       1
Name: cast, Length: 25951, dtype: int64
```

In [412…

```python
tc[tc['type']=='Movie']['cast'].value_counts().sum()
```

Out[412]:

```
44475
```

In [413…

```python
tc[tc['type']=='TV Show']['cast'].value_counts()
```

Out[413]:

```
Takahiro Sakurai           25
Yuki Kaji                  19
Daisuke Ono                17
Junichi Suwabe             17
Ai Kayano                  17
                           ..
Bhumibhat Thavornsiri       1
Thanongsak Suphakan         1
Kanjanaporn Plodpai         1
Boonsong Nakphoo            1
Hina Khawaja Bayat          1
Name: cast, Length: 14863, dtype: int64
```

In [414…

```python
tc[tc['type']=='TV Show']['cast'].value_counts().sum()
```

Out[414]:

```
19651
```

In [415…

```
tc[tc['type']=='Movie']['cast'].unique()
```

Out[415]:

```
array(['Vanessa Hudgens', 'Kimiko Glenn', 'James Marsden', ...,
       'Malkeet Rauni', 'Anita Shabdish', 'Chittaranjan Tripathy'],
      dtype=object)
```

In [416…

```
tc[tc['type']=='Movie']['cast'].nunique()
```

Out[416]:

```
25951
```

In [417…

```
tc[tc['type']=='TV Show']['cast'].nunique()
```

Out[417]:

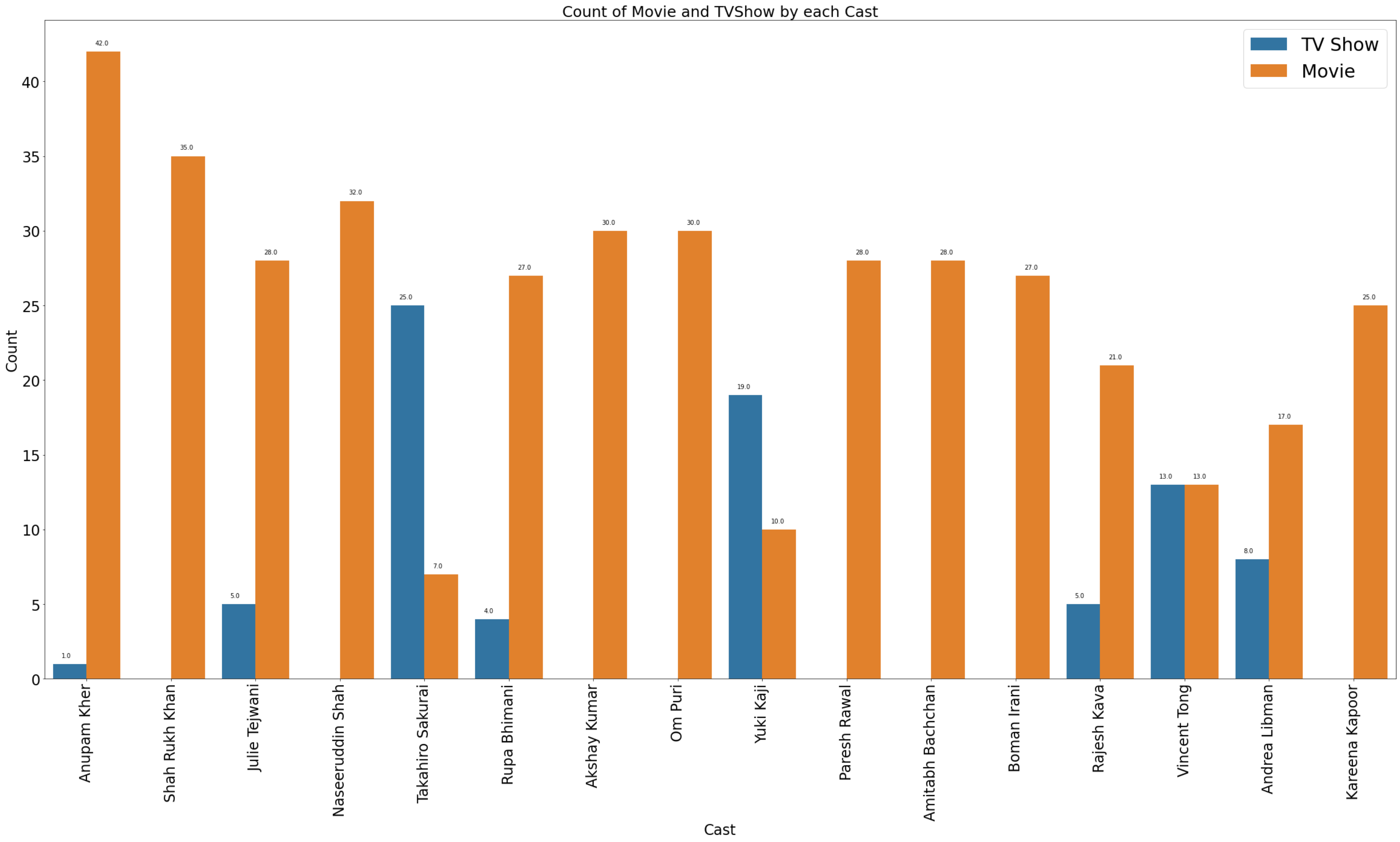
```
14863
```

Bi- Variate Analysis

In [418…

```
plt.figure(figsize=(40,20))
plt.title('Count of Movie and TVShow by each Cast', fontsize=25)
a=sns.countplot(x="cast", data=tc, order=tc['cast'].value_counts().index[0:16],hue="type")
plt.xticks(rotation=90,fontsize=24)
plt.yticks(fontsize=24)
plt.xlabel('Cast',size=24)
plt.ylabel('Count',size=24)
plt.legend(loc="upper right", frameon=True, fontsize=30)
for p in a.patches:
   a.annotate('{:.1f}'.format(p.get_height()), (p.get_x()+0.10, p.get_height()+0.45))

plt.show()
```



Comment- The highest number of movies has been casted by 'Anupam Kher' where as the highest number of TV shows has been casted by 'Takahiro Sakurai'.

Unnesting of Title, Director and Type

In [419…

```
td1=df[["title","director","type"]]
td1["director"]=td1["director"].str.split(", ")
td=td1.explode("director")
td.replace("nan",
          np.nan, inplace=True)
td.dropna(inplace=True)
td
```

```
C:\Users\Sadiq\AppData\Local\Temp\ipykernel_11692\496677526.py: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
  td1["director"]=td1["director"].str.split(", ")
```

Out[419]:

	title	director	type
0	Dick Johnson Is Dead	Kirsten Johnson	Movie
2	Ganglands	Julien Leclercq	TV Show
5	Midnight Mass	Mike Flanagan	TV Show
6	My Little Pony: A New Generation	Robert Cullen	Movie
6	My Little Pony: A New Generation	José Luis Ucha	Movie
...	...	...	...
8801	Zinzana	Majid Al Ansari	Movie
8802	Zodiac	David Fincher	Movie
8804	Zombieland	Ruben Fleischer	Movie
8805	Zoom	Peter Hewitt	Movie
8806	Zubaan	Mozez Singh	Movie

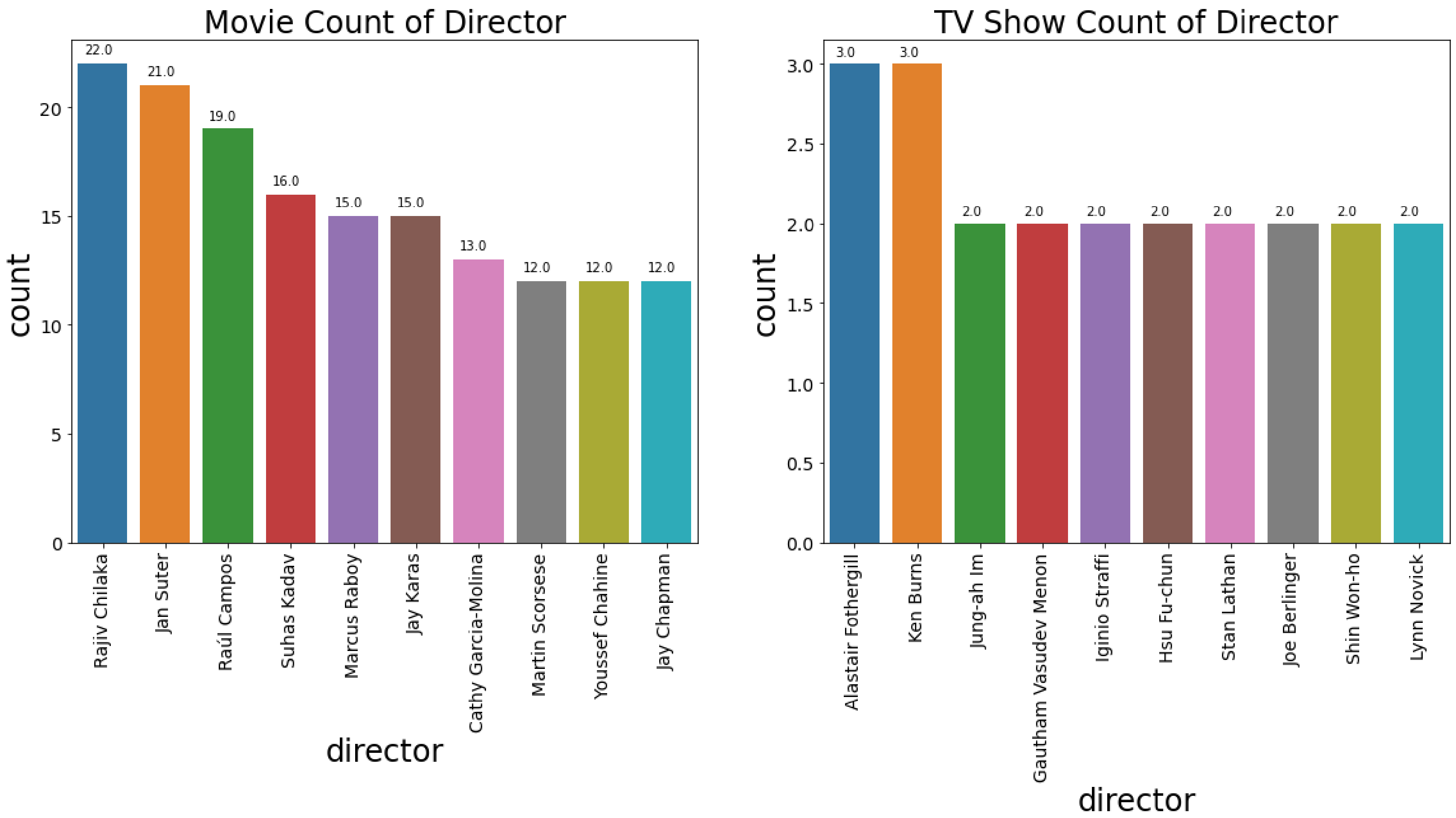
6978 rows × 3 columns

In [420…
```
td[td['type']=='Movie']['director'].value_counts()
```
Out[420]:
```
Rajiv Chilaka        22
Jan Suter            21
Raúl Campos          19
Suhas Kadav          16
Marcus Raboy         15
                     ..
Vrinda Samartha       1
Nicholaus Goossen     1
Stig Bergqvist        1
Paul Demeyer          1
Mozez Singh           1
Name: director, Length: 4777, dtype: int64
```
In [421…
```
td[td['type']=='Movie']['director'].value_counts().sum()
```
Out[421]:
```
6666
```
In [422…
```
td[td['type']=='TV Show']['director'].value_counts()
```
Out[422]:
```
Alastair Fothergill       3
Ken Burns                 3
Jung-ah Im                2
Gautham Vasudev Menon     2
Iginio Straffi            2
                         ..
Jesse Vile                1
Ellena Wood               1
Picky Talarico            1
Pedro Waddington          1
Michael Cumming           1
Name: director, Length: 299, dtype: int64
```
In [423…
```
td[td['type']=='TV Show']['director'].value_counts().sum()
```
Out[423]:
```
312
```
In [424…
```
td[td['type']=='Movie']['director'].nunique()
```
Out[424]:
```
4777
```
In [425…
```
td[td['type']=='TV Show']['director'].nunique()
```
Out[425]:
```
299
```

Bi- Variate Analysis

In [426…
```
plt.figure(figsize=(30,16))
plt.subplot(2, 3, 1)

plt.xlabel("Director")
plt.ylabel(" Movie count")
plt.xticks(rotation=90,fontsize=14)
plt.yticks(fontsize=14)
plt.xlabel('Director',size=24)
plt.ylabel('Count',size=24)
plt.title("Movie Count of Director", fontsize=24);
a=sns.countplot(x="director", data=td[td["type"]=="Movie"], order=td[td['type']=='Movie']['director'].value_counts().in
for p in a.patches:
   a.annotate('{:.1f}'.format(p.get_height()), (p.get_x()+0.10, p.get_height()+0.45))
plt.subplot(2, 3, 2)
plt.xlabel("Director")
plt.ylabel("TV Show")
plt.xlabel('Director',size=24)
plt.ylabel('Count',size=24)
plt.xticks(rotation=90,fontsize=14)
plt.yticks(fontsize=14)
plt.title("TV Show Count of Director",fontsize=24);
b=sns.countplot(x="director", data=td[td["type"]=="TV Show"], order=td[td['type']=='TV Show']['director'].value_counts(
for p in b.patches:
    b.annotate('{:.1f}'.format(p.get_height()), (p.get_x()+0.10, p.get_height()+0.05))
plt.show()
```



Comment- The highest number of Movies has been directed by 'Rajiv Chilaka' whereas highest number of TV shows have been directed by 'Alastair Fothergill' and 'Ken Burns'.

Unnesting of Title, Country and Type

In [427…

```
tcy1=df[["title","country","type"]]
tcy1["country"]=tcy1["country"].str.split(", ")
tcy=tcy1.explode("country")
tcy.replace("nan",
            np.nan, inplace=True)
tcy.dropna(inplace=True)
tcy
```

```
C:\Users\Sadiq\AppData\Local\Temp\ipykernel_11692\792483585.py: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
  tcy1["country"]=tcy1["country"].str.split(", ")
```

Out[427]:

	title	country	type
0	Dick Johnson Is Dead	United States	Movie
1	Blood & Water	South Africa	TV Show
4	Kota Factory	India	TV Show
7	Sankofa	United States	Movie
7	Sankofa	Ghana	Movie
...	...	...	...
8801	Zinzana	Jordan	Movie
8802	Zodiac	United States	Movie
8804	Zombieland	United States	Movie
8805	Zoom	United States	Movie
8806	Zubaan	India	Movie

10014 rows × 3 columns

In [428…

```
tcy[tcy['type']=='Movie']['country'].value_counts()
```

Out[428]:

```
United States     2751
India              962
United Kingdom     532
Canada             319
France             303
                  ... 
Bermuda              1
Angola               1
Armenia              1
Mongolia             1
Montenegro           1
Name: country, Length: 122, dtype: int64
```

In [429…

```
tcy[tcy['type']=='Movie']['country'].value_counts().sum()
```

Out[429]:

```
7374
```

In [430…

```
tcy[tcy['type']=='TV Show']['country'].value_counts()
```

Out[430]:

```
United States           938
United Kingdom          272
Japan                   199
South Korea             170
Canada                  126
                       ... 
Malta                     1
Belarus                   1
United Arab Emirates      1
Uruguay                   1
Switzerland               1
Name: country, Length: 66, dtype: int64
```

In [431…

```
tcy[tcy['type']=='TV Show']['country'].value_counts().sum()
```

Out[431]:

```
2640
```

In [432…

```
tcy[tcy['type']=='Movie']['country'].value_counts().sum()
```

Out[432]:

```
7374
```

In [433…

```
tcy[tcy['type']=='Movie']['country'].nunique()
```

Out[433]:

```
122
```

In [434…

```
tcy[tcy['type']=='TV Show']['country'].nunique()
```

Out[434]:

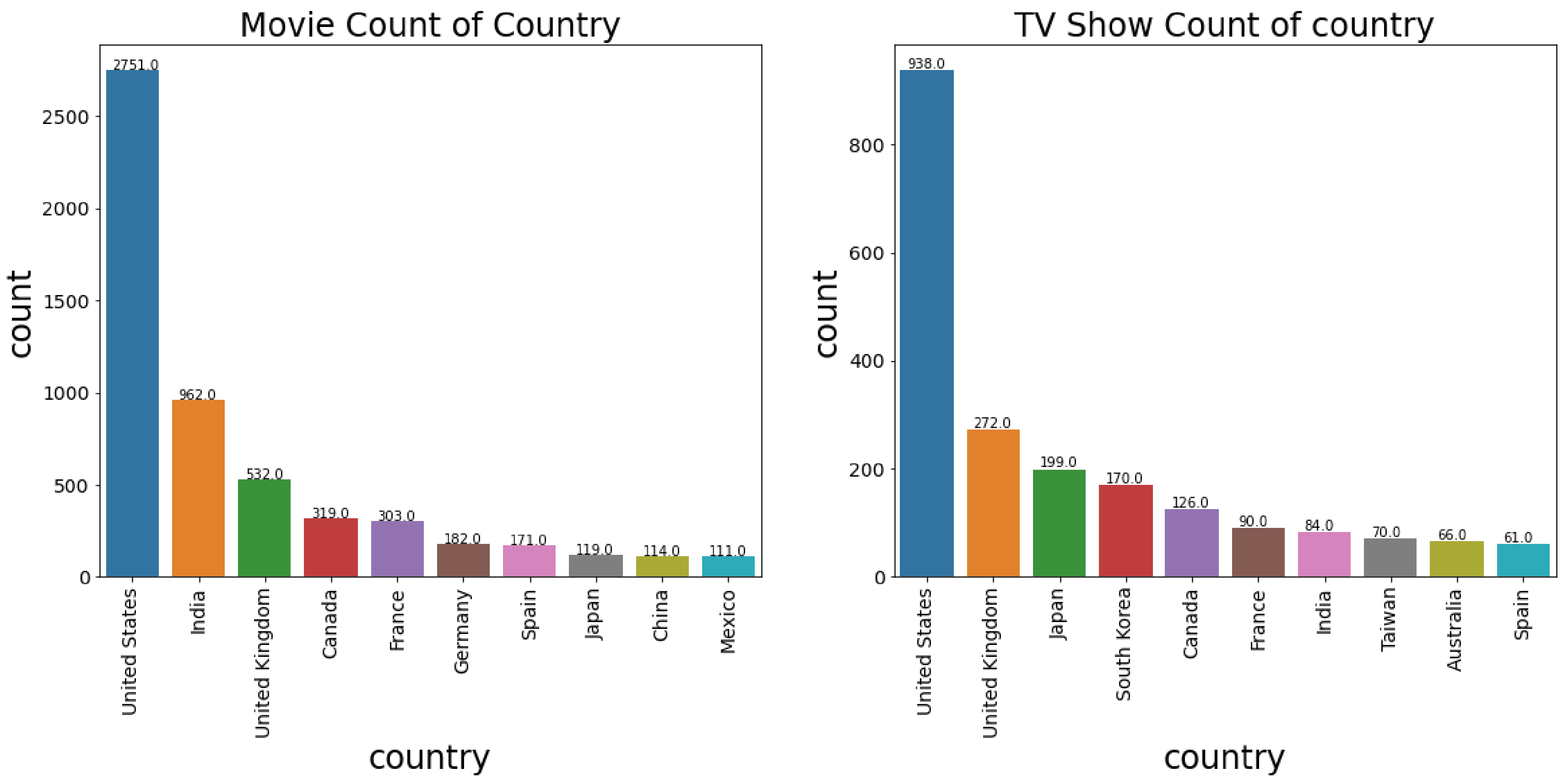
```
66
```

Bi- Variate Analysis

In [435…

```
plt.figure(figsize=(30,16))
plt.subplot(2, 3, 1)

plt.xlabel("country")
plt.ylabel(" Movie count")
plt.xticks(rotation=90,fontsize=14)
plt.yticks(fontsize=14)
plt.xlabel('country',size=24)
plt.ylabel('Count',size=24)
plt.title("Movie Count of Country", fontsize=24);
a=sns.countplot(x="country", data=tcy[tcy["type"]=="Movie"], order=tcy[tcy['type']=='Movie']['country'].value_counts().
for p in a.patches:
   a.annotate('{:.1f}'.format(p.get_height()), (p.get_x()+0.10, p.get_height()+0.45))
plt.subplot(2, 3, 2)
plt.xlabel("country")
plt.ylabel("TV Show")
plt.xlabel('country',size=24)
plt.ylabel('Count',size=24)
plt.xticks(rotation=90,fontsize=14)
plt.yticks(fontsize=14)
plt.title("TV Show Count of country",fontsize=24);
b=sns.countplot(x="country", data=tcy[tcy["type"]=="TV Show"], order=tcy[tcy['type']=='TV Show']['country'].value_count
for p in b.patches:
   b.annotate('{:.1f}'.format(p.get_height()), (p.get_x()+0.10, p.get_height()+2.95))
plt.show()
```



Comment- The highest number of Movies and TV Show in Netflix are from 'United States' country.

Unnesting of Title, Listed_in and Type

In [438…

```
tli1=df[["title","listed_in","type"]]
tli1["listed_in"]=tli1["listed_in"].str.split(", ")
tli=tli1.explode("listed_in")
tli.replace("nan",
          np.nan, inplace=True)
tli.dropna(inplace=True)
tli
```

```
C:\Users\Sadiq\AppData\Local\Temp\ipykernel_11692\2181864249.py: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
  tli1["listed_in"]=tli1["listed_in"].str.split(", ")
```

Out[438]:

	title	listed_in	type
0	Dick Johnson Is Dead	Documentaries	Movie
1	Blood & Water	International TV Shows	TV Show
1	Blood & Water	TV Dramas	TV Show
1	Blood & Water	TV Mysteries	TV Show
2	Ganglands	Crime TV Shows	TV Show
...	...	...	...
8805	Zoom	Children & Family Movies	Movie
8805	Zoom	Comedies	Movie
8806	Zubaan	Dramas	Movie
8806	Zubaan	International Movies	Movie
8806	Zubaan	Music & Musicals	Movie

19323 rows × 3 columns

In [439…

```
tli[tli['type']=='Movie']['listed_in'].value_counts()
```

Out[439]:

```
International Movies        2752
Dramas                      2427
Comedies                    1674
Documentaries                869
Action & Adventure           859
Independent Movies           756
Children & Family Movies     641
Romantic Movies              616
Thrillers                    577
Music & Musicals             375
Horror Movies                357
Stand-Up Comedy              343
Sci-Fi & Fantasy             243
Sports Movies                219
Classic Movies               116
LGBTQ Movies                 102
Anime Features                71
Cult Movies                   71
Faith & Spirituality          65
Movies                        57
Name: listed_in, dtype: int64
```

In [440…

```
tli[tli['type']=='Movie']['listed_in'].value_counts().sum()
```

Out[440]:

```
13190
```

In [442…

```
tli[tli['type']=='Movie']['listed_in'].unique()
```

Out[442]:

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

In [444…

```
tli[tli['type']=='Movie']['listed_in'].nunique()
```

Out[444]:

```
20
```

In [443…

```
tli[tli['type']=='TV Show']['listed_in'].value_counts()
```

Out[443]:

```
International TV Shows          1351
TV Dramas                        763
TV Comedies                      581
Crime TV Shows                   470
Kids' TV                         451
Docuseries                       395
Romantic TV Shows                370
Reality TV                       255
British TV Shows                 253
Anime Series                     176
Spanish-Language TV Shows        174
TV Action & Adventure            168
Korean TV Shows                  151
TV Mysteries                      98
Science & Nature TV               92
TV Sci-Fi & Fantasy               84
TV Horror                         75
Teen TV Shows                     69
TV Thrillers                      57
Stand-Up Comedy & Talk Shows      56
Classic & Cult TV                 28
TV Shows                          16
Name: listed_in, dtype: int64
```

In [445…

```
tli[tli['type']=='TV Show']['listed_in'].value_counts().sum()
```

Out[445]:

```
6133
```

In [446…

```
tli[tli['type']=='TV Show']['listed_in'].unique()
```

Out[446]:

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

In [447…

```
tli[tli['type']=='TV Show']['listed_in'].nunique()
```

Out[447]:

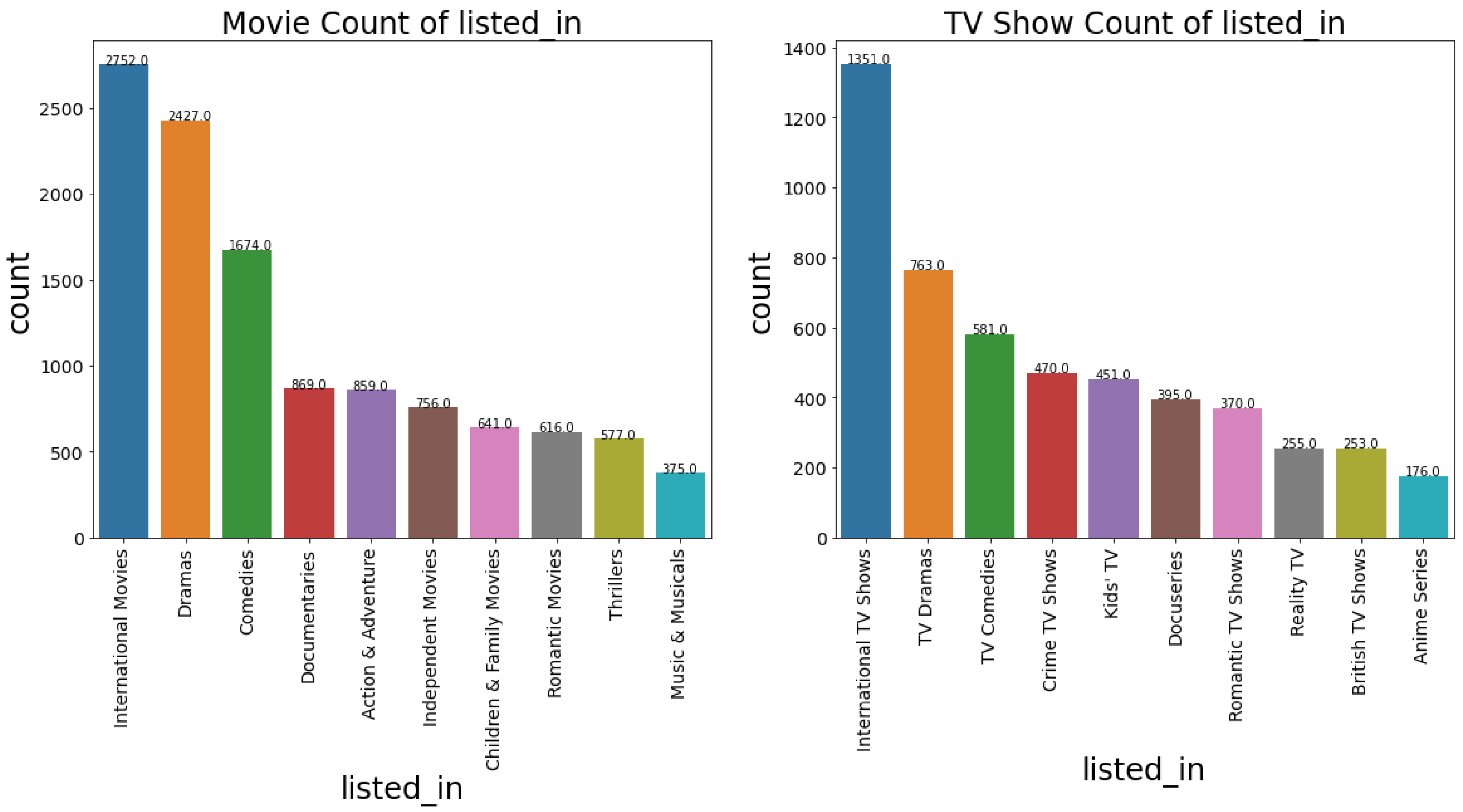
```
22
```

Bi- Variate Analysis

In [460…

```
plt.figure(figsize=(30,16))
plt.subplot(2, 3, 1)

plt.xlabel("listed_in")
plt.ylabel(" Movie count")
plt.xticks(rotation=90,fontsize=14)
plt.yticks(fontsize=14)
plt.xlabel('listed_in',size=24)
plt.ylabel('Count',size=24)
plt.title("Movie Count of listed_in", fontsize=24);
a=sns.countplot(x="listed_in", data=tli[tli["type"]=="Movie"], order=tli[tli['type']=='Movie']['listed_in'].value_count
for p in a.patches:
   a.annotate('{:.1f}'.format(p.get_height()), (p.get_x()+0.10, p.get_height()+0.45))
plt.subplot(2, 3, 2)
plt.xlabel("listed_in")
plt.ylabel("TV Show")
plt.xlabel('listed_in',size=24)
plt.ylabel('Count',size=24)
plt.xticks(rotation=90,fontsize=14)
plt.yticks(fontsize=14)
plt.title("TV Show Count of listed_in",fontsize=24);
b=sns.countplot(x="listed_in", data=tli[tli["type"]=="TV Show"], order=tli[tli['type']=='TV Show']['listed_in'].value_c
for p in b.patches:
   b.annotate('{:.1f}'.format(p.get_height()), (p.get_x()+0.10, p.get_height()+2.95))
plt.show()
```



Comment- The highest number of movies and TV shows are listed under International Movies and International TV shows genre.

In [449…

```
tr1=df[["title","rating","type"]]
tr1["rating"]=tr1["rating"].str.split(", ")
tr=tr1.explode("rating")
tr.replace("nan",
          np.nan, inplace=True)
tr.dropna(inplace=True)
tr
```

```
C:\Users\Sadiq\AppData\Local\Temp\ipykernel_11692\43719155.py: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
  tr1["rating"]=tr1["rating"].str.split(", ")
```

Out[449]:

	title	rating	type
0	Dick Johnson Is Dead	PG-13	Movie
1	Blood & Water	TV-MA	TV Show
2	Ganglands	TV-MA	TV Show
3	Jailbirds New Orleans	TV-MA	TV Show
4	Kota Factory	TV-MA	TV Show
...	...	...	...
8802	Zodiac	R	Movie
8803	Zombie Dumb	TV-Y7	TV Show
8804	Zombieland	R	Movie
8805	Zoom	PG	Movie
8806	Zubaan	TV-14	Movie

8803 rows × 3 columns

In [451…

```
tr[tr['type']=='Movie']['rating'].value_counts()
```

Out[451]:

```
TV-MA       2062
TV-14       1427
R            797
TV-PG        540
PG-13        490
PG           287
TV-Y7        139
TV-Y         131
TV-G         126
NR            75
G             41
TV-Y7-FV       5
NC-17          3
UR             3
74 min         1
84 min         1
66 min         1
Name: rating, dtype: int64
```

In [452…

```
tr[tr['type']=='Movie']['rating'].value_counts().sum()
```

Out[452]:

```
6129
```

In [455…

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

Out[455]:

```
array(['PG-13', 'PG', 'TV-MA', 'TV-PG', 'TV-14', 'TV-Y', 'R', 'TV-G',
       'TV-Y7', 'G', 'NC-17', '74 min', '84 min', '66 min', 'NR',
       'TV-Y7-FV', 'UR'], dtype=object)
```

In [456…

```
tr[tr['type']=='Movie']['rating'].nunique()
```

Out[456]:

```
17
```

In [453…

```
tr[tr['type']=='TV Show']['rating'].value_counts()
```

Out[453]:

```
TV-MA       1145
TV-14        733
TV-PG        323
TV-Y7        195
TV-Y         176
TV-G          94
NR             5
R              2
TV-Y7-FV       1
Name: rating, dtype: int64
```

In [454…

```
tr[tr['type']=='TV Show']['rating'].value_counts().sum()
```

Out[454]:

```
2674
```

In [457…

```
tr[tr['type']=='TV Show']['rating'].unique()
```

Out[457]:

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

In [458…

```
tr[tr['type']=='TV Show']['rating'].nunique()
```

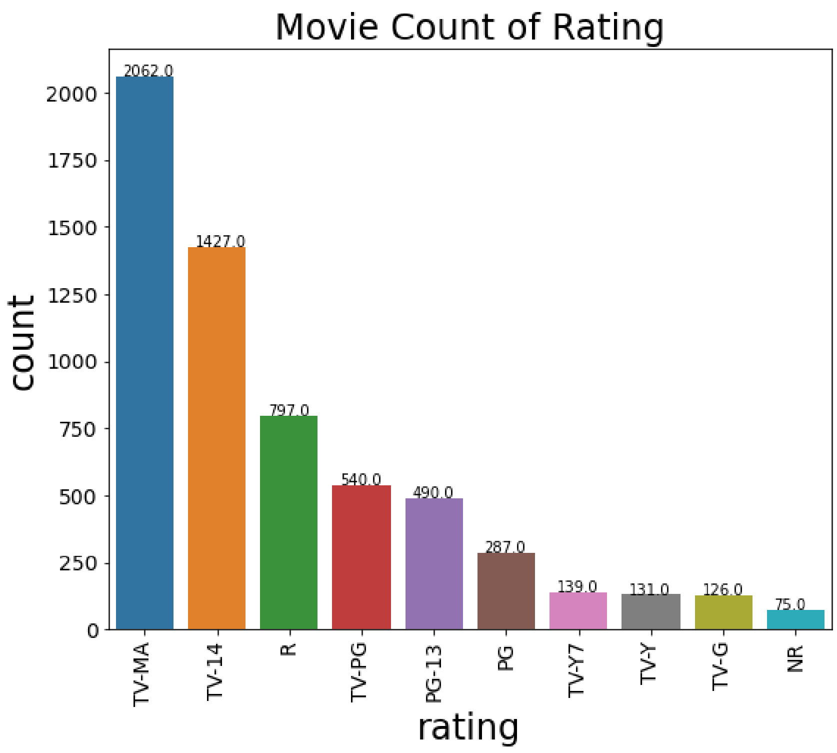
Out[458]:

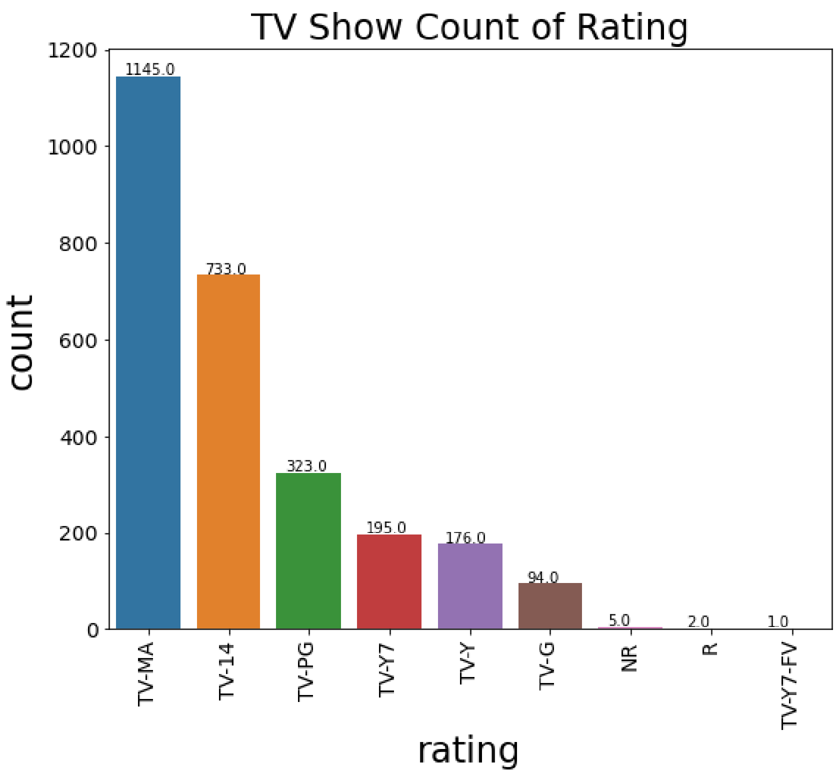
```
9
```

In [462…

```
plt.figure(figsize=(30,16))
plt.subplot(2, 3, 1)

plt.xlabel("Rating")
plt.ylabel(" Movie count")
plt.xticks(rotation=90,fontsize=14)
plt.yticks(fontsize=14)
plt.xlabel('Rating',size=24)
plt.ylabel('Count',size=24)
plt.title("Movie Count of Rating", fontsize=24);
a=sns.countplot(x="rating", data=tr[tr["type"]=="Movie"], order=tr[tr['type']=='Movie']['rating'].value_counts().index[
for p in a.patches:
   a.annotate('{:.1f}'.format(p.get_height()), (p.get_x()+0.10, p.get_height()+0.45))
plt.subplot(2, 3, 2)
plt.xlabel("rating")
plt.ylabel("TV Show")
plt.xlabel('Rating',size=24)
plt.ylabel('Count',size=24)
plt.xticks(rotation=90,fontsize=14)
plt.yticks(fontsize=14)
plt.title("TV Show Count of Rating",fontsize=24);
b=sns.countplot(x="rating", data=tr[tr["type"]=="TV Show"], order=tr[tr['type']=='TV Show']['rating'].value_counts().in
for p in b.patches:
   b.annotate('{:.1f}'.format(p.get_height()), (p.get_x()+0.10, p.get_height()+2.95))
plt.show()
```





Comment- The highest number of rating for movies and TV show is 'TV-MA'.

Unnesting of Title, release_year and Type

In [467…

```
tryy1=df[["title","release_year","type"]]
tryy=tryy1.explode("release_year")
tryy.replace("nan",
          np.nan, inplace=True)
tryy.dropna(inplace=True)
tryy
```

Out[467]:

	title	release_year	type
0	Dick Johnson Is Dead	2020	Movie
1	Blood & Water	2021	TV Show
2	Ganglands	2021	TV Show
3	Jailbirds New Orleans	2021	TV Show
4	Kota Factory	2021	TV Show
...	...	...	...
8802	Zodiac	2007	Movie
8803	Zombie Dumb	2018	TV Show
8804	Zombieland	2009	Movie
8805	Zoom	2006	Movie
8806	Zubaan	2015	Movie

8807 rows × 3 columns

In [502…

```
tryy[tryy['type']=='Movie']['release_year'].value_counts()
```

Out[502]:

```
2017    767
2018    767
2016    658
2019    633
2020    517
       ... 
1966      1
1961      1
1946      1
1963      1
1947      1
Name: release_year, Length: 73, dtype: int64
```

In [473…

```
tryy[tryy['type']=='Movie']['release_year'].unique()
```

Out[473]:

```
array([2020, 2021, 1993, 1996, 1998, 1997, 2010, 2013, 2017, 1975, 1978,
       1983, 1987, 2012, 2001, 2002, 2003, 2004, 2011, 2008, 2009, 2007,
       2005, 2006, 2018, 2019, 1994, 2015, 1982, 1989, 2014, 1990, 1991,
       1999, 2016, 1986, 1984, 1980, 1961, 2000, 1995, 1985, 1992, 1976,
       1959, 1988, 1981, 1972, 1964, 1954, 1979, 1958, 1956, 1963, 1970,
       1973, 1960, 1974, 1966, 1971, 1962, 1969, 1977, 1967, 1968, 1965,
       1945, 1946, 1942, 1955, 1944, 1947, 1943], dtype=int64)
```

In [474…

```
tryy[tryy['type']=='Movie']['release_year'].nunique()
```

Out[474]:

```
73
```

In [470…

```
tryy[tryy['type']=='Movie']['release_year'].value_counts().sum()
```

Out[470]:
```
6131
```

In [471…
```
tryy[tryy['type']=='TV Show']['release_year'].value_counts()
```

Out[471]:
```
2020    436
2019    397
2018    380
2021    315
2017    265
2016    244
2015    162
2014     88
2012     64
2013     63
2010     40
2011     40
2009     34
2008     23
2006     14
2007     14
2005     13
2003     10
2004      9
1999      7
2002      7
2001      5
1993      4
2000      4
1997      4
1998      4
1990      3
1996      3
1992      3
1995      2
1994      2
1988      2
1986      2
1989      1
1967      1
1985      1
1946      1
1981      1
1972      1
1979      1
1977      1
1991      1
1974      1
1925      1
1945      1
1963      1
Name: release_year, dtype: int64
```

In [472…
```
tryy[tryy['type']=='TV Show']['release_year'].value_counts().sum()
```

Out[472]:
```
2676
```

In [475…
```
tryy[tryy['type']=='TV Show']['release_year'].unique()
```

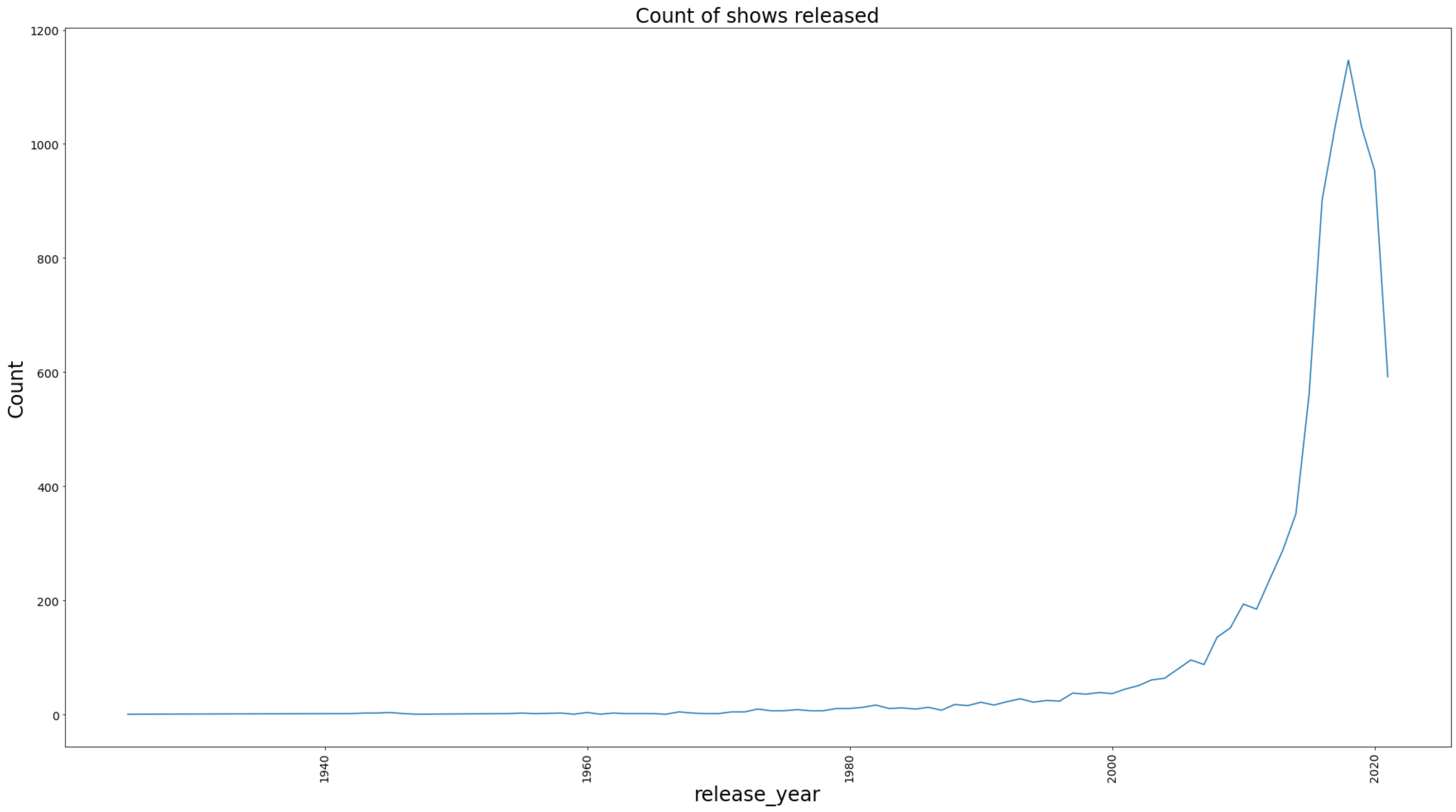
Out[475]:
```
array([2021, 2020, 2018, 2014, 1994, 2015, 2013, 2019, 2017, 2016, 2012,
       1992, 2002, 2009, 2011, 2005, 2008, 2010, 2007, 2001, 2006, 1993,
       1997, 2003, 1945, 1999, 1998, 2000, 2004, 1986, 1995, 1925, 1972,
       1974, 1988, 1991, 1977, 1979, 1990, 1996, 1981, 1946, 1985, 1967,
       1989, 1963], dtype=int64)
```

In [476…
```
tryy[tryy['type']=='TV Show']['release_year'].nunique()
```

Out[476]:
```
46
```

Uni-variate Analysis

In [501…
```
plt.figure(figsize=(30,16))
tryyy=df.groupby(df["release_year"])[["show_id"]].count().sort_values(by=["show_id"],ascending=False).reset_index()
plt.title("Count of shows released", size=24)
plt.xticks(rotation=90,fontsize=14)
plt.yticks(fontsize=14)
plt.xlabel('release_year',size=24)
plt.ylabel('Count',size=24)
a=sns.lineplot(data=tryyy, x=tryyy["release_year"], y=tryyy["show_id"])
```



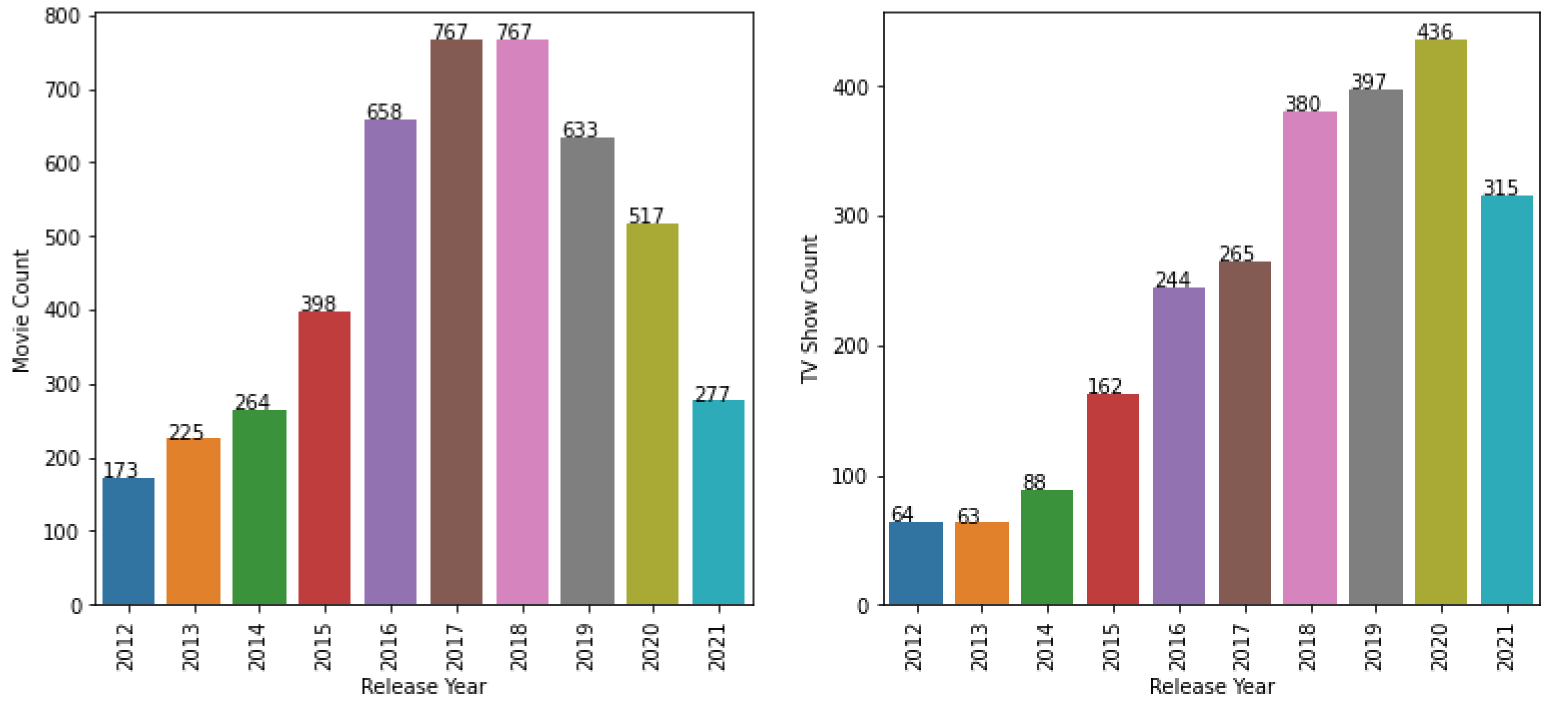
Comment- The highest number of shows have been released on 2020 year.

Bi- Variate Analysis

In [515…

```
plt.figure(figsize=(20,12))
plt.subplot(2, 3, 1)
a=sns.barplot(data=tryy, x=tryy[tryy["type"]=="Movie"]['release_year'].value_counts().index[0:10],
 y=tryy[tryy["type"]=="Movie"]['release_year'].value_counts().head(10))
plt.xticks(rotation=90)
plt.xlabel("Release Year")
plt.ylabel("Movie Count")
for p in a.patches:
   a.annotate('{:.0f}'.format(p.get_height()), (p.get_x()+0.00, p.get_height()+0.55))

plt.subplot(2, 3, 2)
b=sns.barplot(data=tryy, x=tryy[tryy["type"]=="TV Show"]['release_year'].value_counts().index[0:10],
 y=tryy[tryy["type"]=="TV Show"]['release_year'].value_counts().head(10))
plt.xticks(rotation=90)
plt.xlabel("Release Year")
plt.ylabel("TV Show Count")
for p in b.patches:
   b.annotate('{:.0f}'.format(p.get_height()), (p.get_x()+0.00, p.get_height()+0.55))
```

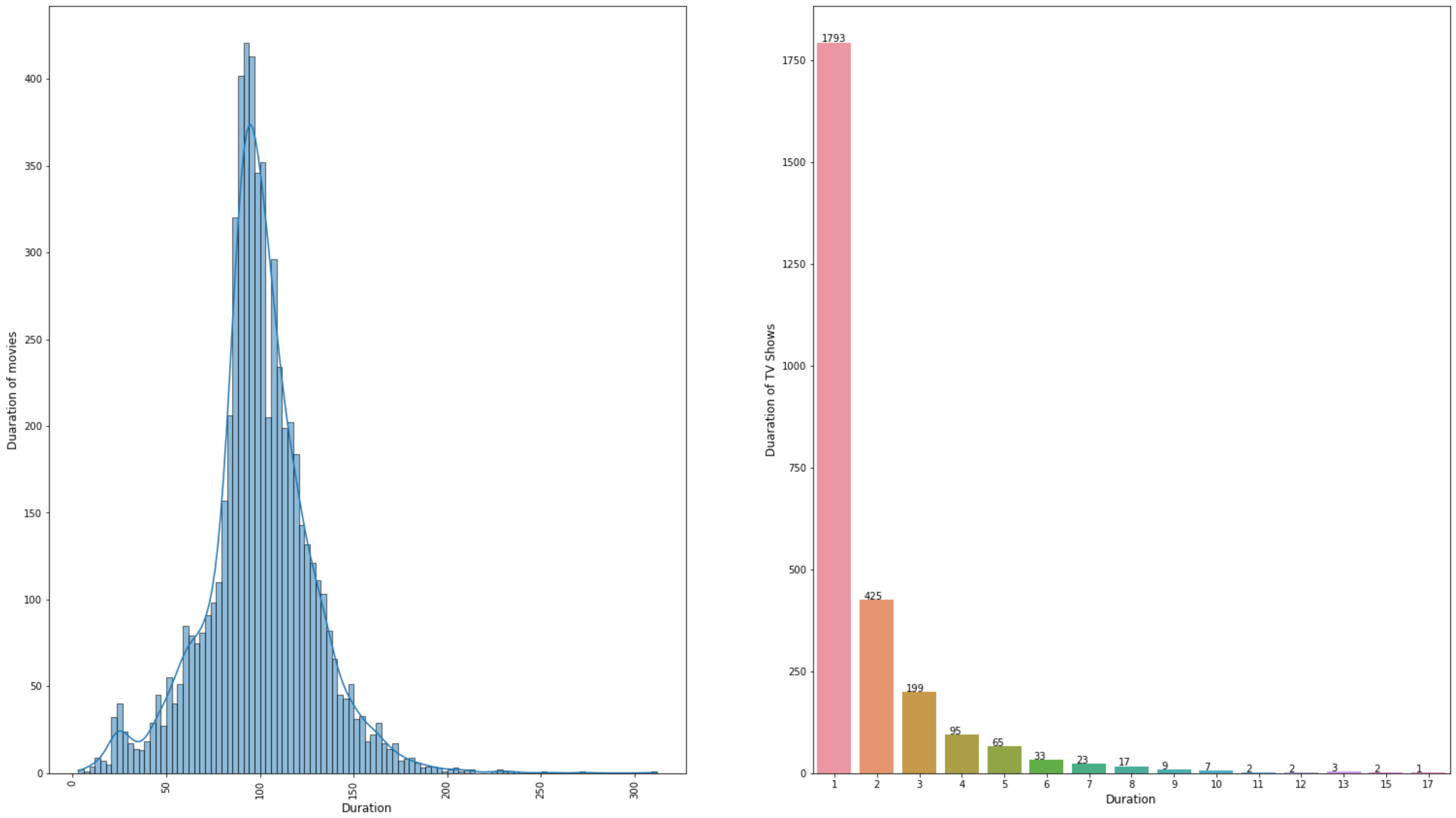


Comment- Highest number of movies have been released on 2017 and 2018 year wheres highest number of Tv shows are released on 2020 year.

In [526…

```
plt.figure(figsize=(40,32))
plt.subplot(2, 3, 1)
sns.histplot(df[df['type']=='Movie']['duration'],kde = True)
plt.xlabel("Duration", fontsize=12)
plt.ylabel("Duaration of movies", fontsize=12)
plt.xticks(rotation=90)
plt.subplot(2, 3, 2)
a=sns.countplot(df[df['type']=='TV Show']['duration'],x = 'duration')
plt.xlabel("Duration", fontsize=12)
plt.ylabel("Duaration of TV Shows", fontsize=12)
for p in a.patches:
   a.annotate('{:.0f}'.format(p.get_height()), (p.get_x()+0.10, p.get_height()+0.85))
```

```
C:\Users\Sadiq\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a k
eyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments withou
t an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```



Comment- The highest number of minutes, users spent over netflix movies varies from 80 to 120 minutes and season 1 TV show has the highest spent duration.

In [553…

```
tcy1=df[["type","title","country","duration"]]
tcy1["country"]=tcy1["country"].str.split(", ")
tcy1=tcy1.explode("country")
cat = tli["listed_in"].value_counts().index[0:5]
tle1 = tli['title'].value_counts().index[0:5]
cny = tcy1['country'].value_counts().index[0:5]
typ = tcy1['type'].value_counts().index[0:2]
typ_data = tcy1.loc[(tcy1['country'].isin(cny)) & (tcy1['type'].isin(typ)) & (tcy1['type'].isin(typ))]
typ_data
```

```
C:\Users\Sadiq\AppData\Local\Temp\ipykernel_11692\2554661986.py: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
  tcy1["country"]=tcy1["country"].str.split(", ")
```

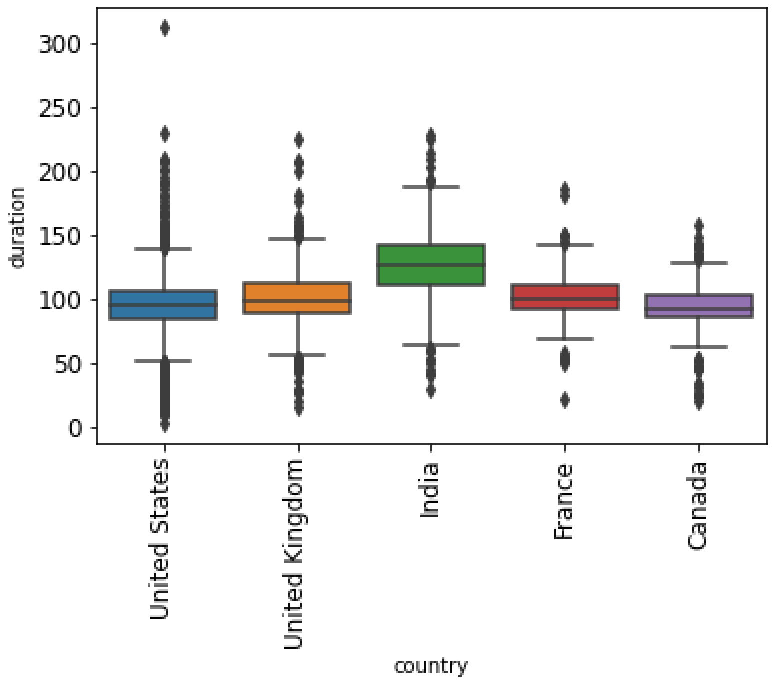
Out[553]:

	type	title	country	duration
0	Movie	Dick Johnson Is Dead	United States	90
4	TV Show	Kota Factory	India	2
7	Movie	Sankofa	United States	125
7	Movie	Sankofa	United Kingdom	125
8	TV Show	The Great British Baking Show	United Kingdom	9
...	...	...	...	...
8799	Movie	Zenda	India	120
8802	Movie	Zodiac	United States	158
8804	Movie	Zombieland	United States	88
8805	Movie	Zoom	United States	88
8806	Movie	Zubaan	India	111

6377 rows × 4 columns

In [554…

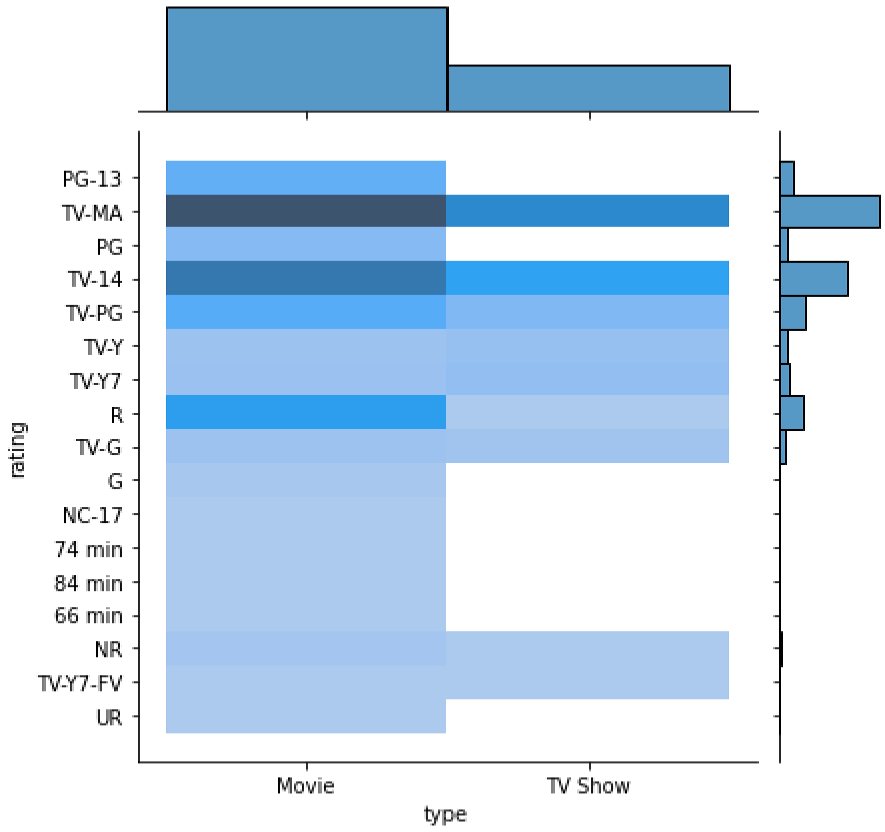
```
sns.boxplot(x='country', y='duration', data=typ_data[typ_data["type"]=="Movie"])
plt.xticks(rotation=90,fontsize=12)
plt.yticks(fontsize=12)
plt.show()
```



Comment- Indian users spent highest duration of time watching Movies at Netflix.

In [528…

```
sns.jointplot(data=df, x="type", y="rating",kind="hist")
plt.show()
```



In [563…

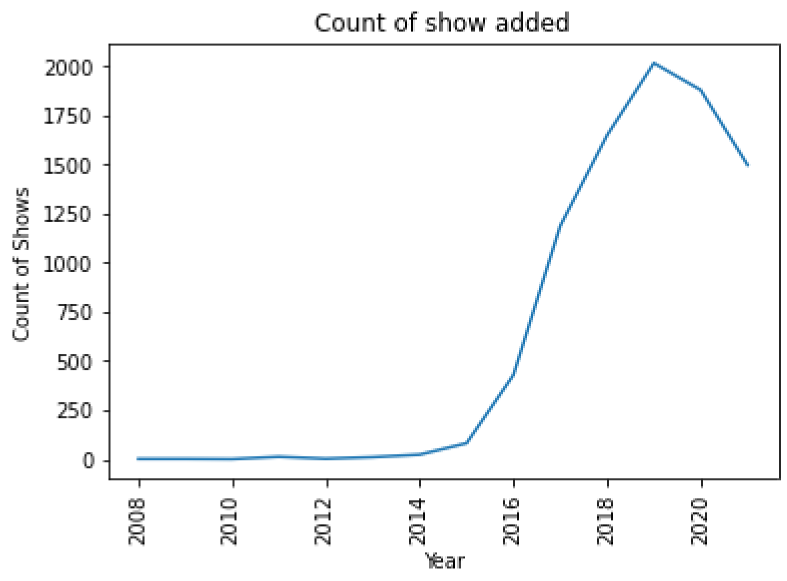
```
df['date_added'] = pd.to_datetime(df["date_added"])
df['year'] = df['date_added'].dt.year
df['month'] = df['date_added'].dt.month
df['week']=df['date_added'].dt.isocalendar().week
df.head()
```

Out[563]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	year	mo
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	2021-09-25	2020	PG-13	90	Documentaries	As her father nears the end of his life, filmm...	2021	
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	2021-09-24	2021	TV-MA	2	International TV Shows, TV Dramas, TV Mysteries	After crossing paths at a party, a Cape Town t...	2021	
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	2021-09-24	2021	TV-MA	1	Crime TV Shows, International TV Shows, TV Act...	To protect his family from a powerful drug lor...	2021	
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	NaN	2021-09-24	2021	TV-MA	1	Docuseries, Reality TV	Feuds, flirtations and toilet talk go down amo...	2021	
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	2021-09-24	2021	TV-MA	2	International TV Shows, Romantic TV Shows, TV ...	In a city of coaching centers known to train I...	2021	

In [540...

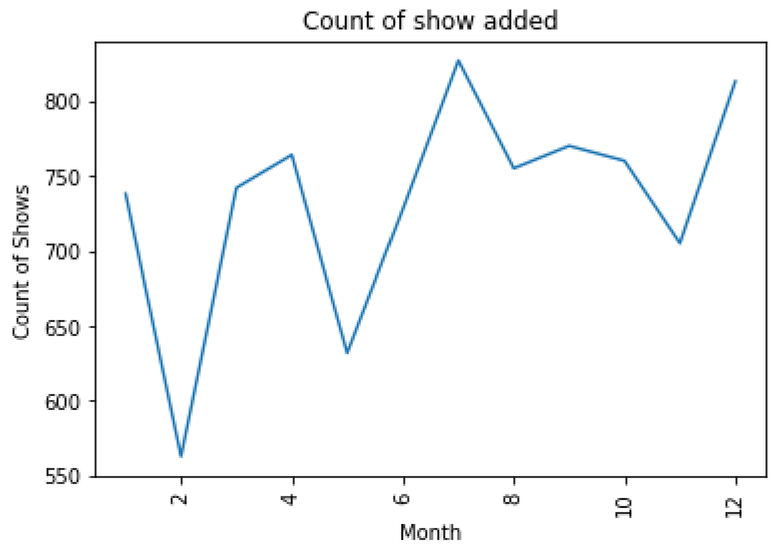
```
yr=df.groupby(df["year"])[["show_id"]].count().sort_values(by=["show_id"],ascending=False).reset_index()
plt.xticks(rotation=90)
plt.xlabel("Year")
plt.ylabel("Count of Shows")
plt.title("Count of show added")
a=sns.lineplot(data=yr, x=yr["year"], y=yr["show_id"])
```



Comment- In the 2020, highest number of movies and TV shows have been added on Netflix.

In [541...

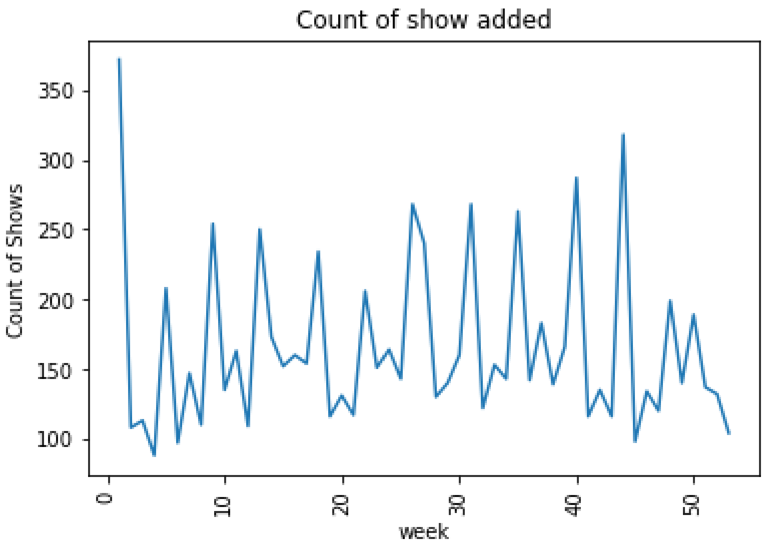
```
mn=df.groupby(df["month"])[["show_id"]].count().sort_values(by=["show_id"],ascending=False).reset_index()
plt.xticks(rotation=90)
plt.xlabel("Month")
plt.ylabel("Count of Shows")
plt.title("Count of show added")
a=sns.lineplot(data=mn, x=mn["month"], y=mn["show_id"])
```



Comment- In July month, highest number of movies or TV shows have been added on Netflix.

In [543…

```python
wk=df.groupby(df["week"])[["show_id"]].count().sort_values(by=["show_id"],ascending=False).reset_index()
plt.xticks(rotation=90)
plt.xlabel("week")
plt.ylabel("Count of Shows")
plt.title("Count of show added")
a=sns.lineplot(data=wk, x=wk["week"], y=wk["show_id"])
```

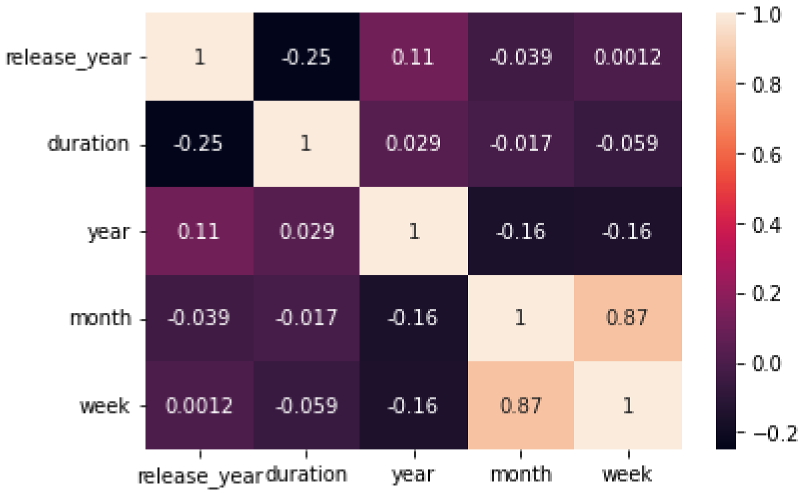


In [561…

```python
corr = df.corr()
sns.heatmap(corr, annot=True)
```

Out[561]:

```
<AxesSubplot:>
```



Comment- The month and week columns have the correlation

5. Missing Value & Outlier check (Treatment optional)

In [558…

```python
# Filling the missing value in date_added column
df['date_added'] = df['date_added'].fillna(df['date_added'].max())
```

In [560…

```python
# Filling the missing value in rating column
df['rating'] = df['rating'].fillna(df['rating'].mode()[0])
```

In [564…

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

Out[564]:

```
show_id            0
type               0
title              0
director        2634
cast             825
country          831
date_added         0
release_year       0
rating             0
duration           0
listed_in          0
description        0
year               0
month              0
week               0
dtype: int64
```

In [567…

```python
# Filling data of cast with director column
tc1=df[["title","cast","type"]]
tc1["cast"]=tc1["cast"].str.split(", ")
tc=tc1.explode("cast")
td1=df[["title","director","type"]]
td1["director"]=td1["director"].str.split(", ")
td=td1.explode("director")
tcd=pd.merge(tc,td, on="title",how="inner")
tcd.isna().sum()
```

```
C:\Users\Sadiq\AppData\Local\Temp\ipykernel_11692\3764346636.py:3: 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
  tc1["cast"]=tc1["cast"].str.split(", ")
C:\Users\Sadiq\AppData\Local\Temp\ipykernel_11692\3764346636.py:6: 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
  td1["director"]=td1["director"].str.split(", ")
```

Out[567]:

```
title           0
cast          960
type_x          0
director    19013
type_y          0
dtype: int64
```

In [570…

```
tcd1=tcd.groupby(['director'])['cast'].agg(pd.Series.mode).to_frame().reset_index().rename(columns={'cast':'actor_mod'}
tcd2=tcd.merge(tcd1,on='director',how='left')
tcd2=tcd2.fillna({'cast':tcd2.actor_mod}).drop('actor_mod',axis=1)
tcd2.isna().sum()
```

Out[570]:

```
title           0
cast          352
type_x          0
director    19013
type_y          0
dtype: int64
```

In [571…

```
#Filling the country value with rating column
tcr=df[["type","country","rating"]]
tcr["country"]=tcr["country"].str.split(",")
tcr1=tcr.explode("country")
tcr1.isna().sum()
```

```
C:\Users\Sadiq\AppData\Local\Temp\ipykernel_11692\1560121334.py:3: 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
  tcr["country"]=tcr["country"].str.split(",")
```

Out[571]:

```
type          0
country     831
rating        0
dtype: int64
```

In [576…

```
tcr2=tcr.groupby(['rating',"type"])['country'].agg(pd.Series.mode).to_frame().reset_index().rename(columns={'country':'
tcr3=tcr1.merge(tcr2,on=['rating',"type"],how='left')
tcr3=tcr3.fillna({'country':tcr3.country_mod}).drop('country_mod',axis=1)
tcr3.isna().sum()
```

Out[576]:

```
type       0
country    0
rating     0
dtype: int64
```

Comment- Missing values have been reduced after filling the values using mod imputation.

7. Business Insights

1. There are '36439' different cast that have acted in '7982' different Movies and Tv shows on Netflix. Among them, 'Anupam Kher' has acted in the highest number of movies and Tv show total, i.e. '43'.
2. There are '4993' different directors who have directed for '6173' different number of Movies and TV shows. Among them, 'Rajiv Chilaka' has directed the highest number of movies in total '22'.
3. Netflix has added '7976' movies and Tv shows having their corresponding '127' countries. Among them, the country 'United states' has the highest number of Movies and TV show with the count of '3689'.
4. The Netflix TV shows and movies have '42' different genres. Among them, the 'International Movies' genre has the highest count of '2752' movies and TV shows.
5. There are 13 different ratings for Netflix TV shows and Movies. Among them, the highest rating is 'TV-MA' with a total of '3207'.
6. The Netflix movies and TV shows have been released in 74 different years across different countries. Among them, In 2018, the highest number of Movies and TV shows have been released.
7. Out of '25951' different cast, 'Anupam Kher' has acted in the highest number of movies with count of '42'. Out of '14863' different cast, 'Takahiro Sakurai ' has acted in the highest number of TV Shows with count of '25'.
8. Out of '4777' different directors, 'Rajiv Chilaka' has directed the highest number of movies with count of '22'. Out of '299' different directors, 'Alastair Fothergill' and Ken Burns have directed the highest number of TV Shows with count of '3'.

9. The United States country is leading for having the highest number of movies i.e ‘2751’ and highest number of TV shows i.e ‘938’.
10. As per trend the decrease in adding movies and TV shows have been observed after 2019. As per trend, July and December are the months where the highest number of TV shows and movies were added.

8. Recommendations

- As the largest number of users prefer to watch Tv shows having only 1 or 2 seasons. Creating new Tv shows having a single season could be beneficial.
- As Indian users have spent the highest duration on Netflix movies, it would be beneficial to launch new movies with good advertisements in India.
- As the highest count of TV shows and movies having an Adult rating, it would be beneficial to produce movies and TV shows which can be viewed by any age, as it would attract and include children as well.
- It would be beneficial to launch new movies and TV shows during the Holiday season such as December and January.
- To increase the duration of users on Netflix, more TV shows and movies have to be added.