Netflix Case Study Analysis - Mayank Khanduja

In [ ]:

Summary

The consolidated summary of the report is depicted below: 1) Most Popular Categories - Based on Data Analysis the most popular categories comes out to be International Movies/TV shows, Dramas and Comedies

2) Trending Release Year - 2018 year marked the maximum releases of TV shows and Movies combined 2017,2018 marked the highest release year for MOVIES where as 2019,2020 marked the highest release year for TV SHOWS

3) Country wise Distribution - Based on Data Analysis we found that Neflix primarily target 3 Countries i.e. USA , India and UK.

4) Content Duration - The MEDIAN value of MOVIES is 100 minute while for TV shows it is 1 season.

5) Rating Analysis - Netflix make and target individuals who falls into “TV-MA” and “TV-14” category

6) Popular Director and Cast- The most popular Director and Cast combo comes out to be "Rajiv Chilaka-Julie Tejwani" AND "Rajiv Chilaka-Rajesh Kava"

7) Description Analysis 7.1) Summary Statistics- The mean length of description comes to be 143 characters 7.2) Most common words in description- The most common words if we focus only on

specific words comes out to be Young , Life , Family , Love, Friends , Documentary 7.3) Sentiment Analysis- 332 descriptions sounds positive while 2500 sounds negative

8) Movie v/s TV Shows 8.1) Counts - Netflix invest more in movies than TV Shows 8.2) Trend of Movies and TV Shows - The trend for movies as well as TV Shows both increases with time.

9) Popular Genre by Country - Depicts the unique genre country wise

10) Popular Director by Genre - Depicts the popular directors by genre.

In [2]:
```
import numpy as np
import pandas as pd
from datetime import datetime as dt
import matplotlib.pyplot as plt
```

In [3]:
```
df = pd.read_csv("C:\\Users\\Test\\Downloads\\Netflix.csv")
df.shape
```

Out[3]:
```
(8807, 12)
```

Most Popular Categories

In [4]:
```
listed_in= df["listed_in"].apply(lambda x:str(x).split(", ")).tolist()
dfL= pd.DataFrame(listed_in,index=df["title"])
dfL = pd.DataFrame(dfL.stack())
dfL.reset_index(inplace=True)
dfL = dfL[["title",0]]
df_listed_in = df.merge(dfL, on="title",how="inner")
df_listed_in
df_listed_in_final = df_listed_in[["type","title","country","rating","release_year",0]]
df_listed_in_final.rename(columns={0:"category"},inplace=True)
df_listed_in_final.groupby("type")["category"].value_counts()
```

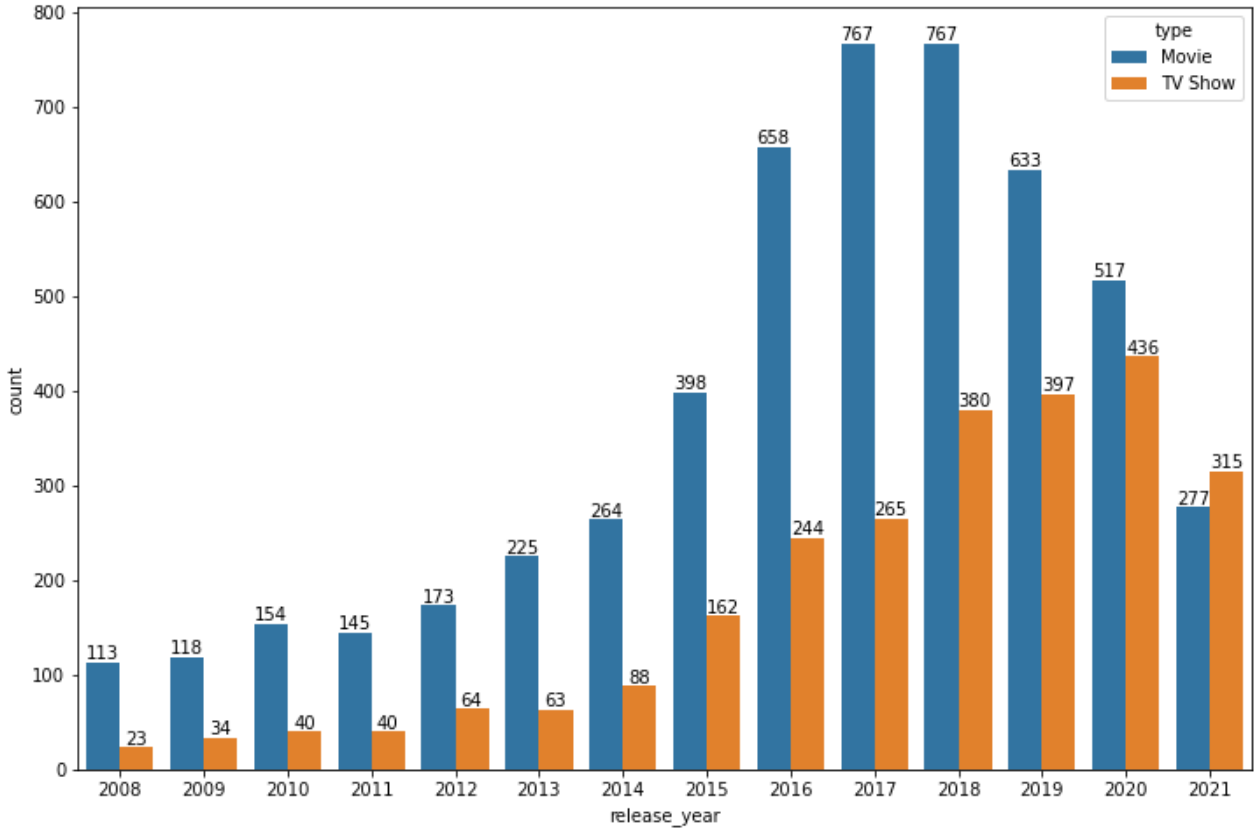
Out[4]:
```
C:\Users\Test\anaconda3\lib\site-packages\pandas\core\frame.py:5039: SettingWithCopyWarn
ing:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
  return super().rename(
type     category
Movie    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
TV Show         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: category, dtype: int64
```

Trending Release Year

In [5]:

```
import seaborn as sns
#taking past 10 year data
df2= df.loc[df["release_year"]>= 2008]
plt.figure(figsize=(12,8))
ax= sns.countplot(data=df2,x="release_year",hue="type")
for label in ax.containers:
    ax.bar_label(label)
plt.show()
```



In [6]:
```
df["release_year"].value_counts()
```

Out[6]:
```
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
```

Country Wise Distribution

In [7]:
```
#handling Null Values
# 831 Null values which contributes 9.5% of data
# MODE IMPUTATION IS DONE TO FILL THE NULL VALUES
```

In [8]:
```
df["country"].fillna("United States",inplace=True)
```

In [9]:
```
dfC= pd.DataFrame(df["country"].apply(lambda x:str(x).split(", ")).tolist(),index=df["t
dfC= pd.DataFrame(dfC.stack())
dfC.reset_index(inplace=True)
dfC= dfC[["title",0]]
```

```
dfCf = df.merge(dfC,on="title",how="inner")
dfCf= dfCf[["type","director","release_year","date_added","duration",0]]
dfCf.rename(columns={0:"country"},inplace=True)
dfCf
```

Out[9]:

	type	director	release_year	date_added	duration	country
0	Movie	Kirsten Johnson	2020	September 25, 2021	90 min	United States
1	TV Show	NaN	2021	September 24, 2021	2 Seasons	South Africa
2	TV Show	Julien Leclercq	2021	September 24, 2021	1 Season	United States
3	TV Show	NaN	2021	September 24, 2021	1 Season	United States
4	TV Show	NaN	2021	September 24, 2021	2 Seasons	India
...	...	...	...	...	...	...
10840	Movie	David Fincher	2007	November 20, 2019	158 min	United States
10841	TV Show	NaN	2018	July 1, 2019	2 Seasons	United States
10842	Movie	Ruben Fleischer	2009	November 1, 2019	88 min	United States
10843	Movie	Peter Hewitt	2006	January 11, 2020	88 min	United States
10844	Movie	Mozez Singh	2015	March 2, 2019	111 min	India

10845 rows × 6 columns

In [10]:

```
dfCf["country"].value_counts().head(10)
```

Out[10]:

```
United States     4520
India             1046
United Kingdom     804
Canada             445
France             393
Japan              318
Spain              232
South Korea        231
Germany            226
Mexico             169
Name: country, dtype: int64
```

In [11]:

```
df_filtered= dfCf.loc[dfCf["country"].isin(["United States","India","United Kingdom","C
df_filtered
```

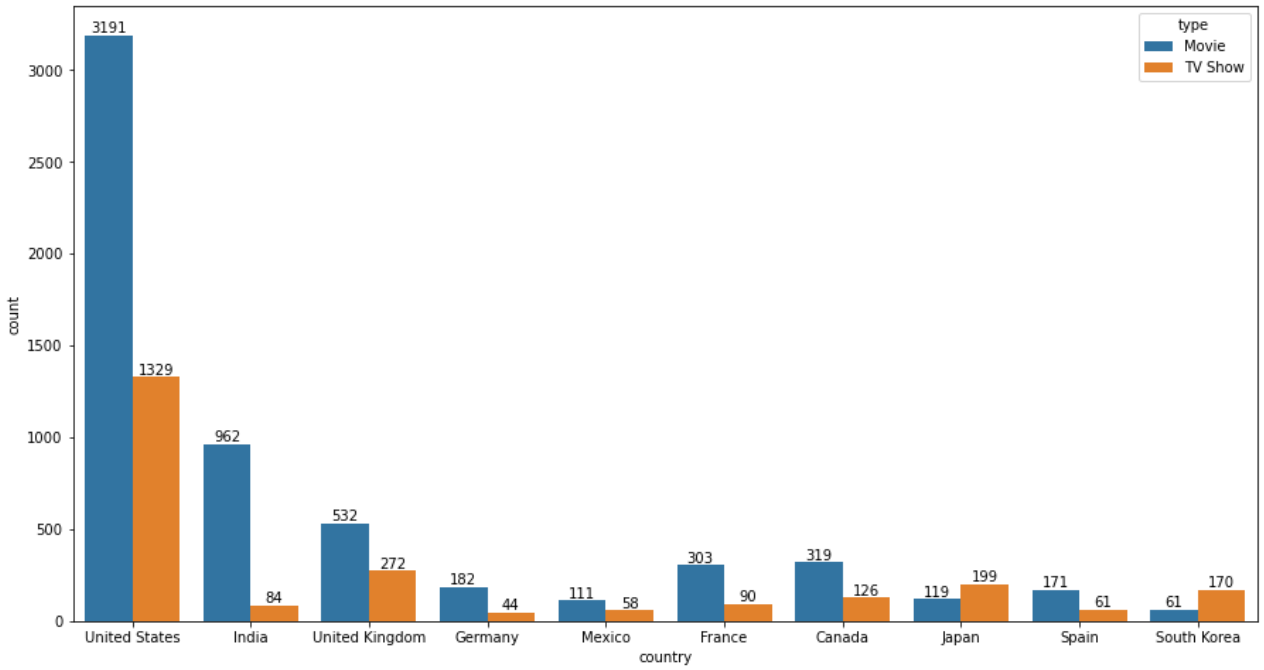
Out[11]:

	type	director	release_year	date_added	duration	country
0	Movie	Kirsten Johnson	2020	September 25, 2021	90 min	United States
2	TV Show	Julien Leclercq	2021	September 24, 2021	1 Season	United States
3	TV Show	NaN	2021	September 24, 2021	1 Season	United States
4	TV Show	NaN	2021	September 24, 2021	2 Seasons	India
5	TV Show	Mike Flanagan	2021	September 24, 2021	1 Season	United States
...	...	...	...	...	...	...

	type	director	release_year	date_added	duration	country
10840	Movie	David Fincher	2007	November 20, 2019	158 min	United States
10841	TV Show	NaN	2018	July 1, 2019	2 Seasons	United States
10842	Movie	Ruben Fleischer	2009	November 1, 2019	88 min	United States
10843	Movie	Peter Hewitt	2006	January 11, 2020	88 min	United States
10844	Movie	Mozez Singh	2015	March 2, 2019	111 min	India

8384 rows × 6 columns

In [12]:
```
import seaborn as sns
#taking top 10 countries
plt.figure(figsize=(15,8))
ax= sns.countplot(data=df_filtered,x=df_filtered["country"],hue="type")
for label in ax.containers:
    ax.bar_label(label)
plt.show()
```



Content Duration

In [13]:
```
df.loc[df["duration"].isna()]
```

Out[13]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	lis
5541	s5542	Movie	Louis C.K. 2017	Louis C.K.	Louis C.K.	United States	April 4, 2017	2017	74 min	NaN	N
5794	s5795	Movie	Louis C.K.: Hilarious	Louis C.K.	Louis C.K.	United States	September 16, 2016	2010	84 min	NaN	N

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	lis
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	N

In [14]:
```
# Here rating and duration columns are interchanged > we need to fix this
df.loc[5541,"duration"]= df.loc[5541,"rating"]
df.loc[5794,"duration"]= df.loc[5794,"rating"]
df.loc[5813,"duration"]= df.loc[5813,"rating"]
```

In [15]:
```
df.loc[[5541,5794,5813]]
```

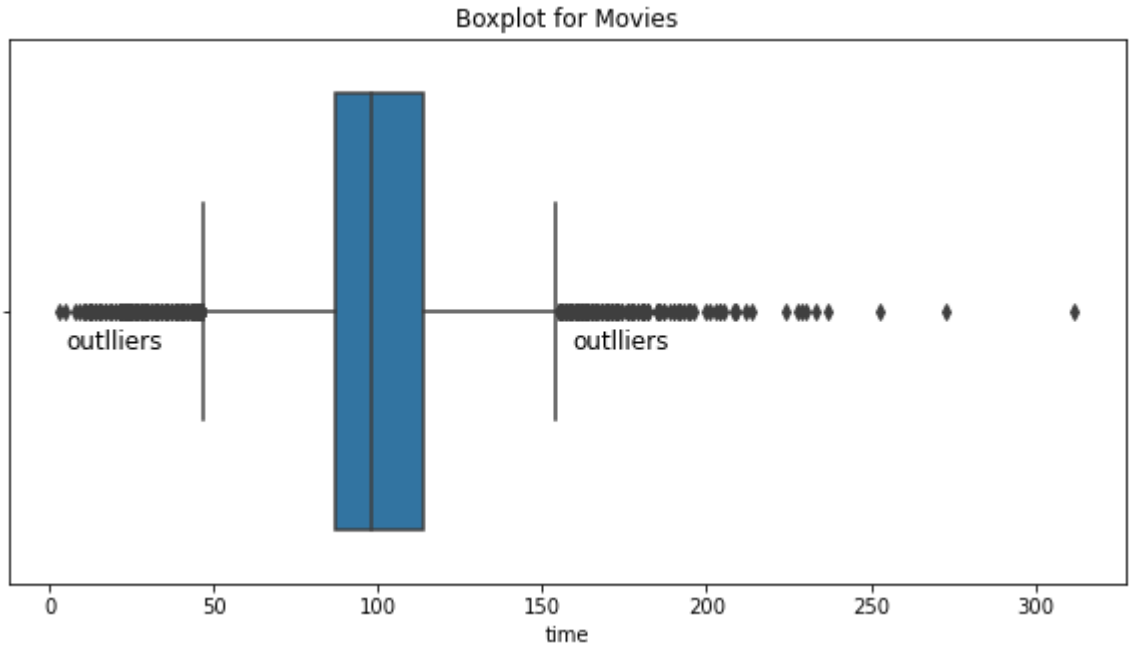
Out[15]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	lis
5541	s5542	Movie	Louis C.K. 2017	Louis C.K.	Louis C.K.	United States	April 4, 2017	2017	74 min	74 min	N
5794	s5795	Movie	Louis C.K.: Hilarious	Louis C.K.	Louis C.K.	United States	September 16, 2016	2010	84 min	84 min	N
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	66 min	N

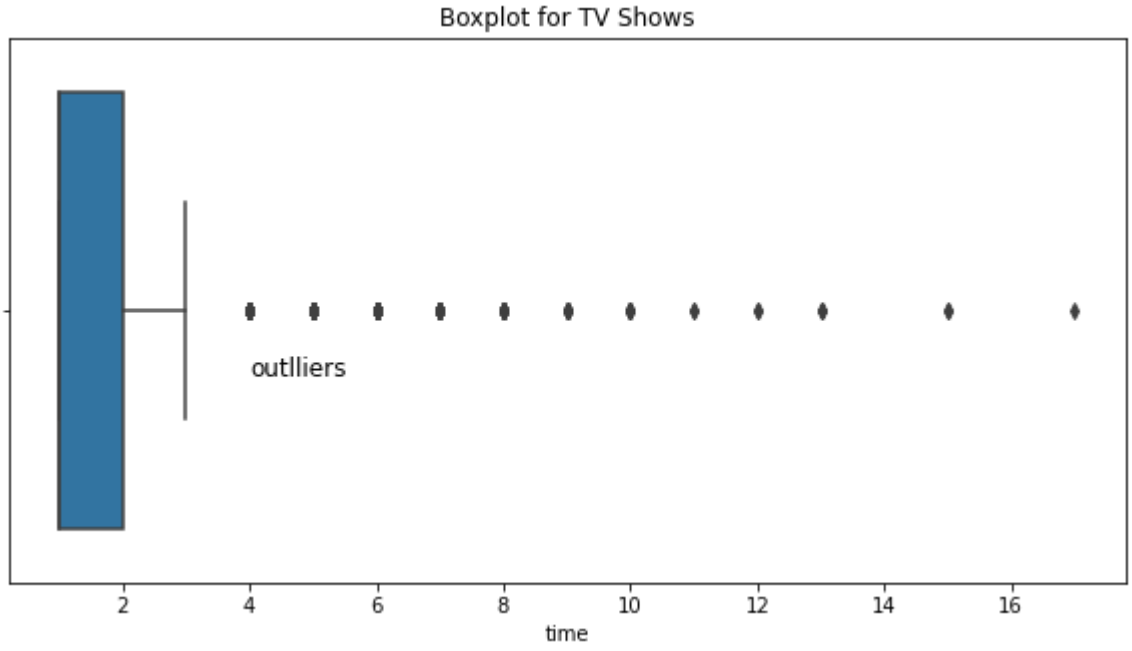
In [16]:
```
df["time"]= pd.DataFrame(df["duration"].apply(lambda x:str(x).split(" ")).tolist())[0]
df["time"]= df["time"].astype(int)
```

In [17]:
```
#segregating movies and tv show type of data
dfM= df.loc[df["type"]=="Movie"]
dfT = df.loc[df["type"]=="TV Show"]
```

In [18]:
```
plt.figure(figsize=(10,5))
sns.boxplot(x=dfM["time"])
plt.text(159,0.07,"outlliers",fontsize=12)
plt.text(5,0.07,"outlliers",fontsize=12)
plt.title("Boxplot for Movies")
plt.show()
```



In [19]:
```
plt.figure(figsize=(10,5))
sns.boxplot(x=dfT["time"])
plt.text(4,0.12,"outlliers",fontsize=12)
plt.title("Boxplot for TV Shows")
plt.show()
```



In [20]:
```
dfT.describe()["time"]
```

Out[20]:
```
count    2676.000000
mean        1.764948
std         1.582752
min         1.000000
25%         1.000000
50%         1.000000
75%         2.000000
```

```
max        17.000000
Name: time, dtype: float64
```

Rating Analysis

In [21]:
```
df["rating"].unique()
```

Out[21]:
```
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 [22]:
```
df.loc[[5541,5794,5813]]
```

Out[22]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	lis
5541	s5542	Movie	Louis C.K. 2017	Louis C.K.	Louis C.K.	United States	April 4, 2017	2017	74 min	74 min	N
5794	s5795	Movie	Louis C.K.: Hilarious	Louis C.K.	Louis C.K.	United States	September 16, 2016	2010	84 min	84 min	N
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	66 min	N

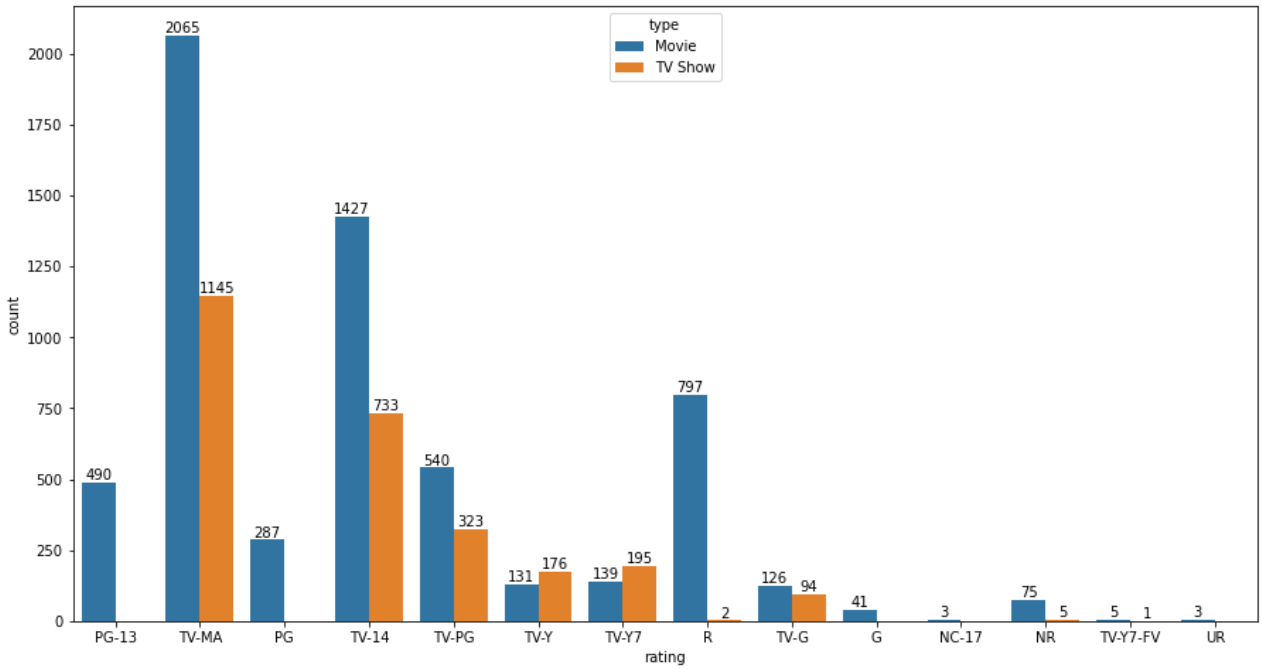
In [23]:
```
# Here rating of these 3 indexes are wrong so changing it to TV-MA since I googled all
df.loc[5541,"rating"]= 'TV-MA'
df.loc[5794,"rating"]= 'TV-MA'
df.loc[5813,"rating"]= 'TV-MA'
```

In [24]:
```
df.loc[[5541,5794,5813]]
```

Out[24]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	lis
5541	s5542	Movie	Louis C.K. 2017	Louis C.K.	Louis C.K.	United States	April 4, 2017	2017	TV-MA	74 min	N
5794	s5795	Movie	Louis C.K.: Hilarious	Louis C.K.	Louis C.K.	United States	September 16, 2016	2010	TV-MA	84 min	N
5813	s5814	Movie	Louis C.K.: Live at the Comedy Store	Louis C.K.	Louis C.K.	United States	August 15, 2016	2015	TV-MA	66 min	N

In [25]:
```
import seaborn as sns
#taking top 10 countried
plt.figure(figsize=(15,8))
ax= sns.countplot(data=df,x=df["rating"],hue="type")
for label in ax.containers:
    ax.bar_label(label)
plt.show()
```



In [26]:
```
df["rating"].value_counts().head(5)
```

Out[26]:
```
TV-MA    3210
TV-14    2160
TV-PG     863
R         799
PG-13     490
Name: rating, dtype: int64
```

Popular Director & Cast

In [27]:
```
# director - 2634 entries are null (29%)
# cast - 825 entries null (9.36%)
# ADD DESCRIPTION
```

In [28]:
```
# CASE 01 - Finding popular director and cast from the data which is available and non-
subset1 = df.loc[(~ df["director"].isna()) & (~df["cast"].isna())]
subset1.reset_index(inplace=True)
subset1.drop(columns=["index"],inplace=True)
subset1
```

```
C:\Users\Test\anaconda3\lib\site-packages\pandas\core\frame.py:4906: SettingWithCopyWarn
ing:
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
  return super().drop(
```

Out[28]:

	show_id	type	title	director	cast	country	date_added	release_year	rating
0	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	United States	September 24, 2021	2021	TV-MA
1	s6	TV Show	Midnight Mass	Mike Flanagan	Kate Siegel, Zach Gilford, Hamish Linklater, H...	United States	September 24, 2021	2021	TV-MA
2	s7	Movie	My Little Pony: A New Generation	Robert Cullen, José Luis Ucha	Vanessa Hudgens, Kimiko Glenn, James Marsden, ...	United States	September 24, 2021	2021	PG
3	s8	Movie	Sankofa	Haile Gerima	Kofi Ghanaba, Oyafunmike Ogunlano, Alexandra D...	United States, Ghana, Burkina Faso, United Kin...	September 24, 2021	1993	TV-MA
4	s9	TV Show	The Great British Baking Show	Andy Devonshire	Mel Giedroyc, Sue Perkins, Mary Berry, Paul Ho...	United Kingdom	September 24, 2021	2021	TV-14
...	...	...	...	...	...	...	...	...	...
5695	s8802	Movie	Zinzana	Majid Al Ansari	Ali Suliman, Saleh Bakri, Yasa, Ali Al-Jabri, ...	United Arab Emirates, Jordan	March 9, 2016	2015	TV-MA
5696	s8803	Movie	Zodiac	David Fincher	Mark Ruffalo, Jake Gyllenhaal, Robert Downey J...	United States	November 20, 2019	2007	R
5697	s8805	Movie	Zombieland	Ruben Fleischer	Jesse Eisenberg, Woody	United States	November 1, 2019	2009	R

	show_id	type	title	director	cast	country	date_added	release_year	rating
					Harrelson, Emma Stone, ...				
5698	s8806	Movie	Zoom	Peter Hewitt	Tim Allen, Courteney Cox, Chevy Chase, Kate Ma...	United States	January 11, 2020	2006	PG
5699	s8807	Movie	Zubaan	Mozez Singh	Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanan...	India	March 2, 2019	2015	TV-14

5700 rows × 13 columns

In [29]:
```
# director
s1= pd.DataFrame(subset1["director"].apply(lambda z:str(z).split(", ")).tolist(),index
s1 = pd.DataFrame(s1.stack())
s1.reset_index(inplace=True)
s1= s1[["title",0]]
s1
subset1= subset1.merge(s1,on="title",how="inner")
subset1.rename(columns={0:"director_new"},inplace=True)
```

In [30]:
```
#cast
s2= pd.DataFrame(subset1["cast"].apply(lambda z:str(z).split(", ")).tolist(),index = su
s2 = pd.DataFrame(s2.stack())
s2.reset_index(inplace=True)
s2= s2[["title",0]]
s2
subset1= subset1.merge(s2,on="title",how="inner")
subset1.rename(columns={0:"cast_new"},inplace=True)
```

In [31]:
```
subset1.shape
```

Out[31]:
```
(76839, 15)
```

In [32]:
```
subset1["ds_duo"]= subset1["director_new"] + "-" + subset1["cast_new"]
```

In [33]:
```
duo_frequency= subset1["ds_duo"].value_counts()
duo_frequency.head(3)
```

Out[33]:
```
Rajiv Chilaka-Julie Tejwani    21
Rajiv Chilaka-Rajesh Kava      21
Rajiv Chilaka-Rupa Bhimani     20
Name: ds_duo, dtype: int64
```

Description Analysis

Summary Statistics

In [34]:
```
description_stats= df["description"].describe()
description_stats['length_mean'] = df['description'].str.len().mean()
description_stats['length_min'] = df['description'].str.len().min()
description_stats['length_max'] = df['description'].str.len().max()
description_stats
```

Out[34]:
```
count                                                  8807
unique                                                 8775
top            Paranormal activity at a lush, abandoned prope...
freq                                                      4
length_mean                                      143.303281
length_min                                               61
length_max                                              248
Name: description, dtype: object
```

Most Common Words

In [35]:
```
from collections import Counter
import nltk

# Tokenize the descriptions into individual words
descriptions_tokens = df['description'].str.lower().str.split().sum()

# Count the occurrence of each word
word_count = Counter(descriptions_tokens)

# Get the most common words
most_common_words = word_count.most_common(50)

# Print the most common words
print("Most Common Words:")
for word, count in most_common_words:
    print(f"{word}: {count} occurrences")
```

```
Most Common Words:
a: 11609 occurrences
the: 8106 occurrences
to: 6439 occurrences
and: 6320 occurrences
of: 5273 occurrences
in: 4334 occurrences
his: 3352 occurrences
with: 2261 occurrences
her: 2077 occurrences
an: 1993 occurrences
for: 1782 occurrences
on: 1763 occurrences
their: 1669 occurrences
when: 1512 occurrences
this: 1395 occurrences
from: 1291 occurrences
as: 1224 occurrences
```

```
is: 1111 occurrences
by: 1004 occurrences
after: 993 occurrences
he: 871 occurrences
that: 820 occurrences
who: 807 occurrences
but: 806 occurrences
at: 739 occurrences
young: 717 occurrences
into: 713 occurrences
new: 693 occurrences
–: 606 occurrences
life: 579 occurrences
up: 574 occurrences
they: 540 occurrences
two: 495 occurrences
she: 473 occurrences
family: 454 occurrences
man: 446 occurrences
out: 418 occurrences
woman: 415 occurrences
must: 397 occurrences
are: 382 occurrences
while: 377 occurrences
world: 372 occurrences
love: 372 occurrences
friends: 366 occurrences
about: 353 occurrences
him: 345 occurrences
find: 336 occurrences
one: 328 occurrences
documentary: 313 occurrences
finds: 312 occurrences
```

Sentiment Values Analysis

In [36]:

```
from textblob import TextBlob

# Perform sentiment analysis
df['sentiment'] = df['description'].apply(lambda x: TextBlob(x).sentiment.polarity)

# Print sentiment statistics
print(df['sentiment'].describe())
```

```
count    8807.000000
mean        0.062682
std         0.263750
min        -1.000000
25%        -0.055051
50%         0.023611
75%         0.210000
max         1.000000
Name: sentiment, dtype: float64
```

In [37]:

```
'''
sentiment value < 0 - Negative
sentiment value 0 to 0.5 - Neutral
sentiment value > 0.5 and close to 1 - Positive
'''
```

```
binss = [-2.0,-0.01,0.5,2.0]
label = ["negative","neutral","positive"]

df["emotion"]= pd.cut(df["sentiment"],bins= binss,labels=label)
```

In [38]:
```
df["emotion"].value_counts()
```

Out[38]:
```
neutral     5860
negative    2615
positive     332
Name: emotion, dtype: int64
```

Movies v/s TV Shows

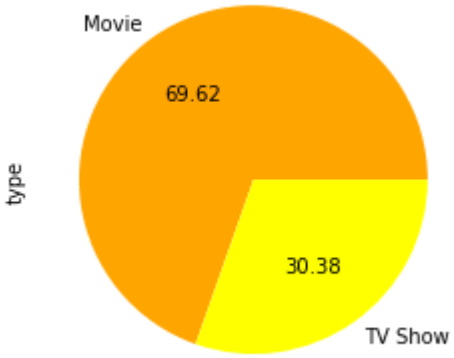
Counts

In [53]:
```
df["type"].value_counts()
```

Out[53]:
```
Movie      6131
TV Show    2676
Name: type, dtype: int64
```

In [52]:
```
df["type"].value_counts().plot(kind="pie",autopct="%.2f",colors=["orange","yellow"])
```

Out[52]:
```
<AxesSubplot:ylabel='type'>
```



Movies and TV Shows Trend

In [65]:
```
# null values
df["date_added"].isna().sum()
```

Out[65]:
```
10
```

In [89]:
```
df[df["date_added"].isna()]
```

Out[89]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	dura
6066	s6067	TV Show	A Young Doctor's Notebook and Other Stories	NaN	Daniel Radcliffe, Jon Hamm, Adam Godley, Chris...	United Kingdom	NaT	2013	TV-MA	Seas
6174	s6175	TV Show	Anthony Bourdain: Parts Unknown	NaN	Anthony Bourdain	United States	NaT	2018	TV-PG	Seas
6795	s6796	TV Show	Frasier	NaN	Kelsey Grammer, Jane Leeves, David Hyde Pierce...	United States	NaT	2003	TV-PG	Seas
6806	s6807	TV Show	Friends	NaN	Jennifer Aniston, Courteney Cox, Lisa Kudrow, ...	United States	NaT	2003	TV-14	Seas
6901	s6902	TV Show	Gunslinger Girl	NaN	Yuuka Nanri, Kanako Mitsuhashi, Eri Sendai, Am...	Japan	NaT	2008	TV-14	Seas
7196	s7197	TV Show	Kikoriki	NaN	Igor Dmitriev	United States	NaT	2010	TV-Y	Seas
7254	s7255	TV Show	La Familia P. Luche	NaN	Eugenio Derbez, Consuelo Duval, Luis Manuel Áv...	United States	NaT	2012	TV-14	Seas
7406	s7407	TV Show	Maron	NaN	Marc Maron, Judd Hirsch, Josh Brener, Nora Zeh...	United States	NaT	2016	TV-MA	Seas
7847	s7848	TV Show	Red vs. Blue	NaN	Burnie Burns, Jason Saldaña, Gustavo Sorola, G...	United States	NaT	2015	NR	Seas

	show_id	type	title	director	cast	country	date_added	release_year	rating	dura
8182	s8183	TV Show	The Adventures of Figaro Pho	NaN	Luke Jurevicius, Craig Behenna, Charlotte Haml...	Australia	NaT	2015	TV-Y7	Seas

Since null values were only 10, upon exploring more on GOOGLE for these movies we found that these movies/TV Shows were removed from netflix. But we got the date of Addition and date of removal , so I have updated the null values with the respective year of addition. REFERENCES- 1) https://usa.newonnetflix.info/info/70281022 2) https://hypebeast.com/2020/5/anthony-bourdain-parts-unknown-netflix-june-release-date 3) https://usa.newonnetflix.info/info/70153412 4) https://usa.newonnetflix.info/info/70204989

In [108…
```
# filling null values
df.loc[6066,"date_added"]= "2014-01-01"
df.loc[6174,"date_added"]= "2020-01-01"
df.loc[6795,"date_added"]= "2016-01-01"
df.loc[6806,"date_added"]= "2015-01-01"
df.loc[6901,"date_added"]= "2016-01-01"
df.loc[7196,"date_added"]= "2017-01-01"
df.loc[7254,"date_added"]= "2015-01-01"
df.loc[7406,"date_added"]= "2017-01-01"
df.loc[7847,"date_added"]= "2014-01-01"
df.loc[8182,"date_added"]= "2014-01-01"
```

In [109…
```
df[df["date_added"].isna()]
```

Out[109…

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	des

In [111…
```
df["date_added"]= pd.to_datetime(df["date_added"])
```

In [119…
```
df["added_year"]= df["date_added"].dt.year
```

In [126…
```
df
```

Out[126…

	show_id	type	title	director	cast	country	date_added	release_year	rating	dura
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	2021-09-25	2020	PG-13	90
1	s2	TV Show	Blood & Water	NaN	Ama Qamata,	South Africa	2021-09-24	2021	TV-MA	Sea

	show_id	type	title	director	cast	country	date_added	release_year	rating	dura
					Khosi Ngema, Gail Mabalane, Thaban...					
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	United States	2021-09-24	2021	TV-MA	1 Se
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	United States	2021-09-24	2021	TV-MA	1 Se
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	2021-09-24	2021	TV-MA	Sea
...	...	...	...	...	...	...	...	...	...	
8802	s8803	Movie	Zodiac	David Fincher	Mark Ruffalo, Jake Gyllenhaal, Robert Downey J...	United States	2019-11-20	2007	R	158
8803	s8804	TV Show	Zombie Dumb	NaN	NaN	United States	2019-07-01	2018	TV-Y7	Sea
8804	s8805	Movie	Zombieland	Ruben Fleischer	Jesse Eisenberg, Woody Harrelson, Emma Stone, ...	United States	2019-11-01	2009	R	88
8805	s8806	Movie	Zoom	Peter Hewitt	Tim Allen, Courteney Cox, Chevy Chase, Kate Ma...	United States	2020-01-11	2006	PG	88
8806	s8807	Movie	Zubaan	Mozez Singh	Vicky Kaushal, Sarah-Jane Dias,	India	2019-03-02	2015	TV-14	111

show_id	type	title	director	cast	country	date_added	release_year	rating	dura
				Raaghav Chanan...					

8807 rows × 16 columns

In [146…

```
movies = df.loc[df["type"]=="Movie"]
movies= pd.DataFrame(movies["added_year"].value_counts())
movies.reset_index(inplace=True)
movies.rename(columns={"index":"year","added_year":"count"},inplace=True)
movies= movies.sort_values(by="year",ascending=True)
```

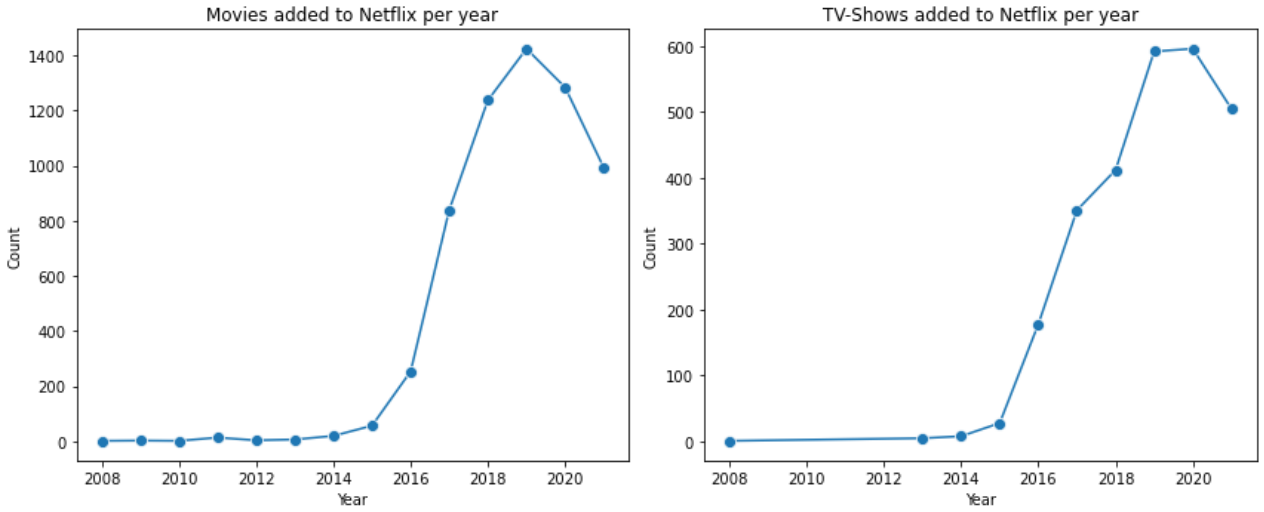
In [145…

```
tv_show = df.loc[df["type"]=="TV Show"]
tv_show = pd.DataFrame(tv_show["added_year"].value_counts())
tv_show.reset_index(inplace=True)
tv_show.rename(columns={"index":"year","added_year":"count"},inplace=True)
tv_show= tv_show.sort_values(by="year",ascending=True)
```

In [151…

```
 plt.figure(figsize=(12, 5))
fig, axes = plt.subplots(nrows=1, ncols=2, figsize=(12, 5))
sns.lineplot(x=movies["year"], y=movies["count"], marker='o', markersize=8, ax=axes[0])
axes[0].set_title("Movies added to Netflix per year")
axes[0].set_xlabel("Year")
axes[0].set_ylabel("Count")
sns.lineplot(x= tv_show["year"], y= tv_show["count"], marker='o', markersize=8, ax=axes
axes[1].set_title("TV-Shows added to Netflix per year")
axes[1].set_xlabel("Year")
axes[1].set_ylabel("Count")
plt.tight_layout()
plt.show()
```

```
<Figure size 864x360 with 0 Axes>
```



Popular Genre by Country

In [167…

```
#There is no null in Genre and Country Column
dfgc = df[["listed_in","country"]]
```

```
dfgc["listed_in"]= dfgc["listed_in"].apply(lambda x: str(x).split(","))
dfgc= dfgc.explode("listed_in").drop_duplicates()

dfgc["country"]= dfgc["country"].apply(lambda x: str(x).split(","))
dfgc= dfgc.explode("country").drop_duplicates()

dfgc
```

```
C:\Users\Test\AppData\Local\Temp/ipykernel_20980/3980877357.py:3: SettingWithCopyWarnin
g:
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
  dfgc["listed_in"]= dfgc["listed_in"].apply(lambda x: str(x).split(","))
```

Out[167…

	listed_in	country
0	Documentaries	United States
1	International TV Shows	South Africa
1	TV Dramas	South Africa
1	TV Mysteries	South Africa
2	Crime TV Shows	United States
…	…	…
8788	Dramas	Montenegro
8788	International Movies	Croatia
8788	International Movies	Slovenia
8788	International Movies	Montenegro
8797	Kids' TV	Indonesia

2441 rows × 2 columns

In [203…

```
new_df = pd.DataFrame(dfgc.groupby("country")["listed_in"].unique())
new_df.reset_index(inplace=True)
new_df.drop(index=0,inplace=True)
new_df.rename(columns={"listed_in":"genre"},inplace=True)
new_df
```

Out[203…

	country	genre
1	Afghanistan	[Documentaries, International Movies]
2	Albania	[Dramas, International Movies]
3	Algeria	[Dramas, Independent Movies, International M…
4	Angola	[Action & Adventure, International Movies]
5	Argentina	[Dramas, Independent Movies, International M…
…	…	…

	country	genre
192	Uruguay	[Docuseries, International TV Shows, Science...
193	Venezuela	[Documentaries, International Movies]
194	Vietnam	[Dramas, International Movies, Romantic Movi...
195	West Germany	[Documentaries, International Movies]
196	Zimbabwe	[Comedies, International Movies, Romantic Mo...

196 rows × 2 columns

Popular Directors by Genre

In [226...

```
dng = df[["director","listed_in","show_id"]]
dng
dng["listed_in"]= dng["listed_in"].apply(lambda x: str(x).split(","))
dng= dng.explode("listed_in").drop_duplicates()
```

```
C:\Users\Test\AppData\Local\Temp/ipykernel_20980/814554416.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
  dng["listed_in"]= dng["listed_in"].apply(lambda x: str(x).split(","))
```

In [227...

```
dng["director"]= dng["director"].apply(lambda x: str(x).split(","))
dng= dng.explode("director").drop_duplicates()
dng
dng = dng.drop(dng[dng['director']=='nan'].index)
dng
```

Out[227...

	director	listed_in	show_id
0	Kirsten Johnson	Documentaries	s1
2	Julien Leclercq	Crime TV Shows	s3
2	Julien Leclercq	International TV Shows	s3
2	Julien Leclercq	TV Action & Adventure	s3
5	Mike Flanagan	TV Dramas	s6
...	...	...	...
8805	Peter Hewitt	Children & Family Movies	s8806
8805	Peter Hewitt	Comedies	s8806
8806	Mozez Singh	Dramas	s8807
8806	Mozez Singh	International Movies	s8807
8806	Mozez Singh	Music & Musicals	s8807

15030 rows × 3 columns

In [228…

```
genre_names = ['Action & Adventure', 'Children & Family Movies', 'Comedies',
'Dramas', 'International Movies', 'International TV Shows',
'Sci-Fi & Fantasy', 'Thrillers']
v_names = ['action', 'family', 'comedies', 'dramas',
'int_movies', 'int_tv', 'scifi', 'thrillers']
data=[]
def genre_wise_directors(genre_name, var_name):
    var_name = dng.loc[(dng['listed_in']==genre_name)].groupby('director')['show_id'].n
    data.append(var_name)
    return data
for genre_name, var_name in zip(genre_names, v_names):
    genre_wise_directors(genre_name, var_name)
data
```

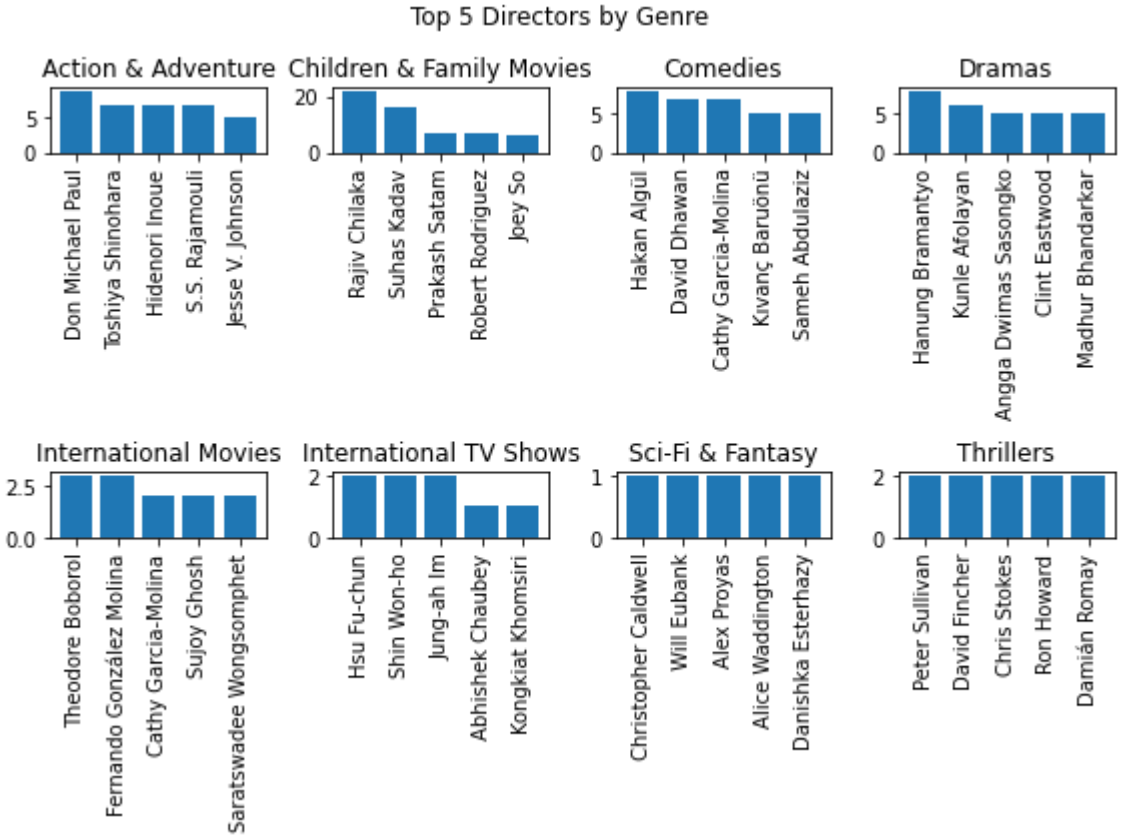
Out[228…

```
[           director  show_id
0   Don Michael Paul        9
1  Toshiya Shinohara        7
2     Hidenori Inoue        7
3     S.S. Rajamouli        7
4   Jesse V. Johnson        5,
          director  show_id
0     Rajiv Chilaka       22
1       Suhas Kadav       16
2     Prakash Satam        7
3  Robert Rodriguez        7
4          Joey So        6,
            director  show_id
0         Hakan Algül        8
1        David Dhawan        7
2  Cathy Garcia-Molina        7
3      Kıvanç Baruönü        5
4     Sameh Abdulaziz        5,
               director  show_id
0      Hanung Bramantyo        8
1        Kunle Afolayan        6
2  Angga Dwimas Sasongko        5
3        Clint Eastwood        5
4     Madhur Bhandarkar        5,
                  director  show_id
0         Theodore Boborol        3
1  Fernando González Molina        3
2      Cathy Garcia-Molina        2
3              Sujoy Ghosh        2
4   Saratswadee Wongsomphet        2,
          director  show_id
0       Hsu Fu-chun        2
1       Shin Won-ho        2
2        Jung-ah Im        2
3   Abhishek Chaubey        1
4  Kongkiat Khomsiri        1,
               director  show_id
0   Christopher Caldwell        1
1           Will Eubank        1
2           Alex Proyas        1
3      Alice Waddington        1
4     Danishka Esterhazy        1,
```

```
          director  show_id
0  Peter Sullivan         2
1    David Fincher        2
2     Chris Stokes        2
3       Ron Howard        2
4     Damián Romay        2]
```

In [239…

```
fig, axes = plt.subplots(2, 4, figsize=(8, 6))
for i, ax in enumerate(axes.flatten()):
    ax.bar(data[i]['director'], data[i]['show_id'])
    ax.set_xticks(np.arange(0, 5))
    ax.set_xticklabels(data[i]['director'], rotation='vertical')
    ax.set_title(f'{genre_names[i]}')
    #ax.set_ylabel('Count of Shows')

plt.suptitle("Top 5 Directors by Genre")
plt.tight_layout()
plt.show()
```



Business Insights

1) Content Strategy: The analysis indicates that the most popular categories on Netflix are International Movies/TV shows, Dramas, and Comedies. This insight suggests that investing in these genres can potentially attract a larger audience and drive viewership.

2) Targeting Specific Countries: The data reveals that Netflix primarily targets the USA, India, and the UK. This insight can inform localization efforts, content acquisition strategies, and marketing campaigns tailored to these specific markets.

3) Duration Consideration: The median duration of movies and TV shows are 100 minutes and 1 season, respectively.

4) Targeted Ratings: Netflix caters to individuals falling into the "TV-MA" and "TV-14" categories.

5) Director and Cast Influence: The most popular director-cast combinations ("Rajiv Chilaka-Julie Tejwani" and "Rajiv Chilaka-Rajesh Kava") provides valuable insights for talent acquisition and content creation. Leveraging successful director-cast pairs can potentially enhance the appeal and success of future productions.

6) Description Optimization: Using words like "Young," "Life," "Family," "Love," "Friends," and focusing on positive sentiment can engage viewers and improve content discoverability.

7) Investment and Trend Analysis: Netflix has invested more on Movies than in TV Shows, but the graph shows upcoming trend in both with time.

Recommendations

1) Strategic Investment in TV Shows: Considering the growing trend of both movies and TV shows, Netflix can strategically allocate more resources and investment in TV shows. This can capitalize on the increasing viewership and demand for TV show content.

2) Genre Localization by Country: By analyzing the "Popular Genre by Country" visualization, Netflix can identify and prioritize genres that are most preferred in each country. This allows for targeted content selection and localization efforts, increasing viewer engagement and satisfaction.

3) Director Selection by Genre: Utilizing the insights from "Popular Director by Genre," Netflix can curate and filter directors based on genre preferences. This enables focused collaborations and content development, aligning with audience preferences and enhancing the quality of the produced content.

4) Positive Sentiment for Enhanced Visibility: Observing that descriptions with a more positive sentiment receive higher visibility and engagement, Netflix can optimize content descriptions to highlight positive elements. This can increase the discoverability and attractiveness of the content to potential viewers.

5) Audience Segmentation and Content Diversification: As Netflix caters to individuals falling into the "TV-MA" and "TV-14" categories, it can further expand its content offerings to cater to the preferences of different age groups. By considering country-specific demographics, such as aging or growing populations, Netflix can diversify its content and engage a broader range of viewers.

6) Multi-Language Support for Diverse Countries: In countries with diverse populations, like India, Netflix can enhance its user experience and expand its reach by providing multi-language support. Offering content in various languages can attract a wider audience and increase engagement among viewers from different linguistic backgrounds.