In [91]:
```
!wget "https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/000/940/original/netflix.csv"
```

```
--2024-04-16 08:35:15--  https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/000/940/original/netflix.csv
Resolving d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)... 18.164.173.110, 18.164.173.58, 18.164.173.18, ...
Connecting to d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)|18.164.173.110|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3399671 (3.2M) [text/plain]
Saving to: ‘netflix.csv.1’

netflix.csv.1       100%[===================>]   3.24M  --.-KB/s    in 0.05s

2024-04-16 08:35:15 (60.2 MB/s) - ‘netflix.csv.1’ saved [3399671/3399671]
```

In [92]:
```
# importing necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [93]:
```
# reading the file
netflix_df = pd.read_csv("netflix.csv")
```

In [94]:
```
# glimpse of netflix_df
netflix_df.head(5)
```

Out[94]:

	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...

Problem Statement :

- Analyze the data and generate insights that could help Netflix decide which type of shows/movies to produce and how to grow the business.

Data Exploration

In [95]:
```
# Analyzing basic metrics
# lets start with observing how many different shows do we have in the netflix dataset given
netflix_df['show_id'].nunique()
```

Out[95]:
```
8807
```

In [96]:
```
# we can achieve the same by using title column as well
```

```
In [96]: netflix_df['title'].nunique()
```

```
Out[96]: 8807
```

```
In [97]: # lets check the shape of the given dataframe
         netflix_df.shape
```

```
Out[97]: (8807, 12)
```

```
In [98]: # lets check the datatypes of each and every column
         netflix_df.dtypes
```

```
Out[98]: show_id         object
         type            object
         title           object
         director        object
         cast            object
         country         object
         date_added      object
         release_year     int64
         rating          object
         duration        object
         listed_in       object
         description     object
         dtype: object
```

```
In [99]: # lets check for null values if any in any columns
         netflix_df.isnull().sum()
```

```
Out[99]: 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 [100…: # lets explore diretor,cast,country,listed_in columns in detail
          netflix_df['director'].head(20)
```

Out[100]:
```
0                                  Kirsten Johnson
1                                              NaN
2                                  Julien Leclercq
3                                              NaN
4                                              NaN
5                                    Mike Flanagan
6                     Robert Cullen, José Luis Ucha
7                                     Haile Gerima
8                                  Andy Devonshire
9                                   Theodore Melfi
10                                             NaN
11                               Kongkiat Komesiri
12                              Christian Schwochow
13                                   Bruno Garotti
14                                             NaN
15                                             NaN
16    Pedro de Echave García, Pablo Azorín Williams
17                                             NaN
18                                       Adam Salky
19                                             NaN
Name: director, dtype: object
```

In [101…
```
netflix_df['cast'].head(20)
```

Out[101]:
```
0                                                   NaN
1     Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...
2     Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...
3                                                   NaN
4     Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...
5     Kate Siegel, Zach Gilford, Hamish Linklater, H...
6     Vanessa Hudgens, Kimiko Glenn, James Marsden, ...
7     Kofi Ghanaba, Oyafunmike Ogunlano, Alexandra D...
8     Mel Giedroyc, Sue Perkins, Mary Berry, Paul Ho...
9     Melissa McCarthy, Chris O'Dowd, Kevin Kline, T...
10                                                  NaN
11    Sukollawat Kanarot, Sushar Manaying, Pavarit M...
12    Luna Wedler, Jannis Niewöhner, Milan Peschel, ...
13    Klara Castanho, Lucca Picon, Júlia Gomes, Marc...
14                                                  NaN
15    Logan Browning, Brandon P. Bell, DeRon Horton,...
16                                                  NaN
17    Luis Ernesto Franco, Camila Sodi, Sergio Goyri...
18    Freida Pinto, Logan Marshall-Green, Robert Joh...
19    Blanca Suárez, Iván Marcos, Óscar Casas, Adriá...
Name: cast, dtype: object
```

In [102…
```
netflix_df['country'].head(20)
```

Out[102]:
```
0                                        United States
1                                         South Africa
2                                                  NaN
3                                                  NaN
4                                                India
5                                                  NaN
6                                                  NaN
7     United States, Ghana, Burkina Faso, United Kin...
8                                           United Kingdom
9                                         United States
10                                                 NaN
11                                                 NaN
12                               Germany, Czech Republic
13                                                 NaN
14                                                 NaN
15                                        United States
16                                                 NaN
17                                               Mexico
18                                                 NaN
19                                                 NaN
Name: country, dtype: object
```

In [103…
```
netflix_df['listed_in'].head(20)
```

Out[103]:
```
0                                        Documentaries
1          International TV Shows, TV Dramas, TV Mysteries
2     Crime TV Shows, International TV Shows, TV Act...
3                                Docuseries, Reality TV
4     International TV Shows, Romantic TV Shows, TV ...
5                     TV Dramas, TV Horror, TV Mysteries
6                               Children & Family Movies
7      Dramas, Independent Movies, International Movies
8                           British TV Shows, Reality TV
9                                     Comedies, Dramas
10    Crime TV Shows, Docuseries, International TV S...
11    Crime TV Shows, International TV Shows, TV Act...
12                          Dramas, International Movies
13                  Children & Family Movies, Comedies
14         British TV Shows, Crime TV Shows, Docuseries
15                               TV Comedies, TV Dramas
16                  Documentaries, International Movies
17    Crime TV Shows, Spanish-Language TV Shows, TV ...
18                                           Thrillers
19    International TV Shows, Spanish-Language TV Sh...
Name: listed_in, dtype: object
```

Observation

- So here each row of (director,cast,country,listed_in) columns, data entered is inconsistent.
- In few rows we have single entry for each row but in some rows we have list of entries in them.
- So lets unnest these columns one by one

Data cleaning wherever required

Unnesting directors column

In [104…
```
# extracting directors from the directors column,splitting them into list,and converting the result into list of lists
directors_list=netflix_df['director'].apply(lambda x: str(x).split(', ')).tolist()
# creating a dataframe frm the list of lists using titles as index
directors_df=pd.DataFrame(directors_list,index = netflix_df['title'])
# stacking the dataframe to convert the columns into rows
directors_df=directors_df.stack()
# resetting the index to move titles from index to a column
directors_df=pd.DataFrame(directors_df.reset_index())
#renaming the column to appropriate name
directors_df.rename(columns={0:'Directors'},inplace=True)
# dropping the unecessary column  level_1
directors_df.drop(['level_1'],axis=1,inplace=True)
# displaying the first few rows of the dataframe
directors_df.head()
```

Out[104]:

	title	Directors
0	Dick Johnson Is Dead	Kirsten Johnson
1	Blood & Water	nan
2	Ganglands	Julien Leclercq
3	Jailbirds New Orleans	nan
4	Kota Factory	nan

Unnesting cast column

In [105…
```
cast_list=netflix_df['cast'].apply(lambda x: str(x).split(', ')).tolist()
actors_df=pd.DataFrame(cast_list,index=netflix_df['title'])
actors_df=actors_df.stack()
actors_df=pd.DataFrame(actors_df.reset_index())
actors_df.rename(columns={0:'Actors'},inplace=True)
actors_df.drop(['level_1'],axis=1,inplace=True)
actors_df.head()
```

Out[105]:

	title	Actors
0	Dick Johnson Is Dead	nan
1	Blood & Water	Ama Qamata
2	Blood & Water	Khosi Ngema
3	Blood & Water	Gail Mabalane
4	Blood & Water	Thabang Molaba

Unnesting listed_in column

In [106…
```
genre_list=netflix_df['listed_in'].apply(lambda x: str(x).split(', ')).tolist()
genre_df=pd.DataFrame(genre_list,index=netflix_df['title'])
genre_df=genre_df.stack()
genre_df=pd.DataFrame(genre_df.reset_index())
genre_df.rename(columns={0:'Genre'},inplace=True)
genre_df.drop(['level_1'],axis=1,inplace=True)
genre_df.head()
```

Out[106]:

	title	Genre
0	Dick Johnson Is Dead	Documentaries
1	Blood & Water	International TV Shows
2	Blood & Water	TV Dramas
3	Blood & Water	TV Mysteries
4	Ganglands	Crime TV Shows

Unnesting country column

In [107…

```
country_list=netflix_df['country'].apply(lambda x: str(x).split(', ')).tolist()
country_df=pd.DataFrame(country_list,index=netflix_df['title'])
country_df=country_df.stack()
country_df=pd.DataFrame(country_df.reset_index())
country_df.rename(columns={0:'countries'},inplace=True)
country_df.drop(['level_1'],axis=1,inplace=True)
country_df.head()
```

Out[107]:

	title	countries
0	Dick Johnson Is Dead	United States
1	Blood & Water	South Africa
2	Ganglands	nan
3	Jailbirds New Orleans	nan
4	Kota Factory	India

Merging all the sub dataframes and creating new_df

In [108…

```
netflix_df1 = directors_df.merge(actors_df,on =['title'],how='inner')
netflix_df2 = netflix_df1.merge(genre_df,on =['title'],how='inner')
new_df = netflix_df2.merge(country_df,on =['title'],how='inner')
new_df.head()
```

Out[108]:

	title	Directors	Actors	Genre	countries
0	Dick Johnson Is Dead	Kirsten Johnson	nan	Documentaries	United States
1	Blood & Water	nan	Ama Qamata	International TV Shows	South Africa
2	Blood & Water	nan	Ama Qamata	TV Dramas	South Africa
3	Blood & Water	nan	Ama Qamata	TV Mysteries	South Africa
4	Blood & Water	nan	Khosi Ngema	International TV Shows	South Africa

Finally merging the new_df with the original dataframe that is netflix_df

In [109…

```
# lets merge the new_df with the main netflix_df and get the new 'df'
df = new_df.merge(netflix_df[['show_id','type','title','date_added','release_year','rating','duration']],on = 'title',how = 'left')
```

```
#lets check the shape of newly created df
df.shape
```

Out[109]:
```
(201991, 11)
```

Handling null values

In [110…
```
# lets check for null values
df.isnull().sum()
```

Out[110]:
```
title             0
Directors         0
Actors            0
Genre             0
countries         0
show_id           0
type              0
date_added      158
release_year      0
rating           67
duration          3
dtype: int64
```

Observation

- Though there are no null values in Directors,Actors,countries column there are certain entries in 'nan' which is a string
- so lets replace all of them with more meaningful entry

In [111…
```
# lets replace all the 'nan' values in the Actors,Directors,countries column with Unknown Actor,Unknown Director,Unknown Country  respectively
df['Actors'].replace(['nan'],['Unknown Actor'],inplace=True)
df['Directors'].replace(['nan'],['Unknown Director'],inplace=True)
df['countries'].replace(['nan'],['Unknown Country'],inplace=True)
```

- duration - 3
- rating - 67
- date_addded - 158

Lets work on duration column and clean it for further data manipulation and analysis

In [112…
```
# since the duration column has only 3 null values in them lets drop them
df.dropna(subset=['duration'],inplace=True)
# now let's see how duration column is grouped into
df['duration'].value_counts()
```

Out[112]:
```
duration
1 Season     35035
2 Seasons     9559
3 Seasons     5084
94 min        4343
106 min       4040
              ...
3 min            4
5 min            3
11 min           2
8 min            2
9 min            2
Name: count, Length: 220, dtype: int64
```

In [113…
```
# Split the 'duration_copy' column into two parts: the number and the word "Seasons"
df[['duration_mins','modes']] = df['duration'].str.split(n=1,expand=True)
df.head()
```

Out[113]:

	title	Directors	Actors	Genre	countries	show_id	type	date_added	release_year	rating	duration	duration_mins	modes
0	Dick Johnson Is Dead	Kirsten Johnson	Unknown Actor	Documentaries	United States	s1	Movie	September 25, 2021	2020	PG-13	90 min	90	min
1	Blood & Water	Unknown Director	Ama Qamata	International TV Shows	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	2	Seasons
2	Blood & Water	Unknown Director	Ama Qamata	TV Dramas	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	2	Seasons
3	Blood & Water	Unknown Director	Ama Qamata	TV Mysteries	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	2	Seasons
4	Blood & Water	Unknown Director	Khosi Ngema	International TV Shows	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	2	Seasons

In [114…
```
# checking the data type of 'duration_mins'
df['duration_mins'].dtypes
```

Out[114]:
```
dtype('O')
```

In [115…
```
# conversion of mins column to int from object data type
df['duration_mins'] = df['duration_mins'].astype(int)
```

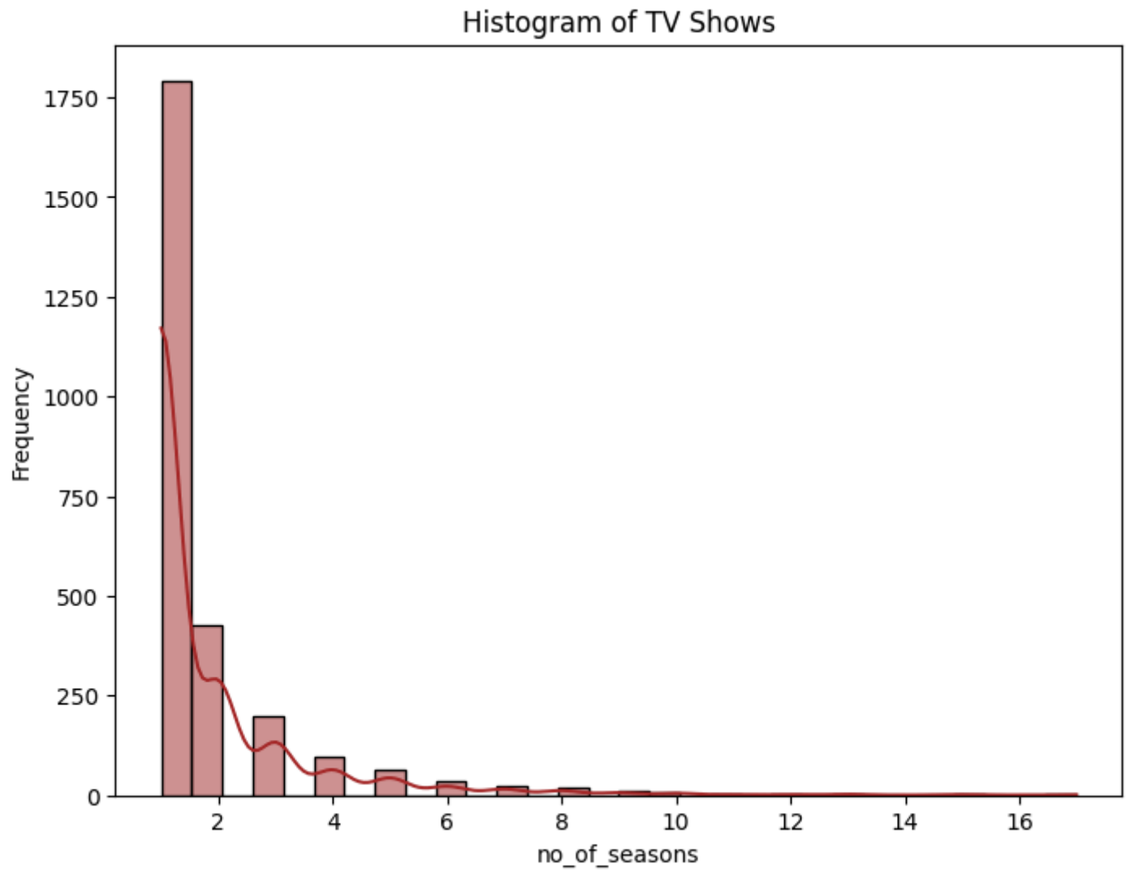
In [116…
```
#creating a separate data frame to understand frequency of TV Shows of different seasons
shows_df=df[df['type']=='TV Show']
shows_df_copy = shows_df.copy()
# renaming the 'duration_mins' column to ''no_of_seasons''
shows_df_copy.rename(columns = {'duration_mins':'no_of_seasons'},inplace =True)
shows_df_copy = shows_df_copy[['title','no_of_seasons']]
shows_df_copy = shows_df_copy.drop_duplicates().reset_index()
shows_df_copy.drop(columns = {'index'},inplace =True)
shows_df_copy.head()
```

Out[116]:

	title	no_of_seasons
0	Blood & Water	2
1	Ganglands	1
2	Jailbirds New Orleans	1
3	Kota Factory	2
4	Midnight Mass	1

In [117…
```
# Plot the histogram for TV Shows
plt.figure(figsize=(8, 6))
sns.histplot(data=shows_df_copy, x='no_of_seasons', bins=30, color='brown',kde=True)
```

```
plt.xlabel('no_of_seasons')
plt.ylabel('Frequency')
plt.title('Histogram of TV Shows')
plt.show()
```



Observation

- Most of the TV Shows on Netflix have only 1 season
- Next ranks between 2 to 4 seasons
- Seasons greater than 4 are very less

In [118…

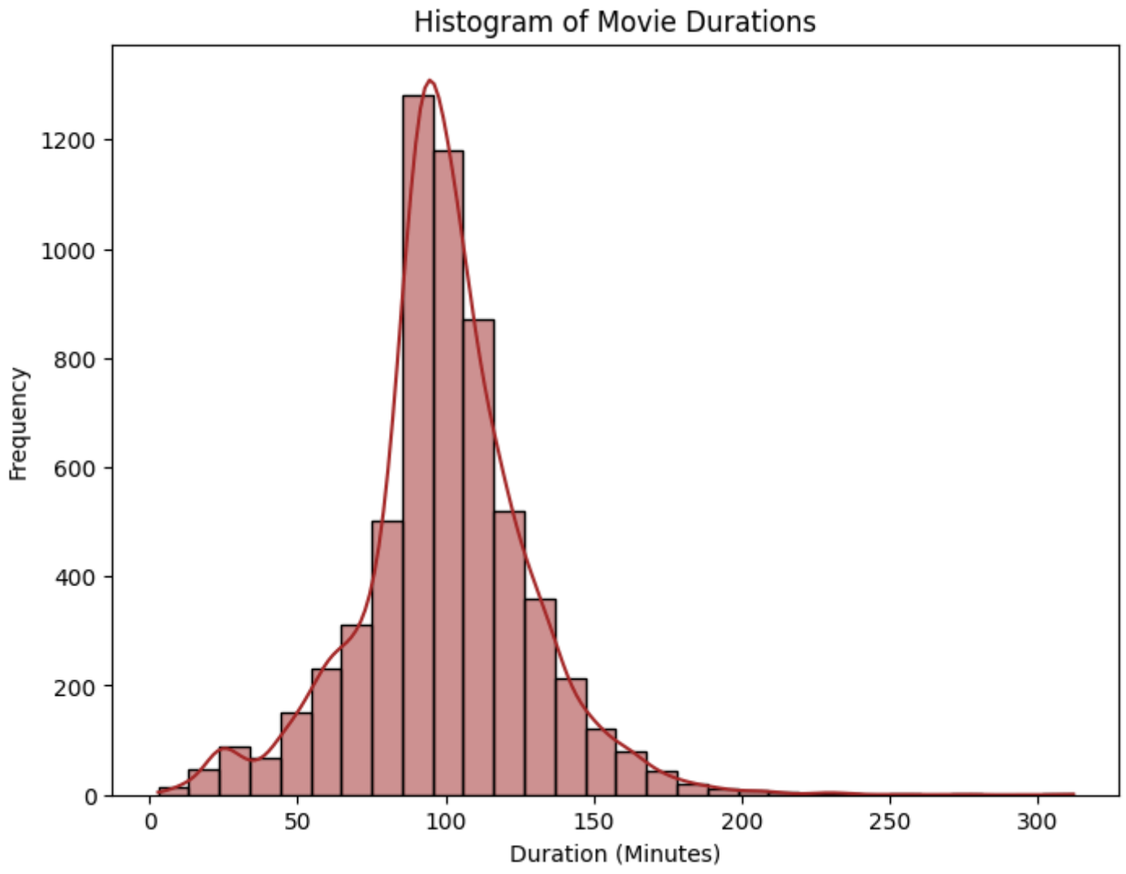
```
#creating a movies dataframe separately
movies_df = df[df['type']=='Movie']
movies_df_copy = movies_df.copy()
movies_df_copy = movies_df_copy[['title','duration_mins']]
movies_df_copy = movies_df_copy.drop_duplicates()
movies_df_copy=movies_df_copy.reset_index()
movies_df_copy.drop(columns = 'index',inplace =True)
movies_df_copy.head()
```

Out[118]:

	title	duration_mins
0	Dick Johnson Is Dead	90
1	My Little Pony: A New Generation	91
2	Sankofa	125
3	The Starling	104
4	Je Suis Karl	127

In [119…

```
# Plot the histogram for 'duration_mins'
plt.figure(figsize=(8, 6))
sns.histplot(data=movies_df_copy, x='duration_mins', bins=30, color='brown', kde=True)
plt.xlabel('Duration (Minutes)')
plt.ylabel('Frequency')
plt.title('Histogram of Movie Durations')
plt.show()
```



Observation

- Movies and its duration distribution :
- Majority of the content falls between **< 90 and 90-120** mins.
- Longer content **> 180** min is less common

Lets work on rating column and clean it for further data manipulation and analysis

In [120…
```
df['rating'].unique()
```

Out[120]:
```
array(['PG-13', 'TV-MA', 'PG', 'TV-14', 'TV-PG', 'TV-Y', 'TV-Y7', 'R',
       'TV-G', 'G', 'NC-17', 'NR', nan, 'TV-Y7-FV', 'UR'], dtype=object)
```

In [121…
```
# Replacing all the 67 null values in rating column by 'NR'
df['rating'].fillna('NR', inplace=True)
```

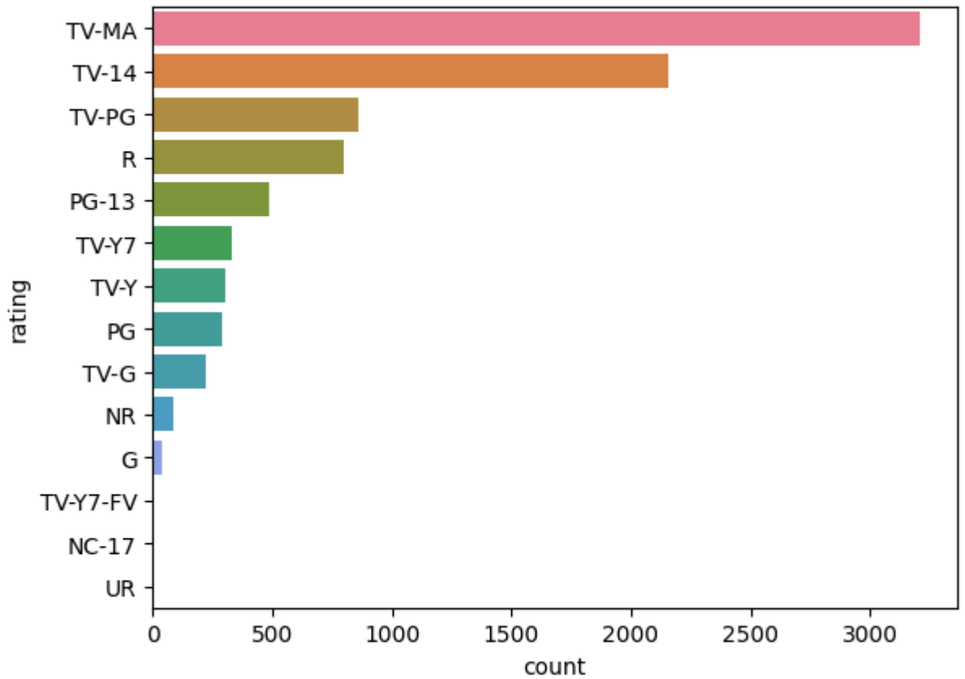
In [122…
```
# lets see how both movies and tv shows rated

# creating a rating_df with only title and rating columns
rating_df = df[['title','rating']]
# creating a copy of it to drop duplicates
rating_df_copy = rating_df.copy()
rating_df_copy.drop_duplicates(inplace =True)
# counting how many shows fall under each rating category
rating_df_copy = rating_df_copy['rating'].value_counts().reset_index()
# renaming the index column which is created after resetting index in the previous step
rating_df_copy
```

Out[122]:

	rating	count
0	TV-MA	3207
1	TV-14	2160
2	TV-PG	863
3	R	799
4	PG-13	490
5	TV-Y7	334
6	TV-Y	307
7	PG	287
8	TV-G	220
9	NR	84
10	G	41
11	TV-Y7-FV	6
12	NC-17	3
13	UR	3

In [123…
```
sns.barplot(data = rating_df_copy ,y='rating',x='count',hue='rating')
```

Out[123]:
```
<Axes: xlabel='count', ylabel='rating'>
```



Insights

- Most common ratings :
- TV-MA and TV-14 are the most common ratings with **3207** and **2160** shows respectively.
- This implies content delivered on netflix is targeted towards adults and older teenagers
- Diverse range of ratings spanning between mature (R, NC-17) to family friendly (TV-Y,TV-G) indicates audiences from different age groups

Lets work on **date_added** column and clean it for further data manipulation and analysis

In [124…
```
# just to understand the initial format of values in df['date_added'] column
df['date_added'].head()
```

Out[124]:
```
0    September 25, 2021
1    September 24, 2021
2    September 24, 2021
3    September 24, 2021
4    September 24, 2021
Name: date_added, dtype: object
```

In [125…
```
# Assuming df is your DataFrame with the 'date_added' column

# Convert 'date_added' column to datetime format
df['date_added'] = pd.to_datetime(df['date_added'], format='mixed')

# Check the datatype of the 'date_added' column after conversion
print(df['date_added'].dtypes)

# Display the first few entries of the converted column
print(df['date_added'].head())
```

```
datetime64[ns]
0   2021-09-25
1   2021-09-24
2   2021-09-24
3   2021-09-24
4   2021-09-24
Name: date_added, dtype: datetime64[ns]
```

Lets change the data type of release year from int to datetime

In [126…
```
# initial dtype of release year-----int64
df['release_year'].dtypes
# converting the release year which is in int64 dtype to date format
df['release_year'] = pd.to_datetime(df['release_year'],format = '%Y')
```

In [127…
```
# imputing the null values in the date_added column by finding the mode of date_added in each release year group
mode_date_by_year = df.groupby('release_year')['date_added'].apply(lambda x : x.mode().iloc[0])
mode_date_by_year.head()
```

Out[127]:
```
release_year
1925-01-01   2018-12-30
1942-01-01   2017-03-31
1943-01-01   2017-03-31
1944-01-01   2017-03-31
1945-01-01   2017-03-31
Name: date_added, dtype: datetime64[ns]
```

In [128…
```
# Loop through each (release_year, mode_date) pair in the dictionary mode_date_by_year.items()
for release_year,mode_date in mode_date_by_year.items():
  # For each release_year, fill missing values in the 'date_added' column with the corresponding mode_date
  df.loc[ df['release_year'] == release_year , 'date_added'] = df.loc[ df['release_year'] == release_year , 'date_added'].fillna(mode_date)
```

In [129…
```
# lets add few more columns to the existing dataframe

df['date'] = df['date_added'].dt.day
df['month'] = df['date_added'].dt.month
df['year'] = df['date_added'].dt.year
df['week'] = df['date_added'].dt.isocalendar().week
df['week_day'] = df['date_added'].dt.strftime('%A')
```

In [130…
```
# finally checking if there are any null values in any of the columns of our df
df.isnull().sum()
```

Out[130]:
```
title           0
Directors       0
Actors          0
Genre           0
countries       0
show_id         0
type            0
date_added      0
release_year    0
rating          0
duration        0
duration_mins   0
modes           0
date            0
month           0
year            0
week            0
week_day        0
dtype: int64
```

In [131…
```
# After data cleaning and adding necessary columns lets just check the shape of our df
df.shape
```

Out[131]:
```
(201988, 18)
```

In [132…
```
# lets drop the unecessary columns if any
df.drop(columns = 'modes',inplace =True)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 201988 entries, 0 to 201990
Data columns (total 17 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   title          201988 non-null  object
 1   Directors      201988 non-null  object
 2   Actors         201988 non-null  object
 3   Genre          201988 non-null  object
 4   countries      201988 non-null  object
 5   show_id        201988 non-null  object
 6   type           201988 non-null  object
 7   date_added     201988 non-null  datetime64[ns]
 8   release_year   201988 non-null  datetime64[ns]
 9   rating         201988 non-null  object
 10  duration       201988 non-null  object
 11  duration_mins  201988 non-null  int64
 12  date           201988 non-null  int32
 13  month          201988 non-null  int32
 14  year           201988 non-null  int32
 15  week           201988 non-null  UInt32
 16  week_day       201988 non-null  object
dtypes: UInt32(1), datetime64[ns](2), int32(3), int64(1), object(10)
memory usage: 24.8+ MB
```

- **So lets see the range of years that we have in hand in the given dataset**

In [133…
```
print(df['release_year'].min()) # ------> These are the years during which movies premiered theatrically and TV Shows were released
print(df['release_year'].max())
print(df['release_year'].max()-df['release_year'].min())
print(df['year'].min()) # ------> These years talk about Netflix releases
print(df['year'].max())
print(df['year'].max()-df['year'].min())
```

```
1925-01-01 00:00:00
2021-01-01 00:00:00
35064 days 00:00:00
2008
2021
13
```

Observation

- we have data of shows that was released from 1925 to 2021
- Similarly they had OTT releases spanning from 2008 till 2021 so far

Let's check number of unique values in each column after cleaning the data and handling missing values

In [134…
```
df.nunique()
```

Out[134]:
```
title           8804
Directors       4993
Actors         36440
Genre             42
countries        128
show_id         8804
type               2
date_added      1714
release_year      74
rating            14
duration         220
duration_mins    210
date              31
month             12
year              14
week              53
week_day           7
dtype: int64
```

Lets Analyze the data further

1. What type of content is available in different countires?
2. How Movies released per year changed over the last 20-30 years?
3. Does Netflix has more focus on TV Shows than Movies?
4. Comparision of TV Shows vs Movies
5. When is the Best time to launch a TV Show?
6. Which genre movies are more popular or produced more?
7. Analysis of Actors / Directors
8. Time taken for OTT release after a movie or a tv show had its Theatrical or Television release

1. What type of content is available in different countries?

In [135…
```
# lets start with exploring type column
df['type'].unique()
```

Out[135]:
```
array(['Movie', 'TV Show'], dtype=object)
```

- **So we have content delivered through both Movies and TV shows on Netflix**

In [136…
```
# filtering the dataframe by excluding those rows where countries data is Unknown Counrty and incluidng year of release from 1991 that's last 30 years data

country_content = df[(df['countries'] != 'Unknown Country') & (df['release_year'].dt.year >= 1991)].groupby(['countries','type'])['title'].nunique().sort_values(ascending=False)
country_content = country_content.reset_index()
country_content.rename(columns = {'title':'no_of_releases'},inplace = True)
country_content.head()
```

Out[136]:

	countries	type	no_of_releases
0	United States	Movie	2618
1	United States	TV Show	926
2	India	Movie	910
3	United Kingdom	Movie	516
4	Canada	Movie	317

Observation

- Here we can observe different content availability in each country in the last 30 years and number of releases respectively

Understanding which country has more number of releases irrespective of type of content

In [137…
```
# finding the total no_of releases in each country
country_total_releases = country_content.groupby('countries')['no_of_releases'].sum().reset_index()
#ranking the countires based on the number of releases
country_total_releases['rank'] = country_total_releases['no_of_releases'].rank(ascending=False)
#sorting it out based on ranking
country_total_releases = country_total_releases.sort_values('rank', ascending = True).reset_index()
country_total_releases.drop(columns = 'index',inplace = True)
country_total_releases.head()
```

Out[137]:

	countries	no_of_releases	rank
0	United States	3544	1.0
1	India	994	2.0
2	United Kingdom	785	3.0
3	Canada	442	4.0
4	France	386	5.0

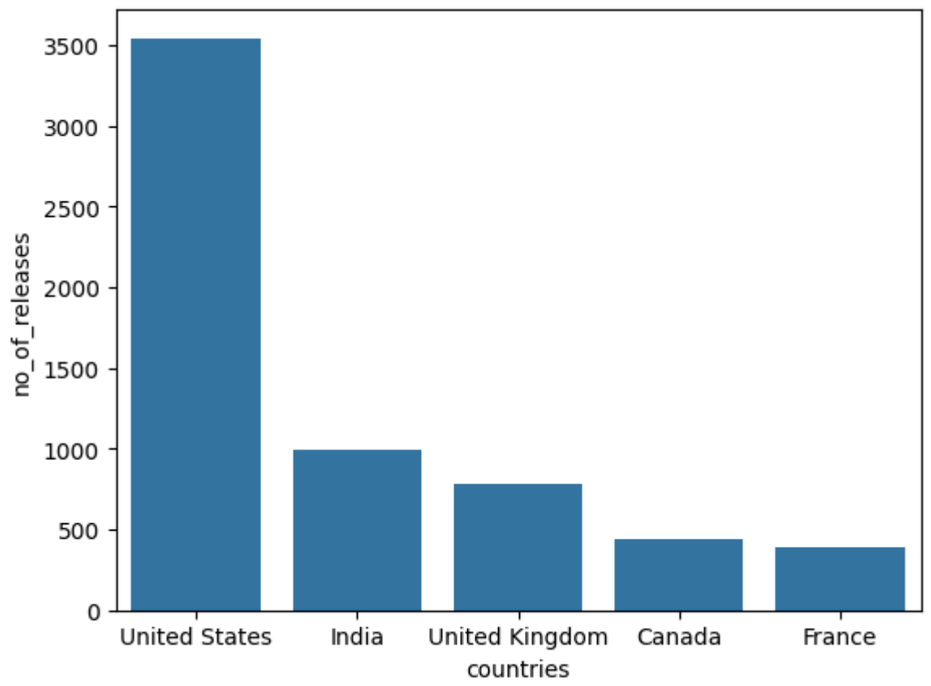
Lets analyse Top 5 countries in terms of there number of releases

In [138…
```
# top 5 countires with highest Netflix releases
top_5_countries = country_total_releases[country_total_releases['rank']<= 5].sort_values('rank',ascending = True)
```

```
print(top_5_countries)
sns.barplot(data = top_5_countries,x = 'countries',y = 'no_of_releases')
```

```
        countries  no_of_releases  rank
0   United States            3544   1.0
1           India             994   2.0
2  United Kingdom             785   3.0
3          Canada             442   4.0
4          France             386   5.0
```

Out[138]: <Axes: xlabel='countries', ylabel='no_of_releases'>



Lets try to further observe in top5 countries

- How many releases are there in each content type?
- Which content type has most number of releases either Movie or TV Shows?

In [139…

```
# creating a dataframe by merging top5_countries with the country_content dataframe,
# to observe the type of content and no_of_releases in only TOP5 countries
merged_df = top_5_countries.merge(country_content,on = 'countries',how = 'left' )
# creating a copy of merged_df by picking only the necessary columns and forming a new dataframe called derived_df
derived_df = merged_df[['countries','type','no_of_releases_y']].copy()
# renaming the column no_of_releases_y into no_of_releases
derived_df.rename(columns = {'no_of_releases_y':'no_of_releases'},inplace = True)
derived_df
```

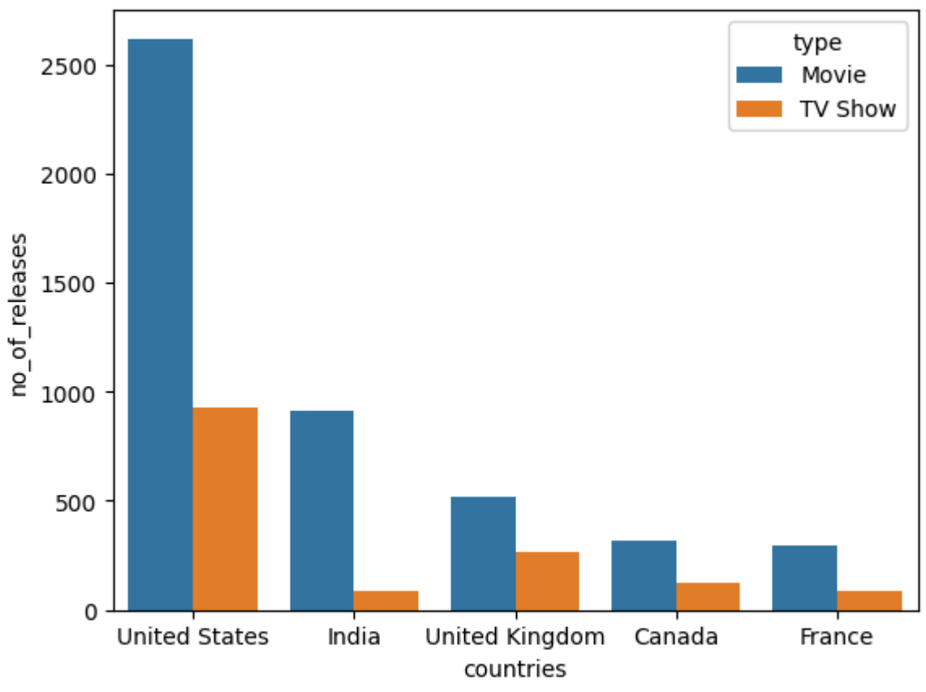
Out[139]:

	countries	type	no_of_releases
0	United States	Movie	2618
1	United States	TV Show	926
2	India	Movie	910
3	India	TV Show	84
4	United Kingdom	Movie	516
5	United Kingdom	TV Show	269
6	Canada	Movie	317
7	Canada	TV Show	125
8	France	Movie	296
9	France	TV Show	90

- *Using a bar plot, let's visualize the most frequently released content types in each country.*

In [140…
```
sns.barplot(data = derived_df,x = 'countries',y ='no_of_releases',hue='type')
```

Out[140]:
```
<Axes: xlabel='countries', ylabel='no_of_releases'>
```



Insights

- In all the top 5 countires movies outnumber tv shows indicating a strong preference for movies
- United States leads in Movie production
- With **2618** movie releases showcasing a thriving movie industry

- TV Shows add diversity although movies dominate, tv shows still contribute significantly offereing diverse content options for viewers
- India ranks high in movie releases
- India ranks second in with 910 movie releases, indicating its growing influence in global cinema
- TV Shows in India have a room to grow
- United kingdom balances content portfolio
- shows a balanced approach with substantial releases in both movies **516** and tv shows **269**
- Similary Canada with movies **317** and TV shows **125**
- France focuses on **296** movies

2. How has the number of movies released per year changed over the last 20-30 years?

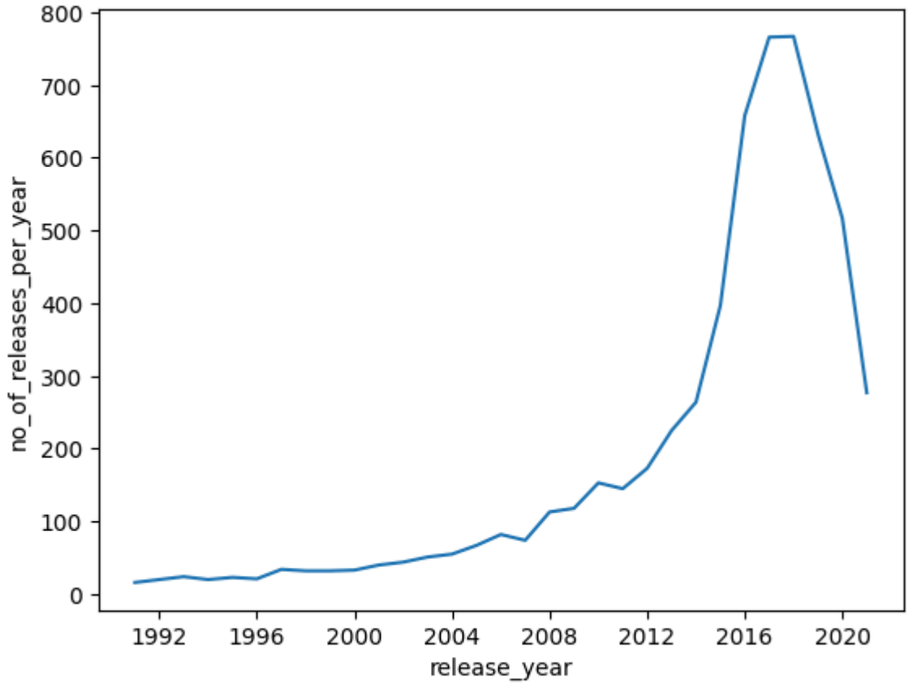
In [141…
```
year_based = df[(df['release_year'].dt.year >= 1991) & (df['type']=='Movie')]
yearly_release = year_based.groupby(['release_year'])['title'].nunique().sort_values(ascending = False)
yearly_release = yearly_release.reset_index()
yearly_release.rename(columns = {'title':'no_of_releases_per_year'},inplace = True)
yearly_release.head()
```

Out[141]:

	release_year	no_of_releases_per_year
0	2018-01-01	767
1	2017-01-01	766
2	2016-01-01	658
3	2019-01-01	633
4	2020-01-01	517

In [142…
```
# line graph to understand the trend of movies across last 30 release years
sns.lineplot(data = yearly_release,x = 'release_year',y = 'no_of_releases_per_year')
```

Out[142]:
```
<Axes: xlabel='release_year', ylabel='no_of_releases_per_year'>
```



Insights

- Years **2016-2018** saw the highest releases per year with **658 to 767** indicating peak content production this time
- Gradual **decrease** in the number of releases from **2018 to 2021** maybe due to corona pandemic
- **Early 2000's** show **moderate activity 35 to 55** releases per year
- Minimal releases in the earlier years that is **before 2000's**

3. Does Netflix has more focus on TV Shows than movies in recent years

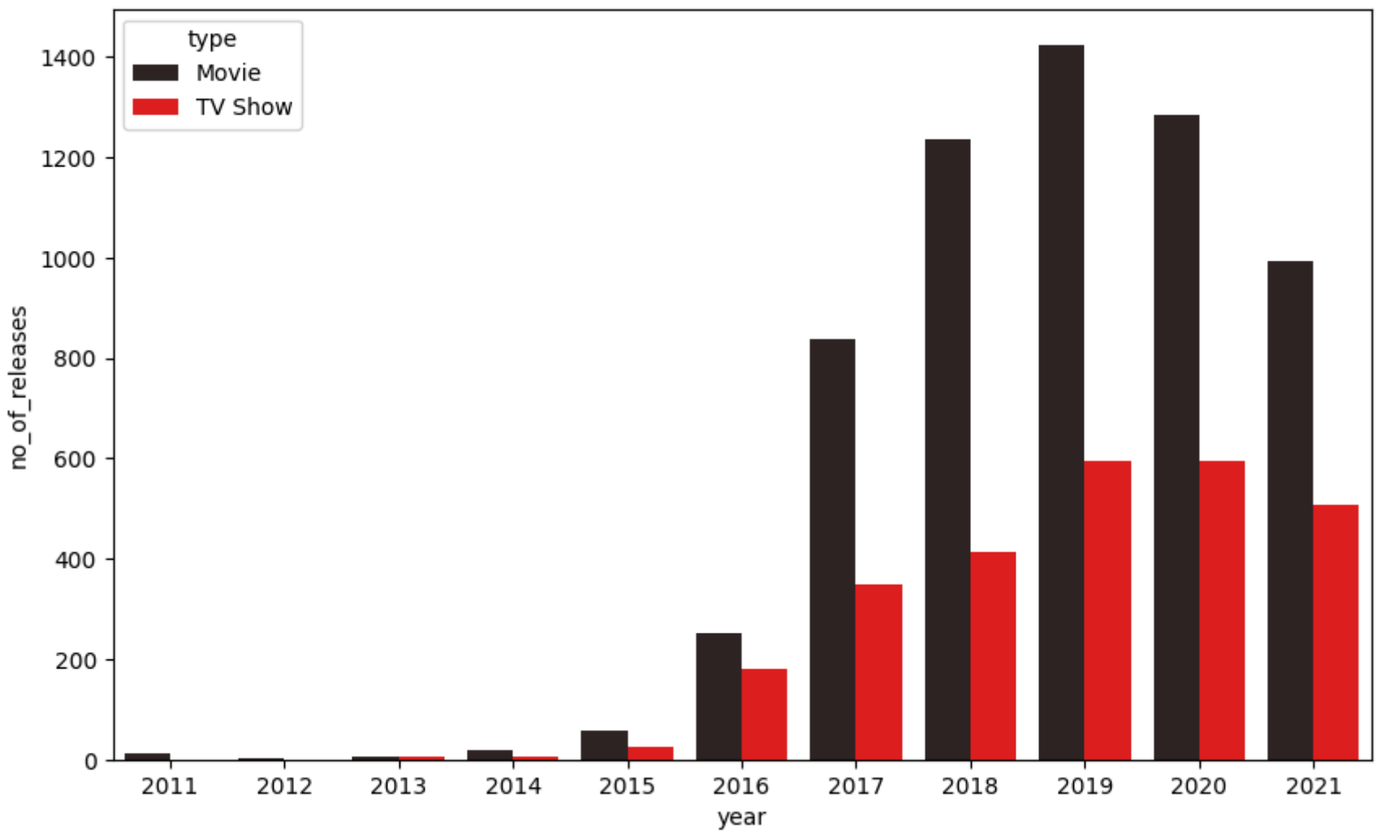
In [143…
```
# lets observe the data of last 10 years of Netflix releases
recent_data = df[df['year'] >= 2011][['type','title','year']]
recent_data = recent_data.groupby(['year','type'])['title'].nunique().reset_index()
recent_data.rename(columns={'title':'no_of_releases'},inplace =True)
recent_data.head()
```

Out[143]:

	year	type	no_of_releases
0	2011	Movie	13
1	2012	Movie	3
2	2013	Movie	6
3	2013	TV Show	5
4	2014	Movie	19

In [144…
```
plt.figure(figsize=(10,6))
sns.barplot(data = recent_data,x = 'year',y = 'no_of_releases',hue = 'type',palette='dark:r')
```

Out[144]: <Axes: xlabel='year', ylabel='no_of_releases'>



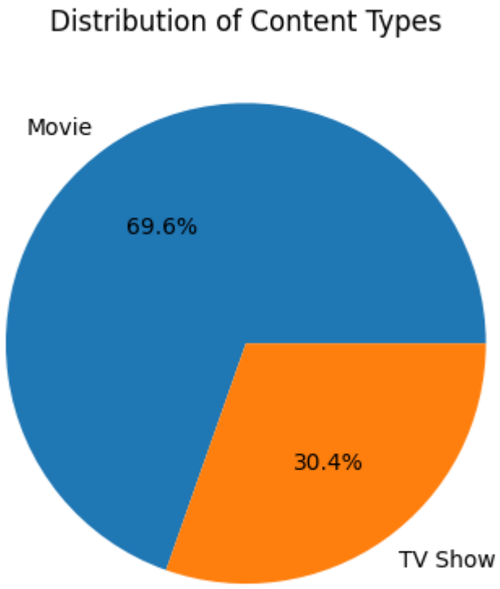
Insights

- In the last 5-6 from 2015 to 2021,there has been a noticeable trend:
- Netflix has had greater Movie releases than TV shows
- When considering the last 10 years,there has been an increase in TV Show releases as well
- This suggests that Netflix is not only mainataining a strong focus on movie releases but also actively improving its TV show releases over time

4. Comparison of Tv shows vs Movies.

- **a. Overall comparision and understanding the distribution**

In [145…
```
type_counts = df.groupby('type')['title'].nunique()
plt.pie(type_counts,labels = type_counts.index,autopct='%1.1f%%')
plt.title('Distribution of Content Types')
plt.show()
```



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

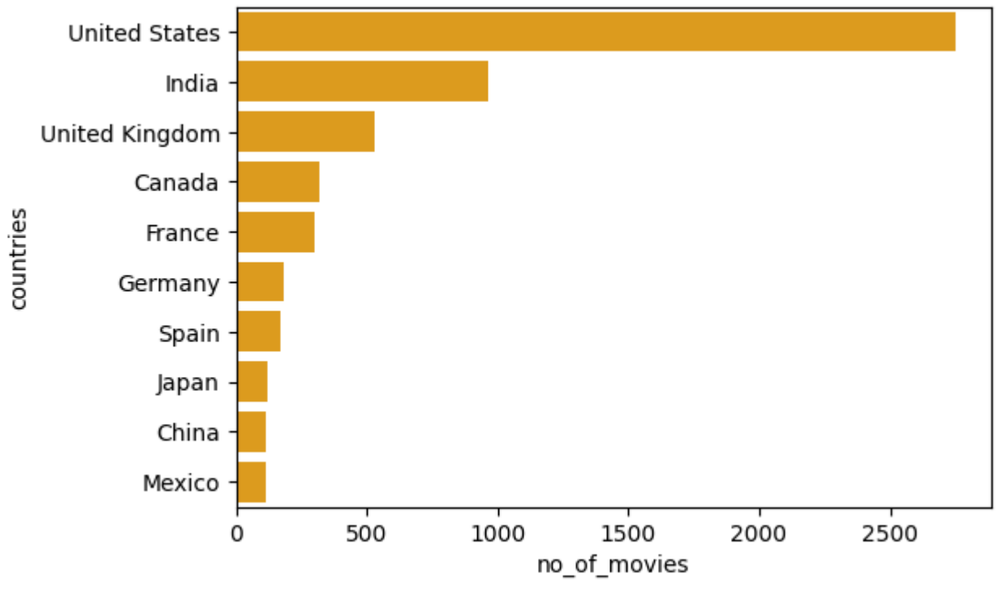
In [146…
```
country_movie_count =df[(df['countries']!='Unknown Country')&(df['type']=='Movie')].groupby('countries')['title'].nunique().sort_values(ascending = False).reset_index()
country_movie_count.rename(columns = {'title':'no_of_movies'},inplace =True)
country_movie_count = country_movie_count.head(10)
country_movie_count
```

Out[146]:

	countries	no_of_movies
0	United States	2748
1	India	962
2	United Kingdom	532
3	Canada	319
4	France	303
5	Germany	182
6	Spain	171
7	Japan	119
8	China	114
9	Mexico	111

In [147…
```
plt.figure(figsize=(6,4))
sns.barplot(data = country_movie_count, x= 'no_of_movies',y = 'countries',color = 'orange')
```

Out[147]:
```
<Axes: xlabel='no_of_movies', ylabel='countries'>
```



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

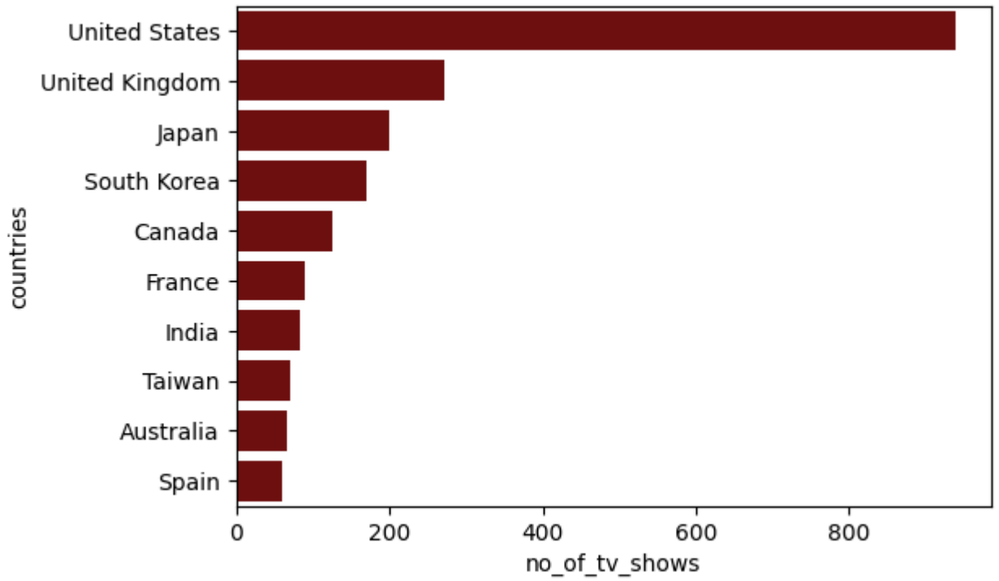
In [148…
```
country_tv_show_count = df[(df['countries']!='Unknown Country')&(df['type']=='TV Show')].groupby('countries')['title'].nunique().sort_values(ascending = False).reset_index()
country_tv_show_count.rename(columns = {'title':'no_of_tv_shows'},inplace =True)
country_tv_show_count = country_tv_show_count.head(10)
country_tv_show_count
```

Out[148]:

	countries	no_of_tv_shows
0	United States	938
1	United Kingdom	272
2	Japan	199
3	South Korea	170
4	Canada	126
5	France	90
6	India	84
7	Taiwan	70
8	Australia	66
9	Spain	61

In [149…
```
plt.figure(figsize=(6,4))
sns.barplot(data = country_tv_show_count, x= 'no_of_tv_shows',y = 'countries',color = 'maroon')
```

Out[149]:
```
<Axes: xlabel='no_of_tv_shows', ylabel='countries'>
```



Insights

- Suggests a pronounced prefernce for movie content over tv shows
- This may be due to audience preferences , production trends and distribution strategies employed by content creators and distributors

5. What is the best time to launch a TV show?

- A. Find which is the best week to release the Tv-show or the movie.
  - Analysis for TV Shows and the best week to release
  - Analysis for Movies and the best week to release
  - which day of the week has more number of movie releases ?

In [150…

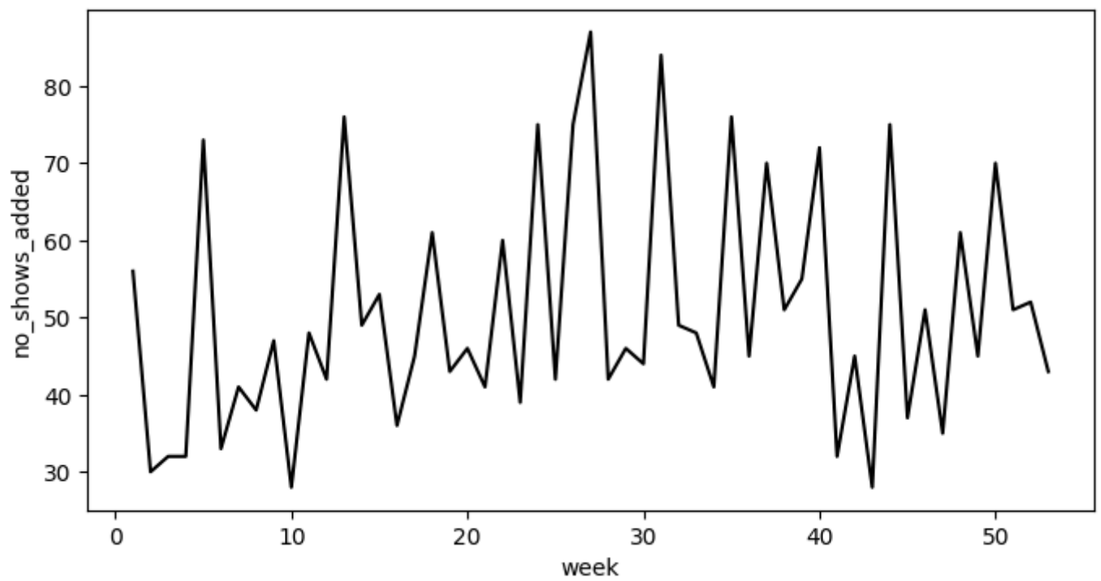
```
# filtering only the TV Show rows
tv_shows = df[df['type']=='TV Show']
#grouping by week and calculating the number of releases (by calculating the count of unique titles)
shows_week = tv_shows.groupby('week')['title'].nunique().reset_index()
# renaming the title ---> no_of_shows_added
shows_week.rename(columns = {'title':'no_shows_added'},inplace =True)
# sorting the dataframe based on the no_of_shows_added in descing order
shows_week = shows_week.sort_values('no_shows_added',ascending=False).reset_index()
# dropping the unecessary index column
shows_week.drop('index',axis = 1,inplace =True)
# showing the weeks and no_of_shows_added from highest to lowest
shows_week.head()
```

Out[150]:

	week	no_shows_added
0	27	87
1	31	84
2	35	76
3	13	76
4	44	75

In [151…
```
plt.figure(figsize=(8,4))
sns.lineplot(data = shows_week , x = 'week',y = 'no_shows_added',color = 'black' )
```

Out[151]:
```
<Axes: xlabel='week', ylabel='no_shows_added'>
```



Insights

- The number of TV shows added per week shows significant variability with fluctuations observed across different weeks.
- Weeks with peak addition of TV shows are 27,31,35 which maps to around June, July,August months
- Indicating Summer months may see an increase in addition of TV Shows
  attracting viewers during vacation periods

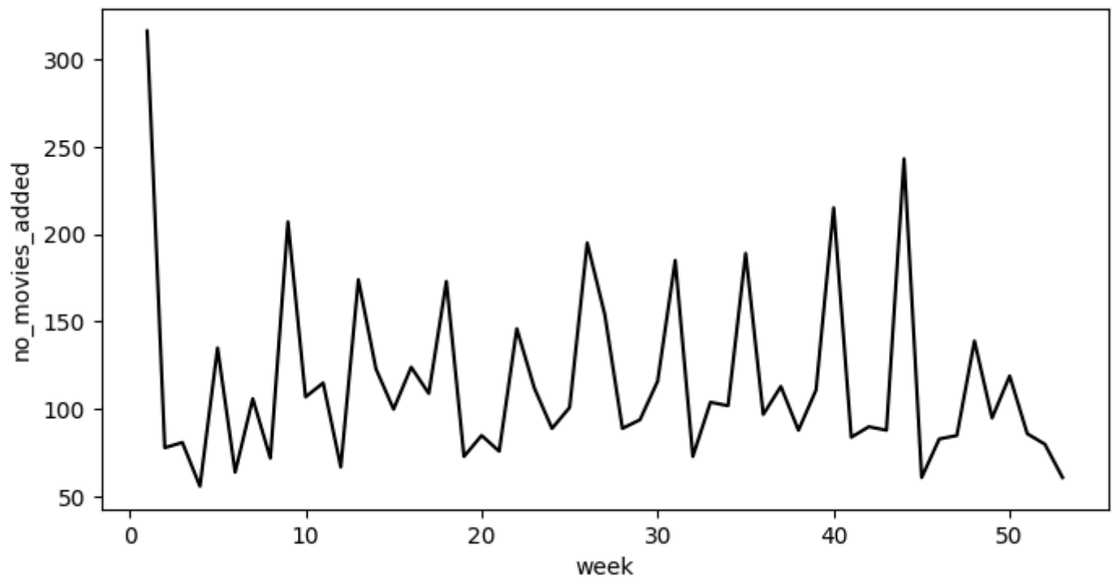
In [152…
```
cinema = df[df['type']=='Movie']
movies_week = cinema.groupby('week')['title'].nunique().reset_index()
movies_week.rename(columns = {'title':'no_movies_added'},inplace =True)
movies_week = movies_week.sort_values('no_movies_added',ascending=False).reset_index()
movies_week.drop('index',axis = 1,inplace =True)
movies_week.head()
```

Out[152]:

	week	no_movies_added
0	1	316
1	44	243
2	40	215
3	9	207
4	26	195

In [153…
```
plt.figure(figsize=(8,4))
sns.lineplot(data = movies_week , x = 'week',y = 'no_movies_added',color = 'black' )
```

Out[153]: <Axes: xlabel='week', ylabel='no_movies_added'>



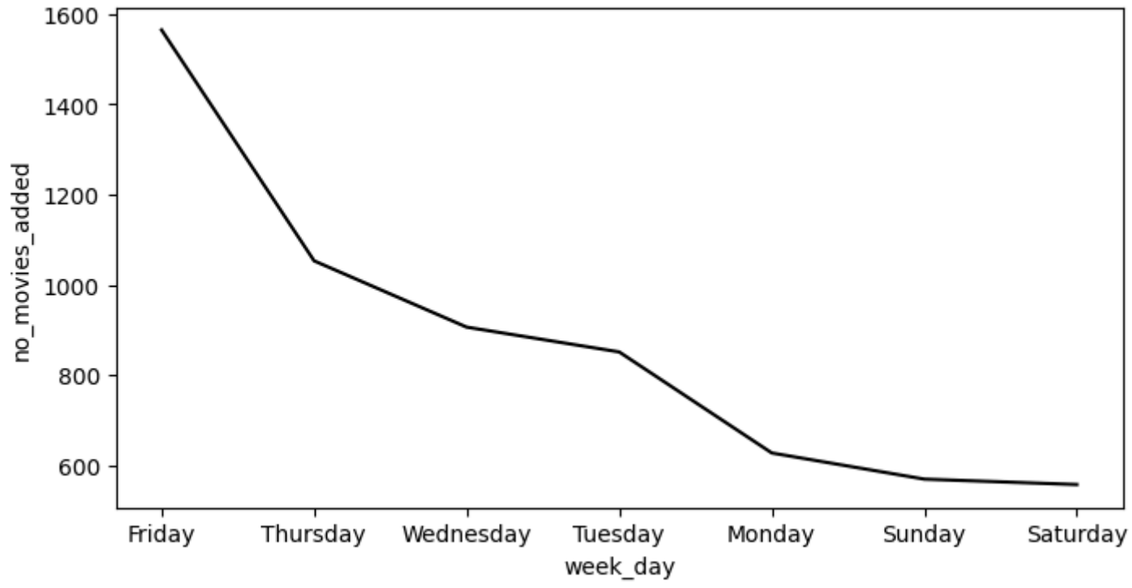
In [154…
```
cinema_day = df[df['type']=='Movie']
movie_day  = cinema_day.groupby('week_day')['title'].nunique().reset_index()
movie_day.rename(columns = {'title':'no_movies_added'},inplace =True)
movie_day = movie_day.sort_values('no_movies_added',ascending=False).reset_index()
movie_day.drop('index',axis = 1,inplace =True)
movie_day.head()
```

Out[154]:

	week_day	no_movies_added
0	Friday	1565
1	Thursday	1053
2	Wednesday	906
3	Tuesday	851
4	Monday	627

In [155…
```
plt.figure(figsize=(8,4))
sns.lineplot(data = movie_day , x = 'week_day',y = 'no_movies_added',color = 'black' )
```

Out[155]: <Axes: xlabel='week_day', ylabel='no_movies_added'>



Insights

- Peak activity on Friday's
- New releases are scheduled towards the weekend
- To coincide with users leisure time to watch movies
- Data also indicates consisitent level of activity throughout the week
- Depicting a good marketing strategy

6. Analysis of actors/directors of different types of shows/movies.

- a. Identify the top 10 actors who have appeared in most movies or TV shows.

In [156…

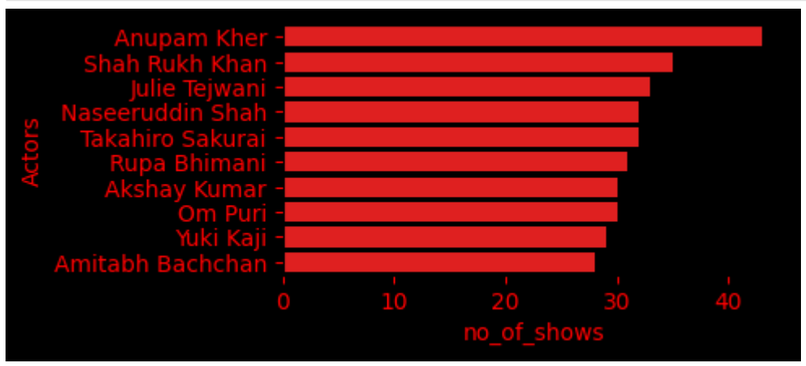
```
actors = df[(df['Actors']!='Unknown Actor')]
actors = actors.groupby('Actors')['title'].nunique().sort_values(ascending = False).head(10).reset_index()
actors.rename(columns = {'title':'no_of_shows'},inplace = True)
actors
```

Out[156]:

	Actors	no_of_shows
0	Anupam Kher	43
1	Shah Rukh Khan	35
2	Julie Tejwani	33
3	Naseeruddin Shah	32
4	Takahiro Sakurai	32
5	Rupa Bhimani	31
6	Akshay Kumar	30
7	Om Puri	30
8	Yuki Kaji	29
9	Amitabh Bachchan	28

In [157…

```
plt.figure(figsize=(4,2))
fig = plt.gcf()

# create bar plot
ax = sns.barplot(data = actors , x = 'no_of_shows',y = 'Actors',color = 'r')

# set the color of thr labels to red
ax.set_xlabel('no_of_shows',color = 'r')
ax.set_ylabel('Actors',color = 'r')

# Set the color of the ticks to red
ax.tick_params(axis = 'x',colors = 'red')
ax.tick_params(axis = 'y',colors = 'red')

# set the background color to black
fig.set_facecolor('black')

# set the background color of the plot area
ax.set_facecolor('black')
```



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

In [158…

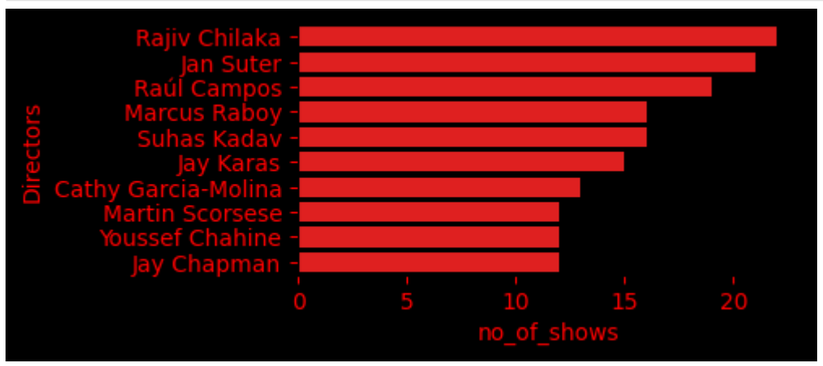
```
directors = df[df['Directors'] != 'Unknown Director']
directors = directors.groupby('Directors')['title'].nunique().sort_values(ascending = False).head(10).reset_index()
directors.rename(columns = {'title':'no_of_shows'},inplace = True)
directors
```

Out[158]:

	Directors	no_of_shows
0	Rajiv Chilaka	22
1	Jan Suter	21
2	Raúl Campos	19
3	Marcus Raboy	16
4	Suhas Kadav	16
5	Jay Karas	15
6	Cathy Garcia-Molina	13
7	Martin Scorsese	12
8	Youssef Chahine	12
9	Jay Chapman	12

In [159…

```
plt.figure(figsize=(4,2))
fig = plt.gcf()

# create bar plot
ax = sns.barplot(data = directors , x = 'no_of_shows',y = 'Directors',color = 'r')

# set the color of thr labels to red
ax.set_xlabel('no_of_shows',color = 'r')
ax.set_ylabel('Directors',color = 'r')

# Set the color of the ticks to red
ax.tick_params(axis = 'x',colors = 'red')
ax.tick_params(axis = 'y',colors = 'red')

# set the background color to black
fig.set_facecolor('black')

# set the background color of the plot area
ax.set_facecolor('black')
```



7. Which genre movies are more popular or produced more

In [160…

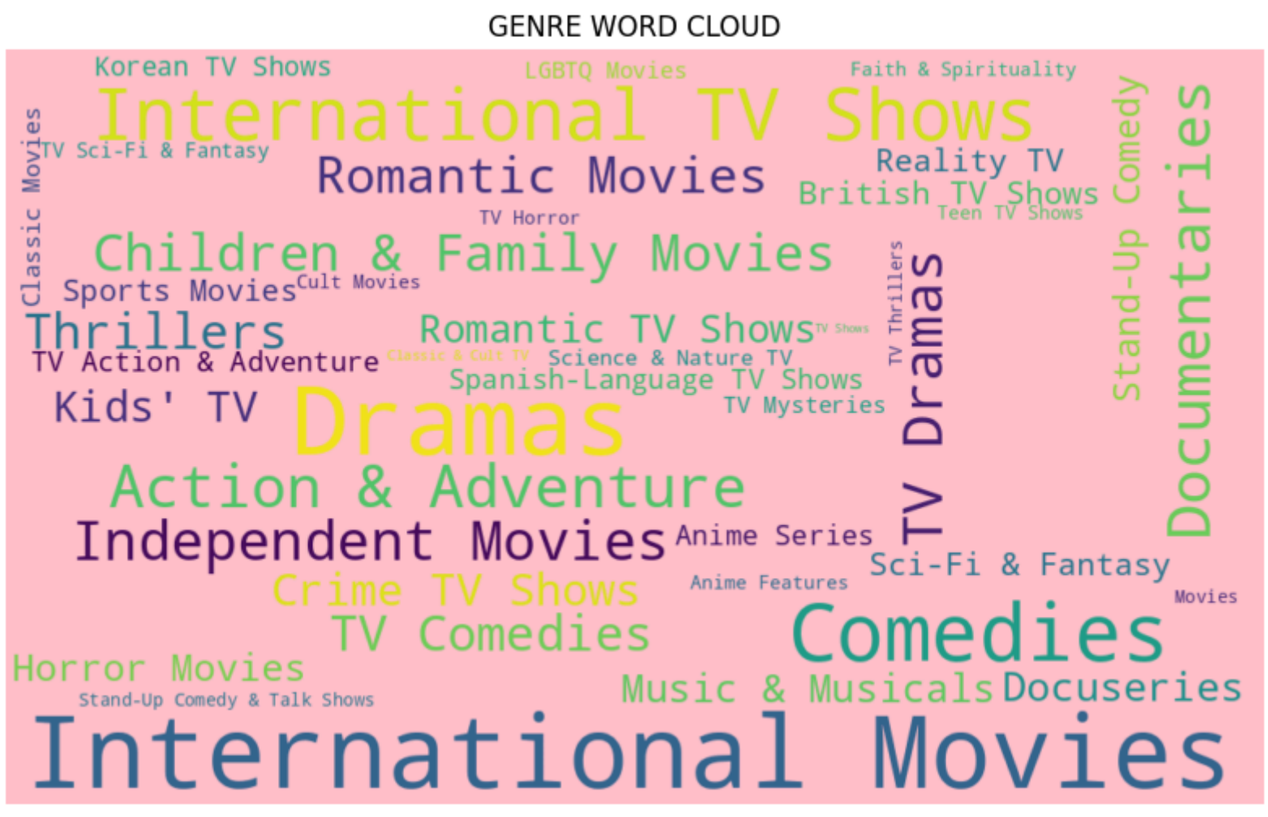
```
genres_section = df.groupby('Genre')['title'].nunique().sort_values(ascending = False).reset_index()
genres_section.head()
```

Out[160]:

	Genre	title
0	International Movies	2752
1	Dramas	2427
2	Comedies	1674
3	International TV Shows	1351
4	Documentaries	869

In [161…

```
from wordcloud import WordCloud
import matplotlib.pyplot as plt

genre_frequency_dict = dict(zip(genres_section['Genre'],genres_section['title']))

wordcloud = WordCloud(width = 1000,height = 600,background_color = 'pink')

wordcloud.generate_from_frequencies(genre_frequency_dict)

plt.figure(figsize=(10,8))
plt.imshow(wordcloud,interpolation='bilinear')
plt.axis('off')
plt.title('GENRE WORD CLOUD')
plt.show()
```



Understanding the Genres in different countries

In [162…

```
# creating a dataframe for country wise genres and finding number of releases under each genre in each country
country_genre = df[df['countries'] != 'Unknown Country'].groupby(['countries','Genre' ])['title'].nunique().sort_values(ascending=False)
country_genre=country_genre.reset_index()
country_genre.rename(columns = {'title':'no_of_releases'},inplace = True)
# identifying the top genre in each country and assigning ranks
country_genre['rank']=country_genre.groupby(['countries'])['no_of_releases'].rank(ascending = False,method = 'first')
country_genre
```

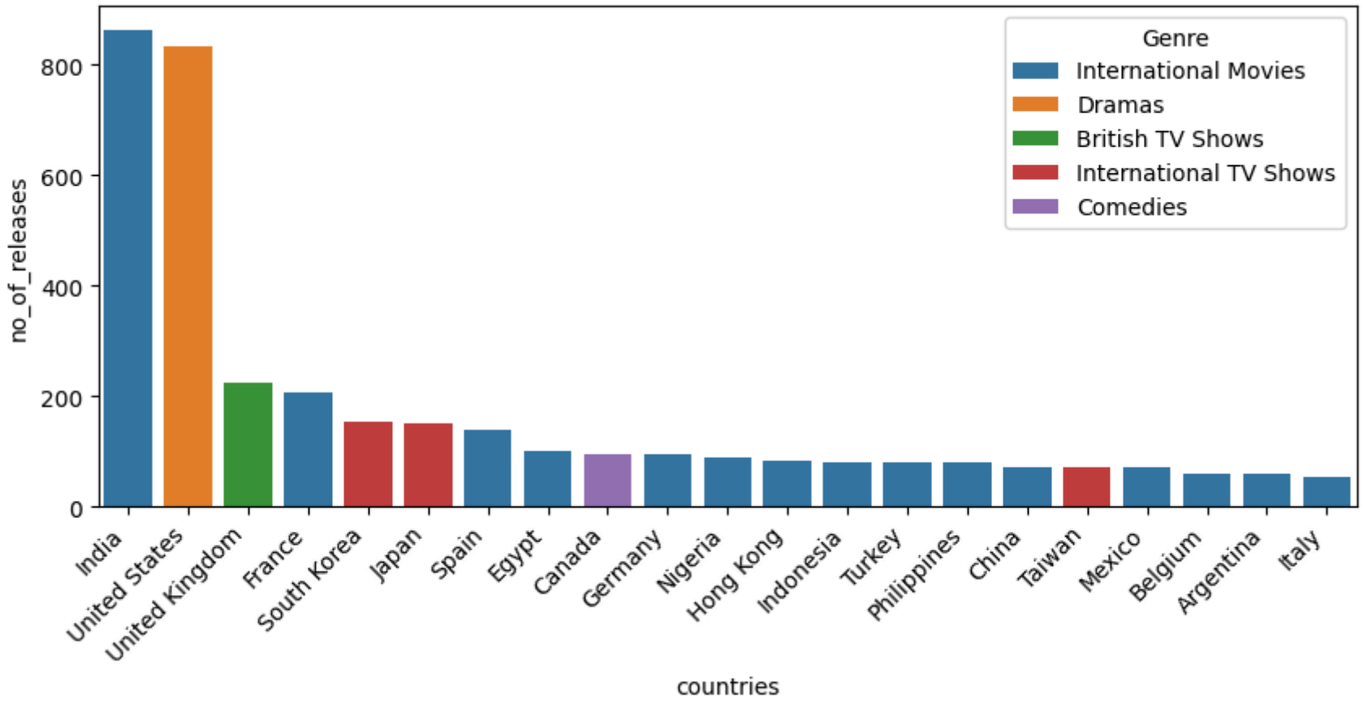
Out[162]:

	countries	Genre	no_of_releases	rank
0	India	International Movies	864	1.0
1	United States	Dramas	835	1.0
2	United States	Comedies	680	2.0
3	India	Dramas	662	2.0
4	United States	Documentaries	511	3.0
...	...	...	...	...
1417	Mauritius	International TV Shows	1	3.0
1418	Mauritius	TV Dramas	1	4.0
1419	Mexico	Classic Movies	1	30.0
1420	Mexico	Faith & Spirituality	1	31.0
1421	Zimbabwe	Romantic Movies	1	4.0

1422 rows × 4 columns

In [163…

```
# filtering only the top genre in each country from the country_genre dataframe
top_genre = country_genre[(country_genre['rank']==1) & (country_genre['no_of_releases']>=50)]
top_genre
plt.figure(figsize=(10,4))
sns.barplot(data = top_genre,x = 'countries',y ='no_of_releases',hue='Genre')
plt.xticks(rotation=45, ha='right')
plt.show()
```



Insights

- **International Movies** Genre prevails worldwide
- It stands out as a popular choice and has got universal appeal
- **Inida** Favours International Movies genre reflecting its popularity and influence among the Indian audience
- Popularity of **Dramas** in the **United States**,showing strong preference for this genre among American **viewers**
- Preferences for British TV Shows in **UK**
- Showing audience inclination towards **locally produced content and the cultural significance of British television**
- Comedy Genre Dominates in **Canada**
- suggesting a preference for **humorous and light-hearted** content among canadian audiences

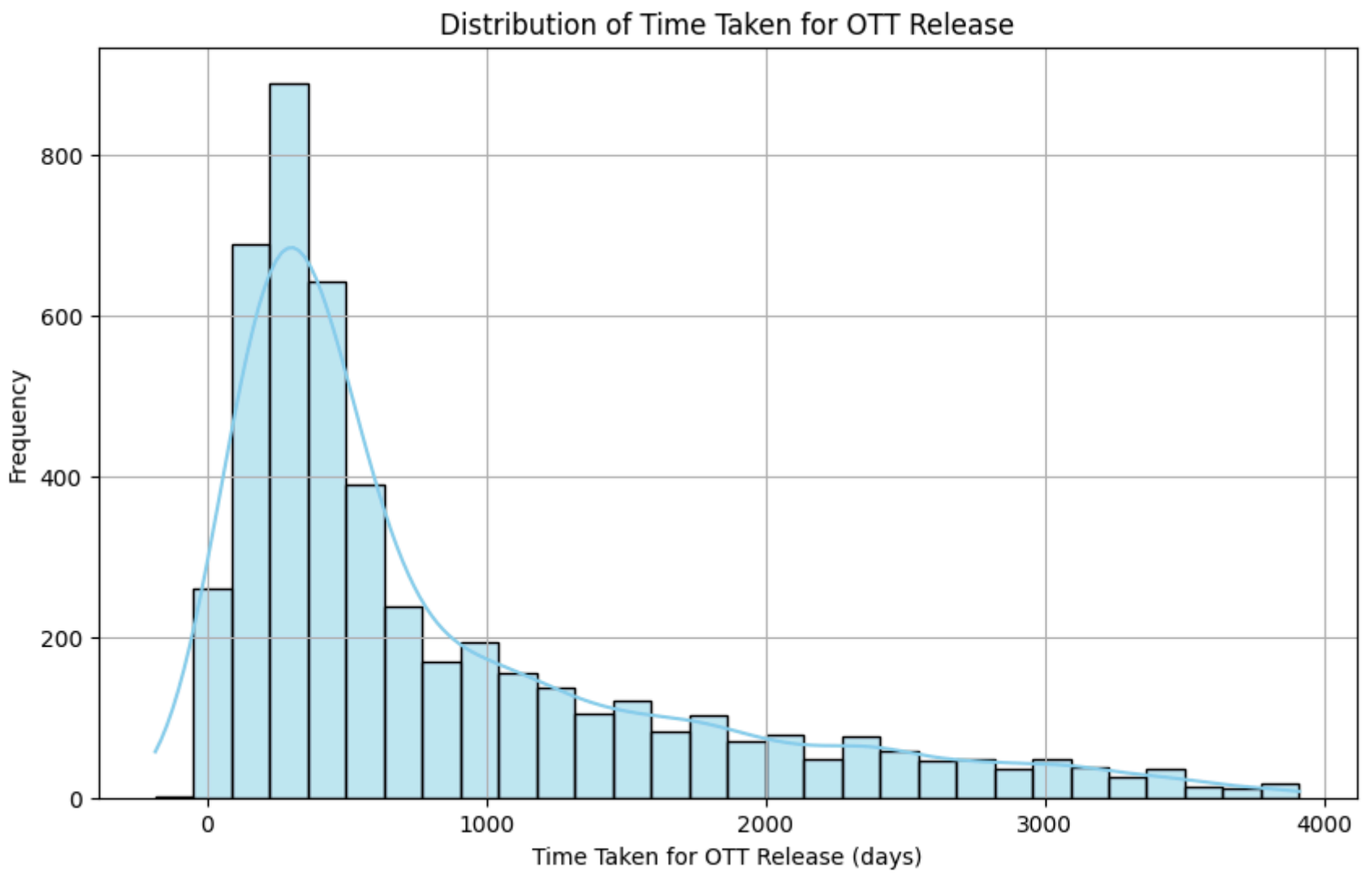
8. Find After how many days the movie will be added to Netflix after the release of the movie (you can consider the recent past data)

In [164…

```
# recent_past I am considering from 2011
recent_releases = df[(df['release_year'].dt.year>=2011) & (df['type']=='Movie')]
# finding the mode of diffrence
new_table = recent_releases[['release_year','title','date_added']]
new_table_copy = new_table.copy()
new_table_copy.drop_duplicates(inplace =True)
new_table_copy['time_taken_for_ott_release'] = (new_table_copy['date_added']-new_table_copy['release_year']).dt.days
days_taken_for_OTT_release = new_table_copy['time_taken_for_ott_release'].mode().values[0]
# on an average its taking 334 days
days_taken_for_OTT_release
# if i have to show it in years
round(days_taken_for_OTT_release/365)
```

Out[164]: 1

In [165…
```
# Plot histogram for time taken for OTT release
plt.figure(figsize=(10, 6))
sns.histplot(new_table_copy['time_taken_for_ott_release'], bins=30, kde=True, color='skyblue', edgecolor='black')
plt.title('Distribution of Time Taken for OTT Release')
plt.xlabel('Time Taken for OTT Release (days)')
plt.ylabel('Frequency')
plt.grid(True)
plt.show()
```



Insights

- Rapid OTT adoption
- The average time taken after the theatrical release is approximately 1 year.
- Indicating a swift adoption of digital streaming platforms by content distributors.

- Impact of COVID - 19:

- The pandemic has likely sped up the release of movies on OTT platforms, enhanicng viewers convenience , allowing audiences to watch movies at there preferred location and time

Little more into understanding top actors in the field of movies and TV industry separately

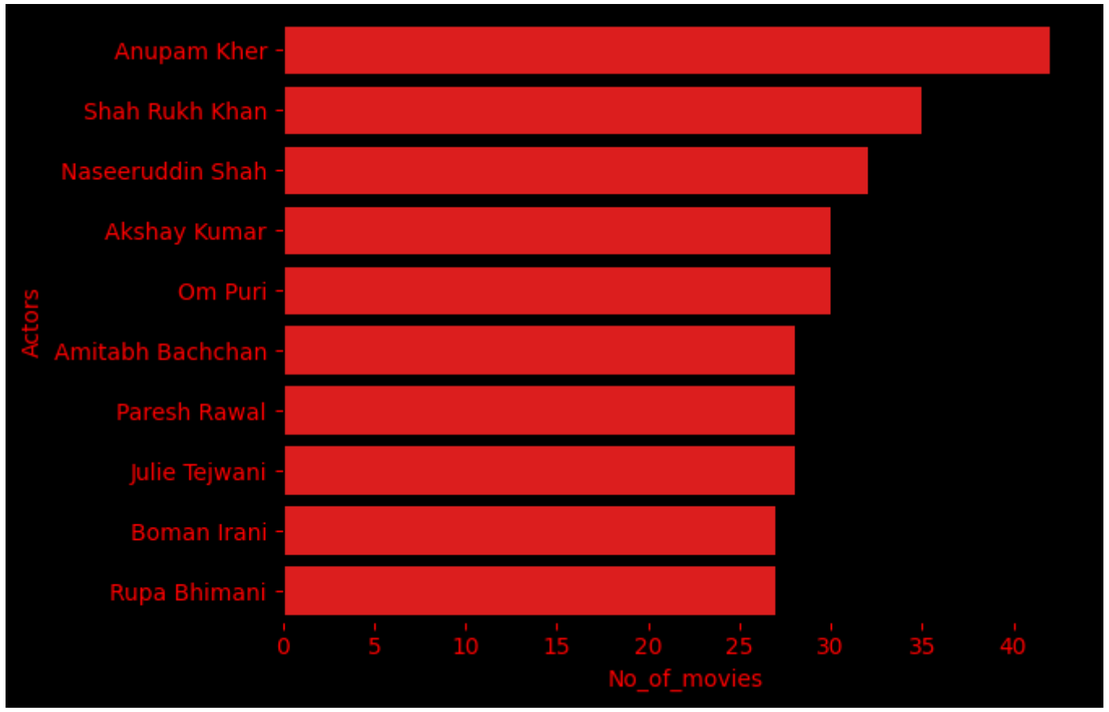
TOP 10 actors from the Movies

In [166…
```
actors = df[(df['Actors']!='Unknown Actor') & (df['type']=='Movie')]
actors = actors.groupby('Actors')['title'].nunique().sort_values(ascending = False).head(10).reset_index()
actors.rename(columns = {'title':'No_of_movies'},inplace = True)
actors
```

Out[166]:

	Actors	No_of_movies
0	Anupam Kher	42
1	Shah Rukh Khan	35
2	Naseeruddin Shah	32
3	Akshay Kumar	30
4	Om Puri	30
5	Amitabh Bachchan	28
6	Paresh Rawal	28
7	Julie Tejwani	28
8	Boman Irani	27
9	Rupa Bhimani	27

In [167…
```
fig = plt.gcf()

# create bar plot
ax = sns.barplot(data = actors , x = 'No_of_movies',y = 'Actors',color = 'r')

# set the color of thr labels to red
ax.set_xlabel('No_of_movies',color = 'r')
ax.set_ylabel('Actors',color = 'r')

# Set the color of the ticks to red
ax.tick_params(axis = 'x',colors = 'red')
ax.tick_params(axis = 'y',colors = 'red')

# set the background color to black
fig.set_facecolor('black')

# set the background color of the plot area
ax.set_facecolor('black')
```



TOP 10 Directors from the Movies

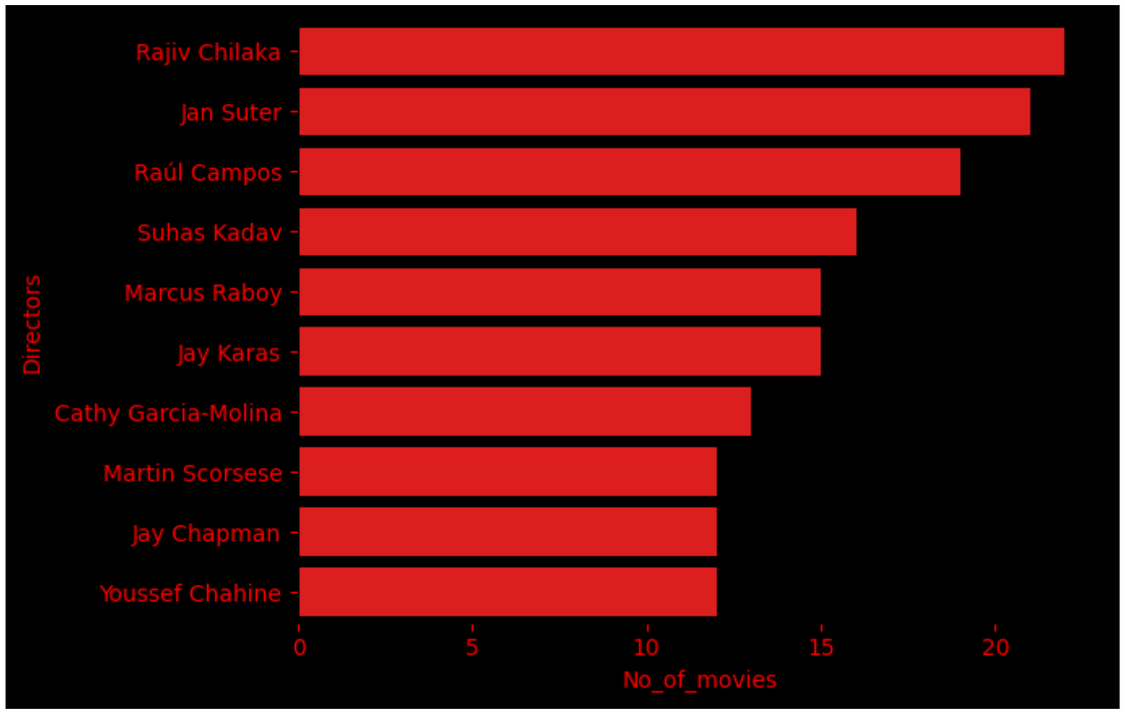
In [168…
```
directors = df[(df['Directors'] != 'Unknown Director') & (df['type']=='Movie')]
directors = directors.groupby('Directors')['title'].nunique().sort_values(ascending = False).head(10).reset_index()
directors.rename(columns = {'title':'No_of_movies'},inplace = True)
directors
```

Out[168]:

	Directors	No_of_movies
0	Rajiv Chilaka	22
1	Jan Suter	21
2	Raúl Campos	19
3	Suhas Kadav	16
4	Marcus Raboy	15
5	Jay Karas	15
6	Cathy Garcia-Molina	13
7	Martin Scorsese	12
8	Jay Chapman	12
9	Youssef Chahine	12

In [169…
```
fig = plt.gcf()

# create bar plot
ax = sns.barplot(data = directors , x = 'No_of_movies',y = 'Directors',color = 'r')

# set the color of thr labels to red
ax.set_xlabel('No_of_movies',color = 'r')
```

```
ax.set_ylabel('Directors',color = 'r')

# Set the color of the ticks to red
ax.tick_params(axis = 'x',colors = 'red')
ax.tick_params(axis = 'y',colors = 'red')

# set the background color to black
fig.set_facecolor('black')

# set the background color of the plot area
ax.set_facecolor('black')
```



TOP 10 Actors from TV shows

In [170…

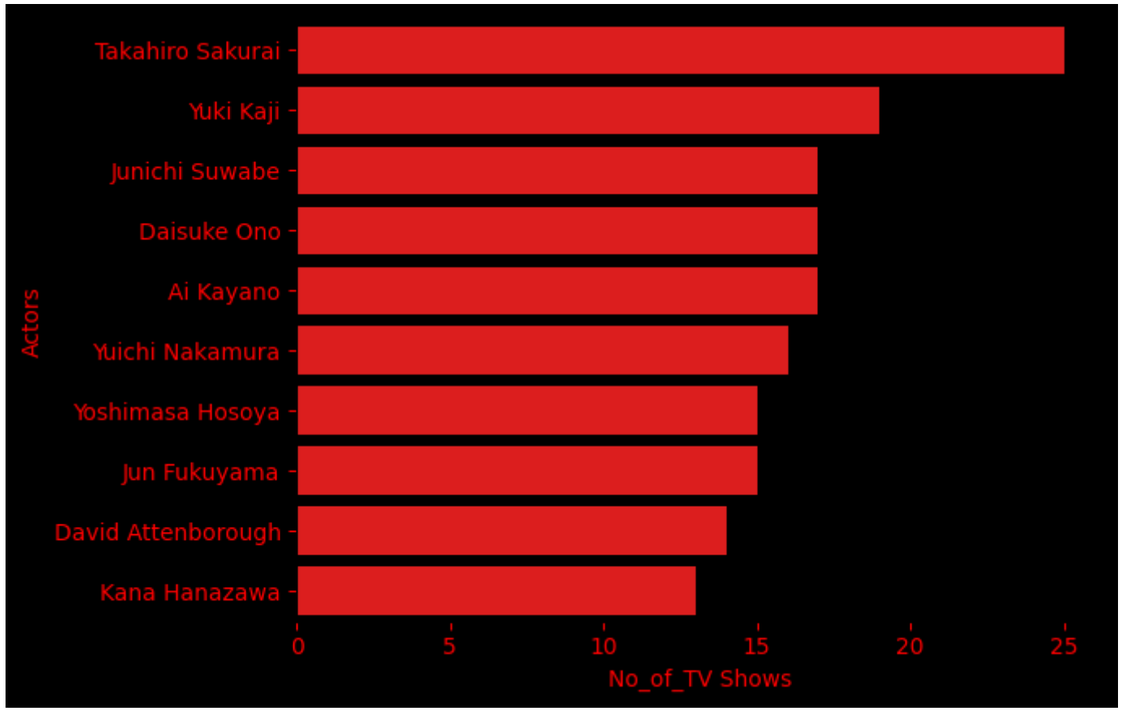
```
actors = df[(df['Actors']!='Unknown Actor') & (df['type']=='TV Show')]
actors = actors.groupby('Actors')['title'].nunique().sort_values(ascending = False).head(10).reset_index()
actors.rename(columns = {'title':'No_of_TV Shows'},inplace = True)
actors
```

Out[170]:

	Actors	No_of_TV Shows
0	Takahiro Sakurai	25
1	Yuki Kaji	19
2	Junichi Suwabe	17
3	Daisuke Ono	17
4	Ai Kayano	17
5	Yuichi Nakamura	16
6	Yoshimasa Hosoya	15
7	Jun Fukuyama	15
8	David Attenborough	14
9	Kana Hanazawa	13

In [171…

```
fig = plt.gcf()

# create bar plot
ax = sns.barplot(data = actors , x = 'No_of_TV Shows',y = 'Actors',color = 'r')

# set the color of thr labels to red
ax.set_xlabel('No_of_TV Shows',color = 'r')
ax.set_ylabel('Actors',color = 'r')

# Set the color of the ticks to red
ax.tick_params(axis = 'x',colors = 'red')
ax.tick_params(axis = 'y',colors = 'red')

# set the background color to black
fig.set_facecolor('black')

# set the background color of the plot area
ax.set_facecolor('black')
```



TOP 10 directors from TV shows

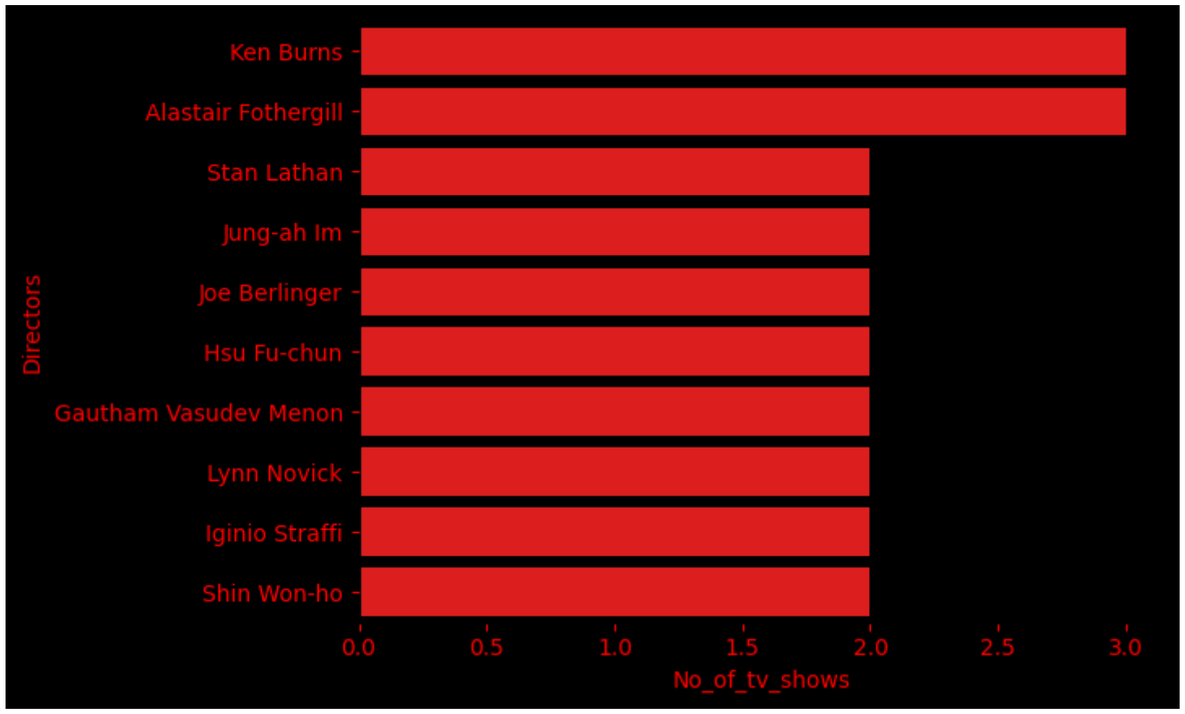
In [172…
```
directors = df[(df['Directors'] != 'Unknown Director') & (df['type']=='TV Show')]
directors = directors.groupby('Directors')['title'].nunique().sort_values(ascending = False).head(10).reset_index()
directors.rename(columns = {'title':'No_of_tv_shows'},inplace = True)
directors
```

Out[172]:

	Directors	No_of_tv_shows
0	Ken Burns	3
1	Alastair Fothergill	3
2	Stan Lathan	2
3	Jung-ah Im	2
4	Joe Berlinger	2
5	Hsu Fu-chun	2
6	Gautham Vasudev Menon	2
7	Lynn Novick	2
8	Iginio Straffi	2
9	Shin Won-ho	2

In [173…
```
fig = plt.gcf()

# create bar plot
ax = sns.barplot(data = directors , x = 'No_of_tv_shows',y = 'Directors',color = 'r')

# set the color of thr labels to red
ax.set_xlabel('No_of_tv_shows',color = 'r')
```

```
ax.set_ylabel('Directors',color = 'r')

# Set the color of the ticks to red
ax.tick_params(axis = 'x',colors = 'red')
ax.tick_params(axis = 'y',colors = 'red')

# set the background color to black
fig.set_facecolor('black')

# set the background color of the plot area
ax.set_facecolor('black')
```



Insights

- Identifying the top 10 actors and directors from movies and shows will shed a light on versatality of actors and there lasting impression on audiences
- Their choice of genre also attracts audiences, that helps content creators to cater similar genres with similar casts

In an attempt to understand directors Genre of interest

In [174…

```
# directors and there genre of interest
director_genre = df[['Directors','Genre','title','countries']][(df['Directors']!='Unknown Director') & (df['countries']!='Unknown Country')]
director_genre = director_genre.groupby(['Directors','Genre','countries'])['title'].nunique().sort_values(ascending = False).reset_index()
director_genre.rename(columns = {'title':'no_of_releases'},inplace=True)
director_genre.head(10)
```

Out[174]:

	Directors	Genre	countries	no_of_releases
0	Jay Karas	Stand-Up Comedy	United States	14
1	Marcus Raboy	Stand-Up Comedy	United States	14
2	Cathy Garcia-Molina	International Movies	Philippines	13
3	Youssef Chahine	Dramas	Egypt	12
4	Jay Chapman	Stand-Up Comedy	United States	12
5	Jan Suter	Stand-Up Comedy	Mexico	12
6	Raúl Campos	Stand-Up Comedy	Mexico	10
7	Youssef Chahine	International Movies	Egypt	10
8	Shannon Hartman	Stand-Up Comedy	United States	9
9	Cathy Garcia-Molina	Dramas	Philippines	9

In [175…

```
# indian directors and there genre of interest
top5_IND_directors_genres=director_genre[director_genre['countries']=='India'].head(10).reset_index()
top5_IND_directors_genres.drop(columns = 'index',axis = 1,inplace =True)
top5_IND_directors_genres
```

Out[175]:

	Directors	Genre	countries	no_of_releases
0	David Dhawan	Comedies	India	9
1	Anurag Kashyap	International Movies	India	8
2	David Dhawan	International Movies	India	8
3	Umesh Mehra	International Movies	India	8
4	Dibakar Banerjee	International Movies	India	7
5	Dibakar Banerjee	Dramas	India	6
6	Sooraj R. Barjatya	International Movies	India	6
7	Sooraj R. Barjatya	Dramas	India	6
8	Ashutosh Gowariker	International Movies	India	6
9	Priyadarshan	International Movies	India	6

Lets explore few Genre's precisely and see which country top's in it

- 'Children & Family Movies'
- 'Horror Movies'
- 'Anime Features'

In [176…

```
# top 5 directors in the field of children and family movies genre
top_5_directors = director_genre[director_genre['Genre']=='Children & Family Movies'].head().reset_index()
top_5_directors.drop(columns = 'index',axis = 1,inplace =True)
top_5_directors
```

Out[176]:

	Directors	Genre	countries	no_of_releases
0	Robert Rodriguez	Children & Family Movies	United States	7
1	Steven Spielberg	Children & Family Movies	United States	6
2	William Lau	Children & Family Movies	United States	6
3	Rajiv Chilaka	Children & Family Movies	India	5
4	Ishi Rudell	Children & Family Movies	United States	5

In [177…

```
# top 5 directors in the field of horror genre
top_5_directors = director_genre[director_genre['Genre']=='Horror Movies'].head().reset_index()
top_5_directors.drop(columns = 'index',axis = 1,inplace =True)
top_5_directors
```

Out[177]:

	Directors	Genre	countries	no_of_releases
0	Rocky Soraya	Horror Movies	Indonesia	6
1	Poj Arnon	Horror Movies	Thailand	4
2	Kevin Smith	Horror Movies	United States	3
3	Banjong Pisanthanakun	Horror Movies	Thailand	3
4	David R. Ellis	Horror Movies	United States	3

In [178…

```
# top 5 directors in the Anime genre
top_5_directors = director_genre[director_genre['Genre']=='Anime Features'].head().reset_index()
top_5_directors.drop(columns = 'index',axis = 1,inplace =True)
top_5_directors
```

Out[178]:

	Directors	Genre	countries	no_of_releases
0	Toshiya Shinohara	Anime Features	Japan	7
1	Masahiko Murata	Anime Features	Japan	5
2	Noriyuki Abe	Anime Features	Japan	3
3	Kazuchika Kise	Anime Features	Japan	3
4	Hiroyuki Seshita	Anime Features	Japan	3

Insights

- Incorporating elements from multiple genres can attract a diverse range of viewers.
- Utilizing insights from genre analysis helps in creating content that captures people's attention effectively.

Recommendations :

- Offer a **balanced mix of both TV Shows and Movies** to cater to diverse audience preferences
- Capitalize on the **potential for growth in TV Show** releases especially in key markets like **US and UK**
- Schedule **releases in weekends for movies** and **summer months for TV Shows** to maximize viewership

- Customize **content delivery based on genre choices** like for example in countries like **Canada comedy** is the people's choice so try releasing more such content more often , similarly in **US it is Dramas**.
- **Optimize OTT release timing** to reach audiences effectively, consider releasing movies on OTT platform sooner, to capitalize on viewer demand and maximize revenue potential.
- Invest in **local content** especially in countries like India and UK producing original and culturally relevant content that resonates well with local audiences can strengthen market presence.
- Most of the content falls under **90-120** mins duration so cater content by considering audiences attention span
- Majority of our audiences are **older teenagers and adults** so cater more content to this age group
- **Crossing genres** may lead to increased overall audience engagement and satisfaction.**For example in US stand up comedy** is something thats well accepted, and **comedy is one such genre with a natural appeal that transcends borders and demographics**.

In [178…