Insights of Data

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
import plotly.graph_objs as go
```

```
!wget https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/000/940/original/netflix.csv -O Netflix.csv
```

```
--2023-08-07 04:03:38--  https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/000/940/original/netflix.csv
Resolving d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)... 108.157.172.173, 108.157.172.10, 108.157.172.183, ...
Connecting to d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)|108.157.172.173|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3399671 (3.2M) [text/plain]
Saving to: ‘Netflix.csv’

Netflix.csv         100%[===================>]   3.24M  --.-KB/s    in 0.1s

2023-08-07 04:03:38 (22.0 MB/s) - ‘Netflix.csv’ saved [3399671/3399671]
```

```
df= pd.read_csv('Netflix.csv')
```

```
df.head()
```

	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...

```
df.shape
```

```
(8807, 12)
```

```
df.columns
```

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

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   show_id       8807 non-null   object
 1   type          8807 non-null   object
 2   title         8807 non-null   object
```

```
 3   director      6173 non-null   object
 4   cast          7982 non-null   object
 5   country       7976 non-null   object
 6   date_added    8797 non-null   object
 7   release_year  8807 non-null   int64
 8   rating        8803 non-null   object
 9   duration      8804 non-null   object
 10  listed_in     8807 non-null   object
 11  description   8807 non-null   object
dtypes: int64(1), object(11)
memory usage: 825.8+ KB
```

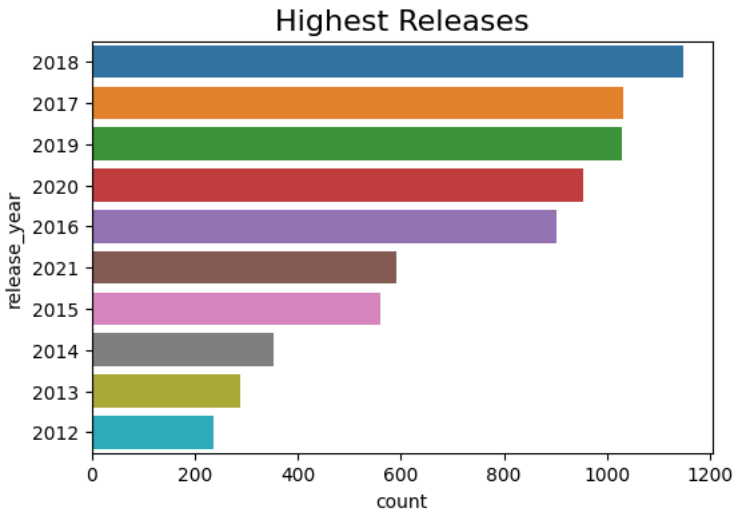
```
df['release_year'].max()
```

```
2021
```

```
df['release_year'].min()
```

```
1925
```

```
plt.figure(figsize=(6,4))
plt.title("Highest Releases", fontsize= 16)
sns.countplot(y='release_year', data=df, order=df.release_year.value_counts().index[:10])
plt.show()
```




```
df.describe()
```

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

Missing Values

```
#null values in dataset
```

```
df.isnull().sum().sort_values(ascending=False)
```

```
director        2634
country          831
cast             825
date_added        10
```

```
rating            4
duration          3
show_id           0
type              0
title             0
release_year      0
listed_in         0
description       0
dtype: int64
```

```
#percentage of null values in dataset

round(df.isnull().sum()/df.shape[0]*100,2).sort_values(ascending=False)
```

```
director        29.91
country          9.44
cast             9.37
date_added       0.11
rating           0.05
duration         0.03
show_id          0.00
type             0.00
title            0.00
release_year     0.00
listed_in        0.00
description      0.00
dtype: float64
```

```
#dropping rows-- duration

df.dropna(subset=['duration'],axis=0, inplace=True)
df.dropna(subset=['date_added'], axis=0, inplace=True)
```

```
##replace missing values of director,country,cast,  date_added and rating

df['director'].replace(np.NaN, 'No Director', inplace=True)
df['country'].replace(np.NaN, 'Unknown', inplace=True)
df['cast'].replace(np.NaN, 'No Cast', inplace=True)
df['rating'].replace(np.NaN, 'No Rating', inplace=True)
```

```
#number of missing values after droping and replacing
df.isnull().sum().sort_values(ascending=False)
```

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

```
#Removing duplicates

df.drop_duplicates(inplace= True)
```

```
df.shape
```

```
(8794, 12)
```

Adding Year, Month and day columns for further calculation

```
df.date_added.info()
```

```
<class 'pandas.core.series.Series'>
Int64Index: 8794 entries, 0 to 8806
Series name: date_added
Non-Null Count  Dtype
--------------  -----
8794 non-null   object
```

```
dtypes: object(1)
memory usage: 137.4+ KB
```

```
#adding new columns as year
df['year']=df['date_added'].apply(lambda x: x.split(', ')[-1]).astype('int', True)
```

```
#adding new column as month

from datetime import datetime
def month(mname):
  mnum = datetime.strptime(mname, '%B').month
  return mnum

df['month']=((df['date_added'].apply(lambda x: x.lstrip().split(' ')[0])).apply(month)).astype('int', True)
```

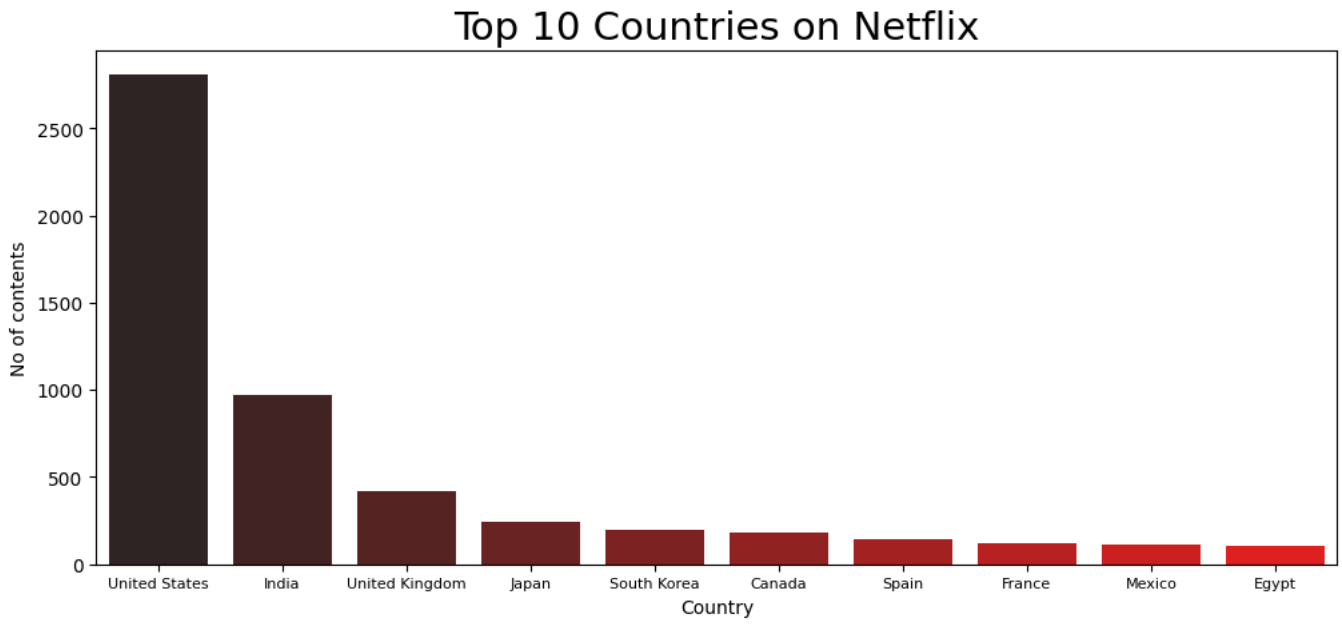
```
#adding new column as date
df['day']=((df['date_added'].apply(lambda x: x.lstrip().split(' ')[1])).apply(lambda x: x.replace(',' , '') if ',' in x else x)).astype('int'
```

```
df.head(2)
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	year	mor
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	No Cast	United States	September 25, 2021	2020	PG-13	90 min	Documentaries	As her father nears the end of his life, filmm...	2021	
1	s2	TV Show	Blood & Water	No Director	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...	2021	

Content in different countries

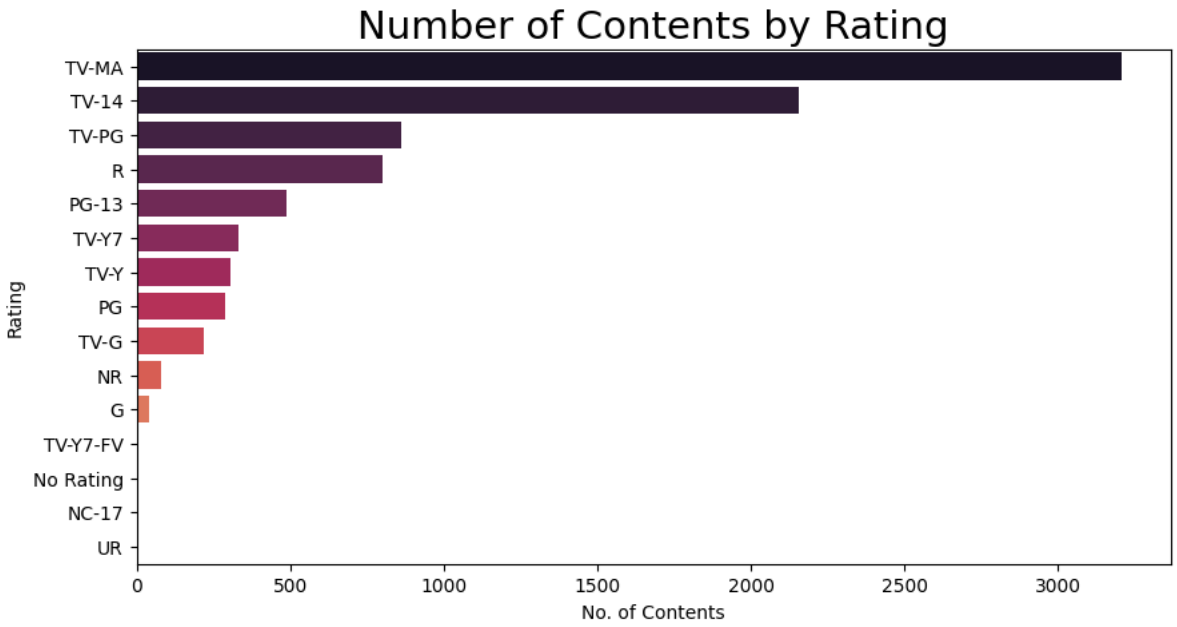
```
#Top 10 content producing countries

Countries = df[(df['country'] != 'Unknown')]['country']
plt.figure(figsize=(12,5))
sns.countplot(x = Countries, order=Countries.value_counts().index[:10], palette='dark:red')
plt.title('Top 10 Countries on Netflix', fontsize=21)
plt.xlabel('Country')
plt.xticks(fontsize=8)
plt.ylabel('No of contents')
plt.show()
```



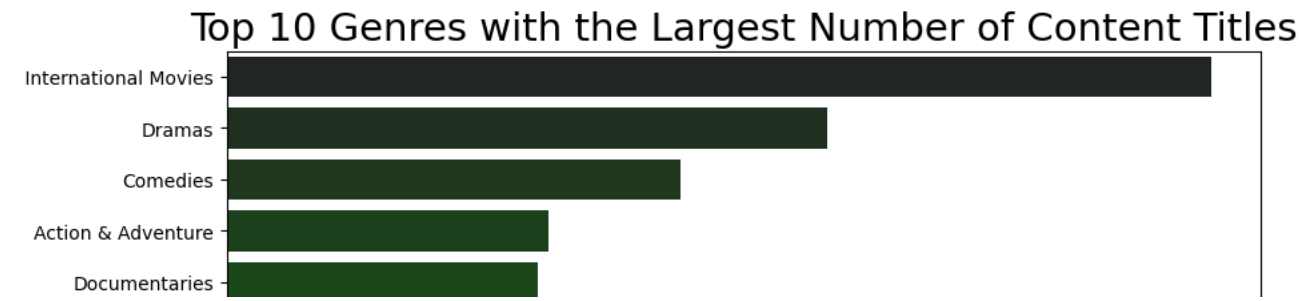
```
df.rating.value_counts()
```

```
TV-MA        3205
TV-14        2157
TV-PG         861
R             799
PG-13         490
TV-Y7         333
TV-Y          306
PG            287
TV-G          220
NR             79
G              41
TV-Y7-FV        6
No Rating       4
NC-17           3
UR              3
Name: rating, dtype: int64
```

```
#No of contents by rating

plt.figure(figsize=(10,5))
sns.countplot(y='rating', data=df, order=df.rating.value_counts().index, palette='rocket')
plt.title('Number of Contents by Rating', fontsize=21)
plt.xlabel('No. of Contents')
plt.ylabel('Rating')
plt.show()
```




```
#Top 10 genres

top_genres = df.set_index('title').listed_in.str.split(',',expand=True).stack().reset_index(level=1, drop=True)
plt.figure(figsize=(10, 5))
sns.countplot(y=top_genres, order=top_genres.value_counts().index.to_list()[:10], palette='dark:green')
plt.title('Top 10 Genres with the Largest Number of Content Titles', fontsize=21);
plt.xlabel('No. of Contents')
plt.show()
```

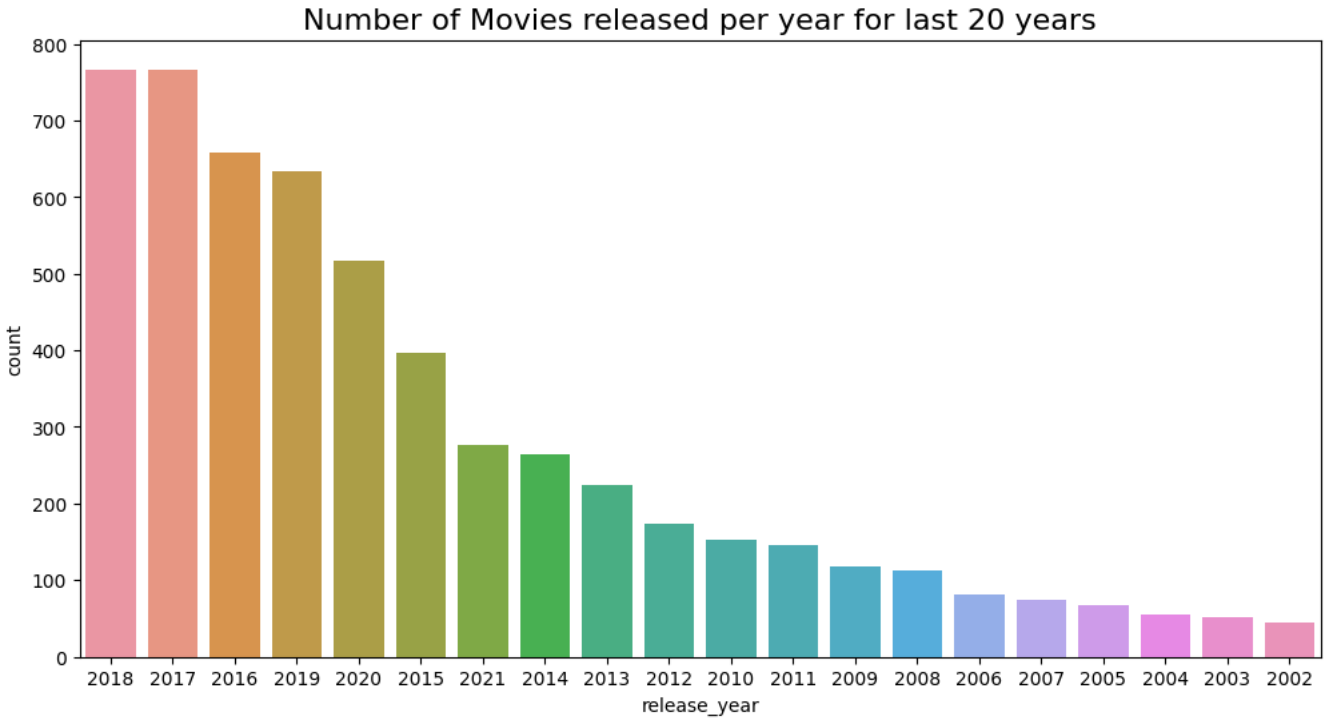


The number of movies released per year changed over the last 20-30 years

```
top20= df[df['type']=='Movie'].release_year.value_counts().sort_values(ascending=False).index[:20]
top20
```

```
Int64Index([2018, 2017, 2016, 2019, 2020, 2015, 2021, 2014, 2013, 2012, 2010,
            2011, 2009, 2008, 2006, 2007, 2005, 2004, 2003, 2002],
           dtype='int64')
```

```
plt.figure(figsize=(12,6))
plt.title("Number of Movies released per year for last 20 years", fontsize= 16)
sns.countplot(x='release_year', data=df[df['type']=='Movie'], order=top20)
#plt.xticks(rotation=90)
plt.show()
```

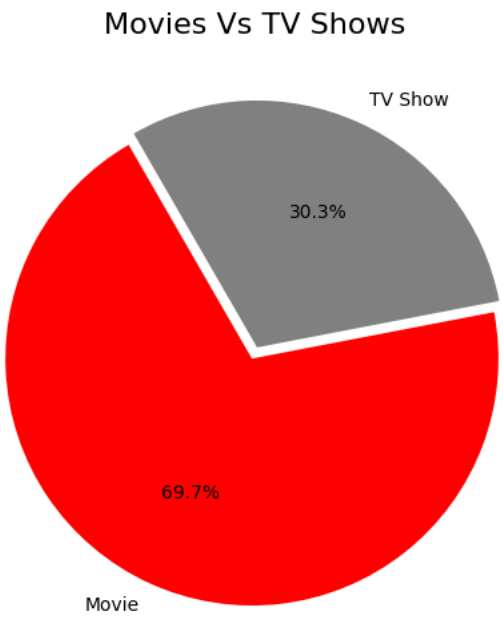


Movies Vs TV Shows

```
#No. of Movies and TV showes
df.type.value_counts()
```

```
Movie      6128
TV Show    2666
Name: type, dtype: int64
```

```
# Movies Vs TV Shows

plt.figure(figsize=(8,6))
plt.title('Movies Vs TV Shows',fontsize=16)
plt.pie(df.type.value_counts(),explode=(0.025,0.025), labels=df.type.value_counts().index, colors=['red','grey'],  autopct='%1.1f%%', startan

plt.show()
```




```
#converting movies duration column--> string to int

movies_df= df.loc[(df['type']=='Movie')]

movies_df['duration']= movies_df.duration.apply(lambda x: x.replace('min', '') if 'min' in x else x)

movies_df.loc[:,['duration']]= movies_df.loc[:,['duration']].apply(lambda x: x.astype('int64', errors='ignore'))

movies_df.describe()
```

```
<ipython-input-33-b672f3f6e156>:5: 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-co
  movies_df['duration']= movies_df.duration.apply(lambda x: x.replace('min', '') if 'min' in x else x)
<ipython-input-33-b672f3f6e156>:7: 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-co
  movies_df.loc[:,['duration']]= movies_df.loc[:,['duration']].apply(lambda x: x.astype('int64', errors='ignore'))
<ipython-input-33-b672f3f6e156>:7: DeprecationWarning: In a future version, `df.iloc[:, i] = newvals` will attempt to set the values inp
  movies_df.loc[:,['duration']]= movies_df.loc[:,['duration']].apply(lambda x: x.astype('int64', errors='ignore'))
```

	release_year	duration	year	month	day
count	6128.000000	6128.000000	6128.000000	6128.000000	6128.000000
mean	2013.121084	99.577187	2018.850522	6.607866	12.083714
std	9.680300	28.290593	1.561276	3.453028	9.935257
min	1942.000000	3.000000	2008.000000	1.000000	1.000000
25%	2012.000000	87.000000	2018.000000	4.000000	1.000000
50%	2016.000000	98.000000	2019.000000	7.000000	12.000000
75%	2018.000000	114.000000	2020.000000	10.000000	20.000000
max	2021.000000	312.000000	2021.000000	12.000000	31.000000

```
#shortest movie

shortest_movie= movies_df.loc[(movies_df['duration']==np.min(movies_df.duration))]
shortest_movie
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	year	month	da
3777	s3778	Movie	Silent	Limbert Fabian, Brandon	No Cast	United States	June 4, 2019	2014	TV-Y	3	Children & Family Movies,	"Silent" is an animated short film	2019	6	

```
#Longest movie

longest_movie= movies_df.loc[(movies_df['duration']==np.max(movies_df.duration))]
longest_movie
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	ye
4253	s4254	Movie	Black Mirror: Bandersnatch	No Director	Fionn Whitehead, Will Poulter, Craig Parkinson...	United States	December 28, 2018	2018	TV-MA	312	Dramas, International Movies, Sci-Fi & Fantasy	In 1984, a young programmer begins to question...	20

```
#converting TV shows duration column--> string to int

tvshows_df= df.loc[(df['type']=='TV Show')]


tvshows_df['duration']= tvshows_df.duration.apply(lambda x: x.replace('Season', '') if 'Season' in x else x)
tvshows_df['duration']= tvshows_df.duration.apply(lambda x: x.replace('s', '') if 's' in x else x)

tvshows_df.loc[:,['duration']]= tvshows_df.loc[:,['duration']].apply(lambda x: x.astype('int64', errors='ignore'))

tvshows_df.describe()
```

```
<ipython-input-36-4a6d41f9c72f>:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
  tvshows_df['duration']= tvshows_df.duration.apply(lambda x: x.replace('Season', '') if 'Season' in x else x)
<ipython-input-36-4a6d41f9c72f>:7: 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-co
  tvshows_df['duration']= tvshows_df.duration.apply(lambda x: x.replace('s', '') if 's' in x else x)
<ipython-input-36-4a6d41f9c72f>:9: 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-co
  tvshows_df.loc[:,['duration']]= tvshows_df.loc[:,['duration']].apply(lambda x: x.astype('int64', errors='ignore'))
<ipython-input-36-4a6d41f9c72f>:9: DeprecationWarning: In a future version, `df.iloc[:, i] = newvals` will attempt to set the values inp
  tvshows_df.loc[:,['duration']]= tvshows_df.loc[:,['duration']].apply(lambda x: x.astype('int64', errors='ignore'))
```

	release_year	duration	year	month	day
count	2666.000000	2666.000000	2666.000000	2666.000000	2666.000000
mean	2016.625656	1.751313	2018.923856	6.762941	13.448987
std	5.733559	1.550176	1.601306	3.397725	9.716112
min	1925.000000	1.000000	2008.000000	1.000000	1.000000
25%	2016.000000	1.000000	2018.000000	4.000000	4.000000
50%	2018.000000	1.000000	2019.000000	7.000000	14.000000
75%	2020.000000	2.000000	2020.000000	10.000000	21.000000
max	2021.000000	17.000000	2021.000000	12.000000	31.000000

```
#shortest TV Shows

shortest_tvshows= tvshows_df.loc[(tvshows_df['duration']==np.min(tvshows_df.duration))]
shortest_tvshows.info()
```

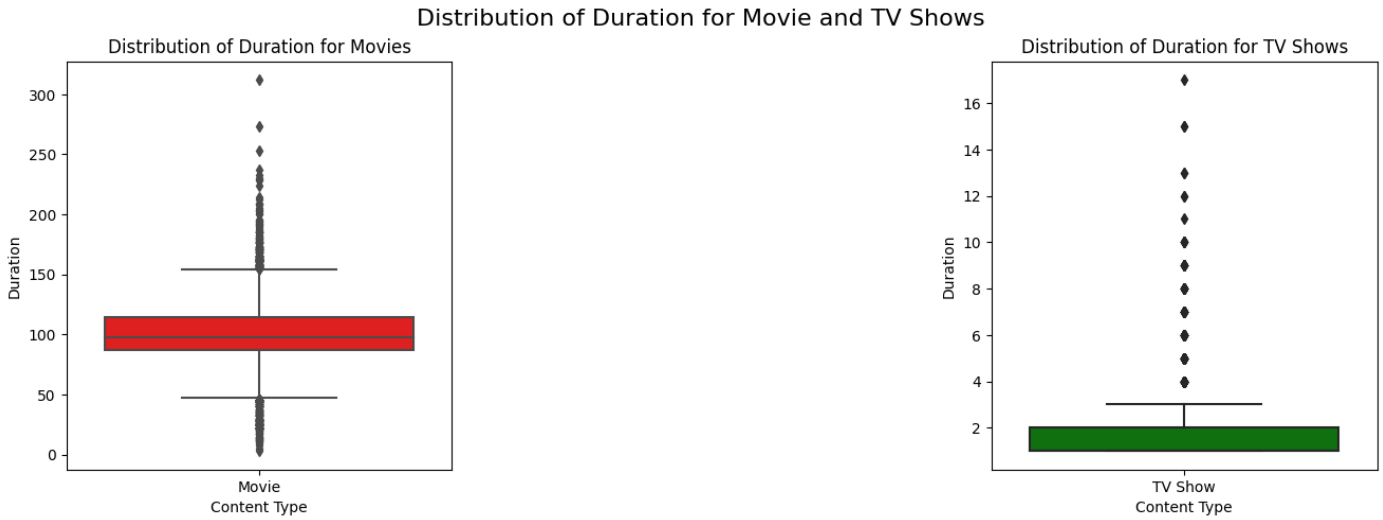
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1793 entries, 2 to 8800
Data columns (total 15 columns):
```

```
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   show_id       1793 non-null   object
 1   type          1793 non-null   object
 2   title         1793 non-null   object
 3   director      1793 non-null   object
 4   cast          1793 non-null   object
 5   country       1793 non-null   object
 6   date_added    1793 non-null   object
 7   release_year  1793 non-null   int64
 8   rating        1793 non-null   object
 9   duration      1793 non-null   int64
 10  listed_in     1793 non-null   object
 11  description   1793 non-null   object
 12  year          1793 non-null   int64
 13  month         1793 non-null   int64
 14  day           1793 non-null   int64
dtypes: int64(5), object(10)
memory usage: 224.1+ KB
```

```
#Longest tvshows

longest_tvshows= tvshows_df.loc[(tvshows_df['duration']==np.max(tvshows_df.duration))]
longest_tvshows
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	year	month
548	s549	TV Show	Grey's Anatomy	No Director	Ellen Pompeo, Sandra Oh, Katherine Heigl, Just...	United States	July 3, 2021	2020	TV-14	17	Romantic TV Shows, TV Dramas	Intern (and eventual resident) Meredith Grey f...	2021	7

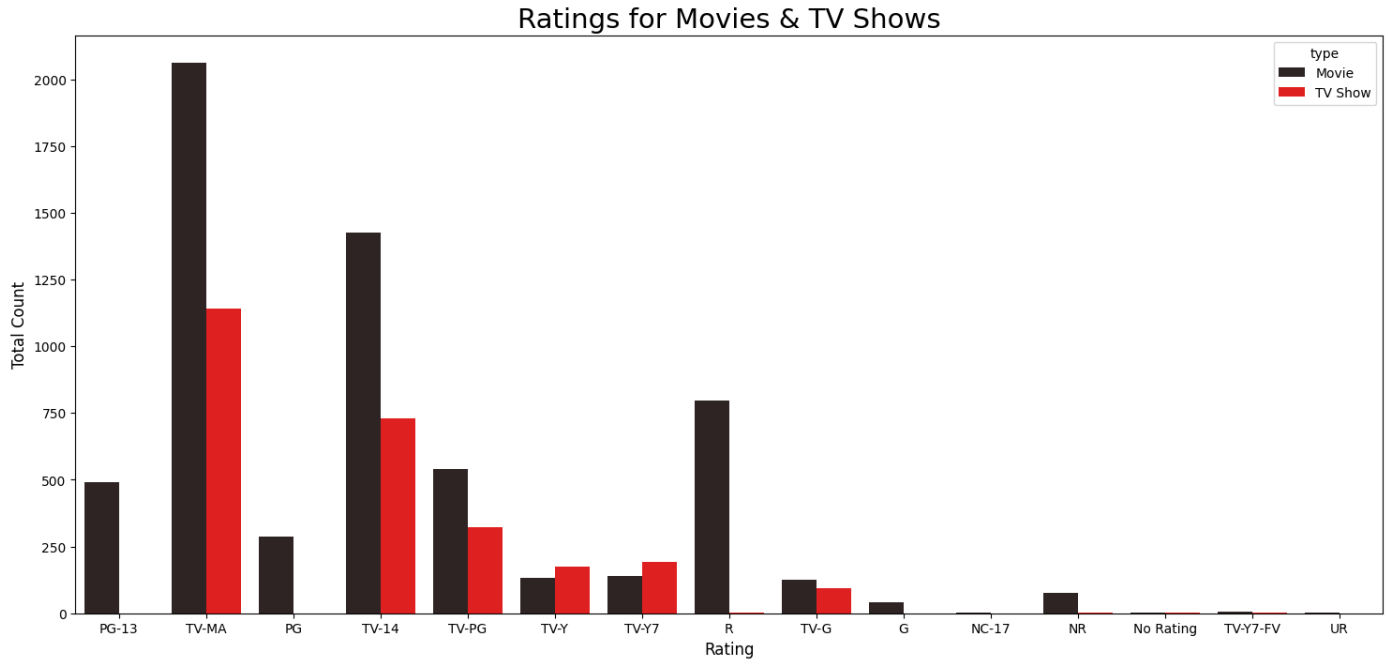
```
#Duration Distribution for Movies and TV Shows

plt.figure(figsize=(16,5)).suptitle('Distribution of Duration for Movie and TV Shows', fontsize= 16)

plt.subplot(1,3,1)
sns.boxplot(data=movies_df, x='type', y='duration', color='red')
plt.xlabel('Content Type')
plt.ylabel('Duration')
plt.title('Distribution of Duration for Movies')

plt.subplot(1,3,3)
sns.boxplot(data=tvshows_df, x='type', y='duration', color='green')
plt.xlabel('Content Type')
plt.ylabel('Duration')
plt.title('Distribution of Duration for TV Shows')
plt.show()
```



```
df.rating.unique()
```

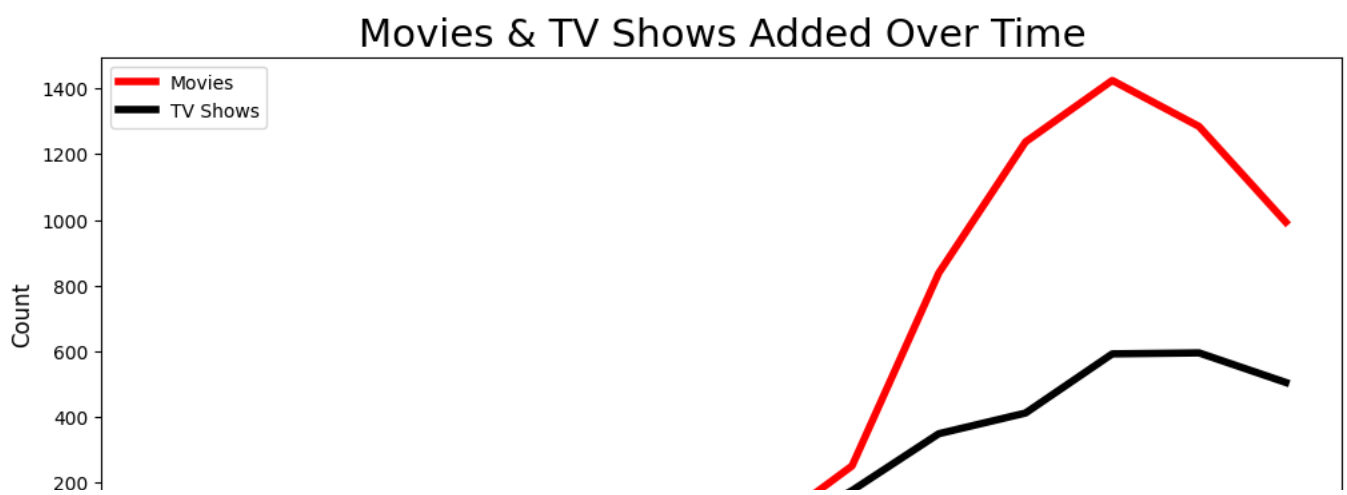
```
array(['PG-13', 'TV-MA', 'PG', 'TV-14', 'TV-PG', 'TV-Y', 'TV-Y7', 'R',
       'TV-G', 'G', 'NC-17', 'NR', 'No Rating', 'TV-Y7-FV', 'UR'],
      dtype=object)
```

```
#Rating comparison for TV Shows and Movie

order = df.rating.unique()
plt.figure(figsize=(18,8))
sns.countplot(x=df.rating, order=order, hue=df.type,  palette="dark:red");
plt.title("Ratings for Movies & TV Shows", fontsize=21)
plt.xlabel("Rating", fontsize=12)
plt.ylabel("Total Count", fontsize=12)
plt.show()
```

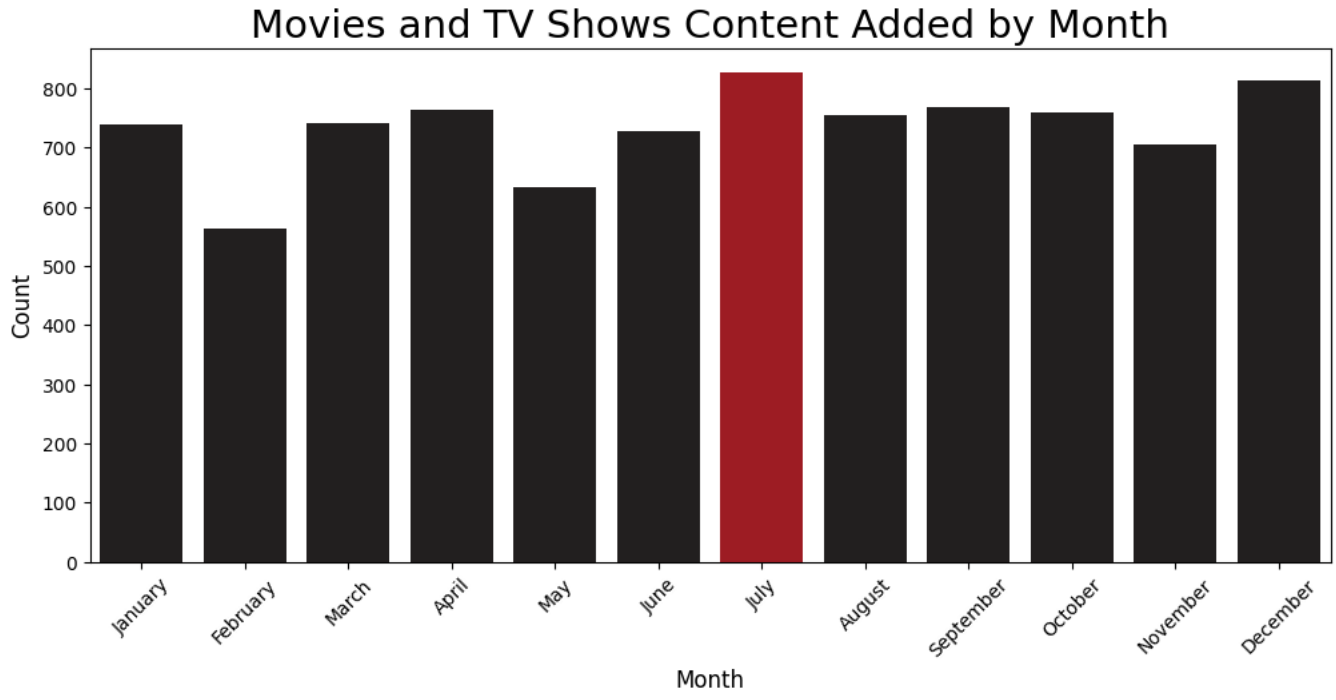


The best time to launch a TV show

```
#Movies & TV Shows Added Over Time

movies_df = df[df['type'] == 'Movie']
tvshows_df = df[df['type'] == 'TV Show']

movies_count = movies_df['year'].value_counts().sort_index()
tvshows_count = tvshows_df['year'].value_counts().sort_index()

plt.figure(figsize=(12,5))
plt.plot(movies_count.index, movies_count.values, color='red', label='Movies', linewidth=4)
plt.plot(tvshows_count.index, tvshows_count.values, color='black', label='TV Shows', linewidth=4)

plt.xlabel('Year', fontsize=12)
plt.ylabel('Count', fontsize=12)
plt.title('Movies & TV Shows Added Over Time', fontsize=21)
plt.legend()

plt.show()
```

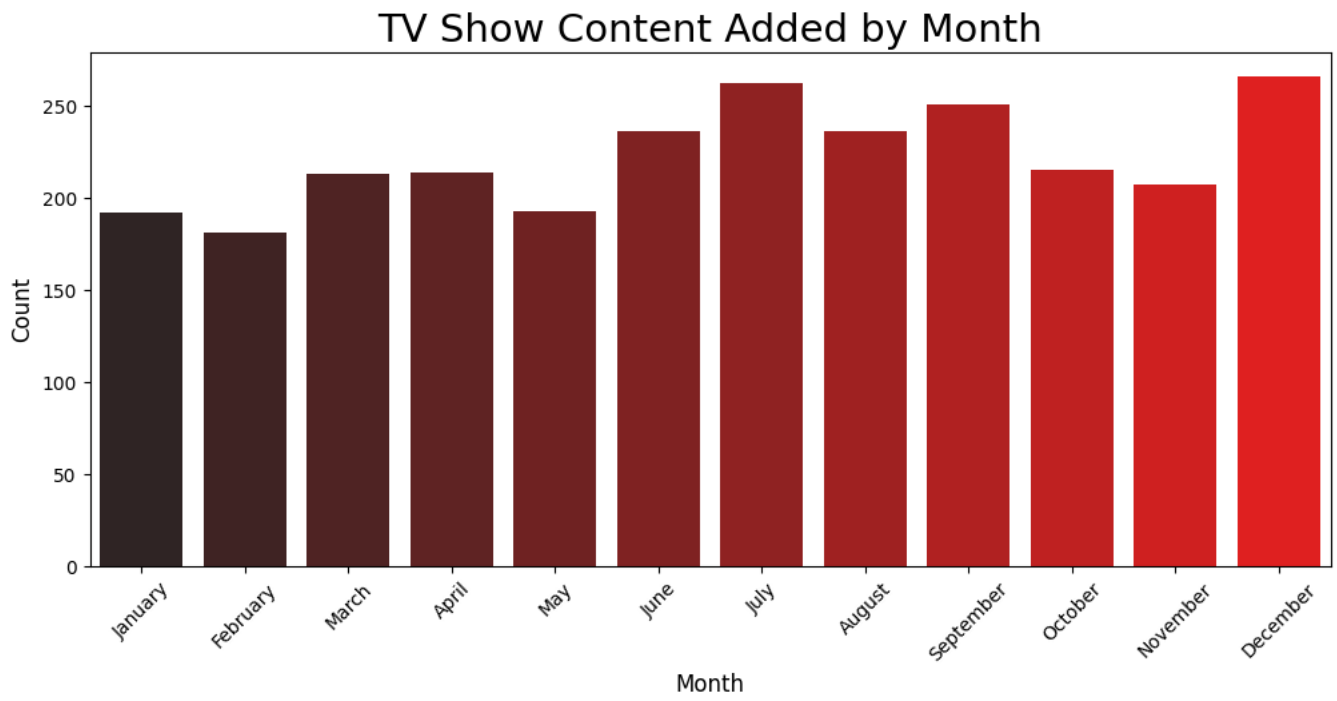



```
# Content added by month(for movies and tv shows)

month_added = pd.to_datetime(df['date_added']).dt.month_name()

month_order = ['January', 'February', 'March', 'April', 'May', 'June', 'July',
               'August', 'September', 'October', 'November', 'December']
monthly_counts = month_added.value_counts().loc[month_order]

max_count = monthly_counts.max()
colors = ['#b20710' if count == max_count else '#221f1f' for count in monthly_counts]

plt.figure(figsize=(12, 5))
sns.barplot(x=monthly_counts.index, y=monthly_counts.values, palette=colors)

plt.xlabel('Month', fontsize=12)
plt.ylabel('Count', fontsize=12)
plt.title('Movies and TV Shows Content Added by Month', fontsize=21)

plt.xticks(rotation=45)
plt.show()
```




```
# TV Shows Content added by month

month_added = pd.to_datetime(df[(df['type']=='TV Show')]['date_added']).dt.month_name()

month_order = ['January', 'February', 'March', 'April', 'May', 'June', 'July',
               'August', 'September', 'October', 'November', 'December']
```

```
monthly_counts = month_added.value_counts().loc[month_order]

#max_count = monthly_counts.max()
#colors = ['#b20710' if count == max_count else '#221f1f' for count in monthly_counts]

plt.figure(figsize=(12, 5))
sns.barplot(x=monthly_counts.index, y=monthly_counts.values, palette='dark:red')

plt.xlabel('Month', fontsize=12)
plt.ylabel('Count', fontsize=12)
plt.title('TV Show Content Added by Month', fontsize=21)

plt.xticks(rotation=45)
plt.show()
```



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

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

```
No Director                      2624
Rajiv Chilaka                      19
Raúl Campos, Jan Suter             18
Marcus Raboy                       16
Suhas Kadav                        16
                                 ...
Raymie Muzquiz, Stu Livingston      1
Joe Menendez                        1
Eric Bross                          1
Will Eisenberg                      1
Mozez Singh                         1
Name: director, Length: 4528, dtype: int64
```

```
#total no. of directors

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

```
4528
```

```
#Top 10 movie Directors
df[(df['type']=='Movie') & (df['director'] != 'No Director')]['director'].value_counts().sort_values(ascending=False)[:10]
```

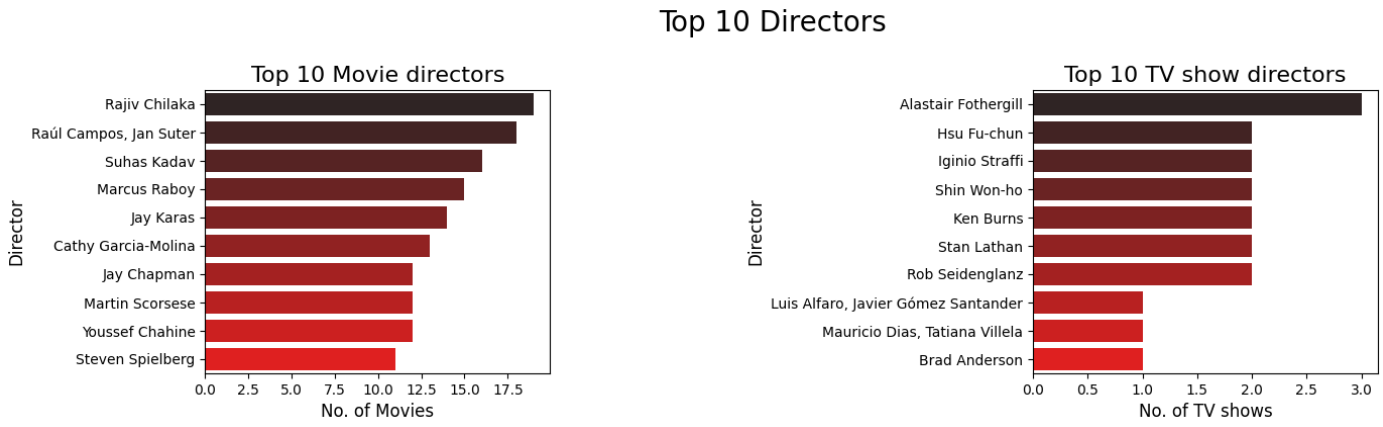
```
Rajiv Chilaka             19
Raúl Campos, Jan Suter    18
Suhas Kadav               16
Marcus Raboy              15
Jay Karas                 14
Cathy Garcia-Molina       13
Jay Chapman               12
```

```
Martin Scorsese          12
Youssef Chahine          12
Steven Spielberg         11
Name: director, dtype: int64
```

```
#Top 10 TV show Directors
df[(df['type']=='TV Show') & (df['director'] != 'No Director')]['director'].value_counts().sort_values(ascending=False)[:10]
```

```
Alastair Fothergill                   3
Hsu Fu-chun                           2
Iginio Straffi                        2
Shin Won-ho                           2
Ken Burns                             2
Stan Lathan                           2
Rob Seidenglanz                       2
Luis Alfaro, Javier Gómez Santander   1
Mauricio Dias, Tatiana Villela        1
Brad Anderson                         1
Name: director, dtype: int64
```

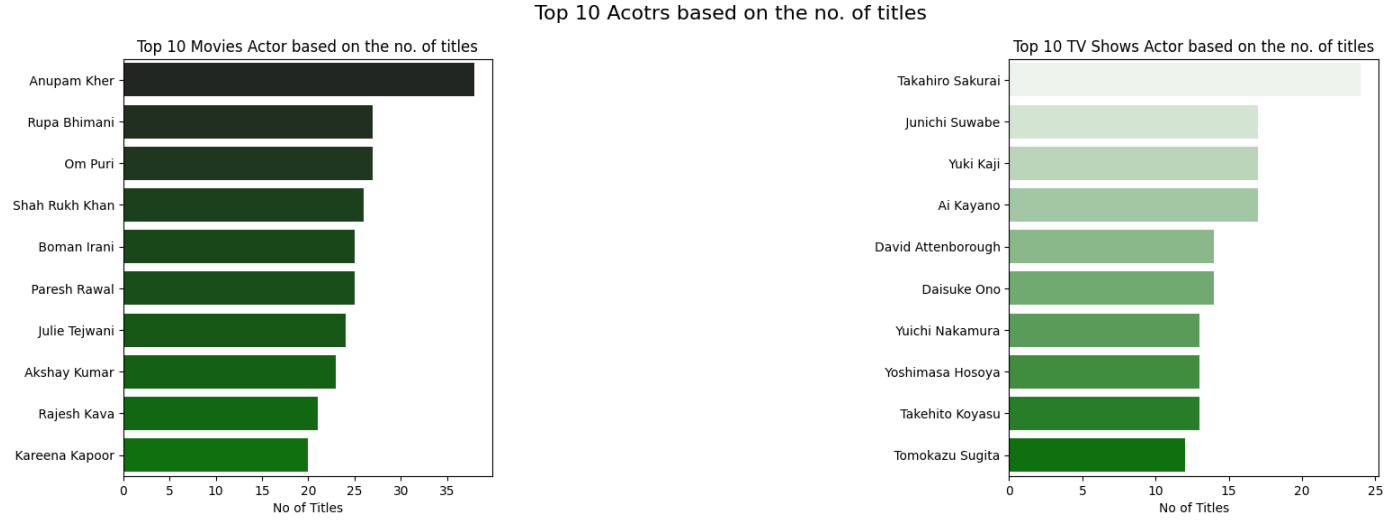
```
#Top 10 directors

plt.figure(figsize=(15,8)).suptitle("Top 10 Directors",fontsize=20)

plt.subplot(2,3,1)
plt.title('Top 10 Movie directors',fontsize=16)
sns.countplot(y= 'director', data=df[(df['type']=='Movie')], order= df[(df['type']=='Movie') & (df['director'] != 'No Director')]['director']
              palette='dark:red')
plt.xlabel('No. of Movies', fontsize=12)
plt.ylabel('Director',fontsize=12)

plt.subplot(2,3,3)
plt.title('Top 10 TV show directors', fontsize=16)
sns.countplot(y= 'director', data=df[df['type']=='TV Show'], order= df[(df['type']=='TV Show') & (df['director'] != 'No Director')]['director
              palette= 'dark:red')
plt.xlabel('No. of TV shows', fontsize=12)
plt.ylabel('Director',fontsize=12)

plt.show()
```



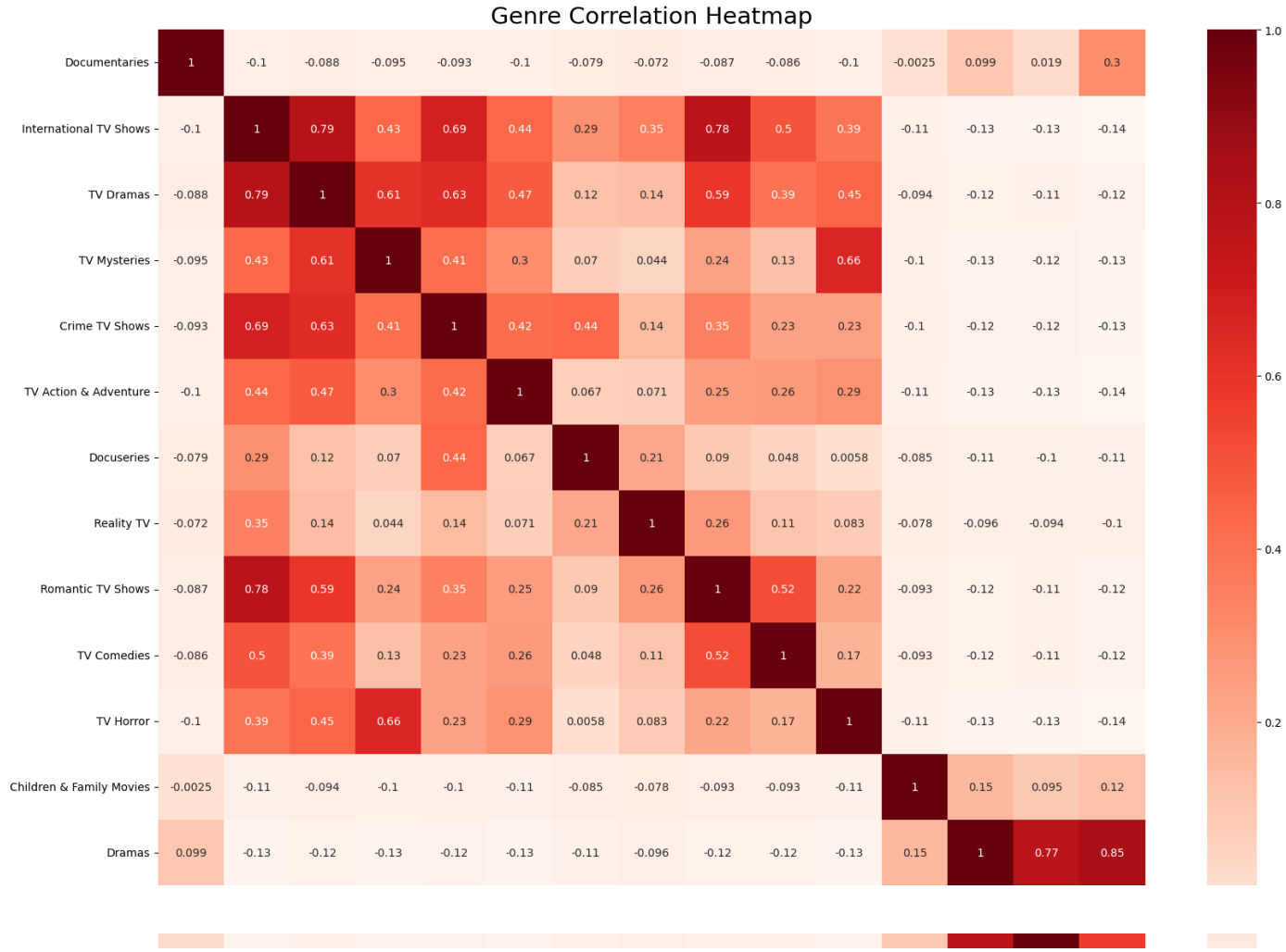

```
##Top 10 actors

cast_movies=df[(df.cast != 'No Cast') & (df.type=='Movie')].set_index('title').cast.str.split(',',expand=True).stack().reset_index(level=1, d
cast_tvshows=df[(df.cast != 'No Cast') & (df.type=='TV Show')].set_index('title').cast.str.split(',',expand=True).stack().reset_index(level=1

plt.figure(figsize=(18,6)).suptitle('Top 10 Acotrs based on the no. of titles', fontsize=16)

plt.subplot(1,3,1)
plt.title('Top 10 Movies Actor based on the no. of titles')
sns.countplot(y=cast_movies, order=cast_movies.value_counts().index[:10], palette='dark:green')
plt.xlabel('No of Titles')

plt.subplot(1,3,3)
plt.title('Top 10 TV Shows Actor based on the no. of titles')
sns.countplot(y=cast_tvshows, order=cast_tvshows.value_counts().index[:10], palette='light:green')
```

```
plt.xlabel('No of Titles')
plt.show
```

```
<function matplotlib.pyplot.show(close=None, block=None)>
```

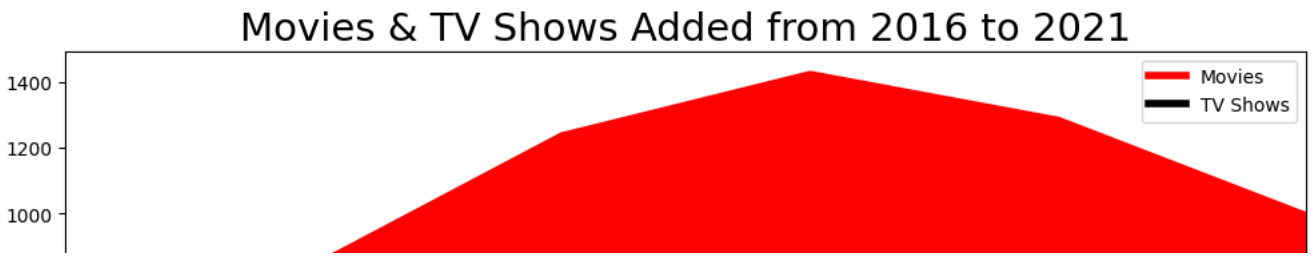



```
# Genre correlation heatmap

genres = df['listed_in'].str.split(', ', expand=True).stack().unique()

genre_data = pd.DataFrame(index=genres, columns=genres, dtype=float)

genre_data.fillna(0, inplace=True)

for _, row in df.iterrows():
    listed_in = row['listed_in'].split(', ')
    for genre1 in listed_in:
        for genre2 in listed_in:
            genre_data.at[genre1, genre2] += 1

correlation_matrix = genre_data.corr()

correlation_matrix= correlation_matrix.iloc[:15,:15]


plt.figure(figsize=(20, 16))
sns.heatmap(correlation_matrix, annot=True, cmap='Reds', square= True)

plt.title('Genre Correlation Heatmap', fontsize=21)
plt.xticks(rotation=90)
plt.yticks(rotation=0)

plt.show()
```



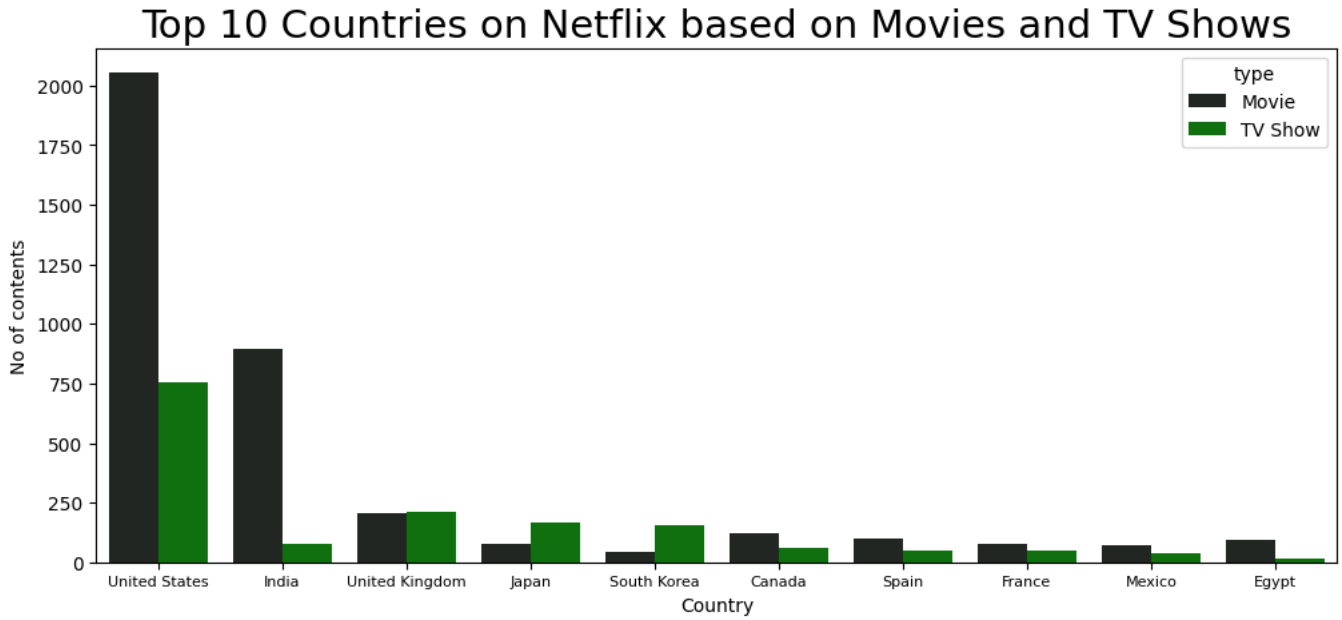
Does Netflix has more focus on TV Shows than movies in recent years

```
#Movies & TV Shows Added from 2016-2021

plt.figure(figsize=(12,5))
plt.plot(movies_count.index, movies_count.values, color='red', label='Movies', linewidth=4)
plt.plot(tvshows_count.index, tvshows_count.values, color='black', label='TV Shows', linewidth=4)

plt.fill_between(movies_count.index, movies_count.values, color='red')
plt.fill_between(tvshows_count.index, tvshows_count.values, color='black')

plt.xlabel('Year', fontsize=12)
plt.ylabel('Count', fontsize=12)
plt.title('Movies & TV Shows Added from 2016 to 2021', fontsize=21)
plt.xlim(2016,2021)
plt.legend()

plt.show()
```



Content is available in different countries

```
#Top 10 countries based on Moivies & TV shows

Countries = df.loc[(df['country'] != 'Unknown')]
plt.figure(figsize=(12,5))
sns.countplot(x = Countries.country, order=Countries.country.value_counts().index[:10], hue=Countries.type, palette='dark:green')
plt.title('Top 10 Countries on Netflix based on Movies and TV Shows', fontsize=21)
plt.xlabel('Country')
plt.xticks(fontsize=8)
plt.ylabel('No of contents')
plt.show()
```



0s completed at 11:34 AM