> NETFLIX

NOTE: link to google collab: some of the graphs are not clearly visible plese view them in the collab note book; https://colab.research.google.com/drive/1cdSg8J723G3ltrHVIrpDdweuoQ4Gg6rl?usp=sharing

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

Layout to solve the business case study: To solve every business problem the following steps will be followed:

The given open ended business problem would be mentioned. Followed by the code which would help us derive the result will be mentioned. For some problems after the result part insights will be derived and at the end suitable recommendations would also be mentioned.

```
import pandas as pd
import numpy as np
!wget
        "https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/000/940/original/netflix.csv"
         -O netflix.csv
```

```
--2023-06-27 06:53:34--
https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/000/940,

Resolving d2beiqkhq929f0.cloudfront.net
(d2beiqkhq929f0.cloudfront.net)... 18.172.139.61, 18.172.139.210,
18.172.139.46, ...
Connecting to d2beiqkhq929f0.cloudfront.net
(d2beiqkhq929f0.cloudfront.net)|18.172.139.61|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3399671 (3.2M) [text/plain]
Saving to: ‘netflix.csv’


netflix.csv           0%[                    ]       0  --.-KB/s
netflix.csv         100%[===================>]   3.24M  --.-KB/s    in
0.08s
```

```
2023-06-27 06:53:34 (41.1 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	list
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG-13	90 min	Docume
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	2021	TV-MA	2 Seasons	Internat TV Shov Dramas, Mysteri
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	September 24, 2021	2021	TV-MA	1 Season	Crime T Shows, Internat TV Shov Act...
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	NaN	September 24, 2021	2021	TV-MA	1 Season	Docuser Reality T
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	September 24, 2021	2021	TV-MA	2 Seasons	Internat TV Shov Romant Shows, T

```
df.shape
```

```
(8807, 12)
```

```
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.nunique()
```

```
show_id         8807
type               2
title           8807
director        4528
cast            7692
country          748
date_added      1767
release_year      74
rating            17
duration         220
listed_in        514
description     8775
dtype: int64
```

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

```
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
```

```
df.isnull().sum()/len(df)*100
```

```
show_id          0.000000
type             0.000000
title            0.000000
director        29.908028
cast             9.367549
country          9.435676
date_added       0.113546
release_year     0.000000
rating           0.045418
duration         0.034064
listed_in        0.000000
description      0.000000
dtype: float64
```

Insight: director,cast and country have a lot of missing values,these should be taken care of.

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

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration
count	8807	8807	8807	6173	7982	7976	8797	8807.000000	8803	8804
unique	8807	2	8807	4528	7692	748	1767	NaN	17	220
top	s1	Movie	Dick Johnson Is Dead	Rajiv Chilaka	David Attenborough	United States	January 1, 2020	NaN	TV-MA	1 Season
freq	1	6131	1	19	19	2818	109	NaN	3207	1793
mean	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2014.180198	NaN	NaN
std	NaN	NaN	NaN	NaN	NaN	NaN	NaN	8.819312	NaN	NaN

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration
min	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1925.000000	NaN	NaN
25%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2013.000000	NaN	NaN
50%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2017.000000	NaN	NaN
75%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2019.000000	NaN	NaN
max	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2021.000000	NaN	NaN

preprocessing of data - (unnesting fields like actor, director and country)

```
#unnesting the cast column
df['cast'] = df['cast'].str.split(',')
df = df.explode(['cast'],ignore_index=True)
df['cast']
```

```
0                          NaN
1                   Ama Qamata
2                   Khosi Ngema
3                 Gail Mabalane
4                Thabang Molaba
                    ...
64946          Manish Chaudhary
64947              Meghna Malik
64948             Malkeet Rauni
64949            Anita Shabdish
64950     Chittaranjan Tripathy
Name: cast, Length: 64951, dtype: object
```

```
#unnesting the director column
df['director'] = df['director'].str.split(',')
df = df.explode(['director'],ignore_index=True)
df['director']
```

```
0        Kirsten Johnson
1                    NaN
2                    NaN
3                    NaN
4                    NaN
              ...
70807        Mozez Singh
70808        Mozez Singh
70809        Mozez Singh
70810        Mozez Singh
```

```
70811        Mozez Singh
Name: director, Length: 70812, dtype: object
```

```
#unnesting the country column
df['country'] = df['country'].str.split(',')
df = df.explode(['country'],ignore_index=True)
df['country']
```

```
0        United States
1         South Africa
2         South Africa
3         South Africa
4         South Africa
             ...
89410            India
89411            India
89412            India
89413            India
89414            India
Name: country, Length: 89415, dtype: object
```

```
# splitting any extra spaces,columns
df['country'].str.strip()
```

```
0        United States
1         South Africa
2         South Africa
3         South Africa
4         South Africa
             ...
89410            India
89411            India
89412            India
89413            India
89414            India
Name: country, Length: 89415, dtype: object
```

```
df['listed_in'] = df['listed_in'].str.split(',')
df = df.explode(['listed_in'],ignore_index=True)
df['listed_in']
```

```
0                  Documentaries
1         International TV Shows
2                     TV Dramas
3                   TV Mysteries
```

```
4         International TV Shows
                  ...
202060      International Movies
202061          Music & Musicals
202062                   Dramas
202063      International Movies
202064          Music & Musicals
Name: listed_in, Length: 202065, dtype: object
```

```
df['listed_in'] = df['listed_in'].str.strip()
df['director'] = df['director'].str.strip()
df['cast'] = df['cast'].str.strip()
df['country'] = df['country'].str.strip()

#converting the duration from object  to numeric datatype
df['duration'] = df['duration'].str.extract('(\d+)')
df['duration'] = pd.to_numeric(df['duration'])

df['date_added'] = pd.to_datetime(df['date_added'],errors='coerce')

 df['release_year'] = pd.to_numeric(df['release_year'])
 df.head()
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	2021-09-25	2020	PG-13	90.0	Documentari
1	s2	TV Show	Blood & Water	NaN	Ama Qamata	South Africa	2021-09-24	2021	TV-MA	2.0	International TV Shows
2	s2	TV Show	Blood & Water	NaN	Ama Qamata	South Africa	2021-09-24	2021	TV-MA	2.0	TV Dramas
3	s2	TV Show	Blood & Water	NaN	Ama Qamata	South Africa	2021-09-24	2021	TV-MA	2.0	TV Mysteries

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_
4	s2	TV Show	Blood & Water	NaN	Khosi Ngema	South Africa	2021-09-24	2021	TV-MA	2.0	International TV Shows

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 202065 entries, 0 to 202064
Data columns (total 12 columns):
 #   Column        Non-Null Count   Dtype
---  ------        --------------   -----
 0   show_id       202065 non-null  object
 1   type          202065 non-null  object
 2   title         202065 non-null  object
 3   director      151422 non-null  object
 4   cast          199916 non-null  object
 5   country       190168 non-null  object
 6   date_added    201907 non-null  datetime64[ns]
 7   release_year  202065 non-null  int64
 8   rating        201998 non-null  object
 9   duration      202062 non-null  float64
 10  listed_in     202065 non-null  object
 11  description   202065 non-null  object
dtypes: datetime64[ns](1), float64(1), int64(1), object(9)
memory usage: 18.5+ MB
```

Range of data

```
print(df['date_added'].min())
print(df['date_added'].max())
```

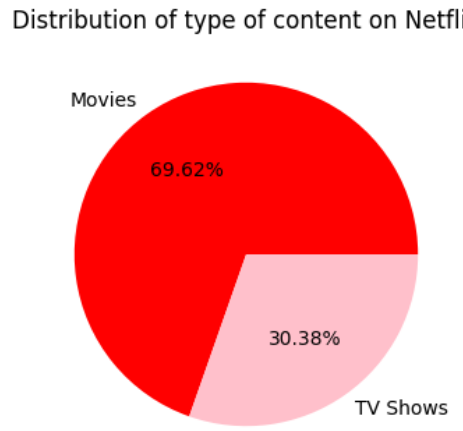
```
2008-01-01 00:00:00
2021-09-25 00:00:00
```

```
from matplotlib import pyplot as plt
import seaborn as sns
import pandas as pd
```

Univariate and Bivariate analysis

content distribution

```
type_count = df.groupby(['type'])['title'].nunique()
labels = ['Movies',"TV Shows"]
total_ = type_count.loc['Movie']+type_count.loc['TV Show']

mper = ((type_count.loc['Movie']/total_)*100).round(2)
tvper = ((type_count.loc['TV Show']/total_)*100).round(2)

plt.figure(figsize=(12,4))
plt.pie([mper,tvper],labels=labels,autopct='%1.2f%%',colors=['red','pink'])

plt.title('Distribution of type of content on Netflix')
plt.show()
```



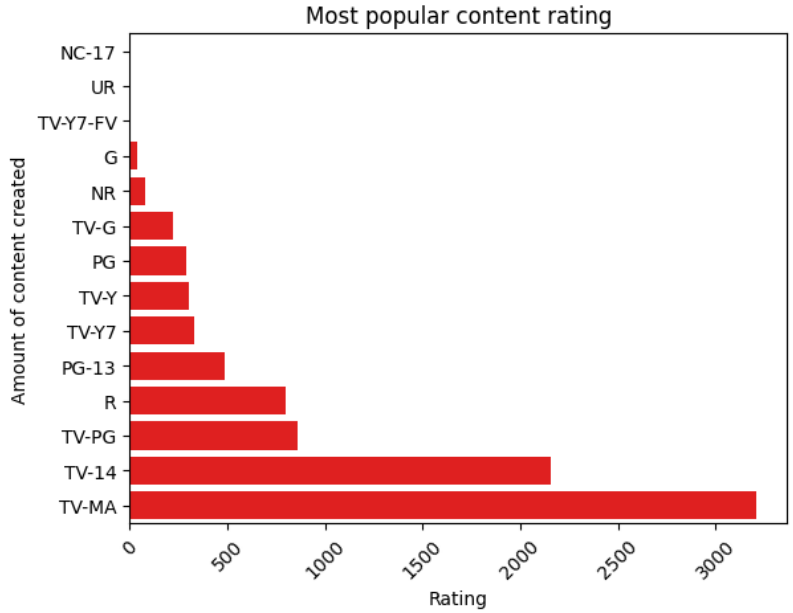
Insights:

movies make up 70 % of the content and Tv shows about 30%

Distribution of Rating

```
rating =df.groupby(['rating'])
        ['title'].nunique().reset_index().sort_values(by='title')
var = rating.loc[rating['rating'].isin(['74 min','84 min','66
        min'])].index
rating.drop(var,inplace=True)

sns.barplot(data=rating,x='title',y='rating',color='red')

plt.xticks(rotation=45)
plt.xlabel('Rating')
plt.ylabel('Amount of content created')
```

```
plt.title("Most popular content rating")
plt.show()
```



Insgigts:

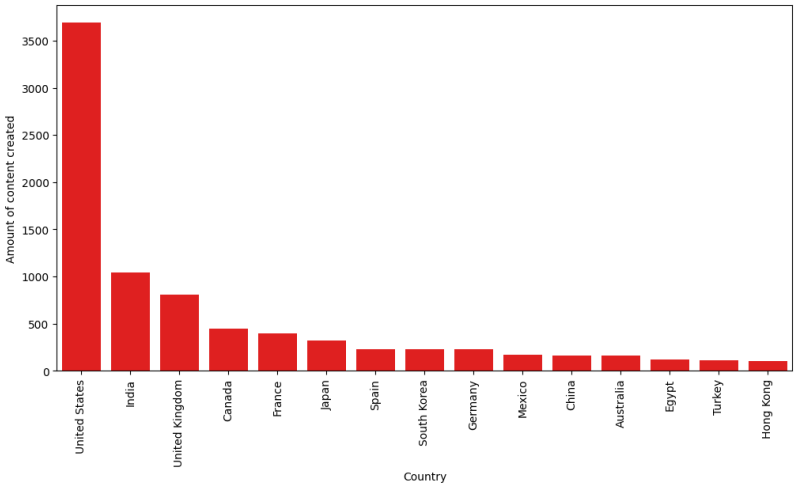
TV-MA refers to mature and adult content that may not be suitable for ages under 17 and TV-14 refers to shows that are unsuitable for ages under 14.

Content that falls under these categories is preferred, followed by TV-PG and R, R indicates restricted content, and TV-PG indicates parental guidance.

We can say that most of the netflix audince are mature people

Country wise distribution of data

```
country_dist = df.groupby(['country'])
        ['title'].nunique().reset_index().sort_values(by='title',ascending=False)
        [:15]


fig=plt.figure(figsize=(12,6))
sns.barplot(data=country_dist,x='country',y='title',color='red')
plt.xticks(rotation=90)
plt.xlabel('Country')
plt.ylabel('Amount of content created')

Text(0, 0.5, 'Amount of content created')
```

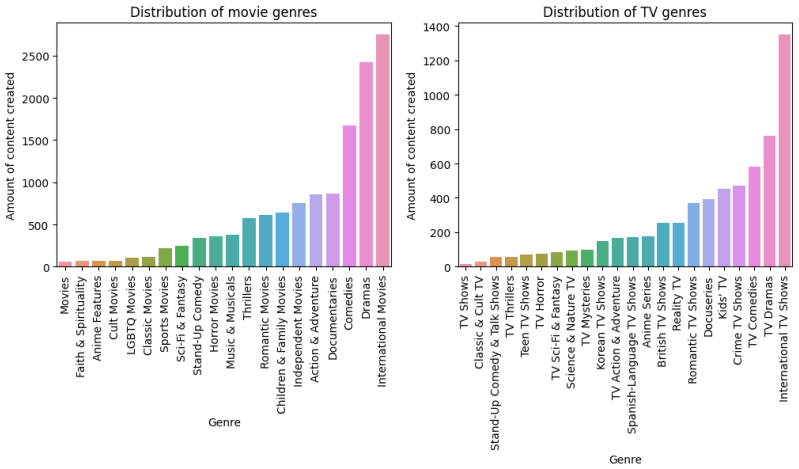


Insights: most of the content belongs to US, India, UK, Canada and France.

Comparisn between Tv shows and Movies

distribution of movies and Tv Shows created across genres

```
movie_df =df.loc[df['type'] == 'Movie']
tv_df = df.loc[df['type'] == 'TV Show']

genre_mov = movie_df.groupby(['listed_in'])
        ['show_id'].nunique().reset_index().sort_values(by='show_id')
genre_tv = tv_df.groupby(['listed_in'])
        ['show_id'].nunique().reset_index().sort_values(by='show_id')

fig = plt.figure(figsize=(12,4))
plt.subplot(1,2,1)
sns.barplot(data=genre_mov,x='listed_in',y='show_id')
plt.xticks(rotation=90)
plt.xlabel('Genre')
plt.ylabel('Amount of content created')
plt.title('Distribution of movie genres')
plt.subplot(1,2,2)
sns.barplot(data=genre_tv,x='listed_in',y='show_id')
plt.xticks(rotation=90)
plt.xlabel('Genre')
plt.ylabel('Amount of content created')
plt.title('Distribution of TV genres')
plt.show()
```



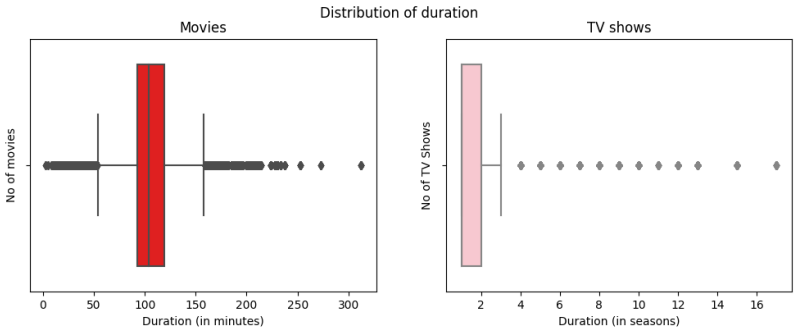
Insights: International show/movie is any movie/show that is made outside of the USA and is not in english language,international movies are in high demand followed by dramas,comedies and documentries in Movies,where as in TV shows we have international Tv shows followed by drams,comedies and crime Tv shows.

We can see that In movies and tv shows exactcly same orsimilar genres are in demand

Duration

```
median = movie_df['duration'].median()
no_of_movies = movie_df['duration'].loc[movie_df['duration'] ==
        median].count()
```

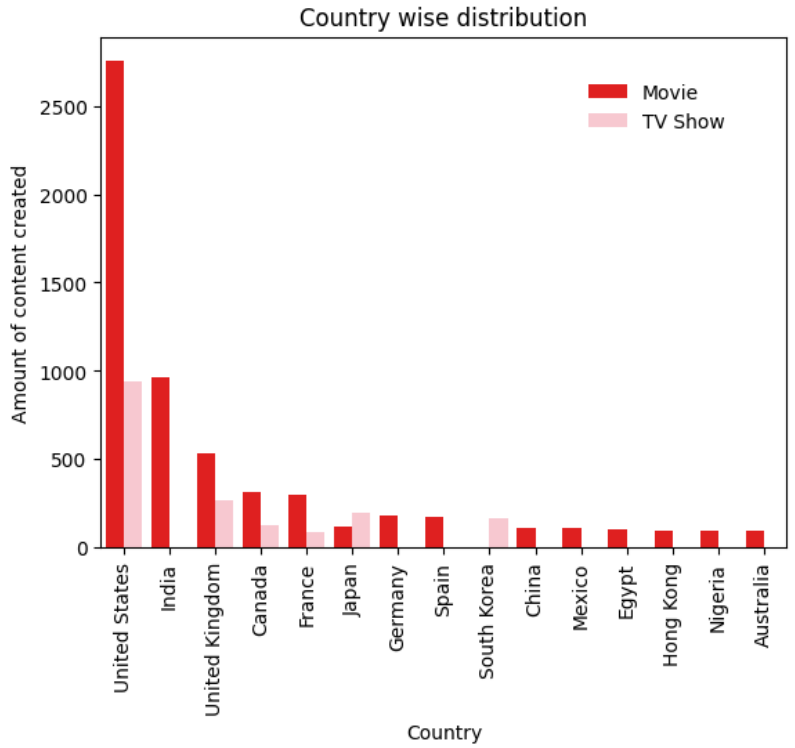
```
fig = plt.figure(figsize=(12,4))
plt.subplot(1,2,1)
sns.boxplot(data=movie_df,x='duration',color='red')
plt.xlabel('Duration (in minutes)')
plt.ylabel('No of movies')
plt.title("Movies")
plt.subplot(1,2,2)
sns.boxplot(data=tv_df,x='duration',color='pink')
plt.xlabel('Duration (in seasons)')
plt.ylabel('No of TV Shows')
plt.title("TV shows")
plt.suptitle("Distribution of duration")
plt.show()
```



Insights: median duration for movies is more than 100 minutes and for Tv show is one season

Type of content in differnt countries

```
country_hue = df.groupby(['country','type'])
        ['title'].nunique().reset_index().sort_values(by='title',ascending=False)
        [:20]


sns.barplot(data=country_hue,x='country',y='title',hue='type',palette=['red',
        'pink'])


plt.xticks(rotation=90)
plt.xlabel('Country')
plt.ylabel('Amount of content created')
plt.legend(loc=(0.7,0.8),frameon=False)
plt.title('Country wise distribution')
plt.show()
```



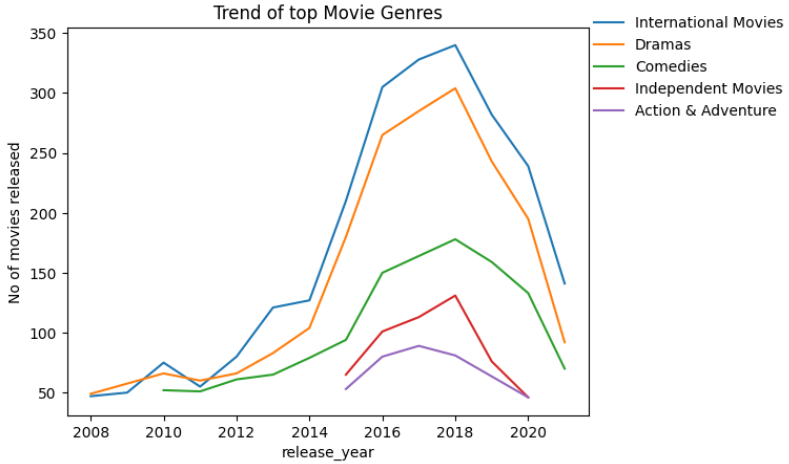
Insights:

Movie content created is more than TV content but in South Korea and Japan Tv shows are popoular

Top 5 movie and Tv genres across years

```
top_movie_list = movie_df['listed_in'].value_counts().reset_index()
        ['index'].head()
top_movie_df =
        movie_df.loc[movie_df['listed_in'].isin(top_movie_list)]
genre_trnd = top_movie_df.groupby(['release_year','listed_in'])
        ['title'].nunique().reset_index().sort_values(by='title',
        ascending=False)[:50]

sns.lineplot(data =
        genre_trnd,x='release_year',y='title',hue='listed_in',ci=None)


plt.legend(loc=(1,0.75),frameon=False,ncol=1)
plt.ylabel('No of movies released')
plt.title('Trend of top Movie Genres')
plt.show()
```

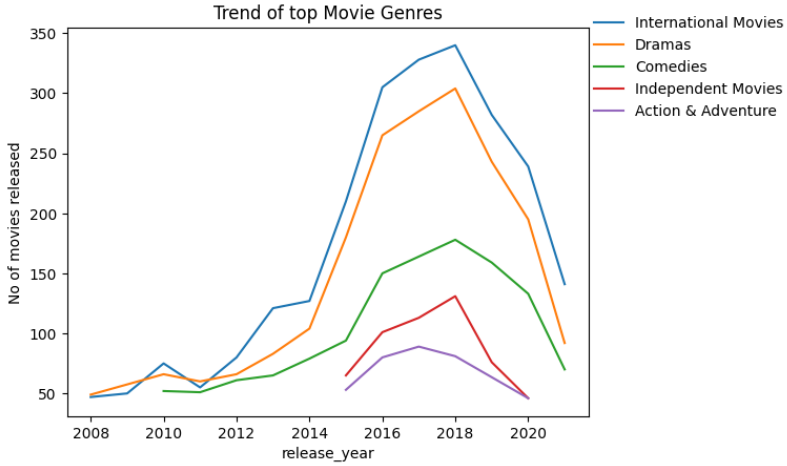
```
<ipython-input-57-e61a5bb3408f>:1: FutureWarning:

The `ci` parameter is deprecated. Use `errorbar=None` for the same
effect.

  sns.lineplot(data =
genre_trnd,x='release_year',y='title',hue='listed_in',ci=None)
```




```
top_movie_list = movie_df['listed_in'].value_counts().reset_index()
        ['index'].head()
top_movie_df =
        movie_df.loc[movie_df['listed_in'].isin(top_movie_list)]
genre_trnd = top_movie_df.groupby(['release_year','listed_in'])
        ['title'].nunique().reset_index().sort_values(by='title',
        ascending=False)[:50]

top_tv_list = tv_df['listed_in'].value_counts().reset_index()
        ['index'].head()
top_tv_df = tv_df.loc[tv_df['listed_in'].isin(top_movie_list)]
genre_trnd = top_movie_df.groupby(['release_year','listed_in'])
        ['title'].nunique().reset_index().sort_values(by='title',
        ascending=False)[:50]

sns.lineplot(data =
        genre_trnd,x='release_year',y='title',hue='listed_in',ci=None)

plt.legend(loc=(1,0.75),frameon=False,ncol=1)
plt.ylabel('No of movies released')
plt.title('Trend of top Movie Genres')
plt.show()

<ipython-input-59-c78ccc144c7d>:5: FutureWarning:
```

```
The `ci` parameter is deprecated. Use `errorbar=None` for the same
effect.
```

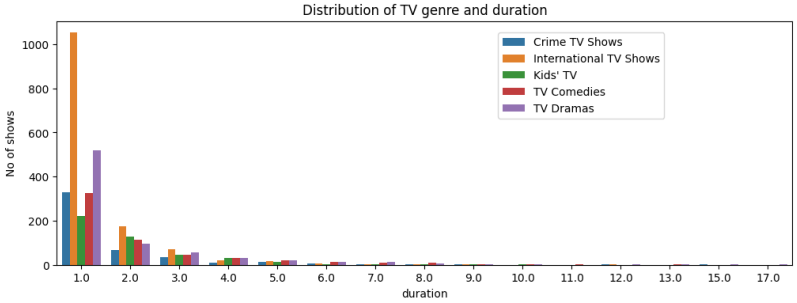
```
  sns.lineplot(data =
genre_trnd,x='release_year',y='title',hue='listed_in',ci=None)
```



insgits

Duration with respect to genre

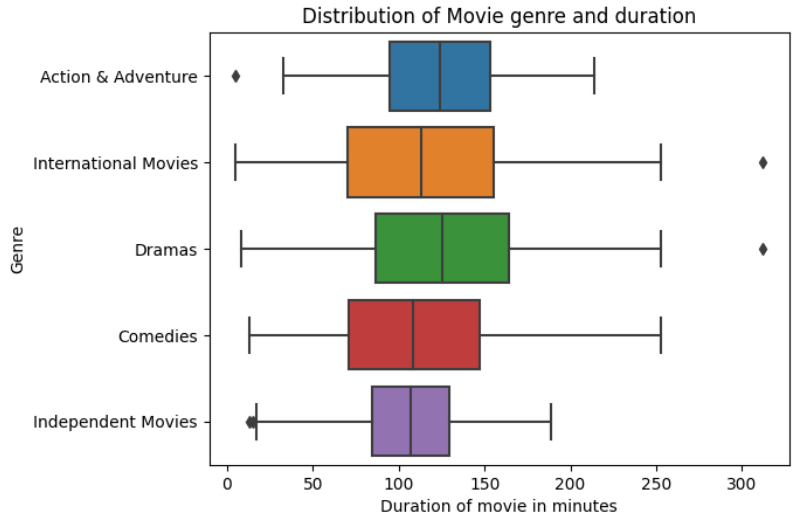
```
top_tv_list = tv_df['listed_in'].value_counts().reset_index()
        ['index'].head()
top_tv_df = tv_df.loc[tv_df['listed_in'].isin(top_tv_list)]
duration_genre = top_tv_df.groupby(['duration','listed_in'])
        ['title'].nunique().reset_index()


plt.figure(figsize=(12,4))
sns.barplot(data =
        duration_genre,x='duration',y='title',hue='listed_in')
plt.ylabel('No of shows')
plt.legend(loc=(0.6,0.6))
plt.title('Distribution of TV genre and duration')
plt.show()
```



Insights:

duration of tv comedies and shows is high.

```
duration_genre_movie = top_movie_df.groupby(['duration','listed_in'])
        ['title'].nunique().reset_index()


sns.boxplot(data=duration_genre_movie,x='duration',y = 'listed_in')
plt.xlabel('Duration of movie in minutes')
plt.ylabel('Genre')
plt.title("Distribution of Movie genre and duration")
plt.show()
```



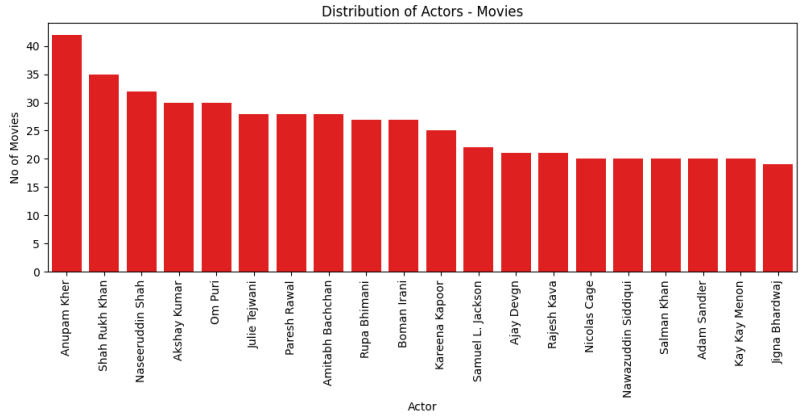
insights:

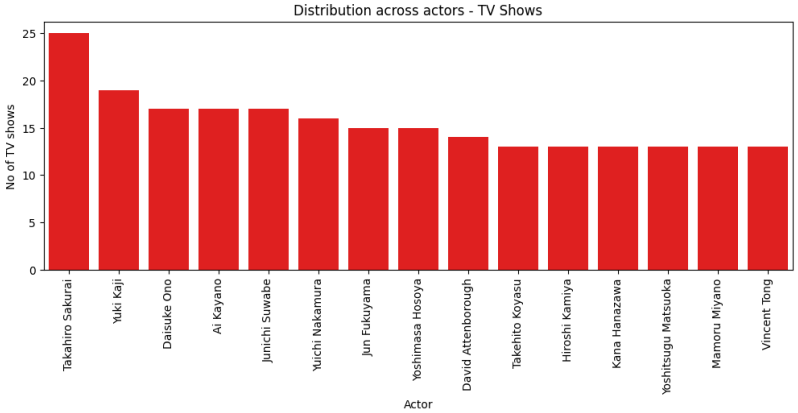
movies range between 75 mins to 175 mins

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

```
top15_tvcast = tv_df.groupby(['cast'])
        ['title'].nunique().sort_values(ascending=False).reset_index().head(15)

top20_cast = movie_df.groupby(['cast'])
        ['title'].nunique().sort_values(ascending=False).reset_index().head(20)
```

```
plt.figure(figsize=(12,4))
sns.barplot(data=top20_cast,x='cast',y='title',color='red')
plt.xticks(rotation=90)
plt.xlabel("Actor")
plt.ylabel("No of Movies")
plt.title("Distribution of Actors - Movies")
plt.figure(figsize=(12,4))
sns.barplot(data=top15_tvcast,x='cast',y='title',color='red')
plt.xticks(rotation=90)
plt.xlabel("Actor")
plt.ylabel("No of TV shows")
plt.title("Distribution across actors - TV Shows")
plt.show()
```



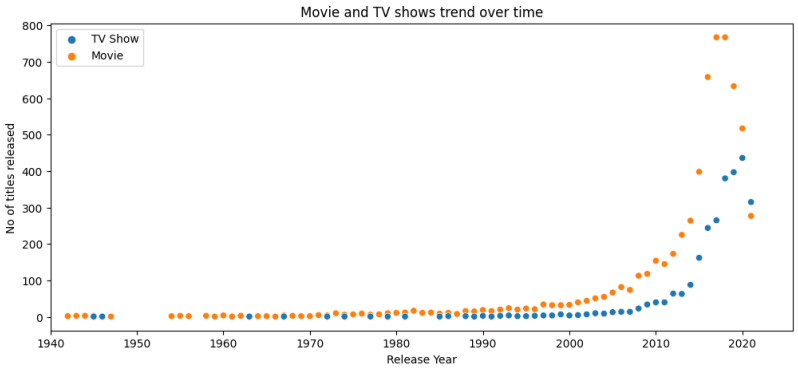


For movies, top actors: Anupam Kher, Shah Rukh Khan, Naseeruddin Shah, Akshay Kumar, Om Puri, Julie Tejwani,Paresh Rawal, Amitabh Bachchan, Rupa Bhimani, Boman Irani,Kareena Kapoor.

For TV shows,top actors are: Takahiro Sakurai, Yuki Kaji, Daisuke Ono, Ai Kayano, Junichi Suwabe,Yuichi Nakamura, Jun Fukuyama, Yoshimasa Hosoya, David Attenborough, Takehito Koyasu, Hiroshi Kamiy

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

```
type_trend = df.groupby(['release_year','type'])
        ['title'].nunique().reset_index()
plt.figure(figsize=(12,5))
sns.scatterplot(data=type_trend,x='release_year',y='title',hue='type')
plt.ylabel("No of titles released")
plt.xlabel("Release Year")
plt.title('Movie and TV shows trend over time')
plt.legend(title=None)
plt.xlim(1940)
plt.show()
```



Insights:

The analysis of the data reveals that the rate of growth for movies started to slow down after 2010. However, the decline in the growth rate of TV content was observed only after 2020. Furthermore, the data for 2019 and 2020 indicates a significant decrease in the number of movies, while the number of TV shows launched during those years increased compared to previous years. These trends suggest a shift in content creation, with a focus on TV shows and a decline in the production of movies during the mentioned period

Understanding what content is available in different countries

```
top5_tv_genre_list = tv_df['listed_in'].value_counts().reset_index()
        ['index'].head()
top5_tv_genre_df =
        tv_df.loc[tv_df['listed_in'].isin(top5_tv_genre_list)]
tv_genre_cntry = top5_tv_genre_df.groupby(['country','listed_in'])
        ['title'].nunique().reset_index().sort_values(by='title',ascending=False)
        [:20]
```

```
tv_genre_cntry.head()
```

	country	listed_in	title
235	United States	TV Comedies	258
236	United States	TV Dramas	232
234	United States	Kids' TV	214
190	South Korea	International TV Shows	152
112	Japan	International TV Shows	151

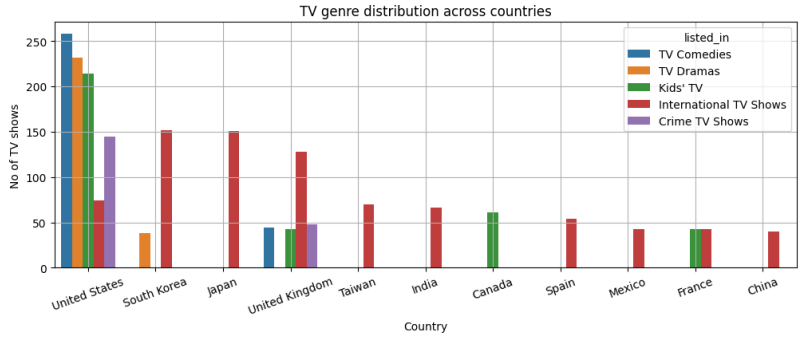
```
top5_movie_genre_list =
        movie_df['listed_in'].value_counts().reset_index()
        ['index'].head()
top5_movie_genre_df =
        movie_df.loc[movie_df['listed_in'].isin(top5_movie_genre_list)]

genre_country_trnd = top5_movie_genre_df.groupby(['country','listed_in'])
        ['title'].nunique().reset_index().sort_values(by='title',ascending=False)
        [:20]
```

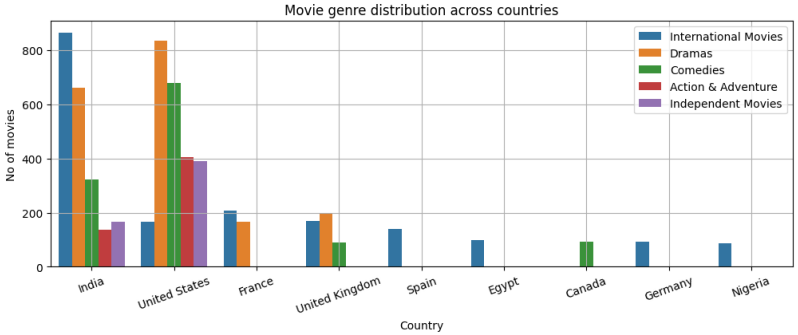
```
genre_country_trnd.head()
```

	country	listed_in	title

	country	listed_in	title
143	India	International Movies	864
368	United States	Dramas	835
367	United States	Comedies	680
141	India	Dramas	662
366	United States	Action & Adventure	404

```
plt.figure(figsize=(12,4))
sns.barplot(data=tv_genre_cntry,x='country',y='title',hue='listed_in')
plt.xlabel('Country')
plt.ylabel("No of TV shows")
plt.title('TV genre distribution across countries')
plt.xticks(rotation=20)
plt.grid()
plt.figure(figsize=(12,4))
sns.barplot(data=genre_country_trnd,x='country',y='title',hue='listed_in')

plt.legend(loc='upper right')
plt.xlabel('Country')
plt.ylabel("No of movies")
plt.title('Movie genre distribution across countries')
plt.xticks(rotation=20)
plt.grid()
plt.show()
```





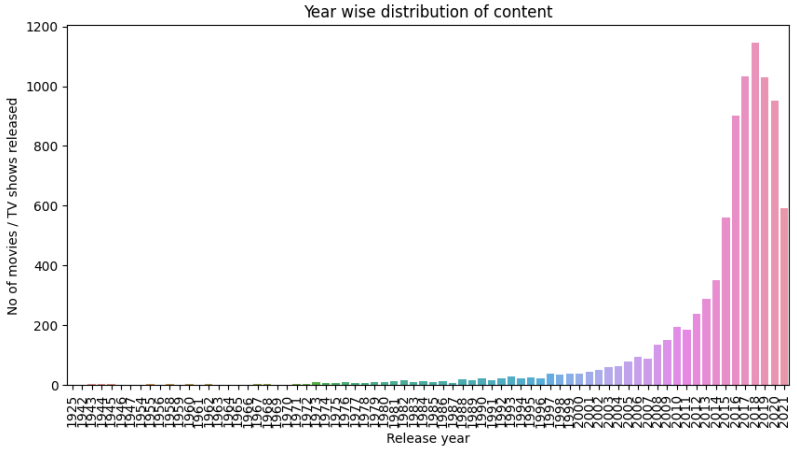
Insights:

Dramas takes the lead with no of movies released, followed by comedy

What is the best time to launch a TV show?

year wise content

```
year_df = df.groupby(['release_year'])
        ['title'].nunique().reset_index()
fig=plt.figure(figsize=(10,5))
sns.barplot(data=year_df,x='release_year',y='title')
plt.xlabel('Release year')
plt.ylabel('No of movies / TV shows released')
plt.xticks(rotation=90)
plt.title("Year wise distribution of content")
plt.show()
```

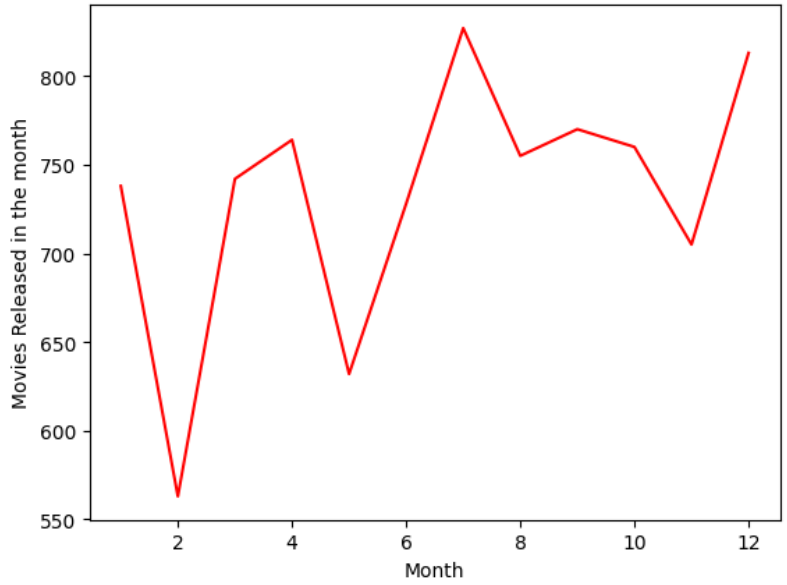


We can observe that the movies/tv shows released steadily increased initially and then exponentially from 2015.

```
df['new_date'] = pd.to_datetime(df['date_added'])
df['month_new'] = df['new_date'].dt.month
df['week_new'] = df['new_date'].dt.week
df['year_new'] = df['new_date'].dt.year
df.head()
```

```
df_year=df.groupby(['month_new']).agg({"title":"nunique"}).reset_index()

sns.lineplot(data=df_year, x='month_new', y='title',color='red')
plt.ylabel("Movies Released in the month")
plt.xlabel("Month")
plt.show()
```

```
<ipython-input-68-6fe0900e32ab>:3: FutureWarning: Series.dt.weekofyear
and Series.dt.week have been deprecated. Please use
Series.dt.isocalendar().week instead.
  df['week_new'] = df['new_date'].dt.week
```

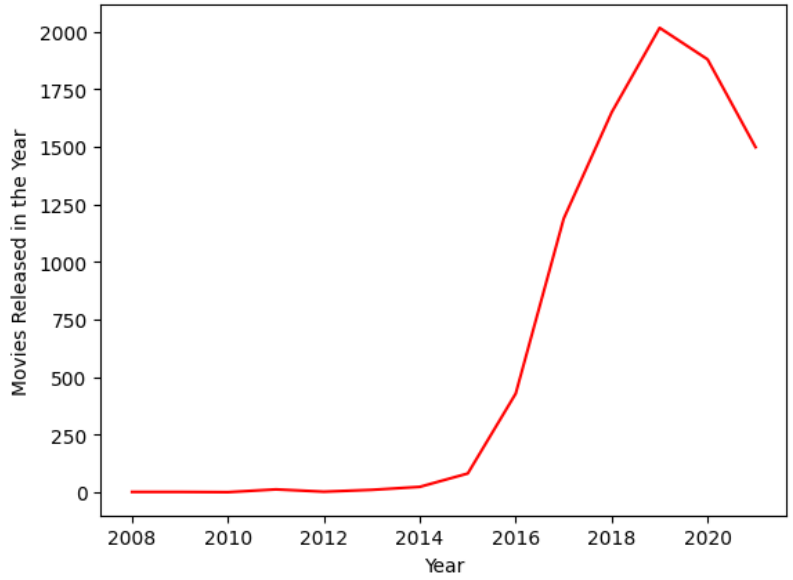


Insights: Most of the content is added in the first and last months across Netflix

```
df.groupby(['year_new']).agg({"title":"nunique"})
```

```
df_year=df.groupby(['year_new']).agg({"title":"nunique"}).reset_index()
```

```
sns.lineplot(data=df_year, x='year_new', y='title',color='red')
plt.ylabel("Movies Released in the Year")
plt.xlabel("Year")
plt.show()
```



Insights: content release which are later uploaded to Netflix has increased since 1980 till 2020 though later reduced certainly due to COVID-19

Recommendations:

Based on the given data, drama and comedy are the most popular genres with the top countries.while creating content in comedy and drama genres, we should also focus on adding and producing content from genres like crime TV, action and adventure movies, kids movies and TV shows and documentaries.

The target audience is recommended to be 14+ and above ratings while for UK, its recommended to be completely Mature/R content, if we increse the now of tv shows /movies and family related content,consumer base can increase

While creating content, take into consideration the popular actors/directors for that country. Also take into account the director-actor combination which is highly recommended.

TV show should be released in the months of July or December.