SCALER

BUSINESS CASE

ON

NETFLIX

By

Aman Priyadarshi

DSML Nov’23 Morning Batch

Google Colab Link - https://colab.research.google.com/drive/17G8roUJrkCMCCcqCAQHHN8PcUnD8g3He?usp=drive_link

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

```
#reading CSV file
df = pd.read_csv("/content/netflix.csv")
```

```
df.shape #to get the shape of dataform
```

```
(8807, 12)
```

DataForm is found to be having 8807 rows and 12 columns

+ Code + Text

```
#to get the basic informations of the data
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
```

We get to know the non null contents and data types of all the attributes of the dataform.

```
#to get the sample of the dataform to make the overall understanding for analysis.
df.sample(5)
```

	show_id	type	title	director	cast	country	date_added	release_
1651	s1652	Movie	The Christmas Chronicles: Part Two	Chris Columbus	Kurt Russell, Goldie Hawn, Darby Camp, Jahzir ...	Canada	November 25, 2020	
1758	s1759	TV Show	LEGO Ninjago	NaN	Michael Adamthwaite, Paul Dobson, Kelly Metzge...	Canada	November 1, 2020	
			Abducted					

We get to see the overall data and range and variations of the attributes of the dataform

```
#to get the statistical analysis of the attributes having numbers.
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

Release year is the only number attribute in the dataform for which stastical analysis has been obtained.

```
#to get sum of null values for each attribute
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
```

We can see that six of the attributes are having null values for which the sum has been obtained. Director column has the most nulls followed by country, cast, date_added and rating.

```
#Formatting of the director column
df_director_r = pd.DataFrame(df['director'].apply(lambda x: str(x).split(',')).tolist(), index =df['title'])

df_director = df_director_r.stack().reset_index()
df_director.drop('level_1', axis = 1, inplace = True)
df_director.rename(columns ={0:'director'}, inplace = True)
df_director.head()
```

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

```
#Formatting of the cast column
df_cast_r = pd.DataFrame(df['cast'].apply(lambda x: str(x).split(',')).tolist(), index =df['title'])

df_cast = df_cast_r.stack().reset_index()
df_cast.drop('level_1', axis = 1, inplace = True)
df_cast.rename(columns ={0:'cast'}, inplace = True)
df_cast.head()
```

	title	cast
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

```
#Formatting of the country column
df_country_r = pd.DataFrame(df['country'].apply(lambda x: str(x).split(',')).tolist(), index =df['title'])

df_country = df_country_r.stack().reset_index()
df_country.drop('level_1', axis = 1, inplace = True)
df_country.rename(columns ={0:'country'}, inplace = True)
df_country.head()
```

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

import seaborn as sns df=sns.load_dataset('taxis') import seaborn as sns df=sns.load_dataset('titanic')

```
#Formatting of the listed_in column
df_listedin1 = pd.DataFrame(df['listed_in'].apply(lambda x: str(x).split(',')).tolist(), index =df['title'])

df_listedin = df_listedin1.stack().reset_index()
df_listedin.drop('level_1', axis = 1, inplace = True)
df_listedin.rename(columns ={0:'listed_in'}, inplace = True)
df_listedin.head()
```

	title	listed_in
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

```
#merging the new formatted columns
df_new1=df_director.merge(df_cast,how='inner', on='title')
df_new1
```

	title	director	cast
0	Dick Johnson Is Dead	Kirsten Johnson	nan
1	Blood & Water	nan	Ama Qamata
2	Blood & Water	nan	Khosi Ngema
3	Blood & Water	nan	Gail Mabalane
4	Blood & Water	nan	Thabang Molaba
...	...	...	...
70807	Zubaan	Mozez Singh	Manish Chaudhary
70808	Zubaan	Mozez Singh	Meghna Malik
70809	Zubaan	Mozez Singh	Malkeet Rauni
70810	Zubaan	Mozez Singh	Anita Shabdish
70811	Zubaan	Mozez Singh	Chittaranjan Tripathy

70812 rows × 3 columns

```
df_new2=df_new1.merge(df_country,how='inner', on='title')
df_new2
```

	title	director	cast	country
0	Dick Johnson Is Dead	Kirsten Johnson	nan	United States
1	Blood & Water	nan	Ama Qamata	South Africa
2	Blood & Water	nan	Khosi Ngema	South Africa
3	Blood & Water	nan	Gail Mabalane	South Africa
4	Blood & Water	nan	Thabang Molaba	South Africa
...	...	...	...	...
89410	Zubaan	Mozez Singh	Manish Chaudhary	India
89411	Zubaan	Mozez Singh	Meghna Malik	India
89412	Zubaan	Mozez Singh	Malkeet Rauni	India
89413	Zubaan	Mozez Singh	Anita Shabdish	India
89414	Zubaan	Mozez Singh	Chittaranjan Tripathy	India

89415 rows × 4 columns

year_added

```
df["date_added"] = df["date_added"].apply(lambda x: str(x).strip())
df["date_added"] = pd.to_datetime(df["date_added"])
df["date_added"] = df["date_added"].dt.year
```

```
df_new3=df_new2.merge(df_listedin,how='inner', on='title')
df_new3
```

	title	director	cast	country	listed_in
0	Dick Johnson Is Dead	Kirsten Johnson	nan	United States	Documentaries
1	Blood & Water	nan	Ama Qamata	South Africa	International TV Shows
2	Blood & Water	nan	Ama Qamata	South Africa	TV Dramas
3	Blood & Water	nan	Ama Qamata	South Africa	TV Mysteries
4	Blood & Water	nan	Khosi Ngema	South Africa	International TV Shows
...	...	...	...	...	...
202060	Zubaan	Mozez Singh	Anita Shabdish	India	International Movies
202061	Zubaan	Mozez Singh	Anita Shabdish	India	Music & Musicals
202062	Zubaan	Mozez Singh	Chittaranjan Tripathy	India	Dramas

```
#merging formatted columns columns with the dataframe
df_final = df_new3.merge(df[['show_id', 'type', 'title','release_year','date_added','rating', 'duration','description']] , how = 'inner
df_final.sample(10)
```

	title	director	cast	country	listed_in	show_id	type	release_y
27599	Insidious	James Wan	Patrick Wilson	United States	Thrillers	s1119	Movie	2
82827	Chicago Typewriter	nan	Ji Dae-han	South Korea	Romantic TV Shows	s3463	TV Show	2
95775	The Last Runway	Pietro Scappini	Luis Aguirre	Argentina	International Movies	s4065	Movie	2
130694	Bleach The Movie: Fade to Black	Noriyuki Abe	Masakazu Morita	Japan	Action & Adventure	s5749	Movie	2
145521	Chashme Buddoor	David Dhawan	Ashish Verma	India	International Movies	s6450	Movie	2

Here we get the formatted data for the columns director, cast,listed_in and country

```
#to get the shape of final formatted data
df_final.shape
```

```
(202065, 12)
```

Now we see that after formatting, the dataframe has 202065 rows with 12 columns

```
#to remove the nan values for the columns cast, director, country and listed in
df_final['cast'].replace (['nan'], ['Unknown Actor'], inplace = True)
df_final['director'].replace (['nan'], ['Unknown Director'], inplace = True)
df_final['country'].replace (['nan'], ['Unknown country'], inplace = True)
df_final['listed_in'].replace (['nan'], ['Unknown list'], inplace = True)
df_final.sample(5)
```

	title	director	cast	country	listed_in	show_id	type	release_y
69259	Cuddle Weather	Rod Cabatana Marmol	Aleck Bovick	Philippines	Romantic Movies	s2913	Movie	2
98099	Just Love	Andy Caballero	Daniela Aita	Argentina	Dramas	s4174	Movie	2
		Unknown	Lino		Crime TV		TV	

It is now seen through the sample data that nan values have eradicated from the four columns

```
#to sort the data based on the release year in ascending order
df_final.sort_values(by=['release_year'])
df_final.head(10)
```

	title	director	cast	country	listed_in	show_id	type	release_year
0	Dick Johnson Is Dead	Kirsten Johnson	Unknown Actor	United States	Documentaries	s1	Movie	2020
1	Blood & Water	Unknown Director	Ama Qamata	South Africa	International TV Shows	s2	TV Show	2021
2	Blood & Water	Unknown Director	Ama Qamata	South Africa	TV Dramas	s2	TV Show	2021

```
#to remove the rows with zero as the value of duration column
df_final[df_final['duration'].isnull()]
```

	title	director	cast	country	listed_in	show_id	type	release_year
126582	Louis C.K. 2017	Louis C.K.	Louis C.K.	United States	Movies	s5542	Movie	2017

```
#to count the number of unique elements for each attribute
df_final.nunique()
```

```
title            8807
director         5121
cast            39297
country           198
listed_in          73
show_id          8807
type                2
release_year       74
date_added         14
rating             17
duration          220
description      8775
dtype: int64
```

Output gives us the number of unique elements for each attribute.We can see that the row cast has the maximum number of unique elements followed by title and description whereas, type has the minimum number of unique elements followed by date added and rating.

```
#to count the number of countries
country_count = df_final["country"].value_counts()
country_counts = country_count.head(10)
country_counts
```

```
country
United States      49868
India              22139
Unknown country    11897
United Kingdom      9733
 United States      9482
Japan               7317
South Korea         4628
Canada              4395
Spain               4255
France              4182
Name: count, dtype: int64
```

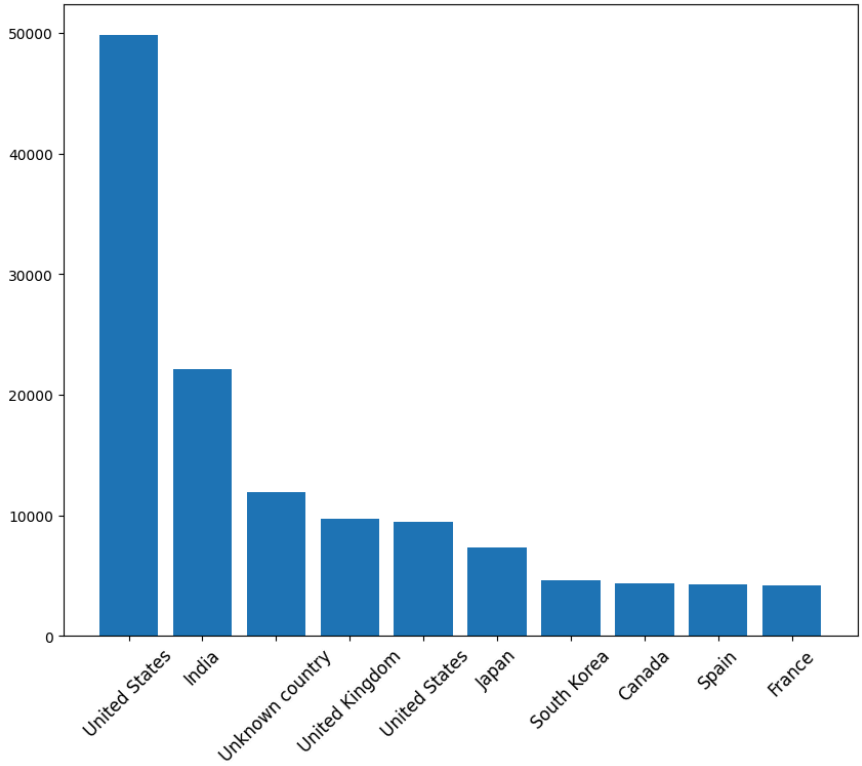
```
#declaring x-axis
x_data = country_counts.index
x_data
```

```
Index(['United States', 'India', 'Unknown country', 'United Kingdom',
       ' United States', 'Japan', 'South Korea', 'Canada', 'Spain', 'France'],
      dtype='object', name='country')
```

```
#declaring y-axis
y_data = country_counts
y_data
```

```
country
United States      49868
India              22139
Unknown country    11897
United Kingdom      9733
 United States      9482
Japan               7317
South Korea         4628
Canada              4395
Spain               4255
France              4182
Name: count, dtype: int64
```

```
#plotting bar plot to find the distribution of titles among top 10 countries
plt.figure(figsize = (10, 8))
plt.bar(x_data, y_data)
plt.xticks(rotation = 45, fontsize = 12,)
plt.show()
```



Here we can see that US tops the list with most number of releases followed by India and UK.

Recommendation - The countries with less number of releases should be focused and it should be increased with more of global contents in local languages along with more of local content addition.

```
#to get the year wise trend of releases done
ry_counts = df_final['release_year'].value_counts()
ry_counts2=ry_counts.head(10)
ry_counts2
```

```
release_year
2018    24441
2019    21931
2017    20516
2020    19697
2016    18465
```

```
2015    14128
2021    11894
2014     9098
2013     7745
2012     6354
Name: count, dtype: int64
```

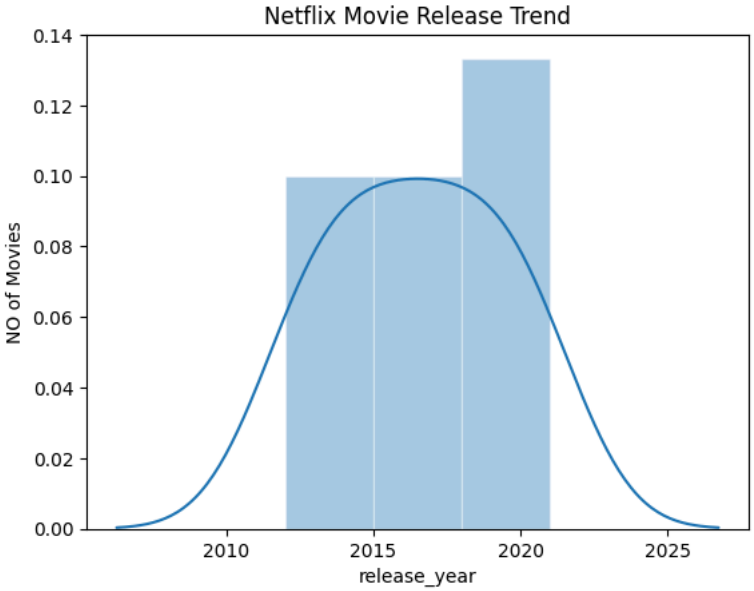
```
x_data = ry_counts2.index
x_data
```

```
Index([2018, 2019, 2017, 2020, 2016, 2015, 2021, 2014, 2013, 2012], dtype='int64', name='release_year')
```

distplot

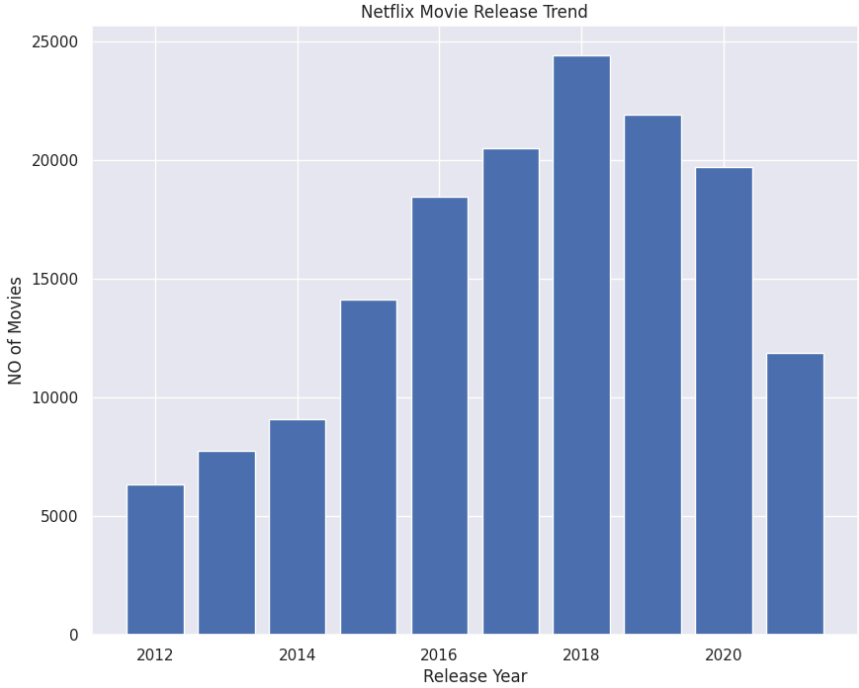
```
#to plot the displot to get the yearwise release trend
plt.title("Netflix Movie Release Trend")
plt.xlabel("Release Year")
plt.ylabel("NO of Movies")
sns.set(rc={"figure.figsize": (10,8)});np.random.seed(0)
ax = sns.distplot(x_data)
plt.show()
```

```
<ipython-input-31-9af9467b6696>:5: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  ax = sns.distplot(x_data)
```

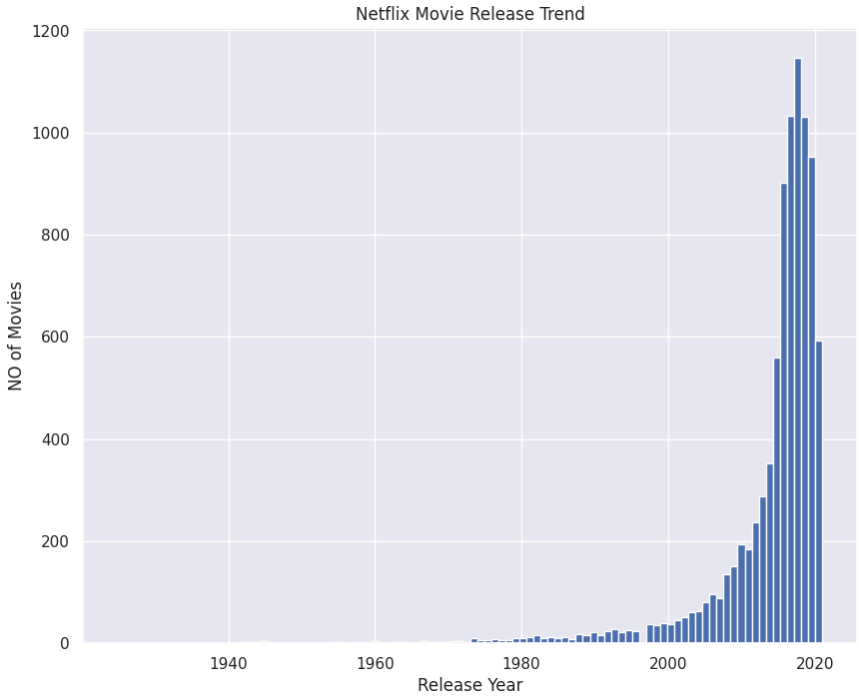



```
#to plot bar graph to make the trend more clear
plt.title("Netflix Movie Release Trend")
plt.xlabel("Release Year")
plt.ylabel("NO of Movies")
x_bar=year_counts2.index
y_bar=year_counts2
plt.bar(x_bar,y_bar)
```

```
<BarContainer object of 10 artists>
```




```
plt.title("Netflix Movie Release Trend")
plt.xlabel("Release Year")
plt.ylabel("NO of Movies")
plt.hist(df["release_year"],bins=100)
plt.show()
```



The above three graphs shows us the trend which increased gradually till 2018 but went downside for the next 3 years.

Recommendation - It should be made sure that the creators keep seeking Netflix with new contents in a regular manner for which the the number of audience should be consistent. This should be made sure with good offers and plans to both the parties as the domain of OTT is increasing gradually.

```
d_counts = df_final['duration'].value_counts()
d_counts2=d_counts.head(10)
d_counts2
```

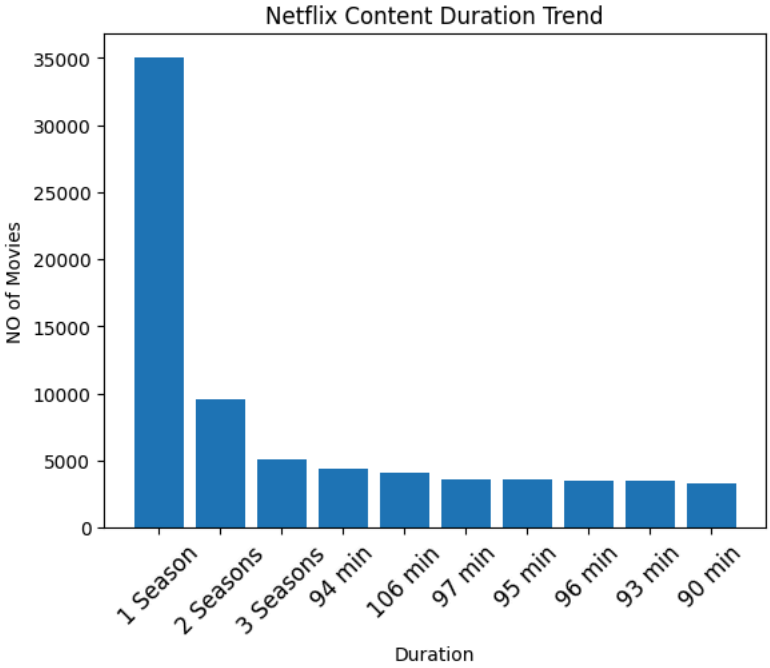
```
duration
1 Season     35035
2 Seasons     9559
3 Seasons     5084
94 min        4343
106 min       4040
97 min        3624
95 min        3560
96 min        3511
93 min        3480
90 min        3305
Name: count, dtype: int64
```

```
x_data2 = d_counts2.index
x_data2
```

```
Index(['1 Season', '2 Seasons', '3 Seasons', '94 min', '106 min', '97 min',
       '95 min', '96 min', '93 min', '90 min'],
      dtype='object', name='duration')
```

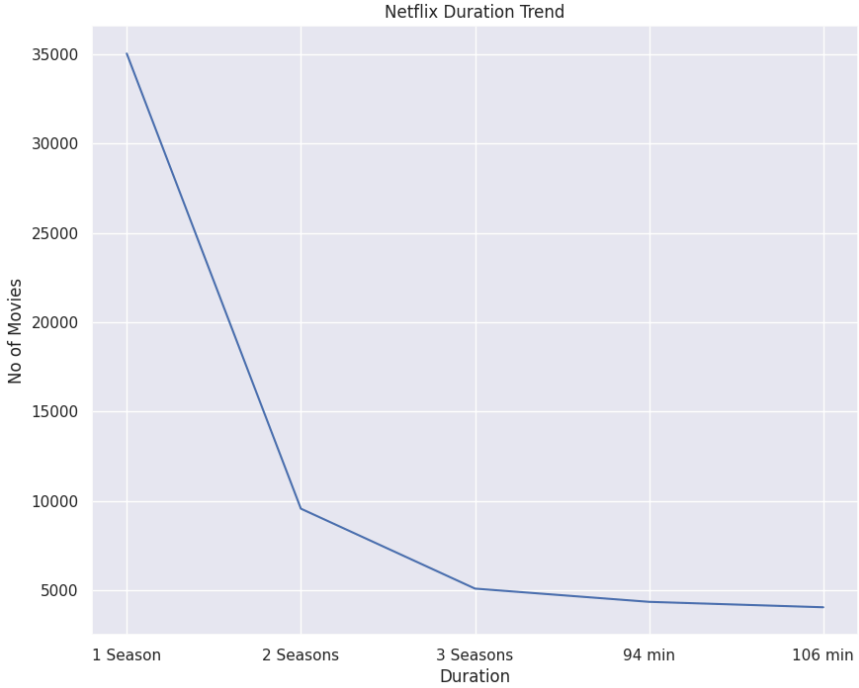
```
#to plot bar graph to make the trend more clear
plt.title("Netflix Content Duration Trend")
plt.xlabel("Duration")
plt.ylabel("NO of Movies")
x_bar=d_counts2.index
y_bar=d_counts2
plt.bar(x_bar,y_bar)
plt.xticks(rotation = 45, fontsize = 12,)
```

```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9],
 [Text(0, 0, '1 Season'),
  Text(1, 0, '2 Seasons'),
  Text(2, 0, '3 Seasons'),
  Text(3, 0, '94 min'),
  Text(4, 0, '106 min'),
  Text(5, 0, '97 min'),
  Text(6, 0, '95 min'),
  Text(7, 0, '96 min'),
  Text(8, 0, '93 min'),
  Text(9, 0, '90 min')])
```




```
plt.title("Netflix Duration Trend")
plt.xlabel("Duration")
plt.ylabel("No of Movies")
x_bar=duration_counts2.index
y_bar=duration_counts2
plt.plot(x_bar,y_bar)
```

```
[<matplotlib.lines.Line2D at 0x7d95c1dc0b50>]
```



in the above 2 plotsThe duration trend of the contents has been obtained.

Recommendation - Most of the contens can be seen having 1 season and some with 2 and 3 seasons. Rest all the Contents are of approximately same duration but with a single episode.The proportion should be checked with the engagement of audience and such contents should be focused on.

```
year_counts = df_final['release_year'].value_counts()
year_counts2=year_counts.head(10)
year_counts2
```

```
release_year
2018    24441
2019    21931
2017    20516
2020    19697
2016    18465
2015    14128
2021    11894
2014     9098
2013     7745
2012     6354
Name: count, dtype: int64
```

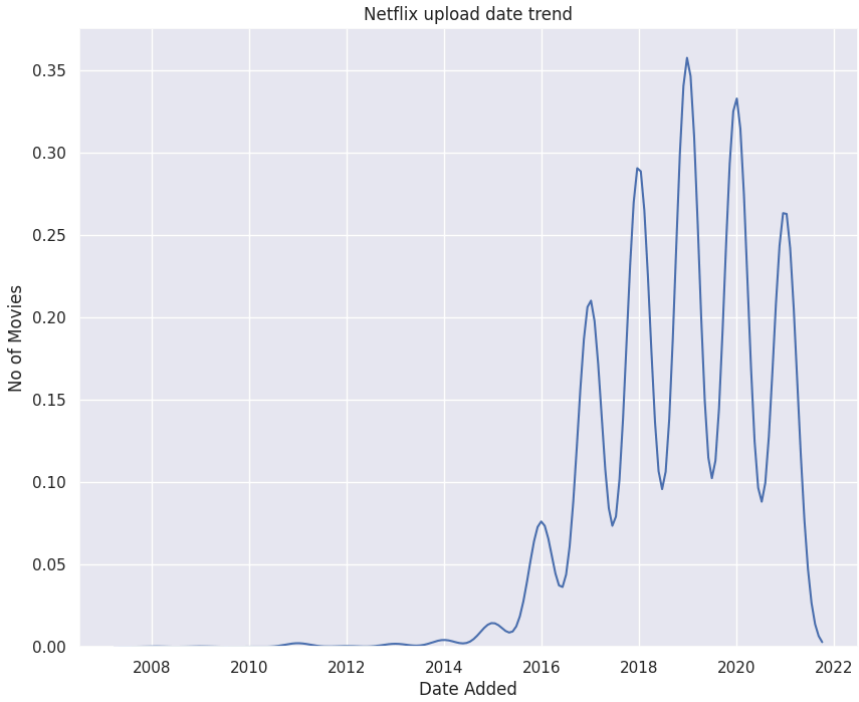
```
#to get the upload date trend for contents
type_counts = df_final['type'].value_counts()
type_counts
```

```
type
Movie      145917
TV Show     56148
Name: count, dtype: int64
```

```
plt.title("Netflix upload date trend")
plt.xlabel("Date Added")
plt.ylabel("No of Movies")
```

```
sns.kdeplot(df[ date_added ])
```

```
<Axes: title={'center': 'Netflix upload date trend'}, xlabel='Date Added',
ylabel='No of Movies'>
```

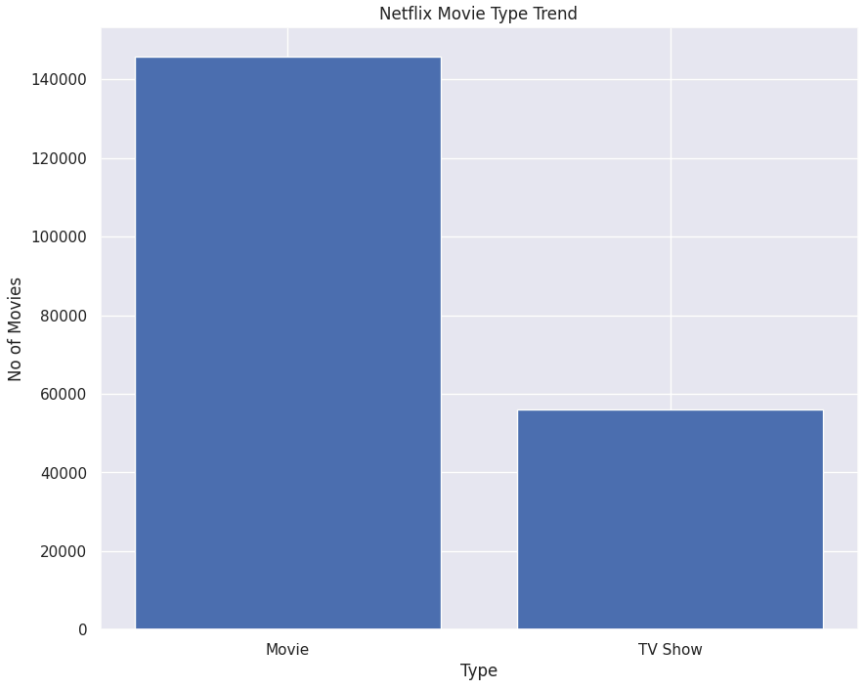


The above kdeplot is naturally in sync with the release year trend

Recommendation - The recommendation for release year trend should completely be adopted here

```
#to get the types of content on platform
plt.title("Netflix Movie Type Trend")
plt.xlabel("Type")
plt.ylabel("No of Movies")
x_bar=type_counts.index
y_bar=type_counts
plt.bar(x_bar,y_bar)
```

```
<BarContainer object of 2 artists>
```



Here we can see that there are only two kinds of contents on Netflix

Recommendation - We can see a significant difference in the two kinds of contents. The number of TV Shows should be increased in the proportion of audience engaged for it.

```
duration_counts = df_final['duration'].value_counts()
duration_counts2=duration_counts.head(5)
duration_counts2
```

```
duration
1 Season     35035
2 Seasons     9559
3 Seasons     5084
94 min        4343
106 min       4040
Name: count, dtype: int64
```

```
df_final.head()
```

	title	director	cast	country	listed_in	show_id	type	release_year	d
0	Dick Johnson Is Dead	Kirsten Johnson	Unknown Actor	United States	Documentaries	s1	Movie	2020	
1	Blood & Water	Unknown Director	Ama Qamata	South Africa	International TV Shows	s2	TV Show	2021	

```
#to get the top 10 data
top10_directors = df["director"].value_counts().index[:10]
top10_casts = df["cast"].value_counts().index[:10]
top10_titles = df["title"].value_counts().index[:10]
top10_ryears = df["release_year"].value_counts().index[:10]
top10_dadded = df["date_added"].value_counts().index[:10]
```

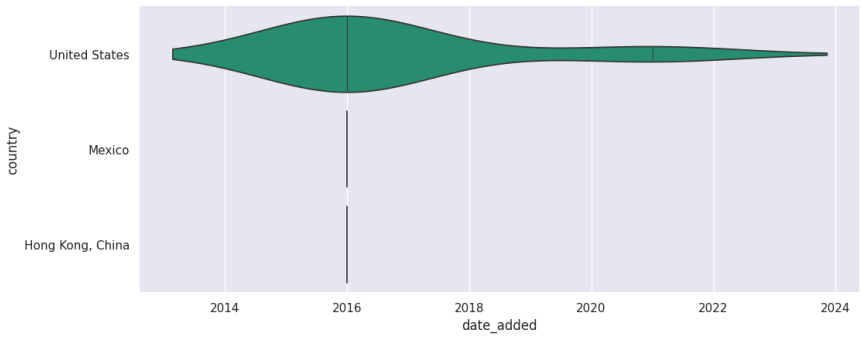
```
top10_data = df.loc[(df["title"].isin(top3_titles)) & (df["date_added"].isin(top3_dadded))]
```

```
top10_data
```

	show_id	type	title	director	cast	country	date_added	release_
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	2021.0	
5849	s5850	Movie	Team Foxcatcher	Jon Greenhalgh	NaN	United States	2016.0	
5868	s5869	Movie	Hannibal Buress: Comedy Camisado	Lance Bangs	Hannibal Buress	United States	2016.0	
5869	s5870	TV Show	Turbo FAST	NaN	Reid Scott, John Eric Bentley, Amir Talai, Phi...	United States	2016.0	
5870	s5871	TV Show	Masha's Tales	NaN	NaN	NaN	2016.0	

```
# country vs date_added

figsize = (12, 1.2 * len(top10_data['country'].unique()))
plt.figure(figsize=figsize)
sns.violinplot(top10_data, x='date_added', y='country', inner='stick', palette='Dark2')
sns.despine(top=True, right=True, bottom=True, left=True)
```

```
<ipython-input-70-935748e289d7>:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.

  sns.violinplot(top10_data, x='date_added', y='country', inner='stick', palette='Dar
```

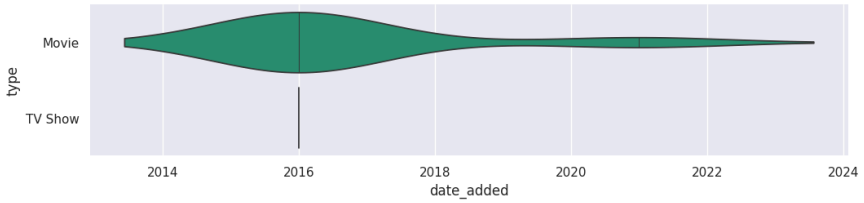


Here we can see that most of the contents were added in US highest between 2014 and 2018.

Recommendation - The scope of contents uploading should be increased among other countries as well using local contents and global contents in local languages of the respective countries

```
# type vs date_added

figsize = (12, 1.2 * len(top10_data['type'].unique()))
plt.figure(figsize=figsize)
sns.violinplot(top10_data, x='date_added', y='type', inner='stick', palette='Dark2')
sns.despine(top=True, right=True, bottom=True, left=True)
```

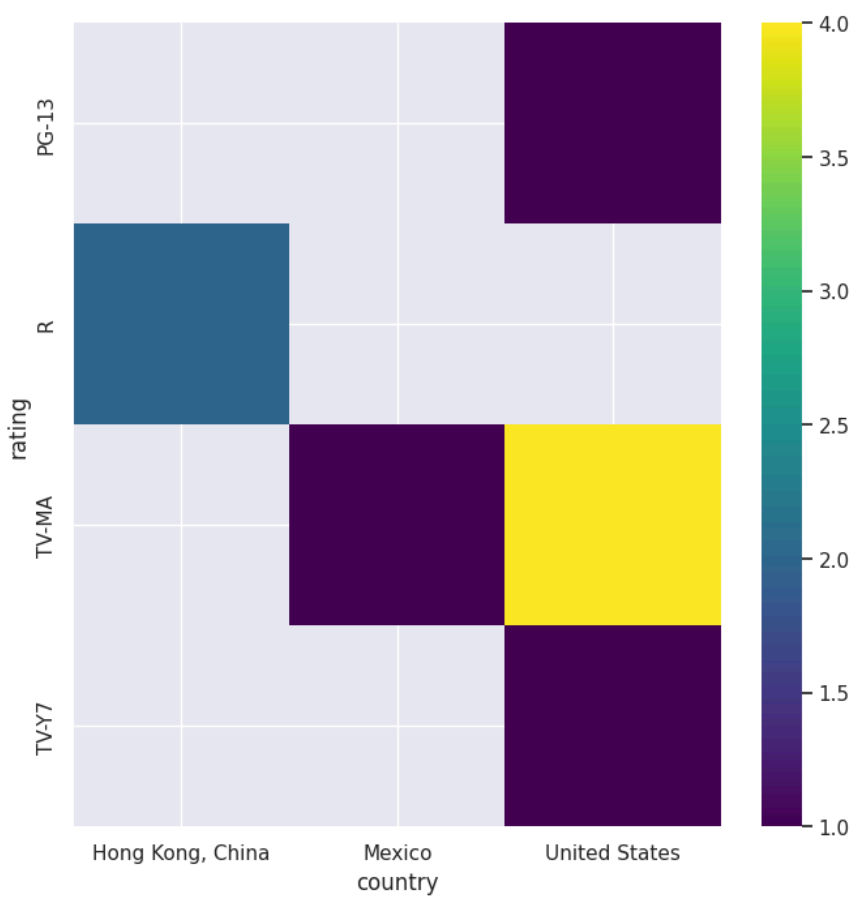
```
<ipython-input-71-24d47eec17da>:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.

  sns.violinplot(top10_data, x='date_added', y='type', inner='stick', palette='Dark2'
```



Same time trend can be seen here as well.

Recommendation - The above recommendation should be adapted here as well

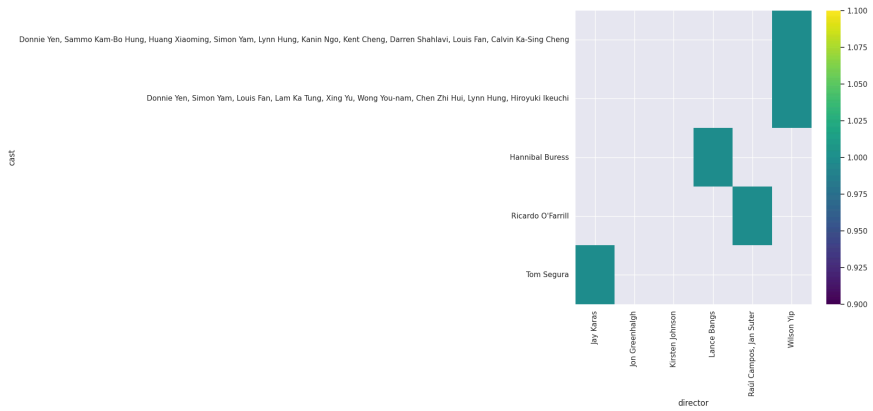
```
#country vs rating

plt.subplots(figsize=(8, 8))
df_2dhist = pd.DataFrame({
    x_label: grp['rating'].value_counts()
    for x_label, grp in top10_data.groupby('country')
})
sns.heatmap(df_2dhist, cmap='viridis')
plt.xlabel('country')
_ = plt.ylabel('rating')
```



Country wise rating trend can be seen here.US has low rating for TV-Y7 category and medium rating for TV-MA category and highest for PG-13 category.Hong Kong has highest rating for R category.

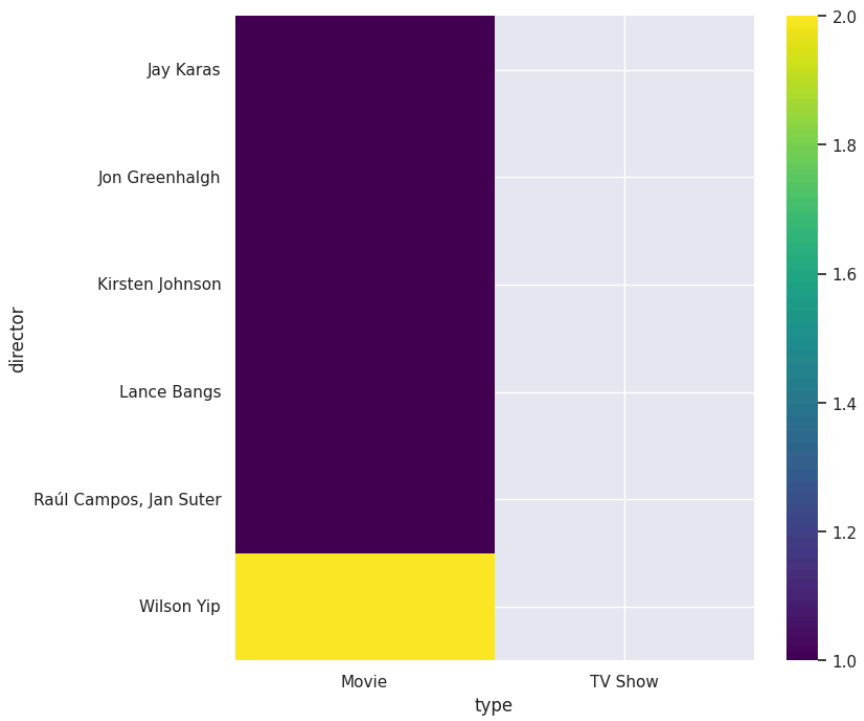
Recommendation - Considering ratings country wise and catgory wise, content quality needs to be improved.

```
# director vs cast

plt.subplots(figsize=(8, 8))
df_2dhist = pd.DataFrame({
    x_label: grp['cast'].value_counts()
    for x_label, grp in top10_data.groupby('director')
})
sns.heatmap(df_2dhist, cmap='viridis')
plt.xlabel('director')
_ = plt.ylabel('cast')
```



The heatmap shows the combination of cast and directors who have worked more together.

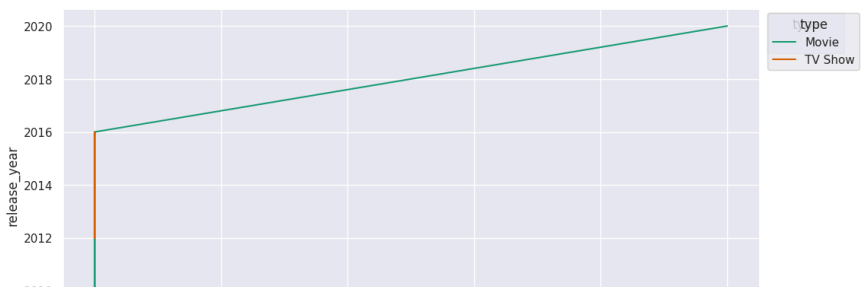
```
# type vs director

plt.subplots(figsize=(8, 8))
df_2dhist = pd.DataFrame({
    x_label: grp['director'].value_counts()
    for x_label, grp in top10_data.groupby('type')
})
sns.heatmap(df_2dhist, cmap='viridis')
plt.xlabel('type')
_ = plt.ylabel('director')
```



This plot shows what kind of content has been made more by the directors.

Recommendation - The director having expertise in particular genre should be involved more in the same genre projects.

```
#date_added vs release_year

def _plot_series(series, series_name, series_index=0):
  palette = list(sns.palettes.mpl_palette('Dark2'))
  xs = series['date_added']
  ys = series['release_year']

  plt.plot(xs, ys, label=series_name, color=palette[series_index % len(palette)])

fig, ax = plt.subplots(figsize=(10, 5.2), layout='constrained')
df_sorted = top10_data.sort_values('date_added', ascending=True)
for i, (series_name, series) in enumerate(df_sorted.groupby('type')):
  _plot_series(series, series_name, i)
  fig.legend(title='type', bbox_to_anchor=(1, 1), loc='upper left')
sns.despine(fig=fig, ax=ax)
plt.xlabel('date_added')
_ = plt.ylabel('release_year')
```



The graph shows variation in the upload date and release date for a content.We can see that most contents were added and released during 2016-2020 for both series and TV shows.

Recommendation - same and consistent uploading and release frequency should be ensured.

```
# top directors
```

```
top10_data.groupby('director').size().plot(kind='barh', color=sns.palettes.mpl_palette('Dark2'))
plt.gca().spines[['top', 'right',]].set_visible(False)
```