Mayank Agrawal - DSML Nov'23 TTS Evening batch

Scaler Case study - Instructions

Mindset

- Evaluation will be kept lenient, so make sure you attempt this case study. Read the question carefully and try to understand what exactly is being asked. Brainstorm a little. If you're getting an error, remember that Google is your best friend.
- You can watch the lecture recordings or go through your lecture notes once again if you feel like you're getting confused over some specific topics. Discuss your problems with your peers. Make use of the Slack channel and WhatsApp group.
- Only if you think that there's a major issue, you can reach out to your Instructor via Slack or Email. There is no right or wrong answer. We have to get used to dealing with uncertainty in business. This is exactly the skill we want to develop.

About NETFLIX

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.

Business Problem Analyze the data and generate insights that could help Netflix in deciding which type of shows/movies to produce and how they can grow the business in different countries

Dataset

Link: Netflix Data

(After clicking on the above link, you can download the files by right-clicking on the page and clicking on "Save As", then naming the file as per your wish, with .csv as the extension.)

The dataset provided to you consists of a list of all the TV shows/movies available on Netflix:

- Show_id: Unique ID for every Movie / Tv Show
- Type: Identifier - A Movie or TV Show

- Title: Title of the Movie / Tv Show
- Director: Director of the Movie
- Cast: Actors involved in the movie/show
- Country: Country where the movie/show was produced
- Date_added: Date it was added on Netflix
- Release_year: Actual Release year of the movie/show
- Rating: TV Rating of the movie/show
- Duration: Total Duration - in minutes or number of seasons
- Listed_in: Genre
- Description: The summary description

Hints

1. The exploration should have a goal. As you explore the data, keep in mind that you want to answer which type of shows to produce and how to grow the business.
2. Ensure each recommendation is backed by data. The company is looking for data-driven insights, not personal opinions or anecdotes.
3. Assume that you are presenting your findings to business executives who have only a basic understanding of data science. Avoid unnecessary technical jargon.
4. Start by exploring a few questions: What type of content is available in different countries?
    - How has the number of movies released per year changed over the last 20-30 years?
    - Comparison of tv shows vs. movies.
    - What is the best time to launch a TV show?
    - Analysis of actors/directors of different types of shows/movies.
    - Does Netflix has more focus on TV Shows than movies in recent years
    - Understanding what content is available in different countries

Evaluation Criteria (100 Points):

1. Defining Problem Statement and Analysing basic metrics (10 Points)

2. Observations on the shape of data, data types of all the attributes, conversion of categorical attributes to 'category' (If required), missing value detection, statistical summary (10 Points)

3. Non-Graphical Analysis: Value counts and unique attributes (10 Points)

4. Visual Analysis - Univariate, Bivariate after pre-processing of the data

Note: Pre-processing involves unnesting of the data in columns like Actor, Director, Country

4.1 For continuous variable(s): Distplot, countplot, histogram for univariate analysis (10 Points)

4.2 For categorical variable(s): Boxplot (10 Points)

4.3 For correlation: Heatmaps, Pairplots (10 Points)

1. Missing Value & Outlier check (Treatment optional) (10 Points)

2. Insights based on Non-Graphical and Visual Analysis (10 Points)

   6.1 Comments on the range of attributes

   6.2 Comments on the distribution of the variables and relationship between them

   6.3 Comments for each univariate and bivariate plot

3. Business Insights (10 Points) - Should include patterns observed in the data along with what you can infer from it

4. Recommendations (10 Points) - Actionable items for business. No technical jargon. No complications. Simple action items that everyone can understand

Submission Process:

1. Type your insights and recommendations in the rich-text editor.
2. Convert your jupyter notebook into PDF (Save as PDF using Chrome browser's Print command), upload it in your Google Drive (set the permission to allow public access), and paste that link in the text editor.
3. Alternatively, you can directly submit your PDF on the portal.
4. Optionally, you may add images/graphs in the text editor by taking screenshots or saving matplotlib graphs using plt.savefig(...).
5. After submitting, you will not be allowed to edit your submission.

Solution

Basic Data Exploration

Importing the libraries and checking the data

In [ ]:
```
#importing the libraries useful for the analysis
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [ ]:
```
!gdown 1vngLDiVgY3jFoWwUriLKXW61ANd8Xm4p
```

```
Downloading...
From: https://drive.google.com/uc?id=1vngLDiVgY3jFoWwUriLKXW61ANd8Xm4p
To: /content/d13_netflix.csv
100% 3.40M/3.40M [00:00<00:00, 9.17MB/s]
```

In [ ]:
```
#uploading the data as dataframe and checking on few first lines
df = pd.read_csv('d13_netflix.csv')
```

Understanding data - data tyep, nested data

In [ ]:
```
#checking on sample of data , to check variety of the data available in diffrent columns
df.sample(5)
```

Out[ ]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
4242	s4243	Movie	Taylor Swift reputation Stadium Tour	Paul Dugdale	Taylor Swift	NaN	December 31, 2018	2018	TV-PG	125 min	Music & Musicals	Taylor Swift takes the stage in Dallas for the...
8308	s8309	Movie	The Force	Peter Nicks	NaN	United States	January 29, 2018	2017	TV-MA	92 min	Documentaries	This documentary follows the

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
												police of Oakland...
1054	s1055	Movie	Wave of Cinema: One Day We'll Talk About Today	Adriano Rudiman	Isyana Sarasvati, Kunto Aji, Sisir Tanah, Chik...	NaN	April 15, 2021	2020	TV-14	71 min	Documentaries, International Movies, Music & M...	Delve into a musical concert and relive moment...
6804	s6805	Movie	Friday the 13th	Marcus Nispel	Jared Padalecki, Danielle Panabaker, Amanda Ri...	United States	January 1, 2020	2009	R	97 min	Horror Movies	A group of teenage camp counselors raises the ...
1861	s1862	Movie	Dil	Indra Kumar	Aamir Khan, Madhuri Dixit, Saeed Jaffrey, Deve...	India	October 12, 2020	1990	TV-14	165 min	Comedies, Dramas, International Movies	A miser's scheme to set his son up with a mill...

- There are multiple entries in director, cast, country and listed_in columns, we need to unnest this data before start using this

In [ ]:
```
# checking the length of the data
len(df), df.shape
```

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

In [ ]:
```
#checking on the different datatypes available in columsn
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
```

- Above we can see that show_id is unique primary key for show_id irrespective of the type of show.
- Total unique line items are 8807.
- There are some missing values in columns like director, cast, country, date_added, rating and duration, later we need to handle these missing values through imputaion by static or some other target imputation like grouped mode for that column.
- date_added column is object type, we might need to convert it to data_time columns based on the requirement later.
- There is only one column which is integer, other are all object type columns , so need to take care of column data type in the analysis.

In [ ]:
```
#let's check for unique values in each column, this will give us an idea about the variety of the data in columns
#might be useful later for the grouping to understand the categorical data
df.nunique()
```

Out[ ]:
```
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
```

Checking on null values

In [ ]:
```
#we need the sum of null values in each column
df.isnull().sum()
```

Out[ ]:
```
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 [ ]:
```
#we can also calculate % wise null values in each column
round(df.isnull().sum()*100/len(df),2)
```

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

- highest missing values are in director, cast, country, which might need imputation later based on the requirement.

Checking on wrong data entries

In [ ]:
```
#check the rating columnn once for values in there
df['rating'].value_counts()
```

Out[ ]:
```
rating
TV-MA       3207
TV-14       2160
TV-PG        863
R            799
PG-13        490
TV-Y7        334
TV-Y         307
PG           287
TV-G         220
NR            80
G             41
TV-Y7-FV       6
NC-17          3
UR             3
74 min         1
84 min         1
66 min         1
Name: count, dtype: int64
```

- there are 3 entries with min in rating column

In [ ]:
```
#check once for duration also
df['duration'].value_counts()
```

Out[ ]:
```
duration
1 Season     1793
2 Seasons     425
3 Seasons     199
90 min        152
94 min        146
             ...
16 min          1
186 min         1
193 min         1
189 min         1
191 min         1
Name: count, Length: 220, dtype: int64
```

- Similarly there are season entered in duration

Checking on the columns with nested data

In [ ]:
```
#lets also check for few other columns
df['director'].value_counts()
```

Out[ ]:
```
director
Rajiv Chilaka                     19
Raúl Campos, Jan Suter            18
Marcus Raboy                      16
Suhas Kadav                       16
Jay Karas                         14
                                  ..
Raymie Muzquiz, Stu Livingston     1
Joe Menendez                       1
Eric Bross                         1
Will Eisenberg                     1
Mozez Singh                        1
Name: count, Length: 4528, dtype: int64
```

In [ ]:
```
#lets also check for few other columns
df['cast'].value_counts()
```

Out[ ]:
```
cast
David Attenborough
19
Vatsal Dubey, Julie Tejwani, Rupa Bhimani, Jigna Bhardwaj, Rajesh Kava, Mousam, Swapnil
14
Samuel West
10
Jeff Dunham
7
David Spade, London Hughes, Fortune Feimster
6

..
Michael Peña, Diego Luna, Tenoch Huerta, Joaquin Cosio, José María Yazpik, Matt Letscher, Alyssa Diaz
1
Nick Lachey, Vanessa Lachey
1
Takeru Sato, Kasumi Arimura, Haru, Kentaro Sakaguchi, Takayuki Yamada, Kendo Kobayashi, Ken Yasuda, Arata Furuta, Suzuk
i Matsuo, Koichi Yamadera, Arata Iura, Chikako Kaku, Kotaro Yoshida     1
Toyin Abraham, Sambasa Nzeribe, Chioma Chukwuka Akpotha, Chioma Omeruah, Chiwetalu Agu, Dele Odule, Femi Adebayo, Bayra
```

```
y McNwizu, Biodun Stephen                                                                                    1
Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanana, Manish Chaudhary, Meghna Malik, Malkeet Rauni, Anita Shabdish, Chittar
anjan Tripathy                                                                                               1
Name: count, Length: 7692, dtype: int64
```

- There are multiple director and cast name is there for one movie in same cell, let's try to split it in multiple rows for same movie.

Unnest the data

Cast

- Lets try to split the values from the cast column.

In [ ]:
```
#take the cast column and splitting the values in it by comma sperator and saving as list
#splitted > saved in list > convereted in data frame with split value in differet columns
#title as index
df_cast_r = pd.DataFrame(df['cast'].apply(lambda x: str(x).split(',')).tolist(),index = df['title'])
df_cast_r
```

Out[ ]:

title	0	1	2	3	4	5	6	7	8	9	...	40	41	42
Dick Johnson Is Dead	nan	None	None	None	None	None	None	None	None	None	...	None	None	None
Blood & Water	Ama Qamata	Khosi Ngema	Gail Mabalane	Thabang Molaba	Dillon Windvogel	Natasha Thahane	Arno Greeff	Xolile Tshabalala	Getmore Sithole	Cindy Mahlangu	...	None	None	None
Ganglands	Sami Bouajila	Tracy Gotoas	Samuel Jouy	Nabiha Akkari	Sofia Lesaffre	Salim Kechiouche	Noureddine Farihi	Geert Van Rampelberg	Bakary Diombera	None	...	None	None	None
Jailbirds New Orleans	nan	None	None	None	None	None	None	None	None	None	...	None	None	None
Kota Factory	Mayur More	Jitendra Kumar	Ranjan Raj	Alam Khan	Ahsaas Channa	Revathi Pillai	Urvi Singh	Arun Kumar	None	None	...	None	None	None
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
Zodiac	Mark Ruffalo	Jake Gyllenhaal	Robert Downey Jr.	Anthony Edwards	Brian Cox	Elias Koteas	Donal Logue	John Carroll Lynch	Dermot Mulroney	Chloë Sevigny	...	None	None	None

title	0	1	2	3	4	5	6	7	8	9	...	40	41	4
Zombie Dumb	nan	None	None	None	None	None	None	None	None	None	...	None	None	Non
Zombieland	Jesse Eisenberg	Woody Harrelson	Emma Stone	Abigail Breslin	Amber Heard	Bill Murray	Derek Graf	None	None	None	...	None	None	Non
Zoom	Tim Allen	Courteney Cox	Chevy Chase	Kate Mara	Ryan Newman	Michael Cassidy	Spencer Breslin	Rip Torn	Kevin Zegers	None	...	None	None	Non
Zubaan	Vicky Kaushal	Sarah-Jane Dias	Raaghav Chanana	Manish Chaudhary	Meghna Malik	Malkeet Rauni	Anita Shabdish	Chittaranjan Tripathy	None	None	...	None	None	Non

8807 rows × 50 columns

In [ ]:
```
#let's stack the data in columns to rows
df_cast = df_cast_r.stack().reset_index()
df_cast
```

Out[ ]:

	title	level_1	0
0	Dick Johnson Is Dead	0	nan
1	Blood & Water	0	Ama Qamata
2	Blood & Water	1	Khosi Ngema
3	Blood & Water	2	Gail Mabalane
4	Blood & Water	3	Thabang Molaba
...	...	...	...
64946	Zubaan	3	Manish Chaudhary
64947	Zubaan	4	Meghna Malik
64948	Zubaan	5	Malkeet Rauni
64949	Zubaan	6	Anita Shabdish
64950	Zubaan	7	Chittaranjan Tripathy

64951 rows × 3 columns

In [ ]:
```
#need to drop the level_1 column and rename the column '0' to cast
df_cast.drop(['level_1'], axis=1, inplace = True)
df_cast.rename(columns = {0: 'cast'}, inplace = True)
df_cast.head()
```

Out[ ]:

	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

In [ ]:
```
#in summary the whole code

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()
```

Out[ ]:

	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

director

- Now let's try to split the values from the director column

In [ ]:
```
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()
```

Out[ ]:

	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

Country

In [ ]:
```
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()
```

Out[ ]:

	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

listed_in

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

Out[ ]:

	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 unnested data

merging the director with cast , both unnested

In [ ]:
```
df_new = df_director.merge(df_cast, how = 'inner', on= 'title')
df_new.head()
```

Out[ ]:

	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

merging country unnested columns

In [ ]:
```
df_new = df_new.merge(df_country, how = 'inner', on= 'title')
df_new.head()
```

Out[ ]:

	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

merging the listed_in unnested column

In [ ]:
```
df_new = df_new.merge(df_listedin, how = 'inner', on= 'title')
df_new.head()
```

Out[ ]:

	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

Merging the the merged data with the original dataframe

In [ ]:
```
df.columns
```

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

In [ ]:
```
# lets merge all other columns in dataframe with he unnested columns
df_final = df_new.merge(df[['show_id', 'type', 'title','date_added',
       'release_year', 'rating', 'duration', 'description']], how = 'inner', on = 'title')
df_final.sample(10)
```

Out[ ]:

	title	director	cast	country	listed_in	show_id	type	date_added	release_year	rating	duration	description
166834	Little Dragon Maiden	Hua Shan	Chen Kuan-tai	Hong Kong	Romantic Movies	s7312	Movie	August 1, 2018	1983	TV-14	92 min	Seeking to improve his combat skills, a young ...
98752	And Breathe Normally	Ísold Uggadóttir	Babetida Sadjo	Belgium	Dramas	s4213	Movie	January 4, 2019	2018	TV-14	102 min	An Icelandic single mom struggling with povert...
177066	Power Rangers Ninja Storm	nan	Bruce Hopkins	New Zealand	Kids' TV	s7770	TV Show	January 1, 2016	2003	TV-Y7	1 Season	When the elite warriors from the Wind Ninja Ac...
113490	Black Crows	nan	Maram Al Bloushi	Syria	TV Dramas	s4904	TV Show	April 30, 2018	2017	TV-14	1 Season	This drama portrays women and kids living unde...
109942	Chillar Party	Nitesh Tiwari	Sanath Menon	India	Comedies	s4721	Movie	August 2, 2018	2011	TV-PG	127 min	Eight feisty boys befriend young drifter Fatka...
64796	The Death of Stalin	Armando Iannucci	Rupert Friend	United States	Independent Movies	s2713	Movie	April 3, 2020	2017	R	107 min	The death of Russian dictator Joseph Stalin th...
163123	Kahlil Gibran's The Prophet	Joan C. Gratz	John Krasinski	Qatar	Dramas	s7165	Movie	October 1, 2017	2014	PG	85 min	A troubled young girl and her mother find sola...
107418	Paradise Lost	Monique Gardenberg	Malu Galli	Brazil	Dramas	s4618	Movie	September 29, 2018	2018	TV-MA	111 min	A cop moonlights as the bodyguard for a young ...
169775	Merlí	nan	Marta Domingo	Spain	Teen TV Shows	s7448	TV Show	December 1, 2016	2015	TV-MA	1 Season	An unconventional high school

	title	director	cast	country	listed_in	show_id	type	date_added	release_year	rating	duration	description
												philosophy teach...
159242	Hotel Beau Séjour	nan	Roel Vanderstukken	Belgium	International TV Shows	s7007	TV Show	March 16, 2017	2017	TV-14	1 Season	Caught in an afterlife limbo, teenage Kato inv...

Handling the missing values

In [ ]:
```
# checking the shape of df_final, check the number of rows increased exponentially due to unnesting and merging
df_final.shape
```

Out[ ]:
```
(202065, 12)
```

In [ ]:
```
df_final.isnull().sum()
```

Out[ ]:
```
title            0
director         0
cast             0
country          0
listed_in        0
show_id          0
type             0
date_added     158
release_year     0
rating          67
duration         3
description      0
dtype: int64
```

- there is nan present in the director and other columns, but as that is string isnull won't be able to catch that
- replace nan value of Actor and Director with the 'Unknown Actor' and 'Unknown Director' respectively

Cast and Director

In [ ]:
```
df_final['cast'].replace(['nan'], ['Unknown Actor'], inplace = True)
df_final['director'].replace(['nan'], ['Unknown Director'], inplace = True)
```

In [ ]:
```
#also replacing the string 'nan' with the country nan
df_final['country'].replace(['nan'],[np.nan],inplace=True)
df_final.head()
```

Out[ ]:

	title	director	cast	country	listed_in	show_id	type	date_added	release_year	rating	duration	description
0	Dick Johnson Is Dead	Kirsten Johnson	Unknown Actor	United States	Documentaries	s1	Movie	September 25, 2021	2020	PG-13	90 min	As her father nears the end of his life, filmm...
1	Blood & Water	Unknown Director	Ama Qamata	South Africa	International TV Shows	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...
2	Blood & Water	Unknown Director	Ama Qamata	South Africa	TV Dramas	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...
3	Blood & Water	Unknown Director	Ama Qamata	South Africa	TV Mysteries	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...
4	Blood & Water	Unknown Director	Khosi Ngema	South Africa	International TV Shows	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...

In [ ]:
```
df_final.isnull().sum()
```

Out[ ]:
```
title             0
director          0
cast              0
country       11897
listed_in         0
show_id           0
type              0
date_added      158
release_year      0
rating           67
duration          3
description       0
dtype: int64
```

Duration

In [ ]:
```
df_final[df_final['duration'].isnull()]
```

Out[ ]:

	title	director	cast	country	listed_in	show_id	type	date_added	release_year	rating	duration	description
126582	Louis C.K. 2017	Louis C.K.	Louis C.K.	United States	Movies	s5542	Movie	April 4, 2017	2017	74 min	NaN	Louis C.K. muses on religion, eternal love, gi...
131648	Louis C.K.: Hilarious	Louis C.K.	Louis C.K.	United States	Movies	s5795	Movie	September 16, 2016	2010	84 min	NaN	Emmy-winning comedy writer Louis C.K. brings h...
131782	Louis C.K.: Live at the Comedy Store	Louis C.K.	Louis C.K.	United States	Movies	s5814	Movie	August 15, 2016	2015	66 min	NaN	The comic puts his trademark hilarious/thought...

- duration column for these 3 rows is NaN because, its values are mistakenly entered into the rating column.
- for these columns we can copy the duration from the rating column and paste it there
- first creating a mask by accessing the duration equal to null and then copy pasting rating value in the duration

In [ ]:
```
df_final.loc[df_final['duration'].isnull(), 'duration'] = df_final.loc[df_final['duration'].isnull(),'rating']
```

In [ ]:
```
df_final.isnull().sum()
```

Out[ ]:
```
title              0
director           0
cast               0
country        11897
listed_in          0
show_id            0
type               0
date_added       158
release_year       0
rating            67
duration           0
```

```
description         0
dtype: int64
```

- see now the duration null values are not there

Rating

In [ ]:
```
df_final['rating'].value_counts()
```

Out[ ]:
```
rating
TV-MA       73915
TV-14       43957
R           25860
PG-13       16246
TV-PG       14926
PG          10919
TV-Y7        6304
TV-Y         3665
TV-G         2779
NR           1573
G            1530
NC-17         149
TV-Y7-FV       86
UR             86
74 min          1
84 min          1
66 min          1
Name: count, dtype: int64
```

- Rating also contains the mins as values which is incorrect, we can replace it with NR, means No rating

In [ ]:
```
#let's convert the 'Min' in ratings columns as NR and NaN values also seperatly as NR
df_final.loc[df_final['rating'].str.contains('min', na = False), ['rating']] = 'NR'
df_final['rating'].fillna('NR', inplace = True)
df_final['rating'].value_counts()
```

Out[ ]:
```
rating
TV-MA       73915
TV-14       43957
R           25860
PG-13       16246
```

```
TV-PG       14926
PG          10919
TV-Y7        6304
TV-Y         3665
TV-G         2779
NR           1643
G            1530
NC-17         149
TV-Y7-FV       86
UR             86
Name: count, dtype: int64
```

Country

- replace with mode of director and Actor respectively for country

In [ ]:
```
df_final.isnull().sum()
```

Out[ ]:
```
title              0
director           0
cast               0
country        11897
listed_in          0
show_id            0
type               0
date_added       158
release_year       0
rating             0
duration           0
description        0
dtype: int64
```

- country is having Nulls , which we can imputate based on director and then with actor if director has only appeared once, and even if it is not filled we will fill it with 'Unknown Country'
- based on google research i found that transform() method to apply the mode function to each group separately and then fill the NaN values in the original DataFrame.

In [ ]:
```
#df_final['country_mode'] = df_final.groupby(['director'])['country'].agg(lambda x: x.mode().iloc[0])
```

In [ ]:
```
#df_final[df_final['country'].isnull()][['country', 'country_mode']]
```

In [ ]:
```
#replacing the country NaN value with country mode value
#df_final.loc[df_final['country'].isnull(), 'country'] = df_final.loc[df_final['country'].isnull(), 'country_mode']
```

In [ ]:
```
df_final.isnull().sum()
```

Out[ ]:
```
title              0
director           0
cast               0
country        11897
listed_in          0
show_id            0
type               0
date_added       158
release_year       0
rating             0
duration           0
description        0
dtype: int64
```

In [ ]:
```
df_final['country'].fillna('Unknown Country',inplace=True)
df_final.isnull().sum()
```

Out[ ]:
```
title            0
director         0
cast             0
country          0
listed_in        0
show_id          0
type             0
date_added     158
release_year     0
rating           0
duration         0
description      0
dtype: int64
```

DateAdded

- date added is to be imputed based on the release year

Modifying the column values

In [ ]:
```
df_final.sample(5)
```

Out[ ]:

	title	director	cast	country	listed_in	show_id	type	date_added	release_year	rating	duration	description
126185	The 101-Year-Old Man Who Skipped Out on the Bi...	Felix Herngren	Caroline Boulton	Sweden	Comedies	s5518	Movie	April 25, 2017	2016	TV-MA	108 min	In need of money, an eccentric ex-spy and his ...
140568	Bad Match	David Chirchirillo	Zedrick Restauro	Singapore	Thrillers	s6215	Movie	March 11, 2018	2017	TV-MA	83 min	A player who uses the internet to facilitate h...
128637	Jim Gaffigan: Cinco	Jeannie Gaffigan	Jim Gaffigan	United States	Stand-Up Comedy	s5640	Movie	January 10, 2017	2017	TV-14	74 min	America's king of clean comedy delivers wicked...
198160	Tucker and Dale vs. Evil	Eli Craig	Philip Granger	United Kingdom	Horror Movies	s8639	Movie	March 29, 2019	2010	R	89 min	Expecting to relax at their "vacation" cabin, ...
29654	Çarsi Pazar	Muharrem Gülmez	Esin Eden	Turkey	Comedies	s1209	Movie	March 12, 2021	2015	TV-MA	97 min	The slacker owner of a public bath house ralli...

Duration

- Let's try to replace the min as duration to bins values
- for the first we need to remove 'min' from the data
- create a copy of column and create bins for min as duration , except seasons

- replace min with bin values for duration

In [ ]:
```
df_final['duration'].value_counts()
```

Out[ ]:
```
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 [ ]:
```
# replacing min from the duration
df_final['duration'] = df_final['duration'].str.replace('min','')
df_final.head()
```

Out[ ]:

	title	director	cast	country	listed_in	show_id	type	date_added	release_year	rating	duration	description
0	Dick Johnson Is Dead	Kirsten Johnson	Unknown Actor	United States	Documentaries	s1	Movie	September 25, 2021	2020	PG-13	90	As her father nears the end of his life, filmm...
1	Blood & Water	Unknown Director	Ama Qamata	South Africa	International TV Shows	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...
2	Blood & Water	Unknown Director	Ama Qamata	South Africa	TV Dramas	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...
3	Blood & Water	Unknown Director	Ama Qamata	South Africa	TV Mysteries	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...
4	Blood & Water	Unknown Director	Khosi Ngema	South Africa	International TV Shows	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...

In [ ]:
```
df_final['duration1']= df_final['duration'].copy()
df_final['duration1'].describe()
```

Out[ ]:
```
count       202065
unique         220
top       1 Season
freq         35035
Name: duration1, dtype: object
```

In [ ]:
```
#currently in the duration column , there is text as season is also present
df_final.loc[df_final['duration1'].str.contains('Season'), 'duration1'] = 0
```

In [ ]:
```
df_final['duration1'].describe()
```

Out[ ]:
```
count     202065
unique       206
top            0
freq       56148
Name: duration1, dtype: int64
```

In [ ]:
```
df_final['duration1'] = df_final['duration1'].astype('int')
```

In [ ]:
```
df_final['duration1'].describe()
```

Out[ ]:
```
count    202065.000000
mean         77.152065
std          52.262613
min           0.000000
25%           0.000000
50%          95.000000
75%         112.000000
max         312.000000
Name: duration1, dtype: float64
```

In [ ]:
```
#trying to create a distribution plot with the help of the duration1 column which now have
#the time duration of the shows in int without the season
#this will give us some idea about what bins to create
sns.distplot(df_final['duration1'], hist=True, kde=True,
bins=int(36), color = 'darkblue',
hist_kws={'edgecolor':'black'},
kde_kws={'linewidth': 4})
plt.show()
```

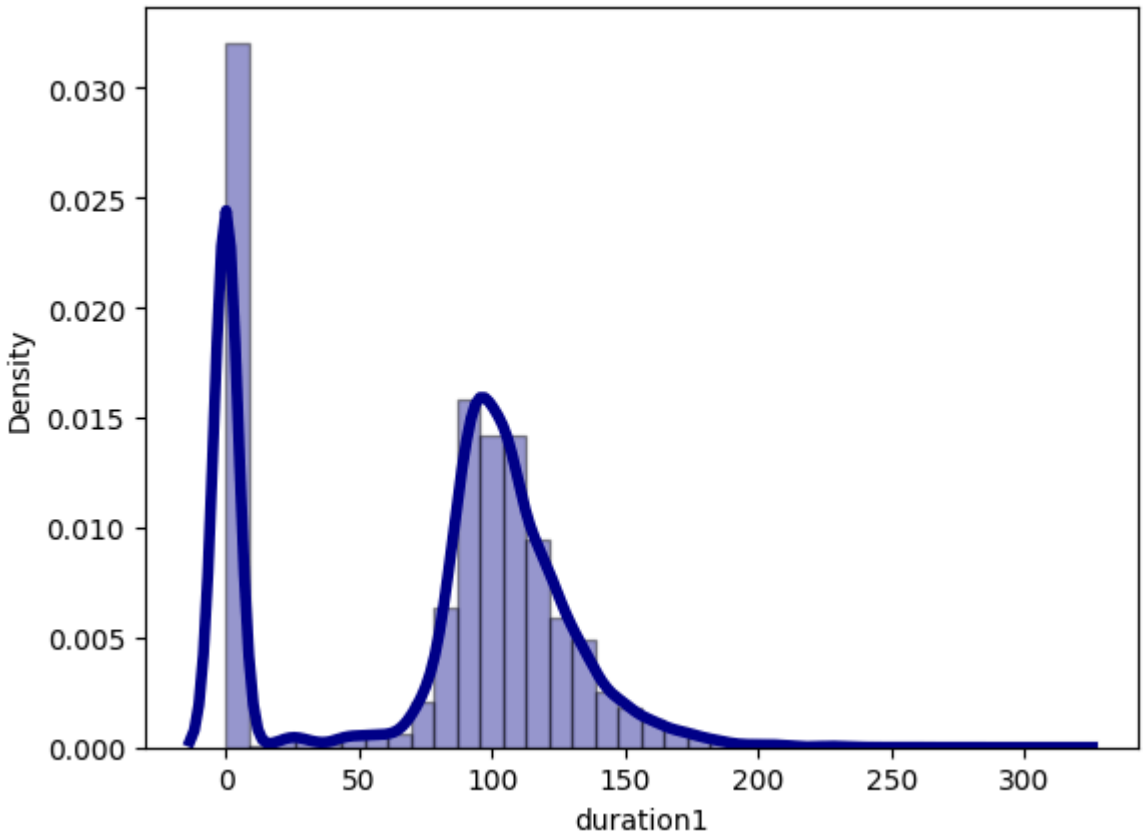
```
<ipython-input-200-4d88176915e4>:4: UserWarning:

`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

  sns.distplot(df_final['duration1'], hist=True, kde=True,
```



In [ ]:

```
bins = [-1, 1, 50, 80, 100, 150, 200, 315]
# labels will be one less than bins
labels = ['<1', '1-50','50-80', '80-100', '100-150', '150-200', '200-315']

df_final['duration1'] = pd.cut(df_final['duration1'],bins = bins, labels = labels)
df_final.head()
```

Out[ ]:

	title	director	cast	country	listed_in	show_id	type	date_added	release_year	rating	duration	description	duration1
0	Dick Johnson Is Dead	Kirsten Johnson	Unknown Actor	United States	Documentaries	s1	Movie	September 25, 2021	2020	PG-13	90	As her father nears the end of his life, filmm...	80-100

	title	director	cast	country	listed_in	show_id	type	date_added	release_year	rating	duration	description	duration1
1	Blood & Water	Unknown Director	Ama Qamata	South Africa	International TV Shows	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...	<1
2	Blood & Water	Unknown Director	Ama Qamata	South Africa	TV Dramas	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...	<1
3	Blood & Water	Unknown Director	Ama Qamata	South Africa	TV Mysteries	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...	<1
4	Blood & Water	Unknown Director	Khosi Ngema	South Africa	International TV Shows	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...	<1

In [ ]:
```
# now replace the duration with bins in duration column wherever Season is not there
# using '~' in mask to filter where Season is not there

df_final.loc[~df_final['duration'].str.contains('Season'), 'duration'] = df_final.loc[~df_final['duration'].str.contain
df_final['duration'].value_counts()
```

Out[ ]:
```
duration
100-150      75415
80-100       52992
1 Season     35035
2 Seasons     9559
50-80         7701
150-200       6737
3 Seasons     5084
```

```
1-50           2548
4 Seasons      2134
5 Seasons      1698
7 Seasons       843
6 Seasons       633
200-315         524
8 Seasons       286
9 Seasons       257
10 Seasons      220
13 Seasons      132
12 Seasons      111
15 Seasons       96
17 Seasons       30
11 Seasons       30
Name: count, dtype: int64
```

DateAdded

- coverting the date added column as date time column
- we can extract the month, date and year from this column to do the analysis.

In [ ]:
```
df_final.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 202065 entries, 0 to 202064
Data columns (total 13 columns):
 #   Column        Non-Null Count   Dtype   
---  ------        --------------   -----   
 0   title         202065 non-null  object  
 1   director      202065 non-null  object  
 2   cast          202065 non-null  object  
 3   country       202065 non-null  object  
 4   listed_in     202065 non-null  object  
 5   show_id       202065 non-null  object  
 6   type          202065 non-null  object  
 7   date_added    201907 non-null  object  
 8   release_year  202065 non-null  int64   
 9   rating        202065 non-null  object  
 10  duration      202065 non-null  object  
 11  description   202065 non-null  object  
 12  duration1     202065 non-null  category
```

```
dtypes: category(1), int64(1), object(11)
memory usage: 18.7+ MB
```

In [ ]:
```
df_final['date_added'] = pd.to_datetime(df_final['date_added'], format='%B %d, %Y', errors='coerce')
```

In [ ]:
```
#now datatype of date_added column is changed
#let's extract other columns like month, week and year

df_final['year_added'] = df_final['date_added'].dt.year
df_final['month_added'] = df_final['date_added'].dt.month
df_final['week_added'] = df_final['date_added'].dt.isocalendar().week
df_final.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 202065 entries, 0 to 202064
Data columns (total 16 columns):
 #   Column        Non-Null Count   Dtype         
---  ------        --------------   -----         
 0   title         202065 non-null  object        
 1   director      202065 non-null  object        
 2   cast          202065 non-null  object        
 3   country       202065 non-null  object        
 4   listed_in     202065 non-null  object        
 5   show_id       202065 non-null  object        
 6   type          202065 non-null  object        
 7   date_added    200319 non-null  datetime64[ns]
 8   release_year  202065 non-null  int64         
 9   rating        202065 non-null  object        
 10  duration      202065 non-null  object        
 11  description   202065 non-null  object        
 12  duration1     202065 non-null  category      
 13  year_added    200319 non-null  float64       
 14  month_added   200319 non-null  float64       
 15  week_added    200319 non-null  UInt32        
dtypes: UInt32(1), category(1), datetime64[ns](1), float64(2), int64(1), object(10)
memory usage: 22.7+ MB
```

In [ ]:
```
df_final.head()
```

Out[ ]:

	title	director	cast	country	listed_in	show_id	type	date_added	release_year	rating	duration	description	duration1	year_
0	Dick Johnson Is Dead	Kirsten Johnson	Unknown Actor	United States	Documentaries	s1	Movie	2021-09-25	2020	PG-13	80-100	As her father nears the end of his life, filmm...	80-100	
1	Blood & Water	Unknown Director	Ama Qamata	South Africa	International TV Shows	s2	TV Show	2021-09-24	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...	<1	
2	Blood & Water	Unknown Director	Ama Qamata	South Africa	TV Dramas	s2	TV Show	2021-09-24	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...	<1	
3	Blood & Water	Unknown Director	Ama Qamata	South Africa	TV Mysteries	s2	TV Show	2021-09-24	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...	<1	
4	Blood & Water	Unknown Director	Khosi Ngema	South Africa	International TV Shows	s2	TV Show	2021-09-24	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...	<1	

Title

Removing the content in brackets like Hindi, Tamil version fo same movie to make the title same

In [ ]:

```
#using the regex patter
df_final['title']=df_final['title'].str.replace(r"\(.*\)","")
df_final.head()
```

Out[ ]:

	title	director	cast	country	listed_in	show_id	type	date_added	release_year	rating	duration	description	duration1	year_
0	Dick Johnson Is Dead	Kirsten Johnson	Unknown Actor	United States	Documentaries	s1	Movie	2021-09-25	2020	PG-13	80-100	As her father nears the end of his life, filmm...	80-100	
1	Blood & Water	Unknown Director	Ama Qamata	South Africa	International TV Shows	s2	TV Show	2021-09-24	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...	<1	
2	Blood & Water	Unknown Director	Ama Qamata	South Africa	TV Dramas	s2	TV Show	2021-09-24	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...	<1	
3	Blood & Water	Unknown Director	Ama Qamata	South Africa	TV Mysteries	s2	TV Show	2021-09-24	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...	<1	
4	Blood & Water	Unknown Director	Khosi Ngema	South Africa	International TV Shows	s2	TV Show	2021-09-24	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...	<1	

EDA

Univariate analysis based on Title

Title based on Genre

In [ ]:

```
df_genre = df_final.groupby('listed_in')['title'].nunique().sort_values(ascending = False)
df_genre
```

Out[ ]:
```
listed_in
 International Movies        2624
Dramas                       1600
Comedies                     1210
Action & Adventure            859
Documentaries                 829
                             ...
Romantic Movies                 3
Spanish-Language TV Shows       2
TV Sci-Fi & Fantasy             1
LGBTQ Movies                    1
Sports Movies                   1
Name: title, Length: 73, dtype: int64
```

-Above is series, but we need the dataframe to create the plots easily and access the particlur columns.

- we might need to write [['title']] as dataframe and add by = ['title] in the sort value to make it as dataframe

In [ ]:
```
#we have to use the nunique only because it is unnested data
df_genre = df_final.groupby('listed_in')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_genre_top10 = df_genre.head(10)
df_genre_top10
```

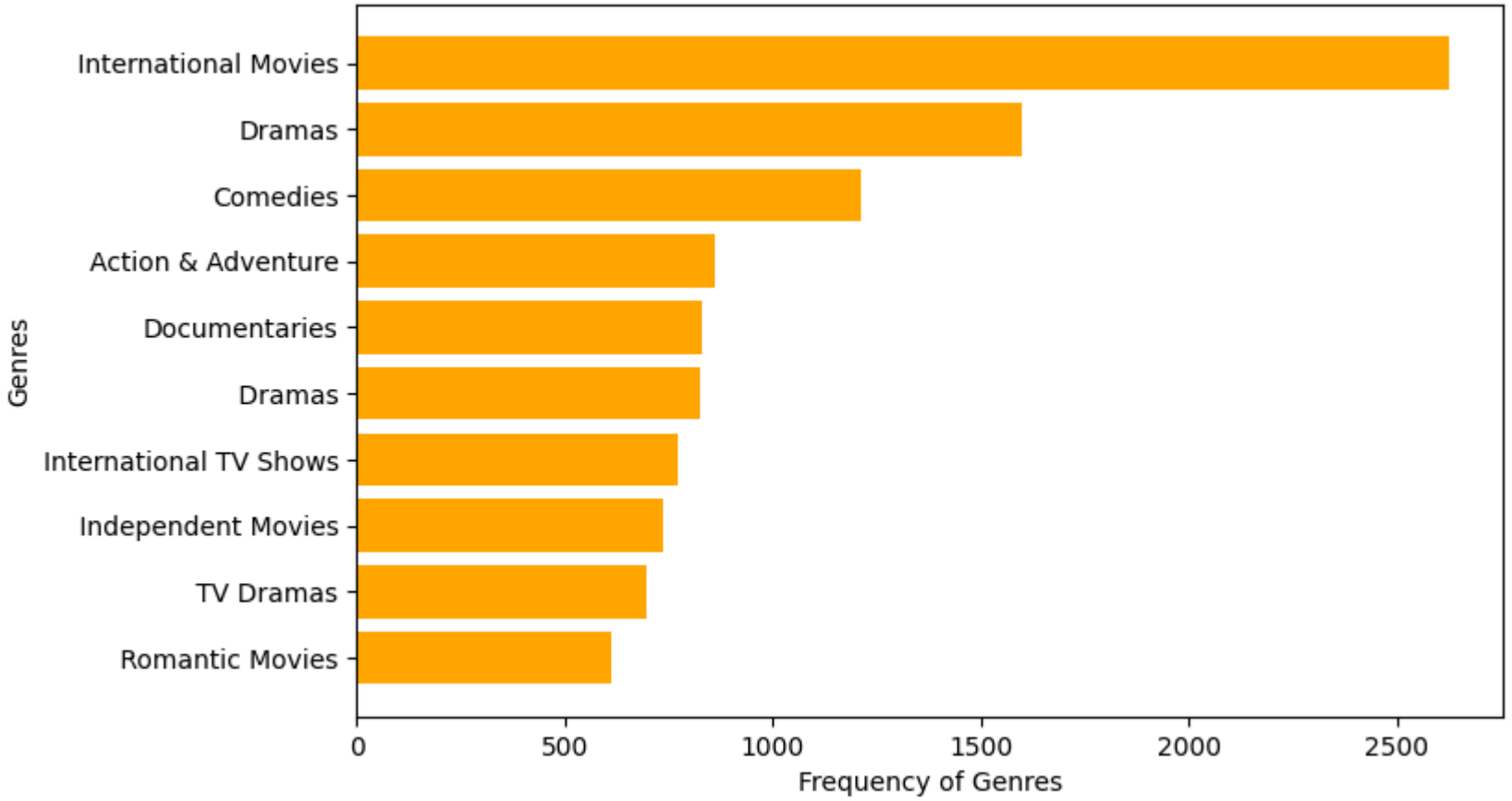
Out[ ]:

listed_in	title
International Movies	2624
Dramas	1600
Comedies	1210
Action & Adventure	859
Documentaries	829
Dramas	827
International TV Shows	774
Independent Movies	736
TV Dramas	696

listed_in	title
Romantic Movies	613

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
plt.barh(df_genre_top10.index[::-1], df_genre_top10['title'][::-1], color='orange')
plt.xlabel('Frequency of Genres')
plt.ylabel('Genres')
plt.show()
```



- International Movies, Dramas and Comedies are the most popular .

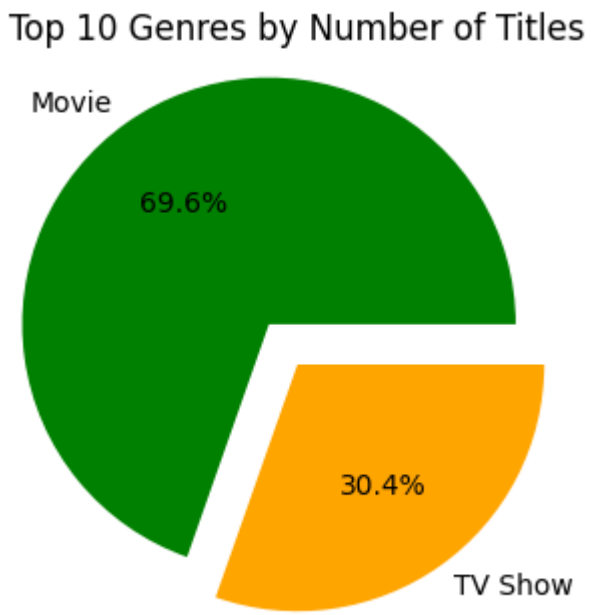
Title based on Type

In [ ]:
```
df_type= df_final.groupby('type')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_type_top10 = df_type.head(10)
df_type_top10
```

Out[ ]:

type	title
Movie	6131
TV Show	2676

In [ ]:
```
plt.figure(figsize=(4, 4))
plt.pie(df_type_top10['title'],
        labels = df_type_top10.index,
        explode = (0.2,0),
        colors= ('green', 'orange'),
        autopct = '%.1f%%')
plt.title('Top 10 Genres by Number of Titles')  # Title of the pie chart
plt.show()
```



- 69.6 % of releases are Movies and 30.4% are TV shows
- we have 70:30 ration of movies and TV shows

Title by country

In [ ]:
```
df_country= df_final.groupby('country')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_country_top10 = df_country.head(10)
df_country_top10
```

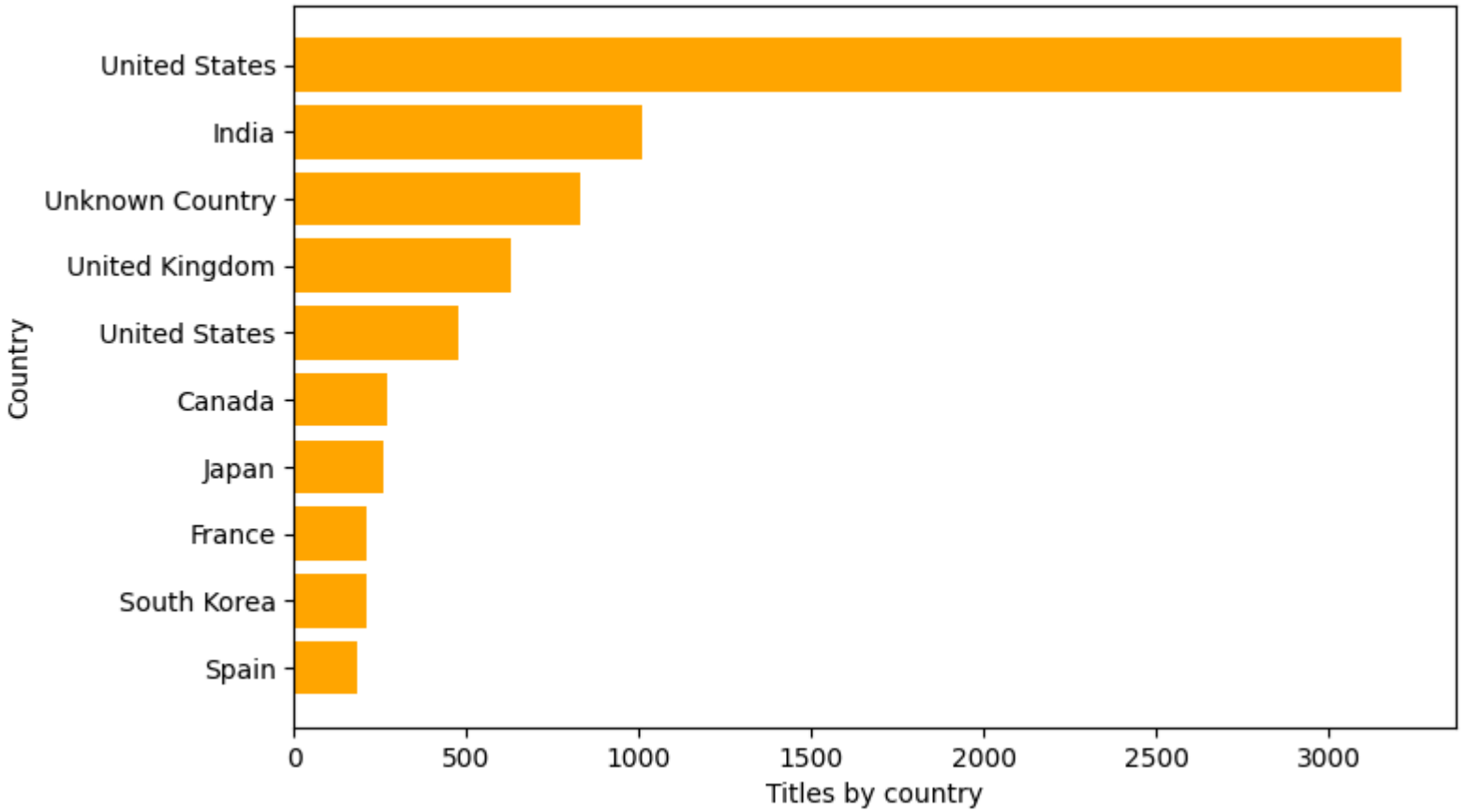
Out[ ]:

country	title
United States	3211
India	1008
Unknown Country	831
United Kingdom	628

country	title
United States	479
Canada	271
Japan	259
France	212
South Korea	211
Spain	181

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
plt.barh(df_country_top10.index[::-1], df_country_top10['title'][::-1], color='orange')
plt.xlabel('Titles by country')
plt.ylabel('Country')
plt.show()
```



- we can see top-10 countries by unique title count
- US, India, UK and Canada are on the top

Title by Ratings

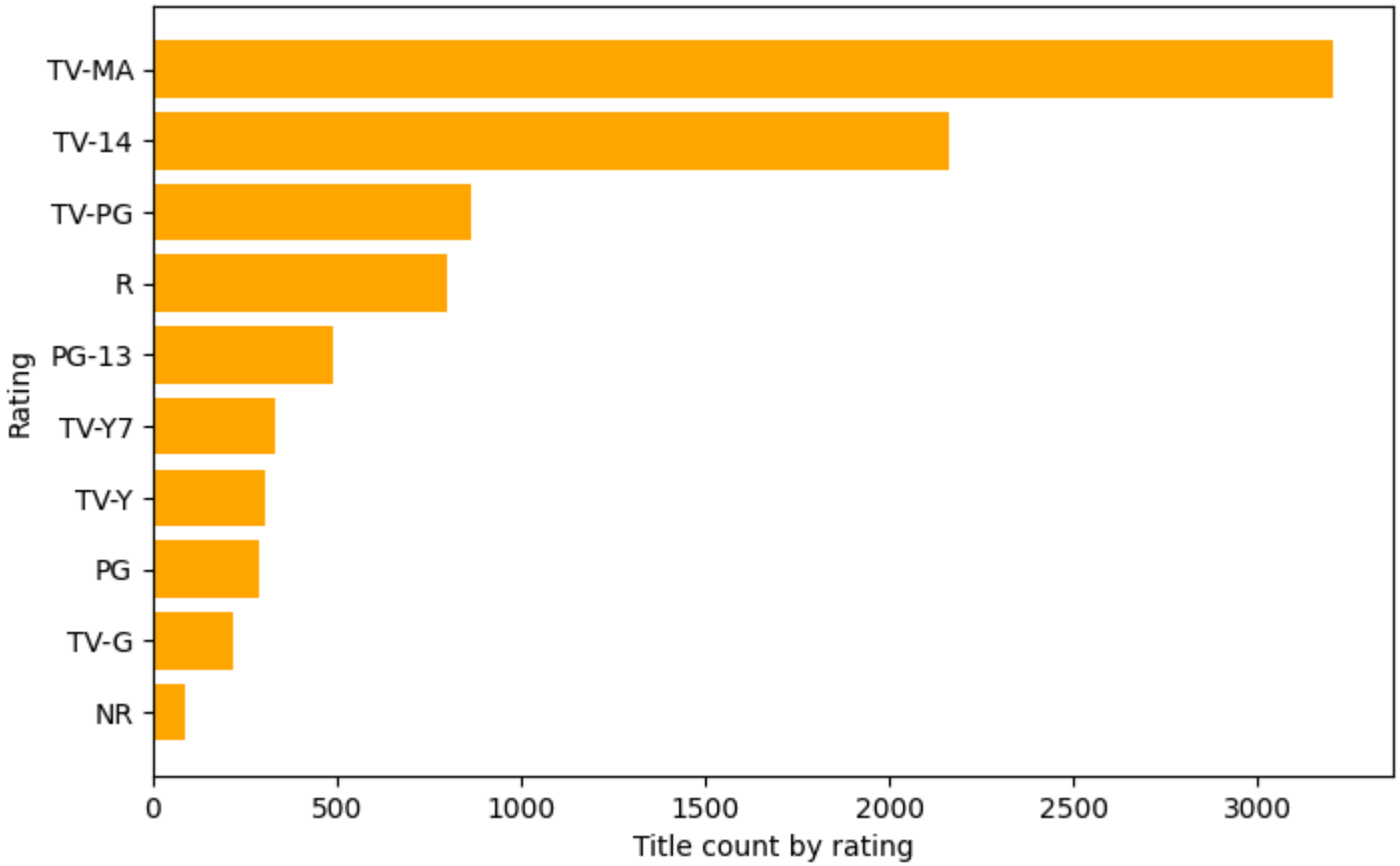
In [ ]:
```
df_rating= df_final.groupby('rating')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_rating_top10 = df_rating.head(10)
df_rating_top10
```

Out[ ]:

rating	title
TV-MA	3207
TV-14	2160
TV-PG	863
R	799
PG-13	490
TV-Y7	334
TV-Y	307
PG	287
TV-G	220
NR	87

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
plt.barh(df_rating_top10.index[::-1], df_rating_top10[::-1]['title'], color='orange')
plt.xlabel('Title count by rating')
plt.ylabel('Rating')
plt.show()
```



- Ratings TV-MA, TV-14 and TV-PG are on the top by title count.
- Meaning most of the mature content is released.

Title based on duration of movie

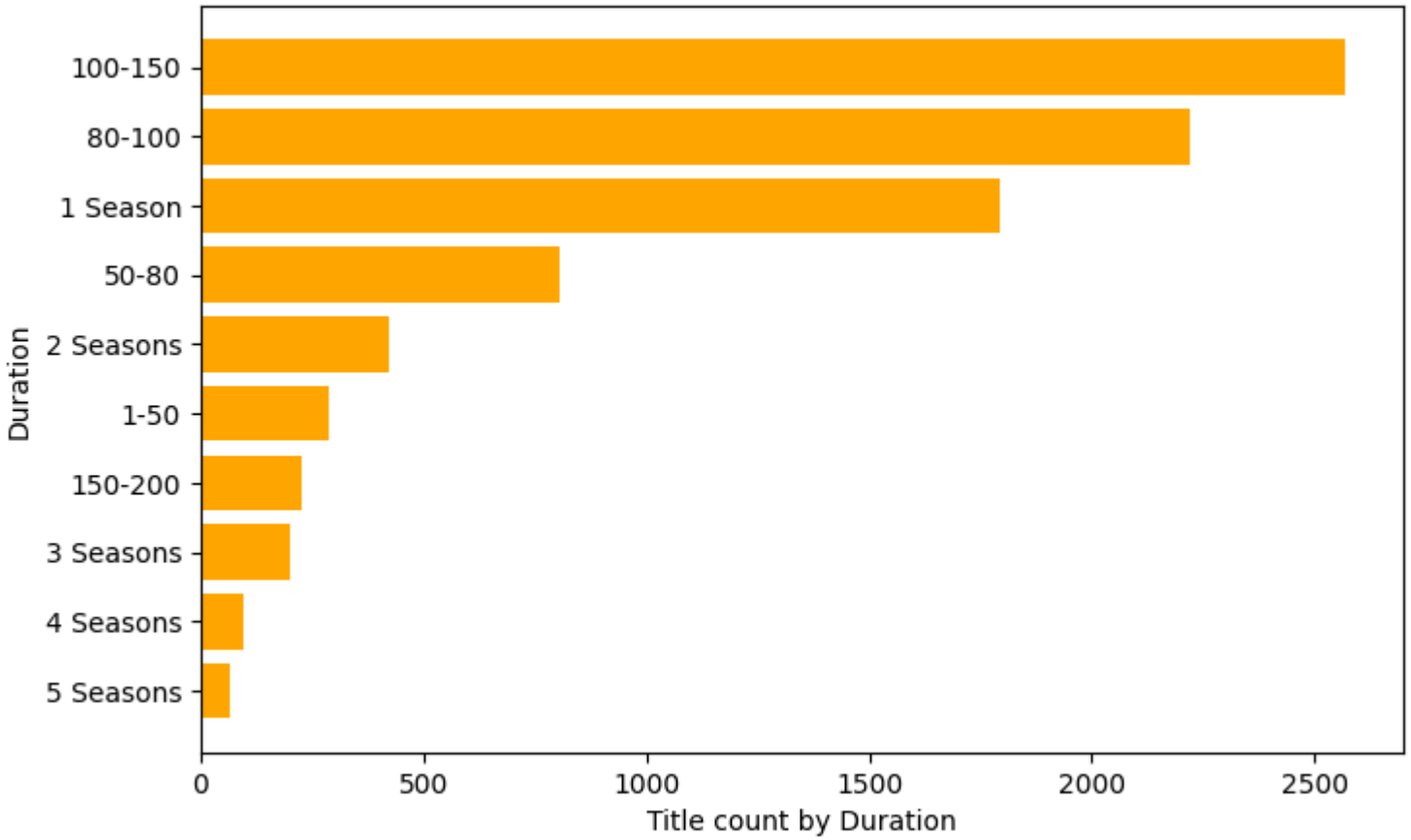
In [ ]:
```
df_duration= df_final.groupby('duration')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_duration_top10 = df_duration.head(10)
df_duration_top10
```

Out[ ]:

duration	title
100-150	2569
80-100	2222
1 Season	1793
50-80	808
2 Seasons	425
1-50	287
150-200	226
3 Seasons	199
4 Seasons	95
5 Seasons	65

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
plt.barh(df_duration_top10.index[::-1], df_duration_top10[::-1]['title'], color='orange')
plt.xlabel('Title count by Duration')
plt.ylabel('Duration')
plt.show()
```



- Most of the content is of duration of 100-150, 80-100 min and contains 1 season.

Title by Actors

In [ ]:
```
df_actor= df_final.groupby('cast')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_actor_top10 = df_actor.head(10)
df_actor_top10
```

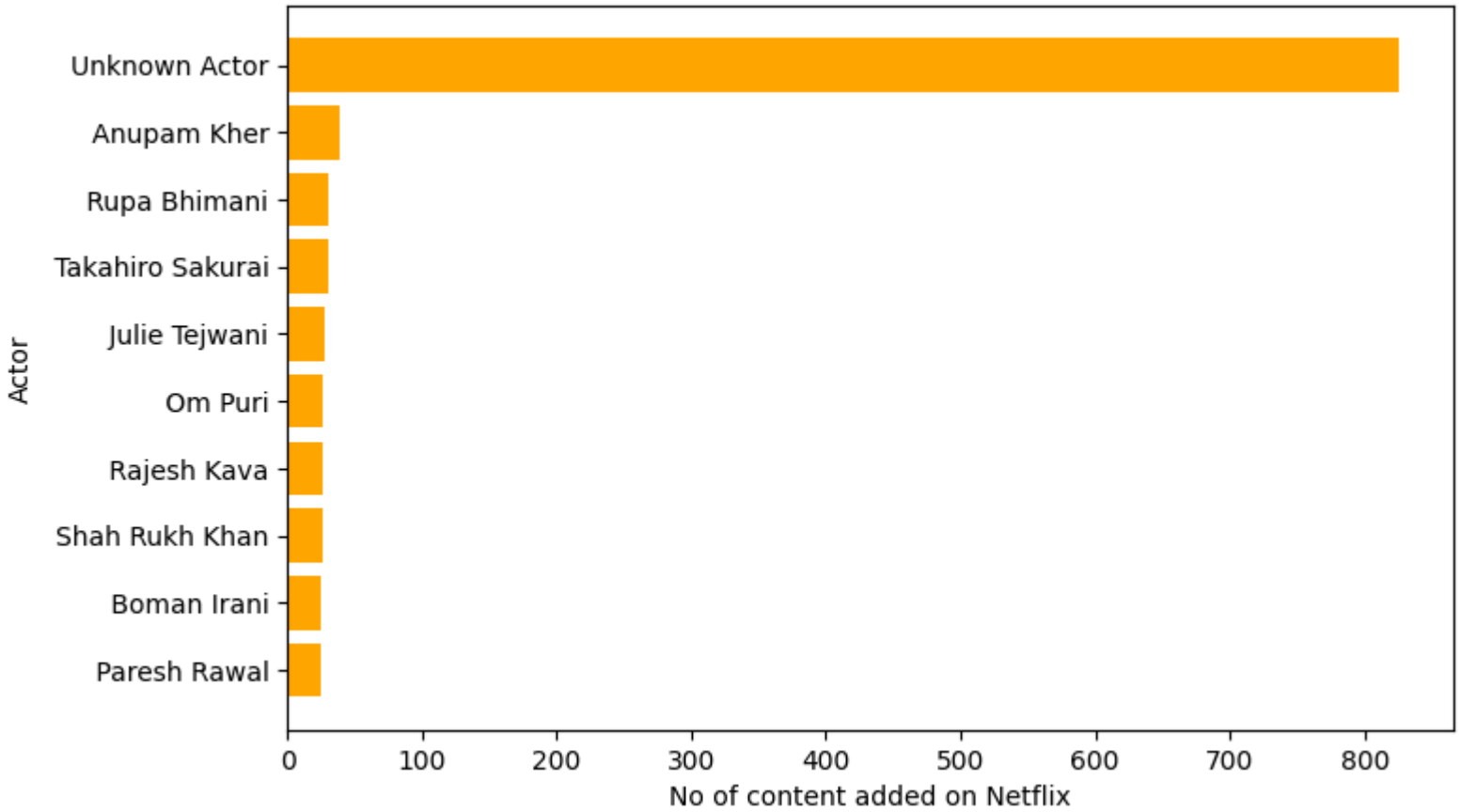
Out[ ]:

cast	title
Unknown Actor	825

cast	title
Anupam Kher	39
Rupa Bhimani	31
Takahiro Sakurai	30
Julie Tejwani	28
Om Puri	27
Rajesh Kava	26
Shah Rukh Khan	26
Boman Irani	25
Paresh Rawal	25

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
plt.barh(df_actor_top10.index[::-1], df_actor_top10[::-1]['title'], color='orange')
plt.xlabel('No of content added on Netflix')
plt.ylabel('Actor')
plt.show()
```



Anupam Kher,SRK,Julie Tejwani, Naseeruddin Shah and Takahiro Sakurai occupy the top stop in Most Watched content.

Title by Directors

In [ ]:
```
df_director= df_final.groupby('director')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_director_top10 = df_director.head(10)
df_director_top10
```

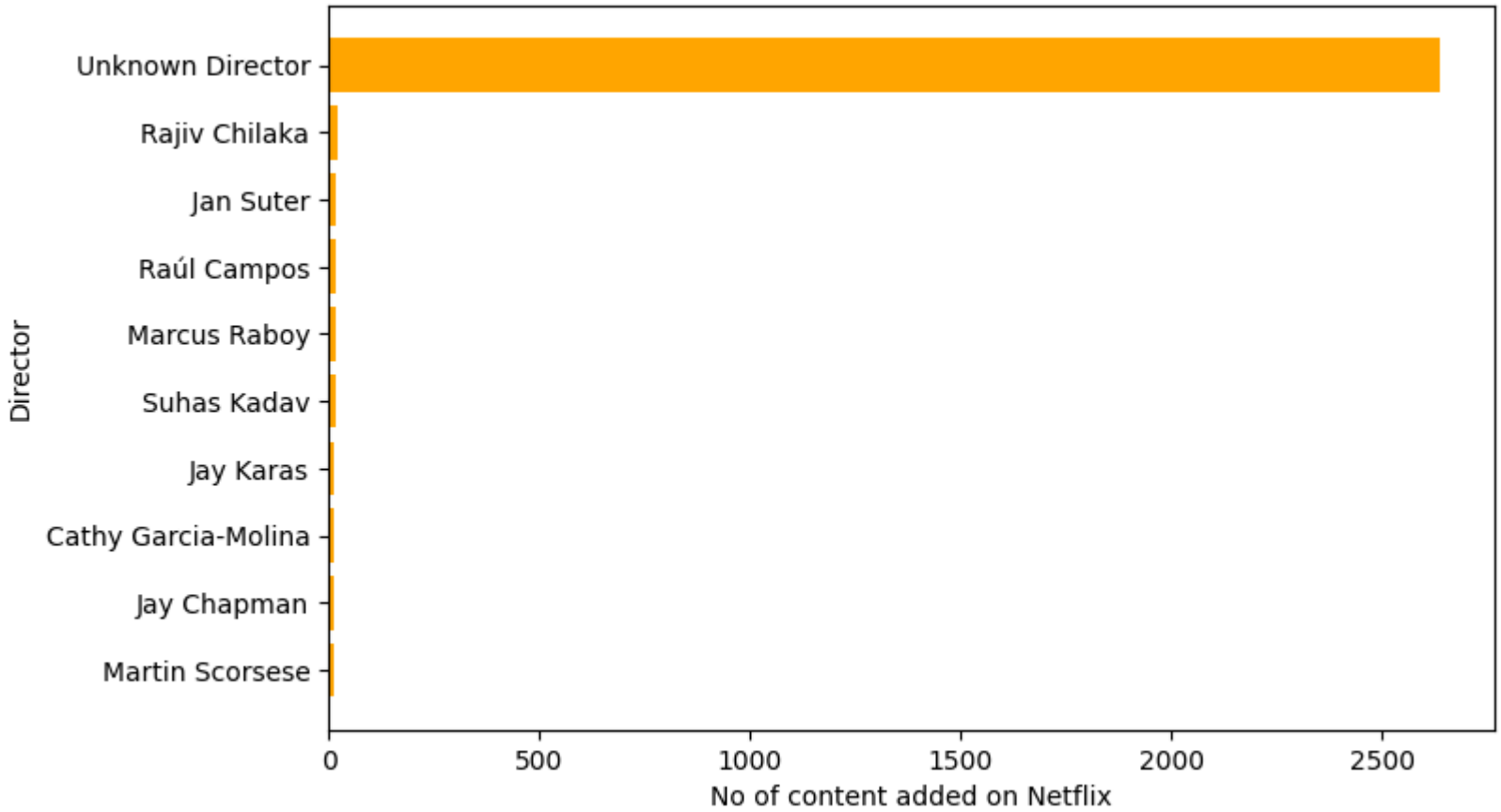
Out[ ]:

director	title
Unknown Director	2634
Rajiv Chilaka	22

director	title
Jan Suter	18
Raúl Campos	18
Marcus Raboy	16
Suhas Kadav	16
Jay Karas	15
Cathy Garcia-Molina	13
Jay Chapman	12
Martin Scorsese	12

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
plt.barh(df_director_top10.index[::-1], df_director_top10[::-1]['title'], color='orange')
plt.xlabel('No of content added on Netflix')
plt.ylabel('Director')
plt.show()
```



Title by Year added

In [ ]:
```
df_year_added= df_final.groupby('year_added')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_year_added.head()
```

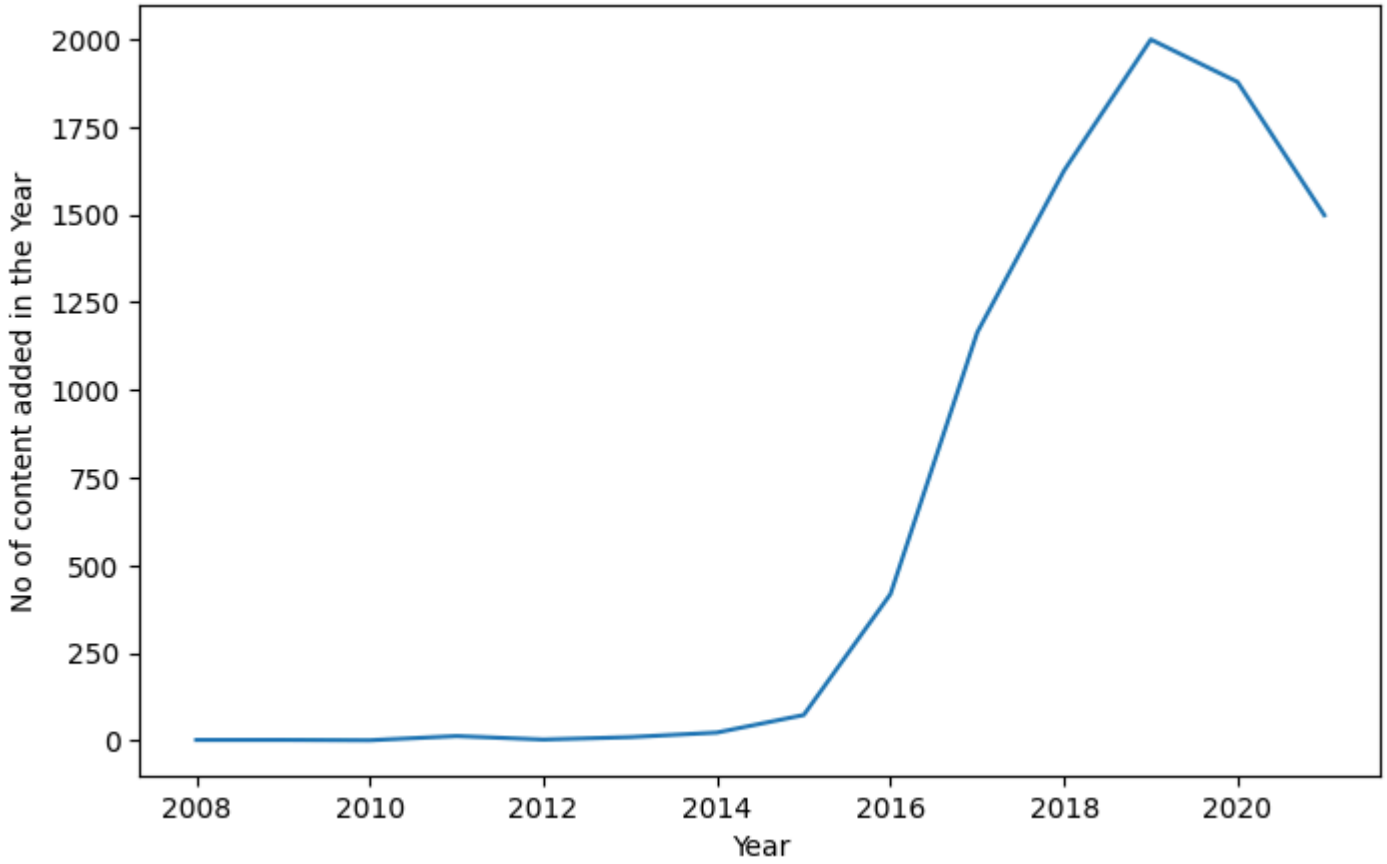
Out[ ]:

year_added	title
2019.0	1999
2020.0	1878
2018.0	1625
2021.0	1498

year_added	title
2017.0	1164

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
sns.lineplot(data=df_year_added, x='year_added', y='title')
plt.ylabel("No of content added in the Year")
plt.xlabel("Year")
plt.show()
```



- Here you can see that no of content added on the Netflix peaked at 2019.
- Content added continuously increased exponentially after 2014 untill 2019.
- started come down after 2019 onwards

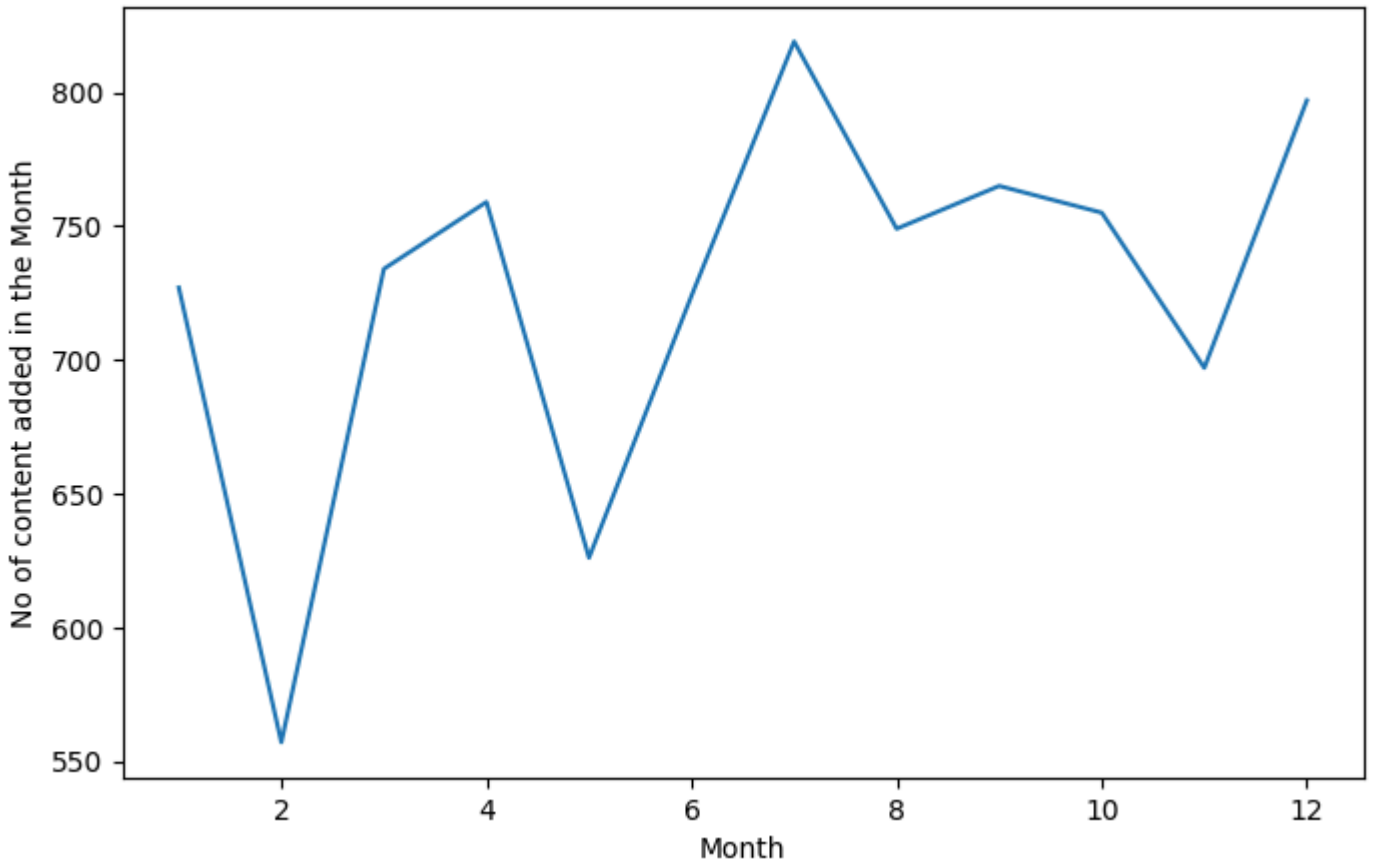
Title by Month added

In [ ]:
```
df_month_added= df_final.groupby('month_added')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_month_added
```

Out[ ]:

month_added	title
7.0	819
12.0	797
9.0	765
4.0	759
10.0	755
8.0	749
3.0	734
1.0	727
6.0	724
11.0	697
5.0	626
2.0	557

In [ ]:
```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
sns.lineplot(data=df_month_added, x='month_added', y='title')
plt.ylabel("No of content added in the Month")
plt.xlabel("Month")
plt.show()
```



- Most of the content is added in the 7th and 12th month

Title by Week added

In [ ]:
```
df_week_added= df_final.groupby('week_added')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_week_added.head()
```

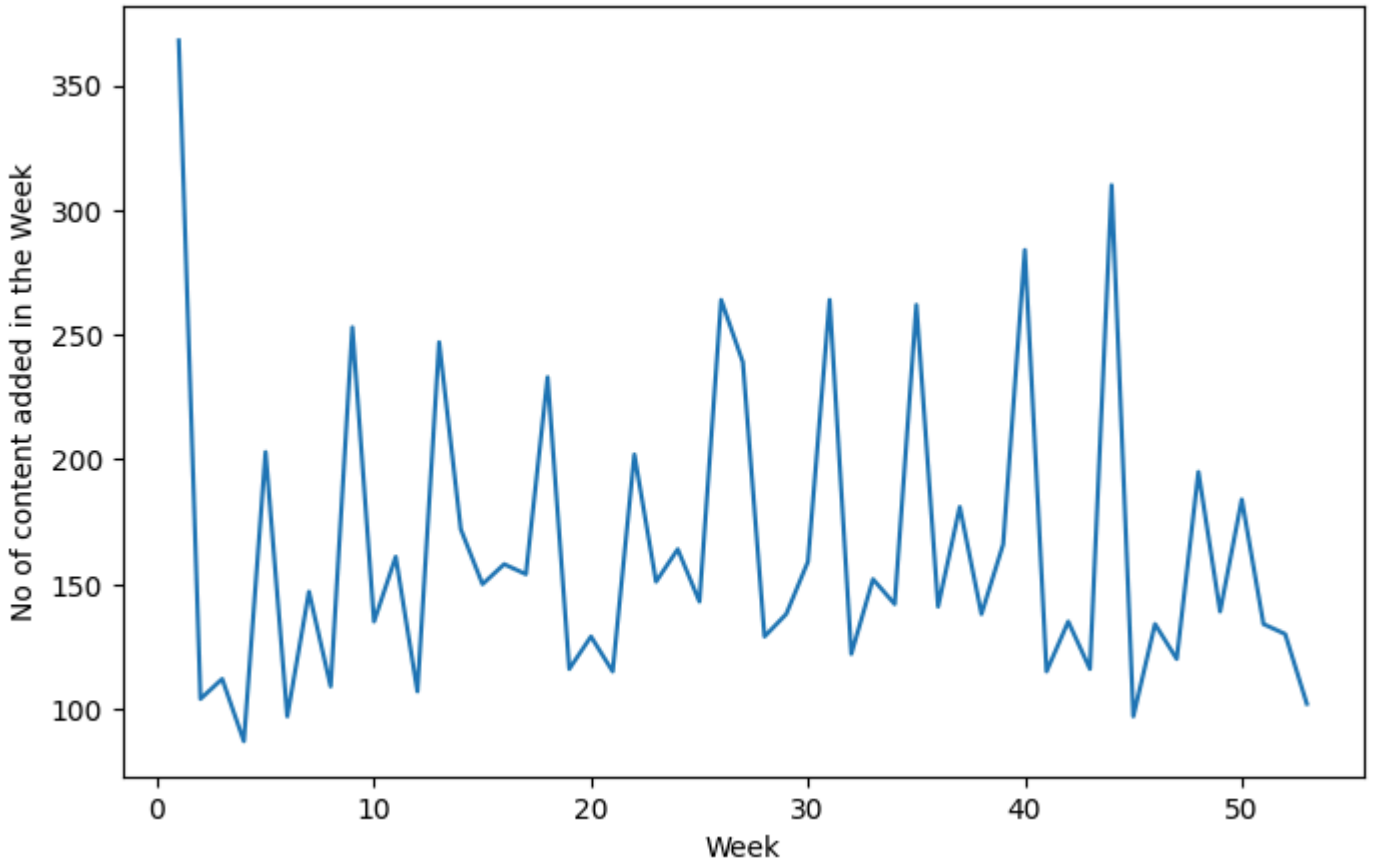
Out[ ]:

week_added	title
1	368
44	310

week_added	title
40	284
31	264
26	264

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
sns.lineplot(data=df_week_added, x='week_added', y='title')
plt.ylabel("No of content added in the Week")
plt.xlabel("Week")
plt.show()
```



- Most of the content is added in 1st week of the year and at the end of the year again.

Title by Release Year

In [ ]:
```
df_year_released= df_final.groupby('release_year')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_year_released
```

Out[ ]:

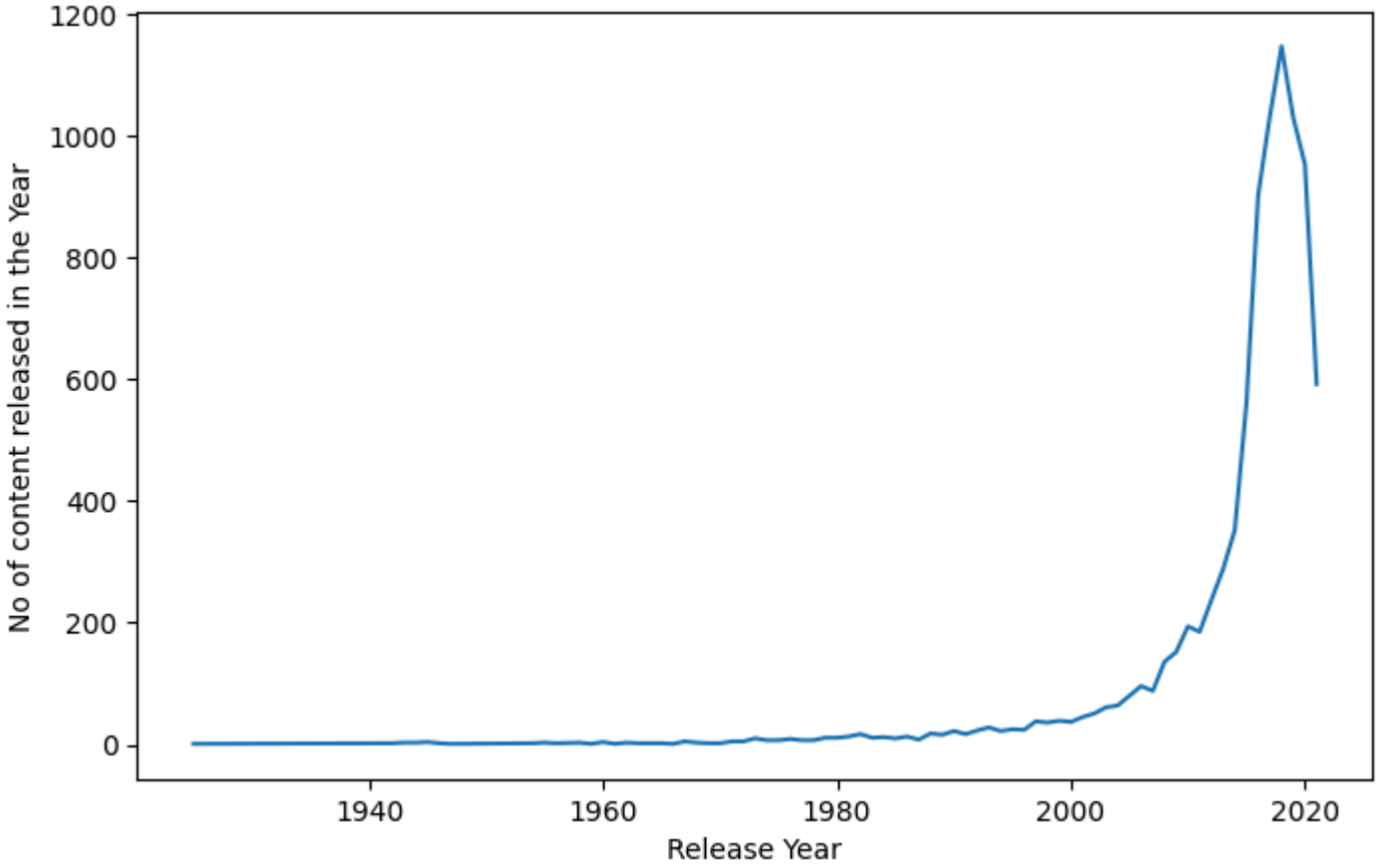
release_year	title
2018	1147
2017	1032

release_year	title
2019	1030
2020	953
2016	902
...	...
1959	1
1961	1
1947	1
1966	1
1925	1

74 rows × 1 columns

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
sns.lineplot(data= df_year_released, x='release_year', y='title')
plt.ylabel("No of content released in the Year")
plt.xlabel("Release Year")
plt.show()
```



- Most of the content avaialble on Netflix is released only in 2018.
- content release picked up after till 2018 and then started coming down.
- decrease in the content release can be due to COVID after 2020 onwards due to lockdown n all.

Univariate Analysis - TV Shows Only

In [ ]:
```
df_shows=df_final[df_final['type']=='TV Show']
df_shows.head()
```

Out[ ]:

	title	director	cast	country	listed_in	show_id	type	date_added	release_year	rating	duration	description	duration1	year_added
1	Blood & Water	Unknown Director	Ama Qamata	South Africa	International TV Shows	s2	TV Show	2021-09-24	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...	<1	2021.0
2	Blood & Water	Unknown Director	Ama Qamata	South Africa	TV Dramas	s2	TV Show	2021-09-24	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...	<1	2021.0
3	Blood & Water	Unknown Director	Ama Qamata	South Africa	TV Mysteries	s2	TV Show	2021-09-24	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...	<1	2021.0
4	Blood & Water	Unknown Director	Khosi Ngema	South Africa	International TV Shows	s2	TV Show	2021-09-24	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...	<1	2021.0
5	Blood & Water	Unknown Director	Khosi Ngema	South Africa	TV Dramas	s2	TV Show	2021-09-24	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...	<1	2021.0

TV Shows based on Genre

In [ ]:

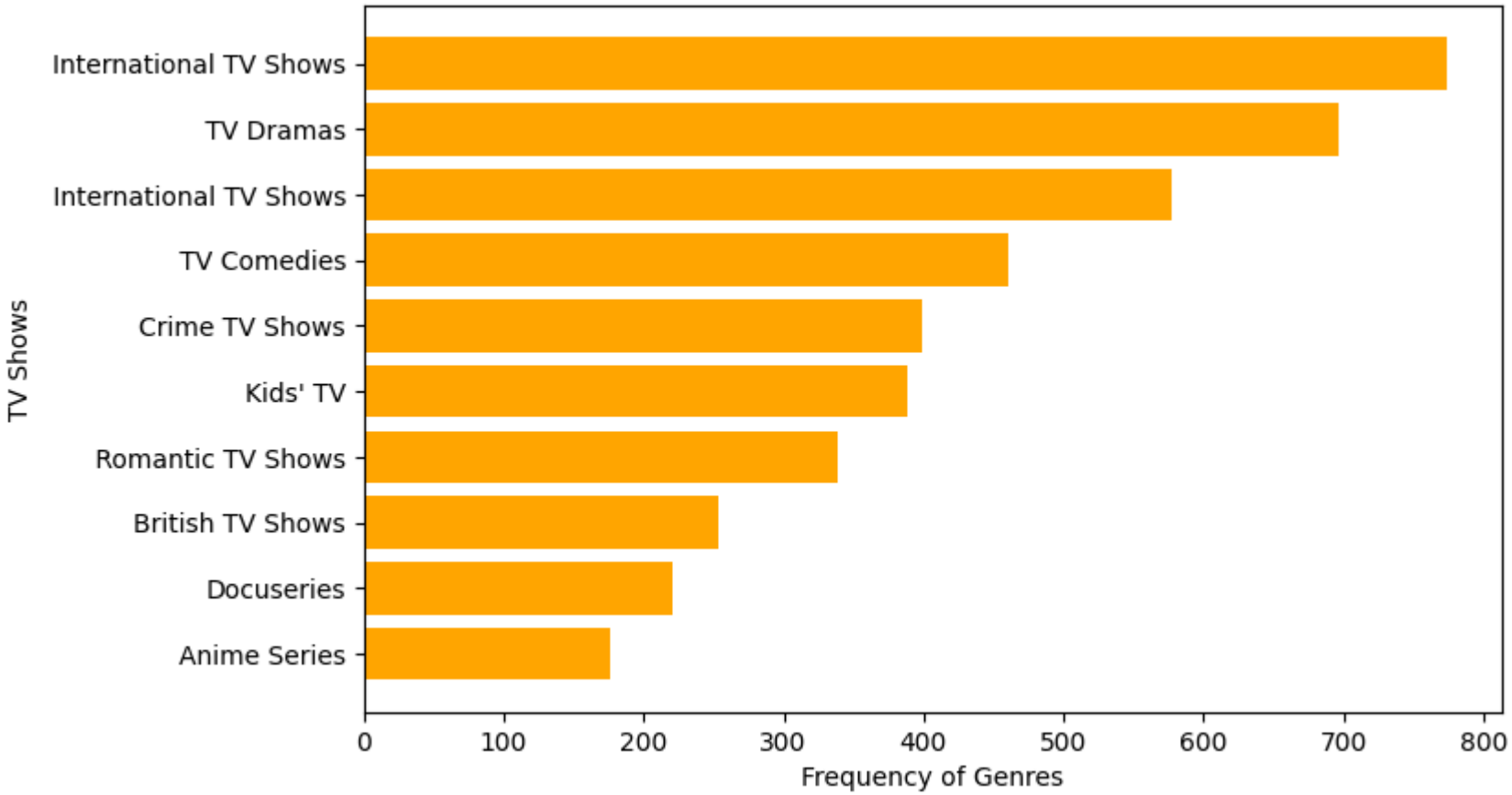
```
#we have to use the nunique only because it is unnested data
df_genre = df_shows.groupby('listed_in')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_genre_top10 = df_genre.head(10)
df_genre_top10
```

Out[ ]:

listed_in	title
International TV Shows	774
TV Dramas	696
International TV Shows	577
TV Comedies	461
Crime TV Shows	399
Kids' TV	388
Romantic TV Shows	338
British TV Shows	253
Docuseries	221
Anime Series	176

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
plt.barh(df_genre_top10.index[::-1], df_genre_top10['title'][::-1], color='orange')
plt.xlabel('Frequency of Genres')
plt.ylabel('TV Shows')
plt.show()
```



- International TV shows, TV Dramas and TV Comedies are the most popular .

TV Shows by country

In [ ]:
```
df_country= df_shows.groupby('country')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_country_top10 = df_country.head(10)
df_country_top10
```

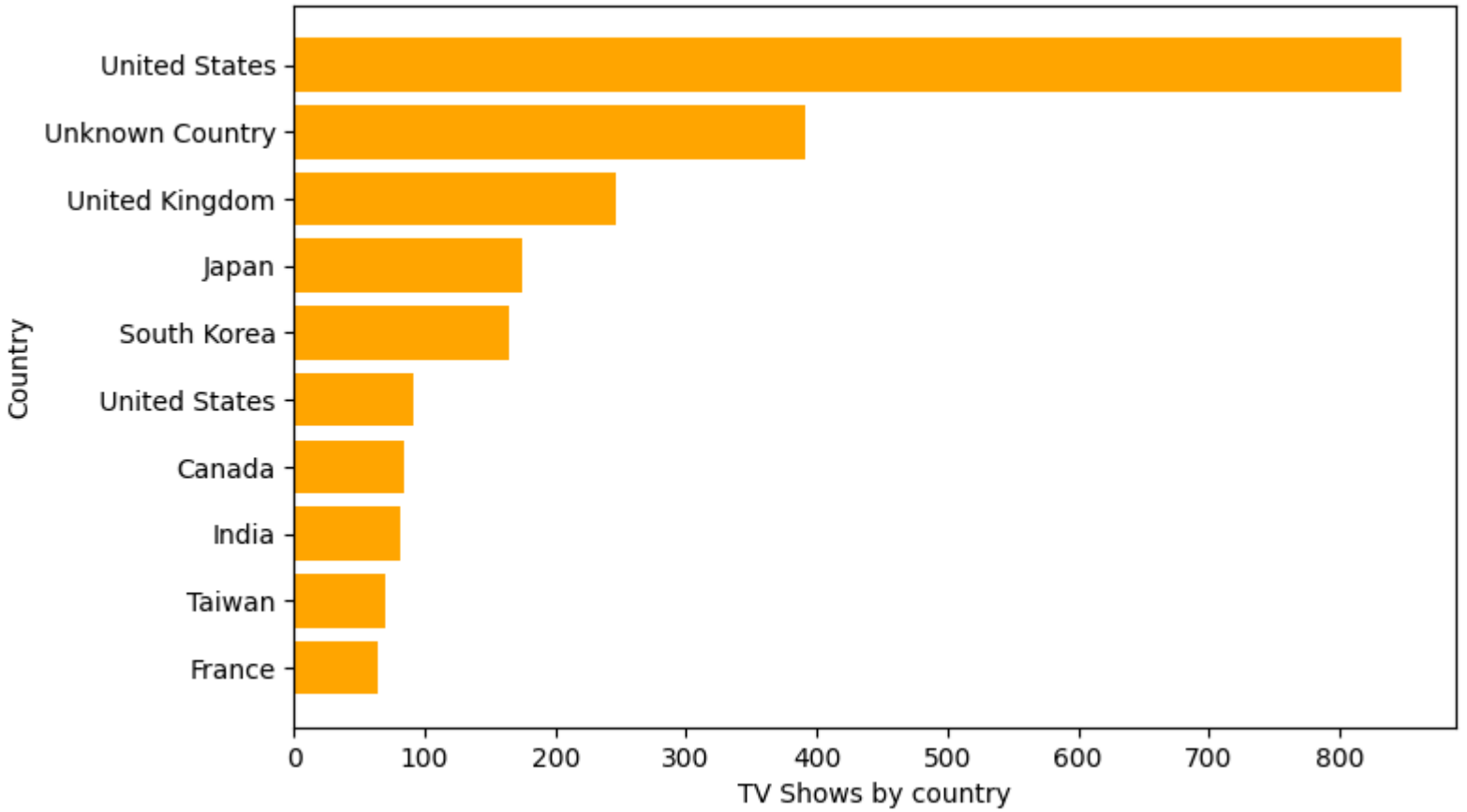
Out[ ]:

country	title
United States	847

country	title
Unknown Country	391
United Kingdom	246
Japan	174
South Korea	164
United States	91
Canada	84
India	81
Taiwan	70
France	64

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
plt.barh(df_country_top10.index[::-1], df_country_top10['title'][::-1], color='orange')
plt.xlabel('TV Shows by country')
plt.ylabel('Country')
plt.show()
```



- we can see top-10 countries by unique title count
- US, India, UK and Canada are on the top

TV Shows by Ratings

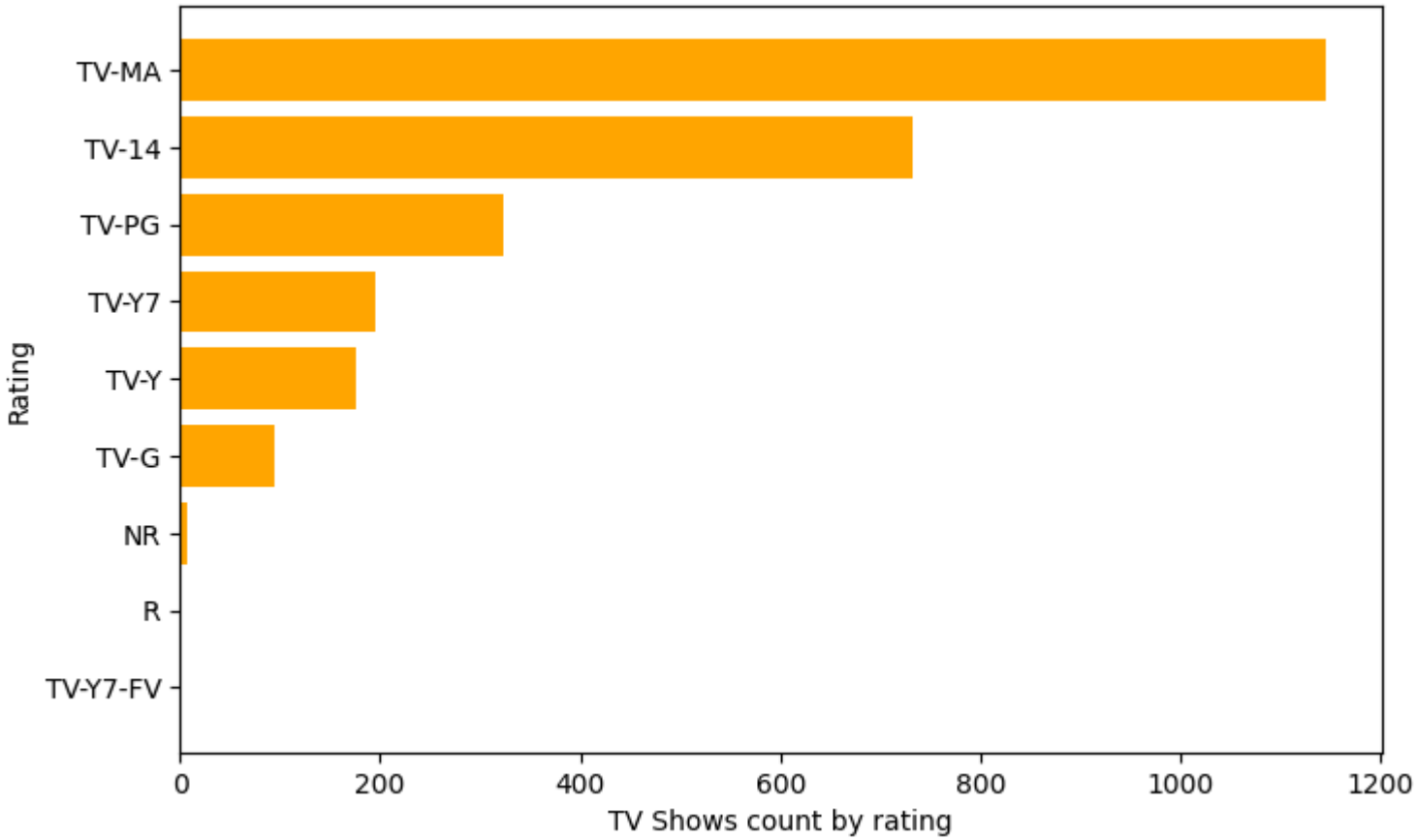
In [ ]:
```
df_rating= df_shows.groupby('rating')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_rating_top10 = df_rating.head(10)
df_rating_top10
```

Out[ ]:

rating	title
TV-MA	1145
TV-14	733
TV-PG	323
TV-Y7	195
TV-Y	176
TV-G	94
NR	7
R	2
TV-Y7-FV	1

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
plt.barh(df_rating_top10.index[::-1], df_rating_top10[::-1]['title'], color='orange')
plt.xlabel('TV Shows count by rating')
plt.ylabel('Rating')
plt.show()
```



- Ratings TV-MA, TV-14 and TV-PG are on the top by TV shows
- Meaning most of the mature content is released.

TV Shows based on duration

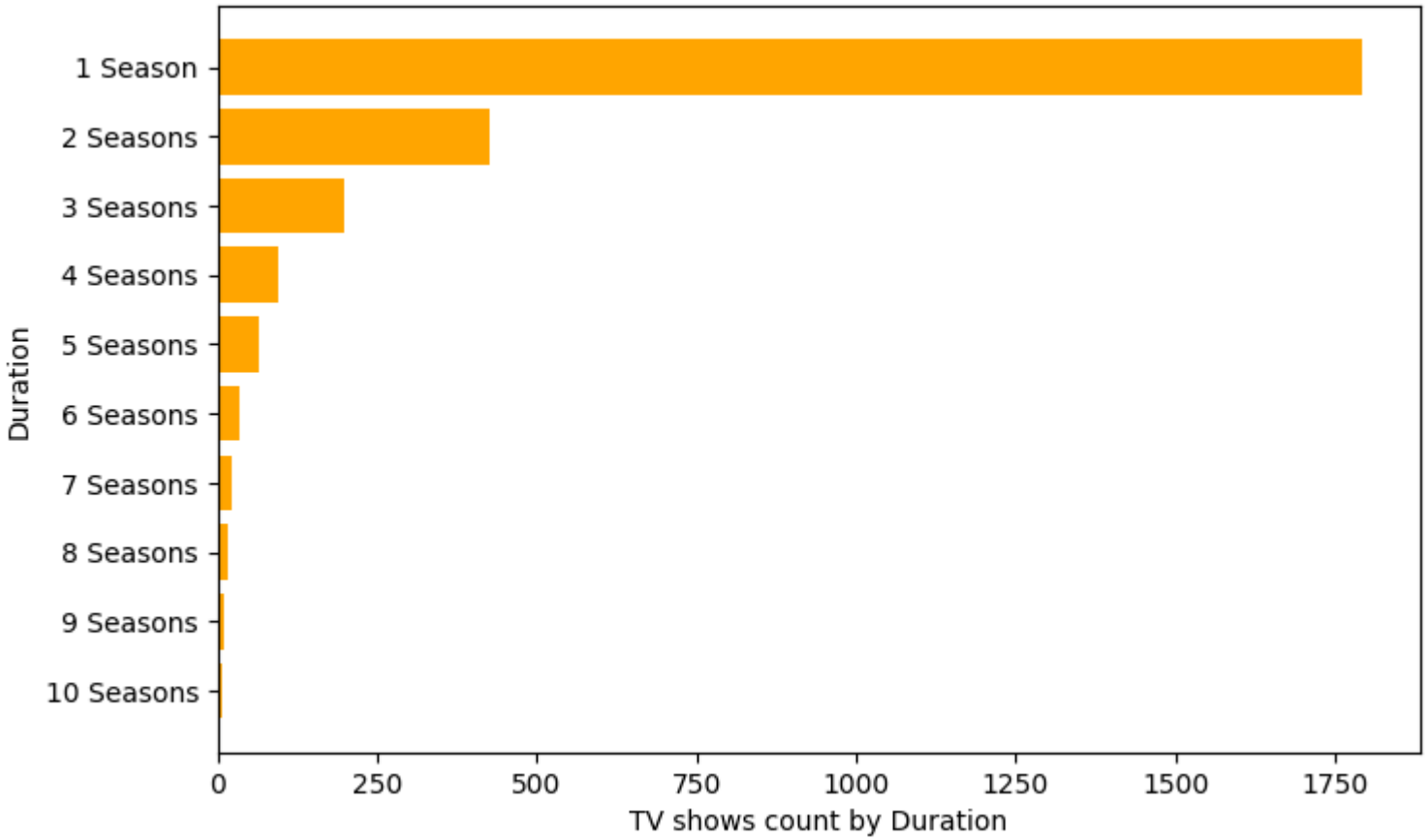
In [ ]:
```
df_duration= df_shows.groupby('duration')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_duration_top10 = df_duration.head(10)
df_duration_top10
```

Out[ ]:

duration	title
1 Season	1793
2 Seasons	425
3 Seasons	199
4 Seasons	95
5 Seasons	65
6 Seasons	33
7 Seasons	23
8 Seasons	17
9 Seasons	9
10 Seasons	7

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
plt.barh(df_duration_top10.index[::-1], df_duration_top10[::-1]['title'], color='orange')
plt.xlabel('TV shows count by Duration')
plt.ylabel('Duration')
plt.show()
```



- Most of the TV shows added have only 1 or 2 seasons.

TV Shows by Actors

In [ ]:
```
df_actor= df_shows.groupby('cast')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_actor_top10 = df_actor.head(10)
df_actor_top10
```

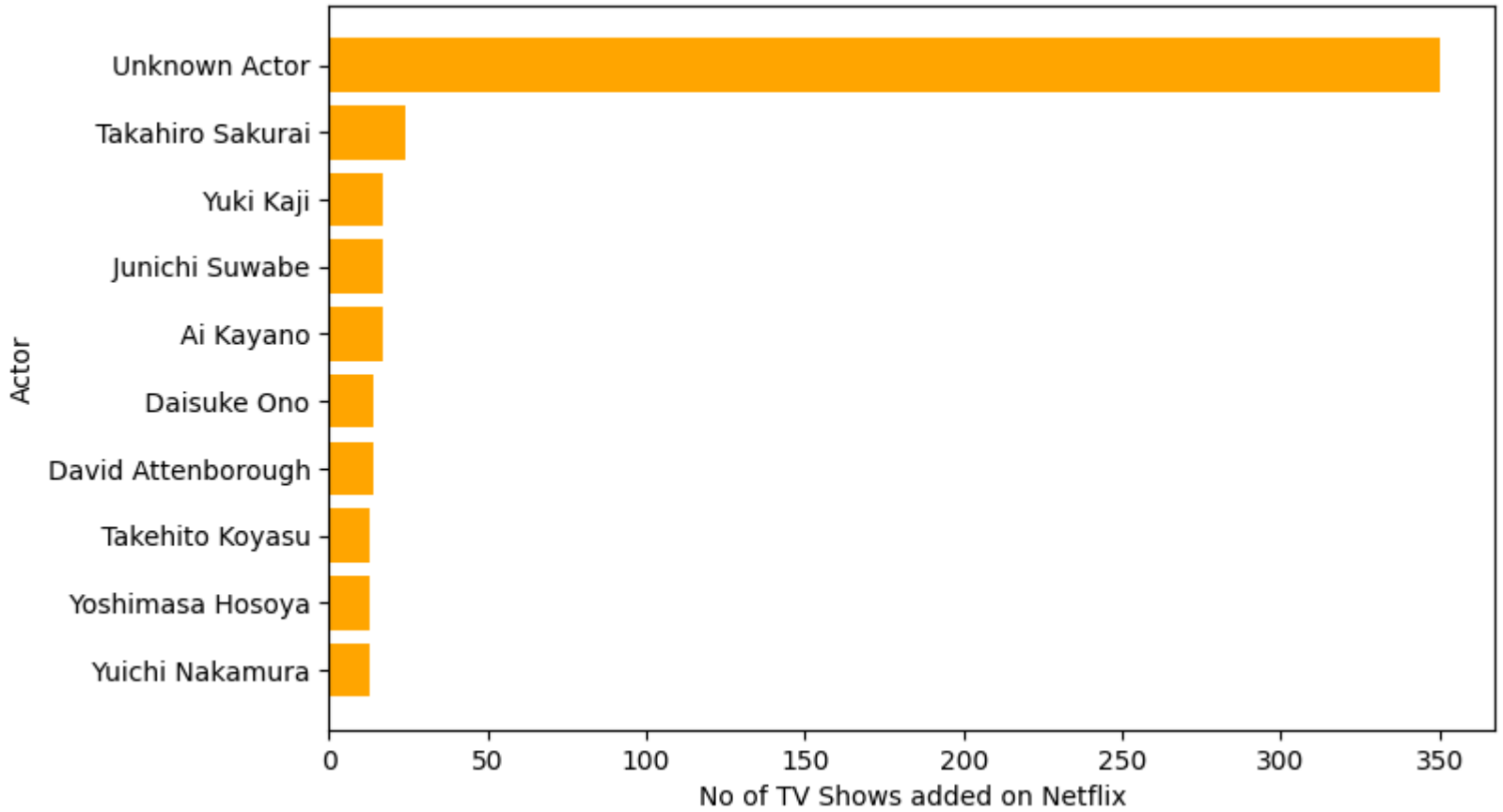
Out[ ]:

cast	title
Unknown Actor	350

cast	title
Takahiro Sakurai	24
Yuki Kaji	17
Junichi Suwabe	17
Ai Kayano	17
Daisuke Ono	14
David Attenborough	14
Takehito Koyasu	13
Yoshimasa Hosoya	13
Yuichi Nakamura	13

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
plt.barh(df_actor_top10.index[::-1], df_actor_top10[::-1]['title'], color='orange')
plt.xlabel('No of TV Shows added on Netflix')
plt.ylabel('Actor')
plt.show()
```



Anupam Kher,SRK,Julie Tejwani, Naseeruddin Shah and Takahiro Sakurai occupy the top stop in Most Watched content.

TV Shows by Directors

In [ ]:
```
df_director= df_shows.groupby('director')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_director_top10 = df_director.head(10)
df_director_top10
```

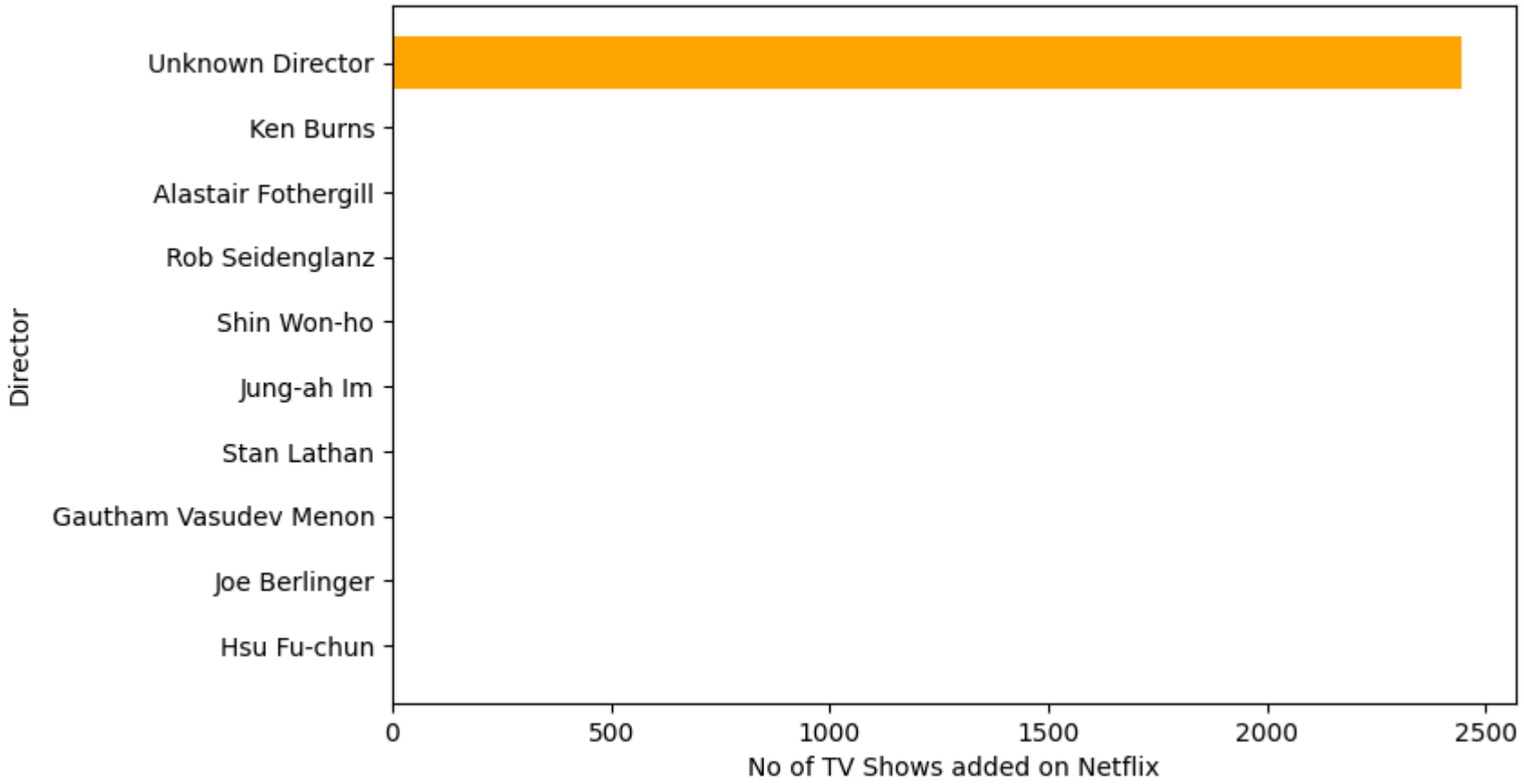
Out[ ]:

director	title
Unknown Director	2446
Ken Burns	3

director	title
Alastair Fothergill	3
Rob Seidenglanz	2
Shin Won-ho	2
Jung-ah Im	2
Stan Lathan	2
Gautham Vasudev Menon	2
Joe Berlinger	2
Hsu Fu-chun	2

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
plt.barh(df_director_top10.index[::-1], df_director_top10[::-1]['title'], color='orange')
plt.xlabel('No of TV Shows added on Netflix')
plt.ylabel('Director')
plt.show()
```



TV Shows by Year added

In [ ]:
```
df_year_added= df_shows.groupby('year_added')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_year_added.head()
```

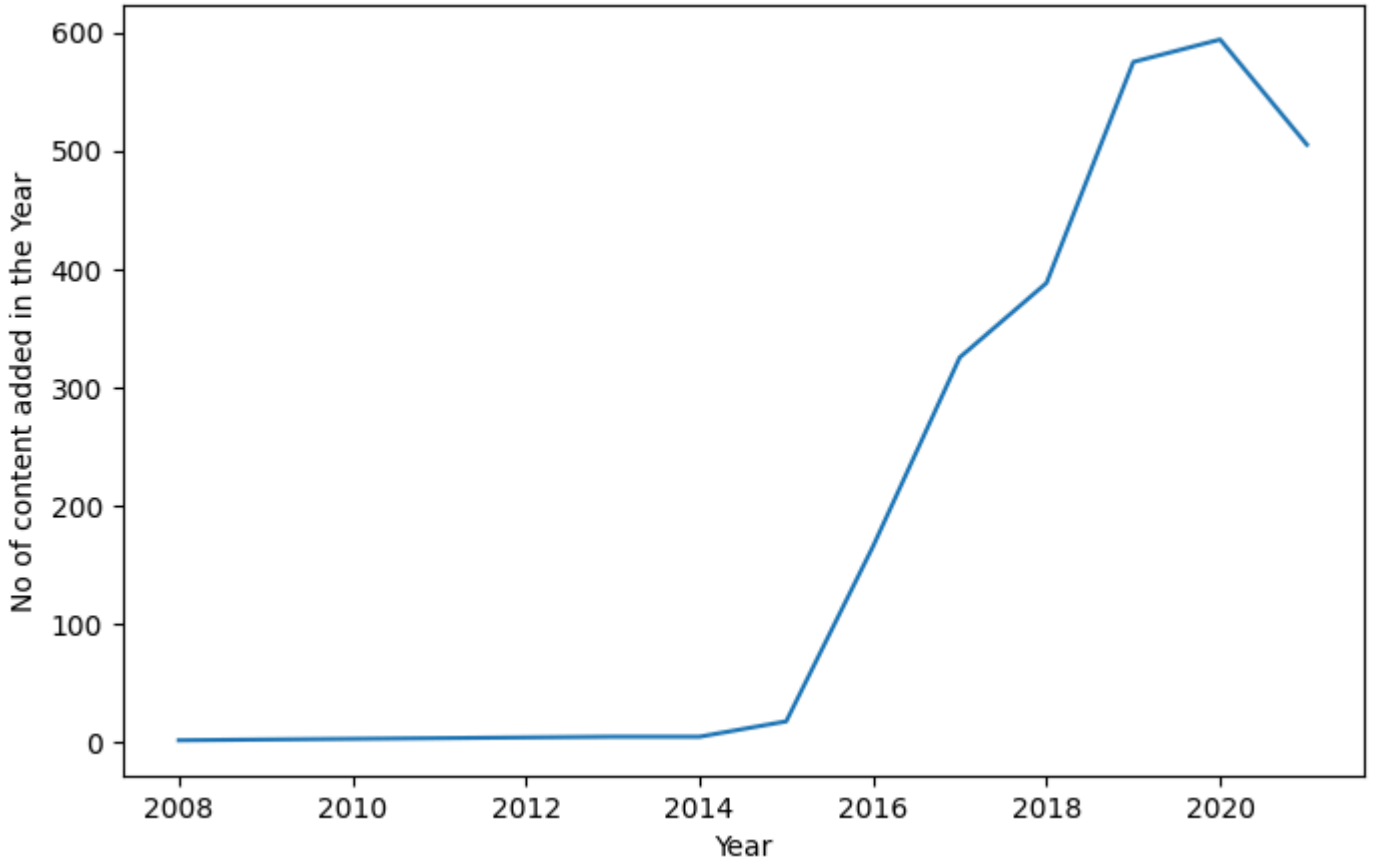
Out[ ]:

year_added	title
2020.0	594
2019.0	575
2021.0	505
2018.0	388

year_added	title
2017.0	325

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
sns.lineplot(data=df_year_added, x='year_added', y='title')
plt.ylabel("No of content added in the Year")
plt.xlabel("Year")
plt.show()
```



- Here you can see that no of TV shows added on the Netflix peaked at 2019.
- TV shows added continuously increased exponentially after 2014 untill 2019.
- started come down after 2019 onwards

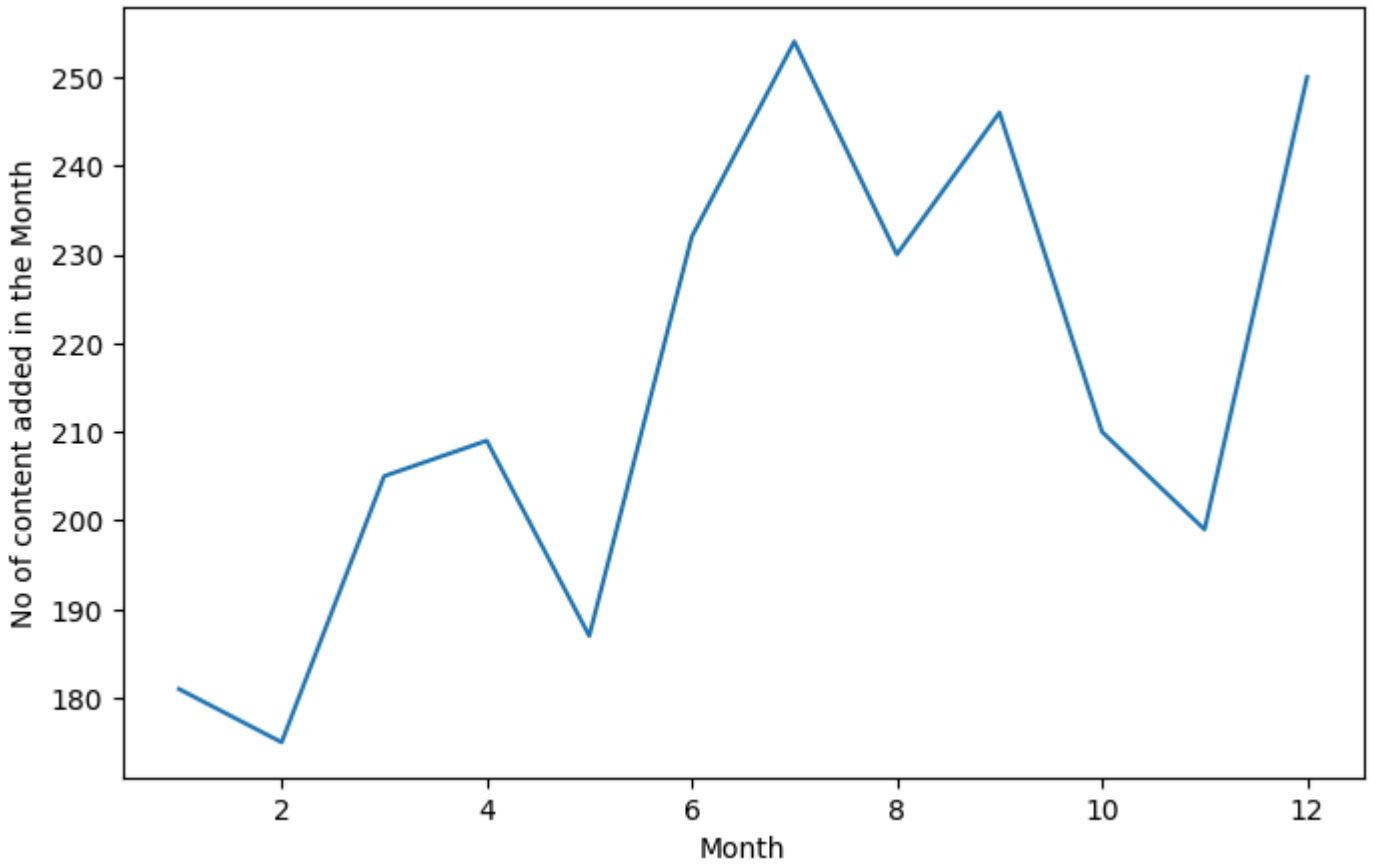
TV Shows by Month added

In [ ]:
```
df_month_added= df_shows.groupby('month_added')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_month_added
```

Out[ ]:

month_added	title
7.0	254
12.0	250
9.0	246
6.0	232
8.0	230
10.0	210
4.0	209
3.0	205
11.0	199
5.0	187
1.0	181
2.0	175

In [ ]:
```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
sns.lineplot(data=df_month_added, x='month_added', y='title')
plt.ylabel("No of content added in the Month")
plt.xlabel("Month")
plt.show()
```



- Most of the TV shows are added in the 7th and 12th month

TV Shows by Week added

In [ ]:
```
df_week_added= df_shows.groupby('week_added')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_week_added.head()
```

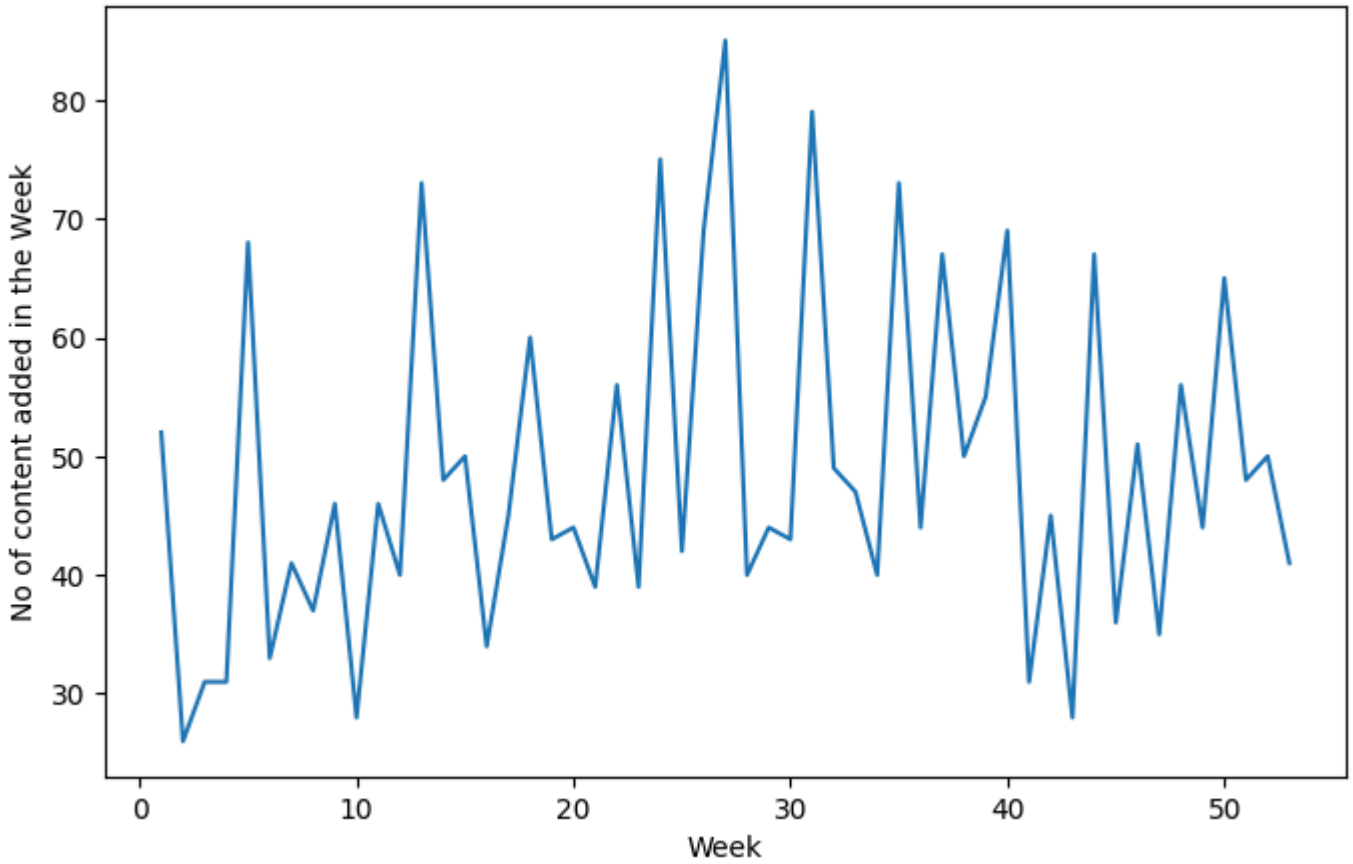
Out[ ]:

week_added	title
27	85
31	79

week_added	title
24	75
35	73
13	73

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
sns.lineplot(data=df_week_added, x='week_added', y='title')
plt.ylabel("No of content added in the Week")
plt.xlabel("Week")
plt.show()
```



- Most of the TV shows are added around 25th week

TV Shows by Release Year

In [ ]:
```
df_year_released= df_shows.groupby('release_year')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_year_released.head()
```

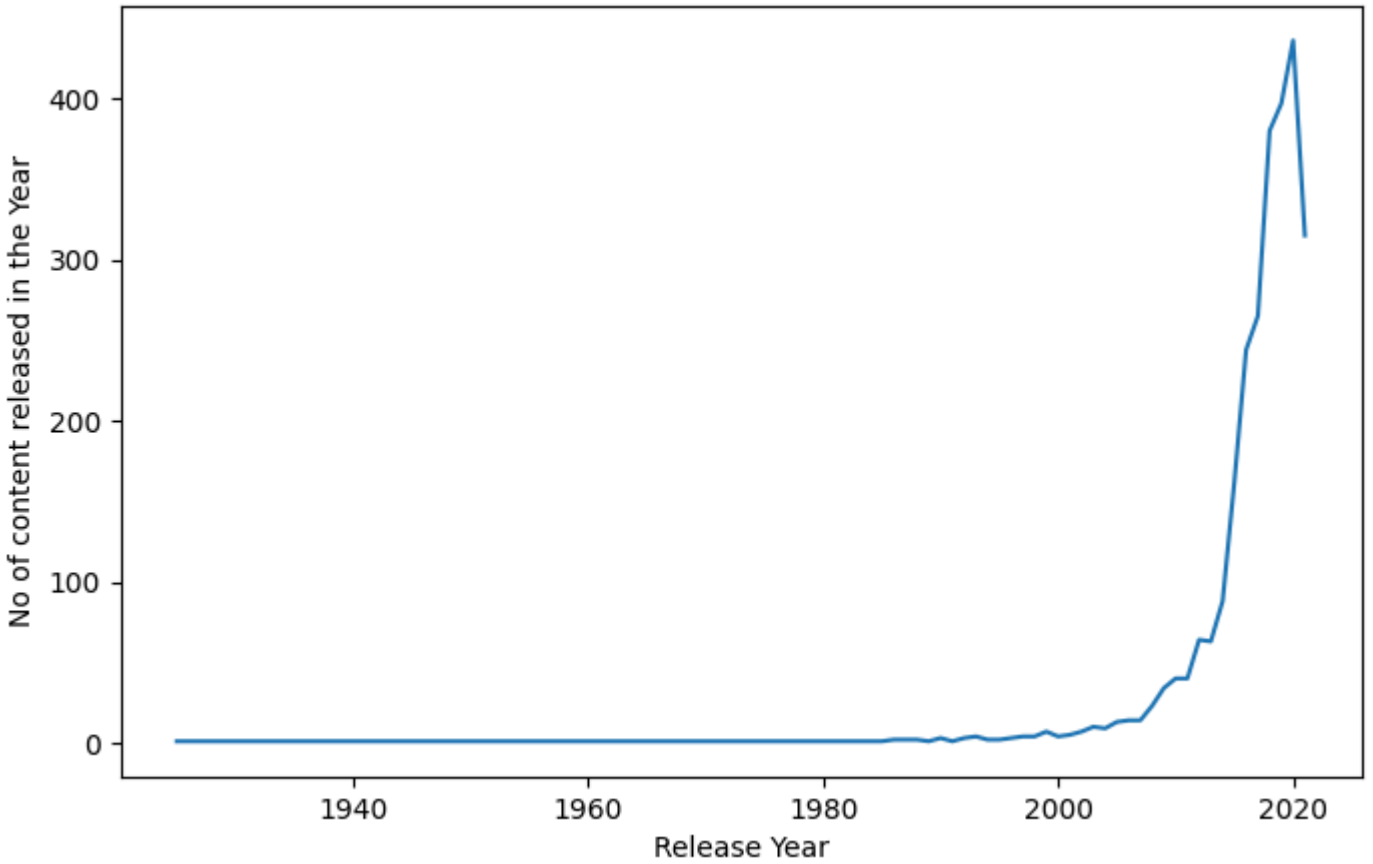
Out[ ]:

release_year	title
2020	436
2019	397

release_year	title
2018	380
2021	315
2017	265

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
sns.lineplot(data= df_year_released, x='release_year', y='title')
plt.ylabel("No of content released in the Year")
plt.xlabel("Release Year")
plt.show()
```



- Most of the TV Shows avaialble on Netflix is released only in 2018.
- content release picked up after till 2018 and then started coming down.
- decrease in the content release can be due to COVID after 2020 onwards due to lockdown n all.

Univariate Analysis - Movies Only

In [ ]:
```
df_movies=df_final[df_final['type']=='Movie']
df_movies.head()
```

Out[ ]:

	title	director	cast	country	listed_in	show_id	type	date_added	release_year	rating	duration	description	duration1
0	Dick Johnson Is Dead	Kirsten Johnson	Unknown Actor	United States	Documentaries	s1	Movie	2021-09-25	2020	PG-13	80-100	As her father nears the end of his life, filmm...	80-100
159	My Little Pony: A New Generation	Robert Cullen	Vanessa Hudgens	Unknown Country	Children & Family Movies	s7	Movie	2021-09-24	2021	PG	80-100	Equestria's divided. But a bright-eyed hero be...	80-100
160	My Little Pony: A New Generation	Robert Cullen	Kimiko Glenn	Unknown Country	Children & Family Movies	s7	Movie	2021-09-24	2021	PG	80-100	Equestria's divided. But a bright-eyed hero be...	80-100
161	My Little Pony: A New Generation	Robert Cullen	James Marsden	Unknown Country	Children & Family Movies	s7	Movie	2021-09-24	2021	PG	80-100	Equestria's divided. But a bright-eyed hero be...	80-100
162	My Little Pony: A New Generation	Robert Cullen	Sofia Carson	Unknown Country	Children & Family Movies	s7	Movie	2021-09-24	2021	PG	80-100	Equestria's divided. But a bright-eyed hero be...	80-100

Movies based on Genre

In [ ]:

```
#we have to use the nunique only because it is unnested data
df_genre = df_movies.groupby('listed_in')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_genre_top10 = df_genre.head(10)
df_genre_top10
```

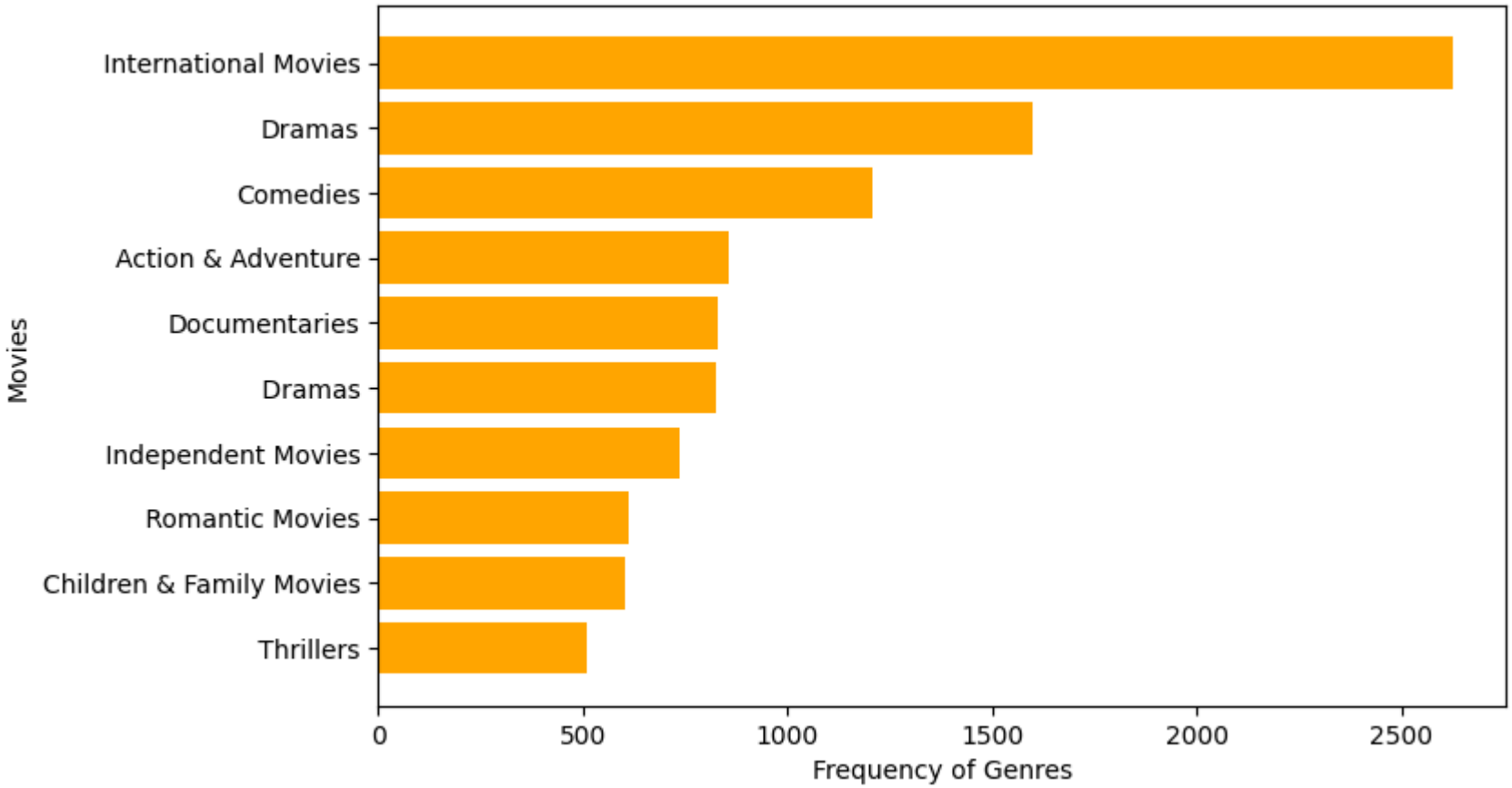
Out[ ]:

listed_in	title
International Movies	2624

listed_in	title
Dramas	1600
Comedies	1210
Action & Adventure	859
Documentaries	829
Dramas	827
Independent Movies	736
Romantic Movies	613
Children & Family Movies	605
Thrillers	512

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
plt.barh(df_genre_top10.index[::-1], df_genre_top10['title'][::-1], color='orange')
plt.xlabel('Frequency of Genres')
plt.ylabel('Movies')
plt.show()
```



- International Movies, Dramas and Comedies are the most popular .

Movies by country

In [ ]:
```
df_country= df_movies.groupby('country')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_country_top10 = df_country.head(10)
df_country_top10
```

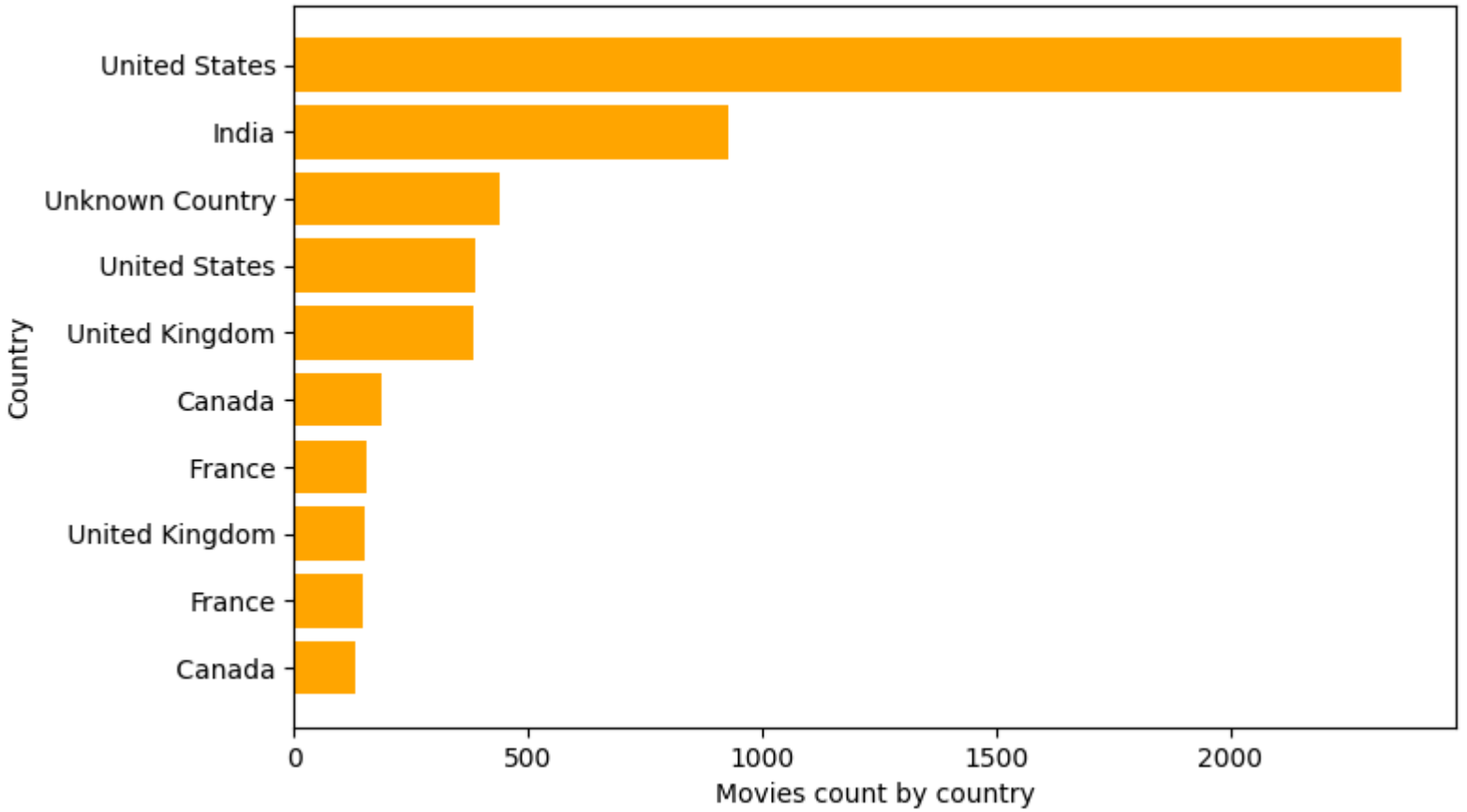
Out[ ]:

country	title
United States	2364

country	title
India	927
Unknown Country	440
United States	388
United Kingdom	382
Canada	187
France	155
United Kingdom	152
France	148
Canada	132

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
plt.barh(df_country_top10.index[::-1], df_country_top10['title'][::-1], color='orange')
plt.xlabel('Movies count by country')
plt.ylabel('Country')
plt.show()
```



- we can see top-10 countries by unique title count
- US, India, UK and Canada are on the top

Movies by Ratings

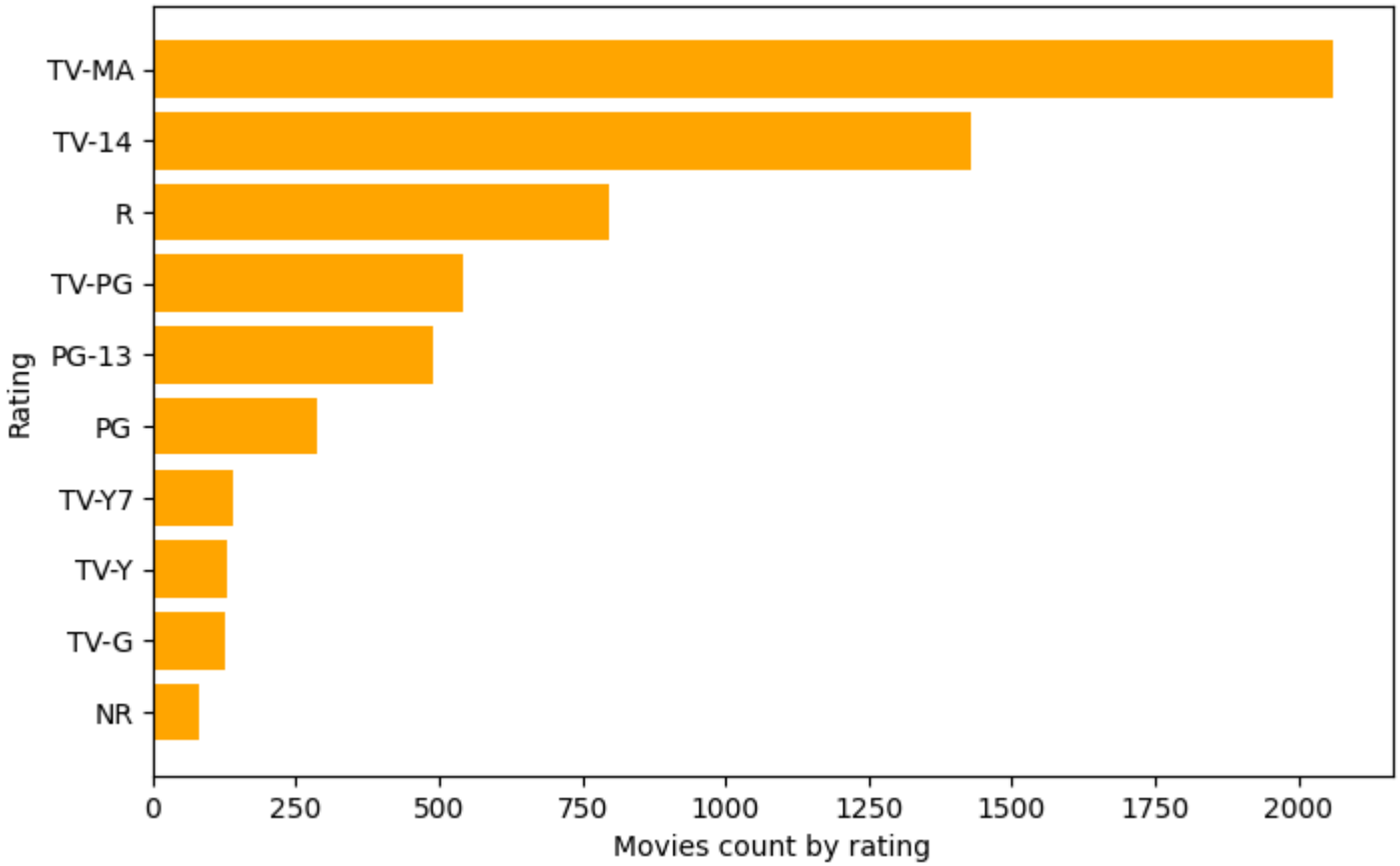
In [ ]:
```
df_rating= df_movies.groupby('rating')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_rating_top10 = df_rating.head(10)
df_rating_top10
```

Out[ ]:

rating	title
TV-MA	2062
TV-14	1427
R	797
TV-PG	540
PG-13	490
PG	287
TV-Y7	139
TV-Y	131
TV-G	126
NR	80

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
plt.barh(df_rating_top10.index[::-1], df_rating_top10[::-1]['title'], color='orange')
plt.xlabel('Movies count by rating')
plt.ylabel('Rating')
plt.show()
```



- Ratings TV-MA, TV-14 and R are on the top by Movies
- Meaning most of the mature content is released.

Movies based on duration

In [ ]:

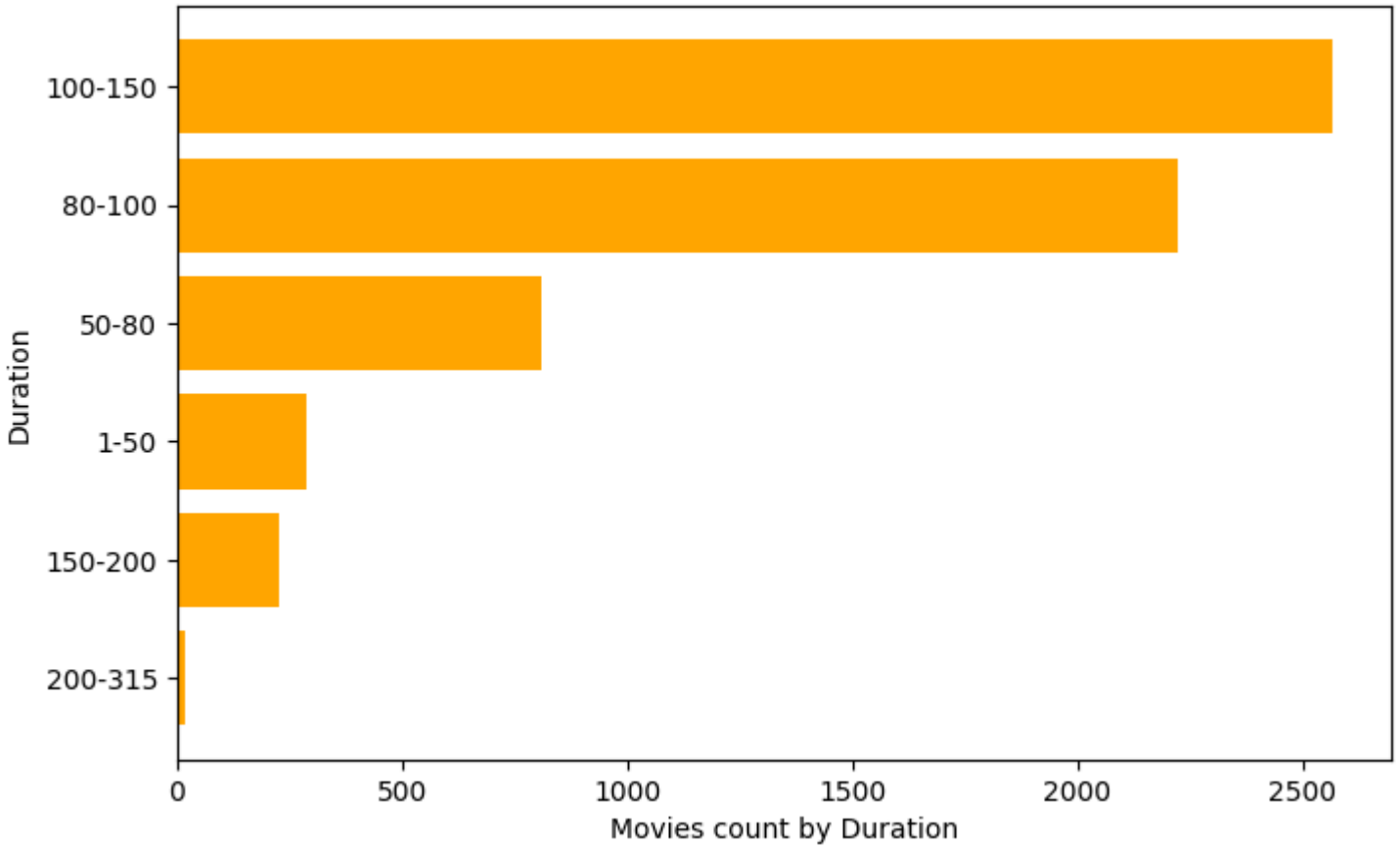
```
df_duration= df_movies.groupby('duration')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_duration_top10 = df_duration.head(10)
df_duration_top10
```

Out[ ]:

duration	title
100-150	2569
80-100	2222
50-80	808
1-50	287
150-200	226
200-315	19

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
plt.barh(df_duration_top10.index[::-1], df_duration_top10[::-1]['title'], color='orange')
plt.xlabel('Movies count by Duration')
plt.ylabel('Duration')
plt.show()
```



- Most of the movies added are of duration 100-150 mins and 80-100 mins

Movies by Actors

In [ ]:
```
df_actor= df_movies.groupby('cast')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_actor_top10 = df_actor.head(10)
df_actor_top10
```

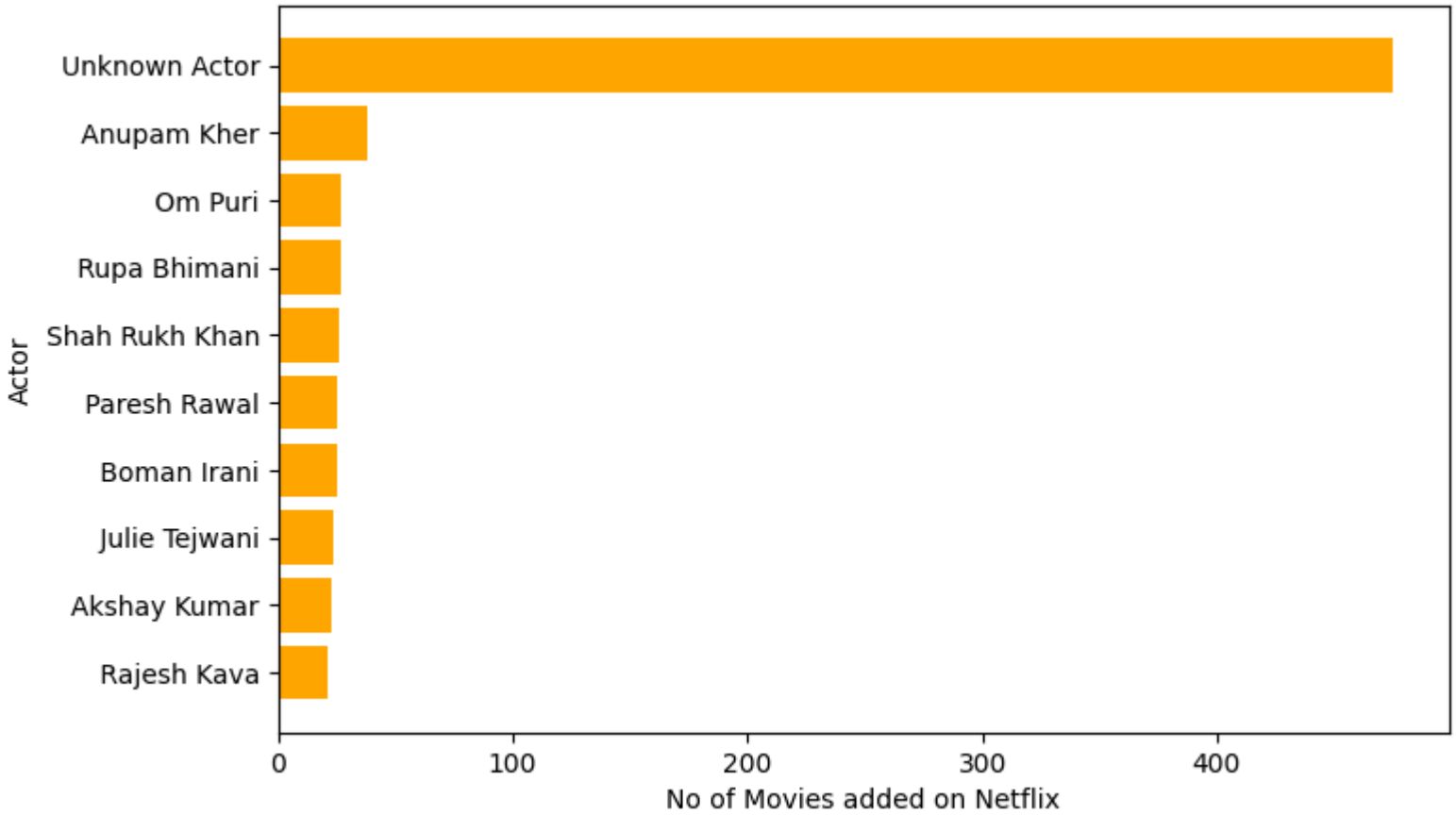
Out[ ]:

cast	title
Unknown Actor	475

cast	title
Anupam Kher	38
Om Puri	27
Rupa Bhimani	27
Shah Rukh Khan	26
Paresh Rawal	25
Boman Irani	25
Julie Tejwani	24
Akshay Kumar	23
Rajesh Kava	21

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
plt.barh(df_actor_top10.index[::-1], df_actor_top10[::-1]['title'], color='orange')
plt.xlabel('No of Movies added on Netflix')
plt.ylabel('Actor')
plt.show()
```



Anupam Kher,Om Puri and Rupa Bhimani occupy the top stop in Most Watched actors in movies

Movies by Directors

In [ ]:
```
df_director= df_movies.groupby('director')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_director_top10 = df_director.head(10)
df_director_top10
```

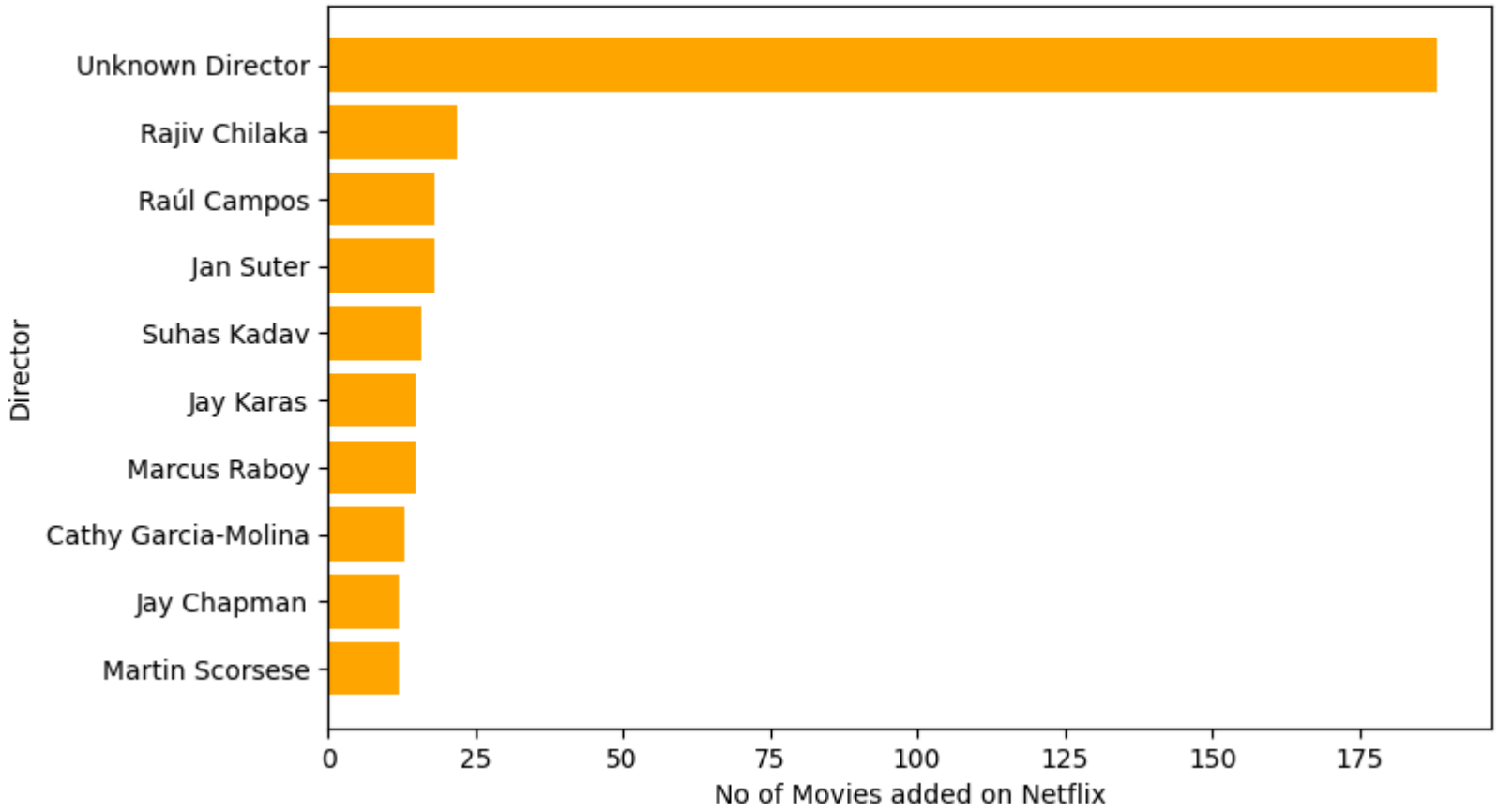
Out[ ]:

director	title
Unknown Director	188
Rajiv Chilaka	22

director	title
Raúl Campos	18
Jan Suter	18
Suhas Kadav	16
Jay Karas	15
Marcus Raboy	15
Cathy Garcia-Molina	13
Jay Chapman	12
Martin Scorsese	12

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
plt.barh(df_director_top10.index[::-1], df_director_top10[::-1]['title'], color='orange')
plt.xlabel('No of Movies added on Netflix')
plt.ylabel('Director')
plt.show()
```



Movies by Year added

In [ ]:
```
df_year_added= df_movies.groupby('year_added')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_year_added.head()
```

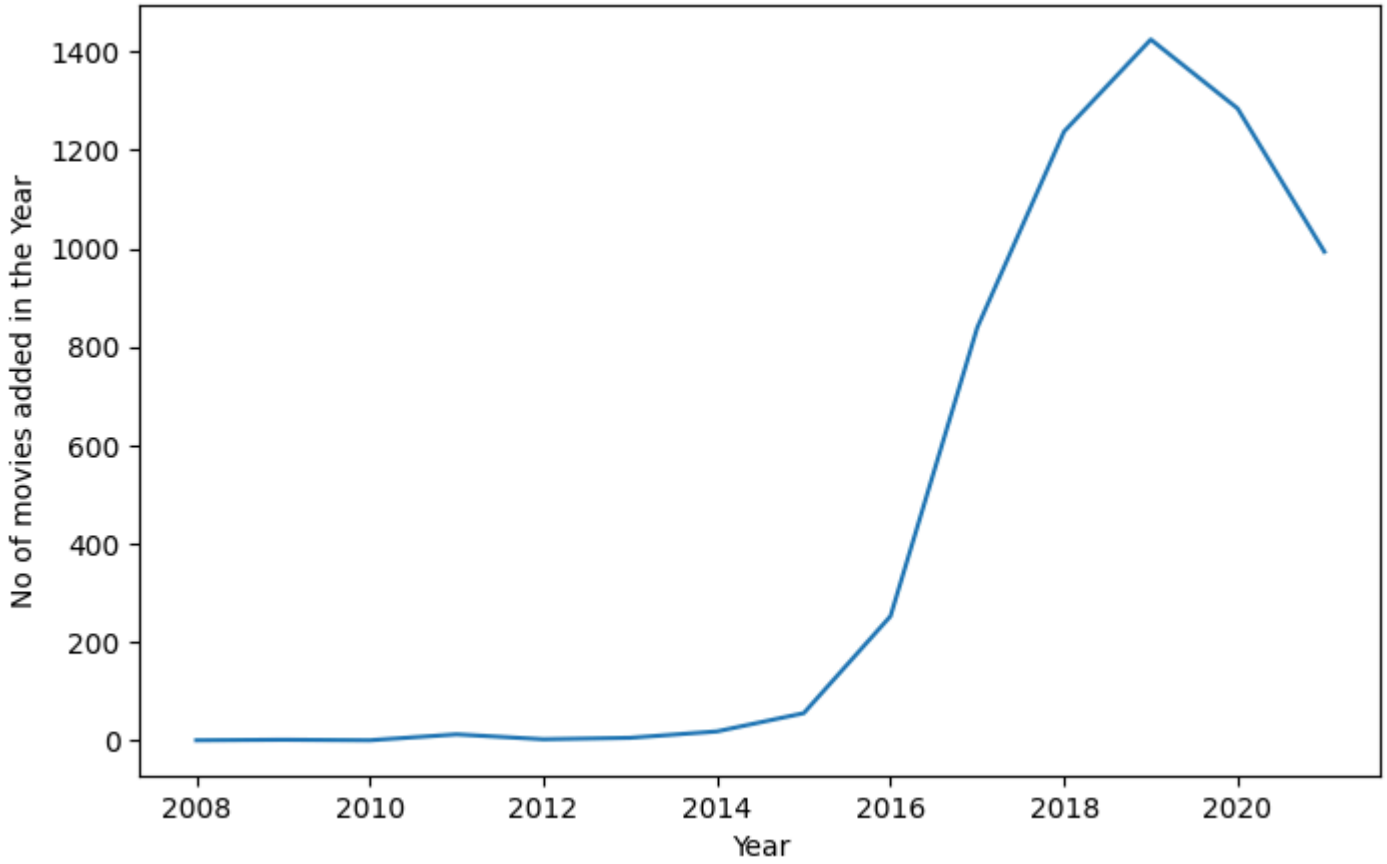
Out[ ]:

year_added	title
2019.0	1424
2020.0	1284
2018.0	1237
2021.0	993

year_added	title
2017.0	839

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
sns.lineplot(data=df_year_added, x='year_added', y='title')
plt.ylabel("No of movies added in the Year")
plt.xlabel("Year")
plt.show()
```



- Here you can see that no of Movies added on the Netflix peaked at 2019.
- Movies added continuously increased exponentially after 2014 untill 2019.
- started come down after 2019 onwards

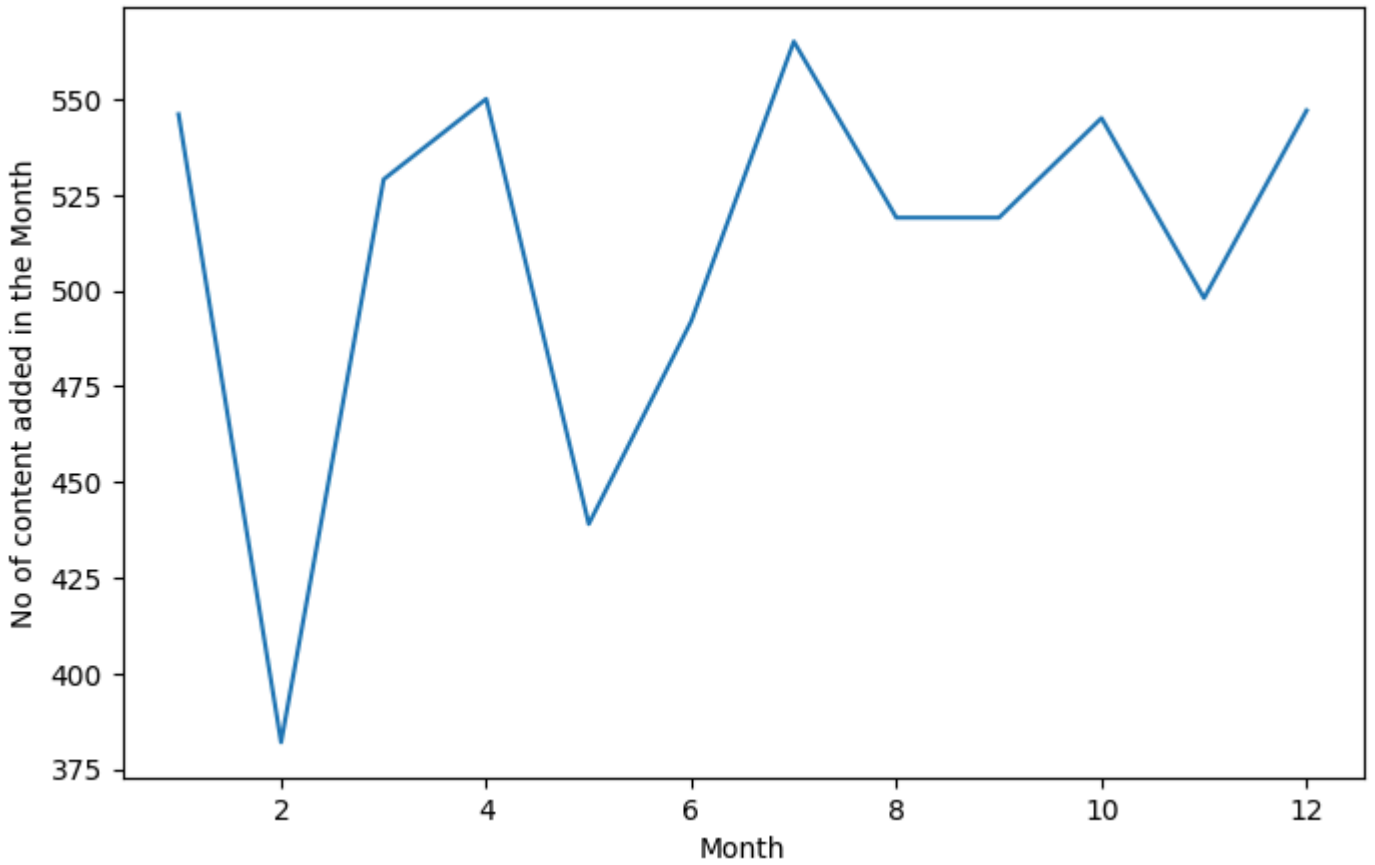
Movies by Month added

In [ ]:
```
df_month_added= df_movies.groupby('month_added')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_month_added
```

Out[ ]:

month_added	title
7.0	565
4.0	550
12.0	547
1.0	546
10.0	545
3.0	529
8.0	519
9.0	519
11.0	498
6.0	492
5.0	439
2.0	382

In [ ]:
```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
sns.lineplot(data=df_month_added, x='month_added', y='title')
plt.ylabel("No of content added in the Month")
plt.xlabel("Month")
plt.show()
```



- Most of the movies are added in the 1st, 7th and 12th month

Movies by Week added

In [ ]:
```
df_week_added= df_movies.groupby('week_added')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_week_added.head()
```

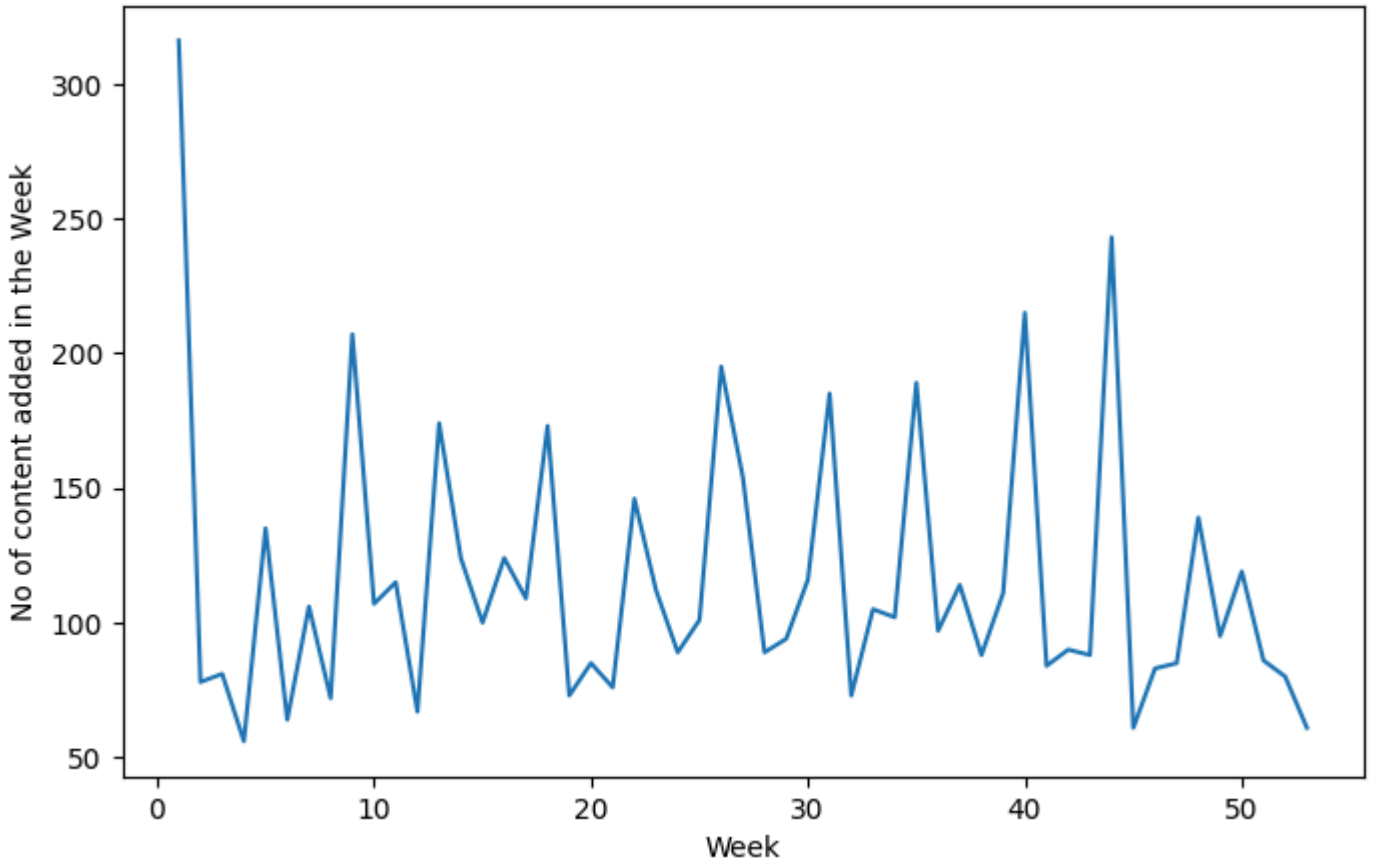
Out[ ]:

week_added	title
1	316
44	243

week_added	title
40	215
9	207
26	195

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
sns.lineplot(data=df_week_added, x='week_added', y='title')
plt.ylabel("No of content added in the Week")
plt.xlabel("Week")
plt.show()
```



- Most of the movies are added in 1st week

Movies by Release Year

In [ ]:
```
df_year_released= df_movies.groupby('release_year')[['title']].nunique().sort_values(by =['title'], ascending = False)
df_year_released.head()
```

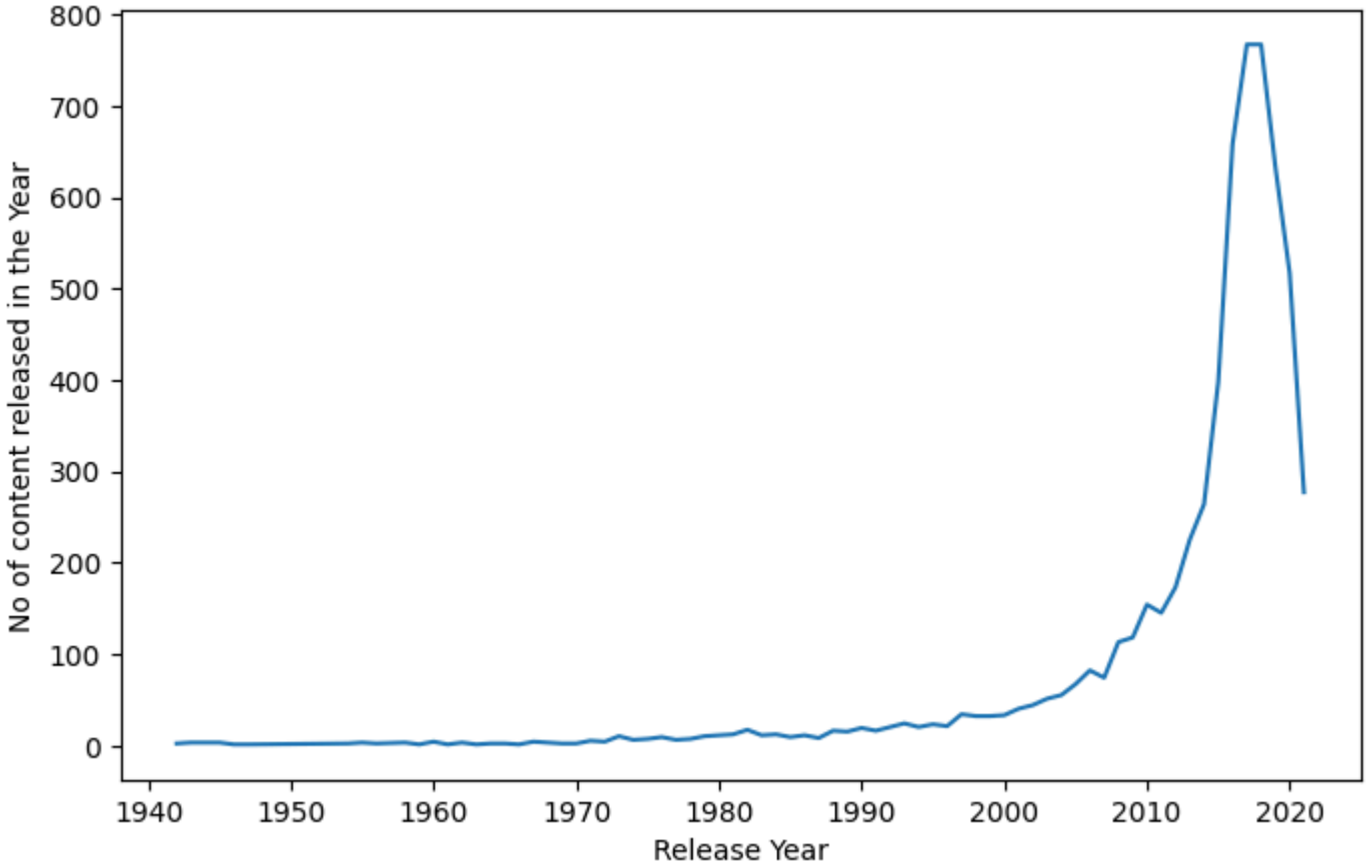
Out[ ]:

release_year	title
2018	767
2017	767

release_year	title
2016	658
2019	633
2020	517

In [ ]:

```
#let plot this in horizonal bar graph
plt.figure(figsize=(8,5))
sns.lineplot(data= df_year_released, x='release_year', y='title')
plt.ylabel("No of content released in the Year")
plt.xlabel("Release Year")
plt.show()
```



- Most of the movies avaialble on Netflix is released only in 2018.
- movie release picked up after till 2018 and then started coming down.
- decrease in the movie release can be due to COVID after 2020 onwards due to lockdown n all.

How many days after the movie is added after it is released.

In [ ]:
```
df_final.head()
```

Out[ ]:

	title	director	cast	country	listed_in	show_id	type	date_added	release_year	rating	duration	description	duration1	year_
0	Dick Johnson Is Dead	Kirsten Johnson	Unknown Actor	United States	Documentaries	s1	Movie	2021-09-25	2020	PG-13	80-100	As her father nears the end of his life, filmm...	80-100	
1	Blood & Water	Unknown Director	Ama Qamata	South Africa	International TV Shows	s2	TV Show	2021-09-24	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...	<1	
2	Blood & Water	Unknown Director	Ama Qamata	South Africa	TV Dramas	s2	TV Show	2021-09-24	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...	<1	
3	Blood & Water	Unknown Director	Ama Qamata	South Africa	TV Mysteries	s2	TV Show	2021-09-24	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...	<1	
4	Blood & Water	Unknown Director	Khosi Ngema	South Africa	International TV Shows	s2	TV Show	2021-09-24	2021	TV-MA	2 Seasons	After crossing paths at a party, a Cape Town t...	<1	

In [ ]:

```
df_final['diff'] = df_final['year_added']- df_final['release_year']
df_movie = df_final[df_final['type'] == 'Movie']
df_movie.head()
```

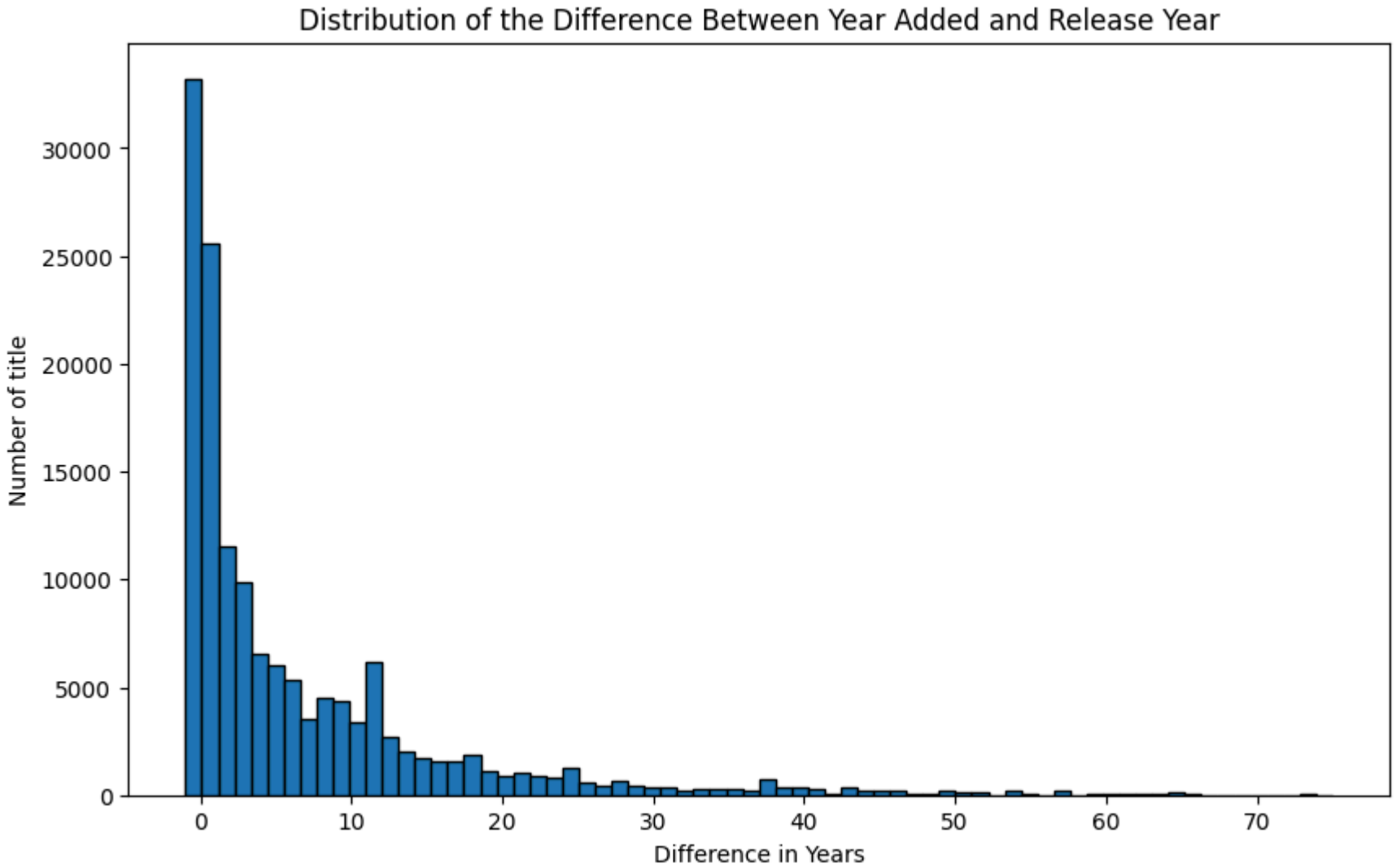
Out[ ]:

	title	director	cast	country	listed_in	show_id	type	date_added	release_year	rating	duration	description	duration1
0	Dick Johnson Is	Kirsten Johnson	Unknown Actor	United States	Documentaries	s1	Movie	2021-09-25	2020	PG-13	80-100	As her father nears	80-100

	title	director	cast	country	listed_in	show_id	type	date_added	release_year	rating	duration	description	duration1
	Dead											the end of his life, filmm...	
159	My Little Pony: A New Generation	Robert Cullen	Vanessa Hudgens	Unknown Country	Children & Family Movies	s7	Movie	2021-09-24	2021	PG	80-100	Equestria's divided. But a bright-eyed hero be...	80-100
160	My Little Pony: A New Generation	Robert Cullen	Kimiko Glenn	Unknown Country	Children & Family Movies	s7	Movie	2021-09-24	2021	PG	80-100	Equestria's divided. But a bright-eyed hero be...	80-100
161	My Little Pony: A New Generation	Robert Cullen	James Marsden	Unknown Country	Children & Family Movies	s7	Movie	2021-09-24	2021	PG	80-100	Equestria's divided. But a bright-eyed hero be...	80-100
162	My Little Pony: A New Generation	Robert Cullen	Sofia Carson	Unknown Country	Children & Family Movies	s7	Movie	2021-09-24	2021	PG	80-100	Equestria's divided. But a bright-eyed hero be...	80-100

In [ ]:

```
plt.figure(figsize=(10, 6))
plt.hist(df_movie['diff'], bins=70, edgecolor='black')
plt.title('Distribution of the Difference Between Year Added and Release Year')
plt.xlabel('Difference in Years')
plt.ylabel('Number of title')
plt.show()
```



- From the histogram above we can see that most of the movies are added as soon they are released or within 1 year

word cloud on the genre columns to know which kind

of genre is produced

In [ ]:
```
!pip install wordcloud
from wordcloud import WordCloud
```

```
Requirement already satisfied: wordcloud in /usr/local/lib/python3.10/dist-packages (1.9.3)
Requirement already satisfied: numpy>=1.6.1 in /usr/local/lib/python3.10/dist-packages (from wordcloud) (1.25.2)
Requirement already satisfied: pillow in /usr/local/lib/python3.10/dist-packages (from wordcloud) (9.4.0)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (from wordcloud) (3.7.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud)
(1.2.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud) (0.
12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->wordclou
d) (4.51.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->wordclou
d) (1.4.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud)
(24.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud)
(3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib->wordcl
oud) (2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotl
ib->wordcloud) (1.16.0)
```

In [ ]:
```
# Filter out missing description values
genre_data = df_final['listed_in']

# Join all the cast names into a single string
genre_text = ' '.join(genre_data)

# Generate the word cloud
wordcloud = WordCloud(width=800, height=400, background_color='white').generate(genre_text)

# Create a figure and axes
fig, ax = plt.subplots(figsize=(15, 10))

# Display the word cloud
ax.imshow(wordcloud, interpolation='bilinear')
ax.set_title('Word Cloud of Netflix Content Genre')
ax.axis('off')
```

```
# Show the plot
plt.show()
```