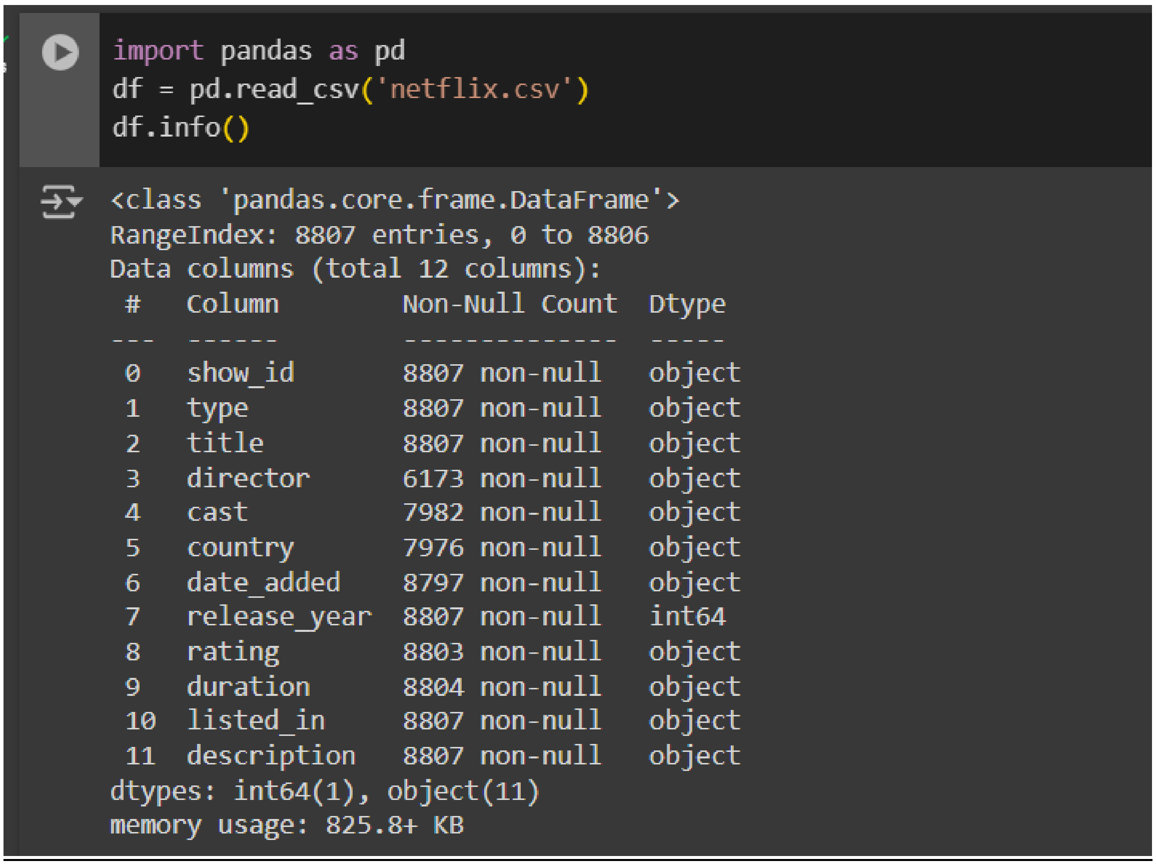
Netflix - Data Exploration and Visualisation (Collab link: Netflix_Data Analysis)

Dataset: It consists of movies and shows that are available in Netflix. Cast & Directors of those movies & shows and countries where the movies or shows produced in. Movies & Tv Shows released date and the date they got added into the Netflix. Genre & Ratings of them.

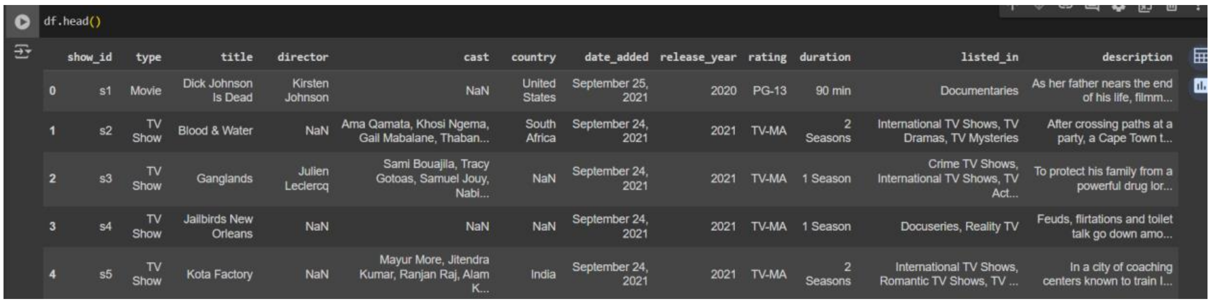
Dataset Info:



```python
import pandas as pd
df = pd.read_csv('netflix.csv')
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
```

Dataset:



```python
df.head()
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG-13	90 min	Documentaries	As her father nears the end of his life, filmm...
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	2021	TV-MA	2 Seasons	International TV Shows, TV Dramas, TV Mysteries	After crossing paths at a party, a Cape Town t...
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	September 24, 2021	2021	TV-MA	1 Season	Crime TV Shows, International TV Shows, TV Act...	To protect his family from a powerful drug lor...
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	NaN	September 24, 2021	2021	TV-MA	1 Season	Docuseries, Reality TV	Feuds, flirtations and toilet talk go down amo...
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	September 24, 2021	2021	TV-MA	2 Seasons	International TV Shows, Romantic TV Shows, TV ...	In a city of coaching centers known to train l...

From the dataset we can see that there are multiple values in a single cell, so we need to clean the data.

First, we will strip the data in the cells by using strip () method.

This will delete the trailing and leading spaces or commas.

Code to strip the required columns:

Code:

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

Now we dealt with leading and trailing spaces or special characters. We will split the multiple values in the cells using split () method.

Code to split the cells:

```
df['cast'] = df['cast'].apply(lambda x: x.split(',') if pd.notna(x)
else x)
df['listed_in'] = df['listed_in'].apply(lambda x: x.split(',') if
pd.notna(x)  else x)
df['director'] = df['director'].apply(lambda x: x.split(',') if
pd.notna(x)  else x)
df['country'] = df['country'].apply(lambda x: x.split(',') if
pd.notna(x)  else x)
```

We used apply () function to apply the split method to each row and we used notna () function to filter the Nan or Null values

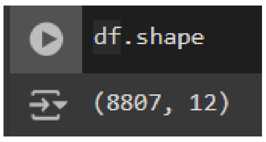
```
df['cast'] = df['cast'].apply(lambda x: x.split(',') if pd.notna(x)
else x)
df['listed_in'] = df['listed_in'].apply(lambda x: x.split(',') if
pd.notna(x)  else x)
df['director'] = df['director'].apply(lambda x: x.split(',') if
pd.notna(x)  else x)
df['country'] = df['country'].apply(lambda x: x.split(',') if
pd.notna(x)  else x)
```

Now this split method will create list of elements in each cell, we can use explode () function to create a separate row for each value in the list.

Code:

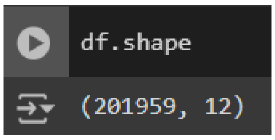
```
df = df.explode('cast')
df = df.explode('listed_in')
df = df.explode('country')
df = df.explode('director')
```

Original Dataset shape:



```
df.shape
(8807, 12)
```

After using explode ():



```
df.shape
```

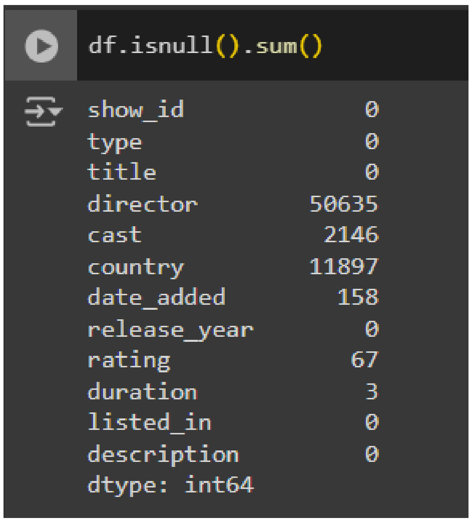
```
(201959, 12)
```

We need to strip the data as after performing explode () there may be some trailing or leading spaces.

Code:

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

As we can see there are null values in few columns, we need to deal with Null values.



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

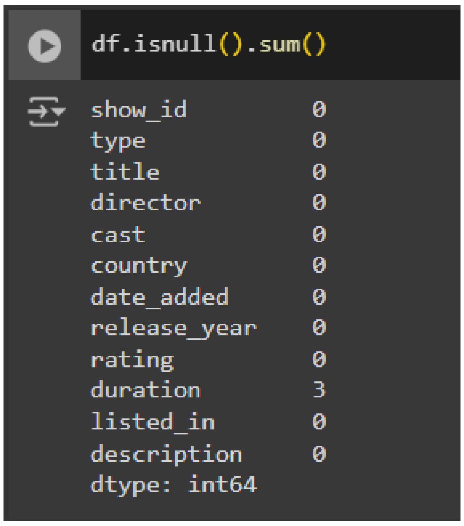
```
show_id             0
type                0
title               0
director        50635
cast             2146
country         11897
date_added        158
release_year        0
rating             67
duration            3
listed_in           0
description         0
dtype: int64
```

We will fill the null values using fillna () function.

Code:

```
df['director'] = df['director'].fillna('Unknown Director')
df['cast'] = df['cast'].fillna('Unknown Cast')
df['country'] = df['country'].fillna('Unknown Country')
df['date_added'] = df['date_added'].fillna(0)
df['rating'] = df['rating'].fillna('Unknown rating')
```

After performing **Data Imputation**

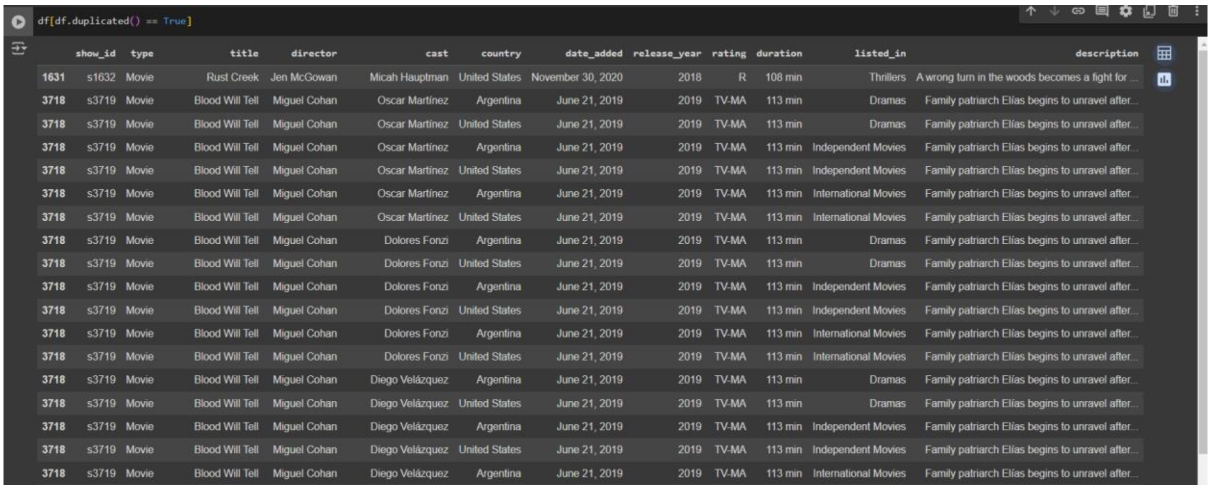


```
df.isnull().sum()
```

```
show_id         0
type            0
title           0
director        0
cast            0
country         0
date_added      0
release_year    0
rating          0
duration        3
listed_in       0
description     0
dtype: int64
```

We can see that there are no null values anymore

From the below image we can see that there are 55 duplicated rows



```
df[df.duplicated() == True]
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
1631	s1632	Movie	Rust Creek	Jen McGowan	Micah Hauptman	United States	November 30, 2020	2018	R	108 min	Thrillers	A wrong turn in the woods becomes a fight for ...
3718	s3719	Movie	Blood Will Tell	Miguel Cohan	Oscar Martínez	Argentina	June 21, 2019	2019	TV-MA	113 min	Dramas	Family patriarch Elías begins to unravel after...
3718	s3719	Movie	Blood Will Tell	Miguel Cohan	Oscar Martínez	United States	June 21, 2019	2019	TV-MA	113 min	Dramas	Family patriarch Elías begins to unravel after...
3718	s3719	Movie	Blood Will Tell	Miguel Cohan	Oscar Martínez	Argentina	June 21, 2019	2019	TV-MA	113 min	Independent Movies	Family patriarch Elías begins to unravel after...
3718	s3719	Movie	Blood Will Tell	Miguel Cohan	Oscar Martínez	United States	June 21, 2019	2019	TV-MA	113 min	Independent Movies	Family patriarch Elías begins to unravel after...
3718	s3719	Movie	Blood Will Tell	Miguel Cohan	Oscar Martínez	Argentina	June 21, 2019	2019	TV-MA	113 min	International Movies	Family patriarch Elías begins to unravel after...
3718	s3719	Movie	Blood Will Tell	Miguel Cohan	Oscar Martínez	United States	June 21, 2019	2019	TV-MA	113 min	International Movies	Family patriarch Elías begins to unravel after...
3718	s3719	Movie	Blood Will Tell	Miguel Cohan	Dolores Fonzi	Argentina	June 21, 2019	2019	TV-MA	113 min	Dramas	Family patriarch Elías begins to unravel after...
3718	s3719	Movie	Blood Will Tell	Miguel Cohan	Dolores Fonzi	United States	June 21, 2019	2019	TV-MA	113 min	Dramas	Family patriarch Elías begins to unravel after...
3718	s3719	Movie	Blood Will Tell	Miguel Cohan	Dolores Fonzi	Argentina	June 21, 2019	2019	TV-MA	113 min	Independent Movies	Family patriarch Elías begins to unravel after...
3718	s3719	Movie	Blood Will Tell	Miguel Cohan	Dolores Fonzi	United States	June 21, 2019	2019	TV-MA	113 min	Independent Movies	Family patriarch Elías begins to unravel after...
3718	s3719	Movie	Blood Will Tell	Miguel Cohan	Dolores Fonzi	Argentina	June 21, 2019	2019	TV-MA	113 min	International Movies	Family patriarch Elías begins to unravel after...
3718	s3719	Movie	Blood Will Tell	Miguel Cohan	Dolores Fonzi	United States	June 21, 2019	2019	TV-MA	113 min	International Movies	Family patriarch Elías begins to unravel after...
3718	s3719	Movie	Blood Will Tell	Miguel Cohan	Diego Velázquez	Argentina	June 21, 2019	2019	TV-MA	113 min	Dramas	Family patriarch Elías begins to unravel after...
3718	s3719	Movie	Blood Will Tell	Miguel Cohan	Diego Velázquez	United States	June 21, 2019	2019	TV-MA	113 min	Dramas	Family patriarch Elías begins to unravel after...
3718	s3719	Movie	Blood Will Tell	Miguel Cohan	Diego Velázquez	Argentina	June 21, 2019	2019	TV-MA	113 min	Independent Movies	Family patriarch Elías begins to unravel after...
3718	s3719	Movie	Blood Will Tell	Miguel Cohan	Diego Velázquez	United States	June 21, 2019	2019	TV-MA	113 min	Independent Movies	Family patriarch Elías begins to unravel after...
3718	s3719	Movie	Blood Will Tell	Miguel Cohan	Diego Velázquez	Argentina	June 21, 2019	2019	TV-MA	113 min	International Movies	Family patriarch Elías begins to unravel after...

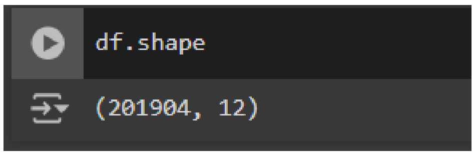
We need to drop these rows to perform a precise analysis

Code:

```
df.drop_duplicates(inplace = True,keep = 'first')
```

We used inplace = True and keep = 'first' properties to keep the first duplicated row and make the changes permanent.

After performing Data Cleaning our dataset shape is



```
df.shape
```

```
(201904, 12)
```

```
df.reset_index(drop = True, inplace = True)
```

➔ After performing explode operation rows got duplicated and explicit index is repeating so we need to reset the index to get the index in order. We are drop = True to drop the old index.

Exploratory Data Analysis

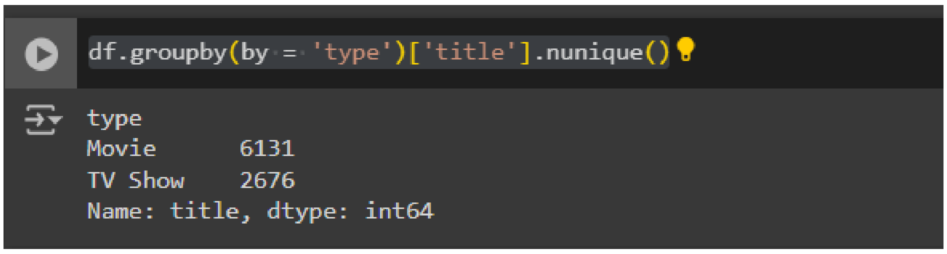
Values counts of each category:

Type Columns:

Code:

```
df.groupby(by = 'type')['title'].nunique()
```

Output:



```
df.groupby(by = 'type')['title'].nunique()
```

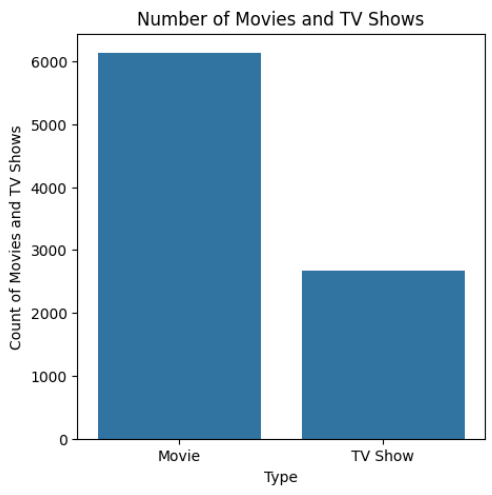
```
type
Movie       6131
TV Show     2676
Name: title, dtype: int64
```

Visualization:

Code:

```
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize = (5,5))
sns.barplot(x= df.groupby(by = 'type')['title'].nunique().index, y=
df.groupby(by = 'type')['title'].nunique().values)
plt.title('Number of Movies and TV Shows')
plt.xlabel('Type')
plt.ylabel('Count of Movies and TV Shows')
plt.show()
```

Output:



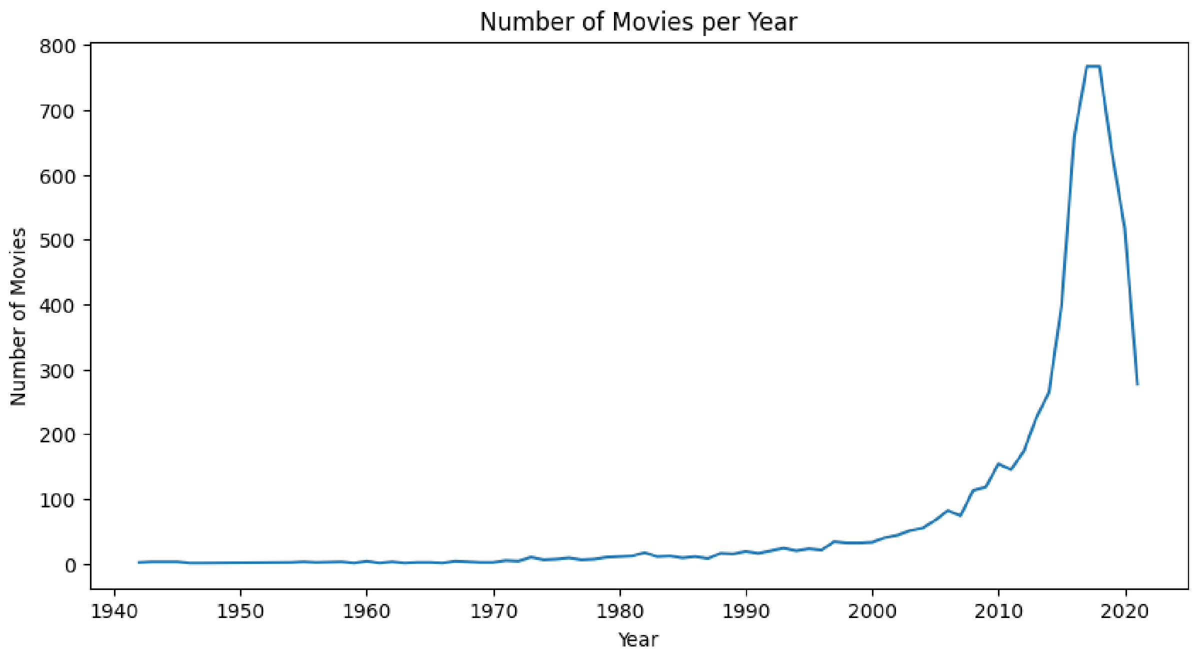
Insights: From the bar graph we can clearly derive that no of movies produced by Netflix are greater than TV shows.

No of movies released per year

Code:

```
plt.figure(figsize = (10,5))
sns.lineplot(x = Movies.groupby(by
='release_year')['title'].nunique().index, y = Movies.groupby(by
='release_year')['title'].nunique().values)
plt.title('Number of Movies per Year')
plt.xlabel('Year')
plt.ylabel('Number of Movies')
plt.show()
```

Graph: (Trend analysis)



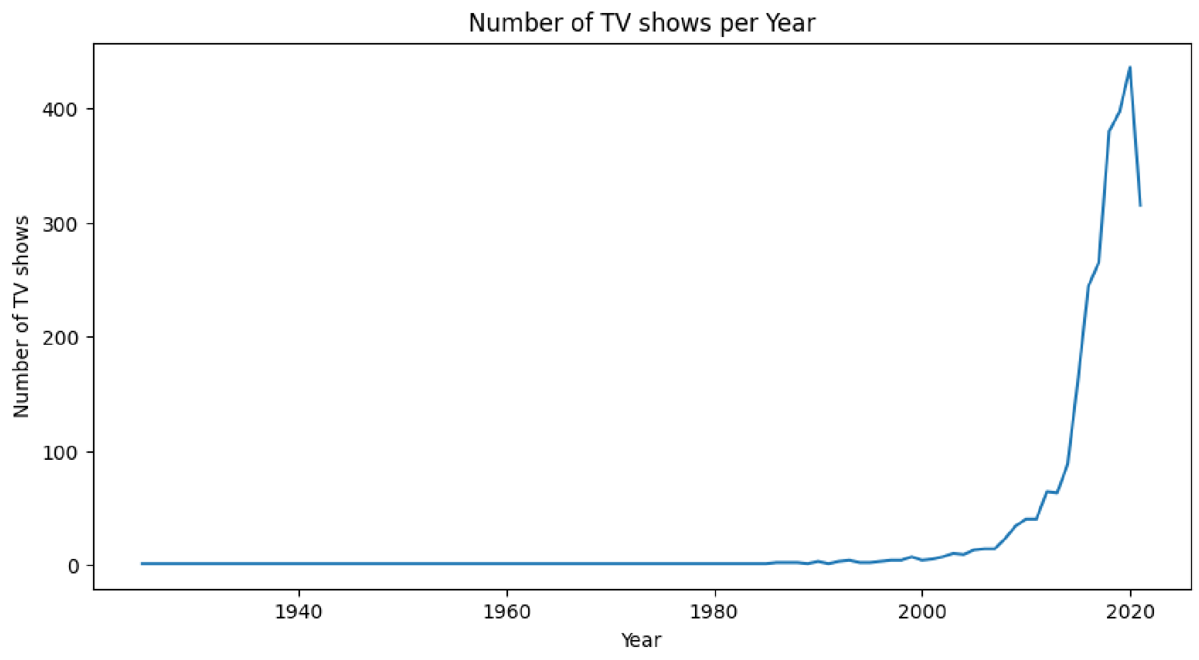
Insights: From the graph we can see the rapid increase in no of movies released per year.

Number of TV Shows released per year

Code:

```
plt.figure(figsize = (10,5))
sns.lineplot(x = TV_shows.groupby(by
='release_year')['title'].nunique().index, y = TV_shows.groupby(by
='release_year')['title'].nunique().values)
plt.title('Number of TV shows per Year')
plt.xlabel('Year')
plt.ylabel('Number of TV shows')
plt.show()
```

Graph:



Insights: From the graph we can derive that no of TV shows released per year increased rapidly every year.

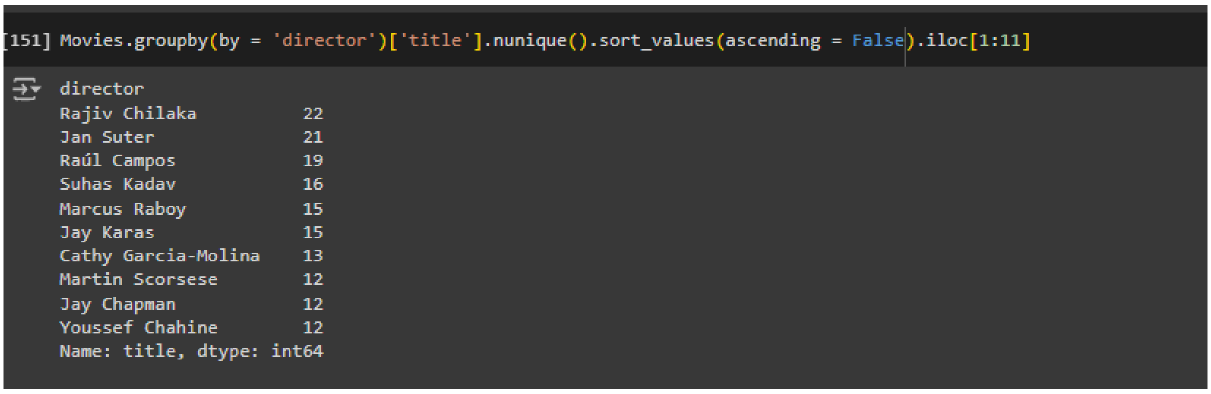
Non-Graphical Data Analysis:

Top 10 Directors who directed the most n.o of movies

Code:

```
Movies.groupby(by =
'director')['title'].nunique().sort_values(ascending =
False).iloc[1:11]
```

Output:



```
[151] Movies.groupby(by = 'director')['title'].nunique().sort_values(ascending = False).iloc[1:11]

director
Rajiv Chilaka          22
Jan Suter              21
Raúl Campos            19
Suhas Kadav            16
Marcus Raboy           15
Jay Karas              15
Cathy Garcia-Molina    13
Martin Scorsese        12
Jay Chapman            12
Youssef Chahine        12
Name: title, dtype: int64
```

Insights: From the output we can find the list of directors who directed the most movies. Rajiv Chilaka has directed most no of movies

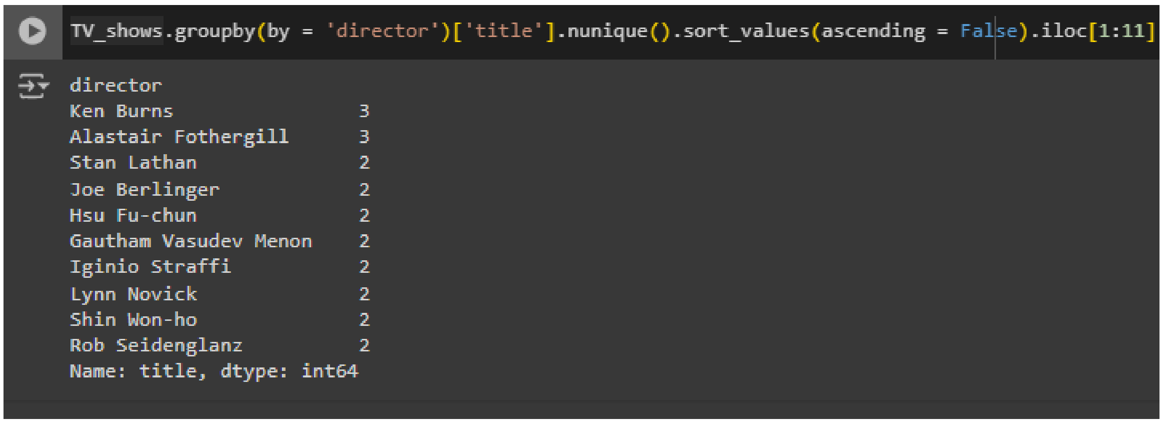
---

Top 10 Directors who directed the most n.o of TV shows

Code:

```
TV_shows.groupby(by =
'director')['title'].nunique().sort_values(ascending =
False).iloc[1:11]
```

Output:



```
TV_shows.groupby(by = 'director')['title'].nunique().sort_values(ascending = False).iloc[1:11]

director
Ken Burns                 3
Alastair Fothergill       3
Stan Lathan               2
Joe Berlinger             2
Hsu Fu-chun               2
Gautham Vasudev Menon     2
Iginio Straffi            2
Lynn Novick               2
Shin Won-ho               2
Rob Seidenglanz           2
Name: title, dtype: int64
```

Insights: Ken Burns and Alastair Fothergill are the directors who directed most TV shows.

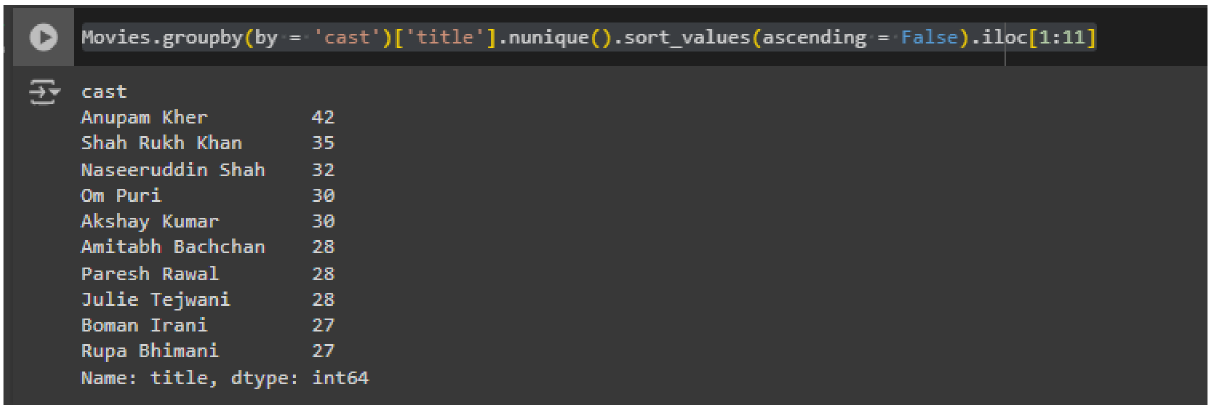
Top 10 actors who have appeared in most movies

Movies:

Code:

```
Movies.groupby(by = 'cast')['title'].nunique().sort_values(ascending =
False).iloc[1:11]
```

Output:



```
Movies.groupby(by = 'cast')['title'].nunique().sort_values(ascending = False).iloc[1:11]

cast
Anupam Kher          42
Shah Rukh Khan       35
Naseeruddin Shah     32
Om Puri              30
Akshay Kumar         30
Amitabh Bachchan     28
Paresh Rawal         28
Julie Tejwani        28
Boman Irani          27
Rupa Bhimani         27
Name: title, dtype: int64
```

Insights: From the output we can see top 10 casted actors in movies.

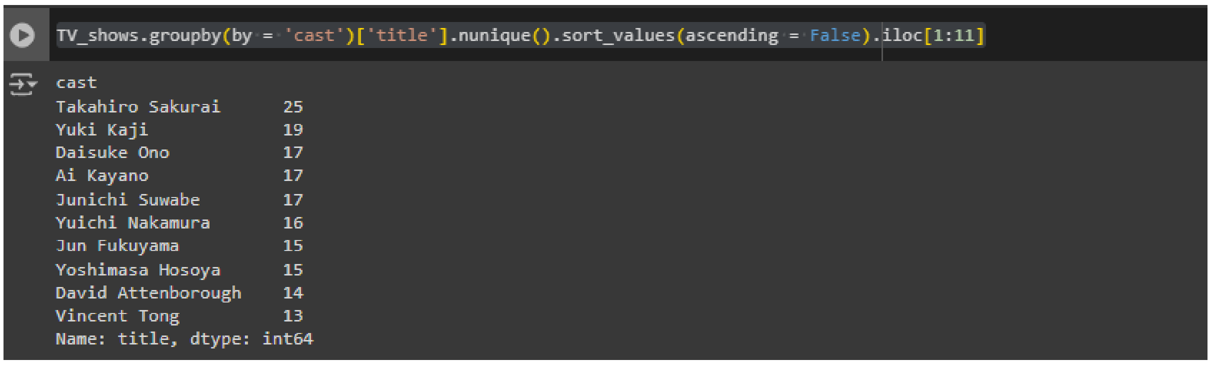
Recommendations: Anupam Kher is the most casted actor, he must be popular so we can add collection of top actors in our platform so viewers can directly select his movies.

Top 10 actors who have appeared in most movies

Code:

```
TV_shows.groupby(by = 'cast')['title'].nunique().sort_values(ascending
= False).iloc[1:11]
```

Output:



```
TV_shows.groupby(by = 'cast')['title'].nunique().sort_values(ascending = False).iloc[1:11]
```

```
cast
Takahiro Sakurai      25
Yuki Kaji             19
Daisuke Ono           17
Ai Kayano             17
Junichi Suwabe        17
Yuichi Nakamura       16
Jun Fukuyama          15
Yoshimasa Hosoya      15
David Attenborough    14
Vincent Tong          13
Name: title, dtype: int64
```

Insights: From the output we can see top 10 casted actors in TV shows.

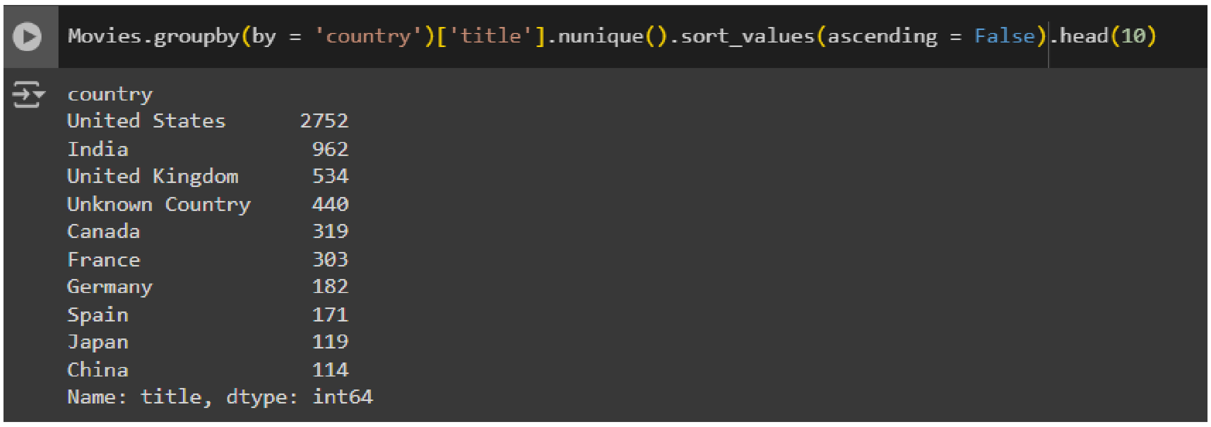
Recommendations: Takahiro Sakurai is the most casted actor, he must be popular so we can add collection of top actors in our platform so viewers can directly select his TV shows.

Top 10 countries with most n.o of Movies produced

Code:

```
Movies = df[df['type'] == 'Movie']
Movies.groupby(by = 'country')['title'].nunique().sort_values(ascending
= False).head(10)
```

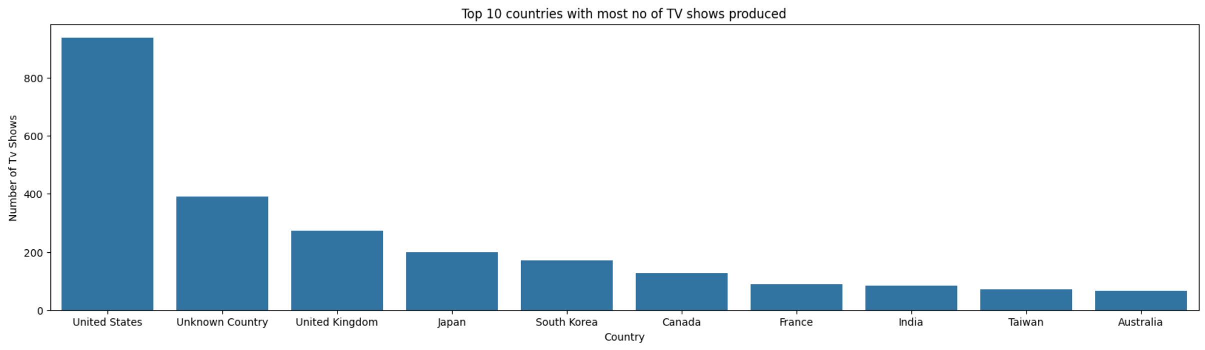
Output:



```
Movies.groupby(by = 'country')['title'].nunique().sort_values(ascending = False).head(10)
```

```
country
United States      2752
India               962
United Kingdom      534
Unknown Country     440
Canada              319
France              303
Germany             182
Spain               171
Japan               119
China               114
Name: title, dtype: int64
```

Graphical Representation:



Code:

```
plt.figure(figsize = (20,5))
sns.barplot(x = TV_shows.groupby(by =
'country')['title'].nunique().sort_values(ascending =
False).head(10).index, y = TV_shows.groupby(by =
'country')['title'].nunique().sort_values(ascending =
False).head(10).values)
plt.xlabel('Country')
plt.ylabel('Number of Tv Shows')
plt.title('Top 10 countries with most no of TV shows produced')
plt.show()
```

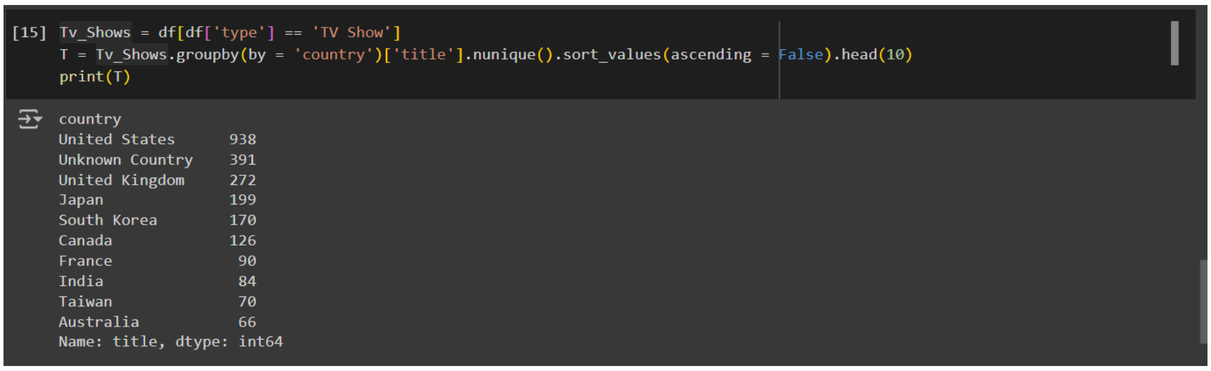
Insights: From the graph we can clearly see that united states has produced most no of movies.

Top 10 Countries with most n.o of TV shows produced

Code:

```
Tv_Shows = df[df['type'] == 'TV Show']
Tv_Shows.groupby(by =
'country')['title'].nunique().sort_values(ascending = False).head(10)
```

Output:



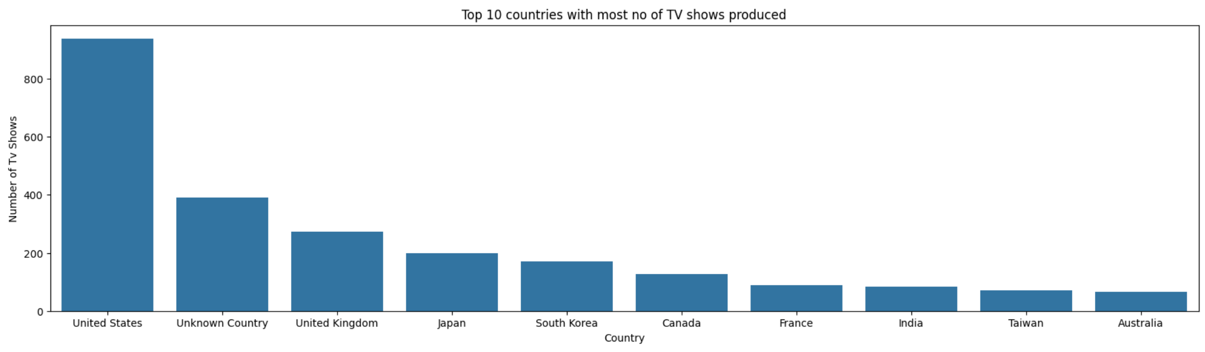
```
[15] Tv_Shows = df[df['type'] == 'TV Show']
     T = Tv_Shows.groupby(by = 'country')['title'].nunique().sort_values(ascending = False).head(10)
     print(T)

     country
     United States      938
     Unknown Country    391
     United Kingdom     272
     Japan              199
     South Korea        170
     Canada             126
     France              90
     India               84
     Taiwan              70
     Australia           66
     Name: title, dtype: int64
```

Graphical Analysis:

Code:

```
plt.figure(figsize = (20,5))
sns.barplot(x = TV_shows.groupby(by =
'country')['title'].nunique().sort_values(ascending =
False).head(10).index, y = TV_shows.groupby(by =
'country')['title'].nunique().sort_values(ascending =
False).head(10).values)
plt.xlabel('Country')
plt.ylabel('Number of Tv Shows')
plt.title('Top 10 countries with most no of TV shows produced')
plt.show()
```

Graph:

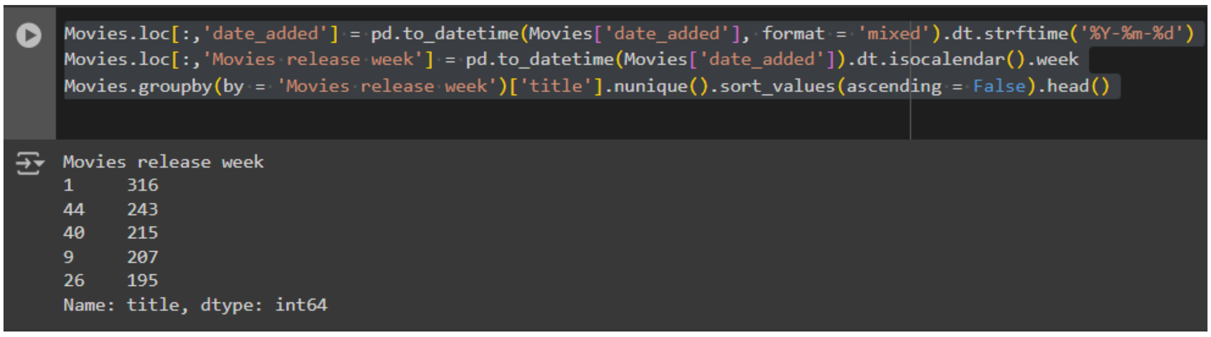


<u>**Week-wise movie count**</u>

Code:

```
Movies.loc[:,'date_added'] = pd.to_datetime(Movies['date_added'],
format = 'mixed').dt.strftime('%Y-%m-%d')
Movies.loc[:,'Movies release week'] =
pd.to_datetime(Movies['date_added']).dt.isocalendar().week
Movies.groupby(by = 'Movies release
week')['title'].nunique().sort_values(ascending = False).head()
```

Output:



```
Movies.loc[:,'date_added'] = pd.to_datetime(Movies['date_added'], format = 'mixed').dt.strftime('%Y-%m-%d')
Movies.loc[:,'Movies release week'] = pd.to_datetime(Movies['date_added']).dt.isocalendar().week
Movies.groupby(by = 'Movies release week')['title'].nunique().sort_values(ascending = False).head()
```

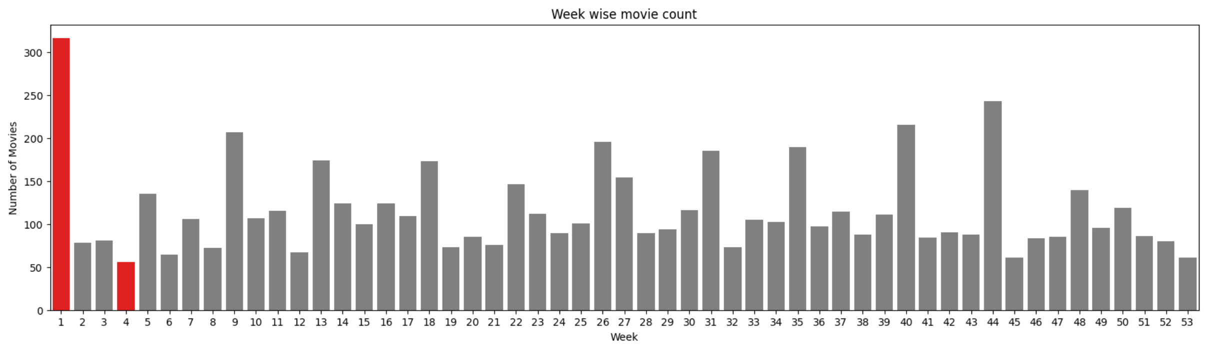
```
Movies release week
1     316
44    243
40    215
9     207
26    195
Name: title, dtype: int64
```

Graphical representation of week-wise movie count

Code:

```
plt.figure(figsize = (20,5))
wwmd = pd.DataFrame(Movies.groupby(by = 'Movies release
week')['title'].nunique()).reset_index()
wwmd.rename(columns = {'title':'count'},inplace = True)
Movie_max_value = wwmd['count'].idxmax()
Movie_min_value = wwmd['count'].idxmin()
colors = ['gray' if i != Movie_max_value else 'red'for i in
range(len(wwmd))]
sns.barplot(x = Movies.groupby(by = 'Movies release
week')['title'].nunique().index, y = Movies.groupby(by = 'Movies
release week')['title'].nunique().values, palette = colors)
plt.xlabel('Week')
plt.ylabel('Number of Movies')
plt.title('Week wise movie count')
plt.show()
```

Graph:



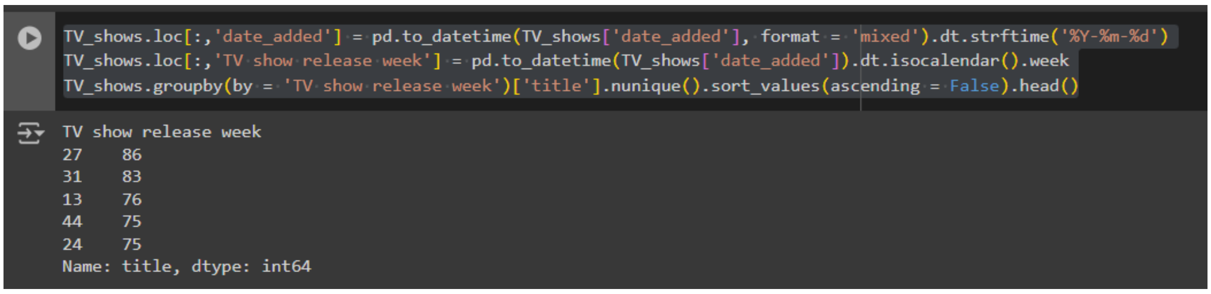
Insights: From the graph we can clearly see that most of no of movies were added to the Netflix during 1st and least no of movies were added during 4th week

TV Shows:

Code:

```
TV_shows.loc[:,'date_added'] = pd.to_datetime(TV_shows['date_added'],
format = 'mixed').dt.strftime('%Y-%m-%d')
TV_shows.loc[:,'TV show release week'] =
pd.to_datetime(TV_shows['date_added']).dt.isocalendar().week
TV_shows.groupby(by = 'TV show release
week')['title'].nunique().sort_values(ascending = False).head()
```

Output:



```
TV_shows.loc[:,'date_added'] = pd.to_datetime(TV_shows['date_added'], format = 'mixed').dt.strftime('%Y-%m-%d')
TV_shows.loc[:,'TV show release week'] = pd.to_datetime(TV_shows['date_added']).dt.isocalendar().week
TV_shows.groupby(by = 'TV show release week')['title'].nunique().sort_values(ascending = False).head()
```

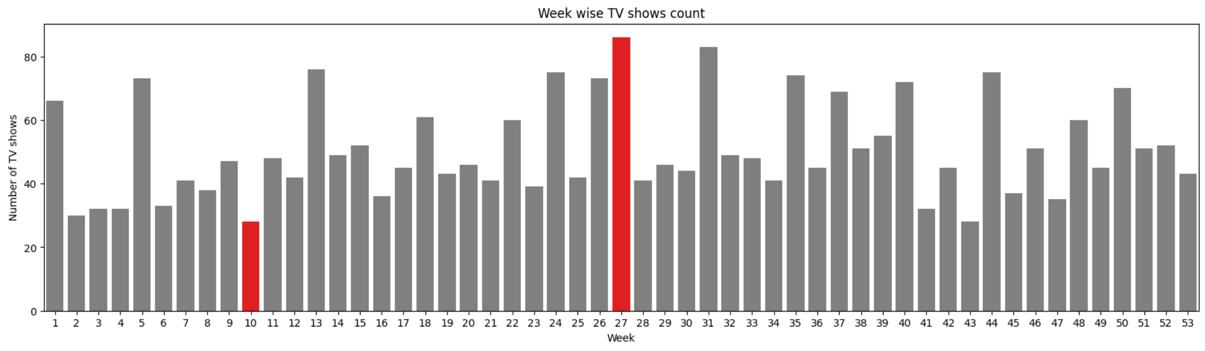
```
TV show release week
27    86
31    83
13    76
44    75
24    75
Name: title, dtype: int64
```

Graphical representation:

Code:

```
plt.figure(figsize = (20,5))
wwtsd = pd.DataFrame(TV_shows.groupby(by = 'TV show release
week')['title'].nunique()).reset_index()
wwtsd.rename(columns = {'title':'count'},inplace = True)
TV_showmax_value = wwtsd['count'].idxmax()
TV_showmin_value = wwtsd['count'].idxmin()
colors = ['gray' if (x!= TV_showmax_value and x!= TV_showmin_value)
else 'red' for x in range(len(wwtsd))]
sns.barplot(x = wwtsd['TV show release week'], y = wwtsd['count'],
palette = colors)
plt.xlabel('Week')
plt.ylabel('Number of TV shows')
plt.title('Week wise TV shows count')
plt.show()
```

Graph:



Insights: From the graph we can clearly see that most of no of TV shows were added to the Netflix during 27th week and least no of TV shows were added during 10th week

---

Find which is the best month to release the Tv-show or the movie. Do the analysis separately for Tv-shows and Movies

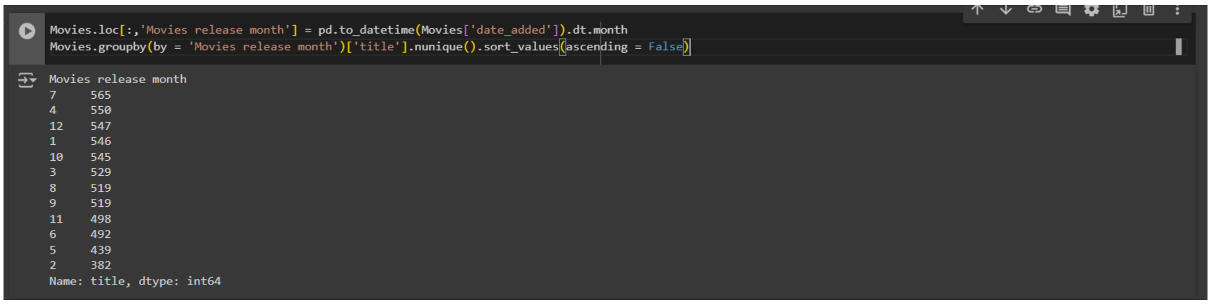
Movies

Code:

```
Movies.loc[:,'Movies release month'] =
pd.to_datetime(Movies['date_added']).dt.month
```

```
Movies.groupby(by = 'Movies release
month')['title'].nunique().sort_values(ascending = False)
```

Output:



```
Movies.loc[:,'Movies release month'] = pd.to_datetime(Movies['date_added']).dt.month
Movies.groupby(by = 'Movies release month')['title'].nunique().sort_values(ascending = False)
```

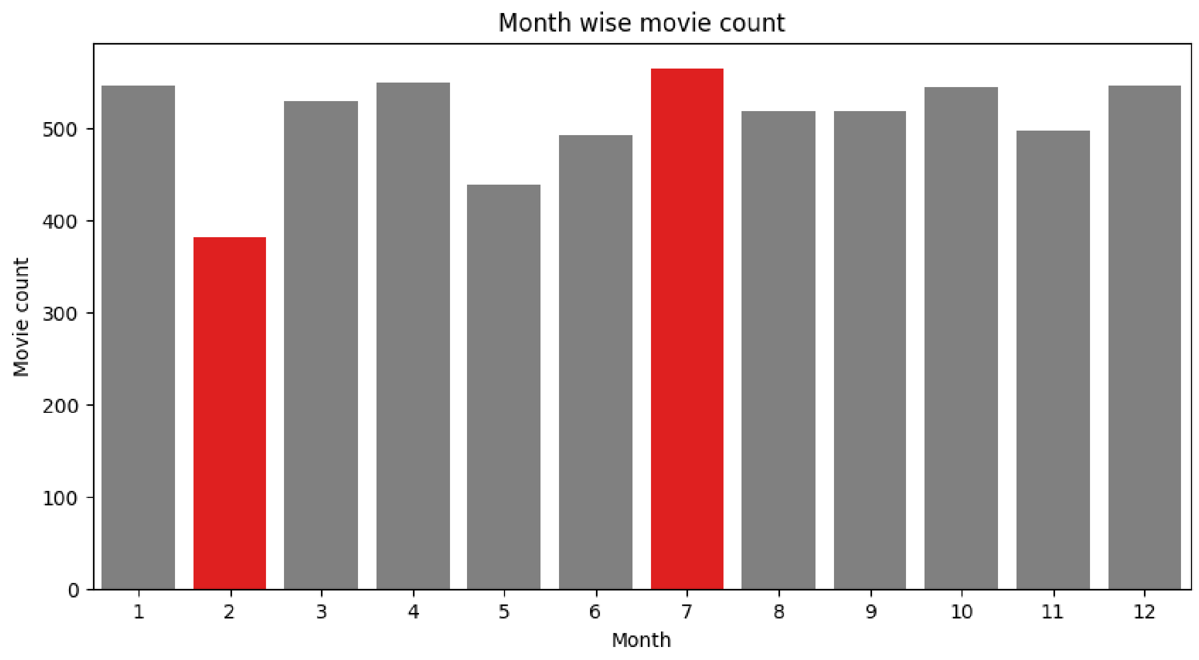
```
Movies release month
7     565
4     550
12    547
1     546
10    545
3     529
8     519
9     519
11    498
6     492
5     439
2     382
Name: title, dtype: int64
```

Graphical Representation:

Code:

```
plt.figure(figsize = (10,5))
mwmd = pd.DataFrame(Movies.groupby(by = 'Movies release
month')['title'].nunique()).reset_index()
mwmd.rename(columns = {'title':'count'},inplace = True)
Movie_month_max_value = mwmd['count'].idxmax()
Movie_month_min_value = mwmd['count'].idxmin()
colors = ['gray' if x != Movie_month_max_value and x!=
Movie_month_min_value else 'red' for x in range(len(mwmd))]
sns.barplot(x = Movies.groupby(by = 'Movies release
month')['title'].nunique().index, y = Movies.groupby(by = 'Movies
release month')['title'].nunique().values, palette = colors)
plt.title('Month wise movie count')
plt.xlabel('Month')
plt.ylabel('Movie count')
plt.show()
```

Graph:



Insights: From the graph we can clearly see that most of no of movies were added to the Netflix during 7th week and least no of movies were added during 2nd week.

Recommendations: As many movies are getting added during 7th month, we need to increase our marketing at least 1 or 2 weeks before this month to attract more customers to take our subscription.

2. As least no of movies are getting during 2nd month, we need to understand the reason behind it. If it's an unavoidable reason then we need to try to bring popular movies during 2nd month to increase the subscription count and to keep our existing customers.

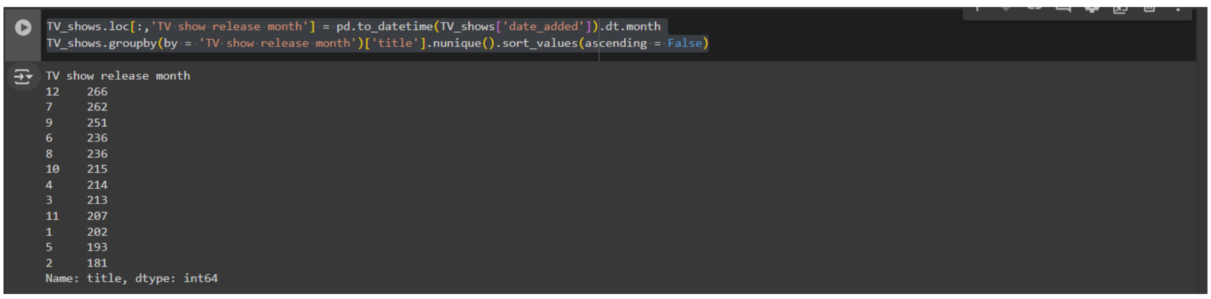
3. If we are not able to get the bid for popular movies then we need to offer 3months/6months at a discount price to maintain our viewership.

TV Shows:

Code:

```
TV_shows.loc[:,'TV show release month'] =
pd.to_datetime(TV_shows['date_added']).dt.month
TV_shows.groupby(by = 'TV show release
month')['title'].nunique().sort_values(ascending = False)
```

Output:



```
TV_shows.loc[:,'TV show release month'] = pd.to_datetime(TV_shows['date_added']).dt.month
TV_shows.groupby(by = 'TV show release month')['title'].nunique().sort_values(ascending = False)
```

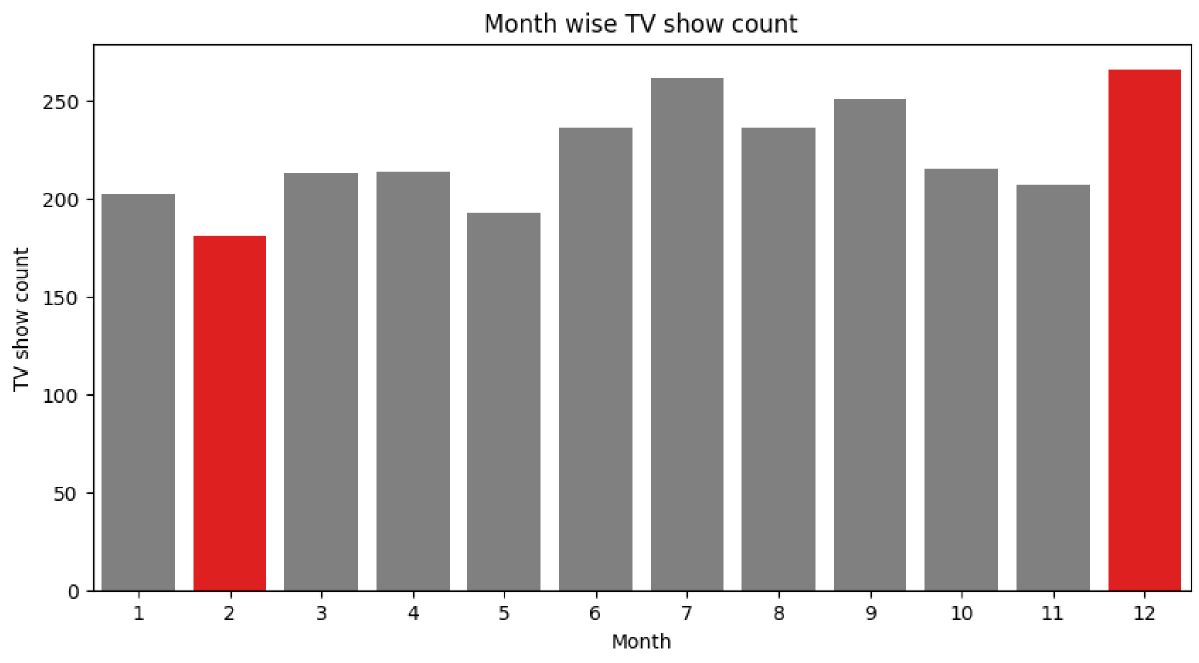
```
TV show release month
12    266
7     262
9     251
6     236
8     236
10    215
4     214
3     213
11    207
1     202
5     193
2     181
Name: title, dtype: int64
```

Graphical representation:

Code:

```
plt.figure(figsize = (10,5))
mwtsd = pd.DataFrame(TV_shows.groupby(by = 'TV show release
month')['title'].nunique()).reset_index()
mwtsd.rename(columns = {'title':'count'},inplace = True)
TV_month_max_value = mwtsd['count'].idxmax()
TV_month_min_value = mwtsd['count'].idxmin()
colors = ['gray' if x != TV_month_max_value and x!= TV_month_min_value
else 'red' for x in range(len(mwtsd))]
sns.barplot(x= TV_shows.groupby(by = 'TV show release
month')['title'].nunique().index, y = TV_shows.groupby(by = 'TV show
release month')['title'].nunique().values, palette = colors,legend =
False)
plt.title('Month wise TV show count')
plt.xlabel('Month')
plt.ylabel('TV show count')
plt.show()
```

Graph:



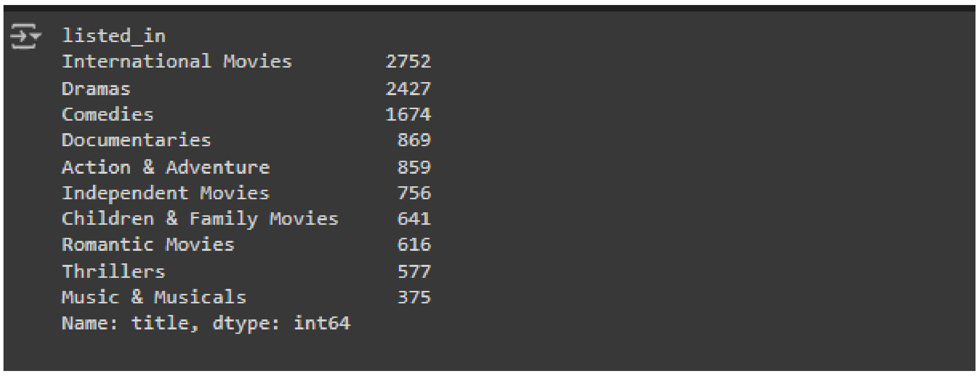
Insights: From the graph we can clearly see that most of no of movies were added to the Netflix during 12th week and least no of movies were added during 2nd week.

Top 10 most produced Movie genres

Code:

```
Movies.groupby(by =
'listed_in')['title'].nunique().sort_values(ascending =
False).iloc[0:10]
```

Output:



```
listed_in
International Movies        2752
Dramas                      2427
Comedies                    1674
Documentaries                869
Action & Adventure           859
Independent Movies           756
Children & Family Movies     641
Romantic Movies              616
Thrillers                    577
Music & Musicals             375
Name: title, dtype: int64
```

Insights: From the output we can derive that "international movies" are the most produced movies.

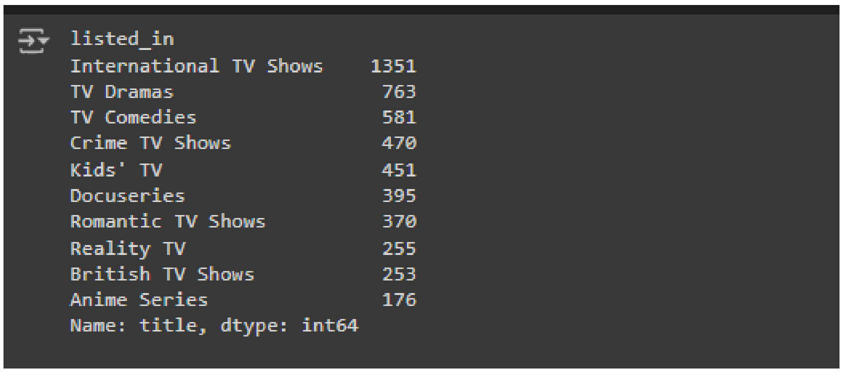
Recommendations: International movies are most produced movies, first we need to check whether these movies are giving us profits, if not we need to check genre is bringing the profits to focus on those genres.

Top 10 most produced TV show genres

Code:

```
TV_shows.groupby(by =
'listed_in')['title'].nunique().sort_values(ascending =
False).iloc[0:10]
```

Output:



```
listed_in
International TV Shows    1351
TV Dramas                  763
TV Comedies                581
Crime TV Shows             470
Kids' TV                   451
Docuseries                 395
Romantic TV Shows          370
Reality TV                 255
British TV Shows           253
Anime Series               176
Name: title, dtype: int64
```

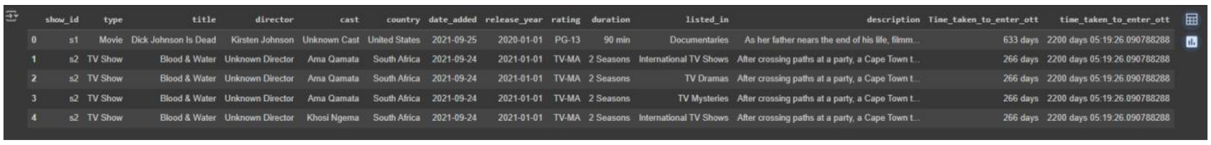
Insights: From the output we can derive that International TV shows are the most produced TV shows.

Time taken for Movies and TV shows to enter Netflix after getting released

Code:

```
df['date_added'] = pd.to_datetime(df['date_added'], errors = 'coerce')
df['release_year'] = pd.to_datetime(df['release_year'], errors =
'coerce') + pd.offsets.YearBegin(0)
df['release_year'] =
pd.to_datetime(df['release_year']).dt.strftime('%Y-%m-%d')
df['release_year'] = pd.to_datetime(df['release_year'])
df['Time_taken_to_enter_ott'] = df.apply(lambda x:
(pd.to_datetime(x['date_added']) - pd.to_datetime(x['release_year']))
if pd.notna(x['date_added']) and pd.notna(x['release_year']) else None
,axis=1)
df.head()
```

Output:



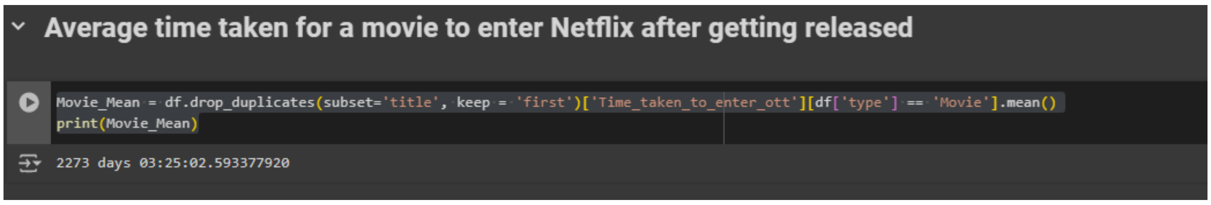
	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	Time_taken_to_enter_ott	time_taken_to_enter_ott
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	Unknown Cast	United States	2021-09-25	2020-01-01	PG-13	90 min	Documentaries	As her father nears the end of his life, filmm...	633 days	2200 days 05:19:26.090788288
1	s2	TV Show	Blood & Water	Unknown Director	Ama Qamata	South Africa	2021-09-24	2021-01-01	TV-MA	2 Seasons	International TV Shows	After crossing paths at a party, a Cape Town t...	266 days	2200 days 05:19:26.090788288
2	s2	TV Show	Blood & Water	Unknown Director	Ama Qamata	South Africa	2021-09-24	2021-01-01	TV-MA	2 Seasons	TV Dramas	After crossing paths at a party, a Cape Town t...	266 days	2200 days 05:19:26.090788288
3	s2	TV Show	Blood & Water	Unknown Director	Ama Qamata	South Africa	2021-09-24	2021-01-01	TV-MA	2 Seasons	TV Mysteries	After crossing paths at a party, a Cape Town t...	266 days	2200 days 05:19:26.090788288
4	s2	TV Show	Blood & Water	Unknown Director	Khosi Ngema	South Africa	2021-09-24	2021-01-01	TV-MA	2 Seasons	International TV Shows	After crossing paths at a party, a Cape Town t...	266 days	2200 days 05:19:26.090788288

Average time taken for movie to enter Netflix

Code:

```
Movie_Mean = df.drop_duplicates(subset='title', keep =
'first')['Time_taken_to_enter_ott'][df['type'] == 'Movie'].mean()
print(Movie_Mean)
```

Output:



Average time taken for a movie to enter Netflix after getting released

```
Movie_Mean = df.drop_duplicates(subset='title', keep = 'first')['Time_taken_to_enter_ott'][df['type'] == 'Movie'].mean()
print(Movie_Mean)
```

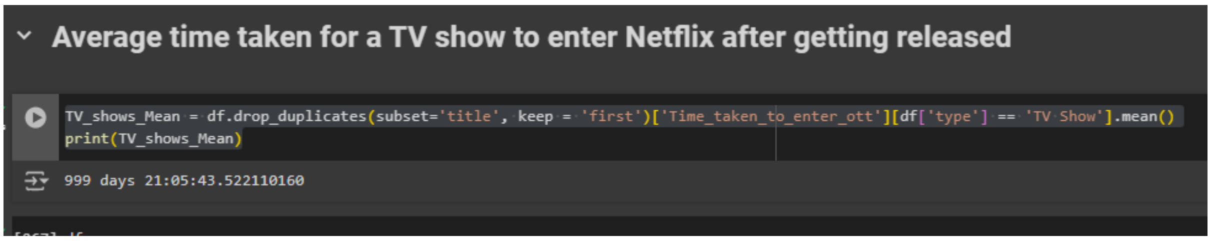
```
2273 days 03:25:02.593377920
```

Average time taken for TV show to enter Netflix

Code:

```
TV_shows_Mean = df.drop_duplicates(subset='title', keep =
'first')['Time_taken_to_enter_ott'][df['type'] == 'TV Show'].mean()
print(TV_shows_Mean)
```

Output:



Average time taken for a TV show to enter Netflix after getting released

```
TV_shows_Mean = df.drop_duplicates(subset='title', keep = 'first')['Time_taken_to_enter_ott'][df['type'] == 'TV Show'].mean()
print(TV_shows_Mean)
```

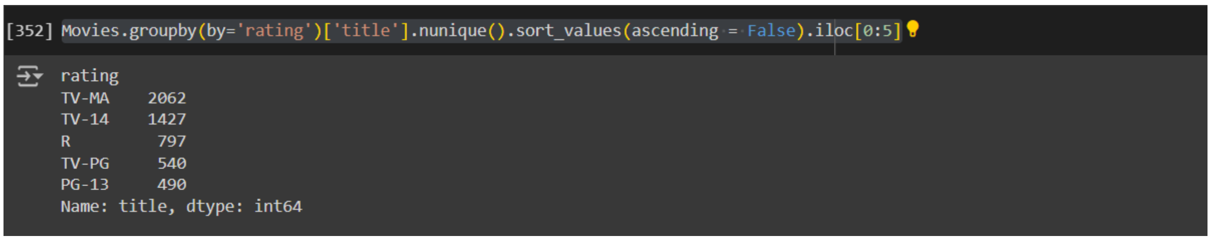
```
999 days 21:05:43.522110160
```

Top 5 most rated movies

Code:

```
Movies.groupby(by='rating')['title'].nunique().sort_values(ascending = 
False).iloc[0:5]
```

Output:



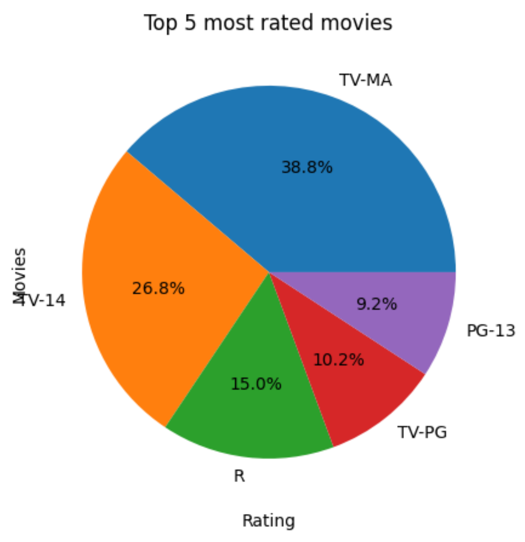
```
[352] Movies.groupby(by='rating')['title'].nunique().sort_values(ascending = False).iloc[0:5]
```

```
rating
TV-MA    2062
TV-14    1427
R         797
TV-PG     540
PG-13     490
Name: title, dtype: int64
```

Graphical Representation:

```
Movie_Ratings_plot = 
pd.DataFrame(Movies.groupby(by='rating')['title'].nunique().sort_values
(ascending = False)).reset_index().head()
Movie_Ratings_plot.rename(columns = {'title':'count'},inplace = True)
plt.figure(figsize = (10,5))
plt.pie(Movie_Ratings_plot['count'], labels = 
Movie_Ratings_plot['rating'], autopct='%1.1f%%')
plt.title('Top 5 most rated movies')
plt.xlabel('Rating')
plt.ylabel('Movies')
plt.show()
```

Graph:



Insights:

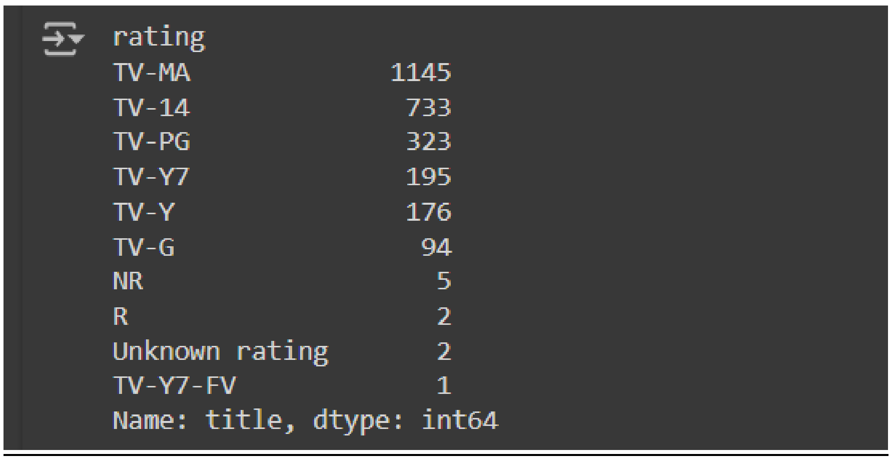
From the graph we can clearly see that most movies in the Netflix are TV-MA rated (38.8%)

<u>Top 5 most rated TV shows</u>

<u>Code:</u>

```
TV_shows.groupby(by='rating')['title'].nunique().sort_values(ascending
= False)
```

<u>Output:</u>



```
rating
TV-MA              1145
TV-14               733
TV-PG               323
TV-Y7               195
TV-Y                176
TV-G                 94
NR                    5
R                     2
Unknown rating        2
TV-Y7-FV              1
Name: title, dtype: int64
```

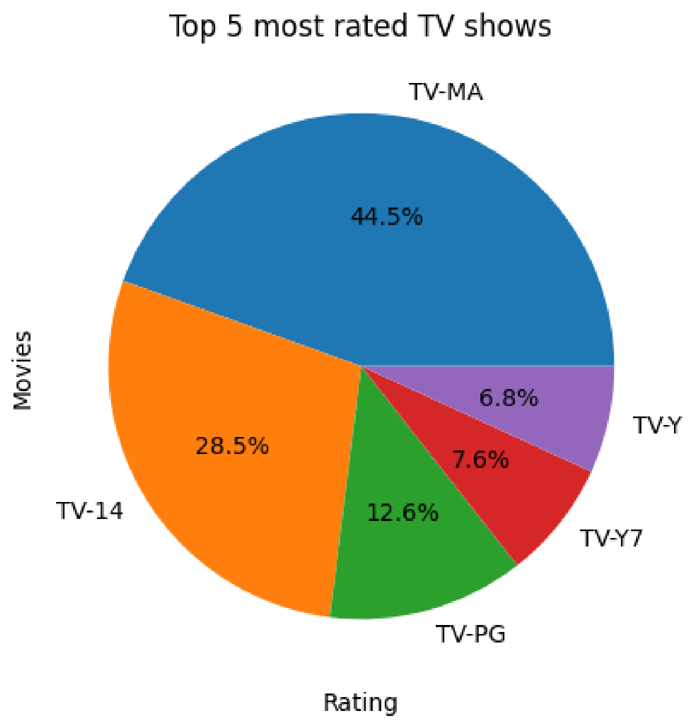
<u>Graphical Representation:</u>

<u>Code:</u>

```
TV_shows_Ratings_plot =
pd.DataFrame(TV_shows.groupby(by='rating')['title'].nunique().sort_valu
es(ascending = False)).reset_index().head()
TV_shows_Ratings_plot.rename(columns = {'title':'count'},inplace =
True)
```

```
plt.pie(TV_shows_Ratings_plot['count'], labels =
TV_shows_Ratings_plot['rating'], autopct='%1.1f%%')
plt.title('Top 5 most rated TV shows')
plt.xlabel('Rating')
plt.ylabel('Movies')
plt.show()
```

Graph:



Insights: From the graph we can clearly see that most TV shows in Netflix are TV-MA rated.

Recommendations: From the graph we can see most TV shows are TV-MA rated, if these rated movies bringing us more customers, we need to focus more marketing them by ads, if not we need to either decrease the count of bringing the most popular ones in these genres.