BUSINESS CASE:
NETFLIX DATA EXPLORATION

- Netflix, Inc. is an American technology and media services provider and production company headquartered in Los Gatos, California. Netflix was founded in 1997 by Reed Hastings and Marc Randolph in Scotts Valley, California. The company's primary business is its subscription-based streaming service, which offers online streaming of a library of films and television series, including those produced in-house.
- Netflix is one of the most popular media and video streaming platforms. They have over 8000 movies or tv shows in about 45 languages available on their platform. They have over 247.15 M Subscribers globally as of Oct,2023.
- Netflix is one of the most popular digital streaming media service providers today. Netflix provides streaming services for movies and tv shows from various countries in the world.
- The particular business case focuses on the Netflix show data and provides insightful information on 8807 shows
- Analysing the data and generating insights helps Netflix decide which type of shows/movies to produce and how to grow the business.
- Netflix data analysis will be performed using several python libraries like numpy,pandas,matplotlib,seaborn.
- Data analysis using Python libraries is widely popular among the Data Scientists and Data Analysts.
- 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.
- The following data is available in a single csv file :
  - Show ID : The ID of the 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

BUSINESS CASE:
NETFLIX DATA EXPLORATION

- Rating: TV Rating of the movie/show
- Duration: Total Duration - in minutes or number of seasons
- Listed in: Genre
- Description: The summary description

The questions under line of analysis, Google Colab Notebook commands along with a screenshot of the output are submitted below along with the valuable insights that I drew from my analysis and recommendations regarding the company's growth and profit perspective are submitted below.

BASIC ANALYSIS :

Downloading the .csv file and uploading it into Google Colab Notebook

Loading The Dataset :

Import Libraries :

Importing the libraries, we need.

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

Let's check the first 5 data :

BUSINESS CASE:
NETFLIX DATA EXPLORATION

```
df.head(5)
```

	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 I...

df

	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 I...
...	...	...	...	...	...	...	...	...	...	...	...	...
8802	s8803	Movie	Zodiac	David Fincher	Mark Ruffalo, Jake Gyllenhaal, Robert Downey J...	United States	November 20, 2019	2007	R	158 min	Cult Movies, Dramas, Thrillers	A political cartoonist, a crime reporter and a...
8803	s8804	TV Show	Zombie Dumb	NaN	NaN	NaN	July 1, 2019	2018	TV-Y7	2 Seasons	Kids' TV, Korean TV Shows, TV Comedies	While living alone in a spooky town, a young g...
8804	s8805	Movie	Zombieland	Ruben Fleischer	Jesse Eisenberg, Woody Harrelson, Emma Stone, ...	United States	November 1, 2019	2009	R	88 min	Comedies, Horror Movies	Looking to survive in a world taken over by zo...
8805	s8806	Movie	Zoom	Peter Hewitt	Tim Allen, Courteney Cox, Chevy Chase, Kate Ma...	United States	January 11, 2020	2006	PG	88 min	Children & Family Movies, Comedies	Dragged from civilian life, a former superhero...
8806	s8807	Movie	Zubaan	Mozez Singh	Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanan...	India	March 2, 2019	2015	TV-14	111 min	Dramas, International Movies, Music & Musicals	A scrappy but poor boy worms his way into a ty...

8807 rows × 12 columns

The dataset contains over 8807 titles,12 descriptions. It is a typical movie/TV shows dataframe. We can also see that there are NaN values in some columns.

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.

<u>To get All attributes df.columns :</u>

```
netflix_df.columns
```

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

<u>The Shape of the Data:</u>

```
[26] df.shape
```

df.shape - o/p is (8807,14)

```
(8807, 14)
```

BUSINESS CASE:
NETFLIX DATA EXPLORATION

Columns of the dataset: (df.columns)

```
[38] df.columns

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

The Dimension of the dataframe and Data types of all the attributes :

df.ndim and df.info()

```
[29] df.ndim

2
```

```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
Data columns (total 14 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
 12  str_split     7976 non-null   object
 13  cast_split    7982 non-null   object
dtypes: int64(1), object(13)
memory usage: 963.4+ KB
```

Based on the output obtained, it can be seen that the majority of columns have a total of **8807** non-null values, but there are **8** columns that have a varying number of non-null values. In addition, it can be seen that **13** columns have an **object** data type and **1** column has a **int64** data type. Out of which 2 columns were added by me for unnesting of cast data and country data i.e., str_split and cast_split(12 & 13 columns).

Statistical Summary Before Data Cleaning:

df.describe()

BUSINESS CASE:
NETFLIX DATA EXPLORATION

```
[31] df.describe()
```

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

Missing Value Detection Data Profiling & Cleaning :

Data Cleaning means the process of identifying incorrect, incomplete, inaccurate, irrelevant, or missing pieces of data and then modifying, replacing, or deleting them as needed. Data Cleansing is considered as the basic element of Data Science.

df.isnull().any()

```
df.isnull().any()
```

```
show_id         False
type            False
title           False
director         True
cast             True
country          True
date_added       True
release_year    False
rating           True
duration         True
listed_in       False
description     False
str_split        True
cast_split       True
dtype: bool
```

#Number of null values per column

BUSINESS CASE: NETFLIX DATA EXPLORATION

df.isna().sum() / df.isnull ().sum()

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

```
show_id            0
type               0
title              0
director        2634
cast             825
country          831
date_added        10
release_year       0
rating             4
duration           3
listed_in          0
description        0
str_split        831
cast_split       825
dtype: int64
```

From the output above, we know that there are **8** columns with null values. '**director**' column has **2,634** null values, '**cast**' has **825** null values, '**country**' has **831** null values, '**date_added'** has **10** null values, '**rating**' has **4** null values, '**duration**' has **3** null values, '**str_split**'(column for unnesting countries) has **831** null values, and '**cast_split**'(column added for unnesting cast) has **825** null values.

The null values in total are 5963.

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

```
5963
```

<u>Checking for duplicate Values:</u> There are no duplicate values in the data.

df.duplicated()

<u>Data Cleansing:</u> Imputation is a treatment method for filling of missing values by using certain techniques. Can use mean, mode, or use predictive modelling. In this case study, we will discuss the use of the fillna function from Pandas for this imputation. Drop rows containing missing values using the dropna function from Pandas

BUSINESS CASE:
NETFLIX DATA EXPLORATION

#Imputing missing values

Panda's fillna()/replace functions can be used to fill in missing values in a dataset. I replaced the null values in the 'country' and "str_split" column with cast Unknown Country, I also replaced the null values in the 'cast' column and "cast_split" column with Unknown Cast, and replaced the null values in the 'director' column with Unknown Director(as per Solution Approach Sheet).

```
df["director"].replace(np.nan,"Unknown Director",inplace=True)
df["cast"].fillna("Unknown Cast",inplace=True)
df["country"].replace(np.nan,"Unknown Country",inplace=True)
df["cast_split"].fillna("Unknown Cast",inplace=True)
df["str_split"].replace(np.nan,"Unknown Country",inplace=True)
df.isnull().sum()
```

```
show_id          0
type             0
title            0
director         0
cast             0
country          0
date_added      10
release_year     0
rating           4
duration         3
listed_in        0
description      0
str_split        0
cast_split       0
dtype: int64
```

** The rating column has 3 time(in minutes) datatype as shown below.

```
df["rating"].value_counts()
```

```
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: rating, dtype: int64
```

BUSINESS CASE:
NETFLIX DATA EXPLORATION

Hence I have dropped those three rows as the number is quite low.

```
df.drop(df[df["rating"]=="66 min"].index,inplace=True)
df.rating.value_counts()
```

```
<ipython-input-103-c66b98757d85>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/par
  df.drop(df[df["rating"]=="66 min"].index,inplace=True)
TV-MA        3205
TV-14        2157
TV-PG         861
R             799
PG-13         490
TV-Y7         333
TV-Y          306
PG            287
TV-G          220
NR             79
G              41
TV-Y7-FV        6
NC-17           3
UR              3
Name: rating, dtype: int64
```

#Dropping missing values:

Columns ‘date_added’, ‘rating’, ‘duration’ have very less missing values. Therefore, I will remove any rows that have missing values in those columns.

```
df=df.dropna()
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          0
listed_in         0
description       0
str_split         0
cast_split        0
dtype: int64
```

Now let’s check for the total null values:

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

```
0
```

The total null values are 0 after data cleaning

BUSINESS CASE:
NETFLIX DATA EXPLORATION

#Dataset information: The following is the dataset information after cleaning the data.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 8790 entries, 0 to 8806
Data columns (total 14 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   show_id       8790 non-null   object
 1   type          8790 non-null   object
 2   title         8790 non-null   object
 3   director      8790 non-null   object
 4   cast          8790 non-null   object
 5   country       8790 non-null   object
 6   date_added    8790 non-null   object
 7   release_year  8790 non-null   int64
 8   rating        8790 non-null   object
 9   duration      8790 non-null   object
 10  listed_in     8790 non-null   object
 11  description   8790 non-null   object
 12  str_split     8790 non-null   object
 13  cast_split    8790 non-null   object
dtypes: int64(1), object(13)
memory usage: 1.0+ MB
```

Statistical Summary After Data Cleaning:

```
df.describe()
```

	release_year
count	8790.000000
mean	2014.183163
std	8.825466
min	1925.000000
25%	2013.000000
50%	2017.000000
75%	2019.000000
max	2021.000000

What does 'good' look like?

1. Find the counts of each categorical variable both using graphical and non-graphical analysis.

a. For Non-graphical Analysis: I found out the value_counts() for all the categorical data which shown as below :

BUSINESS CASE:
NETFLIX DATA EXPLORATION

```
1) df["title"].value_counts()
```

```
Dick Johnson Is Dead                1
Cooked                              1
My Beautiful Broken Brain           1
Pee-wee's Big Holiday               1
Netflix Presents: The Characters    1
                                   ..
Sleepless Society: Insomnia         1
Palazuelos mi rey                   1
Narcos: Mexico                      1
Love Is Blind                       1
Zubaan                              1
Name: title, Length: 8790, dtype: int64
```

```
2) df["director"].value_counts()
```

```
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: director, Length: 4528, dtype: int64
```

```
3) df["rating"].value_counts()
```

```
TV-MA        3205
TV-14        2157
TV-PG         861
R             799
PG-13         490
TV-Y7         333
TV-Y          306
PG            287
TV-G          220
NR             79
G              41
TV-Y7-FV        6
NC-17           3
UR              3
Name: rating, dtype: int64
```

```
4) df.release_year.value_counts().sort_index(ascend
   ing=False)
```

BUSINESS CASE:
NETFLIX DATA EXPLORATION

```
2021     592
2020     953
2019    1030
2018    1146
2017    1030
        ...
1945       4
1944       3
1943       3
1942       2
1925       1
Name: release_year, Length: 74, dtype: int64
```

```
5) df["type"].value_counts()
```

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

```
6) df.date_added.value_counts()
```

```
January 1, 2020      109
November 1, 2019      89
March 1, 2018         75
December 31, 2019     74
October 1, 2018       71
                     ...
December 4, 2016       1
November 21, 2016      1
November 19, 2016      1
November 17, 2016      1
January 11, 2020       1
Name: date_added, Length: 1767, dtype: int64
```

```
7) df.duration.value_counts()
```

BUSINESS CASE:
NETFLIX DATA EXPLORATION

```
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: duration, Length: 220, dtype: int64
```

```
8) df.listed_in.value_counts()
```

```
Dramas, International Movies                          362
Documentaries                                         359
Stand-Up Comedy                                       334
Comedies, Dramas, International Movies                274
Dramas, Independent Movies, International Movies      252
                                                     ...
Kids' TV, TV Action & Adventure, TV Dramas              1
TV Comedies, TV Dramas, TV Horror                       1
Children & Family Movies, Comedies, LGBTQ Movies        1
Kids' TV, Spanish-Language TV Shows, Teen TV Shows      1
Cult Movies, Dramas, Thrillers                          1
Name: listed_in, Length: 514, dtype: int64
```

```
9) df.show_id.value_counts()
```

```
s1        1
s5875     1
s5869     1
s5870     1
s5871     1
         ..
s2931     1
s2930     1
s2929     1
s2928     1
s8807     1
Name: show_id, Length: 8807, dtype: int64
```

```
10)    df["str_split"]=df.country.str.split(", ")
df.explode("str_split").str_split.value_counts()
```

BUSINESS CASE:
NETFLIX DATA EXPLORATION

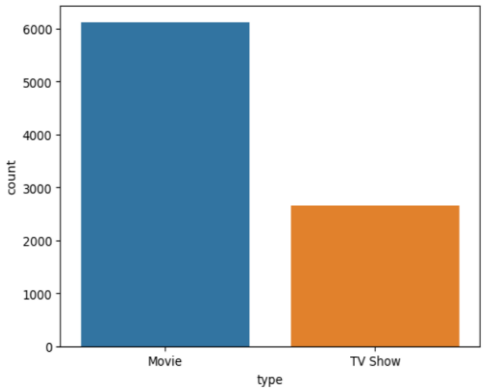
```
United States     3689
India             1046
United Kingdom     804
Canada             445
France             393
                  ...
Bermuda              1
Ecuador              1
Armenia              1
Mongolia             1
Montenegro           1
Name: str_split, Length: 127, dtype: int64
```

```
11)    df["cast_split"]=df.cast.str.split(", ")
df.explode("cast_split").cast_split.value_counts()
```

```
Anupam Kher                43
Shah Rukh Khan             35
Julie Tejwani              33
Naseeruddin Shah           32
Takahiro Sakurai           32
                           ..
Maryam Zaree                1
Melanie Straub              1
Gabriela Maria Schmeide     1
Helena Zengel               1
Chittaranjan Tripathy       1
Name: cast_split, Length: 36439, dtype: int64
```

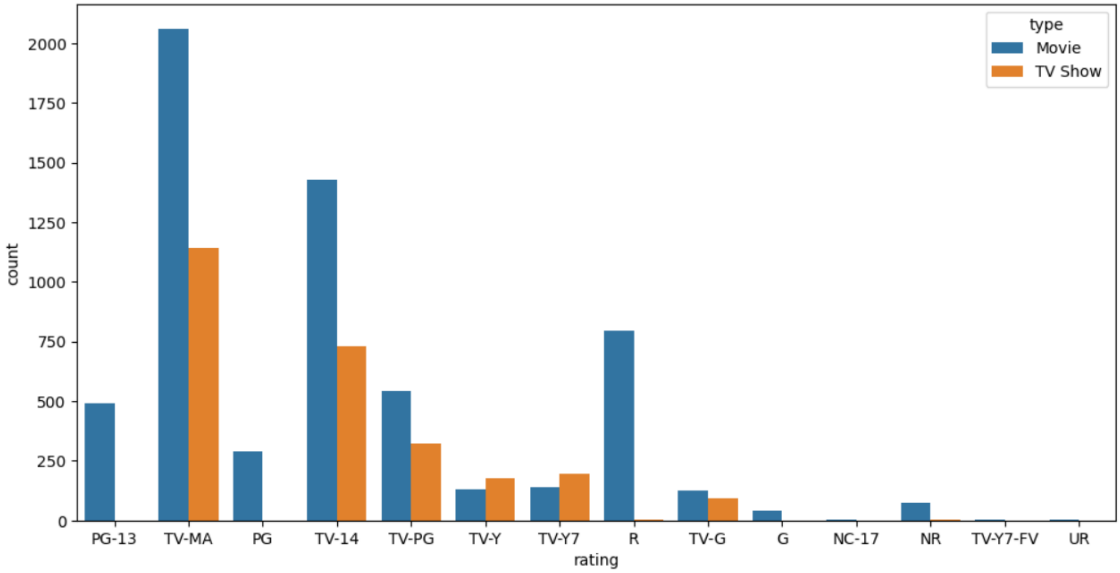
b. For graphical analysis:

```
sns.countplot(x="type",data=df)
plt.show()
```

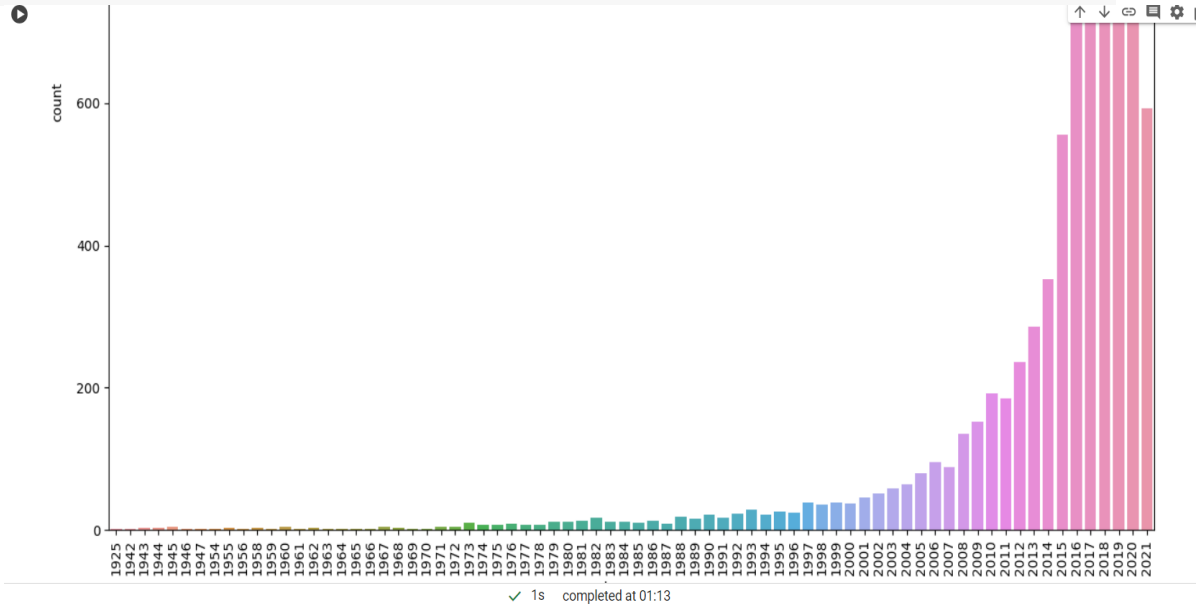


BUSINESS CASE:
NETFLIX DATA EXPLORATION

```
plt.figure(figsize=(12,6))
sns.countplot(x="rating",data=df,hue="type")
plt.show()
```




```
plt.figure(figsize=(15,11))
sns.countplot(x="release_year",data=df)
plt.xticks(rotation=90)
plt.show()
```



2. Comparison of tv shows vs. movies.

BUSINESS CASE:
NETFLIX DATA EXPLORATION

a) Find the number of movies produced in each country and pick the top 10 countries.

```
df_movies = df[df["type"] == "Movie"]
df_movies = df_movies.assign(country =
df_movies["country"].str.split(", ")).explode("country")
movie_count
=df_movies["country"].value_counts().sort_values(ascending=False)
top_10 = movie_count.head(10).to_frame().reset_index()
top_10.columns = ["country", "movie_count"]
print(top_10)
```

	country	movie_count
0	United States	2492
1	India	942
2	United Kingdom	475
3	Canada	295
4	France	285
5	Germany	168
6	Spain	156
7	Japan	114
8	China	108
9	Hong Kong	100

b. Find the number of Tv-Shows produced in each country and pick the top 10 countries.

```
df_tv_shows = df[df["type"] == "TV Show"]
df_tv_shows = df_tv_shows.assign(country =
df_tv_shows["country"].str.split(", ")).explode("country")
tv_shows_count
=df_tv_shows["country"].value_counts().sort_values(ascending=False)
top_10 = tv_shows_count.head(10).to_frame().reset_index()
top_10.columns = ["country", "tv_show_count"]

print(top_10)
```

	country	tv_show_count
0	United States	782
1	United Kingdom	232
2	Japan	193
3	South Korea	168
4	Canada	119
5	France	76
6	Taiwan	70
7	India	65
8	Australia	58
9	Mexico	55

BUSINESS CASE: NETFLIX DATA EXPLORATION

3. What is the best time to launch a TV show?

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

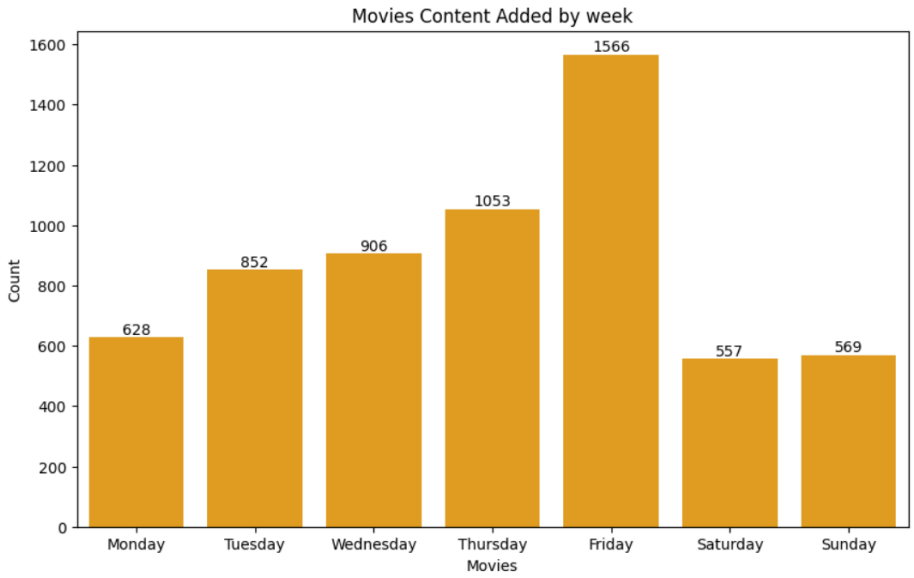
For Movies: Best Day of the Week to release a movie is Friday

```
df["week_added"]=pd.to_datetime(df["date_added"]).dt.day_name(
)
week_order=
['Monday','Tuesday','Wednesday','Thursday','Friday','Saturday'
,'Sunday']
weekly_show_count=df["week_added"].value_counts().loc[week_ord
er]
df_movies = df[df['type'] == 'Movie']
movies_count =
df_movies['week_added'].value_counts().loc[week_order]
movies_count
```

```
Monday         628
Tuesday        852
Wednesday      906
Thursday      1053
Friday        1566
Saturday       557
Sunday         569
Name: week_added, dtype: int64
```

```
plt.figure(figsize=(10,6))
bar_plot = sns.barplot(x=movies_count.index, y=movies_count.values,
color="orange")
plt.xlabel('Movies')
plt.ylabel('Count')
plt.title('Movies Content Added by week')
for index, value in enumerate(movies_count.values):
    bar_plot.text(index, value, str(value), ha='center', va='bottom')
plt.show()
```

BUSINESS CASE:
NETFLIX DATA EXPLORATION



For TV SHOWS : Best Day to add is Friday

```
df["week_added"]=pd.to_datetime(df["date_added"]).dt.day_name(
)
week_order=
['Monday','Tuesday','Wednesday','Thursday','Friday','Saturday'
,'Sunday']
weekly_show_count=df["week_added"].value_counts().loc[week_ord
er]
df_tv_shows = df[df['type'] == 'TV Show']
tv_shows_count =
df_tv_shows['week_added'].value_counts().loc[week_order]
tv_shows_count
```

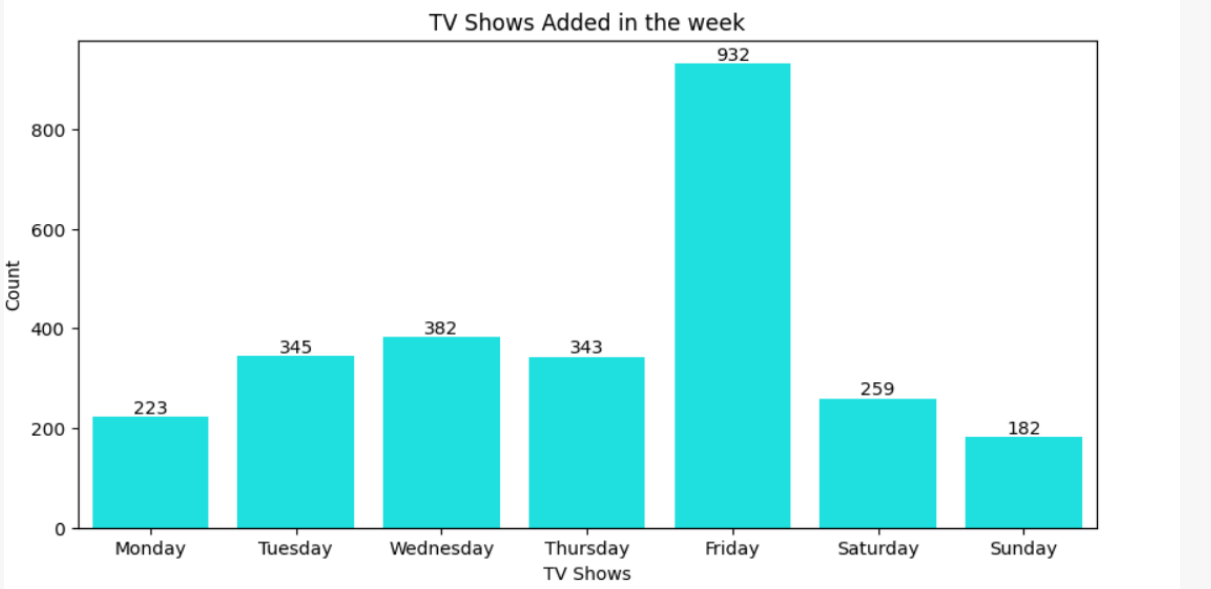
```
[9]  tv_shows_count

     Monday        223
     Tuesday       345
     Wednesday     382
     Thursday      343
     Friday        932
     Saturday      259
     Sunday        182
     Name: week_added, dtype: int64
```

BUSINESS CASE:
NETFLIX DATA EXPLORATION

```
plt.figure(figsize=(10,5))
```

```
bar_plot = sns.barplot(x=tv_shows_count.index, y=tv_shows_count.values,
color="cyan")
plt.xlabel('TV Shows')
plt.ylabel('Count')
plt.title('TV Shows Added in the week')
for index, value in enumerate(tv_shows_count.values):
    bar_plot.text(index, value, str(value), ha='center', va='bottom')
plt.show()
```



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

For TV SHOWS : BEST MONTH is December

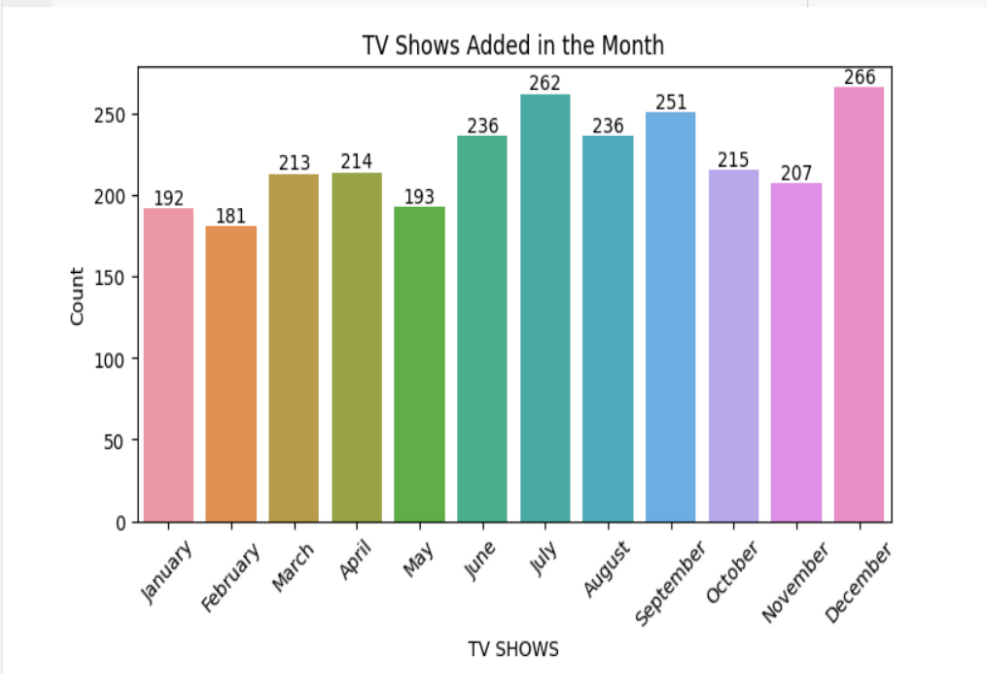
```
tv_show=df[df["type"]=="TV Show"]
df["month_added"]=pd.to_datetime(df["date_added"]).dt.month_name()
month_order= ['January', 'February', 'March', 'April', 'May', 'June',
'July','August', 'September', 'October', 'November', 'December']
monthly_tv_show_count =
df_tv_shows['month_added'].value_counts().loc[month_order]
monthly_tv_show_count
plt.figure(figsize=(8,4))
bar_plot = sns.barplot(x=monthly_tv_show_count.index,
y=monthly_tv_show_count.values)
plt.xlabel('TV SHOWS')
plt.ylabel('Count')
```

BUSINESS CASE:
NETFLIX DATA EXPLORATION

```
plt.title('TV Shows Added in the Month')
plt.xticks(rotation=45)
for index, value in enumerate(monthly_tv_show_count.values):
    bar_plot.text(index, value, str(value), ha='center', va='bottom')
plt.show()
```

```
[36] tv_shows_count
```

```
January      192
February     181
March        213
April        214
May          193
June         236
July         262
August       236
September    251
October      215
November     207
December     266
Name: month_added, dtype: int64
```



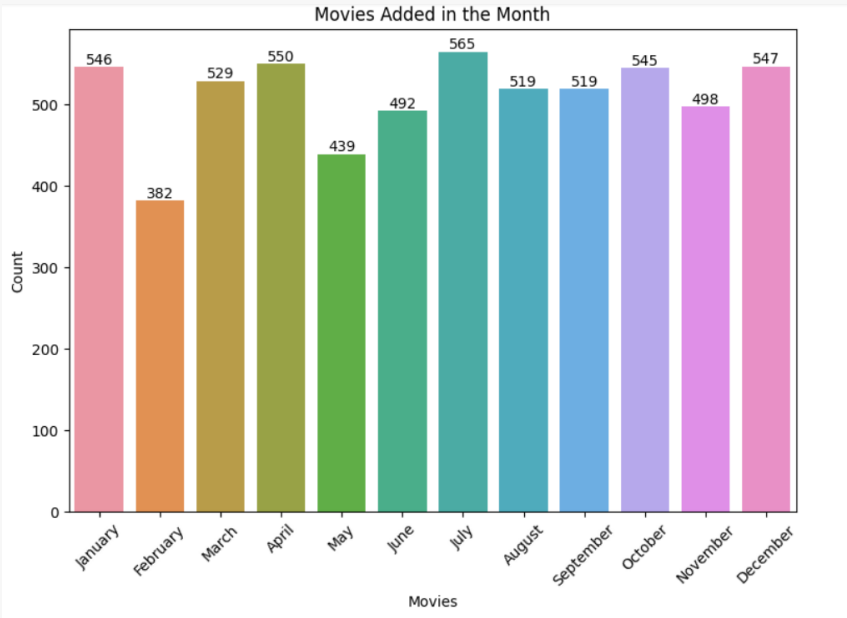
FOR MOVIES: Best Month is July

```
df["month_added"]=pd.to_datetime(df["date_added"]).dt.month_name()
month_order= ['January', 'February', 'March', 'April', 'May', 'June',
'July','August', 'September', 'October', 'November', 'December']
```

BUSINESS CASE:
NETFLIX DATA EXPLORATION

```
movies_count = df_movies['month_added'].value_counts().loc[month_order]
movies_count
plt.figure(figsize=(9,6))
bar_plot = sns.barplot(x=movies_count.index, y=movies_count.values)
plt.xlabel('Movies')
plt.ylabel('Count')
plt.title('Movies Added in the Month')
plt.xticks(rotation=45)
for index, value in enumerate(movies_count.values):
    bar_plot.text(index, value, str(value), ha='center', va='bottom')
plt.show()
```

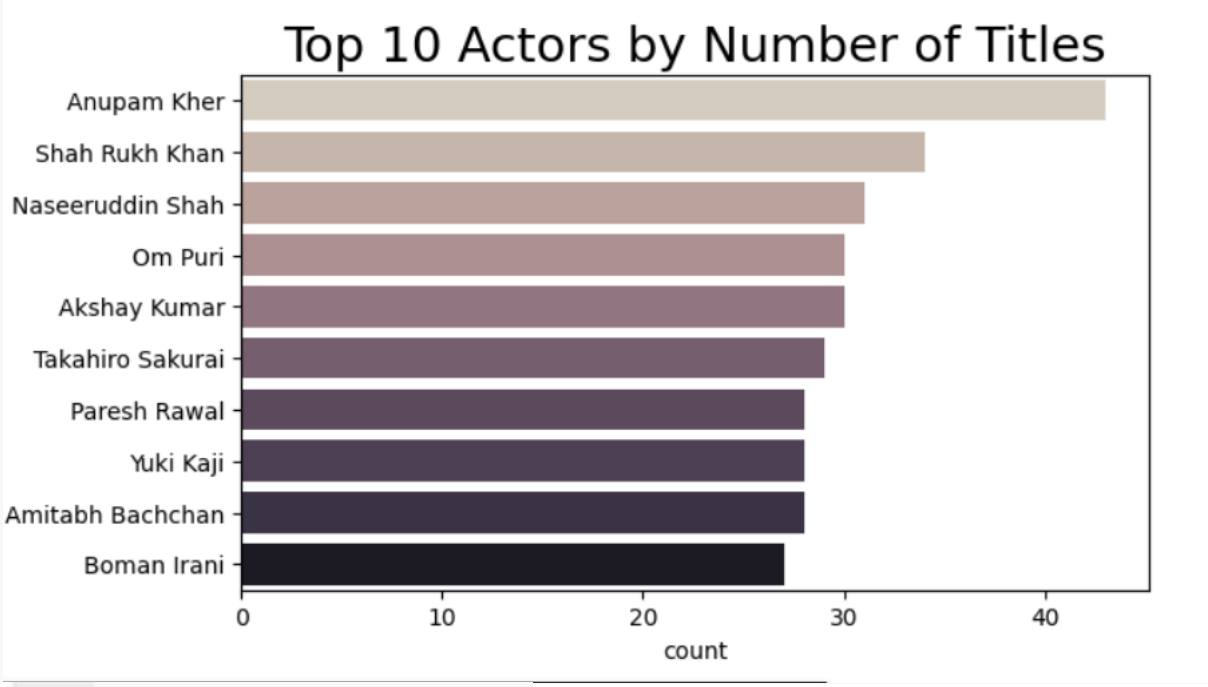
```
January      546
February     382
March        529
April        550
May          439
June         492
July         565
August       519
September    519
October      545
November     498
December     547
Name: month_added, dtype: int64
```



BUSINESS CASE:
NETFLIX DATA EXPLORATION

4. Analysis of actors/directors of different types of shows/movies.
a. Identify the top 10 directors who have appeared in most movies or TV shows.

```
plt.figure(figsize=(10,8))
df_cast = df[df.cast != 'Unknown
Cast'].set_index('title').cast.str.split(', ',
expand=True).stack().reset_index(level=1, drop=True)
sns.countplot(y = df_cast, order=df_cast.value_counts().index[:10],
palette='magma_r', saturation=.2)
plt.title('Top 10 Actors by Number of Titles', fontsize=21);
plt.show()
```

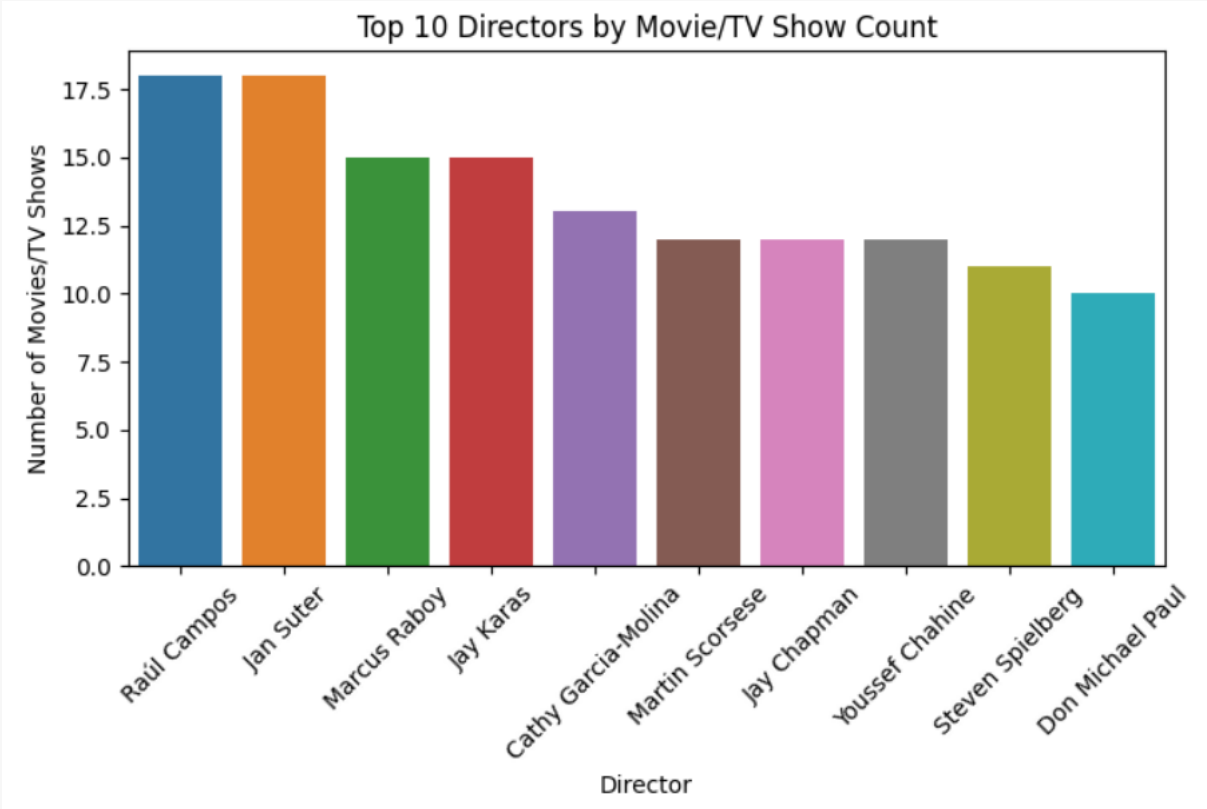



```
Anupam Kher         43
Shah Rukh Khan      34
Naseeruddin Shah    31
Om Puri             30
Akshay Kumar        30
Takahiro Sakurai    29
Paresh Rawal        28
Yuki Kaji           28
Amitabh Bachchan    28
Boman Irani         27
```

BUSINESS CASE:
NETFLIX DATA EXPLORATION

b. Identify the top 10 directors who have appeared in most movies or TV shows.

```
df_director = df['director'].str.split(',', expand=True).stack()
df_director = df_director.reset_index(level=1,
drop=True).to_frame('director')
df_director['show_id'] = df['show_id']
director_counts = df_director['director'].value_counts()[1:]
top_10_directors = director_counts.head(10)
plt.figure(figsize=(9, 5))
bar_plot = sns.barplot(x=top_10_directors.index,
y=top_10_directors.values)
plt.xlabel('Director')
plt.ylabel('Number of Movies/TV Shows')
plt.title('Top 10 Directors by Movie/TV Show Count')
plt.xticks(rotation=45)
plt.show()
```



BUSINESS CASE:
NETFLIX DATA EXPLORATION

```
top_10_directors
```

```
Raúl Campos            18
 Jan Suter             18
Marcus Raboy           15
Jay Karas              15
Cathy Garcia-Molina    13
Martin Scorsese        12
Jay Chapman            12
Youssef Chahine        12
Steven Spielberg       11
Don Michael Paul       10
Name: director, dtype: int64
```

5. Which genre movies are more popular or produced more

```
from wordcloud import WordCloud
text= " ".join(str(each) for each in df.listed_in)
wordcloud=WordCloud(max_words=200,background_color="black").generate(te
xt)
colormaps = ['plasma', 'Purples','inferno','PuRd', 'Blues',
'BuGn','cividis', 'PuRd','YlGn','Greens',
'YlOrBr', 'BuGn','YlOrRd',  'Oranges', 'RdPu',
  'YlGnBu', 'PuBuGn', 'OrRd','magma','Reds','BuGn' ]
plt.figure(figsize=(8,4))
plt.figure(figsize=(10,8))
plt.imshow(wordcloud,interpolation="Bilinear")
plt.title("More Popular Movies or More Produced Genres",fontsize=25)
plt.axis("off")
plt.show()
```

BUSINESS CASE:
NETFLIX DATA EXPLORATION

From the graph, we know that International Movies take the first place, followed by dramas and comedies.

6) Find After how many days the movie will be added to Netflix after the release of the movie (you can consider the recent past data)

```
df["year_added"]=pd.to_datetime(df.date_added).dt.year
df["year_added"].dropna().astype(int)

df_movies = df[df['type'] == 'Movie']
df_movies ['year_added'] = pd.to_datetime(df_movies.date_added).dt.year
df_tv_shows = df[df['type'] == 'TV Show']
df_tv_shows.loc[:,"year_added"]=pd.to_datetime(df_tv_shows.date_added).
dt.year
year_df= df.loc[:,
'year_added'].value_counts().to_frame().reset_index().rename(columns={"
index": "year",
"year_added":"count"})
year_df = year_df[year_df.year != 2020]
print(year_df)
```

BUSINESS CASE:
NETFLIX DATA EXPLORATION

	year	count
0	2019	1722
2	2018	1404
3	2021	1056
4	2017	989
5	2016	358
6	2015	70
7	2014	20
8	2011	13
9	2013	11
10	2012	3
11	2009	2
12	2008	1
13	2010	1

```
movies_year_df
=df_movies.year_added.value_counts().to_frame().reset_index().rename(co
lumns={"index":
"year", "year_added":"count"})
movies_year_df = movies_year_df[movies_year_df != 2020]
movies_year_df
```

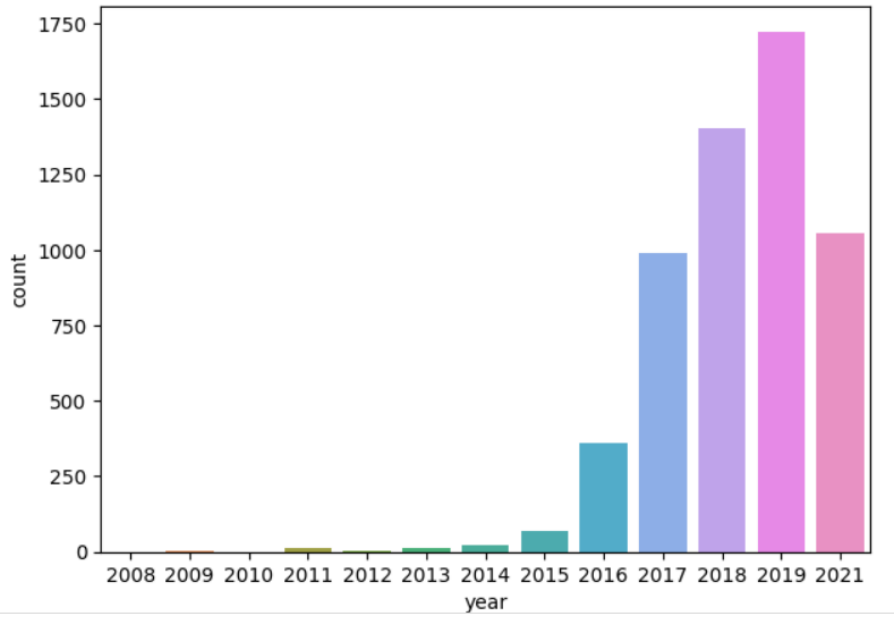
	year	count
0	2019.0	1261
1	NaN	1164
2	2018.0	1114
3	2021.0	735
4	2017.0	710
5	2016.0	205
6	2015.0	47
7	2014.0	15
8	2011.0	13
9	2013.0	6
10	2012.0	3
11	2009.0	2
12	2008.0	1
13	2010.0	1

```
shows_year_df =
df_tv_shows.year_added.value_counts().to_frame().reset_index().rename(c
olumns={"index":
"year", "year_added":"count"})
shows_year_df = shows_year_df[shows_year_df != 2020]
shows_year_df
```

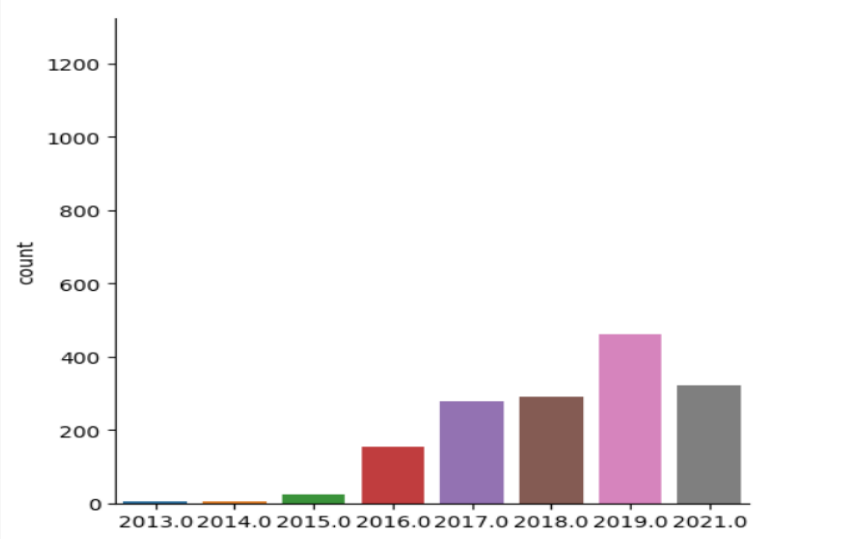
BUSINESS CASE:
NETFLIX DATA EXPLORATION

	year	count
0	NaN	476
1	2019.0	461
2	2021.0	321
3	2018.0	290
4	2017.0	279
5	2016.0	153
6	2015.0	23
7	2014.0	5
8	2013.0	5

```
fig, ax = plt.subplots(figsize=(7, 5))
sns.barplot(data=year_df, x='year', y='count')
sns.displot(data=movies_year_df, x='year', y='count')
sns.barplot (data=shows_year_df, x='year', y='count')
ax.set_xticks(np.arange(2008, 2020, 1),rotation=90)
plt.title("Total content added across all years (up to 2019)")
plt.legend(['Total','Movie','TV Show'])
plt.ylabel("Releases")
plt.xlabel("Year")
plt.show()
```



BUSINESS CASE:
NETFLIX DATA EXPLORATION




```
df["year_added"]=pd.to_datetime(df.date_added).dt.year
df["year_added"].dropna().astype(int)
```

```
1       2021
4       2021
7       2021
8       2021
9       2021
        ...
8801    2016
8802    2019
8804    2019
8805    2020
8806    2019
Name: year_added,
```

```
diff_years = df["year_added"]-df["release_year"]
mode_diff=df_movies["diff_years"].mode()
diff_years
```

```
1        0
4        0
7       28
8        0
9        0
        ..
8801     1
8802    12
8804    10
8805    14
8806     4
Length: 7290, dtype: int64
```

BUSINESS CASE:
NETFLIX DATA EXPLORATION

6) Find After how many days the movie will be added to Netflix after the release of the movie (you can consider the recent past data)

```
df.loc[df["type"] == "Movie", "date_added"] =
pd.to_datetime(df["date_added"])
# Calculate the difference between the release_year and
year_added columns
df.loc[df["type"] == "Movie", "diff_years"] =
df["release_year"] - df["year_added"]
# Find the mode of the difference
mode_diff = df_movies["diff_years"].mode()
diff_years
df_movies_added =
df.groupby('diff_years').size().reset_index(name='movies_added
')
df_movies_added[:10]
```

	diff_years	movies_added
0	-1 days +23:59:59.999999925	1
1	-1 days +23:59:59.999999927	1
2	-1 days +23:59:59.999999928	1
3	-1 days +23:59:59.999999929	1
4	-1 days +23:59:59.999999930	1
5	-1 days +23:59:59.999999934	2
6	-1 days +23:59:59.999999935	2
7	-1 days +23:59:59.999999936	2
8	-1 days +23:59:59.999999937	1
9	-1 days +23:59:59.999999938	2

BUSINESS CASE:
NETFLIX DATA EXPLORATION

Conclusions :

- Country that produces the largest number of content titles on Netflix is the **United States** with **2,000++** content titles production.
- The genre with the largest number of content titles is **International Movies** with **1,700++** content.
- The number of content titles on Netflix continued to **increase** from **2012** to **2019**.
- The actor with the largest number of content titles on Netflix is **Anupam Kher followed by Shah Rukh Khan**.
- The total number of movies are **6131** and the total number of TV shows are**2676** of the total content.
- The Director with the largest number of content titles on Netflix is **Jan Suter and Raul Campos** who has directed **17++** number of content titles on Netflix.
- The best day to release new TV Shows or Movies is **FRIDAY.**
- The best month to release new TV Shows is **December** and for new movies is **July.**
- The most popular Genres produced are International Movies and TV Shows.
- Most of the TV shows with shorter duration are most watched ones.

BUSINESS CASE:
NETFLIX DATA EXPLORATION

As the streaming industry evolves, understanding these patterns and trends becomes increasingly essential for navigating the dynamic landscape of Netflix and its vast library.

Our data analysis journey showcased the power of data in unravelling the mysteries of Netflix's content landscape, providing valuable insights for viewers and content creators.

RECOMMENDATIONS :

- Netflix has to focus on TV Shows also because there are people who will like to see tv shows rather than movies
- By approaching the top directors we can plan some more movies/tv shows in order to increase the popularity
- •Not only reaching top director we can also meet the director with less no of movies with quality content and having high rating as there may be some financial issues or any other issue which can be sorted out in order to get good content on Netflix
- We have seen most no of international movies genre so it is needed to give priority to other geners like horror,comedy..etc and also to concentrate mostly on Internation Movies Genre to get more subscribers
- In TV Shows we may focus on thriller genre which will be helpful for having more no of seasons

BUSINESS CASE:
NETFLIX DATA EXPLORATION

- Most of the movies released in OTT are in a year 2019 so we need to go on increasing this value in order to attract people by showing that getting subscription is useful as Netflix is releasing more movies per year

- Mainly the release of films or tv shows in Netflix should focus on the festival holidays, year end and week ends which is to be mainly focussed

- Some movies can be released directly into Netflix platform to gain more popularity which has some positive talk which may help in improving subscriptions

- Should focus on a actor who has immense following and make use of it by doing a TV Shows or web series.

- Advertisement between the content should be very less and movies released should be increased to drag customers for subscription.

Link to python Colab Notebook :
https://colab.research.google.com/drive/10h6xm1Nl_1iZiIPxQIN3eN4badARnwyj?usp=drive_link

---

BUSINESS CASE:
NETFLIX DATA EXPLORATION