Netflix - Data Exploration & Visualisation - Case Study 2

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 ijn deciding which type of shows/movies to produce and how they can grow the business in different countries

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

```
!wget "https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/000/940/original/netflix.csv"
```

```
--2023-03-08 05:30:26--  https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/000/940/original/netflix.csv
Resolving d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)... 18.65.40.33, 18.65.40.189, 18.65.40.103, ...
Connecting to d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)|18.65.40.33|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3399671 (3.2M) [text/plain]
Saving to: ‘netflix.csv’

netflix.csv         100%[===================>]   3.24M  --.-KB/s    in 0.09s

2023-03-08 05:30:27 (34.8 MB/s) - ‘netflix.csv’ saved [3399671/3399671]
```

```
df = pd.read_csv("netflix.csv")  # listed_in == genre  , description can be neglected as we cant analyse
df
```

	show_id	type	title	director	cast	country	date_added	release_year	rat
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	2021	TV-
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	September 24, 2021	2021	TV-
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	NaN	September 24, 2021	2021	TV-
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K	India	September 24, 2021	2021	TV-

```
df[df["director"].apply(lambda x : "," in str(x))]    ## director column checking if data seperated by ","
```

	show_id	type	title	director	cast	country	date_added	release_year
6	s7	Movie	My Little Pony: A New Generation	Robert Cullen, José Luis Ucha	Vanessa Hudgens, Kimiko Glenn, James Marsden, ...	NaN	September 24, 2021	202
16	s17	Movie	Europe's Most Dangerous Man: Otto Skorzeny in ...	Pedro de Echave García, Pablo Azorín Williams	NaN	NaN	September 22, 2021	202(
23	s24	Movie	Go! Go! Cory Carson: Chrissy Takes the Wheel	Alex Woo, Stanley Moore	Maisie Benson, Paul Killam, Kerry Gudjohnsen, ...	NaN	September 21, 2021	202
30	s31	Movie	Ankahi Kahaniya	Ashwiny Iyer Tiwari, Abhishek Chaubey, Saket C...	Abhishek Banerjee, Rinku Rajguru, Delzad Hiwal...	NaN	September 17, 2021	202

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   show_id       8807 non-null   object
 1   type          8807 non-null   object
 2   title         8807 non-null   object
 3   director      6173 non-null   object
 4   cast          7982 non-null   object
 5   country       7976 non-null   object
 6   date_added    8797 non-null   object
 7   release_year  8807 non-null   int64 
 8   rating        8803 non-null   object
 9   duration      8804 non-null   object
 10  listed_in     8807 non-null   object
 11  description   8807 non-null   object
dtypes: int64(1), object(11)
memory usage: 825.8+ KB
```

```
df.shape
```

```
(8807, 12)
```

```
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

```
df.describe(include='object').T
```

	count	unique	top	freq
show_id	8807	8807	s1	1
type	8807	2	Movie	6131
title	8807	8807	Dick Johnson Is Dead	1
director	6173	4528	Rajiv Chilaka	19
cast	7982	7692	David Attenborough	19
country	7976	748	United States	2818

```
df.nunique() ## Unique values count for each columns of over all dataset
```

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

```
df["type"].value_counts()  ## type column has 2 unique values and it count
```

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

```
df["release_year"].value_counts() ## Release year unique value count and no.of movies& TV shows released  in that year
```

```
2018    1147
2017    1032
2019    1030
2020     953
2016     902
        ...
1959       1
1925       1
1961       1
1947       1
1966       1
Name: release_year, Length: 74, dtype: int64
```

NULL Values

Q5. Missing Value & Outlier check

```
# Percentage of null values in respective columns
df.isnull().sum()/len(df)*100

# Nearly 25 to 30% is the maximum missing values are allowed, so we can use mode computation and proceed further
# And the missing values columns are director,cast,country,date_added,rating,duration
```

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

Filling null values

```
df["director"].value_counts() ## Rajiv Chilaka has the highest freq in director column
```

```
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
```

```
df["cast"].value_counts() # As max freq of cast is single value we can impute this value to NaN values
```

```
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, Suzuki 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, Bayray McNwizu,
Biodun Stephen                                                 1
Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanana, Manish Chaudhary, Meghna Malik, Malkeet Rauni, Anita Shabdish, Chittaranjan
Tripathy                                                       1
Name: cast, Length: 7692, dtype: int64
```

```
df["country"].value_counts()  # Max freq of country column is US so we can proceed to fill NaN values with it.
```

```
United States                              2818
India                                       972
United Kingdom                              419
Japan                                       245
South Korea                                 199
                                           ...
Romania, Bulgaria, Hungary                    1
Uruguay, Guatemala                            1
France, Senegal, Belgium                      1
Mexico, United States, Spain, Colombia        1
United Arab Emirates, Jordan                  1
Name: country, Length: 748, dtype: int64
```

```
df["date_added"].value_counts() # January 1, 2020 has high date_added freq
```

```
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
```

```
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
```

- As we check the null values rows in duration values are missing for movies type so we have to choose minutes values to fill

  So get the value count and check the highest freq duratio in movies and do fillna

```
df[df["duration"].isna()]
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	dur
5541	s5542	Movie	Louis C.K. 2017	Louis C.K.	Louis C.K.	United States	April 4, 2017	2017	74 min	
			Louis	Louis	Louis	United	September			

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

```
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
```

```
 ## Filling the director column null values using mode imputation and the same way other columns as well
df["director"] = df["director"].fillna(df["director"].value_counts().index[0])
df["cast"] = df["cast"].fillna(df["cast"].value_counts().index[0])
df["country"] = df["country"].fillna(df["country"].value_counts().index[0])
df["date_added"] = df["date_added"].fillna(df["date_added"].value_counts().index[0])
df["rating"] = df["rating"].fillna(df["rating"].value_counts().index[0])
df["duration"] = df["duration"].fillna(df["duration"].value_counts().index[3]) ## high freq movie duration value
```

```
df.info()
```

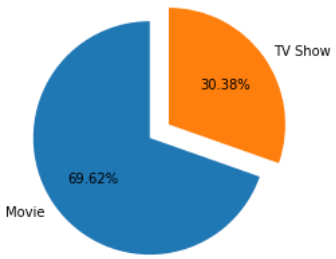
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   show_id       8807 non-null   object
 1   type          8807 non-null   object
 2   title         8807 non-null   object
 3   director      8807 non-null   object
 4   cast          8807 non-null   object
 5   country       8807 non-null   object
 6   date_added    8807 non-null   object
 7   release_year  8807 non-null   int64 
 8   rating        8807 non-null   object
 9   duration      8807 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
```

- Comparison of tv shows vs. movies.

```
## Univariant

type_count = df['type'].value_counts()

plt.pie(type_count, labels=type_count.index,autopct='%.2f%%',
       startangle=90,
       explode=(0.2,0))
plt.show()
```




```
uni_count = df.groupby("release_year")["title"].nunique().sort_values()

uni_count
```

```
release_year
1925       1
1966       1
1947       1
1961       1
1959       1
          ...
2016     902
2020     953
2019    1030
2017    1032
2018    1147
Name: title, Length: 74, dtype: int64
```

Pre-Processing Data(Unnesting)

1. Defining Problem Statement and Analysing basic metrics

a) ***Multiple values are available in cast , director , listed_in,country for each movie title so we cant get value count of each category properly***

So we have to **unnest** each column value w.r.t to title and rows will be increased accordingly and then merge with original data to get remaining columns values , And now we see final dataset - Netflix

Unnesting

```
#Code for Unnesting cast:
constraint=df['cast'].apply(lambda x: str(x).split(', ')).tolist()
df_cast=pd.DataFrame(constraint,index=df['title'])
df_cast=df_cast.stack()
df_cast=pd.DataFrame(df_cast)
df_cast.reset_index(inplace=True)
df_cast=df_cast[['title',0]]
df_cast.columns=['title','cast']
df_cast
```

	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
...	...	...
64946	Zubaan	Manish Chaudhary
64947	Zubaan	Meghna Malik
64948	Zubaan	Malkeet Rauni
64949	Zubaan	Anita Shabdish
64950	Zubaan	Chittaranjan Tripathy

64951 rows × 2 columns

```
# OTHER WAY using explode

small_df= df[['title','cast']]
small_df['cast'] = small_df['cast'].apply(lambda x: str(x).split(', '))
small_df= small_df.explode('cast')
small_df
```

```
<ipython-input-15-a001957bc799>:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guid
  small_df['cast'] = small_df['cast'].apply(lambda x: str(x).split(', '))
```

	title	cast
0	Dick Johnson Is Dead	nan
1	Blood & Water	Ama Qamata
1	Blood & Water	Khosi Ngema
1	Blood & Water	Gail Mabalane
1	Blood & Water	Thabang Molaba
...	...	...
8806	Zubaan	Manish Chaudhary
8806	Zubaan	Meghna Malik
8806	Zubaan	Malkeet Rauni
8806	Zubaan	Anita Shabdish
8806	Zubaan	Chittaranjan Tripathy

64951 rows × 2 columns

```
#Code for Unnesting director:
constraint=df['director'].apply(lambda x: str(x).split(', ')).tolist()
df_dir=pd.DataFrame(constraint,index=df['title'])
df_dir=df_dir.stack()
df_dir=pd.DataFrame(df_dir)
df_dir.reset_index(inplace=True)
df_dir=df_dir[['title',0]]
df_dir.columns=['title','director']
df_dir
```

	title	director
0	Dick Johnson Is Dead	Kirsten Johnson
1	Blood & Water	Rajiv Chilaka
2	Ganglands	Julien Leclercq
3	Jailbirds New Orleans	Rajiv Chilaka
4	Kota Factory	Rajiv Chilaka
...	...	...
9607	Zodiac	David Fincher
9608	Zombie Dumb	Rajiv Chilaka
9609	Zombieland	Ruben Fleischer
9610	Zoom	Peter Hewitt
9611	Zubaan	Mozez Singh

9612 rows × 2 columns

```
#Code for Unnesting country:
constraint=df['country'].apply(lambda x: str(x).split(', ')).tolist()
df_cont=pd.DataFrame(constraint,index=df['title'])
df_cont=df_cont.stack()
df_cont=pd.DataFrame(df_cont)
df_cont.reset_index(inplace=True)
df_cont=df_cont[['title',0]]
df_cont.columns=['title','country']
df_cont.head(15)
```

	title	country
0	Dick Johnson Is Dead	United States
1	Blood & Water	South Africa
2	Ganglands	United States
3	Jailbirds New Orleans	United States
4	Kota Factory	India
5	Midnight Mass	United States
6	My Little Pony: A New Generation	United States
7	Sankofa	United States
8	Sankofa	Ghana
9	Sankofa	Burkina Faso
10	Sankofa	United Kingdom
11	Sankofa	Germany
12	Sankofa	Ethiopia

```
#Code for Unnesting listed_in/genre:
constraint=df['listed_in'].apply(lambda x: str(x).split(', ')).tolist()
df_gen=pd.DataFrame(constraint,index=df['title'])
df_gen=df_gen.stack()
df_gen=pd.DataFrame(df_gen)
df_gen.reset_index(inplace=True)
df_gen=df_gen[['title',0]]
df_gen.columns=['title','listed_in']
df_gen
```

	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
...	...	...
19318	Zoom	Children & Family Movies
19319	Zoom	Comedies
19320	Zubaan	Dramas
19321	Zubaan	International Movies
19322	Zubaan	Music & Musicals

19323 rows × 2 columns

```
# merging each sub unnested data of title,cast,director,listed_in(genre)

df_c_d = df_dir.merge(df_cast, on="title")
df_c_d = df_c_d[["title","director","cast"]]
df_cdc = df_c_d.merge(df_cont, on="title")
df_cdc = df_cdc[["title","director","cast","country"]]
df_final = df_cdc.merge(df_gen, on="title")
df_final = df_final[["title","director","cast","country","listed_in"]]
df_final
```

	title	director	cast	country	listed_in
0	Dick Johnson Is Dead	Kirsten Johnson	David Attenborough	United States	Documentaries
1	Blood & Water	Rajiv Chilaka	Ama Qamata	South Africa	International TV Shows

```
## merging the final data with original data to get remaining columns

netflix = df_final.merge(df,on="title")
netflix.drop(["director_y","cast_y","country_y","listed_in_y"], axis=1,inplace = True)
netflix.rename({"director_x":"director","cast_x":"cast","country_x":"country","listed_in_x":"listed_in"},axis=1, inplace = True)
netflix
```

	title	director	cast	country	listed_in	show_id	type	date_added	re
0	Dick Johnson Is Dead	Kirsten Johnson	David Attenborough	United States	Documentaries	s1	Movie	September 25, 2021	
1	Blood & Water	Rajiv Chilaka	Ama Qamata	South Africa	International TV Shows	s2	TV Show	September 24, 2021	
2	Blood & Water	Rajiv Chilaka	Ama Qamata	South Africa	TV Dramas	s2	TV Show	September 24, 2021	
3	Blood & Water	Rajiv Chilaka	Ama Qamata	South Africa	TV Mysteries	s2	TV Show	September 24, 2021	

```
# Name of the director who has most no.of movies released over all

netflix.groupby("director")["title"].nunique().sort_values()

# So Rajiv Chilaka has the most no.of movies/tv shows streaming on netflix among all the directors
```

```
director
Jesse Adang         1
Lisa Arnold         1
Lisa Cortés         1
Liu Bang-yao        1
Liu Jiang           1
                  ...
Marcus Raboy       16
Suhas Kadav        16
Raúl Campos        19
Jan Suter          21
Rajiv Chilaka    2656
Name: title, Length: 4993, dtype: int64
```

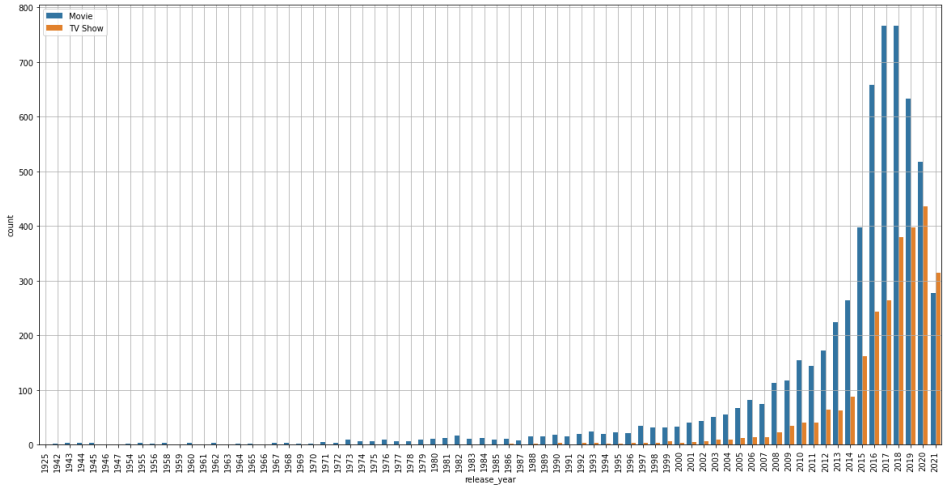
How has the number of movies released per year changed over the last 20-30 years?

The no.of movies released per year has been drastically increased over past 20-30 years is as below

```
# Bivariant - Numeric & Category

plt.figure(figsize=(20,10))
sns.countplot(data = df, x="release_year", hue="type")   ## OR KDE plot can be used
plt.xticks(rotation= 90)
plt.legend(loc="upper left")
plt.grid()
plt.show
```

```
<function matplotlib.pyplot.show(*args, **kw)>
```



TypeCasting **data_added** column to datetime datatype,

By doing so, we will be getting to know month wise and year wise movies/tv shows popularity/ demand & which month can be the best for releasing movies/series for netflix.

```
df["date_added"]=pd.to_datetime(df["date_added"])
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      8807 non-null   object        
 4   cast          8807 non-null   object        
 5   country       8807 non-null   object        
 6   date_added    8807 non-null   datetime64[ns]
 7   release_year  8807 non-null   int64         
 8   rating        8807 non-null   object        
 9   duration      8807 non-null   object        
 10  listed_in     8807 non-null   object        
 11  description   8807 non-null   object        
dtypes: datetime64[ns](1), int64(1), object(10)
memory usage: 825.8+ KB
```

```
df["date_added_year"] = df["date_added"].dt.year
df["date_added_month"] = df["date_added"].dt.month
df["date_added_day"] = df["date_added"].dt.day_name()
```

What is the best time to launch a TV show?

As per plots below, TV shows addition to netflix is recommende for **Nov & December** months are the best time to launch TV shows as seasonal holidays available like Thanks giving/Christmas eve & year end holidays all come along, so people with families can sit along and watch TV shows and it ultimately gains popularity.

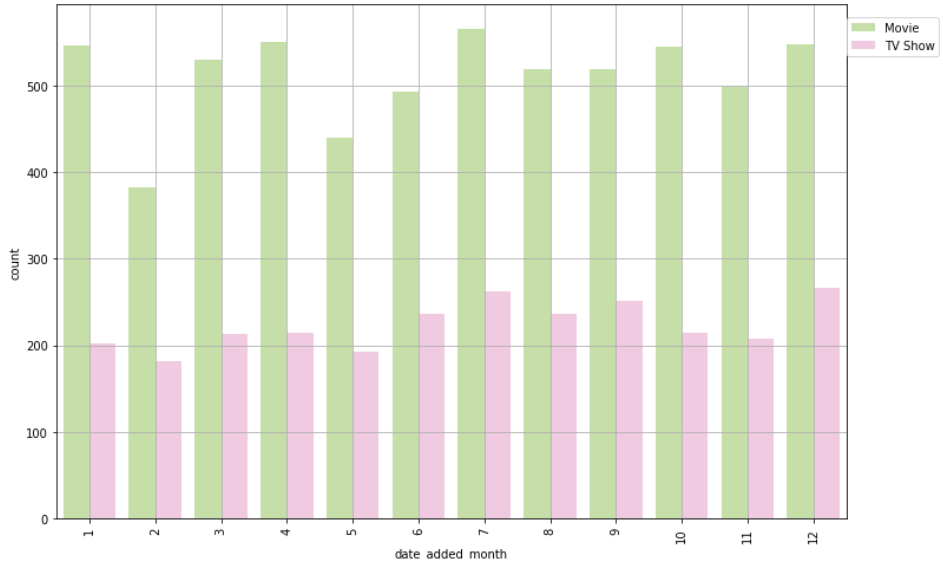
And as per Weeks, Weekend(Sat/Sun) could be THE best time to launch a TV show as there is value count and can attract more users over weekends.

```
# Bivariant - Numeric & Category(Dodged Plot)

# Month wise count of movies & TV shows added to Netflix

plt.figure(figsize=(12,8))
sns.countplot(data = df, x="date_added_month", hue="type", palette = "PiYG_r")
plt.xticks(rotation= 90)
plt.legend(loc=(1.0,0.9))
plt.grid()
plt.show
```

```
<function matplotlib.pyplot.show(*args, **kw)>
```

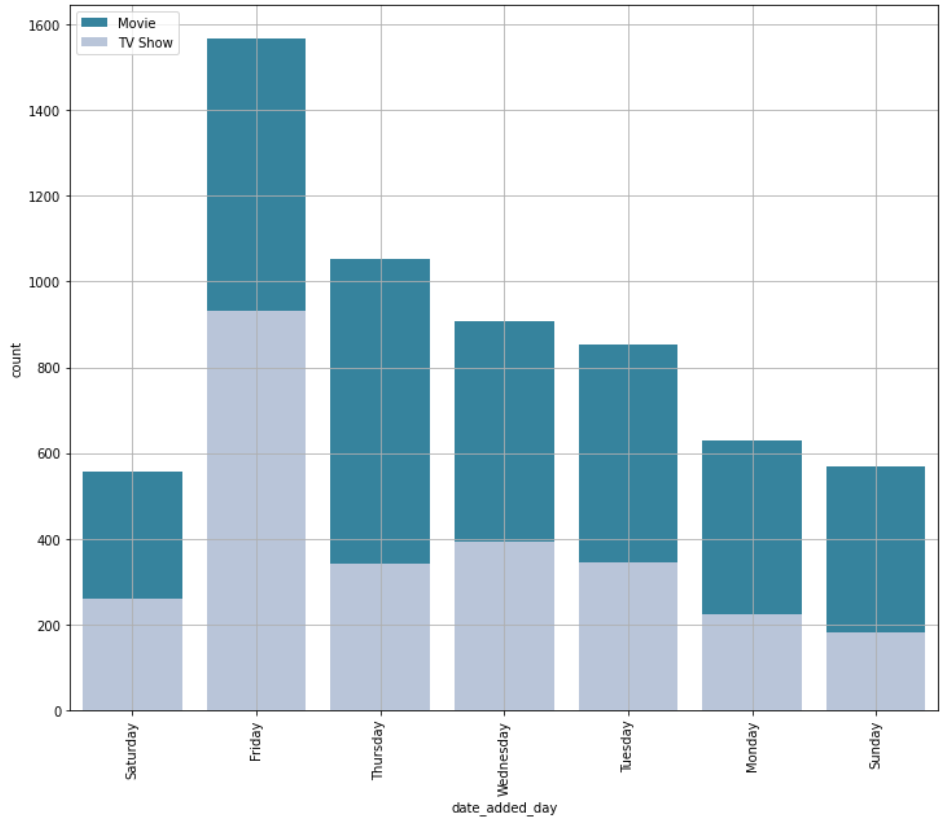



```
# Bivariant - Numeric & Category(Stacked plot)

# week wise count of movies & TV shows added to Netflix

plt.figure(figsize=(12,10))
sns.countplot(data = df, x="date_added_day", hue="type", dodge=False, palette = "PuBuGn_r")
plt.xticks(rotation= 90)
plt.legend(loc="upper left")
plt.grid()
plt.show
```

```
<function matplotlib.pyplot.show(*args, **kw)>
```



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

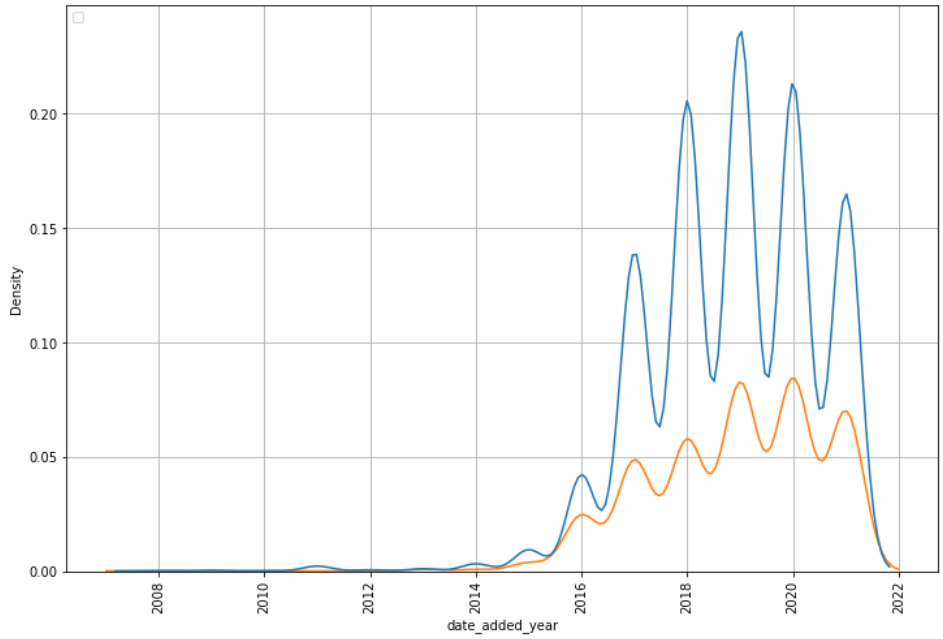
As we can see from below plot, netflix focus on movies is higher as usual and recommended to release more TV shows as it has less count over all.

```
# Year wise count of movies & TV shows added to Netflix

plt.figure(figsize=(12,8))
sns.kdeplot(data = df, x="date_added_year", hue="type")   ## OR KDE plot can be used
```

```
plt.xticks(rotation= 90)
plt.legend(loc="upper left")
plt.grid()
plt.show
```

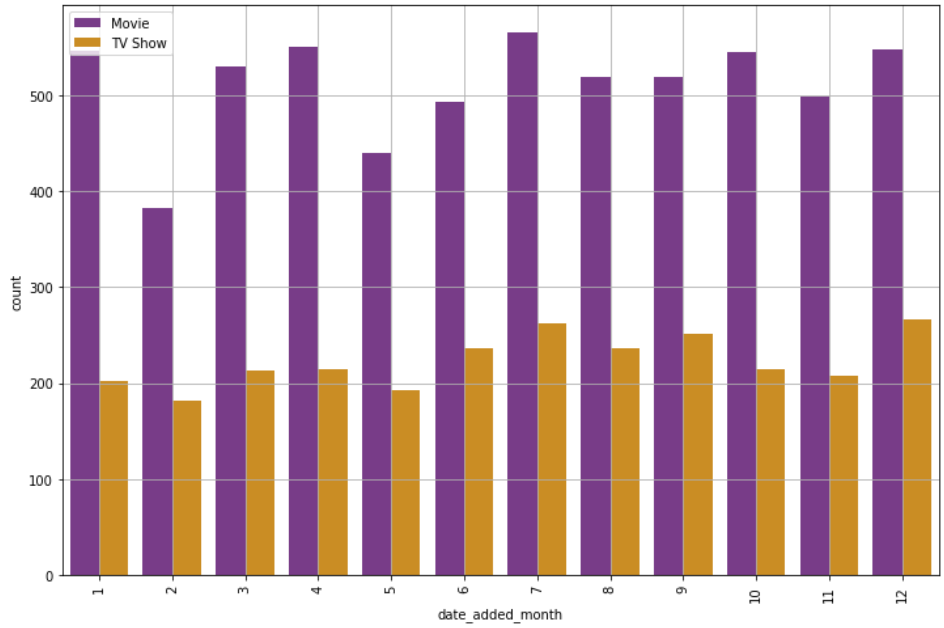
```
WARNING:matplotlib.legend:No handles with labels found to put in legend.
<function matplotlib.pyplot.show(*args, **kw)>
```




```
# Month wise count of movies & TV shows added to Netflix

plt.figure(figsize=(12,8))
sns.countplot(data = df, x="date_added_month", hue="type", palette = "CMRmap")
plt.xticks(rotation= 90)
plt.legend(loc="upper left")
plt.grid()
plt.show
```

```
<function matplotlib.pyplot.show(*args, **kw)>
```



c)**Type Casting & Modifications on Duration Column**

Inorder to so we can extract the int values from **duration** object datatype and then we can get to know average runtime/seasons for movies/TV Shows runtime

```
duration_int = netflix['duration'].apply(lambda x: str(x).split(' ')[0]).tolist()

netflix['duration'] = pd.DataFrame(duration_int)
netflix['duration'] = netflix['duration'].astype(int)
```

```
netflix
```

	title	director	cast	country	listed_in	show_id	type	date_added	re
0	Dick Johnson Is Dead	Kirsten Johnson	David Attenborough	United States	Documentaries	s1	Movie	September 25, 2021	
1	Blood & Water	Rajiv Chilaka	Ama Qamata	South Africa	International TV Shows	s2	TV Show	September 24, 2021	
2	Blood & Water	Rajiv Chilaka	Ama Qamata	South Africa	TV Dramas	s2	TV Show	September 24, 2021	
3	Blood & Water	Rajiv Chilaka	Ama Qamata	South Africa	TV Mysteries	s2	TV Show	September 24, 2021	

```
netflix["release_year"] = netflix["release_year"].astype('int')
netflix["date_added"] = pd.to_datetime(netflix["date_added"])
netflix["date_added_year"] = netflix["date_added"].dt.year
netflix["date_added_month"] = netflix["date_added"].dt.month
```

```
netflix.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 201991 entries, 0 to 201990
Data columns (total 14 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   title             201991 non-null  object
 1   director          201991 non-null  object
 2   cast              201991 non-null  object
 3   country           201991 non-null  object
 4   listed_in         201991 non-null  object
 5   show_id           201991 non-null  object
 6   type              201991 non-null  object
 7   date_added        201991 non-null  datetime64[ns]
 8   release_year      201991 non-null  int64
 9   rating            201991 non-null  object
 10  duration          201991 non-null  int64
 11  description       201991 non-null  object
 12  date_added_year   201991 non-null  int64
 13  date_added_month  201991 non-null  int64
dtypes: datetime64[ns](1), int64(4), object(9)
memory usage: 23.1+ MB
```

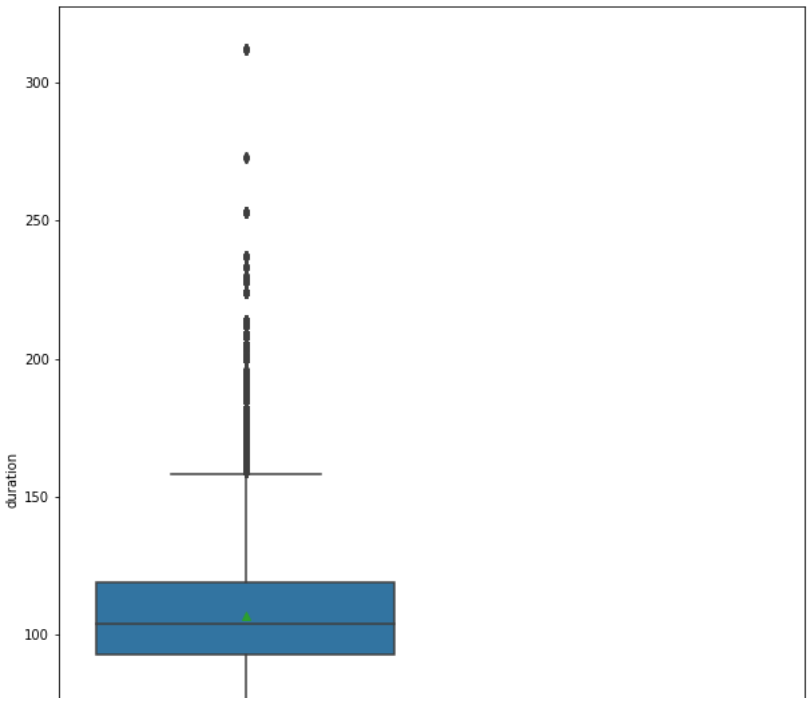
```
# Average runtime for movies / tv shows
netflix.groupby("type")["duration"].median()

# "So we can see the average movie runtime is 104 mins and TV Shows is 1 season"
```

```
type
Movie      104.0
TV Show      1.0
Name: duration, dtype: float64
```

```
# N & C - Bivariate     & we have outliers both for movies & TV shows duration
plt.figure(figsize=(10,13))
sns.boxplot(data = netflix, x="type" ,y="duration", showmeans=True)
plt.show
```

```
<function matplotlib.pyplot.show(*args, **kw)>
```



Understanding what content is available in different countries

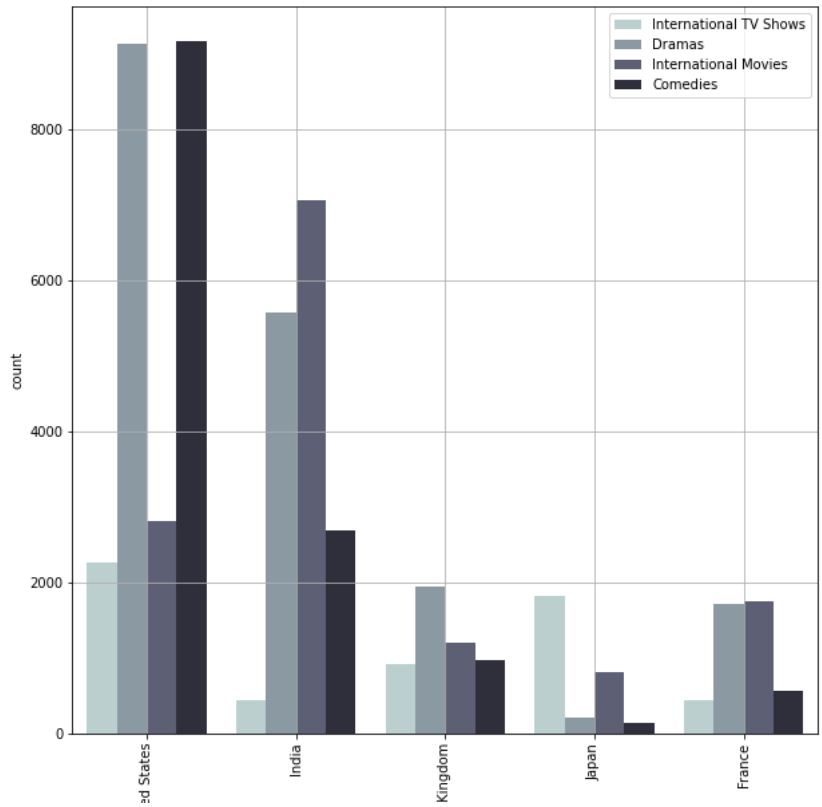
Below is the example for top 5 countries and top 4 content/genre/listed in over all data

```
top4_gen = netflix["listed_in"].value_counts().index[:4]
top5_country = netflix["country"].value_counts().index[:5]
top5_data = netflix.loc[(netflix["listed_in"].isin(top4_gen))&(netflix["country"].isin(top5_country))]
top5_data
```

	title	director	cast	country	listed_in	show_id	type	date_added	rel
59	Ganglands	Julien Leclercq	Sami Bouajila	United States	International TV Shows	s3	TV Show	2021-09-24	
62	Ganglands	Julien Leclercq	Tracy Gotoas	United States	International TV Shows	s3	TV Show	2021-09-24	
65	Ganglands	Julien Leclercq	Samuel Jouy	United States	International TV Shows	s3	TV Show	2021-09-24	
68	Ganglands	Julien Leclercq	Nabiha Akkari	United States	International TV Shows	s3	TV Show	2021-09-24	
71	Ganglands	Julien Leclercq	Sofia Lesaffre	United States	International TV Shows	s3	TV Show	2021-09-24	
...	...	...	...	...	...	...	...	...	...
201983	Zubaan	Mozez Singh	Malkeet Rauni	India	International Movies	s8807	Movie	2019-03-02	

```
plt.figure(figsize=(10,10))
sns.countplot(data = top5_data, x="country" ,hue ="listed_in", palette = "bone_r")   ## OR KDE plot can be used
plt.xticks(rotation= 90)
plt.legend(loc="upper right")
plt.grid()
plt.show
```

```
<function matplotlib.pyplot.show(*args, **kw)>
```

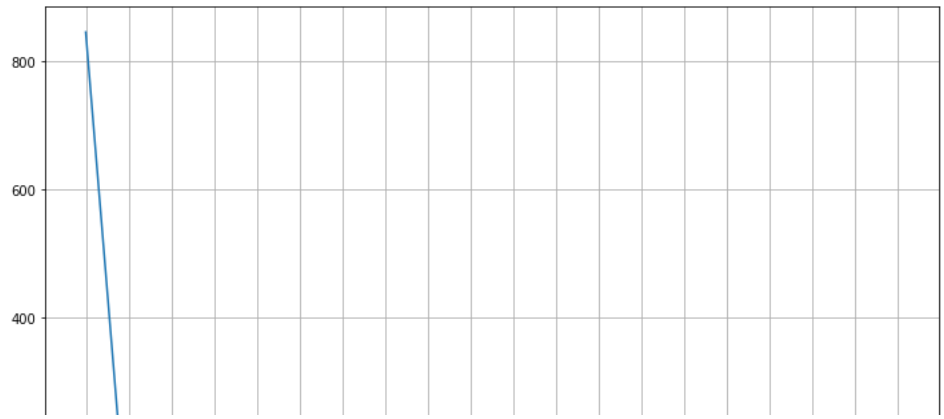



```
cast = netflix.groupby("cast")["title"].nunique()
cast = cast.sort_values(ascending = False)
cast_top = cast[:20]
cast_top
```

```
cast
David Attenborough    845
Anupam Kher            43
Shah Rukh Khan         35
Julie Tejwani          33
Takahiro Sakurai       32
Naseeruddin Shah       32
Rupa Bhimani           31
Akshay Kumar           30
Om Puri                30
Yuki Kaji              29
Paresh Rawal           28
Amitabh Bachchan       28
Boman Irani            27
Rajesh Kava            26
Vincent Tong           26
Andrea Libman          25
Kareena Kapoor         25
Samuel L. Jackson      24
John Cleese            24
Fred Tatasciore        23
Name: title, dtype: int64
```

```
plt.figure(figsize=(12,8))
sns.lineplot(data=cast_top , x = cast_top.index , y = cast_top.values)
plt.xticks(rotation= 90)
plt.grid()
plt.show()
```

```
## David Attenborough is the most popular Cast who has done most no.of movies from over all data.
```

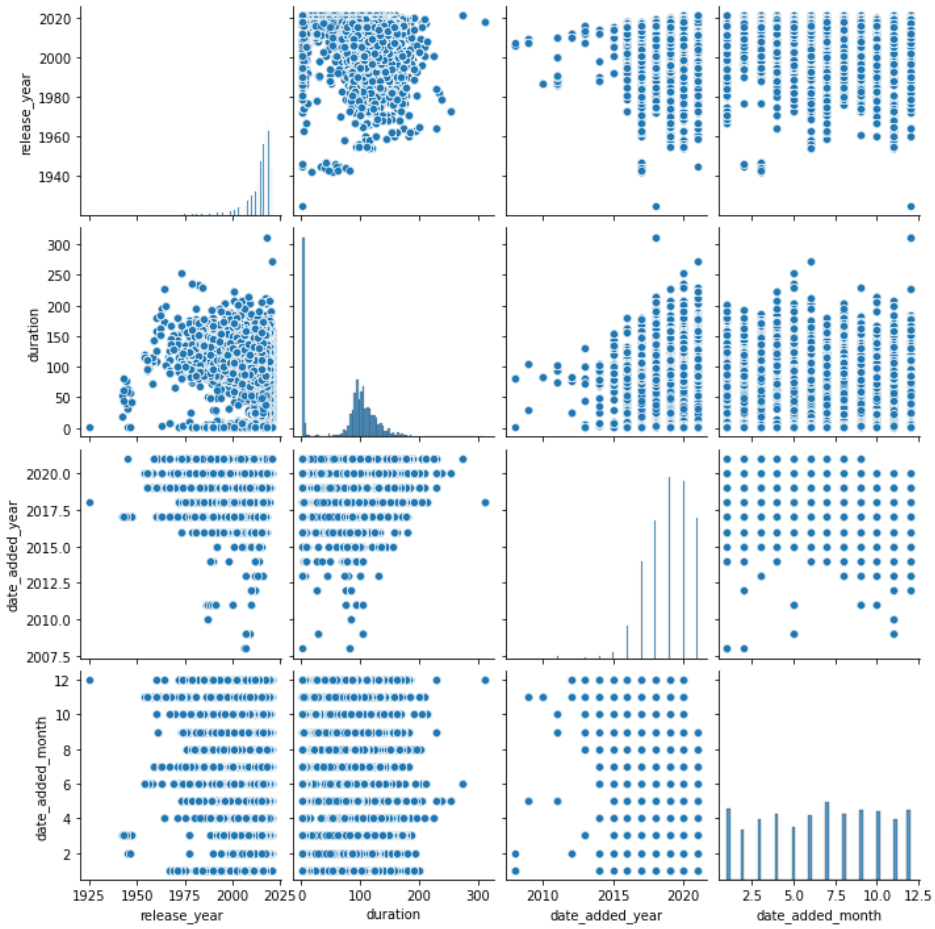



```
netflix.loc[netflix["cast"] == "David Attenborough"]["title"].unique()
```

Pair Plot

```
sns.pairplot(data=netflix)
```

```
<seaborn.axisgrid.PairGrid at 0x7fb1dd1c9d30>
```



Correlation

```
netflix.corr()
```

	release_year	duration	date_added_year	date_added_month
release_year	1.000000	-0.304545	0.052838	-0.031314
duration	-0.304545	1.000000	0.016498	-0.001705
date_added_year	0.052838	0.016498	1.000000	-0.167556
date_added_month	-0.031314	-0.001705	-0.167556	1.000000

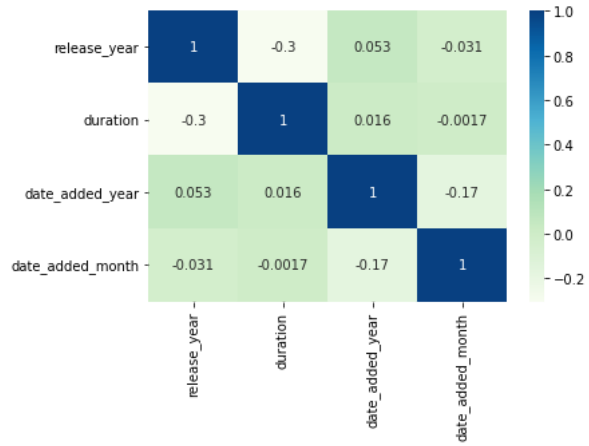
Heat Map

From below Heat map, There is no coulumn that is strongly correlated to each other then itself. And to describe below briefly

-0.17 -> least/not-strong negatively co-related 0.3 -> least/not-strong positively co-related

```
sns.heatmap(netflix.corr(), cmap="GnBu", annot=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb1dc093370>
```



Insights based on Non-Graphical and Visual Analysis

Insights

1) Most no.of movies were released in 2017 and TV shows in year 2020

2) Most no.of movies added to netflix is in 2019 & TV Shows in year 2020

3) Top 5 Countries where most no.of movies/TV shows were released and its mostly released Genre

United States main focus is on Dramas & Comedies Genre

India main focus is on International Movies Genre

United Kingdom main focus is on Dramas Genre

Likewise Japan on International TV Shows, France on Dramas&International Movies

4) Average runtime for movies / tv shows is 104 Mins / 1 Season

5) Netflix Data is mostly focused on and being added Movies compared to TV shows

6) David Attenborough is the most popular Cast who has done most no.of movies from over all data.

7) Movies & TV shows addition to netflix has increased tremendously over past year till 2019 and then the count is being dropped as per data

Recommendations

1)There is no seasonality observed w.r.t to Months so, This is an area which can be worked upon to boost popularity and user subscrption

Recommended to add TV shows to netflix for the month of **Nov & December** months as seasonal holidays like Thanks giving/Christmas eve & year end holidays all come along, so there would be more chances to gain popularity of this platform in users

2) And as per Weekdays, Weekend(Sat/Sun) could be THE best time to launch a TV show/movies as there is less value count and can attract more users over weekends.

3) Most of the Movies are being released/ being added from United States, India. Netflix can focus on other countries to attract more customers and boost subscriptions

4) Highest Avg seasons for TV shows is 1 season which is low. This can be worked upon to get TV shows with more no. of seasons and can attract TV shows loves more.

5) Movies & TV shows addition to netflix has increased tremendously over past year till 2019 and then the count is being dropped as per data and this can be the area to be worked upon most.