```
# Import the libraries that will be used
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
# load the data set
# !gdown 1QBxu4XhFpoqDiXsyVU4w56MROHn1jZQM
```

```
!gdown 1QBxu4XhFpoqDiXsyVU4w56MROHn1jZQM
```

```
Downloading...
From: https://drive.google.com/uc?id=1QBxu4XhFpoqDiXsyVU4w56MROHn1jZQM
To: /content/netflix.csv
100% 3.40M/3.40M [00:00<00:00, 193MB/s]
```

```
netflix=pd.read_csv('netflix.csv')
```

```
#explore the data
```

```
netflix.head()
```

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

```
netflix.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
```

```
netflix.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
date_added	8797	1767	January 1, 2020	109
rating	8803	17	TV-MA	3207
duration	8804	220	1 Season	1793

```
netflix.shape
```

```
(8807, 12)
```

```
netflix.isnull().sum()
```

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

```
netflix.isnull().sum()*100/len(netflix)
```

```
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
```

```
#Count and Percentage of different show types in dataset
```

```
netflix['type'].value_counts()
```

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

```
netflix['type'].value_counts(normalize=True)*100
```

```
Movie      69.615079
TV Show    30.384921
Name: type, dtype: float64
```

```
#Data Preprocessing
#Unnesting the cast,listed_in , country columns
```

```
cast=netflix.loc[:,['show_id','cast']]
genre=netflix.loc[:,['show_id','listed_in']]
country=netflix.loc[:,['show_id','country']]
```

```
cast['cast']=cast['cast'].str.split(',')


cast=cast.explode('cast',ignore_index=True)


netflix_df=netflix.merge(cast,how='inner',on='show_id')


# rename cast_x and cast_y


netflix_df=netflix_df.rename({'cast_x':'cast_combined','cast_y':'cast'},axis=1)


#Unnesting the listed_in column


genre=netflix.loc[:,['show_id','listed_in']]


genre['listed_in']=genre['listed_in'].str.split(',')


genre=genre.explode('listed_in',ignore_index=True)


netflix_df=netflix_df.merge(genre,on='show_id',how='inner')


netflix_df=netflix_df.rename({'listed_in_x':'listed_in','listed_in_y':'genre'},axis=1)



#Unnesting the country column


country['country']=country['country'].str.split(',')


country=country.explode('country',ignore_index=True)


netflix_df=netflix_df.merge(country,how='inner',on='show_id')


netflix_df=netflix_df.rename({'country_x':'country_combined','country_y':'country'},axis=1)


# Dealing with null value


netflix_test=netflix_df.copy()


#Impute missing cast with NA


netflix_test['cast_combined']=netflix_test['cast_combined'].fillna('NA')
netflix_test['cast']=netflix_test['cast'].fillna('NA')


#We see missing values in duration column are present in rating column, so we copy them


netflix_test[netflix_test['duration'].isnull()]
netflix_test.loc[netflix_test['duration'].isnull(),'duration']=netflix_test.loc[netflix_test['duration'].isnull(),'rating']


netflix[netflix['duration'].isnull()]
netflix.loc[netflix['duration'].isnull(),'duration']=netflix.loc[netflix['duration'].isnull(),'rating']


netflix['rating'].unique()
```

```
    array(['PG-13', 'TV-MA', 'PG', 'TV-14', 'TV-PG', 'TV-Y', 'TV-Y7', 'R',
           'TV-G', 'G', 'NC-17', '74 min', '84 min', '66 min', 'NR', nan,
           'TV-Y7-FV', 'UR'], dtype=object)
```

```
#We can see few errogrneous values in rating column, so we remove them


netflix_test.loc[(netflix_test['rating']=='74 min')|(netflix_test['rating']=='84 min')|(netflix_test['rating']=='66 min'),'rating']=np.nan
```

```
netflix_test['rating'].isnull().sum()

     70
```

```
#creating a rating lookup dataframe
```

```
rating=netflix_test.groupby(['type','genre'])['rating'].agg(pd.Series.mode).reset_index()
```

```
rating['rating'].value_counts()


     TV-MA    36
     TV-14    15
     R         8
     TV-PG     5
     PG        4
     TV-Y7     3
     PG-13     2
     Name: rating, dtype: int64
```

```
netflix_test=netflix_test.merge(rating,how='left',on=['type','genre'])
```

```
netflix_test=netflix_test.rename({'rating_x':'rating','rating_y':'rating_mode'},axis=1)
```

```
netflix_test['rating'].fillna(netflix_test['rating_mode'],inplace=True)
```

```
netflix_test['rating'].fillna('NA',inplace=True)
```

```
netflix_test.drop(columns=['rating_mode'],inplace=True)
```

```
#we will be using group by and bfill() , ffill() to impute null values in date_added column
```

```
def date_add(df):
  df['date_added'].bfill(inplace=True)
  df['date_added'].ffill(inplace=True)
  return df
```

```
netflix_test=netflix_test.groupby(['release_year']).apply(date_add)
netflix=netflix.groupby(['release_year']).apply(date_add)

     <ipython-input-41-07770c1e50bf>:1: FutureWarning: Not prepending group keys to the result index of transform-like apply. In the future,
     To preserve the previous behavior, use

             >>> .groupby(..., group_keys=False)

     To adopt the future behavior and silence this warning, use

             >>> .groupby(..., group_keys=True)
       netflix_test=netflix_test.groupby(['release_year']).apply(date_add)
     <ipython-input-41-07770c1e50bf>:2: FutureWarning: Not prepending group keys to the result index of transform-like apply. In the future,
     To preserve the previous behavior, use

             >>> .groupby(..., group_keys=False)

     To adopt the future behavior and silence this warning, use

             >>> .groupby(..., group_keys=True)
       netflix=netflix.groupby(['release_year']).apply(date_add)
```

```
netflix_test['date_added']=pd.to_datetime(netflix_test['date_added'],errors='ignore')
netflix['date_added']=pd.to_datetime(netflix['date_added'],errors='ignore')
```

```
netflix_test['year_added']=netflix_test['date_added'].dt.year
netflix_test['month_added']=netflix_test['date_added'].dt.month
netflix_test['week_added']=netflix_test['date_added'].dt.week
netflix['year_added']=netflix['date_added'].dt.year
netflix['month_added']=netflix['date_added'].dt.month
```

```
netflix['week_added']=netflix['date_added'].dt.week
```

```
<ipython-input-43-797bb0ea98d9>:3: FutureWarning: Series.dt.weekofyear and Series.dt.week have been deprecated. Please use Series.dt.isc
  netflix_test['week_added']=netflix_test['date_added'].dt.week
<ipython-input-43-797bb0ea98d9>:6: FutureWarning: Series.dt.weekofyear and Series.dt.week have been deprecated. Please use Series.dt.isc
  netflix['week_added']=netflix['date_added'].dt.week
```

```
netflix_test['year_added']=netflix_test['year_added'].astype('int64',errors='ignore')
netflix_test['month_added']=netflix_test['month_added'].astype('int64',errors='ignore')
netflix_test['week_added']=netflix_test['week_added'].astype('int64',errors='ignore')
netflix['year_added']=netflix['year_added'].astype('int64',errors='ignore')
netflix['month_added']=netflix['month_added'].astype('int64',errors='ignore')
netflix['week_added']=netflix['week_added'].astype('int64',errors='ignore')
```

```
#creating a directors lookup dataframe
```

```
directors=netflix_test.groupby(['type','country','genre'])['director'].agg(pd.Series.mode).reset_index()
```

```
netflix_test=netflix_test.merge(directors,how='left',on=['type','country','genre'])
```

```
netflix_test=netflix_test.rename({'director_x':'director','director_y':'director_mode'},axis=1)
```

```
netflix_test['director'].fillna(netflix_test['director_mode'],inplace=True)
```

```
netflix_test['director'].fillna('NA',inplace=True)
```

```
netflix_test.drop(columns=['director_mode'],inplace=True)
```

```
#Imputation of country column
```

```
netflix_test['country'].fillna('Unknown',inplace=True)
```

```
netflix_test['country_combined'].fillna('Unknown',inplace=True)
```

```
#Checking for data consistency
```

```
netflix_test['type'].value_counts()
```

```
Movie      131931
TV Show     54468
Name: type, dtype: int64
```

```
netflix_test['rating'].value_counts()
```

```
TV-MA       67734
TV-14       42054
R           23990
PG-13       15233
TV-PG       13778
PG           9011
TV-Y7        5804
TV-Y         3152
TV-G         2650
NR           1521
G            1151
NC-17         149
TV-Y7-FV       86
UR             86
Name: rating, dtype: int64
```

```
netflix_test['duration'].value_counts()
```

```
1 Season     33444
2 Seasons     9470
3 Seasons     5084
94 min        3591
```

```
97 min        3434
             ...
5 min            3
9 min            2
3 min            2
11 min           2
8 min            1
Name: duration, Length: 220, dtype: int64
```

```
netflix_test[['dur','col2']]=netflix_test['duration'].str.split(pat=' ',expand=True)
netflix_test.drop(columns=['col2','duration'],inplace=True)
netflix_test=netflix_test.rename({'dur':'duration'},axis=1)
```

```
netflix[['dur','col2']]=netflix['duration'].str.split(pat=' ',expand=True)
netflix.drop(columns=['col2','duration'],inplace=True)
netflix=netflix.rename({'dur':'duration'},axis=1)
```

```
netflix_test['genre'].value_counts().index
```

```
Index([' International Movies', 'Dramas', 'Comedies', 'Action & Adventure',
       ' Dramas', ' Independent Movies', ' TV Dramas',
       'Children & Family Movies', 'International TV Shows',
       ' Romantic Movies', ' Thrillers', ' Comedies',
       ' International TV Shows', 'Crime TV Shows', 'Kids' TV', ' TV Comedies',
       ' Sci-Fi & Fantasy', 'Horror Movies', ' Music & Musicals',
       ' Romantic TV Shows', 'Anime Series', ' Spanish-Language TV Shows',
       'Documentaries', 'British TV Shows', ' TV Action & Adventure',
       ' Sports Movies', ' TV Mysteries', 'TV Comedies', ' Korean TV Shows',
       'International Movies', ' Horror Movies', ' TV Sci-Fi & Fantasy',
       'TV Dramas', 'Classic Movies', ' Cult Movies', 'Thrillers',
       ' LGBTQ Movies', ' Kids' TV', ' Teen TV Shows', ' TV Horror',
       ' TV Thrillers', ' Crime TV Shows', ' Faith & Spirituality',
       'TV Action & Adventure', ' Anime Features', 'Docuseries',
       'Stand-Up Comedy', ' Classic Movies', ' Children & Family Movies',
       'Reality TV', 'Movies', ' Reality TV', 'Romantic TV Shows',
       ' Docuseries', 'Independent Movies', 'Classic & Cult TV',
       'Anime Features', 'TV Horror', 'Stand-Up Comedy & Talk Shows',
       'Cult Movies', ' Science & Nature TV', 'Sci-Fi & Fantasy',
       ' Documentaries', 'TV Shows', ' Stand-Up Comedy & Talk Shows',
       'Music & Musicals', 'Spanish-Language TV Shows', ' Classic & Cult TV',
       'Romantic Movies', ' Stand-Up Comedy', 'TV Sci-Fi & Fantasy',
       'LGBTQ Movies', 'Sports Movies'],
      dtype='object')
```

```
netflix_test['genre']=netflix_test['genre'].str.lstrip()
netflix_test['genre']=netflix_test['genre'].str.rstrip()
```

```
netflix_test['genre'].nunique()
```

```
42
```

```
netflix_test.loc[netflix_test['genre'].str.startswith('D'),['genre']].value_counts()
```

```
genre
Dramas           27799
Documentaries     2042
Docuseries         801
dtype: int64
```

```
netflix_test['country']=netflix_test['country'].str.lstrip()
netflix_test['country']=netflix_test['country'].str.rstrip()
```

```
netflix_test['country'].value_counts().index
```

```
Index(['United States', 'India', 'United Kingdom', 'Unknown', 'Japan',
       'France', 'Canada', 'Spain', 'South Korea', 'Germany',
       ...
       'Afghanistan', 'Sri Lanka', 'Mongolia', 'Panama', 'Armenia', 'Samoa',
       'Botswana', 'Nicaragua', 'Kazakhstan', 'Uganda'],
      dtype='object', length=124)
```

```
netflix_test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 186399 entries, 0 to 186398
```

```
Data columns (total 18 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   show_id           186399 non-null  object
 1   type              186399 non-null  object
 2   title             186399 non-null  object
 3   director          186399 non-null  object
 4   cast_combined     186399 non-null  object
 5   country_combined  186399 non-null  object
 6   date_added        186399 non-null  datetime64[ns]
 7   release_year      186399 non-null  int64
 8   rating            186399 non-null  object
 9   listed_in         186399 non-null  object
 10  description       186399 non-null  object
 11  cast              186399 non-null  object
 12  genre             186399 non-null  object
 13  country           186399 non-null  object
 14  year_added        186399 non-null  int64
 15  month_added       186399 non-null  int64
 16  week_added        186399 non-null  int64
 17  duration          186399 non-null  object
dtypes: datetime64[ns](1), int64(4), object(13)
memory usage: 27.0+ MB
```

```
netflix_test['type'].value_counts()
```

```
Movie      131931
TV Show     54468
Name: type, dtype: int64
```

```
#Create separate data frame for Movies and TV Series
```

```
movies=netflix_test.loc[netflix_test['type']=='Movie']
```

```
series=netflix_test.loc[netflix_test['type']=='TV Show']
```

```
#Datasets for Analysis
#netflix --> original dataset
#netflix_test --> preprocessed dataset
#movies --> dataset containing only movies data
#series --> dataset containing only TV series data
```

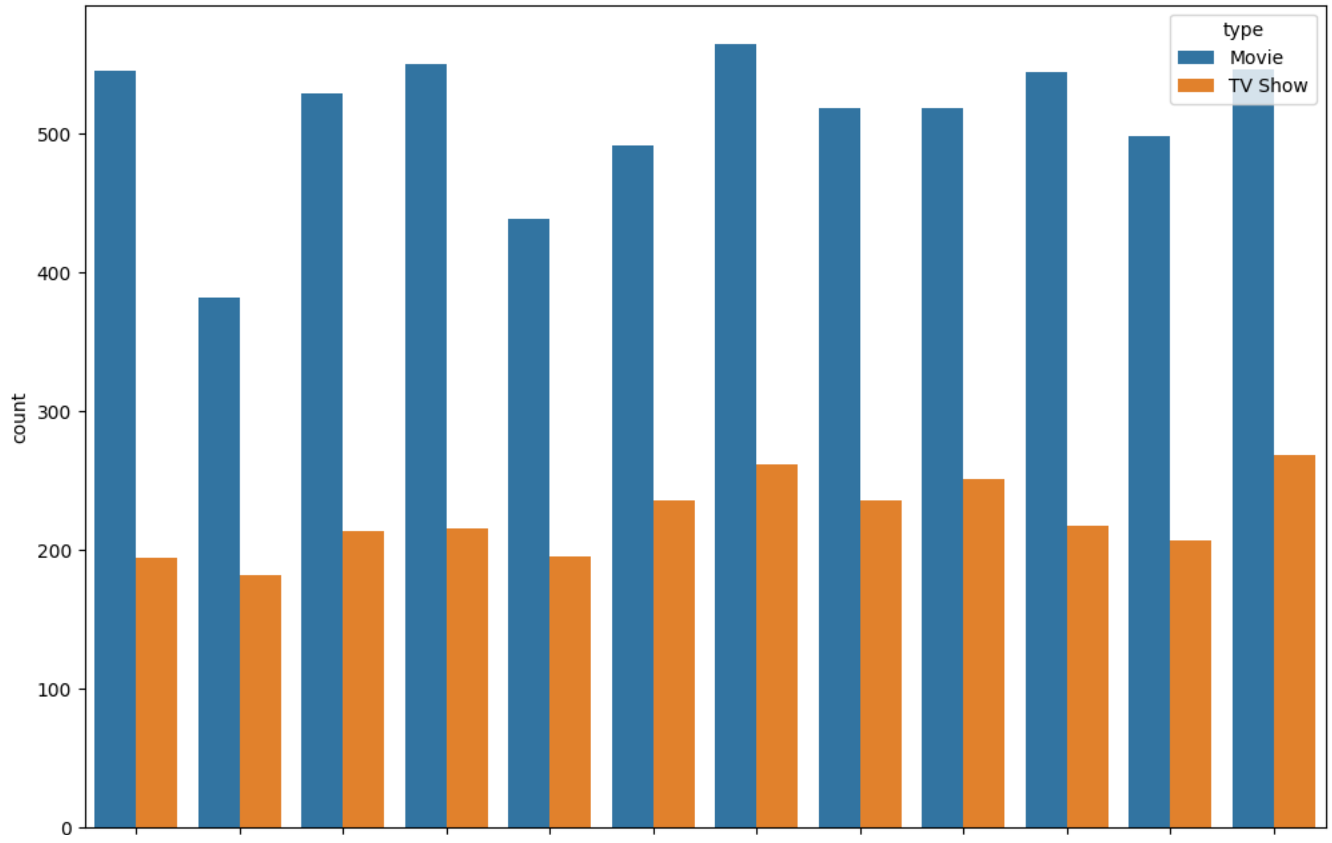
```
#Content Release in each month and week
```

```
netflix.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 8807 entries, 0 to 8806
Data columns (total 15 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    8807 non-null   datetime64[ns]
 7   release_year  8807 non-null   int64
 8   rating        8803 non-null   object
 9   listed_in     8807 non-null   object
 10  description   8807 non-null   object
 11  year_added    8807 non-null   int64
 12  month_added   8807 non-null   int64
 13  week_added    8807 non-null   int64
 14  duration      8807 non-null   object
dtypes: datetime64[ns](1), int64(4), object(10)
memory usage: 1.3+ MB
```

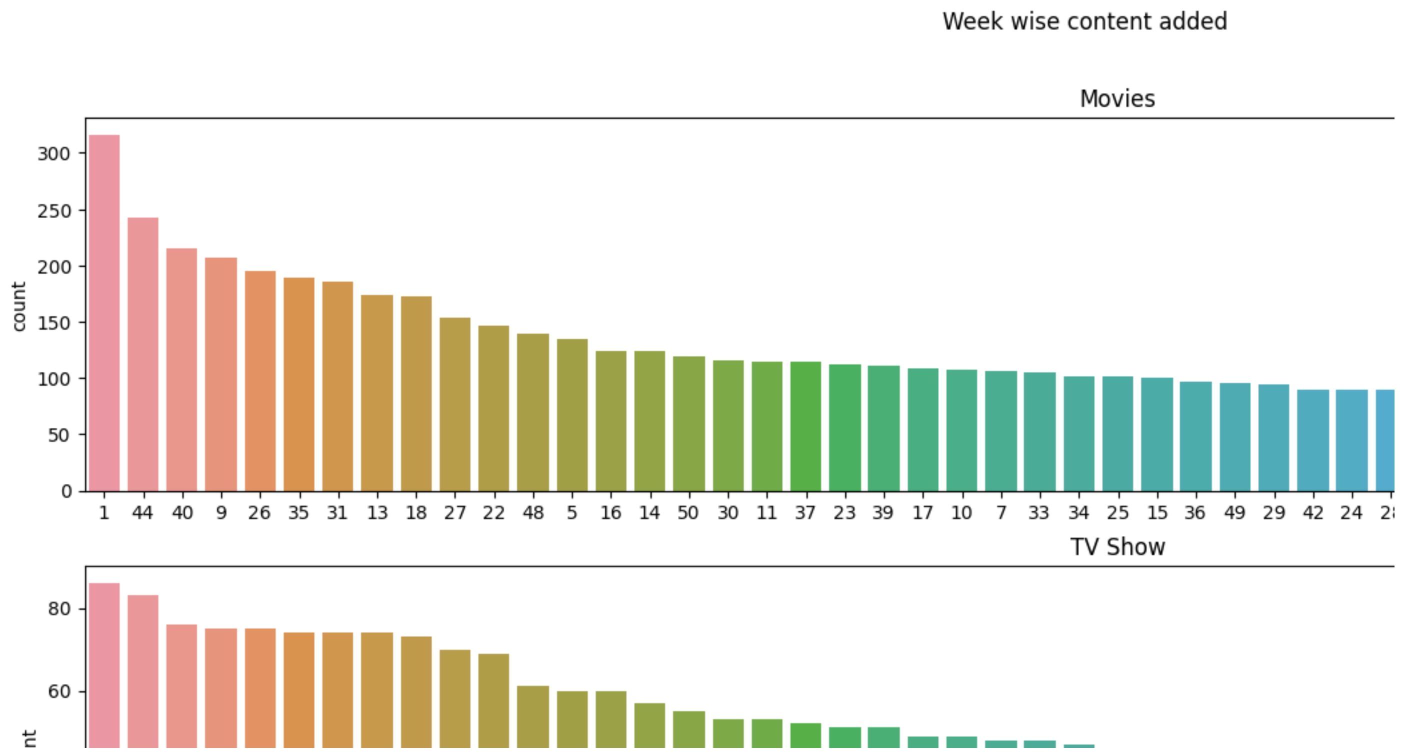
```
plt.figure(figsize=(12,8))
sns.countplot(data=netflix,x='month_added',hue='type')
```

```
<Axes: xlabel='month_added', ylabel='count'>
```



```
plt.figure(figsize=(20,8))
plt.subplot(2,1,1)
plt.suptitle("Week wise content added")
sns.countplot(x='week_added',data=movie_df,order=movie_df['week_added'].value_counts().index)
plt.xlabel('',loc='left')
plt.title('Movies')
plt.subplot(2,1,2)
sns.countplot(x='week_added',data=series_df,order=series_df['week_added'].value_counts().index)
plt.title('TV Show')
```

```
Text(0.5, 1.0, 'TV Show')
```

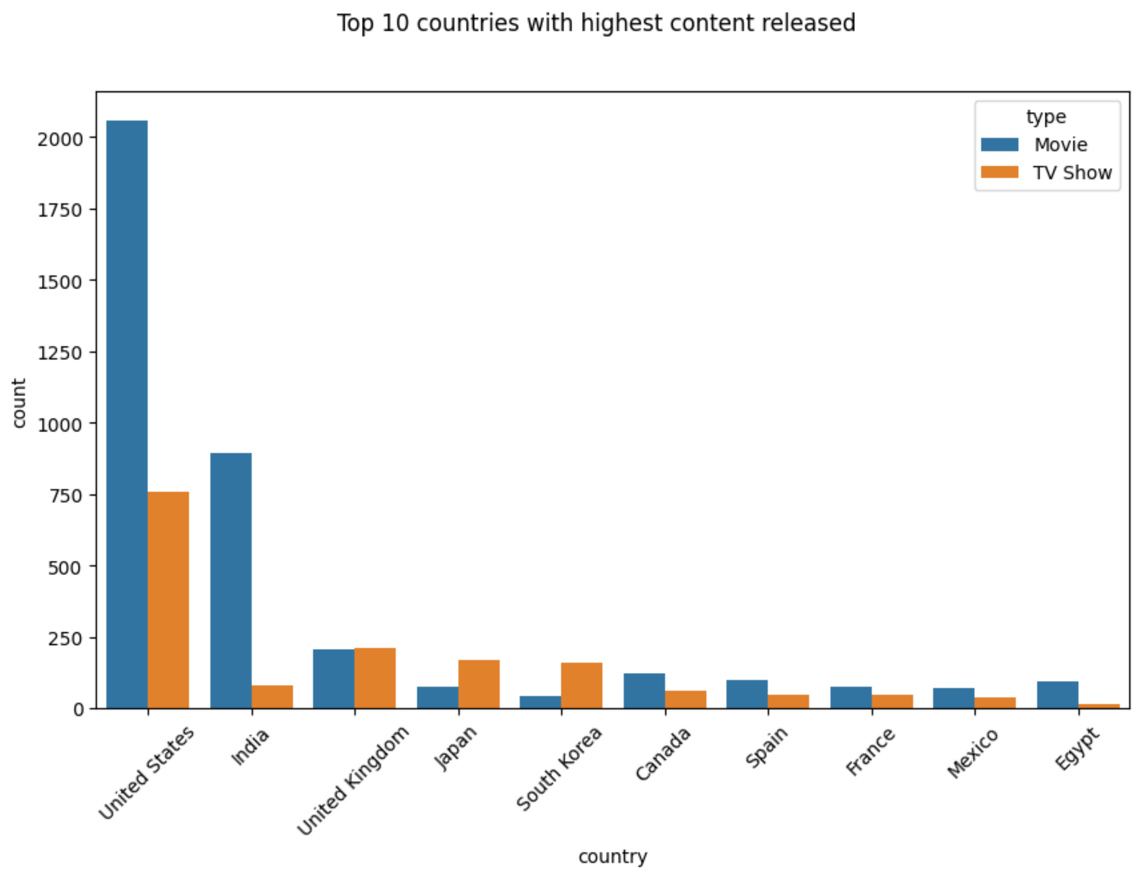


```
# Top 10 countries with highest content released


top_10_countries=netflix.loc[~netflix['country'].isnull()]


plt.figure(figsize=(10,6))
sns.countplot(data=top_10_countries,x='country',order=top_10_countries['country'].value_counts().index[:10],hue='type')
plt.suptitle('Top 10 countries with highest content released')
plt.xticks(rotation=45)
```

```
    (array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]),
     [Text(0, 0, 'United States'),
      Text(1, 0, 'India'),
      Text(2, 0, 'United Kingdom'),
      Text(3, 0, 'Japan'),
      Text(4, 0, 'South Korea'),
      Text(5, 0, 'Canada'),
      Text(6, 0, 'Spain'),
      Text(7, 0, 'France'),
      Text(8, 0, 'Mexico'),
      Text(9, 0, 'Egypt')])
```



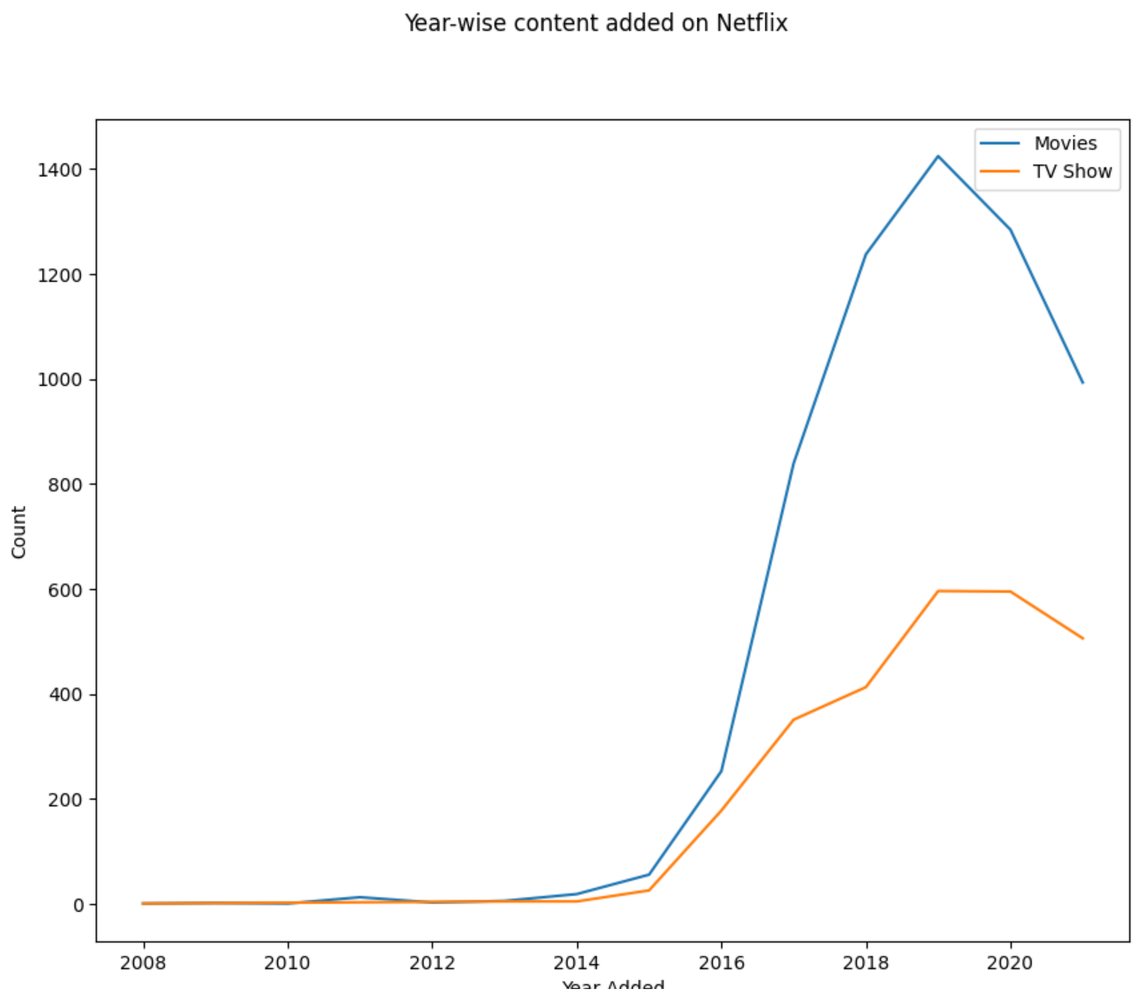
```
# Year wise content added


movie_yoy=netflix.loc[netflix['type']=='Movie'].groupby('year_added')['show_id'].count().reset_index()
series_yoy=netflix.loc[netflix['type']=='TV Show'].groupby('year_added')['show_id'].count().reset_index()
movie_yoy.rename({'show_id':'count'},axis=1,inplace=True)
series_yoy.rename({'show_id':'count'},axis=1,inplace=True)


movie_yoy['year_added']=movie_yoy['year_added'].astype('int64')
series_yoy['year_added']=series_yoy['year_added'].astype('int64')


plt.figure(figsize=(10,8))
sns.lineplot(data=movie_yoy,x='year_added',y='count',label='Movies')
sns.lineplot(data=series_yoy,x='year_added',y='count',label='TV Show')
plt.suptitle('Year-wise content added on Netflix')
plt.xlabel('Year Added')
plt.ylabel('Count')
```

```
Text(0, 0.5, 'Count')
```



```
# Released year of contents

movie_rel=netflix.loc[netflix['type']=='Movie'].groupby('release_year')['show_id'].count().reset_index()
series_rel=netflix.loc[netflix['type']=='TV Show'].groupby('release_year')['show_id'].count().reset_index()
movie_rel.rename({'show_id':'count'},axis=1,inplace=True)
series_rel.rename({'show_id':'count'},axis=1,inplace=True)


plt.figure(figsize=(10,8))
sns.lineplot(data=movie_rel,x='release_year',y='count',label='Movies')
sns.lineplot(data=series_rel,x='release_year',y='count',label='TV Show')
plt.suptitle('Release Timeline of the contents')
plt.xlabel('Release Year')
plt.ylabel('Count')
```

```
Text(0, 0.5, 'Count')
```

Release Timeline of the contents
800
700
600
500
Movies
TV Show

```
#Ideal duration of a Movie or TV show

#Preprocessing

movie_df=netflix.loc[netflix['type']=='Movie']
series_df=netflix.loc[netflix['type']=='TV Show']

movie_df['duration']=movie_df['duration'].astype('int64')
```

```
<ipython-input-113-0fb6e40dcc7a>:1: 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_guide/indexing.html#returning-a-view-versus-a-cc
  movie_df['duration']=movie_df['duration'].astype('int64')
```

```
movie_df['hrs']=pd.cut(movie_df['duration'],bins=[0,60,90,120,150,180,320],labels=['<1hr','1-1.5 hr','1.5-2 hr','2-2.5 hr','2.5-3 hr','> 3hr'
```

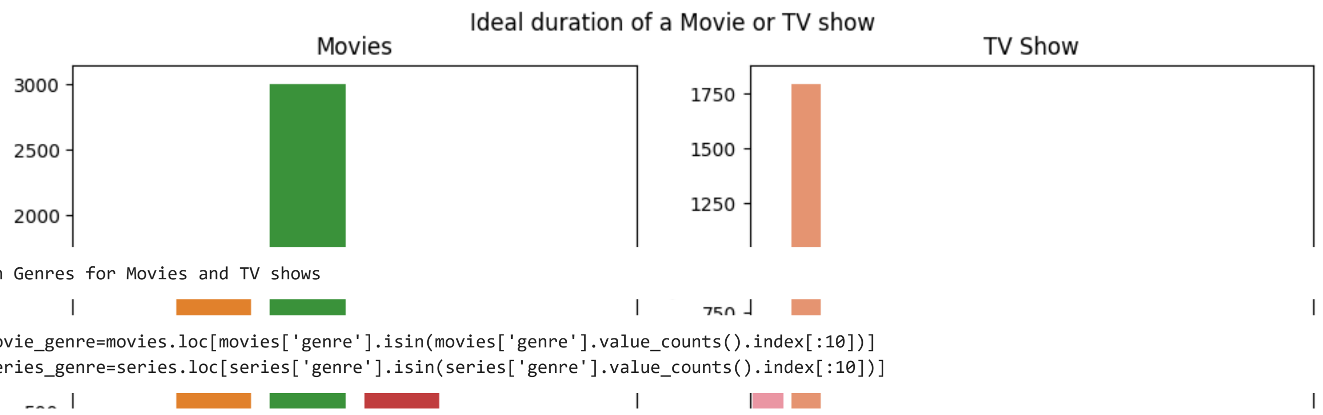
```
<ipython-input-114-6bb08f331285>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
  movie_df['hrs']=pd.cut(movie_df['duration'],bins=[0,60,90,120,150,180,320],labels=['<1hr','1-1.5 hr','1.5-2 hr','2-2.5 hr','2.5-3 hr',
```

```
#Ideal duration of a Movie or TV show
#Plot


plt.figure(figsize=(12,4))
plt.subplot(1,2,1)
plt.suptitle('Ideal duration of a Movie or TV show')
plt.title('Movies')
sns.countplot(data=movie_df,x='hrs')
plt.subplot(1,2,2)
plt.title('TV Show')
sns.countplot(data=series_df,x='duration')
plt.xlabel('Seasons')
```

```
Text(0.5, 0, 'Seasons')
```



```
#Top Ten Genres for Movies and TV shows
```

```
top10_movie_genre=movies.loc[movies['genre'].isin(movies['genre'].value_counts().index[:10])]
top10_series_genre=series.loc[series['genre'].isin(series['genre'].value_counts().index[:10])]
```

```
top10_movie_genre['duration']=top10_movie_genre['duration'].astype('int64')
top10_series_genre['duration']=top10_series_genre['duration'].astype('int64')
```

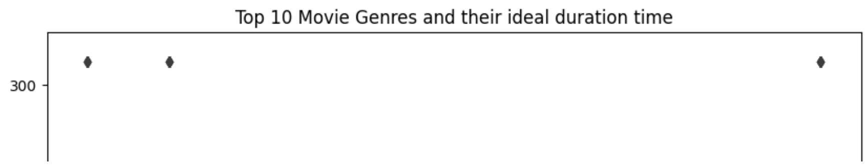
```
<ipython-input-93-e2642d357a2b>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
  top10_movie_genre['duration']=top10_movie_genre['duration'].astype('int64')
<ipython-input-93-e2642d357a2b>:2: 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_guide/indexing.html#returning-a-view-versus-a-cc
  top10_series_genre['duration']=top10_series_genre['duration'].astype('int64')
```

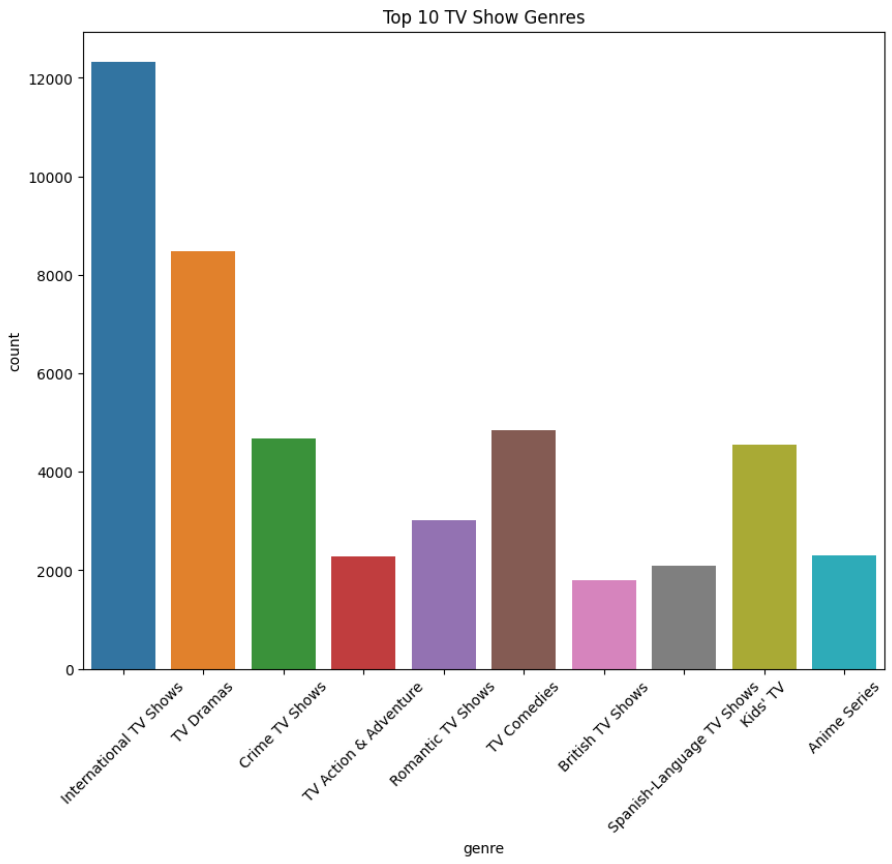
```
plt.figure(figsize=(10,8))
plt.title("Top 10 Movie Genres and their ideal duration time")
sns.boxplot(top10_movie_genre,x='genre',y='duration',order=top10_movie_genre['genre'].value_counts().index[:10])
plt.xticks(rotation=45)
```

```
(array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]),
 [Text(0, 0, 'Dramas'),
  Text(1, 0, 'International Movies'),
  Text(2, 0, 'Comedies'),
  Text(3, 0, 'Action & Adventure'),
  Text(4, 0, 'Independent Movies'),
  Text(5, 0, 'Children & Family Movies'),
  Text(6, 0, 'Thrillers'),
  Text(7, 0, 'Romantic Movies'),
  Text(8, 0, 'Horror Movies'),
  Text(9, 0, 'Sci-Fi & Fantasy')])
```



```
plt.figure(figsize=(10,8))
sns.countplot(data=top10_series_genre,x='genre')
plt.title('Top 10 TV Show Genres')
plt.xticks(rotation=45)
```

```
(array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]),
 [Text(0, 0, 'International TV Shows'),
  Text(1, 0, 'TV Dramas'),
  Text(2, 0, 'Crime TV Shows'),
  Text(3, 0, 'TV Action & Adventure'),
  Text(4, 0, 'Romantic TV Shows'),
  Text(5, 0, 'TV Comedies'),
  Text(6, 0, 'British TV Shows'),
  Text(7, 0, 'Spanish-Language TV Shows'),
  Text(8, 0, "Kids' TV"),
  Text(9, 0, 'Anime Series')])
```

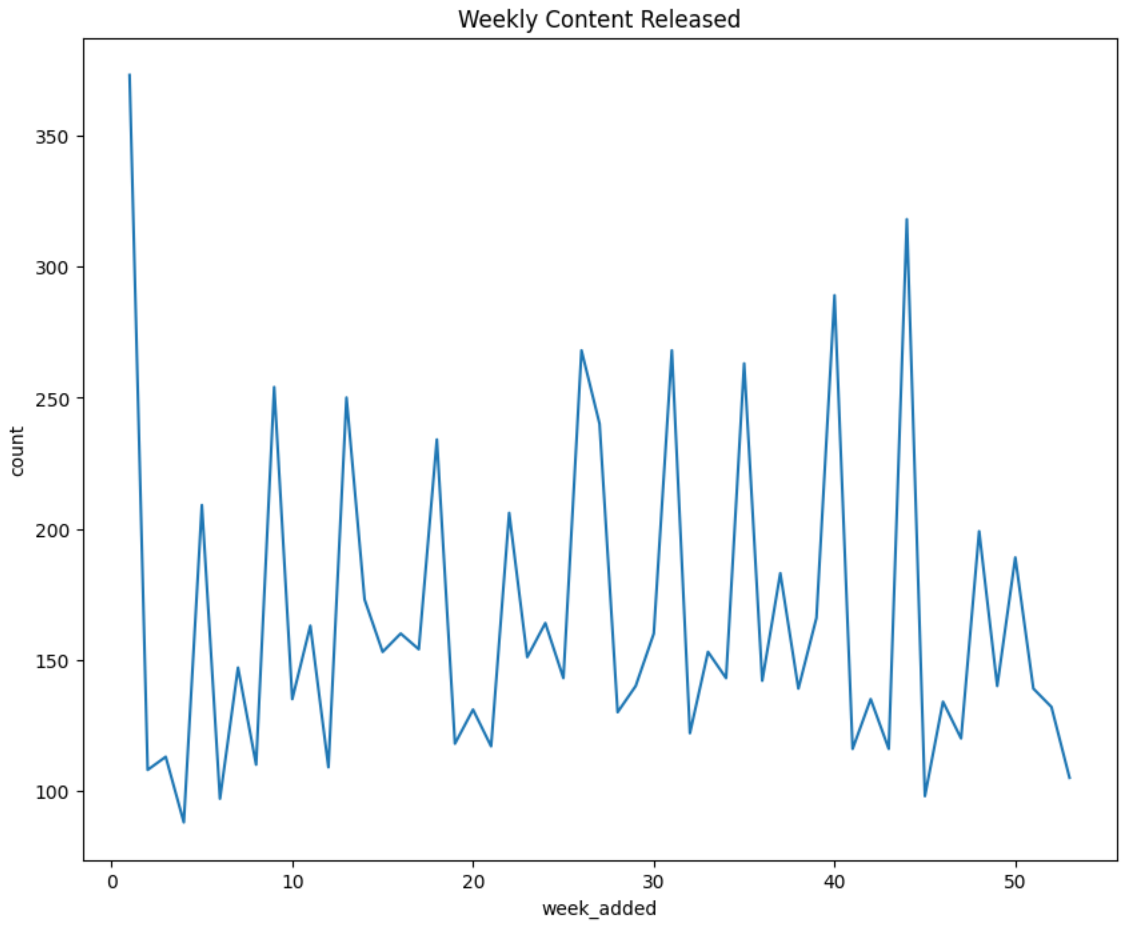


```
# Content released on each week


weekly_release=netflix.groupby(['week_added'])['show_id'].count().reset_index()
weekly_release.rename({'show_id':'count'},axis=1,inplace=True)


plt.figure(figsize=(10,8))
sns.lin(data=weekly_release,x='week_added',y='count')
plt.title('Weekly Content Released')
```

```
    Text(0.5, 1.0, 'Weekly Content Released')
```



```
#


netflix.head()
```

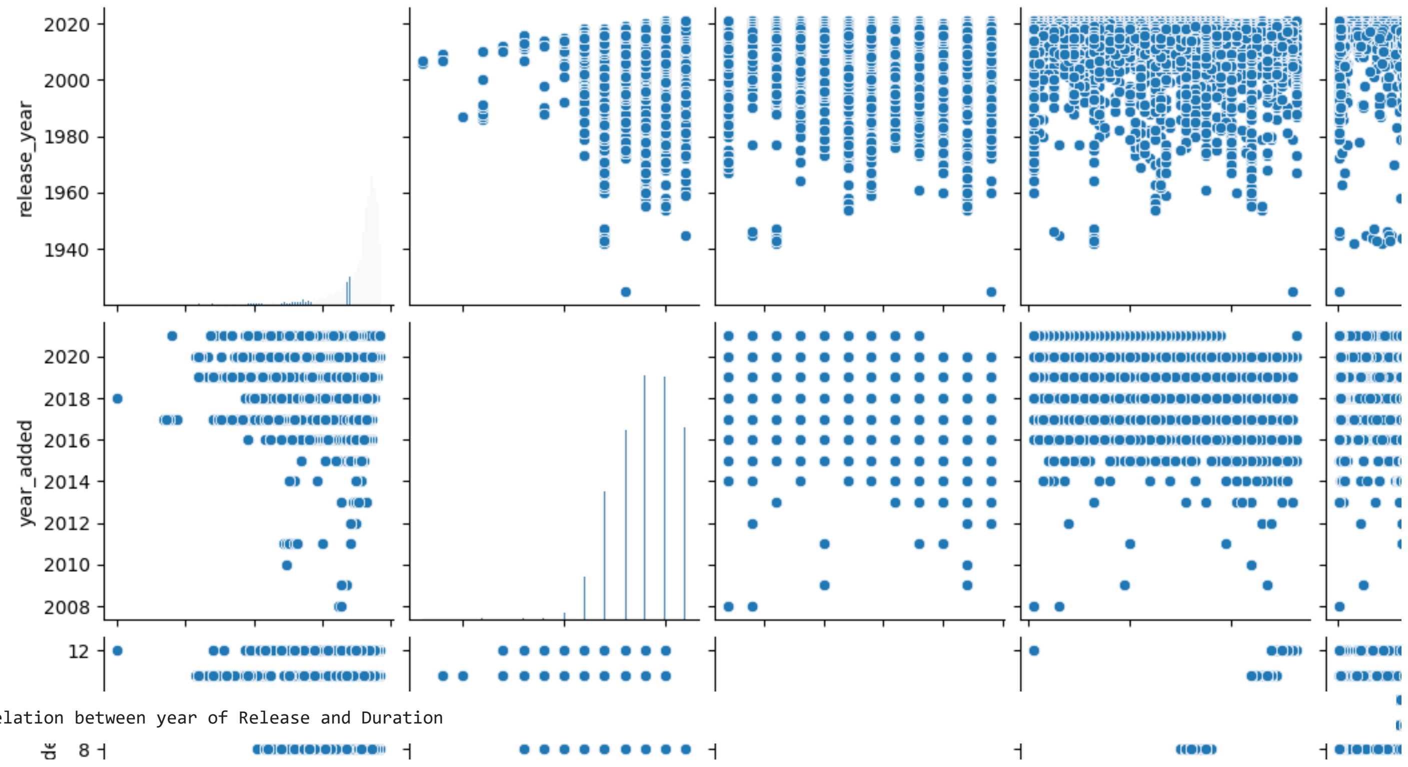
	show_id	type	title	director	cast	country	date_added	release_year	rating	
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	2021-09-25	2020	PG-13	
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	2021-09-24	2021	TV-MA	Internatior Dram
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	2021-09-24	2021	TV-MA	Crime TV Shows
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	NaN	2021-09-24	2021	TV-MA	Docus

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 186399 entries, 0 to 186398
Data columns (total 18 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   show_id           186399 non-null  object
 1   type              186399 non-null  object
 2   title             186399 non-null  object
 3   director          186399 non-null  object
 4   cast_combined     186399 non-null  object
 5   country_combined  186399 non-null  object
 6   date_added        186399 non-null  datetime64[ns]
 7   release_year      186399 non-null  int64
 8   rating            186399 non-null  object
 9   listed_in         186399 non-null  object
 10  description       186399 non-null  object
 11  cast              186399 non-null  object
 12  genre             186399 non-null  object
 13  country           186399 non-null  object
 14  year_added        186399 non-null  int64
 15  month_added       186399 non-null  int64
 16  week_added        186399 non-null  int64
 17  duration          186399 non-null  object
dtypes: datetime64[ns](1), int64(4), object(13)
memory usage: 27.0+ MB
```

```
netflix_test['duration']=netflix_test['duration'].astype('int64')
```

```
sns.pairplot(netflix_test)
```

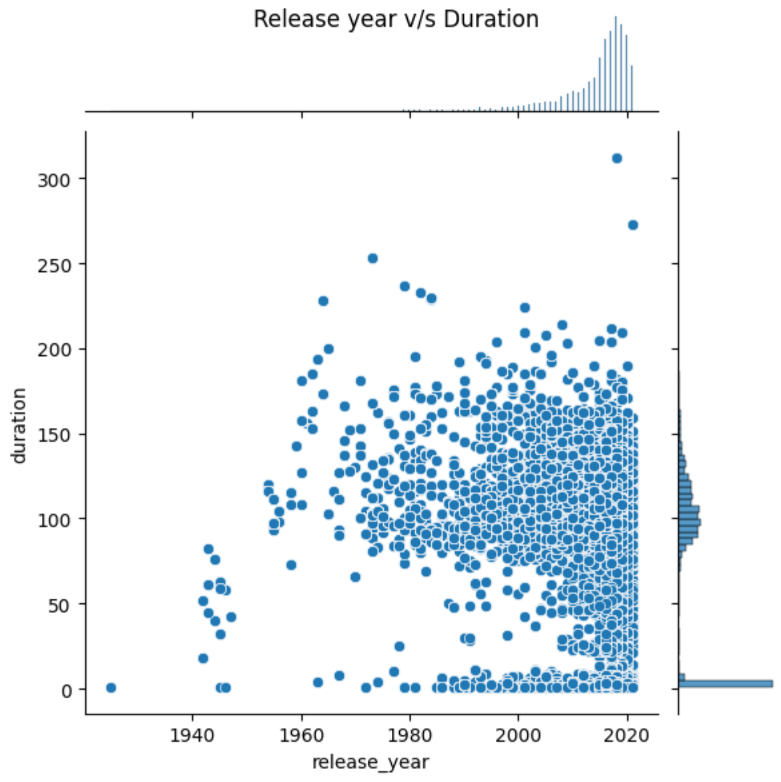
```
<seaborn.axisgrid.PairGrid at 0x7f569e282dd0>
```



```
#Corelation between year of Release and Duration
```

```
sns.jointplot(data=netflix_test,x='release_year',y='duration')
plt.suptitle('Release year v/s Duration')
```

```
Text(0.5, 0.98, 'Release year v/s Duration')
```



```
# Top 5 TV show genres among top 5 countries
```

```
top5_tv_countries=series_df['country'].value_counts().index[:5]
top5_tv_genre=series['genre'].value_counts().index[:5]
```

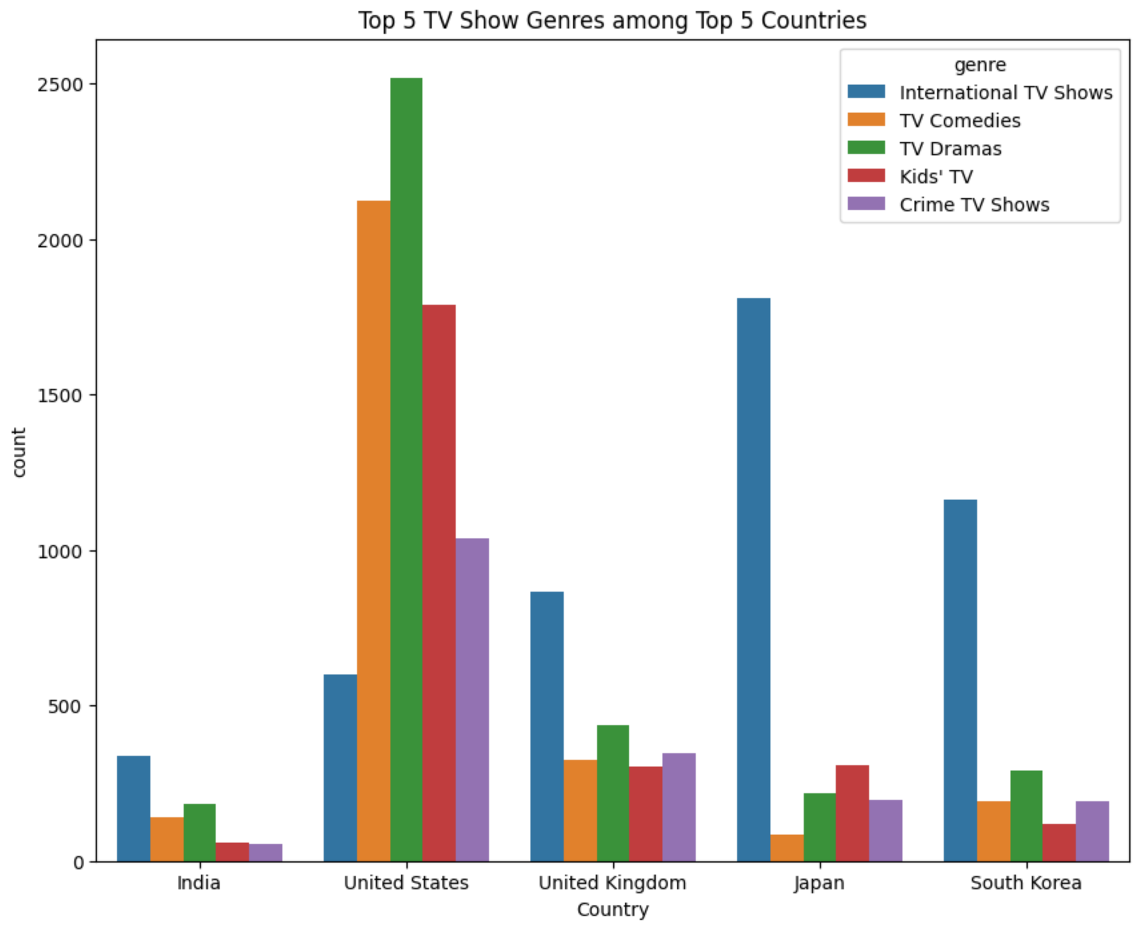
```
top5_tv_df=series.loc[(series['country'].isin(top5_tv_countries))&(series['genre'].isin(top5_tv_genre))]
```

```
top5_tv_df['genre'].value_counts()
```

```
International TV Shows    4775
TV Dramas                 3647
TV Comedies               2865
Kids' TV                  2578
Crime TV Shows            1827
Name: genre, dtype: int64
```

```
plt.figure(figsize=(10,8))
sns.countplot(data=top5_tv_df,x='country',hue='genre')
plt.title('Top 5 TV Show Genres among Top 5 Countries')
plt.xlabel('Country')
```

```
Text(0.5, 0, 'Country')
```



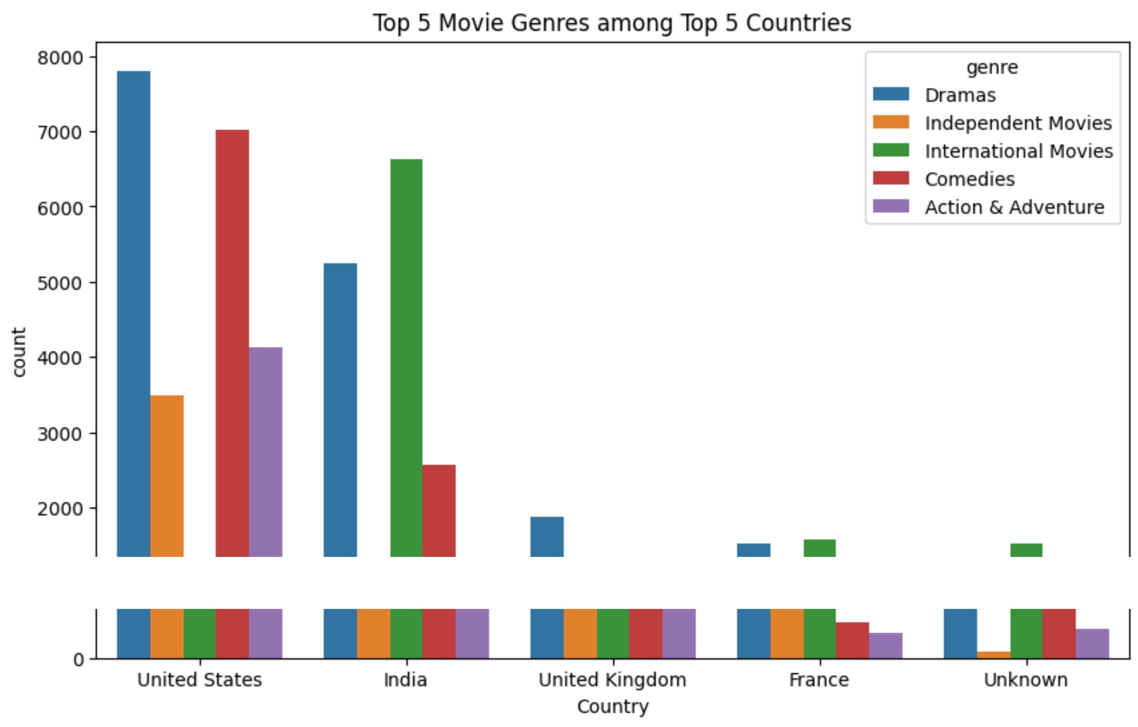
```
# Top 5 movie genres among top 5 countries
```

```
top5_tv_countries=movies['country'].value_counts().index[:5]
top5_tv_genre=movies['genre'].value_counts().index[:5]
```

```
top5_movie_df=movies.loc[(movies['country'].isin(top5_tv_countries))&(movies['genre'].isin(top5_tv_genre))]
```

```
plt.figure(figsize=(10,6))
sns.countplot(data=top5_movie_df,x='country',hue='genre',order=top5_tv_countries)
plt.title('Top 5 Movie Genres among Top 5 Countries')
plt.xlabel('Country')
```

```
Text(0.5, 0, 'Country')
```


Top 5 Movie Genres among Top 5 Countries
genre
Dramas
Independent Movies
International Movies
Comedies
Action & Adventure
count
United States
India
United Kingdom
France
Unknown
Country