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

```
#import input file :

from google.colab import drive
drive.mount('/content/drive')
```

```
    Mounted at /content/drive
```

```
df1 = pd.read_csv('/content/drive/My Drive/netflix.csv')
```

```
df1.head()
```

	show_id	type	title	director	cast	country	date_added	release_ye
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	20
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	20
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi	NaN	September 24, 2021	20

```
df1.shape
```

```
    (8809, 12)
```

```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8809 entries, 0 to 8808
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   show_id       8809 non-null   object
 1   type          8808 non-null   object
 2   title         8807 non-null   object
 3   director      6173 non-null   object
 4   cast          7983 non-null   object
 5   country       7976 non-null   object
 6   date_added    8797 non-null   object
 7   release_year  8807 non-null   object
 8   rating        8803 non-null   object
 9   duration      8804 non-null   object
 10  listed_in     8806 non-null   object
 11  description   8806 non-null   object
dtypes: object(12)
memory usage: 826.0+ KB
```

```
#Dropping description column
df1.drop('description', axis = 1, inplace=True)
```

```
for cl in df1.columns:
  val = round(((df1[cl].isna().sum())/df1.shape[0])*100,2)
  print( "{} column has {} % null values.".format(cl, val))
```

```
show_id column has 0.0 % null values.
type column has 0.01 % null values.
title column has 0.02 % null values.
director column has 29.92 % null values.
cast column has 9.38 % null values.
country column has 9.46 % null values.
date_added column has 0.14 % null values.
release_year column has 0.02 % null values.
rating column has 0.07 % null values.
duration column has 0.06 % null values.
listed_in column has 0.03 % null values.
```

Preprocessing Date

```
df1[df1['date added'].isna()]
```

	show_id	type	title	director	cast	country	date_added	releas
6066	s6067	TV Show	A Young Doctor's Notebook and Other Stories	NaN	Daniel Radcliffe, Jon Hamm, Adam Godley, Chris...	United Kingdom	NaN	
6174	s6175	TV Show	Anthony Bourdain: Parts Unknown	NaN	Anthony Bourdain	United States	NaN	
6795	s6796	TV Show	Frasier	NaN	Kelsey Grammer, Jane Leeves, David Hyde Pierce...	United States	NaN	
6806	s6807	TV Show	Friends	NaN	Jennifer Aniston, Courteney Cox, Lisa Kudrow, ...	United States	NaN	
6901	s6902	TV Show	Gunslinger Girl	NaN	Yuuka Nanri, Kanako Mitsuhashi, Eri Sendai, Am...	Japan	NaN	
7196	s7197	TV Show	Kikoriki	NaN	Igor Dmitriev	NaN	NaN	
7254	s7255	TV Show	La Familia P. Luche	NaN	Eugenio Derbez, Consuelo Duval, Luis Manuel √Å...	United States	NaN	
					Marc Maron,			

```
df1['release_year'] = df1['release_year'].fillna(df1['release_year'].mode()[0])
```

```
df1['date_added'].fillna('January 01, '+df1['release_year'], inplace=True)
```

```
df1[df1['date_added'].isna()]
```

show_id	type	title	director	cast	country	date_added	release_year	ratin

```
df1[['month', 'date', 'year']]= df1['date_added'].str.split(expand = True)
```

```
df1['month'].str.strip(), df1['date'].str.strip(), df1['year'].str.strip()
```

```
(0       September
 1       September
 2       September
 3       September
 4       September
           ...
 8804     November
 8805         July
 8806     November
 8807      January
 8808        March
 Name: month, Length: 8809, dtype: object,
 0       25,
 1       24,
 2       24,
 3       24,
 4       24,
        ...
 8804    20,
 8805     1,
 8806     1,
 8807    11,
 8808     2,
 Name: date, Length: 8809, dtype: object,
 0       2021
 1       2021
 2       2021
 3       2021
 4       2021
         ...
 8804    2019
 8805    2019
 8806    2019
 8807    2020
 8808    2019
 Name: year, Length: 8809, dtype: object)
```

```
df1['month'].value_counts()
```

```
    month
    July         827
    December     813
    September    770
    April        764
    October      760
    August       755
    January      750
    March        741
    June         728
    November     705
    May          632
    February     563
    TV-PG          1
    Name: count, dtype: int64
```

```
df1['date'].apply(lambda x: str(x)[:-1])
```

```
    0       25
    1       24
    2       24
    3       24
    4       24
            ..
    8804    20
    8805     1
    8806     1
    8807    11
    8808     2
    Name: date, Length: 8809, dtype: object
```

```
mapping = {'January':"01",'February': "02", 'March':"03",'April':"04", 'May':"05"
           'October':"10",'November':"11",'December':"12", "TV-PG ":"01"}
```

```
df1['date_new'] = df1['month']+"-"+df1['date']+"-"+df1['year']
df1['date_new'] = pd.to_datetime(df1['date_new'] )
df1.drop(['month','date', 'year'], inplace = True, axis = 1)
```

```
df1.isna().sum()
```

```
show_id            0
type               1
title              2
director        2636
cast             826
country          833
date_added         0
release_year       0
rating             6
duration           5
listed_in          3
date_new           1
dtype: int64
```

```
df1 = df1.loc[~df1['date_new'].isnull()]
df1.isna().sum()
```

```
show_id            0
type               1
title              1
director        2636
cast             826
country          833
date_added         0
release_year       0
rating             6
duration           5
listed_in          2
date_new           0
dtype: int64
```

```
df1['month_name'] = df1['date_new'].dt.month_name()
df1['year'] = df1['date_new'].dt.year
```

```
<ipython-input-21-f52d0e0bd40b>: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/st
  df1['month_name'] = df1['date_new'].dt.month_name()
<ipython-input-21-f52d0e0bd40b>: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/st
  df1['year'] = df1['date_new'].dt.year
```

```
df1.drop('date_added', axis = 1, inplace=True)
```

```
<ipython-input-22-14bf5c0aab3c>: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/pandas-docs/st
  df1.drop('date_added', axis = 1, inplace=True)
```

```
#Dropping rows with nan values for 'rating' and 'duration'

df1 = df1.loc[~df1['rating'].isnull()]
df1= df1.loc[~df1['duration'].isnull()]
```

```
##Replacing nan values in columns director, cast, country with NA
df1.fillna("NA", inplace=True)
df1.isna().sum()
```

```
show_id         0
type            0
title           0
director        0
cast            0
country         0
release_year    0
rating          0
duration        0
listed_in       0
date_new        0
month_name      0
year            0
dtype: int64
```

```
df1.shape
```

```
(8799, 13)
```

```
df1.head()
```

	show_id	type	title	director	cast	country	release_year	rating
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NA	United States	2020	PG-13
1	s2	TV Show	Blood & Water	NA	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	2021	TV-MA
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy	NA	2021	TV-MA

Checking for the nested columns:

```
df1.director.value_counts()
```

```
director
NA                                  2631
Rajiv Chilaka                         19
Ra√∫l Campos, Jan Suter               18
Marcus Raboy                          16
Suhas Kadav                           16
                                    ... 
Raymie Muzquiz, Stu Livingston         1
Joe Menendez                           1
Eric Bross                             1
Will Eisenberg                         1
Mozez Singh                            1
Name: count, Length: 4526, dtype: int64
```

```
df1.cast.value_counts()
```

```
cast
NA
824
David Attenborough
19
Vatsal Dubey, Julie Tejwani, Rupa Bhimani, Jigna Bhardwaj, Rajesh Kava,
Mousam, Swapnil
14
Samuel West
10
Jeff Dunham
7

...
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
Neeraj Kabi, Geetanjali Kulkarni, Danish Husain, Sheeba Chaddha, Paras
Priyadarshan, Anshul Chauhan, Anud Singh Dhaka, Shirin Sewani, Mihir Ahuja,
Vasundhara Rajput                            1
Sanjay Dutt, Arjun Kapoor, Kriti Sanon, Zeenat Aman, Mohnish Bahl, Padmini
Kolhapure, Kunal Kapoor, Suhasini Mulay
1
Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanana, Manish Chaudhary, Meghna
Malik, Malkeet Rauni, Anita Shabdish, Chittaranjan Tripathy
1
Name: count, Length: 7689, dtype: int64
```

```
df1.country.value_counts()
```

```
country
United States                             2814
India                                      972
NA                                         830
United Kingdom                             419
Japan                                      244
                                          ...
Romania, Bulgaria, Hungary                   1
Uruguay, Guatemala                           1
France, Senegal, Belgium                     1
Mexico, United States, Spain, Colombia       1
United Arab Emirates, Jordan                 1
Name: count, Length: 749, dtype: int64
```

```
df1.listed_in.value_counts()
```

```
listed_in
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: count, Length: 514, dtype: int64
```

Columns : cast, director, country, listed_in are nested clolumns, we need to unnest them and then form a new dataframe.

```
#Country

country_df = df1[["title", "country"]]
country_df["unnested_country"] = country_df ["country"].apply(lambda x: str(x).sp
country_df = country_df.explode("unnested_country")
country_df.head(10)
```

```
<ipython-input-31-b37fbfbf412f>: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/st
  country_df["unnested_country"] = country_df ["country"].apply(lambda x: str(
```

	title	country	unnested_country
0	Dick Johnson Is Dead	United States	United States
1	Blood & Water	South Africa	South Africa
2	Ganglands	NA	NA
3	Jailbirds New Orleans	NA	NA
4	Kota Factory	India	India
5	Midnight Mass	NA	NA
6	My Little Pony: A New Generation	NA	NA
7	Sankofa	United States, Ghana, Burkina Faso, United Kin...	United States
7	Sankofa	United States, Ghana, Burkina Faso, United Kin...	Ghana

```
#Cast
cast_df = df1[["title", "cast"]]
cast_df["unnested_cast"] = cast_df["cast"].apply(lambda x: str(x).split(", "))
cast_df = cast_df.explode("unnested_cast")
cast_df.head(10)
```

```
<ipython-input-32-c79afd57513a>:3: 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/st
  cast_df["unnested_cast"] = cast_df["cast"].apply(lambda x: str(x).split(", "
```

	title	cast	unnested_cast
0	Dick Johnson Is Dead	NA	NA
1	Blood & Water	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	Ama Qamata
1	Blood & Water	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	Khosi Ngema
1	Blood & Water	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	Gail Mabalane
1	Blood & Water	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	Thabang Molaba
1	Blood & Water	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	Dillon Windvogel
1	Blood & Water	Ama Qamata, Khosi Ngema, Gail Mabalane,	Natasha

```
#Director
director_df = df1[["title", "director"]]
director_df["unnested_director"] = director_df["director"].apply(lambda x: str(x)
director_df = director_df.explode("unnested_director")
director_df.head(10)
```

```
<ipython-input-33-8726910fcc88>:3: 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/st
  director_df["unnested_director"] = director_df["director"].apply(lambda x: s
```

	title	director	unnested_director
0	Dick Johnson Is Dead	Kirsten Johnson	Kirsten Johnson
1	Blood & Water	NA	NA
2	Ganglands	Julien Leclercq	Julien Leclercq
3	Jailbirds New Orleans	NA	NA
4	Kota Factory	NA	NA
5	Midnight Mass	Mike Flanagan	Mike Flanagan
6	My Little Pony: A New Generation	Robert Cullen, Jos√© Luis Ucha	Robert Cullen
6	My Little Pony: A New Generation	Robert Cullen, Jos√© Luis Ucha	Jos√© Luis Ucha
7	Sankofa	Haile Gerima	Haile Gerima
8	The Great British Baking Show	Andy Devonshire	Andy Devonshire

```
#listed_in
listed_df = df1[["title", "listed_in"]]
listed_df["unnested_listed_in"] = listed_df["listed_in"].apply(lambda x: str(x).s
listed_df = listed_df.explode("unnested_listed_in")
listed_df.head(10)
```

```
<ipython-input-34-11e870ed3cc3>:3: 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/st
  listed_df["unnested_listed_in"] = listed_df["listed_in"].apply(lambda x: str
```

	title	listed_in	unnested_listed_in
0	Dick Johnson Is Dead	Documentaries	Documentaries
1	Blood & Water	International TV Shows, TV Dramas, TV Mysteries	International TV Shows
1	Blood & Water	International TV Shows, TV Dramas, TV Mysteries	TV Dramas
1	Blood & Water	International TV Shows, TV Dramas, TV Mysteries	TV Mysteries
2	Ganglands	Crime TV Shows, International TV Shows, TV Act...	Crime TV Shows
2	Ganglands	Crime TV Shows, International TV Shows, TV Act...	International TV Shows
2	Ganglands	Crime TV Shows, International TV Shows, TV Act	TV Action & Adventure

```
merge_df = pd.merge(
    left=cast_df,
    right=country_df,
    on="title"
)
merge_df.head(10)
```

	title	cast	unnested_cast	country	unnested_country
0	Dick Johnson Is Dead	NA	NA	United States	United States
1	Blood & Water	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	Ama Qamata	South Africa	South Africa
2	Blood & Water	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	Khosi Ngema	South Africa	South Africa
3	Blood & Water	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	Gail Mabalane	South Africa	South Africa
4	Blood & Water	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	Thabang Molaba	South Africa	South Africa

```
merge_df1 = pd.merge(
    left=merge_df,
    right=director_df,
    on="title"
)
```

```
merge_df2 = pd.merge(
    left=merge_df1,
    right=listed_df,
    on="title"
)
```

```
merge_df2.drop(['cast', 'country', 'director','listed_in'], axis = 1, inplace = T
merge_df2.head(10)
```

	title	unnested_cast	unnested_country	unnested_director	unnested_listed_
0	Dick Johnson Is Dead	NA	United States	Kirsten Johnson	Documenta
1	Blood & Water	Ama Qamata	South Africa	NA	International TV Sh
2	Blood & Water	Ama Qamata	South Africa	NA	TV Drar
3	Blood & Water	Ama Qamata	South Africa	NA	TV Myste
4	Blood & Water	Khosi Ngema	South Africa	NA	International TV Sh
5	Blood & Water	Khosi Ngema	South Africa	NA	TV Drar
	Blood &				

```
final_df = pd.merge(
    left=df1[['show_id','type', 'title', 'date_new','release_year','rating','dura
    right=merge_df2,
    on="title"
)
final_df.head()
```

	show_id	type	title	date_new	release_year	rating	duration	listed_ir
0	s1	Movie	Dick Johnson Is Dead	2021-09-25	2020	PG-13	90 min	Documentaries
1	s2	TV Show	Blood & Water	2021-09-24	2021	TV-MA	2 Seasons	Internationa TV Shows, TV Dramas, TV Mysteries
2	s2	TV Show	Blood & Water	2021-09-24	2021	TV-MA	2 Seasons	Internationa TV Shows, TV Dramas, TV Mysteries
3	s2	TV Show	Blood & Water	2021-09-24	2021	TV-MA	2 Seasons	Internationa TV Shows, TV Dramas, TV Mysteries
4	s2	TV Show	Blood & Water	2021-09-24	2021	TV-MA	2 Seasons	Internationa TV Shows, TV Dramas, TV Mysteries

ANALYSIS:

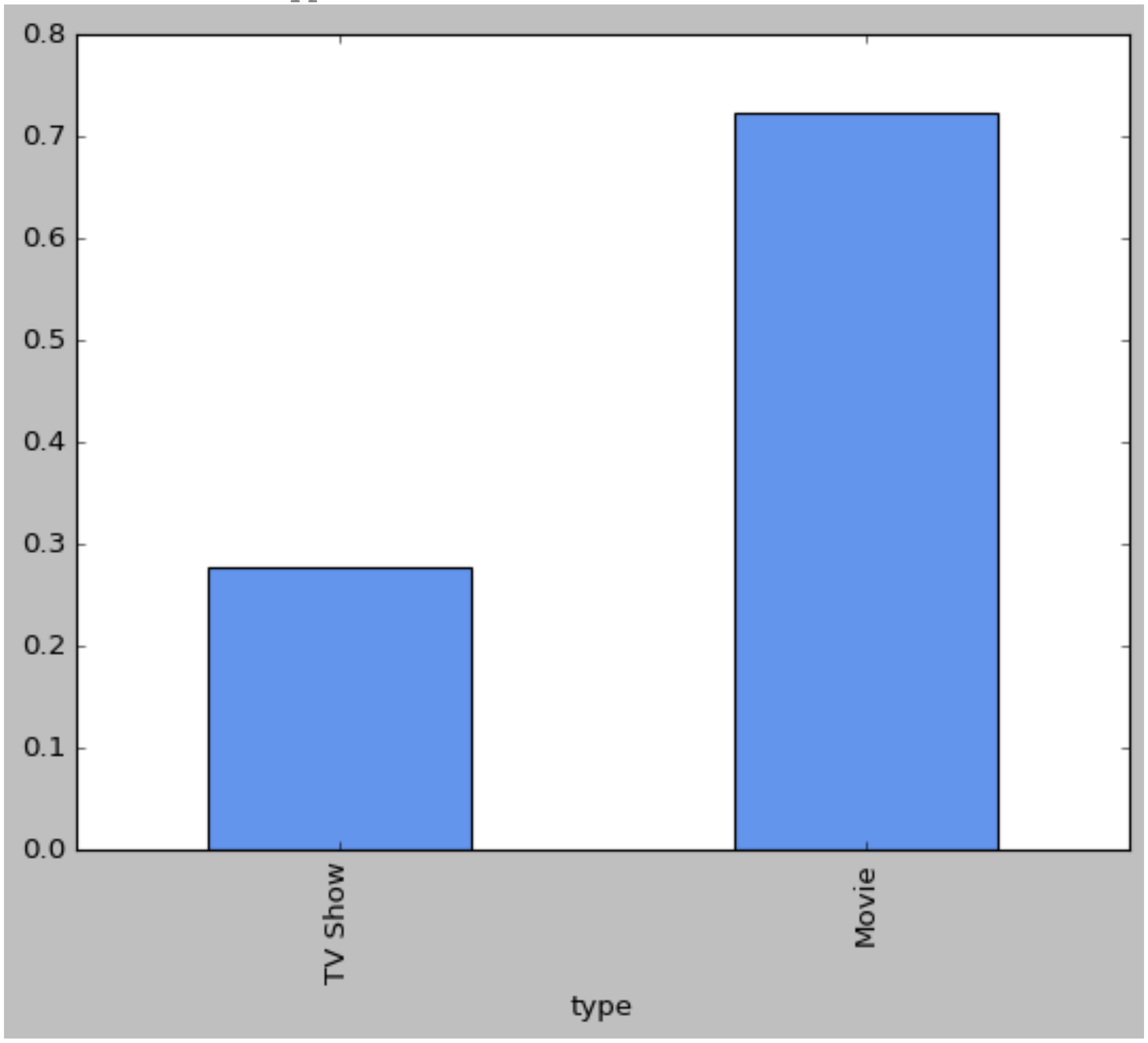
```
plt.style.use('classic')
import seaborn as sns
```

```
final_df.type.value_counts()
```

```
type
Movie      147809
TV Show     56690
Name: count, dtype: int64
```

```
final_df['type'].value_counts(normalize = True).sort_values(ascending=True).plot(
```

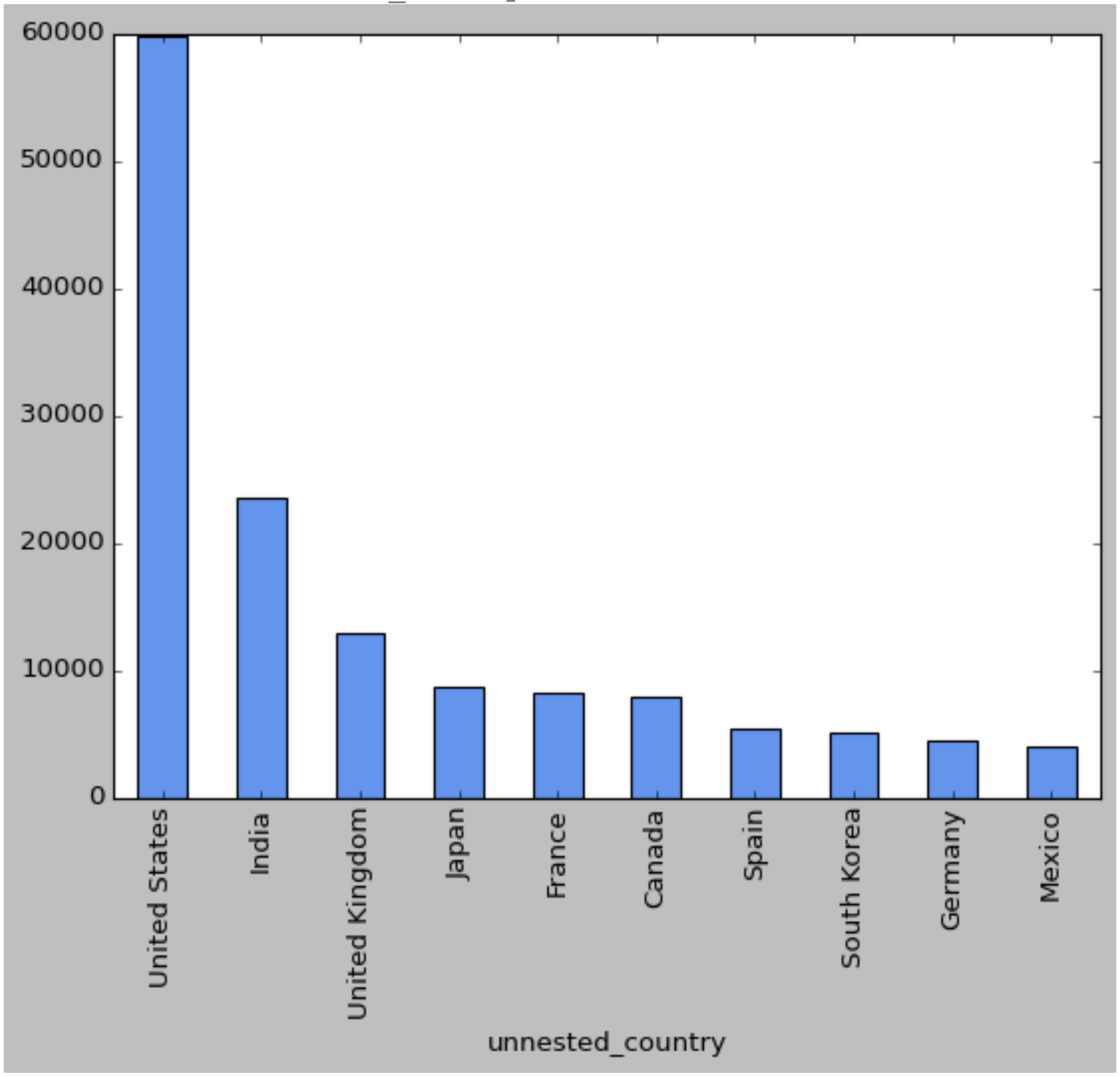
```
<Axes: xlabel='type'>
```



Top 10 countries by content production:

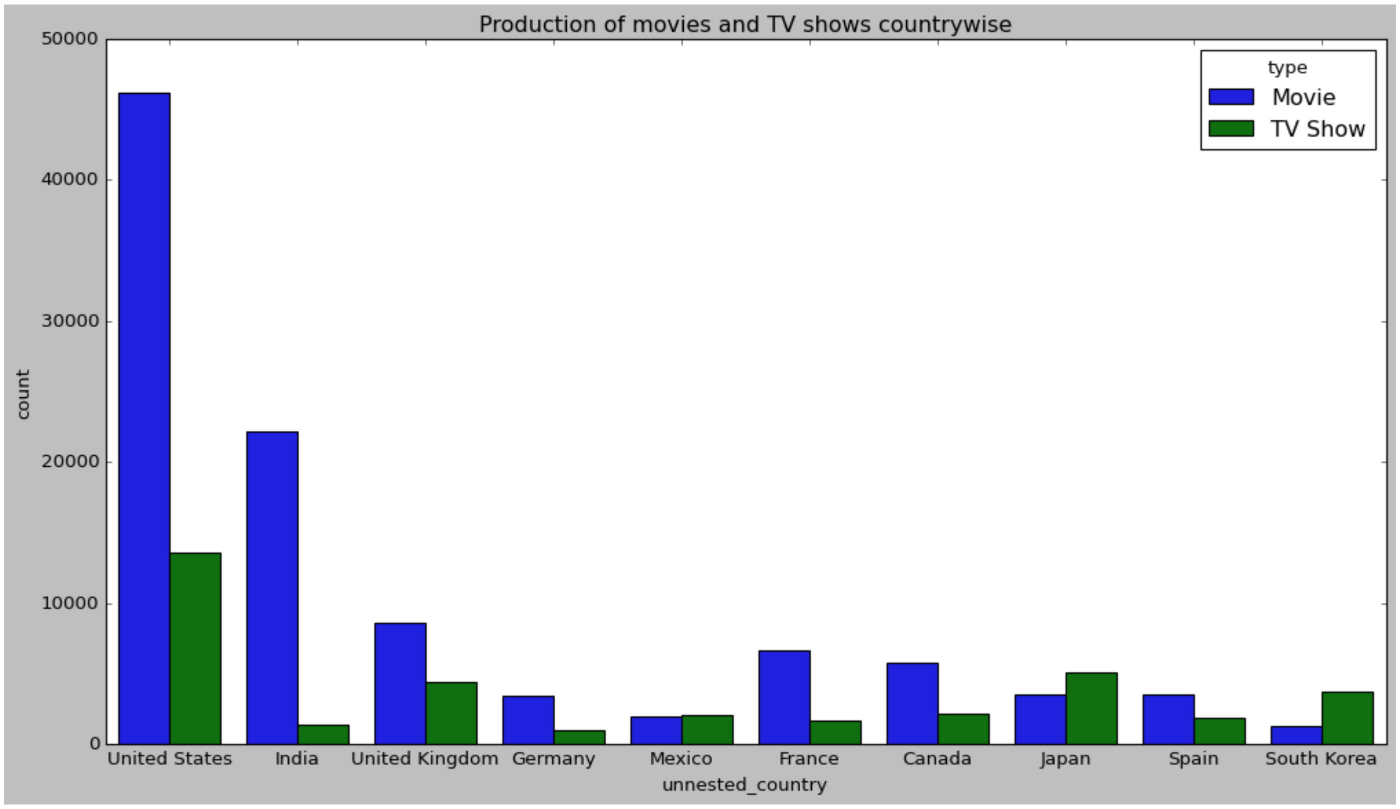
```
top10_country_df = final_df[final_df['unnested_country'] != "NA"]
top10_country_df["unnested_country"].value_counts().head(10).plot(kind = 'bar',co
```

```
<Axes: xlabel='unnested_country'>
```

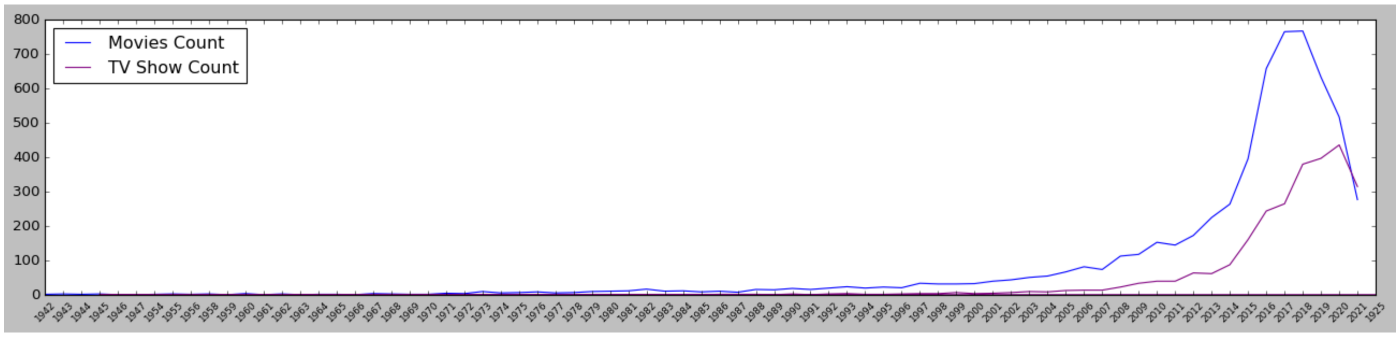


Production of content in the above countries:

```
top10 = top10_country_df["unnested_country"].value_counts().head(10).reset_index(
top10.rename(columns = {"index" :"unnested_country" , "country" : "count"}, inpla
top10_country_df = top10_country_df.merge(top10, how = "inner" , on = "unnested_c
plt.figure(figsize = (15,8))
sns.countplot(x ="unnested_country" , data =top10_country_df , hue = "type" )
plt.title("Production of movies and TV shows countrywise")
plt.show()
```



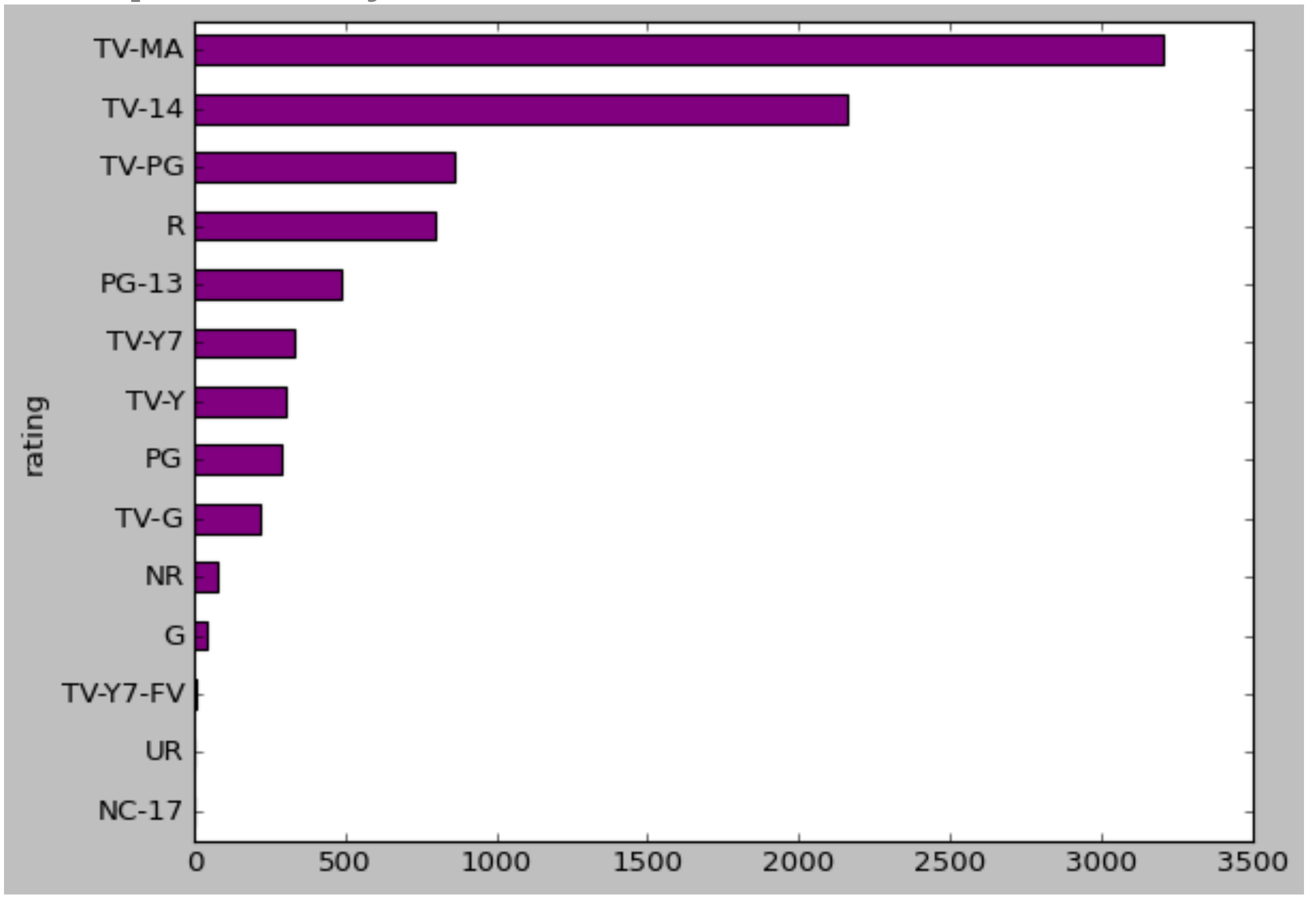
```
figure, axes = plt.subplots()
axes.plot(df1[df1['type']=='Movie'].groupby('release_year').show_id.count(), labe
axes.plot(df1[df1['type']=='TV Show'].groupby('release_year').show_id.count(),col
axes.legend(loc = 'upper left');
plt.xticks(rotation = 45, fontsize = 8)
figure.set_size_inches(20,4)
```



- In 2020, 2021 , maximum no. of TV shows are added which shows that TV shows are becoming very popular.

```
df1.rating.value_counts().sort_values(ascending=True).plot(kind = 'barh', color='
```

```
<Axes: ylabel='rating'>
```



Start coding or generate with AI.

```
df_cast = df1[['title','cast']]
result = df_cast['cast'].str.split(',', expand=True)
#result.head()
new_df = pd.concat([df1[['show_id','type','title']],result], axis=1)
cast_df = pd.melt(new_df,id_vars = ['show_id','type','title'])
#cast_df.head()
cast_df.groupby('value')['type'].count().sort_values(ascending = False)
```

```
value
NA                      824
 Anupam Kher             39
 Rupa Bhimani            31
 Takahiro Sakurai        30
 Julie Tejwani           28
                       ...
 Jo√£o Pessanha           1
 Jo√£o Pedro Zappa        1
 Jo√£o Lagarto            1
 Jo√£o F√°bio Cabral      1
≈û√ºkr√º √ñzyƒ±ldƒ±z      1
Name: type, Length: 39285, dtype: int64
```

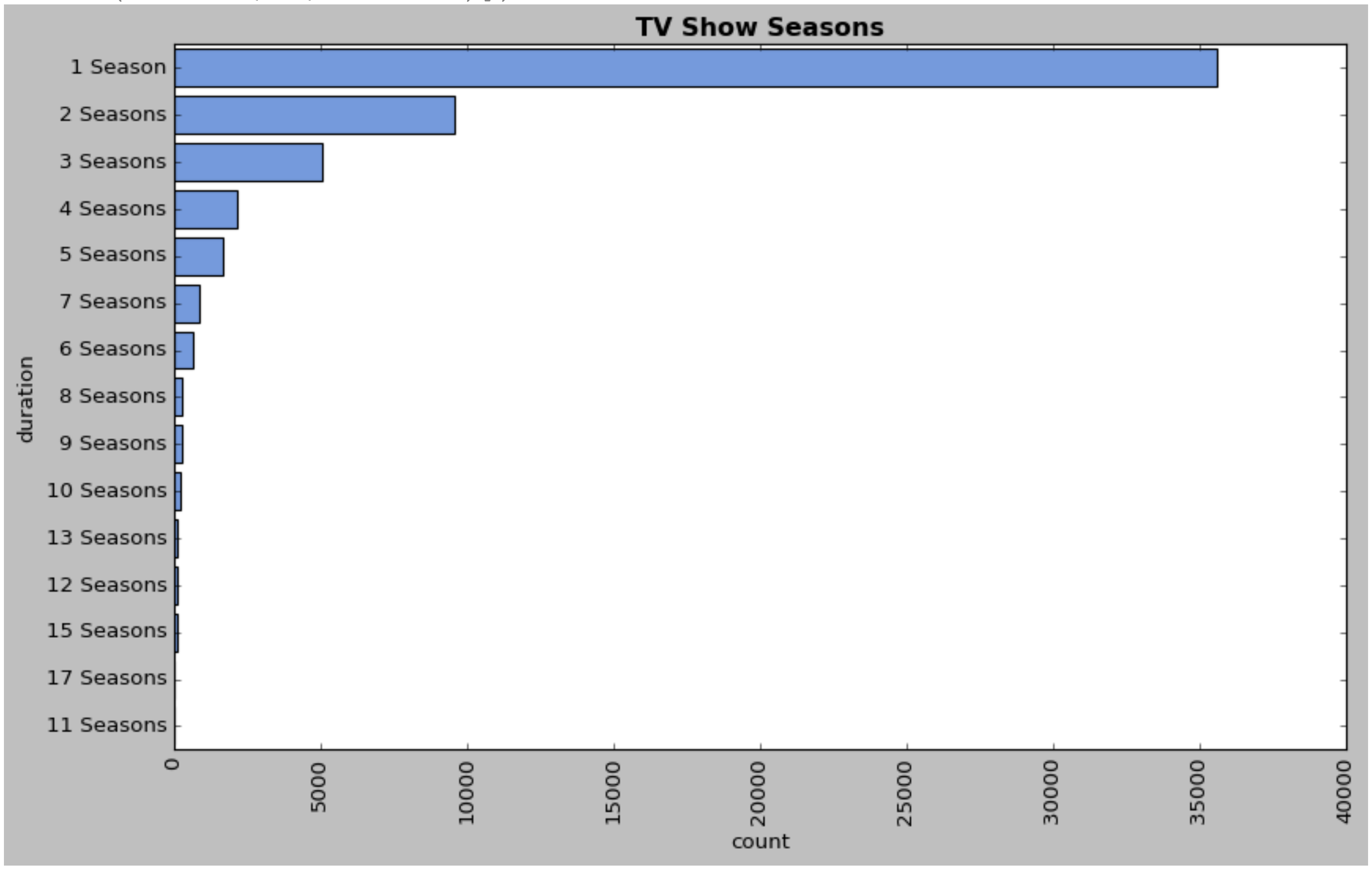
Famous actor: Aunpam Kher

- Production of movies and TV shows has started increasing from around year 2000. with total of 800 movies produced in year 2018.
- Number of movies produced are always more than the TV shows.

```
import seaborn as sns
```

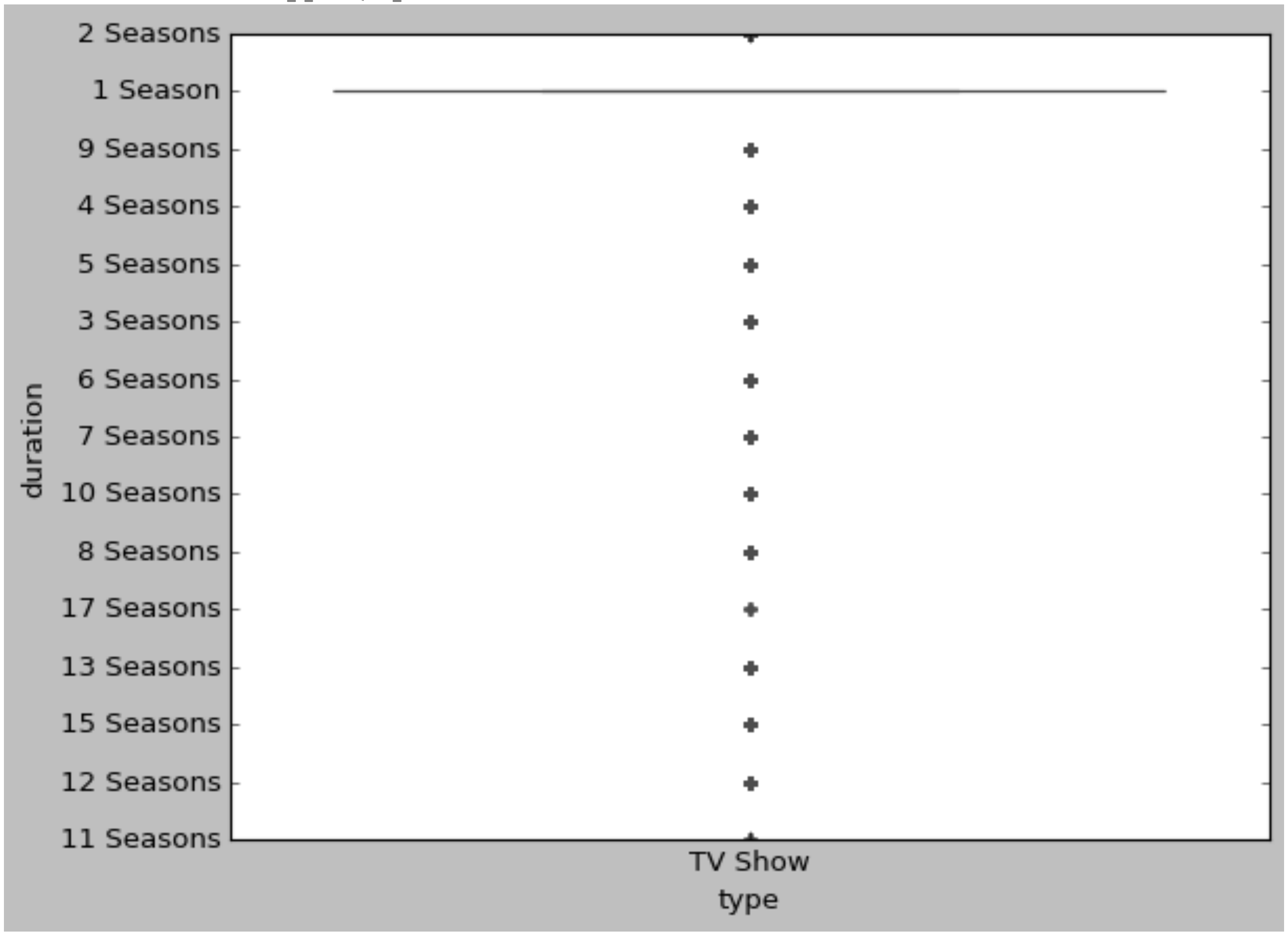
```
Tv_Shows = final_df.loc[final_df['type'] == 'TV Show']

plt.figure(figsize=(12,7))
ax = sns.countplot(Tv_Shows['duration'],order = Tv_Shows['duration'].value_counts
plt.title('TV Show Seasons',fontweight="bold")
plt.xticks(rotation=90)
```

```
(array([    0.,  5000., 10000., 15000., 20000., 25000., 30000., 35000.,
        40000.]),
 [Text(0.0, 0, '0'),
  Text(5000.0, 0, '5000'),
  Text(10000.0, 0, '10000'),
  Text(15000.0, 0, '15000'),
  Text(20000.0, 0, '20000'),
  Text(25000.0, 0, '25000'),
  Text(30000.0, 0, '30000'),
  Text(35000.0, 0, '35000'),
  Text(40000.0, 0, '40000')])
```



```
tv_show_df = final_df.loc[final_df['type'] == 'TV Show']
sns.boxplot(data=tv_show_df, x='type', y='duration')
```

```
<Axes: xlabel='type', ylabel='duration'>
```



TV-Shows with 1 season are more popular.

```
#Most famous actor countrywise:
final_df['unnested_country'].value_counts()
```

```
unnested_country
United States     59764
India             23534
United Kingdom    12945
NA                12495
Japan              8635
                  ...
Palestine             2
Kazakhstan            1
Nicaragua             1
United States,        1
Uganda                1
Name: count, Length: 128, dtype: int64
```

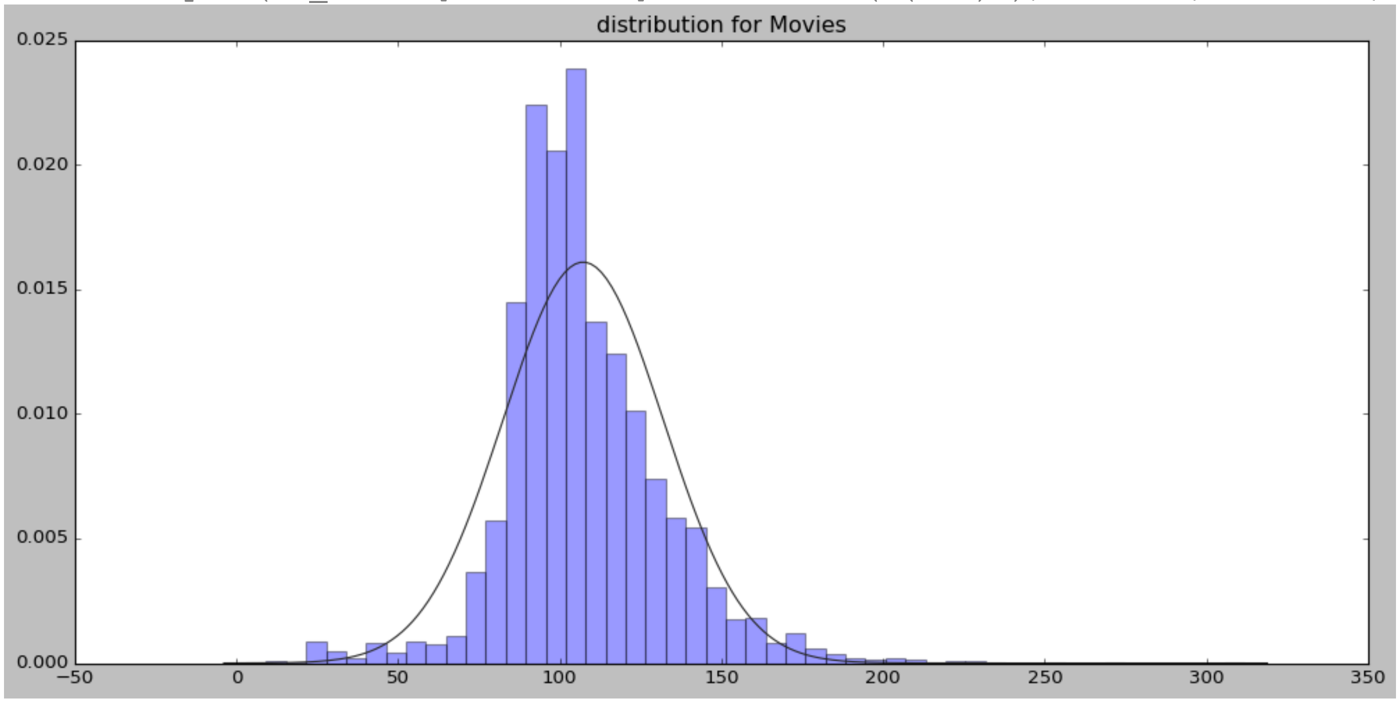
```
movies_df = final_df.loc[final_df['type'] == 'Movie']
```

```
movies_df['duration'].value_counts()
#sns.barplot(movies_df['duration'])#right skewed
```

```
duration
94 min     4343
106 min    4040
97 min     3624
95 min     3560
96 min     3484
           ...
20 min        4
5 min         3
9 min         2
8 min         2
11 min        2
Name: count, Length: 205, dtype: int64
```

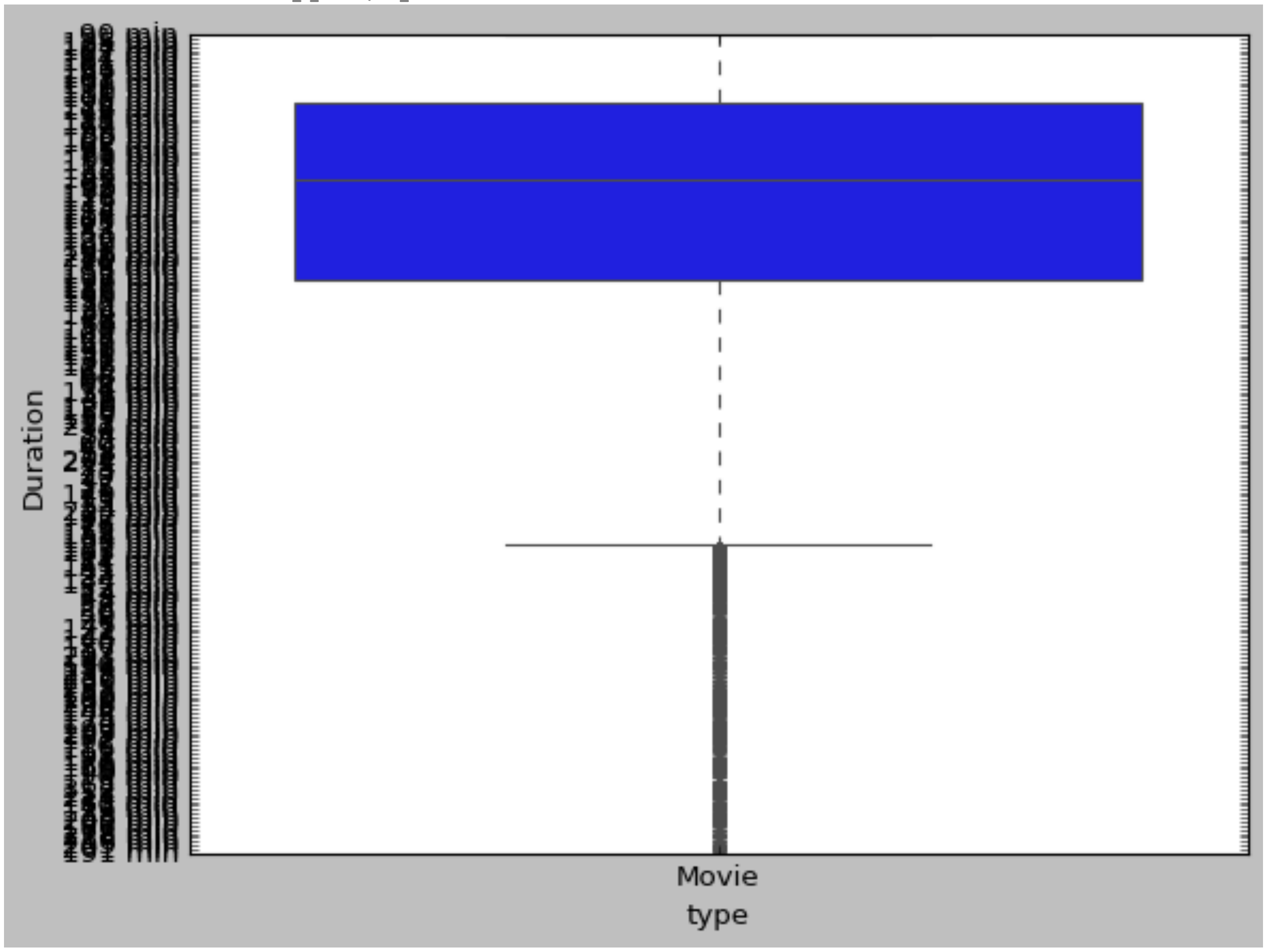
```
from scipy.stats import norm

df_movies = final_df[final_df['type'] == 'Movie']
plt.figure(figsize=(15,7))
sns.distplot(df_movies['duration'].str.extract('(\d+)'),fit=norm,kde=False,color=
plt.title('distribution for Movies')
plt.show()
```

```
<ipython-input-53-10d7290c33ec>:5: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df_movies['duration'].str.extract('(\d+)'),fit=norm,kde=False,c
```



```
movie_df = final_df.loc[final_df['type'] == 'Movie']
plt.ylabel('Duration')
sns.boxplot(data=movie_df, x='type', y='duration')
```

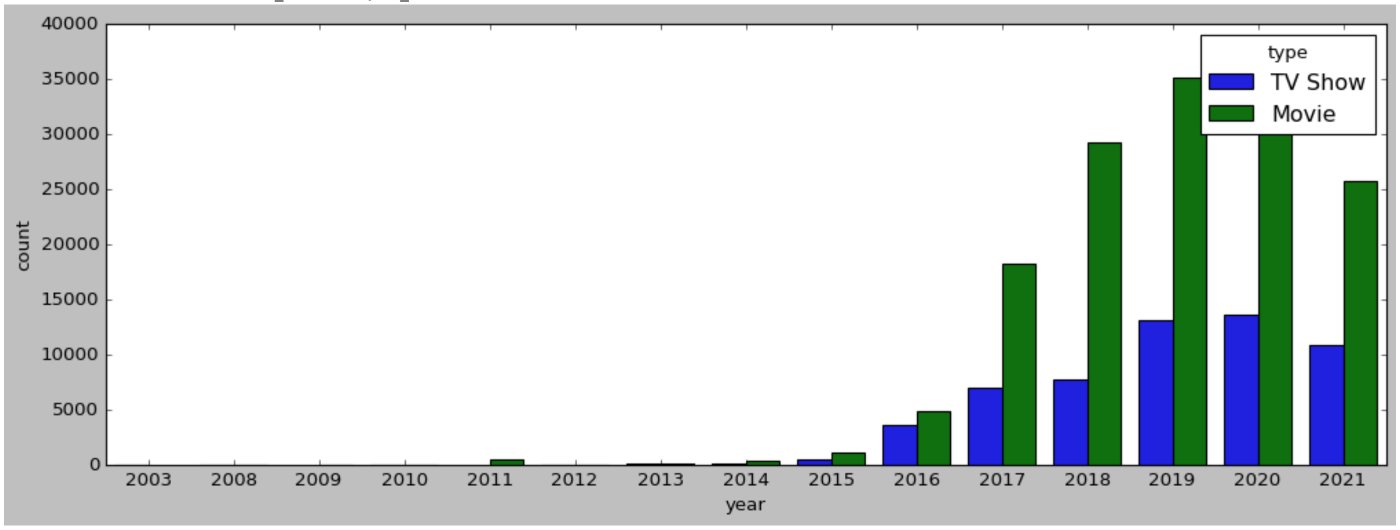
```
<Axes: xlabel='type', ylabel='Duration'>
```



Year wise content production

```
plt.figure(figsize=(15,5))
sns.countplot(x="year",data = final_df, hue="type")
```

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



TOP 3 DIRECTORS FROM EACH COUNTRY

```
def highest_director(x):

  x_without_na = x[(x['unnested_director'] != 'NA') & (x['unnested_director'] !=
  return x_without_na['unnested_director'].value_counts().index.to_list()[:3]
```

```
#Famours actors countrywise
final_df.groupby('unnested_country').apply(highest_director)
```

```
unnested_country
                                                      [Najwa Najjar]
Afghanistan                                      [Pieter-Jan De Pue]
Albania                                          [Antonio Morabito]
Algeria                  [Youssef Chahine, Najwa Najjar, Ma√Øwenn]
Angola                    [Chris Roland, Maradona Dias Dos Santos]
                                             ...
Vatican City                                          [Wim Wenders]
Venezuela        [Sebasti√°n Schindel, Mat√≠as Gueilburt, Jorge...
Vietnam                          [Victor Vu, Ham Tran, Van M. Pham]
West Germany             [Jacek Koprowicz, Mel Stuart, Joachim Fest]
Zimbabwe         [Tomas Brickhill, Camilla Nielsson, Shaul Schw...
Length: 128, dtype: object
```

TOP 3 ACTORS FROM EACH COUNTRY

```
def highest_cast(x):

  x_without_na = x[(x['unnested_cast'] != 'NA') & (x['unnested_cast'] != '')]
  return x_without_na['unnested_cast'].value_counts().index.to_list()[:3]
```

```
final_df.groupby('unnested_country').apply(highest_cast)
```

```
unnested_country
                [Khaled Abol El Naga, Souad Massi, Suhail Haddad]
Afghanistan                                         [Sohrab Nazari]
Albania         [Marco Giallini, Claudio Santamaria, Jerzy Stuhr]
Algeria            [Khaled Abol El Naga, Souad Massi, Ahmed Zaki]
Angola          [Paulo Americano, Raul Rosario, Rapulana Seiph...
                                             ...
Vatican City                                        [Pope Francis]
Venezuela       [Joaqu√≠n Furriel, Luis Ziembrowski, Guillermo...
Vietnam                        [Mai Cat Vi, Le Khanh, Kaity Nguyen]
West Germany               [Ursula Reit, Gene Wilder, Peter Ostrum]
Zimbabwe        [Tendaiishe Chitima, Tendai Nguni, Jesese Mung...
Length: 128, dtype: object
```

```
list = final_df.groupby('unnested_country').apply(highest_cast)
print(list[-10:])
```

```
unnested_country
United Kingdom      [David Attenborough, John Cleese, Michael Palin]
United Kingdom,      [Saleh Bakri, Maryam Kanj, Maryam Kamiel Basha]
United States      [James Franco, Jaden Smith, Anders Danielsen Lie]
United States,                                                    []
Uruguay               [Mirella Pascual, Andr√©s Pazos, Jorge Bolani]
Vatican City                                         [Pope Francis]
Venezuela          [Joaqu√≠n Furriel, Luis Ziembrowski, Guillermo...
Vietnam                       [Mai Cat Vi, Le Khanh, Kaity Nguyen]
West Germany              [Ursula Reit, Gene Wilder, Peter Ostrum]
Zimbabwe           [Tendaiishe Chitima, Tendai Nguni, Jesese Mung...
dtype: object
```

TOP 3 GENRE THAT ARE POPULAR

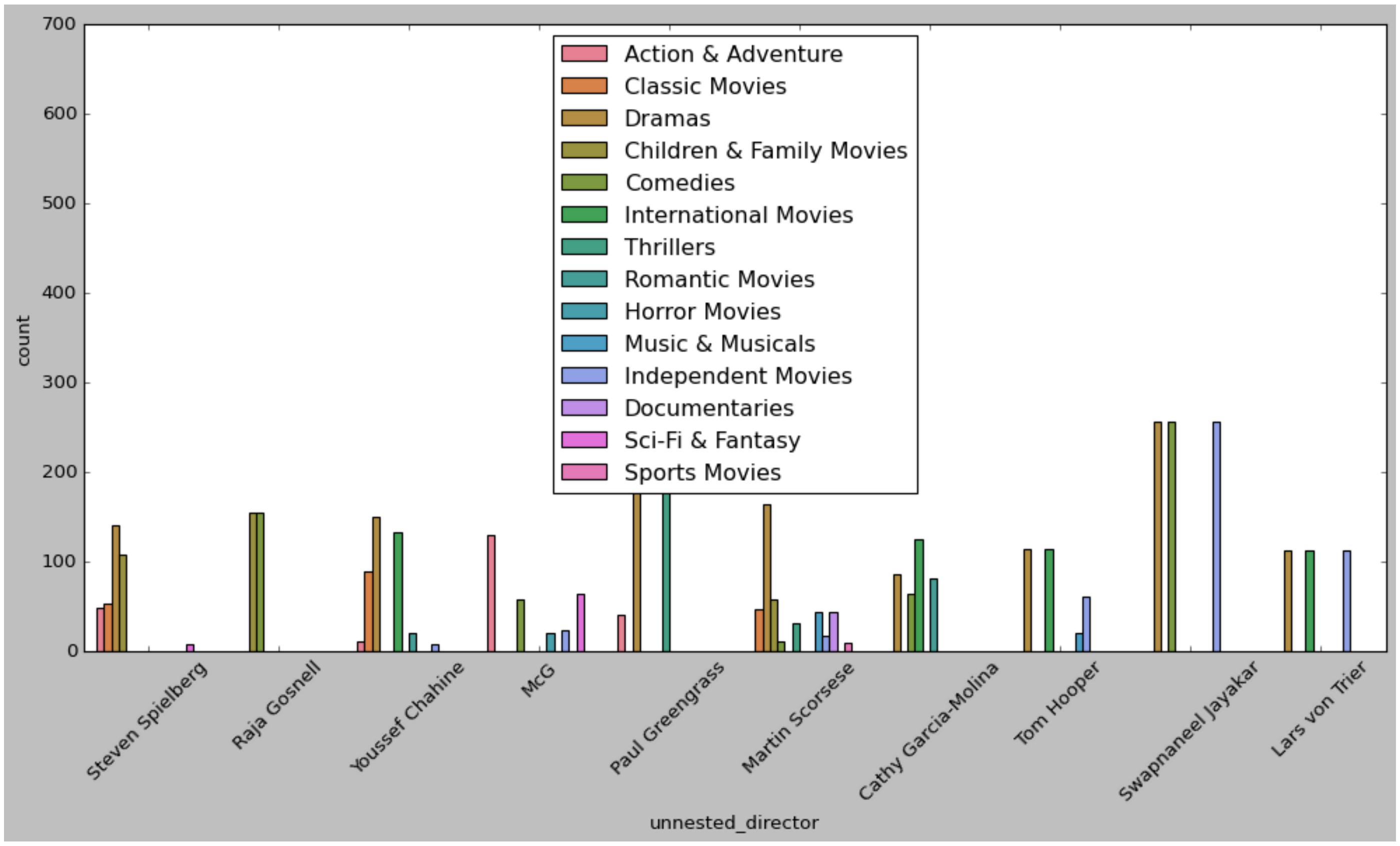
```
lst = final_df['unnested_listed_in'].value_counts().index[:3].to_list()
lst
```

```
['Dramas', 'International Movies', 'Comedies']
```

Type of content produced by famous directors

```
top10_dir = final_df[final_df['unnested_director'] !='NA']
top10_dir = top10_dir['unnested_director'].value_counts().index[:10]
top10_data = final_df[(final_df['unnested_director'].isin(top10_dir))]
```

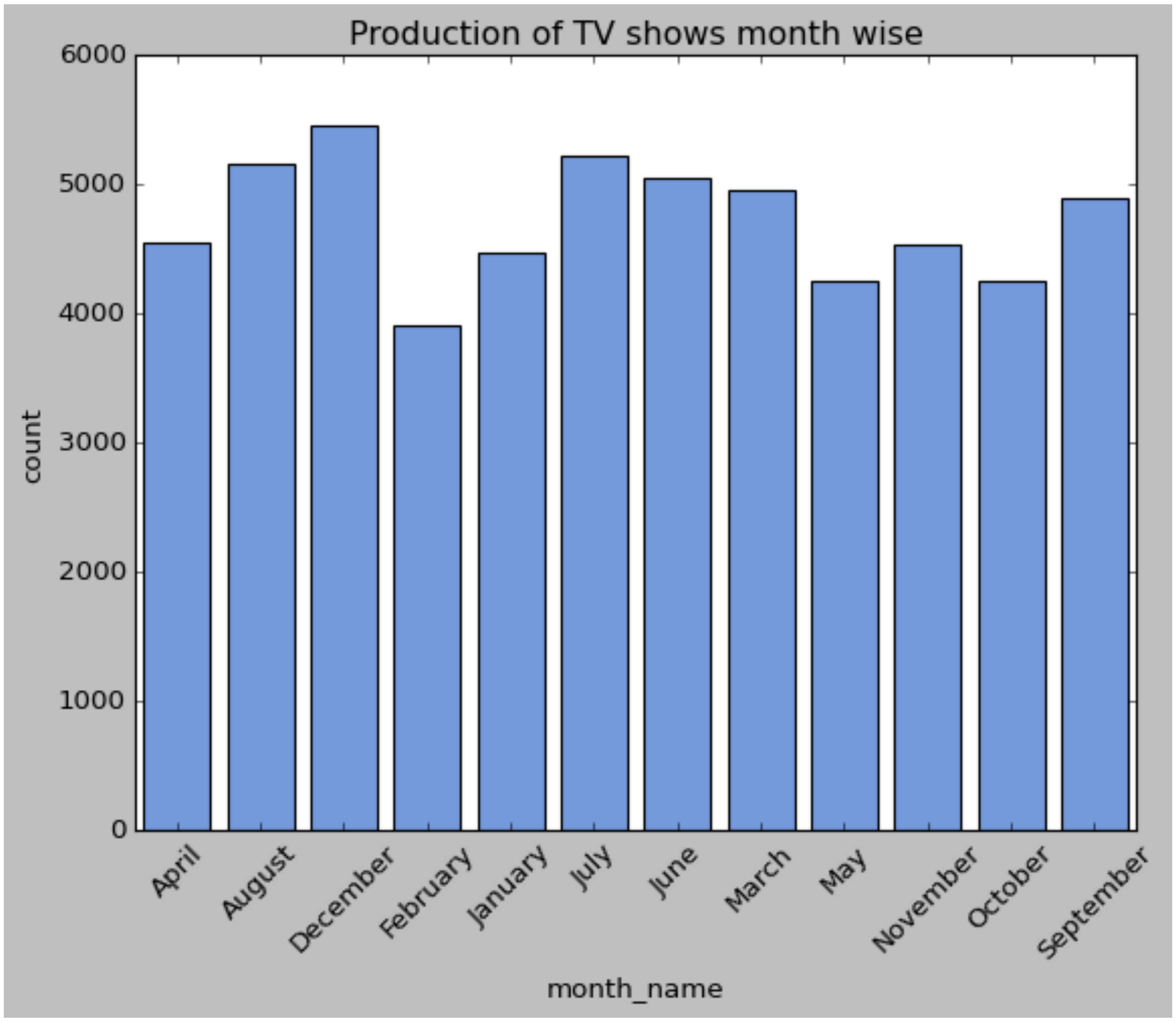
```
plt.figure(figsize=(15,7))
sns.countplot(data = top10_data,x='unnested_director',hue='unnested_listed_in')
plt.legend(loc='upper center')
plt.xticks(rotation=45)
plt.show()
```



- The genre 'International Movies', 'Dramas', 'Action & Adventure' seem to be very popular and content based on this will definitely work

Movies and TV shows added per month

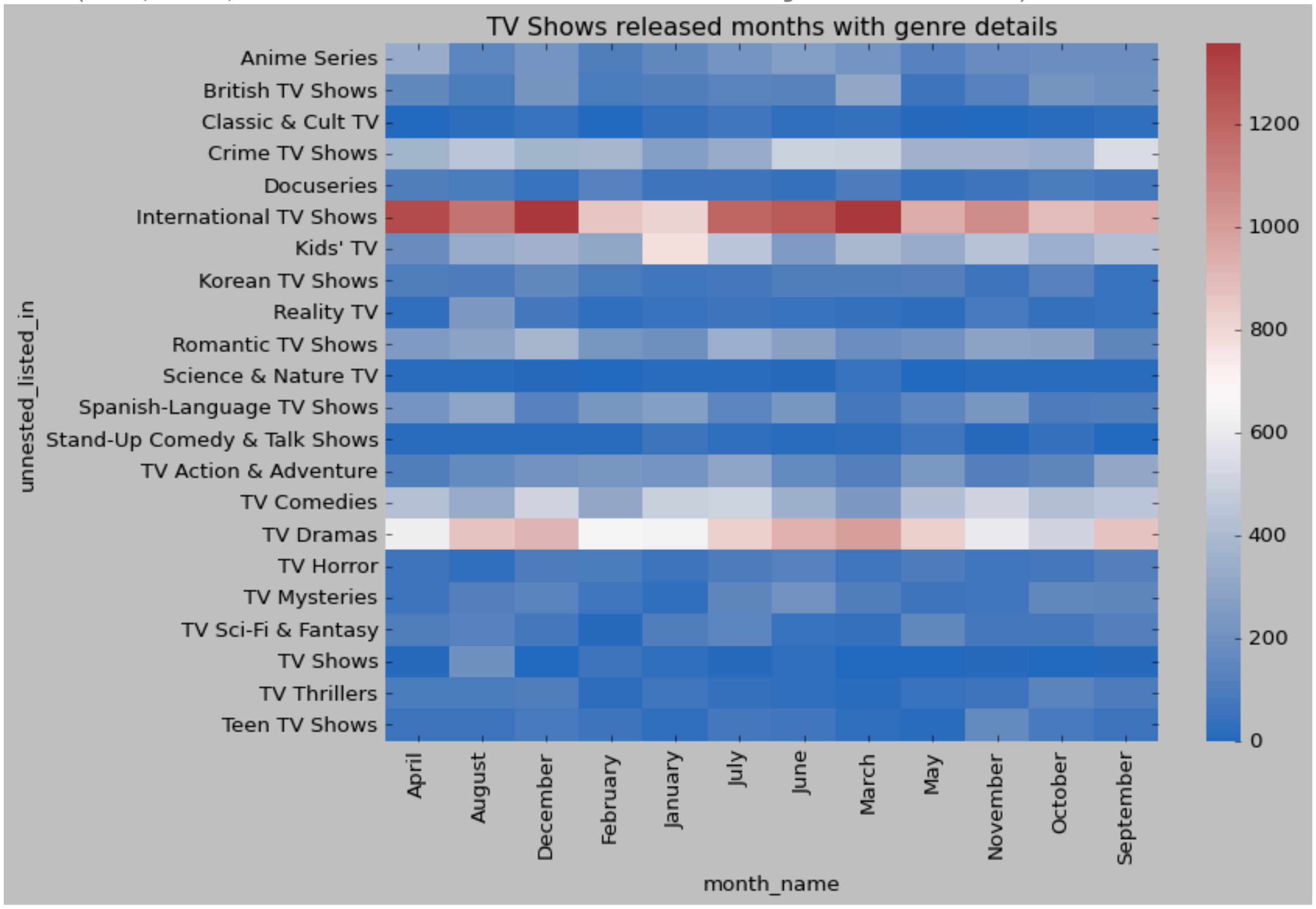
```
TV_show_data = final_df[final_df['type'] == 'TV Show']
counts_df  = TV_show_data.groupby('month_name')['type'].value_counts().reset_inde
counts_df

sns.barplot(x ="month_name" , y = "count", data =counts_df , color = 'cornflowerb
plt.title("Production of TV shows month wise")
plt.xticks(rotation=45)
plt.show()
```



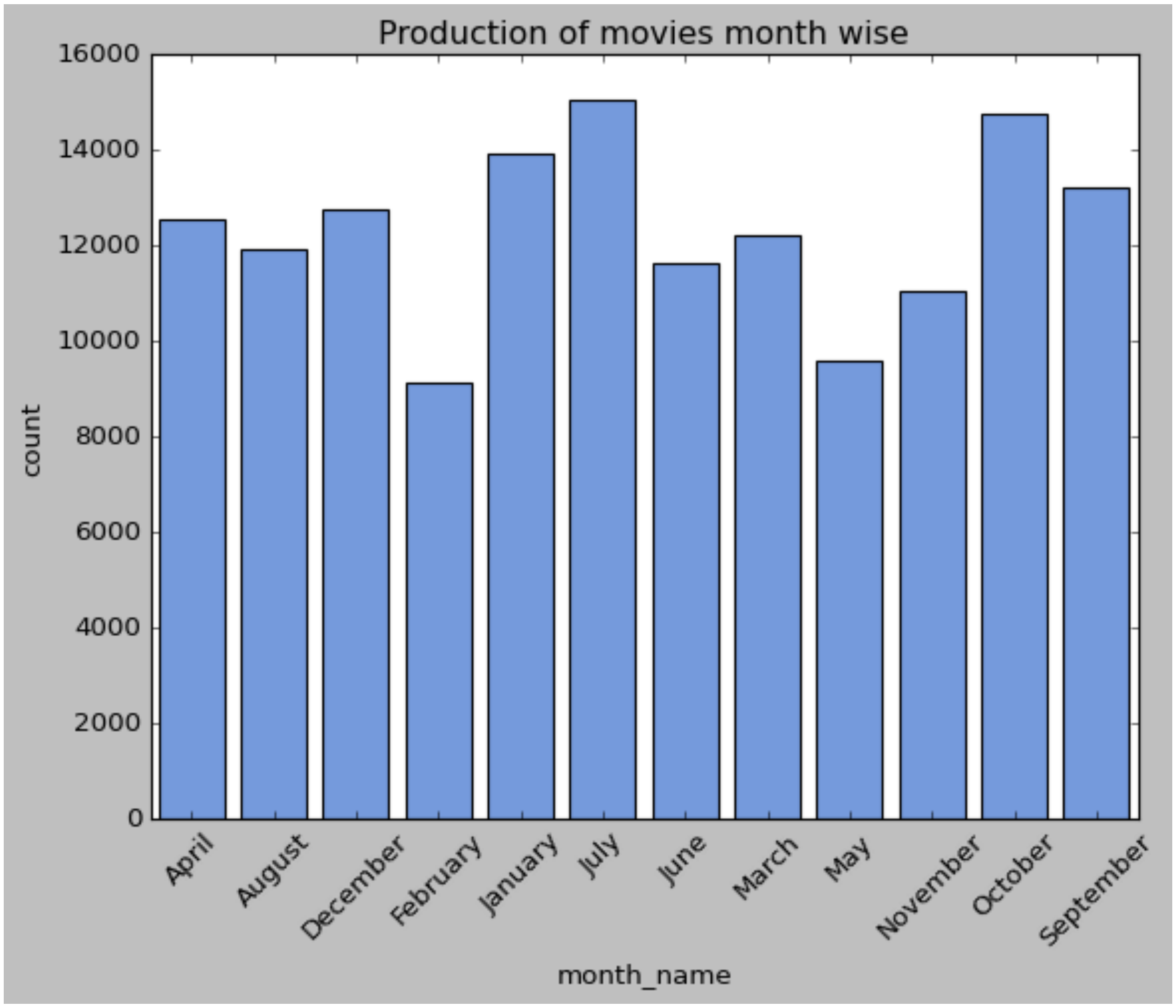
```
#Genre details
```

```
Tv_Shows = final_df.loc[final_df['type'] == 'TV Show']
tv_df = Tv_Shows.groupby('month_name')['unnested_listed_in'].value_counts().unsta

plt.figure(figsize=(10,7))
sns.heatmap(tv_df,cmap="vlag")
plt.title('TV Shows released months with genre details')
```

```
Text(0.5, 1.0, 'TV Shows released months with genre details')
```

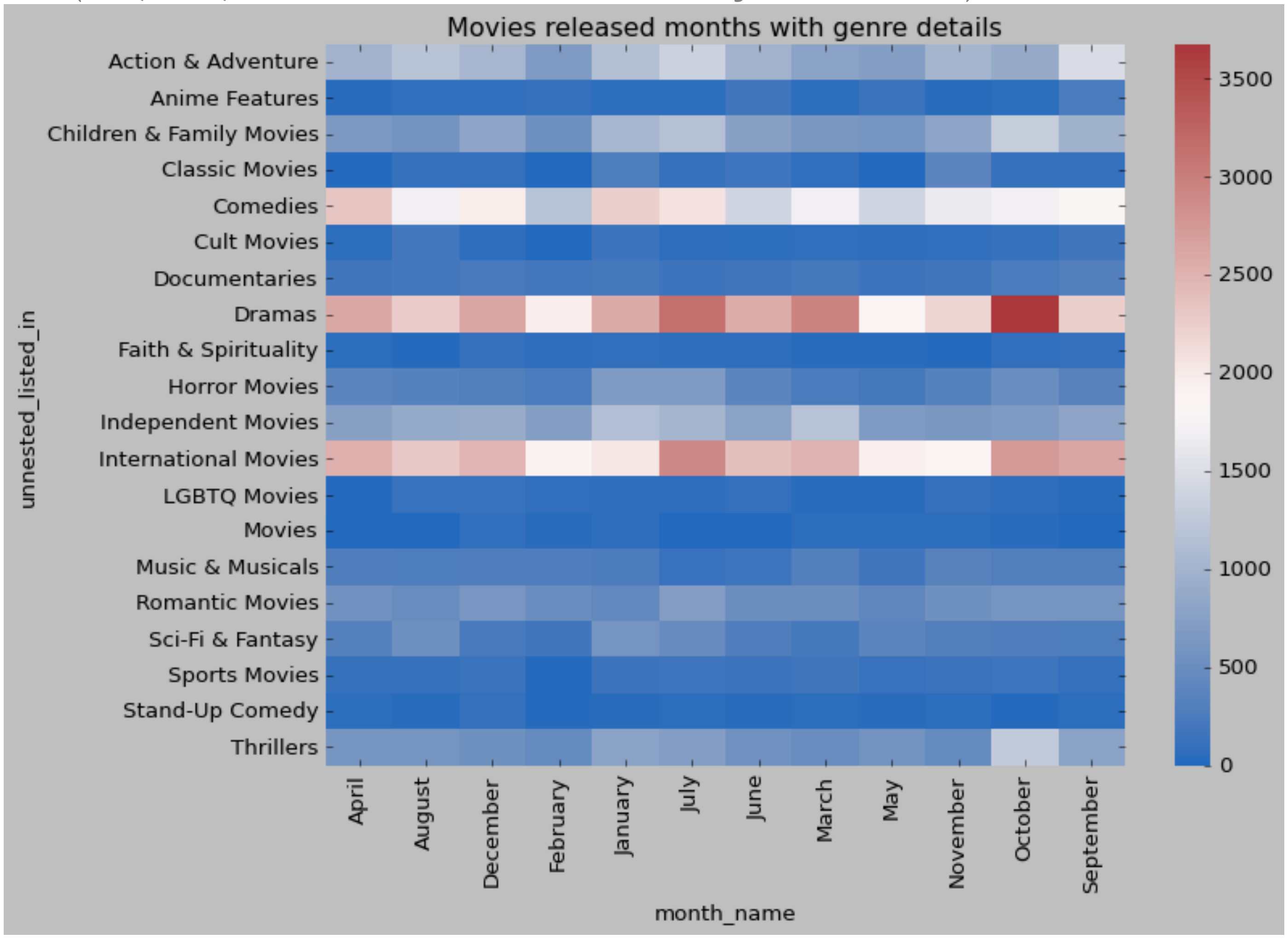


```
Movie_data = final_df[final_df['type'] == 'Movie']
counts_df  = Movie_data.groupby('month_name')['type'].value_counts().reset_index(


sns.barplot(x ="month_name" , y = "count", data =counts_df , color = 'cornflowerb
plt.title("Production of movies month wise")
plt.xticks(rotation=45)
plt.show()
```



```
#Tv_Shows = final_df.loc[final_df['type'] == 'TV Show']
tv_df = Movie_data.groupby('month_name')['unnested_listed_in'].value_counts().uns

plt.figure(figsize=(10,7))
sns.heatmap(tv_df,cmap="vlag")
plt.title('Movies released months with genre details')
```

```
Text(0.5, 1.0, 'Movies released months with genre details')
```



Trial to replace NA values with the most frequently occuring director of that country

```
def replace_director(x):

  x_without_na = x[x['unnested_director'] != 'NA']

  get_dir = x_without_na['unnested_director'].value_counts().index.to_list()[:1]
  x['unnested_director'] =  x['unnested_director'].replace("NA",str(get_dir))
  return x
```

```
#Famours actors countrywise
final_df1 = final_df.groupby('unnested_country', group_keys= False).apply(replace_
final_df1.head(10)
```

	show_id	type	title	date_new	release_year	rating	duration	listed_ir
0	s1	Movie	Dick Johnson Is Dead	2021-09-25	2020	PG-13	90 min	Documentaries
1	s2	TV Show	Blood & Water	2021-09-24	2021	TV-MA	2 Seasons	Internationa TV Shows, TV Dramas, TV Mysteries
2	s2	TV Show	Blood & Water	2021-09-24	2021	TV-MA	2 Seasons	Internationa TV Shows, TV Dramas, TV Mysteries
3	s2	TV Show	Blood & Water	2021-09-24	2021	TV-MA	2 Seasons	Internationa TV Shows, TV Dramas, TV Mysteries
4	s2	TV Show	Blood & Water	2021-09-24	2021	TV-MA	2 Seasons	Internationa TV Shows, TV Dramas, TV Mysteries
5	s2	TV Show	Blood & Water	2021-09-24	2021	TV-MA	2 Seasons	Internationa TV Shows, TV Dramas, TV

								Mysteries
6	s2	TV Show	Blood & Water	2021-09-24	2021	TV-MA	2 Seasons	Internationa TV Shows, TV Dramas, TV Mysteries
7	s2	TV Show	Blood & Water	2021-09-24	2021	TV-MA	2 Seasons	Internationa TV Shows, TV Dramas, TV Mysteries
8	s2	TV Show	Blood & Water	2021-09-24	2021	TV-MA	2 Seasons	Internationa TV Shows, TV Dramas, TV Mysteries
9	s2	TV Show	Blood & Water	2021-09-24	2021	TV-MA	2 Seasons	Internationa TV Shows, TV Dramas, TV Mysteries

INSIGHTS:

- The data has 12 columns and 8809 rows.
- There are null rows for all columns except show_id. Depending on the % of null values, they are treated.
- The date format has been changed from string to date object , with additional columns as month and year that are used further for analysis.
- Columns cast, director, country, listed_in are nested clolumns. These have been unnested and a new dataframe is formed that is used for further analysis.
- The bar plot shows that around 70% of the content produced are movies and thge rest of the content is TV shows.
- Top content producing countries are United States followed by India and United Kingdom.
- For the above mentioned top 10 countries the the the dodged barplot shows that the content proportion varies. In countries like United States, India and United Kingdom theer are more number of movies produced. whereas for countries like Japan and South Korea

there are more number of TV shows produced.

- The line plot show the trend of production of content over years. Around years 2019 to 2021 most movies were produced. Most TV shows prduced around the same time. Though the production declined after 2020.
- Most produced content is Mature content(TV-MA).
- Top 3 popular genres are - 'Dramas', 'International Movies' and 'Comedies'
- TV Shows
    - In general most TV shows are released during December.
    - Genre - 'International TV Shows' are released during the month of March and December and genre 'TV dramas' are released throughout the year.
- Movies
    - In general most movies released during the months of July and October.
    - Most drama movies are released during the month of October and international movies are being released through out the year.

RECOMMENDATIONS:

- Different countries have different predferences for the content like United States, India and United Kingdom produce more movies over TV shows, whereas for countries like Japan and South Korea there are more number of TV shows produced. This insight can help Netflix to understand that the content can be produced accordingly.
- The release monthe of the TV shows and Movies helps Netflix to know what is the best time to release th econtent.
- For TV shows the number of seasons less than 5 seems to work better, whereas the average time duration for movies that viewers prefer is around 100 minutes (i.e. 1 hr 40 min.)
- As most preferred genre in movies looks to be 'Dramas' , so Neflix can consider producing Drama movies more.
- Most preferred genre in TV shows looks to be 'International TV Shows' , so Neflixx can consider producing international Tv shows more.
- There are top 3 directors, cast and genre shown in the notebook. For producing the content

these top choices should be taken into consideration. example - top 3 actors for United Kingdom are United Kingdom [David Attenborough, John Cleese, Michael Palin]

```
Start coding or generate with AI.
```