# 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 and Metric

Analyze the data and generate insights that could help Netflix in deciding which type of shows/movies to produce and how they can grow the business in different countries.

In [1]:

In [2]:

In [3]:

Here the basic Moto is to increase the Business.

# Initial Data Exploration

## *Importing Libraries*

**import** pandas **as** pd **import** numpy **as** np **import** seaborn **as** sns

**import** matplotlib.pyplot **as** plt

df = pd.read\_csv('netflixnew.txt')

df.shape

Out[3]: (8807, 12)

In [4]:

df.info()

<class 'pandas.core.frame.DataFrame'> Rangeindex: 8807 entries, 0 to 8806 Data columns (total 12 columns):

# Column Non-Null Count Dtype

1. show id 8807 non-null object
2. type 8807 non-null object
3. title 8807 non-null object
4. director 6173 non-null object
5. cast 7982 non-null object
6. country 7976 non-null object
7. date added 8797 non-null object
8. release\_year 8807 non-null int64
9. rating 8803 non-null object
10. duration 8804 non-null object
11. listed - in 8807 non-null object
12. description 8807 non-null object

dtypes: int64(1), object(ll)

..,.,................... ,,,...... ,... o-,i::, o. 1/D

In [ 5]: df.head(2)

Out[5]: **show\_id type title director cast country date\_added release\_year rating duration listed\_in description**

Dick As her father

1. s1 Movie Johnson Kirsten NaN United September 2020 PG-13 90 min Documentaries nears the end ls Dead Johnson States 25, 2021 of his life, filmm...

Arna

Qamata, International TV After crossing

1. s2 TV Blood & NaN Khosi South September 2021 TV- 2 Shows, TV paths at a Show Water Ngema, Gail Africa 24, 2021 MA Seasons Dramas, TV party, a Cape

Mabalane, Mysteries Town t...

Thaban...

##### Insights -

* + The Datset consist of 8807 records and 12 Columns.
  + Columns like Director, Cast, Country, Date added, Rating, Duration have few null Values.
  + Release Year Column has int data type, and rest all columns have object Data type.

# Data Preprocessing

In [ 6]: **def** process\_comma\_separated\_column(df, col):

This method will expand columns which have comma separated values

constraint= df[col].apply(lambda x:[i.strip() **for i in** str(x).split(',')]).tolist() df\_new = pd.DataFrame(constraint, index= df['title'])

df\_new = df\_new.stack() df\_new = pd.DataFrame(df\_new)

df\_new.reset\_index(inplace = **True)**

df\_new.drop(axis= 1, columns= ['level\_l'], inplace = **True) return** df new

## Preprocessing Cast Column

In [7]: df\_cast = process\_comma\_separated\_column(df, 'cast') df\_cast.rename(columns = {0:'cast'}, inplace = **True)** df\_cast.loc[df\_cast['cast']=='nan', 'cast']=np.nan df\_cast.head(2)

|  |  |  |
| --- | --- | --- |
| Out[7]: | **title** | **cast** |
|  | 1. Dick Johnson ls Dead 2. Blood & Water | NaN  Arna Qamata |

## Preprocessing Listed\_ln Column

In [8]: df\_listedin = process\_comma\_separated\_column(df, 'listed\_in') df\_listedin.rename(columns = {0:'listed\_in'}, inplace = **True)** df\_listedin.head(2)

|  |  |  |
| --- | --- | --- |
| Out[8]: | **title** | **listed\_in** |
|  | **0** Dick Johnson ls Dead  Blood & Water | Documentaries  International TV Shows |

## Preprocessing Director Column

In [9]: df\_director = process\_comma\_separated\_column(df, 'director') df\_director.rename(columns = {0: 'director'}, inplace = **True)**

df \_director. loc[df\_director ['director']==' nan', 'director'] =np. nan

df\_director.head(2)

|  |  |  |
| --- | --- | --- |
| Out[9]: | **title** | **director** |
|  | **0** Dick Johnson Is Dead  Blood & Water | Kirsten Johnson  NaN |

## Preprocessing Country Column

In [10]: df\_country = process\_comma\_separated\_column(df, 'country') df\_country.rename(columns = {0: 'country'}, inplace = **True)**

df \_country. lac [df\_country ['country']==' nan', 'country'] =np. nan

df \_country. lac [df\_country ['country']==' ', 'country' ]=np. nan df\_country.head(2)

|  |  |  |
| --- | --- | --- |
| Out[10]: | **title** | **country** |
|  | 1. Dick Johnson Is Dead 2. Blood & Water | United States  South Africa |

## Combining all the Data Frames

In [11]: df.drop([ 'director', 'cast', 'country', 'listed\_in', 'description'], axis= 1, inplace = **True)**

df = pd.merge(left = df, right = df\_cast, on = 'title', how = 'left')

df = pd.merge( left = df, right = df\_listedin, on = 'title', how = 'left') df = pd.merge(left = df, right = df\_director, on = 'title', how = 'left') df = pd.merge(left = df, right = df\_country, on = 'title', how= 'left') df.drop\_duplicates(inplace = **True)**

## Final Data Frame

In [12]: df.head(2)

Out[12]: **show\_id type title date\_added release\_year rating duration cast listed\_in director country**

1. s1 Movie Dick Johnson September 25, 2020 PG-13 90 min NaN Documentaries Kirsten United Is Dead 2021 Johnson States
2. s2 TV Blood & Water September 24, 2021 TV- 2 Arna International TV NaN South Show 2021 MA Seasons Shows Africa

Qamata

In [13]:

In [14]:

In [15]:

In [16]:

df.to\_csv('Netflix\_final.csv')

df = pd.read\_csv('Netflix\_final.csv')

df.drop('Unnamed: 0', axis= 1, inplace **=True)**

df.info()

<class 'pandas.core.frame.DataFrame'> Rangeindex: 202010 entries, 0 to 202009 Data columns (total 11 columns):

# Column Non-Null Count Dtype

1. show id 202010 non-null object
2. type 202010 non-null object
3. title 202010 non-null object
4. date added 201852 non-null object
5. release\_year 202010 non-null int64
6. rating 201943 non-null object
7. duration 202007 non-null object
8. cast 199861 non-null object
9. listed in 202010 non-null object
10. director 151367 non-null object
11. country 190007 non-null object dtypes: int64(1), object(10)

memory usage: 17.0+ MB

## Dealing with null values

There are various ways of dealing with Null values -

* Dropping the rows with Null values
* Filling the null values with Mean, Median, Mode
* Treating the missing Values as separate Category

In This Case Sudy, I choose to treat the Missing values as a Separate Category, becuase by imputing them we might end up getting very different Analytical Insights.

Hence I replaced all the missing values as 'Unknown'

In [17]: df['director'] = df['director'].fillna('Unknown') df['country'] = df['country'].fillna('Unknown') df['cast'] = df['cast'].fillna('Unknown')

df['rating'] = df['rating'].fillna('Unknown')

The Durations column has missing values only for Movies Data

In [18]: ## *Extracting the Movie related Dataframe*

df\_movie = df[df['type'] == 'Movie']

df\_movie = df\_movie **[**'title', 'duration']].drop\_duplicates()

## *Extracting the mean of Movie Duration, for imputing the null values*

mean duration= df\_movie['duration'].str.split(' ', expand= True)[0].astype(float).mean()

## *Extracting the number of minutes*

df\_movie['duration\_minutes'] = df\_movie['duration'].str.split(' ' expand= True)[0].astype(float)

## *Imputing the null with Mean value*

df\_movie['duration\_minutes'].fillna(mean\_duration, inplace = **True)**

# Data Exploration

##### Question - The duration (in minutes) of most of the movies present on Netflix is between..

In [19]: plt.hist(df\_movie['duration\_minutes']) plt.title('Movie duration in Minutes') plt.xlabel('Minutes')

plt.show()

**2500**

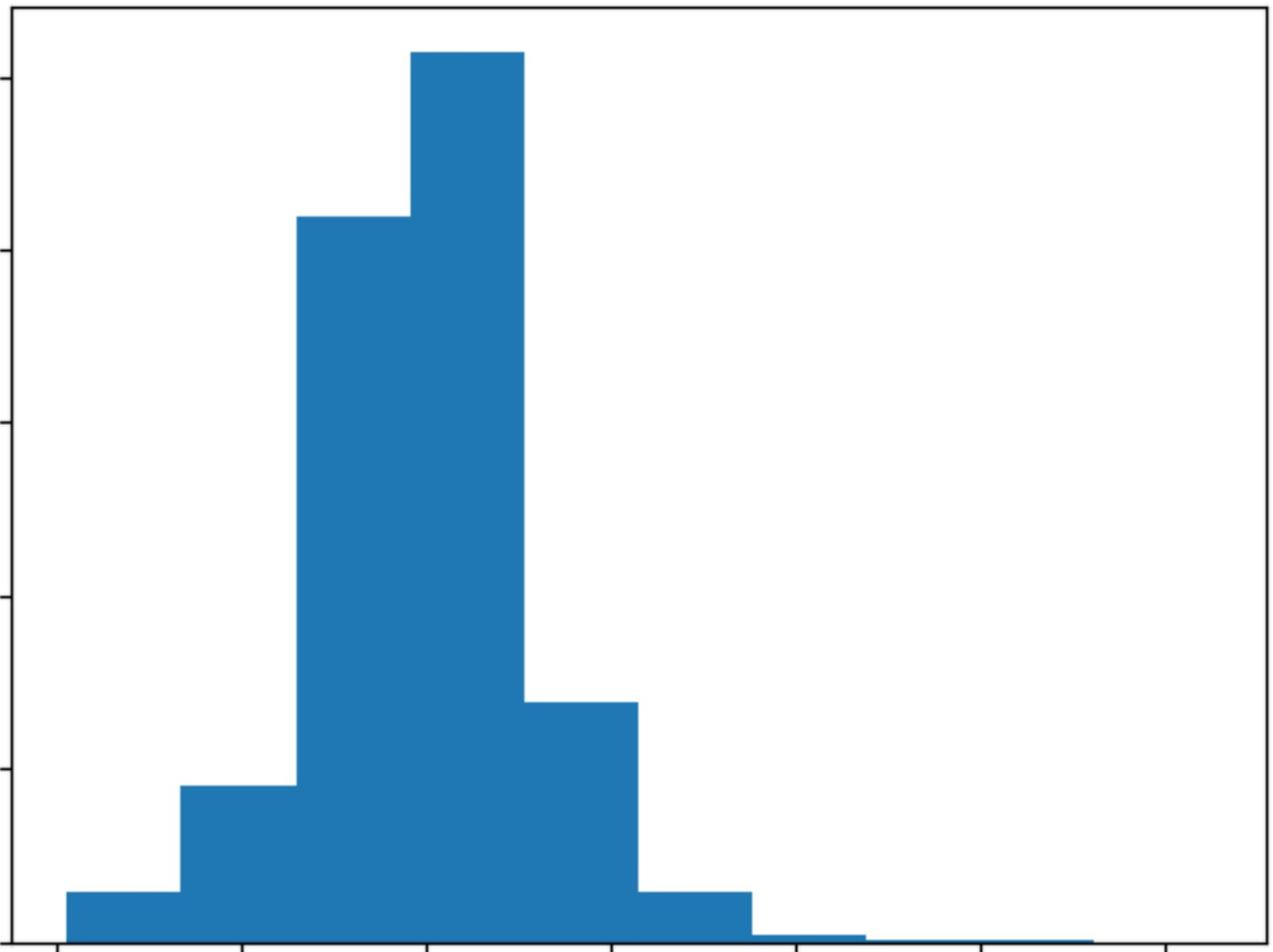
**2000**

**1500**

**1000**

**500**

### Movie duration in Minutes

**0**

**0 50 100 150**

**Minutes**

**200 250 300**

In [20]:

In [21]:

df\_movie['duration\_minutes\_cat'] = pd.cut(df\_movie['duration\_minutes'], bins= [0, 50, 80, 120, 350])

sns.countplot(df\_movie['duration\_minutes\_cat']) plt.title('Movies Duration Categorical Distribution') plt.xlabel('Minutes Category')

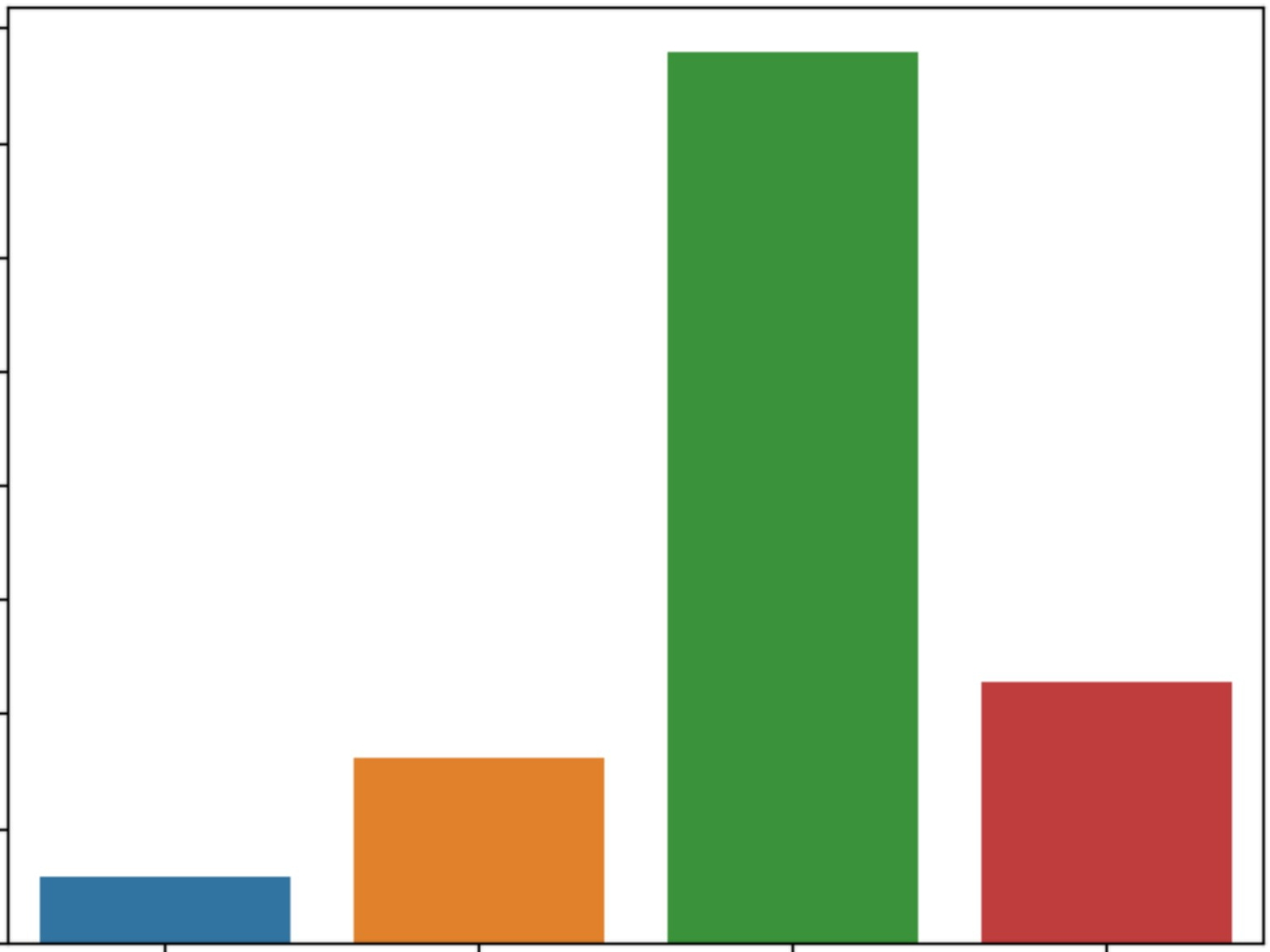
plt.show()

C:\Users\manish\Anaconda3\lib\site-packages\seaborn\\_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be 'data', and passing other arguments with out an explicit keyword will result in an error or misinterpretation.

FutureWarning

...,

**C**



**4000**

**3500**

**3000**

**2500**

### Movies Duration Categorical Distribution

::::, **2000**

8

**1500**

**1000**

**500**

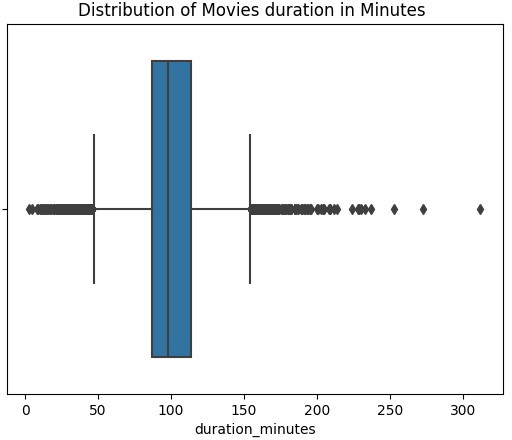
**0**

**(0. 501**

**(50. 801**

**(80. 1201 (120. 3501**

In [22]: sns.boxplot(df\_movie['duration\_minutes']) plt.title('Distribution of Movies duration in Minutes ') plt.show()









In [23]:

In [24]:

df\_tvshow = df[df['type'] == 'TV Show']

df\_tvshow = df\_tvshow[['title', 'duration']].drop\_duplicates()

median\_shows = df\_tvshow['duration'].str.split(' ', expand= True)[0].astype(int).median() print('The Median of number of Sesions on Netflix =', median\_shows)

The Median of number of Sesions on Netflix = 1.0

sns.countplot(df\_tvshow['duration']) plt.xticks(rotation = 45)

plt.title('Distribution of number of Session in TV Shows') plt.show()

C:\Users\manish\Anaconda3\lib\site-packages\seaborn\\_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be 'data', and passing other arguments with out an explicit keyword will result in an error or misinterpretation.

FutureWarning

**1750**i

Insights -

### Distribution of number of Session in TV Shows

■

* Most of the TV Shows have 1 Season, followed by 2 and 3 Season.
* There are very few TV Shows which have greater than 3 Season.

In [25]:

# Data Exploration

##### Question - What is the percentage of TV Shows and Movies Overall? (Comparison of tv shows vs. movies)

df\_type\_title = df[['type', 'title']].drop\_duplicates() df\_type\_title['type'].value\_counts(normalize = **True)**

Out[25]: Movie

TV Show

0.696151

0.303849

In [26]:

Name: type, dtype: float64

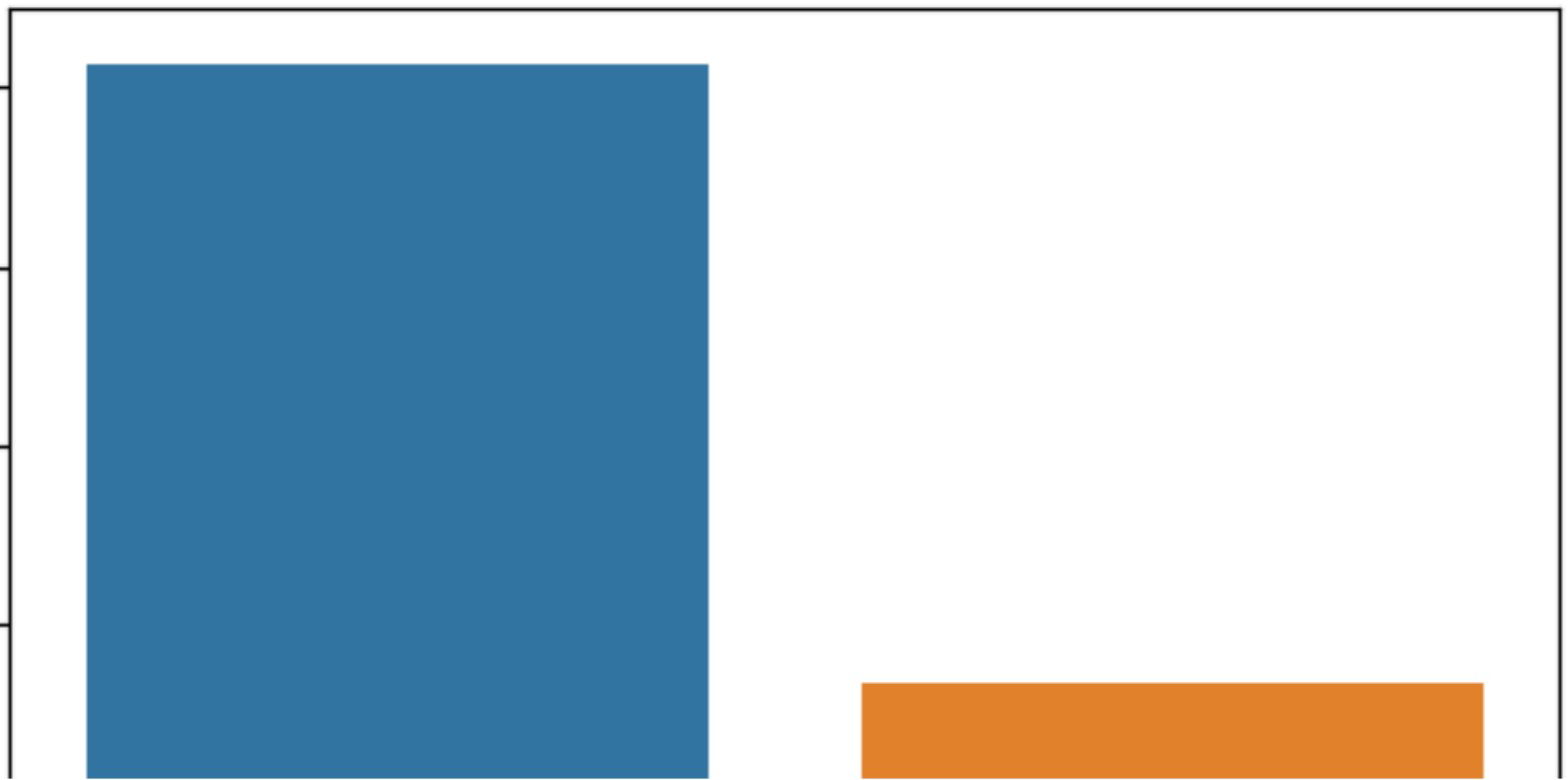
sns.countplot(df\_type\_title['type']) plt.title('Number of Movies vs Number of TV Shows') plt.show()

C:\Users\manish\Anaconda3\lib\site-packages\seaborn\\_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be 'data', and passing other arguments with out an explicit keyword will result in an error or misinterpretation.

FutureWarning

.....

**C**

::,

**6000**

**5000**

**4000**

### Number of Movies vs Number of TV Shows

**8 3000**

Insights -

69.61% of Shows are Movies and 30.38% of Shows are TV Shows

##### Question - What are the top countries with most number of releases of TV Shows and Movies ?

Movies

In [27]: df\_movies = df.loc[df['type'] == 'Movie']

df\_movies\_country = df\_movies[['title','country']].drop\_duplicates()

d = df\_movies\_country.groupby('country').count().reset\_index().sort\_values(by = 'title', ascending= **False).head(10)**

d.columns = ['country', 'count']

d

|  |  |  |
| --- | --- | --- |
| Out[27]: | **country** | **count** |
|  | **110** United States | 2752 |
|  | **41** India | 962 |
|  | **109** United Kingdom | 534 |
|  | **111** Unknown | 446 |
|  | **18** Canada | 319 |

|  |  |  |
| --- | --- | --- |
|  | **country** | **count** |
| **32** | France | 303 |
| **34** | Germany | 182 |
| **97** | Spain | 171 |
| **,tQ** | **l::an::an** | 11 a |

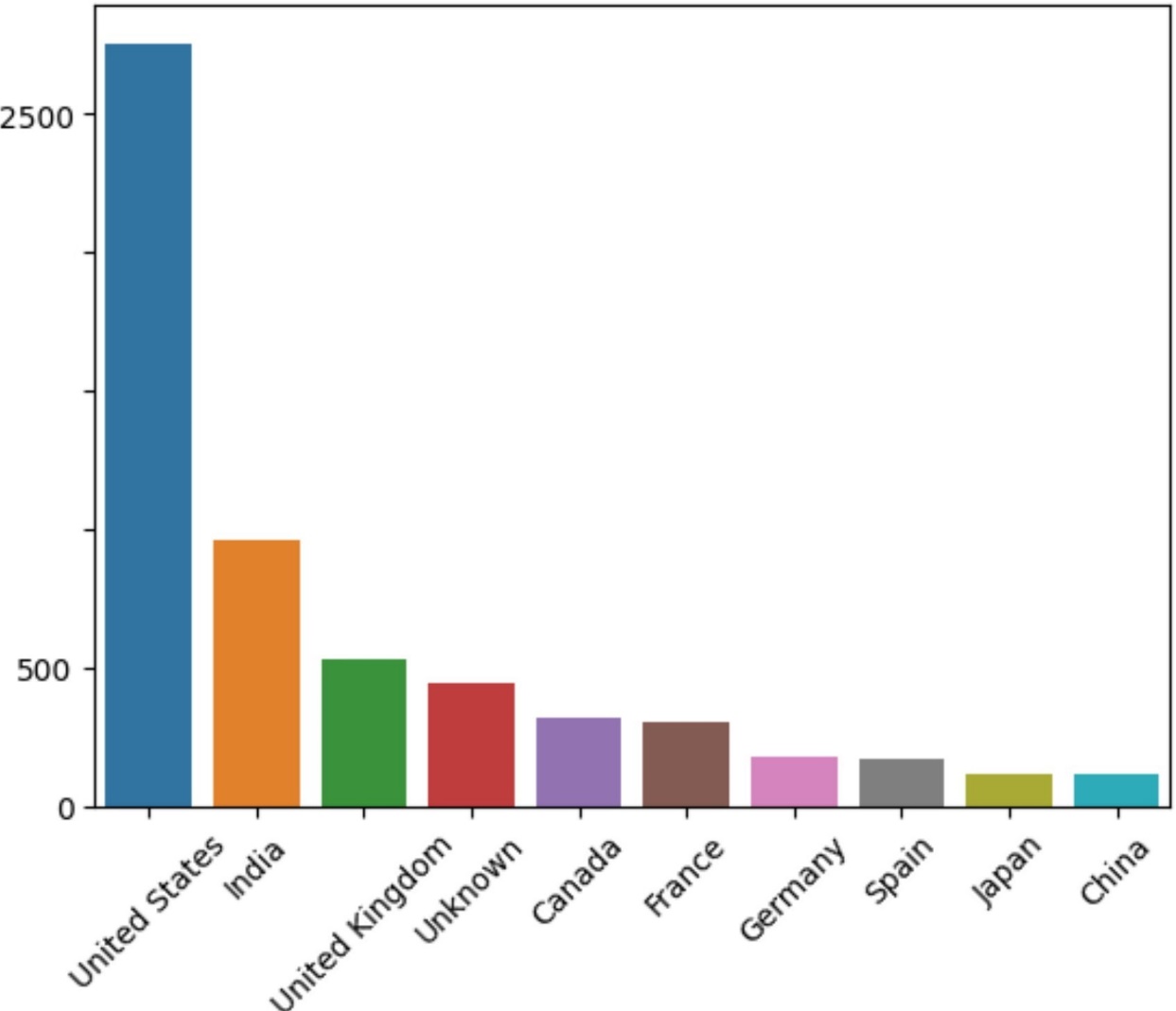
In [28]: sns.barplot(data = d, x = d['country'], y = d['count']) plt.xticks(rotation = 45)

plt.title('Countries with most number of Movie Releases') plt.ylabel('Count of Movies')

plt.show()

### Countries with most number of Movie Releases

**Vl**



**2000**

**1500**

**1000**

**Q)**

'>

**0**

......

.**0**...

**C**

:,

**0u**

**country**

TV Shows

In [29]: df\_tvshows = df.loc[df['type'] == 'TV Show']

df\_tvshows\_country = df\_tvshows[['title','country']].drop\_duplicates()

d = df\_tvshows\_country.groupby('country').count().reset\_index().sort\_values(by = 'title', ascending= **False).head(10)**

d.columns = ['country', 'count']

d

0ut[29]: **country count**

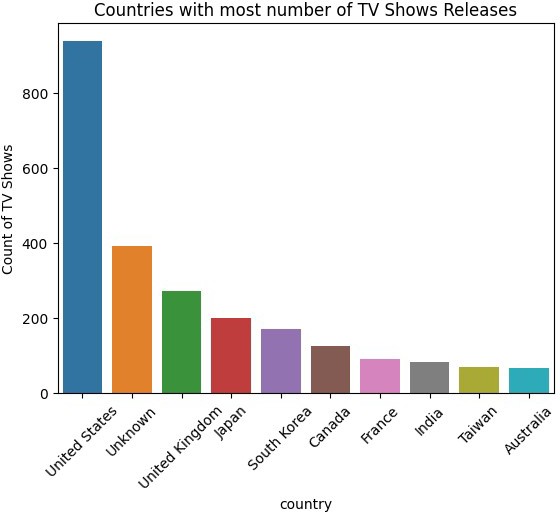
|  |  |  |
| --- | --- | --- |
|  | **country** | **count** |
| **62** | United States | 938 |
| **63** | Unknown | 392 |
| **61** | United Kingdom | 272 |
| **29** | Japan | 199 |
| **51** | South Korea | 170 |
| **7** | Canada | 126 |
| **18** | France | 90 |
| **24** | India | 84 |
| **56** | Taiwan | 70 |

In [30]: sns.barplot(data = d, x = d['country'], y = d['count']) plt.xticks(rotation = 45)

plt.title('Countries with most number of TV Shows Releases') plt.ylabel('Count of TV Shows')

plt.show()

Netflix Business Case Study





##### Question - What are the top countries with most number of releases in the past 5 years?

In [31]: df\_country\_releases = df[['title', 'country', 'release\_year']].drop\_duplicates()

df\_releases = pd.crosstab(df\_country\_releases['country'], columns= df\_country\_releases['release\_year']).reset\_index() df\_releases['average\_releases'] = df\_releases.mean(axis=l).round(2)

In [32]: d

d

= df\_releases.sort\_values(by =

'average\_releases', ascending

= False).head(10).reset\_index().drop('index', axis

= 1)

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Out[32]: | **release\_year** | **country** | **2015** | **2016** | **2017** | **2018** | **2019** | **2020** | **2021** | **average\_releases** |
|  | **0** | United States | 210 | 348 | 442 | 457 | 428 | 384 | 173 | 348.86 |
|  | **1** | Unknown | 44 | 64 | 66 | 111 | 117 | 102 | 210 | 102.00 |
|  | **2** | India | 70 | 80 | 111 | 101 | 93 | 77 | 35 | 81.00 |
|  | **3** | United Kingdom | 66 | 102 | 101 | 96 | 68 | 68 | 36 | 76.71 |
|  | **4** | Canada | 36 | 57 | 58 | 67 | 44 | 39 | 12 | 44.71 |
|  | **5** | France | 33 | 52 | 40 | 47 | 49 | 26 | 13 | 37.14 |
|  | **6** | Spain | 16 | 31 | 33 | 46 | 32 | 31 | 16 | 29.29 |
|  | **7** | Japan | 16 | 25 | 37 | 49 | 36 | 24 | 15 | 28.86 |
|  | **8** | South Korea | 16 | 36 | 33 | 34 | 27 | 31 | 20 | 28.14 |
|  | **9** | Mexico | 9 | 23 | 20 | 25 | 25 | 23 | 13 | 19.71 |

In [33]: yl = d[[2015, 2016, 2017, 2018, 2019, 2020, 2021]].loc[0].values *#USA*

y2 = d[[2015, 2016, 2017, 2018, 2019, 2020, 2021]].loc[2].values *#India*

y3 = d[[2015, 2016, 2017, 2018, 2019, 2020, 2021]].loc[3].values *#UK*

y4 = d[[2015, 2016, 2017, 2018, 2019, 2020, 2021]].loc[4].values *#Canada*

y5 = d[[2015, 2016, 2017, 2018, 2019, 2020, 2021]].loc[5].values *#France*

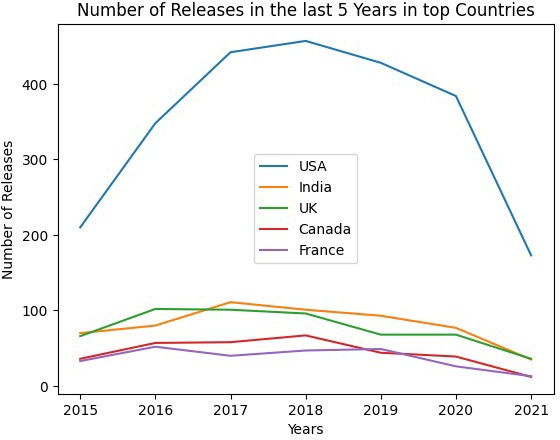
X = [2015, 2016, 2017, 2018, 2019, 2020, 2021]

sns.lineplot(x = x, y = yl, label = 'USA') sns.lineplot(x = x, y = y2, label = 'India') sns.lineplot(x = x, y = y3, label = 'UK') sns.lineplot(x = x, y = y4, label = 'Canada') sns.lineplot(x = x, y = y5, label = 'France') plt.title('Number of Releases in the last 5 Years plt.ylabel('Number of Releases') plt.xlabel('Years')

plt.show()

in top Countries')

Netflix Business Case Study http://localhost:8888/nbconvert/html/Desktop/Coursera/Netflix%20Business%20Case%20Stud...







In [34]:

In [35]:

In [36]:

plt.rcParams["figure.figsize"] = (15,6)

df\_title\_year = df[['title','release\_year', 'type']].drop\_duplicates()

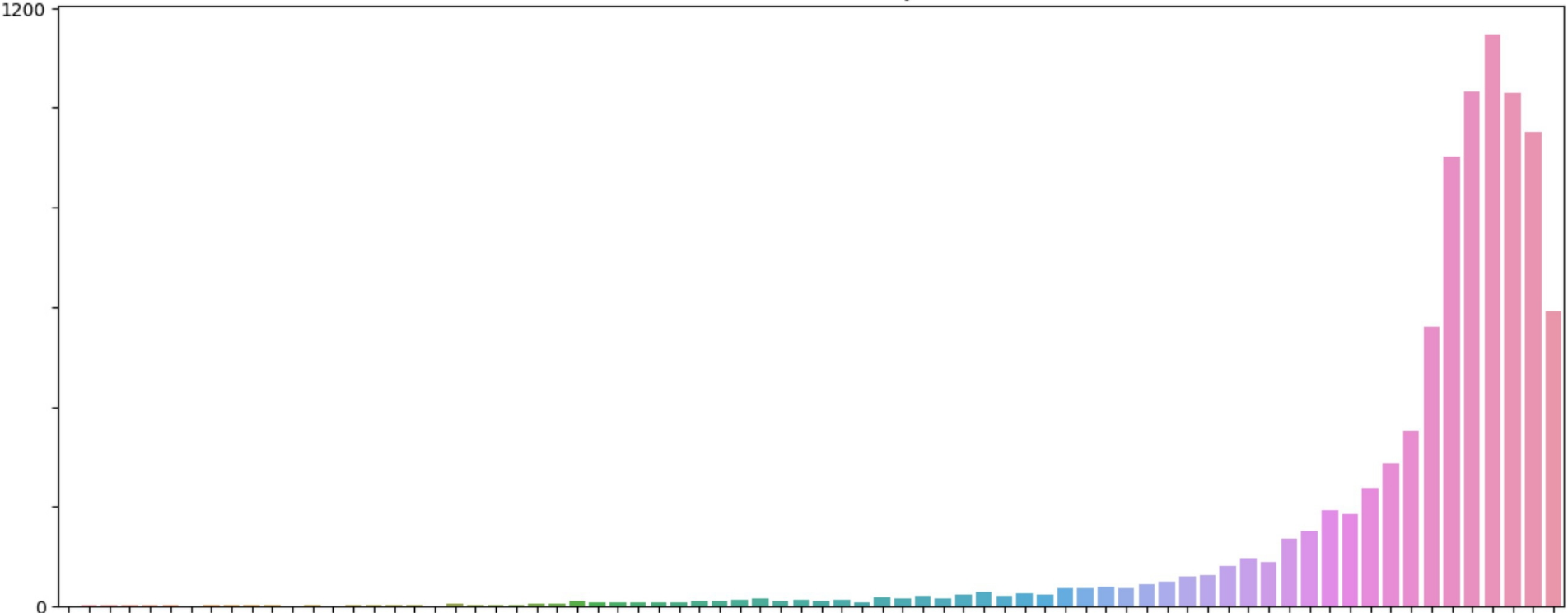
sns.countplot(df\_title\_year['release\_year']) plt.xticks(rotation=90)

plt.title('Count of Releases every Year') plt.show()

C:\Users\manish\Anaconda3\lib\site-packages\seaborn\\_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be 'data', and passing other arguments with out an explicit keyword will result in an error or misinterpretation.

FutureWarning

CountofReleaseseveryYear



1000

800

600

400

200

...,

C:

:,

8

**NMV v romOMNMV romOMNMV romoMNMV romoMNMV romOMNMV romOMNMV romOM**

m**NV**m**V**m**V**m**V**m**V**m**V**mmmmmmmmmmmmmmmmmmmmmmm**r**m**or**m**o**m**r**m**o**m**r**m**o**m**r**m**or**m**o**m**r**m**o**m**r**m**o**m**r**m**or**m**o**m**m**m**m**m**m**m**m**m**m**m**m**m**mm**o**m**o**m**o**O**o**O**o**O**o**O**o**OO**o**O**o**O**o**O**o**O**o**M**o**M**o**MM**o**M**o**M**o**M**o**M**o**M**o**M**o**N**o**N**

**MMMMMMMMMrlMMMMMrlrlrlMriMMMMMMMMMMMrlrlMMMMMMrlMMrlMrlrlrlrlrirlMMNNNNNNNNNNNNNNNNNNNNNN**

**release\_year**

Insights -

* The Number of releases every year increased very gradually in the past. A good jump in growth was seen from 2015.
* The Number of releases every year started to decrease afer reaching the peak in the year 2018. The number of releases decreased

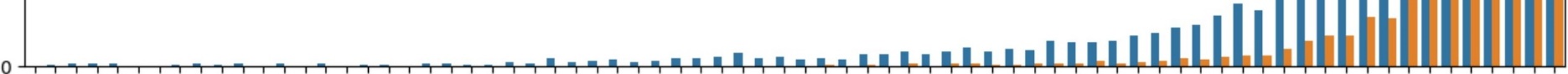
In [37]: sns.countplot(df\_title\_year['release\_year'], hue= df['type']) plt.xticks(rotation=90)

plt.title('Count of Releases every Year') plt.show()

C:\Users\manish\Anaconda3\lib\site-packages\seaborn\\_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be 'data', and passing other arguments with out an explicit keyword will result in an error or misinterpretation.

FutureWarning

Count of Releases every Year



800

700

-

-

type

Movie TVShow

600

500

:, 400

...,

C

8

300

200

100

**NNMVVVvvvvv**

**OOIDOMNMV OOIDOMNMV**

**oOoOoIoDoOoroioNoMoVoooooooomoOoMoNoMmVmmmmomommmOmMmNoMoVoooooooomoooMMNMMMVMMMMoMoMmMoNMN**

**mmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmoooooooooooooooooooooo**

**MrlMMMMMMMMMMMMMMMMrirlMMrlMMMMMMMMMMMrlrlMMMrlMMrlMMMMMMMMMNNNNNNNNNNNNNNNNNNNNNN**

**release\_year**

Insights -

* The number of movies released every year was always greater than number of TV series until 2020. But this trend changed and more number of TV Series were released than Movies in the year 2021.

In [39]:

In [40]:

* The most probable reason could be that after pandamic, the Movie theaters were not operational over a long period, and the OTT culture saw a good growth during this period, giving rise to more number of TV shows getting released.

##### Question - What is the best time to launch a TV show and Movie?

df\_title\_dateAdded = df[['title','date\_added', 'type']].drop\_duplicates()

# *Extracting the month from date\_added*

df\_title\_dateAdded['date\_added'] = pd.to\_datetime(df\_title\_dateAdded['date\_added']) df\_title\_dateAdded['month\_added'] = df\_title\_dateAdded['date\_added'].dt.month df\_title\_dateAdded.head(2)

Out[40]: **title date\_added type month\_added**

**0** Dick Johnson Is Dead 2021-09-25 Movie Blood & Water 2021-09-24 TV Show

9.0

9.0

In [41]: df\_tvshow = df\_title\_dateAdded.loc[df\_title\_dateAdded['type']=='TV Show', :]

month\_added = df\_tvshow['month\_added'].value\_counts().reset\_index().sort\_values('index') x = month\_added['index']

y = month\_added['month\_added'].values

plt.bar(x, height= y)

plt.xticks([l, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12], ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'O

plt.title('Count of TV Shows every Month') plt.xlabel('Months')

plt.ylabel('Count of TV Shows') plt.show()

;v:>:

0.c

Vl

0,.\_,

C:

:,

0u

250

200

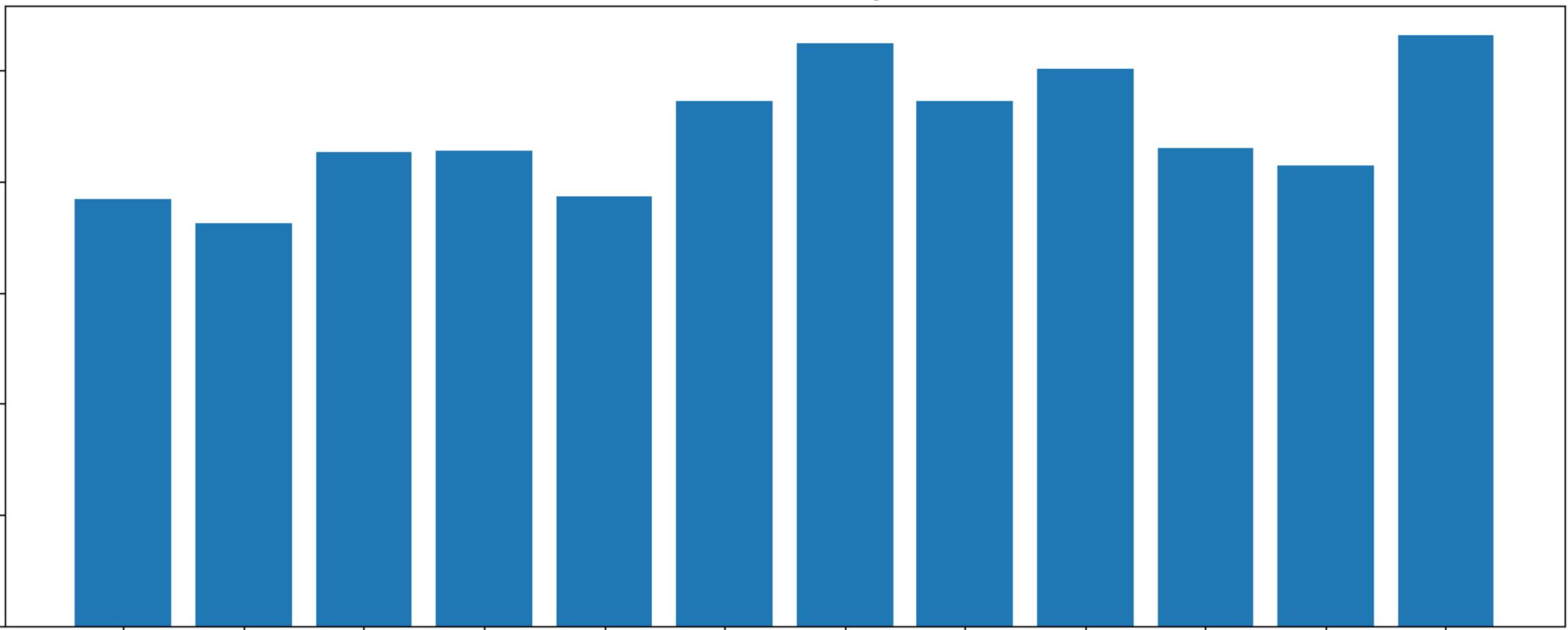
150

100

50

0

Count of lV Shows every Month



Jan Feb Mar Apr May Jun Jul Aug Sep oct Nov Dec

Months

In [43]: d = df\_tvshow['month\_added'].value\_counts().reset\_index() d.columns = ['Month', 'TVShows Added']

d

|  |  |  |  |
| --- | --- | --- | --- |
| Out[43]: |  | **Month** | **TVShows Added** |
|  | **0** | 12.0 | 266 |
|  | **1** | 7.0 | 262 |
|  | **2** | 9.0 | 251 |
|  | **3** | 6.0 | 236 |
|  | **4** | 8.0 | 236 |
|  | **5** | 10.0 | 215 |
|  | **6** | 4.0 | 214 |

|  |  |  |
| --- | --- | --- |
|  | **Month** | **TVShows Added** |
| **7** | 3.0 | 213 |
| **8** | 11.0 | 207 |
| **9** | 5.0 | 193 |
| **1n** | 1 n | 1Q? |

In [44]: df movie= df\_title\_dateAdded.loc[df\_title\_dateAdded['type']=='Movie', :]

month\_added = df\_movie['month\_added'].value\_counts().reset\_index().sort\_values('index') x = month\_added['index']

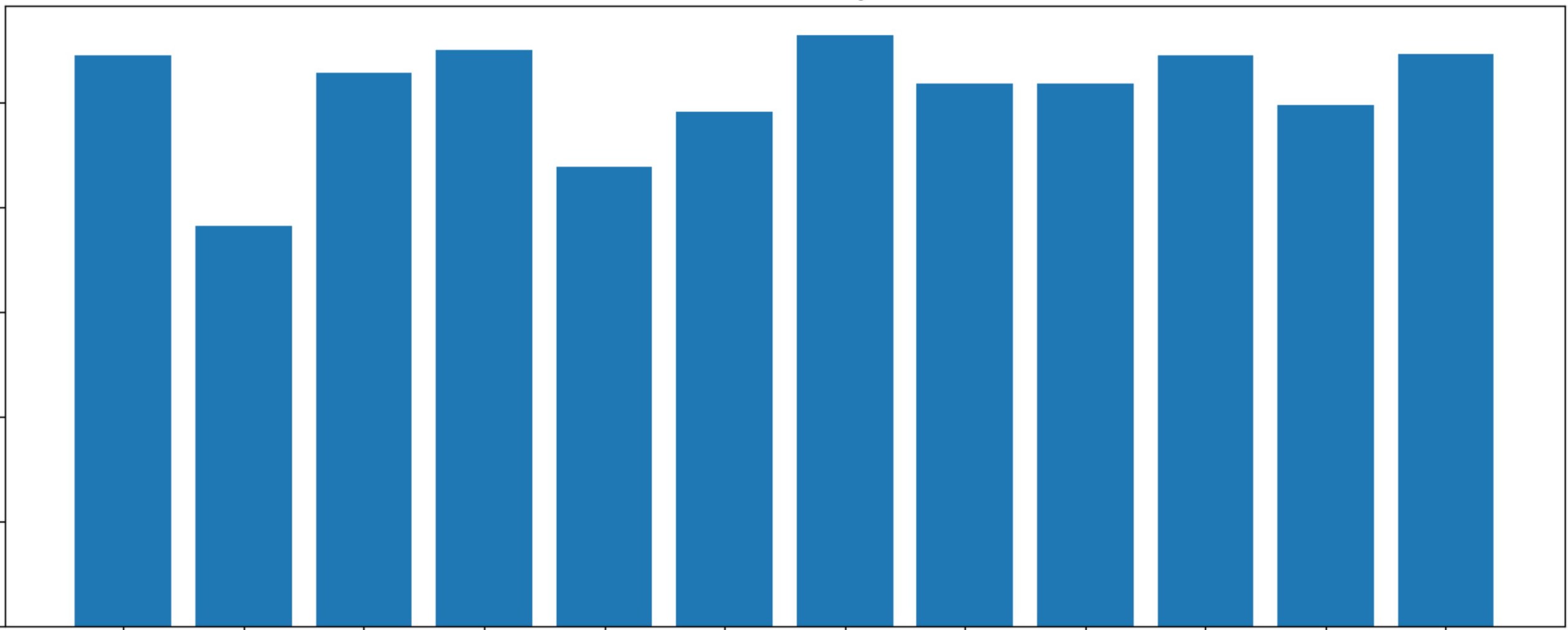
y = month\_added['month\_added'].values

plt.bar(x, height= y)

plt.xticks([l, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12], ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'O

plt.title('Count of Movies every Month') plt.xlabel('Months')

plt.ylabel('Count of Movies') plt.show()

Count of Movies every **Month**

500

400

V)

Q)

·s:

0

:a

..... 300

0

*µ*

C

:,

0u

200

100

0

Jan Feb Mar Apr May Jun Jul Aug Sep oct Nov Dec

Months

In [45]: d = df\_movie['month\_added'].value\_counts().reset\_index() d.columns = ['Month', 'Movies Added']

d

|  |  |  |  |
| --- | --- | --- | --- |
| Out[45]: |  | **Month** | **Movies Added** |
|  | **0** | 7.0 | 565 |
|  | **1** | 4.0 | 550 |
|  | **2** | 12.0 | 547 |
|  | **3** | 1.0 | 546 |
|  | **4** | 10.0 | 545 |
|  | **5** | 3.0 | 529 |
|  | **6** | 9.0 | 519 |

|  |  |  |
| --- | --- | --- |
|  | **Month** | **Movies Added** |
| **7** | 8.0 | 519 |
| **8** | 11.0 | 498 |
| **9**  **1n** | 6.0  c; n | 492 |

Insights -

* In Case of TV Shows, most of them are added in the month of December, July and September
* In Case of Movies, most of them are added in the month of July, April and December
* It can be observed that most of them are added in the midst of Summer and Winter session

###### Business Recommendation

* Its recommended to add shows when people have lesser options to watch, like in the month of February the number of Movies and shows added are less
* Other good time to add shows is when people have more lessure time to watch. In countries like USA, people get holidays during December. So for USA, Decemeber can be a good option to add Shows to Netflix

###### Question - Top Actors of TV Shows

In [47]: df\_tvshow = df.loc[df['type']=='TV Show', :]

df\_tvshow\_cast = df\_tvshow[['cast', 'title']].drop\_duplicates()

*##Hereby using iloc, I made sure that Unknown dosent appear*

d = df\_tvshow\_cast.groupby('cast').count().reset\_index().sort\_values(by = 'title', ascending= **False).iloc[l:].head(10**

d.columns = ['cast', 'count']

d

|  |  |  |  |
| --- | --- | --- | --- |
| 0ut[47]: |  | **cast** | **count** |
|  | **13230** | Takahiro Sakurai | 25 |
|  | **14581** | Yuki Kaji | 19 |
|  | **2873** | Daisuke Ono | 17 |
|  | **251** | Ai Kayano | 17 |

**cast count**

**6804** Junichi Suwabe 17

**14565** Yuichi Nakamura 16

**6761** Jun Fukuyama 15

**14497** Yoshimasa Hosoya 15

**".ii4** ""'-' **r'\\_, .:.-1 1\.,1.J.--'-- -·** ·-'- 1 *A*

In [48]: sns.barplot(data = d, x = d['cast'], y = d['count']) plt.xticks(rotation = 45)

plt.title('Actors with most number of TV Shows Releases') plt.ylabel('Count of TV Shows')

plt.show()

25

20

sV:l

0

.r: 15

Vl

-

.0..,

C

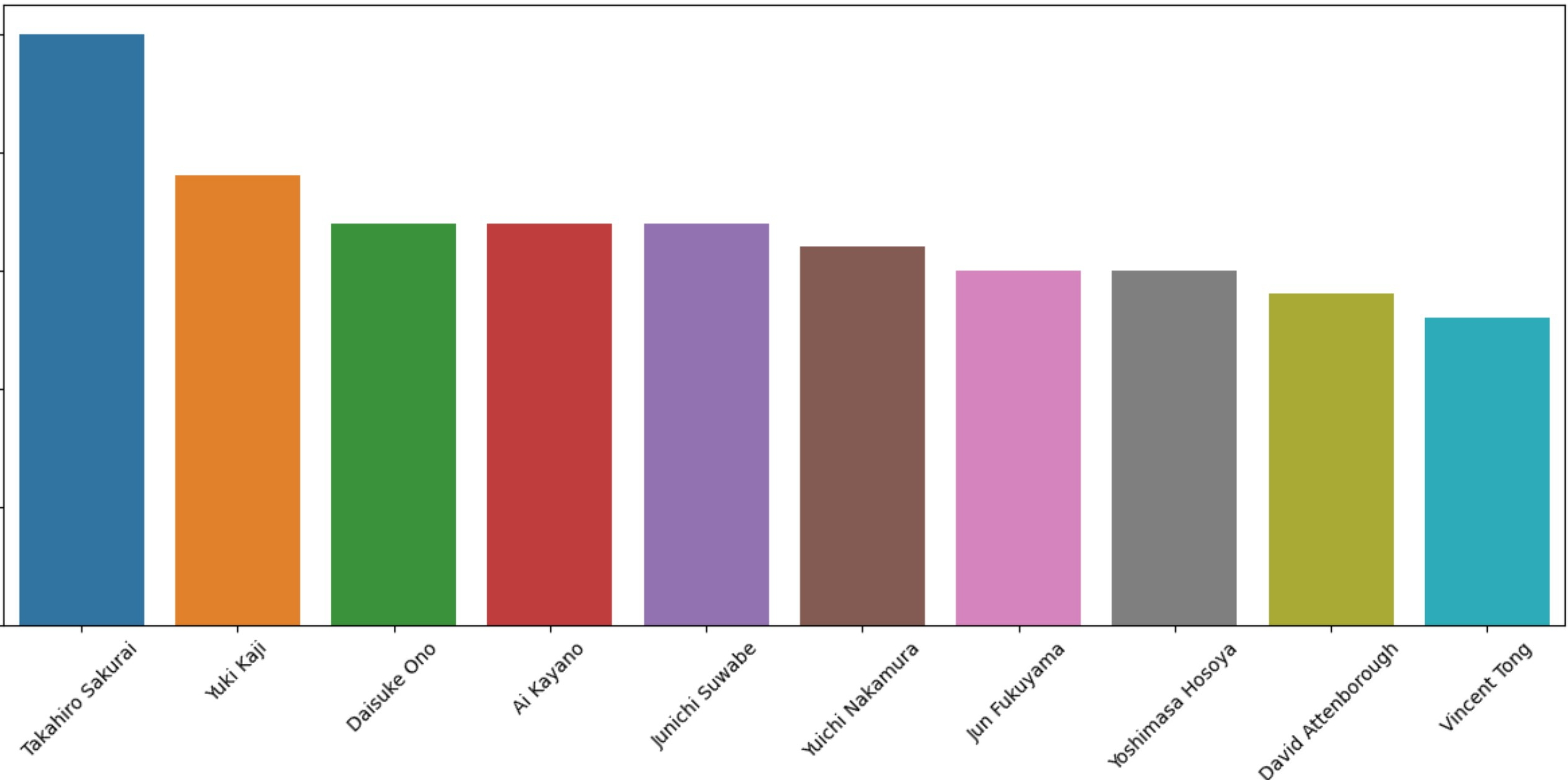
:,

*8* 10

5

0

Actors with most number of lV Shows Releases



cast

##### Insights

* Here Cast Data is missing for many Records.
* Among the existing data Takahiro Sakurai, Yuki Kaji and Daisuke Ono are the top Cast who acted in most number of TV Shows.

##### Question - Top Actors of Movies

In [49]: df\_movies = df.loc[df['type']=='Movie', :]

df\_movies\_cast = df\_movies[['cast', 'title']].drop\_duplicates()

*##Hereby using iloc, I made sure that Unknown dosent appear*

d = df\_movies\_cast.groupby('cast').count().reset\_index().sort\_values(by = 'title', ascending= **False).iloc[l:].head(10**

d.columns = ['cast', 'count']

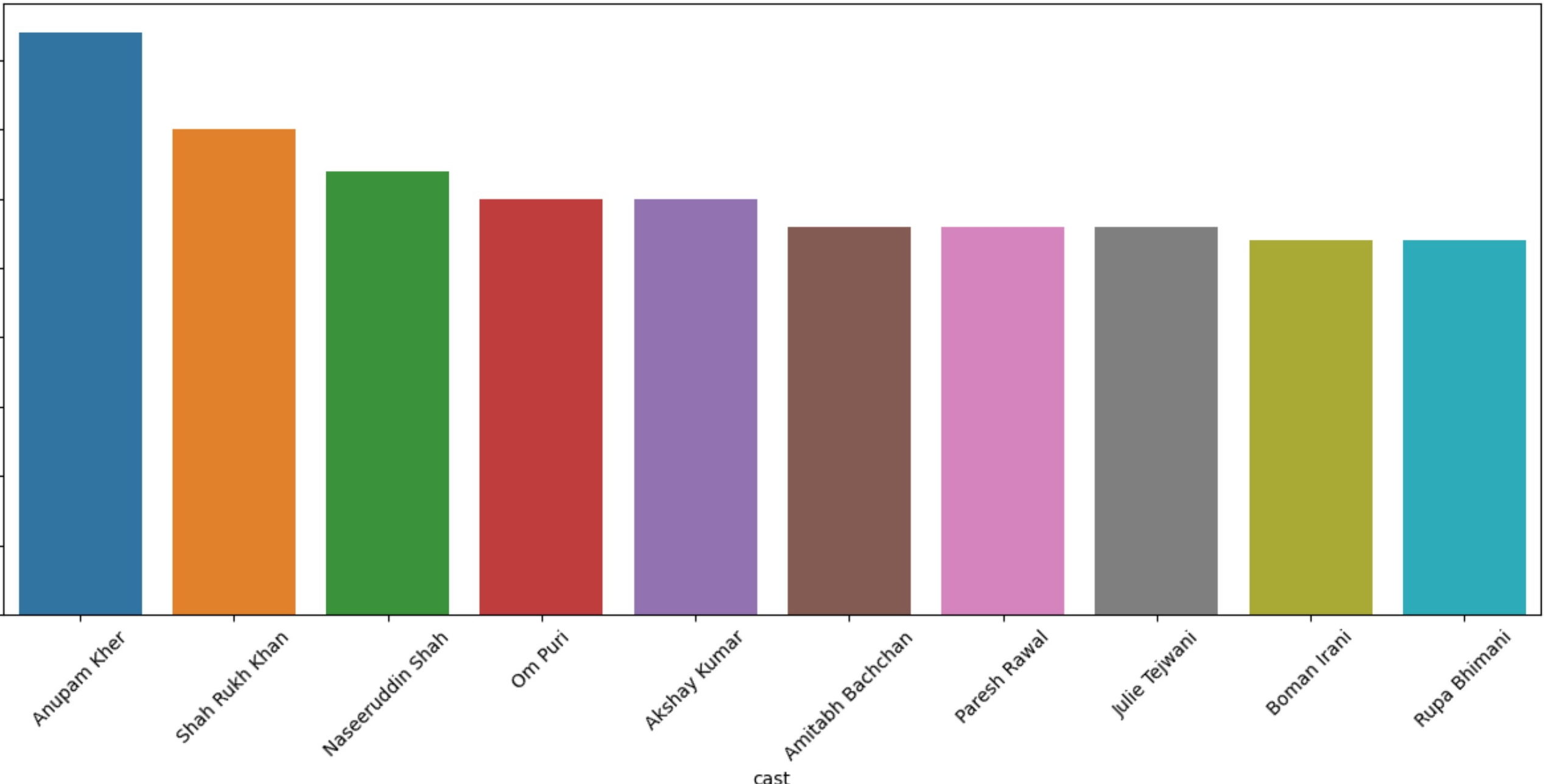
d

|  |  |  |  |
| --- | --- | --- | --- |
| Out [ 49]: |  | **cast** | **count** |
|  | **2104** | Anupam Kher | 42 |
|  | **21781** | Shah Rukh Khan | 35 |
|  | **17193** | Naseeruddin Shah | 32 |
|  | **18064** | Om Puri | 30 |
|  | **637** | Akshay Kumar | 30 |
|  | **1312** | Amitabh Bachchan | 28 |
|  | **18329** | Paresh Rawal | 28 |
|  | **12031** | Julie Tejwani | 28 |
|  | **3353** | Boman Irani | 27 |
|  | **20692** | Rupa Bhimani | 27 |

In [51]: sns.barplot(data = d, x = d ['cast'], y = d [ 'count' ]) plt.xticks(rotation = 45)

plt.title('Actors with most number of Movie Releases') plt.ylabel('Count of Movies')

plt. show()

Actors with most number of Movie Releases

40

35

30

·"c:u;';:25

:;:

0

....

0..,

C: 20

:,

0u

15

10

5

0

###### Insights

* Here Cast Data is missing for many Records.
* Among the existing data Anupam Kher, Shah Rukh Khan and Naseeruddin Shah are the top Cast who acted in most number of Movies.

###### Question - Top Directors of TV Shows

In [52]: df\_tvshow = df.loc[df['type']=='TV Show', :]

df\_tvshow\_director = df\_tvshow[['director', 'title']].drop\_duplicates()

d = df\_tvshow\_director.groupby('director').count().reset\_index().sort\_values(by = 'title', ascending= **False).iloc[l:]**

d.columns = ['Director', 'Count']

d

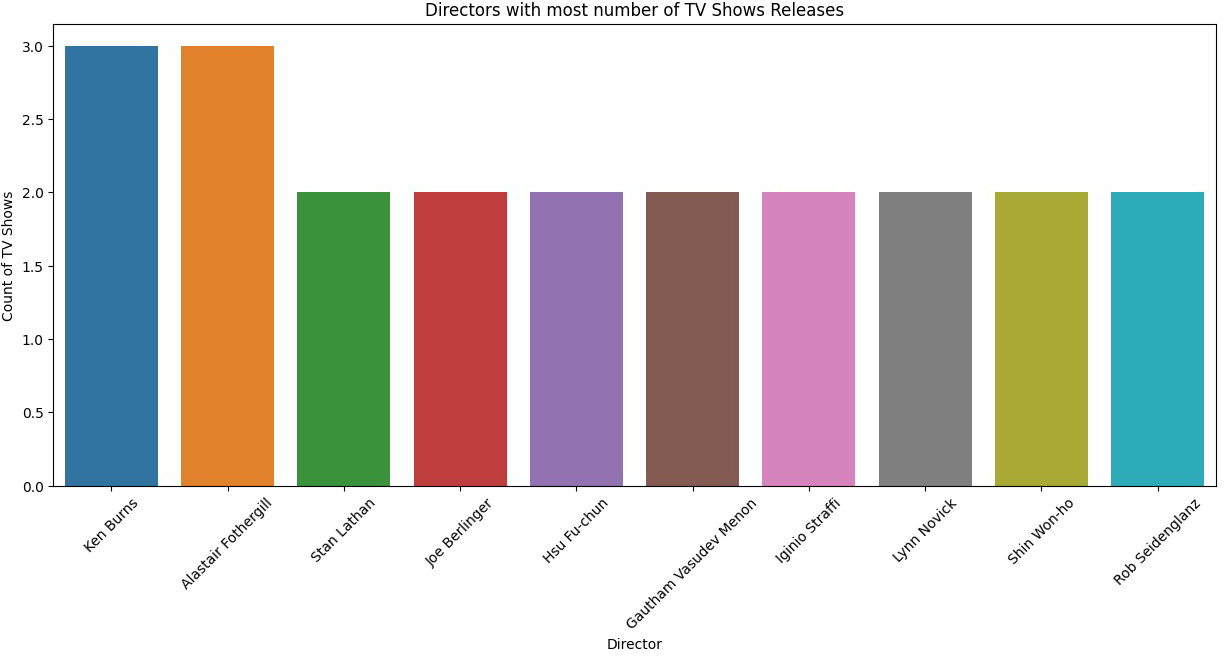
|  |  |  |  |
| --- | --- | --- | --- |
| Out[52]: |  | **Director** | **Count** |
|  | **146** | Ken Burns | 3 |
|  | **8** | Alastair Fothergill | 3 |
|  | **259** | Stan Lathan | 2 |
|  | **128** | Joe Berlinger | 2 |
|  | **100** | Hsu Fu-chun | 2 |
|  | **84** | Gautham Vasudev Menon | 2 |
|  | **103** | lginio Straffi | 2 |
|  | **168** | Lynn Novick | 2 |
|  | **251** | Shin Won-ho | 2 |
|  | **235** | Rob Seidenglanz | 2 |

In [54]: sns.barplot(data = d, x = d['Director'], y = d['Count']) plt.xticks(rotation = 45)

plt.title('Directors with most number of TV Shows Releases') plt.ylabel('Count of TV Shows')

plt.show()

Netflix Business Case Study







In [55]: df movies= df.loc[df['type']=='Movie', :]

df\_movie\_director = df\_movies[['director', 'title']].drop\_duplicates()

d = df\_movie\_director.groupby('director').count().reset\_index().sort\_values(by = 'title', ascending= **False).iloc[l:].**

d.columns = ['Director', 'Count']

d

|  |  |  |  |
| --- | --- | --- | --- |
| Out[55]: |  | **Director** | **Count** |
|  | **3582** | Rajiv Chilaka | 22 |
|  | **1817** | Jan Suter | 21 |
|  | **3633** | Raul Campos | 19 |
|  | **4261** | Suhas Kadav | 16 |
|  | **2739** | Marcus Raboy | 15 |
|  | **1862** | Jay Karas | 15 |
|  | **727** | Cathy Garcia-Molina | 13 |
|  | **2815** | Martin Scorsese | 12 |
|  | **1859** | Jay Chapman | 12 |
|  | **4726** | Youssef Chahine | 12 |

In [56]: sns.barplot(data = d, x = d['Director'], y = d['Count']) plt.xticks(rotation = 45)

plt.title('Directors with most number of Movie Releases') plt.ylabel('Count of Movies')

plt.show()

**20**

**15**

·"**c**:**u**;';:

**0**

....

**0**..,

**C:**

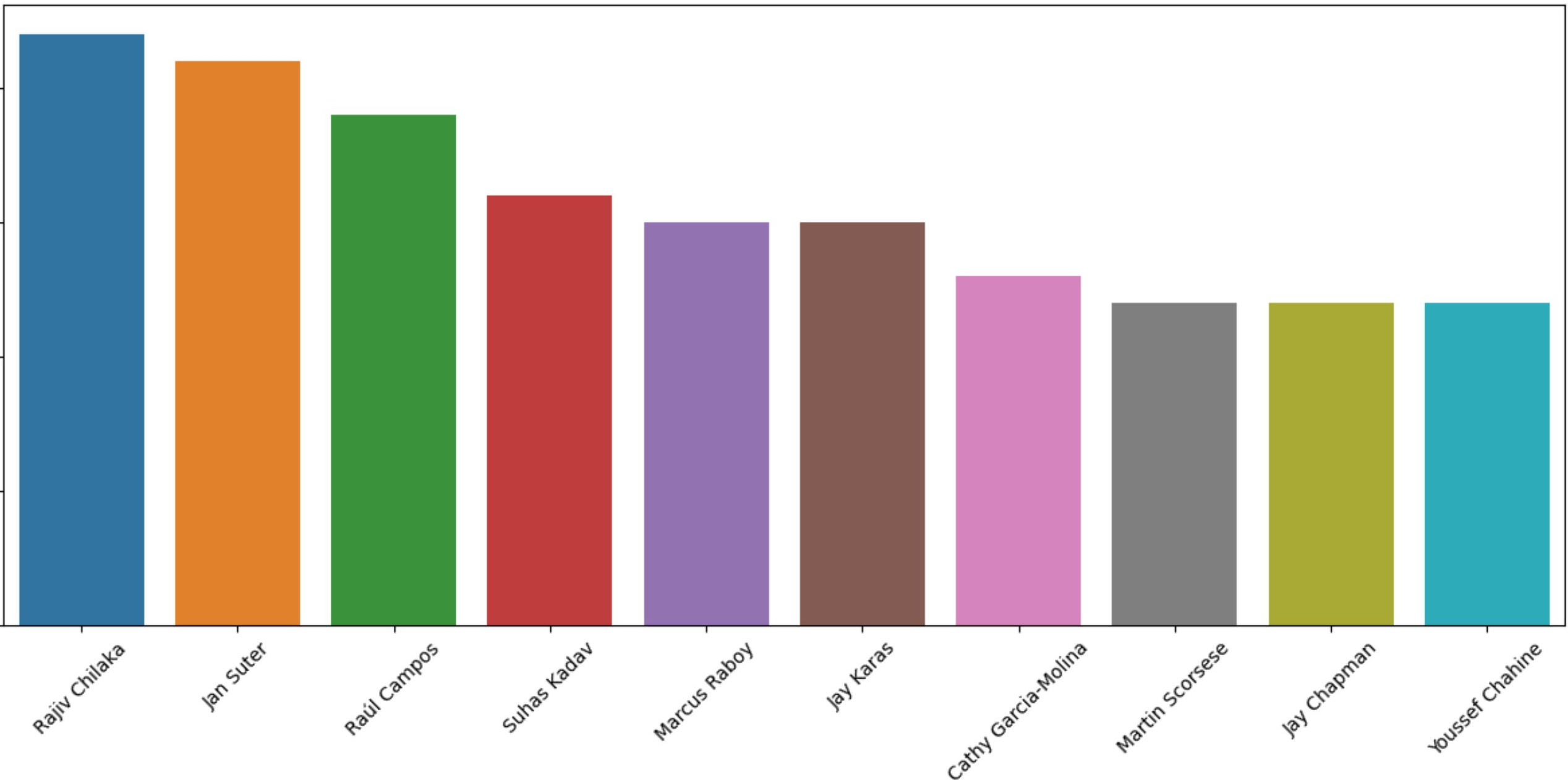
:, **10**

**0u**

**5**

**0**

**Directors with most number of Movie Releases**



**Director**

##### Insights

* Here Director Data is missing for many Records.
* Among the existing data Rajiv Chilaka, Jan Suter and Raul Campos are among the top Directors who directed most number of Movies.

##### Question - What are all the Genres present on the Netflix Platform ?

In [92]: df['listed\_in'].value\_counts()

|  |  |  |
| --- | --- | --- |
| Out[92]: | Dramas | 29787 |
|  | International Movies | 28224 |
|  | Comedies | 20829 |

Netflix Business Case Study





In [57]: df\_tvshow = df.loc[df['type']=='TV Show', :]

df\_tvshow\_gen re = df\_tvshow[['listed\_in' , 'title' ]]•drop\_duplicates()

d = df\_tvshow\_genre.groupby('listed\_in').count().reset\_index().sort\_values(by = 'title', ascending= **False).head(10)**

d.columns = [ 'listed\_in', 'Count' ]

d

|  |  |  |  |
| --- | --- | --- | --- |
| Out[57]: |  | **listed\_in** | **Count** |
|  | **5** | International TV Shows | 1351 |
|  | **15** | TV Dramas | 763 |
|  | **14** | TV Comedies | 581 |
|  | **3** | Crime TV Shows | 470 |
|  | **6** | Kids' TV | 451 |
|  | **4** | Docuseries | 395 |
|  | **9** | Romantic TV Shows | 370 |
|  | **8** | Reality TV | 255 |
|  | **1** | British TV Shows | 253 |
|  | **0** | Anime Series | 176 |

In [58]: sns.barplot(data = d, x = d['listed\_in'], y = d['Count']) plt.xticks(rotation = 45)

plt.title('Genres with most number of TV Shows Releases') plt.ylabel('Count of TV Shows')

plt.show()

1400

1200

1000

:;:

*V,*

.0c 800

Vl

0..,

Genres with most number of TV Shows Releases

<::

::,

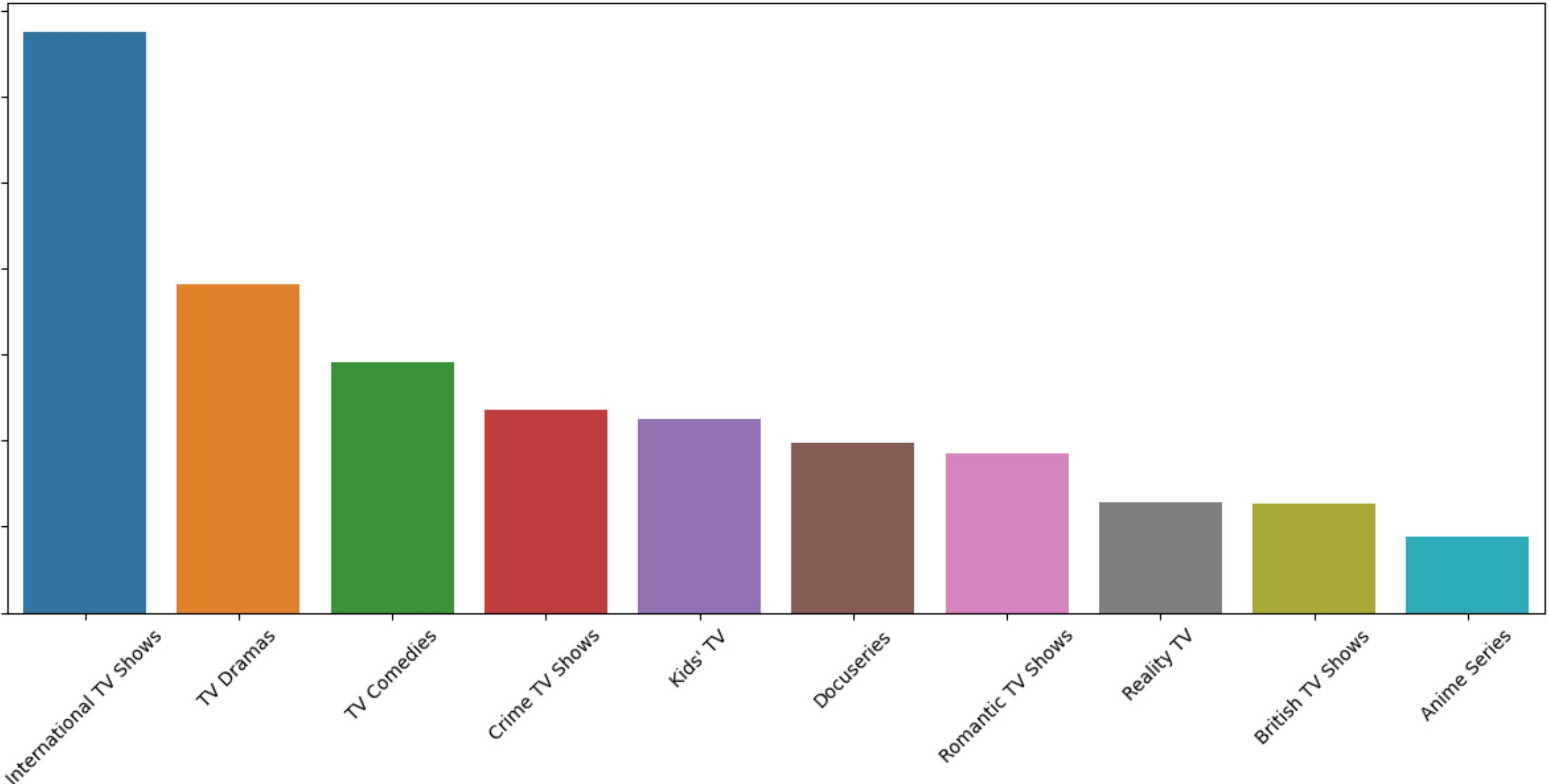
0u

600

400

200

0

listed\_in

##### Question - Top Genre of Movie

In [59]: df\_movie = df.loc[df['type']=='Movie', :]

df\_movie\_genre = df\_movie[['listed\_in', 'title']].drop\_duplicates()

d = df\_movie\_genre.groupby('listed\_in').count().reset\_index().sort\_values(by = 'title', ascending= **False).head(10)**

d.columns= ['listed\_in', 'Count']

d

|  |  |  |  |
| --- | --- | --- | --- |
| 0ut[59]: |  | **listed\_in** | **Count** |
|  | **11** | International Movies | 2752 |
|  | **7** | Dramas | 2427 |

|  |  |  |
| --- | --- | --- |
|  | **listed\_in** | **Count** |
| **4** | Comedies | 1674 |
| **6** | Documentaries | 869 |
| **0** | Action & Adventure | 859 |
| **10** | Independent Movies | 756 |
| **2** | Children & Family Movies | 641 |
| **15** | Romantic Movies | 616 |
| **10** | Th,;11 ,r | r:.77 |

In [60]: sns.barplot(data = d, x = d['listed\_in'], y = d['Count']) plt.xticks(rotation = 45)

plt.title('Genres with most number of Movies Releases') plt.ylabel('Count of Movies')

plt.show()

**VI**

**·QsJ:**

**0**

.**:**..**E**..

**0**..,

**C:**

:,

**0u**

2500

**2000**

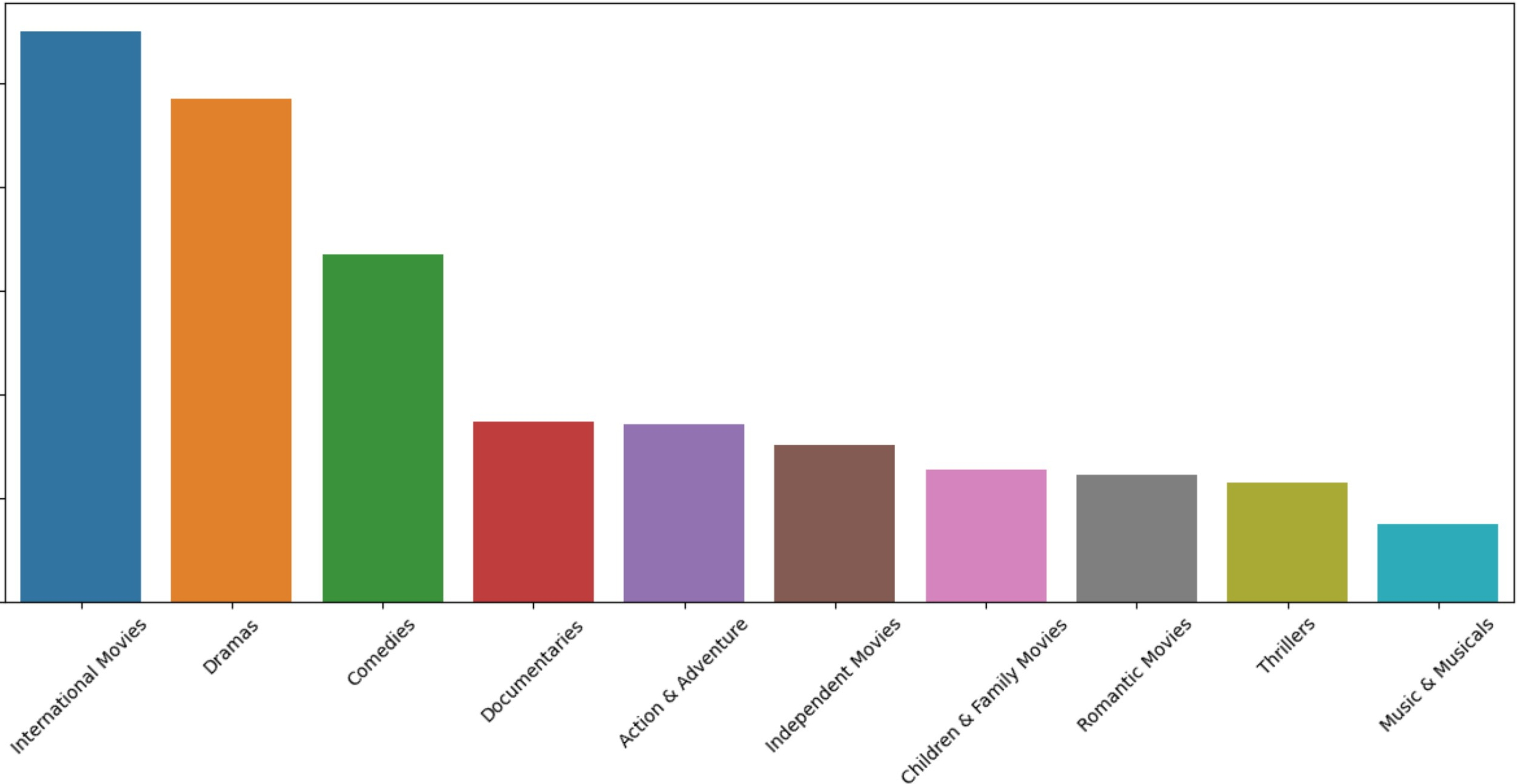
**1500**

**1000**

**500**

**0**

Genres with most number of Movies Releases



**listed\_in**

## Insights

* International TV Shows and TV Dramas are the top Genres of TV Shows
* International Movies and Dramas are the top Genres of Movies

##### Question - Most popular genre (Overall)?

In [61]: df\_genre = df[['title', 'listed\_in']].drop\_duplicates()

d = df\_genre.groupby('listed\_in').count().reset\_index().sort\_values('title', ascending= **False).head(10)**

d.columns=['listed\_in', 'Count']

d

|  |  |  |  |
| --- | --- | --- | --- |
| Out[61]: |  | **listed\_in** | **Count** |
|  | **16** | International Movies | 2752 |
|  | **12** | Dramas | 2427 |
|  | **7** | Comedies | 1674 |
|  | **17** | International TV Shows | 1351 |
|  | **10** | Documentaries | 869 |
|  | **0** | Action & Adventure | 859 |
|  | **34** | TV Dramas | 763 |
|  | **15** | Independent Movies | 756 |
|  | **4** | Children & Family Movies | 641 |
|  | **24** | Romantic Movies | 616 |

##### Question - Most popular genre in India?

In [62]: df\_india = df[df['country']=='India']

df\_india\_genre = df \_india [['title', 'listed\_in']]. drop\_duplicates ()

d = df\_india\_genre.groupby('listed\_in').count().reset\_index().sort\_values('title', ascending= **False).head(10)**

d. columns = [ 'listed\_in', 'count' ]

d

|  |  |  |  |
| --- | --- | --- | --- |
| Out[62]: |  | **listed\_in** | **count** |
|  | **13** | International Movies | 864 |
|  | **9** | Dramas | 662 |
|  | **4** | Comedies | 323 |
|  | **12** | Independent Movies | 167 |
|  | **0** | Action & Adventure | 137 |
|  | **19** | Romantic Movies | 120 |
|  | **17** | Music & Musicals | 96 |

**listed\_in count**

**34** Thrillers 92

**14** International TV Shows 66

Insights -

* Most popular genres across the whole Netflix Platform - 'International Movies', 'Dramas', 'Comedies', 'International TV Shows', 'Documentaries', 'Action & Adventure'

###### Business Insights -

Most of the Movies and Shows are from International Movies, International TV Shows and Dramas Genres. We can infer that Netflix has good number of viewers who watch International shows, and they are not very specific to their Regional Shows.

###### Question - Top 2 Actors who worked the most in Popular Genre

In [78]: df\_genre = df[df['listed\_in'].isin(['International Movies', 'Dramas', 'Comedies', 'International TV Shows', 'Documenta

**def** popular\_cast(df):

**return** df[['title', 'cast']].drop\_duplicates().groupby('cast').count().sort\_values('title', ascending= **False).hea**

d = df\_genre.groupby('listed\_in').apply(popular\_cast)

|  |  |  |
| --- | --- | --- |
|  | d.columns = ['Count'] |  |
| d |
| Out[78]: |  | **Count** |
|  | **listed\_in cast** |  |
|  | **Action & Adventure Bruce Willis** | 13 |
|  | **Amitabh Bachchan** | 12 |
|  | **Comedies Anupam Kher** | 20 |
|  | **Paresh Rawal** | 18 |
|  | **Documentaries Unknown** | 424 |
|  | **Samuel West** | 10 |

**Count**

**listed\_in Dramas**

**International Movies**

**cast**

**Anupam Kher** 28

**Shah Rukh Khan** 28

**Unknown** 178

**Anupam Kher** 38

In [68]:

##### Question - Top 3 Genres in popular Countries

popular\_countries = df[['title', 'country']].drop\_duplicates()['country'].value\_counts().head(10).reset\_index()['index popular\_countries

Out[68]: 0 United States

1. India
2. Unknown
3. United Kingdom
4. Canada
5. France
6. Japan
7. Spain
8. South Korea
9. Germany

Name: index, dtype: object

In [80]:

df\_popular\_countries = df[df['country'].isin(popular\_countries)]

**def** popular\_genre(df):

**return** df[['title', 'listed\_in']].drop\_duplicates().groupby('listed\_in').count().sort\_values('title', ascending=

d = df\_popular\_countries.groupby('country').apply(popular\_genre) d.columns = ['Count']

d

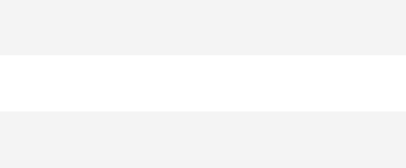
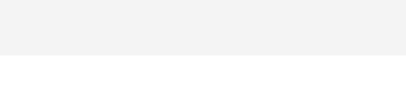
Out[80]: **Count**

**country listed\_in**

**Canada Comedies** 94

Netflix Business Case Study





**Count**

**country United States**

**listed\_in**

**Dramas** 835

##### Insights -

* International Movies and Dramas are top 2 in most of the countries
* Comedies is also one of the Popular Genres.
* In countries like Japan - Anime Series and in South Korea - Korean TV Shows are popular genres.

##### Question - What are top genres in different years?

In [85]: df\_recent\_years = df[df['release\_year'].isin([2021, 2020, 2019, 2018, 2017, 2016, 2015, 2014, 2013, 2012, 2011, 2010])

**def** popular\_rated(df):

**return** df[['title', 'listed\_in']].drop\_duplicates().groupby('listed\_in').count().reset\_index().sort\_values('title' d = df\_recent\_years.groupby('release\_year').apply(popular\_rated).sort\_values('release\_year', ascending= ***False)#.drop(***

d.columns = ['listed\_in', 'count'] d

|  |  |  |  |
| --- | --- | --- | --- |
| Out[85]: | **release\_year** | **listed\_in** | **count** |
|  | **2021 13** | International Movies | 141 |
|  | **14** | International TV Shows | 149 |
|  | **2020 15** | International TV Shows | 214 |
|  | **14** | International Movies | 239 |
|  | **2019 10** | Dramas | 243 |
|  | **14** | International Movies | 282 |
|  | **2018 12** | Dramas | 304 |
|  | **16** | International Movies | 340 |

|  |  |  |  |
| --- | --- | --- | --- |
| **release\_year** |  | **listed\_in** | **count** |
| **2017** | **11** | Dramas | 285 |
|  | **15** | International Movies | 328 |
| **2016** | **11** | Dramas | 265 |
|  | **15** | International Movies | 305 |
| **2015** | **10** | Dramas | 180 |
|  | **14** | International Movies | 210 |
| **2014** | **9** | Dramas | 104 |
|  | **13** | International Movies | 127 |
| **2013** | **11** | Dramas | 83 |
|  | **15** | International Movies | 121 |
| **2012** | **10** | Dramas | 66 |
|  | **14** | International Movies | 80 |
| **2011** | **14** | International Movies | 55 |
|  | **10** | Dramas | 60 |
| Insights |  |  |  |

* International Movies and Dramas are the most Popular genre until 2019
* From 2020 International TV Shows became one of the popular genre

##### Question - Most popular actor-director pair for movies across India?

In [86]: df\_india = df[df['country']=='India']

df\_cast\_director = df\_india[['title','cast', 'director']].drop\_duplicates()

df\_cast\_director.groupby(['cast', 'director']).count().reset\_index().sort\_values('title', ascending= **False).head(10)**

Out[86]: **cast director title**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **cast** | **director** | **title** |
| **7531** | Unknown | Unknown | 18 |
| **817** | Anupam Kher | David Dhawan | 6 |
| **5912** | Salman Khan | Sooraj R. Barjatya | 5 |
| **402** | Alok Nath | Sooraj R. Barjatya | 5 |
| **2811** | Julie Tejwani | Rajiv Chilaka | 4 |
| **5327** | Rajpal Yadav | Priyadarshan | 4 |
| **259** | Ajay Devgn | Prakash Jha | 4 |
| **3851** | Mithun Chakraborty | Umesh Mehra | 4 |

**'I r-,,,1**

The most popular pair is - Anupam Kher and David Dhawan

##### Question - Most of the movies are Rated as ?

In [87]: df\_rating = df[ ['rating', 'title']]. drop\_duplicates () df\_rating.groupby('rating').count().sort\_values('title', **ascending=False).head(10)**

|  |  |  |
| --- | --- | --- |
| Out[87]: | **rating** | **title** |
|  | **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 |

**title**

##### Question - What are most of the movies rated as in top countries ?

In [88]: df\_popular\_countries = df[df['country'].isin(popular\_countries)]

**def** popular\_rated(df):

**return** df[['title', 'rating']].drop\_duplicates().groupby('rating').count().reset\_index().sort\_values('title', asce df\_popular\_countries.groupby('country').apply(popular\_rated)

|  |  |  |  |
| --- | --- | --- | --- |
| 0ut[88]: | **country** | **rating** | **title** |
|  | **Canada** | **8** TV-MA | 107 |
|  |  | **5** R | 79 |
|  |  | **6** TV-14 | 49 |
|  | **France** | **8** TV-MA | 163 |
|  |  | **5** R | 57 |
|  |  | **6** TV-14 | 48 |
|  | **Germany** | **7** TV-MA | 79 |
|  |  | **4** R | 43 |
|  |  | **3** PG-13 | 31 |
|  | **India** | **4** TV-14 | 572 |
|  |  | **6** TV-MA | 266 |
|  |  | **7** TV-PG | 144 |
|  | **Japan** | **6** TV-MA | 101 |
|  |  | **4** TV-14 | 99 |
|  |  | **7** TV-PG | 50 |
|  | **South Korea** | **7** TV-MA | 92 |

|  |  |  |  |
| --- | --- | --- | --- |
| **country** |  | **rating** | **title** |
|  | **5** | TV-14 | 86 |
|  | **8** | TV-PG | 19 |
| **Spain** | **8** | TV-MA | 170 |
|  | **6** | TV-14 | 18 |
|  | **5** | R | 13 |
| **United Kingdom** | **7** | TV-MA | 253 |
|  | **4** | R | 145 |
|  | **5** | TV-14 | 103 |
| **United States** | **11** | TV-MA | 1101 |
|  | **8** | R | 660 |
|  | **9** | TV-14 | 497 |
| **Unknown** | **5** | TV-MA | 281 |

#### Insights

* If we consider on a whole, most of the Movies are rated as TV MA. This is for meant for Matured Audience (17 + age group).
* Other popular Category is TV-14 which ages under 14.

#### Business Insights

* We have most the shows for age group 17+ and under 14
* We can consider these two groups as main Target audience and make more relevant content.