In [3]:
```python
import numpy as np #for algebra
import pandas as pd #for data preparation
import plotly.express as px #for data visualisation
!pip install textblob
from textblob import TextBlob #for sentiment analysis
```

```
Collecting textblob
  Downloading textblob-0.17.1-py2.py3-none-any.whl (636 kB)
     -------------------------------------- 636.8/636.8 kB 2.9 MB/s eta 0:00:00
Requirement already satisfied: nltk>=3.1 in c:\users\windows\anaconda3\lib\site-packages
(from textblob) (3.7)
Requirement already satisfied: click in c:\users\windows\anaconda3\lib\site-packages (fr
om nltk>=3.1->textblob) (8.0.4)
Requirement already satisfied: joblib in c:\users\windows\anaconda3\lib\site-packages (f
rom nltk>=3.1->textblob) (1.1.1)
Requirement already satisfied: tqdm in c:\users\windows\anaconda3\lib\site-packages (fro
m nltk>=3.1->textblob) (4.64.1)
Requirement already satisfied: regex>=2021.8.3 in c:\users\windows\anaconda3\lib\site-pa
ckages (from nltk>=3.1->textblob) (2022.7.9)
Requirement already satisfied: colorama in c:\users\windows\anaconda3\lib\site-packages
(from click->nltk>=3.1->textblob) (0.4.6)
Installing collected packages: textblob
Successfully installed textblob-0.17.1
```

In [5]:
```python
df=pd.read_csv('C:/Users/windows/Downloads/archive/netflix_titles.csv')
```

In [29]:
```python
df.head()
```

Out[29]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	list
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG-13	90 min	Documen
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	2021	TV-MA	2 Seasons	Interna TV Shov Drama Mys
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	September 24, 2021	2021	TV-MA	1 Season	Crir S Interna TV Shov
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	NaN	September 24, 2021	2021	TV-MA	1 Season	Docus Real
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar, Ranjan	India	September 24, 2021	2021	TV-MA	2 Seasons	Interna TV S Roman Shows,

In [30]:
```
df.shape
```
Out[30]:
```
(8807, 12)
```

In [17]:
```
df.dtypes #check data types
```
Out[17]:
```
show_id         object
type            object
title           object
director        object
cast            object
country         object
date_added      object
release_year     int64
rating          object
duration        object
listed_in       object
description     object
dtype: object
```

In [22]:
```
pd.unique(df.duplicated(['title'])) #check if there is any duplicates
```
Out[22]:
```
array([False])
```

In [32]:
```
df['title'].isnull().values.any() #check if there is null values in the titles
```
Out[32]:
```
False
```

In [46]:
```
pd.unique(df['rating'])
```
Out[46]:
```
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)
```

In [49]:
```
df.drop(df[df['rating'] == '74 min'].index, inplace = True) #to delete the rows where th
df.drop(df[df['rating'] == '84 min'].index, inplace = True)
df.drop(df[df['rating'] == '66 min'].index, inplace = True)
pd.unique(df['rating'])
```
Out[49]:
```
array(['PG-13', 'TV-MA', 'PG', 'TV-14', 'TV-PG', 'TV-Y', 'TV-Y7', 'R',
       'TV-G', 'G', 'NC-17', 'NR', nan, 'TV-Y7-FV', 'UR'], dtype=object)
```

In [38]:
```
pd.unique(df['type'])
```
Out[38]:
```
array(['Movie', 'TV Show'], dtype=object)
```

In [39]:
```
pd.unique(df['release_year'])
```
Out[39]:
```
array([2020, 2021, 1993, 2018, 1996, 1998, 1997, 2010, 2013, 2017, 1975,
       1978, 1983, 1987, 2012, 2001, 2014, 2002, 2003, 2004, 2011, 2008,
       2009, 2007, 2005, 2006, 1994, 2015, 2019, 2016, 1982, 1989, 1990,
       1991, 1999, 1986, 1992, 1984, 1980, 1961, 2000, 1995, 1985, 1976,
       1959, 1988, 1981, 1972, 1964, 1945, 1954, 1979, 1958, 1956, 1963,
       1970, 1973, 1925, 1974, 1960, 1966, 1971, 1962, 1969, 1977, 1967,
       1968, 1965, 1946, 1942, 1955, 1944, 1947, 1943], dtype=int64)
```

In [79]:
```
pd.unique(df['director'])
```
Out[79]:
```
array(['Kirsten Johnson', 'No Director Specified', 'Julien Leclercq', ...,
       'Majid Al Ansari', 'Peter Hewitt', 'Mozez Singh'], dtype=object)
```

To begin the task of analyzing Netflix data, I'll start by looking at the distribution of content ratings on Netflix:

In [50]:
```
z = df.groupby(['rating']).size().reset_index(name='counts')
print(z)
```

```
      rating  counts
0          G      41
1      NC-17       3
2         NR      80
3         PG     287
4      PG-13     490
5          R     799
6      TV-14    2160
7       TV-G     220
8      TV-MA    3207
9      TV-PG     863
10      TV-Y     307
11     TV-Y7     334
12  TV-Y7-FV       6
13        UR       3
```

In [57]:
```
pieChart = px.pie(z, values='counts', names='rating',
                  title='Distribution of Content Ratings on Netflix',
                  color_discrete_sequence=px.colors.qualitative.Set1)
pieChart.show()
```

The graph above shows that the majority of Netflix content is categorized as TV-MA, which means that most of the content available on Netflix is intended for viewing by mature and adult audiences.

Let's see the Top 5 succesful directors on Netflix:

In [58]:
```
df['director']=df['director'].fillna('No Director Specified')
```

In [60]:
```
df['director'].isnull().values.any()
```

Out[60]:
```
False
```

In [94]:
```
filtered_directors=pd.DataFrame()
filtered_directors=df['director'].str.split(',',expand=True).stack()
filtered_directors=filtered_directors.to_frame()
filtered_directors.columns=['Director']
directors=filtered_directors.groupby(['Director']).size().reset_index(name='Total Conten
directors=directors[directors.Director !='No Director Specified']
directors=directors.sort_values(by=['Total Content'],ascending=False)
directorsTop5=directors.head()
directorsTop5=directorsTop5.sort_values(by=['Total Content'])
fig1=px.bar(directorsTop5,x='Total Content',y='Director',title='Top 5 Directors on Netfl
fig1.show()
```

From the graph above we see that Rajiv Chilaka, Raul Campos, Jan Suter, Marcus Raboy and Suhas Kadav are the Top 5 directors in this plateform.

Now, let's look at the Top 5 actors on Netflix:

In [81]:
```
df['cast']=df['cast'].fillna('No Cast Specified')
```

In [82]:
```
df['cast'].isnull().values.any()
```

Out[82]:
```
False
```

In [88]:
```
Split_Actors=pd.DataFrame()
Split_Actors=df['cast'].str.split(',',expand=True).stack()
Split_Actors=Split_Actors.to_frame()
Split_Actors.columns=['Actor']
Actors=Split_Actors.groupby(['Actor']).size().reset_index(name='Total Shows')
Actors=Actors[Actors.Actor !='No Cast Specified']
Actors=Actors.sort_values(by=['Total Shows'],ascending=False)
ActorsTop5=Actors.head()
ActorsTop5=ActorsTop5.sort_values(by=['Total Shows'])
fig2=px.bar(ActorsTop5,x='Total Shows',y='Actor',title='Top 5 Actors on Netflix')
fig2.show()
```

From the graph above, the Top 5 Actors in Netflix are:

- Anupam Kher
- Rupa Bhimani
- Takahiro Sakurai
- Julie Tejwani
- Om Puri

Let's analyse the content of Netflix over the years:

In [95]:
```
df1=df[['type','release_year']] #create a new df with these two column
df1=df1.rename(columns={"release_year": "Release Year"}) #to rename the column
df2=df1.groupby(['Release Year','type']).size().reset_index(name='Total Content')
df2=df2[df2['Release Year']>=2010]
```

```
fig3 = px.line(df2, x="Release Year", y="Total Content", color='type',title='Trend of co
fig3.show()
```

The graph above shows that the production of both movies and other TV shows had declined in the last years.

Finally, Let's do a sentiment analysis on the content of Netflix:

In [99]:
```
dfx=df[['release_year','description']]
dfx=dfx.rename(columns={'release_year':'Release Year'})
for index,row in dfx.iterrows():
    z=row['description']
    testimonial=TextBlob(z)
    p=testimonial.sentiment.polarity
    if p==0:
        sent='Neutral'
    elif p>0:
        sent='Positive'
    else:
        sent='Negative'
    dfx.loc[[index,2],'Sentiment']=sent
dfx=dfx.groupby(['Release Year','Sentiment']).size().reset_index(name='Total Content')

dfx=dfx[dfx['Release Year']>=2010]
fig4 = px.bar(dfx, x="Release Year", y="Total Content", color="Sentiment", title="Sentim
fig4.show()
```

So the above graph shows that the overall positive content is always greater than the neutral and negative content combined.

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```