I'll start this Netflix data analysis task with Python by importing the dataset and all the Python libraries needed for this task:

In [2]:
```
import pandas as pd
import numpy as np
import plotly.express as px
from textblob import TextBlob

df = pd.read_csv(r'E:\Power-BI-Tutorial-Files\pavan powwerbi\projects\data analiyst project sql\netflix data analysis\netflix_titles.csv')

df.shape
```

Out[2]:
```
(8807, 12)
```

In [3]:
```
# data consists of 6234 rows and 12 columns
```

In [4]:
```
df
```

Out[4]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG-13	90 min	Documentaries	As her father nears the end of his life, filmm...
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	2021	TV-MA	2 Seasons	International TV Shows, TV Dramas, TV Mysteries	After crossing paths at a party, a Cape Town t...
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	September 24, 2021	2021	TV-MA	1 Season	Crime TV Shows, International TV Shows, TV Act...	To protect his family from a powerful drug lor...
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	NaN	September 24, 2021	2021	TV-MA	1 Season	Docuseries, Reality TV	Feuds, flirtations and toilet talk go down amo...
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	September 24, 2021	2021	TV-MA	2 Seasons	International TV Shows, Romantic TV Shows, TV ...	In a city of coaching centers known to train I...
...	...	...	...	...	...	...	...	...	...	...	...	...
8802	s8803	Movie	Zodiac	David Fincher	Mark Ruffalo, Jake Gyllenhaal, Robert Downey J...	United States	November 20, 2019	2007	R	158 min	Cult Movies, Dramas, Thrillers	A political cartoonist, a crime reporter and a...
8803	s8804	TV Show	Zombie Dumb	NaN	NaN	NaN	July 1, 2019	2018	TV-Y7	2 Seasons	Kids' TV, Korean TV Shows, TV Comedies	While living alone in a spooky town, a young g...
8804	s8805	Movie	Zombieland	Ruben Fleischer	Jesse Eisenberg, Woody Harrelson, Emma Stone, ...	United States	November 1, 2019	2009	R	88 min	Comedies, Horror Movies	Looking to survive in a world taken over by zo...
8805	s8806	Movie	Zoom	Peter Hewitt	Tim Allen, Courteney Cox, Chevy Chase, Kate Ma...	United States	January 11, 2020	2006	PG	88 min	Children & Family Movies, Comedies	Dragged from civilian life, a former superhero...
8806	s8807	Movie	Zubaan	Mozez Singh	Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanan...	India	March 2, 2019	2015	TV-14	111 min	Dramas, International Movies, Music & Musicals	A scrappy but poor boy worms his way into a ty...

8807 rows × 12 columns

In [5]:
```
# now let's look at the column names:
```

In [6]:
```
df.columns
```

Out[6]:
```
Index(['show_id', 'type', 'title', 'director', 'cast', 'country', 'date_added',
       'release_year', 'rating', 'duration', 'listed_in', 'description'],
      dtype='object')
```

In [7]:
```
pip install textblob
```

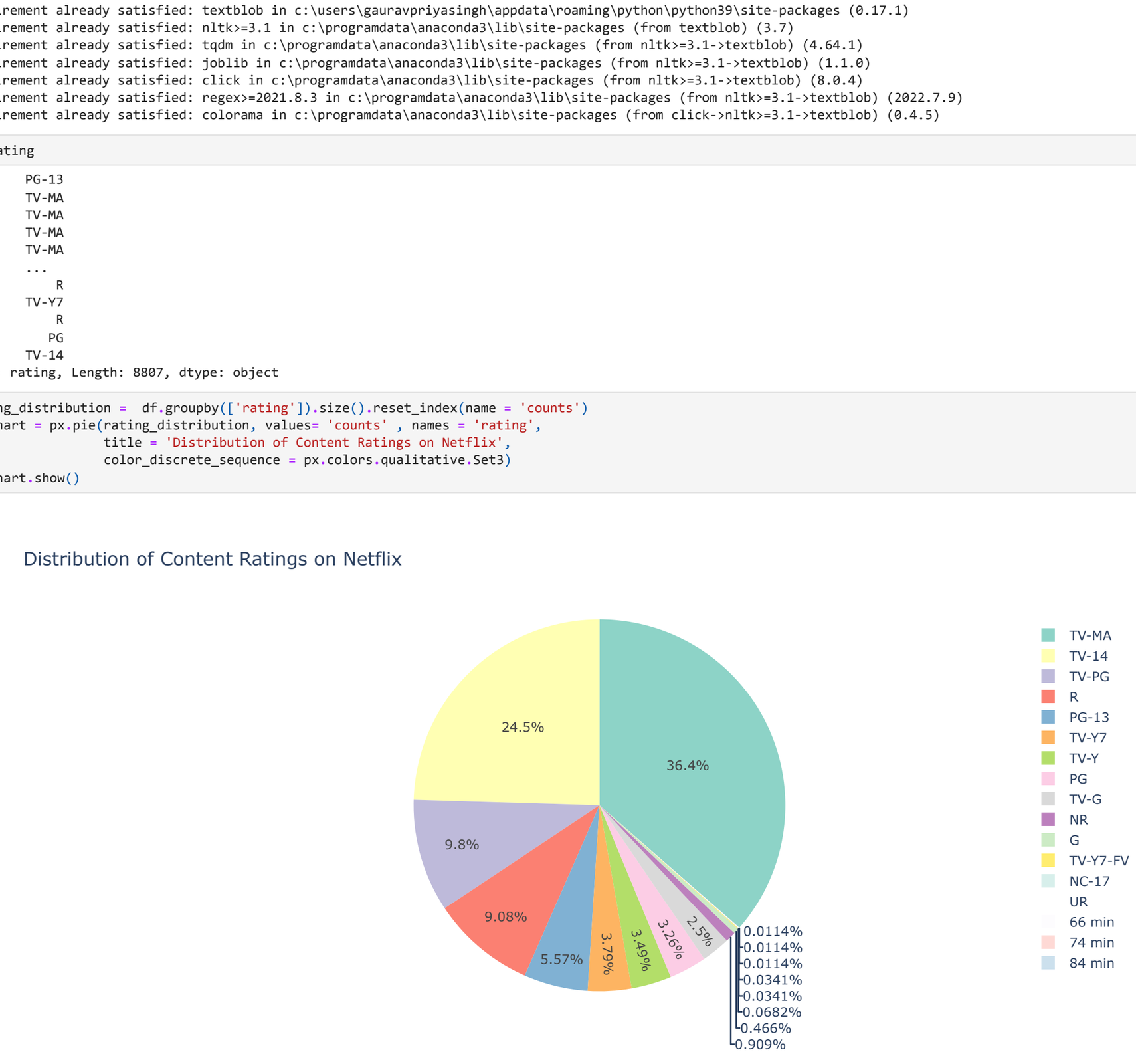
```
Defaulting to user installation because normal site-packages is not writeableNote: you may need to restart the kernel to use updated packages.

Requirement already satisfied: textblob in c:\users\gauravpriyasingh\appdata\roaming\python\python39\site-packages (0.17.1)
Requirement already satisfied: nltk>=3.1 in c:\programdata\anaconda3\lib\site-packages (from textblob) (3.7)
Requirement already satisfied: tqdm in c:\programdata\anaconda3\lib\site-packages (from nltk>=3.1->textblob) (4.64.1)
Requirement already satisfied: joblib in c:\programdata\anaconda3\lib\site-packages (from nltk>=3.1->textblob) (1.1.0)
Requirement already satisfied: click in c:\programdata\anaconda3\lib\site-packages (from nltk>=3.1->textblob) (8.0.4)
Requirement already satisfied: regex>=2021.8.3 in c:\programdata\anaconda3\lib\site-packages (from nltk>=3.1->textblob) (2022.7.9)
Requirement already satisfied: colorama in c:\programdata\anaconda3\lib\site-packages (from click->nltk>=3.1->textblob) (0.4.5)
```

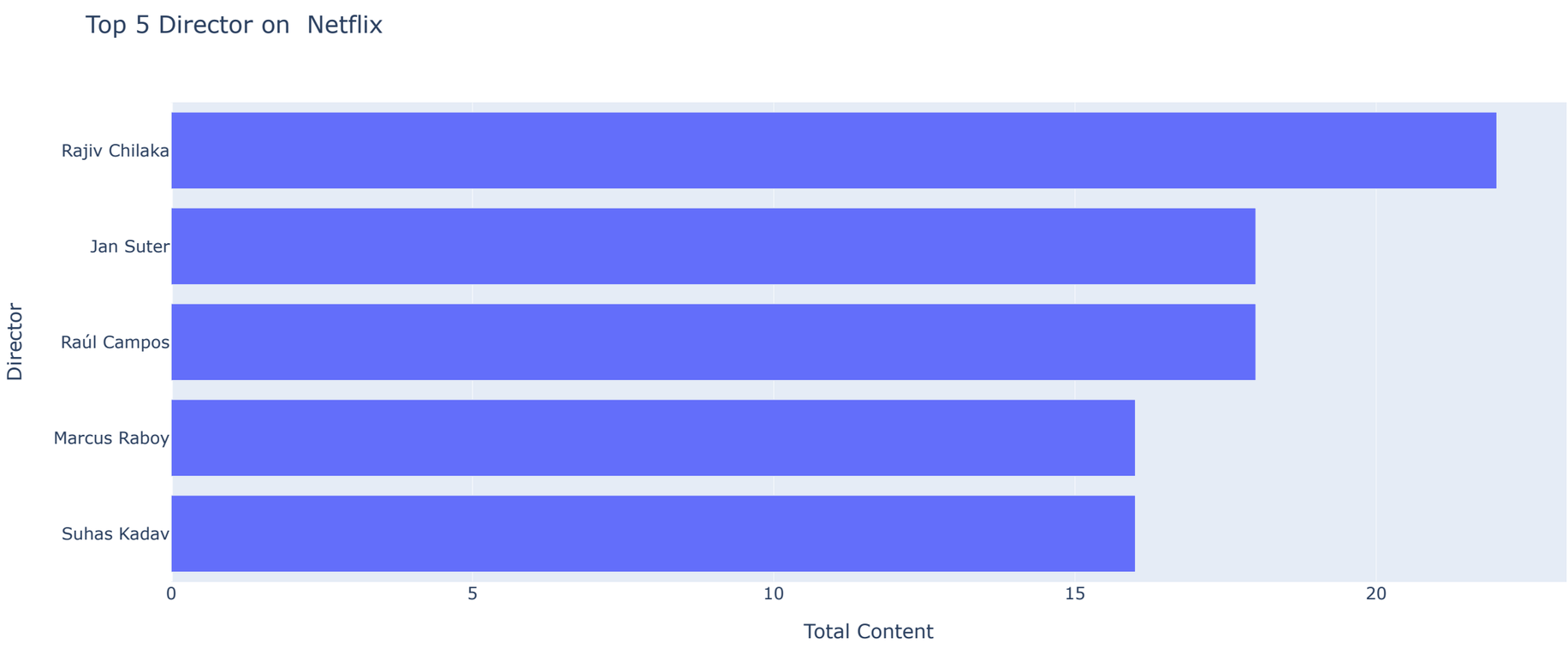
In [3]:
```
df.rating
```

Out[3]:
```
0       PG-13
1       TV-MA
2       TV-MA
3       TV-MA
4       TV-MA
        ...
8802        R
8803    TV-Y7
8804        R
8805       PG
8806    TV-14
Name: rating, Length: 8807, dtype: object
```

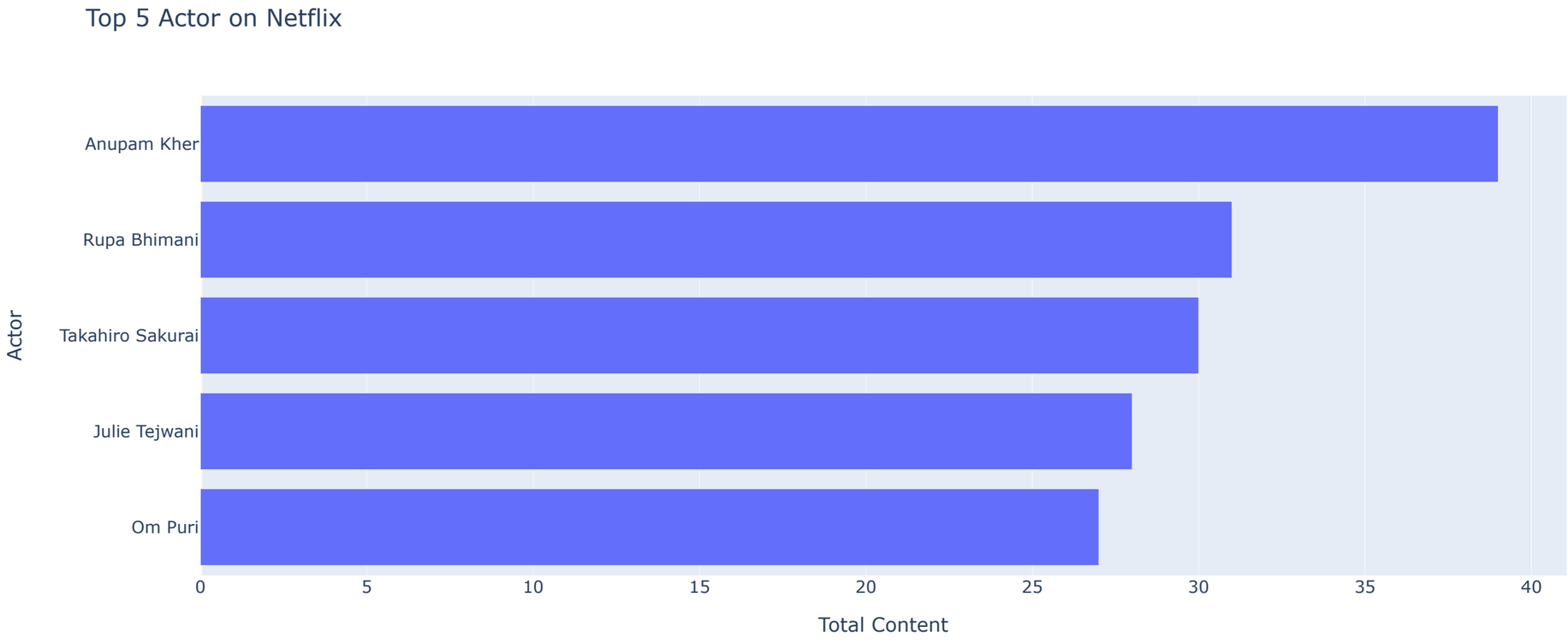
In [9]:
```
rating_distribution =  df.groupby(['rating']).size().reset_index(name = 'counts')
pieChart = px.pie(rating_distribution, values= 'counts' , names = 'rating',
                  title = 'Distribution of Content Ratings on Netflix',
                  color_discrete_sequence = px.colors.qualitative.Set3)
pieChart.show()
```



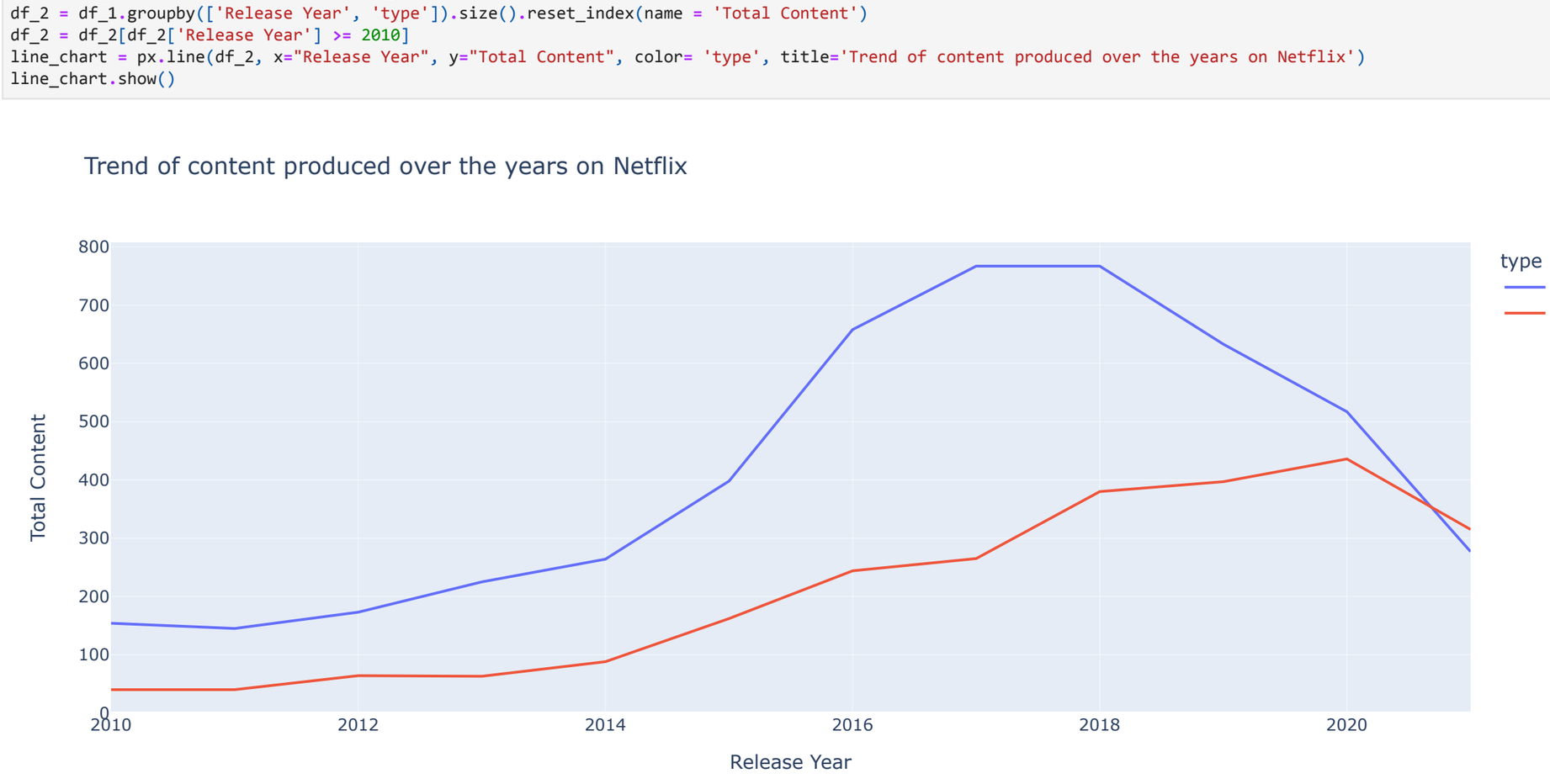
In [10]:
```
df['director']=df['director'].fillna('No Director Specified')
filtered_directors=pd.DataFrame()
filtered_directors = df['director'].str.split(',', expand=True).stack()
filtered_directors = filtered_directors.to_frame()
filtered_directors.columns = ['Director']
filtered_directors
directors = filtered_directors.groupby(['Director']).size().reset_index(name = 'Total Content')
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'])
bar_chart = px.bar(directorsTop5, x = 'Total Content', y='Director', title = 'Top 5 Director on  Netflix ')
bar_chart.show()
```



In [11]:
```
df['cast'] = df['cast'].fillna('No cast Specified')
filtered_cast=pd.DataFrame()
filtered_cast = df['cast'].str.split(',' , expand=True).stack()
filtered_cast = filtered_cast.to_frame()
filtered_cast.columns= ['Actor']
Actors = filtered_cast.groupby(['Actor']).size().reset_index(name = 'Total Content')
Actors = Actors[Actors.Actor != 'No cast Specified']
Actors = Actors.sort_values(by = ['Total Content'] , ascending = False)
ActorsTop5 = Actors.head()
ActorsTop5 = ActorsTop5.sort_values(by = ['Total Content'])
bar_chart = px.bar(ActorsTop5, x = 'Total Content' , y = "Actor" , title= 'Top 5 Actor on Netflix ')
bar_chart.show()
```



In [12]:
```
df_1 = df[['type', 'release_year']]
df_1 = df_1.rename(columns = {'release_year':'Release Year'})
df_2 = df_1.groupby(['Release Year', 'type']).size().reset_index(name = 'Total Content')
df_2 = df_2[df_2['Release Year'] >= 2010]
line_chart = px.line(df_2, x="Release Year", y="Total Content", color= 'type', title='Trend of content produced over the years on Netflix')
line_chart.show()
```



In [13]:
```
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="Sentiment of content on Netflix")
fig4.show()
```