Business Case Study: NETFLIX Python

Details: Name : Tanmoy Basuli

Email-Id : basuli575tanmoy@gmail.com

Introduction

In this study-case, I'll give an Exploratory Data Analysis of the Netflix dataset. I will explore the data and hopefully bring some insights.

- We will try to find some insights on below,
  - the distribution of genres,
  - the distribution of countries,
  - the distribution of ratings, and
  - the distribution of duration
- We will see yearly and monthly contents and how many are those.
- We will understand insides of contents by examining content description using WordCloud

For visualizations I used, seaborn, pyplot, plotly, wordCloud and missingno. Some of the visuals interactive, and some of it static. But there's a lot improve. Feedback is welcome.

Dataset: https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/000/940/original/netflix.csv

The Outline of this notebook is as follows.

1. Basic Data Exploration
   - Feature Exploration
   - Summary Statistics
2. Data Cleaning
   - Null Value Analysis
   - Checking Duplicate Values
   - Handling inconsistent or incorrect data
3. Exploratory data analysis (What is the Story Of Data)

Importing Libraries and Loading the Dataset

```
# Import Relevant Packages
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
import plotly.graph_objects as go
import missingno as msno
from wordcloud import WordCloud
import plotly.figure_factory as ff
```

```
# Load Data set
```

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

Basic Data Exploration

1. Feature Exploration
2. Summary Statistics

1. Feature Exploration

First, let us look at a quick peek of what the first five rows in the data has in store for us and what features we have.

```
# First five rows of the data
df.head()
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	descri
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	25-Sep-21	2020	PG-13	90 min	Documentaries	As her nea end life, fi
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	24-Sep-21	2021	TV-MA	2 Seasons	International TV Shows, TV Dramas, TV Mysteries	cr patl party, a To
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	24-Sep-21	2021	TV-MA	1 Season	Crime TV Shows, International TV Shows, TV Act...	To prot family po dru

```
#Look what all are the columns I have
df.columns
```

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

In this dataset we have,

This dataset contains data collected from Netflix of different TV shows and movies.

type: Can collect 2 different unique values. One is TV Show and another is Movie

title: Gives information about the title of Movie or TV Show

director: It is about the director who directed the Movie or TV Show

cast: It's about the actors who plays role in Movie or TV Show

release_year: It's about the year when Movie or TV Show was released

rating: It's about the Movie or TV Show are in which category (eg like the movies are only for students, or adults, etc)

duration: Gives information about the duration of Movie or TV Show

listed_in: Gives information about the genre of Movie or TV Show

description: Gives information about the description of Movie or TV Show

Next, let us look at how large the data is:

```
#Size of the data
df.shape
```

(8807, 12)

We have 8807 Entity from 12 Features.

Now, Let's look What types of data we have:

```
# Type of all Data
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   show_id       8807 non-null   object
 1   type          8807 non-null   object
 2   title         8807 non-null   object
 3   director      6173 non-null   object
 4   cast          7982 non-null   object
 5   country       7976 non-null   object
 6   date_added    8797 non-null   object
 7   release_year  8807 non-null   int64 
 8   rating        8803 non-null   object
 9   duration      8804 non-null   object
 10  listed_in     8807 non-null   object
```

```
 11  description   8807 non-null   object
dtypes: int64(1), object(11)
memory usage: 825.8+ KB
```

We can easily find out from the above showcase, we have:

- 11 Categorical Feature
- 1 Numeric Feature

2. Summary Statistics(Non-Graphical Analysis: Value counts and unique attributes)

Here we can see basic statistics in the data.

```
# Summary statistics for numerical features
numerical_features = df.select_dtypes(include='number')
# We have only 'release_year' as a numeric feature
numerical_features.describe().T
```

	count	mean	std	min	25%	50%	75%	max
release_year	8807.0	2014.180198	8.819312	1925.0	2013.0	2017.0	2019.0	2021.0

Content release year analysis looks like this;

- We have content that has been the released year from 1925 to 2021.
- We have the mean year 2014.
- We have a standard deviation of ~ 8.82 and this can show us like we have release_year data spreads from 1925 to 2021, probably we have outliers mostly from 1925 to 2014.

Now we will look into categorical data in our dataset.

```
# Summary statistics for categorical features
categorical_features = df.select_dtypes(include='object')
categorical_features.describe().T
```

	count	unique	top	freq
show_id	8807	8807	s1	1
type	8807	2	Movie	6131
title	8807	8804	15-Aug	2
director	6173	4528	Rajiv Chilaka	19
cast	7982	7692	David Attenborough	19
country	7976	748	United States	2818
date_added	8797	1767	1-Jan-20	109
rating	8803	17	TV-MA	3207
duration	8804	220	1 Season	1793
listed_in	8807	514	Dramas, International Movies	362
description	8807	8775	Paranormal activity at a lush, abandoned prope...	4

We can see in this result: frequency of data, unique values, and most repeated data.

Data Cleaning

1. Null Value Analysis
2. Checking Duplicate Values
3. Handling inconsistent or incorrect data

1. Null Value Analysis

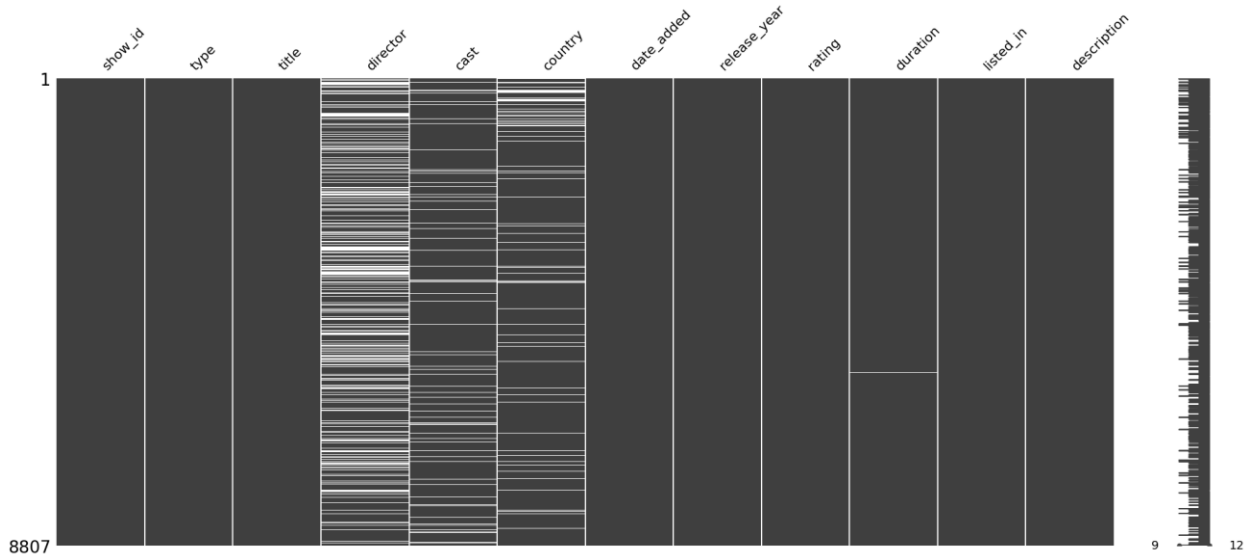
```
# Checking null value if we have
df.isnull().values.any()
```

True

We can see above, its showing true. That means definitely we have null values in our dataset.

Now need to find out where it is.

```
# Which features have how much null values?
msno.matrix(df)
```



We can see, the missing values now become much more apparent and clearer when we visualize it. White bands show us missing data values and dark gray ones are non-missing data.

- We're seeing that dataset a lot of missing director data,
- We have missing data on cast and country as well.
- We have some missing data on duration.

This visualization is good seeing big picture of missing values. But We need specific number and percentage of missing values. As well. Let see below how we can find it out.

```
#Checking the percentage of missing value of each columns

for i in df.columns:
  #get the percentage value
  val = df[i].isnull().sum()/df.shape[0] * 100
  # if value is positive then check how much it is
  if val > 0:
    print(i, ':',round(val,2))
```

```
director : 29.91
cast : 9.37
country : 9.44
date_added : 0.11
rating : 0.05
duration : 0.03
```

Now we need to remove Null values from the dataset.

First of all, I will change my date_added column to pandas date format. It will be easier to analyze and replace the null values.

```
# Changing the date format
df["date_added"] = pd.to_datetime(df['date_added'])

# Creating a new column for year
df['year_added'] = df['date_added'].dt.year

# Creating a new column for month
df['month_added'] = df['date_added'].dt.month
```

Now I will replace null values one by one.

```
# Replacing null value with below.
df['director'].fillna('No Director', inplace = True)
df['cast'].fillna('No Cast', inplace = True)
df['country'].fillna('No Country', inplace = True)
df['date_added'].fillna('Missing', inplace = True)
df['year_added'].fillna('No Year', inplace = True)
df['month_added'].fillna('No Month', inplace = True)
df['rating'].fillna('Missing', inplace = True)
df['duration'].fillna('No Duration', inplace = True)
```

```
# Controlling null values again
df.isnull().sum()
```

```
show_id         0
type            0
title           0
director        0
cast            0
country         0
date_added      0
release_year    0
rating          0
duration        0
listed_in       0
description     0
year_added      0
month_added     0
dtype: int64
```

Now we can see null values are eliminated.

2. Checking Duplicate Values

Checking for duplicating values before EDA always a good idea.

```
# Is there any duplicates
duplicated_rows = df[df.duplicated()]
print(f"Duplicates value number in dataset: {duplicated_rows.shape[0]}")
```

```
Duplicates value number in dataset: 0
```

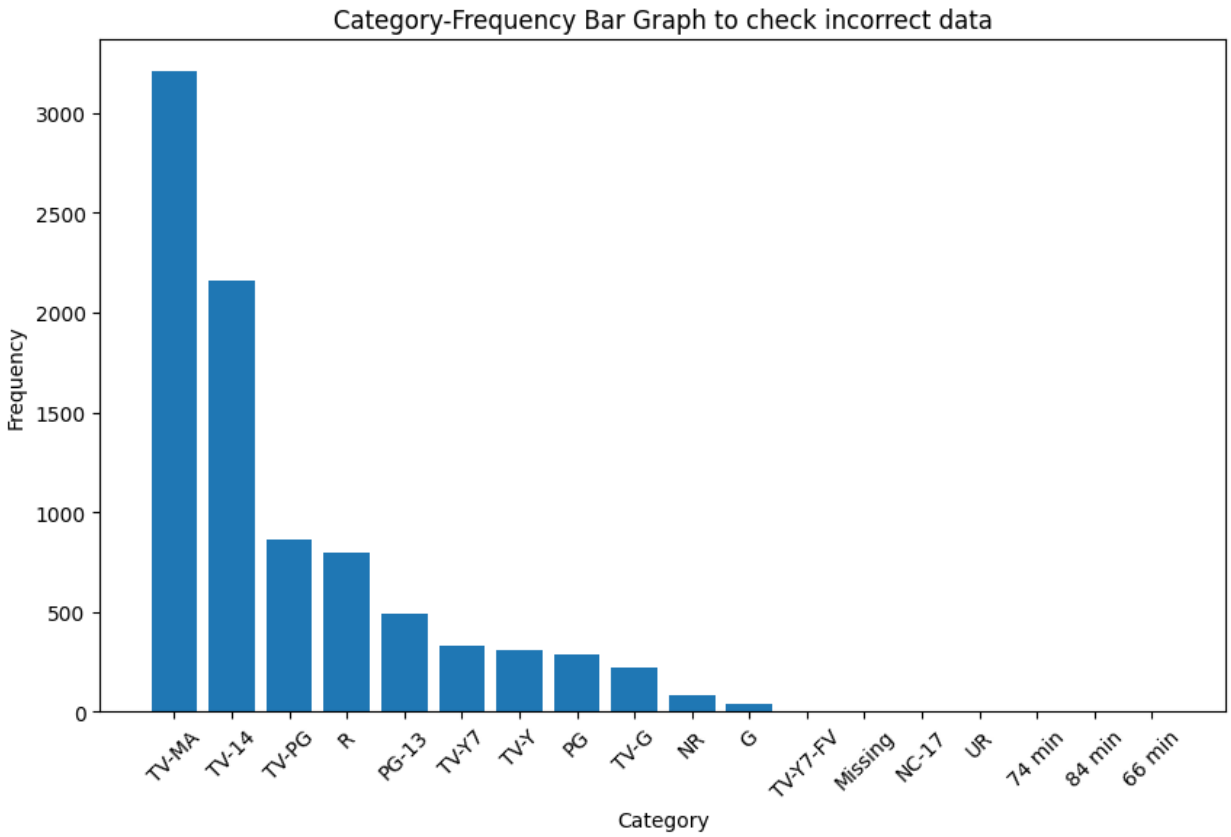
We can see there are no duplicates showing.

3. Handling inconsistent or incorrect data

We need to look for incorrect data now. There's a lot of unique data in dataset, this is normal for movie or tv shows data. But we can look rating data if it has incorrect data.

```
def plot_categorical_frequency_to_check_incorrect_value(data, x_label,
y_label, title):
    """Show Bar Frequency for categorical Features"""
    frequency_counts = data.value_counts()

    plt.figure(figsize=(12, 6))
    plt.bar(frequency_counts.index, frequency_counts.values)
    plt.xlabel(x_label)
    plt.ylabel(y_label)
    plt.title(title)
    plt.xticks(rotation=45)
    plt.show()
```

```
#incorrect data check
plot_categorical_frequency_to_check_incorrect_value(df['rating'],"Category
", "Frequency", "Category-Frequency Bar Graph to check incorrect data")
```



with 'rating' frequency analysis we see three unwanted values entered: '74 min', '84 min', '66 min'

- We can eliminate these with turning values to 'UR'; Because UR means, *Unrated*.
- And we can turn 'missing' values to 'UR' as well.

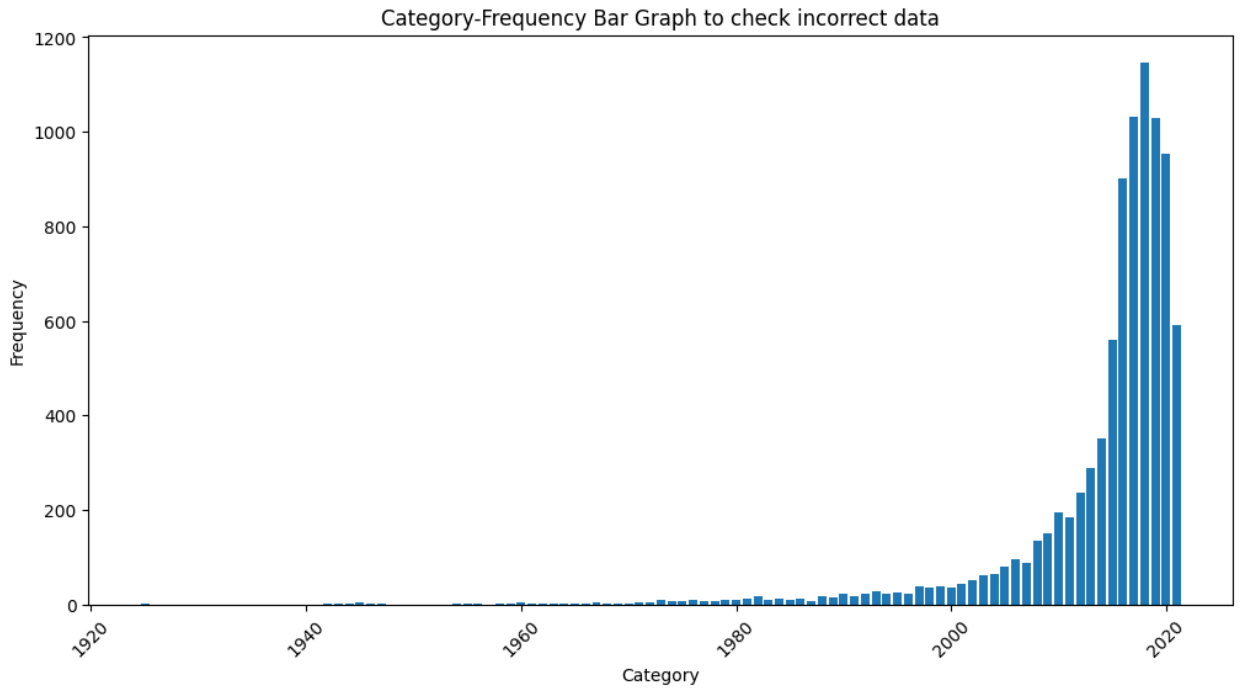
```
# Change '74 min', '84 min', '66 min' Values to 'UR'
df['rating'] = np.where(np.isin(df['rating'], ['74 min', '84 min', '66
min', 'missing']), 'UR', df['rating'])
```

We eliminated all unwanted values from rating now.

Let see one more column for the time being, is there any incorrect value present or not for the release year.

We can use above function only to check this.

```
#incorrect data check for release year
plot_categorical_frequency_to_check_incorrect_value(df['release_year'],"Ca
tegory", "Frequency", "Category-Frequency Bar Graph to check incorrect
data")
```



We can't see any abnormal data for the above graph.

Rest all of column has high unique value, so it won't possible to check by run this function and decide. Although there have solution but I am not presenting those here.

Now start the Visualization and gain some insight from that.

Data Visualization

1. Content Types

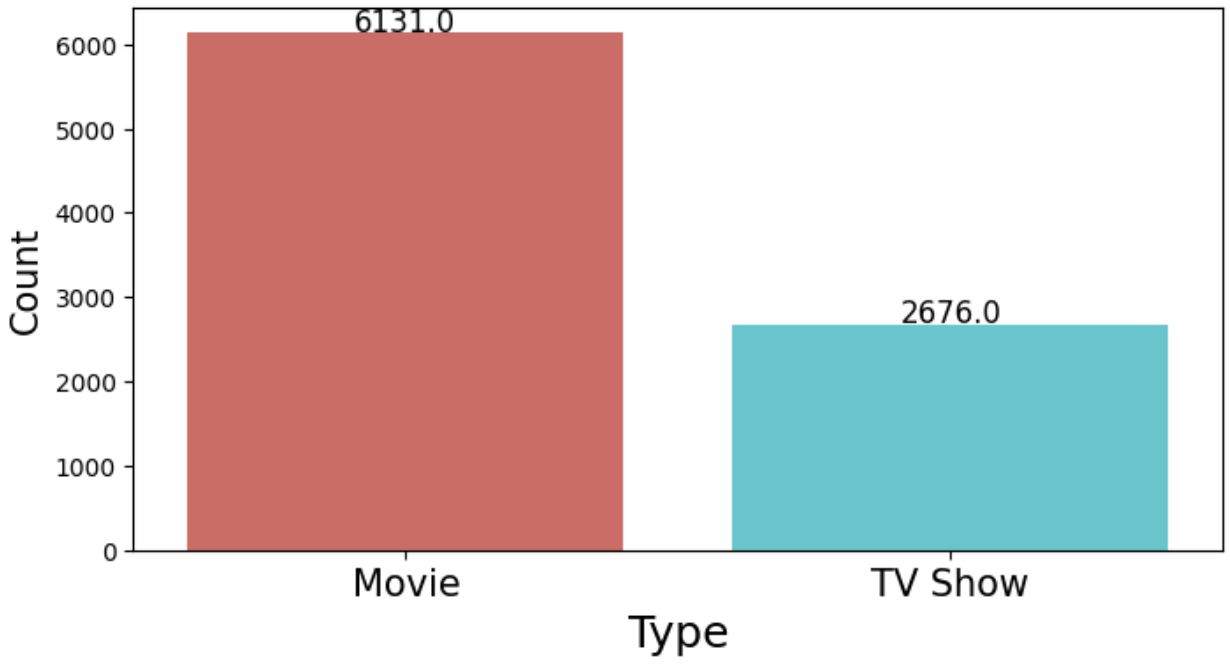
1. First, let's look at the percentage of movies and TV shows in the dataset.

```
# First will check the count of TV Show and Movie from the dataset
type_movie = df[df['type'] == 'Movie'].shape[0]
type_show = df[df['type'] == 'TV Show'].shape[0]
print('Count of Movie is :',type_movie, "and count of TV Show is :",
type_show)
```

```
Count of Movie is : 6131 and count of TV Show is : 2676
```

Now visualize this data using COUNTPLOT.

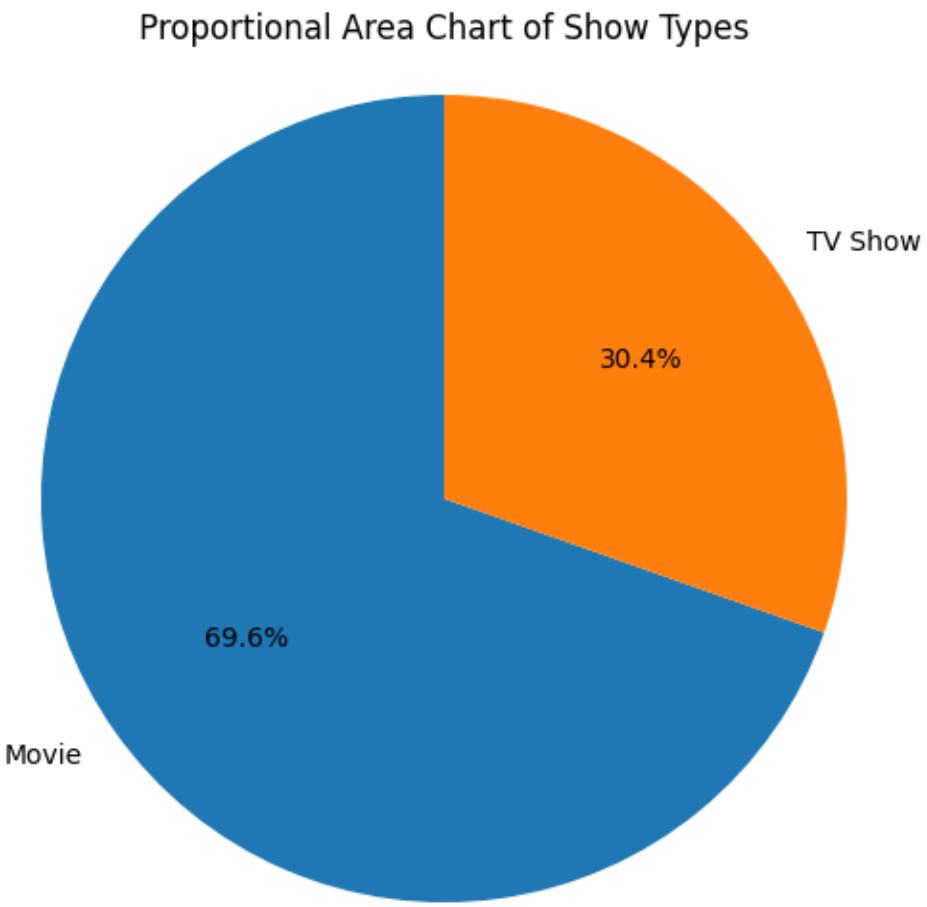
```
# Now visualize the count using countplot
plt.figure(figsize = (8,4), dpi = 100)
plot = sns.countplot(x = 'type', data = df, palette = 'hls')
plt.xticks(fontsize = 15)
plt.xlabel('Type',fontsize = 18 )
plt.ylabel('Count',fontsize = 15 )

for i in plot.patches:
  height = i.get_height()
  plot.text(i.get_x() + i.get_width()/2, height + 20, height, ha =
'center', fontsize=12)
plt.show()
```



Let's check the percentage of total content,

```
type_count = df['type'].value_counts()
total_count = type_count.sum()

# Calculate the proportion of that
percentage_content = type_count/total_count

# Plotting the proportional area chart
```

```
plt.figure(figsize=(8, 6))
plt.pie(percentage_content, labels=percentage_content.index,
autopct='%1.1f%%', startangle=90)
plt.axis('equal')
plt.title('Proportional Area Chart of Show Types')
plt.show()
```



Observation:

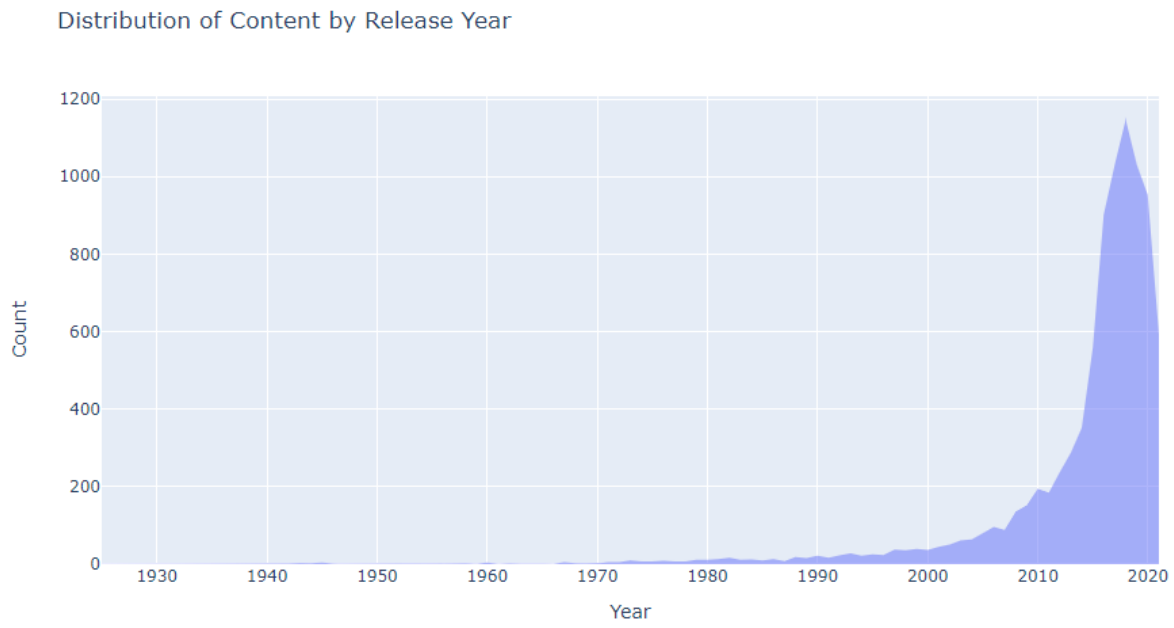
- We can see, audience love to watch movies. Most of the content tends towards movies with 69.6% as compare to TV Shows(30.4%)

2. Time Analysis

1. First, let's look at the distribution of content by **released years.**

```
# Count the occurrences of each year
year_counts = df['release_year'].value_counts().sort_index()

# Create a stacked area graph using Plotly
fig = go.Figure()

fig.add_trace(go.Scatter(
    x=year_counts.index,
    y=year_counts.values,
    mode='none',
    fill='tozeroy',
    hovertemplate='Year: %{x}<br>Count: %{y}<extra></extra>',
    name='Count'
))

# Customize the axes labels and title
fig.update_layout(
    xaxis=dict(title='Year'),
    yaxis=dict(title='Count'),
    title='Distribution of Content by Release Year'
)

# Show the stacked area graph
fig.show()
```

.



Observations

- Release year of contents are in mostly 2018 (1147), 2017 (1032), 2019 (1030)
- After 2010, we can see that movies or series are more concentrated.

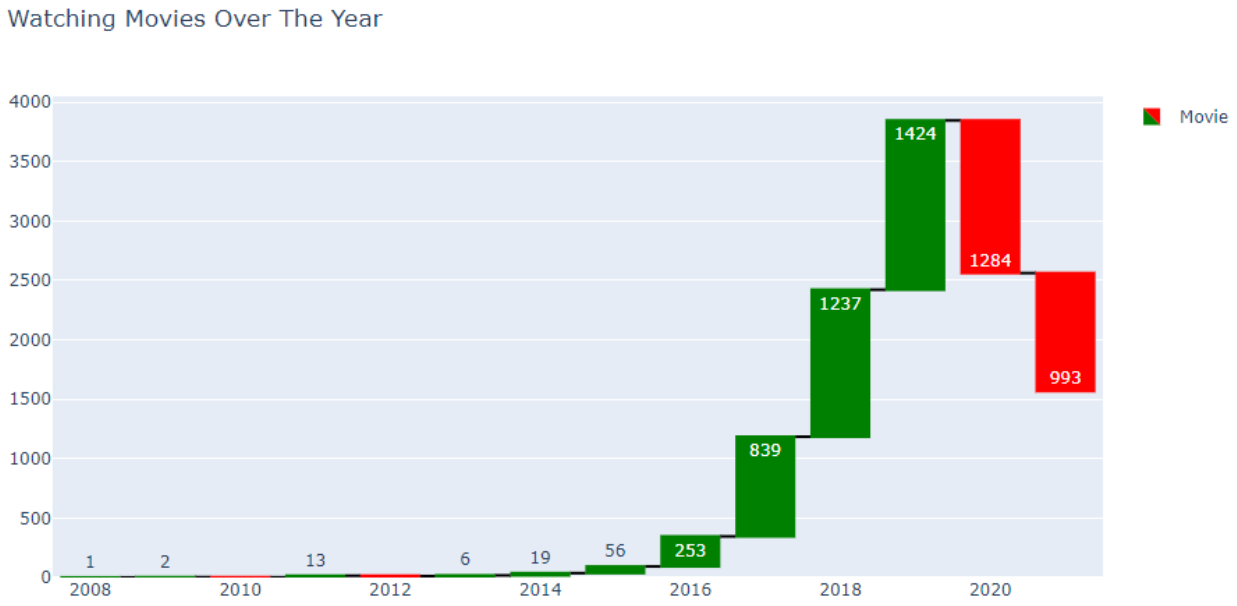
2. We saw release years. But we need to look at another feature 'date_added' for understanding which year and how much content was added to Netflix. Because the release of years of movies or TV Shows is different from the entrance of Netflix.

   Here I already separated the year_added column from the date_added at the time of data cleaning and pre-stuffs. Now it's easy to analyze.

   First, we will see for Movies.

```
3. # Filtering out the movie
4. filtered_data = df[df['type'] == 'Movie']
5.
6. # Assigning a column name for shortcut
7. col ='year_added'
8.
9. # Filtering out the year
10.      filtered_data = filtered_data[filtered_data['year_added'] !=
   'No Year']
11.
12.      # creating a new data set with year and count
13.      filtered_data_new =
   filtered_data[col].value_counts().reset_index().rename(columns =
   {col : "count", "index" : col})
14.
15.
16.      # sorting the year column
17.      filtered_data_new = filtered_data_new.sort_values(col)
18.
19.      # try to visualize
20.      fig1 = go.Figure(go.Waterfall(
21.          name = "Movie", orientation = "v",
22.          x = filtered_data_new['year_added'].values,
23.          textposition = "auto",
24.          text = ["1", "2", "1", "13", "3", "6", "19", "56", "253",
   "839", "1237", "1424", "1284", "993"],
25.          y = [1, 2, -1, 13, -3, 6, 19, 56, 253, 839, 1237, 1424, -
   1284, -993],
26.
```

```
        connector = {"line":{"color":"black"}},
        increasing = {"marker":{"color":"green"}},
        decreasing = {"marker":{"color":"red"}},

    ))

    # updating the title
    fig1.update_layout(
            title = "Watching Movies Over The Year",
            showlegend = True
                        )
    fig1.show()
```



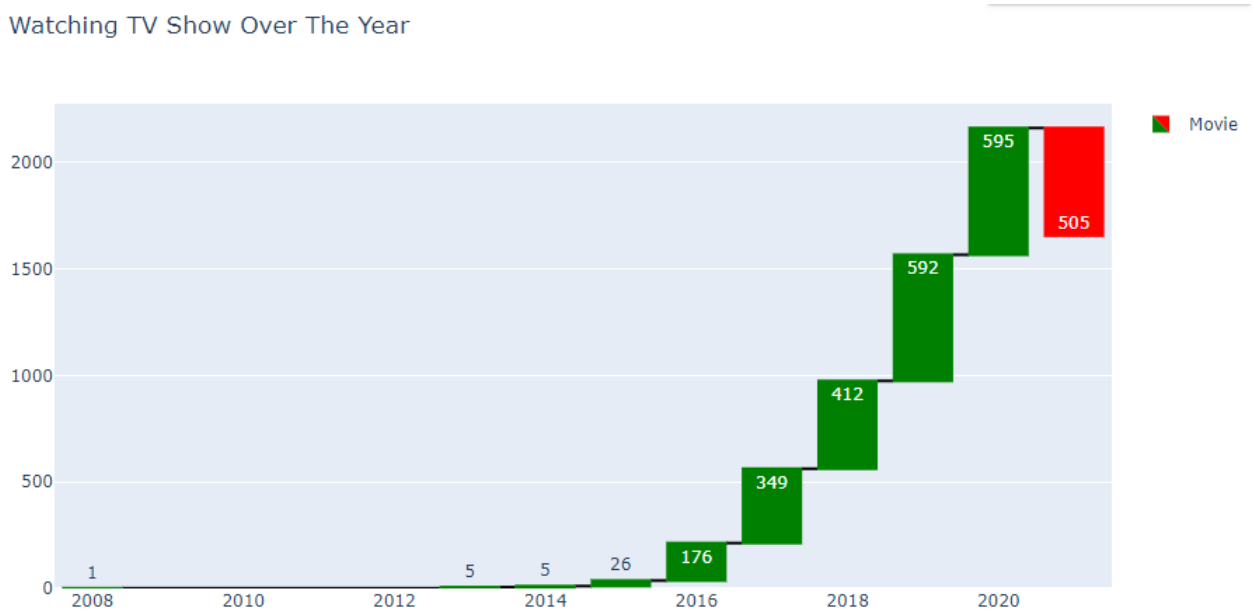
Observations

- Content added year are in mostly increasing after 2014.
- The highest number of movies were released in 2018 and 2019. Due to the covid releasing of movies were significantly dropped afterwards.

Now we will try to see for TV Shows.

```
# Filtering out the TV Show
filtered_data = df[df['type'] == 'TV Show']

# Assigning a column name for shortcut
col ='year_added'

# Filtering out the year
```

```
filtered_data = filtered_data[filtered_data['year_added'] != 'No Year']

# creating a new data set with year and count
filtered_data_new =
filtered_data[col].value_counts().reset_index().rename(columns = {col :
"count", "index" : col})


# sorting the year column
filtered_data_new = filtered_data_new.sort_values(col)

# try to visualize
fig1 = go.Figure(go.Waterfall(
    name = "Movie", orientation = "v",
    x = filtered_data_new['year_added'].values,
    textposition = "auto",
    text = ["1", "5", "5", "26", "176", "349", "412", "592", "595",
"505"],
    y = [1, 5, 5, 26, 176, 349, 412, 592, 595, -505],

    connector = {"line":{"color":"black"}},
    increasing = {"marker":{"color":"green"}},
    decreasing = {"marker":{"color":"red"}},

))

# updating the title
fig1.update_layout(
        title = "Watching TV Show Over The Year",
        showlegend = True
                    )
fig1.show()
```



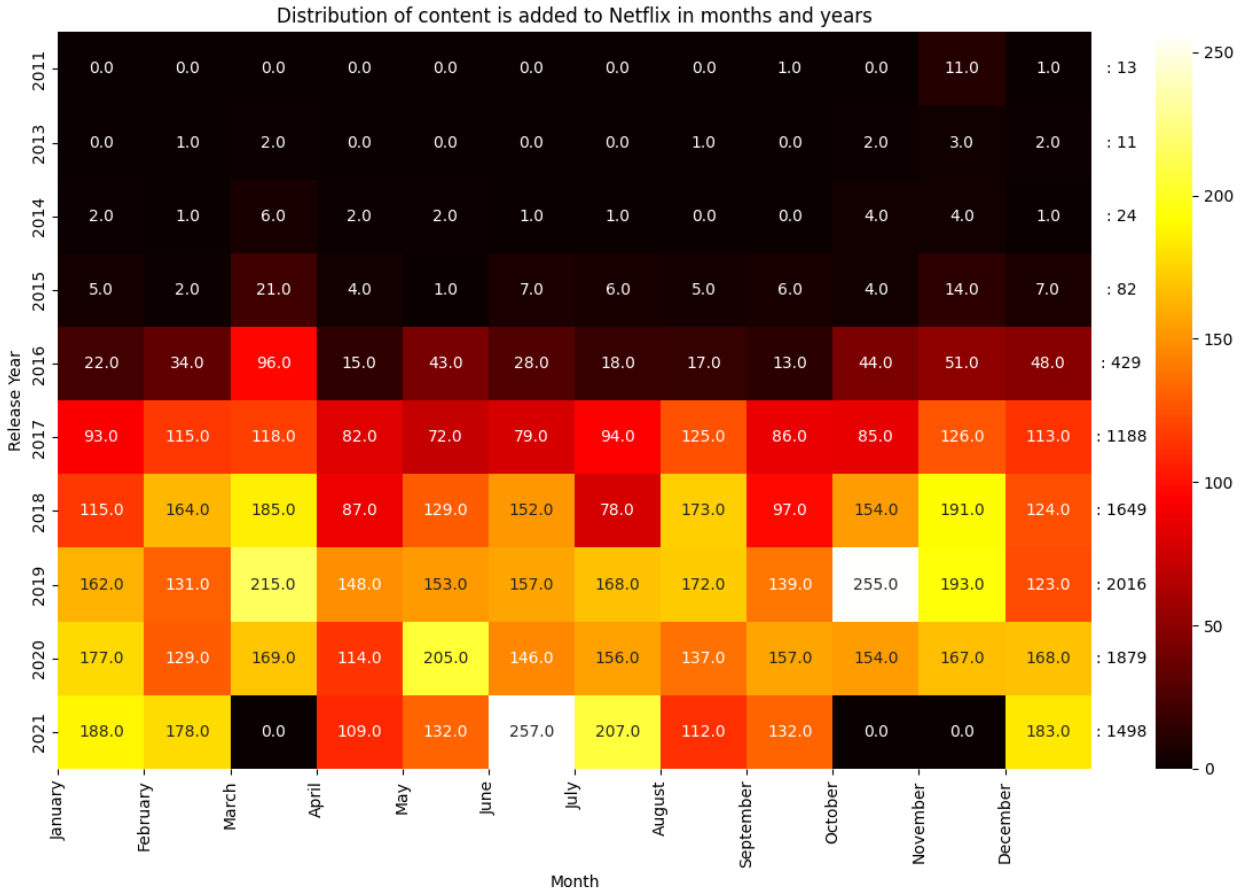
Observations

- First show was released on 2008 then it has been stopped for a while.
- Content added year are in mostly increasing after 2014.
- The highest number of shows were released in 2020.

No will check for Month and year wise and try to find some trend.

```
# Filter out rows with 'missing' in the 'date_added' column
filtered_df = df[df['date_added'] != 'Missing'].copy()

# Change to pandas date_time format
filtered_df['release_date'] = pd.to_datetime(filtered_df['date_added'])

filtered_df['release_day'] = filtered_df['release_date'].dt.day
filtered_df['release_month'] = filtered_df['release_date'].dt.month_name()
filtered_df['release_year'] = filtered_df['release_date'].dt.year

# Count the occurrences of each release year
year_counts = filtered_df['release_year'].value_counts()

# Select the years with the most movies
top_years = year_counts.head(10).index

# Filter the data for the selected years
```

```
filtered_df =
filtered_df[filtered_df['release_year'].isin(top_years)].copy()

# Count the occurrences of each release year and month combination
release_counts = filtered_df.groupby(['release_year',
'release_month']).size().unstack().fillna(0)

# Create a heatmap plot
plt.figure(figsize=(12, 8))  # Set the figure size
sns.heatmap(release_counts, cmap='hot', annot=True, fmt=".1f")

# Add labels and title
plt.xlabel('Month')
plt.ylabel('Release Year')
plt.title('Distribution of content is added to Netflix in months and
years')

# Customize x-axis tick labels to show month names
month_labels = ['January', 'February', 'March', 'April', 'May', 'June',
'July',
                'August', 'September', 'October', 'November', 'December']
plt.xticks(ticks=range(0, 12), labels=month_labels)

# Add total count annotations for each year below the bars
for i, year in enumerate(release_counts.index):
    plt.text(12.35, i + 0.5, f': {year_counts.loc[year]}', ha='center',
va='center')

# Display the plot
plt.tight_layout()
plt.show()
```



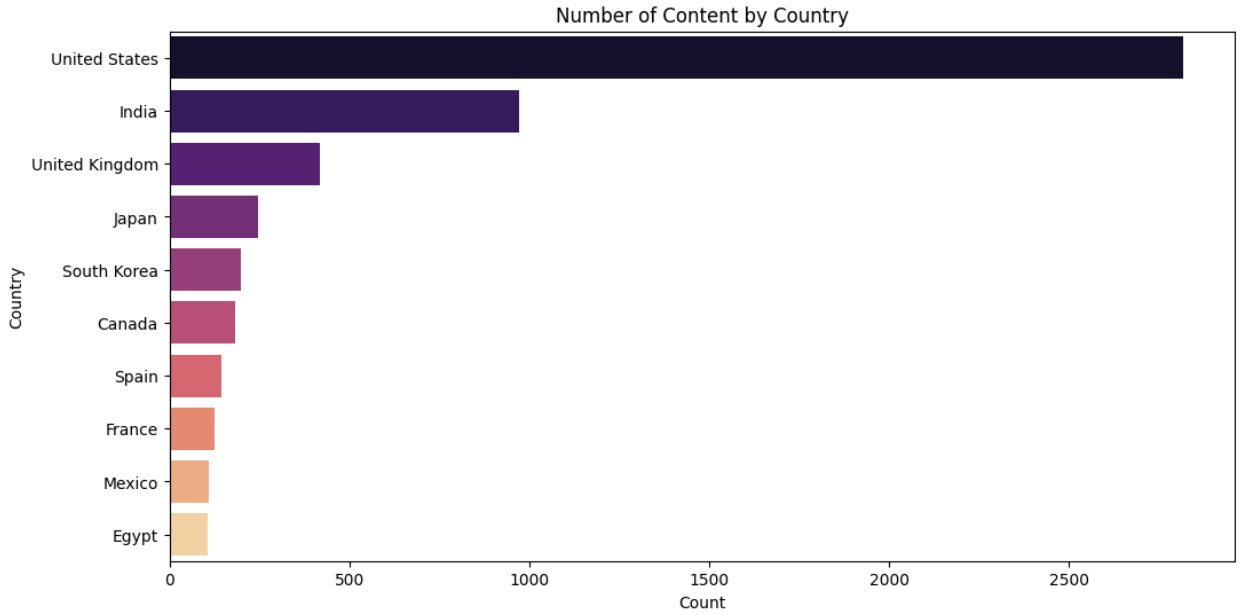
Observations

- There are differences between the release dates of the content and the dates it was added to Netflix. For Example,
    - In 2018, 1147 content was released to the world but Netflix had 1625 content that year.
    - In 2015, 560 content was released to the world but Netflix had only 73 content that year.
        - We can see that most of the content released in 2016 and before has been incorporated into the following years.
- And the density of content numbers in Netflix, from which we received the data, belongs to the years 2019 and 2020.

2. Content Numbers by Country

```
# Filter out rows with 'No Country' in the 'country' column
```

```
new_df = df[df['country'] != 'No Country'].copy()
#count of the country
counts_of_country = new_df['country'].value_counts()

# top 10 country
top_country = counts_of_country.head(10)
# Create a bar plot
plt.figure(figsize=(12, 6))  # Set the figure size
sns.barplot(x=top_country.values, y=top_country.index, palette='magma')

# Add labels and title
plt.xlabel('Count')
plt.ylabel('Country')
plt.title('Number of Content by Country')

# Display the plot
plt.show()
```



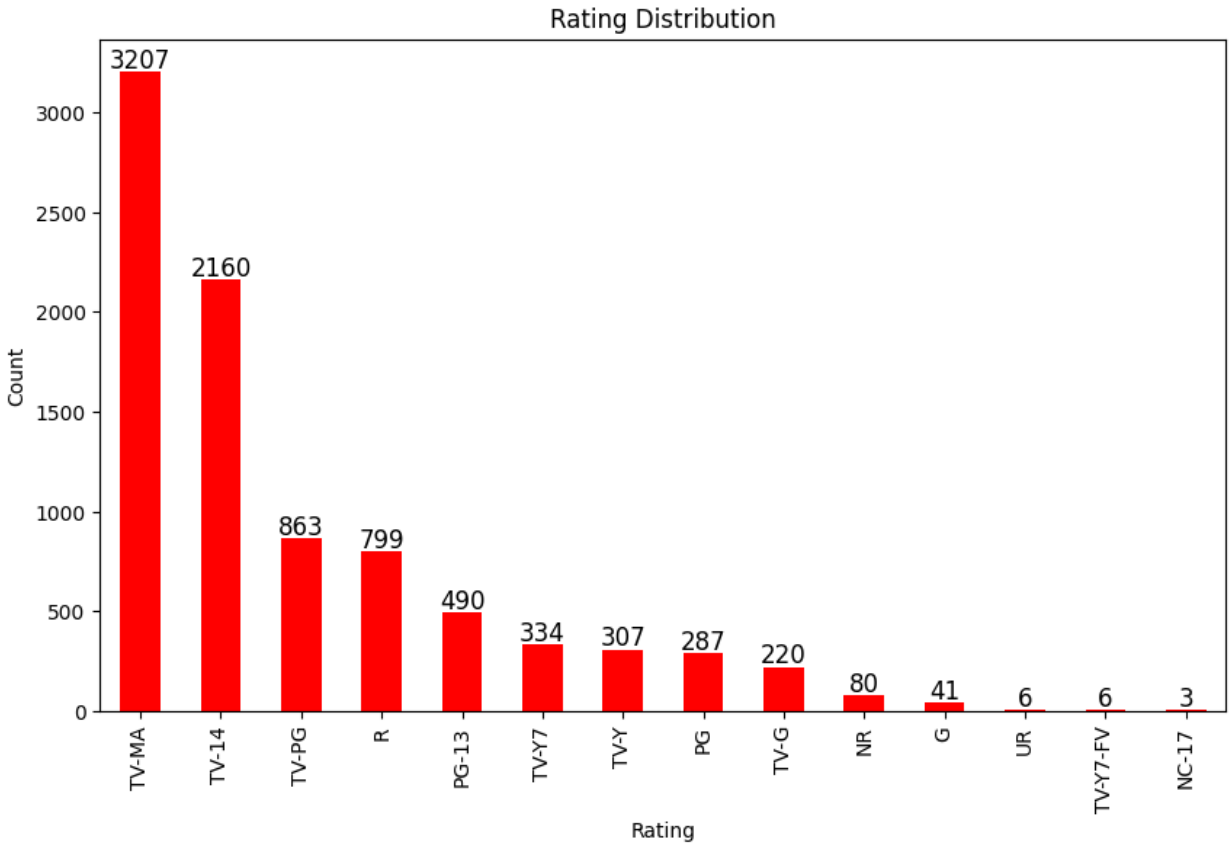
Observations

- We can say here, top 10 countries producing the most content.
    - US, India, UK, Japan and South Korea on top

4. Rating analysis

Distribution of Rating and finding what audience prefer to watch. With this analysis we get to know which content is most preferable by audience. By the view of this result Netflix can get an idea like what type of content they can release next.

```
# Filter out rows with 'missing' in the 'rating' column
filtered_df = df[df['rating'] != 'Missing'].copy()

# Count the occurrences of each rating
rating_counts = filtered_df['rating'].value_counts()

# Create a bar plot
plt.figure(figsize=(10, 6))
plot = rating_counts.plot(kind='bar', color='red')

# Add labels and title
plt.xlabel('Rating')
plt.ylabel('Count')
plt.title('Rating Distribution')

for i in plot.patches:
  height = i.get_height()
  plot.text(i.get_x() + i.get_width()/2., height + 20, height, ha =
'center', fontsize=12)

# Display the plot
plt.show()
```



Observations

- Most used ratings TV-MA, TV-14, TV-PG is used in TV shows.
  - That means Tv shows mostly for Mature or 14+
  - The TV-MA rating is a type of rating given by the TV parental guidelines to a television program.
- R rating in fourth place used in Movies.
  - That means Movies in Netflix mostly for Mature.
- The least preferred rating shows are Nc-17.
- We can say Netflix better place for adult, and mature content.

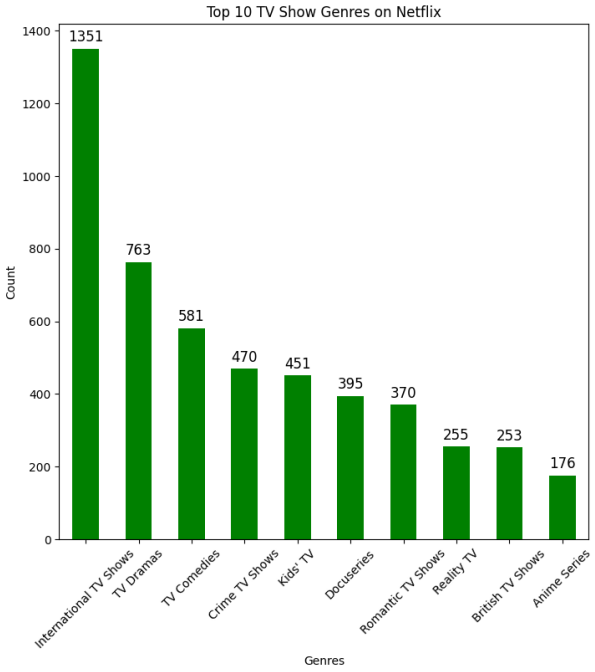
5. Popular Genres

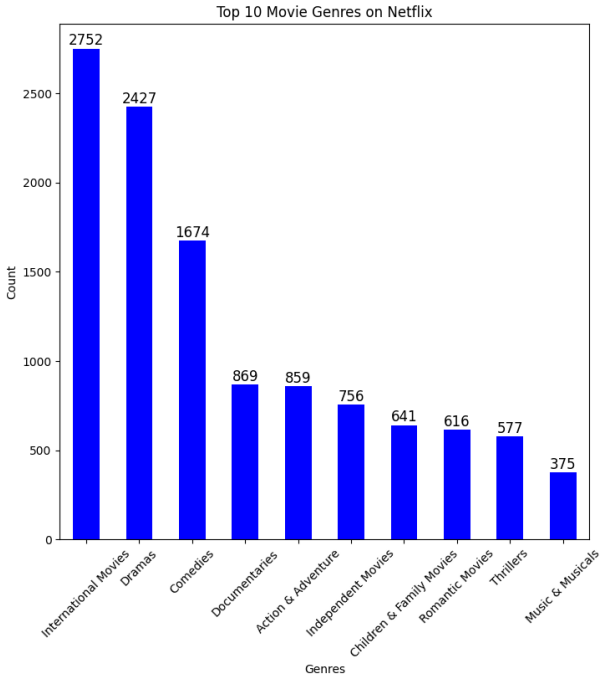
To know more about the distribution of genres and see which type of content do audience prefers to watch. So, Netflix can decide and take movies or tv shows of the highest watched genres which will benefit Netflix in a long run.

Here I will try to check against Movies and TV Show at a time.

```
# Extract the genres for TV Show
genres_show = df['listed_in'].str.split(', ').explode()[df['type'] == 'TV
Show']

# Extract the genres for Movies
genres_movie = df['listed_in'].str.split(', ').explode()[df['type'] ==
'Movie']

# Count the occurrences of each genre for TV Show
genre_counts_show = genres_show.value_counts()

# Count the occurrences of each genre for Movie
genres_count_movie = genres_movie.value_counts()

# Select the top 10 popular genres for TV Show
top_genres_show = genre_counts_show.head(10)

# Select the top 10 popular genres for Movie
top_genres_movie = genres_count_movie.head(10)

plt.figure(figsize=(18, 8))
plt.subplot(1,2,1)
# Create a bar plot
plot = top_genres_show.plot(kind='bar', color='green')

# Add labels and title
plt.xlabel('Genres')
plt.ylabel('Count')
plt.title('Top 10 TV Show Genres on Netflix')

# Rotate x-axis labels for better visibility
plt.xticks(rotation=45)

for i in plot.patches:
  height = i.get_height()
  plot.text(i.get_x() + i.get_width()/2., height + 20, height, ha =
'center', fontsize=12)


plt.subplot(1,2,2)
# Create a bar plot
plot1 = top_genres_movie.plot(kind='bar', color='blue')

# Add labels and title
plt.xlabel('Genres')
```

```
plt.ylabel('Count')
plt.title('Top 10 Movie Genres on Netflix')

# Rotate x-axis labels for better visibility
plt.xticks(rotation=45)

for i in plot1.patches:
  height = i.get_height()
  plot1.text(i.get_x() + i.get_width()/2., height + 20, height, ha =
'center', fontsize=12)
# Display the plot
plt.show()
```





Observations

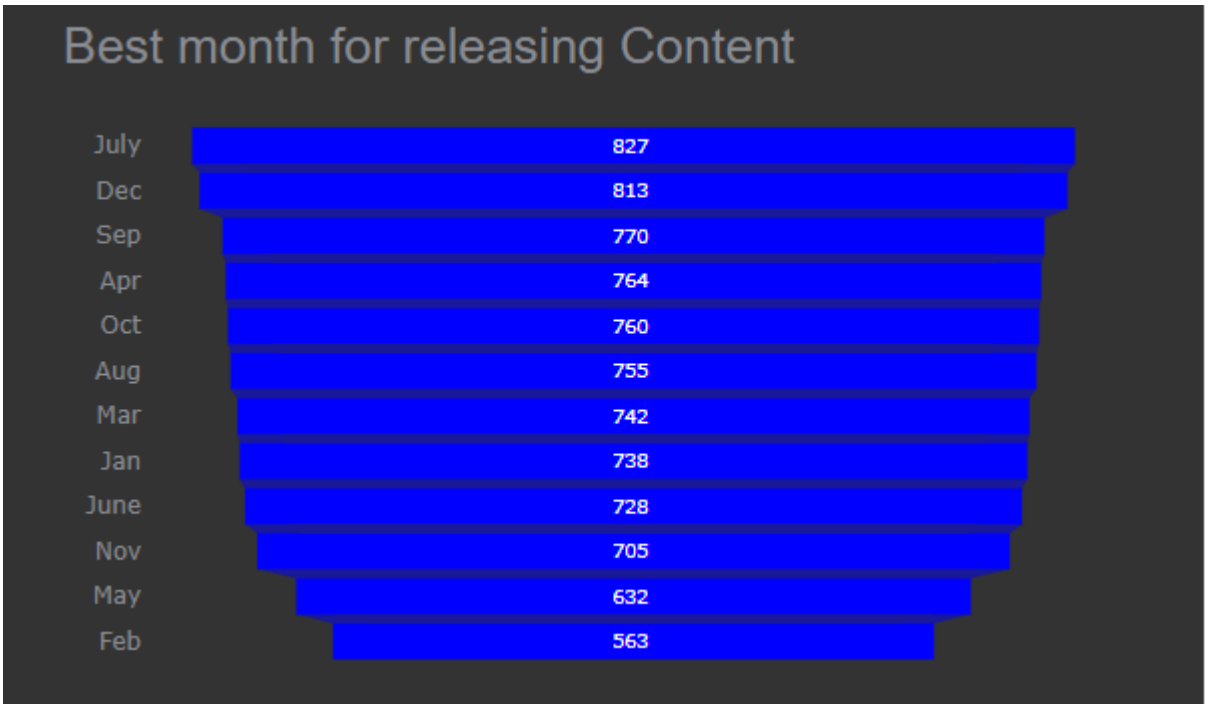
- Most Content in Netflix.
  - International Movies, Dramas and Comedies are top in case of Movies.
  - Wherever International TV Shows, TV Dramas and TV Comedies are in top in case of TV Shows.
- There is international Movies and International Tv shows in top for both the cases, and I think this is good for Netflix. We can say, Netflix has users around the world, and they can get a good revenue.
- If we look just regular genres. Drama is first. Because drama mostly related almost every other genre. There's not a lot of movies or TV Shows pure Drama.

- Other Coming genre is Comedies, and this is I think Obvious, because reel world losing funniness. We should fun more.

6. Best month to produce a new content

The best month to release content so the producer can gain much revenue.

```
# First Filtering out with No month value
filtered_month =  df[df['month_added'] != 'No Month'].copy()

# Creating a dataframe with the months and their count on it.
df_month =
pd.DataFrame(filtered_month.month_added.value_counts()).reset_index().rena
me(columns={'index':'month','month_added':'count'})

# converting month number to month name
df_month['month_final'] = df_month['month'].replace({1:'Jan', 2:'Feb',
3:'Mar', 4:'Apr', 5:'May', 6:'June', 7:'July', 8:'Aug', 9:'Sep', 10:'Oct',
11:'Nov', 12:'Dec'})

#Now design the plot to visualize a good way
fig_month = px.funnel(df_month, x='count', y='month_final', title='Best
month for releasing Content',
                      height=350, width=600,
color_discrete_sequence=['blue'])
fig_month.update_xaxes(showgrid=False, ticksuffix=' ', showline=True)
fig_month.update_traces(hovertemplate=None,
marker=dict(line=dict(width=0)))
fig_month.update_layout(margin=dict(t=60, b=20, l=70, r=40),
                        xaxis_title=' ', yaxis_title=" ",
                        plot_bgcolor='#333', paper_bgcolor='#333',
                        title_font=dict(size=25, color='#8a8d93',
family="Lato, sans-serif"),
                        font=dict(color='#8a8d93'),
                        hoverlabel=dict(bgcolor="black", font_size=13,
font_family="Lato, sans-serif"))
```



Observations

- From the above result set we can say most of the holidays came in July and December month so to releases a Movie or TV show in December and July is the best way to earn a lot of profit as the whole family will be spending time with each other and watching shows.
- The best four month to release a content are July, December, September, and April.

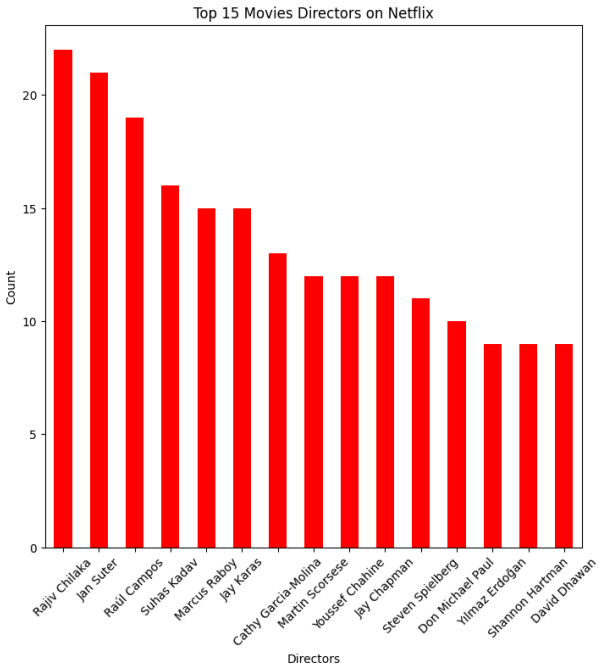
7. The Director

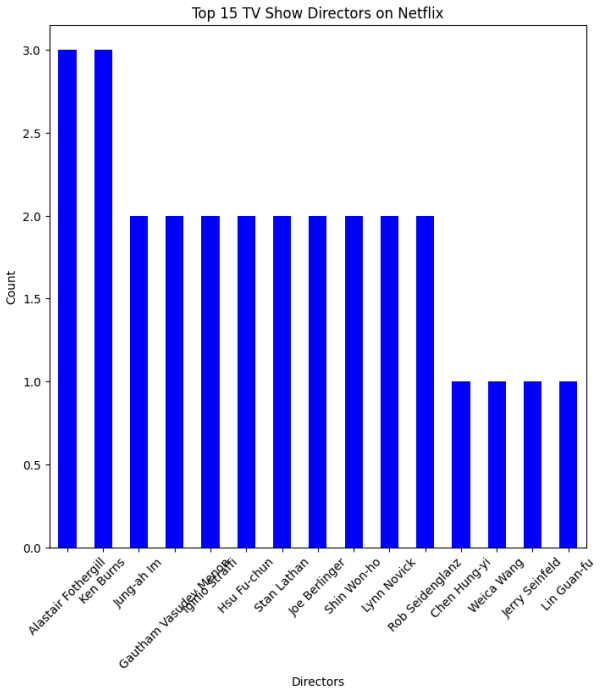
Now we will see top directors who contributed most.

```
# first filtering out the directors. Where we dont have any directors we
will not pick it up
filtered_dir = df[df['director'] != 'No Director']

# As per dataset there is couple/many of directors directed a single movie
or show. Time to split those and make a new dataframe.
# Filterling Director with Movie Categories
filtered_dir_movie = filtered_dir['director'].str.split(',
').explode()[filtered_dir['type']== 'Movie']
```

```
# Now will count the occurence of each director for the Movie
filtered_dir_movie_count = filtered_dir_movie.value_counts()

# now will collect top 15 directors among of them
top15_dir_movie = filtered_dir_movie_count.head(15)


# Filterling Director with TV Show Categories
filtered_dir_show = filtered_dir['director'].str.split(',
').explode()[filtered_dir['type']== 'TV Show']

# Now will count the occurence of each director for the TV Show
filtered_dir_show_count = filtered_dir_show.value_counts()

# now will collect top 15 directors among of them
top15_dir_show = filtered_dir_show_count.head(15)


# Now time to visualize the data
plt.figure(figsize=(18, 8))
plt.subplot(1,2,1)
# Create a bar plot
plot = top15_dir_movie.plot(kind='bar', color='green')

# Add labels and title
plt.xlabel('Directors')
plt.ylabel('Count')
plt.title('Top 15 Movies Directors on Netflix')

# Rotate x-axis labels for better visibility
plt.xticks(rotation=45)


plt.subplot(1,2,2)
# Create a bar plot
plot1 = top15_dir_show.plot(kind='bar', color='blue')

# Add labels and title
plt.xlabel('Directors')
plt.ylabel('Count')
plt.title('Top 15 TV Show Directors on Netflix')

# Rotate x-axis labels for better visibility
plt.xticks(rotation=45)
```

```
# Display the plot
plt.show()
```





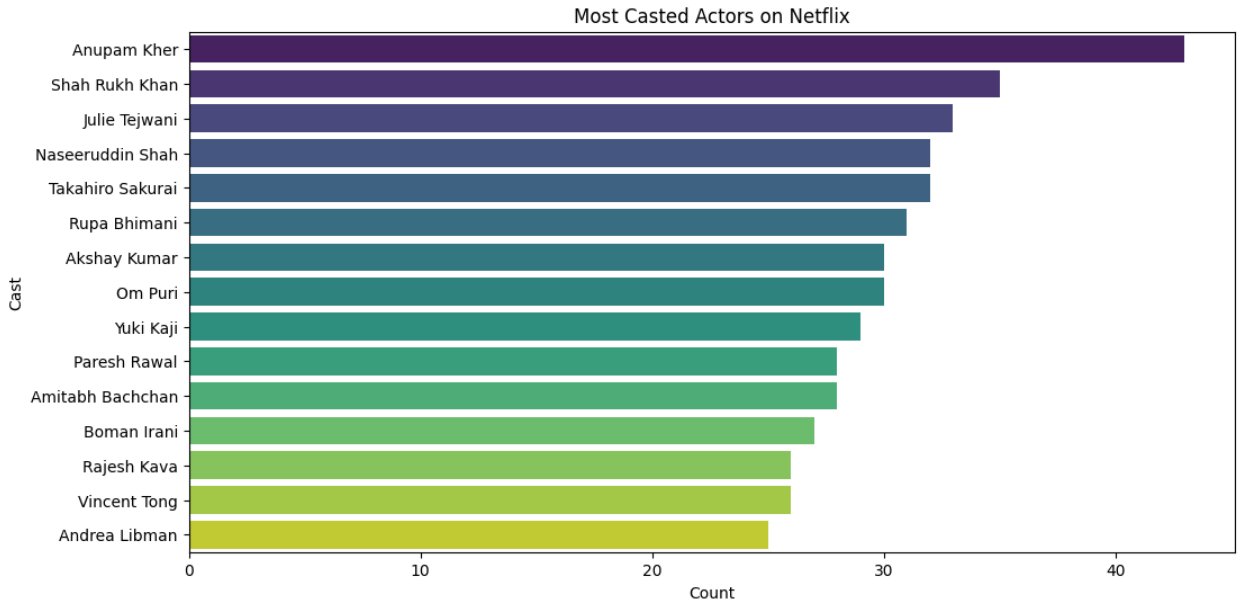
Observations

- Most Directors in Netflix.
  - Rajiv Chilaka, Jan Suter and Rahul Campos are top directors in case of Movies.
  - Wherever Alastair Fothergill, Ken Burns contributed three and Jung-ah Im, Gautham Vasudev Menon, Iginio Straffi, Hsu Fu-chun, Stan Lathan, Joe Berlinger, Shin Won-ho, Lynn Novick, Rob Seidenglanz all are contributed two in case of TV Shows.
- Can see Directors are love make movies instead of TV Shows and Netflix are like to release movies over TV Shows.
- Can see top directors movies are more or equal to twenty where ever the TV shows are very less number.

8. Most Casted Actors

```
# Filtering the cast
```

```
filtered_cast = df[df['cast'] != 'No Cast']

# Splitting the cast to get individual
filtered_cast_new = filtered_cast['cast'].str.split(', ').explode()

# counting cast
filtered_cast_count = filtered_cast_new.value_counts()

# filtering top 15 cast
top_cast = filtered_cast_count.head(15)

plt.figure(figsize=(12, 6))  # Set the figure size
sns.barplot(x=top_cast.values, y=top_cast.index, palette='viridis')

# Add labels and title
plt.xlabel('Count')
plt.ylabel('Cast')
plt.title('Most Casted Actors on Netflix')

# Display the plot
plt.show()
```



Observations

- We can say here, top 15 cast part of that most Movies or TV Shows.

- Topmost Actor is Anupam Kher while ShahRukh Khan is the second most acted star in this dataset.

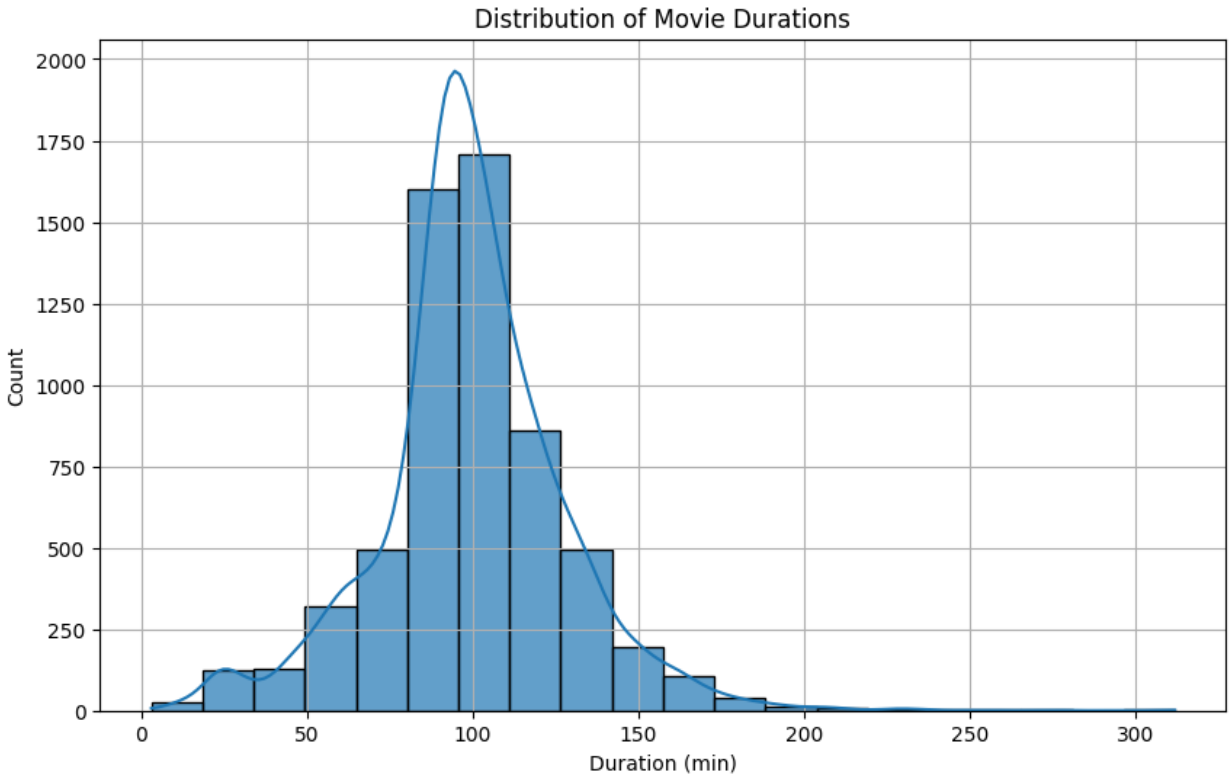
9. Duration Analysis

1. How do movies length change on Netflix? Let's have a look on this and try to figure out some trend.

```
# Extract the movie durations without 'No Duration' values
movie_durations = df[(df['type'] == 'Movie') & (df['duration'] != 'No Duration')]['duration']

# Convert the durations to numeric values (remove 'min' suffix)
movie_durations = movie_durations.str.replace(' min', '').astype(int)

# Calculate summary statistics
min_duration = movie_durations.min()
    max_duration = movie_durations.max()
    mean_duration = movie_durations.mean()
    median_duration = movie_durations.median()

    # Print the summary statistics
    print("Duration Summary Statistics:")
    print("Minimum duration: {} min".format(min_duration))
    print("Maximum duration: {} min".format(max_duration))
    print("Mean duration: {:.2f} min".format(mean_duration))
    print("Median duration: {:.2f} min".format(median_duration))

    # Create a histogram of movie durations
    plt.figure(figsize=(10, 6))  # Set the figure size
    sns.histplot(movie_durations, bins=20, edgecolor='black', alpha=0.7, kde = True)
    plt.xlabel('Duration (min)')
    plt.ylabel('Count')
    plt.title('Distribution of Movie Durations')
    plt.grid(True)
    plt.show()
```

```
Duration Summary Statistics:
Minimum duration: 3 min
Maximum duration: 312 min
Mean duration: 99.58 min
```

```
Median duration: 98.00 min
```



Observations

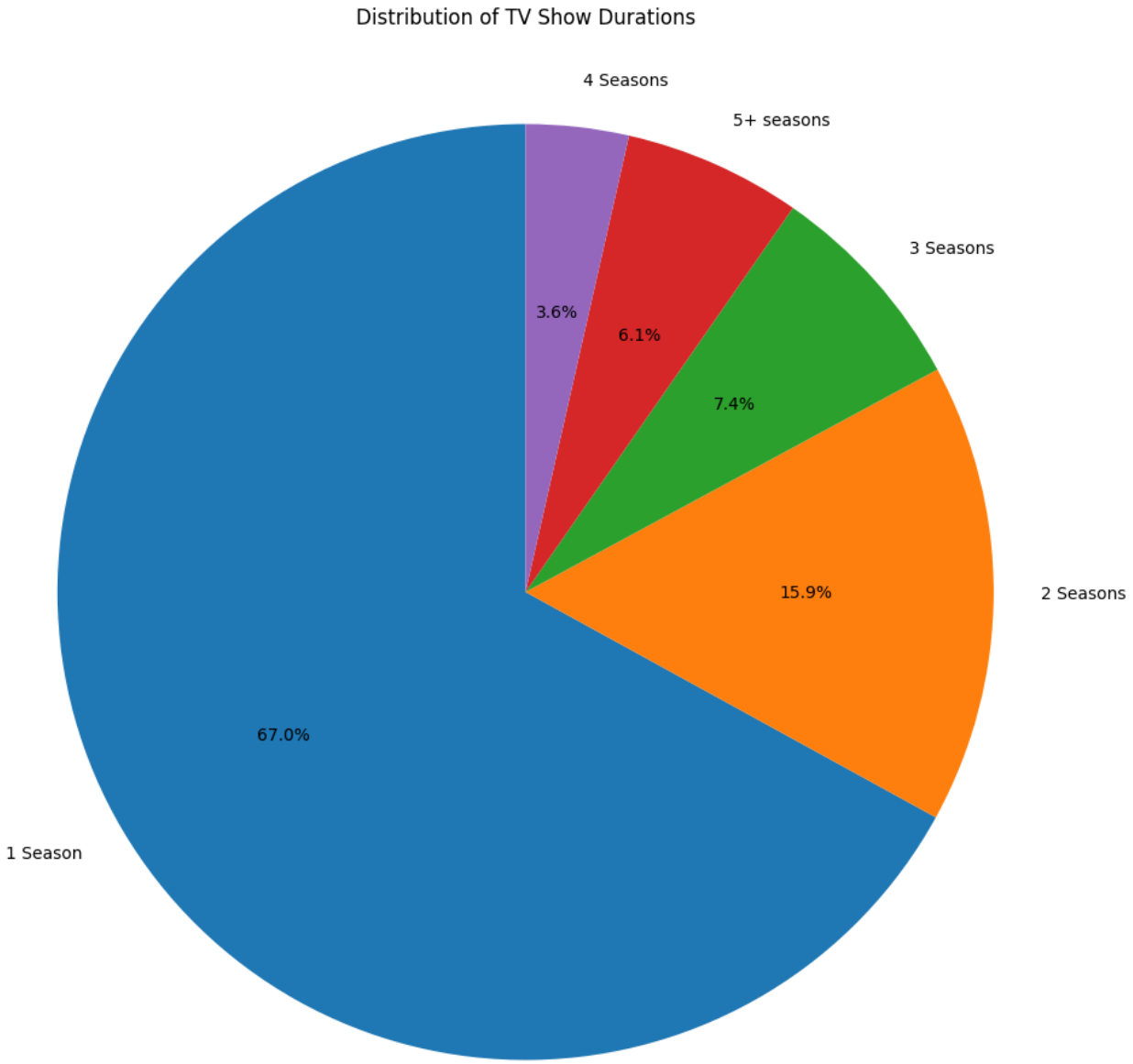
- The average movie length is around 100 minutes.
- And median is 98 minutes.
- Density of duration not really variant, we can see this from median and mean values, they are close to each other. We can say, around 100 min. ideal for audience.

2. What about TV Shows?

   Which season more on demand?

```
# Filter TV show durations without 'No Duration' values
tv_show_durations = df[(df['type'] == 'TV Show') & (df['duration'] != 'No
Duration')]['duration']

# Combine durations after 6 seasons into a single category
tv_show_durations = tv_show_durations.apply(lambda x: '5+ seasons' if
int(x.split(' ')[0]) >= 5 else x)

# Count the occurrences of each duration
duration_counts = tv_show_durations.value_counts()
```

```
# Create a pie chart
plt.figure(figsize=(11, 12))
plt.pie(duration_counts, labels=duration_counts.index, autopct='%1.1f%%',
startangle=90)
plt.title('Distribution of TV Show Durations')

# Equal aspect ratio ensures that pie is drawn as a circle
plt.axis('equal')

# Display the chart
plt.show()
```



Observations

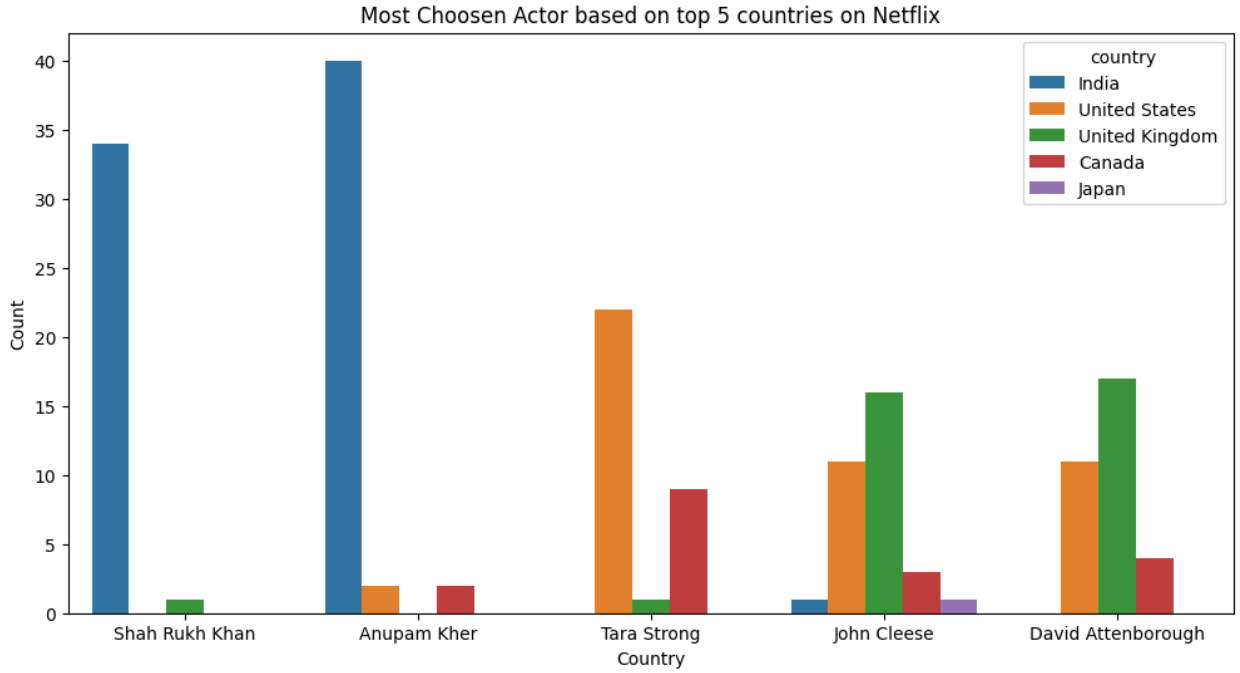
- 67% of tv shows fall under Season 1, that is very sweet.
- Seasons 2 tv shows are 15.9% and 3 seasons tv shows are 7.4% and keeps decreasing. That's normal because, sometimes I want to sit and binge a TV Show. This is imposable if I want to watch Dexter again.

9. We will see here top actors based on top countries.

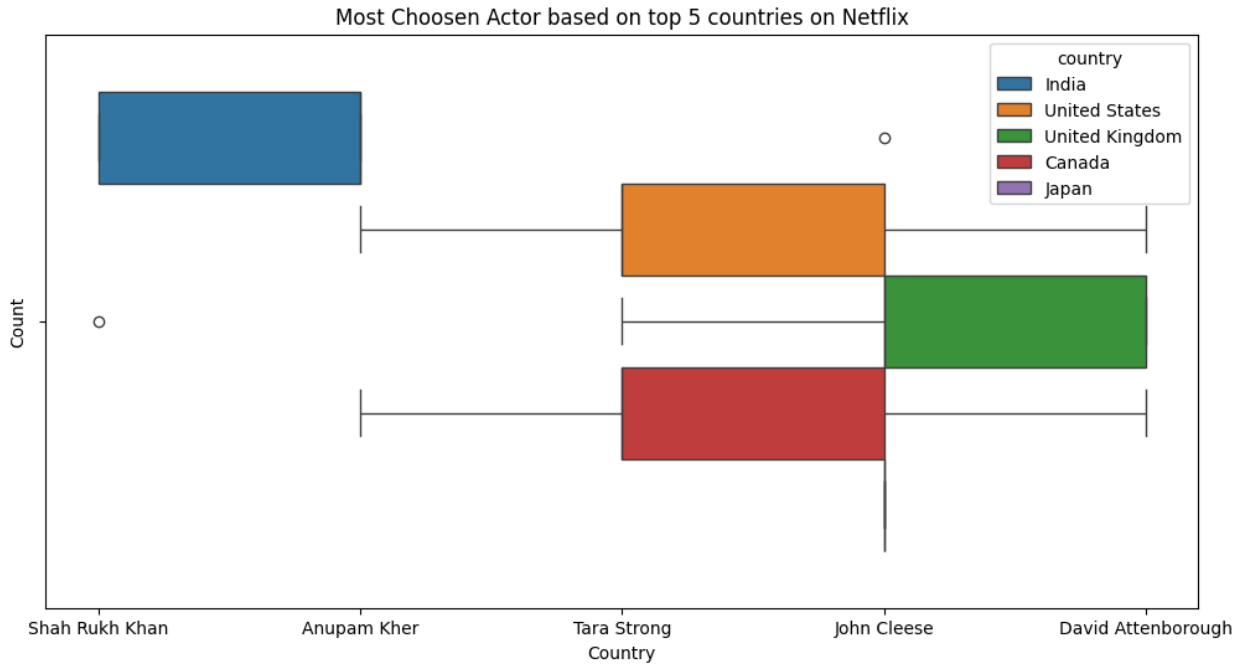
First will try to visualize with Barplot and then Box plot also.

```
# Now we want to check top 5 cast based on top 5 country
# First filtering out the 'No Country' then 'No Cast'
filtered_data = df[(df['country'] != 'No Country') &
                   (df['cast'] != 'No Cast')]

# Now split 'cast' and 'country' one by one
filtered_data['cast'] = filtered_data['cast'].str.split(', ')
filtered_data['country'] = filtered_data['country'].str.split(', ')


# try to explode those column one by one
filtered_data = filtered_data.explode('cast')
filtered_data = filtered_data.explode('country')


# now will reset the index of whole data
filtered_data = filtered_data.reset_index()

# now will delete the index column from the dataset
filtered_data = filtered_data.drop(columns = 'index', axis =1)

# finding top 5 country & cast
top5_country = filtered_data['country'].value_counts().index[:5]
top5_cast = filtered_data['cast'].value_counts().index[:5]


# now trying to get top 5 data for both of cast and country
top5_data = filtered_data[(filtered_data['cast'].isin(top5_cast)) &
                          (filtered_data['country'].isin(top5_country))]
```

```
# now try to visualize the data
plt.figure(figsize=(12, 6))  # Set the figure size
sns.countplot(data = top5_data, x = 'cast', hue = 'country')

# Add labels and title
plt.xlabel('Country')
plt.ylabel('Count')
plt.title('Most Choosen Actor based on top 5 countries on Netflix')


plt.show()
```




```
plt.figure(figsize=(12, 6))  # Set the figure size

sns.boxplot(data = top5_data, x = 'cast', hue = 'country')

# Add labels and title
plt.xlabel('Country')
plt.ylabel('Count')
plt.title('Most Choosen Actor based on top 5 countries on Netflix')
plt.show()
```

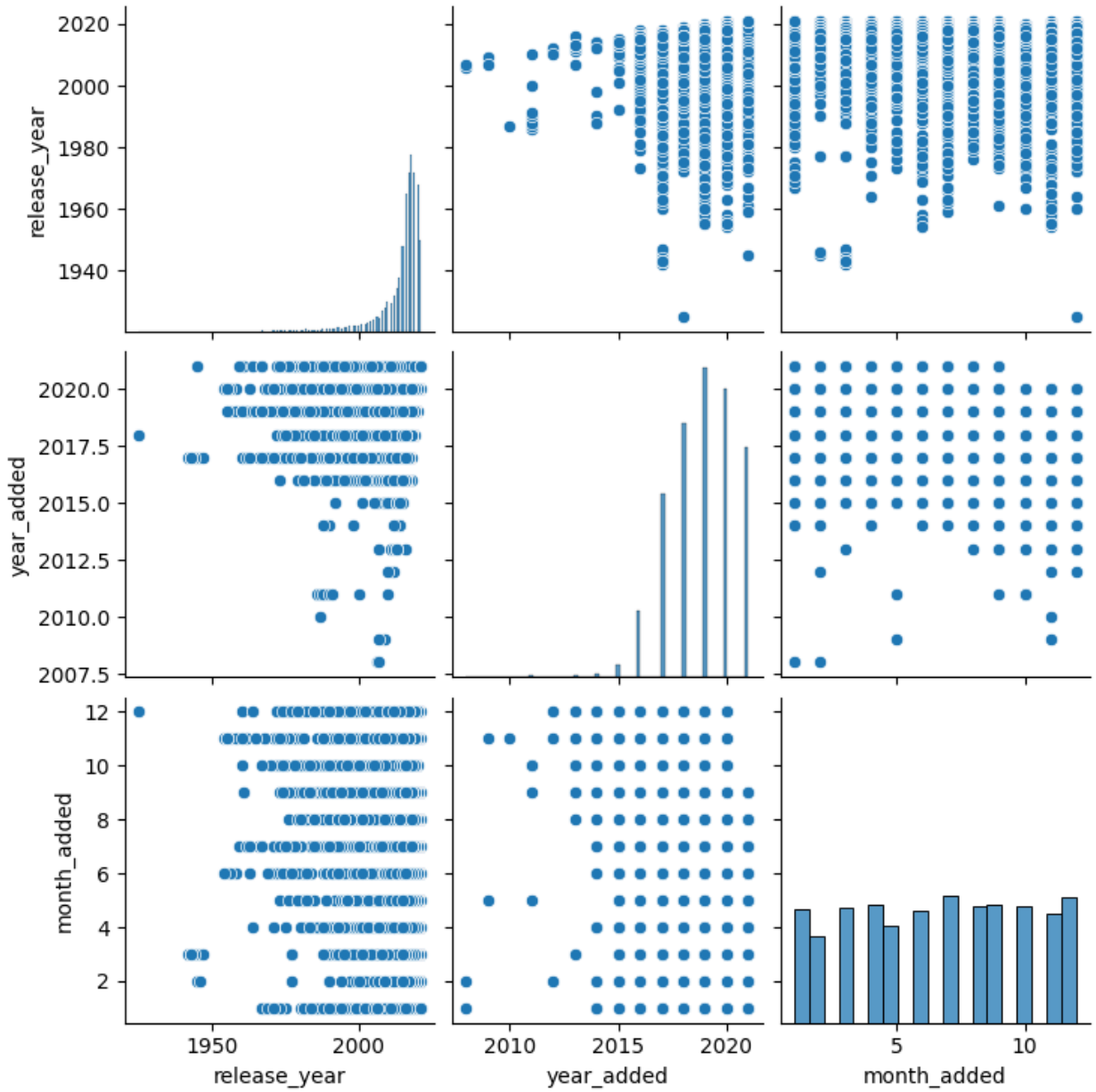


Observations

- Top five actors are Shah Rukh Khan, Anupam Kher, Tara, John and David
- Top five countries are India, US, UK, Canada and Japan.
- Shah Rukh Khan is done his carrier most of India only as compare to others.
- Anupam Kher done India as well some of the movie in US and Canada as well.
- Can see John has done with all five countries itself.

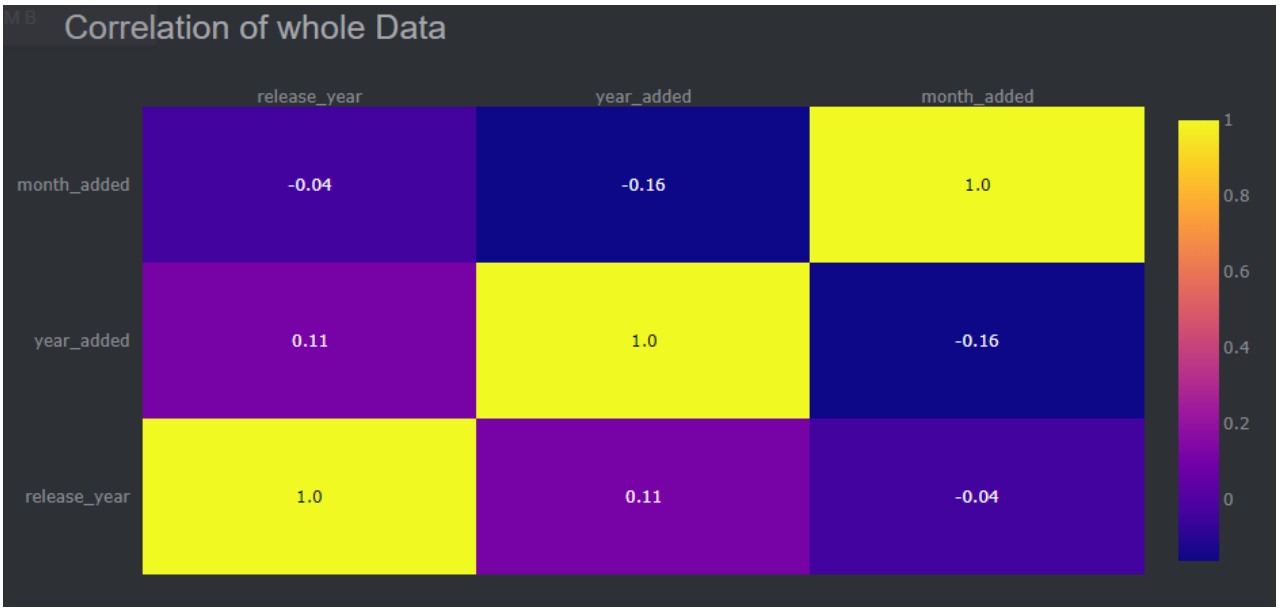
10. We will see here some correlation between features.

```
# Heatmap

# First filtering out the 'year_added' then 'month_added'
filtered_data = df[(df['year_added'] != 'No Year') &
                   (df['month_added'] != 'No Month')]

# Now we need to convert our year and month to int type for genarating
heat map, as we only have release_year
filtered_data['month_added'] =
filtered_data['month_added'].astype('int64')
filtered_data['year_added'] = filtered_data['year_added'].astype('int64')

sns.pairplot(data = filtered_data)
plt.show()
```




```
# Heatmap

# First filtering out the 'year_added' then 'month_added'
filtered_data = df[(df['year_added'] != 'No Year') &
                   (df['month_added'] != 'No Month')]

# Now we need to convert our year and month to int type for genarating
heat map, as we only have release_year
```

```
filtered_data['month_added'] =
filtered_data['month_added'].astype('int64')
filtered_data['year_added'] = filtered_data['year_added'].astype('int64')

# Correlation between the feature show with the help of visualisation
corrs = filtered_data.corr()
fig_heatmap = ff.create_annotated_heatmap(
    z=corrs.values,
    x=list(corrs.columns),
    y=list(corrs.index),
    annotation_text=corrs.round(2).values,
    showscale=True)
fig_heatmap.update_layout(title= 'Correlation of whole Data',
                          plot_bgcolor='black', paper_bgcolor='#2d3035',
                          title_font=dict(size=25, color='#a5a7ab',
family="Muli, sans-serif"),
                          font=dict(color='#8a8d93'))
```



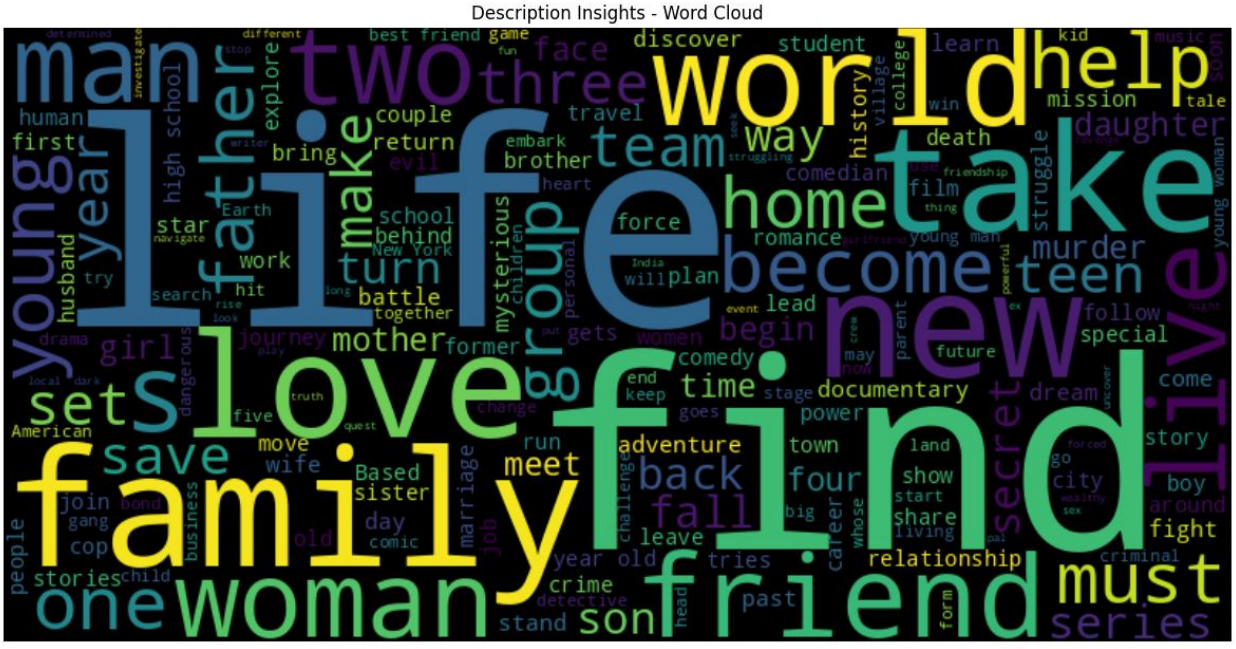
Observations

- We can see correlation is 1 between diagonally.
- There is some positive relation between release year and the content added year in Netflix.
- There is some negative trend or relation between month added - release year and year added – month added.

11. Word Cloud From Content Descriptions

By looking at the descriptions of the content of movies or TV series, we can say how the content is consumed more. There is always a supply and demand relation.

```
# Filter out missing description values
description_data = df[df['description'] != 'missing']['description']

# Join all the cast names into a single string
description_text = ' '.join(description_data)

# Generate the word cloud
wordcloud = WordCloud(width=800, height=400,
background_color='black').generate(description_text)

# Create a figure and axes
fig, ax = plt.subplots(figsize=(15, 10))

# Display the word cloud
ax.imshow(wordcloud, interpolation='bilinear')
ax.set_title('Description Insights - Word Cloud')
ax.axis('off')

# Show the plot
plt.show()
```



Observations

- We can see most used words in here. And we can say in here, which sides more important for us in life.
    - Of course ***Life*** is itself more important than everything,
    - Maybe ***find***ing ***love*** and creating a great ***family*** with good ***friend***s
    - With these comes a ***new world*** for us.
    - In ***year***s ***man become*** a ***father*** and ***woman become*** a ***mother***.
    - To ***discover life must*** first ***fall.***
    - ***Maki***ng a good ***home*** needs a ***battle.***

Insights:

• We can see, audience love to watch movies. Most of the content tends towards movies with 69.6% as compare to TV Shows(30.4%)

• After 2010, we can see that movies or series are more concentrated.

• The highest number of movies were released in 2018 and 2019. Due to the covid releasing of movies were significantly dropped afterwards.

• First show was released on 2008 then it has been stopped for a while. Content added year are in mostly increasing after 2014.

• There are differences between the release dates of the content and the dates it was added to Netflix. For Example, In 2018, 1147 content was released to the world but Netflix had 1625 content that year. In 2015, 560 content was released to the world but Netflix had only 73 content that year. We can see that most of the content released in 2016 and before has been incorporated into the following years.

• And the density of content numbers in Netflix, from which we received the data, belongs to the years 2019 and 2020.

• US, India, UK, Japan and South Korea on top in case to watch the content.

• Most used ratings TV-MA, TV-14, TV-PG is used in TV shows. That means Tv shows mostly for Mature or 14+. The TV-MA rating is a type of rating given by the TV parental guidelines to a television program. R rating in fourth place used in Movies. That means Movies in Netflix mostly for Mature. The least preferred rating shows are Nc-17. We can say Netflix better place for adult, and mature content.

- Most Content in Netflix. International Movies, Dramas and Comedies are top in case of Movies. Wherever International TV Shows, TV Dramas and TV Comedies are in top in case of TV Shows.

- There is international Movies and International Tv shows in top for both the cases, and I think this is good for Netflix. We can say, Netflix has users around the world, and they can get a good revenue.

- If we look just regular genres. Drama is first. Because drama mostly related almost every other genre. There's not a lot of movies or TV Shows pure Drama

- The best four month to release a content are July, December, September, and April as per the data.

- Most Directors in Netflix. Rajiv Chilaka, Jan Suter and Rahul Campos are top directors in case of Movies. Wherever Alastair Fothergill, Ken Burns contributed three and Jung-ah Im, Gautham Vasudev Menon, Iginio Straffi, Hsu Fu-chun, Stan Lathan, Joe Berlinger, Shin Won-ho, Lynn Novick, Rob Seidenglanz all are contributed two in case of TV Shows.

- Can see Directors are love make movies instead of TV Shows and Netflix are like to release movies over TV Shows.

- Can see top directors movies are more or equal to twenty where ever the TV shows are very less number.

- The average movie length is around 100 minutes. And median is 98 minutes. Density of duration not really variant, we can see this from median and mean values, they are close to each other. We can say, around 100 min. ideal for audience.

Recommendation (Actionable Items For business):

Interesting TV Shows should be launched more. May can focus on child section. Now a days Childs are taking their food with watching some interesting facts. If shows targeting them, it will more beneficial and make a profit.

- We all know at the time of covid, everything was stopped. Movies cannot be worked out and release. That time if any show or old funny things release more, audience will obviously make fun of it. In future it should be take care if similar situation happened.

- For some cases there is a huge gap between release of the movies and added into the Netflix platform. Today's world it should be more fast. When release team decide to launch in OTT platform, need to be catch quickly and add to it. Audience has many options. need to play mindfully.

- Need to quick survey on others country as well. What type of content needed with those countries where Netflix users are less. Base on the needs need to add content more.

- As per insights we can say content is based on mostly 14+ or matured. Need to be focused on child part base on cartoon, based on funny things, based on learning. And need to be more on old people also. They mainly like to see some God related things.

- Todays world forget to fun, all are running with the flow. How to take money and all. Best thing is Comedies, and this is I think Obvious, because reel world losing funniness. We should fun more based on the content.

- We can say most of the holidays came in December and January month for UK/US. So, releases a Movie or TV show in December and January is the best way to earn a lot of profit as the whole family will be spending time with each other and watching shows. Not only focus those two countries, need to quick survey among all where business has been spread out. Bases on the vacation release should be more.

- Need to be mined content should not be too long to watch, other way many of audience will keep losing interest.

Conclusion:

We explore the Netflix dataset and saw how to clean the data and then jump into how to visualize the data. We saw some basic and advanced level charts of Plotly like Heatmap, Donut chart, Bar chart, Funnel chart, Bi-directional Bar chart, Waterfall chart, Line chart. By understanding these patterns and trends, businesses can make informed decisions and implement strategies to optimize their operations and drive growth. Here are the key takeaways from the analysis:

Key Takeaways

- The country United States dominates the market more, indicating the need to focus on other country as well for potential growth opportunities.

- The type of content attracts 14+ or matured people. That means indicating the need to focus on other age as well for viewing more content.

- Analyzing audience's demographics can help tailor products and marketing strategies to specific target audiences, leading to increased sales.

- Offering discounts during off-peak seasons can incentivize customers and boost sales during slower periods.