1. Defining Problem Statement and Analysing basic metrics

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

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

```
df.head()
```

	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...

```
df
```

	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

The dataset contains over 8807 titles and 12 descriptions.

2: Observations on the shape of data, data types of all the attributes, conversion of categorical attributes to 'category' (If required), missing value detection, statistical summary

```
df.columns
```

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

```
df.ndim
```

```
2
```

Data types of all the attributes

```
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
```

Statistical Summary Before Data Cleaning:

```
df.describe()
```

	release_year
count	8807.000000
mean	2014.180198
std	8.819312
min	1925.000000
25%	2013.000000
50%	2017.000000
75%	2019.000000
max	2021.000000

Missing Value Detection Data Profiling & Cleaning

```
print('Column with missing value:')
print(df.isnull().any())
```

```
Column with missing value:
show_id         False
type            False
title           False
director         True
cast             True
country          True
date_added       True
release_year    False
rating           True
duration         True
listed_in       False
description     False
dtype: bool
```

```
df.T.apply(lambda x: x.isnull().sum(), axis = 1)
```

```
show_id          0
type             0
title            0
director      2634
cast           825
country        831
date_added      10
release_year     0
rating           4
duration         3
listed_in        0
description      0
dtype: int64
```

```
df.isnull().sum().sum()
```

```
4307
```

There are a total of 4307 null values across the entire dataset

Handling Missing Values in Specific Columns For the 'director' and 'cast' columns, we replace missing values with 'No Data' to maintain data integrity and avoid any bias in the analysis.

```
df['director'].replace(np.nan, 'No Data', inplace=True)
df['cast'].replace(np.nan, 'No Data', inplace=True)
```

In the 'country' column, we fill in missing values with the mode (most frequently occurring value) to ensure consistency and minimize data loss.

```
df['country'] = df['country'].fillna(df['country'].mode()[0])
```

For the 'rating' column, we fill in missing values based on the 'type' of the show. We assign the mode of 'rating' for movies and TV shows separately.

Finding the mode rating for movies and TV shows

```
movie_rating = df.loc[df['type']=='Movie','rating'].mode()[0]
tv_rating = df.loc[df['type']=='TV Show', 'rating'].mode()[0]
```

Filling missing rating values based on the type of content

```
df['rating'] = df.apply(lambda x: movie_rating if x['type'] == 'Movie' and pd.isna(x['rating']) else tv_rating if x['type'] == 'TV Show' and pd.isna(x['rating'])
                                                  else x['rating'], axis=1)
```

For the 'duration' column, we fill in missing values based on the 'type' of the show. We assign the mode of 'duration' for movies and TV shows separately.

```
movie_duration_mode = df.loc[df['type'] == 'Movie', 'duration'].mode()[0]
tv_duration_mode = df.loc[df['type'] == 'TV Show', 'duration'].mode()[0]
```

Filling missing duration values based on the type of content

```
df['duration'] = df.apply(lambda x: movie_duration_mode if x['type'] == 'Movie'
                          and pd.isna(x['duration'])
                          else tv_duration_mode if x['type'] == 'TV Show'
                          and pd.isna(x['duration'])
                          else x['duration'], axis=1)
```

Dropping rows with missing values

```
df.dropna(inplace=True)
```

Date Handling

Converting the 'date_added' column to datetime format. We extract additional attributes from the 'date_added' column to enhance our analysis capabilities. We remove the month and year values to analyze trends based on these temporal aspects.

```
df["date_added"] = pd.to_datetime(df['date_added'])

df['month_added']=df['date_added'].dt.month
df['month_name_added']=df['date_added'].dt.month_name()
df['year_added'] = df['date_added'].dt.year

df.head(3)
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	month_added	month_name_added	year_added
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	No Data	United States	2021-09-25	2020	PG-13	90 min	Documentaries	As her father nears the end of his life, filmm...	9	September	2021
1	s2	TV Show	Blood & Water	No Data	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	2021-09-24	2021	TV-MA	2 Seasons	International TV Shows, TV Dramas, TV Mysteries	After crossing paths at a party, a Cape Town t...	9	September	2021
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	United States	2021-09-24	2021	TV-MA	1 Season	Crime TV Shows, International TV Shows, TV Act...	To protect his family from a powerful drug lor...	9	September	2021

Data Transformation: Cast, Country, Listed In, and Director

To analyze categorical attributes more effectively, we transform them into separate dataframes, allowing for more leisurely exploration and analysis. For the 'cast,' 'country,' 'listed_in,' and 'director' columns, we split the values based on the comma separator and created separate rows for each value. This transformation enables us to analyze the data at a more granular level.

Splitting and expanding the column

```
df_cast = df['cast'].str.split(',', expand=True).stack()
df_cast = df_cast.reset_index(level=1, drop=True).to_frame('cast')
df_cast['show_id'] = df['show_id']

df_country = df['country'].str.split(',', expand=True).stack()
df_country = df_country.reset_index(level=1, drop=True).to_frame('country')
df_country['show_id'] = df['show_id']

df_listed_in = df['listed_in'].str.split(',', expand=True).stack()
df_listed_in = df_listed_in.reset_index(level=1, drop=True).to_frame('listed_in')
df_listed_in['show_id'] = df['show_id']

df_director = df['director'].str.split(',', expand=True).stack()
df_director = df_director.reset_index(level=1, drop=True).to_frame('director')
df_director['show_id'] = df['show_id']
```

we have a clean and transformed dataset ready for further analysis.

3. Non-Graphical Analysis:

```
df.head()
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	month_added	month_name_added	year_added
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	No Data	United States	2021-09-25	2020	PG-13	90 min	Documentaries	As her father nears the end of his life, filmm...	9	September	2021
1	s2	TV Show	Blood & Water	No Data	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	2021-09-24	2021	TV-MA	2 Seasons	International TV Shows, TV Dramas, TV Mysteries	After crossing paths at a party, a Cape Town t...	9	September	2021
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	United States	2021-09-24	2021	TV-MA	1 Season	Crime TV Shows, International TV Shows, TV Act...	To protect his family from a powerful drug lor...	9	September	2021
3	s4	TV Show	Jailbirds New Orleans	No Data	No Data	United States	2021-09-24	2021	TV-MA	1 Season	Docuseries, Reality TV	Feuds, flirtations and toilet talk go down amo...	9	September	2021
4	s5	TV Show	Kota Factory	No Data	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	2021-09-24	2021	TV-MA	2 Seasons	International TV Shows, Romantic TV Shows, TV ...	In a city of coaching centers known to train I...	9	September	2021

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 8797 entries, 0 to 8806
Data columns (total 15 columns):
 #   Column             Non-Null Count  Dtype         
---  ------             --------------  -----         
 0   show_id            8797 non-null   object        
 1   type               8797 non-null   object        
 2   title              8797 non-null   object        
 3   director           8797 non-null   object        
 4   cast               8797 non-null   object        
 5   country            8797 non-null   object        
 6   date_added         8797 non-null   datetime64[ns]
 7   release_year       8797 non-null   int64         
 8   rating             8797 non-null   object        
 9   duration           8797 non-null   object        
 10  listed_in          8797 non-null   object        
 11  description        8797 non-null   object        
 12  month_added        8797 non-null   int64         
 13  month_name_added   8797 non-null   object        
 14  year_added         8797 non-null   int64         
dtypes: datetime64[ns](1), int64(3), object(11)
memory usage: 1.3+ MB
```

4: Exploratory Analysis and Visualization

Distribution of Content Types

Univariate analysis.

Calculate the percentage distribution of content types

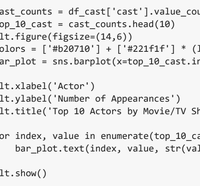
```
x = df.groupby(['type'])['type'].count()
y = len(df)
r= ((x/y)*100).round(2)
```

Create a DataFrame to store the percentage distribution

```
mf_ratio = pd.DataFrame(r)
mf_ratio.rename({'type': '%'}, axis=1, inplace=True)
```

Plot the 3D-effect pie chart

```
plt.figure(figsize=(4,4))
colors = ['#b20710', '#221f1f']
explode = (0.1,0)
plt.pie(mf_ratio['%'], labels=mf_ratio.index, autopct='%1.1f%%',
colors=colors, explode=explode, shadow=True, startangle=90,
textprops={'color': 'white'})
plt.legend(loc='upper right')
plt.title('Distribution of content Types')
plt.show()
```



Top 10 Countries Where Netflix is Popular

Remove white spaces from 'country' column

```
df_country['country'] = df_country['country'].str.rstrip()
```

Find value counts

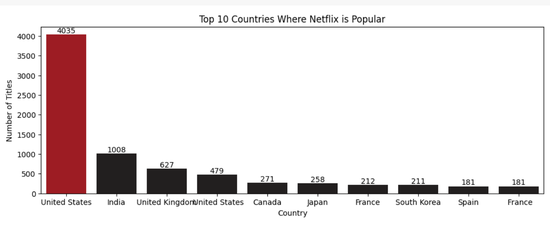
```
country_counts = df_country['country'].value_counts()
```

Select the top 10 countries

```
top_10_countries = country_counts.head(10)
```

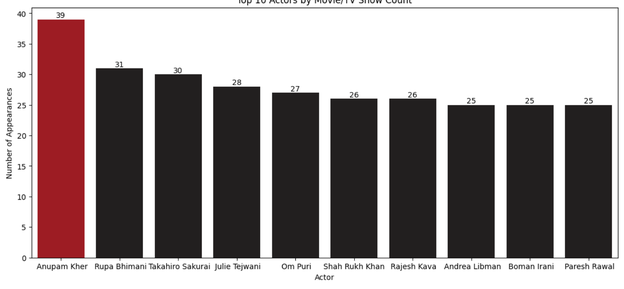
Plot the top 10 countries

```
plt.figure(figsize=(12, 4))
colors = ['#b20710'] + ['#221f1f'] * (len(top_10_countries) - 1)
bar_plot = sns.barplot(x=top_10_countries.index, y=top_10_countries.values, palette=colors)

plt.xlabel('Country')
plt.ylabel('Number of Titles')
plt.title('Top 10 Countries Where Netflix is Popular')

for index, value in enumerate(top_10_countries.values):
  bar_plot.text(index, value, str(value), ha='center', va='bottom')
plt.show()
```



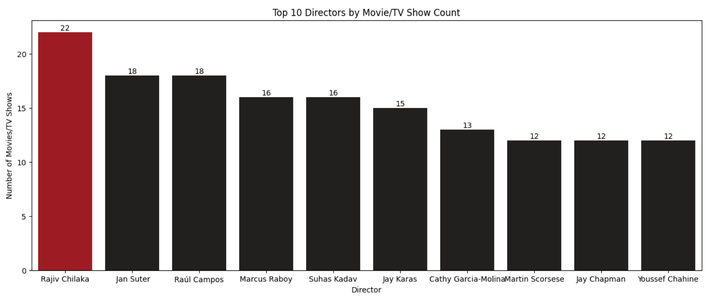
Top 10 Actors by Movie/TV Show Count

```
cast_counts = df_cast['cast'].value_counts()[1:]
top_10_cast = cast_counts.head(10)
plt.figure(figsize=(14,6))
colors = ['#b20710'] + ['#221f1f'] * (len(top_10_cast) - 1)
bar_plot = sns.barplot(x=top_10_cast.index, y=top_10_cast.values, palette=colors)

plt.xlabel('Actor')
plt.ylabel('Number of Appearances')
plt.title('Top 10 Actors by Movie/TV Show Count')

for index, value in enumerate(top_10_cast.values):
    bar_plot.text(index, value, str(value), ha='center', va='bottom')

plt.show()
```



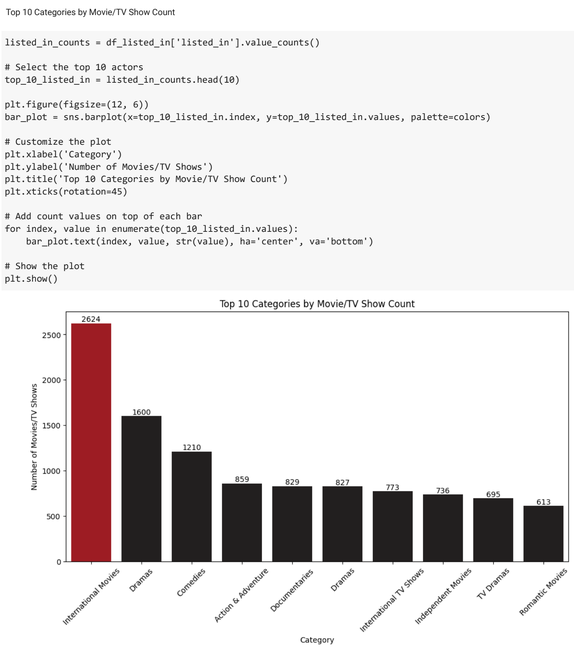
Top 10 Directors by Movie/TV Show Count

```
director_counts = df_director['director'].value_counts()[1:]
top_10_directors = director_counts.head(10)
plt.figure(figsize=(16, 6))
colors = ['#b20710'] + ['#221f1f'] * (len(top_10_directors) - 1)
bar_plot = sns.barplot(x=top_10_directors.index, y=top_10_directors.values, palette=colors)

plt.xlabel('Director')
plt.ylabel('Number of Movies/TV Shows')
plt.title('Top 10 Directors by Movie/TV Show Count')

for index, value in enumerate(top_10_directors.values):
    bar_plot.text(index, value, str(value), ha='center', va='bottom')

plt.show()
```



Top 10 Categories by Movie/TV Show Count

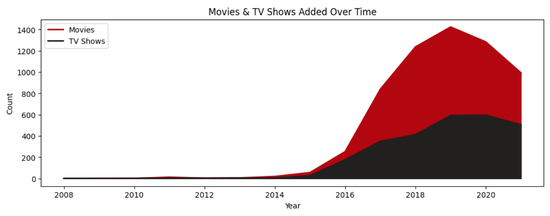
```
listed_in_counts = df_listed_in['listed_in'].value_counts()

# Select the top 10 actors
top_10_listed_in = listed_in_counts.head(10)

plt.figure(figsize=(12, 6))
bar_plot = sns.barplot(x=top_10_listed_in.index, y=top_10_listed_in.values, palette=colors)

# Customize the plot
plt.xlabel('Category')
plt.ylabel('Number of Movies/TV Shows')
plt.title('Top 10 Categories by Movie/TV Show Count')
plt.xticks(rotation=45)

# Add count values on top of each bar
for index, value in enumerate(top_10_listed_in.values):
    bar_plot.text(index, value, str(value), ha='center', va='bottom')

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



Movies & TV Shows Added Over Time

```
# Filter the DataFrame to include only Movies and TV Shows
df_movies = df[df['type'] == 'Movie']
df_tv_shows = df[df['type'] == 'TV Show']

# Group the data by year and count the number of Movies and TV Shows
# added in each year
movies_count = df_movies['year_added'].value_counts().sort_index()
tv_shows_count = df_tv_shows['year_added'].value_counts().sort_index()

# Create a line chart to visualize the trends over time
plt.figure(figsize=(12, 4))
plt.plot(movies_count.index, movies_count.values, color='#b20710',
label='Movies', linewidth=2)
plt.plot(tv_shows_count.index, tv_shows_count.values, color='#221f1f',
label='TV Shows', linewidth=2)

# Fill the area under the line charts
plt.fill_between(movies_count.index, movies_count.values, color='#b20710')
plt.fill_between(tv_shows_count.index, tv_shows_count.values, color='#221f1f')

# Customize the plot
plt.xlabel('Year')
plt.ylabel('Count')
plt.title('Movies & TV Shows Added Over Time')
plt.legend()
```

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

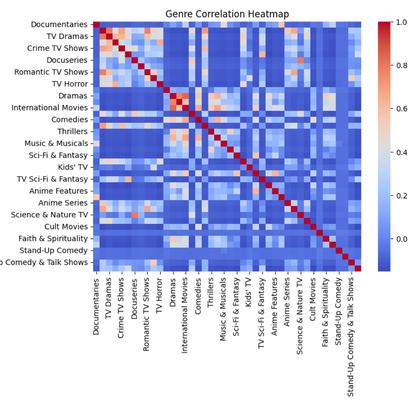


Netflix saw its real growth starting from the year 2015, & we can see it added more Movies than TV Shows over the years.

Genre Correlation Heatmap

Analyzing the correlation between genres can reveal interesting relationships between different types of content.

```
# Extracting unique genres from the 'listed_in' column
genres = df['listed_in'].str.split(', ', expand=True).stack().unique()

# Create a new DataFrame to store the genre data
genre_data = pd.DataFrame(index=genres, columns=genres, dtype=float)

# Fill the genre data DataFrame with zeros
genre_data.fillna(0, inplace=True)

# Iterate over each row in the original DataFrame and update the genre data DataFrame
for _, row in df.iterrows():
    listed_in = row['listed_in'].split(', ')
    for genre1 in listed_in:
        for genre2 in listed_in:
            genre_data.at[genre1, genre2] += 1

# Create a correlation matrix using the genre data
correlation_matrix = genre_data.corr()

# Create the heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=False, cmap='coolwarm')

# Customize the plot
plt.title('Genre Correlation Heatmap')
plt.xticks(rotation=90)
plt.yticks(rotation=0)

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



By analyzing the heatmap, we can identify strong positive correlations between specific genres, such as TV Dramas and International TV Shows, Romantic TV Shows, and International TV Shows.

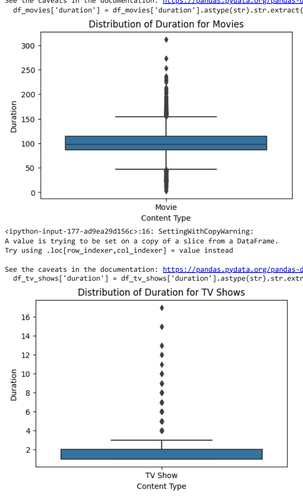
For categorical variable(s): Boxplot

Duration Distribution for Movies and TV Shows

Analyzing the duration distribution for movies and TV shows allows us to understand the typical length of content available on Netflix. We can create box plots to visualize these distributions and identify outliers or standard durations.

```
# Extracting and converting the duration for movies
df_movies = df[df.type.str.contains("Movie")]
df_movies['duration'] = df_movies['duration'].astype(str).str.extract('(\d+)', expand=False).astype(int)

# Creating a boxplot for movie duration
plt.figure(figsize=(6, 4))
sns.boxplot(data=df_movies, x='type', y='duration')
plt.xlabel('Content Type')
plt.ylabel('Duration')
plt.title('Distribution of Duration for Movies')
plt.show()


# Extracting and converting the duration for TV shows
df_tv_shows = df[df.type.str.contains("TV Show")]
df_tv_shows['duration'] = df_tv_shows['duration'].astype(str).str.extract('(\d+)', expand=False).astype(int)

# Creating a boxplot for TV show duration
plt.figure(figsize=(6, 4))
sns.boxplot(data=df_tv_shows, x='type', y='duration')
plt.xlabel('Content Type')
plt.ylabel('Duration')
plt.title('Distribution of Duration for TV Shows')
plt.show()
```

```
<ipython-input-177-ad9ea29d156c>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_movies['duration'] = df_movies['duration'].astype(str).str.extract('(\d+)', expand=False).astype(int)
```




```
<ipython-input-177-ad9ea29d156c>:16: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_tv_shows['duration'] = df_tv_shows['duration'].astype(str).str.extract('(\d+)', expand=False).astype(int)
```


Analyzing the movie box plot, we can see that most movies fall within a reasonable duration range, with few outliers exceeding approximately 2.5 hours. This suggests that most movies on Netflix are designed to fit within a standard viewing time.

For TV shows, the box plot reveals that most shows have one to four seasons, with very few outliers having longer durations. This aligns with the earlier trends, indicating that Netflix focuses on shorter series formats.

Business Insights :

> Quantity: Our analysis revealed that Netflix had added more movies than TV shows, aligning with the expectation that movies dominate their content library.

> Genre Correlation: Strong positive associations were observed between various genres, such as TV dramas and international TV shows, romantic and international TV shows, and independent movies and dramas. These correlations provide insights into viewer preferences and content interconnections.

> Content Addition: July emerged as the month when Netflix adds the most content, closely followed by December, indicating a strategic approach to content release.

> TV Show Episodes: Most TV shows on Netflix have one season, suggesting a preference for shorter series among viewers.

RECOMMENDATIONS

> Netflix has to focus on TV Shows also because there are people who will like to see tv shows rather than movies

> By approaching the top director we can plan some more movies/tv shows in order to increase the popularity

> We have seen most no of international movies genre so need to give priority to other geners like hooro,comedy..etc

> getting subscription is usefull as netflix is releasing more movies per year

> Mainly the release in ott should focus on the festival holidays, year end and week ends which is to be mainly focussed

> Should focus on a actor who has immense following and make use of it by doing a TV Shows or web series